ENHANCING ELEVATOR SAFETY AND COMPLIANCE THROUGH AUTOMATED HARDWARE-IN-THE-LOOP (HIL) TESTING

Gorripati Maneesh Kiran, Chhibber Bhaskar Narayan
CRD, Schindler India Pvt. Ltd.

KEYWORDS: Hardware-in-the-Loop (HIL), Elevator Control Systems, Real-Time Simulation, Test Automation, Embedded System Validation, Safety-Critical Testing, Device under test (DUT), Input/output (I/O)

ABSTRACT

Hardware-in-the-Loop (HIL) testing is a vital methodology for the development and validation of elevator control systems, ensuring safety, reliability, and regulatory compliance before deployment [9], [14]. By integrating real hardware components, such as controllers and sensors, into a simulated environment, HIL testing allows engineers to assess system behavior under various operating conditions without the risks associated with full-scale physical testing [13].

The advantages of HIL testing in elevator systems include early fault detection, reduced development costs, and enhanced safety [16], [5]. It enables the testing of emergency scenarios such as power failures, brake malfunctions, and overspeed conditions in a controlled environment, minimizing risks to human safety [7].

1. INTRODUCTION

HIL testing plays a crucial role in ensuring compliance with industry standards and safety codes by allowing manufacturers to validate system responses to mandatory safety requirements before real-world implementation [12]. Additionally, automated HIL testing enhances full-scale software release testing by enabling continuous and repeatable validation of software updates, minimizing human error, and accelerating development cycles [4]. Automated test execution allows for comprehensive regression testing, ensuring that new software versions do not introduce faults or compromise system safety [2]. By integrating HIL into the software release process, manufacturers can achieve higher reliability, faster certification, and smoother deployment of elevator control systems while maintaining compliance with evolving safety standards [3].

Hardware-in-the-Loop (HIL) simulation has become a cornerstone of embedded system validation in automotive and aerospace industries, enabling real-time testing of control algorithms against simulated plant models [6]. However, its application in the vertical transportation domain-specifically elevator systems-remains limited despite the growing complexity of safety-critical control architectures. By adopting automated HIL testing in the testing life cycle of an elevator control software, several challenges can be addressed in an elevator system which are inherently complex and safety critical [1].

This paper presents a structured approach to employ HIL testing for elevator control systems enabling validation of control logic without the need for a complete physical prototype of the elevator. We describe an integrated automated HIL system with automated test scripts, validation cycles, runtime signal monitoring, and closed-loop validation. This paper also demonstrates how adopting automated HIL testing of elevator controllers addresses major concerns associated with physical test systems which include test coverage, efficiency, safety and scalability.

2. IMPLEMENTATION OF AUTOMATED HARDWARE-IN-THE-LOOP TESTING FRAMEWORK FOR ELEVATOR CONTROL SYSTEMS

2.1 Overview of a HIL System

Hardware-in-the-Loop (HIL) testing is a sophisticated technique employed in the rigorous testing of complex real-time embedded systems [9]. HIL test system involves creating a controlled closed loop simulated environment around the real control hardware which is the Device Under Test (DUT). The simulated environment, called the plant model, precisely simulates the physical environment expected by the controller thereby enabling the controller to interact with a mathematical model of the physical environment as if it was interacting with the actual physical system [11].

Typically, a Hardware in the loop test system consists of several specialized and interconnected components. These components interact with the embedded control unit, that is the device under test to send and receive the required signals thus, forming a closed loop system. These connections between the real controller and the simulated plant are typically established through real analog and digital input/output (also termed as ADIO) and communication interfaces. This enables comprehensive software validation across the development lifecycle of the system [13].

A plant model is a mathematical representation of the actual system that replicates the dynamic behavior of the system that the embedded controller is intended to interact with. These high-fidelity models encompass both mechanical and electrical behavior of the elevator system [15]. These plant models run on real-time processing systems that can execute the plant behavior in response to the controller outputs in real time. A hardware in the loop system is formed by closing the loop between the controller and the plant in a controlled and automated test environment.

A hardware in the loop system can be designed with a single device under test or a group of controllers as devices under test. The HIL architecture in automobile industry generally tests a single controller or a set of controllers performing a common function, and the remaining nodes are simulated [17]. However, elevator systems use multiple controllers with different communication interfaces within the same elevator. Hence, the presented HIL architecture for elevator contains multiple real controllers interfaced with the plant model which resembles a lab car in automotive industry. This architecture ensures comprehensive and extensive testing of software and its interaction with different controllers at the elevator system level. This approach also facilitates testing of interactions between all the controllers at once for a given software release often with actual wiring harness and real load devices.

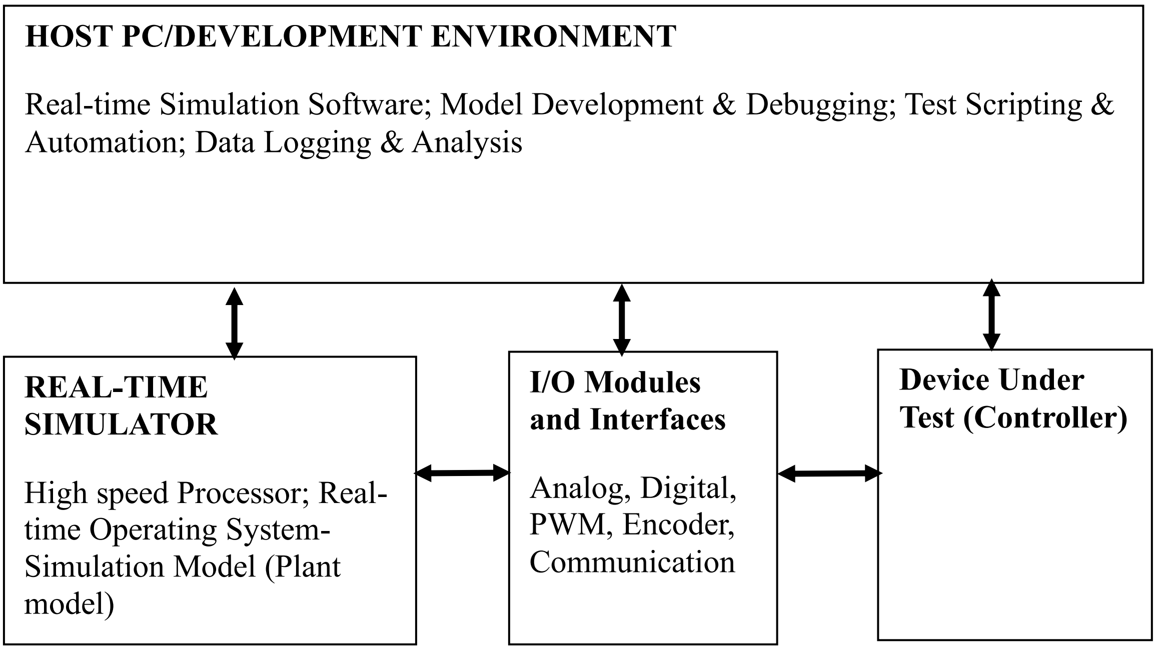
HOST PC/DEVELOPMENT ENVIRONMENT
Real-time Simulation Software; Model Development & Debugging; Test Scripting & Automation; Data Logging & Analysis
REAL-TIME SIMULATOR
High speed Processor; Real-time Operating System-Simulation Model (Plant model)
I/O Modules and Interfaces
Analog, Digital, PWM, Encoder, Communication
Device Under Test (Controller)

Figure 1: Block diagram of a Hardware-in-the-Loop (HIL) system architecture for control system testing

HIL testing empowers the tester with accurate tracking and control of signals between the controller and the simulated environment to ensure reliable closed loop testing of a controller in real-time. This involves actively managing input/output signal value, timing, synchronization and feedback in real time [3]. Graphical User Interface (GUI) systems can be developed and used to track and control these signals. Live tracking of these signals helps in test automation and analyzing the failures. These interfaces act as a powerful tool to visualize what happens underneath the HIL system.

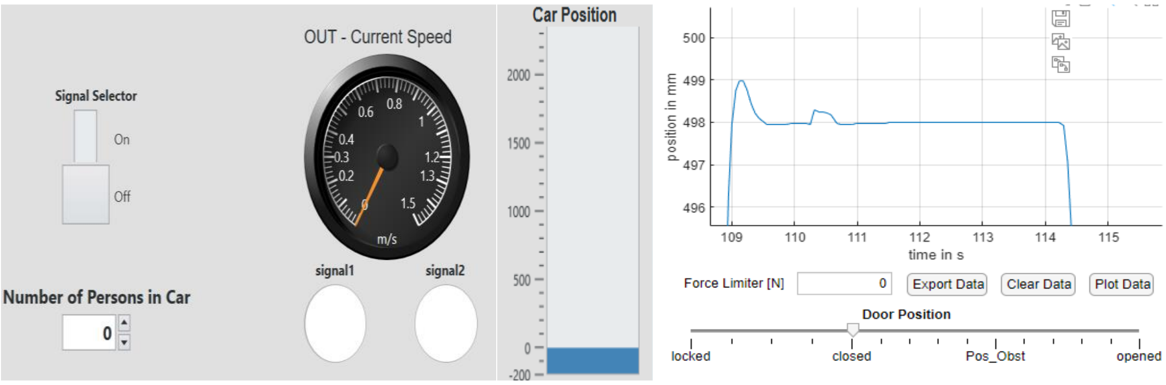

Figure 2: Illustration of a Graphical user interface for real time Hardware in the loop (HIL) testing

2.2 Automation Architecture

Automation of a HIL system adds great potential to its testing capability. With increased complexity of elevator electronic systems, manual testing of software can be error prone and inefficient [4], [6]. An automated HIL system enables execution of large test campaigns overnight and without any manual intervention. Automated test cycles are repeatable and consistent.

Automation in a HIL involves signal manipulation, real time monitoring of signals and logs, result evaluation, reporting and failure analysis. An automation script can be developed using multiple tools depending on the hardware used in the HIL system [1]. This paper discusses a generic automation framework which can be used to automate any HIL irrespective of the make of the hardware and software used on the HIL.

Dynamic Link Libraries (DLL)s, Extensible Markup Language (XML) and batch files serve as key enablers in automation processes. They can be used to automate a test setup. Dynamic-link libraries are powerful sources which can interface with different hardware and software used in HIL. A DLL can be designed to be used to interact with the automation hardware, plant model, data acquisition hardware, controller and other tools used on a HIL. An XML file is often used to define a test case, test parameters, execution sequence, expected inputs/outputs and fault injection scenarios. A batch file can be used for test orchestration and automation. A batch file can be designed to launch HIL environments, load configurations, trigger automated test scripts and start/stop logging [2].

A general automation framework using DLLs, XMLs and Batch files would involve the following steps:

- Define the test configuration in an XML for example testconfig.xml
- Configure the test environment using DLLs for example testenv.dll
- Create a batch file to orchestrate the test for example testcampaign.bat
- Run the batch file to start a test campaign
- Monitor, log and generate test reports for in .xlsx format.

This automation framework is not only modular, flexible and scalable but would also works with different test setups including custom setups [3].

3. BENEFITS OF AUTOMATED HIL TESTING

3.1 Cost-Effectiveness of HIL

Hardware-in-the-Loop (HIL) simulation offers substantial cost-saving benefits throughout the embedded system development lifecycle, particularly in the elevator industry where product complexity and safety standards continue to rise [10].

- Reduces Physical Prototypes: Simulates the plant and environment, minimizing the need for costly mechanical prototypes during early development [7].
- Early Defect Detection: Identifies software and integration issues before system-level testing, avoiding expensive redesigns or late-stage fixes [8].
- Automated Test Execution: Supports unattended overnight runs, reducing manual labor and increasing test throughput without proportional cost [1].

• Reusable Test Assets: XML-based test cases (testconfig.xml), environments (testenv.dll), and scripts can be reused across controller variants and platforms, saving development time.
• CI/CD Compatibility: Easily integrates with DevOps pipelines, supporting continuous testing and early validation with minimal additional cost.
• Lab Utilization Efficiency: Maximizes test bench usage, reduces downtime, and minimizes the need for expensive physical testing.

3.2 Enhanced Safety and Risk Mitigation

HIL testing is especially valuable when it comes to developing and validating safety-critical systems in a controlled and risk-free environment [12]. It allows teams to simulate faults and failure conditions that would be unsafe or impractical to test on real systems, thereby strengthening safety assurance [7].

• Isolated Testing of ECUs: ECUs can be tested independently without connecting to live mechanical components, reducing the risk of unintended motion or system damage [8].
• Safe Fault Injection: Conditions like short circuits, signal noise, and sensor dropouts can be simulated without physical consequences, allowing repeatable validation of failure responses [3]
• Supports Safety Standards: HIL systems are well-suited for verifying safety mechanisms and fail-safe behavior required by safety frameworks [14].
• Reduces Human Error: Automation minimizes manual intervention during test runs, helping avoid setup mistakes that could compromise safety or damage equipment [6].

3.3 Reduced Time to Market

Time-to-market is critical in product development, and HIL contributes significantly by accelerating testing cycles. It enables software validation to begin earlier and progress faster, even when physical hardware or prototypes are not yet available [5], [10].

• Parallel Development and Testing: Software teams can start validation well before the final hardware is ready, reducing bottlenecks in the development process [1].
• Faster Debugging and Iteration: Test failures can be analyzed, corrected, and re-tested quickly, without waiting for physical vehicles or test tracks [4].
• Virtual Integration Testing: Communication between ECUs and other systems can be tested in a virtual setup, revealing integration issues early [2].
• Speeds Up Certification and OEM Review: Automated HIL workflows streamline compliance testing, helping meet regulatory and customer timelines more efficiently [1].

3.4 Improved Test Coverage and Quality

Beyond just saving time and cost, HIL improves the coverage and reliability of testing. It enables comprehensive validation of embedded systems under a wide range of operating conditions that may be difficult or impossible to reproduce physically [7].

• Expanded Edge-Case Testing: Rare or extreme conditions, such as low battery voltage or rapid sensor fluctuation, can be reliably simulated and tested [3].

• Scalable Regression Testing: With automation, large sets of test cases can be re-run after each software change to catch unexpected side effects early [6].
• Detailed Signal Monitoring: Real-time logging of ECU behavior, sensor responses, and internal state variables helps pinpoint subtle issues [1].
• Closed-Loop System Validation: Multiple subsystems-mechanical, electrical, and software can be tested together, improving cross-domain reliability [9].

To understand the practical benefits of automated HIL, consider the scenario of validating an elevator system's response to door obstruction events. In real-world environments, testing how the elevator reacts when its doors are blocked can be challenging and potentially unsafe. Physical testing typically involves placing objects in the doorway, which risks damage to the door mechanisms and introduces variability—especially when simulating edge cases like partial obstructions, rapid re-blocking, or sensor misreads.

Using automated HIL testing, the door sensors and motor control logic can be modelled within a virtual plant, and fault injection scripts emulate obstruction conditions with high precision. These can include scenarios such as a child's hand partially blocking the door, intermittent obstructions, and delayed sensor responses. The control software's behaviour—reversing door motion, issuing alerts, or retrying closure—is monitored and evaluated against expected safety protocols.

During development, this approach offers significant advantages. Engineers can simulate fault conditions early in the software lifecycle, long before physical prototypes are available. Automated HIL testing enables rapid iteration, supports regression testing across software versions, and allows validation of time-sensitive logic under repeatable conditions. Furthermore, the framework can be integrated into continuous integration pipelines, ensuring that any software update is automatically verified against critical safety scenarios. This not only accelerates development but also improves reliability and confidence in the system's ability to handle real-world fault conditions.

The ability to simulate and validate complex fault conditions in a repeatable and safe manner makes automated HIL testing a powerful tool for developing robust elevator control systems. It allows developers to explore edge cases, refine control logic, and ensure consistent system behavior under conditions that are impractical or hazardous to reproduce physically.

This case highlights how automated HIL enables safe and repeatable testing of failure modes that are impractical or hazardous to perform physically.

4. EMERGING DIRECTIONS IN AUTOMATED HIL FOR ELEVATORS

Looking ahead, automated HIL testing in elevator systems can evolve further by integrating emerging technologies:
• Digital Twin Integration: Coupling HIL platforms with digital twin models enables predictive testing and lifecycle monitoring of elevator systems [4].
• Artificial intelligence/Machine learning-based Test Case Generation: Intelligent algorithms can automatically generate test scenarios based on historical fault data, improving coverage of rare or unforeseen events [1].

• Cloud-Based HIL Systems: Distributed test benches hosted on cloud infrastructure can support remote execution, collaboration, and scalability.
• Cybersecurity Testing: With elevators increasingly connected to building automation systems, HIL can be extended to test their resilience against cyberattacks and unauthorized control.

Such advancements position automated HIL not only as a validation tool but as a central pillar for future elevator innovation and safety assurance.

5. CONCLUSIONS

This paper has presented a comprehensive overview of Hardware-in-the-Loop (HIL) testing applied to elevator control systems, emphasizing the need for safe, efficient, and automated testing in this safety-critical domain. By integrating actual control hardware with real-time simulated plant models, HIL enables early validation, safe fault injection, and extensive test coverage without requiring full-scale physical prototypes.

A generic, scalable automation framework using DLLs, XML configurations, and batch orchestration was also described to demonstrate how such systems can be adapted for the unique requirements of elevator systems. The proposed approach facilitates continuous testing, improves software quality, and accelerates compliance with evolving safety standards.

Ultimately, HIL testing stands to transform how elevator systems are developed and validated- reducing costs, improving safety, and shortening time to market. As elevator electronics continue to grow in complexity, adopting a robust, automated HIL strategy will be essential for ensuring system reliability and regulatory compliance in future designs.

***Disclaimer**: This paper presentation is by the authors and is intended for educational purposes without constituting any professional judgment. Statements of facts and opinions provided are those of the participants individually and do not purport to reflect the opinion or statements by the Company. The Company neither approves nor assumes responsibility for the content, accuracy, or completeness of the information provided in the paper presentation.*

6. REFERENCES

[1] “Automating Embedded Software Testing.” (2024). Runtime Rec..
[2]“Benefits of CI/CD with HIL.” (2025). Codewave Insights.
[3] “Challenges in Manual Testing of Elevators.” (2021). Elevator Tech Review.
[4] “Digital Twin Modelling for Elevators.” (2023). Int. Conf. Building Automation.
[5] “Hardware-in-the-Loop Simulation in Industrial Automation.” (2019). IEEE Ind. Electronics Magazine.
[6] Hogan, D., et al. (2019). Testing electro-mechanical actuators via HIL. J. Eng.
[7] Hwang, T., et al. (2006). Development of HIL Systems for Active Brake Control. SICE–ICASE.
[8] Isermann, R. (2008). Mechatronic systems—Innovative products with embedded control. Control Eng. Pract., 16(1).
[9] ISO 22201-1:2017. (2017). Lifts-Programmable electronic systems in safety-related applications.

[10] Kassakian, J. G., et al. (1991). Principles of Power Electronics. Addison-Wesley.
[11] Ljung, L. (1999). System Identification: Theory for the User. Prentice Hall.
[12] "Mastering HIL Testing in Embedded Systems." (2025). Number Analytics.
[13] Singh, A., et al. (2019). Automation in HIL for critical systems. IEEE Trans. Ind. Informatics, 15(4).
[14] T. Johnson et al. (2020). Application of Hardware-in-the-Loop simulation for elevator control systems. Proc. IEEE Embedded Systems.
[15] Törngren, M. (2007). Model-based development. In Handbook of Real-Time Embedded Systems. CRC Press.
[16] Hancke, G. (2021). Embedded Systems in Vertical Transport. J. Vertical Transport, 6.
[17] SAE J2638, "Guidelines for HIL Test Systems in Automotive," SAE Int., 2018.

BIOGRAPHICAL DETAILS

Gorripati Maneesh Kiran received his M.Tech in power electronics and drives from NIT Jamshedpur in 2020. He is currently working as an Engineer at Schindler India Pvt. limited, where his work focuses on Hardware in loop system development, automation and testing. He has over 5 years of experience in the current field.

Chhibber Bhaskar Narayan has a master's degree in electrical engineering (control systems) from COE, Pune. He has over 24 years of experience in fields of model-based systems engineering and electronic controls. Presently, he is working at Schindler India R&D, as engineering lead for HIL development.