UKACC International Conference on Control 2012
Cardiff, UK, 3-5 September 2012

Deployment of full vehicle simulator for electrical control system validation

Georgios Tsampardoukas
Jaguar Land Rover
Warwick, UK, CV35 0RR

Alexandros Mouzakitis
Jaguar Land Rover
Warwick, UK, CV35 0RR

***Abstract*—Development and testing of automotive embedded control systems traditionally depended on the availability of prototype vehicles. Automotive manufactures adopted model based approaches in order to produce quality products faster. Thus, the need of more integrated testing using virtual environment in an automated manner becomes a vital element of product development. The full vehicle simulator aims to provide a fully integrated environment for verification and validation of the embedded automotive software to avoid the dependency of the prototype vehicles. The execution of different automated test scenarios aims to increase the development of the product faster without compromising robustness and quality. This paper deals with both the development of full vehicle simulator and the concept of the automated modelled test cases.**

Keywords- Hardare-in-the-loop; full vehicle simulator; automated modeled test cases, automotive control systems.

I. INTRODUCTION

The embedded software complexity and the number of electronic control units (ECU) integrated in modern luxury vehicles are radically increased due to the increasing distributive functionality, safety requirements and legislation for lower emissions. Today's luxury vehicles include more than 60 interconnected ECUs using various network systems [1]. Four main domains such as body systems, chassis, powe rtrain and infotainment constitute a typical vehicle electrical architecture. Mots vehicle functionality is distributed among these domains as shown in [2],[3] and [4].

The power train and chassis domain contains ECUs responsible to control systems such as engine management, anti-lock brake system, hybrid systems, transmission and vehicle dynamics [5] and [6]**.** These are generally continuous control systems and are interconnected using the high speed CAN network.

The body system domain, however, is responsible to deal with systems like security, locking, wipers, mirrors, start authorization, etc. These control systems are mainly event driven with response time slower than the powertrain domain. The characteristic of the body domain is that the overall functionality (i.e. locking) is distributed to more than one ECU. These ECUs are located on one or more network buses [7] and [8].

The infotainment domain is one of the most popular systems nowadays due to its human machine interface nature between and the driver and the vehicle. This system usually consists of DVD player, amplifier, human machine interface console, TV modules and navigation. Other infotainment features require communication with the external world using external media such as Bluetooth and Wi-Fi. Typically, the response time of these systems is very slow due to the nature of consumer electronics functionality. The data bandwidth required to run an infotainment system is high compared to powertrain and body domains. The driver interaction with the vehicle makes the infotainment system one of the hottest topics in modern automotive industry.

Distributive functionality shared amongst four domains can impact customer's perception about vehicle quality. Software complexity and programme development cycle substantially reduced due to continuously customer's demand for new features. Competition amongst vehicle manufactures radically increased due to demand for robust and quality vehicle systems. Advanced and sophisticated techniques (i.e. hardware-in-the-loop) commonly employed to validate the embedded software in real time early at the product development [4]**.** The drive to reduce dependency in prototype vehicles is still an important initiative for most vehicle manufacturers. The usage of prototype vehicles is therefore aimed mainly for verification and validation activities close to mass production data**.** Automated virtual testing environment promotes more robust, systematic, time efficient and cost effective way for software testing. It has the potential to uncover possible software failure modes and to perform fault diagnostics automated tests prior to development of prototype vehicles [8]**.**

This paper is organised in the following manner: The first section deals with introduction and brief literature review in the area of automotive control system development and test. The second section presents the development of the full vehicle simulator within Jaguar Land Rover (JLR). Two case studies are considered in the next section. The results and benefits of the automated testing are depicted in section four. Section five illustrates the discussion about the main benefits of the automated modelled test cases. The last section gives some concluding remarks of the work presented in this paper.

II. FULL VEHICLE SIMULATOR OVERVIEW

The main scope of this section is to describe the structure of the full vehicle simulator (i.e. fully integrated hardware-in-the-

loop platform) suitable for automated functional and non-functional testing.

A. *Simulator setup and schematics*

The full vehicle simulator consists of three 21 inches cabinets and two load tables. The load tables hold vehicle real loads and ECUs. Figure 1 show the full vehicle simulator which is currently used within the premises of JLR.

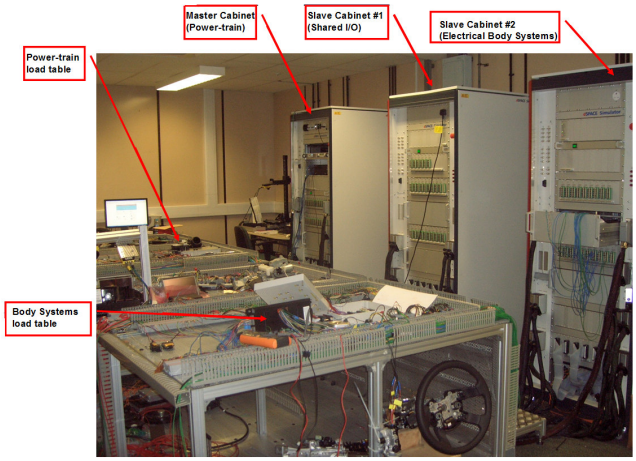

Figure 1. Full vehicle simulator at JLR.

In the following sections the high level requirements including the number of input and output (i.e. I/O) channels specified for the full vehicle simulator are given. The amount of the I/O required to interface with the ECUs to form the vehicle architecture clearly demonstrates the complexity of the system.

The main features of this simulation platform can be summarized as follows.

149 ADC; 145 DAC; 275 digital input; 79 PWM , 195 digital out, 126 digital relay output, 88 PWM output, 24 resistive channels, 22 special channels for powertrain simulation, 110 dedicated power lines and GND, 8 CAN channels, 32 LIN channels used to interface with ECU hardware. Quiescent current measurements an all power lines (i.e. 2 power switches), 1 power supply (i.e. 400A@20V), access to CAN and LIN channels for measurements, Serial ports from processors (i.e. 3 DS1006 quad processors), Integration of the low voltage tester (i.e. LVT), On/Off capability on all power lines, Fault insertion and load boards on the input channels w.r.t (with regards to the) simulator, fault insertion capability on all CAN and LIN channels, ABS valve detection unit, colour coding on the on each I/O type, Special software to interact with the simulator (i.e. Control Desk, Automation Desk, Motion Desk, CAN/LIN multi-message), Measuring point for the main power supply (i.e. 4 banana plugs on the front side of the first cabinet).

The first cabinet (i.e. master cabinet) dedicated for the powertrain, chassis and driveline domain (i.e. high speed CAN network domain). For instance, engine management system (i.e. EMS), chassis control and transmission control ECUs are integrated on the master table. Peripheral loads such as electronic throttle, injectors, differential and transmission solenoids interfaced to ECUs via the master load table to the simulator. The aim of load integration is to enable the ECUs to functionally operate in as close as possible to the environment of the vehicle. Thus, the number of logged diagnostic trouble codes substantially reduced. Engine and transmission plant models developed and executed in real time in order to provide dynamic closed loop control between the EMS module and the models of the engine/driveline.

The second cabinet (Figure 1) is known as first slave to the master cabinet. The use of this cabinet is to provide bulk I/O (i.e. input and output) channels to powertrain and body tables. These I/Os are distributed on both load tables to provide enough channels to ECUs for the interface with the simulator.

The third cabinet (i.e. body systems cabinet) dedicated to electronic body systems (i.e. body control module, door modules, keyless vehicle module, etc) and is known as the second slave of the master cabinet. This domain is dedicated to medium speed CAN modules and their LIN slaves (i.e. intrusion monitoring system). The second load table (i.e. body systems table as shown in Figure 1) is used to accommodate all the medium speed ECUs and their peripheral components..

B. *Integrated model to control the full vehicle simulator*

The VITAL framework is a generic model that built to interface with twelve different core processors (three quad processors DS10006) of the full vehicle simulator. The aim of this framework is to create a structured environment for integration of potentially of vehicle electrical components. Additional features of this model are summarised below:

- The model structure allows integration of three dSPACE Quad core processors or more.
- Enables simulation of Multi-CAN/LIN architectures and other network protocols (i.e. Ethernet)
- Exchange of signals between cores of each processor.
- Selection mechanism for switching between real and modelled ECUs
- Common interface for simulation of ECU loads and actuators.
- Hardware interface between ECUs and simulator arranged per dSPACE IO board..

The above features offer less development time, more optimal model structure resulting in more efficient real time execution. In addition to that the model promotes consistency and reduces the risk for development errors.

Figure 2 show only an example of the VITAL model and the closed loop integration method between the real driver door ECU and the latch plant model. The plant model in this example provides the feedback signals to the driver's door control unit (i.e. DDCU). The door ECU outputs are fed back to the door latch model and the feedback of the door latch is fed back to the door ECU via the simulator IO channels.. The

door latch plant model developed in Stateflow® is a true functional representation of the real latch [9].

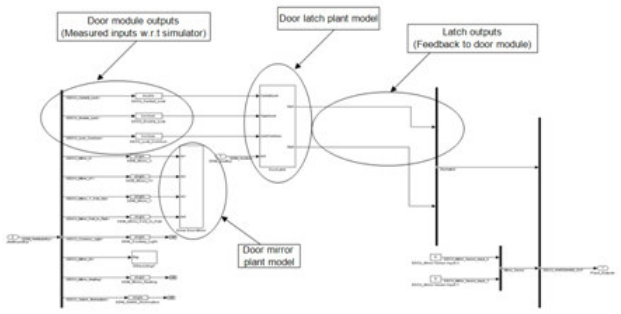

Figure 2. Hardware interface on VITAL framework for closed-loop control

More detailed description about the VITAL framework is presented in [9]

III. CASE STUDY (CLOSED LOOP CONTROL OF BODY SYSTEM FUNCTIONALITY)

The aim of this section is to demonstrate two automated test scenarios using the fully integrated platform described previously. The first scenario is the drive away door locking and the second one is the valet mode.

The drive away door locking function is a security feature that locks all the doors automatically when the vehicle speed exceeds the threshold speed (i.e. 32Km/h). This feature is selectable by the driver, and any operation of the door locks by any other means (i.e. master locking switch on the facial panel) will unlock the doors [10].

Valet mode is also a security feature that allows the vehicle to be driven with the luggage compartment locked with restricted touch screen functionality. This feature is accessible directly from the home menu on the touch screen or from the vehicle settings screen. The vehicle owner enters a four digit Personal Identification Number (PIN) to a soft key pad displayed on the touch screen. This PIN must be entered twice in order for the valet mode to be enabled. A pop-up screen is displayed, confirming that the vehicle is now in valet mode. To cancel the valet mode operation, the PIN number must be entered once again [10].

A. *System Overview*

The deployment of the drive away door locking and the valet mode features is depicted in Figure 3. Several ECUs are required to exchange data amongst different domains in order to interpreter the customer's operation into low level software command (i.e. set the threshold of drive away door locking).The simulation environment to deliver these features required the integration of the following ECUs and models, engine controller; gateway controller; door controllers; infotainment control units; engine and driveline real-time models.

The complexity of system integration significantly increases due to the following reasons, number of ECUs; the scale of the system integration; inter-dependencies across functional areas and domains. For instance, the DDCU receives the command from the gateway to lock the door latch when the EMS transmit the correct vehicle speed from the high speed CAN network. An engine model and accurate sensor simulation required for the real EMS hardware to assume that an engine is in operation and the vehicle is in a drive cycle. Although drive away door locking feature appears to the customer to be a simple operation the effort required to develop the automated virtual environment is certainly a challenging engineering task.

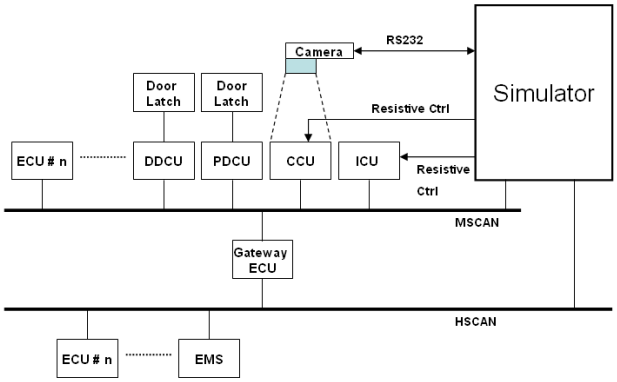

Figure 3. Schamatic of vehicle feature demployment

B. *System Integration*

The full vehicle simulator is interfacing (Figure 3) the ICU using resistive signals for simulation of the driver switch-pack component which is hardwired back to the ICU (this switch-pack controls the navigation menu shown in the CCU'display). The simulator is also providing resistive signals to the ICU in order to emulate driver touch screen selection. High resolution camera is integrated with the simulator via serial (i.e. RS232) connection to provide feedback acquired by the image processing software. This camera is used as feedback sensor to capture the results from the instrument cluster and infotainment display. Detailed description about multi-camera vision system is out of the scope of this paper.

The two front door modules (i.e. driver and passenger), instrument cluster and infotainment display have an interface to the medium CAN bus. The two door latches are integrated to each door module using hardwire connection. The gateway ECU (i.e. ECU that accommodates the core body system functionality) is used to pass network signals from medium speed CAN bus to high speed. The EMS is interfaced with the engine model to high speed CAN in order to provide engine and vehicle speed to the rest of the vehicle systems. The rear door modules are not assessed on this paper since their behaviour is very similar to front doors.

C. *Event driven control Logic*

Event driven control logic is a discrete programming method that is based on the conditional transition between operating modes. This method is used in this paper to model the automated control sequence of the infotainment displays selection.

1) Automatic navigation on instrument cluster

This control logic is employed for automatic selection of the drive away door locking threshold. The model is divided in two main parts.

The first part deals with the simulation of a button (i.e. up, down, left, right and OK button) from the driver's switch pack. Figure 4 shows a snapshot of this model which represents one event of the OK press button. The event sequence is required to alter the simulator's resistive output from idle (X Resistance) to pressed position (Y Resistance). The transition between the two states delayed for 500 milliseconds. This allows the instrument cluster to process the request received from the infotainment control unit. A counter is implemented to capture the complete sequence. It is also used as a transitional condition to another event (i.e. simulation of down button). In addition to counter, the camera feedback is used as an alternative transitional condition. The camera is trained to identify the main menu pattern. This menu is displayed after pressing the OK button. Figure 4 shows that either the camera feedback or the counter conditions must be satisfied in order to continue the sequence.

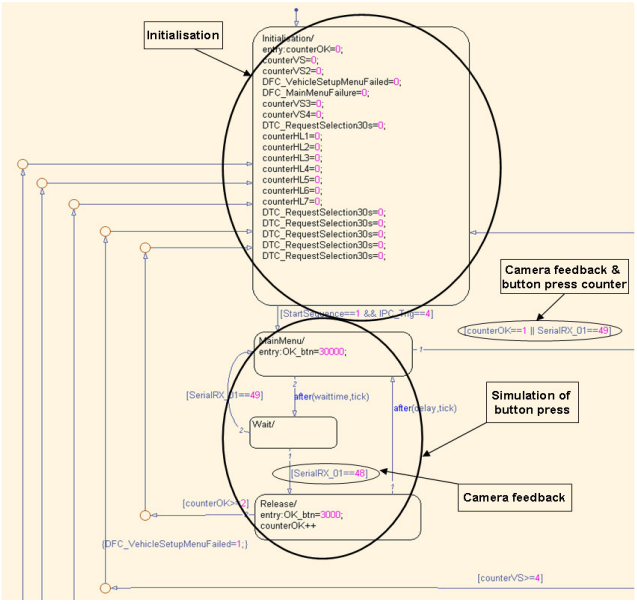

Figure 4. Automatic navigation through the instrument cluster menu

The second part (not shown in this paper) of this control logic is the evaluation of the menu position via the camera's feedback. The camera is placed to point directly the instrument cluster. It is trained to recognise five different positions of the drive away door locking settings menu. The control logic identifies the menu selection and responds to the driver's choice. For instance, different sequence is employed in order to set 32Km/h when the default position is on OFF mode and different sequence when the value is 5Km/h. The difference is due to the number of down or up button event presses. Similar concept is used to control the automatic selection of valet mode.

2) Valet Mode

The driver inserts twice a predefined four digit code to ICU touch screen in order to set the vehicle to valet mode. The automatic sequence requires selection steps on the ICU menu before the driver enters the PIN number. An event driven control logic is developed using Stateflow® to achieve this automated selection

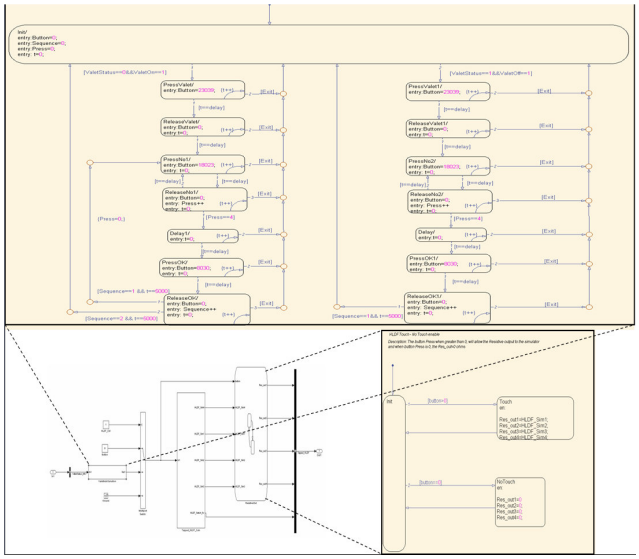

Figure 5. Automatic control of infotaiment display (i.e. valet mode selection)

Figure 5 shows only the sequence to set the valet mode on. The first state defines the initial conditions and determines the idle mode. On this mode the control logic provides the idle resistive output to ICU via the simulator's restive channel. The sequence starts when all the initial transitional conditions are satisfied. Delay of 800ms is implemented between the states after the idle mode. This allows the ICU to process the request from the driver and the infotainment graphics. Exit conditions are implemented on every stage of the sequence to ensure smooth execution avoiding stagnation points. On this particular example a medium speed CAN signal is used to inform about the valet mode status instead of camera feedback.

D. Control desk interface

Control desk layouts are developed in order to control the full vehicle simulator (i.e. 20 controlDesk® tabs) via graphical user interface. Figure 6 shows only a sub-set of the main working layout. This is dedicated to control the automated selection of the drive away door locking. It is also shown that the camera mode and image processing job is controlled from this layout. The manual mode of the driver's switch pack is also part of this interface as shown in Figure 6.

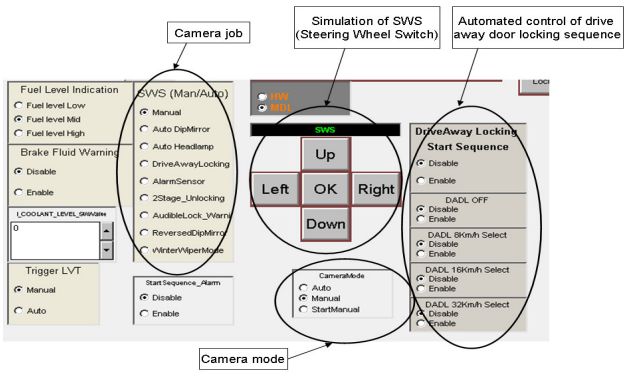

Figure 6. Graphical user interface for automated drive away door locking selection mode.

A Similar graphical user interface is developed to control the automated valet mode selection. However, this is not shown in Figure 6.

E. Manual test scenario description

A manual test scenario is performed to set the drive away door locking threshold from OFF mode to 32Km/h. The flowchart in Figure 7 depicts the manual sequence.

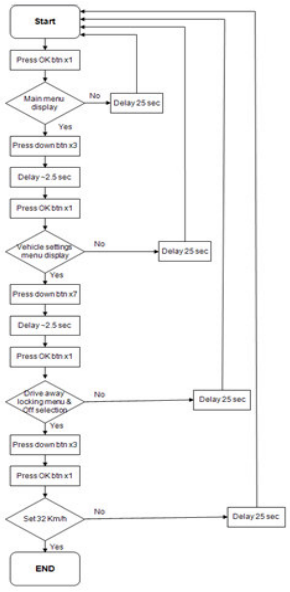

Figure 7. Flowchart of manual drive away door locking from OFF to 32Km/h

The sequence starts from a warning free instrument cluster. Figure 7 shows the exact sequence required for the instrument cluster to display the driver away door locking settings menu. Since this is a submenu, it is required to navigate three layers below the main menu. An extra delay is introduced on every conditional state (approximately 2.5 seconds) to allow the camera to process the captured image. Based on the camera feedback, the control logic decides about the status and the progress of the automated sequence. Similar manual test scenario is produced for the valet mode. The only difference is that the navigation is through the resistive touch screen of the infotainment display. The flowchart for the valet mode selection is not shown in this paper.

IV. TEST AUTOMATION

The purpose of the automated testing is to execute the existing manual test cases in a repeatable manner. MXvDEV® test automation software is used to model the automated test cases.

A. Execution of the automated test sequence

Both automated sequences are executed and the results are captured in a graphical manner. Figure 8 and Figure 9 present the test results for drive away door locking and valet mode, respectively.

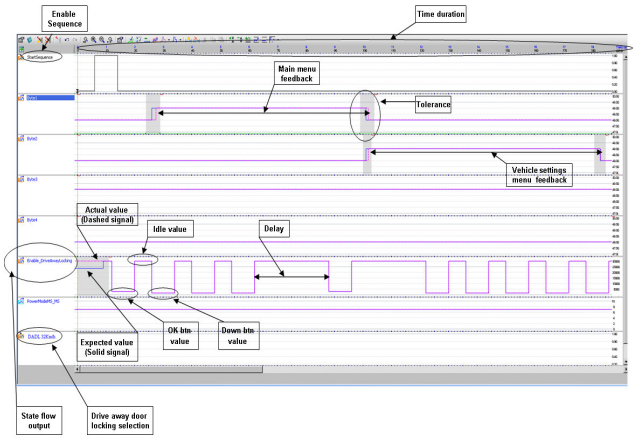

Figure 8. Automated test results for drive away door locking to 32Km/h

For presentation purposes, only half duration of both scenarios are presented in Figures 8 and 9.

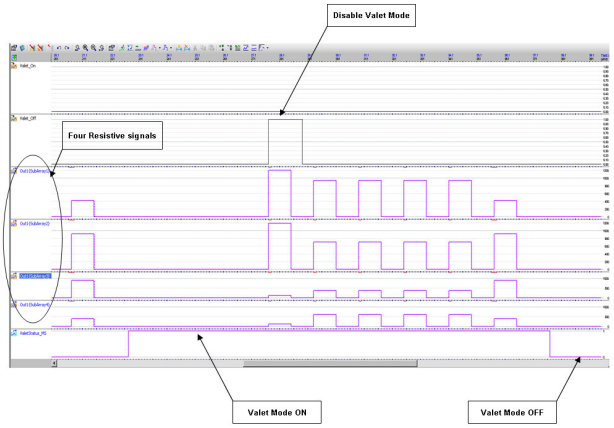

Figure 9. Automated test results to disable the valet mode.

The desired signals are graphically modelled as shown in Figures 8 and 9. It is difficult to distinguish the differences between the actual and desired signal. This is due to exact

match of the two signals. For that purpose, the expected signal (i.e. sixth signal in Figure 9) is altered at the start of the sequence. A time delay is implemented on x-axis (time axis) at the first 500 milliseconds of the test in order to deliberately make the test cases result a fail (i.e. a shadowed area).

V. DISCUSSION

The evaluation of the test results has shown that both case studies (i.e. Figures 8-9) were executed successfully in an automated manner. These scenarios performed with zero tolerance on the y-axis. The expected signal has identically matched the actual measured. Occasional deviations observed on x-axis (i.e. time), where slight time variations (i.e. milliseconds) occurred. This is due to the time delays of the real time processor to fetch and process the results. Thus, signal delay of few milliseconds was introduced in the area of interests (i.e. shadowed areas on x-axis) to avoid failures of the test case.

The benefits of the modelled test cases are summarised below:

- Accurate definition of the test case in terms of time and signal definition.
- Graphical representation of the signal during the test execution.
- Reduced test case duplication due to test case re-use.
- Less specialised knowledge is required to analyse the test results.
- The graphical definition of the test case can uncover failure modes associated with the functional requirements prior to the test execution.
- Test cases can be linked to system requirements.
- Visualisation of signal tolerances (i.e. y-axis).
- The review of the automated test case and scenarios is significantly minimised.

The creation of the modelled test cases require well defined design verification plan and signal specification standards to be in place. This approach of automation helps JLR to move test creation using tabular format to one with graphical representation.

In addition to the above remarks regarding the creation of automated test cases, the deployment of the full vehicle simulator has demonstrated the following potentials.

- Most Electrical functional requirements can be validated prior to prototype build.
- Distributed functionality validation is decoupled from single software release.
- Drive cycles which have functional safety implications, can be executed in a controlled test environment.
- The product design can be evaluated and altered early in the programme before commitment to tier 1 is made.
- Early feature demonstration can help towards concept selection and decision making.
- Enables cross functional team working and explores opportunities towards “what can we do better and how?”

Although the aforementioned characteristics clearly deliver competitive advantage to an OEM, there are points to be considered before full deployment takes place.

- Significant upfront capital investment is required to purchase the hardware and software simulation components.
- Early engineering effort is required to develop product engineering specification.
- Engineering mind set shift from manual vehicle testing to automated simulation based testing.

VI. CONCLUSION

The purpose of this paper was to present a fully integrated hardware-in-the-loop environment for validation of distributive vehicle functionality. The automated modelled test cases concept is introduced with the execution of two case studies. The successful execution of both scenarios has proven that the entire vehicle functionality can be modelled and executed on the full vehicle simulator.

REFERENCES

[1] Waltermann, J., 2009. Hardware-in-the-loop: The Technology for testing Electronic Controls in Automotive Engineering. *6th Paderborn Workshop: Designing Mechatronic Systems, Paderborn, April 2-3,Germany*

[2] Kendall,I,R. and Jones,R,P., 1999. An Investigation into use of hardware-in-the-loop simulation testing for automotive electronic control systems. Control Engineering Practice, 7 (1999), p.p 1343-1356.

[3] Lamberg, K.,Richert, J. and Rasche, R., 2003. A new environment for integrated Development and Management of ECU tests. Proceedings of the SAE World Congress, Detroit,USA.

[4] Mouzakitis,A.,Humphrey,R.,Bennett,P. And Burnham,J,K. 2006. Development, Testing and Validation of Complex Automotive systems. The 10th Mechatronic Forum Biennial International Conference,MX2006,Philadelphia,USA.

[5] Dhaliwal, A., Shreyas,C., Nagaraj,C, and Syed,A. 2009. Harware-in-the-loop Simulation For Hybrid Electric Vehicles-An Overview, Lessons Learnt and Solutions Implemented. SAE Technical Pepers 09AE-0198.

[6] Wu,K., Zhang,Q. And Hansen,A. 2004. Modeling and identification of a hydrostatic transmission hardware-in-the-loop simulator. International Journal of Vehicle Design, Vol.34,No. 1.

[7] Henselmann,H. 1993. Hardware-in-the-loop simulation as a standard approach for development, customisation and production test. SAE technical papers, 930207.

[8] Tsampardoukas,G., Mouzakitis,A. And Sydor,P., 2009. Design Methodology for Integrating Networked Automotive Electronic Control Units Using Hardware-in-the-loop. Proceedings of the 20th International Conference on Systems Engineering ,Coventry, UK.

[9] Huang, Y., McMurran, R., Dhadyalla, G., Jones, P. and Mouzakitis., 2009. Model-based testing of a vehicle instrument cluster for design validation using machine vision. Journal of Measurement Science and Technology, Vol. 20, No. 6.

[10] Jaguar Land Rover, User's Handbook, 2009.