

The Case for Analog Circuit Verification

Chris J. Myers and Reid R. Harrison^{1,2}

*Department of Electrical and Computer Engineering
University of Utah
Salt Lake City, USA*

David Walter, Nicholas Seegmiller, and Scott Little³

*School of Computing
University of Utah
Salt Lake City, USA*

Abstract

The traditional approach to validate analog circuits is to utilize extensive SPICE-level simulations. The main challenge of this approach is knowing when all important corner cases have been simulated. A new alternative is to utilize formal verification techniques. This paper utilizes a simple example to illustrate the potential flaws of a simulation-only based validation methodology and the potential benefits of formal verification of analog circuits.

Key words: Analog circuits, formal verification, hybrid Petri nets

1 Introduction

As integration densities continue to increase, chips are increasingly more likely to include analog circuitry. It is predicted that by 2006 that 75 percent of all chips will include some analog circuits [13]. While these circuits only make up about 2 percent of the devices and 20 percent of the area, they are taking 40 percent of the design effort [13]. It has also been cited that 50 percent of errors found in chips requiring a redesign are due to errors in the analog portion of the design [13]. Therefore, improvements in analog circuit validation methodology are becoming increasingly important.

Today, analog circuit validation takes the form of SPICE-level simulations with certain parameters (e.g., nMOS and pMOS threshold voltages) varied to

¹ This research is supported by SRC contract 2002-TJ-1024.

² Email: {myers,harrison}@ece.utah.edu

³ Email: {dwalter,seegmill,little}@cs.utah.edu

reflect their minimum, typical, and maximum expected values. Often, all permutations of these “corner” simulations are run for major model parameters, resulting in long simulation time.

While these simulations model extreme points resulting from global variation on the fabrication line, they fail to account for local transistor-to-transistor mismatch. The effects of this type of mismatch can be evaluated using Monte Carlo simulations in which the threshold voltage of *each* transistor in the circuit is assigned a random value drawn from a Gaussian distribution [17]. Then, many simulations are run with different “draws” from the distributions, and the variance of the simulation results are studied. This method results in long simulation times, and it is extremely unlikely that the “worst-case” mismatch scenario is stumbled across.

Until recently, the validation of digital circuitry also utilized this simulation-only based methodology. In recent years, however, there has been substantial success in the formal verification of digital circuits. Formal verification utilizes non-determinism and state space exploration methods to validate designs over a range of conditions during one execution.

To date, there has been relatively little research in the formal verification of analog and mixed-signal circuits. In analog and mixed-signal circuits, the state space often includes several continuous variables, such as voltages and currents, which must be tracked. These continuous variables make the state space extremely large and even more difficult to analyze than purely discrete systems. Formal verification of analog circuits, however, has the potential benefit of validating designs over a range of parameters and initial conditions all at once. Non-determinism in the models used for formal verification even allow conditions to change over the course of the execution. These techniques, therefore, provide a promising mechanism to validate designs in the face of noise and uncertain parameters.

Recently, researchers have begun to develop methods to formally verify analog circuits. Perhaps the first work in this area is from Kurshan and McMillan in which analog circuit models are translated to finite state models using homomorphic transformations [14]. Hedrich and Barke have a technique for equivalence checking of two analog circuits using a sampling of the state space [10]. In [11], this work is extended to take into account parameter variations. Hendrix and Claesen use a symbolic approach to verify properties in mixed-signal designs [12]. Ghosh and Vemuri prove the equivalence of piecewise-linear models of synthesized analog designs to their behavioral specifications utilizing automated theorem proving [6]. Salem has a technique to verify the equivalence of a specification and implementation both given in VHDL-AMS using rewriting rules for the analog portion of the design [16]. Hartong et. al verify analog circuits by dividing the continuous state space into regions that are represented in a Boolean manner which allows them to perform model checking using standard Boolean based approaches [8,9]. Gupta et. al utilize their hybrid system verification tool, **CheckMate**, to verify analog

circuits such as a tunnel diode oscillator and a delta-sigma modulator [7]. In [4], Dang et. al extend their hybrid system tool, **d/dt**, to verify a biquad low-pass filter. They also use a bounded verification technique to analyze the delta-sigma modulator example. Finally, in [15], we present a new efficient and conservative algorithm for reachability analysis of hybrid Petri net models. We utilized this tool to rapidly verify the tunnel diode oscillator example as well as find the reachable states of a model of a simple phase-locked loop.

The question that must be asked, however, is whether or not formal verification of analog circuits is necessary and/or useful. In other words, why is a simulation-only based methodology not sufficient? The goal of this paper is to try to answer this question via a simple analog circuit example. This paper argues that while simulation can find all the important behaviors, determining all the necessary simulation runs to find them can be a non-trivial exercise. This paper concludes with a brief discussion of the verification methodology that we are developing to address these problems.

2 Motivating Example

Delays in high-speed circuits are dominated by parasitic capacitances which can be nonlinear and voltage dependent. Variable interconnect length caused by automated place-and-route tools can lead to mismatched interconnect capacitances and thus variable delays. We examine the effect of these variable delays using a simple example shown in Fig. 1 of a discrete-time integrator responding to a 5kHz periodic square wave input, V_{in} . Integrators are a common analog component found in many circuits including delta-sigma modulators such as those that have been analyzed in [4,7].

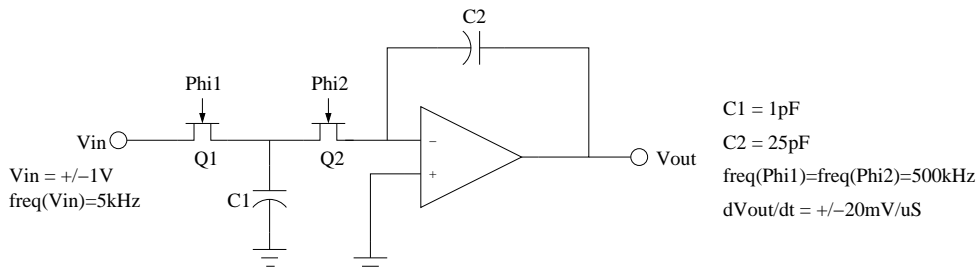


Fig. 1. A switched-capacitor integrator.

Discrete-time integrators typically utilize switched-capacitor circuits to accumulate charge. Capacitor mismatch can cause gain errors in integrators. Also, the CMOS switch elements in switched-capacitor circuits inject charge when they transition from closed to open. This charge injection is difficult to control with any precision, and its voltage-dependent nature leads to circuits that have a weak signal-dependent behavior. This can cause integrators to have slightly different gains depending on their current state and input value. Circuits using integrators, such as delta-sigma modulators, run the risk of the

integrator saturating near one of the power supply rails. It is essential to ensure that this never happens during operation under any possible permutation of component variations. Therefore, the verification property to check is whether or not the voltage V_{out} can rise above 2V or fall below -2V.

3 Timed Hybrid Petri Nets

To analyze analog circuits, one must first develop a model. Analog circuits are traditionally modeled using differential equations. Recently, hardware description languages such as VHDL-AMS [3] have been developed for modeling analog and mixed-signal circuits. These models are typically translated into a graphical representation for formal analysis. Such a representation must be capable of describing both discrete and continuous behavior. *Hybrid automata* (HA) are a common graphical representation that supports such behavior [1]. Recently, there have also been various extensions to Petri nets to incorporate continuous behavior (see for example [2,5]).

Given our past experience with timed Petri net models, we have chosen to utilize the *Timed Hybrid Petri Net* (THPN) model for verifying analog and mixed-signal circuits. We have developed methods for translating differential equation models into THPNs [15]. We are also developing a compiler from VHDL-AMS to THPN models. While THPN models are less expressive than HA models, we believe they are easier to analyze. The verification methods described in this paper, however, could be adapted to some restricted class of HA models, if so desired.

This section briefly describes THPNs by first introducing discrete Petri nets, followed by continuous Petri nets, and finally how they are composed to form THPNs. A discrete Petri net is a tuple $N_d = \langle P_d, T_d, F_d, m_0 \rangle$ such that:

- P_d : is a finite set of discrete places;
- T_d : is a finite set of discrete transitions;
- $F_d \subseteq (P_d \times T_d) \cup (T_d \times P_d)$ is the flow relation;
- $m_0 \subseteq P_d$ is the set of initially marked places.

Discrete places are depicted as circles, discrete transitions as solid boxes, and markings as solid circles (see the initially marked place pI and transition $Decl$ in the THPN for our motivating example shown in Fig. 2).

A continuous Petri net is a tuple $N_c = \langle P_c, T_c, F_c, R, X_0 \rangle$ such that:

- P_c : is a finite set of continuous places;
- T_c : is a finite set of continuous transitions;
- $F_c \subseteq (P_c \times T_c) \cup (T_c \times P_c)$ is the flow relation;
- $R : T_c \rightarrow \mathcal{Q}$ is the flow rate of transitions;
- $X_0 : P_c \rightarrow (\{-\infty\} \cup \mathcal{Q}) \times (\mathcal{Q} \cup \{\infty\})$ assigns a range of initial marking values to the continuous variables $(x_{0l}(p), x_{0u}(p))$.

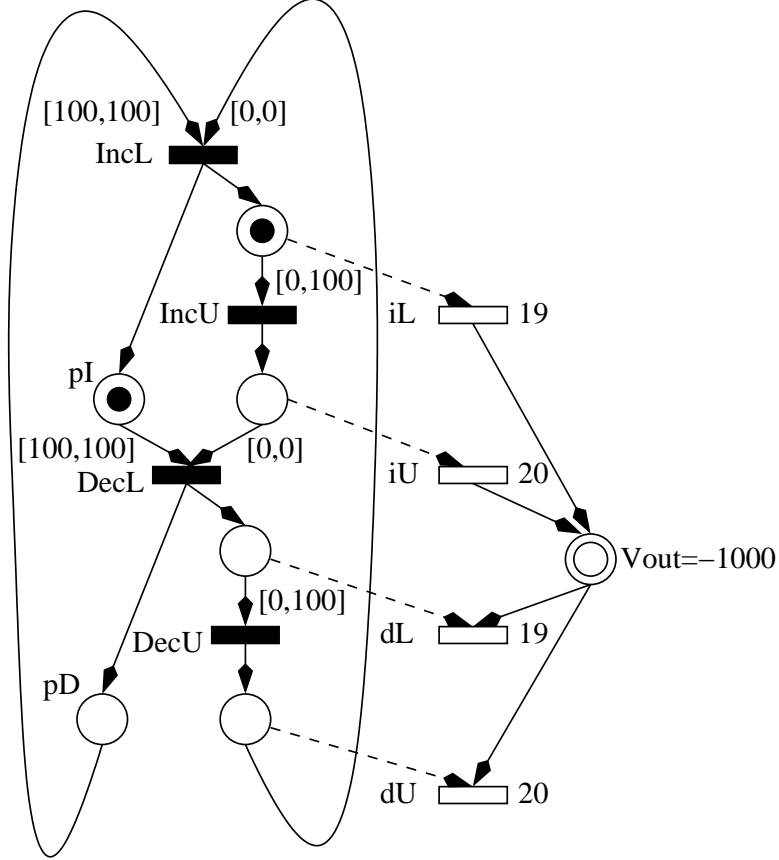


Fig. 2. THPN for our motivating example.

Note that \mathcal{Q} denotes rational numbers. Also, X_0 is a range allowing uncertainty in the initial value of the continuous variables. This is one source of non-determinism in the THPN model. Graphically, continuous places are depicted as double circles, continuous transitions are depicted as empty boxes, and each continuous transition is annotated with its flow rate (see continuous place V_{out} which is initially -1000mV and continuous transition iL with rate $19\text{mV}/\mu\text{S}$ in Fig. 2).

A THPN is defined as a tuple $N_h = \langle N_d, N_c, F_h, B, D, A \rangle$ such that:

N_d : is a discrete Petri net as defined previously;

N_c : is a continuous Petri net as defined previously;

$F_h \subseteq (P_c \times T_d) \cup (P_d \times T_c) \cup (T_d \times P_c)$ is the flow relation between the discrete Petri net and the continuous Petri net;

$B : (P_c \times T_d) \rightarrow (\{-\infty\} \cup \mathcal{Q}) \times (\mathcal{Q} \cup \{\infty\})$ assigns a predicate $[b_l(p, t), b_u(p, t)]$ to arcs from continuous places to discrete transitions where $b_l(p, t) \leq b_u(p, t)$;

$D : ((P_d \cup P_c) \times T_d) \rightarrow \mathcal{Q}^+ \times (\mathcal{Q}^+ \cup \{\infty\})$ assigns a delay $[d_l(p, t), d_u(p, t)]$ to each arc between a place and a discrete transition where $d_l(p, t) \leq d_u(p, t)$.

$A : (T_d \times P_c) \rightarrow (\{-\infty\} \cup \mathcal{Q}) \times (\mathcal{Q} \cup \{\infty\})$ specifies an assignment value $[a_l(t, p), a_u(t, p)]$ on arcs from discrete transitions to continuous places.

Note that \mathcal{Q}^+ are the non-negative rationals. Also, B , D , and A are only defined for arcs specified in the corresponding flow relation. Arcs added to the THPN by the flow relation F_h are denoted with dashed lines. An arc from a continuous place to a discrete transition is annotated with a predicate and a bounded delay assignment. Arcs from a discrete place to a discrete transition are also annotated with a bounded delay assignment. An arc from a discrete transition to a continuous place is annotated with a variable assignment. Arcs from continuous transitions to discrete places are not allowed.

The semantics of THPNs are described informally via the THPN for the integrator shown in Fig. 2. To keep the model simple, this THPN is constructed by hand rather than by our automated method. The input is assumed to be a square wave which is high for $100\mu\text{S}$ and low for $100\mu\text{S}$. Currently, our verification tool requires all rates to be constants. However, we would like to verify the integrator over a range of rates representing uncertainty in the capacitance values and mismatches in transistors. We assume that the integrator voltage V_{out} increases with a rate between 19 and $20\text{mV}/\mu\text{S}$ when the square wave is high, and decreases with a rate between 19 and $20\text{mV}/\mu\text{S}$ when it is low. This is modeled in a piecewise fashion. For example, in the initial state, V_{out} increases with a rate of $19\text{mV}/\mu\text{S}$. After a period of 0 to $100\mu\text{S}$, $IncU$ fires which causes V_{out} to begin to increase with a rate of $20\text{mV}/\mu\text{S}$. After $100\mu\text{S}$ of increase of the voltage V_{out} , the transition $DecL$ fires resulting in V_{out} beginning to decrease at a rate of $19\text{mV}/\mu\text{S}$ which it does for 0 to $100\mu\text{S}$. When $DecU$ fires, V_{out} begins to decrease with a rate of $20\text{mV}/\mu\text{S}$. Finally, after $100\mu\text{S}$ of decrease, $IncL$ fires returning it to the initial state.

4 Simulation Versus Verification

We simulated our THPN model for a variety of rates of integration. In Fig. 3, all rates are set to be $20\text{mV}/\mu\text{S}$. A triangle wave is produced which ranges from -1V to $+1\text{V}$. In Fig. 4, the rates are all changed to $19\text{mV}/\mu\text{S}$ to represent a different capacitance value. Again, a triangle wave is produced, but the range is now only from -1V to 0.9V .

These simulation results are consistent with the types of results that one typically obtains from deterministic SPICE simulations. However, it does not capture the true behavior as rates in the model can vary over time between 19 and $20\text{mV}/\mu\text{S}$. In order to address this, one may try to perform a random simulation. Fig. 5 shows the result from one such random simulation. While the range of the voltage V_{out} is certainly greater, the plots seem to indicate that it is still near -1V and $+1\text{V}$.

In our motivating example, divergent behavior occurs when there is a discrepancy in the rise and fall rates. Fig. 6 shows the result of a simulation in which the rising rates are $20\text{mV}/\mu\text{S}$ and falling rates are $19\text{mV}/\mu\text{S}$. This simulation shows that it is possible for V_{out} to saturate at its supply rail of 2.5V . While simulation can find this anomalous behavior, the question is whether or

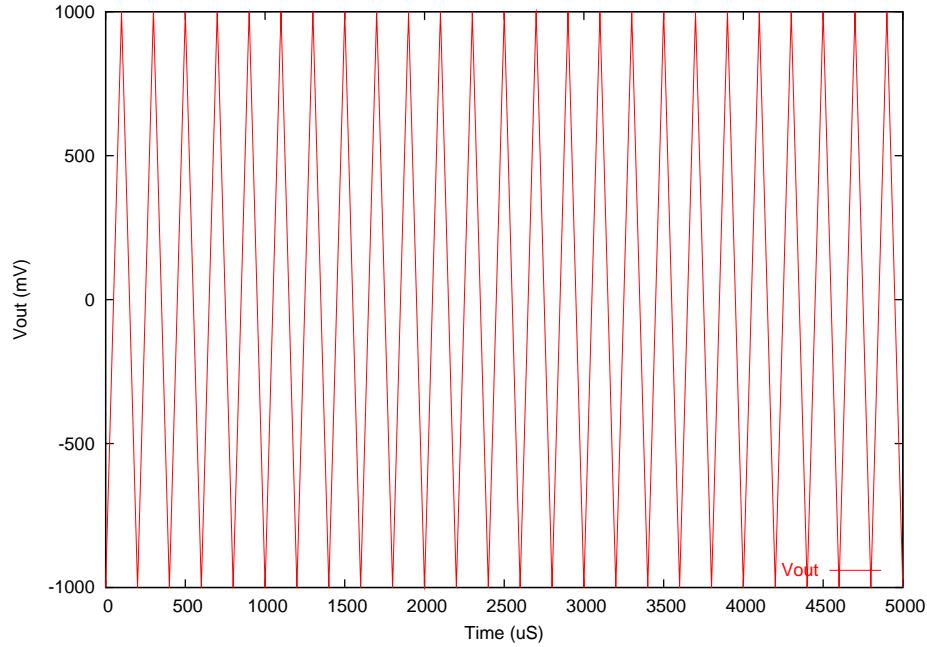


Fig. 3. All rates 20 (i.e., $R(iL)=R(iU)=R(dL)=R(dU)=20$).

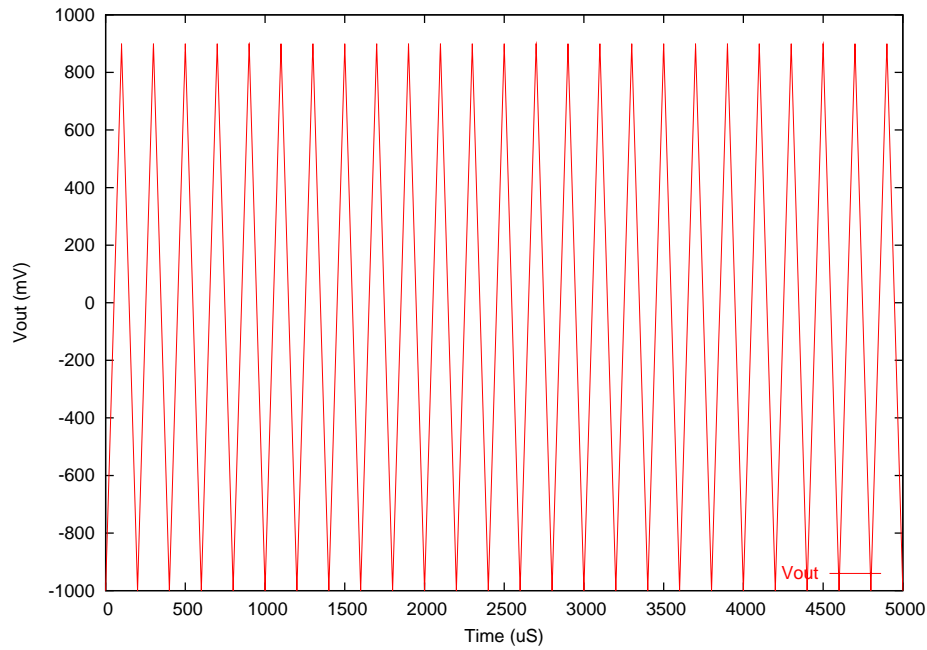


Fig. 4. All rates 19 (i.e., $R(iL)=R(iU)=R(dL)=R(dU)=19$).

not this is a simulation run that a designer would try. For this simple example, one would expect that an expert would. However, for a larger example that includes this integrator as a component, it is not so clear that this problem would be detected using a simulation-only validation methodology especially if the designer is not an expert.

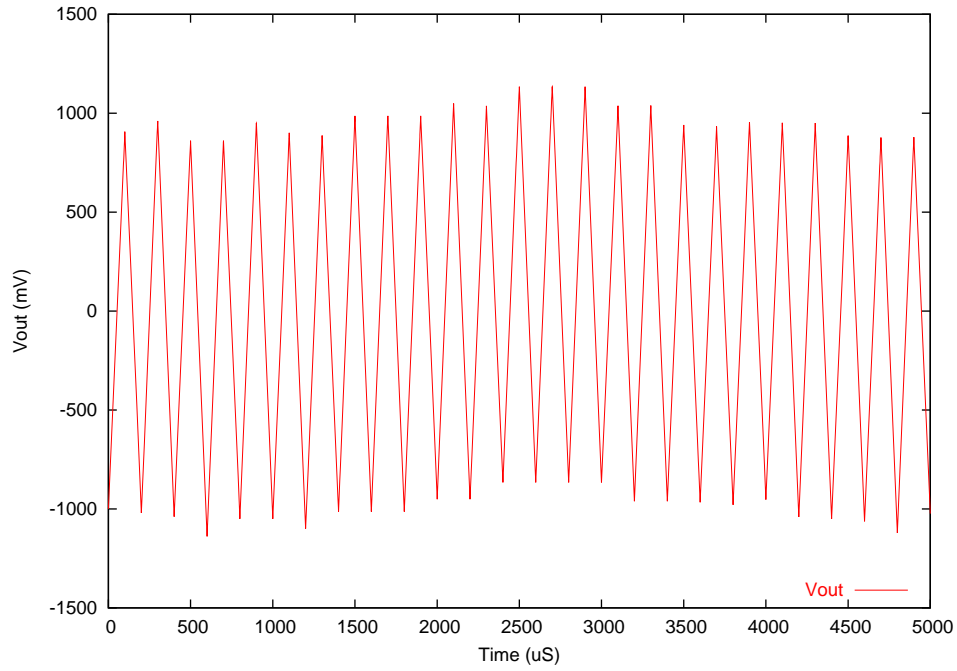


Fig. 5. Random delay (i.e., $R(iL)=19, R(iU)=20, R(dL)=19, R(dU)=20$).

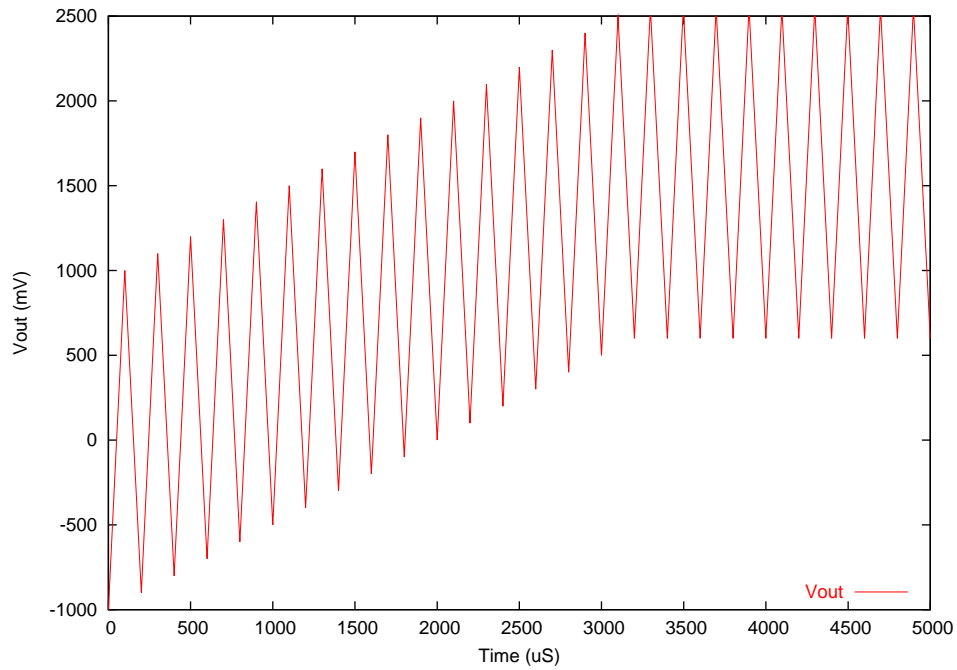


Fig. 6. Differing rise and fall delay (i.e., $R(iL)=R(iU)=20, R(dL)=R(dU)=19$). Note that the range of V_{out} is restricted in the simulation to fall within $-2.5V$ and $2.5V$.

In order to verify this example, we add to the THPN transitions *Fail1* and *Fail2* as shown in Fig. 7. These transitions remove tokens from places *pI* or *pD* when *Vout* exceeds 2V or goes below -2V causing the model to deadlock. We then apply the efficient state space reachability algorithm for THPNs described in [15]. The result is that a deadlock is detected in 0.05 seconds after exploring 198 timed states on a 1.7GHz PentiumM computer.

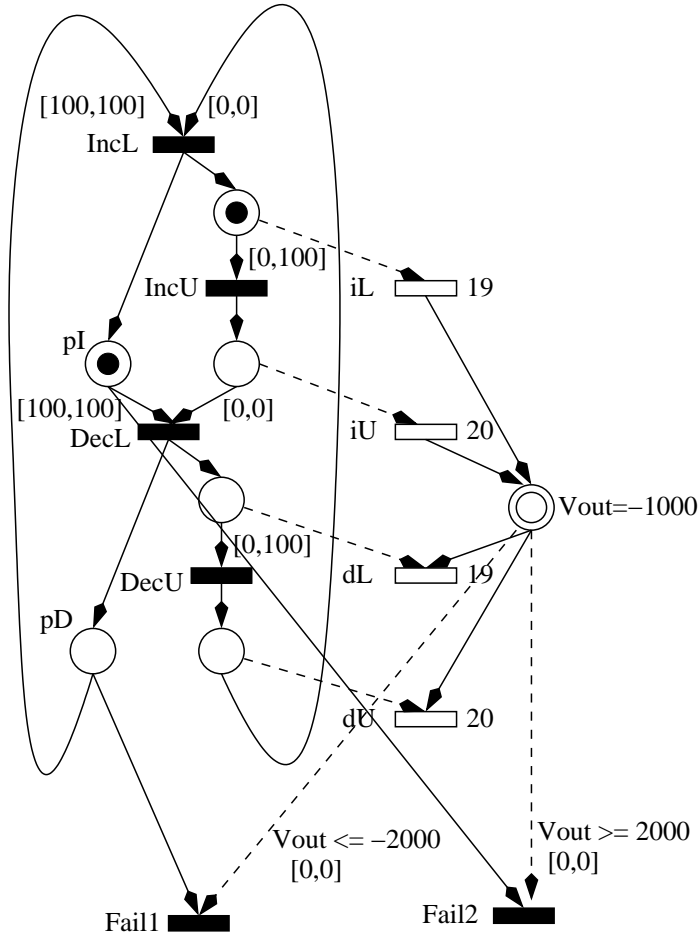


Fig. 7. THPN with fail condition for our motivating example.

We have also applied this verification method to some larger examples [15]. For example, we have performed reachability analysis of a simple PLL model finding its 1012 timed states in 6.9 seconds. We have also verified several versions of the tunnel diode oscillator [7,8,9] in just over 2 seconds and 2000 timed states. While all these examples are certainly small analog circuits, these results seem to indicate the promise of this approach. The bottleneck at this time is in obtaining models which fit the current modeling restrictions. Therefore, in our future work, we are developing both automatic compilers from high-level languages such as VHDL-AMS and extending our analysis algorithms to support new features such as ranges on the rates of change of the continuous variables.

5 Conclusions

While certainly the motivating example described in this paper is extremely simple and likely even a novice would find this problem in simulation, it does illustrate the potential for problems. The analog circuit verification methodology that our research group as well as others are developing should provide a means for analog designers to rapidly find “worst case” situations within their particular circuit, which may often not correspond to the extreme global parameter variations explored in traditional “corner” simulations.

References

- [1] Alur, R., C. Courcoubetis, T. A. Henzinger and P.-H. Ho, *Hybrid automata: An algorithmic approach to the specification and verification of hybrid systems*, in: *Hybrid Systems*, Lecture Notes in Computer Science **736**, 1993, pp. 209–229.
- [2] Chen, H. and H.-M. Hanisch, *Analysis of hybrid systems based on hybrid net condition/event system model*, *Discrete Event Dynamic Systems: Theory and Applications* **11** (2001), pp. 163–185.
- [3] Christen, E. and K. Bakalar, *VHDL-AMS—a hardware description language for analog and mixed-signal applications*, *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing* **46** (1999), pp. 1263–1272.
- [4] Dang, T., A. Donze and O. Maler, *Verification of analog and mixed-signal circuits using hybrid systems techniques*, in: *Formal Methods for Computer Aided Design*, 2004.
- [5] David, R. and H. Alla, *On hybrid petri nets*, *Discrete Event Dynamic Systems: Theory and Applications* **11** (2001), pp. 9–40.
- [6] Ghosh, A. and R. Vemuri, *Formal verification of synthesized analog designs*, in: *International Conference on Computer Design*, 1999, pp. 40–45.
- [7] Gupta, S., B. H. Krogh and R. A. Rutenbar, *Towards formal verification of analog designs*, in: *International Conference on Computer-Aided Design*, 2004, pp. 210–217.
- [8] Hartong, W., L. Hedrich and E. Barke, *Model checking algorithms for analog verification*, in: *Design Automation Conference*, 2002, pp. 542–547.
- [9] Hartong, W., L. Hedrich and E. Barke, *On discrete modeling and model checking for nonlinear analog systems*, in: *Computer Aided Verification. 14th International Conference*, 2002, pp. 401–413.
- [10] Hedrich, L. and E. Barke, *A formal approach to nonlinear analog circuit verification*, in: *International Conference on Computer-Aided Design*, 1995, pp. 123–127.

- [11] Hedrich, L. and E. Barke, *A formal approach to verification of linear analog circuits with parameter tolerances*, in: *DATE '98: Proceedings of the conference on Design, automation and test in Europe* (1998), pp. 649–655.
- [12] Hendricx, S. and L. Claesen, *A symbolic core approach to the formal verification of integrated mixed-mode applications*, in: *European Design and Test Conference*, 1997, pp. 432–436.
- [13] IBS Corporation, *Industry reports*. (2003).
- [14] Kurshan, R. P. and K. L. McMillan, *Analysis of digital circuits through symbolic reduction*, *IEEE Transactions on Computer-Aided Design* **10** (1991), pp. 1356–1371.
- [15] Little, S., D. Walter, N. Seegmiller, C. Myers and T. Yoneda, *Verification of analog and mixed-signal circuits using timed hybrid petri nets*, in: *Automated Technology for Verification and Analysis*, *Lecture Notes in Computer Science* **3299** (2004), pp. 426–440.
- [16] Salem, A., *Semi-formal verification of VHDL-AMS descriptions*, in: *International Symposium on Circuits and Systems*, 2002, pp. 123–127.
- [17] Serrano-Gotarredona, T. and B. Linares-Barranco, *Systematic width-and-length dependent CMOS transistor mismatch characterization and simulation*, *Analog Integrated Circuits and Signal Processing* **21** (1999), pp. 271–296.