Bluetooth Adapter for OBD-II Systems

C. Serra, N. Boa-Alma, T. Dias, P. Sampaio
ADEETC, ISEL – Instituto Superior de Engenharia de Lisboa, Instituto Politécnico de Lisboa, Portugal
36907@alunos.isel.ipl.pt 36873@alunos.isel.ipl.pt tdias@cc.isel.ipl.pt psampaio@cc.isel.ipl.pt

Abstract — **Currently, all vehicles incorporate an OBD-II self-diagnostic system that can be used to assess information related with the general status of the vehicle. In the market, there are many commercial and free applications to analyse such data. However, only a few of them support direct communication with the vehicle, which is of the outmost importance because it can help to prevent potential breakdowns and, consequently, reduce the vehicle's maintenance costs. In this paper, it is presented a low cost embedded system that enables any programmable computing device with a Bluetooth interface to communicate with the OBD-II self-diagnostic system of a vehicle. This system is based on a NXP LPC1769 microcontroller and ensures compatibility with the KWP2000 protocol, which is supported by all car manufacturers.**

Keywords: OBD-II; Self-diagnostic; Embedded System; Bluetooth; KWP2000; NXP LPC1769.

I. Introduction

All vehicles manufactured nowadays incorporate a second generation On-Board self-Diagnostic system (OBD-II) [1] that is responsible for assessing and storing the data reported by the components of the vehicle with the capability to affect its emission performance, as well as for managing specific control parameters of the vehicle to ensure its correct operation. For example, the main parameters that determine the behaviour of the engine and that must be checked to validate its performance are: speed, engine rotations, temperature, fuel consumption, ambient temperature, airflow and exhaust emissions.

To monitor the required parameters, the vehicles include sensors that provide data to Electronic Control Units (ECUs). These ECUs, which interact in simple networks communicating at high speeds, are in charge both of processing such data and of manipulating the corresponding vehicle mechanical component(s), ranging from motor, brakes, wipers, braking, and even steering [1]. The analysis of these parameters allows knowing (in detail) the current state of the vehicle, can help to prevent potential breakdowns or even assist in repairing minor faults without having to resort to a specialized workshop.

There are many applications that can be used to perform this analysis (some of them are available for free), but there are only a few devices that support the required communication through the vehicle's J1962 connector [1]. Nonetheless, most of the existing devices are closed systems (i.e. not updatable) that present limitations on the number of implemented protocols. In general, such systems are composed of three modules: *i*) one module accomplishing the user interface; *ii*) one module responsible for the implementation of the vehicle's communication protocols, usually based on the ELM327 chip; and *iii*) one module implementing the application logic required for the exchange of information between the other modules, which is usually realized using a microcontroller (MCU).

Knowing that such MCU is essential for the system operation and that it can also be exploited to accomplish the communication with the vehicle, even when multiple protocols are considered, we have developed an embedded system [2], for a non-professional use, which enables the communication between the OBD-II system of a vehicle, via its standard OBD-II interface port and using the KWP2000 protocol [1], with any computing device running a generic OBD-II diagnosis application, by using a Bluetooth link.

The remainder of this paper is as follows. Section II introduces the hardware component of the devised Bluetooth adapter for OBD-II systems, while the architecture of the software component is presented in Section III. Some experimental results of the realized prototype are reported and discussed in Section IV.

II. Hardware Platform

The hardware component of the devised Bluetooth Adapter for OBD-II systems [2] consists mostly of the four modules that are depicted in Fig. 1.

The core of the system, identified with the number 1 in Fig. 1, is an LPC1769 MCU from NXP [2] that includes an ARM Cortex-M3 processor and a wide range of peripherals matching the target application requirements, namely two UARTs for communication with the Bluetooth and OBD-II modules flexible GPIO interfaces to interact also with the OBD-II module and a timer, as well as relatively large RAM and Flash memories.

The OBD-II interface is based on a L9637 Integrated Circuit (IC) from ST Microelectronics (identified with the number 3 in Fig. 1), which is used for multiplexing the reception (Rx) and transmitting (Tx) channels of an UART in a single unidirectional line conforming to the ISO 9141 standard, and thus support the communication with the vehicle's K-line.

The Bluetooth interface was implemented using the FB155BC module from FIRMTECH Co Ltd [3] (identified with the number 4 in Fig. 1), which enables using an UART

Fig. 1 - Block diagram of the devised hardware platform.

and AT commands to easily communicate with the user device using a virtual two-way wireless communication channel, with configurable transmission frequencies in the range of 1 Mbps and 3 Mbps.

The system is powered directly from the vehicle's battery through its standard OBD-II connector, for which it includes an optimized power supply module based on the LF33ABDT circuit from ST Microelectronics (identified with the number 2 in Fig. 1) to perform the necessary voltage and current regulations.

III. Software Component

The software component of the system consists of a highly modular bare-metal program, developed using the C programming language, that is structured in three main layers, as illustrated in Fig. 2.

The application is the key part of this architecture, since it is responsible not only for controlling the entire system but also for implementing the two-way communication channel between the Bluetooth and the OBD-II interfaces. To achieve such goal, it interacts with the Bluetooth and OBD-II Interface Managers, which virtualize the communications with the corresponding external physical interfaces by implementing all the required protocols. The control of all the involved MCU peripherals is performed at a lower layer by the UART, GPIO and TIMER Peripheral Managers.

With this approach, data is exchanged between the OBD-II interface of the vehicle and the user device with an active Bluetooth connection in four steps, as it can be seen in Fig. 3:

1. The application stands by awaiting for requests coming from the Bluetooth interface.
2. The request is sent to the vehicle through the ODB-II interface. For the first valid request that is received, a new communication session is started.
3. Then, the application awaits for the vehicle response and sends back such data to the user device using the Bluetooth interface.
4. The application returns to its initial state and waits for new requests from the Bluetooth interface, after which it repeats the flow previously described. In case the received request corresponds to a disconnect command, the communication session is terminated.

In the presented proof-of-concept version, only the

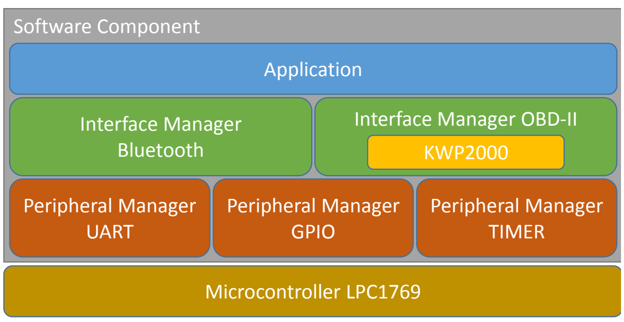

Fig. 2 - Application flowchart.

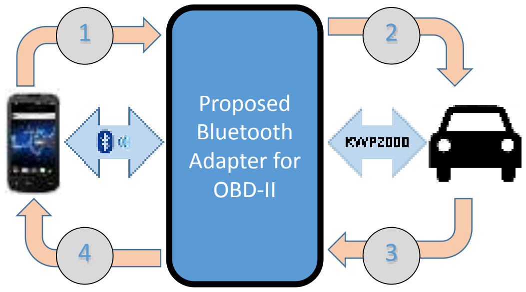

Fig. 3 - Block diagram of the system architecture.

communications using the KWP2000 protocol are supported, which is a protocol implemented by all car manufacturers. Nevertheless, the software can be easily modified to support any set of communication protocols defined in the OBD-II standard [1]. In fact, due to the highly modular structure of the devised software component, to support other communication protocols it is only necessary to develop the corresponding protocol controller and to integrate it in the OBD-II Interface Manager. Together with the ease of loading a new version of the application to the MCU's Flash memory, this makes the presented embedded system a viable long-term Bluetooth adapter for OBD-II systems.

IV. Experimental Results

The devised proof-of-concept program was compiled using the LPCXpresso v8.2.2 [Build 650] IDE with the GNU gcc v5.4.1 compiler with optimize for size and no debug symbols selected and programmed into the LPC1769 MCU, requiring only 11KB of FLASH memory and 2KB of RAM memory for its proper operation. These results not only demonstrate the quite modest memory requirements of the system's software component but also show that the RAM and FLASH memories of the considered MCU have enough capacity to support more elaborate implementations supporting additional OBD-II communication protocols.

The functionality of the whole system was successfully assessed using the Diamex OBD2 Profi Simulator [4] to simulate the vehicle's OBD-II system and a smartphone with the Android OS running the widely used and freeware OBD Dashboard application [5].

References

[1] J. Truglia and R. Birnbaum, Getting to Know OBD II, New York: A S T Training, 2000.

[2] N. Boa-Alma and C. Serra, "Adaptador Bluetooth para Sistema OBD-II," Instituto Superior de Engenharia de Lisboa, Lisbon, 2015.

[3] NXP Semiconductors, *UM10360 - LPC17XX User Manual*, 3.1 ed., 2014.

[4] Diamex Production and Trade GmbH, "Diamex Simulator - User Manual," Berlin, 2015.

[5] N. Frenay, "OBD Dashboard," 2014. [Online]. Available: https://play.google.com/store/apps/details?id=br.com.frenay.nicolas.obddashboard.