

# Real-Time Hardware-In-the-Loop Testing During Design of Power Electronics Controls

Bin Lu, *Student Member, IEEE*, Antonello Monti, *Senior Member, IEEE*, Roger A. Dougal, *Senior Member, IEEE*

**Abstract**— In the process of designing digital controls for power electronics, hardware-in-the-loop testing is increasingly recognized as an effective approach. A high performance simulation environment is necessary to obtain high-fidelity results. Many experiments have been proposed in the literature already. However in most cases, the environments are single platform, single solver, or non real-time. In this paper, we present the implementation and testing procedure of the Real-Time Virtual Test Bed, a multi-platform, multi-solver, hard real-time simulation environment for power electronics controls. By using the Signal Extension Resistive Companion Solver, RTVTB enables natural coupling between hardware and software, and thus makes power exchange with closed-loop hardware possible. As an application example, a boost converter state space feedback control system is studied and analyzed.

**Index Terms**— Power electronics, hardware-in-the-loop, real-time, dSpace, Virtual Test Bed.

## I. INTRODUCTION

Traditional software-based simulation has the disadvantage of being unable to exactly replicate real operational conditions. One way to bridge the gap between simulation and real conditions is hardware-in-the-loop (HIL) simulation. HIL testing in the development cycle of a control system reduces the risks of discovering an error in the very last stage of the on-the-field testing and assembling.

Many instances of HIL simulation have been described in the literature [1]–[7], however, in most cases the systems are soft real-time or even non real-time [2], hardware dependent or based on proprietary solutions [3][4][5], or supported only by a single platform or solver [1][7]. We present here a multi-platform, multi-solver, hard real-time simulation environment for rapid prototyping of power electronics controls, especially with HIL architecture. In the rest of this paper, we will refer to this system as RTVTB, the Real-Time extension of the Virtual Test Bed (VTB).

A brief background of VTB [8]–[10], the Windows based non real-time ancestor of RTVTB, is given in section II. Section III introduces RTVTB and its implementation. A proposed design approach for real-time control for power electronics systems using RTVTB is described in section IV. Finally, the applicability of using RTVTB to perform real-time control for power electronics systems is analyzed through a DC-DC boost converter feedback control system in section V. In section VI, the performance of RTVTB is evaluated based on the experimental results.

## II. VTB ENVIRONMENT

The VTB project is dedicated to developing a new environment for simulation and virtual prototyping of power electronic systems. Within the context of "virtual prototyping" we include not only simulation of system dynamics, but also solid modeling of the system and visualization of the system dynamics. As a high-level virtual prototyping tool, the VTB program addresses many challenges in a field such as power electronics, which encompasses a wide range of disciplines, including analog electronics, digital electronics, power systems, controls, electro-mechanics, and mechanical systems. The complexity of power electronics systems inevitably bridges several areas of technical expertise and the engineers in each of those technical areas traditionally work with their own set of design and simulation tools.

The VTB environment addresses these challenges by supporting:

### 1) *Multi-formalism*

Different languages can be used to build models of the different components of a system. This allows an individual to build models using the preferred language within his or her discipline.

### 2) *Highly interactive environment*

Users can change the system topology or parameters while a simulation executes. This allows the user to rapidly investigate interactions between components or to explore the influence of design parameters on system performance.

### 3) *High-level visualization*

Visualization models of the system can be easily created and linked to live simulation data by the Visualization Extension Engine (VXE). Visualization helps the user to rapidly comprehend the system performance. Visual outputs include data-driven animation of the motion of solid objects, imposition on top of the solid objects of novel representations of abstract simulation data, or simply oscilloscope-like waveforms.

## III. RTVTB ENVIRONMENT

### A. *RTVTB Overview*

In the case that real hardware is involved, the software must deal with additional challenges. These challenges motivate us towards RTVTB, the Real-Time extension of the Virtual Test Bed. RTVTB supports:

### 1) Multi-platform

Hardware-oriented simulation has to deal with more problems than software-based simulation, especially the inexorable forward progression of time. This requires the simulation environment to operate in hard real-time mode. The hardware abstraction layer of Windows makes it difficult and inefficient to achieve hard real-time. UNIX-like systems such as Linux provide better support for real-time simulation. Therefore, Linux was adopted as the underlying operating system for RTVTB. A detailed discussion of this issue is available in [11].

RTVTB, at this moment, does not have all the features of the Windows version VTB. In particular, neither the Schematic Editor nor the VXE visualization tool has yet been exported to Linux. However, RTVTB reads the same file format that is created by the schematic editor, so the system simulation file can be defined on a Windows platform, and then the system file can be saved to a Linux platform, where the real-time simulation will be executed.

### 2) Multi-solver

By using RTVTB, a simulation system can be distributed into several host computers. For each host computer, users can choose the main VTB solver as well as other solvers. This means that any part can be solved with the most appropriate integration step and method without affecting the solution of the rest of the system. Currently, two VTB solvers have been compiled under Linux and are available for RTVTB:

#### a) Simulated Analog Computer (SAC) solver

The SAC solver is a block diagram solver, like Matlab/Simulink®. It solves a system's describing differential equations in the same way an analog computer would. This was the first VTB solver to be exported to Linux due to its ease of handling. Detailed information about this solver is available in [13]. Real-time HIL simulation examples using the SAC solver can be found in [7] and **Error! Reference source not found.**

#### b) Signal extension Resistive Companion (SRC) solver

The heart of the SRC solver is the resistive companion method (RCM), which expresses each component in a system in the form of equation (1).

$$I(t) = G(t-h) * V(t-h) - B(t-h) \quad (1)$$

where  $I$  is the current through the component,  $V$  is the voltage across the component,  $G$  is the conductance,  $B$  is the state source,  $t$  denotes time and  $h$  is the simulation step size. Let us take the capacitor as an example. The voltage and current relationship for the capacitor is as in equation (2).

$$I = C * \frac{dV}{dt} \quad (2)$$

Applying the trapezoidal rule, we have

$$V(t) = V(t-h) + \frac{h}{2} * \left( \frac{I(t)}{C} + \frac{I(t-h)}{C} \right) \quad (3)$$

Equation (3) is rearranged into form (1), where:

$$G(t-h) = \left( \frac{2 * C}{h} \right) \quad (4)$$

$$B(t-h) = \left( \frac{2 * C}{h} \right) * V(t-h) + I(t-h) \quad (5)$$

A circuit-oriented interpretation of the mathematics of the capacitor model according to the RCM theory is shown in Fig.1.

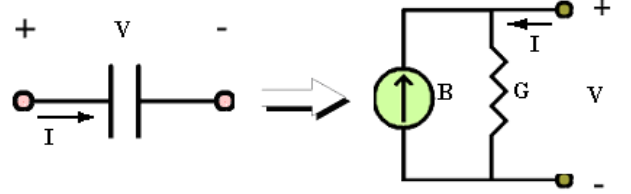


Fig. 1. Circuit-oriented interpretation of the capacitor model according to the RCM theory.

The SRC solver utilizes the RCM theory above. Additionally, it also supports signal coupling due to a multi-backplane that allows for mixed-signal simulation. At this point, the SRC solver is the main solver for both VTB and RTVTB. Detailed information about it is available in [14]. In particular, the SRC solver enables natural coupling between hardware and software, and thus makes the power exchange with closed-loop hardware possible. This is exactly the most significant aspect of HIL simulation, incorporating real hardware into a software-simulation system. The application example in section V is based on the SRC solver.

### B. RTVTB Components

Many systems for HIL simulation have already been developed, such as QNX, RTOS, and Opal-RT. However, all of them are based on proprietary solutions. RTVTB is unique among these commercial real-time systems, because it is completely implemented with public domain software and standard PC hardware. From the software point of view, RTVTB consists of three free software packages.

#### 1) Linux

Linux is selected as the operating system of RTVTB due to its low cost, high performance and support for hard real-time. It provides the basic functions, development tools and the user interface. In the development process of RTVTB, various Linux distributions including Mandrake, Redhat, Caldera and Suse were demonstrated to be suitable as the operating system.

#### 2) Real-Time Application Interface

Hard real-time requires the operating system to be both preemptive and deterministic. In RTVTB, an open-source package, Real-Time Application Interface (RTAI), is used to achieve this requirement. RTAI is a kernel modification and enhancement package of Linux that permits the handling of time-critical tasks. It was initially developed by the "Dipartimento di Ingegneria Aerospaziale Politecnico di Milano", Italy.

#### 3) Comedi

An HIL simulation system requires I/O interfaces to hardware. In RTVTB this is achieved by Comedi, freeware that develops open-source device drivers for many different data acquisition (DAQ) cards. It consists of two complementary

packages: "comedi", which implements the kernel space functionality, and "comedilib", which implements the user space access to the device driver functionality. Comedi works with a standard Linux kernel as well as RTAI.

Detailed RTVTB installation procedures using these software packages are given in [11].

### C. Real-time Implementation

There are three major components in the RTVTB real-time implementation.

#### 1) Real-time Task

RTAI preempts the standard Linux kernel and handles hardware interrupts. In RTVTB, a real-time task is generated by RTAI to manage the 8254 chip (clock generator) to generate a real time clock which is used as the basis for defining the simulation step. This real-time task is a loadable module in Linux; it stays in the kernel-space upon being loaded.

#### 2) Linux Process

An RTVTB solver is realized by a set of standard Linux processes. In this way, it is similar to other Linux programs, such as a text editor. In each step interval, the solver takes in the system input from the analog input port of the DAQ card, solves the system state, and sends the system output through the output port. The Linux process is a user-space application program and thus has no direct communication with the real-time task.

#### 3) Real-time FIFO

Since in RTVTB the real-time clock information has to be passed to the solver, a real-time FIFO (First In First Out) is applied as the "bridge" between the real-time task and the Linux process. Real-time FIFO is a uni-directional read/write buffer created by RTAI. After simulation starts, it continuously records the real-time clock generated by the real-time task. Simultaneously the Linux process polls the real-time FIFO, detects the simulation step and performs the simulation.

The RTVTB real-time implementation process is illustrated by Fig.2.

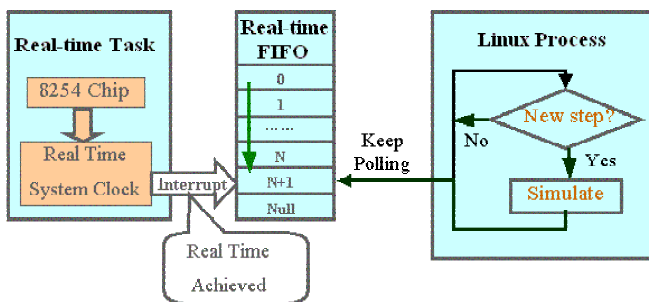


Fig. 2. Real-time implementation in RTVTB.

### D. RTVTB Architecture

Fig.3 shows the architecture of RTVTB. This architecture allows the user to perform a real-time HIL testing of a system that includes real hardware. This test phase can be considered the very last step before real on-line testing of power

electronics controls. Through this process the user can verify not only the algorithmic correctness of the system (in this case a controller) in the simulator, but also its capability to meet the real-time constraints.

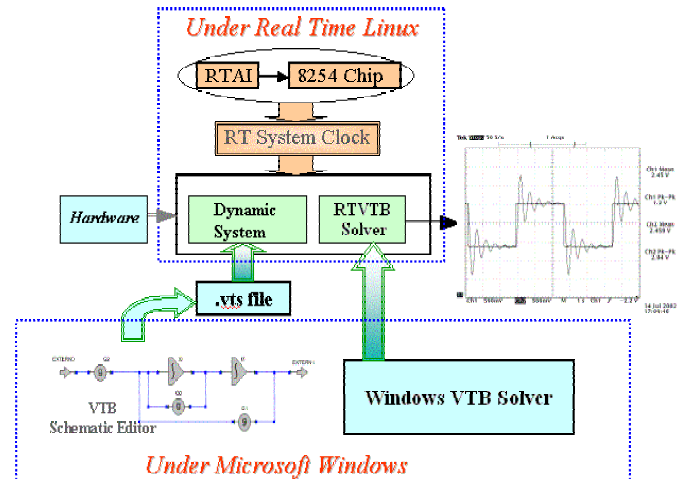


Fig. 3. The architecture of RTVTB.

## IV. APPROACH TO REAL-TIME CONTROL FOR POWER ELECTRONICS USING RTVTB

Power electronics control designers often use simplified models of the plant in the early stages of the system design. By doing this, they can focus on the control algorithm itself rather than the complexity of the plant. However, many factors may affect the viability of the derived controller, such as over-simplification of the model and data communication time delays. Therefore, in order to be confident of the design, the control system must be tested under more strict conditions. In the case of the switching power converter, different levels of models could be considered subsequently.

- Averaged model
- Switching model
- Real converter real-time HIL testing

Reference [8] gives a detailed example of using both average model and switching model of a boost converter. Here, we will mainly focus on real-time HIL testing with real hardware.

The control system design can be summarized in four steps, as shown in Fig.4.

#### 1) Step 1:

Design a high-fidelity system model for both controller and plant in Simulink and Windows VTB. Tune the control parameters if necessary. This will be the ideal result.

#### 2) Step 2:

Link the controller model in Simulink developed in step 1 into Windows VTB and perform the VTB-Simulink co-simulation. This can be used as a comparison to analyze the HIL results in step 3 and step 4.

#### 3) Step 3:

Use the Matlab real-time workshop to compile the Simulink

control system and export it to dSpace. Perform the non hard real-time HIL testing using VTB and dSpace.

4) *Step 4:*

Replace the controller in dSpace with a controller in RTVTB, and replace the VTB plant model with real power electronics hardware. Perform the hard real-time HIL testing.

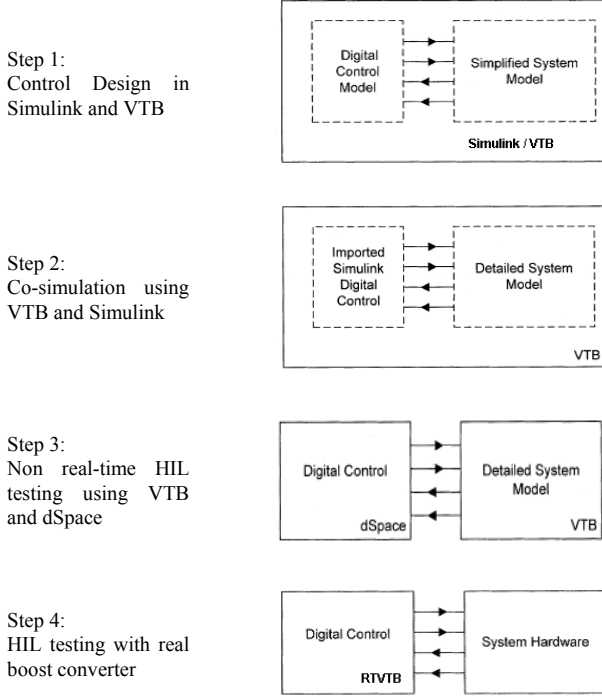


Fig. 4. Design approach of real-time control for power electronics.

## V. APPLICATION EXAMPLE: REAL-TIME STATE SPACE FEEDBACK CONTROL FOR A BOOST CONVERTER

Now let us illustrate the process during the implementation of a state space feedback control for a boost converter, showing all of the steps described in section IV.

First, the analytical control design uses pole placement on the linearized system. Poles are selected so that the closed-loop system has satisfactory dynamic behavior. The designed system is then built in Simulink and VTB to obtain the ideal results. After that, HIL testing is performed on this system using dSpace and VTB. Actually this is not hard real-time, since the simulation is still hosted in Windows. However, it does give a general approach for HIL testing where a hard real-time system is not available. Finally, the focus is fixed on the actual hardware implementation of hard real-time HIL testing using RTVTB.

The experiment described in this section is based on the latest RTVTB version. The system configurations are listed in Table I. The boost converter has the parameters given in Table II.

TABLE I  
RTVTB SYSTEM CONFIGURATIONS

Hardware configuration		Software configuration	
CPU	PIII 800MHz	Linux release	Mandrake 9.0
Hard drive	10 Gigabytes	Linux kernel	2.4.20
RAM	128M SDRAM	RTAI	24.1.10
Video RAM	8M SDRAM	Comedi	0.7.66
Network	10Mbps ether net	Comedilib	0.7.19
Free PCI Slots	1	gcc version	3.2
I/O DAQ Cards	Advantech PCI1710 Advantech PCI1720		

TABLE II  
PARAMETERS OF PROTOTYPE BOOST CONVERTER

Rated input voltage	12V
Rated output voltage	40V
Maximum load	100W
Input inductance	46μH
Output filter capacitance	1.360mF
Main switch	IRF540N
Switching frequency	50kHz

### A. Analytical Control Design

Let us start by considering the classical time-averaged model of the boost converter:

$$\text{Error! Objects cannot be created from editing field codes.} \quad (6)$$

$$\text{Error! Objects cannot be created from editing field codes.} \quad (7)$$

$$0 \leq u \leq 1 \quad (8)$$

Where  $X_1$  is the inductor current,  $X_2$  is the capacitor voltage and  $u$  is the switch duty cycle.

We adopt a state space linear feedback control strategy. A local linearization procedure is applied to the model [15]:

$$\text{Error! Objects cannot be created from editing field codes.} \quad (9)$$

Our objective is to obtain the control law as a function of the variation of state variables  $X_1$  and  $X_2$

$$\text{Error! Objects cannot be created from editing field codes.} \quad (10)$$

The new state matrix including the feedback will be:

$$\Phi_{CL} = \begin{bmatrix} \left( \frac{\bar{X}_2 k_1}{L} \right) & \left( \frac{\bar{X}_2 k_2 - (1-\bar{u})}{L} \right) \\ \left( \frac{(1-\bar{u}) - \bar{X}_1 k_1}{C} \right) & \left( \frac{1}{RC} - \frac{\bar{X}_1 k_2}{C} \right) \end{bmatrix} \quad (11)$$

The characteristic equation of the  $\Phi_{CL}$  is given by:

$$\text{Error! Objects cannot be created from editing field codes.} \quad (12)$$

Let the desired closed-loop poles be  $P_1, P_2$ . Using the properties of the second order equations, we obtain the following equations in matrix form giving feedback coefficients  $k_1, k_2$  as a function of desired poles  $P_1, P_2$ .

$$\text{Error! Objects cannot be created from editing field codes.} \quad (13)$$

This equation has the form

$$\text{Error! Objects cannot be created from editing field codes.} \quad (14)$$

and solving for  $k_1, k_2$  gives

$$\begin{bmatrix} k_1 \\ k_2 \end{bmatrix} = \frac{1}{q_{11}q_{22}-q_{12}q_{21}} \begin{bmatrix} q_{22} & -q_{12} \\ -q_{21} & q_{11} \end{bmatrix} \begin{bmatrix} m_1 \\ m_2 \end{bmatrix} \quad (15)$$

**B. Testing Procedures**

*1) Step 1*

As shown in Fig.5, the analytical closed-loop transfer function of the whole system is implemented in Simulink. Fig. 6 shows that when the reference output voltage steps from 12V to 20V, the actual output voltage follows the reference with a second order behavior, which is determined by the selection of closed-loop poles  $P_1$  and  $P_2$ . In our case, we fixed the two poles at -1000 and -5000. This is regarded as the ideal result.

The closed-loop system, including the controller and the plant model, is then built in VTB as shown in Fig.7. Simulation result of the output voltage is observed to be in reasonably good agreement with the ideal case, as shown in Fig.8.

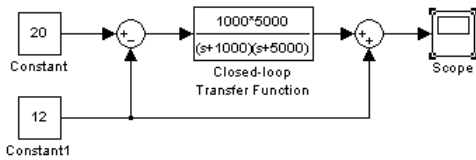


Fig. 5. Schematic based on the analytical closed-loop transfer function of the system in Simulink

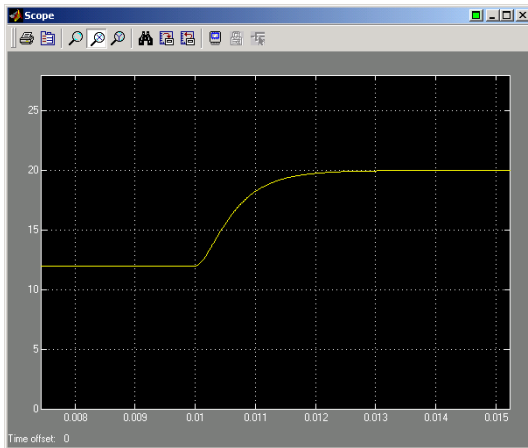


Fig. 6. Output voltage of the closed-loop system in Simulink, when reference voltage steps from 12V to 20V (ideal case).

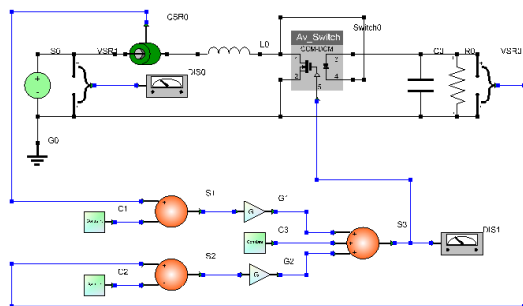


Fig. 7. Schematic of the closed-loop system in VTB.

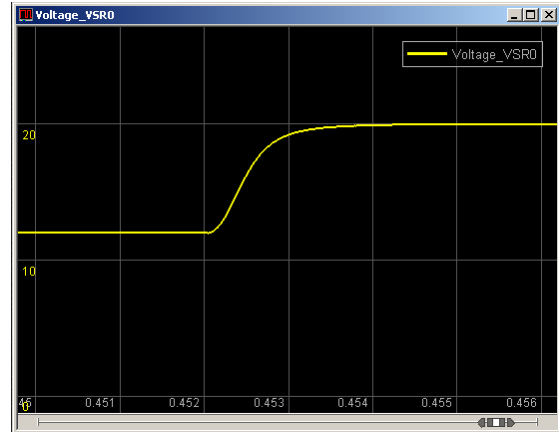


Fig. 8. Output voltage of the closed-loop system in VTB, when reference voltage steps from 12V to 20V (software-only simulation case).

*2) Step 2*

The controller, shown in Fig. 9, is implemented in Simulink. It is then simulated with VTB by using the interactive interface between VTB and Matlab/Simulink. The schematic of the co-simulation system is shown in Fig. 10.

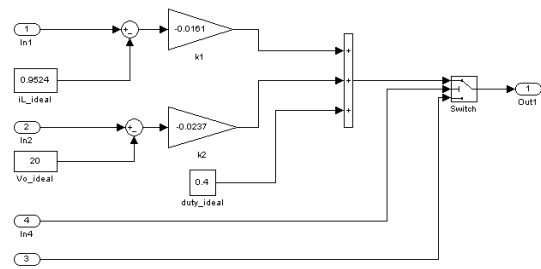


Fig. 9. State space feedback controller block in Simulink.

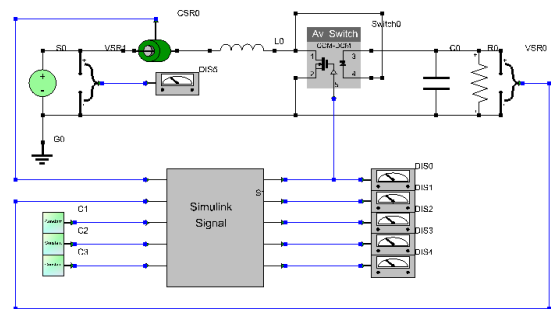


Fig. 10. VTB schematic for interactive co-simulation (with Simulink) of the plant and the controller.

*3) Step 3*

The dSpace system supports direct compilation of the Simulink control definition and downloading of the compiled code onto the dSpace platform without any change. Since step 2 and step 3 do not deal with hard real-time, for simplicity, the simulation results are omitted here. They are available in [2].

*4) Step 4*

As shown in Fig.11, the state space feedback controller is implemented in VTB and then exported into RTVTB. It is nearly identical to the one in Fig. 7. The only difference is that in Fig.11, three “RTVTB PCI-1710 I/O Card” interface objects are used to communicate with the boost converter hardware.

They are driven by Comedi software under RTVTB.

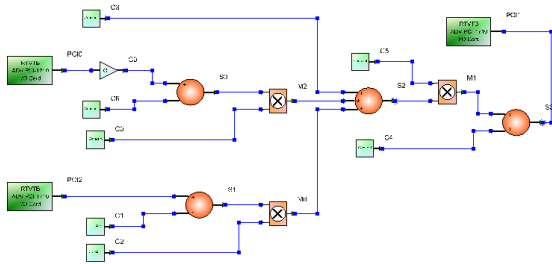


Fig. 11. State space feedback controller block in VTB (exported into RTVTB).

Finally, hard real-time HIL testing is performed with the controller in RTVTB and a real boost converter. Fig.12 shows the boost converter's response when the reference output voltage steps from 12V to 20V. It is in good agreement with the ideal results predicted in step 1.

Notice that the steady state output voltages in Fig.12 (11.6V and 19.2V) are less than the ideal case (12V and 20V). This is due to the fact that the simplified models in the ideal case do not take any loss into account. Also we observe that the experimental result has piecewise-like behavior. This is because of the real-time simulation step size. In this case, it is 300 $\mu$ s. If a faster host computer is used for RTVTB, the time resolution can be easily reduced, and thus the result will be improved.

The agreement between Fig.6, Fig.8 and Fig.12 confirms that the real-time state space feedback control of a boost converter is achieved by using RTVTB hard real-time HIL testing approach.

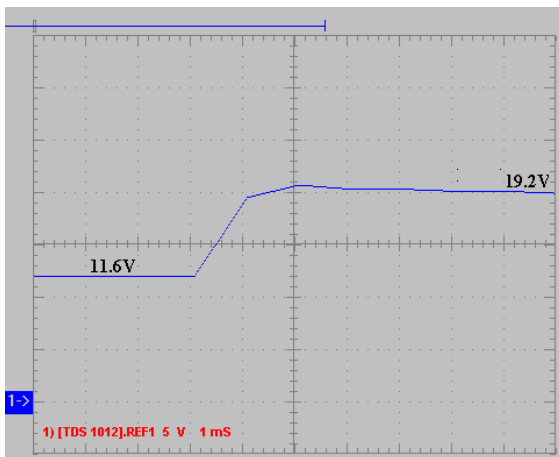


Fig. 12. Output voltage of the real boost converter, when reference voltage steps from 12V to 20V (hard real-time HIL simulation case).

## VI. CONCLUSION

This paper has described a complete procedure for design of real-time controls for power electronics. This approach is based on RTVTB, a multi-platform, multi-solver, hard real-time HIL simulation environment. It allows the designer to verify the control design using real hardware for maximum confidence.

Through an application example, we have shown the

capability of this design approach to achieve a precise state space feedback control for a boost converter.

In RTVTB, the real-time clock has already been proved to reach 300 $\mu$ s under a PIII 800MHz PC. This is fast enough for most time critical hardware and comparable with most commercial real-time systems. Clearly this time resolution could be easily improved by using a newer computer. The beauty of this work is that hard real-time is obtained with open-source software and PC standard hardware. Thus the same environment can be replicated easily with very low cost.

The SRC solver is exported into RTVTB and used as the solver for real-time HIL testing for power electronics control. This makes natural coupling between simulation environment and hardware plant applicable. Benefiting from this, power-in-the-loop testing could be realized in the future.

## ACKNOWLEDGMENT

This work was supported by the US Office of Naval Research under grant N00014-02-1-0623.

## REFERENCES

- [1] R. Dougal, A. Monti, B. Pettus, E. Santi, "High level virtual prototyping with hardware in the loop," *IEEE VIMS00*, Annapolis, MD, Apr. 2000.
- [2] S. Lentijo, A. Monti, E. Santi, C. Welch, and R. Dougal, "A new testing tool for power electronic digital control," accepted to *IEEE PESC03*, Acapulco, MÉXICO, Jun. 2003.
- [3] P. Terwiesch, T. Keller, E. Scheiben, "Rail vehicle control system integration testing using digital hardware-in-the-loop simulation," *IEEE Trans. on Control System Technology*, vol. 3, No. 7, May, 1999, pp: 352-362.
- [4] P. Baracos, G. Murere, C. Rabbath, W. Jin. "Enabling pc-based HIL simulation for automotive applications," *IEEE International Electric Machines and Drives Conference*, 2001, pp: 721 -729.
- [5] E. Acha, O. Anaya-Lara, J. Parle, M. Madrigal, "Real-time simulator for power quality disturbance applications," *Ninth International Conference on Harmonics and Quality of Power*, vol. 3, 2000, pp: 763-768.
- [6] A. Monti, R. Dougal, S. Ayasun, S. Vallieu, "On the stability of hardware-in-the-loop simulation," *Electrimacs 2002*, Montreal, Canada, 2002.
- [7] B. Lu, W. McKay, S. Lentijo, A. Monti, X. Wu, and R. Dougal, "The real time extension of the virtual test bed," *Huntsville Simulation Conference*, Huntsville, AL, Oct. 2002.
- [8] A. Monti, E. Santi, R. Dougal, M. Riva, "Rapid prototyping of digital controls for power electronics," *IEEE Trans. Power Electronics*, vol. 18, No. 3, May 2003.
- [9] R. Dougal, T. Lovett, A. Monti, E. Santi, "A multilanguage environment for interactive simulation and development of controls for power electronics," *IEEE PESC01*, Vancouver, Canada, 2001.
- [10] T. Lovett, A. Monti, R.A. Dougal, "The new architecture of the Virtual Test Bed," *IEEE-COMPEL02*, Mayaguez, Puerto Rico, Jun. 2002.
- [11] B. Lu, "The real-time extension of the virtual test bed: A multi-solver hard real-time hardware-in-the-loop simulation environment," M.S. thesis, University of South Carolina, May 2003.
- [12] W. McKay, A. Monti, R. Pettus and R. Dougal, "Simulated analog computation," *Huntsville Simulation Conference 2002*, Huntsville, AL, Oct 2002.
- [13] W. McKay, "Simulated analog computation: implementation of a block diagram solver," M.S. thesis, University of South Carolina, Dec 2002.
- [14] T. Lovett, "Design and implementation of a multi-discipline simulation environment," M.S. thesis, University of South Carolina, Dec 2002.
- [15] K. Astrom, B. Wittenmark, *Computer Controlled Systems: Theory and Design*, Prentice Hall, 1997, USA.