INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DE SANTA CATARINA – CÂMPUS FLORIANÓPOLIS
DEPARTAMENTO ACADÊMICO DE ELETROTÉCNICA
CURSO DE GRADUAÇÃO EM ENGENHARIA ELÉTRICA

MARIA HELENA DEMETRIO DE SOUSA

ESTUDO E ANÁLISE DE DADOS DA REDE CAN DO VEÍCULO ELÉTRICO BYD QIN

FLORIANÓPOLIS, 2025.

INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DE SANTA CATARINA – CÂMPUS FLORIANÓPOLIS
DEPARTAMENTO ACADÊMICO DE ELETROTÉCNICA
CURSO DE GRADUAÇÃO EM ENGENHARIA ELÉTRICA

MARIA HELENA DEMETRIO DE SOUSA

ESTUDO E ANÁLISE DE DADOS DA REDE CAN DO VEÍCULO ELÉTRICO BYD QIN

Trabalho de Conclusão de Curso submetido ao Instituto Federal de Educação, Ciência e Tecnologia de Santa Catarina como parte dos requisitos para obtenção do título de Engenheiro(a) Eletricista.

Orientador:
Prof. Márcio Silveira Ortmann, Dr.

FLORIANÓPOLIS, 2025.

ESTUDO E ANÁLISE DE DADOS DA REDE CAN DO VEÍCULO ELÉTRICO BYD QIN

MARIA HELENA DEMETRIO DE SOUSA

Este trabalho foi julgado adequado para obtenção do título de Engenheiro(a) Eletricista e aprovado na sua forma final pela banca examinadora do Curso de Graduação em Engenharia Elétrica do Instituto Federal de Educação, Ciência e Tecnologia de Santa Catarina.

Florianópolis, 28 de fevereiro, 2025.

Banca Examinadora:

gov.br
Documento assinado digitalmente
MARCIO SILVEIRA ORTMANN
Data: 04/04/2025 08:55:32-0300
Verifique em https://validar.iti.gov.br

_____

Márcio Silveira Ortmann, Dr.

gov.br
Documento assinado digitalmente
ADRIANO DE ANDRADE BRESOLIN
Data: 04/04/2025 20:05:01-0300
Verifique em https://validar.iti.gov.br

_____

Adriano de Andrade Bresolin, Dr.

gov.br
Documento assinado digitalmente
DANIEL GODOY COSTA
Data: 04/04/2025 20:48:59-0300
Verifique em https://validar.iti.gov.br

_____

Daniel Godoy Costa, Mestre.

AGRADECIMENTOS

Gostaria de agradecer e dedicar este trabalho de conclusão de curso a:

A Deus, que me fortaleceu e me manteve de pé durante toda a minha jornada, sendo a fonte de força e inspiração em todas as minhas conquistas.

Aos meus pais, Assis Santos de Sousa e Lucia Elena Demetrio de Sousa, que sempre foram meu alicerce, oferecendo amor, apoio e dedicação em todas as fases da minha trajetória acadêmica e da minha vida em geral.

Ao meu namorado, Eric Cristhiano Marcelino da Silva, pela paciência, compreensão e por cuidar de tudo ao meu redor, permitindo-me dedicar o tempo necessário a este trabalho. Agradeço também por assistir à minha apresentação antes de todos, sempre com carinho e apoio.

Ao meu orientador, Marcio Silveira Ortmann, por ser um guia não só na minha vida acadêmica, mas também um grande incentivador da minha vida profissional. Sua orientação foi essencial para o desenvolvimento deste trabalho e sua amizade foi essencial para minha posição atual no mercado de trabalho.

Aos engenheiros e profissionais com os quais tive o privilégio de trabalhar em conjunto, especialmente ao Gustavo Henrique Fontes.

Aos meus amigos e colegas, dentro e fora da faculdade, que sempre estiveram ao meu lado, oferecendo conforto, descontração, fidelidade e irmandade.

Agradeço ao Laboratório de Mobilidade Elétrica (EMOL) do Instituto Federal de Santa Catarina pelo acesso à bancada de estudos, viabilizado pelos investimentos do projeto ConVerTE, com o apoio da ANEEL, da Celesc, da Unidade Embrapii do IFSC e da FEESC.

Estendo minha gratidão a todos os professores do EMOL, especialmente ao Adriano Bresolin e Daniel Godoy, que sempre acreditaram no potencial do meu trabalho e contribuíram para que fosse realizado.

Aos demais professores, que contribuíram com seus ensinamentos e conhecimentos para minha formação.

A todos vocês, meu muito obrigada!

RESUMO

A rede CAN é um barramento robusto, originalmente projetado para a comunicação interna entre os módulos eletrônicos dos veículos, permitindo a troca de informações operacionais essenciais para seu funcionamento. Com a modernização dos sistemas, surge a possibilidade de extrair esses dados para fins além da comunicação interna, abrindo novas oportunidades de aplicação e estudo. Este trabalho tem como objetivo viabilizar a leitura e interpretação da rede CAN do veículo elétrico BYD Qin, utilizando as bancadas didáticas do Laboratório de Mobilidade Elétrica, com ênfase em suas aplicações educacionais. O estudo abrange a análise da arquitetura da rede CAN, a identificação e o mapeamento dos dados relevantes, bem como a captura e análise dessas informações. A pesquisa faz uso tanto de um sniffer comercial quanto de um protótipo desenvolvido, que atua como nó receptor em dois níveis distintos de operação. Ao final, são apresentados os dados parametrizados e discutidas suas possíveis aplicações didáticas e práticas.

Palavras-chave: Rede CAN, Veículo Elétrico, Indústria automotiva, BYD.

ABSTRACT

The CAN bus is a robust communication system originally designed for internal communication among the electronic modules of vehicles, enabling the exchange of operational data essential for their proper operation. With the modernization of systems, the possibility arises to extract these data for purposes beyond internal communication, opening up new opportunities for application and study. This work aims to enable the reading and interpretation of the CAN network in the BYD Qin electric vehicle, using the educational workbenches available in the Electric Mobility Laboratory, with a focus on its educational applications. The study covers the analysis of the CAN network architecture, the identification and mapping of relevant data, as well as the capture and analysis of this information. The research employs both a commercial sniffer and a custom-developed prototype, which operates as a receiver node at two distinct levels of operation. Finally, the parameterized data are presented, and their potential educational and practical applications are discussed.

Keywords: CAN Network, Electric Vehicle, Automotive Industry, BYD.

LISTA DE FIGURAS

LISTA DE QUADROS

LISTA DE ABREVIATURAS E SIGLAS

ACK	*Acknowledgment* (Confirmação de Leitura)
CAN	*Controller Area Network (*Rede de Controle de Área)
CAN_H	CAN *High* (CAN Alto)
CAN_L	CAN *Low (*CAN Baixo)
CI	Circuito Integrado
CPU	*Central Processing Unit* (Unidade Central de Processamento)
CRC	*Cyclic Redundancy Check (*Verificação de Redundância Cíclica)
DLC	*Data Length Code* (Código de Comprimento de Dados)
ECU	*Electronic Control Unit* (Unidade de Controle Eletrônico)
EMOL	Laboratório de Mobilidade Elétrica
EOF	*End of Frame* (Fim do Quadro)
EV	*Electric Vehicle* (Veículo Elétrico)
FPGA	*Field programmable gate array (*arranjo de porta programável em campo*)*
ID	*Identifier* (Identificador)
IDE	*Identifier Extension* (Extensão do Identificador)
IFSC	Instituto Federal de Santa Catarina
KBPS	*Kilobits per Second* (Quilobits por Segundo)
MCU	*Microcontroller Unit* (Unidade Microcontroladora)
OBD	*On-Board Diagnostics* (Diagnóstico de Bordo)
RPM	*Revolutions Per Minute* (Rotações por Minuto)
RTR	*Remote Transmission Request* (Solicitação de Transmissão Remota*)*
SRR	*Substitute Remote Request* (Substituto da Solicitação Remota)

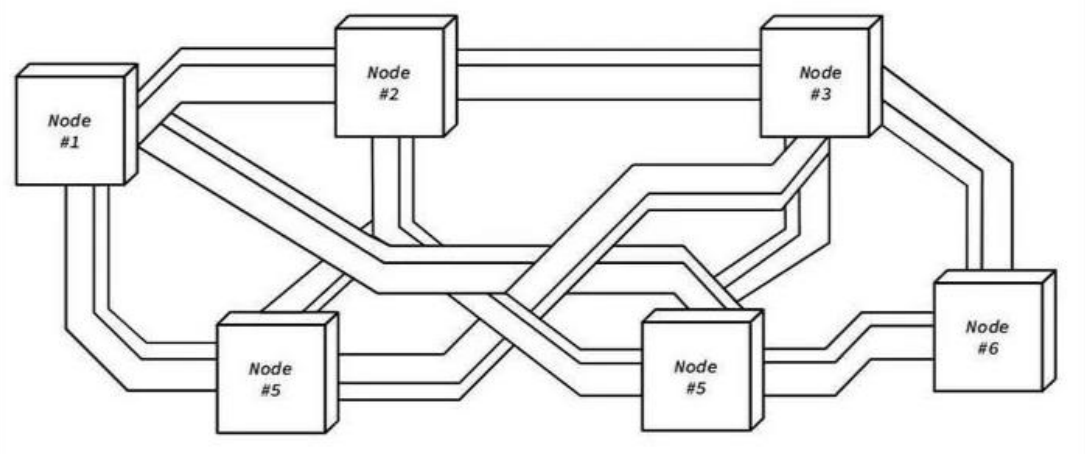
SOC	*State of Charge* (Estado da Carga)
SOF	*Start of Frame* (Início do Quadro)
VHDL	*Very High-Speed Integrated Circuit Hardware Description Language* (Linguagem de Descrição de Hardware para Circuitos Integrados de Alta Velocidade)

SUMÁRIO

1 INTRODUÇÃO

Há pouco mais de três décadas, as conexões entre sistemas eletrônicos em veículos eram realizadas por meio de uma variedade de protocolos de comunicação diferentes e cabos dedicados. Com a crescente modernização dos componentes eletrônicos e ampliação de suas funções, os sistemas se tornavam cada vez mais complexos e labirínticos de fiação, e a confiabilidade das transmissões era colocada em risco. A Figura 1 ilustra como seria as conexões eletrônicas veiculares antigas, antes da rede de comunicação centralizada.

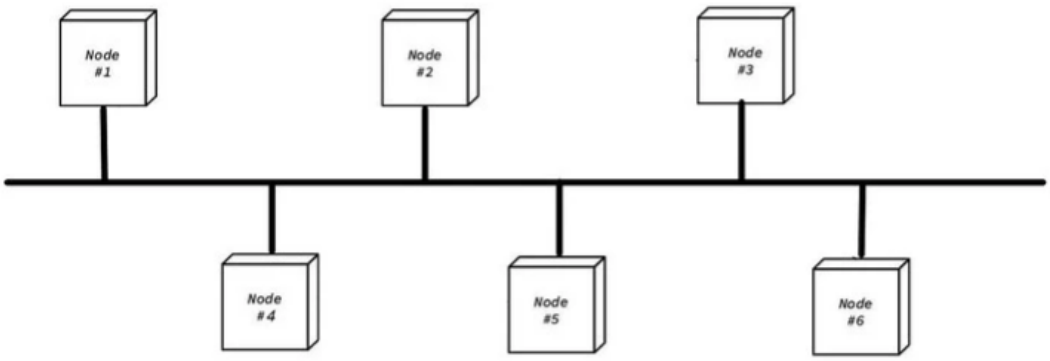
Figura 1 – Conexões eletrônicas sem barramento de comunicação central



Fonte: Weis (2022).

De acordo com Kuhlgatz (2016), esse dilema encontrou sua solução quando uma equipe liderada pelos pesquisadores da Bosch, Siegfried Dais e Uwe Kiencke, concebeu a rede de controle CAN (C*ontroller Area Network*), em 1985. Esse sistema se baseia em uma arquitetura de rede que opera por meio de um barramento de comunicação serial, onde os próprios componentes eletrônicos incorporados, como sensores, unidades de controle e atuadores, atuam como "nós" na rede. Eles podem então trocar informações entre si por meio de uma linguagem padronizada e ordenada. Como contraste à Figura 1, a Figura 2 traz conexões entre nós onde um barramento central existe, ilustrando a rede CAN.

Figura 2 – Conexões eletrônicas com barramento de comunicação central



Fonte: Weis (2022).

Este avanço permitiu a ampliação das capacidades de transmissão de informações e dados entre as unidades de controle eletrônico (ECUs), ao mesmo tempo em que promovia a compatibilidade entre os módulos eletrônicos. Mesmo em seus testes iniciais, a solução já demonstrava uma confiabilidade e qualidade de transmissão excepcional e, como resultado, em 1991 foi instalada na classe S da Mercedes-Benz, marcando o início de sua ampla adoção na indústria de veículos em geral (Kuhlgatz, 2016).

Essa confiabilidade e eficiência impulsionam um desenvolvimento tecnológico contínuo na indústria de veículos, podendo hoje ser considerada uma tecnologia pioneira dos veículos modernos. Atualmente, o número de ECUs em veículos de passeio pode chegar a mais de 100, controlando uma ampla gama de funções que vão desde direção assistida até recursos de luxo, como massagem nos assentos e aquecimento do volante (Mehta; Lippincott e Lu, 2023).

1.1 Justificativa

Embora a rede CAN seja uma solução tecnológica avançada e muito bem difundida na construção dos veículos atuais, que vão desde carros de passageiros até caminhões, ônibus e outros veículos automotores, sua aplicação com aproveitamento máximo ainda está majoritariamente relacionada à comunicação interna entre os próprios sistemas eletrônicos do veículo. Com a capacidade de acessar facilmente informações a partir de um único ponto de conexão e a variedade de dados que podem

ser combinados e processados, surgem inúmeras oportunidades para beneficiar diferentes usuários.

Isso engloba, por exemplo, a gestão de frotas, incluindo análises de motoristas e veículos visando economia, bem como aplicações em oficinas mecânicas para diagnósticos precisos de problemas, reduzindo a necessidade de suposições. Já no automobilismo, *dataloggers* (armazenadores de dados) são essenciais para documentação histórica de desempenho dos componentes eletrônicos e para análises em busca de melhores resultados.

Na gestão de frotas, Moura e Andrade (2022) destacam que certos comportamentos dos motoristas podem acelerar o desgaste dos componentes mecânicos dos veículos e aumentar significativamente os custos com combustível. Exemplos incluem o tempo excessivo com o motor ligado sem uso, mudanças de marchas inadequadas ou excessivas (violações de RPM - Rotações por Minuto, unidade de velocidade ângular), além de freadas e acelerações bruscas. Através da leitura de dados gerados pela telemetria, que registra essas infrações e associa-as à identificação do motorista, é possível utilizar essas informações para otimizar treinamentos de condução econômica. Essa abordagem pode resultar em uma redução de até 25% no consumo de combustível e em uma diminuição de 20% nas despesas com manutenção corretiva, trazendo benefícios significativos para as empresas.

Polese (2017) destaca os benefícios da adoção de dispositivos de telemetria na gestão de frotas, enfatizando a simplificação do monitoramento de veículos e a agilidade na resolução de problemas. Esses dispositivos oferecem acesso instantâneo a informações críticas sobre o estado dos veículos, permitindo uma identificação e diagnóstico de falhas de forma remota, sem a necessidade de deslocamento físico de técnicos ou engenheiros. Essa abordagem reduz significativamente o tempo de inatividade dos veículos, minimiza custos operacionais e proporciona uma solução mais eficiente e responsiva às demandas da frota.

A palestra "Novo Scania Euro 5: Diagnóstico de falhas na rede CAN com Válter" (NOVO..., 2020), apresentada pela Doutor IE, que é uma referência em documentação técnica automotiva, reforça a importância do domínio técnico da rede CAN. Durante a apresentação, foram detalhadas as quatro redes CAN presentes no caminhão Scania Euro 5, incluindo a identificação dos módulos eletrônicos

conectados a cada rede e a implementação de fluxos eficientes para detecção de erros. Essa abordagem evidencia como o profundo entendimento das características específicas da rede CAN pode ser aproveitado para aplicações práticas, como diagnósticos avançados.

Voltado à aplicação que envolvem competições automobilísticas, Santos (2014) afirma que um piloto de corrida, mesmo quando em corridas de teste, não seria capaz de prestar atenção na leitura de todos os medidores necessários, na direção e ainda ao final, lembrar de informar tudo aos engenheiros da equipe. Nesse sentido, a aquisição de dados é crucial para a análise do desempenho do veículo e do motorista, para manter registros de manutenção, para determinar com precisão os parâmetros para construção de simuladores reais, entre outros motivos. No catálogo Bosch para automobilismo, por exemplo, há opções de *dataloggers* justamente para suprir essa necessidade, nas quais possuem comunicação CAN, além de outras conexões e possibilidade de telemetria, com destaque ao produto *Data Logger and Sensor Interface C 80* (Bosch, 2023).

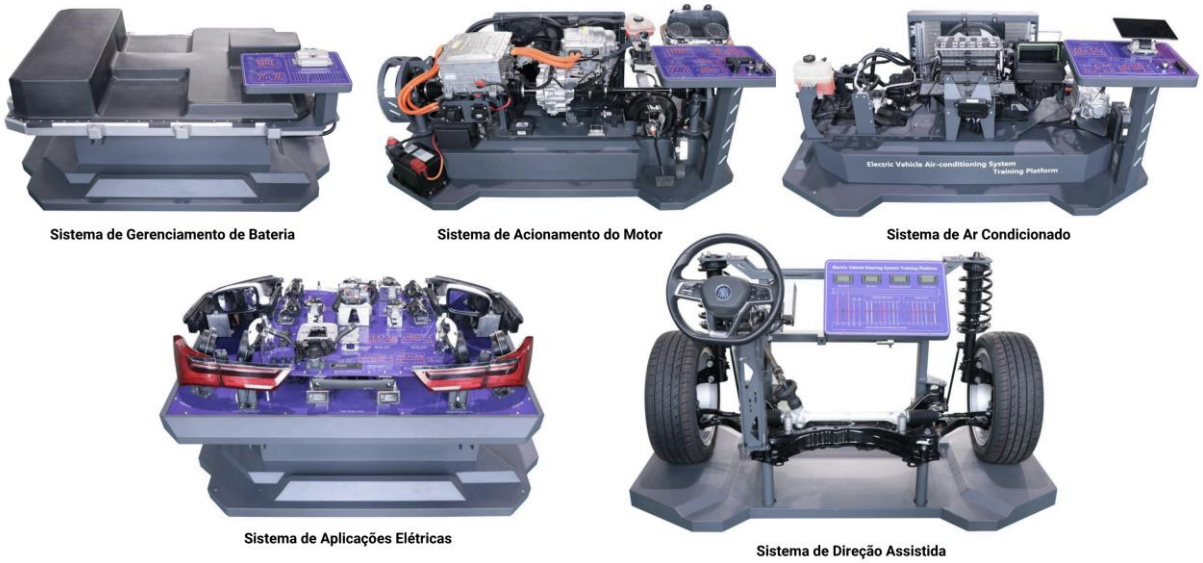
Souza (2022) propôs o desenvolvimento de um leitor OBD com *datalogger* para carros preparados para competições. O dispositivo envia mensagens CAN para coletar informações como RPM, velocidade do veículo, posição do acelerador em porcentagem e tempo de funcionamento do motor em segundos. O estudo demonstrou que o *datalogger* pode traçar estratégias para melhorar o desempenho tanto do veículo quanto do motorista. Durante uma arrancada, por exemplo, observou-se que o tempo de aceleração de 0 a 100 km/h pode ser reduzido com ajustes nas trocas de marcha e maior eficiência na utilização da faixa de RPM. Como proposta de aprimoramento, foram sugeridas futuras capturas de parâmetros adicionais na rede CAN, como mistura ar/combustível, avanço do ponto de ignição e pressão do coletor de admissão, ampliando a capacidade de análise e otimização (Souza, 2022).

Em síntese, a rede CAN possui capacidade de oferecer uma ampla gama de aplicações que ultrapassam a comunicação interna entre os módulos eletrônicos do veículo, promovendo benefícios significativos em diversas outras áreas, com algumas já em andamento e outras ainda a emergir. Assim, as perspectivas para o desenvolvimento de tecnologia baseada nessa rede de comunicação são promissoras.

1.2 Definição do Problema

A *Beifang Teaching Aids R&D Center* é uma empresa dedicada à pesquisa e desenvolvimento de materiais de ensino no setor automotivo, com quase três décadas de experiência. Sua principal atuação é fornecer plataformas de estudo para universidades na China. Um dos sistemas de estudo desenvolvidos é o Sistema de Treinamento com Ligação de Controles Separados para o Veículo Elétrico BYD Qin, um modelo de veículo elétrico da fabricante BYD. Este sistema é composto por cinco bancadas totalmente funcionais e tecnológicas, que abrangem os seguintes sistemas: gerenciamento de bateria, acionamento do motor, ar-condicionado, aplicações elétricas e sistema de direção assistida (Beifang, 2023). Na Figura 3, é possível visualizar uma imagem das bancadas mencionadas.

Figura 3 – Bancada de estudos



Fonte: adaptado de Beifang (2023).

O acesso a esse sistema de treinamento é possível por meio do Laboratório de Mobilidade Elétrica (EMOL) do Instituto Federal de Santa Catarina (IFSC). O projeto faz parte das iniciativas de Pesquisa, Desenvolvimento e Inovação do ConVerTE, com o apoio de importantes entidades, como a ANEEL, a Centrais Elétricas de Santa Catarina (Celesc), a Unidade Embrapii do IFSC e a Fundação de Ensino e Engenharia de Santa Catarina (FEESC).

No contexto deste estudo, o Sistema de Treinamento com Ligação de Controles Separados para o Veículo Elétrico BYD Qin oferece uma valiosa oportunidade para a análise detalhada da rede CAN. As bancadas, que representam os principais controles e sistemas do veículo, são organizadas de forma independente, com bornes de fácil acesso para diferentes sinais elétricos. Embora sejam fisicamente separadas, todas as bancadas estão interconectadas através do barramento CAN, funcionando de maneira integrada e simulando o comportamento sistêmico de um veículo completo.

A rede CAN, por fornecer informações precisas e em tempo real, permite não apenas a detecção e medição dos dados, mas também o seu uso em diversas aplicações. As informações coletadas podem ser aproveitadas para diagnósticos, otimização dos sistemas, desenvolvimento de novas funcionalidades e integração com outras plataformas. Isso amplia significativamente as possibilidades de utilização dos dados do veículo elétrico BYD Qin, promovendo avanços em áreas como eficiência, monitoramento e controle do veículo.

1.3 Objetivo Geral

Viabilizar a leitura e a interpretação da rede CAN do veículo elétrico BYD Qin a partir das bancadas didáticas fornecidas no Laboratório de Mobilidade Elétrica para permitir o uso das informações do barramento em aplicações de ensino.

1.4 Objetivos Específicos

Para que seja possível alcançar o objetivo geral, os objetivos específicos definidos foram os seguintes:

a) estudar a estrutura da rede CAN, a fim de possibilitar a integração como nó na rede e viabilizar a captura das informações relevantes;

b) realizar o mapeamento da rede CAN do veículo BYD, identificando dados de interesse para o presente trabalho;

c) desenvolver um protótipo capaz de se integrar à rede e realizar a leitura dos dados desejados;

d) Apresentar e propor possíveis aplicações para os dados obtidos, evidenciando a relevância e a utilidade das informações extraídas.

2 ARQUITETURA DA REDE CAN

Este capítulo fornece uma base teórica sobre a rede CAN, abordando suas características principais, normas essenciais e estrutura de mensagens. Compreender esses aspectos é crucial para entender o funcionamento do barramento e sua relevância em sistemas embarcados, possibilitando a extração e análise dos dados transmitidos.

2.1 Características Principais

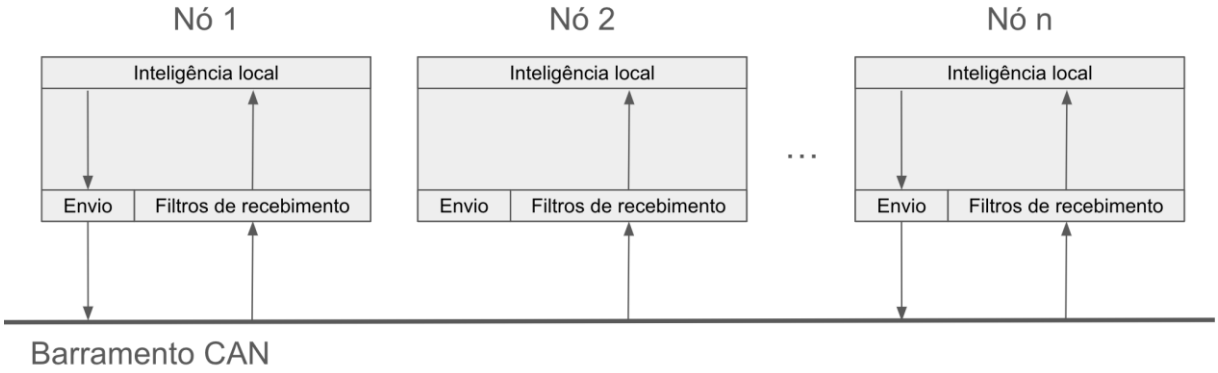
As características intrínsecas da rede CAN são devidamente discorridas por Bosch (1991), sendo elas:

a) priorização de mensagens: permite definir diferentes tempos, prioridades e frequências de envio no barramento;
b) garantia de tempo de latência: assegura a entrega das mensagens dentro de um prazo predefinido;
c) flexibilidade do sistema: a divisão em camadas torna o sistema modular, facilitando manutenção, atualização e expansão;
d) recepção *multicast*: múltiplos nós podem receber e processar simultaneamente a mesma mensagem;
e) capacidade *multimaster*: qualquer nó pode iniciar comunicações, sem um mestre centralizado;
f) consistência de dados: todos os nós compartilham as mesmas informações em tempo real;
g) detecção e sinalização de erros: identifica falhas na transmissão e as sinaliza para correção;
h) retransmissão automática: mensagens corrompidas são reenviadas assim que o barramento fica livre;
i) distinção entre erros temporários e falhas permanentes: melhora o diagnóstico do sistema;
j) desligamento autônomo de nós defeituosos: evita interferências de dispositivos comprometidos.

A recepção *multicast*, a consistência de dados e a capacidade *multimaster* juntas estruturam um modelo de transmissão que garante que todas as mensagens sejam recebidas por todos os nós da rede e processadas conforme sua relevância para cada dispositivo, além de permitir que qualquer nó envie mensagens, tornando a comunicação descentralizada e dinâmica.

Na Figura 4, há um exemplo de operação no qual é possível visualizar um barramento compartilhado com múltiplos nós, onde todos atuam simultaneamente como receptores, e alguns atuam também como transmissores. O diagrama ilustra filtros de recebimento, para demonstrar que cada nó processa apenas as informações relevantes para aquele nó. A inteligência local do nó processa as informações recebidas, aciona os atuadores conforme necessário e, se atua também como transmissor, gera novas mensagens para envio à rede.

Figura 4 – Ilustração básica de receptores e transmissores na rede CAN



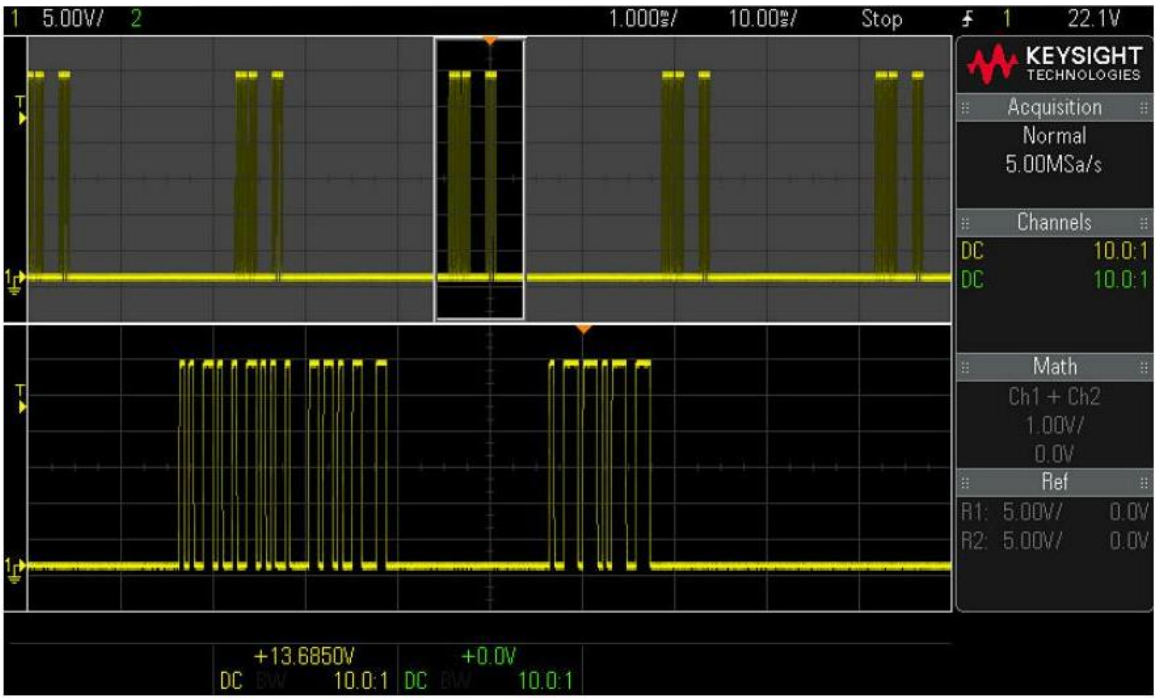
Fonte: Autoria própria.

A priorização de mensagens é fundamental para garantir a consistência do barramento. Como o barramento é compartilhado, a transmissão ocorre de forma ordenada, impedindo que mensagens sejam sobrescritas. Todas as mensagens são organizadas em quadros (*frames*), que contêm um identificador (ID), um campo de dados e bits de controle. O foco inicial recai sobre o ID, sendo as demais características detalhadas ao longo do trabalho.

Embora múltiplas mensagens possam ser enviadas quase simultaneamente, o protocolo evita colisões por meio de um mecanismo de arbitragem

baseado no ID. Durante a arbitragem, a mensagem com menor valor binário tem prioridade e é transmitida primeiro, permitindo um gerenciamento eficiente das informações (Morais, 2012). A organização temporal dessas mensagens, caso analisadas via osciloscópio, seria semelhante à Figura 5.

Figura 5 – Leitura CAN por meio de osciloscópio



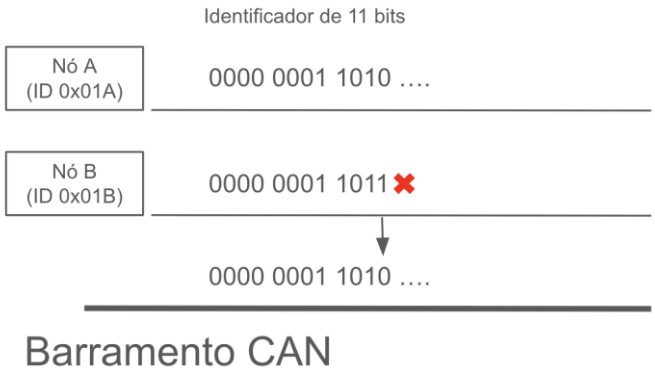
Fonte: KEYSIGHT TECHNOLOGIES (2023).

No barramento físico, os sinais elétricos são representados por níveis de tensão diferencial. No entanto, a interpretação desses sinais ocorre de forma lógica, sendo classificada em dois estados: dominante e recessivo, que correspondem, respectivamente, a nível baixo e alto de tensão diferencial. Mais detalhes sobre essa operação física serão abordados na Subseção 2.2.1 – Camada Física. De forma resumida, o nível dominante está associado ao bit '0' e ocorre quando a tensão diferencial é significativa. Já o nível recessivo representa o bit '1' e ocorre quando essa diferença é mínima.

Durante a transmissão, os nós enviam o ID da mensagem bit a bit e monitoram o barramento, processo chamado *Bit Monitoring*. Caso um nó transmita um bit recessivo e detecte um bit dominante, significa que há outro nó com um ID menor

e, portanto, maior prioridade. Quando isso ocorre, o nó que perdeu a arbitragem interrompe imediatamente sua transmissão, permitindo que apenas o nó com menor ID continue transmitindo sem colisões. Vale destacar que o ID de cada mensagem é único. Esse processo é ilustrado na Figura 6. Além disso, o protocolo garante sua retransmissão dentro do período de latência permitido, assegurando que nenhuma informação seja perdida (Bosch, 1991).

Figura 6 – Processo de arbitragem



Fonte: Autoria própria (2025).

A detecção e sinalização de erros, a distinção entre falhas temporárias e permanentes e o desligamento autônomo de nós defeituosos estruturam um modelo de comunicação que minimiza os impactos na rede. Em um nível mais detalhado, esses mecanismos envolvem técnicas aplicadas tanto na transmissão de bits quanto no nível de quadros. Essas informações serão detalhadas na Seção 2.3 – Construção da Mensagem.

Portanto, as principais características do barramento CAN demonstram sua eficiência na comunicação entre módulos eletrônicos, garantindo confiabilidade, robustez e integridade dos dados.

2.2 Normas essenciais

A topologia básica do barramento CAN é definida pela aplicação de diversas normas que estruturam seu funcionamento em camadas. Elas podem ser definidas em três camadas principais: a camada física, a camada de enlace de dados e a camada de aplicação, cada uma com requisitos e características específicos a serem detalhados no subcapítulo atual.

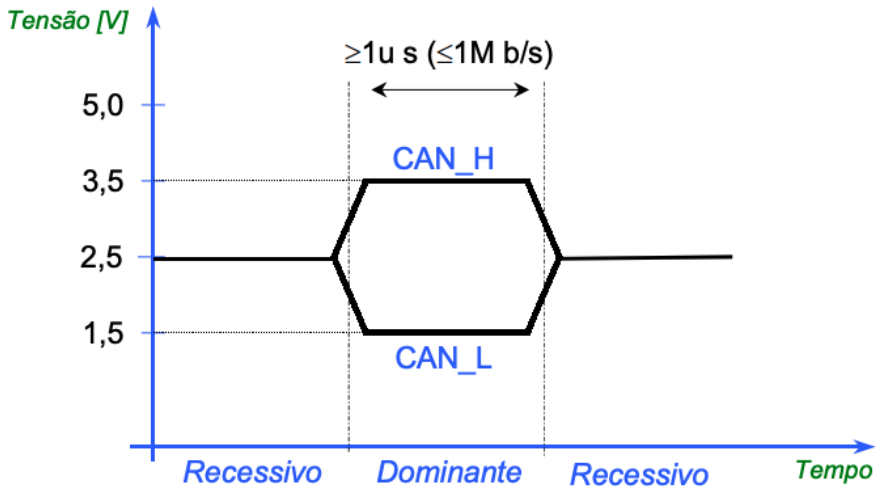
As duas primeiras camadas são padronizadas pelas normas ISO 11898, que definem os modelos de protocolo para redes de alta velocidade, e pela ISO 11519, que estabelece os parâmetros para redes de baixa velocidade. Enquanto as redes de alta velocidade operam com taxas entre 125 kbps e 1 Mbps, as de baixa velocidade funcionam entre 10 kbps e 125 kbps. Embora as normas não especifiquem a camada de alto nível, alguns padrões existem para aplicações específicas (Sousa, Inamasu e Torre Neto, 2001).

Dessa forma, essa estrutura em camadas representa a topologia básica do barramento CAN, e os próximos subtópicos abordarão cada camada separadamente, com ênfase na CAN de alta velocidade, que é o foco principal desta pesquisa.

2.2.1 Camada Física

O barramento atua com informações efetivas em níveis lógicos. Como são duas linhas no barramento, denominadas CAN_H *(*CAN *High)* e CAN_L *(*CAN *Low)*, a transmissão do sinal físico na rede CAN é diferencial e as tensões específicas nas linhas variam entre a CAN de alta velocidade e a de baixa velocidade. No caso da CAN de alta velocidade, os níveis lógicos são definidos da seguinte forma: o nível recessivo (1) ocorre quando as duas linhas apresentam a mesma tensão, aproximadamente 2,5V. Já o nível dominante (0) acontece quando há uma diferença de potencial de 2V entre elas, com CAN_H atingindo cerca de 3,5V e CAN_L cerca de 1,5V (Morais, 2012; Sousa, 2002). A explicação pode ser visualizada na Figura 7.

Figura 7 – Associação entre os níveis de tensão e os estados lógicos



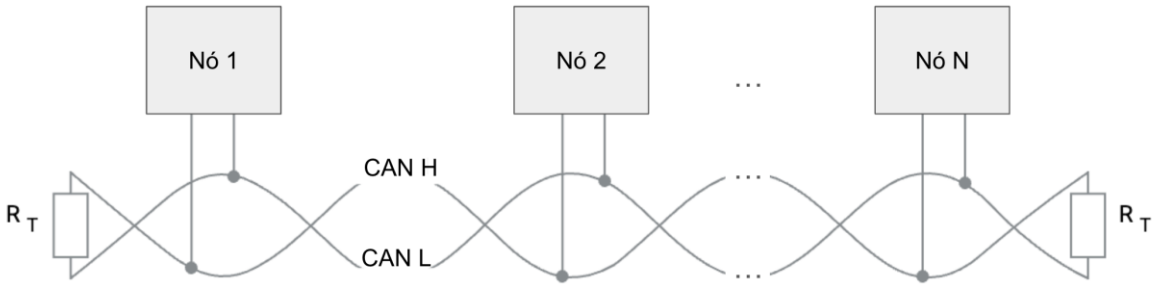
Fonte: (Sousa, 2002).

A integridade da comunicação é reforçada pelo uso de cabos trançados. Essa configuração reduz os efeitos de interferência eletromagnética, pois qualquer perturbação nos fios resultará em uma flutuação conjunta, sem alterar a diferença de potencial entre eles (Guimarães, 2007).

A norma ISO-11898 não especifica tipos específicos de fios e conectores a serem utilizados, deixando essa decisão a cargo do implementador. No entanto, é obrigatória a instalação de resistores terminadores de 120 Ω (nominais) em ambas as extremidades do barramento para garantir a integridade elétrica da rede (Richards, 2002). Sem a terminação adequada, o barramento pode se comportar como uma linha aberta, tornando-se suscetível à captação e irradiação de sinais eletromagnéticos, semelhante a uma antena (Souza, 2013). O resistor de terminação, portanto, desempenha um papel fundamental ao dissipar a energia residual do sinal, estabilizar a transmissão e evitar a propagação de interferências pelo sistema.

Na Figura 8 podem ser visualizados uma representação das linhas trançadas, junto ao resistor de terminação.

Figura 8 – Barramento com aspectos da camada física



Fonte: Autoria própria (2025).

Outra característica associada à camada física é a velocidade de transmissão. Conforme indicado por Richards (2002), a ISO11898 estabelece que um transceptor deve ser capaz de operar em um barramento de 40 metros a 1 Mb/s. Para permitir a operação em um comprimento de barramento maior, é necessário reduzir a taxa de dados. A maioria dos veículos operam nas faixas de 250 e 500 kbps devido ao comprimento de seu barramento. No Quadro 1 é possível visualizar a relação apresentada.

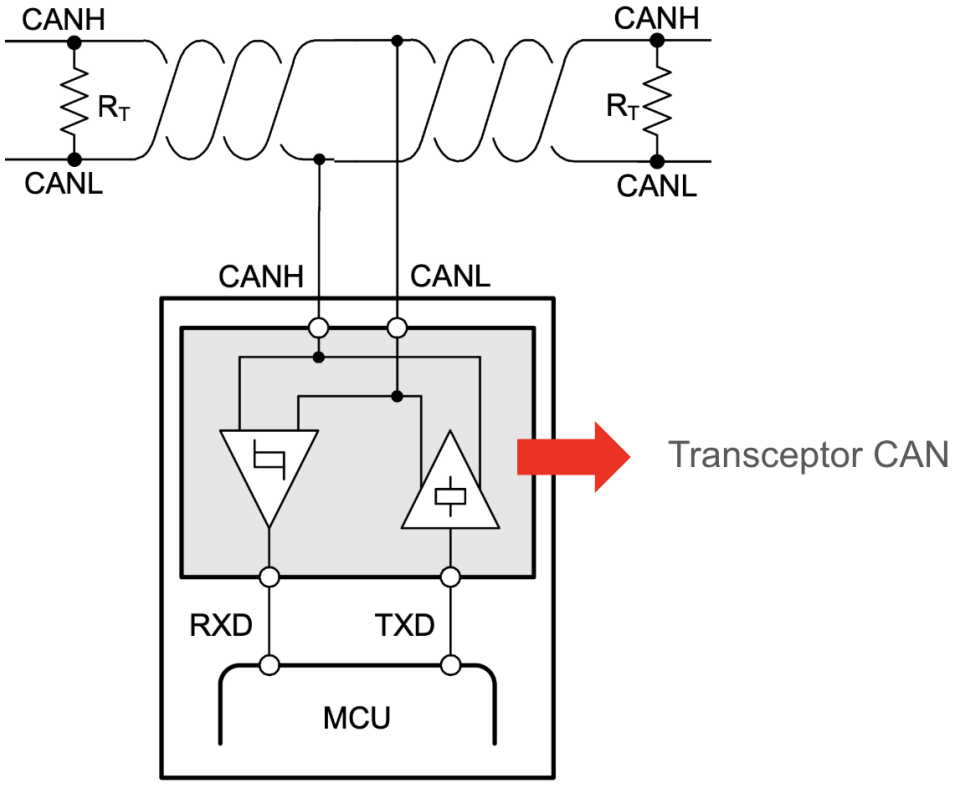
Quadro 1 – Taxa de transmissão x Comprimento de barramento

Taxa de transmissão (kbit/s)	Comprimento do barramento (m)
1000	40
500	130
250	270
125	530
100	620
50	1300
20	3300
10	6700
5	10000

Fonte: Adaptado de Polese (2017).

Anteriormente, a conexão entre o barramento CAN e o controlador CAN era feita por circuitos discretos, compostos por transistores, resistores, diodos e capacitores, responsáveis por converter os níveis de tensão do barramento em pulsos lógicos e vice-versa. Atualmente, essa função é desempenhada por CIs de transceptores CAN dedicados, que oferecem uma solução mais eficiente, reduzindo o consumo de energia, aumentando a imunidade a ruídos e simplificando o design do sistema (VECTOR INFORMATIK GMBH, 2021). Uma conexão do módulo, ilustrando um transceptor CAN é ilustrado na Figura 9.

Figura 9 – Transceptor CAN entre os sinais analógicos e digitais



Fonte: Adaptado de Atkinson (2021).

Esses transceptores apresentam baixas emissões eletromagnéticas e alta resistência a ruídos, além de oferecerem isolação de até 8 kV, que serve de proteção contra descargas eletrostáticas (ESD). O número máximo de dispositivos conectados

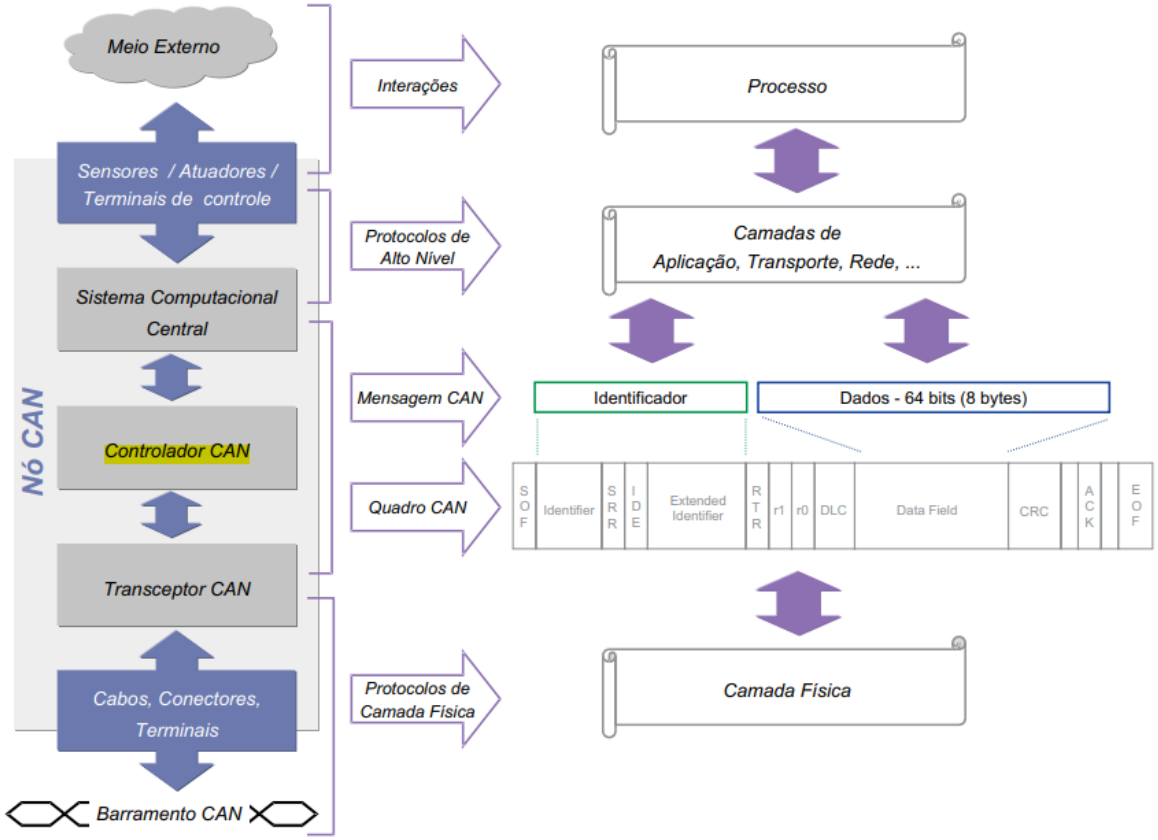
a uma rede CAN pode variar conforme o desempenho dos transceptores utilizados e a velocidade da rede. Ao utilizar o transceptor TJA1050, por exemplo, é possível conectar até 110 nós, conforme especificação (VECTOR INFORMATIK GMBH, 2021).

Estes componentes também diferem quanto à tensão de operação, podendo ser de 3,3V ou 5V, dependendo da aplicação. Enquanto os transceptores de 5V são normalmente utilizados em sistemas automotivos e industriais devido à maior imunidade a ruídos e robustez em redes extensas, os modelos de 3,3V oferecem vantagens em termos de eficiência energética e integração com microcontroladores modernos, eliminando a necessidade de conversores de nível lógico. Além disso, transceptores de 3,3V podem ser totalmente compatíveis com redes CAN de 5V, mantendo conformidade com requisitos de compatibilidade eletromagnética (EMC), garantindo baixos níveis de interferência e operação confiável em ambientes ruidosos (ATKINSON, 2021).

2.2.2 Camada de Enlace de Dados

O transceptor CAN conecta o controlador CAN ao meio de transmissão física, este, por sua vez, atua em toda a implementação da camada enlace de dados (Richards, 2002). É possível analisar a posição dele na organização das camadas, apresentada a seguir na Figura 10.

Figura 10 – Representação das camadas, em destaque o controlador CAN



Fonte: Sousa; Inamasu e Torre Neto (2001).

Conforme Sousa, Inamasu e Torre Neto (2001), a camada de enlace de dados incorpora diversas funcionalidades essenciais, sendo o controlador CAN o responsável pelo gerenciamento da camada. Dentre as funcionalidades, ocorre o encapsulamento de dados, que envolve a organização de informações em pacotes identificáveis, a codificação de quadro, que facilita a estruturação dos bits para uma transmissão eficiente, a detecção de erros, que assegura a integridade dos dados, a sinalização de erros, a confirmação de mensagem e a notificação de falhas.

A comunicação CAN pode ser gerenciada por diferentes abordagens, como o uso de controladores dedicados, a integração de módulos CAN internos em microcontroladores ou até mesmo utilizando Matrizes de Portas Programáveis em Campo (FPGAs) e linguagens de descrição de *hardware*, como VHDL ou Verilog, para criar soluções altamente customizadas. No caso de controladores dedicados,

exemplos incluem o MCP2515 da Microchip e o SJA1000 da NXP, enquanto microcontroladores como STM32, ESP32 e PIC já incorporam módulos CAN internos.

Morais (2012) desenvolveu um controlador CAN em VHDL com o objetivo de operar de forma totalmente independente, eliminando a necessidade de um microcontrolador. Essa implementação permitiu ao hardware enviar, receber e processar mensagens autonomamente, além de identificar falhas na comunicação e se desconectar do barramento em caso de erro. Dessa forma, o controlador criado atendeu aos objetivos do projeto e integrou-se eficientemente à rede CAN.

Um controlador dedicado não depende de um microcontrolador para gerenciar a comunicação CAN, assim como o *hardware* em VHDL proposto por Morais. A principal diferença entre as duas abordagens está no controle total sobre a implementação alcançado por Morais, já que ao projetar um controlador em VHDL, ele tem a liberdade de personalizar todos os aspectos do funcionamento do controlador, ao passo que, ao utilizar um controlador pronto, há restrições impostas pelas funcionalidades e configurações definidas pelo fabricante. Embora essas limitações possam ser adequadas para a maioria das aplicações padrão, elas podem ser limitantes quando é necessário implementar funcionalidades específicas ou otimizar o controlador para um caso de uso particular.

A ausência de intervenção direta do processador principal no gerenciamento das mensagens CAN pode aumentar a eficiência em sistemas onde o microcontrolador já está executando múltiplas tarefas. Em contrapartida, quando o controlador CAN está integrado ao microcontrolador, a comunicação compartilha os recursos do dispositivo, o que pode exigir um planejamento cuidadoso para garantir desempenho adequado em aplicações mais exigentes ou altamente customizadas.

Pires (2005) faz uma comparação entre controladores dedicados e microcontroladores com controlador CAN integrado no decorrer de seu projeto. Ele aponta que, quando se busca flexibilidade e escalabilidade, os controladores dedicados são a melhor escolha, pois eles permitem integrar módulos adicionais e serem usados em diferentes contextos, com mais liberdade de personalização. Por exemplo, é possível combinar o controlador dedicado com diferentes microcontroladores e até adicionar outros circuitos ao redor dele, criando um sistema mais adaptável a várias aplicações. No entanto, os microcontroladores com

controlador CAN integrado são mais simples e baratos, já que combinam o processador e o controlador CAN em um único *chip*. Isso facilita o *design*, diminui o custo e reduz a complexidade, sendo uma solução excelente para sistemas menores ou menos exigentes, onde a modularidade mais relevante pode ser alcançada por meio de *firmware*.

Um exemplo é o ESP32, que possui um periférico TWAI (T*wo-Wire Automotive Interface*), implementando um controlador CAN 2.0A/B compatível com o SJA1000 em um modo mais básico de operação. Isto é, ele não suporta CAN FD — uma versão mais recente do protocolo CAN que permite velocidades superiores a 1 Mbps — e não possui *buffers* avançados, o que limita sua capacidade de lidar com tráfego intenso em comparação com controladores dedicados. Em contrapartida, ele possui módulos com Bluetooth, Wi-Fi, Processador dual-core de 32 bits, que podem facilitar a integração dos dados com a telemetria (Espressif, 2023].

As definições de controlador CAN e microcontrolador dependem, portanto, dos requisitos do projeto a ser desenvolvido. Em sistemas de telemetria veicular ou gestão de condução, por exemplo, um circuito com controlador CAN integrado, mesmo sem *buffers* avançados se torna satisfatório, pois a informação mais relevante, nesses contextos, não costuma ser a captura instantânea de cada dado, mas sim a média ou a tendência ao longo do tempo. Os parâmetros são enviados periodicamente na rede, o que faz com que não tenha apenas uma única oportunidade de leitura de variação do sistema.

2.2.3 Camada de Alto Nível

Ao empregar os componentes necessários para atuar nas camadas física e de enlace de dados, o desenvolvedor de *software* é capacitado a, por meio do *firmware*, configurar o sistema e realizar operações como leitura, escrita e processamento de mensagens na rede CAN, através da camada de aplicação, proporcionando uma abstração mais elevada do funcionamento das camadas inferiores (Sousa; Inamasu e Torre Neto, 2001). Na implementação dessa camada, no entanto, a ausência de padronização abre espaço para uma diversidade significativa de abordagens.

Na construção da arquitetura CAN, a responsabilidade pela padronização recai sobre o desenvolvedor. Em outras palavras, ao garantir que o *hardware* atenda a todos os pré-requisitos operacionais previamente delineados, o desenvolvedor, por meio do *firmware*, adquire a capacidade de configurar a comunicação CAN. Isso inclui, mas não se limita a ajustes como a velocidade de transmissão, a escolha dos identificadores para cada nó, estratégias de detecção e tratamento de erros, além da definição de filtros e máscaras, entre outros parâmetros essenciais.

Guimarães (2002) explora a importância do uso de um dicionário de dados para a criação de uma arquitetura CAN. Isto é, a importância de que cada barramento tenha um dicionário implementado de forma a padronizá-lo. Então as escolhas do desenvolvedor deverão ser compartilhadas entre todos os componentes eletrônicos que integrarão essa rede, de modo a garantir total compatibilidade entre os participantes do barramento.

Para os desenvolvedores que buscam integrar seus projetos a uma rede já existente em um veículo específico, é essencial adotar um processo de deciframento de mensagens, analisando cada bit detalhadamente para compreender o dicionário de dados estabelecido e as configurações específicas realizadas nessa rede. Garantir que o dispositivo esteja em conformidade com os padrões da rede é crucial, tornando imprescindível realizar um estudo prévio abrangente para assegurar uma integração bem-sucedida.

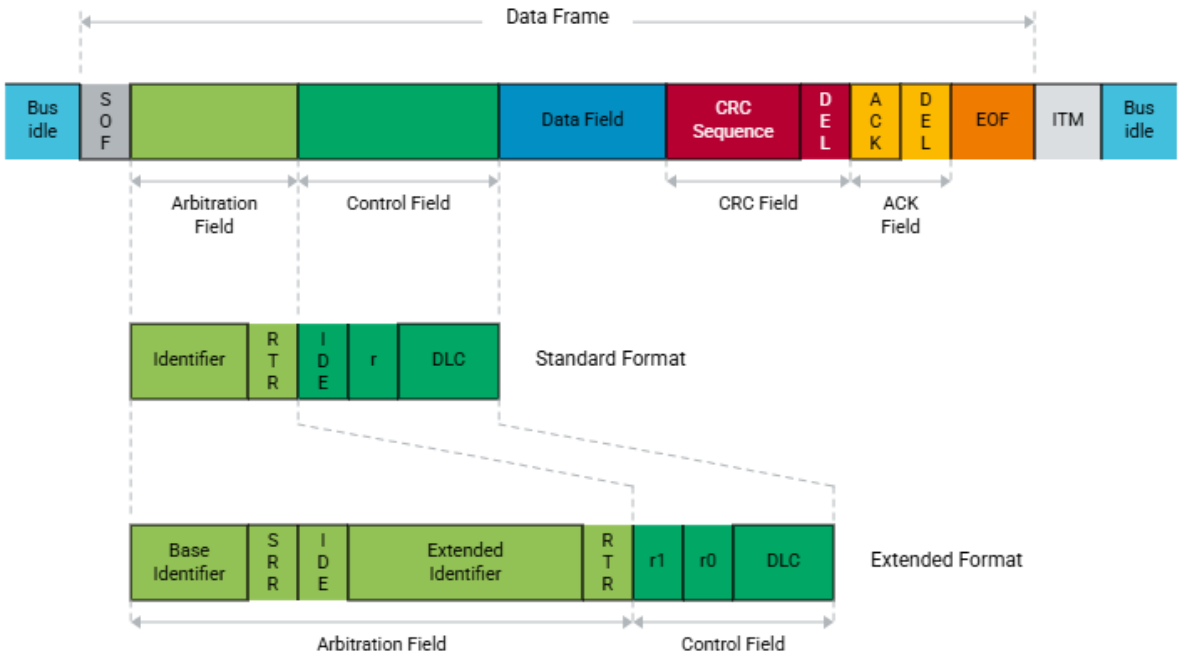
À medida que o sistema cresce e mais dispositivos são integrados, protocolos como CANopen ou J1939, que já são protocolos de alto nível conhecidos, podem ser necessários para garantir interoperabilidade, gerenciamento de dados e facilidade de configuração. Adotar um padrão existente envolve basicamente seguir as especificações e diretrizes fornecidas pelos protocolos para organizar a comunicação entre os dispositivos (VECTOR INFORMATIK GMMBH, 2014).

Na prática, um nó na rede CAN é qualquer dispositivo capaz de transmitir e/ou receber mensagens no barramento. Embora a maioria dos nós seja composta por um MCU, existem casos em que um nó pode ser formado apenas por circuitos lógicos, sensores, atuadores ou transceptores autônomos. Um nó que possua uma lógica fixa de processamento, como lógica combinacional, pode responder a eventos simples sem a necessidade de um processador dedicado, reduzindo a complexidade e o número de camadas envolvidas na comunicação.

2.3 Construção da Mensagem

O campo de identificador no protocolo CAN pode ter 11 bits (formato padrão) ou 29 bits (formato estendido). O frame CAN, tanto na versão padrão quanto na estendida, é mostrado na Figura 11. A versão padrão é chamada CAN 2.0A, enquanto a versão estendida é CAN 2.0B. Identificar corretamente a versão do protocolo no barramento de interesse é essencial para garantir a leitura adequada dos dados. Como o identificador de 11 bits do CAN padrão oferece um espaço limitado de 2.048 identificadores possíveis, ele pode ser insuficiente em redes mais complexas, onde o identificador estendido, com 29 bits, permite uma maior quantidade de identificadores exclusivos, aumentando a flexibilidade do sistema.

Figura 11 – Formato do quadro de dados da CAN padrão e CAN estendida



Fonte: VECTOR INFORMATIK GmbH (2023).

O SOF (*Start of Frame*), transmitido como um bit dominante, marca o início de uma mensagem, criando uma borda de transição do nível recessivo (barramento ocioso) para dominante, o que sinaliza a sincronização da rede. O identificador faz

referência à unidade de controle eletrônico (ECU) responsável pela transmissão. O campo SRR (*Substitute Remote Request*) ou RTR (*Remote Transmission Request*) é utilizado para solicitar dados de outra unidade, indicando que a mensagem não contém dados reais. A informação sobre a extensão do identificador é fornecida pelo IDE (*Identifier Extension*), um bit que determina o formato da CAN (GUIMARÃES, 2007).

Além disso, o quadro CAN contém os bits reservados (R1 e R0), o DLC (*Data Length Code*), que é um campo de 4 bits responsável por indicar o número de bytes no campo de dados, e o campo de dados propriamente dito, que pode transportar até 8 bytes de informação. O CRC (Cyclic Redundancy Check) é usado para verificar a integridade da mensagem, enquanto o ACK (Acknowledgment) confirma o recebimento da mensagem. O EOF (End of Frame), composto por sete bits recessivos, marca o fim da mensagem e não é afetado pelo mecanismo de *bit stuffing*, mecanismo explicado na sequência.

Outro mecanismo importante no protocolo CAN é o *bit stuffing*, utilizado para evitar longas sequências de bits iguais, que poderiam comprometer a sincronização dos receptores. Quando ocorrem cinco bits consecutivos de valor igual, o sistema automaticamente insere um bit de valor contrário. O receptor remove esses bits extras durante o processamento da mensagem, o que pode alterar o tamanho total percebido do quadro. Caso a mensagem contenha seis bits consecutivos iguais, ela será invalidada. Esse processo contribui para a integridade e sincronização da comunicação, mantendo o barramento livre de falhas (VECTOR INFORMATIK GmbH, 2023). Por fim, o *Bus Idle* é um espaço entre o fim de um quadro e o início de outro.

Durante a transmissão do quadro, o *bit monitoring* desempenha um papel fundamental na manutenção da comunicação estável, além de ser o responsável pela arbitragem. O módulo transmissor verifica se o bit enviado corresponde ao bit lido no barramento. Caso haja uma discrepância, como a transmissão de um bit dominante enquanto um recessivo é lido, o sistema detecta uma falha no barramento. Caso seja lido um dominante enquanto um recessivo é escrito, significa que o nó perdeu a prioridade de escrita, pois significa que um outro nó está escrevendo com um valor binário inferior (VECTOR INFORMATIK GmbH (2023).

Com o conhecimento das características, normas e estrutura das mensagens da rede CAN, torna-se possível extrair e analisar de forma precisa os dados transmitidos, o que é fundamental para o desenvolvimento de sistemas embarcados eficientes e confiáveis.

3 EXTRAÇÃO DE DADOS

Este capítulo inicia apresentando a estrutura das bancadas utilizadas no estudo, de modo a contextualizar como os dados podem ser acessados. Em seguida, os métodos de coleta de informações são explorados, para depois serem apresentadas as abordagens adotadas: o uso de um *sniffer* comercial e o desenvolvimento de um protótipo que opera como *sniffer* e decodificador.

3.1 Estrutura das bancadas

O Sistema de Treinamento com Ligação de Controles Separados para o Veículo Elétrico BYD Qin é composto pelos sistemas de gerenciamento de bateria, acionamento do motor, ar-condicionado, aplicações elétricas e sistema de direção assistida. Os sistemas são separados em bancadas didáticas, interconectados em linha de potência e em comunicação.

As bancadas contam com diversos pontos de conexão elétrica (bornes) estrategicamente distribuídos, identificados com base no manual técnico da *Beifang*. Esses terminais permitem a medição de sinais elétricos analógicos e digitais, proporcionando uma melhor compreensão do funcionamento dos sistemas veiculares. Entre os sinais disponíveis para medição, destacam-se os barramentos CAN, que fazem parte da linha interconectada de comunicação.

Conforme descrito nos manuais, as redes CAN são designadas como *Power Network CAN* (identificada, para fins deste trabalho, como CAN principal) e *Battery Controller Area Network* (identificada, para fins deste trabalho, como CAN de baterias). Ademais, são mencionadas algumas linhas secundárias de comunicação, como a *Start Subnet-CAN* e a *Confort Network CAN*, as quais serão abordadas de brevemente ao longo do texto.

A rede CAN principal percorre todos os sistemas, transportando informações de todos os módulos eletrônicos. Seus bornes de acesso, no entanto, estão localizados na bancada de direção assistida ou na porta OBD, que é encontrada na própria estrutura física da bancada de acionamento do motor. A CAN de baterias,

por sua vez, é um sistema isolado, acessado exclusivamente pelos bornes na bancada de baterias. A Figura 12 apresenta a ilustração geral.

Figura 12 – Sistemas disponíveis e acesso à CAN

Fonte: Autoria própria (2025).

Na bancada de direção assistida, por exemplo, a rede CAN principal é acessível por meio de bornes identificados como CAN_H e CAN_L e, além dos bornes, a bancada conta com duas telas que exibem as medições de tensão das linhas CAN_H e CAN_L, possibilitando a visualização dos valores médios em tempo real. Uma imagem da bancada é apresentada na Figura 13.

Figura 13 - Bancada de direção assistida com acesso a rede CAN

Fonte: Autoria própria (2025).

No entanto, essas medições não são suficientes para uma análise detalhada da comunicação na rede CAN, pois a transmissão de dados ocorre em alta velocidade, com variações rápidas de tensão entre os estados dominante e recessivo. Como o sinal é diferencial, um voltímetro comum exibirá apenas um valor médio próximo de 2.5V, sem capturar as transições entre os bits. Dessa forma, é necessário recorrer a métodos mais avançados de leitura, que serão apresentados em seguida.

3.2 Procedimentos e Técnicas para a Coleta

Para uma análise precisa da rede CAN, é essencial utilizar ferramentas adequadas, como um osciloscópio ou um analisador de protocolo CAN (*sniffer*), que pode ser um dispositivo comercial, disponível no mercado para diferentes aplicações, ou pode ser desenvolvido de forma personalizada, utilizando os componentes

eletrônicos necessários para a aplicação. Neste subcapítulo, serão apresentados os diferentes métodos disponíveis para realizar essa leitura.

3.2.1 Leitura por meio de osciloscópio

Um osciloscópio comum permite visualizar os pulsos elétricos ao longo do tempo e identificar a atividade no barramento, mas, ao contrário de um osciloscópio específico para CAN, não realiza a decodificação automática das mensagens.

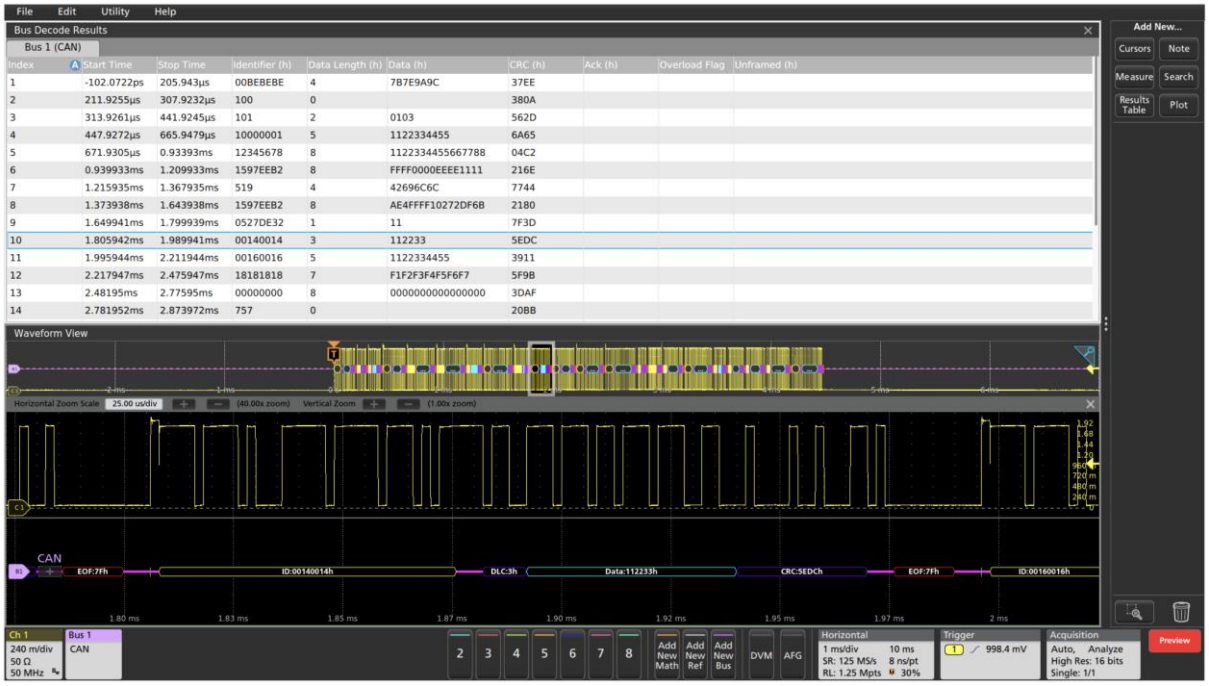
Ao usar um osciloscópio comum, recomenda-se medir o sinal de forma diferencial, posicionando uma sonda em CAN_H e a outra em CAN_L. A função matemática do osciloscópio pode então ser usada para exibir a diferença entre esses dois sinais. A escala de tensão deve ser ajustada para aproximadamente ±5V, o que permite observar as variações entre os estados dominante e recessivo. Isso resultará em uma leitura semelhante à apresentada anteriormente na Figura 5, na subseção 2.1.

Para iniciar a leitura do sinal, é fundamental determinar a velocidade de transmissão do barramento. Isso pode ser feito medindo o tempo de duração de um único bit com um osciloscópio. A partir dessa medição, é possível calcular a taxa de transmissão do barramento aplicando a relação inversa da duração do bit. Por exemplo, se a duração de um bit for de 10 µs, a taxa de transmissão será de 100 kbps. Sem essa determinação da velocidade, não é possível identificar corretamente a separação entre os bits, entender a estrutura das mensagens ou distinguir as transições entre os estados dominante e recessivo.

Com a velocidade do barramento conhecida, pode-se então iniciar a leitura bit a bit de um *frame*. Para identificar o padrão CAN (padrão ou estendido), é preciso observar o campo IDE (*Identifier Extension Bit*), conforme o modelo de mensagem da seção 2.3. Após identificar o padrão, a leitura do frame pode ser validada conforme o número de bits esperados para cada tipo de padrão, respeitando as variações previstas. Além disso, ao capturar um frame completo, é necessário validá-lo de acordo com a estrutura da mensagem CAN, verificando erros de transmissão, como erros de bit ou erro de CRC, que podem invalidar o frame.

Caso a leitura seja feita por um osciloscópio comum, a interpretação dos sinais dentro do *frame* precisará ser manual. Já um osciloscópio com decodificação CAN, os dados já são exibidos automaticamente, facilitando a análise das mensagens e permitindo uma compreensão mais clara. Uma imagem com a leitura por meio de um osciloscópio com decodificação CAN está apresentado na Figura 14.

Figura 14 – Leitura CAN por osciloscópio com decodificação



Fonte: TEKTRONIX, 2018.

Para que um osciloscópio com decodificação CAN apresente a decodificação requerida, é necessário configurar corretamente a taxa de transmissão (kbps) do barramento. Caso a velocidade configurada esteja errada, o osciloscópio ainda exibirá os pulsos elétricos no barramento, mas a interpretação dos frames será incorreta ou inexistente. A taxa de transmissão pode ser obtida de duas maneiras: (i) como descrito anteriormente, determinando a duração do bit e calculando a taxa, ou (ii) por meio de tentativa e erro, ajustando a velocidade até que a decodificação das mensagens seja exibida corretamente no osciloscópio.

Após a configuração correta da taxa de transmissão, o osciloscópio passa a apresentar automaticamente os resultados das mensagens em formatos binários

e/ou hexadecimais. Além disso, em muitos osciloscópios é possível configurar *triggers* específicos para capturar mensagens com determinados IDs, detectar erros na comunicação ou filtrar *frames* de interesse. Dessa forma, o processo de análise se torna muito mais eficiente, eliminando a necessidade de converter os pulsos em mensagens binárias.

Entretanto, mesmo com a separação dos valores hexadecimais e a correspondência facilitada entre o identificador e os dados transmitidos, a interpretação das mensagens ainda apresenta desafios. Isso ocorre porque, em muitos casos, as mensagens relevantes extraídas de um frame CAN com dados úteis de 8 bytes utilizam apenas uma fração desse espaço, podendo ocupar poucos bytes ou até mesmo um único bit para transmitir informações específicas. Como esses bits estão distribuídos dentro do campo de dados do frame e a representação padrão permanece em formato hexadecimal, a interpretação manual ainda é necessária para identificar corretamente o significado de cada bit individual.

Para lidar melhor com essas dificuldades e tornar a análise mais eficiente, é interessante recorrer a ferramentas mais avançadas, como *sniffers* comerciais. Na próxima seção,serão abordados esses dispositivos e as vantagens de utilizados em relação aos osciloscópios dedicados.

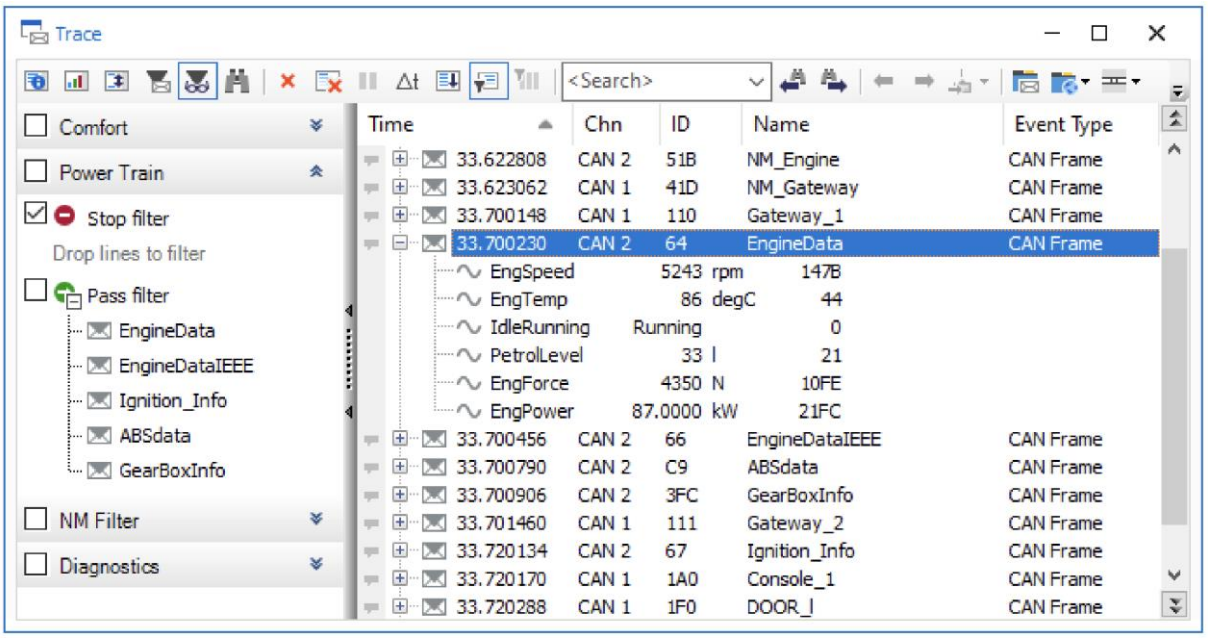
3.2.2 Leitura por meio de s*niffer c*omercial

Os *sniffers* comerciais de CAN, em conjunto com *softwares* dedicados, permitem visualizar e manipular os bits de forma estruturada, tornando possível separá-los, reorganizá-los e aplicar regras específicas para sua decodificação. A configuração adequada do software permite, por exemplo, definir máscaras de bits, aplicar fatores de escala e converter automaticamente os valores brutos para unidades de medida compreensíveis.

A Figura 15 exemplifica essa funcionalidade por meio de uma das telas do *sniffer* CANoe, um software amplamente utilizado para análise CAN. Nessa interface, é possível visualizar um frame CAN já decodificado e organizado, apresentando informações como a rotação do motor em RPM e a temperatura do motor em graus Celsius. Além disso, um dos dados que é apresentado na imagem é a mensagem de

status do veículo, que no nível mais baixo é um dado binário que indica se o veículo está em estado *idle* (parado) ou *running* (em funcionamento).

Figura 15 – Decodificação CAN por meio de *sniffer* comercial



Fonte: VECTOR INFORMATIK GmbH, 2023.

É importante ressaltar que as informações transmitidas pelo barramento CAN não são imediatamente compreensíveis. O software de um *sniffer* comercial desempenha um papel essencial em facilitar a visualização dos dados, mas para isso é necessário configurar corretamente a decodificação das mensagens. O usuário deve definir parâmetros de cada mensagem decodificada, como a posição dos bits dentro do *frame*, o formato numérico (*Big Endian* ou *Little Endian)*, os fatores de escala e offsets aplicáveis, além das unidades de medida apropriadas. Sem essa configuração, os dados permanecerão em um formato bruto.

Uma das grandes vantagens dos *softwares* associados aos *sniffers* é a facilidade de validação dos dados. Embora a decodificação ainda requeira expertise do usuário para identificar a resolução das mensagens, após a configuração, os valores decodificados são automaticamente apresentados. O papel do usuário, então, é avaliar se os parâmetros seguem dentro dos limites esperados. Por exemplo, a

rotação do motor deve permanecer dentro de um intervalo coerente, assim como a temperatura do motor, e esses dados, já decodificados, podem ser exibidos de forma legível, sem a necessidade de interpretação do formato hexadecimal. Assim, possíveis incoerências podem ser encontradas de maneira mais fácil.

Além disso, alguns softwares oferecem a funcionalidade de injeção de mensagens no barramento, permitindo simular condições específicas e testar o comportamento do sistema. Essa capacidade também contribui para a validação dos dados. Por exemplo, se o status do farol estiver identificado em um bit específico dentro de um ID, a injeção de uma mensagem correspondente pode simular o acionamento do farol. Caso o veículo possua um sistema de monitoramento para indicadores de farol, como um ícone no painel, ao escrever essa mensagem no barramento, o sistema poderá registrar que o farol está ligado e exibir o ícone correspondente no painel.

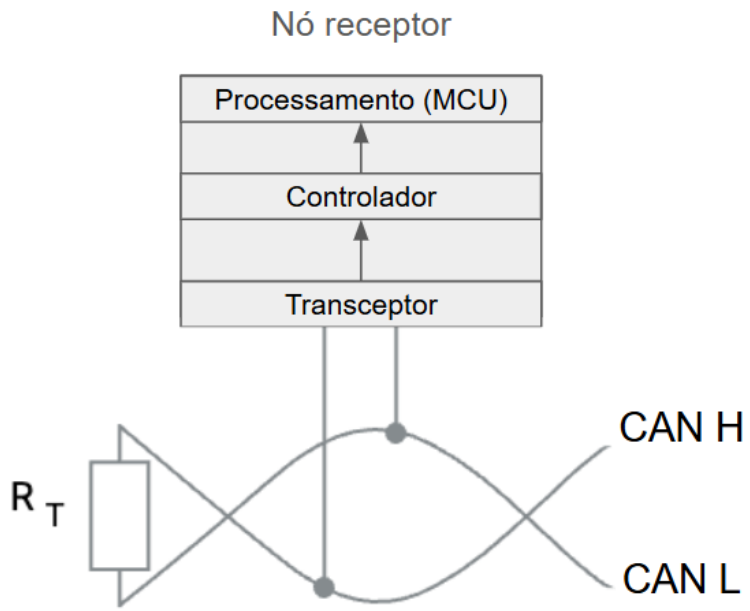
Embora os *sniffers* comerciais sejam ferramentas poderosas para análise do barramento CAN, seu alto custo pode ser um obstáculo, especialmente para desenvolvedores independentes ou projetos que não justifiquem um grande investimento. Diante disso, uma alternativa viável é a criação de um *sniffer* próprio, utilizando hardware acessível e software customizado para atender às necessidades específicas da aplicação.

3.2.3 Criação de s*niffer p*róprio

O desenvolvimento de um *sniffer* dedicado à leitura da rede CAN deve estar em conformidade com as diretrizes apresentadas na Subseção 2.2 deste trabalho, que fazem referência às normas ISO 11898 e ISO 11519-2.

De forma resumida, para garantir a correta integração do *hardware* à rede CAN, é necessário que um nó tenha um transceptor CAN para a conversão entre níveis de tensão, um controlador CAN para gerenciar a comunicação e uma aplicação de alto nível implementada algum tipo de atuador ou executada no microcontrolador. Demais considerações podem ser encontradas ao longo da seção 2 do presente trabalho. Um diagrama dessa aplicação básica é fornecido através da Figura 16, a seguir.

Figura 16 – Diagrama básico de um nó receptor na rede CAN



Fonte: Autoria própria (2025).

Após a escolha do transceptor, do controlador e do microcontrolador, é essencial seguir todas as diretrizes fornecidas pelos fabricantes, incluindo o *datasheet* e o *design guide*. Isso garante que os CIs atuem conforme esperado. Componentes ou circuitos adicionais podem ser incorporados conforme as recomendações do fabricante para aprimorar a funcionalidade, robustez e proteção do *design* de *hardware*.

Utilizando um controlador CAN integrado ao microcontrolador, como o ESP32, a implementação em firmware é simplificada devido à API fornecida pela *Espressif* (*driver/twai.h*), que facilita a configuração do controlador, a definição de filtros e o gerenciamento de interrupções. No caso da manipulação das mensagens, a estrutura *can_message_t* é utilizada para armazenar os dados da mensagem CAN, como o apresentado na Figura 17 (Espressif,2023).

Figura 17 – Estrutura *can_message_t* da biblioteca CAN da Espressif

```
typedef struct {
    uint32_t id;         // Identificador da mensagem (11 ou 29 bits)
    uint8_t dlc;         // Código de comprimento de dados (0-8 bytes)
    uint8_t data[8];     // Payload da mensagem (até 8 bytes)
    uint8_t flags;       // Flags que determinam o tipo de quadro e direção de transmissão
} can_message_t;
```

Fonte: Adaptado de Espressif (2023).

Quando uma mensagem CAN é recebida, portanto, ela é automaticamente organizada na estrutura *can_message_t.* Dessa maneira, o desenvolvedor pode acessar diretamente os dados de interesse, como o conteúdo do campo de dados, sem precisar se preocupar com os detalhes do formato bruto da mensagem.

No caso de um controlador CAN externo, o microcontrolador precisa configurar a interface de comunicação, como SPI (no caso do MCP2515), e enviar os comandos necessários para sua inicialização. Isso envolve a definição da taxa de *baud rate*, a configuração de máscaras e filtros de mensagens, além da escolha dos modos de operação, que são armazenados nos registradores internos do controlador. Embora seja possível programar essa comunicação manualmente, bibliotecas como *mcp_can.h* para Arduino e ESP32 simplificam esse processo, abstraindo a configuração dos registradores e permitindo o envio e recebimento de mensagens sem a necessidade de manipular diretamente os comandos SPI (Fowler, 2017).

Mesmo que a mensagem seja armazenada em uma estrutura pré-definida, no campo de dados, as mensagens ainda são recebidas em formato hexadecimal e precisam ser tratadas. Dessa forma, é sabido que, independentemente da complexidade da ferramenta utilizada, o processo de decodificação é sempre necessário.

3.3 Abordagens metodológicas utilizadas

As abordagens metodológicas adotadas podem ser divididas em dois pontos principais: a visualização bruta dos dados, que envolve o uso de ferramentas como o *sniffer* comercial ou o protótipo funcionando como *sniffer*, juntamente com a coleta de logs, e a visualização codificada e aprimorada, que abrange a reformulação de firmware para permitir a aplicação de filtros, resolução dos dados e a utilização de diversas facilidades para tornar a análise mais eficiente. A atual subseção se divide, portanto, nas duas técnicas de extração de dados utilizadas.

3.3.1 Leitura de dados brutos

Foi inicialmente utilizado um *sniffer* comercial para a captura e análise dos dados no barramento CAN, combinando o *software Vehicle Spy* com o *hardware ValueCAN*. Após indisponibilidade do equipamento, todavia, foi recorrido à criação de um *sniffer* próprio. Nesta subseção, portanto, são apresentados os detalhes da obtenção das mensagens, incluindo, principalmente, as definições de *hardware e firmware* que habilitaram o *sniffer* próprio.

3.3.1.1 Utilização de sniffer comercial

A escolha do *Vehicle Spy* para o projeto foi determinada pela familiaridade pré-existente da autora com o *software*, adquirida em seu ambiente profissional, além da disponibilidade inicial do equipamento. O *software* foi então utilizado para as tarefas iniciais do projeto, como a análise do barramento CAN, a coleta de dados e a validação de sua aquisição. O *software* se mostrou uma ferramenta eficaz devido à sua interface intuitiva e às funcionalidades de análise avançada, que possibilitaram um acompanhamento preciso das comunicações no barramento CAN.

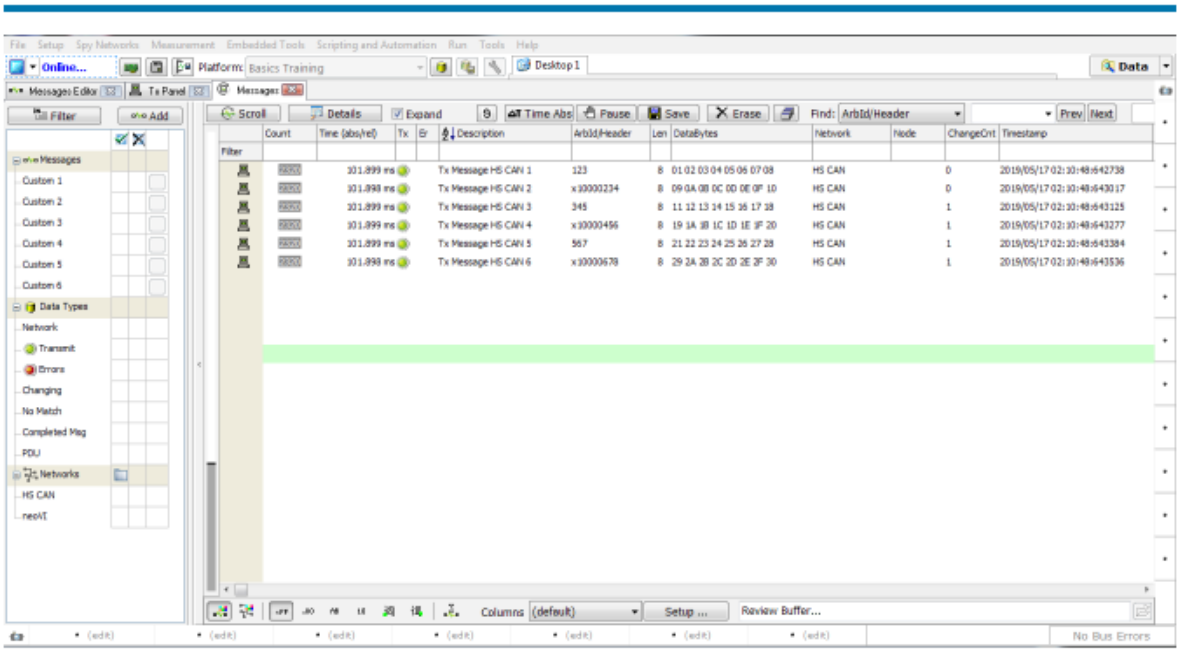
O *ValueCAN* estabelece a conexão física com a rede CAN selecionada, capturando os dados e transmitindo-os digitalmente para o *Vehicle Spy* por meio de uma conexão USB com o computador onde o software está em execução. Um exemplo é apresentado na Figura 18.

Figura 18 – Conexões elétricas entre os sistemas comerciais

Fonte: Autoria própria (2025).

No *Vehicle Spy*, os bits recebidos são organizados em quadros, como demonstrado na Figura 19, e semelhante ao software CANoe, os dados podem ser configurados para serem analisados com resolução aplicada. Além disso, ele permite a realização de registros de dados (*datalogging*), possibilitando a captura de informações para análises posteriores. A extração de dados por meio dos logs foi, portanto, um aspecto essencial para o desenvolvimento deste trabalho.

Figura 19 – Interface do software Vehicle Spy



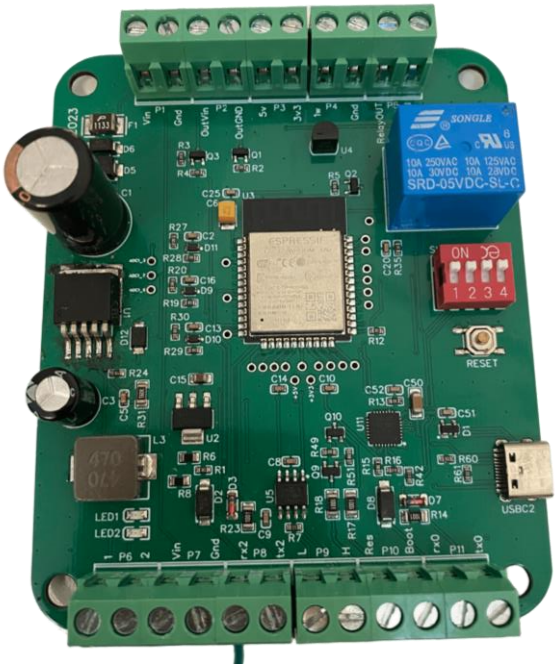
Fonte: INTREPID CONTROL SYSTEMS (2025).

Contudo, durante o progresso do projeto, a indisponibilidade do *Vehicle Spy* para a continuidade das atividades forçou a autora a buscar uma solução alternativa. Para isso, foi necessário empregar um *hardware* próprio para a conexão com a bancada, o que permitiu continuar as análises sem o *software* dedicado.

3.3.1.2 Especificações de hardware para protótipo

O protótipo utilizado foi desenvolvido totalmente pela autora, através do *software Altium Designer*, e contém funcionalidades adicionais além das necessárias para este projeto, pois foi originalmente criado como um simulador CAN na empresa onde a autora trabalhava com redes CAN. Sendo assim, apenas os blocos necessários ao projeto em questão serão apresentados. Há uma foto da PCB apresentada na Figura 20.

Figura 20 – Foto do protótipo



Fonte: Autoria própria (2025).

No contexto do *hardware*, o sistema pode ser resumido em blocos principais:

a) Alimentação: entrada de 12VDC, a qual supre duas fontes reguladas responsáveis por fornecer tensões de 5 V e 3.3 V, garantindo níveis adequados para os componentes do circuito. O ESP32 é alimentado em 3.3 V e o transceptor CAN em 5 V.

b) Processamento central: realizado pelo ESP32, um microcontrolador que integra um controlador CAN.

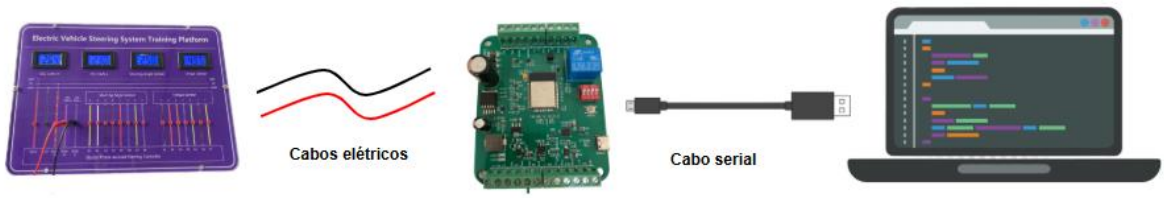
c) Transceptor: A comunicação no barramento CAN é intermediada por um transceptor MCP2551, fabricado pela Texas Instruments.

d) Conexão física: Para conexão com o barramento, foram empregados conectores *Metaltex*, de encaixe parafusado. Os conectores também foram utilizados para alimentar a placa em uma tensão de 12 V.

e) Interface: Viabilizada por um conversor USB, que possibilita tanto a programação do dispositivo quanto o monitoramento das mensagens trocadas no barramento por conexão serial entre o protótipo e o computador.

Sendo assim, o *hardware* foi apto a se integrar na rede como receptor, atuando como um *sniffer*, ou seja, realizando funções similares às do *ValueCAN*. Essa abordagem possibilitou a realização de validações adicionais com os dados previamente coletados, garantindo que os testes e as verificações seguissem no ritmo do desenvolvimento do projeto. A conexão elétrica entre os sistemas é apresentada na Figura 21.

Figura 21 – Conexões elétricas entre os sistemas próprios



Fonte: Autoria própria (2025).

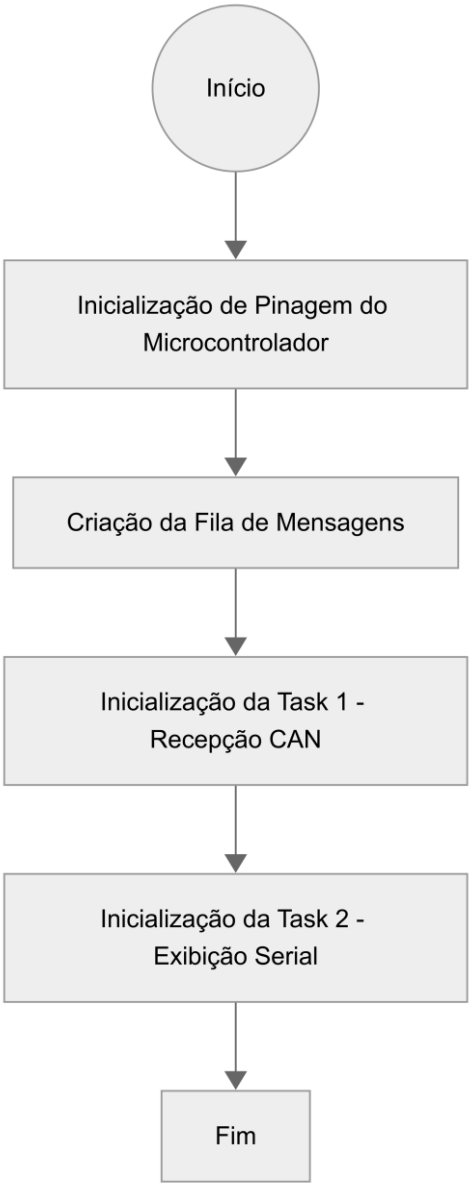
Com as condições de *hardware* atendidas, portanto, o desenvolvimento de *firmware* foi iniciado.

3.3.1.3 Especificações de firmware para protótipo

O *firmware* foi implementado no microcontrolador utilizando a plataforma *Visual Studio Code*, em linguagem C com *freeRTOS,* um sistema operacional em tempo real que permite criar e gerenciar múltiplas tarefas com prioridades definidas, para a utilização é incluído a biblioteca *FreeRTOS.h.* A análise dos dados foi realizada através da interface serial do computador, conectada ao microcontrolador, sem o desenvolvimento de uma interface dedicada como o *Vehicle Spy* ou *CANoe*.

O código foi estruturado em duas tarefas (*tasks*) distintas, ambas inicializadas a partir da função principal (*main*), conforme diagrama na Figura 22. A inicialização e controle das *tasks* foi realizada a partir da biblioteca *task.h.* Na função principal é também iniciado uma fila de mensagens utilizando a biblioteca *queue.h, e* essa fila é compartilhada entre as duas *tasks*. A inicialização dos pinos do ESP32 ocorre na função principal, logo no início da operação. Esses pinos são definidos para atender tanto à comunicação serial quanto ao barramento CAN, que serão configurados dentro das respectivas *tasks*.

Figura 22 – Diagrama de operação da Função Principal

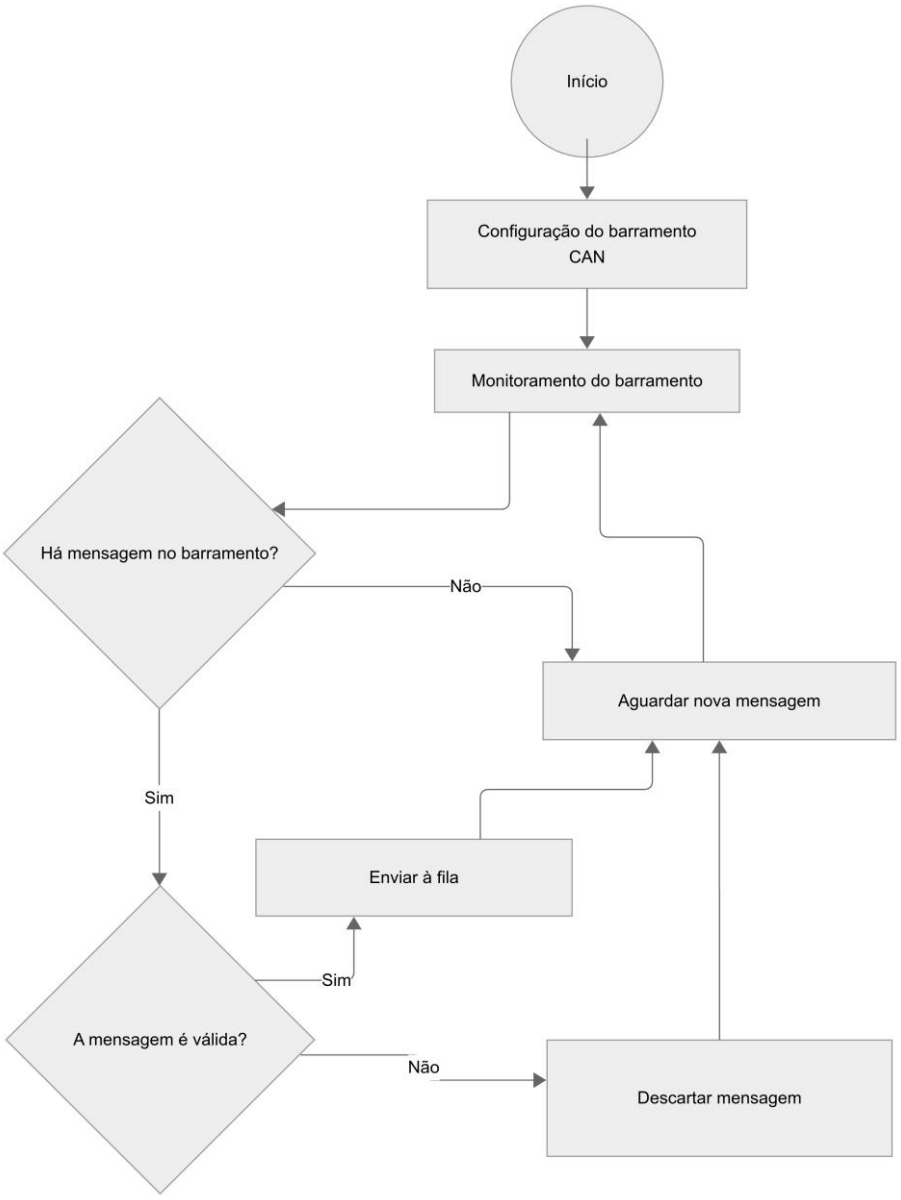


Fonte: Autoria própria (2025).

Na primeira *task, são* realizadas as configurações do barramento *CAN a* partir da biblioteca *twai.h,* antes do início do loop. Essas configurações são, por exemplo, a velocidade do barramento. No loop, a *task* se torna responsável por monitorar continuamente o barramento CAN, verificando se há novas mensagens. Quando uma mensagem válida é detectada, ela é enviada para uma fila de mensagens, permitindo que as informações sejam armazenadas temporariamente

para posterior processamento, o diagrama que apresenta essa operação está na Figura 23.

Figura 23 – Diagrama de operação da Task 1: Recepção CAN

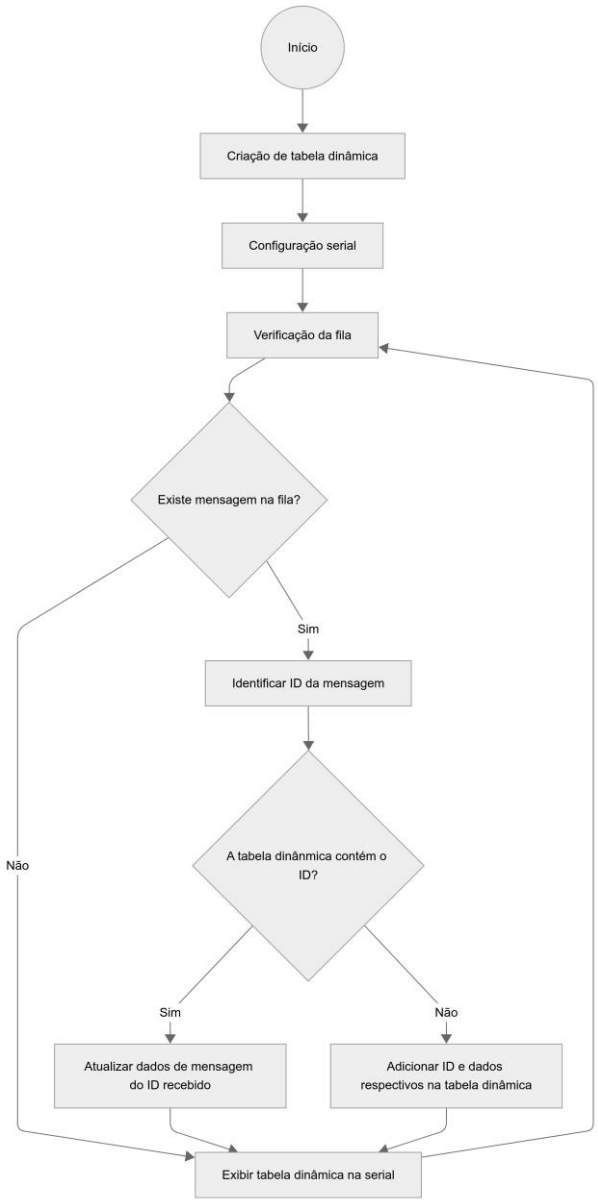


Fonte: Autoria própria (2025).

Paralelamente, a segunda *task* verifica constantemente essa fila e, ao encontrar uma nova mensagem, atualiza uma tabela dinâmica que organiza os dados

por identificador (ID). A tabela atual é exibida na interface serial no fim de cada *loop*, proporcionando uma visualização contínua e atualizada das mensagens recebidas. A operação pode ser visualizada no diagrama da Figura 24. Essa abordagem garante que a recepção das mensagens ocorra sem interrupções, enquanto a exibição dos dados é gerenciada de forma independente. As bibliotecas padrão *stdio.h* e *stdlib.h* são utilizadas para manipulação de entrada/saída de dados e alocação dinâmica de memória, respectivamente. Para a serial, são utilizados os *drivers* do ESP.

Figura 24 – Diagrama de operação da Task 2: Exibição Serial



Fonte: Autoria própria (2025).

Além disso, para a captura e organização dos logs, foi desenvolvido um código em Python, uma linguagem de programação de alto nível. Esse código organiza os dados recebidos pela interface serial em uma planilha, estruturando-os com colunas para o identificador e o campo de dados CAN. Para essa operação, foram utilizadas as bibliotecas *pySerial* para comunicação com a interface serial e *csv* para manipulação e armazenamento dos dados em uma planilha.

Os resultados das extrações de dados brutos, tanto pelo sniffer comercial quanto pelo protótipo, bem como as análises dos logs, serão apresentados em uma seção específica ao longo do texto.

3.3.2 Leitura codificada

Embora a coleta de dados tenha continuado após as parametrizações, não foi mais necessário capturar todas as informações do barramento. Agora, o foco está em coletar apenas as mensagens relacionadas aos parâmetros selecionados. Como o protótipo é uma ferramenta volátil e de fácil modificação de *firmware*, foi implementada uma nova abordagem: exibir apenas as mensagens de interesse, já decodificadas. Este capítulo descreve as modificações no firmware que tornaram essa operação possível.

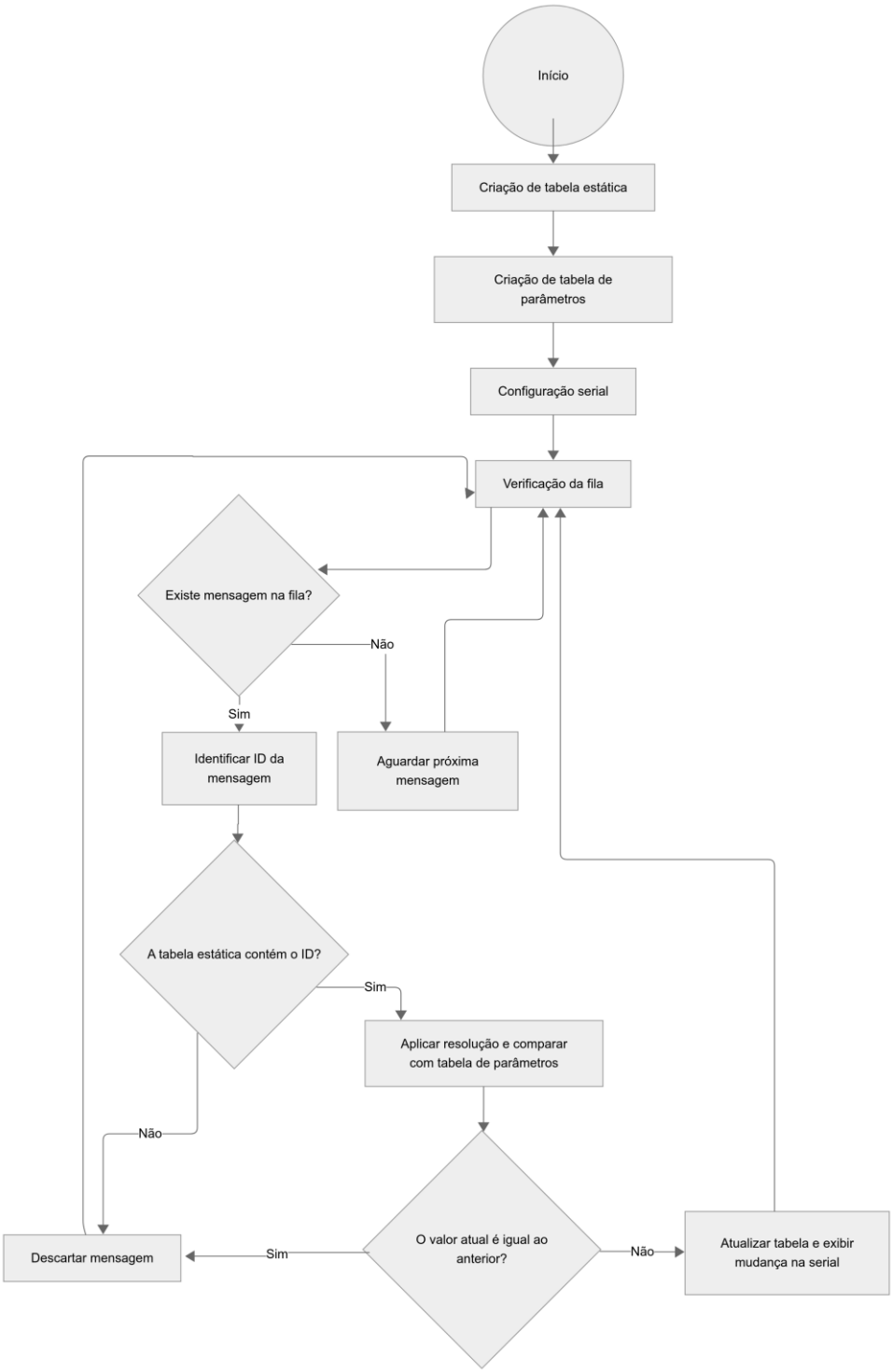
3.3.2.1 Redefinição de firmware do protótipo

Para a nova operação requerida, alterações no firmware foram realizadas na *task* 2. A tabela dinâmica de mensagens foi alterada para conter apenas os IDs de interesse, tornando-se estática. Além dessa tabela, foi criada uma nova tabela focada nos parâmetros, que não só faz associações, mas também aplica a resolução de cada um, transformando os valores binários nos valores físicos reais. Diversas mensagens podem compartilhar o mesmo ID, mas com resoluções diferentes, por isso a tabela de parâmetros foi, portanto, criada para garantir que cada parâmetro fosse tratado de forma adequada.

A tabela de parâmetros também foi estruturada para armazenar o valor decodificado atual e o último valor recebido, facilitando o acompanhamento das mudanças ao longo do tempo.

Quando uma mensagem CAN é recebida na fila da *Task* 2, a tabela geral de IDs verifica se o ID da mensagem já está registrado. Caso o ID não seja encontrado, a mensagem é descartada. Se o ID estiver presente, o campo de dados correspondente é atualizado. Em seguida, é aplicado a resolução referente àquele parâmetro e comparado com a tabela de parâmetros. O sistema então compara os o valor recebido para verificar se há diferenças com o último armazenado. Se uma mudança for detectada, a alteração é registrada na porta serial. Nesse modo de operação, o *loop* da *task* 2 apenas aguarda a chegada de uma nova mensagem na fila, sem exibir constantemente os dados armazenados. A Figura 25 apresenta a operação em forma de diagrama, facilitando o entendimento.

Figura 25 – Diagrama de operação da *Task* 2 pós redefinição



Fonte: Autoria própria (2025).

4 DECODIFICAÇÃO

Neste capítulo, é discutido o processo de decodificação das informações extraídas do barramento. São destacadas as características operacionais da rede CAN do veículo BYD, como velocidade do barramento, a identificação do padrão das mensagens, o processo de interpretação de bytes e bits, o processo de parametrização e a análise de logs.

4.1 Velocidade do barramento

Tanto o *sniffer* comercial quanto o desenvolvido, precisam estar configurados com a velocidade correta da rede CAN para estabelecer comunicação e receber mensagens. Para isso, foram testadas velocidades padrão, como 125 kbps, 250 kbps, 500 kbps e 1 Mbps, até que a recepção de dados fosse estabelecida. Como o número de opções é limitado, a estimativa da velocidade correta pode ser realizada por meio do método de tentativa e erro.

O teste confirmou que a velocidade de 500 kbps é a adequada para a CAN principal. A captação das mensagens ocorreu exclusivamente nessa taxa, indicando que a comunicação estava sincronizada e os dados estavam sendo transferidos à velocidade necessária para o funcionamento correto do sistema.

Em relação às demais redes CAN acessíveis nas bancadas, foi observada uma diferença significativa em suas velocidades de operação. A CAN de baterias opera a uma velocidade de 250 kbps, o que é adequado para os dados que trafegam nela, uma vez que envolvem informações relacionadas ao estado da bateria, como carga e saúde, que não exigem comunicação tão rápida quanto os sistemas principais. A monitorização da bateria, por exemplo, é uma tarefa que ocorre em intervalos mais longos e com dados que não exigem uma resposta imediata, o que justifica a escolha dessa velocidade.

Foi identificado, também, que a rede CAN principal é a mesma utilizada pela porta OBD do veículo, o que confirma sua importância central na comunicação do sistema do veículo. A porta OBD é utilizada para diagnóstico e monitoramento do

desempenho do veículo, permitindo que técnicos e dispositivos externos acessem dados essenciais do sistema do veículo.

Outras redes são identificadas no sistema no manual da *Beifang*, mas, a partir dos testes de leitura, foi possível verificar que essas redes pertencem a sistemas secundários, com um número limitado de IDs e operando a velocidades mais baixas. As redes *Start Subnet-CAN* e *Confort Network CAN*, por exemplo, funcionam a 125 kbps, evidenciando dados não tão críticos ou volumosos.

Observou-se que as mesmas mensagens transmitidas pelas redes secundárias também estão presentes na rede CAN principal, eliminando a necessidade de um mapeamento individual para cada uma delas. Além disso, o acesso às redes secundárias em veículos em sua configuração final, isto é, completamente montados, apresenta desafios significativos, tornando a utilização da rede principal uma alternativa mais eficiente e prática para a extração e análise dos dados.

A única exceção registrada foi a rede CAN das baterias, que não transmitia todos os seus IDs para a rede principal. Apenas uma de suas mensagens era compartilhada entre as duas redes. Essa particularidade diferencia a rede de baterias das demais redes secundárias do sistema, configurando-a como um subsistema isolado e de especial interesse para análise.

4.2 Identificação do padrão de mensagens

A rede CAN principal e as redes secundárias utilizam o padrão CAN 2.0A, que oferece uma estrutura de endereçamento simples e é adequado para redes com um número reduzido de nós. Já a rede CAN das baterias adota o padrão CAN 2.0B, que permite maior capacidade de endereçamento e suporte a mais dispositivos. A escolha entre os padrões depende da complexidade do sistema, sendo o 2.0B mais adequado para sistemas que precisam de maior escalabilidade e robustez.

O protocolo CAN 2.0A oferece até 2.048 identificadores, quantidade geralmente suficiente para a maioria dos sistemas veiculares. Já o CAN 2.0B expande

essa capacidade para até 536 milhões de identificadores, um número que pode parecer excessivo, especialmente considerando que o sistema atual de baterias utiliza apenas uma pequena fração dos identificadores disponíveis no CAN 2.0A.

No entanto, a adoção do CAN 2.0B garante a escalabilidade do sistema, permitindo futuras expansões sem restrições de endereçamento. Isso é vantajoso tanto para aplicações em veículos pesados quanto para a implementação da mesma configuração em soluções industriais, eliminando a necessidade de reconfigurações na comunicação.

Portanto, embora o CAN 2.0A atenda às necessidades atuais, o CAN 2.0B se destaca como uma escolha mais flexível e preparada para o futuro. Essa abordagem evita gargalos na rede de baterias, permitindo que o sistema acompanhe novas configurações e a adição de dispositivos sem comprometer a eficiência da comunicação.

4.3 Interpretação de bytes e bits

A partir das características operacionais de velocidade e do padrão de rede, inicia-se o processo de identificação dos parâmetros. Para isso, são aplicadas perturbações no sistema com o objetivo de detectar variações, as quais se manifestam tanto como grandezas físicas reais para o observador quanto como variações de bits para o nó leitor na rede. Essas variações são então correlacionadas, permitindo a criação de modelos matemáticos que associam as variações no barramento ao valor real da grandeza operacional.

O processo é então iniciado com testes experimentais em bancada. Durante esses testes, o sistema fica constantemente trocando informações entre os nós da rede, o que pode dificultar a identificação precisa dos bytes que representam as grandezas físicas desejadas. A estratégia adotada consiste em minimizar o número de controles alterados simultaneamente, a fim de evitar interferências. Ou seja, as perturbações realizadas devem ser controladas e espaçadas no tempo, para reduzir a influência entre elas.

Dessa forma, é possível identificar bits ou bytes candidatos para representar os parâmetros correspondentes às perturbações aplicadas. O registro referente à leitura de um único identificador em cenário de perturbação pode ser visto no Quadro 2. Esse registro específico foi realizado com o auxílio do protótipo desenvolvido.

Quadro 2 – Log de leitura em perturbação da CAN em ID específico

Byte 0	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7
30	75	FE	0	0C	A1	AF	0
30	75	FE	2	0C	A1	BF	EE
30	75	FE	3	0C	A1	CF	DD
30	75	FE	5	0C	A1	DF	CB
30	75	FE	7	0C	A1	EF	B9
30	75	FE	9	0C	A1	FF	A7
30	75	FE	0A	0C	A1	0F	94
30	75	FE	0C	0C	A1	1F	82
30	75	FE	10	0C	A1	2F	70
30	75	FE	12	0C	A1	3F	5E
30	75	FE	14	0C	A1	4F	4C
30	75	FE	16	0C	A1	5F	3A
30	75	FE	17	0C	A1	6F	29
30	75	FE	18	0C	A1	7F	18
30	75	FE	19	0C	A1	8F	7
30	75	FE	1A	0C	A1	9F	F6
30	75	FE	1B	0C	A1	AF	E5
30	75	FE	1C	0C	A1	BF	D4
30	75	FE	1E	0C	A1	CF	C2
30	75	FE	1F	0C	A1	DF	B1
30	75	FE	20	0C	A1	EF	A0
30	75	FE	22	0C	A1	FF	8E
30	75	FE	23	0C	A1	0F	7D
30	75	FE	25	0C	A1	1F	6B
30	75	FE	27	0C	A1	2F	59

Fonte: Autoria própria (2025).

Embora outros bytes também apresentem alterações no comportamento do registro de log, o byte 3, nesta análise específica, se destaca como o principal candidato. As variações observadas entre os registros coincidem diretamente com as variações físicas realizadas. No caso em questão, a pressão no pedal do acelerador estava sendo mapeada, e o byte 3 demonstrou uma resposta clara: ele incrementava quando o pedal era pressionado e decrementava quando o pedal era solto. Além disso, mantinha-se estático durante o período em que não havia intervenção no controle. Esse comportamento consistente do byte 3, alinhado às variações físicas no

pedal do acelerador, o torna o candidato mais relevante para representar essa grandeza operacional no barramento CAN.

Nem sempre os valores incrementam ou decrementam de forma direta e associada às grandezas requeridas, o que demanda a criação de fatores de calibração. Os dados na rede CAN podem ser analógicos, como a velocidade, ou digitais, representando valores binários, como o status de ativação ou desligamento de um componente. Para os valores analógicos, podem ser adotadas escalas lineares dentro de um intervalo contínuo. Já os valores discretos, como status ou comandos ligados/desligados, geralmente são representados por valores binários (0 ou 1) e não requerem calibração linear, pois não variam ao longo de um intervalo, mas mudam de forma binária, conforme os eventos.

Para os valores analógicos, a decodificação é realizada utilizando dois pontos de referência conhecidos, suficientes para definir uma reta de resolução. Por exemplo, ao mapear a posição do pedal do acelerador, as referências adotadas são a posição "pé não pressionado" (0%) e a posição "pé totalmente pressionado" (100%). Esses pontos, considerados como valores reais e conhecidos, são correlacionados aos valores binários obtidos no byte de alteração correspondente. A equação de decodificação gerada inclui dois componentes principais: o multiplicador e o offset.

O multiplicador é responsável pela razão entre a variação do parâmetro físico e a variação binária correspondente, considerando os dois pontos de medição. Já o offset representa o deslocamento da reta de medição. Ele é necessário para ajustar a medição, por exemplo, caso o ponto inicial da medição (0% de pressão no pedal) não corresponda exatamente a zero na grandeza binária.

A reta linear de resolução para grandezas não binárias, portanto, segue o padrão linear apresentado em 1.1 e 1.2, a seguir:

$$y(x) = a \cdot x + b, \qquad x \in [a, b],\ x \in \mathbb{Z} \tag{1.1}$$

$$a = \frac{\Delta y}{\Delta x} \tag{1.2}$$

Nos quais:

$y =$ Representa o valor resultante da grandeza medida, expresso nas unidades correspondentes à grandeza associada à medição.

$x =$ É o valor numérico obtido a partir dos bits selecionados, convertido para a representação decimal. É sempre um valor inteiro e tem intervalo definido.

$a =$ É o multiplicador, obtida pela equação 1.2.

$b =$ É o ajuste de *offset.*

$\Delta y =$ É a diferença obtida entre dois valores conhecidos em y, nas grandezas associadas à medição.

$\Delta x =$ É a diferença obtida entre dois valores conhecidos em x, nos quais correspondem aos mesmos valores conhecidos em y.

Para grandezas binárias ou reais com opções de resultados definidos, a equação pode ser representada conforme a Equação 2.1. Quando binário, os valores de x só assumem os valores 0 e 1, como a mostrado em 2.2. Já para variáveis com um conjunto restrito de valores definidos, é aplicado a Equação 2.3 como condição de x

$$y(x) = x, \tag{2.1}$$

$$y \in \{0,1\} \tag{2.2}$$

$$y \in \{a, b, c\} \tag{2.3}$$

Em um primeiro momento, as alterações são identificadas pelas variações de bytes no barramento, mas nem sempre um byte inteiro é utilizado para representar uma grandeza física. Sendo assim, a análise é ampliada para dentro do byte correspondente, permitindo a visualização dos bits individuais e como eles correspondem às mudanças de controle físicas. Dessa forma, é possível aprimorar a identificação das grandezas e extrair dados com maior precisão, em um nível mais preciso de interpretação.

O processo envolve a identificação da posição do bit inicial dentro do campo de leitura e a especificação da quantidade total de bits que compõem o valor extraído. Para ilustrar o procedimento, no Quadro 3, é demonstrado um exemplo em que o byte 3 foi selecionado e expandido, e uma quantidade de bits foi selecionada para leitura.

Quadro 3 – Exemplo de extração de bits dentro de um byte

Byte inicial	3							
Bit inicial	2							
Quantidade de bits	4							
Mensagem completa	Byte 0	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7
Bytes da mensagem	30	75	FE	AF	0C	A1	0F	A0
Byte 3 selecionado	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Bits presentes no byte 3	1	0	1	0	1	1	1	1
Dado obtido	Binário	1011						
	Decimal	11						

Fonte: Autoria própria (2025).

A ordem de leitura dos bits dentro de um byte segue o padrão da esquerda para a direita (sequência 0 a 7 no Quadro 3). Para os bytes, a ordem de leitura varia entre os formatos *Little Endian* e *Big Endian*. Na primeira variação, o primeiro byte lido é o menos significativo (o byte de maior valor numérico, por exemplo, o byte 1 é lido antes do byte 0), e no segundo caso, o primeiro byte lido é o mais significativo (o byte de menor valor numérico é lido primeiro). No caso de parâmetros que ocupam menos de 1 byte, a definição entre *Little Endian* e *Big Endian* não é necessária, pois o conceito de ordem de leitura dos bytes não se aplica a dados que já estão contidos em um único byte.

Após a seleção dos bits corretos que representam a variação de controle e a definição da reta de resolução por meio das equações apresentadas, é fundamental validar o processo. Essa validação ocorre por meio de testes com valores intermediários conhecidos, permitindo a comparação entre o valor estimado pela reta e o valor real da grandeza física. Como em medições simples não é possível obter uma estimativa exata, a validação pode ser realizada por aproximação.

Por exemplo, a medição da porcentagem de pressão do pé no pedal do acelerador é aproximada, exceto nos extremos de 0% e 100%, onde os limites físicos garantem medições precisas. No entanto, é possível verificar a coerência dos valores intermediários com o comportamento esperado: à medida que o pedal é pressionado, o valor numérico deve aumentar de forma progressiva, e, conforme a pressão diminui

gradativamente, o valor numérico também deve reduzir proporcionalmente. Além disso, é fundamental observar se as respostas permanecem dentro dos limites propostos, isto é, não deverá existir alguma medição no qual o valor extrapole os valores de 0% e 100%, caso aconteça, a seleção dos bits precisa ser ajustada ou até mesmo o byte, em alguns casos.

Para os parâmetros digitais, a validação é mais direta, pois a variação nos bits ocorre de maneira discreta e previsível, sem valores intermediários que exijam testes de limites e coerência. Nesse caso, os testes realizados se resumem a leituras em diferentes dias, verificando se o bit continua condizendo com a parametrização proposta.

4.4 Processo de parametrização

Tanto da CAN principal como da CAN de baterias seguem o formato *Little Endian* de leitura. Esse padrão é crucial para a interpretação correta dos dados e para garantir a precisão na leitura das mensagens CAN.

A confirmação de que o sistema segue o formato *Little Endian* na CAN principal foi realizada durante a análise dos dados de posição do volante, uma grandeza que ocupa dois bytes. Ao interpretar os valores obtidos, ficou evidente que a variação fazia sentido apenas na leitura *Little Endian*. Um exemplo de mensagem recebida para a posição do volante é apresentado no Quadro 4, enquanto o processo de decodificação, que contém algumas novas medições de teste, é detalhado na Quadro 5 em sequência.

Quadro 4 – Mensagem recebida e destaque de informações relevantes

Identificador	0x11F							
	Byte 0	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7
Mensagem	12	00	00	FF	A9	00	00	00
Resolução	Byte posição: 0, Bit posição: 0, Quantidade de bits: 16							
Byte 0 e Byte 1	12	00	00	FF	A9	00	00	00
Big Endian	1200							
Little Endian	0012							

Fonte: Autoria própria (2025).

Quadro 5 – Processo de decodificação da posição do volante

Status do Volante	Byte 0	Byte 1	Big Endian	Little Endian
Posição inicial + leve inclinação à esquerda	30	00	12288	48
1 giro completo para a esquerda	DD	0D	56589	3549
Máxima posição à esquerda	C1	16	49430	5825
Posição inicial + leve inclinação à direita	CE	FF	52991	65486
1 giro completo para a direita	9E	F1	40689	61854
Máxima posição à direita	E8	E8	59624	59624

Fonte: Autoria própria (2025).

O Quadro 6 apresenta a comparação entre as escalas obtidas de forma prática, destacando as variações nos valores decimais lidos e convertidos nas duas formas de leitura durante os dois sentidos de giro do volante. O volante foi girado em ambas as direções de maneira consistente, e as alterações nos valores decimais dos bytes foram registradas em relação ao valor inicial. Como o movimento de giro foi realizado de forma uniforme, espera-se que as variações entre os valores de origem e destino sejam semelhantes para ambos os sentidos de rotação. Neste sentido, a variação em *Little Endian* foi analisada como coerente. A parametrização utilizada para o critério de posição do volante será fornecida na seção 5.1.

Quadro 6 – Identificação das escalas práticas para posição do volante

Escala prática para giro máximo (travamento de volante)			
Little Endian	Variação	Big Endian	Variação
Direita			
0 a 5825	5825	0 a 49430	49430
Esquerda			
65535 a 59624	5911	65535 a 59624	5911

Fonte: Autoria própria (2025).

Na CAN de baterias, a determinação da característica *Little Endian* foi baseada na leitura valor da tensão unitária dos módulos, que ocupa 2 bytes no campo de dados. A codificação dos valores para os dois modos de leitura é ilustrada na Quadro 7.

Quadro 7 – Identificação de leitura *Little Endian* na CAN de baterias

Exemplo de Mensagem obtida no ID: 1801CC03								
	Byte 0	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7
Mensagem	05	D7	0E	AA	08	CE	0E	01
Seleção	Byte 1 e Byte 2							
Little Endian	0ED7				Interpretação decimal	3799		
Big Endian	D70E					55054		

Fonte: Autoria própria (2025).

Essa parametrização específica foi realizada utilizando exclusivamente os logs gerados pelo *sniffer* comercial, juntamente com as informações presentes no manual do fabricante *Beifang*. O processo de análise e interpretação desses logs será detalhado na próxima seção.

4.5 Análise de logs

Foram capturados logs de duas redes CAN para mapeamento dos IDs presentes. O log da rede CAN de baterias, foi registrado no dia 26 de setembro de 2024, às 18:14:16. Com a captura, foram identificados 60 Identificadores diferentes.

Já o log da rede CAN principal, foi obtido no mesmo dia, às 18:19:09, também com 50.000 mensagens, e resultou na identificação de 49 Identificadores distintos.

Pelo protótipo, foram interceptados também 49 IDs da rede CAN principal, no dia 18 de outubro de 2024, aproximadamente 18:30, com 10.893 mensagens capturadas. Vale ressaltar que o tempo de escuta da rede não foi igual ao do log anterior, pois os tempos não foram cronometrados e não havia uma quantidade mínima de mensagens programadas a serem extraídos nem no *firmware* e nem na aplicação em *Python*. Em relação à rede CAN de baterias, foram capturadas 7.425 mensagens e apenas 40 identificadores, o que sugere um refinamento nas configurações de *firmware* para a atuação do protótipo como *sniffer* na rede de baterias.

Com as tabelas obtidas, é possível realizar análises detalhadas com dados filtrados e selecionados. Ao escolher identificadores para análise isolada, como exemplo, é possível observar as mensagens periódicas enviadas pela ECU e analisar mudanças ao longo do tempo, tanto com variação de controle quanto sem. Em diversos IDs, foi possível identificar bytes contadores dentro do campo de dados dessas mensagens. Esses bytes alternam simetricamente entre as mensagens enviadas, mantendo o padrão de variação independentemente das perturbações na rede.

Esses contadores, que têm a capacidade de incrementar ou decrementar, são extremamente úteis para verificar a perda de mensagens. Variações padronizadas no valor do contador entre as mensagens indicam uma transmissão contínua, o que ajuda na validação da integridade da comunicação.

A análise de perdas foi implementada utilizando esses contadores para avaliar a recepção do protótipo. O resultado, na rede CAN principal, indicou que não houve perdas nos pacotes, sugerindo que as mensagens com contadores estavam sendo recebidas de forma contínua, sem falhas na leitura do protótipo. Em contraste, na rede CAN das baterias, não foram identificados bytes semelhantes aos contadores, o que pode sugerir uma diferença na forma como as mensagens são gerenciadas entre as duas redes. Isso implica que a avaliação de perda de pacotes no protótipo por esse método não é viável, sendo possível apenas realizar essa avaliação por meio da falha na recepção de determinados IDs.

Na rede CAN das baterias, embora os IDs predominantes sigam o padrão CAN 2.0B, foram identificados dois IDs CAN 2.0A, sendo que um deles também circula na rede CAN principal. Tanto no log pelo *sniffer* comercial quanto no *sniffer* próprio o identificador 0x2B6 foi encontrado na transmissão das duas redes, com o mesmo valor no campo de dados.

Das mensagens que seguem o padrão CAN 2.0B, na rede CAN de baterias, todos identificadores são definidos pelo modelo 180*x*CC0*y*, onde x varia entre os valores: 1, 2, 3, 4, 7, 8, 9, A, B (9 possibilidades totais de x). Quando x assume valores como: 1, 2, 3, 4, 8, 9, B (7 possibilidades de x) y adota valores de 0 a 7 (8 possibilidades de y) enquanto quando x valores de 7 e A (2 possibilidades de x), y é 0 (1 possibilidade de y). Dessa forma, há 58 possibilidades criadas dentro do padrão apresentado. O Quadro 8 apresenta essas informações de forma visual. Este nível de padronização reforça a ideia de que o sistema de baterias foi projetado para ser um sistema facilmente escalável e adaptável para outras aplicações.

Quadro 8 – Identificadores do modelo 180xCC0y da CAN de baterias

Modelo 180xCC0y				
Valores em x	Total de possibilidades em x	Valores em y	Total de possibilidades em x	Possibilidades de combinação x,y
1, 2, 3, 4, 8, 9, B	7	0, 1, 2, 3, 4, 5, 6, 7	8	56
7, A	2	0	1	2
Total de possibilidades				58

Fonte: Autoria própria (2025).

Como os parâmetros relacionados à bateria não podem ser modificados instantaneamente por meio de perturbações no controle, a identificação desses parâmetros não segue o mesmo processo descrito na seção 3.3.3. Em vez disso, a análise dos logs é utilizada como a principal ferramenta para a decodificação dos parâmetros. Para esse fim, os dados exibidos no painel do veículo servem como referências físicas reais, especificamente o *State of Charge* (SoC, em português estado da carga), que representa o nível de carga da bateria em relação à sua capacidade total, e a autonomia (em quilômetros), que estima a distância que o veículo pode percorrer com a quantidade de energia restante na bateria.

A especificação técnica das baterias, presente no manual da *Beifang*, também foi utilizada como parâmetro referencial de grandeza física. Os dados incluem a quantidade de células, que totalizam 112 unidades, e a tensão unitária em cada célula, que é de 3,6V. As parametrizações realizadas e outras informações relacionadas serão apresentadas na seção 3.4.2, que trata das parametrizações obtidas na rede CAN de baterias.

Em resumo, o processo de decodificação é essencial para atribuir significados aos bytes e bits extraídos, seja em tempo real ou por meio de análise de logs. A correta parametrização resultante dessa etapa garante a definição precisa dos valores e a compreensão das relações entre os parâmetros, permitindo, assim, a leitura e estimativa de valores intermediários no mundo físico, eliminando a necessidade de comparações constantes com os valores binários.

5 PARAMETRIZAÇÕES E APLICABILIDADE DOS DADOS

Foram realizadas parametrizações tanto na rede CAN principal quanto na rede CAN das baterias. Os resultados serão apresentados e discutidos a seguir. Na rede CAN principal, as afirmações são consideradas conclusivas. Na rede CAN das baterias, no entanto, as análises são preliminares, e sugere-se a realização de um estudo adicional para aprofundar a investigação. Também é discutido brevemente a aplicabilidade dos dados extraídos da rede CAN principal, de forma a contribuir com a continuidade dos estudos relacionados.

5.1 CAN principal

O resumo das parametrizações realizadas na rede CAN principal é apresentado a seguir, no Quadro 9. Para cada parâmetro, são detalhados o "identificador" onde os dados foram localizados, a "posição do byte", a "posição do bit", a "quantidade de bits" e a reta parametrizada (“*y*”), que descreve a resolução, ou seja, a transformação dos valores “*x*” lidos no barramento em valores “*y*” com as unidades de medida reais.

Quadro 9 – Resumo das parametrizações realizadas

Parâmetro	Identificador	Byte posição	Bit posição	Quantidade de bits	Reta parametrizada e intervalos válidos	Unidade de medida
Posição do volante	0x11F	0	0	16	y = 0,1*x, x ∈ [0,6000], x ∈ Z y = - 0,0058*x, x ∈ [5900,65535], x ∈ Z	° (positivo à esquerda e negativo à direita), 1 volta = 360°
Botão Start	0x122	3	0	1	y = x, x ∈ {0,1}	acionado (1), não acionado (0)
Pedal do freio	0x12D	1	0	1	y = x, x ∈ {0,1}	acionado (1), não acionado (0)
Setas	0x133	0	4	2	y = x, x ∈ {0,1, 2}	setas desligadas (0), seta esquerda ligada (1), seta direita ligada (2)
Faróis simples	0x133	3	3	2	y = x, x ∈ {0,1, 2}	faróis desligados (0), farolete ligado (1), farol baixo ligado (2)
Farol alto	0x133	0	3	1	y = x, x ∈ {0,1}	ligado (1), desligado (0)
Parabrisa status	0x133	6	2	1	y = x, x ∈ {0,1}	ligado (1), desligado (0)
Parabrisa acionamento	0x133	6	3	1	y = x, x ∈ {0,1}	acionado (1), não acionado (0)
Pedal do acelerador	0x242	3	0	8	y = x, x ∈ [0,99], x ∈ Z	% pressionado
Modos de condução	0x342	2	4	2	y = x, x ∈ {1,2, 3}	modo Eco (0), modo Sport (1), modo Gelo (2)
Velocidade	0x39E	1	0	8	y = x, x ∈ [0,255], x ∈ Z	km/h

Fonte: Autoria própria (2025).

As retas foram criadas com base nas equações 1 e 2, descritas na seção 3.3.3. As resoluções alcançadas serão apresentadas em ordem de identificador prioritário, ou seja, a partir dos identificadores de menor valor numérico, sequência utilizada também no Quadro 9. Inicia-se então a análise com o identificador 0x11F, no Quadro 10, a seguir.

Quadro 10 – Obtenção e parametrização de dados do identificador 0x11F

Identificador 0x11F								
	Byte 0	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7
Mensagem	7F	FF	0	FF	91			
Velocidade	Byte posição: 0, Bit posição: 0, Quantidade de bits: 16							
Byte 0	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
Binário	0	1	1	1	1	1	1	1
Byte 1	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Binário	1	1	1	1	1	1	1	1
Interpretação	Para a esquerda:		Binário (Byte 1)	Binário (Byte 0)	Hexadecimal	Decimal		
	Valor mínimo:		0000 0000	0000 0000	0x00 0x00	0	0°, 0 voltas completas, 0% de giro de volante permitido	
	Medida intermediária:		0000 1101	1101 1101	0X0D 0xDD	3549	360°, 1 voltas completas, 60,95% de giro de volante permitido	
	Valor máximo (travamento de volante):		0001 0110	1100 0001	0x16 0xC1	5825	591°, 1,64 voltas completas, 100% de giro de volante permitido	
	Para a direita:		Binário (Byte 1)	Binário (Byte 0)	Hexadecimal	Decimal		
	Valor mínimo:		1111 1111	1111 1111	0xFF 0xFF	65535	0°, 0 voltas completas, 0% de giro de volante permitido	
	Medida intermediária:		1111 0001	1001 1110	0XF1 0x9E	61854	-360°, -1 voltas completas, -62,30% de giro de volante permitido	
	Valor máximo (travamento de volante):		1110 1000	1110 1000	0xE8 0xE8	59624	-578.5°, -1,61 voltas completas, -100% de giro de volante permitido	

Fonte: Autoria própria (2025).

Foram realizadas duas parametrizações diferentes para a posição do volante. A primeira utiliza o primeiro giro completo do volante como referência, enquanto a segunda considera o travamento do volante como referência máxima. Em ambas, a posição inicial do volante (com as rodas alinhadas ao eixo do veículo) é tomada como ponto de partida.

A primeira parametrização atribui 360° ao primeiro giro completo e 0° à posição inicial, o valor é considerado positivo se o giro é realizado para a esquerda e negativo caso seja realizado para a direita. Esse tipo de parametrização depende do

observador para definir a referência exata de um giro completo e aplicá-la aos dois sentidos de rotação, o que pode resultar em erros de calibragem ou imprecisão na determinação exata da referência em ambos os sentidos. A medição de valor teste intermediário (no qual o travamento do volante foi a referência), quando aplicada nos dois sentidos de giro, apresentaram resultados semelhantes: 591° e –578,5°.

A segunda parametrização atribui a referência de 100% ao travamento do volante em ambos os sentidos de giro, e 0% à posição inicial. Nesse caso, a medição de valor intermediário para teste foi a posição de 1 volta completa em ambos os sentidos de giro. Os valores encontrados foram 60,95% e -62,30%. A escala se mostrou mais uma vez satisfatória, tendo em vista que o valor intermediário é influenciado pela precisão do observador ao definir a referência.

Em condições normais de condução, espera-se que o ponto inicial do volante oscile entre os valores hexadecimais 0x00 0x00 e 0xFF 0xFF, com variações próximas a esses limites. Isso ocorre porque, em um dos sentidos de giro, o valor decrementa a partir de 0xFF 0xFF, enquanto na direção oposta, o valor incrementa a partir de 0x00 0x00. Isso faz com que intervalos e retas diferentes sejam traçadas.

A reta alcançada ela primeira parametrização é apresentada na Equação 3, sendo que a reta 3.1 é referente ao giro para a esquerda (sentido anti-horário) e a reta 3.2 é referente ao giro para a direita (sentido horário). As duas retas tiveram como base a Equação 1 (Seção 3.3.3). Os limites foram extrapolados para considerar ajustes de calibração. Fora dos limites colocados, no entanto, a reta é considerada inválida.

$$y(x) = 0{,}1 \cdot x, \quad x \in [0{,}6000] \tag{3.1}$$

$$y(x) = -0{,}0058 \cdot x, \quad x \in [59000{,}65535] \tag{3.2}$$

O próximo identificador analisado é 0x122, apresentado no Quadro 11.

Quadro 11 – Obtenção e parametrização de dados do identificador 0x122

Identificador 0x122								
	Byte 0	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7
Mensagem	00	C0	0	80	FF	BF	FF	2F
Botão de start	Byte posição: 3, Bit posição: 0, Quantidade de bits: 1							
Byte 3	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Binário	1	0	0	0	0	0	0	0
Interpretação							Start acionado	1
							Start não acionado	0

Fonte: Autoria própria (2025).

A parametrização encontrada no identificador 0x122 faz referência ao acionamento do botão *Start*. O parâmetro é binário, portanto, só assume dois valores, que seriam o estado acionado (1) e não acionado (0). As equações, portanto, não necessitam ser descritas, sendo o modelo o mesmo da Equação 2.1 e os limites os mesmos da Equação 2.2, ambas equações apresentadas na seção 3.3.3.

Na sequência, o identificador 0x12D está apresentado no Quadro 12.

Quadro 12 – Obtenção e parametrização de dados do identificador 0x12D

Identificador 0x12D								
	Byte 0	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7
Mensagem	40	48	00	10	0C	19	82	92
Pedal de freio	Byte posição: 1, Bit posição: 0, Quantidade de bits: 1							
Byte 1	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Binário	0	1	0	0	1	0	0	0
Interpretação							Pedal de freio acionado	1
							Pedal de freio não acionado	0

Fonte: Autoria própria (2025).

A parametrização encontrada no identificador 0x12D faz referência ao acionamento do pedal de freio. O parâmetro é binário, portanto, só assume dois valores, que seriam o estado acionado (1) e não acionado (0). As equações seguem as mesmas diretrizes da Equação 2.1 e os limites os mesmos da Equação 2.2, apresentadas na seção 3.3.3.

Na sequência, o identificador 0x133 está apresentado no Quadro 13.

Quadro 13 – Obtenção e parametrização de dados do identificador 0x133

Identificador 0x133								
	Byte 0	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7
Mensagem	06	31	31	E0	BF	02	A1	BF
Setas	Byte posição: 0, Bit posição: 4, Quantidade de bits: 2							
Mensagem	06	31	31	E0	BF	02	A1	BF
Byte 0	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Binário	0	0	0	0	0	1	1	0
Interpretação			0	0	Setas desligadas			
			0	1	Seta para a esquerda			
			1	0	Seta para a direita			
Faróis	Byte posição: 3, Bit posição: 3, Quantidade de bits: 2							
Mensagem	06	31	31	E0	BF	02	A1	BF
Byte 3	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Binário	1	1	1	0	0	0	0	0
Interpretação			1	0	Farol ligado			
			0	0	Farol e farolete desligados			
			0	1	Farolete ligado			
Farol alto	Byte posição: 0, Bit posição: 3, Quantidade de bits: 1							
Mensagem	06	31	31	E0	BF	02	A1	BF
Byte 0	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Binário	0	0	0	0	0	1	1	0
Interpretação					1	Farol alto ligado		
					0	Farol alto desligado		
Parabrisa status	Byte posição: 6, Bit posição: 2, Quantidade de bits: 1							
Mensagem	06	31	31	E0	BF	02	A1	BF
Byte 6	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Binário	1	0	1	0	0	0	0	1
Interpretação						1	Parabrisa ligado	
						0	Parabrisa desligado	
Parabrisa acionamento	Byte posição: 6, Bit posição: 3, Quantidade de bits: 1							
Mensagem	06	31	31	E0	BF	02	A1	BF
Byte 6	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Binário	1	0	1	0	0	0	0	1
Interpretação					1	Parabrisa acionado		
					0	Parabrisa não acionado		

Fonte: Autoria própria (2025).

Várias parametrizações foram realizadas com base no identificador 0x133, elas fazem referência às setas, aos faróis, farol alto e à operação do limpador de para-brisa. Alguns valores binários podem ser agrupados, como por exemplo a operação das setas, nesse sentido, a equação base seria a Equação 2.1, com limites no modelo da Equação 2.3 (seção 3.3.3). Os limites das operações das setas são os apresentados pela equação 4.1, a seguir. Semelhantemente, a operação dos faróis segue a mesma especificação.

$$x \in \{0,1,2\} \tag{4.1}$$

Para os parâmetros farol alto, status que indica o limpador de para-brisa e acionamento dele (todos binários), as equações seguem as mesmas diretrizes da Equação 2.1 e os limites os mesmos da Equação 2.2, apresentadas na seção 3.3.3.

O próximo identificador a ser analisado é o 0x242, apresentado no Quadro 14.

Quadro 14 – Obtenção e parametrização de dados do identificador 0x242

Identificador 0x242								
	Byte 0	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7
Mensagem	30	75	FE	00	0C	A1	0F	A0
Pedal do acelerador	Byte posição: 3, Bit posição: 0, Quantidade de bits: 8							
Byte 3	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Binário	0	0	0	0	0	0	0	0
Interpretação			Binário	Hexadecimal	Decimal			
	Valor mínimo:		0000 0000	0x00	0	0% pressionado (obtido)		
	Exemplo intermediário:		0011 1100	0X3C	60	60,606% pressionado (estimado)		
	Valor máximo:		0110 0011	0x63	99	100% pressionado (obtido)		

Fonte: Autoria própria (2025).

Inicialmente, o valor de 100% seria utilizado como o limite máximo de operação do pedal do acelerador. No entanto, ao analisar a variação dos dados, foi observado que os valores de medição para o pedal são representados por incrementos inteiros que variam de 0 a 99. Isso resultaria em todas as medições fracionadas caso o valor de 100% fosse adotado para o inteiro 99. Ao adotar 99% como limite máximo, no entanto, todas as medições tornam-se também inteiras. Dessa forma, a equação modelada é apresentada em 5.1, com limites de operação em 5.2

$$y(x) = x \tag{5.1}$$

$$x \in [0{,}99],\ x \in \mathbb{Z} \tag{5.2}$$

Em sequência, é apresentado o identificador 0x342, apresentado no Quadro 15.

Quadro 15 – Obtenção e parametrização de dados do identificador 0x342

Identificador 0x342								
	Byte 0	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7
Mensagem	00	00	EF	B1	F6	FF	F7	73
Modo de condução	Byte posição: 2, Bit posição: 4, Quantidade de bits: 2							
Byte 2	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Binário	1	1	1	0	1	1	1	1
Interpretação				0	1	Eco		
				1	0	Sport		
				1	1	Gelo		

Fonte: Autoria própria (2025).

Os modos de condução seguem padrões binários, e foram parametrizados em conjunto, conforme Equação 2.1 e limites descritos na Equação 2.3 (seção 3.3.3). Os limites são apresentados a seguir, na Equação 6.1

$$x \in \{1, 2, 3\} \tag{6.1}$$

O identificador 0x39E está apresentado no Quadro 16, a seguir.

Quadro 16 – Obtenção e parametrização de dados do identificador 0x39E

Identificador 0x39E								
	Byte 0	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7
Mensagem	01	00	00	F0	00	F0	FF	FF
Velocidade	Byte posição: 1, Bit posição: 0, Quantidade de bits: 8							
Byte 1	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Binário	0	0	0	0	0	0	0	0
Interpretação			Binário	Hexadecimal	Decimal			
	Valor mínimo:		0000 0000	0x00	0	0 km/h (obtido)		
	Exemplo intermediário:		0011 1100	0X3C	60	60 km/h (obtido)		
	Valor máximo:		1111 1111	0xFF	255	255 km/h (estimado)		

Fonte: Autoria própria (2025).

A parametrização encontrada no identificador 0x39E faz referência à velocidade. O parâmetro é linear. As medidas tidas como referência partiram da visualização do velocímetro. Nesse caso, várias medições foram feitas, de modo que o valor máximo não foi obtido, mas estimado. Foi considerado que a operação pode ocupar 1 byte inteiro, portanto, o limite foi fixado em 255. A equação é apresentada na Equação 7.1 e o limite de operação na Equação 7.2

$$y(x) = x \tag{7.1}$$

$$x \in [0{,}255],\ x \in \mathbb{Z} \tag{7.2}$$

Encerradas as parametrizações realizadas no presente trabalho referentes à CAN principal, a aplicabilidade é, portanto, analisada brevemente na próxima subseção.

5.1.1 Aplicabilidade para os dados extraídos na CAN principal

Os dados coletados pela rede CAN são predominantemente quantitativos, como medições de velocidade, pressão do pedal do acelerador e rotação do volante. Contudo, por si só, esses dados são úteis apenas para o controle interno das ECUs e para a exibição no painel do veículo. Para transformar esses dados em informações valiosas para outros usuários, é essencial aplicar modelos analíticos que convertam esses valores brutos em métricas relevantes, capazes de ser aplicadas a contextos específicos.

Um exemplo claro disso é o status do limpador de para-brisa. Quando analisado isoladamente, esse dado fornece apenas uma informação de *status* de funcionamento para o condutor. No entanto, ele pode servir de contexto para outros dados, indicando a presença de chuva e alterando limites de segurança relacionados à velocidade e a variações bruscas na posição do volante durante a condução.

Nesse sentido, diversas aplicações podem ser desenvolvidas. Por exemplo, a criação de uma ECU que não apenas lê os dados do barramento, mas também é responsável por relacioná-los para fornecer alertas para os motoristas de situações de comportamento imprudente. Nesse contexto, outros dados podem ser incluídos, como variações acentuadas na pressão do pedal do acelerador, excesso de frenagens em alta velocidade ou alterações no controle do volante que indiquem curvas e trocas de faixas sem o uso prévio das setas de sinalização.

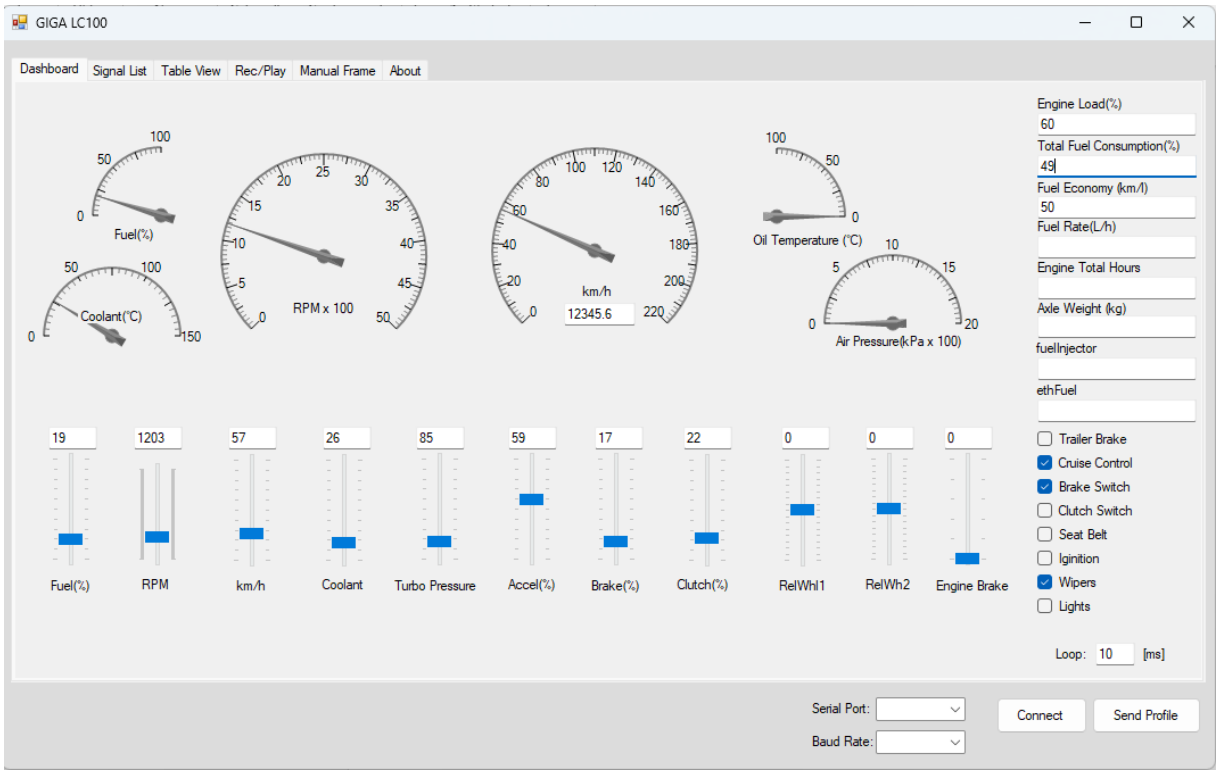
Análises que avaliam o comportamento de condução do motorista são extremamente valiosas em aplicações de telemetria pois um dos grandes interesses do gestor de frotas é o controle da operação eficiente, econômica e prudente dos motoristas com os veículos utilizados no trabalho. Isso porque esses dados ajudam na otimização de custos operacionais, como consumo de combustível e manutenção dos veículos, além de contribuir para a segurança da frota.

Tendo isso em vista, uma oportunidade de estudo é a aplicação de envio de dados da rede CAN para uma central de monitoramento em tempo real de forma *wireless*. Utilizando tecnologias como Wi-Fi, LoRa, NB-IoT ou até mesmo 4G/5G, os dados coletados da rede CAN podem ser transmitidos sem a necessidade de conexões físicas, permitindo o monitoramento remoto dos veículos e do comportamento dos motoristas em tempo real.

Além disso, poderia ser realizado o desenvolvimento de uma ECU que atue como nó transmissor rede CAN, não apenas como receptor, ampliando as experiências práticas e a interação com o sistema CAN presente nas bancadas didáticas. Essa abordagem permitiria a simulação de um ambiente mais completo e dinâmico, onde a ECU não só lê e interpreta os dados das mensagens CAN, mas também envia informações de volta para o barramento. Esse tipo de desenvolvimento seria útil para explorar diferentes cenários de comunicação no barramento CAN e proporcionar uma visão mais profunda da integração de sistemas embarcados no contexto automotivo.

A criação de simuladores CAN também é uma proposta aplicável e extremamente útil. A Figura 26 ilustra uma tela de simulador desenvolvida pela autora durante seu trabalho profissional, este simulador tinha como função testar novas funcionalidades em produtos de telemetria baseados na rede CAN. O simulador gera uma rede física a partir de dados simulados na interface gráfica e parâmetros conhecidos de determinados veículos (já mapeados) que poderiam ser escolhidos a partir de uma biblioteca interna. O *hardware* utilizado no simulador foi o mesmo proposto neste projeto, e com ele foi possível gerar a CAN física simulada. O sistema permite testar e validar novas funções nos produtos com interface CAN sem a necessidade de validação em campo, acelerando o desenvolvimento e evitando atrasos.

Figura 26 – Software simulador próprio



Fonte: Autoria própria (2025).

Outras interfaces também podem ser implementadas, como interfaces de jogos ou simulações interativas, que proporcionam uma forma mais dinâmica e envolvente de visualizar e analisar dados. Essas interfaces não só tornam o processo de aprendizado mais interessante, mas também facilitam a interação com o sistema e a compreensão dos parâmetros. A Figura 27, que traz a interface do *Euro Truck Simulator* pode ilustrar bem essa proposta.

Figura 27 – Interface do jogo Euro Truck Simulator

Fonte: *Euro Tuck Simulator* (2025).

A parametrização dos dados na rede CAN principal abre caminho para diversas aplicabilidades no ensino, oferecendo aos alunos uma ampla gama de oportunidades para aplicações e estudos. Ao estruturar os dados de acordo com as necessidades do usuário final, é possível combinar parâmetros e informações, gerando insights valiosos que ampliam a compreensão e a utilização do sistema.

5.2 CAN de baterias

Inicialmente, a análise foi focada na coleta de informações do identificador 0x2B6, o único presente tanto na rede de baterias quanto na rede principal. Isso se deve à suposição de que, como o SoC e a autonomia são exibidos ao usuário no painel, mas resultam de estimativas do controle de baterias, o identificador que transita entre as redes poderia conter essas informações. Para testar essa hipótese, foram retiradas as seguintes medidas, apresentadas no Quadro 17.

Quadro 17 – Logs do ID 0x2B6 em busca de SoC e autonomia

SoC (%)	Autonomia (km)	ID	Byte 0	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7
71	299	2B6	18	09	1B	05	0A	2C	F4	FF
66	280	2B6	11	01	01	08	11	2F	F7	FF
93	395	2B6	19	02	06	05	22	39	F2	FF

Fonte: Autoria própria (2025).

Ao analisar a relação entre o SoC e a autonomia, observou-se uma correlação linear entre ambos. No entanto, os valores exibidos no painel são truncados ou aproximados. Essas observações sugerem que os valores podem ser resumidos em um único parâmetro na rede CAN, sem excluir a possibilidade de ambos estarem presentes. No entanto, é possível encontrar um valor e estimar o outro com base na relação linear entre SoC e autonomia.

Como as variações entre as medições são conhecidas, a tentativa foi identificar decimais dentro das medições que apresentassem a mesma razão de variação dos valores de SoC ou autonomia. No entanto, essas razões não foram encontradas. Diante disso, surgem duas possibilidades: ou os valores não são repassados por esse identificador específico, ou há a possibilidade de que outra grandeza seja, de fato, a base dos cálculos.

Para comparar os dados com outras grandezas, como energia ou potência, é essencial compreender as relações entre essas grandezas, a fim de identificar relações que possam explicar as variações numéricas entre os bytes encontrados. Como o estudo detalhado do comportamento das baterias está fora do escopo deste trabalho, sugere-se que a rede CAN das baterias seja alvo de uma investigação específica. Esse estudo aprofundado possibilitaria a estimação de curvas de análise mais precisas, além de aprimorar a interpretação dos parâmetros decodificados na rede CAN das baterias.

Em uma etapa posterior, buscou-se identificar a tensão das 112 células. Dois modelos de identificadores se destacaram como potenciais candidatos para representar esses valores: 1801CC0x e 1803CC0x, com x variando de 0 a 7. A escolha desses identificadores baseou-se no fato de que, ao combinar os bytes 1 e 2 em uma leitura *Little Endian*, obteve-se um valor próximo à tensão esperada das baterias pelo fabricante. O Quadro 18, a seguir, mostra a variação de 0 a 7 para o

identificador do modelo 1801CC0x, a fim de verificar se todos eles apresentam valores semelhantes quando aplicada a decodificação proposta.

Quadro 18 – Estimativas de tensão no identificador 1801CC00

Identificadores	Mínimo (mV)	Máximo (mV)	Média (mV)	Variação (mV)
1801CC00	3796	3797	3796,5	0,5
1801CC01	3799	3800	3799,5	0,5
1801CC02	3798	3798	3798	0
1801CC03	3798	3799	3798,5	0,5
1801CC04	3800	3801	3800,5	0,5
1801CC05	3798	3799	3798,5	0,5
1801CC06	3800	3800	3800	0
1801CC07	3798	3799	3798,5	0,5

Fonte: Autoria própria (2025).

Como os identificadores 1801CC0x e 1803CC0x, com x variando de 0 a 7, não abrangem as 112 células da bateria, foi proposta a hipótese de que um byte pudesse atuar como índice, modificando os valores dentro do campo de dados. Dessa forma, o mesmo identificador poderia fornecer os valores de mais células ao alterar o índice. Ao analisar o byte 3 das duas versões, observou-se que, na variação 1801CC0x, ele apresentava 16 possibilidades (variando de A0 a AF), o que não ocorria na versão 1803CC0x. Assim, as 8 combinações de 1801CC0x, juntamente com as 16 alternativas do byte 3, poderiam representar as tensões das células da bateria. No entanto, isso resultaria em 128 resultados, número que excede a quantidade de células da bateria. Caso algumas das combinações de identificadores fossem descartadas, considerando-se 7 alternativas no lugar de 8, seria possível que as leituras se alinhassem de forma mais precisa com as 112 células.

Também seria necessário medir as tensões nas baterias para validação, pois a referência proposta pelo fabricante é de 3700 mV, enquanto os valores sugeridos através da leitura CAN estão mais próximos de 3800 mV. Isso reforça a necessidade de um estudo dirigido com foco na rede CAN das baterias.

É importante ressaltar que a leitura *Little Endian* proposta pode não ser o método técnico adequado para a leitura dessa rede. Pois, caso em um estudo subsequente, a leitura de tensão conforme sugerido neste trabalho seja invalidada, os valores estimados em cada byte não devem ser considerados. Dessa forma, este

estudo serve como um ponto inicial, limitando-se a parametrizações conclusivas na rede CAN principal, e abre espaço para futuras análises mais aprofundadas na rede CAN de baterias do veículo elétrico BYD Qin.

6 CONSIDERAÇÕES FINAIS

O Sistema de Treinamento com Ligação de Controles Separados para o Veículo Elétrico BYD Qin oferece inúmeras possibilidades de estudos baseados na rede CAN, com grande potencial na área de ensino. A utilização deste sistema permite a exploração da comunicação interna do veículo, proporcionando um ambiente prático para a compreensão dos dados que transitam pela rede CAN.

Mesmo com o amplo estudo realizado na rede CAN principal do veículo, a continuidade do trabalho é incentivada para a busca de mais parâmetros e a exploração de aspectos adicionais que possam aprimorar a compreensão e análise do sistema. Na rede CAN das baterias, como mencionado, as análises são preliminares, e um estudo mais aprofundado é recomendado para investigar novos parâmetros e características que possam contribuir para o avanço do projeto.

Uma das observações relevantes realizadas ao longo do trabalho foi que parâmetros semelhantes, como seta direita, seta esquerda, farol alto e farol baixo, podem estar agrupados sob o mesmo ID, tornando a busca por novos parâmetros mais eficiente ao permitir iniciar a análise com identificadores que compartilham características semelhantes aos parâmetros já mapeados, otimizando o processo de decodificação e tornando a análise mais ágil, precisa e eficaz. Além disso, uma estratégia encontrada para verificar a perda de pacotes foi a análise dos bytes contadores presentes nos identificadores; embora nem todos os identificadores possuam esses bytes, a análise foi eficaz sempre que estavam disponíveis, permitindo avaliar a recepção dos dados no protótipo desenvolvido.

A integração do protótipo como nó na rede CAN principal foi realizada com sucesso, proporcionando uma compreensão sólida da estrutura da rede e a captura das informações relevantes. No entanto, na rede CAN das baterias, o protótipo não demonstrou a mesma eficiência, o que sugere a necessidade de otimizações adicionais no código para melhorar a leitura dos dados e a performance global na rede de baterias.

De maneira geral, o protótipo capaz de atuar como nó receptor de dados se mostrou satisfatório, demonstrando a viabilidade da implementação dessa

funcionalidade. O protótipo foi capaz de se integrar à rede, capturar informações relevantes e fornecer uma base sólida para futuras pesquisas e melhorias no sistema.

Os dados de interesse foram devidamente identificados e extraídos, permitindo uma análise detalhada que contribuiu para o desenvolvimento do trabalho. As parametrizações realizadas mostraram-se úteis para novas aplicações além do escopo inicial, que se restringia à comunicação interna veicular. As possibilidades de expansão do estudo, com a utilização dos dados apresentados, foram avaliadas como altamente relevantes e enriquecedoras, especialmente para fins educacionais, oferecendo novas oportunidades para o aprendizado e a aplicação prática de conceitos relacionados à rede CAN em veículos elétricos.

Entre as diversas possibilidades de expansão do estudo sobre a rede CAN, utilizando o Sistema de Treinamento com Ligação de Controles Separados para o Veículo Elétrico BYD Qin, são sugeridos os seguintes trabalhos futuros:

- Remodelamento no *firmware* do *sniffer*, visando aprimorar a capacidade de detecção de mensagens na CAN de baterias;
- Construção de uma interface para facilitar a visualização dos dados obtidos;
- Desenvolvimento de uma ECU atuando como nó transmissor na rede CAN, para permitir outras possibilidades de estudo da rede;
- Implementação de uma solução para transmissão *wireless* dos dados da rede CAN;
- Criação de um simulador CAN.

REFERÊNCIAS

ATKINSON, Christen. **Simplify automotive interface design with fully interoperable and EMC-compliant 3.3V CAN transceivers.** Texas Instruments, 2021. Disponível em: https://www.ti.com/lit/ta/ssztd46/ssztd46.pdf?ts=1739743856897&ref_url=https%253A%252F%252Fwww.ti.com%252Finterface%252Fcan-transceivers%252Foverview.html. Acesso em: 16 fev. 2025.

BEIFANG. New energy series: NEV SEPARATE CONTROL LINKAGE TRAINING SYSTEM BYD QUIN EV (FIVE-PIECE). Product Manual. BEIJING ZHIYANG BEIFANG INTERNACIONAL EDUCATION TECNOLOGY CO., LTD, 2023

BOSCH, Robert. **CAN Specification:** Version 2.0. Stuttgart, 1991. Disponível em: http://esd.cs.ucr.edu/webres/can20.pdf. Acesso em: 17 fev. 2025

BOSCH. **Data Logger and Sensor Interface C 80 Manual**. Disponível em: https://www.bosch-motorsport.com/. Acesso em: 17 fev. 2025

BYD Brasil. **A BYD:** Build your dream. Disponível em: https://www.byd.com.br/sobre/. Acesso em: 12 out. 2023.

ESPRESSIF. **ESP-IDF Programming Guide:** Controller Area Network (CAN) - release-v3.3. [2023]. Disponível em: https://docs.espressif.com/projects/esp-idf/en/release-v3.3/api-reference/peripherals/can.html. Acesso em: 18 fev. 2025.

FOWLER, Cory J. **MCP_CAN Library**, 2017. Disponível em: https://github.com/coryjfowler/MCP_CAN_lib. Acesso em: 19 fev. 2025.

GUIMARÃES, Alexandre de A. **Eletrônica Embarcada Automotiva.** 1 Edição: Editora Érica, 2007. ISBN 9788536518503.

GUIMARÃES, A. A.; SARAIVA, A. M.; **O Protocolo CAN:** Entendendo e Implementando uma Rede de Comunicação Serial de Dados baseada no Barramento “Controller Area Network”. SAE: 2002. Disponível em: http://www.alexag.com.br/Artigos/SAE2002.pdf. Acesso em 15 nov. de 2023.

IFSC - INSTITUTO FEDERAL DE SANTA CATARINA. ConverTE recebe bancadas didáticas de mobilidade elétrica. Disponível em: https://www.ifsc.edu.br/web/noticias/w/converte-recebe-bancadas-didaticas-de-mobilidade-eletrica?p_l_back_url=%2Fsearch%3Fq%3DBYD. Acesso em: 6 fev. 2025.

INTREPID CONTROL SYSTEMS. *VCAN4-2 User Guide.* Disponível em: https://cdn.intrepidcs.net/guides/vcan4-2/vcan4-2_ug.pdf. Acesso em: 6 fev. 2025.

KEYSIGHT TECHNOLOGIES. *Automotive Serial Triggering and Analysis (CAN/LIN) for DSOX1000 Series Oscilloscopes.* Disponível em: https://www.keysight.com/br/pt/product/DSOX1AUTO/automotive-serial-triggering-analysis-can-lin-dsox1000-series.html. Acesso em: 16 fev. 2025.

KUHLGATZ, Dietrich. **Data network for the car:** The Controller Area Network CAN. Bosch Research Blog, 2016. Disponível em: https://www.bosch.com/stories/the-controller-area-network. Acesso em: 27 set. 2023

MEHTA, C.; LIPPINCOTT S.; LU, C. **Communication Protocols in Modern ADAS Architectures.** Texas Instruments, 2023 Disponível em: https://www.ti.com/lit/wp/slyy219/slyy219.pdf?ts=1695866730979&ref_url=https%253A%252F%252Fwww.google.com.br%252F. Acesso em: 27 set. 2023

MENDES, Douglas Rocha. **Redes de Computadores:** Teoria e Prática. 2 Edição: Novatec, 2015.

MICHAEL, Stephen St. **Introduction to CAN (Controller Area Network).** All About Circuits, 2019. Disponível em: https://www.allaboutcircuits.com/technical-articles/introduction-to-can-controller-area-network/. Acesso em: 11 nov. 2023

MORAIS, Daniel Cardoso de. **Controlador CAN em VHDL**. 2012. 52 f. Universidade Federal de Campina Grande. Departamento de Engenharia Elétrica. Trabalho de Conclusão de Curso. Campina Grande, 2012. Disponível em: http://dspace.sti.ufcg.edu.br:8080/jspui/handle/riufcg/18235. Acesso em: 16 fev. 2025.

MOURA, T. C. M.; ANDRADE, I. L. **A UTILIZAÇÃO DA TELEMETRIA NA GESTÃO DE FROTA EM UMA EMPRESA DE LOGÍSTICA.** Universidade Federal do Rio Grande do Norte. Centro de Tecnologia. Departamento de Engenharia Mecânica. Trabalho de Conclusão de Curso. Natal, 2022. Disponível em: https://repositorio.ufrn.br/bitstream/123456789/46423/1/artigo_ata.pdf. Acesso em: 28 set. 2023.

NOVO Scania Euro 5 Diagnóstico de falhas na rede CAN com Válter - palestra gratuita Circuito 2020, 2020. 1 vídeo (77 min). Publicado pelo canal Doutor-IE. Disponível em: https://www.youtube.com/watch?v=joUYykzMG7A. Acesso em: 12 out. 2023.

PIRES, António Júlio Morais. **Comunicação de tempo-real em barramentos CAN baseados no controlador SJA1000.** Faculdade de Engenharia da Universidade do Porto. Engenharia Eletrotécnica e de Computadores. Área de especialização Informática e Automação. Mestrado. Porto, 2005. Disponível em: https://www.researchgate.net/profile/Antonio-Pires-7/publication/37655799_Comunicacao_de_tempo-real_em_barramentos_CAN_baseados_no_controlador_SJA1000_analise_e_implementacao_de_uma_solucao_para_o_escalonamento_de_mensagens/links/5677667e08ae502c99d2f514/Comunicacao-de-tempo-real-em-barramentos-CAN-baseados-no-controlador-SJA1000-analise-e-implementacao-de-uma-solucao-para-o-escalonamento-de-mensagens.pdf. Acesso em: 17 fev. de 2023.

POLESE, Bernardo. **Dispositivo de Telemetria Veicular Multiprotocolo com Transmissão via Internet.** Universidade de Passo Fundo. Departamento de Engenharia Elétrica. Trabalho de Conclusão de Curso. Passo Fundo, 2017.

Disponível em: repositorio.upf.br/bitstream/riupf/1233/1/PF2017Bernardo Polese.pdf. Acesso em: 15 nov. de 2023.

SANTOS, Rodrigo de Oliveira. **DATA ACQUISITION:** 6 REASONS TO USE AND THE FUNDAMENTALS YOU MUST KNOW. Racing Car Dynamics Blog, 2014. Disponível em: http://racingcardynamics.com/data-acquisition-fundamentals/. Acesso em: 11 out. 2023.

SMITH, Grant Maloy. **O que é barramento CAN (Controller Area Network) e como ele se compara a outras redes de barramento de veículos.** DEWESOFT, 2021. Disponível em: https://dewesoft.com/pt/blog/que-e-barramento-can. Acesso em: 15 nov. 2023

SOUSA, Rafael Vieira de. CAN Controller Network: uma abordagem para automação e controle na área agrícola. Escola de Engenharia de São Carlos, Universidade de São Paulo. Departamento de Engenharia Mecânica. Dissertação de Mestrado. São Carlos, 2002. Disponível em https://pdfs.semanticscholar.org/aa0b/8000643c08dfa179e2d234c85ee7f2a040fd.pdf . Acesso em: 16 fev. 2025

SOUSA, R. V.; GODOY, E. P.; PORTO, A. J. V; INAMASU, R. Y.; **Redes Embarcadas em Máquinas e Implementos Agrícolas:** o Protocolo CAN (Controller Area Network) e a ISO11783 (ISOBUS). Ministério da Agricultura, Pecuária e Abastecimento. Embrapa: São Paulo, 2007. Disponível em: https://ainfo.cnptia.embrapa.br/digital/bitstream/CNPDIA-2009-09/11046/1/DOC27_2007.pdf. Acesso em: 04 out. 2023.

SOUSA, R. V.; INAMASU, R. Y.; TORRE NETO, A. **CAN (Controller Area Network):** Um Padrão Internacional de Comunicação de Transdutores Inteligentes para Máquinas Agrícolas. Ministério da Agricultura, Pecuária e Abastecimento. Embrapa: São Paulo, 2001. Disponível em: https://www.infoteca.cnptia.embrapa.br/bitstream/doc/29133/1/CiT122001.pdf. Acesso em: 28 set. 2023.

SOUZA, Paulo Vitor. **Estudo e Elaboração de uma Rede CAN para Aplicação em um Sistema Automotivo.** Centro Federal de Educação Tecnológica de Minas Gerais Engenharia Mecatrônica. Monografia. Divinópolis, 2013. Disponível em: https://www.eng-mecatronica.divinopolis.cefetmg.br/wp-content/uploads/sites/195/2019/12/Paulo-V%C3%ADtor-de-Souza1.pdf. Acesso em: 15 fev. 2025.

TEKTRONIX. **5 Series MSO Serial Triggering and Analysis Applications**: 5-SRAERO, 5-SRAUDIO, 5-SRAUTO, 5-SRAUTOSEN, 5-SRCOMP, and 5-SREMBD Datasheet. 2018. Disponível em: https://www.farnell.com/datasheets/2609221.pdf. Acesso em: 19 fev. 2025.

RICHARDS, Pat. **A CAN Phisical Layer Discussion**. Microchip Technology, 2002. Disponível em: https://ww1.microchip.com/downloads/en/appnotes/00228a.pdf. Acesso em: 15 nov. 2023.

VECTOR INFORMATIK GMBH. CAN Controller: Learning Module CAN. Vector E-Learning, 22 set. 2021. Disponível em: https://elearning.vector.com/mod/page/view.php?id=333. Acesso em: 16 fev. 2025.

VECTOR INFORMATIK GmbH. CANoe DE Family. 2023. Disponível em: https://cdn.vector.com/cms/content/products/canoe/canoe/docs/Product%20Informations/CANoe_ProductInformation_EN.pdf. Acesso em: 18 fev. 2025.

VECTOR INFORMATIK GmbH. Introduction to Higher Level Protocols. Version 1.0. Application notes. Germany, 2014. Disponível em: https://cdn.vector.com/cms/content/know-how/_application-notes/canopen/AN-AND-1-160_Introduction_to_Higher_Level_Protocols.pdf. Acesso em: 19 fev. 2025.

WEIS, Olga. O que é um sistema de barramento CAN?, 2023 Disponível em: https://www.flexihub.com/pt/can-bus/. Acesso em: 14 dez. 2023