



Event Propagation Conditions in Circuit Delay Computation

HAKAN YALCIN and JOHN P. HAYES

University of Michigan

Accurate and efficient computation of delays is a central problem in computer-aided design of complex VLSI circuits. Delays are determined by events (signal transitions) propagated from the inputs of a circuit to its outputs, so precise characterization of event propagation is required for accurate delay computation. Although many different propagation conditions (PCs) have been proposed for delay computation, their properties and relationships have been far from clear. We present a systematic analysis of delay computation based on a series of waveform models that capture signal behavior rigorously at different levels of detail. The most general model, called the exact or W0 model, specifies each event occurring in a circuit signal. A novel method is presented that generates approximate waveforms by progressively eliminating signal values from the exact model. For each waveform model, we derive the PCs that correctly capture the requirements under which an event propagates along a path. The waveform models and their PCs are shown to form a well-defined hierarchy, which provides a means to trade accuracy for computational effort. The relationships among the derived PCs and existing ones are analyzed in depth. It is proven that though many PCs, such as the popular floating mode condition, produce a correct upper bound on the circuit delay, they can fail to recognize event propagation in some instances. This analysis further enables us to derive new and useful PCs. We describe such a PC, called safe static. Experimental results demonstrate that safe static provides an excellent accuracy/efficiency tradeoff.

Categories and Subject Descriptors: B.7.2 [**Integrated Circuits**]: Design Aids—*verification*; B.6.3 [**Logic Design**]: Design Aids—*verification*

General Terms: Performance, Theory, Verification

Additional Key Words and Phrases: Delay computation, event propagation, false path, path sensitization, propagation condition, timing analysis, waveform modeling

1. INTRODUCTION

Accurate computation of delay is a key issue in the design of high-performance VLSI circuits. The complexity of this problem is highly depen-

Authors' addresses: H. Yalcin, Advanced Computer Architecture Laboratory, Department, Electrical Engineering and Computer Science, University of Michigan, Ann Arbor, Michigan 48109-2122; email: {hakan, jhayes}@eecs.umich.edu; J. P. Hayes, Advanced Computer Architecture Laboratory, Department of Electrical Engineering and Computer Science, University of Michigan, Ann Arbor, Michigan 48109-2122; email: {hakan, jhayes}@eecs.umich.edu.

Permission to make digital/hard copy of part or all of this work for personal or classroom use is granted without fee provided that the copies are not made or distributed for profit or commercial advantage, the copyright notice, the title of the publication, and its date appear, and notice is given that copying is by permission of the ACM, Inc. To copy otherwise, to republish, to post on servers, or to redistribute to lists, requires prior specific permission and/or a fee.

© 1997 ACM 1084-4309/97/0700-0249 \$3.50

dent on the assumed waveform model; that is, on the level of detail at which signal behavior is modeled. More detailed waveform models capture the actual signal behavior more accurately, but analysis using such models can be prohibitively slow, so in practice, simplified models are often used. A popular model is floating mode [Chen and Du 1993], where only the latest event (signal transition) on every circuit node, namely the node becoming stable, is considered.

Many kinds of sensitization criteria or propagation conditions (PCs) have been developed for delay computation, often in an ad hoc manner. Examples are static sensitization [Benkoski et al. 1987], the Brand-Iyengar condition [Brand and Iyengar 1986] and the floating mode condition [Chen and Du 1993]. In most cases, the analysis is algorithm-driven, and does not reflect how signal events actually propagate. When improperly used these models can lead to serious errors; for example, it is well known that static sensitization can underestimate the circuit delay [Chen and Du 1993; McGeer and Brayton 1989]. Furthermore, as we prove here, more accurate PCs such as floating mode sometimes fail to recognize whether an event can propagate along a given path. Despite several attempts [Chen and Du 1993; McGeer and Brayton 1991; Silva and Sakallah 1993], a unified theory of PCs has remained elusive, and the relationships among even the most widely studied PCs are far from clear.

In this paper we derive propagation conditions in a systematic way, starting from a general waveform model. As explained in Section 2, we distinguish the conditions intended for delay computation from those that deal with actual event propagation. With this distinction in mind, we develop a series of waveform models in Sections 3–5, based on how closely they match actual signal behavior, and show that they form a well-defined hierarchy. For each waveform model, we derive the PCs based on fundamental cause-and-effect behavior using a 2-input AND gate as a representative example. Extensions to other basic gates and non-zero delays are described in Section 5. This analysis reveals the fundamental relationships among all known PCs, as discussed in Section 6. It further enables us to derive new and potentially useful PCs, examples of which are presented in Section 7. Finally, we give experimental results to evaluate the accuracy of the proposed PCs.

2. BASIC CONCEPTS

A combinational logic circuit is composed of modules (gates, multiplexers, decoders, etc.) which are assumed to have known internal delays. The modules are linked by interconnections which, along with the circuit's primary input-output terminals, define the circuit's signal nodes. We assume that modules are limited to basic gates. To simplify the analysis, we also ignore interconnect delays; however, they can be easily handled using the techniques described here for gates.

Definition 1. Given an input stimulus to the circuit, a change or transition in the value of a circuit node is called an *event*. In addition to

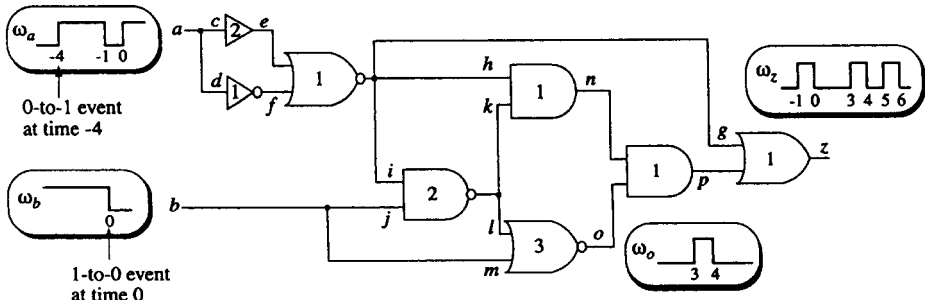


Fig. 1. Event and waveform examples.

transitions between logic 0 and 1, we consider transitions between an unknown value (U) and 0 or 1 as events. The time at which an event occurs is called the *event time*.

Events at the primary inputs propagate through the circuit, are delayed by the gates, and eventually reach the primary outputs. Events can interact in complex ways. Depending on circuit structure, they can take different paths, and hence can experience different delays. Some events are filtered out because they are blocked by other events.

Definition 2. The entire sequence of events occurring at a circuit node x over the time period of interest is called the *waveform* of x and is represented by ω_x .

Figure 1 illustrates several waveforms and events in a small circuit. The first event time of a node is when the node starts to change, referred to as *destabilization*. Likewise, the last event time is when the value of the node settles to its final value, referred to as *stabilization*.

Consider a set of input waveforms W_k applied to a circuit such that the earliest and latest event times at primary inputs are $t_{PI}^m(k)$ and $t_{PI}^M(k)$, respectively. Let $t_{PO}^m(k)$ be the earliest (destabilizing), and $t_{PO}^M(k)$ be the latest (stabilizing) event time over all primary output events occurring as a result of W_k . Let Ω represent the set of all input waveforms under consideration.

Definition 3. The *longest delay* of the circuit corresponding to W_k is $\tau^M(k) = t_{PO}^M(k) - t_{PI}^M(k)$,¹ and the *shortest delay* is $\tau^m(k) = t_{PO}^m(k) - t_{PI}^m(k)$. The longest delay τ^{\max} of the circuit over all input waveforms is $\tau^{\max} = \max_{W_k \in \Omega} \tau^M(k)$, and the shortest delay τ^{\min} is $\tau^{\min} = \min_{W_k \in \Omega} \tau^m(k)$.

¹Implicit here is the assumption that the events $t_{PI}^M(k)$ and $t_{PO}^M(k)$ are causally related, that is, the input event at $t_{PI}^M(k)$ triggers the output event at $t_{PO}^M(k)$, and are chosen such that $t_{PO}^M(k) - t_{PI}^M(k)$ is the maximum over all such input and output events. The case of shortest delay is similar.

For the circuit of Figure 1, assuming Ω consists of input waveforms that can have an arbitrary number of events, the longest and shortest delays are, respectively, $\tau^{\max} = 6$ and $\tau^{\min} = 3$. The case of the shortest delay is symmetrical to that of the longest delay, so we only consider the latter and its computation in this paper. Unless otherwise noted, we assume that the latest input event time $t_{PI}^M(k)$ is 0 for every W_k . Thus, $\tau^M(k) = t_{PO}^M(k)$. In the rest of the paper, the terms “last event time” and “delay” are used interchangeably. In addition, the term “circuit delay” is used to mean the longest delay, and is denoted by τ .

Because the circuit delays are determined by events that propagate to the outputs, it is important that event propagation be accurately characterized.

Definition 4. If an event appears on output line z of a gate in response to an event j on input line x , then event j is said to *propagate* to z . The logical condition under which this occurs is called the *propagation condition* (PC) for event j and is represented by Ψ_{xz-j} .

Definition 4 can be applied to paths, in which case the condition under which a specific event j travels down a path P is the conjunction of the conditions for the event to propagate through every gate in path P , and is represented by Ψ_{P-j} . For example, if P is the path $a-c-e-g-x$ in the circuit of Figure 1, then $\Psi_{P-j} = \Psi_{ce-j} \Psi_{eg-j} \Psi_{gz-j}$, where Ψ_{xz-j} is the PC for an event j going from input x of a gate to output z . The longest delay τ^{\max} through the circuit can be rewritten in terms of the PCs, as follows.

$$\tau^{\max} = \max_{W_k \in \Omega} \{ (t_{PI}^j(k) + d(P)) \cdot \Psi_{P-j}(k) \} \quad (1)$$

The max operator is applied over all input events $j \in W_k$ occurring at time $t_{PI}^j(k)$, and all paths P along which event j can possibly propagate. In this expression, $d(P)$ is the length of path P , and the AND-like blocking operator “ \cdot ” is defined as

$$t \cdot \Psi = \begin{cases} -\infty & \text{if } \Psi = 0 \\ t & \text{if } \Psi = 1 \end{cases}$$

Equation (1) captures the fact that the circuit delay is determined by the input event that is the latest to arrive at any output. If no event propagates, the computed delay is $-\infty$, indicating that no event occurs on the output. Let Δ^{\max} and δ^{\min} represent, respectively, the longest and shortest topological delays of the circuit, so $\delta^{\min} \leq d(P) \leq \Delta^{\max}$. Since $t_{PI}^j(k) \leq 0$, it follows that $\tau^{\max} \leq \Delta^{\max}$. Similarly, it can be shown that $\tau^{\min} \geq \delta^{\min}$.

The set Ω of all input waveforms has a strong impact on determining whether an event can propagate along a path. For the circuit of Figure 1,

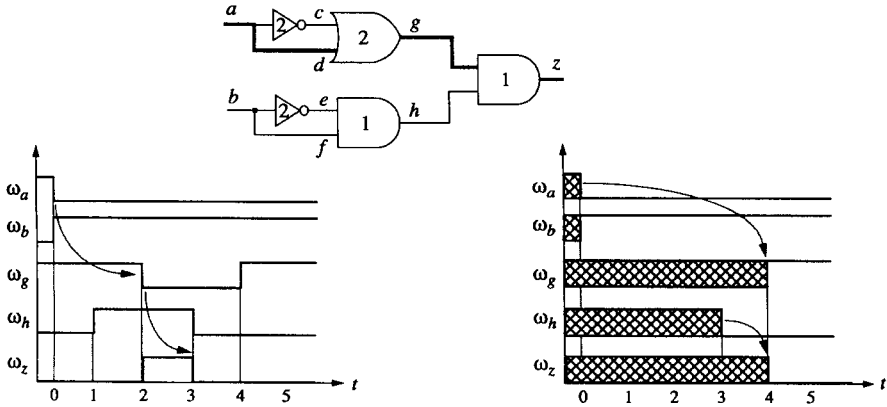


Fig. 2. Example circuit where an event propagates along a path identified as false by the floating mode and viability conditions.

when the input waveforms in Ω are allowed to have any number of events, an event can propagate along the topologically longest path *a-c-e-i-l-o-p-z*, as revealed by the waveforms in the figure. If Ω is restricted to those input waveforms with a maximum of one transition, called the *transition mode* of operation [Devadas et al. 1992], this is no longer the case. Furthermore, the longest circuit delay τ^{max} reduces to 4. Clearly, propagation as well as circuit delays are defined with respect to Ω .

A common concept found in the research literature is path sensitization, where a path along which an event can propagate is referred to as *sensitized*. A path that cannot be sensitized under any input waveform is called *unsensitizable* or *false*. However, we note that this definition of sensitization is not precise in that it does not explicitly specify the event(s) whose propagation is of concern. Most approaches implicitly consider only the last event on each circuit node. This, for example, is the case for floating mode [Chen and Du 1993] and viability [McGeer and Brayton 1989]. These PCs assume that circuit nodes have unknown values until they stabilize. Thus, even though some events previous to the last one may propagate along a path, the path can still be declared to be unsensitizable (false). This phenomenon is illustrated in Figure 2. As shown in the timing diagram on the left, an event on *a* propagates along the highlighted path *a-d-g-z*, and reaches the output *z* as the latest event. According to both the floating mode and viability conditions, however, the path is false because the last event on node *g* does not propagate. One might also interpret path sensitization as follows: If there exists an event that propagates along a path, then the path is sensitized. By this definition, the highlighted path in Figure 2 is sensitized, but the computational effort to identify such paths is questionable.

As we show in this paper, it is not necessary to identify paths as sensitizable or false to perform delay computation. Instead we focus on the more precise notion of event propagation, and derive the conditions under

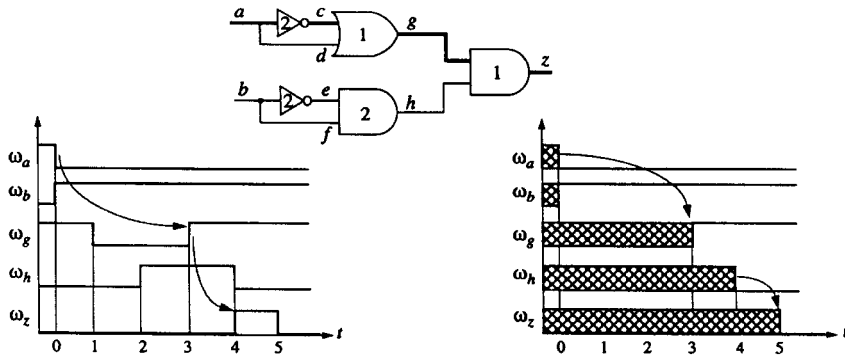


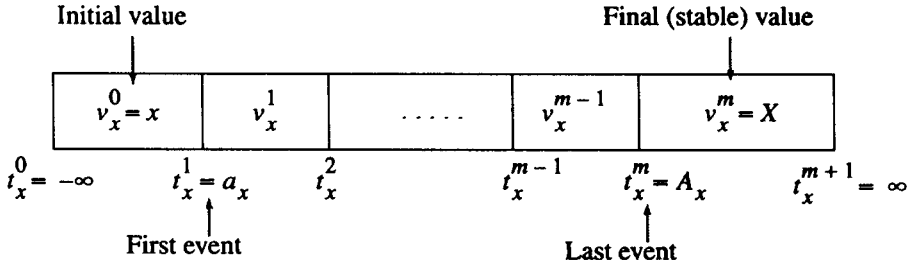
Fig. 3. Example of failure by floating mode to recognize the propagation of an event.

which an event can propagate. Our analysis of event propagation in the following sections reveals that most existing PCs defined for path sensitization are too restrictive in that they can decide that an event does not propagate even when it actually does. The circuit shown in Figure 3, which is similar to that of Figure 2, illustrates this point for the floating mode PC. Even though the last event on node g propagates to the output z , the floating mode condition decides otherwise. The reason is that the event in question on g does not determine the last event at the output z . On the other hand, the floating mode condition does calculate a correct upper bound on the circuit delay.

A number of conditions have been proposed in the literature which, like floating mode, calculate a correct upper bound on the circuit delay, but are not necessarily correct for event propagation. As will be shown later, they include the loose condition [Chen and Du 1993] and static cosensitization [Devadas et al. 1993]. These conditions typically assume that an event propagates from an input of a gate to its output only if it is guaranteed to be the last event at the output. We refer to all such conditions as *delay propagation conditions* (DPCs), while those that reflect actual event propagation are called *event propagation conditions* (EPCs). We denote an EPC by the symbol Ψ , and a DPC by φ . When referring either to EPCs or DPCs, we use the term PC and denote it by λ .

3. WAVEFORM MODELING

We call a logic-level behavior that for each signal x in a circuit specifies every event in a given time interval (t_x^0, t_x^{m+1}) the *W0 waveform model* or the *exact mode*. As depicted in Figure 4, a W0 waveform ω_x contains $m \geq 0$ events, each occurring at time t_x^i , $1 \leq i \leq m$. Without loss of generality, we assume that $t_x^0 = -\infty$ and $t_x^{m+1} = \infty$. For notational convenience, the first and last event times of x are also referred to as a_x and A_x , respectively [Shriver and Sakallah 1992; Silva and Sakallah 1993]. The value assumed by x in the interval (t_x^i, t_x^{i+1}) is represented by v_x^i , 0

Fig. 4. Waveform for a node x under the $W0$ model.

$\leq i \leq m$, with the initial and final values also referred to as x and X , respectively. We require that $v_x^i \in \{0, 1, U\}$. At each transition point t_x^i , we assume that the value of the signal is U . The waveform ω_x is thus fully characterized by a sequence of time-value pairs, as follows.

$$\omega_x^{W0} = ((t_x^0, v_x^0), (t_x^1, v_x^1), \dots, (t_x^m, v_x^m)) \quad (2)$$

Consider a 2-input AND gate with inputs x, y , output z , and zero delay. Let the input waveforms of x and y be, respectively, $\omega_x^{W0} = ((t_x^0, v_x^0), \dots, (t_x^m, v_x^m))$ and $\omega_y^{W0} = ((t_y^0, v_y^0), \dots, (t_y^n, v_y^n))$; thus there are m events on input x , and n on input y . The EPC Ψ for event propagation and the DPC φ for delay computation are derived below. Let λ_x^j be the PC (either EPC or DPC) for an event occurring at time t_x^j on input x . Similarly, let γ_y^k be the PC for an event occurring at t_y^k on input y . With the input waveforms represented symbolically, Equation (1) can be rewritten to obtain the last event time A_z at the output z , as follows:

$$A_z = \max(t_x^1 \cdot \lambda_x^1, \dots, t_x^m \cdot \lambda_x^m, t_y^1 \cdot \gamma_y^1, \dots, t_y^n \cdot \gamma_y^n) \quad (3)$$

This formula is proven below for the $W0$ model and will be used later for all waveform models.

Let Ψ_{xz-j}^{W0} denote the EPC that the event j on input x occurring at time t_x^j propagates to the output z of a 2-input AND gate. In order for this event to propagate, the side input y must have a non-controlling value (1 in the AND case) or unknown value U at the time j appears on input x . The U value is required for correctness or “safeness,” which will be discussed later.

$$\Psi_{xz-j}^{W0} = \sum_{i=0}^n (t_y^i \leq t_x^j \leq t_y^{i+1}) (v_y^i \neq 0) \quad (4)$$

where juxtaposition denotes logical AND, Σ denotes logical OR, and $t_y^{n+1} = \infty$. The summed expression in (4) states that the value v_y^i assumed by

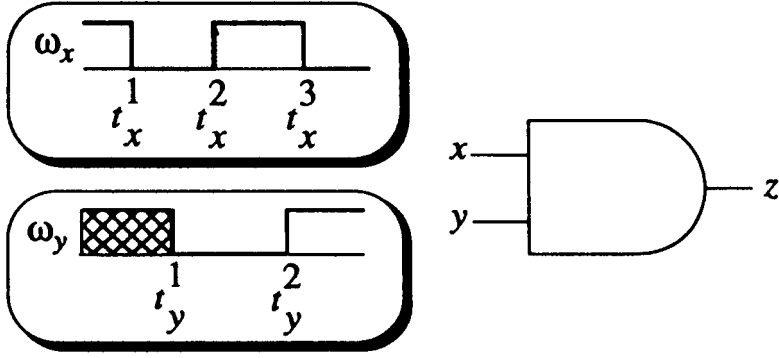


Fig. 5. Event propagation through a 2-input AND gate.

input y in the interval (t_y^i, t_y^{i+1}) coinciding with t_x^j should be non-zero, i.e., non-controlling or U . For the example in Figure 5, the EPC Ψ_{xz-2}^{W0} for the event occurring at time t_x^2 is $(t_x^2 \geq t_y^2) + (t_x^2 \leq t_y^1)$. We note that the EPC of (4) allows the propagation of pulses of any width, which is a worst-case assumption. However, by adding a condition of the form $(t_x^j - t_x^{j-1} > \epsilon)$, pulses of width less than ϵ can be filtered out.

THEOREM 1. Consider a 2-input AND gate with inputs x, y , output z , and zero delay. Let Ψ_{xz-j}^{W0} and Ψ_{yz-k}^{W0} represent, respectively, the EPCs for an event j on x and event k on y to propagate to the output z , where $1 \leq j \leq m$ and $1 \leq k \leq n$. The last event time A_z^{W0} at output z is given by

$$A_z^{W0} = \max(t_x^1 \cdot \Psi_{xz-1}^{W0}, \dots, t_x^m \cdot \Psi_{xz-m}^{W0}, t_y^1 \cdot \Psi_{yz-1}^{W0}, \dots, t_y^n \cdot \Psi_{yz-n}^{W0}) \quad (5)$$

(A proof is given in the Appendix.) As stated in Section 2, DPCs, which are used for delay calculation only, assume that an event is blocked unless it is the *latest* propagating one. Consider an event j on input x occurring at time t_x^j . Strictly speaking, the DPC ϕ_{xz-j}^{W0} for this event must ensure that the event reaches the output no later than any event on both inputs x and y . However, it is sufficient to require that event j propagate no later than any event on input y only, since the application of the max operator in (3) will correctly identify the last propagating event. Let $\mathbf{B}(j, k)$ be the blocking condition for event k and all subsequent events on y that occur after t_x^j .

$$\mathbf{B}(j, k) = \prod_{r=k}^n \overline{\Psi}_{yz-r}^{W0} = \prod_{r=k}^n \sum_{s=j}^m (t_x^s < t_y^r < t_x^{s+1}) (v_x^s = 0)$$

where Π denotes logical AND. The two limiting cases are $\mathbf{B}(j, n + 1) = 1$ and $\mathbf{B}(m, k) = (v_x^m = 0)$, $k \leq n$. Thus, the DPC φ_{xz-1}^{W0} can be expressed as

$$\varphi_{xz-j}^{W0} = \Psi_{xz-j}^{W0} \mathbf{B}(j, k) \quad (6)$$

which can be rewritten as

$$\varphi_{xz-j}^{W0} = \sum_{i=0}^n \mathbf{B}(j, i + 1)(t_x^i \leq t_x^j \leq t_y^{i+1})(v_y^i \neq 0) \quad (7)$$

As an example, the DPC φ_{xz-2}^{W0} for the event occurring at t_x^2 in Figure 5 is $(t_x^2 \geq t_y^2) + (t_x^2 \leq t_y^1)(t_x^1 > t_x^3)$. In this case the condition $B(2,1)$ for the two events on y to be blocked after t_x^2 is $(t_y^1 > t_x^3)$.

THEOREM 2. *Consider again a 2-input AND gate with inputs x, y output z , and zero delay. Let φ_{xz-j}^{W0} and φ_{yz-k}^{W0} represent, respectively, the DPCs for an event j on x and event k on y to propagate to the output z , where $1 \leq j \leq m$ and $1 \leq k \leq n$. The last event time A_z^{W0} at output z can be calculated by*

$$A_z^{W0} = \max(t_x^1 \cdot \varphi_{xz-1}^{W0}, \dots, t_x^m \cdot \varphi_{xz-m}^{W0}, t_y^1 \cdot \varphi_{yz-1}^{W0}, \dots, t_y^n \cdot \varphi_{yz-n}^{W0}) \quad (8)$$

(The proof is similar to that of Theorem 1.) Thus, since (5) and (8) produce the same last event time, either EPC Ψ_{xz}^{W0} or DPC φ_{xz}^{W0} can be used for delay computation. From (6), we have

$$\varphi_{xz}^{W0} \subseteq \Psi_{xz}^{W0} \quad (9)$$

So the DPC φ_{xz}^{W0} cannot be used to determine which events propagate.

As the above analysis illustrates, timing analysis using the W0 waveform model is potentially complex. First, since an arbitrary number of events can occur on circuit nodes, the storage of these events is a problem. Devadas et al. [1994] give an example where the number of events in a circuit is exponential in circuit size. Second, PC calculation becomes difficult due to the potentially high number of conditions relating event times, as revealed by Equations (4) and (7). Thus, simulation-like methods such as that of Devadas et al. [1992; 1994] must often be employed in practice. Complex delay models such as the min-max delay model further complicate the analysis.

Several approximation methods can be used to simplify timing analysis:

- Restrict the waveform model so that a subset of all possible events is considered. For example, floating mode considers only the last event on every circuit node.

- Restrict the delay model. For example, if a gate has many input-output path delays, one can consider only the maximum delay.
- Simplify the PC calculation. An example is the “conditionless” case where all events are assumed to propagate; this is classical topological delay analysis.

The above approximation methods are not independent, however. For instance, if the waveform model is restricted, PC calculation will be restricted as well.

Any approximate method M essentially replaces Ψ^{W0} by some condition Ψ^M in order to calculate circuit delays more efficiently. The following definitions are central to our analysis.

Definition 5. The approximate method M is *correct for event propagation* if and only if $\Psi^M \supseteq \Psi^{W0}$.

Definition 6. Let the last event time (or delay) under the $W0$ waveform model (actual last event time) be τ^{W0} , and the estimate of the approximate method be τ^M . The approximate method is *safe* or *correct for delay computation* if and only if $\tau^M \geq \tau^{W0}$.

THEOREM 3. *If the approximate method M is correct for event propagation, it is correct for delay computation also; that is, it is safe.*

PROOF. If the condition $\Psi^M \supseteq \Psi^{W0}$ in Definition 5 is met, and if an event propagates along a path under the $W0$ waveform model, then it propagates under M as well. Since this is also true of the latest event that determines the circuit delay, the estimate τ^M of M is at least as great as τ^{W0} . Additionally, there can exist events that do not propagate under $W0$ but do so under M , which only leads to delay overestimation. \square

Our approach to simplifying PCs (including both EPCs and DPCs) is to restrict the waveform model; that is, we aim to obtain simpler PCs by approximating the $W0$ waveform model in a systematic fashion. This will be done so that the resulting PCs never violate the condition $\tau^M \geq \tau^{W0}$ in Definition 6; we refer to this as the *safeness requirement*. In the process of deriving approximate waveform models and their associated PCs, we make use of a “smoothing” operator [McGeer and Brayton 1989], which is a special case of existential quantification.

Definition 7. Let f be a function of variables x_1, x_2, \dots, x_n . The *smoothing operator* $S_{x_i} f$ is defined as

$$S_{x_i} f = f_{x_i} + f_{\bar{x}_i}$$

where $f_{x_i} = f(x_i = 1)$ and $f_{\bar{x}_i} = f(x_i = 0)$ are the cofactors of f . The smoothing operator can be directly extended to multiple variables. Let $T = \{x_{i_1}, x_{i_2}, \dots, x_{i_k}\}$ be a subset of x_1, x_2, \dots, x_n . Then $S_{x_{i_1}} S_{x_{i_2}} \dots S_{x_{i_k}} f$

$= S_{x_{i_1} \dots x_{i_k}} f = S_T f$. The order in which the operator is applied to the variables of T is not important, since $S_{x_i} S_{x_j} f = S_{x_j} S_{x_i} f$.

In the context of delay calculation, the smoothing operator captures pessimism in the following way: An event will be assumed to propagate if the corresponding PC evaluates true for at least one combination of the variables in T . We make use of the following property from McGeer and Brayton [1991] to relate different PCs:

$$S_T f \supseteq f \quad (10)$$

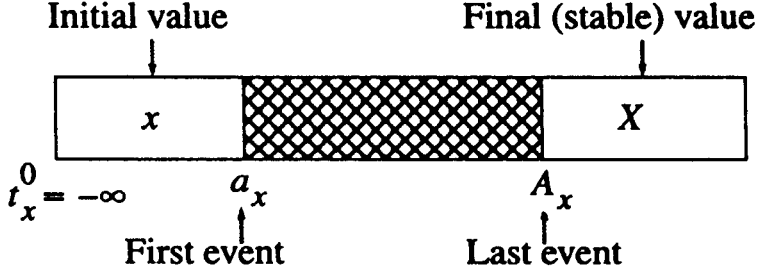
We also generalize the smoothing operator so that the dependence of a function on event times, in addition to Boolean variables, can be smoothed out as well. Consider the Boolean function (predicate) $f = (t_x - t_y \geq 0)$. Suppose we want to eliminate the dependence of f on $t_x - t_y$, which can be positive or negative. When it is positive, f will evaluate to 1. So, by the definition of smoothing, $S_{(t_x - t_y)} f = 1$. In general, consider a (Boolean) function $f(X, t)$, where X is a Boolean input vector and t is an event time, or a function of event times. Suppose that t takes on values from a (closed) interval $[t_l, t_u]$, where t_u is the upper bound and t_l is the lower bound. The *smoothing of f with respect to t* is defined by

$$S_t f = \sum_{t \in [t_l, t_u]} f(X, t) \quad (11)$$

4. APPROXIMATE WAVEFORM MODELS

We obtain the approximate waveform models by smoothing the values of some intervals from the W0 waveform model characterized by (2). The value v_x^i in an interval (t_x^i, t_x^{i+1}) is smoothed out and replaced by U . This operation reduces the number of variables making up ω_x by one. In this fashion, we simplify the W0 waveform model progressively, each time eliminating a number of interval values. The effect on event propagation is to increase the degree of pessimism because in order to comply with Definition 5, we assume that any event whose propagation depends on an interval with value U propagates. This assumption can lead to an overestimation of the circuit delay since some events may actually be blocked by certain signal values; but this is the price paid for simplification.

We examine three approximate waveform models, W1, W2, and W3 in depth. Those cases make the most basic simplifications possible, and as we demonstrate, correspond to, and illuminate, a variety of PCs discussed in the literature. For each model W_i , we derive the EPC Ψ^{W_i} for event propagation, and DPC φ^{W_i} for delay computation. The last event time, which is identical for both cases, is represented by A^{W_i} . We use Karnaugh maps to represent PCs because, as noted by Brown [1990], they are an efficient way to display the cofactors of a Boolean function, and are

Fig. 6. Waveform for a node x under the W1 model.

convenient here for comparing different PCs and last event times. Again, without loss of generality, we consider a 2-input AND gate computing $z = xy$.

W1 waveform model. Our first approximation to the W0 model of timing is to restrict signal waveforms to their first and last events. The remaining events occurring between these two are ignored, and the values v_x^1, \dots, v_x^{m-1} are smoothed out. We call this waveform model W1 or the *first-and-last-event* (FALE) model. This model, which is depicted in Figure 6, is also adopted in Cerny and Zejda [1994] and Shriver and Sakallah [1992]. A waveform ω_x is characterized as

$$\omega_x^{W1} = S_{\{v_x^1, \dots, v_x^{m-1}\}} \omega_x^{W0} = ((t_x^0, x), (a_x, U), (A_x, X)) \quad (12)$$

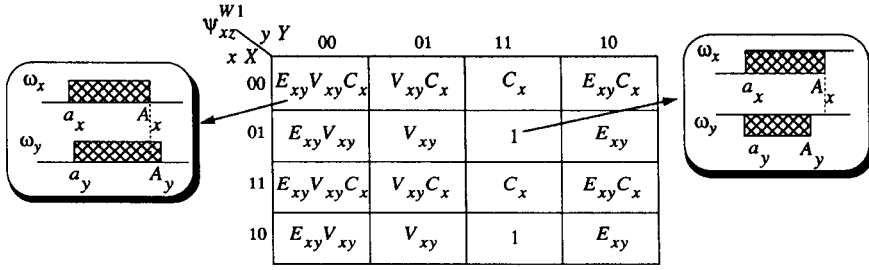
If ω_x contains no events, i.e., it is constant, we assume that $a_x = \infty$ and $A_x = -\infty$. We define a change predicate $C_x = (a_x \leq A_x)$ to indicate the stability of signal x : $C_x = 0$ if x is constant, $C_x = 1$ otherwise.

Calculating PCs for the W1 model is far simpler than for the W0 model, as there are only two events in each waveform. For the W1 model, assuming the output is not constant, the last output event is either the last event on input x or the one on y , whichever propagates—this is, in general, not true for the W0 model. Hence, it is sufficient to consider the propagation of the last event.

Let Ψ_{xz}^{W1} be the EPC for the last event on input x to reach the output z under the W1 waveform model. To derive Ψ_{xz}^{W1} , we first substitute $m = j$ in (4), then apply the appropriate smoothing operator:

$$\begin{aligned} \Psi_{xz}^{W1} &= C_x S_{\{v_x^1, \dots, v_x^{m-1}, v_y^1, \dots, v_y^{m-1}\}} \Psi_{xz}^{W0} = C_x [y(A_x \leq a_y) + Y(A_x \geq A_y) + (A_x \\ &\geq a_y)(A_x \leq A_y)] \quad (13) \end{aligned}$$

The predicate C_x is included to exclude constant signals, as Equation (4) applies to the j th event assuming it is present in ω_x . To simplify (13), we introduce three predicates to relate the event times of x and y :

Fig. 7. The EPC Ψ_{xz}^{W1} of a 2-input AND gate.

- (1) $E_{xy} = (A_x \leq A_y)$; that is, x stabilizes earlier than y . This is the *early* predicate.
- (2) $L_{xy} = (A_x \geq A_y)$; that is, x stabilizes later than y . This is the *late* predicate.
- (3) $V_{xy} = (A_x \geq A_y)$; that is, x stabilizes after y starts to destabilize.

Figure 7 shows Ψ_{xz}^{W1} in the form of a K-map for all combinations of x , X , y , and Y . Notice that when x and X differ, C_x is necessarily 1. The entries in Figure 7 can be inspected to verify the correctness of Ψ_{xz}^{W1} . Input waveforms corresponding to two entries are shown in the figure. For the input combination $(x, X, y, Y) = (0, 0, 0, 0)$, $\Psi_{xz}^{W1} = E_{xy}V_{xy}C_x$; that is, the two intervals must overlap, input x must not be a constant 0, and x must stabilize earlier than y . For the input combination $(x, X, y, Y) = (0, 1, 1, 1)$, Ψ_{xz}^{W1} is 1; that is, the last event on input x always propagates, since the value of the side input y is either non-controlling (1) or U .

The EPC Ψ_{yz}^{W1} for the last event on input y is exactly symmetrical with Ψ_{xz}^{W1} . The last event time A_z^{W1} for the output z can be obtained via Equation (3) or via $A_z^{W1} = S_{\{v_x^1, \dots, v_x^{m-1}, v_y^1, \dots, v_y^{m-1}\}} A_z^{W0}$, as in (13). Figure 8 shows A_z^{W1} in the form of a K-map.

To derive the DPC φ_{xz}^{W1} , we first substitute $m = j$ in (7), then apply the smoothing operation:

$$\varphi_{xz}^{W1} = C_x S_{\{v_x^1, \dots, v_x^{m-1}, v_y^1, \dots, v_y^{m-1}\}} \varphi_{xz}^{W0} = C_x [y \bar{X} (A_x \leq a_y) + Y (A_x \geq A_y) + \bar{X} (A_x \geq a_y) (A_x \leq A_y)] \quad (14)$$

Again, the predicate C_x is included to handle constant signals. The DPC φ_{xz}^{W1} is shown in Figure 9. As in (9), $\varphi_{xz}^{W1} \subseteq \Psi_{xz}^{W1}$ since $L_{xy} \subseteq V_{xy}$. Note that $\Psi_{xz}^{W0} \subseteq \varphi_{xz}^{W1}$ does not hold. For example, consider the input combination $(x, X, y, Y) = (0, 1, 0, 0)$. Although the last event on x can propagate when $A_x \leq A_y$, the DPC equals the constant 0. Therefore, by Definition 6, the DPC φ_{xz}^{W1} is incorrect for event propagation. However, it is correct for

A_z^{W1} $x \backslash y$					
		00	01	11	10
00	X	$V_{xy}V_{yx}C_xC_y \bullet \min(A_x, A_y)$	$V_{xy}C_x \bullet A_x$	$C_x \bullet A_x$	$V_{yx}C_x \bullet \min(A_x, A_y)$
01	Y	$V_{yx}C_y \bullet A_y$	$\max(A_x, A_y)$	$\max(A_x, A_y)$	$V_{yx} \bullet A_y$
11		$C_y \bullet A_y$	$\max(A_x, A_y)$	$(C_x + C_y) \bullet \max(A_x, A_y)$	A_y
10		$V_{xy}C_y \bullet \min(A_x, A_y)$	$V_{xy} \bullet A_x$	A_x	$\min(A_x, A_y)$

Fig. 8. The last event time A_z^{W1} for a 2-input AND gate under the W1 waveform model.

delay computation; i.e., the resulting last event time $\max(\varphi_{xz}^{\varphi 1} \cdot A_x, \varphi_{yz}^{\varphi 1} \cdot A_y)$ is indeed the same as A_z^{W1} , as can easily be verified.

W2 waveform model. Although the W1 model is significantly simpler than the W0 model, it still may not be practical for delay calculations in large circuits. We next ignore initial values, resulting in the *W2 waveform model* depicted in Figure 10. This is known in the literature as *floating mode*, and was introduced by Chen and Du [1993]. In this model, a waveform ω_x is characterized as

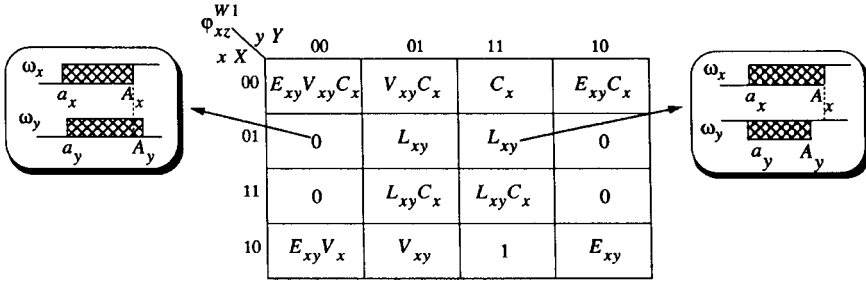
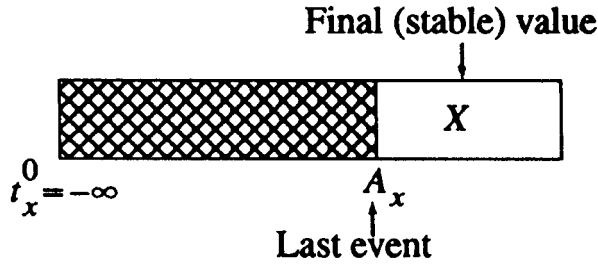
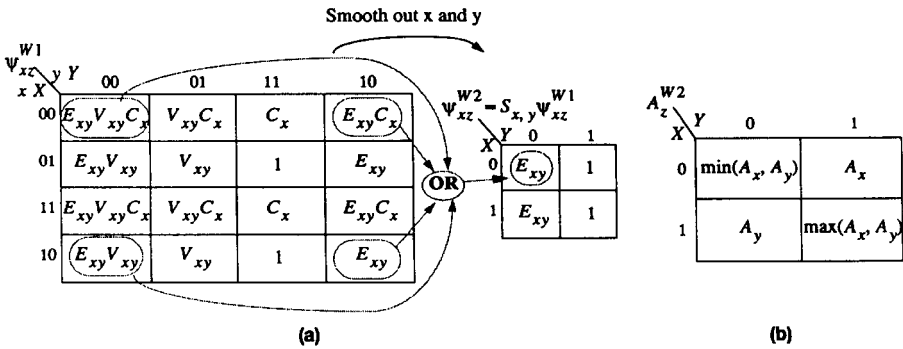
$$\omega_x^{W2} = S_x \omega_x^{W1} = ((t_x^0, U), (A_x, X)) \quad (15)$$

We again consider the propagation of last input events and derive the PCs accordingly.

The EPC Ψ_{xz}^{W2} for the W2 model is obtained from Ψ_{xz}^{W1} by smoothing out the initial values x and y , as shown in Figure 11(a). The EPC Ψ_{xz}^{W2} exactly matches the conditions for *viability* given in McGeer and Brayton [1989]. Under viability an event propagates if the stable value Y of input y is non-controlling, or the stable value Y is controlling, but the event on input x is earlier than that on input y . It can be easily seen that these conditions are equivalent to Ψ_{xz}^{W2} . Thus viability is the correct EPC for event propagation (as well as delay calculation) under the W2 model. The last event time A_z^{W2} can be obtained either by using Equation (3)—note that $\max(E_{xy} \cdot A_x, E_{yx} \cdot A_y) = \min(A_x, A_y)$ —or by $A_z^{W2} = S_{xy} A_z^{W1}$, as shown in Figure 11(b).

We can derive the DPC φ_{xz}^{W2} similarly. The resulting function $\varphi_{xz}^{W2} = S_{x,y} \varphi_{xz}^{W1}$ is shown in Figure 12. This DPC is identical to the floating mode condition.

THEOREM 4. *The floating mode condition is incorrect for event propagation.*

Fig. 9. the DPC ϕ_{xz}^{W1} of a 2-input AND gate.Fig. 10. Waveform for a node x under the W2 model.Fig. 11. (a) The EPC Ψ_{xz}^{W2} ; (b) the last event time A_z^{W2} under the W2 model.

PROOF. When discussing the W1 model, we saw that the DPC ϕ_{xz}^{W1} is incorrect, as it propagates only those input events that are guaranteed to be the latest at the output. Since $\phi_{xz}^{W2} \subseteq \Psi_{xz}^{W2}$, the same holds true for ϕ_{xz}^{W2} , which is also evident from Figure 12.

In Section 2, we saw an example where the floating mode condition fails to recognize a propagating event. Looking at Ψ_{xz}^{W2} and ϕ_{xz}^{W2} , we see that they are different for two input combinations. An example is given in Chen and Du [1993], where it is stated that viability incorrectly identifies a path as true. This example is reproduced in Figure 13 with the path in question

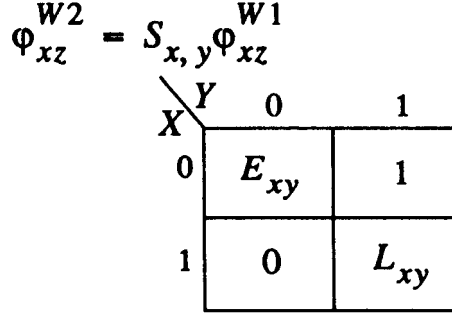


Fig. 12. The DPC ϕ_{xz}^{W2} under the W2 model is identical to the floating mode condition. .

highlighted. From the perspective of event propagation, we find that an event does, in fact, propagate along this path, as indicated in the timing diagram. An 0-width glitch, a special case of a pulse, occurs on node f and propagates to the output z . It should be emphasized that glitches cannot be distinguished from proper transitions in the W2 model because they introduce instability. Hence, due to the safeness requirement, they must be assumed to propagate. The floating mode condition, however, fails to identify this fact. On the other hand, it computes the same last event time, since we have the following from (3), which confirms previous results [Chen and Du 1993].

$$\max(\phi_{xz}^{W2} \cdot A_x, \phi_{yz}^{W2} \cdot A_y) = A_z^{W2}$$

□

An interesting case occurs when we further ignore the difference $A_x - A_y$ between two input events. For ϕ_{xz}^{W2} , this means calculating $S_{(A_1-A_y)} \phi_z^{W2}$, which we denote by ϕ_{xz}^{SC} . From (11), we have $S_{(A_x-A_y)} E_{xy} = S_{(A_x-A_y)} L_{xy} = 1$. Thus

$$\phi_{xz}^{SC} = S_{(A_x-A_y)} \phi_z^{W2} = \overline{X} + Y$$

This DPC is *not* the static sensitization condition, as one might expect; it is actually the static cosensitization condition introduced by Devadas et al. [1994], and is shown in Figure 14 along with the last event time calculated using (3). Since $\Psi_{xz}^{W2} \subseteq \phi_{xz}^{SC}$ is not always satisfied, the following result holds.

THEOREM 5. *The static cosensitization condition is incorrect for event propagation.*

The example given in Figure 2 for the floating mode condition applies to static cosensitization as well. It is correct with respect to delay computation because the last event time A_z^{SC} shown in Figure 14(b) satisfies $A_z^{SC} \geq A_z^{W2} \geq A_z^{W0}$.

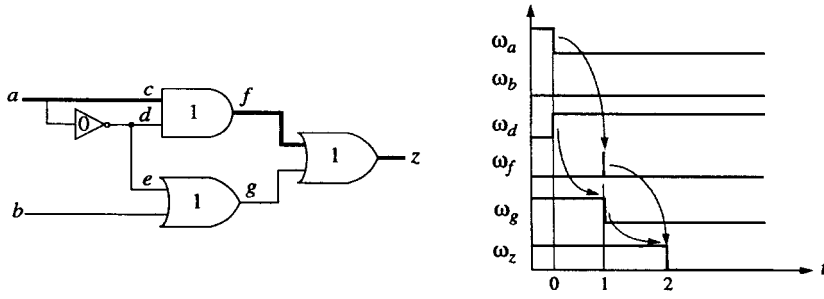


Fig. 13. Example circuit of Chen and Du [1993] where viability correctly identifies the propagation of an event.

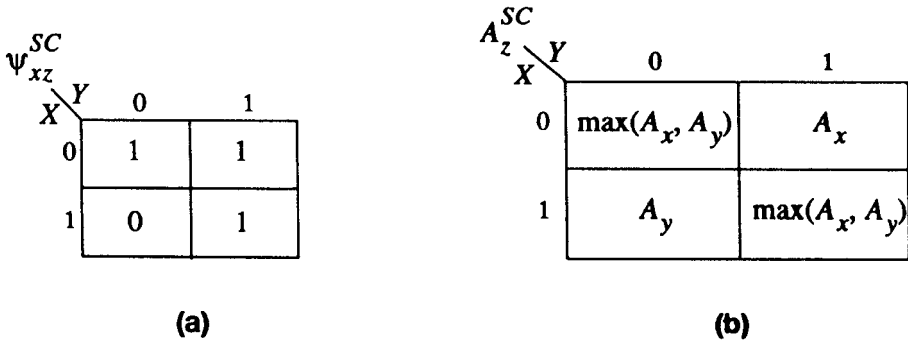


Fig. 14. (a) The static cosensitization condition φ_{xz}^{SC} ; (b) the last event time A_z^{SC} .

As for Ψ_{xz}^{W2} , the above approximation yields $S_{(A_x - A_y)} \Psi_z^{W2} = 1$, that is, the EPC used for topological analysis, which is discussed next. So for event propagation, ignoring dependence on input event times produces results as safe (or pessimistic) as those obtained by ignoring all values of the waveforms.

W3 waveform model. There is only one variable left in the W2 model that we can smooth out, namely, the final value X . The resulting waveform model is called W3.

$$\omega_x^{W3} = S_X \omega_x^{W2} = ((t_x^0, U), (A_x, U)) \quad (16)$$

In this case, no functional information is retained in the PCs. We are therefore left with topological (structural) information only. As a result, delay computation under this model is equivalent to topological delay analysis.

The EPC Ψ_{xz}^{W3} for the W3 model is derived by smoothing X and Y from Ψ_{xz}^{W2} :

$$\Psi_{xz}^{W3} = S_{X,Y} \Psi_{xz}^{W2} = 1$$

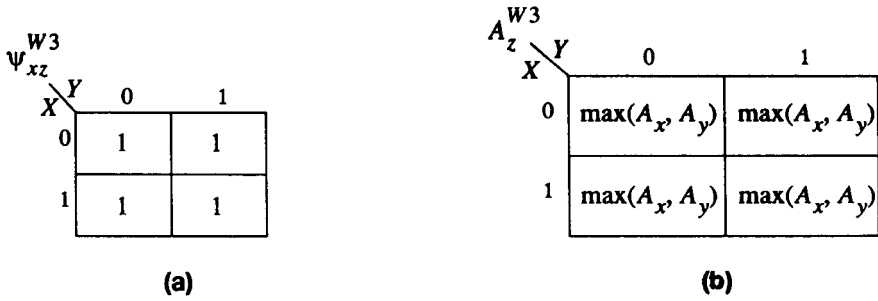


Fig. 15. (a) the EPC Ψ_{xz}^{W3} ; (b) the last event time A_z^{W3} under the W3 model.

which means that all events propagate unconditionally. Computing the last event time A_z^{W3} either via Equation (3) or $A_z^{W3} = S_{X,Y} A_z^{W2}$ yields $\max(A_x, A_y)$, that is, the longest topological path delay to the output of the AND gate. Thus, delay calculation under this model is indeed equivalent to topological delay analysis. For comparison, both the EPC and the last event time are shown in Figure 15 as functions of the final values X and Y .

To derive the DPC φ_{xz}^{W3} , we apply the smoothing $S_{X,Y}$ operator to φ_{xz}^{W2} , which produces $\varphi_{xz}^{W3} = S_{X,Y} \varphi_{xz}^{W2} = 1$; this is identical to Ψ_{xz}^{W3} . Therefore, under the W3 waveform model, event propagation and (topological) delay computation become equivalent.

Summary of waveform models. The waveform models and propagation conditions (EPCs and DPCs) introduced so far are summarized in Figure 16. Model complexity decreases as one moves from the W0 model up to W3. The safeness or pessimism of the PCs increases in the same direction, since approximation implies pessimism. The *smoothing* relation between these PCs is denoted by a thick arrow. For example, if a thick arrow goes from λ^Q to λ^R , then λ^R is obtained by smoothing out from λ^Q some of its variables. A thin arrow, on the other hand, is simply a *covering* relation, that is, if a thin arrow goes from λ^Q to λ^R , then $\lambda^Q \subseteq \lambda^R$, which is a weaker relation than smoothing.

Like the PCs, the circuit delays vary from the exact value calculated under the W0 model to the topological longest delay calculated under the W3 model. A related result by Lam et al. [1993] is worth mentioning here: Under the condition that every gate in a circuit has variable delay and no distinct paths have the same set of gates, floating delay, which is τ^{W2} in our notation, is the same as the exact delay τ^{W0} , referred to as delay by sequences of vectors in Lam et al. [1993].

5. EXTENSIONS

The derivation of PCs and last event times presented in the previous sections for an AND gate can be easily extended to other gates. For brevity, in this section we mostly consider the W2 waveform model. The case of

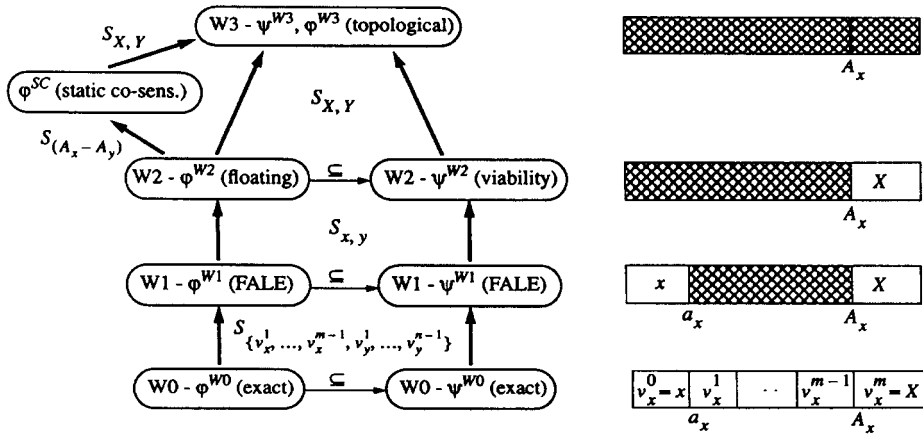


Fig. 16. Summary of the waveform models and their relationships.

buffers and inverters is trivial. All PCs are equal to 1, and the output last event time is the same as the input last event time, that is, with a zero-delay; it is simply shifted with a non-zero delay, as explained later.

For an OR gate, the controlling and non-controlling values are, respectively, 1 and 0, the opposite of those for an AND gate. Thus, the PCs and the last event time can be obtained from those of an AND gate by complementing the inputs. For example, the EPC Ψ_{xz}^{W2} becomes $\bar{Y} + E_{xy}$, and the DPC φ_{xz}^{W2} becomes $XE_{xy} + \bar{Y}L_{xy}$ for a 2-input OR gate.

An XOR gate has no controlling value since the output is always sensitive to changes in its inputs. This implies that the EPC Ψ_{xz}^{W2} for an XOR gate is the constant 1. For the same reason, the last input event always determines the last output event, hence the DPC φ_{xz}^{W2} is L_{xy} . The last event time A_z^{W2} , which can be computed using (3), is $\max(A_x, A_y)$ for all input combinations.

Concerning the basic gates with complemented outputs, namely NAND, NOR, and XNOR gates, event propagation is the same as the uncomplemented case. Hence, the PCs as well the last event times are identical to those for the corresponding uncomplemented gates. All the relationships among PCs that are shown for the AND gate in Figure 16 are valid for other gate types as well. For example, $\Psi_{xz}^{W2} \supseteq \varphi_{xz}^{W2}$ holds true for both OR and XOR gates.

Generalization of the analysis to gates with more than two inputs is straightforward. Consider a 3-input AND gate with inputs w , x , and y , and output z . The EPC and the DPC for the last event on input w and the last event time for z under the W2 model are shown in Figure 17. The early and late predicates introduced in Section 4 are extended to multiple inputs; for example, $E_{wxy} = E_{wx}E_{wy}$ means that w is earlier than both x and y .

We now give the PCs and the last event time for an n -input AND gate; other gates are similar. Let the gate inputs be x_1, x_2, \dots, x_n and the

Ψ_{wz}^{W2}

W	XY	00	01	11	10
0		E_{wxy}	E_{wx}	1	E_{wy}
1		E_{wxy}	E_{wx}	1	E_{wy}

Φ_{wz}^{W2}

W	XY	00	01	11	10
0		E_{wxy}	E_{wx}	1	E_{wy}
1		0	0	L_{wxy}	0

(a)

A_z^{W2}

W	XY	00	01	11	10
0		$\min(A_w, A_x, A_y)$	$\min(A_w, A_x)$	A_w	$\min(A_w, A_y)$
1		$\min(A_x, A_y)$	A_x	$\max(A_w, A_x, A_y)$	A_y

(b)

Fig. 17. (a) the EPC Ψ_{wz}^{W2} and DPC Φ_{wz}^{W2} ; (b) the last event time A_z^{W2} for a 3-input AND gate.

output be z . The EPC $\Psi_{x_1z}^{W2}$ for the last event on x_1 can be expressed as follows.

$$\Psi_{x_1z}^{W2} = \bar{X}_2\bar{X}_3 \dots \bar{X}_n E_{x_1x_2 \dots x_n} + X_2\bar{X}_3 \dots \bar{X}_n E_{x_1x_3 \dots x_n} + \dots$$

$$+ X_2 \dots X_{n-1} \bar{X}_n E_{x_1x_n} + X_2X_3 \dots X_n$$

where each term, except the last one, is of the form $X_{j_1} \dots X_{j_m} \bar{X}_{i_1} \dots \bar{X}_{i_k} E_{x_1x_{i_1} \dots x_{i_k}}$, $m + k = n - 1$, $0 \leq m \leq n - 1$, $0 \leq k \leq n - 1$, and

$$E_{x_1x_{i_1} \dots x_{i_k}} = E_{x_1x_{i_1}} E_{x_1x_{i_2} \dots} E_{x_1x_{i_k}}$$

In this expression, $E_{x_1x_{i_j}}$ is the usual early predicate $A_{x_1} \leq A_{x_{i_j}}$. The DPC $\Phi_{x_1z}^{W2}$ is given by

$$\Phi_{x_1z}^{W2} = \bar{X}_1 \Psi_{x_1z}^{W2} + X_1 X_2 X_3 \dots X_n L_{x_1x_2 \dots x_n}$$

where $L_{x_1x_2 \dots x_n} = L_{x_1x_2} \dots L_{x_1x_n}$, and $L_{x_1x_j}$ is the usual late predicate $A_{x_1} \geq A_{x_j}$. The last event time A_z^{W2} is calculated by

$$A_z^{W2} = \begin{cases} \min(A_{x_{i_1}}, A_{x_{i_2}}, \dots, A_{x_{i_k}}) & X_{i_1} = 0, \dots, X_{i_k} = 0, k > 0 \\ \max(A_{x_1}, A_{x_2}, \dots, A_{x_n}) & X_1 = 1, \dots, X_n = 1 \end{cases}$$

Non-zero gate delays can also be easily taken into account. In Figure 18(a), the delays for an AND gate are shown as a function of its inputs. A delay d_i is assumed for each input combination XY ; this is known as the *state-dependent delay* model [Sakallah 1995; Sun et al. 1994]. The resulting last event time A_z^{W2} for the W2 model is shown in Figure 18(b); for each

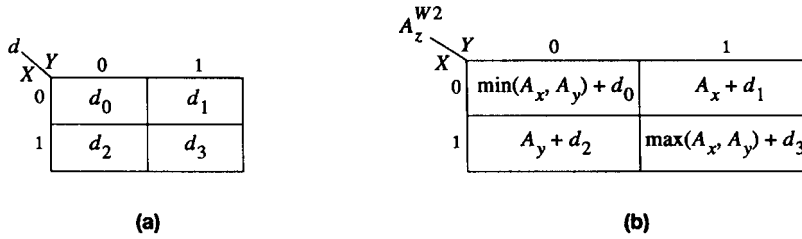


Fig. 18. (a) State-dependent delays for a 2-input AND gate; (b) the last event time.

input combination XY , A_z^{W2} is simply shifted by d_i . Other common delay models are special cases of the state-dependent delay case. For example, all d_i 's are equal for the transport delay model. In the rise/fall delay model, d_0 , d_1 , and d_2 are equal to a falling delay d_f , and d_3 is a rising delay d_r . With more complex waveform models, delays can be a function of other parameters. For example, in the W1 model, delays can be dependent on initial values as well. In such a case, when going from the W1 model to the W2 model, the maximum delay among all values of initial values must be retained for correctness.

The above discussion assumes that delays are fixed. A more realistic model is the min/max delay model, in which the delay d_i of a gate varies within a range $[l_i, u_i]$, where l_i and u_i are, respectively, lower and upper bounds on d_i . An established result [Chen and Du 1993] for floating mode, which therefore applies to the W2 model, is that delay computation with min/max delays gives the same result as does a delay model where the upper bounds are treated as fixed delays. To see why, consider the waveform at the output node of a gate. Since the W2 model ignores all values and events until the node stabilizes, the lower bound on gate delay has no effect on the waveform of the gate output. The same is true of the W3 model. Thus, any delay computation method that assumes the W2 or W3 model and fixed delays is applicable to min/max delays as well. In the W0 and W1 cases, however, lower bounds do have an impact on waveforms. A delay range $[l_i, u_i]$ for a gate creates an interval of length $u_i - l_i$ and value U on the output waveform, which introduces pessimism into PC calculation. The expressions derived in Sections 3 and 4 for EPCs, DPCs, and last event times with fixed delays for the W0 and W1 models can be extended to handle min/max delays.

An issue related to the min/max delay model is robustness or the monotone speedup property [McGeer and Brayton 1993]. A PC λ is called *robust* if the circuit delay computed using λ does not increase when some gate delays are reduced. In the scenario described by McGeer and Brayton [1993], a PC or a delay computation method handles only fixed gate delays that are set to their maximum values or upper bounds. However, because these delays vary within a known range, some gates can have delays less than their maximum values. Robustness requires that the delay computed by the PC be an upper bound for any instance of the circuit. By this

definition, viability and the floating mode condition are robust. The fact that the lower bounds have no effect on the W2 waveforms is alone sufficient to confirm these results. On the other hand, delay computation under the W0 and W1 waveform models is not robust² because lower bounds do affect waveforms in these cases. In our view, if gate delays are uncertain, then delay computation should be performed with min/max delays as described in the preceding paragraph, rather than using fixed delays and then reasoning about the validity of results for min/max delays. Because of the safeness of delay computation, the correct delay will be found under *any* waveform model as long as the gate delays remain within their prescribed ranges.

A final note on this issue is that adding delays into the last event time does not affect the relationships in Figure 16. All the event times will simply shift according to the delay of the gate, as in Figure 18(b). Thus all the results established with zero delays are valid with non-zero delays also.

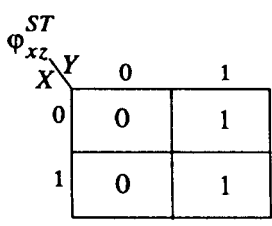
6. RELATIONS AMONG PCS

The basic PCs summarized in Figure 16 can be related to other PCs proposed in the literature. We consider here static sensitization [Benkoski et al. 1987], the loose condition [Chen and Du 1993], the VIPER condition [Chang and Abraham 1993], the condition proposed by Perremans, Claesen and DeMan [Perremans et al. 1989] denoted PCD, the Brand-Iyengar condition [Brand and Iyengar 1986] denoted BI, and the Du-Yen-Ghanta condition [Du et al. 1989] denoted DYG.

Static sensitization. Early attempts at solving the false path problem [Benkoski et al. 1987; Ju and Saleh 1991] employed static sensitization. Like static cosensitization, discussed in Section 4, static sensitization only deals with final stable values, and hence implicitly assumes the W2 waveform model. It is known that this DPC can overestimate [Ju and Saleh 1992; Silva and Sakallah 1993] as well as underestimate [McGeer and Brayton 1989] circuit delays. Under static sensitization, an input event propagates to the output only when the side inputs have a (stable) non-controlling value. For a 2-input AND gate, the PCs are, $\varphi_{xz}^{ST} = Y$, $\varphi_{yz}^{ST} = X$, which are shown in Figure 19(a). The last event time A_z^{ST} calculated using Equation (3) is given in Figure 19(b).

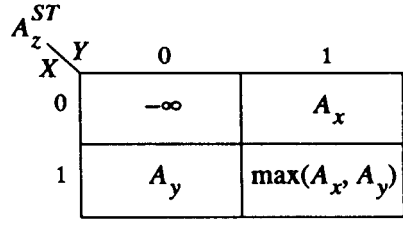
Recall from our analysis in Section 4 that for a condition to be correct for event propagation under the W2 model, it must cover the EPC Ψ^{W2} . For static sensitization, just the opposite is true, that is, $\varphi^{ST} \subseteq \Psi^{W2}$, which implies that static sensitization is incorrect. To see why it can underestimate the circuit delay, consider a 2-input AND gate whose inputs both have a pulse whose initial and final values are 0. Assuming the input pulses overlap and the gate has zero delay, the last event time at the output is

²An example is given in McGeer and Brayton [1989] where the so-called dynamic sensitization, an informal version of propagation under our W0 model, is shown to fail the robustness test.



φ_{xz}^{ST}	Y	0	1
X	0	0	1
	1	0	1

(a)



A_z^{ST}	Y	0	1
X	0	$-\infty$	A_x
	1	A_y	$\max(A_x, A_y)$

(b)

Fig. 19. (a) The static sensitization condition φ_{xz}^{ST} ; (b) the last even time A_z^{ST} .

$\min(A_x, A_y)$. In this case, however, static sensitization produces $A_z^{ST} = -\infty$ regardless of the input last event times. When static sensitization does not underestimate, we have $A^{W0} \leq A^{ST}$. Since $A^{ST} \leq A^{W2}$ is always satisfied, $A^{W0} \leq A^{ST}$, which proves the following result.

THEOREM 6. *When static sensitization does not underestimate delay, its estimate is equal to or better than that of the W2 model.*

Loose/Viper condition. The loose condition proposed by Chen and Du [1993] is equivalent to that used by Viper [Chang and Abraham 1993]; we refer to it as φ^{LV} . It assumes the W2 waveform model. Consider a 2-input gate with inputs x and y and output z . Under this DPC, an event on input x propagates to the output if the stable value Y of input y is non-controlling, or the stable values X and Y are both controlling and the event on input x is earlier than that on input y . For an AND gate, these conditions correspond to $\varphi_{xz}^{LV} = \overline{X}E_{xy} + Y$, which is shown in Figure 20 along with the last event time A_z^{LV} . Since $\varphi_{xz}^{LV} \subseteq \Psi_{xz}^{W2}$, this DPC is incorrect for event propagation. The example given in Figure 2 for the floating mode condition applies to the loose condition also. The loose condition is correct with respect to delay computation because $A_z^{LV} = A_z^{W2}$.

PCD condition. The condition proposed by Perremans et al. [1989] also assumes the W2 waveform model. Their method computes for every node v a dynamic variable T_v which represents an upper bound to the last event time A_v . According to PCD, an event on input x of a 2-input gate propagates to the output z under the following conditions: If the stable value X of input x is non-controlling, then Y must be non-controlling, too, or if the stable value X of input x is controlling and $A_x \leq T_y$, then Y must be non-controlling. From these conditions, we obtain the PCD condition $\varphi_{xz}^{PCD} = \overline{X}E'_{xy} + Y$ for the 2-input AND case, where $E'_{xy} = A_x \leq T_y$; see Figure 21(a). Since $\Psi_{xz}^{W2} \subseteq \varphi_{xz}^{PCD}$ is not satisfied, the PCD condition is incorrect for event propagation. The last event time A_z^{PCD} is given in Figure 21(b), where $A' = \max(E'_{xy} \cdot A_x, E'_{xy} \cdot A_y)$. The fact that $E'_{xy} \supseteq E_{xy} = A_x \leq A_y$

ϕ_{xz}^{LV}	$X \backslash Y$	0	1
0		E_{xy}	1
1		0	1

(a)

A_z^{LV}	$X \backslash Y$	0	1
0		$\min(A_x, A_y)$	A_x
1		A_y	$\max(A_x, A_y)$

(b)

Fig. 20. (a) The loose/Viper condition ϕ_{xz}^{LV} ; (b) the last event time A_z^{LV} .

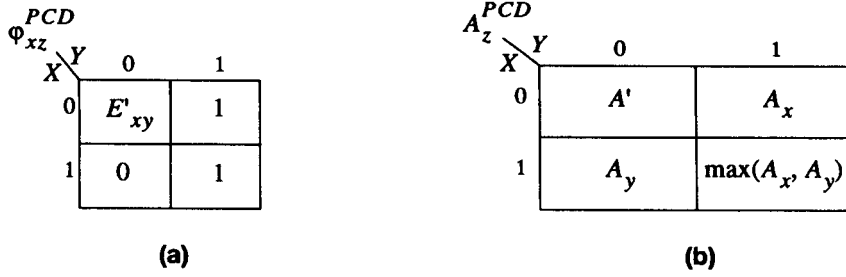
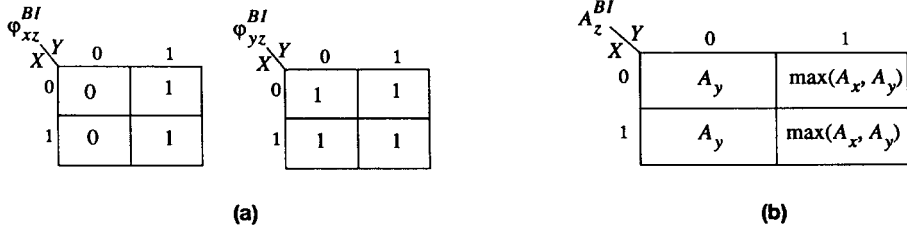
implies $A_z^{W2} \leq A'$; that is, the PCD condition is correct for delay computation. However, its estimate is looser than that of the W2 model.

BI condition. Brand and Iyengar proposed a condition to fix the underestimation problem of static sensitization [Brand and Iyengar 1986]. Their DPC orders the inputs to a gate, and for the sensitization of an input x , it imposes the static sensitization condition only to the inputs following x . The inputs preceding x are ignored. Figure 22 shows the BI DPCs ϕ_{xz}^{BI} and ϕ_{yz}^{BI} and the last event time A_z^{BI} for a 2-input AND gate. Note that the BI condition is not symmetric, unlike those seen so far. Since input y is the last input, the DPC ϕ_{yz}^{BI} is the constant 1. Because for each gate there is one DPC ϕ_{xz}^{BI} that is the same as static sensitization, the BI condition is also incorrect for event propagation. On the other hand, it is correct for delay computation, since $A_z^{W2} \leq A_z^{BI} \leq A_z^{W3}$. As can be seen, the delay for the input combination $XY = 00$, which is the source of error for static sensitization, is chosen to be A_y , the delay up to input y .

DYG condition. Although the BI condition solved the problem of underestimation with static sensitization, the tightness of its estimates are very dependent on how gate inputs are ordered. For example, when the longest topological path is made up of signal nodes all of which are the last input to some gate, this path is assumed sensitized by the BI condition. To reduce the pessimism of the BI condition, Du et al. [1989] propose a DPC that uses topological delays as a “heuristic.” Let δ_v and Δ_v be the length of the shortest and longest path ending in node v , respectively. Consider a 2-input gate with inputs x and y and output z . According to the DYG condition, an event on input x propagates to the output z under the following conditions: If $\delta_y > A_x$, then X must be controlling, or if $\Delta_y < A_x$, then Y must be non-controlling.

Depending on the values δ_x , Δ , δ_y , and Δ_y , there are 7 possible cases for the 2-input AND gate:

$$(1) A_x < \delta_y, \delta_x \leq A_y \leq \Delta_x (A_x < A_y) : \phi_{xz}^{DYG} = \bar{X}, \phi_{yz}^{DYG} = 1$$

Fig. 21. (a) The PCD condition φ_{xz}^{PCD} ; (b) the last event time A_z^{PCD} .Fig. 22. (a) The BI conditions φ_{xz}^{BI} and φ_{yz}^{BI} ; (b) the last event time A_z^{BI} .

- (2) $A_x < \delta_y, A_y > \Delta_x (A_x < A_y) : \varphi_{xz}^{DYG} = \bar{X}, \varphi_{yz}^{DYG} = X$
- (3) $\delta_y \leq A_x \leq \Delta_y, A_y < \delta_x (A_x > A_y) : \varphi_{xz}^{DYG} = 1, \varphi_{yz}^{DYG} = \bar{Y}$
- (4) $\delta_y \leq A_x \leq \Delta_y, \delta_x \leq A_y \leq \Delta_x : \varphi_{xz}^{DYG} = 1, \varphi_{yz}^{DYG} = 1$
- (5) $\delta_y \leq A_x \leq \Delta_y, A_y > \Delta_x (A_x < A_y) : \varphi_{xz}^{DYG} = 1, \varphi_{yz}^{DYG} = X$
- (6) $A_x > \Delta_y, A_y < \delta_x (A_x > A_y) : \varphi_{xz}^{DYG} = Y, \varphi_{yz}^{DYG} = \bar{Y}$
- (7) $A_x > \Delta_y, \delta_x \leq A_y \leq \Delta_x (A_x > A_y) : \varphi_{xz}^{DYG} = Y, \varphi_{yz}^{DYG} = 1$

As with the BI condition, for many cases, we have $\varphi_{xz}^{DYG} \subseteq \Psi_{xz}^{W2}$, so the DYG condition is incorrect for event propagation. Regarding the last event time, we find that for cases 1, 3, and 4, $A_z^{DYG} \geq A_z^{W2}$, and that for cases 2, 5, 6, and 7, $A_z^{DYG} = A_z^{W2}$. So the DYG condition is correct for delay computation.

Although we have established relationships among PCs and last event times for individual gates, they are applicable to an entire circuit, as proved by the following theorem.

THEOREM 7. Consider two delay computation methods Q and R whose respective PCs λ^Q and λ^R satisfy $\lambda^Q \supseteq \lambda^R$ for any gate in a circuit. Then, for any path P in the circuit, $\lambda_P^Q \supseteq \lambda_P^R$ and $\tau^Q \geq \tau^R$, where τ^Q and τ^R are the circuit delays computed by Q and R , respectively.

PROOF. Let path P consist of nodes $0, 1, \dots, k$.

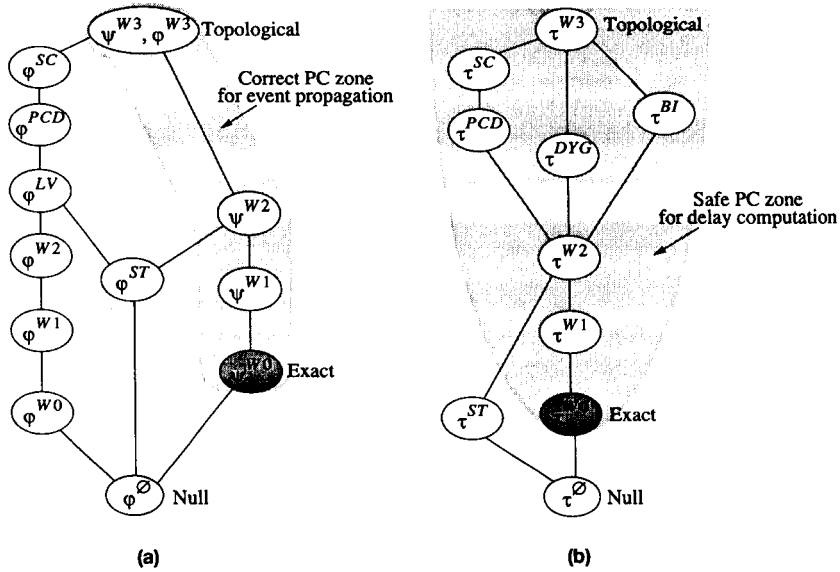


Fig. 23. Hasse diagrams for (a) a poset of PCs; (b) a poset of the delay estimates of the PCs.

$$\lambda_P^Q = \prod_{i=0}^{k-1} \lambda_{i,i+1}^Q, \text{ and } \lambda_P^R = \prod_{i=0}^{k-1} \lambda_{i,i+1}^R$$

By assumption, we have $\lambda_{i,i+1}^Q \supseteq \lambda_{i,i+1}^R$, for $i = 0 \dots k-1$, so $\lambda_P^Q \supseteq \lambda_P^R$. The proof for $\tau^Q \geq \tau^R$ can be obtained from the proof of Theorem 3 simply by substituting Q for M and R for $W0$. \square

The relationships among all PCs can be defined as a partially ordered set (poset) with respect to the covering relation and can be represented by a Hasse diagram. Figure 23(a) shows the Hasse diagram for a poset of all PCs discussed so far, with the exception of the BI and DYG conditions. A *null* PC $\varphi^{\emptyset} = 0$ is also added, which does not allow any event to propagate. The other extreme PC is $\Psi^{W3} = \varphi^{W3} = 1$, which propagates any event. Everything else lies between these extremes, and is tighter than Ψ^{W3} and looser than φ^{\emptyset} . (For clarity, the obvious relations $\Psi^{Wi} \supseteq \varphi^{Wi}$, where $i = 0, 1, 2$, are not shown.) The EPC Ψ^{W0} is the exact one for event propagation. A condition Q for which $\Psi^Q \supseteq \Psi^{W0}$ holds is correct by Definition 5. The conditions (EPCs) with this property are indicated in Figure 23(a). All others are incorrect. The BI and DYG conditions not shown in the figure have conditions dependent on ordering of inputs and topological delays, respectively, and vary between φ^{ST} and Ψ^{W3} . Like static sensitization, they are incorrect for event propagation.

Another poset can be defined for the circuit delay estimates of the PCs based on the \geq relation; see Figure 23(b). The extremes here are $\tau^{\emptyset} =$

$-\infty$, which corresponds to φ^0 , and $\tau^{W3} = \Delta^{\max}$, which is the longest topological delay computed by Ψ^{W3} or φ^{W3} . The exact delay is τ^{W0} , computed by either Ψ^{W0} or φ^{W0} . Correct or safe PCs, indicated by shading in the figure, produce an estimate between τ^{W0} and τ^{W3} . Incorrect PCs such as static sensitization can produce an estimate less than τ^{W0} .

The poset of PCs whose Hasse diagram appears in Figure 23(a) suggests that other PCs can be obtained by creating Boolean functions of the values of inputs and predicates based on the ordering of input events. For the combined W2 and W3 models, the number of all possible PCs is 625.³ This number grows if other parameters such as topological delays are included. Obviously, many of these PCs will be meaningless. However, useful ones can be found, as described in the next section.

7. APPLICATIONS

With the help of the preceding analysis, one can create new and potentially useful PCs. Of particular interest are those that only depend on final stable values, like static cosensitization. We have found two such DPCs that we call φ^{S1} and φ^{S2} ; they are shown in Figure 24 for a 2-input AND gate, along with their corresponding last event times computed according to Equation (3). As the figure reveals, S1 and S2 are a combination of static cosensitization and static sensitization, and hence blend the safety of the former and the tightness of the latter. While S1 imposes static sensitization on input y , S2 imposes static sensitization on input x . Their estimates A_z^{S1} and A_z^{S2} of last event times lie between A_z^{W2} and A_z^{SC} .

While S1 and S2 can be used individually, we propose the following DPC, called *safe static* (SS), that makes use of both. It is defined as follows for any 2-input gate:

$$\varphi^{SS} = \begin{cases} \varphi^{S1} & \text{if } \Delta_x < \Delta_y \\ \varphi^{S2} & \text{otherwise} \end{cases}$$

We note that our safe static SS is different from the similar-sounding DPC used by Silva and Sakallah [1994], which is equivalent to the floating mode condition. The idea here is to impose static sensitization, which has tighter conditions, on the input whose topological delay is longer. This reduces the probability of the longer path being reported true when it is actually false. In this respect, SS is similar to the DYG condition, which improves on the BI condition with the help of topological delays. Comparing the last event time for safe static with that for DYG, we find that for cases 1, 3, and 4, $A_z^{DYG} \geq A_z^{SS}$, and that for cases 2, 5, 6, and 7, $A_z^{DYG} \geq A_z^{SS}$. So safe static provides a better estimate of the circuit delay than DYG.

We now give an example to illustrate the differences in the estimates provided by all the PCs discussed in this paper, with the exception of PCD,

³With three possible orderings of two input event times A_x and A_y , and two final values X and Y , one can create $(3 + 2)^{2^2} = 625$ Boolean functions.

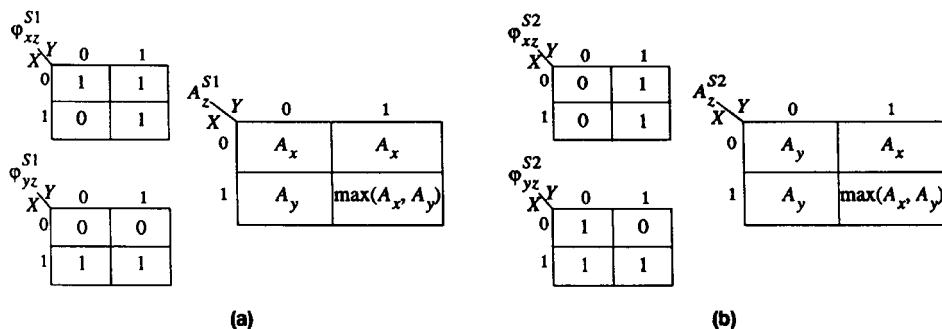


Fig. 24. The S1 conditions ϕ_{xz}^{S1} and ϕ_{yz}^{S1} , and A_z^{S1} ; (b) the S2 conditions ϕ_{xz}^{S2} and ϕ_{yz}^{S2} , and A_z^{S2} .

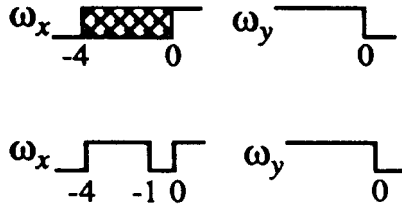
for which the upper bound T_v computed internally at run time for each node cannot be reproduced. The estimates for the last event time (or delay) for the circuit shown in Figure 1 are given in Table I along with the corresponding conditions and longest paths. Observe that the delay estimates agree with the poset of Figure 23(b). The exact delay calculated under the W0 model depends on the selection of Ω , the set of all input waveforms. If any number of events are allowed on the input waveforms, the exact delay is 6. If the number of input events is limited to a maximum of 1 (transition mode), the exact delay reduces to 4. For the former case, the delay calculated by static sensitization is an underestimate, while, for the latter case, it is the same as the exact delay. The new SS condition achieves the same delay as W2. Static cosensitization, on the other hand, is as pessimistic as topological analysis.

To evaluate the tightness of the proposed DPCs S1, S2, and SS, we have performed experiments with CAT [Yalcin and Hayes 1995], a symbolic timing analyzer that can compute a circuit's delays and associated conditions with any PC. The delay estimates of S1, S2, and SS along with those of W3 (topological), static cosensitization, DYG, W2 (floating mode condition and viability), and static sensitization are shown in Table II for some well-studied circuits: the ISCAS-85 benchmarks, carry-skip adders, and several examples from Devadas et al. [1993].

As expected, the estimates of S1, S2, and SS lie between those of W2 and static cosensitization. For the ISCAS-85 circuits, S1 and S2 yield the same delay values as W2 except for c1908, where their estimate is equal to the longest topological delay. For carry-skip adders, S1 and S2 report the longest topological path delay. The delay estimates of static sensitization for cla.16 and tau92ex2 are less than those of W2, which indicates the possibility of underestimation for static sensitization. The estimates of S1 and S2 are all safe, as shown in the table. The estimates of SS are very good; they are the same as those of W2 for all the examples except two cases, where SS overestimates by only 1 (tau92ex1) and 2 (tau92ex2).

Table I. Delay Estimates for the Circuit of Figure 1

Propagation condition	Delay	A longest path along which an event can propagate	Condition/waveforms for an event to propagate along longest path
W3 (topological)	10	$a - c - e - i - l - o - p - z$	1
Static co-sensitization	10	$a - c - e - i - l - o - p - z$	X
BI (Brand-Iyengar)	7	$b - j - l - o - p - z$	\bar{Y}
DYG (Du-Yen-Ghanta)	7	$b - j - l - o - p - z$	\bar{Y}
S1	6	$a - c - e - h - n - p - z$	X
S2	7	$b - j - l - o - p - z$	Y
SS (safe static)	6	$a - c - e - h - n - p - z$	X
W2 (viability, floating)	6	$a - c - e - h - n - p - z$	$X\bar{Y}$
W1 (multiple input events)	6	$a - c - e - i - l - o - p - z$	See Figure 25
W0 (multiple input events)	6	$a - c - e - i - l - o - p - z$	See Figure 26
Static sensitization	4	$a - c - g - z$	X
W1 (single input event)	4	$a - c - g - z$	$xXyY, Ax = Ay =$ $ax = ay = 0$
W0 (single input event)	4	$a - c - g - z$	$xXyY, Ax = Ay =$ $ax = ay = 0$



Figs. 25 and 26.

These results suggest that SS is quite tight, especially considering the fact that it employs stable signal values only.

Finally, we make some observations regarding the computation times. While computation is linear in circuit size for the W3 model (the topological case), it is, in general, exponential for the W2 model. However, the delay computation method, which incorporates a particular PC as well as its implementation, has a significant impact on computation time. A reasonably fair comparison can be done among PCs that assume the same waveform model. In our case, we have observed that the computation times for PCs that only depend on stable signal values, e.g., DYG and safe static, are similar. This, along with the results on delay estimates, suggests that safe static provides an excellent accuracy/computation time tradeoff under the W2 model.

8. CONCLUSIONS

We have presented a systematic and unified analysis of event propagation conditions, which have been mostly studied informally before. A series of waveform models has been developed to represent signal behavior at

Table II. Comparison of Delay Estimates Computed by Various PCs.

Circuit	W3(topological)	Static co-sens.	DYG	S1	S2	SS	W2	Static sens.
c432	17	17	17	17	17	17	17	17
c499	11	11	11	11	11	11	11	11
c880	24	24	24	24	24	24	24	24
c1355	24	24	24	24	24	24	24	24
c1908	40	40	37	30	40	37	37	37
c2670	32	30	31	30	30	30	30	30
c3540	47	46	46	46	46	46	46	46
c5315	49	47	47	47	47	47	47	47
c7552	43	42	42	42	42	42	42	42
csa.32.2	97	97	38	97	97	38	38	38
csa.64.4	161	161	46	161	161	46	46	46
csa.128.8	289	62	289	289	62	62	62	62
cla.16	34	34	34	34	34	34	34	33
tau92ex1	27	27	25	27	26	25	24	24
tau92ex2	93	62	56	55	46	44	42	41

different levels of detail. Our starting point is a rigorous waveform model, called the exact or W0 model, that specifies each event occurring in a circuit signal. With the novel use of a smoothing operator, we have obtained approximate waveforms by eliminating signal values from the exact model. Corresponding to the waveform models, a set of event propagation conditions or PCs is derived, which allows accuracy/computation tradeoffs to be made. We showed that most existing conditions are not based on how events actually propagate, but rather appear to have been developed simply to obtain an upper bound on the circuit delay. The relationships among the derived PCs and current conditions are analyzed. A by-product of this analysis is the ability to derive new PCs. We have demonstrated such a PC, called safe static (SS). The experimental results on a variety of circuits including the ISCAS-85 benchmarks indicate that the proposed SS PC produces more accurate estimates of the circuit delay than existing PCs with similar assumptions on circuit signals.

APPENDIX

PROOF OF THEOREM 1. We use (2-dimensional) induction on m and n , the number of events on inputs x and y , respectively. For this purpose, we write A_z^{W0} as a function of m and n , that is, $A_z^{W0}(m, n)$. The EPCs Ψ_{xz-j}^{W0} and Ψ_{yz-k}^{W0} are functions of m and n as well. So from (4), we have

$$\Psi_{xz-j}^{W0}(n) = \sum (t_y^i \leq t_x^j \leq t_y^{i+1})(v_y^i \neq 0) \quad (17)$$

$$\Psi_{yz-k}^{W0}(m) = \sum (t_x^i \leq t_y^k \leq t_x^{i+1})(v_x^i \neq 0) \quad (18)$$

where $1 \leq j \leq m$, $1 \leq k \leq n$. Equation (5) can thus be rewritten as

$$A_z^{W0}(m, n) = \max(t_x^1 \cdot \Psi_{xz-1}^{W0}(n), \dots, t_x^m \cdot \Psi_{xz-m}^{W0}(n), t_y^1 \cdot \Psi_{yz-1}^{W0}(m), \dots, t_y^n \cdot \Psi_{yz-n}^{W0}(m)) \quad (19)$$

For the basis, we consider $m = 1, n = 0$, and $m = 0, n = 1$. (When $m = n = 0$, that is, no event on x and y , $A_z^{W0}(0,0)$ is trivially $-\infty$.) For the first case, there is a single event on input x and no event on y . For the only event on x occurring at t_x^1 , we have $\Psi_{xz-1}^{W0} = (v_y^0 \neq 0)$. Substituting into (19), we obtain $A_z^{W0}(1,0) = t_x^1 \cdot (v_y^0 \neq 0)$, which evaluates to t_x^1 if $(v_y^0 \neq 0)$, and $-\infty$ otherwise. This means that if input y has the controlling value (0), the event on x does not propagate; otherwise it does, which is indeed correct. The second case, $A_z^{W0}(0,1)$ is similar. Hence, the basis holds.

For the inductive step, assume that the theorem holds for $A_z^{W0}(j, k), j + k < m + n$. We next show that it holds for $A_z^{W0}(m, n)$ also. First suppose that $t_x^m = t_y^n$. Notice that the definition of event implies that the condition in the summation of (17) evaluates to 1 either in the n th or $(n - 1)$ st interval, so $\Psi_{xz-m}^{W0} = 1$. Similarly, $\Psi_{yz-n}^{W0} = 1$. Because of the ordering of the events, we also have $t_x^0 \leq t_x^1 \leq \dots \leq t_x^m$ and $t_y^0 \leq t_y^1 \leq \dots \leq t_y^n$. Hence, $A_z^{W0}(m, n) = t_x^m = t_y^n$, which is the expected result.

Now suppose that $t_x^m \neq t_y^n$. Depending on the values of v_x^m and v_y^n , and whether $t_x^m > t_y^n$ or $t_x^m < t_y^n$, there are eight possible cases, as shown in Table III. As an example, consider the case, $v_x^m = 0, v_y^n = 0, t_x^m > t_y^n$. In this case, we have $\Psi_{xz-m}^{W0} = 0$ from (17). Since t_x^m is the latest event, the condition in the summation of (18) evaluates to 0 for the last iteration $i = m$ for all k . Thus, Ψ_{yz-k}^{W0} is a function of $m - 1$. As a result, $A_z^{W0}(m, n)$ reduces to $A_z^{W0}(m - 1, n)$, for which, by hypothesis, the theorem holds. In another example, consider $v_x^m \neq 0, v_y^n = 0, t_x^m < t_y^n$. It is easy to see that $\Psi_{yz-n}^{W0} = 1$. Since t_y^n is the latest event, we have $A_z^{W0}(m, n) = t_y^n$, as expected. Similarly, in all the other cases in Table III, $A_z^{W0}(m, n)$ either reduces to $A_z^{W0}(j, k), j + k < m + n$, which is assumed to hold, or yields the required value. Hence, by the principle of induction, the theorem holds for all m and n . \square

REFERENCES

- BRAND, D. AND IYENGAR, V. S. 1986. Timing Analysis Using Functional Relationships. In *Computer-Aided Design*.
- BROWN, F. 1090. *Boolean reasoning*. Kluwer Academic Publishers, Hingham, MA.
- CERNY, E. AND ZEJDA, J. 1994. Gate-level Timing Verification Using Waveform Narrowing. In *EuroDAC*, 374–379.
- CHANG, H. AND ABRAHAM, J. A. 1993. An Efficient Vigorously Sensitizable Path Extractor. In *Design Automation*, 112–117.
- CHEN, H.-C AND DU, D.-C. 1993. Path Sensitization in Critical Path Problem. *IEEE Trans. CAD 12* (Feb.), 196–207.

Table III. The Last Event Time A_z^{W0} as a Function of v_x^m and v_y^n .

v_x^m	v_y^n	t_x^m vs. t_y^n	$A_z^{W0}(m, n)$
0	0	$t_x^m > t_y^n$	$A_z^{W0}(m-1, n)$
0	0	$t_x^m > t_y^n$	$A_z^{W0}(m, n-1)$
0	1, U	$t_x^m > t_y^n$	t_x^m
0	1, U	$t_x^m > t_y^n$	$A_z^{W0}(m, n-1)$
1, U	0	$t_x^m > t_y^n$	$A_z^{W0}(m-1, n)$
1, U	0	$t_x^m > t_y^n$	t_y^n
1, U	1, U	$t_x^m > t_y^n$	t_x^m
1, U	1, U	$t_x^m > t_y^n$	t_y^n

- DEVADAS, S. 1992. Certified Timing Verification and the Transition Delay of a Logic Circuit. In *Design Automation*, 549–555.
- DEVADAS, S. 1993. Computation of Floating Mode Delay in Combinational Circuits: Theory and Algorithms. *IEEE Trans. CAD 12* (Dec.), 1913–1923.
- DEVADAS, S. 1994. Event Suppression: Improving the Efficiency of Timing Simulation for Synchronous Digital Circuits. *IEEE Trans. CAD 13* (June), 814–822.
- DU, D. H., YEN, S. H., AND GHANTA, S. 1989. On the general false path problem in timing analysis. In *Design automation conference* (Las Vegas, NV, June 25–29, 1989). ACM Press, New York, NY, 555–560.
- HRAPCENKO, V. 1978. Depth and Delay in a Network. *Soviet Math. Dokl. 19*, 1006–1009.
- JU, Y.-C. AND SALEH, R. A. 1991. Incremental techniques for the identification of statically sensitizable critical paths. In *ACM/IEEE design automation conference* (San Francisco, California, June 17–21, 1991). ACM Press, New York, NY, 541–546.
- LAM, W. K. C., BRAYTON, R. K., AND SANGIOVANNI-VINCENTELLI, A. L. 1993. Circuit delay models and their exact computation using Timed Boolean Functions. In *Design automation conference* (Dallas, TX, June 14–18, 1993). ACM Press, New York, NY, 128–134.
- MCGEER, P. C. AND BRAYTON, R. K. 1989. Efficient algorithms for computing the longest viable path in a combinational network. In *Design automation conference* (Las Vegas, NV, June 25–29, 1989). ACM Press, New York, NY, 561–567.
- MCGEER, P. AND BRAYTON, R. 1991. *Integrating Functional and Temporal Domains in Logic Design: The False Path Problem and its Implications*. Kluwer Academic Publishers, Hingham, MA.
- PERREMANS, S., CLAESEN, L., AND DE MAN, H. 1989. Static timing analysis of dynamically sensitizable paths. In *Design automation conference* (Las Vegas, NV, June 25–29, 1989). ACM Press, New York, NY, 568–573.
- SAKALLAH, K. 1995. *Dynamic Modeling of Logic Gate Circuits*. University of Michigan, Ann Arbor, MI.
- SHRIVER, E. J. AND SAKALLAH, K. A. 1992. Ravel: assigned-delay compiled-code logic simulation. In *Computer-aided design* (Santa Clara, CA, Nov. 8–12, 1992). IEEE Computer Society Press, Los Alamitos, CA, 364–368.
- SILVA, J. AND SAKALLAH, K. 1993. An Analysis of Path Sensitization Criteria. In *Computer Design*, 68–72.
- SILVA, J. AND SAKALLAH, K. 1994. Efficient and Robust Test Generation-Based Timing Analysis. In *Circuits and Systems*, 303–306.
- SUN, S. Z. ET AL. 1994. Efficient Timing Analysis for CMOS Circuits Considering Data Dependent Delays. In *Proceedings of the Conference on Computer Design* (Cambridge, MA, Oct. 10–12, 1994), 156–159.
- YALCIN, H. AND HAYES, J. P. 1995. Hierarchical timing analysis using conditional delays. In *Computer-aided design* (San Jose, CA, Nov. 5–9, 1995). IEEE Computer Society Press, Los Alamitos, CA, 371–377.

Received: December 1996; accepted: April 1997