# INNOVATIVE HIL ARCHITECTURE FOR ELECTRIC POWERTRAIN TESTING

Cătalin VASILIU[1], Nicolae VASILE[2]

*Această lucrare prezintă o nouă abordare a simulării de tip HIL, a cărei flexibilitate permite abordarea unei game largi de aplicații. Modele de simulare validate pe echipamente reale sunt conectate cu o versiune scalată a unei transmisii electrice, rezultând o platformă robustă, destinată prototipurilor. Structura modulară a sistemului permite extensia facilă, deschizând calea către o abordare colaborativă a testării industriale în sfera automobilelor electrice.*

*The paper presents a new approach for a hybrid HIL simulator, whose flexibility makes it suitable for a wide area of applications. Real-life validated simulation models are interconnected with a real scaled-down electric powertrain, resulting in a robust virtual prototyping platform. The structure of the system is modular and easily extendable, streamlining the collaborative approaches of HIL simulation in the electric automotive field.*

**Keywords**: Hardware-In-the-Loop simulation, virtual instrumentation

## 1. Introduction

With a history of almost half of century [1][2], HIL has established itself only in the last decade as an indispensable tool for faster and more reliable prototyping. Globalization has pushed the limits of collaboration inside the companies' organizational structures, creating the need for new methods to share knowledge and resources. Traditionally, hardware and software components are developed independently at different geographical sites, being tested togheter only in the final phases of development. HIL has allowed the integration to be done earlier in the development cycle, providing an elegant solution for the above problems. Naturally, the technology evolved at a fast pace since its early incarnations, but it can be reduced to a key component: real-time systems.

This paper is trying to describe a new approach of building high-performance, cost-effective HIL systems, and reducing their complexity in order to be used even by people with basic training. An innovative application to electric vehicles is used by the authors to demonstrate the validity of a new concept that connects high-level software and hardware.

[1] E.E., PhD, Electrical Engineering Faculty, University POLITEHNICA of Bucharest, Romania, e-mail: catalin.vasiliu.g@gmail.com
[2] Prof., PhD, Engineering Faculty, University "Valahia" of Targoviste, Romania

## 2. Real-time systems and HIL testing

Due to the fact that technology evolves with a fast pace, it is hard to precisely define what a real-time system is. There are two characteristics, common to all real-time systems that could shed some light on the subject: a) a RTS is bounded to a process that can strictly evolve at its natural rate; b) a RTS responds to stimuli in an absolutely determined manner. More characteristics of real-time systems can be found in [3], along with some common misconceptions about them.

The core of every real-time system is a computing platform, which is composed of hardware and the supporting software. The software can be further categorized into the operating system and user-defined programs. Depending on the scale, a RTS can contain one or more computing units, hardware, other software than the real-time one, and can even be distributed across different geographical locations.

Common uses for real-time systems are satellites, military equipment, avionics, medical devices (non-invasive scanners), command and control for power plants (mostly nuclear), airbags and even mobile phones. Some of the implementations imply a strict no-error policy while others can tolerate occasional less-than-critical malfunctions.

HIL systems are hybrid by nature, involving components from different domains. The development and testing techniques of individual components has been perfected in time, but their behavior in real-life scenarios cannot be easily determined by synthetic tests. Manufacturers are obliged to rely on requirement sheets as input vectors for testing, often working around intellectual property constraints, unsure that a product can function as expected before the using them in the real system. HIL eliminates most of the problems generated by this paradigm by using accurate models that can be supplied as black boxes by their owners or by using real data as input to tests. Real-time systems assure a good correlation of the virtual models with reality, eliminating the inconsistencies introduced by running standalone simulations of different components and binding them together at the end.

## 3. Objectives and Implementation

The main goal of the research is to build a real-time system capable of testing electric motors with simulated loads that emulate real-life conditions. The load is generated by a simulation model, constructed with the help of AMESim. The simulation model is being run in real-time on a PXI-based machine from National Instruments, while the controller is on the same or another identical machine and is assembled in LabVIEW. Two identical electrical brushless motors are connected by an axle, providing a motor-load configuration. The motors are

vector-controlled by an adequate system, providing load curves suitable for the chosen set of car parameters. The proposed HIL architecture is shown in Fig. 1.

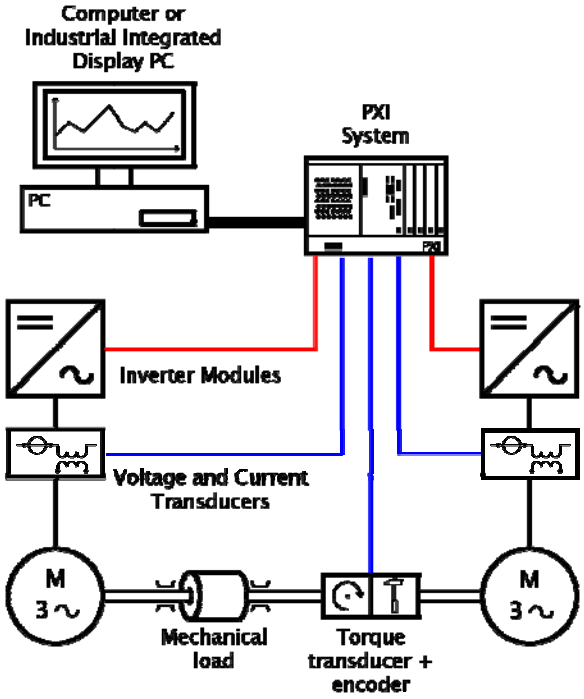


Fig. 1. The original HIL architecture

AMESim is a simulation environment, which, as many others, offers an advanced user interface for building representations of engineering systems. The underlying philosophy is based on bond graphs which imply the use of engineering units for the links between sub models, approach that makes the assembly and validation of complex systems a lot easier [4]. The capabilities of this software package include the modeling of large multiphysics systems, for example the injection systems of combustion engines, hydraulic or electrical networks, entire vehicle dynamics, etc., as well as exporting the models to a different software platform in order to create a hybrid system. A real-time capable model can be created by using a special interface with LabVIEW, among others.

LabVIEW is one of the most used platform for virtual instrumentation, being able to integrate high level programming with low-level DAQ hardware drivers. Using its ability to import user-defined mathematical models, hardware components can be interconnected with them easily, leading the way to a system that can be used for either Model-in-the-Loop, Software in the Loop, or HIL. The mathematical models are encapsulated into .dll files, so no direct information

about its internal structure would be revealed to third parties, offering an acceptable degree of intellectual property protection.

There are a few describing similar implementations, but their focus was either on developing automotive controllers (ECUs) either on testing internal combustion engines and their subsystems. For example, Alles [5] has a similar system, but fewer complexes. In a second paper [6], an improved version of the same setup had been presented. The need for a real time system is reinforced by the research done by Apsley [7], the lack of such system providing sub-optimal performance.

The virtual load is based on a vehicle dynamics model, simulating a car chassis with 15 DoF, driving on a sloped road in the presence of air currents, with adjustable parameters (see Fig. 2). As noted by many authors [8], such a model has many subcomponents and notably stiff elements. Due to the internal constructs of AMESim, it is impossible to decouple slow parts from faster ones in order to run a multi-rate integration scheme.

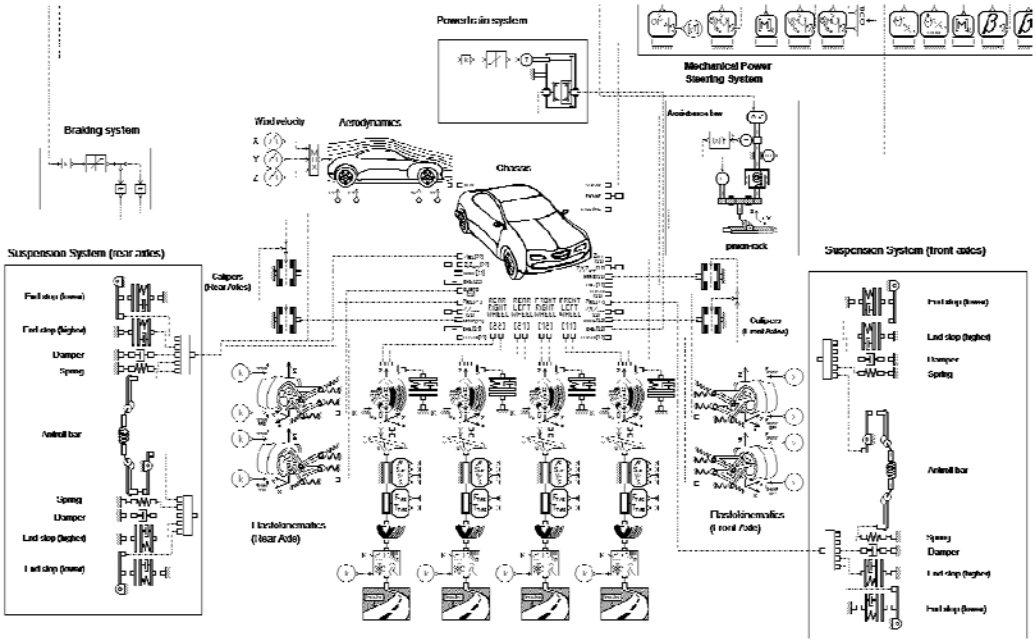


Fig. 2. Vehicle dynamics model used on the test bench – overview

Starting from an existing simulation model designed to be run on a standard platform, some elements had to be adapted for running in real-time, by carefully selecting the proper mathematical formulation, parameters and sometimes by modifying the implicit models. In addition, a modal analysis has

been performed in order to find out the required value of the integration time, which was found to be around 1ms.

From this model, the existing mechanical torque source (the combustion engine) was removed and replaced with a signal coming from LabVIEW. Since the rotational mechanical connection in AMESim is composed of torque and an angular velocity signals (as defined in bond graph theory), the velocity of the shaft to which the real motor would connect to, computed during simulation, is sent back to LabVIEW. The linear speed of the car, also derived during simulation, and other auxiliary variables are routed the same way.

The real-time system tries to emulate normal driving behavior: when the driver accelerates in order to increase or decrease the speed, the torque of the motor is varied. When the desired speed is reached, the driver keeps a constant reference at the acceleration pedal, thus keeping the car running at constant speed. If an obstacle like a different slope or a road bump is encountered, the system accelerates to overcome the forces opposing movement.

To replace the human driver, a PID controller is used to regulate the speed of the car. A predefined profile is generated (e.g. the European Driving Cycle), fed as a reference to the PID controller, and the output used as a torque reference on one of the inverters. Until that point, the system is identical to the ones that are in use today on electric or hybrid vehicles.

The torque at the axle is measured with a dedicated sensor and sent back into the AMESim model, accelerating the car. At the end of a successful integration of the model for a time step, the car and axle speed can be found between the results, while taking into account existing perturbations. The car's speed is sent to the PID controller as the measured process variable, while the axle speed (simulated) is fed to the second inverter as reference. In this way, both motors, as well as the axle, are being subjected to the same static and dynamic speed and torque variations that would be generated in a real vehicle.

The system is equipped with sensors measuring the speed, torque, voltage and current. Taking advantage of the flexibility of the DAQ boards, we can have all the sensors connected to the same board, while generating two output signals, or they can be connected to different boards running at different rates. All the signals carrying information are analog: this assures operation under a strict determinism. The data that is being collected throughout the components can be displayed, stored or both. A GUI can be created to interface the critical signals, while the raw data is saved during operation in a local database, in a streaming fashion. Given the large size of this database, opening and processing its contents could be problematic on slower computers, so it is designed to be portable so it can be analyzed on different workstations. The database solution is the best tradeoff between size and structure.

## 4. Remote data communication

Real-Time PXI systems are headless machines, so they require a remote computer acting as a host to display the graphical user interface, if one is required. The communication is done by standard wired Ethernet networks that can either be using a predefined protocol or a custom user implementation based on UDP or TCP/IP protocols.

LabVIEW can use a special, proprietary, data transfer protocol called NI-PSP, with a transport mechanism called LogosXT. Since version 8.5, the protocol has been switched from UDP to TCP/IP, so LogosXT is designed to minimize the large overhead that this protocol generates. In order to have stable communication, the measured/computed data is custom packed prior to sending it to the host machine; otherwise, very bad performance was observed. Without this step, data rates around 2 – 4 Mbit/s would generally bottleneck the network, keeping CPU usage high on older machines. By packing the data and using the custom algorithm, while taking advantage of the LogosXT stability and security, the data rate was pinned down to less than 1 Mbit/s (150kByte/s). With the above mentioned data rate, 16 channels sampled at 1kS/sec could be recorded simultaneously, and enough bandwidth was saved for future enhancements.

## 5. Experimental results

A preliminary test sequence was used to spot potential problems. To test the acceleration, deceleration and speed regulation of the system, a trapezoidal speed profile with two flat sectors with different amplitudes was generated. No dynamic perturbation had been inserted in the test sequence at this moment.

From Fig. 3 it can be seen that the car follows closely the profile that has been imposed. We can conclude that the PID controller for the automobile's speed is well tuned, therefore the closed hardware-software loop works and is performing as expected. By using a suitable driver model to take into account the delays in response of a human operator, an automatic testing procedure can be set up.

When the angular speed of the electrical motor is imposed, we assume that it is instantly reached. This is a natural assumption, given the fact that modern motor controllers are very advanced and closed loop feedback was used, while the reference was under the nominal values. The Fig. 4 shows the variation of the angular velocity during the test.

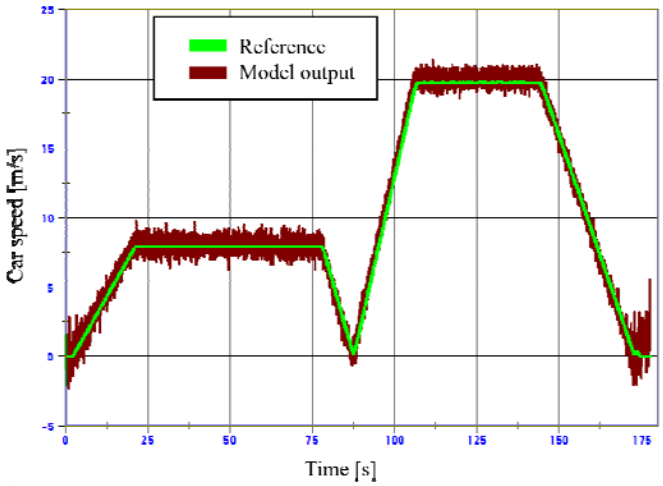


Fig. 3. Car speed regulation

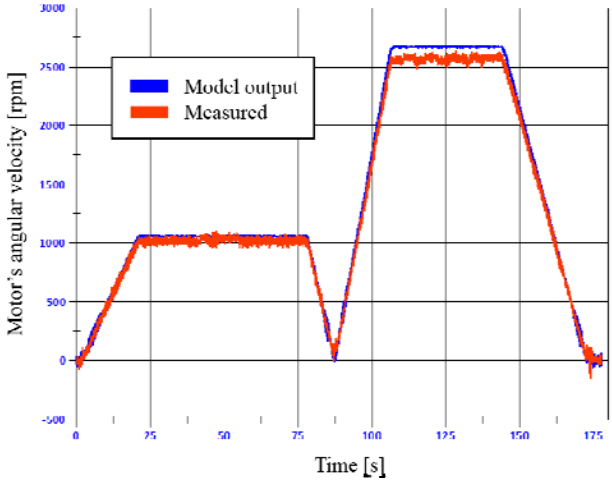


Fig. 4. Motors' angular velocity

A slight difference between the measured and imposed values can be observed when it comes to the torque. A few possible causes have been identified and will be addressed: incorrect calibration – the analogue outputs of the inverters have a smaller resolution than the DAQ board, and since the signal passes through multiple conversions, some coefficients were misaligned; the same is true for the inputs; PID tuning errors; drift of parameters due to temperature.

The torque profile that is required to drive the car, as measured by the acquisition board, is shown in Fig. 5. (10V correspond to 100% of the motor's maximum torque). The motors can be used in all 4 quadrants, as it can be observed from the chart. The car speed is always positive while the car is going forward, while the negative torque indicates the electrical motor is working as a generator (the measured torque has its real sign inverted).

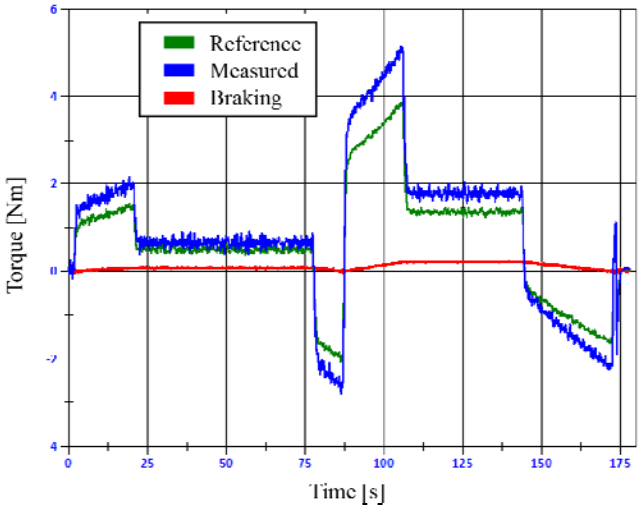


Fig. 5. Motors' torque

In order to characterize an electrical vehicle, we need to have accurate information about all the electrical variables in the system. Therefore, current and voltage sensors were mounted on each phase of the motors. By collecting their corresponding waveforms (see Fig. 6), the power consumption can be derived, as well as having a better controller strategy defined. It is known that modern electric cars use different controller profiles to minimize consumption or to provide sportier behavior, and by gathering data both from the motor and from the car chassis, a suitable controller can be imagined, tested on the real-time system and implemented in real-life.

## 6. Conclusions

Preliminary results obtained so far indicate that the current architecture is well designed but there are a number of things to be improved. Generally, the system behaved as intended, with minor problems left to be solved in the second iteration. The working hardware and software loop proved that the architecture is viable and that it can be used successfully for HIL tests.

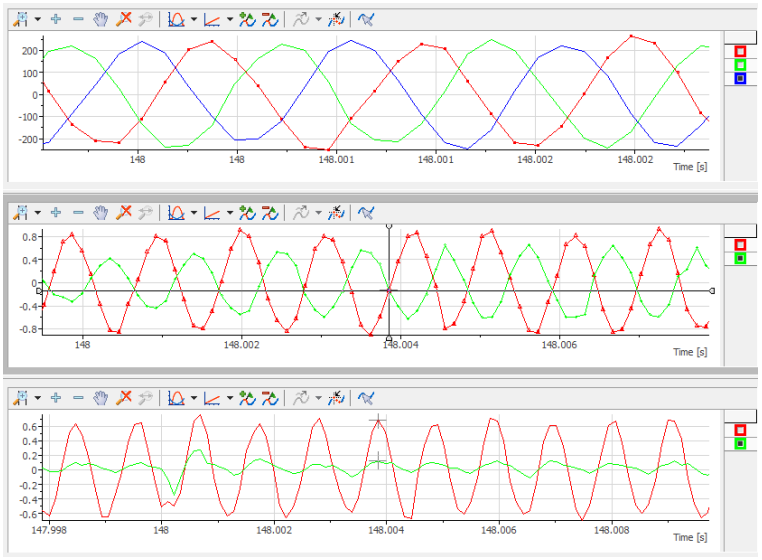


Fig. 6. Measured voltages and currents

The diversified nature of real-time applications makes modeling with reusable units difficult. What sets the current approach apart from similar endeavors is its ability to adapt to different use scenarios. It can be modified for similar applications without major interventions. Because there is no digital communication between hardware components, the motors and inverters can be easily replaced, by adjusting just a few scaling parameters.

The advantages of using a bond-graph modeling approach as opposed to the traditional signal-based are numerous, especially when dealing with realistic physical entities. Lebrun and Claude [4] have written a basic paper on the subject. The bond graph paradigm allowed a natural interconnection between a virtual model and a real motor.

Compared to other similar systems, a large scale vehicle dynamics model was successfully used for real-time purposes without employing special techniques or mathematical formulations. There was no need to write code or to set up obscure parameters of the real-time platform as with other implementations. This would translate in the product development cycle to a faster prototyping time.

Even if the real-time simulation evolved from its early days, making it easier for the users to build complex systems, it still relies much on local optimization, which is done “manually” by the specialized engineers. Our approach has provided a faster and more reliable method for HIL testing, while providing a glimpse of the performance attainable with the current state-of-art.

**Acknowledgement**

The work has been funded by the Sectorial Operational Programme Human Resources Development 2007-2013 of the Romanian Ministry of Labor, Family and Social Protection through the Financial Agreement POSDRU/6/1.5/S/19.

## R E F E R E N C E S

[1]. C. *LIN, L. Zhang,* "Hardware-in-the-loop Simulation and Its Application in Electric Vehicle Development", in IEEE Vehicle Power and Propulsion Conference, September 2008.

[2]. *H. Hanselmann*, "Hardware-in-the-Loop Simulation Testing and its Integration into a CACSD Toolset" in IEEE International Symposium on Computer Aided Control System Design, September 1996, pp. 152–156.

[3]. *J. Stankovic,* "Misconceptions About Real-Time Computing: A Serious Problem For Next Generation Systems". Computer, October 1998, pp. 10-19.

[4]. *M. Lebrun, R. Claude*, "How to create good models without writing a single line of code", in 5th Scandinavian International Conference on Fluid Power, May 1997, pp-.

[5]. *S. Alles, C. Swick, S. Mahmud, F. Lin,* "Real-time Hardware in the loop Vehicle Simulation", in Instrumentation and Measurement Technology Conference, May 1992, pp. 159 – 164.

[6]. S. *Alles, C.A. Swick, M.E. Hoffman, S.M. Mahmud, F. Lin,* "The Hardware Design of a Real-Time HITL For Traction Assist Simulation" in IEEE Transactions on Vehicular Technology, **Vol. 3** (44), August 1995, pp. 668-682.

[7]. *J.M. Apsley, E. Varrone, N. Schofield,* "Hardware-in-the-loop evaluation of electric vehicle drives", in 5th IET International Conference on Power Electronics, Machines and Drives, April 2010, pp.188-193.

[8]. *M. Bacic*, "On hardware-in-the-loop simulation", in Proceedings of the 44th IEEE Conference on Decision and Control, and the European Control Conference, December 2005, pp. 3194-3198.