Transistor-Level Static Timing Analysis by Piecewise Quadratic Waveform Matching

Zhong Wang, Jianwen Zhu
Electrical and Computer Engineering
10 King's College Road
University of Toronto, Ontario M5S 3G4, Canada
{zwang, jzhu}@eecg.toronto.edu

Technical Report TR-02-08-01
August 2002

Electrical and Computer Engineering
10 King's College Road
Electrical and Computer Engineering
University of Toronto, Ontario M5S 3G4, Canada

{zwang, jzhu}@eecg.toronto.edu

Abstract

While fast timing analysis methods, such as asymptotic waveform evaluation (AWE), have been well established for linear circuits, the timing analysis for non-linear circuits, which are dominant in digital CMOS circuits, is usually performed by a SPICE like, time domain integration based approach, involving expensive Newton Raphson iterations at numerous time steps. In this paper, we propose a new technique that leads to the transient solution of charge/discharge paths with a complexity equivalent to only K DC operating point calculations, where K is the number of transistors along the path. This is accomplished by approximating each nodal voltage as a piecewise quadratic waveform, whose characteristics can be determined by matching the charge/discharge currents. Experiments on a wide range of circuits show that a 31.6 times speed-up over SPICE transient simulation with 10ps step size can be achieved, while maintaining an average accuracy of 99%.

CONTENTS

List of Figures

List of Tables

Abstract— **While fast timing analysis methods, such as asymptotic waveform evaluation (AWE), have been well established for linear circuits, the timing analysis for nonlinear circuits, which are dominant in digital CMOS circuits, is usually performed by a SPICE like, time domain integration based approach, involving expensive Newton Raphson iterations at numerous time steps. In this paper, we propose a new technique that leads to the transient solution of charge/discharge paths with a complexity equivalent to only *K* DC operating point calculations, where *K* is the number of transistors along the path. This is accomplished by approximating each nodal voltage as a piecewise quadratic waveform, whose characteristics can be determined by matching the charge/discharge currents. Experiments on a wide range of circuits show that a 31.6 times speed-up over SPICE transient simulation with 10ps step size can be achieved, while maintaining an average accuracy of 99%.**

I. Introduction

Timing analysis is the process of verifying the timing properties, such as propagation delay, setup/hold time violations etc., of a digital VLSI circuit. Since timing properties are inherently associated with the transient response of a circuit, circuit simulators, such as SPICE, have been the fundamental tools to obtain such characteristics. Circuit simulation involves the solution of differential equations whose size is proportional to the size of the circuit. In addition, the equations have to be solved as many times as the number of input combinations. Therefore, many techniques have been devised to reduce the exponential circuit simulation time. *Circuit partitioning* is used so that differential equation solving is confined within small circuit partitions, called *logic stages*. Typically, a logic stage is a set of channel-connected transistors and wire segments , as illustrated in Example 1 . *Gate abstraction* is used so that each logic stage corresponds to a gate, whose timing characteristics can be pre-characterized. *Static timing analysis* can be used so that only the worst case scenario of each stage needs to be simulated and only the timing of the logic stages along the longest paths needs to be considered. While these techniques offer order-of-magnitude speed-up over SPICE for full-chip timing analysis, they offer no help in speeding up the timing analysis of the individual logic stages.

Example 1: In Figure 1, the NAND gate, pass transistor *M1* and wire segment *W1* form a logic stage.

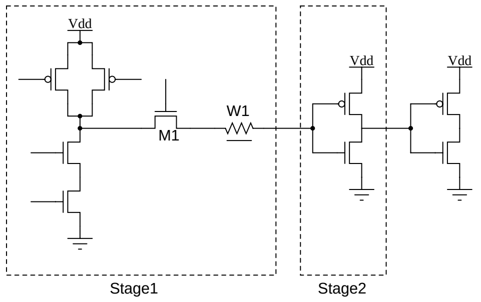

Fig. 1. Logic stage.

The simulation speed and accuracy of each logic stage, however, is essential for high-performance design. First of all, not every design cell created by designers maps naturally to a logic stage, in other words, the output of a cell is not always connected to the gate input of another cell. Therefore, the cell cannot be pre-characterized using the gate abstraction. Instead, a logic stage has to be constructed dynamically, depending on how it is connected to the rest of the circuit, as illustrated in Example 2. Second, transistors are coupled with interconnect, whose electrical properties cannot be ignored in deep submicron design. What makes interconnects particularly challenging is that their geometric shape cannot be pre-determined until routing is completed. This makes it extremely hard even for the pre-characterization of gates, since the output load can no longer be modeled as a lumped capacitor. Furthermore, many common layout structures in high-performance designs contain channel-connected transistors through long wires, e.g., a decoder tree. Therefore, fast, on-the-fly analysis of a logic stage, which boils down to the transient simulation of transistor chains, becomes an absolute necessity.

Example 2: Consider a Manchester carry chain in Figure 2. Note that the outputs of each bitsliced cell, e.g., *C1*, are channel-connected to other cells. Therefore, the cell does not correspond to a logic stage.

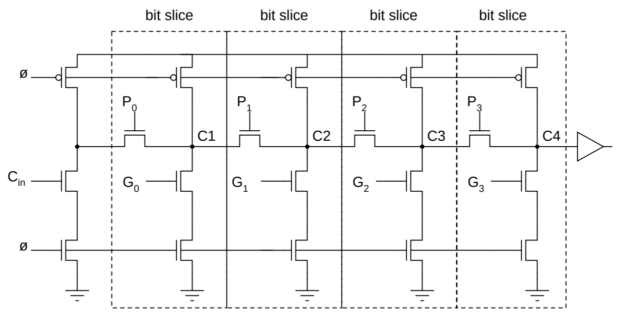

Fig. 2. Manchester carry chain.

Example 3: Consider the memory decoder tree in Figure 3, which is drawn deliberately to mimic the layout structure. Note here the lengths of the bold wire segments (*wire0, wire1, wire2*), which connect transistors' diffusions, grow exponentially with the tree level.

Two methodologies have been pursued in the past for the fast simulation of transistor chains. The first methodology exploits a simplified transistor device model, for

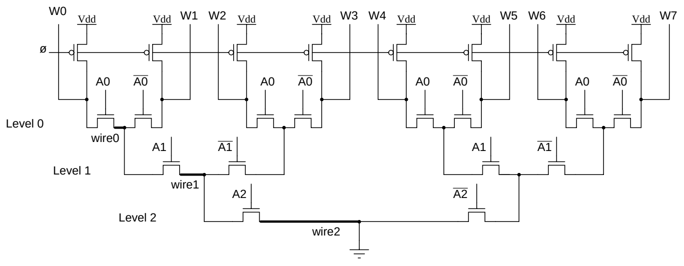

Fig. 3. Memory decoder.

example, a linear or piecewise linear model. This approach enables the modeling of non-linear circuits as linear systems. Efficient, frequency-domain analysis techniques such as asymptotic waveform evaluation (AWE), can then be used. While extremely fast, this approach introduces significant error during the device linearization process. The second methodology continues to use the time-domain numerical integration based approach, however, Newton-Raphson (NR) iteration, as the engine of the solver, is replaced by successive chord (SC) iteration, which is reportedly much faster due to the constant nature of the resultant admittance matrix.

This paper introduces a new methodology not attempted before. We achieve fast simulation speed by avoiding the brute-force solution of differential equations, while maintaining the accuracy of device models. In fact, the circuit only needs to be solved as a system of algebraic equations at K critical points, where K is the number of transistors. This approach is inherently much faster than SPICE-like simulators, since Newton-Raphson iterations only need to be performed at large time steps. To achieve this, we divide the transient process into regions separated by the K critical points. Nodal voltages in each region are then approximated by quadratic waveforms, each of which is characterized by one parameter. These parameters are determined subsequently by matching the charge/discharge currents at the critical points with those predicted by the device I/V relationship.

The rest of the paper is organized as follows. After a brief review of the previous work in Section II, we will then state problem formulation in Section III. Our proposed *piecewise quadratic waveform matching*, or QWM method, is described in detail in Section IV. Finally, we present our experimental results.

II. Related Work

Efficient extraction of timing metrics for linear circuits, typically modeled as RC or RLC networks, is well established. Elmore delay [7] has been used extensively as an improvement over the simple lumped RC metric. Since Elmore delay is inherently linked to the first moment of the circuit transfer function, a natural extension is to use higher order moments to obtain a better approximation of the transfer function by retaining more number of dominant poles. Pileggi and Rohrer pioneered this approach with their asymptotic waveform evaluation (AWE) method [13]. Alpert et al [3] showed that empirical delay metrics can be directly obtained from moments without further computation of dominant poles. Derivatives of AWE [8] [10] solve the numerical problems such as stability and passivity associated with AWE.

No transfer function can be defined for the nonlinear digital circuits. Nevertheless, one can simplify the device model in such a way so that analytical methods developed for linear circuits can be employed. The switch-level simulators, such as Crystal [12] and IRSIM [15], model the transistors as switched resistors. A logic stage can then be reduced into an RC network, for which Elmore delay is computed. The so-called fast SPICE simulators, such as MOM and ACES [5], improve this approach by the piecewise linearization of transistor model, while using AWE to further improve the evaluation accuracy of each linear region. ILLIADS [6] [16] uses a piecewise quadratic device model: while the circuit is still modeled as nonlinear, more efficient approach can be used to solve a system of quadratic differential equations.

Simplification of device models introduces errors too large to tolerate. To avoid that, TETA [4] keeps an accurate, nonlinear device model and remains to use the time-domain integration based approach to solve differential equations. However, it uses tabular device models to avoid the dominant model building time in SPICE. In addition, it replaces Newton-Raphson iteration with successive chords (SC) iteration [11]. While with a theoretically inferior convergence rate, SC can evaluate each iteration much faster because the admittance matrix of the linearized circuit stays constant. The authors later show the efficiency of TETA approach for multi-port logic stages coupled by interconnects [2] .

III. Problem Formulation

In this section, we formulate the timing analysis of logic stages as a *waveform evaluation* problem.

A. Circuit Model

A CMOS logic stage is modeled as a polar directed graph, whose vertices represent the set of circuit *nodes* and edges represent the set of *circuit elements*. The source of the graph represents the power supply and the sink represents the ground. There are three types of circuit elements: NMOS transistor, PMOS transistor and wire segment. Each circuit element is characterized by its geometric parameters, including its width, length, and optionally for the transistor, the area and perimeter of its junctions. The electrical properties of the element can be derived from these the geometric parameters. In addition, a logic stage contains a set of *inputs*, each of which is associated with the gate of a transistor, and a set of *outputs*, which are circuit nodes that are intended to be connected to the inputs of other stages.

Definition 1: A CMOS **logic stage** is a polar directed graph $\langle N, E, s, t, I, O \rangle$, with the set of nodes $N \subseteq Node$, the set of edges $E \subseteq Edge$, the source node $s \in N$, the sink node $t \in N$, the inputs $I \subseteq E$ and the outputs $O \subseteq N$.

```
Node = tuple {
   incoming    : ⟨⟩^Edge;
   outgoing    : ⟨⟩^Edge;
   }
Edge = tuple {
   kind        : Device;
   src, snk    : Node;
   w, l        : R ;
   }
Device = {nmos, pmos, wire};
```

Example 4: The logic stage in Figure 4 (a) can be reduced into the graph model in Figure 4 (b), which can be defined as $V = \{Vdd, N1, N2, N3, N5, GND\}$, $E = \{M1, M2, M3, M4, M5, W1\}$, $I = \{M1, M2, M3, M5, M5\}$, $O = \{N4\}$. $s = Vdd$, $t = GND$.

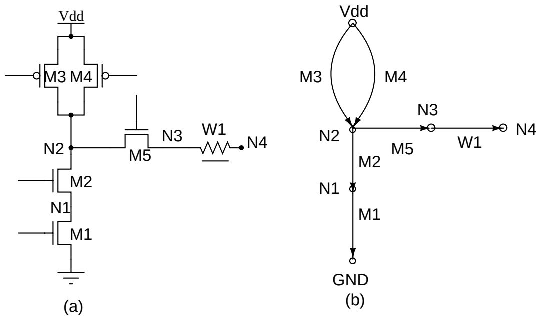

Fig. 4. Circuit model.

B. Device Model

Each circuit element is associated with a device model m. The model defines the device I/V relationship (*iv*) as a mapping from its geometric parameters and terminal voltage configuration to the corresponding current flowing from the source node to the sink node. The terminal voltage includes not only the source and sink voltages, but also the input voltage associated with the circuit edge. For NMOS and PMOS devices, the terminal voltages correspond to the gate, source, and drain voltages. The input voltage for wire segments is not defined. The device model also defines how threshold voltage and saturation voltage is related to the terminal voltages in order to factor in the body effect. In addition, the model defines the parasitic capacitance contributions to the source node and sink node. The parasitic capacitances depend not only on the device geometry, but also the terminal voltages [14]. Miller capacitances are also included.

Definition 2: A **device model** m is a member of *DeviceModel*.

```
DeviceModel = tuple {
   iv          : R × R × TermVoltage ↦ R ;
   threshold   : TermVoltage ↦ R ;
   srcCap      : R × R ↦ R ;
   snkCap      : R × R ↦ R ;
   inputCap    : R × R ↦ R ;
   }
TermVoltage = tuple {
   input       : R ;
   src         : R ;
   snk         : R ;
   }
```

Example 5: Figure III-B plots the projection of the NMOS device model: it demonstrates how current changes (I_{ds}) with respect to the change of source voltage (V_d) and sink voltage (V_s).

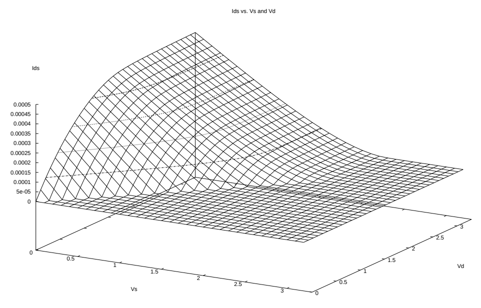

Fig. 5. I/V relationship in device model.

C. Waveform Evaluation Problem

The waveform evaluation process computes the output waveforms given the input waveforms and load capacitances , as shown in Definition 3. Waveform evaluation computes richer informational than traditional timing analysis where only the delay/slope pair is computed. The importance of waveform evaluation is confirmed by a recent paper [9] that in deep submicron circuits, the traditional delay metric can lead to up to 30% error.

Definition 3: A **waveform evaluation** of a logic stage $\langle N,E,s,t,I,O\rangle$ under the set of device models $model$: $Device \mapsto DeviceModel$ computes a set of output voltage waveforms $V : O \times T \mapsto R$, given a set of input voltage waveforms $G : I \times T \mapsto R$ and a set of load capacitances $C_L : N \mapsto R$.

Since we are performing the static timing analysis, only the worst case, in other words, charging/discharging along the longest paths, needs to be considered. Without loss of generality, we consider the discharge case of a stack of K NMOS transistors. Each transistor M^k connects circuit node $k+1$ and k, and has a size of w^k and l^k, as shown in Figure 6. The input waveform is assumed to be G^k. The capacitance of each node to ground is C^k, which equals to the sum of all capacitances contributed by the incident circuit elements and the load capacitance. To further simplify presentation, in later text we make the following assumptions. First, there is only one input switches. Second, the switching input is a step signal, we therefore can ignore the direct path current. Third, all parasitic capacitances are constant. Our implementation, as our experiment demonstrates later, does not make these assumptions.

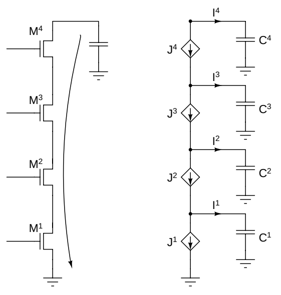

Fig. 6. Discharge along the longest path.

IV. Quadratic Waveform Matching(QWM)

A. Waveform Matching

For each circuit node k, we first assume that the corresponding voltage waveform V^k can be approximated by a waveform with an *analytical form*, for example, a polynomial with respect to time. While the analytical form is pre-defined, its characterizing parameters are to be determined. Each circuit node is associated with a capacitance to ground, with a value C^k equals to the sum of all capacitances contributed by the incident circuit elements and the load capacitance

$$C^k = m.srcCap(w^k,l^k) + m.snkCap(w^{k+1},l^{k+1}) + C_L^k \quad (1)$$

The charge/discharge current waveform can be analytically determined as well:

$$I^k = C^k \frac{dV^k}{dt}, \forall k \quad (2)$$

We then examine a particular time point τ. By examining the I/V relationship defined in the device model, the current flowing through each circuit elements J^k can be determined.

$$J_\tau^k = m.iv(w^k,l^k,G_\tau^k,V_\tau^k - V_\tau^{k-1}) \quad (3)$$

The discharge current at time τ given in Equation (2) should be matched with the difference between currents flowing through neighbor devices:

$$\begin{aligned} I_\tau^k &= J^{k+1} - J^k, \forall k < K & (4)\\ I_\tau^K &= J^K & (5) \end{aligned}$$

We therefore obtain an algebraic equation for each circuit node. If r number of parameters are chosen to characterize each output waveform, then a number of $r \cdot K$ equations need to generated, in other words, r time points need to be chosen to perform waveform matching. Given that, the transient solution of the circuit is then reduced to the solution of a system of algebraic equations!

The art part of the waveform matching methodology is the choice of the analytical waveform model. The discharging currents of all circuit nodes of a stack of 6 NMOS transistors are shown in Figure 7. An interesting and important observation is that each charge/discharge current waveform has a single peak, called *critical point*, coinciding with the time when the transistor above turns on, in other words, when the upper transistor gate drive is equal to its threshold voltage. An intuitive explanation is that for a node k, when its upper transistor M^{k+1} turns on and the channel current J^{k+1} increases, the absolute value of the discharge current I^k, which is the difference between channel currents J^k and J^{k+1}, will start to decrease.

Based on the observation, we use a linear model, $I_t^k = I_\tau^k + \alpha^k(t-\tau)$, to approximate the current waveform between two critical points $[\tau,\tau']$, i.e., the time when the lower and upper transistors turn on respectively.. Integrating Equation (2), we can obtain the *quadratic waveform approximation* of the voltage waveform characterized by a single parameter α^k:

$$V_t^k = V_\tau^k + [I_\tau^k(t-\tau) + 0.5\alpha^k(t-\tau)^2]/C^k \quad (6)$$

We can thus use the *piecewise quadratic waveform matching* strategy: divide the transient process into K regions according to the critical points; then solve for the

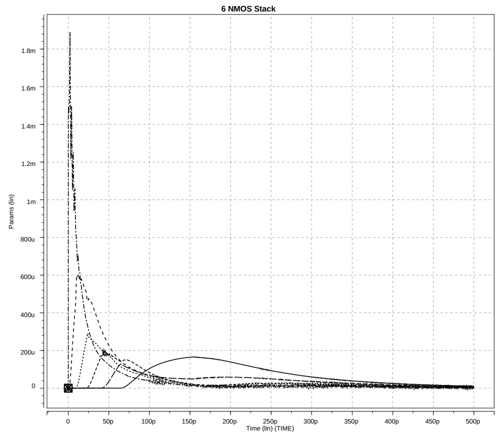

Fig. 7. Discharge current of 6 NMOS transistor stack.

parameters α^k of each region by matching currents at the corresponding critical point. More specifically, given the initial voltage value V_τ^k and current value I_τ^k, the α^k parameters are solved by the system of following algebraic equations at the next critical point τ', when the transistor M^L is turned on:

$$\left\{\begin{array}{lcl} I_{\tau'}^k & = & I_\tau^k + \alpha^k(\tau'-\tau), \forall k \\ V_{\tau'}^k & = & V_\tau^k + [I_\tau^k(\tau'-\tau) + 0.5\alpha^k(\tau'-\tau)^2]/C^k, \forall k \\ J_{\tau'}^k & = & m.iv(w^k, l^k, G_{\tau'}^k, V_{\tau'}^k, V_{\tau'}^{k-1}), \forall k \\ I_{\tau'}^k & = & J_{\tau'}^k - J_{\tau'}^{k+1}, \forall k < L \\ I_{\tau'}^L & = & J_{\tau'}^L \\ G_{\tau'}^L & = & V_{\tau'}^{L-1} + m.threshold(G_{\tau'}^L, V_{\tau'}^L, V_{\tau'}^{L-1}) \end{array}\right. \tag{7}$$

B. Numerical Method

In this study, we solve the equation using the Newton-Raphson method, which updates the guess of solution based on Equation (8) until the error $\mathbf{F}(\mathbf{x})$ or the update $\Delta\mathbf{x} = \hat{\mathbf{A}}^{-1} \cdot \mathbf{F}$ reaches a threshold value.

$$\mathbf{x_{k+1}} = \mathbf{x_k} - \hat{\mathbf{A}}(\mathbf{x_k})^{-1} \cdot \mathbf{F}(\mathbf{x_k}) \tag{8}$$

After rearranging Equation (7) to facilitate $\mathbf{F}$ evaluation and Jacobian matrix construction, $\mathbf{F}$ can be:

$$\left\{\begin{array}{l} \dfrac{I_\tau^k + J_{\tau'}^{k+1}(\mathbf{V}_{\tau'}) - \mathbf{J}_{\tau'}^{\mathbf{k}}(\mathbf{V}_{\tau'})}{2} \cdot \dfrac{T}{C^k} + V_\tau^k - V_{\tau'}^k, \quad \forall k < L \\ \dfrac{2 \cdot C^L \cdot (V_{\tau'}^L - V_\tau^L)}{-J_{\tau'}^L(\mathbf{V}_{\tau'}) + \mathbf{I}_\tau^{\mathbf{L}}} - T \end{array}\right. \tag{9}$$

Except the last column, the Jacobian matrix $\hat{\mathbf{A}} = \partial\mathbf{F}/\partial\mathbf{x}$ is close to a tridiagonal matrix.

$$\hat{\mathbf{A}} = \begin{bmatrix} \hat{\mathbf{A}}_{\mathbf{1,1}} & \hat{\mathbf{A}}_{\mathbf{1,2}} & 0 & \dots & 0 & \dots & \hat{\mathbf{A}}_{\mathbf{1,L}} \\ \hat{\mathbf{A}}_{\mathbf{2,1}} & \hat{\mathbf{A}}_{\mathbf{2,2}} & \hat{\mathbf{A}}_{\mathbf{2,3}} & \dots & 0 & \dots & \hat{\mathbf{A}}_{\mathbf{2,L}} \\ 0 & \hat{\mathbf{A}}_{\mathbf{3,2}} & \hat{\mathbf{A}}_{\mathbf{3,3}} & \hat{\mathbf{A}}_{\mathbf{3,4}} & 0 & \dots & \hat{\mathbf{A}}_{\mathbf{3,L}} \\ & & & \vdots & & & \\ 0 & & \vdots & & 0 & \hat{\mathbf{A}}_{\mathbf{L-1,L-1}} & \hat{\mathbf{A}}_{\mathbf{L,L}} \end{bmatrix}$$

Compared to the $O(N^3)$ complexity of the explicit or implicit matrix inversion, such as LU decomposition, solving a tridiagonal system can be performed in $O(N)$ time.

The last column of $\hat{\mathbf{A}}$ does not complicate the problem too much. By using Sherman-Morrison formula as in [1], $\hat{\mathbf{A}}$ can be expressed as sum of a tridiagonal matrix $\mathbf{A}$ and a matrix whose elements are product of two vectors $\mathbf{u}$ and $\mathbf{v}$.

$$\hat{\mathbf{A}} = \mathbf{A} + \mathbf{u} \otimes \mathbf{v}$$

The udpate $\Delta\mathbf{x} = -(\mathbf{A} + \mathbf{u} \otimes \mathbf{v})^{-1} \cdot \mathbf{F}$ can solved by

$$\mathbf{A} \cdot \mathbf{y} = -\mathbf{F} \qquad \mathbf{A} \cdot \mathbf{z} = \mathbf{u}$$
$$\Delta\mathbf{x} = \mathbf{y} - \frac{\mathbf{v} \cdot \mathbf{y}}{\mathbf{1} + \mathbf{v} \cdot \mathbf{z}} \cdot \mathbf{z}$$

We observe tridiagonal method gives almost twice speed-up over LU decomposition or other traditional linear system solvers.

V. Experimental Result

In this section, we document the device characterization process and experiment setup before we discuss the experimental results.

A. Device Characterization

A direct, tabular implementation of the device model can ensure no loss of accuracy as long as the grid size is fine enough. However, such approach can lead to unacceptable amount of memory usage. Therefore, we use a combination of curve-fitting and interpolation technique to compress the device model data. To characterize transistor I/V relation, we sweep V_s and V_g from 0 volt to 3.3 volt with a step size of 0.1 volt. For each V_s/V_g pair, we then generate polynomial functions to capture the dependence of channel current on drain voltage V_d using curve fitting technique. We use a linear function for the saturation region($\circ$) and a quadratic function for the triode region(+), as shown in Figure 8. Note this is different from MOM in that QWM does not require any property, such as linearity, of the transistor model. Therefore, together with the threshold voltage and saturation voltage, we store 7 parameters for each V_s/V_g pair. If an I/V query is performed with terminal voltages not captured by the grid of the table, the current value will be interpolated from neighbor points. One benefit of this characterization and fitting method is that $\partial I_{ds}/\partial V_d$ and $\partial I_{ds}/\partial V_s$, used in $\hat{\mathbf{A}}$, can be computed very fast.

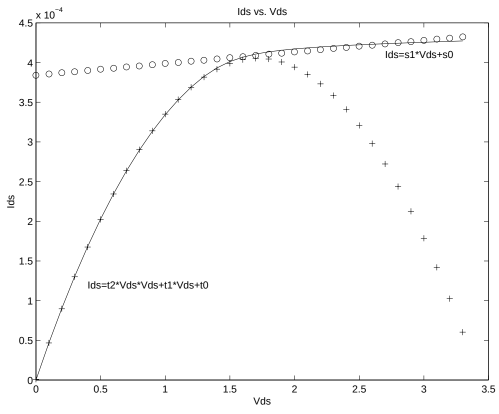

Fig. 8. I/V curve fitting.

B. Experiment Setup

To verify the QWM method, we first characterize the device models using the CMOSP35 technology with $\lambda = 0.25\mu$. The sample data used for characterization are obtained by Hspice simulation using BSIM3 V3.1 model. We then analyze a set of standard CMOS logic gates. To further measure how QWM method scales with the transistor stack length, we also analyze transistor stacks of lengths ranging from 5 to 10, with randomly chosen transistor widths. Since the simulation time of Hspice for small circuits is dominated by the model building time, which is minimal in QWM approach due to its tabular device model, we compare only the **transient time** reported by Hspice to ensure fairness. Since the user-specified step size has an impact on the Hspice simulation time, we perform Hspice simulation with step size of 1ps and 10ps. All experiments are carried on a SUN Blade 100 system running at 500 MHz.

Circuit	Hspice(1ps)		Hspice(10ps)		QWM	
	Run Time	Speed-up	Run Time	Speed-up	Run Time	Error
inv	0.06	600	0.01	100	0.0001	0.77%
nand2	0.13	217	0.02	33.3	0.0006	1.45%
nand3	0.24	240	0.04	40	0.001	1.23%
nand4	0.4	250	0.06	37.5	0.0016	0.76%

TABLE I
QWM VS HSPICE FOR LOGIC GATES.

Size		Hspice(1ps)		Hspice(10ps)		QWM	
		Run Time	Speed-up	Run Time	Speed-up	Run Time	Error
5	ckt1	0.35	184	0.05	26.3	0.0019	0.05%
	ckt2	0.49	258	0.07	36.8	0.0019	3.66%
	ckt3	0.44	232	0.07	36.8	0.0019	0.58%
6	ckt1	0.57	197	0.08	27.6	0.0029	0.61%
	ckt2	0.81	289	0.11	39.3	0.0028	1.42%
	ckt3	0.62	230	0.08	29.6	0.0027	0.12%
7	ckt1	0.99	241	0.13	31.7	0.0041	0.28%
	ckt2	0.75	214	0.1	28.6	0.0035	0.18%
	ckt3	0.9	250	0.12	33.3	0.0036	1.06%
8	ckt1	1.08	225	0.14	29.2	0.0048	0.70%
	ckt2	1.17	249	0.15	31.9	0.0047	0.65%
	ckt3	0.95	207	0.13	28.3	0.0046	0.48%
9	ckt1	1.23	232	0.16	30.2	0.0053	0.78%
	ckt2	2.22	364	0.26	42.6	0.0061	1.21%
	ckt3	2.2	324	0.26	38.2	0.0068	1.99%
10	ckt1	2.16	288	0.26	34.7	0.0075	2.15%
	ckt2	2.38	309	0.28	36.4	0.0077	0.95%
	ckt3	2.23	301	0.27	36.5	0.0074	0.78%

TABLE II
QWM VS HSPICE FOR RANDOMLY GENERATED LOGIC STAGES.

C. Results

We observe an impressive speed-up of QWM over Hspice. Table I shows part of our simulation result(in seconds) on minimum sized logic gates. For the three NAND gates, an average speed-up over 235 for 1ps step size and 37 for 10ps step size with an accuracy around 1.14% is observed. The 600 speed-up for an inverter comes from a close enough initial guess, which dramatically cuts down the number of iterations. In Table II, for each stack length, we show results for three circuit configurations, each of which has different transistor width combination. For timestep size of 1 ps, the average speed up is over 250; for timestep size of 10 ps, the number is over 30. Note that this speed-up is for transient time only. We observe much higher speed-up if total Hspice runtime is compared. In the mean time, the delay metric obtained contains a worst-case error of 3.66% error and average error of 1.00%.

The simulation result of a 6 NMOS stack, which is taken from the longest path in Manchester carry chain in Figure 2, is illustrated in Figure 9. The transient result produced by QWM is simply plotted as straight solid lines connecting the critical points calculated by QWM. The result produced by Hspice is plotted in dashed line. One can observe that QWM result follows quite closely with the Hspice result.

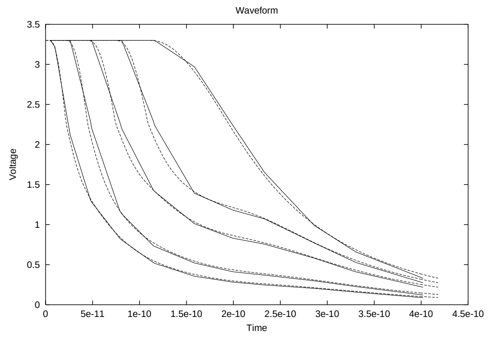

Fig. 9. A 6 NMOS stack simulation result.

Figure 10 demonstrates the simulation result for a decoder tree in Example 3 as shown in Figure 3. Note that the decoder tree has long wires between transistors. We first used AWE approach to build a macro π model for the wire. This can be evidenced by closely spaced waveform pairs in Figure 10, which correspond to the two terminals of wire segments. A speed-up of 26 over Hspice for 10 ps timestep and accuracy of 96.44% is achieved.

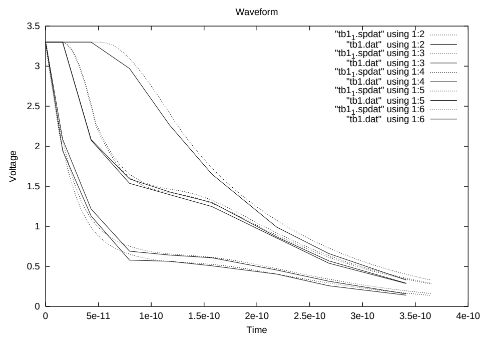

Fig. 10. Decoder tree simulation result.

VI. CONCLUSION

In this paper, we propose a new methodology, called quadratic waveform matching, for the fast timing analysis of logic stages. This approach replaces the solution of a system of differential equations by the solution of a few systems of algebraic equations. One instance of this methodology, called piecewise quadratic waveform matching, produces on average 99% accurate delay metric with order-of-magnitude speedup over SPICE.

In the future, we will study the suitability of other waveforms for the timing analysis problem. More sophisticated waveform model and critical point model may help further improve speed and accuracy.

REFERENCES

[1] *Numerical Receipes in C*. Cambridge University Press, second edition, 1992.

[2] E. Acar, F. Dartu, and L. T. Pileggi. TETA: Transistor-level waveform evaluation for timing analysis. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 21(5):605–616, May 2002.

[3] C. J. Alpert, A. Devgan, and C. Kashyap. A two moment RC delay metric for performance optimizaion. In *International Symposium on Physical Design*, pages 73–78, 2000.

[4] F. Dartu and L. Pileggi. TETA: Transistor-level engine for timing analysis. In *Proceeding of the 35th Design Automation Conference*, pages 595–598, 1998.

[5] A. Davgan and R. A. Rohrer. Adaptively controlled explicit simulation. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 13(6):746–762, June 1994.

[6] A. Dharchoudhury, S. M. Kang, K. H. Kim, and S. H. Lee. Fast and accurate timing simulation with regionwise quadratic models of MOS i-v characteristics. In *Proceeding of the 32nd Design Automation Conference*, pages 190–194, 1995.

[7] W. C. Elmore. The transient analysis of damped linear networks with particular regard to wideband amplifiers. *Journal of Applied Physics*, 19, 1948.

[8] P. Feldmann and F. W. Freund. Efficient linear circuit analysis by padè approximation via the lanczos process. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 14(5):639–649, May 1995.

[9] L. McMurchie and C. Sechen. Wta- waveform-based timing analysis for deep submicron circuits. In *Proceedings of the International Conference on Computer-Aided Design*, 2002.

[10] A. Odabasioglu, M. Celik, and L. T. Pileggi. PRIMA: Passive reduced-order interconnect macromodeling algorithm. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 18(8):645–654, August 1998.

[11] J. M. Ortega and W. R. Rheinbolt. *Iterative Solution of Non-Linear Equations in Several Variables*. New York: Acadmic, 1970.

[12] J. Ousterhout. Crystal: A timing analyzer for nMOS VLSI circuits. In *Third CalTech Conference on VLSI*, pages 57–89, 1983.

[13] L. T. Pillage and R. A. Rohrer. Asymptotic waveform evaluation for timing analysis. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 9(4):352–366, April 1990.

[14] J. M. Rabaey. *Digital Integrated Circuits, A Design Perspective*. Prentice Hall, 1996.

[15] A. Salz and M. Horowitz. IRSIM: An incremental MOS switch-level simulator. In *Proceeding of the 26th Design Automation Conference*, pages 173–178, 1989.

[16] Y.-H. Shih, Y. Leblebici, and S.-M. Kang. ILLIADS: a fast timing and reliability simulator for digital mos circuits. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 12(9):1387–1402, September 1993.