EXACT MINIMUM DELAY COMPUTATION AND CLOCK FREQUENCIES

by

William K.C. Lam, Robert K. Brayton, and
Alberto L. Sangiovanni-Vincentelli

Memorandum No. UCB/ERL M93/40

28 May 1993

COVER PAGE

EXACT MINIMUM DELAY COMPUTATION AND CLOCK FREQUENCIES

by

William K.C. Lam, Robert K. Brayton, and
Alberto L. Sangiovanni-Vincentelli

Memorandum No. UCB/ERL M93/40

28 May 1993

ELECTRONICS RESEARCH LABORATORY

College of Engineering
University of California, Berkeley
94720

EXACT MINIMUM DELAY COMPUTATION AND CLOCK FREQUENCIES

by

William K.C. Lam, Robert K. Brayton, and
Alberto L. Sangiovanni-Vincentelli

Memorandum No. UCB/ERL M93/40

28 May 1993

ELECTRONICS RESEARCH LABORATORY

College of Engineering
University of California, Berkeley
94720

Exact Minimum Delay Computation and Clock Frequencies

William K.C. Lam * Robert K. Brayton Alberto L. Sangiovanni-Vincentelli
Department of EECS, University of California, Berkeley

Abstract

Minimum as well as maximum circuit delays play vital roles in high performance systems. In this paper, we analyze existing and new minimum delay models and show that these models are special cases of a general circuit delay model introduced in [LBSV92a] which also unifies all maximum delay models. Then, we provide algorithms to compute exactly various minimum circuit delays with arbitrary gate delay models under the same framework. Thus, all minimum and maximum circuit delay models are unified and the their correponding delays can be computed exactly. Further, we consider the interactions of various minimum and maximum delays with clock frequencies of circuits. Finally, we provide experimental results on ISCAS benchmarks.

1 Introduction

Much research in performance analysis of combinational circuits has been focused on the computation of the longest delays under various delay models, for instance, viability delays [MB89], floating delays [CD90, DKM91], transition delays [DKM92, LBSV92b], and delays by sequences of vectors [LBSV92a]. Further investigations of high performance design reveal that not only longest delays but also short delays play a vital role. Particularly, in wavepipelining designs where "waves" of input data are fed into a circuit in such short intervals that multiple data co-exist in the circuit simultaneously, the time intervals between data waves in which the circuit computes correctly are determined by both the longest and the shortest paths of the circuit, [LBSV92c]. In this paper, we propose two short delay models which are derivatives of a general delay model in [LBSV92a] and study their exact computations using Timed Boolean Functions (TBF's). We show that the minimum delay model, destabilizing delay, proposed in [CCD92] is the same as one of our proposed model under practical situations. Then, we give theorems exhibiting the effects of minimum and maximum delays on circuit operating speeds under all circuit delay models. Finally, we provide experimental results on ISCAS benchmarks.

2 Why Minimum Delay?

In conventional clocking schemes, data are applied to circuits in such intervals that the present data have propagated through the circuits, at least along the "true paths", before the succeeding data

*Supported by Fannie and John Hertz Foundation and SRC under contract 92-DC-008.

are supplied. That is, the clock intervals are equal to the lengths of true paths. However, in fact, succeeding data can be applied much earlier as long as the succeeding data do not interfere with the computation by the present data. Suppose two paths converge to a gate, with the first path longer than the second. When the present data are propagating along the two paths to the gate, error will result if the succeeding data arrive the gate along the shorter path earlier than the present data arrive the gate along the longer path. Therefore, if the interval between data is greater than the length difference of the two paths, the succeeding data will never catch up its previous data along the shorter path; then, there is a time interval where the output of the gate gives the correct computation. Therefore, if the intervals between consecutive data are greater than the path length differences at all gates in a circuit and the output of the circuit is sampled at the correct times, the circuit computes correctly. Computation in this mode is generally referred as wavepipelining, [Cot69]. [LBSV92c] gives the following relationship between the correct clocking intervals and the path lengths in a circuit.

THEOREM 1 *Let L^{max} and L^{min} be the longest and the shortest topological path lengths of a circuit. The clocking intervals in which the outputs of the circuit are valid are given by*

$$\frac{L^{max}}{H} \leq \tau < \frac{L^{min}}{H-1}$$

where H is some integer. For a specific interval $\frac{L^{max}}{H} \leq \tau < \frac{L^{min}}{H-1}$, the output of the circuit computes the functionality of the circuit delayed by H cycles.

If the longest path and the shortest path are close in length, the clocking intervals can be orders of magnitude smaller than the longest path length; thus, circuits with closely matched paths can operate at very high frequencies. Short paths play an important role in high performance systems.

3 Previous Works

Besides the topological shortest path model, there are currently two short delay models: minimum destabilizing delay [CCD92] and minimum 2-vector delay [LBSV92c]. The minimum destabilizing delay is a single vector delay similar to its longest delay counterpart, floating delay. The minimum destabilizing delay is defined as follows. A circuit is initially stable under a vector, then the nodes of the circuit take on arbitrary values, possibly under the influence of inputs applied later. The earliest time at which the output of the circuit becomes unstable under all possible initial vectors is the minimum destabilizing delay. In [CCD92], a sufficient condition is given in terms of the floating delay and the minimum destabilizing delay of a circuit under which the circuit will function correctly. In minimum 2-vector delay, a circuit is assumed to have settled under a first input vector and a second vector is applied; the earliest time of the first output transition is the minimum 2-vector delay. In [LBSV92c], a sufficient and necessary condition is given in terms of the maximum 2-vector delay (transition delay) and the minimum 2-vector delay of a circuit under which the circuit will function correctly.

Although these two short delay models seem unrelated, they can be regarded as two special cases of a general delay model proposed in [LBSV92a] in which a circuit's delay model consists of three basic elements: the circuit's structure, the gate delay models, and the family of input signals to the circuit.

4 A General Circuit Delay Model

Here we review the general circuit delay model proposed in [LBSV92a]. The idea of this general model is to identify the constituent elements of circuit delays. We identify the three elements as: circuit's structure, gate delay models, and family of input signals. Different combinations of these elements give rise to different circuit delay models. For instance, transition delay is the delay model with the family of input being pairs of vectors, and floating delay is, under practical situations, the delay model with the family of input being sequences of vectors. Similarly, minimum destabilizing delay and minimum 2-vector delay are also derivatives of this general circuit delay model. The following definition of the general circuit delay model is taken from [LBSV92a].

DEFINITION 1 (A General Delay Model) *1. Let circuit C be a connection of gates with delay model $\mathcal{M}_g$. and $\mathcal{I}$, the family of inputs to C (assume that circuit C has already settled before the applications of $\mathcal{I}$). Then, the circuit delay model for interval type is: $[D^{min}(C,\mathcal{M}_g, \mathcal{I}), D^{max}(C, \mathcal{M}_g, \mathcal{I})]$, where D^{min} is the earliest arrival time of the last output transition and D^{max}, the latest arrival time of the last output transition.*

2. For $\mathcal{I} \in \{\omega^-, \omega^+, 2\}$, where ω^- symbolizes sequences of vectors applied at $t \leq 0$ with the last vector at $t = 0$, ω^+, sequences of vectors applied at $t \geq 0$ with the first vector at $t = 0$, and the 2, a pair of vectors. we define:

- *$D^{max}(C, \mathcal{M}_g, \omega^-)$: the latest arrival time of the last output transition, when a sequence of arbitrary number of vectors of arbitrary intervals between vectors is applied to the inputs at $t \leq 0$, and the last vector is applied at $t = 0$. $D^{min}(C, \mathcal{M}_g, \omega^-)$ is the same as $D^{max}(C, \mathcal{M}_g, \omega^-)$ except that it is the earliest arrival time of the last output transition.*
- *$D^{max}(C, \mathcal{M}_g, \omega^+)$: the latest arrival time of the first output transition, when a sequence of arbitrary number of vectors of arbitrary intervals between vectors is applied to the inputs at $t \geq 0$, and the first vector is applied at $t = 0$. $D^{min}(C, \mathcal{M}_g, \omega^+)$ is the same as $D^{max}(C, \mathcal{M}_g, \omega^+)$ except that it is the earliest arrival time of the first output transition.*
- *$D^{max}(C, \mathcal{M}_g, 2)$: the latest arrival time of the last output transition, when a pair of vectors are applied with the first vector applied at $t = -\infty$, the second vector, at $t = 0$. $D^{min}(C, \mathcal{M}_g, 2)$ is the earliest arrival time of the first output transition under the same setting.*

3. The gate delay models considered in this paper are: $\mathcal{M}_g \in \{[d_i^{min}, d_i^{max}], [d_i^{max}, d_i^{max}], [0, d_i^{max}]\}$, commonly referred as bounded, fixed, unbounded delay models, respectively.

Thus, minimum 2-vector delay is $D^{min}(C, M_g, 2)$, or simply $D^{min}(M_g, 2)$. We call $D^{min}(C, M_g, \omega^+)$ the minimum delay by sequences of vectors, which will be shown to be equal to the minimum destabilizing delay under practical situations.

5 Destabilizing Delay and $D^{min}(M_g, \omega^+)$

We first review destabilizing delay. Initially a circuit is assumed to have settled on an input vector v, then inputs are applied to the circuit. To be conservative, the inputs can cause the nodes in the circuit to take on arbitrary values. The destabilizing time of a node is the time for the node's value to change from a stable value to an unstable value. Thus, given destabilizing times of inputs of a gate, the destabilizing time of the gate is the delay of the gate plus either the earliest destabilizing time of its inputs if all inputs are non-controlling or the latest destabilizing time of its controlling inputs. That is, if all inputs of the gate have non-controlling stable values then the first change at any of the inputs will cause the gate's output to change; or if some inputs have controlling stable values, the gate's output will change only when all the controlling inputs have changed, assuming conservatively that when the last controlling input changes all other inputs are non-controlling. An input f to a gate G destabilizes G if either f has the earliest destabilizing time if all inputs of G are non-controlling or f is the controlling input with the latest destabilizing time. A path is destabilizing if each lead on the path destabilizes its succeeding gate under some vector. The destabilizing delay of a circuit is the length of the minimum destabilizing path. In contrast to floating delay, the minimum delays of gates, instead of the maximum delays of gates, are used in the computation of minimum destabilizing delays.

Now we consider minimum delay by sequences of vectors $D^{min}(M_g, \omega^+)$. A circuit is initially settled and an arbitrary sequence of vectors are applied. The time of the first output transition is $D^{min}(M_g, \omega^+)$. As will be seen later destabilizing delay is equal to minimum delay by sequences of vectors under practical situations; that is, the conservative assumption that nodes can take on arbitrary values in destabilizing delay is not conservative at all. The proof follows the same lines as the proof that floating delay is equal to the maximum delay by sequences of vectors under practical situations in [LBSV92a]. So, we will state the results here without proof.

THEOREM 2 *If every gate in a circuit has variable delay, i.e.* $d^{min} \neq d^{max}$, *and no two distinct paths have the same set of gates, then, destabilizing delay* $= D^{min}([d^{min}, d^{max}], \omega^+)$.

Proof. Omitted. ■

In practical situations, gates have a range of delay values due to manufacturing processes; so the assumption $d^{min} \neq d^{max}$ is not too restrictive.

6 Formulation of Minimum Delay Computation Using TBF's

We use Timed Boolean Functions (TBF's) to formulate the problems of computing minimum delay and give exact solutions. With TBF's, both minimum delays can be formulated similarly under various gate delay models. First, review the definition and some properties of TBF's.

7 Timed Boolean Function

DEFINITION 2 [1]

[1] This definition is a refined version of that in [LBSV92c]

1. *A waveform space $B(t)$ is a collection of mappings $f : R \mapsto B$, $B = \{0, 1\}$.*

2. *A Timed Boolean Function (TBF) is any function with domain $B^n(t)$ and range $B(t)$. For analysis on most digital circuits, the following subset of TBF's is sufficient.*

   TBF $F : B^n(t) \mapsto B(t)$, is defined recursively as follows.

   - *The identity function F (i.e. $F(v)(t) = v(t)$, $v(t) \in B(t)$) is a TBF.*
   - *If $G(t) : B^{n_1}(t) \mapsto B(t)$ and $H(t) : B^{n_2}(t) \mapsto B(t)$ are TBF's, then, $\overline{G}(t), G(t) \cdot H(t), G(t) + H(t)$ are also TBF's.*
   - *If $F(t)$ is a TBF, then, for any function $\phi : R \mapsto R$, $F(\phi(t))$ is also a TBF.*

Note that TBF is a natural generalization of Boolean function whose domain and range are restricted to constant mappings in $B(t)$.

Example 1 *Let x, $y \in W$ be the waveforms shown in Figure 1(a) and 1(b); then the TBF $f(a, b)(t) = \overline{a}(t-1) \oplus b(t+1)$ represents the waveform shown in Figure 1(c) if a=x, b=y.*

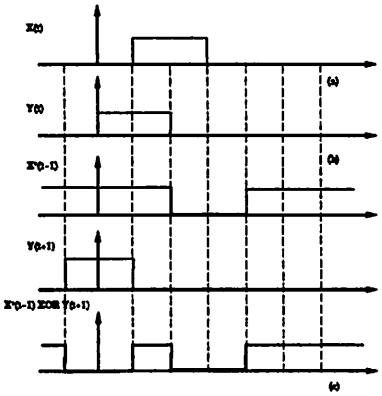

Figure 1: Representing Waveforms by TBF

7.1 Modeling Timing Behavior with TBFs

Before representing a circuit by a TBF, each component of the circuit needs to be modeled by a TBF.

Here, we only illustrate through examples the modeling process for some commonly encountered gates.

1. Gates characterized by a single delay for each input-output pair. The complex gate shown in Figure 2(a) has three inputs; input x_i has a delay τ_i to the output. This gate is modeled with the TBF:

$$y(t) = \overline{x}_1(t - \tau_1) + x_2(t - \tau_2) + x_3(t - \tau_3).$$

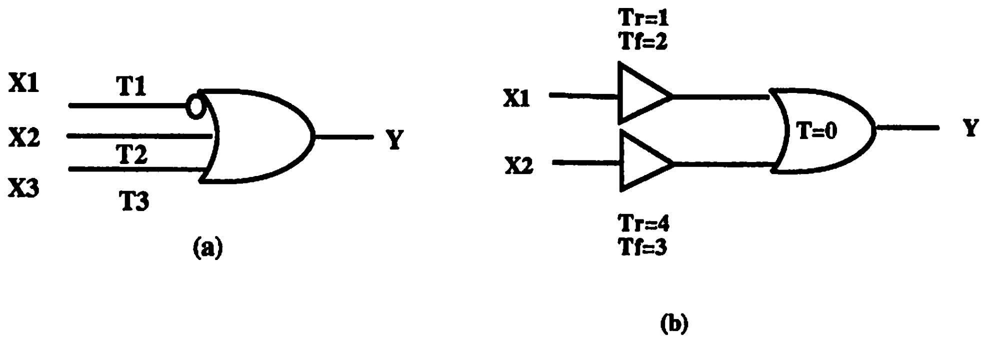

Figure 2: Modeling With TBF

2. Buffers with different rising and falling delays. Let τ_r and τ_f be the rising and falling delays, respectively. If $\tau_r > \tau_f$, then the buffer can be modeled as:

$$y(t) = x(t - \tau_r) \cdot x(t - \tau_f).$$

and if $\tau_r < \tau_f$, the buffer can be modeled as:

$$y(t) = x(t - \tau_r) + x(t - \tau_f).$$

3. Gates with different rising and falling delays for each input-output pair. Rising (falling) delay is the delay when the output is rising (falling). Each input is modeled by a buffer with different rising and falling delays; the ”functional block” assumes zero delay. The overall TBF for the gate is obtained through the usual functional composition. An example of an OR gate is shown in Figure 2(b). Input 1 has a rising delay of 1 and a falling delay of 2, while input 2 has a rising delay of 4 and a falling delay of 3. The buffer modeling input 1 is

$$x_1(t-1) + x_1(t-2)$$

and input 2 is

$$x_2(t-4) \cdot x_2(t-3).$$

Therefore, the OR gate is

$$x_1(t-1) + x_1(t-2) + x_2(t-4) \cdot x_2(t-3).$$

A common problem in digital circuit design is pulse shrinkage or dilation. Pulse shrinkage (dilation) effects occur when a pulse passes through a chain of gates with unequal rising and falling delays; the pulse width becomes narrower (wider) at the end of the chain. With the above modeling technique, this effect is captured.

Once each gate of a circuit is modeled, the TBF for each node in terms of primary inputs can be obtained by composing the gate model's TBF with the TBF's of the fanins of that gate. When the TBF of a circuit output is found, the circuit's behavior at any time can be calculated from the TBF. Suppose $f(t, x_1, ..., x_n, d_1, ..., d_m)$ is the TBF of an output of a circuit, where $x_1, ..., x_n$ are the primary inputs, and $d_1, ..., d_m$ are the delays of gates in the circuit. The value of the output at $t = k$ is given by $f(k, x_1, ..., x_n, d_1, ..., d_m)$.

8 Formulation of Minimum Delay Computation

In minimum delay by sequences of vectors, assume the sequence is applied at $t = 0$; so the input before $t = 0$ is some constant vector, $v^0 = (v_1^0, \ldots, v_n^0)$. After $t = 0$, the sequence of vectors are arbitrary. Let $f(t, x_1, \ldots, x_n, d_1, \ldots, d_m)$ be a TBF of a circuit. Then the minimum delay by sequences of vectors is the minimum t such that $f(t, x_1, \ldots, x_n, d_1, \ldots, d_m)$ is not equal to $f(-\infty, x_1, \ldots, x_n, d_1, \ldots, d_m)$. Symbolically,

$$\min t$$

$$f(t, x_1, \ldots, x_n, d_1, \ldots, d_m) \neq f(-\infty, x_1, \ldots, x_n, d_1, \ldots, d_m)$$
$$d_i^{min} \leq d_i \leq d_i^{max}$$
$$x_i(\tau) = v_i^0, \; \tau \leq 0$$

For 2-vector minimum delay, assume the first vector $v^1 = (v_1^1, \ldots, v_n^1)$ is applied at $t = -\infty$ and the second vector $v^2 = (v_1^2, \ldots, v_n^2)$ is applied at $t = 0$. Therefore, its formulation is:

$$\min t$$

$$f(t, x_1, \ldots, x_n, d_1, \ldots, d_m) \neq f(-\infty, x_1, \ldots, x_n, d_1, \ldots, d_m)$$
$$d_i^{min} \leq d_i \leq d_i^{max}$$
$$x_i(\tau) = v_i^1, \; \tau \leq 0$$
$$x_i(\tau) = v_i^2, \; \tau \geq 0$$

We call this formulation a mixed Boolean linear programming problem, because there are Boolean constraints, e.g. $f(t, x_1, \ldots, x_n, d_1, \ldots, d_m) \neq f(-\infty, x_1, \ldots, x_n, d_1, \ldots, d_m)$, as well as linear constraints, e.g. $d_i^{min} \leq d_i \leq d_i^{max}$.

The semantics of this mixed Boolean linear programming is illustrated by the following example.

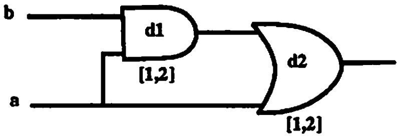

Figure 3: Example for Mixed Boolean Linear Programming

Example 2 *Refer to Figure 3. Assume the input signals are a pair of vectors switching simultaneously at $t = 0$. Let $(a(0^-), b(0^-))$ be the vector applied from $-\infty$ to 0, (a, b), the vector applied at $t = 0$. The TBF of the circuit is:*

$$f(t, a, b, d_1, d_2) = a(t - d_2) + a(t - d_1 - d_2)b(t - d_1 - d_2)$$

The static logical function is:

$$f(-\infty, a, b, d_1, d_2) = a(0^-) + a(0^-)b(0^-) = a(0^-)$$

The ranges of the gate delay variables are $[1,2]$. *The TBF variables can take on either the input vector applied before time 0 or after time 0. For example, TBF variable* $a(t-d_2)$ *equals to* $a(0^-)$ *if* $t < d_2$ *or equals to* a *if* $t > d_2$. *Any assignment of TBF variables to their respective before or after values* $x(0^-)$ *or* x *induces an associated linear programming problem. For instance, the Boolean assignment to the TBF variables:*

$$\begin{aligned} a(t-d_2) &= a \\ a(t-d_1-d_2) &= a(0^-) \\ b(t-d_1-d_2) &= b(0^-) \end{aligned}$$

induces following linear programming problem:

$$\begin{array}{rcl} \min & & t \\ t-d_2 & > & 0 \\ t-d_1-d_2 & < & 0 \\ 1 \le d_1 & \le & 2 \\ 1 \le d_2 & \le & 2 \end{array}$$

The minimum is 1. *For this Boolean assignment, the TBF becomes*

$$a + a(0^-)b(0^-)$$

which is not equal to the static function $a(0^-)$. *Since the shortest topological path length is also* 1, *the minimum* t *from the linear programming is indeed the minimum 2-vector delay of the circuit. Therefore, finding the delay of a circuit is equivalent to finding a Boolean assignment to TBF variables such the induced linear program gives the minimum* t *and the TBF function evaluated at the assignment is not equal to the static logical function.*

9 Computation of Exact Minimum Delays

In this section, we demonstrate a strength of formulating delay computations in TBF's: both $D^{min}(M_g, \omega^+)$ (destabilizing delay) and $D^{min}(M_g, 2)$ (minimum 2-vector delay) can be computed under the same framework. In [LBSV92a], this advantage is also illustrated for maximum delay computations for which previous approaches require separate algorithms for different delay models. The computation of minimum delays parallels to that of maximum delays; thus, we will briefly review the keys of maximum delay computation using TBF's and point out the differences for the case of minimum delay computation.

9.1 Computation of Maximum Delays: a Review

Let $f(.)$ be a TBF, $x_1, \ldots, x_n$, the inputs, $d_1, \ldots, d_m$, the gate delays of a circuit, respectively. Assuming $d_i^{min} \neq d_i^{max}$, computation of $D^{max}(M_g, \omega^-)$, (viability delay, floating delay, and delay by sequences of vectors) of the circuit is formulated as:

$$\begin{array}{rcl} D^{max}(M_g, \omega^-) & = & \max t \\ f(t, x_1, \ldots, x_n, d_1, \ldots, d_m) & \neq & f(\infty, x_1, \ldots, x_n, d_1, \ldots, d_m) \\ d_i^{min} \leq & d_i & \leq d_i^{max} \\ x_i(t) = v_i & & t \geq 0 \end{array}$$

where $(v_1, \ldots, v_n)$ is a constant vector.

The algorithm to solve the above mixed Boolean linear programming is as follows. Associate *implicitly* a pair (α, β) with each path $\pi = (d_1, \ldots, d_p)$ and its corresponding TBF variable $x_i(t-k_i)$, where $k_i = \sum_{i=1}^{p} d_i$, $\alpha = \sum_{i=1}^{p} d_i^{min}$, and $\beta = \sum_{i=1}^{p} d_i^{max}$. It is shown in [LBSV92a] that the maximum for the above programming can be found by searching t at the β's only, starting from the largest β. At each search value of t, a network called TBF network is constructed to represent the TBF evaluated at t. At a particular value of t, the TBF at t is evaluated as follows. The TBF variable $x_i(t-k_i)$ is set to v_i if $t \geq \beta$, and set to a new Boolean variable, otherwise. When the TBF network at t is constructed, its BDD is compared with that of the original network, i.e. checking

$$f(t, x_1, \ldots, x_n, d_1, \ldots, d_m) \neq f(\infty, x_1, \ldots, x_n, d_1, \ldots, d_m)$$

The maximum delay is the first t at which the inequality holds. Interestingly, only the upper bounds d_i^{max}'s matter.

Computation of $D^{max}(M_g, 2)$, maximum 2-vector delay or transition delay, is formulated as:

$$\begin{array}{rcl} D^{max}(M_g, 2) & = & \max t \\ f(t, x_1, \ldots, x_n, d_1, \ldots, d_m) & \neq & f(\infty, x_1, \ldots, x_n, d_1, \ldots, d_m) \\ d_i^{min} \leq & d_i & \leq d_i^{max} \\ x_i(t) & = & \left\{ \begin{array}{ll} v_i^1 & t < 0 \\ v_i^2 & t \geq 0 \end{array} \right. \end{array}$$

where $(v_1^1, \ldots, v_n^1)$ and $(v_1^2, \ldots, v_n^2)$ are constant vectors.

Again, the maximum is found by searching t at the β's. At a particular value of t, the TBF at t is evaluated as follows. The TBF variable $x_i(t-k_i)$ is set to v_i^1 if $t < \alpha$, set to v_i^2 if $t \geq \beta$, otherwise set to $y_i = s_x v_i^2 + \overline{s}_x v_i^1$, where s_x is a new Boolean variable that selects either v_i^1 or v_i^2, because in this case $t - k_i$ can be either negative or positive, i.e. $x_i(t-k_i)$ can be either v_i^1 or v_i^2. Then, the BDD's of the TBF network and the original network are compared. If they are not equal, "cubes" of the difference BDD are enumerated to give sets of inequalities on paths. The sets of inequalities together with $d_i^{min} \leq d_i \leq d_i^{max}$ are checked for feasibility. If all cubes are infeasible, the search continues; otherwise, the maximum is found.

9.2 Computation of Minimum Delays

The computation of minimum delays can also be formulated using TBF's. The computation of $D^{min}(M_g, \omega^+)$, destabilizing delay or minimum delay by sequences of vectors, is given by:

$$\begin{array}{rcl} D^{max}(M_g, \omega^+) & = & \min t \\ f(t, x_1, \ldots, x_n, d_1, \ldots, d_m) & \neq & f(-\infty, x_1, \ldots, x_n, d_1, \ldots, d_m) \\ d_i^{min} \leq & d_i & \leq d_i^{max} \\ x_i(t) = v_i & & t \leq 0 \end{array}$$

where $(v_1, \ldots, v_n)$ is a constant vector.

This formulation can be seen similar to that of maximum delay computation by replacing t by $-\tau$. Thus,

$$\begin{array}{rcl} D^{max}(M_g, \omega^+) & = & \max \tau \\ f(-\tau, x_1, \ldots, x_n, d_1, \ldots, d_m) & \neq & f(\infty, x_1, \ldots, x_n, d_1, \ldots, d_m) \\ d_i^{min} \leq & d_i & \leq d_i^{max} \\ x_i(-\tau) = v_i & & \tau \geq 0 \end{array}$$

The algorithm for solving the programming is similar to that of maximum delay. Instead of searching t on β's starting from the largest β, search t on α's starting from the least α. A TBF variable $x_i(t - k_i)$ is evaluated to v_i if $t < \alpha$, and to a new Boolean variable, otherwise. The rest of the algorithm is the same.

Computation of $D^{min}(M_g, 2)$ is formulated as:

$$\begin{array}{rcl} D^{min}(M_g, 2) & = & \min t \\ f(t, x_1, \ldots, x_n, d_1, \ldots, d_m) & \neq & f(-\infty, x_1, \ldots, x_n, d_1, \ldots, d_m) \\ d_i^{min} \leq & d_i & \leq d_i^{max} \\ x_i(t) & = & \begin{cases} v_i^1 & t < 0 \\ v_i^2 & t \geq 0 \end{cases} \end{array}$$

where $(v_1^1, \ldots, v_n^1)$ and $(v_1^2, \ldots, v_n^2)$ are constant vectors.

Again the search is performed on α's. And a TBF variable $x_i(t - k_i)$ is evaluated to v_i^1 if $t - k < 0$, to v_i^2 if $t - k > 0$. The rest of the algorithm is the same.

10 Minimum, Maximum Delays and Clock Frequencies

The results so far in this paper and those in [LBSV92a] show that the delays $D^{min}(M_g, \omega^+)$, $D^{min}(M_g, 2)$, $D^{max}(M_g, \omega^-)$, and $D^{max}(M_g, 2)$ can be formulated using TBF's and computed under the same framework; thus, viability, floating, transition, destabilizing, and 2-vector delays are unified. Here, we illustrate how these delays interact to affect the performances of synchronous systems. As discussed earlier, topological longest and shortest paths limit operating speeds due to interferences of signals. The interactions between the longest and the shortest paths under the 2-vector sensitization criterion are stated by a theorem in [LBSV92c], which we repeat below.

Let $f[n-H]$ be the functionality of a circuit delayed by H cycles.

THEOREM 3 *If the interval* $[\frac{L^{max}}{H}, \frac{L^{min}}{H-1})$ *is non-empty, the valid clocking interval is*

$$\max(\frac{D^{max}(M_g,2)}{H}, \frac{L^{max}}{H+1}) \leq \tau < \min(\frac{D^{min}(M_g,2)}{H-1}, \frac{L^{min}}{H-2}),\ for\ H > 1$$

and

$$\max(D^{max}(M_g,2), \frac{L^{max}}{2}) \leq \tau,\ for\ H = 1$$

The computed function at τ *is* $f[n-H]$.

Proof. See [LBSVon]. ■

The expressions for the intervals are similar to that for topological paths (see Theorem 1), except for the *max* and *min* functions. The *max* and *min* function limit the ranges of the intervals so that the analyses of $D^{min}(M_g,2)$ and $D^{max}(M_g,2)$ remain valid within the intervals.

Now, we present a theorem relating $D^{min}(M_g,\omega^+)$ and $D^{max}(M_g,\omega^-)$ to valid clocking intervals. $D^{min}(M_g,\omega^+)$ and $D^{max}(M_g,\omega^-)$ have relatively simple computational algorithms and are widely used in delay estimations.

THEOREM 4 *If the interval* $[\frac{L^{max}}{H}, \frac{L^{min}}{H-1})$ *is non-empty, the valid clocking interval is*

$$\frac{D^{max}(M_g,\omega^-)}{H} \leq \tau < \frac{D^{min}(M_g,\omega^+)}{H-1}$$

The computed function at τ *is* $f[n-H]$.

Proof. Observe that $D^{max}(M_g,\omega^-) \leq L^{max}$ and $D^{min}(M_g,\omega^+) \geq L^{min}$. If the interval $[\frac{L^{max}}{H}, \frac{L^{min}}{H-1})$ is non-empty, the interval $[\frac{D^{max}(M_g,\omega^-)}{H}, \frac{D^{min}(M_g,\omega^+)}{H-1})$, can be splited into three sub-intervals: $[\frac{D^{max}(M_g,\omega^-)}{H}, \frac{L^{max}}{H}]$, $[\frac{L^{max}}{H}, \frac{L^{min}}{H-1})$, and $[\frac{L^{min}}{H-1}, \frac{D^{min}(M_g,\omega^+)}{H-1})$.

Obviously, the second interval is valid. Need to show the first and the third intervals are also valid. Here, we show the third interval is valid. Let $f(\ldots, x(n+\left\lfloor\frac{-k}{\tau}\right\rfloor), \ldots)$ be a TBF of the circuit. We want to show that for $\tau \in [\frac{L^{min}}{H-1}, \frac{D^{min}(M_g,\omega^+)}{H-1})$, $f(\ldots, x(n+\left\lfloor\frac{-k}{\tau}\right\rfloor), \ldots) = f[n-H]$. Consider the TBF variables such that $x(n+\left\lfloor\frac{-k}{\tau}\right\rfloor) \neq x(n-H)$. Then, $n+\left\lfloor\frac{-k}{\tau}\right\rfloor \geq n-H+1$; equivalently, $(H-1)\tau \geq k$. Therefore, these variables correspond to the variables with $t \geq k$ in $f(\ldots, x(t-k), \ldots)$, where $t = (H-1)\tau$. The interval $[\frac{L^{min}}{H-1}, \frac{D^{min}(M_g,\omega^+)}{H-1})$ implies $L^{min} \leq (H-1)\tau < D^{min}(M_g,\omega^+)$; or $L^{min} \leq t < D^{min}(M_g,\omega^+)$. By the definition of $D^{min}(M_g,\omega^+)$, the variables with $t \geq k$ are not in the support of $f(\ldots, x(t-k), \ldots)$. Therefore, the variables such that $x(n+\left\lfloor\frac{-k}{\tau}\right\rfloor) \neq x(n-H)$

are not in the support of $f(\ldots, x(n + \lfloor \frac{-k}{\tau} \rfloor), \ldots)$. Hence, $f(\ldots, x(n + \lfloor \frac{-k}{\tau} \rfloor), \ldots) = f[n - H]$. The interval $[\frac{L^{min}}{H-1}, \frac{D^{min}(M_g, \omega^+)}{H-1})$ is valid.

The proof for the interval $[\frac{D^{max}(M_g, \omega^-)}{H}, \frac{L^{max}}{H}]$ is analogous. ■

It is interesting to see this theorem has the exact form as Theorem 1. If the topological interval $[\frac{L^{max}}{H}, \frac{L^{min}}{H-1})$ is not empty, a wider interval can be easily obtained by replacing L^{max} and L^{min} by $D^{max}(M_g, \omega^-)$ and $D^{min}(M_g, \omega^+)$, respectively.

11 Balancing Circuits

From our experimental results on ISCAS benchmarks, all the circuits have very different the longest and the shortest paths, which are not amicable for wavepipelining. We use a very simple algorithm to balance the circuits to make the long and short paths close. The balancing algorithm is as follows.

1. Sort nodes in a topological order so that a node appears later than all of its fanins.
2. For each node, compute the lengths of the longest and the shortest partial paths from the node to the primary inputs.
3. For a node n, consider all its fanins that do not have the longest partial paths from n. Resize the gates of the fanins to increase their lengths. If resizing of some fanins affect some longest paths in the circuit, add delays to the outputs of these fanins instead of resizing.

The complexity of the above algorithm is linear in the number of connections of the circuit.

12 Experimental Results

We implemented the algorithms for computing the exact minimum 2-vector delays and delays by sequences of vectors. The following ISCAS benchmarks were run on a DECstation 5000 (38 mips) with a standard sis script and then mapped to the mcnc library. The actual delay values used are the ones given in the library. The minimum gate delay is assumed to be 80% of the maximum gate delay. L^{max} is the longest topological path length, $D^{min}(2)$, an abbreviation for $D^{min}([d^{min}, d^{max}], 2)$, and $D^{min}(\omega^+)$, abbreviation for $D^{min}([d^{min}, d^{max}], \omega^+)$. After the minimum and maximum delays of the benchmarks were computed, the circuits were balanced using the balancing algorithm. The first table shows the minimum and the maximum delays of the benchmarks before and after balancing. The CPU times are the average CPU times in computing the delays.

	Before Balance			After Balance			
Circuit	L^{max}	$D^{min}(2)$	$D^{min}(\omega^+)$	L^{max}	$D^{min}(2)$	$D^{min}(\omega^+)$	CPU (sec)
C432	39	1.68	1.68	39	31.2	31.2	50
C499	28.9	1.28	1.28	28.9	23.12	23.12	4.35
C880	36.4	0.88	0.88	36.4	29.12	29.12	0.35
C1355	28.9	1.28	1.28	28.9	23.12	23.12	4.29
C1908	41.1	1.28	1.28	41.1	32.88	32.88	1.56
C2670	36.5	0	0	36.5	29.2	29.2	0.76
C3540	51.2	0.8	0.8	53.1	42.48	42.48	0.86
C5315	51.6	0	0	51.6	41.28	41.28	1.38
C6288	140	1.52	1.52	140	112	112	4.31
C7552	59.8	0	0	59.8	47.84	47.84	1.66

The following table gives the valid clocking intervals for the balanced circuits. The balancing algorithm was able to balance perfectly for all circuits, i.e. all paths have the same length. In these balanced circuits, all the maximum delays, e.g. longest topological path lengths, 2-vector delays, floating delays, are equal; thus, the maximum delays are simply denoted by D^{max}. Similarly for the minimum delays.

Circuit	D^{max}	D^{min}	Valid Intervals	H
C432	39	31.2	[39, ∞),[19.5,31.2),[13,15.6),[9.7,10,4)	1,2,3,4
C499	28.9	23.1	[28.9, ∞),[14.5,23.2),[9.6,11.6),[7.2,7.7)	1,2,3,4
C880	36.4	29.1	[36.4, ∞),[18.2,29.1),[12.1,14.6),[9,9.7)	1,2,3,4
C1355	28.9	23.1	[28.9, ∞),[14.5,23.1),[9.6,11.6),[7.2,7.7)	1,2,3,4
C1908	41.1	32.8	[41.1, ∞),[20.6,32.8),[14,16.4),[10,11)	1,2,3,4
C2670	36.5	29.2	[36.5, ∞),[18.3,29.2),[12.2,14.6),[9.1,9.7)	1,2,3,4
C3540	53.1	42.4	[53.1, ∞),[26.6,42.4),[17.7,21.2),[13.2,14.1)	1,2,3,4
C5315	51.6	41.2	[51.6, ∞),[25.8,41.2),[17.2,20.6),[12.9,13.7)	1,2,3,4
C6288	140	112	[140, ∞),[70,112),[46.6,56),[35,37.3)	1,2,3,4
C7552	59.8	47.8	[59.8, ∞),[29.9,47.8),[20,23.9),[15,15.9)	1,2,3,4

As can be seen, after balancing, the circuits can operate at speeds as much as four times faster.

13 Conclusion

In this paper, we analyzed existing and new minimum delay models and showed that these models are special cases of a general circuit delay model introduced in [LBSV92a] which also unifies all maximum delay models. Then, we provided algorithms to compute exactly various minimum circuit delays with arbitrary gate delay models under the same framework. Thus, all minimum and maximum circuit delay models are unified and the their correponding delays can be computed exactly. Further, we considered the interactions of various minimum and maximum delays with clock frequencies of circuits. Finally, we provided experimental results on ISCAS benchmarks.

References

[CCD92] S. Chen, H. Chen, and D. Du. The role of long and short paths in circuit performance optimization. *29th ACM/IEEE Design Automation Conference*, 1992.

[CD90] H. C. Chen and D. H. Du. Path sensitization in critical path problem. *1990 ACM Workshop on Timing Issues in the Specification and Synthesis of Digital Systems*, 1990.

[Cot69] L. Cotten. Maximum rate pipelined systems. *1969 AFIPS Proceedings of Spring Joint Computer Conference*, 1969.

[DKM91] S. Devadas, K. Keutzer, and S. Malik. Delay computation in combinational logic circuits: Theory and algorithms. *IEEE/ACM International Conference on Computer-Aided Design*, Nov. 1991.

[DKM92] S. Devadas, K. Keutzer, and S. Malik. Certified timing verification and transition delay of a logic circuit. *Proc. of the 29th Design Automation Conference*, June, 1992.

[Kog81] P. Kogge. *The Architecture of Pipelined Computers.* Prentice-Hall, Englewood Cliffs, N.J., 1981.

[LB92] W. Lien and W. Burleson. Wave-domino logic: Timing analysis and applications. *1992 ACM Workshop on Timing Issues in the Specification and Synthesis of Digital Systems*, 1992.

[LBSV92a] W. Lam, R. Brayton, and A. Sangiovanni-Vincentelli. Circuit delay models and their exact computation using timed boolean functions. *IEEE/ACM Design Automation Conference '93*, 1992.

[LBSV92b] W. Lam, R. Brayton, and A. Sangiovanni-Vincentelli. Exact delay computation with timed boolean function. *UC Berkeley ERL memorandum: UCB/ERL M92/57*, May 1992.

[LBSV92c] W. Lam, R. Brayton, and A. Sangiovanni-Vincentelli. Valid clocking in wavepipelined circuits. *IEEE/ACM International Conference on Computer Aided Design, '92*, Nov., 1992.

[LBSVon] W. Lam, R. Brayton, and A. Sangiovanni-Vincentelli. Valid clock frequencies and their computation in wavepipelined circuits. *IEEE transaction on CAD*, 1993, (will submitted for publication).

[MB89] P. McGeer and R. Brayton. Provably correct critical paths. *The Proceedings of the Decennial Caltech VLSI Conference*, 1989.

[Ung69] Stephen H. Unger. *Asynchronous Sequential Switching Circuits.* John Wiley and Sons, Inc., 1969.