Automatic Abstraction of Analog Circuits to Hybrid Automata

Ahmad Tarraf and Lars Hedrich
Institute for Computer Science
Goethe Universität Frankfurt a. M., Germany
Email: {tarraf, hedrich}@em.cs.uni-frankfurt.de

Abstract

Formal verification of analog transistor level circuits is still an open problem. In this paper we present a methodology which automatically abstracts a very accurate sampled analog circuit block using a reachability tool with SPICE-accuracy. The resulting hybrid automata (HA) models the – in most analog circuits desired – linear behavior, but also the technology dependent nonlinear and limiting behavior. We present several abstraction variants and evaluate the accuracy of the model using some simulations of the SPICE-netlist and the HA. Finally, we show that it can be used for formal verification on high level with reachability tools.

1 Introduction

Safety critical systems like autonomous driving or medical devices demand for powerful verification methodologies. Formal verification offers this opportunity. However for analog circuits and systems formal verification suffers from complexity and nonlinearity of underlying equations in transistor models. However at higher abstraction levels the analog circuits often behave as nearly linear blocks or with a known wanted nonlinearity or limiting function. For this high level behavior a common useful modeling technique are hybrid systems ([1, 2, 3, 4]). The methods are able to handle up to 20 state variables, if the underlying locations use linear ordinary differential equations (ODEs). Mostly they use reachability analysis to prove safety properties. However, monitor based approaches can be used to prove properties written in LTL or CTL [5].

In this paper we propose a methodology to close the gap between transistor level analog circuits and hybrid automata (HA) by automating an abstraction process. That means we are generating in a controlled way HA-models:

- the method identifies from a sampled state space of a nonlinear transistor level circuit few linear regions
- the regions are clustered and transformed into a hybrid automata
- mathematically a back-transformation is constructed to evaluate the results in the original variable space

2 Previous Work

The problem of generating an abstract model for a transistor level circuit is very old and comes in several flavors, e.g. as automatic behavioral modeling [6, 7]. These techniques are not targeting hybrid automata. They are mainly improving the simulation speed. The method in [8] models the underlying DAE-system of electrical networks using piecewise linear regions for each nonlinear element on the fly. It suffers from using an abstract transistor model and a limited number of transistors to be verified. However it generates a complex hybrid automata on the fly preventing a state explosion problem at initialization and during evaluation of a given input stimulus. Unfortunately the HA is very complex as it is a cross product of all linearized regions of all nonlinearities.

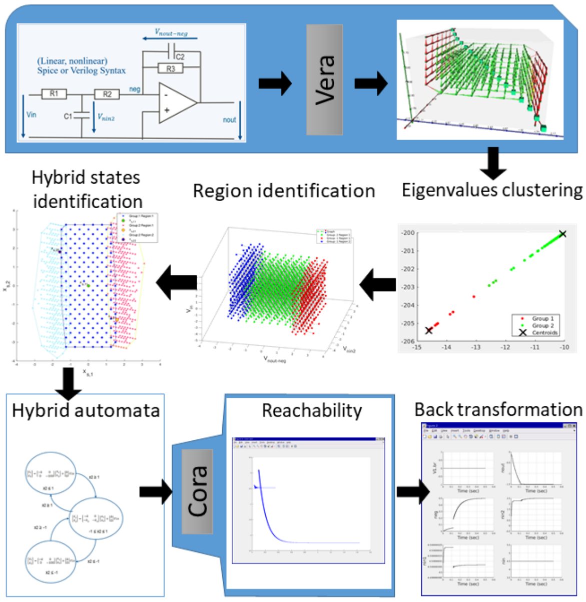

Fig. 1 Overview of the introduced approach

Therefore an abstract linear region is not likely to be identified during the run of this method.

In our approach we want to use a more global view on the block taking all nonlinearities at transistor level into account and directly look for large nearly linear regions in the state space of the block. We rely on a formal verification tool sampling with full BSIM accuracy on transistor level called Vera originally used for equivalence checking [9].

This Vera method samples a full state space of one or two circuits under verification using an on the fly reachability technique. The information about the sampled state space contains

- the dynamics in a directed graph which models the timing behavior,
- the linearized conductance and capacitance matrices: $\underline{G}$, $\underline{C}$,
- the operating point $\vec{x}$ and
- the eigenvalues λ_i, right and left eigenvectors $\underline{F}$, $\underline{E}$

for each point. These data is saved in an own special file format to be exchanged between different verification and visualization tools.

3 Overview

An outline of the introduced approach is presented in Fig. 1. The approach starts from a sampled state space. From these points, states of the hybrid automata are found. This is done first by clustering the points according to their eigenvalues. These groups of same eigenvalues are separated into regions using the spatial proximity in the state space. These few regions make up the states of the generated hybrid automata. Finally for each region guards and invariants are calculated . The hybrid automata at hand can now be used for reachability analysis or other formal techniques. The result from the reachability analysis is than transformed back to the original state space.

4 Abstraction of Analog Behavior

4.1 State Space Sampler

As stated, the state space sampler Vera linearized the nonlinear transistor level circuit in each point of the state space to a SISO[1] system described by the following equation:

$$\begin{aligned} \underline{C}\cdot\dot{\vec{x}}+\underline{G}\cdot\vec{x} &= \vec{b}\cdot u \\ y &= \vec{r}^T\cdot\vec{x} \end{aligned} \tag{1}$$

with the input vector $\vec{b}$, the input u, the output vector $\vec{r}^T$ and the output y. The linearized system can be transformed into a canonical state space $\vec{x}_s$ as described in [9]:

$$\begin{aligned} s\cdot\underline{E}\cdot\underline{\tilde{C}}\cdot\underline{F}\cdot\vec{x}_s+\underline{E}\cdot\underline{\tilde{G}}\cdot\underline{F}\cdot\vec{x}_s &= \underline{E}\cdot\tilde{\vec{b}}\cdot u \\ y &= \tilde{\vec{r}}^T\cdot\underline{F}\cdot\vec{x}_s. \end{aligned} \tag{2}$$

with the transformation matrices $\underline{F}$ and $\underline{E}$ and the transformed matrices and vectors marked by a tilde $\tilde{\cdot}$. $\underline{F}$ are the right eigenvectors of the corresponding generalized eigenvalue problem, $\underline{E}$ is a proper calculated matrix also from that problem (see [9]). Expanding and grouping the transformed equation system leads to a Kronecker form:

$$\begin{aligned} s\cdot\begin{bmatrix}\underline{I} & \underline{0}\\ \underline{0} & \underline{0}\end{bmatrix}\begin{bmatrix}\vec{x}_{s,Re}\\ \vec{x}_{s,\infty}\end{bmatrix}+\begin{bmatrix}\underline{\lambda} & \underline{0}\\ \underline{0} & \underline{I}\end{bmatrix}\begin{bmatrix}\vec{x}_{s,Re}\\ \vec{x}_{s,\infty}\end{bmatrix} &= \begin{bmatrix}\tilde{\vec{b}}_{Re}\\ \tilde{\vec{b}}_{\infty}\end{bmatrix}\cdot u \\ y &= \left[\tilde{\vec{r}}^T_{Re}\tilde{\vec{r}}^T_{\infty}\right]\begin{bmatrix}\vec{x}_{s,Re}\\ \vec{x}_{s,\infty}\end{bmatrix} \end{aligned} \tag{3}$$

where $\underline{I}$ is the identity matrix, $\underline{\lambda}$ is a diagonal or band-diagonal matrix with numerically increasing generalized eigenvalues of the original system. The system is separate into a reduced part (subscript *Re*) and a to be neglected part (subscript ∞). By specifying a reduction order in Vera and using dominant pole order reduction , equation (3) can be further reduced to equation (4):

$$\begin{aligned} s\cdot\begin{bmatrix}\underline{I} & \underline{0}\\ \underline{0} & \underline{0}\end{bmatrix}\begin{bmatrix}\vec{x}_{s,Re}\\ \vec{x}_{s,\infty}\end{bmatrix}+\begin{bmatrix}\underline{\lambda}_{red} & \underline{0}\\ \underline{0} & \underline{I}\end{bmatrix}\begin{bmatrix}\vec{x}_{s,Re}\\ \vec{x}_{s,\infty}\end{bmatrix} &= \begin{bmatrix}\tilde{\vec{b}}_{Re}\\ \tilde{\vec{b}}_{\infty}\end{bmatrix}\cdot u \\ y &= \left[\tilde{\vec{r}}^T_{Re}\tilde{\vec{r}}^T_{\infty}\right]\begin{bmatrix}\vec{x}_{s,Re}\\ \vec{x}_{s,\infty}\end{bmatrix} \end{aligned} \tag{4}$$

where the rank of $\underline{\lambda}_{red}$ indicates the order specified in Vera. We build the HA using the reduced part with the transformed reduced state variables $\vec{x}_{s,Re}\in\mathrm{IR}^{order}$. In the following we will write $\vec{x}_s$ as an abbreviation for the reduced state vector $\vec{x}_{s,Re}$.

4.2 Identifying Linear Regions in Sampled State Space Data

The approach at hand is to identify the linear regions in the reachable state space using eigenvalues. As eigenvalues characterize a linear system, they are suitable for this task. In order to make use of an optimized Matlab clustering algorithms, a parser will be used to translate the output of the sample space discretizer Vera to a Matlab compatible syntax. To bring this approach closer to the reader, a running example will be used throughout this paper.

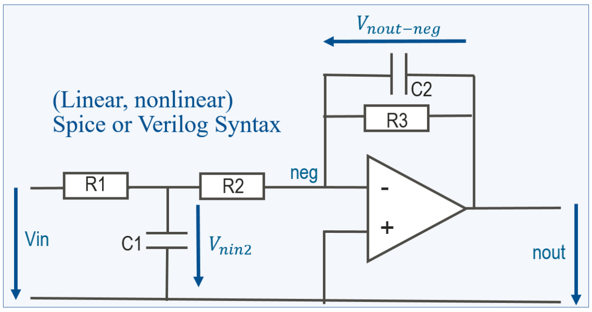

Fig. 2 Second order low pass filter with limitation of the output at 1.5 V

Consider the circuit shown in Fig. 2. The circuit is passed in spice syntax to Vera. The result from the sampling process is shown in Fig. 3. It contains all reachable points in the state space spanned by $x_{s1}=V_{nout}-V_{neg}$, $x_{s2}=V_{nin2}$ and V_{in}.

The large green points indicate DC points. The transitions between the different points of this space are also shown as

[1] For easy of reading we use SISO system here. However the method is also able to handle MIMO systems.

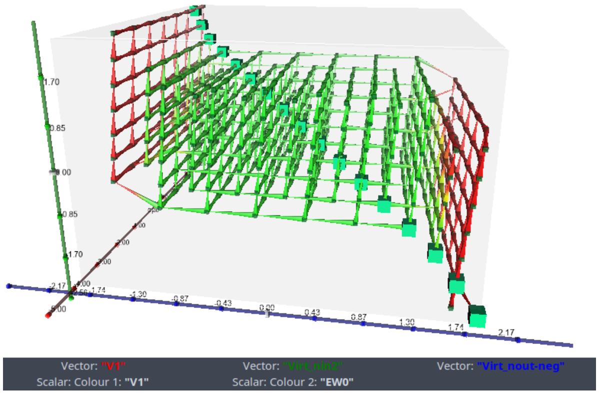

Fig. 3 Discretization of the x state space performed by Vera

edges. The colors in the figure are proportional to the values of the eigenvalues. As seen, the eigenvalues seem like good candidates for the identification of 3 linear regions in this discretization (red – green – red). The green colored region correspond to the desired linear lowpass behavior. The red colored regions correspond to a nonlinear system behavior change due to a limiting function in the operational amplifier.

4.3 Clustering using Eigenvalues

Using a k-means algorithm on the eigenvalues, the system is clustered into groups of nearly equal eigenvalues. It should be noted that both, real and imaginary parts, of each eigenvalue is considered in this clustering. The basic algorithm of k-means is extended using the silhouette coefficient. Continuing the example of the netlist in Fig. 2, the clustering in Fig.4 and Fig.5 are obtained, which yields a system shown in Fig. 6.

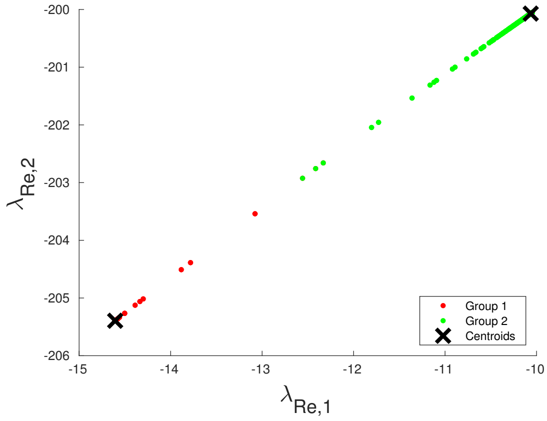

Fig. 4 The real part clustering of the eigenvalues. Since the system from Fig. 2 has only real eigenvalues, the imaginary part have no importance here

The next step is to separate the regions inside a group because numerical similar eigenvalues can by spatially far away. Three approaches have been used to handle this

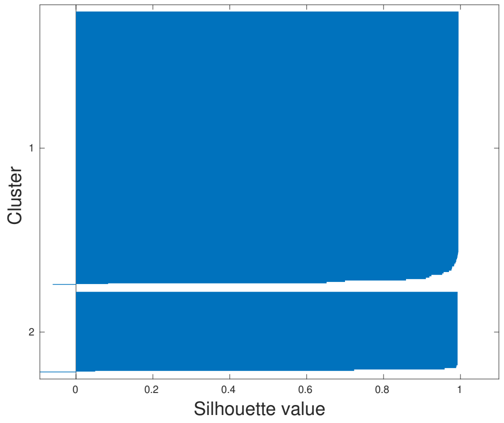

Fig. 5 The silhouette plot of the eigenvalues. As illustrated, almost all eigenvalues have a silhouette coefficient close to 1. For the 1619 points sampled for this example, only 12 have a silhouette coefficient smaller than 0.8

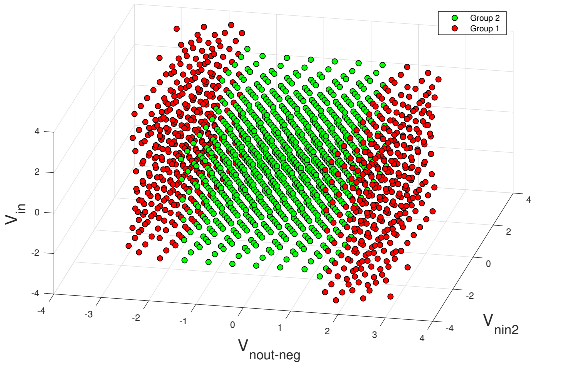

Fig. 6 State clustering using eigenvalues. Two groups can be identified using the k-means algorithm

problem.

The first approach uses k-means a second time, with the difference that instead using the eigenvalues the state variables $\vec{x}_s$ are used. This approach suffers from the fact that it depends on the discretization step size set in Vera. Thus if the groups are not wide enough, unacceptable clustering might occur.

The second and third approaches are both graph based methods that make use of the transitions identified by Vera to overcome this problem.

The second approach uses a breadth-first search on the graph created by Vera, identifying the regions inside a group by checking if they are connected. Considering our previous example, the result of this approach is illustrated in Fig. 7

This approach overcomes the problem of the first approach by making use of the discretization graph. On the other hand, this approach is time consuming, especially when large amounts of points are sampled by Vera. This fact comes from the drawback that all point are covered and a linked list structure is used to save the information.

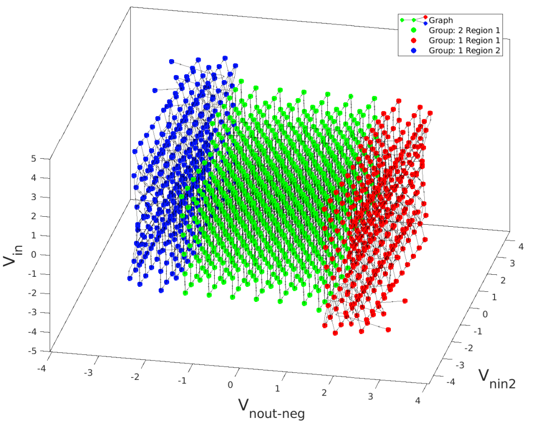

Fig. 7 State clustering using k-means and a breadth-first approach. Two groups can be identified using the k-means algorithm. The first group reveals to have 2 regions

Another major drawback results when few points are not connected at all, or small regions of negligible size are formed due to numeric errors. To overcome this problem, a third approach will be introduced.

The third approach as stated, is also a graph based one. More over, this approach makes use of the connection graph created by Matlab. For our example, Fig. 8 shows the connection graph of group 1. As seen, the two graphs are not connected. Thus, there are 2 regions inside this group. There is also only one region in group 2. Hence we get also 3 different regions.

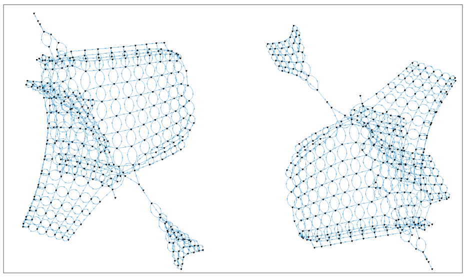

Fig. 8 Plot of the connection graph in each group. There are no connections between the 2 regions of group 1.

This approach solves the problem of small negligible regions, by merging them into larger ones. This is done by comparing the distances between the different points of the different regions to the discretization step size.
Each *region* identified will later become a *location* in the generated HA.

4.4 Calculating the Reference Points for the HA

The operating points for each regions have to be identified next for further calculations. This has to be done in both state spaces: $\vec{x}$ and $\vec{x}_s$. More over, the operating points $\vec{x}_{s,DC}$ in the $\vec{x}_s$ domain are calculated from the operating points in the $\vec{x}$ domain. From the given DC points in the $\vec{x}$ space, the closest one in each region to the surrounding regions is identified as $\vec{x}_{DC}$ and u_{DC}. Then, after identifying the region enclosing the center, the remaining points are recursively calculated by solving the following equation for $\vec{x}_{s,DC,k+1}$:

$$\begin{aligned}\underline{F}_{Re,k}\vec{x}_{s,DC,k+1} = \vec{x}_{DC,k+1} - \vec{x}_{DC,k} + \underline{F}_{Re,k}\vec{x}_{s,DC,k} \\ + \underline{F}_{\infty,k}\underline{E}_{\infty,k}\vec{b}(u_{DC,k} - u_{DC,k+1})\end{aligned} \tag{5}$$

Where $\underline{F}_{\infty,k}$ and $\underline{E}_{\infty,k}$ belong to the lower part of eq. (3) and k numbers the regions. This is done starting from the region containing the center $\vec{x}_{s,DC,0}$. Hence, a smaller k index indicates that the region is closer to the center. Note that the subindex "∞"for $\underline{F}$ and $\underline{E}$ represents submatrices corresponding to the neglected and infinity eigenvalues of the generalized eigenvalue problem.

4.5 Guards and Invariants

Based on the operating points found, all points from the original can be transformed into the x_s space. This is done by applying equation (6) to all sampled points :

$$\begin{aligned}\underline{F}_{Re,k}\vec{x}_{s,i} = \vec{x}_i - \vec{x}_{DC,k} + \underline{F}_{Re,k}\vec{x}_{s,DC,k} \\ + \underline{F}_{\infty,k}\underline{E}_{\infty,k}\vec{b}(u_{DC,k} - u_i)\end{aligned} \tag{6}$$

Where i indicates an index of a point in region k. Now that the $\vec{x}_s$ space is at hand, the regions are enclosed by convex hulls. These hulls represent the invariants for each region. What remains now is to calculate the guards. This is done by searching for the presence of points from neighbor regions close to the edges a convex hull. After that, a dominant guard for each neighbor region is selected by choosing the guard that is in proximity to most neighbor points. These guards are specified as halfspaces. For the example at hand, the regions in the $\vec{x}_s$ space are illustrated in Fig. 9. The operating points $\vec{x}_{s,DC}$ are also shown in this figure. The guards are not plotted for simplicity. Group 1 has 2 guards, one to each region of group 2. Each of group 2 regions has one guard to group 1. For each of the colored region, a location is established with a system Matrix in Jordan normal

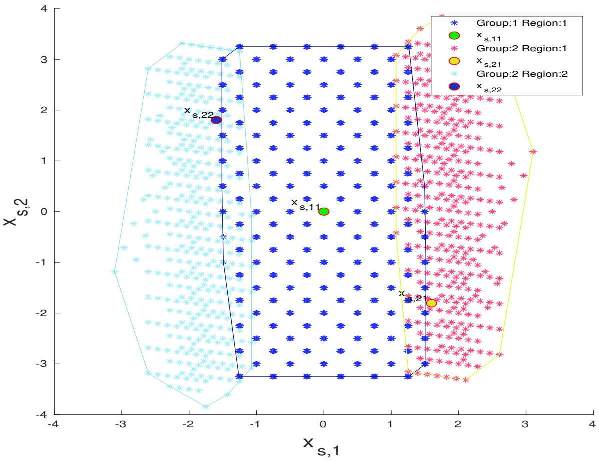

Fig. 9 Regions in the reduced state space $\vec{x}_s$. The different regions are shown in different colors. Convex hulls are used to set the invariant and find the guards

form. For example, if the system has order 3 with 3 distinct real eigenvalues at each location, than the states of the hybrid automata are described as follows:

$$\vec{x}_{hyb,k} = \begin{bmatrix} \lambda_1 & 0 & 0 \\ 0 & \lambda_2 & 0 \\ 0 & 0 & \lambda_3 \end{bmatrix} \vec{x}_{hyb,k} + \underline{E} \cdot \vec{b} \cdot (u - u_{DC,k}) \tag{7}$$

For our example, the state space representations for the first two locations are shown in equation (8) and (9). Location 3 has the same state space representation as location 2 with the difference that $u_{DC,3}$ = -3.5 V.

$$\vec{x}_{hyb,1} = \begin{bmatrix} -10.1 & 0 \\ 0 & -200.2 \end{bmatrix} \vec{x}_{hyb,k} + \begin{bmatrix} 5.1 \\ -99.9 \end{bmatrix} \cdot (u) \tag{8}$$

$$\vec{x}_{hyb,2} = \begin{bmatrix} -14.8 & 0 \\ 0 & -205.4 \end{bmatrix} \vec{x}_{hyb,k} + \begin{bmatrix} 5.2 \\ -97.2 \end{bmatrix} \cdot (u - 3.5) \tag{9}$$

With $\vec{c}$ representing the normal row vector to a halfspace and d representing the offset to the center, such that $\vec{c} \cdot \vec{x}_s \leq d$. E.g. the guards for location 1 are: $\vec{c} = [-1, 0.08]$ with $d = -1.43$ and $\vec{c} = [1, 0]$ with $d = -1.5$. The reset value for x_{hyb} after hitting a guard is equal to the $x_{s,DC}$ of that region. The other guards and invariants are calculated accordingly.

4.6 Back Transformation to the Original State Space

With the created HA, we are able to perform a reachability analysis in the $\vec{x}_s$ space. To better judge the results, the results can be transformed back into the original state space $\vec{x}$. This is done with equation (10)

$$\vec{x} = \underline{F}_{Re,k} \vec{x}_{s,k} + \vec{x}_{DC,k} + \underline{F}_{\infty,k} \underline{E}_{\infty,k} \vec{b} (u - u_{DC,k}) \tag{10}$$

Where k indicates the location of the system. It should be noted that $\vec{x}_{s,k}$ in equation (9) represents a zonotope. Thus, this equation represents a transformation and shifting of the zonotopic solution found by the reachability tool.

4.7 Experimental Results

With the result at hand, we are able to perform a reachability analysis for a hybrid system in Cora. For the example introduced in Fig. 2, the results are illustrated in Fig. 10. For the same input voltage, a simulation was performed in the SPICE simulator Gnucap [10]. The results show good accuracy, although the system is reduced to 3 linear locations. We are able to generate all voltages and currents in the original SPICE system. However there are some jumps in the back transformed voltages due to the nonlinearities approximated by linear locations (see Fig 10 $V_{nout-neg}$).

4.8 Further Examples

The introduced algorithm is additionally used on a third order low pass filter with output limitation at 1.5 V.

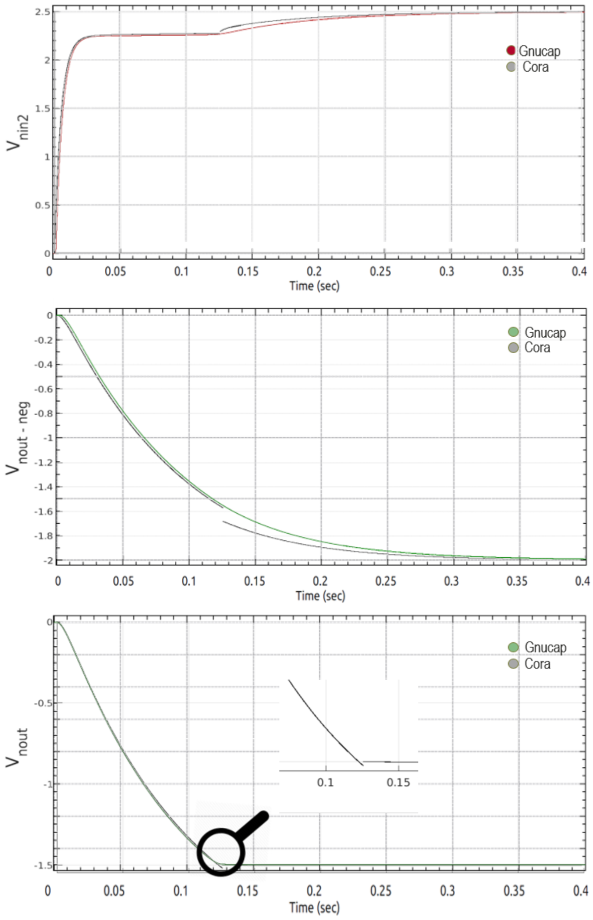

Fig. 10 Rechability analysis of the Hybrid automata created with the stated approach 2 in grey.The Gnucap simulation of the circuit shon in Fig. 2 are ploted in the colors red and green

In the same manner as the previous example, the states of the hybrid system, guards and invariants are identified. Fig. 12 shows the invariants and guards of the reduced system. It should be noted the guards represent hyperplanes. The regions generated for a nonlinear transmission line (RLCD) modeled by resistors, a capacitor, a diode and an inductor is illustrated in Fig. 11. In this figure the guard from group 1 to group 2 is shown. 2 Regions can be identified using the algorithm presented in this paper.
Table 1 summarizes the results of this approach including a last example – a 2^{nd} order bandpass circuit on transistor level with 8 transistors at full BSIM accuracy.

5 Conclusion

In this paper, a new approach for the formal verification of transistor level circuits has been enabled by introducing an automatic abstraction and generation of a hybrid automata of these transistor level circuit. This approach makes use of model order reduction techniques and sampling techniques to identify large linear regions in the nonlinear state space. Some algorithm has been introduced that clusters state space points into connected regions and calculates guards, variable shifts and invariants. The hybrid

Table 1 Results of the abstraction methodology for 4 analog circuits

	2^{nd} o. low pass	3^{rd} o. low pass	RCLD	Bandpass
Sampled points	1619	12256	30716	3801
Reachable points	501	1043	1437	2428
Order	2	3	2	8
Reduced order	2	3	2	2
Dim $\underline{G}$	6x6	7x7	6x6	17x17
Points silhouette < 0.8	10	12	26	590
Percentage of outcasts	0.74 %	1 %	1.8 %	24 %
Parser time	0.72	5.99	21.17	1.664
Matlab read time	4.0	56.5	74.84	15.85
Abstraction time	2.65	10.83	2.66	3.97
A. time with plot	4.753	13.32	5.6	6.51
Max edges convex hull	8	56	10	12
locations	3	3	2	2

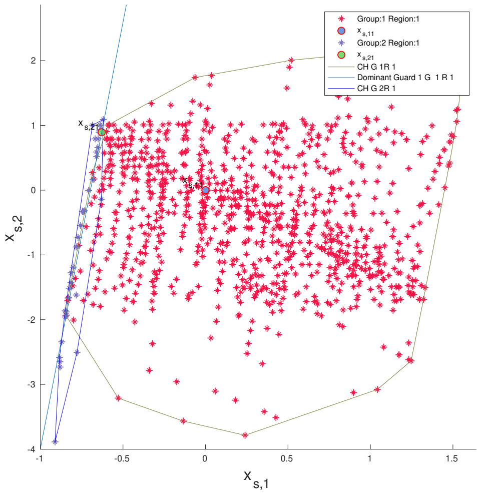

Fig. 11 States of the hybrid automata of the RCLD circuit

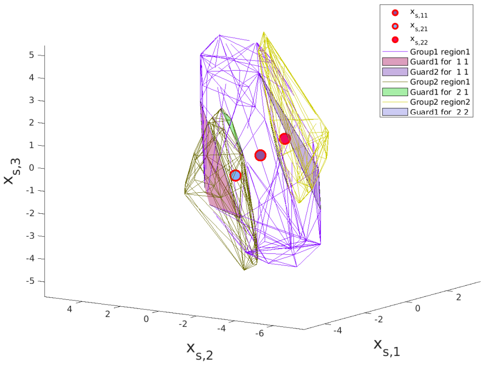

Fig. 12 States of the hybrid automata of a third order low pass filter

automata can finally be used for reachability analysis.

Two main advantages rise from this approach. The first one is that the states of the hybrid automata are now linear. This allows to use the result as a compositional automata, with fewer states than the original system. The second main advantage of this approach is that it reduces the large amount of sampled points into few regions that represent systems. Of course this approach has many challenging topics, such as linearization errors, number of regions to cluster, guards,and many more that will be analyzed in the future. But still, this approach shows promising results as shown throughout this paper.

6 Literatur

[1] T. Dang, A. Donzé, and O. Maler, “Verification of analog and mixed-signal circuits using hybrid system techniques,” in *Formal Methods in Computer-Aided Design*, pp. 21–36, Springer, 2004.

[2] G. Frehse, C. Le Guernic, A. Donzé, S. Cotton, R. Ray, O. Lebeltel, R. Ripado, A. Girard, T. Dang, and O. Maler, “SpaceEx: Scalable verification of hybrid systems,” in *Computer Aided Verification*, pp. 379–395, Springer, 2011.

[3] M. Fränzle, H. Hungar, C. Schmitt, and B. Wirtz, “HLang: Compositional Representation of Hybrid Systems via Predicates,” Reports of SFB/TR 14 AVACS 20, SFB/TR 14 AVACS, July 2007. ISSN: 1860-9821, http://www.avacs.org.

[4] M. Althoff, A. Rajhans, B. H. Krogh, S. Yaldiz, X. Li, and L. Pileggi, “Formal verification of phase-locked loops using reachability analysis and continuization,” *Communications of the ACM*, vol. 56, no. 10, pp. 97–104, 2013.

[5] O. Maler and D. Nikovic, “Monitoring properties of analog and mixed-signal circuits,” *International Journal on Software Tools for Technology Transfer*, vol. 15, no. 3, pp. 247–268, 2013.

[6] W. Zheng, Y. Feng, X. Huang, and H. Mantooth, “Ascend: automatic bottom-up behavioral modeling tool for analog circuits,” in *Circuits and Systems, 2005. ISCAS 2005. IEEE International Symposium on*, pp. 5186–5189 Vol. 5, May 2005.

[7] C. Borchers, “Symbolic Behavioral Model Generation of Nonlinear Analog Circuits,” *IEEE Transactions on Circuits and Systems II: Analog & Digital Signal Processing*, vol. 45, no. 10, pp. 1362–1371, 1998.

[8] H. S. L. Lee, M. Althoff, S. Hoelldampf, M. Olbrich, and E. Barke, “Automated generation of hybrid system models for reachability analysis of nonlinear analog circuits,” in *Design Automation Conference (ASP-DAC), 2015 20th Asia and South Pacific*, pp. 725–730, Jan 2015.

[9] S. Steinhorst and L. Hedrich, “Advanced methods for equivalence checking of analog circuits with strong nonlinearities,” *Formal Methods in System Design*, vol. 36, no. 2, pp. 131–147, 2010.

[10] A. Davis, “An overview of algorithms in Gnucap,” in *University/Government/Industry Microelectronics Symp.*, pp. 360–361, 2003.