Upgrading the Management of Electronic Systems in a Passenger Car

Josip Sučec, Ernest Gungl, Zmago Brezočnik

University of Maribor, Faculty of Electrical Engineering and Computer Science, Koroška cesta 46, 2000 Maribor
E-mail: josip.sucec@student.um.si, ernest.gungl@um.si, zmago.brezocnik@um.si

Upgrading the Management of Electronic Systems in a Passenger Car

Abstract: *This paper introduces the process of modernizing an older model of a passenger car. The mechanical, electrical, and programming solutions were required to upgrade the existing car. Upgrades have been done to improve safety, comfort, and performance. The safety upgrades contain the development and implementation of the brake temperature and tire pressure monitoring systems, respectively. The comfort upgrades include the car's light system and accessory improvements, such as the "follow me home" function and automatic window closing. The performance upgrades include installation of the racing coilovers and the development of the remote damping ratio adjustment system. A vehicle specific Android device was installed for the central logic unit. It controls the local electronic control units made for each specific task and displays tire pressures and brake temperatures. The Android device communicates with the electromechanical car systems by using several serial buses. First, the communication protocols and their messages in the original car had to be reverse engineered, so that later we were able to establish proper communication between the systems in the upgraded car.*

1 Introduction

The car used as a base for upgrading the management of its mechanical and electronic systems is the 2002 BMW 3 series (Figure 1). The car is equipped with two different serial communication buses – the CAN-Bus for time-sensitive and passenger safety equipment, and the K-Bus based on the ISO9141-2 standard for multimedia and devices that are not critical for passenger safety. The usage of two different buses makes our vehicle the best choice for upgrades since working and testing on the K-Bus does not interfere with crucial components connected to the CAN-Bus [1].

Figure 1. 2002 BMW 3 series

Upgrading of the existing car began as a master's thesis and was inspired by the equipment of newer, more modern cars. The newer cars offer many safety and comfort options, such as remotely adjustable suspension or TPMS (Tire Pressure Monitoring System), and those are some of the upgrades we managed to implement in the car.

The car did not have an infotainment system, but only a standard radio receiver. We replaced it with a vehicle specific Android device (Figure 2).

Figure 2. Vehicle specific Android device

The installation of the Android device made all upgrades possible, as it is used as a master device which controls both original as well as the added equipment. The K-Bus is a one-wire, 12 V idle state serial bus with a data link layer very similar to UART [1]. Therefore, the easiest way to establish communication with the bus was via a serial communication port, but since the Android device does not support a serial port, we had to design a special transducer. The USB to K-Bus transducer matches both the physical and data link layers of the buses to establish communication between the Android device and the car. We spent much time on reverse engineering the K-Bus messages, and the decoded protocol enabled many upgrades, such as DRL (daytime running lights) or "follow me home" function as a part of comfort and safety upgrades.

The performance upgrades of the vehicle included installation of racing suspension parts using coilovers (an abbreviation of "coil-over shock absorbers"). [2]. The coilovers allow independent changes of vehicle height, spring preload, and damping ratio. The electro-mechanical system was developed for remote adjustment of the damping ratio, and it is controlled via the Android device.

TPMS and the brake temperature monitoring system were added as safety improvements. They display

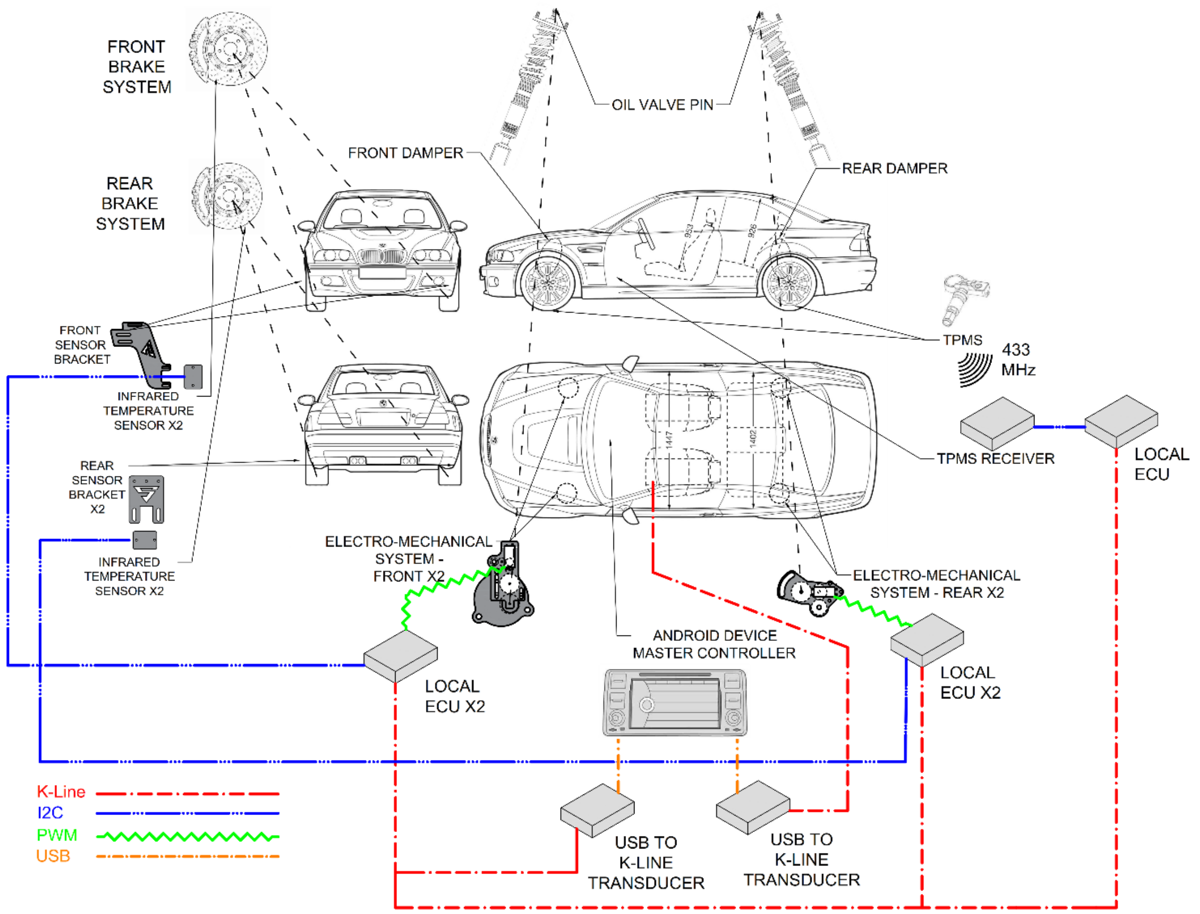

Figure 3. Conceptual solution for car upgrading

information on tire pressure and brake temperature on the Android device.

The conceptual solution and hardware description are given in Chapter 2, while software solutions are described in Chapter 3. In Chapter 4, we describe the realization and test of the added equipment. In the Conclusions, we evaluate the success of the project and give some ideas for adding further functionalities.

2 Hardware

2.1 Conceptual Solution

The conceptual solution of the upgraded car is shown in Figure 3. As we mentioned, the car has multiple serial buses, and we had to design two transducers for communication with the car and the newly developed systems. The coilovers have a damping ratio adjustment pin at the top, and the electromechanical system was designed to establish a remote adjustment of the damping ratio. Also, we had to design a local ECU (Electronic Control Unit) for control of the electromechanical system. We did not design the TPMS receiver by ourselves but bought one that was available. The receiver receives data from the sensors mounted inside the car's wheels via 433 MHz radio communication [3]. We use a local ECU for connecting to the I^2C bus on the circuit board of the TPMS receiver and listen to the data sent to the display of the receiver.

2.2 USB to K-Bus Transducer

The simplified scheme of the USB to K-Bus transducer is shown in Figure 4. The transducer enables communication between the Android device and the car. It is based on an FTDI FT230XS-R USB to UART bridge, which converts USB messages to UART serial data. We designed a two-layer PCB (Printed Circuit Board) for robust and safe operation [4]. The FTDI chip operates at CMOS levels (3.3 V), but the K-Bus at the car battery voltage levels (12 V). The UART uses two wires (RX and TX) while the K-Bus only one bi-directional wire (K-LINE).

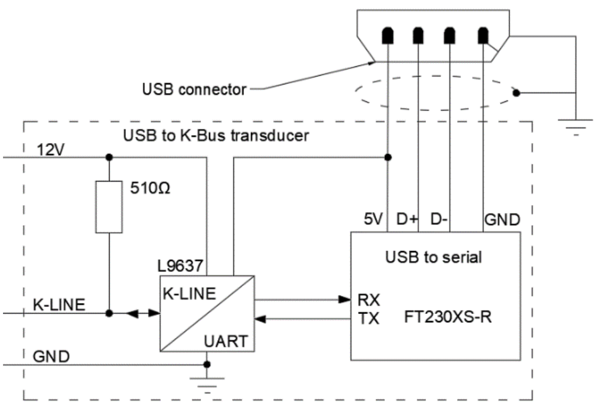

Figure 4. Simplified scheme of the USB to K-Bus transducer

To match the physical layers of the two buses, we had to implement a special transducer, the L9637, made by STMicroelectronics [5]. The layer matching could have been accomplished with the use of a few MOSFET transistors, but during the experimenting, the solution was found to be insufficient. This more straightforward and cheaper method can lead to high voltage induction due to parasitic inductivity and high-speed switching, resulting in destroying the equipment [6]. Therefore, the device used was explicitly made for this application.

2.3 Electromechanical System for Damping Ratio Adjustments

The electromechanical system is used for turning the oil valve inside the damper. By tightening or loosening the valve, we can adjust the damping ratio of the damper. The electromechanical system was designed in AutoCAD, and after many trials and errors, the final design was sent to a local machining shop. The brackets were laser cut out of an aluminium AlMg3 alloy suited for mechanical use. The system consists of a servomotor for turning the oil valve, and a potentiometer for position feedback. Due to minimal bonnet space clearance, the servomotor and the potentiometer had to be moved to the centre of the engine bay to ensure normal bonnet closing. The front electromechanical system for damping ratio adjustments is shown in Figure 5.

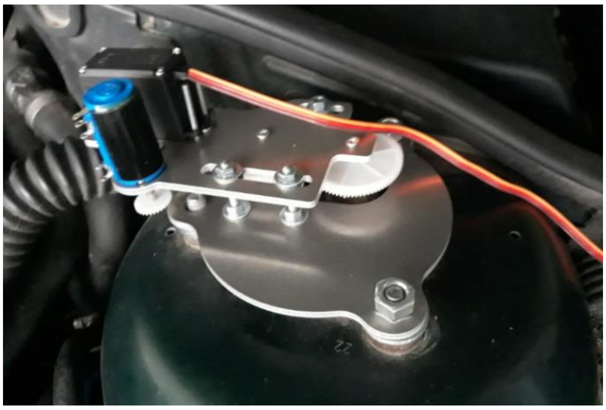

Figure 5. The front electromechanical system

2.4 Local ECU

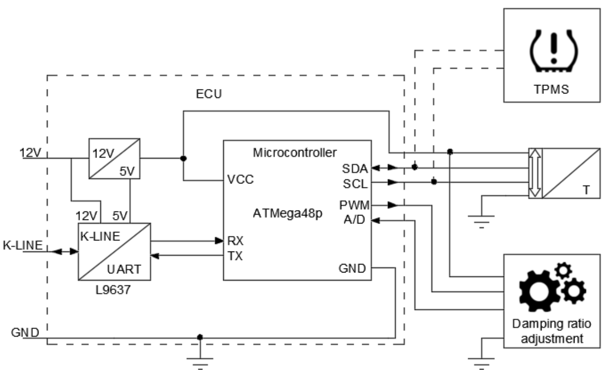

Figure 6. Simplified scheme of the Local ECU

The local ECU is used for controlling the electromechanical system and for the brake temperature monitoring system. We designed its PCB around an automotive grade ATMega48p microcontroller, and with I^2C, A/D, and PWM input and output ports [7].

2.5 Infrared Temperature Sensor

The brake temperature monitoring system is based on a Melexis MLX90614 infrared thermometer [8]. We chose this sensor because it is a low-cost and reliable solution. A significant problem with the sensor is that it is not waterproof. To make it waterproof, we had to choose a material that is transparent to the IR light and would protect the lens of the sensor from water. PVC (polyvinyl chloride) foil was the most straightforward solution, with very low attenuation of the signal. We had to make an enclosure that would hold the sensor in place, and we chose polyester resin and moulded the sensor. The sensor had to be mounted on a bracket, close to the brake disc to provide the best results (Figure 7).

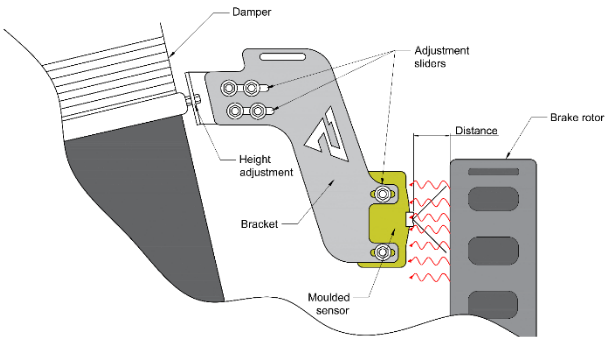

Figure 7. Mounting of the brake temperature sensor

The sensor's readings can be off if not calibrated because the sensor is factory-calibrated to the black body. Usually, a brake disc has an emissivity factor between 0.3 and 0.55 [9]. With proper calibration temperature readings can be very accurate.

3 Software Solution

3.1 Local ECU

The local ECU's software was developed in the Atmel Studio and written in C. The program is written modularly. We first programmed the UART communication, and after that made it as a stand-alone library. The same procedure was made for the A/D, I^2C, and PWM features. The software is very efficient and easy to use for future upgrades.

3.2 Android Application

The application that we made for the Android device was developed in the Android Studio and written in the Java programming language. The software performs several tasks; it communicates with the car via a serial bus, controls the local ECUs, and manages the communication flow.

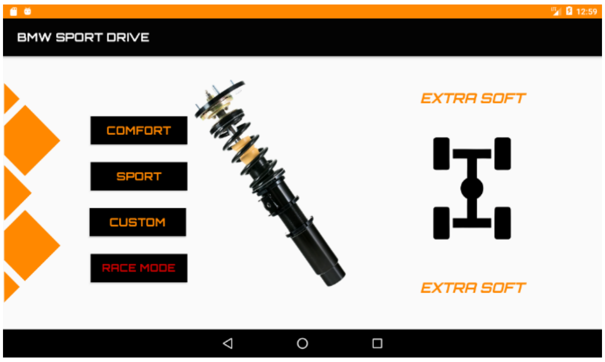

Figure 8. Screenshot of the Android application

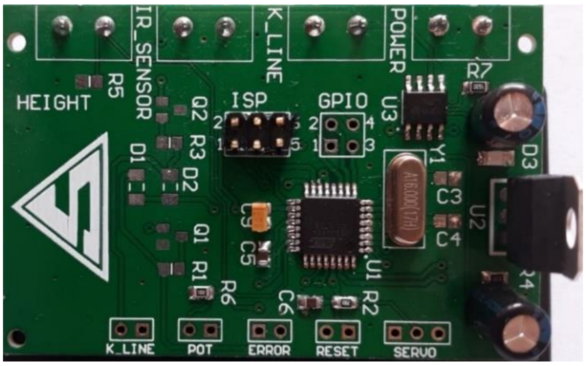

Figure 9. The PCB of the local ECU

For each wheel, it displays brake temperature and tire pressure with a sound warning activated when critical values are reached. We also implemented a car service interval notification and low engine oil level warning.

Figure 8 shows how the damping ratio is set. There are two buttons (Comfort and Sport) for faster and preloaded parameter change and another two buttons (Custom and Race Mode) for more precise parameter setting.

The local ECUs communicate with the Android device via a serial bus (ISO9141-2). There are five different local ECUs. To establish proper and collision-free communication, we had to implement some message arbitration. The Android device sends a request message every 50 ms to the serial bus, and only the device with the specific address responds, thus removing the chance of data collision.

The car's K-Bus is very busy. Often, the device has only 30 or 40 ms of time to send a message. Each device must sense if the bus is busy, and it may start communicating only if the bus is free.

Therefore, we made a background thread which has only one task; it informs the rest of the program if the bus is free or not. If the bus is busy, then the background thread stops data transmission and only receiving is allowed.

4 Implementation and Testing

We used AutoCAD for drawing the design of the electromechanical system. With the use of simple cardboard, we simulated how close to the engine we should move the servomotor and the potentiometer. The plastic gears used as transmission were processed with simple hand tools, and to make a mechanical coupler, we had to use a welding machine and an angle grinder to make a proper mechanical connection. The local ECUs are communicating with the Android device flawlessly since we managed to implement data collision avoidance, and the K-Bus idle state sensing provides robust and safe control over the car’s modules. The two-layer PCB design of the local ECUs was developed in Altium Designer (Figure 9).

5 Conclusions

The main goal of the work was to improve the project car within a limited budget. It turned out that the developed car improvements work as intended and improve the driving experience. The performance, safety and comfort have been improved significantly, thus making the car more modern.

The car can be upgraded further. With the reverse engineering of the CAN-Bus, we could develop an automatic speed limit or a complete data acquisition such as engine load and oil temperature.

Literature

[1] F. Touanen, *I-BUS Inside: Inside the BMW Cars entertainment Serial Bus*, 2002. [Online]. Available: *https://mikrob.ru/download/file.php?id=92802*. [Accessed 4 June, 2019].

[2] HSD Coilovers, "Featured HSD Products," Driftworks LTD, 2019. [Online]. Available: https://www.hsdcoilovers.com/. [Accessed 4 June, 2019].

[3] A. Arnold and S. Piscitelli, "TPMS Receiver Hacking," Worcester Polytechnic Institute, Worcester, 2015.

[4] Future Technology Devices International Ltd, "FT230X (USB to BASIC UART IC)," Future Technology Devices International Ltd, 2016.

[5] STMicroelectronics, "L9637 - Monolithic bus driver with ISO 9141 interface," STMicroelectronics, 2013. [Online]. Available: https://www.st.com/resource/en/datasheet/l9637.pdf. [Accessed 4 June, 2019].

[6] G. Mahajan, S.K. Parchandekar, and M. Tahir, "Implementation and Validation of K Line (ISO 9141) Protocol for Diagnostic Application," *International Research Journal of Engineering and Technology*, vol. 4, no. 7, pp. 708-712, 2017.

[7] Atmel Corporation, "ATmega48/88/168 Automotive," Atmel Corporation, San Jose, 2014.

[8] Melexis, "MLX90614 family," 2018.

[9] Engineering ToolBox, (2003). Emissivity Coefficients Materials. [Online] Available: https://www.engineeringtoolbox.com/emissivity-coefficients-d_447.html. [Accessed 4 June, 2019].