

Hybrid Technique for Real-Time Simulation of High-Frequency-Switched Electrical Systems

Maher A. Hasan, Eric Walters, Michael Boyd, and Jason Wells
PC Krause & Associates

Jon Zumberge and Chad Miller
US Air Force

ABSTRACT

Experimental Hardware-in-the-loop (xHIL) testing utilizing signal and/or power emulation imposes a hard real-time requirement on models of emulated subsystems, directly limiting their fidelity to what can be achieved in real-time on the available computational resources. Most real-time simulators are CPU-based, for which the overhead of an instruction-set architecture imposes a lower limit on the simulation step size, resulting in limited model bandwidth. For power-electronic systems with high-frequency switching, this limit often necessitates using average-value models, significantly reducing fidelity, in order to meet the real-time requirement. An alternative approach emerging recently is to use FPGAs as the computational platform, which, although offering orders-of-magnitudes faster execution due to their parallel architecture, they are more difficult to program and their limited fabric space bounds the size of models that can be simulated. This paper presents a new method for simulating detailed models of power-electronic systems in real-time, targeting simulation platforms that combine both FPGAs and CPUs, but can also be adapted to run on multi-CPU systems. The method creates a multi-rate model by decomposing simulation process into *discrete integration* and *output refinement* stages. In the first stage, model equations are solved on the CPU at the fundamental switching rate to obtain the solution at key event points, whereupon the solution is refined in the second stage at a higher rate on the FPGA to the desired bandwidth. The method is applied to the real-time simulation of a detailed model of an inverter-driven motor switching at 10 kHz, and it is shown that transients up to the switching frequency are accurately captured.

CITATION: Hasan, M., Walters, E., Boyd, M., Wells, J. et al., "Hybrid Technique for Real-Time Simulation of High-Frequency-Switched Electrical Systems," *SAE Int. J. Aerosp.* 9(1):2016, doi:10.4271/2016-01-2028.

INTRODUCTION

The ever increasing complexity exhibited by modern aerospace platforms has made modeling, simulation, and analysis (MS&A) an integral part of the design process, necessary for studying system integration issues and identifying potentially adverse subsystem interactions. However, the inherent inadequacy of models to fully represent real components has caused system analysts and designers to increasingly utilize xHIL as a means of improving the fidelity of testing without incurring the cost of prototyping the full system in hardware [1]. In xHIL testing, representative hardware is used for system components of interest (e.g. a generator), with the rest of the system being emulated in software and interfaced with hardware via signal and/or power emulation. With xHIL, a more accurate representation of the system under test is achieved, subject only to the limitations of the excitation signals and sensor measurements. xHIL testing imposes a hard real-time requirement on the models of the emulated components. This requirement gives rise to a trade-off with model fidelity, since higher model bandwidth requires using smaller time-steps and therefore more computational time, which translates to an upper limit on the bandwidth achievable in real-time. Particularly challenging for real-time simulation are electromechanical and power

electronic system models with high-frequency switching, since the dynamics of interest often necessitate capturing bandwidth beyond the switching frequency, thus requiring either severely limiting the switching frequency or using reduced-order or average-value models, rendering xHIL significantly less effective. It remains a significant computational challenge to simulate accurately in real-time segment-level models of power-electronic subsystems with switching frequency greater than a few kHz, and a methodology by which this can be achieved for switching frequencies up to 20 kHz and beyond is greatly desired.

By far the most prevalent approaches to real-time simulation of dynamic system models utilize the CPU as the computational engine [2]. Typically, a low-overhead real-time operating system (RTOS) is loaded on the CPU, and models are converted to native C or C++ code that is compiled and deployed to the CPU like any other application. The key advantage of a CPU-based framework is the immense flexibility and richness of an instruction-set architecture (ISA) for representing dynamic models and simulation engines, such that offline models developed in commercial off-the-shelf (COTS) simulation programs can easily be adapted to real-time-compatible

form, with many such programs nearly automating this process [3]. This same flexibility, however, is behind the key disadvantage of CPU-based approaches: the overhead of an ISA and RTOS imposes a lower limit on the time-steps that can be taken to simulate the model (typically above a few μs [2]), and thus a corresponding upper limit on the bandwidth of the model that can be effectively captured.

To overcome this limitation, an alternative real-time simulation architecture has emerged in recent years that utilizes field-programmable gate arrays (FPGAs) as the main computational engine [4, 5, 6, 7]. Although the clock rate of FPGAs does not exceed a few percent of commonly available CPUs, their inherently parallel nature allows for executing many more computations per clock cycle, with a substantial overall decrease in overhead compared to CPUs. This enables taking sub- μs time-steps, potentially allowing significantly higher model bandwidth to be captured. There are two key disadvantages, however, associated with FPGAs: the first is the inherent inflexibility of hardware-based parallel architecture at expressing computational models and simulation engines, due to the lack of useful looping and branching constructs. This requires a significant amount of work to convert models developed in COTS simulation languages to FPGA-compatible form. This shortcoming is being partially alleviated with dedicated FPGA-based simulation engines that offer the same model development interface as CPU-based programs, automatically handling the required conversion to FPGA-compatible form of the resulting models [7]. However, a second key disadvantage remains, that of a limited amount of programmable interconnect fabric in an FPGA, which translates to a direct limit on the size of the model that can be deployed.

In this paper, a hybrid technique for simulating dynamic models of high-frequency-switched power-electronic systems is proposed that can leverage both the CPU and FPGA as computational engines. The fundamental idea is that by introducing reasonable relaxation assumptions on slowly-varying signals, one can decompose the process of dynamic simulation into two distinct phases: a computationally intensive *discrete integration* phase in which the solution at certain key points is produced, and a relatively cheap *output refinement* phase wherein the solution at the desired output points is obtained by interpolation. The discrete integration stage involves the solution of model equations and is therefore ideally suited to the CPU, whereas output refinement employs standard signal processing techniques which are particularly well-suited for an FPGA. The decoupling of the solution of model equations from output calculation enables one to carry out discrete integration at a potentially much lower rate than required for output, thereby avoiding the lower limit on the step size imposed by the CPU overhead. Further, the independence of output refinement from model equations alleviates the need for a large fabric space in the FPGA.

The rest of this paper is organized as follows: in the next section, the details of the proposed approach are presented and illustrated on a generic pulse-width-modulated (PWM) inverter/load subsystem. Subsequently, the implementation of the proposed approach on a real-time platform is described, showing the details of combined CPU/FPGA usage. The following section presents the results of

applying this approach to the simulation of an electromechanical actuator model, with a 10 kHz switching frequency and discusses accuracy and performance implications of the technique. The paper concludes with a summary of the approach and description of possible extensions.

PROBLEM FORMULATION AND SOLUTION METHODOLOGY

To introduce the proposed method, a simplified power-electronic system is considered, consisting of a 3-phase inverter driving a Y-connected RL-load, as shown in Figure 1. As high-frequency switching poses particular difficulties for real-time simulation, a PWM control strategy is considered for the converter. In this example, the desired emulation boundary is assumed to lie at the dc link, where the objective is to emulate a power-electronic load on a dc bus, to study, for example, the impact of high-frequency switching on the generation system. In this setup, v_{dc} is the input to the real-time simulation, and i_{dc} is the output that is emulated in hardware. The objective is to capture the relevant frequency content of i_{dc} accurately, up to the bandwidth capability of the emulator.

To keep the focus on the development of the real-time simulation method, it is assumed that the inverter switches are ideal, and that in each phase leg they are switched complementarily. With $\mathbf{S}_{abc} \in \{0,1\}^3$ a binary vector denoting the state of the top switch in each phase leg (1 - on, 0 - off), the equations of the combined inverter/load system can be expressed as

$$\mathbf{v}_{abcs} = \mathbf{S}_{abc} \mathbf{v}_{dc} \quad (1)$$

$$\mathbf{v}_{abcs} = \frac{1}{3} \begin{bmatrix} 2 & -1 & -1 \\ -1 & 2 & -1 \\ -1 & -1 & 2 \end{bmatrix} \mathbf{v}_{abcs} \quad (2)$$

$$\mathbf{v}_{abcs} = R \mathbf{i}_{abcs} + L \frac{d\mathbf{i}_{abcs}}{dt} \quad (3)$$

$$\mathbf{i}_{dc} = \mathbf{S}_{abc}^T \mathbf{i}_{abcs} \quad (4)$$

For given inputs v_{dc} and $\mathbf{S}_{abc}$, the output dc current is obtained by first calculating $\mathbf{v}_{abcs}$ from (1) and (2), then integrating (3) to obtain the load currents $\mathbf{i}_{abcs}$, and finally evaluating (4) for i_{dc} . In a PWM process, $\mathbf{S}_{abc}$ is generated by comparing the duty cycle commands $\mathbf{D}_{abc}$, which are determined by the controller according to the desired objective, with a carrier signal of high frequency f_{sw} . For example, Figure 2a illustrates the process with a 5-kHz triangular carrier signal. The corresponding solutions for $\mathbf{i}_{abcs}$ and i_{dc} are shown in Figures 2b and 2c, respectively. The system parameters for this study are $R = 2 \Omega$, $L = 1 \text{ mH}$, and $v_{dc} = 100 \text{ V}$. A space-vector modulation strategy with 100 Hz voltage command signals was used to generate the duty cycles $\mathbf{D}_{abc}$.

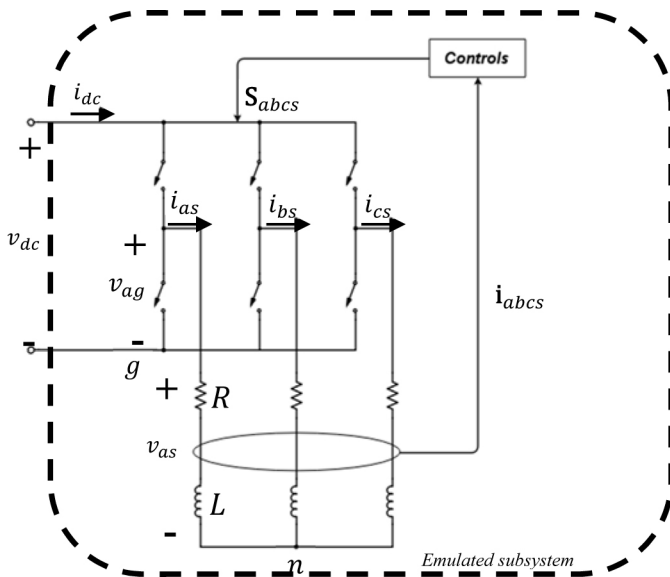


Figure 1. Example power-electronic load to be emulated.

Figure 2 depicts two fundamental switching periods $T_{sw} = \frac{1}{f_{sw}}$ of system operation. Upon examining the solution form, it is seen that each fundamental switching period T_{sw} can be decomposed into subintervals $[t_k, t_{k+1}]$, with boundaries at the intersections of $\mathbf{D}_{abc}$ with the carrier waveform. With this subdivision, each interval has a specific switching signal vector $\mathbf{S}_{abc}^{(k)}$, and as a result of the change in switch state, the corresponding load currents are *piecewise continuous*.

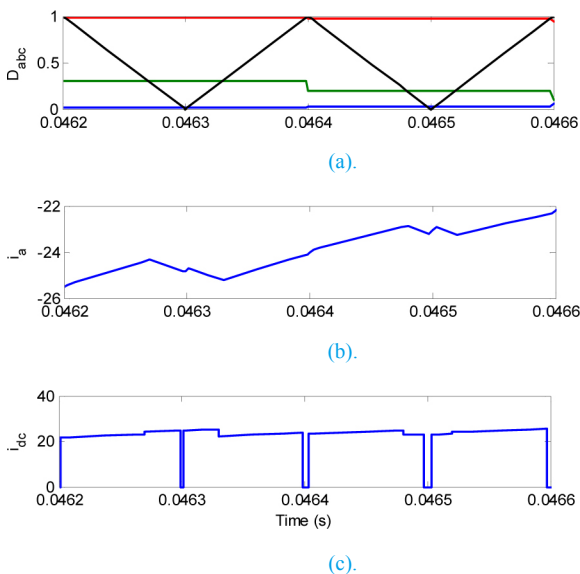


Figure 2. Example solution with a PWM control process.

The piecewise nature of the solution results in the spectrum of the solution waveforms having significant high-frequency content, well beyond the switching frequency f_{sw} . This is especially true for i_{dc} , as it is merely an algebraic combination of load currents according to the switch state in each subinterval $\mathbf{S}_{abc}^{(k)}$, which is discontinuous from one subinterval to the next. This significant high-frequency content poses peculiar difficulties in the context of real-time simulation, which

require fixed-step solvers to ensure deterministic computation time [2]. Specifically, it becomes necessary to use excessively small time-steps (much smaller than T_{sw}), inversely proportional to the highest frequency content of interest, in order to accurately capture the solution. This is illustrated in Figures 3 and 4, which show the solution obtained with fixed-step Forward Euler integration, with step-sizes of $20 \mu s$ and $2 \mu s$, respectively. As can be seen, a step-size 100 times smaller than the fundamental switching period is required to obtain good accuracy in the solution. The Forward Euler integration was implemented as a MATLAB script, and the reference solution was generated by simulating the model in Simulink with a variable step-size stiff solver (ode23tb) with tight tolerances to ensure convergence.

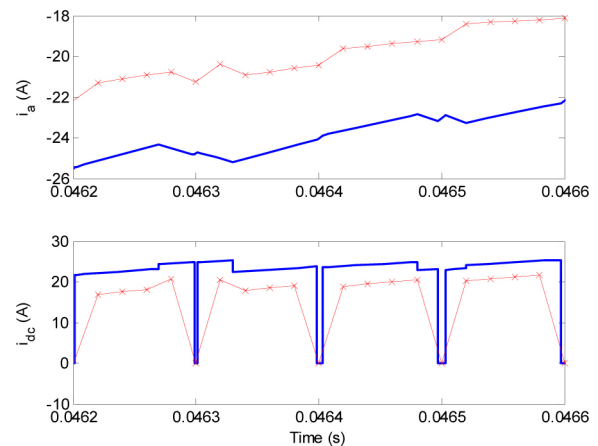


Figure 3. Results of Forward Euler integration with step size $h = 20 \mu s$ (red) vs. reference (blue).

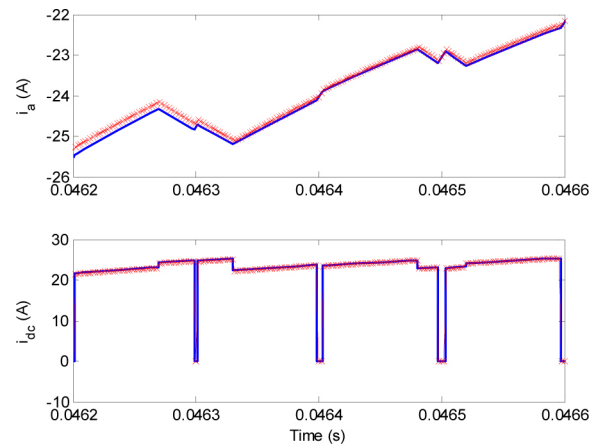


Figure 4. Results of Forward Euler integration with step size $h = 2 \mu s$ (red) vs. reference (blue).

The need for overly small integration step-sizes is the key obstacle for accurate real-time simulation of high-frequency-switched power-electronic systems. Step-sizes of a few μs or smaller cannot be simulated on the CPU due to the overhead associated with it and with the RTOS. As such, an FPGA may often be the only viable alternative currently available for performing these simulations; however, as discussed previously, they suffer from inherently less flexibility in programming and operation, as well as limitations of fabric space.