

# Knowing Your AMS System's Limits: System Acceptance Region Exploration by Using Automated Model Refinement and Accelerated Simulation

Georg Gläser, Hyun-Sek Lukas Lee, Markus Olbrich, and Erich Barke

**Abstract** Virtual prototyping of Analog/Mixed-Signal (AMS) systems is a key concern in modern SoC verification. Achieving first-time right designs is a challenging task: Every relevant functional and non-functional property has to be examined throughout the complete design process. Many faulty designs have been verified carefully before tape out but are still missing at least one low-level effect which arises from interaction between one or more system components. Since these extra-functional effects are often neglected on system level, the design cannot be rectified in early design stages or verified before fabrication. We introduce a method to determine system acceptance regions tackling this challenge: We include extra-functional effects into the system models, and we investigate their behavior with parallel simulations in combination with an accelerated analog simulation scheme. The accelerated simulation approach is based on local linearizations of nonlinear circuits, which result in piecewise-linear systems. High-level simulation speed-up is achieved by avoiding numerical integration and using parallel computing. This approach is fully automated requiring only a circuit netlist. To reduce the overall number of simulations, we use an adaptive sampling algorithm for exploring systems acceptance regions which indicate feasible and critical operating conditions of the AMS system.

**Keywords** Parameter space • Acceptance region • Piece-wise linear • Simulation • Modeling • Bordersearch • Mixed-signal • Virtual prototyping • Automated model refinement • Design automation • Extra-functional properties • Accelerated simulation • System level • Verification

---

G. Gläser (✉)

IMMS Institut für Mikroelektronik- und Mechatronik-Systeme gemeinnützige GmbH,  
Ehrenbergstr. 27, D-98693 Ilmenau, Germany  
e-mail: [georg.glaeser@imms.de](mailto:georg.glaeser@imms.de)

H.-S.L. Lee • M. Olbrich • E. Barke  
Institut für Mikroelektronische Systeme, Applestr. 4, D-30167 Hannover, Germany  
e-mail: [lukas.lee@ims.uni-hannover.de](mailto:lukas.lee@ims.uni-hannover.de); [markus.olbrich@ims.uni-hannover.de](mailto:markus.olbrich@ims.uni-hannover.de);  
[erich.barke@ims.uni-hannover.de](mailto:erich.barke@ims.uni-hannover.de)

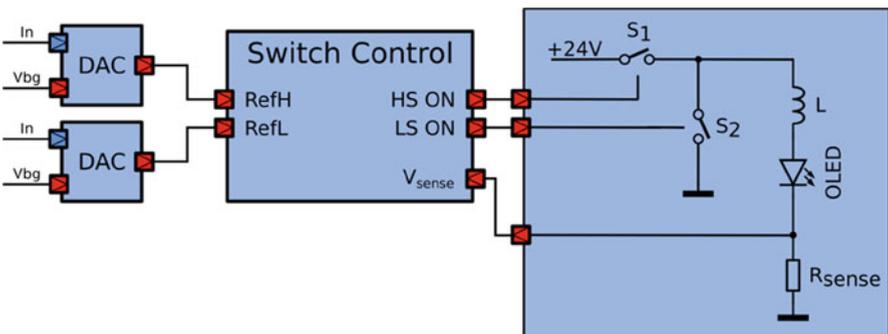
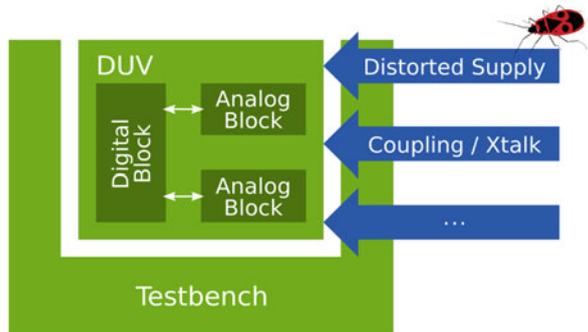
© Springer International Publishing AG 2018

F. Fummi, R. Wille (eds.), *Languages, Design Methods, and Tools for Electronic System Design*, Lecture Notes in Electrical Engineering 454,  
DOI 10.1007/978-3-319-62920-9\_1

# 1 Introduction

Many carefully verified designs fail due to flaws which neither the design nor the verification engineer has identified. As shown in Fig. 1 the design flaw might even be located outside the functional behavior of the system's components: Distorted supplies or parasitic couplings can raise severe problems that are only visible in certain conditions but are crucial to the overall functionality. Such design flaws are usually not covered by abstract system-level models since they neglect low-level effects. Within this contribution, we consider a common DC–DC converter circuit as shown in Fig. 2 for demonstration. This hysteretic current-mode buck converter will always be stable in simulation assuming idealized models and reference voltages [1]. However, its stability can be influenced by distortions not visible on system level. For example, distortions due to ground-bounce or crosstalk from the supply to the reference voltages may cause malfunction. Uncovering these interactions by simulation on transistor or layout level is virtually impossible for such a demonstrator or even more complex systems due to enormous computing times.

**Fig. 1** System-level verification targets at verifying all functional properties using abstract models



**Fig. 2** Hysteretic current-mode buck converter as application scenario. The output current is determined via references generated by digital-to-analog converters (DAC)

In this contribution, we introduce a methodology to determine the system acceptance regions (SAR), i.e., the safe operating regions in the system-level parameter space of the distorting effects. In contrast to process and design parameters, they are not usually included in predefined models since they emerge from parasitic effects not visible on the regarded level of abstraction. We make their parameter space accessible by refining the existing models with parasitic effects using an automated procedure. Its exploration requires a large number of simulations in order to identify the system acceptance regions. This number can be reduced by using a systematic sampling methodology based on the *Bordersearch* algorithm [2]. However, further optimization of the simulation performance is needed due to the analog components in the simulation. We avoid time-consuming numerical integration algorithms by utilizing a piecewise-linear (PWL) modeling scheme that results in a piecewise-linear system. Considering nonlinear system behavior, the switching from one linearized circuit state to another state is a key issue. We have implemented an algorithm for determining the switching points along with multi-core processing for further performance improvement.

Following the discussion of the state of the art in Sect. 2, we introduce a new methodology to refine models with extra-functional properties in Sect. 3. Aiming at a high simulation performance, we use the accelerated analog simulation presented in Sect. 4. Based on the introduced methods, we explain the concept and exploration of system acceptance regions in Sect. 5. In Sect. 6 we demonstrate the method for a designed and fabricated DC–DC converter.

## 2 Related Work

Modern embedded AMS systems have been subject to research for a long time [3]. Still, the challenges arising in today's complex system-on-chips demand new methods for design and verification. Effects like crosstalk causing *signal integrity* issues have been studied [4, 5] especially in the digital domain. Addressing these challenges in analog systems is subject to ongoing research [6] since they are only visible on system level by considering effects from lower levels of abstraction. If the parameters of these effects are not specified at the beginning of the design phase they impose a significant design risk.

The evaluation of the system performance influenced by additional low-level effects requires appropriate models to be implemented. These methods have been studied, e.g., by Alassir et al. [4] and Eo et al. [7]. Introducing these effects automatically into system-level simulation is still uncommon: Fault injection by, e.g., distorting signals with saboteur modules was proposed by Leveugle and Ammari [8] but those approaches lack a general framework for AMS model refinement that could be used for exploring the newly introduced parameter space.

Procedures for extracting fault and acceptance regions were developed, e.g., by Dobler et al. [2] who also discussed the use of *Design of Experiments* based methods [9] in this context. Similarly, methods for extracting feasible regions in parameter

spaces have been developed by Stehr et al. for sizing and optimizing purely analog circuits [10]. However, the use of these methods for extracting acceptance regions in combination with model refinement to gain knowledge about the given system has not yet been published.

Each parameter space exploration algorithm demands for many simulations to be executed—especially for high-dimensional problems. Hence, the simulation performance is crucial to efficiently explore these regions. To speed-up our simulations, we use an accelerated analog simulation approach [11, 12]. This approach uses piecewise-linear models to avoid numerical integration and nonlinear equation solving.

This combination of accelerated analog simulation and automated refinement of component models with extra-functional properties is used to identify the critical scenarios in a AMS system at a very early design stage.

### 3 Automated Model Refinement

A large number of simulations is necessary to reach a sufficiently high verification coverage. Executing these simulations using low-level models on, e.g., transistor or layout level is clearly not feasible due to extreme high computing times. More abstract models try to solve this problem by reducing the simulation complexity by only implementing purely functional properties. However, additional effects as for instance power-supply rejection are usually neglected. The decision which effects can be neglected is crucial: If only a single relevant effect is neglected, the verification might falsely accept the system behavior—whereas the critical effect is not visible in simulation causes the design to fail. Since this set of relevant effects is unique for each design, a flexible modeling approach is needed to adapt to the actual use case.

The implementation and maintenance effort significantly increases if different combinations of effects have to be modeled and their impact on the system evaluated. Consider a system demanding for five effects and their interactions to be regarded. This raises the task of implementing  $2^5 - 1 = 31$  variants of the model to be realized and maintained and urges for an automated approach.

Whereas our approach is not limited to a specific modeling language, we consider an existing system-level component model in SystemC-AMS [13] because of the availability of tools for code analysis. The system refined by an additional effect is shown in Fig. 3. Analyzing the model code automatically using libClang [14] yields structural information—e.g., ports, signals, internal structure, and locations of functions—about the targeted component model and its instances in the overall system. Based on this, a predefined generic text-template [15] is rendered to generate the model code for a wrapper realizing the actual effect. This procedure is repeated for all refinements to be applied to the model. Note that refinements might not be commutative: Consider, e.g., a multiplicative and an additive noise source at a given port of a model or one effect depending on another. In such cases,

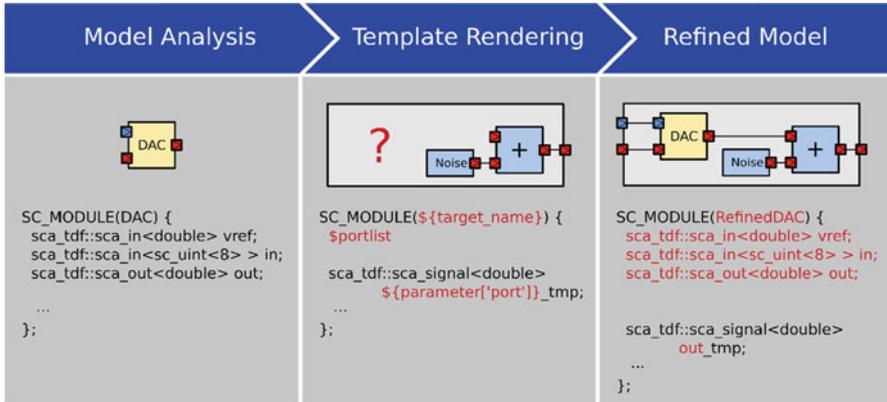


Fig. 3 Refinement flow of a component model

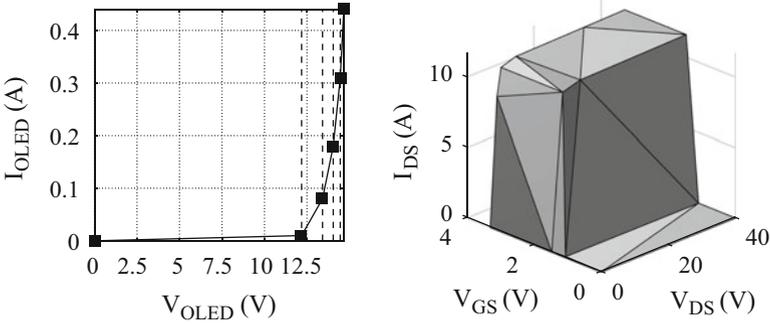
a predefined order of applying refinements has to be ensured. Since the template rendering engine has no knowledge about the nature of the effect, the order has to be defined by the user.

Effects arising in the interaction of several system components cannot be modeled by refining a single component. These effects demand for additional connections to be introduced to the system model. Therefore, it is necessary to modify the given modules by inserting and connecting new ports and signals. In the context of SystemC this results in adding member variables to the classes representing the component's models and connecting them in the constructor code of the top-level module. We realize this direct modification of the model code using information about the code structure obtained by libClang [14].

We limit the approach to refine SystemC-AMS models in this contribution. However, the presented method is also applicable to other modeling languages such as Verilog-AMS as long as the required structural information can be extracted.

## 4 Accelerated Analog Simulation Using Piecewise Linearization

An accelerated simulation of AMS circuits based on piecewise-linear models has been presented in a previous work [12]. Our simulation environment focuses on analog subcircuits as shown in Sect. 1. It provides an accelerated simulation kernel for transient simulations of analog circuits. Speed-up is achieved by avoiding numerical integration and directly using the linear time-domain solution of the system. The time-domain solution is described by sums of exponential terms of the form (1), which can be efficiently evaluated during simulation.



**Fig. 4** Abstract piecewise-linear behavioral model of OLED and low-side switch. (a) OLED behavioral model with five linear sections (*straight lines*). (b) MOSFET behavioral model with 15 linear sections (*triangles*)

$$y_k(t) = y_{k,0} + \sum_i a_{k,i} e^{\lambda_i t} \quad (1)$$

To succeed, this approach requires piecewise-constant inputs (implicitly given by the digital part of AMS circuits) and a linear or at least linearized circuit. We solve the latter problem by replacing all nonlinear devices by piecewise-linear models [16]. Exemplary PWL models for diodes (one-dimensional input:  $V_{\text{OLED}}$ ) and MOS field-effect transistors (two-dimensional inputs:  $V_{\text{GS}}$  and  $V_{\text{DS}}$ ) are shown in Fig. 4. The models are generated by taking advantage of geometric methods [17, 18].

The use of PWL device models results in the generation of multiple linear state-space circuit models describing the analog circuit behavior. The possible combinations of these piecewise-linear models result in a switched-linear systems [19] with different circuit states, which can also be described as a hybrid automaton [20]. It is natural that exactly one state of the automaton is valid at the same time. The modeling approach of static components (i.e., no dynamic nonlinearities) can be applied to nonlinear semiconductor devices as well as to nonlinear macro models. For example operational amplifiers or entire analog driver stages can be treated as a macro model.

Approximating a circuit by a hybrid system with linear continuous dynamics has been used before, see e.g. [21–23]. It is proven and applicable method to control the complexity of system-level modeling. Each circuit state  $v$  corresponds to a discretized state-space representation of the form (2) and (3).

$$\dot{\mathbf{x}}(t) = \mathbf{A}_v \mathbf{x}(t) + \mathbf{B}_v \mathbf{u}(t) \quad (2)$$

$$\mathbf{y}(t) = \mathbf{C}_v \mathbf{x}(t) + \mathbf{D}_v \mathbf{u}(t). \quad (3)$$

## 4.1 Switching Between State-Space Models

To generate simulation results it is necessary to retransform the prepared circuit state to the time domain. The time domain solution yields the monitored output functions. The computed output functions are only valid until the input excitation changes to a new (constant) value or an event was triggered by switching the valid circuit state. It should be noted that all continuity conditions (capacitor voltages and inductor currents) must be fulfilled.

Switching between different PWL models is a significant step for our simulation methodology. The switch-over time depends on the threshold voltages and currents of nonlinear components (e.g., diodes, MOSFETs, and operational amplifiers) and can be determined by finding the first root of the function

$$f(t) = V_{\text{PWLdevice}}(t) - V_{\text{PWLdevice}_{\text{limit}}} \quad (4)$$

in a given interval, where

- $V_{\text{PWLdevice}}$  is the node voltage of a PWL device and
- $V_{\text{PWLdevice}_{\text{limit}}}$  is a limit of the validity range defined by the linear section of each PWL model.

Several root-finding algorithms for such functions are known. We found that the Newton–Raphson method and the bisection method yield unsatisfactory results, as they often do not converge towards the first root and exhibit long runtimes. A specialized root-finding algorithm for this task has been presented in [11]. It guarantees to find the first root in a given interval. This means that for each valid circuit state, which is composed by the combined PWL component model states, the root-finding algorithm must be executed

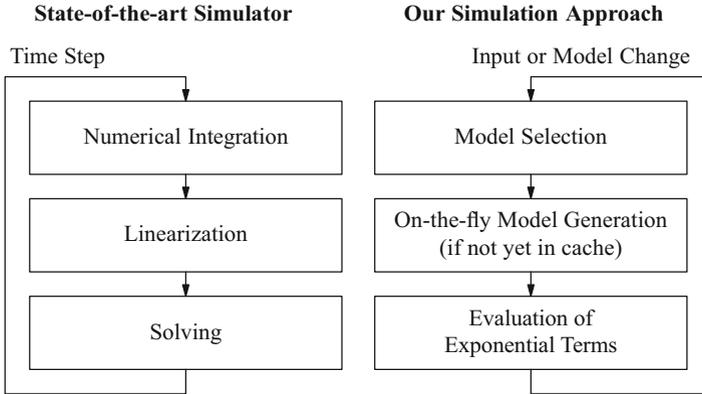
$$R = 2K + 3L \quad (5)$$

times, where

- $K$  is the number of one-dimensional models and
- $L$  is the number of two-dimensional models.

The factors of Eq. (5) yield from the number of PWL section limits. In case of straight lines of a one-dimensional model there are two limits: one upper limit and one lower limit of each linear section. For two-dimensional models exist three limits: all edges of the triangle. After the determination of all first roots in a given interval the earliest switch-over time indicates the next valid circuit model selection.

Figure 5 shows a comparison between our simulation approach with existing analog circuit simulators. Instead of performing numerical integration, linearization and solving the system of equations during each time step, our approach is only sensitive to input changes and internal model switching. As mentioned before, a circuit model switch is triggered by a transition from one linear section of a PWL



**Fig. 5** Comparison between the state of the art and our analog simulation flow

component to another. The active linear section, selected by the simulator kernel, remains valid as long as no change at any input occurs and the circuit does not exceed the current section due to its dynamics. In case of an input change, a new valid section must be calculated along with its new initial values to satisfy the continuity of inductor currents and capacitor voltages. In contrast to an input change, the dynamics of the circuit make switching to an adjacent linear section necessary.

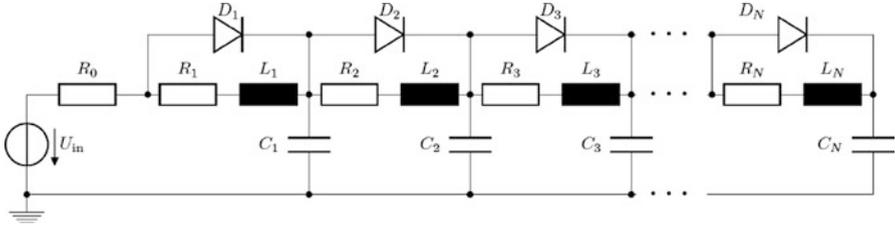
## 4.2 Parallelizing the Specialized Root-Finding Algorithm

Most of today’s advancements in the computing power of processors result from higher levels of parallelization. The processing power available to sequential working threads grows comparably slowly. A sequential program can only use a fraction of the theoretical processing power of most state-of-the-art processors. Therefore, parallelization is of growing importance for performance-critical applications like simulations.

Simulating a large analog circuit including a large number of nonlinear devices causes a long runtime. The root-finding algorithm must be executed very often which can be seen from Eq. (5). The determination of all first roots can be processed independently within a simulation run. For such cases the computations can be run in parallel to speed-up the simulation.

We demonstrate our simulation approach for a scalable nonlinear transmission line (NLTL) with  $N$  stage, see Fig. 6.

In case of a NLTL<sub>128</sub> for each circuit model change the root-finding algorithm must be executed 256 times. Table 1 shows different simulation speed-ups caused by different degrees of parallelization within the simulation kernel.



**Fig. 6** Example of nonlinear transmission line (NLTL<sub>N</sub>)

**Table 1** Simulation runtimes of circuit NLTL<sub>128</sub>

	Tend	Speed-up
Our approach (single threaded)	74.87 s	1.00×
Our approach (multi threaded: 2 threads)	43.01 s	1.74×
Our approach (multi threaded: 4 threads)	26.43 s	2.83×
Our approach (multi threaded: 8 threads)	17.26 s	4.34×
Our approach (multi threaded: 16 threads)	14.73 s	5.08×

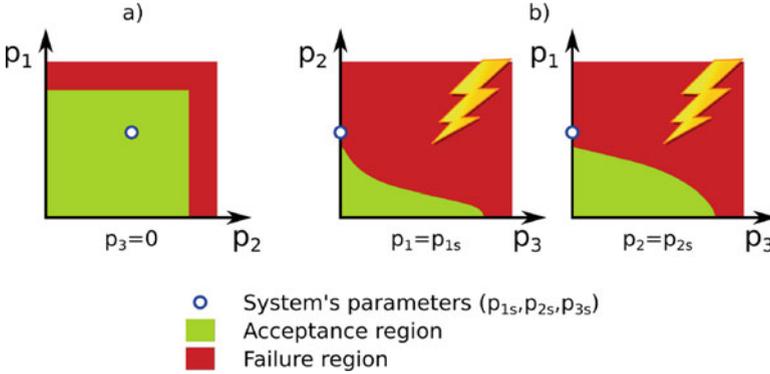
In case of the demonstrated buck converter circuit merely including three non-linear element the parallelization approach is unfortunate. We found that due to the additional parallelization overhead multi-threaded simulations are profitable only for larger circuits.

## 5 System Acceptance Regions

Acceptance and fault regions are well known in Integrated Circuit (IC) testing, e.g., in *Shmoo plotting* [24]. The concept is shown in Fig. 7: For each point in the parameter space, the corresponding system’s behavior is classified into *correct* or *incorrect* using simulations or measurements. The *system acceptance region*, i.e., the typically continuous region with correct behavior, represents all parameter combinations ensuring the system’s function. Note that the border between acceptance and fault region might be *fuzzy* due to stochastic influences on the system (e.g., noise sources) [2].

In this contribution, we assume the system model’s parameters to live in an acceptance region. Still, additional parameters not included in this model might cause the final system to fail. Extracting these cases in a conventional verification is hardly possible due to a missing specification value to test for and a missing model to test with.

These critical scenarios emerge, if an additional effect severely interacts with the parameters included in the simulation. This situation is shown in Fig. 7: The system with parameters  $p_{1s}$ ,  $p_{2s}$ , and  $p_{3s}$  is in the acceptance region taking only the first parameters into account as shown in Fig. 7a. The additional effect with parameter  $p_3$  results in incorrect system behavior due to its interaction shown in Fig. 7b. Hence,



**Fig. 7** System acceptance regions example: a system may live in an acceptance region for two parameters  $p_1$  and  $p_2$  but may be distorted by an additional effect with parameter  $p_3$ . (a) Without  $p_3$ . (b) Acceptance region with  $p_3$

it is important to know the shape of the SAR: Based on the knowledge of the shape, an additional test can be implemented to reduce the design risk. Additionally, the shape unveils the interactions between different parameters. In the figure, the region in the first plot could be parameterized independently for each  $p_1$  and  $p_2$ . The other plots show interactions between parameters, i.e., the description of the regions must include this relationship.

Our goal is to extract the acceptance regions for effects not present in the system-level model by using automated model refinement as introduced in Sect. 3. Based on this, we propose an extraction flow as shown in Fig. 8. After analyzing the given model of the system, the components are refined with the targeted properties. For performance reasons, the analog portions are substituted by PWL models as described in Sect. 4. The extraction of the SAR is done by sampling of the newly introduced and possibly multidimensional parameter space. Since naive sampling strategies are clearly not feasible in higher dimensions, an adaptive strategy is needed. We utilize the *Bordersearch* algorithm described by Dobler et al. [2] for reducing the number of parameter combinations to be simulated.

This algorithm aims to model the border between acceptance and fail regions. Based on this, it automatically selects the parameter combinations to be evaluated close to this border. This adaptive sampling significantly improves exploration quality and runtime especially for high-dimensional problems.

## 6 Application Scenario

For demonstration, we examine the design of a hysteretic buck converter [1] for driving organic LEDs (OLED, organic light emitting diode) shown in Fig. 2. As a first step, the system-level model is created to verify properties of the functional

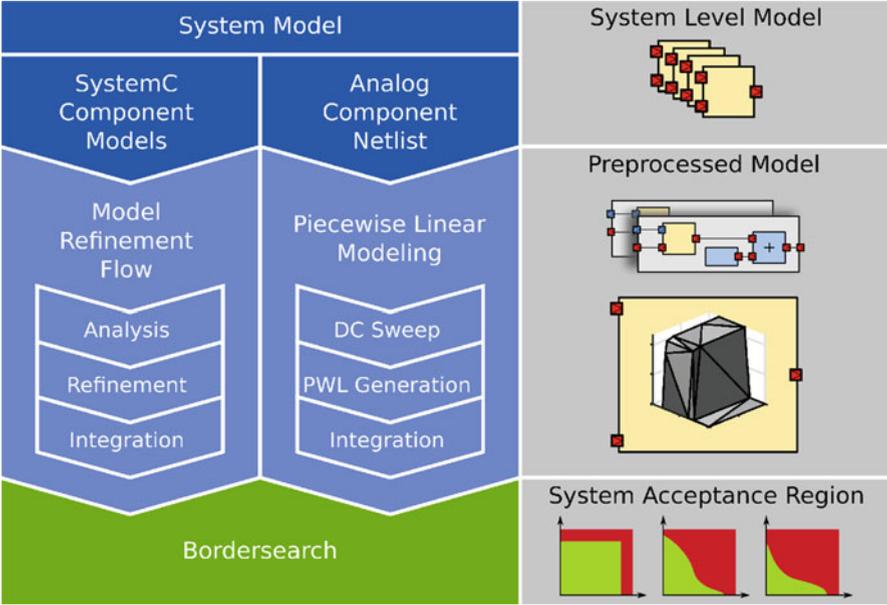
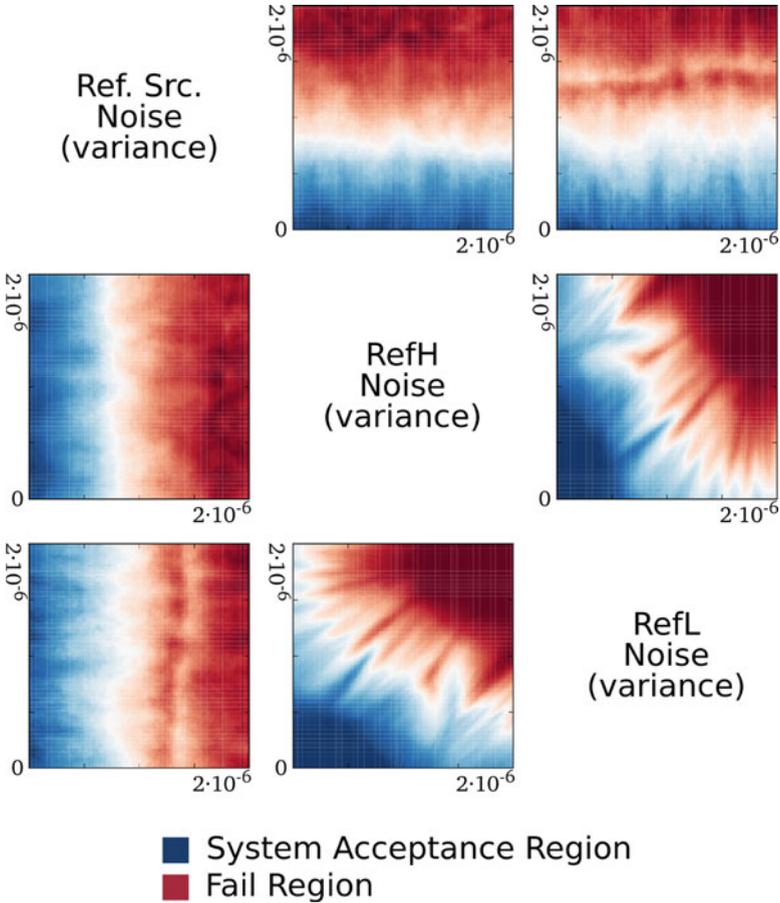


Fig. 8 System acceptance region extraction flow

behavior—in this case, the stability of the overall system. In this context, short circuit between supply and ground through  $S_1$  and  $S_2$  must not occur and the switching frequency must be limited to a certain value. This has been done, e.g., by Dietrich et al. [1] with idealized components. This autonomously switching, externally controlled system has not yet been analyzed formally under influence of low-level parasitic effects. We want to evaluate this question by simulation, since a fabricated test chip for finding out the most critical effects is clearly unwanted for economical reasons (Fig. 9).

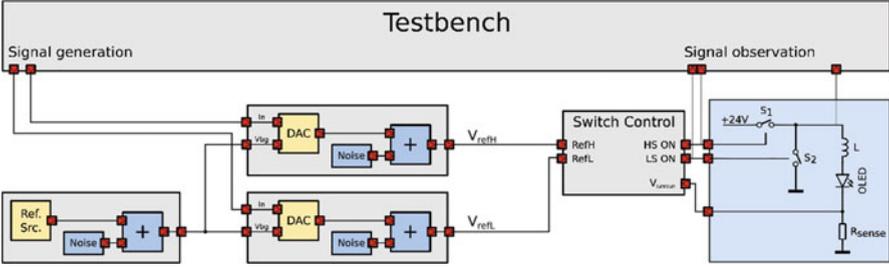
We create the simulation setup as shown in Fig. 10 using the methodology presented before. The reference voltage generators (reference and DAC circuits) are refined by additional effects. Additive white noise is used here to model various distortions. A testbench block generates stimulus signal exercising the system in usual use-cases while observing internal signals. The observed waveforms are checked by the testbench for instable behavior.

To assure the stability of the final system, we explore the acceptance regions for all additional parameters. The simulation runtimes for different exploration strategies are shown in Table 2. Both *Bordersearch* and PWL simulation make this exploration feasible with rather high accuracy. For rough estimates, the simulation runs can be reduced even further. The estimated computing time for normal equidistant sampling of the parameter space is clearly not applicable due to the extremely high number of points to be simulated.



**Fig. 9** Projection of 3D system acceptance region of noise-variances of distortion models

In Fig. 9, we show 2D-projections of the estimated three-dimensional acceptance regions for illustration. Here, we examined the variance of the annotated noise-effects as parameters between 0 and  $2 \times 10^{-6}$ . The axis labels are given by the horizontal or vertical captions. For example, the upper-right plot axis are given by *RefL Noise (variance)* as *x*-axis and *Ref. Src. Noise (variance)* as *y*-axis. The region exhibits typical behavior for noise-effects: The border between acceptance and fail regions are not sharp and the projection of the 3D SAR to 2D also contributes to this effect. In the design process, these regions give the designer a deep insight into the impact of effects to the system, as for instance the interaction distortions of RefH and RefL. This provides the designer with the knowledge for extending the specification and verification plan with checks for the position in the parameter space. Moreover, possibly occurring trade-offs in the design can be evaluated at a very early point in the design process for enhancing the overall system performance.



**Fig. 10** Preprocessed model of hysteretic buck converter circuit for acceptance region exploration

**Table 2** cpu time of acceptance region exploration based on different simulation approaches and sampling strategies

	Reference	Sampling	Bordersearch
Number of points	1	$1 \times 10^6$	$1 \times 10^4$
CPU time (our approach)	2.09 s	580 h	5.8 h
CPU time (Saber)	54.3 s	628 d (est.)	6.28 d (est.)

## 7 Conclusion

In this contribution, we addressed the challenges in virtual prototyping of AMS systems achieving first-time right designs. We proposed a method for automated modeling of extra-functional effects for efficient system-level simulations. It provides system acceptance regions for AMS systems using automated refinement of component models. Since the simulation runtime needed for this process is very high even for small AMS systems, we integrated an accelerated and parallelized simulation approach. Applying the proposed method in later design phases is also possible but challenges the method by even higher complexity due to more signals and possible effects.

For demonstrating our methodology, we examined a DC–DC converter circuit. In this circuit, we regarded exemplarily distortions to generated reference voltages to evaluate their impact on the system’s stability. The extracted acceptance regions show interactions between the effects introduced in the refinement process. This provides the design and the verification engineers with information about critical scenarios and crucial points to avoid or test for.

In future research, this information could be extracted in a more automated way by examining the shape of these regions. This can also be used to reduce the dimensionality of the parameter space to be explored: If the interactions are known, they could be possibly treated in several groups separately. Even a ranking of the criticality of extra-functional effects can be realized.

## References

1. Dietrich, S., Sandner, H., Vanselow, F., Wunderlich, R., & Heinen, S. (2012). In *2012 IEEE 10th International New Circuits and Systems Conference (NEWCAS)* (pp. 369–372). doi:10.1109/NEWCAS.2012.6329033.
2. Dobler, M., Harrant, M., Rafaila, M., Pelz, G., Rosenstiel, W., & Bogdan, M. (2015). *2015 Design, Automation Test in Europe Conference Exhibition (DATE)* (pp. 1036–1041).
3. Kundert, K., Chang, H., Jefferies, D., Lamant, G., Malavasi, E., & Sendig, F. (2000) *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 19(12), 1561. doi:10.1109/43.898832.
4. Alassir, M., Denoulet, J., Romain, O., & Garda, P. (2013). *IEEE Transactions on Components, Packaging and Manufacturing Technology*, 3(12), 2081. doi:10.1109/TCPMT.2013.2262151. <http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=6531653>.
5. Bai, X., & Dey, S. (2004). *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 23(9), 1355. doi:10.1109/TCAD.2004.833612. <http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=1327675>.
6. Barke, E., Fürtig, A., Gläser, G., Grimm, C., Hedrich, L., Heinen, S., et al. (2016). *2016 Design, Automation Test in Europe Conference Exhibition (DATE)*.
7. Eo, Y., Shin, S., Eisenstadt, W. R., & Shim, J. (2002). *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 21(12), 1489. doi:10.1109/TCAD.2002.804381.
8. Leveugle, R., & Ammari, A. (2004). *Proceedings Design, Automation and Test in Europe Conference and Exhibition, 2004* (Vol. 1, pp. 590–595). doi:10.1109/DATE.2004.1268909.
9. Rafaila, M., Decker, C., Grimm, C., Kirscher, J., & Pelz, G. (2010). *2010 Forum on Specification and Design Languages (FDL 2010)* (pp. 1–6). London: IET.
10. Stehr, G., Graeb, H. E., & Antreich, K. J. (2007). *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 26(10), 1733. doi:10.1109/TCAD.2007.895756.
11. Zaum, D., Hoelldampf, S., Olbrich, M., Barke, E., & Neumann, I. (2010). *2010 Forum on Specification Design Languages (FDL 2010)* (pp. 1–6). doi:10.1049/ic.2010.0158.
12. Hoelldampf, S., Zaum, D., Neumann, I., Olbrich, M., & Barke, E. (2011). *2011 IEEE International Systems Conference (SysCon)* (pp. 527–530). doi:10.1109/SYSCON.2011.5929046.
13. Barnasconi, M., Einwich, K., Grimm, C., Maehne, T., Vachoux, A., et al. (2013). *Standard system ams extensions 2.0 language reference manual*. Accellera Systems Initiative (ASI).
14. clang: a C language family frontend for LLVM. <http://clang.llvm.org/>.
15. Mako Templates for Python. <http://www.makotemplates.org/>.
16. Hoelldampf, S., Lee, H. S. L., Zaum, D., Olbrich, M., & Barke, E. (2012). *Proceedings of IEEE International SOC Conference (SOCC)*.
17. Garland, M., & Heckbert, P. S. (1998). *Proceedings of IEEE Visualization (Vis)* (pp. 263–269)
18. Lindstrom, P., & Turk, G. (1999). *IEEE Transactions on Visualization and Computer Graphics*, 5(2), 98.
19. Bemporad, A., Ferrari-Trecate, G., Morari, M., et al. (2000). *IEEE Transactions on Automatic Control*, 45(10), 1864.
20. Lee, H. S. L., Althoff, M., Hoelldampf, S., Olbrich, M., & Barke, E. (2015). *2015 20th Asia and South Pacific Design Automation Conference (ASP-DAC)* (pp. 725–730). Piscataway, NJ: IEEE.
21. Chua, L., & Deng, A. C. (1986). *IEEE Transactions on Circuits and Systems*, 33(5), 511. doi:10.1109/TCS.1986.1085952.
22. Chen, W. K. (2009). *Feedback, nonlinear, and distributed circuits* (3rd ed.). Boca Raton, FL: CRC Press/Taylor and Francis.
23. Zhang, Y., Sankaranarayanan, S., & Somenzi, F. (2012). *Formal methods in computer-aided design (FMCAD), 2012* (pp. 196–203).
24. Baker, K., & von Beers, J. (1996). *Proceedings of the International Test Conference, 1996* (pp. 932–933). doi:10.1109/TEST.1996.557162.