



SIL of Instrumental Cluster by Using Can Rest Bus Simulation for In-Vehicle Network (IVN) Testing and Validation

Dr. B. Vijaya Krishna¹, Mr. G. Sai Goutham², B. Venkata Lakshmi³, G. Vamsi Rakesh⁴,
G. G Pavan Krishna⁵, M. Lakshmi Chaitanya⁶

^{1,2}Asst. Professor, Electrical & Electronics Engineering, Bapatla Engineering College, Bapatla, Ap

^{3,4,5,6}Electrical & Electronics Engineering, Bapatla Engineering College, Bapatla, Ap

E-mail: ¹boyina.vijayakrishna@becbapatla.ac.in; ²saigotham.golive@becbapatla.ac.in;

³bandyvenkatalakshmi@gmail.com; ⁴vamsirakesh0@gmail.com; ⁵pavankrishnagangumalla@gmail.com;

⁶chaitumolabanti@gmail.com

ABSTRACT

In modern vehicle systems, Electronic Control Units such as the Body Control Module (BCM) and Instrument Cluster (IC) are essential to vehicle functionality and interaction between the vehicle and driver. This report details a S/E Loop (SIL) simulation environment designed to validate the functional behaviour of such devices under vehicle operating conditions. Using TS Master as the simulation platform, a fully virtualized test environment was created without actually installing hardware devices such as the Arduino Mega 2560. Peripheral behaviour such as door status, light states, speed of the vehicle, and fuel level were simulated in software in the virtual environment. The interaction between the IC and BCM was done over a virtual CAN bus network in TS Master. The BCM logic was emulated to transmit CAN signals to the emulated IC, enabling vehicle parameters to be displayed in real-time on a virtual dashboard. Turn indication, door ajar indication, low fuel indication, and over-speed indication were the functional tests simulated and validated in the SIL environment. This paper demonstrates how Software-in-the- Loop testing enhances functionality validation effectiveness by allowing early-stage software verification, reduces development and prototype expenses, and facilitates the early identification of faults in safety-critical automotive subsystems.

Keywords: SIL Testing, BCM, Instrument Cluster, CAN Protocol, TS Master, Rest Bus Simulation, In-Vehicle Networking, Functional Validation.

INTRODUCTION

Growing complexity of automotive systems, fueled by the integration of advanced driver assistance systems (ADAS) and digital interfaces, requires strong and disciplined software verification and validation activities. Software-in-the-Loop (SIL) simulation was one of the key methodologies within the automotive development process, which allowed embedded software functions to be validated in simulated environments before the hardware was available. SIL not just facilitates early diagnosis of software flaws but also aids iterative development and safety standard adherence like ISO 26262. Within the vehicle's architecture, the instrument cluster is among the most

critical human-machine interfaces, tasked with presenting key operating information to the driver, such as vehicle speed, engine data, warning messages, and system status displays. Due to its safety-critical nature and user-oriented purpose, the software of the instrument cluster must be extensively validated under a broad set of operating conditions

In this paper, an SIL-based verification technique for the instrument cluster augmented with Rest bus simulation is put forward to emulate the missing Electronic Control Units' (ECUs) communication pattern. Rest bus simulation offers simulation of network traffic and message transfer such that software for the instrument cluster can be rigorously verified against dynamic and realistic communication patterns without a physical system being absolutely necessary. The combination of SIL and Rest bus simulation not only speeds up the development process

but also enhances test scenario coverage such as fault injection and edge case management. The approach suggested in this paper is to improve the quality and reliability of instrument cluster software through a combination of communication-level validation and model-driven simulation. Experimental evidence and case studies are provided to show the proposed approach in delivering high interoperability, functional robustness, and safety assurance levels

LITERATURE REVIEW

The growing reliance on electronics in modern vehicles has led to the development of more advanced and complex electronic control systems, such as the Body Control Module (BCM) and Instrument Cluster (IC). As automotive systems become increasingly integrated, there is a need for more sophisticated testing methods to ensure the reliability, safety, and efficiency of these systems. Traditional testing approaches, relying on physical prototypes and hardware, are time-consuming and expensive. As a result, Hardware-in-the- Loop (HIL) simulation has emerged as a pivotal methodology for validating vehicle control systems, including BCM and IC. Key findings include:

BODY CONTROL MODULE AND INSTRUMENTAL CLUSTER

The Body Control Module (BCM) is a central electronic control unit (ECU) that operates the majority of the vehicle's comfort, convenience, and safety features. A few of them include power window function, central locking function, lighting functions, windshield wipers, and other accessory systems. The BCM has a key role in balancing these features in order to ensure operational efficiency, driver comfort, and passenger safety. Verification of the BCM software is critical as its ultimate function to ensure system reliability and automotive sector compliance.

The Instrument Cluster (IC) or car dashboard is the primary man-machine interface showing real-time information to the driver. The IC is employed to show vital information such as vehicle speed, engine RPM, fuel level, engine heat, and warning lights. IC depends on the correct interpretation and visualization of car condition based on timely and accurate data available from ECUs like the BCM. With vehicle systems becoming increasingly complex and software-driven, there is an urgent need for strict validation of the behaviour of IC and its interaction with other vehicle sub-systems.

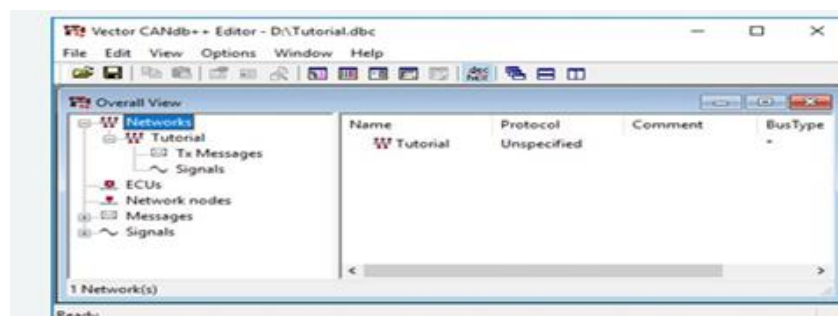
Software-in-the-Loop (SIL) Simulation for Automotive Testing

Software-in-the-Loop (SIL) simulation is among the initial main validation techniques of automobile embedded systems. While compared to physical hardware components being used in Hardware-in-the-Loop (HIL) simulation, SIL testing only uses a virtualized one. Software that will be tested is put in the middle of virtualized model duplicates of encompassing systems within SIL, and that allows strong validation of the function, behaviour of communication, and system interactions with no hardware in place.

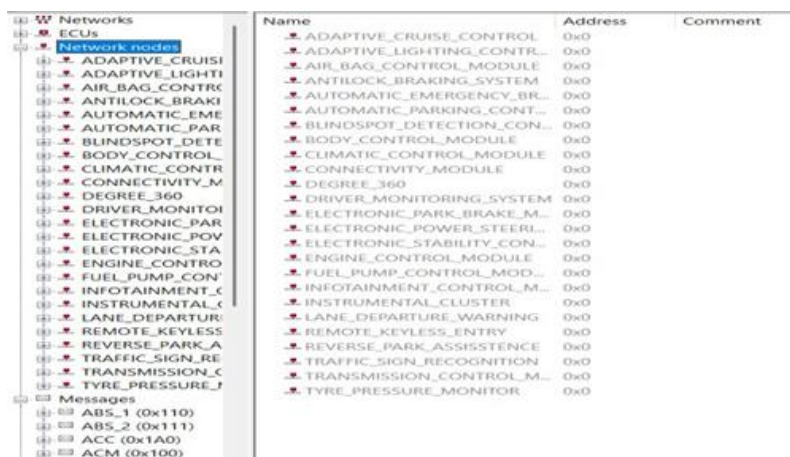
For IC and BC, SIL simulation is a highly useful methodology for the validation of embedded software behaviour in aspects of real and dynamic operating modes. With the use of Rest bus simulation, non-existent ECUs are emulated virtually and communication networks such as Controller Area Network (CAN) and Local Interconnect Network (LIN) can be checked without installing whole systems. This approach supports the detection of software faults at an early stage, helps regression testing of software releases, and reduces development cost by eliminating hardware dependency. With the use of SIL using Rest bus simulation, engineers can validate core vehicle functions including warning light turns on, real-time gauge updates, and handling of diagnostic events, and assure that both BCM and IC are following rigorous performance and safety expectations. In addition, SIL can perform fault injection testing, robustness testing, and scenario testing, all which contribute to higher overall quality and reliability of motor vehicle electronic systems

CAN Database Development using CANdb++

A Controller Area Network (CAN) database, simply referred to as a DBC file, is a foundation for automotive communication systems in the form of a standardized representation and interpretation of CAN messages exchanged between Electronic Control Units (ECUs). In the Instrument Cluster (IC) Software-in-the-Loop (SIL) simulation, a comprehensive CAN database was established using Vector's CANdb++ tool to define and manage the communications infrastructure required for system validation.



The designed CAN database contains 25 ECU nodes, 37 messages (frames), and 227 signals, which form a comprehensive set of vehicle parameters required for the operation of the Instrument Cluster. Each message was assigned a certain arbitration ID, standardized data length codes (DLCs), and designated as cyclic or event-driven based on the functional needs. Signals within the frames were set up with proper bit positions, lengths, scaling factors, offsets, units, and value ranges carefully so that vehicle data could be properly encoded and decoded.



| Name | Address | Comment |
|------------------------------|---------|---------|
| ADAPTIVE_CRUISE_CONTROL | 0x0 | |
| ADAPTIVE_LIGHTING_CONTROL | 0x0 | |
| AIR_BAG_CONTROL_MODULE | 0x0 | |
| ANTILOCK_BRAKING_SYSTEM | 0x0 | |
| AUTOMATIC_EMERGENCY_BRAKE | 0x0 | |
| AUTOMATIC_PARKING_CONTROL | 0x0 | |
| BLINDSPOT_DETECTION_CONTROL | 0x0 | |
| BODY_CONTROL_MODULE | 0x0 | |
| CLIMATIC_CONTROL_MODULE | 0x0 | |
| CONNECTIVITY_MODULE | 0x0 | |
| DEGREE_360 | 0x0 | |
| DRIVER_MONITORING_SYSTEM | 0x0 | |
| ELECTRONIC_PARK_BRAKE_MODULE | 0x0 | |
| ELECTRONIC_POWER_STEERING | 0x0 | |
| ELECTRONIC_STABILITY_CONTROL | 0x0 | |
| ENGINE_CONTROL_MODULE | 0x0 | |
| FUEL_PUMP_CONTROL_MODULE | 0x0 | |
| INFOTAINMENT_CONTROL_MODULE | 0x0 | |
| INSTRUMENTAL_CLUSTER | 0x0 | |
| LANE_DEPARTURE_WARNING | 0x0 | |
| REMOTE_KEYLESS_ENTRY | 0x0 | |
| REVERSE_PARK_ASSISTANCE | 0x0 | |
| TRAFFIC_SIGN_RECOGNITION | 0x0 | |
| TRANSMISSION_CONTROL_MODULE | 0x0 | |
| TYRE_PRESSURE_MONITOR | 0x0 | |

Special care was taken for:

Bitfield and Signal Mapping: to maintain non-overlapping signal definitions within the frames to ensure data integrity.

Scaling and Physical Units: Definition of correct engineering units (e.g., velocity in km/h, temperature in °C) and application of scaling factors to facilitate correct physical value interpretation throughout simulation.

Transmission Properties: Specification of cyclic and spontaneous (event-driven) transmission behaviours according to signal criticality. **Validation and Consistency Checks:** Facilitating automatic checks on signal range validity, avoidance of bitfield overlap, and attribute consistency for every message and ECUs.

This standardized CAN database was the communication spine of both SIL environment and Rest bus simulation configuration. It enabled correct simulation of network traffic, correct signal monitoring, and made sure that the Instrument Cluster received and interpreted data in the same way as it would in an actual vehicle network. By aligning the SIL test framework to the DBC created, simulation environment fidelity was significantly enhanced, allowing for thorough validation of cluster

TS Master for SIL and Rest bus Simulation

TS Master is a comprehensive CAN network analysis, simulation, and validation tool commonly used in automotive embedded system development. It offers an integrated platform for monitoring, transmitting, and simulating Controller Area Network (CAN) messages and is a suitable platform for conducting Software-in-the-Loop (SIL) testing and Rest bus simulation operations.

For Instrument Cluster (IC) validation, TS Master is most

important because it allows emulation of vehicle communication network by loading and interpreting DBC files. Leverage the proven CAN database, TS Master provides the capability to emulate exactly Electronic Control Units (ECUs) that are inaccessible physically. This Rest bus simulation capability allows the Instrument Cluster to receive all the functional validation-needed CAN signals without the need for a full vehicle or hardware installation.

TS Master's main features are:

Rest bus Simulation: Simulation of missing ECUs by transmission of pre-defined CAN messages based on real-time cyclic or event-driven behaviour, according to DBC specifications.

Signal Manipulation and Monitoring: Real-time display of signal values, fault injection support, dynamically changing signal values, or failure simulation scenarios.

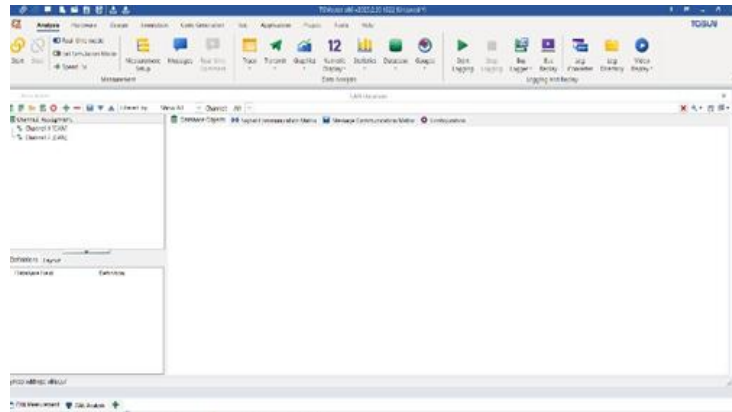
Scripting and Automation: CAPL-like scripting and rule-based automation facilities for defining complex test scenarios, automated validation routines, and fault injection campaigns.

Dashboard Generation: Graphical user-specified dashboards for real-time observation of key signals like speed, RPM, fuel level, and warning signals, making easy to track while conducting SIL.

Logging and Analysis: CAN traffic recording for off-line analysis during simulation, for debugging, and verifying system activity against specified results.

Using TS Master's Rest bus simulation and dashboard capabilities, the SIL organization of the Instrument Cluster is realized with high fidelity and allows validation of normal modes, diagnostic paths, and failure handling modes. Ease of use and adherence to automotive communication protocols make the tool suitable for early-validation

purposes as well, further allowing the development of robust, dependable, and compliant embedded software before hardware integration.

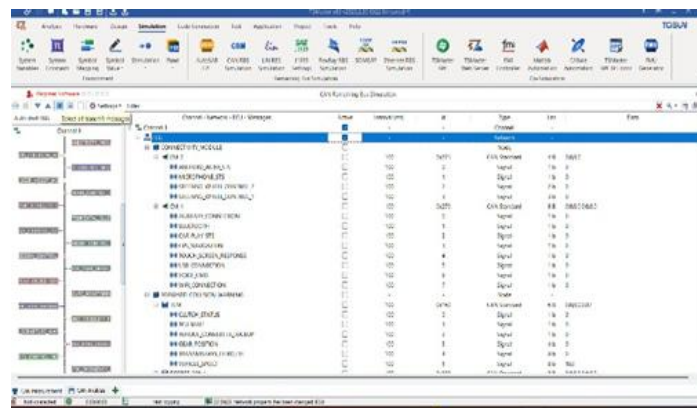


Rest bus Simulation (RBS) in Software-in-the-Loop (SIL) Testing

Rest bus Simulation (RBS) is one of the most important parts of the Software-in-the-Loop (SIL) setup, especially for the verification of complex automotive systems like the Instrument Cluster (IC). In modern automobile architectures, numerous Electronic Control Units (ECUs) exchange information with each other via networks like the Controller Area Network (CAN) for critical data sharing. However, not all ECUs might be accessible for physical testing at the initial stages of development. Rest bus Simulation addresses this issue by emulating the behaviour of absent ECUs, allowing complete system verification without a complete hardware setup.

In the SIL environment in the Instrument Cluster:

Virtual ECU Simulation: RBS replicates the behaviour of virtual ECUs by creating and sending pre-programmed CAN messages according to the information contained in the CAN database (DBC file). This enables the Instrument Cluster to receive required input signals (e.g., vehicle speed, engine speed, warning conditions) as if the car was part of a full integration network.



Communication Network Simulation: Through software such as TS Master, Rest bus Simulation provides a simulated communication network where the Instrument Cluster communicates with real as well as simulated ECUs. Such an environment facilitates real-world testing of the IC's software logic, display refreshes, and diagnostic behaviour.

Scenario-Based Validation: Different operating scenarios— i.e., normal driving, fault conditions, and edge cases—are simulated by controlling the interaction of CAN messages among the virtual ECUs. This allows end-to-end validation of responses of the Instrument Cluster to dynamic signal changes and system events.

Fault Injection and Robustness Testing: RBS allows faulty conditions like loss of signal, corrupt data, or communication failures to be injected. These conditions are used to test the Instrument Cluster's ability to handle faults and the system's robustness.

Regression and Iterative Testing: During software iterations, RBS allows test cases to be executed repeatedly and automatically so that software modifications will not negatively impact previously tested functions.

In the context of an interface with Rest bus Simulation in combination with SIL environment, the Instrument Cluster is testable completely without end-to-end hardware equipment, reducing the development time to a

minimum level while maintaining the functional integrity intact. The interaction of a rigidly formalized CAN database created with CANdb++ and RBS feature of TS Master results in an automaker's-level, flexible, and controlled simulation environment for the testing process so that early Software error detection occurs along with enforcement of automobile safety standards compliance.

Panel for SIL Verification

In Automotive system Software-in-the-Loop (SIL) testing, especially for the Instrument Cluster (IC), graphical panel has a gigantic contribution in system visualization, real-time observation, and communication with the user when simulating. The panel is a simulated interface that mimics the critical vehicle display. It allows the engineers to comprehend the system reactions naturally without requiring real cluster hardware to conduct verification.

In TS Master, there was a graphical panel implemented specifically to provide the SIL and Rest bus simulation of the Instrument Cluster. The panel includes dynamic graphical objects such as speedometer, tachometer, fuel gauge, temperature gauge, warning lights, and message indicators that were all connected to respective CAN signals defined within the CAN database (DBC file).



Important characteristics of the panel are:

Real-Time Display of Signals: Real-time updation of essential parameters such as vehicle speed, engine RPM, and fuel level based on received CAN messages of the Rest bus simulation.

Warning and Indicator LEDs: ON/OFF control of warning indicators such as engine warning, ABS, battery warning based on simulated diagnostic signals and fault conditions.

Interactive Controls: Mimicking user activity (e.g., ignition ON/OFF, indicator switch operation, or gear changes) using interactive sliders and buttons to enable dynamic system state control.

Scenario Representation: Simulating real-world driving scenarios like starting the car, acceleration, and braking, along with fault scenarios, to aid end-to-end validation procedures.

Fault Visualization: Real-time visual verification of fault injection occurrences so that the ability of the Instrument Cluster to detect and show system faults can be confirmed.

The graphical panel is an important intermediary between the virtual simulation space and the test, providing an easy and useful means of observing system behaviour, detecting anomalies, and verifying function and safety criteria. Through joining the panel and the SIL testing environment, it is possible to test the software of the Instrument Cluster both on the level of communication signals as well as the human-machine interface (HMI) level, providing a better and more real system performance test.

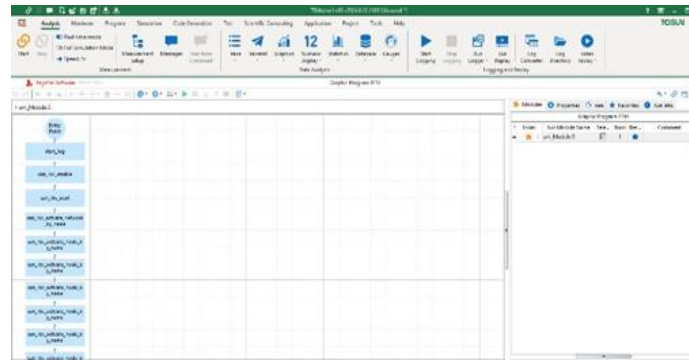
Graphical Programming in TS Master

Graphical programming in TS Master provides a graphical and visual method of creating advanced testing logic, signal processing, and automatic simulation sequences without writing much manual coding. Instead of typing

conventional scripts line by line, the users can create logical flows, test cases, and system behaviour by connecting functional blocks in a graphical environment.

In Software-in-the-Loop (SIL) testing of the Instrument Cluster (IC), graphical programming is utilized to:

Develop and Automate Test Sequences: Logical test sequences for ignition cycles, speed ranges, fault injections, and warning indicator verifications are developed graphically from coupled blocks.



Model System Actions: Actions such as ramping a vehicle speed signal, switching warning lamps, or injecting diagnostic trouble codes (DTCs) are graphically model according to real-world driving and fault conditions.

Use Event-Driven Logic: Graphical conditions (e.g., "if vehicle speed > 100 km/h, send overspeed warning") and event triggers are simpler to implement and debug for complex situations.

Construct Reusable Components: Common testing patterns or sets of logic can be refactored into reusable modules and reused across multiple test cases, making it more efficient and repeatable.

Signal Input and Output Blocks: For viewing, altering, or emulating CAN signals in real time.

Logical Operators: To execute e.g., AND, OR, NOT, comparators, and arithmetic operations to represent decision-making.

Timers and Counters: For measuring events or tracking conditions.

State Machines: To represent system behaviour across several operating states (e.g., off ignition, on ignition, driving, fault mode).

Visualization Blocks: To update graphical panels dynamically based on the test logic.

The use of graphical programming in TS Master greatly enables the development of intricate SIL test cases. It enhances readability, reduces errors compared to hand-written coding, and enables faster adaptation and test procedure expansion. During the Instrument Cluster SIL validation, graphical programming enabled systematic testing of dynamic behaviour, fault tolerance, and real-time signal response in an organized and effective manner.

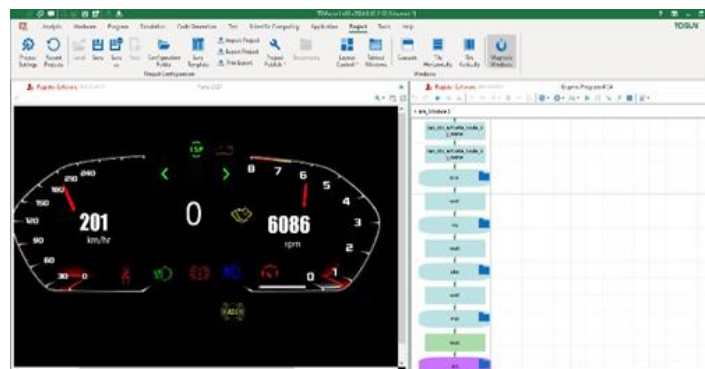
Results and analysis

The Software-in-the-Loop (SIL) test environment established for the Instrument Cluster (IC) proved the success of Rest bus Simulation (RBS) and graphical programming with TS Master. With the use of the generated CAN database in CANDb++, the entire network communication behaviour was simulated and the Instrument Cluster software was tested under realistic operating conditions without physical ECUs.

Correct Signal Simulation: The Rest bus simulation, with 37

frames and 227 signals specified in the DBC file, supplied correct and reliable data flow to the Instrument Cluster. Signals like vehicle speed, engine RPM, fuel level, and diagnostic warnings were communicated and properly processed by the IC application.

Graphical Panel Validation: The application-specific graphical panel presented real-time visualizations of speedometers, tachometers, warning lights, and message indicators. It presented intuitive monitoring of system responses, such as normal operating conditions and fault condition testing.



Automated Scenario Simulation: Graphical programming scenarios were created to simulate ignition cycles, speed ramps, indicator functions, and fault simulations. The use of developing reusable modular logic blocks greatly improved test simulation and promoted consistent behaviour when subjected to repeated rounds of validation cycles.

Fault Handling Verification: Controlled simulation CAN signal manipulation (e.g., loss of speed signal, activation of warning conditions) was used to verify the robustness of the Instrument Cluster in communication error and system fault handling. The cluster responded correctly by showing fault indicators and safe fallback states when required.

Efficiency and Coverage: SIL environment decreased the need for relying on actual hardware, allowing for early-stage validation and yielding better coverage of functional and erroneous cases. It led to more compact development cycles and better overall software quality.

Overall, SIL framework along with Rest bus Simulation and TS Master graphical programming was a highly efficient and effective method for Instrument Cluster validation. The process guaranteed mission-critical functionality was tested exhaustively, robustness was validated in different network environments, and system behaviour aligned with prescribed requirements before integration with real hardware.

Future Scope

The successful implementation of Software-in-the-Loop (SIL) validation for the Instrument Cluster using Rest bus Simulation and TS Master establishes a strong foundation for further advancements in automotive embedded system development and testing. Building upon the current framework, several future enhancements and expansions are identified:

- **Integration of Diagnostic Protocols:** Future work can include implementing and validating On-Board Diagnostics (OBD-II) and Unified Diagnostic Services (UDS) protocols within the SIL environment. This would enable comprehensive validation of diagnostic communication and fault management capabilities of the Instrument Cluster.

- **Extension to Multi-Network Support:** As modern vehicles utilize multiple communication networks such as CAN FD, LIN, FlexRay, and Automotive Ethernet, extending the Rest bus simulation to support these protocols would enhance the completeness and realism of the testing framework.

- **Model-Based Testing:** Incorporating Model-in-the- Loop (MiL) alongside SIL would allow early validation of system-level behaviour using Simulink or other model-based design tools. Co-simulation between SIL and MiL environments could ensure better alignment between model validation and software validation phases.

- **Automated Regression Testing:** Developing an automated regression testing framework using TS Master scripting and graphical programming could further reduce manual effort, ensure repeatability, and support continuous integration (CI) pipelines for software updates.

- **Fault Injection Automation:** Enhancing the graphical programming environment to include automated fault injection libraries would allow systematic evaluation of Instrument Cluster resilience against a wider range of communication and sensor faults.

- **Hardware-in-the-Loop (HIL) Transition:** The validated SIL scenarios can be adapted for Hardware- in-the- Loop (HIL) setups, where real Instrument Clusters and additional hardware interfaces can be tested in conjunction with the simulated vehicle environment, ensuring seamless transition from virtual to physical testing stages.

- **Advanced Visualization Techniques:** Future developments may include integrating 3D cluster visualization or augmented reality (AR)-based dashboards, providing even more accurate simulation of real-world user experiences and HMI interactions.

By pursuing these future directions, the SIL framework can be expanded into a more comprehensive, scalable, and intelligent validation platform, supporting the evolving complexity of automotive systems and contributing to the development of safer, more reliable vehicles.

CONCLUSION

This work successfully demonstrated the application of Software-in-the-Loop (SIL) validation for an automotive Instrument Cluster (IC) using Rest bus Simulation (RBS) and TS Master. By developing a comprehensive CAN database and leveraging TS Master's graphical programming and panel visualization features, a complete virtual environment was created to emulate real-world vehicle communication and system behaviour.

The use of Rest bus Simulation enabled accurate emulation of missing ECUs, facilitating early-stage validation of the Instrument Cluster software without the need for full physical hardware. Graphical programming provided an efficient and modular approach to building complex test scenarios, allowing dynamic control over signal flows, fault injections, and user interactions.

The results confirmed that the Instrument Cluster reliably interpreted simulated CAN signals, correctly displayed operational and fault states, and responded appropriately to network anomalies. Overall, the implemented SIL framework significantly reduced development time, increased test coverage, and improved the robustness and reliability of the Instrument Cluster software.

The approach outlined in this study offers a scalable and cost-effective solution for future automotive ECU validation processes, supporting the industry's growing need for early and efficient verification of complex embedded systems.

REFERENCE

- [1]. Surekha P. Gaikwad and Atmeshkumar S. Patel, "Automotive control system using CAN protocol", in International Journal of Advanced Research, Ideas and Innovations in Technology, 2019. ISSN: 2454-132X.
- [2]. Rakesh Ranjan, K Subramanyam Chari, Rajeev Arya and B Hari, "Design of Controller Area Network Based Automated Safety System for Vehicle", in international Conference on Intelligent Computing, Instrumentation and Control Technologies (ICICT) 2017. ISBN: 978-1-5090- 6106-8/17.
- [3]. A. A. Salunkhe, Pravin P Kamble and Rohit Jadhav, "Design And Implementation of CAN bus Protocol for Monitoring Vehicle Parameters", in IEEE International Conference On Recent Trends In Electronics Information Communication Technology, May 20-21, 2016, India, ISBN: 978-1-5090-0774-5/16.
- [4]. S. N Chikhale,"Automobile Design and Implementation of CAN bus Protocol- A Review", in IJRDO- Journal of Electrical and Electronics Engineering, Volume-4/Issue- 1/January 2018, ISSN: 2456-6055.
- [5]. Guilherme Marcon Zago and Edison Pignaton de Freitas, "A Quantitative Performance Study on CAN and CAN FD Vehicular Networks" in IEEE Transactions On Industrial Electronics, Vol-65, No.-5, May 2018, ISSN: 0278-0046.
- [6]. L. A. Perișoară, D. L. Săcăleanu and A. Vasile, "Instrument clusters for monitoring electric vehicles," 2017 IEEE 23rd International Symposium for Design and Technology in Electronic Packaging (SIITME), Constanta, 2017, pp. 379-382.
- [7]. Y. Tao, Y. Yue and P. Craig, "A hybrid approach to detection and recognition of dashboard information in real- time," 2017 4th International Conference on Systems and Informatics (ICSAI), Hangzhou, 2017, pp. 1141-1145.
- [8]. Y. Kwon, J. Choi, J. Jeon, K. Kim and B. Jang, "Design of Automotive Digital Instrument Cluster Adjustable to Driver's Cognitive Characteristics," 2019 International Conference on Information and Communication Technology Convergence (ICTC), Jeju Island, Korea (South), 2019, pp. 461-463.
- [9]. G. Vora and P. Gundewar, "Survey on Designing of Electric Vehicle Instrument Cluster," 2018 Second International Conference on Intelligent Computing and Control Systems (ICICCS), Madurai, India, 2018, pp. 765- 771.
- [10]. T. Pawlenka, J. Kulhánek, P. Tomčík and R. Zapletal, "Design of Digital CAN Based Car Dashboard Unit," 2019 20th International Carpathian Control Conference (ICCC), Krakow-Wieliczka, Poland, 2019, pp. 1-4.