Simulation of Real World Circuits: Extending Conventional Analysis Methods to Circuits Described by Heterogeneous Languages

Federico Bizzarri, Angelo Brambilla,
Giancarlo Storti Gajani, and Soumitro Banerjee

I. Introduction

Circuit simulation is at the heart of much of electrical engineering today. Both in academia as well as in industry, whenever one has to design a circuit to serve some specific function, the use of a circuit simulator is indispensable. However, the sense that the term

Federico Bizzarri, Angelo Brambilla, and Giancarlo Storti Gajani are with Dipartimento di Elettronica, Informazione e Bioingegneria, Politecnico di Milano, p.za. Leonardo da Vinci, n. 32, I20133 Milano, Italy. Federico Bizzarri is also with the Advanced Research Center on Electronic Systems for Information and Communication Technologies E. De Castro (ARCES), University of Bologna, Italy. Soumitro Banerjee is with the Indian Institute of Science Education & Research, Kolkata, India, and the King Abdulaziz University, Jeddah, Saudi Arabia.

"circuit" carries has changed and is still changing with the progress of technology. Thirty years back, a circuit would have meant a collection of passive and active components connected in series or in parallel. Then switching circuits came into use which can toggle between two or more different configurations depending on the state of the switches.

For some time, there were analog circuits and digital circuits, treated separately. But the trend was towards very large scale integration as seen in extremely complex System On Chips (SOCs) or CPUs comprising billions of MOS transistors [1]–[4]. Presently the trend is towards putting the digital and analog domains together to form "mixed signal" circuits. Naturally, there is a need to develop circuit simulators that can take into account the complexity that such systems offer.

Typical examples of very complex circuits are SOCs. They are composed of a number of heterogeneous parts that, in the "traditional" design paradigm, would be built and assembled using different production technologies and on different physical "objects". SOCs are examples of the "ultimate" evolution of Moore's law, where the largest number of different components, hierarchically organized as heterogeneous subsystems are "crammed" on the same circuit [2]. Interconnection of all these heterogeneous parts requires special techniques [5]–[7], and efficient design and testing methodologies must be available [8]–[11]. SOCs are very complex, and very expensive to design, test and produce [12], [13].

As the intricacy of the design increases, simulation tools become evermore important: high complexity corresponds to high design and implementation costs and a "faulty" design can have severe economic consequences; simulators are the only tools that can help designers in the assessment of the functional aspects of their circuits [14]–[16].

Simulation has two different important roles in any electronic design flow; it is a direct front-end helper tool, used to verify design assumptions and new "unexplored" ideas; at the same time it is a powerful and indispensable back-end tool used to verify post design, or even post layout, system operation.

Complex systems are usually designed as a hierarchy of interconnected less complex subsystems and each subsystem is often designed using different tools corresponding to different perspectives and, ultimately, to different levels of abstraction and description languages. These are used to describe the same entity in different contexts or different entities in the same context. Each of these languages (SPICE-LIKE, VERILOG-A, DSPF, VERILOG, VHDL, SYSTEM-C) has its standard set of simulation tools that allows a wide range of analysis methods, with each analysis providing the answer to a specific design problem. Interconnections of different subsystems can still be simulated, but only few analyses are "transversally" available since the representation of each subsystem is based on different theoretical paradigms and many situations are still not completely or efficiently covered by current simulation tools.

In general, all the aforementioned languages have a one to one correspondence with different modelling philosophies, yielding sufficiently complex models to describe all features that need to be checked, but, at the same time, sufficiently simple to be simulated in a reasonable design cycle time. These languages basically rely on the fundamental *lumped model* description.

A. Geometry vs. Topology in Simulation: The Lumped Model Description

Traditional circuit theory textbooks define as *circuit* a region of space where electromagnetic phenomena can be conveniently described using a *lumped model* approximation. Lumped models are based on the idea of considering these regions sufficiently small, with respect to characteristic wavelengths, that it is possible to approximate propagation delays as "very small" and thus negligible if compared to the other timing characteristics of the model.

The circuit is then partitioned in terms of the components and their interconnections, and single components are characterized once and for all at their ports and for each silicon technology both by direct measurement in specific test circuits and by simulation using field solvers (i.e., Maxwell equations). The lumped circuit model paves the way to circuit theory and to "spice-like" circuit simulation, thus solving the problem of understanding complex electromagnetic physical systems that would be too complex or inconvenient to solve using Maxwell equations [14], [17]–[22].

Even if the lumped model allows a much simpler description of a circuit, large ones are still characterized by an enormous number of variables and, correspondingly, of Ordinary Differential Equations (ODEs) or, more in general, Differential Algebraic Equations (DAEs) [23]. This is even more relevant if parasitic elements are considered. These are parts of the circuit that are not introduced by the designer but that end up being present because of the structure and geometry of the circuit itself once it is implemented as a real physical entity. The

effect of these parasitic components can in some cases be crucial so that their inclusion in the circuit description becomes mandatory.

B. Simulation of Different Functions and Design Strategies

Lumped models greatly simplify our problem but still leave a large number of equations to solve. Models even simpler than "analog" lumped circuits are thus needed. This is possible by adopting a more high-level functional based approach. For example, some parts can be modelled in a "digital" way, with only two states represented as 0 and 1 (sometimes adding the pseudo states "X", unknown, and "Z" high impedance, if needed) where each state is usually mapped to different voltage values and the dynamics ruling their changes is not of interest. With this restriction, simulation can be greatly simplified: each change in the value of a digital signal can be seen as an *event* that occurs in zero time in the simplest case, or in a finite amount of time if a more detailed model is needed [24], [25]. The digital subsystems are usually the most complex in terms of number of components and topology; however, thanks to this high level modeling action, the whole circuit is partitioned in distinct areas governed by different modelling paradigms: digital variables do not admit time derivatives and their variations can be seen as "events" that are caused and have effects on the analog part of the circuit that is modeled by DAEs [24], [26], [27].

Further simplifications are possible: if a detailed knowledge of the inner workings of some parts of the circuit is not needed or is not available, it is common to describe these subunits in a functional way using a *behavioral* description. This is typical, for example, when there are parts that have not yet been designed, but just defined in a general way, so that only their final behavior is known. Simulation of these parts requires yet another simulator, that can also be event driven. Analog and digital subsystems can be both described using a behavioral language, what is important for the designer is that all these can be simulated along with all the other subsystems in a so called "mixed mode" or Analog Mixed Signal (AMS) simulation.

C. Whole System Simulation

A complete heterogeneous circuit can thus be modelled and described using three different methods: analog, digital and behavioral. What every designer wants is the possibility to intermingle these different descriptions in any possible way, and to simulate the resulting model with the desired level of precision and in a reasonable amount of time. Efficient AMS simulation [28] is one of the main goals of companies and researchers working at the design of Electronic Design Automation (EDA) tools [29].

The simulation tools that a designer would like to have ready for use in a AMS project, should be capable of performing a basic set of analysis:

- search for the operating (i.e. equilibrium) points of the circuit;
- computation of the transient time domain behavior of the system starting from any initial condition.

These two basic analyses, respectively known as DC (or in some cases OP) and TRAN are available in any AMS capable simulator and are simple extensions of their counterparts in the fully analog domain. In the TRAN analysis switching can be simply taken care of by appropriately defining the termination conditions in the ODE solver, and by making the initial condition of each phase of the evolution equal to the final condition of the preceding phase;

These first analyses are not sufficient, there are other simulator capabilities in the wish list of many designers:

- search for the steady state behavior in the time domain, i.e., locate the periodic orbits and their stability. This is done in the analog domain by using what is known as the "shooting method". For AMS systems, special techniques have to be adopted, which are outlined in Sections IV-A, IV-B and IV-C;
- computing the frequency response of the system and the relevant transfer functions. This can be done by the periodic small signal analysis, that can be extended to AMS as outlined in Section IV-D;
- computation of periodic noise and phase noise, extension of these methods to AMS circuits is outlined in Section IV-E and IV-F respectively;
- in case the orbit of the system turns out to be aperiodic, one would like to calculate the Lyapunov exponents which quantify the rate of divergence of arbitrarily close trajectories. This is briefly discussed in Section IV-G.

This second set comprises analysis methods that are used in the fully analog domain, are needed in many different applications, from RF to DC/DC power supplies, and are still lacking [30] in the AMS domain.

Among these, steady state methods are, possibly, the most important (see e.g., [17], [31] for a brief but nonetheless technical overview of several different algorithms),[1] there is extensive literature on these methods, and many extensions and generalizations have been proposed in the last 20 years. See e.g., [23], [32]–[35], for the time domain methods, such as Shooting (SH) and derived algorithms, or, in the frequency domain, harmonic balance [18], [36]. Noise analysis and jitter computation

[1]Several companies selling circuit simulators such as MENTOR GRAPHICS, CADENCE, SYNOPSYS, ORORA DESIGN, SOLIDO DESIGN, BERKELEY DESIGN have tackled the problem of fast and efficient AMS simulation of large circuits and published white papers on this topic. However, to the authors knowledge, they did not consider steady state algorithms.

Complex systems are usually designed as a hierarchy of interconnected less complex subsystems and each subsystem is often designed using different tools corresponding to different perspectives and, ultimately, to different levels of abstraction and description languages.

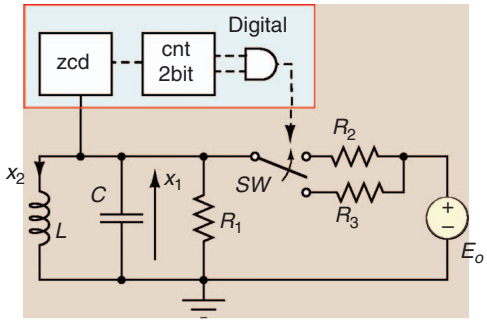
methods are also very important and many different algorithms have been proposed [37].

All these steady state and noise/jitter analysis methods were born and specifically designed for circuits described by strictly analog models. Extension to systems described also by digital parts has been done only in part and often resorting to "workarounds" such as macromodels. In other words: *no push-button steady state and noise/jitter analysis methods seem to be currently available for circuits described by analog, digital and behavioral parts.*

The main reason for this is that most algorithms used to perform steady state and noise simulations require continuity of the system equations and this is intrinsically in contradiction with the AMS model.

In the last decade several researchers have investigated this problem in the field of circuit analysis, among the most relevant works are those by Hiskens (see e.g. [38], [39]) and, more recently, by Banerjee (see e.g. [40]–[42].) These authors have focused most of their attention to power conversion circuits such as buck and boost DC-DC converters, and provide solid theoretical foundations for steady state and noise analysis of this class of circuits.

What was still lacking was a more general approach capable of unifying in a direct and straightforward way



The Netlist of the Oscillator

```
parameters VDD=1
verilog_include comb.v
r1    x   gnd   resistor   r=10
c1    x   gnd   capacitor  c=1
l1    x   gnd   inductor   l=1
sw    x   vdd   y   gnd    SW
+               noisetype=0
vr  vdd   gnd   vsource    vdc=VDD
A2d1  x   gnd   a2d  dignet="cnt.X"
+               vt=0
D2a1  y   gnd   d2a  dignet="cnt.Y"
+               vl=0 vh=VDD
model SW vswitch ron=5 roff=100M
+        von=0.6*VDD voff=0.4*VDD
```

The VERILOG
Description of the Digital Part of the Circuit

```
module cnt;
reg X, Y;
reg [1:0] Count;
initial begin
    Count = 0;
    Y = 1;
end
always @(posedge X) begin
    Count[0] = ! Count[0];
    Count[1] = (!Count[0]) ^ Count[1];
end
always @(Count) begin
    if( Count[0] && Count[1] )
        Y = 1;
    else
        Y = 0;
end
endmodule
```

Figure 1. The schematic of a circuit with a digital Finite State Machine (FSM) part. A zero crossing detector (zcd) is used as input to a 2 bit digital counter (cnt), that closes switch S when both its bits are high. $C = 1$F, $L = 1$H, $R_1 = 10\Omega$, $R_2 = 100\,\text{M}\Omega$, $R_3 = 5\Omega$, $E_o = 1$V.

What was still lacking was a more general approach capable of unifying in a direct and straightforward way *any* kind of lumped model circuit, including power conversion ones, into a single homogeneous class.

any kind of lumped model circuit, including power conversion ones, into a single homogeneous class.

A framework for the extension of steady state and noise algorithms to generic non-continuous circuits and systems was proposed in the last three years by the authors [43]–[53] and has been successfully implemented in the PAN academic circuit simulator. The circuit simulator PAN thus represents the first circuit simulator that includes all the classical "spice-like" features augmented with powerful extensions to mixed-mode circuit simulation.

II. A Motivating Example

The system shown in Fig. 1 is a very simple circuit which can be modeled using a few simple equations. Since it has a rather evident distinction between "what is analog" and "what is digital", it is well suited to introduce AMS simulation. Despite its simplicity the circuit is nonlinear and switching:

- the energy in a linear LC tank slowly decays through resistor R_1;
- a zero crossing detector (zcd) is used to reveal sign changes (from negative to positive) of the state variable x_1;
- a 2 bit counter (cnt) is triggered by zcd;
- the output of the 2-bit counter feeds a logical AND gate;
- when the output of the AND gate is high, switch S is connected to R_3, whose resistance is very low;
- when the output of the AND gate is low, S is connected to R_2, whose resistance is high.

A switching connection is present, so that when E_0 is connected to R_3, the energy lost by the LC tank is restored; this happens, thanks to the counter and the AND gate, once every four consecutive zero crossings of the x_1 voltage.

This circuit is an (AMS) oscillator, and its steady state behavior will be shown in Sec. IV. Moreover, using the modelling framework that is introduced in the next Section, this AMS circuit will be simulated as if it were fully analog.

The zero crossing detector, the 2 bit counter and the AND gate constitute the digital part of the circuit that can be described using the VERILOG language. This part interacts with the analog one, described, in the PAN simulation environment, using a "spice-like" netlist. Even though the circuit is switching, the designer does not have to be even aware of this, and is only required to model his/her circuit through the usual well known formal languages. Even if the extended analysis heavily relies on the non-trivial formalism needed to describe switching dynamical systems, the designer can be, in principle, completely unconcerned with it.

III. Modelling AMS Circuits as Hybrid Systems

A circuit appearing as an interconnection of analog and digital/behavioral parts can be schematically represented as in Fig. 2. It is assumed that the state variables corresponding to the analog part are continuous in time, while variables of the digital/behavioral part are quantized and are discontinuous in time. In most practical cases the time quanta are given by the time period between edges of a clock signal.

Each part can be described by appropriate dynamical equations. The natural way to represent the analog part is using a set of Differential Algebraic Equations (DAEs), which derives directly from the Modified Nodal Analysis (MNA) method employed in most circuit simulators [54]. These are obtained in the form

$$\begin{cases} \dot{x} = f(x, y, t, w_a, v_a) \\ g(x, y, t, w_a, v_a) = 0 \end{cases}, \tag{1}$$

where x are the state variables (usually the voltages across capacitors and the currents through inductors), y are "non-dynamical" variables, t is time and w_a, v_a are variables coming from the digital/behavioral part. The algebraic

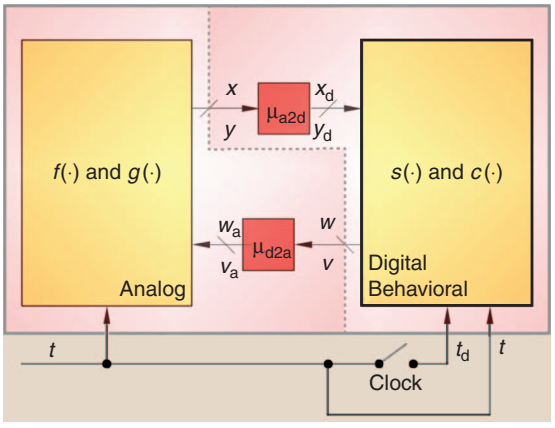


Figure 2. The generic architecture of an AMS circuit. The time variable t is assumed, at least in principle, to be available at both sides of the circuit. In particular, in the digital part, both a clock signal t_d and the continuous variable t are available. Clock-time can trigger sequential events and the continuous variable can be used to store the value of t at specific events if one is interested, for instance, in producing delayed digital outputs.

function g represents a set of constraints that must be fulfilled along the dynamical evolution of the system.

A natural way to represent the digital/behavioral part is derived from event driven simulation environments. It is, in a sense, a mirror image of the DAE describing the analog part: dynamical variables correspond to those that, in the digital part, describe the *sequential* part of the system, while algebraic ones correspond to variables in the *combinatorial* parts of the circuit. This is equivalent to writing a discrete map $s(\cdot)$ whose variables are constrained by expression $c(\cdot)$:

$$\begin{cases} w^{(n+1)} = s\left(w^{(n)}, v^{(n)}, t^{(n)}, x_d^{(n)}, y_d^{(n)}\right) \\ c\left(w^{(n+1)}, v^{(n+1)}, t^{(n+1)}, x_d^{(n+1)}, y_d^{(n+1)}\right) = 0 \end{cases} \tag{2}$$

where $w^{(n)}$ are the variables of the sequential parts of the circuits, i.e., the "registers" of the digital system, evaluated at $t = t^{(n)}$, $v^{(n)}$ the purely combinatorial variables, corresponding to the "wires", and $x_d^{(n)}$, $y_d^{(n)}$ are the variables coming from the analog part and preconditioned by the μ_{a2d} block.

Obviously, all variables in Eq. (1) and Eq. (2) must have consistent initial conditions. Moreover, the *analog to digital* and *digital to analog* mappings must be well defined

$$\begin{cases} x_d = \mu_{a2d}(x) \\ y_d = \mu_{a2d}(y) \\ w_a = \mu_{d2a}(w) \\ v_a = \mu_{d2a}(v) \end{cases} . \tag{3}$$

Table 1.

$W_1^{(n)}$	$W_2^{(n)}$	$W_1^{(n+1)}$	$W_2^{(n+1)}$	$V_1^{(n+1)}$
0	0	1	0	0
1	0	0	1	0
0	1	1	1	1
1	1	0	0	0

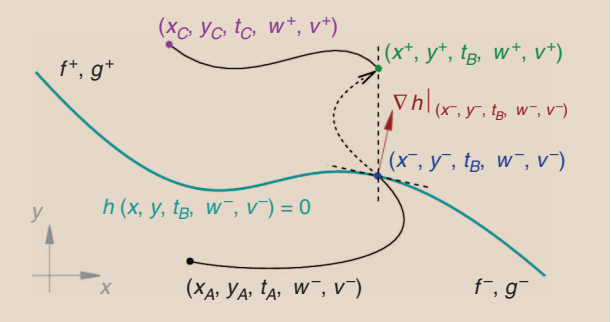


Figure 3. An example of two-dimensional "jumping" trajectory highlighting, as far as the trajectory itself hits the manifold h, the continuity of the differential variable x and the discontinuity of the algebraic variable y.

The μ_{a2d} and μ_{d2a} mapping functions (see also Fig. 2) behave as (pseudo) analog-to-digital and (pseudo) digital-to-analog converters.

Digital variables act on the analog part of the circuit as external parameters that can undergo "instantaneous" changes of value. These parameter variations are equivalent to instantaneous changes of the vector field $f(\cdot)$ and algebraic part $g(\cdot)$; but changes of these parameters, i.e., digital variables w and v, can be caused, in general, by changes of the analog variables x and y. The effects of the analog to digital interaction can thus be seen, from the analog side, as equivalent to the *jump* phenomena described by *hybrid* systems. Discontinuities can occur on all variables except x, which are the electrical state variables that are assumed to be always continuous. This situation is depicted in Fig. 3 where $h(x, y, t, w, v) = 0$ is the equation of a "surface" that may change with time and represents the boundary of a discontinuity in the circuit. This surface is more appropriately called *time-varying manifold* and separates $f^- \equiv f(\cdot,\cdot,\cdot,w^-,v^-)$, $g^- \equiv g(\cdot,\cdot,\cdot,w^-,v^-)$ from $f^+ \equiv f(\cdot,\cdot,\cdot,w^+,v^+)$, $g^+ \equiv g(\cdot,\cdot,\cdot,w^+,v^+)$. These functions describe how the circuit evolves in each one of the two regions, and are commonly known as *vector field*, in the simple case of a system described by ODEs the vector field is the right hand side of the equation.

At time t_B the trajectory hits this manifold, the vector field changes, and all variables, except x, may jump to a new value.

This model clearly is "switching", since we have different vector fields on each side of manifold $h(x, y, t_B, w, v) = 0$, and "impacting" since we allow discontinuities in variables describing the digital part, it is thus in the class of "hybrid" systems.

Consider for example the circuit introduced in Sec. II,[2]

$$\begin{cases} C\dot{x}_1 + x_2 + \dfrac{x_1}{R_1} + \dfrac{x_1 - E_o}{R_2 + v_1(R_3 - R_2)} = 0 \\ L\dot{x}_2 - x_1 = 0 \end{cases} , \tag{4}$$

the effect of the digital part of the circuit on the analog one is viewed as a simple time-varying parameter acting on the expression of the vector field in Eq. (4).[3]

The evolution of the digital variables of the circuit is summarized in Table 1.

This hybrid system modeling framework can similarly describe AMS circuits that are much more complex and that *inherently* induce switching, impacts, manifolds and reset functions. These events and corresponding manifolds are in most cases naturally defined within the

[2]In this example, the analog part is described using state equations, and not a full MNA formulation, thus avoiding the algebraic part of Eq. (1).
[3]Assuming that the logical values *true* and *false* at the digital side are directly mapped to the real numbers 1 and 0 at the analog side.

netlists or the behavioral/digital models that describe the AMS circuit. With reference to our example, $h(x)$ belongs to the zero crossing detector and its definition is implicit in the VERILOG description of this component.

This means that it is possible to set up a simulation environment that is based on well known and standard formal languages and uses all the mathematical tools available for hybrid dynamical systems, without requiring that the designer changes in any way his/her habits. We will see in the following that this is quite appealing since it allows to analyze AMS circuits as if they were analog.

IV. The Analyses That Can Be Extended to AMS

Variational methods are at the heart of many numerical time domain simulation algorithms. Most of these *analysis methods*, as they are usually collectively called by designers, are available with different names, different options and parameters, and are limited to analog-only circuits, in several commercial simulators. These methods are well known to analog designers, but may be unfamiliar to most other people. For this reason, and in order to better appreciate their extension to AMS, a brief review of the analog-only version of these analyses is useful.

A. Variational Models and Switchings

Variational models are, essentially, the linearization of a system in the neighborhood of one of its trajectories. Consider a system with state vector $x(t)$, whose evolution is given by the set of differential equations

$$\dot{x} = f(x(t), t), \quad x \in \mathbb{R}^n. \tag{5}$$

Since building a variational model requires linearization, the vector field that generates the trajectory must have, in each point along this trajectory, a well defined derivative, i.e., *smoothness* of trajectories must be guaranteed.

This is true for conventional "analog-only" circuit models, even if the possible presence of parts with dynamics having very large time scale differences (i.e., "stiff" systems) may generate numerical problems that are very close to those of non-smooth systems but in most cases solvable with relatively simple workarounds.

Non-smooth systems, i.e., those exhibiting non-differentiable or even discontinuous trajectories, such as AMS circuit models, do not admit "as is" a variational model. This means that simple circuit analysis methods, i.e., those that do not require a differentiable system of equations, can be exported with very little effort to AMS circuits. This is the case of basic time-domain analysis, i.e., solution of a simple Initial Value Problem (IVP), known as "transient" analysis which are already available for AMS circuits.

More advanced methods of analyzing the dynamics and stability require different mathematical tools. One important ingredient of the variational method is the evolution operator called *fundamental solution matrix* that describes sensitivity of system trajectories to initial conditions, i.e., the first-order expansion of the dynamics of perturbations around a nominal trajectory.

Suppose that the system governed by (5) has an initial condition x_0 at time t_0, and we perturb it to $\bar{x}_0$ such that the initial perturbation is $\xi(t_0) = x_0 - \overline{x}_0$. If the original trajectory and the perturbed trajectory evolve up to a time t, the perturbation at the end of the period can be related to the initial perturbation by

$$\xi(t) = \Phi(t, t_0)\, \xi(t_0).$$

Here $\Phi(t, t_0)$ is the state transition matrix, which is a function of the initial state, the initial time, and the final time.

If the system is Linear Time Invariant (LTI), for which (5) takes the form

$$\dot{x} = Ax + Bu,$$

then the state transition matrix is given by the matrix exponential

$$\Phi(t, t_0) = e^{A(t-t_0)},$$

which can be explicitly evaluated using, for instance, MATLAB'S `expm` function. If the system is not LTI, the state transition matrix has to be obtained numerically.

Now, if the initial condition is located on a periodic orbit, and if the system is evolved for the full period T, then we obtain an equation relating the perturbation at the end of the period to that at the beginning of the period, i.e.,

$$\xi(T) = \Phi(T, 0)\, \xi(0).$$

In this approach, the stability of such a periodic orbit is understood in terms of the evolution of perturbation. If the initial condition is perturbed and the solution converges back to the orbit, then the orbit is stable. The stability margin can be assessed from the rate of convergence.

The matrix $\Psi = \Phi(T, 0)$ is called the *principal matrix* or *monodromy matrix*. The eigenvalues of this matrix indicates the stability of the orbit. If all the eigenvalues have modulus less than 1, the system is stable. If this matrix has at least one eigenvalue whose modulus is greater than 1, the system is unstable. The computation of the monodromy matrix is thus a significant element for the stability analysis of any periodic behavior.

Now let us turn to the specific problem of AMS systems. Typically, in such a system a periodic orbit would be composed of passages through a number of subsystems. Suppose the state evolves from the instant t_A to the instant t_B, and the state transition matrix for that period is $\Phi(t_B, t_A)$; then it evolves from the instant t_B to t_C, and the state transition matrix in that interval is $\Phi(t_C, t_B)$. It is known that if the evolution from t_A to t_C is smooth (everywhere differentiable), then the state transition matrix from t_A to t_C is simply the product of the two matrices $\Phi(t_C, t_B)\Phi(t_B, t_A)$. However, if a switching occurs at $t = t_B$, the evolution becomes non-smooth at that point since the governing equations before and after the event are different. In such a situation the state transition matrix for the evolution from t_A to t_C *cannot* be obtained as a simple product of the state transition matrices across each part. One, additionally, has to consider the state transition matrix across the switching event.

The state transition matrix S that relates the perturbation just after the switching event to that just before is given as

$$\xi(t_{B^+}) = S\,\xi(t_{B^-}).$$

This matrix is called the "saltation matrix" [55]–[57] which is expressed as

$$S = 1_n + \frac{(f^+ - f^-)\,\eta^T}{\eta^T f^- + \left.\dfrac{\partial h}{\partial t}\right|_{t=t_B}}\ , \qquad (6)$$

where 1_n is the n-th order identity matrix, $h(x,t) = 0$ represents the switching condition (a surface in the state space of the system), $\eta = \nabla_x h(x,t)\big|_{t=t_B}$, i.e., the gradient of $h(\cdot,\cdot)$ with respect to state space, is the vector normal to the switching surface, and η^T is its transpose. f^- represents the right hand side of the differential equations before the switching occurred, and f^+ represents the right hand side of the differential equations after the switching. From this expression the saltation matrix can be evaluated for each switching event.

With this theoretical framework, analog-digital systems can be subject to the same analyses that are applicable to purely analog systems. The only change is that, in finding the monodromy matrix one has to insert the saltation matrices at appropriate places whenever a switching occurs. Thus, if a periodic orbit consists of passages through k subsystems and $k-1$ switching events, then the monodromy matrix has to be expressed as

$$\Psi(T,0) = \Phi(T, t_{k-1})\,S_{k-1}\cdots S_2\,\Phi(t_2,t_1)\,S_1\,\Phi(t_1,0)\ . \quad (7)$$

Excellent books that can be used to have a deeper insight in these topics are [58], [59].

B. Shooting Method

The SH method allows to solve a Boundary Value Problem (BVP) as a small number of simpler IVPs [60], [61]. In a general situation, an n-th order ODE allows setting n independent boundary conditions—these can be initial conditions (as in IVPs), final conditions, or a mix of the two. In the classic ballistic problem one has a fixed position of the cannon and of the target which are the boundary conditions, but has freedom in the tilt of the cannon and does not care about the angle of arrival of the cannonball. If a first shot misses the target, the gunner will change the tilt of the cannon, evaluate how much closer or farther he gets from his objective and finally adjust the tilt in order to (hopefully) hit the target with the next shot. The key of the gunner's method is the perturbation of the initial guess and evaluation of the *sensitivity* of the solution (the arrival position of the cannonball) to this perturbation.

The situation most interesting and useful for circuit design can be visualized as a variant of the ballistic problem where the gunner is using a boomerang instead of a cannon. In this case the boundary conditions represent a periodicity constraint. Let us consider this situation with more detail, since it is the basis for all other analysis methods.

The circuit can be *autonomous* or *non-autonomous*, simply meaning, in the second case, that the Right Hand Side (RHS) of the equation governing its dynamics has explicit dependence on time.[4] For the system of state equations in (5), a boundary condition $x(T+t_0) = x(t_0)$ for some time value T[5] defines, if it exists, a T-periodic orbit Ω. Using any numerical integration method it is possible to find the solution of the IVP from time t_0, with some initial conditions $x(t_0)$, to time $t_0 + T$. Computing in parallel the evolution of $\Phi(t, t_0)$, i.e., the state transition matrix associated to the original nonlinear ODE, provides the sensitivity of this solution to initial conditions. Obviously, with AMS systems, this matrix must be obtained through (7). Using the sensitivity information in Φ, the initial conditions of the IVP are iteratively corrected until condition $x(T+t_0) = x(t_0)$ is (approximatively) met obtaining solution Ω. Along these iterations $\Phi(t_0+T, t_0)$ converges towards the so called Ψ principal matrix.[6]

This is a direct mathematical transposition of the boomerang problem: the state transition matrix represents a measure of the sensitivity of the final conditions with respect to initial ones, i.e. of the boomerang return point

[4]Since we are looking for a periodic steady state, in the *non-autonomous* case the circuit is *periodically* forced.

[5]If the circuit is autonomous, period T must also be determined. This is normally done augmenting the SH equations with a *phase condition* (for details see [58], [62]).

[6]The Ψ principal matrix computed along a periodic orbit Ω is a linearized representation of the evolution operator of the corresponding nonlinear system in a neighborhood of Ω itself.

as a consequence of a specific choice of launch conditions. The sequence of IVPs solved for different initial conditions is equivalent to a sequence of tilt adjustments.

The SH method is available in PAN for analog circuits as well as AMS circuits. For instance, if one were interested in evaluating the periodic steady state of the simple AMS oscillator introduced in Sec. II, the following command would be added in the circuit netlist file

```
Sh0 shooting period=(2*pi)*4
+            autonomous=yes
+            minper=(2*pi)*3.2
```

where `Sh0` is the name of the analysis, `shooting` is the analysis type followed by the parameters; `period` is the guess value of the working period; the simulator automatically determines the correct working period of the oscillator since the `autonomous` boolean parameter is set to "true". The `minper` parameter sets the minimum value that can be assumed by the period automatically determined by the simulator. This is necessary since the oscillator is refilled every 4 oscillations of the *LC* tank, which resonates at 1 rad $\sec^{-1}$.

The result of the simulation is reported in Fig. 4.

C. Stability of Steady State Periodic Solutions

Stability of Ω can then be determined by observing the eigenvalues of Ψ, known as *Floquet (characteristic) multipliers*. Stability requires all the multipliers to reside inside the unit circle (i.e. a modulus smaller than one), with at most one equal to one if the system is autonomous.

In PAN this kind of stability analysis is a by-product of the SH method and can be performed on AMS or non-AMS circuits by adding the **bold** option to the SH analysis command

```
Sh0 shooting period=(2*pi)*4autonomous=yes
+            minper=(2*pi)*3.2
+            floquet=yes
```

For our case study, the Floquet multipliers corresponding to the limit cycle in Fig. 4 are $\mu_1 = 0.99999$ (that corresponds to a value that is theoretically equal to 1) and $\mu_2 = -0.17516$.

D. Periodic Small Signal Analysis and Time-Varying Transfer Functions

A very important application of variational-based analysis is that given by superposition of a "small" periodic signal to a large one, such as the situation found in a carrier modulation system. In this case, it is possible to extend the variational representation to consider also external periodic inputs.

From a mathematical point of view there are no particular limitations on the number of different periodic input waveforms, and on the shape of each one of these. From a practical point of view, one is typically interested in efficiently estimating the effect of a single simple sinusoidal input since the effect of many, more complex, inputs is easily computed by superposition. This kind of analysis is known as Periodic AC (PAC) [63], [64].

The PAC analysis computes the solution of a periodically driven circuit perturbed by a small sinusoidal input at an arbitrary frequency. The nominal unperturbed circuit is represented by a linear periodic time varying model, whose response to a small periodic perturbation is composed by sinusoids at different frequencies; in practice PAC computes a series of transfer functions, one for each frequency, from the single perturbation source to each node of the circuit.

To exemplify the PAC analysis a sinusoidal current generator of magnitude equal to 1 is added in parallel to linear resistor R_1 in Fig. 1 and the

```
Pac01 pac freq=10mHz
```

command is added in the netlist. The simulation results are shown in Fig. 5. As for the previous SH case, `Pac01` is the analysis name and `pac` is the analysis type; the `freq` parameter specifies the frequency of the PAC generator. Note that the PAC analysis can be performed only after a successful SH analysis since the circuit must be linearized along its steady state solution. Furthermore, since it is a time-varying linear analysis, magnitudes of the results linearly scale with that of the source.

A generalization of the PAC analysis, which is the conceptual basis for the Periodic Noise (PNOISE) analysis described in the sequel, is the *periodic transfer function*

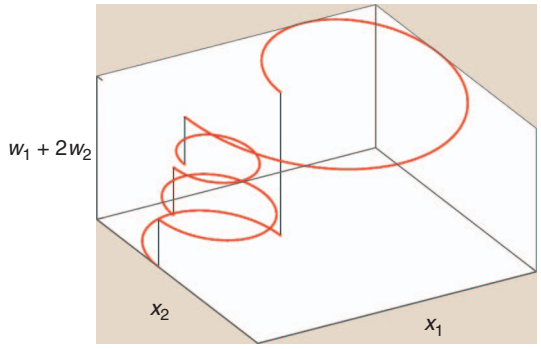


Figure 4. The stable periodic steady state of the AMS oscillator used as a case study. The trajectory is plotted in the $x_1, x_2, w_1 + 2w_2$ space and the black vertical segments are guides (not truly part of the trajectory) that represent the discontinuities corresponding to changes of the counter value variable $w_1 + 2w_2$. The computed working period is $T = 22.1049$.

analysis that computes the transfer function from every source in the circuit at any frequency to a single output at a single frequency. It is in some ways the reverse of the "one to many" ratio of PAC.

E. Periodic Noise Analysis

PNOISE analysis is possibly the most well-established tool to evaluate the effects of small-signal (cycle-stationary) additive noise sources in circuits characterized by a periodic steady-state behavior [63], [65]. The term *additive* means that it is assumed that the noise effects "sum up" to the large signal solution of the *noiseless* circuit. This technique, suited to simulate total noise and phase noise, is based on the variational representation of the circuit model equations. The cycle-stationary noise sources are modeled as a set of sinusoidal equivalent sources and their power (in a 1 Hz bandwidth) is transferred (frequency-wise) to a given output node of the circuit where all the resulting power spectral density contributions are added. In this way the PNOISE analysis inherently handles the *noise folding* phenomenon.

Transfer functions are computed by solving the variational model in the time/frequency domain. From a numerical point of view it is thus equivalent to a number of periodic small signal analysis.

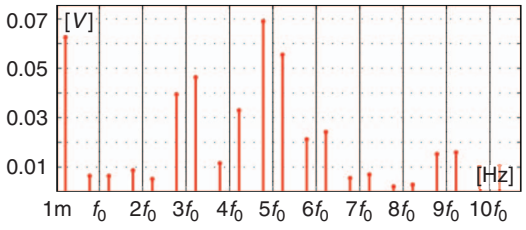


Figure 5. The amplitude spectrum of the small signal voltage across R_1. (see Fig. 1) due to a sinusoidal current generator of magnitude equal to 1 and frequency equal to 10 mHz added in parallel R_1.

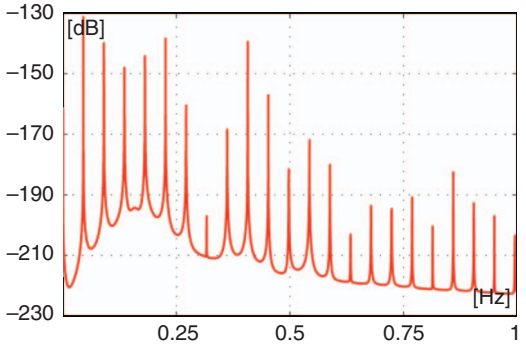


Figure 6. The PSD in dB of the total noise computed by the proposed approach for the oscillator in Fig. 1.

For the chosen AMS reference oscillator, we performed a PNOISE analysis, by assuming that only R_1 generates thermal noise, through the command

```
Pnoise pnoise start=10u stop=1 dec=1000
+              onodes=["x1"] sweeptype=0
```

The `onodes` parameter specifies the node/s at which periodic noise has to be computed, `sweeptype` specifies how Power Spectral Density (PSD) of the noise has to be referred to: in general it is with respect to the oscillator fundamental or the DC component as in this case. If more than one uncorrelated noise source are considered, each contributes to the total noise through the square sum of the time-varying transfer function that relates the noise source to the desired output. The obtained PSD of total noise across the capacitor C, i.e., the additive noise over voltage x_1 is shown in Fig. 6.

F. Phase Noise Analysis in Demir's Unified Theory

A rigorous understanding of the phase noise phenomenon in oscillators was developed by Demir et al. [66]–[68]. Basically, it was shown that, under a set of hypotheses that will not be discussed here, once the PSD of total noise was computed, it can be decomposed in its *phase noise* and *amplitude noise* components, by projecting the total noise along a function $v_1(t)$, referred to as PPV in literature, which is available if the fundamental matrix is defined.

For the case study of AMS oscillator we can compute the evolution of $v_1(t)$ through the option marked in **bold** in the command line below (results are shown in Fig. 7)

```
Sh0 shooting period=25 autonomous=yes
+            minper=(2*pi)*3.2 floquet=yes
+            eigf=yes
```

Note that Floquet eigenfunctions are re-sampled and computed on an even time mesh in order to increase efficiency. A suitable analysis parameter controls the number of time samples.

G. Evaluation of Lyapunov Exponents (Aperiodic Steady State)

This is not a typical circuit design analysis, even though it may have some importance in a few situations. Stability results found for periodic solutions can be extended to generic solutions, including non-periodic or even chaotic ones, through the definition of Lyapunov Exponents (LEs). These represent a measure of how fast arbitrarily close trajectories move away from each other, and the number of LEs equals the dimension of state space, since different directions can have different expansion or contraction

rates. Several numerical techniques to compute LEs of model-explicit, nonlinear, and finite dimensional dynamical systems can be found in the literature [69]. All of these require the system to be ruled by a vector field smooth enough to permit the evaluation of the system fundamental matrix since the definition of LEs is based on the evolution in time of its singular values. Several attempts have been made to extend the computation of these important indicators to different classes of discontinuous systems [70]–[72]. The saltation matrix operator was applied in [73] for the computation of LEs for hybrid neurons.

With PAN LEs can be computed in a straightforward way through a sufficiently long time domain analysis as shown, for the considered case study, by the command

```
Tr0 tran tstop=(2*pi)*200 uic=yes
+          lyapunov=yes
```

where `tstop` sets the ending time instant of the analysis, `uic` tells that the initial conditions given by the user must be used, otherwise the circuit may not "move" from its equilibrium point automatically computed by the DC analysis. The `lyapunov` parameter turns on computation of LEs.

V. Examples

A. A Digitally Controlled DC/DC Converter

We now illustrate the usefulness of the algorithms just described on the typical power electronic converter shown in Fig. 9. Such converters are widely used in industry, and designers need to frequently assess their performance at each step of the design stage. Currently this is done with approaches such as brute-force simulation (by exploiting long lasting transient simulations) or averaged modeling.[7] We show that far more information can be extracted using the proposed algorithms. The system considered is a simple DC/DC converter—a two-phase boost converter—that supplies the load resistor R_o at a voltage level higher than that of supply E_o [74].

Despite the number of components in this circuit, its behavior is quite simple to describe and the few equations necessary to model its dynamic behavior are straightforward to derive. Nevertheless, the circuit is nonlinear and switching is controlled by digital logic that produces the on/off signals for the switch. If switches S_1 and S_2 open and close simultaneously, the dynamics of the circuit can be easily summarized:

- S_1 and S_2 closed: currents i_{L_1} and i_{L_2} through inductors L_1 and L_2 increase. This happens since the currents flowing in the inductors follow the low-resistance path represented by the closed switches and only a negligible portion of these currents flows in the diodes that are reverse biased since the output voltage is positive. The C_o capacitor discharges through the R_o load.
- S_1 and S_2 open: $i_{L_1} = i_{D_1}$ and $i_{L_2} = i_{D_2}$, i.e., the currents in L_1 and L_2 flow through diodes D_1 and D_2, respectively. The diodes are forward biased since in the previous working phase the output voltage decreased and the inductors charged. The load is then supplied and C_o, which smooths the output voltage ripple due to the pulsed i_{D_1} and i_{D_2} currents, is recharged.

In the actual circuit the two switches are not synchronous and S_2 is always closed after a $T/2$ delay with respect to the time instant at which S_1 is closed. This choice halves the period of the output current ripple, improves the converter performance but slightly modifies the chain of events described above, even though

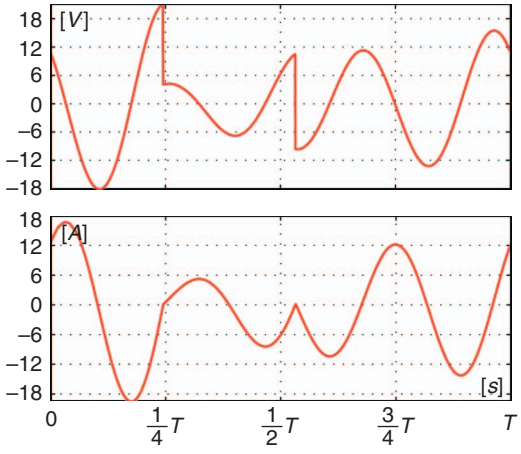


Figure 7. The periodic eigenfunction $v_1(t)$ for the oscillator in Fig. 1. The component of $v_1(t)$ related to x_1 and x_2 are reported in the upper and lower panel, respectively.

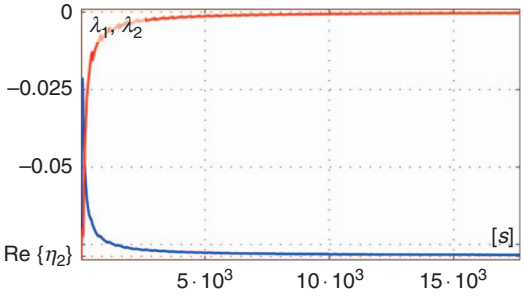
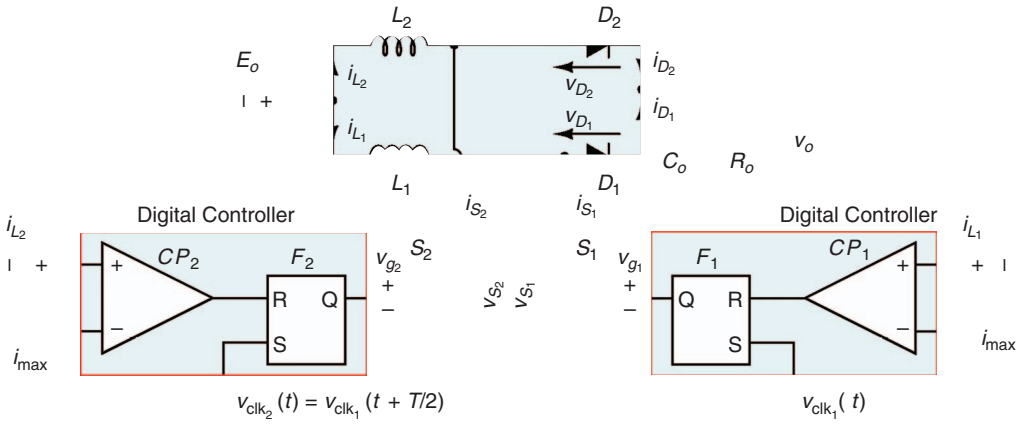


Figure 8. The red and the blue curves represent the dynamical evolution, during their computation, of the LEs of the limit cycle shown in Fig. 4. λ_1 (in red) tends to the theoretical value 0 and corresponds to the Floquet multiplier μ_1. The asymptotic value of λ_2 (in blue) is $(\log \mu_2/T)$.

[7]Basically, a designer uses the controller-converter loop gain and transfer function to study stability with respect to the parameter values.

the working principle is basically the same. Each switch opens when the current flowing through the corresponding inductor reaches the maximum value i_{max}.

The digital controller governing the switches is composed of two identical parts (see the red boxes in Fig. 9), one for each phase (switch) of the boost converter. Let us



The Netlist of the Two Phase Boost Converter

```
Eo    vdd gnd vsource vdc=5
L1    vdd x1 inductor l=1.5mH
S1     x1 gnd vg1 SWITCH
D1     x1 out DIO GD=1k
L2    vdd x2 inductor l=1.5mH
S2     x2 gnd vg2 SWITCH
D2     x2 out DIO GD=1k
Co    out gnd capacitor c=10u
Ro    out gnd resistor r=40
Il1   il1 gnd ccvs sensedev="L1" gain1=1
Il2   il2 gnd ccvs sensedev="L2" gain1=1
Imax ref gnd vsource vdc=0.1
Clk   clk gnd vsource v1=0 v2=VDIG
+      tr=1u tf=1u period=PRD width=PRD/2
X0    clk gnd a2d dignet="Cntr.clk" vt=1/2
X1    il1 ref a2d dignet="Cntr.x1" vt=0
X2    il2 ref a2d dignet="Cntr.x2" vt=0
X3    vg1 gnd d2a dignet="Cntr.g1" vl=0 vh=1
X5    vg2 gnd d2a dignet="Cntr.g2" vl=0 vh=1
verilog_include cntr.v
model DIO nport veriloga="dio.va"
model SWITCH nport veriloga="switch.va"
```

Switch Model

```
module SWITCH(p,n,gate);
inout p, n, gate;
electrical p, n, gate;
parameter real VTH=0.5;
analog begin
  if( v(gate) > VTH )
    v(p,n) <+ 0.0;
  else
    i(p,n) <+ 0.0;
end
endmodule
```

Diode Model

```
module DIO(a,k);
inout a, k;
electrical a, k;
parameter real GDon =1k;
parameter real GDoff=1u;
analog begin
  if( v(a,k) > 0.0 )
    i(a,k) <+ GDon  * v(a,k);
  else if( v(a,k) < 0.0 )
    i(a,k) <+ GDoff * v(a,k);
end
endmodule
```

Controller Model

```
`timescale 1ns/1ns
module Cntr(clk,x1,x2,g1,g2);
input clk, x1, x2;
output g1, g2;
reg g1, g2;
initial begin
g1 = 0; g2 = 0;
end
always @(posedge clk)
    if( x1 == 0 ) g1 = 1;
always @(negedge clk)
    if( x2 == 0 ) g2 = 1;
always @(posedge x1) g1 = 0;
always @(posedge x2) g2 = 0;
endmodule
```

Figure 9. The schematic of the two-phase boost converter and of the controller. The i_{max} pins are driven by a constant voltage source (not shown) that sets the maximum allowed current through the L_1 and L_2 inductors. The v_{clk} pins are driven by a fixed duty cycle square wave generator (not shown) at 10 kHz. The S_1 and S_2 switches close when the corresponding v_{g1} and v_{g2} voltage is at the high logic level (1_L). They open at the low logic level (0_L). The dependent voltage sources sense the i_{L_1} and i_{L_2} currents and drive the comparators at voltage levels equal to the corresponding currents.

focus on phase "1" since the same holds, *mutatis mutandis*, for phase "2". The F_1 FLIP-FLOP is set *at each rising edge* of the $v_{ckl_1}(t)$ clock signal. The FLIP-FLOP closes S_1 by setting voltage v_{g_1} at the logical high level. The CP_1 comparator compares the i_{L_1} current with the $i_{\max}$ maximum value. When $i_{L_1} \geq i_{\max}$, the FLIP-FLOP is reset, v_{g_1} falls to the low logic level and the switch opens.

To allow a "pen-and-paper" analysis as in [74] and thus a direct comparison of results with AMS simulation, at first we model the switches and the diodes as behavioral ideal piece-wise linear elements through the VERILOGA language. More specifically the S_k switch model is

$$\alpha_i i_{sw_k} + \alpha_v v_{sw_k} = 0$$

where $\alpha_i = 0$, $\alpha_v = 1$ when $v_{g_k} == 1_L$ and $\alpha_i = 1$, $\alpha_v = 0$ when $v_{g_k} == 0_L$ and the D_k diode equation is

$$i_{D_k} = \iota(v_{D_k}) = \beta_v v_{D_k}$$

where $\beta_v = g_I$ when $v_{D_k} < 0$ and $\beta_v = g_D$ when $v_{D_k} \geq 0$, with $g_D \gg g_I > 0$.

In other words, the switches behave as open circuits when they are open and as short circuits when they are closed. The diodes are modeled as resistors characterized by a piece-wise linear voltage-dependent resistance.

The dynamics of the analog part of the system is given by the equations

$$\begin{cases} L_1 \dot{x}_1 = (1 - w_{a_1})[x_3 + \iota^{-1}(x_1)] + E_o \\ L_2 \dot{x}_2 = (1 - w_{a_2})[x_3 + \iota^{-1}(x_2)] + E_o \\ C_o \dot{x}_3 = -\dfrac{x_3}{R_o} + (1 - w_{a_1})x_1 + w_{a_1}\iota(-x_3)' \\ \qquad + (1 - w_{a_2})x_2 + w_{a_2}\iota(-x_3) \end{cases} \tag{8}$$

where the analog electrical variables in Fig. 9 are mapped as $x_1 = i_{L_1}$, $x_2 = i_{L_2}$, $x_3 = v_o$ and $\iota^{-1}(x_1)$ is the inverse of the diode equation. For the sake of simplicity, the sequential functions corresponding to the internal evolution of the FLIP-FLOPs are not reported. The digital voltages v_{g_k} become w_{a_k} (for $k = 1, 2$) adopting a $\mu_{d2a}(\cdot)$ block such that $\mu_{d2a}(1_L) = 1$ and $\mu_{d2a}(0_L) = 0$. It is worth noting that the v_{g_k} voltage is mapped to a sequential variable since it represents the state of the F_k FLIP-FLOP, i.e., a single digital register.

After having realized that Eq. (8) belongs to the family of problems described by (1), we can focus on the switching manifolds involved in the dynamics of the system

$$\begin{aligned} h_1(x) &= x_1 - i_{\max} = 0 \\ h_2(x) &= x_2 - i_{\max} = 0 \\ h_3(x) &= x_1 = 0 \\ h_4(x) &= x_2 = 0 \\ h_5(x) &= x_3 = 0\,, \end{aligned} \tag{9}$$

where $h_k(x)$ (for $k \in \{1,2\}$) induces the reset function $w_{a_k} = \mu_{d2a}(0_L) = 0$, i.e., an impact in the digital state of the circuit corresponding to the opening of S_k. We remark that, as far as the closing of S_k is concerned, it is not governed by a manifold in the state space but it is triggered by the rising front of the $v_{ckl_k}(t)$ external signal. The remaining three manifolds rule the switching of the diodes characteristics.

To obtain the large signal steady state solution of the converter, we performed the analyses using the instruction

```
Tr tran stop=5*100E-6 uic=yes method=2 order=2
+   Sh shooting method=2 order=2 restart=no
+   floquet=yes period=100E-6 cmin=no
```

We first performed a time domain large signal analysis (`Tr`) to start up the converter and then a SH one (`Sh`) using as initial condition the solution at the last time point computed by `Tr` (i.e. using option `restart=no`).[8] In Fig. 10 the $v_o(t)$ output voltage, the currents through the L_1 and L_2 inductors and the signals driving the S_1 and S_2 voltage controlled switches are shown. It is easy to see that the falling edges of these driving signals are positioned at the time instants when the currents through the L_1 and L_2 inductors hit the manifolds at $i_{\max} = 100$ mA (see the circuit netlist) [74]. Since this converter shows different behaviors and instability depending on the level of the switching manifold, we performed a stability analysis by computing the Floquet multipliers while increasing $i_{\max}$ from 100 mA till 300 mA as reported in [74]. This analysis was first performed with behavioral models of the switches and diodes as reported in Fig. 9 and then we repeated the same analysis after having substituted these ideal elements with accurate MOS transistor and diode power models.[9] The loci of the most significant Floquet multipliers (three: one real and two forming a complex pair for some values of the parameter) with behavioral elements is shown in blue and those with power MOS transistor and diode models is shown in red in Fig. 11. It can be seen that the complex pair of Floquet multipliers goes outside the unit circle in both cases giving rise to a Neimark-Sacker bifurcation [62]. The behavior with power component models is very similar to that with behavioral models and shows the effectiveness of the approach. We shall underline the versatility of the tool and approach since the designer has simply to switch through a simple command from ideal models to more

[8] Comments concerning the `Tr` and `Sh` options `method`, `order` and `cmin` can be found in the PAN online help.

[9] For the sake of conciseness the netlist of the circuit after the insertion of MOS transistors and diodes is not reported here but can be downloaded from the PAN website.

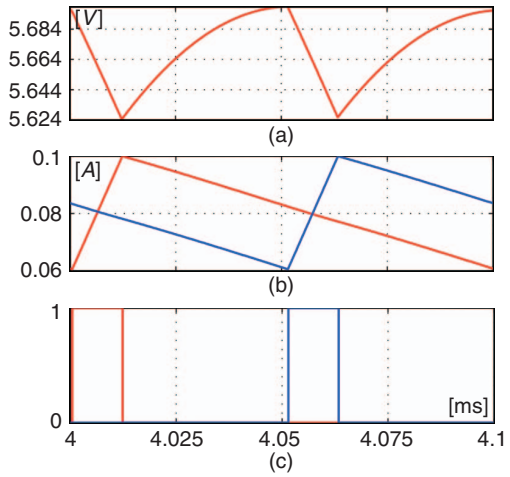


Figure 10. (a) The $v_o(t)$ periodic output voltage of the converter shown in Fig. 9. (b) Currents through the L_1 (red) and L_2 (blue) inductors, respectively. (c) Triggering signals of the S_1 (red) and S_2 (blue) switches.

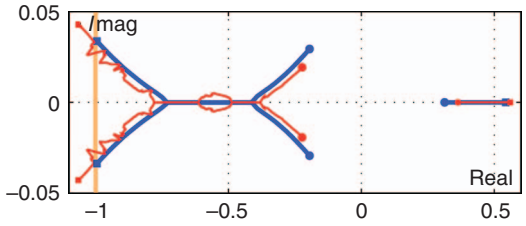


Figure 11. The loci of points described by the three largest Floquet multipliers related to the converter shown in Fig. 9. Blue curves refer to the analysis with behavioral elements (switches and piece-wise linear diode characteristics). The red curves refer to the analysis with power MOS transistor and diode models. Such multipliers are exactly 3 when MOS transistors and diodes are not used and their number increases if these components are used. This happens because of the state variables introduced by parasitic components embedded in their model. The orange (almost vertical) line is a portion of the unit circle in the complex plane. When a Floquet multiplier goes outside this circle the circuit becomes unstable. Dots show the starting position of the Floquet multipliers, i.e., when $i_{\max} = 100\,\text{mA}$ and squares show the final position, i.e. when $i_{\max} = 300\,\text{mA}$.

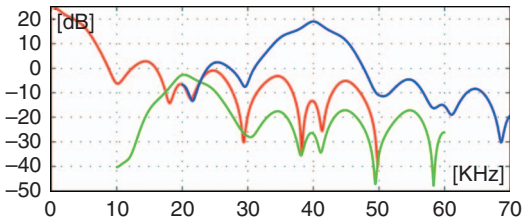


Figure 12. The moduli of the small signal beats at the output of the boost converter at f_u, $f_o + f_u$ and $2f_o + f_u$ frequencies generated by a small sinusoidal source that varies the $i_{\max}$ switching reference current level.

realistic ones and completely ignore how problem formulation and manifolds modify since this is automatically taken into account by the simulator.

To have an idea of the closed loop input (inductor current control level) output (v_o voltage) transfer function of the converter we performed a PAC analysis through the command

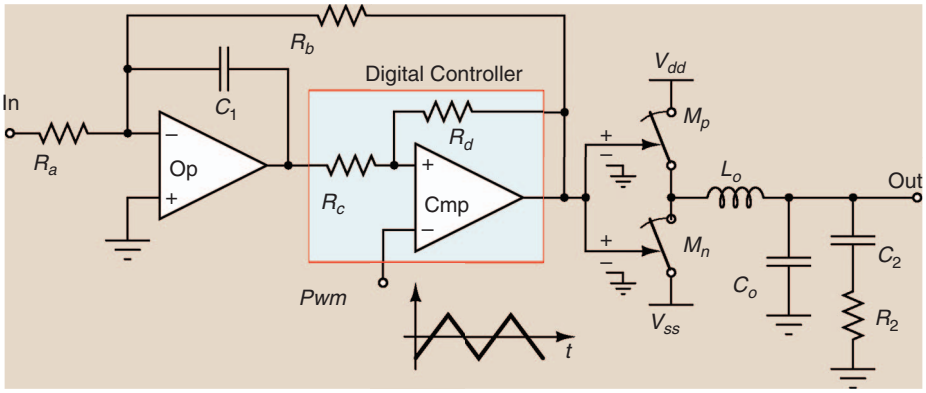
```
Pac pac start=1 stop=50k lin=10k
```

where `start` is the starting value of the frequency sweep, `stop` is the final value and `lin` specifies the type of frequency grid (linear in this case) and the number of frequency samples.

The $u(t)$ small signal at the f_u frequency added to $i_{\max}$ affects the v_o large signal solution by giving contributions at each component of the large signal spectrum. If for example the DC, the fundamental at $f_o = 10\,\text{kHz}$ (the clock period is 100 µs), and the first harmonic are considered, we have that the small signal beats with these components and effects locate as right sided signals at $f_u, f_o + f_u$ and $2f_o + f_u$. The moduli of these components of the small signal output v_o are shown in Fig. 12 in red, green and blue, respectively. It can be easily seen that effects due to the $u(t)$ up conversion at $2f_o$ are relevant being about $+20\,\text{dB}$ at $40\,\text{kHz}$.

B. A Class-D Audio Amplifier

The schematic of the class-D audio amplifier is shown in Fig. 13. The input signal at node *In* is conditioned by the feed-back loop constituted by the *Op* operational amplifier, the R_a, R_b resistors and the C_1 capacitor. The signal at the output of *Op* is compared by the *Cmp* comparator to the sawtooth waveform at the *Pwm* node. The model of the comparator is ideal and implemented through a behavioral model and generates a digital output, i.e., its output instantaneously switches between its upper and lower values (and vice versa). This circuit leads to a DAE model with a discontinuous vector field. In fact by circuit inspection, we can see that the instantaneous switching of *Cmp* and of the driven M_p and M_n switches leads to a discontinuous voltage across L_o and to a discontinuous current through C_1. Furthermore, the manifold characterizing *Cmp* is voltage dependent, i.e., the time instant at which there is a commutation of *Cmp* depends on the instantaneous value assumed by the voltages at the output of *Op* and at the *Pwm* node. The output of *Cmp* drives the M_p and M_n voltage controlled switches. The models of these switches are such that M_n closes when the driving signal is larger than a given threshold while M_p closes when the driving signal is less than this threshold. The sawtooth waveform at node *Pwm* has a period of 1 µs. The pulse width modulated large signal (and



The Netlist of the CLASS-D Audio Amplifier

```
parameters VDIG=1 PRD=1u
Vdd vdd gnd vsource vdc=20
vss gnd vss vsource vdc=20
Rin in neg resistor r=1k
C1 neg o1 capacitor c=1n
Op1 o1 gnd gnd neg vcvs gain1=1M
Rb neg gt resistor r=10k
Lim lm gnd o1 vcvs func=limit(v(o1),-(1-1m)*VDIG,(1-1m)*VDIG)
Cmp gt gnd pwm lm vcvs func=v(lm,pwm) > 0 ? VDIG : -VDIG
+                       digital=yes trtime=1n
Sw1 vdd x1 gt gnd VSW1
Sw2 x1 vss gt gnd VSW2
Lo x1 out inductor l=10u/3
Co out gnd capacitor c=10u/3
R2 out x2 resistor r=100
C2 x2 gnd capacitor c=1u
Rl out gnd resistor r=4
Vsaw pwm gnd vsource t=0 v=-VDIG t=PRD/2 v=VDIG t=PRD v=-1 period=PRD
model VSW1 vswitch ron=10m roff=1M voff=-0.1*VDIG von =0.1*VDIG
model VSW2 vswitch ron=10m roff=1M von =-0.1*VDIG voff=0.1*VDIG
```

Figure 13. The schematic of the class-D audio amplifier.

power) is filtered by the LC filter made up of L_o and C_o. The values of these elements were chosen considering the impedance of the load (loudspeaker). The value of R_b was chosen equal to $10\,R_a$ in order to have an input/output gain of the CLASS-D amplifier close to $+45$ dB.

As for the previous circuit we are interested in the input/output small signal periodic transfer function of this CLASS-D amplifier when it is driven by a small signal superimposed on the large signal.

We first computed steady state solutions with the extended SH method for different frequencies of the input sinusoidal small signal. An item of this frequency grid is computed through the command

```
Sh shooting fund=FREQ restart=no solver=2
+    method=2 order=2 fft=yes fftharms=32
+    eabstol=10m
```

where `FREQ` is a parameter specifying the frequency of the input signal, `fft=yes` turns on Fast Fourier Transform to compute the spectrum of large signal periodic steady state solution with 32 harmonics. As in the previous case the shooting analysis is restarted from

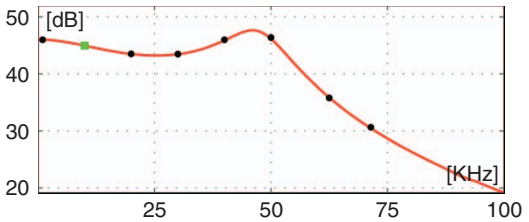
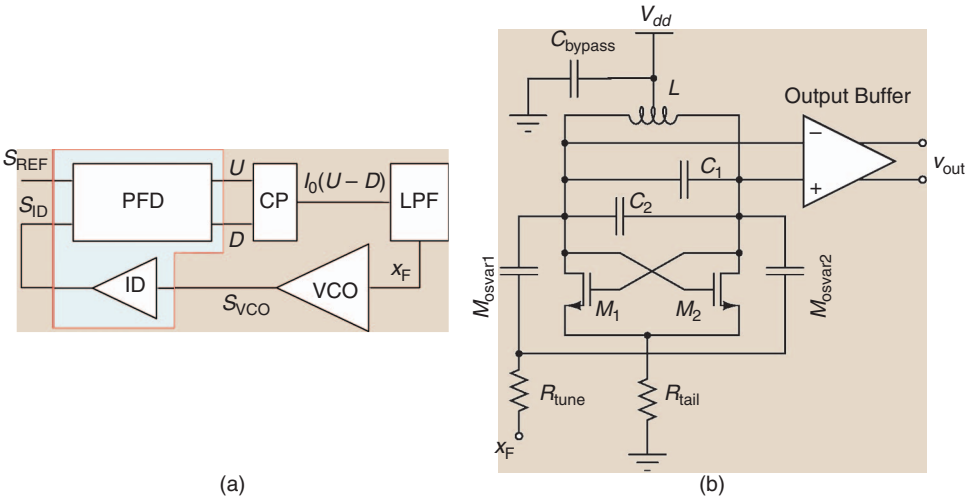


Figure 14. Red plot represents the input/output transfer function of the CLASS-D amplifier. It was computed through a PAC analysis when the CLASS-D amplifier is driven in input by a small signal. Black dots and the green square represent the result computed by the large signal SH analysis and refer to the magnitude of the component of the *Out* voltage at the frequency of the *In* signal computed by the Fast Fourier transform. In particular, the green square was obtained by setting FREQ = 10kHz (see the command line in the text).

previous results and these frequencies were chosen in order to have periodic solutions, i.e., the period of the *In* signal must be a multiple of the *Pwm* waveform (1 µs). We then performed a Fast Fourier analysis of the *Out* output waveforms; the obtained modulus of the component at the frequency of the *In* signal are reported in Fig. 13 (black dots). Each of these SH analyses is time consuming, a more efficient result can be obtained once more through the extended PAC analysis. Fig. 13 shows the obtained modulus of the input/output periodic transfer function. As it can be seen from Fig. 13 the modulus is "almost flat" till about 50 kHz then it exhibits a peak



(a) (b)

The VERILOG Code Implementing the Integer Divider (ID) and Phase/Frequency Detector (PDF) Blocks

```
`timescale 1ps/1ps
`delay_mode_path

module pd (up, down);
output up, down;
reg res_n, res_c, up, down, reset, ref, fb, fbx;
reg [15:0] Count;
initial begin
  reset = 0; up = 0; down = 0; res_c = 0; res_n = 0; fb = 0; fbx = 0;
end
always @(posedge fb or !res_c) begin
  if( !res_c ) Count = 0;
  else begin
    Count = Count + 1;
    if( Count >= 24 ) begin Count = 0; fbx = !fbx; end
  end
end
always @(posedge ref or posedge reset or !res_n) begin
  if (reset | !res_n) up = 0;
  else up = 1;
end
always @(posedge fbx or posedge reset or !res_n) begin
   if (reset | !res_n) down = 0;
   else down = 1;
end
always @(up or down) begin
  if (up & down)
    begin
      #2 reset = 1;
      #2 reset = 0;
    end
end
always @ (!res_n) reset = !res_n;
endmodule
```

Figure 15. The block schematic of the PLL (a). The simplified schematic of the VCO modeled through the TSMC 180nm PDK (b).

and then drops with a slope due to the output filter. Furthermore the results from the large signal SH analyses almost perfectly overlap the PAC result.

C. A PLL with a Real VCO

Phase Locked Loops (PLLs) are largely used in RF applications to generate, for instance, periodic signals with very good frequency stability. Oscillators with very good frequency stability can be obtained by exploiting crystals; unfortunately they can work only at a relatively low frequency (no larger than about 50 MHz). Oscillators operating in the GHz range with very good frequency stability can be obtained by "synchronizing" them to a reference crystal oscillator working at a lower frequency. The circuit doing this is referred to as PLL. A block schematic of a PLL is shown in Fig. 14 (left). The Voltage Controlled Oscillator (VCO) block implements the high frequency oscillator, whose frequency can be varied by acting on a voltage controlled terminal through x_F. The VCO s_{VCO} output signal drives a digital Integer Divider (ID) that generates a periodic waveform (typically a piecewise linear signal) at a frequency that is an integer fraction of the VCO one. The Phase/Frequency Detector (PDF) block compares the time instants at which a rising front of the s_{ID} waveform produced by ID occurs, with that of the s_{REF} waveform signal generated by the reference crystal oscillator. The PDF generates the U and D digital output signals that drive the Charge Pump (CP). The $I_o(U-D)$ output current is suitably integrated by the Low Pass Filter (LPF) block and the resulting x_F closes the loop. If the rising front of s_{ID} comes after that of s_{REF}, say with a delay Δt, the U signal is high in Δt and the current by CP increases the value of x_F thus "accelerating" the VCO. On the contrary, if the rising front of s_{REF} comes after that of s_{ID}, the D signal "decelerates" the VCO.

We simulated a PLL circuit by modeling the PDF and ID blocks as digital, the CP one as a behavioral analog block and the VCO at transistor level. The VERILOG code implementing the ID and PDF blocks is reported in the insert. The frequency of the s_{REF} reference signal was set at 50 MHz and ID divides by 48. When locked the PLL forces the VCO to work at 2400 MHz. A detailed model comprehending parasitics and bonding of the VCO based on the TSMC 180 nm PDK (Process Design Kit) was used [33]. Its simplified schematic is shown in Fig. 14 (right). Frequency is varied by acting on the nonlinear capacitance of the M osvar 1 and M osvar 2 capacitors. This VCO was realized on silicon; a microphotography is shown in Fig. 15. The complete model of the PLL is constituted of 172 capacitors, 220 resistors, 63 inductors, 1 coupled inductor, 50 diodes and 12 BSIM3 MOSFETS. We applied the proposed approach to compute the total noise at the output of the VCO when the PLL is locked. Noise is an important figure of merit for designers since noisy oscillators can interfere with adjacent communication channels and compromise signal quality.

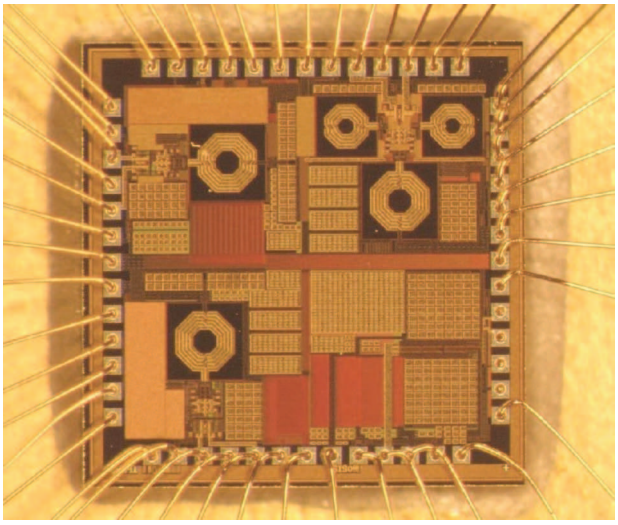


Figure 16. The silicon implementation of the VCO. The same layout implements different versions of VCOs that can be identified by the spiral inductors that occupy a large portion of the chip area. That used in the PLL is the lower left one.

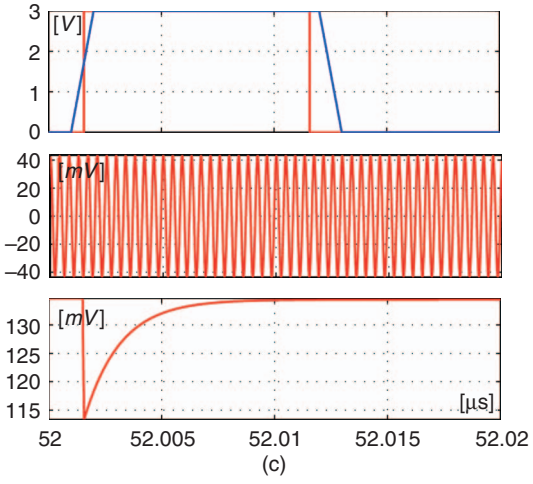


Figure 17. From upper panel to lower panel: the waveforms at the output of ID (s_{ID} in red) and the s_{REF} reference signal (in blue); the s_{VCO} VCO output waveform; the x_F signal.

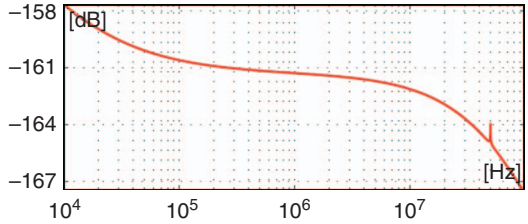


Figure 18. The PSD of total noise in dB related to the right lateral band from the fundamental of the VCO. x-axis shows the displacement from the fundamental at 2400 MHz.

With the current trend towards very large scale circuits and systems that include analog, digital, and behavioral parts, it is necessary to develop new tools and methods for the design of such systems.

We first performed a quite long time domain analysis to bring the PLL to work sufficiently close to the steady state condition. We then performed a SH analysis to determine the periodic orbit as in the previous examples and finally two PNOISE analyses

```
Tr tran tstop=TSTOP tmax=0.1/FVCO tmin=100f
+           cmin=no method=2 order=2
Sh shooting restart=no fund=FREF solver=2
+           floquet=yes method=2 order=2
+           eabstol=100m save="Sh.save"
Pn1 pnoise start=10k stop=100M dec=200
+           onodes=["out1a","out2a"]
Pn2 pnoise start=40M stop=60M lin=1k
+           onodes=["out1a","out2a"]
```

With respect to the other similar analysis cards shown before, in the `shooting` case results are saved in the `"Sh.save"` file since they can be used as initial guess in possible other shooting analyses, saving, in this way, the start-up time consumed by the `tran` analysis. Simulations were performed on a computer equipped with an Intel-i5@3.10GHz CPU, 8Gbyte. The start-up phase took 3400 s, the SH analysis took 83 s and the PNOISE analysis took 30 s per frequency point.

Some results computed by the SH analysis are shown in Fig. 17 and the PSD of noise at the output of the VCO is shown in Fig. 18. By observing Fig. 17 it can be seen that in a working period the VCO output waveform does 48 oscillations and that the two rising fronts of the ID and s_{REF} waveforms are "synchronised". Note that these two waveforms have different duty cycles. By observing Fig. 17 we see an almost flat PSD in the PLL loop bandwidth; the LPF stabilizes the PLL by inserting a zero/pole pair. The zero is at 100 kHz and the pole at 10 MHz, the loop gain equals 1 at 1 MHz. The spike at 50 MHz is due to the switching nature of the PLL at the frequency of the s_{REF} reference signal.

VI. Conclusions

With the current trend towards very large scale circuits and systems that include analog, digital, and behavioral parts, it is necessary to develop new tools and methods for the design of such systems. This new class of analysis and design instruments should be able to locate the orbits, calculate their stability, perform periodic noise analyses, small signal analysis, etc. So far such a tool was not available due to the theoretical difficulties in handling such systems. In recent times these issues have been addressed theoretically, setting the stage for the development of the necessary computational tools.

In this article we have reported the development of the first such design tool, called PAN. We have illustrated its usefulness using a digitally controlled DC/DC converter, a class-D audio amplifier, a phase locked loop with a voltage controlled oscillator. We invite the circuits and systems designers to try out the program on various systems that fit the above description, and to send us feedbacks which will enable us to improve the capabilities and user-interface further. We sincerely hope this development will open new vistas in the design and fabrication of complex electronic systems.

References

[1] S. Sawant, U. Desai, G. Shamanna, L. Sharma, M. Ranade, A. Agarwal, S. Dakshinamurthy, and R. Narayanan, "A 32nm westmere-ex xeon enterprise processor," in *Proc. IEEE Int. Solid-State Circuits Conf. Dig. Technical Papers (ISSCC)*, 2011, pp. 74–75.

[2] G. E. Moore, "Cramming more components onto integrated circuits," *Proc. IEEE*, vol. 86, no. 1, pp. 82–85, 1998.

[3] "Intel® Xeon® Processor E5-1600/ E5-2600/E5-4600 product families," Intel Corporation, Reference Number: 326508, Revision: 002, vol. 1, May 2012.

[4] "Intel® Xeon® Processor E5-1600/ E5-2600/E5-4600 product families," Intel corporation, Reference Number: 326509, Revision: 003, vol. 2, May 2012.

[5] E. Bolotin, I. Cidon, R. Ginosar, and A. Kolodny, "Cost considerations in network on chip," *VLSI J. Integr.*, vol. 38, no. 1, pp. 19–42, 2004.

[6] A. Agarwal, C. Iskander, and R. Shankar, "Survey of network on chip (noc) architectures & contributions," *J. Eng., Comput. Architect.*, vol. 3, no. 1, pp. 21–27, 2009.

[7] S. Pasricha and N. Dutt, *On-Chip Communication Architectures: System on Chip Interconnect*. Burlington, MA: Morgan Kaufmann, 2010.

[8] L.-T. Wang, C. E. Stroud, and N. A. Touba, *System-on-Chip Test Architectures: Nanometer Design for Testability*. Burlington, MA: Morgan Kaufmann, 2010.

[9] G. Martin, B. Bailey, and A. Piziali, *ESL Design and Verification: A Prescription for Electronic System Level Methodology*. Burlington, MA: Morgan Kaufmann, 2010.

[10] R. Marculescu, U. Ogras, L.-S. Peh, N. Jerger, and Y. Hoskote, "Outstanding research problems in NOC design: System, microarchitecture, and circuit perspectives," *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, vol. 28, no. 1, pp. 3–21, 2009.

[11] D. Leenaerts, G. Gielen, and R. A. Rutenbar, "Cad solutions and outstanding challenges for mixed-signal and RFIC design," in *Proc. 2001 IEEE/ACM Int. Conf. Computer-aided Design*, ser. ICCAD'01. Piscataway, NJ: IEEE Press, 2001, pp. 270–277.

[12] A. Sangiovanni-Vincentelli. (2002). Defining platform-based design. [Online]. Available: http://www.eetimes.com/document.asp?doc˙id=1204965

[13] A. L. Sangiovanni-Vincentelli and R. Passerone, *Platform-Based design*. New York: Springer, 2012.

[14] J. Vlach and K. Singhal, *Computer Methods for Circuit Analysis and Design*. New York: Van Nostrand, 1983.

[15] R. Saleh, "Iterated timing analysis and splice1," EECS Department, Univ. of California, Berkeley, Tech. Rep. UCB/ERL M84/2, 1984.

[16] A. Newton, "The simulation of large-scale integrated circuits," EECS Dept., Univ. of California, Berkeley, Tech. Rep. UCB/ERL M78/52, 1978.

[17] K. Kundert, *The Designer's Guide to Spice and Spectre*. New York: Kluwer Academic, 1995.

[18] K. Kundert and A. Sangiovanni-Vincentelli, "Simulation of nonlinear circuits in the frequency domain," *IEEE Trans. Comput. Aided Des.*, vol. CAD-5, no. 4, pp. 521–535, Oct. 1986.

[19] J. White and A. Sangiovanni-Vincentelli, *Relaxation Techniques for the Simulation of VLSI Circuits,* vol. 20. New York, NY: Kluwer Academic, 1987.

[20] P. Antognetti and G. Massobrio, *Semiconductor Device Modeling with SPICE*, P. Antognetti, Ed. New York: McGraw-Hill, 1988.

[21] L. Nagel, "Spice2: A computer program to simulate semiconductor circuits," EECS Department Univ. of California, Berkeley, Tech. Rep. UCB/ERL M520, 1975.

[22] J. Attia, *PSPICE and MATLAB for Electronics: An Integrated Approach*. Boca Raton, FL: CRC Press, 2010.

[23] J. Stoer, R. Bulirsch, R. Bartels, W. Gautschi, and C. Witzgall, *Introduction to Numerical Analysis*, vol. 2. New York: Springer, 1993.

[24] A. Newton, "Techniques for the simulation of large-scale integrated circuits," *IEEE Trans. Circuits Syst.,* vol. 26, no. 9, pp. 741–749, 1979.

[25] M. Bohr, "The new era of scaling in an soc world," in *Proc. IEEE Int. Solid-State Circuits Conf. Dig. Technical Papers (ISSC),* 2009, pp. 23–28.

[26] A. Newton, "Timing, logic and mixed-mode simulation for large MOS integrated circuits," *Comput. Des. Aids VLSI Circuits*, 1981.

[27] R. J. Baker, CMOS: Circuit Design, *Layout, and Simulation*, vol. 18. Wiley-IEEE Press, 2011.
[28] R. A. Saleh and A. R. Newton, *Mixed-Mode Simulation*. Norwell, MA: Kluwer Academic, 1990.
[29] M. Demler. (2011). Price for a new SATA I/O [dollar] 700M. A complete AMS verification? Priceless! [Online]. Available: http://www.edn.com/electronics-blogs/other/4311208
[30] M. Demler. (2011). Complete ic simulation requires a full toolbox of hardware and software. [Online]. Available: http://www.edn.com/design/integrated-circuit-design/4369582
[31] R. Telichevesky, K. Kundert, I. Elfadel, and J. White, "Fast simulation algorithms for RF circuits," in *Proc. IEEE Custom Integrated Circuits Conf.,* 1996, pp. 437–444.
[32] R. Telichevesky, K. S. Kundert, and J. K. White, "Efficient steady-state analysis based on matrix-free Krylov-subspace methods," in *Proc. 32nd Annu. ACM/IEEE Design Automation Conf.,* 1995, pp. 480–484.
[33] A. Brambilla, G. Gruosso, M. Redaelli, G. Gajani, and D. Caviglia, "Improved small-signal analysis for circuits working in periodic steady state," *IEEE Trans. Circuits Syst. I: Reg. Pap*., vol. 57, no. 2, pp. 427–437, Feb. 2010.
[34] A. Brambilla, G. Gruosso, and G. Storti Gajani, "Determination of floquet exponents for small-signal analysis of nonlinear periodic circuits," *IEEE Trans. Comput.-Aid. Des. Integr. Circuits Syst*., vol. 28, no. 3, pp. 447–451, 2009.
[35] A. Brambilla, G. Gruosso, and G. Storti Gajani, "FSSA: Fast steady-state algorithm for the analysis of mixed analog/digital circuits," *IEEE Trans. Comput.-Aid. Des. Integr. Circuits Syst.,* vol. 29, no. 4, pp. 528–537, Apr. 2010.
[36] A. Brambilla, G. Gruosso, and G. Storti Gajani, "Robust harmonic-probe method for the simulation of oscillators," *IEEE Trans. Circuits Syst. I: Reg. Pap*., vol. 57, no. 9, pp. 2531–2541, 2010.
[37] A. Brambilla, G. Gruosso, A. M. Redaelli, and G. Storti Gajani, "Periodic noise analysis of electric circuits: Artifacts, singularities and a numerical method," *J. Circuit Theory Applicat.,* to be published.
[38] I. Hiskens and M. Pai, "Trajectory sensitivity analysis of hybrid systems," *IEEE Trans. Circuits Syst. I: Fundam. Theory Applicat*., vol. 47, no. 2, pp. 204–220, 2000.
[39] V. Donde and I. Hiskens, "Shooting methods for locating grazing phenomena in hybrid systems," *Int. J. Bifurc. Chaos*, vol. 16, no. 3, pp. 671–692, 2006.
[40] D. Giaouris, S. Banerjee, B. Zahawi, and V. Pickert, "Stability analysis of the continuous-conduction-mode buck converter via Filippov's method," *IEEE Trans. Circuits Syst. I: Reg. Pap*., vol. 55, no. 4, pp. 1084–1096, 2008.
[41] D. Giaouris, S. Maity, S. Banerjee, V. Pickert, and B. Zahawi, "Application of Filippov method for the analysis of subharmonic instability in dc-dc converters," *Int. J. Circuit Theory Applicat.*, vol. 37, no. 8, pp. 899–919, 2009.
[42] K. Mandal, S. Banerjee, and C. Chakraborty, "Symmetry-breaking bifurcation in series-parallel load resonant dc-dc converters," *IEEE Trans. Circuits Syst. I: Reg. Pap*., vol. 60, no. 3, pp. 778–787, 2013.
[43] F. Bizzarri, A. Brambilla, S. Perticaroli, and G. Storti Gajani, "Noise in a phase-quadrature pulsed energy restore oscillator," in *Proc. 20th European Conf. Circuit Theory Design (ECCTD)*, 2011, pp. 465–468.
[44] F. Bizzarri and X. Wei, "Phase noise analysis of a mechanical autonomous impact oscillator with a MEMS resonator," in *Proc. 20th European Conf. Circuit Theory Design (ECCTD)*, 2011, pp. 729–732.
[45] F. Bizzarri, A. Brambilla, and G. Storti Gajani, "Phase noise simulation in analog mixed signal circuits: An application to pulse energy oscillators," *IEEE Trans. Circuits Syst. II: Exp. Briefs*, vol. 58, no. 3, pp. 154–158, 2011.
[46] F. Bizzarri, A. Brambilla, and G. Storti Gajani, "Steady state computation and noise analysis of analog mixed signal circuits," *IEEE Trans. Circuits Syst. I: Reg. Pap*., vol. 59, no. 3, pp. 541–554, 2012.
[47] F. Bizzarri, A. Brambilla, and G. Storti Gajani, "Extension of the variational equation to analog/digital circuits: numerical and experimental validation," *Int. J. Circuit Theory Applicat.*, vol. 41, no. 7, pp. 743–752, 2013.
[48] F. Bizzarri, A. Brambilla, G. Gruosso, and G. Storti Gajani, "Steady state simulation of mixed analog/digital circuits," in *Integrated Circuits for Analog Signal Processing*, E. Tlelo-Cuautle, Ed. New York: Springer, 2013, pp. 243–270.
[49] F. Bizzarri, A. Brambilla, S. Saggini, and G. Storti-Gajani, "Mixed-mode simulations to check stability of an adaptive constant on-time dc-dc converter," in *Proc. European Conf. Circuit Theory and Design (ECCTD)*, 2013, pp. 1–4.
[50] F. Bizzarri, A. Brambilla, and S. Saggini, "Voltage regulators design through advanced mixed-mode circuit simulation," *IEEE Trans. Power Electron.*, vol. 29, no. 9, pp. 4496–4499, Sept. 2014.
[51] M. Biggio, F. Bizzarri, A. Brambilla, and M. Storace, "Efficient transient noise analysis of non-periodic mixed analog/digital circuits," *IET Circuits, Devices Syst.* to be published.
[52] F. Bizzarri and A. Brambilla, "Stability analysis of voltage regulators versus different digital control strategies by analog-mixed-signal circuit simulation," in *Proc. Int. Symp. Power Electronics, Electrical Drives, Automation and Motion (SPEEDAM)*, May 2014, pp. 1049–1053.
[53] M. Biggio, F. Bizzarri, A. Brambilla, and M. Storace, "Accurate and efficient PSD computation in mixed-signal circuits: a time domain approach," *IEEE Trans. Circuits Syst. II: Exp. Briefs,* to be published.
[54] L. Chua, C. A. Desoer, and E. S. Kuh, *Linear and Nonlinear Circuits*. New York: McGraw-Hill, 1987.
[55] M. A. Aizerman and F. R. Gantmakher, "On the stability of periodic motions," *J. Appl. Math. Mech.* (translated from Russian), pp. 1065–1078, 1958.
[56] A. F. Filippov, "Differential equations with discontinuous right-hand side," *Amer. Math. Soc. Transl.*, vol. 42, no. 2, pp. 199–231, 1978.
[57] R. I. Leine and H. Nijmeijer, *Dynamics and Bifurcations in Non-Smooth Mechanical Systems*. Berlin: Springer-Verlag, 2004.
[58] M. Farkas, *Periodic Motions*. New York, NY: Springer-Verlag, 1994.
[59] M. Di Bernardo, C. Budd, A. Champneys, and P. Kowalczyk, *Piecewise-Smooth Dynamical Systems, Theory and Applications*. London: Springer-Verlag, 2008.
[60] T. Aprille and T. Trick, "A computer algorithm to determine the steady-state response of nonlinear oscillators," *IEEE Trans. Circuit Theory*, vol. 19, no. 4, pp. 354–360, July 1972.
[61] J. Aprille and T. Trick, "Steady-state analysis of nonlinear circuits with periodic inputs," *Proc. IEEE*, vol. 60, no. 1, pp. 108–114, Jan. 1972.
[62] Y. A. Kuznetsov, *Elements of Applied Bifurcation Theory*, 3rd ed. New York: Springer-Verlag, 2004.
[63] M. Okumura, T. Sugawara, and H. Tanimoto, "An efficient small signal frequency analysis method of nonlinear circuits with two frequency excitations," *IEEE Trans. Comput.-Aid. Des. Integr. Circuits Syst.*, vol. 9, pp. 225–235, Mar. 1990.
[64] R. Telichevesky, K. Kundert, and J. White, "Receiver characterization using periodic small-signal analysis," in *Proc. IEEE Custom Integrated Circuits Conf. 1996,* May 1996, pp. 449–452.
[65] R. Telichevesky, K. Kundert, and W. J., "Efficient ac and noise analysis of two-tone rf circuits," in *Proc. DAC*, 1996, pp. 292–297.
[66] A. Demir, "Phase noise in oscillators: Daes and colored noise sources," in *Proc. IEEE/ACM Int. Conf. Computer-Aided Design (ICCAD 98), Dig. Technical Papers*, 8–12, 1998, pp. 170–177.
[67] A. Demir and J. Roychowdhury, "Phase noise in oscillators: A unified theory and numerical methods for characterization," *IEEE Trans. Circuits Syst. I: Fundam. Theory Applicat.*, vol. 47, no. 5, pp. 655–674, 2000.
[68] A. Demir and J. Roychowdhury, "A reliable and efficient procedure for oscillator ppv computation, with phase noise macromodeling applications," *IEEE Trans. Comput.-Aid. Des. Integr. Circuits Syst.*, vol. 22, pp. 188–197, Feb. 2003.
[69] L. Dieci, R. D. Russell, and E. S. V. Vleck, "On the computation of Lyapunov exponents for continuous dynamical systems," *SIAM J. Numer. Anal.*, vol. 34, no. 1, pp. 402–423, 1997.
[70] P. C. Müller, "Calculation of Lyapunov exponents for dynamic systems with discontinuities," *Chaos, Solitons Fractals*, vol. 5, no. 9, pp. 1671–1681, 1995.
[71] S. L. de Souza and I. L. Caldas, "Calculation of Lyapunov exponents in systems with impacts," *Chaos, Solitons Fractals*, vol. 19, no. 3, pp. 569–579, 2004.
[72] D. Zhou, Y. Sun, A. Rangan, and D. Cai, "Spectrum of Lyapunov exponents of non-smooth dynamical systems of integrate-and-fire type," *J. Computat. Neurosci.*, vol. 28, pp. 229–245, 2010.
[73] F. Bizzarri, A. Brambilla, and G. Storti Gajani, "Lyapunov exponents computation for hybrid neurons," *J. Computat. Neurosci.*, vol. 35, no. 2, pp. 201–212, 2013.
[74] D. Giaouris, S. Banerjee, F. Stergiopoulos, S. Papadopoulou, S. Voutetakis, B. Zahawi, A. V. Pickert, A. Abusorrah, M. Hindawi, and Y. Al-Turki, "Foldings and grazings of tori in current controlled interleaved boost converters," *Int. J. Circuit Theory Applicat.,* 2013.