

Voltage-Aware Static Timing Analysis

Dionysios Kouroussis, *Member, IEEE*, Rubil Ahmadi, *Member, IEEE*, and
Farid N. Najm, *Fellow, IEEE*

Abstract—Static timing analysis (STA) techniques allow a designer to check the timing of a circuit at different process corners, which typically include corner values of the supply voltages as well. Traditionally, however, this analysis only considers cases where the supplies are either all low or all high. As will be demonstrated, this may not yield the true maximum delay of a circuit because it neglects the possible mismatch between the supplies of successive gates on a path. A new methodology for timing analysis is proposed, where, in a first step, the critical paths of a circuit are identified under an assumption that all the supply nodes are independent of one another, thus allowing for mismatch between the supplies. Then, given these critical paths, the authors incorporate into the analysis the relationships between the supply node voltages by considering the power grid that they are tied to, and refine the worst case time delay values on a per-critical-path basis. This refinement is posed as a sequence of optimization problems where the operation of the circuit is abstracted in terms of current constraints. The authors present their technique and report on the implementation results using benchmark circuits tied to a number of test-case power grids.

Index Terms—Power grid, rail voltage variations, static timing analysis, verification tools.

I. INTRODUCTION

DUE to the reduction in supply voltages, resulting from technology scaling, the timing of modern integrated circuits has become highly sensitive to supply voltage fluctuations. Thus, in the analysis and verification of high-performance chips, it is essential that static timing analysis (STA) takes into account power supply variations. Traditionally, this has been done by performing STA with a setting of the supply voltages that results in worst case delay for each gate on the path under study. However, we have found that using worst case gate delays in the context of traditional STA does not necessarily yield the worst case path delay. This is due to the fact that mismatch between the supply settings of successive gates on a path turns out to have a bigger effect on the worst case path delay, as we will show. Therefore, it emerges that one really has to consider the voltages on the power supply grid and consider what their worst case arrangements are and what the corresponding worst case delay is. In other words, it is not

enough to work with local worst case gate delays, one must look more globally at the whole path delay and consider how it depends on the voltages on the grid.

To address this problem, we are developing a framework for timing analysis that looks for worst case delay, taking into account supply voltage variations. A key part of the solution requires one to capture exactly how the “power tap” voltages (nodes where individual gates or cells draw their current from the power supply network) are related, if any. These node voltages are not independent of one another due to the fact that the power taps are all part of the on-chip power/ground network (simply, the power grid). Thus, the structure and the currents in the power grid become part of the overall problem of chip timing verification.

Our framework is in two phases. In a first phase, we apply an STA approach that assumes that all the power taps have voltages that are completely independent of one another. This technique will be described in Section III, a preliminary version of which has appeared in [1]. This technique allows for two successive gates on a path to have a big mismatch between their supplies, and is clearly not realistic for all gates (although it may be realistic for some cases). Nevertheless, as a result of this first-phase STA, we know the absolute worst case delay for the circuit and we have a list of the critical paths.

In the second phase of our approach, we take into account the presence of the power grid and operate on the list of critical paths resulting from the first-phase STA. For each path, we reduce its delay estimate, making it more realistic. This corrective action is applied to every critical path, starting with the one with the largest delay, until a path is reached whose corrected delay is larger than the uncorrected delay of the next path on the original list. When this happens, the path in hand has the worst case delay for this circuit and the analysis is complete.

The corrective action applied to each path must somehow take into account the currents and voltages on the power grid in order to discover the relationships among the power tap node voltages on that path. This is a very difficult problem because of the wide range of behaviors that the power grid can exhibit. The grid captures the exact relationship between the power tap nodes via the dynamical system equations that represent the grid. Most techniques for power grid analysis use some form of circuit simulation to compute the voltage fluctuations. However, given the very large number of possible circuit behaviors, one needs to simulate the circuit (for the currents) and the grid (for the voltage drops) for a large number of clock cycles or vector sequences, which is impractical. Add to this the fact that modern grids are huge, and it becomes clear that this straightforward simulation-based approach is prohibitively expensive. As an alternative, we will describe a “vectorless”

Manuscript received October 29, 2004; revised February 8, 2005 and June 30, 2005. This work was supported in part by Micronet, by ATI Technologies, by Altera Corporation, and by the Semiconductor Research Corporation (SRC) under Contract 2003-TJ-1070. This paper was recommended by Associate Editor S. Sapatnekar.

D. Kouroussis and F. N. Najm are with the Department of Electrical and Computer Engineering, University of Toronto, Toronto, ON M5S 2E4, Canada (e-mail: diony@eecg.utoronto.ca).

R. Ahmadi was with the Department of Electrical and Computer Engineering, University of Toronto, Toronto, ON M5S 2E4, Canada. He is now with ATI Technologies, Toronto, ON M5K 1B2, Canada.

Digital Object Identifier 10.1109/TCAD.2005.860953

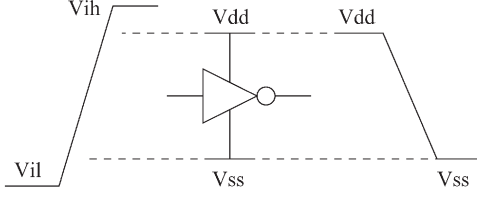


Fig. 1. Modeling parameters.

power grid analysis technique in which the worst case voltages on the grid are found without having complete information on the circuit currents and behaviors. This contribution is described in Section IV, a preliminary version of which has appeared in [2]. This technique is used to verify that the worst case voltage drop, on the power tap nodes for a given logic path, does not exceed some voltage threshold specification for that grid or path. Our technique requires only incomplete information about the circuit currents in the form of current constraints. These constraints take the form of local and global upper bounds on the circuit currents, but they can also be more general than that.

In Section V, we will then describe how the first-phase STA and the vectorless grid analysis are combined to apply corrective action to each critical path, in an iterative fashion, yielding an overall STA approach that does not require complete knowledge of the circuit currents. It will turn out that the problem can be formulated as a nonlinear programming problem (NLP), which we solve for the maximum delay subject to the current constraints. Lastly, we will present some results in Section VI that show the utility of this approach, and give some conclusions in Section VII.

II. OVERVIEW

Voltage fluctuations on the power grid are a result of many sources, such as IR drop, Ldi/dt drop, and resonance between the grid and the package. When inductive effects are negligible, which is the case for many technologies, simulation of the power grid is focused on only IR drop effects, given the RC structure of the grid. Thus, we will start with an RC model of the power and ground grids, and will then reduce the analysis further into a dc verification problem. Thus, in the present version of this work, we are able to include the effect of dc voltage drop on the grid on the circuit timing (loosely speaking; see Section IV for full details). The full (RC or RLC) transient version of the grid verification problem is more difficult and is part of our continuing work under this project.

Consider the diagram in Fig. 1, where an inverter is shown with its input and output waveforms. The power supply nodes of the inverter are considered, the reference V_{dd} and V_{ss} , and the input is assumed to rise from V_{il} to V_{ih} . The output of the inverter, as does the output of its fanout interconnect network, falls from V_{dd} to V_{ss} . It is instructive to consider what is a practical range of variations of the power supply values. In order for the circuit to function properly, the transistors must be able to turn OFF, which sets a limit on how large the supply variations may be. For one thing, we should have $|V_{ss} - V_{il}| < V_{tn}$ and $|V_{ih} - V_{dd}| < |V_{tp}|$. In the worst case, if we consider

opposite variations for (V_{ss}, V_{il}) and (V_{ih}, V_{dd}) , then

$$|\Delta V_{ss}| + |\Delta V_{il}| < V_{tn} \Rightarrow \text{roughly, } |\Delta V_{ss}| < \frac{V_{tn}}{2} \quad (1)$$

$$|\Delta V_{dd}| + |\Delta V_{ih}| < |V_{tp}| \Rightarrow \text{roughly, } |\Delta V_{dd}| < \left| \frac{V_{tp}}{2} \right|. \quad (2)$$

Throughout this paper, we have used 0.13- μm CMOS technology with a nominal supply voltage of 1.2 V, and assumed a 12.5% drop around V_{dd} and V_{ss} . This is equivalent to a 0.15-V drop around the nominal power supply and a 0.15-V ground bounce. Therefore, V_{ih} and V_{dd} can vary from 1.05 to 1.2 V, and V_{il} and V_{ss} can vary from 0 to +0.15 V. If there is a feasible arrangement of circuit currents that causes the voltage drop to exceed these bounds, we consider the grid to be unsafe and to require some improvement before one can proceed to study its impact on timing. Thus, in this paper, we are concerned with grids whose worst case voltages all fall within these bounds.

As an overview, our proposed timing verification flow is as follows.

- 1) Extract and enumerate the critical paths of the circuit under an assumption of independent power grid voltages.
- 2) Set up the current constraints for the power grid; the grid equations implicitly define the true relationships among node voltages on the grid.
- 3) Verify that the node voltages of each critical path do not exceed a 12.5% drop on the power grid.
- 4) For each critical path, solve for its worst case delay, taking the grid equations into account, leading to a new (reduced, corrected) delay value for that path.

The process does not have to exhaustively go through all critical paths identified in step 1. Instead, we start with a list of these paths that is sorted by decreasing delay and repeatedly apply step 4 until we encounter a path whose corrected delay is higher than the uncorrected delay of the next path on the list. When this happens, we have found the worst case circuit delay and we may stop.

III. FIRST-PHASE VOLTAGE-AWARE STA

In order to develop a timing analysis approach in the presence of power supply and ground voltage fluctuations, one needs to first develop a delay model for cells and interconnect that is dependent on these voltages. We will first define delay in a variable voltage environment, then introduce our delay models for gates and paths, and finally describe the STA.

A. Delay Definition

The notion of signal delay needs careful definition when the supplies are potentially different between the driver and the load. Consider the typical timing waveforms in Fig. 2. The series gate delay is defined as $t_{d1} = t_2 - t_1$, where t_1 is the time at which the input signal reaches $(V_{ih} + V_{il})/2$ and t_2 is the time at which the output reaches $(V_{dd} + V_{ss})/2$. The series interconnect delay is defined as $t_{d2} = t_3 - t_2$, where t_2 and t_3 are the times at which the input and output signals of the interconnect network reach $(V_{dd} + V_{ss})/2$.

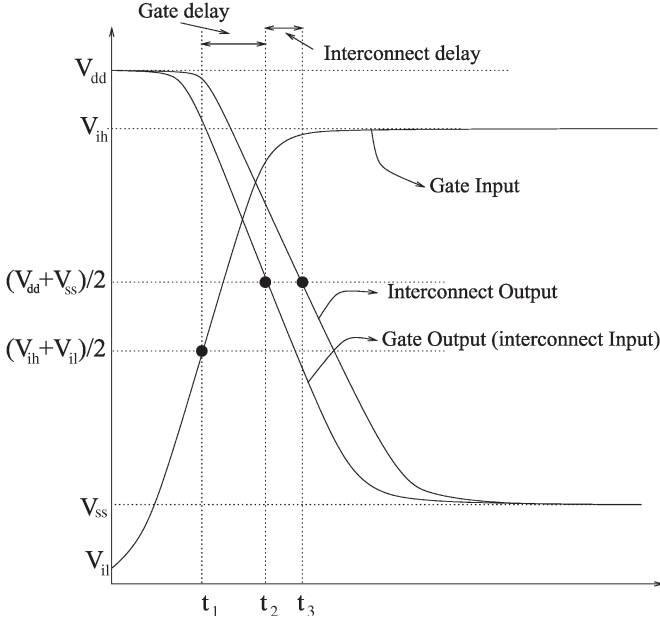


Fig. 2. Gate and interconnection delay.

B. Gate Delay Model

Gate delay depends on the traditional parameters of input signal slope and output load. In addition, in this paper, we model the dependence of gate delay on the four supply voltages defined in Fig. 1. Thus, six parameters are considered as part of the gate delay model: the input high signal level (V_{ih}), the input low signal level (V_{il}), the gate's power supply (V_{dd}), the gate's ground (V_{ss}), the input slope (S_{in}), and the gate's output load (C_l). The series input slope is defined as the slope (dV/dt) of the input waveform at the time when it crosses $(V_{il} + V_{ih})/2$.

It is instructive to consider how variable the cell delays are and how strong is their sensitivity to the supply voltages. To this end, we have built a library of cells containing two-input and three-input NAND, NOR, AND, OR, and NOT gates. In our experiments, the load, transistor widths, and four voltage levels of the gates were varied across their valid ranges (see Section II). Transistor width was allowed to vary from 160 (the minimum size for 0.13- μm technology) to 2400 nm, and the loads from 1 to 32.5 fF (as a comparison, the input capacitance of a minimum size gate for this technology is near 1 fF). Furthermore, different combinations of consecutive gates were tested. Fig. 3 shows all possible gate type combinations along with valid parameter ranges.

Modern cell libraries represent the delay of cells using four two-dimensional (2-D) tables for each timing arc (a timing arc is an input-output node pair). In case of a falling output, one table gives the propagation delay and another gives the output slope. Another two tables correspond to the rising output case. Each table covers the range of valid input slope and output load values. Simple extension of this model to our case would require six-dimensional (6-D) tables, which would be impractical in terms of model size and cost of building the model. In order to simplify the model, we found that the delay dependence on each voltage is near linear in the (narrow) range of valid voltages. However, to be more accurate, we have used

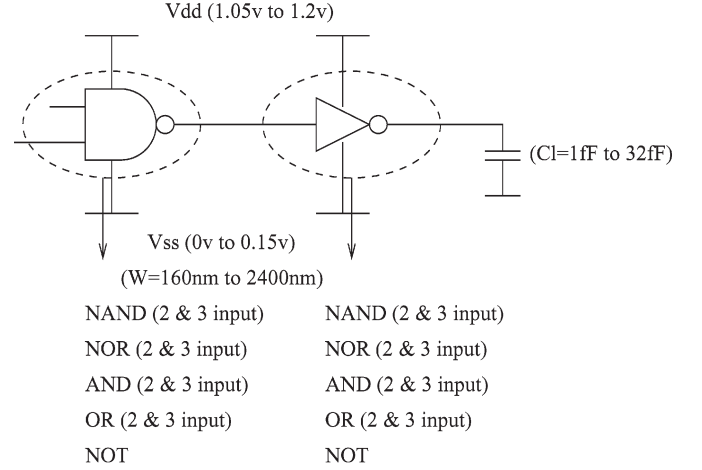


Fig. 3. Possible gates and parameters combination.

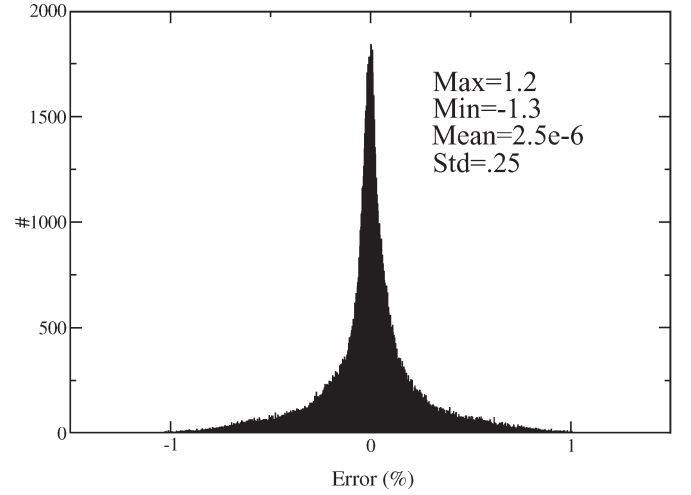


Fig. 4. Modeling error.

a quadratic polynomial to represent the dependence of delay on each voltage and made allowance for cross-product terms by using a template expression for delay as

$$t_d = \sum_k \alpha_k V_{ih}^{a_k} V_{il}^{b_k} V_{dd}^{c_k} V_{ss}^{d_k} \quad (3)$$

where $\alpha_k \in \mathcal{R}$ and $a_k, b_k, c_k, d_k \in \{0, 1, 2\}$. The regression coefficients α_k are found by using a standard least mean square (LMS) regression method [3]. Regression is performed for each grid point in the [slope, load] table so that each cell in the [slope, load] table contains the values for a number of coefficients $\alpha_1, \alpha_2, \dots, \alpha_m$. A similar model to this gives the output slope in terms of all four voltages and input slope and output load. We characterized (built the delay models for) all the cells in our library and then tested the error in delay between HSPICE and the library model. The results are shown in Fig. 4 for the propagation delay. It is seen that the model has very good accuracy. The output slope component of the model was also tested, and it shows an error of under 3%, which is also good.

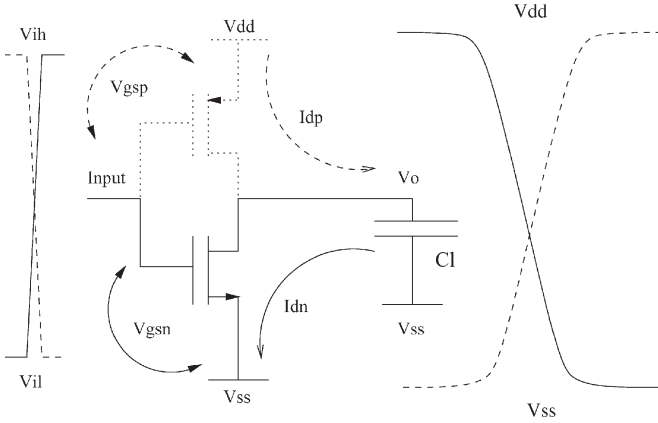


Fig. 5. Inverter with falling output.

C. Interconnect Delay

Interconnect delay can be modeled by any of the modern ways using either Elmore delay [4], moment matching [5], or other higher order modeling approaches. Interconnection delay is independent of both the driver and the load gate's voltages, and it just depends on interconnect model values and the transition time of the driver gate. Therefore, interconnect delay requires no special treatment.

D. Worst Case Gate Delay

Given a logic gate with variable supplies, it is important to look for the supply configuration that gives the worst case gate delay. The situation is complicated due to the number of variables involved, especially for complex CMOS gates. We will first consider this in the easy special case of an inverter, where analytical expressions are possible, and then generalize to the case of arbitrary CMOS gates.

1) *Special Case—Inverter:* In this section, we will consider inverters with rising and falling input signals. Simple quadratic equations are used for the NFET and PFET transistor currents and a delay expression is derived that shows, among other things, the dependence of delay on the supply and ground voltages. We then consider the sensitivity of delay to supply/ground variations and highlight the sign of the sensitivity terms, as this will turn out to be important in the rest of the paper. For more complex logic gates, for which analytical results are not possible, we will give empirical data to show the sign of the sensitivity terms.

a) *Step input:* Fig. 5 shows an inverter with an output load C_l . With V_{tp} and V_{tn} as the PFET and NFET threshold voltages, respectively, we let V_{gsp} and V_{gsn} be the gate-source voltage of the PFET and NFET transistors.

Falling delay: Consider a rising step signal as the input signal of the inverter, as shown in Fig. 5. Initially, the input of the inverter is low, NFET is in cutoff, and PFET is in saturation. When the input becomes high, the output load is discharged through the NFET and the output voltage may be found as the solution of the differential equation

$$C_l \frac{\partial V_o}{\partial t} = -I_{dn} \quad (4)$$

where $V_o(0) = V_{dd}$ and where

$$I_{dn} = \begin{cases} 0, & \text{for } V_{gsn} < V_{tn} \\ \beta_n \left((V_{gsn} - V_{tn})V_o - \frac{V_o^2}{2} \right), & \text{for } V_o < (V_{gsn} - V_{tn}) \\ \frac{\beta_n}{2} (V_{gsn} - V_{tn})^2, & \text{for } V_o > (V_{gsn} - V_{tn}) \end{cases} \quad (5)$$

where $V_{gsn} = V_{ih} - V_{ss}$. Solving for the falling delay [the time when V_o reaches $(V_{dd} + V_{ss})/2$] leads to

$$t_{df,step} = \ln \left(\frac{4V_{ih} - 5V_{ss} - 4V_{tn} - V_{dd}}{V_{dd} + V_{ss}} \right) + \frac{2(V_{ss} + V_{tn} + V_{dd} - V_{ih})}{(V_{ih} - V_{ss} - V_{tn})} \times \frac{C_l}{\beta_n(V_{ih} - V_{ss} - V_{tn})}. \quad (6)$$

We define the sensitivity of this delay to V_{dd} to be given by $\partial t_{df,step} / \partial V_{dd}$, and likewise for the other voltage variables. These sensitivities can be found analytically by differentiation; it is found that, for the whole range of allowable voltage variations, the sensitivity of this delay to V_{dd} and V_{ss} is positive and its sensitivity to V_{ih} is negative, so that

$$\frac{\partial t_{df,step}}{\partial V_{dd}} \geq 0, \quad \frac{\partial t_{df,step}}{\partial V_{ss}} \geq 0, \quad \frac{\partial t_{df,step}}{\partial V_{ih}} \leq 0, \quad \frac{\partial t_{df,step}}{\partial V_{il}} = 0. \quad (7)$$

Therefore, the worst case inverter falling delay may be found by setting $V_{dd} = H$, $V_{ss} = H$ and $V_{ih} = L$ (H stands for the highest allowable value and L stands for the lowest allowable value), which may be represented by the mnemonic

$$\begin{pmatrix} L \\ * \end{pmatrix} \begin{pmatrix} H \\ H \end{pmatrix}. \quad (8)$$

Rising delay: In the case when the input is a falling step, similar results can be found as follows. While the input signal is initially high, the PFET is in the cutoff mode and NFET is in saturation. When the input falls, the output load will be charged through PFET, as shown in Fig. 5. The output voltage may be found as the solution of the following differential equation:

$$C_l \frac{\partial V_o}{\partial t} = I_{dp} \quad (9)$$

where $V_o(0) = V_{ss}$ and where

$$I_{dp} = \begin{cases} 0, & \text{for } V_{gsp} > V_{tp} \\ \beta_p \left((V_{gsp} - V_{tp})V_{dsp} - \frac{V_{dsp}^2}{2} \right), & \text{for } V_{dsp} > (V_{gsp} - V_{tp}) \\ \frac{\beta_p}{2} (V_{gsp} - V_{tp})^2, & \text{for } V_{dsp} < (V_{gsp} - V_{tp}) \end{cases} \quad (10)$$

where $V_{gsp} = V_{il} - V_{dd}$ and $V_{dsp} = V_o - V_{dd}$. Solving for the rising delay (the time when V_o reaches $(V_{dd} + V_{ss})/2$) leads to

$$t_{dr,step} = \frac{C_l}{\beta_p(V_{il} - V_{dd} - V_{tp})} \left[\frac{2(V_{il} - V_{tp} - V_{ss})}{(V_{il} - V_{dd} - V_{tp})} + \ln \left(\frac{(V_{dd} - V_{ss})}{(-4V_{il} + 3V_{dd} + 4V_{tp} + V_{ss})} \right) \right]. \quad (11)$$

The sensitivities of this delay to various voltages can be found analytically. It is seen that $t_{dr,step}$ is independent of V_{ih} and that the sensitivities to V_{dd} and V_{ss} are both negative while the sensitivity to V_{il} is positive, i.e.,

$$\frac{\partial t_{dr,step}}{\partial V_{dd}} \leq 0, \quad \frac{\partial t_{dr,step}}{\partial V_{ss}} \leq 0, \quad \frac{\partial t_{dr,step}}{\partial V_{ih}} = 0, \quad \frac{\partial t_{dr,step}}{\partial V_{il}} \geq 0. \quad (12)$$

Therefore, the worst case inverter rising delay may be found by setting $V_{dd} = L$, $V_{ss} = L$ and $V_{il} = H$, which may be represented by the mnemonic

$$\begin{pmatrix} * \\ H \end{pmatrix} \begin{pmatrix} L \\ L \end{pmatrix}. \quad (13)$$

b) Ramp input: The previous section was based on an assumption of a step input. In order to obtain more realistic results, we consider a saturated ramp input. In this case, analytical results are possible, based on a case analysis [6] in which the input slope value is used to select one of two cases: 1) the input is fast, fast enough that it reaches its final value before the transistor (NFET for rising input, PFET for falling input) exits the saturation region and 2) the input is slow, slow enough that the transistor (NFET for rising input, PFET for falling input) exits the saturation region before the input reaches its final value.

Fast input case: For the fast input case, new differential equations can be formulated, and the falling and rising delays are given by

$$t_{df} = t_{df,step} + \frac{V_{ih} + 2V_{tn} + 2V_{ss} - 3V_{il}}{6S} \quad (14)$$

$$t_{dr} = t_{dr,step} + \frac{3V_{ih} - V_{il} - 2V_{dd} - 2V_{tp}}{6S} \quad (15)$$

where S is the slope of the input signal. It is helpful to rewrite these equations in the form

$$t_{df} = C_l g_f(\mathbf{V}) + \frac{h_f(\mathbf{V})}{S} \quad (16)$$

$$t_{dr} = C_l g_r(\mathbf{V}) + \frac{h_r(\mathbf{V})}{S} \quad (17)$$

where g_f , g_r , h_f , and h_r are functions of the four voltages (\mathbf{V} is a vector of the four voltages V_{dd} , V_{ss} , V_{ih} , and V_{il}) whose analytical expressions are clear from (6), (11), (14), and (15). Sensitivities can again be obtained by differentiation, leading to

$$\frac{\partial t_{df}}{\partial V_*} = C_l \frac{\partial g_f}{\partial V_*} + \frac{1}{S} \frac{\partial h_f}{\partial V_*} \quad (18)$$

$$\frac{\partial t_{dr}}{\partial V_*} = C_l \frac{\partial g_r}{\partial V_*} + \frac{1}{S} \frac{\partial h_r}{\partial V_*} \quad (19)$$

where V_* is any of the four voltages V_{dd} , V_{ss} , V_{ih} , or V_{il} .

Notice that $\partial g_f / \partial V_*$ has the same sign as $\partial t_{df,step} / \partial V_*$ and $\partial g_r / \partial V_*$ has the same sign as $\partial t_{dr,step} / \partial V_*$. Therefore, for a falling output, notice that whenever $\partial g_f / \partial V_*$ has the same sign as $\partial h_f / \partial V_*$, then $\partial t_{df} / \partial V_*$ has the same sign as $\partial t_{df,step} / \partial V_*$. Thus, the only case when $\partial t_{df} / \partial V_*$ and $\partial t_{df,step} / \partial V_*$ may have different signs is when $\partial g_f / \partial V_*$ has a different sign from

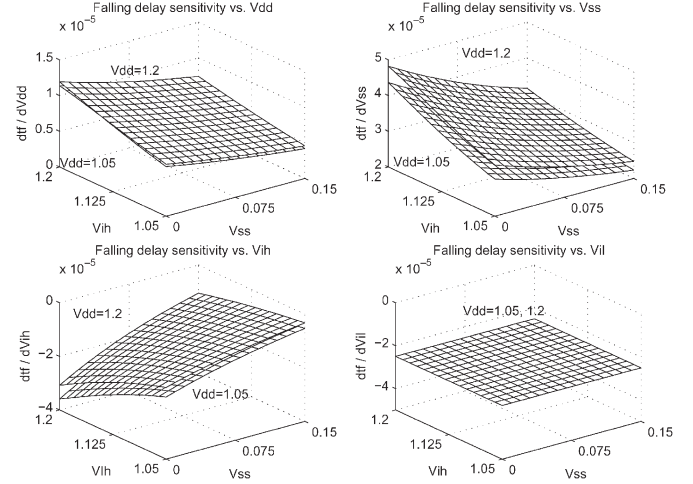


Fig. 6. Falling delay sensitivities for ramp input with V_{il} set at nominal value.

$\partial h_f / \partial V_*$, which one can easily show occurs only when V_* corresponds to V_{ih} (in which case $\partial g_f / \partial V_{ih}$ is negative and $\partial h_f / \partial V_{ih}$ is positive) and for small values of S and C_l . Since both S and C_l are bounded, we have set them at their minimum values and computed sensitivities for different voltage combinations. Across the whole range of allowed voltages, it was found that $\partial t_{df} / \partial V_*$ has the same sign as $\partial t_{df,step} / \partial V_*$, as can be seen in Fig. 6, which was generated for the case when S and C_l are at their respective minimum values. Therefore, an important conclusion is that, in the fast input case, with the output falling, the sensitivities have the same signs as was found in the step input case for all possible values of input slope and output load, i.e.,

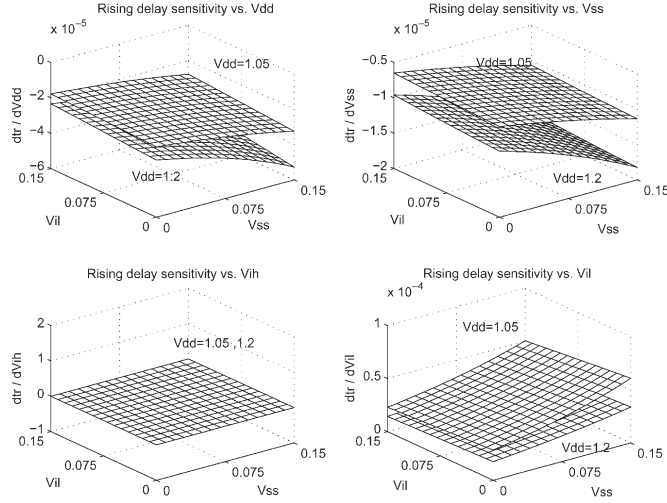
$$\frac{\partial t_{df}}{\partial V_{dd}} \geq 0, \quad \frac{\partial t_{df}}{\partial V_{ss}} \geq 0, \quad \frac{\partial t_{df}}{\partial V_{ih}} \leq 0, \quad \frac{\partial t_{df}}{\partial V_{il}} \leq 0. \quad (20)$$

A similar analysis applies to t_{dr} . The only case where $\partial g_r / \partial V_*$ has a different sign from $\partial h_r / \partial V_*$ is when V_* corresponds to V_{il} , in which case $\partial g_r / \partial V_{il}$ is positive and $\partial h_r / \partial V_{il}$ is negative. Again, setting both S and C_l to their minima, it was found that, for all voltages in the allowed range, $\partial t_{dr} / \partial V_*$ has the same sign as $\partial t_{dr,step} / \partial V_*$, as can be seen in Fig. 7, which was generated for the case when S and C_l are at their respective minimum values. Therefore, an important conclusion is that, in the fast input case, with the output rising, the sensitivities have the same signs as was found in the step input case for all possible values of input slope and output load

$$\frac{\partial t_{dr}}{\partial V_{dd}} \leq 0, \quad \frac{\partial t_{dr}}{\partial V_{ss}} \leq 0, \quad \frac{\partial t_{dr}}{\partial V_{ih}} \geq 0, \quad \frac{\partial t_{dr}}{\partial V_{il}} \geq 0. \quad (21)$$

Slow input case: For the slow input case, the analysis becomes much more complicated. It is possible to obtain expressions for the rising and falling delays, but the sensitivities were then obtained by numerical differentiation (finite-difference approximation). The same results are found, as (20) and (21), for the signs of the sensitivities.

c) Summary—inverter case: Sensitivities of inverter delay to supply voltage variations have the signs given in (20) and (21) for all possible voltages, slopes, and loads in the allowed

Fig. 7. Rising delay sensitivities for ramp input with V_{ih} set at nominal value.TABLE I
INVERTING GATE DELAY SENSITIVITY

Input Signal	$\frac{\partial t_d}{\partial V_{dd}}$	$\frac{\partial t_d}{\partial V_{ss}}$	$\frac{\partial t_d}{\partial V_{ih}}$	$\frac{\partial t_d}{\partial V_{il}}$
Rising	+	+	-	-
Falling	-	-	+	+

ranges. Correspondingly, the worst case voltage configuration is given by

$$\text{for falling output: } \begin{pmatrix} L \\ L \end{pmatrix} \begin{pmatrix} H \\ H \end{pmatrix} \quad (22)$$

$$\text{for rising output: } \begin{pmatrix} H \\ H \end{pmatrix} \begin{pmatrix} L \\ L \end{pmatrix}. \quad (23)$$

2) *General Case—Arbitrary Gates*: In order to generalize the analysis, we consider a cascade of two gates, as in Fig. 3, where the supplies of the “driver gate” are V_{ih} and V_{il} and the supplies of the “load gate” are V_{dd} and V_{ss} for the following reason. In practice, every CMOS gate is driven by another CMOS gate so that a variation of the supply and ground of the driver gate would affect its output slope, and hence the input slope of the load gate. Thus, it is important that the sensitivities and the worst case settings of V_{ih} , V_{il} , V_{dd} , and V_{ss} be made in a realistic situation where changes of V_{ih} and V_{il} have an effect on the input slope of the load gate. An analytical study of this situation is not possible. Instead, all combinations of gates, of varying sizes, were simulated in the configuration in Fig. 3. All inverting gates in our library show the same sign pattern that was found analytically for the inverter, summarized in Table I, and which leads to the worst case voltage settings in (22) and (23).

3) *Gates With Connected Supplies (Blocks)*: We now extend the analysis to handle combinations of gates whose supplies are not independent. Especially interesting is the special case when several consecutive inverting gates on a path share a common power supply and ground; we call this structure a block. Thus, a block may be a simple AND cell from a cell library or a general path of consecutive inverting gates with connected supplies. The case of a block consisting of a single gate will be considered a degenerate or trivial case and will be referred to as a triv-

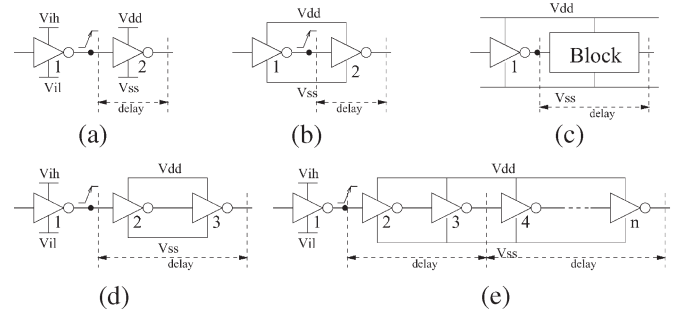


Fig. 8. Various test cases.

ial block. In general, the term “block” will refer to a nontrivial block. For block analysis, analytical methods are not available, and we will use empirical data to study delay sensitivities.

Recall that, for a case such as in Fig. 8(a), where the output of gate 1 is rising, the worst case delay of gate 2 corresponds to $\begin{pmatrix} L \\ L \end{pmatrix} \begin{pmatrix} H \\ H \end{pmatrix}$. If the two supplies of the driver and load gates are connected, such as in Fig. 8(b), then the worst case setting for the delay of gate 2 is simply $\begin{pmatrix} L \\ H \end{pmatrix}$, irrespective of signal polarity in fact. This is a commonly known fact and can easily be shown analytically by replacing V_{ih} by V_{dd} and V_{il} by V_{ss} in (14) and (15) and differentiating both equations. Indeed, it is not hard to see that irrespective of the type of gates and the length of the path, for an arrangement such as in Fig. 8(c), the worst case delay of the block identified in the figure corresponds to $\begin{pmatrix} L \\ H \end{pmatrix}$.

Consider now the case in Fig. 8(d), where, for a rising input to gate 2, we are interested in the delay of the block composed of gates 2 and 3. In this case and according to the preceding discussion, the worst case delay of gate 2 is achieved for $V_{dd} = H$ while the worst case delay of gate 3 requires $V_{dd} = L$. How is this conflict to be resolved? We have found, empirically, that under all conditions of slopes and loads, the sensitivity of gate 3 is dominant, so that the worst case combination turns out to be $\begin{pmatrix} L \\ L \end{pmatrix} \begin{pmatrix} L \\ H \end{pmatrix}$. This happens because the delay of a logic gate whose output is being pulled high (such as gate 3) is more dependent on the value of V_{dd} than the delay of a gate whose output is being pulled low (such as gate 2). This conclusion was also found to apply for all cases where gates 1, 2, and 3 are any other inverting gate from our library.

Finally, consider Fig. 8(e). Since it has the same supplies as its driver, the worst case delay for the block composed of gates 4, 5, \dots , n corresponds to $V_{dd} = L$, $V_{ss} = H$. Since the worst case delay for the block composed of gates 2 and 3 is also $\begin{pmatrix} L \\ H \end{pmatrix}$, then the general conclusion (we have similarly analyzed the falling input case) is that for any (nontrivial) block the worst case block delay corresponds to

$$\text{for a rising input: } \begin{pmatrix} L \\ L \end{pmatrix} \begin{pmatrix} L \\ H \end{pmatrix} \quad (24)$$

$$\text{for a falling input: } \begin{pmatrix} H \\ H \end{pmatrix} \begin{pmatrix} L \\ H \end{pmatrix}. \quad (25)$$

A summary for the worst case block delay configurations, for both inverting gates (trivial blocks) and for general (nontrivial) blocks, which includes noninverting cells, is given in Table II.

TABLE II
VOLTAGE CONFIGURATIONS FOR WORST CASE DELAY

Gate	Input Falling				Input Rising			
	V_{dd}	V_{ss}	V_{ih}	V_{il}	V_{dd}	V_{ss}	V_{ih}	V_{il}
Inverting gate	L	L	H	H	H	H	L	L
Block	L	H	H	H	L	H	L	L

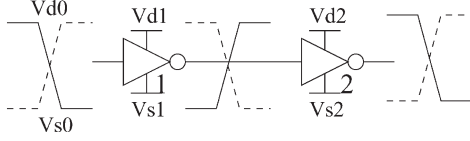


Fig. 9. Connected inverting gates with independent supplies and grounds.

E. Worst Case Path Delay

The total delay of a signal along a path of gates (specifically, along a path of timing arcs) is the sum of the individual delays of all the timing arcs on the path. Consider all the supply voltages of the gates on a path. If these voltages are viewed as independent variables, then what is the combination of supply values that gives the worst case path delay? The path delay corresponding to this setting is the absolute worst case in practice and is therefore worth studying. We first consider this question, and then consider the case when the path includes both gates with independent supplies and blocks.

1) *Gates With Independent Supplies*: Consider the simple two-gate path shown in Fig. 9 with a falling input. In the following, we will use the following simplified notation so as to simplify the presentation. For a gate “ i ,” we will denote its delay sensitivity to “its” supply voltage as $\partial t_{dri}/\partial V_{dd}$ (for the rising output case), even though that supply node may be labeled differently on the diagram. For instance, in Fig. 9, the sensitivity of gate 1 to its supply voltage will be denoted $\partial t_{dr1}/\partial V_{dd}$ and the sensitivity of the delay of gate 2 to its supply voltage will be denoted $\partial t_{df2}/\partial V_{dd}$, even though the two supply nodes are labeled V_{d1} and V_{d2} in the figure, likewise for the other voltages.

If $t_p = t_{dr1} + t_{df2}$ is the total path delay, then $\Delta t_p = \Delta t_{dr1} + \Delta t_{df2}$ leads to

$$\begin{aligned} \Delta t_p = & \frac{\partial t_{dr1}}{\partial V_{ih}} \Delta V_{d0} + \frac{\partial t_{dr1}}{\partial V_{il}} \Delta V_{s0} + \frac{\partial t_{dr1}}{\partial V_{dd}} \Delta V_{d1} \\ & + \frac{\partial t_{dr1}}{\partial V_{ss}} \Delta V_{s1} + \frac{\partial t_{df2}}{\partial V_{ih}} \Delta V_{d1} + \frac{\partial t_{df2}}{\partial V_{il}} \Delta V_{s1} \\ & + \frac{\partial t_{df2}}{\partial V_{dd}} \Delta V_{d2} + \frac{\partial t_{df2}}{\partial V_{ss}} \Delta V_{s2} \end{aligned} \quad (26)$$

and, collecting terms, this leads to

$$\begin{aligned} \Delta t_p = & \left(\frac{\partial t_{dr1}}{\partial V_{dd}} + \frac{\partial t_{df2}}{\partial V_{ih}} \right) \Delta V_{d1} + \left(\frac{\partial t_{dr1}}{\partial V_{ss}} + \frac{\partial t_{df2}}{\partial V_{il}} \right) \Delta V_{s1} \\ & + \left(\frac{\partial t_{df2}}{\partial V_{dd}} \right) \Delta V_{d2} + \left(\frac{\partial t_{df2}}{\partial V_{ss}} \right) \Delta V_{s2} \\ & + \left(\frac{\partial t_{dr1}}{\partial V_{ih}} \right) \Delta V_{d0} + \left(\frac{\partial t_{dr1}}{\partial V_{il}} \right) \Delta V_{s0}. \end{aligned} \quad (27)$$

Considering Table I, it is clear that the coefficient of ΔV_{d1} in (27) is negative. Therefore, in order to have the maximum delay, one should set $V_{d1} = L$. Likewise, for the other voltages, $V_{d0} = H$, $V_{d2} = H$, $V_{s0} = H$, $V_{s1} = L$, and $V_{s2} = H$. For a rising input signal, we have the same expression with different signs, leading to the following worst case voltage setting: $V_{d0} = L$, $V_{d1} = H$, $V_{d2} = L$, $V_{s0} = L$, $V_{s1} = H$, and $V_{s2} = L$. Since Table I is valid for all inverting gates, not only inverters, then this result is general and applies to arbitrary inverting gates.

It is interesting that the worst case delay is so easily identifiable and corresponds to a setting of

$$\begin{pmatrix} H \\ H \end{pmatrix} \begin{pmatrix} L \\ L \end{pmatrix} \begin{pmatrix} H \\ H \end{pmatrix} \quad (28)$$

for the falling input case, and the opposite setting for the rising input case. The reason this works so well is that the individual worst case assignments of the gates match exactly due to the reversed polarity of the transitions at the outputs of consecutive gates.

Indeed, it is clear that this result extends naturally to paths of arbitrary length by induction. Therefore, for a multigate path composed of all inverting gates with independent supplies, the worst case voltage setting for a falling input is given the staggered arrangement

$$\begin{pmatrix} H \\ H \end{pmatrix} \begin{pmatrix} L \\ L \end{pmatrix} \begin{pmatrix} H \\ H \end{pmatrix} \begin{pmatrix} L \\ L \end{pmatrix} \begin{pmatrix} H \\ H \end{pmatrix} \cdots \quad (29)$$

and for a rising input it is given by the alternate staggered arrangement

$$\begin{pmatrix} L \\ L \end{pmatrix} \begin{pmatrix} H \\ H \end{pmatrix} \begin{pmatrix} L \\ L \end{pmatrix} \begin{pmatrix} H \\ H \end{pmatrix} \begin{pmatrix} L \\ L \end{pmatrix} \cdots \quad (30)$$

2) *Mix of Independent Gates and Blocks*: When considering a path that mixes noninverting gates and general blocks, then it is possible to observe a conflict between the sensitivities to the supplies so that the solution is not necessarily the nice staggered arrangements seen above. In theory, a conflict in the supply voltage assignment can always be resolved during timing analysis (as will be described below) by generating and following various alternatives. The mechanism for doing this will be seen to require the generation of additional “signals” to be propagated during the timing analysis. However, in order to reduce the overhead due to these signals, we will describe ways in which certain conflicts can be resolved easily without the need for additional signals during timing analysis.

Conflicts can be resolved easily in case of a series connection of two blocks. If the first block is a trivial block (a single inverting gate), then there is actually no conflict. To see this, consider the case when the signal at the intermediate node (input of the second block) is rising. Then, based on the preceding analysis, the worst case delays of both the gate and the block are achieved when the gate’s supplies are set to $\begin{pmatrix} L \\ L \end{pmatrix}$. If that signal is falling, then the gate’s supplies should be $\begin{pmatrix} H \\ H \end{pmatrix}$ in order to maximize both the gate delay and block delay, and there is no conflict. Conflict arises when the first block is nontrivial, as follows.

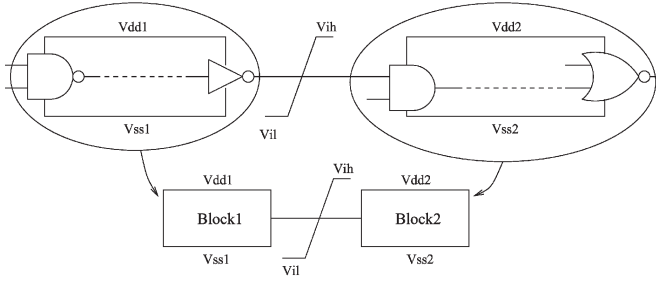


Fig. 10. Consecutive blocks with independent supplies.

Consider two consecutive blocks with independent supplies, as shown in Fig. 10, and consider the case when the output of the first block is rising. The first (nontrivial) block requires a setting of $\binom{L}{H}$ for its supplies. The second block requires a setting of $\binom{L}{L}$. Thus, there is a conflict in the setting of the ground value of the first block. We have found, empirically, that the sensitivity of t_{d2} (delay of block 2) to V_{ss1} is always smaller (in magnitude; recall, this sensitivity is negative) than the (positive) sensitivity of t_{d1} (delay of block 1) to V_{ss1} , leading to the conclusion that V_{ss1} must be set to H in order to maximize the path delay. Basically, the sensitivity of the delay of a logic gate to its supply voltage turns out to be larger than its sensitivity to the input signal level. When the intermediate signal is falling, the conflict has to do with the value of V_{dd1} , and we have found that the worst case corresponds to setting V_{dd1} to L . We now show some empirical justification for these conclusions.

Fig. 11(a) shows $\partial t_{df2}/\partial V_{il}$ and $\partial t_{dr1}/\partial V_{ss}$ in the same histogram when the first block (Fig. 10) is a cascade of two inverting gates and the second block is a single inverting gate. It is seen that the former sensitivity is negative and the latter is positive, but the minimum value of the latter is greater than the absolute value of the minimum value of the former. Therefore, the sensitivity of the path delay to V_{ss1} is positive and V_{ss1} must be set to H . Fig. 11(b) shows $\partial t_{dr2}/\partial V_{ih}$ and $\partial t_{df1}/\partial V_{dd}$ for the same circuit when the intermediate node is falling. Here, the former sensitivity is positive and the latter is negative and the maximum value of the former is less than the absolute value of the maximum value of the latter, therefore the overall delay sensitivity of the path to V_{dd1} is negative and V_{dd1} must be set to L . The above data were obtained for all combinations of gates in our library. If the first block is longer than simply two gates, its sensitivity to its supply or ground will only increase (in magnitude) so that the same conclusions hold. If the second block is nontrivial, then it has more delay and its sensitivity to V_{ss1} or V_{dd1} increases in a way which could, in theory, negate our conclusion. However, since the input signal mainly affects the delay of the first one or two gates in the path (again, we have established this empirically but it is not hard to see why it is true), this does not happen, and the conclusion is intact.

F. STA

STA gives the maximum delay of a combinational circuit. The available techniques range from the early work of Kirkpatrick and Clark [7] and Hitchcock *et al.* [8] to recent work by Blaauw *et al.* [9], which is significant in that it carefully takes into account the effect of the input slope on path delay

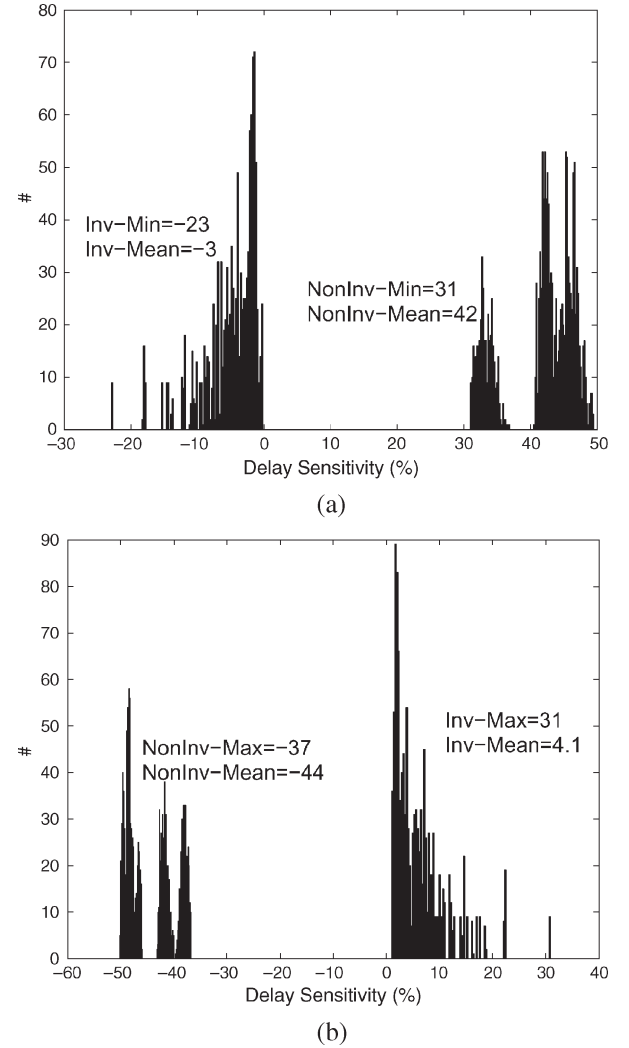


Fig. 11. (a) V_{ss} and V_{il} sensitivity for falling output. (b) V_{dd} and V_{ih} sensitivity for rising output.

during signal propagation. Our implementation of STA is based on [9]. We consider that supply nodes of the logic gates in a circuit are either tied together, in arbitrary combinations, or are independent. By “tied together,” we mean connected by a short circuit, so that they are the same electrical node on the grid. We also assume that if the power supply nodes of two gates are tied in this way, then their grounds are tied as well, and vice versa.

For each primary input, two signals are created, one rising and one falling, each with an arrival time of 0. For each logic gate, we propagate the signals at its inputs to its output, and then we prune the signal set at that output node. If the supply nodes of that gate are tied nowhere else, then each signal at a gate’s input node yields one signal at the gate’s output node, which has arrival time and slope as determined by our timing model, using the worst case supply settings for that gate and for that polarity of transition. This supply setting becomes part of the signal description and is carried along. Once all the signals at the gate’s inputs have been propagated thus to its output, the signal set at the output is pruned as in [9]. At the circuit primary outputs, the signal with the latest arrival time determines the circuit delay and the voltage assignment tagged to that signal is the worst case voltage assignment for that circuit.

If, however, the supply nodes of the logic gate are tied elsewhere, meaning that this gate is either part of a block or that this gate's supplies are tied to some other gate's supply elsewhere in the circuit, then a conflict may arise in that the voltage assignment that one would like to make for this gate's supplies may conflict with other assignments that are already part of this signal's description, or may conflict with future assignments that one may want to make for these supplies in connection with another gate downstream. Conflict resolution is done by generating extra signals. Each signal at an input of this gate is propagated as two new signals at the gate output, each with a different setting of the supplies. Since there are two supplies (V_{dd} and V_{ss}), one would think that four signals would be required. However, we actually use only two signals, which are chosen in a conservative way, meaning that we may err slightly but pessimistically on the delay. At the gate output, signals whose voltage assignments do not conflict are pruned separately, as a subgroup. The latest arrival time at the primary outputs is identified as the worst case circuit delay. In order to find the path in which the latest arriving signal has passed through, a backtracking process must be applied, from the circuit primary output that has the latest arrival time toward the primary input. Starting from that output pin, maximum gate delay is subtracted from the signal arrival time at the gate output and the previous gate arrival time in the critical path is computed. STA looks through all the input pins of the current gate and finds the gate with the same signal arrival time as the computed arrival time. It then flags the gate in the previous stage as the gate with the latest signal arrival time and continues the backtracking process up to the primary input.

IV. VECTORLESS GRID ANALYSIS

We need to verify that the circuit delay never exceeds certain bounds for all possible voltages on the grid. Thus, we are in essence checking for the largest value of a function of node voltages. In this section, we will focus on the node voltages themselves and consider how we can verify what the largest voltage drop is on the grid at a given node. In the following section, we will then extend this to the function of node voltages, i.e., circuit delay.

Focusing then on checking the voltage at a given node on the grid, we reiterate the point made earlier, that grid analysis by simulation is prohibitively expensive and does not offer a complete guarantee. Instead, we will develop an approach that does not rely on knowing the circuit currents. We will require only incomplete information on the circuit currents via current constraints. The resulting approach does not depend on knowing the circuit activity patterns, hence it may be referred to as vectorless. Finally, since the analysis of the power supply network is similar to that of the ground network, we will focus on one of them only. In the following, we will only focus on the power supply network, which will be referred to as the power grid, or simply the grid.

A. Preliminaries

We consider an RC model of the power grid, where each branch of the grid is represented by a resistor and where

there exists a capacitor from every grid node to ground. In addition, some grid nodes have ideal current sources (to ground) representing the current drawn by the circuit tied to the grid at that point and some grid nodes have ideal voltage sources (to ground) representing the connections to the external voltage supply.

Let the power grid consist of $n + p$ nodes, where nodes $1, 2, \dots, n$ have no voltage sources attached and nodes $(n + 1), (n + 2), \dots, (n + p)$ are the nodes where the p voltage sources are connected. Let c_k be the capacitance from every node k to ground. Let $i_k(t)$ be the current source connected to node k , where the direction of positive current is from node to ground. We assume that $i_k(t) \geq 0$ and that $i_k(t)$ is defined for every node $k = 1, \dots, n$ so that nodes with no current source attached have $i_k(t) = 0 \forall t$. Let $\mathbf{i}(t)$ be the vector of all $i_k(t)$ sources, $k = 1, \dots, n$. Let $u_k(t)$ be the voltage at every node k , $k = 1, \dots, n$, and let $\mathbf{u}(t)$ be the vector of all $u_k(t)$ signals, $k = 1, \dots, n$. Applying Kirchhoff's current law (KCL) at every node, $k = 1, \dots, n$, leads to

$$\mathbf{G}\mathbf{u}(t) + \mathbf{C}\dot{\mathbf{u}}(t) = -\mathbf{i}(t) + \mathbf{G}\mathbf{V}_{DD} \quad (31)$$

where \mathbf{G} is an $n \times n$ conductance matrix resulting from the application of the traditional modified nodal analysis formulation [10] (simplified by the fact that all the voltage sources in this case are from node to ground), \mathbf{C} is an $n \times n$ diagonal matrix of node capacitances, and \mathbf{V}_{DD} is a constant vector, each entry of which is equal to V_{DD} . The matrix \mathbf{G} has several useful properties. It is symmetric positive definite [11] and can be shown to be an M-matrix [12], which means, among other things, that its inverse consists of only nonnegative values. We may rewrite (31) as $\mathbf{G}[\mathbf{V}_{DD} - \mathbf{u}(t)] - \mathbf{C}\dot{\mathbf{u}}(t) = \mathbf{i}(t)$ and if we now define $v_k(t) = V_{DD} - u_k(t)$ to be the voltage drop at node k and let $\mathbf{v}(t)$ be the vector of voltage drops, then the system equation can be written as

$$\mathbf{G}\mathbf{v}(t) + \dot{\mathbf{v}}(t) = \mathbf{i}(t). \quad (32)$$

This is a revised system equation that one can solve directly for the voltage drop values. Notice that the circuit described by this equation consists of the original power grid, but with all the voltage sources set to zero (short circuit) and all the current source directions reversed. In the following, we will mainly be concerned with this modified power grid and the revised system of (32).

We now point out a key monotonicity property of the power grid, which will be useful. If we increase any of the currents driving the grid, at any point in time, then the overall voltage waveform at any point on the grid will either decrease or stay the same, but will not rise. We can formally express this (see [13] for a proof) as follows.

Proposition 1 (Monotonicity): If $\mathbf{v}(t)$ is the voltage drop due to $\mathbf{i}(t)$ and $\mathbf{v}^*(t)$ is the voltage drop due to $\mathbf{i}^*(t)$, then the power grid has the property

$$\text{if } \mathbf{i}^*(t) \geq \mathbf{i}(t) \quad \forall t \geq 0, \text{ then } \mathbf{v}^*(t) \geq \mathbf{v}(t) \quad \forall t \geq 0 \quad (33)$$

which we will express by saying that the grid is monotone.

A similar result was earlier proven [14] for the special case of an RC tree driven by a single voltage source. Based on the monotonicity property, we can now make a couple of statements that will be useful below. Let I_k be an upper bound on $i_k(t)$ over the time period of interest, say $0 \leq t \leq \infty$. Let I_1, I_2, \dots, I_n form a $n \times 1$ vector \mathbf{I} and let \mathbf{V} be the solution of the system when the dc currents \mathbf{I} are applied as inputs, which may be found by solving the dc system

$$\mathbf{G}\mathbf{V} = \mathbf{I}. \quad (34)$$

Then, from the monotonicity property, it is clear that $\mathbf{i}(t) \leq \mathbf{I} \forall t \geq 0$ leads to $\mathbf{v}(t) \leq \mathbf{V} \forall t \geq 0$. Finally, another related result is that, considering the dc system (34), if $\mathbf{I}^* \geq \mathbf{I}$, then $\mathbf{V}^* \geq \mathbf{V}$.

B. Current Constraints

In order to achieve a vectorless approach, we will use two types of incomplete current specification, referred to as current constraints: local constraints and global constraints.

1) *Local Constraints*: A local constraint relates to a single current source. For instance, one may specify that current $i_k(t)$ never exceeds a certain fixed level $I_{L,k}$, i.e., $i_k(t) \leq I_{L,k} \forall t \geq 0$. This upper bound may be simply known from prior simulation if the cell or block is already available, or it may be a best-guess based on the area of the cell or block and on perhaps the power density of the design (total power divided by total area). If further information is available on the circuit behavior over time, then the user may be able to specify an upper bound waveform, as a time function, so that $i_k(t) \leq i_{L,k}(t) \forall t \geq 0$. We will assume that every current source tied to the power grid has an upper bound associated with it, be it a fixed bound or a waveform bound. If a grid node does not have a current source attached to it, i.e., $i_k(t) = 0 \forall t \geq 0$, then we specify a fixed zero-current upper bound for that node, $I_{L,k} = 0$. This convention will be useful later on. In this way, we have a local constraint associated with every node of the power grid. We express these constraints in vector form as

$$\mathbf{0} \leq \mathbf{i}(t) \leq \mathbf{I}_L \quad \forall t \geq 0. \quad (35)$$

Notice that, if only local constraints are provided, then checking for a worst case voltage drop on a node is trivial due to the monotonicity of the power grid: Simply set each current source to its maximum allowable value and simulate the grid. The resulting voltage drops are the maximum that can exist under these constraints. Of course, with only local constraints, the results can be very pessimistic because it is never the case that all chip components simultaneously draw their maximum current, hence the need for global constraints. Handling global constraints, however, is not as straightforward.

2) *Global Constraints*: A global constraint relates to all current sources or to subgroups of current sources. For instance, if the total power dissipation of the chip is known, even approximately, then one may say that the sum of all the current sources is no more than a certain upper bound. In general, a global constraint corresponds to the case when the sum of the currents for a group of current sources is specified to have an

upper bound. These constraints are useful to express the fact that certain groups of current sources (corresponding to certain functional blocks, or perhaps to the whole chip) draw no more than a certain total level of current, corresponding perhaps to the known total power dissipation for that block. The upper bound, corresponding to the j th global constraint, may be a fixed bound $I_{G,j}$ or a waveform bound $i_{G,j}(t)$. If m is the number of available global constraints, then we express all the global constraints in matrix form as

$$\mathbf{U}\mathbf{i}(t) \leq \mathbf{I}_G \quad (36)$$

where \mathbf{U} is a $m \times n$ matrix that contains only 0s and 1s.

3) *Combining Constraints*: The local and global constraints can be combined into a single matrix inequality as

$$\mathbf{L}\mathbf{i}(t) \leq \mathbf{I}_m \quad \text{with} \quad \mathbf{i}(t) \geq \mathbf{0} \quad \forall t \geq 0 \quad (37)$$

where \mathbf{L} is an $(n+m) \times n$ matrix of 0s and 1s, whose first n rows form an identity matrix (1s on the diagonal and 0s everywhere else) and whose remaining m rows correspond to the matrix \mathbf{U} , and where \mathbf{I}_m is a $(n+m) \times 1$ vector.

C. Node Robustness

Consider the case where one is verifying node voltage under fixed (dc) currents: One is dealing with dc inputs \mathbf{I} , dc voltages \mathbf{V} , and the dc system $\mathbf{G}\mathbf{V} = \mathbf{I}$. The local constraints become $\mathbf{0} \leq \mathbf{I} \leq \mathbf{I}_L$, the global constraints are $\mathbf{U}\mathbf{I} \leq \mathbf{I}_G$ and their combination is

$$\mathbf{L}\mathbf{I} \leq \mathbf{I}_m, \quad \mathbf{I} \geq \mathbf{0}. \quad (38)$$

Fixed current upper bounds will be referred to as dc constraints, otherwise one is working with transient constraints.

Suppose that we are given, for each node k , the maximum allowable voltage drop at that node $V_{m,k}$ (a voltage threshold). We define a node to be robust for a given set of constraints if and only if $V \leq V_{m,k}$ for any current vector $\mathbf{i}(t)$ that satisfies these constraints. In general, the voltage thresholds and the voltages themselves may be functions of time. If one is given a set of dc constraints \mathbf{I}_m , then the following result is useful.

Proposition 2: A node is robust for all transient currents whose peak values satisfy a given set of dc constraints if and only if it is robust for all dc currents that satisfy these constraints.

Proof: The forward direction is trivial. Since a dc current is a special limiting case of a transient current, then robustness under a class of transient currents implies robustness under any dc current that also belong to that class. The reverse direction is true because of the monotonicity property: Assuming that a node is robust under dc currents, then given a transient current assignment whose peaks satisfy the constraints, we can construct a dc current assignment by setting a dc current value equal to the peak value of each current source. This dc current assignment satisfies the constraints, therefore the node must be robust under this assignment. Now, since for each current source the transient current is always below the corresponding dc current, then by (33), the node is also robust for that transient current assignment.

This result is useful because it provides that, when the constraints are given as dc constraints, checking node robustness under all dc currents that satisfy these constraints provides more than just robustness under dc currents—it provides that under a large class of transient currents (as given in Proposition 2), the node is also robust. Fixed upper-bound dc constraints are much easier to specify than transient upper-bound constraints. This is because in order to provide transient constraints one requires much more design knowledge, including some notion of system or circuit timing. ■

D. DC Robustness

Obviously, solving the node robustness problem under dc conditions, what we will call the dc robustness problem, is much easier than solving under transient current conditions. The rest of this paper is focused on this dc robustness approach. It is understood that this is not a complete solution to the problem, but three comments may be made in this regard: 1) based on Proposition 2, dc robustness implies robustness under transient currents for certain classes of current; 2) the dc approach can be used to identify gross or major problems with the grid and can then be followed up with more detailed analysis of the problematic portion of the grid; and 3) we continue to work on the transient verification problem and we are finding that it is based in a large part on this dc approach as an underlying technique.

E. Voltage Formulation

By making use of the relationship $\mathbf{I} = \mathbf{G}\mathbf{V}$, we can express the dc constraints in terms of voltages

$$\mathbf{LGV} \leq \mathbf{I}_m, \quad \mathbf{V} \geq 0. \quad (39)$$

Thus, the node robustness checking problem can be expressed as follows.

Problem 1: Check if $V \leq V_m$ is satisfied for all voltages \mathbf{V} that satisfy $\mathbf{LGV} \leq \mathbf{I}_m, \mathbf{V} \geq 0$.

Notice that the system equation $\mathbf{I} = \mathbf{G}\mathbf{V}$ is implicitly satisfied by the first n rows of the matrix inequality $\mathbf{LGV} \leq \mathbf{I}_m$. This is because, as was expressed in relation to (35), that set of inequalities covers all the nodes, and any node with no current source attached is assigned a fixed zero-current upper bound constraint.

F. Linear Programming

It is significant that the constraints (39) are linear, and we propose to construct a linear program (LP) around them as a way to check robustness. We will refer to the space of voltages represented by these constraints as the feasible space. Note that this space, being bounded by joint linear constraints, is convex, as in a standard LP. Thus, one way of checking robustness is to take the nodes one at a time and solve an LP every time in which the objective is to maximize the voltage at that node, i.e.,

$$\begin{aligned} &\text{maximize} && v_k \\ &\text{subject to} && \mathbf{LGV} \leq \mathbf{I}_m \\ &&& \mathbf{V} \geq 0. \end{aligned} \quad (40)$$

As one solves the LP, one can of course stop when V_k exceeds a V_m value, and declare a violation. If, instead, the maximum voltage at that node is less than V_m , then that node is “safe” and one can then switch to another node and start solving a new LP with the new objective function. One can keep doing this until a violation is found or all nodes have been proven safe.

V. COMBINED APPROACH

Recall that the STA that was presented in Section III assumes that all supply nodes have independent voltages. In Section IV, we saw how the relationships among these voltages imposed by the power grid can be captured by an optimization approach in a way that does not depend on complete knowledge of the currents that load the grid. We will now combine the two techniques, leading to a voltage-aware STA that respects the relationships among the supply nodes imposed by the grid. The resulting approach will also be an optimization, in which we look for the worst case arrangement of the supply nodes allowed by the grid and by the current constraints.

This will be done on a per-critical-path basis. Given a critical path, resulting from the first-phase STA, and considering the supply taps of all the gates on that path, we will look for the true maximum delay of that path under all possible loading currents that satisfy the current constraints. The path delay t_d is a polynomial function of the voltages, as in (3), so that the problem becomes a nonlinear programming (NLP) problem in the form of

$$\begin{aligned} &\text{maximize} && t_d = f(\mathbf{V}) \\ &\text{subject to} && \mathbf{LGV} \leq \mathbf{I} \\ &&& \mathbf{I}_{vdd} = \mathbf{I}_{gnd} \\ &&& \mathbf{V} \geq 0 \end{aligned} \quad (41)$$

where $f(\mathbf{V})$ is a nonlinear quadratic function of the voltage vector and \mathbf{G} is the conductance matrix of both power and ground grids. In order to have correct dc formulation, we have introduced the constraint $\mathbf{I}_{vdd} = \mathbf{I}_{gnd}$, which ensures that whatever current is consumed from power is sunk by the ground grid. This constraint set is mapped into the voltage domain as $\mathbf{G}_{vdd}\mathbf{V}_{dd} - \mathbf{G}_{gnd}\mathbf{V}_{gnd} = 0$.

Consider the delay of two consecutive gates on a path whose supply and ground nodes are unique nodes on the grid. Fig. 12 shows that the delay of two such gates is always monotone in variations of power supply and ground voltages and the sensitivity is either positive or negative depending to the signal polarity. Delay sensitivities of all gates and gate combinations in our library (for example as in Fig. 10) have been checked, and it is confirmed that the sensitivity of the delay to a given voltage variable does not change sign as that voltage is varied across the whole range. Further, it was observed that for the valid feasible space of optimization, the quadratic function modeling delay has near-linear characteristics. A similar finding of delay in terms of voltage supply being near linear was shown in [15].

In order to find the worst case delay of a path, we solve for the maximum of the path delay as a function of voltages using the SNOPT solver [16], which can only find local maxima. Strictly speaking, there are no efficient techniques that

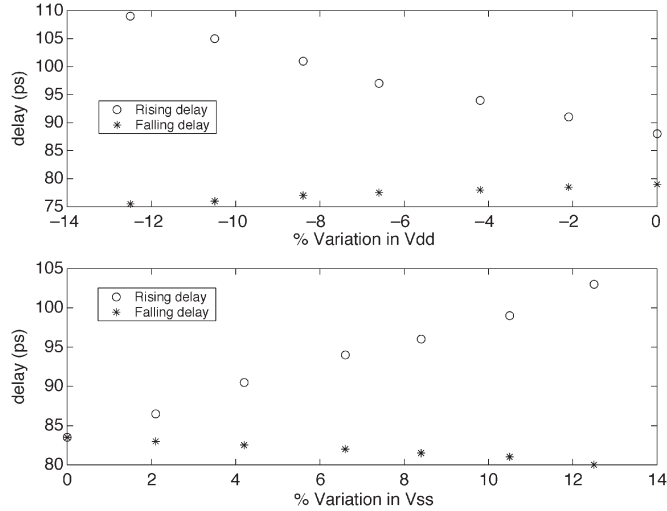


Fig. 12. Delay versus voltage of two-gate system.

guarantee finding the global maximum of a constrained quadratic problem such as ours. However, given the near-linear behavior of the function in our case, empirical evidence has shown that SNOPT actually finds the true worst case delay. In any case, and if this proves to be an issue in practice, there is a fall-back position: One can easily replace the quadratic function of every gate on the path by a linear surface that dominates it everywhere (i.e., is larger than the quadratic at all points), then add these to replace the quadratic path delay function by a linear function, which can then be easily maximized to give a tight conservative estimate of the true maximum. This alternative approach would work well because the quadratics are near linear to begin with, but, as mentioned, we did not see the need to go to this approach in our (quadratic function based) implementation. To improve efficiency, we pregenerate the functions of all the gradients of our objective function. Since only the objective function is nonlinear, the problem is linearly constrained, which tends to solve more easily than general nonlinear programs with nonlinear constraints. The solver uses a sparse sequential quadratic programming method using limited memory quasi-Newton approximations to the Hessian of the Lagrangian.

Starting with the most critical paths first, we go down the list and apply the optimization. Notice that optimization can only reduce the delay of that path, which was estimated in the first-phase STA. If $t_{d(STA)}$ is the delay estimate for a path obtained in the first-phase STA and $t_{d(NLP)}$ is the delay estimate of that same path after the NLP optimization step, then $t_{d(NLP)} \leq t_{d(STA)}$, and we may view $t_{d(NLP)}$ as being the corrected delay of that path. In this way, we proceed down the list of critical paths. When we encounter a path whose $t_{d(NLP)}$ is greater than $t_{d(STA)}$ of the next path on the list, we are done and $t_{d(NLP)}$ is the worst case delay of that circuit.

VI. EXPERIMENTAL RESULTS

Our technique was implemented and tested on ISCAS85 and the combinational parts of ISCAS89 benchmarks. Experiments were run on a 1-GHz Sun machine with 4-GB memory. The execution time of first-phase STA was very fast, under 12 s for every circuit that we tested, and less than 1 s for most of them.

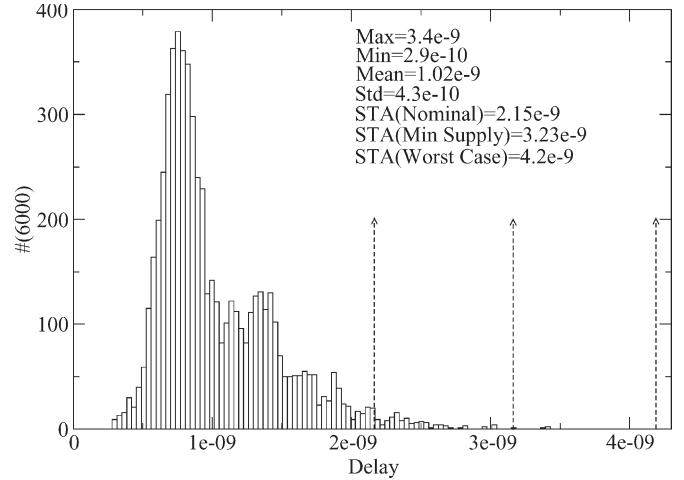


Fig. 13. C880 delay with falling outputs.

The results of the analysis of circuit C880 with independent power supplies and grounds, shown in Fig. 13, illustrate a key point. The figure shows a histogram of the circuit delay (using HSPICE) for 6000 different input vector pairs, with a worst case setting of the supply voltages (within their allowable ranges), as identified by our STA for that circuit. This does not exhaustively cover all vector pairs for this circuit, but will help illustrate the point. The figure also shows the circuit delay as measured by our STA using three different settings for the supplies. The first setting (Nominal) gives the circuit delay when all supplies are set at their nominal (ideal, no voltage drop) values. It is clear from the figure that this significantly underestimates the circuit delay. The second (Min Supply) setting corresponds to the case when all V_{dd} supplies are set to low and all grounds to high, within their allowable ranges. This case corresponds to what one is able to do today with existing STA tools. Here too, it is clear that this analysis is not adequate because there are paths with longer delay than that given by the Min Supply setting. Finally, the third setting corresponds to the case where our STA considers all possible mismatches between the supply nodes and finds the maximum delay, in this case assuming that all supplies are independent. Note that there are no vector pairs that violate our estimate of worst case delay.

Further results on all the benchmarks are presented in Table III. This table gives the delay values measured by our STA and by HSPICE in the three cases of Nominal, Min Supply, and Worst-Case, explained above. The percentage values given in parentheses represent the relative increase of delay over the Nominal case. Getting the exact delay using HSPICE is not possible because of the large number of possible vector pairs. Therefore, for each circuit, once the critical path is identified by our STA, we extract that path and simulate it with HSPICE. Notice that the critical path may be different in the Nominal, Min Supply, and Worst-Case scenarios.

Notice that the delays under the SPICE Min Supply column are higher than the delays of the Nominal case. The advantage of our technique and the need for it are evident from the last column (SPICE, Worst-Case). The significant increase of delay over Nominal and over Min Supply underscores the fact that allowing mismatch between supplies leads to a higher worst

TABLE III
STA AND HSPICE WORST CASE DELAY FOR FULL-RANGE CASE

Circuit	STA(ns)			SPICE(ns)		
	Nominal	Min Supply	Worst-Case	Nominal	Min Supply	Worst-Case
C1355	2.48	3.83	4.54 (4.6%)	2	3.41 (70%)	4.34 (117%)
C1908	3.15	4.96	5.9 (1%)	2.8	4.71 (68%)	5.84 (108%)
C2670	3.39	5.44	5.82 (8.5%)	3	5 (66%)	5.36 (78%)
C3540	4.26	6.88	7.43 (13%)	3.5	5.9 (68%)	6.55 (97%)
C432	4.33	6.67	7.22 (1.7%)	3.6	6.27 (74%)	7.1 (97%)
C499	1.97	3.13	3.67 (-4.2%)	1.8	3 (66%)	3.8 (111%)
C5315	3.7	5.91	6.79 (4.3%)	3.3	5.45 (65%)	6.51 (97%)
C7552	2.89	4.56	5.48 (-1%)	2.7	4.41 (63%)	5.54 (105%)
C880	2.15	3.23	4.2 (0.6%)	1.85	2.97 (60%)	4.16 (125%)
S1494	1.78	2.86	3 (1.7%)	1.68	2.75 (63%)	2.95 (76%)
S420	1.23	1.88	2.13 (6.5%)	1	1.8 (80%)	2 (100%)
S444	1.13	1.67	2.16 (-6.5%)	.96	1.6 (67%)	2.31 (140%)
S510	.99	1.48	1.85 (2.8%)	.83	1.32 (59%)	1.8 (116%)

case delay. Finally, notice that the delay comparisons between the corresponding columns of STA and HSPICE are very good and show that the gate delay model works well in this case.

Not having access to power grids from industrial designs and in order to test our approach under different conditions, we have opted to generate a number of grids ourselves. The grid generation process is automatic and employs a random number generator as well as user-specified technology and topology parameters. Starting with a square uniform grid of a given size, we proceed to randomly delete a user-specified percentage of nodes, thus rendering the grid structurally nonuniform. Typical geometric and physical grid characteristics (e.g., grid dimensions) as well as characteristics of the fabrication process (e.g., sheet resistance of a particular level of metallization) are given by the user, leading to an initial value of the conductance of every branch. When a node is deleted, the conductances of the remaining surrounding edges (branches) are increased by a random amount around a user-specified percentage of their initial values. The rationale behind this is to allow the non-uniform grid to be loaded with currents comparable to its uniform predecessor while exhibiting comparable IR drops. The numbers of V_{dd} sites and current sources are supplied by the user and are then distributed at random over the grid nodes. The supplies of the critical paths extracted from ISCAS benchmarks were then randomly connected to our power grids. This random process of circuit to power grid connection was done in order to best emulate all the possible designs that could be encountered from critical paths within specific blocks to paths that may span the geometry of the entire chip.

For verifying individual node voltages, we have improved on what was reported in [2] by implementing an Interior Point Method with sparse matrix techniques. As a result, the time required for one check of a node voltage is in the order of half a minute or so for the larger sized grids as shown in Table IV. This check may be easily extended to larger grids.

Table V shows some of our STA results. A number of benchmark critical paths randomly connected to varying sized power grids, from 1000 to 40 000 nodes, were simulated using our NLP approach. The worst case delay found under the influence of power grid is smaller than that found using STA analysis with independent supplies and typically falls around the neighborhood of the SPICE min. analysis. The difference

TABLE IV
NODE VOLTAGE ROBUSTNESS VERIFICATION COMPUTATION
TIMES WITH INTERIOR POINT METHOD

Grid Size	avg. time/node	avg. time/path
1320	0.3s	23s
5832	3.2s	4.8min.
10073	8.6s	10min
27055	31.5s	25min
43106	98.1s	2.7h

is seen to vary between -30% and 8% . The computation time for solving each worst case circuit delay time is seen to be a minimal 1–130 s. This reported time is only the time required to solve for the optimal solution of each critical path. It does not include the time required to perform preconditioning on the linear component of the problem, which may run in the order of 10–15 min for larger sized grids. This computational time overhead, however, is only required once for any power grid. Further, it was observed that our technique used about 100 MB of memory for the large grids, thus may be easily applied to even larger grids.

It is interesting to note the difference between our NLP calculation and the delay calculated by SPICE using min. supply. In general, for power grids that are symmetric between their V_{dd} and V_{ss} planes, if we are working with robust grids, it is a safe assumption to expect a delay that will be less than the min. supply as the results of Table V indicate. However, one should also notice that for the case of circuits C499 and C5315, as the same nonuniform and asymmetric grids were used for both circuits, we were able to find a delay that was more than that of the min. SPICE supply analysis. This shows that our technique, given real placement, will provide a more accurate measure of the worst case delay associated with a critical path, and if no placement is available then NLP analysis using voltage drops and random placement will give a good indication of the worst possible conditions.

VII. CONCLUSION

In today's integrated circuit designs, timing and its sensitivity to supply voltage fluctuations are key concerns. Analysis of voltage variations by simulation is a complicated task due to the requirement of stimulus (vectors, patterns, waveforms) in order to complete the simulation. It is hard in practice to obtain

TABLE V
NLP WORST CASE DELAY WITH GRID

Circuit	# of paths extracted	average # of gates/path	# of Powergrid nodes	min. supply SPICE (ns)	worst-case $t_{d(NLP)}$ (ns)	# of paths analyzed	Difference NLP vs. SPICE	NLP solution time (sec.)
C1355	1	28	1320	3.41	3.27	1	-4.1%	0.95
C1908	7	43	1320	4.71	4.34	6	-7.8%	5.75
C2670	3	36	1320	5.0	3.70	3	-26.0%	2.67
C3540	17	48	5832	5.9	5.08	12	-13.9%	25.32
C432	5	43	5832	6.27	6.15	3	-1.9%	7.62
C499	1	26	12740	3	3.11	1	3.7%	2.74
C5315	5	42	12740	5.45	5.87	3	7.7%	9.13
C7552	20	37	27055	4.41	4.13	9	-6.4%	47.8
C880	8	25	27055	2.97	2.32	8	-21.9%	43.4
S420	9	11	27055	1.8	1.27	9	-29.4%	42.8
S510	7	11	43106	1.32	1.19	6	-9.9%	47.4
C6288	21	113	43106	15.6	14.1	19	-9.6%	131.2

such stimulus. Further, even if it were made available, the simulation would be required to run for prolonged periods of time with high computational cost overhead. We have proposed a method whereby we abstract circuit behavior in the form of user-supplied current constraints. By using a delay model that is expressed in the form of supply voltage variations of the path and running a nonlinear program, we may solve for the worst case time delay.

REFERENCES

- [1] R. Ahmadi and F. N. Najm, "Timing analysis in presence of power supply and ground voltage variations," in *Proc. Int. Conf. Computer-Aided Design*, San Jose, CA, 2003, pp. 176–183.
- [2] D. Kouroussis and F. N. Najm, "A static pattern-independent approach for power grid voltage integrity verification," in *Proc. Design Automation Conf.*, Anaheim, CA, 2003, pp. 99–104.
- [3] B. Widrow, *Adaptive Signal Processing*, 1st ed. Englewood Cliffs, NJ: Prentice-Hall, 1985.
- [4] J. Rubenstein, P. Penfield, Jr., and M. A. Horowitz, "Signal delay in RC tree networks," *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, vol. CAD-2, no. 3, pp. 202–211, Jul. 1983.
- [5] C. L. Ratzlaff and L. T. Pillage, "RICE: Rapid interconnect circuit evaluation using AWE," *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, vol. 13, no. 6, pp. 763–776, Jun. 1994.
- [6] N. Hedenstierna and K. O. Jeppson, "CMOS circuit speed and buffer optimization," *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, vol. CAD-6, no. 2, pp. 270–281, Mar. 1987.
- [7] T. I. Kirkpatrick and N. R. Clark, "PERT as an aid to logic design," *IBM J. Res. Develop.*, vol. 10, no. 2, pp. 135–141, Mar. 1966.
- [8] R. B. Hitchcock, S. G. L. Smith, and D. D. Cheng, "Timing analysis of computer hardware," *IBM J. Res. Develop.*, vol. 26, no. 1, pp. 100–105, Jan. 1982.
- [9] D. Blaauw, V. Zolotov, and S. Sundareswaran, "Slope propagation in static timing analysis," *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, vol. 21, no. 10, pp. 1180–1195, Oct. 2002.
- [10] L. T. Pillage, R. A. Rohrer, and C. Visweswaraiiah, *Electronic Circuit and System Simulation Methods*. New York: McGraw-Hill, 1995.
- [11] G. H. Golub and C. F. V. Loan, *Matrix Computations*. Baltimore, MD: The Johns Hopkins Univ. Press, 1996.
- [12] A. Berman and R. J. Plemmons, *Nonnegative Matrices in the Mathematical Science*. New York: Academic, 1996.
- [13] H. Kriplani, F. N. Najm, and I. Hajj, "Pattern independent maximum current estimation in power and ground buses of CMOS VLSI circuits: Algorithms, signal correlations, and their resolution," *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, vol. 14, no. 8, pp. 998–1012, Aug. 1995.
- [14] J. Rubenstein, P. Penfield, and M. A. Horowitz, "Signal delay in RC tree networks," *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, vol. CAD-2, no. 3, pp. 202–211, Jul. 1983.
- [15] S. Pant, D. Blaauw, V. Zolotov, S. Sundareswaran, and R. Panda, "Vectorless analysis of supply noise induced delay variation," in *Proc. Int. Conf. Computer-Aided Design*, San Jose, CA, 2003, pp. 184–191.
- [16] P. Gill, W. Murray, and M. Saunders, "SNOPT: An SQP algorithm for large-scale constrained optimization," *SIAM J. Optim.*, vol. 12, no. 4, pp. 979–1006, 2002.



Dionysios Kouroussis (S'01–M'05) received the B.A.Sc. and M.A.Sc. degrees in electrical and computer engineering (ECE) from the University of Toronto, Toronto, ON, Canada, in 1996 and 1998, respectively, where he is currently working toward the Ph.D. degree in ECE.

From 1998 to 2000, he was an ASIC Design Engineer at ATI Technologies, Thornhill, Ontario. In 2005, he rejoined ATI in the ASIC Methodology group and is currently working on power grid verification and synthesis, as well as leakage reduction techniques using power gating.



Rubil Ahmadi (M'04) received the B.Sc. degree from Sharif University of Technology, Tehran, Iran, in 2000, and the M.A.Sc. degree from the University of Toronto, Toronto, ON, Canada, in 2003, both in electrical and computer engineering.

In 2004, he was an Engineer at ATI Technologies Inc., Toronto. He is currently working on hardware modeling, technology planning, and computer-aided design (CAD) methodology for high density ASIC design.

Mr. Ahmadi is a member of the Professional Engineers Ontario (PEO).



Farid N. Najm (S'85–M'89–SM'96–F'03) received the B.E. degree in electrical engineering from the American University of Beirut (AUB), Beirut, Lebanon, in 1983, and the M.S. and Ph.D. degrees in electrical and computer engineering (ECE) from the University of Illinois at Urbana-Champaign (UIUC), Urbana in 1986 and 1989, respectively.

From 1989 to 1992, he was with Texas Instruments, Dallas, TX. Then he joined the ECE Department at UIUC as an Assistant Professor, becoming an Associate Professor in 1997. In 1999, he joined

the ECE Department at the University of Toronto, Toronto, ON, Canada, where he is currently a Professor and the Vice-Chair of ECE. He coauthored *Failure Mechanisms in Semiconductor Devices* (2nd Ed., Wiley, 1997). His research is on computer-aided design (CAD) for integrated circuits, with emphasis on circuit-level issues related to power dissipation, timing, and reliability.

Dr. Najm is an Associate Editor for the IEEE TRANSACTIONS ON COMPUTER-AIDED DESIGN OF INTEGRATED CIRCUITS AND SYSTEMS (CAD). He received the IEEE TRANSACTIONS ON CAD Best Paper Award in 1992, the National Science Foundation (NSF) Research Initiation Award in 1993, the NSF CAREER Award in 1996, and was the Associate Editor for the IEEE TRANSACTIONS ON VERY LARGE SCALE INTEGRATION SYSTEMS (VLSI) from 1997 to 2002. He served as the General Chairman for the 1999 International Symposium on Low-Power Electronics and Design (ISLPED-99) and as Technical Program Co-Chairman for ISLPED-98. He has also served on the technical committees of ICCAD, DAC, CICC, ISQED, and ISLPED.