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

Body Control Module using the SAM-V71 development board

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

Presenta: **CUAUHTÉMOC THOMAS NIKITA REYES GARCÍA**
RICARDO RAMÍREZ CASTELLANOS
IVAN SEGURA GARCÍA

Asesor: **HÉCTOR ANTONIO RIVAS SILVA**

Tlaquepaque, Jalisco.18 de julio de 2019

Acknowledgments

We want to express our gratitude to our families, for all the support given.

We want to thank to the Consejo Nacional de Ciencia y Tecnología for the scholarships provided: (CVU 937825, CVU 935976, CVU 930847).

Thanks to the Instituto Tecnológico y de Estudios Superiores de Occidente (ITESO) and the Specialization in Embedded Systems program, for all the material and equipment used in this program, as well as the teachers for all their patience, motivation, and the knowledge transmitted.

Special thanks to our adviser Hector Rivas, for guiding us throughout the process, for the suggestions and all the feedback given.

Abstract

The Body Control Module is one of the main devices inside a car since it is responsible of the critical aspects for the correct function of the vehicle including the safety and comfort of all passengers. However, these features come at a high cost. Therefore, the aim of this project was to perform a BCM capable of executing the basic functions of a commercial module in a car but with a lower cost. This was achieved using the SAMV71 development board and its embedded CAN protocol communication port and following the V-cycle which has two main branches: planning and integration of their parts and validation. This model facilitates keeping track of any progress during the development stage. The device successfully read analog and digital inputs, processed the information and sent it through the CAN bus for further processing. AUTOSAR was the standard used trough the development process, since it is the most employed in the automotive industry. It specifies that the software components shall be in layers, helping the process of integration and giving portability to the project. With this BCM it is possible to adapt a classic internal combustion engine car that lacks modern electronics to a battery electric vehicle.

Introduction

One of the main contributions of the automotive industry is developing new features in comfort, safety and security. These characteristics include devices that measure pressure, acceleration and temperature in tires, including electronic boards to control the engine system or a module to manage the exterior and interior lights, wipers, tire pressure, battery voltage, etc. This module is known as the Body Control Module (BCM) [1].

Generally, most cars include a BCM, but in developing countries, they include an on-board computer with limited characteristics or in the worst case, neither of these features are considered due to economic constraints. [2]. In this sense, for lower segment cars, a BCM with minimum features in terms of comfort, safety, security, and affordability is in high demand.

The advantages of a BCM include a lower production cost, fewer cables needed and lower complexity and a reduced weight, among others. A BCM helps reduce the complexity of the architecture, currently, a car includes nearly 100 hundred ECUs, and a BCM helps make the system more flexible. To date, the design of a BCM requires focusing on cost, efficiency, reliability, and safety.

Nowadays, the BCM sends and receives the necessary information for the driver between different Electronic Control Units (ECU) in real-time to know the status of critical aspects for the correct function of the vehicle. This information is sent through a communication protocol developed by BOSCH in the 1980s known as Control Area Network (CAN) [3]; however, the more features a car has, the more expensive it becomes. Therefore, the purpose of this project is to replicate the operation of a conventional BCM using the SAM V71 board. It is not only a low-cost development board, but it also has a 32-bit ARM Cortex-M7 RISC Microcontroller specially designed for automotive applications and it operates at a maximum speed of 300 MHz (one of the fastest on the market) with multiple analogic and digital inputs plus a CAN transceiver [4].

1. OVERALL DESIGN

In the automotive industry there is a standard for the development of new systems, the V-Cycle. This model describes the necessary steps and connections between the individual development stages of this project. The V-model has two main branches as shown in Fig. 1-1, the left describes the planning and the right represents the integration of their parts and validation.

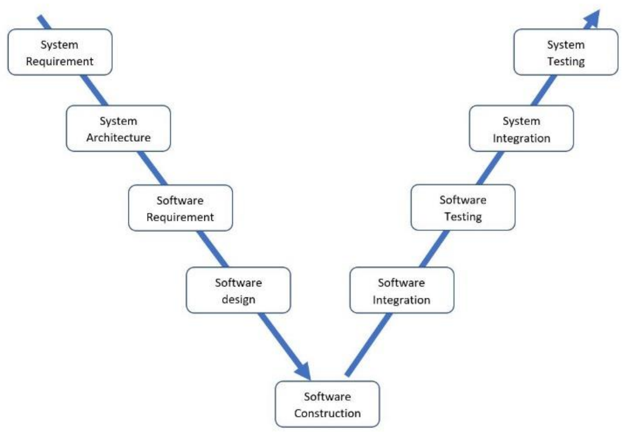

Fig. 1-1 V-Cycle.

The advantage of the V-Cycle is its intuitive nature since it is easy to use in a systematic way and it also facilitates keeping track of any progress during the development stage. The disadvantage of this model is that the code is developed midway in the system's life cycle, so there are no early software prototypes, if there is a change to be made, one needs to go back to the first step in the model, specifically, to the requirements and modify them. Another big disadvantage is that almost all resources are focused on the development phase leaving little to no time for testing purposes.

1.1 Requirements

The BCM controls different signals that a user activates with external switches. These signals are exterior lights, such as turn indicators, hazards, hand-brake, high beams, low beams, position lights, a simulation of speedometer of the vehicle, tachometer, fuel level, battery voltage, environment temperature and door warning (one or more doors are open). Also, there are other signals that are activated by the system, in other words, without an intervention of an external user. These are: fuel efficiency, tire pressure, status of the compass, engine warning ABS brake warning, air bug status, oil and motor temperature warnings, seatbelt status, and satellite notification.

1.2 Architecture

Once the requirements were defined, we proceeded to design the architecture of the system based on the interaction of each part with the environment, with other devices (for example, other ECU's) and within this project's architecture. In the software development phase, we applied AUTOSAR Layered Software Architecture, which is the standard for automotive software development, it specifies that the software must be divided into layers, the way layers interact between them, and what it should not be done, for example, interfaces bypassing two or more software layers. [5]. The architecture used is shown in Fig. 1-2.

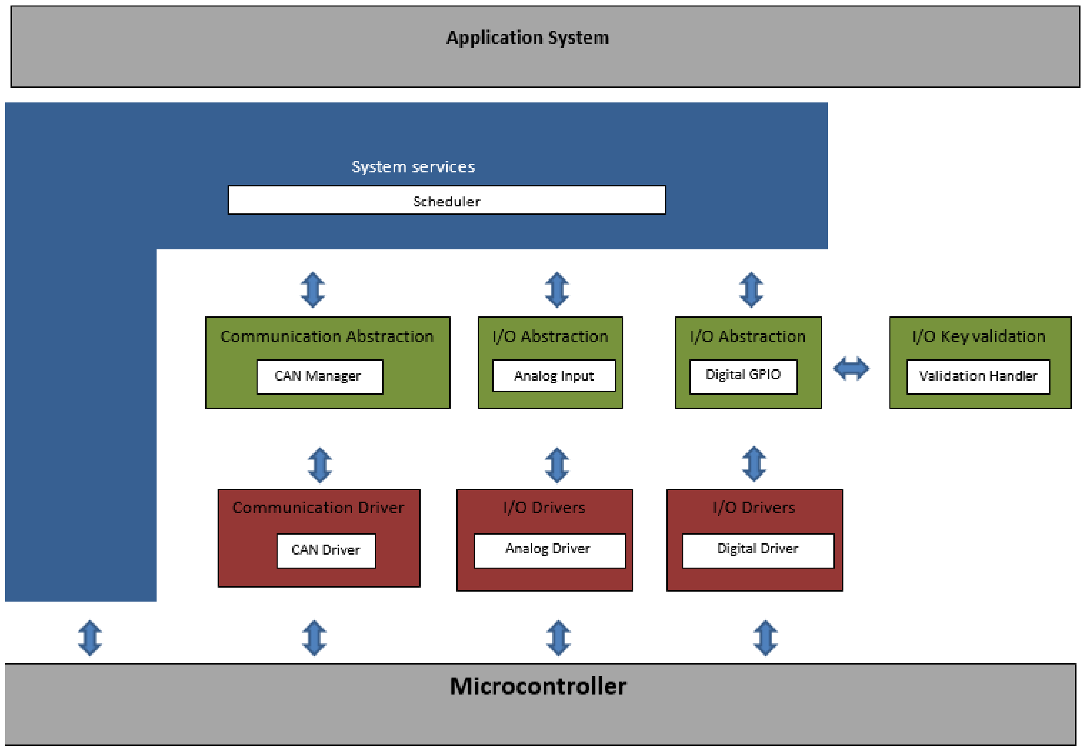

Fig. 1-2 Software Architecture of the BCM

1.3 Integration

To develop the BCM software, we used C programming language [6]. Based on the AUTOSAR layer, driver and abstraction codes, CAN manager, a code which validates the input signals if they are true or not, and finally, a scheduler that manages the functionality of the software used.

The driver codes were used to configure the ports as inputs, afterwards, with the key validation code, the input signals through input ports, were validated as shown in Fig. 1-3.

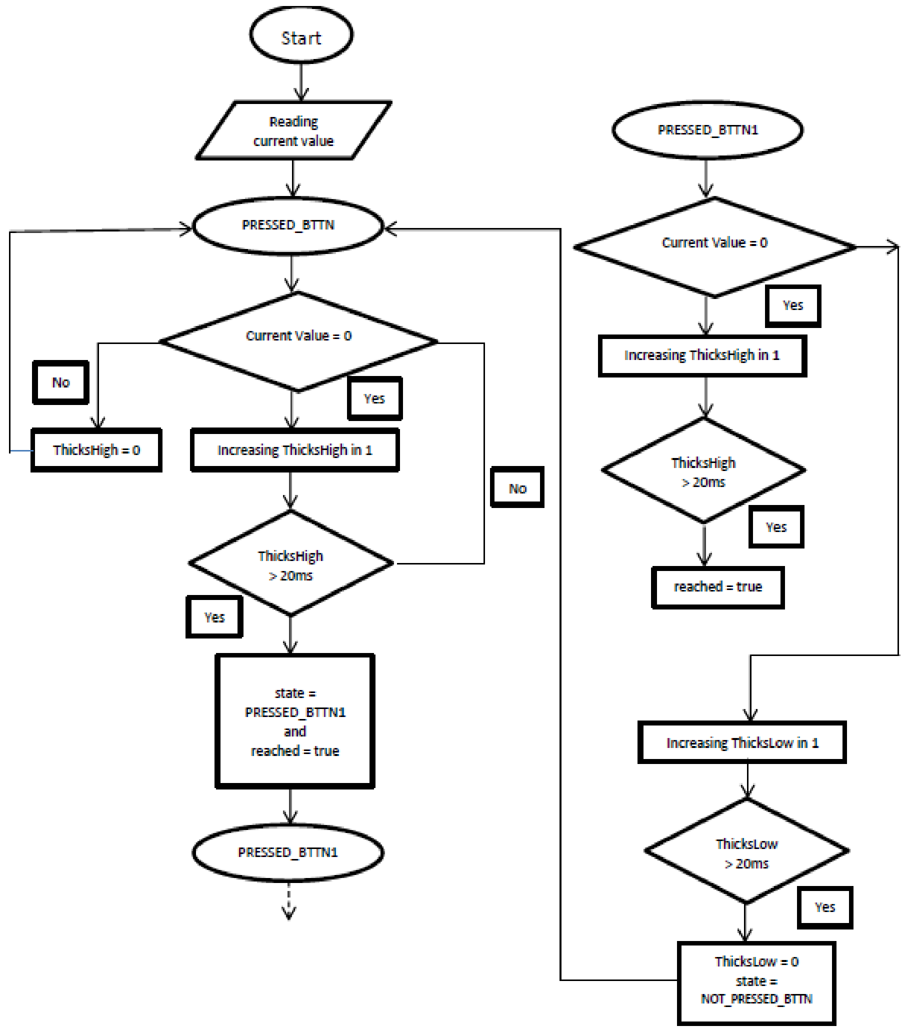

Fig. 1-3 Key validation diagram

Then, the abstraction codes obtain and interpret the information that the ports provide after being validated by key validation; afterwards, a small section of this code places the information in the CAN message structure shown in TABLE 1 and TABLE II.

TABLE I

CAN Message 010 Structure

Message 010

Byte	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	Handbrake	Fuel level	Fuel level	Fuel level	Fuel level	High Beams	Low Beams	Position ligths
1	Car Speed	Car Speed	Car Speed	Car Speed	Car Speed	Car Speed	Car Speed	Car Speed
2	Tachometer	Tachometer	Tachometer	Tachometer	Tachometer	Tachometer	Tachometer	Tachometer
3	Empty	Empty	Gear position	Gear position	Gear position	Battery level	Battery level	Battery level
4	Check Engine	Fuel efficiency	Fuel efficiency	Fuel efficiency	Fuel efficiency	Fuel efficiency	Fuel efficiency	Fuel efficiency
5	Empty	Empty	Compass position	Compass position	Compass position	Hazards	Turn rigth Indicator	Turn Left Indicator
6	Polarity temperature	Enviroment temperature	Enviroment temperature	Enviroment temperature	Enviroment temperature	Enviroment temperature	Enviroment temperature	Enviroment temperature
7	Empty	Satelite notification	Door open warning	Seatbelt status	Motor temperature	OIL warning	Airbags warning	ABS Warning

TABLE II

CAN Message 020 Structure

Message 020

Byte	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	Odometer	Odometer	Odometer	Odometer	Odometer	Odometer	Odometer	Odometer
1	Odometer	Odometer	Odometer	Odometer	Odometer	Odometer	Odometer	Odometer
2	Odometer	Odometer	Odometer	Odometer	Odometer	Odometer	Odometer	Odometer
3	Empty	Warning Tyre 1	Tyre 1 Pressure	Tyre 1 Pressure	Tyre 1 Pressure	Tyre 1 Pressure	Tyre 1 Pressure	Tyre 1 Pressure
4	Empty	Warning Tyre 2	Tyre 2 Pressure	Tyre 2 Pressure	Tyre 2 Pressure	Tyre 2 Pressure	Tyre 2 Pressure	Tyre 2 Pressure
5	Empty	Warning Tyre 3	Tyre 3 Pressure	Tyre 3 Pressure	Tyre 3 Pressure	Tyre 3 Pressure	Tyre 3 Pressure	Tyre 3 Pressure
6	Empty	Warning Tyre 4	Tyre 4 Pressure	Tyre 4 Pressure	Tyre 4 Pressure	Tyre 4 Pressure	Tyre 4 Pressure	Tyre 4 Pressure
7	Empty	Empty	Empty	Empty	Empty	Empty	Empty	Empty

The CAN manager obtains information from the drivers to build the CAN messages and after that sends the message through the CAN bus in the format shown in TABLE III.

TABLE III

CAN Message transmitted in standard format

CAN Message 010 B7 B6 B5 B4 B3 B2 B1 B0											
BUS IDLE	Start of frame	Arbitration field	RTR IDE r0 DLC	Data Field	CRC Field	DEL	ACK	DEL	End of frame	Inter Frame Space	BUS IDLE

Then, the scheduler decides when the driver will configure the ports, and at that precise moment, CAN messages are built and sent.

2. RESULTS

The BCM can output 34 unique signals: 9 digital, 5 analog and 20 are software emulated. One special input signal is the key switch as seen in Fig. 2-1, when this switch is activated, it sends the signal called tester present shown in Fig. 2-2 with an ID 0x271 and one-byte data. This message wakes up the system by turning the cluster on and the BCM starts validating the digital signal for 20 ms to avoid a false input. The BCM also takes 19 analog samples in 95 ms, so, by the 100 ms mark, all the data are averaged and sent through the corresponding CAN message. The rest of the signals are software generated, this means that a counter is assigned to each one and after a certain amount of time, the status changes.

Fig. 2-1 Engine switch key

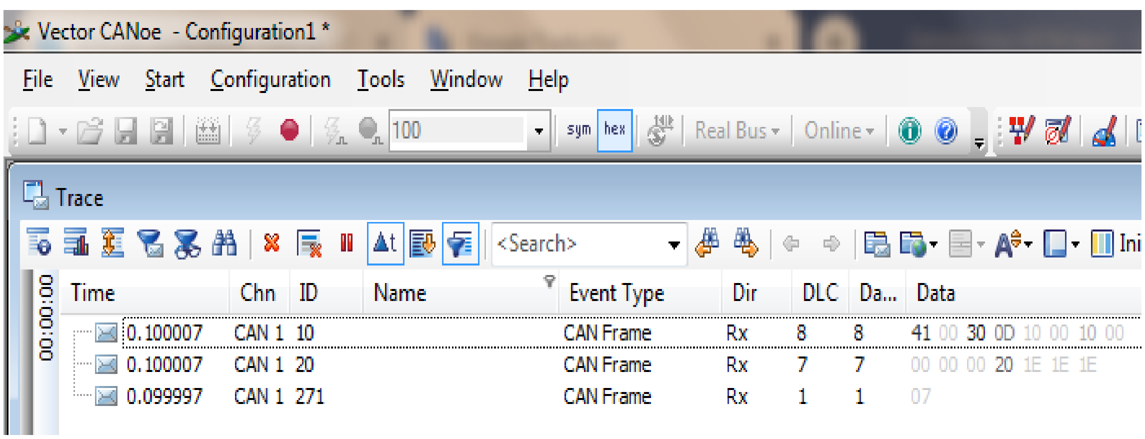

Fig. 2-2 Response received in CANoe tool

CONCLUSION

In conclusion, a low-cost BCM based on the V-cycle and AUTOSAR standard that included a module ready to interact with an indicator cluster using CAN protocol, was successfully developed. Also, this project gives the opportunity for other developers to perform a more robust system by incorporating other modules for custom applications, such as converting an internal combustion engine car to an electric vehicle.

REFERENCES

[1] Bosch Engineering, "Body Computer Module (BCM)," [Online]. Available: https://www.bosch-engineering.de/media/de/pdfs/einsatzgebiete_1/produktdatenblaetter/110621_BEG-BCM_EN.pdf.

[2] B. M. Guruswamy and P. Nagaraj, "Automotive car body module control using PIC16F877A," in International Conference on Circuits, Communication, Control and Computing, 2014, pp. 146–152.

[3] B. Ling, F. Peng, and A. Li, "The Car Body Control Bus Design Based on CAN/LIN Bus," in 2011 International Conference on Computational and Information Sciences, Chengdu, China, 2011, pp. 885–888.

[4] Microchip Technology Inc., SAMV71 http://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-44003-32-bit-Cortex-M7-Microcontroller-SAM-V71Q-SAM-V71N-SAM-V71J_Datasheet.pdf

[5] AUTOSAR, "Layered Software Architecture," [Online]. Available: https://www.autosar.org/fileadmin/user_upload/standards/classic/4-3/AUTOSAR_EXP_LayeredSoftwareArchitecture.pdf.

[6] Anandamurugan, S., "Introduction to C", in C Programming for Juniors. Hauppauge, New York: Nova Science Publishers, Inc. 2014.

[7] VECTOR, "CANoe," [Online]. Available: https://www.vector.com/us/en-us/products/products-a-z/software/canoe/.