Dynamic Variable Time-Stepping Schemes for Real-Time FPGA-Based Nonlinear Electromagnetic Transient Emulation

Zhuoxuan Shen, *Student Member, IEEE*, and Venkata Dinavahi, *Senior Member, IEEE*

***Abstract*—Electromagnetic transient (EMT) simulation of nonlinear elements in power systems is a particular challenge due to the requirements of an accurate representation and an efficient solution. The existing real-time simulators utilize a piecewise linear representation along with a fixed time step for the solution of nonlinear elements. This paper proposes the detailed methodologies for applying variable time stepping to real-time EMT simulation to improve the simulation accuracy and efficiency. The challenges, the feasible solutions, and corresponding restrictions of applying various variable time-stepping schemes along with nonlinear element solution methods in real time are discussed. The offline simulation and the real-time hardware emulation of two case studies, a full-bridge diode circuit and a power transmission system, are presented. The case studies were implemented on the field-programmable gate array device (Xilinx Virtex-7 XC7VX485T) in real time using high-level synthesis tool to achieve a parallelized and pipelined hardware design with minimum coding effort. The real-time emulation results captured by an oscilloscope are validated against the offline simulation on Saber and PSCAD/EMTDC software tools.**

***Index Terms*—Field-programmable gate array (FPGA), high-level synthesis (HLS), iterative scheme, nonlinear element, parallel processing, power system simulation, real-time system, surge arrestor, variable time stepping.**

I. Introduction

ELECTROMAGNETIC transients (EMTs) occur frequently in power systems due to various reasons such as lightning strikes, switching surges, or power frequency overvoltages, which threaten the safety and reliability of power system equipment. The frequency range of such transients can vary from low-frequency oscillations (∼0.1 Hz), to switching overvoltages (50 Hz–20 kHz), to very fast surges as in lightning and current breaker restrike transients (0 kHz–50 MHz) [1]. The EMT simulation is essential to study the impact of their phenomena, and to design adequate insulation and protection strategies for the host power system, which often contains nonlinear elements such as surge arresters, magnetic saturation, hysteresis, switches, and power electronic devices [2].

Since the transients can happen in just few tens of microseconds and the overvoltages or overcurrents can be multiple times the corresponding ratings, small time steps are required for the EMT simulation to accurately capture the transient behavior. However, for the simulation of a normal steady-state operation, a small time step is not necessary for observation and can be a great computational burden for optimum simulation speed. Currently existing offline EMT simulation tools such as ATP, PSCAD/EMTDC, EMTP-RV, etc., all employ a fixed user-defined simulation time step [3]–[5]. The concept of the variable time step, which is not new, has been traditionally used in many device-level power electronics circuit simulation tools such as PSpice, HSPICE, Saber [6]–[8]. Such simulation tools often use iterative methods, like Newton–Raphson (N-R) combined with various dynamic time-stepping schemes to solve the nonlinear circuit system. It is noted that some EMT-type software tools have already used iterative schemes with a fixed time step, such as EMTP-RV.

However, in the context of real-time simulation, which is a powerful tool for hardware-in-the-loop testing, the fixed time step is exclusively used in commercial EMT-type simulators due to the following main reasons.

1) The real-time simulators are often required to interface with external device under test (DUT) such as a control system or a protection system. The sampling process of the interface signals, such as assigned instantaneous voltage and current values collected by control and protection system, or the gating signals of switching devices received by the simulator, is inherently required at fixed time intervals.
2) Computation efforts are large and are unable to be predicted precisely for iterative and variable time-stepping schemes. The iterative methods used in device-level circuit simulation tools often have less stability, and the solution may not converge under certain circumstances.

Nevertheless, imperfectness exists in the interface process between the real-world power system and the control and protection system, which includes communication delay, analog-to-digital conversion delay, drive circuit delay, the reaction and process time of execution devices, etc. These inherent delays are considered in the design of control systems. A small amount

Manuscript received May 7, 2016; revised August 18, 2016 and October 24, 2016; accepted December 8, 2016. Date of publication January 16, 2017; date of current version April 10, 2017. This work was supported in part by the Natural Science and Engineering Research Council of Canada and in part by the China Scholarship Council.

The authors are with the Department of Electrical and Computer Engineering, University of Alberta, Edmonton, AB T6G 2V4, Canada (e-mail: zshen@ualberta.ca; dinavahi@ualberta.ca).

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/TIE.2017.2652403

of delay for the real-time simulator is tolerable depending on the system type and control purpose, varying from tens of microseconds to a few milliseconds. Based on this fact, variable time-stepping schemes are implementable within a fixed longer sampling period for real-time simulation.

In many academic research projects, the real-time simulator often works in a standalone mode, where the control algorithm is built inside the simulator along with the host system model. In some industrial training applications, the real-time simulator is used to train and test the response of the operators under certain circumstances. For such applications, reliable and accurate results and overall real-time performance are required over the simulation duration; however, it is not necessary to ensure the real-time performance for all time instances.

At this point, the concept of hard real time, firm real time, and soft real time, which originally comes from the area of general real-time computing, is discussed within the scope of power system real-time simulation [9]. The hard real-time applications refer to those where real-time constraint must be met in every simulation time step; therefore, the fixed time step must be used. The firm real-time applications refer to those where the interface sampling period is significantly larger than the minimum simulation time step; however, it is still less than a certain value, such as 1 ms, where a variable simulation time step can be used restrictedly. The soft real-time applications refer to those that do not require hardware interfacing or have a very large interface sampling period, where variable time-stepping schemes can be used fully or with minimum restriction. The usage of firm and soft real time does not necessarily mean the degradation of the simulation results, but the emphasis on the different functions and purposes. As later demonstrated in this paper, with the same computation capability, fast transients can be better presented if a variable time-stepping scheme is applied.

This paper proposes methodologies for using the variable time stepping to real-time EMT simulation. Various combinations of nonlinear solution methods and dynamic time-stepping schemes are explored in detail, revealing the challenges and corresponding characteristics. This paper discusses the restrictions applied for the appropriate time-step range, changing criteria, and the measurements to make variable time stepping more efficient in the real-time simulation. The improved accuracy and efficiency for using variable time step are demonstrated with two case studies: a diode full-bridge circuit employing a physics-based diode model showing a reverse recovery phenomenon and a three-phase power transmission system with nonlinear surge arresters.

A field-programmable gate array (FPGA) is chosen as the platform in this work, since it provides numerous reconfigurable logic slices and rich I/O resources, where full parallelism can be exploited to significantly accelerate the computation speed [10]. The FPGA is currently adopted in many applications [11]–[19], including real-time EMT simulation [20]–[37]. One major disadvantage of the FPGA in the circuit simulation application is its high design effort using a hardware description language (HDL). However, this problem is relieved by recently developed high-level synthesis (HLS) technology [38], where software programming languages, such as C/C++, are used for the FPGA design. The case studies were emulated on the Xilinx Virtex-7 XC7VX485T FPGA device using the proposed time-stepping schemes.

This paper is organized as follows. Section II discusses some major nonlinear element solution methods, variable time-stepping schemes, the feasible combinations, and the restrictions and corresponding measurements for the real-time simulation. Section III presents the circuit structure, major element models, and offline simulation results of two case studies. Section IV briefly describes the real-time emulation applications on an FPGA, introduces HLS, and then presents the FPGA implementation and the real-time emulation results of the case studies, followed by the conclusions in Section V.

II. Nonlinear Element Solution Methods and Dynamic Time-Stepping Schemes

The selection of the time-stepping scheme is related to the nonlinear element solution method, which provides certain variables for the time-step control algorithm. Therefore, the combinations can be various, but not arbitrary. In this section, several widely used nonlinear element solution methods and time-stepping schemes are introduced [2]–[8], [39]–[43]. The challenges of applying variable time stepping for the real-time simulation and the mitigation solutions are then elaborated.

A. Nonlinear Element Solution Methods

1) N-R Method: Applying the N-R method to solve a general nonlinear circuit using nodal analysis, the matrix equation is giving by

$$\mathbf{G}(\mathbf{v}^k)\mathbf{v}^{k+1} = \mathbf{I}_{\text{eq}}(\mathbf{v}^k) \tag{1}$$

where $\mathbf{v}$ is the unknown node voltage vector, $\mathbf{I}_{\text{eq}}$ is the equivalent current source vector, k is the iteration number, and $\mathbf{G}$ is the Jacobian matrix for conductances, given by

$$\mathbf{G}(\mathbf{v}^k) = \begin{pmatrix} \frac{\partial i_1}{\partial v_1}\big|_{\mathbf{v}^k} & \frac{\partial i_1}{\partial v_2}\big|_{\mathbf{v}^k} & \cdots & \frac{\partial i_1}{\partial v_n}\big|_{\mathbf{v}^k} \\ \vdots & \vdots & & \vdots \\ \frac{\partial i_n}{\partial v_1}\big|_{\mathbf{v}^k} & \frac{\partial i_n}{\partial v_2}\big|_{\mathbf{v}^k} & \cdots & \frac{\partial i_n}{\partial v_n}\big|_{\mathbf{v}^k} \end{pmatrix}. \tag{2}$$

The computation effort of the N-R method is relatively large, because of the iteration process, which often involves the matrix refactorization. The complex process of the N-R method, on the other hand, provides various convergence information, such as iteration counts and local truncation errors (LTEs), which can be used as the criteria for time-stepping schemes.

2) Piecewise Linearization (PL) Method: This method, also known as the pseudononlinear method, uses piecewise linearized segments to fit the nonlinear curve. The segment is determined at the beginning of each time step based on the node voltages of previous time instance, which may cause the problem of overshoot. This method is less accurate, however faster, compared with other iteration methods, which is often used by conventional real-time simulators.

3) Piecewise N-R (PNR) Method: The N-R method uses continuous functions to describe the nonlinear elements, while the PNR method instead uses piecewise linearized

functions, which simplifies the nonlinear function resolving process and lowers the convergence criteria. The PNR method can also be seen as the iterative version of the PL method. During fast transients, iteration is necessary to locate the correct piecewise segment without a delay, which significantly increases the result accuracy.

B. Dynamic Time-Stepping Schemes

1) LTE Method: The LTE is generated locally from the last time t_{n-1} to the current time t_n due to the approximation of discrete integration schemes for dynamic elements, which can be estimated by the Taylor series, given as

$$\text{LTE} = C(\Delta t)^{p+1} x^{(p+1)}(t_n) + 0((\Delta t)^{p+2}) \tag{3}$$

where C is the coefficient determined by the Taylor series, Δt is the time step, and p is the order of the integration method. In simulation software, the predictor–corrector method is often used to estimate the LTE. Various predictor methods are available such as using interpolation polynomial estimate, given as

$$x^0_{n+1} = x_n + \sum_{k=1}^{p} \left(\prod_{j=0}^{k-1} (t_{n+1} - t_{n-j}) \right) x[t_n, ...t_{n-k}] \tag{4}$$

where

$$x[t_n, ..., t_{n-k}] = \frac{x[t_n, ..., t_{n-k+1}] - x[t_{n-1}, ..., t_{n-k}]}{t_n - t_{n-k}}, \tag{5}$$

$$x[t_n] = x_n. \tag{6}$$

The predictor results are good initial guesses for the N-R iteration with order p or larger integration method. The error of the predictor and the final corrector result obtained by using the N-R iteration is used to estimate the LTE, which then determines the time step of next time instance. The scheme can be applied for a general m-dimensional nonlinear system by simply using the norm or maximum LTE of all variables.

2) Iteration Count Method: This method simply records the iteration count for each time instance. If many iterations are required to obtain the convergent results, the time step will decrease for future time instances.

3) DVDT or DIDT Method: DVDT and DIDT are the derivatives of the node voltages and branch currents detecting the large changing rates. When a sudden large voltage or current surge appears in the circuit, small time steps are used to capture the transients more clearly and accurately.

The three time-stepping schemes mentioned above have the major focus on the LTE induced by dynamic element modeling, the degree of convergence for nonlinear elements, and the observations needed for accurate large-amplitude high-frequency transients, respectively. The circuit type, modeling method, and the simulation purpose all affect the selection of the time-stepping scheme and the corresponding threshold parameters. Another issue is the choice of the time-step changing rate, and multiple methods exist in the literature [41]–[43]. This paper uses multiplying or dividing by 2 as the changing rate for the time step, which is straightforward and has been adopted by Saber software [8], [39].

C. Combinations

Various combinations of nonlinear element solution methods and dynamic time-stepping schemes are feasible for real-time simulation. The DVDT or DIDT method can be used with all three above-mentioned nonlinear solution methods with or without iterations. It is noted that the combination of the PL method and the DVDT/DIDT method is compatible with an offline EMT-type software engine. Therefore, this combination could be helpful for boosting the offline simulation for some applications. Although the PNR method does require iterations for some time instances, however, in other circumstances, the nonlinear segment does not change, and the results can be solved without iteration, which is the same as the PL method. Therefore, the iteration count scheme is not available for PNR and PL methods. The LTE scheme is normally used with an N-R-like method, where highly nonlinear functions exist. Under such circumstances, the numerical stability is relied on the appropriate choice of time steps. Since PNR and PL methods generally have good numerical stability, the LTE scheme is not necessary, though the predictor–corrector procedures may still be applicable for some applications.

Since the computation effort for the N-R method is higher, the simulated circuit scale is smaller than those using other simpler methods. The smaller circuit using the N-R method can be cooperated with the other larger linear circuit by using various circuit separation schemes, such as introducing delays, using transmission line model for connection. The connected larger circuit can use the fixed time step with the interface similar to the one used for the external control system.

D. Restrictions and Measurements for Real-Time Simulation

The relatively large computation effort for iterative schemes and the requirement of a fixed interface sampling rate are major challenges for applying variable time-stepping schemes. Some restrictions and measurements are applied to mitigate the issues, listed as follows.

1) Limit the Range of Time Steps: The smaller range, especially limiting the maximum time step, can improve the convergence property of the iteration schemes and reduce the chance of rejecting solutions since large LTE or iteration counts are often caused by using large time steps. To maximize the usage of the computation capability, the variable time steps shall be changed around the maximum fixed time step, which ensures the real-time performance. The maximum time step cannot exceed the interface sampling period, while the minimum time step is restricted by the computation capability.

2) Apply Conservative Time-Step Changing Criteria and Avoid Frequent Time-Step Changing: The time-step changing criteria are referred to the threshold values of the LTE, iteration counts, or DVDT. The conservative threshold criteria choose smaller time steps, which may increase the total time instances. However, because of the smaller time steps, the total iteration counts tend to be smaller, and the chance of solution rejection is smaller. Combined with the limited time-step

range, the computational effort is relatively stabler and easier to predict during the simulation.

3) Use Smaller Maximum Iteration Count Number: The purpose of this restriction is to limit the computation burden for a time instance. It is particularly useful for the PNR method with numerical stable applications, where solution rejection is not necessary, since limiting iterations just affects the result accuracy. For such applications, the extreme case of applying this restriction is the same as using the PL method.

4) Calculate the Constants for Different Time Steps in Advance: When applying smaller range of time steps and using a constant changing rate for the time step, there are only several determined time steps during the simulation. Before the simulation begins, the constants and the conductance matrices for all these time steps are calculated and partially decomposed. When the time step is changed, these precalculated constants are loaded to the solver directly without recalculation. The loading process can be extremely fast for the FPGA with merely one clock cycle delay.

5) Use the Foreknown Transient Information: Some test conditions, such as faults and surges, are often preset to the simulator and triggered at a determined time instance. This knowledge can let the simulator set the time step to a minimum value just before the condition happens, since high-frequency transients may occur. This measurement will not affect testing results, since the DUT, which can be a controller, is not aware of any test conditions in advance.

III. Offline Simulation of Two Case Studies

Two case studies, a diode full-bridge circuit (Case 1) and a power transmission system (Case 2), are presented in this section, using different time-stepping schemes. The parameters of the case studies are listed in the Appendix. The offline circuit simulation programs for the case studies are written in MATLAB script, where the restrictions and measurements for the real-time simulation are considered. The result accuracy is verified by comparing with Saber or PSCAD/EMTDC software. The solution combinations with the best performance are then implemented on the FPGA and run in real time.

A. Diode Full-Bridge Circuit

1) Diode Model Description: The topology of a diode bridge circuit is shown in Fig. 1, where a physics-based nonlinear p-i-n diode model including accurate reverse recovery phenomenon is used [44], given as

$$i(t) = \frac{q_E(t) - q_M(t)}{T_M} \tag{7}$$

$$0 = \frac{dq_M(t)}{dt} + \frac{q_M(t)}{\tau} - \frac{q_E(t) - q_M(t)}{T_M} \tag{8}$$

$$q_E(t) = I_S \tau \left[e^{\frac{v(t)}{nV_T}} - 1 \right] \tag{9}$$

where $q_E(t)$ is the junction charge variable, $q_M(t)$ is the charge in the middle of the intrinsic region, $i(t)$ and $v(t)$ are the current and voltage across the diode, respectively, T_M is the diffusion

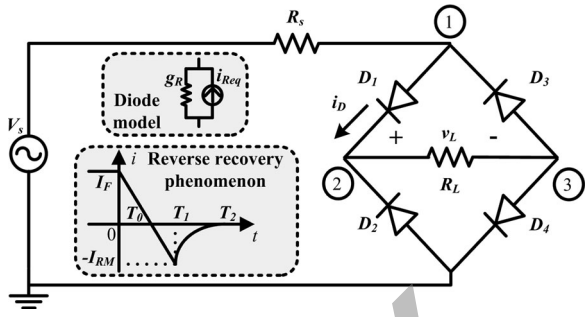

Fig. 1. Circuit topology, the diode model, and the reverse recovery phenomenon for the diode full-bridge circuit case study.

transit time, τ is the carrier lifetime, I_S is the diode saturation current constant, V_T is the thermal voltage, and n is 2 for high-level injection. Fig. 1 presents the equivalent circuit model of the p-i-n diode and the reverse recovery phenomenon from T_0 to T_2 with the peak value I_{RM} at T_1. Since the diodes are modeled by highly nonlinear and relatively complex functions, the N-R method is chosen to obtain the results accurately. The trapezoidal rule is applied to discretize and linearize the differential equation (8), given as

$$q_M(t) = \frac{\Delta t}{2T_M(1+k_1)} \cdot q_E(t) + \frac{q_{\text{hist}}(t-\Delta t)}{1+k_1} \tag{10}$$

where

$$q_{\text{hist}}(t-\Delta t) = q_M(t-\Delta t) \cdot (1-k_1) + \frac{\Delta t}{2T_M} \cdot q_E(t-\Delta t) \tag{11}$$

$$k_1 = \frac{\Delta t}{2}\left(\frac{1}{\tau} + \frac{1}{T_M}\right). \tag{12}$$

The time step Δt for the calculation of $q_M(t)$ and $q_{\text{hist}}(t-\Delta t)$ can be different when applying dynamic time-stepping schemes. Using the N-R method, the equivalent conductance g_R and the current source i_{Req} are given as

$$g_R = \frac{\partial i_R}{\partial v(t)} = k_2 I_S \tau \cdot \frac{1}{nV_T} \cdot e^{\frac{v(t)}{nV_T}} \tag{13}$$

$$i_{\text{Req}} = k_2 I_S \tau \left[e^{\frac{v(t)}{nV_T}} - 1 \right] - k_3 q_{\text{hist}}(t-\Delta t) - g_R \cdot v(t) \tag{14}$$

where

$$k_2 = \frac{1}{T_M} - \frac{\Delta t}{2T_M^2(1+k_1)} \tag{15}$$

$$k_3 = \frac{1}{T_M(1+k_1)}. \tag{16}$$

2) Offline Simulation Results: The offline circuit simulation results for the diode bridge circuit using LTE and iteration count schemes based on the N-R method are shown in Fig. 2(b) and (c). Since the voltage and current waveforms do not change abruptly, DVDT and DIDT methods are not effective for this application. The offline simulation results, including the diode current i_D for D_1 and load voltage v_L, are compared with

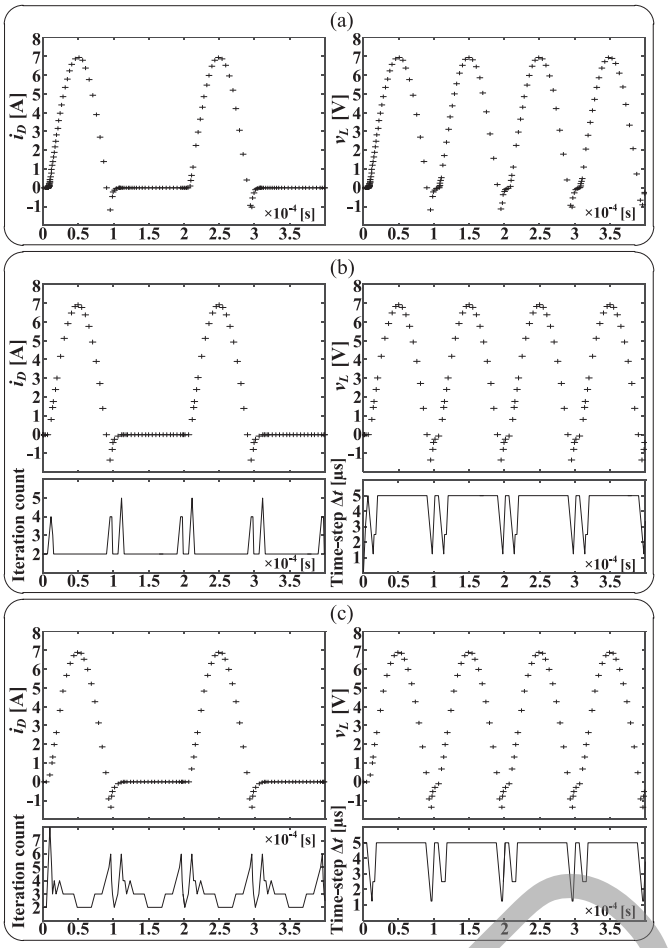

Fig. 2. Offline results of diode current i_D for D_1, load voltage v_L, iteration count, and time step Δt. (a) Saber. (b) LTE scheme based on the N-R method. (c) Iteration count scheme based on the N-R method.

Saber [see Fig. 2(a)], where the power diode model is employed. All the current and voltage waveforms have sufficient accuracy clearly showing the diode reverse recovery phenomenon. The time steps for both LTE and iteration count schemes are limited within the range of 1.25–5 μs, while the minimum time step is not limited for Saber result. By limiting the range, the calculated points are reduced, while the accuracy is preserved. Using a fixed small time step to limit the LTE for all points ensuring the sufficient accuracy can be computationally very inefficient. Compared with the iteration count scheme, the LTE scheme utilizes less total iteration counts; therefore, it is chosen to be implemented on the FPGA.

B. Power Transmission System

1) Circuit Structure: The three-phase power transmission system shown in Fig. 3 is simulated to observe the transients of the lightning strike, occurring in the middle of a transmission line. On the generator side, an equivalent voltage source V_s in series with a resistor R_s is connected to a Y–Y-connected transformer T. The series-connected inductor L_L and resistor R_L are used to represent the load, connected

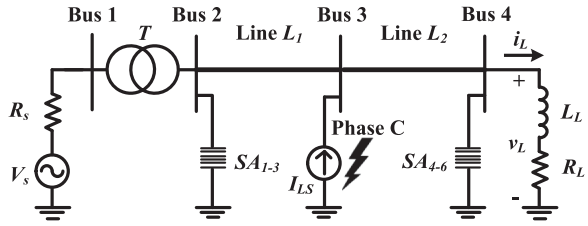

Fig. 3. Single-line diagram for the power transmission system case study.

through a 100-km transmission line using the Bergeron line model (distributed parameter traveling wave model). To determine the history current value $i(t-\tau)$ for the Bergeron line model with the variable time-stepping scheme, the exact time of the history current value is required to be stored. The $i(t-\tau)$ can be obtained by linear interpolation of the two history current values between $t-\tau$. The six gap-less surge arresters SA_{1-6} are connected to the both sides of the transmission line, which is separated into two segments L_1 and L_2, with the lightning directly hitting phase C represented by a current source I_{LS}.

2) Surge Arrester Model and Lightning Surge Waveform: A metal–oxide surge arrester is commonly used to suppress the overvoltage caused by lightning, switching, faults, etc. [45]. The PL method is applied to model the surge arrester, consuming less computation effort than using a continuous nonlinear function. The segment data for surge arresters are presented in the Appendix.

The standard 8/20-μs current surge testing waveform is used in this work, given as [46]

$$I_{LS}(t) = AI_p t^k \exp(-t/\tau) \tag{17}$$

where $A = 0.01243\,(\mu\text{s})^{-3}$, $k = 3$, $\tau = 3.911\,\mu$s, and I_p, the maximum value of the surge current, is 5000 A.

3) Offline Simulation Results: PNR and PL methods combined with the DVDT scheme are utilized for the offline power transmission system. The lightning current surge starts from exactly 10 ms of the simulation. The results are compared with PSCAD/EMTDC using fixed time steps, shown in Fig. 4. By changing the time-step value from 5 to 10 μs in PSCAD/EMTDC, the result accuracy is significantly reduced, which can be observed by comparing the maximum values of overvoltages and overcurrents. When using PNR with various time steps between 5 and 20 μs, the results are very close to the one from PSCAD/EMTDC with 5 μs as the time step. If the PL scheme is used, the results from variable time stepping are no longer accurate; however, they are still better than the PSCAD/EMTDC results with 10 μs. Since the transient frequency is very high, the change of time steps can pose great impact on the result accuracy. In this application, by using dynamic time stepping, the computation effort is similar to the one using 10 μs as the fixed time step, while results with the PNR method are more accurate, which are close to the PSCAD/EMTDC results using 5 μs as the fixed time step. Admittedly, the small-amplitude high-frequency transients are somewhat diminished, which is due to the DVDT algorithm used for the application.

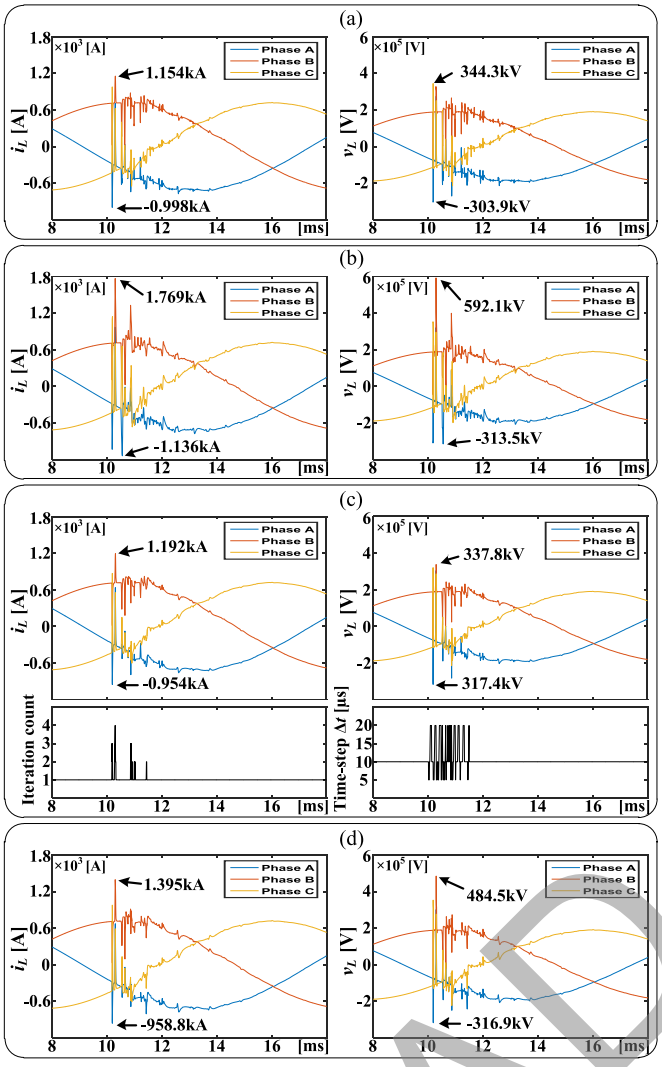

Fig. 4. Offline results of load current i_L, load voltage v_L, iteration count, and time step Δt. (a) PSCAD/EMTDC with a fixed time step 5 μs. (b) PSCAD/EMTDC with a fixed time step 10 μs. (c) DVDT scheme based on the PNR method. (d) DVDT scheme based on the PL method.

For hardware implementation, the PNR method is chosen for its better accuracy.

IV. FPGA Implementation and the Real-Time Emulation Results

FPGAs provide a large amount of configurable hardware slices, which are beneficial for tasks requiring periodic and parallel processing. FPGAs have been extensively used in industrial detection and control applications [12]–[19]. FPGAs have also been explored and employed for real-time emulations in both literature and industry for detailed modeling of various equipment such as transformers, machines, power electronics, and large-scale systems [20]–[37]. The existing research studies cover the study of solution methods, the linear and nonlinear element modeling, corresponding hardware design schemes, etc. In commercial real-time simulators, FPGAs are used as the standard acceleration unit for modular multilevel converter calculation.

A. High-Level Synthesis

Traditionally, FPGA programmers needed to work on both designing the algorithm of specific tasks and the complex digital circuits in the register-transfer-level structure using the HDL, which is time consuming and error prone. HLS, provided by Xilinx, translates the C/C++ language to the HDL with a highly paralleled hardware structure [38]. HLS supports the arbitrary data precision and provides abundant directives for optimization, such as loop unroll, array partition, pipelining, etc. Since the C/C++ language has much higher abstraction than the HDL, the coding effort is substantially reduced.

However, some restrictions are applied for HLS, for example, system clock-based units, where outputs are directly decided or affected by the clock signals, such as counters, are unable to write with the C/C++ language, and smaller pieces of the C/C++ code instead of the entire project can be synthesized more efficiently. Therefore, mix usage of HLS and HDL coding is applied for the implementation of the real-time emulator.

B. FPGA Implementation

The hardware implementation and data flow of the two case studies are presented in Fig. 5. The cells in dashed-line boxes are implemented for the diode full-bridge case study and the power transmission system case study, respectively. For other parts, the two case studies share the same hardware structure.

The finite-state machine (FSM) of the real-time emulator controls the overall emulation flow by interfacing with other modules through control signals, and the state chart is shown in Fig. 6. It receives the exact time t_{real} from the real-time counter and communicates with another internal FSM in the real-time I/O control module, which manages the I/O data transfer between the emulator and the external devices. When the reset signal is detected, the state changes from S_0 to S_1, starting the initialization module, which calculates the constants for all time steps. These constants are processed from user-defined circuit parameters to avoid unnecessary repetitive calculations, and some of them can be applied directly to the conductance matrix entries, such as the conductance of a linear resistor. Taking the example of the nonlinear diode, the expression $k_2 I_s \tau$ is a typical Δt-dependent constant, while $1/nV_T$ is a precalculated constant without such dependence in (13) and (14). The time-step Δt-dependent constants are stored in the Δt-dependent constant memory connecting to a multiplexer, while others are transferred to the corresponding modules directly.

After the initialization and loading process, the FSM then enters to the repetitive processes from S_2 to S_8, which marks the start of the real-time emulation enabling t_{real}. In S_2, the Δt-dependent constants are reloaded to the electrical source module, linear element module, and nonlinear element module through the multiplexer with Δt as the selection signal.

Once S_2 is done, all the cells in the linear element module and the electrical source module start simultaneously in S_3, which execute only once for one time step. These cells calculate the conductance entries and the current terms of various elements, similar to the g_R and i_{Req} in (13) and (14), but for other sources and linear elements, such as the transformer and the transmission

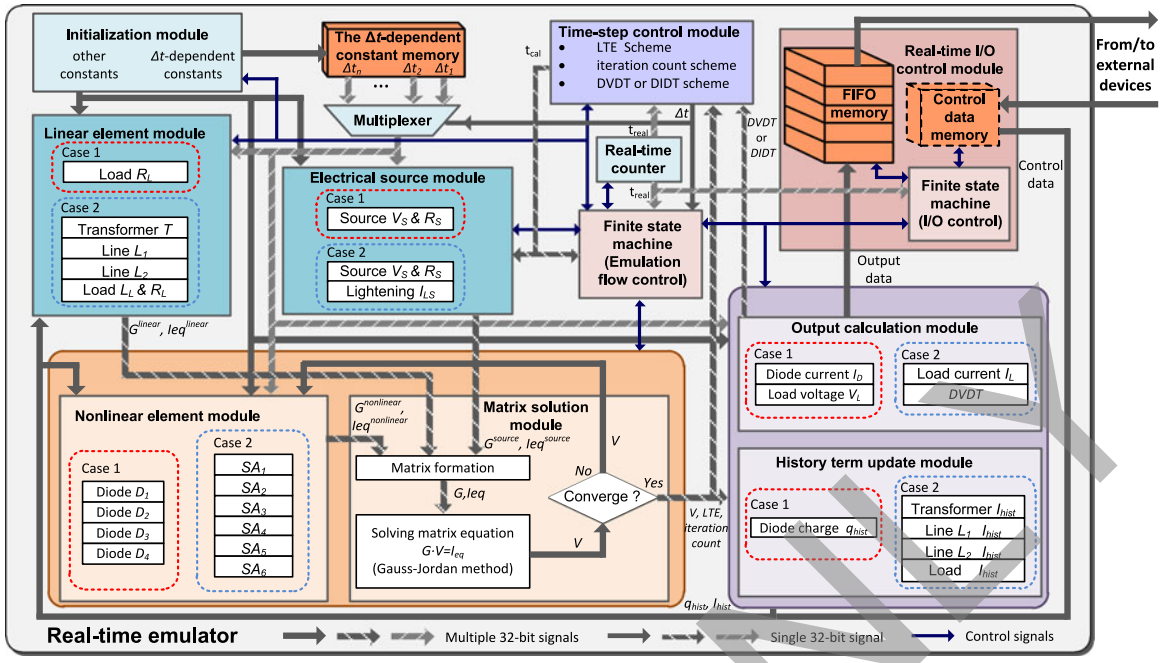

Fig. 5. Hardware implementation and data flow of the real-time emulator for the case studies.

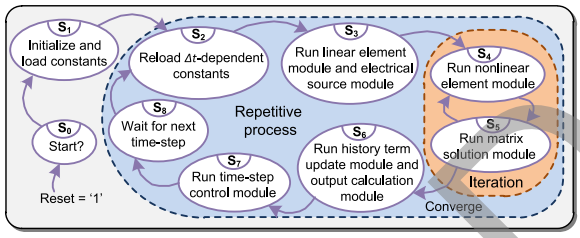

Fig. 6. State chart for real-time emulation flow control.

lines, using the trapezoidal rule for discretization. Then, in S_4, the equivalent conductances and current terms for nonlinear elements are calculated, which are exactly the g_R and i_{Req} of the diodes for Case 1. For nonlinear elements, iterations between S_4 and S_5 may be required, and g_R and i_{Req} need to be recalculated several times until the results converge.

The matrix solution module running in S_5 is composed of three sequential processes: matrix formation, matrix solution, and convergence judgment. Matrix formation is applied adding the conductances and current terms of multiple elements connecting to the same node, and forming the matrix $\mathbf{G}$ and vector $\mathbf{I}_{eq}$, which are then solved by the Gauss–Jordan method. For the matrix solver, LU factorization, conventional Gauss elimination, and Gauss–Jordan method could all be the candidates for the real-time emulator [47]. LU factorization is very useful when the conductance matrix G does not change over time. However, when nonlinear elements contained, the factorization may need to be reapplied if any element of G changes. In such a case, LU factorization is less efficient than simply using Gauss elimination, since the LU method requires two sequential substitution processes (forward and backward substitutions). Both Gauss elimination and Gauss–Jordan methods include the forward elimination and backward substitution processes. The Gauss–Jordan method can effectively parallel the two processes, which always run completely for every solution calculation, though the total sequential computation effort is slightly higher. For the purpose of achieving highest parallelism and lowest latency, the Gauss–Jordan method is applied for the FPGA-based implementation. Then, the matrix solution module conducts the convergence judgment by comparing the node voltages or the segments of surge arresters of the last two sets of results in Case 1 and Case 2, respectively.

Once the converged results are obtained, the history term update module and the output calculation module runs simultaneously in S_6. The history term update module calculates some variables that will be used for future calculations for dynamic elements, such as the q_{hist} in (11) for diode. Various outputs, such as the currents of the diodes in Case 1, require further calculation, which is accomplished by the output calculation module. In S_7, the next Δt is calculated in the time-step control module.

When the next time instance comes, the state changes from S_8 to S_2, and the computation for the next time step begins. Due to the usage of variable time stepping, S_8 may go through without actual waiting, when the delay time t_{delay}, the difference between t_{real} and the time for the current calculated solution t_{cal}, is positive.

The algorithms of the time-step control module for the case studies are shown in Fig. 7 in the form of a pseudocode. The time steps can be either limited to a range between the maximum value $\Delta t_{\max}$ and the minimum value $\Delta t_{\min}$ or predetermined with several choices. For the diode full-bridge circuit, the LTE is compared with the threshold value $\mathrm{LTE}_{\mathrm{th1}}$ and $\mathrm{LTE}_{\mathrm{th2}}$ to determine the next time step. While for the power transmission system, the time step is not only decided by DVDT and the threshold

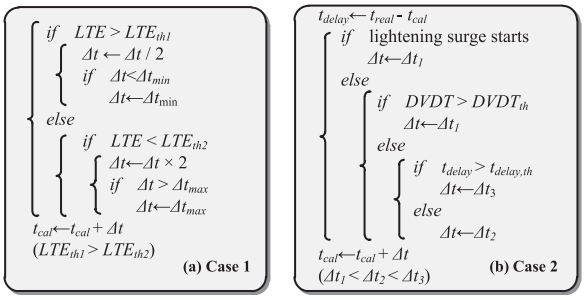

Fig. 7. Pseudocode for the time-step control module. (a) Diode full-bridge circuit. (b) Power transmission system.

value $\mathrm{DVDT_{th}}$, but also the delay time t_{delay} and the threshold value $t_{\mathrm{delay,th}}$. When the computation time exceeds the minimum time step, a large time step will be applied to balance the overall computation time ensuring real-time performance. Due to numerical stability consideration, this strategy is not applied for the diode full-bridge case study. For the diode full-bridge case study, the results can be calculated slightly ahead of real time, when interfacing is not required (soft real-time condition). In other words, t_{delay} can be negative, while the results are sent out in exact real time.

The real-time I/O control module synchronizes and exchanges data between the emulation system and external devices, such as monitors and control and protection systems. For the case studies in this paper, the external device is a four-channel oscilloscope connected by a 16-bit digital-to-analog converter (DAC). The first-in first-out (FIFO) memory receives the results and sends them out during the next interfacing period. The data of several time instances will be available during one interfacing period, and the external devices can receive the data of either one time instance or all time instances with the delay maximum of up to one interfacing period based on the setting in the internal FSM. A control data memory can be added to the emulator, which is not implemented for the two case studies. When the control data arrive, it will be first registered and remain the same for the interfacing period. It will then be loaded to other modules when the calculation for the next time step starts in $\mathrm{S_2}$. For the case studies, the FSM for the I/O control module only has two states: the idle state and the real-time output data sending state, which can be expanded for controller interface or other user-defined features. Once the results of the output calculation module are ready, they are sent to the FIFO memory, which is not controlled by the FSM of the real-time I/O control module.

The initialization module, the linear element module, the electrical source module, the nonlinear element module, the matrix solution module, the history term update module, and the output calculation module, shown in Fig. 5, are synthesized by HLS and then packaged as user-designed IP cores, respectively. These modules generally include the relatively complex algorithm, and the design effort can be much lowered by C programming. The real-time counter and the real-time I/O control module are designed by the VHDL code for their strict timing requirement. The FSM (emulation flow control), the Δt-dependent constant memory, and the multiplexer have their own fixed or classic hardware structures and can be designed by the VHDL code very efficiently and easily. IEEE 32-bit floating-point number precision is applied for the hardware design. All these modules and structures are assembled with VHDL coding and then implemented generating the configuration bit stream.

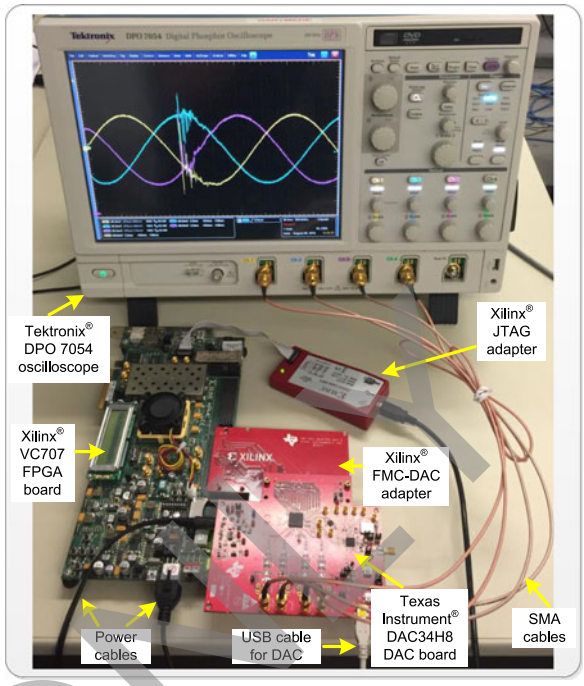

Fig. 8. Hardware setup for the FPGA-based real-time emulation.

TABLE I
HARDWARE RESOURCE UTILIZATION FOR DIODE FULL-BRIDGE CIRCUIT AND POWER TRANSMISSION SYSTEM CASE STUDIES

Resource	Diode full-bridge circuit	Power transmission system
LUT	51043 (16.81%)	62495 (20.58%)
LUTRAM	903 (0.69%)	539 (0.41%)
FF	32678 (5.38%)	63804 (10.51%)
BRAM	7 (0.68 %)	102 (9.90%)
DSP	344 (12.29%)	327 (11.68%)

C. Hardware Setup, Resource Utilization, and Latency

The hardware setup for the case studies are presented in Fig. 8. The bit stream was downloaded to the Xilinx VC707 board using the Virtex-7 XC7VX485T FPGA through the JTAG adapter. The case studies were emulated on the Xilinx Virtex-7 XC7VX485T FPGA at the 100-MHz clock frequency, which contains 303 600 lookup tables (LUTs), 130 800 LUTRAMs, 607 200 Flip Flops (FFs), 1030 block RAMs (BRAMs), 2800 DSP slices, etc. The hardware resource consumption for the case studies is presented in Table I. The outputs from the FPGA board were converted to analog signals by the Texas Instrument DAC34H8 DAC board [48], which requires basic configuration setting. The Tektronix DPO 7054 oscilloscope was used to capture and present the analog signals from the DAC board.

The latencies of various states and the total latency for the two case studies are presented in Table II. The latencies for adder/subtracter, multiplier, and divider are four, four, and nine

TABLE II
STATE LATENCY FOR DIODE FULL-BRIDGE CIRCUIT AND POWER TRANSMISSION SYSTEM CASE STUDIES

State	Module	Case 1	Case 2
S_1	Initialization module	0.36 μs	1.54 μs
S_2	Δt-dependent constant memory and multiplexer	0.01 μs	0.01 μs
S_3	Linear element module	0.01 μs	2.27 μs
	Electrical source module	0.16 μs	0.43 μs
S_4	Nonlinear element module	0.29 μs	0.05 μs
S_5	Matrix solution module	0.81 μs	2.8 μs
S_6	History term update module	0.4 μs	0.86 μs
	Output calculation module	0.4 μs	0.19 μs
S_7	Time-step control module	0.16 μs	0.28 μs
	Total Min. latency (iteration number)	2.93 μs (two)	6.27 μs (one)
	Total Max. latency (iteration number)	5.13 μs (four)	14.82 μs (four)
	Total CPU-based Min. latency (iteration number)	10.86 μs (two)	24.34 μs (one)
	Total CPU-based Max. latency (iteration number)	17.38 μs (four)	68.47 μs (four)

clock cycles, respectively, for the case studies with 10 ns as the clock period. The latencies may vary for different modules, due to the module scale, and are also influenced by the FPGA device and the uncertainty of the clock period. Since S_1 only executes once before the repetitive process, it is, therefore, not taken into account in the total latency. For both S_3 and S_6, two modules runs concurrently, and the maximum latency of the modules is used as the latency of the corresponding state. The latency of the linear element module in Case 2 is relatively long, because it requires multiple comparisons for the transmission line model to find the correct history term and then conduct the linear interpolation. The maximum iteration number is four for both case studies, with the given parameters and test conditions. For the case studies, the average allocated computation times for one time step are 5 and 10 μs for the two case studies. In such a setting, the maximum delays due to the use of the variable time stepping and the iterative method are 9.1 and 75 μs, respectively. The minimum average computation times for a time step could be 3.9 and 8.2 μs, measured from the time scopes of 0.15–0.65 and 7–17 ms, for the two case studies. However, the maximum delay could be much larger, if using the minimum average computation time as the allocated value with no redundancy.

The CPU-based computation times for the case studies are also presented in Table II for the two case studies. The C program complied by Visual Studio 2012 was running on Intel Xeon E5-2609 CPU at 2.40 GHz with a Windows 7 operating system. With the same computer system, Saber took 0.172-s computation time for the simulation of 0.4 ms, which is 430 times slower than real time for Case 1, and PSCAD/EMTDC took 0.561-s computation time for the simulation of 0.02 s, which is 28 times slower than real time for Case 2.

D. Real-Time Emulation Results

Real-time emulation results for the case studies obtained from the oscilloscope are shown in Figs. 9 and 10, with the same parameters and test condition as the offline simulations. All calculated data points were captured and sent to the oscilloscope. Since the matrix solver and the data precision of the real-time emulator are different from the ones from MATLAB, the results are slightly different from the offline simulation. For both case studies, the real-time emulator used the iteration counts maximum up to four times for any time instance and did not reject any solution. The results from the

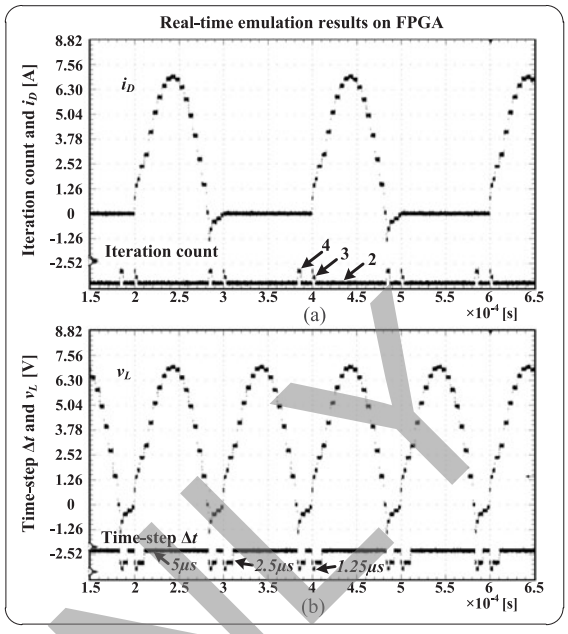

Fig. 9. Real-time emulation results on the FPGA captured by a oscilloscope for the diode full-bridge circuit case study. (a) Diode current i_D for D_1 and iteration count. (b) Load voltage v_L and time step Δt.

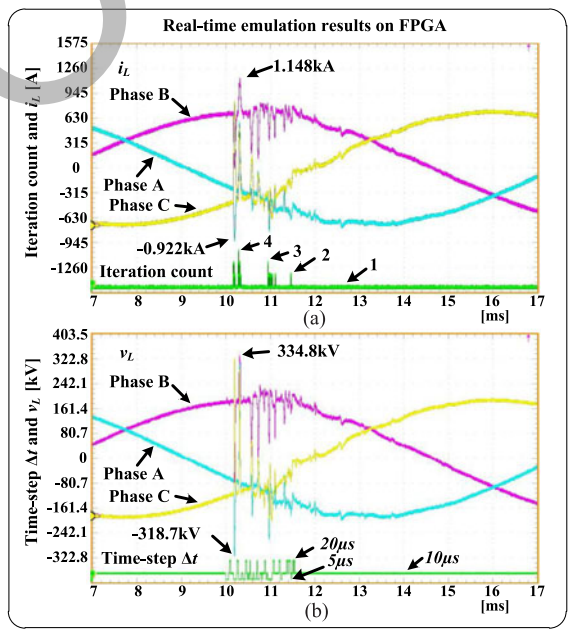

Fig. 10. Real-time emulation results on the FPGA captured by an oscilloscope for the power transmission system case study. (a) Load current i_L and iteration count. (b) Load voltage v_L and time step Δt.

hardware emulation have a good accuracy compared with the offline Saber and PSCAD/EMTDC, respectively.

E. Scalability

Scalability is composed of two major parts: the increase of sequential calculation and the parallelism capability and corresponding cost of a larger system.

During periodic calculation, the two major steps are the matrix element update (S_3, S_4, and S_6) and the matrix solution (S_5). The longest latency for the matrix element update is only dependent on the most complex single element, not the element number. For the EMT-type power system real-time simulation, the matrix is sparse and can be naturally decomposed to multiple smaller matrices through the existing transmission line model by properly numbering the nodes, which is typically used in commercial real-time simulators. The latency of the matrix solution is, therefore, decided by the size of the largest submatrix.

The iteration number is mostly affected by the nonlinearity of the element, modeling schemes, and the time-stepping schemes. The system size may also have an effect, since more nonlinear elements are involved. It can affect the stability for the N-R method, while only affect the result accuracy for the PNR method with a certain maximum iteration number.

The FPGA provides very high parallelism capability and data transfer bandwidth. The resource can be a major limitation for the hardware implementation of a large circuit system using the proposed variable time-stepping schemes; however, it can be resolved by using an advanced FPGA device, such as UltraSCALE+ XCVU440 containing 3780k logic cells and 12 288 DSP slices, or by using multiple interconnected FPGA boards [49].

V. Conclusion

Real-time emulation of nonlinear transients in power systems can be greatly enhanced in both accuracy and efficiency by adopting variable time steps. This paper presented multiple dynamic variable time-stepping schemes coupled with nonlinear element solution methods and discussed the strengths and limitations for the real-time emulation in detail. Two case studies were presented to illustrate the performance of the proposed variable time-stepping schemes and their efficiency in capturing nonlinear transients in both offline and real time. For optimum realization of paralleled hardware implementation and smaller coding effort, the FPGA and HLS were chosen to emulate the proposed schemes on hardware. Future work included extending the proposed schemes to simulate larger and more complex power systems with external control and protection systems.

Appendix

Parameters of Case 1: Vs amplitude (peak–peak), frequency, and R_s: 10 V, 5 kHz, and 1 mΩ. R_L: 1 Ω. I_s, τ, T_m, V_T, and n: 10^{-12} A, 10 μs, 5 μs, and 25.9 mV.

Parameters of Case 2: Base values: 200 MW and 60 Hz. $Vs(L-L)$ and Rs: 120 kV and 1 Ω. Transformer voltage, leakage inductance, and copper loss: 115 kV/230 kV, 0.16 p.u., and 0.004 p.u. Line segment L_1 and L_2 length: 50 km and 50 km. Line inductance: mode +: 0.9337 mH/km and mode 0: 4.1264 mH/km. Line capacitance: mode +: 12.74 nF/km and mode 0: 7.751 nF/km. L_L and R_L: 2 mH and 265 Ω. *V–I* characteristics for surge arresters: [(0 kV, 0 kA), (263.82 kV, 0.001 kA), (317.19 kV, 0.1 kA), (362.42 kV, 2.8 kA), (429.89 kV, 200 kA)].

References

[1] J. A. Martinez, "Parameter determination for power systems transients," in *Proc. IEEE Power Eng. Soc. Gen. Meeting*, 2007, pp. 1–6.

[2] H. W. Dommel, *EMTP Theory Book.* Portland, OR, USA: Bonneville Power Admin., 1984.

[3] ATP overview, (2016). [Online]. Available: http://www.emtp.org

[4] *EMTDC User's Guide: A Comprehensive Resource for EMTDC*, *Version 4.7*, Manitoba HVDC Research Centre, Winnipeg, MB, Canada, 2010.

[5] EMTP-RV overview, (2016). [Online]. Available: http://emtp-software.com/Overview

[6] *PSpice User's Guide*, Cadence Design Systems, Inc., San Jose, CA, USA, 2000.

[7] *HSPICE User Guide: Simulation and Analysis, Version B—2008.09*, Synopsys, Inc., Mountain View, CA, USA, 2008.

[8] *Saber User Guide, Version V-2004.06-SP1*, Synopsys, Inc., Mountain View, CA, USA, 2004.

[9] K. Shin and P. Ramanathan, "Real-time computing: A new discipline of computer science and engineering," *Proc. IEEE*, vol. 82, no. 1, pp. 6–24, Jan. 1994.

[10] *VC707 Evaluation Board for the Virtex-7 FPGA User Guide, UG885 (v1.6.1)*, Xilinx, Inc., San Jose, CA, USA, 2015.

[11] A. Dinu, M. N. Cirstea, and S. E. Cirstea, "Direct neural-network hardware-implementation algorithm," *IEEE Trans. Ind. Electron.*, vol. 57, no. 5, pp. 1845–1848, May 2010.

[12] E. Monmasson and M. N. Cirstea, "FPGA design methodology for industrial control systems—A review," *IEEE Trans. Ind. Electron.*, vol. 54, no. 4, pp. 1824–1842, Aug. 2007.

[13] M. W. Naouar, E. Monmasson, A. A. Naassani, I. Slama-Belkhodja, and N. Patin, "FPGA-based current controllers for AC machine drives—A review," *IEEE Trans. Ind. Electron.*, vol. 54, no. 4, pp. 1907–1925, Aug. 2007.

[14] M. N. Cirstea and A. Dinu, "A VHDL holistic modeling approach and FPGA implementation of a digital sensorless induction motor control scheme," *IEEE Trans. Ind. Electron.*, vol. 54, no. 4, pp. 1853–1864, Aug. 2007.

[15] O. Gulbudak and E. Santi, "FPGA-based model predictive controller for direct matrix converter," *IEEE Trans. Ind. Electron.*, vol. 63, no. 7, pp. 4560–4570, Jul. 2016.

[16] F. Xu, H. Chen, X. Gong, and Q. Mei, "Fast nonlinear model predictive control on FPGA using particle swarm optimization," *IEEE Trans. Ind. Electron.*, vol. 63, no. 1, pp. 310–321, Jan. 2016.

[17] Z. Hajduk, B. Trybus, and J. Sadolewski, "Architecture of FPGA embedded multiprocessor programmable controller," *IEEE Trans. Ind. Electron.*, vol. 62, no. 5, pp. 2952–2961, May 2015.

[18] J. J. Rodríguez-Andina, M. D. Valdés-Peña, and M. J. Moure, "Advanced features and industrial applications of FPGAs—A review," *IEEE Trans., Ind. Informat.*, vol. 11, no. 4, pp. 853–864, Aug. 2015.

[19] Ó Jiménez, Ó Lucía, I. Urriza, L. A. Barragán, and D. Navarro, "Analysis and implementation of FPGA-based online parametric identification algorithms for resonant power converters," *IEEE Trans. Ind. Informat.*, vol. 10, no. 2, pp. 1144–1153, May 2014.

[20] Y. Chen and V. Dinavahi, "An iterative real-time nonlinear electromagnetic transient solver on FPGA," *IEEE Trans. Ind. Electron.*, vol. 58, no. 6, pp. 2547–2555, Jun. 2011.

[21] Z. Shen and V. Dinavahi, "Real-time device-level transient electrothermal model for modular multilevel converter on FPGA," *IEEE Trans. Power Electron.*, vol. 31, no. 9, pp. 6155–6168, Sep. 2016.

[22] N. Roshandel Tavana, and V. Dinavahi, "A general framework for FPGA-based real-time emulation of electrical machines for HIL applications," *IEEE Trans. Ind. Electron.*, vol. 62, no. 4, pp. 2041–2053, Apr. 2015.

[23] 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. 3587–3597, Jul. 2014.

[24] Y. Wang and V. Dinavahi, "Low-latency distance protective relay on FPGA," *IEEE Trans. Smart Grid*, vol. 5, no. 2, pp. 896–905, Mar. 2014.

[25] W. Wang, Z. Shen, and V. Dinavahi, "Physics-based device-level power electronic circuit hardware emulation on FPGA," *IEEE Trans., Ind. Informat.*, vol. 10, no. 4, pp. 2166–2179, Nov. 2014.

[26] M. Dagbagi, A. Hemdani, L. Idkhajine, M. W. Naouar, E. Monmasson, and I. Slama-Belkhodja, "ADC-based embedded real-time simulator of a power converter implemented in a low-cost FPGA: Application to a fault-tolerant control of a grid-connected voltage-source rectifier," *IEEE Trans. Ind. Electron.*, vol. 63, no. 2, pp. 1179–1190, Feb. 2016.
[27] B. Jandaghi and V. Dinavahi, "Hardware-in-the-loop emulation of linear induction motor drive for maglev application," *IEEE Trans. Plasma Sci.*, vol. 44, no. 4, pp. 679–686, Apr. 2016.
[28] F. E. Fleming and C. S. Edrington, "Real-time emulation of switched reluctance machines via magnetic equivalent circuits," *IEEE Trans. Ind. Electron.*, vol. 63, no. 6, pp. 3366–3376, Jun. 2016.
[29] Y. Chen and V. Dinavahi, "Hardware emulation building blocks for real-time simulation of large-scale power grids," *IEEE Trans. Ind. Informat.*, vol. 10, no. 1, pp. 373–381, Feb. 2014.
[30] J. Liu and V. Dinavahi, "Detailed magnetic equivalent circuit based real-time nonlinear power transformer model on FPGA for electromagnetic transient studies," *IEEE Trans. Ind. Electron.*, vol. 63, no. 2, pp. 1191–1202, Feb. 2016.
[31] K. Ou *et al.*, "MMC-HVDC simulation and testing based on real-time digital simulator and physical control system," *IEEE J. Emerg. Sel. Topics Power Electron.*, vol. 2, no. 4, pp. 1109–1116, Dec. 2014.
[32] A. Hasanzadeh, C. S. Edrington, N. Stroupe, and T. Bevis, "Real-Time emulation of a high-speed microturbine permanent-magnet synchronous generator using multiplatform hardware-in-the-loop realization," *IEEE Trans. Ind. Electron.*, vol. 61, no. 6, pp. 3109–3118, Jun. 2014.
[33] N. R. Tavana and V. Dinavahi, "Real-time nonlinear magnetic equivalent circuit model of induction machine on FPGA for hardware-in-the-loop simulation," *IEEE Trans. Energy Convers.*, vol. 31, no. 2, pp. 520–530, Jun. 2016.
[34] H. F. Blanchette, T. Ould-Bachir, and J. P. David, "A state-space modeling approach for the FPGA-based real-time simulation of high switching frequency power converters," *IEEE Trans. Ind. Electron.*, vol. 59, no. 12, pp. 4555–4567, Dec. 2012.
[35] 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.
[36] G. G. Parma and V. Dinavahi, "Real-time digital hardware simulation of power electronics and drives," *IEEE Trans. Power Del.*, vol. 22, no. 2, pp. 1235–1246, Apr. 2007.
[37] T. Ould-Bachir, H. F. Blanchette, and K. Al-Haddad, "A network tearing technique for FPGA-based real-time simulation of power converters," *IEEE Trans. Ind. Electron.*, vol. 62, no. 6, pp. 3409–3418, Jun. 2015.
[38] *Vivado Design Suite User Guide, High-Level Synthesis, UG902 (v2015. 1)*, Xilinx, Inc., San Jose, CA, USA, 2015
[39] F. N. Najm, *Circuit Simulation.* Hoboken, NJ, USA: Wiley, 2010.
[40] L. Chua, "Efficient computer algorithms for piecewise-linear analysis of resistive nonlinear networks," *IEEE Trans. Circuit Theory*, vol. CT-18, no. 1, pp. 73–85, Jan. 1971.
[41] G. P. Fang, "A new time-stepping method for circuit simulation," in *Proc. Des. Autom. Conf.*, 2013, pp. 1–10.
[42] G. Söderlind and L. Wang, "Adaptive time-stepping and computational stability," *J. Comput. Appl. Math.*, vol. 185, no. 2, pp. 225–243, Jan. 2006.
[43] C. Deml and P. Turkes, "Fast simulation technique for power electronic circuits with widely different time constants," *IEEE Trans. Ind. Appl.*, vol. 35, no. 3, pp. 657–662, May/Jun. 1999.
[44] P. O. Lauritzen and C. L. Ma, "A simple diode model with reverse recovery," *IEEE Trans. Power Electron.*, vol. 6, no. 2, pp. 188–191, Apr. 1991.
[45] *IEEE Guide for the Application of Metal-Oxide Surge Arresters for Alternating-Current Systems*, IEEE Standard C62.22-2009, Jul. 3, 2009, pp. 1–142.
[46] R. B. Standler, "Equations for some transient overvoltage test waveforms," *IEEE Trans. Electromagn. Compat.*, vol. 30, no. 1, pp. 69–71, Feb. 1988.
[47] G. H. Golub and C. F. Van Loan, *Matrix Computations*, 3rd ed. Baltimore, MD, USA: Johns Hopkins Univ. Press, 1996.
[48] *DAC348x EVM User's Guide (Rev. A)*, Texas Instruments, Inc., Dallas, TX, USA, May 2016
[49] Xilinx Virtex UtralScale+ product table, (2016). [Online]. Available: https://www.xilinx.com/products/silicon-devices/fpga/virtex-ultrascale-plus.html#productTable

Zhuoxuan Shen (S'14) received the B.Eng. degree in electrical engineering from Jiangsu University, Zhenjiang, China, in 2013. He is currently working toward the Ph.D. degree in electrical and computer engineering at the University of Alberta, Edmonton, AB, Canada.

His research interests include real-time simulation of power systems, power electronics, and field-programmable gate arrays.

Venkata Dinavahi (SM'08) received the Ph.D. degree from the University of Toronto, Toronto, ON, Canada, in 2000.

He is a Professor of electrical and computer engineering at the University of Alberta, Edmonton, AB, Canada. His research interests include real-time simulation of power systems and power electronic systems, large-scale system simulation, and parallel and distributed computing.