Reconfigurable Real-Time Hardware-in-the-Loop Environment for Automotive Electronic Control Unit Testing and Verification

Krisztian Enisz, Denes Fodor, Istvan Szalay, and Laszlo Kovacs

To satisfy the ever increasing needs for safety, comfort, and environment protection of today's vehicles, the Electronic Control Units (ECU) and various sensors are getting more and more complex [1], [2]. Therefore, developing new control algorithms and cost efficient verification tools for a new generation of ECUs and sensors has become a highly important issue. This paper deals with the design and implementation of a versatile automated Hardware-in-the-Loop (HIL) test environment, which facilitates the development of control algorithms, calibration, and verification of *state of the art* sensors and ECUs [3]. The environment is able to emulate the vehicles' dynamic behavior, reduces the time required for development and testing, and eliminates the need for using expensive real vehicles for testing purposes.

Some existing verification and test systems are available from various manufacturers like dSpace, ETAS, Systerra, etc. These systems are suitable for special tasks such as development or verification of a specific sensor and ECU [4]; however, these environments have certain disadvantages:

- inflexibility, lack of easy adaptability for different projects; and
- a long project preparation time (often requiring the support of the manufacturer to setup or update the test environment).

Based on a market survey of existing simulation environments and on consultations with industrial partners, a National Instruments (NI) based real-time hardware and software environment was selected. The functionality of the system was demonstrated by the integration of a previously tested Antilock Breaking System (ABS) electronic control unit and by the implementation of communication and function tests.

System Requirements

During the definition of the simulation environment, recommendations of an industrial partner Continental Automotive Hungary Ltd. were taken into account. Until recently at the laboratory of Continental Automotive Hungary, old Systerra type Versa Module Europa (VME) bus-based simulators had been in use. The simulation environment had five main parts (Fig. 1).

The real-time simulator (RTS) provided the simulation environment and simulated every signal for the ECU. The RTS had a real-time operating system (VxWorks) and had to emulate both digital and analog sensors and different communication methods, e.g., a Controller Area Network (CAN) or FlexRay.

The electronic failure generator (EFG) was an extension of the RTS, which could test the Hydraulic Control Unit's (HCU) analog valve activity and was capable of generating valve errors, too. The so-called troubleshooter was a simple jumper-box. It enabled the user to setup the test environment more accurately if it was necessary in regard to the tested ECU, or to generate further failures. Hence, the use of this unit was optional. The data acquisition system (DAS) and the high-speed module (HSM) were the observation parts of the HIL environment. They ensured that the outputs of the ECU's software were measurable in detail. These simulation systems

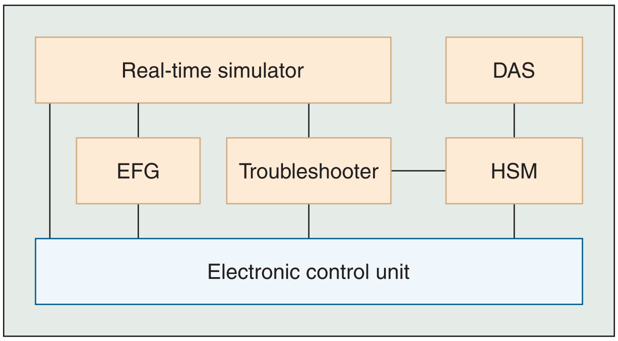

Fig. 1. Architecture of the old Systerra based simulation environment.

We thankfully acknowledge the financial support of the Hungarian State and the European Union under the TAMOP-4.2.2-A-11/1/KONV-2012-0012 Cooperation between Higher Education, Research Institutes, and the Automotive Industry project and TAMOP-4.1.1.C-12/1/KONV-2012-0017.

1094-6969/14/$25.00©2014IEEE

have become out-of-date, and they are not suitable for complex maneuver based tests because the maneuvers had to be created manually by writing different script language based programs.

To overcome the limitations, the goal was to create a more advanced environment with an up-to-date software and hardware architecture which is suitable for industrial projects and also for educational and research activities at the University of Pannonia. To support development and testing of different kinds of ECUs, like battery sensors and deflection detection systems as well as ABS and Electronic Stability Program (ESP) control units, the following requirements were defined:

- facilitate the development of vehicle dynamics based algorithms;
- be able to test electronic control units and control algorithms in many respects;
- have the capability to create and to run automated test sequences;
- be able to generate customizable reports;
- be extendable and flexibile to be adjusted easily for different tasks;
- provide short teach-in time for users, researchers, and students with basic knowledge of NI Lab VIEW and MATLAB/Simulink; and
- be capable of running stimuli (signal, sensor emulation) and vehicle dynamics based simulations and tests.

Considering the above requirements, a new simulation environment was developed with the contribution of National Instruments and Continental Automotive Hungary Ltd.

Simulation Environment

The HIL simulator (Fig. 2) has a modular architecture and can easily be reconfigured for different projects. Almost the whole development and test process can be performed on the same system, from the design of new algorithms to the HIL tests [5].

The hardware architecture (Fig. 3) consists of three main parts:

- a standard high performance industrial computer;
- an NI PXIe (Peripheral component interconnect eXtensions for Instrumentation Express) real-time system with field-programmable gate array (FPGA) based reconfigurable I/O and communication interfaces; and
- an electric motor emulator for electric car ECUs.

The Windows-based high performance computer (PC) does not have any special interfaces. It provides the software environment for modeling vehicle dynamics and for creating test maneuvers.

On the NI-PXIe-8135 real-time controller, a real-time operating system (Pharlap) can be run which provides the run-time environment for Lab VIEW and Veristand. The PXIe system has several interfaces such as CAN (PXI-8513/2), MOST (VN2610), FlexRay (PXI-8517/2), LIN (PXI-8516), and FPGA based cards (PXI-7851R) for digital and analog I/O. The FPGA cards can be extended with NI CompactRIO (Reconfigurable Input and Output) communication digital and analog I/O modules. A great advantage of these FPGA cards is that there

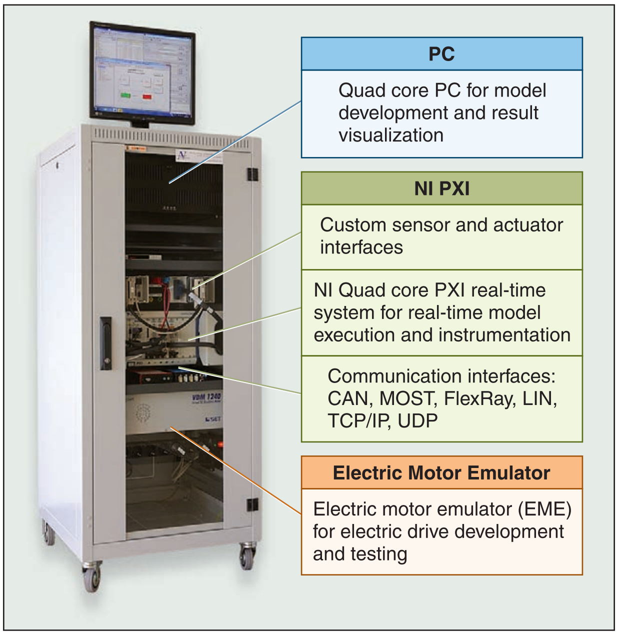

Fig. 2. Physical appearance of the new simulator.

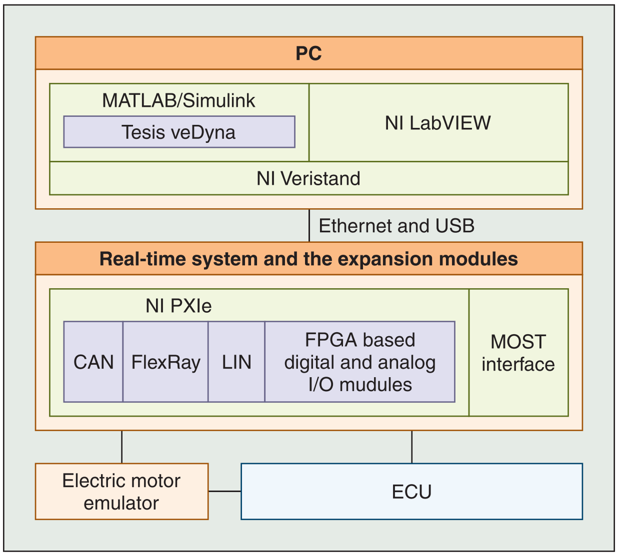

Fig. 3. The anatomy of the HIL simulation environment.

are a lot of predefined programs for various functionalities like signal conditioning, filtering, Pulse-Width Modulation (PWM) generation, etc. Programing of the FPGA cards can be effectuated in the LabVIEW development environment, and therefore, special skills in very high-speed integrated circuit hardware description language (VHDL) are not required.

The simulation environment has an electric motor emulator part which facilitates the development of ECU control algorithms for electric drives. The software architecture has a hierarchical structure; different development tasks can be implemented on different software levels. The lowest level is for modeling and algorithm development. For vehicle dynamics modeling, Tesis veDyna was used, which provides a MATLAB/Simulink based library and user interface where new algorithms can easily be implemented and integrated into vehicle dynamics. The vehicle model design of the veDyna is based on a multi-body system structure (MBS) with additional features (e.g., elastokinematics and tire model) and time optimized equations of motion. The model is suitable for real-time simulation and guarantees stability for a vast range of operating points [6]. In this environment, it is easy to create new maneuvers, parameterize various vehicle models, and define even road surface characteristics. It can be extended freely with new control algorithms for vehicle stability systems. The Tesis veDyna can be used without the real-time system in *offline* mode, and it can run simulations on the PC as well. The preferred development environment for the lower level modeling tasks is LabVIEW.

NI VeriStand is a software environment for configuring real-time testing applications, which makes it possible to execute tasks such as real-time stimulus generation and data acquisition for high-speed and conditioned measurements. It can import control algorithms, simulation models, and other tasks from NI LabVIEW, MATLAB/Simulink, ANSI C/C++, etc., and the operator can monitor and interact with these tasks using the run-time editable user interface.

Validation of the Simulation Environment

The implemented hardware and software environment was validated with testing the ESP sensor clusters, ABS and ESP ECUs, and intelligent battery sensors. In these test cases, the communication and functional behavior was verified by monitoring the input and output signals of the ECUs and communication signals such as CAN and LIN frames.

The integration procedure and verification of the control algorithm of an ABS ECU will be presented in the following sections. Based on standard industrial verification procedures, a real test sequence was executed. During the test, standard emergency braking maneuvers were executed in the simulation environment, and the brake pressure modulations were examined. The results are shown in the section titled Test Implementation and Evaluation.

ABS ECU Integration

The real ABS ECU was extended with simple and small electric circuits to emulate the wheel speed sensors and the valves and motor of the HCU (Fig. 4). The motor and the valves were controlled by the ECU; the Real-Time system measures the actual state of these components. Wheel speeds were calculated by running the vehicle dynamics based simulation (veDyna model). The wheel speed signals were generated as square voltage signals, and their duty cycle and frequency were modulated according to the calculated wheel speed values. The voltage signals were converted by the wheel speed sensor emulation circuit to current signals suitable for the ABS ECU. The emulated hardware components could be replaced by the Real-Time system; however, in the current implementation, real electric circuits were used to facilitate the comparison of the new environment with the former verification and validation environment of the industrial partners. The CAN

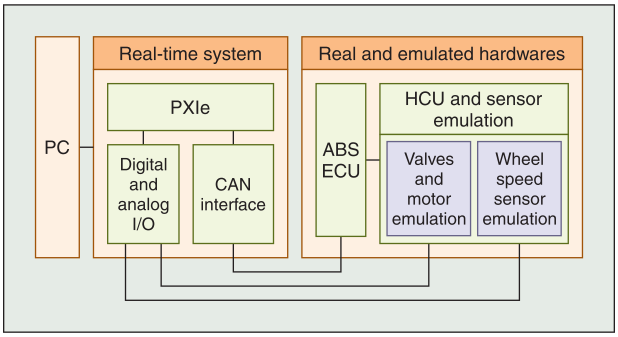

Fig. 4. Hardware integration of the ABS ECU.

communication interface was used to monitor the internal variables and the actual state of the ABS ECU and to support the communication tests.

After hardware integration was accomplished, the next step was software integration. The most important task was extending the veDyna Simulink model with a new HCU model, which made it possible to modulate the brake pressure independently for each wheel based on the ABS ECU control signals (valve positions and motor state shown in Fig. 5), as in a real vehicle.

The HCU model was implemented according to (1):

$$p(n) = p_{\text{target}}(n) - (p_{\text{diff}}(n))e^{\frac{-t(n)}{\tau}} \tag{1}$$

where:

- n is the discrete time variable;
- $p(n)$ is the calculated brake pressure for the wheel;
- $p_{target}(n)$ is the target brake pressure;
- $p_{diff}(n)$ is the difference between the current and the target brake pressure;
- $t(n)$ is the current time of the cycle; and
- is the time constant of the system.

The parameters of the main equations were calculated by an algorithm which had six different modules:

- state selection (`increase`, `hold`, `decrease`);
- main pressure change control;
- cycle interval calculation;
- target pressure calculation;
- pressure difference calculation; and
- brake pressure calculation.

A main pressure change control was necessary to follow the evolution of linear variation of brake pressure caused by the driver. The model built up a linear pressure adjustment with small steps where the step size p_{max_change} was kept at a constant value and p_{main} was the unmodified brake pressure for the wheel in (2) and (3). `If` the selected state was `increase`, and $|p_{main}(n)-p_{main\_start}(n-1)| \geq p_{max\_change}$, `then`:

$$p_{\text{main\_start}}(n) = p_{\text{main}}(n) \;\;,\;\; P_{\text{main\_changed}}(n) = true \tag{2}$$

`else`,

$$p_{\text{main\_start}}(n) = p_{\text{main\_start}}(n-1) \;\;,\;\; P_{\text{main\_changed}}(n) = false \;. \tag{3}$$

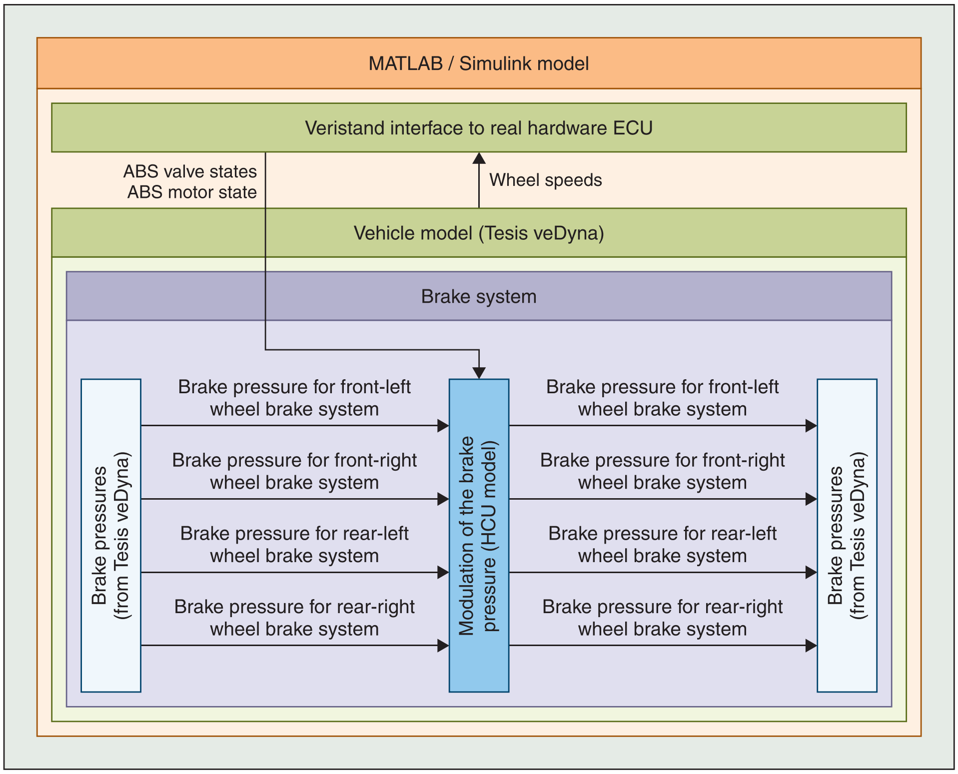

Fig. 5. Integration of the real ABS ECU and the simulated HCU into the Tesis veDyna.

The values of the current cycle interval were provided by the cycle interval calculation module, where the t_{step} constant was the simulation step length (4 and (5). `If` the present state did not match the previous state, and ($p_{main\_changed}(n)$ = true or $|p(n-1) - p_{main}(n)| \le p_{max\_change}$), `then`:

$$t(n) = t_{step} \;,\; t'(n) = t(n-1) + t_{step} \tag{4}$$

`else`

$$t(n) = t(n-1) + t_{step} \;,\; t'(n) = t(n-1) + t_{step} \;. \tag{5}$$

The target pressure was selected according to the following three cases (6) – (8). `If` the present state did not match the previous state, $p_{main\_changed}(n)$ = true, and the selected state was `increase`, `then`:

$$p_{target}(n) = p_{main}(n) \;. \tag{6}$$

If the present state did not match the previous state, $p_{main\_changed}(n)$ = true, and the selected state was `hold`, then:

$$p_{target}(n) = p_{target}(n-1) - p_{diff}(n-1)e^{\frac{-t'(n)}{\tau}} \;. \tag{7}$$

In any other cases:

$$p_{target}(n) = p_{target}(n-1) \;. \tag{8}$$

The difference between the current and the target brake pressure was calculated by the pressure difference calculation module (9) and (10). `If` the present state did not match the previous state, and $p_{main\_changed}(n)$ = true, `then`:

$$p_{diff}(n) = p_{target}(n) - p(n-1) \tag{9}$$

`else`

$$p_{diff}(n) = p_{diff}(n-1) \;. \tag{10}$$

The model and algorithm presented above were suitable for implementation in the Veristand environment.

Test Implementation and Evaluation

Using the previously integrated ABS ECU, which was already verified by industrial verification tools (Systerra based Continental EZS 2007 v4.0), the test environment as a whole was tested through vehicle dynamics based ABS ECU function and verification tests. During the implementation, a new Veristand project was created to realize the test sequences of the ABS ECU, including communication, ABS actuation, and fail-safe tests.

The communication test sequences covered the CAN conformance and reliability analyses. The conformance tests analyzed the correctness of the messages sent and the processing of the CAN frames received during the operation. It was tested if the CAN messages were sent with correct repetition rate, correct length, etc., and whether the payload modification of the frames were correct due to applying a stimulus (i.e., increasing or decreasing the wheel speeds, etc.). The reliability tests examined the reaction of the ABS ECU to lost and

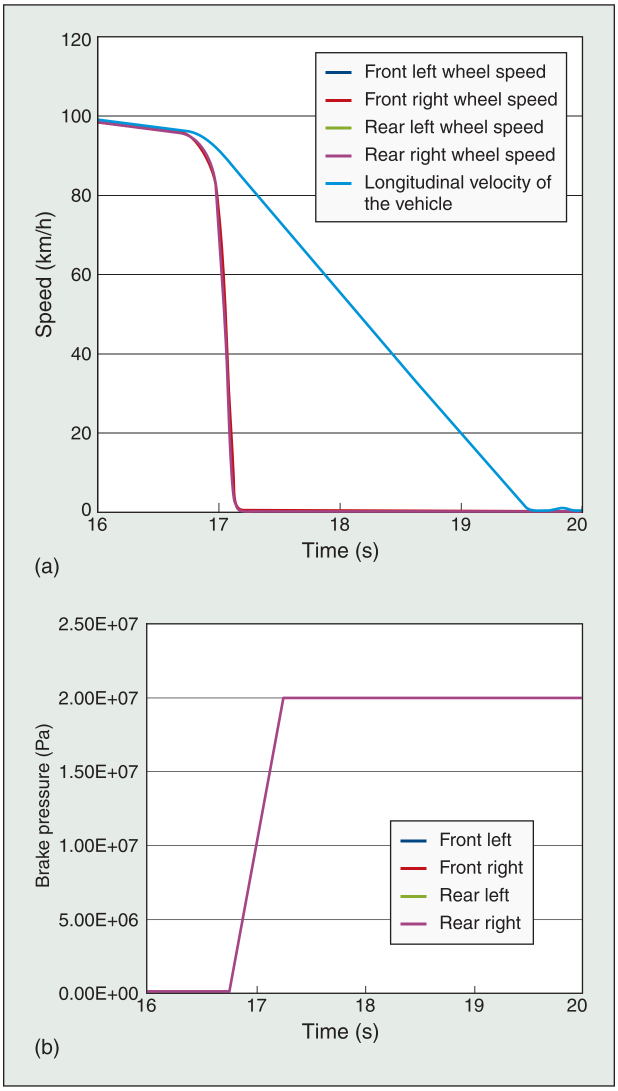

Fig. 6. Vehicle velocity, wheel speeds, and brake pressures of the ABS ECU with the ignition off.

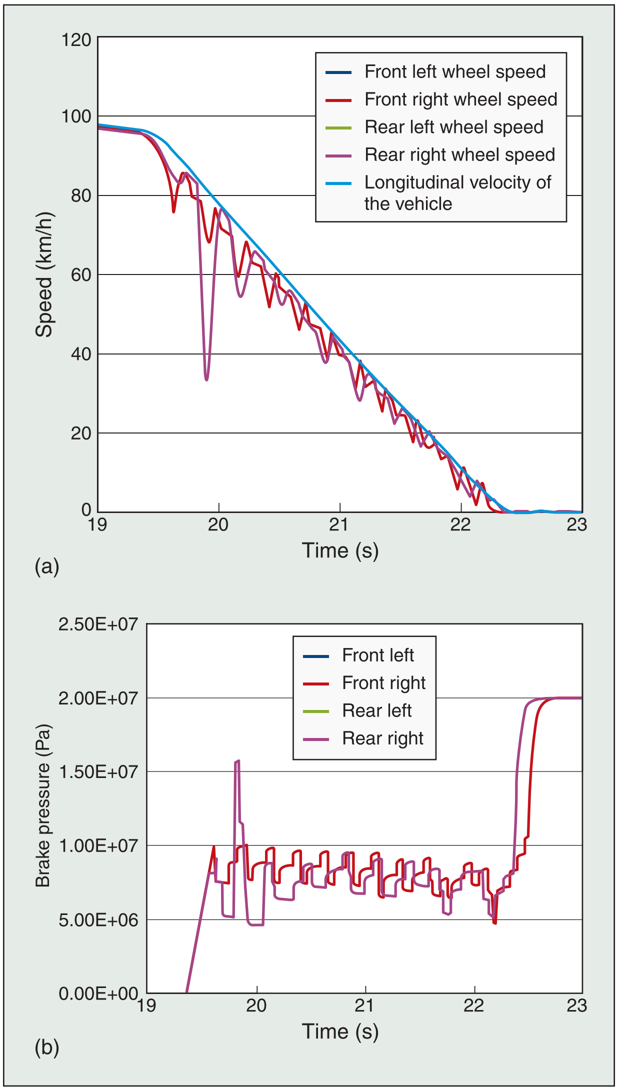

Fig. 7. Vehicle velocity, wheel speeds, and brake pressures of the ABS ECU with the ignition on.

corrupted CAN frames and physical failures such as short circuit, overvoltage, and open circuit on the CAN lines.

During actuation tests, the behavior of the ABS ECU was checked after various braking scenarios, such as emergency braking on different road surfaces and mu-split maneuvers. In the fail-safe tests, the response of the ABS ECU to implausible vehicle dynamics conditions and HCU failures were observed (extreme acceleration, invalid wheel speed sensor values, and unexpected pressure values in the emulated HCU). In the actuation and fail-safe tests, the vehicle velocity, wheel speed, brake pressure, and valve and pump activation signals were tracked and compared to reference ranges.

Numerous tests were executed successfully; in Fig. 6 and Fig. 7, characteristic wheel speed and brake pressure values can be seen, measured during emergency braking from 115 km/h with *ABS ignition off* and *ABS ignition on*, respectively.

For the CAN reliability and fail safe test cases, a fault generation program was implemented on the FPGA based I/O module, extended with a solid-state relay module. The fault generation module facilitated real-time modification of the CAN frames and creation of physical level errors on the CAN lines.

The final step in the process of development was the automation of test sequences under Teststand. Test sequences consisted of the tests previously implemented in Veristand. Teststand can deploy Veristand projects automatically to real-time systems and can generate customizable reports from the results in XML, ATML, and HTML format.

During a six-month long testing period, a great deal of experience was gained. The adaptation of the test scripts of the older Systerra based simulations was smooth. The time needed for developing the new test cases was just half of the development time required for the older environment, thanks to the well-supported hardware components and graphical programming and modeling environments such as LabVIEW and Simulink. The maintenance time of the new system was also less than that for the old one. The old simulation environment is not well supported by the Systerra because of its inherent legacy hardware components, and therefore, the maintenance cost of the system is high. To keep the Systerra based system in operation, three engineers were needed, while in the case of the NI based system one engineer was enough.

Summary

A new, flexible, and easily reconfigurable HIL simulation and test environment was created for function development and test automation of state-of-the-art automotive electronic control units. The implemented hardware and software environment was validated and successfully tested with the integration and verification of a mass-produced ABS ECU. A new pressure model was elaborated for the emulation of the hydraulic control unit (ABS HCU). In the HIL environment communication, ABS actuation and fail-safe tests were implemented and performed successfully; the results were in concordance with the expectations.

As a conclusion, we can state that a powerful reconfigurable National Instruments PXIe based HIL simulation and test environment was designed with respect to hardware and software architecture. The elaborated new system is suitable for long-term hardware maintenance and easy software migration from the old systems, and according to the measurements, it was proved that it is a good alternative to existing automotive simulation environments.

Acknowledgments

We thankfully acknowledge the intensive technical support of Continental Automotive Hungary Ltd. and National Instruments Hungary Ltd.

References

[1] P. E. Rieth, S. A. Drumm, and M. Harnishfeger, *Electronic Stability Program: The Brake That Steers*. Landsberg am Lech, Germany: Verlag Moderne Industrie, 2002.

[2] F. Gustafsson, "Automotive safety systems, replacing costly sensors with software algorithms," *IEEE Signal Processing Mag.*, vol. 26, pp. 32-47, 2009.

[3] R. Rajamani, N. Piyabongkarn, J. Lew, K. Yi and G. Phanomchoeng, "Tire-road friction coefficient estimation, real-time estimation methods for active automotive safety applications," *IEEE Control System Mag.*, vol. 30, pp. 54-69, 2010.

[4] B. A. Guvenc, L. Guvenc and S. Karaman, "Robust yaw stability controller design and hardware-in-the-loop testing for a road vehicle," *IEEE Trans. Vehicular Tech.*, vol. 58, pp. 555-571, 2009.

[5] D. Fodor, K. Enisz, R. Doman, and P. Toth, "Tire road friction coefficient estimation methods comparison based on different vehicle dynamics models," in *Proc. IEEE International Vehicle Power and Propulsion Conference, IEEE VPPC,* 2011.

[6] G. Rill, *Simulation von Kraftfahrzeugen*. Braunschweig/Wiesbaden, Germany: Vieweg, 1994.

Krisztian Enisz (eniszk@almos.uni-pannon.hu) is a PhD student at the University of Pannonia, Faculty of Engineering, Institute of Mechanical Engineering, Department of Automotive Mechatronics, Hungary. His primary fields of interest are hardware in the loop simulation techniques, test automation, and development of automotive electronic control units.

Denes Fodor (fodor@almos.uni-pannon.hu) holds the DSc degree in electrical engineering. Currently he is the Head of School of Automotive System Engineering and Head of the Department of Automotive Mechatronics at the University of Pannonia, Faculty of Engineering, Institute of Mechanical Engineering, Hungary. His research interests are in the field of power electronics, motion control, industrial communication, signal processing, and automotive electronic systems. Dr. Fodor is author of six book chapters and more than 95 scientific publications and serves as a PhD supervisor.

Istvan Szalay is a PhD student at the University of Pannonia, Faculty of Engineering, Institute of Mechanical Engineering, Department of Automotive Mechatronics, Hungary. His main fields of interest are electric machines and sensorless control of electric drives.

Laszlo Kovacs (laszlo.kovacs@epcos.com) is a PhD student at the University of Pannonia, Faculty of Engineering, Institute of Mechanical Engineering, Department of Automotive Mechatronics, Hungary, and has been product development engineer at the EPCOS TDK Ltd. in Hungary since 2005. His current research interest is focused toward DAQ systems and test automation used in development of aluminum electrolyte capacitors for pulse with modulated power electronic converters.

For additional information and technical details, see our sister publication, the *IEEE Transactions on Instrumentation and Measurement*.

D. I. Katzourakis, I. Papaefstathiou, and M. G. Lagoudakis, "An Open-Source Scaled Automobile Platform for Fault-Tolerant Electronic Stability Control, "*IEEE Trans. Instrum. Meas.*, vol. 59, pp. 2303-2314, 2010.