ELSEVIER

Electrical Power and Energy Systems 26 (2004) 703–713

ELECTRICAL POWER & ENERGY SYSTEMS

www.elsevier.com/locate/ijepes

Real-time digital simulation and experimental verification of a D-STATCOM interfaced with a digital controller

V.R. Dinavahi[a,*], M.R. Iravani[b], R. Bonert[b]

[a]*Department of Electrical and Computer Engineering, University of Alberta, Edmonton, Alta., Canada T6G 2V4*
[b]*Department of Electrical and Computer Engineering, University of Toronto, Toronto, Ont., Canada M5S 3G4*

Received 14 August 2002; revised 6 April 2004; accepted 28 May 2004

Abstract

This paper presents the implementation and experimental details of a real-time digital simulator designed for representing a Distribution Static Compensator (D-STATCOM) interfaced with a digital controller. Real-time simulation using conventional fixed time-step integration algorithms can lead to erroneous results due to the lack of synchronism between the simulator time-step and the discrete switching signals coming from the digital controller. The real-time simulation algorithm implemented in this paper overcomes the drawbacks of a fixed time-step algorithm. The real-time algorithm relies on precise registration of timing of the incoming discrete firing pulses and a subsequent *correction* procedure to calculate the system state. This approach has been implemented on a hybrid digital processing platform comprising of a Digital Signal Processor (DSP) and a Field Programmable Gate Array (FPGA). The platform represents both the power electronic system and the digital control system. Experimental results, for a 5 kV A D-STATCOM system, in verification of the real-time simulator results are also provided.

Keywords: Real-time simulation; Power electronics; Digital control; DSP; FPGA

1. Introduction

Rapid development in power electronic (PE) switches and availability of microprocessors and digital ICs for the realization of digital controls indicate proliferation of digitally controlled PE-based apparatus in the generation, transmission, distribution and utilization of electrical energy. Due to the ever increasing computational capability of microprocessors and digital ICs, digital control systems of PE apparatus assume more and more diagnostic, user-interface, control and protective functions. A practical challenge is to comprehensively evaluate and verify digital control system functions prior to the field installation.

One approach is based on off-line digital simulation of the control system, PE-based apparatus and the host power system. There are elaborate software tools for off-line simulation of PE converters and power systems. These tools conceptually permit off-line simulation of digital control systems as well. However, representation of a practical digital controller for off-line simulation is not a trivial task and furthermore prone to inaccuracy which defeats the purpose of verification.

A more attractive approach is to interface the actual digital controller with a real-time simulator. Real-time digital simulation of power systems is becoming increasingly popular for testing of new FACTS and HVDC controllers and protection systems [1,2]. The simulator provides real-time (on-line) simulation of the PEs and the host power system. Thus the real-time performance of the controller can be examined/verified. The main bottleneck in this approach is the numerical solution of the system equations in real-time. This bottleneck will persist for a long time despite rapid developments in CPU capabilities of computers. This indicates that the largest possible discretization

* Corresponding author. Tel.: +1-780-492-1003; fax: +1-780-492-1811.

E-mail addresses: dinavahi@ece.ualberta.ca (V.R. Dinavahi), iravani@ecf.utoronto.ca (M.R. Iravani), bonert@ecf.utoronto.ca (R. Bonert).

0142-0615/$ - see front matter © 2004 Elsevier Ltd. All rights reserved.
doi:10.1016/j.ijepes.2004.05.003

time-step has to be adopted for digital real-time simulation of the system. Thus synchronization between the simulator digitized process and the interfaced signals of the digital controller, e.g. switch firing signals cannot be achieved. Problems associated with the lack of synchronization and a solution approach for them, in case the simulator uses a fixed step-size, are described in Ref. [3].

This paper addresses the practical feasibility of the approach presented in Ref. [3]. It gives a detailed description of the real-time implementation of the proposed approach on a digital processing platform comprising of a Digital Signal Processor (DSP) and a Field Programmable Gate Array (FPGA). The system under study is a 5 kV A Distribution Static Compensator (D-STATCOM) based on a Pulse Width Modulated Voltage Source Converter (PWM-VSC) system interfaced with a digital controller. An experimental set-up of this system is used to validate the results of the real-time digital simulator.

The paper is organized as follows. Section 2 briefly presents the simulation method. Section 3 gives a detailed description of the real-time simulator architecture and Section 4 describes the experimental set-up of the D-STATCOM system. Results are presented in Section 5 followed by conclusions in Section 6.

2. Real-time digital simulation algorithm

Digital simulation of electromagnetic transients is essentially an initial value numerical integration process. Given the system state at time t_n and before, the problem is to find the state at time t_n+1 $(=t_n+\Delta t)$ where Δt is the step-size. Real-time digital simulation of PE systems when interfaced with an actual digital controller presents a problem due to the asynchronous switching events such as device firing pulses coming in between simulator calculation steps. The system simulator cannot respond until it completes its calculation cycle. The resulting delay in switching causes errors in the simulation output. Fig. 1 illustrates a simulation time-grid in which a discrete switching event comes in at time t_e. States x_n $(n=0,1,2,\ldots,)$ are simulator calculated states and y_n $(n=0,1,2,\ldots,)$ are the true system states that the actual physical system would have. Using the fixed step-size method (Fig. 1(A)) the simulator acknowledges the event at time t_2 when it has already calculated the incorrect state x_2. The specified change can only be carried out when the simulator calculates state x_3 at time t_3.

In the *F*ixed step-size with *I*nterpolation and *C*lock *S*ynchronization (FICS) algorithm (Fig. 1(B)), at the beginning of every time-step the simulator looks for switching event and its timing information in the previous time-step. In the presence or absence of this knowledge, one of the following operational paths are undertaken:

(1) Normal operation (when no switching event has been detected).
(2) Post-event operation (when a switching event has been detected).

2.1. Normal operation

The simulated system is divided into two subsystems: one containing the PE module and the other containing the network part.

2.1.1. Power electronic module

The circuit state of the PE module is dependent on the type of switches it is made of, such as naturally commutated thyristors or forced-commutated GTOs or IGBTs. Accordingly, the circuit variables $\mathcal{V}$ of the PE module,

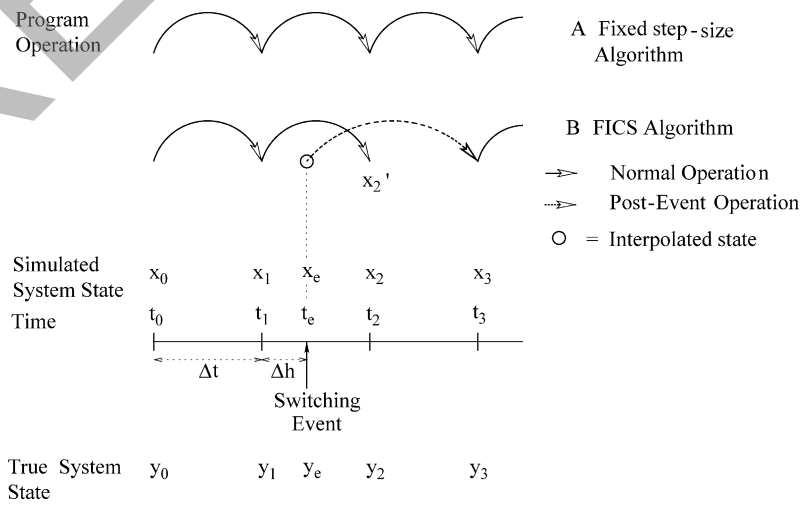

Fig. 1. Switching event in digital simulation: (A) fixed step-size approach, (B) FICS (Fixed step-size with Interpolation and Clock Synchronization) algorithm.

for example, voltages at the point of common coupling, may be functions of the firing signal $\mathcal{G}$ and/or the network state x

$$\mathcal{V} = f(\mathcal{G}, x) \tag{1}$$

The simulator then proceeds to obtain the numerical solution of the network state equations for the given time-step.

2.1.2. Network solution

For network solution, a fixed step-size program, e.g. the EMTP, uses the trapezoidal numerical integration method to convert continuous-time differential equations into discrete-time difference equations. This allows for recursive solution of the state equations. All electrical elements are represented as equivalent admittances and current injections. Nodal analysis is then used to formulate a set of linear algebraic equations. The transient solution proceeds by solving the linear equations for the unknown node voltages at each time-step. A simple first-order formulation of the method is given below.

The state equation for a linear combination of lumped elements such as R, L or C can be written in general as

$$\dot{x} = ax + bu \tag{2}$$

subject to the initial conditions $x(t_0)=x_0$ and $u(t_0)=u_0$, where x is the circuit state (e.g. inductor currents and capacitor voltages) and u is the input (e.g. AC system voltages). The parameters a and b are dependent on the circuit constants R, L and C. Using trapezoidal integration the value of x at time $(t+\Delta t)$ can be determined in terms of $x(t+\Delta t)$ and $x(t)$. The subscript n denotes quantities at time t and $n+1$ quantities at time $t+\Delta t$

$$x_{n+1} = x_n + \frac{\Delta t}{2}[ax_n + bu_n + ax_{n+1} + bu_{n+1}] \tag{3}$$

Eq. (3) can be re-arranged as

$$x_{n+1} = \alpha x_n + \beta(u_n + u_{n+1}) \tag{4}$$

where $\alpha = (1 + a\Delta t/2)/(1 - a\Delta t/2) = f_\alpha(\Delta t)$ and $\beta = (b\Delta t/2)/(1 - a\Delta t/2) = f_\beta(\Delta t)$. Recursive solution of Eq. (4) starting with $n=0$ yields the desired approximate solution of Eq. (2). Eq. (4) can be re-written as

$$x_{n+1} = \beta u_{n+1} + I_n \tag{5}$$

where $I_n = \alpha x_n + \beta u_n$. If the state x is a current and input u is a voltage then Eq. (5) represents an equivalent circuit consisting of a history current source I_n in parallel with an admittance β.

2.2. Post-event operation

At time t_2 (Fig. 1) the simulator possesses the following information:

- States x_2 and x_1
- Knowledge that the switching event occurred at time t_e relative to t_1.

Next, the simulator performs the following operations:

(1) Linearly interpolates the network variables (all states and inputs) at time t_e based on x_1, x_2 and Δh.

$$x_i = x_n + \frac{\Delta h}{\Delta t}(x_{n+1} - x_n),\ u_i = u_n + \frac{\Delta h}{\Delta t}(u_{n+1} - u_n) \tag{6}$$

(2) Computes the coefficients $\alpha_i = f_\alpha(2\Delta t - \Delta h)$ and $\beta_i = f_\beta(2\Delta t - \Delta h)$ based on the time-step $(2\Delta t\text{-}\Delta h)$ instead of Δt.
(3) Updates the PE model (1) based on the new gating information. This state of the PE model remains unchanged till the next switching event is detected.
(4) Calculates the history current sources I_n associated with the discrete time representation of the network components. During this calculation the results from steps 1 and 3 are used to properly initiate the history terms at time t_e. This operation incorporates the new changes made in the PE model and the network part acquired through linear interpolation.
(5) Calculates the state of the system at time t_3 using Eq. (5) and information from step 2.

Steps 3–5 are similar to the normal operation. The extra computation effort goes into steps 1 and 2, which indeed

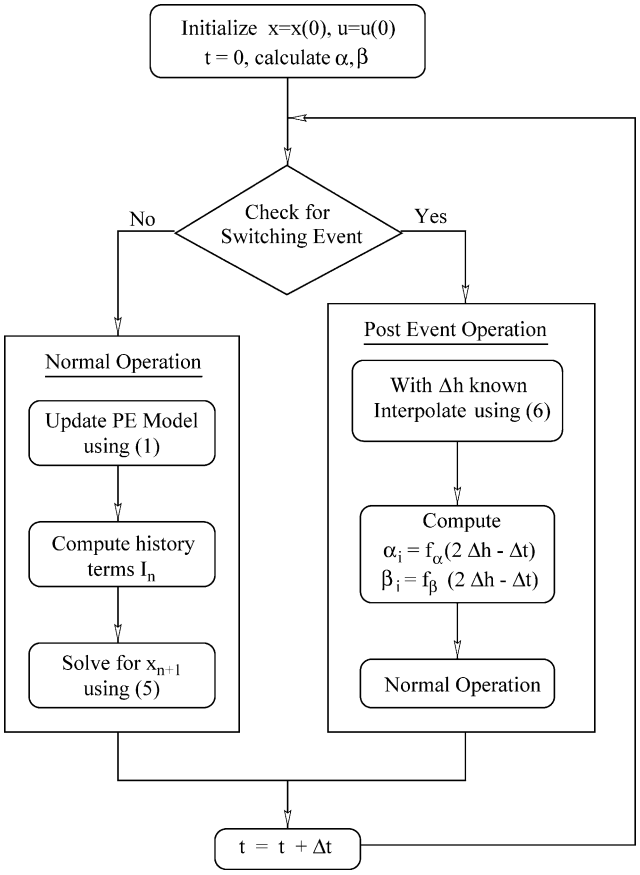

Fig. 2. Pseudo-flowchart of the FICS real-time simulation algorithm.

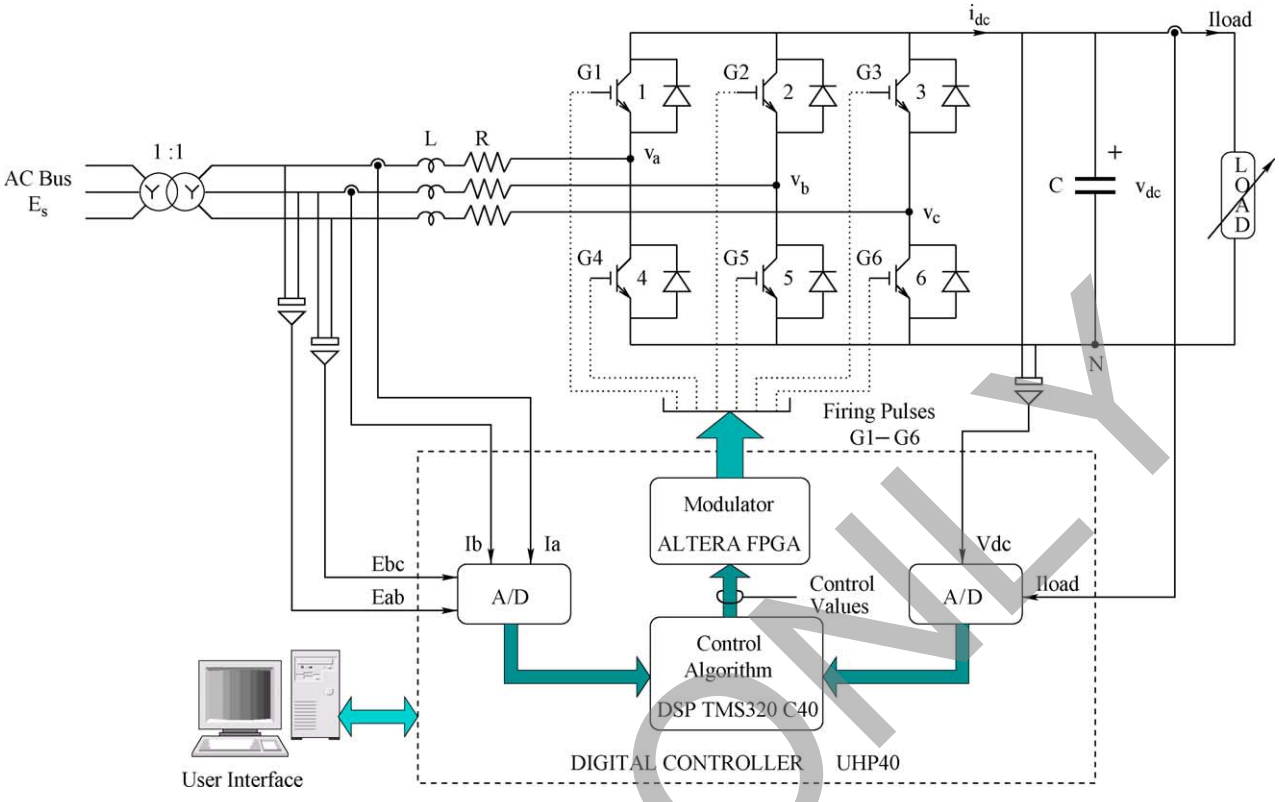

Fig. 3. Experimental set-up of the D-STATCOM system.

form the *correction*, after a switching event is detected. From the perspective of an external observer the apparent time-step of the simulator is still fixed at Δt, there is only an internal adjustment of the time-step whenever a switching event is detected. For the practical implementation, however, there should be a provision to precisely capture a switching event and relay its timing information to the simulator at the beginning of every time-step. The flowchart (Fig. 2) summarizes the real-time simulation algorithm. The following section gives the implementation details of the real-time simulator.

3. Real-time digital simulator architecture

The real-time digital simulation of the VSC system and its digital controller shown in Fig. 3 is carried out with the aid of the Universal High Performance (UHP40) [5] digital processing platform. The VSC system operates as a D-STATCOM. The architecture of the real-time simulator is based on a 32-bit floating point DSP TMS320C40 and a FLEX 8000 FPGA (Fig. 4). The FPGA shares the global data and address space of the DSP. The UHP40 board is interfaced to the PC with a data communication link Hotlink

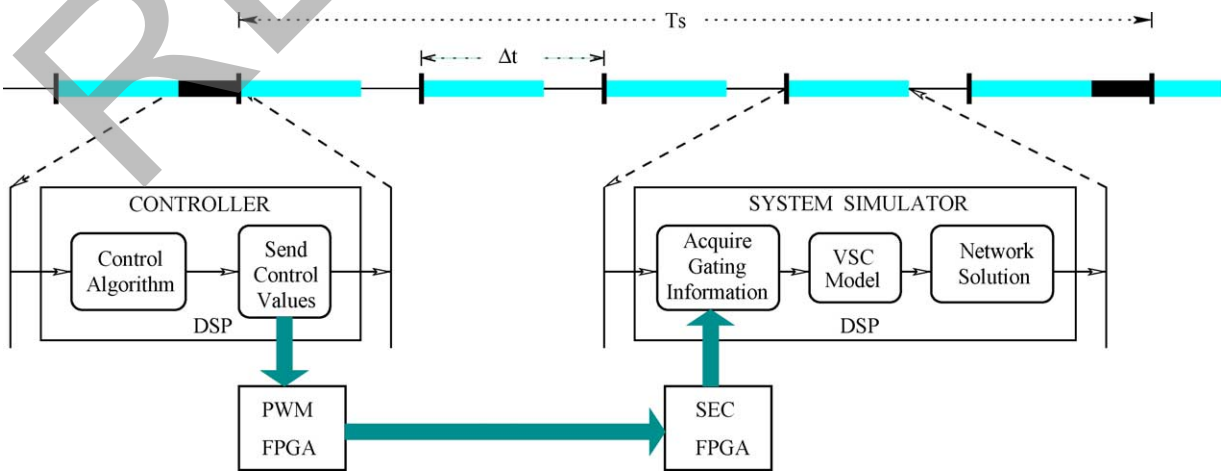

Fig. 4. Real-time program structure.

[5] capable of 20 MB/s data transfer. A Monitor program [5] running on the PC provides two-way interaction between the user and the UHP40. The program permits simultaneous (i) monitoring of the internal signals of the DSP and (ii) change in the system parameters while the simulation is in progress.

The simulation test-bed (Fig. 4) consists of five major modules: system simulator, controller, pulse width modulator, switching event capture (SEC) unit and the global bus interface. The DSP performs the functions of the simulator and the controller, while the FPGA does the functions of modulator, SEC and the bus interface. The DSP source code is written in C and the FPGA is programmed using the schematic design method of the MAXPLUS software. The scenario of a real-time digital simulator interacting with a digital controller is made possible by a synchronized bidirectional data transfer between the DSP and the FPGA managed by the global bus interface. Fig. 5 shows the various modules on the simulation time grid. This diagram only shows the sequence of operations carried out, their exact timing is given later in the paper. The functions of each of the five modules are described below.

3.1. System simulator

The simulator code is executed in the DSP as an interrupt service routine at every time-step Δt programmed by an on-board timer. The code performs three major functions. Firstly, it acquires the gating pulses and the timing information from the FPGA at the beginning at each time-step. Secondly, based on the gating signals, the states of the six switches in the VSC are decided. The IGBT switches are modeled as ideal bidirectional switches with gate turn-on and turn-off controls. The VSC model is based on discrete switching functions $S_k(t)$

$$u_{kN}(t) = S_k(t)v_{dc}(t), \quad k = a, b, c \tag{7}$$

where $S_k(t)=1$ if $G_k=$high and $S_k(t)=0$ if $G_k=$low, G_k are the gating signals. Then depending on whether a switching event is detected in the previous time-step one of the two paths shown in the flow-chart of Fig. 2 is adopted for the network solution. The R–L network is represented by three differential equations

$$L\frac{di_k}{dt} + Ri_k(t) = v_k(t) - e_k(t), \quad k = a, b, c \tag{8}$$

The dc side equations are

$$-C\frac{dv_{dc}}{dt} = i_{dc}(t) \tag{9}$$

$$i_{dc}(t) = \sum_k S_k(t)i_k(t), \quad k = a, b, c \tag{10}$$

Once every T_s ($=n\Delta t$; n integer) the simulator transfers voltage and current signals to the controller module. This action emulates the D/A–A/D data transfer channel between a real-time simulator and a digital controller. In the present design, however, the data transfer being fully digital, the quantization effects resulting from implementing such a channel are ignored.

3.1.1. Initialization

The simulator is initialized by a user-defined function in the source code. This function specifies the initial values of the system and control variables, parameters, and PWM information such as frequency, resolution, etc. It also specifies the FPGA address space; memory addresses to where the control values should be sent to and addresses from where the gating information can be obtained. In addition, look-up tables are created for one period of sine and cosine functions, which serve as sources in the simulator. Since power-frequency (60 Hz) voltage sources are periodical, they need to be calculated for only one period, stored in a look-up table and applied at every subsequent cycle (every 16.66 ms). Once initialized, the real-time simulation can run indefinitely on the DSP. The user can alter the system parameters or initiate transients on-line and view the system evolve over time.

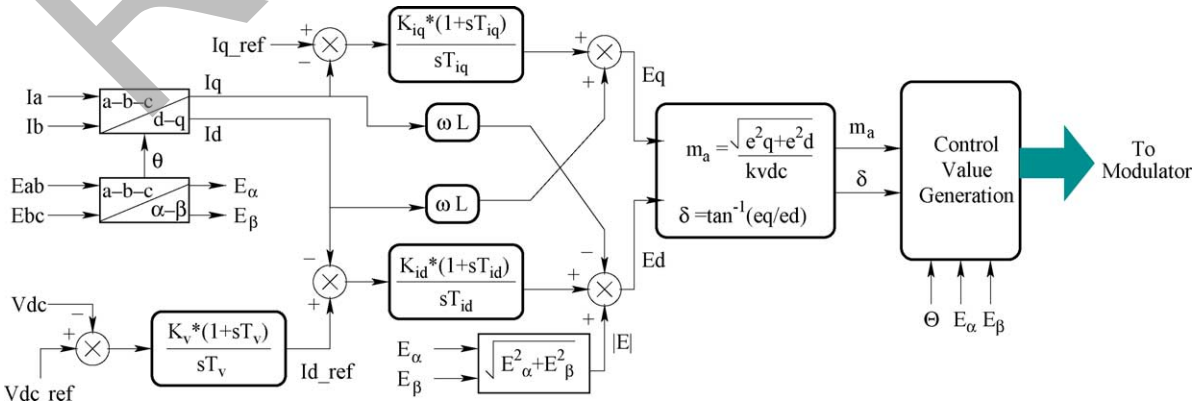

Fig. 5. Control block for the D-STATCOM system.

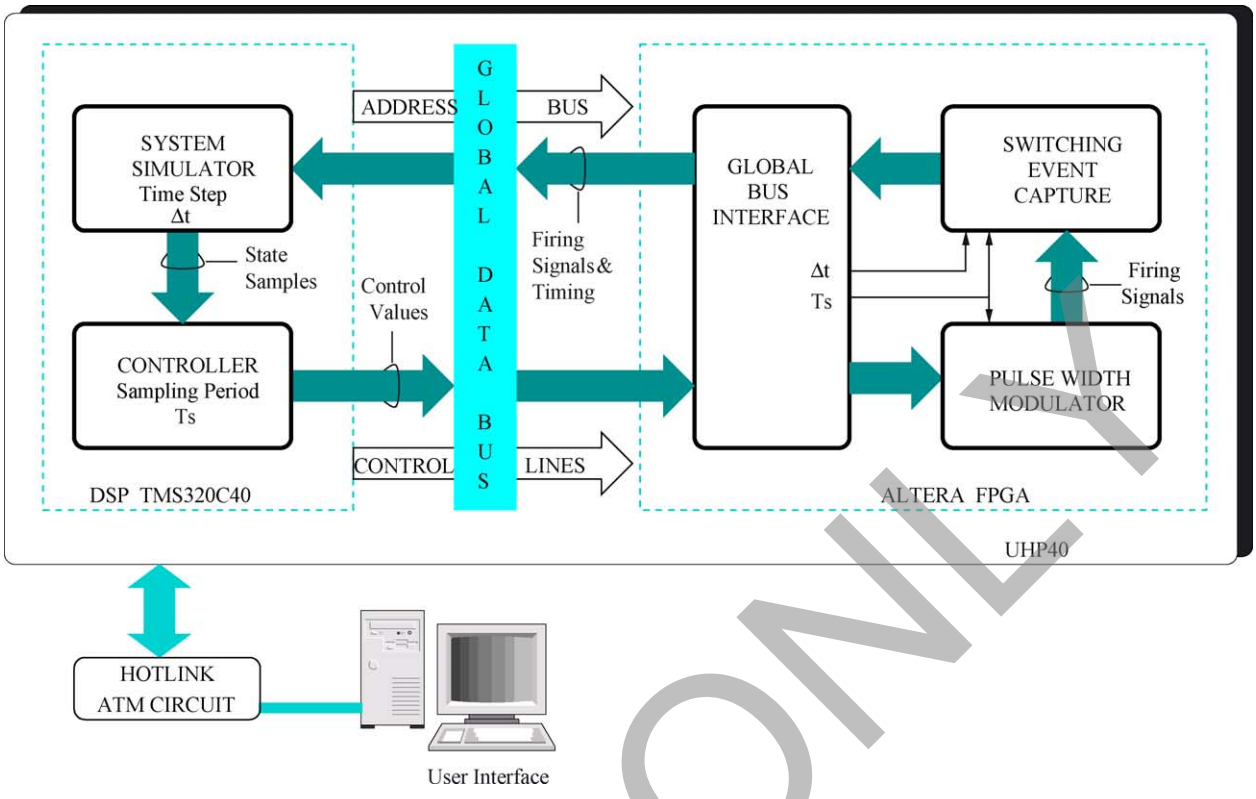

Fig. 6. Real-time digital simulator hardware.

3.2. Digital controller

The control algorithm is executed once every T_s which is the controller sampling period. The code executes the following three functions: acquisition of signals from the system simulator, the control algorithm, and transfer of sinusoidally modulated control values to the FPGA. The control principle [4] of the VSC system is shown in Fig. 6. A coordinate transformation from *abc* frame to a synchronous d*q* frame is performed on the signals sampled from the simulator. Then a decoupled control of the currents i_d and i_q is performed using PI compensators for each of the current loops. The dc-link voltage is regulated through an external feedback loop that provides the real current reference i_{dref}.

3.3. Modulator

Sinusoidal pulse width modulation is implemented in the Modulator part of the FPGA. A triangular carrier waveform, synchronized with the controller sampling interval T_s, is compared with the sinusoidal control value acquired from the controller module in the DSP. Gating pulses are generated at the intersection of the carrier and control waves. These pulses are stored in an output buffer in the Global Bus Interface and transferred to the system simulator at the beginning of each time-step Δt. The output pins of the FPGA are accessible for the user to examine the firing signals with an oscilloscope.

3.4. Switching event capture (SEC)

This digital hardware design in the FPGA is used to time-stamp the firing pulses coming from the modulator. It determines the location of the firing pulses with respect to the start of the controller sampling interval T_s and stores it as a counter value whose resolution is 50 ns/count. This value is transferred to the system simulator in the DSP at the beginning of the simulation time-step Δt. Based on the state of the firing pulses and the counter value, the simulator can then determine the occurrence and the location Δh of an event in the previous time-step. This information is used for interpolation and the step-size change.

3.5. Global bus interface

The primary function of this module is to arbitrate bidirectional data transfer between the DSP and the FPGA. In addition to the communication signals, it provides the Δt and T_s signals to the modulator and the SEC circuit to synchronize their respective operations.

4. Experimental set-up

An experimental set-up corresponding to the system of Fig. 3 was used to verify the real-time simulation results.

4.1. Power circuit

The power circuit parameters are as follows:

Three-phase supply $V_s = 110V_{ll}$, 60 Hz
Three-phase transformer 115 V/115 V, 10 kV A, 60 Hz, $X_T = 3.06\%$, $R_T = 2.71\%$
$R = 0.5\ \Omega$, $L = 3.0$ mH, $C = 4900\ \mu$F

The three-phase VSC is composed of a six-pack 1200 V, 50 A IGBT module.

4.2. Digital controller

The control principle used for the test set-up is same as the one used for the real-time simulation. The data acquisition of system voltages and currents is carried out by dedicated A/D cards mounted on the DSP board. The control algorithm is executed on the C40 DSP. The PI constants used for the control are $K_{id} = 0.8$, $K_{iq} = 1.5$, $T_{id} = T_{iq} = 50$ ms, $K_v = -2.0$, and $T_v = 0.5$ s. Sinusoidal PWM with a carrier frequency of 1 *kHz* is implemented in the FPGA. Every sampling period T_s ($= 500\ \mu$s), the modulator takes in three-phase control values from the DSP and gives out switching signals for the six inverter switches. The FPGA inserts a dead time of 2 μs into the switching sequence. In case of maloperation, the user can send a trip signal to the FPGA via the DSP, to disable the gating pattern generator allowing shutdown of the unit.

5. Results and discussion

5.1. Real-time digital simulation and experimental results

For direct comparison of results the real-time simulation was carried under the same operating conditions as for the experimental set-up. Open and closed loop tests were conducted on the VSC system. A 100 μs time-step was chosen for the real-time simulation. This choice was governed by the performance of the C40 DSP. A smaller time-step could not be chosen under the present design conditions. Figs. 7–12 show two sets of results—one obtained from the real-time simulation and the other from the experimental set-up. Figs. 7–10 show the open loop voltage and current under steady state operation of the VSC with $m_a = 0.8$ and $\delta = 20°$. Fourier analysis of these waveforms showed a close agreement between the real-time simulation and experimental results. Figs. 11 and 12 show the step response of the i_q current controller under closed-loop operation. An off-line simulation with a $\Delta t = 10$ μs was also carried out and the results were found to be agreeable with those obtained with the experimental set-up and the real-time simulation. Besides the fact that the adopted simulation method only approximates the behaviour of a fixed step-size method with a small time-step, the difference in the results can be attributed to the following unmodeled effects in the real-time simulation:

- Quantization effects of the A/D transfer.
- *Filtering effects*. Low pass filters were used in the experimental set-up to reduce noise levels on sampled

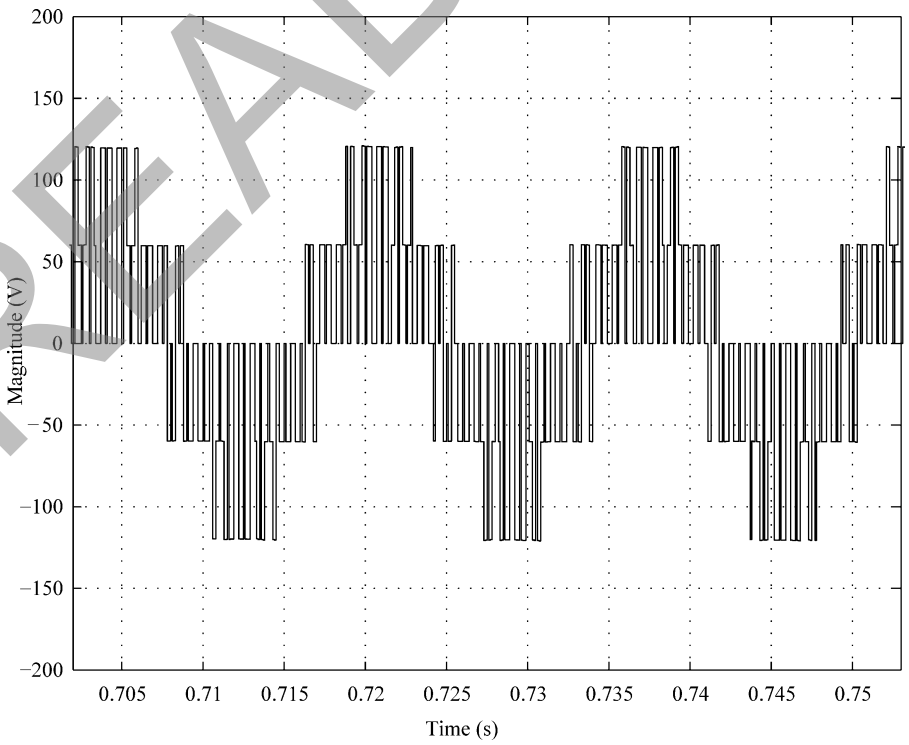

Fig. 7. Steady-state voltage v_{an} under open-loop control (RTS).

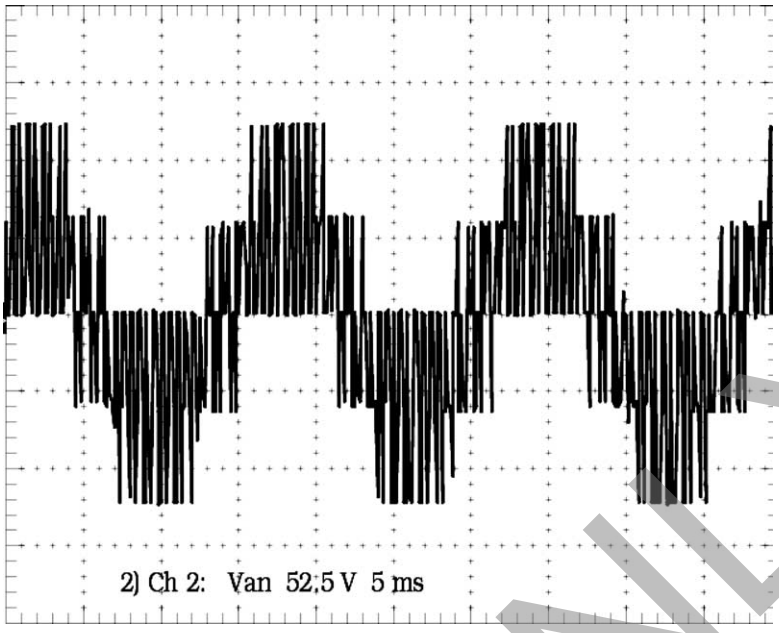

Fig. 8. Steady-state voltage v_{an} under open-loop control (experiment). Scale x-axis: time (5 ms/1.2 cm), y-axis: voltage (52.5 V/1.2 cm).

signals before they are fed to the controller. Their phase lag contribution is compensated in the control algorithm.

- *PWM dead-time*. In the real-time simulator, switches in the VSC are modelled as ideal and switches in the same inverter leg are turned-on and off simultaneously. The real IGBT switches have finite turn-on and turn-off times and there is a 2 μs delay between the turn-off of an upper switch and the turn-on of a lower switch in the same leg and vice-versa. This effect could not be modeled due to the in-feasibility of accounting a 2 μs shift in turn-on and turn-off pulses on a time-step of 100 μs. For a low switching frequency, this is acceptable but for higher switching frequencies when the PWM dead-time becomes comparable to the switching times, this effect cannot be ignored.
- Unbalance in the voltages and parameters of the experimental set-up.
- Stray inductances and capacitances of the experimental set-up.

Despite these considerations, the two sets of results match favourably.

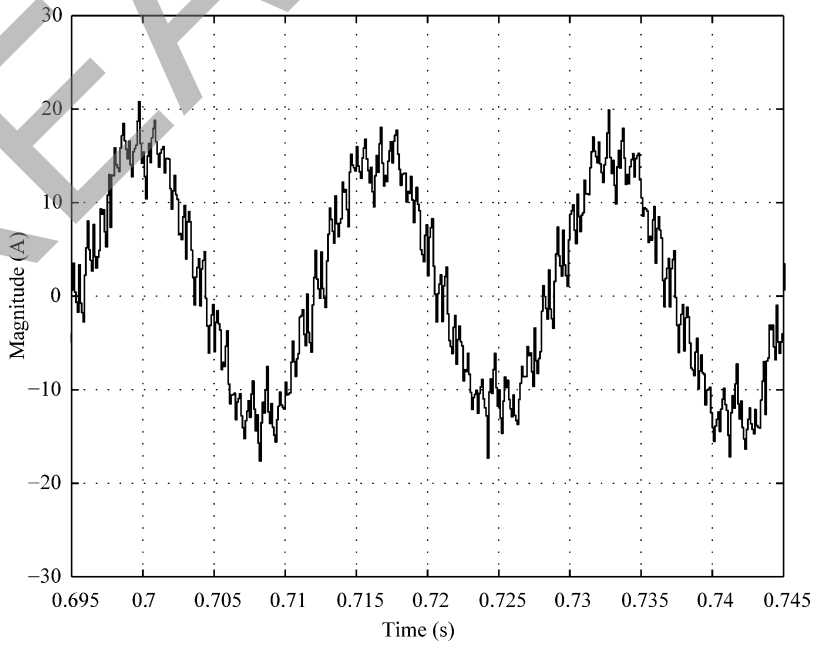

Fig. 9. Steady-state current i_a under open-loop control (RTS).

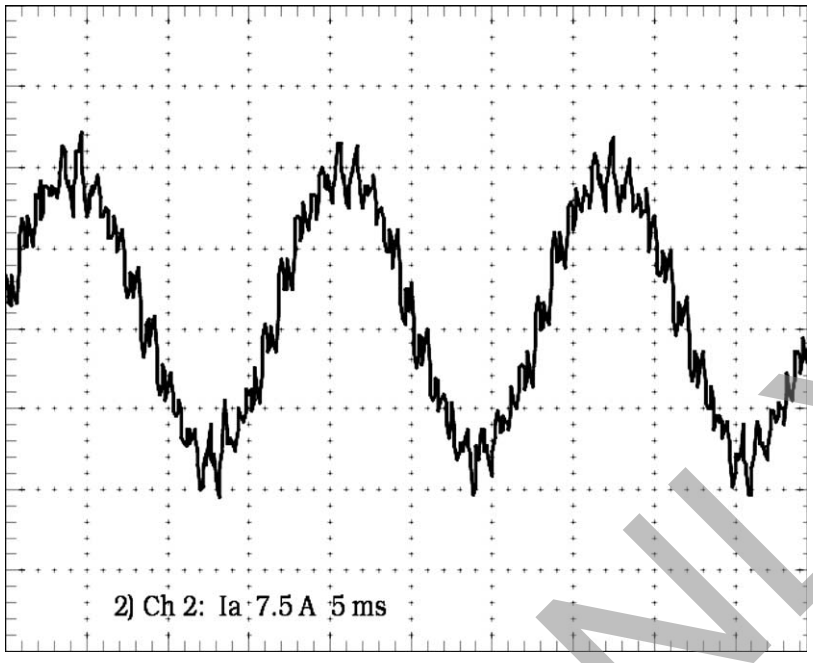

Fig. 10. Steady-state current i_a under open-loop control (experiment). Scale x-axis: time (5 ms/1.2 cm), y-axis: current (7.5 A/1.2 cm).

5.2. Processor timing

The time taken by the C40 DSP to execute each of the modules in the user interrupt function are measured. The real-time simulation uses a time-step of 100 μs, 84 μs of which is the total simulation time and 16 μs overhead is used for tasks such as initializing the DSP kernel program (4 μs) and managing data acquisition and display of up to three signals (12 μs) on the PC. The simulation time comprises of 36 μs used by the simulator module and 48 μs used by the controller module. The controller runs at a frequency of 2 kHz which amounts to a control period of

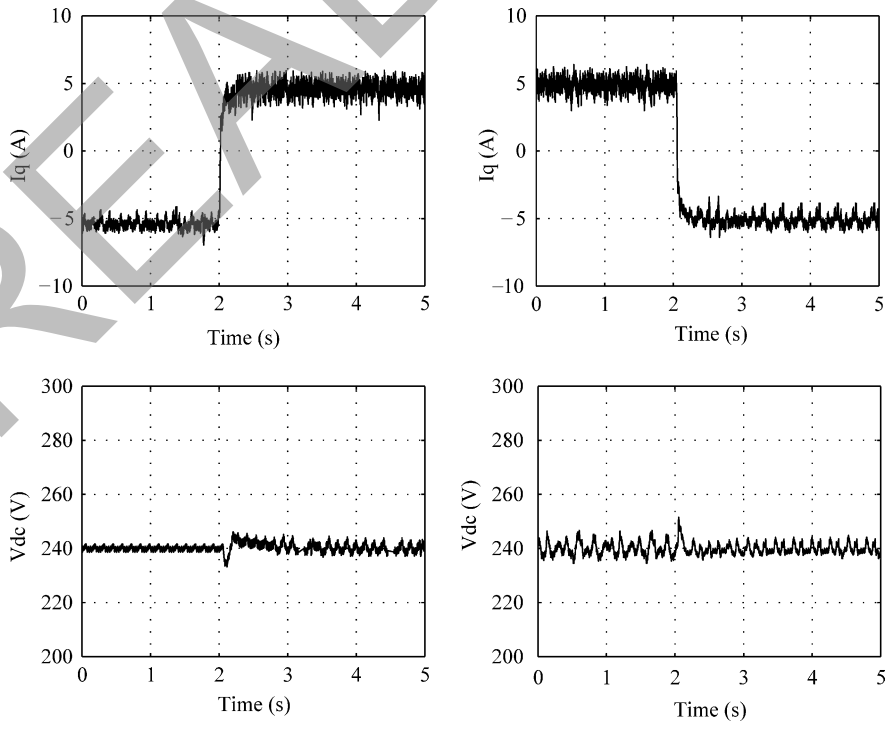

Fig. 11. i_q transient under closed-loop control (RTS).

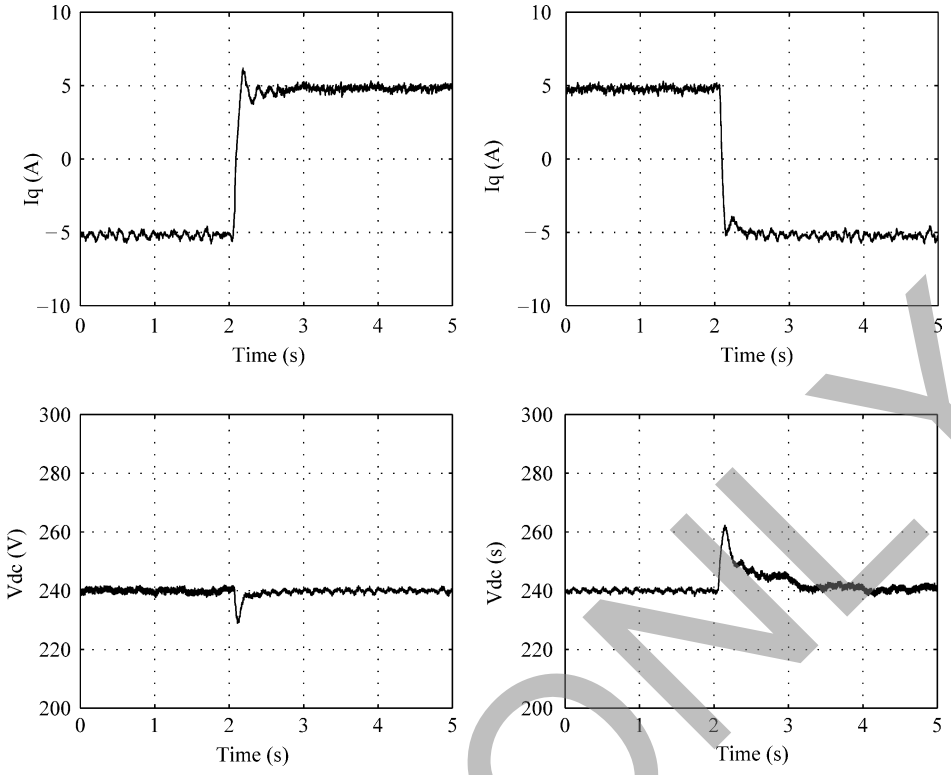

Fig. 12. i_q transient under closed-loop control (experiment).

500 μs. In the simulator module, the network solution takes two paths depending on whether a discrete event has been detected in the previous time-step. The linear interpolation and variable step-size path takes 26 μs to execute while the normal path (fixed step-size) takes 16 μs to execute.

The FPGA operates at a clock frequency of 20 MHz. The PWM and the SEC in the FPGA are carried out in parallel with the simulation in the DSP. The discrete event capture unit takes about 15 ns to detect a firing pulse and transfers its timing information to the simulation. From the point of view of the simulation in the DSP, the FPGA response time can be considered to be instantaneous.

These timings are only indicative of the performance of the C40 DSP under the present design. A more complex system or a higher switching frequency for the same system would require the simulation to be carried out at a smaller time-step necessitating the use of a faster DSPs. The code for the simulator and the controller has been developed in C, which makes them readily portable to a platform consisting of a faster DSP such as the C67.

6. Conclusions

Real-time digital simulators that use a fixed simulation step-size give inaccurate results when modeling switching power circuits, which depend on external firing pulses for a change of state. To alleviate this problem, a simulation method that relies on linear interpolation and variable step-size numerical integration is presented. This paper reports results confirming the practical feasibility of such a simulator using commercially available digital components. Experimental tests on a 5 kV A VSC system validate the results of the real-time simulator and also show that the simulation method is accurate and reliable. A major limitation of the real-time simulator is the modeling complexity that can be attempted without violating real-time requirements. An example of the modeling complexity involved is the model of a switching device. In the simulation, the switches are treated as ideal and their turn-on and turn-off characteristics are not simulated. But the IGBTs used in the physical set-up have turn-on and turn-off times of 0.5 and 1.5 μs, respectively. Simulation of switching characteristics of this device in real-time would therefore entail a step-size of less than 0.5 μs. This is not feasible using existing DSP technology such as the C40 DSP and will require much faster DSPs to carry out this level of real-time simulation. Another limitation is the size of the circuit that is simulated. The way to partly remove this restriction is to parallelize computations in a multi-DSP environment. However, such an approach could eventually hit an upper-limit on the number of DSPs employed when inter-DSP communication times outweigh the computational gains of parallelization.

Despite the emerging high-speed highly integrated DSP technology, there will always be a lack of processing power when bigger and more complex systems need to be simulated. A compromise is usually needed in terms of model simplification to emulate the physical system as closely as possible while meeting real-time constraints at the same time.

References

[1] Jakominich D, Krebs R, Retzmann D, Kumar A. Real time digital power system simulator design considerations and relay performance evaluation. IEEE Trans Power Deliv 1999;14(3): 773–81.
[2] Brandt D, Wachal R, Valiquette R, Wierckx R. Closed loop testing of a joint VAR controller using a digital real-time simulator. IEEE Trans Power Syst 1991;6(3):1140–6.
[3] Dinavahi V, Iravani MR, Bonert R. Real-time digital simulation of power electronic apparatus interfaced with digital controllers. IEEE Trans Power Deliv 2001;16(4):775–81.
[4] Schauder C, Mehta H. Vector analysis and control of advanced static VAR compensators. IEE Proc-C 1993;140(4):299–306.
[5] Krebs S. UHP40 user's manual. Technical report. University of Toronto; 1995.

Venkata Dinavahi received his B.Eng. (1993) from Nagpur University, India, and MTech. (1996) from the Indian Institute of Technology, Kanpur, India. He received his PhD (2000) in Electrical and Computer Engineering from the University of Toronto, Canada. Presently, he is an assistant professor at the University of Alberta, Edmonton, Canada. He is a member of IEEE, CI-GRE and he is a Professional Engineer in the Province of Alberta, Canada. His research interests include electromagnetic transient analysis and real-time simulation and control. He can be reached at dinavahi@ece.ualberta.ca.

Reza Iravani received his BSc degree in Electrical Engineering in 1976 from Tehran Polytechnique University and started his career as a consulting engineer. He received his MSc and PhD degrees also in Electrical Engineering from the University of Manitoba, Canada in 1981 and 1985, respectively. Presently he is a professor at the University of Toronto. His research interests include power electronics and power system dynamics and control.

Richard Bonert received the Dipl.-Ing. and the doctorate degrees in electrical engineering from the University of Karlsruhe, Germany, in 1969 and 1977, respectively. He joined Brown Boveri Germany in 1969 as a project engineer. He was also with the Elektrotechnisches Institut of the University of Karlsruhe as a Research Associate and Chief Engineer. He joined the ECE Department at the University of Toronto in 1980 where he is a professor. His main interest is in the field of electrical drives, automatic feedback control systems for electric drives and power electronic circuits.