Identifying Undetectable Defects Using Equivalence Checking

Lars Hedrich *, Inga Abel †, Jaafar Mejri †, Vladimir Zivkovic ‡
* Goethe University Frankfurt, Germany
† Infineon Technologies AG, Munich, Germany
‡ Infineon Technologies, Denmark, now with Siemens EDA A/S, Denmark

***Abstract*—The paper proposes a method for identifying undetectable defects in mixed-signal circuits to exclude them from further consideration in defect campaigns. The method uses an equivalence checker to generate counterexamples or prove the undetectability of a defect.**

I. Introduction

A defect simulation – or broader a defect campaign – for analog circuits is a major challenge during the sign-off procedure of analog circuits with 0 DPPM requirements. Recently, this task is accompanied by activities to standardize defect modeling and coverage (IEEE Std. P2427,[1]).

Since some years defect simulators are available from a number of commercial EDA vendors. In principle, a defect simulation for every defect can be conducted. However, the number of defects is huge and an analog simulation for many circuit classes needs long simulation times resulting in infeasible runtimes of defect campaigns.

In this paper, we propose a new approach to reduce the number of defects for a defect campaign utilizing an equivalence checker (EC) to identify undetectable defects. The principle is shown in Fig. 1 and will be explained in Sec. III.

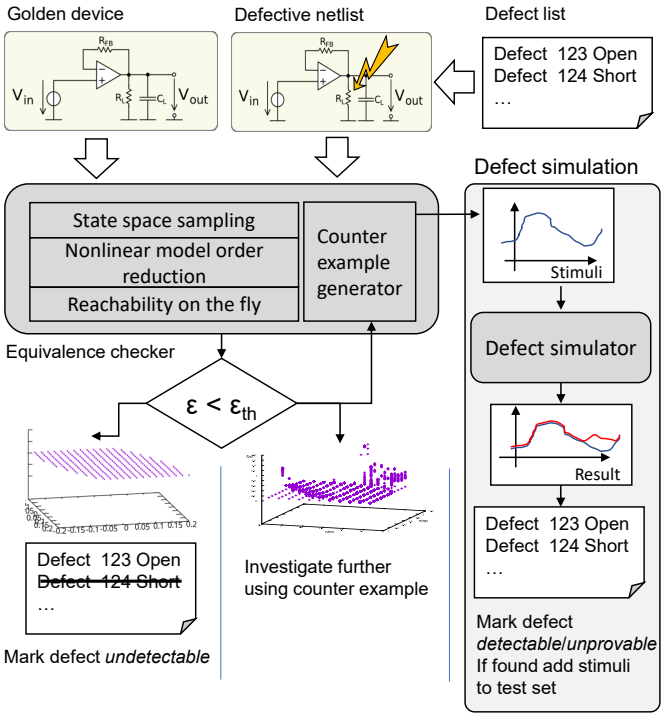

Fig. 1: The proposed approach overview. A golden netlist and a defective netlist are compared using an equivalence checker. Depending on the result the defect may be marked as undetectable or further investigations are needed using a defect simulation flow with the generated counterexamples stemming from the equivalence checker

II. Previous Work

Fault/defect simulation has been used for a long time and moved around 2010 from academia to commercial tools. An overview of defect injection and simulation is given in [2]. As the defect simulation is also more relevant for the industry, more efforts e.g. in the form of an IEEE working group P2427 are conducted to standardize the methods, defect models and measures for defect campaigns - [3] gives an introduction and an overview. Two challenges for real-world applications can be identified: the number of defects is really large, and each defect has to be simulated in the worst case. The pure number renders a full defect simulation infeasible. However, defect collapsing may reduce the number significantly [3]. Weighting may additionally focus on important defects [4].

The second challenge is using the right test stimuli. For analog circuits, no well-defined Boolean model as in digital circuits exists, which enables an automated test pattern generation (ATPG) (e.g. D-Algorithm, SAT).

Such ATPG methodologies have also been applied to analog circuits, e.g. for linear circuits in [5] using direct computation of stimuli in the frequency domain. The caveat is, that circuits on a tester are not operated in the frequency domain but in the transient domain. The methods should also be able to handle nonlinearities like the limiting behavior of operational amplifiers. [6] gives a method to optimize transient stimuli by an evolutionary algorithm for a leapfrog filter. However, the underlying algorithm needs a lot of simulations to generate one stimulus and is due to the stochastic property not able to prove that a defect can not be detected.

In parallel formal methods have been developed for analog circuits that could help identify proofs for undetectable defects. An overview of formal methods in that area is presented in [7]. [8] describes a method enabling model checking techniques based on transient analysis. In [9] and [10] methods for equivalence checking are presented. They can prove with an appropriate error measure the equivalence of two circuits.

Primarily intended for comparison of behavioral models versus a transistor-level implementation they can also be used to compare two transistor-level circuits. This opens the possibility of comparing a golden circuit with a defective circuit. If the equivalence can be proved, the defect is proven to be undetectable.

III. Concept of Defect Identification

The proposed method uses equivalence checking (EC) for the identification of undetectable defects supported by a counterexample generation followed by defect simulation for the detectable defects. Fig. 1 shows the principle: The original circuit and the circuit with the defect are supplied to an equivalence checker. If the circuits are equivalent (given by an appropriate threshold) the defect can be identified as undetectable. If the EC does not finish or aborts, the defect can not be identified and is handled as undecided. If the EC reports differences exceeding the given threshold, a counterexample is generated and fed into a defect simulator to check if the counterexample leads to a detection of the defect in the defect simulation setup. This two-step process ensures that only defects that are detectable by simulation are marked as detectable. Due to the properties of the EC algorithm, the counterexample may be a bit too optimistic. This may lead to an erroneous marking of a defect as detectable, where no real detection exists. In such cases, we mark them as unprovable.

In summary, the contributions of this paper are:

- use of an analog equivalence checker to identify undetectable defects based on state space models and reachability analysis,
- introduction of a new error measure for the equivalence checker to reflect the electrical and physical limits restricting the capabilities of a tester to detect the defect,

Defect model: In this paper, we use for defects only opens, mod-opens for gates and shorts according to the IEEE Standard Working Group P2427[1]. We do not use parametric defects, while this would be in principle possible with the same methodology.

IV. Equivalence Checking

The two circuits under equivalence checking are each described mathematically by a system of implicit differential algebraic equations (DAEs):

$$\begin{aligned} \boldsymbol{f}(\boldsymbol{x}(t),\dot{\boldsymbol{x}}(t),\boldsymbol{u}(t)) &= \mathbf{0}. \\ \boldsymbol{y}(t) &= \boldsymbol{r}^T \cdot \boldsymbol{x}(t) \end{aligned} \tag{1}$$

where $\boldsymbol{x}(t)$ is a vector of n system variables $x_i(t), i \in 1..n$, $\boldsymbol{u}(t)$ is the vector of n_i input variables $u_k(t), k \in 1..n_i$ of the circuit, $\boldsymbol{r}$ is a selection vector and $\boldsymbol{y}$ are the n_o output variables of the circuit. This system of equations can be set up by a Modified Nodal Approach (MNA) in our case automatically by the underlying circuit simulator [11].

The equivalence checking algorithm from [9] linearizes both systems in each state space point and maps them to a Kronecker Canonical Form (KCF) [12]. These KCFs are further reduced by a dominant pole order reduction [13]. This results in a n_z-dimensional canonical state space for both circuits with a state vector $z \in \mathbb{R}^{n_z}$ and a corresponding transformation $\boldsymbol{x} = \mathbf{F}_z \cdot \boldsymbol{z}$. The transformation matrix $\mathbf{F}_z \in \mathbb{R}^{n \times n_z}$ consists of the first n_z eigenvectors.

As an example consider Fig. 2 with an OTA connected as a first-order lowpass. The number of finite eigenvalues is 9. The first eigenvalues are real: $[-4 \cdot 10^{-5}, -8 \cdot 10^{-8}, -5 \cdot 10^{-9}]$. The values indicate that the first eigenvalue is the main, wanted for the functionality of the Gm/C filter circuit. The second and third plus the additional 6 not shown eigenvalues are stemming from parasitic capacitors. Hence, the order will be reduced from 9 to $n_z = 1$.

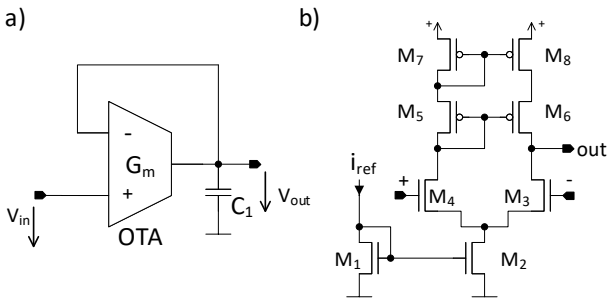

Fig. 2: Running example: One stage OTA connected as a Gm/C lowpass filter. a) Top-level view b) Netlist of the OTA

However, due to the nonlinear circuit characteristic, we have to deal not only with one linearization point in the state space and its KCF but with a whole sampled nonlinear system.

Starting from corresponding DC points the equivalence checking algorithm samples the nonlinear state space by stepping through these mapped spaces by steps calculated from the KCF of the previous point:

$$\boldsymbol{x}_{i+1} = \boldsymbol{x}_i + \boldsymbol{f}_d \cdot \pm \Delta z_d \tag{2}$$

with $d \in 1..n_z$ being the actual sample direction in the nonlinear state space, Δz_d the step size in that direction, $\boldsymbol{f}_d$ the corresponding eigenvector.

Finally, the difference (error) between both circuits in the state space can be formulated as:

$$\varepsilon_y = ||\mathbf{r}_A^T \cdot \boldsymbol{x}_A - \mathbf{r}_B^T \cdot \boldsymbol{x}_B|| \tag{3}$$

with $\mathbf{r}_{A,B}$ being the selection vector for the output variable for each of both circuits A (golden) and B (defective). $||\cdot||$ is an appropriate norm.

On top of the state space sampling a reachability analysis is needed to prevent considering unreachable states.To enable this, the sample points (nodes) are connected by two types of directed edges to form a graph $\mathscr{G}$:

- Input edges: edges between different input values but the same state value.
- Transient edges: edges connecting a start state with a nearest neighbor state modeling the transient behavior in a certain amount of time.

Algorithm 1 Equivalence checking

procedure Equivalence checking($Circ_A$ $Circ_B$)
  set up the nonlinear DAE systems $\boldsymbol{f}_A, \boldsymbol{f}_B$
  for every input value $\boldsymbol{u}$ in given ranges and step size Δu_i **do**
    DC analysis $\rightarrow$ initial state vector $\boldsymbol{x}_{DCA,B}$
    compute KCFs for systems A and B
    for each sample point sp_i in state space **do**
      compute new step size Δx_d and estimate state vectors
      $\boldsymbol{x}_{est} = \boldsymbol{x}_{old} + \mathbf{F_d}\Delta x_d$
      compute consistent operating point $\boldsymbol{x}_{cons}$
      compute the new eigenvector matrix $\mathbf{F}$
      $\boldsymbol{x}_{old} = \boldsymbol{x}_{cons}$
      $S_{All} = S_{All} \cup \boldsymbol{x}_{cons}$
      $\mathscr{G}$ = connect states by input and transient edges(S_{All})
      S_{reach} = reachability analysis(S_{All}, $\mathscr{G}$)
      compute error ε_y
    end for
  end for
  compute energy measure $\varepsilon_{eng,y}$ for each reachable
    state space point $\in S_{reach}$
  T = compute counter example to the reachable state space
    point with the maximum energy error $\varepsilon_{eng,y,max}$
end procedure

On that graph, the *reachable region* is the connected region starting from all DC-operating points. Using the directed graph $\mathscr{G}$, a counterexample can be generated for a given target point $\boldsymbol{z}_t$ by searching a path from a DC point to the given sample point $\boldsymbol{z}_t$. All steps of this EC method are summarized in Alg. 1.

In Fig. 3 the reachable states of the running example are shown. The state space is spanned by the voltage over C_1 (transient edges) and the input voltage V_{in} (input edges). The mesh represents the connections due to reachability. The green cubes are the DC-operating points. The energy measure on the z-axis is explained below.

A. Energy Error Measure

The error on the output defined in (3) is valid for only that point in state space. This is not convenient for testing purposes, because the time measuring that error may be infinitely short. Hence, we propose an error measure that also incorporates the time span, at which the error is visible: an energy-like error measure based on (3):

$$\varepsilon_{y_{eng}} = \int_0^{\tau} \varepsilon_y^2 \, dt. \tag{4}$$

with τ being a maximum tester-specific time to integrate. While this can be easily evaluated on a transient trajectory, in the state space, we have to find trajectories and approximate that measure over these trajectories. However, as we already found counterexamples on the graph $\mathscr{G}$, we can calculate that error measure on these counterexamples by turning the integral into a sum:

$$\varepsilon_{y_{eng}} \approx \sum_{sp_i \in trajectory}^{\sum \Delta t_{sp_{i+1},sp_i} < \tau} \varepsilon_{y,sp_i}^2 \cdot \Delta t_{sp_{i+1},sp_i} \tag{5}$$

with sp_i being a sample point in the state space, ε_{y,sp_i} the deviation at the output between both circuits in that sample point and Δt the time evolving on the transient edge from this sample point to the previous. This measure should be insusceptible to spikes in trajectories and can be compared to a tester-specific threshold.

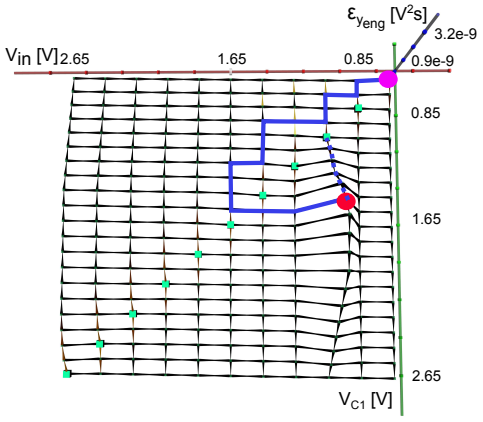

Fig. 3: Energy plotted for each point in state space for drain-source short in M5 in Gm/C filter. The energy at the green DC points is too small to be detected. An ideal stimulus (counter example) has been generated to reach a point with high energy: P (red circle). This stimulus results in a trajectory (blue) starting from a DC point (purple circle) and ending in target Point P (red). The dashed blue line corresponds to the tail ($t > 28\mu s$) of the trajectory.

In Fig. 3 the energy value $\varepsilon_{y_{eng}}$ for the trajectory reaching each given state space point with the highest energy is plotted over the state space. The energy value is calculated for the defect M5-short-drain-source. This defect results in a slightly different cascode voltage for M6 (see Fig. 2). However, M6 is still working as a cascode transistor. Exciting the circuit with DC like signals is not sufficient, as the DC points in the state space (green boxes in Fig. 3) have a low energy value. The point with the largest energy value is marked with the red circle (P). To reach this point P, a dynamic counter-example can be generated and evaluated by simulation. The results from a simulation are shown in Fig. 4. The difference between the output of the defective and the good circuit is shaded purple. The availability of such a generated signal is tester dependent. In principle other signals like sine or ramps could be generated leading to the desired state space point. However we are interested in identifying the principle detectability which is given using the generated step signal.

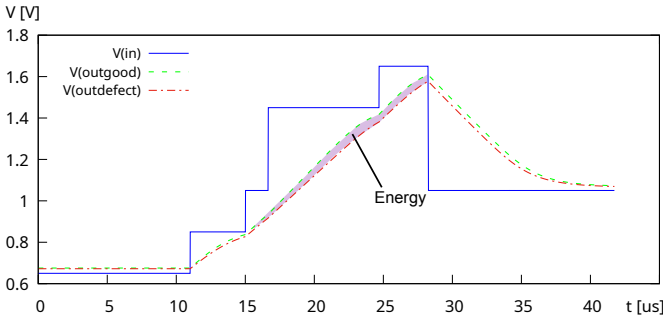

Fig. 4: Response of the defective (red) and the good circuit (green) for a generated input signal (blue). The defect is a drain-source short in M5 in the Gm/C filter. The purple area corresponds to the energy with a value of $3.1 \cdot 10^{-9} V^2 s$.

TABLE I: Statistics of example circuits. The Gm/C filter is the running small example, the others are from [14]

Circuit	MOS Fets	Bipolar	Passives	Defects	Opens	Shorts	Time of standard stimulus	Inputs	Order	
									orig n_{orig}	reduced n_z
Simple Gm/C filter	8	0	2	52	34	18	50us	V_{in}	9	1
Bandgap	40	4	49	438	275	163	110ms †	V_{DD}	38	2
Choppered OP	125	0	2	882	517	365	3.5us ∗	V_{ref},V_{clk}	45	2
SC filter	51	0	2	361	206	155	50us ∗	V_{sc_in},V_{clk}	21	3

† This is the original test time of the benchmark circuit. , ∗ Simple test stimuli.

V. Defect Classification

We define three different classes of defects:

detectable: the detection is proven by a transient simulation,

undetectable: the defect is not detectable (see below),

unprovable: the defect status is unknown, as it can not be assigned to one of the other classes.

Using the equivalence checking algorithm (Alg. 1) with the improved energy error measure from Sec. IV-A, a methodology for identifying undetectable defects can be given:

a) If the maximum energy error $\varepsilon_{y_{eng},max}$ of an equivalence checking of a defective circuit with the golden circuit is above a certain tester dependent threshold ε_{detect}:

$$\varepsilon_{y_{eng},max} \geq \varepsilon_{detect} \tag{6}$$

the defect may be detectable.

b) If the maximum energy error $\varepsilon_{y_{eng},max}$ is lower than the threshold ε_{detect}:

$$\varepsilon_{y_{eng},max} < \varepsilon_{detect} \tag{7}$$

The defect is considered as *undetectable*.

For case a) we generate a counterexample and simulate the defect. If this reveals the defect it is marked as *detectable*. If the counterexample defect simulation fails to reveal the defect, we can not prove the undetectability and also not the detectability. Hence, we mark that defect as *unprovable*.

The reason for that additional check in case a) is a property of the backward stepping state space sampling algorithm: Due to discretization errors and the backward stepping in time, the reachable region may be overestimated leading to unrealistic counterexamples. Hence we check the vadility of the generated counterexample by simulation. For runtime and simplicity reasons, we are doing that only for one counterexample. A possible improvement would be to compute all other.

One other case exists: If the equivalence checking algorithm does not compute a sufficient reachable region - which can be checked versus the reachable region of the good circuit - or aborts for other reasons, and the defect simulation with standard input signals does not reveal the defect, we have also to mark that defect as *unprovable*.

VI. Experimental Results

The method was tested on benchmark circuits from the P2427 group [14] using an Intel Xeon E5 2683 machine with 32 CPUs. We use gnucap [11] as the simulator. Additionally, this simulator has been extended to serve as the EC backplane to calculate operating points and G, C matrix etc. Statistics about the circuits can be found in Table I. The choppered operational amplifier (OP) and the switched capacitor (SC) filter are parts of the low dropout voltage regulator (LDO) benchmark circuit. For each circuit the original order n_{orig} (number of finite eigenvalues) and the reduced order n_z after model order reduction are given.

TABLE II: Results of defect campaign using methodology from Fig. 1

Defect property			Gm/C filt.	Bandgap	Chop. OP	SC filt.
shorts	detectable	stim *	14	100	253	106
		EC †	3	10	41	31
	undetectable		1	35	53	8
	unprovable		0	18	18	10
opens	detectable	stim *	27	96	105	23
		EC †	4	44	94	26
	undetectable		3	53	114	62
	unprovable		0	82	204	95
overall defects n_{all}			52	438	882	361
% undetectable			7.7%	20.1%	18.9%	19.4%
runtime	eq. check. [s]		35	84152	315890	220804
	defect sim. [s]		108	416172	4349980	361052
	wall-clock [s]		18	56372	188681	55586

* Found defects from defect simulation with standard stimulus: n_{stim}.
† From remaining defects found by generated counterexample from equivalence checker: n_{EC}.

The running example (shown in Fig. 2) is analyzed first. The results are shown in Tab. II. It has about 8% of undetectable defects using an energy measure of $\varepsilon_{y_{eng}} > 1 \cdot 10^{-7} V^2 s$. The methodology can decide for all defects if they are detectable or undetectable. Following, no unprovable defects are reported. For some defects, e.g. M2 short-drain-gate, the sine stimulus was not sufficient to find that defect. A stimulus generated by the equivalence checking for these defects successfully reveals their detectability (see rows detectable–EC).

In Fig. 5 the number of identified undetectable defects is plotted versus different energy levels $\varepsilon_{y_{eng}}$. Note that for big limits we get too many defects as undetectable, while for values below $1 \cdot 10^{-6}$ the number of undetectable defects is reasonable. The exact choice depends on the tester.

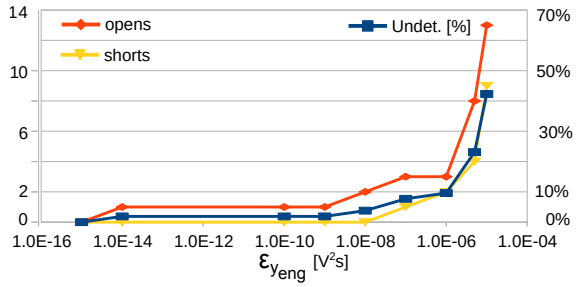

Fig. 5: Analysis of defects identified as undetectable versus energy $\varepsilon_{y_{eng}}$ as detection limit for the Gm/C filter. Plotted are the number of opens, shorts (left axis) and the percentage of undetectable to all defects (right axis).

The second circuit is a bandgap reference circuit. It has a power down mode which we distinguish from the active mode. We perform the analysis for both. A defect will be reported undetectable if it is undetectable in both modes. In general, a power-down network leads to more undetectable defects.

The third circuit is a choppered OP embedded in a control loop for the LDO of the benchmark suite [14]. The two switching states of the choppered OP are each states with a not interrupted feedback loop, leading to two connected parts in the state space. The large number of undetectable defects is also a result of the power-down circuitry, but they are also a result of the switching cells containing transmission gates and from the digital edge equalizer circuits.

The 4th circuit is an SC filter also included in the LDO. It is challenging in generating the state space, as the information (e.g. a difference voltage resulting from a defect) has to be transported using the switches and the capacitors. E.g. an open in a capacitor can only be detected if the input switch carries some new charge onto and the output switch is turned on at some later time point to transport that charge to the output. Consequently, the construction of the state space has to take the clock input into account.

Overall, the results in Tab. II confirm that a substantial portion of the defects can be identified as undetectable by the equivalence checker. Additionally, generated stimuli can reveal a lot of the defects not found by the simple input stimulus.

TABLE III: Comparision of coverages

Coverage		Gm/C filt.	Bandgap	Chop. OP	SC filt.
defect simul.∗	$\frac{n_{stim}}{n_{all}}$	78.8%	44.7%	40.6%	35.7%
+ EC stim. †	$\frac{n_{stim}+n_{EC}}{n_{all}}$	92.3%	57.1%	55.9%	51.7%
+ undet. ‡	$\frac{n_{stim}}{n_{all}-n_{undet.}}$	85.4%	56.0%	50.1%	44.3%
+ EC, undet.	$\frac{n_{stim}+n_{EC}}{n_{all}-n_{undet.}}$	100.0%	71.4%	69.0%	63.9%

∗ Coverage using standard defect simulation. n_{all} is the number of all defects.
† Use beside standard stimuli the counterexamples generated by EC.
‡ Subtract undetectable defects from denominator, where $n_{undet.}$ is the number of undetectable defects.

In Tab. III coverage numbers are calculated for the 4 examples. We compare the standard coverage definition, detected defects by standard defect simulation related to all defects, with the advanced method. The first improvement is to use counterexamples generated by the equivalence checker. The second improvement is to subtract undetectable defects from all defects. Both together lead to a substantial improvement in the coverage (last row in Tab. III).

Identification of undetectable defects with EC leads not only to higher and more reliable coverage, but can also reduce the defect campaign size. As such, a broader defect campaign e.g. Monte Carlo or PVT simulations or embedding these circuits in a bigger system can exclude these defects in defect simulations saving some time.

VII. Conclusion

In this paper, we developed an identification method for undetectable defects proved by an equivalence checking method. This method can prove a defect to be undetectable or provide a counterexample for improving the test stimulus set. In case the equivalence checker fails or aborts, or the proposed counterexample does not reveal a defect, the latter will be marked as unprovable. Results show the feasibility of the approach for small and medium-size analog and mixed-signal circuits and reveal about 20% undetectable defects. The identification method directly increases defect coverage number and may improve the test quality of a circuit by concentrating on follow-up defect campaigns with the EC-generated stimulus.

References

[1] "Draft standard for analog defect modeling and coverage." in *IEEE P2427 Group for Standardization*, 2024. [Online]. Available: https://standards.ieee.org/ieee/2427/7233/

[2] S. Azam, N. Dall'Ora, E. Fraccaroli, R. Gillon, and F. Fummi, "Analog defect injection and fault simulation techniques: A systematic literature review," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 43, no. 1, pp. 16–29, Jan. 2024.

[3] S. Sunter, "Analog fault simulation - a hot topic!" in *2020 IEEE European test symposium (ETS)*, 2020, pp. 1–5.

[4] K. Jurga and S. Sunter, "Measuring mixed-signal test stimulus quality," in *2018 IEEE 23rd European Test Symposium (ETS)*, 2018, pp. 1–6.

[5] N. Nagi, A. Chatterjee, and J. Abraham, "Fault simulation of linear analog circuits," *Journal of Electronic Testing: Theory and Applications*, vol. 4, pp. 345–360, 1993.

[6] S. Chakrabarti and A. Chatterjee, "Partial simulation-driven ATPG for detection and diagnosis of faults in analog circuits," in *International Conference on Computer Aided Design. (ICCAD)*. IEEE, 2000.

[7] G. Gielen, N. Xama, K. Ganesan, and S. Mitra, "Review of methodologies for pre-and post-silicon analog verification in mixed-signal SOCs," in *Design, Automation & Test in Europe Conference (DATE)*, 2019.

[8] T. R. Dastidar and P. P. Chakrabarti, "A verification system for transient response of analog circuits," *ACM Transactions on Design Automation of Electronic Systems (TODAES)*, vol. 12, no. 3, May 2008.

[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. Singh and P. Li, "On behavioral model equivalence checking for large analog/mixed signal systems," in *Proceedings of the International Conference on Computer-Aided Design*. IEEE Press, 2010, pp. 55–61.

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

[12] P. V. Dooren, "The Computation of Kroneckers Canonical Form of a Singular Pencil," *Journal on Linear Algebra and its Applications*, vol. 27, pp. 103–140, 1979.

[13] N. K. Vu and H. Q. Nguyen, "Model order reduction algorithm based on preserving dominant poles," *International Journal of Control, Automation and Systems*, vol. 19, no. 6, pp. 2047–2058, 2021.

[14] "Publicly available set of benchmarking circuits for analog defect simulation, courtesy by Infineon Technologies," 2022. [Online]. Available: https://github.com/Infineon/adsbenchmark