Automotive Event Logger Based On The ESP32 Platform

Daniel Trovão Simões
Leimar Silva Schottz Mafort
Robert Bosch Ltda.

ABSTRACT

The TimeStamp Tag system is a data acquisition board created to help driver assistance developers and testers. Using ESP32 microcontrollers, this data logger records any types of abnormality detected by drivers, for example: skidding, lane departures, adaptive cruise control faults and breaking fails. The system performs the continuous reading of automobile variables using the OBD system and embedded sensors, such as accelerometer, gyroscope, compass and GPS. By pressing the button, the user saves the last 25 seconds of data on a memory card; as well as, the next 5 seconds after that event. The purpose of this platform is to assist drivers, users or testers to reduce the time of searching for these events in the vehicle video files and to generate analysis of performance in eventual anomalies.

1. INTRODUCTION

Advanced driver assistance systems (ADAS) are intelligent systems inside the vehicles that collects data through a variety of sensors and use them to perform driving and parking functions, providing safety and comfort to the driver. These systems can obtain vital traffic information such as traffic jams, blocked and congestion roads. They may judge driver fatigue and distraction and send driver's alert. In addition, they can assist in vehicle control with adaptive cruise control (ACC) and parking pilot (PP). These systems also actuate directly in the driver's safety as lane keeping support (LKS) and automatic emergency braking (AEB) [1].

Because they deal directly with human lives, the safety performance must be considered and tested to search for faults to search for faults and improvement possibilities. Several international organizations work to ensure safety and regard performance scores for these safety-focused systems, such as NCAP (New Car Evaluation Program) [2]. NCAP has several branches and is responsible for operating in the Brazilian automobile market or in Latin NCAP [3].

The purpose of this article is to present a low cost equipment aimed to help developers and evaluators on driver assistance (DA) at Robert Bosch Latin America Company.

This article is organized into five sections. This introduction is located in Section 1, In section 2, the problem case is described, showing the motivation for the development of this equipment. Section 3 demonstrates the structure and description of the system. This section is divided into hardware devices and software elements. The section 4 discuss results and analyzes with the functions and tests of the system. Finally, the conclusion of article is published in section 5, explaining the system of success or not as also future steps and improvements.

2. MOTIVATION

Just like NCAP, Robert Bosch Latin America is concerned about the safety and performance of the ADAS systems. In order to verify the operational conditions of these systems, Bosch uses cameras installed on the vehicle windshield to monitor the traffic situation and record abnormalities in the ADAS system operation during the development tests. However, it is generally hard for the evaluators to find specific sections among the recorded data, once the abnormalities were present during just a few seconds among hours of image. To make these abnormalities easier to be found, Brazilian Bosch engineering team proposed to develop a low-cost equipment that helps to locate much faster interest points inside the amount of data.

This ESP-32 platform based equipment contains a "tag button", which is pressed by the driver always that he notice an abnormality in the assistant systems, for example undesired braking, false positives or under actuation of the system. By pressing this button, the system records the date and time of the event as well as a set of extra data, such as acceleration, speed, position of the accelerator and inclination. By doing this, this equipment decreases the search time for the abnormality events on the recorded videos and gives a range of data for performance evaluation.

3. TIMESTAMP ESP32 PLATFORM

This section contains the hardware configuration of the equipment. Figure 1 shows the inputs and outputs of this project. In addition, this section contains two additional subsections: the hardware devices and the software elements.

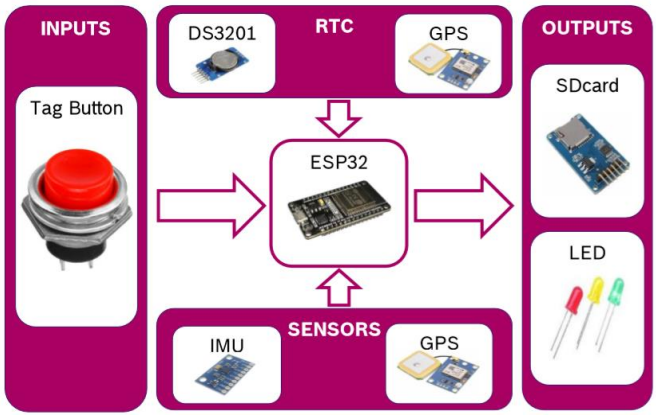

Figure 1: TimeStamp ESP32 Platform Project

According to Figure 1, the structure of the project divided in five main parts: inputs, ESP32, sensors, RTC and outputs. Inputs are the commands that the user can send to the system, in this case is the tag button. ESP32 is the microcontroller used for the project. Sensors are the IMU (Inertial Measurement Unit) and the GPS. In the block above, RTC (real time clock) has DS3201 and GPS features, which responds the time measuring. Finally, outputs are the SD card module used for saving the information and the system status LED.3.1 HARDWARE DEVICES

The components used to assemble the physical part of the project chosen based on the cost and availability in the national market. Table 1 lists the components used for the project and the costs in BRL currency.

Item	QTY	COD	Cost(R$)
ESP32	1	Esp-wroom-32	R$ 50,00
Module SD CARD	1	MicroSD Adap	R$ 10,00
Module RTC	1	DS3291	R$ 10,90
IMU	1	MPU9250	R$ 27,00
SD Card	1	Sandisk	R$ 39,00
GPS	1	u-blox neo-6m	R$ 99,00
Other Components	1	-	R$ 15,00
Total			R$ 252,00

Table 1: TimeStamp ESP32 Platform budget

ESP32 - The ESP 32 microcontroller was chosen because it supports several communication protocols: I2C, SPI, UART and mainly the CAN protocol used in automotive vehicles. In addition, this device has integrated Bluetooth and Wi-Fi enabling the project to be IoT (Internet of Things) [4].

Figure 2: ESP32 Microcontroller

SD CARD MODULE - The SD card module uses SPI communication protocol, supported by ESP32, to receive the date, time information as well as other variables coming from the sensors and stores this information on micro SD card. The microcontroller creates a CSV file (Comma Separated Values) on this SD, which can be opened in Excel file and can be easily manipulated for generating graphs [5].

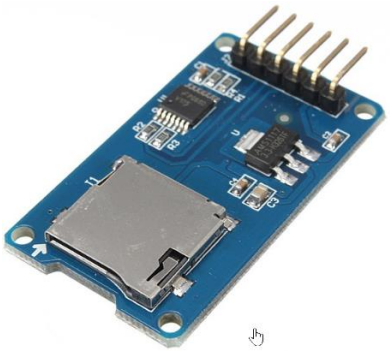

Figure 3: Micro SD Module

RTC MODULE - The DS3201 RTC (Real Time Clock) module generates the date and time if the GPS system is down. Despite the fact that ESP32 has an internal RTC, this system may suffer out of sync during routine execution [6].

Figure 4: Real Time Clock DS3201

IMU - The MPU 9250 sensor, which has 9 degrees of freedom (3 gyroscope, 3 acceleration and 3 magnetometer), reads angular variations, seek for a skid case or to check some type of vibration. [7].

Figure 5: MPU 9250

SD Card - In this project, a 16GB micro SD card was used. However, as the system records small intervals of time and only when requested by the user, a smaller (and cheaper) micro SD card can be used, allowing a cost reduction. [8].

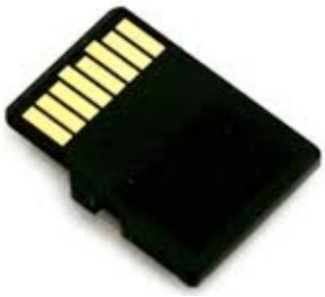

Figure 6: SD Memory Card

GPS - The GPS system used in this project has a dual function. It can both provide data on the vehicle's position and distance traveled over a given time or calibrate the TimeStamp ESP32 platform RTC system. In this case, an algorithm was developed to check if the project data were in accordance with GPS, because it has a more reliable time precision for correct synchronization with dash camera videos [9].

Figure 7: GPS u-BLOX 6M

Other Components - Other components used in the project were LEDs, PCBs (Printed Circuit Boards) and resistors.

3.2 SOFTWARE ELEMENTS

The software was programmed using C++ and created and created on the Arduino IDE, which is an open source environment. The program has several execution routines that are grouped into two main routines: the initialization and the main loop. [10]

In Figure 8, the flowchart of the startup routine is done in 4 steps. In step 1, the variables referring to RTC, IMU and GPS are generated. In step 2, all devices are checked for connection failures or devices faults. Soon later, the generation routine, step 3, of the CSV extension file is created on SD card.

In this routine, the date information is collected in the RTC, after that, the software checks if there is any "MMDDYY.csv" file on the SD card. If not, then the file is created. If there is a document, this routine goes directly to step 4. Finally, the initialization routine is finished in step 4.

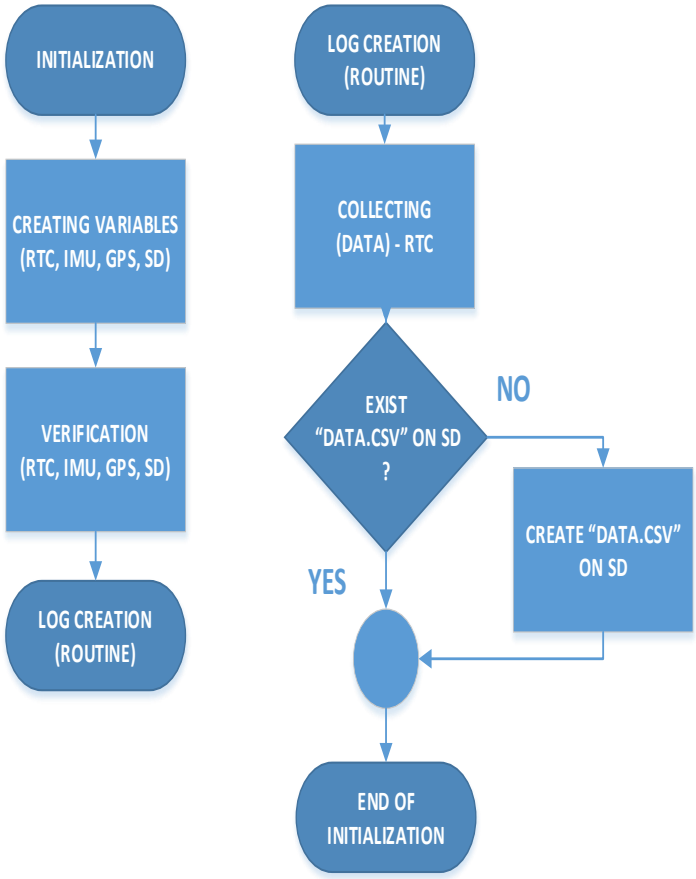

Figure 8: Initialization Flowchart

In Figure 9, just after the initialization routine, the main program initiates. This program has the RTC calibration steps, in which the information of the RTC collects the date and time and compares with the same data from the GPS. As the GPS is more accurate than the RTC, if the values of these entries are different, the RTC is overwritten with the current GPS value.

After the RTC calibration routine, data collection from RTC, GPS, IMU and OBD is completed. In this stage, the data of these devices are analyzed and treated.

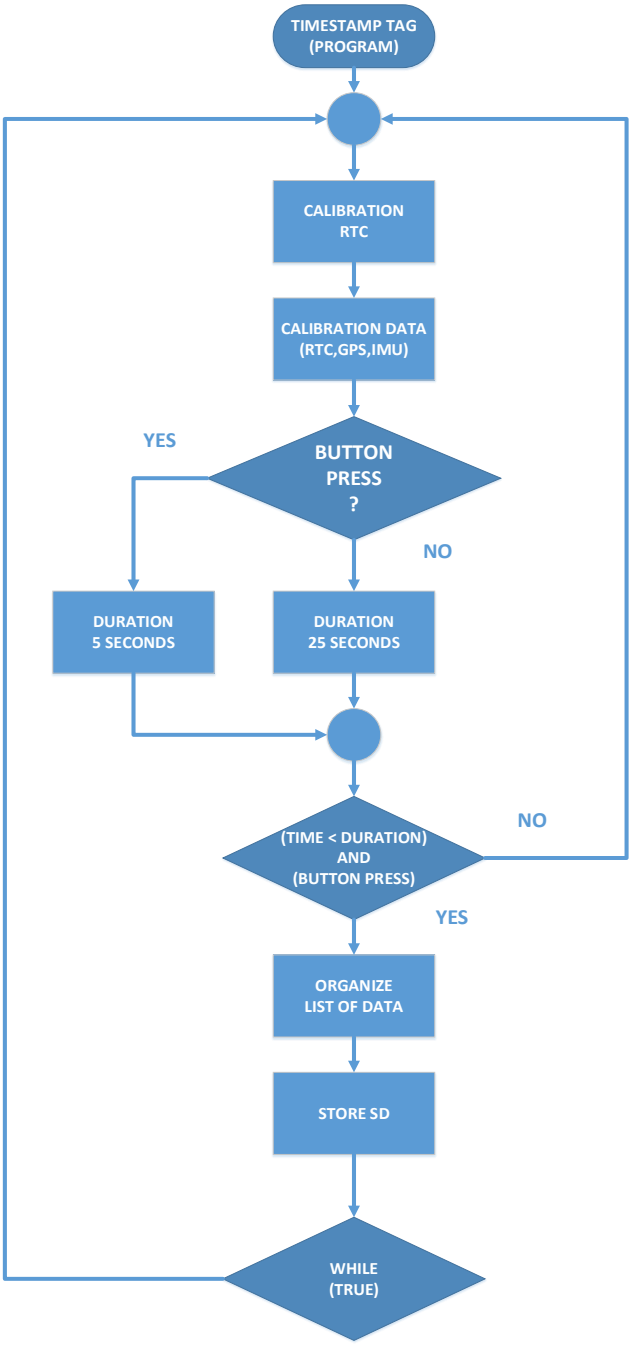

Figure 9: Main Routine Flowchart

This routine records 25-second on the memory buffer with a sample rate of 100 milliseconds. After collecting data from the sensors, the system verifies whether button is pressed. If the user does not press the button, the system will continue collecting information in the buffer. When the user presses the button, the Esp32 immediately records the last 25 seconds that already are in the buffer and additionally collects the next 5 seconds of data from the sensors. Additionally collects the next 5 seconds of data from the sensors. These 30 seconds of information are organized in the memory buffer and stored in a new "MMDDYY.csv" file, causing the main cycle to be restarted.

4. ANALYSIS AND RESULTS

TimeStamp ESP32 Platform system was validated by Bosch Brazil engineering team and successfully passed in all internal tests. For implementation, the final product was built using a different I²C, SPI and UART communication buses. Figure 10 shows the final assembly of the product.

Figure 10: TimeStamp ESP32 Platform

A connection between the vehicle and the microcontroller was done by establishing a communication between the Bluetooth interface from ESP32 and an OBD-Bluetooth adapter connected in the vehicle OBD port. Using this connection, it was possible to read the RPM, speed and throttle position information and save this data directly into the "MMDDYY.csv" file. In addition, these Bluetooth data was graphically checked using the Arduino IDE serial monitor, as shown in Figure 11.

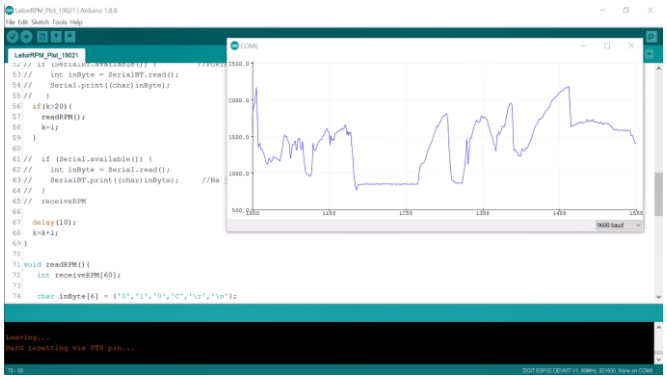

Figure 11: Real Time RPM Reading

In Figure 11, it is possible to see that RPM data varies gradually following the accelerator variation. In addition, it is possible to notice that there is a minimum value of RPM, that is the rotation in neutral.

5. CONCLUSIONS

In this project, the objective was to develop a tool that would improve the way developers, testers identify, receive and analyze system performance, and safety issues data during ADAS system development. Bosch Brazilian engineering team could reach this goal by developing and

building a low-cost mechanism based on an open sourced programming language and a hardware easily found in the market. The device was successfully validated and is planned to be implemented in customer projects development by middle of 2020.

As suggestion for a future work, the development of a software responsible for collecting real time data and displaying in a graphical interface using a GUI made in Python would be a very useful feature and hence an important upgrade in the platform.

REFERENCES

[1] D. Science, "Science Direct," [Online]. Available: https://www.sciencedirect.com/science/article/pii/B9780128037294000040. [Acesso em 20 05 2020].

[2] N. Euro, "Euro NCAP," [Online]. Available: https://www.euroncap.com/. [Acesso em 15 02 2020].

[3] L. NCAP, "Latin NCAP," [Online]. Available: https://www.latinncap.com/. [Acesso em 15 05 2020].

[4] S. Espressif, "ESP32-WROOM-32 Datasheet," 2019.

[5] I. Catalex, "MicroSD module," [Online]. Available: http://datalogger.pbworks.com/w/file/fetch/89507207/Datalogger%20-%20SD%20Memory%20Reader%20Datasheet.pdf. [Acesso em 15 02 2020].

[6] I. Maxim Integrated Product, "DS3231 Extremely Accurate I2C-Integrated RTC/TCXO/Crystal," 2015.

[7] I. InvenSense, "MPU-9250 Product Specification Revision 1.1," San Jose, 2016.

[8] [Online]. Available: https://www.sdcard.org/. [Acesso em 20 05 2020].

[9] i. Hackady, "DATALOGGER USES ESP32 AND ESP8266 LOW POWER MODES," Hackaday, [Online].Available:https://hackaday.com/2017/09/24/datalogger-uses-esp32-and-esp8266-low-power-modes/. [Acesso em 10 02 2020].

[10] I. uBlox, " u-blox 6 Receiver Description," 2013.