PROCEEDINGS OF THE SIXTH INTERNATIONAL NASECODE CONFERENCE
Edited by J. J. H. Miller—Dublin (Ireland) 11th to 14th July 1989 — Boole Press Ltd.

Techniques for Switching Power Converter Simulation

M. Reichelt, J. White
Research Laboratory of Electronics
Dept. of Electrical Engineering and Computer Science
Massachusetts Institute of Technology
Cambridge, MA 02139

Abstract

In this paper we describe techniques for accelerating two types of simulations used by switching power converter designers. The first is a technique for accelerating circuit level transient simulation of switching converters. Such simulations are computationally expensive because converters are clocked at a frequency whose period is orders of magnitude smaller than the time interval of interest to the designer. It is possible to reduce the simulation time without compromising much accuracy by computing "envelopes" of the high-frequency clock. Recently developed envelope-following techniques are described, and results simulating switching converters presented. As distributed effects in power devices contribute to switching converter efficiency, device-level transient simulation is also used by power converter designers. In the second part of this paper we describe a waveform relaxation-based technique for device-level transient simulation. Preliminary results for 2D MOS transistor simulation are presented.

1 Introduction

In general, switching power converter designers rely heavily on circuit simulation programs like SPICE [nagel75] to insure the correctness of their circuit-level designs, and device-level simulation programs like MINIMOS [selberherr84] or PISCES [pinto84] to determine the performance of the transistor switches. In this paper we describe techniques for accelerating these two types of simulations. In the first part of this paper, a technique for accelerating circuit-level transient simulation of switching converters is presented. Such simulations are computationally expensive because converters are clocked at a frequency whose period is orders of magnitude smaller than the time interval of interest to the designer. It is possible to reduce the simulation time without compromising much accuracy by computing "envelopes" of the high-frequency clock. Recently developed envelope-following techniques are described, and implementation results given. In the second part of this paper, a waveform-relaxation (WR) based technique for device-level transient simulation is described. Standard spatial discretization techniques are used to convert the semiconductor device equations into a large, sparsely-connected system of algebraic and ordinary differential equations in time, to which we apply WR. There are several reasons for investigating WR for device transient simulation: different sections of the device can use different timesteps; the equations can be solved in a more decomposed fashion which is suitable for parallel implementations; and we like WR. Finally, preliminary results for WR applied to 2D MOS transistor simulations are presented. In particular, we show that WR is at least as fast as direct methods for transient device simulation, and we show that the Poisson equation dominates the WR convergence at all time scales.

2 Envelope Following

Consider the simplified "buck" converter circuit given in Fig. 1. The voltage waveforms in steady state for the switch and output nodes are plotted in Fig. 1. As can be seen from the plots, the switch node waveform is not very smooth over a clock cycle, and for this reason the numerical integration algorithm used to compute the voltages placed more than twenty timepoints per clock cycle. Additionally, in a typical buck converter, the RLC time constant is large compared to the switching clock period, and therefore the power-up transient spans many clock cycles. In Fig. 3, the output voltage waveform for the entire power-up transient is plotted. As can be seen from the fuzziness of the plot, in this case the power-up transient spans more than 100 cycles of the high frequency clock, and the total simulation used more than 2000 timepoints.

The infeasibility of simulating such circuits with classical techniques has led designers to explore a variety of fast approximate simulation techniques based on treating the switching converter's switches as ideal, and the remaining circuitry as linear[hsiao87]. Approximate techniques such as these can reduce the cost of computing the behavior of a switching converter circuit over one high-frequency clock cycle to the point where it becomes computationally feasible to simulate the circuit for the hundreds of cycles needed to construct a complete transient.

Programs based on the above techniques are reasonably efficient, but they are based on idealizations of the circuits involved which may eliminate behavior that is important to a designer. Another approach to transient simulation of switching power circuits that does not require simplifying the circuit exploits the fact that a designer typically is not interested in the details of the behavior in every cycle, but rather is interested in the "envelope" of the behavior. Such methods, referred to as **envelope following** methods[petzold81], can much more efficient than classical methods

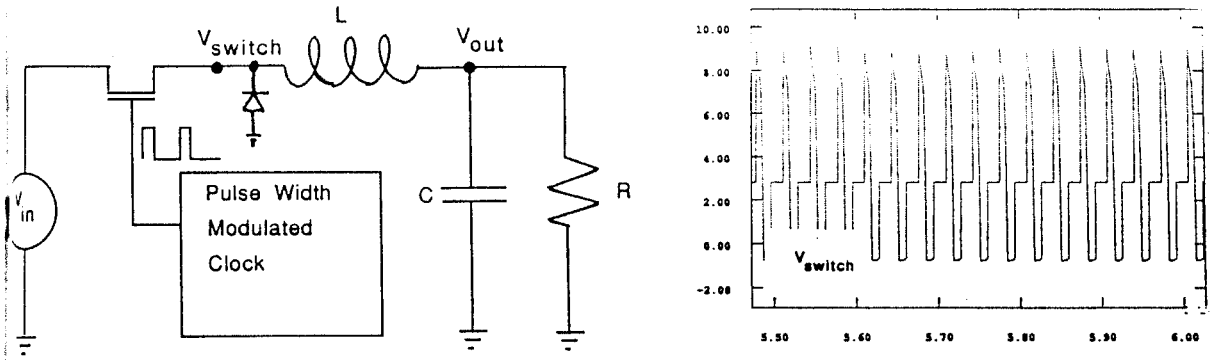

Figure 1: A Buck Converter And Switch Waveform

when the clock period is small compared to the simulation interval. Specifically, we define the "envelope" to be a continuous function derived by interpolating the sequence formed by sampling the state every clock period T. The key advantage of considering just the envelope is that if the sequence formed by sampling the state at the beginning of each clock cycle, $x(0), x(T), x(2T), \ldots, x(mT), \ldots$, changes slowly as a function of m, the clock cycle number, it is possible to approximate the envelope without computing every clock cycle.

In the next section we derive a simple method for computing envelopes which involves solving a sequence of two-point boundary value problems. The two-point boundary value problems are solved with a "shooting" or Newton method, as described in Section 2.3. Their implementation in the program *Nitswit* along with results from using *Nitswit* to simulate several switching power circuits is described in Section 2.4.

2.1 Computing the Envelope

Most switching power converter circuits can be described by a system of differential equations of the form

$$\frac{d}{dt}p(x(t), u(t)) + f(x(t), u(t)) = 0, \tag{1}$$

where $x(t) \in \Re^N$, the state, is the vector of capacitor voltages and inductor currents, $u(t) \in \Re^M$ is the vector of input sources, $p(x(t), u(t)) \in \Re^N$ is the vector of capacitor charges and inductor fluxes, and $f(x(t), u(t)) \in \Re^N$ is the vector of resistive currents and inductor voltages. If the state x is known at some time t_0, it is possible to solve (1) and compute the state at some later time t_1. In general, one can write

$$x(t_1) = x(t_0) + \phi(x(t_0), t_0, t_1) \tag{2}$$

where $\phi : \Re^N \times \Re \times \Re \rightarrow \Re^N$ is a state transition function for the differential equation.

A "differential-like" equation can also be written for the envelope, that is the sequence $x(0), x(T), x(2T), \ldots$ associated with one envelope of the solution to (1). Applying (2), the elements of the sequence can be related by

$$x(mT) - x((m-1)T) = \phi(x((m-1)T), (m-1)T, mT). \tag{3}$$

The relation in (3) indicates how rapidly the initial point of each clock cycle changes from one cycle to the next, and in that sense is like a differential equation. This similarity can be exploited to derive methods for approximately solving for the $x(mT)'s$. For example, the value of $x((m+l)T)$ can be approximated by $x((m+1)T)+(l-1)\phi(x(mT), mT, (m+1)T)$, which is loosely analogous to solving a differential equation by forward Euler.

To compute an envelope for a system with period T using a forward-Euler envelope-following algorithm with a fixed cyclestep l, a simple repetitive two-step process can be used. Given $x(0)$, the first step is to calculate $x(T)$ by solving (1) over the interval $[0, T]$ using a standard discretization technique. The second step is to set $x(lT) = x(T) + (l-1)[x(T) - x(0)]$. This process is repeated to compute $x(2lT), x(3lT), \ldots$. Note that calculating the solution over a long interval only requires solving the differential equation every l^{th} cycle.

Although simple to describe, a forward-Euler-based envelope-following method is not very effective for solving switching circuits because maintaining stability severely limits the size of the cyclestep l, just as with the standard forward-Euler algorithm. A more stable algorithm is to approximate the value of $x((m+l)T)$ by

$$x((m+l)T) - x(mT) \approx l\phi(x([m+l-1]T), [m+l-1]T, [m+l]T), \tag{4}$$

which is analogous to backward Euler for the differential case. This approach allows for larger cyclesteps than the forward-Euler-based approach, but leads to a more complicated equation to compute each cyclestep. To see this, consider computing $x(lT)$ given $x(0)$ based on (4). An $x((l-1)T)$ must be found such that when used as an initial condition for (1), the $x(lT)$ computed with standard discretization techniques satisfies $x(lT) - x(0) = l[x(lT) - x((l-1)T)]$. This is a boundary value problem, and is in general difficult to solve. For the case of switching power or filter circuits, the above boundary value problem can be solved efficiently using a Newton method, and this is presented in the next section.

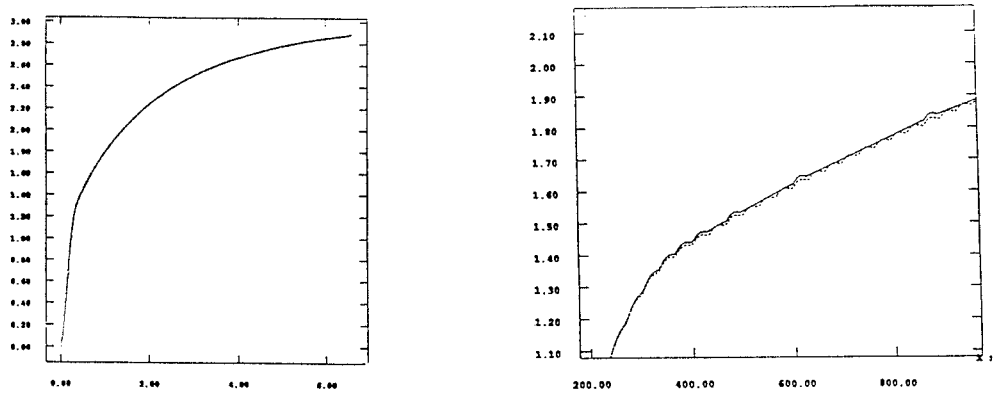

Figure 2: Transient Calculation and a Comparison of Classical and Envelope-Following Solutions

2.2 Solution by Newton

As mentioned in the previous section, each cyclestep of the backward-Euler envelope-following method requires the simultaneous solution of

$$x((m+l)T) - x(mT) - l[x((m+l)T) - x((m+l-1)T)] = 0 \tag{5}$$

and

$$x((m+l)T) - x((m+l-1)T) - \phi(x((m+l-1)T), (m+l-1)T, (m+l)T) = 0 \tag{6}$$

for $x((m+l-1)T)$ and $x((m+l)T)$, where $x(mT)$ is presumed known. Therefore, (5) and (6) represent a system of $2N$ equations in $2N$ unknowns, which we denote as $F(x((m+l)T), x((m+l-1)T)) = 0$.

An iterative Newton's method can be applied to solving the above system. In general, the Newton method applied to the problem of finding an $x \in \Re^N$ such that $F(x) = 0$, $F : \Re^N \rightarrow \Re^N$, yields the iteration equation $J_F(x^k)[x^{k+1} - x^k] = -F(x^k)$, where k is the Newton iteration count and $J_F \in \Re^{n \times n}$ is the Jacobian of F. In the case of the equation system represented by (5) and (6), $J_F(x((m+l)T), x((m+l-1)T))$ is given by

$$\begin{bmatrix} (1-l)I_N & lI_N \\ I_N & -I_N - \frac{\partial}{\partial x}\phi() \end{bmatrix} \tag{7}$$

where I_N is the identity matrix of size N.

The most time-consuming computation in this Newton iteration is evaluating J_F and F, which involves computing the state transition function, $\phi(x((m+l-1)T), (m+l-1)T, (m+l)T)$, and its derivative. The state transition function can be evaluated by numerically integrating (1) from $(m+l-1)T$ to $(m+l)T$ given $x((m+l-1)T)$. The derivative of the state transition function, referred to as the sensitivity matrix, represents the sensitivity of $x((m+l)T)$ to changes in $x((m+l-1)T)$, and can be computed with a small amount of additional work during the numerical integration [aprille72].

2.3 Implementation and Test Results

An envelope-following method has been implemented in the *Nitswit* [kundert88] simulation program. The program is written in "C", and runs under the UNIX operating system. The program uses a trapezoidal-rule-based envelope-following algorithm in which the cyclesteps are selected based on local truncation error. The boundary value problems generated at each cyclestep are solved with the shooting or Newton method described above, with a slight modification. Although switching power supplies and filters are not linear circuits, the state transition function over one cycle is nearly affine (linear plus a constant). This property can be exploited to reduce the computation by only computing J_F for the boundary value Newton algorithm on the first iteration at each timestep. This is a significant savings, as then the sensitivity matrix need only be computed once per cyclestep.

Exactly how the envelope-following method behaves can be seen by examining Fig. 2. The coarse dotted line is a portion of the solution to a buck-derived circuit (from [hsiao87]) computed with the classical method, and the fine dotted line is the result produced by the envelope-following method. As can be seen, the envelope-following method computes only occasional cycles, but the cycles computed are within a few percent of those computed with the classical method.

In the table below we present a comparison between the cpu time used by classical and envelope-following methods in simulating the start-up transient from three types of switching power supplies, a push-pull flyback converter, *fly*, a resonant converter, *res*, and a buck-derived circuit, *buck*. In each case, the clocking is provided by a user-defined source. As can be seen from the table, the envelope-following method can be very efficient, particularly when the simulation interval is long compared to the clock cycle.

Circuit	Nodes	Inteval/Clock	Classical	Env. Fol.
buck	8	1000	940	97
quasi	7	200	144	38
fly	32	40	274	167

Table 1: CPU Time (in seconds) Comparisons for Classical vs Envelope-Following Simulation, based on a SUN4

3 WR for Transient Device Simulation

For most applications, the lumped MOS models used in programs like SPICE accurately reflect the behavior of terminal currents and charges, but in some cases, these models are inadequate. For example, distributed effects in power MOS devices cannot be ignored when considering the efficiency of a transistor switch. For switching converter applications, sufficiently accurate transient simulations can be performed if, instead of using a lumped model for each transistor, the transistor terminal currents and charges are computed by solving the Poisson equation and the drift/diffusion partial differential equations governing charge transport in the device.

3.1 Device Simulation

In accordance with classical semiconductor device physics, a device is assumed to be governed by the Poisson equation, the electron and hole current-continuity equations, and the drift/diffusion approximations for current density:

$$\begin{aligned}
\frac{\epsilon kT}{q}\nabla^2 u + q\,(p - n + N_D - N_A) &= 0 \\
\nabla\cdot\vec{J}_n - q\left(\frac{\partial n}{\partial t} + R\right) &= 0 \quad \text{where} \quad \vec{J}_n = -qD_n\,(n\,\nabla u - \nabla n) \\
\nabla\cdot\vec{J}_p + q\left(\frac{\partial p}{\partial t} + R\right) &= 0 \quad \text{where} \quad \vec{J}_p = -qD_p\,(p\,\nabla u + \nabla p)
\end{aligned}$$

where u is the normalized electrostatic potential, n and p are the electron and hole concentrations, and $\vec{J}_n$ and $\vec{J}_p$ are the electron and hole current densities. Also, q is the magnitude of electronic charge, N_D and N_A are the donor and acceptor concentrations, D_n and D_p are the diffusion coefficients, ϵ is the permittivity, and R is the net generation/recombination rate. In the drift/diffusion equations, the diffusion constants and electron and hole mobilities are assumed to be related by the Einstein relations. Note that $\vec{J}_n$ and $\vec{J}_p$ are typically eliminated from the current continuity equations using the drift/diffusion equations, leaving a differential/algebraic system of three equations in three unknowns, u, n, and p.

Given a rectangular mesh that covers a two-dimensional slice of a MOSFET, a common approach to spatially discretizing the device equations is to use a finite-difference formula for the Poisson equation, and an exponentially-fit finite-difference formula for the current-continuity equations. The discretized Poisson equation at each mesh node i is

$$\epsilon\sum_j\left\{\frac{d_{ij}}{L_{ij}}(\psi_i - \psi_j)\right\} - qA_i\,(p_i - n_i + N) = 0$$

where the sum is taken over the four nodes adjacent to i (to the north, south, east, and west), d_{ij} is the distance from node i to node j, and L_{ij} is the length of the perpendicular bisector of the edge between nodes i and j. The discretized electron current continuity equation with the drift/diffusion approximation is

$$qD_n\sum_j\left\{\frac{d_{ij}}{L_{ij}}\left[n_j B(u_j - u_i) - n_i B(u_i - u_j)\right]\right\} - qA_i\left(\frac{dn_i}{dt} + R_i\right) = 0$$

where $B(u)$ is the Bernoulli function $B(u) = u/(e^u - 1)$ used to exponentially fit the potential variation to the electron concentration variation. The discretized hole current continuity equation with the drift/diffusion approximation is similar.

If there are N mesh points, then the result of applying the spatial discretization to the device equations is a sparse differential/algebraic system of $3N$ equations in as many unknowns. The Poisson equations form N algebraic constraints on the $2N$ nonlinear ordinary differential equations formed by the electron and hole current equations. About one thousand mesh points are typically needed to accurately represent a 2D slice of a MOS transistor, so that simulating a circuit where even a few transistors are treated by numerically solving the device equations leads to an enormous coupled system of algebraic and differential equations. In the next section we describe the WR algorithm for solving such a system of equations, and in Section 3.3 experimental results for 2D simulation are presented.

3.2 The WR Approach

The standard approach used to solve the differential/algebraic system generated by spatial discretization of the device equations is to discretize the d/dt terms with a low order integration method such as backward Euler. The result is a

sequence of nonlinear algebraic systems in $3N$ unknowns, each of which can be solved with some variant of Newton's method and/or relaxation[mayaram88]. Another approach is to apply relaxation directly to the differential/algebraic equation system with a WR algorithm[lelar82]. The relaxation equations for a point Gauss-Jacobi WR algorithm are:

$$\begin{aligned}
\frac{\epsilon kT}{q}\sum_j\left\{\frac{d_{ij}}{L_{ij}}(u_i^{k+1}-u_j^k)\right\}-qA_i\left(p_i^{k+1}-n_i^{k+1}+N_D-N_A\right) &= 0\\
qD_n\sum_j\left\{\frac{d_{ij}}{L_{ij}}\left[n_j^k B(u_j^k-u_i^{k+1})-n_i^{k+1}B(u_i^{k+1}-u_j^k)\right]\right\}-qA_i\left(\frac{dn_i^{k+1}}{dt}+R_i\right) &= 0\\
qD_p\sum_j\left\{\frac{d_{ij}}{L_{ij}}\left[p_i^{k+1}B(u_j^k-u_i^{k+1})-p_j^k B(u_i^{k+1}-u_j^k)\right]\right\}+qA_i\left(\frac{dp_i^{k+1}}{dt}+R_i\right) &= 0
\end{aligned}$$

The WR algorithm reduces the problem of simultaneously solving $2N$ differential equations and N algebraic equations to one of iteratively solving $3m$ independent equations. At each mesh node i, the equations governing the $u_i(t)$, $n_i(t)$, and $p_i(t)$ waveforms can be solved with a numerical integration method such as backward-Euler. The inherent advantage of the WR approach is that the differential equations are solved in a decomposed fashion, and therefore different sets of timesteps can be used at different mesh points to calculate the time evolution of u, n, and p. Therefore, if the device exhibits multi-rate behavior, WR can be very efficient provided it converges rapidly.

3.3 2D MOS Transistor Experiments

Previous theoretical results[lelar82, reichelt88] guarantee the convergence of the WR algorithm and suggest that the WR algorithm will converge in a uniform manner for the device problem. In order to verify that the uniformity of convergence that is essential to WR efficiency occurs in practice, a two-dimensional rectangular mesh, two-carrier, transient device simulation program was written and applied to a MOS device.

Experiments were conducted using a 2.2 μm n channel MOSFET. For this device, the drain and source p^+ doping is $N_a = 10^{20}cm^{-3}$, with an abrupt junction depth of 0.2 μm. The substrate doping is $N_a = 2.5\times10^{16}cm^{-3}$, and there is a channel implant of $N_a = 10^{16}cm^{-3}$ that extends to a depth of 0.05 μm. Oxide thickness is 0.05 μm.

During the simulation, Dirichlet boundary conditions were imposed by a gate contact and by ohmic contacts at the drain and source and along the bottom of the substrate. Neumann reflecting boundary conditions were imposed along the left and right edges of the region. The gate contact was held at $4v$, the source at $0v$, the substrate at $-3v$, and the drain was raised from $4v$ to $5v$ over a time equal to 1/10 of the simulation interval.

A rectangular mesh consisting of 23 horizontal lines and 31 vertical lines was imposed upon the device to spatially discretize the problem. The grid lines were placed closer together at points where u, n, and p were expected to exhibit rapid spatial variation, so that the mesh nodes were not evenly spaced. There were 550 silicon nodes and 101 oxide nodes, so that the total problem contained 1751 sparsely coupled algebraic and differential equations in as many unknowns.

method	CPU sec	% matrix
direct Newton/sparse GE	933.18	97.6%
natural WSOR	927.91	27.8%
red/black WSOR	970.33	27.5%

Table 1: Results for 23x31 Mesh, with 10 timepoints.

timestep (sec)	SOR iters
∞ (DC)	65
10^0	64
10^{-12}	62
10^{-13}	54
10^{-21}	53

Table 2: SOR iterations for different sized timesteps.

First, we compared the CPU time cost of solving this system by a direct method and by waveform relaxation. To solve directly, we discretized the system in time with the backward Euler method at ten uniformly spaced timepoints. This yielded a system of nonlinear algebraic equations at each timepoint, which were solved with Newton's method. Each linear equation system was solved with sparse Gaussian elimination. To solve with waveform relaxation, we also used ten uniformly spaced timepoints and the backward Euler method. We broke the problem into 31 blocks, defined by the vertical mesh lines. At each timepoint, the equations governing nodes on the same vertical mesh line were solved simultaneously using Newton's method and sparse Gaussian elimination. The blocks were processed in natural (left to right) order. After 10 Gauss-Seidel WR iterations over all blocks, waveform SOR was used to accelerate convergence, with a user specified relaxation parameter ω. As it is a more parallel algorithm, waveform SOR with red/black ordering of the vertical lines was also tested.

The results show that waveform SOR performed as well as direct methods. This is encouraging, because we did not exploit the multirate behavior that was found in the device. We expect that by using different timesteps for different blocks, we will lessen the WSOR CPU time substantially. Note that the red/black ordering didn't worsen the WSOR convergence rate. This implies that for this example, a factor of 15 speedup can be obtained by implementing the algorithm on a parallel machine.

Next, we investigated the effect of varying the time scale upon the convergence of SOR applied to the nonlinear algebraic problem at each timepoint. For an ordinary differential equation system, as the timestep h shrinks, the d/dt terms of the differential equations contribute more to diagonal dominance, and relaxation convergence is significantly improved. This may not be the case for a differential/algebraic system, as in our problem. The Poisson equation does not have a d/dt term, so decreasing h does not improve the diagonal dominance of the Poisson part of the Jacobians. In order to demonstrate that the Poisson equation dominates the SOR relaxation convergence at each timestep for the MOS transistor example, we compared the convergence of SOR on one timestep on several very different time scales.

Our $2.2\mu m$ device has a transit time on the order of 10 psec, so that 1 or 0.1 psec is a small enough timestep to capture transient device behavior. For timesteps much smaller than this, the Poisson equation dominates convergence, since the current equations become increasingly diagonally dominant and easier to solve. The results for large timesteps and for static solution confirm the Poisson dominance. Because the number of SOR iterations did not increase substantially, it appears that **the Poisson equations dominate SOR convergence at all time scales.**

Note also that the number of relaxation iterations (≈ 60) for SOR applied to a reasonably sized timestep (10^{-12} or 10^{-13} sec) is comparable to the number of WSOR iterations (77) required for convergence on a 10^{-11} second time interval. This indicates that in order for WSOR to have a speed advantage over SOR applied to each timestep, we will need to take advantage of the multirate behavior we observed in the device.

4 Conclusions and Acknowledgements

In the first part of this paper it is shown that envelope-following can substantially reduce the computation needed to simulate a switching power converter. Futher improvements to the method under consideration include exploiting the fact that most of the entries in the sensitivity matrix remain close to zero, and using optimized cycle boundary placement. In the second part of the paper we presented preliminary results that indicate the effectiveness of WR for 2D MOS transistor transient simulation. An appropriately blocked WR method is shown to be competitive with direct methods, and WR used a similar number of iterations as nonlinear relaxation applied at each timestep. This implies much promise for WR, as WR allows for multirate integration, is more suitable for parallel implementation, and can be made very efficient by refining the timesteps with iterations and using a single waveform-Newton iteration to solve the nonlinear WR iterations.

The authors would like to thank Professor Jon Allen and the students in the Custom Integrated Circuits group at MIT, and Steven Leeb at MIT's Laboratory for Electromagnetic and Electronic Systems for their assitance. This work was supported by the Defense Advanced Research Projects Agency contract N00014-87-K-825, the National Science Foundation contract MIP-8858764, and grants from Analog Devices, AT & T Bell Laboratories, and IBM.

References

[aprille72] T. Aprille, T. Trick, "Steady-state analysis of nonlinear circuits with periodic inputs." *Proc. IEEE*, Jan 72.

[hsiao87] C.J. Hsiao, R.B. Ridley, H. Naitoh, F.C. Lee, "Circuit-Oriented Discrete-Time Modeling and Simulation for Switching Converters." *IEEE Power Electronics Specialists' Conf. Rec.*, 1987

[kundert88] K. Kundert, J. White, A. Sangiovanni-Vincentelli, "An Envelope-Following Method for the Efficient Transient Simulation of Switching Power Converters." *Proc. Int. Conf. on Computer-Aided Design*, Santa Clara, California, October 1988.

[lelarasmee82] E. Lelarasmee, A. Ruehli, A. Sangiovanni-Vincentelli, "The Waveform Relaxation Method for the Time Domain Analysis of Large Scale Integrated Circuits." *IEEE Trans. on CAD*, Vol. 1, No. 3, July 1982.

[mayaram88] K. Mayaram, D.O. Pederson, "CODECS: A Mixed-Level Device and Circuit Simulator," *Proc. Int. Conf. on Computer-Aided Design*, Santa Clara, California, October 1988.

[nagel75] L.W. Nagel, "SPICE2: A Computer Program to Simulate Semiconductor Circuits", Electronic Research Laboratory *Report No. ERL-M520*, University of California, Berkeley, May 1975.

[petzold81] L. Petzold, "An Efficient Numerical Method for Highly Oscillatory Ordinary Differential Equations." *SIAM J. Numer. Anal.*, Vol. 18, No. 3, June 1981.

[pinto84] M.R. Pinto, C.S. Rafferty, and R.W. Dutton "PISCES-II: Poisson and continuity equation solver", Stanford Electronics Laboratory Technical Report, September 1984.

[reichelt88] M. Reichelt, J. White, J. Allen and F. Odeh, "Waveform Relaxation Applied to Transient Device Simulation," *1988 Int'l. Symp. on Circuits and Systems*, Espoo, Finland.

[selberherr84] S. Selberherr, *Analysis and Simulation of Semiconductor Devices*, Springer-Verlag, New York, 1984.