Efficient DC Fault Simulation of Nonlinear Analog Circuits: One-Step Relaxation and Adaptive Simulation Continuation

C.-J. Richard Shi, Michael W. Tian, and Guoyong Shi

***Abstract*—Efficient dc fault simulation of nonlinear analog circuits is addressed in this paper. Two techniques, one-step relaxation and adaptive simulation continuation, are proposed. By one-step relaxation, only one Newton–Raphson iteration is performed for each faulty circuit with the dc solution of the good circuit as the initial point, and the approximate solution is used for detecting the fault. The paper shows experimentally and justifies theoretically that approximate dc fault simulation by one-step relaxation can accomplish almost the same fault coverage as exact dc fault simulation. Exact dc fault simulation by adaptive simulation continuation is first to order faulty circuits based on the results of one-step relaxation, and then to use the solution of the previous faulty circuit as the initial point for the Newton–Raphson iteration of the next faulty circuit. Experiments on a set of 29 MCNC Circuit Simulation and Modeling Workshop benchmark circuits show that exact dc fault simulation by adaptive simulation continuation can achieve an average speedup of 4.4 and as high as 15 over traditional stand-alone fault simulation.**

***Index Terms*—Analog testing, dc fault simulation, fault coverage, fault ordering, first-order approximation, simulation continuation, Newton–Raphson iteration.**

I. Introduction

Testing the analog portion of mixed-signal circuits and systems has been recognized as an important issue that affects both time-to-market and final product cost. Among several methods of analog testing, dc testing is generally cheaper since it does not require expensive test equipment and has a shorter testing time. Traditionally, dc testing is performed before ac and transient testing, by which majority of faults have been detected; hence, the overall testing time and cost are reduced. Much progress has been made in low-cost dc test generation [2], [8] and dc built-in self-test [1].

DC fault simulation is to simulate the dc behavior of an analog circuit for a given list of faults. It is an important means for fault-coverage analysis, test generation, and built-in self-test design [16]. While efficient fault simulation for linear analog circuits has been extensively studied [10], [13], [18], [19], less progress has been made in dc fault simulation of nonlinear analog circuits—a more difficult and practically important problem. In general, dc simulation of nonlinear analog circuits amounts to solving a set of nonlinear algebraic equations, which is usually solved by the Newton–Raphson iterative algorithm. This procedure is computationally expensive for large and highly nonlinear circuits. Moreover, faults often deteriorate the convergence speed. We note the progress in reducing the number of faults to be simulated by inductive fault analysis (IFA) [15], in minimizing the simulation complexity by behavioral modeling [11], and in shortening the equation setup time and exploiting concurrency [22].

Manuscript received December 24, 2004; revised April 26, 2005. This work was supported in part by the U.S. Defense Advanced Research Projects Agency (DARPA) EPI Program under Grant F33615-96-1-5601 from the Air Force Research Laboratory, Manufacturing Technology Directorate, and in part by the DARPA NeoCAD Program under Grant N66001-01-8920 from the Navy Space and Naval Warfare Systems Command (SPAWAR). This paper was recommended by Associate Editor K. Chakrabarty.

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

M. W. Tian is with Cadence Design Systems, San Jose, CA 95134 USA.

G. Shi was with the Department of Electrical Engineering, University of Washington, Seattle, WA 98195 USA. He is now with the School of Microelectronics, Shanghai Jiao Tong University, Shanghai 200030, China (e-mail: shiguoyong@ic.sjtu.edu.cn).

Digital Object Identifier 10.1109/TCAD.2005.855884

Our work was inspired by the very success of simple stuck-at fault models used in fault-driven digital testing. Stuck-at faults are the first-order abstraction of failure mechanisms in digital integrated circuits. They are simple enough to be amenable to efficient fault simulation and test generation. Nevertheless, the most fundamental reason why stuck-at fault models are so successful and have enjoyed industry-wide use is that test vectors computed using stuck-at fault models tend to work well in practice even when the actual failure mechanisms may not be precisely abstracted by stuck-at models. Therefore, what is really important for fault simulation and test generation is not necessarily the absolutely accurate modeling of failure mechanisms, rather the relatively accurate modeling in such a way that it can predict failure behavior effectively.

This paper describes two techniques for the efficient dc fault simulation of nonlinear analog circuits. The first technique is based on the observation that for most applications the absolute accuracy of fault simulation is difficult to achieve, but actually not needed. We thus propose to perform only one Newton–Raphson iteration for the faulty circuit and to use the approximate solution for detecting faults. The second technique aims at reducing the total number of Newton–Raphson iterations required for exact fault simulation using fault ordering. The idea is to sort the faults by the closeness of the approximate solutions obtained from the one-step relaxation technique so that the previous faulty circuit response is used as the good initial point for the next faulty circuit. Previously, ordering faults has been used in analog IC testing in several different contexts, such as [6] and [8], where easily detectable faults are ordered with a higher precedence in order to minimize the test time.

Some preliminary results of this paper were presented in [20] and [21], and have been extended later by others for transient fault simulation [3], [4]. Engin and Kerkhoff explored dc bias based fault grouping for fast transient fault simulation [3]. Hou and Chatterjee developed efficient techniques for concurrent transient fault simulation, where first-order projection is used to order the faulty circuits for efficient nonlinear iteration at the next time step. In this paper, we provide a rigorous and unified treatment of our earlier work. Over the years of research and development, the subject of analog and mixed-signal testing and verification still remains a challenge.

This paper is organized as follows. Section II formulates the dc fault simulation problem and introduces some metrics and terminology. Section III presents the proposed one-step relaxation method for approximate dc fault simulation, its rationale, and implementation. Section IV presents an efficient exact dc fault simulation method based on the application of one-step relaxation to adaptive fault ordering and simulation continuation. The methods have been implemented in a prototype dc fault simulator built on SPICE3f5 and validated on a set of benchmark circuits. The experimental setting for testing the two proposed methods is described in Section V. Experimental results are reported in Section VI. Section VII concludes the paper.

II. DC Fault Simulation and Detection

DC simulation of analog circuits amounts to solving a system of nonlinear algebraic equations

$$\mathbf{g}(\mathbf{x}) = \mathbf{0} \tag{1}$$

where $\mathbf{x} \in \mathbb{R}^n$ is the vector of circuit unknowns (very often currents and voltages). The standard algorithm for solving the general nonlinear equation (1) is the well-known Newton–Raphson algorithm, which is an iterative process starting from an initial approximation $\mathbf{x}^{(0)}$. The iteration consists of the following steps:

- Guess $\mathbf{x}^{(0)}$.
- Solve $\mathbf{J}_g(\mathbf{x}^{(k)})(\mathbf{x}^{(k+1)} - \mathbf{x}^{(k)}) = -\mathbf{g}(\mathbf{x}^{(k)})$ for $k = 0, 1, 2, \ldots$ until the convergence criteria are met.

Here, $\mathbf{J}_g(\mathbf{x}^{(k)})$ denotes the Jacobian matrix of $\mathbf{g}(\mathbf{x})$ at $\mathbf{x} = \mathbf{x}^{(k)}$ defined by

$$\mathbf{J}_g(\mathbf{x}^{(k)}) = \left.\frac{\partial \mathbf{g}(\mathbf{x})}{\partial \mathbf{x}}\right|_{\mathbf{x}=\mathbf{x}^{(k)}}. \quad (2)$$

Analog testing is to measure certain circuit quantities, which may be referred to as the measurement vector, denoted as $\mathbf{y} \in \mathbb{R}^p (p \le n)$. In general

$$\mathbf{y} = \mathbf{h}(\mathbf{x}) \quad (3)$$

where $\mathbf{h}$ is a vector of nonlinear measurement functions. Very often, $\mathbf{y}$ is scalar and is a linear combination of a few critical circuit unknowns easily measurable from a circuit under test.

The task of fault detection is to detect whether the designed circuit behaves as desired. In the dc framework, we denote the desired (good) circuit equation by

$$\mathbf{g}(\mathbf{x}) = 0 \quad (4)$$

and its exact solution by $\mathbf{x}_g$, i.e., $\mathbf{g}(\mathbf{x}_g) = 0$. Due to fault, the dc equation for the faulty circuit deviates from the good one, thus is represented by another set of nonlinear equations

$$\mathbf{f}(\mathbf{x}) = 0 \quad (5)$$

whose exact solution is denoted $\mathbf{x}_f$, which satisfies $\mathbf{f}(\mathbf{x}_f) = 0$.

We define the normalized absolute distance between the two nonzero dc responses $\mathbf{x}_g$ and $\mathbf{x}_f$ as

$$d(\mathbf{x}_g, \mathbf{x}_f) = \frac{\|\mathbf{x}_g - \mathbf{x}_f\|}{\|\mathbf{x}_g\|} \quad (6)$$

where the norm $\|\cdot\|$ is typically 1-norm or 2-norm and we assume that $\|\mathbf{x}_g\| \ne 0$.

In view of (3), we define the detection threshold by

$$\mathcal{T}(\mathbf{y}_g, \mathbf{y}_f) = \frac{\|\mathbf{y}_g - \mathbf{y}_f\|}{\|\mathbf{y}_g\|} \quad (7)$$

where $\mathbf{y}_g = \mathbf{h}(\mathbf{x}_g)$ and $\mathbf{y}_f = \mathbf{h}(\mathbf{x}_f)$ are the measurements of $\mathbf{x}_g$ and $\mathbf{x}_f$, respectively, and we assume that $\|\mathbf{y}_g\| \ne 0$. For example, we may classify a circuit $\mathbf{f}$ as faulty if the computed threshold exceeds 0.1%. In this paper, we also use the distance defined above for measuring the closeness of two faulty circuits (faults).

III. One-Step Relaxation

In this section, we introduce and justify an approximate dc fault simulation method for nonlinear analog circuits known as one-step relaxation. We further show that one-step relaxation can be efficiently implemented using Householder's formulas.

By one-step relaxation, we mean that only one Newton–Raphson iteration is carried out starting from an initial point and the approximate solution is used for fault detection. Since the exact dc solution of the good circuit is known for fault detection, it is natural to use the good dc solution for the initial point in one-step relaxation. Intuitively, this technique detects the fault of an actual nonlinear faulty circuit by solving the circuit linearized at the dc solution of the good circuit.

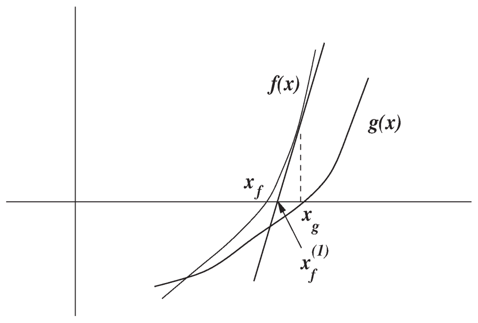

Fig. 1. Illustration of one-step relaxation.

With the notations introduced before, one-step relaxation for dc fault simulation is to solve the following equation for $\mathbf{x}_f^{(1)}$ given $\mathbf{x}_g$, i.e.,

$$\mathbf{J}_f(\mathbf{x}_g)\left(\mathbf{x}_f^{(1)} - \mathbf{x}_g\right) = -\mathbf{f}(\mathbf{x}_g) \quad (8)$$

where $\mathbf{J}_f(\mathbf{x}_g)$ is the Jacobian matrix of the vector function $\mathbf{f}(\mathbf{x})$ evaluated at $\mathbf{x}_g$. The solution $\mathbf{x}_f^{(1)}$ of (8) is used as an approximate of the exact solution of the faulty circuit and passed to the measurement (3). The threshold $\mathcal{T}(\mathbf{h}(\mathbf{x}_f^{(1)}), \mathbf{h}(\mathbf{x}_g))$ defined in (7) is then computed to determine whether the circuit is faulty. Geometrically, one-step relaxation is illustrated in Fig. 1.

The accuracy of one-step relaxation can be analyzed from the circuit perturbation perspective. We shall establish quantitatively that, under certain conditions, a fault detectable by the exact solution $\mathbf{x}_f$ can also be detected by using the first-order approximate solution $\mathbf{x}_f^{(1)}$.

Viewing the dc model $\mathbf{f}(\mathbf{x})$ as a perturbation from the function $\mathbf{g}(\mathbf{x})$, we define the perturbation to be

$$\delta\mathbf{g}(\mathbf{x}) = \mathbf{f}(\mathbf{x}) - \mathbf{g}(\mathbf{x}). \quad (9)$$

Because of circuit perturbation, the exact dc solution is also perturbed from $\mathbf{x}_g$ to $\mathbf{x}_f$. Let $\Delta\mathbf{x} = \mathbf{x}_f - \mathbf{x}_g$ be the amount of perturbation from the exact dc solution $\mathbf{x}_g$. Since both $\mathbf{f}(\mathbf{x}_f) = 0$ and $\mathbf{g}(\mathbf{x}_g) = 0$, we have

$$\begin{aligned} 0 &= \mathbf{f}(\mathbf{x}_f) - \mathbf{g}(\mathbf{x}_g) \\ &= [\mathbf{f}(\mathbf{x}_f) - \mathbf{g}(\mathbf{x}_f)] + [\mathbf{g}(\mathbf{x}_f) - \mathbf{g}(\mathbf{x}_g)] \\ &= \delta\mathbf{g}(\mathbf{x}_g + \Delta\mathbf{x}) + [\mathbf{g}(\mathbf{x}_g + \Delta\mathbf{x}) - \mathbf{g}(\mathbf{x}_g)] \\ &= \delta\mathbf{g}(\mathbf{x}_g) + \mathbf{J}_{\delta g}(\mathbf{x}_g)\Delta\mathbf{x} + \mathbf{J}_g(\mathbf{x}_g)\Delta\mathbf{x} + O(\|\Delta\mathbf{x}\|)^2 \end{aligned} \quad (10)$$

where $\mathbf{J}_{\delta g}$ is the Jacobian matrix of $\delta\mathbf{g}$ and $O(\|\Delta\mathbf{x}\|^2)$ denotes the second-order term. In the last step, we used the Taylor expansions of $\delta\mathbf{g}(\mathbf{x}_f + \Delta\mathbf{x})$ and $\mathbf{g}(\mathbf{x}_g + \Delta\mathbf{x})$ at $\Delta\mathbf{x} = 0$.

One can solve the exact amount of perturbation $\Delta\mathbf{x}$ from (10). However, if $\Delta\mathbf{x}$ is sufficiently small, i.e., the exact dc solution $\mathbf{x}_g$ is not perturbed too much, $\Delta\mathbf{x}$ can be solved approximately from the first-order approximation of (10), i.e.,

$$\delta\mathbf{g}(\mathbf{x}_g) + [\mathbf{J}_{\delta g}(\mathbf{x}_g) + \mathbf{J}_g(\mathbf{x}_g)]\,\widetilde{\Delta}\mathbf{x} = 0 \quad (11)$$

where $\widetilde{\Delta}\mathbf{x} := \widetilde{\mathbf{x}}_f - \mathbf{x}_g$ and $\widetilde{\mathbf{x}}_f$ is an approximation to $\mathbf{x}_f$. Noting that

$$\begin{aligned} \delta\mathbf{g}(\mathbf{x}_g) &= \mathbf{f}(\mathbf{x}_g) - \mathbf{g}(\mathbf{x}_g) \\ &= \mathbf{f}(\mathbf{x}_g) \\ \mathbf{J}_{\delta g}(\mathbf{x}_g) + \mathbf{J}_g(\mathbf{x}_g) &= \mathbf{J}_f(\mathbf{x}_g) - \mathbf{J}_g(\mathbf{x}_g) + \mathbf{J}_g(\mathbf{x}_g) \\ &= \mathbf{J}_f(\mathbf{x}_g) \end{aligned}$$

we obtain from (11) that if the Jacobian matrix $\mathbf{J}_f(\mathbf{x}_g)$ is nonsingular

$$\widetilde{\mathbf{x}}_f = \mathbf{x}_g - \mathbf{J}_f(\mathbf{x}_g)^{-1}\mathbf{f}(\mathbf{x}_g) \tag{12}$$

which is exactly the one-step relaxation formula, i.e., $\widetilde{\mathbf{x}}_f = \mathbf{x}_f^{(1)}$. In other words, the solution from one-step relaxation approximates the exact solution of the perturbed (faulty) circuit better if the perturbation is smaller (or the faulty is not serious).

To quantify the error bounds of one-step approximation, we shall establish a relation between the two error rates $\|\mathbf{y}_f^{(1)} - \mathbf{y}_g\|/\|\mathbf{y}_g\|$ and $\|\mathbf{y}_f - \mathbf{y}_g\|/\|\mathbf{y}_g\|$, where $\mathbf{y}_f^{(1)} = \mathbf{h}(\mathbf{x}_f^{(1)})$. To this end, we introduce two inequalities, whose proofs can be found in [12].

Lemma 1: Let $\|\mathbf{J}_g^{-1}(\mathbf{x}_g)\| \leq \alpha$ and $\|\mathbf{J}_{\delta g}(\mathbf{x}_g)\| \leq \beta$ with $\alpha\beta < 1$. Then

$$\left\|\mathbf{J}_f^{-1}(\mathbf{x}_g)\right\| \leq \frac{\alpha}{1-\alpha\beta}. \tag{13}$$

Proof: Since $\mathbf{J}_f(\mathbf{x}_g) = \mathbf{J}_g(\mathbf{x}_g) + \mathbf{J}_{\delta g}(\mathbf{x}_g)$, the inequality follows directly from the Perturbation Lemma [12, p. 45, 2.3.2]. ■

Lemma 2: If $\mathbf{f}$ is differentiable in the neighborhood of $\mathbf{x}_g$, the following inequality holds for all $\mathbf{x}_f$ near $\mathbf{x}_g$, i.e.,

$$\|\mathbf{f}(\mathbf{x}_f) - \mathbf{f}(\mathbf{x}_g) - \mathbf{J}_f(\mathbf{x}_g)(\mathbf{x}_f - \mathbf{x}_g)\| \leq M\,\|\mathbf{x}_f - \mathbf{x}_g\| \tag{14}$$

where $M := \sup_{0\leq t\leq 1} \|\mathbf{J}_f(\mathbf{x}_g + t(\mathbf{x}_f - \mathbf{x}_g)) - \mathbf{J}_f(\mathbf{x}_g)\|$.

Proof: This inequality follows immediately from [12, p. 70, 3.2.5]. ■

From the preceding two inequalities, we obtain the following result.

Theorem 1: Let α, β, and M be defined as in the preceding two lemmas. Let $\gamma = \alpha M/(1-\alpha\beta)$. If $\gamma < 1$, then we have

$$\left\|\mathbf{x}_f^{(1)} - \mathbf{x}_f\right\| \leq \gamma|\mathbf{x}_f - \mathbf{x}_g\| \tag{15}$$

$$\left\|\mathbf{x}_f^{(1)} - \mathbf{x}_g\right\| \geq (1-\gamma)\|\mathbf{x}_f - \mathbf{x}_g\|. \tag{16}$$

Proof: Noting that $\mathbf{f}(\mathbf{x}_f) = 0$ and using (8), we have

$$\begin{aligned}
&\|\mathbf{f}(\mathbf{x}_f) - \mathbf{f}(\mathbf{x}_g) - \mathbf{J}_f(\mathbf{x}_g)(\mathbf{x}_f - \mathbf{x}_g)\| \\
&\quad = \left\|-\mathbf{f}(\mathbf{x}_g) - \mathbf{J}_f(\mathbf{x}_g)\left[\left(\mathbf{x}_f - \mathbf{x}_f^{(1)}\right) + \left(\mathbf{x}_f^{(1)} - \mathbf{x}_g\right)\right]\right\| \\
&\quad = \left\|\mathbf{J}_f(\mathbf{x}_g)\left(\mathbf{x}_f^{(1)} - \mathbf{x}_f\right)\right\|.
\end{aligned}$$

Hence, by Lemma 2, it holds that

$$\left\|\mathbf{J}_f(\mathbf{x}_g)\left(\mathbf{x}_f^{(1)} - \mathbf{x}_f\right)\right\| \leq M\|\mathbf{x}_f - \mathbf{x}_g\|.$$

Then, by Lemma 1, we obtain that

$$\begin{aligned}
\left\|\mathbf{x}_f^{(1)} - \mathbf{x}_f\right\| &\leq \left\|\mathbf{J}_f^{-1}(\mathbf{x}_g)\right\| \left\|\mathbf{J}_f(\mathbf{x}_g)\left(\mathbf{x}_f^{(1)} - \mathbf{x}_g\right)\right\| \\
&\leq \frac{\alpha M}{1-\alpha\beta}\|\mathbf{x}_f - \mathbf{x}_g\| \\
&= \gamma\|\mathbf{x}_f - \mathbf{x}_g\|.
\end{aligned}$$

Furthermore, by the triangular inequality, we have

$$\left\|\mathbf{x}_f^{(1)} - \mathbf{x}_g\right\| \geq \|\mathbf{x}_f - \mathbf{x}_g\| - \left\|\mathbf{x}_f^{(1)} - \mathbf{x}_f\right\| \geq (1-\gamma)\|\mathbf{x}_f - \mathbf{x}_g\|.$$

This completes the proof. ■

Since the fault is detected from the measurement vector $\mathbf{y}$, we would like to have a bound for the error rate in terms of $\|\mathbf{y}_f^{(1)} - \mathbf{y}_g\|/\|\mathbf{y}_g\|$ as well. For this purpose, we introduce a fundamental assumption for the measurement function $\mathbf{h}(\mathbf{x})$

$$\eta_1\|\mathbf{x} - \mathbf{x}'\| \leq \|\mathbf{h}(\mathbf{x}) - \mathbf{h}(\mathbf{x}')\| \leq \eta_2\|\mathbf{x} - \mathbf{x}'\| \tag{17}$$

where $\eta_2 \geq \eta_1 > 0$ are two constants. This inequality basically reflects the fault observability (or strong testability) condition. If the two responses $\mathbf{x}$ and $\mathbf{x}'$ are the same, then the measurements should also be the same; meanwhile, if the two responses $\mathbf{x}$ and $\mathbf{x}'$ are distinguishable, then the measurements $\mathbf{h}(\mathbf{x})$ and $\mathbf{h}(\mathbf{x}')$ should also be distinguishable.

Theorem 2: Let γ be defined as in Theorem 1. If $\gamma < 1$ and the assumption (17) holds, then we have

$$\frac{\left\|\mathbf{y}_f^{(1)} - \mathbf{y}_f\right\|}{\mathbf{y}_g} \leq \frac{\eta_2\gamma}{\eta_1}\frac{\|\mathbf{y}_f - \mathbf{y}_g\|}{\mathbf{y}_g} \tag{18}$$

$$\frac{\left\|\mathbf{y}_f^{(1)} - \mathbf{y}_g\right\|}{\mathbf{y}_g} \geq \frac{\eta_1(1-\gamma)}{\eta_2}\frac{\|\mathbf{y}_f - \mathbf{y}_g\|}{\mathbf{y}_g}. \tag{19}$$

Proof: By Assumption (17) and Theorem 1

$$\begin{aligned}
\left\|\mathbf{y}_f^{(1)} - \mathbf{y}_f\right\| &\leq \eta_2\left\|\mathbf{x}_f^{(1)} - \mathbf{x}_f\right\| \\
&\leq \eta_2\gamma\|\mathbf{x}_f - \mathbf{x}_g\| \\
&\leq \frac{\eta_2\gamma}{\eta_1}\|\mathbf{y}_f - \mathbf{y}_g\|.
\end{aligned}$$

Similarly

$$\begin{aligned}
\left\|\mathbf{y}_f^{(1)} - \mathbf{y}_g\right\| &\geq \eta_1\left\|\mathbf{x}_f^{(1)} - \mathbf{x}_g\right\| \\
&\geq \eta_1(1-\gamma)\|\mathbf{x}_f - \mathbf{x}_g\| \\
&\geq \frac{\eta_1(1-\gamma)}{\eta_2}\|\mathbf{y}_f - \mathbf{y}_g\|.
\end{aligned}$$

Then (18) and (19) follow. ■

Remark 1: For the small perturbation (parametric faults) and weak nonlinearity, i.e., α and M are small, and $\gamma < 1$ holds, inequality (18) tells us that if a fault is sufficiently minor, then the measurement based on one-step relaxation would be very close to that by the real measurement. On the other hand, inequality (19) justifies that one-step relaxation would most likely predict the same faulty behavior as the exact fault simulation would do because a large error rate by the exact solution would imply a large error rate by the one-step relaxation solution.

Remark 2: For large parametric faults or open/short faults, α and M could be large, for which $\gamma < 1$ would not hold in general. In this case, the approximate solution from one-step relaxation would be drastically different from that by exact fault simulation. However, in this case, the difference between the faulty circuit behavior and the good circuit behavior is likely to be so significant that the error rate produced by one-step relaxation would still predict that the circuit is faulty as the exact solution would do.

Remark 3: Since one-step relaxation can reliably detect both small perturbation faults and large variational faults (including open/shorts), statistically it should be able to produce almost the same fault coverage as the exact fault simulation can do for a given large set of faults. This has been observed in our experiments reported in Section VI.

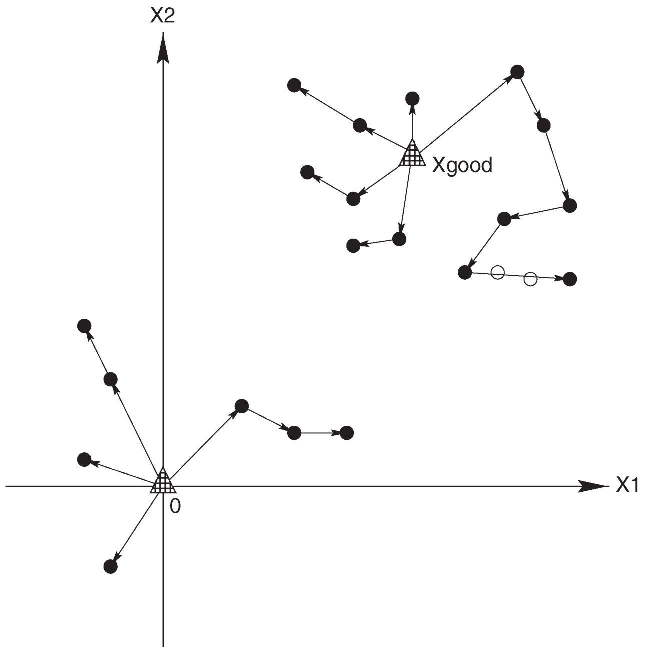

Fig. 2. Illustration of fault simulation continuation.

```
DC_FAULT_SIMULATION_CONTINUATION
1  $\mathbf{x}_g \leftarrow$ result of good circuit simulation;
2  for the $i$th fault in the fault list ($i$ = 1, ..., $m$)
3        $\mathbf{x}_{f,i} \leftarrow$ result of one-step relaxation;
4        $d(\mathbf{x}_{f,i}, \mathbf{x}_g) \leftarrow$ distance between $\mathbf{x}_{f,i}$ and $\mathbf{x}_g$;
5  if the $k$th fault has the minimum distance, then
6        exchange the 1st and $k$th faults;
7  if $d(\mathbf{x}_{f,1}, \mathbf{x}_g) > 1$, then
8        $\mathbf{x}^{(0)} \leftarrow \mathbf{0}$;
9  else
10       $\mathbf{x}^{(0)} \leftarrow \mathbf{x}_g$;
11 for the $i$th fault in the fault list ($i = 1, ..., m$)
12    $\mathbf{x}_{f,i} \leftarrow$ simulation result of the $i$th fault
           using $\mathbf{x}^{(0)}$ as initial point;
13    $d(\mathbf{x}_{f,i}, \mathbf{x}_{f,j}) \leftarrow$ distance between $\mathbf{x}_{f,i}$ and $\mathbf{x}_{f,j}$
           $j = i+1, \cdots, m$;
14    Let $d(\mathbf{x}_{f,i}, \mathbf{x}_{f,k})$ be the minimum distance;
15    if $d(\mathbf{x}_{f,i}, \mathbf{x}_{f,k}) < d(\mathbf{x}_{f,i+1}, \mathbf{x}_g)$, then
16       exchange the $(i+1)$th and $k$th faults;
17       $\mathbf{x}^{(0)} \leftarrow \mathbf{x}_{f,i}$;
18    else if $d(\mathbf{x}_{f,i+1}, \mathbf{x}_g) > 1$, then
19       $\mathbf{x}^{(0)} \leftarrow \mathbf{0}$;
20    else
21       $\mathbf{x}^{(0)} \leftarrow \mathbf{x}_g$;
```

Fig. 3. Continuation-based dc fault simulation algorithm.

To conclude this section, we show that one-step relaxation can be efficiently implemented using Householder's matrix inversion formulas [5]

$$(\mathbf{A}+\mathbf{USW})^{-1} = \mathbf{A}^{-1} - \mathbf{A}^{-1}\mathbf{U}(\mathbf{S}^{-1}+\mathbf{WA}^{-1}\mathbf{U})^{-1}\mathbf{WA}^{-1} \tag{20}$$

where $\mathbf{A}$, $\mathbf{U}$, $\mathbf{S}$, and $\mathbf{W}$ are real matrices with compatible dimensions and the matrices $\mathbf{A}$ and $\mathbf{S}$ are nonsingular. This formula has been applied by many researchers to the fault simulation of linear analog circuits [13], [17]–[19].

In general, a fault in a nonlinear analog circuit changes the Jacobian matrix $\mathbf{J}_g(\mathbf{x}_g)$ to $\mathbf{J}_f(\mathbf{x}_g)$. Suppose that an individual fault occurs to the nonlinear device connected between node i and node j. Such a fault causes the Jacobian matrix to be updated by the expression

$$\mathbf{Y}_f = \mathbf{Y}_g + \Delta g_{ij}\mathbf{e}_{ij}\mathbf{e}_{ij}^T \tag{21}$$

where

$$\mathbf{Y}_g = \mathbf{J}_g(\mathbf{x}_g), \quad \mathbf{Y}_f = \mathbf{J}_f(\mathbf{x}_g)$$

and $\mathbf{e}_{ij}$ is the connectivity (column) vector of the faulty component [17] with zero entries except that the ith entry is 1 and the jth entry is -1 for $i \neq j$. Hence, (8) for the faulty circuit can be rewritten as

$$\left(\mathbf{Y}_g + \Delta g_{ij}\mathbf{e}_{ij}\mathbf{e}_{ij}^T\right)\left(\mathbf{x}_f^{(1)} - \mathbf{x}_g\right) = \mathbf{w} \tag{22}$$

where $\mathbf{w} = -\mathbf{f}(\mathbf{x}_g)$. From Householder's formula, we have

$$\mathbf{Y}_f^{-1} = \mathbf{Y}_g^{-1} - \frac{\mathbf{Y}_g^{-1}\mathbf{e}_{ij}\mathbf{e}_{ij}^T}{(\Delta g_{ij})^{-1} + \mathbf{e}_{ij}^T\mathbf{Y}_g^{-1}\mathbf{e}_{ij}}\mathbf{Y}_g^{-1}. \tag{23}$$

This equation can further be simplified for each specific type of faults.

- Parametric faults

$$\mathbf{x}_f^{(1)} = \mathbf{x}_g + \left(\mathbf{I} - \frac{\mathbf{Y}_g^{-1}\mathbf{e}_{ij}\mathbf{e}_{ij}^T}{(\Delta g_{ij})^{-1} + \mathbf{e}_{ij}^T\mathbf{Y}_g^{-1}\mathbf{e}_{ij}}\right)\mathbf{Y}_g^{-1}\mathbf{w}. \tag{24}$$

- Open fault: They can be modeled by setting Δg_{ij} to $-Y_{ij}$, then

$$\mathbf{x}_f^{(1)} = \mathbf{x}_g + \left(\mathbf{I} - \frac{\mathbf{Y}_g^{-1}\mathbf{e}_{ij}\mathbf{e}_{ij}^T}{-Y_{ij}^{-1} + \mathbf{e}_{ij}^T\mathbf{Y}_g^{-1}\mathbf{e}_{ij}}\right)\mathbf{Y}_g^{-1}\mathbf{w}. \tag{25}$$

- Short fault: They can be modeled by setting Δg_{ij} to ∞, then

$$\mathbf{x}_f^{(1)} = \mathbf{x}_g + \left(\mathbf{I} - \frac{\mathbf{Y}_g^{-1}\mathbf{e}_{ij}\mathbf{e}_{ij}^T}{\mathbf{e}_{ij}^T\mathbf{Y}_g^{-1}\mathbf{e}_{ij}}\right)\mathbf{Y}_g^{-1}\mathbf{w}. \tag{26}$$

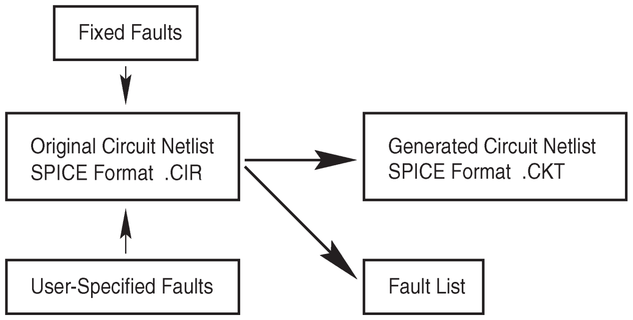

Fig. 4. Architecture of the dc fault list generator.

TABLE I
SUMMARY OF FAULTS HANDLED BY THE FAULT LIST GENERATOR

Device	Fixed Faults	User-Specified Faults
Resistor	short, open	parameter deviation
Diode	short, open	model parameter deviation
BJT	open at C, B, E	model parameter deviation
	short between C-B, C-E, and B-E	pipe between E-C
MOS	open at D, G, S	model parameter deviation
	short between D-G, D-S, and G-S	pinhole between G-S, G-D, and G-channel
Interconnect	—	short, open

TABLE II
DC FAULT COVERAGE OF CIRCUITSIM90 BENCHMARK CIRCUITS

Circuit	No. of Devices			No. of Faults	No. Iter. Good Cir.	Output Voltage Measurement				Supply Current Measurement			
	BJT	MOSFET	RES			Full		One-Step		Full		One Step	
astabl	2	0	4	112	9	56	50%	56	50%	60	54%	59	53%
bias	13	0	5	308	14	147	47%	143	46%	47	20%	59	19%
bjtff	41	0	26	1176	25	392	33%	392	33%	540	46%	551	47%
latch	14	0	10	72	12	72	100%	72	100%	72	100%	72	100%
loc	96	0	276	180	45	44	24%	44	24%	122	68%	122	68%
nagle	23	0	11	588	28	225	38%	224	38%	389	66%	372	63%
rca	11	0	12	416	6	304	73%	304	73%	317	76%	317	76%
schmitecl	4	0	8	224	31	177	79%	178	79%	170	76%	171	76%
vreg	20	0	10	520	251	343	70%	327	63%	361	69%	355	68%
ab_ac	0	31	1	1384	142	1188	86%	1188	86%	1083	78%	1100	79%
ab_integ	0	31	3	1424	16	989	69%	945	66%	1118	79%	1116	78%
ab_opamp	0	31	24	1444	11	903	63%	847	59%	1105	77%	1098	76%
e1480	0	28	130	60	25	1	2%	1	2%	1	2%	1	2%
gm6	0	5	0	220	7	152	69%	152	69%	161	73%	161	73%
hussamp	0	16	1	724	21	602	83%	602	83%	606	84%	606	84%
mosrect	0	4	2	216	12	70	32%	70	32%	112	52%	112	52%
nand	0	25	0	1100	6	131	12%	131	12%	265	24%	265	24%
pump	0	1	0	44	3	0	0%	0	0%	3	7%	3	7%
reg0	0	13	30	600	3	233	39%	233	39%	351	59%	351	59%
ring	0	34	0	1496	6	506	34%	506	34%	1088	73%	1088	73%
schmitfast	0	6	0	264	8	156	59%	155	59%	181	69%	181	69%
schmitslow	0	8	0	352	9	199	57%	199	57%	243	69%	243	69%
slowlatch	0	14	1	636	11	388	61%	388	61%	391	61%	391	61%
gm1	0	46	7	276	16	276	100%	276	100%	276	100%	276	100%
gm17	0	56	3	764	173	366	48%	376	49%	354	46%	360	47%
gm19	0	162	1	460	23	440	96%	454	99%	450	98%	460	100%
gm2	0	7	0	44	8	44	100%	44	100%	44	100%	44	100%
mike2	0	12	0	528	9	131	25%	130	25%	527	100%	528	100%
todd3	0	13	1	592	20	379	64%	378	64%	399	67%	396	67%
Average	8	29	20	559	33	316	57%	313	57%	374	65%	374	65%

Note that (24)–(26) can be computed very efficiently without any matrix inversion [17]. Furthermore, open and short faults do not cause any numerical difficulty.

IV. ADAPTIVE SIMULATION CONTINUATION

It is well known that the convergence property of the Newton–Raphson algorithm for solving (1) depends on the choice of the initial point $\mathbf{x}^{(0)}$. However, choosing a good initial point is not an easy task in general. In SPICE, the default initial point is set to $\mathbf{0}$.

Homotopy simulation continuation is a technique used in nonlinear circuit simulation for improving the convergence property of difficult-to-converge circuits [14]. The basic idea is to construct a continuum of circuits parameterized by a scalar parameter in such a way that one end of the continuum is the target circuit to be solved while the other end of the continuum is a "pseudo" circuit that is easily solved. By solving the circuits in the continuum successively using the previous solution as the initial point for the next circuit, the difficult-to-converge circuit is finally solved with success. Two simple schemes known as Gmin stepping and source stepping are implemented in SPICE [9], while more complicated schemes involve the design of circuit traces (a list of "pseudo" circuits), using the homotopy concept [14].

The homotopy idea can also be used for exact dc fault simulation. Given a list of faults, fault simulation is performed in such an order that the simulation result of the previous fault is used as the initial point for the next fault simulation. The faults are ordered in such a way that the total number of iterations is minimized. In case difficult-to-converge fault circuits are encountered, "pseudo" faults can be inserted into the simulation sequence to help the convergence.

This idea is especially attractive for parametric faults and those structural faults that do not cause the catastrophic failure of the circuit. For catastrophic faults with dramatically different responses, one can use either $\mathbf{0}$ or the good circuit response $\mathbf{x}_g$ as the initial point for fault simulation. Fig. 2 illustrates the simulation traces based on this idea, where solid circles indicate fault responses and empty circles indicate those "pseudo" fault responses added to help the convergence. The two shadowed triangles mark the good circuit response $\mathbf{x}_g$ and the origin $\mathbf{0}$, respectively.

Suppose that there are m faulty circuits. Our adaptive fault ordering scheme consists of three components. First, the one-step Newton–Raphson iteration is performed for each fault using the good circuit response $\mathbf{x}_g$ as the initial point, and the results are denoted $\mathbf{x}_{f,i}$, $i = 1, 2, \ldots, m$. Second, for the list of faults, the results of fault simulation by one-step relaxation are ordered by their distances $d(\mathbf{x}_{f,i}, \mathbf{x}_g)$ to the good circuit response with the distance $d(\cdot, \cdot)$ defined in (6). Third, once faults are accurately simulated, their responses can be used to update the distances between faulty circuits.

A greedy algorithm that implements the above three components is described in pseudocode in Fig. 3. The faulty circuit with minimum distance to the good circuit will be selected as the first to simulate. If its distance is less than 1, then $\mathbf{x}_g$ is used as the initial point; otherwise, $\mathbf{0}$ is used. This process is repeated for all the faults in the list. Each time, the distances from the result of previously simulated fault to that of all the unfinished faulty circuits (approximated by one-step relaxation) are calculated and used for selecting the next faulty circuit together with the next initial point to complete the simulation.

V. EXPERIMENTAL SETTING

The proposed two techniques, one-step relaxation and adaptive simulation continuation, have been implemented in SPICE3F5. A set of benchmark circuits from the 1990 MCNC Circuit Simulation and

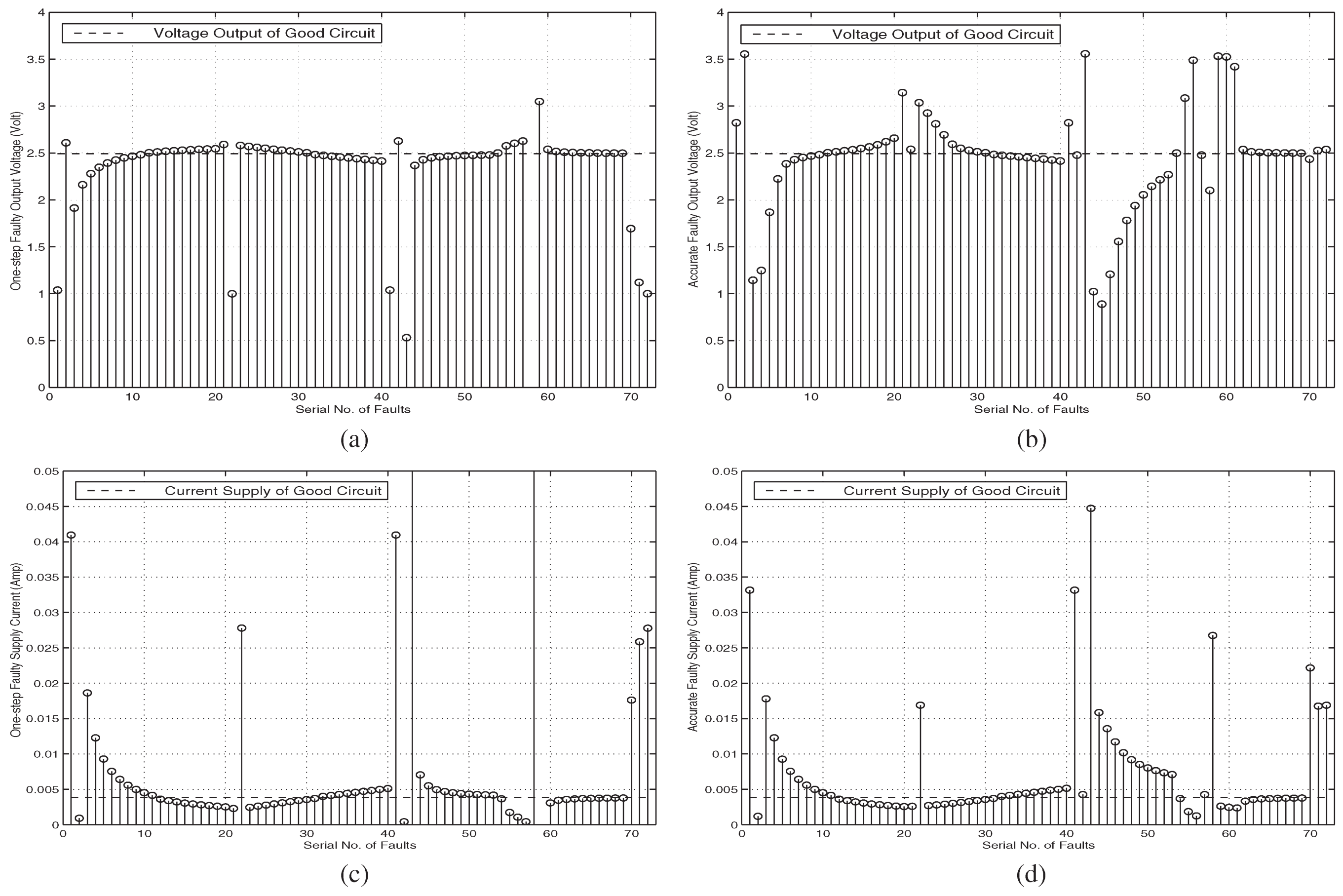

Fig. 5. One-step relaxation and accurate simulation results of the static latch circuit. (a) Output voltage by one-step simulation. (b) Output voltage by accurate simulation. (c) Supply current by one-step simulation. (d) Supply current by accurate simulation.

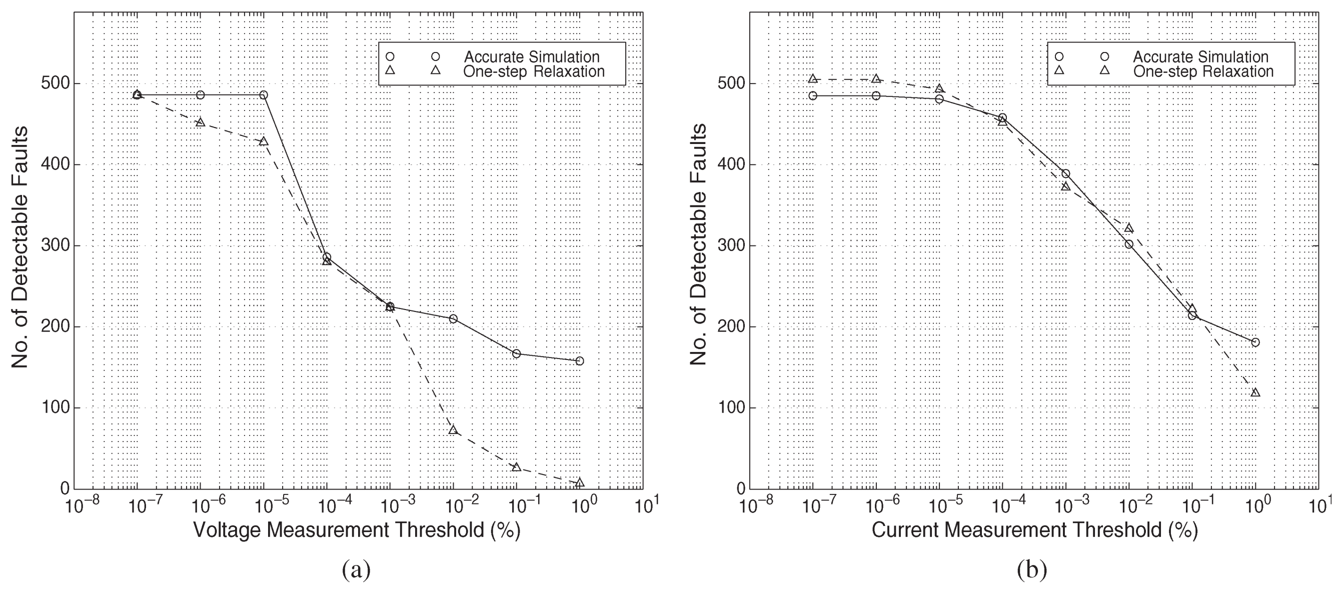

Fig. 6. Number of detectable faults versus measurement threshold for the μA741 operational amplifier. (a) Output voltage measurement. (b) Supply current measurement.

Modeling Workshop [7] was used to test the proposed methods. We also implemented an automatic fault list generator to insert faults into the benchmark circuits.

The structure of our fault list generator is shown in Fig. 4. It supports two methods of fault injection: fixed and user specified. Fixed faults are device-oriented hard (i.e., open or short) faults. For example, a resistor can have one short fault and one open fault; a transistor has a total of six short/open faults. For a MOS transistor, there are three shorts (between gate–drain, gate–source, drain–source) and three opens (at gate, drain, source). User-specified faults include parametric faults

TABLE III
SPEEDUP RESULTS OF CIRCUITSIM90 BENCHMARK CIRCUIT FAULT SIMULATION

Circuit	No. of Devices			No. of	No. Iter	Init. Point **0**		Init. Point $\mathbf{x}_g$		Fault Ordering	
	BJT	MOS	RES	Faults	Good Cir.	No. Iter	CPU sec.	No. Iter	Speedup	No. Iter	Speedup
astabl	2	0	4	112	9	1112	1.02	372	2.99	340	3.27
bias	13	0	5	308	14	10747	30.14	8838	1.22	8074	1.33
bjtff	41	0	26	1176	25	34518	409.88	7882	4.38	6729	5.13
latch	14	0	10	72	12	1083	3.04	667	1.62	573	1.89
loc	96	0	276	180	45	8010	326.38	612	13.09	535	14.97
nagle	23	0	11	588	28	15712	77.44	5519	2.85	4874	3.22
rca	11	0	12	416	6	3438	16.52	2509	1.37	2185	1.57
schmitecl	4	0	8	224	31	13944	12.02	1407	9.91	1359	10.26
vreg	20	0	10	520	251	87568	201.60	23823	3.68	9308	9.41
ab_ac	0	31	1	1384	142	67122	563.16	12808	5.24	10468	6.41
ab_integ	0	31	3	1424	16	35207	406.40	10611	3.32	10566	3.33
ab_opamp	0	31	24	1444	11	23925	349.10	8620	2.78	8575	2.79
e1480	0	28	130	60	25	1626	12.74	324	5.02	253	6.43
gm6	0	5	0	220	7	1539	4.56	590	2.61	549	2.80
hussamp	0	16	1	724	21	59001	155.3	54862	1.08	20867	2.83
mosrect	0	4	2	216	12	2257	5.88	506	4.46	371	6.08
mux8	0	64	0	2818	11	34473	1531.56	11488	3.00	11117	3.10
nand	0	25	0	1100	6	6796	146.86	2210	3.08	1980	3.43
pump	0	1	0	44	3	132	0.40	44	3.00	44	3.00
reg0	0	13	30	600	3	1800	12.74	951	1.89	907	1.98
ring	0	34	0	1496	6	43823	467.40	57849	0.76	11699	3.75
schmitfast	0	6	0	264	8	2221	7.66	790	2.81	665	3.34
schmitslow	0	8	0	352	9	3137	13.70	1099	2.85	839	3.74
slowlatch	0	14	1	636	11	10095	51.34	4368	2.31	3833	2.63
toronto	0	58	0	2552	30	43356	1227.32	14679	2.95	12059	3.60
arom	0	116	2	5144	11	57520	6130.72	9822	5.86	9439	6.09
b330	0	330	0	14520	14	342553	33340.57	196385	1.74	115113	2.98
counter	0	220	0	9680	19	212690	27392.62	77222	2.75	57688	3.69
gm1	0	46	7	276	16	4416	31.42	4278	1.03	828	5.33
gm17	0	56	3	764	173	92810	526.56	25191	3.68	24727	3.75
gm19	0	162	1	460	23	20892	470.18	25163	0.83	13234	1.58
gm2	0	7	0	44	8	350	0.68	304	1.15	117	2.99
jge	0	348	0	15312	31	320292	27262.29	122552	2.61	73948	4.33
mike2	0	12	0	528	9	5033	30.46	5669	0.89	2297	2.19
rich3	0	106	2	4704	18	84392	5288.18	12014	7.02	9677	8.72
todd3	0	13	1	592	20	17878	60.48	4030	4.44	3842	4.65
Average	6	50	16	1971	30	46430	2193.0	19890	2.7	12213	4.4

and global interconnect short/open faults. Table I summarizes the types of faults handled by the fault list generator.

In our experiments, the following set of user-specified faults are injected into all of the benchmark circuits.

- Resistors: ±10%–±90% parameter deviations with 10% incremental.
- BJT transistors: Emitter–collector pipe is modeled by 500 Ω–5 kΩ resistors with 500 Ω incremental.
- MOS transistors: ±10%–±90% width/length ratio deviations with 10% incremental. Pinholes between G–S and G–D are modeled by 500 Ω–kΩ resistors with 500 Ω incremental.

In the current implementation, the fault list generator does not inject faults into subcircuits described in the SPICE netlist. Therefore, the total number of faults is not proportional to the number of devices for benchmark netlists containing subcircuits.

We note that in order to simulate certain faults, especially shorts, extra circuit nodes have to be added into the original netlist. Therefore, the fault list generator creates not only the fault list but also the expanded circuit netlist in SPICE format. In practice, it is recommended that the IFA be used to generate the realistic set of faults [15].

VI. EXPERIMENTAL RESULTS

The set of benchmark circuits we used consists of 29 circuits. Their fault statistics are summarized in the columns *No. of Devices* and *No. of Faults* in Table II. They are classified into three groups. The first group contains nine BJT circuits, the second group contains 14 MOSFET circuits using the level 2 MOSFET model, and the last group contains six MOSFET circuits using the level 3 MOSFET model. The column *No. Iter. Good Cir.* gives the numbers of iterations required for the simulation of good circuits.

To validate one-step relaxation, we conducted two sets of experiments by choosing two different measurements. First, the output voltage is used for measurement. For circuits with multiple outputs, the results for the output terminal that detects most faults are given. The column *Output Voltage Measurement* in Table II lists the number of faults detected and the fault coverage by accurate fault simulation (subcolumn *Full*) and by one-step relaxation (subcolumn *One-Step*). It can be seen that fault coverages by accurate fault simulation and by one-step relaxation are identical for 23 out of 29 circuits and differ only in a few percents for the rest of the six circuits.

Second, the supply current, i.e., the current flowing through VCC for BJT circuits or the current through VDD for MOSFET circuits, is used for measurement. The results are given in the *Supply Current Measurement* column in Table II. It is seen that fault coverages by accurate fault simulation (subcolumn *Full*) and by one-step relaxation (subcolumn *One Step*) are identical for 19 out of 29 circuits and differ in only 1%–3% for the rest of the ten circuits.

To illustrate how far the results of one-step relaxation are away from that of accurate simulation, Fig. 5 plots the results of simulating

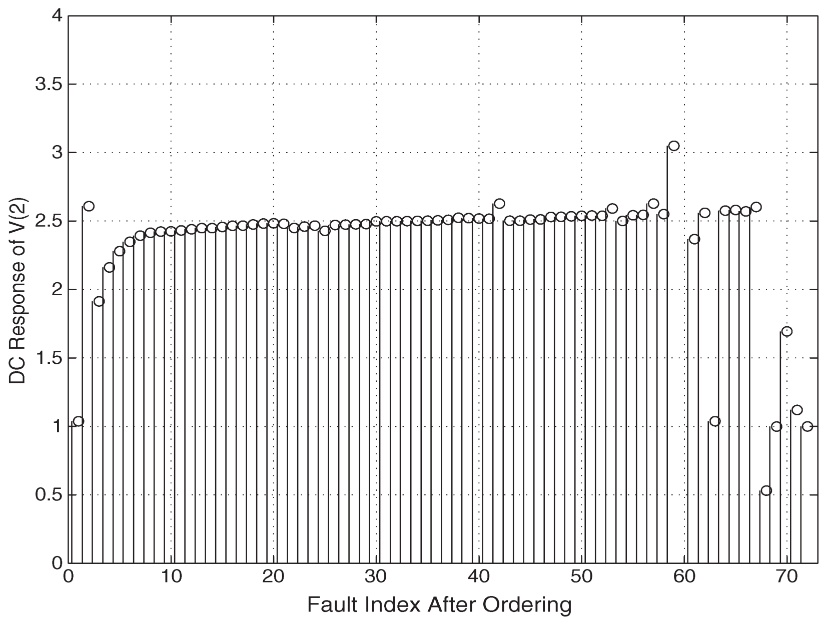

Fig. 7. One-step relaxation results after ordering.

70 faults (one after another) in the static latch circuit latch in Table II. The output voltage and supply current of the good circuit are also plotted in dash lines in the same plots for reference. The plots show that the one-step relaxation results approximate very well the accurate simulation results for a majority of faults.

The detection threshold for both voltage and current measurements is 0.1% in the latch example above. Plotted in Fig. 6 are the numbers of faults detected versus measurement threshold for Nagle's μA741 operational amplifier in Table II using one-step relaxation and exact fault simulation for the two separate choices of measurements. The total number of faults injected was 588. We can see that one-step relaxation may overestimate or underestimate fault coverage dependent upon the chosen threshold, but in general it is a good approximation of accurate simulation for the purpose of fault detection.

The speedup results are shown in Table III. Listed in column *Init. Point* $\mathbf{0}$ are the total number of iterations and the CPU time used for simulating all the faulty circuits using $\mathbf{0}$ as the initial point (stand-alone fault simulation). In column *Init. Point* $\mathbf{x}_g$, one subcolumn gives the total number of iterations used for simulating all the faulty circuits using $\mathbf{x}_g$ (the result of the good circuit simulation) as the initial point, and the other subcolumn *Speedup* lists the ratio of the number of iterations compared with the one in the column *Init. Point* 0. The two subcolumns under the column *Fault Ordering* list the total number of iterations required by the adaptive simulation continuation method with ordering and its speedup over the stand-alone fault simulation method. The CPU time is collected on an UltraSparc workstation.

We note that speedup depends heavily on each individual circuit and its injected faults. The proposed adaptive simulation continuation method with fault ordering achieved an average speedup of 4.4, and as high as 15, over the stand-alone fault simulation. If good circuit response is used as the initial point for all faulty circuits, an average speedup of 2.7 was achieved. We have observed in our experiments that the proposed method offers a significant improvement for the following cases:

- when the number of iterations required per circuit simulation is substantial;
- when a circuit has many parametric faults to simulate.

Finally, we illustrate by an example the effect of adaptive fault ordering used in our experiment. For the static latch circuit in Table III, Fig. 7 shows the result of dc responses ordered by the adaptive ordering algorithm described in Fig. 3. One can see a well-ordered dc response, which indicates continuity in the sense that the dc responses of two successive simulations are close to each other in the distance metric we defined before. For this circuit, the total numbers of Newton–Raphson iterations required by stand-alone fault simulation, fault simulation using $\mathbf{x}_g$ as the initial point, and adaptive fault simulation continuation (with ordering) were 1083, 667, and 573, respectively. If simulation continuation is applied directly to the fault list without performing fault ordering, the total number of iterations required was 1415.

VII. Conclusion

This paper has investigated a simple idea of performing only one-step Newton–Raphson iteration, called one-step relaxation, for approximate analog dc fault simulation, as well as its application to fault ordering for exact dc fault simulation. The effectiveness of one-step relaxation for dc fault simulation has been analyzed theoretically and validated experimentally. In addition, one-step relaxation has been explored for fault ordering to speed up exact fault simulation based on simulation continuation.

References

[1] A. Chatterjee, B. C. Kim, and N. Nagi, "DC built-in self-test for linear analog circuits," *IEEE Des. Test Comput.*, vol. 13, no. 2, pp. 26–33, 1996.
[2] G. Devarayanadurg and M. Soma, "Analytical fault modeling and static test generation for analog ICs," in *Proc. IEEE/ACM Int. Conf. Computer-Aided Design (ICCAD)*, San Jose, CA, Nov. 1994, pp. 44–47.
[3] N. Engin and H. G. Kerkhoff, "Fast fault simulation for nonlinear analog circuits," *IEEE Des. Test Comput.*, vol. 20, no. 2, pp. 40–47, Mar. 2003.
[4] J. Hou and A. Chatterjee, "Concurrent transient fault simulation for analog circuits," *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, vol. 22, no. 10, pp. 1385–1398, Oct. 2003.
[5] A. S. Householder, "A survey of some closed methods for inverting matrices," *SIAM J. Appl. Math.*, vol. 5, no. 3, pp. 155–169, Sep. 1957.
[6] S. D. Huss and R. S. Gyurcsik, "Optimal ordering of analog integrated circuit tests to minimize test time," in *Proc. IEEE/ACM Design Automation Conf.*, San Francisco, CA, Jun. 1991, pp. 494–499.
[7] "CircuitSim90," *Circuit Simulation and Modeling Workshop at MCNC*, 1990. [Online]. Available: http://www.intusoft.com/benchmarks.htm

[8] L. Milor and A. L. Sangiovanni-Vincentelli, "Minimizing production test time to detect faults in analog circuits," *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, vol. 13, no. 6, pp. 796–813, Jun. 1994.
[9] L. W. Nagel, "SPICE2: A computer program to simulate semiconductor circuits," Dept. Elect. Comput. Eng., Univ. California, Berkeley, ERL Memorandum M520, May 1975.
[10] N. Nagi, A. Chatterjee, and J. A. Abraham, "Fault simulation of linear analog circuits," *Analog Integr. Circuits Signal Process.*, vol. 4, no. 3, pp. 245–260, Nov. 1993.
[11] M. J. Ohletz, "Realistic faults mapping scheme for the fault simulation of integrated analogue CMOS circuits," in *Proc. Int. Test Conf.*, Washington DC, 1996, pp. 776–785.
[12] J. M. Ortega and W. C. Rheinboldt, *Iterative Solution of Nonlinear Equations in Several Variables*. New York: Academic, 1970.
[13] A. Pahwa and R. A. Rohrer, "Band-faults: Efficient approximations to fault bands for the simulation before fault diagnosis of linear circuits," *IEEE Trans. Circuits Syst.*, vol. CAS-29, no. 2, pp. 81–88, Feb. 1982.
[14] J. S. Roychowdhury and R. C. Melville, "Homotopy techniques for obtaining a DC solution of large-scale MOS circuits," in *Proc. IEEE/ACM Design Automation Conf.*, Las Vegas, NV, Jun. 1996, pp. 286–291.
[15] M. Sachdev and B. Atzema, "Industrial relevance of analog IFA: A fact or a fiction," in *Proc. Int. Test Conf.*, Washington DC, 1995, pp. 61–70.
[16] C.-J. Shi, "Fault simulation," in *Analog and Mixed-Signal Test*, B. Vinnakota, Ed. Upper Saddle River, NJ: Prentice-Hall, 1998, ch. 3, pp. 55–92.
[17] K. Singhal and J. Vlach, "Solution of modified systems and applications," in *Proc. IEEE Int. Symp. Circuits and Systems*, Chicago, IL, 1981, pp. 620–623.
[18] G. C. Temes, "Efficient methods of fault simulation," in *Proc. 20th Midwest Symp. Circuits Systems*, Lubbock, TX, Aug. 1977, pp. 191–194.
[19] M. W. Tian and C.-J. Shi, "Rapid frequency-domain analog fault simulation under parameter tolerances," in *Proc. 34th IEEE/ACM Design Automation Conf. (DAC)*, Anaheim, CA, Jun. 1997, pp. 275–280.
[20] ——, "Efficient DC fault simulation of nonlinear analog circuits," in *Proc. Design, Automation, and Test Eur. Conf. and Exhibition (DATE)*, Paris, France, Feb. 1998, pp. 899–904.
[21] ——, "Nonlinear analog DC fault simulation by one-step relaxation," in *Proc. IEEE VLSI Test Symp. (VTS)*, Monterey, CA, Apr. 1998, pp. 126–131.
[22] M. Zwolinski, A. D. Brown, and C. D. Chalk, "Concurrent analogue fault simulation," in *Proc. 3rd IEEE Int. Mixed-Signal Testing Workshop*, Seattle, WA, 1997, pp. 824–838.

Analytical Model for Crosstalk and Intersymbol Interference in Point-to-Point Buses

Sampo Tuuna, Jouni Isoaho, and Hannu Tenhunen

***Abstract*—In this paper, an analytical model to estimate crosstalk noise and intersymbol interference on capacitively and inductively coupled point-to-point on-chip buses is derived. The derived closed-form equation for output voltage enables the usage of the model in computer-aided design (CAD) tools for complex systems where high simulation speed is essential. The model also combines together properties such as inductive coupling, initial conditions, signal rise time, input phases, and bit sequences that have not been included in a single closed-form model before. The model is compared to HSPICE and previous models. The model and HSPICE are in good agreement with each other.**

***Index Terms*—Crosstalk, integrated circuit, interconnect, intersymbol interference, signal integrity.**

Manuscript received August 22, 2004; revised December 6, 2004, and March 3, 2005. This paper was recommended by Associate Editor D. Blaauw.

S. Tuuna and J. Isoaho are with the Department of Information Technology, University of Turku, Turku 20520, Finland (e-mail: satatu@utu.fi; jisoaho@utu.fi).

H. Tenhunen is with the Royal Institute of Technology, Kista 16440, Sweden (e-mail: hannu@imit.kth.se).

Digital Object Identifier 10.1109/TCAD.2005.855973

I. Introduction

Global communication has become a bottleneck in deep submicrometer integrated circuits. The speed of communication is not limited by the communicating devices themselves, but the wires connecting them. Wires in modern integrated circuits have resistive, capacitive, and inductive parasitics that affect the signals propagating on them. These parasitics distort signal waveforms and induce noise. One major source of noise is crosstalk, which is caused by capacitive and inductive coupling between wires. Crosstalk may cause logic hazards and signal delay variation. Crosstalk noise avoidance is especially important for on-chip buses, since, in buses, several interconnects run parallel to each other for long distances.

Signals traveling on buses may also corrupt later ones if the bus does not reach steady state between signals. This intersymbol interference is caused by stored energy in reflections, circuit ringing, and charge storage [1]. Ringing and temporary charge storage in interconnects are problematic in buses, where interconnects are wide, resulting in a large capacitive load and inductive noise. The amount of crosstalk noise and intersymbol interference depends not only on the electrical properties of interconnects, but also on the signal transitions. Capacitive coupling causes an increase in propagation delay when coupled interconnects are switching in the opposite direction, and a decrease when they are switching in the same direction. Furthermore, the induced crosstalk voltage on a quiet interconnect depends on the transition activity of neighboring interconnects. Intersymbol interference, on the other hand, is dependent on successive transitions. Certain bit sequences can cause an interconnect not to reach steady state.

Signals may also arrive at different phases due to unbalanced signal paths, or because of deliberate timing intervals to reduce crosstalk noise or peak current draw [2]. The relative input timing of aggressor and victim nets influences strongly the coupling noise on a victim net [3], [4].

Crosstalk noise modeling approaches can be loosely classified into two categories based on their tradeoff between accuracy and efficiency [5]. The first class of methods aims to achieve maximum accuracy and a substantial speedup over SPICE via model order reduction [6], [7]. These methods are useful for verification where accuracy is a key requirement. The second class aims to further improve the efficiency of noise analysis via simple template circuits that are analytically modeled. These methods are useful in physical design automation tools to assess and avoid crosstalk noise in the early stages of the design flow.

Over the last decade, several such analytical models have been proposed for the estimation of crosstalk noise in coupled interconnects. In [8], a model for up to five coupled lines has been presented. However, the model is based on a single L-segment and does not consider inductance. In [9] and [10], a Π-model is used, but inductance is neglected. A model for a bus consisting of distributed resistance–capacitance (RC) lines has been suggested in [11], but signal rise times and inductance are not included. It has also been assumed that every other wire in the bus carries the same signal. A propagation delay model for a bus has been presented in [12], but the model does not consider inductance and is also based on switch factor analysis. In [13], a model for two distributed resistance–inductance–capacitance (RLC) wires has been presented, but inductive coupling has been ignored. It has been shown that the effects of inductive coupling can be significant for long interconnects [14]. The accuracy of RC models in crosstalk evaluation is also no longer sufficient for deep submicrometer circuits [15].

None of the mentioned models take into account both crosstalk and intersymbol interference. In this paper, an analytical RLC Π-model