

PAPER • OPEN ACCESS

Development of the upper-layer electronic control unit based on a production-type unit for a commercial vehicle

To cite this article: D A Astapenko *et al* 2019 *IOP Conf. Ser.: Mater. Sci. Eng.* **534** 012012

View the [article online](#) for updates and enhancements.



IOP | ebooks™

Bringing you innovative digital publishing with leading voices to create your essential collection of books in STEM research.

Start exploring the **collection** - download the first chapter of every title for free.

Development of the upper-layer electronic control unit based on a production-type unit for a commercial vehicle

D A Astapenko¹, D O Butarovich² and A A Smirnov³

Bauman Moscow State Technical University, 105005 Baumanskaya 2th st., 5,
Moscow, Russian Federation

E-mail: ¹ astap-astap@yandex.ru, ² buta73@mail.ru, ³ smr_a@mail.ru

Abstract. The development of a modern vehicle is associated with the need to co-operate with a variety of components from different manufacturers, due to the fact that companies mainly specialize in the manufacture of specific equipment (powertrain, steering, braking system, dashboard, etc.). Majority of manufacturers of automotive electronic units do not allow changes to their finished products, therefore, when integrating components into a single vehicle, there is a problem with the coordination of control signals and communication protocols. In response to this problem, it is required to use an additional electronic unit - the electronic control unit of the upper layer (ECU). This device generates signals for each component in accordance with the requirements of manufacturers. This unit may have different components and different functionality depending on the requirements. The unit is based on a microcontroller with a control program. To control the power discrete output, power transistor MOSFET keys are used. Optocouplers or specialized microchips are used for processing discrete digital signals. Moreover, the unit is equipped with specialized microcircuits, which allow the exchange of information with other electronic units via digital protocols.

The modern development of motor vehicles is characterized by the extensive use of electronic devices to control various units and systems: powertrain, steering, braking system, cushion system, etc. These motor vehicles electronic control systems are most often integrated into a single communication network to enable data exchange and effective work with the ability to implement intersystem functions.

The most common way to integrate electronic control systems in commercial motor vehicles is the CAN communication bus, for which the link-layer and physical layer protocols are standardized by the ISO 11898 standard group.

American standard SAE J1939-71 [1] is most prevalent in commercial motor vehicles for the protocols of the upper (applied) layer, in which standard parameters are determined and are transmitted as bus messages. However, despite the presence of the SAE J1939-71 standard, when integrating ECU from different manufacturers in one motor vehicle, there is a problem with the coordination of control signals and communication protocols.

There is a practice in the Russian automotive industry in which, when integrating ECU of foreign manufacturers into a newly developed motor vehicle, foreign partners, as a rule, do not provide an opportunity to correct (parameterize) the ECU internal software. Often, only the so-called CAN matrix of messages received and sent by the ECU is provided. As a result, a modern commercial motor vehicle equipped with a large number of control systems contains several ECUs with a list of messages that are defined in accordance with SAE J1939-71, but often there are situations when CAN messages from



ECUs of different manufacturers have Identical identifiers (IDs), different definitions of the beginning of the bytes and bits counting, etc. As a result, the integration of these units in the motor vehicle becomes difficult to implement.

It is necessary to use an additional electronic unit in the motor vehicle - the electronic control unit of the upper layer (ECU) to solve this problem. This device generates signals for each ECU component in accordance with the requirements of manufacturers.

There are several solutions to the problem of integrating components: use a ready-made commercial product with extensive parametrization capabilities or the possibility of adjustments; order on a specialized enterprise the development of a new electronic control unit for a vehicle; develop a prototype electronic control unit for debugging algorithms of work on a prototype vehicle [2, 3, 4, 5].

All of these methods require time and financial resources, while it should be noted that most of the integrated foreign ECU have abundant characteristics for processing input and output signals [6, 7].

Currently, the specialists of Bauman Moscow State Technical University have been tasked with integrating several ECUs aboard a city bus with a combined power plant application consistent with the use of modern ECUs of type safety ADAS level 1 in the design of this vehicle. The solution of these tasks is carried out in accordance with the technical requirements of the work "Development of scientific and technical solutions for the creation of a Russian combined power plant for small-class urban and suburban buses", carried out under the Agreement No. 14.574.21.0178 (a unique work identifier: RFMEFI57417X0178) and with the financial support from the Ministry of Higher Education and Science of Russia.

Earlier, the self-engineered ECU based on the STM32 microcontroller was used on the first prototypes during the electric bus LiAZ-6274 development performed by LLC "LiAZ" in cooperation with Bauman Moscow State Technical University.

The use of the self-engineered ECU allowed integration of many ECUs aboard the electric bus and performing factory tests in the shortest period.

According to the results of the electric bus factory tests, the hardware and software of the self-engineered ECU were completed due to the need to increase the number of processed signals [6].

Based on the test results and experience gained during the electric bus test operation, technical requirements were formed for the ECU for a modern commercial motor vehicle: bus, electric bus, combined power plant bus, trolley bus, etc.

Electronic control unit may have different components and different functionality depending on the requirements. The unit is based on a microcontroller with a control program. Power transistor MOSFET keys are used to control the power discrete outputs. Optocouplers or specialized microchips are in operation for processing discrete digital signals. In addition, the ECU can be equipped with specialized microcircuits, which allow the exchange of information with other electronic units using digital protocols.

Due to limited demand for such ECUs when performing the above tasks in cooperation with Industrial Partner LLC "United Engineering Center" we made a decision to select and use commercial control system electronic units produced in Russia in our future work.

As a result of the review and analysis, the serial engine control unit MD22 produced by NPP "ITELMA" (Figure 1) was selected as the basic unit.

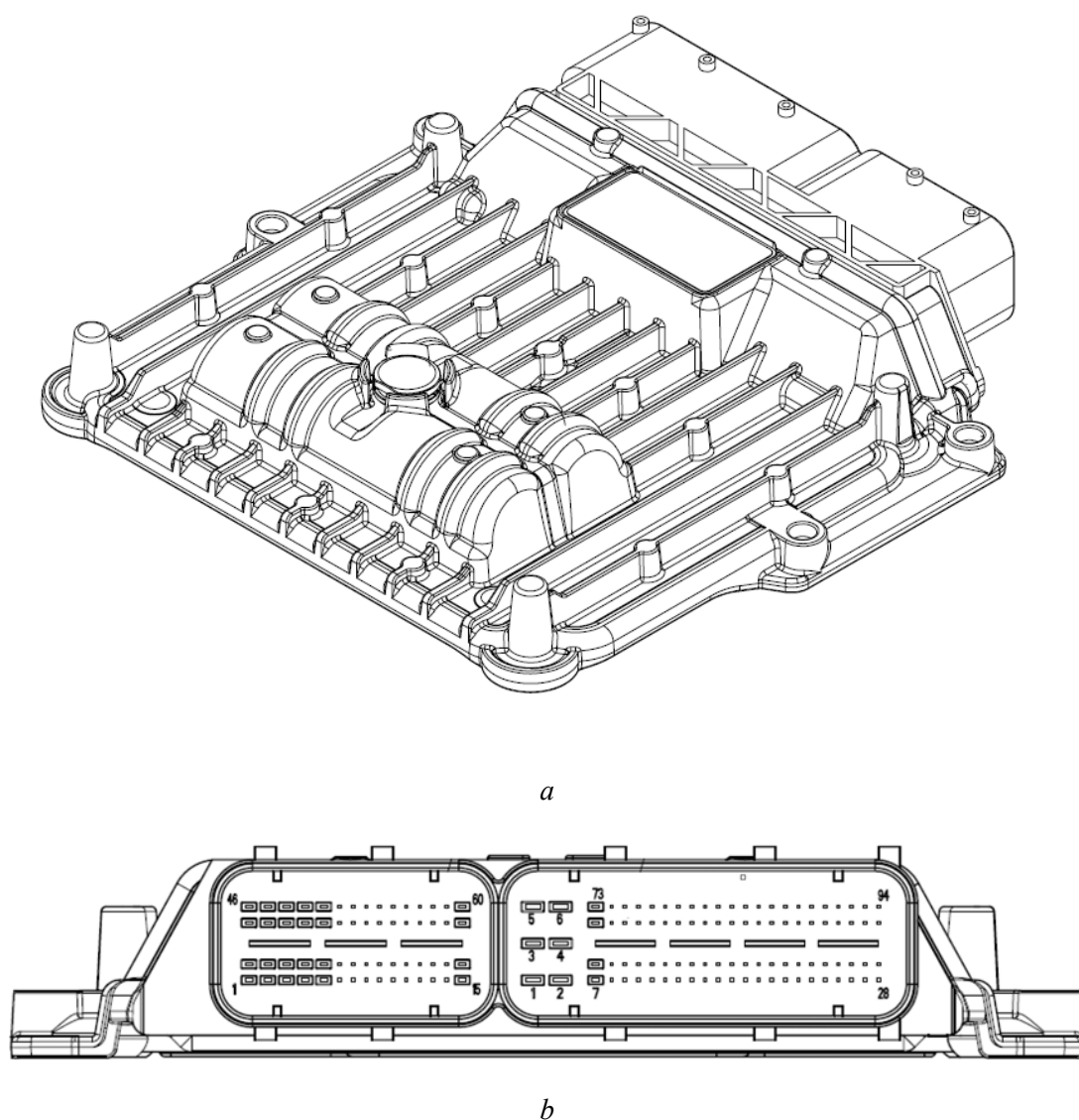


Figure 1. ECU MD22: a – case MD22, b – Unit connector MD 22.

Control system electronic unit MD22 is used in the internal combustion engine control system for automotive truck with a supply voltage of 24V. Brief characteristics control system electronic unit MD22 are shown in Table 1.

Table 1. SCEU MD22 brief characteristics.

№	Interfaces	Quantity	Note
1	Digital Input	15	
2	Analog Input	20	Voltage 0...5 V
3	Frequency Input	3	
4	Electric output («back-ups»)	20	0,2A...2A
5	Electric output («upper key»)	8	up to 5 A
5	Electric output PWM	6	
6	CAN 2.0b	2	Two-direction

Previously arising questions regarding the requirements of tightness, vibration resistance, temperature range, the use of electronic components of the AEC (Automotive Electronics Council) standard and safety, dependent on the control system electronic unit architecture, were settled automatically when using the MD22 unit.

The unit electronic circuit and the software of the lower layer remained unchanged when developing a control system electronic unit based on MD22, and the upper layer software was developed newly in the MSE Pro environment provided by NPP “ITELMA”. The MSE Pro environment is based on MATLAB / Simulink software and uses the same user interface as Simulink.

Figure 2 shows the block diagram of the on-board control system electronic unit of the city bus, in which the control system electronic unit MD22 is presented.

As an example of working with the unit software, the description of the top-layer algorithms implementation is given below: receiving and sending CAN messages between different control system electronic unit with data frame correction and control of the high-voltage contactors for the traction battery charging.

Receiving and sending CAN messages is carried out using higher layer functions. In order to work with a function, it is required to set its input and output parameters in the correct format (see Figure 3). A function is called at different layers of software by the name of this function (Action name), and it does not matter how the function is implemented at the lower layer.

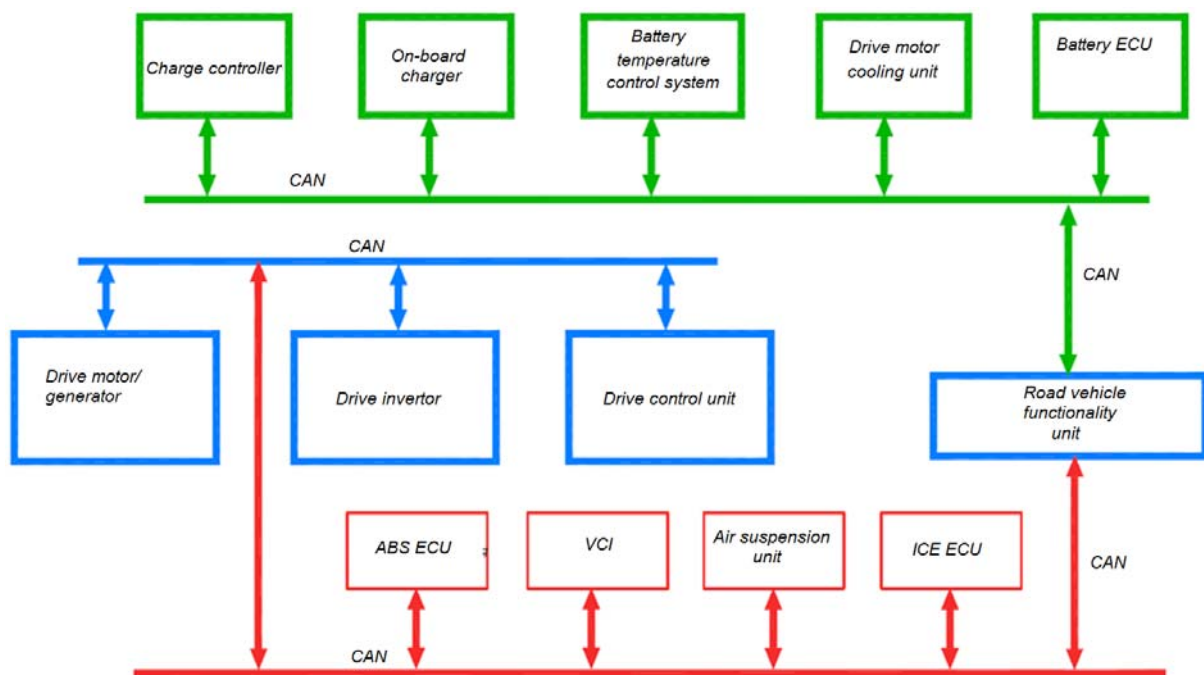


Figure 2. City bus electronic control system flow diagram.

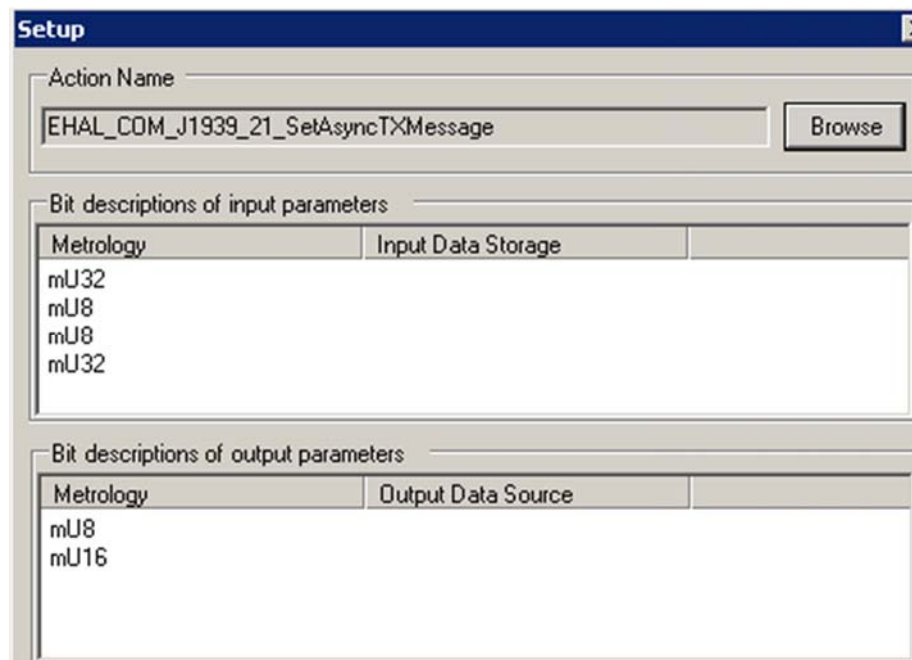


Figure 3. Called function description.

It is necessary to initialize receiving and sending CAN messages to get started (Figure 4).

It is required to submit the number of the parameter group (PGN) of the message (`c_canRxPgnBMUtoVCU_1`) and the bus number (`q_canNetGAZ_CAN`) from which it will be received when receiving the initialization function. The function of sending input parameters requires more: PGN messages, bus number, message priority (`c_canPriority_0x18`), an indication to a callback function (`c_NULL_POINTER`). The initialization function returns its result (0 - completed successfully, 1 - an error occurred), and an indication to the CAN message (RxHandler). Further work with the CAN message occurs at the returned pointer.

In the real-time system, the function of receiving and sending CAN messages is triggered with a frequency of 10 ms. Working with a CAN message consists of three subtasks: a counter (to receive and send messages with the desired frequency), work with a buffer (reading or writing), and clearing the buffer. Each subtask has its own priority (Figure 5).

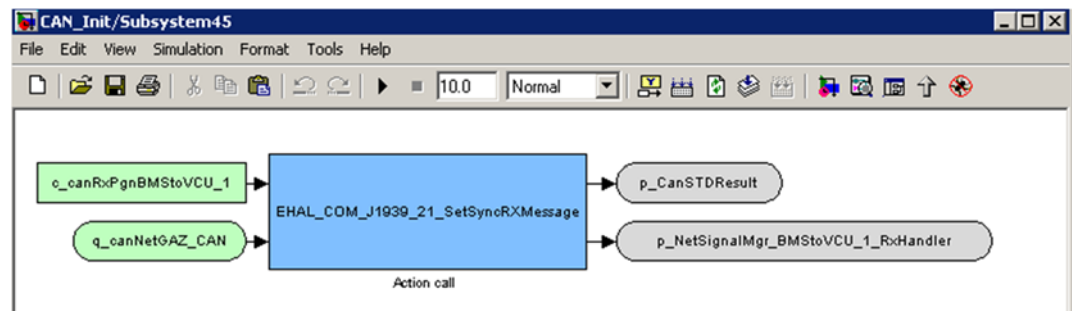
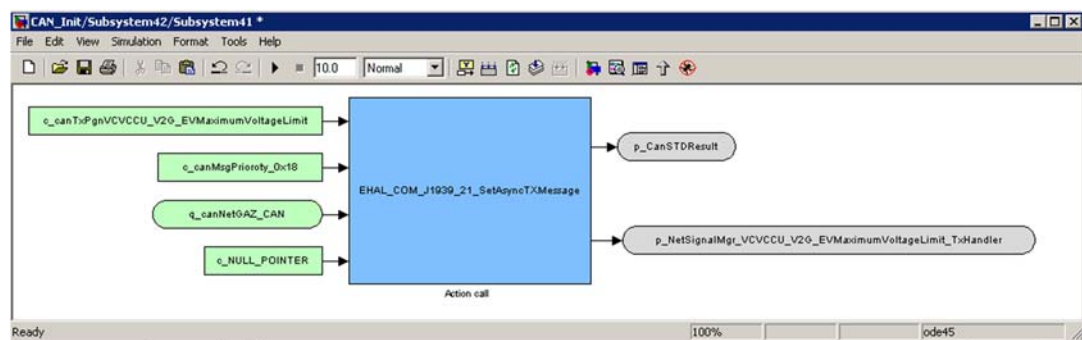
*a**b*

Figure 4. CAN messages initialization: *a* – receiving, *b* – sending.

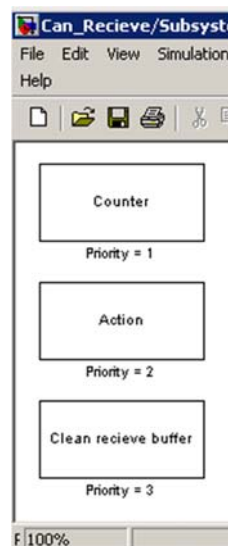


Figure 5. Subtasks call during work with CAN messages.

The counter implementation is shown in Figure 6. The $p_counter$ variable stores the current value of the counter, it is incremented by the value of $p_Time_t_Period_10msLv$ for each function call, while the top is limited by the limit (c_delay) $\neg$. The limit is the period of sending the desired CAN message.

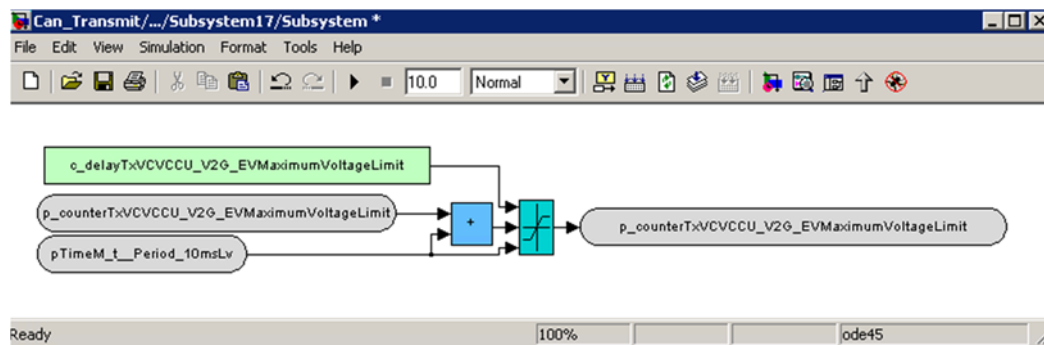
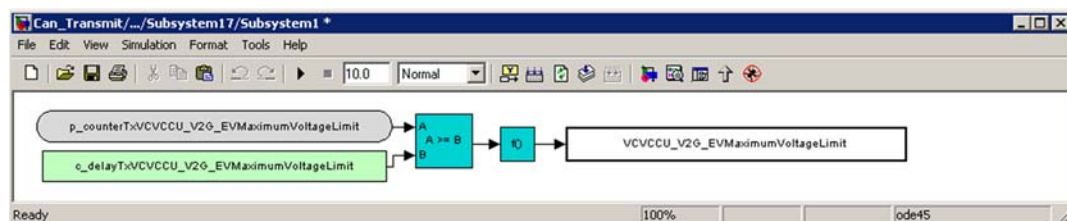
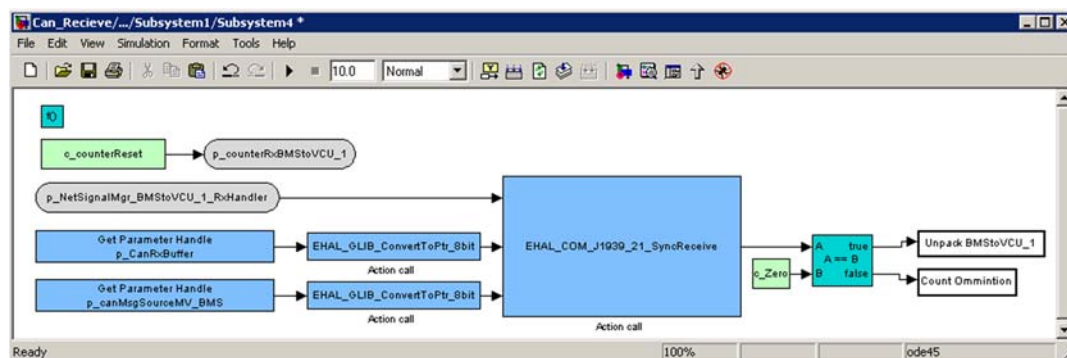


Figure 6. Counter realization.

It checks whether the counter reaches the desired value, and if it is reached, work with the buffer begins at each task invocation (see Figure 7a). When a message is received, the upper layer function is called EHALL_COM_21_J1939_SyncReceive, and if the function returns 0, then the received message is available for further work (see Figure 7b).



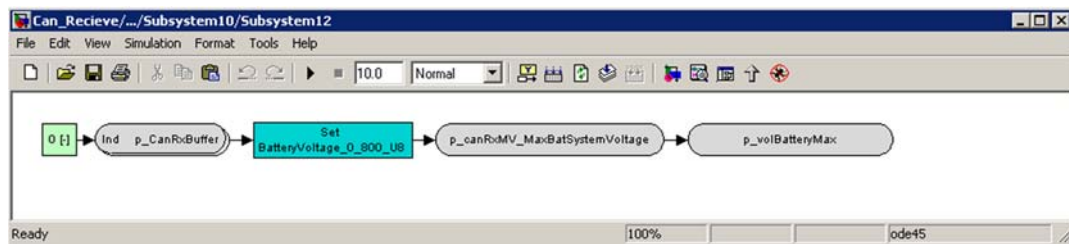
a



b

Figure 7. Receiving CAN message function: *a* – counter check, *b* – receipt function call.

A variable with the required metrology is required for the purpose of reading the required parameter from the CAN message correctly (Figure 8a). The parameter “Maximum battery voltage” considered as an example. The parameter “Maximum battery voltage” is contained in the 0th byte, has a length of 8 bits, and the resolution is 3.2 V per bit. A variable `p_canRxMv_MaxBatSystemVoltage` with the metrology `BatteryVoltage 0_800_U8` is created to store the parameter (Figure 8b, 8c). Further, the value of the variable `p_canRxMv_MaxBatSystemVoltage` is written to the variable `p_volBatteryMax`, which has a different metrology, and the conversion of values takes place in the program automatically.

*a*

3. Data properties	
Metrology	BatteryVoltage_0_800_U8 (3,2 [V])
Minimum value	0
Maximum value	816

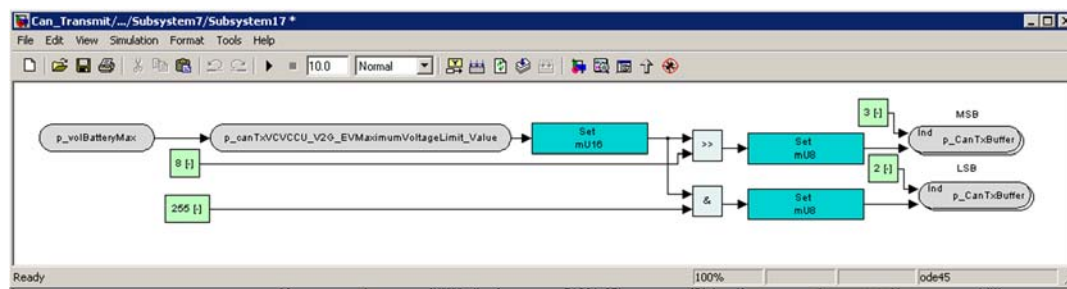
b

1. Description	
Name	BatteryVoltage_0_800_U8
Title	Напряжение батарей от 0В до 800В с шагом 3.2В
Comments	
Author	bmstu (04.12.2017)
Modifier	bmstu (11.01.2018)
2. Code generation	
Built-in type	8 Bit Unsigned Integer
3. Data properties	
Coefficients	32 / 10
Minimum value	0
Maximum value	816
Value step	3,2
Transfer coefficient	1 / 1
Units	bVoltage
4. Appearance	
Derived unit name	V
Format string	%3.2

c

Figure 8. Signal receipt: *a* – parameter reading from CAN buffer, *b* – metrology choices, *c* – metrology realization.

A reverse operation is applied to send a CAN message. The value of the variable `p_volBatteryMax` is written to the variable `p_canTxVCVCUU_V2G_EVMaximumVoltageLimit_Value` (Figure 9a). with `BatteryVoltage_S16` metrology (Figure 9b). The resulting variable `p_canTxVCVCUU_V2G_EVMaximumVoltageLimit_Value` is written to the buffer for sending via the CAN interface.

*a*

1. Description	
Name	BatteryVoltage_S16
Title	Напряжение батарей от -32768 V до +32768 V с шагом 1 V
Comments	
Author	bmstu (14.08.2018)
Modifier	bmstu (01.10.2018)
2. Code generation	
Built-in type	16 Bit Signed Integer
3. Data properties	
Coefficients	1 / 1
Minimum value	-32768
Maximum value	32767
Value step	1
Transfer coefficient	1 / 1
Units	bVoltage
4. Appearance	
Derived unit name	V
Format string	%5.0

b

Figure 9. Signal sending: *a* – parameter recording in CAN buffer, *b* – metrology realization.

As the result, the interface of the upper layer software allows one to get rid of the errors of converting quantities from one format to another by creating different metrologies.

The work with digital input and digital output of control system electronic unit is considered on the example of the operation of a high-voltage circuit contactor. The contactor has one control signal and one feedback signal. Feedback allows to control the current state of the contact, which is necessary because a powerful electric arc may strike and the contactor will stick when the contactor is opened under load. In normal condition, the contactor is open. In order to close the contactor, it is required to apply a voltage of 24 V to the control signal, and if the contactor successfully closes, a voltage of 24 V is also set at the feedback signal.

It is essential to configure a digital output to supply a control signal and a digital input to read the feedback signal to implement the algorithm in the program. The function `EHAL_IO_MGR_D_SetChannelState` is used to control the digital output (Figure 10a) and it takes two parameters as input: the output contact number is `q_nOutputUltrafastChargeContactor1` and the output state is `p_pinUltrafastChargeContactor1` (0 is off, 1 is on). The function `EHAL_IO_MGR_D_GetChannelState` is used for reading digital inputs (Figure 10b) and it takes one parameter as input: the number of the input contact is `q_nInputUltrafastChargeContactor1`, and has two output parameters: the state of the input is `p_inUltrafastChargeContactor1Raw` (0 - off, 1 - on) and the result of the function (0 - successful execution, 1 - error). The input signal has an unstable state due to bounce at the first moment of time. To get rid of it, the additional AntiBounce function is used (Figure 10c), at the output of which the filtered value of the digital input `p_stUltrafastChargeContactor1` is set.

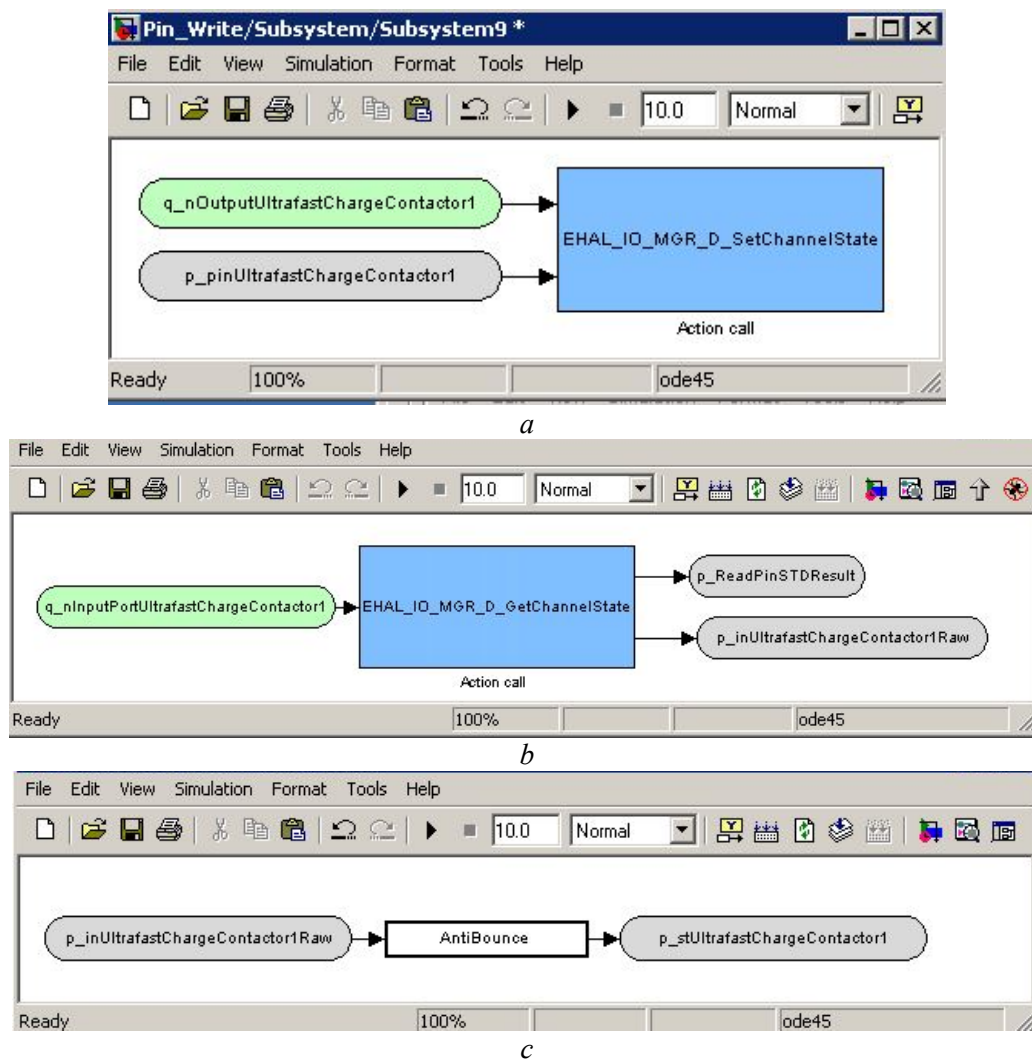


Figure 10. Functions for digital input/output operation:
a – digital output control, *b* – digital input read operation,
c – contact bounce screening.

The digital input read and digital outputs record are functions that are performed in a separate sub-task. The contactor functionality is usually implemented in another subtask, which also implements the main control flow of the system. Figure 11 shows the part of the algorithm that is responsible for working with the contactor.

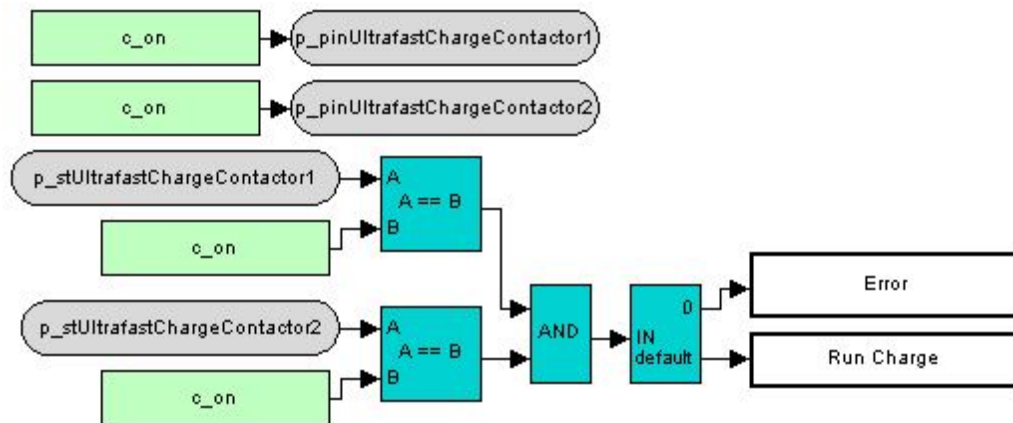


Figure 11. Power contactor operation algorithm.

A real-time operating system (RTOS) is implemented in the MSE Pro environment. RTOS allows implementing preemptive multitasking, i.e. the system operation in which several tasks are performed almost simultaneously due to the rapid switching between them. Distribution of the developed individual algorithms (receiving / sending CAN messages, operation of a power contactor, etc.) into tasks with a specific call period (10 ms, 100 ms, etc.) is as appropriate condition for the program correct operation. The implementation of the RTOS and an example of the inclusion of individual algorithms in the tasks are presented in Figure 12.

Consequently, upper layer software is developed that implements the new required functionality: receiving and sending CAN messages, digital outputs control, reading the status of digital and analog inputs, etc by means of visual programming tools in MSE Pro. Automatic C code generation is performed in the MSE Pro environment after working on upper layer algorithms has been finished. The compiler is used to extract the executable code (firmware) generated from the C code as a hex file. Having downloaded the new firmware to the control system electronic unit, originally intended for controlling the operation of the internal combustion engine, it can be used in another motor vehicle for other purposes, for example, as a controller of the electric bus upper layer.

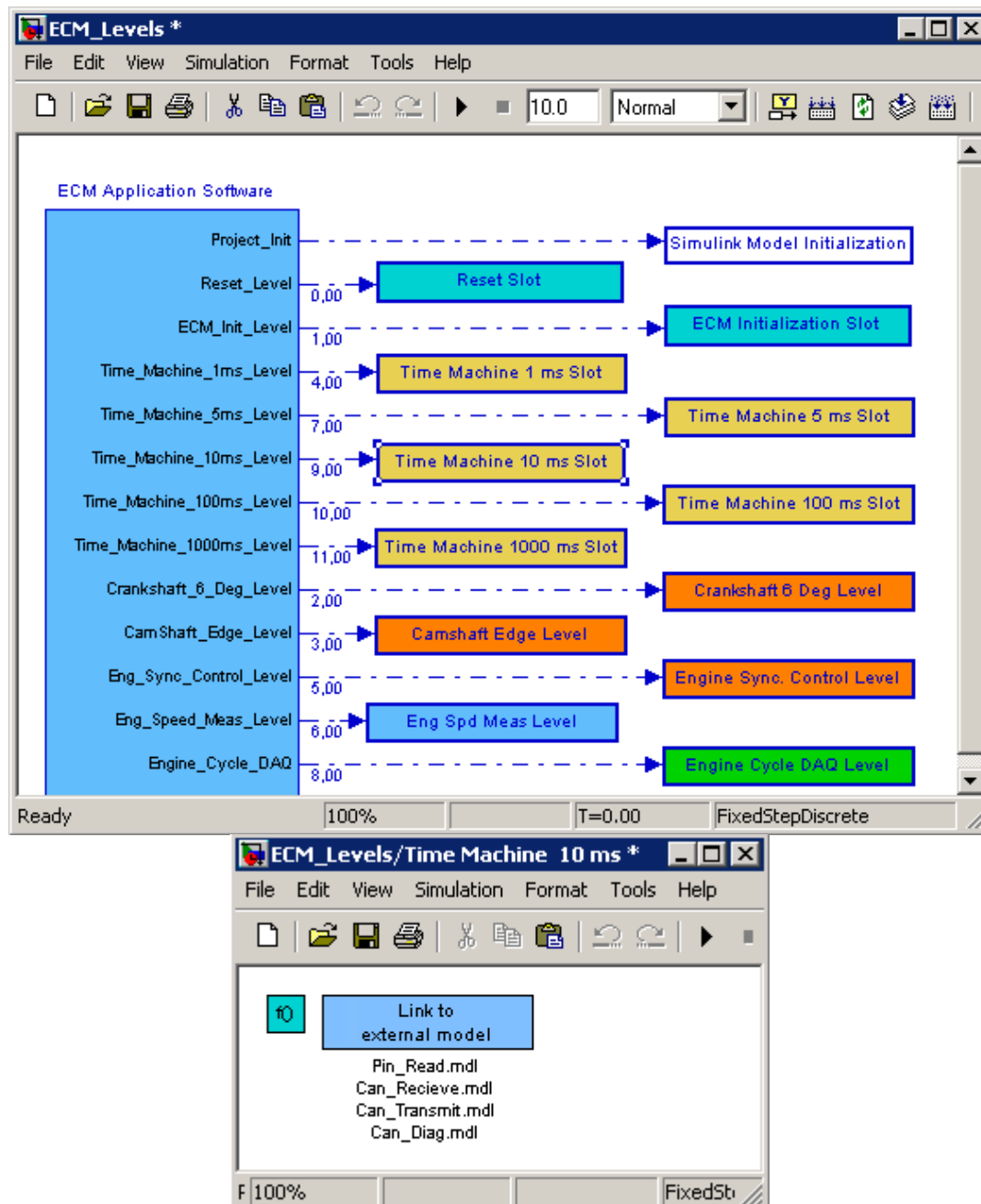


Figure 12. RTOS in MSE Pro environment.

The results of the preliminary tests of the vehicle with the installed control system electronic unit MD22 confirmed the possibility of using serial EBSUs, originally developed for the control system of the internal combustion engine as the ECU of the entire vehicle. This approach is much cheaper than ordering the development of a new electronic unit from Western manufacturers.

It should be noted that if a commercial control system electronic unit has more than 3 CAN interfaces it can be used for many different intelligent onboard security systems of the ADAS type for wide range data exchange with many onboard control system electronic units.

References

- [1] SAE J1939-71 Vehicle Application Layer
- [2] Buzunov N V, Kotiev G O and Miroshnichenko A V 2015 Improving the development of onboard information management systems of wheeled and tracked vehicles *Automotive Engineer Journal* **4**(93) 11-15
- [3] Keller A V, Gorelov V A, Vdovin D S, et al. 2015 Mathematical model of all-terrain truck *Proceedings of the ECCOMAS Thematic Conference on Multibody Dynamics* pp 1285–1296
- [4] Keller A V, Gorelov V A and Anchukov V V 2015 Modeling truck driveline dynamic loads at differential locking unit engagement *Procedia Engineering* Vol. **129** pp 280–287
- [5] Buzunov N V 2016 Analysis of «real time» plane nonlinear automobile motion model with wheel configuration 6x6 *Moscow State Technical University MAMI News* **3**(29) 2-8
- [6] Philippitskiy A V, Shedrin I V and Artyushkov A Y 2016 Development issues of software for working with an electronic vehicle control unit *Modern trends in the development of science and technology* **3-3** 120-123
- [7] Galin D A and Nazarkin A S 2016 Electronic control units programming *XLIV Ogarev reading materials of the scientific conference: 3 parts* 168-174 (responsible for the release P.V. Senin)
- [8] Butarovich D O, Kositsyn B B and Kotiev G O 2017 A method for developing an energy-efficient law to control an electric bus when driving on a city route *NAMI Proceedings* **2**(269) 16–27