This is a preprint of the paper published in Formal Methods in System Design, 36(2), pp. 131-147 with DOI 10.1007/s10703-009-0086-9.

Advanced Methods for Equivalence Checking of Analog Circuits with Strong Nonlinearities

Sebastian Steinhorst · **Lars Hedrich**

Abstract In this contribution two extensions for an analog equivalence checking method are proposed, enabling the checking of strongly nonlinear circuits with floating nodes such as digital library cells. Therefore, a structural recognition and mapping of eigenvalues, representing the dynamics, to circuit elements via circuit variables is presented. Additionally, the introduction of reachability analysis is significantly restricting the investigated state space to the relevant parts, avoiding false negatives. The newly introduced methods are compared to existing ones by application to industrial examples.

Keywords Equivalence Checking · Analog Circuits · Formal Verification

1 Introduction

Increasing circuit complexity and the nowadays widely used mixed-signal circuits require formal methods in analog circuit design, accompanying the existing digital formal verification methods.

While digital design efficiency benefited massively from the introduction of formal verification methods, the design efficiency of analog circuits is still derogated by time consuming manual verification tasks. The development of new approaches for automated analog verification is mandatory to keep up with the economic constraints of decreasing time to market and shortening product life cycles.

Analog circuits in general belong to the class of nonlinear circuits. While weakly nonlinear circuits exhibit small deviations from linear behavior near an operating point and can be approximated by Volterra kernels, strongly nonlinear circuits can totally change the analog behavior [1]. Examples for the former class are filters, operational amplifiers, mixers,

This work was partly developed within the project VeronA (project label 01 M 3079) which is funded within the Research Program ICT 2020 by the German Federal Ministry of Education and Research (BMBF).

Sebastian Steinhorst · Lars Hedrich
Electronic Design Methodology
Institute for Computer Science
Goethe University of Frankfurt/Main
Robert-Mayer-Str. 11-15
D-60325 Frankfurt/Main, Germany
E-mail: [steinhorst, hedrich]@em.cs.uni-frankfurt.de

etc. considered around their operating point. The latter class is, for example, represented by digital circuits investigated in the analog domain, and analog circuits driven into supply voltage limitations.

Formal verification approaches can generally be divided into equivalence checking and model checking. Being the more universal method, model checking enables the proof of properties of a system, while equivalence checking attempts to prove the input-output equivalence of two systems. This problem could also be solved indirectly by model checking. However, this approach nearly doubles the state space dimensions. Due to the exponential relation between runtime complexity and state space dimensions, the model checking approach is generally impracticable for equivalence checking problems. Hence, in the digital domain as well as in the analog domain, special equivalence checking methods can handle larger circuits and fit well into the design flow.

However, equivalence and model checking methods could learn from each other. Until recently, a lot of approaches evolved for model checking of hybrid / analog systems. Due to the emphasis of this contribution on strongly nonlinear analog circuits described on transistor or block level, only methods for nonlinear systems will be considered.

A first approach to formally deal with hybrid systems is given in [2], followed by extensions to differential algebraic equations (DAEs) [3]. A recent method to deal with special circuits in an abstract way using recurrence equations is given in [4]. In [5] piecewise nonlinear ordinary differential equations (ODEs) are used to model an analog oscillator. Methods dealing with strongly nonlinear implicit equations use sampling [6], simulation traces [7] and affine arithmetics [8]. Most of the above methods mainly deal with reachability analysis and do not offer an equivalence checking methodology.

In the approaches to equivalence checking of analog circuits [9–11], sampling methods are also applied to prove whether two nonlinear analog circuits are equivalent or not.

There are three major areas of application for analog equivalence checking:

- Proving the equivalence of behavioral models and the implemented transistor netlist. Consequently, the former could be safely used for faster system simulations.
- Checking an extracted transistor netlist with many parasitic elements from interconnect versus the nominal netlist.
- Comparing structural transformations regarding the equivalence of their behavior.

This contribution is organized as follows. Section 2 introduces to analog equivalence checking methodology. Subsequently, new eigenvalue mapping methods using observability and structural information as well as the addition of reachability analysis to the methodology are described in Section 3, preventing the detection of false negatives. The results in Section 4 demonstrate the feasibility of the newly introduced approaches. Section 5 concludes.

2 Equivalence Checking Method

The basic goal of equivalence checking of nonlinear analog circuits is to verify if both systems have the same input-output behavior. Due to the focus on nonlinear dynamic systems, the internal (continuous) states have to be considered in addition. In this section an equivalence checking method is presented, incorporating the state space description, the transformation to a canonical state space, and the comparison of the obtained canonical state space representations.

2.1 State Space Description

In contrast to other methods dealing with nonlinear ordinary differential equations [5], this contribution focuses on circuit descriptions based on a system of implicit differential algebraic equations (DAEs)

$$\vec{f}(\vec{x}(t), \dot{\vec{x}}(t), \vec{u}(t)) = \vec{0}\,. \tag{1}$$

where $\vec{x}(t)$ is the vector of n system variables $x_i(t), i \in 1..n$ and $\vec{u}(t)$ is the vector of m input variables $u_k(t), k \in 1..m$ of the circuit. This system of equations can be set up by a Modified Nodal Approach (MNA) [12]. Application of the MNA results in implicit equations for each circuit node with the system variables usually being the node voltages, some device currents, and additional variables resulting from device equations or behavioral description of parts of the analog circuit. Another characteristic of the MNA is the high occurrence of algebraic equations and the occurrence of only some differential equations.

The state space of the system is a subspace of $\mathbb{R}^n$, spanned by n_z state variables $x_i^z(t), i \in 1..n_z$. The number of independent state variables is not always clear, e.g. due to capacitor loops. Additionally, extracted netlists have lots of resistor-capacitor paths leading to many state variables which may not all be of interest for the main input-output behavior. The identification of the state space variables independently spanning the state space will be introduced later.

In the following, we will not always denote the dependency on t, hence $x(t)$ can be written as x. The output of the system is described as a subset of predefined output variables $x_k^o, k \in 1..n_o$. The vector of output variables $\vec{x}^o$ can be calculated using a selection matrix $\mathbf{r}$:

$$\vec{x}^o = \mathbf{r} \cdot \vec{x}\,, \tag{2}$$

with $\mathbf{r} \in \{0,1\}^{n_o \times n}$.

2.2 Transformation to Canonical State Space

In order to verify the equivalence of two circuits A and B, the following approach is applicable:

- Sample the state space of both systems iteratively and perform the following transformation on each sample point.
- Transform the dynamics $\vec{f}$ in the variable space spanned by $\vec{x}$ and $\vec{u}$ of each system into a canonical state space spanned by a vector $\vec{z}(t)$ of n_z canonical state variables $z_i(t), i \in 1..n_z$. The transformation is defined by:

$$\begin{aligned} \vec{x} &= \vec{F}(\vec{z})\,, \\ \vec{z} &= \vec{F}^{-1}(\vec{x}) \end{aligned} \tag{3}$$

- The dynamics will then be transformed according to:

$$\vec{h}(\vec{z}(t), \dot{\vec{z}}(t), \vec{u}(t)) = \vec{f}(\vec{F}(\vec{z}(t)), \tfrac{\partial \vec{F}(\vec{z}(t))}{\partial t}, \vec{u}(t))\,. \tag{4}$$

  If the transformation is well chosen, the resulting system $\vec{h}$ will have only n_z nontrivial dynamic equations describing the system behavior. The remaining algebraic equations should be trivial (e.g. $0 = 0$).

2.3 Canonical State Space Comparison

After obtaining a canonical state space representation of both systems A and B, the transformed system functions $\vec{h}_A$ and $\vec{h}_B$ have to be compared in the canonical state space $\vec{z}$. This assumes that both circuits A and B are transformed to canonical state space variables $\vec{z}_A$ and $\vec{z}_B$ with equal size $n_{z_A} = n_{z_B} = n_z$. The dynamics of the system functions can be compared by checking the values of the state derivatives $\dot{\vec{z}}_A$, $\dot{\vec{z}}_B$ to be equal for each state in the state space. This comparison should be performed numerically, resulting in an error value with an appropriate norm:

$$\varepsilon_{\dot{z}} = ||\dot{\vec{z}}_A - \dot{\vec{z}}_B|| \tag{5}$$

Obviously this error will never be zero for analog circuits. Therefore, the user has to define an error value limit. If the error is below this limit in the whole reachable state space, the circuits are regarded as equivalent. From our experience, this error is very sensitive to differences in the circuits under verification. For example, a relative error limit of 10% can be considered as appropriate. Besides the internal dynamics, the output variables of the systems

$$\begin{aligned} \vec{x}^o_A &= \mathbf{r}_A \cdot \vec{F}_A(\vec{z}), \\ \vec{x}^o_B &= \mathbf{r}_B \cdot \vec{F}_B(\vec{z}) \end{aligned} \tag{6}$$

can be compared with a similar error measure:

$$\varepsilon_y = ||\vec{x}^o_A - \vec{x}^o_B|| \tag{7}$$

Comparing the systems' dynamics, as well as their output variables, assures that dynamic and static behavior of the systems under verification have been considered for equivalence checking, resulting in high confidence in the obtained results. The described equivalence checking methodology is summarized in Figure 1.

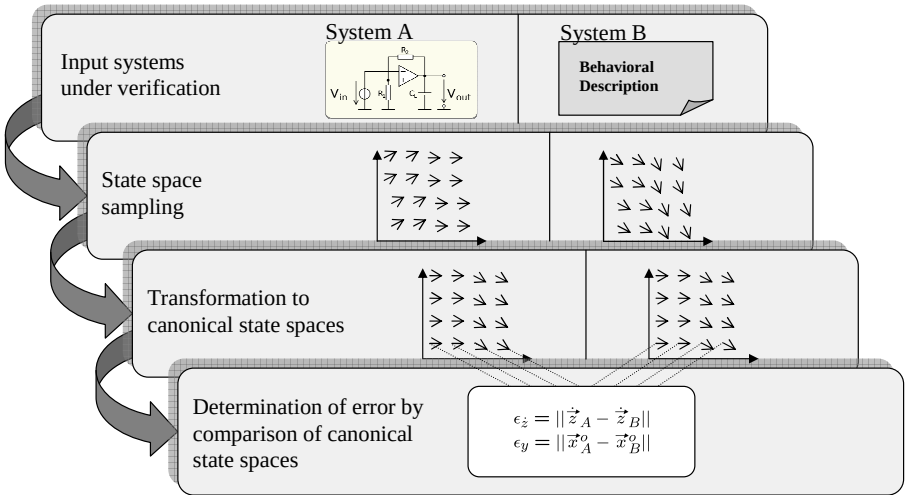


Fig. 1 Structure of the equivalence checking methodology.

2.3.1 Numerical Transformation

In general, a transformation fulfilling Equation (3) and (4) can not be found in an analytical way. However, it can be obtained numerically. The idea is to calculate the transformation Equation (3) pointwise for a set of sample points in the state space [11]. For each sample point, a local linear transformation

$$\begin{aligned} \vec{x} &= \mathbf{F}_r \cdot \vec{z}, \\ \vec{z} &= \mathbf{F}_r{}^{-1} \cdot \vec{x} \end{aligned} \tag{8}$$

with the transformation matrix $\mathbf{F}_r \in \mathbb{R}^{n \times n_z}$ can be calculated (see Section 2.3.2). Numerically integrating these local transformations leads to a numerical approximation of the global nonlinear transformation $\vec{F}$ for each system. In Figure 2 a graphical illustration of the transformation process for both systems is shown.

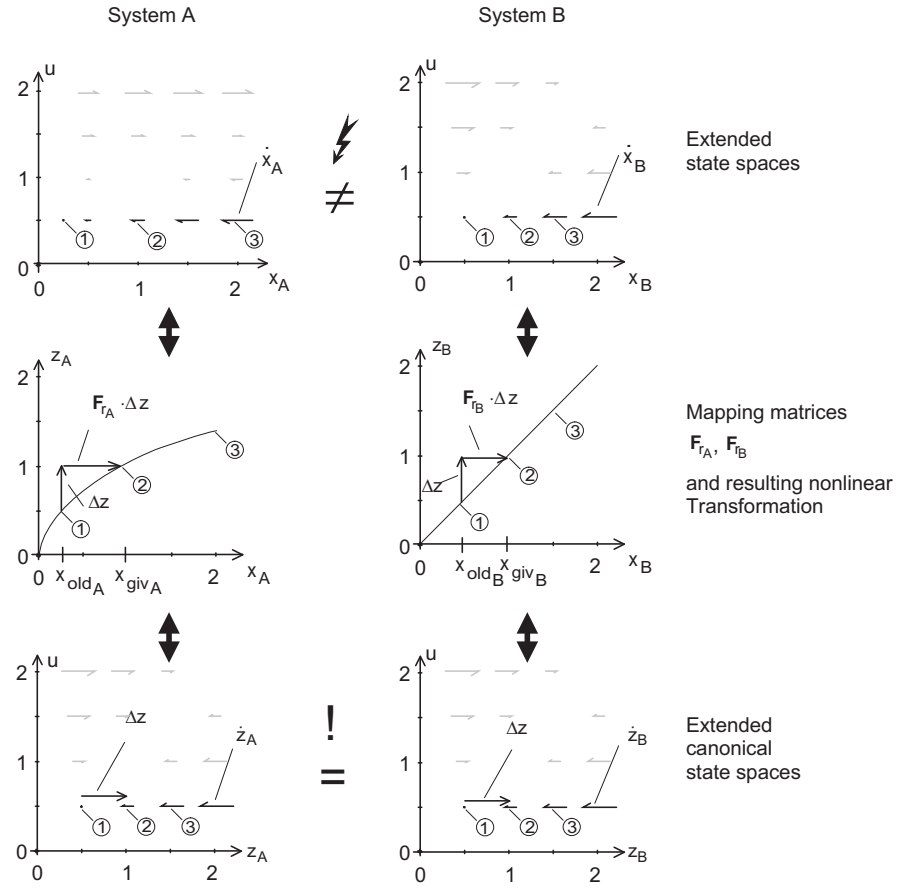


Fig. 2 Construction of the nonlinear mapping.

In the first row of Figure 2, the extended state spaces of both systems with their discrete sample points and corresponding state derivative vectors $\dot{\vec{z}}_A$, $\dot{\vec{z}}_B$ are shown. In the

last row of the Figure, the same information is shown for the, at the beginning unknown, transformed canonical state space. During the algorithm all points in that state space are calculated successively. From each local transformation at each sample point, the overall nonlinear transformation function between the original extended state spaces and the canonical state spaces is collected, as illustrated in the middle row. The numbering 1, 2, 3 for the points in each row represents the sequence of sampling the state space performed by the equivalence checking algorithm.

Starting at point 1 the transformation matrices $\mathbf{F}_{r_A}$ and $\mathbf{F}_{r_B}$ are calculated. Both allow to determine the next sample point 2 being equidistant in the canonical state space and being transformed in the original state spaces by the backward transformation of the step size Δz. The transformation and the step calculation to the next point 2 is illustrated by the arrows in the middle row, resulting in a transformation-dependent sampling grid in the original state space. The sampling step size could be further controlled by changing the predefined Δz value or activating an automatic step size control [13] often used in circuit simulators.

2.3.2 Kronecker Form

In the linear case, the canonical form is the Kronecker canonical form [14,15] in the frequency domain. In order to calculate it, a linearization of the system has to be used. As it is linear it can easily be transformed into the frequency domain using the Laplace transformation. The resulting system can be written as

$$s \cdot \mathbf{C} \cdot \vec{X} + \mathbf{G} \cdot \vec{X} = \mathbf{Q} \cdot \vec{U} \tag{9}$$

where s is the complex damping and frequency variable resulting from Laplace transformation, **C** is the capacitance matrix having in general a low rank and many zero entries, **G** is the conductance matrix, **Q** is an input matrix, and the upper case variable/input vectors $\vec{X}$ and $\vec{U}$ are the linearized variables/inputs valid around the actual linearization point.

It is possible to transform **C** and **G** into Kronecker's canonical form

$$\mathbf{E} \cdot \mathbf{G} \cdot \mathbf{F} = \tilde{\mathbf{G}} = \begin{bmatrix} -\lambda_1 & 0 & \cdots & 0 & \cdots & 0 \\ 0 & -\lambda_2 & & 0 & & 0 \\ \vdots & & \ddots & \vdots & & \vdots \\ 0 & 0 & \cdots & -\lambda_n & & \vdots \\ \vdots & & & & \ddots & 0 \\ 0 & 0 & \cdots & \cdots & 0 & 1 \end{bmatrix} \tag{10}$$

$$\mathbf{E} \cdot \mathbf{C} \cdot \mathbf{F} = \tilde{\mathbf{C}} = \begin{bmatrix} 1 & \cdots & 0 & \cdots & 0 \\ \vdots & \ddots & 0 & & \vdots \\ 0 & 0 & 1 & & \vdots \\ \vdots & & & \ddots & 0 \\ 0 & 0 & \cdots & 0 & 0 \end{bmatrix} \tag{11}$$

with suitably chosen matrices **E** and **F** (see [11]). The number of non-zero entries in $\tilde{\mathbf{C}}$ corresponds to the number n of eigenvalues λ_i in the generalized eigenvalue problem and to the number of independent states. However, some of these states are also a result of parasitic components such as RC-models of a wire, having values in frequency ranges high

above the frequency of interest for the system. Hence, a reasonable approach is to reduce the number of eigenvalues to the non-parasitic ones being in the frequency range of interest of the circuit (as in model order reduction techniques [16,17]). This leads to reduced non-square transformation matrices (see Equation (8))

$$\mathbf{E}_r \cdot \mathbf{G} \cdot \mathbf{F}_r = \tilde{\mathbf{G}}_r = \begin{bmatrix} -\lambda_1 & 0 & \cdots & 0 \\ 0 & -\lambda_2 & & 0 \\ \vdots & & \ddots & \vdots \\ 0 & 0 & \cdots & -\lambda_{n_z} \end{bmatrix}, \mathbf{E}_r \cdot \mathbf{C} \cdot \mathbf{F}_r = \tilde{\mathbf{C}}_r = \begin{bmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & & 0 \\ \vdots & & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \end{bmatrix} \tag{12}$$

with $n_z \leq n$ eigenvalues.

The corresponding reduced system in the canonical state space can now be written as:

$$s \cdot \begin{bmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & & 0 \\ \vdots & & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \end{bmatrix} \cdot \vec{z} + \begin{bmatrix} -\lambda_1 & 0 & \cdots & 0 \\ 0 & -\lambda_2 & & 0 \\ \vdots & & \ddots & \vdots \\ 0 & 0 & \cdots & -\lambda_{n_z} \end{bmatrix} \cdot \vec{Z} = \mathbf{Q}_r \cdot \vec{U}, with \quad \mathbf{Q}_r = \mathbf{E}_r \cdot \mathbf{Q}. \tag{13}$$

In [11] a method for calculating the transformation matrices $\mathbf{E}_r$ and $\mathbf{F}_r$ is proposed. Starting from the generalized eigenvalue problem

$$(\mathbf{C} \cdot \alpha + \mathbf{G} \cdot \beta) \cdot \vec{x} = 0 \tag{14}$$

with the eigenvalues $\lambda = \frac{\alpha}{\beta}$, it uses the matrix $\mathbf{V}_R$ of all right-hand side eigenvectors for the calculation of the transformation matrices:

$$\begin{aligned} \mathbf{F} &= \mathbf{V}_R, \\ \mathbf{E} &= \tilde{\mathbf{G}} \cdot \mathbf{V}_R^{-1} \cdot \mathbf{G}^{-1}. \end{aligned} \tag{15}$$

In order to reduce the system, submatrices of $\mathbf{F}$ and $\mathbf{E}$ have to be chosen. $\mathbf{F}_r$ are the first n_z eigenvectors (columns) of $\mathbf{F}$. $\mathbf{E}_r$ are the first n_z rows of $\mathbf{E}$.

The transformation into the reduced Kronecker's canonical form (13) is performed at each sample point for each system with the same target number of reduced eigenvalues n_z (Step 3 in Figure 1). The resulting transformation matrices $\mathbf{F}_{r_A}$ and $\mathbf{F}_{r_B}$ are used to construct the mapping at each sample point (see Figure 2). Another observation from this technique is that the states corresponding to the reduced eigenvalues together with the input variables span the extended canonical state space (see last row of Figure 2).

3 Prevention of False Negatives

While the presented equivalence checking method is capable of dealing with many circuit implementations, there are special circuit structures causing the transformation not to correctly map the corresponding eigenvalues, consequently leading to false negative results. In the following, this issue is explained and solutions are presented. Furthermore, another important approach for increasing robustness of the equivalence checking method and for preventing false negative results by application of a reachability analysis is introduced.

3.1 Eigenvalue Mapping

The reduction and transformation techniques presented in the preceding section are extremely sensitive to the proper selection of the λ_i's and their proper ordering in the reduced matrix $\tilde{\mathbf{G}}_r$, leading to the same ordering of the corresponding eigenvectors in the transformation matrix $\mathbf{F}_r$.

Up to now, the following method is used: Sorting the eigenvalues according to their magnitude:

$$|\lambda_1| < |\lambda_2| < |\lambda_3| < ... \tag{16}$$

Accordingly, small eigenvalues (corresponding to a low frequency or large time constant) will be the first ones in the list. The disadvantage of this method is the change of the eigenvalue's numerical value in different regions of the state space due to nonlinearities. Hence, in subsequent transformation steps at each point in the state space, different orderings of the state variables will occur, resulting in wrong mappings of state variables between both systems. This leads to false negative equivalence checking results.

Therefore, in this paper we propose two new methods to sort these eigenvalues more accurately, in order to maintain the correct mapping while traversing the state space:

- Sorting eigenvalues by the observability of the dynamics of each considered eigenvalue.
- Sorting eigenvalues according to their structurally corresponding energy storage element.

3.1.1 Observability

If the dynamics of the shifting eigenvalues contribute directly to the output behavior, meaning the dynamics are observable, a relatively easy case is given. As it is known which variable is the output (see Equation 2), we can directly compute a measure for the observability w_i of the dynamic behavior of each eigenvalue λ_i using the corresponding eigenvector $\vec{v}_{R,i}$:

$$w_i = |\mathbf{r} \cdot \vec{v}_{R,i}| \,. \tag{17}$$

Including the observability, the resulting sorting is defined as follows:

$$|\lambda_1 \cdot \frac{1}{w_1}| < |\lambda_2 \cdot \frac{1}{w_2}| < |\lambda_3 \cdot \frac{1}{w_3}| < ... \tag{18}$$

3.1.2 Structural Correspondence

If the states are not observable due to their position inside the circuit and no path exists with enough gain to an output in the actual state, this method fails and the following approach can be applied.

In many cases, the dynamics of an eigenvalue are caused by a hardware structure such as a R-C element in the circuit. As these hardware structures are unchangeable references, it will be useful to have a mapping from the corresponding eigenvalues to these hardware structures. Additionally, this information can help the designer by giving a hint which eigenvalue belongs to which circuit element.

In order to calculate this information, a mapping from eigenvalues to variables in the DAE-system is constructed. Each state variable $x_i^z(t)$ spans a dimension in the state space. In general, more state variables than finite eigenvalues exist. Hence, this onto mapping can be described as follows. Find an eigenvalue to state variable mapping such that each eigenvector

$\vec{v}_{R,k}$ points into the direction of the mapped state variable unity vector $\vec{e}_i$. The latter can be measured by the angle between both vectors:

$$\varphi_{i,k} = \arccos\left(\frac{\vec{e}_i \cdot \vec{v}_{R,k}}{||\vec{e}_i|| \cdot ||\vec{v}_{R,k}||}\right) \tag{19}$$

The overall mapping problem can now be expressed by a combinatorial optimization problem:

$$\begin{array}{ll} \text{Minimize} & K = \sum_{k=1}^{n_z} \sum_{i=1}^{n} \varphi_{i,k} \cdot \chi_{i,k} \\ \text{Subject to} & \forall k : \sum_{i=1}^{n} \chi_{i,k} = 1 \\ & \forall i : \sum_{k=1}^{n_z} \chi_{i,k} \leq 1 \\ & \chi_{i,k} = \begin{cases} 1 \text{ if } x_i \text{ is mapped on eigenvalue } \lambda_k \\ 0 \text{ else} \end{cases} \end{array} \tag{20}$$

The matrix χ with the elements $\chi_{i,k}$ holds the mapping after the optimization.

The combinatorial optimization problem can be very large in terms of eigenvalues. For example, an extracted netlist of a small analog circuit can easily contain 10 to 100 capacitances with corresponding eigenvalues. Hence, the problem is solved by a heuristic branch and bound, where some bounds will be estimated, resulting in reasonable runtimes even for large problems.

Once the mapping for each circuit is obtained, the eigenvalues are ordered corresponding to a reference ordering, which will be computed in advance with one of the methods mentioned above:

$$\lambda_{1,\text{ref}}, \lambda_{2,\text{ref}}, \ldots, \lambda_{n_z,\text{ref}}\,. \tag{21}$$

Using a reference run of the minimization problem (20), the eigenvalues are mapped with the resulting mapping $\chi_{i,j,\text{ref}}$ onto the state variables:

$$\lambda_{1,\text{ref}} \rightarrow \chi_{i,1,\text{ref}} = 1 \rightarrow x^z_{i,\text{ref}},\ \lambda_{2,\text{ref}} \rightarrow \chi_{j,2,\text{ref}} = 1 \rightarrow x^z_{j,\text{ref}}, \ldots,\ \lambda_{n_z,\text{ref}} \rightarrow \chi_{k,n_z,\text{ref}} = 1 \rightarrow x^z_{k,\text{ref}}$$

Subsequently, in each sample point the minimization problem (20) is solved to get an actual mapping. Using the resulting mapping of eigenvalues to state variables reversely, the eigenvalues can be sorted according to their state variables:

$$\begin{array}{c} \chi_{i,1,\text{ref}} = 1 \Rightarrow \exists \lambda_{p,\text{act}} : \chi_{i,p,\text{act}} = 1\,, \\ \chi_{j,2,\text{ref}} = 1 \Rightarrow \exists \lambda_{q,\text{act}} : \chi_{j,q,\text{act}} = 1\,, \\ \ldots\,, \\ \chi_{k,2,\text{ref}} = 1 \Rightarrow \exists \lambda_{r,\text{act}} : \chi_{k,r,\text{act}} = 1 \end{array} \tag{22}$$

resulting in a map

$$\lambda_{1,\text{ref}} \leftrightarrow x^z_{i,\text{ref}} \leftrightarrow \lambda_{p,\text{act}},\ \lambda_{2,\text{ref}} \leftrightarrow x^z_{j,\text{ref}} \leftrightarrow \lambda_{q,\text{act}},\ \ldots,\ \lambda_{n_z,\text{ref}} \leftrightarrow x^z_{k,\text{ref}} \leftrightarrow \lambda_{r,\text{act}},$$

finally resulting in the ordering

$$\lambda_{p,\text{act}}, \lambda_{q,\text{act}}, \ldots, \lambda_{r,\text{act}} \tag{23}$$

of the actual eigenvalues, which may be different from magnitude ordering or observability ordering.

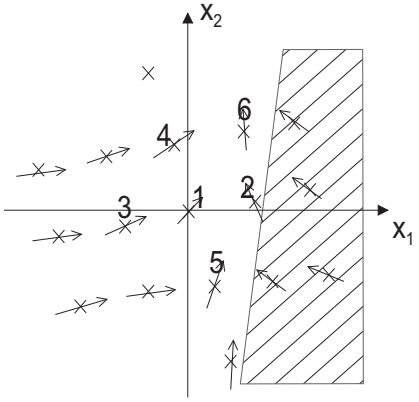


Fig. 3 Sequence of sample point calculations of a state space starting from a DC operating point at the origin. An unreachable region is hatched.

3.2 Reachability Analysis

Another reason for false negatives in equivalence checking can be posed by reachability issues. During exploration of the canonical state space, both systems move point-symmetric in outer regions of the state space as illustrated by the sequence of numbered sample points in Figure 3. At a certain point, they will discover regions of the state space which are not reachable from DC operating points. In those unreachable regions, both systems are allowed to have differing behavior as this behavior will never be seen at the outputs, although comparing the complete state spaces will report high errors. Unfortunately, this scenario is often given when behavioral descriptions of analog circuits are compared with transistor netlist descriptions.

An easy way to prevent this kind of false negatives is to incorporate a reachability analysis of the calculated points in the state space. A successor/predecessor relation for the states is required for reachability analysis. A successor for each sample point can be calculated by virtually searching for the nearest sample point in the direction of the state derivatives $\dot{\vec{x}}$. We use the following criteria:

$$\begin{aligned} \rho_{i,k} &= \arccos\left(\frac{\dot{\vec{x}}_i \cdot \dot{\vec{x}}_k}{||\dot{\vec{x}}_i|| \cdot ||\dot{\vec{x}}_k||}\right) &&< 30^\circ \\ \mathrm{dist}_{i,k} &= ||\vec{x}_i - \vec{x}_k|| &&< \sqrt{2} \cdot ||\Delta \vec{x}||\,, \\ \mathrm{cost}_{i,k} &= \rho_{i,k} \cdot \mathrm{dist}_{i,k} \end{aligned} \tag{24}$$

where $\rho_{i,k}$ is the angle to the potential successor, $\mathrm{dist}_{i,k}$ is the distance to the successor and $\Delta \vec{x}$ is the euclidian distance of the underlying grid in state space. From all potential successors fulfilling this criteria, the one with the smallest $\mathrm{cost}_{i,k}$ will be selected. A similar measure is chosen for the calculation of successors due to input changes. With this successor structure, a standard reachability analysis can be started from the DC operating points, which are reachable by definition. In Figure 4 this algorithm is described.

```
Reachability ()
  Φ_reach = Φ_pend = set of DC operating points
  while Φ_pend ≠ ∅
    Φ_new =Compute Successors(Φ_pend)
    Φ_reach = Φ_reach ∪ Φ_new
    Φ_pend = Φ_new \ Φ_pend \ Φ_reach
  end while
  return Φ_reach
```

Fig. 4 Algorithm for reachability analysis.

4 Results

The proposed approach was implemented in C++ and directly coupled with the circuit simulator Titan (Titan Simulator, Qimonda AG, Munich, Germany), offering the possibility of handling large analog circuits with up to hundreds of transistors. An interface between the verification algorithm and the simulator passes DC operating point and sample point information in terms of variable values $\vec{x}$ and matrix values of the **G** and **C** matrices from and to the simulator. An industrial example of a NAND gate circuit is investigated using several runs of the equivalence checking approach with and without the new eigenvalue mapping strategies and reachability analysis. In addition, the obtained results are compared to those of an active bandpass filter and a mixer circuit, representing two other circuit classes. The following results were processed on a 1.5 GHz Pentium M with 1.5 GB memory.

4.1 NAND Gate

An industrial NAND gate shown in Figure 5 is chosen as an example. It has a strongly nonlinear behavior and represents a class of small library cells for the digital world, modeled at the analog netlist level. The goal is to compare this model with one extracted from the layout of the circuit.

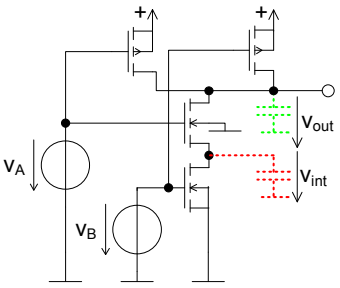


Fig. 5 NAND gate: green: parasitic capacitance at the output, red: parasitic capacitance at the floating node of stacked transistors.

The extended state space of the circuit is spanned by both input voltages $V_{A/B}$ and the voltages V_{out} and V_{out} across the parasitic capacitances, which were in fact the transistor drain-bulk capacitances. In the extracted circuit, some more capacitances from internal nodes to ground are present.

The circuit has $n = 13$ variables with 6 state variables in non-reduced form. The canonical state space is built for two cases:

- Z1: Only one canonical state variable $n_z = 1$ (the voltage over C_{out}, known from structural mapping);
- Z2: Two canonical state variables $n_z = 2$ (the voltage over C_{out} and C_{int});

In Figure 6 the reachable state space for the case Z1 is shown in the original variable space $\vec{x}$. Only one state variable is present and the input B is fixed in order to show a 2-dimensional plot. For a better overview, only the reachable states are shown while the unreachable ones are outside the shown area.

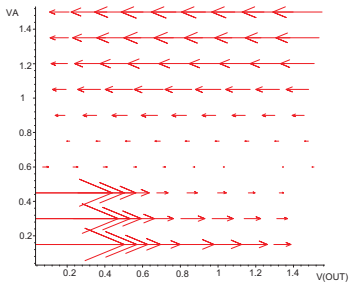


Fig. 6 Extended state space of NAND gate with one state in canonical state space and voltage $B = 1.5V$.

From inspection, the circuits are expected to show a similar behavior with small deviations around 1% - 10%, because added capacitors from extraction are in that range. Hence, the relative errors have to be low, which means below 10%.

For the case Z1, we have calculated the errors for all combinations of the mapping methods. The results are given in Table 1. Column 3 and 4 show the number of calculated and reachable state points, column 5 to 7 reflect the methods used, and in column 8 and 9 the relative errors of the derivatives (see Equation (5)) and the outputs (see Equation (6)) are given.

It can clearly be stated that one has to use at least the observability method (Obs.; see Section 3.1.1) or the structural mapping (Struct.; see Section 3.1.2) to obtain the low errors expected. The combination of both, meaning that the reference ordering of the structural mapping (see Equation (21)) is performed using the observability method, does not improve the results further. However, it will be a safer choice.

The reachability analysis does not show any further improvement but clearly reduces the number of points being considered for error calculation.

Moreover, this circuit could also be modeled with two states in the canonical state space, resulting in slightly higher accuracy. The capacitor C_{int} is responsible for the additional state. Two slices of the 4-dimensional extended state space are shown in Figure 7. One can identify the DC operating point (fixpoint) in the upper right corner of Figure 7 a) and in the lower left corner of Figure 7 b). They represent the logical 0 or the 1 at the output for both input combinations.

The whole set of the state space calculated without a reachability analysis is shown in Figure 8 a), while the reachable set is shown for the same state space (NAND, case Z2) in Figure 8 b). As a result from some aborted points and mainly as a result from the nonlinear backward transformation from the canonical state space, the sampled points of the whole

Example	States	Points		Method			rel. Error		Runtime
	n_z	calc	reach	Obs.	Struct.	Reach	Dyn. $\varepsilon_{\dot{z}}$	Out. ε_y	
NAND #1	1	524	n/a	no	no	no	40.9%	5.4%	1.6s
NAND #2	1	540	n/a	yes	no	no	0.58%	0.42%	1.35s
NAND #3	1	540	n/a	no	yes	no	2.1%	0.15%	1.34s
NAND #4	1	540	n/a	yes	yes	no	0.58%	0.42%	1.35s
NAND #5	1	540	215	yes	yes	yes	0.7%	0.08%	1.58s

Table 1 Comparison of different methods for NAND-circuit with 1 canonical (case Z1).

set are not on an equidistant grid and do not cover the whole state space (Figure 8 a)). Nevertheless, as expected, the reachable space is considerably smaller and reasonably fits into the ranges of the supply voltages.

Inspecting the results for the NAND in Table 2, one can notice that - as expected - the structural mapping is mandatory (row NAND #25 and NAND #26). Comparing row NAND #22 and NAND #26, one can conclude that also the reachability is important, further reducing the error by orders of magnitude.

The different number of reachable points (row NAND #23 - #26) is a result from performing the reachability analysis in the canonical state space. Depending on the mapping method, different point locations in the canonical state space are leading to slightly different results in the reachability analysis.

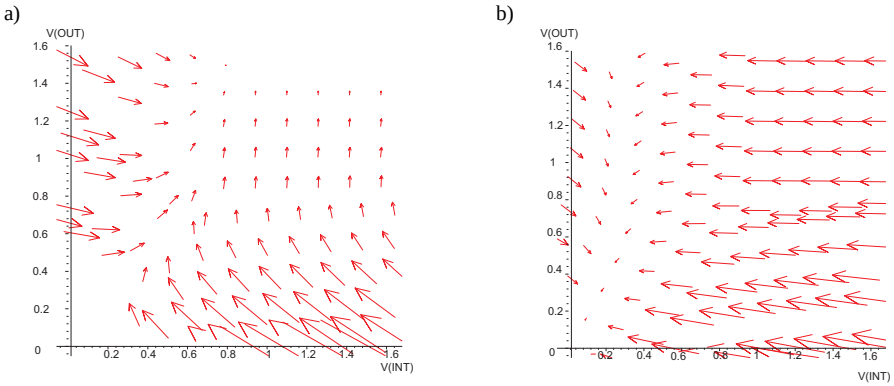


Fig. 7 State space of NAND gate with 2 states in canonical state space at voltages a) $A = 0V$, $B = 1.5V$ b) $A = 1.5V$, $B = 1.5V$.

a) b)

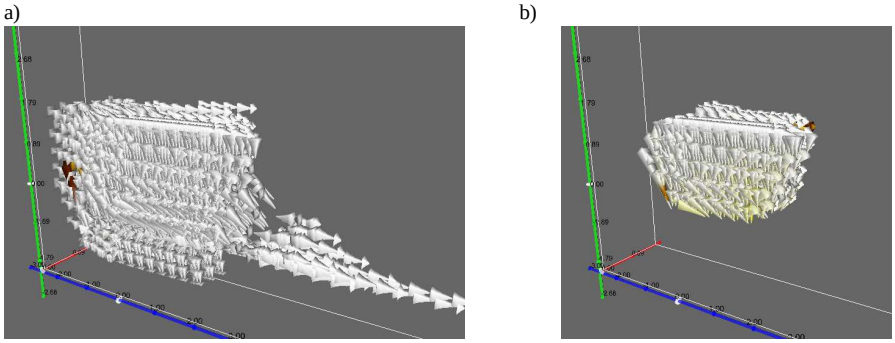


Fig. 8 State space of NAND gate with 2 states in canonical state space: a) all points b) reachable points. Visible are the calculated transitions between two state points. Axes are: red: input A, green: V_{out} and blue: V_{int}. The picture is a projection on input B.

4.2 Bandpass Filter and Mixer

A bandpass circuit as given in Figure 9 a) is compared with an abstract VHDL-AMS description of the bandpass transfer function partially given in Figure 9 b). Inspecting the results for the bandpass (BP) in Table 2, a large difference between using the reachability algorithm or not can be noticed. Due to the nonlinear characteristics of the transistors in the netlist, the output and the internal states of the bandpass circuit are limited essentially by the supply voltage. In contrast, the linear behavioral model does not exhibit this limiting behavior. However, due to the limited input swing, the output and internal states of the bandpass will not reach this limitation area and therefore stay in the linear ranges. This restriction can only be identified by using the reachability algorithm and is reflected exactly by the results in Table 2. It is clearly shown that reachability analysis in this case is mandatory for the comparison of a linear behavioral model with a nonlinear netlist, leading to the correct low error values.

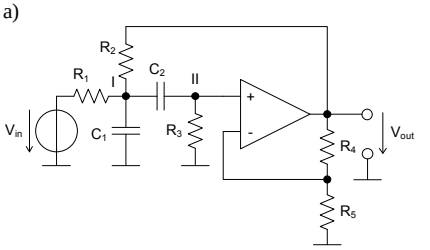
The next example is a mixer circuit with inherent nonlinear behavior as is illustrated in Figure 9 c). Again, it is compared with a nonlinear behavioral description in VHDL-AMS. In contrast to the previous implementation, this behavioral description is also modeling the nonlinear limiting behavior and therefore the results do not differ very much. The reachability algorithm leads to longer runtimes but correctly excludes some areas with slightly larger deviations.

All examples prove that the proposed methods prevent false negatives for these different kinds of circuits.

5 Conclusion

In recent years, equivalence checking was successfully introduced into the verification flow for digital circuits, making possible an almost fully automated verification. For analog circuits, equivalence checking will also contribute to an automatization of the design flow by allowing the formal verification of a behavioral model or an extracted netlist versus a nominal transistor netlist; a procedure of high importance for seamless transition between design abstraction levels.

a)



b)

```
ENTITY bandpass IS
  GENERIC( den0 : real := ... );
  PORT( terminal inp, outp,
        gnd : electrical );
end BANDPASS;

ARCHITECTURE behave OF bandpass IS
  QUANTITY uout ACROSS iout THROUGH
           outp TO gnd;
  ...
BEGIN
  uint == 1E-3*i_uint'dot +4E-7*i_uint;
  uint == 1.0/den1*(-den0*0.001*i_uint
          - den2*uint'dot
          + num1*uin);
  uout == uint+Rout*iout;
  ...
END behave;
```

c)

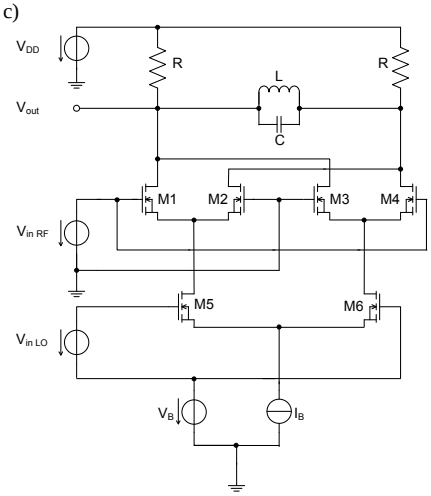


Fig. 9 Circuit schematic a) and behavioral model b) of the bandpass filter. Circuit schematic of the mixer c).

In this paper, a method for handling strongly nonlinear circuits such as digital library cells was introduced, representing a circuit class that exhibits nodes with difficult dynamic behavior. The equivalence checking method depends on correct mapping of these dynamics given by the eigenvalues and eigenvectors of the circuits. Accordingly, two new approaches were proposed, which in combination are now able to calculate a correct mapping.

In order to prevent the equivalence checking method from detecting large differences (false negatives) in unreachable parts of the state space, a reachability analysis was presented in addition. For a strongly nonlinear example, the combination of structural mapping and reachability analysis presents a reliable result. Furthermore, it supports the designer in finding the origin of relevant differences between the two circuits under verification.

6 Acknowledgments

The authors would like to thank Georg Denk and Carsten Hammer (Titan Simulator Development, Qimonda AG, Munich, Germany) for very fruitful discussions and their suggestions to this work. The Electronic Design Methodology Group of the University of Frankfurt is a subcontractor of Qimonda AG in the VeronA research project.

Example	States	Points		Method			rel. Error		Runtime
	n_z	calc	reach	Obs.	Struct.	Reach	Dyn. $\varepsilon_{\dot{z}}$	Out. ε_y	
NAND #21	2	7801	n/a	no	no	no	200%	15.1%	21.2s
NAND #22	2	8120	n/a	yes	yes	no	199%	152.1%	111.8s
NAND #23	2	7830	2432	no	no	yes	200%	168%	121.1s
NAND #24	2	8121	4425	yes	no	yes	200%	145%	113.0s
NAND #25	2	8100	2792	no	yes	yes	15.1%	0.13%	110.9s
NAND #26	2	8019	1984	yes	yes	yes	8.5%	0.09%	111.8s
BP #1	2	6615	n/a	no	no	no	200.0%	193.6%	24.7s
BP #2	2	8379	n/a	yes	no	no	200.0%	200.0%	34.0s
BP #3	2	6615	n/a	no	yes	no	200.0%	196.5%	24.7s
BP #4	2	6867	n/a	yes	yes	no	200.0%	200.0%	27.2s
BP #5	2	6615	1385	no	no	yes	0.216%	0.103%	78.2s
BP #6	2	8379	1385	yes	no	yes	0.216%	0.103%	120.2s
BP #7	2	6615	1385	no	yes	yes	0.216%	0.103%	78.4s
BP #8	2	6615	1385	yes	yes	yes	0.216%	0.103%	79.6s
Mixer #1	2	9801	n/a	no	no	no	0.087%	0.031%	42.1s
Mixer #2	2	9801	121	yes	yes	yes	0.0%	0.025%	144.6s

Table 2 Comparison of different methods for circuits with 2 canonical states.

References

1. G.G.E. Gielen. Cad tools for embedded analogue circuits in mixed-signal integrated systems on chip. *Computers and Digital Techniques, IEE Proceedings* -, 152(3):317–332, May 2005.
2. T.A. Henzinger and P.-H. Ho. Algorithmic Analysis of Nonlinear Hybrid Systems. *CAV '95: International Conference on Computer-Aided Verification, LNCS*, 939(7):225–238, 1995.
3. T. Dang, A. Donze, and O. Maler. Verification of analog and mixed-signal circuits using hybrid system techniques. *FMCAD 04: Formal Methods in Computer-Aided Design*, 2004.
4. Ghiath Al-Sammane, Mohamed H. Zaki, and Sofiène Tahar. A symbolic methodology for the verification of analog and mixed signal designs. *DATE '07: Proceedings of the conference on Design, automation and test in Europe*, pages 249–254, 2007.
5. Goran Frehse, Bruce H. Krogh, and Rob A. Rutenbar. Verifying analog oscillator circuits using forward/backward abstraction refinement. *DATE '06: Proceedings of the conference on Design, automation and test in Europe*, pages 257–262, 2006.
6. W. Hartong, L. Hedrich, and E. Barke. On Discrete Modeling and Model Checking for Nonlinear Analog Systems. *CAV '02: International Conference on Computer-Aided Verification, LNCS*, 2404:401–413, 2002.
7. Tathagato Rai Dastidar and P. P. Chakrabarti. A verification system for transient response of analog circuits using model checking. *VLSID '05: International Conference on VLSI Design*, 00:195–200, 2005.

8. D. Grabowski, M. Olbrich, Ch. Grimm, and E. Barke. Analog Circuit Simulation Using Range Arithmetics. *ASPDAC*, pages 762–767, 2008.
9. M.H. Zaki, S. Tahar, and G. Bois. Formal Verification of Analog and Mixed Signal Designs: Survey and Comparison. *IEEE North-East Workshop on Circuits and Systems*, pages 281–284, 2006.
10. L. Hedrich and E. Barke. A Formal Approach to Nonlinear Analog Circuit Verification. *ICCAD '95: International Conference on Computer Aided Design*, pages 123–127, 1995.
11. W. Hartong, R. Klausen, and L. Hedrich. Formal Verification for Nonlinear Analog Systems: Approaches to Model and Equivalence Checking. *Advanced Formal Verification, R. Drechsler, ed., Kluwer Academic Publishers, Boston*, pages 205–245, 2004.
12. C.W. Ho, A.E. Ruehli, and P.A. Brennan. The Modified Nodal Approach to Network Analysis. *IEEE Transactions on Circuits and Systems*, 22(6):504–509, 1975.
13. L. Hedrich and W. Hartong. Approaches to formal verification of analog circuits. In P. Wambacq, editor, *Low-Power Design Techniques and CAD Tools for Analog and RF Intergrated Circuits*, pages 155–191. Kluwer Academic Publishers, Boston, 2001.
14. R. März. Numerical Methods for Differential Algebraic Equations. *Acta Numerica*, pages 141–198, 1991.
15. P. Van. Dooren. The Computation of Kronecker's Canonical Form of a Singular Pencil. *Journal on Linear Algebra and its Applications*, 27:103–140, 1979.
16. L. Fortuna, G. Nunnari, and A. Gallo. Model Order Reduction Techniques with Applications in Electrical Engineering. *Springer-Verlag, Berlin*, 1992.
17. P. Feldmann and R.W. Freund. Efficient Linear Circuits Analysis by Pade Approximation via the Lanczos Process. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 14(5):639–649, 1995.