Coverage Measures and a Unified Coverage Model for Analog Circuit Design

Andreas Fürtig and Lars Hedrich
Institute for Computer Science
Goethe Universität
Frankfurt a. M., Germany

Walter Hartong and
Louis-Francois Tanguay
Cadence Design Systems
München, Germany / Montreal, Canada

Markus Olbrich and
Malgorzata Rechmal
Institute of Microelectronic Systems
Leibniz Universität Hannover, Germany

***Abstract*—The metric driven verification methodology using a coverage concept is well-established in digital design. Analog circuits are much more heterogeneous with continuous parameter and state spaces. Therefore, the coverage concept has not been well defined and has not been used for analog circuits so far. Rapidly increasing verification quality demand motivate a new attempt to analog verification coverage. We present a systematic approach to define coverage based verification of analog and mixed-signal circuits. A base coverage model is derived by starting from the fundamental equation system. Based on this, several applied coverage models can be identified. This leads to a set of coverage metric definitions and their relation to real-life verification tasks. Examples illustrate the presented systematic approach and how existing verification tools will fit into the coverage based methodology presented and how upcoming methods like formal analog verification help to increase the verification coverage.**

I. Introduction

'Can't we use the coverage concept known from digital verification in analog as well?' is a typical question asked by verification leads on mixed signal project. It is relatively hard to answer this question. This article will get into the fundamental details and into the background of this question. The term coverage is used in different context, for example in production test / defect analysis. This contribution only considers the circuit verification flow and the definition of coverage in this context.

A. Coverage metric in digital applications

Digital verification has changed and evolved drastically over the last 10-15 years. New languages, tools and methodologies have been invented to increase the verification efficiency at the same rate as the complexity of the functionality is increasing. The metric driven verification (MDV) methodology [1] is the de-facto standard for digital functional verification today. The core idea is a separation of the verification metric (a measurement on how much of a verification goal is achieved) and the actual verification tasks themselves.

As long as the verification metric and the verification goals are correct and complete, the specific verification tasks are less important. This opens up the possibility to use random stimulus and combinations of different verification techniques, such as formal verification, assertion based verification, emulation, etc. [2]. Whichever verification approach is used, the metric will tell how close the process is in relation to the overall verification goal. One essential assumption in the MDV methodology is a good and complete verification plan defining all the verification requirements.

The working style of most analog verification engineers is fundamentally different from the digital environment. The focus is mostly on the performance of the circuit, while the functionality is often not even discussed. The working environment is mostly graphical/UI based. In these environments, analog users setup their simulations, create measurements, set specification, debug waveforms etc. Historically the analog verification approach is very interactive. The 'art' of analog design is often hard to capture in formulas and boundaries. An experienced analog designer can tell in a fraction of a second if a waveform looks right or wrong, while trying to capture this knowledge in a precise measurement can be tedious.

Nevertheless, analog verification is automated more and more. The amount of different modes of operation in the circuits and the different environmental conditions (corners/sweeps) make it more or less impossible for the analog user to manually check everything. Measurements and specifications are used in literally all designs today and the use of assertions and device checks is growing rapidly. This helps the analog designer to focus their attention on the important and complex problems while simple and easy to formalize tasks are automated in the environment.

As mentioned, the core functionality of an analog circuit is often trivial — everybody knows that a voltage controlled oscillator (VCO) creates an output clock with a frequency dependent on the input voltage. However, analog behavior is very sensitive about changes in the environmental conditions: temperature, supply/bias voltage, process variations, matching between devices. Consequently, a lot of the analog verification time is spend in ensuring the correct functionality and performance in all possible scenarios.

What is analog coverage? Analog coverage is not (well) defined. A few approaches can be found in literature [3]–[5], but none of these ideas have been established on a broader scale today. Going back to the fundamental idea: 'Have I done everything that I'm supposed to be doing in verification', analog designer often mention, corners, sweeps and monte carlo variations as the closest definition of coverage.

Due to the limited size of this contribution we could not publish all results. We intend to make them available online [6].

II. From DAE to Base Coverage Model (BCM)

There are several aspects which need to be covered when analog system verification is performed. In this section we will derive these aspects starting from the base analog system model, the Differential Algebraic Equations (DAE) system [7].

A. Mathematical Model

The most fundamental model of the analog design under verification (DUV) is a DAE system. It can be expressed in the general form

$$\mathbf{f}(\dot{\mathbf{x}}, \mathbf{x}, \mathbf{u}, \mathbf{y}, \mathbf{p}) = \mathbf{0} \tag{1}$$

where $\mathbf{x}$ and $\dot{\mathbf{x}}$ are internal states and their derivatives, respectively, $\mathbf{u}$ are input signals, $\mathbf{y}$ are output signals, and $\mathbf{p}$ are system parameters.

Verification makes only sense if the environment of the DUV is defined in which the system must show some expected behavior. This environment does not only provide stimuli but also defines external circuitry connected to the DUV. The environment and excitations are defined using a testbench and the input excitations (signal sources) that can also be described by a DAE system:

$$\mathbf{g}(\dot{\mathbf{z}}, \mathbf{z}, \mathbf{u}, \mathbf{y}, \mathbf{p}) = 0 \tag{2}$$

Here, $\mathbf{z}$ and $\dot{\mathbf{z}}$ are internal states of the testbench and their derivatives. The signals $\mathbf{u}$ and $\mathbf{y}$ build the connections between the DUV and the testbench; u is generated by the sources in the testbench.

Additionally, boundary (starting) conditions for the state values must be given: $\mathbf{x}(t=0) = \mathbf{x_0}$ and $\mathbf{z}(t=0) = \mathbf{z_0}$. The verification task means to assure that given system properties remain within specified ranges. Since system properties can be extracted from the system states, we can formulate the verification criteria as

$$\mathbf{h}(\mathbf{x}, \mathbf{y}) \geq \mathbf{0} \tag{3}$$

which results in a pass/fail decision.

B. Base Coverage Model (BCM)

In the following we will assemble the aspects that can be covered during the verification process. To complete a verification process (the so called verification goal) several verification tasks (like a DC sweep simulation) are needed. Our target is to identify a list of independent (orthogonal) dimensions, for which in each dimension a key aspect of a verification task can be assigned independently from the others. These dimensions build our proposed base coverage model (BCM):

Definition 1: *Base Coverage Model (BCM)*

- *I Model (abstraction level) ($\mathbf{f}$)*
- *II Environment and excitation (Eq. (2))*
- *III Condition*
  - *a) PVT parameters ($\mathbf{p}$)*
  - *b) Initial states ($\mathbf{x_0}$)*
- *IV Criteria (Eq. (3))*
  - *a) Measurement (part of Eq. (3))*
  - *b) Output (component of $\mathbf{y}$)*
- *V Solver*

Depending on the effects that are investigated, different system representations with certain simplifications and levels of abstraction are suitable. The BCM supports this by allowing a variety of different functions $\mathbf{f}$ in the model (I) dimension. Several testbenches together with input stimuli (II) can be applied to the DUV to assure that it fulfills the specification in different scenarios. An operating and environmental condition (III) is a combination of PVT parameters (IIIa) and initial states (IIIb). Therefore, PVT parameters and initial states build a two dimensional subspace inside the BCM. The verification criteria (IV) also build a subspace: They consist of a combination of measurement (IVa) and output (IVb) to which the measurement is applied. A measurement means the observation of a signal and the extraction of signal properties. A measurement can be applied to several output signals (components of $\mathbf{y}$). In general, each measurement should deliver a pass/fail decision in combination with the given specification (see Eq. (3)). A dimension that is not directly referred to the mathematical model is the solver (V) that is used. The solver means a simulator such as transient, AC or noise simulator or even formal methods. This dimension is interesting in that way, that for example a simple netlist can be simulated with different solvers with different forms of abstraction and focus on speedup.

The dimensions of the BCM are independent in a sense that a complete assignment to all dimensions defines a single specific verification point; The BCM spans a verification space

$$B = B_I \times B_{II} \times B_{III} \times B_{IV} \times B_V, \tag{4}$$

where B_i is the set of possible aspects in the i-th BCM dimension, like a corner case parameter variation simulation in dimension B_{III}.

The result of one verification task can cover a subspace of the BCM space or just a single point (*see* Fig. 1). This is an important observation of the BCM: It allows to separate simulation setups (verification tasks) from the coverage of the BCM space (some integral measure over the visited verification points) which is achieved by the verification task.

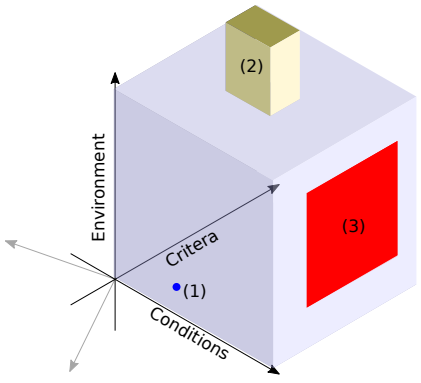

Fig. 1. Example illustration of the high dimensionality of the BCM with only three dimensions (Environment and excitation B_{II}, Conditions B_{III} and Criteria B_{IV}). The result of a verification task can either be a single point (1), an area (3) or a volume (2) in this reduced example.

Obviously, trying to cover everything inside this high dimensional space of all dimensions in B is nearly impossible and matches with the key problem of analog coverage metrics in comparison to digital coverage ones: The continuous value range of real valued variables, signals and states. A truly full coverage for any given (analog) circuit will never be possible and would take a prohibitively large time to compute (*see* Fig. 1). Nevertheless, keeping this in mind helps to create analog coverage metrics. Additionally, we will show that a full coverage of the entire BCM space is not needed to ensure the correct behavior of a DUV. Not all possible verification points in B are reasonable and relevant. Since only the design engineer is able to identify these combinations, he/she needs to define the relevant BCM subspace

$$B_r \subseteq B. \tag{5}$$

C. Coverage Metrics

With the examined relevant BCM subspace $B_e \subseteq B_r$ we define the general verification coverage

$$C = \frac{|B_e|}{|B_r|}. \tag{6}$$

In case of a discrete amount of different possible values in each dimension the cardinal numbers become numbers of verification points. In other words, verification coverage C equals the fraction of examined verification points:

$$C = \frac{\text{\# Examined relevant verification points}}{\text{\# Relevant verification points}} \tag{7}$$

In case of a continuous ("analog") BCM space, verification coverage C equals the fraction of examined relevant subspace volume:

$$C = \frac{\text{Examined relevant subspace volume}}{\text{Relevant subspace volume}} \tag{8}$$

Observing single BCM dimensions leads to specific coverages, e.g. environment & excitation (Dimension II)

$$C_{\text{EE}} = C_{II} = \frac{|B_{II,e}|}{|B_{II,r}|}. \tag{9}$$

III. Additional Coverage Views

To answer the very important question 'when is the overall verification job done' is still challenging. The previously described methods are massively based on the engineers experience and there will be always a high possibility of missing some aspects which can lead to faulty behavior. On the other hand a full coverage of the whole 5-dimensional BCM space is not possible even for small examples. Using a dimension by dimension oriented approach can succeed for each dimension but is not sufficient for the overall verification task.

Fortunately, a compromise can be offered by additional coverage metrics based on cross-sections. We propose to add to the standard BCM coverage views (see Definition 1) the following additional coverage views to be used as metrics for analog circuit verification:

1) Assertion coverage,
2) Code coverage,
3) State space coverage and
4) Specification coverage.

These additional coverage views (incomplete list) are included as cross sections in the yellow boxes in Fig. 2) for visualization. Each additional coverage view can be used to indirectly measure and maximize single or multiple dimensions. The cross sections can appear as partly orthogonal subspaces (e.g. code coverage) or as dimensions in parallel to existing BCM dimensions (e.g. parameter coverage, state space coverage). We will explain them in the following shortly.

For example code coverage can be defined by the following measures:

$$\text{Line coverage} = \frac{\text{\# Visited lines of code}}{\text{\# Executable lines of code}} \tag{10}$$

$$\text{Path coverage} = \frac{\text{\# Visited paths}}{\text{\# Executable paths}} \tag{11}$$

Code coverage measures are mostly available for behavioral models [8]. However, they have slightly different meaning for the analog domain as they argue over code lines which are not part of the DAE system. As analog behavioral models are interpreted in a more block-wise manner which corresponds to parts of the DAE-system and if-then-else blocks can be viewed as different branches of an piecewise defined mathematical function, we map these measure also back to the BCM model. We assume that we have no loops in the code. This coverage measure is also able to rate the quality of the used testbenches.

The from the digital design known assertion coverage counts the number of evaluated assertions during an verification run. An analog state space coverage can judge input stimuli with respect to their ability to drive the circuit into all corners of the state space. Finally the specification coverage can argue over the amount of specifications checked.

IV. From Base to Practical Coverage Model

The most direct way to overcome the complexity of the high dimensionality of the BCM can be a simple verification plan. By defining a set of different tests for each dimension of the BCM, the knowledge of an experienced designer can be combined with the concept of the BCM. With this, the possibility of missing a verification goal is reduced massively. In Fig. 2 a possible sketch of the BCM coverage views and the additional cross section defined by other coverage views are shown. The five blocks define the 5 dimensions of the BCM coverage metric. Each test line (Test X.Y) is a freely definable point proposed in the BCM model (*verification point* or *verification task*). Theoretically the plan can have infinite many lines as we have an continuous BCM-space. Each line corresponds to a coverage task which has its own coverage goal, not executed tasks are indicated by a grey color, successfully executed parts of this task are marked by green, failed parts with red color. The results of the additional coverage views (*see* Sec. III) are computed in parallel and can be part of the result of each coverage task. Depending on the experience of the designers or verification engineers, the

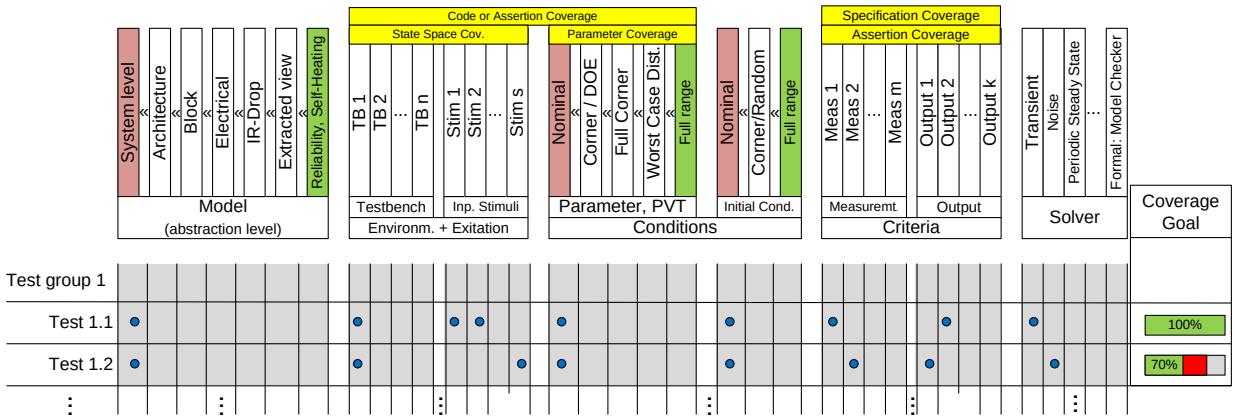

Fig. 2. Five dimensions of the BCM model in a verification plan. Each line corresponds to a verification task. A user defined goal marks the result of each task and is indicated by a blue dot. The progress is indicated by a task bar in the right. Additional coverage views (yellow boxes, *see* Sec. III) can be computed, but must not be part of the user defined goal.

quality of this verification plan can vary. In most cases, each test is a simple pass/fail combination with desired result. It is very important to know, that for every task the expected goal needs to be defined. This is very easy for example for single stimulus for a testbench, but its more demanding for a PVT Corner analyse on a set of stimuli. But the progress of defining the expected goal itself helps to reflect the need of a special task and the feasibility of it. We will show several examples on that in Sec. V.

At least, each verification task must be run once to see if the output fulfills the expectation, otherwise there should be a possibility for changing the expecting result. But again, this helps to increase the confidence into the processed verification plan. It should be clear, that only feasible and reachable tests are useful for adding to the verification plan. This view helps the verification engineer to focus on a well-defined set of verification tasks. In addition, it provides a clear 'done'/'not done' — meaning 'covered'/'not covered' — result.

V. Examples

In this section we will demonstrate our previously defined coverage model on two examples: A more theoretical based consideration of a basic SRAM cell and a more industrial focused example on a state of the art flash memory interface. Both example will show how the proposed BCM can help to increase the designers confidence in his verification effort.

A. Academic Example: SRAM-Cell

To show how to include current state of the art analog coverage metrics from a theoretical view we will create a verification plan for this basic SRAM cell. An excerpt of the final verification plan can be seen in Fig. 3. The usage of different coverage metrics helps to increase the verification confidence of the correct behavior and error free design of an analog block. For each verification task we will describe how this task evolves and explain its specific coverage goal.

Line 1) Coverage task "Read"/"Write"

- **Model**: Electrical netlist description of the circuitry.
- **Environment and Excitation**: The testbench was set to a reading/writing configuration. The stimulus was chosen accordingly.
- **Conditions**: Nominal parameter set for all used components, initial conditions set to $0\,\mathrm{V}$ at $U_{GS,M1}$ and VDD at $U_{GS,M1}$ (1) and vice versa.
- **Criteria**: Read/write the stored values from/to the SRAM as voltages on the corresponding capacitances. The output voltages needs to remain inside the given logic levels during the write/read process.
- **Solver**: Transient Simulation

Coverage goal: Run the simulations and successfully read/write the wanted information from/into the SRAM cell. **Coverage result and interpretation**: These obviously trivial tests are simple cases for the functionality of the circuitry. Simulations were run with expected result, so we can mark both coverage task with $100\,\%$.

In the same manner we have investigated each line corresponding to a coverage task. We will only shortly comment on the differences in the lines: The *coverage task "Abstraction" of Line 2)* checks the correctness of the behavioral model using code coverage, which is indicated in the last column of Fig. 3.

The *coverage task "Disturbance" of Line 3)* tries to proof that the information storage works correct even if an additional disturbance current is flowing into the drain of one transistor of the SRAM cell. The *coverage task "Variations" of Line 4)* is the well known corner simulation. Hence the coverage indicator at the bottom of the column "Parameter, PVT" in Fig. 3 is not at $100\,\%$ as we could also use some advanced methods like worst case analysis or range arithmetics. However, the coverage goal of using corner simulations has been reached. The more academic *coverage task "Formal State Space Analysis" of Line 5)* uncovers an error, indicated by the red bar in Fig. 3. In contrast to 3) we created the state space for each process variation corner. Each resulting state

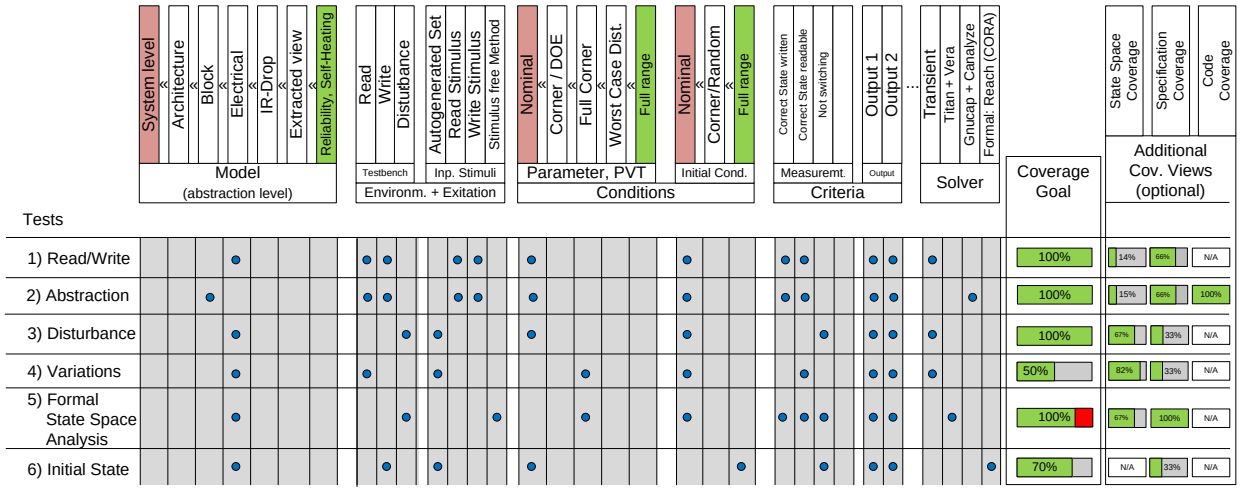

Fig. 3. BCM based verification plan for a SRAM example. Combined dimensions are left out. The verification process for the SRAM cell is not finished yet, but a full coverage can be reached easily. Details for each verification task can be found in Sec. V.

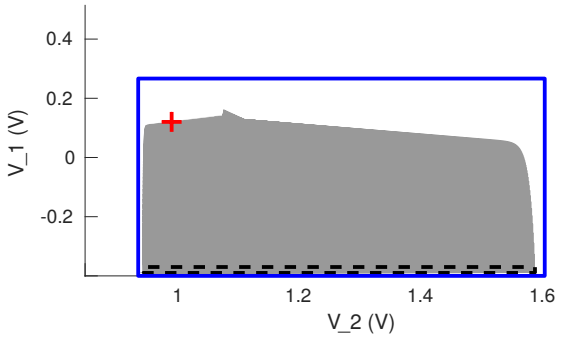

Fig. 4. Reachability Analysis of the SRAM example contributing to initial state space coverage.

space was formally checked using the ASL [9] method to ensure that for each variation and disturbance to the cell always two stable states exists. With a high disturbance current of $I_{dist} = -3\,\mathrm{mA}$ and $I_{dist} = 3\,\mathrm{mA}$ the formal check fails which is indicated by the red part of the coverage bar, showing the designer that sufficiently enough test were run, but some of these test needs further investigation. The also advanced verification method (see [10]) of proving the correctness for all initial states (*coverage task "Initial State", Line 6)* reaches an overall coverage of $70.55\,\%$ (see Fig. 4). In this case the testbench models a disturbed situation and the coverage goal is to cover the blue box in Fig. 4 with the analysis if all states in this region reach the stable point (red cross).

In summary, the verification is not finished yet, however, a full coverage is easy to reach by simply running sufficient simulations for the temperature variation, which will lead to a coverage of $100\,\%$ for Task 4). As Task 5) is based on this step and runs in parallel to these simulations, this task will automatically reach the desired expectation. Task 6) needs some refinement in the coverage goal, as the previous description might be to demanding and not target-oriented with the desired expectation.

In the next example we will show, that the proposed BCM will fit into the current industrial verification process and can be used to raise the confidence in the running verification steps.

B. Industrial Example: State of the Art Flash Memory Interface

To put the analog verification problem in perspective, consider the design of a NAND flash memory interface circuit (NAND IO) implemented in CMOS technology using latest generation FinFet node. NAND IO interfaces contain 4 main blocks: a receiver, a transmitter, a voltage reference, and a calibration block. The receiver can operate either in Current Mode Logic or standard CMOS mode. Performance in these modes must be characterized over various input common-mode voltages, input signal voltage amplitudes, and output loads. The transmitter, with programmability over weak pull up/down resistances, output slew rate, output drive strength, and on-die termination resistance value, must also be characterized over a range of input signal edge rates and output loads. The same applies to the reference voltage generator and the calibration block.

The simulation plan of each block contains a set of common performance and functional simulations that must be run, including EM/IR, reliability asserts, aging and self-heating, power supply sequencing, electrical overstress, as well as functional simulations such as power down mode, digital interface timing, power and leakage, supply noise sensitivity, self-induced supply noise, etc.

The compliance matrix of each block contains 50 electrical parameters on average, and each parameter is associated to one of the tests mentioned above. Simulations with extracted views are required over all PVT corners and, at a minimum, in the five global corners for Monte Carlo mismatch. The prohibitive simulation time of power-up sequencing and data-dependent jitter tests force designers to focus on the corners that show the least margin. To complicate matters even more, deep sub-micron processes show large variations and mismatch.

The current verification approach is similar to the plan described in Fig. 2. The different tasks are listed as lines in a simulation/verification plan. Commercial analog design environments, like Virtuoso ADE Assembler [11] , help the designer to accomplish the simulation tasks and measurements described above. Tracking the block characterization progression and results in a simulation plan and a compliance matrix using a spreadsheet is a time consuming and error prone task. A first generation of commercial analog plan driven verification tools has been introduced. Virtuoso ADE Verifier [12] enables the analog designers to plan the verification and automatically collect the results data (see Fig. 5).

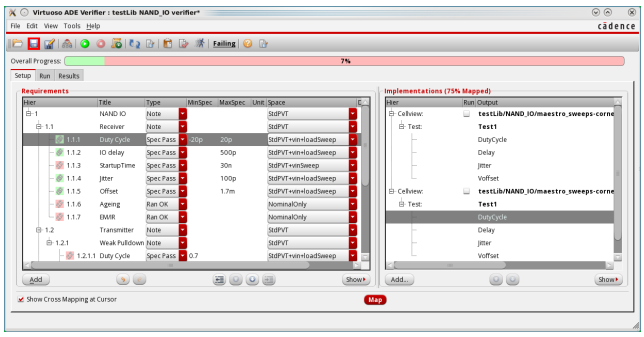

Fig. 5. Verifier setup tab shows the different verification goals for an analog project. In addition to the typical minium/maximum specification values a verification space is defined. The verification space sets the coverage goal in terms of sweeps and corners per requirement.

The concept of creating pre-defined verification goals in addition to tracking the overall results can help the designer significantly in quantifying the verification progress versus the goal. This adds the concept of coverage as presented above into the analog tool chain. Extending the requirements definition in terms of variable sweep values, PVT corners, different analysis types needs to be considered - corresponding to the B_{III} and B_{IV} dimensions of the BCM model. This allows the tool to calculate the coverage of the results in a comprehensive format. Fig. 6 shows an early implementation of this enhancement. The green part of the coverage bar indicates the verification space part which is covered and the specification is passing. The red part is covered but outside of specification and the gray part indicates uncovered parts.

With this capability the designer can handle the complexity highlighted above and ensure all important metrics to be met. Trade-offs between verification effort, quality, and project timelines can be guided by detailed coverage data.

VI. Conclusion

We started from a DAE system as a general analog circuit model and derived a basic coverage model (BCM). The BCM allowed to define coverage measures in a systematic way. By adding practical aspects to the BCM a resulting advanced coverage model was presented. Based on this model we proposed a coverage centric verification method. It consists of (1) defining relevant coverage model sub spaces (coverage goal), (2) performing verification tasks for covering the sub

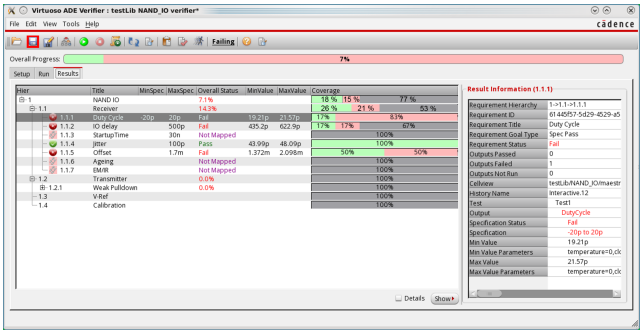

Fig. 6. The results tab of Verifier shows the coverage collection for the given requirements and the set of simulation results. The green/red/gray status bar indicates the pass/fail coverage and the uncovered verification space.

spaces and (3) quantifying the verification coverage. These steps can be performed in an iterative way. Finally, we illustrated the proposed coverage centric verification method by examples using the latest commertial verification tools available and advanced verification techniques like formal analog verification.

References

[1] H. Carter and S. Hemmady, *Metric Driven Design Verification: An Engineer's and Executive's Guide to First Pass Success*. Springer US, 2007.

[2] A. Dahan, D. Geist, L. Gluhovsky, D. Pidan, G. Shapir, Y. Wolfsthal, L. Benalycherif, R. Kamdem, and Y. Lahbib, "Combining System Level Modeling with Assertion Based Verification," in *6th International Symposium on Quality of Electronic Design (ISQED 2005), 21-23 March 2005, San Jose, CA, USA*, pp. 310–315, 2005.

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

[4] M. H. Zaki, S. Tahar, and G. Bois, "Formal verification of analog and mixed signal designs: A survey," *Microelectronics Journal*, vol. 39, no. 12, pp. 1395–1404, 2008.

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

[6] A. Fürtig, W. Hartong, L. Hedrich, M. Olbrich, M. Rechmal, and L.-F. Tanguay, "Coverage Measures and a Unified Coverage Model for Analog Circuit Design (Extended Version)," 2018. https://www.ims.uni-hannover.de/fileadmin/IMS/pdf/analog-coverage.pdf [Online].

[7] R. Riaza, *DAEs in Circuit Modelling: A Survey*, pp. 97–136. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013.

[8] A. Fürtig, M. Paschke, and L. Hedrich, "Comparing code coverage metrics for analog behavioral models," in *Synthesis, Modeling, Analysis and Simulation Methods and Applications to Circuit Design (SMACD), 2017 14th International Conference on*, pp. 1–4, IEEE, 2017.

[9] S. Steinhorst and L. Hedrich, "Analog assertion-based verification on partial state space representations using ASL," in *Specification and Design Languages (FDL), 2012 Forum on*, pp. 98–104, Sept 2012.

[10] H.-S. L. Lee, M. Althoff, S. Hoelldampf, M. Olbrich, and E. Barke, "Automated generation of hybrid system models for reachability analysis of nonlinear analog circuits," in *20th Asia and South Pacific Design Automation Conference (ASP-DAC), 2015*, pp. 725–730, IEEE, 2015.

[11] Cadence Design Systems, Inc. https://www.cadence.com/content/cadence-www/global/en_US/home/company/newsroom/press-releases/pr/2016/cadence-unveils-next-generation-virtuoso-platform-featuring-advanced-analog-verification-technologies-and-10x-performance-improvements-across-platform.html, 2016. [Online, accessed 20-Jul-2018].

[12] W. Hartong, "Plan-based analog verification methodology," 2017. https://www.cadence.com/content/dam/cadence-www/global/en_US/documents/tools/custom-ic-analog-rf-design/virtuoso-plan-based-analog-verification-wp.pdf [Online; accessed 20-Jul-2018].