A Mixed Frequency-Time Approach for Finding the Steady-State Solution of Clocked Analog Circuits

K. Kundert, J. White, A. Sangiovanni-Vincentelli
Dept. of Electrical Engineering and Computer Science
Massachusetts Institute of Technology
Cambridge, MA 02139

Abstract

Performing detailed simulation of clocked analog circuits (e.g. switched capacitor filters and switching power supplies) with circuit simulation programs like SPICE is computationally very expensive. In this paper we present a new, more efficient, method for computing the detailed steady-state solution of clocked analog circuits. The method exploits the property of such circuits that the waveforms in each clock cycle are similar but not exact duplicates of the proceeding or following cycles. Therefore, by computing accurately a few selected cycles, the entire steady-state solution can be constructed efficiently.

1 Introduction

In general, analog circuit designers rely heavily on circuit simulation programs like SPICE [nagel75] or ASTAP [weeks73] to insure the correctness and the performance of their designs. These programs simulate a circuit by first constructing a system of differential equations that describes the circuit, and then solving the system numerically with a time discretization method such as backward-Euler. When applied to simulating clocked analog circuits, like the switched-capacitor filters used in integrated circuits or the switching converters used in high power applications, the classical circuit simulation algorithms become extraordinarily computationally expensive. This is because the period of the clock is usually orders of magnitude smaller than the time interval of interest to a designer. The nature of the calculations used in a circuit simulator implies that an accurate solution must be computed for every cycle of the clock in the interval of interest, and this can involve thousands of cycles.

In this paper we present another approach to the simulation of clocked analog circuits for the particular case of computing the steady-state solution. The method exploits the property of these circuits that node voltage waveforms over a given high frequency clock cycle are similar, but not exact duplicates, of the node voltages waveforms in proceeding or following cycles. This suggests that it is possible to construct a solution accurate over many high frequency clock cycles by calculating the solution accurately for a few selected cycles.

We begin, in the next section, by describing our assumptions about clocked analog circuits in steady-state and presenting a mixed frequency-time method. In section three we will discuss some of the computations involved in this method. In section four we examine the application of the frequency-time method to simulating switched-capacitor filters and present comparison results. Finally, in section five, we present our conclusions and acknowledgements.

2 The Mixed Frequency-Time Method

Very little can be assumed about behavior of the node voltage waveforms in a clocked analog circuit over a given clock cycle, because the circuits involved are very nonlinear and are usually switching rapidly. However, the node voltage waveforms over a whole clock cycle usually vary slowly from one cycle to the next, as controlled by the input signal. This implies that if the input is periodic, and the clocked analog circuit is in steady-state, then the sequence formed by sampling the node voltages at the beginning of each clock cycle is periodic (Fig. 1). We derive our method by assuming this to be true, and further assuming that the periodic function that describes the sequence of initial points in each clock cycle can be accurately represented as a truncated Fourier series using few terms.

If the sequence of initial points of each clock cycle can be described by a Fourier series with J terms, then once J initial points are known, all the initial points are known. This implies that given our Fourier assumption, to compute the steady state behavior of a clocked analog circuit we need only find the initial points of J clock cycles.

In the next two subsections we describe two relationships that can be exploited to construct a nonlinear algebraic system of J equations in J initial points (solving this system is discussed in section 3). The first relation, described in section 2.1, is derived from the Fourier series assumption, and is a linear relationship between the initial points of an evenly distributed set of J cycles and the initial points of the corresponding J cycles that immediately follow. The second relation is derived from solving the differential equation system that describes the analog circuit, for the time interval of one clock cycle, J times, each time using one of the distributed set of J initial points as an

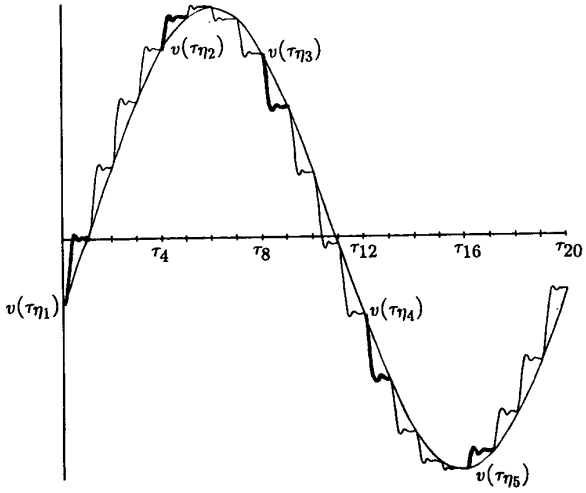

Figure 1: The response of a clocked analog circuit and the periodic function of the initial points. The J cycles used in the calculation are emphasized.

CH2584-1/88/0000-0027 $1.00 © 1988 IEEE

initial condition. This results in another set of values for the initial points of the following J cycles. Insisting that this set match the set resulting from the Fourier relation yields a nonlinear algebraic system in J unknowns, which can be solved for the J initial points, and this is described in section 2.2.

2.1 The Delay Operator

Consider the sequence of initial points of each clock cycle at some circuit node n, and denote the sequence by $v_n(\tau_1), v_n(\tau_2), \ldots, v_n(\tau_S)$ where S is the number of clock cycles in an input period. It is assumed that this sequence can be accurately approximated by a truncated Fourier series, and therefore

$$V_0 + \sum_{k=1}^{K} (V_k^C \cos k\omega\tau_s + V_k^S \sin k\omega\tau_s) = v_n(\tau_s), \tag{1}$$

where ω is the fundamental frequency of the input signal, K is the number of harmonics and $J = 2K + 1$ is the number of unknown coefficients. Given (1), there is a linear relation between any collection of J initial points and any other collection of J initial points. However, as mentioned above, we are most interested in the linear operator that maps a collection $v(\tau_{\eta_1}), \ldots, v(\tau_{\eta_J})$ into $v(\tau_{\eta_1} + T), \ldots, v(\tau_{\eta_J} + T)$ where T is the clock period and $\{\eta_1, \ldots, \eta_J\}$ is a subset of $\{1, \ldots, S\}$. This linear operator will be referred to as the delay matrix.

Deriving the delay matrix is a two stage process. First, the J points, $v(\tau_{\eta_1}), \ldots, v(\tau_{\eta_J})$ are used to calculate the Fourier coefficients. Then the Fourier series (using these coefficients) is evaluated at the J times, $\tau_{\eta_1} + T, \ldots, \tau_{\eta_J} + T$. The Fourier coefficients are then eliminated to yield the desired direct relation. To compute the Fourier coefficients, write (1) as a system of J linear equation in J unknowns [kundert88a],

$$\Gamma^{-1} \begin{bmatrix} V_0 \\ V_1^R \\ V_1^S \\ \vdots \\ V_K^R \\ V_K^S \end{bmatrix} = \begin{bmatrix} v_n(\tau_{\eta_1}) \\ v_n(\tau_{\eta_2}) \\ v_n(\tau_{\eta_3}) \\ \vdots \\ v_n(\tau_{\eta_J}) \end{bmatrix} \tag{2}$$

where $\Gamma^{-1} \in \Re^{J \times J}$ is given by

$$\begin{bmatrix} 1 & \cos\omega\tau_{\eta_1} & \sin\omega\tau_{\eta_1} & \cdots & \cos K\omega\tau_{\eta_1} & \sin K\omega\tau_{\eta_1} \\ 1 & \cos\omega\tau_{\eta_2} & \sin\omega\tau_{\eta_2} & \cdots & \cos K\omega\tau_{\eta_2} & \sin K\omega\tau_{\eta_2} \\ 1 & \cos\omega\tau_{\eta_3} & \sin\omega\tau_{\eta_3} & \cdots & \cos K\omega\tau_{\eta_3} & \sin K\omega\tau_{\eta_3} \\ \vdots & \vdots & \vdots & & \vdots & \vdots \\ 1 & \cos\omega\tau_{\eta_J} & \sin\omega\tau_{\eta_J} & \cdots & \cos K\omega\tau_{\eta_J} & \sin K\omega\tau_{\eta_J} \end{bmatrix}. \tag{3}$$

The matrix Γ^{-1} maps the Fourier coefficients to a sequence and is referred to as the inverse discrete Fourier transform. If the times $\tau_{\eta_1}, \ldots, \tau_{\eta_J}$ are reasonably evenly distributed over one period of the input signal, then Γ^{-1} is invertible. Its inverse, the forward discrete Fourier transform, is denoted by Γ. We can also write

$$\Gamma^{-1}(T) \begin{bmatrix} V_0 \\ V_1^R \\ V_1^S \\ \vdots \\ V_K^R \\ V_K^S \end{bmatrix} = \begin{bmatrix} v_n(\tau_{\eta_1} + T) \\ v_n(\tau_{\eta_2} + T) \\ v_n(\tau_{\eta_3} + T) \\ \vdots \\ v_n(\tau_{\eta_J} + T) \end{bmatrix}, \tag{4}$$

where $\Gamma^{-1}(T) \in \Re^{J \times J}$ is given by

$$\begin{bmatrix} 1 & \cos\omega(\tau_{\eta_1} + T) & \sin\omega(\tau_{\eta_1} + T) & \cdots & \sin K\omega(\tau_{\eta_1} + T) \\ 1 & \cos\omega(\tau_{\eta_2} + T) & \sin\omega(\tau_{\eta_2} + T) & \cdots & \sin K\omega(\tau_{\eta_2} + T) \\ 1 & \cos\omega(\tau_{\eta_3} + T) & \sin\omega(\tau_{\eta_3} + T) & \cdots & \sin K\omega(\tau_{\eta_3} + T) \\ \vdots & \vdots & \vdots & & \vdots \\ 1 & \cos\omega(\tau_{\eta_J} + T) & \sin\omega(\tau_{\eta_J} + T) & \cdots & \sin K\omega(\tau_{\eta_J} + T) \end{bmatrix}. \tag{5}$$

Given a sequence, a delayed version is computed by applying Γ to the sequence to compute the Fourier coefficients, and then multipling the vector of coefficients by $\Gamma^{-1}(T)$.

$$\begin{bmatrix} v_n(\tau_{\eta_1} + T) \\ v_n(\tau_{\eta_2} + T) \\ \vdots \\ v_n(\tau_{\eta_J} + T) \end{bmatrix} = \Gamma^{-1}(T)\Gamma \begin{bmatrix} v_n(\tau_{\eta_1}) \\ v_n(\tau_{\eta_2}) \\ \vdots \\ v_n(\tau_{\eta_J}) \end{bmatrix}. \tag{6}$$

Thus, the delay matrix, $\mathcal{D}(T) \in \Re^{J \times J}$, is defined as

$$\mathcal{D}(T) = \Gamma^{-1}(T)\Gamma. \tag{7}$$

As the delay matrix is a function only of ω, K, $\{\tau_{\eta_1}, \ldots, \tau_{\eta_J}\}$ and T, it can be computed once and used for every node.

2.2 The Differential Equation Relation

We assume that any clocked analog circuit to be simulated can be described by a system of differential equations of the form

$$\frac{d}{dt} q(v(t), u(t)) + i(v(t), u(t)) = 0, \tag{8}$$

where $v(t) \in \Re^N$ is the vector of node voltages, $u(t) \in \Re^M$ is the vector of input sources, $q(v(t), u(t)) \in \Re^N$ is the vector of sums of charges entering each node, and $i(v(t), u(t)) \in \Re^N$ is the vector of sums of currents entering each node. If the node voltages are known at some time t_0, it is possible to solve (8) and compute the node voltages at some later time t_1. In general, one can write

$$v(t_1) = \phi(v(t_0), t_0, t_1) \tag{9}$$

where ϕ is referred to as the state transition function for the differential equation and can be expanded as

$$\phi(v(t_0), t_0, t_1) = \begin{bmatrix} \phi_1(v(t_0), t_0, t_1) \\ \vdots \\ \phi_N(v(t_0), t_0, t_1) \end{bmatrix} \tag{10}$$

where $\phi_n : \Re^{N \times 1 \times 1} \to \Re$ for all $n \in \{1, \ldots, N\}$.

Now reconsider the J initial points at some circuit node n, $v_n(\tau_{\eta_1})$, $\ldots, v_n(\tau_{\eta_J})$. For each $j \in \{1, \ldots, J\}$ and each $n \in \{1, \ldots, N\}$ we can write

$$v_n(\tau_{\eta_j} + T) = \boldsymbol{\phi_n}(\boldsymbol{v}(\tau_{\eta_j}), \tau_{\eta_j}, \tau_{\eta_j} + T) \tag{11}$$

where T is the clock period. Note that $v_n(\tau_{\eta_j} + T)$ is the initial point of the cycle immediately following the cycle beginning at τ_{η_j}. Also, the node voltages at τ_{η_j} can be related to the node voltages at $\tau_{\eta_j} + T$ by the delay matrix, $\mathcal{D}(T)$. That is,

$$\mathcal{D}(T) \begin{bmatrix} v_n(\tau_{\eta_1}) \\ \vdots \\ v_n(\tau_{\eta_J}) \end{bmatrix} = \begin{bmatrix} v_n(\tau_{\eta_1} + T) \\ \vdots \\ v_n(\tau_{\eta_J} + T) \end{bmatrix}. \tag{12}$$

It is possible to use (11) to eliminate the $v_n(\tau_{\eta_j} + T)$ terms from (12), which yields

$$\mathcal{D}(T) \begin{bmatrix} v_n(\tau_{\eta_1}) \\ \vdots \\ v_n(\tau_{\eta_J}) \end{bmatrix} = \begin{bmatrix} \phi_n(v(\tau_{\eta_1}), \tau_{\eta_1}, \tau_{\eta_1} + T) \\ \vdots \\ \phi_n(v(\tau_{\eta_J}), \tau_{\eta_J}, \tau_{\eta_J} + T) \end{bmatrix} \tag{13}$$

for each $n \in \{1, \ldots, N\}$.

3 Solution by Newton-Raphson

The collection of equations given in (13) can be reorganized into a system of NJ equations in NJ unknowns as

$$F\left(\begin{array}{c} v(\tau_{\eta_1}) \\ \vdots \\ v(\tau_{\eta_J}) \end{array}\right) = \mathcal{D}_N(T)\left[\begin{array}{c} v_1(\tau_{\eta_1}) \\ \vdots \\ v_N(\tau_{\eta_1}) \\ \vdots \\ v_1(\tau_{\eta_J}) \\ \vdots \\ v_N(\tau_{\eta_J}) \end{array}\right] - \left[\begin{array}{c} \phi_1(v(\tau_{\eta_1}), \tau_{\eta_1}, \tau_{\eta_1}+T) \\ \vdots \\ \phi_N(v(\tau_{\eta_1}), \tau_{\eta_1}, \tau_{\eta_1}+T) \\ \vdots \\ \phi_1(v(\tau_{\eta_J}), \tau_{\eta_J}, \tau_{\eta_J}+T) \\ \vdots \\ \phi_N(v(\tau_{\eta_J}), \tau_{\eta_J}, \tau_{\eta_J}+T) \end{array}\right] \tag{14}$$

and

$$F\left(\begin{array}{c} v(\tau_{\eta_1}) \\ \vdots \\ v(\tau_{\eta_J}) \end{array}\right) = 0, \tag{15}$$

where $F : \Re^{NJ} \rightarrow \Re^{NJ}$, and $\mathcal{D}_N \in \Re^{NJ \times NJ}$ is given by

$$\mathcal{D}_N(T) = \left[\begin{array}{ccc} d_{11}I_N & \dots & d_{1J}I_N \\ \vdots & & \vdots \\ d_{J1}I_N & \dots & d_{JJ}I_N \end{array}\right] \tag{16}$$

where $d_{ij} \in \Re$ is the ij^{th} element of the delay matrix $\mathcal{D}(T)$, and $I_N \in \Re^N$ is the identity matrix.

Applying Newton's method to (14) leads to the iteration equation

$$J_F\left(\begin{array}{c} v^{(l)}(\tau_{\eta_1}) \\ \vdots \\ v^{(l)}(\tau_{\eta_J}) \end{array}\right)\left[\begin{array}{c} v^{(l+1)}(\tau_{\eta_1}) - v^{(l)}(\tau_{\eta_1}) \\ \vdots \\ v^{(l+1)}(\tau_{\eta_J}) - v^{(l)}(\tau_{\eta_J}) \end{array}\right] = -F\left(\begin{array}{c} v^{(l)}(\tau_{\eta_1}) \\ \vdots \\ v^{(l)}(\tau_{\eta_J}) \end{array}\right) \tag{17}$$

where l is the iteration number and $J_F \in \Re^{NJ \times NJ}$ is the Frechet derivative of F given by

$$\mathcal{D}_N(T) - \text{diag}\left(\frac{\partial\phi(v(\tau_{\eta_1}), \tau_{\eta_1}, \tau_{\eta_1}+T)}{\partial v(t_{\eta_1})}, \dots, \frac{\partial\phi(v(\tau_{\eta_J}), \tau_{\eta_J}, \tau_{\eta_J}+T)}{\partial v(t_{\eta_J})}\right). \tag{18}$$

There are two important pieces to the computation of one Newton iteration. Factoring the matrix J_F, which is sparse, and evaluating J_F and F, which involves computing the state transition function, $\phi(v(\tau_{\eta_j}), \tau_{\eta_j}, \tau_{\eta_j}+T)$, and its derivative for each $j \in \{1, \dots, J\}$. The state transition functions can be evaluated by numerically integrating (8) over the J periods. The derivatives of the state transition functions, referred to as the sensitivity matrices, can be computed with a small amount of additional work during the numerical integration.

To show how the computation of the state transition function and its derivative fit together, consider numerically integrating (8) with backward-Euler, which we chose for simplicity and because it appears to be one of the best formulas for clocked analog circuits. Given some initial time t_0 and some initial condition, $v(t_0)$, applying backward-Euler to (8) results in the following algebraic equation,

$$f(v(t_0+h), v(t_0)) = \frac{1}{h}(q(v(t_0+h)) - q(v(t_0))) + i(v(t_0+h)) = 0 \tag{19}$$

where $h \in \Re$ is the timestep. Note we have dropped explicitly denoting the dependence of q and i on the input u for simplicity.

Equation (19) is usually solved with Newton-Raphson, for which the iteration equation is

$$J_f(v^{(l)}(t_0+h))(v^{(l+1)}(t_0+h) - v^{(l)}(t_0+h)) = -f(v^{(l)}(t_0+h), v^{(l)}(t_0)) \tag{20}$$

where $J_f(v(t)) \in \Re^{N \times N}$ is the Frechet derivative of the nonlinear equation in (19) and is given by

$$J_f(v(t)) = \frac{\partial f(v(t), \cdot)}{\partial v(t)} = \frac{1}{h}\frac{\partial q(v(t))}{\partial v(t)} + \frac{\partial i(v(t))}{\partial v(t)}. \tag{21}$$

Solving (19) yields an approximation to $v(t_0+h) = \phi(v(t_0), t_0, t_0+h)$. Implicitly differentiating (19) for $v(t_0+h)$ with respect to $v(t_0)$ yields

$$J_f(v(t_0+h))\frac{\partial v(t_0+h)}{\partial v(t_0)} = \frac{1}{h}\frac{\partial q(v(t_0))}{\partial v(t_0)}\frac{\partial v(t_0)}{\partial v(t_0)}. \tag{22}$$

Given a $v(t_0)$, (19) can be repeatedly applied to find $v(t_0+T) = \phi(v(t_0), t_0, t_0+T)$, and (22) can be repeatedly applied to find the sensitivity matrix $\partial v(t_0+T)/\partial v(t_0) = \partial\phi(v(t_0), t_0, t_0+T)/\partial v(t_0)$ [kundert88b]. Note that J_f is required in both (20) and (22), and thus the sensitivity matrix update can be made more efficient by factoring J_f once and using it for both computations. However, the sensitivity matrix is still expensive to compute, because it is an $N \times N$ full matrix. We return to this point at the end of section 4.

4 Switched-Capacitor Filter Simulation

It has been possible for several years to fabricate complex analog circuits like switched-capacitor (SC) filters on a single integrated circuit. As with any integrated circuit design, since the cost of fabrication is very high, it is desirable to first simulate extensively the design. Detailed simulation of such circuits with SPICE is too computationally expensive, and therefore other, faster but less accurate, approaches are used. However, the mixed frequency-time approach described above can perform fast and accurate detailed simulation of SC filters, and is particularly efficient on this class of circuit for several reasons. First, SC filters are usually followed by a sampler, and so only the initial point of each cycle is needed. Second, the circuits are designed so that the distortion present in the sequence of initial points is small, so if driven by a sinusoid, only a few harmonics are significant and only a few clock cycles need to be computed. Finally, the state transition function for an SC filter over a clock cycle is near linear, and therefore the Newton method in (17) converges in just a few iterations.

In this section we describe our program *Nitswit*, a detailed simulator for SC filters. We start in the next section by describing previous work in SC filter simulation, and then in Section 4.2 we present our results.

4.1 Comparison to Previous Work

The most common approach to simulating an SC filter is first to break the circuit up into functional blocks such as operational amplifiers and switches. Each functional bock is simulated, using a traditional circuit simulator, for some short period. The simulations of the functional blocks are used to construct extremely simple macromodels, which replace the functional blocks in the circuit. The result is a much simplified circuit that can be simulated easily. This simplified circuit is then simulated for the thousands of clocks cycles necessary to construct a solution meaningful enough to verify the design.

Ad hoc simulators of this macromodeling sort have commonly been written by frustrated analog designers, but the techniques have also been formalized in programs like *Diana* [deman80] and *Switcap* [tsividis79]. Although these programs have served designers well, a macromodeling approach is only as good as the macromodel. If a second order effect in a functional block changes overall performance, but this effect is not included in the macromodel, the effect will never be seen in the simulation.

The simulators traditionally intended for use with SC filters, such as *Diana* and *Switcap*, also make the "slow-clock" approximation. After each clock transition, every node in the circuit is assumed to reach its equilibrium point before another transition occurs. This assumption, along with the use of algebraic macromodels, allow the filter to be treated as a discrete-time system with one time point per clock transition. A set of difference equations is then used to describe the filter.

The slow-clock approach suffers from several serious drawbacks. First, SC filters are being pushed to operate at ever higher frequencies, and the assumption that signals reach equilibrium between clock transitions is often violated. Also, since signals between clock transitions are not computed, it is possible to miss events that occur in these intervals that might interfere with proper and reliable operation (e.g., clock feed-through spikes causing an operational amplifier to saturate). Lastly, it is not possible to capture the effects of dynamic distortion processes, such as the important effect of the channel conductance on charge redistribution when a transistor switch turns off.

The mixed frequency-time approach can accelerate the detailed simulation of SC filters without resorting to macromodeling or the slow-clock assumption. Thus, it does not suffer from the limitations detailed above. It also does not require a large investment in macromodeling and is suitable for use with automated circuit extractors. Since our approach finds the steady-state solution directly and performs a circuit-level simulation, it is capable of accurately predicting distortion performance. Though not specifically described in this paper, this mixed frequency-time approach can also be used when the input consists of the sum of two periodic signals at unrelated frequencies. Thus, the intermodulation distortion can be directly computed, which is particularly useful for bandpass filters. Also, the fact that steady-state is computed directly, implies an additional advantage over transient methods when high-Q filters are simulated. One final point, the mixed frequency-time method can also be adapted to the macromodeling approach used in other SC filter simulators, accelerating those methods as well when the steady-state solution is desired.

4.2 Results

Nitswit is a C program that uses the algorithms presented in this paper to simulate SC filters. It contains two algorithms capable of finding the steady-state response of a circuit. The first is simply a transient analysis that continues until any steady-state is achieved. The second, of course, is the mixed frequency-time algorithm. Coding both algorithms into the same simulator provides a fair evaluation of the mixed frequency-time approach.

Results for three circuits are given below. The first, *sclpf*, is an RC one-pole SC filter. The second, *scop*, is a one pole active CMOS low pass filter. The last, *frog*, is a five pole Chebyshev active CMOS leap frog filter with 0.1dB ripple. This circuit is driven with a 1MHz clock, has a 20kHz bandwidth, and is being driven with a 1kHz test signal to measure its distortion.

circuit			direct	mixed frequency-time		
name	nodes	cycles/ period	time (sec)	harmonics, cycles	Newton iterations	time (sec)
sclpf	2	33	24.5	3,7	3	4.3
scop	13	100	522	3,7	6	90
frog	77	1000	12,987	3,7	6	1228

Table 1. *Nitswit* results from a VAX 8650 running ULTRIX 2.0.

Examination of the results above indicate as much as an order of magnitude speed increase over traditional methods, but this is not as much as one would expect. Much of the CPU time for large circuits is spent calculating the dense sensitivity matrix and factoring the Jacobian in (18). It does turn out however, that almost all the entries of the sensitivity matrix are approximately zero, and this suggests significant speed improvements can be achieved by ignoring those terms. In addition, we expect to get improved performance by switching to relaxation techniques to solve (14). Preliminary experiments indicate the relaxation converges quickly and reliably, and is much faster than sparse LU factorization.

5 Conclusion

A new mixed frequency and time method for computing the steady-state solution of clocked analog circuits has been presented. The method works by computing the solution to the differential equation system associated with a circuit for only J clock cycles, where J is the number coefficients needed in the Fourier series to represent accurately the sequence of initial points in each clock cycle. Thus, this method is particularly efficient when the number of coefficients in the Fourier series is many fewer than the number of clock cycles in one input signal period.

We have shown the mixed frequency-time algorithm to be an efficient method for the detailed simulation of switched-capacitor filters. It also appears promising for use on other traditionally hard to simulate circuits like switching power supplies and phase-locked loops. Another important aspect of this algorithm is that, upon examination of (14), it is clear that the J integrations of the differential equation to compute the J ϕ's and their derivatives are independent. The other step, solving the sparse matrix problem in (17), seems, as mentioned above, to be very amenable to solution by relaxation. Therefore, the mixed frequency-time algorithm is extremely well suited to implementation on a parallel processor.

5.1 Acknowledgements

We would like to acknowledge the discussions with Professor A. R. Newton; the help of Robert Armstrong, Mark Richelt, and Hormoz Yaghutiel; and the support of Barbara Bratzel and Mary Glanville. In addition, we would like to thank Res Saleh, for bailing us out, again. This work was supported by the Defense Advanced Research Projects Agency contract N00014-87-K-825, Hewlett-Packard, and the California MICRO program.

References

[deman80] H. De Man, J. Rabaey, G. Arnout, J. Vandewalle. "Practical implementation of a general computer aided design technique for switched capacitor circuits." *IEEE Journal of Solid-State Circuits*, vol. SC-15, pp. 190-200, April 1980.

[kundert88a] K. S. Kundert, G. B. Sorkin, A. Sangiovanni-Vincentelli. "Applying harmonic balance to almost-periodic circuits." *IEEE Transactions on Microwave Theory and Techniques*, vol. MTT-36, February 1988.

[kundert88b] K. S. Kundert, A. Sangiovanni-Vincentelli, T. Sugawara. "Techniques for finding the periodic steady-state response of circuits." In *Analog Methods for Circuit Analysis and Diagnosis*. To be published by Marcel Dekker in 1988.

[nagel75] L. W. Nagel. *SPICE2: A Computer Program to Simulate Semiconductor Circuits*. Electronics Research Lab Report, ERL M520, Univ. of Calif., Berkeley, May 1975.

[tsividis79] Y. P. Tsividis. "Analysis of switched capacitor networks." *IEEE Transactions on Circuits and Systems*, vol. CAS-26, pp. 935-946, November 1979.

[weeks73] W. T. Weeks, A. J. Jimenez, G. W. Mahoney, D. Mehta, H. Quasemzadeh, T. R. Scott. "Algorithms for ASTAP — A Network Analysis Program." *IEEE Transactions on Circuit Theory*, pp. 628-634, Nov. 1973.