# Qualitative Abstraction based Verification for Analog Circuits

Mohamed H. Zaki*, Sofiène Tahar*, Guy Bois**

*Dept. of Electrical & Computer Engineering, Concordia University
Montreal, Quebec, H3G 1M8, Canada
{mzaki, tahar}@ece.concordia.ca
**Genie Informatique, Ecole Polytechnique de Montreal
Montreal, Quebec, H3C 3A7 , Canada
guy.bois@polymtl.ca

**Abstract.** The verification of analog designs is a challenging and exhaustive task that requires deep understanding of the physical behaviors. In this paper, we propose a qualitative based predicate abstraction method for the verification of a class of non-linear analog circuits. The method is based on combining techniques from constraint solving and computer algebra along with symbolic model checking. We have implemented the proposed verification algorithms using the computer algebra system Mathematica and the SMV model checker.

## 1 Introduction

A cornerstone in embedded systems are analog designs, which are integrated circuits, required at the interfaces with the real world environment. Computer aided design (CAD) has been under intense research during the past decades to overcome challenges in the design process of analog designs. For verification purposes, simulation approaches are usually applied to check that a design is robust with respect to different types of inaccuracies. However, with designs growing in complexity, introducing more rigorous verification techniques, complementing simulation is becoming of great importance.

Formal methods like model checking have been advocated for the verification of digital designs where their correctness is proved mathematically against some formal properties. Motivated by the success of the application of formal methods in the verification of digital designs, researchers started investigating the formal verification of analog designs. In analog designs, one is interested in global properties connected to the dynamic behavior of the design. Unlike its digital counterpart, analog designs verification is a challenging and exhaustive task that requires deep understanding of their behavior. Challenging problems like non- linear effects make a direct application of formal methods very difficult and abstraction techniques are required in order to achieve this task. *Predicate abstraction* is one of the most successful abstraction approaches developed in (8), for the verification of systems with infinite state space. In this approach, the state space is divided into a finite set of regions and a set of rules is used to build the transition between these regions in a way that the generated state transition system can be verified using model checking. Among the proposed enhancements of predicate abstraction is the lazy abstraction approach (11). The basic idea here is instead of generating the entire abstract model, a region is abstracted only when it is needed in the verification step. Refinement is applied starting from the earliest state at which the abstract counterexample fails to have a concrete counterpart.

Inspired by the concept of lazy abstraction, we propose a qualitative abstraction approach for analog designs represented by non-linear ordinary differential equations (ODEs), such that satisfaction of the property in the abstract model guarantees its satisfaction in the circuit-level model. In the proposed abstraction, the state space is initially partitioned based on the qualitative properties of the analog model and symbolic constrained based methods are applied to check for property validation. In case of failure, an iterative verification/refinement process is applied where the regions violating the property are refined and symbolic model checking is applied for the property validation. We implemented the proposed verification algorithm using the computer algebra system *Mathematica* (17), along with the model checker *SMV* (3).

**Related Work.** Literature touching the different aspects of the work in this paper is wide enough and spans through different research domains. We will highlight in the following the most relevant work while in depth investigations can be found in references therein. The common trend in analog and mixed signal design formal verification is the necessity of the explicit state space exploration using approximate reachable sets corresponding to the continuous dynamics of the system, to deduce the validity of the properties of interest. This relies on the discretization using over-approximating domains like polyhedra and intervals. In (10), the authors proposed to use variable size hypercubes, enhancing the precision for the constructed state space. Variant approaches of the polyhedral based analysis were proposed. For instance, the model checking tools $d/dt$ (4), *Checkmate* (9) and *PHaver* (6) were adapted and used in the verification of a biquad low-pass filter (4), a tunnel diode oscillator and a $\Delta\Sigma$ modulator (9), and voltage controlled oscillators (6). In (18), the authors proposed a more precise approximation for the reachable states, where state space exploration algorithms are handled with Taylor approximations over interval domains. In fact, all above surveyed formal methods limit the verification of the circuit to a predefined time bound as they depend on explicit states exploration. In contrast, we propose in this paper qualitative based methods for the abstract models construction and verification, therefore, overcoming the time bound requirement. In addition, the predicate abstraction presented in the current paper allows non-linear partitioning of the state space based on qualitative information extracted from the system model.

In (1), the authors combined predicate abstraction with polyhedral analysis for the verification of reachability properties of linear hybrid systems. Similar, but a more general abstraction approach was proposed in (2). In (15; 16) a qualitative based approach was developed for abstract model generation for hybrid systems, based on higher derivative analysis and invariance generation. A similar invariant based approach was proposed in (14), where more general invariants are constructed. In (13), the authors proposed the idea of barrier certificates. Barrier certificates if they exists, are invariants that separate system behavior from a bad state, hence providing a safety verification approach. The work presented in this paper is in line with the above mentioned. However, we distinguish ourselves in several aspects. First, we propose to extract qualitative predicates from the system behavior which can complement the qualitative predicates presented in (15). We also propose different ideas for transition relation, based on a variant of the mean value theorem. In addition, the construction of the invariant predicates we use do not require a priori knowledge of initial conditions and in contrast to barrier certificates give knowledge of the whole system behavior rather than specific behaviors.

The rest of the paper is organized as follows: We introduce the proposed verification methodology in Section 2. After that, we proceed in Section 3 by defining the analog system model, explaining predicate abstraction and introducing a class of qualitative invariants. The constraint based verification approach is explained in Section 4, followed by the abstract model construction in Section 5. Implementation issues along with illustrative experimental results are described in Section 6, before concluding with a discussion in Section 7.

## 2 Proposed Methodology

The verification methodology we propose is illustrated in Figure 1. Starting with a circuit description as a system of ODEs, along with specification properties provided in computational temporal logic (∀CTL) (3), we symbolically extract qualitative predicates of the system. The abstract model is constructed in successive steps. In the basis step, we only consider predicates that define the invariant regions for the system of equations based on the Darboux theory of integrability (7). Informally, the Darboux theory is concerned with the identification of the different qualitative behaviors of the continuous state space of the system. We make use of such idea to divide the analog design state space into qualitatively distinct regions where no transition is possible between states of the different regions. Satisfaction of properties is verified on these regions using constraint based methods, which rely on qualitative properties of the system, by generating new constraints that prove or disprove a property. The property verification hence provides the advantage of avoiding explicit computation of reachable sets.

If the property cannot be verified at this stage, refinement is needed only for the non-verified regions by adding more predicates. Conventional model checking is then applied on the newly generated abstract model. The extraction of the predicates is incremental in the sense that more precision can be achieved by adding more information to the original construction of the system. When the property is marked violated, one possible reason is because of the false negative problem due to the over-approximation of the abstraction. In this case, refinement techniques may be introduced [1].

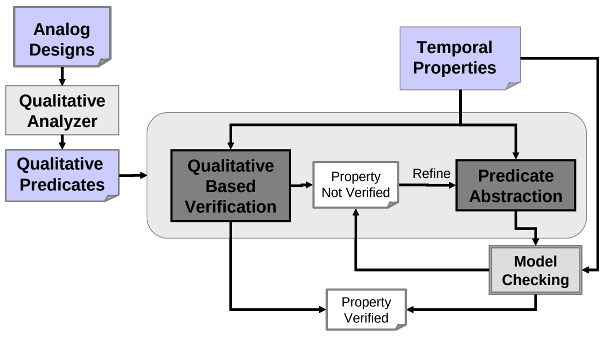


FIG. 1 – *Verification Methodology*

[1] We use a simple refinement procedure based on interval methods for ODEs that identify and eliminate the spurious counterexamples, however its description is outside the scope of this paper.

## 3 Preliminaries

### 3.1 System Description

We are concerned in this paper with analog circuits that are composed of basic components like inductors, capacitors and transistors in addition to non-linear sources (e.g., voltage controlled current sources). Such circuits can be described by non-linear polynomial ODEs as follows:

**Definition 1 Analog Circuit Model.** *An Analog circuit model is a tuple $\mathcal{A} = (\mathcal{X}, \mathcal{X}_0, \mathcal{P})$, with $\mathcal{X} = V_{c_1} \times V_{c_n} \times \ldots \times I_{l_m} \subseteq \mathbb{R}^d$ as the continuous state space with d-dimensions, where $V_{c_i}$ and $I_{l_j}$ are the voltage across the capacitance $c_i$ and the current through the inductance $l_j$, respectively. $\mathcal{X}_0 \subseteq \mathcal{X}$ is the set of initial states (initial voltages on the capacitances and currents through the inductance) and $\mathcal{P} : \mathcal{X} \to \mathbb{R}^d$ is the continuous vector field.*

The behavior of such analog circuit model $\mathcal{A}$ is governed by polynomial differential equations of the form: $\dot{x_k} = \frac{dx_k}{dt} = \mathcal{P}_k(x_1, \ldots, x_d) = a_0 + \sum_{l=1}^{m} \mathcal{P}_{l,k}(x_1, \ldots, x_d)$, where $t$ is the independent real time, $\mathcal{P}_k$ $(k = 1, \ldots, d)$ is a polynomial of degree $m$, $a_0$ is a constant and $\mathcal{P}_{l,k}$ is a polynomial of degree $l$, $\mathcal{P}_{l,k} = \sum_{i_1+\ldots+i_d=l} a_{i_1,\ldots,i_d} x_1^{i_1} \ldots x_d^{i_d}$, where $a_{i_1,\ldots,i_d}$ is a constant. We assume that the differential equation has a unique solution for each initial value. The semantics of the analog model $\mathcal{A} = (\mathcal{X}, \mathcal{X}_0, \mathcal{P})$ over a continuous time period $T_c = [\tau_0, \tau_1] \subseteq \mathbb{R}^+$ can be described as a trajectory $\Phi_x : T_c \to \mathcal{X}$ for $x \in \mathcal{X}_0$ such that $\Phi_x(t)$ is the solution of $\dot{x_k} = \mathcal{P}_k(x_1, \ldots, x_d)$, with initial condition $\Phi_x(0) = x$ and $t \in T_c$, is a time point. We can view the behavior of the analog model $\mathcal{A}$ as a transition system:

**Definition 2 Analog Transition System.** *The transition system for analog model $\mathcal{A}$ is described as a tuple $\mathcal{T}_{\mathcal{A}} = (Q, Q_0, \sigma, L)$ where $q \in Q$ is a configuration $(x, \Gamma)$, $x \in \mathcal{X}$ and $\Gamma$ is a set of intervals where $\cup_{i \geq 0} t_i \subseteq \mathbb{R}^+$, $t_i \in \Gamma$. We have $t_1, t_2 \in \Gamma$ for $\Phi_{x'}(t_1) = \Phi_{x''}(t_2) = x$ and $x', x'' \in \mathcal{X}_0$. $q \in Q_0$, when $t_0 \in \Gamma$ and $t_0$ is the singular interval, $\sigma \subseteq Q \times Q$ is a transition relation such that $(q_n, q_m) \in \sigma$ iff $\exists t_n \in \Gamma_n$, $\exists t_m \in \Gamma_m$. $t_n < t_m$ and $\lim_{t_n \to t_m} \Phi_x^{q_n}(t_n) = \Phi_x^{q_m}(t_m)$, $x \in \mathcal{X}_0$. Finally, L is an interpretation function such that $L : Q \to \mathbb{R}^n \times 2^{\mathbb{R}^+}$.*

The set of reachable states *Reach* can then be defined as: $Reach := \{q' \in Q | \exists q \in Reach^{(0)}, t \in L_\Gamma(q'), x' = L_x(q'), x = L_x(q)$ such that $\Phi_x(t) = x'\}$, where $Reach^{(0)} := Q_0$.

### 3.2 Predicate Abstraction

Predicate abstraction is a method where the set of abstract states is encoded by a set of Boolean variables representing each a concrete predicate. Based on (1), we define a discrete abstraction of the analog model $\mathcal{A}$ with respect to a given n-dimensional vector of predicates $\Psi = (\psi_1, \ldots, \psi_n)$, where $\psi : \mathbb{R}^d \to \mathbb{B}$, with $\mathbb{B} = \{0, 1\}$ and $d$ is the ODEs system dimension. A polynomial predicate is of the form $\psi(x) := \mathcal{P}(x_1, \ldots, x_d) \sim 0$, where $\sim \in \{<, \geq\}$. Hence, the infinite state space $\mathcal{X}$ of the system is reduced to $2^n$ states in the abstract system, corresponding to the $2^n$ possible Boolean truth evaluates of $\Psi$.

**Definition 3 Abstract Transition System.** An abstract transition system *is a tuple $\mathcal{T}_\Psi = (Q_\Psi, \rightsquigarrow, Q_{\Psi,0})$, where:*

- *$Q_\Psi \subset L \times \mathbb{B}^n$ is the abstract state space for a n-dimensional vector predicates, where an abstract state is defined as a tuple $(l,b)$, with $l \in L$ is a label and $b \in \mathbb{B}^n$.*
- *$\rightsquigarrow \subseteq Q_\Psi \times Q_\Psi$ is a relation capturing abstract transition such that $\{b \rightsquigarrow b' | \exists x \in \Upsilon_\Psi(b), t \in \mathbb{R}^+ : x' = \Phi_x(t) \in \Upsilon_\Psi(b') \wedge x \to x'\}$, where the concretization function: $\Upsilon_\Psi : \mathbb{B}^n \to 2^{\mathbb{R}^d}$ is defined as $\Upsilon_\Psi(b) := \{x \in \mathbb{R}^d | \forall j \in \{1,,\ldots,n\} : \psi_j(x) = b_j\}$.*
- *$Q_{\Psi,0} := \{(l,b) \in Q_\Psi | \exists x \in \Upsilon_\Psi(b), x \in \mathcal{X}_0\}$ is the set of abstract initial states.*

We define the set of reachable states as: $Reach_\Psi = \bigcup_{i\geq 0} Reach_\Psi^{(i)}$, where $Reach_\Psi^{(0)} = Q_{\Psi,0}$, $Reach_\Psi^{(i+1)} = Post_c(Reach_\Psi^{(i)})$, $\forall i \geq 0$ and $Post_c(l,b) := \{(l',b') \in Q_\Psi | (l,b) \rightsquigarrow (l',b')\}$. We can then deduce the following property between concrete and abstract reachable states.

**Lemma 1** *Given an Analog abstract transition system $\mathcal{T}_\Psi(\mathcal{A})$ and a vector of predicates $\Psi$, the following holds: $Reach \subseteq \{q \in Q | \exists (l,b) \in Reach_\Psi : x \in \Upsilon_\Psi(b) \wedge L_x(q) = x\}$*

## 3.3 Invariants

Usually, a continuous system has a behavior that varies in different regions of phase space which boundaries are defined by special system solutions known in the literature as Darboux invariants (7). These invariants partition the concrete state space into a set of qualitative distinctive regions.

**Definition 4** *Given the system of ODEs $\frac{dx_k}{dt} = \mathcal{P}_k(x_1(t),\ldots,x_d(t))$, with $k = 1,\ldots d$ ($\frac{d\mathbf{x}}{dt} = \mathbf{P}(\mathbf{x})$, $\mathbf{x} \in \mathbb{R}^d$ and $\mathbf{P} = (\mathcal{P}_1,\ldots,\mathcal{P}_d)$), we define the corresponding vector field as $\mathcal{D}_\mathbf{P} = \mathbf{P}.\partial_\mathbf{x} = \sum_{k=1}^d \mathcal{P}_k \frac{\partial}{\partial x_k}$.*

The correspondence between the system of ODEs and the vector field $\mathcal{D}_\mathbf{P}$ is obtained by defining the time derivative of functions of $\mathbf{x}$ as follows. Let $\mathcal{G}$ be a function of $\mathbf{x}$: $\mathcal{G} : \mathbb{R}^k \to \mathbb{R}$, then $\frac{d\mathcal{G}}{dt} := \dot{\mathcal{G}} = \mathcal{D}_\mathbf{P}(\mathcal{G}) = \mathbf{P}.\partial_\mathbf{x}\mathcal{G}$. The time derivative is called the derivative along the flow since it describes the variation of function $\mathcal{G}$ of $\mathbf{x}$ with respect to $t$ as $\mathbf{x}$ evolves according to the differential system. When $D_\mathbf{P}(\mathcal{G}) = 0$, $\forall \mathbf{x} \in \mathbb{R}^k$, we have a time independent first integral of $D_\mathbf{P}$. Several methods were developed recently based on Darboux integrability theory (7), which is a theory concerned with finding closed form solutions of system of ODEs, to tackle the problem by looking for a basis set of invariants, i.e., Darboux invariants. Rather than looking at functions which are constant on all solutions, we look at functions which are constant on their zero level set. Darboux polynomials $\mathcal{J}_i$ provide the essential skeleton for the phase space from which all other behaviors can be qualitatively determined.

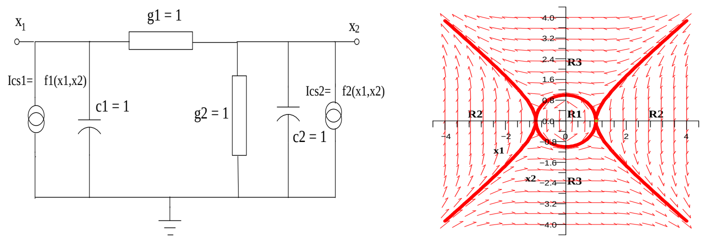
**Definition 5** **Darboux Polynomials** *(7). Given a vector field $\mathcal{D}_\mathcal{P} = \sum_{i=1}^{d} \mathcal{P}_i \frac{\partial}{dx_i}$ associated with the system $\frac{d\mathbf{x}}{dt} = \mathbf{P}(\mathbf{x})$, a Darboux polynomial is of the form $\mathcal{J}(\mathbf{x}) = 0$ with $\mathcal{J} \in \mathbb{R}[\mathbf{x}]$, when $D\mathcal{J} = \mathcal{K}\mathcal{J}$, where $\mathcal{K} = \mathcal{K}(\mathbf{x})$ is a polynomial called the cofactor of $\mathcal{J} = 0$.*

**Lemma 2** *Given a system of ODEs and a vector field $\mathcal{D}_\mathbf{f}$, $\mathcal{J}$ is an invariant of the system if $\mathcal{J}$ divides $\mathcal{D}_\mathbf{f}$, more formally, if there exists $\mathcal{K} \in \mathbb{R}[\mathbf{x}]$ such that $\mathcal{D}_\mathbf{f}(\mathcal{J}) = \mathcal{K}\mathcal{J}$. The solution set of the system vanishes on the curve of $\mathcal{J}$.*

In the context of abstraction, we define the invariant regions as conjunction of Darboux invariant predicates. An invariant region can be considered as an abstraction of the state space that confines all the system dynamics initiated in that region:

**Definition 6** *We say that a region $\mathcal{V}$ is an invariant region of an analog model $\mathcal{A}$ such that $\mathcal{P}(\mathbf{x}(0)) = s_0 \models \mathcal{V}$, $\mathcal{P}(\mathbf{x}(\varsigma)) = \mathbf{s}_\varsigma \models \mathcal{V}$ and $\forall t \in [0,\varsigma], \mathcal{P}(\mathbf{x}(t)) = s_t \models \mathcal{V}$. Let $\mathcal{V} = \{x \in \mathbb{R}^k | x \models \Gamma\}$, be an invariant region, where $\Gamma$ is a conjunct of Darboux predicates (each is of the form $p(\mathbf{x}) \sim 0$, where $p$ is a polynomial function and $\sim \in \{<, \geq\}$). If $\mathbf{x}(0)$ is some initial state, then $\mathcal{V} = \mathcal{V}(\mathbf{x}(0))$ denotes an over-approximation of the set of states reachable from $\mathbf{x}(0)$.*

**Example 1** *Consider the non-linear circuit shown in Figure 2.a, where the non-linearity comes from the voltage controlled current sources which currents $I_{cs1}$ and $I_{cs2}$ are described, respectively, as $f_1 = -x_2^3 + x_1 - x_2$ and $f_2 = -x_1^3 + 2x_2$. The voltages across the capacitors $c_1$ and $c_2$ can be described using ODEs, respectively, as follows: $\dot{x_1} = -x_2^3$* and *$\dot{x_2} = x_1 - x_1^3$. We identify the corresponding invariants: $j_1 = 1 - x_1^2 - x_2^2$ and $j_2 = 1 - x_1^2 + x_2^2$, which are used to form three invariant regions: $R_1 = j_1 \geq 0 \wedge j_2 \geq 0$, $R_2 = j_1 < 0 \wedge j_2 < 0$ and $R_3 = j_1 < 0 \wedge j_2 \geq 0$ as shown in Figure 2.b. Note that $j_1 \geq 0 \wedge j_2 < 0$ is infeasible and therefore discarded.*



(a) Analog Circuit Schematic (b) Darboux Invariants

FIG. 2 – *Illustrative Non-linear Analog Circuit*

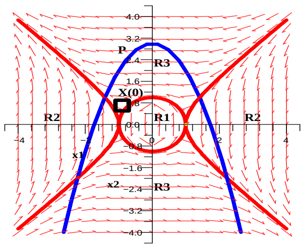
## 4 Constraint Based Verification

In this section, we propose a qualitative verification approach for analog circuits based on constraint based methods. The basic idea is to apply quantified constraint based techniques to answer questions about qualitative behaviors of the designs, by constructing functions that validate or falsify the property. The idea is different from conventional approaches as it does not require the explicit reachable states computation. We consider two types of properties that can be verified using that approach, namely safety and switching properties.

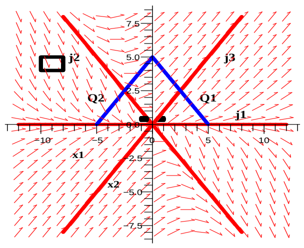
**Safety Properties.** Safety properties can be expressed in $CTL$ (3) as $\forall\Box p$; meaning that always on all executions the constraint predicate $p$ is satisfied for a set of initial conditions. The verification starts by getting the dual property $\exists\Diamond\neg p$ (which means that there is an execution falsifying the constraint $p$) and applies constraint solving on the dual property within the invariant regions of interest. In case of unsatisfiability, we conclude that the original property is satisfied in the region, otherwise we cannot conclude the truth of the property and a refinement model providing more details of the region is constructed.

**Proposition 1** ***Safety Property Verification.*** *$\forall\Box\mathcal{P}$ is always satisfied in an invariant region $\mathcal{V}$, if its dual property $\exists\Diamond\neg\mathcal{P}$ is never satisfied in that region.*

**Example 2** *Consider the circuit in Example 1, with initial conditions $x_1(0) \in [-0.7, -1.1]$ and $x_2(0) \in [0.5, 0.9]$. Suppose the property to check is $\forall\Box\mathcal{P} := x_1^2 + x_2 - 3 < 0$ (see Figure 3.a for details), meaning that all flows initiated from $\boldsymbol{x}(0) = (x_1(0), x_2(0))$, will be bounded by $x_1^2 + x_2 - 3$. The following regions satisfy the initial conditions $R_1 = j_1 \geq 0 \wedge j_2 \geq 0$ and $R_3 = j_1 < 0 \wedge j_2 \geq 0$. We check whether $\exists\Diamond\mathcal{P} := x_1^2 + x_2 - 3 \geq 0$ is satisfiable in the invariant regions $R_1$ and $R_3$. By applying constraint solving in Mathematica, we find that for the region $R_3$, the constraints system is satisfiable, hence the original property cannot be verified, and the state space of the region needs to be refined. For the region $R_1$, the constraints system is infeasible, therefore we conclude that the safety property is satisfied.*



(a) Safety Verification (Example 2)



(b) Switching Verification (Example 3)

FIG. 3 – *Constraint Based Verification for the Circuit in Figure 2.a*

**Switching Properties.** A special case of reachability verification $\exists\Diamond\mathcal{Q}$ is the switching condition verification, i.e., starting from a set of initial conditions, the system will eventually cross a threshold, triggering a switching condition. Such property is of great importance, for instance, a MOSFET transistor acting as switch changes states based on the voltage condition applied on its gate. We consider here a restricted form of the switching property, where we assume that threshold predicates divide the invariant region by intersecting the invariant region boundaries (at least two Darboux predicates). Given an invariant region $\mathcal{V}$, a predicate $\mathcal{Q}$ is a *switching condition* if $\bigwedge_{i=0}^{k} \exists\mathbf{x}.(\mathcal{Q}(\mathbf{x}) = 0) \wedge (I_i(\mathbf{x}) = 0)$, where $k \leq 2$ and $I$ is a Darboux invariant. The switching verification can be stated as follows:

**Proposition 2** **Switching Property Verification.** *$\exists\Diamond\mathcal{Q}$ is satisfied in a region $\mathcal{V}$, if $\mathcal{Q}(\mathbf{x}(0)) < 0$ and $\mathcal{D}_{\mathbf{P}}(\mathcal{Q}) > 0$ or if $\mathcal{Q}(\mathbf{x}(0)) > 0$ and $\mathcal{D}_{\mathbf{P}}(\mathcal{Q}) < 0$, in the region $\mathcal{V}$. If these conditions are satisfiable, we conclude that the property is verified and switching occurs.*

**Example 3** *Consider the circuit shown in Figure 2.a, where the voltages across the capacitors $c_1$ and $c_2$ are described, respectively, as follows: $\dot{x}_1 = x_1^2 + 2x_1x_2 + 3x_2^2$ and $\dot{x}_2 = 4x_1x_2 + 2x_2^2$, with initial conditions $x_1(0) \in [0.5, 1]$ and $x_2(0) \in [0.3, 0.5]$. Suppose that the switching condition property to check is $\exists\Diamond x_1 + x_2 - 5 = 0$, meaning that switching occurs when a certain trajectory will cross the threshold $Q_1 := x_1 + x_2 - 5 = 0$ (see Figure 3.b). We construct the Darboux functions: $j_1 := x_2, j_2 := x_1 + x_2, j_3 := x_1 - x_2$. The region $R_1 = j_1 > 0 \wedge j_2 > 0 \wedge j_3 > 0$ satisfies the initial conditions. In addition, the predicate $x_1 + x_2 - 5 < 0$ satisfies the initial condition and $\mathcal{D}_{\mathbf{P}}(x_1 + x_2 - 5) > 0$ because $\mathcal{D}_{\mathbf{P}}(x_1 + x_2 - 5) = (x_1 + x_2)(x_1 + 5x_2)$ is always positive in $R_1$. Consider the initial conditions $\mathbf{X}(0)_1 := (x_1(0) \in [-10, -8]$ and $x_2(0) \in [4, 5])$*

*and* $\mathbf{X}(0)_2 := (x_1(0) \in [-0.5,-1]$ *and* $x_2(0) \in [0.3,0.5])$ *in the invariant region* $R_2 = j_1 > 0 \wedge j_2 < 0 \wedge j_3 < 0$. *For the switching condition* $Q_2 := -x_1 + x_2 - 5 = 0$, *we find that the initial condition* $\mathbf{X}(0)_1$ *satisfies* $-x_1 + x_2 - 5 > 0$, *and* $\mathbf{X}(0)_2$ *satisfies* $-x_1 + x_2 - 5 < 0$ *while* $\mathcal{D}_{\mathbf{P}}(-x_1 + x_2 - 5) = -(x_1 - x_2)^2$ *will be always negative in region* $R_2$, *therefore we conclude that the switching will occur for the initial condition* $\mathbf{X}(0)_1$ *but not for* $\mathbf{X}(0)_2$.

Sometimes constraint based verification fails to provide answers for the verification problem as the above methods are not complete in general. In addition, more complex properties like oscillation cannot be proved using the above method. We complement the approaches described in this section, by the predicate abstraction method allowing conventional model checking to be applied.

## 5 Predicate Abstraction

### 5.1 Abstract State Space

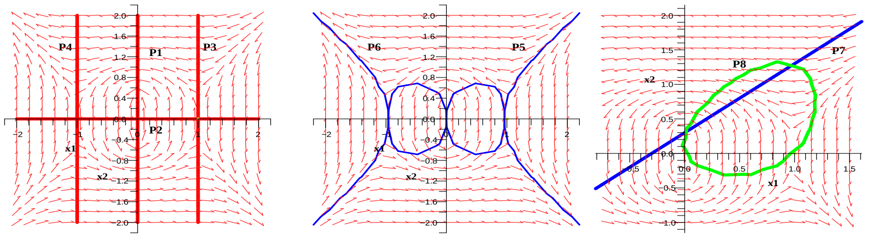
In general, the effectiveness of the predicate abstraction method depends on the choice of predicates. In addition of using Darboux predicates described in Section 3.3, we choose predicates identified in the properties of interest. In addition to temporal property predicates, basic ideas from the qualitative theory of continuous systems can be adapted within the predicate abstraction framework. The termination of the predicate generation phase is not necessary for creating an abstraction. We can stop at any point and construct the abstract model. A larger predicate set yields a finer abstraction as it results in a larger state space in the abstract model.

A set of predicates can be constructed using the notion of *critical forms*, which are special functions along them, the vector field direction is either vertical or horizontal. In between these forms, there can be no vertical nor horizontal vectors. In a region (abstract state) determined by the critical forms, all vectors follow one direction. These predicates can be obtained easily by setting $\dot{\mathbf{x}} = 0$. A generalization of critical forms is the concept of *isoclines*. Isoclines are functions over which the system trajectories have a constant slope. A predicate $\pi$ is an isocline of ODEs system if and only if $\exists a_i \in \mathbb{R}$ with $i = 1, \ldots d$ such that $\Sigma_{i=1}^{d} a_i \mathcal{P}_i(\mathbf{x})|_\pi = 0$. Isocline and critical forms provide qualitative information about the system behavior. Hence, such information can be used in refuting certain behavior that is shown unreachable. For instance by knowing different constants $a_i$, we deduce the direction of the flow crossing the isoclines and therefore we decide how to build transitions between abstract states. Finding different isocline predicates within an invariant region can be achieved by solving constraints on the parameters of predefined forms of an isocline predicate.

Another kind of predicates, we propose, referred to as *conditioned predicates*, have the property that under specific conditions, they provide certain information about the solution flow. A predicate $\pi$ is a conditioned predicate of an ODEs system with conditions $\Gamma_1, \ldots, \Gamma_d$, if it is of the form $\Sigma_{i=1}^{n} \Gamma_i \mathcal{P}_i(\mathbf{x})|_\pi = 0$, where the conditions $\Gamma_i$ are polynomials with $i = 1, \ldots d$ and $d$ is the system dimension. For instance, consider the 3-dimensional system with the state variables $x, y, z$, and the property predicate $z > 1$. We can construct another predicate that intersects $z > 1$ at specific conditions, say $\frac{\dot{y}}{\dot{x}} = 0$. Then, the new predicate is of the form $\dot{y} - (z-1)\dot{x} = 0$.

**Example 4** *Consider the analog circuit in Example 1. The critical forms predicates are* $p_1 := x_1$, $p_2 := x_2$, $p_3 := 1 - x_1$ *and* $p_4 := 1 + x_1$, *as shown in Figure 4.a. For illustration purposes, we choose two isocline predicates* $p_5 := x_1 - x_1^3 + x_2^3$ *and* $p_6 := x_1 - x_1^3 - x_2^3$ *to show in Figure*

*4.b. Suppose, we are interested to verify a property including the predicate $p_7 := x_2 - x_1 > 0.3$, we can construct the conditioned predicate $p_8 := \dot{x_2} - (x_2 - x_1 - 0.3)\dot{x_1} = 0$ as shown in Figure 4.c. To build the abstract state space, we have three invariant regions and eight predicates. As certain combination of predicates are infeasible, the number of abstract states is $< 2^8$ abstracts states. In fact, region $R_1 = j_1 \geq 0 \wedge j_2 \geq 0$ is subdivided into 29 abstract states.*



(a) Critical Forms Predicates (b) Isocline Predicates (c) Conditioned Predicates

FIG. 4 – *Predicates for the Circuit in Figure 2.a*

## 5.2 Computing Abstract Transitions

One main issue in constructing abstract state transition systems is the identification of the possible transitions. As we divide the state space into invariant regions, we need only to construct transitions between abstract states within a region. Therefore, we do not need to construct an abstract model for the whole state space. In general, information from the solution of the ODEs is required to describe transitions between abstract states. In practice, each abstract transition is initialized to the trivial relation relating all states and then stepwise refined by eliminating unfeasible transitions. This guarantees that any intermediate result represents an abstraction and the refinement can be stopped at any point of time. In the remaining of this section, we use a set of different rules to construct transition between abstract states.

The simplest rule to use is the *Hamming distance rule* (15). The Hamming distance (HD) is the number of predicates for which the corresponding valuations are different in different abstract states. For instance, the Hamming distance between state $s_1 := (p_1 = 1 \wedge p_2 = 0 \wedge p_3 = 1 \wedge p_4 = 1)$ and state $s_2 := (p_1 = 1 \wedge p_2 = 0 \wedge p_3 = 0 \wedge p_4 = 1)$ is 1, written $HD(s_1, s_2) = 1$. Given two abstract states $s_1$ and $s_2$, we say that a transition can exist between two abstract states only if $HD(s_1, s_2) = 1$. The next rule we apply is based on the *generalized mean value theorem* (5), which is an extension of the mean value theorem (MVT) for n-dimension.

**Theorem 1 Generalized Mean Value Theorem** *(5). If $\boldsymbol{x}(t)$ is continuous on a time interval $t_1 \leq t \leq t_2$, and differentiable on $t_1 < t < t_2$, and assuming that there exists a vector $\boldsymbol{V}$ orthogonal to $\boldsymbol{x}(a)$ and to $\boldsymbol{x}(b)$, then there is $t_c : t_1 < T_c < t_2$ such that $\boldsymbol{V}$ is orthogonal to $\dot{\boldsymbol{x}}(t_c)$*

We use quantified constraint based methods to check whether such condition is satisfied between two abstract states. If the MVT is not satisfied, we deduce that no transition exists between the two states. The above rules give an over-approximation of the transition system as no information about the vector field direction is used. In order to remove such redundant transitions in the region of interest, we complement the above rules by applying the *intermediate value theorem* as a way to identify the flow direction. In the context of abstraction, a transition between two abstract states exists if a predicate valuation change during the execution over an interval domain as follows:

**Theorem 2 Intermediate Value Theorem** *Given a predicate $\lambda$, two states $S_1 = (l,b)$ and $S_2 = (l',b')$ differing only on the valuation of $\lambda$ and a time step interval solution $I : \{a_1 \leq x \leq a_2\}$, there is a transition between $S_1$ and $S_2$ if $b \models [\![\lambda]\!]_{a_1}$ (i.e., $\lambda(a_1) \in \gamma(b)$), $b' \models [\![\lambda]\!]_{a_2}$ (i.e., $\lambda(a_2) \in \gamma(b')$), $[\![\lambda]\!]_{a_1} \neq [\![\lambda]\!]_{a_2} \neq 0$ and $\exists x$ such that $[\![\lambda]\!]_x = 0$, with the interpretation function $[\![.]\!] : \mathbb{R}^d \rightarrow \{+,-,0\}$*

To check for the above condition, we use interval analysis to guarantee that the solution is reliable; the real solutions are enclosed by the computed intervals. Such guarantee is derived from the fundamental theorem of interval analysis (12).

**Theorem 3** *(12) Let $f : \mathbb{R}^d \rightarrow \mathbb{R}$ be a continuous function, then $F : \mathbb{I}^d \rightarrow \mathbb{I}$ is an interval extension of $f$ if $\{f(x_1,\ldots,x_d) | x_1 \in X_1,\ldots,x_d \in X_d\} \subseteq F(X_1,\ldots,X_d)$, where $\mathbb{I}$ is the interval domain.*

## 5.3 Abstraction Based Verification

Given the analog model transition system $\mathcal{T}_{\mathcal{A}}$ and a property $\varphi$ expressed in $\forall$CTL. The problem of checking that the property holds in this model written as $\mathcal{T}_{\mathcal{A}} \models \varphi$ can be simplified to the problem of checking that a related property holds on an approximation of the model $\mathcal{T}_{\Psi}$, i.e., $\mathcal{T}_{\Psi} \models \tilde{\varphi}$, with $\tilde{\varphi} = \mu(\varphi)$, where $\mu$ is a mapping function: $\mu : \mathbb{R}^d \rightarrow \mathbb{B}$ which is a function associating to each predicate $\lambda(x_1,\ldots,x_d)$ an atomic proposition $P$. The main preservation theorem can be stated as follows (2):

**Theorem 4** *Suppose $\mathcal{T}_{\Psi}$ is an abstract model of $\mathcal{T}_{\mathcal{A}}$, then for all $\forall\tilde{C}TL$ state formulas describing $\mathcal{T}_{\Psi}$ and every state of $\mathcal{T}_{\mathcal{A}}$, we have $\tilde{s} \models \tilde{\varphi} \Rightarrow s \models \varphi$, where $s \in \gamma(\tilde{s})$. Moreover, $\mathcal{T}_{\Psi} \models \tilde{\varphi} \Rightarrow \mathcal{T}_{\mathcal{A}} \models \varphi$.*

If a property is proved on an abstract model $\mathcal{T}_{\Psi}$, then we are done. If the verification of $\mathcal{T}_{\Psi}$ reveals $\mathcal{T}_{\Psi} \not\models \tilde{\varphi}$, then we cannot conclude that $\mathcal{T}_{\mathcal{A}}$ is not safe with respect to $\tilde{\varphi}$, since the counterexample for $\mathcal{T}_{\Psi}$ may be spurious. In order to remove spurious counterexamples, refinement methods on the abstract model can be applied (2).

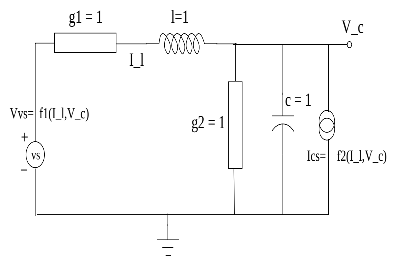
# 6 Implementation and Experiments

For experimentation purposes, we used Mathematica's algebraic manipulation and quantified constraint solving capabilities (17) for the constraint based verification and for the construction of the abstract model. Conventional model checking on the abstract models is applied using SMV (3). For instance, the built-in Mathematica function *Reduce*[*expr*, *vars*] simplifies the statement *expr* by solving equations or inequalities for the state variables $vars = \{v_1, v_2, \ldots, v_m\}$ and eliminating quantifiers. *Reduce* gives *True* if the *expr* is proved to be always true, *False* if *expr* is proved to be always false and a reduced *expr* otherwise. For example, the safety verification problem in Example 2 can be formulated using *Reduce* as follows: $Reduce[Exists[\{x_1,x_2\}, 1-x_1^2-x_2^2 \geq 0 \,\&\&\, 1-x_1^2+x_2^2 \geq 0, -3+x_1^2+x_2 \geq= 0], \{x_1,x_2\}]$.

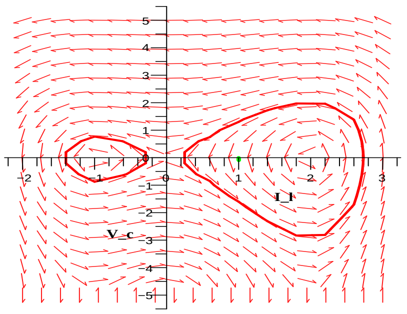
The problem of finding invariants is an important part of the methodology. We need to find Darboux invariants and in the case of reachability verification, we look for invariants bounding the reachable states. Finding invariants is based on the evaluation of the coefficients of the predefined forms of polynomials. In this algorithm, we start with an invariant form with an initial degree and check if such invariant exists, if not, we increase the degree to form a new polynomial. A bound on the degree must also be specified to ensure termination of the search of the invariants. An arbitrarily assigned bound at the beginning of the algorithm is usually proposed hence ensuring termination. This is possible using the Mathematica *FindInstance* function, for example. $FindInstance[expr, vars]$ finds an instance of *vars* that makes *expr True* if an instance exists, and gives $\{\}$ if it does not. The result of *FindInstance* is of the form $\{\{v_1 \rightarrow inst_1, v_2 \rightarrow inst_2, \ldots, v_m \rightarrow inst_m\}\}$ where $inst_i$ is the provided value. For example, to find the Darboux invariants $j$ we apply *FindInstance* as follows: $FindInstance[ForAll[\{x,y\}, \mathcal{D}j == \mathcal{K}j], \{coefs\}]$, where $j$ is a polynomial in $x, y$, with unknown coefficients $coefs$ and $K$ is the cofactor.

## 6.1 Experimentation

We applied the proposed verification to a variety of analog circuits. In addition to the different configurations of the circuit in Figure 2, we verified the oscillation properties for the circuit shown in Figure 5.a, with non-linear voltage source *vs* and non-linear current source *cs*. Given the state space and the generated invariant regions (boundaries shown in bold in Figure 5.b), we verify the following ACTL property on the set of trajectories: $\forall\Box(\forall\Diamond(V_c > I_l)) \wedge \forall\Box(\forall\Diamond(V_c < I_l))$, which can be understood as on every computation path, whenever the capacitor voltage $V_c$ value exceeds the inductor current value $I_l$, it will eventually decrease below $I_l$ again and vise-versa. We constructed the abstract model of the oscillator and verified the property using SMV. We found indeed that the circuit will always oscillate only inside the bounded regions shown in Figure 5.b (Experimental details can be found in (19).



(a) Analog Circuit



(b) Phase Portrait and Invariant Regions

FIG. 5 – *Non-Linear Oscillator*

## 7 Conclusion

In this paper, we proposed a qualitative verification approach of analog circuits. The approach is based on abstracting and verifying the qualitative behavior of the circuits using a combination of techniques from predicate abstraction and constraint solving along with model checking. The principle novelties in this paper are: (1) We adapted the concept of lazy abstraction for the verification of analog circuits. To this aim, we identified a set of basic qualitative predicates (Darboux polynomials) as invariance predicates which helps avoid the construction of an abstract model for the whole state space. (2) We proposed a constraint solving approach for the verification of safety and reachability properties. Such method does not require explicit representation of state space, but relies on generating functions that prove of disapprove the properties. Our methodology overcomes the time bound limitations of exhaustive methods developed in related work. Future work includes extending the predicate abstraction to support analog and mixed signal systems and exploring more case studies.

## References

[1] R. Alur, T. Dang, F. Ivancic. Reachability Analysis Via Predicate Abstraction. In HSCC, LNCS 2289, pp. 35-48. Springer, 2002.

[2] E. Clarke, A. Fehnker, Z. Han, B. Krogh, O. Stursberg, M. Theobald. Verification of Hybrid Systems based on Counterexample-guided Abstraction Refinement. In TACAS, LNCS 2619, pp. 192–207, Springer, 2003.

[3] E. Clarke, O. Grumberg , D.A. Peled. Model Checking. MIT Press, 1999.

[4] T. Dang, A. Donze, O.Maler: Verification of Analog and Mixed-Signal Circuits Using Hybrid System Techniques. In FMCAD, LNCS 3312, pp. 14-17, Springer, 2004.

[5] M. Furi, M. Martelli. A Multidimensional Version of Rolle's Theorem. The AMS, 102(3), 1995, pp. 243-249.

[6] G. Frehse, B. H. Krogh, R. A. Rutenbar. Verifying Analog Oscillator Circuits Using Forward/Backward Abstraction Refinement. In DATE, IEEE/ACM, pp. 257-262, 2006.

[7] A. Goriely. Integrability and Nonintegrability of Ordinary Differential Equations, Advanced Series on Nonlinear Dynamics, Vol 19, World Scientific, 2001.

[8] S. Graf and H. Saidi. Construction of Abstract State Graphs with PVS. In CAV, LNCS 1254, pp. 72-83. Springer, 1997.

[9] S.Gupta, B.H. Krogh, R.A. Rutenbar. Towards Formal Verification of Analog Designs, In ICCAD, IEEE/ACM, pp. 210-217, 2004.

[10] W. Hartong, R. Klausen, L. Hedrich. Formal Verification for Nonlinear Analog Systems: Approaches to Model and Equivalence Checking, In Advanced Formal Verification, pp. 205-245, Kluwer, 2004.

[11] A. Henzinger, R. Jhala, R. Majumdar, and G. Sutre. Lazy Abstraction. In POPL, ACM, pp. 58–70, 2002.

[12] R. E. Moore. Methods and Applications of Interval Analysis, SIAM, 1979.

[13] S. Prajna, A. Jadbabaie. Safety Verification of Hybrid Systems Using Barrier Certificates. In HSCC, pp. 477-492, Springer, 2004.

[14] S. Sankaranarayanan, H. Sipma, Z. Manna. Constructing Invariants for Hybrid Systems. In HSCC, LNCS 2993, pp 539-554, Springer, 2004.

[15] A. Tiwari and G. Khanna. Series of Abstractions for Hybrid Automata. In HSCC, LNCS 2289, pp. 465-478, Springer, 2002.

[16] A. Tiwari and G. Khanna. Nonlinear Systems: Approximating Reach Sets. In HSCC, LNCS 2993, pp. 600-614, Springer, 2004.

[17] S. Wolfram. Mathematica: A System for Doing Mathematics by Computer. Addison Wesley Publishing, 1991.

[18] M. Zaki, G. Al Sammane, S. Tahar, and G. Bois. Combining Symbolic Simulation and Interval Arithmetic for the Verification of AMS Designs, In FMCAD, IEEE, 2007.

[19] M. Zaki, S. Tahar, and G. Bois: Combining Constraint Solving and Formal Methods for the Verification of Analog Designs. Technical Report, ECE Department, Concordia University, Montreal, Quebec, Canada, May 2007, website http://hvg.ece.concordia.ca/Publications/TECH_REP/ANA_CBV_TR07