Instituto Tecnológico y de Estudios Superiores de Occidente

Reconocimiento de validez oficial de estudios de nivel superior según acuerdo secretarial 15018, publicado en el Diario Oficial de la Federación del 29 de noviembre de 1976.

Departamento de Electrónica, Sistemas e Informática

Especialidad en Sistemas Embebidos

Cost-Effective USB-CAN Interface for Automotive Testing and Development Processes: Bridging the Gap with Software

---

TRABAJO RECEPCIONAL que para obtener el **GRADO** de
ESPECIALISTA EN SISTEMAS EMBEBIDOS

Presenta: **ARTURO ALEJANDRO CERVANTES VELOZ**

Director **DIEGO ANTONIO MEJÍA SANCHEZ**

Tlaquepaque, Jalisco. 31 de Julio de 2023

Cost-Effective USB-CAN Interface for Automotive Testing and Development Processes: Bridging the Gap with Software

Arturo A. Cervantes-Veloz
Department of Electronics, Systems and Informatics DESI, ITESO
(*Instituto Tecnológico y de Estudios Superiores de Occidente*),
Tlaquepaque, Jalisco, 45604 Mexico
arturo.cervantes@iteso.mx

Diego Antonio Mejía Sánchez
Department of Electronics, Systems and Informatics DESI, ITESO
(*Instituto Tecnológico y de Estudios Superiores de Occidente*),
Tlaquepaque, Jalisco, 45604 Mexico
diegomejia@iteso.mx

***Abstract*— The implementation of Controller Area Network (CAN) in automotive testing and development processes often involves the use of licensed interfaces to communicate between computers and Electronic Control Units (ECUs). The main objective of this work is to present a cost-effective alternative to the Universal Serial Bus (USB) protocol interfaces currently available in the market. An iterative Printed Circuit Board (PCB) design is integrated with bespoke software, similar to CANoe and VehicleSPY, in terms of testing development. The results demonstrate that an in-house customized, upgradable, cost-effective USB-CAN interface bridges the gap with the software reducing the dependence on licensed interfaces. This research contributes to the advancement of hardware development for USB-CAN interfaces and showcases an affordable option for automotive software testing and development processes.**

Keywords*— *Controller Area Network (CAN), Electronic Control Units (ECU), Universal Serial Bus (USB), CANoe, Interface, Automotive Industry

I. INTRODUCTION

The Controller Area Network (CAN) communication protocol stands out as the premier choice in the automotive industry due to its numerous and unparalleled advantages, including reliability, scalability, and error handling capabilities [1]. Consequently, it is widely employed in the automotive software development process, enabling individual interaction with each Electronic Control Unit (ECU) and simulating interactions between the ECUs. This simulation is achieved through a computer using an USB-CAN interface, which facilitates communication between the computer and the programmed or tested ECU [2].

However, USB-CAN interfaces can be costly, with prices reaching up to ~€10,000 [3], [4]. This is primarily due to the licensing costs associated with the software as suppliers often link the software license with the hardware itself. Consequently, development companies often acquire fewer interfaces than the number of engineers they employ, leading not only to potential bottlenecks in the development process [4], but also increased development costs due to the extended developing time. To circumvent the expenses associated with software licenses and hardware, alternative software designs have been introduced. These designs offer limited functionality compared to commercial software but are freely licensed.

An extensively adopted approach for developing alternative software involves utilizing the USB2CAN Transceiver [5], a hardware interface developed by LAWICEL AB, which is available at a price of $151 USD. In [6], this device was integrated as an adjunct component within an easy-to-use software platform designed for CAN protocol training. However, the authors observed limitations in its performance, thereby highlighting the desirability of operating at higher speeds. This insight reinforces the development of a bespoke interface to provide an advantage in terms of customization. In [7], the USB2CAN device served as an alternative for an ECU health monitor, effectively circumventing the need for the cost-prohibitive CANoe tool [8] used for CAN communication. This substantiates the potential cost-effectiveness associated with the endeavor to develop an in-house CAN-USB interface. In [9] an ECU diagnostics validator with safety robustness compliance is presented. Their findings suggest the advancement of building an ECU health monitoring system, highlighting the importance of focusing on future expansion and upgradability in the design of a proprietary CAN-USB interface.

The above-mentioned cost-saving measures have not focused on hardware development. This is primarily due to the significant licensing expenses associated with software, which account for ~85% of the total cost to establish ECU communication via CAN. In [3], a software solution to develop and execute functional tests using the CAN protocol is developed as free software compatible with most USB-CAN interfaces since it has a delimited hardware abstraction [3]. This solution was tested with different interfaces available in the market, including USB2CAN [5], VN1630A [10], PCAN-USB [11] and ValueCAN3 [12]. In each case, the identified limitations include the absence of expandability and upgradability due to the closed nature of these devices, as well as limited customization options to cater the specific requirements, and a low cost-effectiveness as they are bound to a specific licensed software. Nevertheless, USB2CAN has the drawback of relying on a virtual RS232 port, which introduces an additional layer of processing on the CPU due to its ASCII command-based transmission mechanism [5].

This paper aims to present a cost-effective alternative to the USB protocol interfaces currently available in the market, without the need for serial communication adaptations while also ensuring integration with the software developed in [3]. The document is structured as follows: Section II discusses the hardware architecture. Section III describes the development process and difficulties. Section IV contains the overall costs and results of the project. Section V presents the conclusion.

II. USB-CAN INTERFACE DESIGN

The block diagram "Fig. 1" illustrates the interconnections between different functional blocks within the Printed Circuit Board (PCB). The blocks are described below:

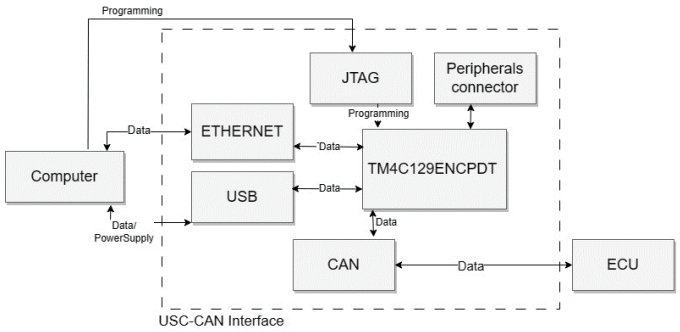

Fig. 1. USB-CAN Interface hardware architecture.

A. *USB, microcontroller, CAN and power supply.*

The modules within the dashed line box correspond to the USB-CAN interface which conveys communication between the computer and the ECU, converting signals from USB 2.0 to CAN 2.0. The cortex-M4F processor core TM4C129ENCPDT microcontroller [13] in the center supports the USB and CAN protocols. To safeguard the USB module (located at the left) against capacitive loads, short circuits, and electrostatic discharges, we opted for the TPD4S012 four-channel electrostatic discharge (ESD) protection and the TPS2052BDx power-distribution switch. At the bottom, the CAN module is equipped with a SN65HVD23x CAN bus transceiver which supports high-speed CAN (ISO 11898-2). Both the transceiver and the microcontroller are powered by a 3.3V supply, which is provided by the TPS73733DRV low-dropout regulator to adapt the USB 5V supply to 3.3V.

B. *JTAG and auxiliary peripheral module*

The JTAG debug probe connector is a compact TI 20-Pin cTI header pin out, designed to be compatible with the XDS100V3 programmer [14]. The pins of the auxiliary peripheral module connector are purposefully selected to provide supplementary support in the event of additional debugging needs. These pins encompass a range of functionalities, including analog and digital I/O, power distribution, as well as UART, I2C, and SSI peripherals.

C. *Ethernet module*

Due to the presence of Ethernet MAC and PHY within the microcontroller's hardware, the integration of 10/100 Ethernet functionality merely required the inclusion of the power over Ethernet (PoE) HX1198FNL power-enhanced single-port Ethernet connector, alongside a few essential coupling components. The widespread implementation of Ethernet in on-board entertainment systems of certain transportation vehicles [15] has enhanced its appeal as a feasible and compelling solution. While the Ethernet module remains inactive within the scope of this paper, its inclusion serves to enable future expansion and upgradability.

III. DEVELOPMENT PROCESS OF THE USB-CAN INTERFACE

The flowchart shown in "Fig. 2" illustrates the iterative development process for the USB-CAN interface, which continues until hardware integration with the software [3] is achieved. The steps are next described:

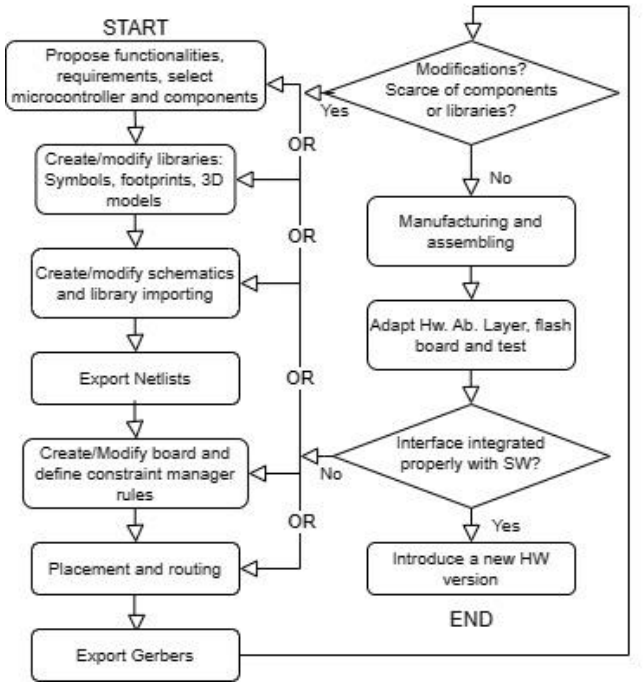

Fig. 2. Development process for USB-CAN Interface

A. *Requirements*

The process starts with developmental requirements which are influenced by component availability in the market, libraries in various online digital component databases, and the need to consider future expansion, upgradability, and compatibility with external hardware such as debuggers, ECUs, and connectors. More importantly, the process carefully addresses the essential modules to fulfill the specific needs we aim to tackle. These modules strictly consist of the USB and CAN modules, complemented by the power feed.

B. *Libraries, schematics, netlists, constraint manager, placement and routing*

Subsequently, the process involves collecting all the necessary libraries. Once gathered, schematics are created, incorporating the previously collected libraries. Following this, netlists are exported to ensure all aspects are prepared for PCB creation. With the preparations in place, the next step involved the design of an 80x50mm board. The constraint rules dictate a

line width of 0.2mm for achieving 50Ω conductor impedance and 0.17mm for 90Ω differential pair impedance, while maintaining a conductor spacing of 0.2mm. Differential pairs are subject to a maximum static and dynamic phase tolerance of 0.254mm and a maximum length of 10.0mm. The layout must align with the microcontroller's pinout as much as possible. Signal lines for data, control, and power are routed to ensure reliable communication and minimize interference. The interconnections are optimized to uphold signal integrity, comply with protocol specifications, and uphold the overall performance of the USB-CAN interface PCB.

C. Gerbers, manufacturing and assembling

The next stage involves Gerber exportation, signifying the completion of the first iteration. At this point, there is an opportunity to detect any necessary modifications and, if needed, start anew from the initial step. Early detection of silk layer compatibility mismatches with the manufacturer was enabled by a Gerber viewer, thereby preventing manufacturing errors. Then the Gerbers are sent to the manufacture stakeholder and all the components are acquired from the different online electronics component retailers. Once all constituent elements are gathered the board is assembled and tested to be alive.

D. Hardware Abstraction Layer, flash board, test and versioning

A preexisting firmware for Tiva C series was developed at [3], and it was minimally adapted in the hardware abstraction layer to suit this microcontroller. Subsequently, the board is flashed. To verify the correct integration of the interface with the preexisting software, the complete system is connected as illustrated in "Fig. 3", which shows the connections of the entire system. The setup consists of a computer responsible for interpreting and exchanging data with the USB-CAN interface, running the software designed for test development in embedded systems using the CAN protocol [3]. Adjacent to the interface, a VN1630 is connected to a computer, serving as a generic ECU simulator that sends generic CAN messages to the CAN bus. It is important to note that this simulation can be replaced with an actual ECU as needed. Upon observing messages received at the recipient computer, as shown in "Fig. 4", the USB-CAN Interface is designated as v.1.0.

Fig. 3. USB-CAN interface in a CAN bus system

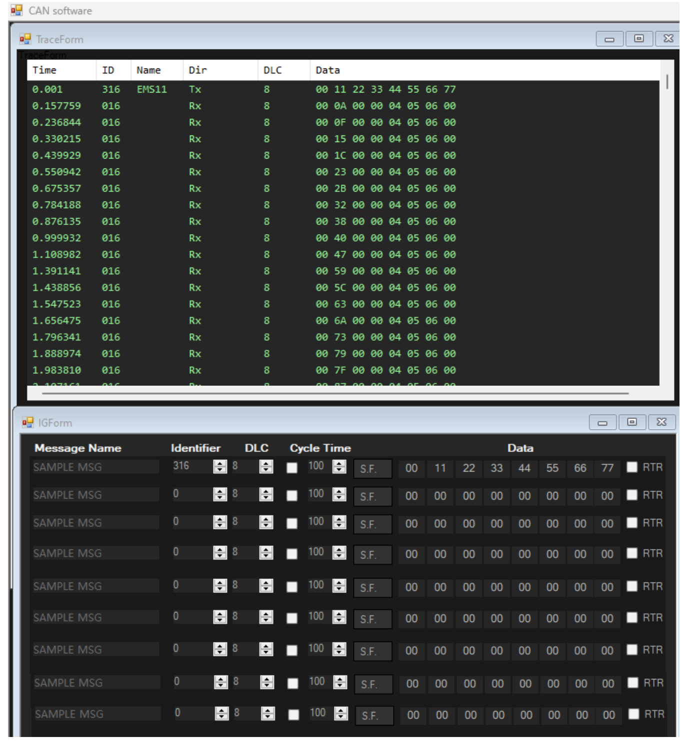

Fig. 4. Software designed for test development in embedded systems using the CAN protocol running.

IV. RESULTS

"Fig. 5" depicts the overall interconnections between layers 1 and 4 on the 80x50mm PCB. For enhanced clarity, power shapes are excluded, and layers 2 and 3 are omitted from this illustration. Layer 2 functions as a ground-shaped layer, effectively managing the electrical grounds returns of layer 1. Layer 3 is a power-shaped layer with different shapes distributed on the board including ground, 3.3v, and 5v lines. Ground shapes in this layer handle electrical ground returns from layer 4. Pink spots are ground connections, red and purple spots represent 3.3v power lines and blue color tones indicate 5v power lines. Moreover, traces and spots in shades of maroon and orange represent differential pairs (Ethernet, USB, and CAN). Lastly, green traces denote other signals, including analog and I/O digital signals, clocks, and various other connections.

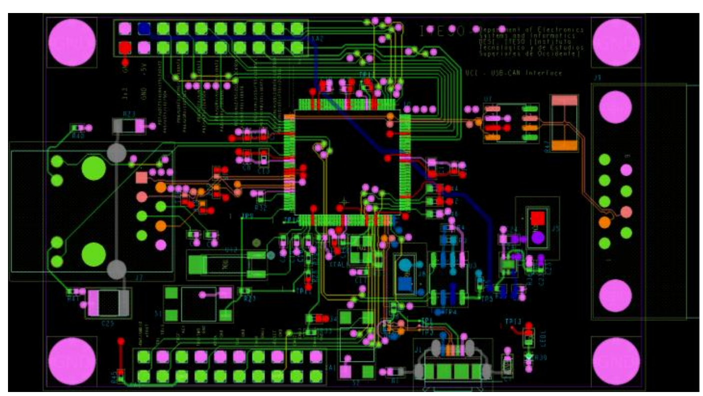

Fig. 5. Top and bottom connections of the USB-CAN interface layout

The microcontroller occupies the central position on the board, while the DB9 CAN connector is situated on the right side, and the Ethernet connector on the opposite side. The USB connector is positioned at the bottom, adjacent to the power circuitry that provides 5v which is between the 3.3v circuit components at the right and the clocks of the processor at the left. The 20-pin JTAG connector is situated on the bottom-left, adjacent to the USB connector. Finally, the auxiliary peripheral module connector runs along the top-left section of the board. The rest of components and connections are buttons, coupling capacitors and other elements.

The overall cost of the USB-CAN interface was $125 USD including components, board manufacturing (5 units), taxes, shipping fees and customs costs. “Fig. 6” depicts the assembled board. The USB-CAN interface has been successfully integrated with the software designed for test development in embedded systems using the CAN protocol [3]. The interface does not use a virtual port; it uses the USB protocol.

Fig. 6. Top view of the assembled PCB

V. CONCLUSION

The project resulted in the development of a USB-CAN interface, facilitating communication between a computer and a CAN bus. By successfully integrating with the software designed for test development in embedded systems using the CAN protocol, the interface effectively bridged the gap between the existing software and the ECU communication, all without the need of a license. Additionally, it offers a slightly more budget-friendly alternative to the USB2CAN device while eliminating the need for a virtual port by utilizing native USB. The final product is an additional choice among the options available in the market; it sets the tone to reduce the excessive reliance on conventional interfaces. Furthermore, the board is equipped with an Ethernet interface and has the capability to process encrypted data, enabling a wide range of future expansion and upgradability possibilities.

ACKNOWLEDGMENT

This work has been partially supported by the Mexican federal government through the CONACyT agency.

REFERENCES

[1] N. M. Ben Lakhal, O. Nasri, L. Adouane, and J. B. Hadj Slama, “Controller area network reliability: overview of design challenges and safety related perspectives of future transportation systems,” IET Intell. Transp. Syst., vol. 14, no. 13, pp. 1727–1739, 2020, doi: 10.1049/iet-its.2019.0565.

[2] S. Abbott-McCune and L. A. Shay, “Techniques in hacking and simulating a modem automotive controller area network,” in 2016 IEEE International Carnahan Conference on Security Technology (ICCST), Oct. 2016, pp. 1–7. doi: 10.1109/CCST.2016.7815712.

[3] D. A. Mejía-Sánchez, “Software para desarrollo de pruebas para sistemas embebidos utilizando el protocolo CAN,” Dec. 2017, Accessed: Jun. 25, 2023. [Online]. Available: https://rei.iteso.mx/handle/11117/ 5152

[4] Y. Lichtenstein, S. Dujmovic, and C. Baden-Fuller, “Strategies for Competing in the Automotive Industry’s Software Ecosystem: Standards and Bottlenecks,” IEEE Softw., vol. 36, no. 3, pp. 45–49, May 2019, doi: 10.1109/MS.2018.290105946.

[5] “CAN Tools by LAWICEL | canusb.com.” https://www.canusb.com/ (accessed Jun. 25, 2023).

[6] D. Wetzel, A. Reindl, H. Meier, M. Niemetz, and M. Farmbauer, “A Customized Python Interface for Windows OS for a Low Budget USB-to-CAN-Adapter,” in 2022 International Conference on Electrical, Computer and Energy Technologies (ICECET), Jul. 2022, pp. 1–5. doi: 10.1109/ICECET55527.2022.9872574.

[7] A. D. Sutar and S. B. Shinde, “ECU Health Monitor Using CANUSB,” in 2018 Second International Conference on Inventive Communication and Computational Technologies (ICICCT), Apr. 2018, pp. 415–419. doi: 10.1109/ICICCT.2018.8473000.

[8] “CANoe | ECU & Network Testing,” Vector Informatik GmbH. https://www.vector.com/int/en/products/products-a-z/software/canoe/ (accessed Jun. 25, 2023).

[9] A. D. Sutar and S. B. Shinde, “ECU diagnostics validator using CANUSB,” in 2017 International Conference on Inventive Computing and Informatics (ICICI), Nov. 2017, pp. 856–860. doi: 10.1109/ICICI.2017.8365257.

[10] “VN1600 Family - Network Interfaces for CAN, CAN FD, LIN, K-Line, J1708 and IO,” Vector Informatik GmbH. https://www.vector.com/int/en/products/products-a-z/hardware/network-interfaces/vn16xx/ (accessed Jun. 25, 2023).

[11] “PCAN-USB: PEAK-System.” https://www.peak-system.com/PCAN-USB.199.0.html?&L=1 (accessed Jun. 25, 2023).

[12] “Intrepid Control Systems.” https://store.intrepidcs.com/valuecan3-dw-2-channel-p/vcan-dw3.htm (accessed Jun. 25, 2023).

[13] “TM4C129ENCPDT data sheet, product information and support | TI.com.” https://www.ti.com/product/TM4C129ENCPDT (accessed Jul. 12, 2023).

[14] Olimex, “TMS320-XDS100-V3,” Olimex. https://www.olimex.com/Products/DSP/Emulators/TMS320-XDS100-V3/ (accessed Jul. 08, 2023).

[15] R. M. Daoud, H. H. Amer, H. M. Elsayed, and Y. Sallez, “Ethernet-Based Car Control Network,” in 2006 Canadian Conference on Electrical and Computer Engineering, May 2006, pp. 1031–1034. doi: 10.1109/CCECE.2006.277777.