NATIONAL INSTRUMENTS
APPLICATION NOTES

Key Considerations for Powertrain HIL Test

Abstract

Safety, availability, and cost considerations can make performing thorough tests of embedded control devices using the complete system impractical. Hardware-in-the-loop (HIL) simulation is a real-time test technique used to test these devices more efficiently. During HIL test, the physical system that interfaces to an embedded control device is simulated on real-time hardware, and the outputs of the simulator mimic the actual output of the physical system. The embedded controller "thinks" it is in a real system. HIL simulation meticulously tests embedded control devices in a virtual environment before proceeding to real-world tests of the complete system. This application note covers recommended best practices for powertrain HIL testing.

NATIONAL INSTRUMENTS

Introduction to Engine ECU HIL Test

What is HIL Simulation?

In a closed-loop control system, the current state of the system being controlled is fed back to the controller through sensor measurements. In the case of an automobile, an electronic control unit (ECU) uses these measurements to determine the appropriate actuator values to attain a desired operating condition. For example, the engine ECU receives information regarding throttle position, engine speed, and exhaust oxygen levels to determine the appropriate actuator commands for fuel injector position, spark plug timing, and air intake to maintain maximum engine performance and minimize harmful emissions. Physical test of the complete system is ultimately required; however, engineers can use HIL simulation to thoroughly test the ECU and gain critical insight without a vehicle or even an engine.

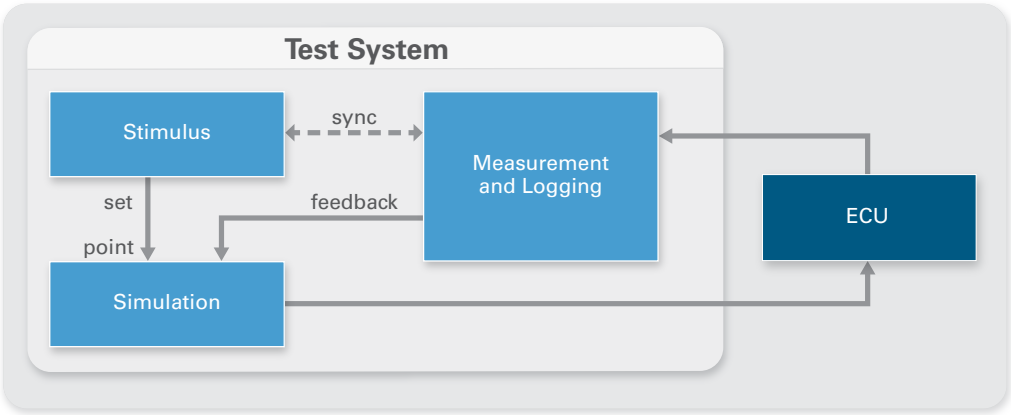

Figure 1. A Hardware-in-the-Loop Test System Contains Several Key Functions When Simulating the World Around an ECU

Increasing Efficiency With HIL Simulation

HIL simulation replicates the physical system with which the ECU interacts and performs test executive functions such as stimulus generation and test result data logging. Stimulus generation of an engine ECU might entail creating a series of desired engine speeds (to simulate pushing the gas pedal) and vehicle loads (to represent a variety of road conditions). The entire system is observed to ensure that all components operate correctly, and the test results are logged to compare with ideal system responses.

Though HIL simulation does not entirely replace the need for physical testing, it does help reduce test cost and improve product quality by helping you:

- **Test earlier in the development process**—identify design errors earlier when they are less expensive to correct and have a smaller impact on time to market
- **Reduce test cost**—reduce capital, repair, and maintenance expenses for test fixtures without the need for a physical system
- **Increase test coverage**—test ECUs under extreme conditions that might not be practical for physical testing because of safety or equipment damage concerns
- **Increase test flexibility**—expand test capabilities regardless of external factors (ex: simulate winter road conditions for a vehicle under test even in the heat of summer)
- **Increase test repeatability**—isolate deficiencies in an ECU even if they occur only under certain circumstances

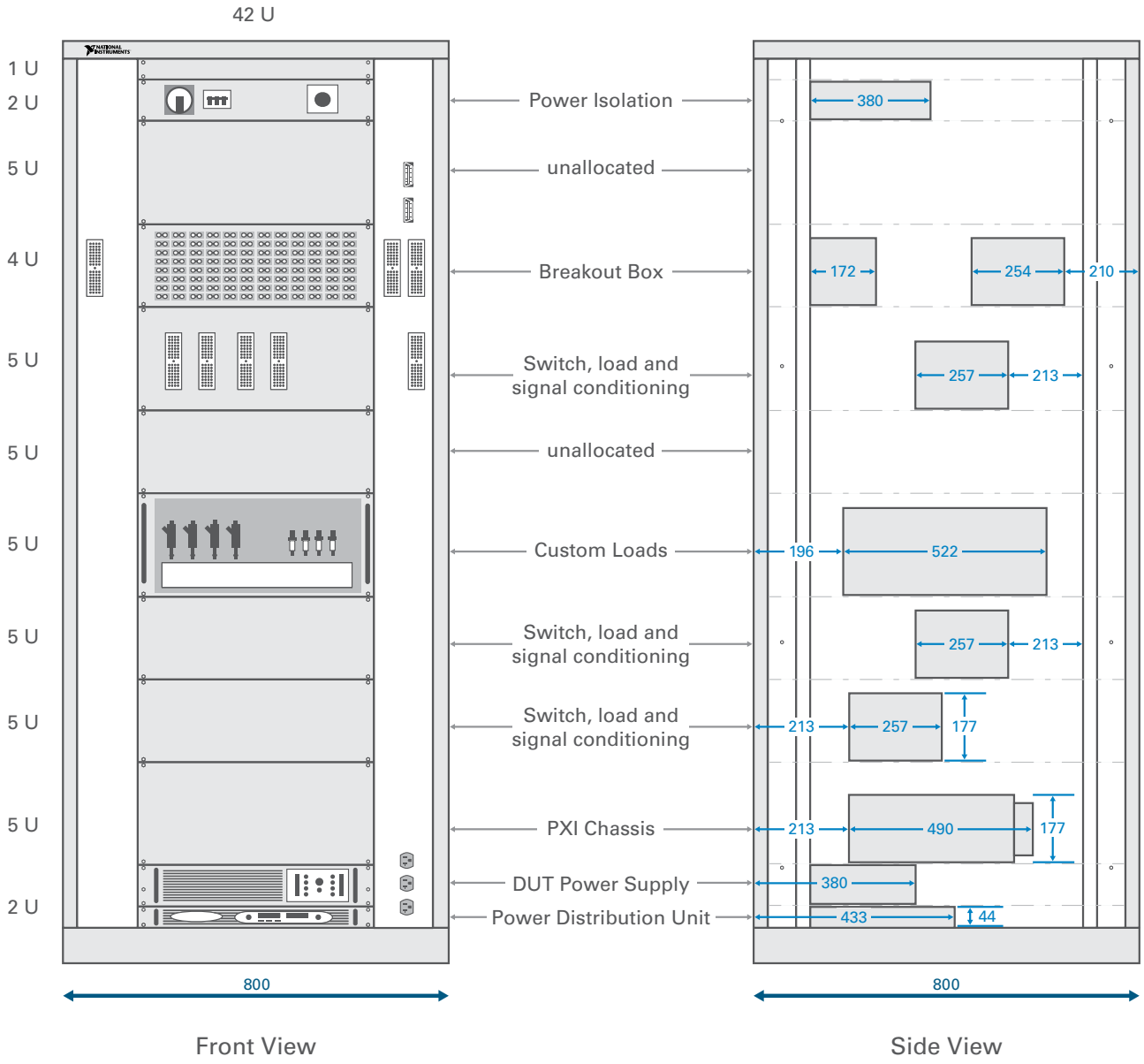

Figure 2. A Typical HIL Test System Contains Many Common Hardware Components

HIL Test Software Fundamentals

The fundamental motivation for organizations to adopt HIL simulation is to improve the operational efficiency of their development and test processes. The hardware and software tools used for HIL simulation need to enable engineers to focus on testing ECUs, not configuring, supporting, and maintaining the test systems. HIL test software should balance ease of use and flexibility to adapt to evolving requirements. The value of an HIL system is defined by how much time it saves as well as how much it improves your product.

In an industry with extremely strict time-to-market requirements, you need to get to your first test as fast as possible. You must quickly and easily connect a simulation model to physical I/O devices as well as update your configurations when ECU parameters change. In addition to configuring it, your system should allow you to create and execute test profiles to exercise the ECU. To drive the virtual car, you must be able to provide patterns of signals that represent environmental variables the ECU might encounter. For example, if you want to perform an EPA drive-cycle test, you need to generate the speed setpoints and road conditions required to run the test. You can choose from many options to generate test profiles,

ranging from traditional programming and scripting languages to graphical displays and sophisticated data playback. The best methods often vary based on ECU requirements, and HIL test systems should accommodate these different requirements.

Simulation Fundamentals

Plant Models and Model Sources

Simulation models are used extensively in HIL test to "trick" the embedded device under test (DUT) into acting as if it is controlling a real mechanical system. Mathematical tools are used to replicate the real-world physics of a mechanical system, and the signals generated by these tools are sent over electrical lines and connected to the DUT. In the past, physical simulation was an extremely challenging process that required in-depth knowledge of the underlying mathematical principles associated with mechanical processes such as fluid dynamics, stress characteristics, and material properties.

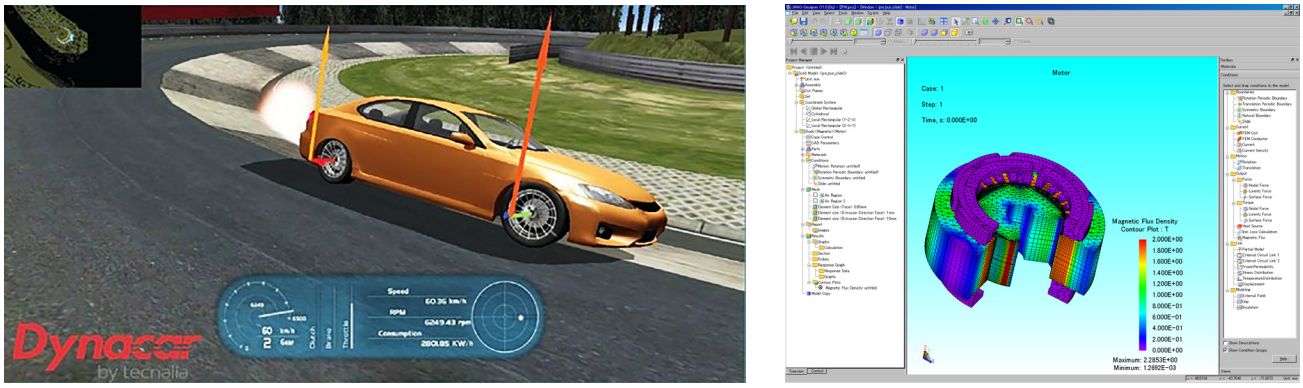

Figure 3. Various Modeling Environments Can be Used to Ensure Efficient and Effective HIL Test

Today, you can choose from many commercial off-the-shelf (COTS) tools to model physical and electrical systems without knowing the underlying math used to create the systems. There are many tools for physical simulation, especially in the area of automotive powertrain HIL test. Common automotive modeling tools and their primary uses include the following:

GT-SUITE by Gamma Technology—GT-SUITE is a simulation software tool with capabilities and libraries for a variety of automotive engineering applications ranging from fast concept design and detailed system or subsystem/component analyses to design optimization and root cause investigations. GT-SUITE architecture gives you the unique ability to build integrated models of systems, including integration across the subsystems you are modeling, physical domains, and modeling levels.

BOOST and CRUISE by AVL—BOOST is a fully integrated virtual engine simulation tool with advanced models for accurately predicting engine performance, acoustics, and the effectiveness of exhaust gas after treatment devices. By using this in your engine development, you can deliver the required torque and power for a given vehicle concept along with optimized emissions, fuel consumption, and passenger comfort (acoustics and transient behavior).

CRUISE is a system-level simulation tool for everyday tasks in vehicle system and driveline analysis throughout all development phases from concept planning to launch. Its application envelope covers the range from conventional vehicle powertrains to highly advanced hybrid systems and pure electric vehicles. CRUISE provides parameter optimization and component matching, which help you achieve practical and attainable solutions.

SimulationX by ITI—SimulationX standard software assesses the interaction of all technical system components. With ready-to-use model libraries for 1D mechanics, 3D multibody systems, power transmission, hydraulics, pneumatics, thermodynamics, electrics, electrical drives, magnetics, and controls, SimulationX is the universal CAE tool for modeling, simulating, and analyzing physical effects.

CarSim by Mechanical Simulation Corp—CarSim simulates the dynamic behavior of passenger cars, race cars, light trucks, and utility vehicles. It animates simulated tests and outputs over 800 calculated variables to plot and analyze or to export to other software such as Excel.

Ensuring Accurate Simulation Through Determinism

The effectiveness of HIL test depends on how accurately the simulation reflects the world around the ECU. Simulation models must provide mathematically accurate responses to ECU commands, and those responses must occur within a timeframe that is consistent with the mechanical system being simulated. For this reason, most HIL applications need deterministic execution using a real-time system. You also can use a real-time system to execute real-time test sequences for gaining valuable insight into the functionality and robustness of the ECU. If the real-time system can deterministically represent the mechanical system with high enough fidelity, you may also be able to calibrate the ECU parameters to optimize and tune the performance of the entire closed-loop system. The performance of the real-time system largely determines the amount of testing that you can move from the dynamometer to the lab, which directly impacts test cost and time to market.

Key Considerations for Different Types of Powertrains

Performing HIL test on powertrain control modules (PCMs) introduces new system requirements. Because of their specialized role, PCMs rely on dedicated coprocessors in addition to a generic microcontroller. For example, a combustion engine PCM must handle high-speed engine-specific signals such as rotational position, knock, cylinder pressure, and precise actuator control [2]. PCM test requires a correspondingly sophisticated test system capable of exercising these unique I/O sets and coprocessors. Therefore, PCM test systems, like the units under test, use dedicated coprocessors to provide sufficient I/O sophistication. Additionally, though companies continually release new PCM hardware and software, test systems are typically used for many years. Enormous flexibility is a fundamental test system requirement.

Today, FPGAs are the ideal devices to address these needs. Their high performance and flexibility are perfectly suited to the rapidly changing test needs of today's advanced PCMs [3]. FPGAs offer significant advantages such as parallel processing, design scalability, ultra-fast pin-to-pin response time, design portability, and lifetime upgradability. All of these benefits translate into a powerful, adaptive test system. However, FPGA programming is typically the role of a specialized engineer who is a valuable and limited resource. The availability of high-level FPGA programming software and readily accessible premade FPGA libraries has greatly increased the feasibility of deploying FPGA-based test systems.

Combustion Powertrains

Fundamentally, a combustion ECU is responsible for turning the engine. To do so, it monitors engine position via sensor feedback from carefully designed encoder wheels, such as the crank wheel shown in Figure 4, and activates fuel injectors and spark plugs to generate power.

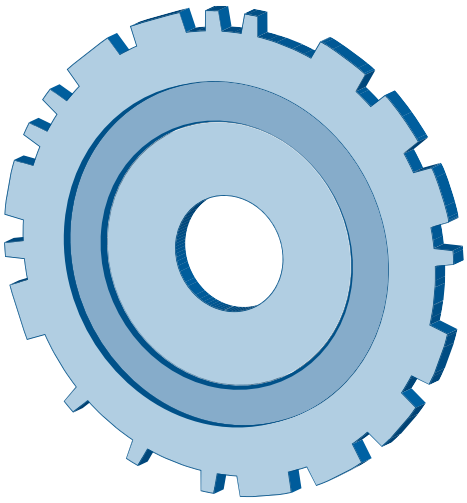

Figure 4. Today's crank wheels can feature complex patterns, such as this one that is unique and changing.

The defining responsibility of a combustion ECU HIL tester is the measurement and generation of these combustion-specific signals, in addition to measuring user inputs such as pedals. Table 1 shows typical engine ECU signals.

Name	Type and Tester Signal Direction	Description
Accelerator pedal position (APP)	Analog output	Driver foot pedal
Airflow	Analog/digital output (depends on sensor type)	Measurement of air mass entering engine
Intake manifold pressure (IMP)	Analog output	Affects air density
Intake manifold temperature (IMT)	Analog output	Affects air density
Fuel pressure	Analog output	Affects fuel dispensed per duration of injector actuation
Crank	Analog/digital output (depends on sensor type)	High-speed signal; rotational position information
Cam	Analog/digital output (depends on sensor type)	High-speed signal; rotational position information
Lambda/O2	Analog output	Exhaust chemistry feedback
Knock	Analog output	High-speed signal; cylinder vibration feedback
Throttle position	Analog output	Throttle body feedback
Throttle command	Digital PWM input	ECU's throttle setpoint
Spark	Analog or digital input	High-speed signal; pulse edges indicate charge begin and discharge location in relation to engine position
Injector	Analog or digital input	High-speed signal; pulse edges indicate fueling begin and end duration in relation to engine position; can be multiple pulses per cylinder per rotation

Table 1. These common engine ECU signals should be considered during tester build.

Figure 5 shows the block diagram of a typical engine ECU tester with loads and switching removed. The tester comprises a real-time OS running on a CPU to perform low-speed (1 kHz) to medium-speed (10 kHz) modeling. The CPU is paired with discrete I/O such as analog, digital, and bus communication for low- to medium-speed signals updated synchronously to model execution. These are the core components of any ECU test system, but to specifically address combustion ECU test, an FPGA coprocessor instantiating an angle processing unit (APU) is added.

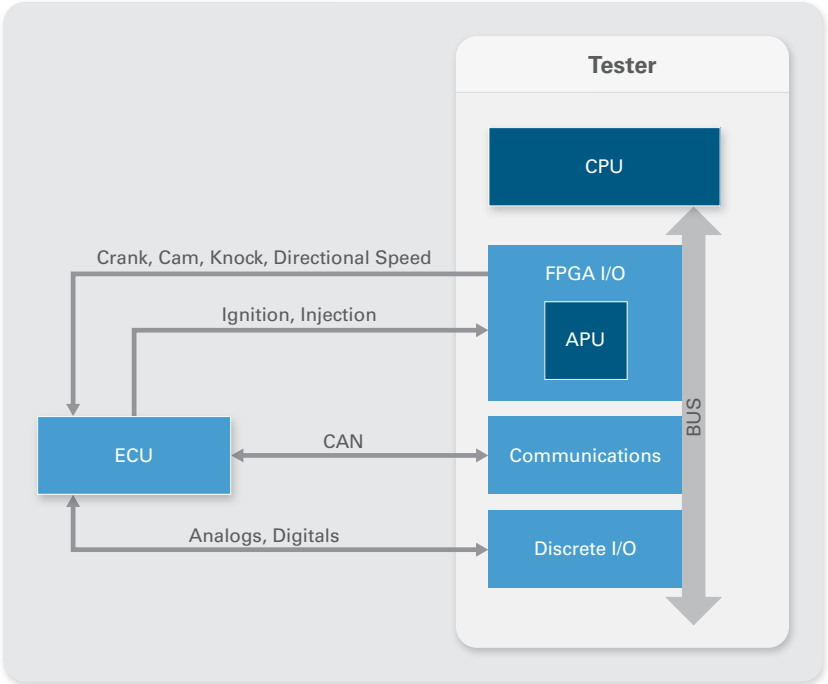

Figure 5. A typical engine ECU tester block diagram showcases the APU coprocessor instantiated on an FPGA.

An APU performs high-speed, high-fidelity engine rotational simulation. Rotational simulation is the process of accepting a speed input value and, over time, continuously publishing a value of simulated rotational positions between 0 and 360 degrees. Because ECUs are programmed to control the engine in relation to its rotational position, ECU verification requires simulating rotational positions in the tester and making measurements correlated to the rotational positions.

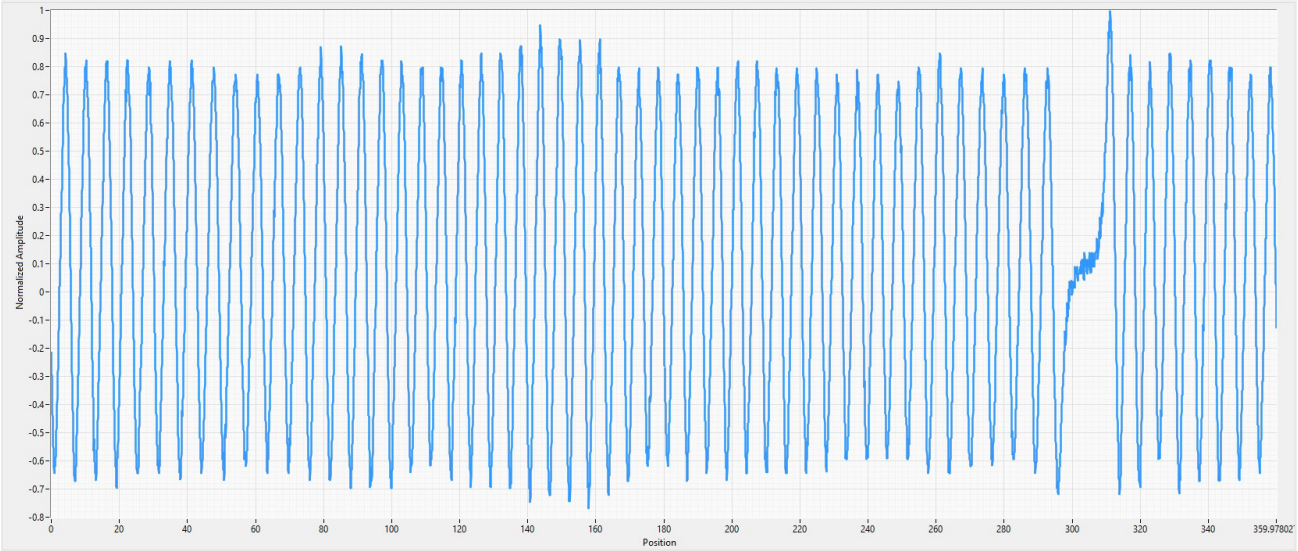

Figure 6. Rotational signal from a simulated Variable Reluctance Sensor.

Offloading rotational simulation from the CPU has many benefits. First, with dedicated hardware, an APU can run at very high speeds without interference from high-level real-time OS constructs such as a thread scheduler. To achieve 0.1 degree of resolution at 10,000 rpm, a rotational simulation must run at a minimum of 600,000 Hz, which is impractical on a general-purpose CPU.

Second, you can run the APU and CPU asynchronously. This allows the CPU to run physical plant models with regular time-stepped intervals, which work better for many plant modeling and real-time OS toolchains, while harvesting angle-based information from the APU coprocessor.

Finally, by locating the APU close to the I/O pins, you can achieve a low-latency connection between rotational simulation and pertinent data to correlate with a simulated position. Latency between when an event happened and when it was correlated to a position directly results in measurement error. You can avoid it by placing the APU inside the same FPGA fabric as the I/O.

Combining the previously discussed ECU signals, an APU coprocessor on an FPGA, and an engine physics model completes a closed-loop engine ECU HIL tester. Figure 7 shows a diagram of data flow in this system using real actuator loads.

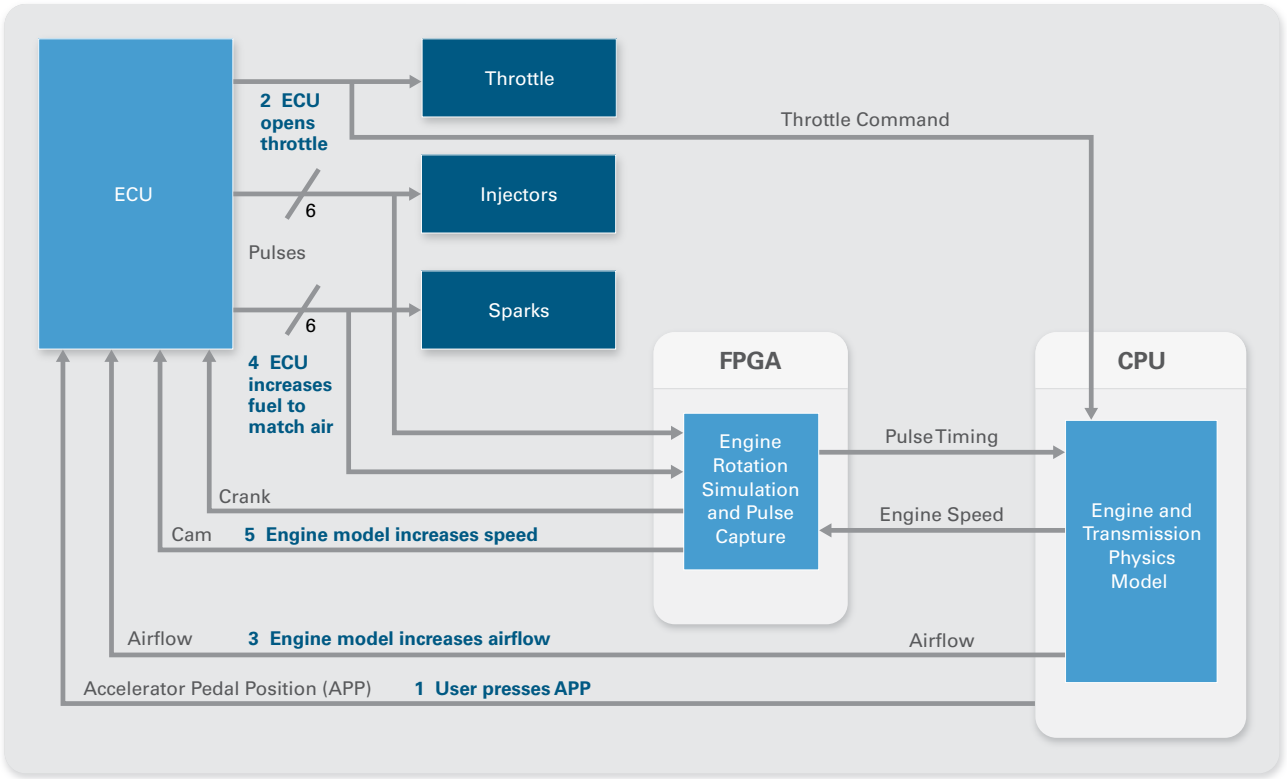

Figure 7. Engine ECU HIL Closed-Loop Data Flow

Measuring fuel injector actuation is another important consideration for the ECU HIL system of a combustion powertrain. One of the best ways to do this is to include the real actuators inside the test system just as they are in a vehicle. Pass the ECU wires through a current measurement signal conditioning card on the way to the injector, and provide the current measurement to the test system's FPGA. This allows the FPGA to measure the current flowing through the injector to determine when the solenoid is considered on and off (Figure 8).

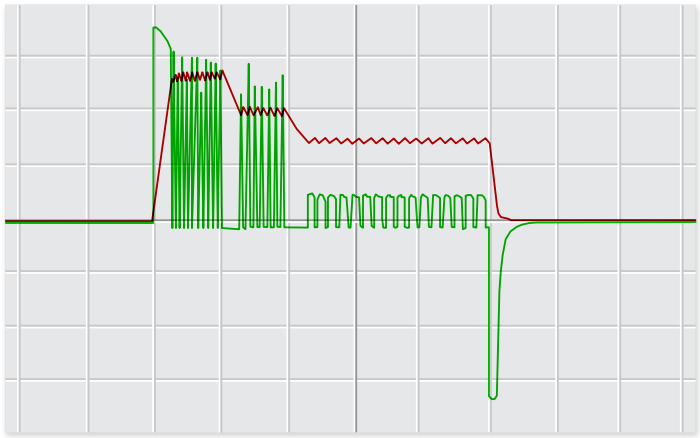

Figure 8. A current and voltage trace of a diesel fuel injector highlights that current must be measured for accurate timing capture

For gasoline injectors, an easier alternative is to directly measure the digital outputs of the ECU to detect when the injectors are commanded on and off. However, measuring the current through the actual injector is more accurate because it requires a certain amount of activation current for the solenoid to open. Additionally, most ECUs throw diagnostic faults if real loads are not attached to their injector outputs.

Hybrid and All-Electric Powertrains

In many all-electric or hybrid powertrains, the ECU has to manage the power produced by multiple independent sources. For example, a hybrid drivetrain includes one or more electric motors alongside an internal combustion engine. No matter which type of hybrid drivetrain you're using, this means that the ECU must control, in a safe and repeatable manner, two coupled plants with potentially very different speeds of dynamics. This requires extensive testing to ensure control system stability.

For example, under icy driving conditions, a wheel can suddenly lose traction. During acceleration, this can cause a dramatic increase in motor speed that needs to be handled safely. However, this safety behavior cannot be physically reproduced on a dynamometer and is time-consuming and difficult to reproduce on a test track. Since complex control algorithms for specific safety conditions like this must be developed and verified, the test needs to account for outlying operating conditions to satisfy the quality level required for a production model vehicle.

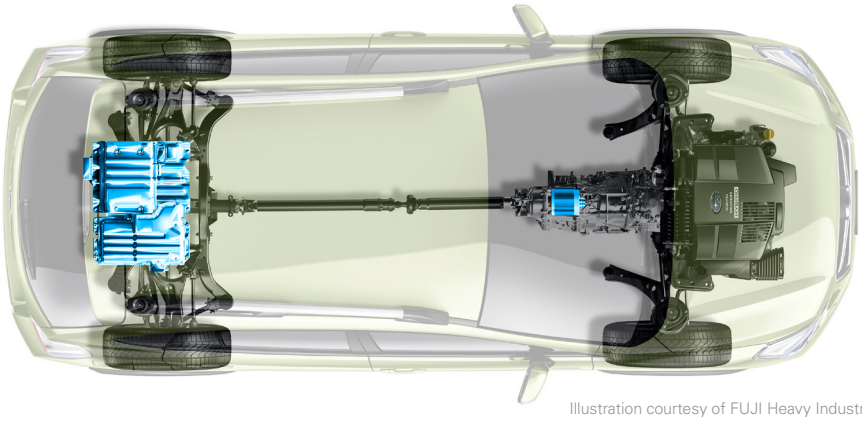

Illustration courtesy of FUJI Heavy Industries, Ltd.

Figure 9. Hybrid Powertrains Introduce New Considerations for HIL Test

Both hybrid and all-electric drivetrains add complexity to ECU test. In each case, driving an electric motor requires the ECU to generate high-speed PWM signals to drive the power electronics hardware. For the HIL test system to respond accurately to those high-speed digital signals from an ECU under test, the simulation must run at very fast loop rates on the order of magnitude of 1 µs. To complicate matters, electric motors exhibit complex nonlinear behavior, such as magnetic saturation and cogging torque, which can be difficult to model directly. You can use a linear model to test an ECU for basic functionality, but you also need to model more complex behavior for more stringent testing, tuning, and optimization.

Traditional simulation systems, which cannot reach 1 µs loop rates, limit the test capabilities of control system designers and force them to rely heavily on costly dynamometer or field testing. In cases like the loss of traction, the field testing required to ensure safety at all of the potential operating points can be incredibly expensive or even impossible. However, improving simulation speed and fidelity helps you conduct more repeatable testing in simulation, which reduces the time and cost of the physical testing level.

Reaching the 1 µs range for simulation periods requires a paradigm shift in the design of electric motor and power electronics HIL test systems. The key approach to simulating systems at such high rates is moving from traditional processor-based HIL systems to FPGA-based simulators.

Traditional processor-based HIL systems offer maximum speeds of only around 50 kHz because a communication bus separates the processor and the I/O. During a single time step of the simulation, the inputs are sampled, that data is transferred to the processor, the result is transferred back to the I/O node, and the outputs are updated. On a PCI or PXI bus, the latency of the communication can typically take up to three quarters of the overall simulation period. Moving the calculations to an FPGA increases the speed of the calculation itself. However, the biggest boost to speed comes from collocating the processing node and the I/O node on a single device to minimize communication latency.

The next challenge you face when conducting a real-time simulation of advanced motor drives is achieving an adequate combination of simulation fidelity and speed. Whereas a simple constant parameter or linear model can be sufficient to conduct functional-level HIL tests, you often need increased simulation fidelity for more robust testing and optimization of advanced motor drives. An effective way to increase simulation fidelity without adding to calculation complexity is to replace the model parameters with lookup tables and update those parameters on every simulation iteration.

Using finite-element analysis results or experimentally derived tables, you can simulate complex nonlinear behavior, such as cogging torque or magnetic saturation, and design a controller that responds appropriately to the complex phenomena. In each of these cases, the lookup table captures the complex behavior without modeling it directly in the simulation.

Maximizing Test Coverage

Building Test Cases

Developing a test plan and test cases for an ECU involves close collaboration between the design and test teams. Documentation of an ECU's requirements is a key step in this process. Typically, these requirements can be organized into three high-level categories: safety, functionality, and performance.

Safety

During product design, a process called failure mode and effects analysis (FMEA) is applied to identify, qualitatively, the possible failures and their overall effects on the system. Developed in the late 1940s by reliability engineers for military systems, FMEA is still used today. When a possible failure is identified, it is described in detail and assigned a probability value, severity value, and calculated overall risk value (the product of probability and severity). Table 2 shows an example of an FMEA table.

Description of Component or Subsystem	Failure Mode (Hazard)	Symptom	Effect	Probability of Failure	Severity of Effect	Risk Index
Component A	Incorrect Wiring	Incorrect or zero output or moto just spins	May cause system failure or damage to device	D	III	III–D
	Short Circuit	Pins soldered together	May cause system failure or damage to device	D	III	III–C
Component B	Blown fuse	No output from board to motor	Board will not send signals to the motor	C	IV	IV–E
	Faulty terminals	No output from faulty terminal	Board will not send correct signals to the motor	E	IV	IV–E

Table 2. Design engineers create failure mode and effects analysis tables as effective safety test case starting points. (Table courtesy of Quanser)

After completing the FMEA, design engineers add features to mitigate the riskiest items identified. For example, they can add a sensor to detect the failure of a mechanical component, and software can then automatically transition the vehicle to limp home mode to prevent further damage. Testing these mitigations is a key part of ECU verification and validation. To design test cases for each item, you need to obtain the fault tree analysis (FTA) from the design team. Figure 10 shows an example FTA.

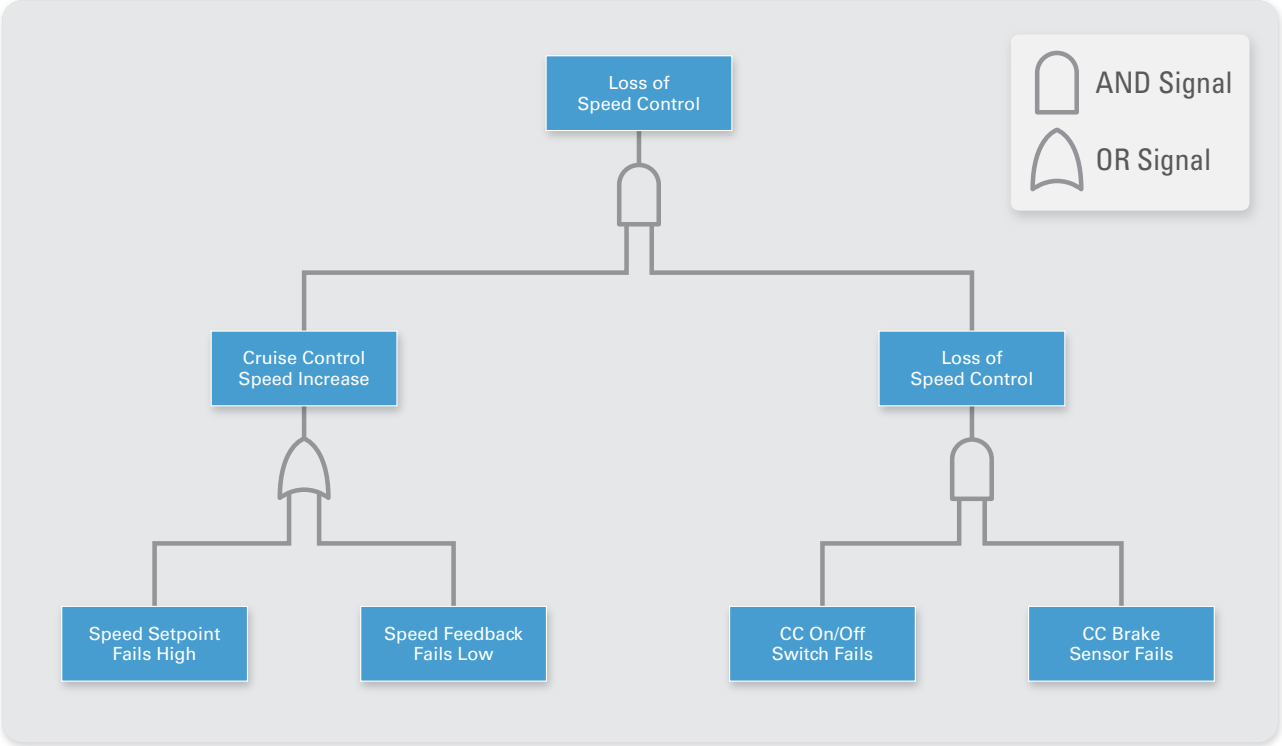

Figure 10. A fault tree analysis is an important reference document when creating safety-critical test cases.

Using the FTA as a flowchart, you can design test cases for each FMEA item of high enough risk (the threshold of "high enough" being set by product management). Considering how dangerous some of the failures can be, the ability to test these in HIL simulation is an investment well made.

When validating to safety requirements, make sure that the test is accurate, especially for regulated industries like automotive and aerospace that enforce functional safety standards. Incorrectly verifying something could lead to extremely negative consequences when the project is in production and in a safety-critical situation. Also, because of the general increase in electronic complexity, you have more to test in the same amount of time, which introduces the need to automate some of the verification. But how do you know if the test automation tools you are using are working as expected? The cost to develop your own test automation tool can be very high, especially if you want to ensure that it is designed with functional safety in mind. On top of all this, you need specific and thorough documentation for overall validation. Correctly creating this documentation can be time-consuming, so any tools you use need to produce the appropriate artifacts, which leads to the assumption that manual tool qualification is the only way.

Using COTS verification tools that are qualified for functional safety projects in specific ways can address this need and give you the appropriate confidence in your test tool. NI Alliance Partner CertTech has done this by creating a qualification kit for NI test automation software tool TestStand. TestStand is ready-to-run test management software designed to help you develop, execute, and deploy automated test systems faster. Because of CertTech engineers' extensive experience in regulated industries and functional safety standards, they understand the requirements for using qualified tools specified by standards like DO-178C and ISO 26262. The qualification kit for TestStand provides comprehensive requirements and test coverage for the most commonly used features, a full suite of tests that verify the provided requirements, and a readily extensible framework, so coverage may be extended as needed. In addition, CertTech produces the required documentation from the tool used as a necessary artifact for compliance. This documentation is essential because the overall goal is complete transparency for the verification process so the test can be re-created. CertTech can cut the time needed to produce this documentation by 95 percent using a qualification kit.

Some of the newer functional safety standards like ISO 26262 and DO-178C require projects to use "qualified tools" for verification and validation activities that will not be manually reviewed, which makes using qualified tools like TestStand more important. These standards require you to assess the overall impact of the tool not testing properly and then assign what standards like ISO 26262 call the Tool Confidence Level (TCL). Two main components determine the TCL: Tool Impact (TI) and Tool Error Detection (TD). TI1 and TI2 are the two classes of TI. TI1 is chosen when the malfunctioning software tool cannot possibly violate a safety requirement. For all other cases, TI2 is chosen. TD levels range from TD1 to TD3. TD1 is chosen if there is a high degree of confidence an error will be detected, TD2 for medium, and TD3 for low. The different TCL levels for a test tool mean the extra burden placed on the user fluctuates.

		Tool Error Detection		
		TD1	TD2	TD3
Tool Impact	TI1	TCL1	TCL1	TCL1
	TI2	TCL1	TCL2	TCL3

Table 3. Tool Confidence Levels in the ISO 26262 Standard Imply Different Amounts of Work in Qualifying a Tool

TCL2-designated tools provide the most value to users because any tool with a TCL1 classification either does not have a substantial impact on safety or already has a high degree of confidence and does not need much extra qualification and documentation. In contrast, a tool with a TCL3 rating generates a low degree of confidence, so it will need some sort of manual qualification no matter what.

Functionality

At a high level, testing ECU functionality is simple. Feature-by-feature testing is easy enough to understand; however, details of the embedded software can illuminate potentially vulnerable times or transitions that require aggressive testing. Therefore, designing functional tests, again, relies on close collaboration with the ECU design team.

The features of a particular ECU can then be combined with the state diagram to guide test case creation.

Performance

Unlike safety and functional test cases, developing performance-based tests without ECU design team collaboration can be helpful. The development of these tests should often be implemented from the user's perspective. What may be acceptable to the design team may be unacceptable to a user, which is important feedback to capture. Thankfully, some user-based performance concerns such as miles per gallon (MPG) have federally defined test procedures that can be directly implemented as a test case. Figure 11 shows the vehicle speed over time graph that must be followed as a federal test procedure for city driving (FTP-75).

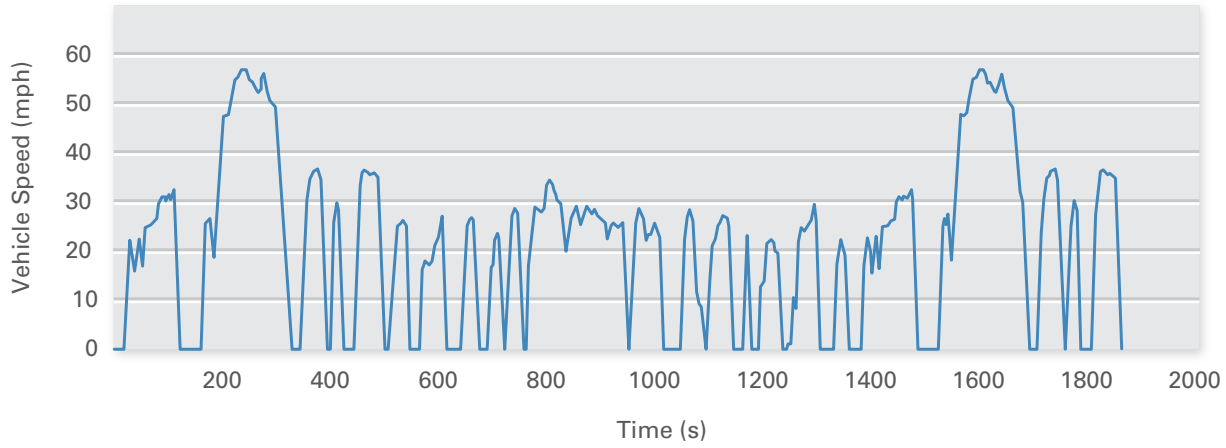

Figure 11. Federal governments typically provide standardized tests for vehicle MPG performance, such as this FTP-75 drive cycle.

Another performance-based test is internal performance, such as bus message timing, ECU CPU utilization, or ECU event response time. These types of tests can require additional tester capability, including an ECU calibration or a debug data link to read microprocessor parameters and/or the ability to post process data-log timestamps to verify acceptable bus messaging behavior. See Viewing Time Correlated NI VeriStand Data Logs for an example of performing this analysis with NI DIAdem software.

Requirements Creation, Traceability, and Fulfillment

The Importance of Requirements Traceability

To ensure embedded software quality, and software quality in general, it has been demonstrated that you need to be able to trace requirements artifacts through all the stages of the software development process. A typical software process includes research, definition, development, test, and deployment. Tracking to establish relationships and analyze the impact of software changes is a common attribute of the software development process, especially in areas where the cost of failure is high or failure could result in the loss of life.

Software projects lacking adequate requirements traceability have been shown to contain an increased number of defects that significantly impact system safety and reliability. Making perceived small changes can have a large knock-on effect that leads to a final product not completely addressing all of the requirements as described at the project kickoff.

Regulation agencies, driven primarily by safety concerns, compounded with companies, driven primarily by the potential for costly product recalls, has resulted in a plethora of standards, best practices, and software tools for requirements management. Traceability could be mandated on your next project.

Test Automation

Modern test systems can be automated from the highest level down to the measurement instrument. This can be a complex task that spans multiple tools from different vendors and multiple OSs, some of which may need to execute on a real-time HIL system. Check with your tools provider to ensure compatibility as early as you can. Test automation can be a major factor in ensuring cost-effective requirements traceability.

In addition to executing test scripts to drive the virtual car, forward-thinking organizations take test execution and automation one step further by using a test automation framework. With these frameworks, you can run batches of tests, perform post processing and analysis on test data, and generate reports without any run-time human interaction. You simply configure the test system, and the test execution happens independently. Test automation also allows teams of engineers to communicate effectively by automatically linking product requirements and test cases to test results. This streamlines operational efficiency by removing the manual comparison of test data to requirements.

A high-level goal of an ECU test team should be to develop a library of test cases that provide sufficient test coverage. This library is the key element in building confidence in ECU quality. As the test case library is expanded, it can be automated and scheduled to run overnight or in software-change-triggered regression tests. Reports from timely regression tests can prevent a recently introduced embedded software bug from living on for weeks and becoming progressively harder to fix.

Selecting the Right HIL System for Your ECU

Openness, Extensibility, Flexibility

When selecting an HIL system, you should first consider whether you want to purchase components and integrate the system yourself or buy a turnkey system. Most turnkey system vendors do not sell components, but component vendors often deliver turnkey systems through partner channels.

Purchasing components requires you to have engineering staff on hand with domain knowledge to integrate the components, but it gives you more control over system extensibility and tailoring. Buying a turnkey system relieves the engineering burden, but you must ensure that the system will meet your current and future needs. One way to guarantee this is to purchase an “open” and “extensible” platform. An open platform supported by multiple vendors offers the best possible value and protects your investment.

The Importance of HIL Test System Flexibility

You can choose from many options to incorporate HIL simulation in your test system. With the drive to continually reduce the cost of test, a flexible solution is essential to making HIL simulation practical in your development process. An effective HIL simulation solution should rapidly adapt to changes encountered during and between development cycles. A small change in the test process or configuration should not require a major renovation of your HIL simulator. At the current rate of innovation, you cannot expect a single vendor to simultaneously meet the time-to-market, quality, and cost expectations for all the latest technologies. An open HIL simulation solution based on COTS tools ensures that you can always integrate the technologies required to test your ECU.

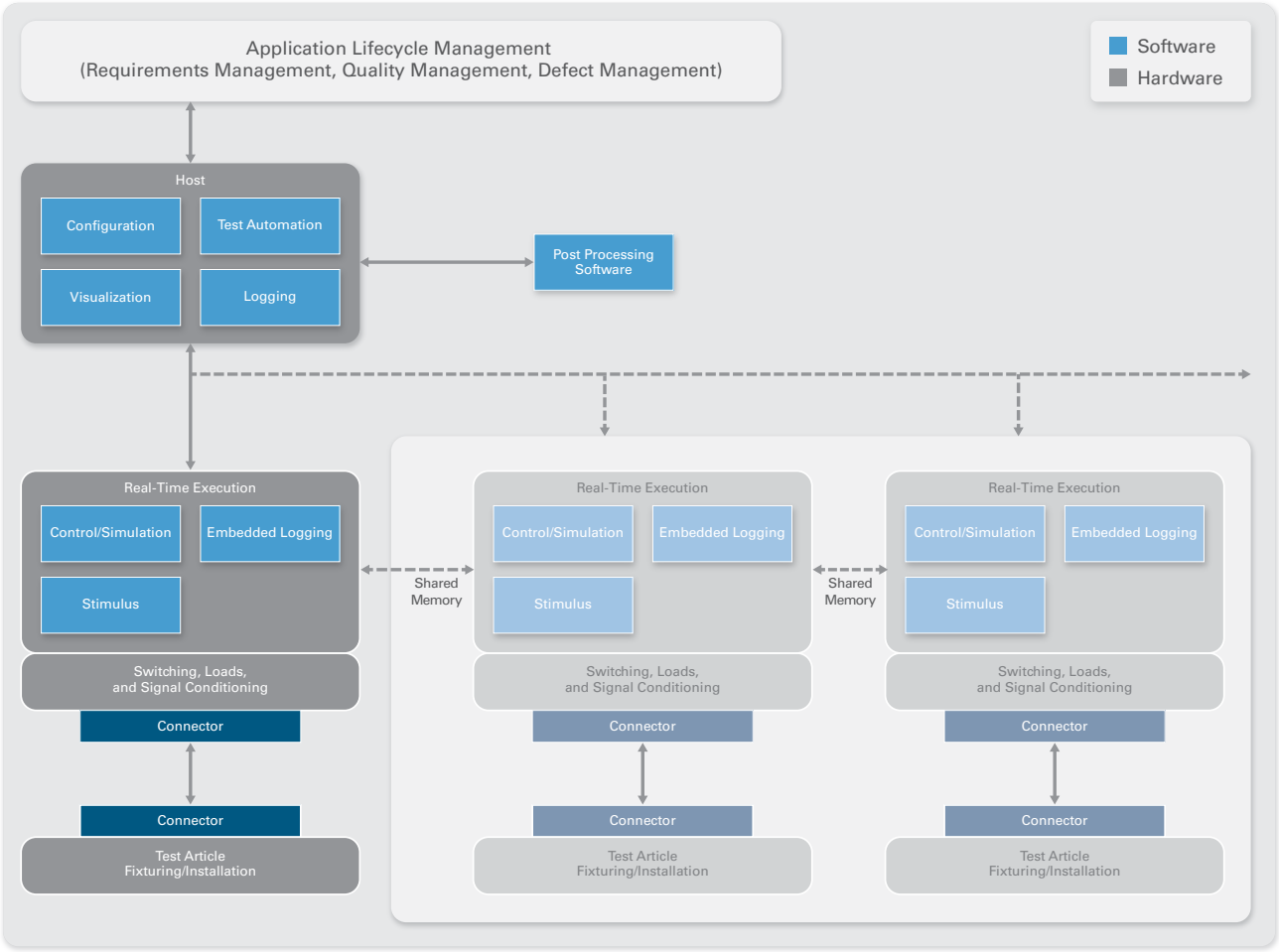

Figure 12. Flexible HIL Test Systems are Future-Proof for Requirements and Project Expansion.

Although HIL systems are becoming ubiquitous in the embedded test world, they are nevertheless another step in the test continuum. When choosing your HIL test strategy, be sure to consider how these systems will integrate into your test workflow beyond embedded software validation. Test tool companies with a more holistic view of test offer a far better value proposition than those focused on a particular area of the test cycle.

The NI HIL platform is a COTS solution designed to be extended and customized to meet your changing requirements. You can use NI tools on small desktop systems and, because of their modular architecture and open software, extend them to be used on distributed, high-channel-count systems with tight synchronization, such as Iron Bird aircraft simulators. NI designs products to meet the needs of various industries, from industrial control to consumer electronics. The performance, reliability, and flexibility needed for these demanding applications are also available to engineers performing HIL simulation, which makes NI an ideal partner for embedded software test.

Global Services and Support

Getting the most value from an HIL system extends past the initial system setup and requires not only keeping the system operational and in service but also training personnel in the tools deployed on the system. Selecting the right vendors with the right services, including technical support, system maintenance, sparing, and training, can be beneficial to maximizing your investment.

Since companies may split development across multiple continents, finding a vendor with a globally consistent support methodology and infrastructure is also important. NI has a global presence with offices in 50 countries. Support teams all over the world employ experienced engineers to help you succeed with your HIL test, which makes NI a reliable provider of tools for embedded software test.

References:

1. Horner, T., "Knock Detection Using Spectral Analysis Techniques on a Texas Instruments TMS320 DSP," SAE Technical Paper 960614, 1996, doi:10.4271/960614.

2. Viele, M., Stein, L., Gillespie, M., and Hoekstra, G., "A PC and FPGA Hybrid Approach to Hardware-in-the-Loop Simulation," SAE Technical Paper 2004-01-0904, 2004, doi:10.4271/2004-01-0904.

NATIONAL INSTRUMENTS™