DTT: Direct Truncation of the Transfer Function—An Alternative to Moment Matching for Tree Structured Interconnect

Yehea I. Ismail, *Member, IEEE*, and Eby G. Friedman, *Fellow, IEEE*

***Abstract*—A method is introduced to evaluate time domain signals within *RLC* trees with arbitrary accuracy in response to any input signal. This method depends on finding a low frequency reduced-order transfer function by direct truncation of the exact transfer function at different nodes of an *RLC* tree. The method is numerically accurate for any order of approximation, which permits approximations to be determined with a large number of poles appropriate for approximating *RLC* trees with underdamped responses. The method is computationally efficient with a complexity linearly proportional to the number of branches in an *RLC* tree. A common set of poles is determined that characterizes the responses at all of the nodes of an *RLC* tree which further enhances the computational efficiency. Stability is guaranteed by the DTT method for low-order approximations with less than five poles. Such low-order approximations are useful for evaluating monotone responses exhibited by *RC* circuits.**

***Index Terms*—Circuit simulation, inductance, interconnect, RLC, VLSI.**

I. INTRODUCTION

IT has become well accepted that interconnect delay dominates gate delay in current deep submicrometer VLSI circuits [1]–[8]. With the continuous scaling of technology and increased die area, this situation is becoming worse [9]–[14]. In order to properly design complex circuits, accurate characterization and simulation of the interconnect behavior and signal transients are required. This high accuracy is necessary for analyzing performance critical modules and nets and to accurately anticipate possible hazards during switching. Also, increasing performance requirements has forced a reduction of the safety margins used in worst case design, requiring more accurate interconnect delay characterization. Thus, the process of characterizing signal waveforms in tree structured interconnect (or nearly tree structured) is of primary importance since most interconnect in a VLSI circuit is tree structured [15]–[17].

Asymptotic waveform evaluation (AWE)-based algorithms [18]–[24] have gained popularity as a more accurate delay model as compared to the Elmore delay model. AWE uses moment matching to determine a set of low frequency dominant poles that approximate the transient response at the nodes of an *RLC* tree. However, AWE suffers two primary problems [19]–[23]. The first problem is that the AWE method can lead to an approximation with unstable poles even for low-order approximations [19]–[23]. The second problem is that AWE becomes numerically unstable for higher order approximations which limits the order of the approximations determined using AWE to less than approximately eight poles (of which some poles may be unstable and are discarded) [19]–[23]. This limited number of poles is inappropriate for evaluating the transient response of an underdamped *RLC* tree which requires a much greater number of poles to accurately capture the transient response at all of the nodes. To overcome this limitation, a set of model order reduction algorithms has been developed to determine higher order approximations appropriate for *RLC* circuits based on the state space representation of an *RLC* network. Examples are Pade via Lanczos (PVL) [25], Matrix Pade via Lanczos (MPVL) [26], Arnoldi Algorithms [27], Block Arnoldi Algorithms [28], passive reduced-order interconnect macromodeling algorithm (PRIMA) [29], [30], and the SyPVL Algorithm [31]. However, these model order-reduction techniques have significantly higher computational complexity than AWE. The complexity of PVL techniques is superlinear with n when inductance is present, where n is the order of the *RLC* tree and is equal to the total number of capacitors and inductors in the tree. As for PRIMA, the complexity is quadratic with the approximation order q [25]–[31]. This complexity is much higher than the complexity of AWE which is linearly proportional to n and q for an *RLC* tree [19]–[23]. Note that n can be on the order of thousands for a typical large industrial *RLC* circuit and q can be as high as 40.

The moments of a transfer function of order n results from expanding the transfer function into a Taylor series around $s = 0$ as given by

$$\begin{aligned} T(s) =& \frac{1 + a_1 s + a_2 s^2 + \cdots + a_m s^m}{1 + b_1 s + b_2 s^2 + \cdots + b_n s^n} \\ =& 1 + m_1 s + m_2 s^2 + m_3 s^3 + \cdots \end{aligned} \tag{1}$$

The i^{th} moment of the transfer function m_i is the coefficient of s^i in the series expansion. An explicit moment matching technique such as AWE calculates a reduced-order transfer function of the form

$$\begin{aligned} T_q(s) =& \frac{\hat{a}_0 + \hat{a}_1 s + \hat{a}_2 s^2 + \cdots + \hat{a}_{q-1} s^{q-1}}{1 + \hat{b}_1 s + \hat{b}_2 s^2 + \cdots + \hat{b}_q s^q} \\ =& 1 + m_1 s + m_2 s^2 + m_3 s^3 + \cdots + m_{2q-1} s^{2q-1} \end{aligned} \tag{2}$$

which has the first $2q$ moments as $T(s)$ in (1) where q is much smaller than n. The moments of the circuit are first calculated

Manuscript received November 5, 1999; revised August 17, 2001. This paper was recommended by Associate Editor M. Pedram.

Y. I. Ismail is with the Department of Electrical and Computer Engineering, Northwestern University, Evanston, IL 60208 USA (e-mail: ismail@ece.northwestern.edu).

E. G. Friedman is with the Department of Electrical and Computer Engineering, University of Rochester, Rochester, New York 14627 USA.

Publisher Item Identifier S 0278-0070(02)01049-7.

and then the parameters of $T_q(s)$ $\hat{a}_0 - \hat{a}_{q-1}$ and $\hat{b}_1 - \hat{b}_q$ are determined such that $T_q(s)$ have the same first $2q$ moments as $T(s)$ [19]–[23]. By matching the first $2q$ moments of $T(s)$, $T_q(s)$ represents a low frequency approximation of $T(s)$ since if s is sufficiently small, the terms with higher powers of s, $m_{2q}s^{2q} + m_{2q+1}s^{2q+1} + \cdots$, are negligible as compared to the terms with the lower moments. The higher the number of moments matched by $T_q(s)$ (or higher q), the higher the frequencies for which $T_q(s)$ accurately approximates $T(s)$.

This paper introduces another method by which a low-frequency approximation can be calculated. The new method is based on directly truncating the higher powers of s in the numerator and denominator of the original transfer function in (1). Hence, a q^{th} order approximate transfer function is given by

$$T_q(s) = \frac{1 + a_1 s + a_2 s^2 + \cdots + a_x s^x}{1 + b_1 s + b_2 s^2 + \cdots + b_q s^q} \tag{3}$$

where $q < n$. The numerator order is $x = m$ if $m \leq q - 1$; otherwise $x = q - 1$. Hence, this method in a sense matches the first q coefficients of s in the numerator and denominator of the transfer function $T(s)$ instead of the moments. If s (or the frequency) is sufficiently small, the terms with higher powers of s in the denominator and numerator polynomials ($b_{q+1}s^{q+1} - b_n s^n, a_{x+1}s^{x+1} - a_m s^m$) are negligible with respect to the lower power terms in $T_q(s)$. Thus, for low frequencies, $T_q(s)$ is an accurate representation of $T(s)$. Note that the coefficients $a_0 - a_x$ and $b_1 - b_q$ are exactly the same in $T_q(s)$ and $T(s)$.

The direct transfer function truncation (DTT) model order reduction method has much better numerical stability at higher approximation orders as compared to moment matching techniques due to the relation between the coefficients $b_1 - b_q$ and the poles of the transfer function given by

$$\begin{aligned} b_1 &= -\sum_{i=1}^{n} \frac{1}{p_i} \\ b_2 &= \sum_{j=1}^{n} \sum_{k=j+1}^{n} \frac{1}{p_j p_k} \\ b_3 &= -\sum_{i=1}^{n} \sum_{j=i+1}^{n} \sum_{k=j+1}^{n} \frac{1}{p_i p_j p_k}, \quad \cdots \end{aligned} \tag{4}$$

To illustrate the relation between the moments, poles, and residues of the transfer function, (1) can be expressed as a partial fractions sum given by

$$H(s) = \frac{k_1}{s - p_1} + \frac{k_2}{s - p_2} + \cdots + \frac{k_n}{s - p_n} \tag{5}$$

where p_i is the i^{th} pole of the transfer function and k_i is the corresponding residue. By expanding each term in (5) into powers of s, the moments of $H(s)$ can be expressed as

$$\begin{aligned} m_0 &= -\left(\frac{k_1}{p_1} + \frac{k_2}{p_2} + \cdots + \frac{k_n}{p_n}\right) \\ m_1 &= -\left(\frac{k_1}{p_1^2} + \frac{k_2}{p_2^2} + \cdots + \frac{k_n}{p_n^2}\right) \\ &\vdots \\ m_{2n-1} &= -\left(\frac{k_1}{p_1^{2n}} + \frac{k_2}{p_2^{2n}} + \cdots + \frac{k_n}{p_n^{2n}}\right). \end{aligned} \tag{6}$$

The poles with larger magnitudes are truncated when added to the dominant poles with smaller magnitudes in higher order moments due to the addition of poles raised to large powers. This behavior, in addition to the need to invert ill-conditioned matrices [18]–[21], renders AWE incapable of calculating higher order approximations to simulate complicated waveforms. As for DTT, larger magnitude poles are multiplied by smaller magnitude poles in all of the terms of the coefficients higher than b_1, and hence information about larger poles is present in $b_1 - b_q$ for much larger q than in the case of the moments. This relation between the denominator coefficients and the poles permits the poles to be determined with much larger magnitudes than AWE is capable of determining through moment matching.

The objective of this paper is therefore to describe the DTT method [32] for evaluating the transient response at the nodes of a general *RLC* tree which is capable of determining high-order approximations appropriate for underdamped *RLC* trees in a computationally efficient manner (complexity linear with n). A single line as a special case of a tree with only one output (or sink) is covered by this tree analysis methodology. This new method also has improved pole stability properties for low-order approximations as compared to AWE, a useful feature with *RC* trees which do not require higher order approximations. The rest of the paper is organized as follows. A description of the DTT method is provided in Section II. In Section III, the complexity and stability characteristics of the DTT method are discussed. The transient responses based on the DTT method for several *RC* and *RLC* trees are compared to SPICE simulations in Section IV. Finally, some conclusions are offered in Section V. Pseudocode describing the DTT method is provided in the Appendix.

II. The DTT Method

The concepts used to develop the DTT method are explained in this section. The rules governing the poles and zeros in an *RLC* tree are defined in Section II-A The method used to calculate the exact transfer functions at the nodes of an *RLC* tree is introduced in Section II-B The use of transfer function truncation to determine a reduced-order approximation is discussed in Section II-C. The process of determining the set of common poles describing the transient response of an *RLC* tree and the corresponding residues at each node of the tree is described in Section II-D.

A. Pole-Zero Behavior in RLC Trees

The poles and zeros of an *RLC* tree maintain specific relations to the poles and zeros of the subtrees forming the *RLC* tree. These rules are established in this subsection and are used in the following subsection to develop an algorithm to dete rmine the poles and zeros of a general *RLC* tree by recursively subdividing the tree into smaller subtrees.

Rule 1: The poles of an RLC circuit are zeros of the impedance seen at the input of the circuit.

This rule can be understood by referring to Fig. 1 and noting that the transfer functions describing the capacitor voltages and inductor currents have a common denominator (the characteristic equation of the tree) [33]–[37]. Thus, the transfer function

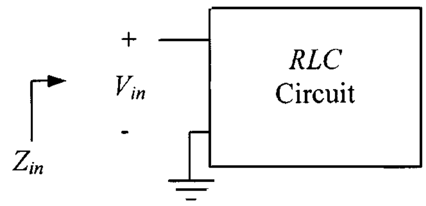


Fig. 1. A general *RLC* circuit.

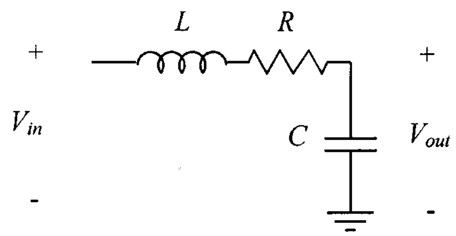


Fig. 2. Simple *RLC* circuit.

at an arbitrary node i of an *RLC* tree and the input admittance of the tree are given by

$$\frac{V_i(s)}{V_{in}(s)} = \frac{N_i(s)}{D(s)} \tag{7}$$

$$Y_{in}(s) = \frac{I_{in}(s)}{V_{in}(s)} = \frac{N_{I_{in}}(s)}{D(s)} \tag{8}$$

respectively, where $N_i(s)$ and $N_{Iin}(s)$ are functions of s dependent on the circuit structure and $D(s)$ is the common denominator of the circuit. The input impedance is

$$Z_{in}(s) = \frac{V_{in}(s)}{I_{in}(s)} = \frac{D(s)}{N_{Iin}(s)}. \tag{9}$$

Thus, the common denominator of an *RLC* circuit is the numerator of the input impedance which proves rule 1.

As an example, consider the single section *RLC* circuit shown in Fig. 2. This circuit has a transfer function and an input impedance given by

$$\frac{V_{out}(s)}{V_{in}(s)} = \frac{1}{s^2LC + sRC + 1} \tag{10}$$

$$\frac{V_{in}(s)}{I_{in}(s)} = sL + R + \frac{1}{sC} = \frac{s^2LC + sRC + 1}{sC} \tag{11}$$

respectively. Note that the denominator of the transfer function is the numerator of the input impedance. Another way to interpret Rule 1 is that an *RLC* circuit has a short-circuit input impedance when s is equal to the poles of the circuit.

Rule 2: The poles of an RLC circuit driven at node x are zeros of the transfer function at node x.

This rule can be explained by referring to Fig. 3. Note that the *RLC* circuit 2 is driven by the *RLC* circuit 1 at node x. Applying rule 1, Z_{in2} is a short-circuit between node x and the ground at frequencies equal to the poles of circuit 2. Hence, $V_x(s)$ is equal to zero when s is equal to the poles of circuit 2, i.e., the poles of circuit 2 are zeros of the transfer function at node x.

As an example, consider the circuit shown in Fig. 4. Note that the *RLC* subcircuit 2 is driven at node x and that if not connected, subcircuit 2 has a denominator given by $1+R_2C_2s+L_2C_2s^2$. The transfer functions at node x and the output node are

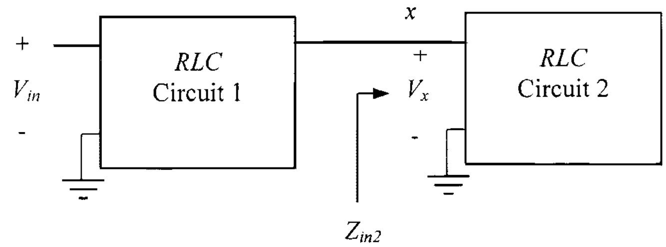


Fig. 3. A general *RLC* circuit composed of two *RLC* subcircuits connected together.

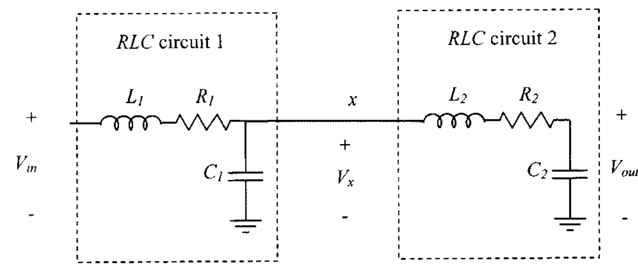


Fig. 4. A ladder *RLC* circuit composed of two *RLC* sections in series.

$$\frac{V_x(s)}{V_{in}(s)} = \frac{1 + R_2C_2s + L_2C_2s^2}{\begin{pmatrix} 1 + [R_1(C_1+C_2) + R_2C_2]s \\ + [L_1(C_1+C_2) + L_2C_2 + R_1C_1R_2C_2]s^2 \\ + [R_1C_1L_2C_2 + R_2C_2L_1C_1]s^3 + [L_1C_1L_2C_2]s^4 \end{pmatrix}} \tag{12}$$

$$\frac{V_{out}(s)}{V_{in}(s)} = \frac{1}{\begin{pmatrix} 1 + [R_1(C_1+C_2) + R_2C_2]s \\ + [L_1(C_1+C_2) + L_2C_2 + R_1C_1R_2C_2]s^2 \\ + [R_1C_1L_2C_2 + R_2C_2L_1C_1]s^3 + [L_1C_1L_2C_2]s^4 \end{pmatrix}} \tag{13}$$

Note that the numerator at node x is the same as the denominator of the disconnected subcircuit 2 in accordance with rule 2.

Rule 3: The poles of an RLC circuit driven at node x are zeros of the transfer functions at all of the nodes of parallel RLC circuits driven at the same node x.

This rule can be explained by referring to Fig. 5. The *RLC* subcircuits $2, 3, \ldots, k$ are driven by *RLC* subcircuit 1 at node x. Applying rule 1, Z_{in2} is a short-circuit at frequencies equal to the poles of circuit 2. Hence, $V_x(s)$ is equal to zero and all of the current supplied by circuit 1 is sunk to ground by Z_{in2} when s is equal to the poles of circuit 2. Since $V_x(s)$ is equal to zero and no current is supplied to the subcircuits $3, \ldots, k$ when s is equal to the poles of circuit 2, the voltages at all of the nodes of subcircuits $3, \ldots, k$ are equal to zero. Alternatively, the poles of circuit 2 are zeros of the transfer functions at all of the nodes of the parallel subcircuits driven at node x. The same is true for the poles of subcircuits $3, \ldots, k$ which are zeros of the transfer functions at all of the nodes of the parallel subcircuits driven at node x.

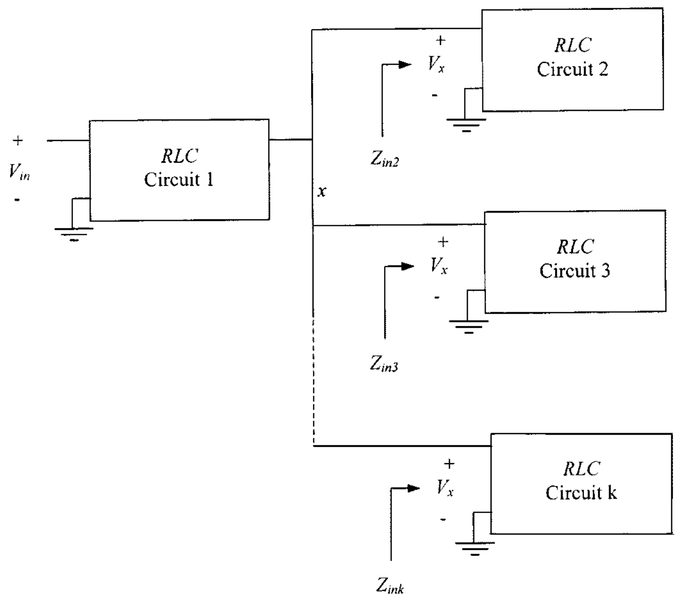


Fig. 5. A general *RLC* circuit composed of an *RLC* subcircuit driving several subcircuits connected in parallel.

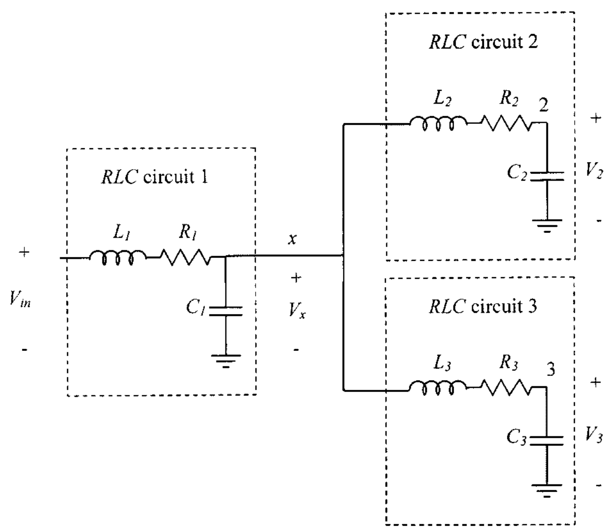


Fig. 6. An *RLC* tree composed of three *RLC* sections.

As an example, consider the *RLC* tree shown in Fig. 6. The *RLC* of section 1 drives the two parallel *RLC* in section 2 and section 3. The transfer functions at nodes x, 2, and 3 are given by

$$\frac{V_x(s)}{V_{in}(s)} = \frac{(1+R_2C_2s+L_2C_2s^2)(1+R_3C_3s+L_3C_3s^2)}{D} \tag{14}$$

$$\frac{V_2(s)}{V_{in}(s)} = \frac{(1+R_3C_3s+L_3C_3s^2)}{D} \tag{15}$$

$$\frac{V_3(s)}{V_{in}(s)} = \frac{(1+R_2C_2s+L_2C_2s^2)}{D} \tag{16}$$

respectively, where D is the common denominator and is a polynomial in s of order six. The specific form of D is not of interest here. The denominators of subcircuits 2 and 3 are $1+R_2C_2s+L_2C_2s^2$ and $1+R_3C_3s+L_3C_3s^2$, respectively. Note that both denominators are multiplied in the numerator of the transfer function at node x showing that the poles of subcircuits 2 and 3 are zeros of the transfer function at the driving node x in accordance with rule 2. Note also that the poles of subcircuit 2 are zeros of the transfer function at node 3 and vice versa, which verifies rule 3.

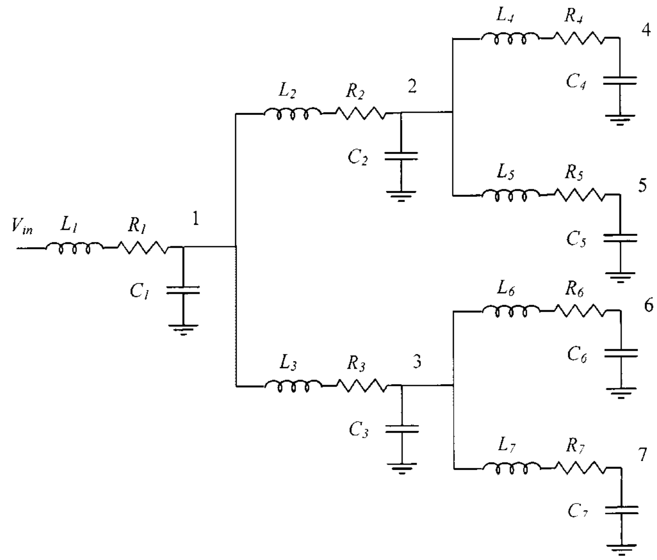


Fig. 7. General *RLC* tree.

B. Calculating the Transfer Functions at the Nodes of an RLC Tree

It is illustrated in this subsection how to recursively calculate the transfer functions at the nodes of an *RLC* tree using the concepts developed in the previous subsection. Consider the general *RLC* tree shown in Fig. 7. The current sunk to ground by a capacitor k is given by $C_k dv_k(t)/dt$ where $v_k(t)$ is the voltage across C_k. Thus, the current passing through the resistance R_1 and the inductance L_1 is given by

$$i_1(t) = \sum_k C_k \frac{dv_k(t)}{dt} \tag{17}$$

where the summation index k operates over all of the capacitors in the tree. The voltage drop across R_1 and L_1 is given by

$$\begin{aligned} v_{in}(t) - v_1(t) =& R_1 i_1(t) + L_1 \frac{di_1(t)}{dt} \\ =& R_1 \sum_k C_k \frac{dv_k(t)}{dt} \\ &+ L_1 \sum_k C_k \frac{d^2 v_k(t)}{dt^2}. \end{aligned} \tag{18}$$

In the frequency domain, this relation transforms to

$$V_{in}(s) - V_1(s) = (sR_1 + s^2 L_1) \sum_k C_k V_k(s). \tag{19}$$

Dividing (19) by $V_{in}(s)$, the following relation results:

$$1 - T_1(s) = (sR_1 + s^2 L_1) \sum_k C_k T_k(s) \tag{20}$$

where $T_1(s)$ is the transfer function at node 1 and $T_k(s)$ is the transfer function at node k. Note that determining the transfer function at node 1 is sufficient to determine the poles of the entire circuit since the transfer functions at all of the nodes of

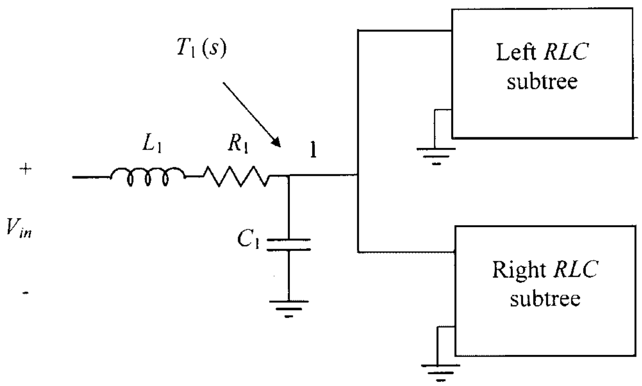


Fig. 8. Building block of a general *RLC* tree.

an *RLC* tree have a common denominator (as was mentioned previously).

Now consider the structure shown in Fig. 8 which depicts an *RLC* section driving left and right subtrees. Without loss of generality, a binary branching factor is used here since a general tree with an arbitrary branching factor can be transformed into a binary tree by inserting zero impedance branches [38], [39]. The structure shown in Fig. 8 can be used recursively to fully represent any *RLC* tree since the left and right subtrees can in turn be represented by the same structure. The transfer function at node 1 of Fig. 8 is given by (20), which can be reformulated by using the rational representations of the transfer functions, $T_1(s) = N_1(s)/D(s)$ and $T_k(s) = N_k(s)/D(s)$, and is

$$D(s) - N_1(s) = (sR_1 + s^2L_1)\sum_k C_kN_k(s). \tag{21}$$

Assume that the transfer functions at all of the nodes of the left and right *RLC* subtrees (when the trees are disconnected) are known and are given by $T_{lk1}(s) = N_{lk1}(s)/D_l(s)$ at node k_1 of the left subtree and $T_{rk2}(s) = N_{rk2}(s)/D_r(s)$ at node k_2 of the right subtree. The numerator at node 1, $N_1(s)$ of Fig. 8, can be directly calculated by applying rule 2 described in the previous subsection and is

$$N_1(s) = D_l(s) \bullet D_r(s). \tag{22}$$

The "•" operator above represents a polynomial multiplication. The denominator $D(s)$ can be determined from (21) as

$$D(s) = N_1(s) + (sR_1 + s^2L_1)M_1 \tag{23}$$

where M_1 is defined as

$$M_1 = \sum_k C_kN_k(s) \tag{24}$$

and characterizes the summation of the numerators of the transfer functions across the capacitors in the tree multiplied by the corresponding capacitances. The summation in M_1 operates over all of the capacitors in the tree and can be divided into three components

$$M_1 = C_1N_1(s) + \sum_{k1} C_{k1}N_{k1}(s) + \sum_{k2} C_{k2}N_{k2}(s) \tag{25}$$

where k_1 covers the capacitors in the left subtree and k_2 covers the capacitors in the right subtree. By applying rule 3, the numerators in the left subtree can be described in terms of the parameters of the disconnected left and right subtrees as $N_{k1}(s) = N_{lk1}(s) \bullet D_r(s)$. Similarly, $N_{k2}(s) = N_{rk2}(s) \bullet D_l(s)$. Thus, (25) can be reconfigured as

$$M_1 = C_1N_1(s) + \left(\sum_{k1} C_{k1}N_{lk1}(s)\right) \bullet D_r(s) + \left(\sum_{k2} C_{k2}N_{rk2}(s)\right) \bullet D_l(s). \tag{26}$$

Note that the two summations above are M_l and M_r of the disconnected left and right subtrees, respectively. Hence, M_1 can be fully calculated in terms of the disconnected left and right subtree parameters as

$$M_1 = C_1N_1(s) + M_l(s) \bullet D_r(s) + M_r(s) \bullet D_l(s). \tag{27}$$

Thus, by knowing the parameters of the left and right subtrees, $M_l(s)$, $D_l(s)$, $M_r(s)$, and $D_r(s)$, (22), (27), and (23) can be used in that order to determine $N_1(s)$, $M_1(s)$, and $D(s)$, respectively. The parameters of the left and right subtrees, $M_l(s)$, $D_l(s)$, $M_r(s)$, and $D_r(s)$, can be determined in turn in terms of their left and right subtrees by using the structure shown in Fig. 8 and (22), (27), and (23). This process is repeated recursively until the left and right subtrees are nonexistent. If the left subtree does not exist, then $M_l(s) = 0$ and $D_l(s) = 1$. If the right subtree does not exist, then $M_r(s) = 0$ and $D_r(s) = 1$.

After this recursion process terminates, the denominator and numerator across each capacitance C_k in the tree represent the transfer function for the subtree rooted at the *RLC* section k. For example, for the tree shown in Fig. 7, $D(s)$ and $N(s)$ at node 1 represent the transfer function at node 1 for the entire tree. However, $D(s)$ and $N(s)$ at node 2 represent the transfer function at node 2 for the subtree composed of the *RLC* sections, 2, 4, and 5. Also, $D(s)$ and $N(s)$ at node 4 represent the transfer function at node 4 for the subtree composed of *RLC* Section IV. Thus, after the recursion process terminates, the only relevant parameters for the entire *RLC* tree are $D(s)$ and $N(s)$ across the capacitor closest to the input (C_1 in the case of the tree shown in Fig. 7). The denominators and numerators at all of the other nodes are incorrect. The denominators at these nodes need not be corrected since these denominators are the same as the denominator at the node closest to the input. However, the numerators differ at each node and need to be corrected. According to rule 3, all of the numerators in the left subtree have to be multiplied by $D_r(s)$ and all of the numerators in the right subtree have to be multiplied by $D_l(s)$. This process is repeated recursively starting at the root of the tree and advancing toward the sinks.

Thus, the process of determining the transfer function at all of the nodes of an *RLC* tree consists of two steps. The first step is to calculate the common denominator of the *RLC* tree and is accomplished by the function Cal_Denominator presented as pseudocode in the Appendix which uses the recursive equations in (22), (27), and (23). The common denominator is the denominator at the node closest to the input of the *RLC* tree after the recursion terminates. The second step is to correct the numerators of the transfer functions at the nodes of the *RLC* tree. This task is achieved by the function Correct_Numerators which is also described as pseudocode in the Appendix.

C. Transfer Function Truncation and Approximation Order

The process of calculating the exact transfer functions at all of the nodes of an *RLC* tree has been described in the previous subsection. However, calculating the exact transfer function can be time consuming since n can be in the order of thousands for typical large industrial *RLC* trees. In practice, there is no need to calculate the thousands of poles characterizing an *RLC* tree since the transient behavior can be accurately characterized by a few number of low-frequency dominant poles [18]–[24] (typically several tens of poles). Thus, a low frequency approximation is required that can correctly anticipate the set of dominant poles without calculating the exact high-order transfer function.

Assume that the exact transfer function at a specific node of the *RLC* tree is given by

$$T(s) = \frac{1 + a_1 s + a_2 s^2 + \cdots + a_m s^m}{1 + b_1 s + b_2 s^2 + \cdots + b_n s^n} \tag{28}$$

where $b_1 - b_n$ and $a_1 - a_m$ are positive real constants. The system order n is equal to the total number of capacitors and inductors in the tree. The order of the numerator polynomial m is less than n and is dependent on the node at which the transfer function is calculated. A q^{th} order approximate transfer function is found by direct truncation of the exact transfer function $T(s)$ in (28) and is given by

$$T_q(s) = \frac{1 + a_1 s + a_2 s^2 + \cdots + a_x s^x}{1 + b_1 s + b_2 s^2 + \cdots + b_q s^q} \tag{29}$$

where $q < n$. The numerator order $x = m$ if $m \leq q - 1$; otherwise $x = q - 1$. The order of the numerator has to be less than the order of the denominator for a causal approximation. If s (or the frequency) is sufficiently small, the terms with higher power of s in the denominator and numerator polynomials ($b_{q+1}s^{q+1} - b_n s^n, a_{x+1}s^{x+1} - a_m s^m$) are negligible with respect to the lower power terms in $T_q(s)$. Thus, for low frequencies, $T_q(s)$ is an accurate representation of $T(s)$. The range of frequencies for which $T_q(s)$ is accurate increases as q increases.

The calculation of a q^{th} order approximation for the transfer functions at all of the nodes of an *RLC* tree can be accomplished by an order limited polynomial multiplication. To better understand this concept, assume that A and B are two polynomials of orders n_a and n_b, respectively. The polynomial C given by $A \bullet B$ has an order of $n_c = n_a + n_b$. The polynomials A, B, and C are given by

$$A = \sum_{i=0}^{n_a} a_i s^i \tag{30}$$

$$B = \sum_{i=0}^{n_b} b_i s^i \tag{31}$$

$$C = \sum_{i=0}^{n_c} c_i s^i \tag{32}$$

respectively, where the coefficients c_i are

$$c_i = \sum_{j=0}^{n_a} a_j b_{i-j}. \tag{33}$$

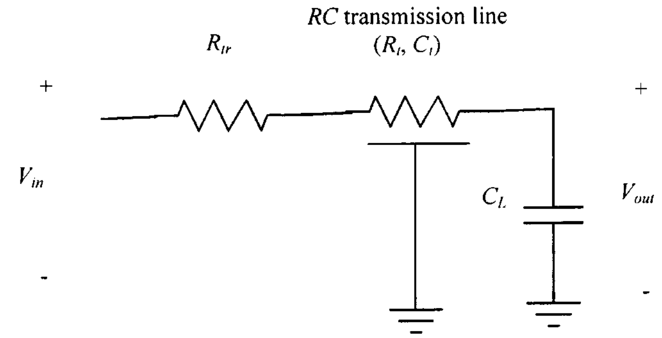


Fig. 9. An *RC* transmission line with a source resistance and a load capacitance.

Note that b_{i-j} is equal to zero if $i - j$ is out of the range of 0 to n_b. For a q limited polynomial multiplication, the highest desired power of s in C is q rather than n_c and the coefficients of higher powers of s do not need to be calculated. Also, A and B can be limited by q since higher powers than s^q in both polynomials cannot produce powers of s in C less than or equal to q. Hence, if a q^{th} order approximation is sought, all of the polynomial multiplications of the DTT method described in the previous subsection are q limited. These q limited polynomial multiplications are much less expensive than full polynomial multiplications since q is typically much less than n. The number of scalar multiplications required for a q limited polynomial multiplication is at most $q(q+1)/2$ when the polynomial orders, n_a and n_b, are equal to q. As is explained in Section III, the actual number of scalar multiplications performed by the DTT method is much less than the number of multiplications anticipated using the $q(q+1)/2$ complexity of a polynomial multiplication.

D. Determining the Poles, Residues, and the Transient Response

Once the common denominator of order q, $D_q(s)$ is determined, as described in the previous subsections, the first q dominant low frequency poles of the *RLC* tree can be calculated as the roots of the polynomial $D_q(s)$. A numerical method for evaluating the roots of a polynomial can be used to determine the *RLC* tree poles, $p_1 - p_q$, e.g., [40], [41]. The residues corresponding to each pole at a specific node can be efficiently calculated by direct substitution of the poles into the numerator of the transfer function at this node. The residues corresponding to the pole p_i at node j of an *RLC* tree can be calculated as

$$k_i^j = \frac{N_j(s = p_i)}{DP_i} \tag{34}$$

where

$$DP_i = b_q \prod_{\substack{r=1 \\ r \neq i}}^{q} (p_i - p_r) \tag{35}$$

where b_q is the coefficient of s^q in $D_q(s)$. Note that DP_i is independent of the node at which the residues are evaluated. Thus, DP_i can be evaluated once and used to calculate the residues at any number of nodes, which reduces the computational complexity when the transient response is required at many nodes.

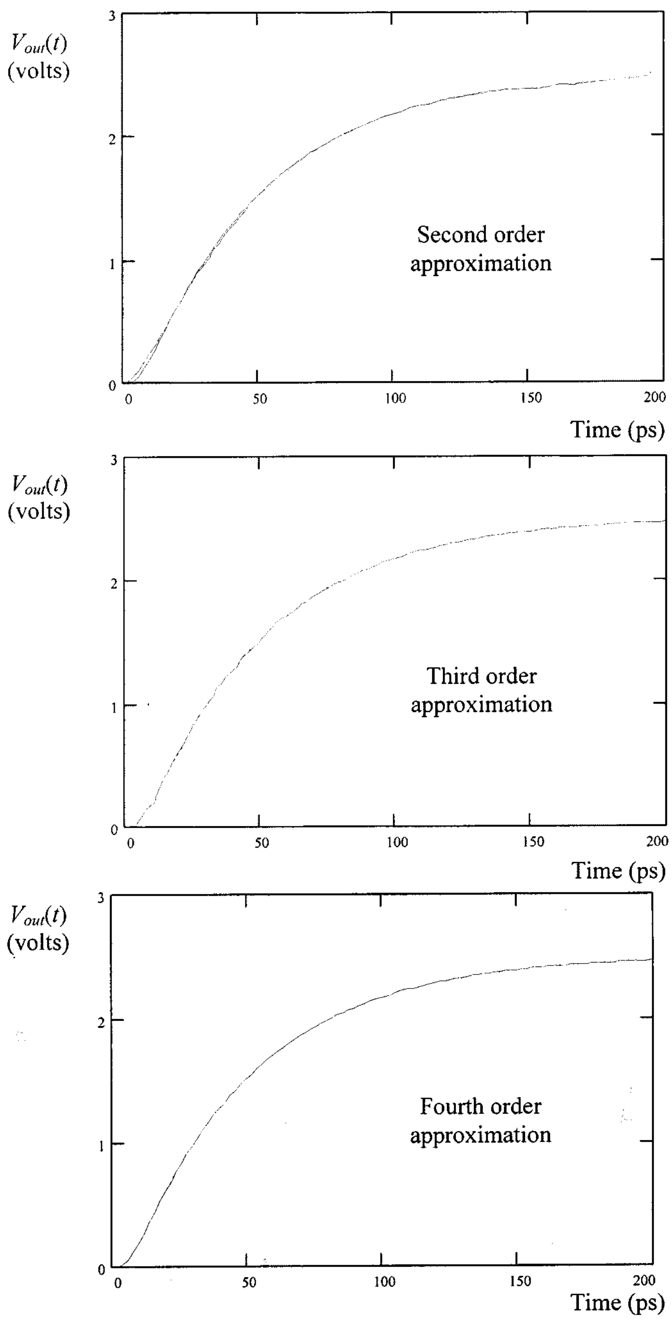


Fig. 10. Transient response evaluated using the DTT method as compared to SPICE simulations for the circuit shown in Fig. 9 using different approximation orders. SPICE simulations are represented by a solid line and the DTT simulations are represented by a dashed line. The circuit shown in Fig. 9 is simulated with $R_t = 50\ \Omega$, $C_t = 1$ pF, $R_{tr} = 25\ \Omega$, and $C_L = 0.05$ pF.

The poles of the circuit and the corresponding residues at node j of an *RLC* tree can be used to characterize the transfer function at node j as

$$T_j(s) = \sum_{i=1}^{q} \frac{k_i^j}{(s - p_i)}. \tag{36}$$

This transfer function can be used to calculate the time domain response at node j for an arbitrary input by multiplying the Laplace transform of the input by $T_j(s)$ and calculating the inverse Laplace transform of the resulting expression. For

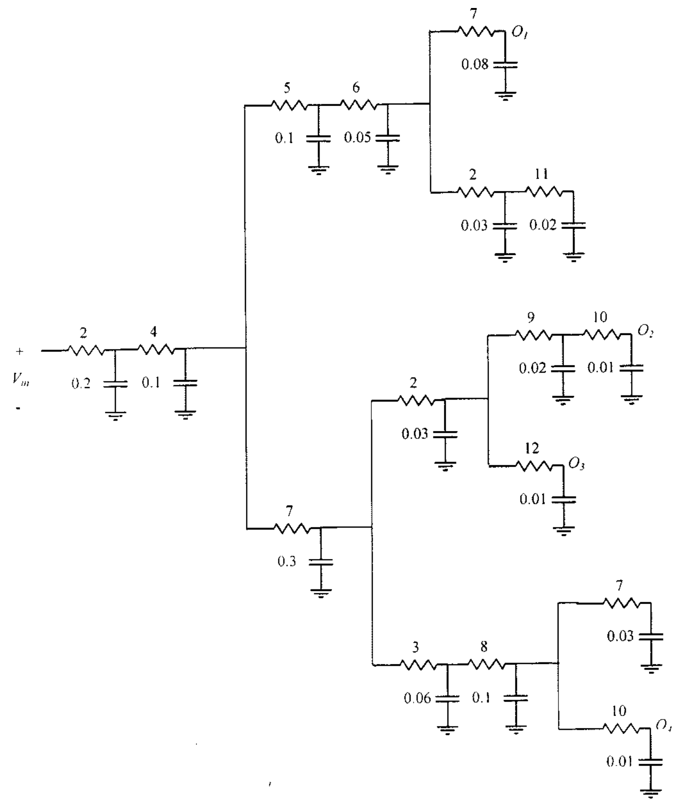


Fig. 11. A general *RC* tree. The resistance values shown are in ohms, inductance values are in nH, and capacitance values are in picofarads (pFs).

example, for a unit step input, the output response at node j, $e_j(t)$ is

$$e_j(t) = 1 + \sum_{i=1}^{q} \left[\frac{k_i^j}{p_i} e^{p_i t} \right]. \tag{37}$$

For an exponential input of the form

$$v_{in}(t) = 1 - e^{-t/\tau} \tag{38}$$

the transient response at node j is given by

$$e_j(t) = 1 + e^{-t/\tau} \left[\sum_{i=1}^{q} \frac{k_i^j \tau}{p_i \tau + 1} \right] + \sum_{i=1}^{q} \left[\frac{k_i^j}{p_i} \frac{1}{p_i \tau + 1} e^{p_i t} \right] \tag{39}$$

where τ is the time constant of the input signal. Some of the poles determined using the DTT method can be unstable due to the truncation of the denominator polynomial as discussed in the following section. These unstable poles can be simply discarded from the summations in (37) and (39). However, all of the poles should be included when calculating the residues using (34) and (35).

III. Complexity and Stability of the DTT Method

The DTT method has a complexity linearly proportional to the order of the tree n, which is twice the number of *RLC* sections in the tree since each *RLC* section has one capacitor and one inductor. This linear complexity occurs because the DTT method traverses each section in the tree only once as illustrated in the previous section and in the Appendix. At each section of

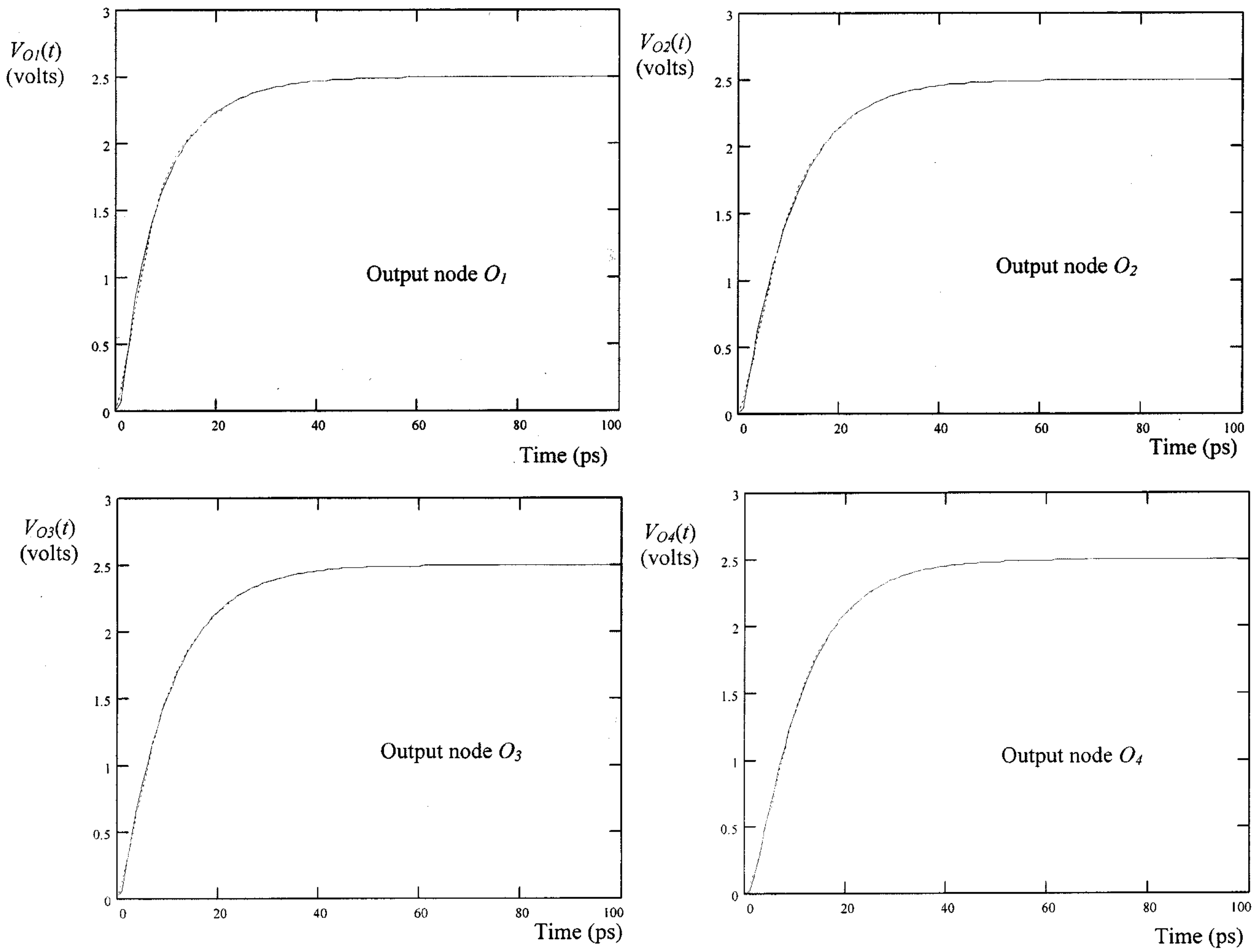


Fig. 12. Transient response evaluated using the DTT method as compared to SPICE simulations at different nodes of the *RC* tree depicted in Fig. 11. SPICE simulations are represented by a solid line and the DTT simulations are represented by a dashed line. A fourth-order approximation is used.

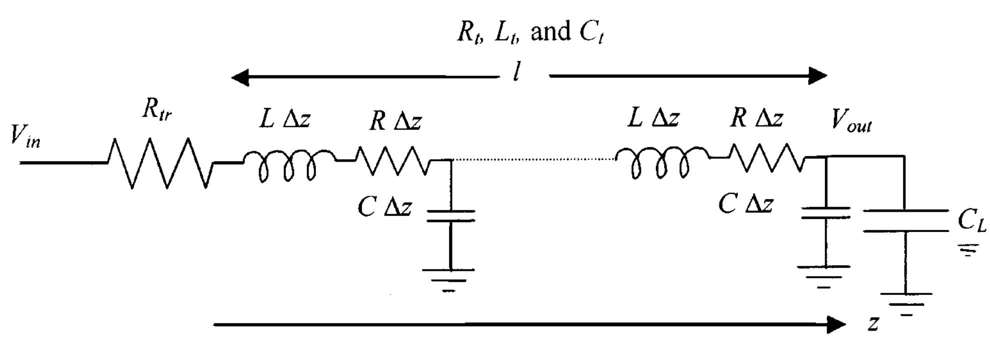


Fig. 13. An *RLC* transmission line with a source resistance and a load capacitance.

the *RLC* tree, polynomial multiplications are required to calculate the common denominator as given by (22), (27), and (23). Although polynomial multiplication has an apparent complexity proportional to q^2 for a q^{th} order approximation, the average number of scalar multiplications required per section is much lower than q^2 for any *RLC* tree. To better explain this argument, consider the following cases. A node of an *RLC* tree with the right subtree nonexistent has $M_r = 0$ and $D_r = 1$. Thus, (22), (27), and (23) become

$$N_1(s) = D_l(s) \tag{40}$$

$$M_1 = C_1 N_1(s) + M_l(s) \tag{41}$$

$$D(s) = N_1(s) + (sR_1 + s^2 L_1) M_1 \tag{42}$$

respectively. Note that the DTT method has no polynomial multiplication at a node of a tree driving only one branch. The DTT method is therefore particularly efficient for single lines and in those cases where branches of a tree can be subdivided into several series *RLC* sections to model the distributed nature of the interconnect impedance.

A binary tree (such as the tree illustrated in Fig. 7 with a total of r branches has $r/2$ leaves. These $r/2$ leaves are driven by $r/4$

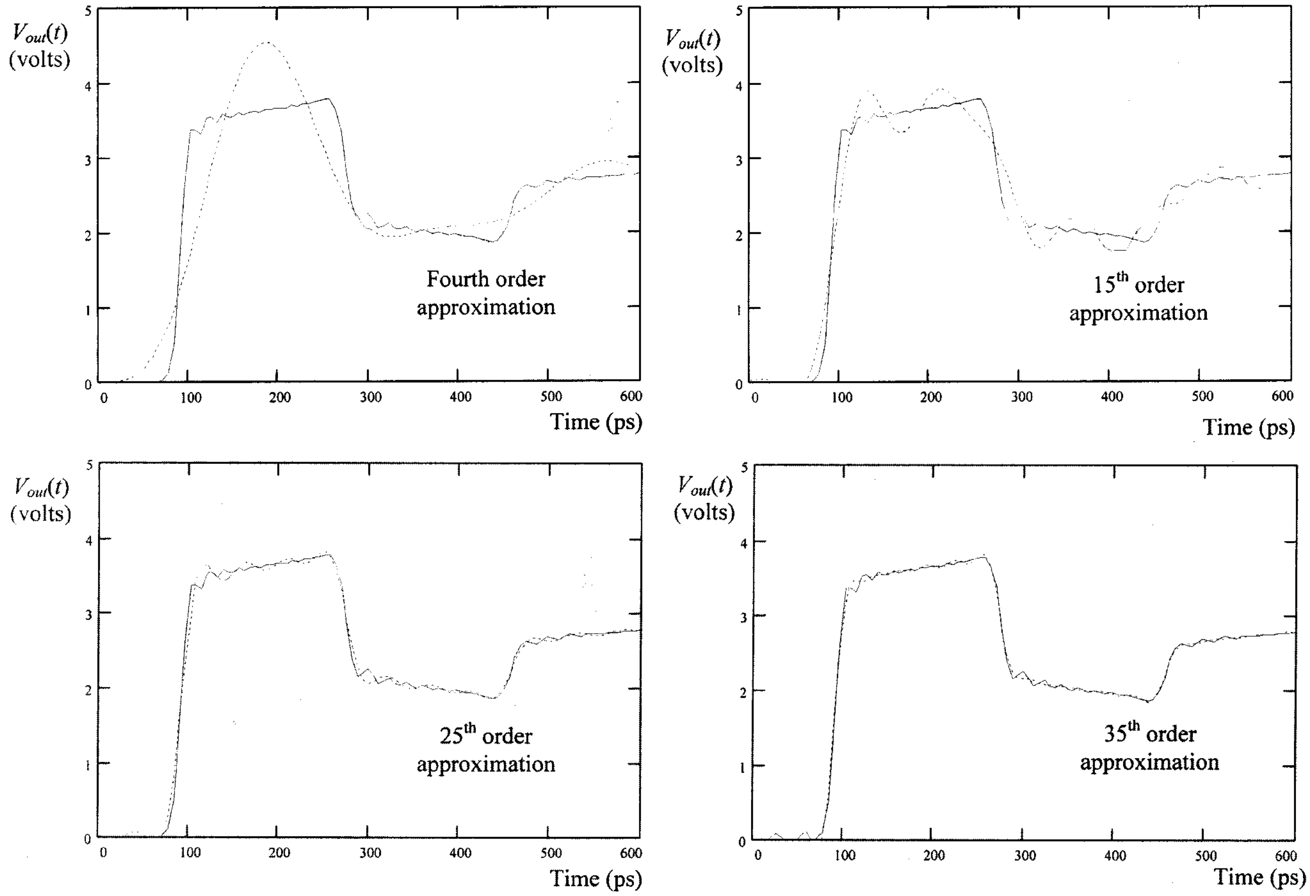


Fig. 14. Transient response evaluated using the DTT method as compared to SPICE simulations for the circuit shown in Fig. 13 using different orders of approximation. SPICE simulations are represented by a solid line and the DTT simulations are represented by a dashed line. The circuit shown in Fig. 13 is simulated with $R_t = 40\ \Omega$, $L_t = 7$ nH, $C_t = 1$ pF, $R_{tr} = 10\ \Omega$, and $C_L = 0.1$ pF.

branches, which are in turn driven by $r/8$ branches, and so on. Determining $N(s)$, $M(s)$, and $D(s)$ at the $r/2$ leaves requires only two scalar multiplications independent of the desired approximation order since for leaf i, $N(s) = 1$, $M(s) = C_i$, and $D(s) = 1 + R_i C_i s + L_i C_i s^2$. Applying these values at the next level with $r/4$ branches, the number of scalar multiplications required to determine $N(s)$, $M(s)$, and $D(s)$ is ten multiplications for a fourth-order approximation or higher. Thus, for a binary tree, the average number of scalar multiplications required by the DTT method is much less than q^2 multiplications per polynomial multiplication. For example, calculating a fourth-order approximation at all of the nodes of a binary tree requires a total number of scalar multiplications, SM, given by

$$SM_4 = 2 \cdot \frac{r}{2} + 10 \cdot \frac{r}{4} + 25 \cdot \frac{r}{4} = 9.75r. \tag{43}$$

Thus, the average number of scalar multiplications per branch of the tree is 9.75. The number of scalar multiplications calculated based on the q^2 polynomial multiplication complexity is $62r$ which greatly overestimates the complexity. The overestimation is even worse for higher values of q. For $q = 60$, the actual number of scalar multiplications is $160r$ multiplications while the q^2 model would predict $11\,000r$ multiplications. As the branching factor of an *RLC* tree increases, the overestimation by the q^2 model increases. This trend occurs because the leaves of the tree (which require only two scalar multiplications) constitute a larger fraction of the total number of branches with higher branching factors. For example, a tree with a branching factor of ten has almost 9/10 of its branches as leaves. For a general tree with a random branching factor at each node, the average number of scalar multiplications per node is much less than the q^2 model.

The above analysis demonstrates that the complexity of calculating the transfer functions at all of the nodes of an *RLC* tree is almost linear with the desired order of approximation, q. This feature greatly decreases the expense of calculating higher order approximations. Also, the method depends on simple polynomial multiplications, which are numerically accurate for very high orders of approximation [42]–[44].

An analysis of the stability of the approximations calculated using the DTT method shows that a DTT approximation with an order less than five is guaranteed to be stable. Assume that the exact common denominator of an *RLC* tree is given by

$$D(s) = 1 + b_1 s + b_2 s^2 + \cdots + b_n s^n. \tag{44}$$

The common denominator of a q^{th} order approximation is therefore given by

$$D_q(s) = 1 + b_1 s + b_2 s^2 + \cdots + b_q s^q. \tag{45}$$

For a second-order approximation, the condition for stability is that b_1 and b_2 are positive [33]. Since b_1 and b_2 are the coefficients of s and s^2 in the exact common denominator $D(s)$, b_1

and b_2 are guaranteed to be positive. This behavior occurs because a passive *RLC* tree is guaranteed to be stable [33]–[37] and stability requires that all of the coefficients of s in the denominator are positive. Therefore, a second-order approximation is always stable. For a third-order approximation, the Routh–Hurwitz criterion for stability [33] requires that $b_1 b_2 > b_3$. The coefficients b_1, b_2, and b_3 are given by

$$b_1 = -\sum_{i=1}^{n} \frac{1}{p_i} \tag{46}$$

$$b_2 = \sum_{j=1}^{n} \sum_{k=j+1}^{n} \frac{1}{p_j p_k} \tag{47}$$

$$b_3 = -\sum_{i=1}^{n} \sum_{j=i+1}^{n} \sum_{k=j+1}^{n} \frac{1}{p_i p_j p_k} \tag{48}$$

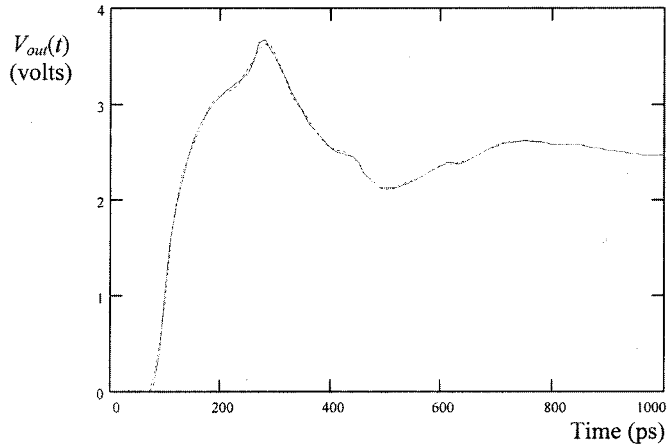
respectively, where $p_1, p_2, \ldots, p_n$ are the poles of the exact common denominator and have negative real parts due to the stability of a passive *RLC* circuit. Thus, the quantity $b_1 b_2 - b_3$ is given by

$$\begin{aligned} b_1 b_2 - b_3 &= -\sum_{i=1}^{n} \sum_{j=1}^{n} \sum_{k=j+1}^{n} \frac{1}{p_i p_j p_k} + \sum_{i=1}^{n} \sum_{j=i+1}^{n} \sum_{k=j+1}^{n} \frac{1}{p_i p_j p_k} \\ &= -\sum_{i=1}^{n} \sum_{j=1}^{i} \sum_{k=j+1}^{n} \frac{1}{p_i p_j p_k}. \end{aligned} \tag{49}$$

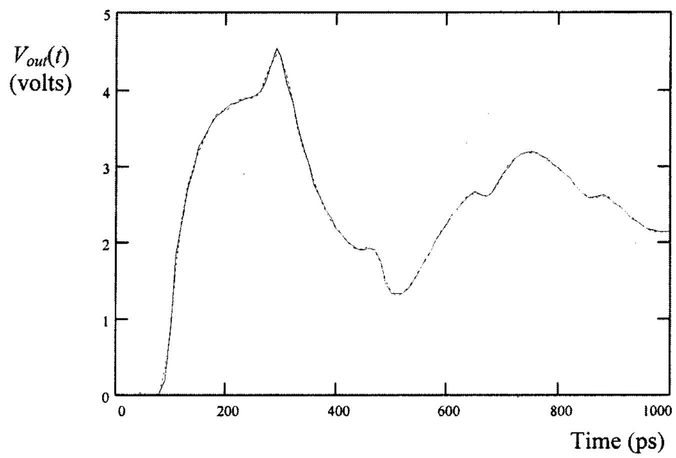
Note that the quantity $b_1 b_2 - b_3$ is positive since $p_1, p_2, \ldots, p_n$ have negative real parts. Thus, a third-order approximation is also guaranteed to be stable. The same procedure can be repeated for a fourth-order system. It can be shown that stability is also guaranteed for a fourth-order system. These low-order approximations are useful for *RC* trees since the signals within an *RC* tree can typically be approximated with a few dominant poles due to the monotone nature of the response. Approximations of order five or higher are not guaranteed to be stable. However, since the DTT method is numerically stable for any order of approximation and since the computational complexity increases slowly with the approximation order, high order approximations can always be determined using the DTT method to correctly detect all of the poles in the frequency range of interest.

IV. Experimental Results

The DTT method is applied in this section to calculate the transient response of several *RC* and *RLC* trees. The resulting transient responses are compared to SPICE simulations to evaluate the accuracy of the DTT method. The DTT method is applied first to evaluate the transient response of the *RC* circuit shown in Fig. 9. The circuit is composed of a distributed *RC* transmission line driven by a lumped resistance R_{tr} (which represents the output impedance of the driving gate) and a load capacitance C_L (which represents the input capacitance of the driven gate). The line has a total resistance of R_t and a total capacitance of C_t. The transient response based on the DTT method with approximation orders of two, three, and four are compared to SPICE in Fig. 10. Note that a second-order approximation has a negligible error in the transient response as compared to SPICE and that the third and fourth order approximations are practically exact.



(a)



(b)

Fig. 15. Transient response evaluated using the DTT method as compared to SPICE simulations for the circuit shown in Fig. 13 using different line parameters. SPICE simulations are represented by a solid line and the DTT simulations are represented by a dashed line. (a) $R_t = 30\ \Omega$, $L_t = 7$ nH, $C_t = 1$ pF, $R_{tr} = 20\ \Omega$, $C_L = 0.5$ pF, and approximation order $= 20$. (b) $R_t = 20\ \Omega$, $L_t = 8$ nH, $C_t = 1$ pF, $R_{tr} = 10\ \Omega$, $C_L = 0.4$ pF, and approximation order $= 25$.

The second circuit simulated using the DTT method is the *RC* tree shown in Fig. 11. The transient response at several nodes of the tree are calculated based on the DTT method and compared to SPICE in Fig. 12. A fourth-order DTT approximation is used to calculate the transient responses shown in Fig. 12. Note that a fourth-order approximation is accurate as compared to SPICE simulations. In general, a fourth-order approximation is sufficiently accurate for most *RC* trees. The guaranteed stability of a fourth-order approximation is therefore a valuable feature for *RC* circuits. Note that despite the fact that an *RC* circuit cannot produce complex poles [34]–[37], a reduced-order approximation based on the DTT method can result in complex poles for an *RC* circuit. However, the resulting complex poles for *RC* circuits always produce accurate stable monotone responses.

The circuit shown in Fig. 13 represents an *RLC* transmission line with a lumped source resistance and a load capacitance and

TABLE I
A GENERAL *RLC* TREE. THE TREE HAS SEVERAL *RLC* SECTIONS, EACH SECTION OF WHICH COMPRISES A ROW OF THE HAS AN ID NUMBER. THE ID NUMBERS OF THE LEFT AND RIGHT *RLC* SECTIONS DRIVEN BY AN *RLC* SECTION ARE GIVEN IN THE FIFTH AND SIXTH COLUMNS. A ZERO IN THESE COLUMNS IMPLIES THAT THE LEFT OR RIGHT SECTIONS DO NOT EXIST

RLC section number	R (Ω)	L (nH)	C (pF)	Left section number	Right section number
1	2	0.07	0.2	2	0
2	4	0.06	0.1	4	3
3	7	0.04	0.3	6	7
4	5	0.05	0.1	5	0
5	6	0.03	0.05	12	11
6	6	0.06	0.03	10	9
7	3	0.06	0.06	8	0
8	8	0.04	0.1	15	16
9	12	0.05	0.01	0	0
10	9	0.04	0.02	14	0
11	2	0.05	0.03	13	0
12	7	0.03	0.08	0	0
13	11	0.07	0.02	20	0
14	10	0.03	0.01	19	0
15	7	0.04	0.03	17	18
16	10	0.02	0.01	0	0
17	12	0.02	0.01	0	0
18	3	0.04	0.1	24	0
19	15	0.04	0.02	22	23
20	5	0.06	0.07	21	0
21	5	0.06	0.07	0	0
22	5	0.05	0.05	0	0
23	8	0.04	0.03	27	26
24	8	0.05	0.02	25	0
25	8	0.06	0.02	30	0
26	2	0.04	0.02	0	0
27	7	0.03	0.04	28	29
28	16	0.02	0.06	0	0
29	5	0.05	0.06	0	0
30	8	0.04	0.02	0	0

is simulated using the DTT method. The transient response is calculated based on the DTT method with approximation orders of 4, 15, 25, and 35 and is compared to SPICE in Fig. 14. Note that an approximation order between 25 and 35 is required for an underdamped response with second-order oscillations to achieve a SPICE-like accuracy. Such high-order approximations cannot be achieved by AWE [19]–[23] due to its numerical instability with high approximation orders. Other methods capable of calculating such high-order approximations [25]–[31] have a much higher computational complexity as compared to the DTT method. The computational efficiency of the DTT method and its numerical accuracy for very high orders of approximation makes it suitable for accurately simulating *RLC* trees. Several simulations of the circuit shown in Fig. 13 are shown in Fig. 15 with different line parameters and source and load impedances. The DTT method accurately characterizes the waveform details as compared to SPICE.

The transient response at several nodes of the *RLC* tree characterized in Table I are evaluated based on the DTT method and compared to SPICE in Fig. 16. A 40th-order approximation is used and is highly accurate as compared to SPICE. A 45th-order approximation is used to evaluate the transient response of a large copper interconnect tree based on a 0.25-μm CMOS IBM technology. The tree has 673 capacitors and 673 inductors. The transient responses based on the DTT method and SPICE are compared in Fig. 17. Note that the DTT method is capable of accurately characterizing the transient response of large industrial *RLC* trees with complicated nonmonotone underdamped responses.

V. CONCLUSION

The DTT method has been introduced to evaluate the transient responses within *RLC* trees with arbitrary accuracy for any input signal. The DTT method is numerically accurate for any order of approximation, which permits approximations to be determined with a large number of poles appropriate for approximating *RLC* trees with underdamped responses. The DTT method is computationally efficient with a complexity linearly proportional to the number of branches in the tree. A common set of poles is determined that characterizes the responses at all of the nodes of an *RLC* tree, which further enhances the computational efficiency of the proposed method. The stability is guaranteed by the DTT method for low-order approximations with less than five poles, which is useful for efficiently analyzing *RC* circuits.

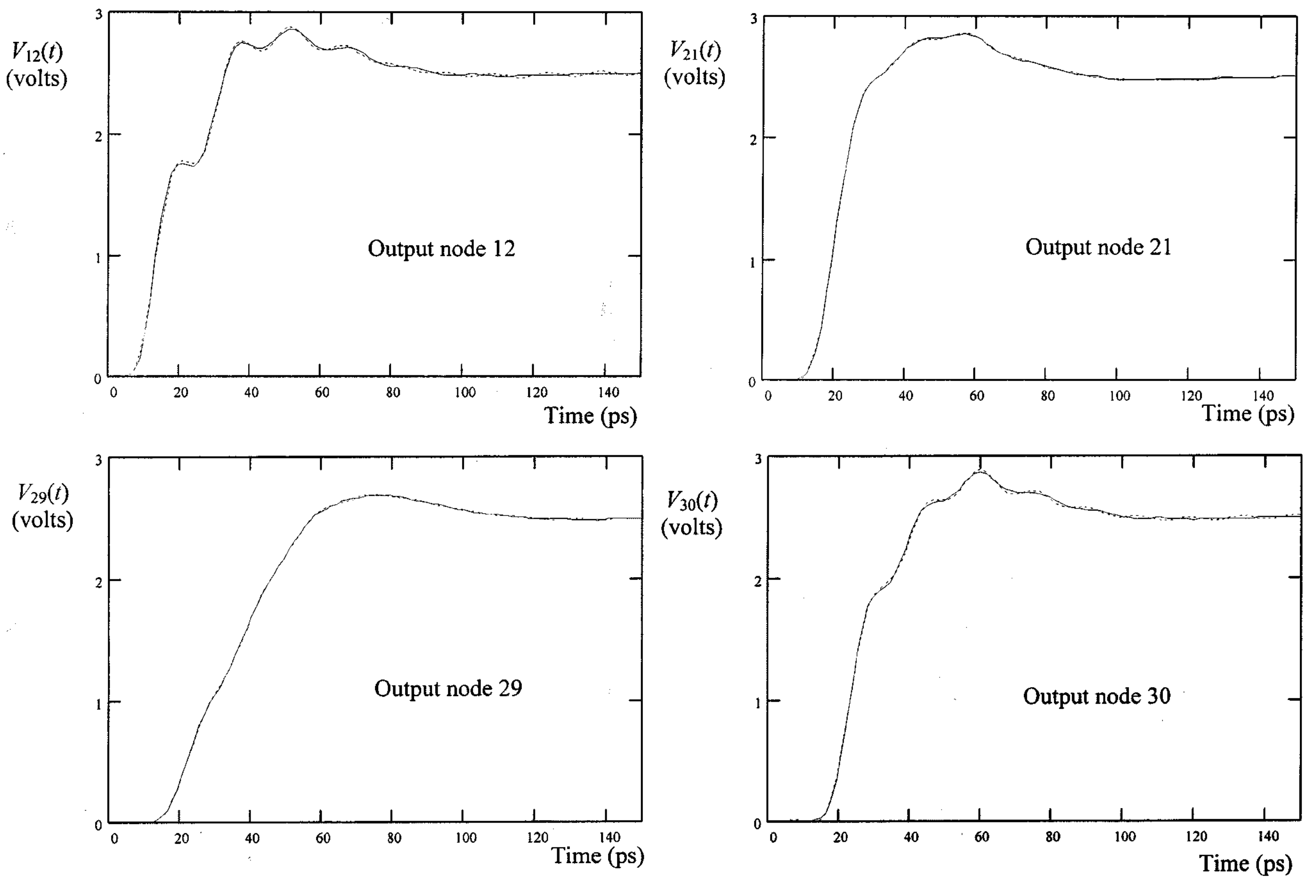


Fig. 16. Transient response evaluated using the DTT method as compared to SPICE simulations at different nodes of the *RLC* tree characterized in Table I. SPICE simulations are represented by a solid line and the DTT simulations are represented by a dashed line. A 40th approximation order is used.

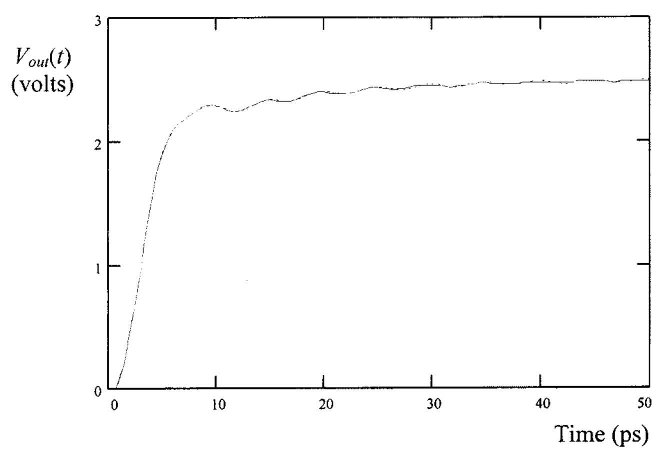


Fig. 17. Transient response evaluated using the DTT method as compared to SPICE simulations at a particular leaf node of a large copper interconnect *RLC* tree based on an IBM 0.25 μm CMOS technology. SPICE simulations are represented by a solid line and the DTT simulations are represented by a dashed line. A 45th approximation order is used.

```
Cal_Denominator (section* w)
{
    if(right(w)=0)          /* there is no right section driven by w */
        {D_r=1; M_r=0;}
    else                    /* there is a right section driven by w */
        {Cal_Denominator(right(w)); D_r=right(w)->D; M_r=right(w)->M;}

    if(left(w)=0)           /* there is no left section driven by w */
        {D_l=1; M_l=0;}
    else                    /* there is a left section driven by w */
        {Cal_Denominator(left(w)); D_l=left(w)->D; M_l=left(w)->M;}

    w->N = D_l•D_r;
    w->M = w->C*w->N + M_l•D_r + M_r•D_l;
    w->D = w->N + (w->M)•[(w->R)*s+(w->L)*s^2];
}
```

Fig. 18. Pseudocode for calculating the common denominator of an *RLC* tree.

Appendix
The DTT Algorithm

A general *RLC* tree is composed of several connected *RLC* sections. Each *RLC* section has a series resistance, inductance, and capacitance with the capacitance grounded as shown in Fig. 2. The objective is to calculate the transfer functions across all of the capacitors in the *RLC* tree. The function to calculate the common denominator of an *RLC* tree rooted at the *RLC* section w_1 is Cal_Denominator and uses the DTT algorithm as explained in Section II. A pseudocode that performs this task is described in Fig. 18.

The function is initially called by Cal_Denominator (w_1) and recursively calculates the common denominator. The structure "section" has the elements *R*, *L*, and *C*, which represent the resistance, inductance, and capacitance of an *RLC* section, respectively. The structure also has the arrays N, M, and D, which represent the polynomials of the numerator, M in (27), and the denominator of the transfer function across the capacitor of the

```
Correct_Numerators(section *w, Poly F_in)
{
        if(right(w)≠0)          /* w drives a right section */
                {F_r = F_in•D_l; Correct_Numerator(right(w), F_r);}

        if(left(w)≠0)           /* w drives a left section */
                {F_l = F_in•D_r; Correct_Numerator(left(w), F_l);}

        w->N = w->N•F_in;
}
```

Fig. 19. Pseudocode for correcting the numerators of the transfer functions at all of the nodes of an *RLC* tree.

RLC section, respectively. The operator "•" represents a polynomial multiplication. An efficient limited order polynomial multiplication function should be used as discussed in Section II. The functions, left (w) and right (w), return pointers to the left and right sections driven by w, respectively. If no left (right) section is driven by w, left $(w) = 0$ (right $(w) = 0$). The function uses (22), (27), and (23) and the recursion termination conditions described by the DTT method in Section II-B.

The second step is to correct the numerators of the transfer functions at the nodes of the *RLC* tree. The function performing this task is described in Fig. 19. The function is initially called by Correct_Numerators $(w_1, 1)$ and recursively corrects the numerators at all of the nodes of the *RLC* tree as described in Section II-B. Note that the Correct_Numerators function has to be called after the Cal_Denominator function has been called.

REFERENCES

[1] D. A. Priore, "Inductance on silicon for sub-micron cmos VLSI," in *Proc. IEEE Symp. VLSI Circuits*, May 1993, pp. 17–18.
[2] D. B. Jarvis, "The effects of interconnections on high-speed logic circuits," *IEEE Trans. Electron. Computers*, vol. EC-10, pp. 476–487, Oct. 1963.
[3] Y. Eo and W. R. Eisenstadt, "High-Speed VLSI interconnect modeling based on S-parameter measurement," *IEEE Trans. Components, Hybrids, and Manuf. Technol.*, vol. 16, pp. 555–562, Aug. 1993.
[4] A. D. Deutsch *et al.*, "High-Speed signal propagation on lossy transmission lines," *IBM J. Res. Development*, vol. 34, no. 4, pp. 601–615, July 1990.
[5] ——, "Modeling and characterization of long interconnections for high-performance microprocessors," *IBM J. Res. Development*, vol. 39, pp. 547–667, Sept. 1995.
[6] ——, "When are transmission-line effects important for on-chip interconnections? ," *IEEE Trans. Microwave Theory Techn.*, vol. 45, pp. 1836–46, Oct. 1997.
[7] Y. I. Ismail, E. G. Friedman, and J. L. Neves, "Figures of merit to characterize the importance of on-chip inductance," in *Proc. IEEE/ACM Design Automation Conf.*, June 1998, pp. 560–565.
[8] M. P. May, A. Taflove, and J. Baron, "FD-TD modeling of digital signal propagation in 3-d circuits with passive and active loads," *IEEE Trans. Microwave Theory and Techniques*, vol. 42, pp. 1514–1523, Aug. 1994.
[9] T. Sakurai, "Approximation of wiring delay in MOSFET LSI," *IEEE J. Solid-State Circuits*, vol. SC-18, pp. 418–426, Aug. 1983.
[10] G. Y. Yacoub and H. P. G. Friedman, "A system for critical path analysis based on back annotation and distributed interconnect impedance models," *Microelectron. J.*, vol. 18, no. 3, pp. 21–30,, June 1988.
[11] J. Torres, "Advanced copper interconnections for silicon CMOS technologies," *Appl. Surface Sci.*, vol. 91, no. 1, pp. 112–123, Oct. 1995.
[12] C. F. Webb, "A 400 MHz S/390 microprocessor," in *Proc. IEEE Int. Solid-State Circuits Conf.*, Feb. 1997, pp. 448–449.
[13] L. T. Pillage, "Coping with RC(L) interconnect design headaches," in *Proc. IEEE/ACM Int. Conf. Computer-Aided Design*, Sept. 1995, pp. 246–253.
[14] P. J. Restle and A. Duetsch, "Designing the best clock distribution network," in *Proc. IEEE VLSI Circuit Symp.*, June 1998, pp. 2–5.
[15] W. C. Elmore, "The transient response of damped linear networks," *J. Appl. Phys.*, vol. 19, pp. 55–63, Jan. 1948.
[16] J. L. Wyatt, *Circuit Analysis, Simulation and Design*, North-Holland: Elsevier Science, 1987.
[17] Y. I. Ismail, E. G. Friedman, and L. N. Jose, "Equivalent Elmore delay for RLC trees," in *Proc. ACM/IEEE Design Automation Conf.*, June 1999, pp. 715–720.
[18] L. T. Pillage and R. A. Rohrer, "Delay evaluation with lumped linear RLC interconnect circuit models," in *Proc. Caltech Conf. VLSI*, May 1989, pp. 143–158.
[19] L. T. Pillage, R. A. Rohrer, and C. Visweswariah, *Electronic Circuit and System Simulation Methods*. New York: McGraw-Hill, 1994.
[20] L. T. Pillage and R. A. Rohrer, "Asymptotic waveform evaluation for timing analysis," *IEEE Trans. Computer-Aided Design*, vol. 9, pp. 352–366, Apr. 1990.
[21] D. F. Anastasakis, N. Gopal, S. Y. Kim, and L. T. Pillage, "On the stability of approximations in asymptotic waveform evaluation," in *Proc. IEEE/ACM Design Automation Conf.*, June 1992, pp. 207–212.
[22] C. L. Ratzlaff, "A fast algorithm for computing the time moments of RLC circuits," Masters thesis, Univ. Texas at Austin, Austin, TX, 1991.
[23] C. L. Ratzlaff, N. Gopal, and L. T. Pillage, "RICE: Rapid interconnect circuit evaluator," in *Proc. IEEE/ACM Design Automation Conf.*, June 1991, pp. 555–560.
[24] T. K. Tang and M. S. Nakhla, "Analysis of high-speed VLSI interconnects using the asymptotic waveform evaluation techniques," *IEEE Trans. Computer-Aided Design*, vol. 11, pp. 341–352, Mar. 1992.
[25] P. Feldmann and R. W. Freund, "Efficient linear circuit analysis by Pade approximation via the Lancozos process," *IEEE Trans. Computer-Aided Design*, vol. 14, pp. 639–649, May 1995.
[26] ——, "Reduced-Order modeling of large linear subcircuits via block Lanczos algorithm," in *Proc. IEEE/ACM Design Automation Conf.*, June 1995, pp. 474–479.
[27] M. Silveira, M. Kamon, and J. White, "Efficient reduced-order modeling of frequency-dependent coupling inductances associated with 3-D interconnect structures," in *Proc. IEEE/ACM Design Automation Conf.*, June 1995, pp. 376–380.
[28] D. L. Boley, "Krylov space methods on state-space control models," *J. Circuits, Syst. and Signal Processing*, vol. 13, pp. 733–758, May 1994.
[29] A. Odabasioglu, M. Celik, and L. T. Pillage, "PRIMA: Passive reduced-order interconnect macromodeling algorithm," *IEEE Trans. Computer-Aided Design*, vol. 17, pp. 645–654, Aug. 1998.
[30] ——, "PRIMA: Passive reduced-order interconnect macromodeling algorithm," in *Proc. IEEE/ACM Int. Conf. Computer-Aided Design*, Nov. 1997, pp. 58–65.
[31] P. Feldmann and R. W. Freund, "Reduced-Order modeling of large passive linear circuits by means of the SyPVL algorithm," in *Proc. IEEE/ACM Int. Conf. Computer-Aided Design*, Nov. 1996, pp. 280–287.
[32] "Fast and accurate method for simulating VLSI interconnect," Patent Application Pending..
[33] B. C. Kuo, *Automatic Control Systems, A Design Perspective*. New Delhi, India: Prentice Hall, 1989.
[34] B. J. Ley, S. G. Lutz, and C. F. Rehberg, *Linear Circuit Analysis*. New York: McGraw-Hill, 1959.
[35] C. A. Desoer and E. S. Kuh, *Basic Circuit Theory*. New York: McGraw-Hill, 1969.
[36] G. F. Paskusz and B. Bussell, *Linear Circuit Analysis*. Englewood Cliffs, NJ: Prentice Hall, 1963.
[37] R. E. Scott, .
[38] L. P. P. P. van Ginneken, "Buffer placement in distributed RC-tree networks for minimal elmore delay," in *Proc. IEEE Int. Symp. Circuits Syst.*, May 1990, pp. 865–868.
[39] C. J. Alpert, "Wire segmenting for improved buffer insertion," in *Proc. IEEE/ACM Design Automation Conf.*, June 1997, pp. 588–593.
[40] P. Rabinowitz, *Numerical Methods for Non-Linear Algebraic Equations*. London, U.K.: Gordon and Breach Science, 1970.
[41] M. Daehlen and A. Tveito, *Numerical Methods and Software Tools in Industrial Mathematics*. Boston, MA: Birkhauser, 1997.
[42] S. Winograd, *Arithmetic Complexity of Computations*. Bristol, U.K.: J. W. Arrowsmith, 1980.
[43] R. C. Agarwal and J. W. Cooley, "New algorithms for digital convolution," *IEEE Trans. Acoustics, Speech Signal Processing*, vol. 25, pp. 392–410, May 1995.
[44] A. V. Aho, J. E. Hopcroft, and J. D. Ullman, *The Design and Analysis of Computer Algorithms*. Reading, MA: Addison-Wesley, 1976.

Yehea I. Ismail (M'01) was born in Giza, Egypt, on November 11, 1971. He received the B.Sc. degree in electronics and communications engineering, with distinction and honors, from the School of Engineering, Cairo, Egypt. He received the M.S. degree in electronics from Cairo University (with distinction), Egypt, June 1996. He received a second M.S. degree in electrical engineering from the University of Rochester, Rochester, NY, in 1998, and the Ph.D. degree, in April 2000, from the same university.

He is currently with Northwestern University as an Assistant Professor. He has authored more than 35 technical papers and a book. He was with IBM Cairo Scientific Center (CSC) from 1993 to 1996 on a part-time basis and worked with IBM Microelectronics at East Fishkill, NY, the summers of 1997, 1998, and 1999. His primary research interests include interconnect, noise, innovative circuit simulation, and related circuit level issues in high performance VLSI circuits.

Dr. Ismail is an Associate Editor in the IEEE TRANSACTIONS ON VERY LARGE SCALE INTEGRATION SYSTEMS, an Associate Editor of the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS, and a Guest Editor for a special issue of the IEEE TRANSACTIONS ON VERY LARGE SCALE INTEGRATION SYSTEMSon On-Chip Inductance in High Speed Integrated Circuits.

Eby G. Friedman (S'78–M'79–SM'90–F'00) received the B.S. degree from Lafayette College, in 1979, and the M.S. and Ph.D. degrees from the University of California, Irvine, in 1981 and 1989, respectively, all in electrical engineering.

From 1979 to 1991, he was with Hughes Aircraft Company, rising to the position of Manager of the Signal Processing Design and Test Department, responsible for the design and test of high performance digital and analog IC's. He has been with the Department of Electrical and Computer Engineering at the University of Rochester since 1991, where he is a Professor, the Director of the High Performance VLSI/ 1C Design and Analysis Laboratory, and the Director of the Center for Electronic Imaging Systems. His current research and teaching interests include high-performance synchronous digital and mixed-signal microelectronic design and analysis with application to high-speed portable processors and low-power wireless communications. He is the author of more than 150 papers and book chapters and the author or editor of six books in the fields of high-speed and low-power CMOS design techniques, high speed interconnect, and the theory and application of synchronous clock distribution networks.

Dr. Friedman is the Editor-in-Chief of the IEEE TRANSACTIONS ON VERY LARGE SCALE INTEGRATION SYSTEMS, Regional Editor of the *Journal of Circuits, Systems, and Computers*, a Member of the editorial boards of IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS II: Analog and Digital Signal Processing, *Analog Integrated Circuits and Signal Processing*, and *Journal of VLSI Signal Processing*, a Member of the Circuits and Systems (CAS) Society Board of Governors, and a Member of the technical program committee of a number of conferences. He previously was a Member of the editorial board of the TRANSACTIONS ON CIRCUITS AND SYSTEMS II, Chair of the IEEE TRANSACTIONS ON VERY LARGE SCALE INTEGRATION SYSTEMS steering committee, CAS liaison to the Solid-State Circuits Society, Chair of the VLSI Systems and Applications CAS Technical Committee, Chair of the Electron Devices Chapter of the IEEE Rochester Section, Program or Technical Chair of several IEEE conferences, Editor of several special issues in a variety of journals, and a recipient of the Howard Hughes Masters and Doctoral Fellowships, an IBM University Research Award, an Outstanding IEEE Chapter Chairman Award, and a University of Rochester College of Engineering Teaching Excellence Award. He is also a Senior Fulbright Fellow.