

Model-driven AMS Test Setup Validation Tool

prepared for IEEE P1687.2

Leon M. A. van de Logt, Vladimir A. Zivkovic*, Ingrid H. A. van Baast

D4T Systems BV

*Cadence Design Systems / D4T Systems BV

Abstract— This paper presents a software package called fanTESTic to improve and speed up the test development process of designs with a significant portion of Analog and Mixed-signal circuits. The test development flow of the tool is based on pre-silicon validation through test setup simulation. The tool maps a number of standard test procedures typically executed at commercial tester platforms, into the simulation domain. Both the automatic testbench generator and post-processing are integrated into the tool, compatible to any commercial Electronic Design Automation (EDA) environment. The approach is also applicable on embedded test instruments. To our best knowledge, fanTESTic is the first commercially available IEEE P1687.2-ready software kit.

Keywords—Analog/Mixed-signal, Test Automation, Test Instruments, P1687.2.

I. INTRODUCTION

Analog mixed-signal (AMS) testing is traditionally seen as a bottleneck during test development and production test for any application [1,2]. As such, it increases the overall design/production cycle and influences the time-to-market. Long and error-prone test development process, difficult product debugging, lack of commercial software tools, high cost of channels for analog test at automated test equipment (ATE) are only a few factors attributed to the mixed-signal test that often cause product re-spin and significant time-to-market delay. Although not new, these factors have only recently unilaterally been recognized as areas of concern in semiconductor community, triggering an IEEE work group P1687.2, to address a number of the above-mentioned aspects through standardization [3].

A number of approaches have been taken in the past to help with AMS test automation and we will list here the most remarkable ones. Spice-based approaches [4,5] achieved little success because the circuit complexity outweighs by far the capabilities of the simulator at the circuit level. A logical step to remedy this situation was to use a various modeling techniques to describe the test at higher levels of hierarchy, i.e. to use the methods employing Hardware Description Languages (HDL). The initial attempt in this direction is presently known as Virtual Test Engineering (VTE) [6]. Virtual Test concept is an attempt to reduce the Time-to-Market by debugging test programs in a simulation environment before first silicon. This first attempt included a very complex synergy between Teradyne, COSSAP and an EDA simulator and was lacking the robustness and portability between different environments. In [7], a test-setup simulation concept employing VHDL-based VTE solution is presented, based on VHDL modeling of the hardware, virtual tester written in VHDL and a snapshot test data extractor linking the test program to a VHDL simulator. However, the approach is not structural and/or hierarchical, i.e., the whole chip is

simulated at once, the method is not portable to another SoC environment and it is lacking the automatic test bench generation. [8] describes an enhancement of the mixed-simulation techniques for optimal use of virtual test, while [9] elaborates further the VTE claiming a significant reduction of the test development time. Nonetheless, their approaches were tester dependent followed with a heavy co-simulation time. A tester independent approach is tackled in [10] engaging the STIL.AMS, but it could not capitalize on it, not only because STIL.AMS standardization efforts eventually grounded to a halt, but also because it is not structural and lacks an automatic test bench generation.

The primary target of the approach in this article is to decrease the test development time for analog and mixed-signal modules while bringing more automation in the test development process, thereby bridging the gap between a DfT/IC designer and a test engineer. To achieve this, we created the Computer-Aided Test (CAT) environment, capable of generating the automatic test bench for simulation while taking into account the industrial test specifications and non-idealities of the automatic test equipment (ATE) or embedded test instruments. To smoothen the data handover between different project team members, post-processing capabilities are also integrated into the tool, to be carried out on the simulation response data. The ultimate goal of the tool suite is the pre-silicon validation of the test specifications using the AMS simulation as if it were running on an ATE platform and is particularly suitable for Big A / Small D applications. The interaction with any commercial circuit simulator has been consistently maintained, while the tool framework itself is prepared to interface with IEEE P1687.2 ICL and PDL deliverables. Tool readiness for P1687.2 has been achieved fast and rather easily because the majority of the current rules and recommendations in the standard were addressed already during the tool creation in the past years.

The second section elaborates on a number of premises and definitions that are typically used in the mixed-signal test and that served as a foundation to build the CAT environment. The tool architecture and P1687.2 readiness are outlined in section three. The fourth section illustrates the flow on a few examples and section 5 emphasizes the benefits through key performance indicators typically used in test engineering. Conclusion is drawn in section six.

II. CAT PREMISES

These premises can be also described as a heuristic, yet this heuristic can be rather seen as a union of a number of practical standard procedures during an arbitrary AMS test.

A. Test Specification aspects

The test specification is a set of requirements defining the test performance, test conditions, and test instrument

equipment to verify proper operation of a Device Under Test (DUT), consisting of the following components:

- *test stimuli* either applied or known, combined with a set of observed responses and criteria for comparing these responses to a known standard or reference.
- *test protocol*, being an ordered series of execution of tests from a related test group (or test bin), containing the test stimuli.
- *test outcome*, which is a mapping from an observation to one set of discrete possibilities to detect all faults and out of tolerance conditions.
- *test program*, which is a direct consequence of the first three components and is defined as an implementation of the tests, test methods, and test protocols to be performed on a DUT to verify conformance with its test specification.

All these components are always applied to test the performance of the DUT in accordance with the specifications and are always valid irrespective whether the testing takes place on actual hardware test setups or in simulation environments.

B. Modelling aspects

The simulation needs to include models of the following components, as illustrated at Figure 1:

- *Device Under Test (DUT)*, that is in practice HDL description of the mixed-signal device at an arbitrary level of abstraction. Simulating the DUT described at the transistor level is often not feasible not only because of the potentially very long simulation times, but also because of the proprietary aspects when design and test are carried out by different parties.
- *Device Interface Board (DIB)* described at an arbitrary level of abstraction and specifying the hardware connections between the DUT and a test instrument, as well as data path and associated connectivity conditions.
- *Instrument* model, that contains descriptions of the hardware connections between instrument ports and the DIB and all relevant information on the instruments

used during the mixed-signal testing from the datasheet. The modelling aspects include e.g. test instrument setup times, triggering, memory capabilities, transmission line characteristics and any other test hardware relevant information. Note that the test instruments cannot be described at transistor-level, for the same practical reasons that are valid for DUT model.

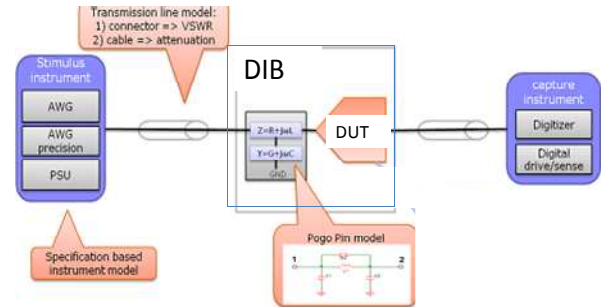


Figure 1. AMS test setup simulation modelling components.

III. FANTESTIC ARCHITECTURE OUTLINE

The tool environment named “fanTESTic” is built upon the premises listed in section 2. This is enabled by linking the test specifications and ATE capabilities to the hardware aspects of the entire test setup using Verilog.AMS [11] testbench as a bridge between the two domains. We have opted to use Verilog.AMS because it is a widely used HDL standard and has inherent capability of modeling the devices and procedures at various levels of abstraction. Using Verilog.AMS rather than Matlab-like approach is further justified by maintaining the direct link to the real-life physical (hardware) connections between IC and tester platform as well as the simulation possibilities in an arbitrary EDA environment. fanTESTic runs on a modern Java-based server within either LINUX or Windows based operating system. The software has a graphical user interface (GUI) to capture the test specifications and design interface. Fig. 2 illustrates the main tool panes. The panes on the left are used to capture the design database together with the interface to DIB and ATE (P1687.2 ICL), whereas the test protocol is described in the panes to the right (P1687.2 PDL).

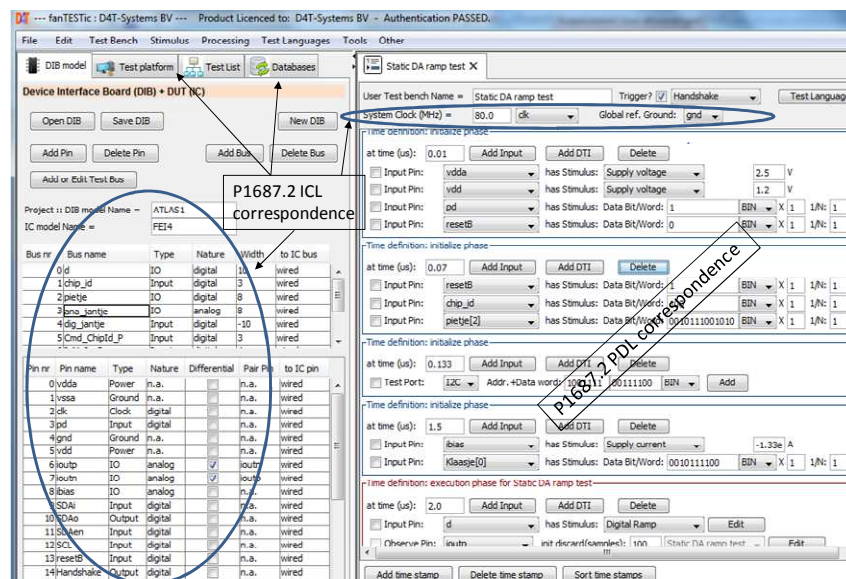


Figure 2. fanTESTic main page.

All ports in the design interface description adhere to P1687.2 prescribed types and contain associated properties, as per standard workgroup ongoing discussions [3]. For example, the tool supports analog, power, ground, clock and digital ports, shown in the left bottom field, together with the information on differential nature. In addition to this, the tool also considers coherent testing through the specification of system clock as well as the definition of global ground (both shown in the upper right pane).

In addition, tool embedded database captures the instrument models (or instrument capabilities as per P1687.2 nomenclature). Fig. 3a shows a database snapshot for one of the typical arbitrary waveform generator (AWG) instruments in a window that pops up after clicking 'Database' tab in the upper part of the left pane in Fig. 2. The instrument model has the following properties

- **Accuracy** on generated values, both absolute and relative based on a 4sigma normal distributed noise model and applicable to horizontal (time) and vertical (voltage, current) axe. The modeling introduces a probability density on the reproduction of a calculated value.
- **Range**: related to the accuracy of the value, the range of the instrument is set as determined by instrument specifications.
- **Settling time** based on pulse response settling model.
- **Slew rate** from ATE characteristic timing.
- **Resolution** as discretization.
- **Sampling rate** and sampling time.
- Mathematical functions to generate arbitrary waveforms for each instrument type.
- Cable and connector models.

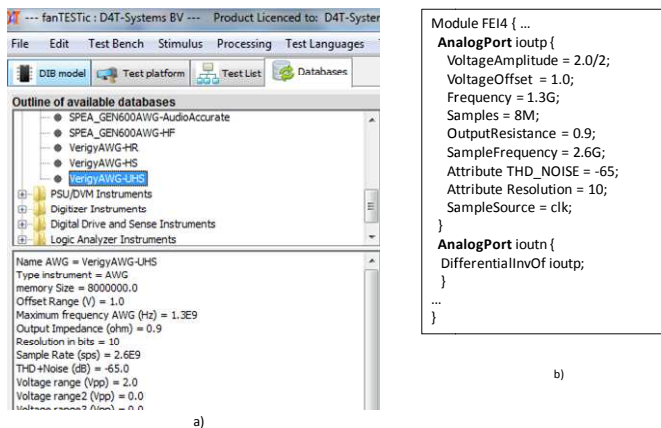


Figure 3. a) Test Instrument Capabilities b) Corresponding ICL excerpt

There is a correspondence between those categories and a number of P1687.2 instrument properties [3], where any difference between fanTESTic and P1687.2 standard property can still be described as an attribute per P1687.2 set of permissions. Once the standard becomes final, the property names will be aligned. Fig. 3b shows an excerpt of an ICL of the design in Fig. 2 emphasizing one of the analog ports connected to an instrument whose properties are described in Fig. 3a.

The concept for modeling the instrument is explained in Fig. 4. The instrument is decomposed in a number of

building blocks where the impacted parameters are determined by instrument spec sheet and introduced as non-ideal behavior into the model.

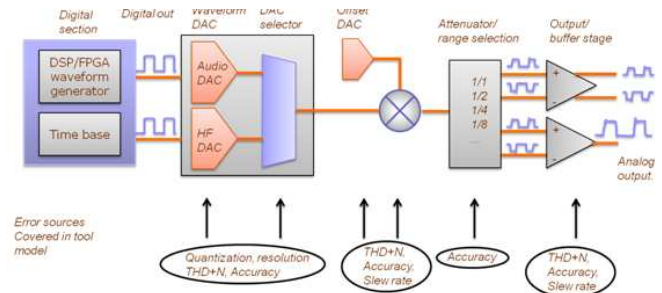


Figure 4. AWG Model Concept for Real-time Test Environment.

In such a way, a generated test stimulus is rather a sequence of discrete states with specified accuracy and noise than an ideal waveform. In the same way the test response capture is limited by the specifications of the given test instrument. The DSP processing capabilities includes the commonly used post processing algorithms as used in mixed-signal test. Amongst others, this includes coherency calculations, FFT analysis, as well as the jitter (rms noise). The test stimuli is typically passed to Verilog.AMS input module as a simulation input, while the simulation output (test response) is typically written to a file and passed to the fanTESTic database for post-processing. FFT analysis and statistical non-linearity calculations are supported for the most common set of AMS testing. Since different manufacturers often apply different interpretations of key parameters, the tool model will use converted values for those parameters depending on availability in the vendor sheets. These settings, though, can be changed by the user in an instrument specific text file.

The right side of the main GUI in Figure 2 captures the actual test protocol and test outcome. It contains an initialization phase followed by an execution phase, as a typical sequence running on an ATE. The initialization phase is completely in hands of the user, nevertheless, assisted by the tool in terms of any signal pre-conditioning in terms of setup, triggering and timing. The tool supports commonly used digital interfaces such as I2C and SPI, typically used in Big A/Small D applications. There is a list of predefined parameters for the execution phase, such as e.g. input test frequency, sampling rate, number of cycles, amplitude, offset, coherent sampling etc, determined by a user-selectable test instrument. Fig. 5 shows such form for the sinusoidal DSP signal.

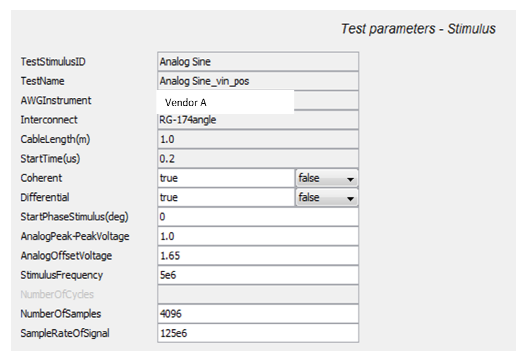


Figure 5. Execution Phase form.

The tool takes care of the timing mechanism that will be generated and inserted into the testbench. This is largely aided by a co-simulation mechanism of the Verilog.AMS language and simulation, i.e. the event-driven simulation where analog and digital blocks are having separate timing mechanism, yet they do interfere through the variable passing [11]. In addition, a user may choose either absolute or relative timing for the test.

This test protocol description can be seen as a PDL part of P1687.2. For example, in the current version of P1687.2 the test stimuli in Fig. 5 applied on arbitrary differential ports can be described as:

```
iForceVoltage vin_pos,vin_neg ${DSPstimuliFileList}
-requirements {VoltageAmplitude=0.5; SampleFrequency=125M;
NumberOfSamples=4096; Delay=0; Frequency=5M; VoltageHighDC =
1.65;}
iApply;
```

Since the entire tool setup is targeted for test engineers working on Big A / Small D applications, it is still debatable whether the tool will generate P1687.2 deliverables from GUI or it will accept P1687.2 ICL/PDL as an input (or both). This will become clear once the standard is adopted and start gaining the ground in test community. In any case, the idea of the workflow of the tool is to ease test setup validation tasks of test engineers without burdening them with design /EDA specific details.

IV. EXAMPLES

A. Data Converter Test

The typical workflow of the tool will be described on an industrial 12-bit ADC application where the test specification has an analog ramp as input stimulus. A number of 'what if' scenarios will be analyzed based on different test instruments in order to define the test quality and limitations. Three ATE instruments from the fanTESTic database are selected for this test from two different ATE vendors, called instruments A, B and C. The setup for static ADC test of this application includes:

- 12-bit ADC as DUT with realistic analog frontend. For reference an almost ideal 12-bit AD model tested with the high precision instrument C model is used resolving to INL= 0.29, INL fit (corrected for trending) = 0.21, DNL= -0.25, gain = 1.00002, offset = 0.016.
- 16-bit stimulus from instrument A and B.
- 24-bit stimulus from high resolution instrument C.
- Logic analyzer for capturing digital response.

The ramp stimulus uses the following parameters:

- 4096 code levels, each code level is assessed using 4 “subcodes” (in order to improve the required resolution for testing a 12-bit device), which requires $4 \times 4096 = 16384$ equidistant time stamps.
- Sample rate stimulus = 300kps.
- On the tester, each “subcode”, is tested 32 times to reach an averaging-out of measurement noise. The tool uses models that incorporate noise artefacts due to cable, loadboard and tester instrument. For this reason, it is not required to simulate 32 times the same timestamp since this will give the same result

from a simulator. But, the ramp speed will be adapted to it in the test simulation.

- stimulus duration: $(16384 \times 32) / 300 \text{kps} = 1.748 \text{s}$
- sample rate stimulus/subcode = $1 / (1.75 / 16384) = 9373 \text{Hz}$.
- $V_{pp} = 2.0 \text{ V}$ for full dynamic range.

An issue in the stimulus generation occurs when the stimulus is made with instrument A with 16384 discrete steps as proposed in the setup, shown in Fig. 6. The test setup together with pins and defined actions is shown on the left side of the figure, whereas the result of a stimulus calculation with a zoomed inset is shown on the right.

When the stimulus is generated with the proposed number of samples (16384), the settling time of the source cannot be met and the model will return a settling time violation. The inset picture demonstrates the effect of noise and quantization on the power ramp. The quantization occurs from the actual 12 bit stimulus setting and is not preferred for testing the 12 bit ADC.

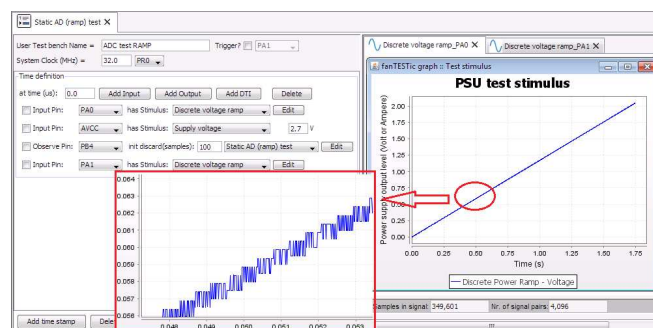


Figure 6. Overview of a complete test setup generation on ATE instrument A.

To improve on the accuracy of the ramp an alternate instrument source B is used. It has faster settling time and higher accuracy. Using all of the subcodes improves the resolution. Fig. 7 shows that quantization is, as expected, hardly observable anymore and that only noise dominates. Settling times are met with instrument B with the high resolution setting.

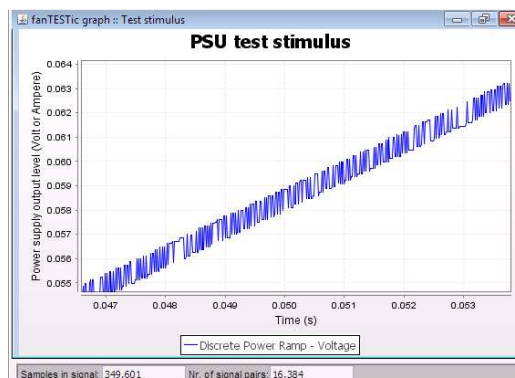
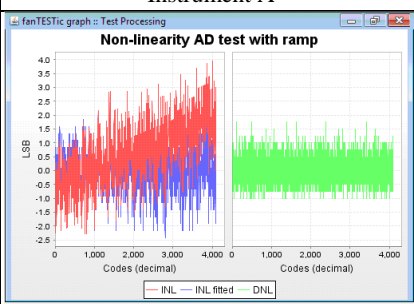
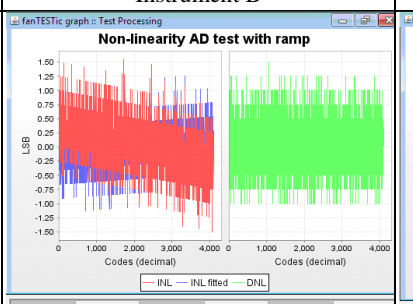
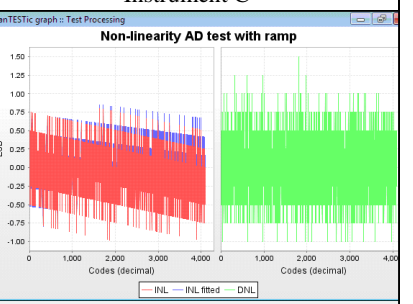


Figure 7. Zoomed detail of the upramp using 16384 subcodes on ATE instrument B.

Next to that, a more realistic scenario is applied where the DUT has a non-ideal analog frontend. The DUT model assumes noise and non-linearity effects in the analog frontend. Outcome of the processing model results are shown in the Table I and demonstrate the following:

TABLE I SUMMARY OF PROCESSED RESULTS ON 12 BIT ADC

Instrument A	Instrument B	Instrument C
		
INL(lsb) = 3.94867 INL fitted(lsb) = 2.58759 DNL(lsb) = 1.75168 Gain = 1.00061 Offset = -0.58776	INL(lsb) = 1.54035 INL fitted(lsb) = 1.45752 DNL(lsb) = 1.49969 Gain = 0.99983 Offset = 0.40973	INL(lsb) = -0.97765 INL fitted(lsb) = -0.91713 DNL(lsb) = 1.49969 Gain = 0.99995 Offset = 0.02444
INL=3.95, INL fit = 2.58, DNL=1.75 gain = 1.00061, offset = -0.59	INL=1.54, INL fit = 1.46, DNL=1.50 gain = 0.99983, offset = 0.41	INL= -0.98, INL fit = -0.92, DNL= 1.50 gain = 0.9999, offset = 0.024

Instrument A proves to be noisier than the analog frontend. The realistic DUT model does not significantly limit the INL and DNL performance parameters with regard to the ideal DUT model and the histogram has excessive variation. Instrument A is therefore the least suited instrument choice for this test setup.

Instrument B has comparable non-linearity behavior as compared to the DUT. INL/DNL performance is only slightly decreased with the more realistic DUT model. This instrument is therefore an acceptable choice for the test setup.

Instrument C will not significantly limit the measurement precision for this DUT setup. Here, the DUT model restricts the INL/DNL performance as can be compared to the reference numbers for the ideal DUT model with this AWG stimulus. This AWG is therefore the most preferred option as stimulus generator source for this test setup.

B. Embedded instrument test

A second implementation of the tool flow addresses the ability to quickly assess the capability of embedded test instruments (ETI) in AMS circuits. Fig. 8 shows a modular DC offset sensor [12]. The sensor is aimed to detect DC drifts over aging of the DUT. The embedded sensor model encompasses 4 building blocks:

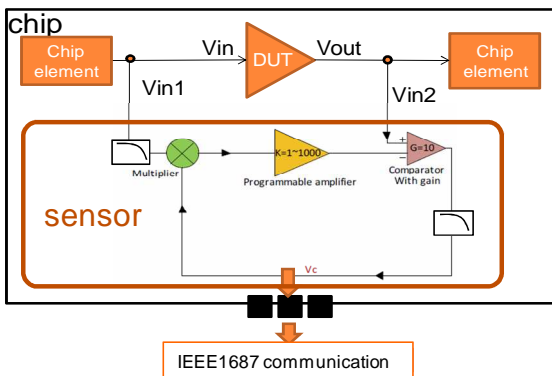


Figure 8. Diagram of DC offset sensor in DUT.

- The test data measurement or data capture interface.
- The sensor transfer component.
- The output processing or the user verification metric.
- The operational control, settings and check class.

The ETI is implemented as model in our prototype tool and will be compared to the transistor level simulation. Two verification steps of the model are investigated:

- **Ideal model simulation** meant for calibration purposes. Transistor level simulations of the sensor from factory out test (0 years) up to 20 years of usage are compared to the model. It verifies if the model implementation predicts the correct output response.
- **Statistical model simulation** (in Monte Carlo setup) applying different parameter model variations to the sensor. The goal is to identify the robustness of the sensor to noise and higher order non-linear effects, hence, the capability to measure the correct offset under a noisy environment. The Monte Carlo model simulation extends to multiple (20) aging years and is shown in Fig. 9. illustrating the *Voffset* detection histograms from the sensor model. The prediction indicates a statistically relevant DC offset is to be expected and that this sensor has sufficient accuracy to measure the DUT DC drift.

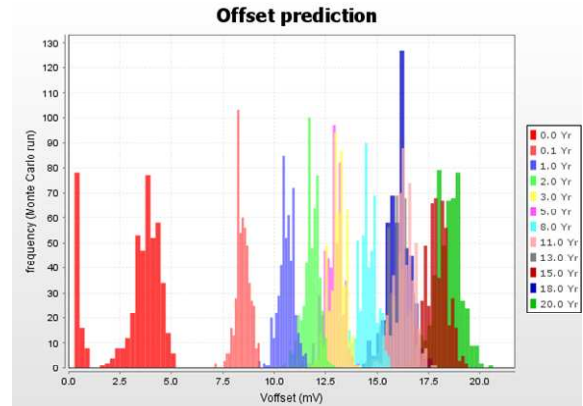


Figure 9. Statistical result of predicted DC offset from the model simulation.

Our model approach has the advantage that statistical information can be included in a very short time which will otherwise be an expensive computational factor in design. The statistical outcome of the new model can be compared to results from the transistor monitor circuit and the directly observed DUT responses from the SPICE simulator as shown in Fig. 10. In all cases self-induced offsets are taken into account. The legend in the figure indicates the following:

- DUT: *Voffset* measured from SPICE simulator view

- Transistor: V_{offset} measured, in SPICE simulator, by the actual transistor implementation of the sensor.
- Model: V_{offset} measured, with model simulator, using the statistical model of the sensor.

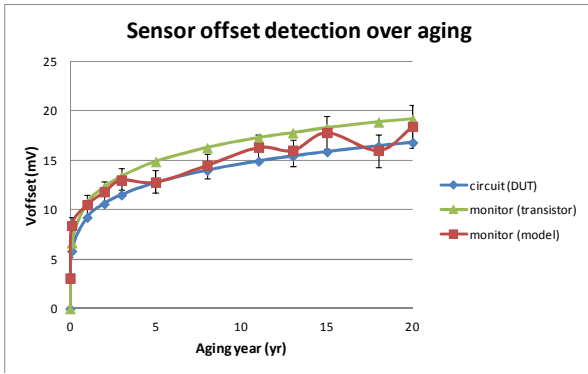


Figure 10. Comparison of model performance to transistor-level circuit and from EDA.

Both transistor and model monitor overestimate the real V_{offset} , though the model is closer to the DUT V_{offset} than the transistor monitor response. When considering statistical variation, the model captures more accurately the actual V_{offset} . This full statistical simulation for all aging years runs in 20 minutes and clearly identifies the accuracy and responsiveness of the ETI model.

V. QUANTIFICATION

Table II shows that simulation time for the ADC test setup is very limited (< 5 minutes) when models are used for DUT and tester instruments. Modeling time for stimulus and processing is negligible.

TABLE II: COMPUTATIONAL PERFORMANCE FOR TEST SETUP VERIFICATION USING MODELS AND SIMULATION

	Instrument A	Instrument B	Instrument C
Model stimulus	0.11 sec	1.46 sec	0.094 sec
Simulation	91.7 sec	358.0 sec	33.0 sec
Load+Processing	0.376 sec	0.204 sec	0.335 sec

Considering test development and debug as key drivers in test cost definition [13], we have explored the business case of the tool on an industrial design. The cost model is shown with the equation below in which the time for test development engineer, operator and test hardware are expressed in EURO cost equivalents. Although empirical, the formula has been often applied in this format within project management and can also be traced to the Non-Recurrent Engineering cost considerations in [13].

$$C_{NRE} = \sum_{i=1}^{\#Tests} (C_{dev} + w_i t_{debug} \times C_{ATE}) \quad (1)$$

C_{dev} is a cost of test development, which is typically split in (functional) core bins, with each bin consisting of multiple sub-tests. C_{ATE} is a cost of tester equipment per unit of time, whereas weighting coefficient w_i indicates how difficult it is to debug a particular function (with w_i being set to 0 indicates that no debug was necessary).

Applying this cost model to both the traditional approach and the new simulation tool approach on the design considered in section IVa leads to the key

performance indicators in Table III that clearly shows the benefits of the tool deployment.

TABLE III: QUANTIFICATION OF FANTESTIC TEST SYNTHESIS APPROACH

Savings test debug	55.7% in hours
Savings test development	11.0% in hours
Overall savings Pre-Si validation	39.5% time reduction
	44.5% cost reduction

VI. CONCLUSION

We presented a software package called fanTESTic that decreases test development time of AMS circuits through automation of test setup validation. It enables test engineering team to efficiently assess the influence of test equipment on their product through automatic testbench generation for simulation with post-processing capabilities. The tool is compatible with any commercial EDA simulator and it tackled and resolved majority of the problems currently addressed within the upcoming IEEE P1687.2 standard.

ACKNOWLEDGMENT

This work received substantial support through EU subsidized TOETS, ELESIS and HADES projects within CATRENE, ENIAC and PENTA programs, respectively.

REFERENCES

- [1] F. Poehl, et al, "Production test Challenges for Highly Integrated Mobile Phone SoCs: A Case Study," *Proc. of the European Test Symposium (ETS)*, 2010.
- [2] "Why Analog Design Fail," <https://semiengineering.com/making-analog-more-reliable/>
- [3] P1687.2 - IEEE Standards Association, <http://sites.ieee.org/sagroups-1687-2/>
- [4] T. Austin, H. Webster, "Creating a Mixed Signal Simulation capability for Concurrent IC Design and Test Program Development," *Proc. Of the IEEE International Test Conference* 1993, Washington DC, 1993, pp 125-132.
- [5] B. A. Webster, "An Integrated Analog Test Simulation Environment," *Proc. International Test Conference*, Washington DC, 1989, pp. 567-571.
- [6] K. Einwich, G. Krampl, R. Hoppenstock, P. Koutsandreas, S. Sattler, "A Multi-Level Modelling Approach rendering Virtual Test Engineering (VTE) Economically Viable for Highly Complex Telecom Circuits," *Proc. Of Design, Automation and Test in Europe (DATE)*, Munich, Germany, March 1999, User's Forum, pp. 227-231.
- [7] G. Krampl, M. Rona, H. Tauber, "Test Setup Simulation – A High-Performance VHDL-Based Virtual Test Solution Meeting Industrial Requirements," *Proc of International Test Conference* Washington DC, 2002, pp 870-879.
- [8] M. Fitchet, J. Revie, A. Patterson, "Mixed-Signal Simulation Technique applied to Virtual Test," *Proc. of 4th IEEE International Mixed-Signal Testing Workshop (IMSTW)*, The Hague, 1998.
- [9] T. Hogan, D. Heffernan, "Virtual test reduces semiconductor product development time," *Eng. Sci. Education Journal*, vol 10, issue 3, June 2001, pp 106-112.
- [10] M. Dufils, J-L. Carbonero, P. Planelle, "Mixed-Signal Simulation and Test Generation," *Proc. of Design and Test of Integrated Systems in Nanoscale Technology*, DTIS 2006..
- [11] Verilog.AMS Language Reference Manual, version 2.3.1, June 2009.
- [12] J. Wan, H.G. Kerkhoff, "An embedded offset and gain instrument for OpAmp IPs," *Proc. of Design, Automation and Test in Europe Conference (DATE)*, March 2014, pp 1-4.
- [13] *International Technology Roadmap for Semiconductors*, 2011 Edition, <http://www.itrs2.net/2011-itrs.html>.