Verifying Analog Oscillator Circuits
Using Forward/Backward Abstraction Refinement

Goran Frehse
VERIMAG
2, av. de Vignate
38610 Gières, France
goran.frehse@imag.fr

Bruce H. Krogh, Rob A. Rutenbar
Carnegie Mellon University
5000 Forbes Ave
Pittsburgh, PA 15213, USA
{krogh | rutenbar}@ece.cmu.edu

Abstract

Properties of analog circuits can be verified formally by partitioning the continuous state space and applying hybrid system verification techniques to the resulting abstraction. To verify properties of oscillator circuits, cyclic invariants need to be computed. Methods based on forward reachability have proven to be inefficient and in some cases inadequate in constructing these invariant sets. In this paper we propose a novel approach combining forward- and backward-reachability while iteratively refining partitions at each step. The technique can yield dramatic memory and runtime reductions. We illustrate the effectiveness by verifying, for the first time, the limit cycle oscillation behavior of a third-order model of a differential VCO circuit.

1. Introduction

In contrast to today's highly automated methodologies for digital circuit design, analog circuit design remains expert-intensive. Extensive simulation experiments are required to evaluate analog circuit designs, such as high-frequency RF circuits with difficult periodic steady state responses. Unfortunately, simulation alone can never completely verify a circuit. Ideally, we would like to have formal verification tools similar to those available for digital design, which would make it possible to verify properties of analog circuit designs for entire sets of initial states and continuous ranges of parameters. Our aim is to develop such tools using recently developed methods to perform model checking for hybrid dynamic systems, that is, systems characterized by both continuous and discrete state variables.

Hybrid system verification is based on the construction of conservative abstractions that represent efficiently whole sets of state trajectories, rather than individual simulation traces [1]. These abstractions are typically constructed by partitioning the continuous state space and then performing forward reachability computations, connecting the regions of the state space that can be reached by trajectories beginning from some set of initial states. If the state-space partition is too coarse, the resulting overapproximation of the set of reachable states may be too conservative to verify the desired properties, in which case the partition is refined to compute a less conservative overapproximation.

To verify properties of cyclic behaviors, it is necessary to compute a cyclic invariant of the abstraction. That is, one must show that all behaviors starting from the set of initial states return to some subset of the set of initial states. When such an invariant is found, the designer can conclude that all behaviors of the system will remain within this set indefinitely, and critical properties such as bounds on cycle time and jitter can be computed [7]. In this paper we introduce a new abstraction refinement technique that makes it possible to construct invariants for the cyclic behaviors of oscillator circuits in cases where standard forward reachability fails.

Model checking of nonlinear analog circuits was first proposed in [10], where the continuous state space is discretized, and an abstract transition relation is computed for the finite, discrete model. Conventional model checking can be applied to this abstraction. This approach has been extended to verify timing properties of analog circuits in [8]. In [9], analog circuits were verified using the tool Check-Mate, which computes an abstract transition relation between user-defined regions of the state space using polyhedral enclosures of the continuous trajectories. The tool named d/dt computes reachability by discrete-time integration over polyhedral sets of states, applied to analog circuits in [3]. PHAVer, a relatively recent development of our group, is a formal verification tool that allows us to target more complex designs while retaining guarantees of mathematical soundness [6].

Early attempts at sound verification of hybrid systems were ill-fated due to implementation issues, and some of

the tools have been in relaxed correctness to gain efficiency by accepting non-conservative approximations. While more efficient on a basic level, such approximating methods incur an overhead in dealing with numerical difficulties, either increasing the error or the chance of wrong results. The approach implemented in PHAVer is radically different: we compute with exact arithmetic and unbounded data structures, and employ conservative overapproximation to limit the complexity of the resulting objects.

The following section illustrates the use of forward reachability computations to characterized cyclic behaviors of two analog oscillator circuits. We show that the desired invariant set is computed successfully for the model of a second-order tunnel diode circuit, but forward reachability fails to find an invariant for a third-order VCO circuit. Section 3 provides an overview of how hybrid systems are modeled in PHAVer, and how it partitions the state space to overapproximate complex dyanmics and compute the set of reachable states. Section 4 describes a new method for refining the state space partition iteratively using forward- and backward reachability computations, and Sect. 5 presents the results of applying this method to compute an invariant successfully for the VCO circuit introduced in Sect. 2. The concluding section summarizes the contributions of this paper and describes other applications of the forward/backward abstraction refinement procedure.

2. Verification of oscillator circuits

Special techniques have been developed to simulate the periodic steady state behaviors of analog oscillator circuits, such as shooting methods and harmonic balance methods [11]. The aim of verification of oscillators is to evaluate properties of circuit behaviors in a neighborhood of the periodic steady state, starting from a set of initial conditions rather than from a single initial state. To accomplish verification using time-domain reachability computations, it is necessary to compute a set of state trajectories that returns to the set of initial states so that the reachability computation over one cycle characterizes the circuit behavior for all future time.

To illustrate the computation of a cyclic invariant set using forward reachability computations, we first consider the tunnel-diode oscillator (TDO) circuit shown in Fig. 1(a). With the inductor current I_L and diode voltage drop V_d as state variables, the second-order state equations for this circuit are given by

$$\dot{V}_d = 1/C(-I_d(V_d) + I_L), \tag{1}$$
$$\dot{I}_L = 1/L(-V_d - 1/G \cdot I_L + V_{in}), \tag{2}$$

where $C = 1\,pF$, $L = 1\,\mu H$, $G = 5\,m\Omega^{-1}$, $V_{in} = 0.3\,V$, and the diode current is given by a characteristic shown

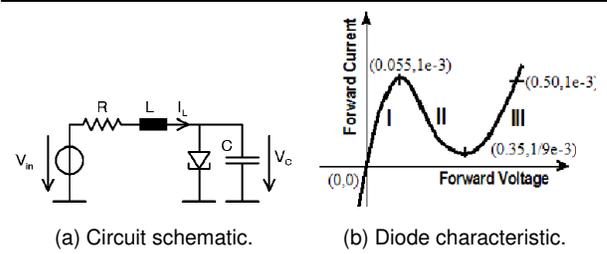(a) Circuit schematic.   (b) Diode characteristic.

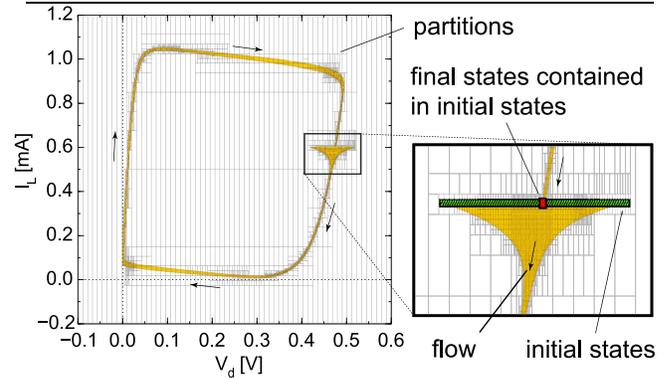**Figure 1. Tunnel diode oscillator circuit.**

Figure 2. Reachable states of the TDO.

in Fig. 1(b). To model this circuit in PHAVer, a piecewise affine envelope is constructed for the tunnel diode characteristic $I_d(V)$. We choose 64 intervals for the range $V_d \in [-0.1, 0.6]$ to yield sufficient accuracy and so obtain a piecewise affine model for (1).

Figure 2 shows the states reachable from a set of initial states given by $V_d \in [0.42V, 0.52V]$, $I_L = 0.6mA$. The vertical lines correspond to the 64 intervals of the affine diode characteristic, and the rest of the partitioning was generated during the analysis. It can be seen in Fig. 2 that the states reachable at the end of one cycle are contained in the set of initial states. The entire set of reachable states is therefore an invariant of the circuit, and, with some additional checks to exclude equilibria and local cycles, we can use this invariance to deduce properties of the oscillations. This circuit is simple and well-behaved enough to be analyzed with forward reachability. Reachability results for this circuit have also been obtained by Hartong et al. [10].

Next we consider a standard voltage controlled oscillator (VCO) circuit [4]. The circuit model shown in Fig. 3 was obtained under the following assumptions: an ideal current source I_b is biasing the VCO; the diodes function as capacitors; the substrate capacity is negligible; the circuit is perfectly symmetric; and the control voltage is constant. We use the Schichman-Hodges PMOS model [5], where the current $I_{DS}(V_{GS}, V_{DS})$ is given piecewise as follows:
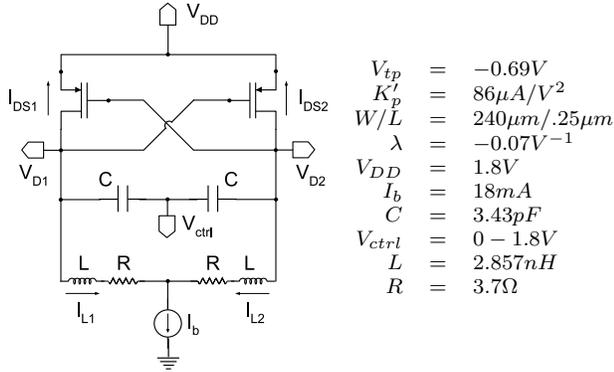
Figure 3. Differential VCO circuit.

$$
\begin{aligned}
V_{tp} &= -0.69V \\
K'_p &= 86\mu A/V^2 \\
W/L &= 240\mu m/.25\mu m \\
\lambda &= -0.07V^{-1} \\
V_{DD} &= 1.8V \\
I_b &= 18mA \\
C &= 3.43pF \\
V_{ctrl} &= 0 - 1.8V \\
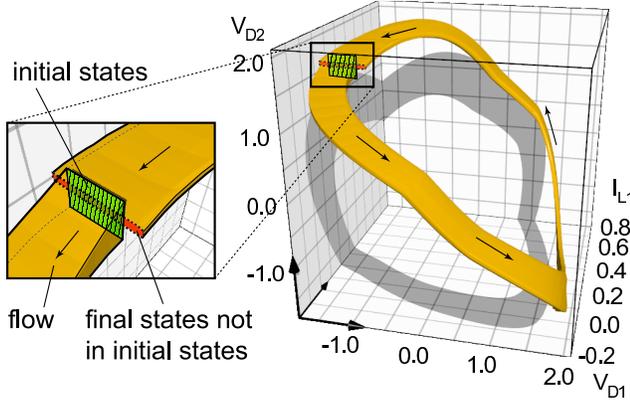L &= 2.857nH \\
R &= 3.7\Omega
\end{aligned}
$$

Figure 4. Reachable states in VCO

- $V_{GS} > V_{tp}$ (off): $I_{DS} = 0$

- $V_{GS} \leq V_{tp} \wedge V_{DG} > -V_{tp}$ (triode region):
  $I_{DS} = K'_P \frac{W}{L} \left[(V_{GS} - V_{TP})V_{DS} - \frac{1}{2}V_{DS}^2 \right] (1 - \lambda V_{DS})$

- $V_{GS} \leq V_{tp} \wedge V_{DG} \leq -V_{tp}$ (saturation):
  $I_{DS} = \frac{K'_P}{2} \frac{W}{L} (V_{GS} - V_{TP})^2 (1 - \lambda V_{DS})$

Using the algebraic constraint $I_{L2} = I_b - I_{L1}$ we obtain three state equations:

$$\dot{V}_{D1} = -\frac{1}{C}(I_{DS1}(V_{D2} - V_{DD}, V_{D1} - V_{DD}) + I_{L1}), \quad (3)$$

$$\dot{V}_{D2} = -\frac{1}{C}(I_{DS2}(V_{D1} - V_{DD}, V_{D2} - V_{DD}) + I_b - I_{L1}), \quad (4)$$

$$\dot{I}_{L1} = \frac{1}{2L}(V_{D1} - V_{D2} - R(2I_{L1} - I_b)). \quad (5)$$

Figure 4 shows the set of reachable states for the VCO circuit computed for initial states given by $V_{D1} \in [-1.4, -1.0]$, $V_{D2} \in [1.6, 1.9]$, and $I_{L1} = 0$. The limit cycle of the VCO is significantly less contractive than the TDO, so that the overapproximation introduced by our forward reachability algorithm is too large

to show that the states at the end of a cycle are contained in the initial states. In Sect. 4 we present a new alternative to forward reachability that successfully computes a cyclic invariant for this circuit.

3. Reachability analysis using PHAVer

In its core, PHAVer analyzes *linear hybrid automata*, which are characterized by linear inequalities defining transitions and all state sets, and conjuncts of constraints

$$a_i^T \dot{x} \bowtie_i b_i, \qquad a_i \in \mathbb{Z}^n, b_i \in \mathbb{Z}, \bowtie_i \in \{<, \leq\} \quad (6)$$

defining the dynamics. For this class of hybrid systems, the computation is exact algorithmically and, being purely state-based, ranges over infinite time. PHAVer uses polyhedra to represent sets of states, and exact arithmetic based on the Parma Polyhedra Library [2], which uses unbounded integer representations. If one is not careful in using exact arithmetic, the complexity of the linear predicates representing states typically increases prohibitively in the course of the analysis. This complexity is managed by enforcing user-controllable limits on the number of bits and constraints used per polyhedron. Polyhedra that exceed these limits are overapproximated conservatively. If the invariants of the automaton are bounded, limiting the number of bits forces the reachability analysis to terminate eventually, since only a finite number of predicates are possible.

PHAVer can compute an overapproximation of the set of reachable states for hybrid systems with affine dynamics that are specified in a relaxed form, i.e., as a conjunction of constraints of the form

$$a_i^T \dot{x} + \hat{a}_i^T x \bowtie_i b_i, a_i, \hat{a}_i \in \mathbb{Z}^n, b_i \in \mathbb{Z}, \bowtie_i \in \{<, \leq\}. \quad (7)$$

Using inequalities to describe the dynamics allows one to conservatively bound nonlinear dynamics, or to model bounded nondeterminism. The overapproximation of affine dynamics in PHAVer introduces a loss of accuracy that depends on the size of the location and the curvature of the vector field. To improve accuracy during reachability computations, locations are recursively split in two along a hyperplane, which is chosen for each split according to a set of criteria that aims at minimizing the number of partitions. The partitioning can be adapted to the dynamics by choosing the hyperplane that minimizes the angle spanned by the derivatives in the location, and stopping the splitting once a lower threshold Δ in the partition size is reached.

To illustrate the application of PHAVer, the results presented in Fig. 2 were generated as follows. The parameters of the algorithm were the direction of the splitting planes, a minimum and maximum size for the partitions, here $1/256$th and $1/16$th of the visible region. We stop the partitioning when the degree of widening of the vector falls below a certain threshold, which adapts the parti-
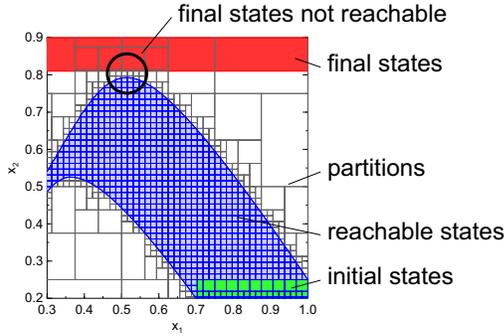
Figure 5. Forward reachability results in small, costly partitions in the entire reachable space for LIN1.

tions to the dynamics. To manage the complexity, the coefficients of polyhedra are limited to 24 bits, and the polyhedra to 32 constraints. The reachability analysis takes 72.8 s and 126.7 MB RAM (on a 2.8GHz Xeon with 4GB RAM under 32-bit Linux). The computation of the set of reachable states for the VCO shown in Fig. 4 was obtained with a minimum partition size of $1/128$th and takes 1567s and 941MB RAM. To obtain invariance the partition size needs to be $1/512$th, which could result in 64 times as many partitions, and thus is prohibitively expensive.

4. Forward/backward refinement

The verification of safety properties can be formulated as the question whether a particular set of undesired states, called *final states*, is reachable. In principle, a simple forward analysis can answer this question, but it may be unnecessarily costly. Even when the partitioning in a forward reachability analysis is efficient with respect to the flow of the vector field, it is obtained without regard to what part of the initial states actually leads to the final states. Here we introduce a way to use coarse partitions to quickly identify the regions where more refined partitions are required, and then create smaller partitions only in these regions. This new refinement procedure iterates between the computation of forward reachability from the initial states and backward reachability from the final states.

We say that a hybrid automaton H is *safe* if the set of reachable states $Reach(H)$ is disjoint from the final states S_F. Before we can introduce the algorithm, we need two operators: The *reverse* of H is the automaton H^{-1} obtained by reversing the transition relations, reversing the sign of the derivatives in all flow predicates, and swapping initial and final states. The *restriction* of H to a set of states R is the automaton $H|_R$ obtained by intersecting invariants and final states with R. The operators preserve safety as follows:

Proposition 4.1 H^{-1} *is safe if and only if H is safe. Given that $Reach(H) \subseteq R$, $H|_R$ is safe if and only if H is safe.*

We use $Reach_\Delta(H)$ to denote the conservative overapproximation of $Reach(H)$ that is computed by PHAVer using the partition size Δ. The forward/backward refinement (f/b-refinement) procedure is described by the following simple algorithm, which takes as input the automaton H and the parameters Δ_{min} and Δ_{max}, which represent the minimum and maximum level of overapproximation:

1. Initialize $\Delta = \Delta_{max}$.

2. Compute $R = Reach_\Delta(H)$.

3. If $R \cap S_F = \emptyset$ return *safe*; else if $\Delta > \Delta_{min}$, decrease Δ, set $H := (H|_R)^{-1}$, and go to 2; otherwise return *inconclusive*.

To illustrate the f/b-refinement algorithm, consider a system LIN1 with variables x_1, x_2, a flow determined by

$$\begin{pmatrix} \dot{x}_1 \\ \dot{x}_2 \end{pmatrix} = \begin{pmatrix} -4 & 2 \\ 6 & -4 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix},$$

initial states $0.7 \leq x_1 \leq 1 \wedge 0.2 \leq x_2 \leq 0.25$, and final states $x_2 \geq 0.81$. To show safety using forward reachability, a maximum partition size of $\Delta = 1/128$ in the directions of both axes is required, see Fig. 5.

We now show that LIN1 is safe using f/b-refinement. With an initial choice of $\Delta_{max} = \Delta_1 = 1/32$ we obtain the reachable set of states $R_1 = Reach_{\Delta_1}(H)$ of the first iteration shown in Fig. 6(a). The intersection of R_1 and the final states is not empty. As mandated by the algorithm we restrict H to R_1, which avoids adding irrelevant states in the backward reachability computation that follows. The backward computation is done by reversing the causality and time in the automaton, i.e., computing reachability for $H_2 = (H|_{R_1})^{-1}$, with a decreased partition size of $\Delta_2 = 1/64$. Figure 6(b) shows that the reachable states $R_2 = Reach_{\Delta_2}(H)$ cover a substantially smaller space than R_1, or even the actual $Reach(H)$. The intersection of R_2 and the final states of H_2, which were the initial states of H, is still not empty, but only a fraction of the initial states we started out with. Again, we restrict the automaton to the reachable states, reverse it and compute the reachable states of $H_3 = (H_2|_{R_2})^{-1}$, now with a partition size of $\Delta_3 = 1/128$. The resulting states $R_3 = Reach_{\Delta_3}(H_2)$, shown in Fig. 6(c), do not intersect with the final states, and therefore H_3 is safe. Recalling that safety with an overapproximation of the reachable states implies the safety of the actual automaton, we conclude that H is safe. F/b-refinement takes 5s and 11MB RAM compared to 28s and 45MB RAM for simple forward reachability (on a 1.9GHz P4m with 768MB RAM), thus awarding gains in speed and memory of more than a factor 4.

To be sound, f/b-refinement requires that a guaranteed overapproximation of the reachable states is computed,

(a) Iteration 1.

(b) Iteration 2.

(c) Iteration 3.
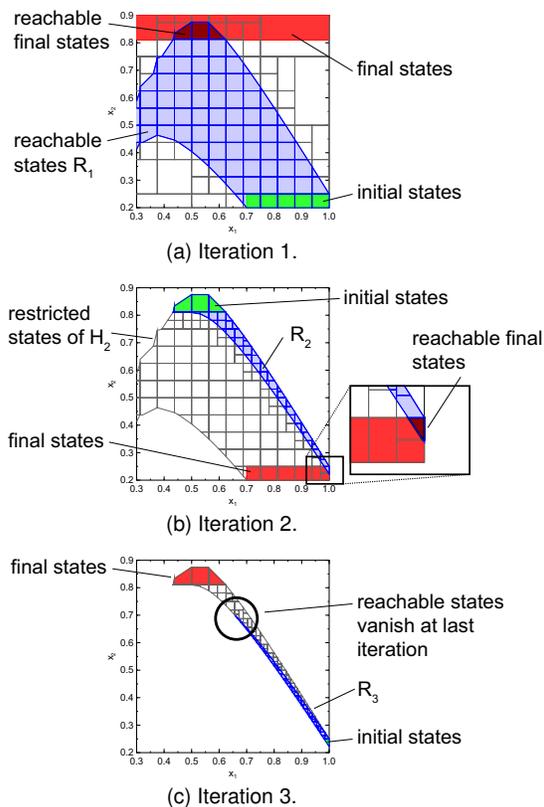
Figure 6. F/b-refinement of system LIN1.

which also implies that it must range over infinite time (unless time is inherently limited by the system). Both requirements are core characteristics of PHAVer that set it apart from most other tools currently available for hybrid systems. The algorithm has the useful property that overapproximation errors do not accumulate with more iterations. Also, overapproximations that are not invariant with respect to the forward- and backward-reachability computations are successively erased by the algorithm. This can be witnessed in practice, as "bumps" and edges that can arise through PHAVer's complexity reduction are smoothed away within a few iterations.

The scalability and benefit of f/b-refinement are subject of ongoing research. As the experimental results suggest, results vary. In the worst case, the partitioning is the same as for forward reachability, and since it is obtained in several iterations it is costlier in such a case.

5. Application of f/b-refinement

We show using f/b-refinement that the initial states of the VCO are mapped back onto themselves after one cycle, and compute an invariant for this cycle. We define a cross-section of the cycle at $I_{L1} = 0, V_{D1} \leq 0$, and de-

fine the final states to be the complement of the initial states on the cross-section. If the final states are not reachable, we can conclude that any cycle through the initial states passes the cross-section only at the initial states. The analysis shows this after 15 refinement iterations. Figure 7 shows the superimposed reachable states for the forward iterations, plus the states of the last backward iteration, which vanish halfway through the cycle. While the computation takes 11.5h and 1.7GB RAM, it can be carried out in parallel for different sets of states. E.g., showing that all states with $V_{D1} < -1.4$ are not reachable separately from showing it for $V_{D1} > -1.0$ succeeds in 5.7h on two processors using 1.2GB RAM each. This computation establishes a formal proof that the initial states contain a limit cycle, but it does not compute the limit cycle itself.

We compute the limit cycle by again applying f/b-refinement, this time defining the final states to be the initial states at the end of the cycle, i.e., after passing through a cross-section $I_{L1} = 0, V_{D1} > 0$. The states before and after passing through the cross section are distinguished by introducing different locations in the hybrid automaton, which are connected with transitions accordingly. At each iteration of the f/b-refinement, we intersect the states at the beginning and end of the cycle to further shrink the invariant towards the limit cycle. The refinement algorithm terminates when the minimum partition size is reached, and returns an efficiently partitioned set of reachable states that is guaranteed to be an invariant. Since the system is symmetric, we can alternatively use the initial states with V_{D1} and V_{D2} interchanged. Figure 8 shows an invariant computed with using both sets of initial states, which would have been by far too costly to compute just using forward reachability. The computation was performed with the same parameters (partition size, number of bits and constraints, derivative spread, etc.) as the forward reachability in Fig. 4 and took 2825s and 736MB RAM. For comparison, a rough estimate of the cost of a forward analysis can be obtained by extrapolating from the partition size necessary. The f/b-refinement terminated with a partition size of 1/512th for each of the 3 dimensions, i.e., $4^3 = 64$ times more partitions than the forward analysis in Sect. 3, which uses a size of 1/128th. Assuming that time and memory grow linear with the number of partitions, establishing invariance and computing the limit cycle using f/b-refinement consumes less than 44% of the estimated time and 2.8% of the memory of the forward analysis, not accounting for possible parallelization.

As another example, we analyzed the TDO circuit from Sect. 2 in parallel with a monitor automaton with a timer, thus being a 3-dimensional system [7]. We considered an uncertainty in the input voltage, bounded in amplitude by ± 0.1 V. Using forward reachability we obtain bounds on the cycle time of 12.6 to $15.3 \mu s$ in 999s with 1.5GB RAM.

Verifying the same bounds using f/b-refinement takes 1260s and 500MB RAM, i.e., it uses less than a third of the memory at a cost of 25% in speed. Since memory is usually the limiting factor in our experiments, this enables us to move on to more complex circuits and properties.

6. Conclusions

This paper presents a new method for verifying properties of analog oscillator circuits by computing overapproximations of the sets of possible state-space trajectories. In contrast to methods that use only forward reachability, refinement of the state space partitioning is carried out on iterations between forward and backward reachability. By focusing exclusively on the regions of the state space that need to be refined in each iteration, behavioral invariants are obtained more quickly and, in some cases, forward/backward iteration obtains invariants that are too costly to be computed by only forward reachability. The resulting set, which contains all periodic and quasi-periodic behaviors of the circuit, can be used to verify critical properties such as bounds on voltages, currents, cycle time (frequency), and jitter. These techniques can be extended to include parametric variations and can be used to analyze properties of time-bounded, non-cyclic behaviors.

Acknowledgments

References

[1] R. Alur, C. Courcoubetis, N. Halbwachs, T. A. Henzinger, P.-H. Ho, X. Nicollin, A. Olivero, J. Sifakis, and S. Yovine. The algorithmic analysis of hybrid systems. *Theoretical Computer Science*, 138(1):3–34, 1995.

[2] R. Bagnara, E. Ricci, E. Zaffanella, and P. M. Hill. Possibly not closed convex polyhedra and the Parma Polyhedra Library. In M. V. Hermenegildo and G. Puebla, editors, *Static Analysis: Proc. Int. Symp.*, volume 2477 of *LNCS*, pages 213–229. Springer, 2002.

[3] T. Dang, A. Donze, and O. Maler. Verification of analog and mixed-signal circuits using hybrid system techniques. In *FMCAD 2004, Austin, Texas*, 2004.

[4] B. De Muer and M. Steyaert. *CMOS Fractional-N Synthesizers*. Kluwer, 2003.

[5] D. A. Divekar. *Fet Modeling for Circuit Simulation*. Kluwer, 1988.

[6] G. Frehse. PHAVer: Algorithmic verification of hybrid systems past HyTech. In M. Morari and L. Thiele, editors, *Hybrid Systems: Computation and Control (HSCC'05), Mar. 9–*
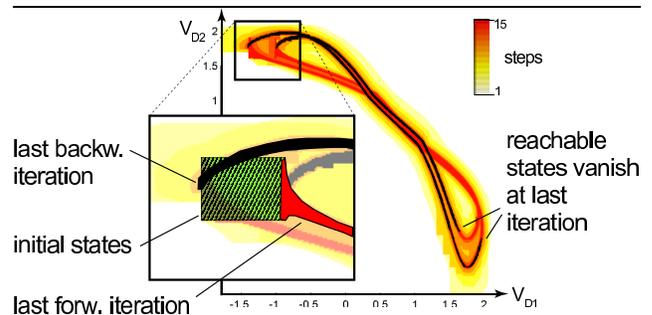
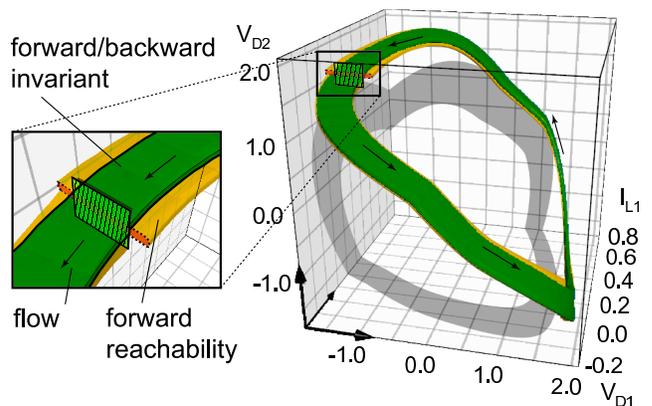**Figure 7. Reachable states during forward/backward refinement.**

Figure 8. Final invariant for the VCO circuit from forward/backward refinement.

11, 2005, Zürich, CH, 2005. PHAVer is available at `http://www.cs.ru.nl/~goranf/`.

[7] G. Frehse, B. H. Krogh, R. A. Rutenbar, and O. Maler. Time domain verification of oscillator circuit properties. In *Workshop on Formal verification of Analog Circuits (ETAPS Satellite Event), Edinburgh, Scotland, April 2-10*, 2005.

[8] D. Grabowski, D. Platte, L. Hedrich, and E. Barke. Time constrained verification of analog circuits using model-checking algorithms. In *Formal verification of Analog Circuits*, Edinburgh, Scotland, April 2005.

[9] S. Gupta, B. H. Krogh, and R. A. Rutenbar. Towards formal verification of analog designs. In *ICCAD 2004, San Jose, CA (USA)*, 2004.

[10] W. Hartong, L. Hedrich, and E. Barke. On discrete modeling and model checking for nonlinear analog systems. In E. Brinksma and K. G. Larsen, editors, *CAV*, volume 2404 of *LNCS*, pages 401–413. Springer, 2002.

[11] K. Kundert, J. White, and A. Sangiovanni-Vincentelli. *Steady-State Methods for Simulating Analog and Microwave Circuits*. Kluwer Academic Publishers, 1990.