A Parallel Approach to Real-Time Simulation of Power Electronics Systems

Andrea Benigni, *Member, IEEE*, and Antonello Monti, *Senior Member, IEEE*

***Abstract*—We define here a new parallel simulation method designed for real-time execution. This method is highly parallelizable and scalable, and the simulation execution time is fully predictable, which is very important for real-time execution. The stability conditions for the method are defined. Furthermore, we show how to define appropriate flow variable injections so that this method can be extended to multiphysic applications and how to implement multirate time step in the defined method. Finally, we present simulation tests that validate the presented method.**

***Index Terms*—Circuit simulation, parallel algorithms, power electronics, power system simulation, real-time systems.**

I. Introduction

In the development cycle of any industrial product, prototyping and testing are known to be some of the most expensive and risky steps. In some cases, the risks and cost can become prohibitively high. Simulation is a very powerful tool that can significantly reduce both cost and risks: real-time (RT) simulation combined with hardware in the loop (HIL) or power HIL (PHIL) testing can drastically improve the power of simulation, filling the gap between field test and simulation.

Historically, one of the main interests for RT simulation in the electrical engineering field was the testing of relays for terrestrial power system as, for example, in [1]. In [2], an RT, resistive companion type of solver is presented; in [3], applying the Multi Area Thévenin Equivalent (MATE) [4] concept, a 78-node power system is executed in RT on a PC cluster. In the same period, the early 90s, also RTDS—one of the most recognized commercial tool for power system RT simulation—was developed [1], [5].

Starting in the same years, but with a significant growth of interest in recent years, RT simulation [6], [7], [8] and HIL methods [9], [10], [11] have attracted the interests of both academia and industry for power electronic systems as well.

The main advantage of RT simulation and HIL techniques over traditional simulation is that they allow designers to perform experiments where part of the system is emulated on computational units (simulated) while part of the system is physically installed in the laboratory. [12]–[14] are some examples of how HIL is used to support a heterogeneous set of research questions. The main advantage of HIL over traditional laboratory tests is its closeness to real-life scenarios.

The strong nonlinear behavior of the system and the small time step size required by the high switching frequencies are between the main challenges of RT simulation of power electronics converters. In the past few years, to face these challenges, there has been a change also in the type of processors used; mixed solutions based on DSPs/CPU and FPGA are more and more common. FPGAs are used both as interface [15] and also for computation: in [16]–[18], an ac machine, a power converter, and a nonlinear power transformer are directly simulated on the FPGA. In [19], an MMC converter is simulated using an FPGA in combination with a CPU.

With the growing role of power electronic converters in power systems, especially in relation to distributed energy sources, RT simulation for smart-grid applications became an hot topic [20], with particular interest in simulating grid-connected power converters [13], [21], [22]; the work presented in [22] is based on another very popular commercial tool: Opal-RT.

The goal of the work presented in this paper is the development of a multiphysic simulation method for RT simulation of power systems with high penetration of power electronics converters. Naturally, the solution method should be highly parallelizable, computationally as light as possible, and satisfy the strict requirement of RT execution without any risk of overrun. Another important requirement for the simulation method is that parallelization should be completely automated so not to require any specific action by the user. The simulation method should also allow for multirating execution. Although no specific hardware platform has been targeted, in order to test the method described in this paper a DSP and a x86-based platform have been used.

At the same time, the application considered presents some peculiarities that have to be taken in consideration in the development of this new method. Power electronics systems simulation, if switching models are considered, is well known to require small time step size as consequence of the relative high switching frequency used. If this requirement on one side represents a significant challenge for RT execution, on the other side, the small time step significantly increase the latency effect of the circuit reactive elements. Thus, the main idea of this new method is to exploit the small time step required by the target application to explicitly integrate part of the simulated system. In this way, we are able to decouple the integration of individual components from the one of the system, significantly increasing the possibility to parallelize.

It is worth to specify that unlike transient stability simulation—that is the solution of a system of differential

Manuscript received June 19, 2014; revised September 8, 2014; accepted September 22, 2014. Date of publication October 8, 2014; date of current version April 15, 2015. Recommended for publication by Associate Editor Prof. S. K. Mazumder.

A. Benigni is with the Department of Electrical Engineering, University of South Carolina, Columbia, SC 29201 USA (e-mail: benignia@cec.sc.edu).

A. Monti is with the E.ON Energy Research Center, Rhenish-Westphalian Technical Aachen-University, 52074 Aachen, Germany (e-mail: AMonti@eonerc.rwth-aachen.de).

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/TPEL.2014.2361868

algebraic equations (DAEs)—our method targets electromagnetic transient simulation and thus the solution of a set of ordinary differential equations (ODEs). The parallelization of transient stability simulation solvers has been widely investigated in the last 20 years: in [23]–[25], several theoretical approaches have been proposed; in [26] and [27], the use of a GPUs and of a cluster system have been investigated. In recent years, due to the increasing interest—among others also of the US Department of Energy—in dynamic security assessment, faster than RT transient stability simulation [28] based on parallelization methods and on the use of high-performance computing techniques [29], [30] have received more and more attention.

The main difference between the parallelization of transient stability simulation and the parallelization of electromagnetic transient simulation is in the quasi-static assumption for the interconnections that is done in transient stability simulation. As consequence the dynamic differential equations describing the system are decoupled and thus their solution can be parallelized.

Another important aspect to be considered is that transient stability simulation typically targets a time step size varying from tens of millisecond to hundreds of millisecond, while electromagnetic transient simulation targets a time step size varying from tens of microsecond to hundreds of nanosecond; this creates different requirements and possibilities for the selection of the integration methods. To conclude, the main goal of faster than RT simulation is to calculate as quickly as possible the solution for a defined simulation interval. Iterative methods and time step stacking are solutions that fit very well this application but they can hardly be applied to RT simulation: in this case, the hard RT requirement has to be respected at each simulation instant and a predictable execution time is required.

II. Approach

To satisfy the requirements listed in Section I, we define a simulation method that supports automatic parallelization of circuit simulation for RT execution. We name this method latency based linear multistep compound (LB-LMC) method.

The main idea of this method is to exploit latency at individual component level, isolating their solution from the system solution. For this purpose, for example, the following can be exploited: induction machine stator inductance and rotor inertia, snubber circuits of power electronics switching devices, and input and output power converter filters. From now on, we will refer to "components solution" for the solution of components like converter, machine, as well as individual devices, like diode, transistor, etc., and to "network solution" for the solution of the interconnection between the components. Latency is exploited explicitly integrating the components solution.

The system to be simulated is modeled using a traditional resistive companion approach, but the nonlinear (including time variant) components are identified and substituted with an equivalent current or voltage source (their dynamics are described by a separate set of state equations, which are discretized using an explicit integration method). Since the obtained resistive companion model is linear and time invariant, then the LU factorization of the conductance matrix can be calculated off-line. At each time step, the state equations of the nonlinear components are solved in parallel, on the basis of the previous network solution; the source vector of the resistive companion network formulation is updated with the newly calculated values of currents and voltages. Finally, the network solution is computed by forward and backward substitution using the updated source vector.

Since the part of the system solved with resistive companion is linear, and the nonlinear parts are solved with state equation integrated explicitly, no iterations are required during the same time step and so the execution time can be fully predicted. Partitioning at single-component level offers a higher level of parallelization compared to the partitioning in zones, typical of other simulation methods given in [4] and [31]–[33]; moreover, parallelization is fully automatized and no action is required to the simulator user.

Due to the RT execution target of the presented method, it is important to make a clear distinction between off-line and online tasks.

The off-line step can be summarized as follows: describe the network; identify and substitute the nonlinear components in the network with current or voltage sources (dynamics are described by the state equations embedded in the model of the nonlinear component); transform the now linear network into its associated resistive companion representation; and factorize and store the obtained conductance matrix.

During the online phase, the differential equations of the nonlinear components are solved in parallel on the base of the previous network solution performed with the resistive companion. The source vector of the resistive companion is updated with the contribution of the nonlinear components and of the linear dynamic components solved with resistive companion, and forward and backward substitutions are performed using the conductance matrix that is previously factorized and stored.

It is important to underline that the proposed approach fits both proper nonlinear components (e.g., machines and diodes modeled according to Shockley equation) as well as systems with many different topologies based on the state of the switches (e.g., power converter modeled using ideal switching devices). In this last case, the proposed method, decoupling the single components solution from the network solution (representing each converter as a set of voltage and current sources), limits the number of state matrices to be precalculated that scales linearly with the number of converters involved.

If the power system to be simulated is really large, the forward and backward substitution could still be too computationally intensive to fulfill the requirement of RT execution. One important consideration here is that, due to the modeling approach used for the nonlinear components, the conductance matrix is block diagonal, as consequence, the factorization and the forward and backward substitution of every single block can be done autonomously, gaining another level of parallelization.

Before proceeding with the mathematical formalization of the presented method, it is worth spending few words on the method presented in [34], which present significant similarity with the method presented in this paper. In [34], the authors

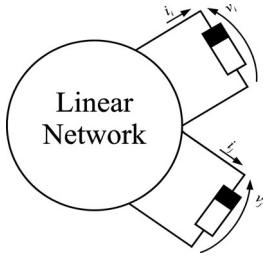

Fig. 1. Linear network with two nonlinear components.

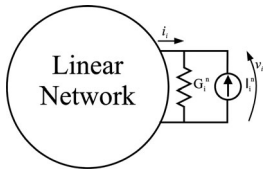

Fig. 2. Linear network with one current-type nonlinear component.

propose a new solution method that combines nodal method with state space. The two main advantages of such a method are the possibility of benefiting from the advantages of the most suitable method, which can so be freely selected, and the possibility of parallelizing execution. One of the main motivations for this paper seems to be the possibility of maintaining the solution of switching devices independent. This is extremely relevant since a solution flow that considers precalculation of state matrix has been adopted: partitioning the system and keeping the switching devices separate reduce the combinations and the number of matrices to be recalculated and stored. At the same time since the whole system is implicitly integrated - and thus the LU factorization and forward and backward substitutions have still to be performed on line - overruns are not completely ruled out.

III. Latency-Based LMC Method Solution Flow

In this section, the solution process of the LB-LMC method is illustrated. It is shown how the solution of nonlinear components is isolated while still keeping a resistive companion framework; how each nonlinear component is solved independently using explicitly integrated state equation; and how the component results are combined by the resistive companion solver.

From now on, we will use the term "internal step" to refer to the execution of one time step of the state equation describing the nonlinear components. With the term "network solution," we will refer to the execution of one time step of the resistive companion solver. Let us now formalize the different steps of the presented method.

Let us start by considering a circuit like the one shown in Fig. 1, composed of linear components only, except for components *i* and *j*. In Fig. 2(a), a nonlinear component, represented using an ideal current source with a conductance in parallel, is connected to the linear network. The conductance G_i^n is inserted to keep the standard form of resistive companion components; numerically, since we use an explicit integration method for the nonlinear components, G_i^n will always be equal to zero. First, assumption for applying the presented method is that every nonlinear component can be expressed as in (1) or (2). From now on, we will refer to components like (1) as current-type components and to components like (2) as voltage-type components. In general, multiterminal components can be described as a mix of current-type and voltage-type terminal

$$\frac{di_i^n}{dt} = f(v, i, x_i^n, u_i^n, t) \tag{1}$$

$$\frac{dv_j^n}{dt} = f(v, i, x_j^n, u_j^n, t) \tag{2}$$

where v is the vector of the network node voltages, i the vector of the network branch currents, x_i^n the vector of the state variable internal to the *i*th nonlinear component, and u_i is the vector of the input internal to the *i*th the nonlinear component.

Equations (1) and (2) are explicit discretized so that the following equations are obtained:

$$I_i^n(k+1) = f(v(k), i(k), x_i^n(k), u_i^n(k), k) \tag{3}$$

$$V_j^n(k+1) = f(v(k), i(k), x_j^n(k), u_j^n(k), k). \tag{4}$$

Assuming more nonlinear components in the circuit and grouping them according to their types (voltage or current), the following equations are obtained:

$$I^n(k+1) = f(v(k), i(k), x(k), u(k), k) \tag{5}$$

$$V^n(k+1) = f(v(k), i(k), x(k), u(k), k). \tag{6}$$

Equations (5) and (6) represent the internal step for all nonlinear components. Since the calculation of new values of voltage and current to update the relative sources only require the knowledge of the network solution at previous time step, the internal step can be easily parallelized simply broadcasting vector v and i to the designated kernels. Since the equations describing each nonlinear component are independent from one another, each kernel will solve the internal step only for a subset of the nonlinear components described by (5) and (6). The size of the subset to be solved by each kernel has to be determined on the base of the computational cost of each component and of the imposed time step size (if RT simulation is the target). With the updated values $I^n(k+1)$ and $V^n(k+1)$, a new network solution can be computed solving the nodal equation given by

$$Gx(k+1) = b(v(k), i(k), I^n(k), V^n(k), k). \tag{7}$$

Since the conductance matrix *G* is never updated, the LU factorization, as in linear resistive companion, can be performed off-line and only the forward and backward substitutions are executed online at each iteration.

In Fig. 3, the flow diagram of the solution process of the presented LB-LMC method is compared to the one of traditional nonlinear resistive companion. The flow diagram shows that the execution of the nonlinear components internal step can be parallelized. Comparing the two diagrams it is clear that the iteration needed to simulate nonlinear components in traditional resistive companion is avoided in the presented LB-LMC method. This leads to fully predictable execution time of the simulation.

Analyzing the diagram of Fig. 3 more in detail, we see that before entering the simulation loop, the first step is to initialize the

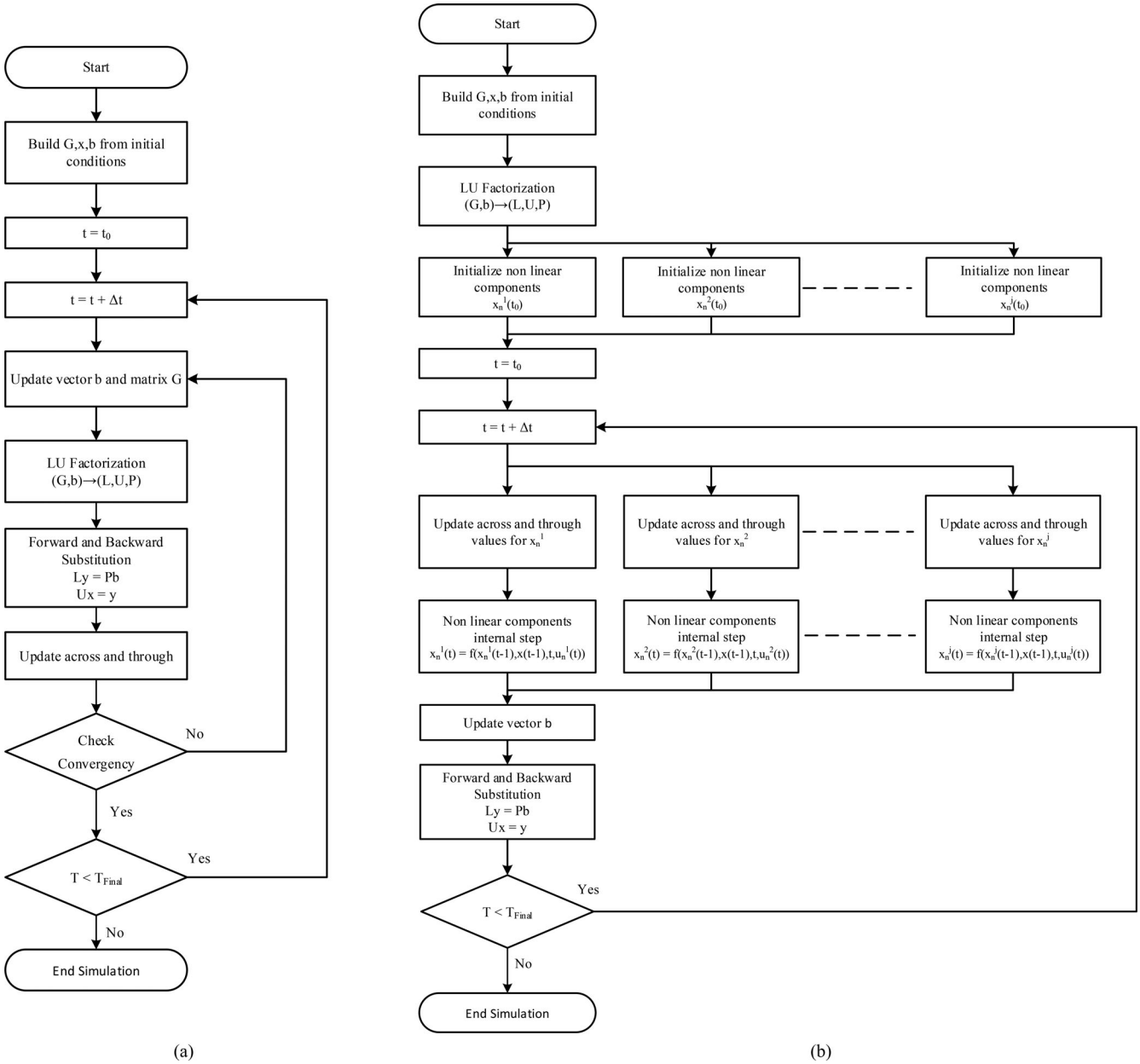

Fig. 3. (a) Traditional nonlinear resistive companion solution flow. (b) LB-LMC method solution flow.

nodal equation in (7) as well as the nonlinear components state equations. LU factorization is then performed and the obtained matrices $\{L, U, P\}$ are stored. Within the simulation loop, the internal step of the nonlinear components [see (5) and (6)] is performed after the update of the across and through quantities using the previous time step network solution; the source vector is then updated with the newly computed values and afterward, exactly like in the linear resistive companion case, forward and backward substitutions are executed.

IV. Stability Analysis

We discuss here the stability of the presented method. For the purpose of the stability analysis, we assume that both the nonlinear components and the network are modeled in a state-space framework, contrary to what the presented method defines: the network should be modeled using resistive companion. The analysis remains valid under the assumption that any LTI system can always be described as in (8) even if the system, like in this case, is actually solved using resistive companion. In other words, a state-space model can always be created, using the same discretization algorithm, with the purpose of studying the stability of the considered simulation scenario, even if we don't actually use the state-space model to compute the network solution. In [35], a method to obtain a state-space representation of a circuit starting from the nodal conductance matrix is presented.

The state-space model of the network is formulated in (8). We assume that the network equations have already been

discretized

$$x(k+1) = Ax(k) + B'u'(k+1) \tag{8}$$

where $u'(k+1)$ is the input vector and can be separated in two parts: the vector of input sources originally connected to the network $u(k)$ and a vector of new input sources $\varphi(k)$ that represents the components solved using state equations

$$u'(k) = \begin{bmatrix} u(k) \\ \varphi(k) \end{bmatrix}. \tag{9}$$

Inserting (9) in (8) and expanding, we obtain

$$x(k+1) = Ax(k) + Bu(k+1) + B_\varphi \varphi(k+1). \tag{10}$$

An output equation has to be written for the dual quantity of nonlinear component, that is, an output equation that returns the voltage at the terminals of the component has to be written for each current-type component. And vice versa, an output equation that returns the currents at the terminals of the each voltage-type component has to be written. These equations look like

$$z(k+1) = Cx(k+1) + Du(k+1) + D_\varphi \varphi(k+1). \tag{11}$$

In contrast to what have presented until now, here it is assumed that the components isolated from the network are also linear and can be expressed in the form of (12). This is done because the main purpose of the analysis presented in this section is to provide a tool to easily analyze how the selection of implicit and explicit integration method affect the stability of the solution and that could be used in the future to identify specific requirements for the components development. A stability study that also considers the nonlinear case was presented in [36]; the limit of this study is that it allows only defining sufficient but not necessary conditions for the stability

$$x_n(k+1) = A_n x_n(k) + B' u'_n(k). \tag{12}$$

Also in this case, the input vector has to be separated into two parts: the vector of input sources internally connected to the network $u_n(k)$ and the new input vector $z(k)$. It is important to point out that since the components are assumed to be solved with an explicit integration method, inputs are from the previous time step. Actually, for u_n, the use of the value at $(k+1)$ would not make any difference in the derivation of the following equations. Vice versa, it is extremely important to use the value of z at (k). In this way, as said previously, the solution of the components can be computed in a single step, without iterations, starting from the network solution computed at the previous time step

$$u'_n(k) = \begin{bmatrix} u_n(k) \\ z(k) \end{bmatrix}. \tag{13}$$

Substituting (13) into (12), the following expression is obtained:

$$x_n(k+1) = A_n x_n(k) + B_n u_n(k) + B_z z(k). \tag{14}$$

For each component, an output equation like in (15) has to be written. The quantity φ is the value of the equivalent source added in substitution of the component in the network solution

$$\varphi(k+1) = C_n x_n(k+1). \tag{15}$$

The quantity φ has to be expressed as linear combination of state variables.

Substituting (15) and (14) into (8), the following equation is obtained:

$$\begin{aligned} x(k+1) = {} & Ax(k) + Bu(k+1) + B_\varphi C_n A_n x_n(k) \\ & + B_\varphi C_n B_n u_n(k) + B_\varphi C_n B_z z(k). \end{aligned} \tag{16}$$

Substituting now z into (11) from the previous time step, the following equation is obtained:

$$\begin{aligned} x(k+1) = {} & Ax(k) + Bu(k+1) + B_\varphi C_n A_n x_n(k) \\ & + B_\varphi C_n B_n u_n(k) + B_\varphi C_n B_z C x(k) \\ & + B_\varphi C_n B_z D u(k) + B_\varphi C_n B_z D_\varphi C_n x_n(k). \end{aligned} \tag{17}$$

Looking at the components equations, substituting (11) from the previous time step into (14), we have

$$\begin{aligned} x_n(k+1) = {} & A_n x_n(k) + B_n u_n(k) + B_z C x(k) \\ & + B_z D u(k) + B_z D_\varphi C_n x_n(k). \end{aligned} \tag{18}$$

From (17) and (18), the overall system representation is obtained

$$\begin{aligned} \begin{bmatrix} x(k+1) \\ x_n(k+1) \end{bmatrix} = {} & A_{\text{full}} \begin{bmatrix} x(k) \\ x_n(k) \end{bmatrix} \\ & + B_{\text{full1}} u(k+1) + B_{\text{full2}} \begin{bmatrix} u(k) \\ u_n(k) \end{bmatrix} \end{aligned} \tag{19}$$

where

$$A_{\text{full}} = \begin{bmatrix} A + B_\varphi C_n B_z C & B_\varphi C_n A_n + B_\varphi C_n B_z D_\varphi C_n \\ B_z C & A_n + B_z D_\varphi C_n \end{bmatrix} \tag{20}$$

$$B_{\text{full1}} = \begin{bmatrix} B \\ 0 \end{bmatrix} \tag{21}$$

$$B_{\text{full2}} = \begin{bmatrix} B_\varphi C_n B_z D & B_\varphi C_n B_n \\ B_z D & B_n \end{bmatrix}. \tag{22}$$

The solution performed with the presented LB-LMC method will be asymptotically stable if all eigenvalues of A_{full} have magnitude strictly smaller than 1.

A simple example shows how the presented method affects the stability and accuracy of the simulation.

In Fig. 4, the test case selected to study the stability properties of the presented LB-LMC method is illustrated. In Table I, the relative parameters are reported. We assume that the inductor is solved separately by using the state equation explicitly integrated and that the rest of the circuit is solved using resistive companion and discretized implicitly.

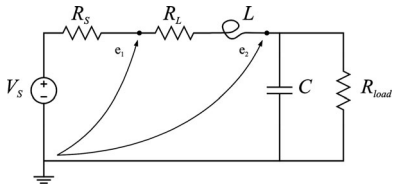

Fig. 4. Stability linear case.

TABLE I
STABILITY AND MULTIRATING TEST CASE

V_S	10 V
R_S	0.001 Ω
R_L	0.001 Ω
L	0.1 H
C	0.01 F
R_{load}	10 Ω

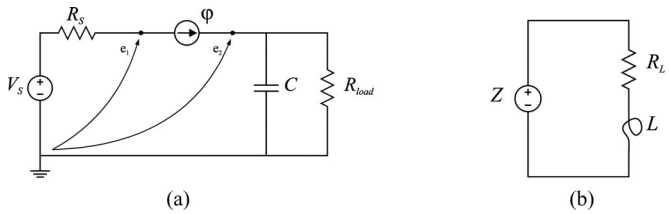

Fig. 5. Stability test case. (a) Resistive companion section. (b) State equations section.

TABLE II
MATRIX A_{full} EIGENVALUES

	$\Delta t = 73.9$ ms	$\Delta t = 74$ ms
λ1	0.9824	1.0001
λ2	0.5845	0.5738

In Fig. 5(a), the equivalent circuit to be solved with resistive companion is reported. Fig. 5(b) shows the equivalent circuit of the part of the circuit solved using state equations.

A state-space representation of the two subsystems was formulated to study the stability of the simulation as a function of the time step. The resistive companion part has been discretized using Euler backward, while the state equation part using Euler forward. Any other implicit or explicit method would fit the presented method and the stability study performed. Clearly, the use of different integration methods can make the solution more or less accurate and determine different stability limits.

Also, a full state-space representation (using Euler backwards as the discretization method) of the test case is used as reference.

Let us now use matrix A_{full} (see (20)) to identify the critical time step for the considered circuit. A screening of the time step is performed with a resolution of 100 μs and the eigenvalues of matrix A_{full} are checked for every value of time step. The largest time step, for which all the eigenvalues have magnitude strictly smaller than 1, is 73.9 ms. In Table II, the eigenvalues of matrix A_{full} computed for $\Delta t = 73.9$ ms and for $\Delta t = 74$ ms are reported, showing clearly that this is the stability limit.

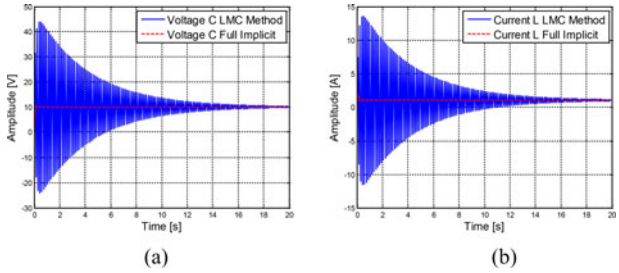

Fig. 6. Stability test case results ($\Delta t = 73.9$ ms): (a) capacitor voltage and (b) inductor current.

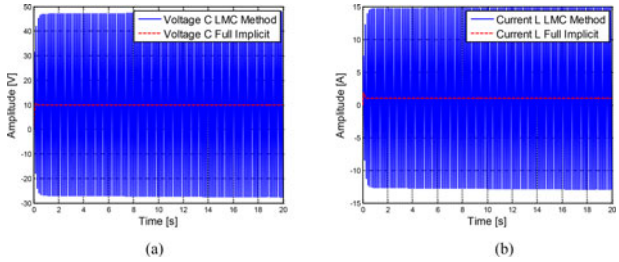

Fig. 7. Stability test case results ($\Delta t = 74$ ms): (a) capacitor voltage and (b) inductor current.

Fig. 6(a) shows the behavior of capacitor voltage when computed with $\Delta t = 73.9$ ms, while Fig. 6(b) shows the current in the inductor, also computed with the same time step. As the two behaviors show, with a time step equal to 73.9 ms, we are just within the stability limit.

Even if the oscillations of both voltage and current are not characteristics of the physical system but are introduced by the solution method, they are damped and the solution remains bounded. In contrast, in Fig. 7(a), the behavior of the capacitor voltage when computed with $\Delta t = 74$ ms is reported; in Fig. 7(b), the current in the inductor, also computed with the same time step, is reported. In this case, as easy to imagine looking at Table II, the oscillations are undamped and the solution diverges.

It is important to emphasize that the eigenvalues shown in Table II have been computed using matrix A_{full} where both circuits of Fig. 5 were modeled in a state-space framework. Vice versa, the results of Figs. 6 and 7 were obtained using a resistive companion representation of the circuit in Fig. 5(a). These results show the consistency between the two modeling approaches when the same discretization method is applied.

Even if using a time step equal to 73.9 ms allows a stable simulation of the system, the results accuracy is quite poor. In Figs. 8 and 9, results from the same system simulated with a time step of 1 ms are reported and compared to the reference solution. In Fig. 8, the voltage of the capacitor is reported with the relative error. In Fig. 9, the current in the inductor is reported with the relative error.

To complete the stability analysis of the presented LB-LMC method, we now analyze the behavior of the circuit when fully

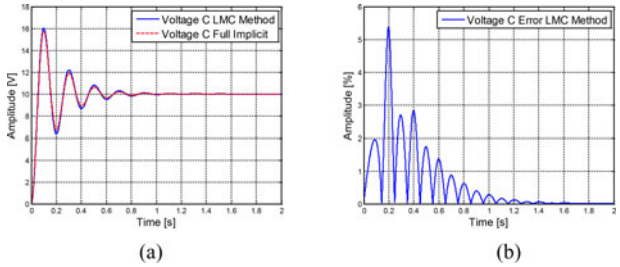

Fig. 8. Stability test case results ($\Delta t = 1$ ms): (a) capacitor voltage and (b) capacitor voltage error.

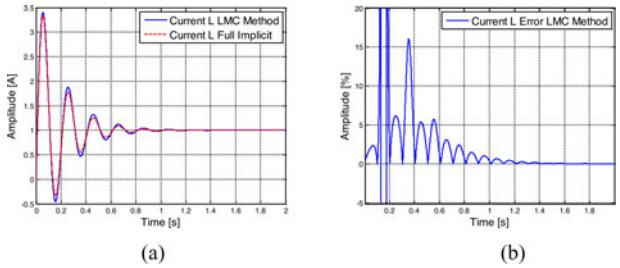

Fig. 9. Stability test case results ($\Delta t = 1$ ms): (a) inductor current and (b) inductor current error.

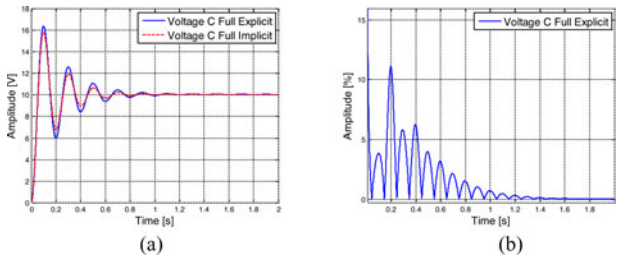

Fig. 10. Explicit integration test case results ($\Delta t = 1$ ms): (a) capacitor voltage and (b) capacitor voltage error.

explicitly discretized. This test is particularly interesting since an explicit discretization of the full circuit, modeled using state equation, would allow a perfect parallelization of the solution. A new screening of the time step size is performed, and the result is that the maximum time step to obtain a stable solution is equal to 9.9 ms. It is clear that fully explicit discretization significantly reduces the range of stability.

As easy to imagine, the accuracy of the simulation will also be negatively influenced by the use of an explicit method for the integration of the whole circuit. With this purpose, in Figs. 10 and 11, results of the simulation performed using a fully explicit integration with a time step of 1 ms are reported and compared with the reference case.

Fig. 10 shows the voltage across the capacitor with the relative error. Fig. 11 shows the current through the inductor with the relative error.

Comparing Figs. 8(b) and 9(b) to Figs. 10(b) and 11(b), it is clear that the error is significantly larger. This shows an important property, as reported in [36], of LB-LMC methods that are valid also for nonlinear systems. The LB-LMC method presents

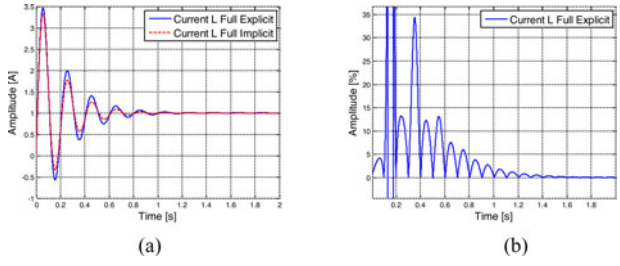

Fig. 11. Explicit integration test case results ($\Delta t = 1$ ms): (a) inductor current and (b) inductor current error.

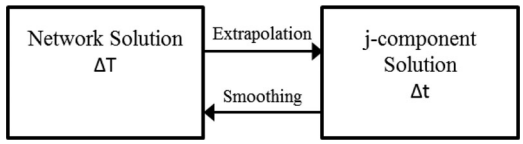

Fig. 12. j-Component solution with a time step smaller than the network solution one.

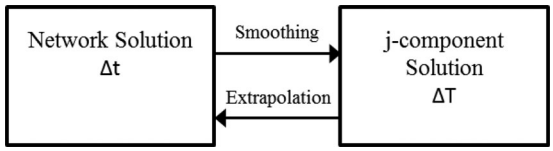

Fig. 13. j-Component solution with a time step larger than the network solution one.

a stability behavior that is better than the one offered by the worst integration method used.

V. Multirate Execution

It is reasonable to imagine that different components of the circuit will have different internal dynamics that may require different refresh rates. Let us consider, for example, the case of a big electric machine compared to power electronics converter. In this context, it makes sense to update the different components with the most suitable time step rather than computing the resistive companion solution as well as the solution of all nonlinear components with the time step required by the component with the fastest dynamic.

Using different time steps for different parts of the system is a well-proven technique to reduce the computational load, but it introduces the problem of defining the data to exchange between zones that have different time steps.

As illustrated in Fig. 12, data produced by the integration of a component with a small time step Δt must be smoothed to calculate a single representative value to be used for the network solution computed with the large time step ΔT. The values computed by the network solution must be extrapolated to many values to feed to the component solution. Fig. 13 illustrates the opposite situation, in which the network solution is executed with a smaller time step than the component. The process can be extended to multiple components and also to multiple time steps: the only constraint for the method applied in this paper is that the largest time step should be a multiple integer of the smallest one.

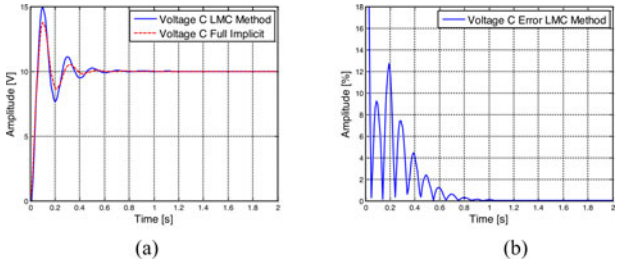

Fig. 14. Multirating test case $\Delta t = 1$ ms, $\Delta T = 10$ ms: (a) capacitor voltage and (b) related error.

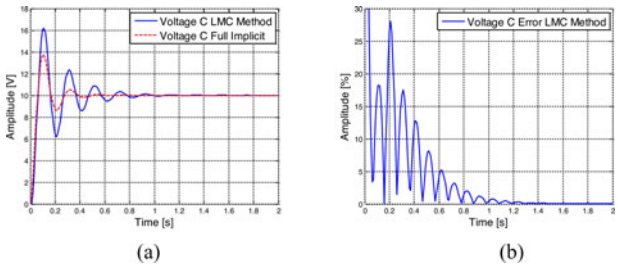

Fig. 15. Multirating test case $\Delta t = \Delta T = 10$ ms: (a) capacitor voltage and (b) related error.

For smoothing and extrapolation, we applied the same method defined in [32]. For the smoothing process, that is strictly necessary when a multirate approach is used, we used a linear interpolation. As distinct from smoothing, extrapolation is not strictly necessary to perform a multirate simulation, but it can improve the simulation accuracy. The idea is that instead of keeping the value calculated by the slow subsystem constant throughout the interval ΔT, a better approximation of the real evolution of the associated variable can be created by extrapolating from previously calculated values. This kind of numerical technique is very well known, and increasing the number of previous points taken into account allows a higher order approximation to be created. Even though this procedure may appear to increase the accuracy, the use of such higher order approximations can create numerical instabilities when the interconnection variables are discontinuous or have fast harmonic components.

The same test case used for the stability analysis (see Fig. 4) has also been used to investigate the multirate execution. Considering the values of inductance and capacitance, we solve the resistive companion part of the circuit in Fig. 5(a) with a time step equal to 1 ms and the state equation related to the inductance [see Fig. 5(b)] with a time step of 10 ms.

Fig. 14 shows the voltage across the capacitor and the related error, both smoothing and extrapolation have been directly implemented. As expected, the results obtained are less accurate than the one obtained simulating the whole circuit with a time step equal to 1 ms (see Fig. 8).

It is interesting to also take a look at results accuracy when the whole circuit is simulated with a single time step size equal to 10 ms. As shown in Fig. 15, the accuracy is very low and probably unacceptable for any practical application. In Table III, the mean and the variance of the error in the voltage calculation is reported for the three cases considered. Values are reported

TABLE III
ACCURACY OF VOLTAGE ON CAPACITOR

	Transitory	
	Mean	**Variance**
$\Delta t = \Delta T = 10$ ms	7.3088	56.8102
$\Delta t = \Delta T = 1$ ms	1.3754	0.3190
$\Delta t = 1$ms, $\Delta T = 10$ ms	2.4275	10.3405

TABLE IV
ACROSS AND THROUGH QUANTITIES FOR VARIOUS DISCIPLINES

	Across	**Through**
Electrical	Voltage	Current
Thermal	Temperature	Heat flow
Fluid	Pressure	Flow rate
Mechanical rotation	Speed	Torque
Mechanical translation	Velocity	Force

only for the first second of the simulation since in steady state the error approaches zero.

Concluding on multirate execution, it is clear that the selection of multirating execution represents a compromise between accuracy (higher accuracy can be obtained performing the whole simulation with the smaller time step) and step/s size. In general, it is always preferable to select a unique and smaller time step if allowed by the RT execution requirement. Multirating can play significant role when computationally heavy and slow dynamic components do not allow RT execution at the time step required by other, more dynamic, elements in the circuit.

VI. MULTIPHYSIC APPLICATION

In the previous sections, we describe the presented method only referring to electrical examples. In reality, resistive companion and obviously state-space modeling can be applied to many other fields as long as models based on the connection of a set of lumped parameter components can be formulated. Voltage and current should be substituted with the appropriate across and through quantities typical of object-oriented multiphysics modeling languages [37].

According to Table IV, where the analogs between mechanic rotational quantities and electrical quantities are summarized, we now explain step by step how to model of an induction machine so that is executable with the presented LB-LMC method.

Let us consider a seventh-order model expressed in the phase variables [38]. For sake of simplicity, we do not consider here transformation in the $\beta\alpha 0$ or $dq0$ reference frame that should be implemented to keep the inductance matrix constant. The electrical part of the machine is described by six state equations in the stator and rotor fluxes

$$\frac{d\psi_S}{dt} = v_S + R_S I_S \tag{23}$$

$$\frac{d\psi_R}{dt} = v_R + R_R I_R \tag{24}$$

where R_S and R_R (in ohms) are the matrices of rotor and stator resistances

$$\psi_S = \begin{bmatrix} \psi_{S1} \\ \psi_{S2} \\ \psi_{S3} \end{bmatrix} \tag{25}$$

$$\psi_R = \begin{bmatrix} \psi_{R1} \\ \psi_{R2} \\ \psi_{R3} \end{bmatrix} \tag{26}$$

where (25) is the vector of the stator fluxes and (26) is the vector of the rotor fluxes

$$v_S = \begin{bmatrix} v_{S1} \\ v_{S2} \\ v_{S3} \end{bmatrix} \tag{27}$$

$$v_R = \begin{bmatrix} v_{R1} \\ v_{R2} \\ v_{R3} \end{bmatrix}. \tag{28}$$

Here, (27) is the vector of the stator voltages and (28) is the vector of the rotor voltages

$$I_S = \begin{bmatrix} I_{S1} \\ I_{S2} \\ I_{S3} \end{bmatrix} \tag{29}$$

$$I_R = \begin{bmatrix} I_{R1} \\ I_{R2} \\ I_{R3} \end{bmatrix}. \tag{30}$$

Here, (29) is the vector of the stator currents and (30) is the vector of the rotor currents.

We want to obtain a current-type model for the stator terminals of the machine expressed in terms of stator currents. The relation between the stator and rotor currents and stator and rotor fluxes is reported in the following expressions:

$$\psi_S = L_S I_S + M_S I_R \tag{31}$$

$$\psi_R = L_R I_R + M_R I_S. \tag{32}$$

In this case, the stator currents are not directly state variables of the model, as it was the case for the inductor current in the example used in the stability study; instead, it is a linear combination of them. Matrix C_n of (11) is given by (34)

$$C = \begin{bmatrix} L_S & M_S \\ M_R & L_R \end{bmatrix}^{-1} \tag{33}$$

$$C_n = C(1-3,:). \tag{34}$$

The model is then completed with one mechanical equation (35)

$$\frac{d\Omega_m}{dt} = \frac{(T_t - T_e)}{J} \tag{35}$$

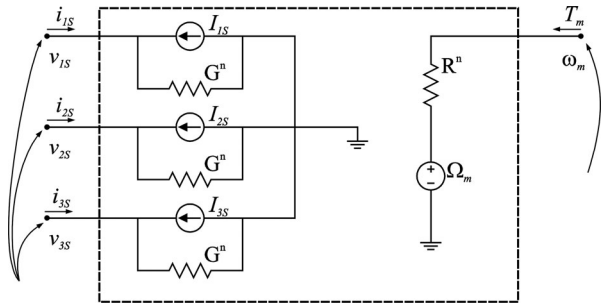

Fig. 16. Induction machine resistive companion equivalent circuit.

where: Ω_{m} (in radians per second) is the shaft speed, p the number of pole pairs, J the rotor inertia, T_t (N·m) is the mechanical torque, and T_e (N·m) is the electrical torque obtained as follows:

$$T_e = p \begin{bmatrix} I_s^t & I_R^t \end{bmatrix} \begin{bmatrix} \psi_S \\ \psi_R \end{bmatrix}. \tag{36}$$

The shaft speed has to be interpreted as an across quantity (similarly to voltage), and the mechanical torque as a through quantity (similarly to current). In Fig. 16, the induction machine equivalent circuit to be used in the resistive companion part of the presented LB-LMC method is presented.

The stamp is performed properly by adding matrix (37) to the conductance matrix and vector (38) to the source vector. During each iteration, the contributions to the source vector are updated according to (23), (24), (31), (32), (35), and (36)

$$\begin{array}{c} \begin{bmatrix} G^n & & & \\ & G^n & & \\ & & G^n & \\ & & & \dfrac{1}{R^n} \end{bmatrix} \begin{matrix} p_1 \\ p_2 \\ p_3 \\ s \end{matrix} \\ \begin{matrix} p_1 & p_2 & p_3 & s \end{matrix} \end{array} \tag{37}$$

$$\begin{bmatrix} I_{1s} \\ I_{2s} \\ I_{3s} \\ \dfrac{\Omega_m}{R^n} \end{bmatrix} \begin{matrix} p_1 \\ p_2 \\ p_3 \\ s \end{matrix}. \tag{38}$$

VII. RT Testing

The presented LB-LMC method has been developed in the framework of a larger research effort for developing an RT simulator for multiphysic applications, and the simulation software Virtual Test Bed has been used as front end. The LB-LMC method can be tested using the desktop version of this tool,

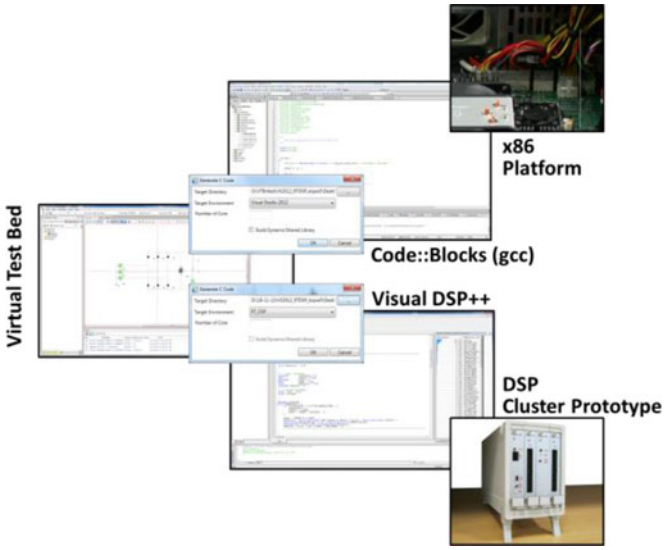

Fig. 17. Code generation process.

but it is also possible to generate the C code for RT execution on a DSP cluster prototype or on a x86-based platform. The DSP cluster prototype is based on two TigerSHARC TS201 DSP with a 600 MHz clock. The code execution is distributed between the two processors, and exchange of information is implemented with a message-passing approach; for more details and hardware characteristics, see [39] and [40]. The x86 platform is based on a i7, 3.4 GHz processor and on an RT Linux kernel (3.13.0–24 generic kernel patched with the 3.14.3-rt4). Open-MP has been used in order to parallelize the code. The DSP platform has FPGA-based input output interfaces and so better fits power-electronics-oriented HIL testing. Nevertheless, the DSP prototype scalability is quite limited, and we cannot use it to execute large models. The x86 platform suffers from higher latency on the I/O management, but due to the higher number of cores and to the higher clock speed, it can be used to simulate significantly larger systems.

In Fig. 17, the code generation process is illustrated: for the DSP platform, the C code generated using Virtual Test Bed (VTB) is compiled using Visual DSP++ and afterward downloaded on the DSP cluster prototype, and for the x86 platform, the code generated using VTB is directly compiled on the Linux system using GCC 4.8.2.

From the implementation point of view, starting from a resistive companion-based solver, the modification required for the solver structure is mainly limited to memory management for parallel execution. New models for the nonlinear components, if previously developed for nonlinear resistive companion solvers, have to be developed, which can be large and time-consuming effort. At the same time, if new models are needed, modeling with state equation is typically easier and less error prone if compared to what is typically required by resistive companion solvers.

To test the RT execution, the performance, and the accuracy of the presented method, four simulation examples have been implemented in (VTB). The first example is a simple single-phase diode bridge, and the second one is a three-phase dc-ac converter. Both examples are executed on the DSP prototype with a 20 μs time step. The third example is a microgrid composed of a dc bus, a dc–dc converter that controls the voltage of the dc bus, and three dc–ac three-phase converters. The fourth example is a multiphysic system composed of a three-phase dc–ac converter and its liquid cooling system. Both examples are executed on the x86 platform. The microgrid example is simulated with a 15 μs time step. The fourth example is simulated using a multirating approach: a 15 μs time step has been used for the electromechanical part of the system and a 10 ms time step for the hydraulic and thermal ones.

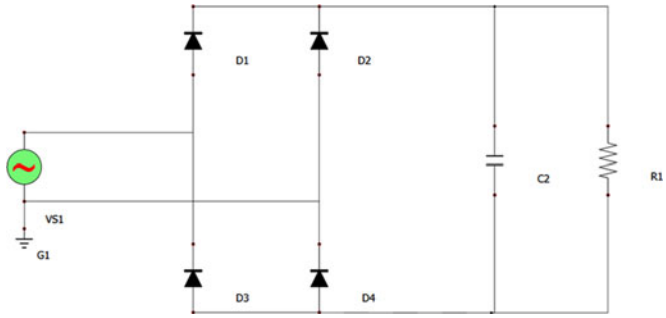

Fig. 18. Single-phase diode bridge schematic tin VTB.

TABLE V
SINGLE-PHASE DIODE BRIDGE EXAMPLE PARAMETERS

VS1	10 V
C1	10 mF
R1	5 Ω

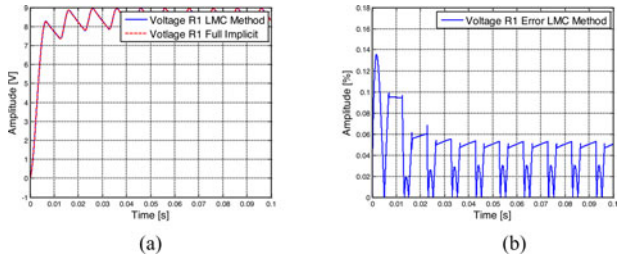

Fig. 19. (a) R1 voltage; (b) related error.

A. Single-Phase Diode Bridge

This first circuit created is the single-phase diode bridge shown in Fig. 18: this example, even if extremely small, is interesting for the stability problem that affects simulation of this type of circuits when no snubber circuit is considered for the diodes. The diodes are modeled by explicitly solving the Shockley diode equation.

In Table V, the parameters for the circuit of Fig. 18 are reported.

The code generated using VTB is then executed on the DSP cluster. To evaluate the accuracy of the presented method, the results obtained with the implementation of the LB-LMC method are compared to the one obtained using a traditional resistive

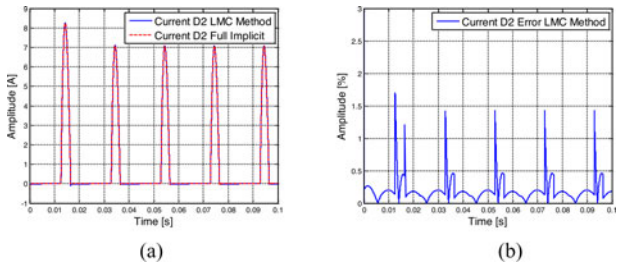

Fig. 20. (a) D2 current; (b) related error.

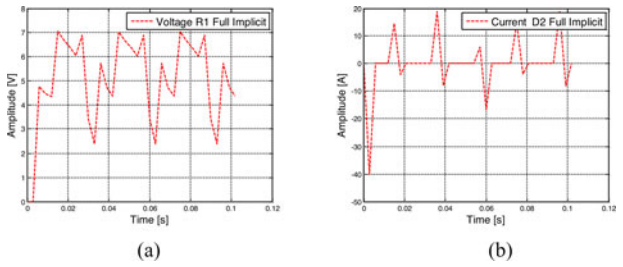

Fig. 21. Full implicit solver $\Delta t = 3$ ms: (a) R1 voltage; (a) D2 current.

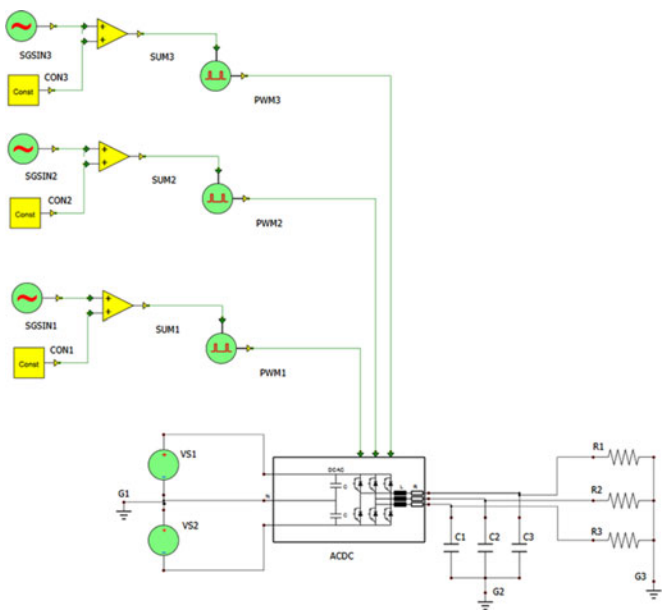

Fig. 22. Three-phase dc–ac converter schematic in VTB.

companion solver. In Fig. 19, the voltage across the resistor R1 is reported with the relative error; in Fig. 20, the current through the diode D2 is reported with the relative error.

The results are extremely positive, showing that in this particular case, the presented method introduces an extremely small root-mean-square (rms) error: 0.06% for the voltage on R1 and 0.35% for the current through D2.

Using the examples in this final section, we also want to verify the stability performance of the presented method when dealing with nonlinear systems. For this first example, the LB-LMC model of the circuit is unstable for a time step bigger than 3 ms; this value is significantly larger than the target time step and indicates that we are operating significantly far from the stability limit. It is also worth to underline that even if the results of the fully implicit method are stable using a 3 ms time step, as reported in Fig. 21, the accuracy is unacceptable for any practical application.

TABLE VI
THREE PHASES DC–AC CONVERTER EXAMPLE PARAMETERS

VS1, VS2	600 V
C	10 mF
L	6 mH
R	0.05 Ω
C1, C2, C3	2 μF
R1, R2, R3	5 Ω
f_{sw}	4 kH

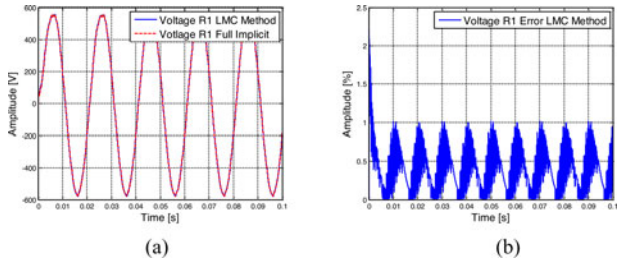

Fig. 23. (a) R1 voltage; (b) related error.

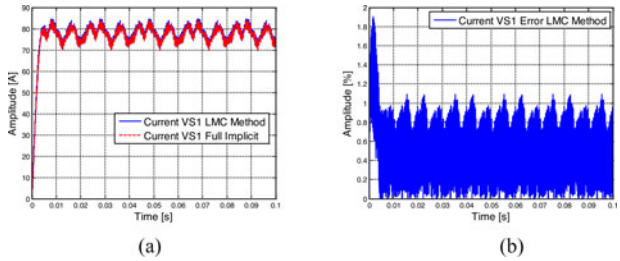

Fig. 24. (a) VS1 current; (b) related error.

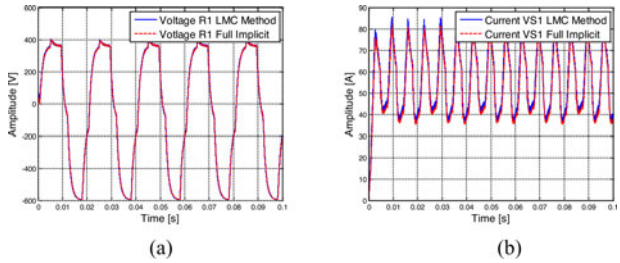

Fig. 25. Full implicit solver $\Delta t = 250$ μs: (a) R1 voltage; (a) VS1 current.

B. *Three-Phase DC–AC Converter*

The second case is based on a three-phase dc/ac converter. This case has been selected to test the presented method with a larger and more useful circuit. The dc/ac converter is modeled using a switching function approach assuming ideal devices.

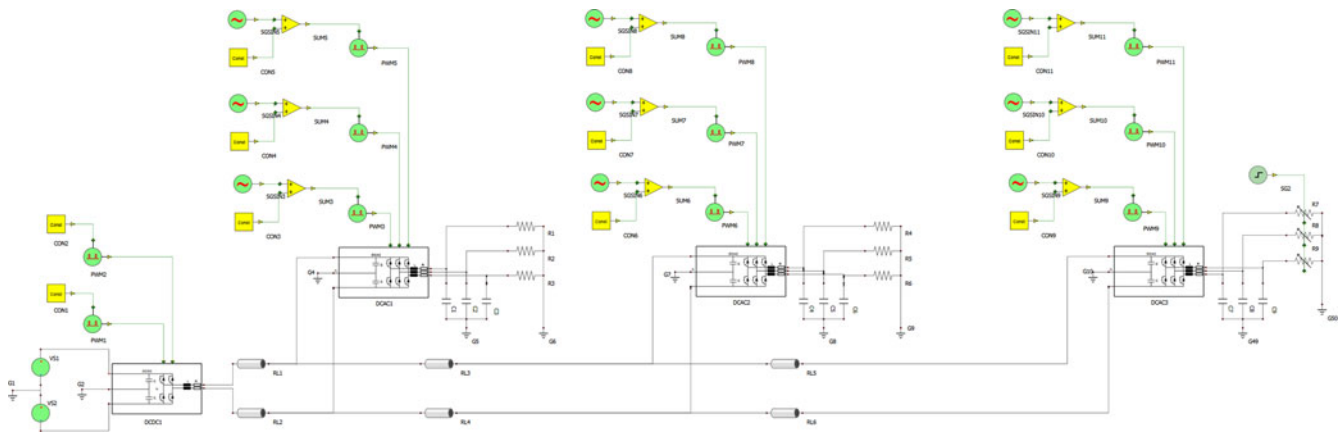

Fig. 26. Microgrid converter schematic in VTB.

TABLE VII
DC–DC CONVERTER PARAMETERS

C	10 mF
L	1 mH
R	0.1 Ω
f_{sw}	4 kH

TABLE VIII
CABLES PARAMETERS

R	0.1 Ω
L	1 μH

The VTB schematic is reported in Fig. 22, and the parameters of the circuit are listed in Table VI. As in the case of the single-phase diode bridge, the code generated using VTB is executed on the DSP cluster prototype.

As in the previous example, to evaluate the accuracy of the presented method, the results obtained by the implementation of the LB-LMC method are compared to the ones obtained using a traditional resistive companion solver. In Fig. 23, the voltage across the resistor R1 is reported with the relative error. In Fig. 24, the current through the voltage source VS1 is reported with the relative error. Also in this case, the rms error introduced by the presented solution method is small: 0.48% for the voltage on R1 and 0.68% for the current through VS1.

For this example, the LB-LMC solver-based simulation is unstable for a time step bigger than 25 ms. As in the previous example, this value is significantly larger than the targeted time step. Moreover, considering a switching frequency of 4 kHz, it is clear that a time step of 25 ms is not acceptable independently of the used solver. In Fig. 25, we show how with a time step equal to 250 μs the results obtained by the fully implicit method and by the LB-LMC ones are still stable and comparable but already strongly inaccurate.

C. Microgrid

The third test case is a microgrid composed of a dc bus, a dc/dc converter that control the voltage of the dc bus, and three dc/ac three-phase converters. This test case has been executed on the x86-based platform where a 15 μs time step has been used. In Fig. 26, the VTB schematic is reported; to model each of the three-phase converter, we used the same approach and parameters of the previous example. The parameters of the dc/dc converter and of the cables are reported in Tables VII and VIII. After 0.2 s, the load of the third inverter changes from 5 to 2 Ω.

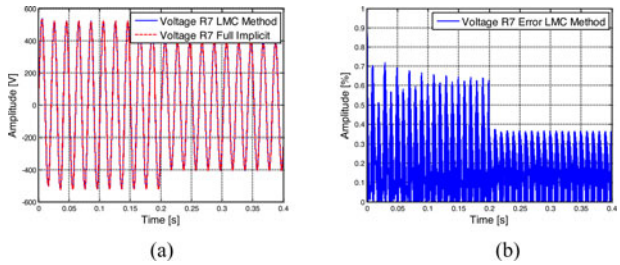

Fig. 27. (a) R7 voltage; (b) related error.

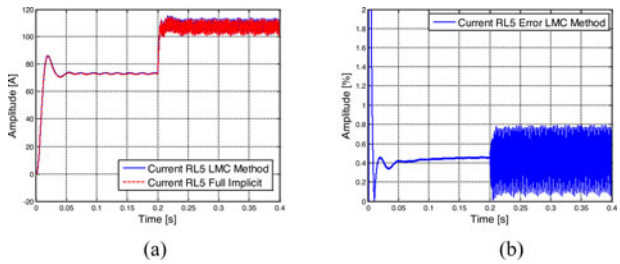

Fig. 28. (a) RL5 current; (b) related error.

As in the previous examples, the results obtained by the implementation of the LB-LMC method are compared to the ones obtained using a traditional resistive companion solver.

In Fig. 27, the voltage across the resistor R7 is reported with the relative error; the same is done in Fig. 28 for the current through the cable RL5, and in Fig. 29 for the current through the cable RL1.

Also, in this case, the rms error introduced by the presented solution method is small: 0.23% for the voltage on R7, 0.43% for the current through RL5 before the load change, 0.47% for

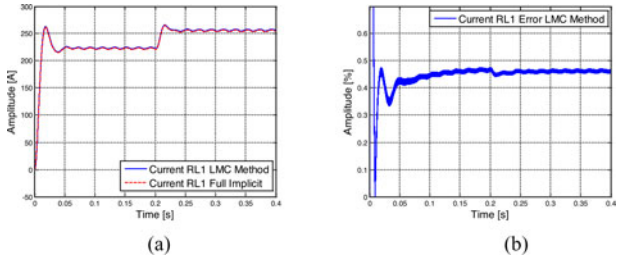

Fig. 29. (a) RL1 current; (b) related error.

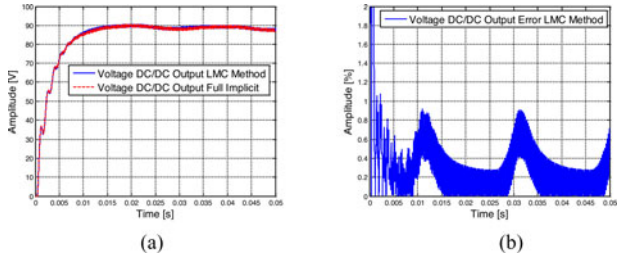

Fig. 31. (a) DC/DC converter output voltage; (b) related error.

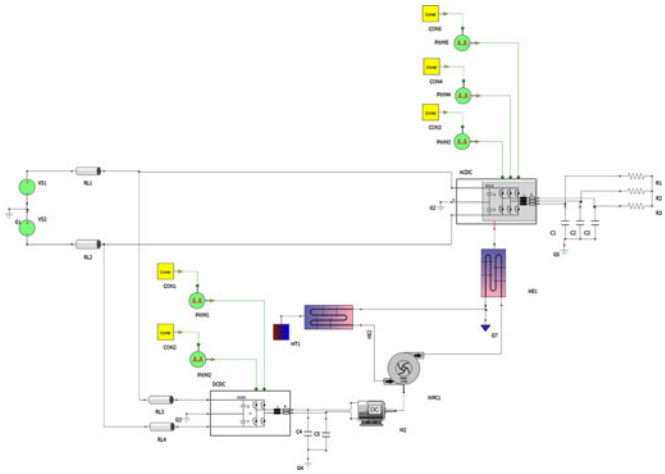

Fig. 30. Three-phase inverter cooling schematic in VTB.

TABLE IX
DC–DC CONVERTER PARAMETERS

C_{in}	10 mF
L	100 μH
R	0.1 Ω
C_{out}	500 μF
f_{sw}	4 kH

TABLE X
DC MACHINE PARAMETERS

R_{arm}	1 Ω
L_{arm}	0.5 H
K_{motor}	1 N · m/A
Inertia	0.01 kg · m^2
Friction	0.1 N · m · s

TABLE XI
PUMP AND HEATERS

$\text{Pump}_{\text{OpenFlow}}$	2.35 GPM
$\text{Pump}_{\text{InternalVolume}}$	0.02 m^3
$R_{\text{case-sink}}$	0.009 K/W
Ambient Temp	298 K
$\text{Pipe}_{\text{Roughness}}$	0.05 mm

the current through RL5 after the load change, and 0.45% for the current through RL1. In this example, the LB-LMC solver-based simulation is unstable for a time step bigger than 250 μs. This value is significantly smaller than the one obtained in the previous tests; this is due to the small cable inductance. Increasing the inductance of the cables significantly increases the value of the stability limit. This test confirms that also in this case the selection of the time step to be used is mainly driven by the switching frequency; even small reactive elements are sufficient to make the solution with the LB-LMC solver stable and accurate. As shown in the previous example, the use of a larger time step size is mainly limited by the switching frequency used.

D. *Three-Phase Inverter Cooling*

To conclude, we present a multiphysic example composed of a three phase dc/ac converter and of its liquid cooling system. This example has been selected because electrical, mechanical, thermal, and hydraulic domains are present at the same time. The main components of the circuit are a three-phase converter (a model with a thermal port has been used), a dc/dc converter that supply a dc motor connected to a pump, and two heat exchangers. In Fig. 30, the VTB schematic is reported; for the dc/ac converter and for the cables, we used the same parameters used in the previous examples. The remaining significant parameters of the system are listed in Tables IX–XI. The system has been simulated on the x86-based platform using a multirating approach: all the electrical components, including the dc machine, have been simulated using a time step equal to 15 μs, while for the hydraulic pump and the heat exchangers, a time step equal to 10 ms has been used (see Fig. 31).

Also, in this case, the results obtained are compared to the ones obtained using a traditional resistive companion solver executed using a 15 μs time step. In Fig. 31, the output voltage of the dc/dc converter is reported with the relative error. The same is done in Fig. 32 for the dc machine speed, and in Fig. 33 for the outlet temperature of the heat exchanger connected to the dc/ac converter.

Also, in this last case, the rms error introduced by the LB-LMC method is always significantly smaller than 1%: 0.35% for the dc/dc converter output voltage, 0.25% for the dc machine speed, and 0.09% for the outlet temperature of the heat exchanger.

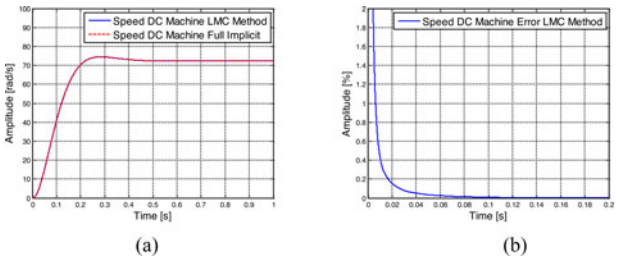

Fig. 32. (a) DC machine speed; (b) related error.

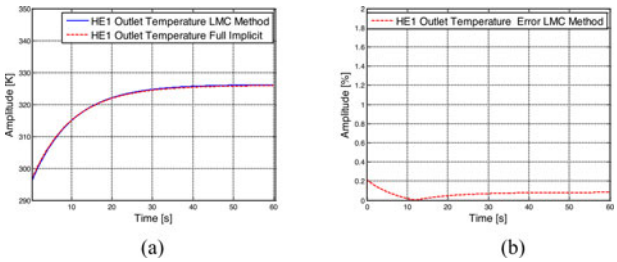

Fig. 33. (a) HE1 outlet temperature; (b) related error.

VIII. Conclusion

In this paper, we presented a new simulation method that we named latency-based linear multistep compound method. The method relying on a mixed integration scheme separates the solution of single nonlinear components from the one of the network interconnecting the components. This approach makes the method more parallelizable and scalable than typical partitioning methods, without requiring any action by the simulator user.

At the same time, the explicit integration of the nonlinear components can make the simulation unstable if the time step selected is too large. In Section VII, we have shown that for the considered power electronic systems, the time step selection is mainly driven by the switching frequency involved and not by the stability limit of the LB-LMC method. In all the examples, the time step required by the switching frequency was smaller than the one required for a stable and accurate simulation.

The main limitation of the presented implementation of this method is a minimum time step of 10–20 μs. These values, which are strictly related to the test platform used, limit the field of applicability to systems with switching frequency smaller than 10 kHz (if models averaged over the time step are not used). To simulate systems with higher switching frequencies, implementation on other platforms, like FPGA, should be considered. Still these have not been considered in this paper.

Another limit of the proposed approach is that for multirating simulation, we still rely on user involvement. Even if the partitioning is fully automatized, the selection of the most appropriate time step for each component of the system is still done by users.

References

[1] P. McLaren, R. Kuffel, R. P. Wierckx, J. Giesbrecht, and L. H. Arendt, "A real time digital simulator for testing relays," *IEEE Trans. Power Del.*, vol. 7, no. 1, pp. 207–213, Jan. 1992.

[2] J. R. Martí and L. R. Linares, "Real-time EMTP-based transients simulation," *IEEE Trans. Power Syst.*, vol. 9, no. 3, pp. 1309–1317, Mar. 1994.

[3] J. R. Marti and J. A. Hollman, "Real time network simulation with PC-cluster," *IEEE Trans. Power Syst.*, vol. 18, no. 2, pp. 563–569, Feb. 2003.

[4] M. Armstrong, J. R. Marti, L. R. Linares, and P. Kundur, "Multilevel MATE for efficient simultaneous solution of control systems and nonlinearities in the OVNI simulator," *IEEE Trans. Power Syst.*, vol. 21, no. 3, pp. 1250–1259, Feb. 2006.

[5] R. Kuffel, J. Giesbrecht, T. Maguire, R. P. Wierckx, and P. McLaren, "RTDS a fully digital power system simulator," in *Proc. Int. Conf. Digital Power Syst. Simulators*, 1995, pp. 19–24.

[6] G. G. Parma and V. R. Dinavahi, "Real-time digital hardware simulation of power electronics and drives," *IEEE Trans. Power Del.*, vol. 22, no. 2, pp. 1235–1246, Feb. 2007.

[7] V. R. Dinavahi, R. R. Iravani, and R. Bonert, "Real-time digital simulation of power electronic apparatus interfaced with digital controllers," *IEEE Trans. Power Del.*, vol. 16, no. 4, pp. 775–781, Apr. 2001.

[8] R. Champagne, L. A. Dessaint, H. Fortin-Blanchette, and G. Sybille, "Analysis and validation of a real-time AC drive simulator," *IEEE Trans. Power Electron.*, vol. 19, no. 2, pp. 336–345, Feb. 2004.

[9] L. Bin, X. Wu, H. P. Figueroa, and A. Monti, "A low-cost real-time hardware-in-the-loop testing approach of power electronics controls," *IEEE Trans. Ind. Electron.*, vol. 54, no. 2, pp. 919–931, Feb. 2007.

[10] L. Yunwei, D. M. Vilathgamuwa, and L. P. Chiang, "Design, analysis, and real-time testing of a controller for multibus microgrid system," *IEEE Trans. Power Electron.*, vol. 19, no. 5, pp. 1195–1204, May 2005

[11] Z. R. Ivanovic, E. M. Adzić, M. S. Vekić, S. U. Grabić, N. L. Celanović, and V. A. Katić "HIL evaluation of power flow control strategies for energy storage connected to smart grid under unbalanced conditions," *IEEE Trans. Power Electron.*, vol. 27, no. 11, pp. 4699–4710, Nov. 2012

[12] M. S. Vekić, S. U. Grabić, D. P. Majstorović, I. L. Celanović, N. L. Celanović, and V. A. Katić, "Ultralow latency HIL platform for rapid development of complex power electronics systems," *IEEE Trans. Power Electron.*, vol. 27, no. 11, pp. 4436–4444, Nov. 2012.

[13] J. Jin-Hong, K. Jong-Yul, K. Hak-Man, K. Seul-Ki, C. Changhee, K. Jang-Mok, A. Jong-Bo, and N. Keo-Yamg, "Development of hardware in-the-loop simulation system for testing operation and control functions of microgrid," *IEEE Trans. Power Electron.*, vol. 25, no. 12, pp. 2919–2929, Dec. 2010.

[14] L. W. Chung and D. Drury, "Development of a hardware-in-the-loop simulation system for testing cell balancing circuits," *IEEE Trans. Power Electron.*, vol. 28, no. 12, pp. 5949–5959, Dec. 2013.

[15] H. P. Figueroa, A. Monti, and X. Wu, "An interface for switching signals and a new real-time testing platform for accurate hardware-in-the-loop simulation," in *Proc. IEEE Int. Symp. Ind. Electron.*, 2004, pp. 883–887.

[16] M. Matar and R. R. Iravani, "FPGA implementation of the power electronic converter model for real-time simulation of electromagnetic transients," *IEEE Trans. Power Del.*, vol. 25, no. 2, pp. 852–860, Feb. 2010.

[17] M. Matar and R. R. Iravani, "Massively parallel implementation of AC machine models for FPGA-based real-time simulation of electromagnetic transients," *IEEE Trans. Power Del.*, vol. 26, no. 2, pp. 830–840, Feb. 2011.

[18] J. Liu and V. Dinavahi, "A real-time nonlinear hysteretic power transformer transient model on FPGA," *IEEE Trans. Ind. Electron.*, vol. 61, no. 7, pp. 1254–1260, Jul. 2014.

[19] H. Saad, T. Ould-Bachir, J. Mahseredjian, C. Dufour, S. Dennetiere, and S. Nguefeu, "Real-time simulation of MMCs using CPU and FPGA," *IEEE Trans. Power Electron.*, vol. 30, no. 1, pp. 259–267, Jan. 2015.

[20] C. Dufour and J. Bélanger, "On the use of real-time simulation technology in smart grid research and development," *IEEE Trans. Indus. Appl.*, Early Access.

[21] V. Jalili-Marandi, L. F. Pak, and V. R. Dinavahi, "Real-time simulation of grid-connected wind farms using physical aggregation," *IEEE Trans. Ind. Electron.*, vol. 57, no. 9, pp. 3010–3021, Sep. 2010.

[22] W. Li, G. Joós, and J Bélanger, "Real-time simulation of a wind turbine generator coupled with a battery supercapacitor energy storage system," *IEEE Trans. Ind. Electron.*, vol. 57, no. 4, pp. 1137–1145, Apr. 2010.

[23] J. Wu, A. Bose, J. Huang, A. Valette, and F. Lafrance, "Parallel implementation of power system transient stability analysis," *IEEE Trans. Power Syst.*, vol. 10, no. 3, pp. 1226–1233, Aug. 1995.
[24] M. La Scala and A. Bose, "Relaxation/Newton methods for concurrent time step solution of differential-algebraic equations in power system dynamic simulations," *IEEE Trans. Circuits Syst. I: Fundam. Theory Appl.*, vol. 40, no. 5, pp. 317–330, May 1993.
[25] P. Aristidou, D. Fabozzi, and T. Van Cutsem, "Dynamic simulation of large-scale power systems using a parallel schurcomplement-based decomposition method," *IEEE Trans. Parallel Distrib. Syst.*, vol. 25, no. 10, pp. 2561–2570, Oct. 2014
[26] V. Jalili-Marandi, Z. Zhou, and V. Dinavahi, "Large-scale transient stability simulation of electrical power systems on parallel GPUs," *IEEE Trans. Parallel Distrib. Syst.*, vol. 23, no. 7, pp. 1255–1266, Jul. 2012.
[27] G. Soykan, A. J. Flueck, and H. Dag, "Parallel-in-space implementation of transient stability analysis on a linux cluster with infiniband," in *Proc. North Amer. Power Symp.*, 2012, pp. 1–5.
[28] Z. Huang, R. Diao, S. Jin, and Y. Chen, "Predictive dynamic security assessment through advanced computing," in *Proc. IEEE Power Energy Soc. Gen. Meeting*, 2014, pp. 1–5.
[29] S. Jin, Z. Huang, R. Diao, D. Wu, and Y. Chen, "Parallel implementation of power system dynamic simulation," in *Proc. IEEE Power Energy Soc. Gen. Meeting*, 2013, pp. 1–5.
[30] Z. Huang, S. Jin, and R. Diao, "Predictive dynamic simulation for large-scale power systems through high-performance computing," in *Proc. 2012 SC Companion High Performance Comput. Netw. Storage Analysis*, pp. 347–354.
[31] S. Y. R. Hui and K. K. Fung, "Fast decoupled simulation of large power electronic systems using new two-port companion link models," *IEEE Trans. Power Electron.*, vol. 12, no. 3, pp. 462–473, Mar. 2007.
[32] A. Benigni, A. Monti, and R. Dougal, "Latency based approach to the simulation of large power electronics systems," *IEEE Trans. Power Electron.*, vol. 29, no. 6, pp. 3201–3213, Jun. 2014.
[33] T. Kato, K. Inoue, T. Fukutani, and Y. Kanda, "Multirate analysis method for a power electronic system by circuit partitioning," *IEEE Trans. Power Electron.*, vol. 24, no. 12, pp. 2791–2802, Dec. 2009.
[34] C. Dufour, J. Mahseredjian, and J. Belanger, "A combined state-space nodal method for the simulation of power system transient," *IEEE Trans. Power Del.*, vol. 26, no. 2, pp. 928–935, Feb. 2011.
[35] J. A. Hollman and J. R. Marti, "Step-by-step eigenvalue analysis with EMTP discrete-time solutions," *IEEE Trans. Power Syst.*, vol. 25, no. 3, pp. 1220–1231, Mar. 2010.
[36] M. Marin, A. Benigni, H. Lakhdar, A. Monti, and P. J. Collins, "Towards the implementation of a parallel real-time simulator for DSP cluster," in *Proc. Int. Simul. Multiconf.*, 2012, pp. 1–6.
[37] R. S. Cooper, *The Designer's Guide to Analog & Mixed-Signal Modeling*. Freront, CA, USA, Abvant! Corporation, 2001.
[38] P. Vas, "Electrical machines and drives: A space-vector theory approach," in *Monographs in Electrical and Electronic Engineering*. Oxford, U.K.: Oxford Science/Clarendon, 1993.
[39] F. Adler, A. Benigni, H. Stagge, A. Monti, and R. W. De Doncker, "A new versatile hardware platform for digital real-time simulation: Verification and evaluation," in *Proc. IEEE 13th Workshop Control Modeling Power Electron.*, 2012, pp. 1–8.
[40] F. Adler, H Stagge, and R. W. De Doncker, "A versatile modular hardware platform for distributed large-scale real-time simulation," in *Proc. Int. Power Electron. Motion Control Conf.*, 2012, pp. 1–6.

Andrea Benigni (S'09–M'14) received the Ph.D. degree from RWTH-Aachen University, Aachen, Germany, in 2013, and the M.Sc. and B.Sc. degrees from Politecnico di Milano, Milan, Italy, in 2008 and 2005, respectively.

From 2009 to 2013, he was a Research Associate at the Institute for Automation of Complex Power System, E.ON Energy Research Center, Rhenish-Westphalian Technical University (RWTH Aachen University), Aachen, Germany. He is currently an Assistant Professor with the Department of Electrical Engineering, University of South Carolina, Columbia, SC, USA.

Antonello Monti (M'94–SM'02) received the M.S. and Ph.D. degrees in electrical engineering from Politecnico di Milano, Milan, Italy, in 1989 and 1994, respectively.

From 1990 to 1994, he was with the Research Laboratory of Ansaldo Industria of Milan, where he was involved in designing the digital control of a large power drives. In 1995, he joined the Department of Electrical Engineering, Politecnico di Milano as an Assistant Professor. From 2000 to 2008, he was an Associate Professor and then a Full Professor with the Department of Electrical Engineering, University of South Carolina, Columbia, USA. He is currently the Director of the Institute for Automation of Complex Power Systems, E.ON Energy Research Center, Rhenish-Westphalian Technical University (RWTH Aachen University), Aachen, Germany.