An Algorithm for Simulating Power/Ground Networks using Padé Approximants and its Symbolic Implementation

Jatan C. Shah, Ahmed A. Younis, Sachin S. Sapatnekar and Marwan M. Hassoun [*†]

July 9, 1998

Abstract

This paper presents an event-driven algorithm and its symbolic implementation for the analysis of power and ground bus networks. The algorithm uses frequency domain techniques and moment matching approaches based on Padé approximants to estimate the transfer function at each node in the P/G network. Afterwards, the transient waveforms are extracted for each node. The process requires repetitive simulation of a linear and time-variant (from one time event to the next) circuit model for the P/G network which is the reason a symbolic implementation was produced. The P/G network is modeled by a hierarchical combination of mesh and tree structures that are composed of a collection of RC-π-segments and pulldown (or pullup) switches. The switches are symbolically represented by Boolean variables and a compiled symbolic code is generated only once for each P/G network. The transient waveforms are then produced by repetitive evaluation of the symbolic output. The results show that the symbolic implementation is an order of magnitude faster, with reasonably good accuracy, than using a traditional analog circuit simulator like SPICE.

Keywords: VLSI, power/ground networks, Padé approximants, symbolic analysis.

[*]J. C. Shah, A. A. Younis and M. M. Hassoun are with the Department of Electrical and Computer Engineering at Iowa State University, Ames, IA 50011, USA. S. S. Sapatnekar is with the Department of Electrical and Computer Engineering, University of Minnesota, Minneapolis, MN 55455, USA. Contact email: sachin@ece.umn.edu.

[†]This work is supported in part by NSF through grant MIPS-9502556 and by a grant from Texas Instruments, Inc.

1 Introduction

Switching activity in VLSI circuits introduces transient currents in the power and ground (P/G) buses. These currents produce two major effects: voltage drops across the buses and high current densities leading to electromigration of metal atoms. Excess voltage drops can manifest themselves as glitches on the P/G buses and can cause unacceptable circuit performance either due to reduced switching speed or by causing unwanted signal transitions. Electromigration places a limit on the useful lifetime of a chip, and if not properly accounted for, can present serious reliability problems. One mechanism that is frequently used by designers to overcome both of these problems is to selectively widen the wires in the P/G network.

Even for a fixed set of P/G network parameters (such as wire widths), the problem of accurately identifying the worst case scenario is a difficult one and requires repeated simulations to determine the worst-case set of events that cause the largest voltage drop and/or susceptibility to electromigration. For the diagnosis of these problems, it is essential to find fast but accurate techniques to simulate the network repeatedly. Moreover, the procedure of optimizing P/G buses to control voltage drops and electromigration problems requires repeated simulations under different sets of parameters.

A procedure, like the one presented herein, that requires repeated simulations is a good candidate for the application of symbolic analysis. This paper applies symbolic analysis using moment-matching approaches based on Padé approximants to speed up the computation. Our work produces a compiled code for a given circuit structure and enables a designer to run this compiled code with various values of the circuit parameters (namely, any list of switching events and any set of widths of the wire segments).

Traditionally, symbolic circuit analysis methods have focused on finding transfer function solutions for general circuits. Several symbolic simulators exist, such as SNAP [1], FLOWUP [2], ISAAC [3], ASAP [4], BRAINS [5], SYNAP [6], SSPICE [7], MASSAP [8], SCAPP [9], EASY [10], and SAPWIN [11], in addition to programs in [12, 13]. These programs can be used to generate the initial transfer functions for the P/G networks. However, the P/G network, as modeled herein, is a circuit with a specific mesh/tree topology and a specific transfer function construct. An application specific symbolic analyzer was developed in order to eliminate the overhead associated with the use

any of the above mentioned general symbolic simulators and provide an efficient solution mechanism for this specific problem. Another major difference between the general purpose program and the implementation here is in the goal and usage of the transfer function approximations. While some of the general purpose programs use approximation algorithms to produce a "visually" interpretable transfer function [14–17], these approximations are centered around specific component values and are not necessary optimal for accurate numerical evaluations over a large range of component values. The goal of the symbolic analysis in this paper is to produce an approximation that is valid over a large range of component values and one that is optimal for the repetitive evaluation process needed to produce the ultimate numeric solution. The "visual" nature of the final transfer function is of no value here due to the large sizes of P/G networks. Hence, a sequence of expressions approach was used to generate the symbolic code in order to maintain the accuracy of the functions with a linear growth in the number of arithmetic evaluations in the generated code [9, 13]. A further difference between general purpose symbolic analysis programs and the algorithm reported herein is in the way that the time-variant P/G network is handled. General symbolic analysis programs operate on fixed topology (time-invariant) circuits. Our work provides the ability to include Boolean switches that reflect changes in the P/G circuit topology between events.

The moment-matching techniques for circuit analysis based on Padé approximants, which are used in this work, gained popularity after the publication of the Asymptotic Waveform Evaluation (AWE) approach [18]. Unlike traditional approaches such as SPICE that perform a full numerical simulation of the circuit, moment-matching approaches view the circuit as a high order control system. Realizing that the control system is dominated by a small number of poles for many applications, these methods approximate the exact transfer function of the circuit by a reduced order transfer function that captures the effects of these dominant poles. While no theoretical guarantees of accuracy or stability exist, it has been seen practically that moment-matching approaches have been hugely successful in approximating large interconnect systems [19].

The problem of P/G bus simulation is well suited to the application of moment-matching methods. Due to the presence of a large number of switches changing state at different times, the voltage waveforms at any given node can assume a very complex shape. Consequently the analysis of these waveforms is often a computationally intensive process. Previous work in [20, 21] has used AWE macromodels in optimizing supply networks using a coarse grid for distribution, with

consideration of substrate effects. However, this approach was used only to design the uppermost distribution network and not the entire network.

Various algorithms and simplified device models for power/ground buses that offer faster but less accurate results have been explored in the past. Some of these techniques utilize the fact that most P/G networks tend to be tree-like structures so as to allow the use of path tracing algorithms for efficiency [22,23]; these techniques assume resistance-only models for the network. The efficiency is also improved by using pre-characterized current models for gates [24–26]. In addition to these methods, which relied on the user to specify the worst-case pattern, some pattern independent approaches were proposed in [27–29]. The proposed approach presents a fast yet accurate simulation technique that relies on designer input for the worst case switching pattern(s). This method incorporates the effect of wire capacitances and decoupling capacitances in the network and the fact that these could have nonzero initial conditions during simulation. For efficient simulation of such situations, we present an approach that augments the path-tracing techniques used in, for example, [30,31] to incorporate the effects of nonzero initial conditions without explicitly considering the circuit as a multiport network with a source corresponding to each initial condition.

2 Modeling P/G buses

The procedure for the design of power buses and ground buses that convey the V_{dd} and GND signals, respectively, across the chip are essentially symmetric problems. In other words, it is enough to develop a solution technique for one of these, and the analysis technique for the other will be performed in a similar manner. This is under the assumption that the performance of the P/G network is dominated by the charge/discharge current and the short-circuit currents have negligible effects. We do not consider the effect of the substrate and the package in our model. The proposed model can incorporate the effects of wire widths and decoupling capacitances for any given topology that follows the general style of an overlying mesh with underlying trees.

Due to the symmetry described above, we generically refer to the problem as the P/G bus problem. In our descriptions, we will chiefly focus on the design of ground buses, realizing that power buses may be designed in like fashion.

In large integrated circuits, the P/G buses may support a number of switching elements,

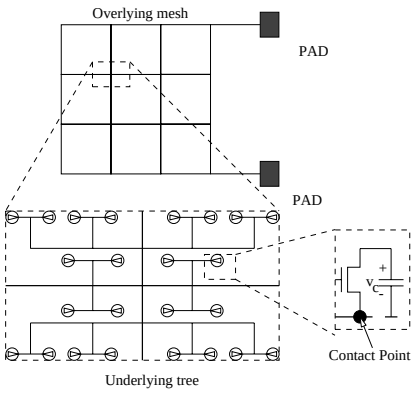

Figure 1: Typical ground bus topology

each of which draws current from the power bus, or injects current into the ground buses during transitions. Hence P/G buses are best modeled as a network of wire segments with one or more switches connected to each *contact point*. While tree-based P/G topologies are easy to analyze, their performance in terms of avoiding voltage drop and electromigration is not as good as mesh topologies. On the other hand, dense mesh topologies occupy large routing areas. An approach that is frequently used, which combines the advantages of both topologies, is to build an overlying mesh over the area of chip to distribute the P/G signals, and to feed these signals locally using underlying local tree structures, as shown in Figure 1.

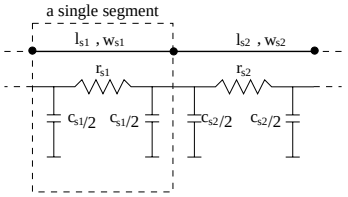

Figure 2: Wire segment model for P/G buses

Each wire on the mesh or tree structure in a P/G bus can be modeled as a set of connected segments, with each segment modeled using lumped RC parameters as a L-element, π-element or T-element [32]. In this work the individual segments are modeled as π-elements as shown in Figure 2, where

$$r_s = \rho l_s / w_s$$

$$c_s = (\beta w_s + \alpha) l_s$$

where l_s and w_s are the length and the width of the segment, and the parameters ρ, β and α are the sheet resistance per square, sheet capacitance per square and the fringe capacitance per unit length of the metal layer that is being used for routing the P/G buses. The segments are connected at contact points (nodes). The size of the individual segments is decided based on the accuracy desired, and also the location of the *contact points*. Specifically, a contact point must always be located at the end of one or more segments.

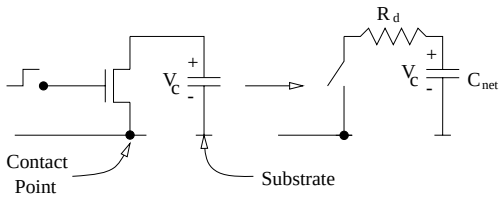

Figure 3: Switch Model for Ground Net

To model the current drawn by gates, we first reduce each gate to an equivalent inverter whose NMOS transistor represents the worst case pull-down, while the PMOS transistor represents the worst case pull-up of the gate. For the case of the ground net (the power net can be handled similarly), the pull-down is modeled as a switch as shown in Figure 3. When turned on, this switch connects a charged capacitor to the ground net, thereby injecting a discharge current through it. The switches are connected to the ground net at the prespecified contact points, and are controlled by a Boolean control variable that determines whether the switch is on or off. The switch is modeled as a zero resistance when turned on, and zero conductance when turned off.

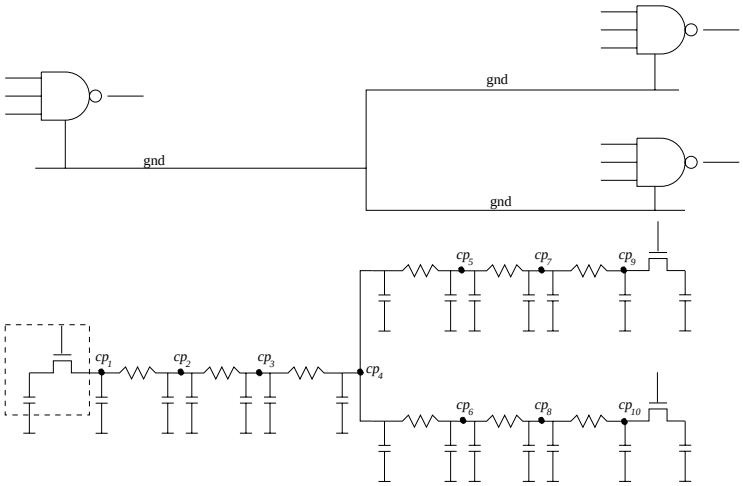

Figure 4: Model for Ground Net

Using the above models for switches and the segments, the ground bus can be represented by a network of π-elements connected between a pair of contact points, and a set of switches connected to the specified contact points. This is shown in Figure 4. The switches are assumed to be turned on or off at various time points, as described by a user-specified event list. The time points are graphically shown in Figure 5, and information on which a specific switch is being turned on or off at each time instant (not shown in the figure) is also specified by the user.

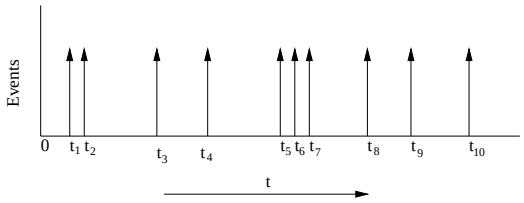

Figure 5: Switching activity in a circuit

3 Overview of AWE

The Asymptotic Waveform Evaluation method uses Padé approximants to approximate a high order system by a low order transfer function that captures the dominant poles of the system and their residues. The basic AWE algorithm [18] involves the computation of the first few terms of the Maclaurin expansion of the transfer function of the system, and then matches these terms with the coefficients of a strictly proper reduced order n-pole rational function [19].

The remainder of this section will summarize the AWE procedure; for further details, the reader is referred to [33]. We begin with a modified nodal equation description for a circuit of the type:

$$C\dot{\mathbf{x}}(t) + G\mathbf{x}(t) = \mathbf{e}(t) \tag{1}$$

where $C, G \in \mathbf{R}^{n\times n}$, $\mathbf{x}, \mathbf{e} \in \mathbf{R}^n$, where n is the number of modified nodal variables in the circuit. To begin with, we will assume that the circuit has zero initial conditions and a single excitation; our work shows how this assumption can be relaxed to handle circuits with nonzero initial conditions. Without loss of generality, we will also assume that the excitation is applied at node 1.

3.1 Moment generation

The first step involves the generation of the moments, i.e., the terms in the Maclaurin series expansion of the system transfer function. Setting $\mathbf{e}(t) = [\delta(t), 0, 0, \cdots, 0]^T$, and taking the Laplace transform of Equation (1),

$$(G + sC)\mathbf{X}(s) = \mathbf{E}(s) \tag{2}$$

where $\mathbf{E}(s) = [1, 0, 0, \cdots, 0]^T$, and $\mathbf{X}(s)$ represents the transfer function from the single input to all the nodes in the circuit. We expand $\mathbf{X}(s)$ in the form of a Maclaurin series as:

$$\mathbf{X}(s) = \mathbf{m}_0 + \mathbf{m}_1 s + \mathbf{m}_2 s^2 + \cdots = \sum_{i=0}^{\infty} \mathbf{m}_i s^i \tag{3}$$

Substituting Equation (3) in Equation (2), and matching the coefficients of the powers of s, results in a set of linear equations for each moment vector $\mathbf{m}_i$:

$$\begin{aligned} G\mathbf{m}_0 &= \mathbf{E} \\ G\mathbf{m}_1 &= -C\mathbf{m}_0 \\ &\vdots \\ G\mathbf{m}_i &= -C\mathbf{m}_{i-1} \end{aligned} \tag{4}$$

It should be noted that G can be LU-decomposed once to solve for $\mathbf{m}_0$ and the higher order moment vectors can then be generated by forward and backward substitutions.

3.2 Generating system approximations

The form of a one-input-one-output transfer function having Q poles can be described as

$$H(s) = \frac{a_0 + a_1 s + a_2 s^2 + \cdots + a_P s^P}{1 + b_1 s + b_2 s^2 + b_3 s^3 + \cdots + b_Q s^Q} \tag{5}$$

Note that for real circuits, $P \leq Q$. In large networks, Q can be of the order of hundreds or thousands, and generating all the Q poles may become prohibitively expensive. It is, however, observed that for most practical electrical networks it is sufficient to extract a very small number

of dominant poles, while completely ignoring the remaining poles, in order to capture the transient behavior of the network. Further, it can be shown that by using a small number of moments generated, a low-order M-zero, N-pole ($M \leq N$) approximation to Equation (5) can be constructed in the form

$$\tilde{H}(s) = \frac{A_0 + A_1 s + A_2 s^2 + \cdots + A_M s^M}{1 + B_1 s + B_2 s^2 + B_3 s^3 + \cdots + B_N s^N} \quad (6)$$

$$= \sum_{j=1}^{N} \frac{\hat{k}_j}{s - \hat{p}_j} \quad (7)$$

where the poles $\hat{p}_j$ approximate the set of dominant poles and $\hat{k}_j$ their residues. This is referred to as an $[M/N]$ Padé approximant. The coefficients of this reduced order approximation can be computed by matching them with the moments of the original transfer function. The procedure for calculating the poles and residues for a general circuit is described in [18, 33].

4 Outline of the Solution

The input to the problem is a given P/G network topology and various technology parameters. The P/G network is modeled by RC sections as described is Section 2, with switches connected to the tree corresponding to gates that make transitions. The switches undergo state changes at pre-defined time points and the objective is to compute the time domain voltage waveform at each node for the specified time period. The procedure can be divided into two steps that are executed every time one or more switch turns on or off:

(1) A set of equations of the form

$$(G + sC)\mathbf{V}(s) = \mathbf{J}(s) \quad (8)$$

is solved. An efficient path tracing algorithm that exploits the structure of the P/G network is used to obtain V(s), a Maclaurin series approximation in the frequency domain to the actual voltage.

(2) Given V(s) at every node, the time domain response is then computed by approximating V(s) with a rational Padé approximation.

We will also incorporate the presence of nonzero initial conditions in the P/G network.

The objective here is to produce a compiled code containing the symbolic expressions that describe the circuit topology. The symbolic expressions are generated in sequence of expressions format in order to minimize the number of arithmetic operations needed for the code evaluation. It has been shown that for linear circuits, which is the case here, the number of symbolic variables grow linearly with the size of the circuit if a hierarchical sequence of expressions format is used in contrast to an exponential growth if a single expression transfer function is generated [9], [13]. Once this code is generated, the user may specify a set of parameters (any event list and any set of wire widths) and execute this code for those specified parameters. Due to the fact that the actual analysis is only performed once on the network and the repetitive numerical evaluations are performed using the symbolic code, this method is faster than repetitive full SPICE simulations on the same network. The results presented in Section 6 support this claim.

We will now describe the algorithm that is embedded within the compiled code. The algorithm is event-driven, with events specified at the time points at which one or more switch changes state. The events are user-specified, and the first task of the algorithm is to construct a list of time intervals during which there is no switching activity. All switches that switch at the same time point are grouped together and placed at the beginning of a new interval that is inserted in the interval list. This new interval now spans the period between the specified time point of the current set of transitioning switches to the next switching activity in the circuit. For instance, for the switching activity depicted in Figure 5, the interval list is $\{(0, t_1), (t_1, t_2), (t_2, t_3) \cdots\}$.

The current injected in the ground net of a CMOS circuit is essentially a transient current that is the effect of the load capacitances discharging into the ground net through the pull-down logic. The peak of this current is therefore directly related to the initial voltage across the load capacitance when the pull-down comes on. With multiple switching activities occurring at different instants of time, it is essential to take into consideration the initial conditions on all the load capacitances which inject current in the ground net. A similar argument holds for the power bus except that the current injected is negative, and so is the initial condition with respect to the power supply voltage. In addition, the capacitance of the P/G buses can retain a voltage above the actual ground, which should be taken into consideration.

The simulation now proceeds one interval at a time, starting from the first interval until the last, updating the corresponding switch states while moving from one interval to the next. The

procedure is best explained through the example of Figure 5, where time points $t_1, \cdots, t_{10}$ are the points at which one or more nodes in the P/G net change states. The procedure begins at time 0, with all the initial conditions set to 0, and each switch is evaluated to determine if its state has changed. The interval $(0, t_1)$ is then processed to compute the transient voltages at all nodes during that interval. At t_1, the final conditions due to the transient response in the previous interval are computed and are used as the initial conditions for the next interval. The new state of each switch that changes states is set, and the simulation progresses with these updated switch states and computed initial conditions for the interval (t_1, t_2). The procedure continues until all the intervals are processed.

The outline of the algorithm is given in Figure 6. S is the set of switches connected to the P/G bus topology T, and $E = \{e_1, \cdots, e_m\}$ is the set of all the segments in T, as shown in Figure 4. $I = \{i_k \mid k = 1 \cdots I_{max}\}$ is the interval list, consisting of a total of I_{max} intervals, where i_k is the k^{th} interval. Associated with each interval i_k of time span t_k is a subset of switches $s_k \subseteq S$ that change at the beginning of i_k to the new state $p_k = \{State(j) \; \forall \; j \in \; s_k\}$.

```
BEGIN ALGORITHM Vdrop()
    Read in I = IntervalList();
    for each n ∈ Nodes(T)
        SetInitialCondition(n,0)
    for each i_k ∈ I in chronological order{
        for each j ∈ s_k and state(j) ∈ p_k{
            SetSwitch(T, j, state(j))
            PropagateActivity(T, j, state(j));
        }
        for each n ∈ Nodes(T) {
            f_n(t) = TimeDomainResponse(n);
            SetInitialCondition(n,f_n(t_k))
        }
    }
END ALGORITHM Vdrop()
```

Figure 6: Voltage Estimation Algorithm

5 Simulation of one interval

In this work, we have assumed a P/G topology consisting of a mesh that globally covers the circuit and drives a set of trees for local distribution. This idea was described earlier in Figure 1, and the algorithm proceeds in three stages:

Reduction of the trees The trees are reduced to Padé-approximated Norton equivalents. This is shown in Figure 7b, where each of the subtrees T_1 through T_6 from Figure 7a have been reduced to their equivalent admittances and current sources.

Solving the mesh The modified nodal formulation is then applied to solve for the Padé approximation for the voltage at each node in the mesh.

Propagating waveforms down the trees The computed voltages at the mesh nodes are propagated down the tree to compute all voltages within the tree.

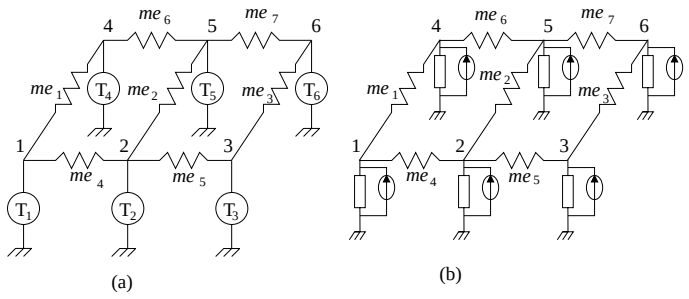

Figure 7: Mesh topology with local trees

5.1 Reduction of the trees

This section discusses in detail the computation of moments of the voltage at any given node by recursively reducing the subtrees rooted at the given node n to an admittance $Y_{T_i}(n)$ and a current source $J_{T_i}(n)$, where $Y_{T_i}(n)$ is the equivalent admittance of the subtree T_i, as seen from node n, to the ground, and $J_{T_i}(n)$ is the combined effect of all the switch elements in the given subtree T_i. The equivalent current sources and admittances are computed for every node in linear time using a path tracing algorithm that will be described shortly. Note that the Y_{T_i}'s and J_{T_i}'s are symbolic expressions in terms of the Boolean variables that control each switch (whose values are set at run-time by the user-specified event list) and the wire widths.

The algorithm proceeds from the leaf-nodes, which are the contact points at the terminating segments of the ground net. A terminating segment e_t of a tree T is a segment that connects exactly one contact point (leaf node) to the rest of the ground net. That is, a leaf node could have zero or more switches but exactly one segment connected to it, as shown in Figure 8. The admittance $Y(n)$, which is the equivalent downstream admittance at node n as seen from edge e_t, is the sum total admittance of all the switches connected to n, and the current source $J(n)$ represents the equivalent current source corresponding to the total effect of all the switches connected to node n.

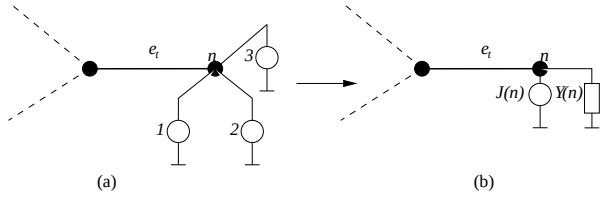

Figure 8: Computing equivalent devices at terminating segments

The equivalent devices $J(n)$ and $Y(n)$ can be computed using Norton's theorem, which states that any two port network can be replaced by an equivalent current source which is the short circuit current that flows through the two ports, and an equivalent admittance, which is the admittance of the network as seen from the ports, with all the voltage sources shorted, initial conditions set to zero, and current sources removed from the network. The equivalent current source of a single switch i with resistance R_d, connecting a load capacitance C_L to the network, is given by:

$$J^i = b \cdot \frac{V_c}{s} \frac{sC_L}{1 + sR_dC_L} \tag{9}$$

where V_c is the initial voltage on the capacitor C_L at the start of the given interval, and b is a Boolean variable whose value is 1 if the switch is closed and 0 otherwise. The pole at the origin in Equation (9) indicates that the capacitance C_L with an initial voltage V_c is modeled as a voltage source $V_c \cdot u(t)$ in series with a capacitance C_L having zero initial conditions, where $u(t)$ is a unit step input, implying that the initial condition appears in the current expression only when the switch closes. Generating moments for $J(n)$ using the Maclaurin expansion of $\frac{1}{1+sR_dC_L}$, we get

$$J^i(n) = b \cdot V_c(C_L - R_d{C^2}_L s + {R^2}_d{C^3}_L s^2 + \cdots) \tag{10}$$

The admittance of a closed switch is equal to the combined admittance of the driving resistance R_d and the load capacitance C_L:

$$Y^i = b \cdot \frac{C_L s}{1 + R_d C_L s} \tag{11}$$

The moments of Y^i can be computed using a Maclaurin expansion as

$$Y^i = b \cdot (C_L s - R_d {C^2}_L s^2 + {R^2}_d {C^3}_L s^3 + \cdots) \tag{12}$$

The total downstream admittance at node n as seen from segment e_t is,

$$Y(n) = \sum Y^i \; \forall \; i \; \in \text{Switches at } n \tag{13}$$

and the equivalent current source at n is,

$$J(n) = \sum J^i \; \forall \; i \; \in \; \text{Switches at } n \tag{14}$$

In the next step, the effect of the equivalent current sources is propagated to every node in the network. To illustrate the recursive step of moving up the tree, consider a terminating segment e_t in the ground network, connecting a single contact point u to the rest of the network, rooted at v. This is depicted in Figure 9. Given that the equivalent admittance and current source of all the switches connected to the terminating side of e_t are given by Equation (13) and Equation (14), as applied at node u, the admittance and the equivalent current source, as seen into the edge e_t from node v is computed as follows. The equivalent admittance at v can be computed based on the method used in REX [34], an algorithm for handling trees with zero initial conditions.

$$\begin{aligned} Y(v) &= \frac{1}{R(e) + \frac{1}{(Y_0(u) + (Y_1(u) + C(e)/2)s + \cdots + Y_{2n-1}(u) s^{2n-1})} + \frac{C(e)}{2}} \\ &= \frac{Y_0(u) + (Y_1(u) + C(e)/2)s + \cdots + Y_{2n-1} s^{2n-1}}{(1 + R(e)Y_0(u)) + R(e)(Y_1(u) + C(e)/2)s + \cdots + R(e)Y_{2n} s^{2n}} + \frac{C(e)}{2} \\ &= Y_0(v) + Y_1(v)s + Y_2(v)s^2 + \cdots + Y_k(v)s^k + \cdots \end{aligned} \tag{15}$$

where $Y_k(v)$ is given by

$$Y_k(v) = \frac{Y_k(u) - \sum_{i=1}^{k} R(e) Y_{k-i}(v) Y_i(u)}{1 + R(e) Y_0(u)} \tag{16}$$

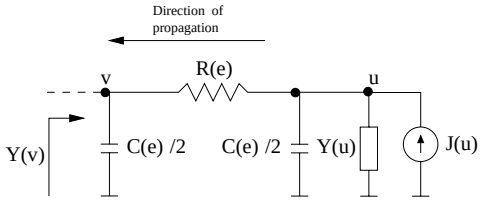

Figure 9: Reduction to equivalent current source and admittance

Unlike REX, we must deal with downstream current sources due to the presence of initial conditions. To find $J(v)$, we apply Norton's theorem by short-circuiting the node v to ground and calculating the corresponding current. Proceeding in a manner similar to the admittance computation, it can be shown that $J(v)$, which is equal to the short circuit current, is given by

$$J_k(v) = J_0(v) + J_1(v)s + J_2(v)s^2 + \cdots + J_k(v)s^k + \cdots \tag{17}$$

where

$$J_k(v) = \frac{J_k(u) - \sum_{i=1}^{k} R(e) J_{k-i}(v) Y_i(u)}{1 + R(e) Y_0(u)} \tag{18}$$

Thus, we say that the equivalent admittance and current source have been *propagated* from node u to v. It should be noted that although we have so far only considered the case where switches are connected to leaf-nodes, switches can, in general, be connected to any node. In cases where a node u has more then one segment and/or multiple switches connected, the switches are processed in a similar manner, except that the admittance and current source can be propagated when all downstream edges have been processed.

5.2 Solving the mesh

In general a mesh M comprises p segments $me_1, \cdots, me_p$, and q mesh nodes $mn_1, \cdots, mn_q$, with a local tree T_{mn_i} rooted at each node mn_i, where the prefix m on each of these indicates that the node or edge lies on the mesh. The problem is to compute all the mesh node voltages $V_{mn_i}, i = 1, \cdots, q$, and then use the solution to compute the voltages at all the internal nodes within the subtrees. Using the technique presented above, the trees T_{mn_i} can be reduced to an equivalent admittance

$Y(mn_i)$ and currents source $J(mn_i)$.

Starting from Equation (1), the expression is modified to include higher order moments of $Y(mn_i)$ and $J(mn_i)$:

$$(G + Y)\mathbf{V} = \mathbf{J} \tag{19}$$

where G is the conductance matrix of the mesh edges (leaving out the load admittance equivalents of the i^{th} local subtree, $Y(mn_i)$), Y is a diagonal matrix with its i^{th} diagonal elements set to $Y(mn_i)$, and $\mathbf{J}$ is the vector of current sources which represent the equivalent current sources of the local subtrees at each mesh node. Expanding both sides of Equation (19) as a series in s, equating the coefficients of powers of s, and rearranging yields the following set of equations:

$$\begin{aligned} G^{'}\mathbf{V}_0 &= \mathbf{J}_0 \\ G^{'}\mathbf{V}_1 &= \mathbf{J}_1 - Y_1\mathbf{V}_0 \\ &\vdots \\ G^{'}\mathbf{V}_k &= \mathbf{J}_k - \sum_{j=1}^{k} Y_j\mathbf{V}_{k-j} \end{aligned} \tag{20}$$

where

$$G^{'} = G + Y_0 \tag{21}$$

is the combined conductance matrix of the mesh segments and the resistive components of subtree admittances. The above sets of linear equations are used to compute the voltages at the mesh nodes.

The matrix $G^{'}$ can be LU-factorized while solving for V_0 and the subsequent moments can be computed by computing the RHS and performing one forward and backward substitution. Since, as mentioned before, the order of $G^{'}$ is typically much smaller than the total number of nodes in the circuit, the solution for the mesh voltages can be found cheaply for the proposed supply network structure.

5.3 Propagating waveforms down the trees

The solution obtained from Equation (20) is used to set the node voltages to the computed value. These values are used to recursively compute the voltage at each of the internal node in the local

trees. The procedure is explained below.

Consider a subtree T_{mn_k} rooted at the mesh node mn_k. Let the calculated voltage at mn_k be $V(mn_k)$. Then the current in $Y(mn_k)$ is given by,

$$I(mn_k) = V(mn_k) \times Y(mn_k) \tag{22}$$

$I(mn_k)$ is the total current flowing into the subtree T_{mn_k}. T_{mn_k} is progressively decomposed into multiple subtrees, and the current flowing into each of the subtrees is calculated as a fraction of $I(mn_k)$. For example, if the node mn_k was connected to r edges in the tree T_{mn_k} as shown in Figure 10, then the tree T_{mn_k} can be decomposed into r subtrees $\{T_1, \cdots, T_r\}$ rooted at mn_k. The current flowing in the subtree T_i will be given by

$$I_{T_i}(mn_k) = V(mn_k) \cdot Y_{T_i}(mn_k) \tag{23}$$

where $Y_{T_i}(mn_k)$ is the admittance of the subtree T_i, as seen from node mn_k.

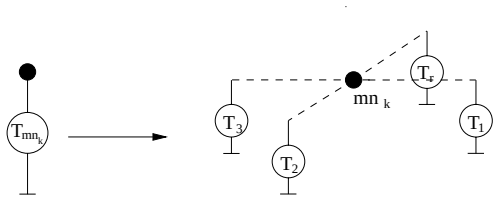

Figure 10: Computing individual branch currents

The individual subtree currents are used to determine the voltage at the nodes connected to mn_k by computing the voltage drop across the branches and subtracting from $V(mn_k)$. That is, given a segment e_j connecting the nodes mn_k and n_j, and given that the current through the subtree rooted at node mn_k and includes e_j is computed as $I_{T_j}(mn_k)$ from Equation (23), the voltage at n_j is given by

$$V(n_j) = V(mn_k) - I_{T_j} R(e_j) \tag{24}$$

where $R(e_j)$ is the resistance of the edge e_j.

This process is continued until the voltages at all the nodes in the tree T_{mn_k} have been computed. In this situation where the node of interest has a nonzero initial condition, a simple application of the Final Value Theorem [35] shows that an $[N/N]$ approximant is essential to capture

Circuit	# Nodes	$T_{COMPILE}$	T_{SIM}	T_{HSPICE}
Circuit 1	20	4.5s	0.33s	14.24s
Circuit 2	52	9.0s	0.81s	22.15s
Circuit 3	104	17.5s	1.58s	35.75s
Circuit 4	204	40.4s	3.29s	63.39s
Circuit 5	404	142.2s	7.78s	124.48s

Table 1: Execution time comparisons with HSPICE

the initial condition. The moments are used to generate these $[N/N]$ Padé approximations at each node (using the methods described in Section 3 B), which is then used to generate the time domain approximations for the given interval.

6 Experimental results

The symbolic algorithm was implemented in C++, and the results on several interconnect meshes were tested. We used the commercial simulator, HSPICE [36], to analyze the speed and accuracy of our approach.

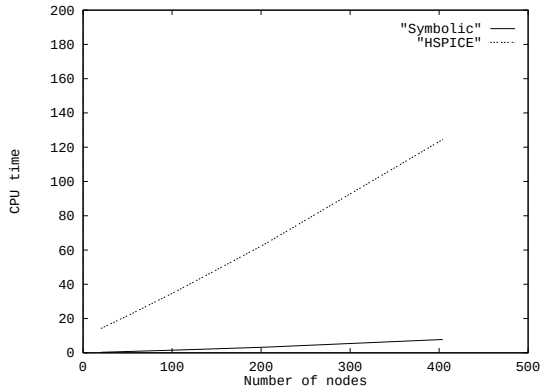

Figure 11: Execution time comparisons with HSPICE

In the experiments shown here, the compiled code first attempts a fourth order Padé approximant at each node. It has been observed in the literature that for interconnect circuits, reducing the order of Padé approximants usually improves the stability of the solution (Chapter 5 of [33]). If the fourth order approximation is unstable, then successive lower order approximations (third order, second order, etc.) are attempted until a stable approximant is found; the stability is tested

by applying the Routh-Hurwitz criterion [35]. Our experimental results showed that the program was never required to go below a second order approximant. The advantage of using approximants of at most order 4 is that closed-form formulæ can be used to calculate the roots of the transfer function denominator. We tested the use of higher order approximants, but observed that these did not improve the accuracy significantly and increased the execution time substantially. This increase is attributable to the slower iterative techniques that must be used to find the roots of equations of order > 4.

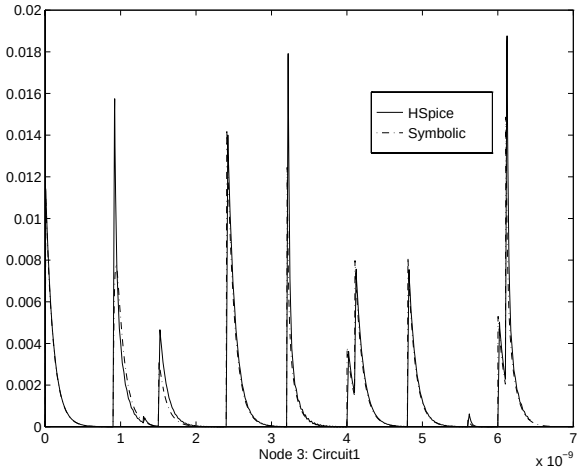

Figure 12: Waveform at node 3, Circuit 1

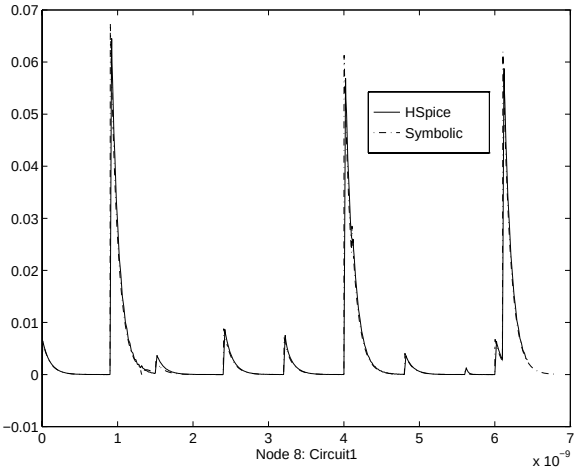

Figure 13: Waveform at node 9, Circuit 1

The CPU time of applying this method to five circuits, Circuit 1 through Circuit 5, of various sizes, are shown in Table 1. The quantity $T_{COMPILE}$ refers to the CPU compile time for

the generated code, T_{SIM} shows the time required for one simulation of our compiled code, and T_{SPICE} represents the simulation time required by HSPICE. A plot of the simulation time for each run (after the code has been compiled) against the circuit size is shown in Figure 11. For a fair comparison, the same switch models were used in HSPICE as for our program, i.e., each switch was represented as a resistor and a capacitor, rather than by a transistor.

To illustrate the accuracy of this method, Figures 12 through 15 show the waveform at two nodes for circuits Circuit 1 and Circuit 5, respectively using the switch model for both our method and for HSPICE.

A more realistic measure of the accuracy can be obtained by comparing the transistor model used for HSPICE with the switch model for our method, using an equivalent linear resistor that is equivalent to the transistor. Due to the nonlinearities associated with the transistor, it was infeasible to simulate large networks using HSPICE in a reasonable time. We show a comparison of the waveforms from our approach with the HSPICE waveforms at different nodes in Circuit 1, a 20-node network, in Figures 16 through 19. From the waveforms, it is clear that the match between the results of our approach and HSPICE is very good.

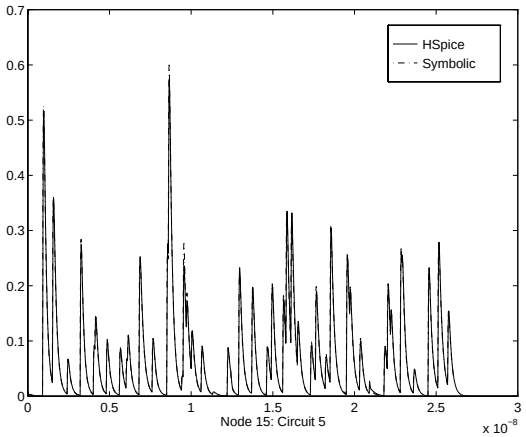

Figure 14: Waveform at node 15, Circuit 5

7 Conclusion

A new algorithm for the analysis of power/ground network topologies using Padé approximations, handling nonzero initial conditions, has been presented. The application was found to be very

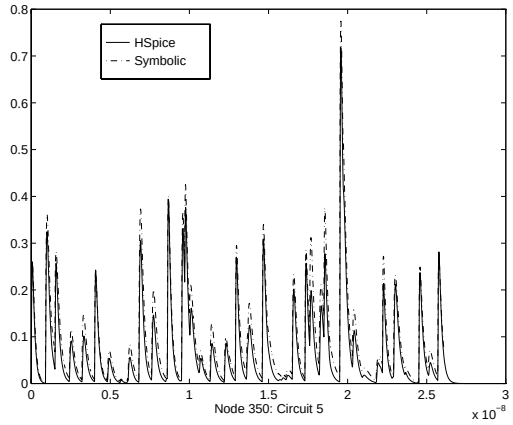

Figure 15: Waveform at node 350, Circuit 5

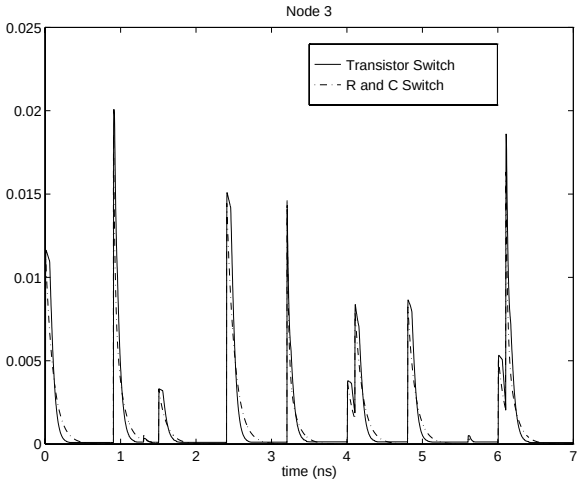

Figure 16: Waveform at node 3 of Circuit 1

amenable to symbolic analysis since the P/G network of each chip must be repeatedly analyzed by designers under different parameters and event sets before an optimal design is achieved. Experimental results show that it achieves improvements of more than an order of magnitude over SPICE in terms of execution time. The results generated by the algorithm are observed to be as accurate as SPICE at many nodes within the network. At other nodes where it is not as accurate, such as the waveform shown in Figure 15, it tracks the peaks correctly and captures the behavior of the highest peaks with a good degree of accuracy. This symbolic algorithm is very well suited for use in an optimization environment, where a good tradeoff must be made between speed and accuracy.

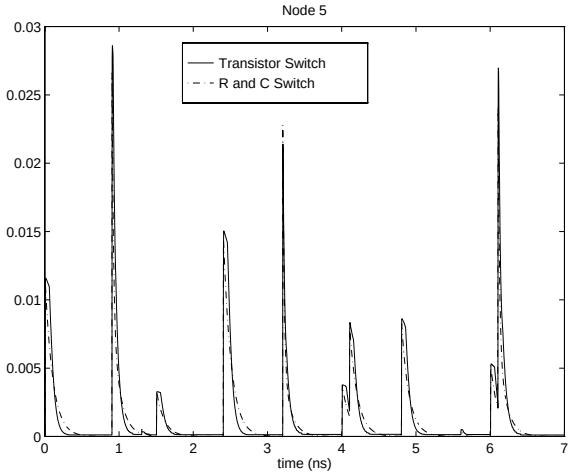

Figure 17: Waveform at node 5 of Circuit 1

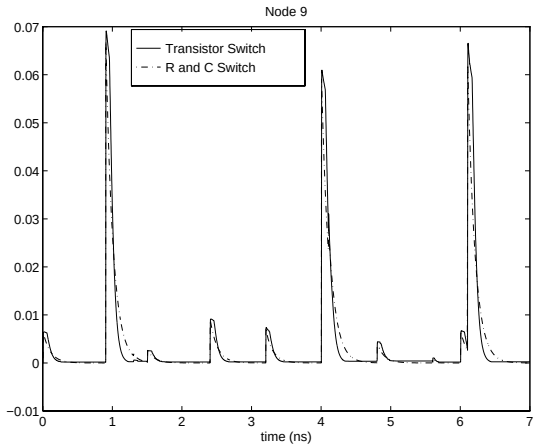

Figure 18: Waveform at node 9 of Circuit 1

References

[1] P. M. Lin and G. E. Alderson, "SNAP - A computer program for generating symbolic network functions," Tech. Rep. TR-EE 70-16, School of EE, Purdue University, West Lafayette, IN, Aug. 1970.

[2] J. A. Starzyk and A. Konczykowska, "Flowgraph analysis of large electronic networks," *IEEE Transactions on Circuits and Systems*, vol. CAS-33, pp. 302–315, Mar. 1993.

[3] G. Gielen, H. Walscharts, and W. Sansen, "ISAAC: A symbolic simulator for analog integrated circuits," *IEEE Journal of Solid-State Circuits*, vol. SC-24, pp. 1587–1597, Dec. 1989.

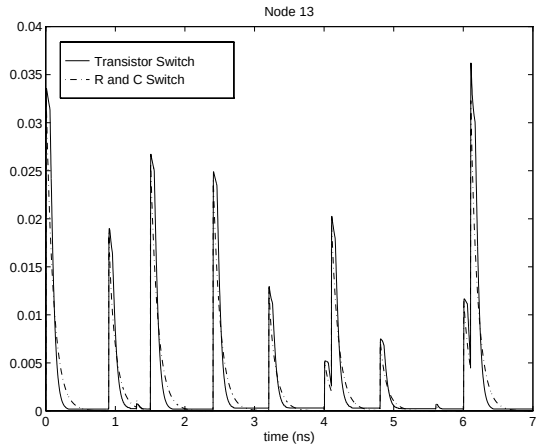

Figure 19: Waveform at node 13 of Circuit 1

[4] F. V. Fernández, A. Rodríguez-Vázquez, and J. L. Huerta, "An advanced symbolic analyzer for the automatic generation of analog circuit design equations," in *Proceedings of the IEEE International Symposium on Circuits and Systems*, pp. 810–813, 1991.

[5] G. DiDomenico *et al.*, "BRAINS: A symbolic solver for electronic circuits," in *International Workshop on Symbolic Methods and Applications to Circuit Design*, (Bagneux), 1991.

[6] S. Seda, M. Degrauwe, and W. Fichtner, "Lazy-expansion symbolic expression approximation in SYNAP," in *Proceedings of the IEEE/ACM International Conference on Computer-Aided Design*, pp. 310–317, 1992.

[7] S. M. Chang, J. F. MacKay, and J. G. M. Wierzba, "Matrix reduction and numerical approximation during computation techniques for symbolic analog circuit analysis," in *Proceedings of the IEEE International Symposium on Circuits and Systems*, pp. 1153–1156, 1992.

[8] M. Hassoun and K. McCarville, "Symbolic analysis of large-scale networks using a hierarchical signal flowgraph approach," *International Journal on Analog Integrated Circuits and Signal Processing*, vol. 3, pp. 31–42, Jan. 1993.

[9] M. M. Hassoun and P.-M. Lin, "A hierarchical network approach to symbolic analysis of large-scale networks," *IEEE Transactions on Circuits and Systems I*, vol. 42, pp. 201–211, Apr. 1995.

[10] R. Sommer, D. Ammermann, and E. Hennig, "More efficient algorithms for symbolic network analysis: supernodes and reduced loop analysis," *International Journal on Analog Integrated Circuits and Signal Processing*, vol. 3, Jan. 1993.

[11] A. Liberatore, A. Luchetta, S. Manetti, and M. Piccirilli, "A new symbolic program package for the interactive design of analog circuits," in *Proceedings of the IEEE International Symposium on Circuits and Systems*, pp. 2209–2212, 1995.

[12] Q. Yu and C. Sechen, "Efficient approximation of symbolic network function using matroid intersection algorithms," in *Proceedings of the IEEE International Symposium on Circuits and Systems*, pp. 2088–2091, 1995.

[13] B. S. Rodanski and M. Pierzchala, "Symbolic analysis of large electronic networks - an algebraic approach," in *Proceedings of the IEEE International Symposium on Circuits and Systems*, pp. 2075–2078, 1995.

[14] P. Wambacq, F. V. Fernández, G. Gielen, and W. Sansen, "Approximation during expression generation in symbolic analysis of analog integrated circuits," *Alta Frequenza Rivista Di Elettronica*, vol. 5, pp. 48–55, Nov. 1993.

[15] J. J. Hsu and C. Sechen, "Accurate extraction of simplified symbolic pole/zero expressions for large analog ics," in *Proceedings of the IEEE International Symposium on Circuits and Systems*, pp. 2083–2087, 1995.

[16] F. V. Fernández *et al.*, "Approximating nested format symbolic expressions," *Alta Frequenza Rivista Di Elettronica*, vol. 5, Nov. 1993.

[17] P. Wambacq, F. Fernández, G. Gielen, W. Sansen, and A. Rodríguez-Vázquez, "A family of matroid intersection algorithms for the computation of approximated symbolic network functions," in *Proceedings of the IEEE International Symposium on Circuits and Systems*, pp. IV–806–IV–809, 1996.

[18] L. T. Pillage and R. A. Rohrer, "Asymptotic waveform evaluation for timing analysis," *IEEE Transactions on Computer-Aided Design*, vol. 9, pp. 352–366, Apr. 1990.

[19] V. Raghavan, R.-A.Rohrer, L.-T.Pillage, J.-Y.Lee, J.-E.Bracken, and M.-M. Alaybeyi, "AWE-inspired," in *Proceedings of the IEEE Custom Integrated Circuits Conference*, pp. 18.1.1–18.1.8, 1993.

[20] B. R. Stanisic, N. K. Verghese, R. A. Rutenbar, L. R. Carley, and D. J. Allstot, "Addressing substrate coupling in mixed-mode IC's: Simulation and power distribution synthesis," *IEEE Journal of Solid-State Circuits*, vol. 29, pp. 226–238, Mar. 1994.

[21] B. R. Stanisic, R. A. Rutenbar, and L. R. Carley, "Addressing noise decoupling in mixed-signal IC's: Power distribution design and cell customization," *IEEE Journal of Solid-State Circuits*, vol. 30, pp. 321–326, Mar. 1995.

[22] S. Chowdhry and J. S. Barkatullah, "Estimation of maximum currents in MOS IC logic cirtuits," *IEEE Transactions on Computer-Aided Design*, vol. 9, pp. 642–654, June 1990.

[23] D. Stark and M. Horowitz, "Techniques for calculating currents and voltages in VLSI power supply networks," *IEEE Transactions on Computer-Aided Design*, vol. 9, pp. 126–132, Feb. 1990.

[24] U. Jagau, "SIMCURRENT- an efficient program for the estimation of current flow of complex CMOS circuits," in *Proceedings of the IEEE/ACM International Conference on Computer-Aided Design*, pp. 396–399, 1990.

[25] F. Rouatbi, B. Haroun, and A. J. Al-Khalili, "Power estimation tool for sub-micron CMOS VLSI circuits," in *Proceedings of the IEEE/ACM International Conference on Computer-Aided Design*, pp. 204–209, Nov. 1992.

[26] A. Nabavi-Lishi and N. Rumin, "Delay and bus current evaluation in CMOS logic circuits," in *Proceedings of the IEEE/ACM International Conference on Computer-Aided Design*, pp. 198–203, Nov. 1992.

[27] A. Tyagi, "Hercules: a power analyzer for MOS VLSI circuits," in *Proceedings of the IEEE/ACM International Conference on Computer-Aided Design*, pp. 530–533, 1987.

[28] F. N. Najm, R. Burch, Ping Yang, and I. N. Hajj, "Probabilistic simulation for reliability analysis of CMOS VLSI circuits," *IEEE Transactions on Computer-Aided Design*, vol. 9, pp. 439–450, Apr. 1990.

[29] H. Kriplani, F. N. Najm, and I. N. Hajj, "Pattern independent maximum current estimation in power and ground buses of CMOS VLSI circuits: Algorithms, signal correlations, and their resolution," *IEEE Transactions on Computer-Aided Design*, vol. 14, pp. 998–1012, Aug. 1995.

[30] C. L. Ratzlaff, N. Gopal, and L. T. Pillage, "RICE: Rapid interconnect circuit evaluator," in *Proceedings of the ACM/IEEE Design Automation Conference*, pp. 555–561, 1991.

[31] M. Sriram and S. M. Kang, "Fast approximation of the transient response of lossy transmission line trees," in *Proceedings of the ACM/IEEE Design Automation Conference*, pp. 691–696, 1993.

[32] H. B. Bakoglu, *Circuits, Interconnections and Packaging for VLSI.* Reading, MA: Addison-Wesley, 1990.

[33] E.-Chiprout and M.-S.-Nakhla, *Asymptotic Waveform Evaluation.* Boston, MA: Kluwer Academic Publishers, 1994.

[34] M. Sriram and S.-M. Kang, *Physical Design for Multichip Modules.* Boston, MA: Kluwer Academic Publishers, 1994.

[35] B. C. Kuo, *Automatic Control Systems.* Englewood Cliffs, NJ: Prentice-Hall, 6th ed., 1993.

[36] Meta-Software, Inc., *HSPICE user's manual, Version 96.1 for Release 96.1*, 1996.