Chapter

Using of Virtual Instrumentation in the Teaching of Autotronics

Matúš Danko, Ondrej Hock, Jozef Šedo and Branislav Hanko

Abstract

This chapter presents using of virtual instrumentation for improvement of teaching and demonstration of communication of one of the most popular automotive buses CAN BUS. Virtual instrumentation is used in several ways for easier understanding of communication by using the automotive bus. The first way is a demonstration by visualization of data of programable gateway. The second way of using virtual instrumentation is the visualization of the instrument cluster which is controlled by the student's development kit with the microprocessor. The visualization is also used for the demonstrational panel which consists of a development kit, instrument cluster, turbocharger, and accelerator pedal. The communication with these components is also visualized. The last usage of virtual instrumentation is controlling real instrument clusters from Skoda and VW vehicles using LabVIEW and CAN BUS interfaces. In both, controlling and visualizing of instrument cluster are used the same messages as in real vehicle, so it is easier for students to understand the structure of communication through automotive buses of the real vehicle.

Keywords: virtual instrumentation, LabVIEW, CAN BUS, automotive bus, autotronics

1. Introduction

The faculty of Electrical Engineering and Information Technology at the University of Žilina (UNIZA) is one of the oldest parts of the entire university (since the university's origin in 1953). For more than a half of century, this faculty is following novel trends in electrical engineering, electronics, industry, and science.

Last decade, several car manufacturers or manufacturers of car equipment and devices (e.g. KIA, Land Rover, Volkswagen, and Brose) established their factories and also R&D departments in the region of Žilina (northwest part of Slovakia). At this time, we can see decreased demand from mentioned companies for employees in this field. Department of Mechatronics and Electronics (DME) at the University of Žilina immediately started a new study program "Autotronics" to bring the specialists focused on car parts and sensors functionality, electrical devices in vehicles, supplying systems in hybrid or electrical vehicles, programming the car control units, and understanding the auxiliary infrastructure and economic aspects of car manufacturing and transportation.

As we can see in **Figure 1**, the interdisciplinary character of the entire Department of Mechatronics and Electronics is fully integrated into the Autotronics program.

Following the modern trends in education, adapting to the demands of manufacturers and the industrial environment, and also finding the educational methods during the specific situation caused by COVID-19, we would like to introduce the educational model for selected autotronics courses provided by our department DME. As we will see in the next sections, the model composed of various parts is joined into the functional complex system through the development system LabVIEW from the NI company. Even though LabVIEW was dedicated to creating virtual instruments in a practical environment, it was attractive for education from 1980s as documented in [1]. The studies [2, 3] focus on the advantages of virtual instrumentation in LabVIEW in the engineering education process and the creation of a remote laboratory. NI responded to this trend (usage of LabVIEW in the teaching process) and now we can find many articles, forums, and recommendations for teachers and students directly on NI websites [4]. Researching the internet sources, we can find several complex systems for the education of automotive programs and courses [5, 6]. But these simulators are the systems which contain only real parts of vehicles, so customization is not possible. Also, these simulators are not delivered with visualization software. In [7], the vehicle CAN BUS simulator that is programmed in C language is presented. In [8, 9], systems where CAN BUS interface is controlled and visualized by the original software of the interface or in combination with MATLAB are presented. Thus, these systems are not fully open systems, and customization is more difficult than customization of the LabVIEW application which is used for control and visualization at the same time. This advantage of easy customization led to the cooperation of teachers and students in development: the results of selected bachelor or diploma theses can be integrated to extend the existing model.

One of the most important parts of modern vehicles is the communication between many different systems used in vehicles [7, 8]. There are several automotive buses that can be divided by transmitting speed, the medium and maximum length of the bus, or devices connected to the bus. Real-time control of the internal

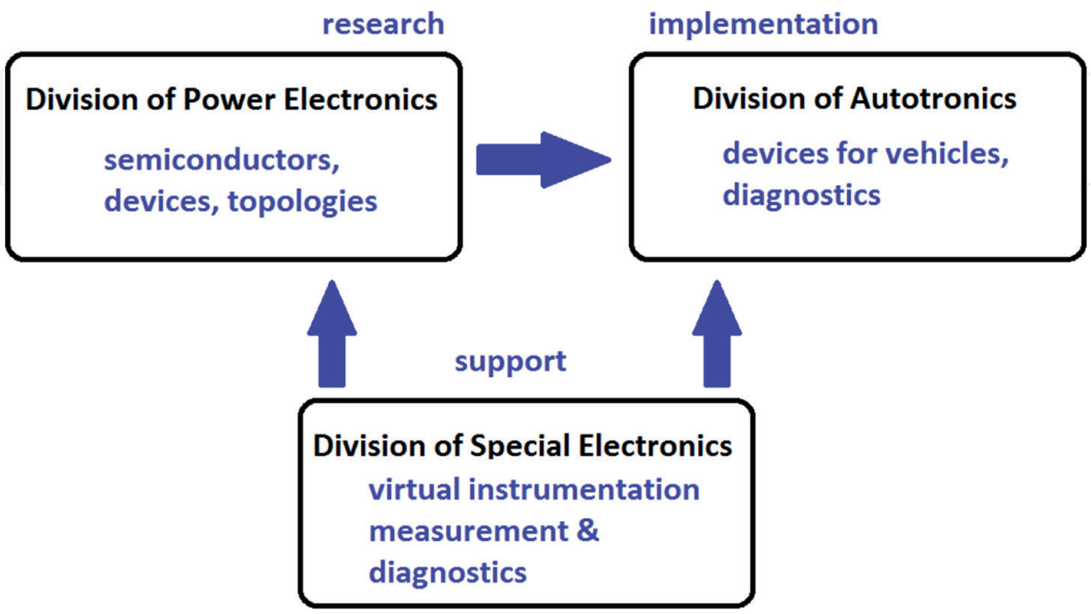

research
implementation
Division of Power Electronics
semiconductors, devices, topologies
Division of Autotronics
devices for vehicles, diagnostics
support
Division of Special Electronics
virtual instrumentation
measurement & diagnostics

Figure 1.
Mutual cooperation between divisions of DME.

combustion engine (ICE) of the converter of the electric drive is needed high-speed communication with high reliability. For a less important system like comfort systems, the speed of the bus can be slower, and the transmission media can be simplified. Based on this division, each automotive bus is suitable for the connection of a specific group of devices. Because of its properties, the CAN BUS is used in various systems which make it one of the most common and well-known automotive buses. For purpose of teaching how this automotive bus works, the workbenches of the laboratory were modified. Every workbench contains a connector for connecting students' evaluation boards with the microprocessor and proposed systems to CAN BUS. Workbenches are connected between each other and to the programable gateway which is connected to the superior workbench. This programable gateway is one of the proposed LabVIEW applications, which is used during the teaching process. During the operation of this gateway, the structure of CAN BUS is demonstrated, which means that there are different frames, the content of each type of frame, and the processing of messages. Following LabVIEW applications are used with components from real vehicles mainly instruments cluster, because on this vehicle component it is easy to see the correctness of communication. These applications serve for demonstration of transmitting data by using scaling and offset which are necessarily high and negative values that can be transmitted directly. From selected vehicle components, demonstration panel which is used as the source of communication for the proposed LabVIEW application with visualization of the instrument cluster was created. The last proposed application generates messages/signals for controlling instrument clusters.

2. CAN BUS

The CAN (Controlled Area Network) bus was developed by BOSCH company as a multimaster messaging system. The CAN BUS specifies a maximum speed up to 1 megabit per second (1 Mbps). The CAN BUS network sends small data blocks point to point from node A to node B under the supervision of a central bus, instead of large blocks like traditional networks such as USB or Ethernet. In CAN bus, data are sent and received in many short messages like temperature, engine speed, pressure, etc. This system of many short messages ensures data consistency for every node in the system [10, 11].

CAN BUS utilizes differential transmission of signals, which results in robust noise immunity and fault tolerance. The transmission of differential signals reduces noise coupling and enables high transmission speed by using twisted pair of wires. The differential transmission utilizes current flowing in each signal wire with the same amplitude but in the opposite direction, which results in a field cancelation effect. This is key for low emission of electromagnetic noise and also for high immunity of the electromagnetic noise.

The CAN protocol defines two logic states of the bus, dominant and recessive states. ISO-11898 defines a differential voltage which represents dominant and recessive state, respectively, bits as shown in **Figure 2**. In the dominant state (a logic '0'), the differential voltage between CANL and CANH must be greater than the minimum threshold value (under 0.5 V for receiver input and under 1.5 V for transmitter output). In the recessive state (a logic "1"), the differential voltage across CANH and CANL must be greater than the minimum threshold. The dominant bit wakes up the recessive bit on the bus to achieve non-destructive bit arbitration [11, 12].

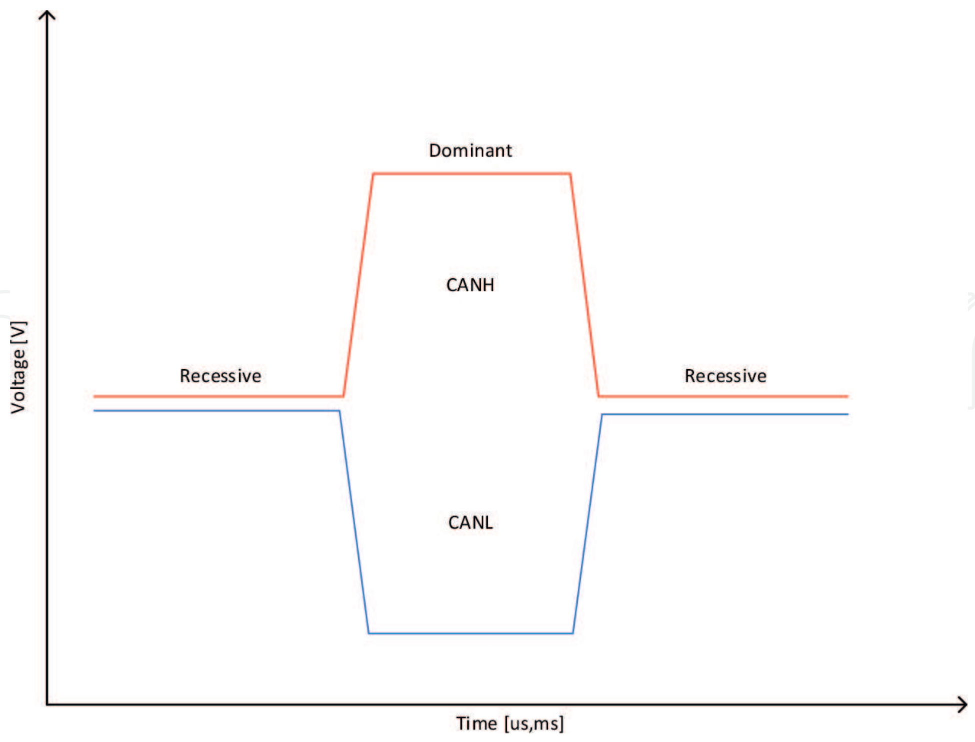


Figure 2.
Differential voltage defined by ISO-11898.

The CAN BUS is a serial communication bus defined by the International Organization for Standardization (ISO), which was developed for the automotive industry for the replacement of a complex wiring harness with a simple two-wire bus. The ISO specification requires high immunity against the electromagnetic interference and the ability to self-diagnostic and the possibility of data error corrections. These features helped to increase the popularity of CAN BUS in various fields of industries like manufacturing, building automation, medicine, etc. The communication protocol CAN BUS by ISO-11898:2003 describes how information is transferred between network devices and corresponds to the OSI (Open Systems Interconnection) model. This model is defined in terms of layers. The physical layer defines communication between nodes that are connected by physical media. The two lowest layers of the seven-layer OSI/ISO model which are the physical and data-link layers are defined by ISO 11898 [11, 13].

Figure 3 shows the application layer that establishes a communication link with a specific higher-level application protocol, such as the vendor-independent CANopen protocol. This protocol is supported by an international group of users and manufacturers from all over the world. Many protocols are designed for the specific applications such as industrial automation, diesel engines, or aerospace [9]. Other examples of CAN-based industry standard protocols are CAN KVASER and CAN from Rockwell Automation's DeviceNet [13, 14]. Even though ISO 11898 defines the electrical specification of wires and connector, it does not define wires and connector directly. ISO 11898 defines the requirement for termination resistors at each end of the bus with a value of 120 Ω, which can be seen in **Figure 4**. Most often, this termination resistor is part of hardware dedicated to CAN BUS, so a simple connection of two devices is necessary for the establishment of the communication. The disadvantage of the fact that most devices contain termination resistors is that, if lots of devices are used, the resistance of the bus will be too low. In this case, the termination resistor of some devices must be removed [12, 15].

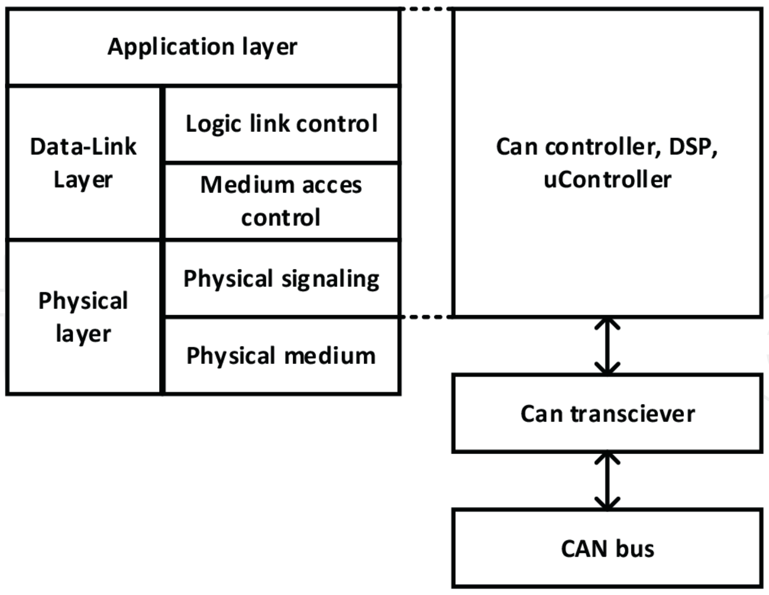


Figure 3.
Physical and data-link layer definition by ISO 11898 Standard Architecture.

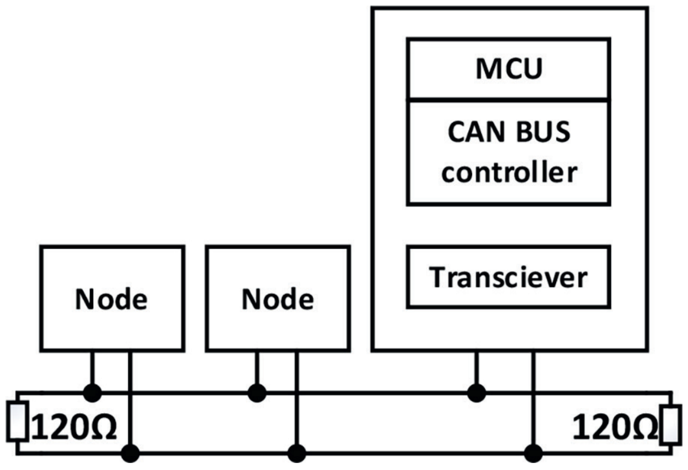


Figure 4.
Connection of nodes to bus with termination resistors.

The CAN communication protocol is a multiple access protocol in terms of the carrier signal with the detection of collision and the message priority decision (CSMA/CD + AMP). CSMA means that each bus node must wait a defined time of inactivity before attempting to send a message. CD + AMP means that collisions of the transmission of more nodes are solved using bit arbitration based on the priority of each message. This priority is defined in the identifier field of each message. The access to the bus and transmission is always given by the identifier field with the higher priority [11, 14]. The communication via CAN BUS is realized with a different type of message or frame with two lengths of identifier. The identifier can be standard with a length of 11-bit or extended with a 29-bit length. The first is the data frame which is used for the transmission of data between nodes of the bus. The second type of frame is the remote frame which is transmitted to the bus for the request of data transmission of a specific node. The third type of frame is the error frame, which is transmitted by the node, which detects the error. The last type of frame is the overload frame which is transmitted by a node to delay the transmission data of the bus to get extra time to process data.

Transfer speed defined by the ISO-11898 standard are from 125 kbps to 1 Mbps. This standard defines standard 11-bit identifier an extended 29-bit identifier [9, 16]. The content of standard 11-bit identifier is shown in **Figure 5**. This identifier provides up to 2048 different message identifiers, while the extended 29-bit identifier, which is shown in **Figure 6** provides up to 537 million identifiers [11, 17].

The meaning of the bit fields in **Figure 5** is as follows:

- SOF – one dominant Start of Frame (SOF) bit indicates the start of a message and is also synchronized bus nodes after being idle.

- Identifier – the standard 11-bit CAN identifier determines the priority of the message, the lower value in binary form means a higher priority.

- RTR – the single remote transfer request (RTR) bit is dominant when information is requested from another node. All nodes of the bus receive this request, but the identifier defined the specified node of the bus. Corresponding data are also received by all nodes and used by all involved nodes. In this way, all data used in the system are uniform.

- IDE – the single Identifier Dominant Extension (IDE) bit means that a standard CAN identifier is transmitted without an extension.

- r0 – reserved bit (for possible of future supplementation of standard).

- DLC – the 4-bit field of Data Length Code (DLC) defined the number of data bytes to be transmitted.

- Data – up to 64 data bits can be transmitted.

- CRC –15 bits with delimiter used for Cyclic Redundancy Check (CRC) contains a checksum of previous application data for error detection.

- ACK – each node receiving an accurate message overwrites this recessive bit in the original message with a dominant bit, indicating that an error-free message has been sent. If the receiving node detects an error and leaves this bit recessive, it discards the message, and the sending node retries the message after rebitting. In this way, each node acknowledges (ACK) the integrity of its data. ACK is 2 bits, one is for the acknowledgment bit and the other is used as the delimiter.

SOF	11-bit identifier	RTR	IDE	r0	DLC	8 bytes data	CRC	ACK	EOF	IFS

Figure 5.
Standard CAN BUS frame with an 11-bit identifier.

SOF	11-bit identifier	SRR	IDE	18-bit identifier	RTR	r0	r1	DLC	8 bytes data	CRC	ACK	EOF	IFS

Figure 6.
Extended CAN bus frame with a 29-bit identifier.

- EOF – this 7-bit end-of-frame (EOF) field marks the end of a CAN (message) frame and disables bit padding, indicating a padding error when dominant. When 5 bits of the same logic level occur in a row during normal operation, a bit of the opposite logic level is inserted into the data.

- IFS - this 7-bit inter-frame space (IFS) contains the time it takes for the controller to move a correctly received frame to its correct position in the message buffer area.

As shown in **Figure 6**, the Extended CAN message is the same as the standard message with the addition of:

- SRR – the Surrogate Remote Request (SRR) bit replaces the RTR bit in the standard message location as an extended format placeholder.

- IDE – a recessive bit in the identifier extension (IDE) indicates that more identifier bits follow. The 18-bit extension follows the IDE.

- r1 – After the RTR and r0 bits, an additional reserve bit was included before the DLC bit.

3. Analysis and processing of CAN BUS messages using programable gateway

Data transfer of CAN BUS using fully programable gateway is demonstrated. During the operation of this gateway, the structure of CAN BUS communication is demonstrated, which means that there are different frames, the content of each type of frame, and the processing of messages. A gateway, in general, is a device that is used for the connection of two different buses with the different transmission protocols or buses with the same protocol but with different speeds. A gateway is used for two main reasons: the first reason is the length of the bus at a maximum speed of 1 mbit/s which is only 40 m. With the existing arrangement, it is not possible to connect all students' workbenches with a superior workbench or teacher's workbench with one bus. The second reason is the connection of this superior workbench which is used for remote access to students' workbenches. By using the gateway, it is possible to connect these two CAN BUSES which can use different speeds and is possible to filter messages and resend them from the first bus to the other and vice versa. This gateway can be controlled from a superior workbench with LabVIEW. From this workbench, the message of both parts of the laboratory CAN BUS is visualized. Also, the filtering of messages is controlled from this workbench. This gateway, which block diagram is shown in **Figure 7**, utilizes NI CompactRIO 9082 as the main part. It is the system that combines a real-time processor for processing and monitoring and FPGA for high-speed logic and precise timing [18]. This system can use up to eight modules; however, this gateway utilizes only two modules. The first of the modules is CAN BUS module NI9853 which disposes of with two high-speed ports with standard DB9 connectors. The second module is the module for SD card NI9802 which is used for transferring data logs to SD cards.

As mentioned before, the system CompactRIO combines real-time operation using a processor and FPGA; therefore, the programming approach is different from

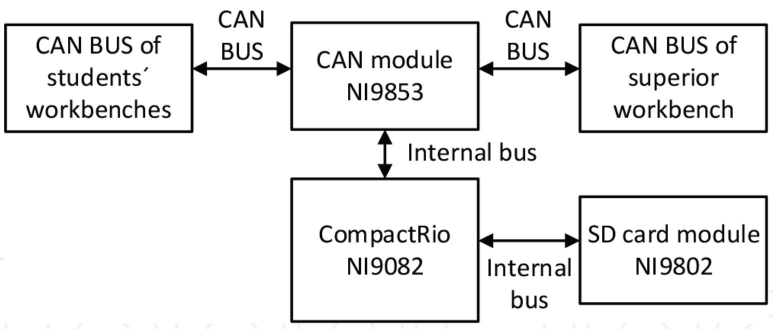


Figure 7.
Block diagram of the connection of gateway.

programming other NI measuring and communication cards, NI hardware, or third-party hardware. LabVIEW project is divided into three programs, each for one level of processing – FPGA, real-time, and host computer as we can see in **Figure 8**. The main program for the acquisition or transmission of data (signals) is based on the usage of specific modules, because some modules are dedicated for usage with FPGA processing and some modules for processing with the processor of CompactRIO. Both used modules, for CAN BUS communication and data transmission to SD cards, are dedicated to the FPGA level. At this level, the FPGA program is used for the acquisition and transmission data of CAN modules like identifier, type of frame, DLC, and an array of data. This main FPGA program is also used for saving data logs to SD cards. This program use mainly functions from the FPGA library. The next level is a real-time program that can be used for processing and visualization, but it is limited in the computation power of the CompactRIO processor. Instead of processing, this level is used for update of shared variables which are used for transmitting data to the host computer. In the host computer data, processing and visualization are limited by the computation power of the PC that is used. This approach is suitable where for visualization a large number of indicators, graphs, etc., like virtual instrument

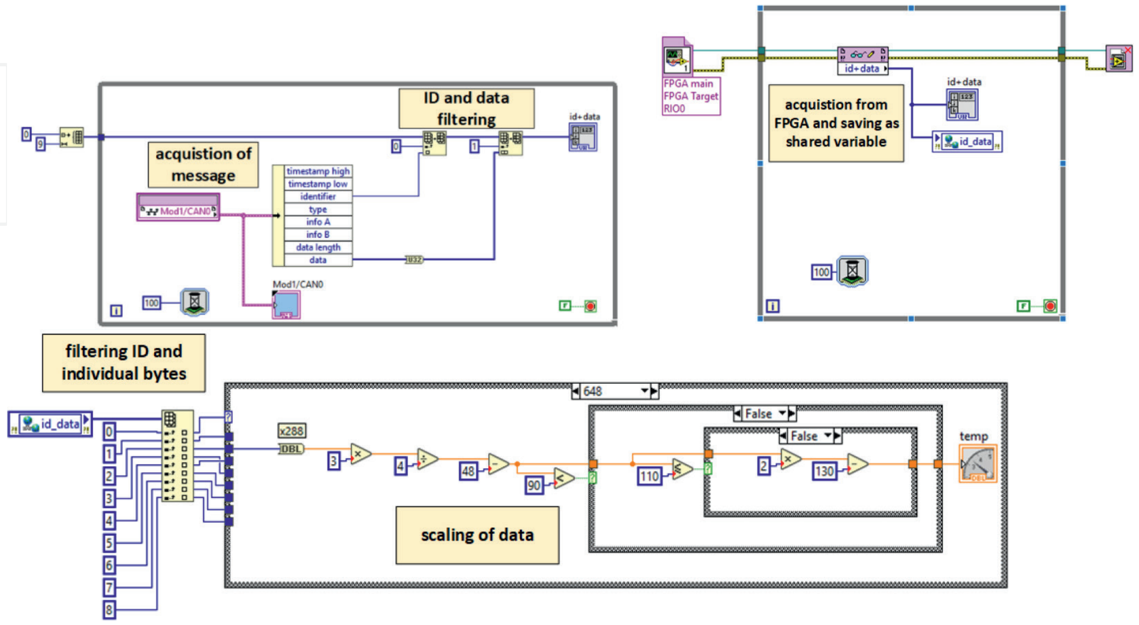
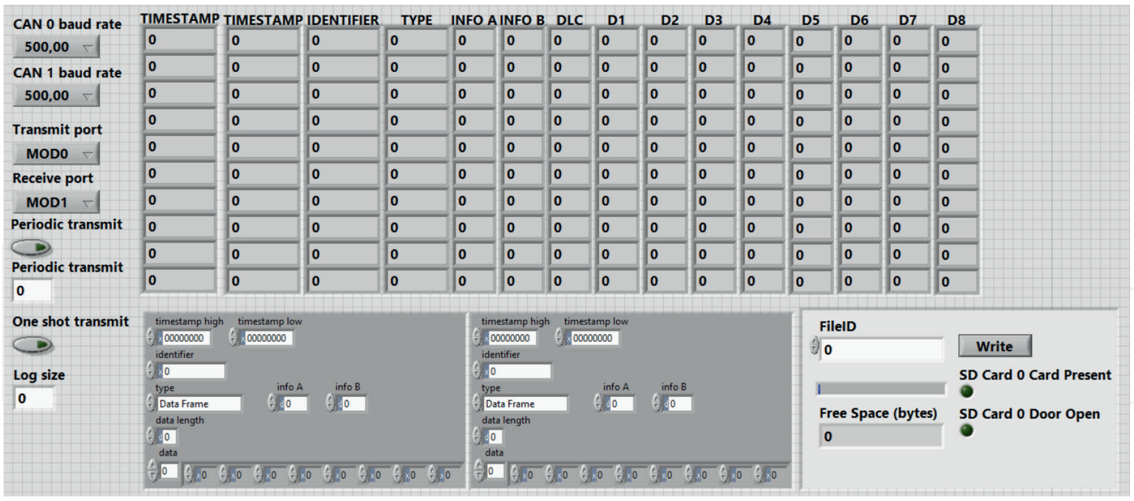


Figure 8.
Block diagram of three programs, each for one level of message processing, top left for FPGA, top right for real time, and bottom for the host computer.


CAN 0 baud rate: 500,00
CAN 1 baud rate: 500,00
Transmit port: MOD0
Receive port: MOD1
Periodic transmit
Periodic transmit: 0
One shot transmit
Log size: 0

TIMESTAMP	TIMESTAMP	IDENTIFIER	TYPE	INFO A	INFO B	DLC	D1	D2	D3	D4	D5	D6	D7	D8
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

timestamp high: x00000000 timestamp low: x00000000
identifier: x0
type: Data Frame info A: d0 info B: d0
data length: d0
data: 0 x0 x0 x0 x0 x0 x0 x0 x0

timestamp high: x00000000 timestamp low: x00000000
identifier: x0
type: Data Frame info A: d0 info B: d0
data length: d0
data: 0 x0 x0 x0 x0 x0 x0 x0 x0

FileID: 0
Write
SD Card 0 Card Present
Free Space (bytes): 0
SD Card 0 Door Open

Figure 9.
Front panel of the program for control of the gateway.

clusters application were used. Also, the complex calculation is easier with this approach because FPGA uses only fixed-point number, so some functions of libraries are missing. In this last level, data from the first or second bus are visualized.

Using front panel controls, it is possible to control the gateway as shown in **Figure 9**, a basic setting like the speed of each port and the way how the gateway works. The default mode of the gateway is resending data from the first to the second bus without filtering, so the gateway works as a repeater. This mode work with buses with the same speed or resending data from the bus with a lower speed to a bus with a higher speed. To keep the gateway simple, no buffer is needed. In this mode, data are resending when are available on the bus. The next mode is when the gateway sends data on the bus periodically and data are updated when data on the first bus were changed. This mode is used for communication with real hardware because this hardware needs precise periodic communication. The control unit controls the time between messages, and if this time is smaller or greater than the defined time control unit evaluates it as an error in the communication. From the front panel, the size of the log and the start of logging are controlled. The last mode of the operation of the gateway is the filtering work with the same speed of bus or from lower speed bus to higher speed bus because of buffering. This mode of operation has no preset filtering, and filtering is realized in the block diagram individually depending on devices connected to buses.

4. Demonstration panel of CAN BUS communication of control unit with instrument cluster and VNT turbocharger actuators and its visualization

For the demonstration of real-time communication of the control unit with sensor and actuator, a demonstration panel was created. This panel with visualization is shown in **Figure 10**. This demonstrational panel consists of a development kit with a processor with a CAN BUS module which simulated some messages between the control unit of the internal combustion engine and some sensors and actuators based on the position of the accelerator pedal. In **Figure 11**, we can see block diagram with connection of these components. This communication demonstrates the communication

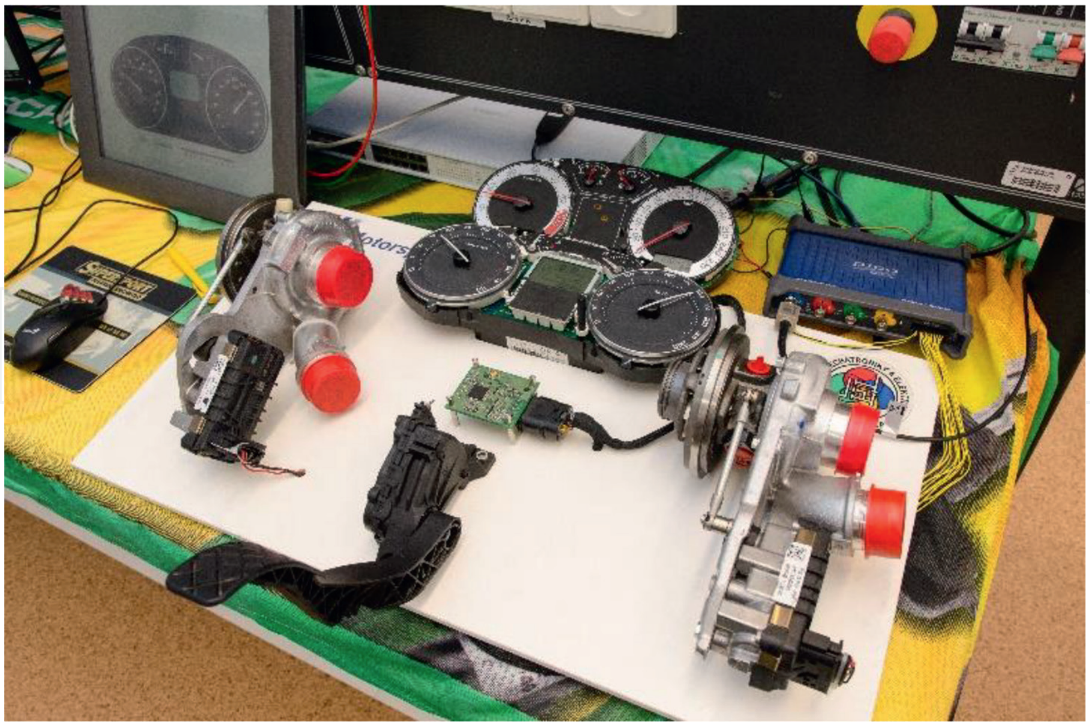


Figure 10.
Demonstration panel with the accelerator pedal, simulated control unit, instrument cluster, and turbochargers.

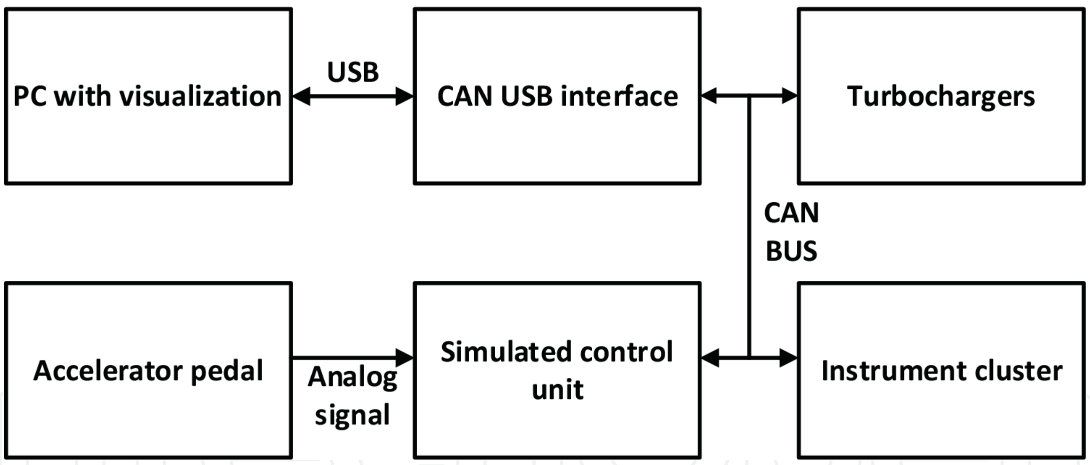


Figure 11.
Block diagram of demonstration panel and its visualization.

of real components which used scaling and offset for transmission of negative values and values greater than 255, which is the maximum value for the transmission.

The accelerator pedal is connected to two channels of development kit ADC. Two channels are used in the vehicle because of safety reasons. Both signals from the accelerator position sensor are voltage signals in the range from 0 to +5VDC and are equivalent to the position of the pedal. Most often one signal is the half-value of the other signal, or signals have opposite values. This secures that it is possible to check the validity of the signal from the accelerator pedal position sensor. If signals are not in the right relation, the signal is not valid, and the control unit works in safe mode. The next parts of the demonstration panel are two VNT turbochargers. These turbochargers use vanes to regulate boost pressure. These vanes are powered by the electric actuator, and in modern vehicles, actuators are controlled directly by CAN BUS. The last part of this demonstrational panel is the instrument cluster which shows engine speed, coolant temperature and speed of the vehicle, and some indicators. Some signals like fuel-level indicators and some warning indicators are still analog in vehicles.

For communications PC with demonstration panel USB-CAN interface, Kvaser Leaf semi-pro was chosen because of the support of the driver for LabVIEW [19]. LabVIEW drivers contain a library with functions for programming applications with this interface. Function of this library and math library are used most often in this application. This interface is capable of having speed up to 1 mbit/s and supports standard and also extended data frames. For the measurement of analog signals like accelerator pedal position, myRIO was used. For visualization, which can be seen in **Figure 12**, a virtual instrument cluster was created using gauges and some decorative elements as much as a real instrument cluster.

The virtual instrument clusters show engine speed, vehicle speed, coolant temperature, and odometer as a trip which is calculated from vehicle speed. As mentioned before, fuel level is an analog value and on the demonstration panel, the fuel level sensor is not used (**Figure 13**).

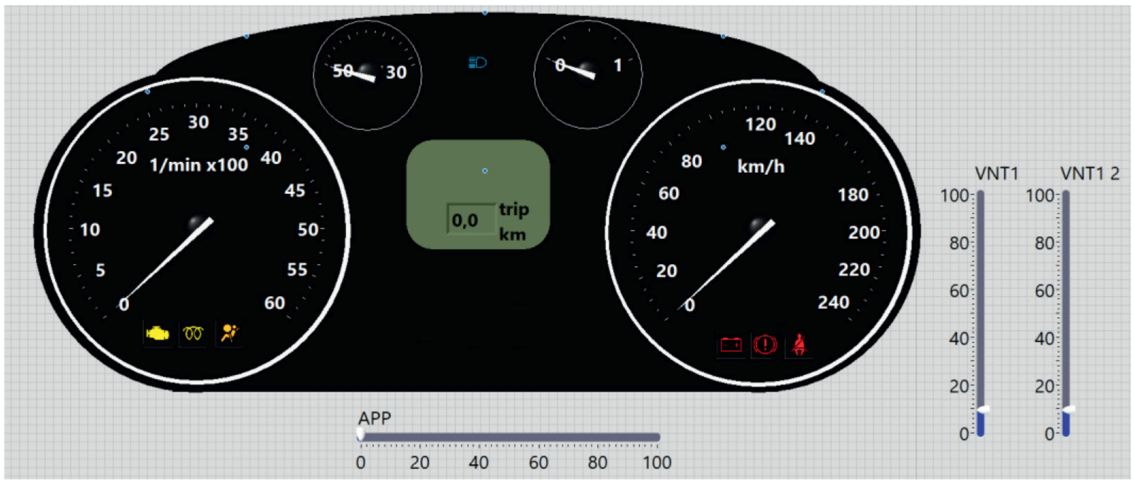


Figure 12.
Visualization of demonstration panel as virtual instrument cluster and positions of the accelerator pedal and turbochargers actuators.

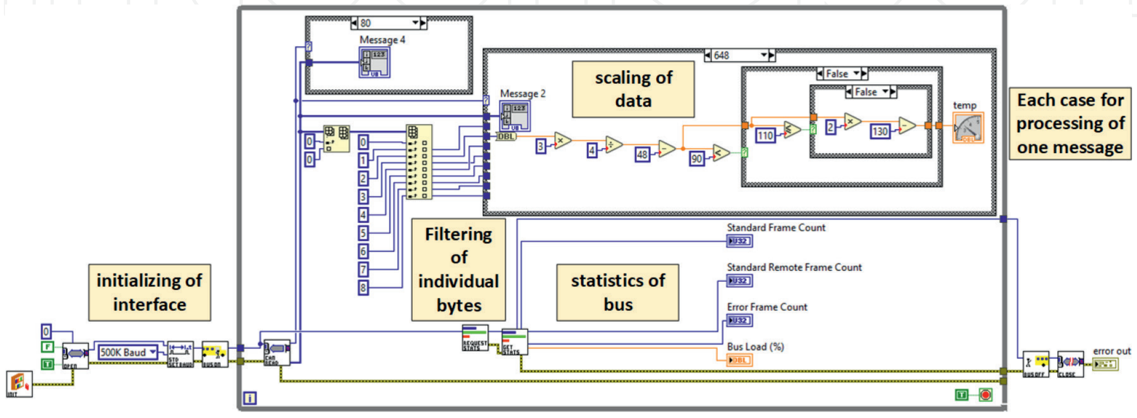


Figure 13.
Block diagram of visualization of demonstration panel as virtual instrument cluster and positions of the accelerator pedal and turbochargers actuators.

As previously mentioned, visualization must process data as a real instrument cluster (**Figure 14**). The values of engine speed and vehicle speed are processed with offset and scale, similarly, as shown in **Figure 15**. The processing of coolant temperature is respective value on indicator that is different from other indicators. It is caused by hysteresis of the thermostat of ICE cooling, so during operation the temperature rises and falls repeatedly. This means that without modifying of processing temperature, indicator will show the temperature the same way. Temperature is processed to indicator to show 90°C in the operating range of temperature (85–110°C) as shown in **Figure 13**. This means that up to operating temperature, indicator shows real temperature and in the interval of operating temperature shows 90°C. It is a temperature higher than the operating temperature, and the temperature on the indicator rises with a greater slope.

At the task of creating messages, students learn about scale and offset in the calculation of data through CAN BUS. Scaling and offset are necessary because of the structure of the data protocol of CAN BUS. In the description of the protocol was mentioned that it is possible to transmit only unsigned 8-bit numbers. That means to transmit negative values, like temperatures, usage of offset is required. For transmission of a number higher than 255, which is a maximum of one byte, scaling is needed. The second way of transmission of high values is like the speed of the engine, and the 16-bit number is split into two 8-bit numbers and these numbers are transmitted to the bus.

Figure 14 shows the structure of the real message where we can see the usage of two bytes for transmission and usage offset and scale. This message is defined by Fleet Management System (FMS) which is the protocol used in trucks of the main European manufacturers like Scania, Man, Volvo, etc. [20]. This protocol is used as an example because it is one of the few publicly available protocols. Almost all manufacturers of passenger vehicles, respectively, concern of vehicles manufacturer use their own protocols which are not usually publicly available.

This virtual instrument cluster can be used also with students´ development kits not only with the demonstration panel. It is easier for students to debug their program with real communication using this visualization instead of using a real instrument cluster which is quite big in the workspace and does not need a power supply. When students control this virtual instrument cluster, they must use defined messages to control the tachometer, speedometer coolant gauge, and other messages, for example warning indicators are defined by our visualization.

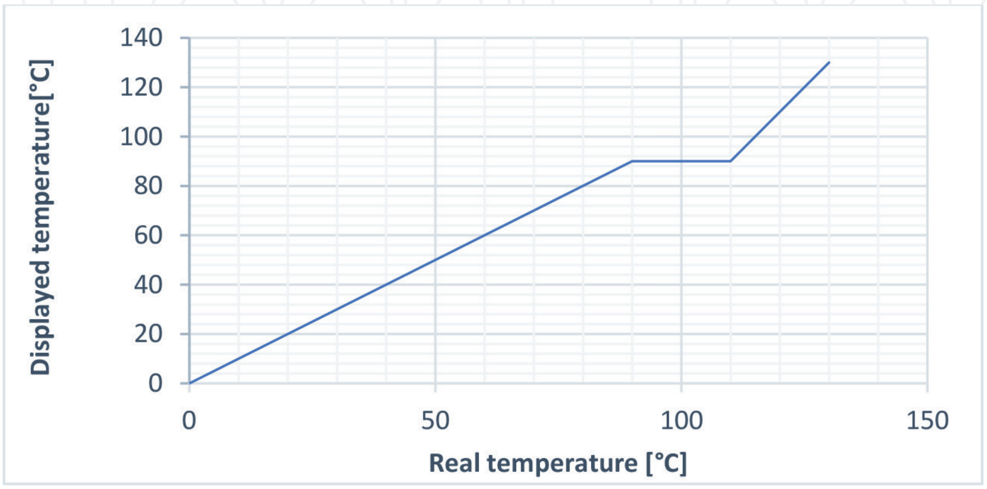
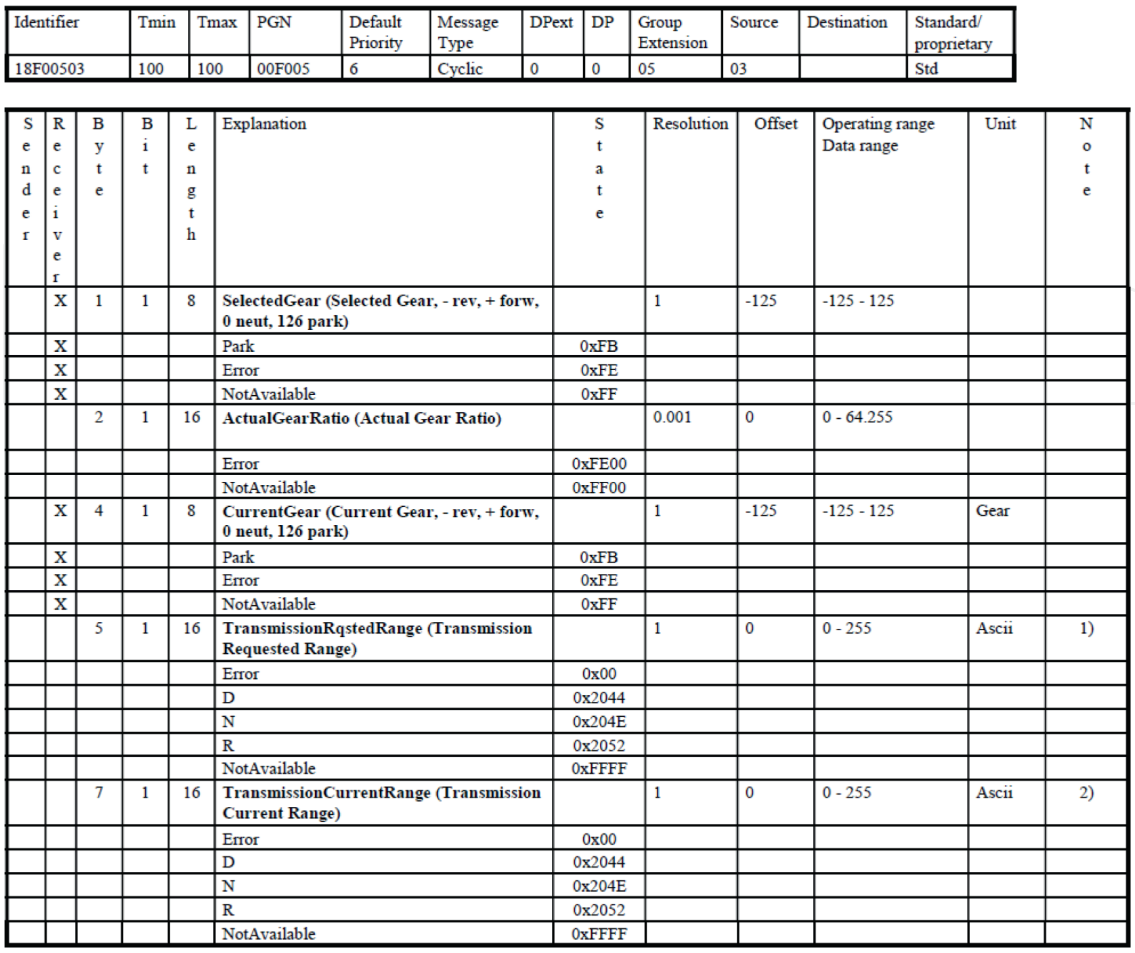


Figure 14.
Nonlinearized curve of temperature indicator.



Identifier	Tmin	Tmax	PGN	Default Priority	Message Type	DPext	DP	Group Extension	Source	Destination	Standard/ proprietary
18F00503	100	100	00F005	6	Cyclic	0	0	05	03		Std

Sender	Receiver	Byte	Bit	Length	Explanation	State	Resolution	Offset	Operating range Data range	Unit	Note
	X	1	1	8	**SelectedGear (Selected Gear, - rev, + forw, 0 neut, 126 park)**		1	-125	-125 - 125		
	X				Park	0xFB					
	X				Error	0xFE					
	X				NotAvailable	0xFF					
		2	1	16	**ActualGearRatio (Actual Gear Ratio)**		0.001	0	0 - 64.255		
					Error	0xFE00					
					NotAvailable	0xFF00					
	X	4	1	8	**CurrentGear (Current Gear, - rev, + forw, 0 neut, 126 park)**		1	-125	-125 - 125	Gear	
	X				Park	0xFB					
	X				Error	0xFE					
	X				NotAvailable	0xFF					
		5	1	16	**TransmissionRqstedRange (Transmission Requested Range)**		1	0	0 - 255	Ascii	1)
					Error	0x00					
					D	0x2044					
					N	0x204E					
					R	0x2052					
					NotAvailable	0xFFFF					
		7	1	16	**TransmissionCurrentRange (Transmission Current Range)**		1	0	0 - 255	Ascii	2)
					Error	0x00					
					D	0x2044					
					N	0x204E					
					R	0x2052					
					NotAvailable	0xFFFF					

Figure 15.
Structure of real CAN BUS message [12].

5. Control of instrument clusters using LabVIEW

Visualization of instrument clusters can be modified to be used for controlling real instrument clusters. Gauges are used as controls, so the value is chosen with the position of the gauge needle. Based on the position of the needle of gauges, data of CAN BUS messages are calculated. Data are calculated opposite as for visualization, for example for visualization of engine speed, data from the bus are divided by 4, and for control, data are multiplied by 4. Block diagram for calculation of data with visualization is shown in **Figure 16**. For demonstration and easier understanding of calculation of message data, subresults and results are shown in decimal and also hexadecimal form, as shown in **Figure 17**. For communication, CAN-USB interface Kvaser Leaf semi-pro is used. This control of the instrument cluster by using virtual instrumentation serves for the analysis of communication with the real instrument cluster. With this analysis, it is easier for the student to program their own development kit for controlling the real instrument cluster or its visualization.

For the demonstration of the advantages and disadvantages of using the bus in the vehicle, an instrument cluster from an old BMW was used, which was constructed without any automotive bus. Processing signals are shown in **Figure 18**.

On this BMW cluster, all indicators and gauges are controlled by an analog signal. The front panel of the program for control of which cluster is shown in **Figure 19**. For

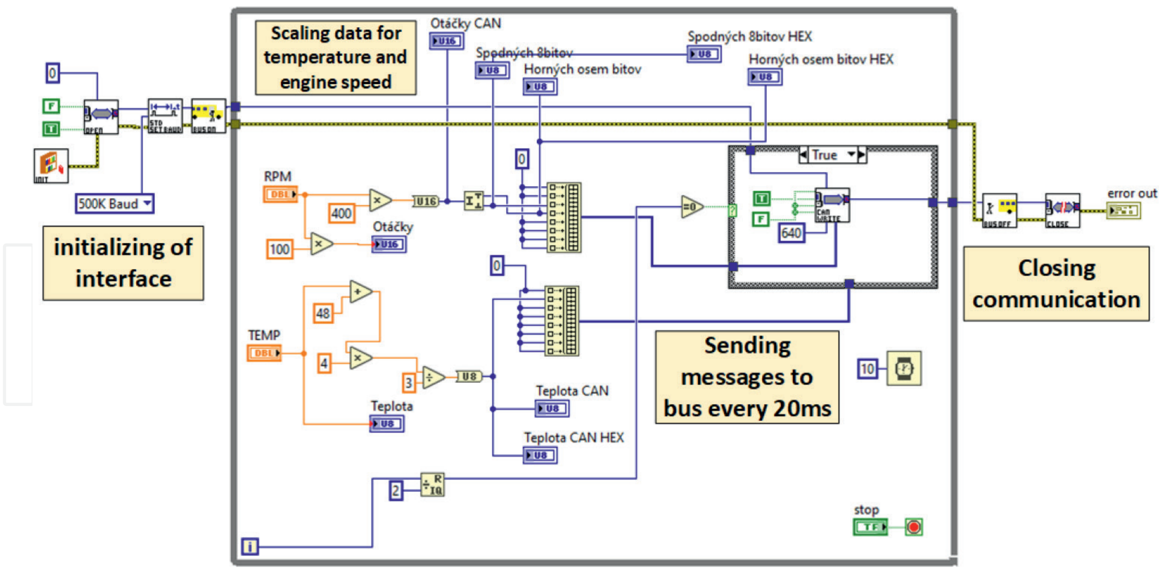


Figure 16.
Block diagram of LabVIEW program for control of VW instrument cluster through visualization with the calculation of values in decimal and hexadecimal format.

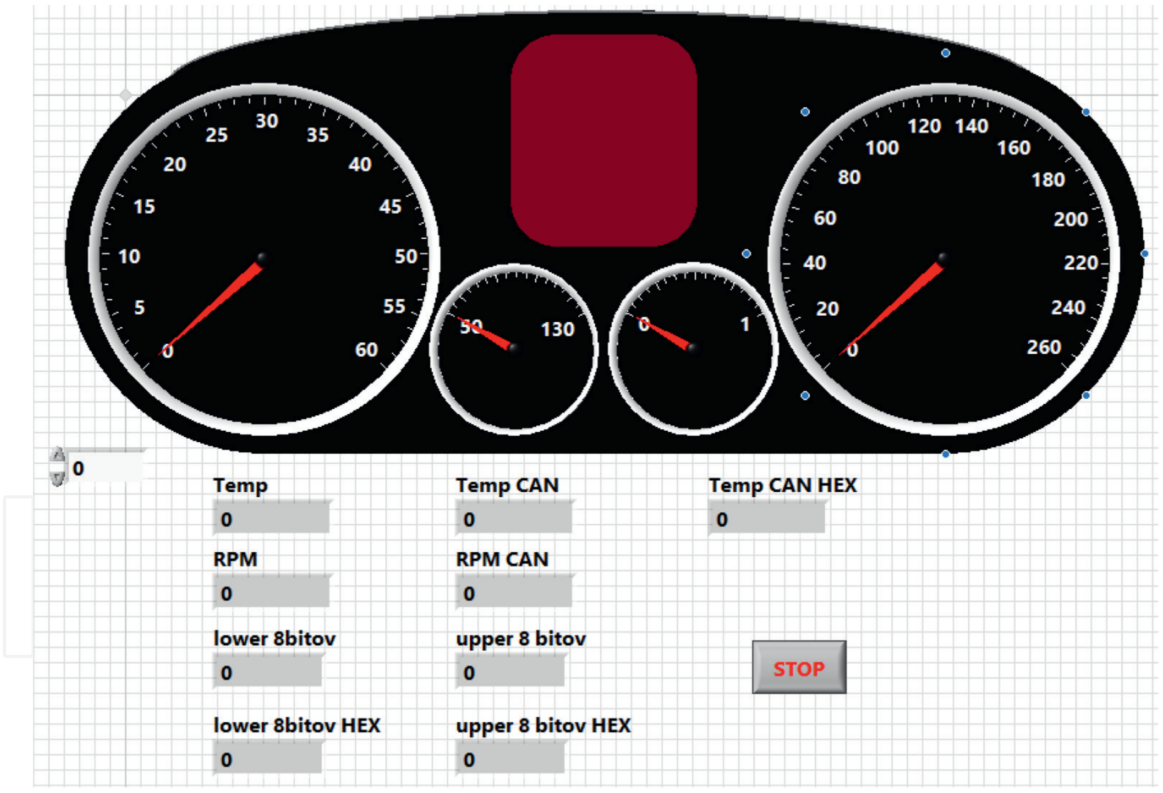


Figure 17.
Control of VW instrument cluster through visualization with the calculation of values in decimal and hexadecimal format.

indicator, it means control with connection +12VDC or GND. For engine speed and vehicle speed, the hall sensor or the inductive sensor is used within the circuit for the signal processor, so the signal is a square wave with a duty cycle of 50%. NTC thermistor serves as the coolant temperature sensor, so the output signal is resistance. The potentiometer is used as a fuel-level sensor, so the output is resistance as well. These two gauges are controlled by PWM with fixed frequency and variable duty cycle.

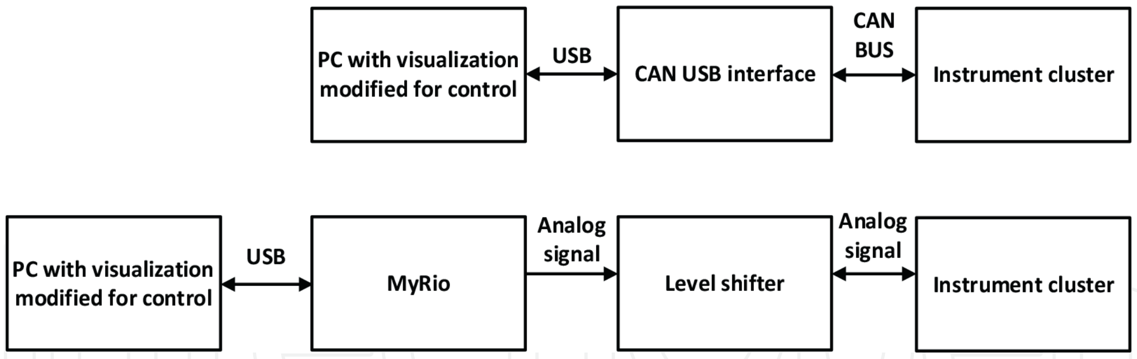


Figure 18.
Block diagram of the connection of the control instrument cluster with CAN BUS (top) and using analog signals (bottom).

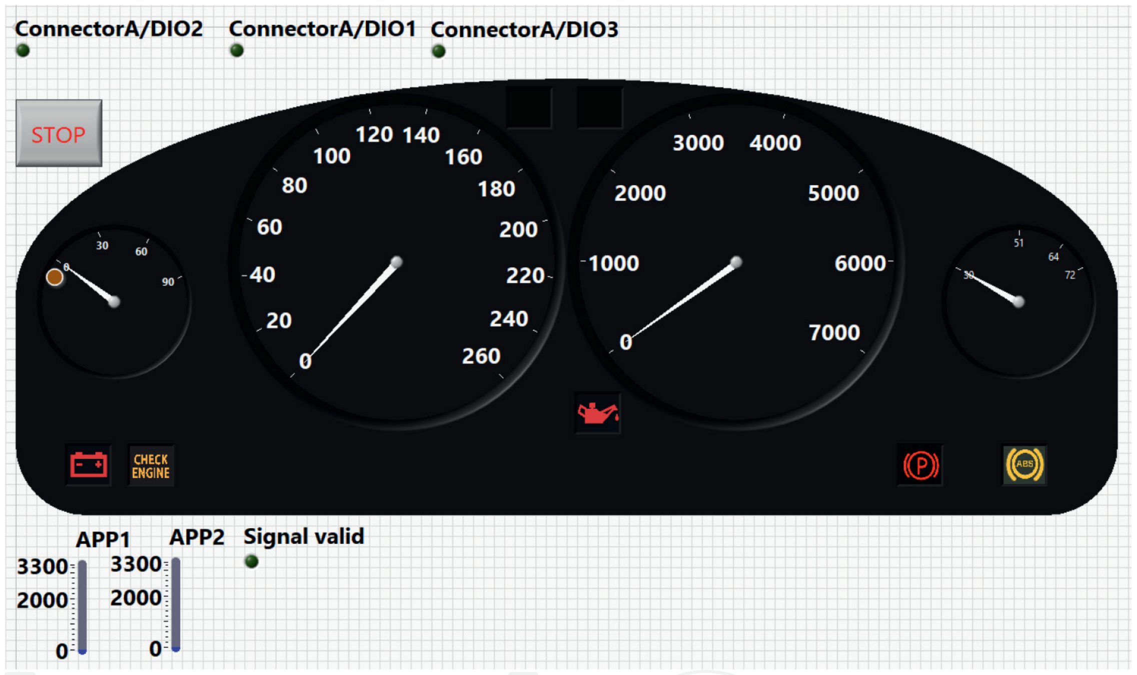
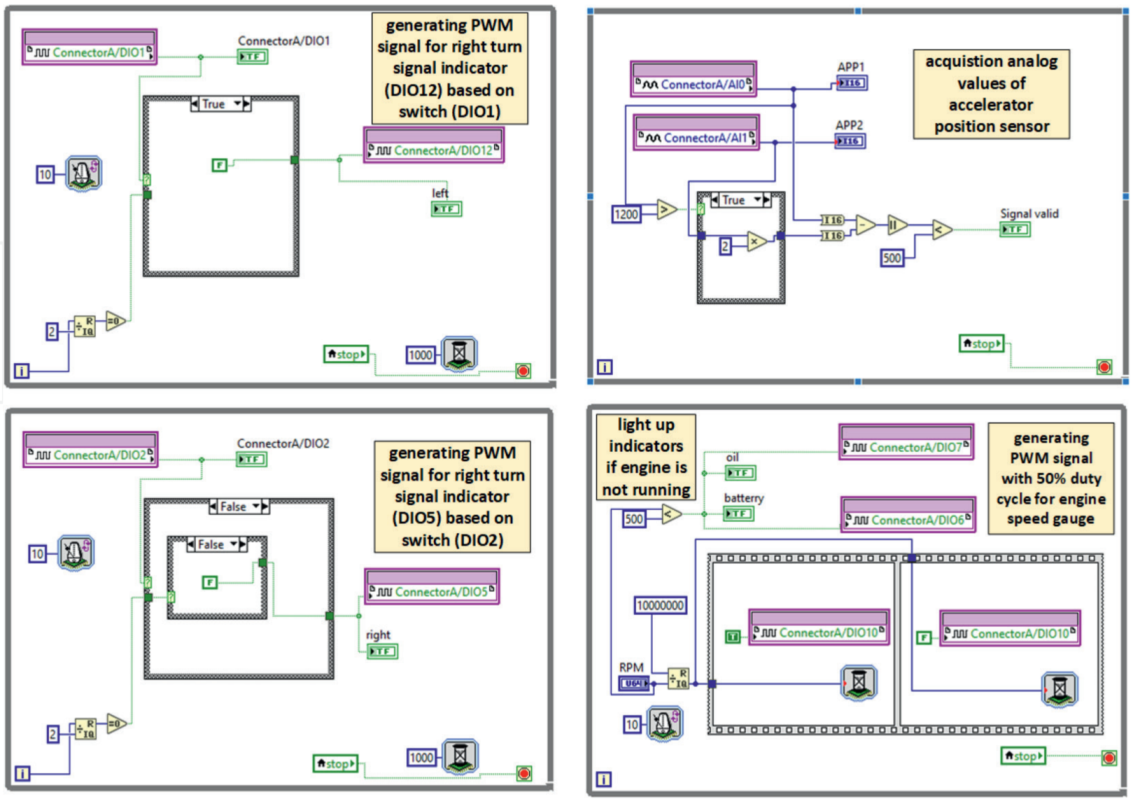


Figure 19.
Control of BMW instrument cluster through visualization with the generation of analog signals.

For generating PWM signal and logic levels for indicator, myRIO was used. MyRIO is a portable device dedicated for students to the design of control, robotics, and mechatronics system. MyRio contains two identical 34-pin connectors, connectors A and B, with 16 digital inputs/outputs, three 0–5 V analog inputs, and one analog output, 3.3 V and 5 V power output. Two voltage levels are available, because this compact device can use as digital input logic of both voltage levels, but the output has 3.3 V voltage only. Digital inputs/outputs have internal pullup resistors to 3.3 V, and this means that the state of this input without a signal will be always logic 1. Thus, these inputs are controlled by logic 0 (connection to ground). This device also contains a third 20-pin connector C with two ±10 V differential analog inputs and two outputs, eight digital input/outputs, 5 V, +15 V/ -15 V power outputs. Digital inputs/outputs of this connector have internal pulldown resistors to ground, and this means that the state of this input without a signal will be always logic 0. Thus, these inputs are controlled by logic 1 (connection to power output). Each of the analog outputs



generating PWM signal for right turn signal indicator (DIO12) based on switch (DIO1)
acquistion analog values of accelerator position sensor
generating PWM signal for right turn signal indicator (DIO5) based on switch (DIO2)
light up indicators if engine is not running
generating PWM signal with 50% duty cycle for engine speed gauge
ConnectorA/DIO1
ConnectorA/DIO12
left
ConnectorA/AI0
ConnectorA/AI1
APP1
APP2
Signal valid
ConnectorA/DIO2
ConnectorA/DIO5
right
ConnectorA/DIO7
ConnectorA/DIO6
ConnectorA/DIO10
oil
battery
RPM
stop

Figure 20.
Block diagram of myRIO FPGA application for generating PWM for control of BMW instrument cluster.

of all three connectors has its own digital-to-analog converter (DAC), so all analog outputs can be updated simultaneously. All digital-to-analog converters of analog outputs are controlled from FPGA by two serial buses, one for connectors A and B and the second for connector C [21]. This hardware was chosen because of the need to generate four PWM channels and eight GPIO pins in a compact package. With myRIO is used a small board with logic-level transistors to shift logic levels from 3.3 V (myRIO) to +12 V (instrument cluster). Since, myRIO contains FPGA and a processor like cRIO, the programming approach is similar. In **Figure 20**, we can see part of the block diagram of the program for generating PWM signals. For communication with CAN BUS, Kvaser Leaf semi-pro interface was used, as in other applications. Here, students may realize how CAN BUS can simplify communication and save cabling. For controlling this instrument cluster, theoretically, one message will be enough but with analog control, 22 analog signals (we use only 8 indicators and 12 signals total) would be needed.

6. Conclusions

This chapter is focused on the usage of virtual instrumentation, respectively, LabVIEW to make understanding of autotronics easier for students. Due to the growing amount of electronics in modern cars, where control units, sensors, and actuators communicate with buses, we aim for automotive bus CAN BUS. This bus was chosen because this is one of the most popular automotive buses which is now also used in various fields of industry. A demonstration panel and several applications were created to demonstrate the structure and possibilities of the communication, advantages, and disadvantage of using the bus. The first application is the programable gateway which is implemented in CompactRio with a CAN BUS module and SD card module. This programable gateway can work in different modes. In all working modes, data are transferred from the bus with a lower speed to the bus with a higher speed, or bus with the same speed because of using no buffers. In the first mode, the gateway is only resending CAN BUS frames from one bus to another like a repeater. The data are transferred only when are data available on the bus. In the next mode, data are periodically sent on the second bus, and data are refreshed when data appear on the first bus. This mode is more suitable for usage with real hardware, because real hardware measures the time between messages, to detect the error in communication. That means if the time between messages is greater or smaller than defined, the control unit evaluates corrupted communication. The last mode is the mode with filtering where filtering is implemented in a block diagram based on connected hardware, and there are no preset filters. Settings of the gateway are on the front panel of the application, like the speed of both ports, the working mode of the gateway, and the start of the log to the SD card. The students working with this gateway can better understand the frames of CAN BUS protocol, the structure of frames, etc. The second application is the visualization of the demonstration panel which consists of a development kit (control unit), accelerator pedal, and two turbochargers. The development kit simulates the control unit and based on the position of the accelerator pedal sends messages to actuators of turbocharges and to the instrument cluster. The visualization of this panel visualizes the instrument cluster, the position of actuators of both turbochargers, and the position of the accelerator pedal. This visualization can be used with students' development kits; therefore, the debugging is easier, and no additional hardware is needed. The CAN BUS communication needs scaling and offset to transmit negative values and values greater than 255, which is a maximum of one byte. With this visualization, students adopt the calculation of data for messages to control the instrument cluster. The last application controls the instrument cluster from the vehicle which instrument cluster communicates with CAN BUS and the second instrument cluster which needed analog signals. Students can see the difference in complexity of wiring with analog signals compared to simple wiring of CAN BUS.

Acknowledgements

This publication was realized with the support of Operational Program Integrated Infrastructure 2014–2020 of the project: Innovative Solutions for Propulsion, Power and Safety Components of Transport Vehicles, code ITMS 313011 V334, co-financed by the European Regional Development Fund and grant no. VEGA 1/0593/20: Research on power flow control in the network using a smart transformer.

Conflict of interest

The authors declare no conflict of interest.

Author details

Matúš Danko*, Ondrej Hock, Jozef Šedo and Branislav Hanko
Faculty of Electrical Engineering and Information Technology, Department of Mechatronics and Electronics, University of Zilina, Zilina, Slovakia

*Address all correspondence to: matus.danko@feit.uniza.sk

References

[1] Vento JA. Application of Lab VIEW in higher education laboratories. Proceedings Frontiers in Education Conference. 1988;**1988**:444-447. DOI: 10.1109/FIE.1988.35023

[2] Stefanovic M, Cvijetkovic V, Matijević M, Simic V. A lab VIEW-based remote laboratory experiments for control engineering education. Computer Applications in Engineering Education. 2011;**19**:538-549. DOI: 10.1002/cae.20334

[3] Ponce Cruz P, Molina Gutiérrez A. Lab VIEW for intelligent control research and education. In: 2010 4th IEEE International Conference on E-Learning in Industrial Electronics. Glendale, AZ, USA. 2010. pp. 47-54

[4] NI, How Do I Use LabVIEW to Teach Engineering Students?, Available online: https://www.ni.com/cs-cz/shop/labview/how-do-i-use-labview-to-teach-engineering-students.html (Accessed 26 August 2022)

[5] Automotive Training Equipment Car Chassis System Training Simulator Vocation Educational Equipment for School, Available online: https://www.alibaba.com/product-detail/Training-Simulator-Car-Training-Simulator-Automotive_1600282573224.html?spm=a2700.7724857.0.0.59bd2e76RoWeK9&s=p (Accessed 26 August 2022)

[6] Automotive Training Equipment Car Electronic Cruise Control System Trainer School Laboratory Equipment, Available online: https://www.alibaba.com/product-detail/Automotive-Training-Equipment-Car-Electronic-Cruise_1600267487948.html?spm=a2700.details.0.0.63fa69daHuHV12 (Accessed 26 August 2022)

[7] Vdovic H, Babic J, Podobnik V. Specialized vehicle CAN Bus simulator: From modelling to validation. In: 2020 5th International Conference on Smart and Sustainable Technologies (SpliTech). Split, Croatia. 2020. pp. 1-7

[8] Li Y, Liyuan T, Xiaochuan T. Simulation for vehicle comfort system based on CAN bus. In: 2010 International Conference on Computer, Mechatronics, Control and Electronic Engineering. Changchun, China. 2010. pp. 227-230

[9] Liu C, Luo F. A co-simulation-and-test method for CAN Bus system. Journal of Communications. 2013;**8**(10):681-689

[10] NXP, Inc., Bosch Controller Area Network (CAN) Version 2.0, Available online: https://www.nxp.com/docs/en/reference-manual/BCANPSV2.pdf (Accessed 26 August 2022)

[11] Texas isntrument, Introduction to the Controller Area Network (CAN). 2008, Available online: https://www.nxp.com/docs/en/reference-manual/BCANPSV2.pdf (Accessed 26 August 2022)

[12] CAN protocol used in the STM32 bootloader, Application note, Available online: file:///C:/Users/User/Downloads/an3154-can-protocol-used-in-the-stm32-bootloader-stmicroelectronics.pdf (Accessed 26 August 2022)

[13] Richards P. A CAN Physical Layer Discussion. U.S.A.: Microchip Technology Inc; 2002

[14] Watterson C. Controller Area Network (CAN) Implementation Guide, APPLICATION NOTE, Available online: https://www.analog.com/media/en/technical-documentation/

application-notes/an-1123.pdf (Accessed 26 August 2022)

[15] Connecting C166 and C500 Microcontroller to CAN, Application Note,V1.0. 2004. Available online: https://www.infineon.com/dgdl/p2900010_C166_C500_CAN.pdf?fileId=db3a304412b407950112b409fb420402 (Accessed 26 August 2022)

[16] Yang H-R, Jeong C-S, Kim H-S, Yang S-Y. A study on overall vehicle monitoring system for black box using LabVIEW. In: 2011 11th International Conference on Control, Automation and Systems. Gyeonggi-do, Korea (South). 2011. pp. 1217-1220

[17] Luo F, Li R. LIN network simulation system based On LabVIEW. In: 2010 International Conference On Computer Design and Applications. Qinhuangdao, China. 2010. pp. V5-299-V5-303

[18] cRIO-9082 Features, Available online: https://www.ni.com/docs/en-US/bundle/crio-9082-feature/page/um-purpose.html (Accessed 26 August 2022)

[19] KVASER LEAF SEMIPRO HS manual, Available online: https://canlandbucket.s3-eu-west-1.amazonaws.com/productionResourcesFiles/f5b04eef-2c58-4bab-9db6-b361d8b329fc/73-30130-00242-5_EN.pdf (Accessed 26 August 2022)

[20] CAN Communication Specification for FMS, Available online: https://bodybuilder.scania.com/content/dam/bodybuilder/local-page/thailand/CAN_Communication_Specification_for_FMS.PDF (Accessed 26 August 2022)

[21] myRIO Toolkit, Available online: https://www.ni.com/docs/en-US/bundle/labview-2019-myrio-toolkit/page/myriohelp/myriohelp.html (Accessed 26 August 2022)