UNIVERSIDADE FEDERAL DE SANTA CATARINA
CENTRO DE CIÊNCIAS, TECNOLOGIAS E SAÚDE
CURSO DE GRADUAÇÃO EM ENGENHARIA DE COMPUTAÇÃO

Gabriel Bittencourt de Souza

Desenvolvimento de um Leitor OBD com Data Logger para Carros Preparados para Competições

Araranguá
2022

Gabriel Bittencourt de Souza

Desenvolvimento de um Leitor OBD com Data Logger para Carros Preparados para Competições

Trabalho de Conclusão de Curso submetida ao Curso de Graduação em Engenharia de Computação da Universidade Federal de Santa Catarina para a obtenção do título de Bacharel em Engenharia de Computação.
Orientador: Prof. Anderson Luiz Fernandes Perez, Dr.

Araranguá
2022

Ficha de identificação da obra elaborada pelo autor,
através do Programa de Geração Automática da Biblioteca Universitária da UFSC.

Souza, Gabriel Bittencourt de
Desenvolvimento de um Leitor OBD com Data Logger para Carros Preparados para Competições / Gabriel Bittencourt de Souza ; orientador, Anderson Luiz Fernandes Perez, 2022.
28 p.

Trabalho de Conclusão de Curso (graduação) - Universidade Federal de Santa Catarina, Campus Araranguá, Graduação em Engenharia de Computação, Araranguá, 2022.

Inclui referências.

1. Engenharia de Computação. 2. OBD2. 3. ESP32. 4. Datalog. 5. Automobilismo. I. Luiz Fernandes Perez, Anderson. II. Universidade Federal de Santa Catarina. Graduação em Engenharia de Computação. III. Título.

Gabriel Bittencourt de Souza

Desenvolvimento de um Leitor OBD com Data Logger para Carros Preparados para Competições

O presente trabalho em nível de bacharel foi avaliado e aprovado por banca examinadora composta pelos seguintes membros:

Documento assinado digitalmente
Analucia Schiaffino Morales
Data: 04/08/2022 15:45:49-0300
CPF: 622.256.420-87
Verifique as assinaturas em https://v.ufsc.br

Profª. Analúcia Schiaffino Morales, Drª.
Coordenadora de Curso

Documento assinado digitalmente
Fabio Rodrigues de La Rocha
Data: 04/08/2022 18:17:44-0300
CPF: 963.007.520-20
Verifique as assinaturas em https://v.ufsc.br

Prof. Fábio Rodriges de la Rocha, Dr.
Avaliador
Universidade Federal de Santa Catarina

Documento assinado digitalmente
Jim Lau
Data: 04/08/2022 12:55:01-0300
CPF: 613.464.702-00
Verifique as assinaturas em https://v.ufsc.br

Prof. Jim Lau, Dr.
Avaliador
Universidade Federal de Santa Catarina

Certificamos que esta é a **versão original e final** do trabalho de conclusão de curso que foi julgado adequado para obtenção do título de Bacharel em Engenharia de Computação.

Araranguá, 01 de agosto de 2022.

Profª. Analúcia Schiaffino Morales, Drª.
Coordenadora de Curso

Documento assinado digitalmente
Anderson Luiz Fernandes Perez
Data: 04/08/2022 13:27:03-0300
CPF: 020.803.459-58
Verifique as assinaturas em https://v.ufsc.br

Prof. Anderson Luiz Fernandes Perez, Dr.
Orientador

AGRADECIMENTOS

Agradeço a todos que participaram dessa jornada até a graduação. Agradeço a minha família a qual me amparou, me deu suporte e me guiou nessa jornada árdua. Agradeço em especial minha mãe Marciléia que não mediu esforços para que eu concluísse a graduação nesta renomada universidade me incentivando, me guiando e amparando. Me inspirando sobretudo como professora ao me mostrar desde cedo a importância da educação. A minha amada irmã Rafaela a qual me apoia e me inspira a ser uma pessoa melhor todos os dias. Ao meu padrasto Fábio o qual sempre me incentivou e me auxiliou durante estes anos.

A minha esposa Nathália, a qual a graduação me presenteou, agradeço pelo apoio e incentivo todos os dias me auxiliando também em boa parte de minha carreira acadêmica e na elaboração deste trabalho. Agradeço a todos os professores que me marcaram nesta jornada de uma forma ou de outra, em especial ao professor orientador Anderson Fernandes que desde o início da graduação se mostra um docente que incentiva tanto a pesquisa, quanto o empreendedorismo, sendo um grande apoiador na fundação de nossa Empresa Júnior.

Agradeço a todos os amigos que fiz na graduação, a aqueles que mesmo de forma passageira serviram de companhia ou auxílio nas atividades da engenharia ou que em momentos de descontração serviram de alívio para enfrentar esta jornada. Em especial agradeço a todos aqueles que as amizades se mostram perdurar para além da graduação, se mostrando verdadeiros amigos para a vida.

Agradeço a toda a equipe da Three Pixels Sistemas e da Octopus Atuadores Eletrônicos os quais me proporcionaram o estágio obrigatório e oportunidade de trabalho, contribuindo com o conhecimento necessário, o auxílio no desenvolvimento e em estrutura para o desenvolvimento deste dispositivo.

Agradeço também a todos os produtores de conteúdo automotivo, sejam eles, jornalistas, pilotos, engenheiros ou entusiastas, os quais contribuem muito para alimentar minha paixão por carros, automobilismo e engenharia.

Dedico este trabalho a minha família, minha esposa, meus amigos, meus colegas de trabalho, a todos os entusiastas automotivos e em especial a meu saudoso pai José Nilton que me acompanha em pensamento todos os dias.

Desenvolvimento de um Leitor OBD com Data Logger para Carros Preparados para Competições

Gabriel Bittencourt de Souza

2022, Julho

Resumo

A busca pelo controle de emissões de poluentes causadores das mudanças climáticas, em conjunto com a evolução da tecnologia embarcada, levou a indústria automotiva criar métodos de diagnósticos que objetivam o bom funcionamento do veículo. Estes dados passaram também a ser utilizados na análise de comportamento do piloto e do veículo em competições automobilísticas. Este trabalho visou desenvolver um dispositivo para aquisição de dados veiculares via *OBD2*, realizando um *Datalog* dos dados captados e apresentando posteriormente os resultados ao usuário. Para isto foi utilizado o microcontrolador ESP32, beneficiando-se de sua IDE nativa a ESP-IDF utilizando o sistema embarcado *FreeRTOS*, o dispositivo se mostrou eficaz na captura e envio dos dados, podendo viabilizar melhoras na pilotagem e no ajuste do veículo.

Palavras-chaves: CAN BUS. OBD2. ESP32. Eletrônica Embarcada Automotiva. Data Logger. Automobilismo.

Development of an OBD Reader with Data Logger for motorsport focused cars.

Gabriel Bittencourt de Souza

2022, Julho

Abstract

The search for the control of pollutant emissions that cause climate change, together with the evolution of embedded technology, led the automotive industry to create diagnostic methods that aim at the proper functioning of the vehicle. These data started to be used at the pilots behavior and the vehicle performance at motorsport competition. This academic work aimed to develop a OBD2 datalog device, aqcuiring the data and righ after showing the output to the user. To acomplish this development the ESP32 microcontroller was used, benefiting from its native ESP-IDF IDE using the FreeRTOS embedded system. The device proved to be effective in capturing and sending data, enabling improvements in the handling and adjustment of the vehicle.

Keywords: CAN BUS. OBD2. ESP32. Automotive On-board Electronics. Data Logger. Motorsport.

1 Introdução

O automobilismo pode ser considerado um dos esportes mais populares do mundo, com pelo menos 5 milhões de telespectadores por corrida, podendo ser comparado apenas com o futebol no quesito de alcance mundial. Mesmo com a popularidade do esporte, o estado da arte acerca do tema é pouco desenvolvido, enquanto o futebol possui de 300 a 400 artigos sobre atributos físicos e psicológicos dos atletas, o automobilismo possui apenas cerca de 73 artigos. (FERGUSSON, 2019)

A falta de estudos sobre o esporte pode ser justificada pela falsa percepção de que um piloto e sua equipe não são atletas, que não utilizam sua aptidão fisiológica, nem suas habilidades psicológicas para pilotar um carro de corrida e que apenas guiam o veículo. Entretanto inúmeros dados fisiológicos dos pilotos e demais membros da equipe, mostram que as taxas de batimentos cardíacos, que os níveis de açúcar no sangue e que o esforço físico durante uma corrida são compatíveis com a execução de uma atividade física e que são portanto considerados atletas. (FERGUSSON, 2019)

Durante uma transmissão de uma corrida de automobilismo, são apresentados aos espectadores diversos dados de telemetria dos veículos em pista, como a velocidade no ápice da curva, força G, pressão aplicada ao acelerador e ao freio. O piloto e os mecânicos da equipe tem acesso também à pressão de óleo, pressão de turbina, temperatura de arrefecimento, entre outros dados de sensores que são essenciais para o desempenho do veículo. Já em casos mais específicos, é possível obter dados fisiológicos do piloto, por exemplo a medição da glicose no sangue de Charlie Kimball, piloto da Fórmula *Indy*, devido este sofrer da doença diabetes (FERGUSSON, 2019).

Os dados de telemetria fornecem ao piloto, mecânicos e os fiscais de prova uma orientação sobre a performance do piloto e do carro. Para os treinadores dos pilotos, parâmetros sobre o comportamento destes durante a corrida são mais importantes, tais como pressão no acelerador ou freio, esterçamento do volante ou rotação do motor nas trocas de marcha. Também é relevante citar que algumas categorias impõem certos limites de acerto de calibração e de parâmetros do veículo, como por exemplo, a Copa Truck que limita a quantidade de fumaça gerada pelo caminhão a fim de controlar o nível de emissões de poluentes e acerto de potência (CBA, 2021).

Cabe destacar também, que o uso de automóveis movidos à combustão estão entre os principais responsáveis pela emissão de gases de efeito estufa. De acordo com o Inventário de emissões atmosféricas do transporte rodoviário de passageiros no município de São Paulo (2017), a quantidade de dióxido de carbono (CO_2) na atmosfera aumentou cerca de 35% desde a era industrial e os veículos são responsáveis por cerca de 72,6% destas emissões. Entretanto, mesmo com o aumento da frota de 2006 a 2017 de 10 milhões para aproximadamente 15 milhões de automóveis, a emissão de CO_2 caiu cerca de 49,8%, de 658.10^3 para 328.10^3t. (CETESB, 2017; MMA, 2013)

Desde o surgimento dos veículos automotores, a indústria busca sempre evoluir e disponibilizar ao consumidor o que há de melhor em seus produtos. Contudo, soluções que envolviam questões ambientais e a emissão de gases causadores do efeito estufa foram deixadas em segundo plano durante alguns anos, devido a grande demanda do mercado automotivo e a busca por maior agilidade na produção das montadoras.

Devido a desatenção das montadoras sobre questões ambientais, o controle das emissões de gases poluentes se tornou um ponto de regulamentação e de cobrança pelos órgãos governamentais responsáveis, tal como a *California Air Resources Board* (CARB), que em 1982 iniciou o desenvolvimento de uma legislação a qual exigia que a partir do ano de 1988, todo veículo a combustão vendido na Califórnia saísse de fábrica com um sistema de diagnóstico a bordo capaz de detectar falhas no motor que pudessem acarretar no aumento das emissões de CO_2 e outros gases. (ELECTRONICS, 2021)

Sistemas de monitoramento e diagnóstico eletrônico veicular, os *On-Board Diagnostics - OBD*, se tornaram obrigatórios nos Estados Unidos da América a partir do ano 1996 (EPA, 2012), tornando padrão em todos os veículos a combustão posteriormente a este ano. Os dispositivos OBD's foram padronizados e recomendados pela *Society of Automotive Engineers - SAE* em conjunto com a *International Organization for Standardization - ISO* e *Environmental Protection Agency - EPA* para todos os fabricantes de automóveis da Europa, sendo implementado em vários países ao longo do tempo e se tornando obrigatório no Brasil em 2010. (ELECTRONICS, 2021).

Com os sistemas de monitoramento pode-se obter os mais diversos dados do veículo, desde informações em tempo real como temperatura do sistema de admissão, temperatura do óleo do motor, eficiência da queima de combustível, velocidade de rotação do motor

entre outros e até mesmo dados armazenados. Tais dados são por exemplo os códigos de erro, visto que todas as centrais do carro se comunicam e assim que se tem algo fora do esperado sendo uma falha mecânica ou elétrica a central armazena esta informação para que possa ser lida depois, geralmente durante a revisão na concessionaria do veículo ou em oficina especializada.

A partir dos dados obtidos pelos sistemas OBD, é possível realizar a telemetria do comportamento do motorista e do desempenho do veículo. Estes dados podem ser utilizados para análise de comportamento de pilotos de automobilismo, assim como para gerenciamento de grandes frotas. Estas análises podem ser realizadas por meio de relatórios de falhas ou de comportamento, ou através de dados obtidos que não são apresentados originalmente no painel do veículo, mas que podem ser de interesse ao motorista, piloto ou gestor de frotas.

Visto brevemente as diversas possibilidades de uso dos dados obtidos através da telemetria, o objetivo deste trabalho é desenvolver um dispositivo capaz realizar *Data Logging*, que consiste em um dispositivo eletrônico que automaticamente capta, grava e analisa informações com alta velocidade e eficiência, durante um teste ou medição, (BADHIYE et al., 2011), afim de obter informações referentes ao comportamento do veículo e do motorista, além de demonstrar como estes dados podem ser úteis na análise de desempenho de ambos.

O dispositivo que será apresentado neste trabalho é baseado em um microcontrolador da Espressif, o ESP32 e demais periféricos para a comunicação e alimentação do dispositivo, levando em consideração o sistema nativo da ESP32 o ESP-IDF e utilizando *Real-time Operating System for Microcontrollers - FreeRTOS*^{TM} que é um sistema operacional embarcado. Também é utilizado um transciever para estabelecer a comunicação via rede *Controller Area Network* (CAN) a qual se baseia o padrão OBD.

2 Objetivo Geral

O presente trabalho tem como objetivo estudar os principais aspectos acerca do desenvolvimento de um dispositivo que ligado à central eletrônica do carro, que permita a leitura e transmissão de diversos parâmetros dos veículos, utilizando um microcontrolador ESP32 com base em um sistema operacional de tempo real. Esses estudos visam contribuir para o aperfeiçoamento do estado da arte da temática em questão.

2.1 Objetivos específicos

A fim de cumprir o objetivo principal deste trabalho, alguns objetivos específicos são propostos:

- Obter melhor conhecimento do estado da arte dos temas abordados na pesquisa, os quais compreendem aspectos relacionados à sistemas operacionais embarcados, arquitetura da rede CAN, padrão OBD e os parâmetros veiculares relevantes para análise de desempenho do veículo e do motorista.
- Desenvolver um protótipo funcional para *data logging* de parâmetros veiculares utilizando o padrão *OBD2*. Utilizar o sistema nativo da ESP32 o ESP-IDF e o *Real-time Operating System for Microcontrollers - FreeRTOS*^{TM}.

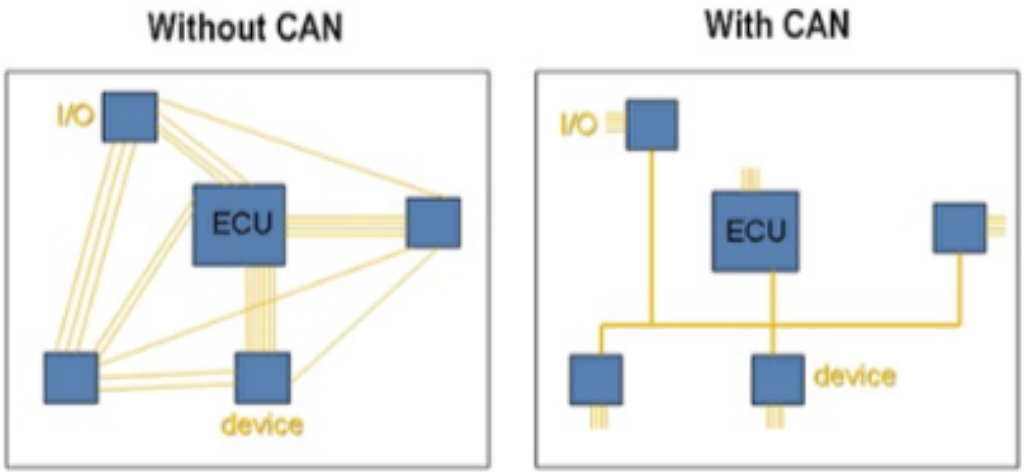
- Analisar as funcionalidades e limitações impostas pela indústria, ou não suportadas pelo dispositivo.
- Demonstrar pontos de melhoria na pilotagem em situação de competição automobilística vindos da aquisição dos dados a serem capturados pelo dispositivo.

3 Referencial teórico

3.1 CAN - Controller Area Network

A integração da computação moderna e das tecnologias de comunicação aos automóveis, promoveu melhorias significativas relacionadas ao conforto e a segurança dos passageiros e do motorista. Tal integração se deve ao principal meio de comunicação embarcado, a Rede *CAN - Controler Area Network*, que ao conectar todas as centrais eletrônicas, *Eletronic Controller Unity - ECU's*, do automóvel, pode acompanhar e trocar informações sobre o comportamentos do motorista e das condições do veículo. A rede CAN pode ser melhor compreendida através da Figura 1, que ilustra a comparação da mesma com outras redes. (QIN; YAN; JI, 2021).

Figura 1 – Comparação entre uma rede CAN e outras redes.

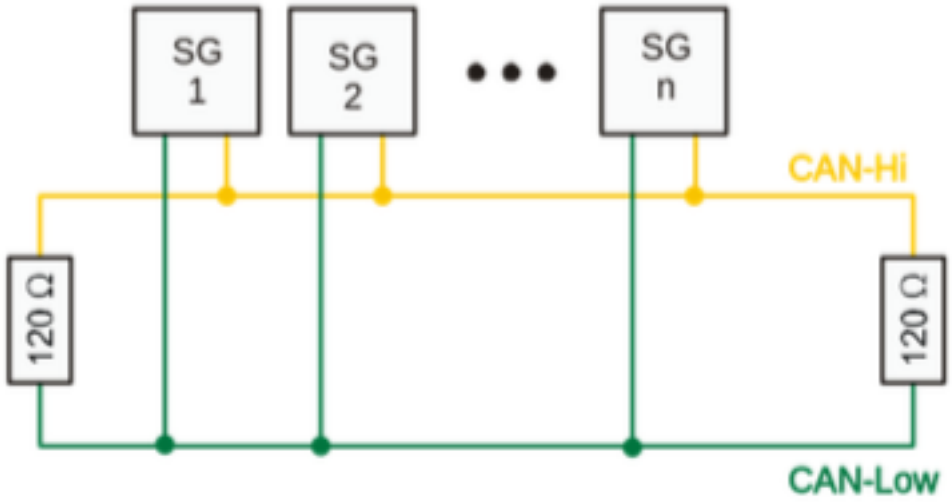


(NI, 2022)

A Figura 2 ilustra a arquitetura da rede CAN, a qual permite sua ampliação sem que haja modificação da rede pré existente, sendo somente necessário o término da rede com dois resistores de 120 ohm. A transmissão do sinal é realizada através de um par de fios trançados, correspondendo o *CAN HIGH* ao 1 e o *CAN LOW* ao 0 sendo respectivamente o recessivo e o dominante. (KING; YU, 2017)

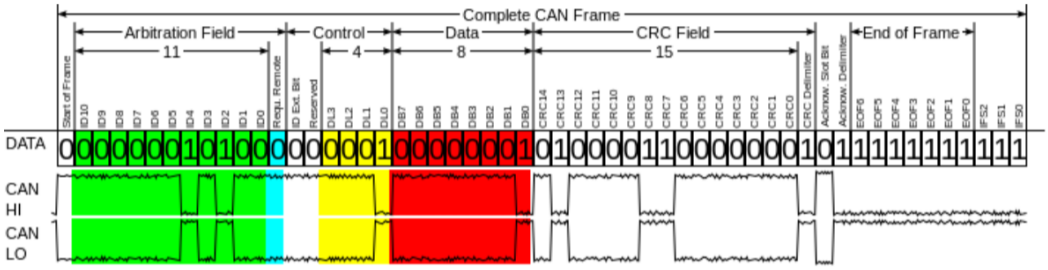
A transmissão de dados na rede CAN é realizada por meio de pacotes, sendo composta pelo identificador do pacote, o código de tamanho do dado e pelo campo contendo o dado. Normalmente o identificador *frame ID* é composto por 11 bits e pode receber um valor entre zero e 2047, como é representado em hexadecimal, até 0x7FF. O *frame ID* também é responsável por identificar o receptor da informação, a fim de evitar colisões entre as mensagens no barramento. (KING; YU, 2017)

Figura 2 – Arquitetura da rede CAN



(KING; YU, 2017)

Figura 3 – Estrutura de um quadro CAN



(KING; YU, 2017)

O *frame ID* indica também a prioridade do pacote, quanto menor seu valor maior sua prioridade, se seu ID for zero, terá prioridade máxima.

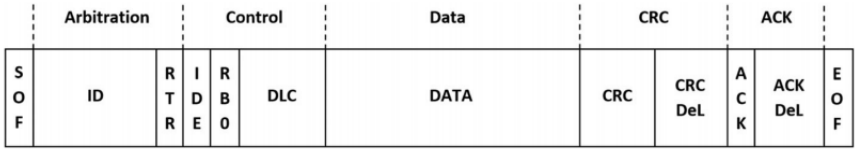
A Figura 3 ilustra os campos compõe um quadro CAN, os quais são formados por:

- *Start of Frame - SOF*: Contém um bit dominante identificando o início de uma mensagem CAN notificando todos os nós da rede, iniciando a transmissão da mensagem CAN.
- *Arbitration*: É identificador que pode conter 11 bits ou até 29 bits no formato extendido. Cada identificador poderá ser usado como um filtro para os demais nós a fim de identificar se a mensagem pode ser utilizada por eles ou não.
- *Control*: O campo *Control* ou verificação, informa ao receptor se todos os pacotes pretendidos chegaram com sucesso.
- *Data*: Campo de dados, contém a informação que foi enviada, pode conter de 0 a 8 bits.

- *CRC*: Campo de verificação da validade dos pacotes que consiste em um método de detecção de falhas de 15-bits e é realizado por meio de um *Cyclic Redundancy Code*.
- *Acknowledge ACK*: Campo de confirmação que certifica que o recebedor recebeu os pacotes de forma correta. Assim que é detectado algum erro durante o processo de transmissão, o transmissor é notificado imediatamente enviando os pacotes novamente ao destinatário.
- *End of Frame - EOF* : Indica o fim do *frame* com um bit recessivo.(LOKMAN; OTHMAN; ABU-BAKAR, 2019)

A Figura 4 ilustra de uma outra forma a estrutura de um quadro CAN.

Figura 4 – Quadro CAN



(LOKMAN; OTHMAN; ABU-BAKAR, 2019)

O campo responsável pela transmissão de informações é o de CAN *Data*, todos os outros campos servem para garantir a correta comunicação do sistema.

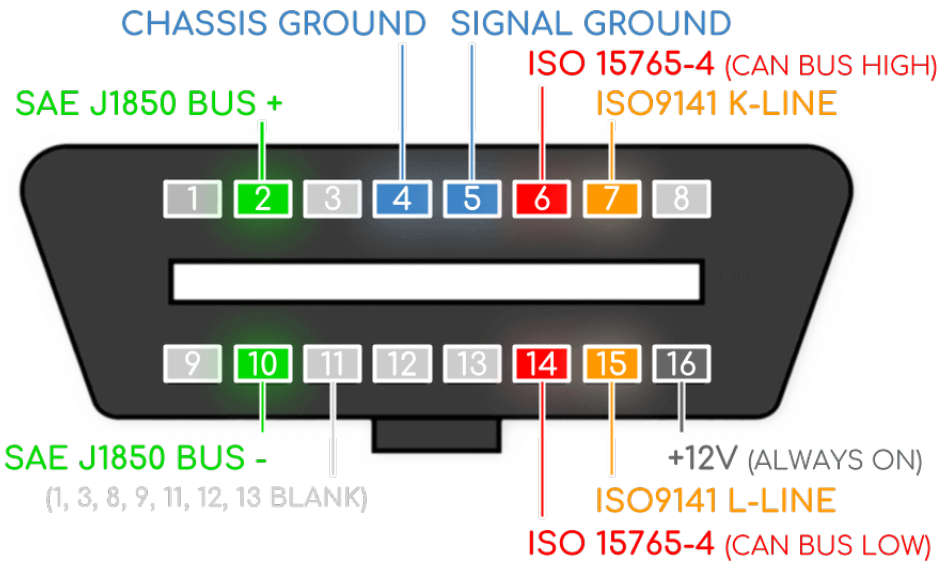
Como aplicações tem-se principalmente o uso automotivo, devido a robustez da rede a interferências, geradas por inúmeros sensores, atuadores e diferenças de temperatura. Essa tecnologia também é utilizada em larga escala pela indústria no caso de (DIAZ et al., 2008) a aplicação em uma rede de iluminação por exemplo e até o seu uso em comunicação interna de satélites (JANSCHEK; BRAUNE, 2000).

3.2 On-Board Diagnostics - OBD

O *On-Board Diagnostics - OBD* consiste em um sistema de auto-diagnóstico presente em veículos de pequeno porte, o qual permite apresentar informações simultâneas sobre o estado de vários componentes do veículo, bem como armazenar códigos de falhas nos mesmos. Desenvolvidos pelas montadoras entre os anos 70 e 80, no início destes sistemas de diagnósticos poderiam ser obtidas poucas informações sobre o estado dos veículos. A principal utilidade do *OBD* era reportar a existência de problemas em sensores relacionados à consumo e emissões, não sendo capaz ainda de constatar e indicar o problema existente. Para reportar, por exemplo, o problema um, a luz CHECK ENGINE (MIL) era acionada no painel do veículo. (Miguel Gomes Ribeiro, 2015)

Para regulamentar o sistema *OBD* foi desenvolvida a norma *On Board Diagnostics 2 OBD2* nos Estados Unidos e na Europa em 1996. Esta norma possui uma série de padrões, estabelece o padrão do conector, formato das mensagens e sinais elétricos. A partir da padronização física do conector, do seu formato e localização, os sistemas de leitura *OBD* passaram a não necessitar de adaptadores, tornando-se as ferramentas de diagnóstico mais baratas do mercado. O padrão utilizado pelo conector OBD2 é a norma SAE J1962 ilustrado na Figura 5. No Brasil, se tornou obrigatória nas montadoras somente em 2010. (OSS, 2018)

Figura 5 – Conector padrão J1962



(ELECTRONICS, 2021)

Para requisitar dados do veículo são utilizados os códigos PID's estabelecidos pela última norma SAE J1979. Onde existem códigos que as montadoras são obrigadas a normatizar e tornar padrão em todos os veículos, já outros são estabelecidos pela montadora e são proprietários de cada modelo de veículo. A SAE J1979 define como padrão os seguintes modos de operação para diagnóstico, sendo representados por dois números hexadecimais, como ilustrado na Figura 6: (AMARASINGHE et al., 2016)

Figura 6 – Modos de operação

Mode (hex)	Description
01	Show current data
02	Show freeze frame data
03	Show stored Diagnostic Trouble Codes
04	Clear Diagnostic Trouble Codes and stored values
05	Test results, oxygen sensor monitoring (non CAN only
06	Test results, other component/system monitoring (Test results, oxygen sensor monitoring for CAN only)
07	Show pending Diagnostic Trouble Codes (detected during current or last driving cycle)
08	Control operation of On-Board component/system
09	Request vehicle information
0A	Permanent Diagnostic Trouble Codes (DTCs) (Cleared DTCs)

(AMARASINGHE et al., 2016)

- Modo 01: Mostra o dado em tempo real baseado na lista de ID's suportados;
- Modo 02: Mostra o *frame* congelado dos dados;
- Modo 03: Lista os códigos de falha gerados e armazenados *DTC's*, normalmente associados com emissões e que geram a indicação da luz de *check engine* ou *Malfunction Indication Light- MIL*;
- Modo 04: Realiza a limpeza da memória onde os códigos de erros *DTC's* foram armazenados;
- Modo 05: Apresenta os resultados dos testes do sensor de Oxigênio;
- Modo 06: Apresenta os resultados dos testes de outros componentes;
- Modo 07: Apresenta os códigos de erros *DTC's* pendentes, ou seja, gerados no presente ciclo de utilização do veículo;
- Modo 08: Realiza o controle de equipamentos e sistemas embarcados do veículo, podendo alterar seus parâmetros de funcionamento;
- Modo 09: Oferece informações gerais sobre o veículo, como versão de software instalado na ECU, bem como o *Vehicle Identification Number - VIN* que se trada de um identificador único de veículos e caminhões, número de ignições, relatórios sobre o catalisador e testes de autodiagnóstico (OSS, 2018).

Dentro desses modos de operação tem-se os códigos de parâmetros, *Parameter ID's - PID's*, que se estendem a cerca de 256 *PID's*, sendo também representados por um número hexadecimal de dois dígitos (OSS, 2018). Como na operação 01 estão os dados em tempo real vindos da ECU em valores também hexadecimais deve-se além de identificá-los, convertê-los para os valores condizentes com as unidade de medida. Na Figura 7 são apresentados alguns códigos PID's.

Para entender a relação entre OBD2 e rede CAN pode-se estabelecer uma analogia, onde a rede CAN é o meio de comunicação, por exemplo um telefone e o OBD2 seria o padrão de linguagem para a troca das informações, onde uma representação do quadro OBD2 pode ser visualizado de Figura 8.

No campo data de bytes representa o dado recebido, o modo de operação solicitado, o PID correspondente, e os campos A,B,C E D correspondem aos bytes mais significantes respectivamente, sendo utilizados nas fórmulas da lista de PID's para conversão dos dados recebidos.

3.3 FreeRTOS

Um Sistema Operacional Embarcado de Tempo Real, *Real Time Operational Sytem - RTOS*, é um sistema operacional embarcado cuja a capacidade de sincronização e execução entre múltiplas tarefas, comunicação entre processos, manuseio de interrupções e gerenciamento de memória ocorre em tempo real. No caso da ESP32, sua IDE nativa se baseia no *FreeRTOS*, um RTOS grátis e *open source* desenvolvido pela Real Time Engineers LTD.(M. Desai; J. Patoliya, 2018)

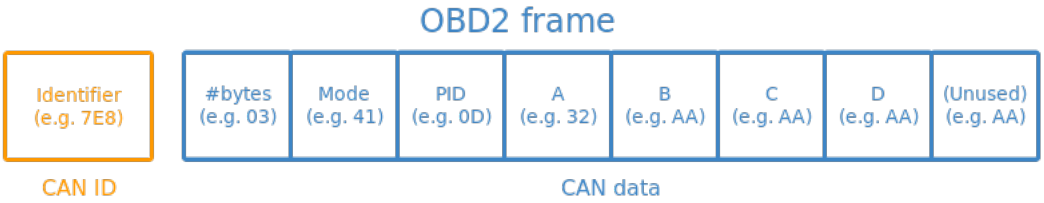
O Free RTOS é um Kernel de RTOS para sistemas embarcados que possui em sua arquitetura alguns recursos básicos, tais como: utilização de tempo ocioso, manuseio de

Figura 7 – Modos de operação

PID (hex)	Description	Output bytes	Equation	Range & Unit
00	PIDs supported (01 - 20)	4	-	-
04	Engine Load	1	A*100/255	0-100 %
05	Engine Coolant Temperature	1	A - 40	-40 - 215 °C
0C	Engine RPM	2	((A*256)+B)/4	0 - 16,383.75 rpm
0D	Vehicle Speed	1	A	0 - 255 km/h
52	Ethanol fuel percentage	1	A*100/255	0 - 100 %
5C	Engine oil temperature	1	A - 40	-40 - 210 °C
5E	Engine Fuel Rate	2	((A*256)+B)*0 .05	0 - 3212.75 L/h

(AMARASINGHE et al., 2016)

Figura 8 – Quadro OBD2



(ELECTRONICS, 2021)

interrupção de forma flexível, gestão de filas, rotina de interrupção de serviço, manipulador de tarefas, tarefas periódicas e mecanismos de sincronização de tarefas.

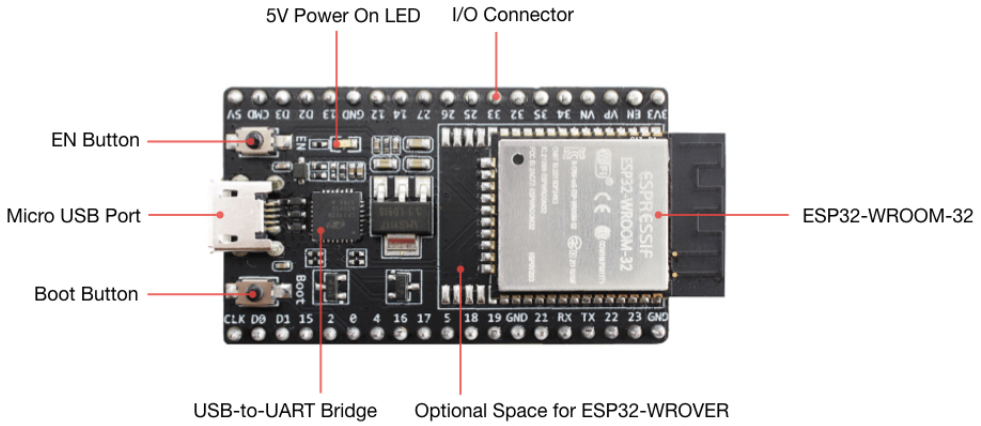
Uma das vantagens do *FreeRTOS* é a versatilidade ao executar um número ilimitado de tarefas, sendo que em um momento específico de tempo, apenas uma tarefa é executada por vez. Uma tarefa, ou do Inglês *Task*, é um pequeno trabalho a ser realizado em um prazo limitado e passa por uma série de estados ao ser executada, para que o Sistema seja capaz de executar todas as tarefas estabelecidas.

3.4 Microcontrolador ESP32

Para o desenvolvimento de um protótipo do dispositivo ou um produto final, é necessário levar em consideração o microcontrolador a ser utilizado, pois a partir do microcontrolador são definidos tanto os custos do projeto quanto a linguagem de implementação e capacidade de captura e apresentação dos dados. Uma das melhores alternativas do mercado é a ESP32 da *ESPRESSIF*, sendo esta uma solução muito versátil e completa para o desenvolvimento de hardware sendo eles protótipos ou produtos finais. Contendo

Wi-Fi e *Bluetooth* integrado, bem como as interfaces: SD card, UART, SPI, SDIO, I2C, LED PWM, Motor PWM, I2S, Infra Vermelho, contador de pulso, GPIO, sensor de *touch* capacitivo, ADC e DAC, inculindo tambem o suporte a rede CAN. (ESPRESSIF, 2021)

Figura 9 – ESP32-DevKitC V4 com o módulo ESP32-WROOM-32



(ESPRESSIF, 2022)

O Kit de desenvolvimento da ESP32 utilizado será o *DevKitC*, apresentado na Figura 9, o qual facilita a prototipagem ao ser utilizada com *protoboard* mesmo contento apenas 30 pinos em comparação com os 38 pinos do módulo *WROOM-32*, pois conta com conversor e conector USB para a comunicação UART, LED's e botões para interface e pinos PCI, o layout de pinos é apresentado na Figura 10.

Como recursos o microcontrolador ESP32 possui um microprocessador *Xtensa*® dual-core de 32-bit LX6 com velocidade de até 240 MHz, 448 KB de memória ROM para o *booting* e controle de núcleos e 520 KB SRAM dados e instruções. Possui 4 MB de memória flash SPI, Wi-Fi 802.11b/g/n, *Bluetooth* V4.2 BR/EDR e LE, Tensão de operação entre 3,0 a 3,6 V e temperatura de operação de –40 a 85 °C. (ESPRESSIF, 2021)

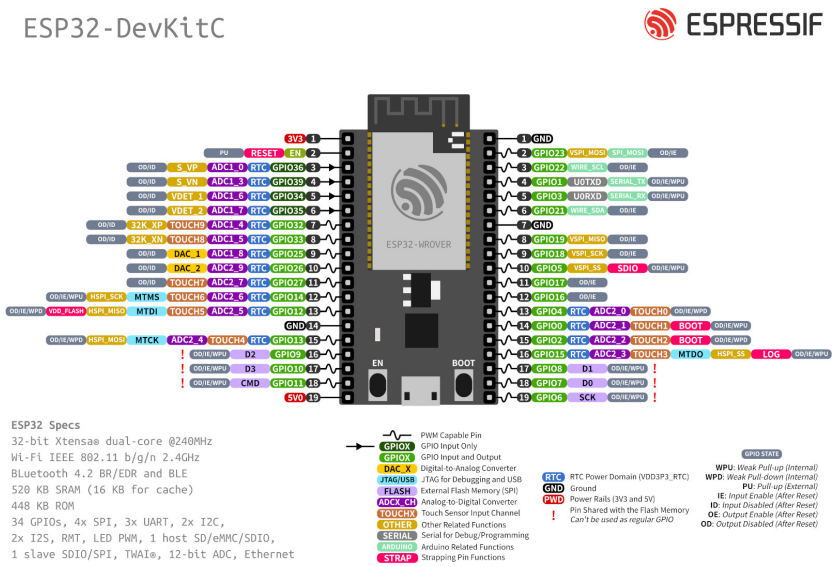
3.5 MCP2551

Para efetivar a interface entre a comunicação UART da ESP32 e o protocolo CAN deve-se utilizar um *transciever* CAN. Devida a ampla utilização, fácil acesso e pela tensão de funcionamento ser entre 4,5V e 5,5V, podendo ser alimentado pela própria ESP32, estando conectada a porta USB de um computador. Neste trabalho optou-se pelo o módulo MCP2551 fabricado pela Microchip, o qual permite taxas de transferência de até 1 Mbit/s. O módulo MCP2551 pode ser visualizado na Figura 11.

3.6 Trabalhos Relacionados

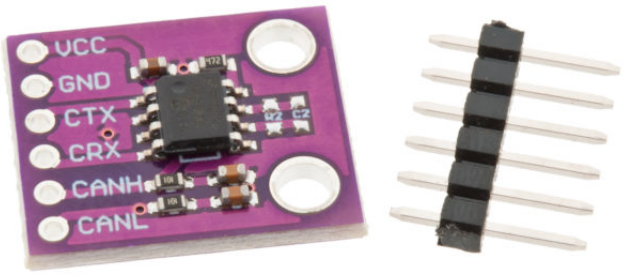
Pesquisando trabalhos sobre o tema, nota-se que os autores em sua grande maioria utilizam-se do dispositivo *ELM327* para captura dos dados, o qual é fabricado pela empresa *ELM Electronics*, realizando o tratamento dos dados em outras plataformas desenvolvidas

Figura 10 – ESP32-DevKitC Layout de pinos



(ESPRESSIF, 2022)

Figura 11 – Módulo MCP2551



(SMARTKITS, 2022)

posteriormente. Como pode-se verificar no trabalho de (OSS, 2018), o qual realiza a leitura dos dados captados pelo *ELM327* e os envia um *Smartphone* via *Bluetooth*. O autor utiliza da plataforma *App Inventor* para apresentar os dados em um *Web Server* a fim de visualizar os dados a medida que são captados, realizar o diagnóstico de erros indicados pela ECU do veículo e apagar o registro de tais erros a fim de reparar o veículo.

Já o autor (Miguel Gomes Ribeiro, 2015) realiza a integração do dispositivo *FM500-Blue* da empresa *BCE (Baltic Car Equipment)* para auxiliar na gestão de frotas, em conjunto com um dispositivo GPS para a integração com o sistema *GEOCAR*. O sistema desenvolvido monitora parâmetros como o constante uso de altas rotações no

motor, consumo de combustível excessivo, temperatura de operação do motor e alertas de inspeções ou manutenções a vencer. Apresentando também a localização exata do veículo.

No trabalho de (AMARASINGHE et al., 2016), foi utilizado o já mencionado *ELM327* para a captura de dados OBD2, afim de realizar a telemetria do veículo para a detecção de anomalias no comportamento de direção por parte do motorista ou de falhas causadas por possíveis defeitos, realizando até mesmo a predição de defeito no sensor de oxigênio do veículo e no sensor de massa de ar. Os dados são captados pelo *ELM327* e enviados via *bluetooth* para um *smartphone* o qual através de rede 3G envia a um servidor que realiza a análise dos dados em Nuvem.

No trabalho de (SOUZA, 2019), foi utilizado um módulo contendo o microcontrolador *dsPIC33EP64MC502* programando o mesmo na plataforma *MPLAB* para requisitar os dados a ECU, os dados são coletados através da conexão *USB* com um computador, apresentando-os posteriormente em gráficos para análise.

Ao realizar a análise de trabalhos relacionados, pode-se notar que em sua grande maioria são utilizados dispositivos prontos, realizando apenas a demonstração e análise dos dados obtidos pelo dispositivo com os mais variados objetivos. Nos poucos casos que o autor realiza o desenvolvimento total do dispositivo, a apresentação dos dados não é satisfatória ou os resultados obtidos não são expressivos.

Neste trabalho o diferencial é o desenvolvimento de um dispositivo utilizando apenas os módulos para prototipagem e a plataforma de compartilhamento dos dados já prontos, desenvolvendo toda a implementação do dispositivo em sua plataforma nativa, apresentando inúmeros recursos e possibilidades oferecidas pela ESP32.

3.7 Cenário Comercial ou Mercadológico

No Brasil existem inúmeros dispositivos de interface OBD2, em sua maioria baseados no controlador ELM327, o qual possui plena capacidade de comunicação com o protocolo CAN e vários outros protocolos, porém como o controlador suporta vários protocolos, se torna muito abrangente e por apenas converter as informações para RS232 acaba necessitando de outros periféricos para sua comunicação, seja ela *bluetooth*, *Wi-Fi* ou até mesmo uma interface visual.

Atualmente a linha de injeções programáveis da marca *FuelTech* se mostra referência no gerenciamento e controle de um motor de combustão interna realizando tanto o acerto do veículo, como o controle do sistema de combustível, sistema de ignição e comando de periféricos, quanto ao realizar o *Logging* de todos os parâmetros controlados por ela, podendo também realizar o log da rede CAN em paralelo a instalação da ECU, porém demanda a realização de instalação especializada, tando para controle do motor quanto para o log.

O *Datalogger* AQ-1 OBD2 da marca AEM, referência na gestão de motores de competição, permite a gravação de vários parâmetros, selecionáveis por meio de uma aplicação para *Windows*. O equipamento é composto por um conjunto de fios que se conectam até a central , porém necessita de conexão física para o acesso ao Log que fica armazenado no dispositivo, dependendo também de um software próprio para utilização.

Como método de medição de velocidade e parâmetros do ambiente, pode-se citar o Draggy, que utiliza de sinal de GPS para a aquisição de dados como, velocidade, inclinação, aceleração de 0-100Km/h , 100-200 Km/h e força G. O equipamento utiliza uma conexão

bluetooth com um celular para apresentar os dados, contando também com uma rede social para compartilhamento dos logs.

4 Metodologia de Desenvolvimento

O dispositivo proposto neste trabalho consistirá em um microcontrolador para tratamento e manipulação dos dados em conjunto com um *transciever* CAN para a adequação de tensões e interface do sinal para rede serial para o microcontrolador.

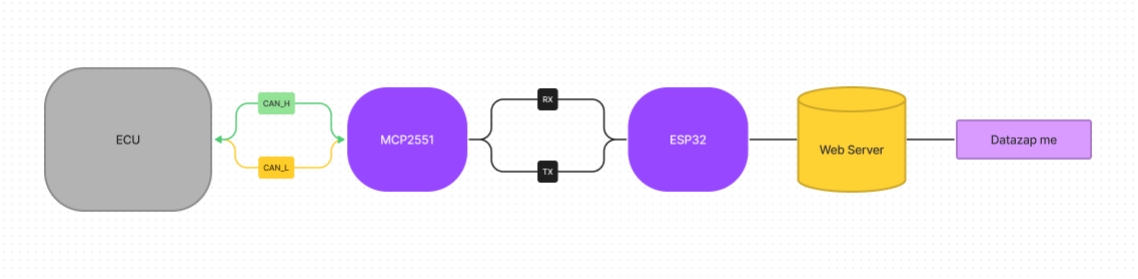
O dispositivo então se conectará através de uma rede *Wi-Fi*, podendo assim enviar o registro dos dados capturados e armazenados em um arquivo no formato *Comma-separated Values - CSV* a um servidor através de uma conexão HTTP, podendo ser analisado em qualquer interpretador de dados compatível posteriormente.

Será utilizado o site *Datazap.me* que consiste em uma plataforma de compartilhamento de LOGs de dados veiculares, tanto para diagnóstico de defeitos, quanto para análise de performance.

Como apresentado na Figura 12 o fluxo de comunicação do dispositivo consistira em:

- Uma consulta a ECU do veículo pela ESP32, sendo enviado por meio da UART até o trancsiever MCP2551, o qual converte as interfaces para a comunicação com a ECU por meio do protocolo CAN.
- Após a coleta dos dados pela ESP32, a mesma se conecta por meio de uma rede Wi-Fi, gerada pelo *smartphone* do usuário a um *Web Server* remoto e envia o registro dos dados por meio de uma requisição HTTP.
- É consultado então o servidor para realizar o *upload* do log para a plataforma a serem interpretados, no caso a *Datazap.me*.

Figura 12 – Fluxo de comunicação



Próprio Autor

Como o dispositivo se trata uma interface de *Hardware* e *Firmware*, deve-se ser certificado o pleno funcionamento dos recursos a serem aplicados, dentre eles sendo a IDE nativa da ESP32 a ESP-IDF disponibilizada pela própria fabricante a *Espressif*, bem como dos módulos a serem utilizados, tando o microcontrolador quanto o dispositivo para interface CAN.

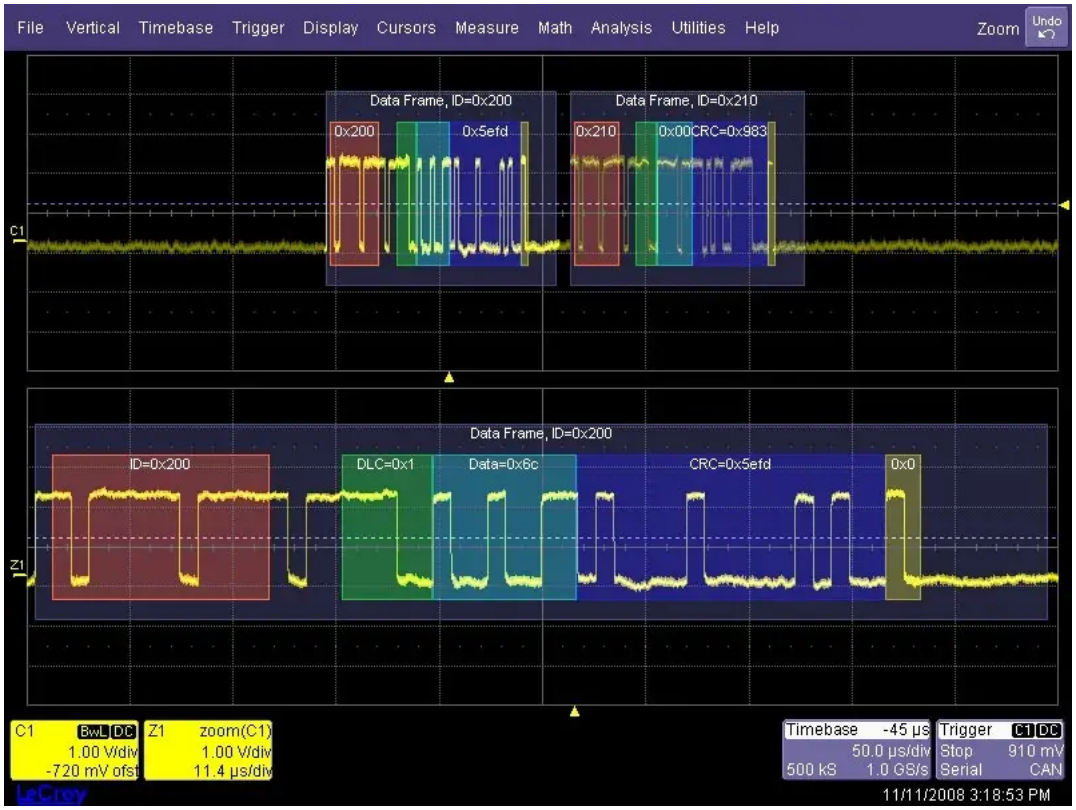
Como a *Espressif* fornece um grande acervo de bibliotecas através de sua *IDE - Integrated Development Enviroment* devido aos vários periféricos a serem conectados na placa ESP32 o primeiro passo foi instalar a IDE nativa fornecida no site da fabricante e configurá-la com um projeto inicial para a verificação do funcionamento.

Com a verificação da IDE completa foi validada a biblioteca CAN estabelecendo a escrita e leitura de mensagens entre o mesmo dispositivo através das portas de comunicação serial RX e TX. Foi realizado um teste conectando os pinos de TX ao próprio pino de RX da ESP32, verificando assim a montagem do *Driver CAN* e o envio e recebimento da mensagem, validando o funcionamento do periférico até então.

Para a comunicação via protocolo CAN será necessário o uso do *transciever* MCP2551, o qual fará a interface entre as portas TX e RX da esp32 para os sinais *CAN HIGH* e *CAN LOW*, necessitando também o aterramento em comum dos dispositivos a serem conectados.

Através da leitura do sinal gerado por meio de um osciloscópio, seria possível identificar o quadro de informações que está trafegando pela rede, afim de identificar o inicio da mensagem e os campos correspondentes, como ilustra a Figura 13.

Figura 13 – Visualização da comunicação CAN por meio de um osciloscópio



(TELEDYNE, 2022)

Após a verificação do funcionamento da biblioteca e a comunicação via CAN, foi realizada a validação em um veículo. A ligação das portas se deu da seguinte forma: As portas CAN_H E CAN_L do *transciever* foram conectadas na porta OBD2 do veículo, nos pinos 14 e 6 respectivamente, bem como o GND para o aterramento do sinal no pino

13 da porta OBD2. Foram configurados os parâmetros do veículo a ser testado, neste caso um *Volkswagen* Polo MSI 1.6 16V 2018 que possui a injeção BOSCH 032906032L. Dentre os parâmetros estão a taxa de comunicação, que pode ser de 250Kbps, 500 Kbps e 1Mbps.

Após a realização de um teste, foi possível identificar que a taxa compatível com o veículo é de 500 Kbps, já entre o *transciever* e o microcontrolador foram ligados os pinos de comunicação RX e TX as portas 21 e 22 da ESP32. Inicialmente foi utilizado o método somente de leitura, obtendo uma espécie de *"Hello World"* da ECU do veículo.

Para a obtenção das outras informações desejadas da injeção do veículo, é necessário enviar uma mensagem utilizando a formatação correta do padrão OBD2. Por isso, dentro do modo de escrita devem-se estabelecer algumas informações fundamentais para que seja recebida uma resposta da ECU, primeiro em relação ao CAN ID que seria o responsável por distinguir entre as mensagens de requisição que possuem o ID 7DF e as mensagens de resposta que possuem ID's do 7E8 ao 7EF em hexadecimal. Após isso deve-se preencher totalmente o campo de DATA composto pelos campos *Lenght*, *Mode*, PID, A, B, C e D.

Como exemplo na Figura 14, foi requisitado os dados de velocidade de rotação do motor em RPM. Visto que sempre que requisitado uma informação deve-se preencher o campo *Lenght* com o tamanho 2 já utilizando o modo MODE CURRENT (01) e o PID que nos interessa seria o 0C, os demais campos serão preenchidos com 55, pois não são utilizados no modo de *request*. A Figura 14 ilustra uma demonstração dos dados enviados como *request* e os dados recebidos como resposta.

Foi utilizado também o recuso de multitarefas do *FreeRTOS*, utilizando uma tarefa para realizar o envio da requisição, outra tarefa para receber a resposta gerada pela ECU e uma terceira tarefa para realizar o envio do log via HTTP, não necessitando nesse caso da utilização de outro recurso de gerenciamento de tarefas do *FreeRTOS*.

Figura 14 – Requests OBD2

BYTE		BYTE 0	BYTE 1	BYTE 2	BYTE 3	BYTE 4	BYTE 5	BYTE 6	BYTE 7
DEFINIÇÃO	CAN ID	DATA LENGH	SERVICE MODE	PID	A	B	C	D	NA
PERGUNTA	7DF	02	01	0C	55	55	55	55	55
RESPOSTA	7E8	02	01	0C	12	34	55	55	55

Próprio Autor

A resposta referente a requisição contém os seguintes valores: 07 E8 01 0C 12 34 44 FF FF FF. O primeiro e segundo campos referem-se a ID de resposta da ECU, o quarto campo é o PID de RPM, o campo A como 12 o B como 34 e o restante vazio. Após receber os dados solicitados deve-se fazer a conversão dos valores obtidos em A e B utilizando a Equação (1) para o RPM.

$$RPM = \frac{(A * 256) + B)}{4} \tag{1}$$

$$RPM = \frac{(34 * 256) + 44)}{4} \tag{2}$$

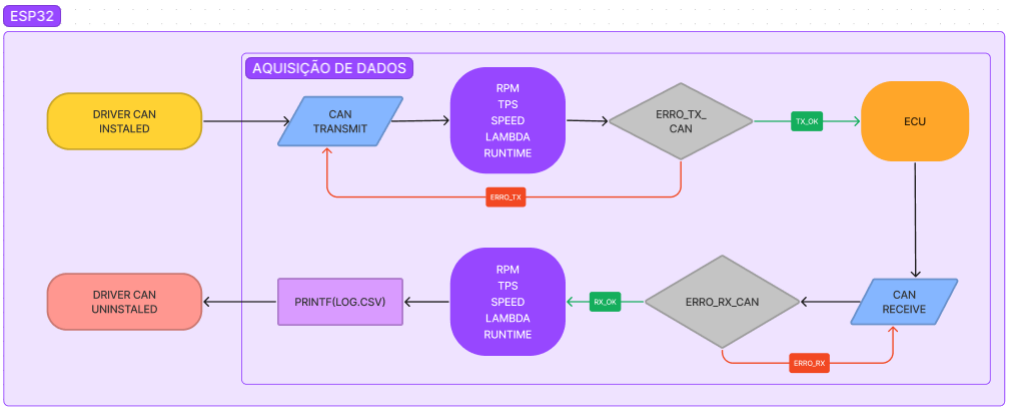
$$RPM = 2.187RPM \tag{3}$$

Assim obtendo o valor (3) correspondente ao RPM do motor no instante da requisição.

Os demais parâmetros também devem ser convertidos por sua respectivas fórmulas a cada interação, para termos os dados convertidos para suas respectivas unidades. Desta forma foi desenvolvido o código do dispositivo de forma a apresentar os dados já convertidos no arquivo de log gerado, sendo necessário somente apresentar os dados visualmente depois do envio das informações.

O fluxo de aquisição de dados, ilustrado na Figura 15 se deu da seguinte forma: o *driver CAN* é instalado, em seguida é enviado uma mensagem CAN solicitando o dado desejado. Foram requisitados os seguintes dados: RPM do motor, a velocidade do veículo em Km/h, a posição do acelerador, *Throttle Position Sensor - TPS*, dado em porcentagem e o tempo do veículo ligado em segundos. Foi verificado se a mensagem foi enviada com sucesso. Se a solicitação foi bem sucedida a resposta da central do veículo é armazenada no arquivo CSV. O ciclo se repetirá até acabar o número de interações solicitadas pelo usuário.

Figura 15 – Fluxo de captura dos dados



Próprio Autor

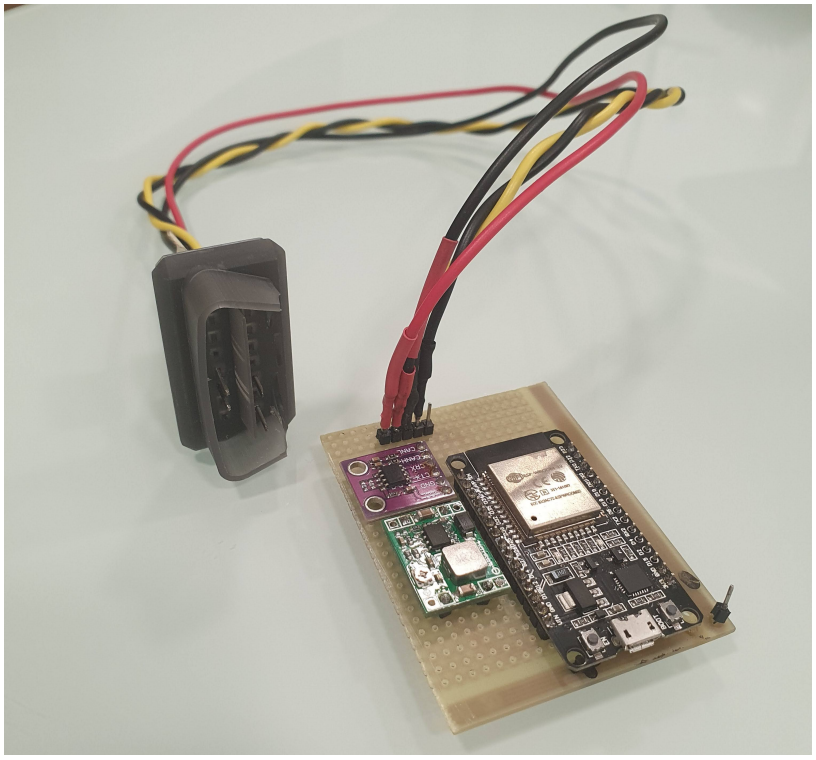
Após as interações serem finalizadas o dispositivo se conecta a internet através de uma rede sem fio disponibilizada pelo usuário e realiza o *upload* do arquivo a um servidor HTTP, o qual os dados puderam ser consultados e analisados posteriormente.

Para a plena conexão e alimentação entre os periféricos foi desenvolvida uma *protoboard* com os módulos, bem como uma fonte de alimentação para alimentar o sistema, demonstrada na Figura 16.

5 Resultados dos Testes Realizados

Com o desenvolvimento do dispositivo protótipo finalizado, foi possível passar para a etapa de aquisição e análise dos dados para efetivar o desenvolvimento do dispositivo de *Datalog* para veículos de competição. Considerando que o foco do trabalho é a captura de dados para análise de desempenho do veículo, é necessário de realizar a captura em ambiente controlado e em uma competição, o que não foi possível de ser feito. Entretanto foi possível validar o pleno funcionamento do dispositivo em vias públicas, respeitando o limite de velocidade da via.

Figura 16 – Protoboard



Próprio Autor

Com o dispositivo conectado a porta OBD2 do veículo, como mostra a Figura 17, foi realizada a primeira captura com o veículo parado, visando validar a aquisição dos valores solicitados. Neste caso foi analisado somente a pressão de acelerador e rotação do motor, a qual pode-se verificar o recebimento correto e realizar ajustes a alguns parâmetros para a melhor apresentação dos dados, os quais estão ilustrados na Figura 18.

Ao ser realizada a captura dos dados foi possível identificar que o parâmetro solicitado correspondente a posição relativa do pedal, a qual possui como PID o valor 0x45, não é suportado pelo veículo, desta forma foi realizada a captura de posição relativa a qual possui o PID 0x11. Neste caso a posição relativa varia de 14% a 85%, correspondendo a 0% e 100% respectivamente.

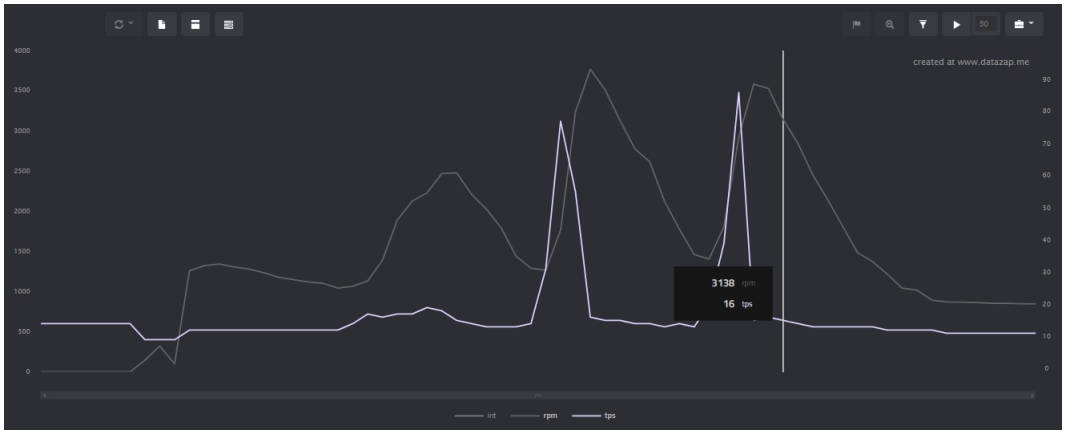
Já com o veículo em movimento uma amostra de aceleração foi capturada, considerando a condição de pista sendo tempo seco, no asfalto comum seco com temperatura ambiente próximo dos 25 °C. O veículo foi acelerado até 90 Km/h, e logo em seguida a velocidade foi reduzida até os 20 Km/h e retomando a aceleração até 80 Km/ e estabilizando posteriormente em 60 Km/h capturando assim o RPM do motor, posição do pedal de acelerador e a velocidade em Km/h. O resultado deste experimento está ilustrado na Figura 19.

É possível perceber a utilização de 100% de pedal durante a aceleração, situação que neste caso foi adequada visto o objetivo de realizar a aceleração no menor espaço de tempo possível dado as condições de pista e do veículo, entretanto tal comportamento pode causar perda de desempenho se tratando de piso molhado e baixa aderência dos pneus por exemplo.

Figura 17 – Protoboard conectada a porta OBD2 do veículo

Próprio Autor

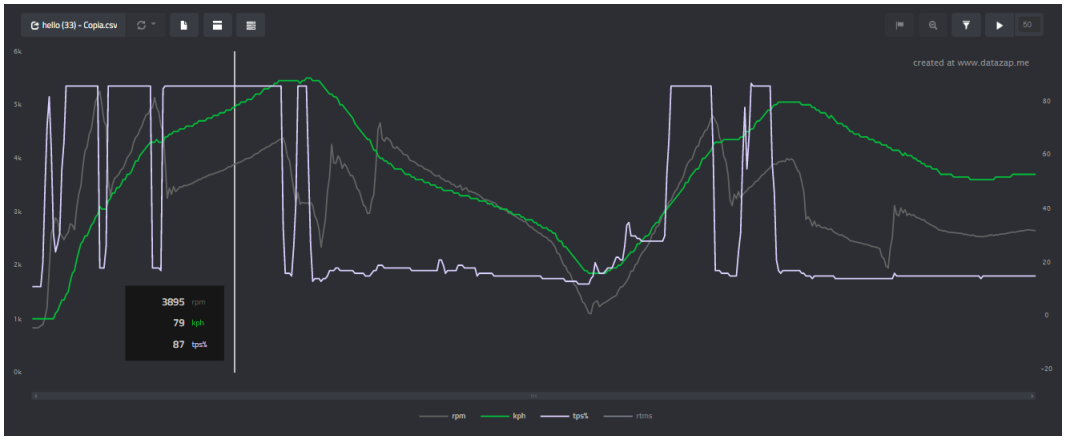
Figura 18 – Log inicial parado



Próprio Autor

Um parâmetro que pode ser usar como comparação entre voltas ou arrancadas recorrentes é a troca de marchas, o qual pode ser relacionado tanto em qual rotação foi realizado, levando-se em consideração o limite de RPM do motor, escalonamento de marchas e pico de torque, quanto o tempo necessário para se realizar a troca, a qual envolve a redução de TPS, o desacoplamento da embreagem, seguido do engate da seleção de

Figura 19 – Log de aceleração, frenagem e retomada.

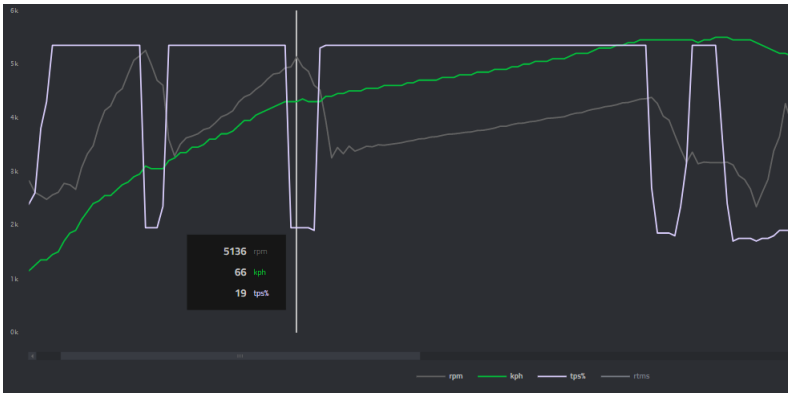


Próprio Autor

marchas, o reacoplamento da embreagem e a retomada da pressão no pedal do acelerador.

Observando o período de aceleração representado na Figura 20 nota-se que o tempo para a realização da troca de marchas foi adequado. Pode-se observar apenas que a faixa de RPM utilizado não foi otimizada, pois foram realizadas as trocas sem passar das 5255 rotações, não explorando idealmente os 6500 RPM de limite, tendo assim margem para a diminuição do tempo de aceleração de 0 a 100 Km/h.

Figura 20 – Faixa de aceleração



Próprio Autor

Analisando o início do log, representado pela Figura 21 pode-se visualizar que houve uma diminuição da pressão do pedal por parte do piloto, para compensar a perda de tração, visto que a rotação do motor subiu de forma diferente do que a velocidade do veículo, podendo ser identificado possivelmente a diminuição de atrito na embreagem e o deslizamento em excesso em relação ao volante do motor ou até mesmo o a perda de tração por apenas uma das rodas do veículo. Neste caso, para sabermos o quanto isso representa em tempo de pista deve-se realizar o log mais vezes buscando replicar as condições de pista e do veículo, afim de validar o tempo de aceleração e velocidade final por exemplo.

Na Figura 22, é possível observar que após a aceleração o piloto reduz a velocidade

Figura 21 – Início do log

Próprio Autor

utilizando somente a embreagem para a redução de marchas, com pouco auxílio do acelerador, sobrecarregando assim a embreagem, porém contando com a inércia do motor para auxiliar na redução de velocidade. No entanto, por se tratar de um veículo sem sobrealimentação não foi levado em consideração a pressão de turbina em um veículo que contenha um turbo compressor, o qual necessita de altas rotações para manter a pressão de trabalho da turbina na faixa ideal, auxiliando assim a retomada de aceleração em uma saída de curvam por contar com mais potência disponível por exemplo.

Figura 22 – Redução de marchas sem a utilização do acelerador

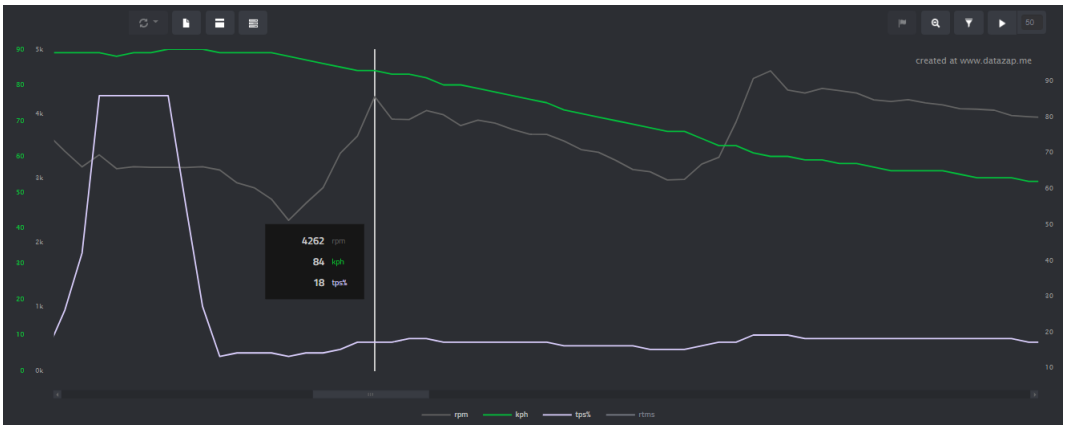


Figura 23 – Próprio Autor

O dispositivo desenvolvido se mostra capaz de apresentar dados factíveis de análise e melhora de desempenho, seja ele em uso de pista ou para controle de frotas por exemplo, dada a precisão satisfatória dos dados, e o bom funcionamento do dispositivo desenvolvido.

As possibilidades de analise são inúmeras, conforme mais dados são captados e analisados para o melhor funcionamento do veículo e comportamento do piloto, mais precisas serão as melhorias de desempenho. Mesmo utilizando todos os parâmetros que a comunicação via OBD2 proporciona, ainda faltariam dados que a montadora não apresenta

e ficam fechados para outros sistemas do veículo ou por não serem realmente captados. Como exemplo o grau de rotação do volante, força G, ou até mesmo no caso de um carro que recebeu alguma modificação externa a rede CAN do veículo.

No caso de um carro sobrealimentado, seria interessante saber o RPM da turbina, pressão de turbo ou até mesmo a temperatura dos gases de escape por cilindro individualmente. Nestes casos poderia ser utilizado as portas disponíveis do microcontrolador como entradas de dados auxiliares, caso o veículo não tivesse suporte aos dados desejados, podendo também atuar em conjunto com algum periférico externo, seja ele um relógio manômetro, luzes de indicação ou até mesmo no caso de haver algum parâmetro fora do adequado, por exemplo, alertas ou até mesmo a tomada de decisão de forma a proteger o motor.

Considerações Finais

Através do desenvolvimento deste dispositivo foi possível entender toda a arquitetura presente na comunicação de sensores e atuadores de um veículo equipado com rede CAN embarcada, para que seja possível a captura dos dados. A fim de realizar a aquisição destes dados, foi desenvolvido um protótipo, necessitando dimensionar todo o *hardware* e realizar a integração eficiente com o *firmware* para somente então efetivar a comunicação e análise das condutas do piloto e o desempenho do veículo.

O dispositivo se mostrou muito eficiente e capaz de captar dados factíveis de serem analisados e condizentes com as informações que foram solicitadas para a ECU. A visualização dos dados foi possível pela adequação ao padrão CSV, o qual é facilmente interpretável por inúmeros *softwares*, tanto o utilizado para a análise, quanto interpretadores de planilhas.

Como possíveis melhorias no projeto relacionadas ao *firmware*, pode-se citar uma interface para controle do dispositivo, visto a possibilidade de controlar o inicio e fim do registro do log, ou até mesmo gerenciar o armazenamento e envio dos arquivo, bem como a captura de outros parâmetros adicionais de comportamento do veículo podendo ser eles: mistura de Ar/Combustível, avanço de ponto de ignição, pressão do coletor de admissão, entre outros. Outra melhoria possível seria ampliar as conexões disponíveis seja ela por meio de *bluetooth*, por exemplo.

Também a adição de uma interface de comunicação através de um aplicativo ou um *display* com botões contendo *feedbacks* luminosos, bem como o desenvolvimento da placa PCB em conjunto com um suporte adequado para fácil conexão e uma embalagem para armazenar a placa, seriam melhorias bem vindas no projeto.

Referências

AMARASINGHE, M. et al. Cloud-based driver monitoring and vehicle diagnostic with OBD2 telematics. In: *15th International Conference on Advances in ICT for Emerging Regions, ICTer 2015 - Conference Proceedings*. [S.l.]: Institute of Electrical and Electronics Engineers Inc., 2016. p. 243–249. ISBN 9781467394413. Citado 3 vezes nas páginas 8, 10 e 13.

BADHIYE, S. et al. Data logger system: A survey. *International Journal of Computer Technology and Electronics Engineering*, p. 2011, 2011. Disponível em: <https://www.researchgate.net/publication/271964052>. Citado na página 4.

CBA, C. B. D. A. *CAMPEONATO BRASILEIRO DE CAMINHÕES - COPA TRUCK REGULAMENTO DESPORTIVO 2021*. 2021. 1-26 p. Disponível em: <https://cba.org.br/upload/downloads//576/copa-truck-regulamento-desportivo-2021-.pdf>. Citado na página 3.

CETESB. *EMISSÕES VEICULARES NO ESTADO DE SÃO PAULO 2017 SÉRIE RELATÓRIOS GOVERNO DO ESTADO DE SÃO PAULO-SECRETARIA DO MEIO AMBIENTE CETESB-COMPANHIA AMBIENTAL DO ESTADO DE SÃO PAULO*. São Paulo, 2017. 1–214 p. Disponível em: <https://cetesb.sp.gov.br/veicular/relatorios-e-publicacoes/>. Citado na página 3.

DIAZ, J. et al. CAN Bus embedded system for lighting network applications. *Midwest Symposium on Circuits and Systems*, p. 531–534, 2008. ISSN 15483746. Citado na página 7.

ELECTRONICS, C. *OBD2 Explained - A Simple Intro (2021) – CSS Electronics*. 2021. Disponível em: <https://www.csselectronics.com/pages/obd2-explained-simple-intro>. Citado 3 vezes nas páginas 3, 8 e 10.

EPA, U. *US EPA | Transportation Conformity: Chronological List of Rulemakings*. 2012. Disponível em: <https://www.epa.gov/state-and-local-transportation/transportation-conformity-chronological-list-rulemakings>. Citado na página 3.

ESPRESSIF. *ESP32 Wi-Fi and Bluetooth Modules I Espressif*. 2021. Disponível em: <https://www.espressif.com/en/products/modules/esp32>. Citado na página 11.

ESPRESSIF. *ESPRESSIF - ESP32-DevKitC V4 Getting Started Guide - ESP32 - — ESP-IDF Programming Guide latest documentation*. 2022. Disponível em: <https://docs.espressif.com/projects/esp-idf/en/latest/esp32/hw-reference/esp32/get-started-devkitc.html>. Citado 2 vezes nas páginas 11 e 12.

FERGUSSON, D. P. *David P. Fergusson - The Science of Motorsport*. [S.l.: s.n.], 2019. 1-290 p. ISBN 9780203732311. Citado 2 vezes nas páginas 2 e 3.

JANSCHEK, K.; BRAUNE, A. Application of industrial CAN bus technology for LEO-satellites. *Acta Astronautica*, v. 46, n. 2, p. 313–317, 2000. ISSN 0094-5765. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0094576599002210>. Citado na página 7.

KING, Z.; YU, S. Investigating and securing communications in the Controller Area Network (CAN). *2017 International Conference on Computing, Networking and Communications, ICNC 2017*, Institute of Electrical and Electronics Engineers Inc., p. 814–818, mar 2017. Citado 2 vezes nas páginas 5 e 6.

LOKMAN, S.-F.; OTHMAN, A. T.; ABU-BAKAR, M.-H. Intrusion detection system for automotive Controller Area Network (CAN) bus system: a review. *EURASIP Journal on Wireless Communications and Networking 2019 2019:1*, SpringerOpen, v. 2019, n. 1, p. 1–17, jul 2019. ISSN 1687-1499. Disponível em: <https://link.springer.com/articles/10.1186/s13638-019-1484-3>. Citado na página 7.

M. Desai, M.; J. Patoliya, J. Design & Implementation of Data Acquisition & Controlling System Based on Free RTOS Platform. *International Journal of Advanced Networking Applications*, International Journal of Advanced Networking and Applications - IJANA, v. 10, n. 02, p. 3816–3821, 2018. Citado na página 9.

Miguel Gomes Ribeiro, H. Dados CAN / OBD2 em tempo real de viaturas auto. 2015. Citado 2 vezes nas páginas 7 e 12.

MMA. *Emissões Veiculares*. 2013. Disponível em: <https://antigo.mma.gov.br/mma-em-numeros/emissoes-veiculares>. Citado na página 3.

NI, N. I. C. *Controller Area Network (CAN) Overview - NI*. 2022. Disponível em: <https://www.ni.com/pt-br/innovations/white-papers/06/controller-area-network--can--overview.html>. Citado na página 5.

OSS, M. Leitura OBD2 através de smartphone. Universidade Tecnológica Federal do Paraná, dec 2018. Disponível em: <http://repositorio.utfpr.edu.br:8080/jspui/handle/1/19812>. Citado 3 vezes nas páginas 7, 9 e 12.

QIN, H.; YAN, M.; JI, H. Application of Controller Area Network (CAN) bus anomaly detection based on time series prediction. *Vehicular Communications*, Elsevier, v. 27, p. 100291, jan 2021. ISSN 2214-2096. Citado na página 5.

SMARTKITS. *Módulo MCP2551 Transceptor CAN*. 2022. Disponível em: <https://www.smartkits.com.br/modulo-transceptor-mcp2551>. Citado na página 12.

SOUZA, D. F. R. de. *Desenvolvimento de um dispositivo para leitura da rede CAN veicular. 2019. Trabalho de Conclusão de Curso (Bacharelado em Engenharia Eletrônica) - Universidade Tecnológica Federal do Paraná, Campo Mourão, 2019.* 2019. Disponível em: <http://repositorio.utfpr.edu.br/jspui/handle/1/24152>. Citado na página 13.

TELEDYNE. *Teledyne LeCroy - Software Options - Serial Trigger and Decode*. 2022. Disponível em: <https://teledynelecroy.com/options/productdetails.aspx?modelid=1847&categoryid=12&groupid=88>. Citado na página 15.

ANEXO A – Projeto completo disponível na plataforma GitHub

O projeto com os códigos utilizados no trabalho se encontram em: <https://github.com/bittsouza/OBD2_DATALOGGER>

ANEXO B – Log apresentado no presente trabalho

O log apresentado no trabalho se encontra em: <https://datazap.me/u/bittsouza/rolling-0-88-0807?log=0&data=0-1-2>