Design of an Educational Emulation Framework for Mechatronics Control Unit Development

Tamás Bécsi
Department of Control for Transportation and Vehicle Systems
Budapest University of Technology and Economics
Budapest, Hungary
Email: becsi.tamas@mail.bme.hu

Szilárd Aradi
Department of Control for Transportation and Vehicle Systems
Budapest University of Technology and Economics
Budapest, Hungary
Email: aradi.szilard@mail.bme.hu

Péter Gáspár
Systems and Control Laboratory
Computer and Automation Research Institute
Hungarian Academy of Sciences
Budapest, Hungary
Email: gaspar.peter@sztaki.mta.hu

***Abstract*—Teaching mechatronics is a great challenge since the subject is a multidisciplinary area of electrical, mechanical and IT knowledge. Students must deepen their knowledge in these slightly related fields of engineering. Tasks and solving problems in mechatronics require cognitive and operational knowledge and practical experience about systems design and analysis. Since mechatronics engineering is a fast-developing and practice-oriented field placing more emphasis practical and project-based education at universities is strongly encouraged by the industry. The paper presents an educational framework designed for students with a specialization in vehicle mechatronics. The aim of the system is to provide the emulation of a real vehicle module, which must be operated by appropriately designed controller unit.**

I. Introduction and Motivation

Except for the recession in 2008 worldwide vehicle market has been growing steadily. According to the figures of The International Organization of Motor Vehicle Manufacturers (OICA) the vehicle industry requires the employment of more than eight million people directly in making the vehicles and the parts that go into them. This accounts for over five percent of the world's total manufacturing employment [1].

Competition in the industry motivates the manufacturers to introduce new technologies, comfort and safety solutions resulting in the high employment level of their research and development departments. Therefore there is a strong demand for well-trained vehicle engineers on the labor market. These engineers need various knowledge in the field of electronics, mechanics, hardware and software design, and also in industry-related technologies, such as standardised intra-vehicle communication. Understanding this needs the Department of Control for Transportation and Vehicle Systems Budapest University of Technology and Economics introduced a specialization programme covering most of these fields.

Teaching mechatronics is a great challenge since the subject is a multidisciplinary area of electrical, mechanical and IT knowledge. Students must deepen their knowledge in these slightly related fields of engineering. Tasks and solving problems in mechatronics require cognitive and operational knowledge and practical experience about systems design and analysis [2]. All mechatronics-related curricular papers state

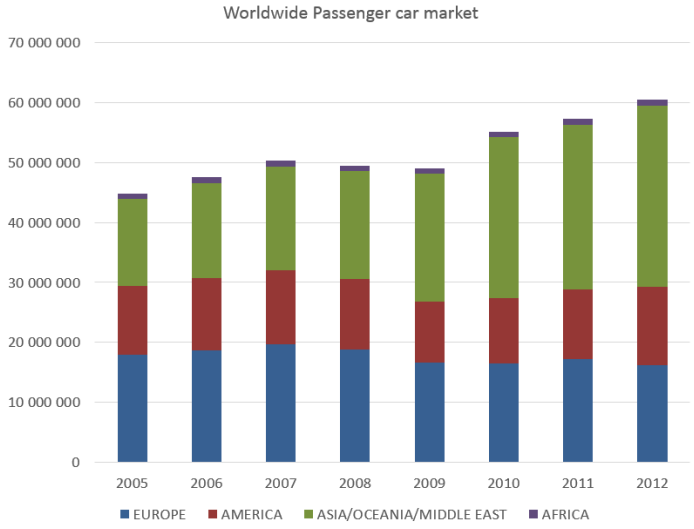

Fig. 1. Worldwide Passenger car sales (source:oica.net)

that besides the formal teaching of basic techniques, such as sensors and actuators, software and hardware design or dynamics, certain kind of practical material is needed in this field [3] [4] [5]. There is a clear consensus amongst the participants of education and industry about the strong need of project-based teaching in the education of mechatronic systems [6] [7] [8]. Although methodologies may differ a university student project only simulates the real industrial development conditions. However, giving individual or group assignments to students with a related problem to solve is the best practice to acquire skills in this field. Didactic questions, such as what kind of hierarchical or development structure should they use, emerge during the organization of the students into groups. Grimheden [9] proposed the agile method while Scrum [10] proposed other curricula classic development methods such as the V-model, see also [11] [12].

The educational framework presented in this paper serves similar purposes. The system aids the students throughout a complex, yet simplified functional development of an electronic control unit (ECU) of a vehicle. As an individual project

a specification of an ECU is given to the students defining the functional description of the driven vehicle component the details of the analogue or digital inputs and outputs of its sensors and actuators, and also the communication needs of the ECU including the status and command messages that must be sent or processed. A student's task is to analyse the problem and develop the functionalities described in the specification. For testing purposes the student developers need units for testing the controller's functionality. In some cases it could be sufficient to provide the suitable vehicle parts, but in other cases it is not sufficient. Moreover the testing of erroneous behaviours is much simpler in an emulation environment.

These are the main reasons why the decision to develop a complex framework providing all functionalities of the modules to be driven has been made. The paper presents the hardware and software-related solutions of this emulation environment.

II. Student tasks

As previously mentioned, the aim of the systems is to provide an environment for the development tasks students must fulfil. To cover the real-world development process as much as possible several aspects have been taken into consideration. The areas and skills covered by the student project are:

- Understanding and analysing functional specifications;
- Knowledge of standard vehicular network description formats, such as *DBC* or *LDF*;
- CAN/LIN/FlexRay communication via microcontroller;
- Software testing;
- Failure mode and effect analysis and error handling;
- Design of test cases;
- Documentation.

Each student task represents a control unit of a vehicle with different tasks. Their common feature is the CAN communication, and each of them works together with an other unit by providing information, or receiving commands. The ECU functionalities involved in the tasks are the followings:

1) Electric Window Lift (EWL): The EWL module must control an electric window of a vehicle with two direction inputs and a proper PWM signal. With a short (less than half a second) input it must drive the window to the end position, otherwise it drives the window until the input button is pressed. Failures may occur when the limit switches are malfunctioning: the window becomes stuck or the drive motor fails.
2) Data Acquisition Unit (DAU)
3) Trip Computer Unit (TCU): The TCU is a simple display module for the driver, which collects information from other ECUs via CAN and calculates average speed and trip distance on a daily and full lifetime basis. The display shows warning on opened doors.
4) Blindspot Assistant (BSA): The BSA must implement simple functionalities of a blindspot assistant. The ECU gives warning based on the measured side distances of objects, the steering wheel angle, turn signal state and vehicle speed. Some information is gathered through the I/Os, and other is received via CAN communication. Based on the severity level, the ECU actuates warning LEDs, or shakes the steering wheel.
5) Turn Signal (TS): The TS unit drives four turn signal lamps of the vehicle with the information provided by their control switches and also the switch of the hazard warning. The ECU also considers the angle of the steering wheel, which is provided as a simple analogue input. The system must monitor the state of the lamps and malfunctions can be emulated in the framework.
6) Headlight Control (HLC): The HLC implements the functionalities of a headlight control unit, where the standard low or full-beam functions are driven by digital IOs, although cornering light is controlled by information acquired via the CAN bus. For error handling purposes the measurement of lamp states are given as a feedback.
7) Central Lock (CL): The CL unit controls the central locking system of the vehicle based on the state of the doors and the control inputs.
8) Electronic Cruise Control (ECC): The ECC module implements the Electric Cruise Control function, with the driver inputs of on/off and acceleration/coast. It also uses the information of the wheel speed sensor. The ECC gives the throttle command via CAN, and the emulation framework simulates simple vehicle dynamics and external forces such as wind or inclinations.

III. The Educational Framework

A. Overview

Several subsystems must be designed to emulate the working environment of an ECU. The subsystems and their relations in the framework are presented on Fig.2.

First the behaviour emulation of the driven mechanical part must be developed. The dynamics, mechanical or electrical constraints all have to be taken into consideration with the capability of error injection. These functionalities differ in each task, with different complexities.

To ensure connection to the environment the simulation of actuators and sensors are necessary. Actuator and sensor dynamics must be separated from the base behaviour model. An other useful feature of this solution is that erroneous state simulation can be modelled this way.

The integration of the CAN communication into the framework serves two purposes. Some tasks has their main functionalities operated via CAN while others only use the vehicular network for sending state messages. However, the entire system only operates without emulated CAN messages when all tasks are finished by all students. Therefore in the phase of individual development the students need the emulation of the communication of the ECUs designed by the others.

The capabilities of error insertion widen the possibilities of the projects, since in some cases the simple implementation

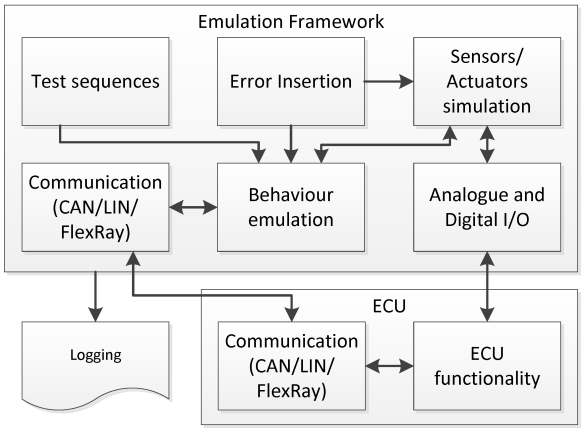

Fig. 2. Framework overview

of the functionality is not a great challenge but when the ECU must be prepared to detect, log or even handle errors, the simplest objective can be easily turned into a much more complex problem. Moreover this option opens a new subtask for the students on a related field where the failure modes, their effects must be evaluated and an effective action plan must be designed with states of normal, degraded, or erroneous operation.

To interact and generate system states, each behaviour simulation module provides controls for changes in the environmental conditions or the operability of modules. In most cases it is sufficient to manually operate the driven parts emulation but sometimes for a more complex environment simulation test sequences are needed, which can be previously configured. These previously built test sequences offer the possibility for the students to refine ECU operation repeatedly during the development process. Parallel with the automatic test sequences automatic logging feature is also developed for similar reasons. On the one hand it aims at the development process on the other hand it is useful for the final evaluation of the project.

B. The educational ECU network

As previously mentioned the designed ECUs all have the requirement to communicate on a CAN network. There are two basic types of messages that must be handled by the units. The first is the state message, in which all measurable signals are periodically sent along with the error states detected by the software. The other type of messages are commands when one ECU instructs the other to do something. In our imaginary vehicle whose functions are implemented by the individual devices all units are connected to one common CAN bus with an additional supervisor device as shown on Fig.3.

The reason for connecting all units to one communication bus is the same as in any commercial vehicle, to ensure the cross communication between all units. As an example, when the CL unit receives the "*lock doors*" command from its interfaces, it sends a command to the EWL unit to close all

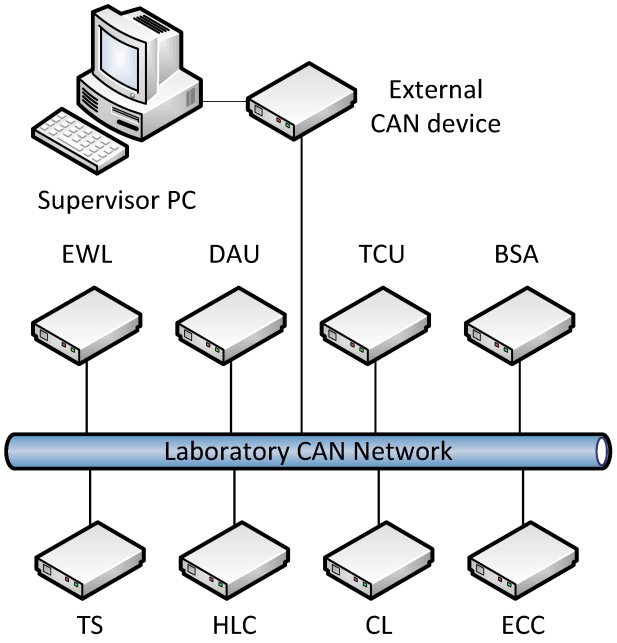

Fig. 3. Laboratory CAN Network

windows and a command to the TS unit to blink two with all turn signals. If the door locking operation can not be performed it sends a one-blink command to the TS. Table I shows the cross-interaction chart of the ECUs.

TABLE I. COMMUNICATION CHART OF THE LABORATORY NETWORK (RST: READ STATUS; CMD: GET COMMAND)

	BSA	CL	DAU	ECC	EWL	HLC	TCU	TS
BSA	X			RST		RST		RST
CL		X						
DAU			X				CMD	
ECC				X				
EWL		CMD			X			
HLC				RST		X		
TCU		RST	RST	RST			X	
TS		CMD						X

The CAN Signals and messages are defined in a dbc file which is a quasi-standard description database format for CAN communication. The dbc is given to the students along with the specification.

IV. HARDWARE FRAMEWORK AND IMPLEMENTATION

Since the controller education at the department is based on the Atmel AVR family the ECU development tasks also require this platform. Atmel Studio is the integrated development platform for developing and debugging Atmel ARM Cortex-M and Atmel AVR microcontroller-based applications.

The students are provided with a BIGAVR6 development board with AT90CAN128 microcontroller to fulfil the task though the framework could operate with any other controller family. BIGAVR6 (see Fig.4.) is a full-feature development board for Atmel AVR microcontrollers. [13] The board has

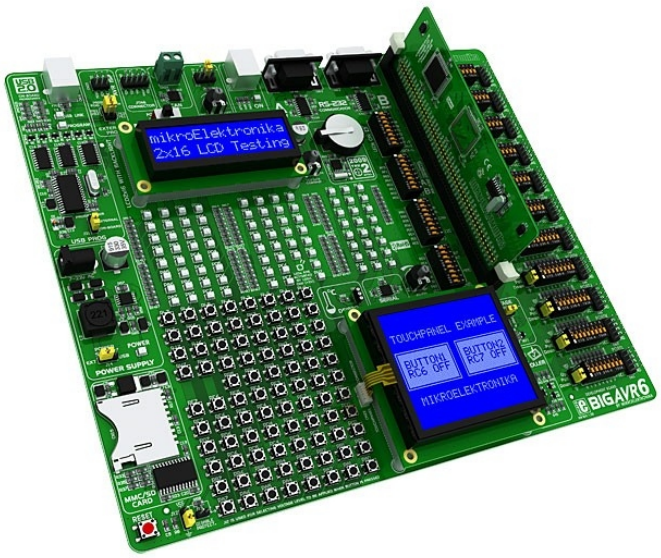

Fig. 4. BigAVR development board

all necessary digital and analogue I/Os, CAN tranciever, and displays, with on-board programmer. AT90CAN128 [14] is a high-performance, low-power Atmel 8-bit AVR RISC-based microcontroller with 128 KB ISP flash memory, 4KB EEPROM, 4KB SRAM, 53 general purpose I/O lines, 32 general purpose working registers and V2.0A/V2.0B compliant CAN controller.

For communication purposes WeCAN USB is used which is a general purpose CAN interface unit with USB connectivity. [15]

Fig. 5. National Instruments MyDAQ

The analogue and digital I/O interface of the framework is realized with NI myDAQ (see Fig. 5.) which is a low-cost portable data acquisition (DAQ) device, which works together with NI LabVIEW-based softwares, allowing students to measure and analyze real-world signals. NI MyDAQ has two analogue inputs and two outputs and also 8 digital iO channels. Combined with NI LabVIEW on the PC, students can analyze and process acquired signals and control simple processes.[16]

A. *Example: the EWL unit*

As a simple example, the EWL units emulation is presented in the current section. The EWL units module emulates the functionalities of an electric window unit. The window is lifted by a motor drive that needs two signals to drive, a digital input for direction, and a PWM signal as speed command. Two switches provide the command of the passenger, and two limit switches sense the end position of the window with a redundant analogue window position measurement signal. The current of the electric motor is provided as a feedback. Erroneous states can occur in all sensors, the limit switches, current and position measurement may fail. Simultaneously, short circuit can occur in the motor and the window may become stuck resulting in an increase in the motor current. These failures all must be examined by the students and it is proposed to reveal the redundancies and the possible substitutions of the sensor function by combining all acquired information. The example screenshot of the EWL units emulation software is shown in Fig. 6.

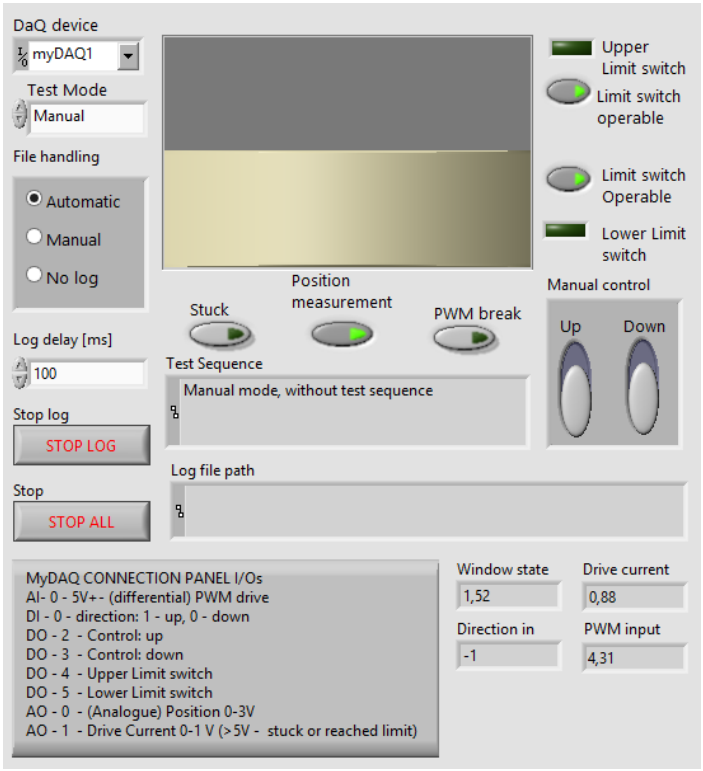

Fig. 6. Example screenshot of the EWL emulator

The dbc describing the CAN communication identifies two messages that must be handled by the EWL ECU.The first is the state message ”MSG_ST_EWL”, where the state of the sensors, window position, operability and motor current are transferred. The message ”MSG_CMD_EWL” carries command from other units that tell the ECU to lift the window to a specific position.

The formal description of the messages are the following:

BO_ 2149187380 MSG_ST_EWL: 8 EWL_ECU
SG_ St_Wnd_Sw_Up : 27—1@1+ (1,0) [0—0]
SG_ St_Wnd_Sw_Down : 26—1@1+ (1,0) [0—0]
SG_ St_Wnd_EP_Low : 25—1@1+ (1,0) [0—0]
SG_ St_Wnd_EP_High : 24—1@1+ (1,0) [0—0]
SG_ St_Wnd_Position : 16—8@1+ (0.4,0) [0—102] "%"
SG_ St_Wnd_Operability : 8—8@1+ (1,0) [0—0]
SG_ St_Wnd_Current : 0—8@1+ (0.04,0) [0—0] "A"

BO_ 2148138804 MSG_CMD_EWL: 8 CL_ECU
SG_ Cmd_Wnd_Pos : 0—8@1+ (0.4,0) [0—100]

V. Conclusions and future expansion possibilities

Working on connected yet individual tasks has both competitive and cooperative aspects. The students working on the same tasks exchange ideas a compare their results during the design process. The individuals working on ECUs that must work together spend hours testing the interaction and communication of their units. The framework inherently encourages team work since at the end of the semester all units are connected together to perform joint operation just as they would do in a real vehicular environment. Naturally a high amount of educational effort is needed in this kind of development since continuous consultations with the developer students is inevitable.

Universities need to provide the industry with engineers for the interdisciplinary field of mechatronics. Specially in the field of vehicle electronics and mechatronics since it is a mix of electrical, mechanical and software engineering

Acknowledgment

The research has been conducted as part of the project TÁMOP-4.2.2.A-11/1/KONV-2012-0012: Basic research for the development of hybrid and electric vehicles. The Project is supported by the Hungarian Government and co-financed by the European Social Fund.

References

[1] "Organisation internationale des constructeurs d'automobiles (oica)," online:www.oica.net.

[2] H.-H. Erbe and F. W. Bruns, "Didactical aspects of mechatronics education," in *5th IFAC International Symposium on Intelligent Components and Instruments for Control Applications (SISACA). Aveiro-Portugal*, 2003.

[3] Y. Wang, B. Schwarz, Y. Pan, B. Chen, H. Wiedmann, C. Xie, H. Wang, Y. Yu, and X. Feng, "Mechatronics education at {CDHAW} of tongji university: Structure, orientation and curriculum," *Mechatronics*, vol. 18, no. 3, pp. 172 – 177, 2008.

[4] S. Meek, S. Field, and S. Devasia, "Mechatronics education in the department of mechanical engineering at the university of utah," *Mechatronics*, vol. 13, no. 1, pp. 1 – 11, 2003.

[5] J. B. Hargrove, "Curriculum, equipment and student project outcomes for mechatronics education in the core mechanical engineering program at kettering university," *Mechatronics*, vol. 12, no. 2, pp. 343 – 356, 2002, mechatronics Education in Europe and the United States.

[6] K. Tan, K. Tang, and T. Lee, "Foreword for special issue on design-centric mechatronics education," *Mechatronics*, vol. 23, no. 8, pp. 917 –, 2013.

[7] Y. Wang, Y. Yu, H. Wiedmann, N. Xie, C. Xie, W. Jiang, and X. Feng, "Project based learning in mechatronics education in close collaboration with industrial: Methodologies, examples and experiences," *Mechatronics*, vol. 22, no. 6, pp. 862 – 869, 2012, special Issue on Intelligent Mechatronics (LSMS2010, ICSEE2010).

[8] Y. Wang, Y. Yu, C. Xie, X. Zhang, and W. Jiang, "A proposed approach to mechatronics design education: Integrating design methodology, simulation with projects," *Mechatronics*, vol. 23, no. 8, pp. 942 – 948, 2013.

[9] M. E. Grimheden, "Can agile methods enhance mechatronics design education?" *Mechatronics*, vol. 23, no. 8, pp. 967 – 973, 2013.

[10] L. Rising and N. Janoff, "The scrum software development process for small teams," *Software, IEEE*, vol. 17, no. 4, pp. 26–32, Jul 2000.

[11] H. Komoto and T. Tomiyama, "A system architecting tool for mechatronic systems design," *CIRP Annals-Manufacturing Technology*, vol. 59, no. 1, pp. 171–174, 2010.

[12] K. Forsberg and H. Moozhor, "The relationship of system engineering to the project cycle," in *The 12th INTERNET World Congress on Project Management*.

[13] *BIGAVR User's Manual*, Mikroelektronika, 2006.

[14] *AT90CAN32/64/128 Complete Datasheet*, Atmel Corporation, 2008.

[15] *WECAN USB PC ADAPTER*, Inventure Automotive Electronics.

[16] *NI myDAQ User Guide and Specifications*, National Instruments Corporation, 2011.