

# MIXING THE REAL WITH THE VIRTUAL



**THE BENEFIT OF REAL-TIME  
HARDWARE-IN-THE-LOOP  
SIMULATION AND ITS FALLING  
COSTS ARE ENCOURAGING  
A WIDER AUDIENCE  
TO ADOPT THE TECHNIQUE.**

**H**IL (HARDWARE-IN-THE-LOOP) simulation is a technique that combines and interfaces real and virtual components into an operational configuration to simulate and test the dynamic behavior of complex systems. Since the 1950s, mission- and safety-critical systems, such as those developed for the defense and aerospace industries, have successfully employed HIL simulation. The cost for

an HIL platform for these systems can exceed \$1 million; however, its ability to minimize the risk of killing a vehicle operator or destroying an even more expensive prototype system during testing may justify its cost.

During the 1990s, the automotive industry began to employ HIL simulation. According to Chris Wunderlich, a marketing manager at Infineon Technologies, "The cost for each hardware-in-the-loop box or chassis for automotive systems comes in [at] up to \$50,000." HIL simulation enables automotive designers to engage in rapid prototyping and algorithm

tuning for their control systems and to cost-effectively automate the testing of their systems to meet increasingly complex safety regulations and requirements.

The cost and capability of a design team harnessing sufficient computing power to realize real-time simulation continue to improve. Silicon and tool providers are providing increasingly mature development platforms that can scale with each evolutionary generation of a product's design life cycle and can justify and spread the cost of creating a testing platform across those generations. These two factors, lower cost for a

*At a glance*.....58

*For more information* .....60

real-time simulation capability and design teams' ability to use a single development platform for multiple generations of a design, suggest that HIL simulation can and will soon apply to a growing array of applications other than defense, aerospace, and automotive systems. In fact, design teams are beginning to or are already employing HIL simulation for robotics, active noise-cancellation applications, and even consumer appliances.

**WHAT IS HIL?**

Designers use software-simulation tools, such as The MathWorks' Matlab and Simulink, to develop complex electronic control systems. The designer first models the new components in software and then runs simulations of the new component in conjunction with models of the rest of the vehicle to study the behavior of the overall system and to optimize the algorithms and routines of a new component before building a prototype. It is unnecessary, and generally impossible, for the fully software simulation to operate in real-time.

An HIL simulation extends the fully software simulation by allowing the developer to replace portions of it with physical components. The HIL simulation incorporates and manifests the real-time interaction characteristics, such as sampling and time lags, as if the complete real system were operating. Electronic-control units, such as an automobile's an-

**AT A GLANCE**

- ▶ Hardware-in-the-loop simulation integrates real and virtual subsystems in a simulation platform.
- ▶ Hardware-in-the-loop simulation is becoming feasible for more types of applications as the cost of computing power continues to drop.
- ▶ Iterative platform-based designs provide a natural opportunity to consider hardware-in-the-loop simulation.

tilock-braking system, are typical targets for substitution in automotive-application HIL simulation. An aircraft's autopilot system is another electronic-control unit that is a good candidate for HIL simulation.

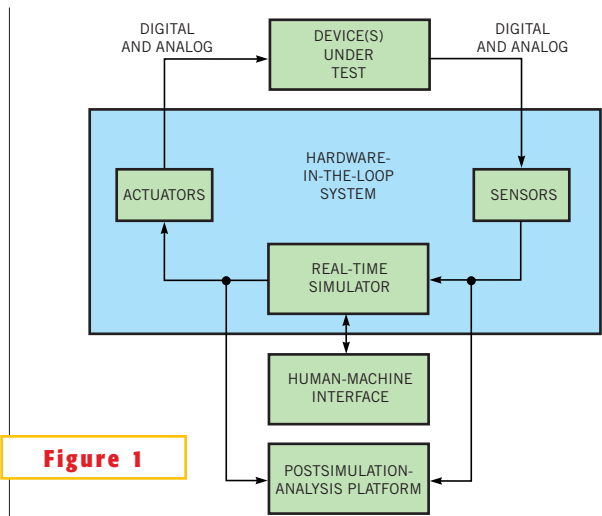
HIL simulation enables developers to test new hardware components and prototypes while interfacing with software models that simulate the rest of the system and environment. The physical components or subsystems respond to simulated signals as though they were operating in a real system because the simulated signals generated by software models accurately and in real time mimic the signals that would occur in the environment and with other real subsystems. Substituting software simulations for the environment and other system components can reduce the complexity

of testing sophisticated controller algorithms over a range of scenarios. HIL enables developers to test a system's control unit under extreme conditions that might be difficult and impractical to perform in the real world, especially if in the real world the test scenario could destroy the system.

A typical HIL platform consists of the DUT (device under test); sensors to capture analog and digital signals coming from the DUT; actuators to condition and send digital and analog signals to the DUT; a real-time simulator to process, model, and generate real-world signals from the simulated portion of the system; a human-machine interface; and a postsimulation-analysis platform (Figure 1). The DUT and simulator must operate together in real time; the emulated sensors and actuators bridge the interface between the real and virtual subsystems.

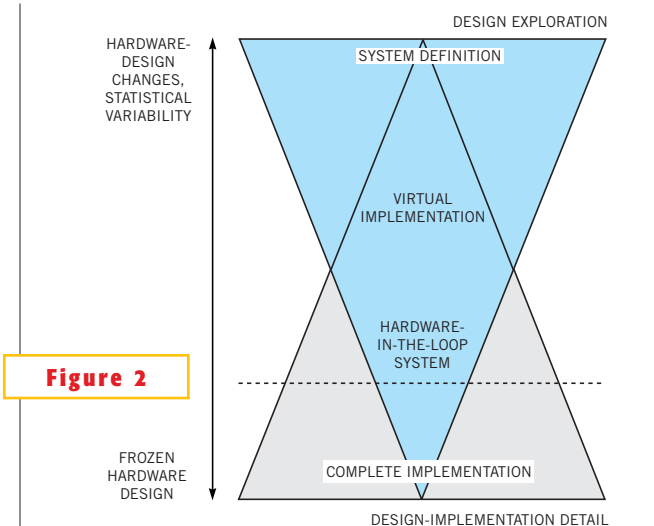
In a closed-loop HIL simulation, the real-time simulator controls the value of each emulated sensor, in hard real time, in response to the outputs from the DUT. Changes in the DUT's control signals affect the real-time simulator's sensor values. Incorrect or late changes to the sensor values by the real-time simulator may invalidate the accuracy of the HIL simulation.

An open-loop HIL (not shown) replaces the real-time simulator with a canned set of sensor values that the system feeds to the DUT controller. During an open-loop HIL simulation, the system



**Figure 1**

An HIL system includes the devices under test, sensors, actuators, a real-time simulator, a human-machine interface, and a postsimulation-analysis platform (courtesy National Instruments).



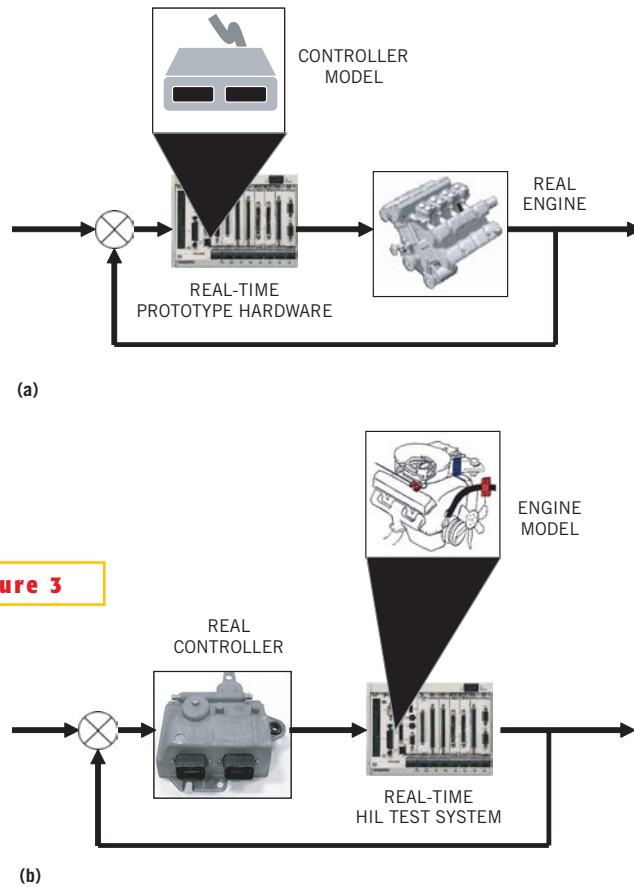
**Figure 2**

The transition from a virtual to a real system involves several stages. HIL simulation supports the gradual and complete integration of real hardware throughout the design and verification process (courtesy Synopsys).

still operates in real time. However, the HIL system captures the DUT controller outputs for postsimulation analysis. Open-loop HIL simulation is useful for testing the interaction between subsystems before developers have integrated all of the functions or components into the system.

Because the functions and the interfaces between the DUT and simulator are application-specific, in the past, these components were completely custom-built. More recently, companies such as Applied Dynamics International, dSpace, and National Instruments began offering off-the-shelf tools and hardware components that, with some configuring, can support HIL simulation. Vivek Moudgal, a manager at dSpace, reveals, "With a modular architecture, hardware-in-the-loop platforms can reuse 95% of the hardware between design generations with the new hardware addressing the interface changes." Many of the tools work with Matlab and Simulink tools.

Unlike the sensors, actuators, and real-time simulator, the human-machine interface and postsimulation-analysis platform are outside the system-control loop, so they need not operate in hard real time. The human-machine interface can act as a window to monitor the simulation during operation



**Figure 3**

**Rapid prototyping supports algorithm development by modeling the controller against a real hardware system or engine (a). HIL testing validates an actual controller against a simulated system on real-time hardware (b) (courtesy National Instruments).**

and can act as a mechanism for injecting manual signals, such as fault conditions, into the simulation. The real-time-simulation data that the HIL system logs can support postsimulation analysis to improve performance and to replay the simulation step by step. The postsimulation-

analysis platform allows a designer to analyze the test data using visualization tools, such as graphs and tables. Some analysis tools, such as from National Instruments, allow designers to import data logs from different test configurations and generate reports using a common template.

**WHY HIL?**

Employing fully software simulation can provide a designer insight into the behavior of a system under varying internal and external conditions. However, for complex systems, it is often impractical or impossible to accurately model every characteristic of a system's behavior. HIL simulation supports the development, verification, and integration of complex electronic-control systems in a systematic and stepwise process—from a fully virtual environment to a fully implemented design (Figure 2). By integrating physical subsystems into the simulation, the DUT introduces true behavior in real time to the system, so the developer can verify the system's behavior

under real operating conditions. The developer can also verify and refine the accuracy of the modeled characteristics that will become part of the software simulation.

HIL simulation is especially effective when there is no other feasible approach

**FOR MORE INFORMATION...**

For more information on products such as those discussed in this article, contact any of the following manufacturers directly, and please let them know you read about their products in *EDN*.

**Altera**  
1-408-544-7000  
www.altera.com

**CPU Technology**  
1-925-224-9920  
www.cputech.com

**Infineon Technologies**  
1-408-501-6000  
www.infineon.com

**Opal-RT Technologies**  
1-514-935-2323  
www.opal-rt.com

**Xilinx**  
1-408-559-7778  
www.xilinx.com

**Analog Devices**  
1-800-262-5643  
www.analog.com

**dSpace**  
1-248-567-1300  
www.dspaceinc.com

**MathWorks**  
1-508-647-7000  
www.mathworks.com

**Synopsys**  
1-650-584-5000  
www.synopsys.com

**Applied Dynamics International**  
1-734-973-1300  
www.adi.com

**Green Hills Software**  
1-805-965-6044  
www.ghs.com

**National Instruments**  
1-888-280-7645  
www.ni.com

**Virtio**  
1-408-341-0844  
www.virtio.com

to integrate and verify a system, and it is ideal for systems that will operate in difficult- or expensive-to-reach environments, such as in orbit or at extreme geographic locations. HIL simulation allows developers to test their control systems with less risk of destroying the system. For example, a developer can test an automobile's antilocking-braking system or an aircraft's autopilot controller without destroying an actual car or plane—even during extreme scenarios. With an HIL simulation, a developer can simulate any environmental condition.

HIL simulation allows a designer to test an incomplete hardware subsystem or software control algorithm to facilitate today's tight development schedules. HIL simulation enables a developer to test the real hardware controller beyond its limits, including safety margins; it also supports reproducible test runs that can assist in uncovering and tracking down race conditions in the interactions between subsystems.

HIL simulation significantly supports the process of collecting usability data from man-in-the-loop testing, for subsystems that will have a human interface. Changes in control-algorithm parameters can result in more or less responsiveness in the control system, which can translate into more or less response for a given input from the man in the loop. The optimal values for the control parameters may be subjective and require input from many man-in-the-loop tests.

#### **BEYOND TESTING**

HIL simulation offers strong support for system integration, and it can safely and reliably provide a platform for performing fault-tolerance, reliability, and endurance testing for electronic-controller subsystems. However, the concepts behind HIL simulation can apply to the entire design cycle of a real-time embedded system, most notably rapid prototyping. A main difference between HIL simulation and rapid prototyping is which part of the system is real-time hardware and which part is modeled (Figure 3). In HIL, designers simulate the environment, engine, or plant model in real time with target hardware for the controller unit. Rapid prototyping implements the environment, engine, or plant in real hardware and simulates the controller model in real time. This reversal also affects the direction that the

inputs and outputs flow through the sensors and actuators.

For system designs that are iterating on the same hardware platform, HIL-simulation techniques can apply to every phase of the development process through strategic use of open- and closed-loop testing, because every design iteration can reuse the work to develop the HIL platform. HIL simulations are also useful in open-loop testing, which enables developers to use simpler I/O models with no complex real-time feedback requirements. Inputs for the controller can reside as canned files, and the HIL platform captures and records the output responses for postsimulation analysis. Open-loop HIL simulation can be sufficient to perform unit-level testing of the application software. To verify the software requirements, a developer can check the outputs from the software for a set of inputs. Open-loop HIL simulation can benefit from the use of test scripting and some level of dynamic input generation. This type of testing naturally extends to combining and integrating these software units.

The ability to work in hard real time is a key component of HIL simulation. The computational requirements for hard real-time simulation are high and can limit the level of detail a real-time simulator can achieve; however, as the cost-effectiveness and availability of computational power continue to improve, implementing HIL simulation for more application designs will become more and more feasible. HIL-tool and -component providers are positioning their products to support applications in which HIL simulation is becoming more practical each day. □



#### **AUTHOR'S BIOGRAPHY**

*Technical Editor Robert Cravotta worked in the early 1990s with hardware-in-the-loop simulations as part of developing small autonomous spacecraft. You can reach him at 1-661-296-5096 and via e-mail at rcravotta@edn.com.*

#### **TALK TO US**

*Post comments via TalkBack at the online version of this article at [www.edn.com](http://www.edn.com).*