Feature based State Space Coverage of Analog Circuits

Andreas Fürtig*, Sebastian Steinhorst§ and Lars Hedrich*

* Institute for Computer Science, Goethe Universität Frankfurt am Main, Germany
§ Department of Engineering, Aarhus University, Denmark
Email: {*fuertig, hedrich*}*@em.cs.uni-frankfurt.de*, sebastian.steinhorst@eng.au.dk

***Abstract*—This paper proposes a systematic and fast analog coverage-driven verification methodology which could increase the confidence in verification of today's analog blocks. We define an appropriate coverage metric to score simulations and then minimize the simulation effort for achieving full state space coverage with an algorithm generating appropriate input stimuli. Our proposed method uses characteristic properties of a discretized representation of the state space such as the spatial distribution of eigenvalues, guiding the generation of short and purposeful stimuli. The experimental results show a significant speed-up with similar accuracy compared to the state-of-the-art.**

I. Introduction and Related Work

Traditionally, analog circuit design and verification needs sophisticated designers and verification engineers to prevent faulty behavior and expensive redesigns. Nowadays, the pressure on them due to the significant analog part on common chips (automotive, consumer) and short design cycles is further increasing. Unfortunately there are not many systematic approaches to tackle the functional verification problem for analog circuits – the standard procedure to prevent hard to find bugs is to use expert knowledge from experienced designers.

One direction to systematically check analog circuits may be the full automatic characterization [1] based on formalized specifications using machine readable specifications [2] or formal languages such as PSL [3]. However, the effort to setup these specifications is sometimes large and, even worse, they do not guarantee to find unknown bugs because they rely on predefined input stimuli for each performance test case. With simulation only, there still exist uncovered scenarios which may later arise as a bug in the field.

Formal verification for analog circuits [4], [5] will certainly help as it can guarantee to find problematic design flaws violating the specification. However it suffers from long runtimes, hard to interpret results and the perennial "translate specification into a formal language" problem.

A compromise could be the use of coverage metrics and coverage-increasing measures. The digital world has developed a lot of coverage metrics [6], [7], [8] and uses them with success. Depending on the complexity of the Device Under Verification (DUV), the methods are more or less complete. The complete methods investigate for example Finite State Machines (FSM) [9] and have some means to try to restrict the simulation input stimuli to the relevant part of the state space (see SFSM in [9]). The less complete methods (code coverage, specification coverage) use measures to guide the verification to the most probable bug location for example by systematically visiting each conditional branch in an HDL-description.

For analog circuits, a very low number of coverage investigating approaches besides the above explained formal verification techniques exist. There are some approaches stemming from the test community measuring and increasing the analog fault coverage [10], [11]. However, they are not intended to find functional faults. Horowitz et al. [12] also tries to increase the confidence in the functional verification using a high-level functional model but without a systematic method to increase some underlying measure. Two other approaches are built for hybrid systems [13], [14], suffering from being able to handle strongly nonlinear analog circuits on transistor level. Steinhorst et al. [15] and Karthik et al. [16] concentrate on the analog state space to systematically implement formal verification, hence being accurate and complete. However, they also have no well defined measure for the coverage and suffer from the large state space to investigate. As a remedy, in this paper, we later propose a coverage optimization algorithm that takes into account the dynamics of the state space. Consequently, the algorithm can identify regions of critical nonlinear behavior requiring a very dense coverage, as well as regions with highly linear behavior which is not critical for the verification coverage. The latter will enable us to drastically reduce the volume of the visited state space.

Contributions. This paper introduces a complete methodology for optimization of analog verification coverage by analyzing the dynamics of the state space of the Design under Verification (DUV), providing four main contributions outlined in the following.

- We present a state space coverage metric in Section III which creates a relation between transient simulation waveforms and states of a discrete representation of the DUV which we introduce in Section II.
- Based on the coverage metric, we introduce a coverage optimization algorithm that maximizes the defined coverage metric.
- The state space coverage metric and algorithm are further developed into the proposed λ state space coverage metric, to identify interesting regions and neglect uniform parts of the state space in Section IV.
- Our metrics and algorithms are evaluated on several analog transistor level circuits in Section V and clearly show the advantages over a state-of-the-art approach.

II. State Space Model Generation

The state space coverage analysis we are proposing in this paper requires a discrete model of the analog circuit. We use a trajectory based state space discretization proposed in [17]. This method is based on discretizing the underlying DAE-System of the circuit in the state space. The discretization is performed using an electrical circuit simulator with full SPICE accuracy [18].

With these method we can construct a discrete state space model M_{ATS}:

$$\text{Electrical Circuit} \overset{\text{discrete modeling}}{\longrightarrow} M_{ATS} \tag{1}$$

Analog Transition System (ATS)
For the ATS we define a five-tuple $M_{ATS} = (\Sigma, R, L_V, T, L_\lambda)$ where

- Σ is a finite set of states of the system.
- $R \subseteq \Sigma \times \Sigma$ is a total transition relation, hence for every state $\sigma \in \Sigma$ there exists a state σ' such that $(\sigma, \sigma') \in R$.
- $L_V : \Sigma \to \mathbb{R}^{n_d}$ is a labeling function that labels each state with the vector of n_d variables containing the values of the state space variables and the inputs of the DAE system.
- $T : R \to \mathbb{R}_0^+$ is a labeling function that labels each transition from σ to σ' with a real valued positive or zero transition time that represents the time required for the trajectory in the state space between these states.
- $L_\lambda : \Sigma \to \mathbb{R}^{n_\lambda}$ is a labeling function that labels each state with a vector of the n_λ eigenvalues associated with the state.

Within the structure M_{ATS}, a path π beginning at state σ is a sequence of states $\pi = \sigma_0, \sigma_1, \sigma_2, ..., \sigma_n$ with $\sigma_0 = \sigma$ and $(\sigma_i, \sigma_{i+1}) \in R$ for $0 \leq i < n$.

In an extension to the method of [17] we calculate and store the eigenvalues of each state during the discretization process. For this purpose, the system's dynamics are linearized in the specific state and then transformed into the frequency domain using Laplace transformation. The number of non-zero entries in Kronecker's canonical form of the transformed capacitance matrix of the frequency domain representation corresponds to the number n_λ of eigenvalues in the generalized eigenvalue problem. For a detailed description of the eigenvalue decomposition, please refer to [19].

III. State Space Coverage

This section describes a method to match a transient simulation response to the previously described state space. Our goal is to automatically create an input stimulus for an analog simulator to maximize the presented analog state space coverage. A path finding algorithm is presented afterwards, as well as possible restrictions of this method.

The state space coverage ζ denotes the ratio between visited states and the sum of all reachable states Σ_R of a given circuit. The wanted coverage metric should assess a simulation response based on the following characteristics:

- A coverage value near 100% implies a high probability that all possible faults of the circuits could be detected.
- The measure has to be monotonic in the number of visited states: If more states are visited, the measure should increase.

The resulting number can be used to compare different input stimuli for an analog simulator leading the designer to much more useful test cases, reducing the possibility of missing possible design flaws.

The Analog Transition System M_{ATS} described in Section II creates a vast number of states which could possibly not be reachable at all. Using the number of states $|\Sigma|$ of the full system results in a metric not able to gain full coverage. Hence, a set of reachable states Σ_R is computed from all states Σ visited by the state space discretization using a simple set based reachability algorithm. For our purpose, the number of reachable states is lower or equal the number of all states.

A. *State Space Coverage Calculation*

A transient simulation response consist of a set of different data points, representing the state of an analog circuit at a given time step. To match each of these points to a set inside our M_{ATS}, we use an Euclidean distance to mark a state as covered by a simulation.

In a very first straight-forward approach, one can compute a nearest neighbor for every data point. For that, we store the previously defined Analog Transition System M_{ATS} in a suitable space-partitioning data structure in form of a k-d tree [20]. The number of nodes in this tree equals the number of states in the system. Hence, if a discretization only consists of very few states, each point of a simulation response will lead to a covered state, although the state is very far off. Obviously this simple approach does not calculate a smooth and adequate measure. Since every point has a nearest neighbor, the distance is not considered (cf. Fig. 1, upper left).

A much better approach is to select every state in a given distance around a data point of the transient simulation response. This allows to have a measure independent of the sampling distance in the state space as well as the sampling distance of the transient simulation result. Fig. 1 shows different values for a distance to accept different states inside a M_{ATS} marked as covered. It can be seen, that a maximum distance must be chosen adequately, since using a too large distance could mark states with different behavior compared to the transient trajectory under investigation, while a too small distance will underestimate the set of covered states C.

A good starting point for the distance is to select the *median* distance between two neighbor states in the discrete state space or to use a percentage of the diameter of the reachable state space. Here, we conservatively take the median length of all transitions R inside the M_{ATS}.

Consequently, the coverage of a given transient simulation response can now be computed using the cardinality of the

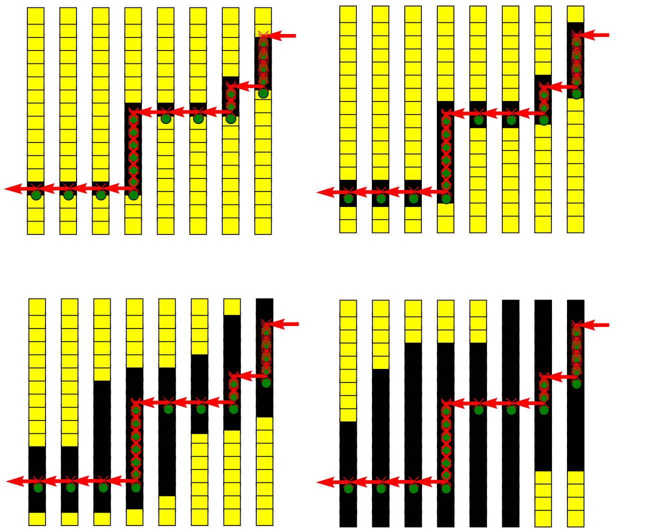
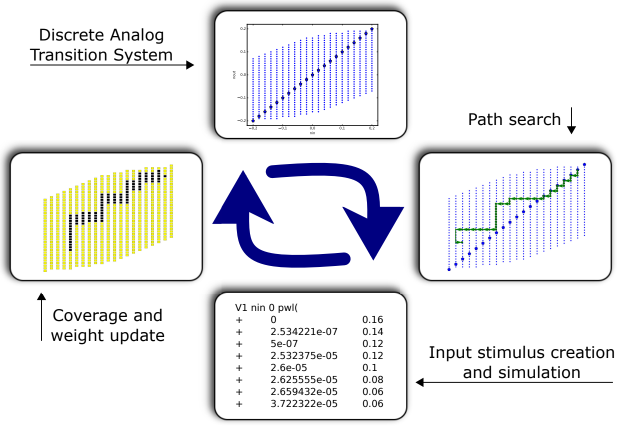


Fig. 1. Euclidian distance method for selecting the *covered* states of a simulation result. Red crosses indicate the trajectory corresponding to the transient simulation result. Black boxes are marked as covered, yellow boxes are marked as uncovered.


Discrete Analog Transition System
Path search
Coverage and weight update
Input stimulus creation and simulation

```
V1 nin 0 pwl(
+        0               0.16
+        2.534221e-07    0.14
+        5e-07           0.12
+        2.532375e-05    0.12
+        2.6e-05         0.1
+        2.625555e-05    0.08
+        2.659432e-05    0.06
+        3.722322e-05    0.06
```

Fig. 2. Coverage maximization algorithm based on discrete state space modeling.

elements in the set C and the number of states in the reachable discrete state space Σ_R:

$$\zeta = \frac{|C|}{|\Sigma_R|} \tag{2}$$

This gives us the possibility to rate a set of test by calculating a coverage for each test as long as a full coverage is reached. Full coverage in this context means every reachable state inside a M_{ATS} was reached by simulation. Hence, no unexpected behavior can occur. To enhance the coverage ζ a designer could develop new tests as long as uncovered states exists. As we will see in Section IV an alternative is to restrict the number of "to be reached" states from $|\Sigma_R|$ to $|\Sigma_\lambda|$.

B. Path Planning

As mentioned in the previous subsection, a full coverage of a discrete state space is a desirable goal. For automation purposes, a path planning method is introduced, as the creation of appropriate input stimuli is crucial for the usage of the coverage described before.

First of all, the M_{ATS} is enhanced by a labeling function $\omega_\sigma : \Sigma \to \mathbb{N}_0^+$ that labels each state with a weight, denoting the number of visits of this state by a simulation. Together with the relation R, this eases a path finding inside the discrete state space. Another helping aspect is an additional set Σ_{DC}, which holds a set of DC operation points of the M_{ATS}. As stated beforehand a path π through the discrete state space can be directly used to create an input stimulus for a simulation software, as the labeling function L_V also holds the inputs to the system. Timing informations can be gathered from the transitions between two states in the M_{ATS}.

Using an A^* algorithm, it is easy to compute a path through the state space targeting an uncovered state. This method will lead to a vast number of very small simulations. As a result, the startup time of the simulation software will dominate the simulation time. To avoid this behavior, a path planning algorithm is needed to create simulation input stimuli which meet the following characteristics:

- The resulting path should avoid already visited states.
- It should consist of as many unvisited states in the M_{ATS} as possible.

An approach to satisfying these criteria exists in [15], but with larger circuit size a full input stimulus created using this method consists of significantly more data points than $|\Sigma_R$ itself. More complex circuits lead to a very long runtime of the simulation, due to the increased state space dimensions and more state space points. As we will see in the results section, the constructed single stimulus by that method performs badly in terms of the achieved state space coverage.

To improve this method, we introduce a weight-based path planning: Let π be a set of states describing the path from a starting to a target state. The length $|\pi|$ is the number of states inside the path. Since every state has a weight ω_σ, the weight of a path is $\omega_\pi = \Sigma_i^n \omega_{\sigma_i}$. An unvisited, randomly chosen state σ_u (indicated by its weight $\omega_\sigma = 0$) is used to compute *all* possible paths from every operation point $\sigma_d \in \Sigma_{DC}$. Additionally, we compute the *longest* path starting in state σ_u back to the operating points. This gives us the possibility to concatenate the resulting paths, producing a longer overall path. To avoid very long paths (like the ones created by the method in [15]), the length of the resulting path is limited by the number of unvisited states in the whole system.

C. Coverage Maximization Algorithm

With bigger analog circuits, the possibility to reach full coverage with one single input stimulus is very small. For that, we introduce an algorithm to cover all reachable states inside a M_{ATS}. It is very easy to see, that one single stimulus created by the path finding algorithm described beforehand, will not reach all states in the system.

The presented Coverage Maximization Algorithm is shown in Fig. 2. In every step, the algorithm selects an unvisited state and calculates a path targeting that state. Selecting the longest path with minimum cost maximizes the possibility to cover at most unvisited points at once. While traversing the graph, we are able to create an input stimulus for a path. Each data point of the resulting transient response is mapped to a state inside the M_{ATS}, increasing the weight of that state as well as the overall coverage of the whole system. The algorithm stops, if every state in the state space is covered by a simulation.

With increasing complexity of the investigated circuits, it is furthermore not possible to reach high coverage measures ζ in a reasonable computation time and with short overall input stimuli length. Hence, the number of states to inspect must be reduced, which will be described in the next section.

IV. λ State Space Coverage

As we will see later on in the results section, trying to reach a full coverage is a very time consuming procedure even for small devices like a Schmitt trigger or lowpass filter circuit. Visiting every single reachable state in an analog circuit often makes no sense, since many regions of the state space have a homogeneous behavior – in most cases linear behavior –and can be investigated by one trajectory through these regions. To reduce the number of states to cover without missing regions with a heterogeneous behavior and important states, we are segmenting the discrete state space into different regions. Namely, these are regions with uniform (linear) behavior, non-linear parts with high dynamic (such as limited output voltage swings) or border regions of the discretization. Regions with nonlinear or static nonlinearities needs much deeper investigation, too. In this section we will describe different classes of analog circuits and suggest some methods to detect them.

To differentiate the states in the M_{ATS} distance based methods are used as well as eigenvalues, which are computed and stored during the discretization process in every state of the system. Static circuits like mixer or Low-dropout regulators can be compared using their linear or translinear behavior. In sum, this leads to five different coverage value vectors which will be described in the following. Each vector has the same length as the amount of states in the discrete state space and is either set to 0 or 1, depending on the response of the according method of inspect.

1) Local linear regions: Many analog circuits have a linear behavior and huge regions inside the discrete state space with similar behavior. To detect those regions, we are using the previously stored eigenvalues in every state of the system. Each state σ in the system has a list of ancestors and successor states. $\vec{L}_\sigma$ is set to 1 if one of the neighboring states has a significantly ($|\cdot| > 50\%$) different eigenvalue than σ, otherwise it is set to 0. Fig. 3 shows a simple lowpass filter circuit and two large linear regions. As the circuit is ideal, no nonlinearities occur and we have a large linear region (blue).

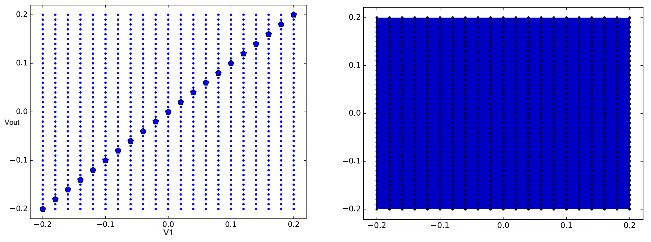


Fig. 3. Detection of linear regions: The discrete state space of a simple RC-lowpass filter circuit (left) and the values of $\vec{L}_\sigma$ (right). As the whole system is linear we got only one big linear region (blue).

On the other hand, in Fig. 4 the same analysis is conducted for an inverting active RC lowpass with an operational amplifier. This circuit has a large linear region and small nonlinear regions. The latter is due to shifted eigenvalues when the operational amplifier output reaches saturation at the supply rails and in this case also $0.7V$ before reaching the $2.5V$ positive supply rail.

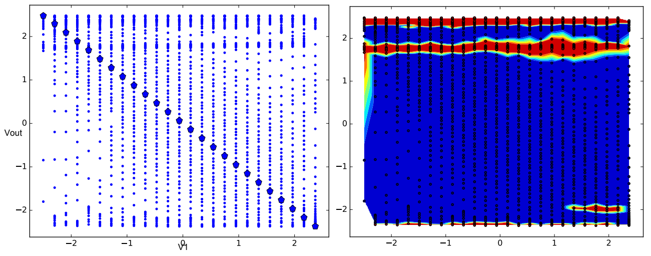


Fig. 4. Detection of linear regions: The discrete state space of an inverting active RC-lowpass filter circuit with limiting due to the used operational amplifier (left) and the values of $\vec{L}_\sigma$ (right).

2) Global linear regions: Similar to the previous described detection of local linear regions, global linear regions can be detected using the eigenvalues of the whole system. First of all, we compute the median of all eigenvalues of the whole M_{ATS}, as this indicates the basic dynamic level of the analog circuit. $\vec{D}_\sigma$ is then the absolute difference to the median value for each state $\sigma \in \Sigma_R$. All values are normalized to $[0, \ldots, 1]$ to ease the later summation process. Fig. 5 shows the results of this detector for a basic Schmitt trigger circuit.

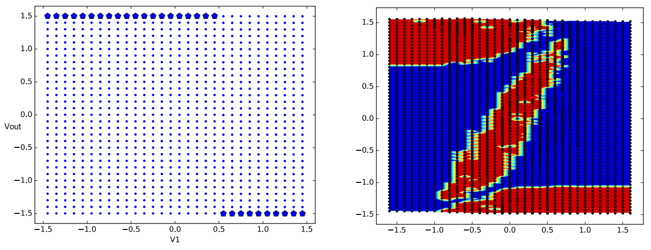


Fig. 5. Detection of global linear dynamic regions: The discrete state space of a Schmitt trigger circuit (left) and the resulting areas with nonlinear dynamics (right, red points).

3) Border regions: As mentioned before, border regions are interesting and should be visited in any case by the path finding algorithm. To compute the states in the border region of the reachable set, the convex hull $conv(\Sigma_R)$ of all reachable states of the circuit is computed using the approach from [21]. $\vec{B}_\sigma$ is set to 1 if the state σ is located within a Euclidean distance on the edges of the resulting polytope, otherwise it is set to 0.

4) DC operating points: In the same manner as the border regions, the direct neighborhood of each DC operating point is computed. $\vec{O}_\sigma$ is set to 1 if the state σ lies in the neighborhood of the DC operating point or is the state itself. Fig 6 shows the result of the DC operating point detector as well as the border regions.

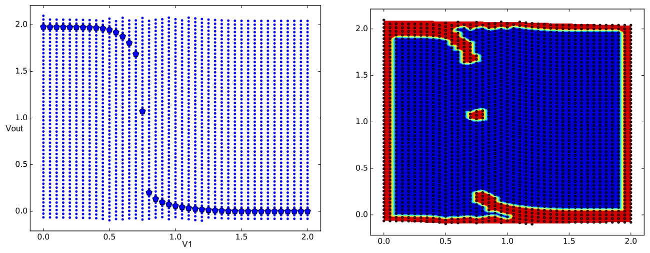


Fig. 6. Detection of regions at the border and around DC operating points: Discretization of an inverter example (left) and the resulting areas (left).

5) Static circuits: Besides the so far described circuits, linear constant (Low-dropout regulators) or so called translinear circuits (mixer circuit) exists. For these class of analog circuits – which are easily discernible as they only consists of DC operating points – an optimal output function $\vec{f}_{out}$ exists. This function is currently guessed but could be automatically gathered by some sort of optimization process. $\vec{S}_\sigma = |\vec{f}_{out} - \vec{f}_{meas}|$ is the absolute error between the output function and measured output voltage of the analog circuit normalized to $[0, \ldots, 1]$. Fig. 7 shows the result of the static circuit area detector.

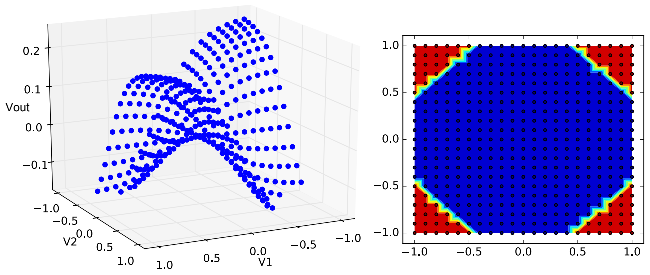


Fig. 7. Detection of static circuit regions: Discretization of a mixer circuit (left) and the resulting normalized error (right, red points indicate a high error).

As every step of the analysis described beforehand indicates possible interesting states of the full M_{ATS} system, the region of interest of the device under test is formed by the non-zero entries in $\vec{I}$ defined as:

$$\vec{I} = \vec{L} + \vec{D} + \vec{B} + \vec{O} + \vec{S} \tag{3}$$

The importance of each state is now indicated by the according value in the vector $\vec{I}$. On the other hand, if its value is 0, none of the previously described detectors marked this state as important. This information can now be used and integrated in the path planning algorithm presented in Section III. In this algorithm, each state was initialized with a node weight $\omega_\sigma = 0$ indicating that this state was never visited before by a transient simulation. According to this, we initialize the weight of a state by its interest factor I_σ:

$$w_\sigma = \begin{cases} 0, & \text{if } I_\sigma \geq t, \\ 1, & \text{otherwise} \end{cases} \tag{4}$$

, where t is a given threshold. All states with a low weight ω_σ (and therefore a high interest value $I_\sigma \geq t$) are now preferred by the path finding algorithm. States with a higher weight are not removed from the path planning algorithm, so that there is still a small possibility that a simulation covers this state.

With these information we are now able to create a reduced set of states $\Sigma_\lambda \subseteq \Sigma_R$ which consists of all interesting state space points with an interest factor $I_\sigma \geq t$:

$$\Sigma_\lambda = \{\sigma \in \Sigma_R | I_\sigma \geq t\} \tag{5}$$

The λ *state space coverage* can now be defined as the number of visited states divided by the number of states in the reduced set Σ_λ, where in the numerator only states are counted which belong to that reduced set Σ_λ:

$$\zeta_\lambda = \frac{|C \cap \Sigma_\lambda|}{|\Sigma_\lambda|} \tag{6}$$

The definition of a λ state space coverage and experimental results show clear evidence that the concept is still very pessimistic uncovering all design flaws with large possibility.

V. Results

In this chapter we will demonstrate our proposed method on various analog circuits on transistor level as well as on some selected Verilog-A implementations (see Table I). The examples try to cover many possible types of analog circuits: static nonlinear systems, dynamic linear systems and dynamic nonlinear system to show the wide scope of our method.

In Table II three methods are presented. The *normal* method is taking every state of the discretization into account to calculate a coverage, while the *proposed* method only uses interesting states based on the criterion of Section IV. To show the overall speedup, both methods are compared against the *single* method [15]. The experiments are carried out on a 3.4 Ghz Dual-Core machine.

Starting from some very basic analog circuits, a lowpass filter has a high amount of states depending on the discretization accuracy. For this often used circuit, a lot of simulations seems to be needed to gain full coverage for a straight forward "normal" method. In comparison to that, the analysis of the state space reduces the number of interesting states by 74%. Full coverage can be reached by one simple simulation. This should be desired for circuits of that size. For another very basic example, the inverter circuit, the amount of states can be reduced by 77.6% as well.

A bandpass filter [22] example with one input and two dimensions (Fig. 8) has more than 5000 states after the discretization process. Due to the heavy nonlinearities at the limiting region of the operational amplifier full coverage in this example is not possible, as not every as reachable marked state can really be reached by a simulation path. This is because

TABLE I. Statistics of applied analog circuits on transistor level. The *Mixer* and *Low-dropout regulator* examples are implemented as a Verilog-A behavior model and therefore have no transistors as the other examples.

Analog Circuit	Number of inputs	Number of transistors	State space dimensions	States in reachable set $\|\Sigma_R\|$	States in λ reduced set $\|\Sigma_\lambda\|$	Schematic
RC lowpass filter	1	0	2	861	227	Basic
inverter circuit	1	2	2	1510	338	Basic
Schmitt trigger	1	10	2	1903	356	[22]
Bandpass filter	1	8	3	5660	1948	[22]
Level shifter	1	6	3	1081	641	[23]
log domain filter	1	13	2	2526	394	[24]
gmC filter	1	69	3	344	239	[24]
Mixer	2	—	2	442	120	industrial
Low-dropout regulator	2	—	2	469	214	industrial

there are always some discretization errors during the creation of the state space. Hence, a trajectory can be computed with the presented path finding algorithm from section III, but the created input stimulus for the simulation does not reach all wanted target states. After 173 simulations a coverage of 83.06% is obtained. With the presented state space analysis, only 1948 states are marked as interesting, so the overall sum of simulations can be reduced to only 18 simulations, reducing the overall runtime of the simulation by 70.4%. As the discretization error still exists, full coverage cannot be reached for that example, too.

Another unsophisticated, but also an example with strong nonlinearities is a Schmitt trigger circuit where the simulation runtime could be reduced by 95%. As this circuit has big areas with linear behavior and only a small region with nonlinear dynamics, the number of important states can be decreased dramatically.

To demonstrate our approach on static nonlinear system, we calculate a coverage for two static examples: a mixer and a Low-dropout regulator circuit. Both implementations are used in an industrial environment. The resulting state space could be decreased to speed up the simulation effort. Our presented method detects the interesting regions (e.g. high load for the LDO) automatically and calculates a high coverage by only running few simulations to that region.

To complete our result section, we are also able to create a discretization of a Verilog-A model description of an analog circuit (see mixer and low-drop regulator). This speeds up the simulation process on the one hand and also eases the implementation effort of sophisticated circuits. Using this approach, our presented method can also be used to create input stimuli to check the equivalence of two different implementations of the same system with large confidence.

VI. Conclusion

In this paper, a new coverage metric for analog circuits has been proposed. The λ state space coverage uses the eigenvalues and structural properties of the reachable state space of a nonlinear analog circuit on transistor level to extract a set of states in the state space which have to be visited by input stimuli. It keeps strongly nonlinear regions in that set while neglecting linear, uniform regions, resulting in 6.8 times speed up of the simulation time of the generated stimuli. The quality of the input stimuli is still as high as with the presented standard state space based coverage method and better than state-of-the-art methods. Experimental results show that hidden faults can be uncovered and real industrial circuits with up to 69 transistors can be handled efficiently. We can conclude that the confidence in the input stimuli for a certain analog circuit can be measured by the proposed metric and that the verification coverage can be significantly increased with a small simulation overhead using the proposed λ state space coverage maximization algorithm.

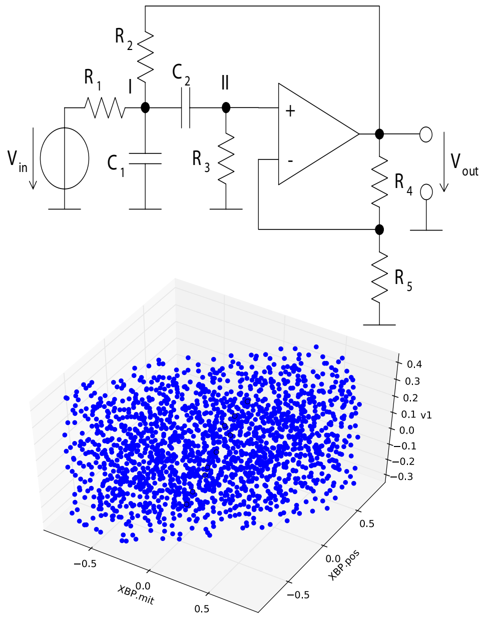


Fig. 8. Schematic of a bandpass filter [22] (top) and the result of the discretization process with 5660 states (bottom).

Future work will concentrate on improving the scalability to work on bigger examples. Additionally the path finding algorithm can be further refined to prevent double visiting of states, resulting in shorter input stimuli and faster simulation runs.

VII. Acknowledgments

This work has been carried out in the project ANCONA, funded by the German Federal Ministry of Education and Research (BMBF) in the ICT2020 program under grant no. 16ES021. The project is supported by the industry partners Infineon Technologies AG, Robert Bosch GmbH, Intel AG and Mentor Graphics GmbH.

References

[1] J. Eckmüller, M. Gröpl, and H. Gräb. Hierarchical Characterization of Analog Integrated Circuits. *DATE '98: Design, Automation and Test in Europe*, 1998.

TABLE II. RESULTS OF THE PROPOSED COVERAGE CALCULATION ALGORITHM. *normal* IS THE PRESENTED PATH PLANNING ALGORITHM WITH UNDERLYING STANDARD STATE SPACE COVERAGE METRIC. *proposed* IS THE PROPOSED PATH PLANNING ALGORITHM WITH UNDERLYING λ STATE SPACE COVERAGE METRIC. *single* IS THE PATH PLANNING ALGORITHM FROM [15] (NOT AVAILABLE FOR *Low-dropout regulator* AND *mixer* EXAMPLES).

		Coverage $> 50\%$		Coverage $> 75\%$		Overall Coverage in %		
Analog Circuit	Method	Simulations	Runtime	Simulations	Runtime	Coverage	Simulations	Runtime
RC lowpass filter	*normal*	1	4.88	2	5.62	ζ 100.00	5	7.28
	proposed	1	1.60	1	1.60	ζ_λ 100.00	1	1.60
	single	1	7.11	1	7.11	ζ 100.00	1	7.11
inverter	*normal*	1	0.15	2	0.23	ζ 92.72	5	0.49
	proposed	1	0.15	1	0.15	ζ_λ 94.32	2	0.24
	single	1	3.79	1	3.79	ζ 89.93	1	3.79
Schmitt trigger	*normal*	3	98.96	5	134.06	ζ 87.06	17	215.83
	proposed	1	5.62	2	8.16	ζ_λ 91.40	3	10.79
	single	1	19.29	1	19.29	ζ 30.32	1	19.29
Bandpass filter	*normal*	6	74.87	15	124.95	ζ 83.06	173	556.86
	proposed	1	31.08	6	97.37	ζ_λ 88.99	18	164.75
	single	1	388.96	—	—	ζ 62.59	1	388.96
Level shifter	*normal*	—	—	—	—	ζ 43.91	9	81.46
	proposed	1	12.81	—	—	ζ_λ 72.21	6	27.97
	single	1	266.24	—	—	ζ 66.39	1	266.24
log domain filter	*normal*	1	1.91	2	2.15	ζ 100.00	7	3.23
	proposed	1	0.40	1	0.40	ζ_λ 100.00	4	1.66
	single	1	16.65	1	16.65	ζ 100.00	1	16.65
gmC filter	*normal*	2	0.58	7	2.98	ζ 76.57	8	3.98
	proposed	1	0.47	3	1.18	ζ_λ 85.17	4	1.92
	single	1	8.59	—	—	ζ 65.71	1	8.59
Mixer	*normal*	2	0.49	4	2.23	ζ 100.00	8	5.23
	proposed	1	0.43	3	1.48	ζ_λ 100.00	6	3.92
	single	—	—	—	—	—	—	—
Low-dropout regulator	*normal*	1	1.37	3	2.40	ζ_λ 100.00	5	2.79
(Verilog-A)	*proposed*	1	1.13	2	2.19	ζ 100.00	3	2.31
	single	—	—	—	—	—	—	—

[2] Mingyu Ma, Lars Hedrich, and Christian Sporrer. ASDeX: a formal specification for analog circuit enabling a full automated design validation. *Design Automation for Embedded Systems*, 2012.

[3] Zhi Jie Dong Ghiath Al Sammane, Mohamed H. Zaki and Sofiene Tahar. Towards Assertion Based Verification of Analog and Mixed Signal Designs Using PSL. *Forum on Design Languages (FDL)*, 2007.

[4] Mohamed H Zaki, Sofiène Tahar, and Guy Bois. Formal verification of analog and mixed signal designs: A survey. *Microelectronics Journal*, 39(12):1395–1404, 2008.

[5] S. Steinhorst and L. Hedrich. Model Checking of Analog Systems using an Analog Specification Language. In *Proc. Design, Automation and Test in Europe DATE '08*, pages 324–329, 10–14 March 2008.

[6] Finn Haedicke, Daniel Große, and Rolf Drechsler. A guiding coverage metric for formal verification. In *Design, Automation & Test in Europe Conference & Exhibition (DATE), 2012*, pages 617–622. IEEE, 2012.

[7] Shai Fine and A. Ziv. Coverage directed test generation for functional verification using Bayesian networks. In *Design Automation Conference, 2003. Proceedings*, pages 286–291, June 2003.

[8] Andrew Piziali. *Functional Verification Coverage Measurement and Analysis*. Springer Publishing Company, Inc., 1st edition, 2007.

[9] Jing-Yang Jou and C Liu. Coverage analysis techniques for hdl design validation. *Proc. Asia Pacific CHip Design Languages*, 1999.

[10] K. Arabi and B. Kaminska. Parametric and catastrophic fault coverage of analog circuits in oscillation-test methodology. In *VLSI Test Symposium, 1997., 15th IEEE*, pages 166–171, Apr 1997.

[11] Joonsung Parky, S. Madhavapeddiz, A. Paglieri, C. Barrz, and J.A. Abraham. Defect-based analog fault coverage analysis using mixed-mode fault simulation. In *Mixed-Signals, Sensors, and Systems Test Workshop, 2009. IMS3TW '09. IEEE 15th International*, pages 1–6, June 2009.

[12] Mark Horowitz, Metha Jeeradit, Frances Lau, Sabrina Liao, ByongChan Lim, and James Mao. Fortifying Analog Models with Equivalence Checking and Coverage Analysis. In *Proceedings of the 47th Design Automation Conference*, DAC '10, pages 425–430, New York, NY, USA, 2010.

[13] A. Julius, G. Fainekos, M. Anand, I. Lee, and G. Pappas. Robust Test Generation and Coverage for Hybrid Systems. *Proceedings of the 10th International Conference on Hybrid Systems: Computation and Control (HSCC)*, pages 329–342, 2007.

[14] Tarik Nahhal and Thao Dang. Test coverage for continuous and hybrid systems. In *Computer Aided Verification*. Springer, 2007.

[15] S. Steinhorst and L. Hedrich. Improving Verification Coverage of Analog Circuit Blocks by State Space-Guided Transient Simulation. In *IEEE International Symposium on Circuits and Systems*, May 2010.

[16] Aadithya V Karthik, Sayak Ray, Pierluigi Nuzzo, Alan Mishchenko, Robert K Brayton, and Jaijeet Roychowdhury. ABCD-NL: Approximating continuous non-linear dynamical systems using purely Boolean models for analog/mixed-signal verification. In *ASP-DAC*, 2014.

[17] S. Steinhorst and L. Hedrich. Trajectory-directed discrete state space modeling for formal verification of nonlinear analog circuits. In *Proceedings of the International Conference on Computer-Aided Design*, pages 202–209. ACM, 2012.

[18] A. T. Davis. An overview of algorithms in Gnucap. In *University/Government/Industry Microelectronics Symp.*, 2003.

[19] Sebastian Steinhorst and Lars Hedrich. Advanced methods for equivalence checking of analog circuits with strong nonlinearities. *Formal Methods in System Design*, 36(2):131–147, 2010.

[20] Jon Louis Bentley. Multidimensional Binary Search Trees Used for Associative Searching. *Commun. ACM*, 18(9), September 1975.

[21] Bernard Chazelle. An Optimal Convex Hull Algorithm in Any Fixed Dimension. *Discrete & Computational Geometry*, 10:377–409, 1993.

[22] 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.

[23] Wen-Tai Wang, Ming-Dou Ker, Mi-Chang Chiang, and Chung-Hui Chen. Level shifters for high-speed 1 v to 3.3 v interfaces in a 0.13 μm cu-interconnection/low-k cmos technology. In *VLSI Technology, Systems, and Applications, 2001. Proceedings of Technical Papers. 2001 International Symposium on*, pages 307–310. IEEE, 2001.

[24] L. Hedrich and W. Hartong. *Low-Power Design Techniques and CAD Tools for Analog and RF Integrated Circuits*, chapter Approaches to Formal Verification of Analog Circuits, pages 155–192. Kluwer Academic Publishers, 2001.