SILCA: SPICE-Accurate Iterative Linear-Centric Analysis for Efficient Time-Domain Simulation of VLSI Circuits With Strong Parasitic Couplings

Zhao Li and C.-J. Richard Shi, *Fellow, IEEE*

***Abstract*—A new circuit analysis method, named SPICE-accurate iterative linear-centric analysis (SILCA), is proposed for the efficient and accurate time-domain simulation of deep submicron very large scale integrated (VLSI) circuits with strong parasitic couplings. SILCA consists of two key linear-centric techniques applied to time-domain nonlinear circuit simulation. For numerical integration, explicit-formula substitution and iterative-formula transformation are presented to convert implicit variable time-step integration to fixed leading coefficient (FLC) variable time-step integration. This paper characterizes both convergence and stability properties of the resulting FLC integration formulae. For nonlinear iteration, a successive variable chord (SVC) method is used as an alternative to the Newton–Raphson method. Further, the low-rank update technique is implemented for fast LU factorization. With these techniques, the number and cost of required LU factorizations are reduced dramatically. Experimental results on nonlinear circuits coupled with substrate and power/ground networks have demonstrated that SILCA achieves more than an order of magnitude speedup over SPICE3 in terms of both the cost of LU factorization and the overall CPU time. SILCA is suitable for efficient SPICE-like time-domain simulation of parasitic-coupled VLSI circuits, where the number of linear parasitic elements dominates the number of nonlinear devices.**

***Index Terms*—Circuit simulation, time-domain analysis.**

I. Introduction

With increasing operation frequency, lower supply voltage, and smaller device feature size, parasitic coupling effects are becoming more and more important for modern deep submicron very large scale integrated (VLSI) circuit designs [24]. The increasing demand to integrate digital, analog, and radio frequency (RF) circuits into one single chip requires accurate analysis of very large scale integrated (VLSI) circuits together with extracted parasitic elements arising from interconnect lines, common substrate, power/ground networks, etc. [1], [20], [24], [30], [32]. Meanwhile, on-chip and packaging inductances are no longer ignorable for accurate circuit analysis [8]. For such purposes as well as coupled circuit and electromagnetic modeling [33], SPICE-like simulators are desirable for accurate transistor-level time-domain simulation.

However, efficient simulation of such systems presents a complexity challenge to SPICE [21]. For time-domain circuit simulation, SPICE uses numerical integration formulae [2], [19] to form companion models for capacitors and inductors at each time point, and applies the Newton–Raphson method [19] to linearize nonlinear devices. Then the circuit is simulated at each time point by iteratively solving a system of linearized equations in the form of $\boldsymbol{Ax} = \boldsymbol{b}$, where $\boldsymbol{A}$ is typically the so-called modified nodal analysis (MNA) circuit matrix [19], [21] which is a Jacobian matrix. It is known that device evaluation dominates simulation of small to medium size circuits, and its cost can be reduced with device bypass [14], [21], table lookup [1], parallel computation techniques [13], etc. However, for a system with strong parasitic couplings, the per-iteration cost of SPICE time-domain simulation is dominated by LU factorization [19] of the circuit matrix $\boldsymbol{A}$. In practice, the cost for LU factorization by sparse matrix solvers [12] is $O(n^{1.1\sim1.5})$ for sparse circuits, where n is the circuit matrix size. However, strong parasitic couplings present in deep submicron circuits can cause the circuit matrix to become much denser, even with model order reduction [22], [28] the cost of LU factorization can approach its worst case $O(n^3)$ [24].

One key idea to improve the efficiency of SPICE-like circuit simulation is to keep the circuit matrix as constant as possible during the entire time-domain simulation and, therefore, reduce the number of LU factorizations required. This has been implemented in both numerical integration and nonlinear iteration stages. For numerical integration, several strategies have been proposed on reformulating the backward differentiation formulae (BDF) [2], [19] to keep the leading coefficient constant since it is the leading coefficient that contributes to a Jacobian matrix. These include fixed coefficient methods [9] (i.e., in LSODE [25]), fixed leading coefficient (FLC) methods [10] (i.e., in DASSL [23]), overdetermined polynomial methods (ODPM) [5], etc. All these methods have been shown to be effective in bypassing Jacobian matrix factorization. However, the stability of fixed coefficient and FLC methods is worse than that of variable coefficient methods [10]. Furthermore, for fixed coefficient and FLC methods, interpolation must be performed at each time point, which will unfortunately introduce extra errors and increase simulation cost. The overdetermined

Manuscript received May 4, 2003; revised February 12, 2004, December 23, 2004, March 7, 2005, and April 28, 2005. This work was supported in part by the U.S. Defense Advanced Research Projects Agency NeoCAD Program under Grant 66001-01-1-8920, in part by the National Science Foundation (NSF) CAREER Award under Grant 9985507, and in part by the NSF/Semiconductor Research Corporation Joint Mixed-Signal Initiative under Grant CCR0120371. This paper was recommended by Associate Editor S. Sapatnekar.

Z. Li was with the Department of Electrical Engineering, University of Washington, Seattle, WA 98195 USA. He is now with Cadence Design Systems, Inc., San Jose, CA 95134 USA (e-mail: zhaoli@cadence.com).

C.-J. R. Shi is with the Department of Electrical Engineering, University of Washington, Seattle, WA 98195 USA (e-mail: cjshi@ee.washington.edu).

Digital Object Identifier 10.1109/TCAD.2005.855943

polynomial method [5] overcomes the interpolation problem by introducing an extra coefficient in the BDF. However, the stability of the overdetermined polynomial method is worse and the ODPM-1 formula [5] has been shown to be stable only when the present time step-size h_n is less than or equal to the predefined basis time step-size h. Therefore, a large basis time step-size h has been adopted in [5] to have an h_n/h ratio of less than 1. The efficiency of the overdetermined polynomial method is thus limited.

To reduce the number of LU factorizations during the nonlinear iteration process, quasi-Newton methods [6], [29] have been studied extensively and applied in circuit simulation [1] and mixed-mode circuit and device simulation [35]. The successive chord method [19] has been explored for fast transistor-level gate-delay calculation [1], where each transistor is modeled as a fixed linear resistor (called chord) combined with a variable nonlinear current source. Since a fixed chord is used, the circuit matrix will not change during nonlinear iteration and only one LU factorization is required overall if a fixed time step-size is used. Unfortunately, there are two principal difficulties that restrict the success use of this linear-centric idea to the simulation of general VLSI circuits. 1) Most VLSI circuits have widely distributed time constants and require variable time step-size control for simulation efficiency and accuracy. With variable step-sizes, the circuit matrix is no longer constant across time points unless an FLC numerical integration formula as discussed before is used. 2) The successive chord method may need an excessive amount of iterations to converge, and thus offsets the gain from the reduction of LU factorizations.

Recently, in the contexts of power grid analysis [4], substrate analysis [24], and parasitic extraction [11], Krylov-subspace-based iterative methods such as the conjugate gradient algorithm, generalized minimum residual (GMRES) algorithm, etc., have been shown to be more efficient than the method of LU factorization and forward/backward substitution (named the direct method). However, there is no report of successful and robust applications of iterative methods to classical time-domain nonlinear circuit simulation in the literature.

This paper presents SPICE-accurate iterative linear-centric analysis (SILCA), a new direct method capable of analyzing VLSI circuits containing strong parasitic coupling effects with SPICE-like accuracy yet orders of magnitude faster. SILCA consists of applying the linear-centric principle to both numerical integration and nonlinear iteration to keep circuit matrices as constant as possible during variable step-size time-domain nonlinear circuit simulation.

- Two general techniques, namely explicit-formula substitution and iterative-formula transformation, are presented to convert implicit integration formulae in SPICE-like simulators to FLC integration formulae. These formulae lead to constant equivalent conductance in capacitor/inductor companion models.
- Successive variable chord (SVC) method, a variant of the successive chord method, is introduced to keep linearized conductance of nonlinear devices constant for a larger voltage/current range by incorporating device-related behavioral knowledge. With the SVC method, a piecewise weakly nonlinear (PWNL) MOSFET model is introduced for the calculation of Jacobian matrices. The low-rank update technique is further applied for fast LU factorization by noting the fact that the number of nonlinear devices, switching operating PWNL regions, is only few at a single time point.

With these, the number of required LU factorizations can be reduced by orders of magnitude with a moderate increase of iterations. Thus, rather than solving a newly linearized system by another costly LU factorization, we are able to achieve the same accurate results by several efficient forward/backward substitutions on a previously linearized system. The entire method is robust, accurate, and has been implemented into SPICE3. Further, the proposed method is compatible with other circuit analysis methods, such as model order reduction [22], [28], to achieve even greater simulation speedup.

Some preliminary results of this paper were presented in [16]. The rest of this paper is organized as follows. Section II presents new FLC integration schemes, the analysis of their stability and convergence properties, and methods for adaptive step-size control. Section III presents the SVC method and the low-rank update technique. The SILCA algorithm is described in Section IV. Section V shows experimental results on substrate and power/ground coupling analyses. Section VI concludes the paper.

II. FLC Integration Schemes

In this section, we present and characterize two general techniques of taking any implicit integration formula to derive such an integration formula that yields a constant circuit matrix for variable step-size time-domain circuit simulation. Mathematically, let $x_n, x_{n-1}, \ldots, x_0$ and $\dot{x}_n, \dot{x}_{n-1}, \ldots, \dot{x}_0$ be the values and first-order time derivatives of variable x at time points $t_n, t_{n-1}, \ldots, t_0$, then any linear multistep numerical integration formula implemented in SPICE-like simulators can be written in the general form

$$\dot{x}_n = \sum_{i=0}^{k} a_i x_{n-i} + \sum_{j=1}^{l} b_j \dot{x}_{n-j} \tag{1}$$

where $a_i,\ i = 0, 1, \ldots, k$, and $b_j,\ j = 1, 2, \ldots, l$, are coefficients of the integration formula, the leading coefficient a_0 is nonzero (hence implicit), and $h_n = t_n - t_{n-1}$ is the current time step. Let h be some kind of basis time step-size, the current time step-size can be rewritten as $h_n = \alpha h$, where α is a positive real number.

Notice that only the leading coefficient a_0 contributes to the circuit matrix. In general, a_0 is a function of αh. Since α changes with time points, the circuit matrix would change. To keep the circuit matrix constant, we rewrite the integration formula above as

$$\dot{x}_n = a_0(h) x_n + a_0(\alpha h) x_n + \sum_{i=1}^{k} a_i x_{n-i} + \sum_{j=1}^{l} b_j \dot{x}_{n-j} \tag{2}$$

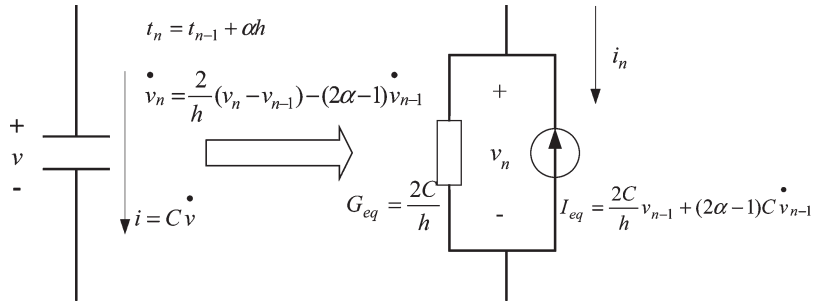


Fig. 1. Capacitor companion model using the mixed trapezoid FE formula.

where $a_0(h)$ is independent of α. Then, we would like to substitute x_n in the second term by all the known values from the previous time points. The first technique is to replace x_n in the second term using an explicit integration formula. This is called explicit-formula substitution. The second technique is to replace x_n in the second term using an initial guess and then iterate to convergence. This is called iterative-formula transformation. With these, the resulting formulae have an FLC and are referred to as FLC integration formulae, following the convention of Jackson and Sacks-Davis [10].

In the following subsections, we use the standard trapezoid formula as an example to derive FLC integration formulae based on explicit-formula substitution and iterative-formula transformation. We characterize both stability and convergence properties of the resulting integration formulae. We note that these derivation and analyses can be applied to any implicit integration formula used in a circuit simulator. Furthermore, we present how the resulting formulae can be used in a way similar to the classical predictor-corrector integration scheme, how to adaptively control the basis time step-size, and how to control stability.

A. FLC Integration by Explicit-Formula Substitution

With $h_n = \alpha h$, the standard trapezoid formula can be rewritten as

$$\begin{aligned}\dot{x}_n &\approx \frac{2}{h_n}(x_n - x_{n-1}) - \dot{x}_{n-1} \\ &= \frac{2}{\alpha h}(x_n - x_{n-1}) - \dot{x}_{n-1} \\ &= \frac{2\alpha - (2\alpha - 2)}{\alpha h}(x_n - x_{n-1}) - \dot{x}_{n-1} \\ &= \frac{2}{h}x_n - \frac{2}{h}x_{n-1} - \frac{2\alpha - 2}{\alpha h}(x_n - x_{n-1}) - \dot{x}_{n-1}. \end{aligned} \tag{3}$$

Now we would like to substitute x_n in the third term by using any explicit Adams–Bashforth formula [19]. The simplest is the forward Euler (FE) formula with a step-size αh defined as

$$x_n \approx x_{n-1} + \alpha h\, \dot{x}_{n-1}. \tag{4}$$

After the x_n in the third term in (3) is approximated by (4), the mixed trapezoid FE formula with a time step-size $h_n = \alpha h$ is obtained as

$$\dot{x}_n = \frac{2}{h}x_n - \frac{2}{h}x_{n-1} - (2\alpha - 1)\dot{x}_{n-1}. \tag{5}$$

The mixed trapezoid FE formula is an implicit integration formula. When $\alpha = 1$, it reduces to the standard trapezoid formula. When $\alpha = 1/2$, it represents the backward Euler (BE) formula with a step-size $h/2$.

To see the circuit interpretation of the mixed trapezoid FE formula (5), the companion model of a linear capacitor is shown in Fig. 1. Note that even though the actual time step-size is αh, the equivalent conductance of the companion model is constant as long as the basis time step-size h is a constant.

The local truncation error (LTE) measures how closely a numerical integration formula approximates the differential operator. We can prove the following result.

Theorem 1: The LTE ε of the mixed trapezoid FE formula (5) with time step-size αh is given by

$$\varepsilon = \left(1 - \frac{1}{\alpha}\right)\left(\frac{\dddot{x}_\xi}{2}\right)(\alpha h)^2 + \left(1 - \frac{1.5}{\alpha}\right)\left(\frac{\ddot{x}_\xi}{6}\right)(\alpha h)^3 \tag{6}$$

where t_ξ is between t_n and t_{n-1}.

Proof: The proof is similar to the LTE estimation for the standard trapezoid formula [19]. ■

According to Theorem 1, when $\alpha = 1$, the LTE of the mixed trapezoid FE formula reduces to that of the standard trapezoid formula. When $\alpha = 1/2$, it represents the LTE of the BE formula using time step-size $h/2$. The mixed trapezoid FE formula is a second-order integration formula only if $\alpha = 1$ and degenerates to a first-order formula if $\alpha \neq 1$.

In contrast to LTE, stability is a global property related to the growth or decay of the local error introduced at each time point and propagated to the following time points. "Absolute stability" requires that $|\varepsilon_n| < |\varepsilon_{n-1}|$. It is often studied with the use of an *RC* test circuit as shown in [19, Fig. 5.1]. The stability property of the mixed trapezoid FE formula can be proved as below.

Theorem 2: The absolute stability region of the mixed trapezoid FE formula (5) with time step-size αh is defined by

$$\left|\frac{1 + (2\alpha - 1)z}{1 - z}\right| < 1 \tag{7}$$

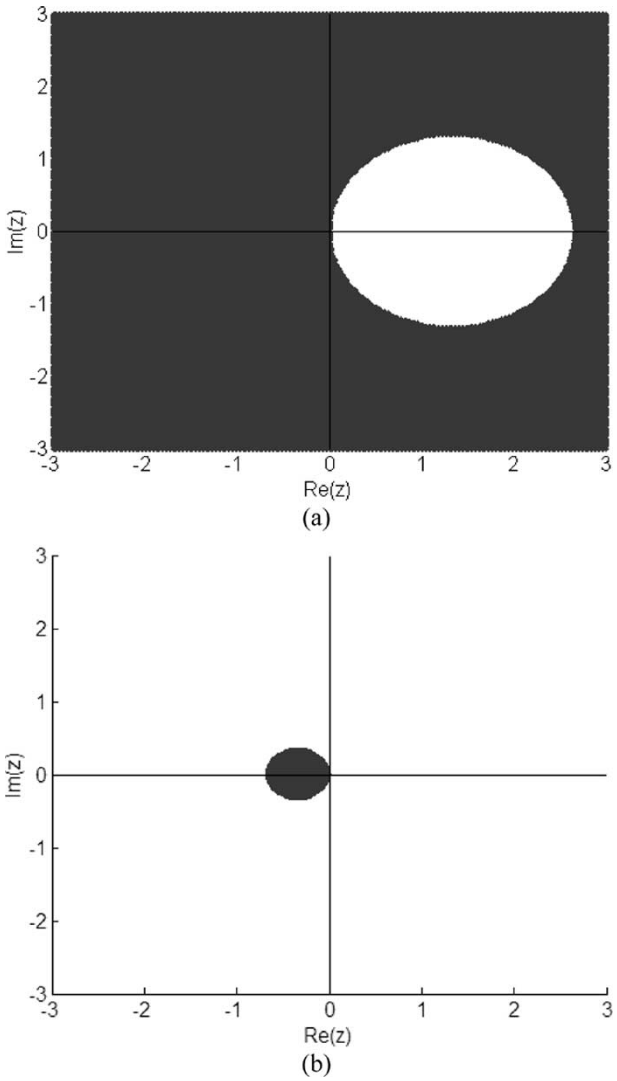


Fig. 2. Absolute stability regions of the mixed trapezoid FE formula for (a) $\alpha = 0.625$ and (b) $\alpha = 2.5$.

where $z = -h/(2\tau)$ and τ is the time constant of the *RC* test circuit.

Proof: The proof is similar to the stability analysis for the standard trapezoid formula [19]. ■

The absolute stability regions for $\alpha = 0.625$ and $\alpha = 2.5$ are shown in Fig. 2(a) and (b), respectively. From Theorem 2, two observations can be made on the stability of the mixed trapezoid FE formula.

- When $\alpha > 1$, the absolute stability region moves closer to that of the FE formula. The mixed trapezoid FE formula is not A-stable [19] or stiff stable [9], [19], and cannot be used as a variable time step-size control scheme when $\alpha > 1$.
- When $\alpha < 1$, the absolute stability region includes the open left half plane of the complex z-plane and the mixed trapezoid FE formula is A-stable. When α approaches 1/2, the absolute stability region approaches that of the BE formula. Further, the smaller α, the better the stability. However, according to Theorem 1, for a fixed time step-size $h_n = \alpha h$, a small α will unfortunately result in a large LTE. Therefore, there exists a tradeoff between the stability and the LTE.

Due to the mentioned LTE and stability problems, FLC integration formulae derived from explicit-formula substitution are not suggested for analog circuit simulation with high accuracy requirements. However, they can be used to enhance timing analysis of digital circuits, for example, TETA [1].

B. FLC Integration by Iterative-Formula Transformation

The LTE and stability problems of the mixed trapezoid FE formula come from the replacement of x_n in the third term of (3) by an approximate x_n defined using the explicit FE formula (4). In this subsection, rather than using explicit integration formulae, x_n in the third term of (3) is replaced by the $(k-1)$th iteration solution $x_n^{(k-1)}$ at the present time point and a new kth iteration solution $x_n^{(k)}$ is obtained by solving (3), where k is the iteration number. This leads to the iterative-formula transformation of (3), called the iterative trapezoid formula, written as

$$\begin{aligned}\dot{x}_n^{(k)} &= \frac{2}{h}x_n^{(k)} - \frac{2}{h}x_{n-1} - \frac{2\alpha-2}{\alpha h}\left(x_n^{(k-1)} - x_{n-1}\right) - \dot{x}_{n-1} \\ &= \frac{2}{h}x_n^{(k)} - \frac{2}{h}x_n^{(k-1)} + 2\frac{x_n^{(k-1)} - x_{n-1}}{\alpha h} - \dot{x}_{n-1}. \end{aligned} \tag{8}$$

The final solution is said to be converged if $|x_n^{(k)} - x_n^{(k-1)}|$ is less than a predefined error tolerance. If the iterative trapezoid formula (8) converges, its LTE will approach that of the standard trapezoid formula.

Next, we characterize both convergence and stability properties of the iterative integration formula. To study the convergence property, let us write the linear(ized) circuit equation as used in [22] as

$$\boldsymbol{G}\boldsymbol{x} + \boldsymbol{C}\dot{\boldsymbol{x}} = \boldsymbol{b} \tag{9}$$

where $\boldsymbol{G}$ and $\boldsymbol{C}$ represent the conductance and capacitance (susceptance) matrices, and $\boldsymbol{b}$ is the vector due to input sources and nonlinear devices. Replacing first-order time derivatives by the iterative trapezoid formula (8), we have

$$\begin{aligned}&\left(\boldsymbol{G} + \frac{2\boldsymbol{C}}{h}\right)\boldsymbol{x}_n^{(k)} \\ &\quad = \left(1 - \frac{1}{\alpha}\right)\frac{2\boldsymbol{C}}{h}\boldsymbol{x}_n^{(k-1)} + \frac{2\boldsymbol{C}}{\alpha h}\boldsymbol{x}_{n-1} + \boldsymbol{C}\dot{\boldsymbol{x}}_{n-1} + b. \end{aligned} \tag{10}$$

Clearly, the iterative trapezoid formula converges if

$$\left\| \left(\boldsymbol{G} + \frac{2\boldsymbol{C}}{h}\right)^{-1} \left(1 - \frac{1}{\alpha}\right)\frac{2\boldsymbol{C}}{h} \right\| < 1 \tag{11}$$

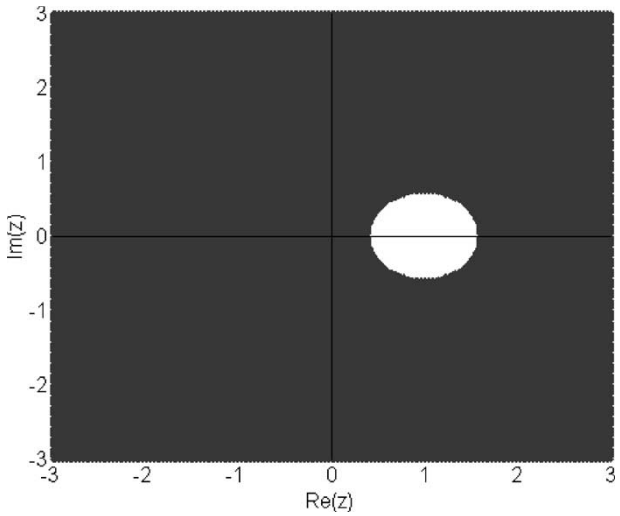


Fig. 3. Convergence region of the iterative trapezoid formula for $\alpha = 0.625$ and 2.5.

where $\| \bullet \|$ represents the spectral radius of the iteration matrix. The above (11) can be rewritten as

$$\left| \frac{1 - \frac{1}{\alpha}}{1 - z} \right| < 1 \tag{12}$$

where $z = -h/(2\tau)$ and τ is an eigenvalue of the matrix $\boldsymbol{G}^{-1}\boldsymbol{C}$. τ represents the time constant of the *RC* test circuit.

With this, we can show the following generalized convergence property.

Theorem 3: The convergence region of the iterative trapezoid formula (8) with a time step-size αh is defined by (12).

From (12), to ensure that the iterative trapezoid formula converges for any decaying or stable oscillatory system $(\mathrm{Re}(z) \leq 0)$, i.e., to have the convergence region include all of the left half of the complex z-plane, we must choose $\alpha > 0.5$. In our implementation, to speed up the convergence, $0.625 < \alpha < 2.5$ is used in our experiments. The convergence region for $\alpha = 0.625$ and 2.5 is shown in Fig. 3. It represents the worst-case convergence region for $0.625 < \alpha < 2.5$. In practice, a maximum iteration number limit for each iteration step is set. In case that the iteration number exceeds the maximum limit (due to either slow convergence or nonconvergence), the solution process with the same time step-size will be attempted one more time with the standard trapezoid formula before the time step-size is decreased.

Theoretically, an iterative implicit integration formula shall have the same stability as the corresponding original implicit formula if the iterative implicit integration formula is solved exactly (iterated to infinity). However, in practice, the iterative implicit integration formula is terminated either when the iteration number exceeds a predefined maximum limit or when the convergence criteria is met with the predefined error tolerance. In such case, the stability of the iterative formula can deteriorate. The stability of the iterative trapezoid formula (8) can be characterized by the following theorem.

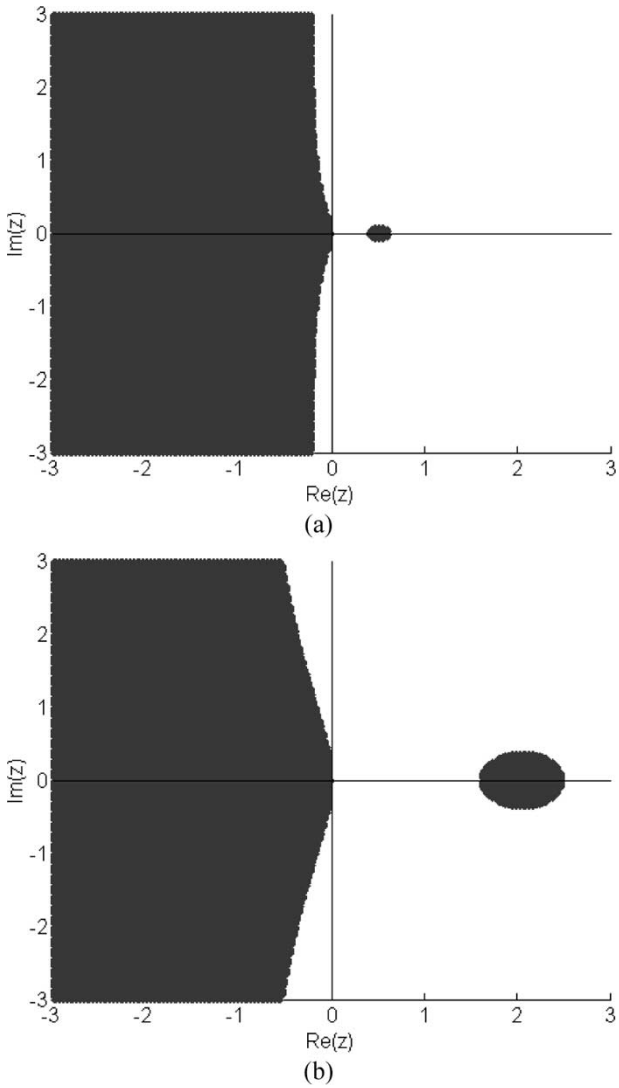


Fig. 4. Absolute stability regions of the iterative trapezoid formula with $k = 2$ for (a) $\alpha = 0.625$ and (b) $\alpha = 2.5$.

Theorem 4: The absolute stability region of the iterative trapezoid integration formula (8) starting with $x_n^{(0)} = x_{n-1}$ with a time step-size αh is defined by

$$\left| \left(\frac{1 - \frac{1}{\alpha}}{1 - z} \right)^k \left(\frac{2z}{z - \frac{1}{\alpha}} \right) + \frac{\frac{1}{\alpha} + z}{\frac{1}{\alpha} - z} \right| < 1 \tag{13}$$

where $z = -h/(2\tau)$ and τ is the time constant of the *RC* test circuit.

Proof: The proof is given in the Appendix. ■

The absolute stability regions for $\alpha = 0.625$ and $\alpha = 2.5$ with $k = 2$ are shown in Fig. 4(a) and (b), respectively, which can be proven to satisfy the "stiff stability" requirements suggested by Gear [9]. For a fixed iteration number k, the absolute stability region of the iterative trapezoid formula will approach that of the standard trapezoid integration formula with $\alpha \to 1$. Furthermore, we give the following stability property of the iterative trapezoid formula.

Theorem 5: When $k \to +\infty$, the absolute stability region of the iterative trapezoid formula (8) includes the entire open left

half of the complex z-plane and excludes the entire right half of the complex z-plane.

Theorem 5 can be interpreted by noting the following fact: (8) is mathematically equivalent to applying the standard trapezoidal method with basis time step-size h to the differential equation, and then using a quasi-Newton method [6], [29] to solve the resulting equation. Therefore, when (8) is solved exactly $(k \to +\infty)$, the absolute stability region of the iterative trapezoid formula will be the same as that of the standard trapezoid formula. This can also be verified by setting $k \to +\infty$ in (13).

C. FLC Integration by Predictor-Corrector Scheme

In SILCA, we first apply (5) and then apply (8) in a way similar to the classical predictor and corrector procedure [9]. Noting how (5) is derived, we can see that applying (5) as a predictor and (8) as an iterative corrector with k iterations is mathematically equivalent to applying an explicit predictor (the FE formula in this case) as a predictor and (8) as an iterative corrector with $k+1$ iterations.

Using (5) to predict an initial guess for the iterative trapezoidal formula (8) can lead to faster convergence than using the previous time-point value as the initial guess for (8). Very often, we may choose to carry (8) for one or a finite number of iterations, and then use the LTE to adjust time step-sizes, similar to what is done in the classical predictor and corrector procedure. In this case, the predictor–corrector use leads to the stability region worse than that of applying only (8). We can prove the following result.

Theorem 6: The absolute stability region of applying (5) as a predictor and (8) as an iterative corrector with iteration number k is defined by

$$\left| \left(\frac{1-\frac{1}{\alpha}}{1-z} \right)^{k+1} \left(\frac{2\alpha z^2}{z-\frac{1}{\alpha}} \right) + \frac{\frac{1}{\alpha}+z}{\frac{1}{\alpha}-z} \right| < 1 \qquad (14)$$

where $z = -h/(2\tau)$ and τ is the time constant of the *RC* test circuit.

Proof: The proof is given in the Appendix. ■

The absolute stability regions for $\alpha = 0.625$ and 2.5 with $k = 1$ are shown in Fig. 5(a) and (b). Compared to Fig. 4, it can be seen that when $\alpha = 0.625$, the absolute stability region of the predictor–corrector scheme is larger than that of applying (8) alone. However, when $\alpha = 2.5$, the predictor–corrector scheme becomes less stable than applying (8) alone, and it is even no longer stiff stable. Therefore, in SILCA, to ensure stability, (5) is applied as a predictor only if $\alpha < 1$.

D. Illustration of Basis Time Step-Size (h) Control

As discussed in Section II-B, to satisfy the convergence property defined by Theorem 3, $0.5 < \alpha < +\infty$ is required. The limited α range means that it is impossible to use only one single basis time step-size during transient simulation in our framework. When h_n/h is out of the α range, a new basis time step-size has to be chosen (i.e., the present time step-size h_n), which means the circuit matrix has to be updated and a new

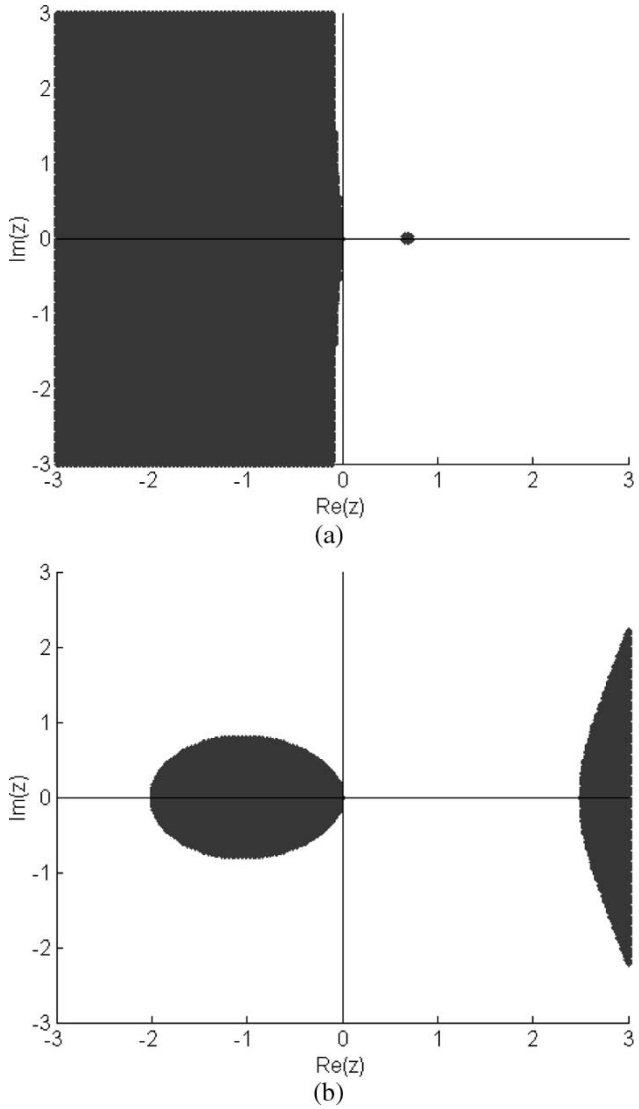


Fig. 5. Absolute stability regions of the predictor–corrector scheme with $k = 1$ for (a) $\alpha = 0.625$ and (b) $\alpha = 2.5$.

LU factorization is required. In this sense, a large α range is preferred to decrease the total number of LU factorizations. However, according to Theorem 3, the linear convergence rate of the iterative trapezoid formula is related to $|1 - 1/\alpha|$ and a smaller $|1 - 1/\alpha|$ means a faster convergence rate. Obviously, a small α range is ideal to reduce the total number of iterations. Therefore, in practice, $0.625 < \alpha < 2.5$ $(|1 - 1/\alpha| < 0.6)$ is chosen to achieve a balance between the number of LU factorizations and the number of iterations, which will reduce the error to less than 5% of the original error after six iteration steps. Considering that SPICE3 needs at least two iteration steps to converge at a time point, the number of iteration steps with SILCA is approximately $3\times$ over that with SPICE3 for general circuits. The detailed basis time step-size control scheme will be described in Algorithm I of Section IV and a linear circuit example is shown in Section V to illustrate the efficiency and validity of the iterative trapezoid formula.

It should be noted that SILCA could be combined with fixed time step-size methods to enlarge the range of α. As shown in the Appendix, when $\alpha > 1$, the first iteration with the iterative

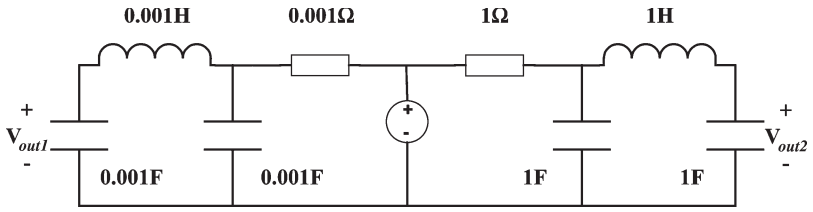


Fig. 6. Linear *RCL* circuit example.

trapezoid formula is equivalent to applying the standard trapezoid formula with the basis time step-size h. Then, one idea is to apply the standard trapezoid formula with the basis time step-size h for multiple iterations if α is large. For example, if $\alpha = 3.5$, we could apply the standard trapezoid formula with the basis time step-size h for the first two iterations and then the iterative trapezoid formula for the rest iterations with $\alpha = 3.5 - 1 = 2.5$. By this way, the convergence and stability properties will not be affected since the range of α for the iterative trapezoid formula is kept unchanged ($\alpha = 2.5$ for the previous example). The extra cost is that more iterations will be required with more step-sizes performed by a fixed time step-size method.

E. Illustration of Stability Control

As discussed in Section II-B, the iterative trapezoid formula satisfies the stiff stability [9], [19] and is applicable to stiff circuits [19], [34], such as *RC* circuits, as long as circuit poles are not so close to the imaginary axis of the complex z-plane. For oscillatory circuits with poles close to the imaginary axis in the complex z-plane, according to Theorem 3, the absolute stability region of the iterative trapezoid formula with a finite iteration number k will become worse than that of the standard trapezoid formula. This can be illustrated using the linear *RCL* circuit example in Fig. 6.

The transfer function of $V_{\rm out}$ for the *RCL* circuit shown in Fig. 6 can be written as

$$H(s) = \frac{V_{\rm out}}{V_{\rm in}} = \frac{1}{RC^2Ls^3 + CLs^2 + 2RCs + 1}. \qquad (15)$$

There are three poles for $V_{\rm out2}$— -0.5689 and $-0.2151 \pm j1.3071$, and three poles for $V_{\rm out1}$— -999999 and $-0.5 \pm j1000$. Noting that $z = (h\lambda)/2$, among these six poles, -0.5689, -999999, and $-0.2151 \pm j1.3071$ are on or close to the negative real axis of the complex z-plane, therefore they will not cause stability problems since the iterative trapezoid formula has the stiff stability property. However, the rest of the two poles $-0.5 \pm j1000$ are far away from the negative real axis and close to the imaginary axis of the complex z-plane, which may not be covered by the absolute stability region when convergence is achieved in a small iteration number k (i.e., the blank region in the left half of the complex z-plane as shown in Fig. 4(a) and (b) for $k = 2$). The simulation results with SPICE3 and SILCA (without the stability control) are shown in Fig. 7. Unstable simulation results are observed with SILCA and the number of iteration steps with SILCA is $2\times$ of that with SPICE3. It should be noted that BDF [19] with the order

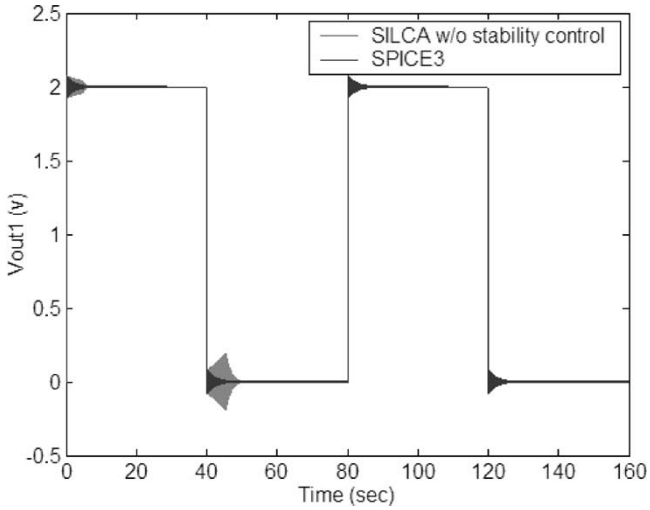


Fig. 7. Time-domain output waveform of $V_{\rm out1}$ for a linear *RCL* circuit example.

larger than two have the same stability problem for oscillatory circuits.

This can be explained by comparing Figs. 3 and 4, which show that the stability region is smaller than the convergence region. In other words, the stability region might not cover all circuit poles upon convergence if the stability requirement (Theorem 4 or Theorem 6) is stricter than the convergence requirement (i.e., user-specified error tolerance for convergence justification). A tighter error tolerance for convergence will help alleviate the stability problem. However, more iterations and/or more time points have to be simulated. Therefore, SILCA is not recommended for highly oscillatory circuits.

III. SVC Method

SPICE-like circuit simulators use the Newton–Raphson method to solve a set of nonlinear equations. Typically, for each Newton–Raphson iteration, a new LU factorization is required. This can be extremely costly for a circuit with strong parasitic coupling effects or with reduced dense linear networks. The successive chord method [19] always uses a fixed chord as the first-order derivative during nonlinear iteration. Hence, at each time point, only one LU factorization is needed for nonlinear iteration. But it is often hard to choose a single fixed chord for a (strongly) nonlinear curve to always ensure a good convergence rate. In general, a chord that ensures global convergence will unfortunately lead to a slow convergence rate.

To achieve a good balance between the number of LU factorizations and that of iterations, we propose the SVC method. The basic idea is to divide a nonlinear curve into different

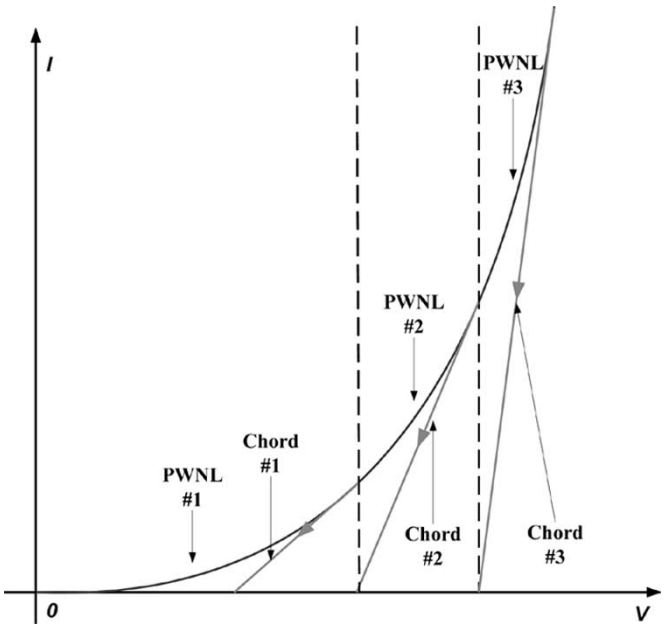


Fig. 8. PWNL example implemented with the SVC method.

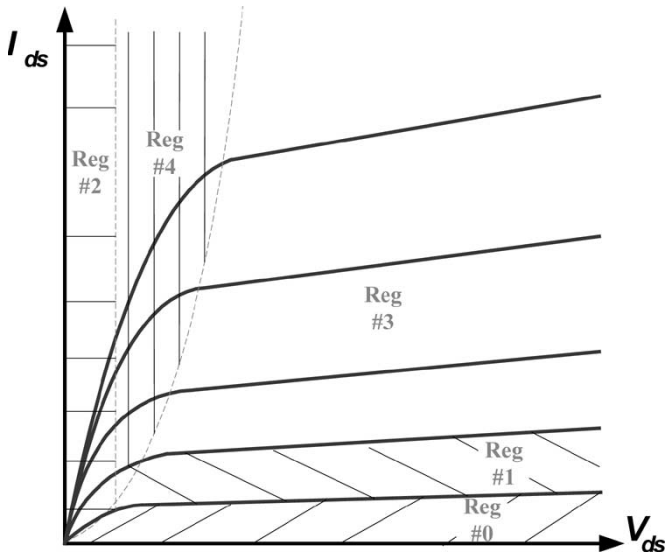


Fig. 9. PWNL operating regions of MOSFETs for digital applications.

segments, each of which represents a weakly nonlinear curve and the same (local) chord is used for the same segment during nonlinear iteration-so-called PWNL analysis. As shown in Fig. 8, the nonlinear curve is divided into three PWNL segments with three local chords, each of which represents the maximum derivative for the corresponding segment. A new LU factorization is performed only if the nonlinear curve enters a different PWNL segment, where a new local chord is used. By this method, similar convergence speed and accuracy can be achieved as the Newton–Raphson method while the number of LU factorizations can be decreased. We emphasize that the PWNL model of a nonlinear device is used only for the calculation of first-order derivatives while the nonlinear function is still evaluated using the original nonlinear device model.

The PWNL idea implemented with the SVC method can be very effective due to the following facts. 1) Since MOSFETs in analog applications generally operate linearly around their operating points, only weakly nonlinear properties may be present. A fixed chord representing the $g_{\rm m}$, $g_{\rm mbs}$, and $g_{\rm ds}$ of MOSFETs at operating points is generally sufficient. A linear-centric harmonic balance analysis method has been proposed in [15]. 2) MOSFETs in digital applications reside in two regions at most time points—cutoff region and well-conducted linear region with a very small source-to-drain voltage, both regions have a relatively constant $g_{\rm m}$, $g_{\rm mbs}$, and $g_{\rm ds}$. The only situation where $g_{\rm m}$, $g_{\rm mbs}$, and $g_{\rm ds}$ change a lot is the time when MOSFETs switch from the cutoff region through the saturation region to the linear region (or vice versa). This process only occupies a small fraction of the total simulation time for a MOSFET in a large-scale digital circuit. Hence, a fixed chord for these situations will not significantly affect the total iteration process.

With the above considerations, five MOSFET PWNL operating regions for digital circuit applications are defined as shown in Fig. 9, and $g_{\rm m}$, $g_{\rm mbs}$, and $g_{\rm ds}$ for different operating regions are listed in Table I. In Table I, Reg#0 represents the cutoff region, Reg#1 and Reg#3 are saturation regions, and Reg#2 and Reg#4 are linear regions. $g_{\rm m-max}$ and $g_{\rm mbs-max}$ are the maximum values in all the regions (defined by $V_{\rm dd}$), and $g_{\rm ds-i}$ is defined (generally the maximum values) for different regions to ensure convergence. It should be noted that, theoretically, the convergence rate of the SVC method is linear, but in practice it can be maintained close to that of the Newton–Raphson method by using more PWNL regions if needed.

TABLE I
$g_{\rm m}$, $g_{\rm mbs}$, AND $g_{\rm ds}$ FOR DIFFERENT MOSFET PWNL REGIONS

	Reg#0	Reg#1	Reg#2	Reg#3	Reg#4
g_m	0	0	0	$g_{m\text{-}max}$	$g_{m\text{-}max}$
g_{mbs}	0	0	0	$g_{mbs\text{-}max}$	$g_{mbs\text{-}max}$
g_{ds}	0	$g_{ds\text{-}1}$	$g_{ds\text{-}2}$	$g_{ds\text{-}3}$	$g_{ds\text{-}4}$

Another advantage of the SVC method is that chords can be precalculated and stored before simulation, no derivative calculation is required during nonlinear iteration as in the Newton–Raphson method. This can lead to a significant saving in device loading time. Furthermore, table lookup models can be easily implemented in SILCA than in SPICE since there is no need of lookup tables for first-order derivatives.

The proposed SVC method is more accurate and effective than *ad hoc* device bypass techniques utilized in modern circuit simulators [14], [21], where device evaluations are bypassed when terminal voltages of a nonlinear device are kept almost constant for a few continuous nonlinear iteration steps. It has been reported that the voltage/current range of device bypass has to be kept small enough to avoid incorrect simulation results [21]. However, for high-frequency deep submicron circuit applications, the efficiency is limited, since terminal voltages of most nonlinear devices are not completely constant but are changing slowly. The SVC method defines PWNL segments and local chords based on the behaviors of specific nonlinear devices under study; therefore, it can keep the circuit matrix constant for a larger voltage/current range and requires much less LU factorizations.

By the above MOSFET PWNL operating region definition, only five sets of $g_{\rm m}$, $g_{\rm mbs}$, and $g_{\rm ds}$ are used during time-domain simulation for digital systems. We further have the following observations. 1) At one time point, most MOSFETs in a large digital system will stay in their PWNL operating regions as defined above while only a few may switch from one region to another region. 2) For a switching MOSFET, the update of $g_{\rm m}$, $g_{\rm mbs}$, and $g_{\rm ds}$ is regionwise. In other words, the change of $g_{\rm m}$, $g_{\rm mbs}$, and $g_{\rm ds}$ from Reg#i to Reg#j is fixed. Therefore, in the case that a small amount of MOSFETs change their PWNL operating regions, we can compute the new L and U matrices directly from the old L and U matrices using the low-rank update technique [7], [31] rather than performing costly LU factorization for the entire circuit matrix.

Suppose that the previous circuit matrix is Y and one MOSFET is now switching from Reg#1 to Reg#2. The new circuit matrix for the next iteration can be expressed by

$$Y' = Y + cr^T \tag{16}$$

where c and r are sparse column vectors representing values of updated elements. In this case, $c = r = [0, \ldots, 0, \ e, \ 0, \ldots, 0, -e, \ 0, \ldots]^{\rm T}$, and $e = \sqrt{|g_{\rm ds-2} - g_{\rm ds-1}|}$. Noting that there are only four different elements between the matrix Y and Y', the new L and U matrices for Y' can be updated from the previous ones for Y efficiently with the low-rank update technique. The worst-case cost of m low-rank updates for a dense matrix is $O(m^* n^2)$, where m is the number of updated elements and n is the matrix size. If m is much less than n, the low-rank update will perform much faster than a regular LU factorization, whose worst-case cost is $O(n^3)$ for a dense matrix. With the introduced MOSFET PWNL definition, m will be kept small enough at a time point since the number of MOSFETs, whose terminal voltages change so violently that the operating region is switched, is generally small.

Furthermore, the low-rank update cost can be decreased dramatically by exploiting sparse matrix techniques [7], [31] and nonlinear/linear circuit partitioning to place matrix elements due to nonlinear devices at the bottom-right corner of a circuit matrix [7]. By this way, only matrix elements whose values need to be updated are recomputed while all other matrix elements are kept the same as before, i.e.,

$$Y = \begin{bmatrix} Y_{\rm lin} & Y_{\rm coup} \\ Y_{\rm coup}^{\rm T} & Y_{\rm non} \end{bmatrix} \quad Y_{\rm non} = \begin{bmatrix} \otimes & & \otimes & & \oplus \\ & \times & & \times & \times \\ \otimes & & \otimes & & \oplus \\ & \times & & \times & \times \\ \oplus & \times & \oplus & \times & \oplus \end{bmatrix}. \tag{17}$$

For example, the circuit matrix Y in (17) is partitioned into the linear part $Y_{\rm lin}$, the nonlinear/linear coupling part $Y_{\rm coup}$, and the nonlinear part $Y_{\rm non}$. Whenever a nonlinear device changes its operating region (affecting four matrix elements of $Y_{\rm non}$ in this example, marked by $\otimes$), $Y_{\rm lin}$ and $Y_{\rm coup}$ are kept the same. For the sparse matrix $Y_{\rm non}$, only nine matrix elements need to be updated (marked by $\otimes$ and $\oplus$) and the other eight are kept unchanged (marked by $\times$). Therefore, the matrix sparsity can be fully exploited by the low-rank update technique.

TABLE II
ALGORITHM FOR SILCA TIME-DOMAIN SIMULATION

```
DC analysis
Choose an initial step-size h_0, the basis step-size h = h_0, t = 0
WHILE (t<T_final){
  OUTER LOOP: do{
    α = h_n/h, iter_no = 0
    INNER LOOP: do{
      IF(0.625<α<2.5){
        IF(α<1 && iter_no==0) {
          Apply mixed Trap-FE integration predictor Eq. (5)
        }
        Apply iterative Trap integration corrector Eq. (8)
      }ELSE{
        IF(iter_no==0) { h = h_n }
        Apply standard Trap integration formula
      }
      Apply the SVC method on nonlinear devices
      IF (0.625<α<2.5) {
        IF (chord is changed) { Apply low-rank update & FBS }
        ELSE { Apply FBS}
      }ELSE{ Apply LU factorization & FBS }
      iter_no = iter_no + 1
    } while (not converged)
    Choose a new h_n based on LTE requirement
  } while (LTE greater than predefined error limit)
  t = t + h_n
}
```

SILCA utilizes the sparse matrix solver package SPARSE1.3 [12], and a sparse low-rank update algorithm has been implemented successfully in SPARSE1.3. In practice, if the value of a diagonal element (L_{ii}) during low-rank updates becomes smaller than the predefined threshold value, the diagonal element will not be suitable for the following steps. In this case, a regular LU factorization is restored.

IV. SILCA ALGORITHM

The basic algorithm for SILCA time-domain simulation is shown in Table II. Practical considerations, such as processing breakpoints [21], are not included for clarity. A new LU factorization is only required if the standard implicit integration scheme is used. In case that only local chords of nonlinear devices change, low-rank update is performed for fast LU factorization. No LU factorization is needed in any other case.

Nonlinear capacitors can be handled in SILCA by combining the proposed iterative trapezoid integration formula and the proposed SVC, illustrated as

$$\begin{aligned} \dot{Q}_n^{(k)} &= \frac{2}{h_n}\left(Q_n^{(k)} - Q_{n-1}\right) - \dot{Q}_{n-1} \\ &\approx \frac{2}{h_n}\left[Q_n^{(k-1)} + C_n^{(k-1)}\left(V_n^{(k)} - V_n^{(k-1)}\right) - Q_{n-1}\right] \\ &\quad - \dot{Q}_{n-1} \\ &= \frac{2C_n^{(k-1)}}{h_n} V_n^{(k)} - \frac{2C_n^{(k-1)}}{h_n} V_n^{(k-1)} \\ &\quad + \frac{2}{h_n}\left(Q_n^{(k-1)} - Q_{n-1}\right) - \dot{Q}_{n-1} \\ &\approx \frac{2C}{h} V_n^{(k)} - \frac{2C}{h} V_n^{(k-1)} \\ &\quad + \frac{2}{h_n}\left(Q_n^{(k-1)} - Q_{n-1}\right) - \dot{Q}_{n-1}. \end{aligned} \tag{18}$$

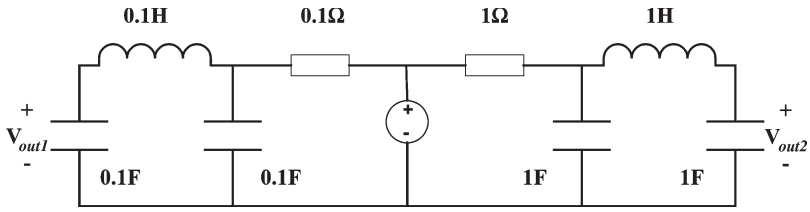


Fig. 10. Linear *RCL* circuit example.

In the above derivation, a linearized capacitance C is introduced to represent the PWNL definition of the nonlinear capacitor. The basis step-size h is used as previously. For clarity, the nonlinear charge is assumed to be the function of a single voltage. In practical MOSFET models, nonlinear charges are generally affected by three voltages $V_{\rm gs}$, $V_{\rm ds}$, and $V_{\rm bs}$. For example, the nonlinear charge between the drain and the source of a MOSFET is $Q_{\rm ds} = Q(V_{\rm gs}, V_{\rm ds}, V_{\rm bs})$. Suppose that both linearized capacitors of $Q_{\rm ds}$ ($C_{\rm ds}$, $C_{\rm m}$, and $C_{\rm mbs}$) and linearized conductors of $I_{\rm ds}$ ($g_{\rm ds}$, $g_{\rm m}$, and $g_{\rm mbs}$) in a MOSFET need to be updated due to the switch of PWNL regions, the contribution of the MOSFET to the circuit matrix is as follows:

$$\begin{array}{c} \quad\ \ \mathrm{D} \qquad \mathrm{G} \qquad\qquad\qquad \mathrm{S} \qquad\qquad\qquad \mathrm{B} \\ \begin{array}{c}\mathrm{D}\\ \mathrm{S}\end{array}\!\!\begin{bmatrix} \Delta G_{\rm ds} & \Delta G_{\rm m} & -\Delta G_{\rm ds} - \Delta G_{\rm m} - \Delta G_{\rm mbs} & \Delta G_{\rm mbs} \\ -\Delta G_{\rm ds} & -\Delta G_{\rm m} & \Delta G_{\rm ds} + \Delta G_{\rm m} + \Delta G_{\rm mbs} & -\Delta G_{\rm mbs} \end{bmatrix} \end{array}$$

$$= \begin{bmatrix} \sqrt{|a|} \\ -\sqrt{|a|} \end{bmatrix} \begin{bmatrix} \dfrac{\Delta G_{\rm ds}}{\sqrt{|a|}} & \dfrac{\Delta G_{\rm m}}{\sqrt{|a|}} & -\dfrac{a}{\sqrt{|a|}} & \dfrac{\Delta G_{\rm mbs}}{\sqrt{|a|}} \end{bmatrix}$$

$$\Delta G_{\rm ds} = \frac{2\Delta C_{\rm ds}}{h} + \Delta g_{\rm ds}, \quad \Delta G_{\rm m} = \frac{2\Delta C_{\rm gs}}{h} + \Delta g_{\rm m},$$

$$\Delta G_{\rm mbs} = \frac{2\Delta C_{\rm bs}}{h} + \Delta g_{\rm mbs}, \quad a = \Delta G_{\rm ds} + \Delta G_{\rm m} + \Delta G_{\rm mbs}. \tag{19}$$

There are a total of eight matrix entries to be updated. With the above representation, the rank-one update algorithm [7], [31] can be used to realize fast LU factorization. In case that more matrix entries (at most 16 for a MOSFET) are affected by the switch of PWNL regions, a series of rank-one or rank-m updates [3] are required to perform fast LU factorization. In this case, the efficiency of low-rank updates may be reduced.

V. Experimental Results

Four sets of experiments are reported to demonstrate the validity and efficiency of the introduced linear-centric techniques. The first test uses a simple linear *RLC* circuit to demonstrate the proposed predictor–corrector integration scheme. The second test uses a variety of analog, digital, and RF circuits with relatively small sizes to evaluate the effectiveness of the SVC method implemented with the low-rank update technique. The last two examples are circuits coupled with substrate and power/ground networks, which are used to demonstrate the scalability of SILCA on larger circuits, where a substantial portion of the circuits are linear parasitic devices. The level 1 model of MOSFETs is implemented with the proposed PWNL idea in SILCA, and nonlinear capacitors in MOSFETs are simplified as linear ones in both SILCA and SPICE3. To make a fair evaluation of the benefits of the proposed linear-centric techniques, no table lookup models of MOSFETs are used and no *RC*(*L*) model order reduction algorithm is utilized.

TABLE III
Simulation Results of a Linear Circuit Example

	# Total points	# Accepted points	# Iteration	# LU
SPICE3	2630	1965	5258	5258
SILCA	2645	1970	12831	55

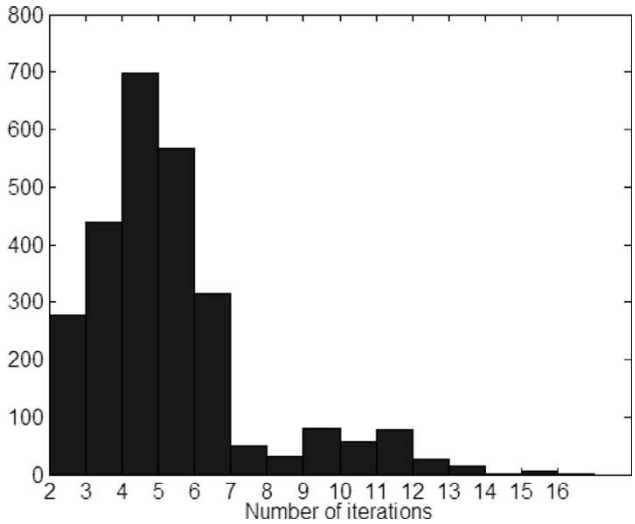


Fig. 11. Histogram of the number of iterations for a linear circuit example.

A. Evaluation of Predictor–Corrector Integration Scheme

The efficiency of the predictor–corrector integration scheme can be illustrated with the simple linear circuit example shown in Fig. 10. It includes two *RCL* circuits with different time constants. The input is a pulse signal (initially in the low voltage level 0v) with 50% duty ratio and 80-s period. The simulation length is set to 160 s. Since the minimum time constant is 0.01 s for the left half *RCL* circuit, at least 16 000 time points are required for a fixed time step-size simulation.

The simulation results are shown in Table III, where #Total points represents the number of total simulated time points and #Accepted points represents the number of accepted time points. The rejected time points are those violating the LTE requirement or exceeding the maximum iteration limit. Since in SILCA a similar adaptive time step control scheme as that in SPICE3 is applied based on the LTE requirement, it can be seen from Table III that SILCA and SPICE3 achieve similar #Total points and #Accepted points, which are much less than that required by a fixed time step-size method. Furthermore,

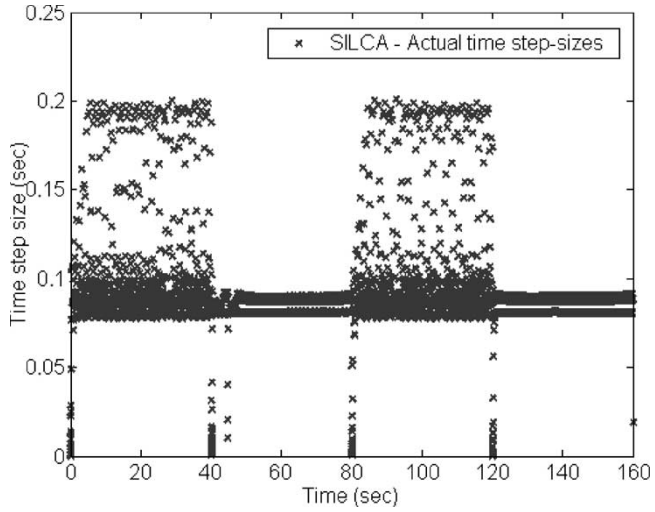


Fig. 12. Distribution of actual time step-sizes for a linear circuit example.

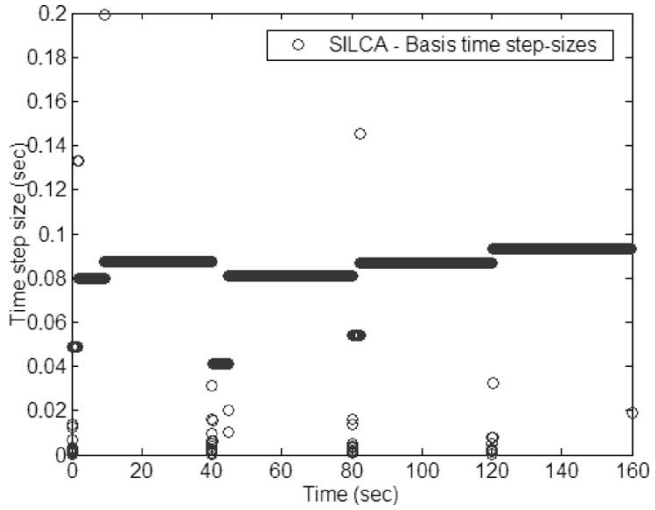


Fig. 13. Distribution of basis time step-sizes for a linear circuit example.

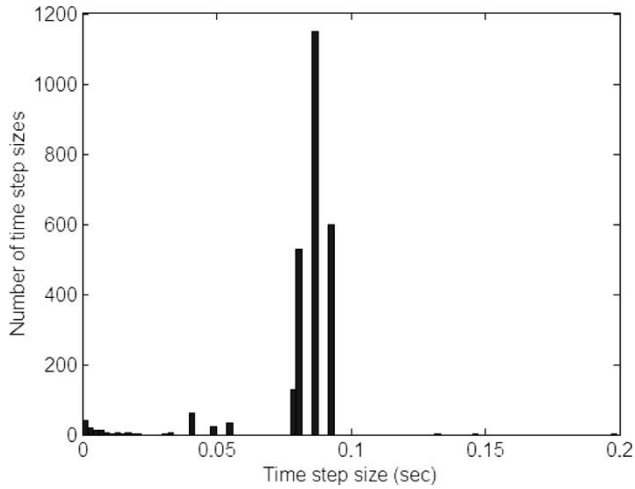


Fig. 14. Histogram of basis time step-sizes for a linear circuit example.

TABLE IV
SIMULATION RESULTS OF NONLINEAR TEST CIRCUITS*

Test Circuits	#Total points	#Accepted points	#Iter	#LU
Inv	142	127	344	344
	145	129	538	58
20-stage inverter chain	369	266	1193	1193
	355	259	2384	56
Nand2	132	123	306	306
	123	114	531	51
One-shot trigger	501	421	1525	1525
	466	401	3511	181
Comparator	145	127	444	444
	156	138	1251	58
Opamp follower	19812	13816	74216	74216
	19686	13797	98318	11
Ring oscillator	243	173	1022	1022
	251	180	2403	32
VCO	1506	1045	7621	7621
	1574	1118	18935	485

*For each circuit, the first row is the SPICE3 result and the second row is the SILCA result.

the number of LU factorizations used by SILCA decreases to 1.14% of that of SPICE3 (or 87.63× LU factorization cost saving). The number of iterations increases to about 2.5×. Fig. 11 shows the histogram of the number of iteration steps, in which it can be seen that most of the iterations converge in two to six steps.

Fig. 12 shows the distribution of actual time step-sizes $(h_n = \alpha h)$ during SILCA simulation. It can be seen that most simulated time step-sizes are between 0.05 and 0.2 s, centering around 0.08 s. Recall that since we choose $0.625 < \alpha < 2.5$, it is possible that fewer basis time step-sizes are required. This is confirmed by Fig. 13, which shows the distribution of basis time step-sizes (h) during SILCA simulation. It can be seen that most basis time step-sizes are the same and near 0.08 s. In SILCA, it is the basis time step-size that is used for circuit matrix construction. Therefore, SILCA keeps the circuit matrix constant as long as the basis time step-size is constant.

The histogram of basis time step-sizes with SILCA is shown in Fig. 14. Compared to Fig. 13, it can be concluded that most basis time step-sizes are near 0.08 s and constant during the following time intervals: 10–40, 45–80, 80–120, and 120–160 s.

It should be noted that SILCA is mainly designed for speeding up circuit simulation in case that most of the time step-sizes h_n are close to the basis time step-size h, i.e., $0.625 < (h_n/h) < 2.5$ in our experiments. In general, for transient simulation of parasitic-sensitive circuits, most of time step-sizes are close to the basis time step-size for a relatively long time interval when the transient behavior of circuits does not change significantly, i.e., staying either in the logical "0" state or logical "1" state. In case that time step-sizes h_n change violently, a new basis time step-size h will be chosen. However, based on our experiences, such chances are only few (i.e., near break points). Further, SILCA can be combined with fixed time step-size methods to enlarge the range of (h_n/h), as discussed in Section II-D.

B. Evaluation of SVC and Low-Rank Update

To illustrate the efficiency of the SVC method and low-rank update techniques, simulations on several analog, digital, and RF circuits have been performed, and the results are shown in Table IV. It can be seen that the number of iterations generally

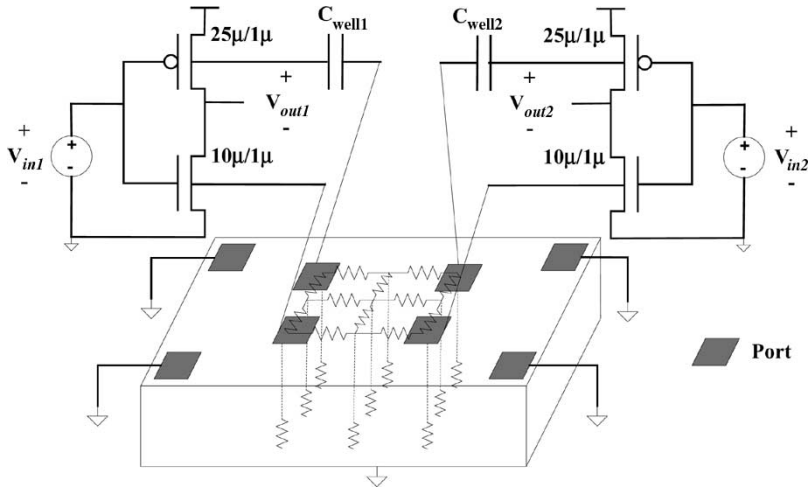


Fig. 15. Substrate coupling example.

increases to 1.5–2.5× of that with SPICE3. But the number of LU factorizations used by the SVC method with low-rank update decreases to 3%–20% of that used by SPICE3. We can see more saving in LU factorization with low-rank update for larger circuits, such as a 20-stage inverter chain, a ring oscillator, and a voltage-controlled oscillator (VCO). In general, a low-rank update technique will be more efficient for simulating a nonlinear circuit with a large-scale (potentially dense) network of linear elements since only the L and U matrices for the sparse nonlinear part need to be updated during nonlinear iteration and the dense linear part remains unchanged.

It should be pointed out that although the number of LU factorizations is reduced dramatically with the SVC method and low-rank update techniques, the speedup for circuits in Table IV is not dramatic since the simulation cost is dominated by device evaluation. As a relaxed direct method, SILCA has to take more device evaluations than SPICE3 since more iteration steps are required. For the Opamp follower example (including 32 MOSFETs, eight capacitors, and four current sources), SPICE3 runs for 17.87 s, in which 12.06 s is spent on device loading, while SILCA requires 18.65 s with 13.89 s on device loading. In this case, SILCA is more costly than SPICE3 since the simulation time is dominated by device loading. Therefore, SILCA is more suitable for parasitic-coupled VLSI circuits, where the number of linear parasitic elements dominates the number of nonlinear devices.

C. Coupled Circuit and Substrate Analysis

The third example is a simple substrate network, as shown in Fig. 15, coupled with two inverters with pulse inputs in different operating frequencies—the first inverter operates at a low frequency and the second inverter operates at a high frequency. The bulk contacts of nMOSFETs are directly connected to P-substrate ports and those of pMOSFETs are connected to P-substrate ports through a capacitor between the N-well and the P-substrate [27]. There are four other P-substrate ports connecting to the ground, and the backplane of the substrate is also connected to the ground. *RCL* loads are added at the output of each inverter (not shown in Fig. 15). The substrate is modeled as a network consisting of a three-dimensional (3-D) dense resistor mesh with multiple layers [32]. In Fig. 15, a one-layer resistor network is illustrated to model the substrate part among four inverter bulk contacts.

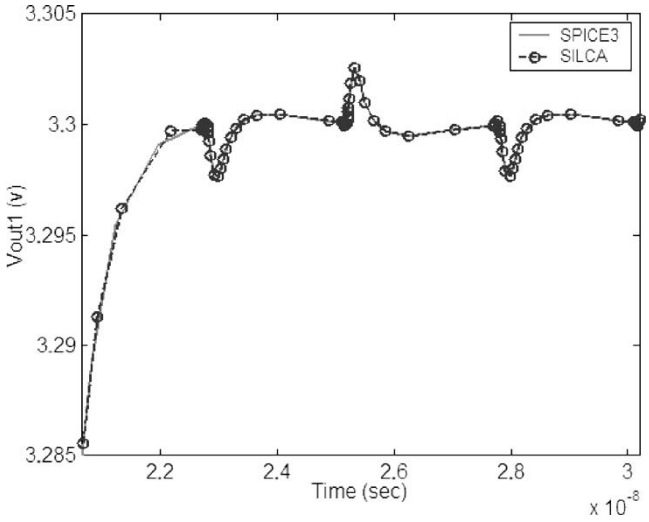


Fig. 16. Transient waveform of V_{out1} for the substrate coupling example.

Although simplified truncated substrate models have been proposed to capture dominant coupling conductance [20], [27], they are likely to underestimate coupling effects in circuit systems designed to be noise immune [24]. Furthermore, the accuracy with simplified substrate models may not be sufficient. Therefore, accurate analysis of a circuit with a fully modeled substrate is desirable for high fidelity circuit design and verification. Fig. 16 shows the time-domain output waveform of the first inverter when the output signal is a digital "1" (the high voltage level). First, the result from SILCA matches that from SPICE3. Second, it can be seen that high-frequency feed-through signals from the second inverter are present in Fig. 16. This is an important first-pass design failure reason in deep submicron digital and analog circuit designs, which may often not be captured by simplified substrate analysis.

TABLE V
SIMULATION RESULTS OF SUBSTRATE COUPLING EXAMPLES

#Layer x#Res_Per_Layer	SPICE3					*SILCA*					Speedup	
	#Iter (LU)	LU (sec)	FBS (sec)	Load (sec)	LU/ FBS	#Iter	#LU	LU (sec)	FBS (sec)	Load (sec)	LU	Overall
1x1281	12453	22.31	2.80	6.06	7.97	33157	375	0.62	7.85	0.64	35.98	3.42
2x1281	8522	131.94	7.58	12.16	17.41	21332	217	3.41	18.70	0.66	38.69	6.66
3x1281	9680	504.31	17.64	28.09	28.59	26623	273	14.28	49.64	1.52	35.32	8.41
4x1281	10280	1.374e3	32.07	50.96	42.84	28370	285	38.07	86.32	1.97	36.09	11.53
5x1281	8764	2.326e3	46.79	68.21	49.71	23666	251	66.24	118.08	3.10	35.11	13.02
6x4961	7020	1.231e5	1.492e3	1.490e3	82.50	17835	271	4.724e3	3.838e3	63.69	26.06	14.62
7x4961	9256	2.721e5	2.860e3	2.685e3	95.14	23147	312	8.933e3	7.014e3	99.96	30.46	17.30

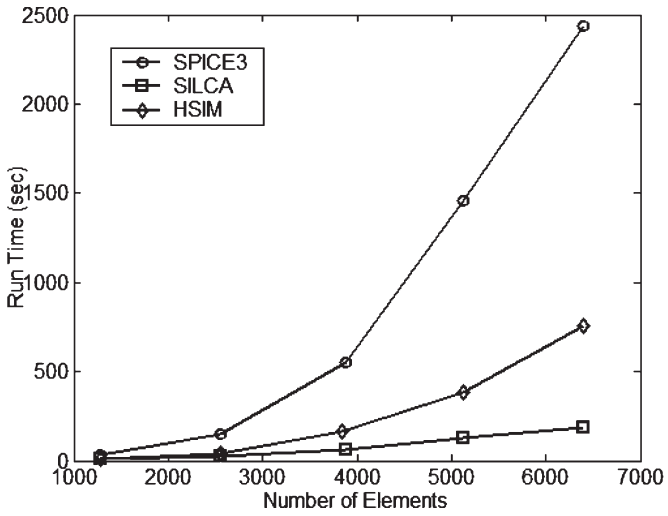


Fig. 17. Runtime comparison of the substrate coupling example.

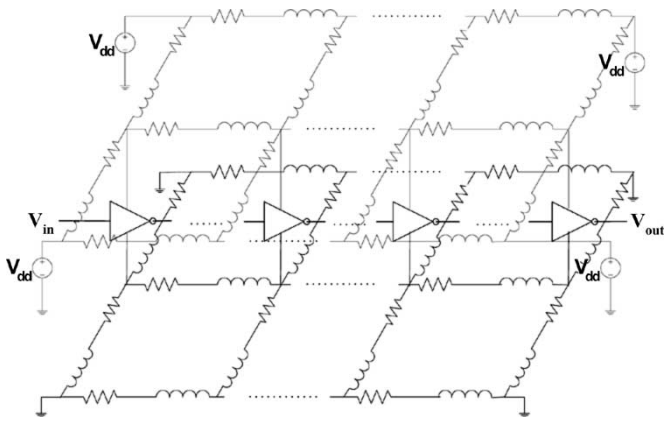


Fig. 18. Power/ground network coupling example.

Table V shows the statistics of running SILCA on a number of substrate coupling examples with varying circuit substrate network complexity compared to SPICE3. In our experiments, the number of layers and the number of resistors per layer are changed to vary the total number of circuit elements. A maximum 38.69× LU factorization cost saving and 17.30× overall speedup (with about 35 000 elements) are achieved for this simple substrate coupling analysis example, and the cost of forward/backward substitution is increased to 2.5–2.75×. No low-rank update technique is used for this example. The run time comparison is shown in Fig. 17.

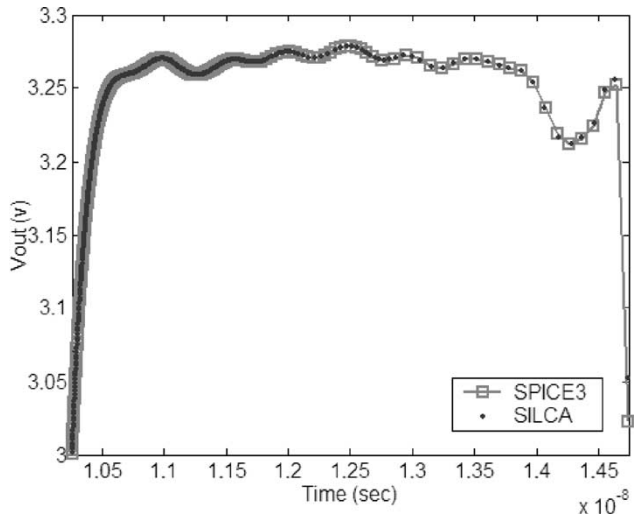


Fig. 19. Transient waveform of $V_{\rm out}$ for the power/ground network example.

Several observations are as follows. 1) The larger the circuit is (therefore the larger LU/FBS cost ratio), the more overall speedup can be achieved with SILCA. SILCA is very suitable for deep submicron VLSI circuits with strong parasitic coupling effects. 2) Device load cost with SILCA is decreased, which is proportional to the LU factorization cost saving. The reason is that in SILCA, device loads are only performed when circuit matrix elements need to be updated due to nonlinear devices and/or capacitors/inductors. For the substrate coupling examples, since most devices are resistors, their device loads are only performed when a new LU is required. 3) The more savings on LU factorization, the more iterations are required, which means more cost on forward/backward substitution and device evaluation. Therefore, there exists a tradeoff between the cost of LU factorization and that of forward/backward substitution and device evaluation. The maximum overall speedup will approach the LU factorization speedup for large strongly coupled systems.

We also compare SILCA with a fast SPICE-like circuit simulator HSIM 1.3 [36], and the results are also collected in Fig. 17. HSIM 1.3 uses the BE integration formula, a table lookup MOS level 2 model, and device bypass techniques. Further, HSIMSPEED $= 1$ is set in HSIM 1.3 so that the number of total simulated time points is close to that of SPICE3 and SILCA to achieve the same accuracy. It can be seen from Fig. 17 that the larger the circuit is, the more speedup can be achieved with SILCA. Note that SILCA does not use table lookup MOSFET models.

TABLE VI
SIMULATION RESULTS OF POWER/GROUND NETWORK COUPLING EXAMPLES

# Elems	SPICE3			*SILCA*							Speedup			
					w/o low-rank			with low-rank			w/o low-rank		with low-rank	
	#Iter (LU)	LU (sec)	Overall (sec)	#Iter	#LU	LU (sec)	Overall (sec)	#LU	LU (sec)	Overall (sec)	LU	Overall	LU	Overall
4002	4056	378.3	416.8	14614	215	19.73	81.83	47	4.59	66.74	19.17	5.09	82.42	6.24
34802	3964	4.261e4	4.466e4	16887	213	2.263e3	6.841e3	52	647.74	5.095e3	18.83	6.53	77.80	8.77
61602	4377	1.763e5	1.812e5	18027	218	8.894e3	2.045e4	52	1.992e3	1.294e4	19.82	8.86	88.50	14.00

TABLE VII
SIMULATION RESULTS OF POWER/GROUND NETWORK COUPLING EXAMPLES WITH THE GMRES SOLVER ($\varepsilon = 1\mathrm{e}-8$)

#Elem	#Tran Iter	#Tran LU	#GMRES	#GMRES Iter	#Precond	Tran LU (sec)	GMRES (sec)	Tot Tran (sec)	Speedup
4002	6086	49	5945	19872	24766	4.42	114.12	132.48	3.15
34802	7083	51	6922	22140	27696	661.66	9572.16	10648.52	4.19
61602	7207	63	7000	22476	28019	2848.69	20549.69	24275.38	7.46

D. Coupled Circuit and Power/Ground Network Analysis

The fourth example is a power/ground network as shown in Fig. 18. The power and ground supply networks are modeled as two *RCL* meshes (parasitic coupling capacitors are not shown in Fig. 18). Between these two layers is a 20-stage inverter chain, different inverters of which are connected to different power/ground nodes. *RCL* loads are added to each inverter to model interconnect lines between stages.

Fig. 19 shows the time-domain output waveform of the inverter chain when the output signal is digital "1" (the high voltage level). The "1" signal has been disturbed due to the IR drop and L^*dI/dt effects of the power/ground network (V_{dd} is 3.3 V). Table VI shows the simulation results with varied numbers of elements modeling the power/ground network. In our experiments, the sizes of two *RCL* meshes are changed to vary the number of elements. We can see that SILCA achieves more speedup for larger circuits. The number of iterations increases to 3.5–4.2× with SILCA. It is worthy noticing that the maximum LU factorization cost saving and overall speedup reach 88.50× and 14.00× (with about 60 000 elements), respectively, with the rank-one update technique, which are 19.82× and 8.86×, respectively, with only the SVC method.

For comparison purposes, we have implemented a coupled iterative/direct solver for nonlinear circuits with large-scale power/ground networks [17]. In this coupled solver, power/ground networks are formulated with a nodal analysis (NA) circuit matrix [19], which is symmetric positive definitive, and solved by the conjugate gradient method with an incomplete Cholesky decomposition preconditioner [4]. Nonlinear circuits are formulated with an MNA circuit matrix and solved by the direct method based on LU factorization and Newton–Raphson iteration as in SPICE. The iterative method and direct method are coupled together by a Gauss–Seidel relaxation scheme [34]. Experimental results on the above power/ground coupling examples show that the coupled iterative/director solver achieved similar speedup over SPICE3 as SILCA. However, it should be noticed that the coupled iterative/direct solver is efficient only if there exists a good partition with only a few boundary nodes between linear circuit parts and nonlinear circuit parts and the coupling effects between those two parts are weak.

Very recently, we developed a new GMRES solver with an LU factorization preconditioning scheme [18] for time-domain simulation of nonlinear circuits with large-scale power/ground networks. The basic idea is to apply the same time step-size controlling scheme as that used in SILCA. Whenever time step-sizes change violently, a new basis time step-size is chosen and a regular LU factorization is performed. If time step-sizes change in the range of $0.625 < \alpha < 2.5$, rather than using linear-centric analysis methods in SILCA, a GMRES solver is applied with the previous factorized L and U matrices as the preconditioner. Meanwhile, to make a fair comparison with SILCA, low-rank update has been applied to the preconditioning L and U matrices whenever a nonlinear device switches its operating region. The GMRES solver is implemented following the left-preconditioned GMRES algorithm in [26].

The simulation results with the new GMRES solver ($\varepsilon = 1\mathrm{e}-8$) are shown in Table VII. It is seen that the average number of GMRES iterations $((\#\text{GMRES Iter})/(\#\text{GMRES}))$ with the LU factorization preconditioner is about 3–3.5 for a GMRES solving process, which shows that the preconditioner is very efficient. It is shown in Table VII that the speedup over SPICE3 with the GMRES solver is less than that with SILCA. The main reason is that the number of forward/backward substitutions with the GMRES solver (#Precond in Table VII) is generally larger than that with SILCA (#Iter in Table VI). Furthermore, extra costs due to matrix–vector product operations have to be taken during the GMRES solving process. It can be expected that the simulation cost will be increased if the error tolerance of the GMRES solver is made tighter. It should be noticed that the number of nonlinear iterations (#Tran Iter) is less than that with SILCA since there is no FLC integration scheme required for capacitors/inductors. However, the number of nonlinear iterations is larger than that with SPICE3 due to the PWNL definition of MOSFETs.

VI. CONCLUSION

In this paper, a new nonlinear time-domain circuit simulation method called SILCA has been proposed for deep submicron VLSI circuit design and verification, which requires accurate modeling of parasitic coupling effects. New variable time step-size FLC numerical integration formulae are developed to

ensure constant equivalent conductance for capacitor/inductor companion models. We have characterized convergence and stability properties of the newly introduced integration formulae. As an alternative to the Newton–Raphson method, an SVC method is proposed for nonlinear circuit simulation and the low-rank update technique has been implemented for efficient LU factorization. With these techniques, SILCA can reduce dramatically the number of costly LU factorizations for time-domain simulation. Experimental results on coupled circuit, substrate, and power/ground network analysis have demonstrated that SILCA can achieve SPICE-like accuracy yet with orders of magnitude speedup over SPICE. Future research includes handling of nonlinear capacitors, optimum PWNL model generation for nonlinear device models, exploiting incomplete LU preconditioners [26] for GMRES, and applications of SILCA to coupled electrical, electromagnetic, and thermal simulation.

APPENDIX
PROOF OF THEOREM 4 AND THEOREM 6

Proof: Applying the iterative trapezoid formula (8) to an *RC* test example, the iterative relationship can be derived ($\tau = RC$), i.e.,

$$\begin{aligned}
&\tau\dot{x}_n^{(k)} + x_n^{(k)} = 0\\
&\frac{2\tau}{h}x_n^{(k)} - \frac{2\tau}{h}x_n^{(k-1)} + \frac{2\tau}{\alpha h}\left(x_n^{(k-1)} - x_{n-1}\right)\\
&\quad - \tau\dot{x}_{n-1} + x_n^{(k)} = 0\\
&\frac{2\tau}{h}x_n^{(k)} - \frac{2\tau}{h}x_n^{(k-1)} + \frac{2\tau}{\alpha h}\left(x_n^{(k-1)} - x_{n-1}\right)\\
&\quad + x_{n-1} + x_n^{(k)} = 0\\
&x_n^{(k)} = \frac{1-\frac{1}{\alpha}}{1-z}x_n^{(k-1)} + \frac{\frac{1}{\alpha}+z}{1-z}x_{n-1}, \quad z = -\frac{h}{2\tau}. \qquad \text{(A.1)}
\end{aligned}$$

Since the proof is for the stability property of the iterative trapezoid formula, the initial guess $x_n^{(0)}$ of x_n is the solution of the previous time point $x_n^{(0)} = x_{n-1}$. Then, the derivation can be carried out as

$$\begin{aligned}
&k = 1\\
&\quad x_n^{(1)} = \frac{1-\frac{1}{\alpha}}{1-z}x_{n-1} + \frac{\frac{1}{\alpha}+z}{1-z}x_{n-1} = \frac{1+z}{1-z}x_{n-1}\\
&k = 2\\
&\quad x_n^{(2)} = \frac{1-\frac{1}{\alpha}}{1-z}x_n^{(1)} + \frac{\frac{1}{\alpha}+z}{1-z}x_{n-1}\\
&\qquad = \left(\frac{1-\frac{1}{\alpha}}{1-z} * \frac{1+z}{1-z} + \frac{\frac{1}{\alpha}+z}{1-z}\right)x_{n-1}\\
&\vdots\\
&k = m\\
&\quad x_n^{(m)} = \frac{1-\frac{1}{\alpha}}{1-z}x_n^{(m-1)} + \frac{\frac{1}{\alpha}+z}{1-z}x_{n-1}\\
&\qquad = \left[\left(\frac{1-\frac{1}{\alpha}}{1-z}\right)^{m-1}\frac{1+z}{1-z}\right.\\
&\qquad\quad \left. + \left(\frac{1-\frac{1}{\alpha}}{1-z}\right)^{m-2}\frac{\frac{1}{\alpha}+z}{1-z} + \cdots + \frac{\frac{1}{\alpha}+z}{1-z}\right]x_{n-1}. \qquad \text{(A.2)}
\end{aligned}$$

Noting that the terms in the square bracket of (A.1) are a geometric series except the first term, it can be written further in the following format if $(1-1/\alpha)/(1-z) \neq 1$, i.e.,

$$\begin{aligned}
&k = m\\
&\quad x_n^{(m)} = \left[\left(\frac{1-\frac{1}{\alpha}}{1-z}\right)^{m-1}\frac{1+z}{1-z} + \frac{1-\left(\frac{1-\frac{1}{\alpha}}{1-z}\right)^{m-1}}{1-\frac{1-\frac{1}{\alpha}}{1-z}}\frac{\frac{1}{\alpha}+z}{1-z}\right]x_{n-1}. \qquad \text{(A.3)}
\end{aligned}$$

According to (A.2), it is easy to check that the absolute stability condition cannot be satisfied if $(1-1/\alpha)/(1-z) = 1$. Therefore, the absolute stability region of the iterative trapezoid formula is then expressed by the inequality

$$\left|\left(\frac{1-\frac{1}{\alpha}}{1-z}\right)^{m-1}\frac{1+z}{1-z} + \frac{1-\left(\frac{1-\frac{1}{\alpha}}{1-z}\right)^{m-1}}{1-\frac{1-\frac{1}{\alpha}}{1-z}}\frac{\frac{1}{\alpha}+z}{1-z}\right| < 1. \qquad \text{(A.4)}$$

Finally, we have the result

$$\left|\left(\frac{1-\frac{1}{\alpha}}{1-z}\right)^{m}\left(\frac{2z}{z-\frac{1}{\alpha}}\right) + \frac{\frac{1}{\alpha}+z}{\frac{1}{\alpha}-z}\right| < 1. \qquad \text{(A.5)}$$

This completes the proof of Theorem 4. ■

If the mixed trapezoid FE formula is applied as an integration predictor when $\alpha < 1$, the absolute stability region of the iterative trapezoid formula is derived as

$$\begin{aligned}
&x_n^{(0)} = \frac{1+(2\alpha-1)z}{1-z}x_{n-1}\\
&k = 1\\
&\quad x_n^{(1)} = \frac{1-\frac{1}{\alpha}}{1-z}x_n^{(0)} + \frac{\frac{1}{\alpha}+z}{1-z}x_{n-1}\\
&\vdots\\
&k = m\\
&\quad x_n^{(m)} = \frac{1-\frac{1}{\alpha}}{1-z}x_n^{(m-1)} + \frac{\frac{1}{\alpha}+z}{1-z}x_{n-1}\\
&\qquad = \left[\left(\frac{1-\frac{1}{\alpha}}{1-z}\right)^{m-1}\frac{1+(2\alpha-1)z}{1-z}\right.\\
&\qquad\quad \left. + \left(\frac{1-\frac{1}{\alpha}}{1-z}\right)^{m-2}\frac{\frac{1}{\alpha}+z}{1-z} + \cdots + \frac{\frac{1}{\alpha}+z}{1-z}\right]x_{n-1}. \qquad \text{(A.6)}
\end{aligned}$$

Therefore, the absolute stability region of the iterative trapezoid formula is then expressed by the inequality

$$\left|\left(\frac{1-\frac{1}{\alpha}}{1-z}\right)^{m+1}\left(\frac{2\alpha z^2}{z-\frac{1}{\alpha}}\right)+\frac{\frac{1}{\alpha}+z}{\frac{1}{\alpha}-z}\right|<1. \tag{A.7}$$

This completes the proof of Theorem 6. ■

ACKNOWLEDGMENT

The authors would like to thank Prof. K. Mayaram of Oregon State University and Dr. J. Rockway of the Naval Space and Warfare System Center, San Diego, for several helpful discussions. The authors are also grateful to the anonymous reviewers for their detailed and constructive comments that greatly enhanced this paper.

REFERENCES

[1] E. Acar, F. Dartu, and L. T. Pileggi, "TETA: Transistor-level waveform evaluation for timing analysis," *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, vol. 21, no. 5, pp. 605–616, May 2002.

[2] U. M. Ascher and L. R. Petzold, *Computer Methods for Ordinary Differential Equations and Differential-Algebraic Equations*. Philadelphia, PA: SIAM, 1998.

[3] H. W. Buurman, "From circuit to signal-development of a piecewise linear simulator," Ph.D. dissertation, Dept. Elect. Eng., Eindhoven Univ. Technol., Eindhoven, The Netherlands, Jan. 1993.

[4] T. Chen and C. C.-P. Chen, "Efficient large-scale power grid analysis based on preconditioned Krylov-subspace iterative methods," in *Proc. IEEE/ACM Design Automation Conf.*, Las Vegas, NV, Jun. 2001, pp. 559–562.

[5] P. F. Cox, R. G. Burch, P. Yang, and D. E. Hocevar, "New implicit integration method for efficient latency exploration in circuit simulation," *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, vol. 8, no. 10, pp. 1051–1064, Oct. 1989.

[6] J. E. Dennis and J. J. Moré, "Quasi-Newton methods, motivation and theory," *SIAM Rev.*, vol. 19, no. 1, pp. 46–89, Jan. 1977.

[7] T. Fujisawa, E. S. Kuh, and T. Ohtsuki, "A sparse matrix method for analysis of piecewise-linear resistive networks," *IEEE Trans. Circuit Theory*, vol. CT-19, no. 6, pp. 571–584, Nov. 1972.

[8] K. Gala, V. Zolotov, R. Panda, B. Young, J. Wang, and D. Blaauw, "On-chip inductance modeling and analysis," in *Proc. IEEE/ACM Design Automation Conf.*, Los Angeles, CA, Jun. 2000, pp. 63–68.

[9] C. W. Gear, *Numerical Initial Value Problems in Ordinary Differential Equations*. Upper Saddle River, NJ: Prentice-Hall, 1971.

[10] K. R. Jackson and R. Sacks-Davis, "An alternative implementation of variable step-size multistep formulas for stiff ODEs," *ACM Trans. Math. Softw.*, vol. 6, no. 3, pp. 295–318, Sep. 1980.

[11] S. Kapur and D. E. Long, "Large-scale capacitance calculation," in *Proc. IEEE/ACM Design Automation Conf.*, Los Angeles, CA, Jun. 2000, pp. 744–749.

[12] K. S. Kundert and A. Sangiovanni-Vincentelli, *Sparse User's Guide—A Sparse Linear Equation Solver Version 1.3a*. Berkeley: Univ. California, Apr. 1988.

[13] P. M. Lee, S. Ito, T. Hashimoto, J. Sato, T. Touma, and G. Yokomizo, "A parallel and accelerated circuit simulator with precise accuracy," in *Proc. Int. Conf. VLSI Design*, Bangalore, India, Jan. 2002, pp. 213–218.

[14] E. Lelarasmee and A. Sangiovanni-Vincentelli, "RELAX: A new circuit simulator for large scale MOS integrated circuits," in *Proc. IEEE/ACM Design Automation Conf.*, Las Vegas, NV, Jun. 1982, pp. 682–690.

[15] P. Li and L. Pileggi, "A linear-centric modeling approach to harmonic balance analysis," in *Proc. IEEE/ACM Design, Automation and Test Eur. Conf.*, Paris, France, Mar. 2002, pp. 634–639.

[16] Z. Li and C.-J. R. Shi, "SILCA: Fast-yet-accurate time-domain simulation of VLSI circuits with strong parasitic coupling effects," in *Proc. IEEE/ACM Int. Conf. Computer-Aided Design*, San Jose, CA, Nov. 2003, pp. 793–799.

[17] ——, "A coupled iterative/direct method for efficient time-domain simulation of nonlinear circuits with power/ground networks," in *Proc. IEEE Int. Symp. Circuits and Systems*, Vancouver, Canada, May 2004, vol. 5, pp. 165–168.

[18] ——, "An efficiently preconditioned GMRES method for fast parasitic-sensitive deep-submicron VLSI circuit simulation," in *Proc. IEEE/ACM Design, Automation and Test Eur. Conf.*, Munich, Germany, Mar. 2005, vol. 2, pp. 752–757.

[19] W. J. McCalla, *Fundamentals of Computer-Aided Circuit Simulation*. Boston, MA: Kluwer, 1988.

[20] M. Nagata, J. Nagai, K. Hijikata, T. Morie, and A. Iwata, "Physical design guides for substrate noise reduction in CMOS digital circuits," *IEEE J. Solid-State Circuits*, vol. 36, no. 3, pp. 539–549, Mar. 2001.

[21] L. W. Nagel, "SPICE: A computer program to simulate semiconductor circuits," Univ. California, Berkeley, Tech. Rep. UCB/ERL M520, May 1975.

[22] A. Odabasioglu, M. Celik, and L. T. Pillegi, "PRIMA: Passive reduced-order interconnect macromodeling algorithm," *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, vol. 17, no. 8, pp. 645–654, Aug. 1998.

[23] L. R. Petzold, "A description of DASSL: A differential/algebraic system solver," in *IMACS Transactions on Scientific Computing*, vol. 1. Amsterdam, The Netherlands: North-Holland, R. Stepleman *et al.*, Eds.1983, pp. 65–68.

[24] J. R. Phillips and L. M. Silveira, "Simulation approaches for strongly coupled interconnect systems," in *Proc. IEEE/ACM Int. Conf. Computer-Aided Design*, San Jose, CA, Nov. 2001, pp. 430–437.

[25] K. Radhakrishnan and A. C. Hindmarsh, "Description and use of LSODE, the Livermore solver for ordinary differential equations," Lawrence Livermore Nat. Lab., Livermore, CA, LLNL Tech. Rep. UCRL-ID-113855, 1993.

[26] Y. Saad, *Iterative Methods for Sparse Linear Systems*, 2nd ed. Philadelphia, PA: SIAM, 2003.

[27] A. Samavedam, A. Sadate, K. Mayaram, and T. S. Fiez, "A scalable substrate noise coupling model for design of mixed-signal IC's," *IEEE J. Solid-State Circuits*, vol. 35, no. 6, pp. 895–904, Jun. 2000.

[28] B. N. Sheehan, "TICER: Realizable reduction of extracted RC circuits," in *Proc. IEEE/ACM Int. Conf. Computer-Aided Design*, San Jose, CA, Nov. 1999, pp. 200–203.

[29] A. H. Sherman, "On Newton-iterative methods for the solution of systems of nonlinear equations," *SIAM J. Numer. Anal.*, vol. 15, no. 4, pp. 755–771, 1978.

[30] H. Su, K. H. Gala, and S. S. Sapatnekar, "Fast analysis and optimization of power/ground networks," in *Proc. IEEE/ACM Int. Conf. Computer-Aided Design*, San Jose, CA, Nov. 2000, pp. 477–480.

[31] J. T. J. van Eijndhoven and M. T. van Stiphout, "Latency exploitation in circuit simulation by sparse matrix techniques," in *Proc. IEEE Int. Symp. Circuits and Systems*, Espoo, Finland, Jun. 1988, pp. 623–626.

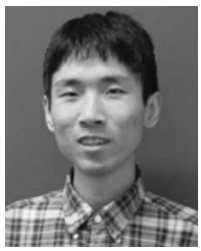
[32] N. K. Verghese, T. J. Schmerbeck, and D. J. Allstot, *Simulation Techniques and Solutions for Mixed-Signal Coupling in Integrated Circuits*. Norwell, MA: Kluwer, 1995.

[33] Y. Wang, V. Jandhyala, and C.-J. R. Shi, "Coupled electromagnetic-circuit simulation of arbitrarily-shaped conducting structures," in *Proc. IEEE Conf. Electrical Performance Electronic Packaging*, Cambridge, MA, Oct. 2001, pp. 233–236.

[34] J. K. White and A. Sangiovanni-Vincentelli, *Relaxation Techniques for the Simulation of VLSI Circuits*. Norwell, MA: Kluwer, 1987.

[35] M. Zwoliński and R. W. Allen, "Practical algorithms for fully decoupled mixed-mode simulation of electronic circuits," in *Proc. IEEE Int. Symp. Circuits and Systems*, Sydney, Australia, May 2001, pp. 451–454.

[36] *User Guide—HSIM Version 1.3*, Nassda Corp., Santa Clara, CA, Apr. 2001.



Zhao Li received the B.S. degree in electronics and the M.S. degree in microelectronics and solid-state electronics from Tsinghua University, Beijing, China, in 1998 and 2000, respectively, and the Ph.D. degree in electrical engineering from the University of Washington, Seattle, in 2005.

He is currently with Cadence Design Systems, Inc., San Jose, CA. His research interests include mixed-signal and deep submicron circuit simulation, symbolic analysis, behavioral modeling for analog/RF circuit application, device modeling, and optimization algorithms.

C.-J. Richard Shi (M'91–SM'99–F'06) received the Ph.D. degree in computer science from the University of Waterloo, Waterloo, ON, Canada, in 1994.

From 1994 to 1998, he was with Analogy, Rockwell Semiconductor Systems, and the University of Iowa. In 1998, he joined the University of Washington, Seattle, WA, where he is currently a Professor in electrical engineering. His research interests include several aspects of the computer-aided design and test of integrated circuits and systems, with particular emphasis on analog/mixed-signal and deep submicron circuit modeling, simulation, and design automation. He is a key contributor to the IEEE Std. 1076.1-1999 (VHDL-AMS) standard for the description and simulation of mixed-signal circuits and systems. He founded the IEEE International Workshop on Behavioral Modeling and Simulation (BMAS) in 1997.

Dr. Shi was an Associate Editor, as well as a Guest Editor, of the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS—II, ANALOG AND DIGITAL SIGNAL PROCESSING. Since 1999, he has been the Associate Editor of the IEEE TRANSACTIONS ON COMPUTER-AIDED DESIGN OF INTEGRATED CIRCUITS AND SYSTEMS. He has received several awards for his research including a Doctoral Prize from the Natural Science and Engineering Research Council of Canada (1995), a Best Paper Award from the 1998 IEEE VLSI Test Symposium, a Best Paper Award from the 1999 IEEE/ACM Design Automation Conference, a National Science Foundation CAREER Award (2000), and an SRC-TECHCON Best Paper Award (2003).