FPGA-BASED REAL-TIME SIMULATION OF POWER ELECTRONIC SYSTEMS

Roy Crosbie[1], John Zenor[1], Dale Word[1], Richard Bednar[1] and Narain Hingorani[2]

[1]California State University, Chico, Department of Electrical & Computer Engineering
400 West First Street, Chico, CA., USA
[2]Consultant, Los Altos Hills, CA, USA

rcrosbie@csuchico.edu

Abstract

Modern power systems depend on power electronic components that use high-speed switching controllers to provide improved stability, flexibility and conditioning of power supplies. The key components are switching circuits (converters) that convert alternating current to direct current (rectification) and vice versa (inversion). As switching speeds are increased in order to reduce low-order harmonics and improve controllability, the duration of the time steps used in simulations of these power systems must be reduced to avoid significant timing errors. This is particularly critical for real-time simulations because the required frame times have now become shorter than can be delivered by conventional real-time simulators based on available real-time operating systems. This developing need for high-speed real-time simulation has resulted in research into alternative methods of achieving frame times for real-time simulation of 1 microsecond or even less. The paper describes research that uses field-programmable gate arrays (FPGAs) in real-time simulations with frame times of less than one microsecond. A complete real-time simulator based on these techniques needs to be integrated with a conventional computer platform that can provide a user interface and also host those parts of the simulation that do not need the speed of the FPGA in a real-time, multi-rate, distributed simulation. The user interface on the host system is required to support development, implementation, setup and control of the simulation. Graphical display of results is also required. The paper describes a number of possible configurations including one in which the FPGA and a Windows system can be combined to produce such a simulator at an affordable cost that would be very suitable for installation in university teaching laboratories.

Keywords: Real-time simulation, power electronics, FPGA.

Presenting Author's biography

Roy Crosbie is a Professor Emeritus at California State University Chico. He has a Ph.D. degree in Electrical and Electronic Engineering from the University of Liverpool and is a Chartered Engineer (UK), a Fellow of IET and a Fellow of SCS. Prior to joining CSU, Chico in 1983 he held appointments at Marconi, Bell Canada, and University of Salford. He was the first Chair of the Dept of Computer Engineering. He was President of SCS from 1988-90. In 2001 he initiated a research program on high-speed real-time simulation aimed initially at power electronic systems which has been supported by the Office of Naval Research since 2004.

1 Introduction

Modern power systems depend on power electronic components that use high-speed switching controllers to provide improved stability, flexibility and conditioning of power supplies. The key components are switching circuits (converters) that convert alternating current to direct current (rectification) and vice versa (inversion). As switching speeds are increased in order to reduce low-order harmonics and improve controllability, the duration of the time steps used in simulations of these power systems must be reduced to avoid significant timing errors. This is particularly critical for real-time simulations because the required frame times have now become shorter than can be delivered by conventional real-time simulators based on available real-time operating systems.

This developing need for high-speed real-time simulation has resulted in research into alternative methods of achieving frame times for real-time simulation of 1 μS or even less. The paper describes research that uses field-programmable gate arrays (FPGAs) in real-time simulations with frame times of less than 1 μS.

A complete real-time simulator based on these techniques needs to be integrated with a conventional computer platform that can provide a user interface and also host those parts of the simulation that do not need the speed of the FPGA in a real-time, multi-rate, distributed simulation. The user interface on the host system is required to support development, implementation, setup and control of the simulation. Graphical display of results is also required.

The paper describes a number of possible configurations including one in which the FPGA and a Windows system can be combined to produce such a simulator at an affordable cost that would be very suitable for installation in university teaching laboratories. Brief details of the essential elements of a high-speed real-time simulator are presented.

2 High-speed real-time applications

2.1 General considerations

The need for special techniques to achieve high-speed real-time simulation arises from the increasingly rapid switching frequencies being used in the design of modern power converters. Higher switching frequencies lead to reduced lower-frequency harmonics in the waveforms and this means that the size and cost of filters used to further reduce harmonic distortion can be greatly reduced.

Real-time simulation is used when there is a need to include hardware or software in the loop. In a real-time simulation the program execution is required to advance in discrete time steps, which are normally of equal duration, and which are synchronized to a real-time clock. This requires that the amount of computation involved in advancing each time step should normally be the same for each time step. This is particularly important in time-critical situations to ensure that any idle time is minimized or eliminated. For applications in which the math model includes differential equations, this means that variable-step integration algorithms, in which the time taken to advance a given time increment can vary, are unsuitable and a fixed-step algorithm is required. The critical focus in these applications is to develop programming that meets the twin requirements of accurate simulation and a computation time within the specified time-step duration.

2.2 Power electronic applications

Power electronic systems are based on switching circuits that either convert an alternating current input into a direct current output (rectifier) or a direct current input into alternating current at the required frequency (inverter). These can be combined in a variety of ways to produce a broad range of power converters from the small converters that power laptops, mobile phones and other small electronic devices to high-power systems used in the transmission and distribution of electric power that may be rated in megawatts.

The research described here has been particularly focused on power systems for all-electric ships which use a large number of converters to match the generated power to a wide variety of loads [1][2]. Fig. 1 provides a simplified representation of a ship power system. Even a simplified system like this involves a significant modeling and simulation effort and a simpler system was defined to act as a basis and benchmark for the initial development of high-speed, real-time simulations. Fig. 2 shows the benchmark, a 3-phase system which couples two a.c. systems via a d.c. link (in the center of the figure). Each converter is represented by 6 switches which turn on and off in pairs at times controlled by a switch controller (not shown). Power can flow in either direction. One of the converters acts as a rectifier transferring energy from the a.c. system to the d.c. link. The other converter acts as an inverter that passes d.c. energy to the other a.c. system. The system also includes filters to reduce harmonic distortion and transformers to modify voltage levels. The math model for this system consists of 23 differential equations, 12 ideal switches, and two PID controllers. A simple investigation of this system can hint at the approximate duration of integration steps required to maintain reasonable accuracy. Assume that the two switch controllers

(one for each converter) operate with a basic switching frequency of 2 kHz. Typical controllers use pulse-width modulation which leads to unequal intervals between switching events for a particular pair of switches (known as a phase leg),

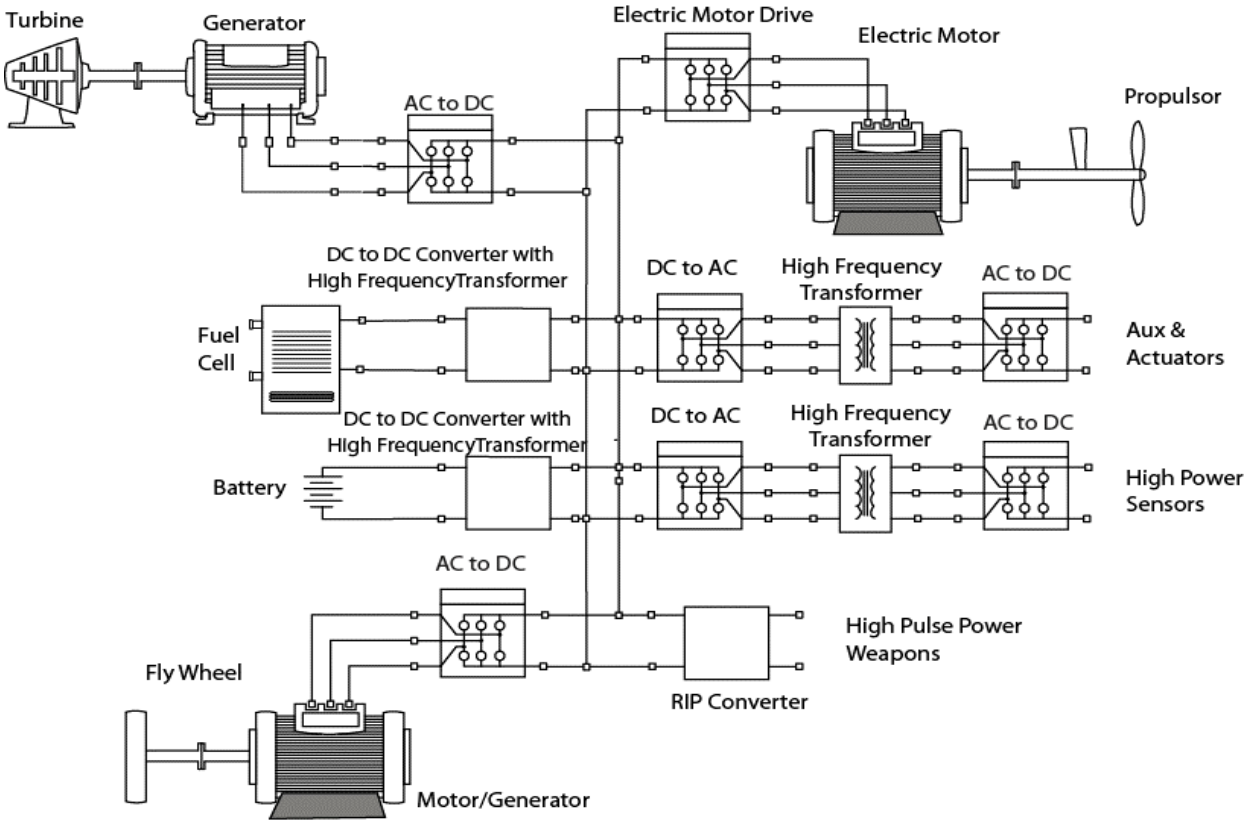

Fig.1 Simplified representation of power system for an electric ship

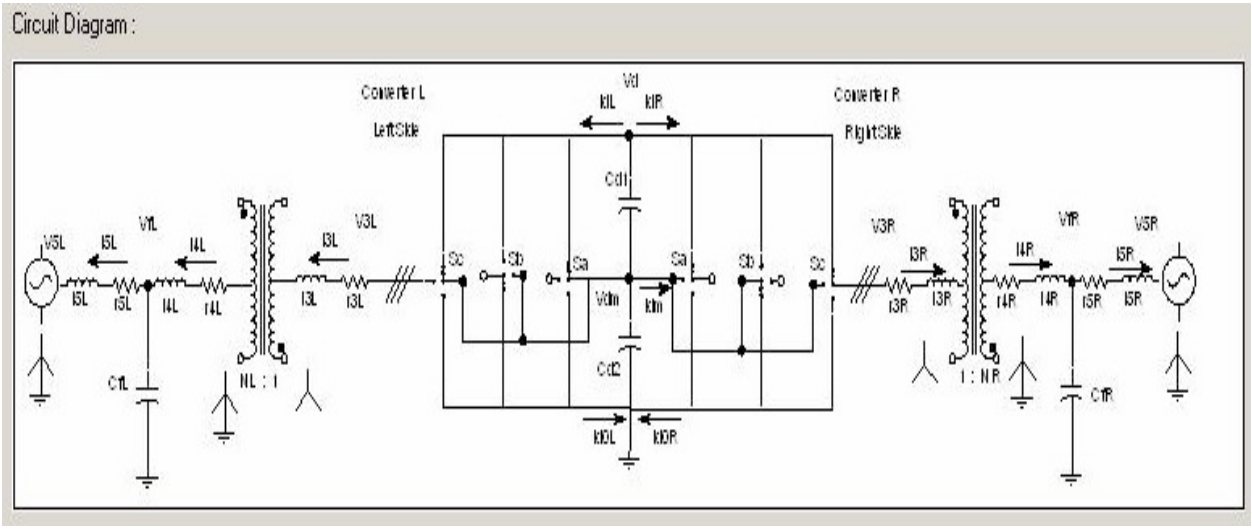

Fig 2: 6-pulse back-to-back converter system used as a benchmark

but on average each converter will experience switching events at a 6 kHz rate. With two converters this becomes 12 kHz which produces a mean interval between switching events of about 80

microseconds. Experience suggests that the integration step size should be no longer than approximately 1/40th of this duration giving a required time step, or real-time simulation frame time, of at most 2 μS. This frame time is not attainable using available real-time operating systems (RTOS), including several real-time variants of Linux. Most of these systems are limited to minimum frame times in the 10-20 μS range. Many power electronic systems are therefore candidates for special high-speed techniques when real-time simulations are required.

3 High-speed real-time techniques

3.1 Basic requirements

The requirements for a real-time simulation system include one or more simulation processors that execute the simulation model code and a host operating system that provides a user interface. The primary requirement for high-speed real-time simulation is that the processor or processors that are executing the simulation are allowed to complete the necessary computations without interruption by the host operating system.

The simulation processor(s) should be capable of executing the calculations required to advance the simulation by one time step within the selected frame time. The operating system must allow the operator to set up the simulation, set parameters both of the model and for simulation execution, select variables for display or other forms of output and control the execution of the simulation.

Conventional real-time operating systems, including those designed specifically for real-time simulation, do not provide absolute priority to the simulation process with the result that so-called operating system jitter can cause interruptions that pre-empt the simulation process for several microseconds. This is the reason for the lower limit of 10-20 μS for frame times for this type of system.

3.2 First alternative: digital signal processors

The first solution to this problem that was attempted by the CSU Chico research group was to use a commercially available board containing 4 digital signal processors (DSPs) that was inserted into the PCI bus of a conventional Windows PC [3][4]. The Windows PC was configured to run the Virtual Test Bed (VTB) [5], which was used to provide a user interface and graphical display capability. The execution of the real-time simulation program on the DSPs was initiated by the VTB and was allowed to continue uninterrupted until complete.

Using this approach a frame time of 2 μS was obtained for the 6-pulse back-to-back benchmark. Achieving the shortest possible frame time depended on a number of factors. Programming was performed using the C language and the optimizing compilers provided as support for the Analog Devices TS 201 processors proved very efficient for these programs after some problems with earlier versions. Numerical integration was performed using a method based on state-transition methods [6] which provided a fixed-step method with superior error and stability characteristics for this class of problems. Great care was required in partitioning the program between the four processors and a very even distribution of workload was achieved.

A review of possible improvements using DSPs concluded that no significant reduction in minimum frame-size was likely in the immediate future and attention turned to the use of field-programmable gate arrays (FPGAs) as an alternative way of achieving sub-microsecond frame times.

3.3 Second alternative: FPGAs

Because there was a demand for real-time power-electronic simulations using frame times of less than the minimum of 2 μS achieved with DSPs, attention was turned to using FPGAs[7]. The FPGA is the latest development in a long sequence of programmable logic devices dating from the early 1970s. The first commercial FPGA was introduced in 1985; it provided 64 configurable logic blocks and 3 look-up tables. The research described here used Xilinx Virtex-5 FPGAs which contain up to 330,000 logic cells and a potential for massively parallel interconnection of its functional units. The research is currently transferring to the Virtex-6, which has about 4 times this capacity.

The huge number of functional units provided by the modern FPGA makes it possible to essentially configure a special processor that is matched to the nature of the program to be executed. This reverses the normal approach of designing a program that matches the problem to the fixed architecture of the processor. As a result FPGAs are now being used to address a much wider range of applications than could be tackled by simpler programmable logic devices, including simulation applications.

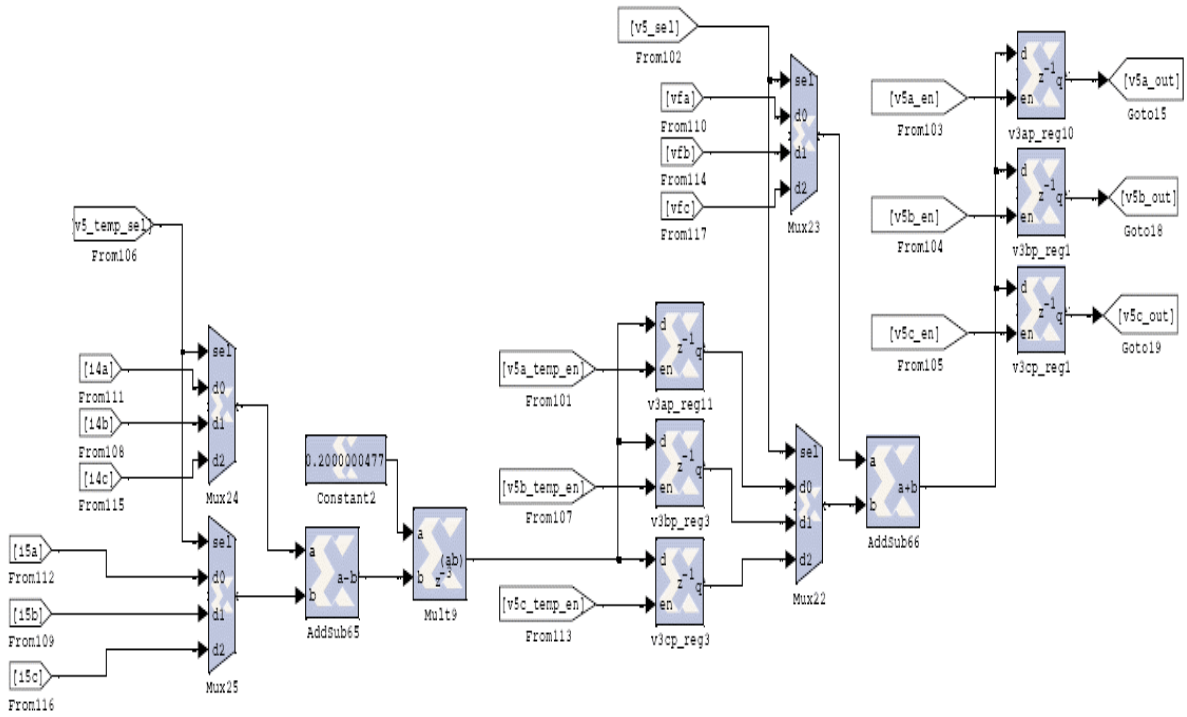

Fig. 3 Part of a Simulink FPGA Toolbox diagram

These advantages come at a cost. The arithmetic units in the FPGA are integer units. Computation is performed in fixed-point arithmetic. Floating-point computation is possible, but it uses much more FPGA capacity to implement and must normally be programmed by the user. Fortunately, it is possible to adjust both the word length and the position of the binary point in FPGA arithmetic which can do much to compensate for the traditional problems (magnitude scaling and round-off error) associated with fixed-point arithmetic.

Two approaches are available for programming the FPGA. One is to use a hardware description language such as VHDL; the other is to program the FPGA using a graphical approach in which a block diagram is constructed of the interconnection of functional units in the FPGA. A Simulink FPGA toolbox is available for the Xilinx devices used and this was the initial approach to be adopted. The choice was influenced by the fact that the math model for these applications consists mainly of difference equations that require the computation of sums of products, for which programming is reasonably straightforward using the graphical toolbox.

The procedure is first to establish a standard state-space form of the differential equations that describe the behavior of the circuit or machine. These are then combined with the numerical integration algorithm to produce a set of difference equations. The form of these difference equations is

a set of next states computed as a sum of products based on the current states. These equations are then programmed into the FPGA.

A section of a Simulink FPGA toolbox diagram is shown in Fig. 3. Multiplexers are used to select pairs of operands for the multiplier and adder units and temporary storage is provided to hold the results of these arithmetic operations. Using the Virtex-5 FPGA the 6-pulse back-to-back benchmark problem was broken down into 7 subsystem units; two converters (left and right); two controllers (left and right), two generators (left and right) and the shunt dc capacitor that connects the two dc sides together. The execution time for each section, measured in terms of the number of 10-nS clocks each requires are: a) controllers and converters, 25 clocks each; b) generators 18 clocks each; and c) shunt capacitor 17 clocks. The converter and generator pairs are set up to run sequentially taking a total of 43 clocks (25+18) for each pair. All other subsystems run in parallel. An additional 2 clocks are required for the data buffer making a total of 45 clocks for the full simulation which yields a frame time of 450 nS.

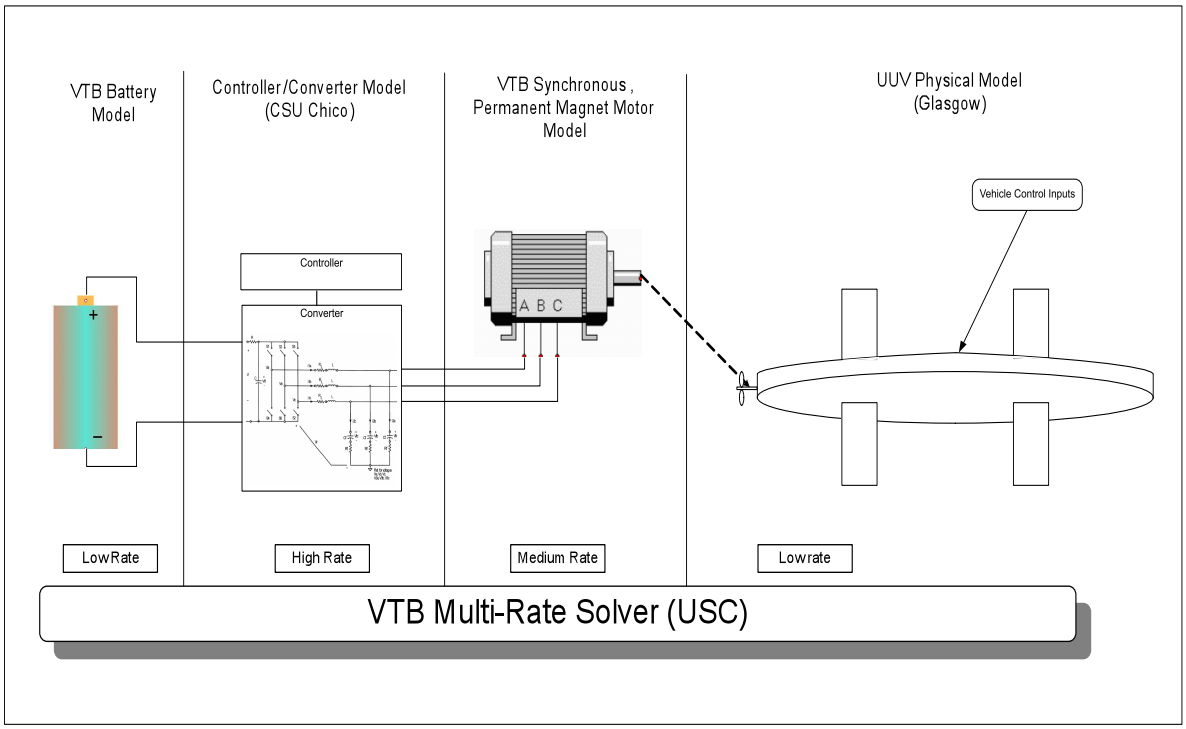

Fig. 4 Simplified representation of UUV model

4 The need for multi-rate simulation

In practical real-time applications, high-speed simulations of power electronic components are combined with simulations of other components, such as motors, generators, batteries, mechanical components etc. that do not require such short frame times and which can be simulated on more conventional computer platforms. The full simulation thus consists of high-speed FPGA-based elements combined with lower-speed simulations executing on, for example, a real-time Linux platform. The complete simulation uses multiple frame rates in a distributed multi-rate real-time simulation [8]. Multi-rate simulation is a familiar technique that has been used for many years in both real-time and non-real-time applications. It is often used as a way of speeding up the simulation. In high-speed real-time applications there is the additional incentive of minimizing the proportion of the simulation that is executed on the high-speed (DSP or FPGA) part of the system.

4.1 A multi-rate application

The second example is a simulation of an unmanned underwater vehicle (UUV) which was developed to test multi-rate techniques (Fig.4). This includes a battery, converter, induction motor, propeller drive and vessel complete with control surfaces. This system has been simulated in real-time with up to 5 frame times varying from 2 µS for the power electronics to 100 milliseconds for the slowest components. A simulation has been developed in which the fast and medium-speed components are simulated in a multi-rate program on the FPGA with the slower components on the host Windows PC.

Ethernet was used to connect the FPGA to the PC in the initial implementation of this simulation. This interface was directly programmed on the FPGA with preformed headers to allow small data packages to be transmitted every 2 µS. A version is planned in which the FPGA board will plug directly into the PCI bus of the host.

4.2 Hardware-in-the-Loop

A version of the UUV simulation has been implemented in which the motor and propeller drive is represented by an induction motor, d.c. generator and programmable load. Although these does not directly represent the actual system components the setup provides a test bed for investigating hardware-in-the-loop implementation. In this simulation the FPGA simulates the high-speed part, the medium speed components are implemented in the hardware, and the slow components run in real-time on a Windows PC that also hosts the user interface.

5 Graphical User Interface

An important part of any simulation system is the user interface which allows the user to develop, set up, control and monitor the simulation. The Virtual

Test Bed (VTB) has been used as the GUI in this project. Since the Virtual Test Bed is a Windows product, all the simulations developed have included a Windows host. Some implementations have also used a Linux-based system to execute parts of the simulation. VTB, developed at the University of South Carolina, offers a graphical editor, powerful user interface, a resistive-companion matrix solver that supports natural couplings between simulated subsystems, and flexible graphical output including 3-D animation. The 3-D graphics has proved particularly useful in providing a real-time visualization of the motion and orientation of the UUV.

6 Software Tools

Additional software support for multi-rate and FPGA-based simulations are under development. At this stage a Mathematica-based tool is under development that can parse a VTB schematic, extract the math model, combine it with the selected integration algorithm to generate difference equations and produce executable code for the target processor including the FPGA. This has been tested for a limited range of problems involving linear circuit elements and switches [9].

A collaboration with the developers of the VTB at the University of South Carolina aims to incorporate these features into an enhanced multi-rate version of VTB.

7 Analytical support

The research has been supported by a strong analytical effort. This has helped to identify the best choice of integration algorithm for the high-speed, multi-rate simulations, including analysis of stability and accuracy of different algorithms and frame-rate combinations. The possibility of adapting these techniques to provide user-support tools is the focus of ongoing research**.**

8 Possible simulator architectures

There are several ways to configure a high-speed, real-time, multi-rate, simulator architecture. The required components are a high-speed simulation unit, host processor, lower-speed simulation capability, high-speed and hardware-in-the-loop interfaces, multi-rate simulation environment and graphical displays. Since the high-speed simulation processors are designed to extend the range of available frame times down to as low as 1 μS, a key decision is how to provide for longer more familiar frame times. Anything above about 20 μS should be within the ability of a good real-time operating system and one possibility is a combination of high-speed processor and, say, a real-time Linux system. This requires that the Linux system also offers a suitable simulation environment and although some commercial systems are configured for real-time simulation they do not, as yet, provide full multi-rate support.

The research described here has made considerable use of the VTB as a user interface and for output graphics, and in some cases as a solver for some system components. VTB is a Windows product so any architecture that uses the VTB must incorporate a Windows system. This leads to two possible options. The most obvious is to combine the FPGA for high-speed, a real-time Linux system for slower speeds (say 20 μS and up), and a Windows system as a host using the VTB as the user interface. These can be fairly expensive solutions and one of the goals of the research is to develop a system that is affordable for educational institutions and small companies. This led to a study of the use of Windows as a real-time simulator, not a promising prospect at first glance. In fact it was discovered that, subject to certain restrictions on the user, real-time simulation was feasible on a typical Windows system for frame times of approximately 10 mS and longer.

This offers a third, low-cost, configuration with one or more FPGAs directly connected to the Windows PC, probably via the PCI bus, either in the PC enclosure or in a separate expansion unit [10]. This approach requires that all frame-times that are too fast for the Windows system must be accommodated on the high-speed processors. Implementations of the UUV simulation have been programmed with multi-rate simulation on the FPGA for the converter, controller, and electric motor and with the battery, vessel, 3-D graphics, and user interface handled by the PC. At present only one frame rate at a time can be handled by VTB and a second PC has been used to display graphs of high-speed variables such as converter voltages and currents. Future versions of the VTB that aim to offer support for multi-rate simulation may make the second PC unnecessary.

9 Summary and conclusions

The rapidly improving capability and power of the modern FPGA promises a revolution in the ways we perform high-performance computing. A few short years ago it would have been a very bold prediction to foresee the way in which these devices can be used for high-speed real-time simulation. Using them in this way is still not easy with many challenges including the trade-off between speed and data formats (variable-length fixed point versus floating point), high-speed interfacing, ease of programming, and integration within a complete user-friendly simulation environment. The good news is that these challenges are clearly identified and progress is being made towards meeting them. The research

effort described here is expected to result in a prototype high-speed, real-time, multi-rate simulator within one year.

10 Acknowledgements

Financial support for the research was provided by the US Office of Naval Research. Contributions from Professor Roger Dougal and the VTB team at the University of South Carolina and Dr. John Pearce of ISIM International Simulation Ltd. are also recognized.

11 References

[1] Crosbie, R. E., J. J. Zenor, R. Bednar, D. Word, and N, G. Hingorani, "Multi-Rate Simulation Techniques for Electric Ship Design", *Proceedings of IEEE ESTS 2007*, Arlington VA, 21-23 May, 2007.

[2] Crosbie, R.E., J.J. Zenor, D. Word, R. Bednar, N.G. Hingorani "Advances in High-Speed Real-Time Multi-Rate Simulation Techniques for Ship Power Systems", *Proceedings of IEEE Electric Ship Technologies Symposium,* Baltimore MD, April 20-22, 2009.

[3] Crosbie R.E., J. J. Zenor, D. Word and N. G. Hingorani*,* "Fast Real-Time DSP Simulations for On-Line Testing of Hardware and Software", *Proceedings oft the Summer Computer Simulation Conference,* Society for Modeling and Simulation International, San Jose, July 2004.

[4] Crosbie, R.E., Zenor, J.J., Word, D., and Hingorani, N.G. "Fast Real-Time DSP Simulations for On-Line Testing of Hardware and Software" *Modeling & Simulation*, vol. 3, No 4, Oct-Dec 2004, SCS, San Diego

[5] Dougal, R. A., Liu, S. Gao, L., and Blackwelder, M. *Virtual Test Bed for Advanced Power Sources,* J. of Power Sources, Vol. 110, No. 2, pp. 285-294, 09/02.

[6] Word, D., R. Bednar, J.J. Zenor, and N.G. Hingorani, "High-Speed Real-Time Simulation for Power Electronic Systems. *SIMULATION: Transactions of the Society for Modeling and Simulation International,* Aug 2008, Vol 84, pp 441-456.

[7] Word, D., J.J. Zenor, and R. Powelson, "Using FPGAs for Ultra-High-Speed Real-Time Simulation" *Proceedings of Conference on Grand Challenges in Modeling and Simulation*, Edinburgh, Scotland, June 16-19, 2008.

[8] Crosbie, R.E., J.J. Zenor, R. Bednar, D.Word and N.G. Hingorani, "High-Speed Real-Time Multi-Rate Simulation", *Western Simulation MultiConference,* San Diego, CA January 2007

[9] Zenor, J. J., R. E. Crosbie, R. Bednar, and D. Word, "Software Support Tools for High-Speed, Real-Time Simulations", *Proceedings of Conference on Grand Challenges in Modeling and Simulation,* Istanbul, Turkey, July 13-16, 2009, SCS.

[10] Zenor, J. J., D. Word, R. Bednar, R. E. Crosbie, N. G. Hingorani, "Progress Towards a Low-Cost High-Speed Real-Time Multi-Rate Simulator" *Proceedings of Conference on Grand Challenges in Modeling and Simulation*, Ottawa, Canada, July 12-14, 2010, SCS.