An efficient analog fault-injection flow harnessing the power of abstraction

Renaud Gillon*, Enrico Fraccaroli† and Franco Fummi†

*ON Semiconductor Belgium b.v.b.a., Westerring 15, 9700 Oudenaarde, Belgium, *renaud.gillon@onsemi.com*
†Dept. of Computer Science, University of Verona, Verona, Italy, *name.surname@univr.it*

***Abstract*—Designing an IC for functional safety according to the ISO 26262 standard requires one to identify failure modes and their associated probability of occurrence to assess the possible impact on the realization of the safety goals. Fault-injection is a recommended approach for the identification of failure modes. This paper proposes a multi-tier approach relying on the power of abstraction to limit the overall number of fault-injection simulations to realize and to speed-up their execution by switching to more efficient models of computation.**

I. Introduction

Designing an IC for functional safety according to the ISO 26262 standard [1] requires one to identify failure modes and their associated probability of occurrence to assess the possible impact on the realization of the safety goals. Fault-injection is a recommended approach for the identification of failure modes. In the analog domain, fault injection is performed at transistor-level and is a very lengthy process because of the number of faults to inject as well as the nature of simulations which must often extend to the system-level to include the necessary detection steps.

This paper proposes a multi-tier approach relying on the power of abstraction to limit the overall number of fault-injection simulations to realize and to speed-up their execution by switching to more efficient models of computation. The proposed approach illustrated in Figure 1, relies on three abstraction levels:

- "Layout": risks of defects can be located and quantified using Critical Area analysis.
- "Block": layout defects are abstracted into electrical faults used for fault-injection and identification of failure modes at the sub-circuit level.
- "Functional": impact of failure modes is assessed using abstract behavioral models.

II. Methodology overview

A. Layout-based Fault Extraction

Using Critical Area Analysis [2], [3], it is possible to identify the locations of possible defects in the layout and associate a probability to their occurrence. Identifying topological interactions with devices extents and interconnect shapes, the possible defect sites can be linked via the LVS extraction to such schematic elements as nets and device instances.

According to the flow described in [4], each defect type is then associated with a fault model. For defects affecting devices, the fault is injected by substituting the fault-free device model with a specific fault model. For defects affecting interconnects, fault models are instantiated in the netlist as initially harmless parasitic elements which get activated as a fault to realize the injection.

For short-type defects, the element that gets instantiated as support for the fault injection is a capacitor. Upon activation of the fault, it gets replaced by a resistor of properly selected value. For open-type defects, the support element is a small series resistor, which gets replaced by a much larger value when the fault is activated.

It is important to group defect sites of the same type and affecting the same device or set of nets into a single fault model to minimize the number of possible fault injections. For short-type faults and faults affecting devices, this grouping is straightforward. For open-type defects the operation is more involved, as it requires to identify all defect sites that will break-up an electrical net in the same manner. This can be done by identifying linear interconnect segments and the interconnect junctions where more than two segments come together. All open-type defects affecting a single segment can then be assigned to a single parasitic resistance which acts as support for the fault injection.

Layout-based fault extraction is best performed at the chip-level in order to capture faults which might cross hierarchical cell boundaries. However, as shown in the next section, the hierarchical boundaries of functional blocks are actually exploited to enable the grouping of faults into block-level failure modes. For the block-level fault-injection, short-type faults that bridge to an external net are transformed into two localized faults, by substituting the external net with a resistive connection to a high-level and a low-level supply of the block under study.

B. Block-level Fault Grouping

In order to group faults at the block-level, one needs a suitable distance measure allowing to compare the behavior of the block when various faults are injected. Karaca *et al.* [5] proposed to characterize the block behavior by collecting time-domain responses for a set of stimuli. In order to make a characterization that is sufficiently general so as to cover all possible uses of the block in the target design (and possible future designs), it is essential to collect responses for a very large set of stimuli designed to cover the state-space of the circuit block properly.

In this work, a different approach is taken, based on the collection of small-signal matrices at various operating points selected to cover the state-space of the circuit. As De Jonghe *et al.* [6], [7] have shown that such a collection of AC matrices suffices to characterize the non-linear and broad-band behavior

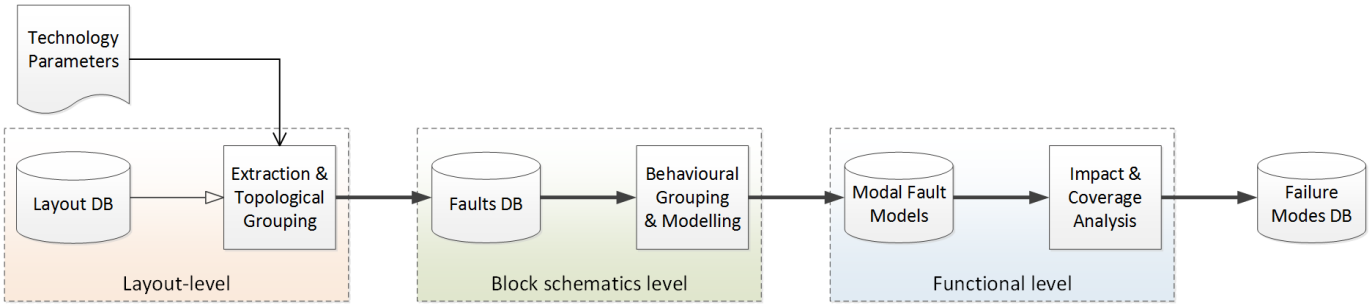

Fig. 1: Methodology overview.

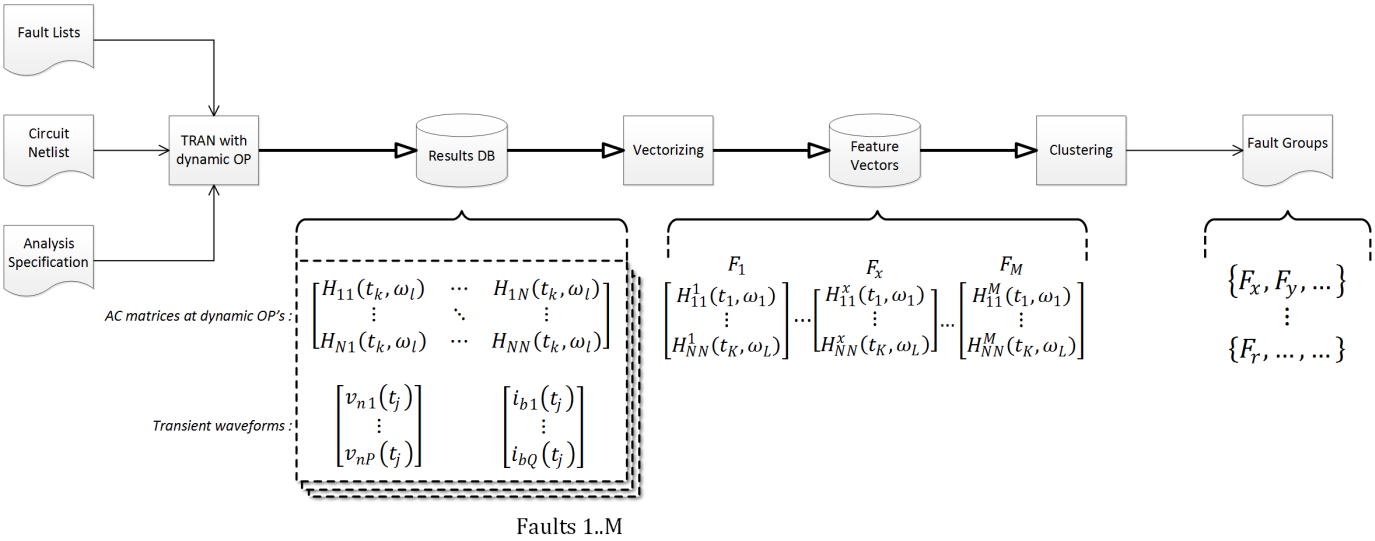

Fig. 2: Fault grouping algorithm.

of circuit-blocks fully, this collection may be used to represent the circuit behavior.

As illustrated in Figure 2, in order to assess the similarity of the behavior induced by various faults, feature vectors are constructed from the elements of the collected AC matrices, and their distance is measured using the Euclidean norm. Faults with similar behavior are grouped, and a set of representative behaviors are selected as a failure mode of the circuit and its probability is determined from that of the underlying faults.

C. Block-level Fault Abstraction

In order to fully benefit from the powers of "abstraction", it is necessary to find a "non-intrusive way" to inject faults at the block-level. Non-intrusive means in this context, that access to internal circuit nodes is not necessary anymore, which then provides the necessary freedom allowing to select the most appropriate behavioral circuit model for the task at hand.

The proposed approach in this paper is that of adding a "wrapper" layer around the fault-free model of the block which transforms the fault-free responses and maps them to the target fault behavior, as shown in Figure 3. This approach was shown to work for the modeling of signal-oriented canonical failure modes by one of the authors in [8].

The method is extended in the present paper, by enabling the extraction of the wrapper parameters from simulated data of the fault-free and the faulty circuit by solving the systems of

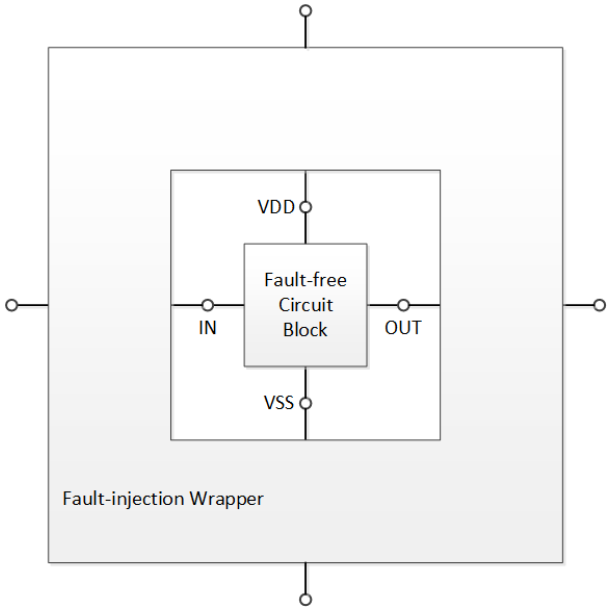

Fig. 3: Schematic of fault-injection wrapper.

equations shown below. Scattering parameters and waves are used instead of voltages and currents as these allow to handle ideal short circuits, opens and through connections without

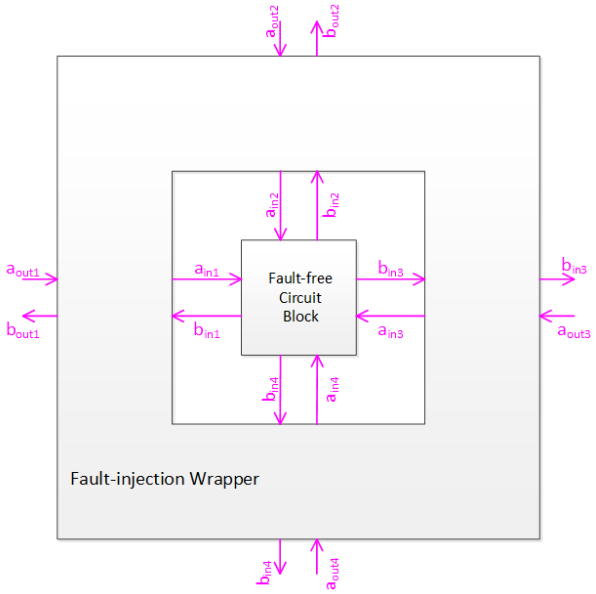

Fig. 4: Definition of key signals for the wrapper.

requiring infinite parameter values [8].

In Equation 7, $\mathbf{S_{FF}}$ stands for the original fault-free scattering-parameters matrix, $\mathbf{S_{FY}}$ stands for a faulty S-matrix (obtained at the external ports of the wrapper), $\mathbf{T_W}$ is a transfer matrix representing the wrapper and the $\tau_{\mathbf{Wxy}}$ represent the deviations of the $\mathbf{T_W}$ from the ideal "thru" case (when the wrapper just reveals the fault-free circuit). The $\tau_{Wxy(i,j)}$ are the unknowns of the system which characterize the wrapper.

$$\mathbf{b_I} = \mathbf{S_{FF}} \cdot \mathbf{a_I} \tag{1}$$

$$\mathbf{b_O} = \mathbf{S_{FY}} \cdot \mathbf{a_O} \tag{2}$$

$$\begin{bmatrix} \mathbf{a_I} \\ \mathbf{b_I} \end{bmatrix} = \mathbf{T_W} \cdot \begin{bmatrix} \mathbf{b_O} \\ \mathbf{a_O} \end{bmatrix} \tag{3}$$

$$\mathbf{T_W} = \begin{bmatrix} \mathbf{T_{Wab}} & \mathbf{T_{Waa}} \\ \mathbf{T_{Wbb}} & \mathbf{T_{Wba}} \end{bmatrix} \tag{4}$$

$$\mathbf{T_{Wxx}} = \mathbf{1} + \tau_{\mathbf{Wxx}} \tag{5}$$

$$\mathbf{T_{Wxy}} = \tau_{\mathbf{Wxy}} \tag{6}$$

$$(\mathbf{1}+\tau_{\mathbf{Wbb}}-\mathbf{S_{FF}}\cdot\tau_{\mathbf{Wab}})\cdot\mathbf{S_{FY}} = \mathbf{S_{FF}}\cdot(\mathbf{1}+\tau_{\mathbf{Waa}})-\tau_{\mathbf{Wba}} \tag{7}$$

Equation 7 defines a systems of equations that relates the wrapper matrix $\mathbf{T_W}$ to the original fault-free circuit matrix $\mathbf{S_{FF}}$ and the faulty matrix $\mathbf{S_{FY}}$ obtained in a similar way as in Section II-B. As the complete behavior of the circuit block is characterized by a collection of fault-free matrices $\mathbf{S_{FY}}(f, OP)$ covering the spectrum and the state-space, it is clear that in order to model an arbitrary fault injection in the circuit, a collection of wrapper matrices $\mathbf{T_W}(f, OP)$ would have to be extracted and then mapped to a behavioral code that reproduces the target behavior in the time-domain using the method of [7].

However, at the present stage of the research, only constant $\mathbf{T_W}$ are being considered, which suffice to capture the following "canonical" failure modes considered for each pin of the circuit, as shown in [8] :

- "shorted pins": resistive connection between pins.
- "floating pin": pin not connected or weakly connected.
- "current offset": short-circuit current error.
- "voltage offset": open-circuit voltage error.

D. Functional-level Fault Abstraction

The final stage aims at simplifying each analog block by anticipating all the computation that would otherwise be done at simulation-time. This process of "analog abstraction", transforms the block-level descriptions to the functional-level [9]. Here the abstraction is combined with the wrappers to sensibly speed-up the fault-injection simulations.

The S-parameter formulation of the wrappers constitutes an advantage in this transformation process as these parameters map naturally to a flow-chart and allow to represent any state of impedance with finite amplitude of the a and b waves.

III. Achieved Results

A. Layout-based Fault Extraction

The layout-based fault extraction was implemented using several tools from the Calibre™ platform from Mentor Graphics [10]. As shown on Figure 5, the flow starts with critical area (CA) extraction and layout-versus-schematic runs, followed by filtering and transformation steps performed on the CA results and allowing to identify interacting nets and devices. In the annotation step, properties are placed on the fault shapes recording the interacting nets, the defect size, the surface of the original CA shape and the technology layer. Then the faults are mapped to original net names from the source netlist and the output netlists are finally produced.

This extraction flow was tested on a series of mixed-signal designs realized in a $0.35\mu m$ BCD-type process. For designs not exceeding hundred thousand components, the chip-level extraction runs in minutes.

An important result is the correct identification of interconnect segments and junctions in analog-style layouts, where the width of traces may vary in very large proportions. Figures 6 and 7 show segments (green outline) and junctions (magenta hash) extracted on a metallic interconnection layer (red hash).

Extensive validation of the layout-based approach is out of the scope of this paper due to the large statistical sample required in order to find enough occurrences of the kind of defects that would cause an apparent fault to occur so as to check the probabilities. However, on a few occasions, defect sites were positively identified during the physical analysis of field returns, and found to match with faults extracted from the layout and ranked sufficiently high in terms of probabilities deduced from the critical area analysis.

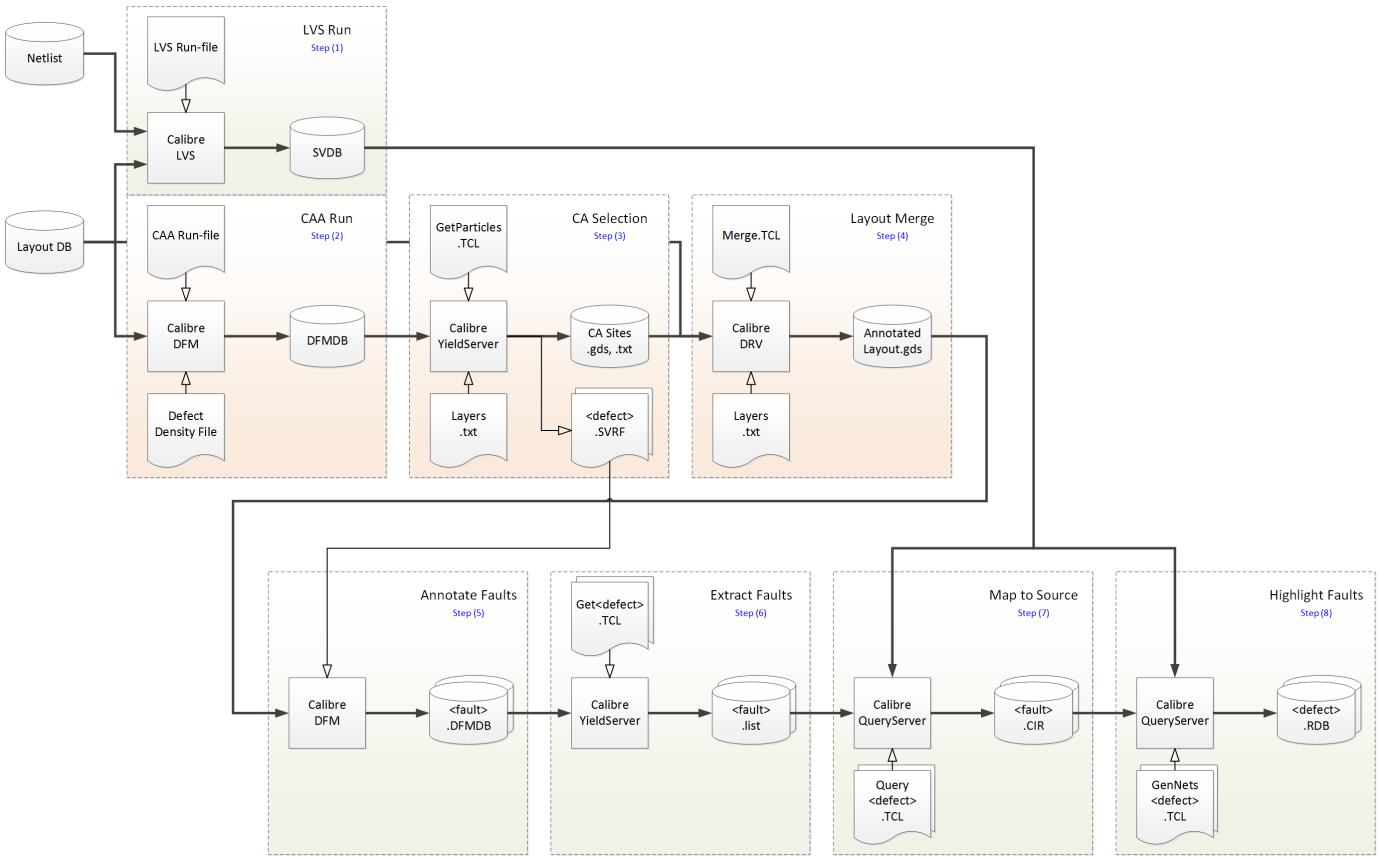

Fig. 5: Layout-based fault extraction flow using Calibre™.

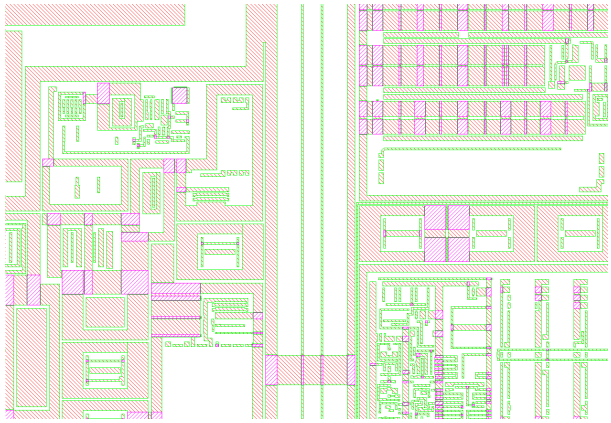

Fig. 6: Intra-layer interconnect junctions (magenta hash), segments (green outline), original metal (red hash).

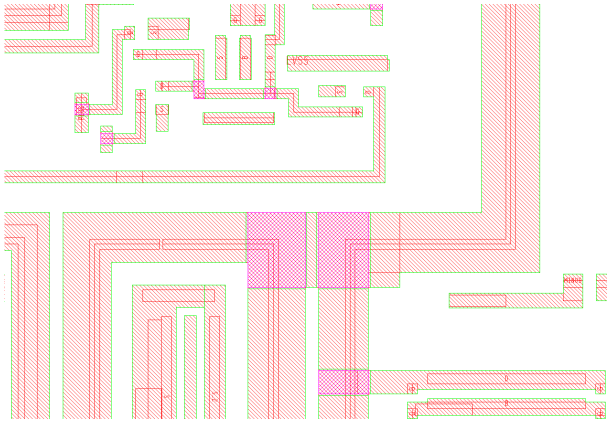

Fig. 7: Details of junctions (magenta hash), and segments (green outline).

B. Block-level Fault Grouping

Experiments were conducted on a few analog circuit blocks. Table 3 summarizes the results obtained on 13 analog circuit blocks. The results show that on average the number of groups allowing optimal clustering of the simulated behaviors is about one tenth of the number of injected faults.

In order to optimize the quality of the clustering, the operation is repeated for a growing series of target cluster counts. The silhouette number $s(p_i)$ is used as optimization goal, where $s(p_i)$ represents one point in feature space, corresponding to the behavior of the circuit with a single injected fault. The silhouette value is computed as the normalized difference between the average distance $b(p_i)$ of point p_i to the closest cluster to which it does not belong and the average within cluster-distance of point p_i to its fellow cluster members $a(p_i)$ (see Equation 8). A silhouette value $s(p_i) \sim 1$ indicates compact and well-separated clusters, with $b(p_i) \gg a(p_i)$.

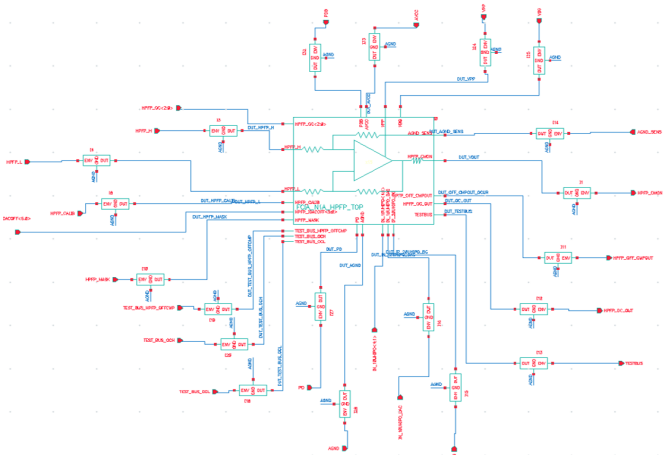

Fig. 8: OTA testbench for AC matrices extraction.

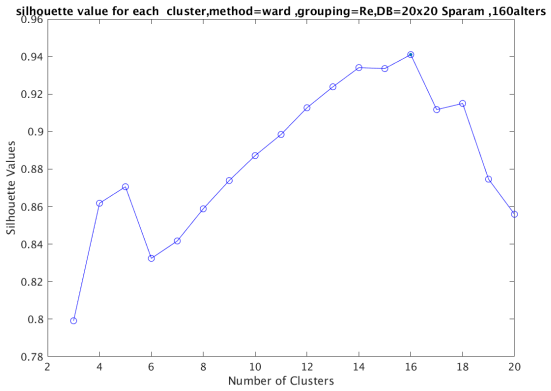

Fig. 9: Silhouette values in function of the number of clusters (OTA test-case).

$$s(p_i) = \frac{b(p_i) - a(p_i)}{max(b(p_i), a(p_i))} \tag{8}$$

Figure 9 illustrates the selection of the optimal number of clusters in the case the OTA circuit shown in Figure 8 where 160 faults were injected. To optimize the total silhouette value which measures the compactness of clusters as well as their separation, 16 clusters are selected. The quality of clustering is shown in Figure 10, revealing silhouette values close to 1.0, meaning well-separated clusters.

The overall quality of the realized fault clustering can also be assessed from the fact that for each of the test-cases, the fault-free behavior was categorized by the algorithm in a separate singleton cluster, avoiding a possible confusion with the faulty clusters. Furthermore, the relevance of the clustering to the time-domain responses of the circuits under test was also systematically assessed. Figure 11 shows output waveforms obtained on the OTA test-case for all fault injections. Responses related to faults belonging to a single group are displayed in the same color. The fault-free case is marked with cross symbols. The faulty responses can always be distinguished from the fault-free one.

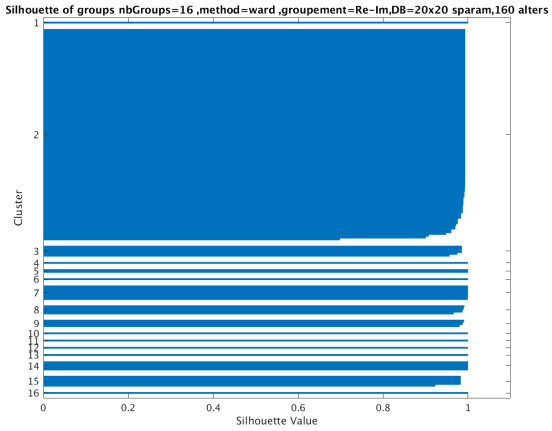

Fig. 10: Example silhouette plot for the optimal number of clusters (OTA test-case).

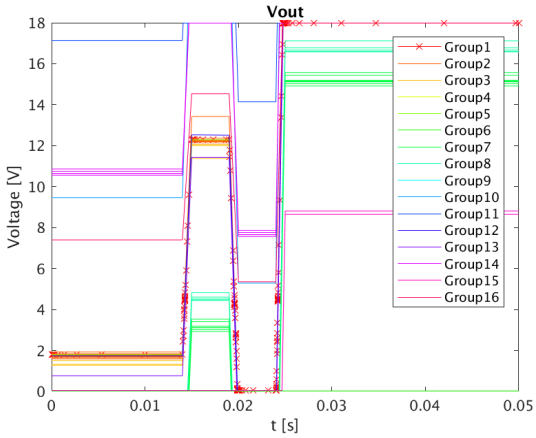

Fig. 11: Output waveforms with fault injections for a testbench sweeping the OTA input rail to rail. Fault-free case in red with crosses.

C. Block-level Fault Abstraction

The extraction of wrapper matrices from fault-injection simulation results was attempted according to the procedure outlined in Section II-C. The fault injections were realized using the canonical failure modes of [8]. Systems of equations corresponding to Equation 7 were constructed and solved using regularization as those systems were strongly underdetermined.

The solutions found to 7 under the constraint of minimizing the magnitude of the $\tau_{\mathbf{Wij}}$ did however not provide satisfactory wrapper implementations. In particular, the wrappers constructed in this fashion appeared to modify too much the operating point of the fault-free circuit, producing time-domain responses not matching those of the target failure mode obtained during the original fault injection.

As the analytical wrapper matrices for the canonical failure modes published in [8] satisfy equation 7 and produce realistic and correct waveforms, it is expected that it should be possible to formulate a series of generic constraints applied to the null-

TABLE I: Overview of the fault-grouping results.

Test-case	Injected Faults	Nbr Clusters	Positive Silhouettes	Fault-free Isolated
Bandgap	399	21	100%	Yes
Enable	248	10	100%	Yes
OD circuit	479	5	100%	Yes
OS circuit	69	25	100%	Yes
Oscillator	176	39	100%	Yes
Power-on Reset	128	6	100%	Yes
Comparator	228	32	100%	Yes
R. Driver	133	11	100%	Yes
Temp. Shutdown	352	30	100%	Yes
V2I	172	21	100%	Yes
WDIO Buffer	218	24	100%	Yes
WDU Driver	198	23	100%	Yes
OTA	160	16	100%	Yes

space of Equation 7 which would produce wrapper coefficients $\tau_{\mathbf{W_{ij}}}$ which honor the target faulty time-domain behavior of the circuit. To this effect a study of the structure of the null-space of Equation 7 is on-going.

D. Functional-level Fault Abstraction

Functional abstraction of a circuit model comprising the original fault-free model and the parametrized fault-free injection wrapper into an equivalent C++ program is a core objective of the present research. So far only abstraction of fault-free blocks was realized with speedups exceeding two orders of magnitude as promising results.

IV. Conclusion

This paper outlined a multi-stage flow harnessing the power of abstraction in order to tackle the issue of the computational burden linked to realizing systematic fault injection. The feasibility of the following critical steps was demonstrated : (a) analog-grade fault-extraction from the layout, (b) fault-grouping at the block-level in analog circuits, (c) wrapper-based fault-injection at block-level for canonical failure modes, (d) functional abstraction of fault-free analog blocks to C++. A data-driven approach for extracting wrapper coefficients from fault-injection simulations was outlined, and the currently open issues were described along with the anticipated solution path.

The main contribution from this paper is most probably the demonstration that clustering based on a collection of small-signal matrices allows performing a meaningful and efficient fault-grouping, providing a 10x reduction in the number of faults to consider for injection.

References

[1] ISO, "ISO/DIS 26262 - Road vehicles – Functional safety," Geneva, Switzerland, Tech. Rep., July 2011.

[2] E. Yilmaz *et al.*, "Fault analysis and simulation of large scale industrial mixed-signal circuits," 2013, pp. 565–570.

[3] D. K. de Vries, "Methods to quantify the detection probability of killing defects," *IEEE Transactions on Semiconductor Manufacturing*, vol. 18, pp. 406–411, 2005.

[4] S. Sunter, K. Jurga, P. Dingenen, and R. Vanhooren, "Practical random sampling of potential defects for analog fault simulation," *Proceedings - International Test Conference*, vol. 2015-Febru, pp. 1–10, 2015.

[5] O. Karaca *et al.*, "Fault grouping for fault injection based simulation of ams circuits in the context of functional safety." IEEE, 2016, pp. 1–4.

[6] D. D. Jonghe and G. Gielen, "Characterization of analog circuits using transfer function trajectories," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 59, no. 8, pp. 1796–1804, Aug 2012.

[7] D. De Jonghe, D. Deschrijver, T. Dhaene, and G. Gielen, "Extracting analytical nonlinear models from analog circuits by recursive vector fitting of transfer function trajectories," in *Design, Automation Test in Europe Conference Exhibition (DATE), 2013*, March 2013, pp. 1448–1453.

[8] R. Gillon, "A framework for analog fault-injection at the behavioral level," in *Guidance of ISO 26262 to Semiconductors USA*. IQPC, June 2018.

[9] M. Lora, S. Vinco, E. Fraccaroli, D. Quaglia, and F. Fummi, "Analog models manipulation for effective integration in smart system virtual platforms," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. PP, no. 99, pp. 1–1, 2017.

[10] R. Gillon, P. Dingenen, and I. Smith, "Layout-based extraction of fault models for analog fault-injection simulations," in *U2U Munich*, 2017. [Online]. Available: https://www.mentor.com/events/user2user