PIECEWISE LINEAR MODELS FOR SWITCH-LEVEL SIMULATION

Russell Kao

Technical Report: CSL-TR-92-532

June 1992

Computer Systems Laboratory
Departments of Electrical Engineering and Computer Science
Stanford University
Stanford, California 943054055

Abstract

Rsim is an efficient logic plus timing simulator that employs the switched resistor transistor model and RC tree analysis to simulate efficiently MOS digital circuits at the transistor level. We investigate the incorporation of piecewise linear transistor models and generalized moments matching into this simulation framework. General piecewise linear models allow more accurate MOS models to be used to simulate circuits that are hard for Rsim. Additionally, they enable the simulator to handle circuits containing bipolar transistors such as ECL and BiCMOS. Nonetheless, the switched resistor model has proved to be efficient and accurate for a large class of MOS digital circuits. Therefore, it is retained as just one particular model available for use in this framework.

The use of piecewise linear models requires the generalization of RC tree analysis. Unlike switched resistors, more general models may incorporate gain and floating capacitance. Additionally, we extend the analysis to handle non-tree topologies and feedback. Despite the increased generality, for many common MOS and ECL circuits the complexity remains linear. Thus, this timing analysis can be used to simulate, efficiently, those portions of the circuit that are well described by traditional switch level models, while simultaneously simulating, more accurately, those portions that are not.

We present preliminary results from a prototype simulator, Mom. We demonstrate its use on a number of MOS, ECL, and BiCMOS circuits.

Key Words and Phrases: AWE, moments matching, Pade approximation, switch-level simulation, circuit simulation, piecewise linear

Acknowledgements

This thesis would not have been possible were it not for the companionship, encouragement, support, and guidance from numerous friends and associates. First of all I would like to thank Neil Wilhelm and Helen Davis for believing in me, inspiring me, and encouraging me and for serving as role models. I don't think I would have attempted it without their support. I would also like to thank Forest Baskett and Sam Fuller for providing me the opportunity to attend Stanford.

My advisor, Mark Horowitz, deserves special mention for his patience and guidance throughout my stay at Stanford. As a instructor he has contributed more to my education here than any other person. As an advisor his insight into circuits and his unerring ability to ask the right (often terribly embarrassing) question have greatly increased the quality of this work.

My understanding of circuits, simulation, and the programming environment has benefited from discussions with many people around CIS; including my second reader: Bruce Wooley, as well as Bob Alverson, Don Stark, John Vlissides, Steve Tjiang, Doug Pan, Teresa Meng, Tom Chanak, and Drew Wingard. I am especially appreciative of the help from and discussions with Arturo Salz, another aficionado of switch-level simulation.

Charlie Orgish and Laura Schrager are gratefully acknowledged for the dependable computing environment at CIS. We are fortunate to have such good natured, talented, and dedicated people performing the difficult task of managing our diverse and dynamic computing resources.

I would also like to thank Richard Swan, the Western Research Laboratory, and Digital Equipment Corporation for generously provided the computing equipment and financial support for the duration of my stay at Stanford.

Finally, I am especially grateful to my new wife, Anna. Her unending love and support have added new dimensions to my life.

To Mom and Dad
and Babes

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Verification of Large Digital Designs

Over the past few decades the number of transistors it is possible to incorporate into a single digital integrated circuit (*IC*) has risen at a breathtaking pace. Unfortunately, as the complexity of integrated circuits increases, so does the likelihood of design errors and the difficulty of detecting and identifying those errors. Consequently, designers have become dependent upon ***simulation programs*** to predict the behavior of **ICs** before they are actually fabricated. These programs make it possible to verify that an IC conforms to logical and timing specifications before committing the vast resources necessary to build it.

In order to deal with the staggering complexity of designs consisting of hundreds of thousands of transistors, a hierarchy of simulation tools and techniques has evolved (Figure 1.1). In general, lower levels of simulation utilize more detailed descriptions of

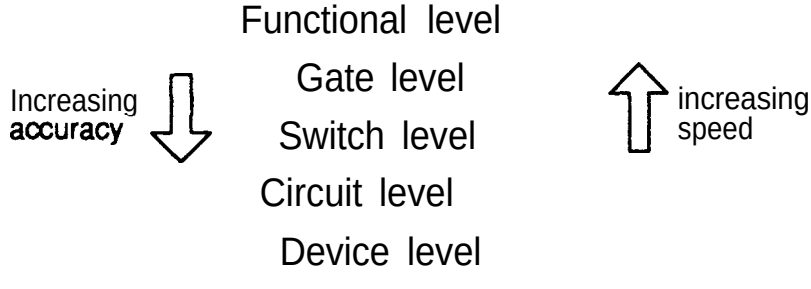


Figure 1.1: Simulation Hierarchy

the design and provide greater accuracy and flexibility. However, this increased accuracy is usually achieved at the expense of decreased efficiency, limiting the size of designs it is possible to simulate. In contrast, higher level simulators utilize simplified higher level models to represent the behavior of collections of lower level objects. Because higher level models abstract away many lower level details, higher level simulators can simulate larger designs more efficiently. However, the assumptions made in formulating the higher level models often compromise their accuracy and flexibility. Thus, tradeoffs exist for every level of simulation. The result is that the design process usually includes simulation at multiple different levels.

Because the focus of this thesis is ***switch-level*** simulation we will narrow our discussion to just the three middle levels of Figure 1.1 . Immediately below the switch-level is ***circuit*** level simulation. Circuit simulators[Nag75, WJM+73] represent the IC as a network of lumped, possibly nonlinear, transistors, resistors, inductors, capacitors, and current and voltage sources (Figure 1.2). Kirchoff's voltage and current laws are used to formulate a

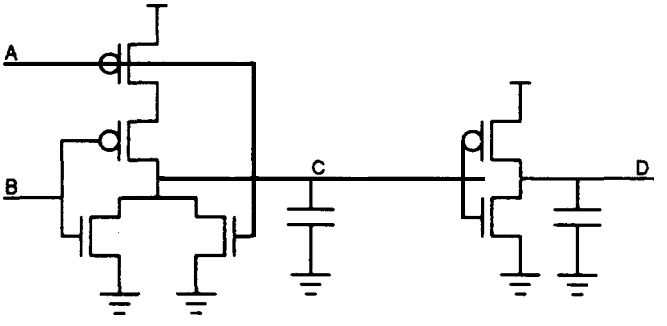


Figure 1.2: Circuit Level Representation

set of coupled nonlinear differential equations describing the behavior of the network, and numerical integration is used to solve these equations. The advantages of circuit simulators are their flexibility and accuracy. They can deal with arbitrary circuit topologies, they employ general non-linear transistor models, and they can generate detailed descriptions of the time behavior of any voltage or current in the network. Their disadvantage is that they are slow. Circuit simulation programs running on contemporary workstations can only simulate approximately 1 logic transition of a logic gate in a second. This speed is inadequate considering that large digital designs can consist of ten to a hundred thousand

logic gates.

In contrast, gate level simulators represent the IC as a network of gates (Figure 1.3). A

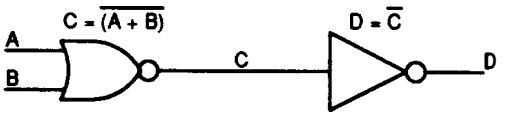


Figure 1.3: Gate Level Representation

gate is a higher level object that represents a collection of transistors. Each gate interacts with the rest of the circuit via a set of ***unidirectional*** input and output terminals. A gate observes the state of wires attached to its inputs, and sets the state of wires attached to its output. Wire states are represented by Boolean values, and a gate's behavior is modeled by a Boolean function that determines the output value as a function of the input values. The primary advantage of gate level simulation is efficiency. Logic simulation programs running on contemporary workstations can simulate up to a million logic transitions of a logic gate in a second. However, there are disadvantages. First of all, the circuit must be partitioned into a number of pre-characterized gates that exhibit unidirectional behavior. While this is readily done for gate arrays and standard cell designs (after all, these designs are composed from gates selected from libraries) custom designs often contain structures (for example, the MOS pass transistor structure) whose behavior is bidirectional and consequently is not readily modeled by the gate abstraction. Secondly, the characterization of the logical and timing behavior of gates is usually performed manually and can be time consuming and error prone.

Switch-level simulation[Bry80, Ter83, RT85b, DvGdG85, Sch85] is a relatively recent innovation which attempts to strike a balance between gate and circuit level simulation. The circuit is described as a network of transistors that are simply modeled by voltage controlled switches. Depending upon the particular approach, each switch has associated with it either a *strength*[Bry80] or a *resistance*[Ter83, RT85b] representing the current driving capabilities of the transistor (Figure 1.4). Because the circuit isn't partitioned into unidirectional gates, switch level simulators eliminate the pre-characterization step' and can simulate a wider variety of circuits than gate level simulators (including those exhibiting

'Instead of pre-characterizing every different logic gate the user only needs to pre-characterize the two different kinds of transistors: NMOS and PMOS.

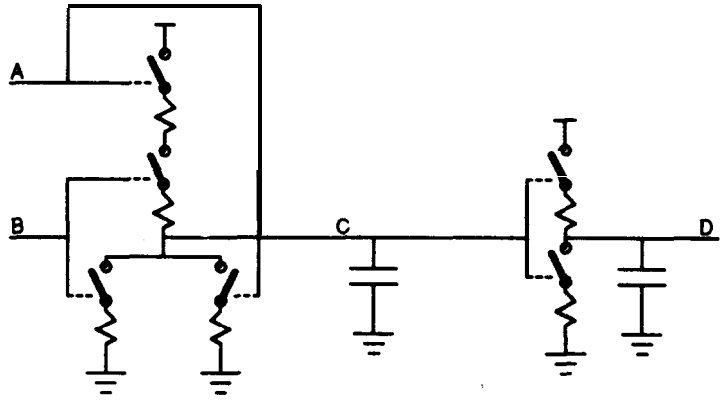


Figure 1.4: Resistive Switch Representation

bidirectional behavior). Simultaneously, the simplified circuit models allow switch-level simulators to be more than three orders of magnitude faster than circuit simulators.

Of course, limitations can arise from the use of overly simplistic transistor models. The switch model was initially developed for the simulation of MOS digital circuits, and works well for the static MOS logic which makes up most of a typical digital MOS IC. Occasionally, however, there are small portions of an IC whose behavior is not well modeled by a network of switches. Typically, these portions must be simulated at the circuit level, thus complicating the verification of the design. Furthermore, resistive or multi-strength switches are poor models for the behavior of bipolar transistors in ECL circuits. Although switch-level models have been extended to allow the simulation of bipolar transistors[HS87, SYH88, KAHS88], real ECL and BiCMOS designs occasionally include circuit techniques (for example, diode decoders, leaker resistors, current source sharing) that foil approaches based upon classical ECL current steering trees.

To address these shortcomings, this thesis attempts to extend the capabilities of switch-level simulation. Noting that the switched resistor model is just a particularpiecewise ***linear*** model, we investigate the incorporation of ***more general*** piecewise linear transistor models into the switch-level framework.[2] Several considerations motivate the use of piecewise

[2] Pillage[Pil89] suggested the incorporation of piecewise linear models and moment analysis into a circuit simulator as a promising future application of his work. Our work differs in emphasis. We neglect the general nonlinear models and circuit topologies necessary to achieve the accuracy of circuit simulation and study instead simple models and restricted topologies in an attempt to match the efficiency of switch-level simulation.

linear models. First, we would like to incorporate more sophisticated MOS models in order to simulate the behavior of MOS circuits that cannot be simulated by the use of switched resistors (for example RAM sense amplifiers). Second, we would like to be able to simulate bipolar transistors which are strongly nonlinear but which seem to be adequately described by fairly simple piecewise linear models[KAHS88]. Meanwhile, we would like to give up as little efficiency as possible. Switch-level simulation has proven itself useful for simulating the large majority of MOS digital circuits and it would be best if we could pay for additional generality for only those portions of the circuit where it was needed. To this end the simulator provides the user with a selection of transistor models of which the switched resistor model is one choice. It turns out that the RC tree analysis techniques can be generalized to efficiently handle trees of our more general piecewise linear devices with only a modest degradation of efficiency. The resulting simulator, ***Mom,*** is a mixed ***mode*** simulator that extends the capabilities of switch-level simulation in the direction of circuit simulation.

1.2 Organization

The next chapter describes previous work in estimating the transient response of digital circuits. In particular the approach taken by circuit simulators (e.g. SPICE) is compared with that taken by switch-level simulators (e.g. Rsim). Although much work has gone into trying to speed up circuit simulation, we approach the problem from a different perspective. That is, rather than starting with a simulator that is accurate and trying to improve its efficiency, we start with a simulator that is efficient and try to improve its accuracy.

The efficiency and flexibility of ***Mom*** are strongly dependent upon the choice of piecewise linear models. Chapter 3 discusses restrictions that are placed upon the models to preserve efficiency. It also explores the utility of simple piecewise linear models and demonstrates that even simple models can greatly extend the capabilities of switch-level simulation.

Networks containing piecewise linear models may not have responses that are well approximated by a single exponential. Therefore a more general ***moments matching*** procedure is used in place of Rsim's single time constant delay estimation. Chapter 4 summarizes our

experience with ***moments matching*** waveform approximation. Although the procedure has been extensively explored by others, there are a number of practical considerations unique to our application.

Chapter 5 describes extensions to RC tree analysis that allow it to handle piecewise linear models. It is demonstrated that as long as the topology of the transistors is a tree, and as long as there is no feedback, the complexity of the analysis remains **O(n).** It turns out that most MOS and ECL circuits meet these restrictions. However, the analysis can also be extended to handle non-tree topologies and feedback. Although the extensions are not as efficient they only need to be used for those portions of the circuit that don't meet the restrictions.

The introduction of piecewise linear models greatly complicates the task of detecting when devices switch. Ultimately, the problem boils down to finding ***the smallest, positive*** root of a multiple-pole exponential waveform. Chapter 6 discusses the techniques used to solve this problem efficiently.

Finally, Chapter 7 demonstrates the utility of ***Mom*** on a number of small CMOS, ECL, and BiCMOS circuits. Additionally, its efficiency is compared with that of existing switch-level simulators.

Chapter 2

Previous Work in Transient Estimation

Many different approaches have been proposed for estimating the transient response (and hence delay) of digital circuits. Here, we will review and contrast two prevalent approaches. One approach is exemplified by circuit and timing simulators. This approach is characterized by the use of nonlinear device models and ***incremental time numerical integration.*** A second approach has been taken by some MOS timing analyzers and switch-level simulators. This approach employs linear device models and ***moment analysis.***

2.1 Circuit and Timing Simulation

Circuit and timing simulation have evolved continuously over the past few decades. The "second generation"[1] simulators[WJM+73, Nag75] reached maturity during the mid 1970's. These simulators have since achieved widespread acceptance and are now regarded as the classic "circuit simulators". However, as ICs increased in size, the circuit simulators were found to require excessive amounts of computer memory and time. Consequently, a "third generation" of simulators emerged which attempted to accelerate the transient simulation of large digital ICs.

[1] Hachtel and Sangiovanni-Vincentelli[HSV81] have found it to be convenient to distinguish between three generations of simulators.

2.1.1 Circuit Simulation

Circuit simulators represent the IC as a network of lumped, possibly nonlinear transistors, resistors, inductors, capacitors, and current and voltage sources. The equations relating the terminal voltages and currents of the network elements are combined with Kirchoff's voltage and current laws to produce a set of coupled nonlinear differential equations describing the electrical behavior of the network. These equations can be written:

$$f(x(t), \mathbf{x'(f), t)} = \mathbf{0} \tag{2.1}$$

$$x \epsilon \Re^n; \quad f() : \Re^n \mathbf{x} \Re^n \mathbf{x} \Re \rightarrow \Re^n$$

where $x(t)$ is a (time varying) vector of network variables (voltages and currents), x'(t) is its time derivative, and t is time. ***The transient response*** of the network is simply the solution of these equations for $t > 0$ subject to the initial conditions: $x(t = 0) = x_0$.

Incremental time numerical integration is used to solve the equations. The procedure involves advancing time in steps:

$$t_{k+1} = \quad t_k + \quad h_k, \quad t_0 = 0 \tag{2.2}$$

(h_k is the size of the Pth step) and computing the response at each step. That is at each time step a ***linear multistep*** integration formula of the general form[2]

$$x(t_{k+1}) = \sum_{i=0}^{P} a_i x(t_{k-i}) + h_k \sum_{j=-1}^{P} b_j x'(t_{k-j}) \tag{2.3}$$

is used to eliminate x'(t) from Equation (2.1). This yields a system of coupled nonlinear ***algebraic*** equations,

$$g(x(t_{k+1})) = 0 \tag{2.4}$$

which is solved using ***Newton-Ruphson*** iteration. That is, starting from an initial guess: $x^0(t_{k+1}) = x(t_k)$, successively improved estimates are computed:

$$\text{for} \quad i = 0, 1, 2, 3 \ldots \textbf{begin}$$

$$x^{i+1}(t_{k+1}) = x^i(t_{k+1}) - \frac{g(x^i(t_{k+1}))}{g'(x^i(t_{k+1}))} \tag{2.5}$$

$$\text{end} \tag{2.6}$$

[2]The coefficients, a_i and b_i, are chosen to ensure that the formula is satisfied exactly if $x(t)$ is a low order polynomial.

(g ') is ***the Jacobian*** of ***g( )***) until some convergence criterion is met:

$$\left\| x^{i+1}(t_{k+1}) - x^{i}(t_{k+1}) \right\| < \epsilon \tag{2.7}$$

(ϵ is some error tolerance). The time step, h_k is carefully chosen to control the local truncation error of the integration formula (2.3). Time step selection involves a compromise because smaller step sizes decrease both the error and simulation efficiency.

Flexibility and accuracy are the primary advantages of circuit simulation. The equation formulation places few restrictions on the network's topology, and the ability to handle nonlinear network elements allows accurate transistor models. Furthermore, the numerical integration procedure allows the computation of the detailed time step by time step behavior of any electrical variable in the circuit. The accuracy of the integration algorithms is limited only by numerical considerations which are almost always insignificant compared to the precision with which components can be fabricated on an IC.

The disadvantage of circuit simulation is the inefficiency that results from processing the entire circuit simultaneously. At each time step circuit simulators compute multiple Newton-Raphson iterations, each of which requires the formulation and inversion of the Jacobian. However, the inversion of the Jacobian of an entire IC can be prohibitively expensive. Even using sparse techniques, the inversion of circuit matrices has been empirically observed to grow superlinearly (for example, $O(n^{1.5})$[Kun86]) with the circuit size. Furthermore, because a single time step is chosen for the entire circuit, the step size is necessarily limited by the accuracy requirements of the fastest moving subcircuit. Thus tiny time steps must be taken for the entire IC if a single gate is switching rapidly, even if nothing else in the rest of the IC is switching at all!

2.1.2 Acceleration of Circuit Simulation

To address these deficiencies a third generation of simulators emerged which attempted to accelerate the transient simulation of large digital ICs. The almost universal theme in these simulators was the ***use*** of ***decomposition*** techniques to partition the IC into smaller pieces that could be analyzed independently[DMHH87]. Partitioning achieved several things. First, it allowed the formulation and analysis of much more moderately sized systems of equations. Second, it open up the possibility of ***multirate*** simulation, that is the

selection of different time steps for different portions of the IC. Third, it became possible to bypass completely *the* analysis of *latent* subcircuits, that is subcircuits that weren't actively switching. We will describe three common techniques that were used to decompose the circuit equations (2.4): circuit tearing, relaxation, and forward Euler integration.

Circuit Tearing

Macro[RSVH79] and Slate[YHT80] employed circuit tearing techniques to reduce the Jacobian to ***bordered block diagonal*** form. Once in this form the system of equations could be solved in two steps. First, each of the blocks was solved independently. Second, the overall solution was assembled from the individual solutions. However, only the non-latent blocks needed to be processed at any particular step. If the state of a block changed little between the last two time steps or Newton Raphson iterations the block was declared ***latent*** and the solution from the previous time step or Newton Raphson iteration was simply reused. Thus, the needless reevaluation of subcircuits that were not changing was bypassed much in the same way that SPICE bypassed devices[Nag75].

Relaxation

MOTIS[CGK75] was the first of the so called "timing simulators" that utilized restricted circuit models, ***nonlinear relaxation,*** and ***time advancement integration.*** When certain restrictions were placed on the circuit (including no inductors, no floating capacitors, a grounded capacitor at each node[3], unidirectional coupling from an MOS transistor's gate to its source and drain, and appropriate ordering of the nodes) the Jacobian ***became nearly lower block tiangular*** and, for sufficiently small step sizes, ***diagonally dominant.*** These characteristics make it efficient to invert the Jacobian using a form of ***Gauss-Jacobi*** relaxation. Nonlinear Gauss-Jacobi relaxation essentially decomposes the system of equations (2.4) into a set of scalar equations by treating all non-diagonal entries of the Jacobian as if they were zero. That is, starting with an initial guess, $x^0(t_{k+1}) = x(t_k)$, each succeeding relaxation iterate, $x^{i+1}(t_{k+1})$, is assembled by solving the jth equation, g_j, for the jth component of x, x_j,

[3] A "floating" capacitor is a capacitor with neither terminal connected to ground or a power supply. A "grounded" capacitor is a capacitor with at least one terminal connected to ground or a power supply.

assuming that all other components, $\{x_l : l \neq j\}$ are fixed at their values from the previous iteration.

for i = 0, 1, 2, 3, . . . begin
  forj = 1, 2, 3, . . . n **begin**
    solve for $x_j^{i+1}(t_{k+1})$:

$$g_j(x_1^i(t_{k+1}), x_2^i(t_{k+1}), \ldots x_{j-1}^i(t_{k+1}), x_j^{i+1}(t_{k+1}), x_{j+1}^i(t_{k+1}), \ldots x_n^i(t_{k+1})) = o$$

  end
end

In principle, the innermost loop uses Newton-Raphson iteration to solve each of the scalar nonlinear equations, and the outermost loop computes successive relaxation iterations until the x^i converge (Equation ***(2.7)).*** However, the ***time advancement*** algorithms utilized only one relaxation step per time step and only one Newton-Raphson step per relaxation step because that was shown to be sufficient to guarantee convergence. MOTIS pioneered the use of time advancement integration algorithms and, in doing so, avoided both sparse Gaussian elimination and Newton-Raphson iteration.

MOTIS was followed by a number of simulators that explored variations of the relaxation procedure. ***Event-driven*** techniques from logic simulation were used by SPLICE1 [New79] to 1) dynamically order the equations thereby achieving faster convergence and 2) bypass the evaluation of latent nodes. Problems with reliability motivated the investigation of alternatives to ***Gauss-Jacobi*** time advancement, including *Gauss-Seidel*[New79], and ***Modified Symmetric Gauss-Seidel***[DMNSV83] algorithms. Additionally, it was realized that relaxation could be applied at different levels, including at the linear equation level (MOTIS2[CS84]), the nonlinear equation level (SPLICE 1.6[Sal83]) and the waveform level (Relax[LSV82]).

Forward Euler Integration

A different approach to decoupling Equation (2.4) was explored by Elogic[KKSN84] and SPECS[dG84]. When certain restrictions were placed on the network (including no inductors, no floating capacitors, and a grounded capacitance at each node) ***Nodal Analysis*** yields

the formulation:

$$Cv'(t) = i(v(t)) \tag{2.8}$$
$$v \epsilon \Re^n; \quad C \epsilon \Re^n \text{ x } \Re^n; \quad i() : \Re^n \rightarrow \Re^n$$

where v(t) is a (time varying) vector of node voltages (measured with respect to ground), v'(t) is its time &rivative, C is a diagonal matrix of node to ground capacitances, and $i()$ gives the currents injected into each node by the non-capacitive elements. Since C is diagonal, it is trivially inverted, and ***Forward Euler*** integration is used to predict the value of v at some time step, h_n, in the future:

$$v(t_{n+1}) = \frac{1}{h_n} C^{-1} i(v(t_n)) \tag{2.9}$$

Note that this formulation decouples the nodes. To predict the future voltage of a node, N, (see Figure 2.1) it is necessary to compute the currents through only those devices directly attached to N (r_2 and r_3) No matrix formulation or inversion is required. Furthermore,

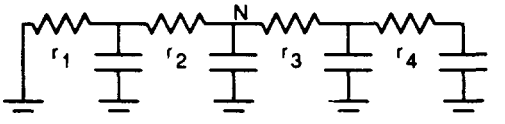


Figure 2.1: Node Decoupling

time steps can be selected independently.

The approach taken by Elogic, SPECS, and WATSWITCH[RVB88] was to partition voltage into a small number of disjoint ranges (Figure 2.2). Then the time step for a node

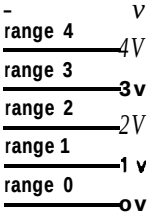


Figure 2.2: Voltage Ranges

was set equal to the amount of time it took for the node's voltage to move from its present

value to just outside its present range. An event was scheduled for a node for the time when its voltage crossed into an adjacent range. When this eventfired, the current through all devices attached to the node were updated and new waveforms were computed for all nodes connected to those devices. Thus, simulation proceeded on an event-driven basis and the evaluation of latent nodes was bypassed.

Later versions of SPECS associated the voltage ranges with the devices rather than the nodes and formalized the simulation in terms of piecewise linear voltages and piecewise constant current device models[RV87]. Furthermore, extensions were made to include floating capacitors and inductors[VFR90]. ADAPTS[SNGR91] further generalized the approach by dynamically selecting each device's voltage ranges (and, hence, the step size) based upon an analytical model for the device and the accuracy requirements of the overall simulation.

In general the third generation simulators achieved speed-ups of up to two orders of magnitude over the classic circuit simulators. However, their restricted circuit models and problems with reliability have impeded their widespread acceptance. Furthermore their speedups, although impressive, are fundamentally limited by the use of numerical integration. Time advancement numerical integration requires that time be advanced in steps whose size is limited by the need to maintain accuracy and, in some cases, stability.

2.2 Moment Based Timing Analysis and Simulation

In spite of the large amount of work invested in trying to speed up circuit simulation a large gap remained between the timing simulators and the gate level simulators. Consequently in the early 1980's a new form of simulator was devised to fill this gap, the ***switch-level*** simulators. One of these, Rsim[Ter83, Hor83], took a fundamentally different approach to transient estimation from the circuit and timing simulators. Instead of modeling the behavior of devices using ***nonlinear*** models and computing the response of the networks using ***numerical integration,*** Rsim modeled the behavior of devices using simple ***linear*** models and computed the response of networks using ***moment analysis.*** Moment analysis has the advantage over time advancement numerical integration that the response is computed once for all time rather than at numerous points in time.

Moment analysis originated in the late 1940's when **Elmore[Elm48]** utilized the first and second moments of the impulse response of a linear amplifier to estimate its step response. In general, the ***k*th** moment of the (presumed causal) impulse response, h(t), is defined:

$$\hat{m}_k = \int_0^\infty t^k h(t) dt \tag{2.10}$$

Elmore found that the quantities $(\sqrt{2\hat{m}_2 - \hat{m}_1^2})$ **and** $\hat{m}_1$ **were good estimates of the step** response's rise time and the delay to its *50%* point, respectively. The delay estimate became known as the ***Elmore delay.***

Interest in the application of moment analysis to the modeling of delays in MOS digital integrated circuits was sparked by **Penfield** and **Rubinstein[PR81]** who modeled the delay of polysilicon interconnect by the step response of ***RC trees.*** An ***RC tree*** was defined to be a tree of resistors with one grounded node and grounded capacitors at the other nodes (Figure 2.3). RC trees were particularly interesting because interconnect usually took the

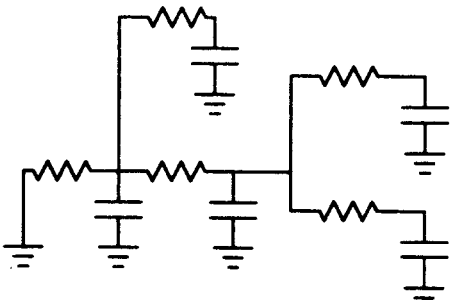


Figure 2.3: RC Tree

form of trees and because trees were easy to analyze. **Penfield** and Rubinstein described a computationally efficient algorithm for computing the first moment of RC trees and derived waveform bounds for the step response based upon the single time constant approximation.

Although this work was initially intended to model the delays of ***linear*** interconnect, it was soon used by a number of MOS timing **analyzers[Put82, Jou83]** to model the delays of networks of ***nonlinear*** MOS transistors. **Horowitz[Hor83]** more carefully justified this approach by deriving nonlinear one and two time constant waveform estimates and bounds. He then retrofitted his nonlinear timing models into an existing switch-level **simulator**[4],

[4] **Although,** this work has not been widely reported in the literature, it was performed as pm of his **PhD**

Rsim[Ter83]. Because this thesis is essentially an extension of Rsim we next describe in greater detail the algorithms used by Rsim.

As mentioned in the introduction, Rsim models transistors with the simple ***resistive switch*** consisting of the series combination of a resistor and a voltage controlled switch (Figure 2.4). The resistor models the current driving capabilities of the MOS transistor

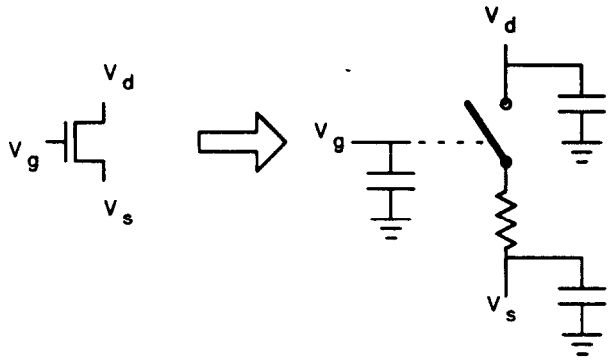


Figure 2.4: Switched Resistor Model

and is sized according to the width and length of the transistor. The switch is either on or *off* and is controlled by the gate voltage measured with respect to ground. If (for an NMOS transistor) the gate voltage is greater than half the power supply voltage then the switch is closed (current flows). Otherwise the transistor is an open circuit. It is assumed that no DC current flows into the gate and, aside from the gate's control of the switch, there is no coupling between the gate and the channel. As with the early third generation simulators, floating capacitors are disallowed and modeled by "equivalent" capacitances to ground.

Replacing on transistors by resistors and *off* transistors by open circuits usually results in a partition of the original circuit into a large number of small, mutually independent subcircuits known as ***stages*** or ***clusters*** (Figure 2.5). Because clusters are decoupled, their responses can be computed independently. Thus in order to compute the transient response of the overall circuit it is only necessary to analyze those clusters that are actively switching at each point in time. Rsim ***uses an event-driven*** simulation algorithm to schedule the evaluation of active clusters thereby avoiding ***the*** analysis of ***latent*** clusters (Figure 2.6).

thesis work on MOS timing models. This improved version of Rsim became part of the standard Berkeley CAD distribution.

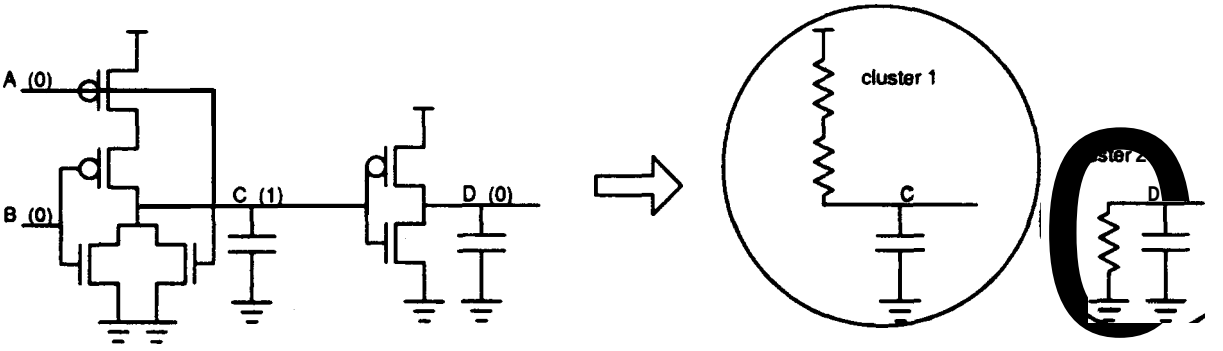


Figure 2.5: Decomposition of Circuit into Clusters.

To compute the response of a cluster Rsim uses moment analysis instead of numerical integration. Moment analysis is a two step process that involves, first, the computation of moments from the network followed by the generation of waveform estimates from those moments. Moments were computed using a procedure equivalent to finding successive DC solutions for the network. Particular advantage was taken of the tree structure possessed by most circuits. Sparse matrix formulation and sparse Gaussian elimination were bypassed in

1. Through the use of an ***event*** queue, select the next device to switch. Advance the simulation time to the time of this event.
2. Construct the ***cluster.*** That is, collect all nodes affected by this switching event.
3. Compute the response of the cluster resulting from the switching event.
4. Reschedule all MOS transistors affected by this cluster. That is, examine every transistor with a gate terminal attached to the cluster and schedule an event for it if the new response causes ir to switch some time in the future.

Figure 2.6: Rsim's Event Driven Simulation Algorithm.

favor of simpler tree analysis techniques whose complexity was guaranteed to be $\mathbf{0}(n)$ in the size of the cluster. In the rare case that non-tree topologies were encountered, heuristics' were used to simply delete resistors closing loops in order obtain an approximate solution.

[5] It was noted that the most commonly occurring case of loops was created by CMOS transmission gates. These were simply handled by parallel resistor combination.

In his thesis Horowitz showed that waveform estimates for ***linear*** RC networks, could be obtained from the moments by assuming that the system function was of the general form[6]

$$\mathcal{V}(s) = \frac{1}{(1 + \tau_1 s)} \tag{2.11}$$

for single time constant estimates, or:

$$\mathcal{V}(s) = \frac{(1 + s\tau_z)}{(1 + s\tau_1)(1 + s\tau_2)} \tag{2.12}$$

for two time constant estimates. Then, the parameters, τ_1, τ_2, and τ_z, were obtained by matching (among other things) the low order moments of the system function with those computed from the circuit. In the time domain, these one and two time constant estimates took the forms:

$$v(t) = e^{-t/\tau 1} \quad \text{and} \tag{2.13}$$

$$v(t) = \frac{(\tau_z - \tau_1)e^{-t/\tau 1} + (\tau_2 - \tau_z\, e^{-t/\tau 2})}{\tau_2 - \tau_1} \tag{2.14}$$

respectively.

However, MOS transistors are nonlinear. Horowitz showed that the single time constant estimate of the response of a nonlinear NMOS network was:

$$v(t) = \begin{cases} 1 - \tanh(t/\tau_1) & v(t) \text{ falling} \\ \frac{t}{t+\tau_1} & v(t) \text{ rising} \end{cases} \tag{2.15}$$

where τ_1 was the first moment of the network obtained by replacing each transistor by a resistor of resistance:

$$R_{\text{eff}} = \frac{2}{k(V_{dd} - V_t)} \tag{2.16}$$

(where $k = \mu C_{ox}\, W/L$ is the device transconductance parameter and L, W, μ, C_{ox}, and V_t are parameters of the quadratic MOS model[MK77, HJ83]).

However, switch-level simulators do not require the computation of the waveform but rather only the delay to the 50% point. For linear networks, the delay can be computed

[6]For simplicity Horowitz normalized the logic swing (usually 5 volts) to 1 volt.

by multiplying the first moment[7] by $-\ln(1/2)$. For NMOS networks the delay can be computed by multiplying the first moment by $\tanh^{-1}(1/2)$ for falling transitions and 1 for rising transitions. Thus although the step response of MOS-capacitor trees is not identical to the step response of resistor-capacitor trees, the computation of single time constant delay estimates of MOS trees could be made identical ***in form*** to that of resistor trees by choosing the resistor values appropriately. It was precisely this observation that justified the use of the switched resistor model. Unfortunately, Horowitz found no corresponding relationship between linear and nonlinear two time constant delay estimates.

Numerous extensions to the approach of Penfield, Rubinstein, and Horowitz have been suggested. A number of researchers have investigated the extension of linear moment analysis to circuits more general than RC trees, including RC trees with multiple sources[Chu88, RT85a], RC meshes[Wya85, LM84], and floating capacitors and controlled sources[SZ87]. Also explored was the use of higher order estimates to model the non-monotonic waveforms arising from linear[RT85a] and nonlinear[Chu88] charge sharing.

However, many of the extensions to linear moment analysis were superceded by the recent discovery[Hua90, Cha91] that single time constant delay estimation was just a special case of the more general ***moments matching*** procedure developed to solve the model order reduction problem of linear control theory. In 1956 Paynter[Pay56] applied the Padé approximation to the approximation of system functions. That approach constructs approximations of ***arbitrary*** order by matching low order moments. Pillage, Rohrer, and Huang[PR90, Hua90] combined general Padé approximation with standard circuit equation formulation and analysis techniques from circuit simulation to generate arbitrarily high order estimates of the responses of general lumped linear networks. They demonstrated the application of those techniques to the estimation of the responses of linear interconnect and the estimation of the poles and zeros of linearized models of operational amplifiers.

[7] For the linear single time constant approximation the result of matching the first moment of Equation (2.11) is $\tau_1 = \hat{m}_1$.

2.3 Improving Rsim

Switch-level simulation has its limitations. A common scenario is that the simulator can simulate 99.9% of a large circuit, although for small portions the simulator fails to produce even a correct logical result. Sometimes those failures don't interfere with the verification of the circuit. For example, the output of a voltage reference generator can be manually fixed in a switch-level simulation after being verified using SPICE. Unfortunately the simulator's failure sometimes hinders the verification of the design. For example, Rsim's inability to deal with sense amplifiers makes it difficult to check the logical correctness of RAMs.[8] However, *most* of such a circuit can be simulated at the switch-level. If it were possible to increase the generality of the simulator just for certain small portions of the circuit it would be possible to validate the entire design.

Piecewise linear models promise to give the user the ability to select different accuracies for different parts of the circuit.[9] For the RAM described above only small portions need to be simulated with more accurate models while the majority of the circuit can be simulated using switch-level models. In principle, if a simulator were designed such that the additional complexity was paid for only when it was used, it would be possible to simulate those circuits with only a moderate impact on the overall efficiency. Since the switched resistor model is a piecewise linear model, it appears promising to simply extend Rsim's simulation framework to allow more general piecewise linear models.

Although Rsim's basic simulation framework can be retained, extensive changes are required. One change involves the representation of node state. Rsim takes advantage of the fact that most digital MOS gates have logic swings from one power supply rail to the other and switching thresholds at the midpoint. Therefore Rsim represents the state of nodes using the Boolean values 0 and *1* and describes state transitions using just the delay and slope at the 50% point. However, because Mom must simulate a wider variety of circuits, it can make fewer assumptions about their properties. For example, ECL circuits have multiple nonoverlapping voltage swings which make it impossible to establish a one to one

[8] Although each of the individual pieces of a RAM is usually verified using a circuit simulator, it is still useful to verify the logical functionality of the entire RAM using a switch-level simulator in order to confirm that the decoders have been hooked up properly, that the data hasn't been inadvertently inverted, etc.

[9] The present version of the simulator depends upon the user to manually choose transistor models. Although it may be possible to automate the selection of models, this wasn't explored.

correspondence between a logical value and a voltage. To avoid building in assumptions about particular logic families, Mom utilizes real voltages to represent the state of nodes and waveforms to describe the shape of transitions.

Although the switched-resistor model is a piecewise linear model, more general piecewise linear models have attributes which necessitate extending the switch-level framework. First, while the switched-resistor model only has two regions of linearity, a more general piecewise linear model can have any number of regions. Devices with more regions are more difficult to schedule. They also generate additional events which cause logic transitions to be made up of multiple segments instead of just a single segment (Figure 2.7). Second, while the linear circuit describing the behavior of the switched-resistor model

Figure 2.7: Multiple Segments per Logical Transition.

(when it is on) is just a resistor, more general piecewise linear models can have more complex circuit models which include voltage, current, and dependent sources. More complex circuit models complicate the estimation of a circuit's response. Not only are the waveform estimates more complex, but the procedures for computing moments require more sophistication.

Unfortunately these changes degrade the efficiency of the simulator. Simpler models yield more efficient simulations, and there are strong incentives to use models that are as simple as possible. The next chapter considers the constraints that must be placed upon piecewise linear models in order to preserve efficiency. Additionally it explores the capabilities of simple piecewise linear models.

2.4 Comparison and Summary

Circuit simulation has proven to be the most general and reliable technique for estimating the transient response of digital circuits. Circuit simulation places few restrictions on the circuit

topology, utilizes general nonlinear transistor models, and employs time advancement numerical integration to solve the circuit equations. The advantages of circuit simulation are its accuracy and generality. The disadvantage of circuit simulation is its inefficiency. The general models and topologies require algorithms whose execution time grows superlinearly with the circuit size, making their use impractical for large ICs.

The classic circuit simulators were followed by a third generation of simulators which focused on the acceleration of the transient simulation of large ICs. These approaches were based upon the use of simplified circuit models and the decomposition of the circuit into pieces that could be analyzed independently. Decomposition accelerates simulation by reducing the size of the systems to be solved, by allowing the independent selection of time steps (multirate) and by bypassing sections of the circuit that are not actively switching (latency). Speedups of up to two orders of magnitude were achieved over classic circuit simulation. However, these speedups are limited because the numerical integration algorithms advance time in steps limited in size by the need to maintain accuracy.

MOS switch-level simulators such as Rsim also use decomposition techniques. However, instead of accurate nonlinear device models and numerical integration, simple linear device models and moment analysis are used to predict the response of circuits. Moment analysis has the fundamental advantage that it eliminates the need to take time steps; the response is computed once for all time. The primary advantage of switch-level simulators is their efficiency. Speedups over circuit simulation of more than three orders of magnitude have been observed. Furthermore, because they restrict the topology of networks to trees, switch-level algorithms have complexities which grow linearly ($O(n)$) with the size of the circuit, making them suitable for the simulation of large ICs. The disadvantage of switch-level simulators is their inflexibility. Simple switched resistor models are unsuitable for some MOS digital circuits, and for most ECL and BiCMOS circuits.

The conjecture explored by this thesis is that the limitations of switch level simulation can be overcome by allowing more general piecewise linear models. The incorporation of piecewise linear models requires many changes to Rsim's simulation framework and these are explored in the following chapters.

Chapter 3

Piecewise Linear Models

Mom is an extension of Rsim that allows more general piecewise linear transistor models. In principle, a simulator that utilizes piecewise linear models should be able to achieve simulations of arbitrary accuracy because piecewise linear models can be made to conform to nonlinear device characteristics with arbitrary precision by simply adding regions of linearity. However, as pointed out in the preceding chapter, efficiency concerns provide strong incentives to use models that are as simple as possible. Therefore, after a brief discussion of the representation of piecewise linear models, this chapter describes restrictions placed on the models in order to preserve the efficiency of simulation. Such restrictions do not appear to be a problem. A number of simple MOS and bipolar models are proposed and simulations are used to demonstrate their capabilities. Even models that are just slightly more complex than the switched resistor model can significantly increase the capabilities of the simulator.

3.1 Piecewise Linear Representation

We represent a piecewise linear device by a collection of linear circuits, each of which represents the linearized behavior of the device for a particular region of operation. Each region of operation is represented by a *polytope*[vB87] in the multi-dimensional space defined by the device's terminal voltages. For example, the piecewise linear description of the switched resistor model is depicted in Figures 3.1 and 3.2. The electrical behavior

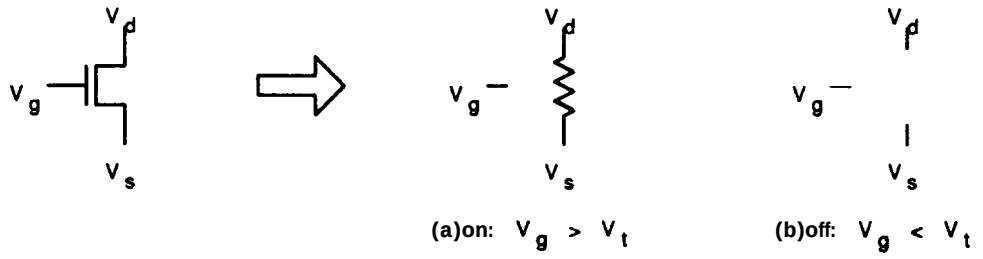


Figure 3.1: Switched Resistor Model.

of the device in each of its two regions is modeled by the circuits in Figures 3.1 (a) and (b). The regions themselves are described by polytopes in the three dimensional space defined by the source, drain, and gate voltages (Figure 3.2). The region to the right of the

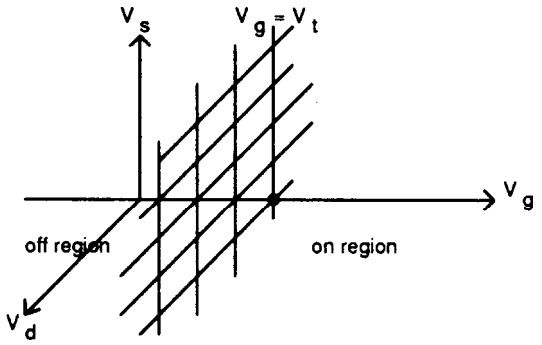


Figure 3.2: Hyperplane Subdivides Space into Regions of Operation.

cross-hatched plane labeled "$V_g = V_t$" is the polytope corresponding to the on region. The region to the left of the plane is the polytope corresponding to the *off* region.

More general models may have circuits consisting of interconnections of linear circuit elements. Additionally, they may have more than two regions of linearity. Examples of more general circuits will be given in the following sections. The remainder of this section discusses how regions of linearity are specified.

In general a device may have ***n*** terminals. Consider the n dimensional space defined by the voltages at those terminals: $\{v_1, v_2, \ldots v_n\}$. Then the set of points whose coordinates satisfy a given linear equation in those voltages:

$$a_0 + a_1 v_1 + a_2 v_2 + \ldots + a_n v_n = 0 \tag{3.1}$$

defines a ***hyperplune*** in the ***n*** dimensional space. The hyperplane is simply the multi-dimensional generalization of the familiar three dimensional plane. Like planes, **hyper-**planes partition space. Points that lie on one side of the hyperplane have coordinates that satisfy the inequality:

$$a_0 + a_1 v_1 + a_2 v_2 \; \mathrm{t} \; \ldots + a_n v_n \; \mathrm{co} \tag{3.2}$$

while points on the opposite side satisfy:

$$a_0 + a_1 v_1 \; \mathrm{t} \; a_2 v_2 \mathrm{t} \; \ldots \mathrm{t} \; a_n v_n \; > 0. \tag{3.3}$$

The pofytope is the multi-dimensional generalization of the ***polyhedron.*** While a polyhedron is a region in three dimensional space bounded by planes, a polytope is a region in n dimensional space bounded by hyperplanes. Equations (3.2) and (3.3) suggest that a polytope can be specified by a conjunction of linear inequalities:

$$\begin{aligned} a_0 + a_1 v_1 + a_2 v_2 + \ldots + a_n v_n \; &> \; 0 \\ b_0 + \; b_1 v_1 + \; b_2 v_2 + \ldots + b_n v_n \; &> \; 0 \\ c_0 + c_1 v_1 \; + c_2 v_2 + \ldots + c_n v_n \; &> \; 0 \\ &\vdots \; \vdots \; \vdots \end{aligned} \tag{3.4}$$

Each inequality bounds the region by a hyperplane.

3.2 Model Restrictions

Much of the speed of MOS switch-level simulation results from the use of transistor models that have been simplified to allow their efficient analysis. Although we intend to generalize those models, we retain certain constraints on the models in order to facilitate analysis.

Because moments matching can only be used to estimate the responses of ***linear*** circuits, the first constraint is that nonlinear capacitors must be approximated by linear (i.e. fixed value) capacitors. Although it may be possible to approximate nonlinear capacitors with piecewise linear capacitors, this was not explored.

Second, we restrict the DC coupling from the gate (base) to the source and dram (emitter and collector) to be unidirectional. This facilitates decomposition because clusters

can be analyzed independently once they have been properly ordered. The unidirectional assumption is reasonable for MOS transistors because the DC current into the gate is negligible. It is also acceptable for nonsaturating bipolar circuits such as ECL. Because a bipolar transistor's current gain, $\beta \equiv I_C/I_B$, is typically on the order of 100, the DC base current is typically two orders of magnitude smaller than the tree current and hence contributes minimally to the switching delay of the preceding gate.' There are digital bipolar circuits such as IIL and TTL which saturate the transistor and hence draw significant base currents. This restriction is not likely to be acceptable for those circuits.[2]

Lastly, we focus our attention on piecewise linear models with small numbers of regions. One of the advantages of the switched resistor model is that as long as a cluster's inputs don't change, the cluster's response can be computed once for all time. However, when transistor models acquire greater numbers of regions, transistors may pass through multiple regions during the course of a single logic transition. The response of a cluster must be recomputed whenever any of its transistors changes its region of operation. In the limit, as piecewise linear models become more detailed, the intervals between recomputation shrink until they become comparable to the time steps taken by simulators employing numerical integration. This would nullify the principle advantage of moment based techniques, that is the ability to take large time steps. Fortunately, fairly simple piecewise linear models often suffice provided that the operating point about which the device is linearized is chosen judiciously.

3.3 MOS Level-0 Model

Our simplest MOS ***Level-0*** model is the switched resistor model described above. The advantages of this model are that it can be analyzed extremely efficiently (the simulation can be very fast) and that it provides good first order estimates of the switching delay of most digital MOS circuits. Figure 3.3 compares the responses of inverters using the

'Base current does affect noise margins in ECL circuits. However, DC noise margins are more efficiently checked through the use of programs that perform a static analysis of the circuit. A dynamic logical simulation is generally unnecessary and much more expensive.

[2] In principle where base current is sufficiently important it can be modeled using a piecewise constant current source. However this was not explored.

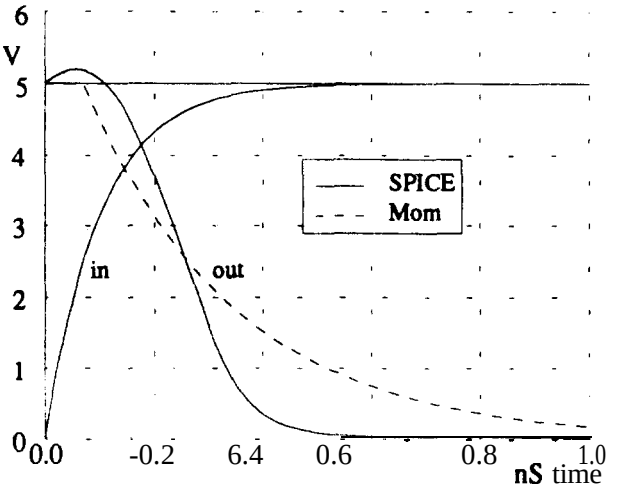


Figure 3.3: Response of Level-O Inverter Compared to SPICE.

switched resistor model and SPICE's nonlinear MOS model. For this simulation nonlinear and/or floating capacitors have been approximated by linear grounded capacitors. Also the model's resistance has been selected assuming that a gate has approximately equal input and output slopes. As expected (see Chapter 2), although the waveform shapes aren't identical, the switched resistor model does provides a good estimate of the delay to the 50% point.

The model has some limitations. Perhaps the most problematic is that it adequately models the behavior of only certain kinds of circuits. Horowitz showed that single time constant estimates can be produced for networks of nonlinear resistors as long as all the resistors possesses identical ***pseudo-linear*** I-V characteristics[Hor83]. However, this restriction excludes circuits with MOS transistors with different gate voltages, circuits with linear resistors in addition to MOS transistors, and even circuits with both NMOS and PMOS transistors in which transistors of both types are simultaneously ***on.***[3] While this assumption is rarely a problem with MOS digital gates, there are circuits, such as the sense amplifiers of dynamic and static MOS RAMs, for which the switched resistor model produces poor predictions of the DC operating points and transient behavior.

Another problem is that while the switched resistor model can be used to predict the

[3] Apparently this excludes many CMOS gates. However, in practice the delay of a CMOS gate is usually dominated by a tree of transistors of a single type driving the output node to Vcc or ground. A tree of the opposite transistor type may also be attached to the output node. However because it usually is small, it contributes little error to the switching delay even if it is 'incorrectly" modeled.

step response of MOS networks, because it theoretically switches between its two states instantaneously, it does not directly account for the affect of finite input slopes upon the delay. Instead, an additional "nonlinear gate" model must be used to account for the affects of finite input slope[Hor83]. However, for our piecewise linear simulator this input slope dependency is most conveniently handled by formally incorporating characteristics of the "nonlinear gate" model directly into a more sophisticated transistor model.

3.4 MOS Level-1 Model

3.4.1 Rationale

The MOS ***Level-1*** model was originally motivated by the observation that ***velocity saturation***, which has become prevalent in modem MOS transistors, tends to linearize the behavior of the device. Velocity saturation occurs because lattice scattering limits the maximum velocity of carriers drifting through a semiconductor. It appears in modem short channel devices because as channel lengths decrease, the electric fields in the channel increase thereby increasing the velocity of carriers[GD85, pages 105–107]. From the circuit designer's point of view, velocity saturation is undesirable because it reduces the current driving capabilities of the device. Ironically, velocity saturation simplifies the modeling of MOS transistors using piecewise linear functions.

One effect of velocity saturation is that it tends to induce saturation at drain-source voltages lower than those predicted by the quadratic model. To illustrate, Figure 3.4 plots the drain current, I_d vs the drain-source voltage, V_{ds}, for two transistors for a single gate–source voltage, $V_{gs} = 5$. The lower curve is for a MOSIS 2μ transistor while the upper curve is for the same transistor with the effects of velocity saturation eliminated.

From the plot it is apparent that the I-V characteristic of the device which isn't velocity saturated is largely quadratic. As such it consists of two segments. For low drain-source voltages the transistor operates in the ***lineur*** region and the characteristic is parabolic. Above a certain drain-source voltage[4] ($V_{ds} \sim 4V$) the transistor operates in the ***saturation***

[4]For any given gate-source voltage, V_{gs}, the quadratic model enters the saturation region when $V_{ds} > V_{gs} - V_t$

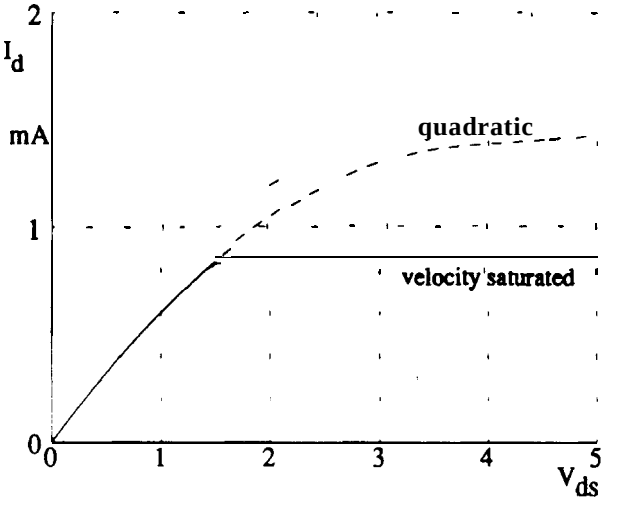


Figure 3.4: MOS I-V Characteristics With and Without Velocity Saturation

region and the characteristic is a (nearly) horizontal line.

From the plot it is also evident that the characteristic of the velocity saturated transistor is identical to that of the quadratic transistor except that it saturates at a much lower voltage ($V_{ds} \sim 1.5V$). This increases the size of the saturation region such that it comprises most of the I-V characteristic. Because this segment is nearly horizontal, the transistor in the saturation region can be approximated by a current source. Simultaneously, the linear region has been truncated so that what remains of the original parabolic segment can be approximated by a straight line. This is equivalent to modeling the transistor in the linear region by a resistor.

Another effect of velocity saturation is that it tends to linearize the dependence of the channel current upon the gate-source voltage in the saturation region. For a quadratic transistor, the transconductance in the saturation region increases linearly with the gate voltage:

$$g_m = 2\mu C_{ox}\frac{W}{L}(V_{gs} - V_t) \tag{3.5}$$

while for a velocity saturated transistor the transconductance asymptotically approaches

$$g_m = C_{ox}W\mathcal{V}_{max} \tag{3.6}$$

where $\mathcal{V}_{max}$ is the maximum velocity of carriers drifting through the semiconductor. This

effect can be observed in Figure 3.5 which plots I_d vs V_{gs} for V_{ds} = 5. While the

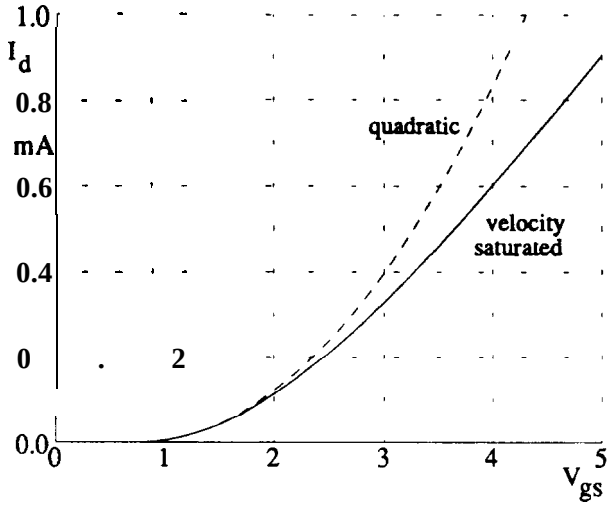
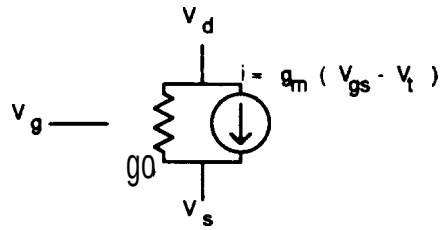


Figure 3.5: Linearization of Transconductance for Large V_{gs}

current through the quadratic transistor (the upper curve) continues to increase in slope with increasing V_{gs} (the curve is a parabola), the slope of the velocity saturated transistor (the lower curve) becomes constant for large V_{gs}. Thus the velocity saturated part of the curve can be approximated by a straight line. This is equivalent to modeling the transistor in the saturation region by a linear voltage controlled current source.

3.4.2 Model

The resulting MOS ***Level-Y*** model is depicted in Figures 3.6– 3.8. In the saturation region (Figure 3.6) the transistor is modeled by a voltage controlled current source shunted by a



$$V_{gs} - (V_t - \frac{g_o}{g_m} V_{ds}) > 0 \qquad (3.7)$$

$$g_l V_{ds} - g_m(V_{gs} - V_t) > 0 \qquad (3.8)$$

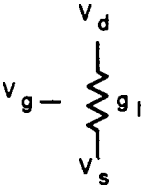
Figure 3.6: Piecewise Linear MOS Model: Saturated Region

resistor. The transconductance is set by the parameter g_m, while the gate-source voltage

for which the current source delivers zero current is V_t. In addition, the upward slope of the saturated segment of the I-V curve' is modeled by g_o.

The inequalities in the figure define the saturation region polytope. Equation (3.7) ensures that transistor has sufficient gate-source voltage to turn it on. Here, the quantity $(V_t - (g_o/g_m)V_{ds})$ is the effective threshold of the device.6 Equation (3.8) ensures that the drain-source voltage is sufficient to saturate the transistor.

In the ***linear*** region the transistor is modeled by a single conductance, g_l (Figure 3.7). Equation (3.9) ensures that current flows from the drain to the source. If that is not the case



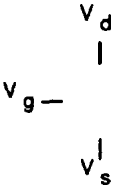
$$V_{ds} > 0 \qquad (3.9)$$

$$g_l V_{ds} - g_m(V_{gs} - V_t) < 0 \qquad (3.10)$$

Figure 3.7: Piecewise Linear MOS Model: Linear Region.

the source and drain are simply interchanged.

Lastly, the *off* region is the obvious open circuit in Figure 3.8.



$$V_{ds} > 0 \qquad (3.11)$$

$$V_{gs} - (V_t - \frac{g_o}{g_m}V_{ds}) < 0 \qquad (3.12)$$

Figure 3.8: Piecewise Linear MOS Model: Off Region.

The regions are plotted in Appendix B. Note that although there are a total of three hyperplanes, any given region is bounded by only two. This is important because the effort required to compute if and when a piecewise linear device changes regions is proportional to the number of hyperplanes bounding the current region.

[5] The variation of drain current with the drain-source voltage in the saturation region is caused by channel *lengthmodulation*[MK77]

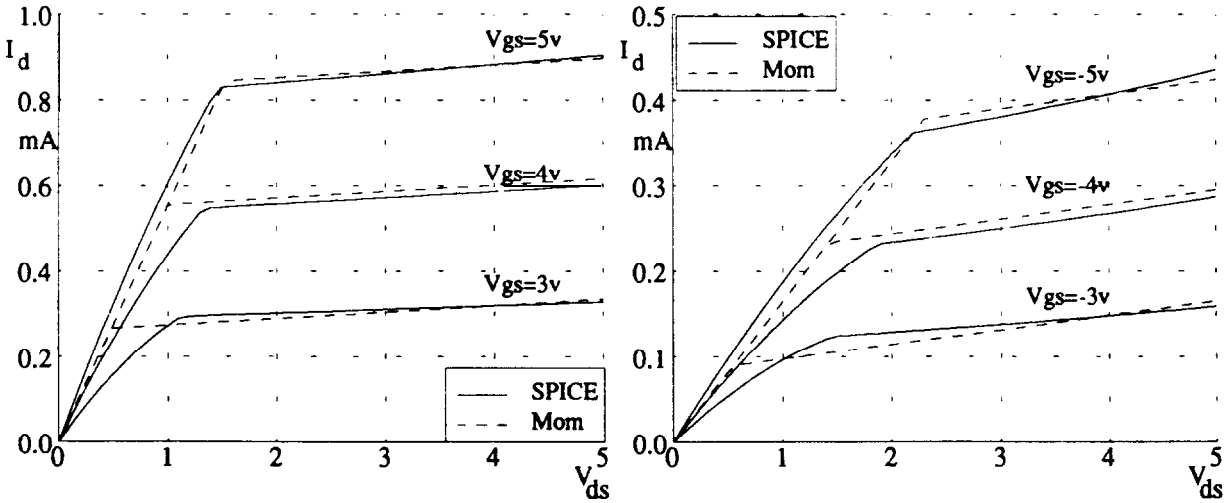
[6] The dependence of the model's threshold upon the drain voltage is explained in Appendix A.

3.4.3 Choosing Parameters

In exchange for its simplicity, the Level-l piecewise linear MOS model loses the ability to duplicate the behavior of a SPICE model over any arbitrary operating range. However, digital circuits often operate their transistors in particular regions. If the parameters of a piecewise linear transistor are chosen based upon the circuit in which it is used, good results can be obtained.

CMOS gates

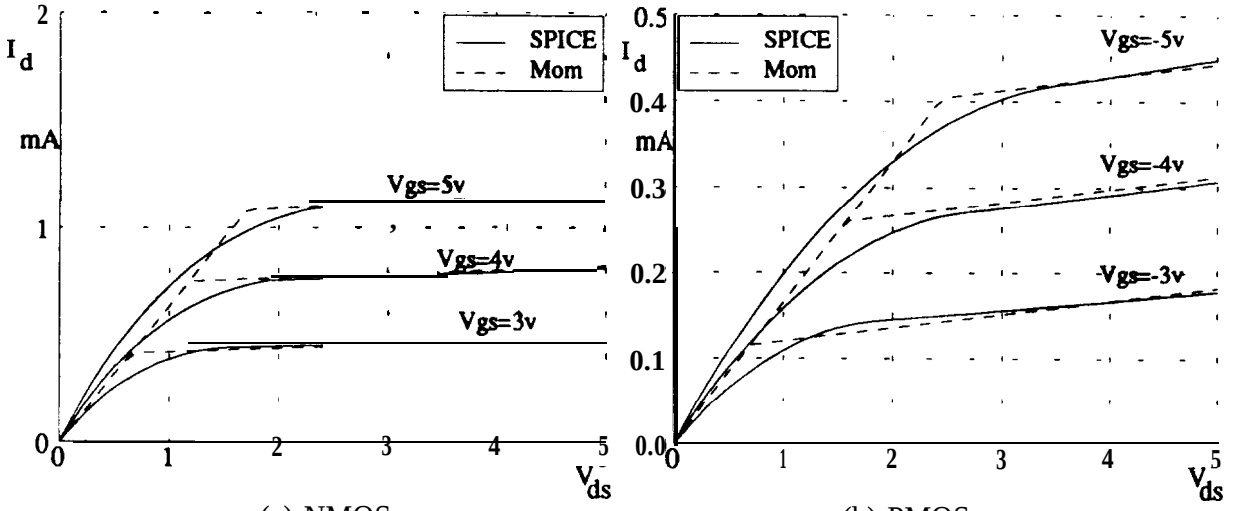
For static CMOS gates, the parameters should be chosen to match the I-V characteristics of the SPICE model in a region of greatest current, that is, $2.5 < V_{gs} < 5$ and $2.5 < V_{ds} < 5$. The rationale is that typically most of the change in voltage at the output of a CMOS gate occurs with the output transistors biased into their high current range. Because the rate of change of voltage is proportional to the current, modeling errors in regions of low current usually produce smaller timing errors than errors in regions of high current. Figures 3.9 and 3.10 illustrate the ability of the piecewise linear models to match the I-V characteristics



(a) NMOS (b) PMOS

Figure 3.9: Piecewise Linear vs SPICE I-V Characteristics: V_{gs} = 3,4, and 5 volts, SPICE Level-2 models for MOSIS 2μ Process.

of MOS transistors from MOSIS 2μ and 1.2μ processes, respectively.



(a) NMOS (b) PMOS

Figure 3.10: Piecewise Linear vs SPICE I-V Characteristics: SPICE Level-3 models for MOSIS 1.2μ Process.

In these figures, the I-V curves of the piecewise linear models are superimposed over those of the SPICE models that they were tailored to match. The figures reveal a very good match in the saturation region for large values of V_{gs} and V_{ds}.

The quality of the match is also born out by a comparison of the transient responses of CMOS inverters (Figure 3.11) using SPICE vs piecewise linear transistors.[7] Figure 3.12

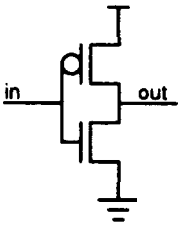


Figure 3.11: CMOS Inverter

shows the responses for fast and slow rising exponential inputs. However, the match of I-V characteristics is not as good for lower values of V_{gs}. Figure 3.13 plots I_d vs V_{gs} and shows

[7] **In this section only the modeling of transistor I-V characteristics is considered. The modeling of nonlinear and/or floating capacitors using linear grounded capacitors is consider& in a following section. Thus, in this section when circuits using SPICE vs piecewise linear transistors are compared, errors due to the modeling of capacitors are eliminated by swamping all device capacitances with large linear grounded capacitors.**

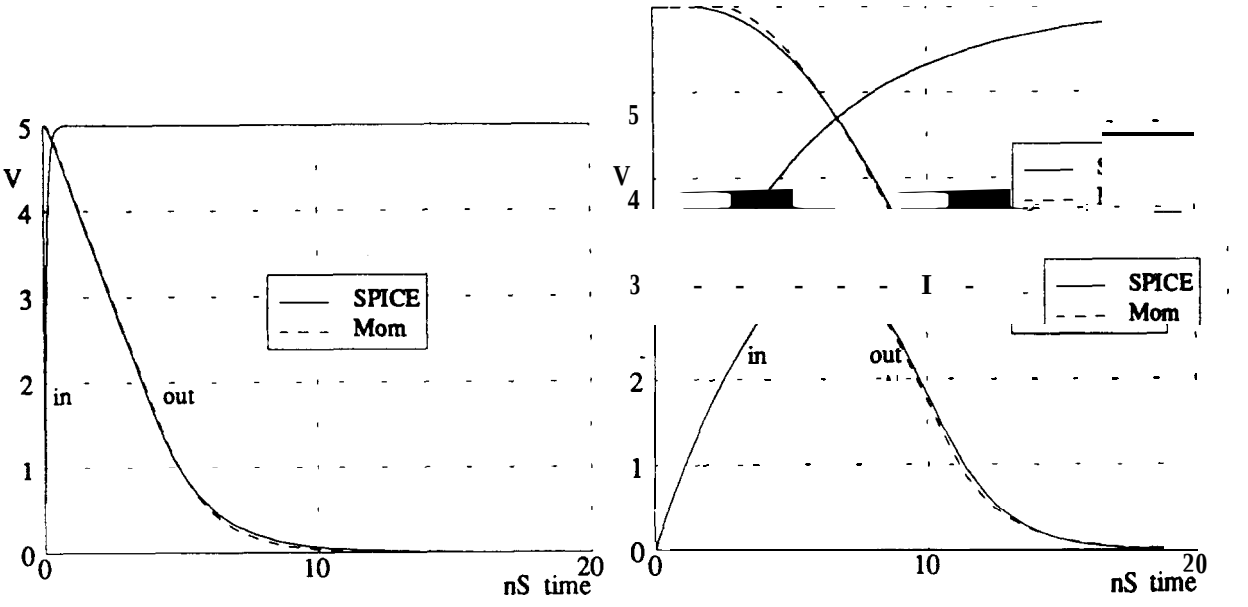


Figure 3.12: Inverters Using Piecewise Linear vs SPICE Transistors.

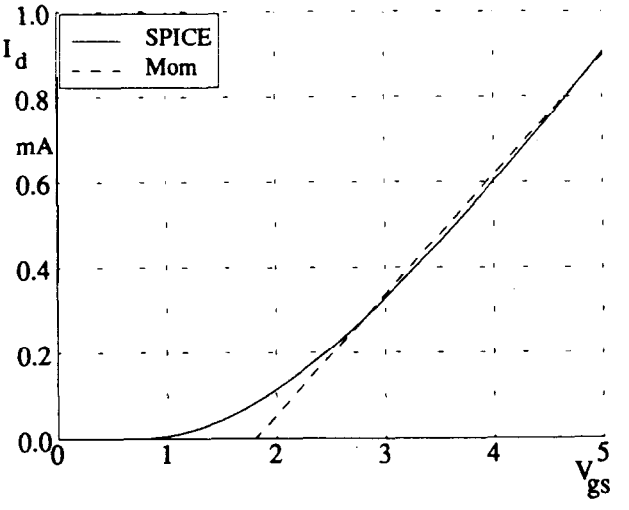


Figure 3.13: Mismatch for Small V_{gs}.

that the piecewise linear model (lower straight line) underestimates the current for small V_{gs} where the SPICE model (upper curve) is no longer velocity saturated. This modeling error is most evident for slow inputs. When the input to a logic gate switches slowly relative to the output, the transistors are never biased into their high current regions (Figure 3.14). However, as pointed out by **Horowitz[Hor83],** timing errors in this case are less important

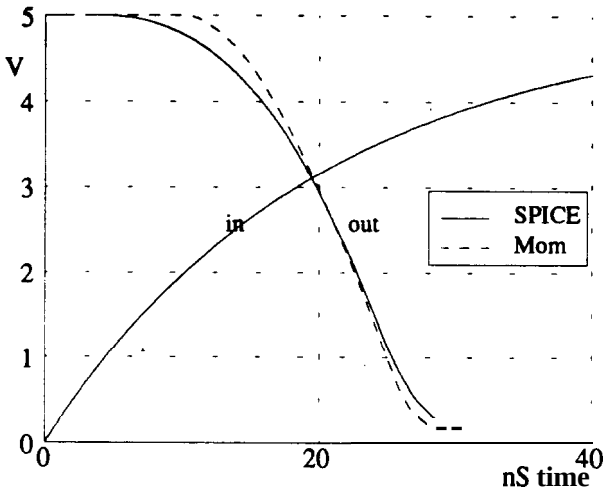
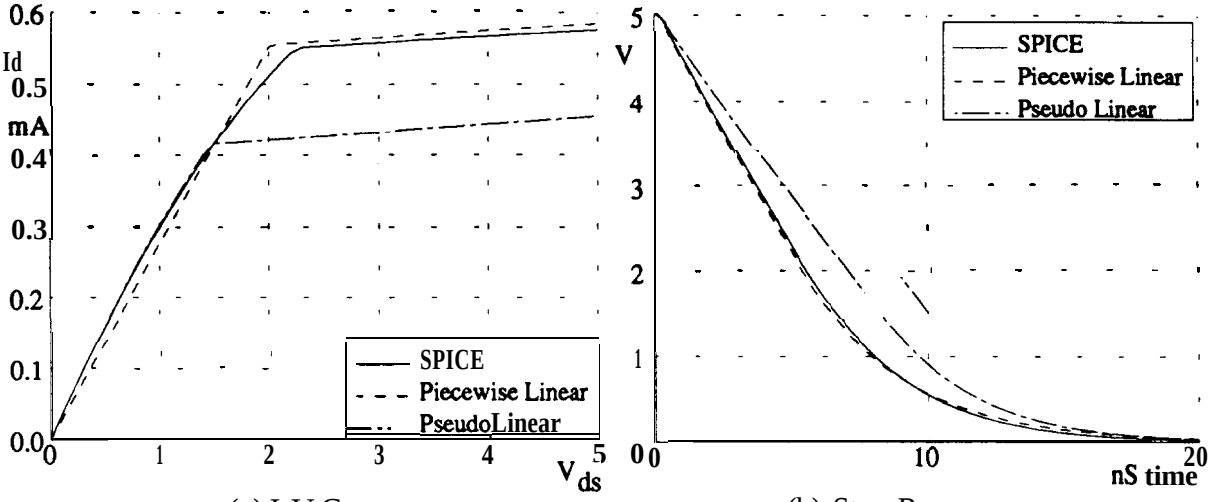


Figure 3.14: Slow Input to CMOS Gate Accentuates Modeling Errors.

because they are small relative to the delay (and hence, error) of the previous stage.

It is interesting to note that the Level-l piecewise linear model predicts the behavior of stacks of transistors more accurately than the switched resistor model. As mentioned above, it is necessary to assume quadratic behavior in order to model MOS transistors as **pseudo-**linear resistors. Unfortunately, velocity saturated transistors are not pseudo-linear. To see why, consider the series connection of a pair of identical MOS transistors. If the devices were pseudo-linear, then it would be possible to perform series combination as if they were resistors. That is the pair could be replaced by a single transistor which supplies half the current (for example, one with **half** the channel width). However, the series connection of two identical transistors is actually equivalent to a single transistor with twice the channel length. Because the channel length has been doubled, velocity saturation (a short channel effect) becomes less pronounced, and the pair will deliver more than half the current of a single transistor. This is confirmed by a comparison of the I-V characteristics and step response of series stacks of pairs of MOS transistors using SPICE, piecewise linear, and

pseudo-linear models (Figure 3.15). Both plots show that the piecewise linear model gives



(a) I-V Curves (b) Step Response

Figure 3.15: NMOS Stacks Using SPICE, Piecewise Linear, and Pseudo-Linear models.

a better fit to the SPICE model than the pseudo-linear model.

Other Circuit Forms

Of course, different circuit forms bias their transistors into different regions of operation. A piecewise linear model formulated for static CMOS gates does not necessarily perform well for other kinds of circuits. To illustrate, consider the differential amplifier circuit in Figure 3.16 which has been used to sense the small voltage swings on the bit lines of

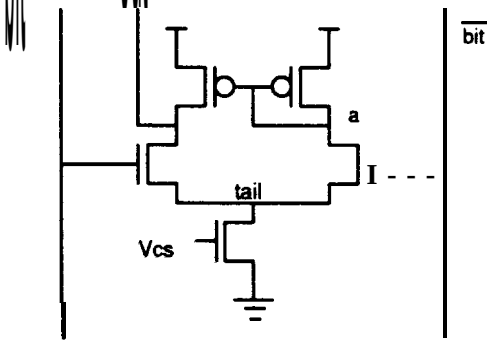
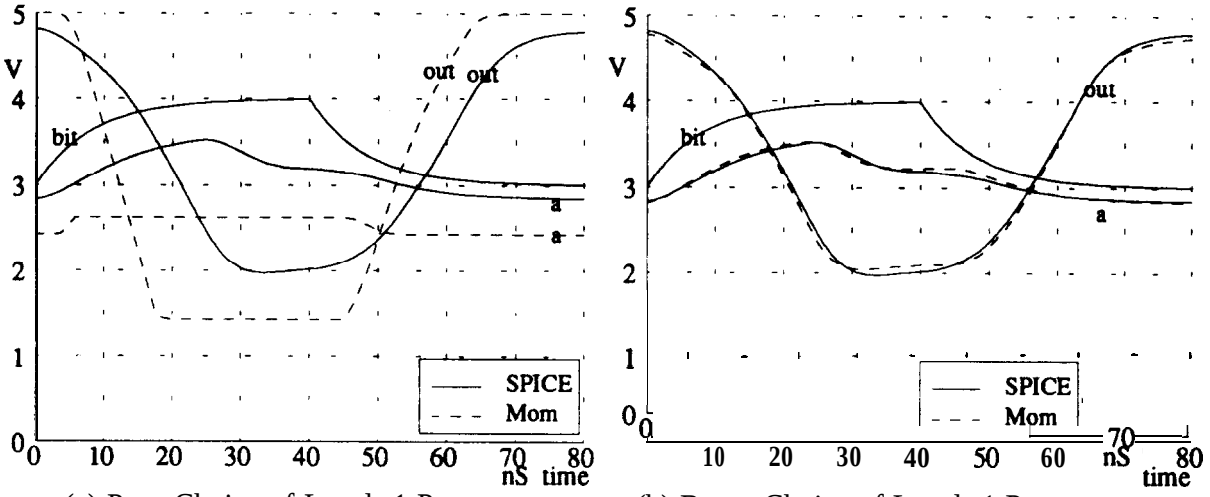


Figure 3.16: SRAM Sense Amplifier

CMOS static **RAMs[MMS+80].** When the Level-l models formulated for static CMOS logic are used in the sense amplifier (Figure 3.17 (a)) (The response of ***out*** and ***a*** are shown



(a) Poor Choice of Level- 1 Parameters. (b) Better Choice of Level- 1 Parameters.

Figure 3.17: Simulated Response of SRAM Sense Amplifier.

for rising and falling transitions on ***bit***.) the match with SPICE is not as good as with MOS static gates. However this result is a vast improvement over Rsim. Mom's output generally does the right thing although there are errors in the timing. In contrast, Rsim reports that all nodes remain in the ***undefined*** state.

It is interesting to investigate the cause of the mismatch particularly in light of the excellent matches obtained for static gates. The reason is that the sense amplifier is very different from a static gate. Not only is the output swing not from 0 to 5 volts (it is from 2 to 5 volts) but the input swing is only 1 volt peak to peak centered about 3.5 volts. Thus, the sense amplifier operates its transistors in different regions than do MOS static gates. If the user takes care to linearize the transistors in the actual operating regions of the circuit better results can be obtained. In this case excellent results are obtained when the transistors are linearized with the circuit biased at its switching threshold $(V_{bit} = V_{\overline{bit}} = 3.5V)$(**Figure** 3.17 (b)). This match is surprisingly good considering that the transistors operate in regions where they are not strongly velocity saturated and, in fact, exhibit substantial quadratic behavior.

To conclude, the MOS Level- 1 model consistently yields better results than the switched-resistor model. Furthermore, if care is taken in choosing the point about which the &vices are linearized, the matches with SPICE can be surprisingly good considering the simplicity of the model.

3.5 Bipolar Model

Bipolar transistors are more difficult to model because their exponential characteristics are more strongly nonlinear than MOS transistors' quadratic characteristics. For the purpose of ECL switch-level simulation satisfactory results have been obtained using the simple Level-0 model shown in Figure 3.18[KAHS88]. For this model, g_m represents the

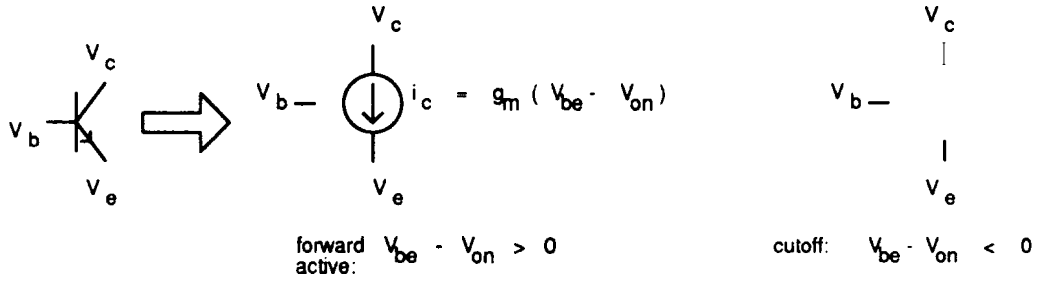


Figure 3.18: Piecewise Linear Bipolar Model.

transconductance of the transistor, and V_{on} the nominal base-emitter voltage drop in the forward active region.

The model incorporates several simplifications. In the forward active region the output conductance caused by base width modulation[MK77] was omitted. The output conductance does not appear to significantly affect the delay, and its omission allows current steering trees to be analyzed in linear time (See Section 5.5). Furthermore, the *saturation* region was omitted altogether because the additional complexity would degrade the performance of the simulation and ECL gates don't normally operate transistors in that region.

The parameters can be obtained by linearizing an ECL inverter (Figure 3.19) about its switching threshold: $V_{in} = V_{ref}$. At the threshold each of the transistors receives half of the tail current. Since the current for an ideal bipolar transistor in the forward active region

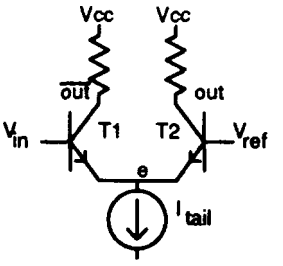


Figure 3.19: ECL Inverter

can be approximated by[MK77]:

$$I_c = I_s e^{\frac{V_{be}}{KT/q}}, \tag{3.13}$$

the parameters are given by

$$g_m = \frac{I_{tail}}{2KT/q} \tag{3.14}$$

$$v_{on} = \frac{KT}{q} \ln \frac{I_{tail}}{2I_s} \tag{3.15}$$

In addition, if the parasitic resistance in series with the emitter terminal is not modeled separately its affect can be incorporated into g,:

$$g'_m = \frac{1}{\frac{1}{g_m} + r_e} \tag{3.16}$$

Figure 3.20 depicts an ECL AND gate and compares the response of the circuit using piecewise linear and SPICE models. Again the match is quite good especially considering the simplicity of the model.

3.6 Second Order Phenomena

The previous section only considered the modeling of I-V characteristics of devices. For the sake of clarity examination of various second order effects were postponed. This section will address the modeling of nonlinear capacitance, floating capacitance, and parasitic resistance. MOS switch-level simulators model the effect of floating capacitors using "equivalent" grounded capacitors. This approximation has been effective for most MOS

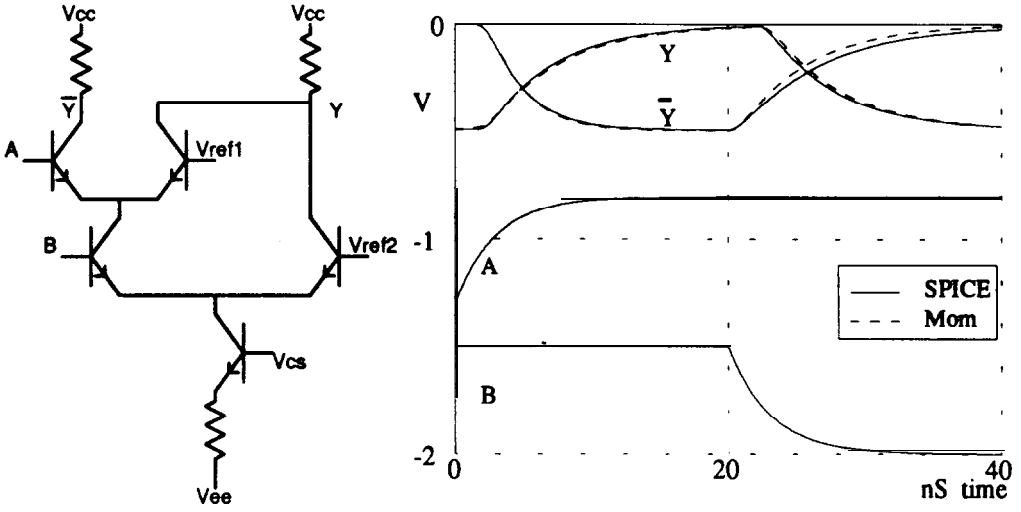


Figure 3.20: ECL AND Gate.

gates because the majority of the capacitance is to ground. However, for circuits with a larger fraction of floating capacitance this approximation can result in significant errors. To illustrate, we will consider the responses of CMOS and ECL inverters with and without floating capacitors (Figure 3.21). To simulate more realistic loading, the output of the

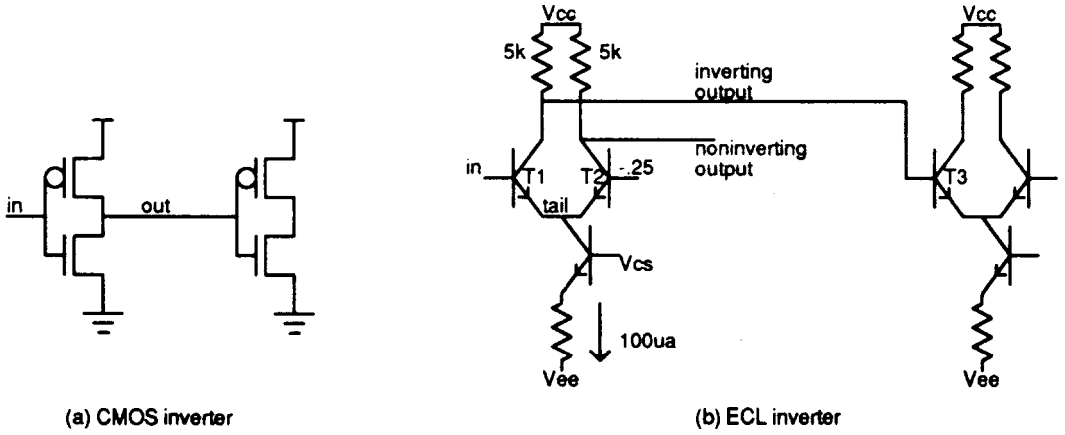
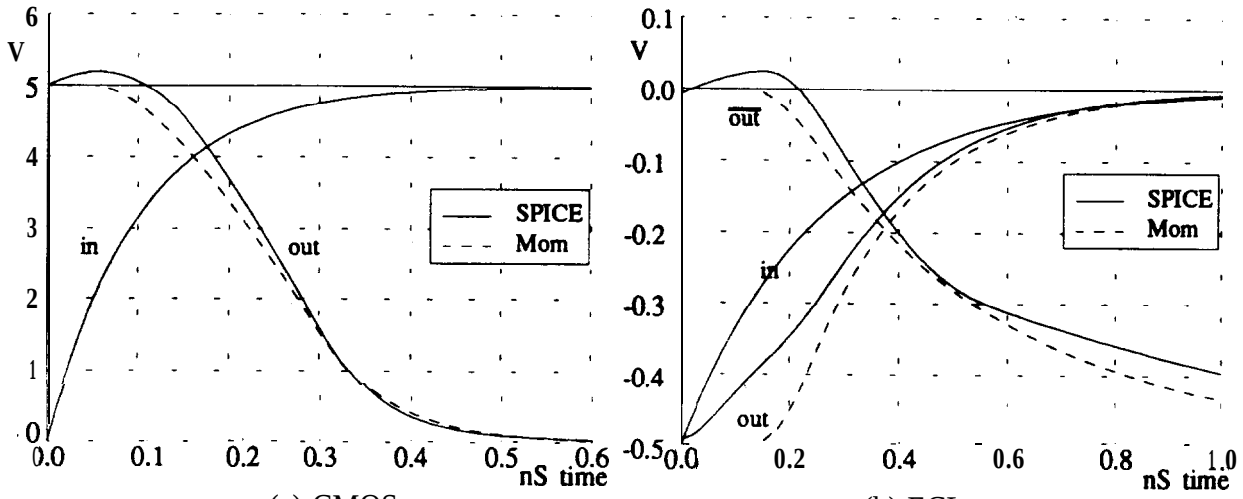


Figure 3.21: CMOS and ECL Test Circuits

CMOS inverter and the inverting output of the ECL inverter drive other inverters. The inputs are rising exponentials.

In Figure 3.22 the inverters using SPICE models include the devices' nonlinear floating capacitors, while the inverters using piecewise linear models use "equivalent" linear



(a) CMOS (b) ECL

Figure 3.22: Grounded Capacitor Approximations.

grounded capacitors. From the plots it can be seen that the grounded capacitor model performs much better for the CMOS inverter than for the ECL inverter. For the CMOS inverter, the most apparent error comes from the floating capacitance between the gate and drain of the MOS transistors. This causes the response of the SPICE inverter to bump up slightly before falling to ground. However, the responses rapidly converge after the bump. The error in switching delay is only 4%.

In contrast, numerous sources of mismatch arise when one attempts to approximate the floating capacitances of ECL gates. Several of these can be observed in the plots. The first has already been observed for the CMOS inverter. The floating base-collector capacitance of *T1* causes the inverting output of the SPICE circuit to bump up slightly before falling. Second, note that the non-inverting output of the SPICE circuit begins to rise when the input begins to rise rather than when the input crosses the switching threshold. When the input rises, current is injected into *tail* via the floating base-emitter capacitance of *T1*. This current cancels some of *the* current extracted from *tail* by the current source thus causing the non-inverting output to rise. Third, after the initial bump the inverting outputs appear to converge up until t=500ps when they again diverge. This occurs because the "equivalent" capacitance seen looking into the input of the second stage changes with time. Initially, this load consists only of the base-collector capacitance of *T3.* However, when

the second stage switches the effective capacitance increases suddenly. When the collector of ***T3*** falls, its effective floating base-collector capacitance increases because of the Miller effect. When the base-emitter voltage of ***T3*** collapses the base-emitter floating capacitance becomes visible. As ***T3*** turns off, its stored base charge must be discharged by the first stage. **The** result is that the switching delay error for the inverting output is 8.3% while the error for the noninverting output is 24%. In general, we have observed that the modeling of floating capacitors using grounded capacitors in ECL leads to errors of up to 30%.

If greater accuracy is required it is necessary to include linearized floating capacitors in the piecewise linear model. Figure 3.23 shows the ***Level-Z*** bipolar model and its

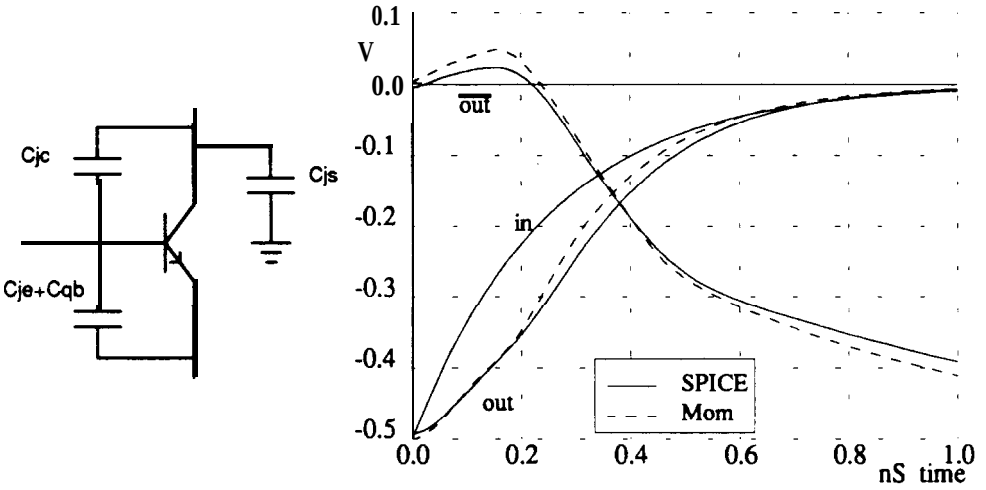


Figure 3.23: Level-l Bipolar Model has Floating Capacitors.

response compared with the SPICE model. (Appendix C describes the generation of linear approximations of the nonlinear capacitances of the bipolar transistor.) It is apparent that the waveform accuracy has increased substantially. The switching delay error of the inverting output has dropped from 8.3% to 4% and that of the non-inverting output from 24% to 19%. These results confirm that much of the error was due to the inadequacies of the grounded capacitance approximations.

Finally, if still greater accuracy is required the parasitic resistances of the bipolar transistor can be included in the piecewise linear model. Figure 3.24 shows the Level-2 bipolar model and its response compared with the SPICE model. The matching is very good indeed. The switching delay error for the inverting output has dropped from 4% to

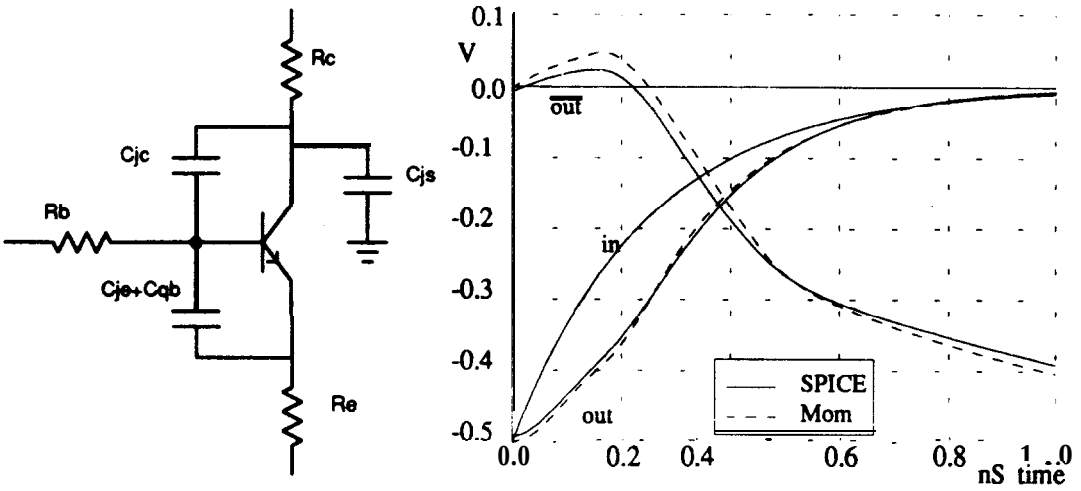


Figure 3.24: Level-2 Bipolar Model has Floating Capacitors and Parasitic Resistors

1.5%, while that for the non-inverting output has dropped from 19% to 4%. The remaining mismatch between the SPICE and piecewise linear models appears to be due mostly to the modeling of non-linear capacitors using linear capacitors. For example, the previously described bump on the inverting output caused by the base-collector floating capacitance of *T1* is consistently smaller for the SPICE models than for the piecewise linear model. The reason is that the non-linear base-collector capacitance varies by more than a factor of two from $.77C_{jc0}$ when the base is low and the collector is high, to $1.65C_{jc0}$ when the base is high and the collector is low. Thus when we are forced to choose an average capacitance, that capacitance will be too large at the beginning of the transition, and too small at the end. The initial over-estimate leads to an over-estimate of the size of the bump.

Before we conclude this section, it should be admitted that these experiments are, perhaps, overly conservative. First, note that the error in the switching delay for the non-inverting output appears to be uniformly larger than the switching &lay for the inverting output. The reason is that the non-inverting output is unloaded and hence has fewer capacitors attached. When larger numbers of capacitors are modeled, individual modeling errors tend to cancel. When this output is loaded the switching delay error decreases. This tendency for unloaded outputs to have larger modeling errors has also been observed for MOS switch-level simulators and timing verifiers. However if the output is truly unloaded then nothing is affected by it and we probably don't care about its error.

Second, only device parasitic capacitances have been included. Wire capacitance has been neglected because it is very layout dependent. However, because wire capacitance is usually modeled as linear and grounded, it tends to swamp out the errors caused by the modeling of floating and nonlinear capacitors. Thus, actual circuits, which include wire capacitance, can be modeled more accurately than might be indicated by these experiments and will be more tolerant of simpler models.

3.7 Summary

A piecewise linear device is represented by collection of linear circuits, each of which represents the device for a particular region of operation. Each region of operation is a ***polyrope*** described by a conjunction of linear inequalities. Each inequality represents a *hyperplane* boundary.

Several restrictions are placed upon the piecewise linear models in order to preserve the efficiency of timing analysis. First, the DC coupling from the gate to the source and drain for a MOS transistor, and from the base to the emitter and collector for a bipolar transistor is modeled as unidirectional. This facilitates circuit partitioning. Secondly, only linear capacitors are allowed in the models. Nonlinear capacitors must be modeled using equivalent linear capacitors. Finally, the number of piecewise linear regions is limited. This is necessary to preserve the principle advantage of moment-based techniques: the ability to take large time steps. Simple models can be constructed by utilizing a priori knowledge about the operating regions of transistors in particular forms of digital circuits. Experiments with a number of MOS and bipolar models indicates that these restrictions are acceptable. In a number of instances predictions of switching delays to within 4% were possible.

Two MOS models were explored. The simplest model, the MOS ***Level-0*** model, is the switched resistor model. This model was included to allow maximally efficient simulation. Experience with Rsim (a MOS switch level simulator) has shown that the model is useful for predicting the delays of most forms of MOS static logic. For circuits where greater accuracy and flexibility are needed, a MOS ***Level-Z*** model was explored. This model differs from the Level-O model in that it models the ***linear*** and ***saturation*** regions separately. The Level-l model was justified on the grounds that ***velocity saturation*** tends to linearize the

behavior of modem MOS transistors. Because of its increased sophistication the Level-l model supplies greater waveform accuracy when simulating static CMOS circuits. The switching error of an inverter employing Level-l models can be as low as 4% relative to SPICE. Additionally, the Level- 1 model can be used to correctly simulate circuits for which the switched resistor model fails to yield even a correct logical simulation.

Three bipolar models were also explored. The bipolar model includes only two regions of linearity, on and off. When it is on, the exponential I-V characteristic of the bipolar transistor is approximated by a voltage controlled current source. However, because of second order affects, it appears to be more difficult to model ECL than CMOS. A large part of the problem is the greater proportion of floating to groundedcapacitance in an ECL gate. For an ECL inverter it was observed that the approximation of floating capacitors by grounded capacitors produced errors as large as 30% in the switching delay. These errors were reduced to 20% when floating capacitors were included in the bipolar model. Additionally, parasitic resistances in the bipolar transistor appear to make significant contributions to the switching delay. When they were also incorporated into the transistor model, these errors were reduced to 4%. Note that all the errors reported are conservative because a large source of linear, grounded capacitance, wire capacitance, was neglected. When wire capacitance is included we expect the errors for all models to decrease.

Overall, the use of piecewise linear models is promising. Although restrictions have been placed on the models in order to preserve efficiency surprisingly simple models appear to significantly extend the accuracy and flexibility of the simulator. The next two chapters will examine the procedures for computing the response of networks made up of these devices.

Chapter 4

Waveform Approximation

The previous chapter showed that simple piecewise linear transistor models can produce good predictions of the behavior of digital circuits. However, once a circuit uses piecewise linear models it may no longer reduce to an RC tree and its response may not be well approximated by an exponential. In fact our experience has been that as transistor models increase in complexity, so do the responses of the circuits containing them. Therefore a more flexible technique for approximating waveforms is required than Rsim's single time constant delay estimation.

Fortunately, Rsim's single time constant techniques have been extended to allow more accurate waveform estimates with multiple time constants. The generalized ***moments matching*** procedure mentioned in Chapter 2 allows Mom to produce more sophisticated estimates of the responses of circuits containing piecewise linear models. This technique possesses several promising characteristics. In contrast to numerical integration algorithms, a single computation yields ***afunction of time,*** $v(t)$, representing the response for all future time, $t \in [0, \infty]$. In contrast to single time constant algorithms, it allows the computation of waveform estimates of ***arbitrary*** accuracy.

This chapter begins with a brief review of the theory behind ***the moments matching*** procedure. It then discusses some practical aspects that must be considered in an implementation. Next the utility of the procedure is demonstrated by showing a number of simulations. The chapter concludes with a description of some of the limitations of the procedure.

4.1 Waveform Approximation

One general approach to waveform approximation begins with the derivation of a low order model of the (presumably high order) system of interest. If the low order model is sufficiently simple it becomes practical to compute its response exactly. That response serves as an approximation of the response of the original system. The ***model or&r*** reduction problem has been studied extensively by linear control theorists. Of the many methods proposed, one of the simplest is based upon moments ***matching***[BL72]. Although more advanced techniques demonstrating superior convergence and stability properties were subsequently derived[Cha91], none of these appears to be efficient enough for use in our particular application.

The moments matching waveform approximation procedure involves two steps[PR90]. First the asymptotic final voltages of each node are computed by finding the DC solution of the network (assuming all piecewise linear devices remain in their present regions). This DC solution corresponds to the ***particular*** solution. Then all DC sources in the circuit are set to zero and an estimar. for the ***homogeneous*** solution is generated by matching moments. The total estimate is the sum of the two solutions.

4.2 Padé Approximation

It has been observed that moments matching is, in fact, just a particular application of the *Padé approximation*[Zak73]. In general, the Laplace transform of the impulse response of a lumped linear time-invariant circuit takes the form of a ratio of polynomials in s:

$$H(s) = \frac{\beta_0 + \beta_1 s + \beta_2 s^2 + \ldots + \beta_m s^m}{\alpha_0 + \alpha_1 s + \alpha_2 s^2 + \ldots + \alpha_n s^n}. \tag{4.1}$$

where the order of the denominator polynomial, ***n,*** is usually equal to the number of storage elements (capacitors and inductors) in the circuit. However, if ***n*** is very large it can be prohibitively expensive to compute H(s) exactly. Instead, a lower order Padé approximation is constructed:

$$\hat{H}(s) = \frac{b_0 + b_1 s + b_2 s^2 + \ldots + b_{j-1} s^{j-1}}{1 + a_1 s + a_2 s^2 + \ldots + a_j s^j} \tag{4.2}$$

($j \ll n$) and used to model the response of the system.

The parameters of the **Padé** approximation are obtained using a two step process. First the $2j$ low order terms of the series expansion in s of $H(s)$ are computed:

$$H(s) = m_0 + m_1 s + m_2 s^2 + m_3 s^3 + \ldots + m_{2j-1} s^{2j-1} + O(s^{2j}) \tag{4.3}$$

Here, the m_i are the ***moments*** of the impulse response' and $O(s^{2j})$ represents all other terms of or&r $2j$ or higher.

Moments are of interest because they are easily computed directly from the circuit even though it is usually impractical to compute H(s) in closed form. Pillage and **Rohrer[PR90]** point out that moment computation can be viewed as a sequence of DC solutions. First the voltages and currents at $t = \infty$ are found by computing the DC solution of the network assuming that capacitors are open circuits and inductors are short circuits. The difference between the initial and final voltages across capacitors and the initial and final currents through inductors are the 0th order moments. Then the $(k+1)$**st** moments are recursively computed from the kth moments by finding the DC solution of the network derived by (Figure 4.1):

1. Setting all independent sources to zero. This has the effect of subtracting out the particular solution.

2. Replacing capacitors with current sources equal to the product of the capacitance times the ***kth*** moment of the capacitor voltage.

3. Replacing inductors with voltage sources equal to the inductance times the ***kth*** moment of the inductor current.

Once the DC solution is found, the resulting capacitor voltages and inductor currents represent the $(k+1)$**st** moments. Note it is quite straight forward to find the DC solution

'Strictly speaking, the **Laplace** coefficients, m_i, are equal to the moments, $\hat{m}_i$, scaled by constant factors:

$$m_i = \frac{(-1)^i}{i!} \int_0^\infty t^i h(t) dt = \frac{(-1)^i}{i!} \hat{m}_i$$

However, in the literature the term moment is used to refer to both m_i and $\hat{m}_i$. We will continue this convenient, **although** somewhat imprecise, use of the term ***moment***.

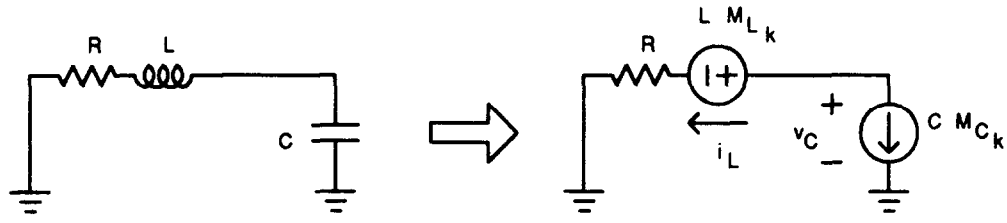


Figure 4.1: Replacing Capacitors and Inductors to Compute (k+l)st Moments.

of a resistor tree. In addition Chapter 5 will demonstrate that those procedures can be generalized to solve trees of piecewise linear transistors.

Once the low $2j$ moments have been computed from the circuit, $\hat{\mathbf{H}}(s)$ is expanded via long division into a power series in s and its coefficients are chosen such that the low $2j$ moments of the approximate response match those of the actual response. It has been shown that this matching constrains the a; and b_i to be linearly dependent on the m_i. The a; and b_i can be obtained by solving the following linear system of equations[Zak73, Hua90]:

$$\begin{vmatrix} m_{j-1} & m_{j-2} & \cdots & m_0 \\ m_j & m_{j-1} & \cdots & m_1 \\ & & & \\ m_{2j-2} & m_{2j-1} & \cdots & m_{j-1} \end{vmatrix} \begin{vmatrix} a_1 \\ a_2 \\ \\ a_j \end{vmatrix} = - \begin{vmatrix} m_j \\ m_{j+1} \\ \\ m_{2j-1} \end{vmatrix} \tag{4.4}$$

$$\begin{bmatrix} m_0 & & & & \\ m_1 & m_0 & & & \\ m_2 & m_1 & m_0 & & \\ & \cdot & \cdot & \cdot & \\ m_{j-1} & m_{j-2} & m_{j-3} & \cdots & m_0 \end{bmatrix} \begin{bmatrix} 1 \\ a_1 \\ a_2 \\ \vdots \\ a_{j-1} \end{bmatrix} = \begin{bmatrix} b_0 \\ b_1 \\ \cdot \\ \\ b_{j-1} \end{bmatrix} \tag{4.5}$$

The matrix on the left hand side of Equation 4.4 is referred to as the ***moment matrix***. Once $\hat{\mathbf{H}}(s)$ is known, its denominator can be factored

$$\hat{\mathbf{H}}(s) = \frac{b_0 + b_1 s + b_2 s^2 + \ldots + b_{j-1} s^{j-1}}{(s\tau_1 + 1)(s\tau_2 + 1)\ldots(s\tau_j + 1)} \tag{4.6}$$

and the result expanded into partial fractions:

$$\hat{\mathbf{H}}(s) = \frac{k_1}{(s\tau_1 + 1)} + \frac{k_2}{(s\tau_2 + 1)} + \cdots + \frac{k_j}{(s\tau_j + 1)} \tag{4.7}$$

But the inverse transform for each term in Equation (4.7) is given by:

$$\mathcal{L}^{-1}\left\{\frac{1}{1+s\tau}\right\} = \frac{1}{\tau}e^{-t/\tau}. \tag{4.8}$$

Thus the inverse transform of Equation (4.7) yields the time domain estimate[2]

$$h(t) = \frac{k_1}{\tau_1}e^{-t/\tau_1} + \frac{k_2}{\tau_2}e^{-t/\tau_2} + \cdots + \frac{k_j}{\tau_j}e^{-t/\tau_j} \tag{4.9}$$

In principle, approximations of arbitrary order can be computed from the moments. It has been shown that when $j = 1$ the Padé approximation is equivalent to the single time constant estimate generated by RC tree analysis[PR90]. Furthermore, as j approaches the order of the actual system being approximated, the Padé approximation, H(s), converges to the actual system function, $H(s)$[Hua90].

4.3 Practical Considerations

Although in principle moments matching encompasses approximations of arbitrary order, in practice it may be impossible to produce an approximation if the order is too high or too low. Furthermore even in those cases when an approximation is possible, it may contain spurious unstable poles. Moments matching has been observed to be numerically sensitive[Hua90] and care must be taken when applying the procedure.

4.3.1 Order Too Low

If the order of the approximation is too low the moments matching procedure can fail to yield any approximation at all. Note that the first row of a jth order moment matrix (Equation (4.4)) is $[m_{j-1} \ldots m_2\, m_1\, m_0]$. If the low j moments are all zero, then that row is all zero, the moment matrix is singular, and it is impossible to produce an approximation. This situation frequently arises as a side effect of floating capacitors. Consider the the circuit in Figure 4.2. Note that except for *n1* all nodes are initially zero and that all final values are zero. The lowest moment, m_0, is given by the negative of the initial condition[PR90].

[2] In practice, we solve for the a_i, factor the polynomial to get the pole frequencies, and then compute the pole coefficients, k_i, directly from the poles frequencies. Se-e [PR90].

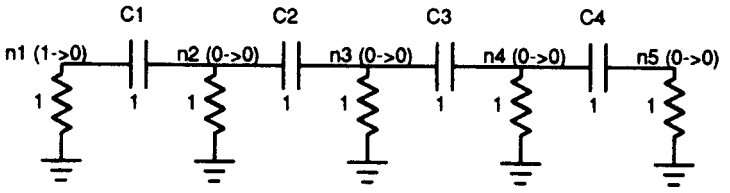


Figure 4.2: Circuit with Low Order Moments Equal to Zero.

Thus the zeroth order moments for nodes $\{n1, n2, n3, n4\}$ are $\{-1, 0, 0, 0\}$, respectively. To compute successive moments each capacitor is replaced by a current source equal to the capacitance times the difference of the moments on its terminals:

$$\begin{aligned} m_0 &: \{ -1, 0, 0, 0 \} \\ m_1 &: \{ 1, -1, 0, 0 \} \\ m_2 &: \{ -2, 3, -1, 0 \} \end{aligned}$$

Thus nodes that are initially at rest but are coupled via a chain of floating capacitors to nodes that are not will have low order moments equal to zero. If the chain is k links long then the k low order moments will be zero and approximations of order $\leq k$ will be impossible. This is undesirable because the lowest order approximation possible grows with the size of the circuit.

However, as will be observed in Chapter 5 the coupling through multiple levels of floating capacitors is not important for digital circuits. Therefore when too many low order moments are zero we simply assume that this was caused by too many levels of floating capacitors and set the particular response to zero.

4.3.2 Order Too High

Although approximations of arbitrarily high order are theoretically possible, in practice numerical considerations limit the order of the approximation. The reason for this is that round-off errors can make it difficult to compute higher order moments with sufficient accuracy. Note that the series expansion of the Laplace transform of a single pole response is:

$$\mathcal{L}\left\{e^{-t/\tau_1}\right\} = \tau_1 - \tau_1^2 s + \tau_1^3 s^2 \ldots \tag{4.10}$$

Therefore by superposition, the general multipole response:

$$v(t) = k_1 e^{-t/\tau_1} + k_2 e^{-t/\tau_2} + \ldots + k_n e^{-t/\tau_n} \tag{4.11}$$

has a **Laplace** transform whose series expansion is:

$$\begin{aligned} \mathcal{V}(s) \;=\; & (k_1\tau_1 + k_2\tau_2 + \ldots + k_n\tau_n) - \\ & (k_1\tau_1^2 + k_2\tau_2^2 + \ldots + k_n\tau_n^2)s + \\ & (k_1\tau_1^3 + k_2\tau_2^3 + \ldots + k_n\tau_n^3)s^2 - \\ & \cdot \end{aligned} \tag{4.12}$$

Now, suppose that the magnitude of τ_1 is much larger than that of all other T'S. In that case the higher order coefficients will be dominated by τ_1. That is for sufficiently large i, $m_i \sim k_1\tau_1^{i+1}$. However if this is the case, then the rows of the moment matrix become linearly dependent, the moment matrix becomes singular, and it becomes impossible to compute an approximation. In practice, as higher order approximations are attempted, the moment matrix tends to become more and more ill-conditioned until a point is reached when it can't be inverted. At that point it becomes necessary to use a lower order approximation.

Huang proposed a technique to enhance the accuracy of moment computation. The technique involves inserting a resistor in parallel with each capacitor and in series with each inductor, where the resistances are sized proportionally to the capacitances and inductances[Hua90]. This effectively shifts all poles and zeros left in the complex plane away from the imaginary axis, thus reducing the tendency of the lowest frequency pole to dominate the other poles. The technique was successfully used to obtain the frequency domain behavior of operational amplifiers and active filters.

Unfortunately, the method does not appear to be suitable for our particular application. One reason is that the resistors that parallel floating capacitors would destroy the tree structure of our circuits. Then the computation of moments would require the formulation and LU factorization of a matrix, a process that is generally superlinear. However, even if a matrix were formulated and factored, it appears that in order to select the shift amount, a priori knowledge about the placement of the poles must be utilized. It does not appear to be possible to choose one shift amount that is good for all circuits. The alternative of trial

and error selection of the shift amount appears to be prohibitively expensive because each new trial modifies the circuit and requires the refactorization of the matrix.

Because of the difficulties associated with generating extremely high or&r waveform estimates, Mom only generates estimates with three poles or less. Fortunately for our purposes low order estimates are usually sufficient.

4.3.3 Unstable Poles

The moments matching procedure occasionally yields defective poles, that is poles that are unrelated to the poles of the system function being approximated. In fact, researchers have long observed that Padé approximations can contain unstable poles even if the system being approximated is stable[Zak73, Cha91]. One source of defective poles is numerical error. From Equation (4.12) we can infer that small errors in the moments may contribute spurious poles to the waveform approximation that have small time constants and coefficients. However, defective poles may also arise in the absence of numerical error. For example, the stable second order impulse response:

$$h(t) = e^{-t} - \frac{1}{3}e^{-t/2} \tag{4.13}$$

has the unstable first order approximation:

$$\hat{h}(t) = -\frac{1}{3}e^{t}. \tag{4.14}$$

Yet if the series expansions of the Laplace transforms of both signals are computed, it can be verified that their two low order terms match ***exactly***.

Defective poles usually pose no problem if they are stable. Because they tend to have small time constants and coefficients they tend to have minimal effect on the waveform estimate. Unfortunately, if the defective pole is unstable it ***dominates*** the waveform. Consequently much work has been done exploring ways of dealing with unstable defective poles. The most common approach is to restrict the problem domain to stable systems. Then the approximation procedure is modified to allow only stable poles[Zak73, GP91].

However, the simulation of digital circuits occasionally requires modeling unstable responses. For example, consider the Schmitt trigger depicted in Figure 4.3. If *out* is

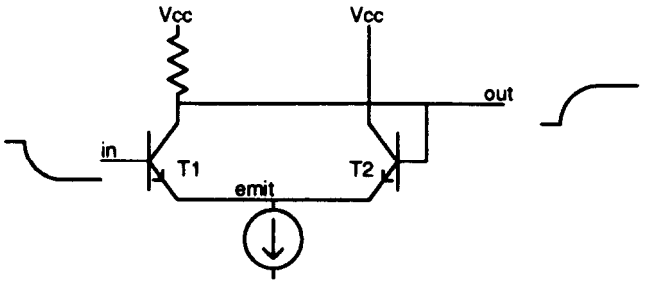


Figure 4.3: Schmitt Trigger

initially low, and *in* falls from high to low, *T2* will eventually switch from *off* to on. When it does, an unstable positive feedback loop is established. That is *T2* turning on tends to turn off *T1* which causes *out* to rise which turns on 7'2 even harder, etc. Because this loop is unstable its response includes an unstable pole. Of course, the unstable pole can persist only briefly. The node *out* very quickly rises to the point where it turns off *T1* completely thus returning the circuit to a stable configuration.

In order to handle the possibility of unstable circuits we employ a heuristic to first distinguish between stable and unstable clusters. When it is known, a priori, that a cluster is stable, all unstable poles can be safely suppressed when estimating its response. Conversely, the response of an unstable cluster is performed in such a way as to require at least one unstable pole.

A fairly simple heuristic can be used to identify commonly occurring unstable digital circuits. They are typically characterized by a feedback loop with an open loop DC gain that is greater than one. Conveniently, this information is generated as a side effect of finding the particular (DC) solution. For our example, because the Schmitt trigger contains feedback, circuit tearing must be employed to solve it. This involves:

1. tearing out the wire connecting the base of *T2* to *out* in order to break the feedback loop

2. computing a value that is equivalent to the open loop DC gain

Thus the routine that computes the particular solution can easily identify clusters with unstable positive feedback.

Our experience with this heuristic indicates that it quite reliably identifies commonly occurring unstable digital circuits. Errors that occur when analyzing more sophisticated circuits (for example Mom will miss the unstable poles of an unstable voltage reference generator) have not been problematic because such circuits are small and can easily be isolated from the predominantly digital design.

4.3.4 Efficiency

Because the generation of a jth order waveform approximation involves the inversion of multiple non-sparse j x j matrices and the factorization of a jth order polynomial, we would expect the cost of waveform approximation to rise superlinearly with the number of poles. This is confirmed by measurements of the execution time of the waveform generation portions of Mom. Table 4.1 gives the average number of **cycles**[3] needed to generate one,

1 Pole	2 Poles	3 Poles
99	380	2030

Table 4.1: Execution Time Required to Generate Waveform Approximations.

two, and three pole waveform estimates for a particular circuit. Note that the cost varies by a factor of 20. As we shall see in a later chapter, this growth can significantly degrade the simulator's efficiency when more complex circuits and models are used.

4.3.5 Error Control

For the sake of computational efficiency it is desirable to employ the lowest order approximation possible. The usual approach is to start with the first order approximation (j = 1) and to compute successive higher order approximations only when necessary. An estimate of the approximation error is produced for each waveform estimate. If that error is above a certain threshold then the next higher order approximation is attempted.

Our estimate for the approximation error is a derivative of one suggested by Shi and Zhang[SZ87]. After generating a jth order approximation from the low $2j$ moments

[3]Estimates of machine cycles are for the MIPS R2000 CPU and were estimated using the ***pixie*** execution profiling tool created by MIPS Computer Systems, Incorporated.

$(m_0 \ldots m_{2j-1})$), we then see how well that approximation matches the $2j$th moment. Comparing the match of higher order moments is computationally efficient and additionally provides a sense of the conditioning of the system. If the 2jth moments and the $(2j+1)$st moments match exactly then there is no use even attempting the next higher order approximation because that system of equations will be ill defined.[4]

In general the threshold for an acceptable approximation error is set to a value between 2% - 20% depending upon the desired level of accuracy. However, it is important to reduce the threshold for large signals because a 10% error in a 1000 volt signal is much worse than a 10% error in a 1 volt signal. At first glance it would seem impossible for such large signals to occur in common forms of digital circuits. However when piecewise linear transistor models are used *asymptotic* swings much greater than the logic swings occur with surprising frequency. For example, consider the two input CMOS NAND gate in Figure 4.4 when piecewise linear MOS Level- 1 models are used. Just after the input has risen, *T1* and

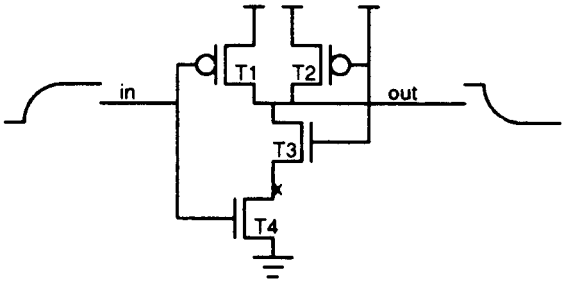


Figure 4.4: 2 Input CMOS NAND Gate.

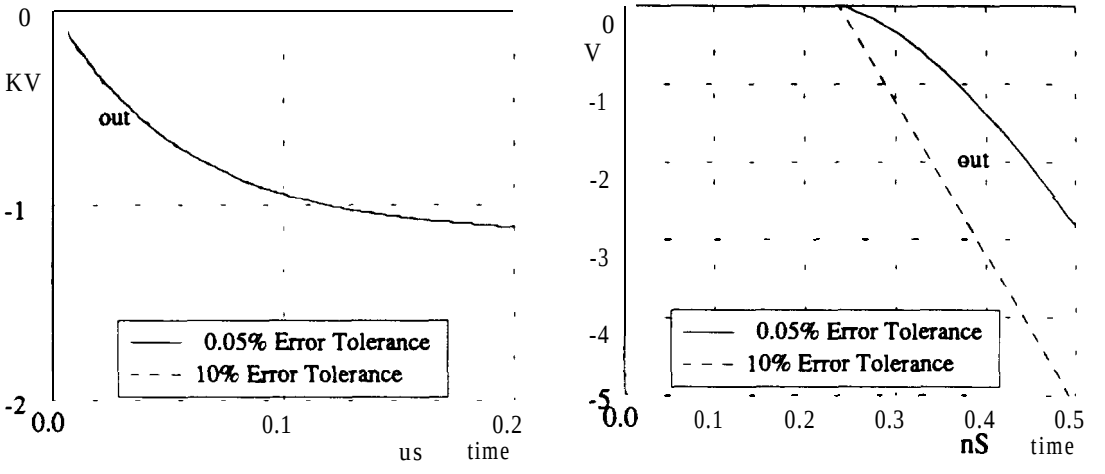
T2 are off and *T3* and *T4* are saturated. The asymptotic final value of the output waveform is computed by finding the DC solution of the network assuming that all devices remain in their present regions of linearity. For our model, $g_m \approx 1/3.6\text{k}$ and $V_t \approx 2$. Since *T4*'s gate is at 5 volts, its internal current source generates $\approx -(5-2)/3.6\text{k} \approx -830\mu\text{A}$ of current. However *T3*'s drain isn't connected to anything. Therefore all of *T4*'s current

[4] To see this, note that if a j pole approximation, $\hat{v}_j(t)$ matches the low 2 j + 2 moments exactly, then so must the j + 1 poleapproximation:

$$\hat{v}_{j+1}(t) \equiv \hat{v}_j(t) + k_{j+1} e^{t/\tau_{j+1}}$$

for $k_{j+1} = 0$. However, for higher order approximation there is no unique value for τ_{j+1}. That is, there are more variables than constraints and the system is ill defined.

must flow into *T4*'s output resistance ≈ 67k. Thus the DC solution forx is about $-830\mu A \times 67k \approx -56$ volts. Furthermore, with no net current flowing through *T3,* the voltage gain from its source to its drain is approximately $67k/3.6k \approx 19$. Thus the DC solution for *out* is about -56 − (56 + 5) x 19 ≈ − 1100 volts! Figure 4.5 compares the response of



(a) Full Swing. (b) Magnified Swing.

Figure 4.5: Large Signal Swing for CMOS NAND Gate.

out predicted by our simulator using 10% and .05% error thresholds. On a scale of 2000 volts (a) the signals are indistinguishable. In fact, the approximation error is just 0.273%. However when the waveform is examined on the scale of 5 volts (the region we actually care about) we see that significant error is introduced. The solution is to scale the error threshold of large signals inversely proportional to their size. The lowest order moment, m_0, is used to estimate the size of the signal.

4.3.6 Frequency Scaling

Another source of errors in the moment matrix arises essentially from a poor choice of units of time. Consider the moments of a multipole response (Equation 4.12). If the τ's are all of the order of lps then $m_0 \sim 10^{-12}, m_1 \sim 10^{-24}, m_2 \sim 10^{-36}, \ldots$. That is, the magnitude of each moment will be approximately 12 orders of magnitude smaller than its predecessor. This large variation in moments introduces large variations in the magnitude of the elements of the moment matrix, thereby rendering it uninvertible.

However, note that the choice of units of time is arbitrary. If, for example, we had chosen lps to be the unit of time, then the r's would all be of the order of 1 and the large variation between successive moments would be eliminated. **Pillage[PR90]** suggested scaling frequencies by the ratio $|m_1/m_0|$. However, consider the following set of moments obtained from an actual simulation.

$$\{m_0, m_1, \ldots m_5\} = \{0, 0, 1.4 \text{ x } 10^{-35}, -3.0 \text{ x } 10^{-30}, 5.5 \times 10^{-39}, -5.9 \text{ x } 10^{-48}\} \quad (4.15)$$

These moments were particularly problematic because the 18 orders of magnitude difference in the sizes of the moments made it impossible for Mom to compute any waveform approximation. Note that because $m_0 = 0$ Pillage's method can't be applied directly.

An obvious modification would be to use the ratio $|m_{k+1}/m_k|$ where ***k*** is smallest subscript for which $m_k \neq 0$. However for our example this makes things even worse.

$$\{m_0, \ldots m_5\} = \{0, 0, 3.3 \times 10^{-46}, -3.3 \text{ x } 10^{-46}, 2.9 \text{ x } 10^{-60}, -1.5 \text{ x } 10^{-74}\} \quad (4.16)$$

Whereas the ratio of the magnitude of the smallest moment to that of the largest used to be 18 orders of magnitude, it is now **28!**

A better approach would be to note that the asymptotic growth of high order moments is dominated by the τ of largest magnitude. If the moments are normalized by the largest time constant then this growth will be curtailed. Furthermore, a good estimate for this time constant would be $|m_k/m_{k-1}|$ where m_k is the highest order moment computed. When this is done the moments become:

$$\{m_0, \ldots m_5\} = \{0, 0, 1.2 \times 10^{-17}, -2.4 \text{ x } 10^{-3}, 4.1 \text{ x } 10^{-3}, -4.1 \times 10^{-3}\} \quad (4.17)$$

Note that the high order moments are almost identical and the the ratio of the magnitudes of the largest and smallest (now 14 orders of magnitude) is better than before the frequency scaling. This procedure has been found to be useful for a large body of simulations.

Finally, note that the example given above is a rare example for which none of the frequency scaling procedures described above was able to condition the moment matrix sufficiently for it to be inverted and a waveform approximation produced. This prompted us to consider whether or not it was possible to do any better. In fact, it turns out that it is possible to find a scaling factor that is "optimum" in the sense that it minimizes the ratio of

the magnitudes of the largest and smallest moments (See Appendix D). When the optimum scaling factor is used:

$$\{m_0, \ldots m_5\} = \left\{0, 0, 2.6 \times 10^{-27}, \text{-7.4 x } 10^{-18}, \text{1.8 x } 10^{-22}, \text{-2.6 x } 10^{-27}\right\} \quad (4.18)$$

the ratio of the largest to smallest moments drops to just 9 orders of magnitude, the moment matrix could be inverted, and a waveform approximation could be produced. Unfortunately, the procedure appears to be too expensive to justify its use. For almost all other waveforms the improvement over using one of the heuristics was negligible and optimal frequency scaling increased the execution time of the program by up to 10%.

4.4 Demonstration

In order to evaluate how well moments matching works for our application, a variety of ring oscillators were simulated. For each circuit, three different simulations were performed. First, SPICE was run using its nonlinear transistor models. Second, Mom was run using each of the piecewise linear models described in Chapter 3. Third, SPICE was run using Mom's piecewise linear models. For each run the period of oscillation was recorded. For each circuit a plot was made overlaying the responses of all three simulations. Table 4.2 presents some statistics gathered from the simulations. The first three columns give the

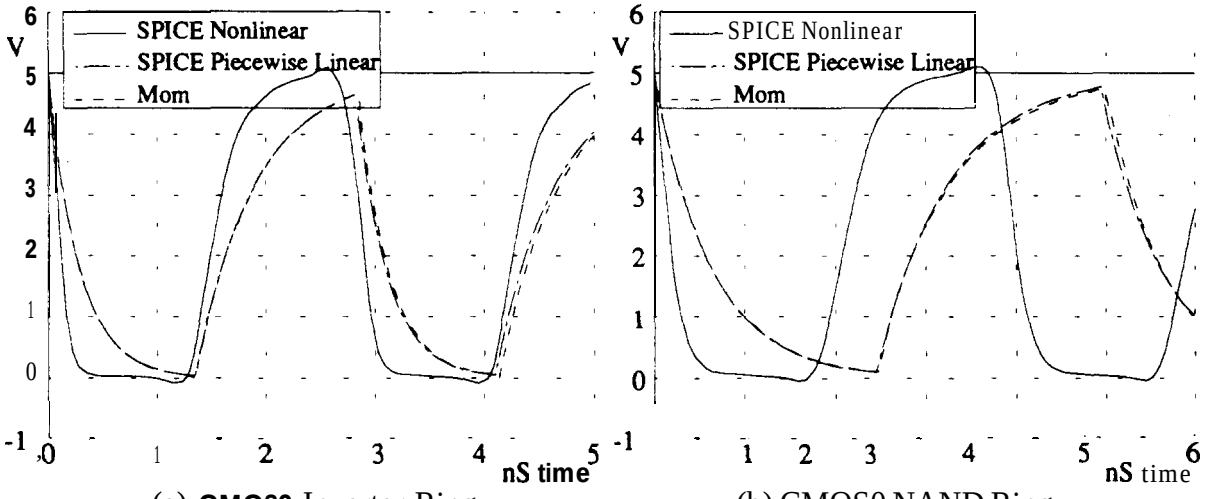
	Pole Distribution			Model	Waveform
	% 1 Pole	% 2 Poles	% 3 Poles	Error	Error
CMOS0 Inverter Ring	100.0	0	0	0.8	1.2
CMOS0 NAND Ring	73.8	26.2	0	28.1	0.3
CMOS 1 Inverter Ring	7.5	90.7	1.8	1.1	0.0
CMOS 1 NAND Ring	16.7	83.2	0.1	5.2	0.1
BJT0 ECL Ring 10%	25.2	31.3	43.4	9.2	0.3
BJT0 ECL Ring 20%	53.9	46.1	0	9.2	6.3
BJT1 ECL Ring	11.8	20.2	67.8	8.0	0.1
BJT2 ECL Ring CapLevels=1	2.2	22.3	75.6	2.5	0.3

Table 4.2: Benchmark Statistics.

fraction of 1, 2, and 3 pole responses produced by Mom. The last two attempt to identify the origin of simulation errors.

The column labeled "Model Error" gives the error of the period of the SPICE piecewise linear simulation relative to the SPICE nonlinear simulation. "Model Error" is the error incurred by modeling nonlinear SPICE transistors using the simple piecewise linear models. The column labeled "Waveform Error" gives the error of the period predicted by Mom relative to the SPICE piecewise linear simulation. "Waveform Error" is the error produced by the moments matching waveform approximation procedure. In principle the total error of our simulator could be the sum of these two errors. In practice, as will be seen in Chapter 7, the errors can randomly sum or cancel.

The circuit "CMOS0 Inverter Ring" is a 5 stage CMOS ring oscillator using inverters composed of piecewise linear MOS Level-O transistors (Figure 4.6 (a)). Because of



(a) **CMOS0** Inverter Ring (b) CMOS0 NAND Ring

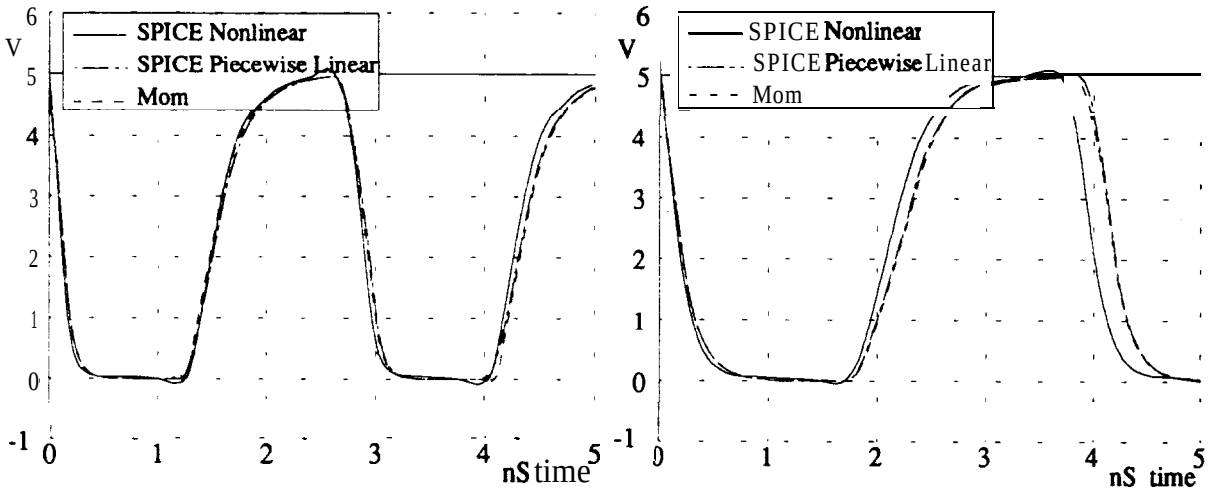
Figure 4.6: CMOS Level-O Ring: SPICE vs PWL vs Mom.

the simplicity of the piecewise linear model and circuit (every cluster reduces to just an RC) the piecewise linear response is exactly single time constant. Thus the Mom and SPICE piecewise linear simulations match well although the piecewise linear and nonlinear simulations don't. This is expected because the switched resistor model can correctly predict the switching delay of a CMOS gate but not necessarily the waveform. The match of the periods is a consequence of using this circuit to calibrate the switched resistor model.

Figure 4.6 (b) shows what happens when 2 input NAND gates are used in the ring ("CMOS0 NAND Ring"). Again note the close match between the Mom and SPICE

piecewise linear simulations. However, the error due to the switched resistor model has increased to 28%. This is not surprising because it has already been observed that the use of switched resistors to model series stacks of velocity saturated devices can lead to errors of the order of 29% (Figure 3.15). Also note that the circuit is slightly more complex with the result that 2 pole approximations are occasionally used for the intermediate node of the NAND stack.

The circuits "CMOS1 Inverter Ring" and "CMOS1 NAND Ring" (Figure 4.7 (a) and (b)) illustrate the use of MOS models with gain. The gain couples multiple nodes together

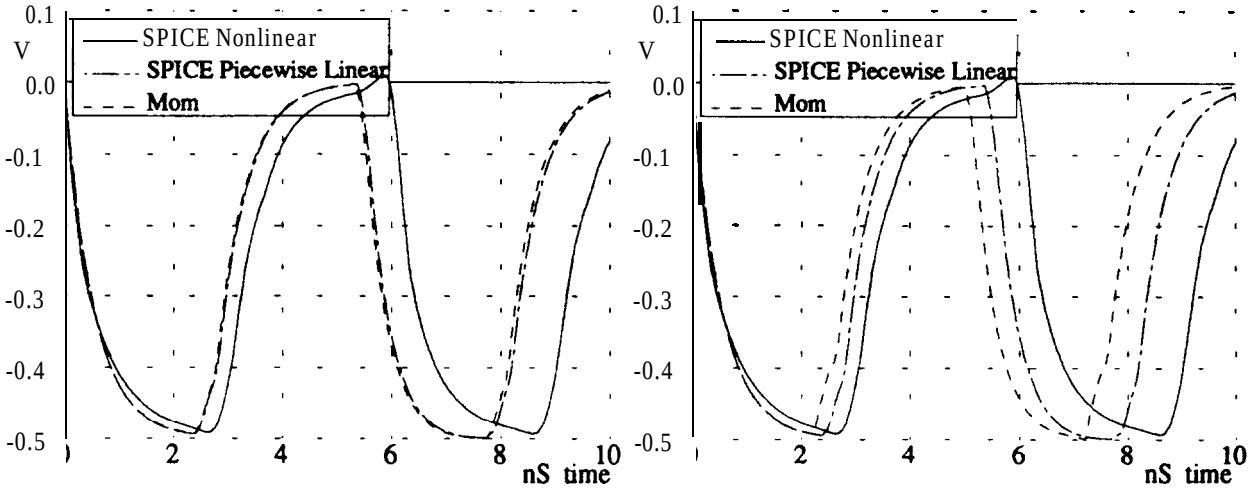


(a) CMOS 1 Inverter Ping (b) CMOS 1 NAND Ring

Figure 4.7: CMOS Level-l Ring: SPICE vs PWL vs Mom.

thereby increasing the complexity of the response. Most of the waveforms are now two poles and the match with SPICE is much improved.

The circuits "BJT0 ECL Ring 10%" and "BJT0 ECL Ring 20%" (Figure 4.8 (a) and (b)) are 9 stage ECL ring oscillators using the piecewise linear Level-O bipolar model. When an error threshold of 10% was used, the SPICE and Mom responses match almost exactly. However the 0.3% error of the waveform approximation is swamped by the 9.2% error of the piecewise linear model. Perhaps a better balance between speed and accuracy is achieved by increasing the error tolerance to 20%. The resulting elimination of 3 pole responses can substantially speed up the simulation, albeit at the expense of increasing the waveform approximation error to 6.3%



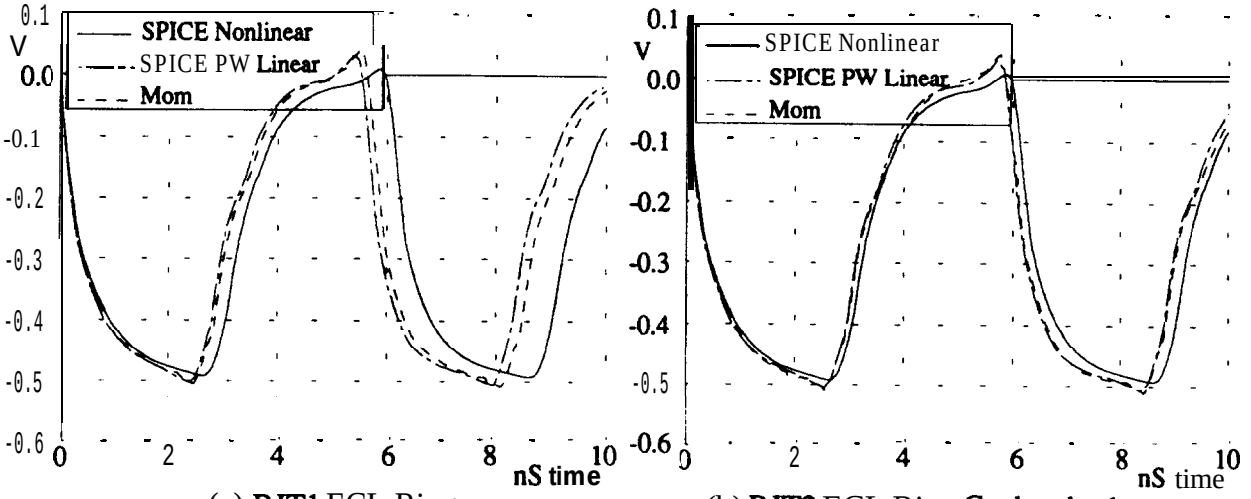
(a) **BJT0** ECL Ring 10% (b) **BJT0** ECL Ring 20%

Figure 4.8: ECL Level-O Ring: SPICE vs PWL vs Mom.

Finally, the ECL ring oscillator was simulated with the Level-l and Level-2 bipolar models (Figure 4.9). **"BJT1** ECL Ring" employs the piecewise linear Level-l bipolar model, and **"BJT2** ECL Ring **Caplevels=1"** employs the piecewise linear Level-2 bipolar model. (The meaning of "Caplevels=l" will be explained in the following chapter.) Note that for these circuits the approximation error threshold had to be reduced from its default value of 10% to 2% in order to achieve the degree of matching depicted in the figures and table.

From this set of simulations a number of observations can be made. In general as the complexity of the models and circuits increases, so does the number of poles needed to represent the response. Both gain and floating capacitors tend to couple together multiple nodes thereby complicating the response and requiring greater numbers of poles. Additionally, as the models become more complex, the error threshold should be reduced commensurately. It is senseless to pay for a low waveform approximation error if it is going to be swamped by the modeling error, and conversely.

Overall, third order waveform approximations appear to be adequate. Note that the error due to waveform approximation can almost always be made smaller than the modeling error. Additionally, the piecewise linear models do quite well despite their simplicity. The switching delay errors due to the Level-2 bipolar and the Level-l MOS models are under



(a) BJT1 ECL Ring (b) BJT2 ECL Ring Caplevels=1

Figure 4.9: ECL Level-l & -2 Rings: SPICE vs PWL vs Mom.

6%.

4.5 Limitations

4.5.1 Unstable Responses

It is apparently more difficult to apply the waveform approximation technique to unstable circuits than to stable circuits. One reason for this is that the Padé approximation tends to capture the low frequency behavior of the response. That is $\hat{H}(s)$ first converges to $H(s)$ in the vicinity of the origin of the complex s plane[Hua90]. For stable circuits this behavior is desirable because the low frequency poles tend to dominate the behavior of the response (hence ***the*** term ***dominant*** time constant). Unfortunately, for unstable circuits the response is dominated by the unstable poles which may not be the lowest frequency poles. In that case the most important poles may be approximated with the greatest error.

This poor approximation of unstable poles is exacerbated by the ***exponential growth*** with time of the approximation error: v(t) – G(t) of an unstable system. In the worst case, this increase of the error with time can actually lead to a misprediction of not only the timing behavior of a circuit but also its ***logical*** behavior. In contrast, for a stable system

both the response and its estimate decay exponentially with time. Consequently for a stable system we are always assured that the error decays exponentially, and that voltages always eventually converge to their correct final values.

To illustrate, consider the response of node *emit* of the Schmitt trigger (Figure 4.3) when *T2* first turns on. The complete third order response may be compared to the first and second or&r approximations:

$$\mathbf{v}(t) = 0.230e^{-t/2{,}000ps} - 0.001e^{-t/13ps} + 0.007e^{t/132ps} \tag{4.19}$$

$$\hat{\mathbf{v}}_1(t) = 0.650e^{-t/1{,}368ps} \tag{4.20}$$

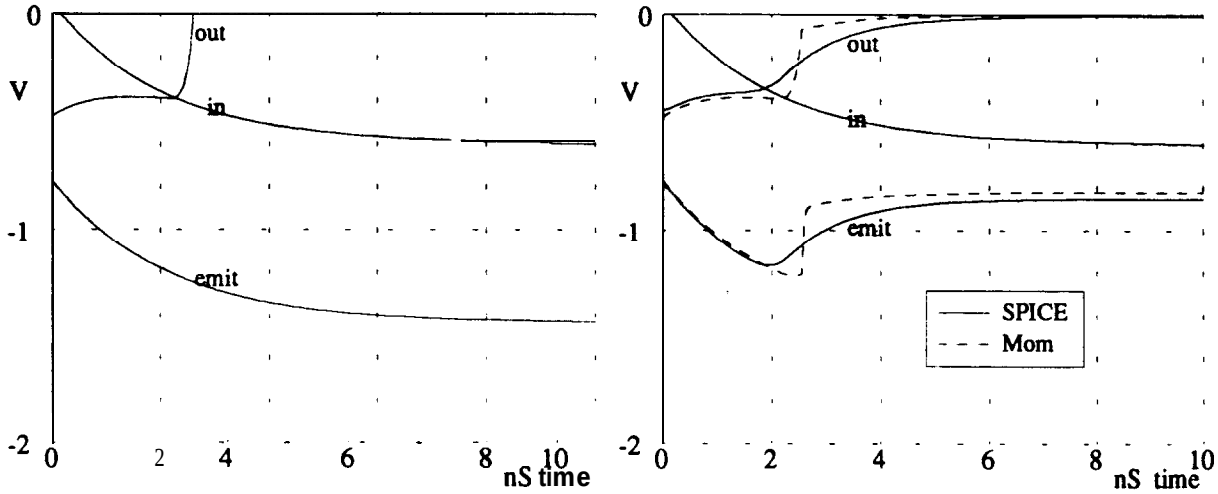
$$\hat{\mathbf{v}}_2(t) = 0.006e^{t/158ps} + 0.230e^{-t/2{,}000ps} \tag{4.21}$$

According to our metric the first order estimate appears to be adequate. In fact, the first *seven* moments match to within 14%. For stable systems such a match usually implies an acceptable approximation. However the first order approximation is missing the crucial unstable pole. Figure 4.10 (a) shows that the use of this estimate leads to an incorrect logical simulation. The lack of an unstable pole causes *emit (the* lowest trace) to erroneously continue falling when *T2* turns on (at $t \sim$ 2ns). In contrast out (initially the middle trace) correctly begins to rise exponentially. The result is that *T1* never turns off and *out* veers wildly off to $+\infty$.

The first order approximation is readily rejected because it has been determined that the circuit is unstable and yet the estimated response lacks an unstable pole. The second order approximation does possess an unstable pole. Furthermore the time constant and coefficient of its dominant pole is accurate to within 3 significant figures and its first seven moments match to within .009%. However, the use of the second order approximation still shows some noticeable errors (Figure 4.10 (b)). Only when the error tolerances are tightened to force the use of the full 3 pole response do we get a good match with SPICE (Figure 4.11).

4.5.2 Circuits with High Gain

Another limitation arises when circuits with extremely high gain place extreme demands on the accuracy of the waveform approximation. The CMOS NAND gate described in



(a) 1 Pole Approx of *emit* (b) 2 Pole Approx of *emit*

Figure 4.10: Simulation of Schmitt Trigger

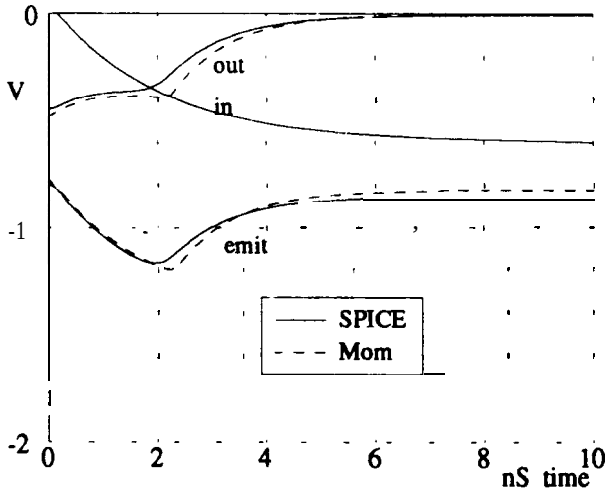


Figure 4.11: Schmitt Trigger using 3 Pole Response for *emit*

Section 4.3.5 is an instance of such a circuit. A more extreme example is the cascade of ECL inverters in Figure 4.12 where all stages are initially biased into their linear regions

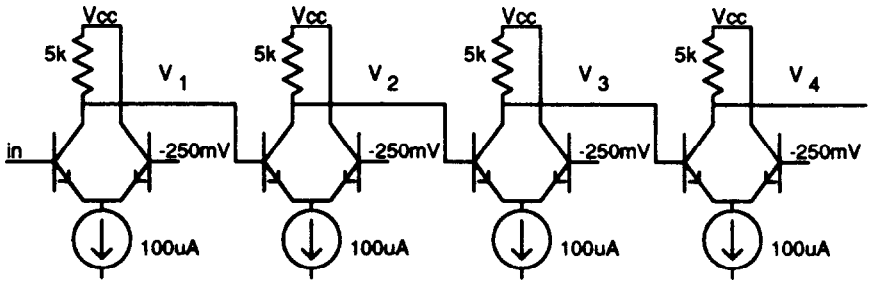
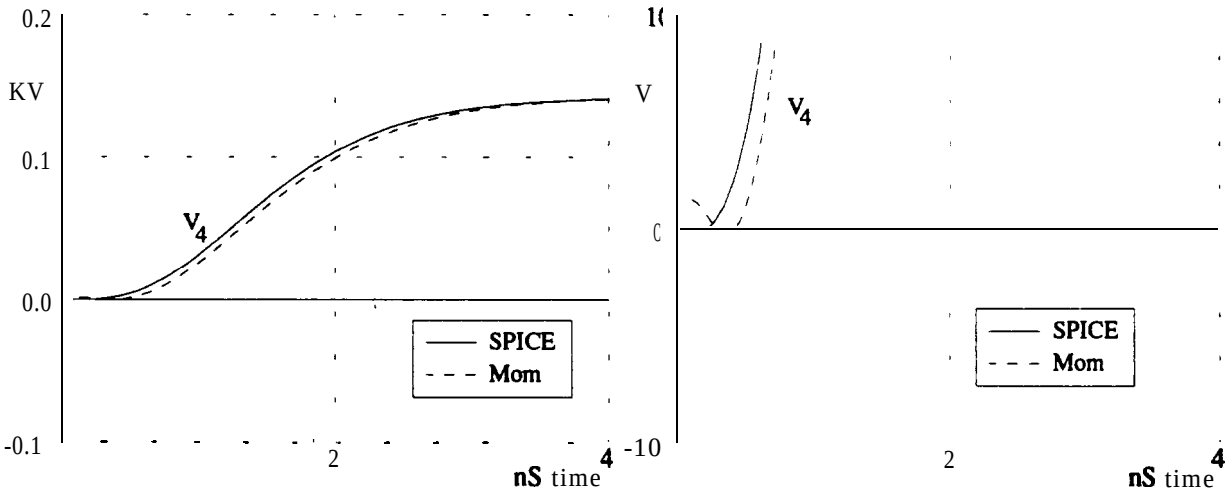


Figure 4.12: Cascade of ECL Inverters

($V_1 = V_2 = V_3 = V_4 = V_{ref}$ = -250mV) and in rises from -250mV to Ov. For the initial segment of the response, all the inverters remain in the linear region. Since each stage has a gain of approximately 5 the initial input signal swing of 250mV gets amplified to about 156 volts by the last stage.

Figure 4.13 compares the initial segment of V_4 predicted by Mom with that generated



(a) 200 Volt scale (b) 10 Volt Scale

Figure 4.13: Initial Segment in the Response of ECL Cascade.

by SPICE. Our simulator's response consists of 3 poles with an approximation error of .25%. On a scale of 200 volts (a) the approximate response looks quite good. However on a scale of 10 volts (b) it is apparent that the approximation is useless. In the region of

interest (i.e. near t=0) the approximation doesn't even move in the correct direction. The problem is that the asymptotic swing of the segment is much larger than swings that can actually occur in the circuit. Thus, the required approximation error must be vanishingly small. Unfortunately, in this case the approximation is already composed of three poles, and our simulator can do no better.

This example is admittedly artificial because digital circuits are extremely non-linear thus making it unlikely that many successive logic stages will be simultaneously biased into their high gain regions. In fact, this situation arose because of a peculiarity of the initial DC solution; the perfect symmetry of the ring caused the simulation to start from the metastable state. Although the ring oscillator's problem can be solved by modifying the initial DC solution, it is possible that circuits similar to the CMOS NAND gate mentioned above will require a more general solution.

4.6 Summary

Piecewise linear models require more sophisticated waveform approximation procedures because as the complexity of the transistor models increases so does the complexity of the responses of circuits using those models. Therefore, Mom uses a general ***moments matching*** procedure to generate higher order waveform estimates. The procedure involves modeling an actual high order system using a low order system chosen such that low order moments of the impulse responses of both systems match. Because the response of the low order system can be computed exactly it is used to approximate the response of the high order system.

Practical implementations of this procedure must address a number of issues. Floating capacitors occasionally lead to problems with low order approximations, roundoff errors in the moments ultimately limit the order of the approximation, approximations can sometimes contain unstable defective poles, and the cost of generating an approximation rises superlinearly with the number of poles. While these issues need to be addressed, practical compromises can deal with them.

Overall, the procedure works well. An examination of a number of CMOS and ECL circuits indicates that 3 poles are usually adequate. In fact, comparisons with SPICE

simulations indicate that the waveform approximation works so well that the errors due to waveform approximation are almost always swamped by the piecewise linear modeling errors.

A couple of limitations persist. First, it is much more difficult to simulate unstable circuits because unstable poles frequently have small coefficients and time constants and hence contribute minimally to the moments. The result is that small errors in matching the moments generate large errors in the time domain response. Second, for brief moments during a switching transient certain combinations of device states can yield circuits with voltage swings that far exceed what is physically possible. The consequence is that only an infinitesimal initial portion of the waveform approximation is used, thereby placing extreme demands on the accuracy of the approximation.

Chapter 5

Moment Computation

In the previous chapter the general ***moments matching*** procedure was introduced and shown to be useful for predicting the responses of circuits containing piecewise linear models. This chapter is concerned with one particular "implementation detail" of the moments matching procedure: the computation of moments from the circuit. Moment computation is carefully considered here because in previous work[PR90] it tended to dominate the overall cost of waveform approximation. One of the factors that made moments matching attractive for MOS timing analysis and switch-level simulation was the relative ease with which moments could be computed from the circuit. When the switched-resistor transistor model was used, the task of computing moments reduced to finding the DC solution of a resistor tree, something that was readily done in linear time without formulating or LU factoring general sparse circuit matricies.

However, when piecewise linear transistor models are allowed, the simulator must deal with circuits that are no longer RC trees. This chapter generalizes RC tree analysis along two dimensions. First, RC tree analysis is extended to apply to piecewise linear transistor models. This generalization retains the efficiency of RC tree analysis for the transistor-capacitor trees found in MOS circuits and the current steering trees found in ECL circuits. Second, circuit tearing is used to handle non-tree topologies and feedback. If the number of branches that need to be tom in order to get a feedback-free tree is small and bounded, then even these more complex circuits can be analyzed efficiently.

Finally, this chapter addresses the efficiency problem caused by allowing floating capacitors in transistor models. Floating capacitors can potentially couple together all nodes in the circuit thereby eliminating the ability of the simulator to take advantage of latency. From a practical standpoint, however, this isn't a problem because the circuit can be repartitioned by simply ignoring coupling through all but a limited number of levels of floating capacitors.

5.1 Background

Many methods have been proposed for the computation of moments. Because of their emphasis on efficiency, switch-level simulators have historically attempted to take advantage of the tree-like topology of most digital MOS circuits. Initially, only grounded capacitors were considered and heuristics were used to break loops so that tree analysis could be applied in linear time[Hor83]. Raghunathan and Thompson[RT85a] and Chu and Horowitz[Chu88] extended tree analysis to handle leaky trees, multiple drivers, and charge sharing while retaining the efficiency of RC tree analysis. Chan[Cha88] extended RC tree analysis to handle floating capacitors.

A number of derivatives of tree analysis have been proposed to handle circuits that are nearly trees. Although these procedures do not have linear complexity, if the number of loops is small, they can be nearly as efficient. Lin and Mead[LM84] proposed an algorithm based on ***Gauss-Seidel relaxation.*** Chan and Karplus[CK89] and Pillage and Dutta[PD90] handle edges closing loops in the tree using ***brunch tearing.*** Ratzlaff et al.[RGP91] utilize a procedure that can be viewed as the application of ***node tearing*** techniques.[1]

General purpose circuit analysis techniques have also been applied to moment computation. Shi and Zhang[SZ87] formulate the problem in terms of ***nodal analysis,*** thereby removing the topological restrictions of tree analysis while allowing independent sources, dependent sources, and floating capacitors. Pillage and Rohrer[PR90] use ***tree link analysis*** to compute moments for networks that may additionally include inductors. However,

[1] Although Ratzlaff et al. do not present their work in terms of *node tearing*, we will justify this point of view later.

although these techniques are more general, they may not be as efficient as RC tree analysis for the particular case of RC trees. Pillage and Dutta point out that when tree-link analysis is applied to RC trees the **loop/cutset** matrix may not be sparse thus leading to super-linear **complexity[PD90].** Furthermore, although properly ordered nodal equations for trees can be LU factored in linear time due to the absence of **fill-ins[SZ87],** it may still be better to utilize an alternate formulation. While studying the DC solution of power nets, **Branin[Bra80]** found that his tree-based method could solve the network in about the time it took simply to formulate the sparse nodal equations. Ratzlaff et al. found that their hybrid tree/nodal analysis was up to two orders of magnitude faster than general LU factorization.

Since our goal is to duplicate the efficiency of switch-level simulators, we have chosen to generalize RC tree analysis. Our analysis is an extension of that of Chu. We first review his analysis in the next section.

5.2 Moment Computation for Leaky Resistor Trees

Chu[Chu88] extended RC tree analysis to include RC trees driven by multiple sources (Figure 5.1). These topologies may continue to be viewed as trees if one redefines

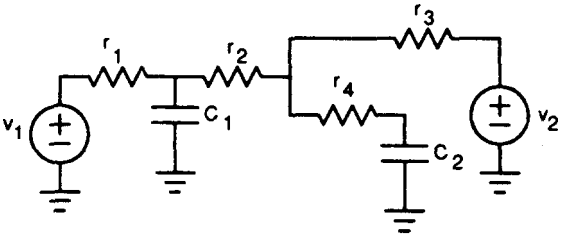


Figure 5.1: Leaky resistor tree.

leaf nodes to be nodes connected to one transistor terminal and either a grounded current or voltage source.[2] The root node is arbitrarily selected from the non-leaf nodes (See Figure 5.2). We will refer to such topologies as a ***leaky trees***.

To review Chu's approach: consider the leaky tree in Figure 5.2 that results when the

[2]We include nodes connected to current sources with zero current.

[3]Chu's thesis describes the analysis in terms of "moving capacitors". We present his work from the slightly different perspective of Norton analysis applied to the network.

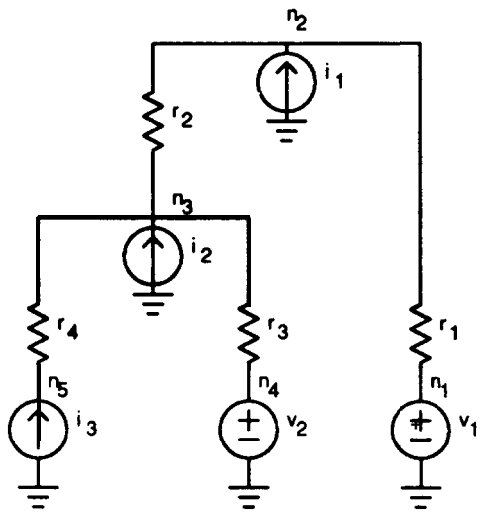


Figure 5.2: Computing moments for leaky tree.

capacitors are replaced by current sources for the purpose of computing moments. The circuit has been redrawn to suggest its tree structure, with the leaf nodes n_1, n_4, and n_5 at the bottom and the root node n_2 at the top. The DC solution can be found by making two passes over the network. The first pass starts at the leaves of the tree and ascends to the root. The second pass starts at the root of the tree and descends to the leaves.

We begin the first pass with the resistors connected to leaf nodes (r_1, r_3, and r_4). For each resistor we compute the Norton equivalent seen looking into its upper terminal. Once we have computed the Norton equivalents of all resistors descending from a particular node (after the first iteration n_3 becomes such a node) we can combine them with the capacitor current sources to produce the Norton equivalent seen looking out of the lower terminal of the (single) resistor ascending from that node (for n_3 this is r_2). We record this aggregate Norton equivalent at the node for use in the second pass. The first pass continues iteratively, replacing each ascending resistor by the Norton equivalent seen looking into its upper terminal and, in turn, computing the Norton equivalent seen by the parent node's ascending resistor. The iteration terminates when we have computed the Norton equivalent of all the resistors descending from the root node.

The second pass starts by solving for the voltage at the root node. This is possible because the root node has no resistors ascending from it and in the first pass we saved for each node the Norton equivalent of all resistors (and capacitor current sources) descending

from that node. Once we know the root's voltage, we can solve for the voltage of its children by utilizing the Norton equivalent saved at each child. We continue descending the tree until we've solved for all the voltages.

In summary, the process utilizes two mapping computations. In the first pass (See Figure 5.3) we are given r_1, ii, and r and need to find r_2 and i_2

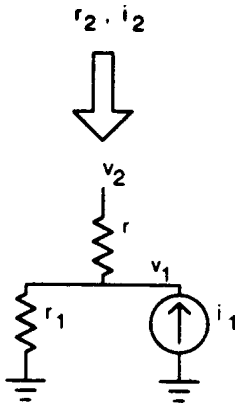


Figure 5.3: Norton calculations.

$$r_2 = r_1 + r \tag{5.1}$$

$$i_2 = i_1 \frac{r_1}{r_1 + r} \tag{5.2}$$

In the second pass we are given, in addition, v_2 and we need to find v_1

$$v_1 = \frac{v_2 r_1 + i_1 r_1 r}{r_1 + r} \tag{5.3}$$

Later, we will show that these two computations can still be performed even if resistor r is replaced by a transistor modeled using piecewise linear functions.

Finally, it should be noted that one of the factors contributing to the efficiency of tree analysis is that the formulation of the equations occurs largely as a side effect of the process of deciding which nodes are affected by the switching event. A cluster consists of all nodes connected by the channels of conducting transistors. Therefore, when a transistor switches, a ***depth first search*** is performed on the interconnection graph[4] along only those edges representing conducting transistors. The construction of a list of those edges in the order

[4] A static interconnection graph of the network is constructed once during a preprocessing phase.

they are encountered, plus a bit in each edge indicating which of its two channel terminals is higher in the tree, constitutes the formulation of the equations.

5.3 Leaky Trees of Three Terminal Networks

The switched resistor model is particularly easy to deal with because of the simplicity of the circuit models that represent its behavior in each region of linearity: the transistor is represented by either a resistor (if it is on) or an open circuit (if it is off). Such simple circuits are not always sufficient. For example, to model the dependence of the drain current on the gate source voltage, the MOS Level-l transistor model must employ a dependent current source. In general, we would like to allow interconnections of resistors and dependent and independent current and voltage sources in transistor models. Once this is done it is no longer apparent that a simple tree walk can be used to find the moments.

This section demonstrates the rather surprising result that trees of piecewise linear devices can be solved as easily as trees of resistors. This general result has only two minor restrictions: the coupling from the gate (base) to the source and drain (emitter and collector) must be unidirectional, and the tree must be feedback free: that is no gate (base) of any transistor in the tree may be connected to any node in the same tree.

In order to compute the moments of a transistor-capacitor tree we need to find the DC solution of a corresponding tree obtained by setting independent DC sources to zero, replacing inductors with voltage sources, and replacing capacitors with current sources. Furthermore, because we assume inputs are unidirectional, MOS gates are considered to be driven by independent, possibly exponentially time varying voltage sources. It can be shown that when formulating the circuit to compute the $(k+1)$st moments, each time-varying source should be replaced by a DC source set equal to the $(k+1)$st moment of its waveform (see Figure 5.4).

In any particular state, each piecewise linear device is equivalent to some linear network. Assuming that each transistor remains in its present state for some finite amount of time, we group each transistor with the voltage source driving its gate and represent the interface that the pair presents to the network by ***the short-circuit admittance parameters*** of a ***three terminal network***[BS65] (Figure 5.5). The parameters are defined by extracting two

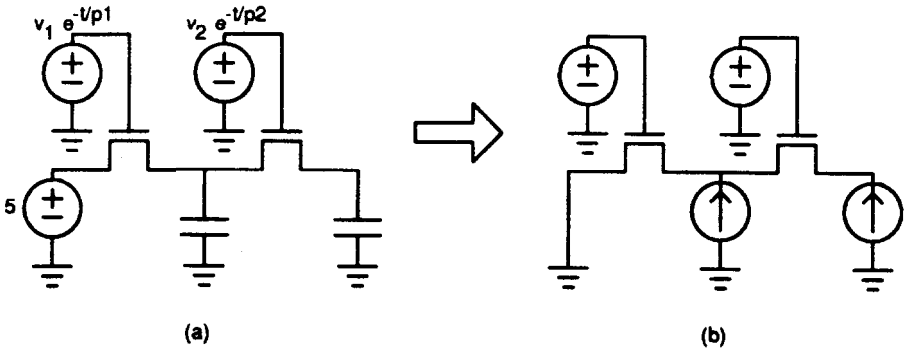


Figure 5.4: Transistor-capacitor tree.

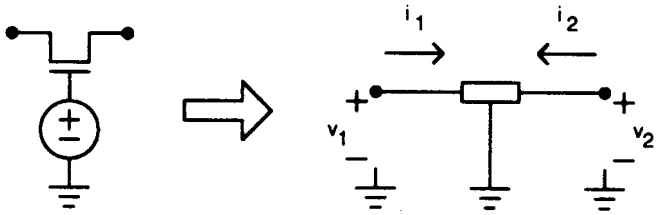


Figure 5.5: Three terminal network model.

voltage ports, one from each terminal to ground[CL75].[5]

$$\begin{bmatrix} i_1 \\ i_2 \end{bmatrix} = \begin{bmatrix} y_{11} & y_{12} \\ y_{21} & y_{22} \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \end{bmatrix} + \begin{bmatrix} i_{s1} \\ i_{s2} \end{bmatrix} \tag{5.4}$$

Figure 5.6 gives a physical interpretation of the six parameters of the admittance formulation.

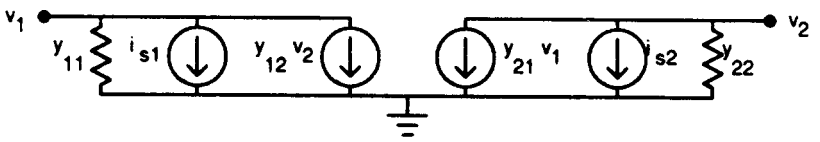


Figure 5.6: Circuit interpretation of admittance parameters.

In order to find the DC solution of leaky trees of these networks[6] we need to be able to do two things. If we replace resistor r in Figure 5.3 by the circuit in Figure 5.6 it can be

[5] It is not always possible to extract two voltage ports for a particular model. For example, admittance parameters cannot be determined for a voltage source. Such devices are handled as special cases. However, to simplify the discussion we assume that the admittance representation exists.

[6] A special case arises if either $y_{12} = 0$ or $y_{21} = 0$. In that case rather than treating the transistor as a branch in a tree it is potentially more efficient to treat it as an arc between clusters. See Section 5.5

shown that if $g_1 = 1/r_1$ and i_1 are known then $g_2 = 1/r_2$ and i_2 are given by

$$i_2 = -\left[i_{s1} + \frac{Y12}{Y22 + 91}(i_1 - is_2)\right]_I \tag{5.5}$$

$$g_2 = y_{11} - \frac{y_{12}y_{21}}{y_{22} + g_1} \tag{5.6}$$

If, in addition, v_2 is known then vi is given by

$$v_1 = \frac{i_1 - i_{s2} - y_{21}v_2}{91 + y_{22}} \tag{5.7}$$

Thus it is possible to generalize the DC analysis of leaky trees of resistors to leaky trees of three terminal networks. Because the above equations take a constant amount of time to compute, the leaky tree analysis remains $O(n)$ in the number of devices in the circuit irrespective of the complexity of the models.

5.4 Series-Parallel Combination

The analogy with resistors goes even further. These three terminal networks are also amenable to series-parallel combination. The admittance parameters of the parallel combination of two networks can be found by simply 'summing their corresponding parameters. The parameters of a series combination can be derived from the series combination of two of the circuits in Figure 5.6. If we let superscripts of 1 and 2 distinguish between the parameters of the two circuits then

$$Yll = y_{11}^1 - \frac{y_{12}^1 y_{21}^1}{y_{22}^1 + y_{11}^2} \tag{5.8}$$

$$y_{12} = -\frac{y_{12}^1 y_{12}^2}{y_{22}^1 + y_{11}^2} \tag{5.9}$$

$$y_{21} = -\frac{y_{21}^1 y_{21}^2}{y_{22}^1 + y_{11}^2} \tag{5.10}$$

$$Y22 = y_{22}^2 - \frac{y_{21}^2 y_{12}^2}{Y22 + y_{11}^2} \tag{5.11}$$

$$i_{s1} = i_{s1}^1 - \frac{y_{12}^1}{y_{22}^1 + y_{11}^2}\left(i_{s2}^1 + i_{s1}^2\right) \tag{5.12}$$

$$i_{s2} = i_{s2}^2 - \frac{y_{21}^2}{y_{22}^1 + y_{11}^2}\left(i_{s2}^1 + i_{s1}^2\right) \tag{5.13}$$

Parallel combination can be useful for breaking small loops produced, for example, by CMOS transmission gates. Although such loops can be handled using the more general tearing techniques described later in the chapter, parallel combination is more efficient.

5.5 Coupled Clusters

The previous section showed how to compute the moments of a single cluster assuming that all its inputs are known. For Rsim this is sufficient because the switched resistor model always yields clusters that are independent. Mom's more general models, however, may include coupling between clusters which requires that multiple clusters be analyzed simultaneously.

Switch-level simulators usually exclude floating capacitors. However, Mom treats a floating capacitor as a ***bidirectional*** coupling which requires the simultaneous evaluation of both terminals. As outlined in Section 4.2 the computation of the moments proceeds by replacing capacitors with current sources. However, instead of inserting a single ***flouting*** current source we insert two ***grounded*** current sources (Figure 5.7). When computing the

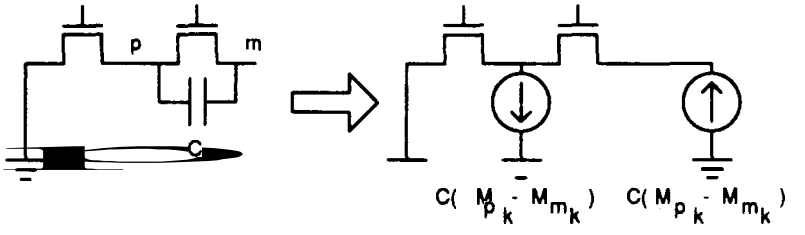


Figure 5.7: Floating capacitor connecting same cluster.

$(k+1)$st moments, the current of an inserted current source becomes

$$i_c = C(M_{p_k} - M_{m_k}) \tag{5.14}$$

where M_{p_k} and M_{p_k} represent the kth moments of voltages of the nodes connected to the plus and minus capacitor terminals, respectively.

If both terminals are connected to the same cluster then the moment computation proceeds as for grounded capacitors. However, if the capacitor links two otherwise disconnected clusters (Figure 5.8) then the moments for both clusters must be computed in

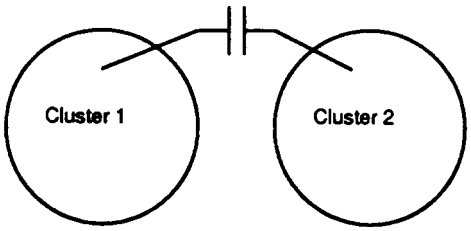


Figure 5.8: Capacitive Coupling Between Clusters.

lock step because the $(k+1)$st moments in each cluster depend upon the capacitor current which, in turn, is a function of the ***kth*** moments of nodes in ***both*** clusters. Thus the 0th moments are computed for Cluster 1 and Cluster 2, followed by the 1 st moments for Cluster 1 and Cluster 2, etc.

Coupling between clusters can also be caused by dependent sources in the transistor models. To see this, consider the circuit in Figure 5.9. If *T2* is modeled by the switched

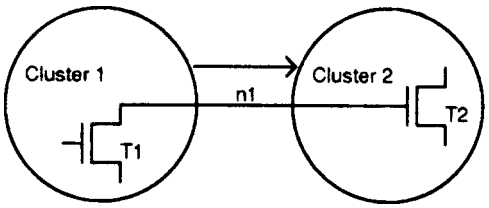


Figure 5.9: Gate to Channel Coupling.

resistor model then Clusters 1 and 2 are independent. Switching events in Cluster 1 (for example *T1* changing regions) which affect waveforms in Cluster 1 (for example *n1*) won't affect waveforms in Cluster 2 unless they cause *T2* to change regions. However, consider what happens if *T2* is our **MOS** Level-l model biased into its ***saturation*** region. In that case *T2's* drain current will be a continuous function of of *T2's* gate voltage. In that case any changes to *n1's* waveform will immediately affect waveforms in Cluster 2 even if *T2* doesn't change regions.

Thus if *T2's* model includes a dependent source coupling its source and drain currents to its gate voltage, then any event that requires Mom to recompute the response of Cluster 1 (including its moments) also requires Mom to recompute the response of Cluster 2 (including ***its*** moments). This implies that not only must the moments be computed in lock step, but the kth moments of Cluster 1 must be computed ***before*** the kth moments of Cluster

2. This ***unidirectional*** coupling is represented by an arc between the clusters.

Another example of unidirectional coupling is illustrated by our bipolar model. Similar to the MOS example, the emitter and collector currents are dependent upon the base voltage. However, while the collector current depends upon the emitter voltage, the emitter current is independent of the collector voltage. This can be represented by an additional unidirectional coupling from the emitter to the collector (Figure 5.10). Thus we represent

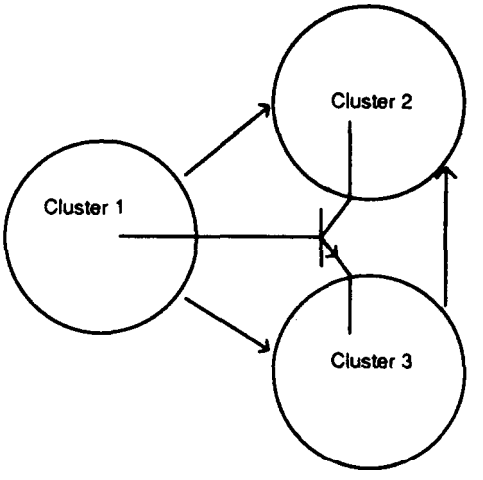


Figure 5.10: Multiple Couplings for Bipolar Transistor.

a bipolar transistor with three arcs.

Note that for this particular model the "channel" of the bipolar transistor (the **emitter**–collector path) is represented by a unidirectional arc rather than a bidirectional edge. Therefore the base, emitter, and collector nodes may **all** reside in different clusters. In general, if either y_{12} or y_{21} of a device model (See Equation 5.4) are equal to zero it can be treated as a unidirectional coupling between possibly different clusters rather than as a branch that couples the two nodes into the same cluster. Later we shall see this allows a potentially more efficient evaluation.

Coupled clusters are collected into a ***group*** and represented by a directed **graph.**[7] The moments of clusters in a group must be computed in lock step, and the arcs between clusters induce an ordering for the computation of each moment. If the directed graph is cycle free then the correct order for the **evaluation** of moments can be found via a ***topological sort*** of

[7]Note that while a cluster's graph is, by definition, connected, a group's directed graph may not be.

the graph.* In this case the group's moments can be computed in linear time. If the directed graph has cycles then the group has feedback and no topological sort exists. Feedback is handled using ***tearing*** as described in a following section.

Figure 5.11 depicts a topological sort of the directed graph of an ECL gate. Note that

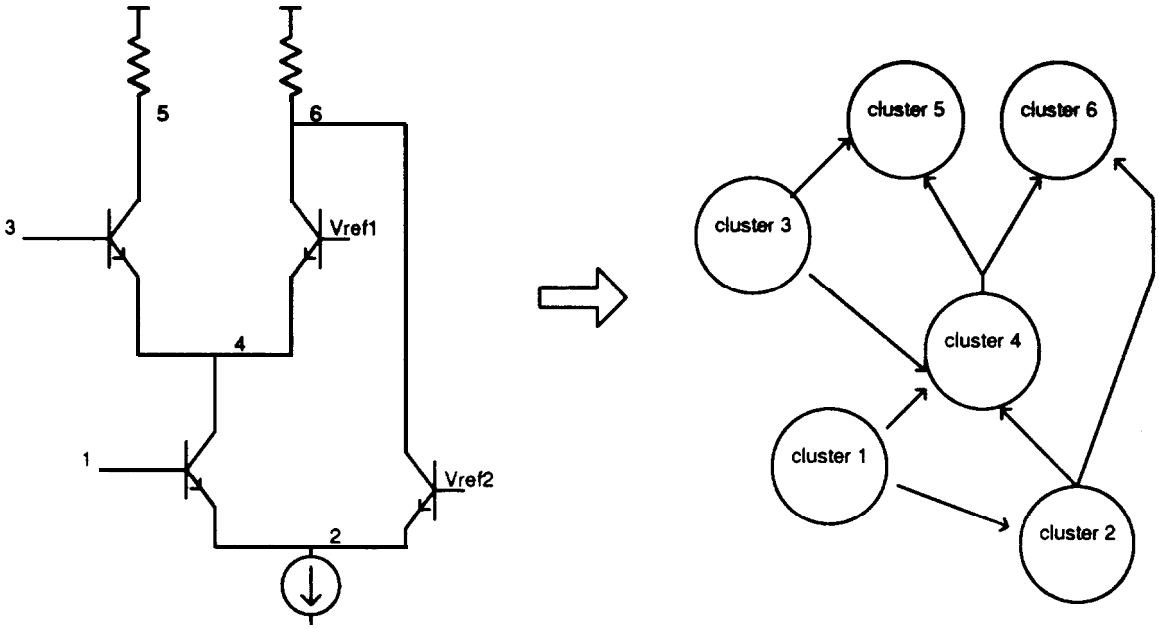


Figure 5.11: Topological Sort of ECL Gate.

since the bipolar model contains no edges, each node in the ECL gate is an independent cluster. The figure suggests that the directed graphs of most ECL current steering trees will, in fact, be cycle free. Thus it is possible to compute the moments of most ECL gates in linear time. In contrast, if the bipolar transistor were treated as an edge rather than an arc (Figure 5.12 shows what happens when the bipolar transistor model is a resistor.) then the resulting undirected graph would have a loop and it wouldn't be possible to compute its moments in linear time. This elimination of loops from the graphs of ECL gates was the primary motivation behind neglecting the output conductance of bipolar transistors.

[8] A topological sort can be performed by *depth first search* with complexity $O(n)$ [AHU85].

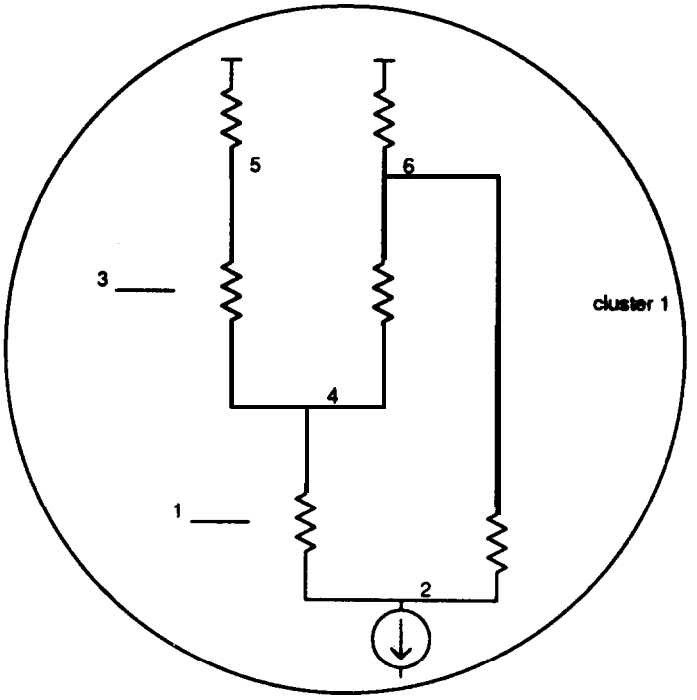


Figure 5.12: Resistor Model for Bipolar Transistor Results in Loops in ECL Gates

5.6 Relaxing Network Restrictions

In the preceding section a method for finding the DC solution (and hence the moments) of (possibly multiple coupled) trees of piecewise linear devices was described. That method has the nice property that its complexity grows linearly with the size of the circuit. However two restrictions were placed on the circuit: the edge graph must be loop free, and the cluster graph must be cycle free. Because of the judicious choice of transistor models these restrictions are seldom a problem. Most CMOS circuits yield leaky trees, and most ECL circuits yield directed acyclic graphs. Thus it is possible to compute the moments of the most common circuits in linear time. However, one occasionally encounters circuits that contain either loops or feedback. For those circuits the techniques described cannot be used directly. In this section we shall describe the use of ***tearing*** to solve those circuits. Although the complexity of tearing is superlinear, if the circuits that need to be tom are small, and if there aren't many of them, tearing can still yield an efficient solution technique.

One example of a circuit with feedback is the Schmitt trigger in Figure 5.13. The

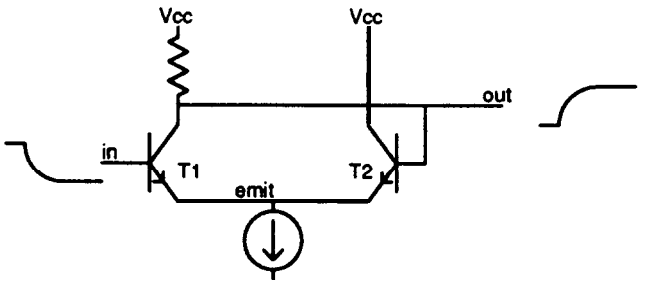


Figure 5.13: Schmitt Trigger

Schmitt trigger has two stable states. If ***in*** is high then ***T1*** is on and our is low. Conversely, if ***in*** is low then ***T2*** is on and ***out*** is high. In either state only one transistor is on. However, consider what happens when ***in*** falls from high to low. ***T1*** is initially the only transistor on but soon ***T2*** turns on as well. When it does, a positive feedback loop is established through both transistors that eventually causes ***T1*** to turn off. Thus during the transition a cycle exists in the directed graph. As another example, consider the diode decoder in Figure 5.14. When the decoder's inputs $x2$ - x0 are low, $i2 - i0$ receive current and $\overline{i2} - \overline{i0}$

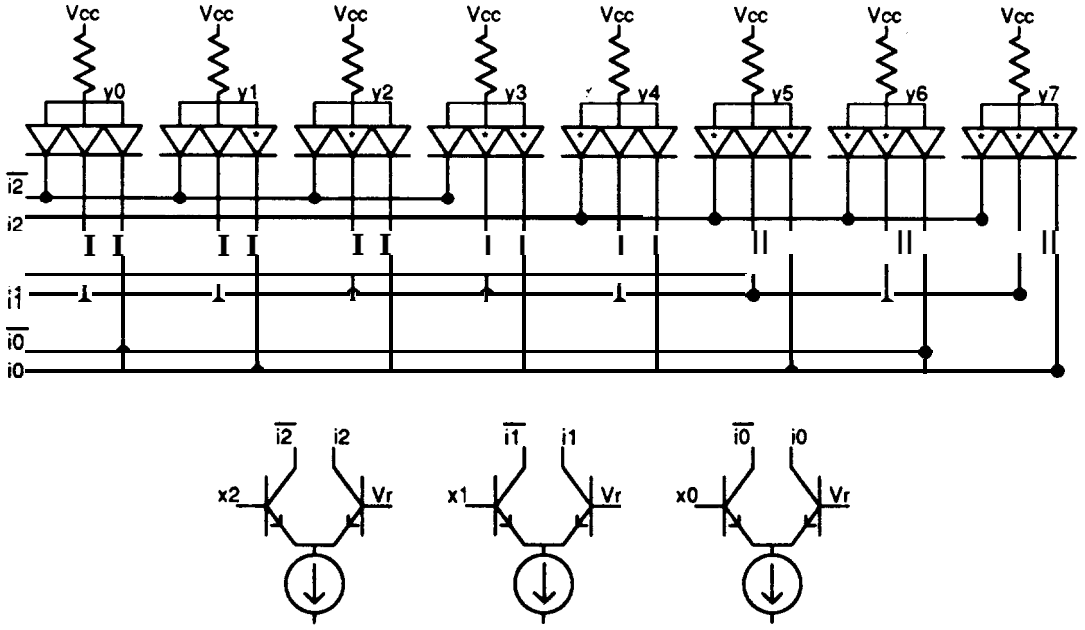


Figure 5.14: Diode Decoder.

don't receive current from the differential pairs. Only diodes connected to wires receiving

current (marked with an asterisk (*)) are on. Since *y0* is the only output whose diodes are all *off*, only it is high. All other outputs are pulled low by one or more *on diodes.* However, it is not trivial to determine the distribution of currents through the diode network. This is because the topology of the on diodes forms a mesh rather than a tree.

If the circuits containing loops and/or feedback are small and if there aren't many of them, it can be practical to handle *just* those circuits using a more general, albeit less efficient, extension of the original algorithm. In this section we will show that the circuit decomposition technique *known as rearing* can be used to handle such circuits.

5.6.1 Node and Branch Tearing

Circuit tearing was originally introduced by Kron[Kro39] who described the solution of large networks by 1) partitioning them into multiple subnetworks 2) solving the subnetworks independently and 3) combining the independent solutions to produce the overall solution. Two particularly interesting techniques have been devised for tearing systems of nodal equations: *branch tearing*[Wu76] and *node tearing*[SVCC77]. Branch tearing can be interpreted as[9] the insertion of independent current sources in series with "tom" branches in order to partition the network (Figure 5.15). In contrast, node tearing can

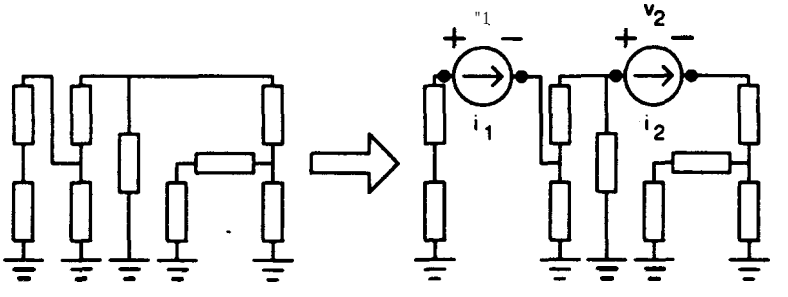


Figure 5.15: Circuit Partitioning via Branch Tearing.

be interpreted as the insertion of independent voltage sources between "tom" nodes and ground in order to partition the network (Figure 5.16). For each of our examples the insertion of two independent sources partitions the network into three subnetworks. The

[9] The intuition behind these interpretations of branch and node tearing are largely due to an insightful paper by Rohrer[Roh88]. However, our interpretation of node tearing differs from the one presented there.

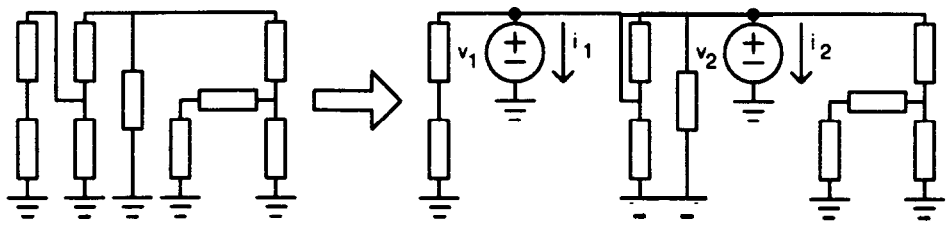


Figure 5.16: Circuit Partitioning via Node Tearing.

network is considered to be "partitioned" because each of the subnetworks may be solved independently once the values of the inserted sources are given.

The solution of the original network can be found using multiple solutions of the subnetworks. To illustrate, consider the case of branch tearing. First the inserted current sources, i_1 and i_2, are set to zero and the subnetworks are solved to find the voltages across the inserted sources v_1 and v_2. Denote the resulting voltage across the kth inserted source by v_{o_k}, that is the response due to sources that were part of the ***original*** network. Next, set all sources (original and inserted) to zero. Then for each of the inserted sources, set only that source (let's say it is the kth source) to some nonzero constant i_{a_k} and solve the subnetworks for the voltages across each of the inserted current sources. Denote the ratio of the voltage across the jth inserted source to i_{a_k} by the transfer resistance r_{jk}. Then by superposition, the total response due to the original sources and with arbitrary settings for the inserted sources is given by (assuming n inserted sources, and v_{t_j} is the total voltage across the jth source)

$$\begin{vmatrix} v_{t_1} \\ v_{t_2} \\ \\ v_{t_n} \end{vmatrix} = \begin{vmatrix} r_{11} & r_{12} & \cdots & r_{1n} \\ r_{21} & r_{22} & \cdots & r_{2n} \\ \vdots & \vdots & \vdots & \vdots \\ r_{n1} & r_{n2} & \cdots & r_{nn} \end{vmatrix} \begin{vmatrix} i_{a_1} \\ i_{a_2} \\ \vdots \\ i_{a_n} \end{vmatrix} + \begin{vmatrix} v_{o_1} \\ v_{o_2} \\ \\ v_{o_n} \end{vmatrix} \tag{5.15}$$

If we set $v_t = 0$ in the above equation and solve for i_a we get the actual currents flowing through the tom wires of the original circuit (ie before augmentation). Finally, if we solve the augmented circuit with those currents we get the DC solution of the rest of the original circuit.

The node tearing case is solved in a similar manner, except that instead of setting the values of the inserted current sources the values of the inserted voltages sources are set,

instead of measuring the voltages across the inserted current sources the currents through the inserted voltage sources are measured, and instead of forming a transfer resistance matrix a transfer conductance matrix is formed.

5.6.2 Non-Leaky Tree Topologies

Node and branch tearing were conceived with the objective of partitioning the network into multiple independent pieces. However, they can also be used to reduce a network to one that is more readily solved. For example, consider a circuit that is nearly a leaky tree (Figure 5.17). The network can be reduced to a leaky tree by finding a spanning tree for

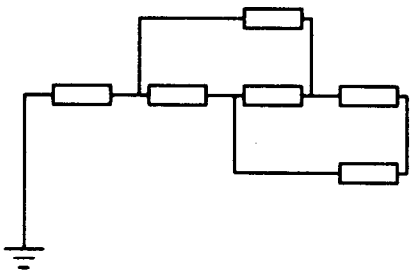


Figure 5.17: Circuit That is Nearly a Leaky Tree.

the network and then using branch tearing to tear out all edges not part of the spanning tree (Figure 5.18). The equivalence of the tom circuit to a leaky tree can be made more apparent

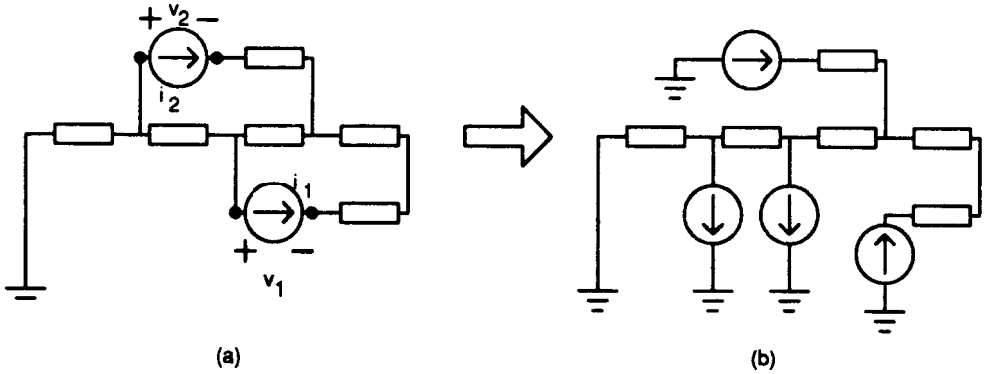


Figure 5.18: Branch Tearing of Nearly Leaky Tree.

by replicating the tearing sources. In a similar manner node tearing can be used to break

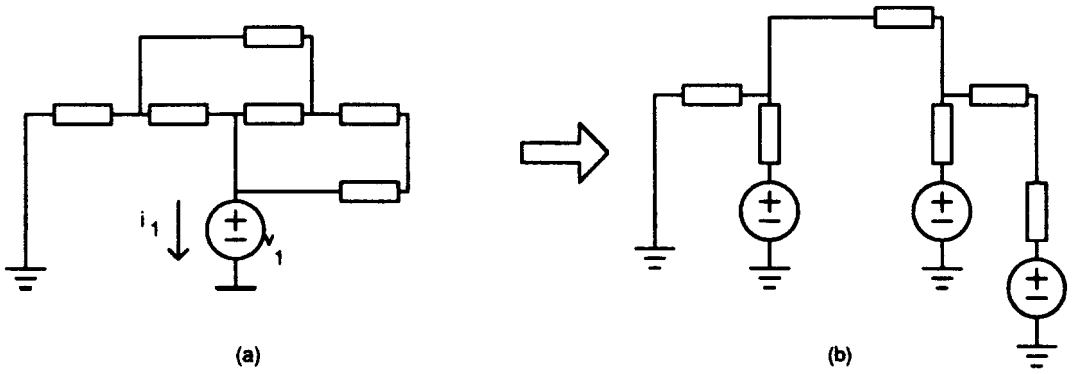


Figure 5.19: Node Tearing of Nearly Leaky Tree.

loops in the circuit (Figure 5.19). In either case, the tom circuit is a leaky tree and hence can be solved in linear time. Of course tearing doesn't really eliminate the work of solving the network. In order to compute the response of the original network from the solution of the tom network a transfer resistance or conductance matrix of dimensionality equal to the number of tom nodes or branches must be formulated and LU factored, a process that is generally superlinear. In general, tearing techniques are not guaranteed to reduce the amount of computation and can actually increase it[Wu76]. However, the experience from switch-level simulation has been that circuits are usually close approximations of trees and the number of tom branches or nodes is much smaller than the total number of branches or nodes in the network. If the number of tom branches or nodes is small and bounded then tearing can be used advantageously.

Chan and Karplus[CK89] and Pillage and Dutta[PD90] used brunch rearing to reduce the network to a tree. Ratzlaff et al[RGP91] *used* a ***circuit collapsing*** technique" which can be viewed as node tearing if one notes that the circuits that are collapsed are simply the partitioned subnetworks and the nodes that remain after collapsing are the tom nodes. The choice of tearing nodes is particularly advantageous in that the resulting tearing matrix is sparse, symmetric, and positive definite and therefore can be more efficiently LU factored than general circuit matrices.

We use a combination of tearing techniques. Branch tearing is used to reduce individual

[10]This was an improvement of a technique first explored by Stark and Horowitz for the solution of large power networks[SH90].

clusters to leaky trees.[11] The difference between our approach and that of Chan and Karplus[CK89] and Pillage and Dutta[PD90] is that we needn't tear branches to ground. Although this complicates the formulation of the tearing matrix (we can no longer simply traverse fundamental loops), the tearing matrix can be smaller and sparser. As demonstrated in [PD90], tearing branches to ground can severely reduce the sparsity of the tearing matrix.

Feedback is handled using an analogous procedure. Instead of tearing out edges to eliminate loops in a cluster's undirected graph, we tear out arcs in order to eliminate cycles from a group's directed graph (Figure 5.20 (a)). The only additional complication arises

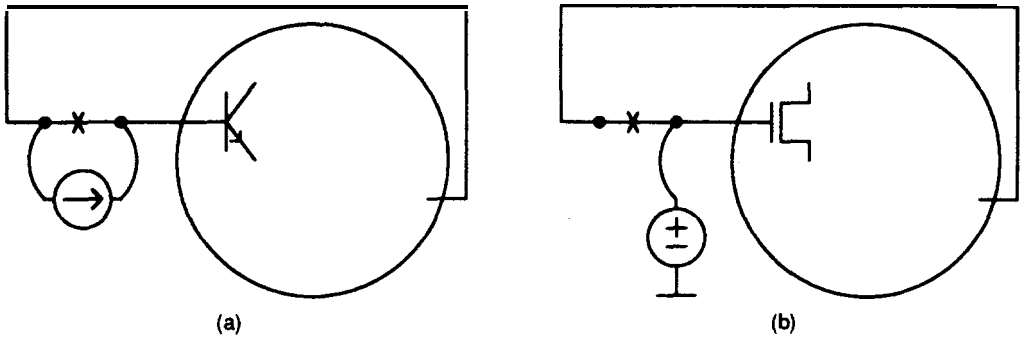


Figure 5.20: Tearing of feedback.

when it is not possible to replace the tom wire with a current source because such a current source would see an infinite impedance. This occurs, for example, when cluster inputs are only connected to MOS gates.[12] In this case we tear out the wire and drive the infinite impedance side with a grounded DC voltage source.

5.7 Partial Evaluation of Floating Capacitive Coupling

The procedures described up to this point can be used to compute the moments of arbitrary interconnections of piecewise linear devices. If the network is free of loops and feedback

[11] Branch tearing is actually suboptimal with respect to node tearing in the sense that it may require more tearing variables. As illustrated by our example, tearing a branch can open up at most one loop while tearing a node can open up several. A similar result was proved by Sangiovanni-Vincentelli et al.[SVCC77]. However, in the context of a switch-level simulator, the implementation of branch tearing is slightly simpler.

[12] In fact, presently all our bipolar models also have zero DC base current although this is not actually a restriction imposed by the simulator.

then the moments can be computed in linear time, otherwise the moments are computed with reduced efficiency. In either case, the moments are computed ***exactly*** (except for the numerical error introduced by executing the algorithms on a real computer).

However, under certain circumstances it may be necessary to give up trying to compute the moments exactly.[13] In particular, the inclusion of floating capacitors into device models introduces an efficiency problem. To illustrate, examine the cascade of three CMOS inverters shown in Figure 5.21. Assume that the circuit has settled and consider the

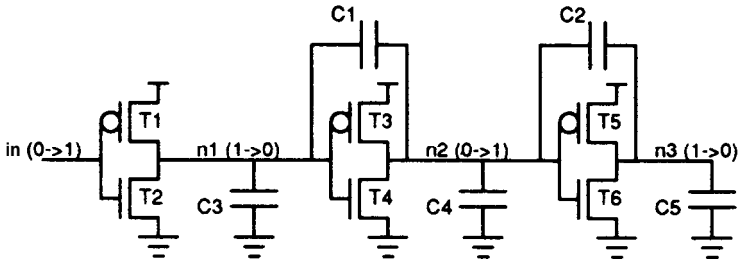


Figure 5.21: Clusters Coupled by Floating Capacitors.

sequence of events that follows ***T2*** turning on. First note that were it not for the floating capacitors ***C1*** and C2, nodes ***n1***, ***n2***, and ***n3*** would reside in three different groups. This is because ***T3 - T6*** are initially biased into either the ***off*** or ***linear*** regions which exhibit zero gain from the gate to the source and drain. However ***C1*** and C2 couple all three nodes into the same group. Although the efficiency of moment computation is still $0(n)$ (no loops or cycles can be introduced by floating capacitors) we must recompute the response of many more nodes than would seem necessary. Intuitively we would expect that the effect of 7'2 switching upon n2 (coupled through 1 level of floating capacitors) to be quite small relative to the logic swing, and the effect on ***n3*** (coupled through 2 levels of floating capacitors) to be negligible. Otherwise the ***logic gate*** abstraction would never have been found to be useful. However, with our present moment computation algorithm, floating capacitors can potentially couple all nodes in a circuit thereby eliminating the ability to take advantage of circuit latency.

A simple solution is to allow a group to expand through only a limited number of levels of floating capacitors. Nodes sufficiently distant (in terms of the number of levels of

[13] This isn't so bad. After all the waveforms that are generated from the moments are just approximations.

floating capacitors) from the switching event are presumed to be essentially unaffected by the event. For our example, if the maximum number of levels is set to **1**, then only ***n2*** would be brought into the same group as ***n1***. However, when the expansion process is truncated there may be some capacitors that only have one terminal belonging to the group. In that case the terminal outside the group is considered to be driven by a (possibly time varying) voltage source that has the waveform presently on the node. That waveform is otherwise undisturbed by the event. For our example, C2 is treated as if its right terminal were driven by a voltage source having ***n3***'s present waveform. Node ***n3*** retains whatever waveform it had prior to ***T2*** switching.

Our initial experience is that this procedure works well. Figure 5.22 plots the response of a 9 stage ECL ring oscillator when Mom expands groups through various numbers of levels of floating capacitors. In each case the output of Mom using the Level-2 bipolar model (which includes parasitic floating capacitors and resistors) is compared with that of SPICE using an identical piecewise linear model. Table 5.1 gives the switching delay error for each case. The figure and table show that the largest change in accuracy is obtained by

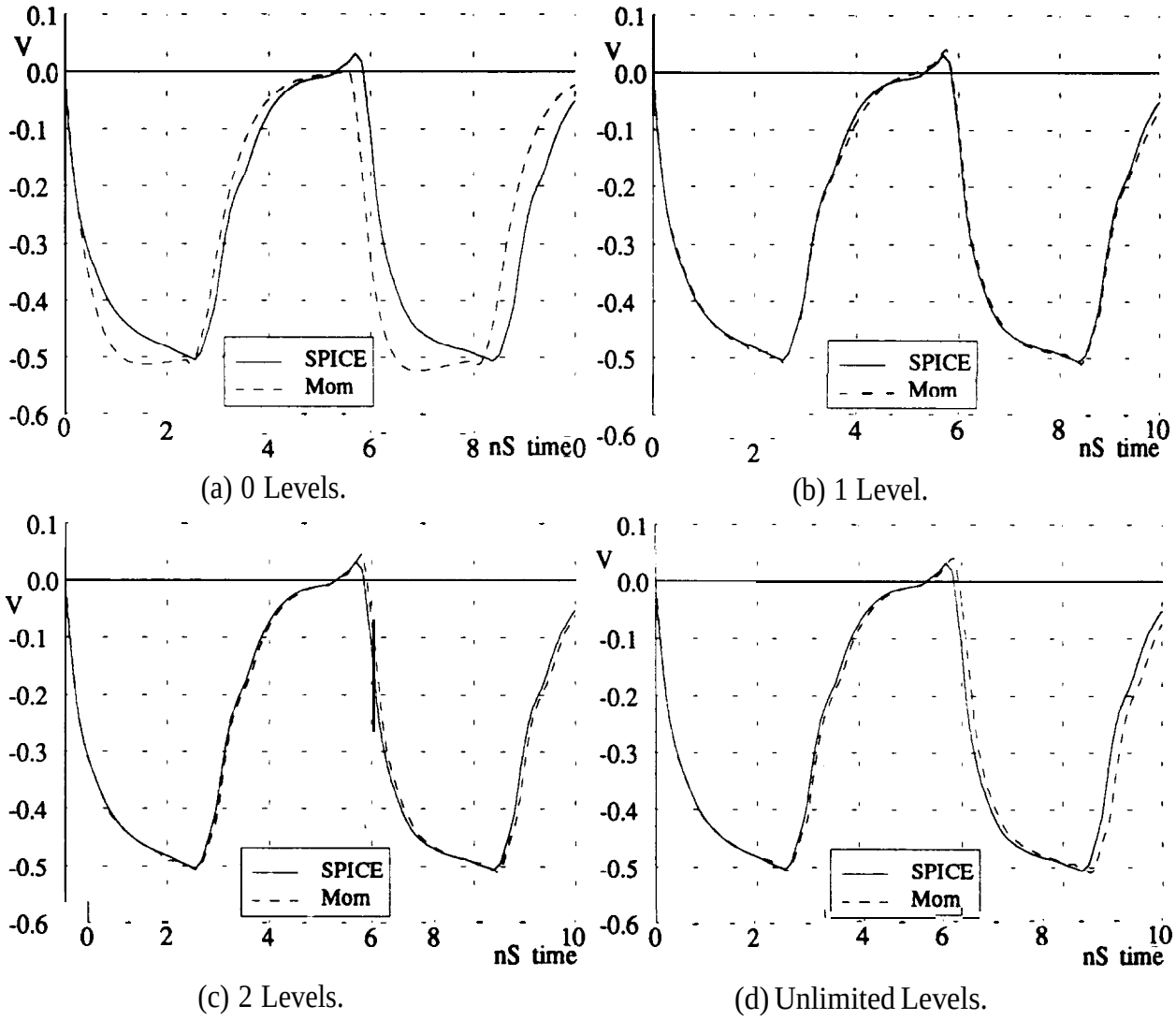
capacitor levels	0	1	2	unlimited
% switching delay error	3.5	0.3	0.7	1.8
run time (seconds)	2.3	6.2	10.0	19.5
nodes per group	7.9	24.6	54.9	108
# waveform computations	869	27	10	4127	7960

Table 5.1: Decrease in Efficiency with Increasing Capacitor Levels.

expanding the group through just 1 level of floating capacitors.[14]

The table also shows the super-linear growth of execution time with increasing numbers of levels. This super-linear growth can be explained. First note that the number of events doesn't change. Thus the execution time is roughly proportional to the number of waveform computations and hence the average size of a group. Also note that because of the nodes and

[14] Mom and SPICE piecewise linear simulations appear to be converging to slightly different waveforms. The most obvious source of error is the waveform approximation. However, there are other possible sources of error. For practical reasons, it is not actually possible for Mom and SPICE to simulate *exactly* identical piecewise linear models. Small parasitic conductances and capacitances must be added to SPICE's piecewise linear models in order to aid convergence of the numerical integration. Hysteresis (+/- 1mV) must be added to Mom's piecewise linear models to enhance the stability of event scheduling. Finally, experimentation with simple circuits that could be solved analytically indicate that SPICE may generate errors as large as 0.3%.



(a) 0 Levels.

(b) 1 Level.

(c) 2 Levels.

(d) Unlimited Levels.

Figure 5.22: Limited Levels of Floating Capacitors.

coupling introduced by the bipolar **parasitics** the fine grain of the interconnection network resembles a mesh. If the number of levels is modeled by the radius of a circle, and the number of nodes in a group by the area of the circle then one would expect a roughly quadratic initial growth of the group size with the number of levels.

Finally, note that for our **small** example a maximum **speedup** of only 8.5: 1 is achieved. This seems consistent with the expectation that at any instant in time **only** one or two of the nine stages of the ring is actively switching. However, larger circuits often exhibit much more latency, and one might expect correspondingly greater speedups for them.

5.8 Summary

Moment computation can dominate the cost of waveform estimation. To compute moments, switch-level simulators use ***RC tree analysis*** which is efficient because it takes advantage of the tree-like topology of most circuits. We generalize RC tree analysis along two dimensions. First, tree analysis is extended to apply when transistor models are generalized from resistors to piecewise linear devices. This generalization retains the efficiency of RC tree analysis for the transistor-capacitor trees found in MOS circuits and the current steering trees found in ECL circuits. Second, tearing is used to handle non-tree topologies and feedback. The advantage of combining a tree analysis with tearing is that most circuits are trees and hence can be analyzed efficiently. The cost of analyzing more general non-tree topologies is paid only when it is needed.

The addition of floating capacitors to device models can degrade simulation efficiency because floating capacitors can potentially couple together all nodes in the circuit thereby eliminating the ability of the simulator to take advantage of circuit latency. However, in digital circuits there is rarely any significant coupling through multiple levels of floating capacitors. Thus, repartitioning of the circuit can be achieved by ignoring coupling through all but a limited number of levels of floating capacitors.

Chapter 6

Detecting Region Changes

The previous chapters showed how ***moments matching*** can be used to compute the response of circuits containing piecewise linear devices. However, moments matching only deals with ***linear*** circuits. That is, the response is computed assuming that all piecewise linear devices remain in ***their present*** regions of linearity. In reality devices may change regions of linearity in the middle of the transient. At the instant any device in the cluster changes region, the simulator must stop and recompute the response with that device relinearized in its new region.

This chapter describes the techniques used by Mom to determine if and when piecewise linear devices change regions of linearity. It turns out that this task is considerably more difficult for Mom than for Rsim. It is important to study this problem because, as we shall see in a later chapter, this computation can dominate the run time of the simulation.

6.1 Mom vs Rsim

The incorporation of more general piecewise linear transistor models complicates the detection of when devices change regions of linearity. In Rsim's case, a transistor switches whenever its gate voltage crosses the switching threshold. Since the step response of RC trees is approximated by a single exponential:

$$V_g(t) = 5.0e^{-t/\tau} \tag{6.1}$$

(where τ is given by the first moment), in order to find out whether a transistor switches Rsim only needs to solve the following equation for t:

$$V_g(t) - 2.5 = 0 \tag{6.2}$$

where the first term on the left hand side is the waveform on the gate terminal and the second term is the negative of the transistor's switching threshold The solution for t can be found explicitly:

$$5.0e^{-t/\tau} - 2.5 = 0 \tag{6.3}$$

$$t = -\tau \ln 1/2. \tag{6.4}$$

Thus Rsim only needs to multiply the first moment by a constant in order to obtain the switching time.

More general piecewise linear models introduce a few complications. They may have several regions of linearity and it is necessary to check whether the model will change from the present region to ***any*** adjacent region. As described in Chapter 3 the boundary separating the present region from an adjacent region is a ***hyperplane*** defined by a linear equation of the form:

$$a_0 + a_1 v_1(t) + a_2 v_2(t) + a_3 v_3(t) = 0. \tag{6.5}$$

where $v_i(t)$ is the voltage on the ith terminal, and the a; are constant coefficients that determine the location of the hyperplane. Since Mom approximates voltage waveforms using sums of **exponentials**:

$$v_i(t) = b_{i0} + b_{i1}e^{\sigma_1 t} + b_{i2}e^{\sigma_2 t} + b_{i3}e^{\sigma_3 t} \tag{6.6}$$

the time at which the transistor crosses that boundary can be obtained by substituting the terminal voltages (Equation (6.6)) into the equation for the hyperplane (Equation (6.5)) and finding the root of the resulting equation:

$$k_0 + k_1 e^{\sigma_1 t} + k_2 e^{\sigma_2 t} + \ldots + k_9 e^{\sigma_9 t} = 0 \tag{6.7}$$

The left hand side of Equation (6.7) defines a time varying waveform which will be referred to as the ***boundary waveform.***

However, we are not interested in finding any root. Instead, Mom must start at the current time and search forward to find the ***first*** root. Since the current time can always be normalized to zero, Mom must find the smallest, positive root of an equation of the form of (6.7). If no positive root is found then that boundary is never crossed. Mom must then check all remaining hyperplanes bounding the current region.

Thus the task of rescheduling a device eventually reduces to (possibly several instances of) the one dimensional root finding problem. The robust and efficient solution of that problem is the focus of the rest of this chapter.

6.2 Overview of Root Finding

Although many general purpose root finding techniques have been described in the literature (for example, Bisection, Newton Raphson, ***regula falsi***, and **Brent's[Atk78]**), there exists no general technique which 1) offers any control over which root is found, 2) provides any guarantee of convergence or 3) can detect when no root exists. Instead, in order to guarantee convergence to a desired root it is necessary to incorporate problem specific knowledge. Mom takes particular advantage of the fact that the left hand side of Equation (6.7) is a weighted sum of exponentials. Different techniques are used depending upon the number of **exponentials.**

The procedure used by Mom for finding roots involves 3 steps:

1. Moments matching is used to produce a low order approximation of the boundary waveform. This is done because root finding is most difficult when there are a large number of poles. Moments matching is used to reduce the number of poles to three or fewer.

2. Once the number of poles has been limited to three, the problem can be broken down into a number of special cases. For each different pole configuration the roots are found by taking advantage of special characteristics of that particular configuration.

3. Although the waveform approximation in the step 1. makes it easier to find roots, it can sometimes introduce small errors that lead to consistency problems with the

simulation. Therefore after the root is found, root polishing is used to eliminate the errors introduced by the waveform approximation.

Each of these steps will be individually addressed in the following three sections.

6.3 Order Reduction

To simplify the task of finding roots, the boundary waveform (which could initially contain as many as 9 poles) is approximated by a waveform having three or fewer poles. This is done by forming the moments of the boundary waveform from the linear superposition of the moments of the terminal voltages and computing a waveform approximation from the resulting moments. The result is a reduced order waveform'

$$b(t) = k_0 + k_1 e^{\sigma_1 t} + k_2 e^{\sigma_2 t} + k_3 e^{\sigma_3 t} \tag{6.8}$$

The poles may ***be*** either ***simple*** (both k_i and σ_i real) or may occur in complex conjugate **pairs (two** poles with σ_i and σ_j complex such that $k_i = k_j^*$ and $\sigma_i = \sigma_j^*$).

6.4 Finding Roots

Once the number have poles has been reduced to three or fewer, Mom only needs to deal with a limited number of configurations:

- 1 simple pole
- 2 simple poles
- 3 simple poles
- 1 complex conjugate pole pair
- 1 simple pole plus 1 complex conjugate pair.

[1] All poles are assumed to be stable. Appendix E shows how to handle unstable poles.

In general, the difficulty of finding roots is dependent upon the pole configuration. In order to maximize efficiency a different technique is used for each different configuration. Occasionally a root can be found explicitly, although most of the time Mom must use some combination of root bracketing and Newton-Raphson iteration. The following subsections describe each of the different techniques.

6.4.1 One Pole

The simplest case is if there is a single pole. In that case b(t) takes the form:

$$b(t) = k_0 + k_1 e^{\sigma_1 t} \tag{6.9}$$

and b(t) = 0 can be explicitly solved for t to yield:

$$t_{root} = \frac{1}{\sigma_1} \ln -\frac{k_0}{k_1} \tag{6.10}$$

Note that if $k_0/k_1 > 0$ then no root exists. Also, if $t_{root} < 0$ then the root occurred in the past and is of no interest.

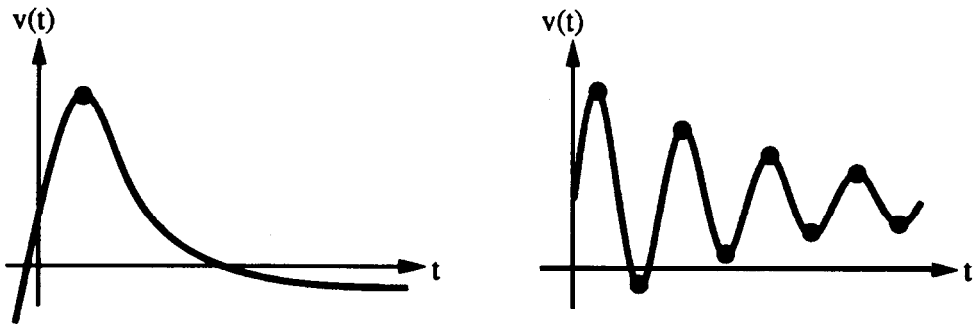
6.4.2 Two Poles

The roots of two pole responses are found by ***bracketing*** the root before using a general purpose algorithm. ***Bracketing*** consists of locating an interval, $t \in [t_0, t_1]$, that contains exactly one root, the desired root. Once the root has been bracketed several general purpose algorithms (for example Bisection) are guaranteed to converge to it.

To bracket the root, the stationary points of $b(t)$ (that is the points at which the derivative is zero) are considered to partition the function into a number of disjoint segments (Figure 6.1). Because the derivative is continuous, it cannot change sign without passing through zero (a stationary point) and hence these segments must be monotonically non-increasing or monotonically non-decreasing. Thus if the sign of the function changes from one end of the segment to the other then that segment must contain exactly one root.

Fortunately, the stationary points of two pole responses may be obtained explicitly. If the response consists of two simple poles (Figure 6.1 (a)):

$$b(t) = k_0 + k_1 e^{\sigma_1 t} + k_2 e^{\sigma_2 t}, \tag{6.11}$$



(a) Simple Poles (b) Complex Conjugate Pair

Figure 6.1: Stationary Points

then the stationary point (if it exists) can be found by setting the derivative of $b(t)$ to zero:

$$t_{stationary} = -\frac{1}{\sigma_1 - \sigma_2} \ln \left(\frac{k_1 \sigma_1}{k_2 \sigma_2} \right) \tag{6.12}$$

Since there is at most one stationary point the waveform may consist of at most two monotonic segments. A comparison of the signs of $b(t_{stationary})$, $b(t = -\infty)$, and $b(t = +\infty)$ reveals whether or not either segment contains a root.

If the response consists of a complex conjugate pole pair, then $b(t)$ can be expressed:

$$b(t) = k_0 + Me^{\sigma t} \cos(wt + \phi), \tag{6.13}$$

which has the stationary points:

$$t_{stationary} = \frac{1}{\omega} \left(\tan^{-1} \frac{\sigma}{\omega} - \phi + \pi k \right) \tag{6.14}$$

for $k = \ldots -1, 0, 1, \ldots$. Although there are an infinite number of stationary points, it is only necessary to examine the first couple of segments past the origin because the waveform decays with time (Figure 6.1 (b)).

6.4.3 Three Poles

Two cases can occur for three pole waveforms. Either all three poles are simple poles, or there is one simple pole and a complex conjugate pair.

Three Simple Poles

The case of three simple poles

$$b(t) = k_0 + k_1 e^{-\sigma_1 t} + k_2 e^{-\sigma_2 t} + k_3 e^{-\sigma_3 t} \tag{6.15}$$

($\sigma_i \in \Re, \sigma_i > 0$) is handled by conceptually transforming the multipole exponential expression into a polynomial. This allows us to use results from the area of polynomial rootfinding.

Assuming the σ_i are rational, they can be expressed as the ratio of integers $\sigma_i = n_i/\tau_{lcd}$ where τ_{lcd} is their least common denominator. Then the transformation of variables:

$$y = e^{-t/\tau_{lcd}} \tag{6.16}$$

results in a polynomial in y:

$$p(y) = k_0 + k_1 y^{n_1} + k_2 y^{n_2} + k_3 y^{n_3}. \tag{6.17}$$

Because y varies from $1 \rightarrow 0$ as t varies from $0 \rightarrow +\infty$, we are interested in the largest root of $p(y)$ in the region $y \in [0, 1]$.

However, it is not practical to simply track down all the roots because the powers to which y is raised (and hence the number of roots) may be extremely large. Instead, theorems that predict the number of roots that lie along segments of the real axis allow us to narrow in on the single *real* root of interest. In particular we employ Decartes' rule of signs[Hou70]:

Definition 1 ***Given a sequence of real numbers*** $a_0, a_1, \ldots a_n$ ***a variation in sign occurs if*** $a_i a_{i+1} < 0$ ***or if*** $a_i a_{i+j} < 0$ ***and*** $a_{i+1} = a_{i+2} = \ldots = a_{i+j-1} = 0$.

Theorem 1 (Decartes' Rule of Signs) ***Let*** $p(x)$ ***be a polynomial*** $p(x) = a_0 + a_1 x + a_2 x^2 + \ldots + a_n x^n$. ***If the sequence of its*** coefficients ***has V variations in sign and the number of roots on the positive real axis is*** r ***then*** $V - r$ ***is a nonnegative even integer.***

Note that this theorem only utilizes the signs of the coefficients of the polynomial and doesn't actually require the computation of the exponents. After sorting the poles in order of increasing frequency magnitudes (left to right, starting from the DC "pole", k_0) it is only necessary to scan the signs of the coefficients.

Often, Decartes' rule of signs provides sufficient information to bracket the root. The simplest example is if there is no variation in sign. In that case we quickly conclude there is no root. For a more complex example, consider the case when the sequence of signs is $--++$. Since there is one variation in sign, there must be exactly one root. Additionally, because the coefficient of the DC pole is negative the final value is negative. Thus if $b(t=0) > 0$ then the interval $t \in [0, +\infty]$ brackets the root. Otherwise $t \in [-\infty, 0]$ brackets the root and the root is of no interest. There are 16 possible combinations of the 4 signs. In 13 of those 16 cases the root can be bracketed by merely inspecting the signs of the coefficients.

When the signs of the coefficients alone don't provide enough information, a generalization of the technique used for two-pole waveforms is employed. That is, the stationary points are used to partition the function into disjoint monotonic segments. The problem of finding stationary points of b(t) is equivalent to seeking the roots of its derivative, b'(t):

$$6\,\dot{b}(t) = -k_1\sigma_1 e^{-\sigma_1 t} - k_2\sigma_2 e^{-\sigma_2 t} - k_3\sigma_3 e^{-\sigma_3 t} \tag{6.18}$$

$$= e^{-\sigma_1 t}(-k_1\sigma_1 - k_2\sigma_2 e^{-(\sigma_2-\sigma_1)t} - k_3\sigma_3 e^{-(\sigma_3-\sigma_1)t}) \tag{6.19}$$

$$= e^{-\sigma_1 t} w(t) \tag{6.20}$$

where w(t) is given by:

$$w(t) = -k_1\sigma_1 - k_2\sigma_2 e^{-(\sigma_2-\sigma_1)t} - k_3\sigma_3 e^{-(\sigma_3-\sigma_1)t} \tag{6.21}$$

Because $e^{-\sigma_1 t} \neq 0$, w(t) has the same roots as b'(t) but has one fewer poles. Thus, the two pole techniques described above can be used to find the roots of u,(t). Since these roots are also the stationary points of b(t) they can be used to bracket the smallest positive root, after which a general purpose rootfinding algorithm can be used to converge to it. Note that this technique can be applied recursively to waveforms consisting of larger numbers of simple poles.

Simple Pole Plus Complex Conjugate Pair

The most difficult case is when the waveform consists of a simple pole plus a complex conjugate pair.

$$b(t) = k_0 + k_1 e^{\sigma_1 t} + M e^{\sigma_2 t} \cos(wt + \phi), \tag{6.22}$$

In this case the smallest positive root is bracketed by constructing a piecewise quadratic approximation to the response that is guaranteed not to deviate from the true waveform by more than an error voltage. The reasoning is that a change of region that erroneously occurs because of a small voltage perturbation (~ 1 mV) will probably be short lived and have little effect on the global behavior of the circuit.

The piecewise quadratic segments are constructed one at a time starting from $t = 0$ (Figure 6.2). After each segment is constructed, its end points and extremurn are checked

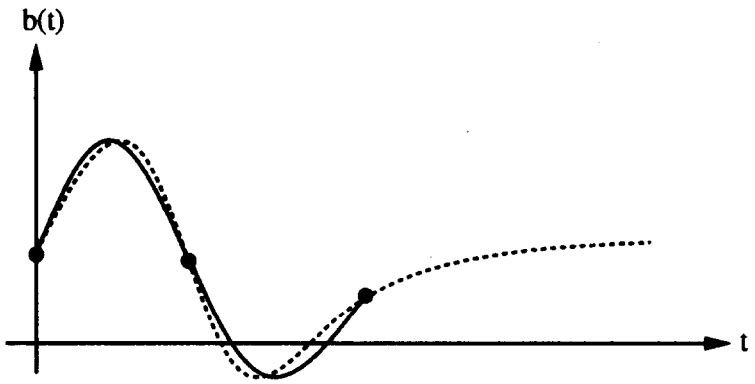


Figure 6.2: Rootfinding using Piecewise Quadratic Segments

to see if the segment crosses zero. If so, the desired root can be bracketed. When a segment is constructed an attempt is made to maximize its length (that is the time between its end points) subject to the constraint of keeping the maximum deviation bounded. Thus, as the waveform decays with time the lengths of successive segments tend to increase. Finally, the maximum possible negative contribution of each of the terms in Equation (6.22) is monitored so that the search can be terminated when it is clear that the remaining waveform cannot cross zero.

This technique is more general than the preceding techniques in that it can be applied to any combination of simple and complex conjugate poles. In fact, the technique was initially used to handle all waveforms. Unfortunately, experience with the technique revealed efficiency problems. Sometimes time must be advanced very far into the future and large numbers of segments must be constructed in order to conclude that there is

no root. Unfortunately, this work often turns out to be unnecessary because some other device switches very early on. Because switching events for different devices are sought independently we are performing what amounts to a ***depth first*** search for the segment bracketing the first switching event. Unfortunately, this particular root finding technique would benefit from a ***breadth first*** search organization.

Another problem is that large numbers of segments are needed to approximate waveforms with large voltage swings. In Chapter 4 it was observed that occasionally waveform approximations have voltage swings that far exceed what is physically possible. In those cases only an infinitesimal initial portion of the waveform approximation is used before some device switches. Unfortunately, the root finding algorithm doesn't take **this** into account and searches the entire waveform for roots. Because the waveform is so large, this entails searching through an excessive number of segments.

6.5 Root Polishing

The previous section showed how to find the roots of waveforms consisting of three or fewer poles. However, remember that these low order waveforms ***are*** only ***approximations*** of the original boundary waveform in Equation 6.7. In fact, the roots produced by this procedure may differ slightly from the roots of the actual boundary waveform. Unfortunately even slight differences can lead to inconsistencies in the event driven simulation algorithm. For example, suppose that because of an approximation error a diode is scheduled to turn on slightly prematurely. When that event fires the voltage across the diode will be insufficient to turn it on. If the simulator fails to check this and turns the diode on anyway it will recompute the response of the cluster only to find that the diode's next event will be to turn off immediately! To avoid this unnecessary work the simulator must check that a device's terminal voltages are consistent with the event. If an inconsistency is detected the event should be discarded and the device rescheduled. We refer to the discarded events as ***spurious*** events. In this section we will show that ***roorpofishing*** and ***hysteresis can be used*** to eliminate spurious events.

By checking each event for consistency the simulator eliminates the biggest expense associated with spurious events, that is the needless recomputation of a group's response.

However, it doesn't avoid the cost of rescheduling the device. Therefore, a number of steps were taken to to reduce the probability of spurious events. First $+/-1mV$ of hysteresis was introduced into every switching decision. For example, if a diode is off then it isn't turned on until $V_{diode} > .801$ volts (assuming a nominal threshold of .800 volts) and if it is on it isn't turned off until $V_{diode} < .799$ volts. Surprisingly, this was not sufficient to completely eliminate spurious events. In fact, for some simulations as many as 50% of the events that fired had to be discarded and rescheduled. When statistics about the voltages of the actual boundary waveform at the estimated root times were collected (Table 6.1) it

100uV < Verr < 1mV	*24%*
1mV < Verr < 10mV	*58%*
10mV < Verr < 100mV	9%
100mV < Verr < 1V	*9%*

Table 6.1: Voltage Error at Estimated Root of Boundary Waveform.

became apparent that errors in the switching time often led to scheduling devices to turn on when their terminal voltages were more than 1mV short of the switching thresholds. In fact in 9% of the cases the voltages were more than 100mV short of the switching thresholds. Similarly, statistics about the difference between the estimated switching time and the actual switching time (Table 6.2) revealed that some events were being scheduled

0 < Terr < 1ps	85%
1ps < Terr < 10ps	9%
10ps < Terr < 100ps	6%

Table 6.2: Time Error of Boundary Waveform Root Estimate.

as much as 100ps too early.

To eliminate the errors in the switching time estimate, ***rootpolishing*** is employed. That is the root of the reduced order waveform serves as the starting point for a few additional Newton-Raphson steps using the real boundary waveform. Table 6.3 shows that typically only a couple of iterations are needed to polish a root. It was found that root polishing completely eliminated spurious events.

number of iterations	fraction of total number of roots
0	70.7%
1	23.2%
2	5.9%
6	0.1%

Table 6.3: Number of Iterations for Root Polishing.

6.6 Measurements

From the description of the algorithms it is apparent that the amount of time needed to find roots increases with the number of poles in the waveform. Table 6.4 shows the average

Pole Configuration	Simpl	Simp2	Conj2	Simp3	Conj3
cost	54	490	350	790	1900

Table 6.4: Average Number of Cycles for Root Finding.

number of machine cycles[2] spent finding roots for each pole configuration. "Simpl" is one simple pole, "Simp2" two simple poles, "Conj2" a complex conjugate pole pair, "Simp3" three simple poles, and "Conj3" a complex conjugate pole pair combined with a simple pole. The data indicate that execution time grows somewhat faster than linearly with the number of poles, although the complex conjugate / simple pole combination stands out as being particularly inefficient. Thus the cost of rescheduling devices can be expected to rise as the complexity of waveforms increases.

The simulator was instrumented to print out the distribution of pole configurations for a number of circuits (Table 6.5). (For descriptions of the circuits see Table 4.2.) The table reflects the previously noted increase in the number of poles for circuits and models of increasing complexity. When there are two poles they are most likely to be simple poles whereas when there are three poles they are most likely to include a complex conjugate pair. Unfortunately, for the circuits employing floating capacitors, the least

[2]These numbers exclude the cost of boundary waveform approximation and root polishing. Estimates of machine cycles are for the MIPS R2000 CPU and were estimated using the *pixie* execution profiling tool created by MIPS Computer Systems, Incorporated.

Pole Configuration	%Simp1	%Simp2	%Conj2	%Simp3	%Conj3
CMOS0 Inverter Ring	100.0	0	0	0	0
CMOS0 NAND Ring	98.1	1.9	0	0	0
CMOS 1 Inverter Ring	21.4	76.2	1.2	1.2	0
CMOS1 NAND Ring	18.6	80.7	0.6	0.1	0
BJT0 ECL Ring 10%	25.0	24.6	13.1	17.0	20.2
BJT0 ECL Ring 20%	37.5	37.5	25.0	0	0
BJT1 ECL Ring	12.2	11.1	8.2	22.0	46.4
BJT2 ECL Ring capLevels=0	0	6.1	35.0	0	37.8
BIT2 ECL Ring capLevels=1	2.6	15.8	2.7	21.1	57.8
BJT2 ECL Ring capLevels=2	3.8	15.3	1.6	16.8	62.2
BJT2 ECL Ring capLevels=∞	5.6	11.2	7.3	24.8	51.1

Table 6.5: Pole Configurations for Root Finding.

efficient configuration, conj3, is the most common.

6.7 Summary

The incorporation of more general piecewise linear devices complicates the detection of when devices change regions because their regions may be bounded by multiple hyperplanes, those hyperplanes may depend on multiple terminal voltages, and the terminal voltages may have multiple poles. Consequently, the task of rescheduling devices is significantly more expensive for Mom than for Rsim.

The time of the soonest region change of a device is found by computing the smallest, *positive* root of all of its *boundary waveforms*. However, no general purpose root finding technique exists which provides any guarantees about convergence for all problems. Therefore it is necessary to take advantage of special characteristics of the sums of exponentials. The procedure involves three steps. First, the problem is simplified by finding an approximation of the boundary waveform that has three or fewer poles. Then, depending upon the particular pole configuration, the root of that approximation is found either explicitly or by using some combination of root bracketing and Newton-Raphson iteration. Finally, small errors introduced by the waveform approximation are eliminated using *root polishing.*

The algorithms for detecting region changes have been instrumented. Not surprisingly,

measurements made on a number of benchmark circuits indicate that the complexity of boundary waveforms also grows with increased model complexity and decreased error tolerances. In turn the cost of finding roots increases with the complexity of the boundary waveforms. In fact, it can take 40 times as much CPU time to find the roots of a third order boundary waveform as a first order boundary waveform. In Chapter 7 we shall see that one consequence of this is that the task of rescheduling &vices can dominate the overall execution time of the simulation.

Chapter 7

Evaluation

The objective behind producing Mom was to create a simulator that extended the accuracy and flexibility of existing MOS and bipolar switch level simulators while simultaneously preserving much of their efficiency for the simple cases. This chapter evaluates the extent to which we have achieved those goals. It begins by examining the application of Mom to a number of CMOS, ECL and BiCMOS circuits that appear to be just beyond the capabilities of existing switch-level simulators. Those simulations show that the additional flexibility is obtained with significant speedups over the circuit simulator SPICE-3d2. Then the performance of Mom is compared with that of the existing switch-level simulators, Irsim and Bisim,[1] on circuits that can be simulated at the switch-level. The benchmarks show that when the simplest switch-level models are used Mom is able to achieve switch-level accuracies with only a moderate degradation in performance compared to dedicated switch-level simulators. However, the benchmarks also reveal that Mom's efficiency degrades precipitously with increasing model complexity. Further measurements reveal that the causes of this degradation appear to be fundamental to this approach to simulation. It appears that this approach loses its speed advantage for those cases where the full accuracy and flexibility of circuit simulation are desired.

[1] SPICE–3d2 is a derivative of the circuit simulator SPICE[Nag75], *Irsim*[SH89] is a derivative of the MOS switch level simulator *Rsim*[Ter83], and Bisim is an ECL switch-level simulator[KAHS88]. Although Irsim is an *incremental* simulator, its incremental capabilities aren't used here and don't adversely affect the efficiency of normal simulation.

7.1 Extending Switch-Level Simulation

In this section the use of Mom is demonstrated on a number of small MOS, ECL, and BiCMOS circuits that can't be simulated by Irsim or Bisim. It is shown that the more general transistor models and circuit analysis techniques allow Mom to handle these "difficult" circuits. Even for these circuits a large fraction of transistors can be modeled using switch-level models. Therefore significant speedups over SPICE can be obtained.

A word of caution is probably in or&r regarding the choice of benchmarks. SPICE does not take advantage of circuit latency whereas Mom, Irsim, Bisim, and most of the timing simulators do. Therefore in a comparison it is possible to make the latter simulators seem arbitrarily good compared to SPICE by simply using benchmark circuits with large amounts of latency. Such benchmarks are not unrealistic because large circuits tend to have large amounts of latency.

However, Mom's ability to take advantage of latency is not the primary issue of interest here. Techniques for partitioning circuits and avoiding the analysis of latent subcircuits have been applied to many circuit, timing, switch, and gate-level simulators. Instead the questions of particular interest here are:

1. How does the use of approximate piecewise linear transistor models and moment analysis compare with the use of accurate nonlinear models and numerical integration?

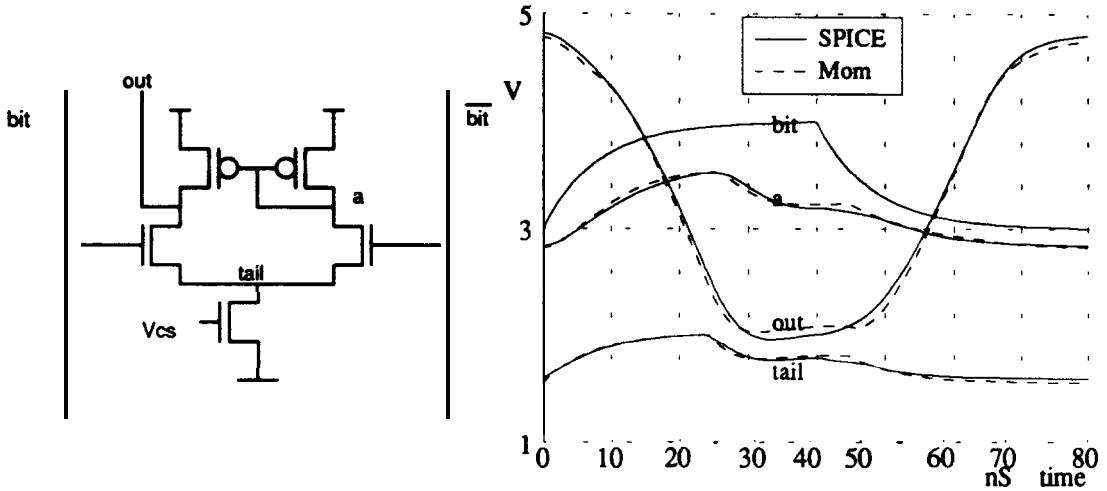
2. How much does the ability to handle more general piecewise linear models impair the efficiency of Mom compared to the efficiency of a dedicated switch level simulator?

Because it is in common use we would like to use SPICE as an example of a simulator employing numerical integration. Therefore in this chapter the issue of latency has been neutralized by considering only small circuits with little latency.

7.1.1 CMOS

To achieve efficient simulation Irsim takes advantage of characteristics shared by most MOS digital circuits, However, circuits are occasionally encountered that violate some

of its assumptions. The static RAM sense amplifier previously described in Chapter 3 (repeated here for convenience in Figure 7.1 (a)) is an example of such a circuit. Because



(a) Circuit (b) Response: SPICE vs Mom

Figure 7.1: Sense Amplifier for Static RAM.

the circuit has outputs that do not swing from rail to rail, has inputs whose thresholds are not halfway between the power rails, has NMOS and PMOS transistors that are simultaneously on and pull the output in opposite directions, and has a device whose input is connected to one of the circuit's outputs (feedback) it can be difficult for Irsim to simulate. In contrast, Mom's more accurate transistor models and its ability to handle more diverse topologies (including feedback) enable it to do a better job of matching SPICE (Irsim simply reports that the state of all nodes are undefined). Figure 7.1 (b) compares the response of SPICE using nonlinear models with the response of Mom using Level-l models. The parameters for the Level-l models were chosen by linearizing the SPICE transistors when the circuit is biased at its switching threshold. The fit is remarkable considering the simplicity of the transistor model.

However, the additional accuracy comes at the expense of reduced simulation efficiency. Table 7.1 compares the execution times of SPICE and Mom for a number of circuits to be considered in this section. From this table we see that for this example Mom is 61 times faster than SPICE. In contrast (as we shall see in a following section) Irsim is usually at least 3 orders of magnitude faster than SPICE.

	SPICE	Mom	$\frac{\text{SPICE}}{\text{Mom}}$
SRAM Sense Amp	1.9	.031	61
DRAM Cell	18.8	.075	250
ECL RAM Cell	4.2	.102	41
Diode Decoder	8.2	.102	81
BiCMOS Buffer	7.5	.054	140
BiNMOS Buffer	5.1	.008	650
BiCMOS RAM Cell, Read	2.3	.008	250
BiCMOS RAM Cell, Write	4.6	.012	390

Table 7.1: Execution Time of Example Circuits (seconds).

The dynamic RAM cell and sense amplifier[SSD72] in Figure 7.2 is, perhaps, a more dramatic demonstration of the capabilities of Mom. Although Irsim can be coerced to yield a logically correct simulation of the SRAM sense amplifier if the resistances and thresholds of the transistors are specially selected, the DRAM cannot be similarly accommodated.

The DRAM stores data as charge on capacitors in the memory cells. A ***read cycle*** begins with both bit lines, ***bit*** and bit precharged to V_{dd} and s precharged to $(V_{dd} - V_{tn})$. When ***wordLine*** is raised, charge sharing takes place between the bit lines and the selected cells with the result that the two bit lines are charged to slightly different voltages. Then the sense amplifier is turned on to magnify this voltage difference. When ***T3*** turns on, s begins to fall which will cause either ***T1*** or ***T2*** to turn on, depending upon which bit line is higher. For example, if ***bit*** is higher then ***T2*** will turn on, $\overline{bit}$ will be pulled to ground and ***bit*** will be pulled to Vdd.

However, note that ***T1*** and ***T2*** are turned on by pulling the source terminals low. Because the switched resistor model can only be turned on by pulling the gate terminal high it is incapable of modeling the behavior of those two transistors. However, if those transistors are simulated using Mom's MOS Level-l model, the correct circuit behavior can be obtained. Figure 7.3 shows plots of the bit line waveforms generated by SPICE and Mom for a read cycle followed by a precharge. For Mom's simulation the Level-l model was only used for ***T1***, ***T2***, and ***T3***. Everywhere else Level-O models were employed. Although Mom's response differs from SPICE's, it is probably adequate for a first order verification of the entire DRAM.

Table 7.1 shows that for this example Mom is 250 times faster than SPICE. The

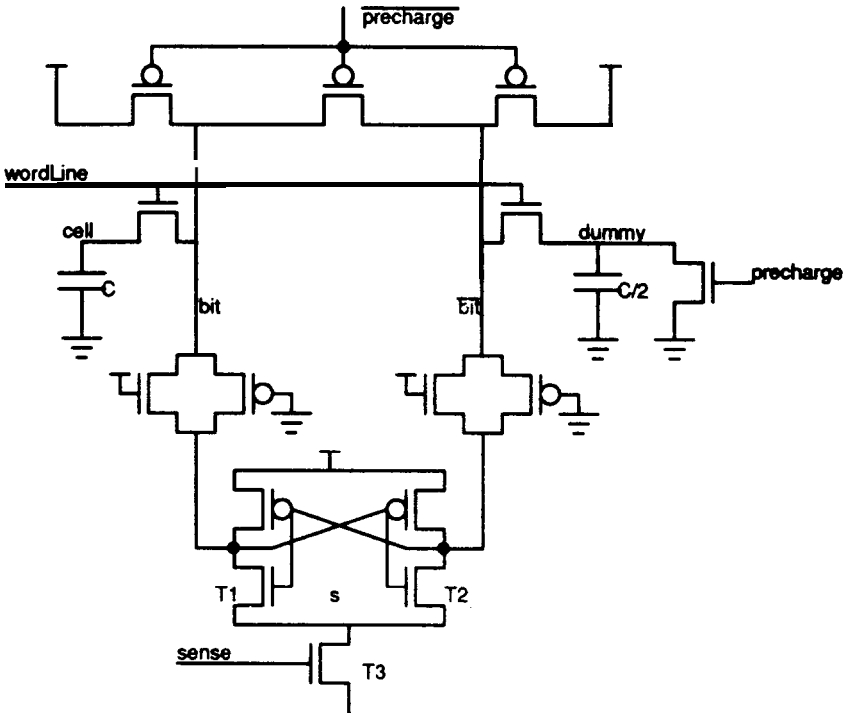


Figure 7.2: Dynamic RAM Cell and Sense Amplifier

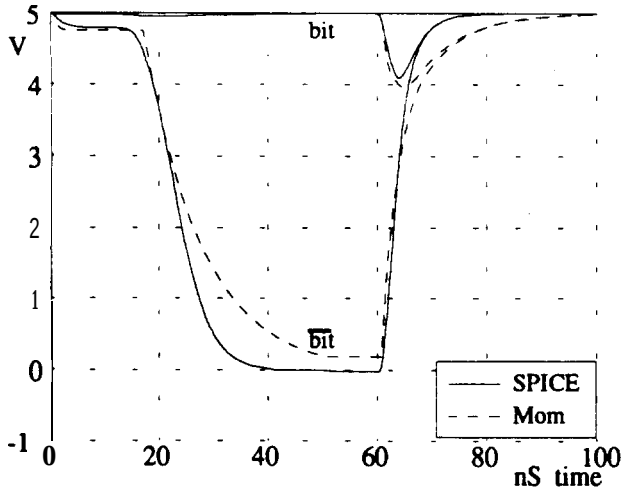


Figure 7.3: DRAM Bit Lines: Read followed by Precharge

improvement in efficiency is probably due to the extensive use of the simple switched resistor model.

7.1.2 ECL

The ECL switch-level simulator, Bisim, is based upon tracing **paths** through current steering networks formed by bipolar transistors (Figure 7.4). Negative current can be thought of as

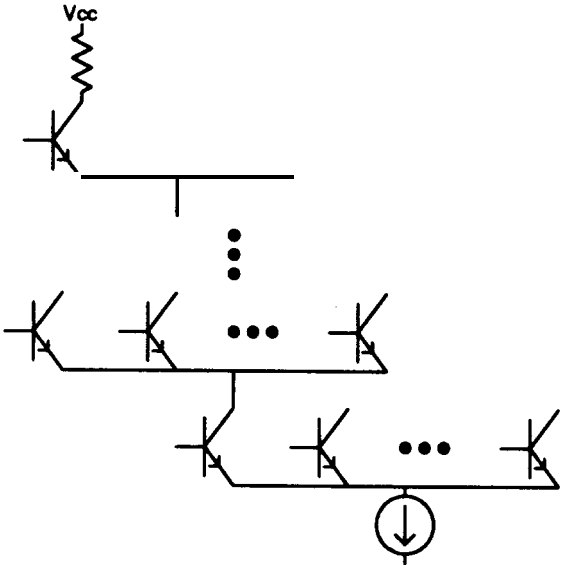


Figure 7.4: ECL Current Steering Networks.

originating from the current source at the bottom of the network and rising towards the top. When the current encounters a node with multiple emitters attached, it is steered through the transistor with the highest base. If the current encounters a resistor then it has reached an output and the resulting voltage drop causes the output to fall. Thus a simple path tracing algorithm is sufficient to determine the behavior of textbook ECL logic gates.

However, real **IC's** often contain a greater variety of circuit forms. For example in or&r to lower an output's logic high level a resistor Tee (Figure 7.5) is sometimes used. Thus, the simulator must be prepared to deal with resistor networks in place of the load resistor. Heuristics can be used to handle simple networks, but the diode decoder described in Chapter 5 (Figure 7.6) is an extreme example of a network that can't be handled by Bisim. Not only does the network contain nonlinear devices, but it also contains loops. However,

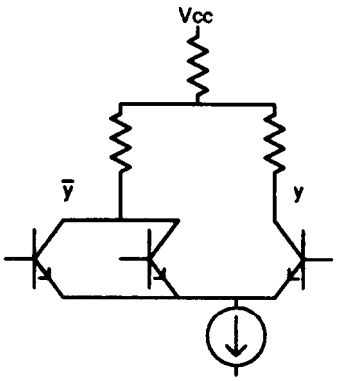


Figure 7.5: Resistor Tee as ECL Load.

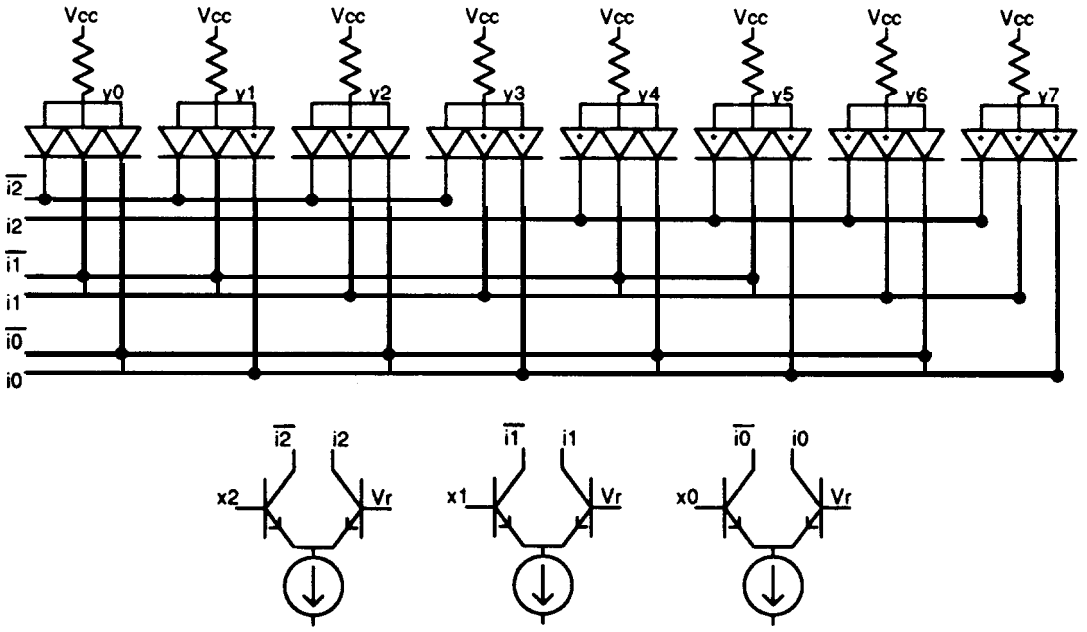


Figure 7.6: Diode Decoder.

because Mom can handle arbitrary networks of piecewise linear devices it can successfully simulate the circuit. Figure 7.7 depicts responses at three diode decoder outputs when one

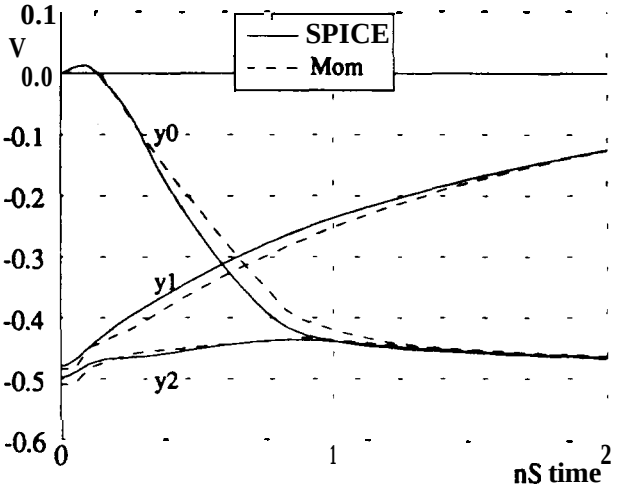


Figure 7.7: Diode Decoder Response.

of the address lines changes. The figure illustrates a reasonable match between Mom and SPICE. For this example Mom utilizes Level-O bipolar models and is 81 times faster than SPICE.

Another problem arises when current is shared between multiple transistors. In the current steering algorithm described above it was assumed that one transistor's base was sufficiently higher than all the others such that all the current went into a single transistor. However, sometimes several bases are so close that the current is divided between them. In fact, degeneration resistors are sometimes deliberately inserted in series with the emitters (Figure 7.8) in order to achieve an even division of current. For many simple cases the

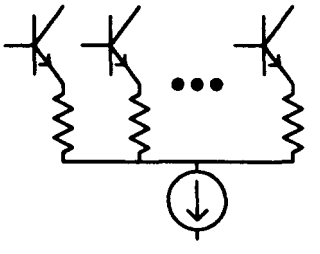


Figure 7.8: Emitter Degeneration Resistors to Cause Current Sharing.

current steering algorithm can be modified to split the current correctly. However the Schottky clamped ECL **RAM[KSM+78]** shown in Figure 7.9 has proven to be beyond the

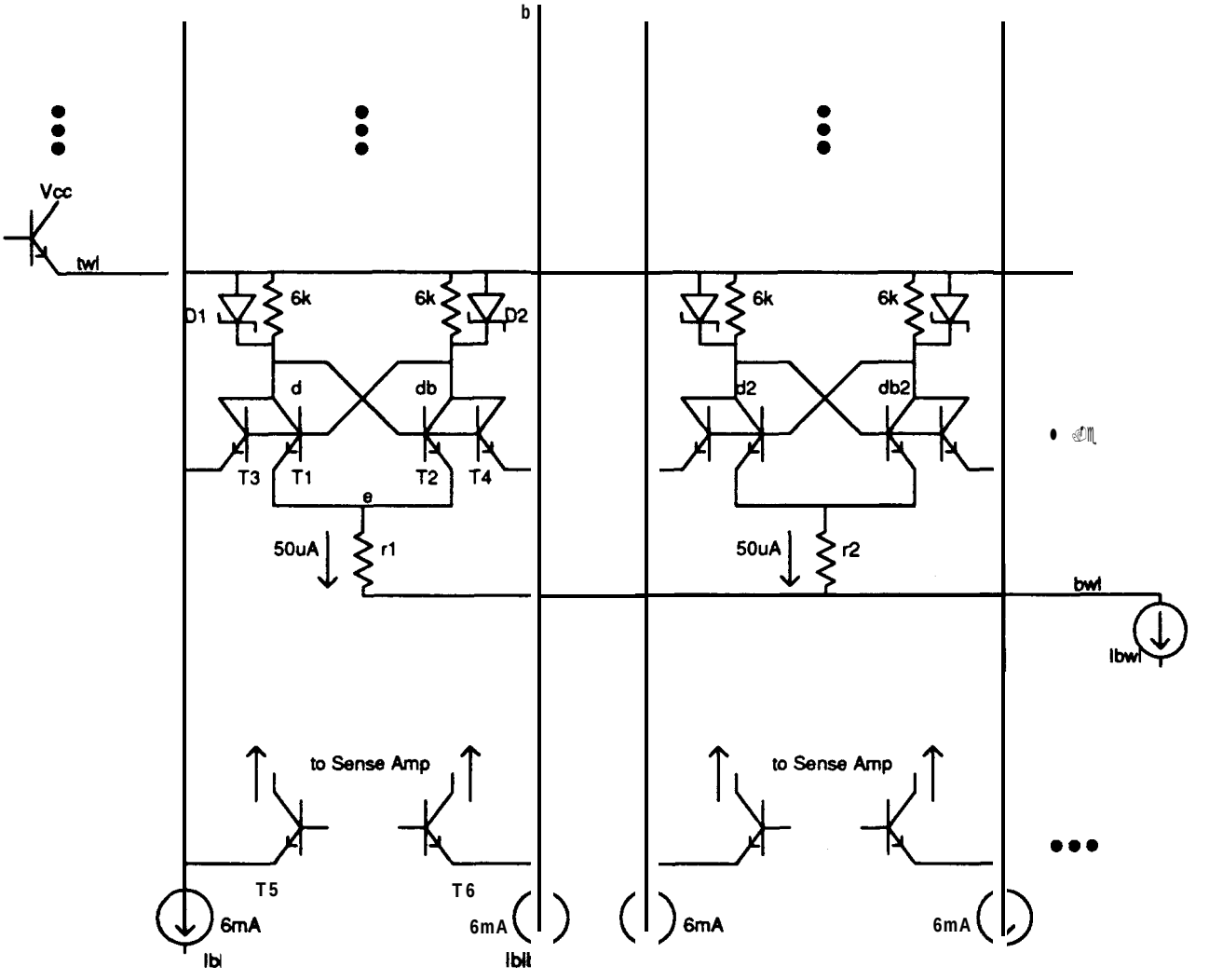


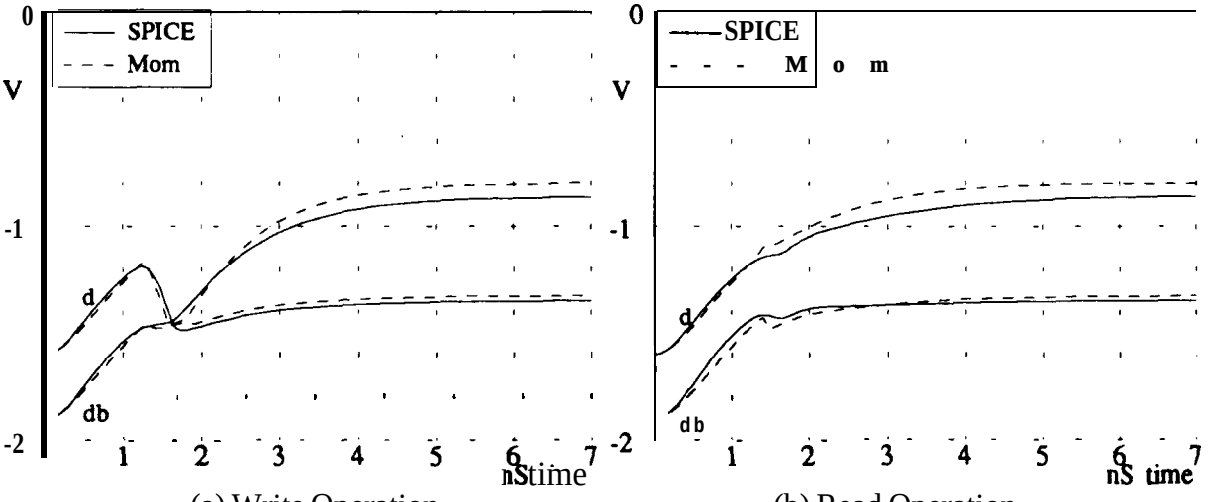
Figure *7.9:* Schottky Clamped ECL RAM.

capabilities of Bisim. Each cell stores data using a cross coupled pair *T1* and *T2* which requires a small ***standby*** *current.* In order to avoid the overhead of a separate current source for each cell, the standby currents are obtained by dividing the current from a single current source, *Ibwl* shared by all cells in a row. Bisim's problem is that in order to divide the current entering the bottom word line, *bwf,* it needs to know the voltages of the cell nodes *d, db, d2, db2,. . . .* In turn, in order to determine the voltages of the cell nodes, Bisim needs

to know how to divide the current.

For our example the standby current is $50\mu A$. Thus depending upon whether a 1 or a 0 is stored, either *d* or *db* will be 300mV below the the top word line, *twl*. A cell is read by raising the top word line and supplying current *I61* and *Iblb* to the bit lines *bl* and *blb*. Then the emitter follower attached to the highest cell node will turn on raising the corresponding bit line. For example, if *d* is high then *T4 will* turn on pulling up *bfb. The* Schottky diode *D2* prevents the bit line current from driving *T4* into saturation. The bases of *T5* and *T6 are* biased halfway between the high and low logic levels of the cell and are used to sense the state of the bit lines. To write the cell the top word line is raised and current is supplied to only one bit line. For example if *twl* is raised and *bl* gets current then that current will flow through *T3* thereby setting *d* to a logic level 0.

A RAM fragment consisting of one row of two cells was constructed and simulated using SPICE and Mom. As a compromise between modeling the effects of floating capacitance between tightly coupled nodes, and allowing the independent analysis of loosely coupled nodes, the Mom simulation utilized Level-l bipolar models for the cross coupled cell transistors *(TI* and *T2)* but Level-O models everywhere else. Figure 7.10 (a) shows the data

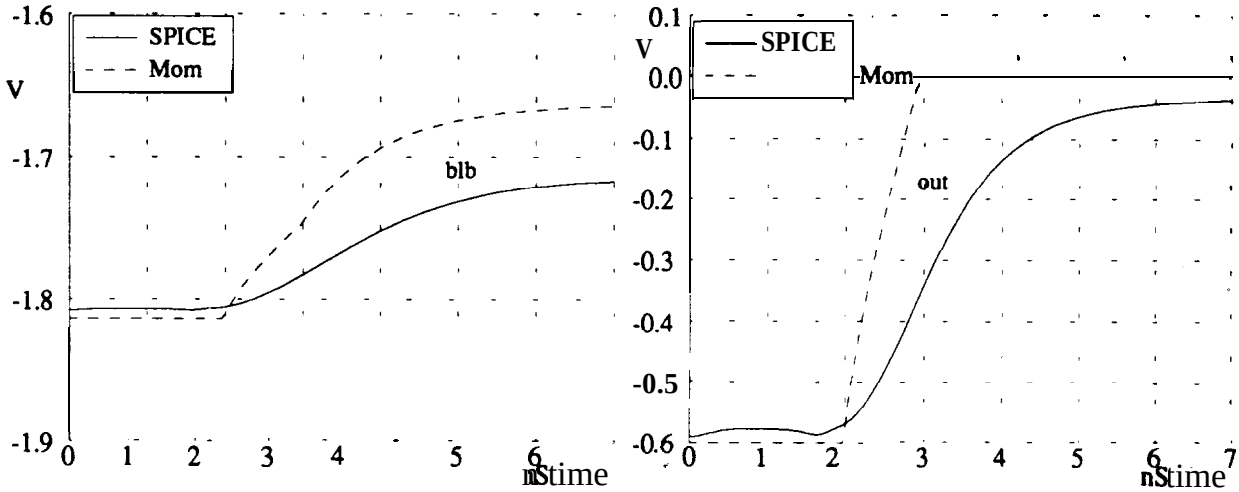


(a) Write Operation (b) Read Operation

Figure 7.10: ECL RAM Cell Internal Nodes.

nodes of a cell during a write operation. Note that for a brief instant of time $(t \sim 1.5ns)$ both members of the cross coupled pair are on and the response includes an unstable pole.

During a read operation (Figure 7.10 (b)) one of the limitations of our bipolar transistor models becomes evident. Because Mom neglects the base current, it fails to take into consideration the degradation of the high level caused by the base current of the emitter follower pulling up the bit line. For example, if *d* is high then $(Iblb/\beta) \sim 80\mu A$ will flow into node *d* thereby degrading its logic high level by about 50mV. Although the cell nodes' errors appear to be moderate, the actual bit line swing is only about 100mV and hence Mom significantly over estimates the bit line swing (Figure 7.11 (a)) and under estimates the



(a) Bit Lines (b) Sense Amp Output

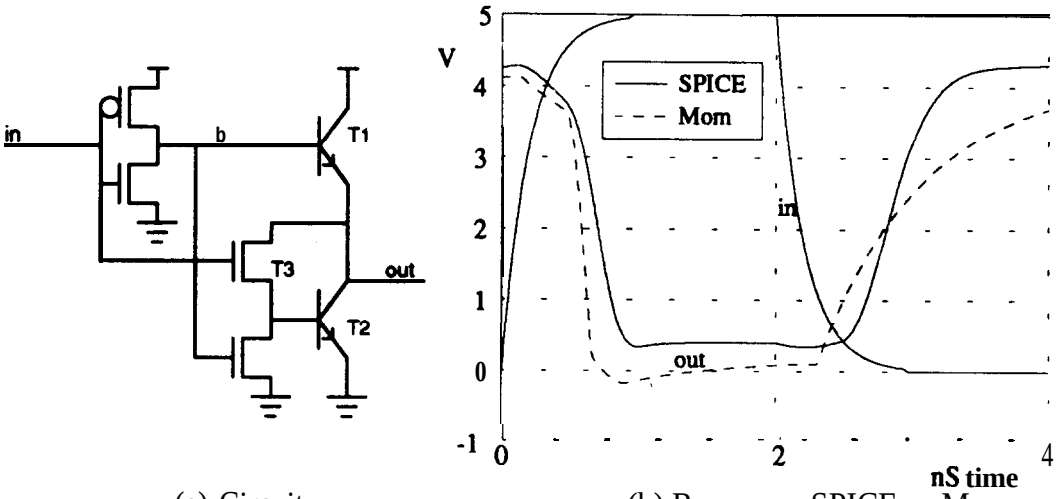
Figure 7.11: ECL RAM Read.

read delay (Figure 7.11 (b)).

However, note that the ECL RAM is not typical of ECL circuits. The desire to minimize both the power consumption and access time of the RAM leads to large ratios of bit line to standby current. In addition the need to keep the cell size small leads to high current densities in the access transistors which in turn causes some β rolloff at the high bit line current levels.

7.1.3 BiCMOS

An increasing number of circuit designs utilize both bipolar and MOS transistors on the same chip. Unfortunately neither Irsim nor Bisim can handle these new BiCMOS designs.



(a) Circuit (b) Response: SPICE vs Mom

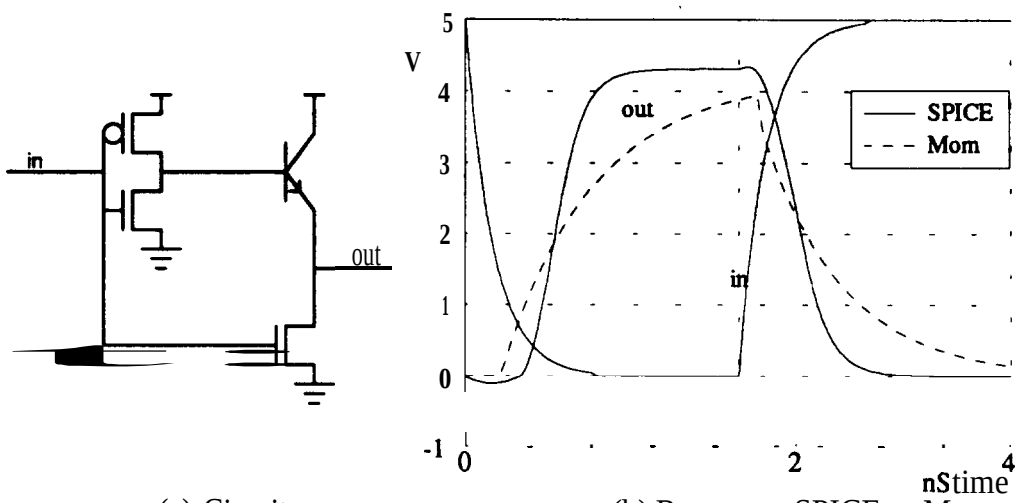
Figure 7.12: BiCMOS Buffer.

Figure 7.12 (a) shows one of several variations of the BiCMOS buffer[GM88]. In general, BiCMOS buffers attempt to take advantage of the high current driving capabilities of bipolar transistors to charge and discharge large capacitive loads. In this circuit *T1* is used to pull the output up to within a diode drop of the upper power supply rail while *T2* is used to pull the output down to within a diode drop of the lower power supply. Hard saturation of *T2* is avoided by arranging for its base drive (via *T3)* to disappear once the output has reached its low level. This circuit was simulated by Mom using Level-O models for all MOS and bipolar transistors. However unlike for ECL, this circuit has no current sources to provide convenient hints about the operating regions of the bipolar transistors. Therefore, the bipolar operating regions were estimated by examining the slopes of the waveforms from a circuit simulation.[2] The results are compared with SPICE in Figure 7.12 (b). The falling output transition predicted by Mom is too rapid and incorrectly drops below the power rail because Mom neglects base current and *T2* actually saturates *momentarily*. Mom's rising output transition takes the form of a simple exponential rather than a ramp because node *b* is driven by switched resistors and consequently has a single time constant waveform estimate. However, the loss of accuracy comes in exchange for a speedup of 140 over

[2]Further experiments reveal that the exact choice of region of operation has little affect on the parameters of our model because the high current levels cause the transconductance of the exponential device to be swamped by the parasitic emitter resistance.

SPICE.

Another variant of the BiCMOS **buffer[WR83]** is shown in Figure 7.13 (a). This



(a) Circuit (b) Response: SPICE vs Mom

Figure 7.13: **BiNMOS** Buffer.

buffer, sometimes referred to as the ***BiNMOS*** buffer, replaces the bipolar **pulldown** with an NMOS in order to obtain a larger output swing. The circuit was simulated by Mom using Level-O models everywhere. The responses predicted by SPICE and Mom are compared in Figure 7.13 (b). Because the falling output transition is now determined by a switched resistor model, it too is a simple exponential. Additionally the more extensive use of switched resistor models results in an improved **speedup** over SPICE of 650: 1.

The final example in this section is a BiCMOS RAM cell (Figure 7.14) that combines techniques from CMOS and ECL RAM **design[YHW88].** In common with CMOS static RAMs, the memory cell consists of a pair of cross coupled CMOS inverters *(TZ-T4).* However, instead of being connected to the top power supply rail, the sources of ***T1***, and ***T2*** are connected to a read word line driven by an emitter follower, ***T5***. In common with ECL RAMs, the cell is read by raising the read word line and sensing the change in a cell's logic high level via the emitter follower *T6.* For example, if cell node ***db*** is high then ***T6*** will turn on and pull up the read bit line. The cell is written by placing the write data on the write bit line, and raising the write word line. The write access transistor, ***T7*** is sized so that it overpowers ***T1*** and *T3.* This circuit was simulated by Mom using Level-O

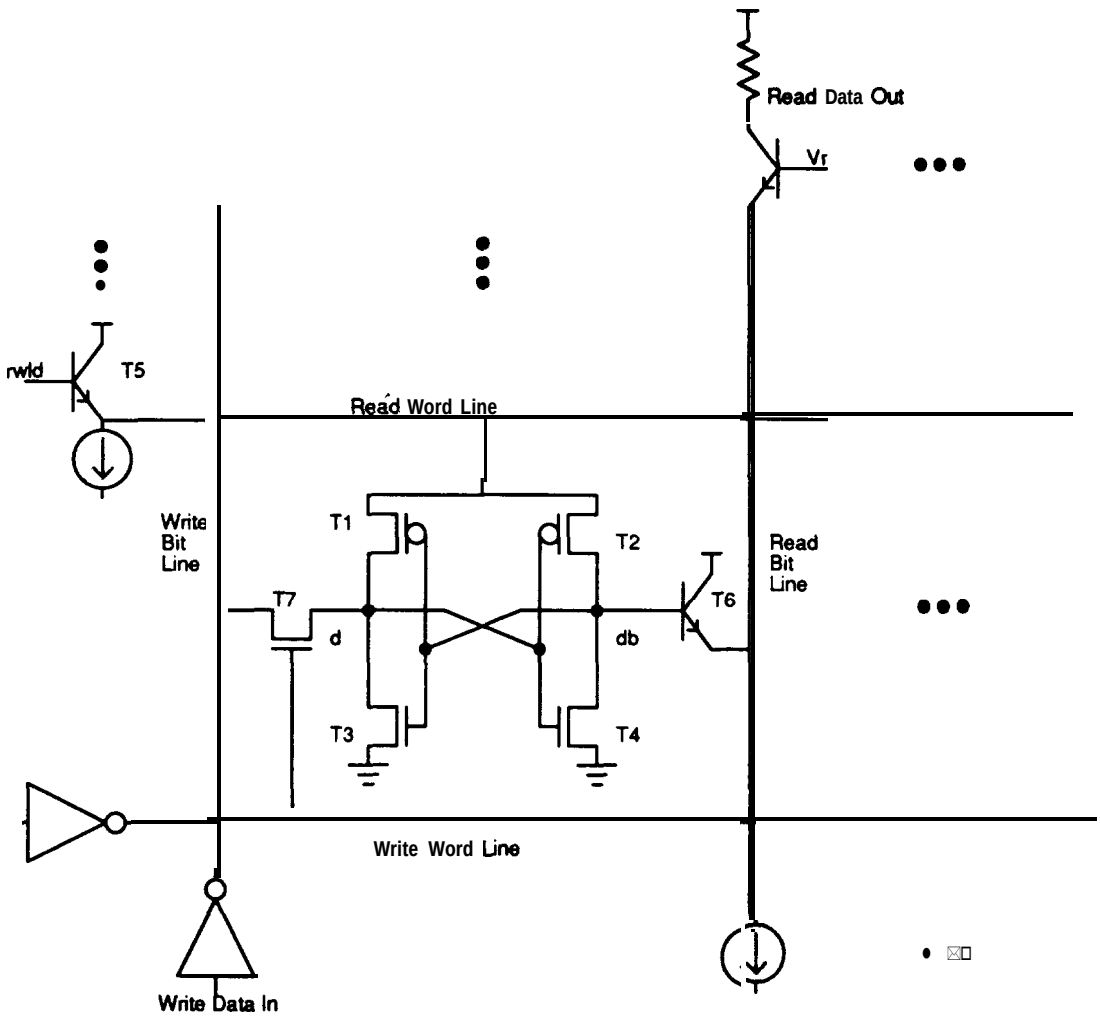
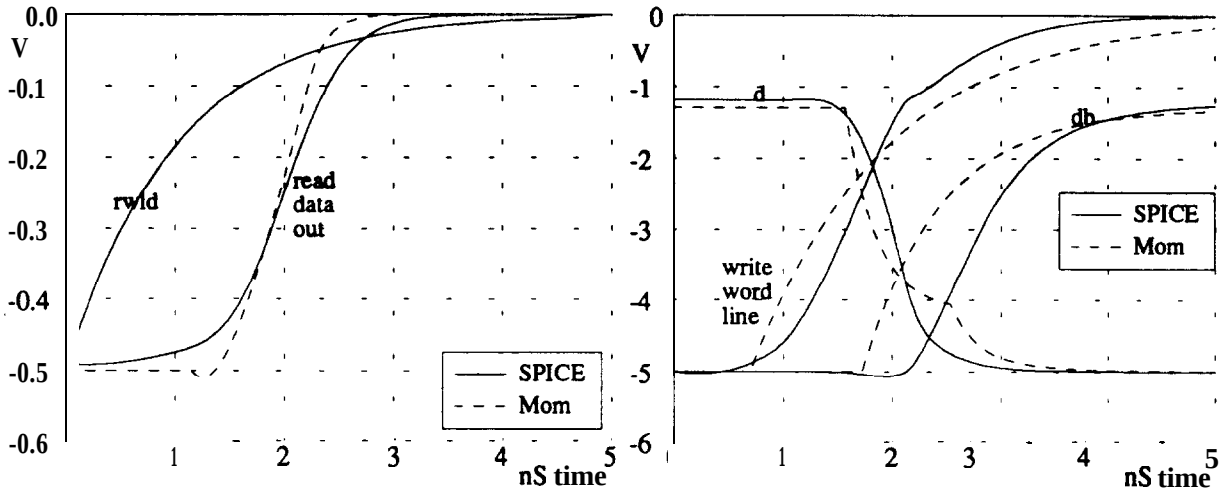


Figure 7.14: BiCMOS RAM.

transistors everywhere. The results for both read and write operations were compared with SPICE. Figure 7.15 (a) shows the data output for a read operation. The earliest rising



(a) Data Output During Read (b) Cell Data Nodes During Write

Figure 7.15: BiCMOS RAM Response.

transition is the input to the read word line driver (*T5*) which initiates the read operation. Figure 7.15 (b) shows the cell nodes during a write operation. The write bit line is zero so ***d*** is written to zero. In the plot the earliest rising pair of waveforms are of the write word line transitions which initiate the write. The falling pair of waveforms are ***d*** and the later rising pair of waveforms are ***db***. Note that there is a significant mismatch between the SPICE and Mom waveforms for *db.* One factor that contributes to this mismatch is the inappropriate parameterization of *T2.* The PMOS Level-O model was characterized for static CMOS logic gates which presume that $|V_{gs} - V_{tp}| \sim 4$ volts. However during a write operation the read word line sits 1.3 volts below the top power supply rail and hence $|V_{gs} - V_{tp}| \sim 2.7$ volts. Thus for this circuit Mom overestimates the current drive of *T2* by at least a factor of 1.5. On the other hand, for the read and write operations, Mom's speedup over SPICE is 250 and 390, respectively.

7.2 Performance Compared to Switch-Level Simulation.

The previous section illustrated the additional flexibility obtained by incorporating piecewise linear models into the switch-level framework. However, increased generality usually comes at the expense of decreased efficiency. To investigate this issue a number of circuits were simulated by Mom, Irsim, and Bisim. Examinations of the execution times, execution profiles and statistics of the simulations lead to a better understanding of the overhead.

The selection of benchmark circuits for comparing circuit simulators and switch-level simulators can be tricky. On one hand we would like the benchmark to be long enough so that execution times of the switch-level simulators are large enough to swamp overhead functions not of interest (reading the wirelist, command parsing, etc.). On the other hand, we would like the benchmark to be short enough so that the execution times of the circuit simulators can complete in a reasonable time. However, these goals can conflict. Even for small circuits with little latency, Rsim can easily achieve speedups of 4000 over SPICE. Therefore a benchmark that takes Rsim 1 minute will take SPICE 2.7 days.

Ring oscillators were selected for benchmarking because it is easy to adjust the simulation interval for each simulator to obtain reasonable execution times. Each ring oscillator was simulated by three simulators: SPICE-3d2, Mom, and (as possible) either Bisim or Irsim. For each simulator the simulation interval was adjusted to guarantee that at least 60 seconds of CPU time were used and that at least 10 periods were simulated. Since faster simulators end up simulating more periods the execution times are reported in terms of the amount of CPU time used to simulate a single logical transition of a logic gate in the ring. For example, if a simulator takes 1 second to simulate 10 periods of a 5 stage ring oscillator its performance is reported as $[1sec/(10 * 5 * 2)] = .010$ seconds per logic gate transition (in 1 period each of the 5 gates in the ring switches twice).

Table 7.2 reports the results. The CMOS and ECL ring oscillators used in the benchmark were described in Chapter 4. In addition ring oscillators using the two BiCMOS buffers described in the preceding section were included (although they obviously couldn't be simulated by Irsim or Bisim). The left most three columns report the amount of CPU time consumed by each of the simulators in units of milliseconds per logic gate transition (msec/LGT). The two middle columns compute the speedup of Mom over SPICE-3d2 and

	CPU time (msec/LGT)			Ratio CPU times		Period
	SPICE	(Bi/Ir)sim	Mom	SPICE / Mom	Mom / (Bi/Ir)sim	Error
CMOS0 Inverter Ring	932	.22	0.59	1600	2.7	0.4
CMOS0 NAND Ring	1850	.54	0.79	2300	1.5	28.0
CMOS 1 Inverter Ring	932	.22	11.70	80	53.2	2.6
CMOS1 NAND Ring	1850	.54	22.70	81	42.0	4.7
BJT0 ECL Ring 10%	824	1.08	38.20	22	35.4	10.2
BJT0 ECL Ring 20%	824	1.08	3.53	230	3.3	16.5
BJT1 ECL Ring	824	1.08	33.70	24	31.2	5.7
BJT2 ECL Ring CapLev=l	824	1.08	76.00	11	70.4	1.2
BiCMOS Buffer Ring	5800	—	30.98	187	—	9.9
BiNMOS Buffer Ring	3260	—	2.25	1400	—	10.5

Table 7.2: Simulator Performance on Ring Oscillators.

the degradation of Mom relative to the switch-level simulators. The last column reports the percentage error in Mom's prediction of the period of oscillation relative to SPICE.

One thing to note from this table is that Mom's performance relative to SPICE appears to be slightly better than indicated by Table 7.1. While Mom was 140 and 650 times faster than SPICE for the BiCMOS and BiNMOS buffers, it is 187 and 1400 times faster for ring oscillators formed from those buffers. This is probably because the simulations are long enough to amortize the cost of various overhead functions.

Also evident is the efficiency of switch-level simulation. Irsim and Bisim are between 4200 and 760 times faster than SPICE. In addition, Mom's increased generality extracts only a moderate performance penalty when switch-level models are employed. For circuits using the MOS switched resistor model ("CMOS0 Inverter Ring" and "CMOS0 NAND Ring") Mom is between 2.7 and 1.5 times slower than Irsim. For the circuit using the bipolar switched resistor model ("BJT0 ECL Ring 20%" and "BJT0 ECL Ring 10%") Mom is between 3.3 and 35 times slower than Bisim (although as discussed later, the last case is an anomaly). Note that for these models the accuracy of Mom is comparable to that of switch-level simulation.

However, as more accurate models are used Mom's efficiency decreases rapidly. When MOS Level- 1 models are used in the CMOS ring ("CMOS 1 Inverter Ring") Mom slows down by an additional factor of 20. When bipolar Level-2 models are used in the ECL ring

("BJT2 ECL Ring CapLevel=l") Mom slows down by an additional factor of 2. In return, Mom achieves increased accuracy. For this pair of circuits the period estimated by Mom is off by only 2.6% and 1.2% relative to SPICE. Such low errors are generally beyond the capabilities of Irsim and Bisim.

These data raise some interesting questions. Before constructing Mom we did not anticipate the extent to which more complex models would slow down the simulation. For example, the MOS Level- 1 model is just the MOS Level-O model with one additional region of linearity. It seemed surprising that one additional region would slow down the simulation by a factor of 20 or 30. In order to find out what was going on additional statistics were gathered from the simulations. Table 7.3 shows profiles of Mom's execution time for a number of benchmarks. The total execution time is first broken down into three categories: the time spent computing the responses of nodes, the time spent rescheduling devices, and the time spent on miscellaneous functions unrelated to the first two categories (reading the wirelist, parsing commands, etc.). Then each of the first two categories is broken down further. Computing the response of nodes involves constructing the group, computing the moments, and computing waveform approximations from those moments. Rescheduling devices involves reducing the order of the boundary waveform, finding the root of the reduced boundary waveform, polishing the root, and miscellaneous other functions.

Table 7.4 shows additional statistics gathered from the simulations. The first column gives the average number of nodes in a group. The second column gives the average number of poles in a waveform approximation. The third column gives the average number of waveform segments making up a single logical transition of a node.

From these tables it can been seen that as more detailed transistor models are used, the number of segments in a node transition increases. Whereas the Level-O CMOS rings have 1 segment per node transition, the Level-l versions of those rings have 5 and 7 segments. While the Level-O ECL rings have 3 segments per node transition the Level-l and Level-2 versions have 18 and 8 segments, respectively. The increase in the number of segments has a couple of causes. First, models with more regions of linearity generate more events as they move between those regions. Second, floating capacitance and gain propagate the effect of one device's region change to other nodes. In the extreme ("BJT1 ECL Ring") floating capacitors couple all nodes together (there are a total of 18 nodes in that ring) such

68.5% **nodeResponse** 27.2% **buildGroup** 35.7% **computeMoments** 5.6% waveformApprox 17.8% reschedule 0.0% **orderReduction** 8.6% **findRoot** 0.0% **rootPolish** 9.2% other 13.7% other (a) CMOS0 Inverter Ring	25 .3% **nodeResponse** 8.5% **buildGroup** 12.9% moments 3.9% **waveformApprox** 69 .0% reschedule 11.6% **orderReduction** 30.4% **findRoot** 10.2% **rootPolish** 16.8% other 5.7% other (b) CMOS 1 Inverter Ring
68.3% **nodeResponse** 25.6% **buildGroup** 33.2% **computeMoments** 9.5% waveformApprox 24.9% reschedule 0.7% **orderReduction** 15.1% **findRoot** 0.0% **rootPolish** 9.1% other 6.8% other (c) **BJT0** ECL Ring 20%	10.0% **nodeResponse** 2.9% **buildGroup** 3.4% moments 3.7% waveformApprox 89.4% reschedule 1.2% **orderReduction** 86.2% **findRoot** 0.8% **rootPolish** 1.2% other 0.6% other (d) **BJT0** ECL Ring 10%
68.7% **nodeResponse** 15.7% **buildGroup** 18 .0% moments 35 .0% waveformApprox 29.7% reschedule 10.7% **orderReduction** 13.3% **findRoot** 1.7% **rootPolish** 4.0% other 1.6% other (e) BJT2 ECL Ring	47.7% **nodeResponse** 15.6% **buildGroup** 23.7% moments 8.4% waveformApprox 47.6% reschedule 1.1% **orderReduction** 37.7% **findRoot** 0.9% **rootPolish** 7.9% other 4.7% other (f) **BiCMOS** Ring

Table 7.3: Mom Execution Profiles for Various Benchmark Circuits.

	nodes / group	poles / segment	segments / node transition
CMOS0 Inverter Ring	1.0	1.0	1.0
CMOS0 NAND Bing	2.0	1.3	1.0
CMOS 1 Inverter Ring	1.3	1.9	5.0
CMOS1 NAND Ring	2.8	1.8	7.0
BJT0 ECL Ring 10%	2.4	2.2	3.0
BJTO ECL Ring 20%	2.4	1.5	3.0
BJT1 ECL Ring	18.0	2.5	18.3
BJT2 ECL Ring CapLev=1	23.2	2.7	8.4
BiCMOS Buffer Ring	1.7	1.6	3.3
BiNMOS Buffer Ring	1.2	1.1	3.0

Table 7.4: Mom's Simulation Statistics for Ring Oscillators.

that every switching event of every transistor generates a new segment in the response of every node.

Apparently another affect of coupling is that waveforms become more complex. Note that while the "CMOS0 Inverter Ring" has only 1 pole in a waveform approximation the "BJT2 ECL Ring CapLev=1" has, on average, 2.7 poles in an approximation. Unfortunately an increase in the number of poles can have a serious impact upon efficiency. Tables 4.1 and 6.4 show that the costs of generating waveform approximations and finding the roots of boundary waveforms increase by factors of 20 and 40, respectively, as the number of poles increases from 1 to 3.

Thus the 20 x degradation caused by the introduction of Level-l models into the "CMOS0 Inverter Ring" can be explained. The table reveals that the Level-l ring has 5 segments per node transition and that each segment consists of, on average, 1.3 poles. In contrast the Level-O ring has only one segment per transition which consists of a single pole. However, the table of execution profiles indicates that something else is going on. For the Level-l ring Mom spends almost 70% of its time rescheduling devices as compared with 18% for the Level-O ring. Note that for the MOS Level-O model each region of linearity is bounded by a single hyperplane. Thus when a Level-O MOS is rescheduled it is only necessary to find the root of a single boundary waveform. However each region of the MOS Level- 1 model is bounded by two hyperplanes. Thus it is necessary to find the roots of two boundary waveforms when rescheduling a Level-l MOS. Also, the region of linearity of a

Level-O MOS is determined by the gate voltage alone. However, for the Level-l MOS the region of linearity depends upon all three terminal voltages. Thus after a new waveform segment is computed for a node it is necessary to reschedule only those Level-O **MOS's** with a gate attached to the node, but all Level- 1 MOS **'s** with either a gate, source, or drain attached to the node. For the inverter ring this means that twice as many Level-l devices must be rescheduled as Level-O &vices for each waveform segment. In all, the "CMOS1 Inverter Ring" requires Mom to perform 5 x 2 x 2 = 20 times as many boundary waveform root computations per logic gate transition as the **"CMOS0** Inverter Ring".

An additional problem with device rescheduling is apparent from the execution time of **"BJT0** ECL Ring 10%". The tenfold increase in execution time resulting from decreasing the error threshold of "BJTO ECL Ring 20%" seems somewhat surprising. More startling, however, is the fact that its execution time exceeds that of **"BJT1** ECL Ring" which combines the entire ring into one group! The program profile reveals that the problem is once again with device rescheduling: **"BJT0** ECL Ring 10%" spends almost 90% of its time rescheduling devices. A more detailed breakdown of the execution profile reveals that almost all this time is spent finding the roots of boundary waveforms consisting of 1 simple plus 2 complex conjugate poles. On average 195 piecewise quadratic segments are examined before a root is found! Clearly the simple, general algorithm used for this case needs to be replaced by an algorithm employing better root bracketing. However, "BJTO ECL Ring 10%" is an anomaly. No other circuit demonstrated such poor performance. For example, both **"BJT1** ECL Ring" and "BIT2 ECL Ring **CapLev=1"** have more than twice the number of simple plus complex-conjugate pole boundary waveforms (46.4% and 57.8% vs 20.2% according to Table 6.5) and yet the average number of segments examined is only 2.6 and 2.9, respectively.

The most striking conclusion to be drawn from the execution profiles is the relatively high cost of rescheduling devices. Although most of the time in Irsim is spent computing delay approximations, for Mom the cost of rescheduling devices often dominates. Even in the best case device rescheduling never takes less than 18% of the total execution time. What's worse, only fairly simple models are considered here. If transistor models as accurate and flexible as SPICE's were implemented we would expect that Mom could easily be slower than SPICE! Thus it appears that device rescheduling is fundamental problem

with Mom's approach to simulation which will limit its utility in those cases where the full accuracy and generality of circuit simulation are desired.

7.3 Summary

The objective behind producing *Mom* was to create a simulator that extended the accuracy and flexibility of existing MOS and bipolar switch level simulators while simultaneously preserving much of their efficiency for the simplest cases. This chapter evaluates the extent to which we have achieved those goals. First, the additional flexibility obtained by incorporating piecewise linear models into the switch-level framework was demonstrated. Mom was applied to a number of circuits that appear to be just beyond the capabilities of existing switch-level simulators. Then Mom's efficiency was compared to that of existing switch-level simulators. It appears that Mom is just a factor of 1.5 to 3.3 times slower than switch-level simulation for comparable accuracy. However, Mom's efficiency decreases rapidly as more sophisticated models are used to obtain greater accuracy. More elaborate models increase the size of groups, increase the number of segments per logic transition, and increase the complexity of waveforms. In addition, the use of more elaborate models can cause device rescheduling to dominate the cost of simulation. More complex models must be rescheduled more frequently, and require more work to reschedule.

Chapter 8

Conclusion

By constructing a prototype simulator, *Mom,* we have shown that it is possible to modify Rsim's switch-level simulation framework to allow more general piecewise linear transistor models in place of the switched resistor model. In addition we have shown that many of Rsim's restrictions can be removed: Mom allows floating capacitors, non-tree circuit topologies, and feedback. Although these enhancements require extensive changes, they don't seriously impair the simulator's efficiency for the simplest cases. That is when the simplest switch-level models are used Mom achieves speeds and accuracies comparable to those of dedicated switch-level simulators. In addition the ability to handle more general piecewise linear models gives the simulator a great deal more flexibility. Mom can simulate circuits that can't be simulated by Rsim or Bisim and yet with substantial speedups over SPICE.

This approach looks particularly promising for simulating circuits that are just beyond the capabilities of switch-level simulation. Frequently most of a circuit can be simulated using switch-level models and only small portions require more accurate models. Because Mom has been structured such that the additional generality is paid for only where it is used it can simulate those circuits with only a minor degradation of efficiency.

However our experiments uncovered some limitations to the approach. Apparently the overhead of rescheduling devices will prevent Mom from replacing circuit simulators. Benchmarks show that Mom's speed falls precipitously as the complexity of transistor models is increased. Unfortunately increased model complexity fundamentally requires

much more work to be done in order to determine when devices change regions of linearity. We expect that the approach will lose its speed advantage when models having the full accuracy and generality of SPICE's nonlinear models are used.

Perhaps the most important contribution of this thesis is that it addresses the question: is moments *matching* a practical alternative to *numerical integration* for computing the transient response of nonlinear electrical networks? The results from this thesis indicate that the answer is "it depends". Consider the space bounded by circuit simulators at one end and switch-level simulators at the other (Figure 8.1). Remember that the timing

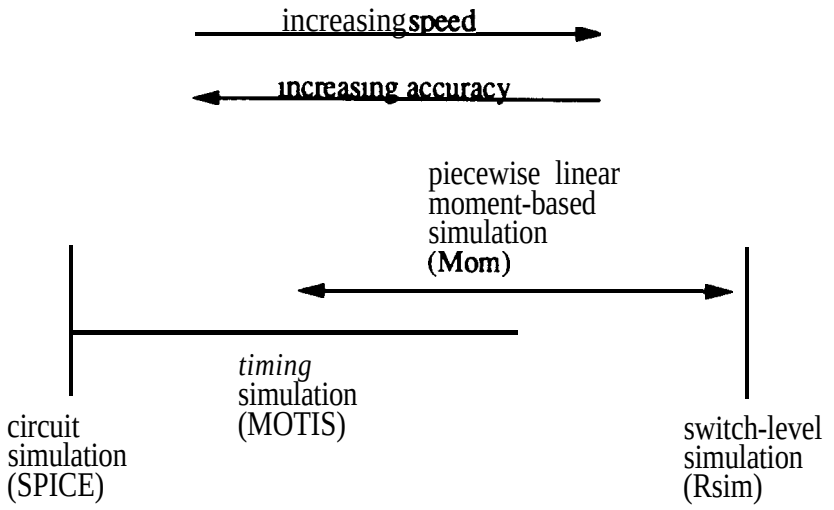


Figure 8.1: Simulation Space

simulators attempted to adapt the basic techniques used by circuit simulators (nonlinear device models and numerical integration) in order to extend the capabilities of circuit simulation in the direction of switch-level simulation. Despite some very impressive speedups, a gap remained; timing simulators never became fast enough to replace switch-level simulators. In an analogous fashion we have tried to adapt some basic techniques used by switch-level simulators (piecewise linear device models and moment analysis) in order to extend the capabilities of switch-level simulation in the direction of circuit simulation. The result is a simulator that fills the gap between timing simulation and switch-level simulation, although the initial indications are that this approach will not yield a replacement for circuit simulation.

An interesting perspective on the tradeoffs faced by the two approaches can be gained by considering both from the standpoint of *waveform approximation. The numerical integration* techniques used by circuit simulators approximate the time domain response using a polynomial in t, where the polynomial is chosen to match the low order terms of the Taylor series expansion of the actual response:

$$h(t) = h_0 + h_1 t + h_2 t^2 + \ldots \tag{8.1}$$

This produces an approximation (Figure 8.2) that converges to the actual response as $t \rightarrow 0$.

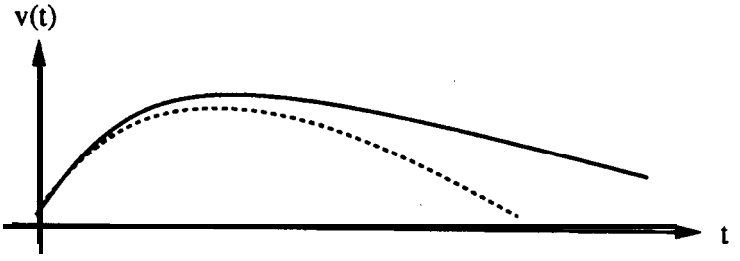


Figure 8.2: Numerical Integration Approximation

The consequence of this approach is that the size of the time step is limited by the need to maintain good convergence between the approximate and actual response. If the device models are strongly nonlinear then past samples of the response are probably not going to be good predictors of the future behavior. In that case it will probably be necessary to take small time steps anyway. However if the models are linear or nearly linear then moments matching offers a better alternative.

The moments matching techniques used by switch-level simulators approximate the **Laplace** transform of the response using a ratio of polynomials in s, where the polynomials are chosen in order to match the low or&r terms of the Taylor series expansion of the **Laplace** transform of the actual response:

$$H(s) = m_0 + m_1 s + m_2 s^2 + \ldots \tag{8.2}$$

The result is a frequency domain approximation that converges to the actual **Laplace** transform as $s \rightarrow 0$ or equivalently a time domain approximation (Figure 8.3) that converges as $t \rightarrow \infty$. Thus the region of convergence begins at $t = \infty$ for low order approximations

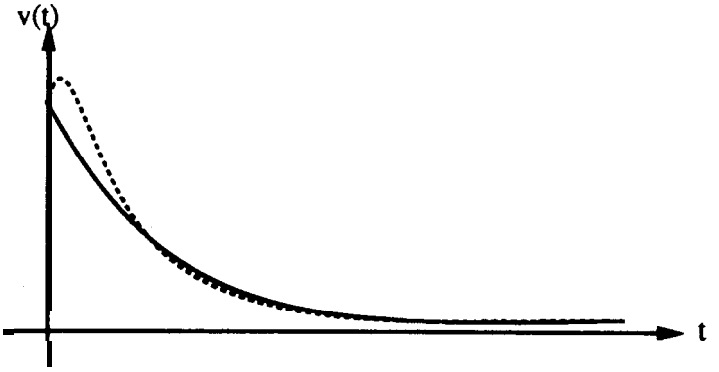


Figure 8.3: Moments Matching Approximation

and moves in towards the origin only as the order of the approximation is increased. For the case of Rsim the models are linear and we are only interested in the response at its 50% point. Apparently this point is far enough away from the origin that low order approximations are usually sufficient. However, as more complex piecewise linear models are used to approximate more strongly nonlinear behavior it becomes more likely that some device will switch in the vicinity of $t \approx 0$. When this happens the work that went into getting a good match for large t is wasted and the simulator retains what is essentially the worst part of the approximation. It appears that moments matching is a poor choice when the piecewise linear models are very detailed and have many regions of linearity.

From this perspective some related approaches appear to be worth investigating. Since numerical integration appears to be advantageous when devices are strongly nonlinear, and moments matching when devices are strongly linear, a hybrid approach could dynamically choose between the two approaches based upon the anticipated step size. This would avoid the work of generating a waveform approximation accurate at $t = \infty$ in those cases when only a small portion around $t = 0$ will be used. Simultaneously the accuracy problems (described in Chapter 4) associated with generating extremely large waveform approximations could be avoided. If a waveform approximation has an amplitude of several thousand volts, probably only a small portion around $t = 0$ will be used. In those cases numerical integration (or some other waveform approximation technique accurate near the origin) should be selected.

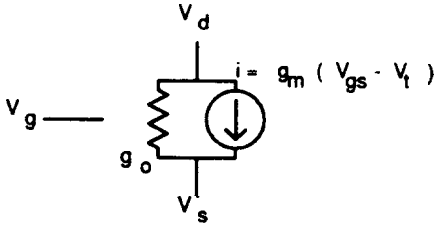
Other waveform approximation techniques are possible[McC89, Cha91]in addition to

those described above. Our experience with Mom indicates that the principle problem that must be addressed when trying to improve the efficiency of our simulator is the overhead of rescheduling devices. To a large extent this overhead is due to the difficulty of finding the roots of weighted sums of exponentials. Therefore, it would be worthwhile to investigate other waveform approximation techniques with the goal of finding one which produces approximate waveforms whose roots can be more readily computed. Such a technique could easily yield a faster approach to simulation.

Appendix A

MOS Level-l Threshold

One of the peculiar aspects of the MOS Level-l model is that the gate-source threshold depends upon the drain voltage. This was done to preserve the continuity of current just as the transistor turns on. In the saturation region, the current through the MOS Level-l



$$V_{gs} - (V_t - \frac{g_o}{g_m} V_{ds}) > 0 \qquad \text{(A.1)}$$

$$g_l V_{ds} - g_m (V_{gs} - V_t) > 0 \qquad \text{(A.2)}$$

Figure A. 1: Piecewise Linear MOS Model: Saturated Region

model (repeated above for convenience) is given by:

$$I_d = g_m (V_{gs} - V_t) + g_o V_{ds} \qquad \text{(A.3)}$$

Unfortunately, when $V_{gs} = V_t$ the model yields **nonzero** current due to the output conductance:

$$I_d = g_o V_{ds} \qquad \text{(A.4)}$$

If the model turns on precisely at $V_{gs} = V_t$, the resulting discontinuity in **current** can result in a non-physical staircase response for some source follower circuits (Figure A.2).

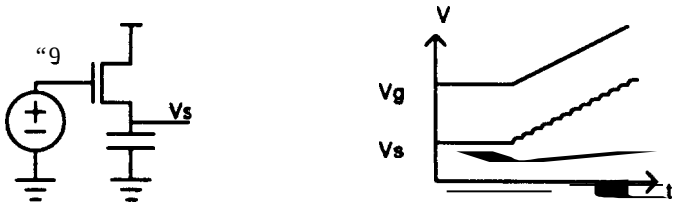


Figure A.2: Staircase Response from Source Follower

However, when the hyperplane bounding the saturation region is changed from

$$V_{gs} = V_t \tag{A.5}$$

to

$$V_{gs} = V_t - \frac{g_o}{\mathrm{Ym}} V_{ds} \tag{A.6}$$

this undesirable non-physical discontinuity is eliminated. That $I_d = 0$ at the new threshold can be verified by substituting Equation (A.6) into Equation (A.3.)

Appendix B

MOS Level-l Polytopes

In order to verify the hyperplane equations it is useful to plot the polytopes of the various regions of operation. Since the model has no internal connection to ground, we can arbitrarily set the source to ground and plot the polytopes in the two dimensions V_d and V_g (Figure B.l). Note that the regions in this figure are slightly more general than those of

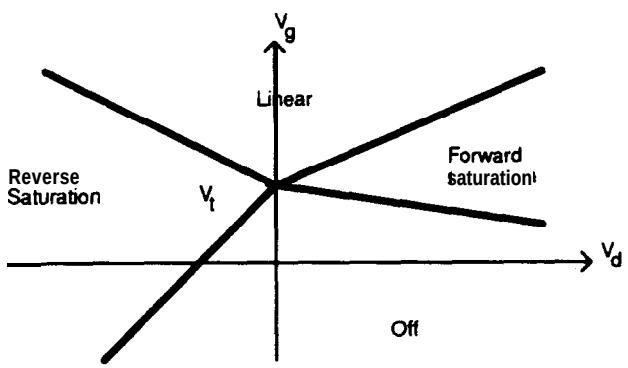


Figure B. 1: Polytopes of MOS Level- 1 model

the Level- 1 model because they include the reverse ($V_{ds} < 0$) as well as the forward modes of operation.

To plot the boundary between the forward saturation and off regions, set $V_s = 0$ in Equation (3.7) and solve for V_g

$$V_g = -\frac{g_o}{g_m} V_d + V_t \qquad \text{(B.1)}$$

We see that this boundary is a negatively sloped line through the point $(0, V_t)$. Similarly the equations for the linear — forward saturation, reverse saturation — linear, and off — reverse saturation boundaries are

$$V_g = \frac{g_l}{g_m} V_d + V_t \tag{B.2}$$

$$V_g = (1 - \frac{g_l}{g_m}) V_d + V_t \tag{B.3}$$

$$V_g = (1 + \frac{g_o}{g_m}) V_d + V_t, \tag{B.4}$$

respectively.

Note that all four hyperplanes pass through the point $(0, V_t)$. Therefore they partition space into four disjoint regions, and each region is bounded by exactly two hyperplanes. However, rather than model all four regions, our implementation restricts the model to the three forward regions by installing a hyperplane at $V_d = 0$ and interchanging the source and drain terminals anytime it is crossed. This simplifies the implementation because fewer regions need to be modeled, although it is probably less efficient because extra model state changes may be required.

Appendix C

Linearization of Bipolar Transistor Capacitances

The SPICE model for the bipolar transistor incorporates four nonlinear parasitic capacitances (Figure C. 1). The capacitors: Cje, Cjc, *and* ***Cjs*** are the capacitances associated with

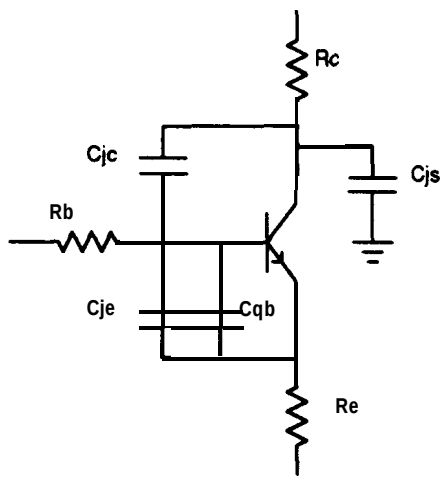


Figure **C.1** : Nonlinear Capacitances in SPICE bjt model

the base-emitter, base-collector, and collector-substrate junctions. The fourth capacitor: ***Cqb*** represents the storage of active charge in the base.

It is well known that semiconductor junction capacitances are well approximated by an "average" capacitance formed by dividing the change in depletion charge by the change in

voltage[HJ83, page 137]:

$$C_{eq} = -\frac{C_{j0}\phi_0}{(V_2 - V_1)(1-m)}\left[\left(1 - \frac{V_2}{\phi_0}\right)^{1-m} - \left(1 - \frac{V_1}{\phi_0}\right)^{1-m}\right] \tag{C.1}$$

Here, C_{j0} is the zero bias capacitance of the junction, V_1 and V_2 are the limits of the voltage swing, ϕ_0 is the built in junction potential, and m is the junction grading coefficient (for abrupt junctions $m = 1/2$, for linearly graded junctions $m = 1/3$).

This result is accurate if the junction is reverse biased or forward biased slightly ($V_f < \phi_0/2$). For larger forward voltages it can be modified to reflect SPICE's model[AM88, page60] of the capacitance of junctions under heavy forward bias:

$$C_{eq} = \frac{Q(V_2) - Q(V_1)}{V_2 - V_1} \tag{C.2}$$

$$Q(V) = \begin{cases} -\frac{C_{j0}\phi_0}{1-m}\left[\left(1 - \frac{V}{\phi_0}\right)^{1-m} - 1\right] & V \le \frac{\phi}{2} \\ -\frac{C_{j0}\phi_0}{1-m}\left[2^{m-1} - 1\right] + & \\ C_{j0}2^m\left[(1-m)\left(V - \frac{\phi_0}{2}\right) + \frac{m}{\phi_0}\left(V^2 - \frac{\phi_0^2}{4}\right)\right] & V \ge \frac{\phi_0}{2} \end{cases} \tag{C.3}$$

It less well known that the small signal model for stored base charge provides a good model for the base charge of an ECL inverter. Our piecewise linear model is roughly the small signal model of a bipolar transistor linearized about the switching point of the ECL gate. For our model:

$$I_c = g_m(V_{be} - V_{on}) \tag{C.4}$$

$$q_b = \tau_f I_c \tag{C.5}$$

$$I_b = \frac{dq_b}{dt} \tag{C.6}$$

$$= \tau_f \frac{dI_c}{dt} \tag{C.7}$$

$$= \tau_f g_m \frac{dV_{be}}{dt} \tag{C.8}$$

Thus the base charge is modeled by the capacitance: $g_m \tau_f$ [MK77, page 276). Figure C.2 compares the response of ECL inverters using SPICE and piecewise linear models. (The output of each inverter is loaded by another inverter.) To highlight the effect of the base

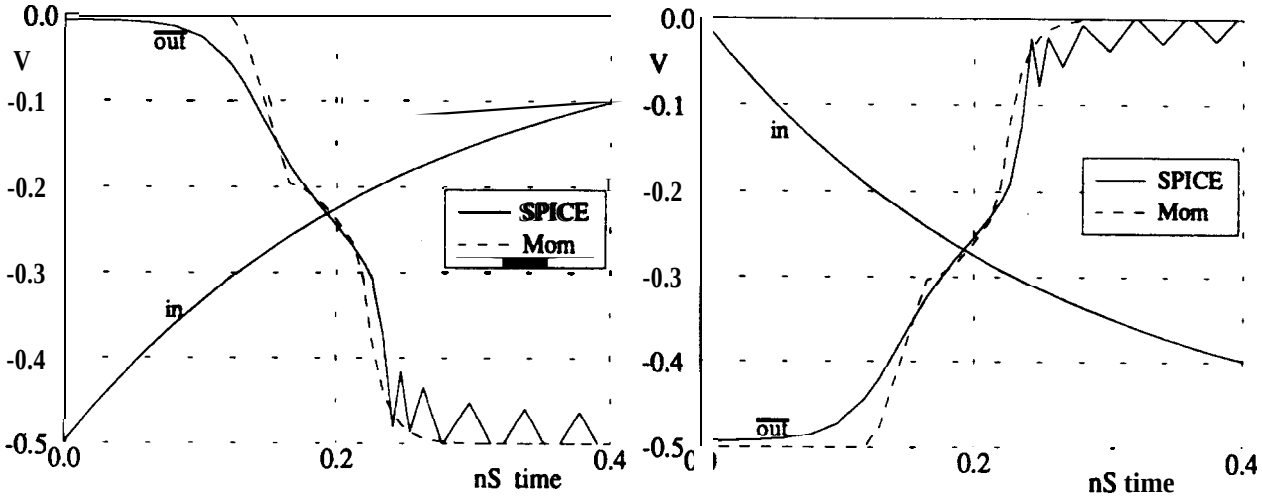


Figure C.2: Linearized Base Charge Model

charge model, all other capacitances have been set to zero. The plots reveal that the model is quite good. (The sawtooth response predicted by SPICE is an artifact of SPICE's solution method and is not indicative of the actual behavior of the model.) Although there are minor deviations between the two responses these will be averaged out once other capacitors are added to the transistor model.

Appendix D

Optimal Frequency Scaling

D.1 Algorithm

It is interesting to note that it is possible to perform "optimal" frequency scaling of moments. That is, if the moments are scaled by l/s units of time:

$$\tilde{m}_i = m_i s^i \tag{D.1}$$

then an s can be found that minimizes ratio of the largest moment's magnitude to the smallest moment's magnitude:

$$\frac{\max_i |\tilde{m}_i|}{\min_j |\tilde{m}_j|} \tag{D.2}$$

The solution is obtained by considering the logarithm of the magnitudes of the scaled moments:

$$\begin{aligned} l_i &= \log(|\tilde{m}_i|) && \text{(D.3)} \\ &= \log(|m_i s^i|) && \text{(D.4)} \\ &= \log(|m_i|) + \mathrm{i}\log(s) && \text{(D.5)} \\ &= q_i + i\sigma && \text{(D.6)} \end{aligned}$$

where we define $q_i = \log(|m_i|)$, $\sigma = \log(s)$. Then the objective is met when that value of σ is found that minimizes the maximum of all pairwise differences: $\delta_{ij} = l_i - l_j$.

However, each difference is now a *linear* function of a:

$$\delta_{i,j}(s) = q_i - q_j + (i-j)\sigma \tag{D.7}$$

Figure D.l shows a plot of **pairwise** differences. At each σ the shaded region is bounded

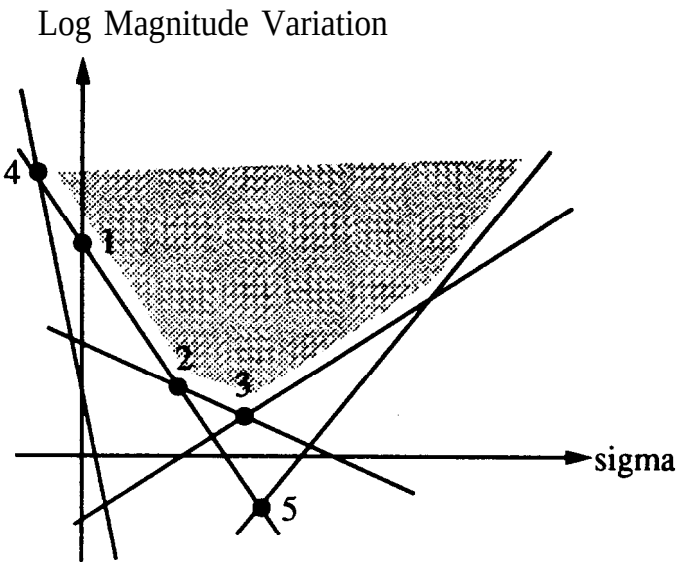


Figure D. 1: Difference Functions

by the line representing the largest difference. This problem can be solved using techniques similar in spirit to those employed by the *Simplex* Method from linear programming. Because the shaded region is *convex,* the minimum must lie on a comer point. Therefore the search can start at a point on the boundary and jump from comer point to comer point until the minimum is found.

D.1.1 Implementation

However a considerable number of parallel lines can be pruned before the search is begun. The **set** of differences: $\{\delta_{j+1,j}\}$ are parallel because they all have the term a; the differences: $\{\delta_{j,j+1}\}$ are parallel because they all have the term --a; the differences: $\delta_{j+2,j}$, are parallel because they include the term 20, etc. For each set of parallel lines only the highest line need be retained:

$$\triangle_j = \max_i (\delta_{i+j} - \delta_i) \tag{D.8}$$

If there are k moments then after all redundant parallel lines are eliminated only $2(k-1)$ will remain.

The general form of the difference functions will be:

$$\begin{vmatrix} \Delta_1(\sigma) \\ \Delta_2(\sigma) \\ \Delta_3(\sigma) \\ \vdots \\ \Delta_{k-1}(\sigma) \\ \Delta_{-1}(\sigma) \\ \Delta_{-2}(\sigma) \\ \Delta_{-3}(\sigma) \\ \vdots \\ \Delta_{-k+1} \end{vmatrix} = \begin{vmatrix} 1 \\ 2 \\ 3 \\ \vdots \\ k-1 \\ -1 \\ -2 \\ -3 \\ \vdots \\ -k+1 \end{vmatrix} \sigma + \begin{vmatrix} b_1 \\ b_2 \\ b_3 \\ \cdot \\ b_{k-1} \\ b_{-1} \\ b_{-2} \\ b_{-3} \\ \vdots \\ b_{-k+1} \end{vmatrix} \tag{D.9}$$

where b_i is the y-axis intercept of $\Delta_i(\sigma)$. The objective is to find the σ that satisfies:

$$\min_{\sigma} \left\{ \max_{j} (\Delta_j) \right\} \tag{D. 10}$$

A reasonable place to begin the search is at the highest point of intersection of the y axis with the difference functions (point 1 in the Figure). That point becomes the **current** point, and difference function passing through it becomes the current line.

Subsequent points are selected to reduce the maximum difference. Candidates for the next point are found by computing the points of intersection of the current line with all other lines. In general, the abscissa of the point of intersection between Δ_p and A, is:

$$\sigma_{cross} = -\frac{b_p - b_q}{p - q}. \tag{D.11}$$

If the current line has a positive slope, the closest point to the left of the current point is selected Otherwise the closest point to the right of the current point is selected. For our example the current line is negatively sloped so the search proceeds to point 2 (point 4 is closer but increases the maximum difference, point 5 reduces the difference but is further from point 1 than point 2). Note that it is important to stop at the closest intersection point to avoid leaving the shaded (in linear programming **terms *feasible*)** region.

Finally, the slope of the new line at the newly selected point is examined. If it has the same sign as the slope of the current line then it becomes the new current line, the intersection point becomes the new current point, and the procedure iterates. Otherwise the global minimum has been found and the optimal scaling factor is given by $e^{\sigma_{cross}}$. For our example the search continues through point 2 but terminates at point 3.

D.2 Efficiency

The biggest drawback of optimal frequency scaling is its inefficiency. If the scaling factor is computed from the ratio of two moments then frequency scaling is O(n) in the number of moments. In contrast the complexity of optimal frequency scaling is $O(n^2)$. The complexity of the pruning algorithm is $O(n^2)$ in the number of moments because the there are $n(n-1)/2$ ways of pairing n moments. The complexity of the search algorithm is also $O(n^2)$ because at each search step $2n-3$ intersection points must be considered and as many as $272-2$ comer points may have to be traversed (although the experience with linear programming indicates this unlikely). Thus when the procedure was implemented, it was found to be too expensive to justify the small improvement for the large majority of waveforms.

However, although they haven't been tried, there are a few possibilities for tuning the routine. A very expensive part of the optimal computation is taking the logarithm of each of the moments. It may be possible to efficiently approximate the logarithm by extracting the exponent bits from the machine's floating point representation. Also note that the divisions in Equation **D.11** can be turned into multiplies if the constants: $1/2, 1/3, 1/4, 1/5, \ldots 1/2n$ are **precomputed** and stored. Finally, the efficiency of the algorithm can be improved if a less demanding criterion for **optimality** is used. If instead of minimizing the ratio of the largest moment to the smallest moment, we simply minimize the largest ratio of ***adjacent*** moments, then the complexity of the algorithm reduces to $O(n)$.

Appendix E

Unstable Waveforms

Chapter 6 ignored the problem of finding the roots of waveforms containing unstable poles. In fact, unstable responses present little additional difficulty because an unstable waveform can be easily mapped into a stable waveform with the same roots. To illustrate, suppose we have the unstable waveform:

$$b(t) = k_0 + k_1 e^{\sigma_1 t} + k_2 e^{\sigma_2 t} + \cdots + k_p e^{\sigma_p t} \tag{E.1}$$

where $\sigma_1 = \alpha_1 + j\beta_1$ is assumed to be the frequency with the most positive real part. If we factor out the exponential $e^{\alpha_1 t}$ we get a new function w(t):

$$b(t) = e^{\alpha_1 t}\left(k_0 e^{-\alpha_1 t} + k_1 e^{\beta_1 t} + k_2 e^{(\sigma_2 - \alpha_1)t} + \cdots k_p e^{(\sigma_p - \alpha_1)t}\right) \tag{E.2}$$

$$= e^{\alpha_1 t} w(t) \tag{E.3}$$

Again note that *w(t)* has the same roots as *b(t)* because $e^{\alpha_1 t} \neq 0$. Furthermore, the assumption that σ_1 has the most positive real part implies that all the frequencies of *w(t)* have real parts that are less than or equal to zero. Thus w(t) is a stable waveform with roots identical to *b(t).*

Bibliography

[AHU85] Alfred V. Aho, John E. Hopcroft, and Jeffrey D. Ullman. *Data* Structures *and Algorithms*. Addison-Wesley Publishing Company, Reading, Massachusetts, 1985.

[AM88] Paolo Antognetti and Giuseppe Massobrio. *Semiconductor Device Modeling with SPICE.* McGraw-Hill Book Company, New York, 1988.

[Atk78] Kendall E. Atkinson. *An Introduction to Numerical Analysis.* John Wiley & Sons, New York, 1978.

[BL72] M. J. Bosley and F. l? Lees. A survey of simple transfer-function derivations from high-order state variable models. *Automatica,* 8:765–775, 1972.

[Bra80] Franklin H. Branin, Jr. The analysis and design of power distribution nets on LSI chips. In *IEEE International Conference on Circuits and Computers,* pages 785–790, 1980.

[Bry80] Randal E. Bryant. An algorithm for MOS logic simulation. *Lambda, the Magazine of VLSI Design,* 1(3):46–53, 1980.

[BS65] Amar G. Bose and Kenneth *N*. Stevens. *Introductory Network Theory*. Harper & Row, New York, 1965.

[CGK75] B. R. Chawla, H. K. Gummel, and P. Kozak. MOTIS-an MOS timing simulator. *IEEE Transactions on Circuits and Systems,* 22:901–909, December 1975.

[Cha88] Pak K. Chan. Signal delay in RC networks with floating capacitors. In *IEEE International Symposium on Circuits and Systems,* page get the page numbers, June 1988.

[Cha91] Pak K. Chan. Comments on "asymptotic waveform evaluation for timing analysis". *IEEE Transactions on* ***Computer-Aided Design,*** **10(8):** 1078-1079, August 1991.

[Chu88] Chomg-Yeong *Chu. Improved Models for Switch-Level Simulation.* **PhD** thesis, Stanford University, November 1988.

[CK89] Pak K. Chan and Kevin Karplus. Computing signal delay in general RC networks by tree/link partitioning. In *26th ACMIIEEE Design Automation Conference,* pages 485-490, June 1989.

[CL75] Leon 0. Chua and Pen-Min Lin. *Computer Aided Analysis* ***of*** *Electronic Circuits: Algorithms and Computational Techniques.* Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1975.

[CS84] Chin Fu Chen and Prasad Subramanium. The second generation **MOTIS** timing simulator – an efficient and accurate approach for general MOS circuits. In *IEEE International Symposium on Circuits and Systems,* pages 538-542, 1984.

[dG84] **Aart** J. de Geus. SPECS: Simulation program for electronic circuits and systems. In *IEEE International Symposium on Circuits and Systems,* pages **534–537, 1984.**

[DMHH87] Giovanni De Micheli, Hsueh Y. Hsieh, and Ibrahim Hajj. Decomposition techniques for large scale circuit analysis and simulation. In A. E. Ruehli, editor, *Circuit Analysis, Simulation* ***and Design,*** 2, chapter 7. Elsevier Science Publishers B. V. (North-Holland), 1987.

[DMNSV83] G. De Micheli, A. R. Newton, and A. Sangiovanni-Vincentelli. Symmetric displacement algorithms for the timing analysis of MOS VLSI circuits. *IEEE Transactions on Circuits and Systems,* pages 167-197, July 1983.

[DvGdG85] P. M. Dewilde, A. J. van Genderen, and A. C. de Graaf. Switch level timing simulation. In *International Conference on Computer-Aided Design,* pages 182-l 84, 1985.

[Elm48] W. Elmore. The transient response of damped linear networks with particular regard to wideband amplifiers. *Journal of Applied Physics,* 19:55–63, January 1948.

[GD85] Lance A. Glasser and Dan W. Dobberpuhl. *The Design and Analysis of VLSI Circuits.* Addison-Wesley Publishing Company, Reading, Massachusetts, 1985.

[GM88] Edwin W. Greeneich and Kevin L. McLaughlin. Analysis and characterization of bicmos for high-speed *digital* logic. *IEEE Journal of Solid State Circuits,* 23(2):558–565, April 1988.

[GP91] Nanda Gopal and Lawrence T. Pillage. Constrained approximation of dominant time-constant(s) in RC-circuit delay models. Technical Report UT-CERC-TR-LTP91-01, The University of Texas at Austin, January 1991.

[HJ83] David A. Hodges and Horace G. Jackson. *Analysis and Design of Digital Integrated Circuits.* McGraw-Hill Book Company, New York, 1983.

[Hor83] Mark Alan Horowitz. *Timing Models for MOS Circuits.* PhD thesis, Stanford University, December 1983.

[Hou70] A. S. Householder. *The Numerical Treatment of a Single Nonlinear Equation.* McGraw-Hill Book Company, New York, 1970.

[HS87] Ibrahim N. Hajj and Daniel Saab. Switch-level logic simulation of digital bipolar circuits. *IEEE Transactions on Computer-Aided Design,* 6(2):25 1–*258,* March 1987.

[HSV81] Gary D. Hachtel and Alberto L. Sangiovanni-Vincentelli. A survey of third-generation simulation techniques. *Proceedings of the IEEE,* 69(10): 1264-1280, October 1981.

[Hua90] Xiaoli Huang. *Padé Approximation of Linear(ized) Circuit Responses.* PhD thesis, Carnegie Mellon University, December 1990.

[Jou83] N. P. Jouppi. TV: An nMOS timing analyzer. In *3rd Caltech Conference on VLSI,* pages 7 l-85, March 1983.

[KAHS88] Russell Kao, Bob Alverson, Mark Horowitz, and Don Stark. Bisim: A simulator for custom ECL circuits. In *International Conference on Computer-Aided Design,* pages 6265, November 1988.

[KKSN84] Young H. Kim, J. E. Kleckner, R. A. Saleh, and A. R. Newton. Electrical-logic simulation. In *International Conference on Computer-Aided Design,* pages 7-9, November 1984.

[Kro39] G. Kron. *Tensor Analysis of Networks.* Wiley, 1939.

[KSM+78] Kuniyasu Kawarada, Masao Suzuki, Hisakazu Mukai, Kazuhiro Toyoda, and Yoshisuke Kondo. A fast 7.5ns access lk-bit RAM for cache-memory systems. *IEEE Journal of Solid State Circuits,* sc-13(5):656–663, October 1978.

[Kun86] Kenneth S. Kundert. Sparse matrix techniques. In A. E. Ruehli, editor, *Circuit Analysis, Simulation and Design,* chapter 6. Elsevier Science Publishers B. V. (North-Holland), 1986.

[LM84] Tzu-mu Lin and Carver A. Mead. Signal delay in general RC networks. *IEEE Transactions on Computer-Aided Design,* 3(4):331–349, October 1984.

[LSV82] E. Lelarasmee and A. Sangiovanni-Vincentelli. Relax: A new circuit simulator for large scale MOS integrated circuits. In *19th ACM/IEEE Design Automation Conference,* pages 682–690, June 1982.

[McC89] Steven Paul McCormick. *Modeling and Simulation of VLSI Interconnections with Moments.* PhD thesis, Massachusetts Institute of Technology, June 1989.

[MK77] Richard S. Muller and Theodore I. Kamins. *Device Electronics for Integrated Circuits.* John Wiley & Sons, Inc, New York, 1977.

[MMS+80] Osamu Minato, Toshiaki Masuhara, Toshio Sasaki, Hideaki Nakamura, Yoshio Sakai, Tokumasa Yasui, and Kiyofumi Uchibori. 2k x 8 bit Hi-CMOS *static* RAM's. *IEEE Journal of Solid State Circuits,* sc-15(4):656–660, August 1980.

[Nag75] L. Nagel. SPICE2: A computer program to simulate semiconductor circuits. Technical Report ERL-520, University of California, Berkeley, May 1975.

[New79] Richard Newton. Techniques for the simulation of large-scale integrated circuits. *IEEE Transactions on Circuits and Systems*, 26(9):741–749, September 1979.

[Pay56] H. M. Paynter. On an analogy between stochastic processes and monotone dynamic systems. In G. Müller, editor, *Regefungstechnik. Moderne Theorien und ihre Verwendbarkeit*, pages 243-250. R. Oldenbourg, 1956.

[PD90] Lawrence T. Pillage and Santanu Dutta. A path tracing algorithm for asymptotic waveform evaluation of lumped RLC circuit delay models. In *ACM Workshop on Timing Issues in the Specification and Synthesis of Digital Systems,* August 1990.

[Pil89] Lawrence T. Pillage. *Asymptotic Waveform Evaluation for Timing Analysis.* PhD thesis, Carnegie Mellon University, April 1989.

[PR81] Paul Penfield, Jr. and Jorge Rubinstein. Signal delay in RC tree networks. In *18th ACMIIEEE Design Automation Conference,* pages 6 13-6 17, June 198 1.

[PR90] Lawrence T. Pillage and Ronald A. Rohrer. Asymptotic waveform evaluation for timing analysis. *IEEE Transactions on Computer-Aided Design,* 9(4):352–366, April 1990.

[Put82] R. Putatunda. Auto-Delay: A program for automatic calculation of delays in LSI/VLSI chips. In *20th ACMIIEEE Design Automation Conference,* pages 616-621, June 1982.

[RGP91] Curtis L. Ratzlaff, Nanda Gopal, and Lawrence T. pillage. RICE: Rapid interconnect circuit evaluator. In *28th ACM/IEEE Design Automation Conference,* pages 555-560, June 1991.

[Roh88] R. A. Rohrer. Circuit partitioning simplified. *IEEE Transactions on Circuits and Systems,* 35(1):2–5, January 1988.

[RSVH79] N. B. Guy Rabbat, Alberto L. Sangiovanni-Vincentelli, and Hsueh Y. Hsieh. A multilevel newton algorithm with macromodeling and latency for the analysis of large-scale nonlinear circuits in the time domain. *IEEE Transactions on Circuits and Systems,* 26(9):733–741, September 1979.

[RT85a] Arvind Raghunathan and Clark D. Thompson. Signal delay in rc trees with charge sharing or leakage. In *19th Asilomar Conference on Circuits, Systems, and Computers,* pages 557-56 1, November 1985.

[RT85b] Vasant B. Rao and Timothy N. Trick. Switch-level timing simulation of MOS VLSI circuits. In *IEEE International Symposium on Circuits and Systems,* pages 229–232, 1985.

[RV87] Ronald A. Rohrer and Chandramouli Visweswariah. SPECS2: An integrated circuit timing simulator. In *International Conference on Computer-Aided Design,* pages 94-97, November 1987.

[RVB88] Genhong Ruan, Jiri Vlach, and James A. Barby. Current-limited switch-level timing simulator for MOS logic networks. *IEEE Transactions on Computer-Aided Design,* 7(6):659–667, June 1988.

[Sal83] R. Saleh. Iterated timing analysis and SPLICE1. Master's thesis, University of California, Berkeley, 1983.

[Sch85] Thomas J. Schaefer. A transistor-level logic-with-timing simulator for MOS circuits. *In 22ndACMIIEEE Design Automation Conference,* pages 762-765, 1985.

[SH89] Arturo Salz and Mark Horowitz. Irsim: An incremental MOS switch-level simulator. In *26th ACM/IEEE Design Automation Conference,* pages 173–178, June 1989.

[SH90] Don Stark and Mark Horowitz. Techniques for calculation currents and voltages in VLSI power networks. *IEEE Transactions on Computer-Aided Design,* 9(2): 126-l 32, 1990.

[SNGR91] Alexander D. Stein, Tuyen V. Nguyen, Binay J. George, and Ronald A. Rohrer. ADAPTS: A digital transient simulation strategy for integrated circuits. In *28th ACM/IEEE Design Automation Conference,* pages 2631, June 1991.

[SSD72] Karl U. Stein, Aame Sihling, and Elko Doering. Storage array and sense/refresh circuit for single transistor memory cells. *IEEE Journal of Solid State Circuits,* sc-7(5):336–340, October 1972.

[SVCC77] Alberto Sangiovanni-Vincentelli, Li-Kuan Chen, and Leon 0. Chua. A new tearing approach – node-tearing nodal analysis. In *IEEE International Symposium on Circuits and Systems,* pages 143–147, 1977.

[SYH88] D. G. Saab, A. T. Yang, and I. N. Hajj. Delay modeling and timing of bipolar digital circuits. In *25th ACMIIEEE Design Automation Conference,* pages 288-293, June 1988.

[SZ87] Chuanjin Shi and Kaihe Zhang. A robust approach for timing verification. In *International Conference on Computer-Aided Design,* pages 5659, November 1987.

[Ter83] *C.* J. Terman. *Simulation Tools for Digital LSI Design.* PhD thesis, Massachusetts Institute of Technology, September 1983.

[vB87] W. M. G. van Bokhoven. Piecewise linear analysis and simulation. In A. E. Ruehli, editor, *Circuit Analysis, Simulation and Design,* 2, chapter 10. Elsevier Science Publishers B. V. (North-Holland), 1987.

[VFR90] Chandramouli Visweswariah, Peter Feldmann, and Ronald A. Rohrer. Incorporation of inductors in piecewise approximate circuit simulation. In *International Conference on Computer-Aided Design,* pages 162-165, November 1990.

[WJM+73] W. T. Weeks, A. J. Jiminez, G. W. Mahoney, D. Mehta, H. Qassemzadeh, and T. R Scott. Algorithms for ASTAP-a network analysis program. *IEEE Transactions on Circuit Theory,* 20:628–634, November 1973.

[WR83] Fred Walczyk and Jorge Rubinstein. A merged CMOS/bipolar VLSI process. In *International Electron Devices Meeting Technical Digest,* pages 59-62, December 1983.

[Wu76] Felix F. Wu. Solution of large-scale networks by tearing. *IEEE Transactions on Circuits and Systems,* 23(12):706–713, December 1976.

[Wya85] J. L. Wyatt. Signal delay in RC mesh networks. *IEEE Transactions on Circuits and System,* 32(5):507–5 10, May 1985.

[YHT80] P. Yang, I. N. Hajj, and T. N. Trick. SLATE: A circuit simulation program with latency exploitation and node tearing. In *IEEE International Conference on Circuits and Computers,* pages 353-355, October 1980.

[YHW88] Tsen-Shau Yang, Mark A. Horowitz, and Bruce Wooley. A 4ns 4kx 1 -bit two-port bicmos sram. *IEEE Journal of Solid State Circuits,* 23(5):1030–1040, October 1988.

G-731 V. Zakian. Simplification of linear time-invariant systems by moment approximants. *International Journal of Control,* 18(3):455–460, 1973.