SFP: A METRIC FOR ASSESSING CIRCUIT FAILURE PROBABILITY DURING PRE-SI VERIFICATION

Cristian Manolache[1], Alexandru Guzu[1], Cristina Andronache[1], Alexandru Caranica[1], Horia Cucu[1], Gazmend Alia[2], Andi Buzo[2] and Georg Pelz[2]

The verification phase represents an essential step during the development of new analog integrated circuits. Depending on the circuit complexity and on the verification tools used, this process can take a considerable amount of time. In this work, we propose a novel metric which indicates the probability of circuit failure during Pre-Silicon (Pre-Si) verification. The proposed metric, denoted Specification Failure Probability (SFP), can be used in conjunction with any existing circuit verification algorithm that employs Gaussian Processes (GP) to model the circuit. Since GPs are Bayesian models, apart from their estimates, they also provide an uncertainty, which can be used to assess circuit failure probability. Thus, the verification engineer can make a decision to either stop the algorithm or to continue to perform simulations. The SFP metric has potential of saving a significant amount of time during the most time-consuming phase of circuit development (i.e., circuit verification). In order to evaluate the proposed metric, we conduct experiments on both synthetic and real circuits. The results show that the proposed metric correlates well with the verification outcome and demonstrate the importance of having an intuitive metric in order to better manage the verification process by balancing between failure risk and time spent performing simulations.

Keywords: Pre-Si Verification, Gaussian Process, Machine Learning, Specification Failure Probability.

1. Introduction

As analog circuit designs constantly increase in complexity, the task of circuit verification has become a topic of great interest. A full circuit verification comprises usually two processes: Pre-Silicon (Pre-Si) and Post-Silicon verification. While the former deals with ensuring that a surrogate model of a circuit design works as intended, the latter consists in testing the physical, integrated circuit [1]. An important advantage of Pre-Si verification is that detecting a problem in early development stages will save a considerable amount of time in an already highly time-consuming task [2]. Pre-Si verification may

[1]National University of Science and Technology POLITEHNICA București, Romania, e-mails: cristian.manolache@upb.ro

[2]Infineon Technologies, Munich, Germany

be split into 3 categories: equivalence checking, model checking and specification testing [3]. While computationally expensive, particularly for analog circuits, specification testing can be approached in a more efficient manner with Machine Learning (ML) techniques [4]. Even though data collected during Pre-Si stages can be used for debugging and diagnosing [5], there are a few methods which indicate the reliability of the final pass/fail verdict.

Considering the current stage in the circuit development field, we propose a metric that indicates the probability of circuit failure during the Pre-Si verification process. This metric has been tailored for our method [6], which is a ML approach designed to sample the input Operating Conditions (OCs) hyperspace in order to identify possible circuit failures. To this end, Gaussian Process (GP) surrogate models are employed to create estimates of the circuit's response functions. Based on these estimates, further candidates are proposed for simulation, which could lead to specification violation. An important note on GP is that, besides the estimate of the function, it also provides a measure of uncertainty. This uncertainty can be then used to compute the probability of circuit failure for each response. Benefiting from the inherent GP uncertainty, we can calculate the Specification Failure Probability (SFP).

The remainder of the paper is structured as follows. Section 2 presents an overview of our verification method and highlights the proposed SFP metric. The experimental setup and results are described in Sections 3 and 4 respectively. Finally, Section 5 provides a few discussion items based on the results, while Section 6 is reserved for drawing some concluding remarks.

2. **Proposed Method**

2.1. **Problem definition**

The main task of the circuit verification algorithm consists in identifying possible circuit failures in the Pre-Si verification stage. This ensures that the circuit's response values stay within an imposed range, regardless of the input Operating Condition Configuration (OCC). Through operating conditions (OCs) we denote the circuit's inputs, while an OCC represents a set of OCs values. The circuit's outputs are denoted as responses.

For example, in the case of a circuit with 3 inputs and 2 outputs, we denote the 3 inputs (or OCs) as $cond_1$, $cond_2$, and $cond_3$. An OCC, which is a particular set of these OCs, can look like: $cond_1$=1, $cond_2$=4, $cond_3$=1. The responses represent different functions with respect to the input OCs: $R_1 = f(cond_1, cond_2, cond_3)$, $R_2 = g(cond_1, cond_2, cond_3)$. An example of a response function for the synthetic circuit can be defined as: $R_1 = 2.5^{(6-cond_1)} + 2.5^{cond_2} + 15cond_3$.

Standard verification usually consists in a virtually exhaustive exploitation of the input hyperspace. For example, for a 6 OCs circuit, if we set to explore all minimum, nominal, and maximum points on each dimension, a standard full factorial (FF) verification consists in 3^6=729 simulations. This

usually is very time-consuming and thus, our aim was to develop an algorithm which reduces the verification time by employing a ML based approach.

2.2. **Candidate verification algorithm**

The steps of our current verification algorithm [6] are presented in Fig. 1. An initial OC evaluation set is created using Orthogonal Arrays (OAs) combined with Latin Hypercube Sampling (LHS). The size of this evaluation set depends on the complexity of the circuit, more specifically, the number of inputs (OCs). A full factorial 3^L standard verification, where L is the number of input OCs, is usually feasible up to 5 OCs. For a higher input space, we use various OA designs and complete the remainder budget with LHS samples. This budget usually varies between 100 and 300 simulations. Moving further, all OCCs in the evaluation set are simulated and the response values are extracted. This process represents the Fixed Planning (FP) step, which initiates a preliminary search in the OC hyperspace. In the FP step, we cover predominantly corner cases through the OA designs (minimum, nominal and maximum values are assigned to the OCs), but we explore intermediate values through LHS as well. This offers a good initial coverage of the input OC hyperspace. The FP evaluation set is also used in the second step of the algorithm, namely, the Adaptive Planning (AP). The FP phase is done once, while the AP step is done iteratively according to a preset number of iterations. During AP, we propose additional candidates, or OCCs, that could lead to possible circuit failures. To propose relevant candidates, we use simulations obtained during FP to train GP surrogate models (one GP per response). After the proposed candidates are simulated, the GPs are retrained with the additional data. The algorithm stops either by engineer intervention, or by itself (if failures have been found or if the simulation budget has been exhausted). The aim of the AP step is to propose better candidates, based on the GPs estimates. This way we explore areas of the input hyperspace where standard approaches would usually miss. These newly proposed candidates can be anywhere in the hyperspace (each OC is defined in a continuous interval).

Although quite complex, the verification algorithm provides no indicator as to how many AP iterations would be considered adequate. Furthermore, there can be no universal guidelines as the circuits can vary greatly in complexity and the time spent during verification should be minimized. In order to aid the verification engineer in deciding when to stop the verification process, we propose a metric that indicates the overall probability of circuit failure (see the following section).

2.3. **Specification Failure Probability (SFP)**

We define SFP as a metric which indicates the probability that a response of a circuit might not meet its specification. The SFP is computed using a

previously trained GP model, which can be used to compute the response estimate and variance in each OCC. Fig. 2 illustrates how the GP uncertainty is used to compute the SFP. For a certain data point, highlighted with a purple cross, the Gaussian distribution marks the uncertainty around that specific point. The probability of that data point to violate the imposed specification is marked by the red filled area (SFP) within the Gaussian distribution. Therefore, as the data point on the GP estimate line is closer, or exceeds the specification green line, the red area will increase.

The SFP can be computed as described above for each OCC in the OC hyperspace. However, the verification engineer needs an estimate of the SFP across the entire input hyperspace (i.e., for all OCCs). In addition, it would benefit the verification engineer to know the SFP for the OCCs where the circuit is most likely to fail. Considering this, we compute the SFP starting from a large set of OCCs in order to have a good coverage of the OC hyperspace and thus a good insight of the probability of failure. More specifically, we use the GP estimates for the OCCs as well as the Probability of Improvement (PI) acquisition function in the SFP computation process. The PI function can be described as the probability of an OCC to generate a response which would be better (lower or higher depending on the case) than an imposed threshold (in our case, the specification) based on the GP estimate. The large set of OCCs is formed from a grid of equally spaced OCCs (3^L OCCs, where L is the number of input OCs), to which we add all the OCCs which were proposed during previous AP iterations as being possible worst cases. The 3^L grid setting of minimum, middle and maximum values is used since it offers good coverage of the hyperspace. Using more than 3 levels to calculate the SFP would be computationally and time expensive. We then cluster these OCCs in k=10 clusters, using K-means, and select the worst case (i.e., highest

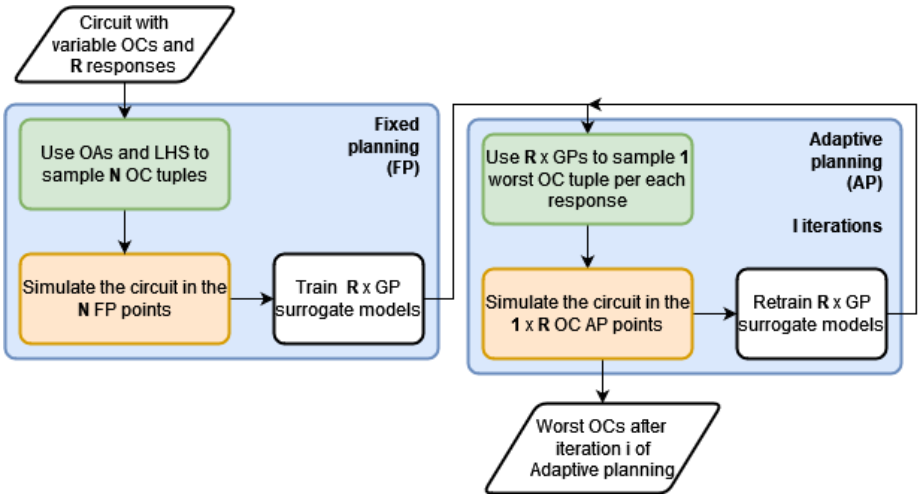

FIGURE 1. Diagram of the circuit verification algorithm

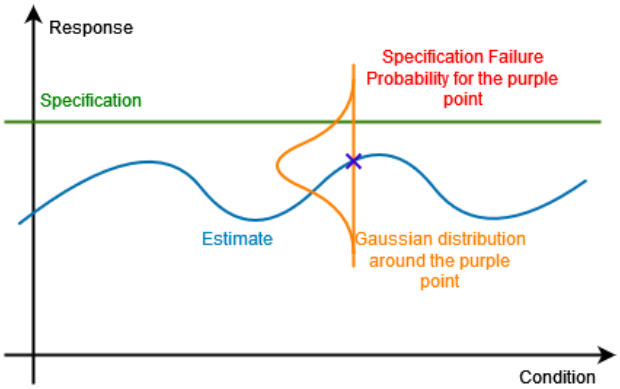

FIGURE 2. SFP for a given data point (purple).

SFP) in each cluster. The aim is to avoid biased results due to highly correlated OCCs. By using K-means to select 10 candidates, the Total SFP metric should better reflect the true specification failure probability. Finally, we combine the individual SFP values for the worst case OCCs to obtain the Total SFP for that response (see Algorithm 1). Firstly, we extract the worst case SFP per cluster. This will lead to a list of 10 Specification Non-Failure Probabilities (SNFP). Then, the Total SFP will be equal:

$$Total\ SFP = 1 - \prod_{i=1}^{10} SNFP_{cluster_i} \tag{1}$$

We note that SFP quality is highly dependant on the GP model. Thus, a better fitted GP is more likely to output a more reliable SFP.

3. Experimental setup

For the experimental setup, we focus on 2 types of circuits: synthetic and real. The synthetic circuit was first used in order to evaluate SFP in a fast (i.e., short function evaluation time) and well established context (i.e., true worst cases are known in advance). Thus, this type of circuit gives a clear view of the reliability of the SFP metric. The synthetic circuit was previously used in our works [6] for setting various benchmarks. The circuit described in [6] has 6 inputs (OCs) and 25 responses. The synthetic functions were proposed by Infineon engineers to be very similar to the way real circuit responses vary with regard to the various operating conditions.

The second circuit, is an Infineon voltage regulator, with 27 design variables, 7 input OCs, one process corner and 3 responses. The three responses are Gain Margin (GM), Phase Margin (PM) and Power Supply Rejection Ratio (PSRR). These must respect the following imposed specifications: GM $>$ 8, PM $>$ 20 and PSRR $<$ –26. We should note that since we are in the Pre-Si

phase, this does not represent a physical circuit, but a model of the circuit simulated through tools such as PSpice or Cadence.

Another aspect regarding the problem of circuit failures is that in our scenario, the circuits are considered black boxes and the inputs are well defined between certain limits and do not stray from those limits. The handling of process corners is done by encoding the categorical values in 2 integer values. This was done in order to be compatible with the BoTorch framework, which we use for our GPs. More details can be found in our prior work [6].

The reliability of the SFP is assessed comparing its value to the known (or estimated) response limit. This takes into consideration values obtained during FP and completed AP iterations as well as true minimum or maximum values (if available). As we obtain closer values to the specification, we would expect higher SFP values, while obtaining a value below/over the imposed minimum/maximum threshold would lead to a SFP of 100%.

4. **Experimental Results**

4.1. **Synthetic circuit with infallible specifications**

The first experiment was performed on the synthetic circuit. Its specifications were set to be 2% lower than the true minimum for all the 25 responses. By setting the specifications slightly below the true minima of the synthetic responses, we make sure they cannot be failed. Thus, we expect to obtain a decreasing trend in the SFP values and eventually values of 0%, as the GPs become more precise after each iteration. The results (see Fig. 3) show that

Algorithm 1: Specification Failure Probability

Input: A set of N OCCs data points comprised from 3^L + *previous AP OCCs* candidates, where L is the number of input OCs

Output: Total SFP value for a response

1 Calculate the SFP value for each data point as the probability of the estimate to exceed the specification.
2 Create 10 clusters using K-means, based on the data points position in the hyperspace.
3 Create the SFP vector of length 10 based on the K-means clusters.
4 **for** $i = 1$ **to** *10* **do**
5 | Add the data point with the best SFP value to the SFP vector
6 **end for**
7 Calculate the Specification Non-Failure Probability (SNFP) as $SNFP = 1 - SFP$
8 Calculate the Total SFP value for the response as: $Total\ SFP = 1 - \prod(SNFP)$

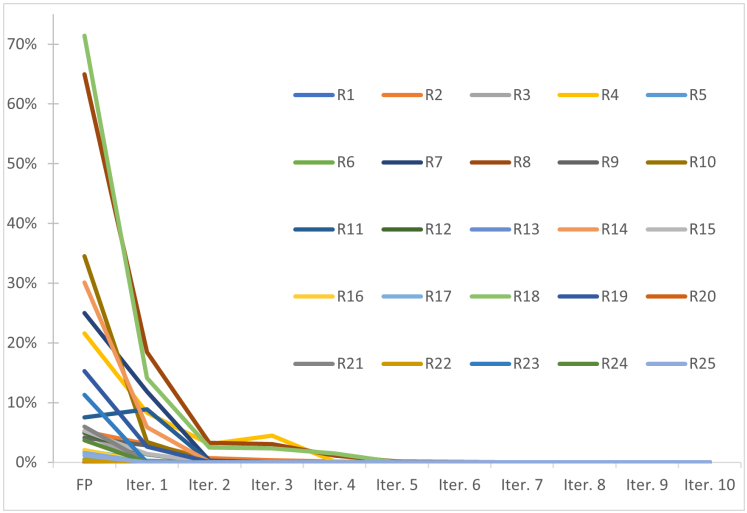

FIGURE 3. SFP values for the 25 responses of the synthetic circuit configured with infallible specifications. Ox axis shows the algorithm phases: it starts with FP and then continues with 10 iterations of AP.

many responses have low SFP values after the FP stage. Eight responses have SFP values higher than 10% after FP and continue to exhibit high values at early AP iterations. However, the algorithm reaches small SFP values for all responses, with a maximum SFP of 1.51%, as early as iteration 4. This indicates very low probability to find OCCs that can generate responses below specification values. This is indeed the case, since the specifications have been set deliberately as infallible. Such behaviour is ideal in terms of SFP: for a well-designed circuit it should point out quickly that there is no reason to spend more time on verification.

4.2. **Synthetic circuit with fallible specifications**

In the second experiment, we set the specifications for the synthetic responses as the true minimum +1% of the response range. This way, the specifications are fallible and we expect to see relatively high values of SFP as we get closer to the imposed specifications. The experimental results are in line with the expectations. Due to the relative simplistic nature of some of the circuit's responses, the verification algorithm managed to find failures for most of the responses during the FP stage. For the responses for which failures were not discovered during FP, we present in Table 1 the specifications values, the worst response values found during FP and the SFP computed after FP. An important conclusion regarding the results presented in Table 1 is that most of the responses have high SFP values after FP. This indicates that the verification engineer should continue to perform simulations for these responses, since there is a high probability circuit failure. Indeed, as it can be observed in

TABLE 1. Synthetic responses for which the verification algorithm did not find failures during FP.

Response	Spec.	Worst FP value	SFP after FP	AP 1 value
4	38.6	38.8	83.2%	38.6
9	51.2	85.8	25.0%	49.5
10	94.9	148.2	78.0%	91.4
11	63.1	83.4	59.9%	61.4
12	80.8	102.8	74.4%	78.0
15	62.9	82.5	84.7%	60.0
16	12.4	19.9	93.0%	9.2
19	47.3	67.5	98.3%	45.0

Table 1, the verification algorithm provides OCCs for which the specifications are failed for all responses in AP iteration 1. Note that for some responses (e.g., 9 and 11), we obtain small SFP values after FP. These are more complex responses for which the GPs are not sufficiently trained with the 100 samples available in FP. Normally, a few iterations of AP would be required in order to take a decision of whether to continue to simulate candidates or stop the verification process.

4.3. **Real circuit (LDO) with realistic specifications**

For this circuit, we initially perform a 10 iteration experiment, with 100 samples in the FP stage. Looking at the results (see Table 2), we notice that GM response has high SFP values from the beginning (FP stage). This indicates high failure probability; 88.2% at iteration 10. The SFP of the PM response starts with a relatively high value of 64.9%, but ending on a less

TABLE 2. SFP for the real circuit responses.

Iter.	SFP GM	SFP PM	SFP PSRR
FP	97.5%	64.7%	7.5%
AP 5	89.3%	15.4%	6.8%
AP 10	88.2%	10.9%	1.1%
AP 15	87.8%	8.2%	1.2%
AP 20	82.1%	3.6%	0.6%
AP 25	76.5%	1.9%	0.1%
AP 30	77.6%	3.3%	0.1%
AP 35	74.4%	2.2%	0.0%
AP 40	62.7%	1.5%	0.0%
AP 45	65.5%	2.6%	0.0%
AP 48	100.0%	6.6%	0.0%
AP 50	100.0%	3.6%	0.0%

pessimistic value of 10.09%; while for the PSRR response we begin with a low value of 7.5% and end on an even lower value of 1.1% indicating that the GP for this response seems almost certain that the PSRR response cannot fail the specification. Given these initial results over 10 iterations, we decide to further simulate candidates in order to find a possible specification failure for the GM response (given its high SFP value). Thus, we sequentially add an additional 10 iterations, while monitoring the SFP values. Since the SFP for GM still has high values, we continue to add iterations up to iteration 50. Looking back at Table 2, we observe that our commitment has been justified, since at iteration 48, the algorithm finds an OCC which leads to a GM value of 7.92, which represents a specification violation (GM needs to be strictly higher than 8). For the PM response we observe a steady decrease of the SFP values over the 50 AP iterations, reaching a value of 28.6 as opposed to the specification of 20. Finally, after initial iterations, the GP model for PSRR should precisely approximate the response and assign under 1% SFP after iteration 20. This is justified by the maximum value found by this response. The found value of -34.1 (compared to the spec. of -26) is relatively far from the response range.

5. **Discussion**

The experimental results showed three possible scenarios:

(1) the SFP is high after fixed planning and decreases steadily during adaptive planning to values close to 0%;
(2) the SFP is relatively high after fixed planning and remains high or increases during adaptive planning;
(3) the SFP is low after FP and remains low during AP.

Scenarios 1) and 3) were observed for the synthetic circuit configured with infallible specifications and the PM and PSRR of the real circuit. In both these cases, the SFP predicted correctly the outcome: the verification algorithm did not find specification failures and could have been stopped early on, saving time. Scenario 2) was observed for the synthetic circuit configured with fallible specifications and for the GM response of the real circuit. Again, in both situations, the SFP is reliable: the verification algorithm identifies specification failures. In a real scenario this would be an important warning for the engineer: continue the verification process as there is a high probability that the circuit fails for some OCCs. Note that SFP cannot be considered an absolute metric; e.g., SFP=50% does not necessarily mean that there is a 50% probability that the circuit would fail the specifications. In order to obtain such an absolute metric one would need to evaluate it in all possible OCCs and with a very accurate surrogate model. The aim of the algorithm consist in finding potential failures using a small number of simulations compared to a standard exhaustive approach, while the SFP offers valuable insight of these failures in a large, albeit limited no. of points in the hyperspace. Overall, the SFP is an objective metric and can be successfully used (i) to compare the likelihood of failure

between circuits, (ii) to provide warnings of possible failures and eventually (iii) to stop the verification early on for adequate circuits. For future work, we plan on testing the SFP metric on more scenarios for both synthetic and real circuits.

6. Conclusions

In this paper we introduced a powerful metric for assessing failure risk at any stage during Pre-Si verification. The specification failure probability (SFP) gives valuable insight to the verification engineer, as it approximates the probability of circuit failure. In this context, SFP can save time that would otherwise be spend on unnecessary simulations or can issue warnings of possible failures early on. The SFP metric was validated on both synthetic and real circuits. The experiments showed various scenarios of SFP values and trends, all of them being correlated with the final outcomes: failure identification or circuit sign-off. Nonetheless, even though SFP can be a valuable tool for the verification engineer, it is worth noting that its values should always be correlated with the quality of the GP surrogate model. As the SFP is based on the GP estimate, these two are intrinsically intertwined.

Acknowledgment

This work was supported by a grant of the Romanian Ministry of Research, Innovation and Digitization, CCCDI - UEFISCDI, project number PN-III-P2-2.1-PTE-2021-0460, within PNCDI III. The scientific paper has been created in the frame of "Important Projects of European Interest on Microelectronics" and collaboration between Infineon and the Faculty of Electronics, Telecommunications and Information Technology at the POLITEHNICA București National University for Science and Technology.

REFERENCES

[1] G. Gielen, N. Xama, K. Ganesan, and S. Mitra. Review of methodologies for pre-and post-silicon analog verification in mixed-signal SOCs. In *Proc. DATE*, pages 1006–1009. IEEE, 2019.

[2] U. Farooq and H. Mehrez. Pre-silicon verification using multi-FPGA platforms: A review. *J. Electron. Test.*, 37(1):7–24, 2021.

[3] S. Deyati, B. Muldrey, and A. Chatterjee. BISCC: Efficient pre through post silicon validation of mixed-signal/RF systems using built in state consistency checking. In *Proc. DATE*, pages 274–277. IEEE, 2017.

[4] H. Hu, Q. Zheng, Y. Wang, and P. Li. HFMV: Hybridizing formal methods and machine learning for verification of analog and mixed-signal circuits. In *DAC*, pages 1–6, 2018.

[5] P. Mukherjee and P. Li. Leveraging pre-silicon data to diagnose out-of-specification failures in mixed-signal circuits. In *Proc. DAC*, pages 1–6, 2014.

[6] C. Manolache, C. M. Andronache, A. Caranica, H. Cucu, A. Buzo, et al. Applying Multi-objective Acquisition Function Ensemble for a candidate proposal algorithm. In *Proc. SpeD*, pages 116–121. IEEE, 2023.