Digital Analog Design: Enabling Mixed-Signal System Validation

Byong Chan Lim, James Mao, and Mark Horowitz
Stanford University

Ji-Eun Jang and Jaeha Kim
Seoul National University

Editor's notes:
Every analog circuit design goes through some kind of electrical validation step before release to manufacture. This paper demonstrates a validation method that is very similar to that used for digital systems. By comparing the initial functional model at the block level with a corresponding model of the transistor implementation, the suitability of the design can be judged.
—Gordon Roberts, McGill University

Over the past half century chip design has made enormous progress. We have moved from chips with a few gates to today's mega system-on-chip (SoC) designs. During this time, we have seen analog circuits first migrate on chip, in the form of simple op-amps, and then grow in complexity to become complete analog systems. Over time, the required performance and precision of these analog systems has continued to increase, while the allowable power dissipation has shrunk. Unlike digital systems where scaling has been a win for performance, power, and cost, for analog designs, technology scaling has helped in some ways but hurt in others. For example, reducing supply voltages reduces maximum signal energy, making high signal-to-noise ratio (SNR) systems harder, and scaled MOS transistors have worse matching property and noise. As a result, it is not possible to build a pure analog system that meets today's system specifications. To achieve the required level of performance, current analog systems use many digital correction loops to improve the performance of the analog system [1]. These can be as simple as an offset correction loop for a comparator, to a complex decision feedback equalization system to compensate for signal loss in a cable with digital correction of nonlinear amplifiers in an analog-to-digital (A/D) converter.

This tight coupling of complex analog systems to complex digital solutions creates great challenges for chip validation groups, since analog and digital designers view design and validation very differently. Two of the core issues are the differing goals of analog and digital validation, and the relative importance given to functional models. Analog designers in general do not trust models; they validate what really matters—the circuits that are actually implemented. They feel only fools would trust a model. Furthermore, since the output(s) are an analog (also known as smooth) function of the inputs, analog designers focus on trying to ensure that the design meets its parametric goals, in terms of gain, bandwidth, noise, power supply rejection ratio, etc. They use accurate transistor-level models and circuit simulators such as SPICE that solve large coupled nonlinear differential equations. While these simulations take significant time per run, this is not an insurmountable issue, since there are not many runs needed for validation. Since the output is a smooth function of the input, generating a large number of input test cases is not needed; all these tests will produce similar results.

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.
Digital Object Identifier 10.1109/MDAT.2014.2361718
Date of publication: 07 October 2014; date of current version: 16 January 2015.

Copublished by the IEEE CEDA, IEEE CASS, IEEE SSCS, and TTTC

The digital validation engineer works in a very different space. They need to ensure this complex digital system really works. Since the system is discrete (also known as not smooth), they must run millions/billions of input vectors into it, and check to ensure it functions properly in all these cases. Given the large number of input vectors that need to be simulated, and the large number of gates in the final implementation, digital designers left simulating transistors and gates many years ago. Instead they design and validate at a higher level, using SystemVerilog or VHDL to create functional models of the design, and use formal checking tools like Formality [2] to ensure that the implementation matches the model.

Coupling an analog design into a complex digital system is difficult. Transistor-level simulation of the analog part is generally out of the question, since it is too slow: even if you are only trying to validate the analog system and its correction loop, the digital correction loop can take tens of thousands of cycles to settle. To handle this issue, real number models of analog circuits in Verilog have been popularized for the verification of mixed-signal designs [3]–[5]. These models use discrete-time, real number values to represent the analog signals at the pins of analog blocks. This signal representation allows one to create good fidelity analog functional modeling in a digital simulation environment. While this approach is a great step forward in system validation, two principle issues remain. The first is to remove the danger of using a functional model of the analog block. Since it has not been validated against the transistor implementation, you can get into a situation where your model works, but your chip does not. The second issue is that writing analog models is still *ad hoc*; it is not understood what is the right granularity for an analog functional model, nor the procedure for writing high-speed, high-fidelity functional models.

To address these remaining issues, we have been working on formalizing the approach to analog design for mixed-signal systems [6], which we call "digital analog design." It tries to take some of the productivity aids used in digital design/validation and adapt them to analog design. The overall approach is described in the next section.

Digital analog design

Given how critical and time consuming top-level validation is for digital designers, we assume that they will drive the full system validation effort. This means that we need to create an analog design approach that will both satisfy the requirements of analog design and fit into the digital validation flow. To us, this means that the system will need to use high-speed functional models for validation of the analog blocks at the top level, since nothing else will be fast enough. The key question is how to do this in a way that provides the same guarantees as we get with using high-speed functional models of the digital blocks, and is enabling rather than limiting for analog designers.

Our methodology starts with partitioning a mixed-signal system into many smaller blocks by designers, and each will become an analog cell in the final design. Designers initially create simple functional models for these blocks to explore the design space. In our design approach, these models are written in SystemVerilog such that it can be used later for validation.[1] After determining roughly the requirements for each of the cells, designers then work at the transistor level to create designs that achieve these results for each of these analog cells. Since there is not a universal set of electrical rule checks (ERCs) for analog cells, designers also create a set of ERCs for each cell to check the environments where the cell is used. During this process, all the circuit-level issues are verified at block level in SPICE simulations, ensuring those issues will not break the functional model.

After the transistor level designs are complete, we essentially have two descriptions of each analog cell. To ensure that the validation engineer can use the functional model with confidence, we need a way to formally compare it to the circuit implementation. This requires us to create a formal definition of an analog function. This abstraction is critical, since it not only helps us formally compare the functional model to the circuit implementation, but it will also guide us in the creation of analog functional models. For digital circuits, for instance, formal checkers use both Boolean value and synchronous time abstractions for the comparison. The rest of this section describes the abstraction that we choose. The Analog Functional Model section then

[1]Today these analog models are often written in Matlab/Simulink [7] or CppSim [8], or other high-productivity languages, but these models cannot be reused later for validation.

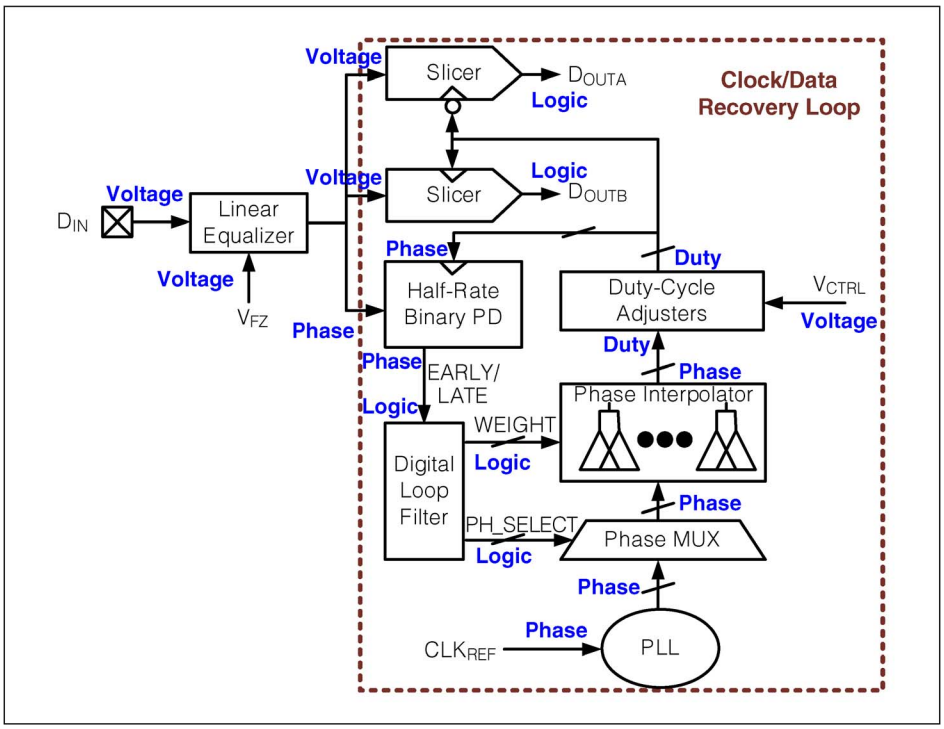

Figure 1. Block diagram of a serial link receiver [10].

describes how to create analog functional models in Verilog, and the Functional Model Validation section explains how circuits can be formally compared to these models.

While it may seem overly simplistic, we believe a piecewise linear model which captures its deviations from linear behavior (to capture weakly nonlinear behavior of the system) is the correct analog abstraction for the verification of analog systems. Remember, what makes a circuit analog is that there is a smooth relationship between inputs and outputs, and that in nearly all cases the designer wants this smooth relationship to be approximately linear. The importance of this linear (piecewise linear) behavior is brought home by the fact that the optimization objectives of analog circuits are either the properties of a linear system (e.g., gain and bandwidth) or the quantities that describe the deviation of the circuit from the linear model that we want to minimize (e.g., input offset of an amplifier, integral nonlinearity/differential nonlinearity of a data converter, and signal distortion of an RF mixer during the frequency conversion).

An obvious objection to this claim is that no system is completely linear and many systems, such as variable gain amplifiers or phase-locked loops (PLLs), do not seem linear at all. While many of these systems are "nonlinear" when viewed in the current, voltage, and time domains, these systems can be transformed into linear systems by variable domain transformation [9]. For example, Figure 1 shows a block diagram of a typical high-speed link receiver and notes the variable domain in which each block could be mapped to a linear system model.[2] A PLL is highly nonlinear from a voltage-time perspective, but can be modeled as a linear system in phase domain of the clock input and output. Another example is a duty-cycle adjuster where its detailed circuit implementation and transfer function are shown in Figure 2 [11]. The duty cycle of the output clock CLKo from the incoming clock CLKi is tuned by controlling the voltage input *Vctrl*. While clearly nonlinear in voltage-time space, the response surface in duty-cycle domain is nearly a hyperplane, as shown in Figure 2b, through the domain translation, duty-cycle-to-voltage for clock input and voltage-to-duty-cycle for clock output.

A variable gain amplifier is representative of another class of nonlinear circuits, where the "linear" output response is modified by the value of a control input; in this case, it is the amplifier's gain. However, these kinds of circuits are easily handled by a slight extension of our linear model, since the system remains linear from input to output when the parameter controlled by the "control" inputs (e.g., gain) is fixed or varies slowly enough that they do not interfere with the main signal path. In these cases, we characterize the linear system as a function of the control input. This characterization accurately models everything except the coupled dynamics

[2]The output of a slicer is a logic signal, which seems to be tricky to find its linear system model. This kind of an A/D conversion circuit needs a special transformation of the output to the input, resulting in a linear model which tracks the input threshold of the original circuit. The detailed description is found in [10].

between the input and the control. In fact, since dynamically coupled input/control interactions are hard to analyze and often lead to unwanted effects (intermodulation output terms), systems are always built where the control loop bandwidth is much slower than the signal path to avoid these effects. Thus, the relationship from the control input to the resulting parameter can also be generally modeled as a linear system as well. On reflection, the fact that a (piecewise) linear model is a good abstraction for analog circuits makes sense, since we tend to build systems that we know how to analyze, and these are systems whose behavior is linear in some domain (e.g., voltage for amplifiers, phase for PLLs, and frequency for mixers).

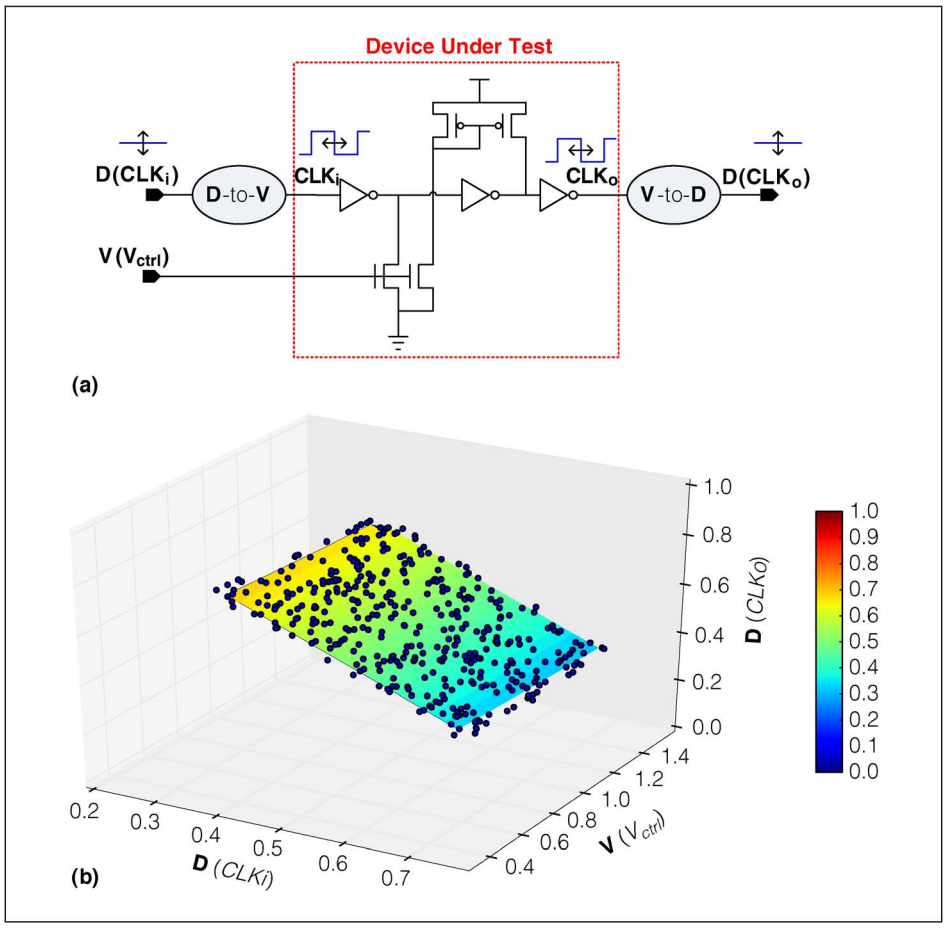

Figure 2. Duty-cycle adjuster: (a) circuit diagram with domain translators; and (b) response surface in duty-cycle domain [11].

Note that this linear analog abstraction is primarily used for model comparison and helps in understanding the circuit behavior for writing functional models. This does not mean that a Verilog model must be written as a linear system model in a possibly transformed variable domain. For example, it is more natural to write a phase detector model with a couple of flops and a few gates with the analog signals represented by the timing of the output pulses than to write it in phase domain. Similarly, while a class-D amplifier can be viewed as a linear system in duty-cycle domain, it can be straightforward to model it as a circuit whose switches are controlled by a periodic, pulse-width-modulated input signal. On the other hand, however, a phase domain model followed by a slicer is preferable in writing a voltage controlled oscillator model, since the circuit's behaviors (e.g., phase noise) are well understood in phase domain. The abstraction's function is that it still defines what needs to be measured to ensure that the two descriptions match.

Analog functional model

Our functional models are event driven to enable efficient simulation in an event-driven logic simulator like SystemVerilog. It implies that each time the input changes, the model evaluates and updates the output waveform. The linear abstraction of analog circuits provides some clues on how to create good analog functional models that fit nicely into a digital validation context. We write a functional model in a way that the new output is the result of a piecewise, nearly linear filter being applied to the input. We initially represented the input as a piecewise-linear waveform, but given the basic linear filtering operation, we have more recently explored representing the input as a sum of complex exponential functions and built a functional simulator, i.e., an extension to a SystemVerilog simulator, called XMODEL. The rest of this section focuses on the basic idea of describing analog functions when

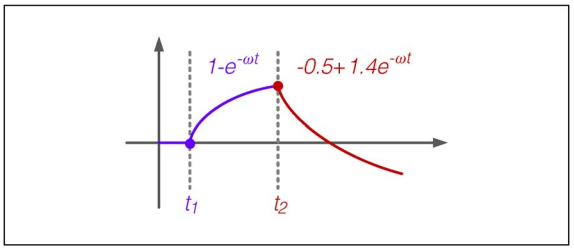

Figure 3. Continuous-time analog signal expressed with a series of events, each of which describes the waveform as a sum of complex exponential functions.

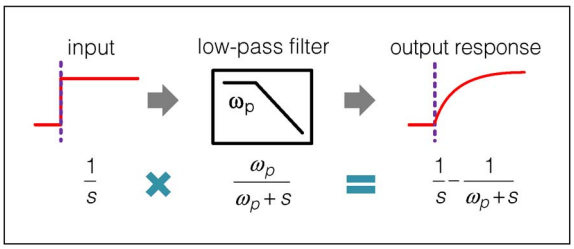

Figure 4. Event-driven computation of linear system response using Laplace *s*-domain computation.

inputs/outputs are represented by Laplace *s*-domain notation as is done in XMODEL.

Given that each analog block is modeled as a filter with a finite number of poles and zeros, its output waveforms, and hence the continuous-time, analog signal $x(t)$ can be expressed in the following functional form:

$$x(t) = \sum_i c_i t^{m_i} e^{-a_i t}, \qquad \text{for } t \geq t_{\text{event}} \qquad (1)$$

which is basically a sum of complex exponentials. In general, a signal is expressed using a series of events, each of which updates the set of coefficients $\{c_i, m_i, a_i\}$, as illustrated in Figure 3. The coefficients can be bundled into a single variable using *struct* in SystemVerilog [12].

The form in (1) is more expressive than the piecewise linear representation. For instance, an exponentially decaying response from a first-order low-pass filter requires just a single event with two exponential terms (one of them is a constant). An RF signal whose amplitude, frequency, or phase is modulated by a lower rate digital signal can be expressed using only as many events as the data symbols. Also, it can be shown that all the waveforms that can be generated using various source elements in SPICE can be captured in this form.

The main advantage of using this functional form is that it enables a purely event-driven simulation of analog circuits [12]. That is, the set of coefficients describing the output of a linear system is updated only when the set of coefficients describing the input has changed. This property allows fast simulation that does not depend on the circuit size but only on the number of events generated. Figure 4 illustrates how the response of a linear system is computed in an event-driven way. First, note that the form in (1) can be directly transformed into a Laplace *s*-domain expression

$$x(t) = \sum_i c_i t^{m_i} e^{-a_i t} \xrightarrow{\mathcal{L}} X(s) = \sum_i \frac{b_i}{(s + a_i)^{m_i+1}}. \qquad (2)$$

Once the input signal is transformed into *s*-domain, the output signal can be computed simply by multiplying it with the *s*-domain transfer function of the system, which can yield the same form as (2) after partial-fraction decomposition. Of course, the cost of this approach is the more complex signal representation.

Figure 5 illustrates an example model of a high-speed link using the sum-of-exponential representation and *s*-domain event-driven simulation approach. The link consists of a preemphasizing transmitter, a channel, a continuous-time linear equalizer (CTLE), and a decision-feedback equalizing (DFE) receiver. First, the transmitter generates a piecewise linear signal modeling the finite rise/fall times. Then, the signal propagates through the channel and the CTLE stage, each of which can be well modeled as linear filters. To model the possible distortion effects in the front–end circuits (e.g., gain compression), a nonlinear element with a piecewise-linear transfer function can be inserted. It is noteworthy that the transmitter generates up to two events per unit interval (UI), and this number remains the same as the signal passes through the channel and equalizer since the computation is purely event driven. Therefore, the simulation speed and accuracy depend very weakly on the time step resolution of the Verilog simulator [12]. Figure 5 also shows the final eye diagram seen by the final data receiver after the DFE.

Functional model validation

Our abstraction of analog circuits provides a formal way to compare two analog representations

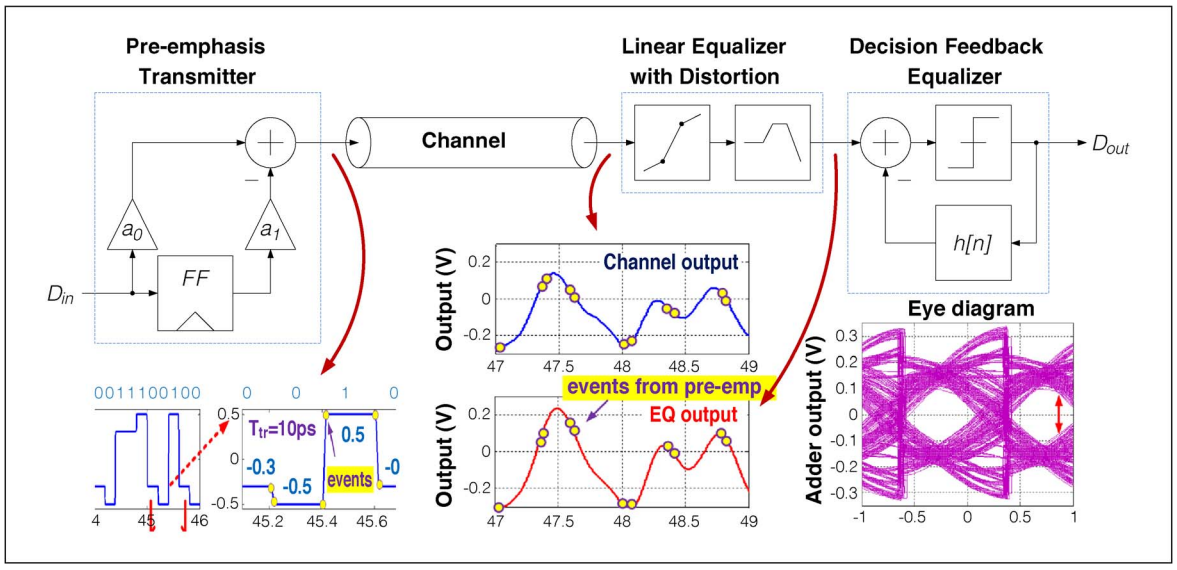

Figure 5. Example of modeling the signal path in a high-speed link.

(e.g., a transistor schematic and an analog functional model): Extract the function from each representation, and check to see if the functions match. If they do, the representations match; otherwise there is a mismatch. Since linear systems are characterized by a transfer function (and possible distortion terms if deviation from linearity is critical in this application), the transfer matrix from the system's inputs to its outputs is extracted from both representations. For piecewise linear models, or units with controllable parameters, this matrix is extracted for the different operating points and corresponding matrixes are compared. If the extracted matrices do not match, then we can use the differences in the matrices to fix this mismatch by either changing the model or the implementation.

Surprisingly, automatically generating the test stimuli needed to extract this characterization of the analog blocks is not difficult. In digital systems, the response surface is completely discontinuous, which means one needs to explore all possible input values to explore its function. Therefore, the hardest part of such validation is how to generate test vectors to cover the complete result space quickly. For analog systems, the response surface is smooth and can be explored with a few samples. With the linearity assumption, the surface is almost hyperplane, so theoretically we can characterize the output response of a circuit using only $N+1$ different input stimulus where N is the number of analog inputs. Although there are digital inputs as well in digitally assisted analog circuits, many of those inputs have analog intent (e.g., quantized inputs), such that the number of test vectors still grows linearly with the number of inputs. True digital inputs, like power-down or calibrate, change the underlying circuit, so the transfer function of each circuit configured by true digital inputs must be compared between the two implementations [11].

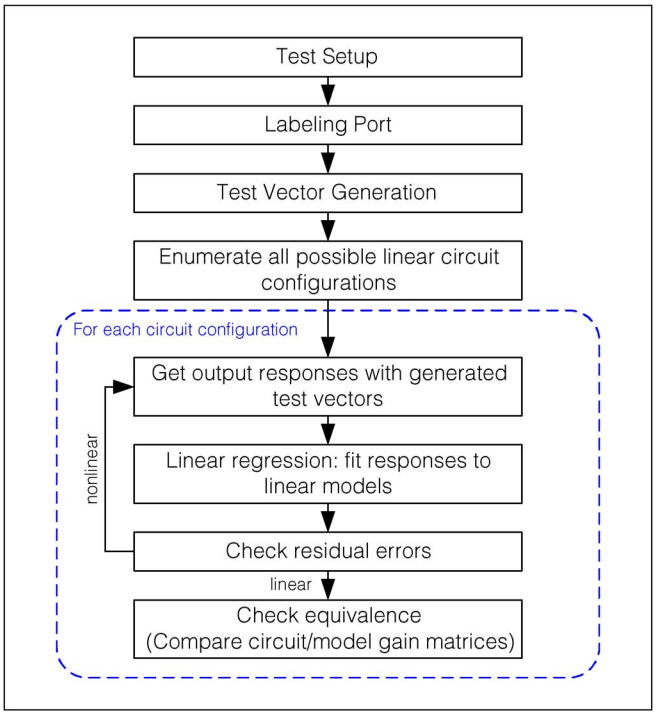

Figure 6. Procedure for model equivalence checking.

Table 1 Equivalence checking results of the duty-cycle adjuster shown in Figure 2. Transfer matrices of various models: $\mathbf{y} = \mathbf{Gx}$, $\boldsymbol{y} = [\mathbf{D}(\mathbf{CLKo})]$, and $\mathbf{x} = [\mathbf{D}\ (\mathbf{CLKi})\mathbf{V}\ (\mathbf{Vctrl})]\mathbf{T}$.

Model	Transfer matrix **G**	Relative error of **G** to Model 1 in %
Model 1 (Circuit netlist)	[–1.0 0.026]	-
Model 2 (Correct)	[–1.0 –0.028]	[0 8]
Model 3 (CLKo inverted)	[1.0 0.028]	[–200 –207]
Model 4 (Vctrl inverted)	[–1.0 0.028]	[0 –207]

In practice, we measure more than the minimal number of input vectors for each circuit for two reasons: to ensure that the extracted models are good estimates of the desired parameters in the presence of noise in the measured analog quantities and to generate more accurate models (e.g., weakly nonlinear models).

Figure 6 describes an overview of model checking procedure given that the port intents are labeled and the test setup for running simulations is prepared. Test vector generation starts with enumerating all possible linear circuit configurations created by true digital inputs. The test vectors of N analog/quantized inputs are generated by design of experiment and used for all created circuit configurations. For robustness to simulation noise and model fitting errors, the number of test vectors is more than the minimum. The generated vectors are exercised on the circuit netlist and SystemVerilog model, and the transfer matrices of both models are then extracted and compared by performing linear regression on the response samples of analog outputs to analog/quantized inputs.

The challenge in analog model validation is that no pair of corresponding values in transfer matrices for two extracted systems has the same values. Even if the same system is validated, the extracted values will be different with different test vectors because of the inherent nonlinearity of circuits and numerical errors in simulation. This issue is handled by creating an allowable tolerance in the extracted parameters. The value of this tolerance is easier to set than one might expect. For example, Table 1 shows that the error is over 100% whenever any pin connection errors exist between the model and the implementation. Thus, small mismatches simply indicate that the functional model does not represent the current circuit performance. Since these models are generally parameterized, the common approach is to extract the current values from the implementation and use those in the functional models. In fact, it is even possible to provide distributions of these parameters for the analog blocks, and perform Monte-Carlo-like simulations at the functional level [11].

This validation method of a functional model is implemented as a tool, and various analog cells are tested with the tool [10], [11]. For example, Figure 7 shows a block diagram of a single-slope ADC which adopts our "digital analog design" flow. The functional models of all the blocks in a PLL (e.g., a phase frequency detector, a charge-pump circuit with a loop filter, a ring oscillator, and a frequency divider) and other subblocks in the figure are checked against their circuit implementations to detect pin connection errors, and their parameters are updated from the checking results. Of course, digital blocks (i.e., a counter and latches) are validated using a digital validation tool. The detailed results are found in [11], and other examples (i.e., subblocks of a serial link receiver) are found in [10].

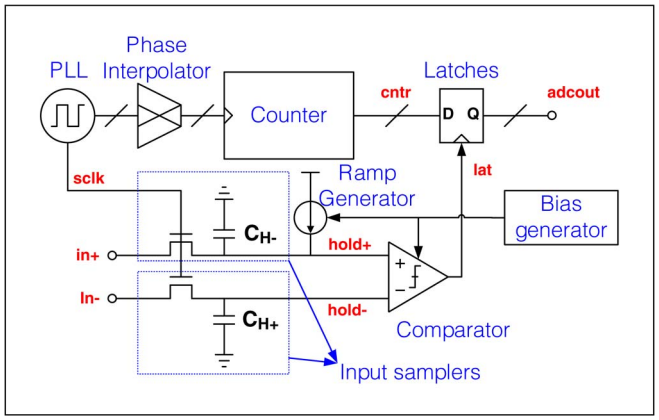

Figure 7. Block diagram of a single-slope ADC as a functional model validation example.

VALIDATION OF COMPLEX mixed-signal chips is forcing a change in analog design methodology. To be able to perform full system validation, high-performance, functional models of the analog blocks are required, and these models must be validated against the transistor level implementations of these blocks, if the system level validation is going to be meaningful. Digital analog design, which creates an event-driven, SystemVerilog functional

model of analog circuits, and then validates these models against the transistor level implementation of this circuit, is an approach to address this problem. It uses a piecewise (mostly) linear abstraction of analog circuits to both guide the representation of analog signals in an event-driven functional model, and to allow one to formally compare two analog descriptions. While this linear model initially seems very limiting, by transforming the circuit into the right domain, and leveraging the ability to handle piecewise linear, controlled systems, and distortion, we have not found a system that cannot be accurately modeled this way. In hindsight, the use of a linear abstraction makes sense, since it is the *lingua franca* of analog design. We are currently working on creating functional model templates for a number of analog blocks to make the method even easier to apply. ■

■ References

[1] B. Murmann, "Digitally assisted analog circuits," *IEEE Micro*, vol. 26, no. 2, pp. 38–47, Mar./Apr. 2006.

[2] *Formality User Guide*, Synopsys, 2013.

[3] J. B. David, "Radio receiver mixer model for event-driven simulators to support functional verification of RF-SOC wireless links," in *Proc. IEEE Int. Behav. Model. Simul. Conf.*, 2010, pp. 42–47.

[4] J. E. Chen, "A modeling methodology for verifying functionality of a wireless chip," in *Proc. IEEE Behav. Model. Simul. Workshop*, 2009, pp. 96–101.

[5] Y. Wang, C. Van-Meersbergen, H. W. Groh, and S. Heinen, "Event driven analog modeling for the verification of PLL frequency synthesizers," in *Proc. IEEE Behav. Model. Simul. Workshop,* 2009, pp. 25–30.

[6] M. Horowitz *et al.* "Fortifying analog models with equivalence checking and coverage analysis," in *Proc. 47th ACM/IEEE Design Autom. Conf.*, 2010, pp. 425–430.

[7] M. van Ierssel, H. Yamaguchi, A. Sheikholeslami, H. Tamura, and W. W. Walker, "Event-driven modeling of CDR jitter induced by power-supply noise, finite decision-circuit bandwidth, and channel ISI," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 55, no. 5, pp. 1306–1315, Jun. 2008.

[8] M. H. Perrott, "Fast and accurate behavioral simulation of fractional-N frequency synthesizers and other PLL/DLL circuits," in *Proc. 39th Design Autom. Conf.*, 2002, pp. 498–503.

[9] J. Kim, K. D. Jones, and M. A. Horowitz, "Variable domain transformation for linear PAC analysis of mixed-signal systems," in *Proc. IEEE/ACM Int. Conf. Comput.-Aided Design*, 2007, pp. 887–894.

[10] B. C. Lim, J. Kim, and M. A. Horowitz, "An efficient test vector generation for checking analog/mixed-signal functional models," in *Proc. 47th ACM/IEEE Design Autom. Conf.*, 2010, pp. 767–772.

[11] B. C. Lim, "Model validation of mixed-signal systems," Ph.D. dissertation, Dept. Electr. Eng., Stanford Univ., Stanford, CA, USA, 2012.

[12] J.-E. Jang, M.-J. Park, D. Lee, and J. Kim, "True event-driven simulation of analog/mixed-signal behaviors in SystemVerilog: A decision-feedback equalizing (DFE) receiver example," in *Proc. IEEE Custom Integr. Circuits Conf.*, 2012, DOI: 10.1109/CICC.2012.6330558.

Byong Chan Lim is a Postdoctoral Scholar at Stanford University, Stanford, CA, USA. His research interests include mixed-signal circuit design and methodologies for improving the productivity of mixed-signal SoC design and validation. Lim has a PhD in electrical engineering from Stanford University (2012).

James Mao is a freelance Software Engineer. His interests include design automation and code reuse. Mao has a PhD in electrical engineering from Stanford University, Stanford, CA, USA.

Mark Horowitz is the Yahoo! Founders Professor at Stanford University, Stanford, CA, USA, and was chair of the Electrical Engineering Department from 2008 to 2012. He cofounded Rambus, Inc., Sunnyvale, CA, USA, in 1990. His research interests are quite broad and span using electrical engineering and computer science analysis methods to problems in molecular biology to creating new design methodologies for analog and digital VLSI circuits. He is a Fellow of the IEEE and the Association for Computing Machinery (ACM) and a member of the National Academy of Engineering and the American Academy of Arts and Science.

Ji-Eun Jang is currently working toward a PhD in electrical engineering at Seoul National University, Seoul, Korea. Her research interests include mixed-signal circuit design and verification methodologies.

Jaeha Kim is currently an Assistant Professor at Seoul National University (SNU), Seoul, Korea. His research interests include low-power mixed-signal circuit design and their verification methodologies. Prior to joining SNU, he was with Stanford University, Stanford, CA, USA, as Acting Assistant Professor; with Rambus, Inc., Sunnyvale, CA, USA, as Principal Engineer; and with Inter-university Semiconductor Research Center (ISRC) at SNU as Postdoctoral Researcher. He is a recipient of the Takuo Sugano award for outstanding far-east paper at the 2005 IEEE International Solid-State Circuits Conference (ISSCC) and a Distinguished ACM Speaker in the area of design automation.

■ Direct questions and comments about this article to Byong Chan Lim, Department of Electrical Engineering, Stanford University, Stanford, CA 94305-9505 USA; bclim@stanford.edu.