# Modeling the Effects of Temporal Proximity of Input Transitions on Gate Propagation Delay and Transition Time 

V. Chandramouli and Karem A. Sakallah<br>Department of Electrical Engineering and Computer Science<br>University of Michigan


#### Abstract

While delay modeling of gates with a single switching input has received considerable attention, the case of multiple inputs switching in close temporal proximity is just beginning to be addressed in the literature. The effect of proximity of input transitions can be significant on the delay and output transition time. The few attempts that have addressed this issue are based on a series-parallel transistor collapsing method that reduces the multi-input gate to an inverter. This limits the technique to CMOS technology. Moreover, none of them discuss the appropriate choice of voltage thresholds to measure delay for a multi-input gate. In this paper, we first present a method for the choice of voltage thresholds for a multi-input gate that ensures a positive value of delay under all input conditions. We next introduce a dual-input proximity model for the case when only two inputs of the gate are switching. We then propose a simple approximate algorithm for calculating the delay and output transition time that makes repeated use of the dual-input proximity model without collapsing the gate into an equivalent inverter. Comparison with simulation results shows that our method performs quite well in practice.


## I. INTRODUCTION

Development of accurate delay models continues to be a critical need for high-performance VLSI applications. The combined effect of submicron feature sizes and larger die areas are forcing a reassessment of the conventional models for gate and interconnect delays. The earliest gate delay models accounted for such effects as load capacitance and transistor sizes [3]. More recently, the dependence of delay on the finite transition times of digital signals has been recognized by several researchers, $[9,12,16]$ to name a few, and incorporated in commercial delay calculators [19, 20]. In this paper, we address the dependence of gate delay on the temporal proximity of input transitions. This effect has been identified by some researchers $[10,15]$ and represents a form of state-dependency [8, 18]. To date, however, the modeling approaches proposed to capture proximity effects are incomplete, inaccurate, or specific to particular design styles. The proximity model we introduce in this paper attempts to remedy these shortcomings.

To illustrate the effect of transition proximity on gate delay consider the three-input CMOS NAND gate shown in Fig. 1(a) and assume that inputs $a$ and $b$ experience, respectively, slow and fast falling transitions while input $c$ is stable at $V_{d d}$. Fig. 1(b) depicts the variation of gate delay as a function of the temporal separation between the transitions on $a$ and $b$. For sufficiently large separations, the transition on $b$ is
blocked by the controlling value on $a$ (logic 0 ) and does not affect gate delay. As the separation decreases, however, the p-transistor connected to $b$ starts to conduct and provides another current path from $V_{d d}$ to the output. As a result, the output rises faster and the effective gate delay is reduced. As the figure shows, the reduction in gate delay due to this proximity phenomenon can be significant. A similar effect can be observed for the rise time on the output. Consider next the case when inputs $a$ and $b$ experience rising transitions while input $c$ is stable at $V_{d d}$. As Fig. 1(c) shows, gate delay becomes a decreasing function of separation. The output fall time shows a similar behavior. This can be readily explained by examining the behavior of the $n$-transistors in the pulldown stack. However, when two inputs transition in opposite directions, the output may glitch as shown in Fig. 2(a). Fig. 2(b) shows the minimum voltage at the output as a function of the separation between $a$ and $b$, for the circuit shown in Fig. 1. Inertial delay can, thus, be viewed as another manifestation of temporal proximity. Due to space limitations, we will illustrate our modeling approach for propagation delay only. Transition time and inertial delay are handled similarly.

It should be clear from this simple example that the variation in delay due to temporal proximity can be significant and should be modeled if accurate delay estimations are sought. However, while the case of single-input switching has received a lot of attention ( $[6,13,17]$ ), the proximity effect is just starting to be addressed in the literature. In [10], the waveforms of the switching inputs are replaced by an equivalent waveform and the multi-input gate is collapsed into an inverter by series-parallel reduction of the transistors. The justification for deriving the equivalent waveform are not clearly stated. In addition, the output loading and input transition times are not taken into account while collapsing the transistors possibly leading to large errors. In [15], an attempt has been made to remove this deficiency. However, the authors' main focus is on calculating the peak supply current and percentage errors for delay and output transition times are not given. Both these techniques give significant errors when we measure delay and output transition time. In this paper we propose a novel technique for computing the delay and output transition times that does not collapse a multi-input gate to an inverter. While our technique can be applied to any technology, we illustrate it with CMOS technology in this paper.


Fig. 1. Proximity effect for a 3-input CMOS NAND gate

(a) NAND gate with opposite going transitions

(b) Minimum voltage at gate output

Fig. 2. Relationship between inertial delay and proximity effect

The rest of this paper is organized as follows. In Section 2, we describe how to choose appropriate voltage thresholds for multi-input gates to ensure positive delays. In Section 3, we formulate the delay and output transition time functions for a two-input gate and propose a temporal proximity model for the gate. In Section 4, we develop the proximity model for a multi-input gate using this dual-input proximity model. The experimental validation of our model for a three-input NAND gate is presented in Section 5. We base our comparisons on circuit simulations performed using HSPICE [14]. We conclude the paper in Section 6, by summarizing our contribution and indicating future work.

## II. DEFINING DELAY THRESHOLDS

Delay is measured from the time when an input signal crosses a certain voltage (input threshold) to the time when the output signal crosses another voltage (output threshold). To ensure causality, delay must always be positive. Therefore, it is important that the delay measurement thresholds be chosen carefully. As has been noted by a few researchers [2, 7], delay can become negative for very slow inputs, if measured using $V_{d d} / 2$ as the threshold. To overcome this problem, some researchers have proposed measuring delay using the unity differential gain points on the Voltage Transfer Curve (VTC) of the gate [5, 12]. A typical Voltage Transfer Curve (VTC) for an inverter is shown in Fig. 3(a). $V_{i l}$ and $V_{i h}$ denote the points where the slope of the VTC is -1 and $V_{m}$ denotes the DC gain threshold of the gate. Thus, delay is measured using $V_{i l}\left(V_{i h}\right)$ for the input threshold and $V_{i h}\left(V_{i l}\right)$ for the output threshold in case of rising (falling) inputs. This definition of delay always gives a monotonically increasing
delay with increasing input transition time. In fact, it can be shown that for slow inputs, a choice of input threshold greater than (less than) $V_{m}$ for a rising (falling) input can give rise to negative delays whereas using $V_{m}$ itself leads to a value of zero delay. Further, it can also be shown that $V_{i l}$ and $V_{i h}$ are better candidates for measuring signal transition times than the usual $10 \%-90 \%$ points.

However, in the case of multi-input gates with $n$ inputs, rather than a single VTC, there are $2^{n}-1$ VTCs corresponding to all possible combinations of stable and switching inputs. Thus, there are $2^{n}-1$ sets of $V_{i l}, V_{i h}$ and $V_{m}$ values. We propose using the minimum value of $V_{i l}$ and the maximum value of $V_{i h}$ from these $2^{n}-1$ sets of values so that the delay will be positive under any combination of input transition times and temporal separations. This is so because the input threshold will be less (more) than any of the $2^{n}-1 V_{m}$ values for rising (falling) inputs thus ensuring positive delay. To illustrate our point, Fig. 3(b) shows the VTCs obtained by circuit simulation of the gate in Fig. 1(a). The curve for the case when $a$ is switched alone and the curve for the case when all of them switch together are the two extreme cases of this family of curves. Therefore, we would choose our $V_{i l}$ and $V_{i h}$ values from the VTC obtained when $a$ is switched alone and the VTC obtained when all the inputs switch together, respectively. These values were found to be 1.25 V and 3.37 V for the circuit shown in Fig. 1(a). In general, for a NAND gate, the $V_{i l}$ would be chosen from the input closest to the ground and $V_{i h}$ would be chosen from the VTC corresponding to all inputs switching at the same time. In the case of NOR gates, $V_{i l}$ would be chosen from the VTC corresponding to all inputs switching at the same time and $V_{i h}$


Fig. 3. Choosing the delay measurement thresholds
chosen from the input closest to the power rail.

## III. A DUAL-INPUT TEMPORAL PROXIMITY MODEL

In this section, we derive the proximity macromodel for the propagation delay of two-input gates. Starting from a complete enumeration of all waveform and circuit parameters that can affect delay, we show how this macromodel can be expressed as a three-argument function. This derivation is based on the application of dimensional analysis and the invocation of reasonable simplifying assumptions.

Consider the black-box model of a two-input CMOS gate shown in Fig. 4. As shown in [11], the macromodel for the case when only one input switches is of the following form:

$$
\begin{equation*}
\frac{\Delta_{1 z}^{(1)}}{\tau_{1}}=D_{1}^{(1)}\left(\frac{C_{L}}{K_{n} V_{d d} \tau_{1}}\right) \tag{1}
\end{equation*}
$$

where $\tau_{1}$ is the transition time of $x_{1}$ (assuming $x_{1}$ is the switching input), $\Delta_{1 z}^{(1)}$ is the delay measured with respect to $x_{1}$ (in general $\Delta_{i z}^{(n)}$ is the delay when $n$ inputs are switching with input $x_{i}$ being the most dominant one), $C_{L}$ is the total load capacitance, $V_{d d}$ is the power supply voltage and $K_{n}$ is the strength ${ }^{1}$ of the n-transistor whose gate is connected to $x_{1}$.

When both inputs are considered, the delay macromodel in its most general form can be written as:

$$
\begin{equation*}
\Delta_{1 z}^{(2)}=D_{1}^{(2)}\left(\tau_{1}, \tau_{2}, s_{12}, C_{L}, V_{d d}, K_{n 1}, K_{n 2}, K_{p 1}, K_{p 2}, V_{t n}, V_{t p}\right) \tag{2}
\end{equation*}
$$

where the delay has been measured with respect to the dominant input $x_{1}$ (to be explained later). We can make some reasonable assumptions for simplifying this function. In most designs, all n -transistors are of the same size as are all p transistors. Therefore, the individual transistor strengths can be replaced by one parameter for the n-transistors and one for the p-transistors. Further, by viewing the proximity effect as a perturbation of the delay due to a single dominant input

[^0]
transition, we can rewrite (2) as:
\[

$$
\begin{equation*}
\Delta_{1 z}^{(2)}=D_{1}^{(2)}\left(D_{1}^{(1)}\left(\tau_{1}, C_{L}, V_{d d}, K_{n}, K_{p}, V_{t n}, V_{t p}\right), \tau_{1}, \tau_{2}, s_{12}\right) \tag{3}
\end{equation*}
$$

\]

Thus, we have separated the effects of the temporal and nontemporal parameters on delay. By using dimensional analysis, we can reduce (3) to the following form which has only four parameters [11]:

$$
\begin{equation*}
\frac{\Delta_{1 z}^{(2)}}{\Delta_{1 z}^{(1)}}=D_{1}^{(2)}\left(\frac{\tau_{1}}{\Delta_{1 z}^{(1)}}, \frac{\tau_{2}}{\Delta_{1 z}^{(1)}}, \frac{s_{12}}{\Delta_{1 z}^{(1)}}\right) \tag{4}
\end{equation*}
$$

So far we have not distinguished between the two inputs in any way. However, a key assumption while deriving (4) was that the effect of proximity of input transitions should be a perturbation on the delay due to a single dominant input. To satisfy this assumption the correct identification of the dominant input among all the inputs is critical. The technique to do this for a two-input NAND gate is shown in Fig. 5. Consider the case when the slower input arrives first (shown in solid) and the faster input (shown dashed) arrives a little later. The rising waveforms $z_{a}$ and $z_{b}$ show the corresponding outputs when each of the inputs is switching by itself. The waveform $z_{a b}$ is the output response due to both inputs. Clearly, it is more appropriate to view input $b$ as the dominant one because the time when $z_{a b}$ crosses $V_{i h}$ (i.e. completes its transition) is closer to the time when $z_{b}$ crosses $V_{i h}$ rather than to the time when $z_{a}$ crosses $V_{i h}$. This agrees with our notion of proximity being a perturbation on the output produced when $b$ alone is switching. Thus, even though $a$ arrives first, it is input $b$ that is identified as the dominant one. However, there is a minimum separation equal to $\left(\Delta_{a z}^{(1)}+\tau_{a z}^{(1)}\right)-\left(\Delta_{b z}^{(1)}+\tau_{b z}^{(1)}\right)$ after which $a$ becomes the dominant input. This is so because beyond this separation, the time when $z_{a}$ crosses $V_{i h}$ will be closer to the time when $z_{a b}$ crosses $V_{i h}$. This implies that the minimum separation could be negative when $\Delta_{b z}^{(1)}+\tau_{b z}^{(1)}>\Delta_{a z}^{(1)}+\tau_{a z}^{(1)}$. Note that this also takes the position of the inputs in the series transistor stack into account since the delays could be different, even for the same transition times for the inputs. Therefore, for a given separation between the two inputs and their transition times, we first determine the dominant input and then use (4) to determine the delay with respect to the dominant input. Thus, if the original inputs are ordered in terms of the one crossing the $V_{i h}$ threshold first, we find a new ordering in terms of the one having the most effect on the output waveform. An analogous argument can be made for the case when the two inputs are rising. While this discussion has centered


Fig. 5. Identifying the dominant input


Fig. 6. Change of dominant input with separation
around a NAND gate, similar arguments can be made for a NOR gate.

Based on Fig. 5 we can also determine the maximum separation between the two inputs beyond which the proximity effect becomes unimportant. We see that for $s_{a b}>\Delta_{a z}^{(1)}$, the transitions on $b$ can be ignored and the delay will be the same as when $a$ was alone. We define this as the proximity window for $b$ to have any effect on the delay. However, $b$ may still influence the transition time on $z$. Its is only when $s_{a b}>\Delta_{a z}^{(1)}+\tau_{a z}^{(1)}$, that the effect of $b$ can be ignored. This then defines the proximity window for $b$ to have any influence on the output transition time. Similar arguments apply when $b$ is the dominant input.

Fig. 6 shows the data obtained from a circuit simulation of the circuit shown in Fig. 1, with inputs $a$ and $b$ falling and $c$ tied to $V_{d d} . \tau_{a}$ was fixed at 0.5 ns and $\tau_{b}$ was fixed in turn at $0.1 \mathrm{~ns}, 0.5 \mathrm{~ns}$ and 1 ns . In each case, $s_{a b}$ was varied from $-\left(\Delta_{b z}^{(1)}+\tau_{b z}^{(1)}\right)$ to $\Delta_{a z}^{(1)}+\tau_{a z}^{(1)}$ and the resultant delay plotted. Also shown is the actual crossover point when the causing

## Algorithm ProximityDelay;

. Relabel inputs to $y_{1} \ldots y_{n}$ such that for any two inputs $y_{i}, y_{j}$, $i<j$ iff $s_{y_{i} y_{j}}>\left(\Delta_{y_{i} z}^{(1)}+\tau_{y_{i} z}^{(1)}\right)-\left(\Delta_{y_{j} z}^{(1)}+\tau_{y_{j} z}{ }^{(1)}\right)$;
$i=2$;
while $\left(i \leq n \& \& \quad s_{y_{1} y_{i}}<\Delta_{y_{1} z}^{(i-1)}\right)\{$
4.

$$
\Delta_{y_{1} z}^{(i)}=\Delta_{y_{1} z}^{(i-1)}+\Delta_{y_{1} z}^{(1)}\left[D_{y_{1}}^{(2)}\left(\frac{\tau_{y_{1}}}{\Delta_{y_{1} z}^{(1)}}, \frac{\tau_{y_{i}}}{\Delta_{y_{1} z}^{(1)}}, \frac{s_{y_{1} y_{i}}+\Delta_{y_{1} z}^{(1)}-\Delta_{y_{1} z}^{(i-1)}}{\Delta_{y_{1} z}^{(1)}}\right)-1\right]
$$

5. $i=i+1 ; / *$ end of while */
6. $i=i-1$;

Fig. 7. Algorithm for computing delay of multi-input gates
input changes, for the case when $\tau_{b}$ was 1 ns . We note that there is a discontinuity in the delay value when the dominant input changes. This is because our reference for measuring delay also changes from input $b$ to input $a$.

Thus, for a two-input gate our delay macromodel is of the form (4). That this macromodel is indeed a function is apparent from the graphs in Fig. 1 and Fig. 6. The actual form of the macromodel is implementation dependent: we could either have a lookup-table or an analytical formula. We next describe the modeling approach for gates with more than two inputs.

## IV. MULTI-INPUT TEMPORAL PROXIMITY MODEL

For an $n$-input gate, equation (4) extends in a straightforward way to:

$$
\begin{equation*}
\frac{\Delta_{i z}^{(n)}}{\Delta_{i z}^{(1)}}=D_{i}^{(n)}\left(\frac{\tau_{1}}{\Delta_{i z}^{(1)}}, \ldots, \frac{\tau_{n}}{\Delta_{i z}^{(1)}}, \frac{s_{i 1}}{\Delta_{i z}^{(1)}}, \ldots, \frac{s_{i n}}{\Delta_{i z}^{(1)}}\right) \tag{5}
\end{equation*}
$$

where $i$ is the most dominant input. Equation (5) has $2 n-1$ parameters and developing a macromodel involving $2 n-1$ parameters is very hard. A closed analytical formula may be impossible to obtain which would force one to use a tablelookup approach. However, the size of these tables would make them impractical. We need to reduce the number of arguments to these functions in order to make the macromodel construction practical. Since all the quantities in these equations have the unit of time, dimensional analysis fails to reduce the number of arguments. Therefore, we need a way of decomposing these functions in terms of simpler, more manageable functions.

Rather than considering all inputs simultaneously, our technique is based on processing only two inputs at a time, starting from the two most dominant inputs. The resultant error from such an approximation is quite small in most cases. The algorithm for computing delay is presented in Fig. 7. The inputs are reordered in Step 1, based on their domi-
nance, by a straightforward extension of the dual-input case. The reordered inputs are labeled as $y_{1} \ldots y_{n}$. To apply the dualinput macromodel equation (4), the cumulative effect of inputs $y_{1} \ldots y_{i-1}$ is represented by an equivalent input waveform $\hat{y}(t)$ such that:

$$
\begin{equation*}
\hat{y}(t)=y_{1}\left(t+\Delta_{y_{1} z}^{(1)}-\Delta_{y_{1} z}^{(i-1)}\right) \tag{6}
\end{equation*}
$$

where $\Delta_{y_{1} z}^{(1)}$ is the delay due to the most dominant input acting alone and $\Delta_{y_{1} z}^{(i-1)}$ is the delay due to $y_{1 \cdots} y_{i-1}$. Equation (6) guarantees that the output waveform caused by $\hat{y}(t)$ crosses the delay measurement threshold at exactly the same time that the waveform due to $y_{1} \ldots y_{i-1}$ would. The effect of the next dominant input $y_{i}$ is now accounted for by applying the dual-input proximity macromodel to $\hat{y}$ and $y_{i}$ :

$$
\begin{equation*}
\Delta_{\hat{y} z}^{(2)}=\Delta_{y_{1} z}^{(1)} D_{y_{1}}^{(2)}\left(\frac{\tau_{y_{1}}}{\Delta_{y_{1} z}^{(1)}}, \frac{\tau_{y_{i}}}{\Delta_{y_{1} z}^{(1)}}, \frac{s_{\hat{y} y_{i}}}{\Delta_{y_{1} z}^{(1)}}\right) \tag{7}
\end{equation*}
$$

The delay $\Delta_{y_{1} z}^{(i)}$ due to $y_{1} \ldots y_{i}$ is easily obtained by changing the reference to $y_{1}$ using (6):

$$
\begin{equation*}
\Delta_{y_{1} z}^{(i)}=\Delta_{y_{1} z}^{(i-1)}+\Delta_{y_{1} z}^{(1)}\left[D_{y_{1}}^{(2)}\left(\frac{\tau_{y_{1}}}{\Delta_{y_{1} z}^{(1)}}, \frac{\tau_{y_{i}}}{\Delta_{y_{1} z}^{(1)}}, \frac{s_{y_{1} y_{i}}+\Delta_{y_{1} z}^{(1)}-\Delta_{y_{1} z}^{(i-1)}}{\Delta_{y_{1} z}^{(1)}}\right)-1\right] \tag{8}
\end{equation*}
$$

Equation (8) clearly shows that the delay due to the $i$ inputs that fall in the proximity window is a perturbation of the delay due to the $i-1$ most dominant inputs.

This process is repeated as long as there are inputs within the proximity window, which for the $i$ th iteration is given by $\Delta_{y_{1} z}^{(i-1)}$. Therefore, if $s_{y_{1} y_{i}}>\Delta_{y_{1} z}^{(i-1)}$, we stop processing any more inputs. We should note that the accuracy of the computation in our algorithm is highly dependent on the correct identification of dominant inputs. The maximum error occurs when the inputs can not be unambiguously ordered on dominance. Specifically: 1) when the inputs switch together with identical transition times and 2) when the dominant input arrives very late within the proximity window. The primary cause for such errors is the inapplicability of the input ordering based on dominance. In the first case, clearly there is no one input that dominates over others. However, when each input is considered by itself, there will be small differences in delays from each input to the output. Based on this, our algorithm will identify one of the inputs as the dominant one and proceed. This leads to errors, with the maximum error occurring when a step signal is applied to all the inputs at the same time. The only way to accurately model such cases is to take all inputs into account which as we have seen leads to a complicated macromodel. In the second case, the transition times and separations of $y_{2}$ through $y_{m}$, where $y_{m}$ is the last input that falls within the proximity window, are such that they affect the output noticeably and $y_{1}$ has the effect of merely hastening the output in crossing the delay measure-

$$
\begin{aligned}
& \begin{array}{l}
x_{1} \longrightarrow
\end{array} \begin{array}{l}
n \text {-input } \\
\text { gate }
\end{array}
\end{aligned} \rightarrow z
$$

Fig. 8. Computational complexity of the method
ment threshold. In such cases, again, our algorithm underestimates the roles of the other inputs and causes errors. In order to retain the simplicity of our approach and still get accurate results we added a corrective term to the delay value obtained by our method. As we show in the next section this gives satisfactory results. Details on how we apply the correction can be found in [4].

Since the computation time of our algorithm is insignificant compared to a full circuit simulation, we discuss only the storage complexity of our approach. Consider the $n$ input gate shown in Fig. 8. The various modeling options are also shown in the figure. The full model shown in Fig. 8(a) requires $n$ functions of $2 n-1$ arguments for delay. However, we have already noted the difficulties of such a model and we consider the compositional model introduced in this paper next. Although, so far we have used $D_{i z}^{(2)}$ to denote the dual-input macromodel, in practice, this actually represents a family of functions, one for each input pair. This is shown in the form of a matrix in Fig. 8(b). Here, $D_{i j z}^{(2)}$ denotes the dual input macromodel of the form (4) with $i \neq j$ and $D_{i z}^{(1)}$ denotes the single-input macromodel of the form (1). The arguments of the functions have been omitted for clarity. From this matrix, it is clear that we need $n$ single input macromodels and $n^{2}-n$ dual-input macromodels. However, our efforts in constructing the dual-input macromodels show that we need only $n$ such macromodels, one for each input being the dominant one. This is shown in Fig. 8(c). Thus, we require at most $n$ macromodels for the single-input case and $n$ macromodels for the dual-input case, making it $2 n$ macromodels for a complete delay model for the gate.

## V. EXPERIMENTAL VALIDATION

We validated our method by simulating the circuit in Fig. 1(a) for a range of input separations and transition times. The fall times of the three inputs were varied from 50 ps to 2 ns . The separation between $a$ and $b$ and between $a$ and $c$ were varied from -500 ps to 500 ps . The window size was chosen to ensure that all three inputs are influential in determining the


Fig. 9. Relative error distribution

TABLE I: COMPARISON OF MODEL WITH CIRCUIT SIMULATION

| Quantity | Delay | Rise time |
| :---: | :---: | :---: |
| Mean error | $1.46 \%$ | $-0.96 \%$ |
| Std-dev | $2.49 \%$ | $4.85 \%$ |
| Max error | $8.56 \%$ | $11.34 \%$ |
| Min error | $-6.92 \%$ | $-13.30 \%$ |

output. In order to precisely control the separations and rise times of the inputs, piecewise-linear inputs were used. The transistor sizes and the load capacitance were fixed at the values shown. We used circuit simulation results as the macromodel for processing the dual-input case. A total of 100 different input configurations were randomly generated and simulated. The relative error of our method versus simulation results are summarized in Fig. 9 and Table I. We observe that in most cases the delay computed by our technique was within $\pm 5 \%$ and the output rise time was within $\pm 10 \%$. Note that the larger error in output transition times can be tolerated since the effect of output transition time gets attenuated by the gain of the following stage [12].

## VI. CONCLUSIONS

We have shown that the temporal parameters of the inputs such as their transition times and their arrival times with respect to each other have a significant effect on the delay of a multi-input gate. We began by prescribing suitable voltage thresholds for delay measurement that always ensure a positive value of delay. We then derived a two-input proximity delay model and presented an approximate algorithm for computing delay for an $n$-input gate based on the two-input model. Our algorithm is significantly faster compared to a detailed circuit simulation while being accurate to within $10 \%$. Moreover, unlike other published methods, our method is not limited to CMOS technology.

Our future efforts will seek to provide a comprehensive delay model for complex gates, both static and precharged. We also plan to illustrate this technique using the CGaAs [1] technology.

## ACKNOWLEDGEMENTS

This research was supported in part by NSF under grant MIP-9404632 and ARPA under grant DAAH04-94-G-0327.

## REFERENCES

[1] J. K. Abrokvah et al., "A Manufacturable Complementary GaAs Process", IEEE GaAs IC Symposium, pp:127-130, 1993.
[2] D. Aurvegne, N. Azemard, D. Deschacht, and M. Robert, "Input Waveform Slope Effects in CMOS Delays," IEEE Journal of Solid State Circuits, 25(6):1588-1590, 1990.
[3] J. R. Burns, "Switching Response of Complimentary-Symmetry MOS Transistor Logic Circuits", RCA Review, 25(12):627-661.
[4] V. Chandramouli and K. A. Sakallah, "Modeling the Effects of Temporal Proximity of Input Transitions on Gate Propagation Delay and Transition Time," CSE-TR-262-95, University of Michigan, October 1995.
[5] F. C. Chang, C. F. Chen, and P. Subramaniam, "An Accurate and Efficient Gate Level Delay Calculator for MOS Circuits," in 25th ACM/ IEEE Design Automation Conference, 1988.
[6] H.-Y. Chen and S. Dutta, "A Timing Model for Static CMOS Gates," in IEEE Conference on Computer-Aided Design, 1989.
[7] S. Dutta, S. S. M. Shetty and S. L. Lusky, "A Comprehensive Delay Model for CMOS Inverters", IEEE Journal of Solid-State Circuits, 30(8):864-871, 1995.
[8] C. T. Gray, W. Liu, and R. K. Cavin III, "Exact Timing Analysis Considering Data Dependent Delays," Technical Report NCSU-VLSI-9204, North Carolina State University, December 1992.
[9] N. Hedenstierna and K. O. Jeppson, "CMOS Circuit Speed and Buffer Optimization", IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 6(2):270-281, 1987.
[10] Y.-H. Jun, K. Jun, and S.-B. Park, "An Accurate and Efficient Delay time Modeling for MOS Logic Circuits using Polynomial Approximation," IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 9(6):1027-1032, 1989.
[11] A. I. Kayssi, "A Methodology for the Construction of Accurate Timing Macromodels for Digital Circuits," Ph.D., University of Michigan, EECS Dept., 1993.
[12] A. I. Kayssi, K. A. Sakallah and T. Mudge, "The Impact of Signal Transition Time on Path Delay Computation", IEEE Transactions on Circuits and Systems-II: Analog and Digital Signal Processing, 40(5):302-309, 1993.
[13] J. T. Kong and D. Overhauser, "Methods to Improve Digital MOS Macromodel Accuracy", IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 14(7):868-881, 1995.
[14] H92A User's Manual, Meta-Software.
[15] A. Nabavi-Lishi and N. C. Rumin, "Inverter Models of CMOS Gates for Supply Current and Delay Evaluation," IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 13(10):1271-1279, 1994.
[16] T. Sakurai and A. R. Newton, "Alpha-Power Law MOSFET Model and its Applications to CMOS Inverter Delay and Other Formulas," IEEE Journal of Solid-State Circuits, 25(4):584-594, 1990.
[17] T. Sakurai and A. R. Newton, "Delay Analysis of Series Connected MOSFETs", IEEE Journal of Solid-State Circuits, 26(2):122131,1991.
[18] S.-Z. Sun, D. H. C. Du, and H.-C. Chen, "Efficient Timing Analysis for CMOS Circuits Considering Data Dependent Delays," in Proc. IEEE International Conference on Computer Design (ICCD), Cambridge, Massachusetts, 1994.
[19] TACTIC User and Reference Manual, Cascade Design Automation Corporation, 1994.
[20] Design Compiler Family Reference, V3.2b, Synopsis Inc.


[^0]:    1. $K_{n}=0.5 \mu_{n} C_{o x}\left(W_{n} / L_{n}\right)$ where $\mu_{n}$ is carrier mobility, $C_{o x}$ is oxide capacitance per unit area and $W_{n}, L_{n}$ are the transistor width and length.
