Computer-Aided Locomotive Simulator

Maciej Witek[a,*]

[a] *State Higher Vocational School in Tarnow, Mickiewicza 8, 33-100 Tarnów, Poland*

Article history:
Received 6 June 2018
Received in revised form
22 June 2018
Accepted 24 June 2018
Available online 27 June 2018

Abstract
This article is an application note of hardware implementation the train simulator driver stand based on own electronic devices and open software platform MaSzyna. Main electronic controller is based on Atmel ATMega2560 8-bit microcontroler. This type of simulators can be used to training train drivers. This is confirmed by train drivers who tested the simulator and teachers of the profession. The developed communication protocol has been used for several locomotive pulpits projects.

Key words: Train simulator, ET22

Introduction

Nowadays, when computers and embedded systems can be found in almost every branch of industry, railway traffic is still controlled by human. Modern locomotives are often equipped with ETCS (European Train Control System) also works in ERTMS (European Rail Traffic Management System). They are helpful for the driver, and do not exclude his presence in the cabin of the driven vehicle. The authorities of railway transport companies pay more attention to the need for training and checking the competence of their employees. The use of simulators for training purpose is already standard in some European Union Countries. In Poland, the statutory obligation to train and verify train drivers on simulators will come into force on 1st January 2019 [4]. In spite of this, some PKP Intercity (Polish State Railways) companies use their own simulators for already several years [5]. On the Polish market, there are available mainly commercial solutions and there is a lack of technical or scientific papers on this subject. The world literature reports only few papers devoted to this issue. Authors of the first of them [1] used the actual panel from the locomotive, to which they connected the LM3S9D92 microcontroller and all other analog and digital elements. Their simulator works well with 2 software platforms: *Railworks TS2013* and *Microsoft Train Simulator 2004*. Authors of another paper [2] built operator panel based on NXP K60 microcontroller. Their model works with *Open Simulator* software, available with open sources. In both cases, HID (Human Interface Device) was made based on real parts of locomotive. In the paper, locomotive simulator which is made from ET22 parts is described. ET22 was the most popular locomotive in Poland and it is still used by – freight transport company – PKP Cargo. In prototype of the simulator elements of the ET22, that consuming a lot of current, were exchanged for low power electronic parts. It was needed especially for mechanical gauges with inductors. The operator panel made as part of the work cooperates with the Polish software *MaSzyna* [3]. It is open source application which provides many possibilities of locomotive control on the screen. The application communicates with mechanical control panel by microcontroller via Virtual COM Port.

Construction of the simulator control panel

In order to make the storage and transport of the desktop easier, the author decided on a modular design of the desktop. The desktop can be divided into 5 parts that are comfortable during transport:

- Main module – the longest element with a power supply and main control module, equipped with a panel with lighting and heating switches (vertical panel), electrical meters and manometers
- Hasler Bern RT9 – registering speedometer mounted to the main module
- Brakes module – main brake valve, locomotive brake valve and the siren valve: high and low tone
- Travel adjuster module – main drive controller, drive direction change and decreasing of the excitation field of truction motors (shunting)

The choice of the simulator (software) was influenced by the cost of its purchase, the realism of the simulation (mainly the mapping of motion physics) and the control enabling the bi-directional data exchange between the hardware and the software.

Commercial software such as *Microsoft Train Simulator*, *Auran Trainz* or *Railworks* provides a high level of realism and generated image. The inability to make changes in the program code, hence changes in the method of controlling and obtaining indications of controls and meters excludes the use of this type of software. The solution of this problem is to use open source

*Corresponding author: m_witek@pwsztar.edu.pl

software for simulation. Programs such as the *Railway Vehicle Simulator MaSzyna* or *Open Rails* have a high level of realism and enablemapping physics and graphics. These projects are developed by railway enthusiasts, thanks to which new routes are created (both real and fictitious) and rolling stock (locomotives, draisines, wagons).

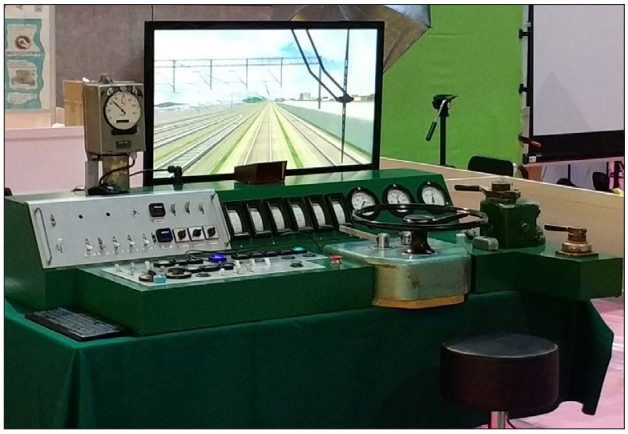

Figure 1. Photo of the ET22 locomotive simulator

The simulator (part of the cabin along with the software – see Fig. 1) enables faithful devotion to the realities of driving this series of locomotives. Appearance and placement of items on the desktop reflect the true distribution of accessories in the driver's cab.

Hardware interface

The hardware interface consists of 4 elements, powered from a common power supply:

– Main controller – is responsible for communication with the computer, reads the position of the brake taps, buttons and switches on the desktop, controls the electrical meters and controls on the desktop, connects and controls other modules;
– Pressure gauge controller – control of ACG (Air Core Gauge) equipped with potentiometers for calibration of meters;
– Speedometer controller – it controls a three-phase, permanent-magnet synchronous motor (PMSM) that drives the recording speedometer (Hasler Bern RT9);
– Adjuster controller (driving, direction and shunting) – reading the current position of adjusters

The main controller (Fig. 2) is based on the ATMega 2560 microcontroller, communicating with the computer via USB thanks to the FT232RL chip. Brake taps are connected to analogue inputs – calibration is available from the PC software level. Due to the large number of buttons and switches that had to be derived, the 74HC165 (parallel-in/serial out shift register) and 4094 (shift-and-store register) chips were used as the I/O expansion modules. The function of analog outputs is performed by 16-bit PWM outputs.

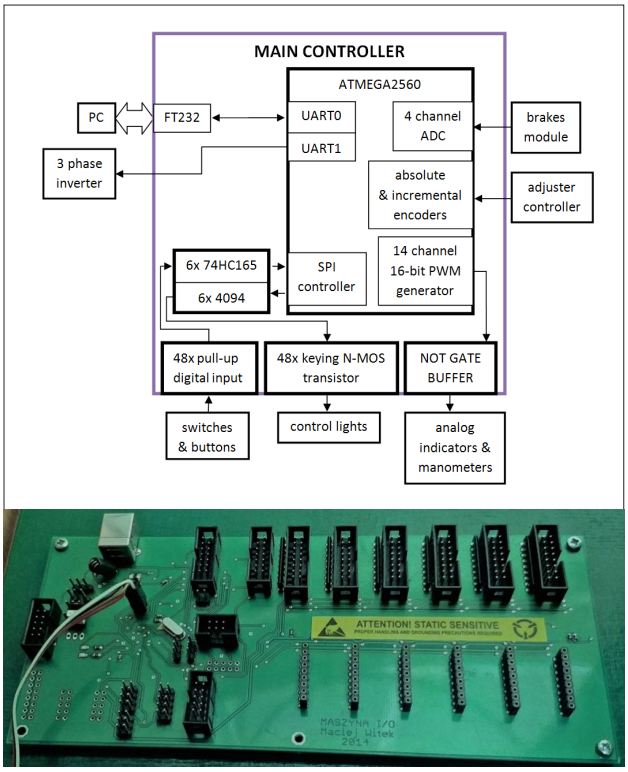

Figure 2. Photo and simplified scheme of main controller

The pressure gauge driver (Fig. 3) was built on CS8190 chip – a single ACG controller with an analogue input. The TL071 operational amplifier operates in the configuration allowing adjustment of gain and DC offset.

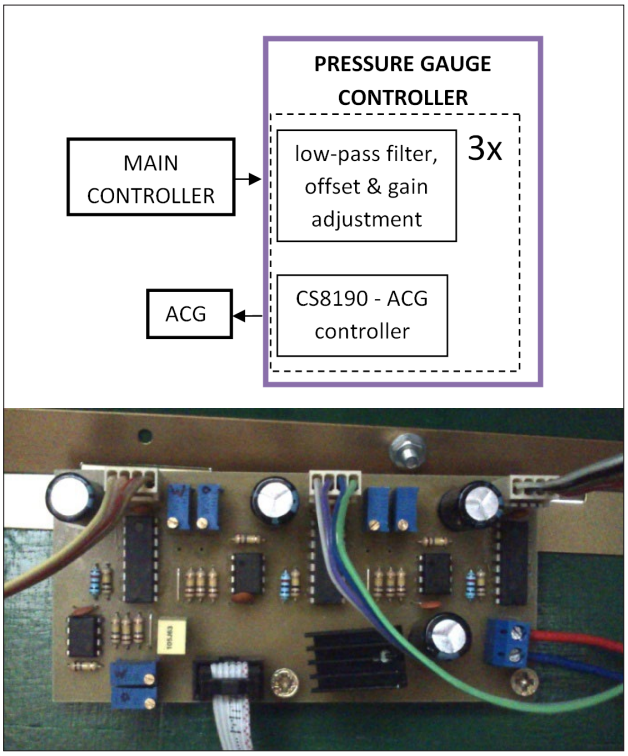

Figure 3. Photo and simplified scheme of pressure gauge controller

Speedometer controller (Fig. 4) is a dedicated three-phase inverter, which communicates with the main controller via UART (Universal Asynchronous Receiver-Transmitter) bus. Power output is the quadruple half-bridge H built on the L6205D chip. Control and communication with the main module is provided by the ATMega168 microcontroller. The microcontroller controls the H bridges using PWM (Pulse Width Modulation) signals about 62,5 kHz (see Fig. 5). If we average the output signal we get trapezoidal waveform. Phase shift between phases is 120 degree.

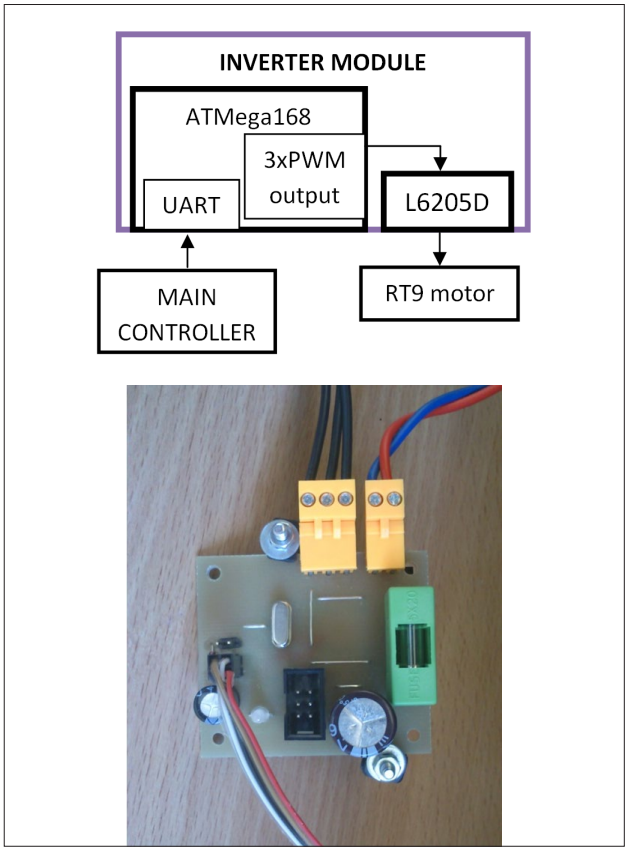

Figure 4. Photo and simplified scheme of speedometer controller

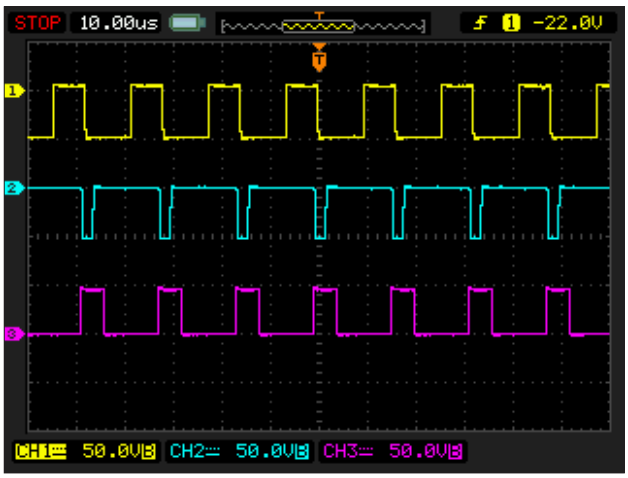

Figure 5. Scheme of the H-bridge and PWM signals at the output

The module of adjusters is a set of three specially designed optical encoders built with slot optocouplers. The power supply module consists of a switched mode power supply with regulated output voltage and 4 step down DC-DC converters built on the A8498 chip.

Software interface

The main program of the controller starts with System_Init() function which is responsible for I/O ports initialization. Next USART, SPI (I/O), timers PWM, ADC converter and global variables are initialized (see Code 1). The interrupts are enabled (sei()) and first data conversion is started (ADC_Start()). It is necessary because ADC is running in interrupt mode. In infinite while loop, I/O support is perform with a delay. If all data are receive (bit NEW_DATA is set in myReg variable) them timers and speed data are refreshed and them bit is cleared.

```
int main(void)
{
     uint16_t i;
     System_Init();
     sei();
     ADC_Start();
   while (1)
    {
     if (!(++i%255))
     {
             SPI_Data();
             CheckDir();
     }
     if (myReg & NEW_DATA)
     {
             RT9_Data();
             Timers_Refresh();
             myReg &= ~NEW_DATA;
     }
    }
}
```

Code 1. Main function in program of microcontroller

On the hardware side, data about actual control, switch settings and indicators are stored in two data types structures: dataIn i dataOut (see Code 1).

```
typedef struct{
     uint8_t dataLength;
     uint8_t speed;
     uint8_t sw0;
     uint8_t sw1;
```

```
        uint8_t sw2;
        uint8_t sw3;
        uint8_t sw4;
        uint8_t sw5;
        uint16_t brakePress;
        uint16_t mainPipiePress;
        uint16_t mainTankPress;
        uint16_t highVMeter;
        uint16_t ampHV1Meter;
        uint16_t ampHV2Meter;
        uint16_t ampHV3Meter;
        uint16_t PWM0;
        uint16_t PWM1;
        uint16_t PWM2;
        uint16_t PWM3;
        uint16_t PWM4;
    }dataIn_t;

    typedef struct{
        uint8_t dataLength;
        uint8_t sw0;
        uint8_t sw1;
        uint8_t sw2;
        uint8_t sw3;
        uint8_t sw4;
        uint8_t sw5;
        uint8_t mainController;
        uint8_t shunt;
        uint16_t mainBrake;
        uint16_t locomotiveBrake;
        uint16_t analog0;
        uint16_t analog1;
    }dataOut_t;
```

Code 2. Input/Output data structures of main controller

Data transfer frequency is set in MaSzyna config file and default value is 20 Hz. Communication is initiated by computer whith MaSzyna software running, which send first byte of data to main controller via virtual USB COM Port. Then automatically receives first byte of data from main controller. Communication is continued until all data are sending in both directions (see Fig. 6), without unused data at the end of the data frame.

The communication is handled in the interruption, so it does not affect other functions (see Code 3). Receiving data updates the PWM timers outputs and the speedometer. The current settings are constantly read in the main loop. An exception are analog inputs, which are handled in the interruption.

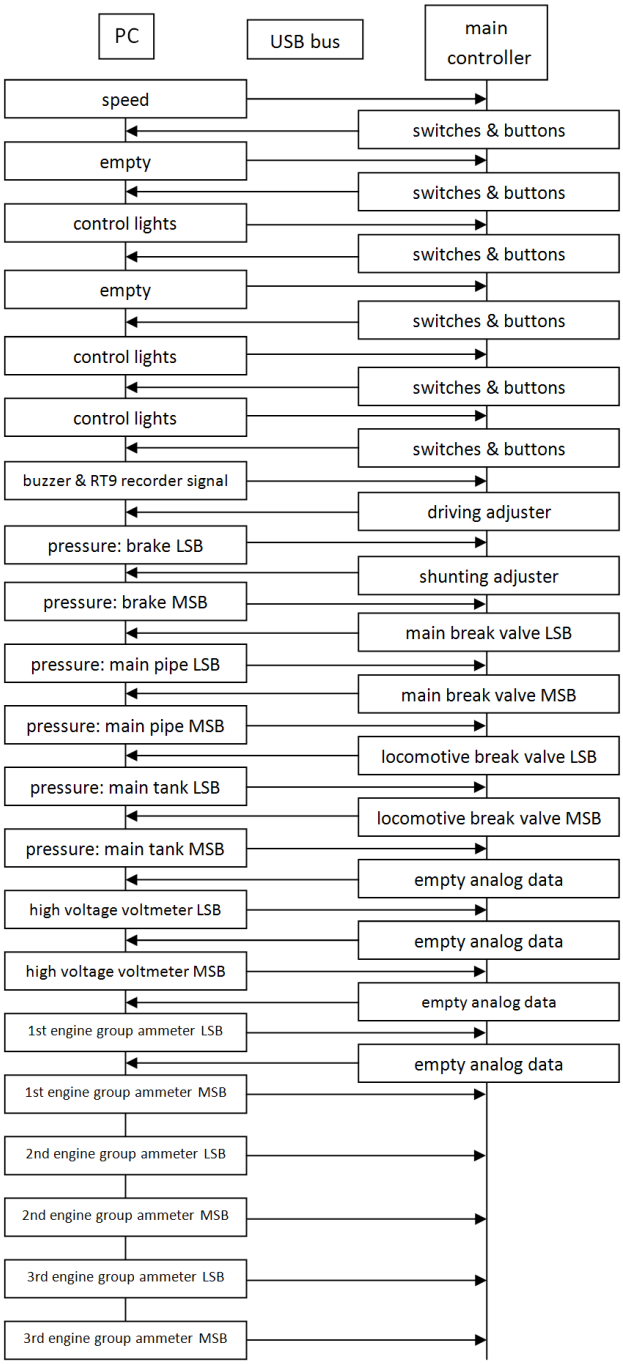

Figure 6. Data sequence diagram between PC and main controller (MSB – Most Significant Byte, LSB –Least Significant Byte).

```
ISR(USART0 _ RX _ vect)
{
     *(pdataIn + byteRx) = UDR0;            // read data
     if (byteRx < dataOut.dataLength)       // check data to send
     {
             while (!(UCSR0A & (1<<UDRE0))); // waiting for empty TX buffer
             UDR0 = *(pdataOut + byteRx);     // send data byte
     }
     byteRx++;
     if (byteRx >= dataIn.dataLength)
     {
             byteRx = 0;                            // RX data counter
             myReg |= NEW _ DATA;           // set NEW _ DATA flag
     }
}
```

Code 3. Receiving data in IRQ handler

Conclusions

Project described in the paper shows a possibility build a low cost locomotive simulator based on real components and open source software at lower cost than professional systems. Simulator provides realistic experience of locomotive behaviour. Few train drivers were invited to test developed prototype. Their evaluation is satisfying – the simulator while driving, does not deviate from the original. Other enthusiasts interested in building own simulator can use the same software, that is realistic and provides good graphic details. It is usually used Arduino platform as a main controller in simulator, because it is easier for programming.

References

1. Sharma A, Purwar R, Mishra P. Realistic Train Simulator. Texas Instruments India Educators, Conference, Bangalore, India 2014.
2. Sovička P, Pácha M, Rafajdus P. Development of an advanced locomotive simulator. 6th International Youth Conference of Energy, Budapest, Hungary 2017.
3. MaSzyna project website – http://eu07.pl/
4. Madrjas J. Obowiązek szkolenia maszynistów na symulatorach o rok później. A były trzy lata… Rynek Kolejowy. 31.10.2017. doi: http://www.rynek-kolejowy.pl/wiadomosci/obowiazek-szkolenia-maszynistow-na-symulatorach-o-rok-pozniej-a-byly-trzy-lata-84197.html
5. Nowoczesny symulator jazdy lokomotywą PKP Intercity już gotowy! PKP Intercity SA. doi: https://www.intercity.pl/pl/site/o-nas/dzial-prasowy/aktualnosci/nowoczesny-symulator-jazdy-lokomotywa-pkp-intercity-juz-gotowy!.html