УДК 629.33.028

V. G. MIKHAILOV

DATA TRANSMISSION WITH SIMULINK ON 6-DOF PLATFORM ON CAN BUS

Use of CAN BUS for data transmission in Real-Time mode with Simulink on control objects is considered (6-DoF a platform).

It is revealed that software of CAN_API.dll adapters, created in the Microsoft Visual Studio (MVS) does not work with TDM-GCC-64 Matlab/Simulink because of different approach in names of the dll functions according to the standard C ++ 11/17. Recompile by the developer of the adapter of its software (dll) in the TDM-GCC-64 environment under Windows is required.

It is established that CAN BUS considerably reduces modeling speed by 4.5 times. The way of information compression and fall forward of exchange twice due to byte-by-byte entering of two float values in the data field is offered. Use of identical values of identifiers is applied to two cylinders 6-DoF of a platform and the subsequent their division in the program microcontrollers of cylinders.

For implementation of a Real-Time mode in addition to compression it is offered to transfer data with the smaller frequency (quantization) by what a modeling clock period. It was considered that 6-DoF platforms reproduce frequency band to 10–12 Hz. The program of transfer/data exchange with Simulink on stand control devices with quantization is developed. Influence of parameter of quantization for the period of modeling is investigated. It is established that the Real-Time mode of modeling is provided in the range of parameters of quantization (chc=1/350–1/1000). Frequency of exchange with 6 cylinders at the same time corresponds to 230, 150 Hz.

Keywords: *Simulation modeling, vehicle, simulator, electroactuator, Matlab/Simulink, CAN BUS, 6-DoF platform.*

Now abroad, to reduce the time of development and development of designs of aircraft, vehicles, various control objects, simulation methods are increasingly being used [1, 2]. In the course of such modeling real control objects, actuation mechanisms and the human-operator are involved [1, 2]. An example of this are aviasimulators, semi-natural modeling of the vehicle at the stand with involvement of the driver with simulating of visualization of a road situation and influence real macro and a microprofile of the road (Fig. 1). All this significantly influences the mode of the movement and loading of the vehicle.

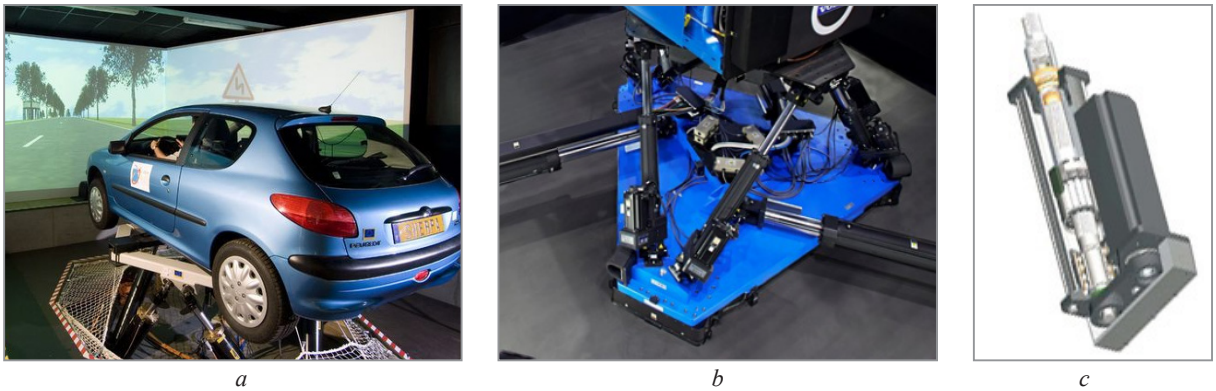
It allows to develop for the vehicle on models a large number of road situations and options of construction. Test them in bench conditions, fulfill ergonomics and considerably to reduce time of creation of new models of machines, without putting at risk of drivers and testers. And aviasimulator allows to simulate critical situations and to train pilots.

As executive elements in them electroactuators in the form of the mechanical cylinder with belt and worm ball drive and the electric motor with the block of electronic control are widely used now. The cost of the Chinese 6-DOF platform ($ 7800) is one-two orders lower than at hydropulsators [3]. According to documentation of MOOG [4] the simulation MB-EP-6DOF/40/10000 platform can provide acceleration of 13 m/s^2, the speed of 0,9 m/s with an amplitude of 0,73 / 0,81 m. Elektroaktuators of MOOG, Parker are used in simulators of firms Mercedes Benz, Volvo, Ford.

Electroactuators are widely used in industrial robots, exercise machines. At the same time on the range of frequencies (0–10/12 Hz) and the enclosed loadings (10–20 kN) electroactuators nevertheless are inferior to hydropulsators (0–30 Hz and 100–500 kN). Despite it electroactuators have the niche of application: robots, research stands, exercise machines, actuation mechanisms of management where big values are not required. Hydropulsators are more used for resource tests.

As simulars the Matlab/Simulink package in the environment of Windows 7–10 64-bit is generally used now. Most efficiently for modeling is creation of the program in language C in S-Function Builder Matlab/Simulink. Development of the last requires use of the compiler TDM-GCC-64.

An important point is ensuring compatibility of all software components on digit capacity and compliances of compilers and OS.



a *b* *c*

Fig. 1. Simulation modeling of the vehicle (*a*) in bench conditions and construction of electroactuator (*b, c*)

Difficult question is simultaneous implementation of modeling of the vehicle in Real-Time mode on Matlab/Simulink and transfer of data from it on control objects with a necessary speed (to 100 Hz for each 6 cylinders, actually 600 Hz) to provide an error of 1 % at a higher frequency of 10 Hz.

Exchange through binary files on network does not provide correct data transmission because of not synchronization of write processes and reading with such frequencies.

For data transmission it is reasonable to use the CAN BUS protocol (1024 kbps). It is widely applied in robots, an avia and automotive industry in management systems to data exchange between microcontrollers, thanks to the simplicity, reliability and speed now.

Feature of CAN BUS is use of the twisted pair cable as a cable (Fig. 2).

Data transmission is carried out by means of the frame containing both the service and transmitted data. The data field consists 8 character bytes. The number of the connected controllers is defined by value of a code of the identifier in a frame. In standard 11 bit option to 127.

Data transmission on CAN BUS requires existence of adapters which connect the computer equipment with control devices (Fig. 2) or existence them in the microcontroller. Now there were industrial adapters at data transmission rate to 10–12 Mbps. However their implementation restrains high cost (> $ 3000). The majority modern electro and hydroactuators are equipped with CAN BUS interfaces and vendors offer instructions and tools for developing of programs on language C.

Data transmission requires still the software for adapters, which differs at different vendors in a look different sale of the adapter. Vendors, as

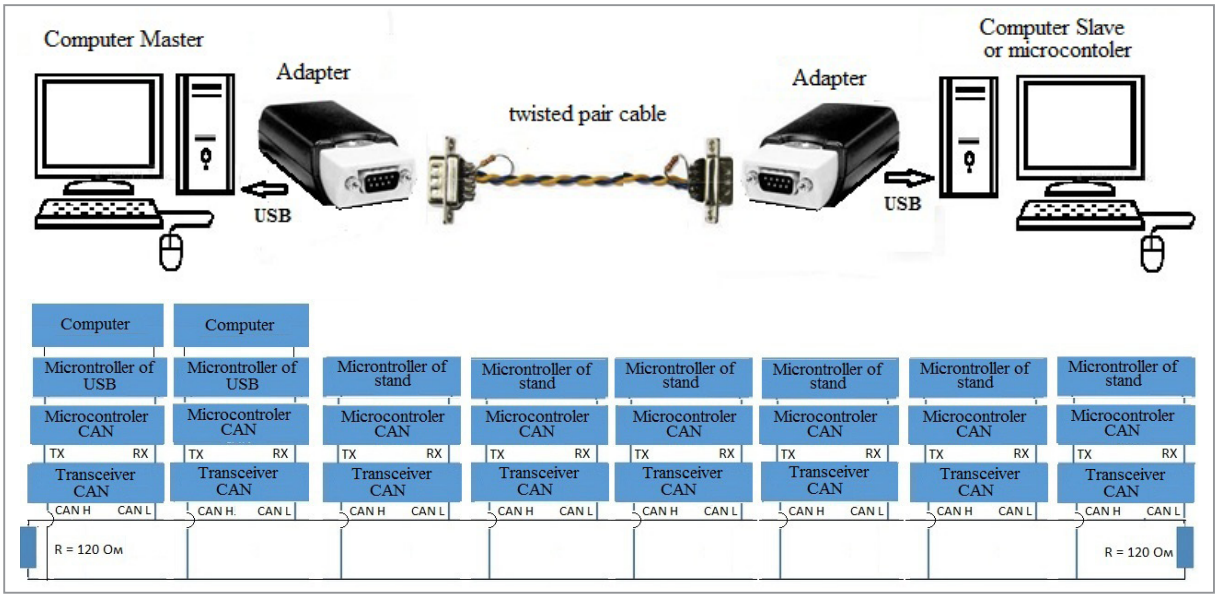


Fig. 2. The offered general transmission scheme of data with Simulink on the stand

a rule, offer software in the form of CAN_API. dll created in MicroSoft Visial Studio (MVS) or Linux without providing source texts.

By testing it is established that CAN_API.dll software, compiled in MVS do not work with TDM-GCC-64 Matlab/Simulink because of different approach in names of the dll functions according to the standard C ++ 11. To fix this problem recompile of dll in the TDM-GCC-64 environment under Windows which only the dll developer can execute is required.

For an exception of decline in the performance of the computer and the adapter it is offered to transfer data not to a modeling clock period, and with quantization is one-two orders less when ensuring acceptable transmit frequency of data on the stand.

The limiting factor of CAN BUS is cable length: at $l < 40$ m can be provided transmission rate of 1–3 Mbps, and already at 100 m only of 500 kbps.

Now in the Matlab/Simulink R2019b version there were components allowing to create communication on CAN BUS through Virtual Channel (500 Kbps), Vehicle Network Toolbox. They demand separate purchase and use of expensive adapters and the special equipment of certain firms, such as, Vector, Kvaser, PEAK-System and others and their software.

Unfortunately, there are no publications on implementation of this solution for stands with a large number of cylinders. Not clearly how to change in the program way the CAN parameters where tabular configuration of the channel is used, how to carry out quantization and whether there will be enough offered exchange rate of 500 Kbps with for 6 actuators and 3 computers (it is actually received 45 Kbps with on an object). And as it will influence speed modeling.

The purpose of this work is the research of use of CAN BUS for data transmission on electroactuators of 6-DoF platform via the S-Function Builder module and implementation of modeling in Matlab/Simulink in Real-Time mode.

1. Choice of the adapter and development tools

The choice of the adapter is the key moment for modeling. Basic data is the possibility of implementation of the frequency of 100 Hz on each cylinder from 6, used in a platform 6-DOF. In this case it is possible to provide an error of 1 % at a higher frequency. Proceeding from it the adapter of TITAN ELECTRONICS INC was selected, SN #T16820100, Taiwan. This adapter is the most optimum, economical and competitive in terms of price and quality [8]. It is the industrial adapter realizing data transmission with a speed up to 3000 Mbps. Other manufacturers usually have speed up to 1000 Mbps and the price is higher or dealers do not provide software for the adapter. Besides, this firm met requirements of me, free of charge provided two adapters and recompiled software under TDM-GCC-64 also helped at the initial stage with CAN BUS.

I am grateful to it for. It removed a problem of incompatibility of the created program with use of CAN_API.dll for Matlab/Simulink.

Features of use of this adapter are given in appendix A.

For modeling the Matlab/Simulink R2015b package was used (ode5 Dormand-Prince, $t = 0{,}001$ s). It works twice quicker than R2018b–R2020b. Perhaps oversaturation is their new functions the reason of deceleration.

Its choice is caused by the fact that it twice quicker than R2018b. Perhaps the reason for the latter is the oversaturation of its new functions, which are not needed for this task.

In the beginning working of of the program was carried out on the old 2-core computer (2.5 GHz, RAM of 8 Gbytes) on Windows 7 64-bit. For development of the program the compiler TDM-GCC-64 (tdm64-gcc-5.1.0–2.exe) was used. Its installation and configuration is described in work [6]. Windows 10 requires still installation of a way on C:\TDM-GCC-64\bin via the Set Path button on the Matlab panel.

Modeling was carried out with use of real perturbation of the road [9] in the form of an array of 96000 records of the float type in electronic memory. By means of the module of S-Function Builder accomplishment of logical actions on switching of the check point, calculation of traction dynamics of the vehicle was carried out. More perfect models providing an error of 10–15 % taking into account a range [6], [7], [9], [10] were used. Besides in separate Simulink blocks vibrations, controllability of the car were modelled. Visualization was carried out by means of the Raspberry 3B+ minicomputer.

The choice of such approach was caused by need of ensuring maximum speed of modeling in Real-Time mode.

2. Implementation of exchange of information

The realization of modeling with data transmission on CAN BUS was enabled by means of the skeleton diagram (Fig. 3) and flowcharts (Fig. 4) in which 18 integrators are involved.

For modeling the computer was used (AMD Rysen 5 2600, 6 core 3,4/4 GHz, RAM of 16 GB DDR-4 2666 MHz, the drive M2. SSD of 2500 Mbps, MB ASUS PRIME B450 PLUS, t = 0,0001 c) and a detail from the game console. Data transmission in an asynchronous mode. Reception of messages was carried out on the old 2 core AMD computer of 2,5 GHz, RAM of = 8 GB DDR-2 400 MHz, MB ASUS M2-VM which imitated the actuation mechanism. Working of of modeling at this stage was carried out on a model desktop sample.

For data transmission with Simulink on stands the program on C is developed for S-Function

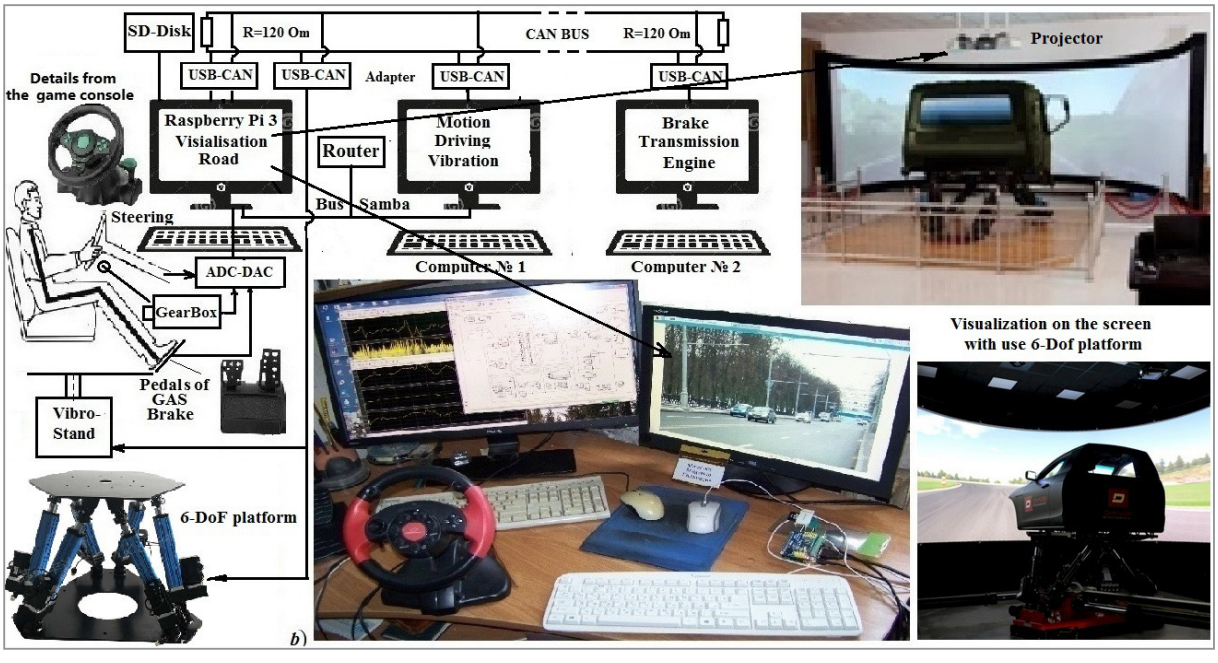


Fig. 3. Scheme of implementation of simulation modeling of the vehicle

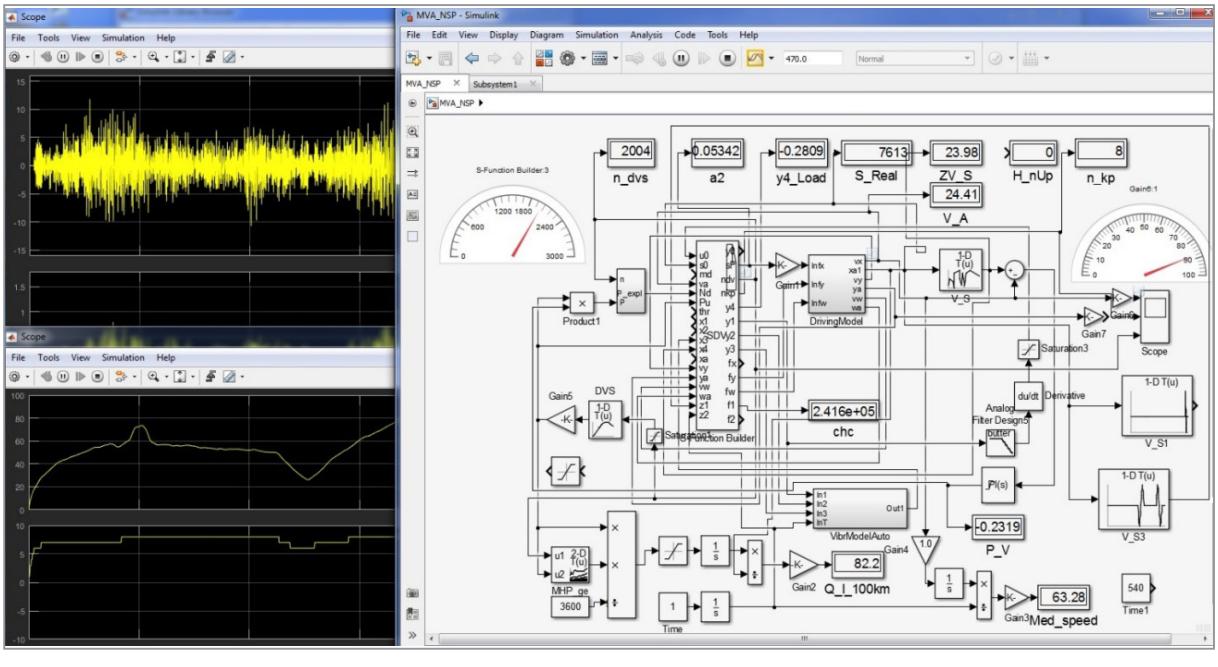


Fig. 4. Process and flowchart of modeling of the movement, fluctuations and controllability of the vehicle

Builder which total amount made 1372 lines of a source text. From them 850 lines are the share of the service S-Function Builder parameters. Initial debugging of transfer and data reception on CAN BUS was carried out on the srd.c program given below which will help you to create the program. Its void srd function (struct ds*ps, struct dr*pr) is used in the general program for S-Function Builder.

The functions used in the program and configuration of the adapter are in detail described in [8]. In the directory with the wrdc.c program the CAN_API.dll and CAN_API.h files should be set. That to compile it enter the team into TDM-GCC-64 gcc wrdc.c. As a result the executable file of a.exe will turn out.

In the beginning information for 6 actuators is brought in a data structure of struct ds sf. As void srd () data compression is realized: entering in the field of 8 bytes of SendMSG.Data of [0] two float values of data is byte-by-byte instead of one (this fragment is highlighted in yellow color). Actually it is data transmission by blocks on two actuators at the same time.

Further as void sd (struct ds*sp, struct dr*rp) realized data compression: entering in the field of 8 bytes of SendMSG.Data of [0] two float values of data is byte-by-byte instead of one (the fragment is highlighted in yellow color).

Actually it is data transmission by blocks on two actuators at once having the identical identifier.

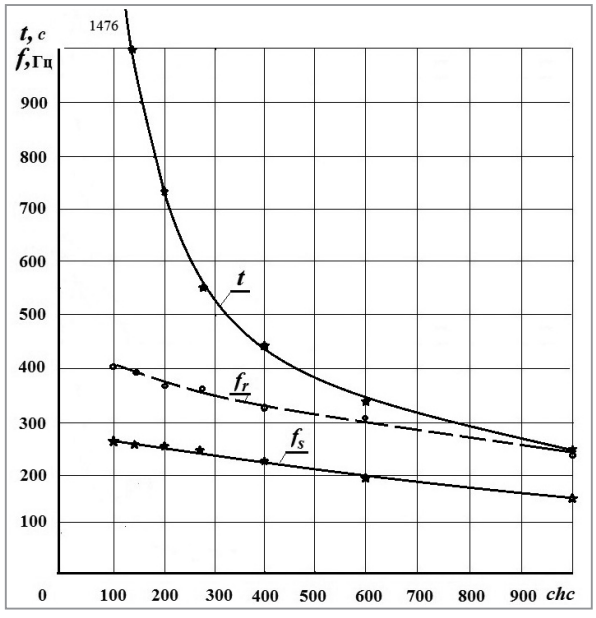
And in the program of the controller of the actuator their program division is made. It allows to increase data exchange productivity twice (thereby to 6 Mbps).

3. Results of modeling

Data transmission was carried out with frequency on a cycle timer *of chc* which was nullified at achievement of the maximum value. The cycle timer *of chc* defines quantity of clock periods of modeling after which there is a data transmission on 6 electroactuators on CAN BUS. It allows to reduce expenses of time for information transfer. Quantization in 100 Hz is quite enough for reproduction by the stand of the higher frequencies of 10 Hz with a margin error 1 %.

Results of modeling are given in the Fig. 5.

Basic time of modeling without CAN BUS on 6 core computer makes 105 s at real process



t – time of modeling, f_s – the frequency of sending of messages, f_r – the frequency of reception of messages

Fig. 5. Influence of parameter of counter/quantization on the frequency of sending and reception of messages and modeling time

of 470 s. According to received data CAN BUS use considerably slows down process, increasing modeling time (by 4,5 times at *chc* = 350). The nature of frequency change of sending and reception of messages has an appearance of small nonlinearity. At reduction *of chc* time of modeling *of t* sharply increases and it defines the choice *of chc*. At the parameter of quantization *chc* = 350 the modeling Real-Time mode is reached, the transmit frequency of data at the same time makes 230 Hz. It is found out that the Real-Time mode can also it is reached by selection of a step of integration, processor frequency in the UEFI BIOS motherboard, connection of other subsystems, functions, use of newer Matlab/Simulink R2019b version working more slowly, failure from data compression; use of more difficult space model of the vehicle, application of a synchronous transfer mode.

At increase in parameter of quantization from 350 to 1000 the transmit frequency of data changes from 230 to 150 Hz. It quite enough for control of stands (100 Hz) also allows to realize further modeling at complication of model of the vehicle.

Testing of weaker 2-core computer with CAN BUS showed that its productivity is not enough for implementation of a Real-Time mode: its speed of modeling is 8.7 times less than at modern.

Frequency of data reading turns out by 1.5 times quicker than their record at the expense of a modeling process exception.

Results of modeling demonstrate that only more modern computers with CPU~4 GHz and Windows 10 can provide data transmission on CAN BUS with Simulink in Real-Time mode for difficult model. It gives diversities of calculations of subsystems between computers when using CAN BUS for a solution of complex challenges and expediency of full transition to CAN BUS despite its strong braking.

It is planned to pass completely to CAN BUS with circuit implementation in Fig. 4 with an exception of Samba network.

In a type of a difficult economic situation in Republic of Belarus it was not succeeded to realize option with the vibrostand on 6-DoF platform with the turning image on the screen (Fig. 3 on the right above).

And at development of methods of simulation modeling in a combination to bench tests of the vehicle on running drums (according to the original scheme offered by me) it is possible to investigate each element of construction and to optimize it. And an important role is played here by use of CAN BUS for information exchange.

Conclusion

1. Use of CAN BUS for data transmission in real time with Simulink on control devices on an example 6-DoF platform is considered.

2. It is established that CAN_API.dll adapter software, compiled in MVS does not work with TDM-GCC-64 because of different approach in names of the dll functions at the compiler MVS. To fix this problem recompile of dll in the TDM-GCC-64 environment which only the dll developer can execute is required.

3. The way of information compression and increase in an exchange rate twice due to byte-by-byte entering of two float values in the data field is offered. Use of identical values of identifiers is applied to two cylinders 6-DoF of a platform and the subsequent their division in the program of microcontrollers of cylinders.

4. It is revealed that CAN BUS considerably reduces modeling speed by 4,5 times. Therefore providing a modeling Real-Time mode with CAN BUS requires use of more high-speed computer (CPU ~ 4 GHz on Windows 10) and a certain mode of quantization with a modeling clock period.

5. The program of transfer/data exchange with Simulink on control devices of stands with quantization of chc=1/350–1/1000 from a modeling clock period is developed. It allows to realize a Real-Time mode of modeling and frequency of exchange 230, 150 Hz.

6. The optimum choice for implementation of data transmission with Simulink on stands with electroactuators on CAN BUS is use of TITAN ELECTRONICS INC adapters. They allow to realize the necessary frequency of exchange more than 100 Hz for 6-DoF platform.

REFERENCES

1. Mercedes-Benz Innovation Vehicle Developing /https://www.mercedes-benz.com/en/mercedes-benz/next/advanced-engineering / [Electronic resource / Electronic resource] / Access mode mode: 22.07.2018.
2. **Emanuele Obialero** A Refined Vehicle Dynamics Model for Driving Simulators // Charhalmers University of Technology / Göteborg, Sweden 2013. Master's thesis, P. 120.
3. Customized Flight Simulator Vehicle Driving Simulation 6 Dof Motion Base Platform/https://szfdra.en.ìade-in-china.com/product/lsymBGZJbIcn/China-Customized-Flight-Simulator-Vehicle-Driving-Simulation-6-Dof-Motion-Base-Platform.html / [Electronic resource / Electronic resource]//Access mode: 22.07.2018.
4. Electric Simulation Table /https://www.moog.com/products/simulation-tables/electric-simulation-table.html/ [An electronic resource / Electronic resource]/ Access mode: 08.09.2019.
5. Troubleshooting and Limitations Compiling C/C ++ MEX Files with MinGW-w64 https://nl.mathworks.com/help/matlab/matlab_external/compiling-c-mex-files-with-mingw.html.//Electronic resource / Electronic resource]/Access mode: 08.09.2019.
6. **Mikhailov, V.G.** Use of S-Function Builder Matlab/Simulink / Systems analysis and applied information science – 2018, No. 4. P. 57–64 (on rus).
7. **Mikhailov V. G.** About some approaches of modeling of the vehicle on simulators / Systems analysis and applied information science – 2019, No. 3. P. 29–35 (on rus).
8. Usb-can user's manual 2017–07–06 edition
9. https://insat.ru/upload/iblock/da3/titan_USB-CAN%20Manual.pdf / [Electronic resource / Electronic resource]/ Access mode: 08.09.2019.
10. **Mikhailov, V.G.** Receiving and use of a uniform array of a longitudinal profile and microprofile of the road for modeling of the CU//journal of automobile engineers No. 2, 2018, P. 4–7 (on rus).

11. **Mikhailov V. G.** About oscillatory model of the truck / V. G. Mikhailov, D. V. Mishuta,//Automotive industry-2016, No. 7. P. 23–27 (on rus).

Поступила *14.10.2020* | *После доработки* *20.02.2021* | *Принята к печати* *01.03.2021*

Mikhailov Vladimir Georgievich.
The specialist in the field of automotive industry, tests of suspenders, frames, pneumatics, hydraulics, strain-gaging, CALS/PLM development of systems (PDM, ERP) on Oracle, developments of programs on C/C ++, Java, microcontrollers, (Arduino/Raspberry), modeling of dynamic systems in the Matlab\Simulink package, estimates of the intense deformed status in the ANSYS package.

Михайлов Владимир Георгиевич.
Специалист в области автомобилестроения, испытаний подвесок, рам, пневматики, гидравлики, тензометрирования, разработки систем CALS/PLM (PDM, ERP) на Oracle, разработки программ на C/C++, Java, микроконтроллеров, (Arduino/Raspberry), моделирования динамических систем в пакете Matlab\Simulink, оценки напряженно-деформированного состояния в пакете ANSYS.
E-mail: sapr7@mail.ru

APPENDICES

Appendix A: Features of use of this adapter

For CAN BUS work it is necessary to install the USB-CAN CDM21228_Setup.exe driver in the beginning. Define what COM PORT is connected (for example, COM3). Then to start in the command line of **CAN_BAUDRATE_SET.exe COM3**. The message should be received

Searching …
Find COM PORT: COM3
Setting every baud rate to 3Mbit …
Set baud rate Success

testimonial of successful installation. Then it is required to disconnect an USB cable from the computer and through 5 second from again it to connect or to switch off and turn on the computer.

***Note**. In some cases to set CAN_BAUDRATE_SET.exe on Windows 10 64 bit it is necessary to set Properties of this file in the Compatibility mode, with noted ticks, as shown in Figure A.1. Then to execute the above-stated command.*

Figure A.1. Installation of the mode of compatibility for CAN_BAUDRATE_SET.exe

Appendix B: Source text of the program of data transmission for CAN BUS

```
#include <stdio.h>
#include <stdlib.h>
#include “CAN_API.h”
#include <windows.h>
//————————————
struct ds {
float a1;
float b1;
float a2;
float b2;
float a3;
float b3;
};
struct ds sf; // struct send data
struct dr {
float a1;
float b1;
float a2;
```

```
float b2;
float a3;
float b3;
};
struct dr rf; // struct read data
double stm=469.997;
int ch;
typedef TCAN_HANDLE (*FNPTR1)(CHAR *, CHAR *, CHAR *, CHAR *, void *, DWORD);
typedef TCAN_HANDLE (*FNPTR2)(TCAN_HANDLE);
typedef TCAN_HANDLE (*FNPTR3)(TCAN_HANDLE, CAN_MSG*);
typedef TCAN_HANDLE (*FNPTR4)(TCAN_HANDLE, CHAR *);
HMODULE hMod;
TCAN_HANDLE Handle;
TCAN_STATUS Status;
//————————-Echange data
void sd(void)
{
byte *yf, *yf1, *yf2;
int q, nbz;
long j;
FILE *fp;
CHAR *ComPort = "COM4";
CHAR *szBitrate = "1000";
//"9216000";
CHAR *acceptance_code = "1FFFFFFF";
CHAR *acceptance_mask = "00000000";
VOID *flags = CAN_TIMESTAMP_OFF;
DWORD Mode = Normal; // a debug mode "LoopBack "
FNPTR1 CAN_Open;
FNPTR2 CAN_Close;
FNPTR2 CAN_Flush;
FNPTR3 CAN_Write;
FNPTR3 CAN_Read;
FNPTR4 CAN_Version;
FNPTR2 CAN_Status;
char version[10];
CAN_MSG SendMSG;
CAN_MSG RecvMSG;
// Handle = –1;
Status = 0;
nbz=0;
j=0;
//————————-
CAN_Open = (FNPTR1) GetProcAddress(hMod, "CAN_Open");
CAN_Close = (FNPTR2) GetProcAddress(hMod, "CAN_Close");
CAN_Flush = (FNPTR2) GetProcAddress(hMod, "CAN_Flush");
CAN_Write = (FNPTR3) GetProcAddress(hMod, "CAN_Write");
CAN_Read = (FNPTR3) GetProcAddress(hMod, "CAN_Read");
CAN_Version =(FNPTR4) GetProcAddress(h-Mod, "CAN_Version");
CAN_Status = (FNPTR2) GetProcAddress(h-Mod, "CAN_Status");
RecvMSG.Flags = CAN_FLAGS_STANDARD; //EXTENDED;
RecvMSG.Size = 8;
//————————————
if (ch==0)
Handle = CAN_Open (ComPort, szBitrate, acceptance_code, acceptance_mask, flags, Mode);
//– Example read data from RecvMSG.Data
RecvMSG.Id = 0x001;
memset (version, 0, sizeof (char) * 10);
Status = CAN_Flush (Handle);
Status = CAN_Version (Handle, version);
for (j=0; j<60000; j++) {
Status = CAN_Read (Handle, &RecvMSG);
if (Status == CAN_ERR_OK) {
switch (RecvMSG.Id) {
case 1: yf=(byte *)&rf.a1; for (q=0; q<8; q++) *yf++=RecvMSG.Data[q]; break;
case 2: yf=(byte *)&rf.a1+8; for (q=0; q<8; q++) *yf++=RecvMSG.Data[q]; break;
case 3: yf=(byte *)&rf.a1+16; for (q=0; q<8; q++) *yf++=RecvMSG.Data[q]; break;
default:
break;
}
fp=fopen("tc.txt","at");
fprintf(fp,"Handle=%d ID=%X Status=%d%.4f%.4f%.4f%.4f%.4f%.4f\n", Handle, RecvMSG.Id, Status, rf.a1, rf.b1, rf.a2, rf.b2, rf.a3, rf.b3);
fclose(fp);
}
}
//– End Example read data from RecvMSG.Data
/*
//————————————
//————————- Example send data into SendMSG.Data
for (nbz=0; nbz<3; nbz++) {
yf=(byte *)&sf.a1+8*nb;
```

```
for (q=0; q<8; q++)
SendMSG.Data[q] = *yf++;
//————————————————-
switch (nbz) {
case 0: SendMSG.Id = 0x001; break;
case 1: SendMSG.Id = 0x002; break;
case 2: SendMSG.Id = 0x003; break;
default:
break;
}
memset (version, 0, sizeof (char) * 10);
Status = CAN_Flush (Handle);
Status = CAN_Version (Handle, version);
Status = CAN_Write (Handle, &SendMSG);
if (nbz==2)
fp=fopen("tcb.txt","at");
fprintf(fp,"Handle=%d  ID=%X  Status=%d%.
4f%.4f%.4f%.4f%.4f%.4f\n", Handle, RecvMSG.
Id, Status, rf.a1, rf.b1, rf.a2, rf.b2, rf.a3, rf.b3);
fclose(fp);
//– End Example send data into SendMSG.Data
*/
if (ch==3999)
Status = CAN_Close (Handle);
} //————————————-End sd()
int main(void)
{
float f1, f2;
int r;
long j;
hMod = LoadLibrary ("CAN_API.DLL");
if (hMod==NULL) {
printf ("LoadLibrary failed\n");
}
for (ch=0; ch<4000; ch++)
sd();
FreeLibrary(hMod);
printf ("Test finish_FreeLibrary\n");
return 0;
}
```