Matej Kubis - Miroslav Gutten - Daniel Korenciak - Matus Danko*

COMMUNICATION OF CLUSTER USING CAN BUS

The paper deals with the communication between the infotainment elements and the drive component of the car using the CAN bus. In the introduction, there is briefly described the communication bus used in cars. It describes its basic features, benefits and utilization. The main task is graphically programmed and builds a dashboard that will resemble to a real dashboard from a Volkswagen Golf 5. Using the dashboard built in LabVIEW, it controls the real dashboard.

Keywords: *communication, CAN bus, LabVIEW, automotive*

1. Introduction

Fieldbuses are now part of every automobile power supply, which are used for receiving, processing and sending of data between sensors, actuators and controllers. Sending the information between devices, for example about the position of the accelerator, it is preferable to encode the information and send it through cheap wires (two twisted wires), as in case that a separate cable will be used for each informative state. The disadvantage is the need for encoders and decoders, and with that the associated need for complex, diagnostic tools. However, if the bus is used in the vehicle, wiring will be simplified, the number of wires reduced as well as terminal blocks and the speed of communication increased [1].

Transmission simulating systems are the important annulus in car factory total assembly workshop automation equipment. By applying the fieldbus technology, it made more reliable transmission system operation, maintenance convenience, which also greatly reduced the overall cost [2].

2. Description of automotive CAN bus

In 1980, Bosch brings to the stage the technical world of networking controllers called CAN - bus. CAN bus is not used only in the automotive industry. CAN bus is a system for the current communication (real time) of control units. It is not used for the direct connection of sensors or actuators, but only to link controllers. The other elements of the system, such as sensors and actuators, are linked by indirect method. In the case of exchange of information via the CAN data bus, all data is transmitted only through two wires. Both bidirectional conductors will carry the same data, regardless of the number of control units and the amount of transmitted data. Transmission of information through the CAN bus has meaning only in case when we need to send a lot of information between multiple controllers [3].

While one unit is connected to the bus transmits data, other units receive them and evaluate them as well. If the unit recognizes the important information, further processes them and sends a signal to actuator. If information is unnecessary, remains passive. For example, the transmission of the information of the vehicle speed display: management control unit measures the value of the speed sensors and the coded sequence to a series of ones and zeros will send it to the CAN bus control unit of the instrument panel. The control unit must decode the message and ensure that it is comprehensible displayed, for example with a digital position indicator or speedometer [4].

The bus may be in two states - the dominant state (dominant) or a recessive (recessive), shown in Figure 1. The dominant state occurs when at least one unit transmits, i.e. "switch is open", the voltage on the bus is about 2 V. On the other hand, recessive condition occurs when the "switch is turned off" and the voltage on the bus is 0 V, it means the "silent bus", there is no sending unit, or any unit can begin transmitting. In case of recessive (silent) mode, the voltage of both wires (CAN-High and CAN-Low) is 2.5 V to the chassis, against each other to 0 V. The CAN-H has a voltage of 3.5 V and CAN-L has 1.5 V in the dominant mode. It is referred differential voltage of 2V. The differential voltage between conductors is very important; therefore, bus is from this perspective differential [5].

The most basic data chip versions are: Basic and Full CAN. In Basic CAN, there is a message identified and controlled by the processor of connected device. Full CAN has a filter in the communication chip that decides whether to accept the message or send it to the processor. Full CAN version burdens less own processor of connected device [6].

These two different versions are still used today although not in the exactly same form. Due to the wide range of chips, it is currently available for more than 60 chips of about sixteen producers. Full CAN has two variants: variant A, the 11-bit identifier and variant B (i.e. magnification) with 29 bits. The versions are not compatible. The rate depends not only on the variant of the protocol, but also on the length of the bus.

* [1]Matej Kubis, [1]Miroslav Gutten, [1]Daniel Korenciak, [2]Matus Danko
[1]Department of Measurements and Applied Electrical Engineering, Faculty of Electrical Engineering and Information Technology, University of Zilina, Slovakia
[2]Department of Mechatronics and Electronics, Faculty of Electrical Engineering and Information Technology, University of Zilina, Slovakia
E-mail: matej.kubis@fel.uniza.sk

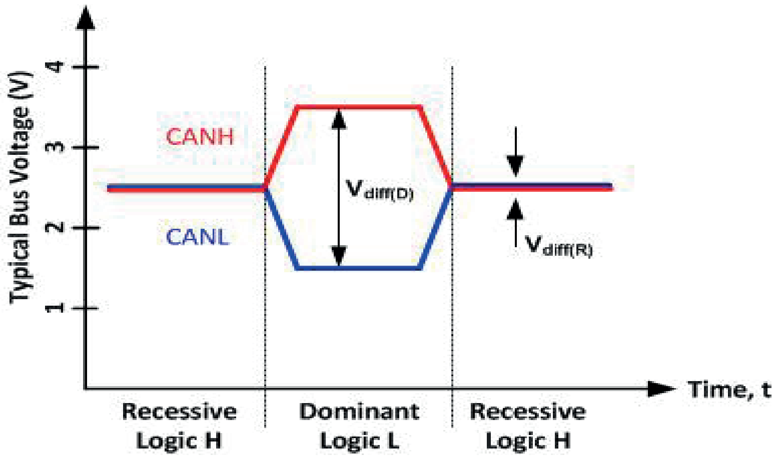
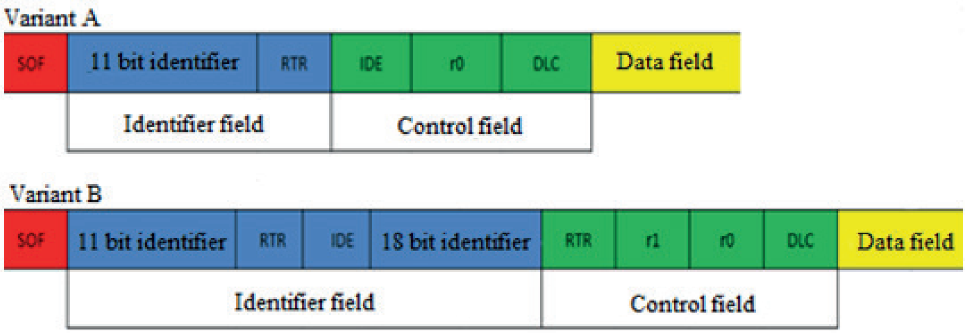


Figure 1 *Voltage level on the CAN*



SOF - Start of frame - initialization and synchronization field
RTR - Remote Request - bit, which is dominant in the data frame
SRR - Substitute Remote Request - RTR replaced by the variant B
IDE - Identifer Extension - bit, which allows to distinguish variants of what it is
DLC - Data Length Code - Code for the length of the data field
r1, r0 - reserved bits (complement DLC)

Figure 2 *Format data frame*

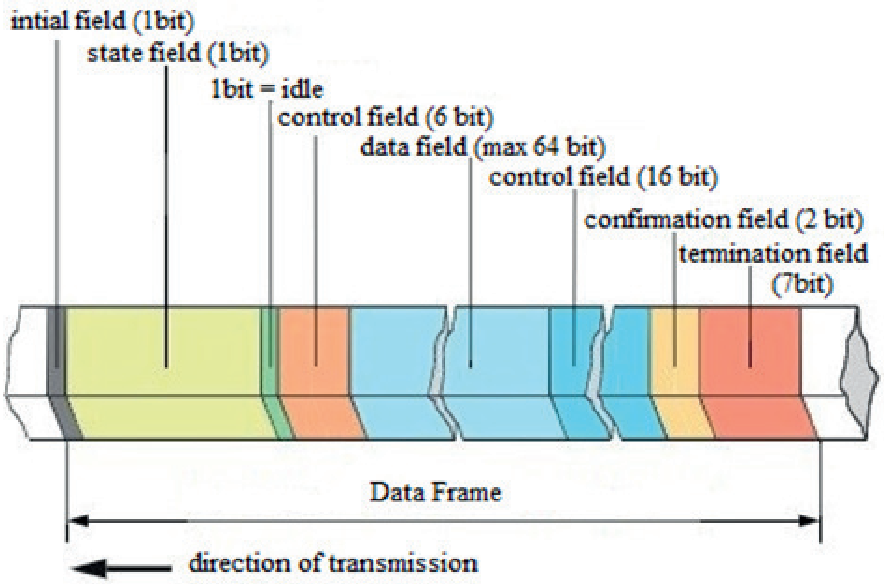


Figure 3 *Field data frame*

A comparison of variant A and variant B is plotted in Figure 2.

Data frame consists of several fields: Start framework, decision-making field, control field, data field, a checksum field, the confirmation field and end frame. Composition data frame is shown in Figure 3.

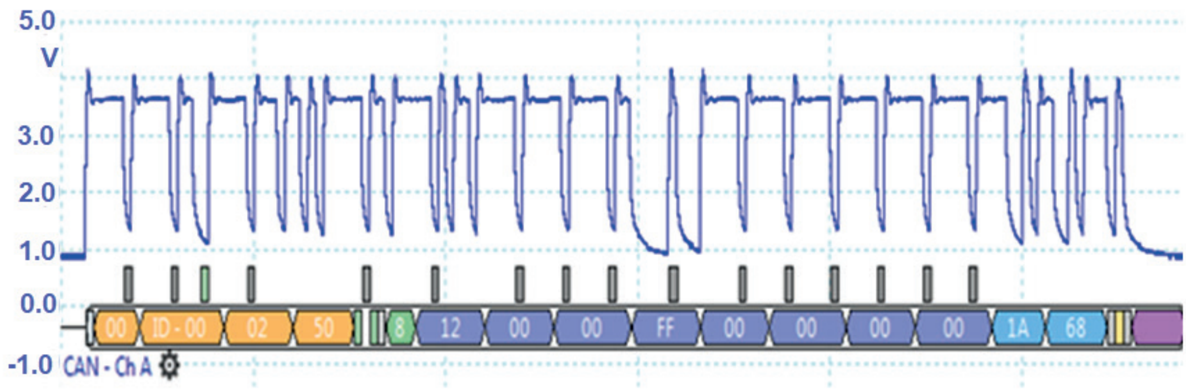


Figure 4 *Extended data frame*

There are three types of CAN buses that are different data rates, use and construction (mainly type exciter):

- HSCAN - high speed CAN bus (High Speed),
- LSCAN - low-speed CAN bus (Low Speed).

Their direct connection is not possible; interconnect through a gateway (gateway). HSCAN is two-wire, control voltage is between drivers about 2V at the end of the interconnected resistors 120 Ohm, unlike LSCAN, consisting of one active conductor and ground and recessive voltage of about 4V (use LSCAN is mostly less intensive systems such as. for comfort electronics.

3. Design and simulation test system dashboard

Implementation of communication was carried out on the data bus CAN. We have chosen CAN bus, because it is the most used bus in automotive applications. Loaded progress on the CAN bus is implemented by using a Pico PC Oscilloscope, sending broadcast messages is provided by the software of bussmaster, where are sent only randomly generated messages [7].

The difference between standard CAN 2.0 A and CAN 2.0 B is the number of bits of the identifier. The standard data frame in the arbitration field contains of 11 - bit identifier and the RTR bit. In Figure 4 there is an enlarged data frame which contains the extended 29 -bit ID, and the RTR bit SRR bit in the arbitration field [8].

The identifier is randomly selected to 0 x 250, other bits are gradually complemented by value, as can be seen in the data frame. This is for showing of data and their generation without using real elements of the car. Behind 0x250 HEX is 8 bits sequentially.

Data analysis was carried out using the CAN hardware Kvaser Leaf SemiPro and software bussmaster. In between each 10 m with a different message is sent over the bus, which sends the real instrument panel, and is, for example: error codes of missing sensor, missing immobilizer, the absence of the control unit of the airbag etc.

In Figure 5 there are shown values of individual bits on the bus along with identifiers that are purposefully deleted due to copyright manufacturer. Inspiring information offers a single interface, where you can record real-time log data without further attachment.

The main task was to program and graphically construct the instrument panel, which will resemble a real dashboard of the car Volkswagen Golf 5.

The first report processes data for speedometer, a given message can be up to 8 bytes and 1 byte can have a value from 0 to 255. If there is needed a larger value, we must use more bytes. There are needed 2 bytes for simulation of the rpmmeter: the second byte as eight low-order bits, the third byte as eight upper bits. Subsequently, the resulting value is divided by four and we get a real running speed. By dividing with the rest, we determine whether the bit is zero or unity. If the final bit is zero, the number is even; therefore, it must be divided by two because of the zero residues. To find out the remaining bits, we must use the logical rotation, so the bit, whose value we want to determine, stands at the end point. In the second report, there are data processed for the temperature and a fuel gauge.

The fuel gauge is one of the few that is not used for its function, CAN bus, but it operates analog, thus it was used only for simulation. The signal for the temperature of the motor is transmitted by the first byte. The signal must firstly be necessary arranged; therefore, he is divided by ¾. The temperature can also be negative; however, we have only positive numbers, so we should subtract the value of 48.

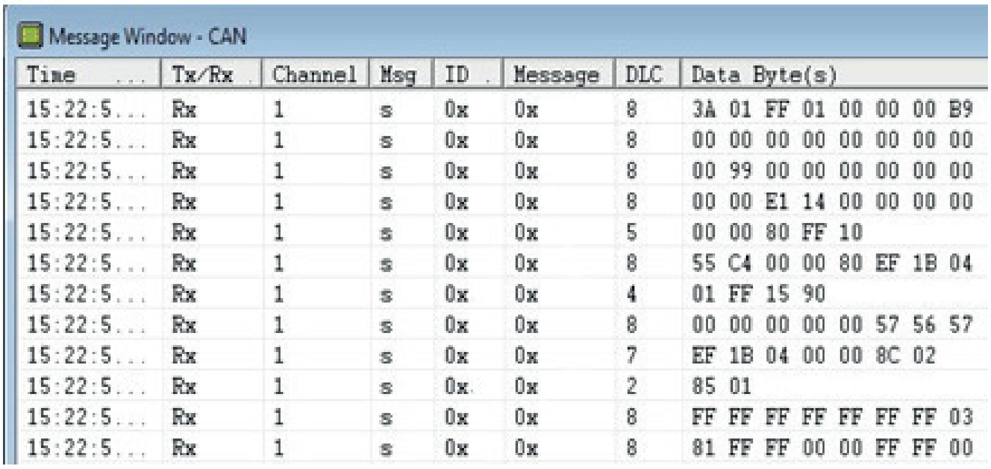
The pointer of the temperature of engine is non-linearized because of the constantly moving pointer on the temperature which is graphically illustrated in Figure 6.

The real temperature of the motor measured by the sensor is displayed to 90 °C. The temperature between 90 °C and 110 °C is working temperature of the engine, and so we have not seen constant movement of a hand for a little change of temperature this tolerance is continuously displayed as 90 °C on the indicator. From 110° C to 130 °C, the pointer indicates the real value to avoid overheating of the motor. The curve is easily adjusted by using the equation of the line:

$$y = k + q \tag{1}$$

$$90 = 110k\text{-}q$$

$$130\ 130k + q =$$



Time ...	Tx/Rx	Channel	Msg	ID	Message	DLC	Data Byte(s)
15:22:5...	Rx	1	s	0x	0x	8	3A 01 FF 01 00 00 00 B9
15:22:5...	Rx	1	s	0x	0x	8	00 00 00 00 00 00 00 00
15:22:5...	Rx	1	s	0x	0x	8	00 99 00 00 00 00 00 00
15:22:5...	Rx	1	s	0x	0x	8	00 00 E1 14 00 00 00 00
15:22:5...	Rx	1	s	0x	0x	5	00 00 80 FF 10
15:22:5...	Rx	1	s	0x	0x	8	55 C4 00 00 80 EF 1B 04
15:22:5...	Rx	1	s	0x	0x	4	01 FF 15 90
15:22:5...	Rx	1	s	0x	0x	8	00 00 00 00 00 57 56 57
15:22:5...	Rx	1	s	0x	0x	7	EF 1B 04 00 00 8C 02
15:22:5...	Rx	1	s	0x	0x	2	85 01
15:22:5...	Rx	1	s	0x	0x	8	FF FF FF FF FF FF FF 03
15:22:5...	Rx	1	s	0x	0x	8	81 FF FF 00 00 FF FF 00

Figure 5 *Bus data analyzed using the software Bussmaster*

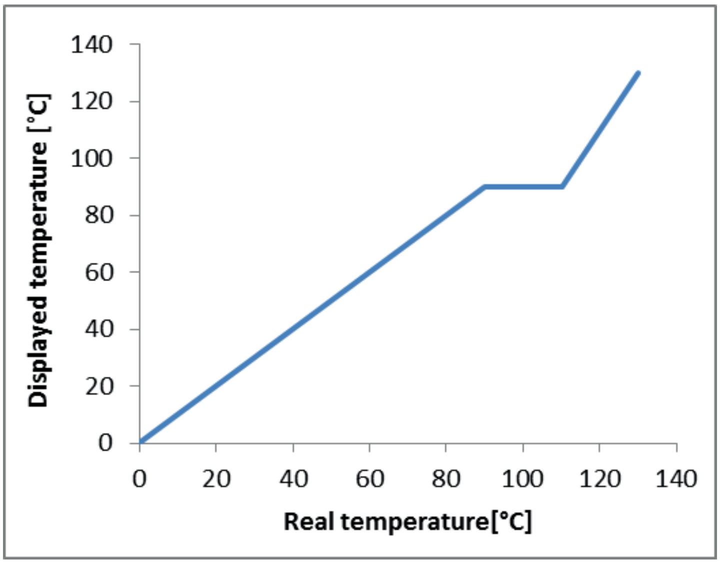


Figure 6 *Characteristics of the pointer to engine temperature*

90 - 130 = 110k - 130k

20k = 40

k = 2

90 * 2 = 110 + Q

q = -220 + 90

q = -130

Final, the third report is essential for processing of engine speed. Motor speed data is transmitted by the second and the third bytes. The second byte is lower eight bits and the third byte is upper eight bits of the resulting number. The resulting number must be divided by 100, so we get speed. Simulated speedometer is shown in Figure 7. The real instrument alarms from the VW Golf V were graphically implemented in LabVIEW and associated with the code.

The real instrument alarms from the VW Golf V were then graphically implemented in LabVIEW and associated with the code. Using CAN analyzer can be analyzed and read messages saved on the SD card. The necessary data to display speed and temperatures have been mentioned above. The basis of these alarms forms a variable speed with the speed of which is completed by the water temperature and the fuel indicator. Because the simulated data processing speed according to predetermined ID instrument manufacturer. In this case, it is a car manufacturer VW Group. Specific identifiers along with the necessary bits administration were associated with a virtual environment and the real-time output. As well as for the engine speed is used in various identifiers for engine temperature and speed. In addition to the fuel level indicator going over the bus all the information, which are needed for the driver. Graphic visualization of the instrument panel is in Figure 8.

We watered the real dashboard by a regulated power supply. The value of the resources we have set at about 12 V. The creation of a network CAN we used industrial computer cRIO containing the CAN card. Test communication of the virtual instrument panel with a real instrument panel shown in Figure 9.

For connecting wires CAN-H and CAN-L, we used a modified connector DSUB9. After starting the communication, we can see that communication takes place in real time, which means

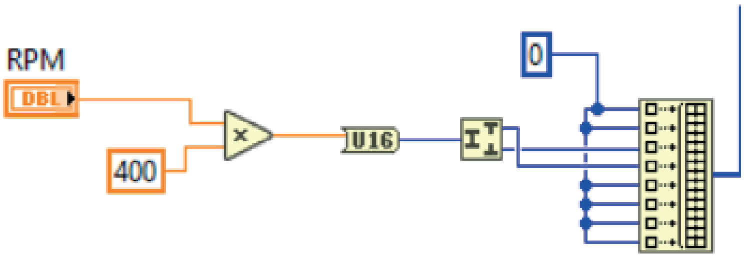


***Figure** 7 Rpmmeter in LabView*

***Figure** 8 Graphic instrument panel in LabView*

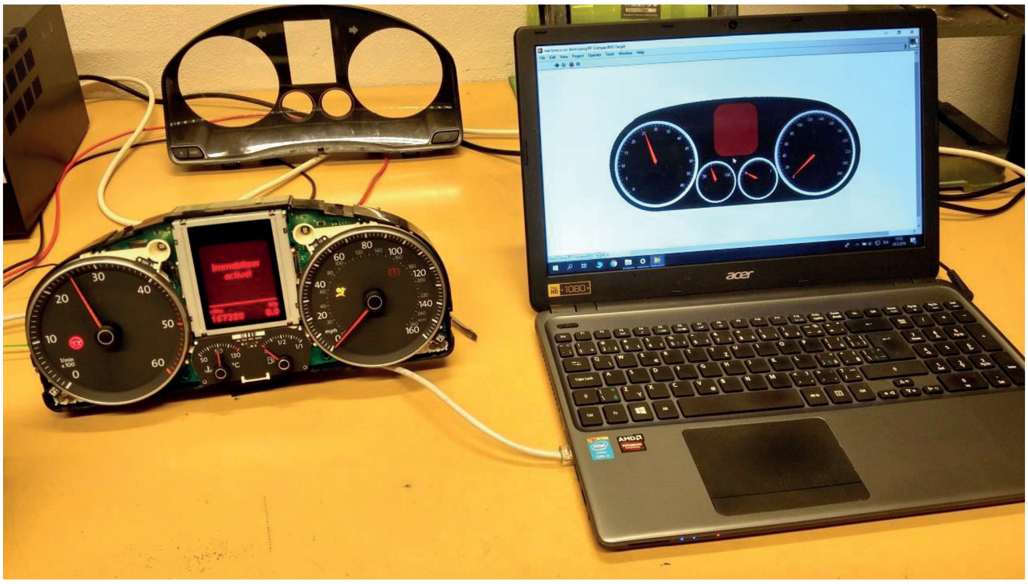


***Figure** 9 Dashboard communication test*

that if you deviate rpmmeter needle or needle pointer engine temperature, so we deflect the needle on a real instrument panel. The function of the speedometer is due to the complexity of the simulation omitted because real dashboard compares multiple messages, that structure is complex. The fuel gauge is analogue and is the only part of the instrument panel which does not use the CAN bus.

4. Conclusion

The aim of the paper was to handle the issue of using of control and communication systems for automotive applications. In the theoretical part of the article we dealt with general CAN bus. From the outset the random data were generated by using of the software bussmaster and later were sent to the bus. In the second part of the implementation, the main task was simulation of the real dashboard of the car Volkswagen Golf 5 by using the software LabVIEW and controlling of it in real time.

The rpmmeter and engine temperature indicator were dominated by us. The fuel gauge was controlled analogue, so it was used only for simulation. Speedometer was not operated by us due to the difficulty of simulation and the need of the other control units.

Acknowledgements

This article is the result of a project implementation: Modern methods of teaching of control and diagnostic systems of engine vehicles, ITMS code 26110230107, supported by the Operational Programme Educational.

References

[1] BOSCH R. Bosch automotive electrics and electronics [online]. Plochingen: Robert Bosch GmbH., 2007. ISBN 978-3-658-01783-5/ eISBN 978-3-658-01784-2. Available from: https://doi.org/10.1007/978-3-658-01784-2

[2] LEI Y., HAIHONG W. The design of car transport line control system based on fieldbus technology. World Automation Congress (WAC) : proceedings. Piscataway, NJ: IEEE, 2012. ISBN 9781467344975.

[3] STERBA P.: Electronic equipment car (in Czech). Brno: Computer Press, 2004. ISBN 978-8-025-10211-4.

[4] SEBOK M., et al: Diagnostics of ignition systems. *JEE - Journal of Electrical Engineering* [online]. 2013, **13**(4), p. 181-186. ISSN 1582-4594. Available from: http://jee.ro/articles/WH1360660218W511a06fad8ebe.pdf

[5] SHINDE, A. CAN - Controller area network. LAP Lambert Academid Publishing, 2012. ISBN 978-3659284946

[6] FRIVALDSKY M., DRGONA P., SPANIK P.: Experimental analysis and optimization of key parameters of ZVS mode and its application in the proposed LLC converter designed for distributed power system application. *International Journal of Electrical Power & Energy Systems* [online]. 2013, **47**(1), p. 448-456. ISSN 0142-0615/eISSN 1879-3517. Available from: https://doi.org/10.1016/j.ijepes.2012.11.016

[7] Automotive oscilloscopes. Pico Technology 2018 [online]. Available from: https://www.picoauto.com/products/automotive-oscilloscope-kit/overview

[8] CAN in modules. Bosch invented for life 2017 [online]. Available from: http://www.bosch-semiconductors.com/ip-modules/