UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ
DEPARTAMENTOS ACADÊMICOS DE ELETRÔNICA E MECÂNICA
CURSO SUPERIOR DE TECNOLOGIA EM MECATRÔNICA INDUSTRIAL

VALÉRIO MENDES MAROCHI

CONTROLE REMOTO DE SINALIZAÇÃO E ILUMINAÇÃO POR COMANDO DE VOZ VIA APLICATIVO PARA CONDUTORES COM DEFICIÊNCIA FÍSICA INTEGRADO À REDE CAN

TRABALHO DE CONCLUSÃO DE CURSO

CURITIBA
2018

VALÉRIO MENDES MAROCHI

CONTROLE REMOTO DE SINALIZAÇÃO E ILUMINAÇÃO POR COMANDO DE VOZ VIA APLICATIVO PARA CONDUTORES COM DEFICIÊNCIA FÍSICA INTEGRADO À REDE CAN

Trabalho de Conclusão de Curso de Graduação, apresentado ao Curso Superior de Tecnologia em Mecatrônica Industrial, dos Departamentos Acadêmicos de Eletrônica e Mecânica, da Universidade Tecnológica Federal do Paraná – UTFPR, como requisito parcial para obtenção do título de Tecnólogo.

Orientador: Prof. Dr. Dyson Pereira Jr.

CURITIBA
2018

TERMO DE APROVAÇÃO

VALÉRIO MENDES MAROCHI

CONTROLE REMOTO DE SINALIZAÇÃO E ILUMINAÇÃO POR COMANDO DE VOZ VIA APLICATIVO PARA CONDUTORES COM DEFICIÊNCIA FÍSICA INTEGRADO À REDE CAN

Este trabalho de conclusão de curso foi apresentado no dia 22 de novembro de 2018, como requisito parcial para obtenção do título de Tecnólogo em Mecatrônica Industrial, outorgado pela Universidade Tecnológica Federal do Paraná. O aluno VALÉRIO MENDES MAROCHI foi arguido pela Banca Examinadora composta pelos professores abaixo assinados. Após deliberação, a Banca Examinadora considerou o trabalho aprovado.

_____
Prof. Dr. Milton Luiz Polli
Coordenador de Curso
Departamento Acadêmico de Mecânica

_____
Prof. M.Sc. Sérgio Moribe
Responsável pela Atividade de Trabalho de Conclusão de Curso
Departamento Acadêmico de Eletrônica

BANCA EXAMINADORA

_____
Prof. Sérgio Luiz Bazan de Paula
UTFPR

_____
Prof. Ubiradir Mendes Pinto
UTFPR

_____
Prof. Dr. Dyson Pereira Jr.
Orientador - UTFPR

AGRADECIMENTOS

Não é possível expressar em palavras, e tampouco numa quantidade tão pequena de linhas, a relevância e a contribuição de cada um daqueles que influenciaram nas escolhas, planos e caminhos por mim adotados e trilhados neste ciclo da minha vida, que se encerra com este trabalho de conclusão de curso. Mesmo assim, fica aqui o singelo agradecimento a todos que protagonizaram a construção gradativa de um novo eu.

Acima de todos, agradeço ao Pai, ao Filho e ao Espírito Santo que, usando de amor e misericórdia, garantiram a minha existência e me sustentaram vivos para ver concluídos planos e sonhos não imaginados no começo desta jornada.

Agradeço à minha esposa, amiga, amada e metade de mim, Monique Miriellen Soares Marochi, por ser coluna e sustentáculo, sempre me apoiando e dividindo o fardo sem desfalecer e desanimar.

Também agradeço à família, por vibrar e se alegrar com as minhas conquistas, em especial aos familiares mais próximos, a saber: meus pais, Lúcia e José; meus sogros, Mirian e Claudionor; minha irmã e meu cunhado Karina e Diego; e minha cunhada, Poliana. Certamente vocês são parte desta conquista.

Agradeço aos meus amigos e colegas de trabalho do Senai Ernesto, Felipe Gabriela, Pedro, Peterson, Wilson, Luiz Fernando e Jorge, que, de forma sincera e determinada, cada um a seu modo, me incentivaram e apoiaram na conclusão desta etapa acadêmica e profissional.

Por fim, agradeço aos amigos e colegas adquiridos na universidade, tanto professores quanto colegas de turma, que tornaram as horas de aprendizados mais produtivas e relevantes por meio da empatia e do mútuo auxílio.

“A ciência é, portanto, uma perversão de si mesma, a menos que tenha como fim último, melhorar a humanidade.”

Nikola Tesla (1856 – 1943)

RESUMO

MAROCHI, Valério Mendes. **Controle remoto de sinalização e iluminação por comando de voz via aplicativo para condutores com deficiência física integrado à rede CAN.** 2018. 85 f. Trabalho de Conclusão de Curso (Curso Superior de Tecnologia em Mecatrônica Industrial), Departamentos Acadêmicos de Eletrônica e Mecânica, Universidade Tecnológica Federal do Paraná. Curitiba, 2018.

Os veículos de um modo geral, para além de meros meios de transporte, tornaram-se grandes vitrines tecnológicas, contendo sistemas embarcados cada vez mais complexos, que demandam o uso de protocolos e redes de comunicação com o intuito de integrar as diversas funcionalidades dos sistemas veiculares dedicados. Mesmo assim, com tanta tecnologia, ainda há muitas oportunidades de desenvolvimento de sistemas e soluções, que atendam necessidades específicas, como é o caso das adaptações automotivas para portadores de deficiência física. O foco deste trabalho é desenvolver um dispositivo de controle remoto que possibilite ao condutor com deficiências motoras o acionamento dos sistemas de sinalização e iluminação do veículo por comando de voz via aplicativo móvel. Para tanto, realizou-se uma pesquisa bibliográfica e documental a fim de evidenciar a importância dos protocolos de redes veicular, mais especificamente a rede CAN, bem como o uso de plataformas de prototipagem no desenvolvimento de soluções embarcadas para acessibilidade e adaptabilidade veicular. Uma vez obtidas as informações necessárias, foi desenvolvido um protótipo e um aplicativo móvel para controle remoto por comando de voz integrado à rede CAN veicular.

Palavras chave: Redes Veiculares. Microcontroladores. Sistemas Embarcados. Condutor deficiente físico. Controle remoto veicular.

ABSTRACT

MAROCHI, Valério Mendes. **Remote control of signaling and lighting by voice command via application for drivers with physical disabilities integrated with CAN network.** 2018. 85 f. Trabalho de Conclusão de Curso (Curso Superior de Tecnologia em Mecatrônica Industrial), Departamentos Acadêmicos de Eletrônica e Mecânica, Universidade Tecnológica Federal do Paraná. Curitiba, 2018.

In general, vehicles, as well as mere means of transport, have become large technological showcases, containing increasingly complex embedded systems, which require the use of protocols and communication networks in order to integrate the various functionalities of the multiple vehicle systems. Even so, with so much technology, there are still many opportunities for developing systems and solutions that address specific needs, such as automotive adaptations for the physically disabled. The focus of this work is to develop a remote control device that enables the driver with motor disabilities to control the vehicle's signaling and lighting systems by voice command via a mobile application. In order to do so, a bibliographical and documentary research was carried out in order to highlight the importance of the vehicular network protocols, more specifically the CAN network, as well as the use of prototyping platforms in the development of embedded solutions for vehicular accessibility and adaptability. Once the necessary information was obtained, a functional workbench was built containing commercial vehicle components with CAN network communication, a microcontroller interface integrated with it, and a mobile application for remote control by voice command, evidencing the opportunities for developing solutions of this kind.

Keywords: Vehicle Networks. Microcontrollers. Embedded Systems. Physically disabled driver. Vehicle Remote Control.

LISTA DE ILUSTRAÇÕES

LISTA DE ABREVIATURAS, SIGLAS E ACRÔNIMOS

ABS – *Anti-lock Brake System*
ACK – *Acknowledgement Error Check*
A/D – Analógico/Digital
ASCII – *American Standard Code for Information Interchange*
BIOS – Basic Input/Output System
BPS – *Bits per second*
BSI – *Boîtier de Servitude Intelligent*
CAN – *Controller Area Network*
CAN H – CAN *High*
CAN L – CAN *Low*
CAR – *Carrosserie*
CI – Circuito Integrado
CONF - *Confort*
CPU – *Central Processing Unit*
CRC – *Cyclic Redundancy Check*
CS – *Chip Select*
CSMA/CR - *Carrier Sense Multiple Access/Collision Resolution*
DF – Data Field
DIN - *Deutsches Institut für Normung*
DLC – *Data Length Code*
ECU – *Electronic Control Unit*
EEPROM – *Electrically-Erasable Programmable Read-Only Memory*
EOF – *End-of-frame*
GmbH – *Gesellschaft mit beschränkter Haftung*
GPS – *Global Positioning System*
I²C – *Inter-Integrated Circuit*
ID – *Identifier*
IDE – *Integrated Development Environment*
IDII – *Interaction Design Institute Ivrea*
I/O's – *Inputs and Outputs*
I/S – *Intersystems*
ISO – *Internacional Standards Organization*
LCD – *Liquid Crystal Display*
LDR – *Light-Dependent Resistor*
LED – *Light Emitting Diode*
LLC – *Logical Link Layer*
MAC – *Medium Access Control*
MCP – *Master Control Program*
MDF – *Medium Density Fiberboard*
MIT – *Massachusetts Institute of Technology*
NMEA – *National Marine Electronic Association*
OBD – *On-Board Diagnostic*
OS – *Operational System*
OSI – *Open System Interconnection*
PCB – *Printed Circuit Board*
PCI – *Peripheral Component Interconnect*
PIC – *Peripherical Interface Controller*
PSA – *Peugeot Citroën Moteurs*

PWM – *Pulse Width Modulation*
PSF1 – *Powertrain Systems Fusebox 1*
RAM – *Read Access Memory*
REC – *Receiver Error Counter*
RFID – *Radio-Frequency IDentification*
ROM – *Read Only Memory*
RTR – *Remote Transmission Request*
SAE – *Society of Automotive Engineers*
SOF – *Start-of-frame*
SPI – *Serial Peripheral Interface*
SRR – Substitute Remote Request
TCP/IP – *Transmission Control Protocol/ Internet Protocol*
USB – *Universal Serial Bus*
VAC – *Voltage Alternated Current*

SUMÁRIO

1 INTRODUÇÃO

Observando-se o cenário tecnológico mundial, a mudança de paradigmas que ocorre no segmento automotivo e a inserção de novos dispositivos e tecnologias nos automóveis, percebe-se que estes, para além de meros meios de transporte, tornaram-se, há algum tempo, grandes vitrines tecnológicas. Agregando a maioria dos sistemas e soluções de ponta que existem num único dispositivo, os automóveis do futuro tendem a oferecer sistemas de direção autônoma, conectividade com dispositivos externos e com outros veículos, dentre outras soluções (KALOGIANNI, 2018). Tal cenário de inovação automotiva só é possível devido à redução dos custos de implantação de novas tecnologias e evolução da arquitetura eletroeletrônica veicular, inicialmente muito simples e com estrutura centralizada (devido a pequena quantidade de dispositivos eletroeletrônicos presentes nos veículos). Com o aumento da quantidade de sistemas e componentes eletrônicos presentes nos carros, oferecendo mais conforto e segurança aos passageiros, houve também a necessidade da mudança da arquitetura elétrica para o modelo distribuído, por meio do qual é possível interligar ECU's (*Electronic Control Units*) e compartilhar informações entre elas (GUIMARÃES, 2012; p. 201-203).

Segundo Bosch (2005; p.1070-1072), o emprego da arquitetura distribuída é uma solução para o aumento da complexidade dos sistemas eletroeletrônicos, devido à grande quantidade de variáveis de entrada e saída e o tamanho físico do sistema, além da necessidade de sistemas com alto nível de confiabilidade e robustez. Para tanto, numa rede distribuída, faz-se necessária a implementação de um meio de comunicação entre as ECU's, o protocolo de comunicação, que organizará as informações transmitidas na rede através de um software de controle (GUIMARÃES, 2012; p. 199-208). A Bosch GmbH desenvolveu o primeiro protocolo de comunicação para aplicação específica automotiva em 1983, intitulado de *Controller Area Network* (CAN). Esse tipo de protocolo é baseado em comunicação serial, por mensagens estruturadas endereçadas a controladores, ou seja, é implementado em redes condicionadas a controladores, que determinam a importância e prioridade das mensagens trocadas entre eles e seus dispositivos escravos.

1.1 TEMÁTICA E DELIMITAÇÃO DO ESTUDO

O presente projeto aborda a temática das redes veiculares dando ênfase ao protocolo CAN, suas características e aplicações. Para tanto há um enfoque em plataformas de prototipagem eletrônica e sistemas embarcados, culminando no desenvolvimento de uma solução capaz de atuar em dispositivos comerciais veiculares de forma integrada ao protocolo CAN, atendendo a demandas por adaptações automotivas voltadas para deficientes físicos.

1.2 PROBLEMA

Os protocolos de comunicação – em especial a rede CAN – são imprescindíveis para aplicações de dispositivos automotivos, sejam eles de série, opcionais ou mesmo acessórios de outros fabricantes que não a própria montadora. Isto se deve ao fato de que na maioria das aplicações eletroeletrônicas atuais, a adição e configuração de funcionalidades ou mesmo a leitura e extração de informações do veículo somente se dá através destes protocolos, não sendo mais possível, na maioria dos casos, conectar circuitos elétricos analógicos (lógica relé) avulsos e fazê-los funcionar com os sistemas originais. Para tanto, a compreensão de tais protocolos e a criação de dispositivos e sistemas neles baseados figuram como conhecimentos importantes para toda e qualquer solução automotiva ou projeto inovador que se pretenda desenvolver.

Um outro olhar sobre o assunto reside no fato de que, existem muitos projetos e estudos com foco na programação de controladores e plataformas de desenvolvimento, porém as aplicações são sempre industriais, não havendo direcionamento de criação e inovação voltadas para o segmento automotivo. Isto se deve em parte ao protecionismo de montadoras em não abrir tecnologias e pesquisas para os mercados emergentes, dentre eles o Brasil, mas também às instituições de ensino e mesmo aos discentes das mesmas, que não dispõe de conhecimento suficiente para abordar projetos nesta área.

Dentro deste contexto de aplicações veiculares e desenvolvimento de novas soluções ou dispositivos embarcados, a acessibilidade veicular é um nicho de mercado crescente que recebe pouca atenção de fabricantes de automóveis e

sistemistas automotivos. Segundo o último Censo Demográfico (INSTITUTO..., 2018; Loschi, 2017), 23,9% da população apresenta algum tipo de deficiência, sendo destes 29% do tipo motora. Poucos são os fabricantes em escala global que desenvolver soluções voltadas à adaptabilidade, não somente de acesso, mas também de operação e interação com o veículo.

Sendo assim, pretende-se, através deste projeto, realizar um estudo dos protocolos de comunicação veiculares e desenvolver uma aplicação acessória voltada para a acessibilidade, que opere em sistemas veiculares através da rede CAN original do fabricante, com o intuito de adaptar comando originais de sinalização e iluminação do carro às necessidades de condutores que apresentem deficiências motoras.

1.3 OBJETIVOS

Nesta seção são apresentados os objetivos geral e específicos do trabalho, relativos ao problema anteriormente apresentado.

1.3.1 Geral

Desenvolver um sistema de controle remoto integrado à rede CAN, que possibilite ao condutor portador de deficiência motora acionar o sistema de sinalização e iluminação por comando de voz via aplicativo instalado em dispositivo móvel.

1.3.2 Objetivos Específicos

Abaixo seguem as etapas planejadas para o atingimento do objetivo geral do projeto:

1. Estudar o protocolo de comunicação CAN, suas características, funcionamento e aplicações, bem como as plataformas de prototipagem eletrônica e os sistemas embarcados;
2. Levantar os dispositivos e plataformas existentes que possam ser utilizados no desenvolvimento da solução, bem como os equipamentos e demais materiais necessários à confecção da mesma;
3. Confeccionar o protótipo da solução e testá-lo com plataforma veicular comercial.

1.4 JUSTIFICATIVA

As razões para a proposição deste projeto são de caráter: a) técnico, apontando para a possibilidade de desenvolvimento de projetos para conectividade relacionados às redes e protocolos veiculares; b) social, dando destaque para a necessidade de provisão, por meio da tecnologia, de acessibilidade aos deficientes num âmbito geral; c) educacional, incentivando estudantes e desenvolvedores das áreas técnicas e tecnológicas a pesquisar e desenvolver dispositivos e soluções inovadoras para o segmento automotivo.

Para tanto, o primeiro objetivo do projeto é o estudo: do protocolo de comunicação CAN, suas características, funcionamento e aplicações; e dos sistemas embarcados e suas características, a fim de levantar os dispositivos e plataformas existentes que possam ser utilizados no desenvolvimento da solução proposta. Uma vez concluído o protótipo, pretende-se evidenciar a necessidade e a possibilidade de desenvolvimento no tema redes de comunicação veiculares, impactando na formação de discentes em cursos nas mais diversas áreas e níveis – mecânica, eletroeletrônica, automotiva –, desde qualificações e cursos técnicos, até graduações e pós-graduações.

Dito isso, acredita-se que, por meio do referido projeto, será possível aplicar e ratificar conceitos e conhecimentos abordados em diversas disciplinas do curso de Tecnologia em Mecatrônica Industrial. Para além disso, dada a importância da área automotiva para a indústria do estado do Paraná e, sendo o Brasil o oitavo maior mercado automotivo do mundo e nono maior produtor de veículos (CHADE, 2018; ORGANISATION..., 2018), acredita-se que o estudo de redes veiculares e a confecção do protótipo oportunizará, não somente ao autor, mas a todos que dele se utilizarem, a agregação de conhecimento e experiência numa área de extrema importância e com grandes oportunidades no que diz respeito ao fomento da tecnologia e inovação no país.

1.5 PROCEDIMENTOS METODOLÓGICOS

O projeto em questão é de natureza aplicada, de caráter documental, bibliográfico e experimental, agregando informações obtidas por meio de observação individual sistemática e entrevista não-estruturada, objetivando gerar conhecimentos para aplicação prática e dirigida à solução de problemas específicos, gerando um produto (protótipo) (SILVA; MENEZES, 2005).

Do ponto de vista dos objetivos, tratar-se-á de uma pesquisa exploratória e explicativa, através do estudo do protocolo de comunicação veicular CAN, das plataformas de prototipagem eletrônica e dos sistemas embarcados, considerando: histórico, características, princípios de funcionamento, arquitetura de software e hardware e aplicações. Os procedimentos adotados para a coleta das informações e dos dados serão as pesquisas bibliográfica, documental e experimental (SILVA; MENEZES, 2005). Ainda do ponto de vista dos procedimentos técnicos, consta que o projeto é do tipo pesquisa-ação participante, a ser desenvolvido por meio da ação e da resolução de problemas, constando das seguintes etapas:

a) Projeto de um dispositivo acessório veicular que possibilite a atuação de sistemas de sinalização e iluminação por meio de comando de voz, considerando estrutura física, arquitetura de software, hardware e funcionalidades;

b) Levantamento dos componentes, dispositivos e plataformas existentes que possam ser utilizados no desenvolvimento da solução automotiva proposta, bem como os equipamentos e demais materiais necessários à confecção da mesma;

c) Aquisição de peças, componentes e insumos, para o desenvolvimento do projeto;

d) Confecção de uma bancada com componentes automotivos que possuam comunicação via protocolo CAN, a fim de permitir o teste do projeto em plataforma automotiva comercial;

e) Desenvolvimento da estrutura física do dispositivo, considerando suportes e alojamentos para os circuitos e componentes;

f) Desenvolvimento do circuito eletroeletrônico, utilizando-se como controlador a plataforma Arduino UNO, módulo de comunicação CAN Bus MCP2515, dentre outros componentes eletrônicos;

g) Desenvolvimento da programação dos controladores da bancada e do protótipo em si;

h) Desenvolvimento de um aplicativo para dispositivos móveis que possibilite o uso de *smartphones* e *tablets* como periférico de entrada do comando por voz;

i) Integração da bancada de testes com o controle remoto e com o aplicativo de comando por voz;

j) Teste das funcionalidades da bancada de teste, do dispositivo e ajustes (se necessários).

1.6 EMBASAMENTO TEÓRICO

No estudo e abordagem dos protocolos de comunicação e de rede as literaturas Tanembaum (2011) e Guimarães serão utilizadas, sendo este último, juntamente com Bosch (2005) e Corrigan (2012) fundamentação para o estudo de redes veiculares e protocolo de comunicação CAN. Para a abordagem de conceitos como microcontroladores tem-se como base as fontes: Conceição (2012), Penido e Trindade (2013), ATMEL (2013), Intel (1978). Sistemas embarcados, plataformas de prototipagem e a temática específica a respeito de Arduino serão expostos tendo em vista a ótica de Noergaard (2013), com dados e informações de Lindsay (2005), Curvello (2014), Arduino (2018), Texas Instruments (2018), Raspberry Pi Foundation (2018), Espressif Systems (2018), Barragán (2016) e Circuits Today (2018). Módulos de prototipagem, módulo CAN Bus e Shields Filipeflop (2018), Microchip (2005), Bench (S.d.), Philips (2003). No que diz respeito à abordagem das adaptações automotivas para deficientes, as fontes são: Instituto... (2010), Garcia (2009) e Motability (2018). Quanto aos dispositivos de controle remoto adaptados para deficientes as fontes de informações são alguns sites e catálogos de fabricantes e instaladores, como: Guidosimplex (2018), Helio (2018), Cavenaghi (2018), Kivi (2018), Bever (S.d.), Lodgesons (2018) e Paravan (2018).

1.7 ESTRUTURA DO TRABALHO

O trabalho em questão conta com cinco capítulos, divididos conforme descrito abaixo:

Capítulo 1 - Introdução: serão apresentados o tema, as delimitações da pesquisa, o problema e a premissa, os objetivos da pesquisa, a justificativa, os procedimentos metodológicos, as indicações para o embasamento teórico, e a estrutura geral do trabalho.

Capítulo 2 – Fundamentação Teórica: discorre a respeito dos conceitos teóricos necessários, à compreensão de protocolos de comunicação e sistemas embarcados, bem como as tecnologias e plataformas de prototipagem disponíveis para a experimentação e o desenvolvimento de soluções microcontroladas.

Capítulo 3 – Desenvolvimento do Tema: apresentará todas as informações e etapas concernentes à confecção da bancada de testes de rede veicular, do sistema de controle por comando de voz integrado à rede CAN e do aplicativo móvel para interface com o mesmo.

Capítulo 4 – Apresentação e Análise dos Resultados: como o título prenuncia, resume os dispositivos e resultados obtidos ao final do desenvolvimento do projeto, bem como os desafios e dificuldades encontrados no decorrer da sua realização.

Capítulo 5 – Considerações finais: apresentação das soluções alcançadas para os problemas propostos no início do trabalho, considerando os objetivos listados, além de indicação dos desdobramentos que os resultados podem acarretar, como modificação do projeto, adaptação e aplicação em outras realidades e sistemas.

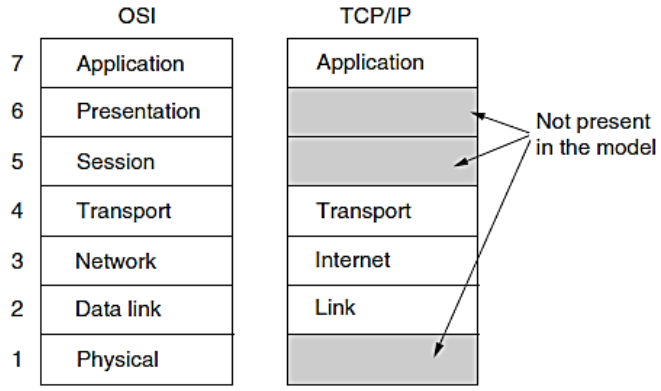
2 FUNDAMENTAÇÃO TEÓRICA

A fim de tornar possível o desenvolvimento do protótipo da solução a que se refere este trabalho, tornou-se necessário compreender, conforme apresentado nos capítulos a seguir: conceitos de redes veiculares, microcontroladores e sistemas embarcados, plataformas de prototipagem eletrônica, módulos e shields e ambientes de desenvolvimento, onde são programados os dispositivos e sistemas embarcados.

2.1 ARQUITETURA DE REDES

Segundo Tanenbaum (2011), na ciência da computação, redes são sistemas que interligam computadores, a fim de permitir a comunicação - troca de dados – entre eles, sem a necessidade de compartilhar o mesmo espaço físico. As arquiteturas das redes podem ser classificadas ou estruturadas tendo como base dois modelos de referência principais: o modelo OSI (ISO 7498) e o modelo TCP/IP, Figura 1. Cada modelo possui características específicas, porém ambos continuam sendo utilizados na estruturação e organização de arquiteturas de rede nas mais diversas aplicações.

Figura 1. Comparativo entre os modelos OSI e TCP/IP.



Fonte: TANENBAUM (2011).

Ainda segundo Tanenbaum (2011), as arquiteturas, por sua vez, podem ser resumidas em dois conceitos principais: camadas e protocolos. Devido à evolução das redes e ao crescente nível de complexidade em termos de quantidade de dispositivos interligados, quantidade de dados em trânsito e necessidade de integração de dispositivos de fabricantes e tecnologias de níveis diferentes, as redes passaram a

ser separadas, como se fosse uma pilha, em camadas ou níveis, que podem ser físicos (*hardware*) e/ou lógicos (*software*). Uma camada, portanto, diz respeito a uma parte constituinte da rede que fornece determinados serviços ou dados para a camada superior a ela. Já os protocolos são convenções que controlam e possibilitam a conexão, a comunicação e a transferência de dados entre dispositivos e/ou sistemas operacionais, ou entre camadas (chamados de interfaces).

Nas aplicações automotivas, os protocolos de comunicação podem ser resumidos como regras que governam a comunicação entre controladores eletrônicos e sensores e/ou atuadores inteligentes. Conforme a arquitetura dos sistemas eletroeletrônicos veiculares evoluiu, os fabricantes foram desenvolvendo soluções para aplicações pontuais, motivo pelo qual existem diversos tipos de protocolos diferentes, alguns inclusive já em desuso. A SAE desenvolveu uma classificação dos principais protocolos, separando-os em três categorias – A, B e C – de acordo com a taxa de transmissão e criticidade da aplicação (GUIMARÃES, 2012):

- Classe A: taxa de transmissão em torno de 10 kbps, voltados para aplicações de conforto e sistemas auxiliares, como iluminação, vidros, assentos, etc., conforme Figura 2.

Figura 2. Tipos e características de protocolos Classe A.

	SINEBUS	I²C	SAE J1708	CCD	ACP	BEAN	LIN
Instituição relacionada	Delco Electronics	Philips	SAE/TCM	Chrysler	Ford	Toyota	Motorola
Aplicação principal	Sistemas de Áudio	Comunicação entre displays e rádios	Controle e Diagnóstico	Controle e Diagnóstico	Sistemas de Áudio	Controle e Diagnóstico	Sensores e atuadores inteligentes
Tipo de barramento	Fio único	Par trançado	Par trançado	Fio único	Par trançado	Fio único	Fio único
Quantidade de dados	10 – 18 bits	-	-	5 Bytes	6 – 12 Bytes	1 – 11 Bytes	8 Bytes
Taxa de transmissão	66,6 – 200 Kbps	1 – 100 Kbps	9600 bps	7812, 5 bps	9600 bps	10 Kbps	20 Kbps
Comprimento máximo do barramento	10 m	-	40 m	-	40 m	-	40 m
Quantidade máxima de nós na rede	-	-	20	10	20	20	16

Fonte: Adapt. de GUIMARÃES (2012).

- Classe B: taxa de transmissão entre 10 e 125 kbps, utilizados em sistemas de entretenimento, navegação, *e-Call*, sistemas de bordo, *infotainment* em geral, de acordo com a Figura 3.

Figura 3. Tipos e características de protocolos Classe B.

	CAN 2.0 ISO 11898 & ISO 11519-2	CAN 2.0 J1939	J1859 Class 2	J1850 SCP	J1850 PCI
Instituição relacionada	SAE & ISO	SAE	GM	Ford	Chrysler
Aplicação principal	Controle e Diagnóstico	Controle e Diagnóstico	Controle e Diagnóstico	Controle e Diagnóstico	Controle e Diagnóstico
Tipo de barramento	Par trançado	Par trançado	Fio único	Par trançado	Fio único
Quantidade de dados	0 – 8 Bytes	0 – 8 Bytes	0 – 8 Bytes	0 – 8 Bytes	0 – 8 Bytes
Taxa de transmissão	10 Kbps – 1 Mbps	250 Kbps	10,4 Kbps	41,6 Kbps	10,4 Kbps
Comprimento máximo do barramento	40 m (p/ 1 Mbps)	40 m (p/ 1 Mbps)	35 m	35 m	35 m
Quantidade máxima de nós na rede	32	32	32	32	32

Fonte: Adapt. de GUIMARÃES (2012).

- Classe C: taxa de transmissão entre 125 kbps e 1 Mbps, aplicado em sistemas de segurança e controle em tempo real, como grupo motopropulsor, suspensão ativa, controle e assistência de frenagem, etc., vide Figura 4.

Figura 4. Tipos e características de protocolos Classe C.

	CAN 2.0 ISO 11898 & ISO 11519-2	CAN 2.0 J1939
Instituição relacionada	SAE & ISO	SAE
Aplicação principal	Controle e Diagnóstico	Controle e Diagnóstico
Tipo de barramento	Par trançado	Par trançado
Quantidade de dados	0 – 8 Bytes	0 – 8 Bytes
Taxa de transmissão	10 Kbps – 1 Mbps	250 Kbps
Comprimento máximo do barramento	40 m (p/ 1 Mbps)	40 m (p/ 1 Mbps)
Quantidade máxima de nós na rede	32	32

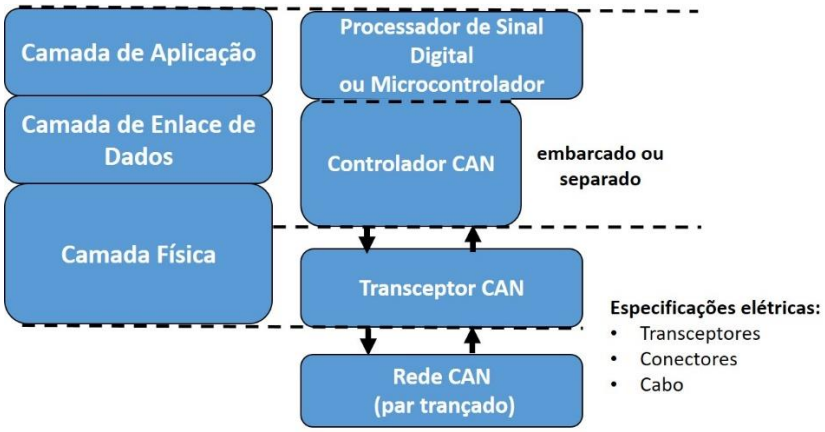
Fonte: Adapt. de GUIMARÃES (2012).

2.2 REDE CAN

O CAN (*Controller Area Network*) é um protocolo de rede assim denominado pela característica de ser dependente (ou baseado no comando) de um controlador atuando em uma planta (área de dispositivos), diferentemente de protocolos de rede como os baseados no modelo TCP/IP, por exemplo. Inicialmente desenvolvido pela

Bosch GmbH em 1983, tinha como finalidade multiplexar sistemas de veículos comerciais (caminhões e ônibus), diminuindo a redundância de dispositivos e a quantidade de conexões elétricas e chicotes no veículo (BOSCH, 2005; GUIMARÃES, 2012). Atualmente, o protocolo CAN é definido por duas normas principais: a ISO 11898, que determina as características das redes de alta velocidade (125 kbps a 1 Mbps), e a ISO 11519-2, para as redes de baixa velocidade (10 kbps a 125 kbps). A arquitetura da rede é fundamentada em três das camadas do modelo OSI: física, enlace de dados, e de aplicação. O desenvolvimento desta última camada torna a rede mais completa do ponto de vista da segurança das mensagens e da integridade do sistema, porém é mais explorada nas aplicações industriais do que nas automotivas, Figura 5.

Figura 5. Arquitetura CAN por camadas (padrão ISO 11898).



Fonte: Adapt. de CORRIGAN (2016).

Dentre suas principais especificações, conforme Figura 6, destacam-se: a comunicação serial, multimestre e *multicast*, a detecção de falhas e a arbitragem de rede (prioridade). Além disso, a estrutura física da rede (par trançado) garante baixo custo de implantação, robustez – resistência a tração, torção e vibrações – e alto nível de proteção contra interferências eletromagnéticas (CORRIGAN, 2016; GUIMARÃES, 2007).

Figura 6. Características do protocolo CAN.

Característica	Descrição
Comunicação serial, assíncrona e balanceada	Informações transmitidas sequencialmente, sem uso de clock e com compensação de nível de tensão (GND comum)
Multimestre	Todos os nó podem se tornar mestres ou escravos de acordo com a situação
Multicast	As mensagens transmitidas estão disponíveis para todos os nós
Robusto	Barramento resistente a interferências eletromagnéticas e esforços mecânicos (tração, torção e vibrações)
Detecção de falhas	Nível lógico: detecta falhas no sinal ou nas mensagens e armazena informação Nível físico: operação em modo de segurança quando a falha é física (curto-circuito entre os fios de dados, curto-circuito para o massa e circuito aberto).
Arbitragem de rede	O barramento é ocupado de acordo com a prioridade da mensagem transmitida.
Baixo custo	Arquitetura simples (par trançado) e de fácil implementação.

Fonte: Autoria própria.

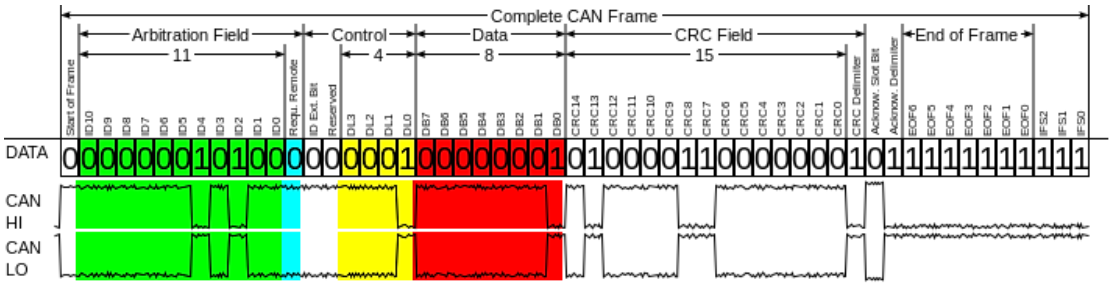
Ainda segundo Corrigan (2016), devido às características anteriormente citadas, o protocolo ganhou proporções e capilaridade em diversas áreas, tendo dado origem a outros protocolos com aplicações específicas, dentre elas: veicular (passeio, comercial, agrícola, construção, marítimo, militar), industrial (equipamentos e plantas), médica (equipamentos médicos e hospitalares) e predial e residencial (automação de dispositivos de monitoramento, sistemas de controle de elevadores).

2.2.1 Formato de Mensagens CAN

Levando-se em consideração que a rede CAN pode operar de forma assíncrona, ou seja, sem a necessidade de *clock* que determine intervalos de tempo entre a transmissão e o recebimento de uma mensagem entre os nós da rede, uma das principais características do protocolo, além das já citadas, é o formato das mensagens. Resumidamente, na rede CAN todos os elementos conectados (nós) podem enviar e receber mensagens, porém não simultaneamente, sendo definida a prioridade de transmissão, ou o uso do barramento, de acordo com o tamanho do identificador (ID) da mensagem. Diferentemente de uma rede tradicional, como USB ou *Ethernet*, que são baseadas na transmissão de longos blocos de mensagens entre dois pontos, sob a supervisão de um mestre a um intervalo de tempo definido, no protocolo CAN as mensagens são disponibilizadas para todos os nós acoplados à

rede em conjuntos mais curtos, conforme Figura 7 (CORRIGAN, 2016; GUIMARÃES, 2007).

Figura 7. Formato de bloco de mensagens CAN (com ID de 11 bits).



Fonte: WIKIPEDIA (2018).

Observando-se ainda a Figura 7, verifica-se que dentro do bloco de dados, cada bit, ou grupo de bits, corresponde a uma função específica do protocolo, como prioridade (verde), verificação de integridade (amarelo), mensagem em si (vermelho), detecção de falha, etc. As mensagens também podem variar dentro de dois formatos padronizados, sendo o último o mais utilizado:

- CAN 2.0A, formato padrão, com identificador de 11 bits, o que possibilita 2048 mensagens diferentes;
- CAN 2.0B, formato estendido, com identificador de 29 bits, o que expande a capacidade da rede para algo em torno de 537 milhões de mensagens diferentes.

No quadro a seguir estão especificados os bits e grupos de bits do bloco de dados, de acordo com a função no protocolo:

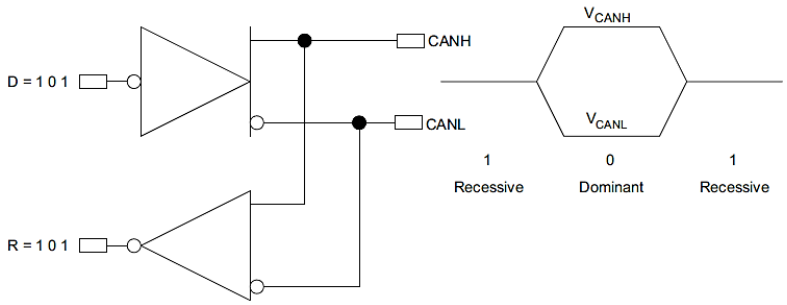
Quadro 1 – Bloco de dados CAN

Campo	Nome do campo	Tamanho (bits)	Função
SOF	(*Start-of-frame*) Começo do bloco	1	Indica o início da transmissão da mensagem
ID A	(*Identifier*) Identificador A	11	Determina a prioridade da mensagem
SRR	(*Substitute Remote Request*) Solicitação de transmissão remota substituta	1	CAN 2.0A – não se aplica CAN 2.0B - Aplicado em caso de uso de bloco remoto de dados
IDE	(*Identifier Extension Bit*) – Bit identificador de Extensão	1	CAN 2.0A – não se aplica CAN 2.0B – recessivo (1)
ID B	(*Identifier*) Identificador B – somente CAN 2.0B	18	Determina a prioridade da mensagem
RTR	(*Remote Transmission Request*) Solicitação de transmissão remota	1	Dominante (0)
-	Bit de extensão do ID	1	CAN 2.0A – dominante (0) CAN 2.0B – recessivo (1)
-	Bit reservado	1	Sempre dominante (0)
DLC	Código de Comprimento de dado	4	Indica o número de Bytes da mensagem
DF	(*Data Field*) Campo de dados	0-64 (0-8 Bytes)	Mensagem a ser transmitida
CRC	(*Cyclic Redundancy Check*) Verificador de redundância cíclica	15	Utilizado pelo nó transmissor para reconhecer o recebimento da mensagem
CRC	Delimitador CRC	1	Indica o fim da verificação de redundância Sempre recessivo (1)
ACK	(*Acknowledgement Error Check*) Verificação de Erro de Reconhecimento	1	Transmissão – recessivo (1) Recepção – dominante (0)
ACK	Delimitador ACK	1	Indica o fim do reconhecimento Sempre recessivo (1)
EOF	(End-of-Frame) Fim do Bloco	7	Sempre recessivo (1)

Fonte: Autoria própria.

Os dados são transmitidos através de pulsos em corrente contínua, por meio de um par trançado de fios – chamados de CAN H (*High*) e CAN L (*Low*), a fim de garantir a proteção eletromagnética do barramento. Para que essa transmissão ocorra desta maneira, o estado lógico do sinal elétrico gerado é invertido entre os cabos, sendo que o bit dominante tem estado lógico “0” e o recessivo estado lógico “1”, vide Figura 8. Observa-se também que esse tipo de sinal gera a “ocupação” do barramento, sempre que um sinal dito dominante (“0”) é disparado na rede. Este método de transmissão possibilita a arbitragem da rede CAN, ou seja, qual mensagem será transmitida primeiro quando dois ou mais nós tentarem ocupar o barramento ao mesmo tempo.

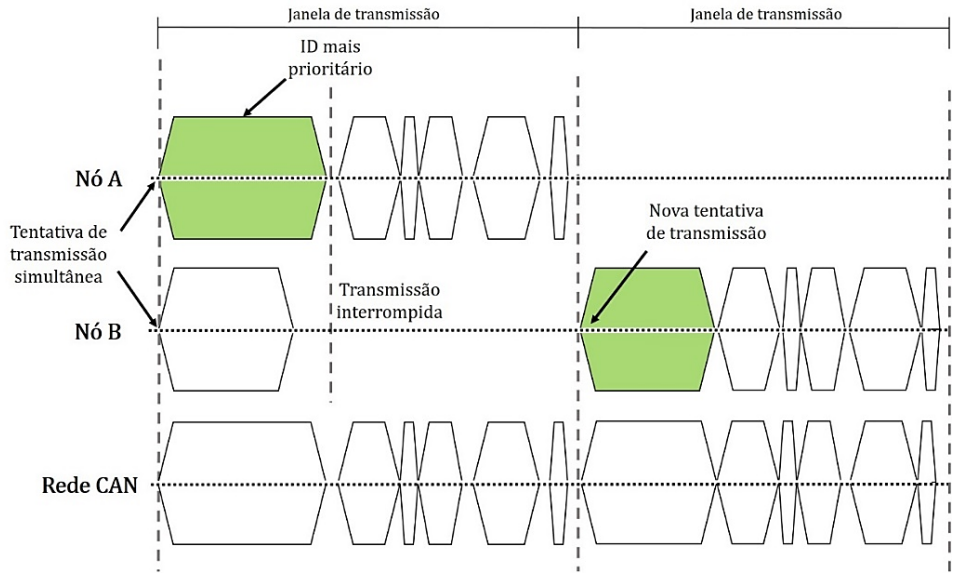
Figura 8. Estado lógico invertido do barramento CAN.



Fonte: CORRIGAN (2016).

Por isso no bloco de dados da mensagem CAN existe um campo de ID. Este campo, como mencionado anteriormente, é o responsável pela definição da prioridade da mensagem: quanto mais dominante ou quanto maior a quantidade de zeros (“0“) no ID de uma mensagem, mais prioritária ela será, pois ocupará o barramento por mais tempo no início da transmissão do que as outras mensagens. Desta forma, os nós com prioridades mais baixas interrompem a transmissão e aguardam a próxima janela de disponibilidade de barramento, conforme Figura 9. Por exemplo, num veículo, mensagens de dados do motor e sistema de controle de frenagem sempre terão ID’s mais prioritários do que sistema de sinalização e iluminação, uma vez que são vitais e estão diretamente relacionados à segurança dos passageiros (CORRIGAN, 2016).

Figura 9. Exemplo de arbitragem de rede entre dois nós.



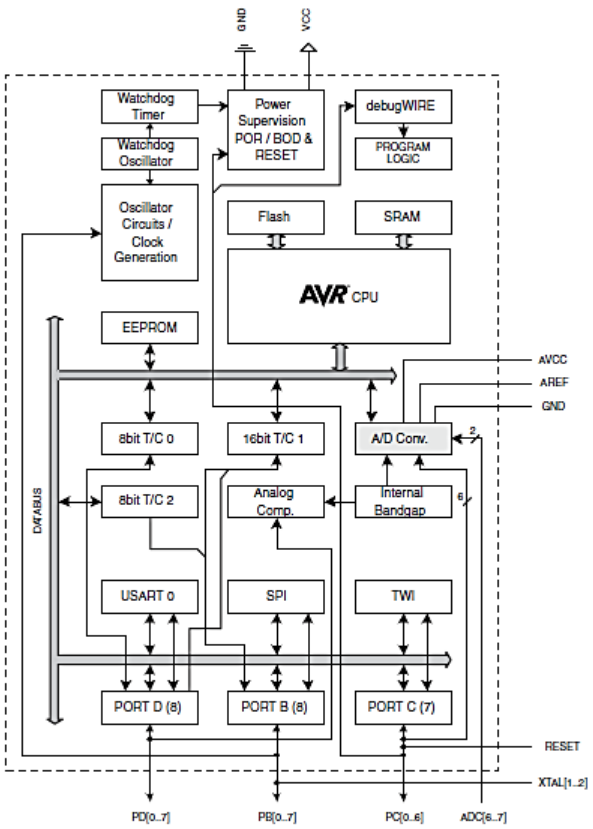
Fonte: Autoria própria.

2.3 MICROCONTROLADORES

Segundo (CONCEIÇÃO, 2012), a partir da década de 1970, vivemos a quinta revolução tecnológica, definida pela difusão da eletrônica e das tecnologias de comunicação e informação, possível graças à evolução e barateamento dos elementos semicondutores (diodos e transistores), principais dispositivos encontrados nos circuitos integrados (CI's). Essa escalada tecnológica possibilitou a criação dos microcontroladores, dispositivos mais compactos, versáteis e poderosos do ponto de vista do processamento de sinais e aplicabilidade.

Segundo Penido e Trindade (2013), os microcontroladores são conjuntos compostos por microprocessador, memórias (RAM, ROM, EEPROM e/ou *Flash*) e alguns periféricos (conversores A/D, contadores, geradores de *clock*, etc.) que conferem autonomia ao conjunto em termos de interação eletrônica, além da possibilidade de programação das funções por meio de interfaces externas, vide Figura 10.

Figura 10. Diagrama de blocos de um microcontrolador Atmel, família ATmega 8-bits.



Fonte: ATMEL (2013).

A possibilidade de programação é uma das principais vantagens do uso de microcontroladores e, de forma resumida, é a definição de como serão tratadas as informações que entram neles, por meio de dispositivos ditos sensores, o gerenciamento destas informações, e a saída determinada de acordo com a ação que se quer executar numa planta ou sistema, por meio de dispositivos conhecidos como atuadores. Ainda segundo Penido e Trindade (2013), os primeiros modelos de microcontroladores surgiram em 1977, tendo a Intel lançado a primeira linha comercial de baixo custo, a família MCS-48 – também chamados de microcomputadores. O modelo mais conhecido é o 8048, programado em linguagem *Assembly*, com 40 pinos, que à época era utilizado em computadores. Abaixo tem-se uma tabela com algumas das suas características, comparadas a um ATMega328P, modelo de microcontrolador atualmente utilizado em plataformas de prototipagem didáticas *open-source*:

Quadro 2 – Comparativo entre Intel 8048 e ATMega328P

Microcontrolador	**Intel 8048**	**ATMega328P**
CPU	8-bits	8-bits
Memória Programável	1 KByte (ROM)	1 KByte (EEPROM) + 32 KB Flash
RAM	64 Bytes	2 KBytes
I/O's	24	23

Fonte: ATMEL (2013); INTEL (1978).

Apesar das características que favorecem a sua utilização em sistemas de controle eletrônico, os microcontroladores possuem algumas restrições, como: capacidade limitada de execução de tarefas simultâneas e incapacidade de comandar dispositivos de potência diretamente; fazendo-se necessárias outras interfaces e/ou dispositivos complementares quando se deseja desenvolver uma solução mais completa para automação, a que chamamos de sistema embarcado.

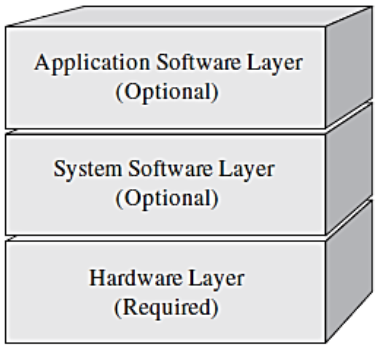
2.4 SISTEMAS EMBARCADOS

Embora o termo possua atualmente um sem número de definições, de forma simplificada, quando são acoplados determinados periféricos a um microcontrolador, e o mesmo é programado para atuar em uma aplicação específica em conjunto com tais dispositivos, chamamos esta solução de sistema embarcado. Por exemplo, um smartphone pode ser considerado um sistema embarcado, assim como uma lavadora

de roupas, uma impressora ou mesmo um sistema de gerenciamento eletrônico de um motor a combustão instalado em um veículo (NOERGAARD, 2013).

Segundo Noergaard (2013), os sistemas embarcados podem ser representados de forma simplificada por camadas, conforme Figura 11, e compartilham uma similaridade entre si em termos de arquitetura, que é o fato de todos os componentes do sistema interligarem-se na camada física (*hardware*), podendo o mesmo ocorrer nas camadas adicionais de *software* do sistema (programação) e de aplicação (interface com usuário). A representação visual das camadas facilita a compreensão do sistema como um todo, sendo utilizada como base para projetos de sistemas embarcados, conforme detalhado a seguir:

Figura 11. Modelo de sistemas embarcados.



Fonte: NOERGAARD (2013).

a) A camada de hardware compreende todos os elementos físicos do sistema, partindo do microcontrolador – em geral embutido em uma placa PCB (Placa de Circuito Impresso, do inglês *Printed Circuit Board*). São exemplos de elementos de hardware: circuitos integrados, elementos passivos (resistores, capacitores e indutores), memórias, fontes de alimentação, portas de entrada e saída (I/O's, do inglês *inputs and outputs*) e redes de comunicação entre os elementos (I^2C, PCI, etc.). Corresponde à primeira camada do sistema OSI, Figura 12.

b) A camada de software do sistema diz respeito a softwares de inicialização e gerenciamento do próprio hardware, sem os quais as tarefas mais simples do sistema embarcado não são possíveis. Estes recursos podem ser divididos em:

i. Bibliotecas de inicialização do hardware, também conhecidos como *device drivers*, que tornam possível o acesso ao mesmo por meio da programação de alto nível. O BIOS (*Basic Input/Output System*) de um computador é um exemplo desse tipo de *software*, sendo um conjunto de todos os *device drivers* necessários à inicialização da máquina.

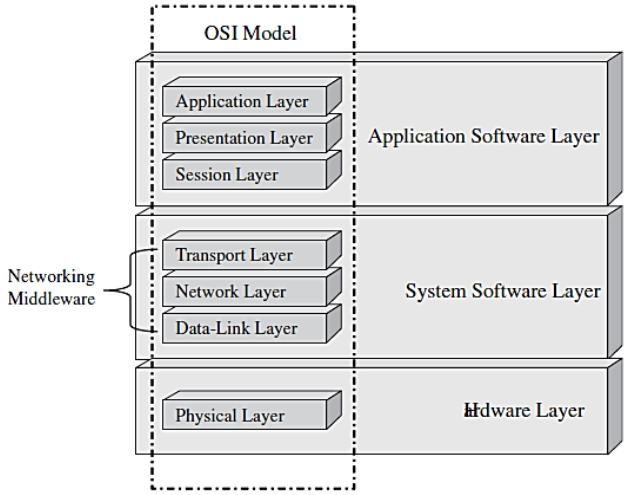
ii. *Middlewares*, que fazem a intermediação entre sistema operacional e *device drivers*. Por exemplo os drivers específicos de alguns dispositivos são tipos de *middlewares*; e

iii. Sistemas Operacionais (OS), que coordenam todas as atividades realizadas pelos dois grupos anteriores, garantindo mais eficiência e confiabilidade ao sistema. Este último grupo é opcional para os sistemas embarcados.

Esta camada corresponde às camadas dois a quatro do sistema OSI (enlace de dados, rede e transporte).

c) A camada de *software* de aplicação, que dependente dos *softwares* de sistema, e define que tipo de dispositivo um sistema embarcado é, uma vez que é na aplicação que se definem as funcionalidades do dispositivo e onde se tem a interação com o usuário. Corresponde às camadas cinco a sete do modelo OSI (sessão, apresentação e aplicação).

Figura 12. Comparativo entre os modelos de sistema embarcado e OSI.



Fonte: NOERGAARD (2013).

2.5 PLATAFORMAS DE PROTOTIPAGEM

O desenvolvimento de dispositivos e soluções na área da eletrônica e da automação – sistemas embarcados – tem sido um dos principais focos tecnológicos da atualidade. Uma vez que o *hardware* (camada física) é sempre o ponto de partida de um projeto eletrônico, e que são necessárias algumas funcionalidades em termos de *software* de sistema e de aplicação comuns a todos os projetos, surgiram algumas soluções pré-formatadas, a fim de acelerar e simplificar o desenvolvimento de sistemas embarcados. Estas soluções são conhecidas como plataformas de prototipagem eletrônica.

A primeira solução do gênero data de 1992. A *Parallax Inc.* (empresa norte americana de tecnologia, fundada em 1987) lançou o módulo *BASIC Stamp*: um microcontrolador de 4 MHz, programável em linguagem BASIC através de porta serial, embarcado em uma PCB com memória interna (16 *Bytes* de RAM e 256 *Bytes* de *EEPROM*), um regulador de tensão de 5 V e 16 portas de entrada e saída (LINDSAY, 2005). Em comparação com as tecnologias disponíveis à época – principal delas os microcontroladores PIC –, o uso do *BASIC Stamp* era significativamente mais simples, pois uma vez familiarizado com as instruções em BASIC era possível programar a mesma plataforma para diversas aplicações. As principais desvantagens residiam no preço do dispositivo – atualmente U$49.00, ainda hoje mais caro que algumas das plataformas mais populares – e a limitação em termos de versatilidade da programação por linhas de instrução *BASIC*.

Atualmente existem diversos fabricantes, modelos e versões de plataformas de prototipagem eletrônica, segue os principais no quadro abaixo:

Quadro 3 – Plataformas de prototipagem eletrônica.

Fabricante/Desenvolvedor	Principal modelo	Ano de lançamento
Parallax	BASIC Stamp	1992
Arduino Project	Arduino UNO	2005
Texas Instruments/Digi-Key	BeagleBone	2008
CubieBoard	CubieBoard1	2012
Raspberry Pi Foundation	Raspberry Pi V3	2012
Espressif Systems	ESP8266	2014

Fonte: CURVELLO (2014); ARDUINO (2018); TEXAS INSTRUMENTS (2018); RASPBERRY PI FOUNDATION (2018); ESPRESSIF SYSTEMS (2018); LINDSAY (2005).

O uso de plataformas de prototipagem evoluiu muito desde o *BASIC Stamp*, principalmente no que diz respeito à programação do microcontrolador. Anteriormente, os *softwares* de programação eram proprietários do próprio fabricante do microcontrolador, sendo necessário adquirir a licença de *software* – a um custo elevado, diga-se de passagem – para operacionalizar o *hardware*. Este modelo dificultava o desenvolvimento de projetos por parte de estudantes de escolas com pouca infraestrutura, “hobistas”, iniciantes e/ou entusiastas da eletrônica.

2.5.1 Arduino

Com o advento de plataformas de prototipagem cada vez mais acessíveis, criaram-se diversos ambientes de desenvolvimento integrado (IDE, do inglês *Integrated Development Environment*), *softwares* que podem rodar em computadores pessoais e até mesmo em *smartphones*, onde são escritos os códigos de programação de sistemas embarcados. Este novo paradigma começou com um projeto de 2002 chamado *Arduino Project*, composto por uma equipe de alunos e professores do Instituto de Design de Interação de Ivrea (IDII), na Itália, e tinha como desafio sanar três dores principais: a) baratear o acesso à plataformas de prototipagem, tanto no custo de aquisição do *hardware* quanto na licença de *software*; b) facilitar a programação da plataforma, com linguagem e interface mais intuitivas, que demandassem menos esforço do desenvolvedor; c) acelerar a prototipação eletrônica de sistemas embarcados que conectassem o mundo físico ao virtual, de forma mais didática. O projeto culminou no lançamento, em 2005, de uma solução completa: uma plataforma de prototipagem de baixo custo chamada *Arduino*, um modelo de programação mais intuitivo e simplificado e uma IDE *open-source* (disponibilizada gratuita e abertamente) (BARRAGÁN, 2016; CIRCUITS TODAY, 2018; ARDUINO, 2018).

Existem diversos modelos de placas Arduino, sendo o mais difundido deles o Arduino UNO, que já está na terceira versão, conforme Tabela 1. A sua estrutura de *hardware* é funcional e simplificada, possibilitando o uso desde projetos de circuitos eletrônicos simples – com resistores, capacitores, *push-buttons*, diodos emissores de luz (LED), *protoboard*, etc. – até aplicações mais complexas, usando módulos e *shields*.

Tabela 1 – Especificações técnicas do Arduino UNO Rev3:

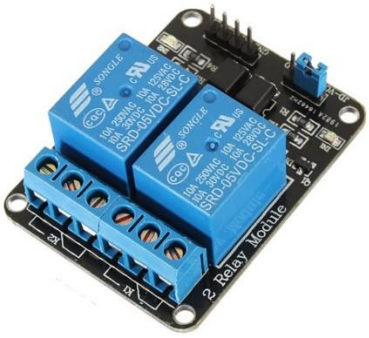
Microcontrolador	ATmega328P
Tensão de operação	5V
Tensão de alimentação (recomendada)	7-12V
Tensão máxima	6-20V
Portas digitais (I/O's)	14 (6 podem operar como saída PWM)
Portas de entrada analógica	6
Corrente contínua de I/O (máx.)	20 mA
Memória Flash	32 KB
SRAM	2 KB
EEPROM	1 KB
Velocidade de clock	16 MHz
Dimensões	68,6 x 53,4 mm
Peso	25 g

Fonte: adapt. de ARDUINO, (2018).

2.5.2 Módulos de Prototipagem

Módulos de prototipagem – ou somente módulos – são conjuntos de componentes montados sobre placa PCB destinados a expandir as possibilidades de uso das plataformas de prototipagem. Existem muitos módulos diferentes, cada um com um circuito específico: relés, *drivers* de motores, conjuntos RFID (do inglês, *Radio-Frequency IDentification*), antenas GPS (do inglês, *Global Positioning System*), circuitos de *clock*, acelerômetros, sensores LDR (do inglês, *Light-Dependent Resistor*), emissores de *laser*, detectores de tensão, sensores de umidade, sensores de gás, etc. Por exemplo, um módulo relé de dois canais (modelo SRD-05VDC-SL-C) contém: dois relés comandados por 5 V a baixa corrente – com capacidade para acionar dispositivos de até 12 A à 250 VAC –, montados em placa PCB com circuito de acionamento opto acoplado incluso, barramento de pinos adequados para ligação com plataformas de prototipagem e bornes dedicados à conexão com atuadores e/ou dispositivos de potência, vide Figura 13.

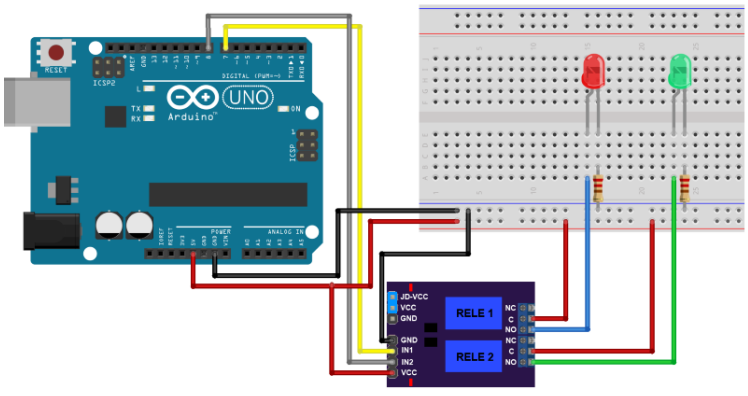
Figura 13. Módulo relé 5 V 2 canais (2 relés modelo SRD-05VDC-SL-C).



Fonte: FILIPEFLOP (2018).

Com este módulo é possível, por exemplo, utilizar o Arduino para comandar saídas de potência como um sistema de iluminação residencial, ou uma janela automatizada. Na Figura 14 tem-se um exemplo de circuito utilizando um Arduino, o módulo relé mencionado e dois LED's simbolizando os atuadores. Nota-se que a ligação elétrica é bem simplificada e também organizada, o que viabiliza o desenvolvimento de soluções embarcadas com relativa qualidade, mesmo no ambiente educacional.

Figura 14. Diagrama de um circuito de acionamento utilizando o módulo relé 2 canais.



Fonte: FILIPEFLOP (2018).

2.5.2.1 Módulo CAN Bus

Dentre os muitos módulos disponíveis no mercado, figuram também interfaces de rede, como é o caso do módulo CAN Bus, que possui embutido nele um controlador MCP2515 e um transceptor TJA1050, dentre outros componentes, vide Figura 15.

Figura 15. Módulo CAN Bus.

Fonte: FILIPEFLOP (2018).

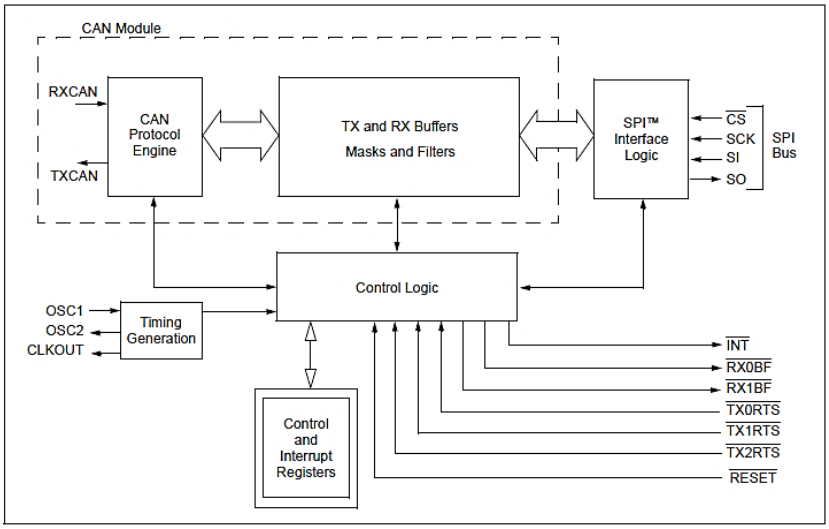
Segundo a Microchip (2005), o MCP2515 é um controlador que implementa as especificações do protocolo CAN 2.0B, podendo receber ou transmitir mensagens numa taxa de 1 Mbps, tanto no formato de identificador padrão (11 bits) quanto no estendido (29 bits), através de comunicação SPI (do inglês, *Serial Peripheral Interface*, numa interpretação livre, Interface para Periféricos em Série). De forma simplificada, o MCP 2515 é composto por três blocos principais, conforme Figura 16:

a) Módulo CAN – gerencia todas as funções de transmissão e recebimento de mensagens da rede CAN, através de *buffers*, filtros e máscaras de protocolo.

b) Lógica de controle – determina as configurações e a operação do Módulo CAN, realizando a interface com os outros blocos e gerenciando as informações transmitidas.

c) Protocolo SPI – faz a interface entre o controlador e os dispositivos externos, sendo responsável pela leitura e escrita da comunicação através de comandos SPI padrão.

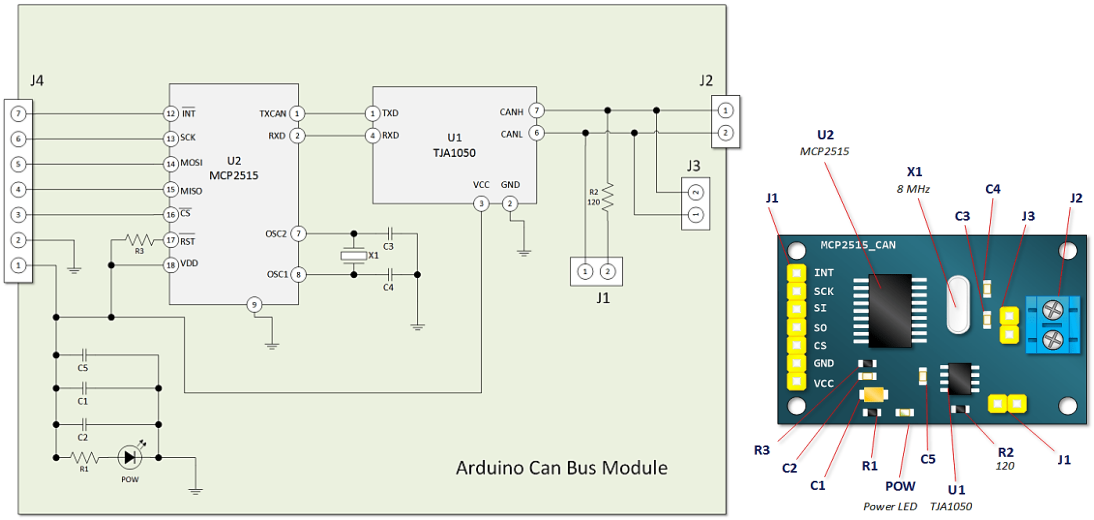
Figura 16. Diagrama de blocos do controlador MCP2515.



Fonte: MICROCHIP (2005).

Para que seja possível a comunicação do MCP2515 com a rede física CAN propriamente dita (par trançado), é necessário inserir no circuito um transceptor (do inglês, *transceiver*), que interprete as mensagens que trafegam na rede para o controlador e vice-versa. No módulo CAN Bus o elemento que realiza esta tarefa é o transceptor TJA1050, da *Philips* (Figura 17). Com esse módulo é possível desenvolver uma plataforma de comunicação CAN utilizando uma plataforma de prototipagem e alguns conhecimentos de programação (BENCH, [S.d.]; PHILIPS, 2003).

Figura 17. Diagrama elétrico e identificação dos componentes do Módulo CAN BUS.



Fonte: BENCH [S.d.].

2.5.3 Shields

Ainda que não exista uma literatura específica distinguindo módulos de *shields*, estes últimos são similares aos módulos no propósito, porém construídos para serem acoplados sobre uma determinada plataforma de prototipagem (daí o termo *shield*, do inglês escudo), como uma extensão da mesma, podendo ser empilháveis e dispensando barramentos ou chicotes elétricos. Também possuem aplicações específicas e, em geral, já agregam circuitos com soluções mais completas, que atendem à sistemas embarcados específicos, conforme alguns exemplos abaixo:

1. Shield LCD para Arduino – Basicamente um display de LCD (do inglês, *Liquid Crystal Display*), com 6 botões de navegação. A vantagem do uso desse *shield* é a praticidade no acoplamento com o Arduino, dispensando o uso de protoboard, jumpers e soldagem de componentes. Além disso, os botões de navegação são endereçados à uma única porta analógica, operando com níveis de tensão diferentes (Figura 18).

Figura 18. Shield Display LCD para Arduino.

Fonte: FILIPEFLOP (2018).

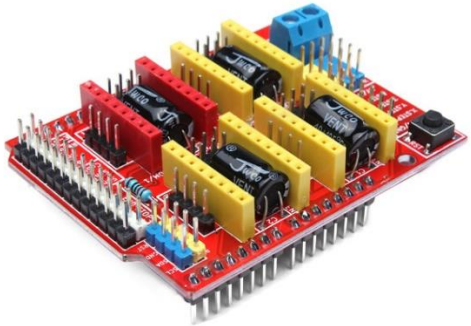
2. Shield *driver* de motor L293D (para Arduino) – *driver* de motores, que possibilita o controle de 4 motores DC, 2 servo-motores ou 2 motores de passo, com tensão de até 16 V e corrente máxima por canal de 600 mA. O *shield* possui controlador próprio, o L293D, que faz a interface entre o Arduino e os motores nele acoplados (Figura 19).

Figura 19. Shield *driver* de motor L293D.

Fonte: FILIPEFLOP (2018).

3. Shield Impressora 3D V3 – pode ser utilizado para se desenvolver uma impressora 3D tendo o Arduino como controlador ou até mesmo uma máquina CNC. Feito para operar com *drivers* de motor de passo modelo A4988, pode controlar até quatro motores de passo, permite alimentação externa de até 36 V e já possui os pinos para controle de sensores (Figura 20).

Figura 20. Shield Impressora 3D V3.



Fonte: FILIPEFLOP (2018).

2.6 ADAPTAÇÕES AUTOMOTIVAS PARA DEFICIENTES

De acordo com os dados do Censo 2010, realizado pelo IBGE (INSTITUTO..., 2010), cerca de 7% da população brasileira apresenta deficiência exclusivamente do tipo motora, dificultando total ou parcialmente a sua movimentação. Esta parcela da população, assim como todo o restante, necessita se locomover de um espaço a outro, a fim de realizar as suas atividades diárias, como trabalhar, estudar, socializar, prestar

e consumir serviços de um modo geral. Para tanto se faz necessária a utilização de algum tipo de transporte, seja ele coletivo ou particular, ambos necessitando de adaptações específicas para o correto acesso e acondicionamento de deficientes motores, sendo estes dois elementos problemas recorrentes no âmbito da mobilidade urbana.

De acordo com Garcia (2009), para cada tipo de dificuldade ou lesão motora apresentada pelo deficiente, existe um tipo de adaptação necessária, a fim de que a condução do veículo seja facilitada. Devido à grande multiplicidade de deficiências motoras e adaptações possíveis – cerca de 400 opções -, podemos classificar as adaptações em três categorias: Condutibilidade (*Driving*), Acondicionamento (*Stowage*) e Acessibilidade (*Access*) (MOTABILITY, 2018). Ainda segundo Motability (2018), tais categorias podem ser subdivididas em grupos de dispositivos, de acordo com a função auxiliar conforme abaixo:

1. Condutibilidade:
    a. Controles manuais (*Hand controls*);
    b. Aceleradores eletrônicos manuais (*Electronic accelerators*);
    c. Aceleradores para pé esquerdo (*Left foot accelerators*);
    d. Modificações de pedais (*Pedal modifications*);
    e. Auxílios de esterçamento (*Steering aids*);
    f. Dispositivos de controle remoto (*Remote controls devices*);
2. Acondicionamento
    a. Guindaste de porta-malas (*Car boot hoist*);
    b. Acondicionador e guindaste de teto (*Rooftop stowage*);
3. Acessibilidade
    a. Plataformas de transposição (*Transfer plates*);
    b. Guindaste elétrico pessoal (*Electric person hoist*);
    c. Assentos rotatórios (*Swivel seats*);

2.6.1 Dispositivos de Controle Remoto

Direcionando-se o foco para a categoria de condutibilidade, mais especificamente nas soluções para controles auxiliares, também chamados de dispositivos de controle remoto, em termos de mercado global, tem-se pouca diversidade de soluções. A maioria dos fabricantes desenvolvem um dispositivo muito similar, tanto funcional, tecnológica quanto esteticamente falando. O dispositivo consiste, basicamente, de um controle remoto por sinal infravermelho ou de rádio, com suporte para fixação no volante, que também funciona como adaptação para facilitar o giro do mesmo (pomo). O controle comunica-se com uma central microcontrolada adaptada ao chicote do veículo, e possui botões programados para substituir os comandos realizados pelas alavancas da coluna de direção (GUIDOSIMPLEX, 2018). Abaixo segue breve descrição de produtos do gênero e seus respectivos fabricantes:

- Asaflex – Comandos elétricos no volante, modelo Cce Asaflex. Possui empunhadura de corpo único, já servindo como pomo de giro do volante na mesma peça, com botões de função na parte superior para fácil alcance com o polegar. É acoplado ao volante com engate rápido, que facilita a remoção do mesmo (Figura 21).

Figura 21. Asaflex CCe com suporte de fixação no volante.

Fonte: HELIO (2018).

- Cavenaghi – Controle de comandos, modelo ERGON CE. Tem as mesmas características do produto acima citado, com leve diferença no arranjo dos botões e formato da empunhadura, vide Figura 22.

Figura 22. Controle de comandos ERGON CE.

Fonte: CAVENAGHI (2018).

- Guidosimplex – Telecomando a infrarossi (Telecomando a infravermelho), modelo 1096. Tem as mesmas características dos produtos listados acima, também com leve diferença no arranjo dos botões e formato da empunhadura (Figura 23).

Figura 23. Controle sem fio Guidosimplex (Telecomando a Infrarossi, modelo 1096).

Fonte: GUIDOSIMPLEX (2018).

- KIVI – *Transmitter on the Wheel* (Transmissor no volante), modelo PV3000. Acoplado ao volante com adaptador do tipo pomo, podendo ser removido separadamente do mesmo (Figura 24).

Figura 24. PV3000 da KIVI.

Fonte: KIVI (2018).

- Bever – *Smartsteer remote control*. Dispositivo de controle remoto desenvolvido para trabalhar em conjunto com outro do mesmo fabricante (Smartcontrol), já interligado à rede CAN (Figura 25).

Figura 25. Smartsteer da Bever.

Fonte: BEVER [S.d.].

- Lodgesons – *Keypad controls*, modelo R213. Este modelo possui várias versões, de acordo com a quantidade de funções programáveis, e pode ser acoplado ao volante com pomo (Figura 26) ou com manopla tipo pirulito (*lollipop*). Assim como a Bever, o dispositivo da Lodgesons

já realiza a comunicação diretamente coma rede veicular (CAN), facilitando e aumentando a confiabilidade da instalação.

Figura 26. R213 da Lodgesons, com adaptador tipo pomo.

Fonte: LODGESONS (2018).

Embora esta solução seja amplamente utilizada, e de operação relativamente facilitada, ainda persistem duas possibilidades de melhoria: uma relacionada a operação do acessório (por parte do usuário), outra, de caráter técnico, relacionada à conexão e interface do dispositivo com os sistemas veiculares (por parte do desenvolvedor do dispositivo).

Em relação à operação do acessório, os dispositivos de controle remoto são requisitados a fim de concentrar os comandos dos sistemas de sinalização, iluminação e conforto, presentes nas alavancas da coluna de direção, ao alcance de uma única mão, a mesma utilizada para operar o volante. Desta forma, a outra mão fica livre para controlar a transmissão e/ou os sistemas de pedais adaptados. Porém, apesar da contribuição no sentido de tornar a condução mais confortável e acessível, ainda se faz necessária a utilização da mão e dos dedos para operar o dispositivo, o que se torna ineficaz em caso de amputações ou deformação das mãos do condutor, ou mesmo tetraplegia. Portanto, faz-se necessária uma solução que elimine também a utilização constante das mãos, possibilitando ao condutor deficiente controlar os mesmos sistemas. A Paravan GmbH (2018), fabricante alemã de soluções em paramobilidade, desenvolveu um sistema de controle de voz, conforme Figura 27, para as “funções secundárias” do veículo, suprindo esta necessidade, sendo este o único dispositivo do gênero. Conforme apresentado por Paravan GmbH (2015) no vídeo demonstrativo do produto, o sistema possibilita configurar comandos de voz

para mais de cem funções diferentes, incluindo a partida do motor do veículo e esterçamento do volante de direção, demandando a instalação de dispositivos e sistemas adicionais ao de comando de voz.

Figura 27. Sistema de comando de voz da Paravan.

Fonte: PARAVAN (2018).

No que diz respeito a conexão dos sistemas de controle remoto com o veículo, a instalação atual da maioria dos dispositivos se dá por meio de ligação elétrica paralela aos comandos originais do veículo. Em resumo: os fios da central são crimpados ou soldados paralelamente aos fios originais do carro que alimentam os dispositivos que se pretende acionar. Desta forma, a interface de controle do sistema está ligada fisicamente ao chicote original do veículo, demandando adaptações e conexões que podem causar interferências, mau funcionamento ou mesmo a falha dos sistemas originais. Alguns dos fabricantes anteriormente citados – Lodgesons e Bever, por exemplo – já oferecem a possibilidade de comunicação do sistema via rede veicular (CAN), evitando os problemas acima citados. Porém esta solução é ofertada separadamente, como um complemento do sistema de controle remoto, o que encarece a sua instalação (BEVER, [S.d.]; LODGESONS, 2018).

3 DESENVOLVIMENTO DO TEMA

Uma vez levantadas as informações necessárias para compreender os sistemas com os quais se pretende atuar, a confecção do projeto será apresentada nos subcapítulos abaixo, que descrevem os materiais e componentes utilizados, bem como os sistemas e soluções desenvolvidas.

3.1 MATERIAIS E COMPONENTES

Para a construção e operacionalização do protótipo foi necessário montar também uma plataforma de testes, neste caso uma bancada de rede CAN, contendo componentes automotivos organizados de forma a substituir ou simular as condições de funcionamento de um veículo. Além do sistema em si e da bancada, também foi desenvolvido um aplicativo para dispositivos móveis, que possibilite utilizar *smartphones* e/ou *tablets* como periférico de entrada de comando de voz via conexão sem fio (*bluetooth*).

3.2 BANCADA DE TESTES

A bancada serve como plataforma de aplicação e teste do controle por voz, agregando os componentes automotivos com comunicação via rede CAN, simulando as condições de operação e funcionalidade encontradas em um veículo de passeio. Os componentes adquiridos são de um carro do fabricante Peugeot, modelo 308, que possui arquitetura eletroeletrônica multiplexada chamada de *Full* CAN, com módulo central de gerenciamento (BSI, *Boîtier de Servitude Intelligent*, Central de Gerenciamento Inteligente), responsável por toda a comunicação entre os sistemas veiculares, sendo esta dividida em três redes distintas: CAN HS I/S, CAN LS CONF e CAN LS CAR (Figura 28).

Figura 28. Vista frontal da BSI do Peugeot 308 (observam-se os conectores e os fusíveis de proteção dos circuitos internos).

Fonte: Autoria própria.

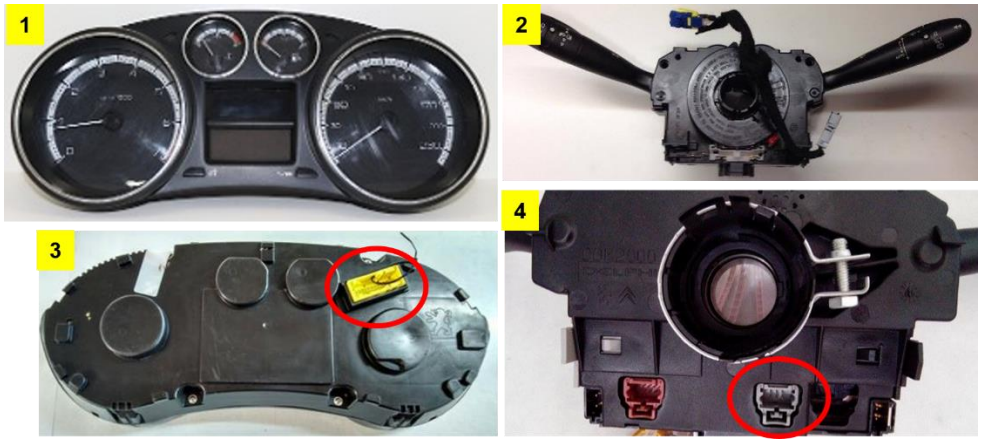
Conforme se observa, há diversos conectores na parte frontal da BSI, através dos quais é possível interligar os diversos módulos à mesma, tanto em nível de rede quanto em nível de sinal elétrico, ou mesmo alimentação. As três redes gerenciadas pela BSI possuem funcionalidade e nível de importância diferentes no veículo, conforme abaixo:

a) CAN HS (*high-speed*) I/S (*Intersystems*): rede de alta velocidade (500 Kbps), responsável pela ligação entre a BSI e os módulos de gerenciamento do conjunto motopropulsor, ou seja, motor e transmissão (calculadores dinâmicos);
b) CAN LS (*low-speed*) CONF (*confort*): rede de baixa velocidade (125 Kbps), que interliga a BSI aos módulos de conforto, como sistema multimídia, condicionador de ar, acionamento elétricos dos vidros, etc;
c) CAN LS (*low-speed*) CAR (*carrosserie*): rede de baixa velocidade (125 Kbps), que estabelece a ligação entre a BSI e os módulos de gerenciamento do chassi, como sinalização e iluminação, ABS (*Anti-lock Brake System*, Sistema de Frenagem Anti-bloqueio), direção eletricamente assistida, etc.

Dentre os módulos controlados pela BSI constam o painel de instrumentos (identificado pelo fabricante como 0004) e o conjunto de alavancas de comando do painel (identificado pelo fabricante como CV00). Ambos se comunicam com a BSI via

rede CAN, fazem parte da rede da carroceria (CAN LS CAR), e constituem a bancada de testes (Figura 29).

Figura 29. Painel de instrumentos (imagem 1) e conjunto de alavancas do painel (imagem 2). Detalhe dos conectores na parte traseira dos componentes (imagens 3 e 4).



Fonte: Autoria própria.

Além dos módulos acima mencionados, foram adicionados alguns elementos de entrada e saída de sinal e comunicação CAN, a fim de possibilitar o teste e validação da comunicação CAN entre os módulos, bem como a complementação de módulos e dispositivos faltantes. Os dispositivos adicionados foram:

1) Interruptor de portas universal – a fim de que seja possível verificar o sinal de porta aberta e porta fechada via CAN (Figura 30);

Figura 30. Interruptor de porta automotivo universal.

Fonte: Autoria própria.

2) Conector aterrado – substitui o dispositivo do cinto de segurança do motorista, a fim de que seja possível monitorar via CAN e via painel de instrumentos a transmissão da mensagem (Figura 31);

Figura 31. Conector simulando cinto de segurança.

Fonte: Autoria própria.

3) Sensor de nível de combustível – que possibilita a variação do indicador de nível de combustível no painel, sinal este também enviado via rede CAN (Figura 32);

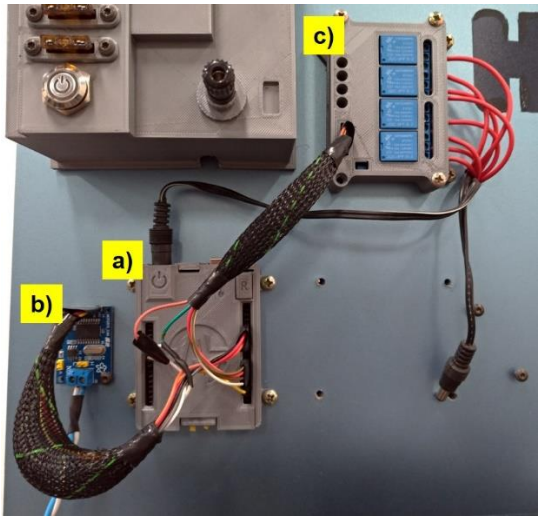
Figura 32. Sensor de nível de combustível linha Peugeot.

Fonte: Autoria própria.

4) Módulo de gerenciamento de sinalização e iluminação – conjunto de aquisição de sinal CAN que tem como função comandar o sistema de

sinalização e iluminação da bancada a partir das mensagens CAN recebidas da BSI (Figura 33).

Figura 33. Módulo de gerenciamento de sinalização e iluminação: a) Arduino UNO encapsulado; b) Módulo CAN Bus MCP2515; c) Módulo relé 4 canais.

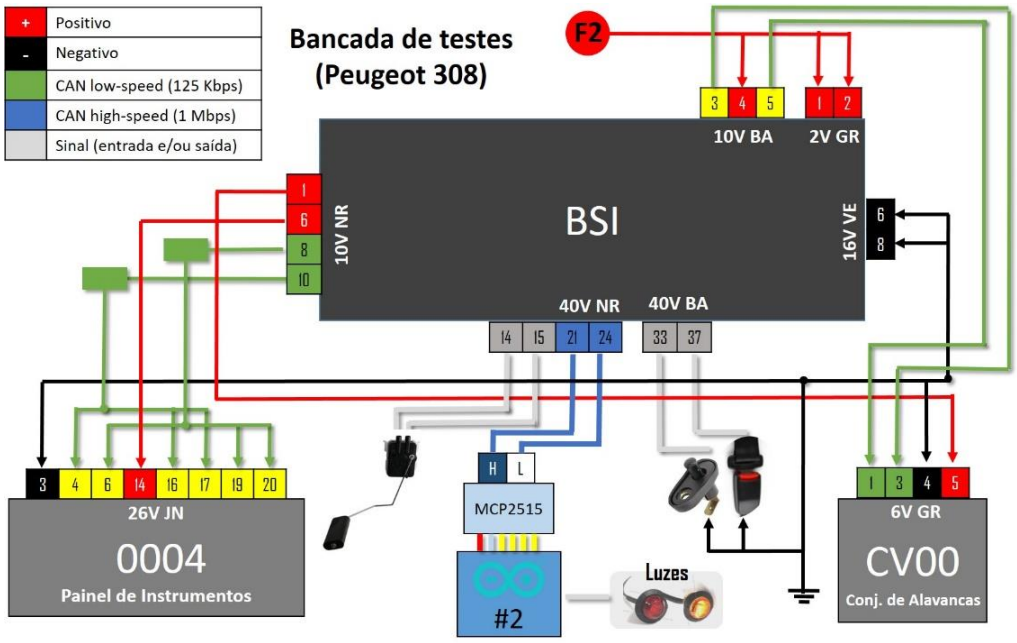


Fonte: Autoria própria.

Uma vez interligados todos os dispositivos e componentes, observa-se que a BSI centraliza todos os sinais da plataforma veicular, quer seja sinais analógicos de sensores e interruptores, quer seja linhas de comunicação via protocolo CAN, ou mesmo alimentação elétrica (Figura 34). Na figura abaixo, as conexões com a BSI estão organizadas de acordo com:

- Os conectores – identificados pelo código do fabricante, que contém o número de vias (pinos) do conector e a cor numa abreviação do francês. Por exemplo, o conector 10V NR, corresponde ao de 10 vias preto (NR, do francês *noir*), e assim por diante. Estas identificações estão marcadas na própria BSI ao lado de cada conector.
- A função – conforme a tabela no canto superior esquerdo da figura, as conexões elétricas apresentadas podem ser de alimentação (positivo ou negativo), de sinal elétrico diretamente de sensores e interruptores, ou de comunicação (CAN de baixa ou de alta velocidade).

Figura 34. Diagrama da interligação dos componentes da bancada com a BSI.



Fonte: Cueto (2018).

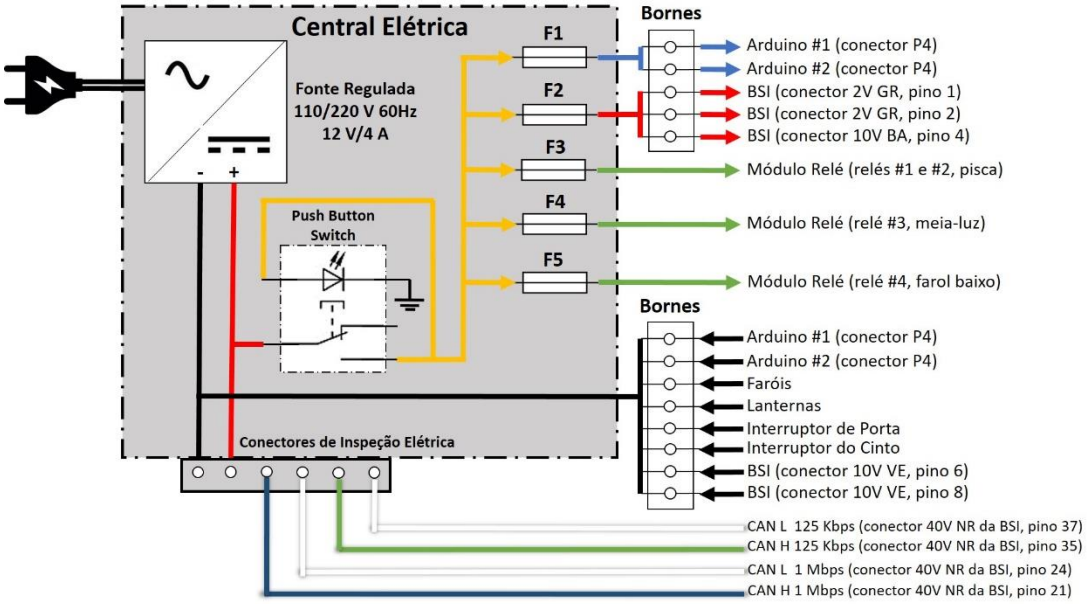
Uma vez que a intenção da bancada é simular um ambiente automotivo, onde há dispositivos que gerenciam a alimentação elétrica dos sistemas e também protegem contra curtos-circuitos e sobrecargas, fez-se necessário agregar ao conjunto uma fonte de alimentação de corrente contínua e uma central de distribuição elétrica. A fonte utilizada é um modelo compacto bivolt, com saída chaveada regulável de 5 a 12 V de tensão e corrente máxima de 10 A, que substitui a bateria. Para controle da alimentação dos diversos dispositivos e sistemas da bancada, foi confeccionada uma central elétrica, contando com um interruptor geral de alimentação, com saídas protegidas por fusíveis automotivos tipo lâmina (Figura 35).

Figura 35. Central de distribuição elétrica. Destacam-se o interruptor geral, os fusíveis de proteção e os conectores (bornes) de inspeção elétrica.

Fonte: Autoria própria.

Uma outra função da central, conforme se observa no diagrama elétrico, é permitir o monitoramento, por meio de conectores de inspeção elétrica, da tensão da fonte e também dos sinais de rede CAN nas linhas de baixa e de alta velocidade, sem necessidade de intervir no chicote da bancada (Figura 36).

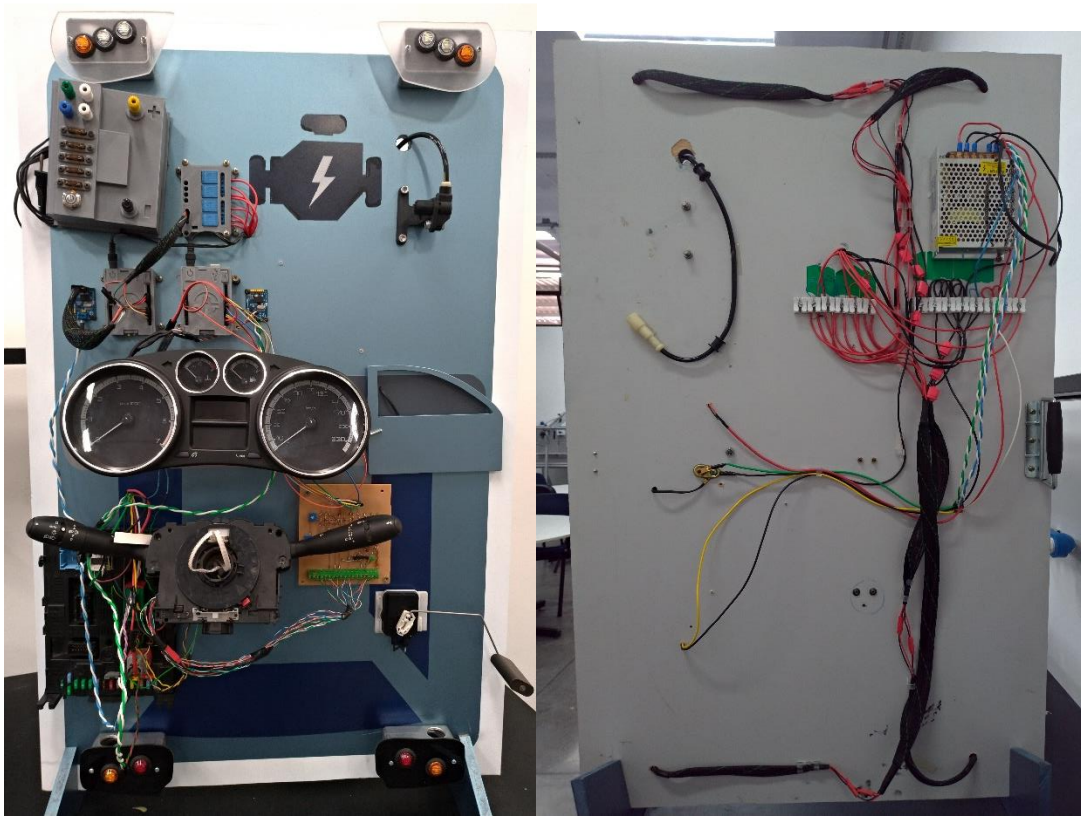
Figura 36. Diagrama da central de distribuição elétrica.



Fonte: Cueto, 2018.

Uma vez tendo em mãos os componentes automotivos e demais dispositivos, construiu-se a bancada, tendo como base para o posicionamento físico dos componentes as posições originais no carro. As conexões elétricas foram realizadas, na medida do possível todas na parte traseira da bancada, a fim de manter uma interface funcional mais limpa e intuitiva. Além disso, foram confeccionados suportes e encapsulamentos em manufatura aditiva (impressão 3D) para melhor acondicionar e organizar os componentes (Figura 37).

Figura 37. Bancada de teste finalizada, frente (esq.) e verso (dir.).



Fonte: Autoria própria.

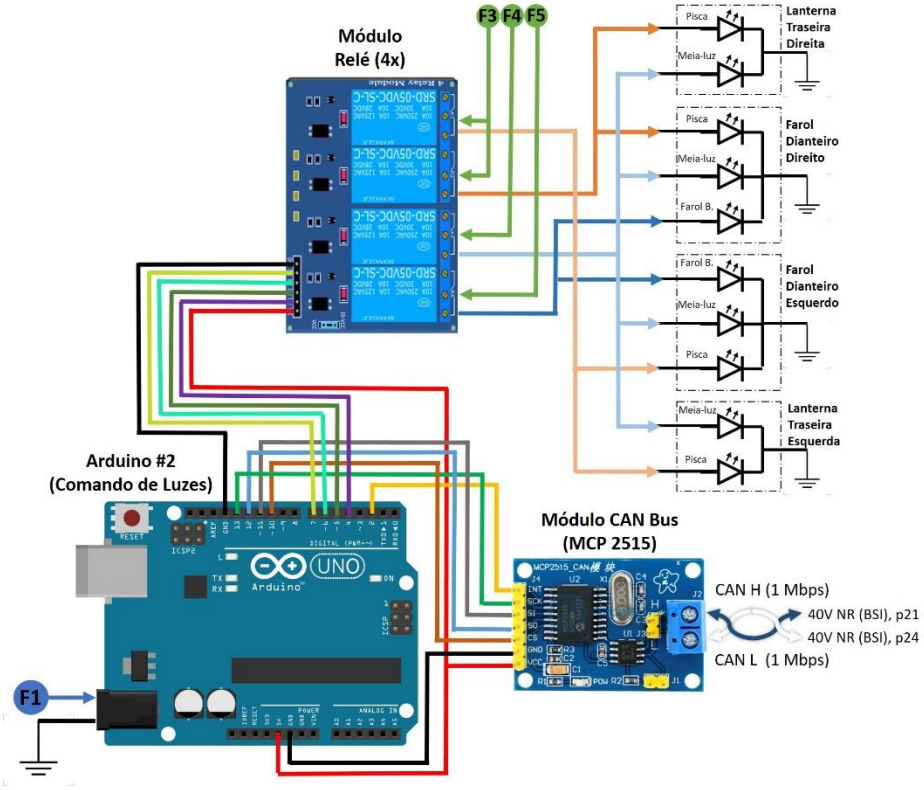
3.2.1 Módulo de Gerenciamento de Sinalização e Iluminação

O conjunto de aquisição de sinal CAN, anteriormente referido, é um sistema embarcado desenvolvido com a finalidade de substituir um dispositivo integrante da plataforma Peugeot 308, intitulado módulo PSF1 (*Powertrain Systems Fusebox* 1, caixa de fusíveis do cofre do motor). Este módulo atua no veículo como uma extensão da BSI dentro do cofre do motor, gerenciando os sistemas de sinalização, iluminação e grupo motopropulsor, dentre outros. Como não foi possível adquirir este

componente, e não há saídas analógicas e de potência para o sistema de sinalização e iluminação a partir da BSI, a solução proposta foi confeccionar um sistema que fosse capaz de substituí-lo (Figura 38), utilizando-se:

- Plataforma de prototipagem eletrônica Arduino UNO;
- Módulo CAN Bus MCP2515;
- Módulo relé 4 canais.
- Conjunto de LED's de sinalização encapsulados (substituindo os faróis e lanternas);

Figura 38. Módulo de gerenciamento de sinalização e iluminação.



Fonte: Autoria própria.

3.2.1.1 Farejamento *(Sniffing)* dos Dados CAN

A atuação do módulo de controle é condicionada ao reconhecimento, dentre todas as mensagens trafegando na rede CAN, de determinadas ID's e mensagens correspondentes aos comandos desejados. Como visto anteriormente, o protocolo CAN (normas ISO898 e ISO519-2) padroniza as camadas física, enlace de dados e aplicação. Ou seja, o formato das mensagens é padrão, porém a mensagem em si,

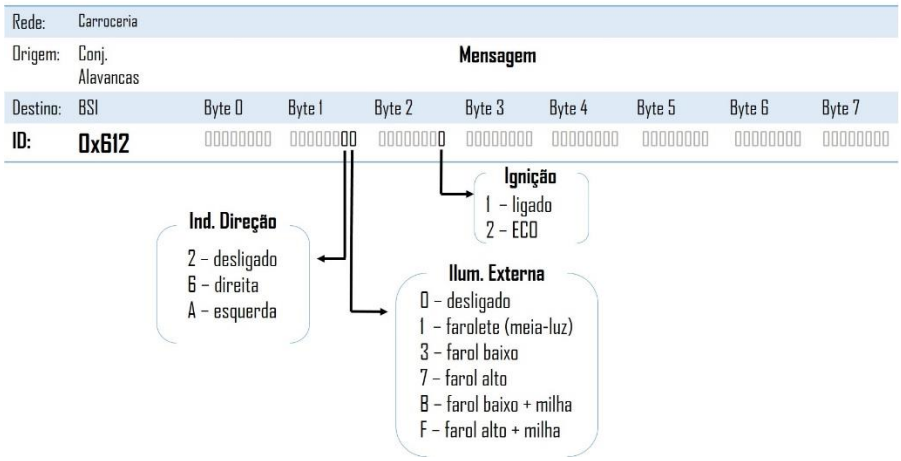
não. Quaisquer valores de ID's e mensagens podem ser utilizados para representar qualquer informação, dependendo do modelo do veículo, fabricante, desenvolvedor do sistema, etc.

Sendo assim, para que se possa descobrir quais dados correspondem a determinada atuação, realizou-se um *sniffing* da rede que, de forma resumida, constitui-se de um método de identificação por tentativa e erro ou varredura, conforme etapas abaixo:

1) O módulo CAN é acoplado à rede (num veículo), monitorando todos os dados que nela trafegam;
2) Realiza-se o acionamento que se deseja identificar, a fim de que se possa observar a variação da mensagem;
3) Uma vez identificada a(s) variação(ões), aplica-se um filtro na programação do monitoramento da rede do módulo CAN;
4) Realiza-se novamente o acionamento, para confirmar a variação da mensagem.

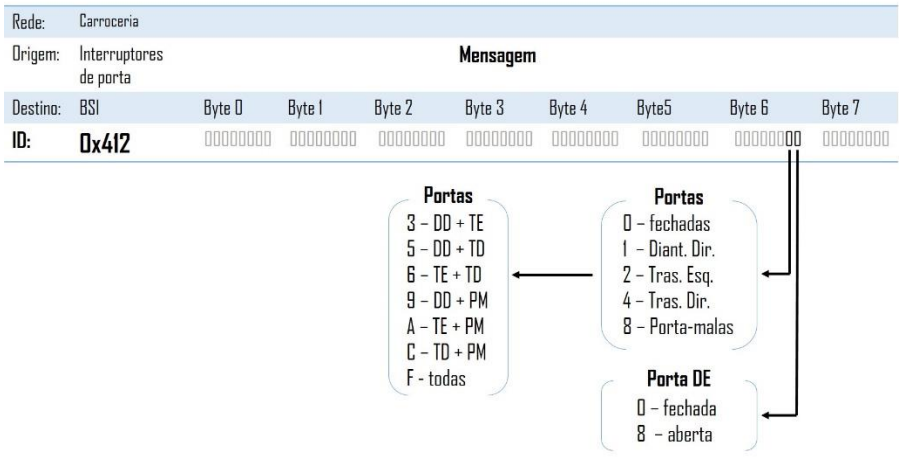
Para o caso da bancada de teste, foi realizado um *sniffing* num veículo Peugeot 308, a fim de identificar os ID's e mensagens correspondentes aos acionamentos: dos indicadores de direção, luzes, interruptores de porta e cinto de segurança, tendo sido obtidas as variações das mensagens nos ID's conforme Figuras 39 e 40.

Figura 39. Mensagens de indicadores de direção e luzes no ID 0x612.



Fonte: Cueto (2018).

Figura 40. Mensagens de interruptores de porta no ID 0x412.



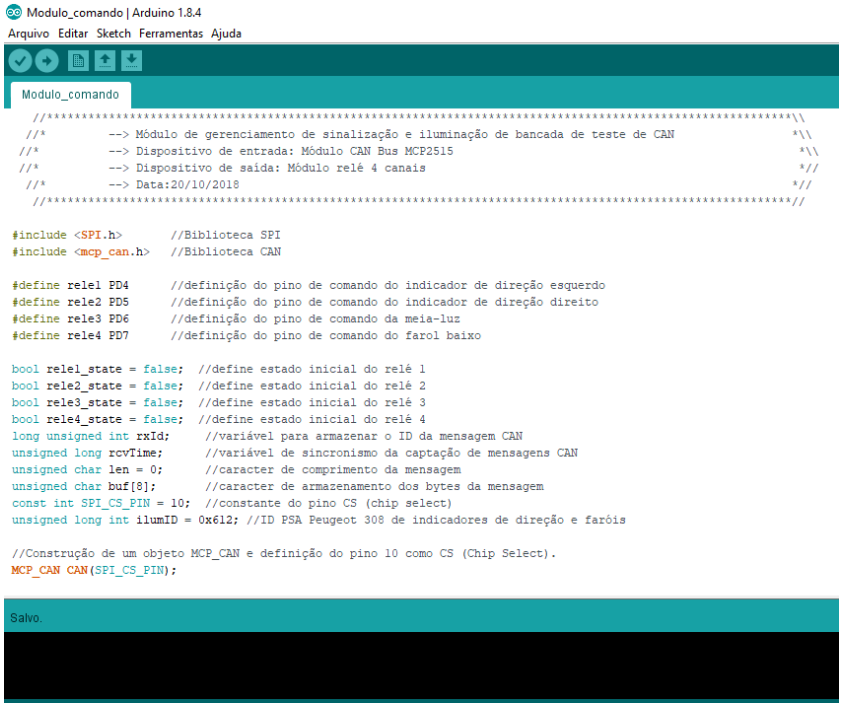
Fonte: Cueto (2018).

Uma vez reconhecidos os ID's e mensagens acima, apresentados em hexadecimal a fim de facilitar a compreensão, pode-se programar o módulo de comando de sinalização e iluminação para atuar integrado à rede CAN. Como é possível observar, o valor lido em um Byte pode significar mais de uma informação, sendo que nele são concatenados valores de variáveis diferentes, a fim de que se possa transmitir a maior quantidade possível de dados numa mesma mensagem (composta por 8 Bytes). Por exemplo, sempre que for transmitido na rede o ID 0x612 com a mensagem hexadecimal '20' no Byte 1 (binário, 0010 0000), significa que tanto as luzes quanto os indicadores de direção do veículo estão desligados. Se no mesmo Byte surgir a mensagem hexadecimal '61' (binário, 0110 0001), então o indicador de direção à direita e o farolete (meia-luz) foram acionados, e assim por diante.

3.2.1.2 Desenvolvimento do Firmware

Para o desenvolvimento da linguagem de programação (*firmware*) aplicada no módulo utilizou-se o ambiente de desenvolvimento integrado (IDE) *Arduino* (versão 1.8.4). O mesmo disponibiliza uma interface simplificada de escrita do código em *Wiring* – linguagem baseada no C e no C++, que possui alguns recursos como classes, objetos e uso de bibliotecas. O software também possibilita a verificação do código desenvolvido, bem como o *upload* (carregamento) do mesmo na plataforma Arduino através de conexão USB (Figura 41).

Figura 41. Detalhe da interface IDE Arduino utilizada na compilação do *firmware.*



Fonte: Autoria própria.

O firmware desenvolvido inclui as bibliotecas necessárias para estabelecer a comunicação do módulo CAN Bus com o Arduino via SPI (Serial Peripherical Interface, Interface Periférica Serial) e para configurar o próprio módulo CAN Bus a fim de que opere adequadamente. Além das bibliotecas, logo no início do *sketch* (rascunho) são definidas as saídas (pinos) que serão utilizadas para comandar os relés do sistema:

```
//*********************************************************************************************************\\
//*         --> Módulo de gerenciamento de sinalização e iluminação de bancada de teste de CAN            *\\
//*         --> Dispositivo de entrada: Módulo CAN Bus MCP2515                                            *\\
//*         --> Dispositivo de saída: Módulo relé 4 canais                                                *//
//*         --> Data:20/10/2018                                                                           *//
//*********************************************************************************************************//

#include <SPI.h>       //Biblioteca SPI
#include <mcp_can.h>   //Biblioteca CAN

#define rele1 PD4      //definição do pino de comando do indicador de direção esquerdo
#define rele2 PD5      //definição do pino de comando do indicador de direção direito
#define rele3 PD6      //definição do pino de comando da meia-luz
#define rele4 PD7      //definição do pino de comando do farol baixo
```

Em seguida, são definidos os estados iniciais dos pinos de acionamento dos relés, a fim de garantir que todos estejam em nível baixo, ou desligados quando o programa começar a rodar. As variáveis a serem utilizadas para as demais funções da programação também são definidas, além do pino de CS (*Chip Select*) conectado ao módulo CAN Bus, que permite ao Arduino selecionar o módulo CAN Bus trocando informações com o mesmo:

```
bool rele1_state = false;  //define estado inicial do relé 1
bool rele2_state = false;  //define estado inicial do relé 2
bool rele3_state = false;  //define estado inicial do relé 3
bool rele4_state = false;  //define estado inicial do relé 4
long unsigned int rxId;     //variável para armazenar o ID da mensagem CAN
unsigned long rcvTime;      //variável de sincronismo da captação de mensagens CAN
unsigned char len = 0;      //caracter de comprimento da mensagem
unsigned char buf[8];       //caracter de armazenamento dos bytes da mensagem
const int SPI_CS_PIN = 10;  //constante do pino CS (chip select)
unsigned long int ilumID = 0x612; //ID PSA Peugeot 308 de indicadores de direção e faróis

//Construção de um objeto MCP_CAN e definição do pino 10 como CS (Chip Select).
MCP_CAN CAN(SPI_CS_PIN);
```

A rotina seguinte (*void setup*), executada somente na inicialização do sistema, como o título preconiza, realiza a configuração inicial do *firmware*, estabelecendo: o *baud rate* (taxa de transmissão de dados) de comunicação serial (em bps, bits/s); o modo de operação dos pinos ligados ao módulo relé (saídas de sinal); a verificação da comunicação do módulo CAN com a rede; a preparação do cabeçalho do Serial Monitor para a organização das mensagens lidas; o teste inicial do módulo relé:

```
//Subrotina de inicialização:
void setup()
{
    Serial.begin(115200);  //Define o baud rate da comunicação serial

    //Definição dos pinos de acionamento dos relés como saída de sinal:
    pinMode(rele1, OUTPUT);
    pinMode(rele2, OUTPUT);
    pinMode(rele3, OUTPUT);
    pinMode(rele4, OUTPUT);

    //Laço de inicialização e verificação da comunicação do MCP2515 com uma rede:
    while (CAN_OK != CAN.begin(CAN_1000KBPS)) // velocidade da rede CAN (high-speed) = 1 Mbps (intersistemas)
    {
        Serial.println("Falha de inicialização do Módulo CAN BUS");
        Serial.println("Tentando novamente....");
        delay(200);
    }

    //Preparação do cabeçalho do Serial Monitor:
    Serial.println("Módulo CAN BUS Inicializado!");
    Serial.println("Tempo\t\tID\tByte0\tByte1\tByte2\tByte3\tByte4\tByte5\tByte6\tByte7");
    digitalWrite(rele1, HIGH);
    digitalWrite(rele2, HIGH);
    digitalWrite(rele3, HIGH);
    digitalWrite(rele4, HIGH);
}
```

A rotina principal do *firmware* é o laço de repetição (*void loop)*, que será repetida indefinidamente enquanto o sistema estiver sendo executado. Dentro do laço constam, principalmente, a leitura dos dados recebidos pelo módulo MCP2515 na rede CAN, o armazenamento dos mesmos nas variáveis de ID e mensagem, e sua escrita organizada no Serial Monitor:

```
//Leitura de ID e mensagem CAN:
void loop(){
   //reinicia as variáveis de ID e mensagem:
   rxId = 0;
   buf[8] = 0;
   len = 0;
   if(CAN_MSGAVAIL == CAN.checkReceive()){  // verificação de chegada de dados
      rcvTime = millis();
      CAN.readMsgBuf(&len, buf);    //leitura da mensagem
      rxId= CAN.getCanId();         //leitura do ID
      if(rxId == 0x612){            //filtro de ID
         Serial.print(rcvTime);
         Serial.print("\t\t");
         Serial.print("0x");
         Serial.print(rxId, HEX);
         Serial.print("\t");
         for(int i = 0; i<len; i++){//impressão do dado no Monitor Serial
            if(buf[i] > 15){
            Serial.print("0x");
            Serial.print(buf[i], HEX);
            }
```

O último trecho da *void loop* direciona a execução do firmware para uma sub-rotina (void rele):

```
            }
            Serial.print("\t");
         }
            Serial.println();
      }
    rele();
    }
}
```

Na sub-rotina acima referida e abaixo apresentada é realizado o acionamento dos pinos de comando dos relés de acordo com a mensagem CAN correspondente a cada estado: iluminação desligada, indicador de direção (seta) esquerdo, indicador de direção (seta) direito, meia-luz, luz baixa e combinações entre os estados anteriores:

```
//Subrotina de atuação dos relés condicionada a mensagem lida:
void rele(){
   rxId= CAN.getCanId();
   CAN.readMsgBuf(&len, buf);
   if(rxId == 0x612){
      if(&len, buf[1] == 0x20){
         digitalWrite(relel, HIGH);
         digitalWrite(rele2, HIGH);
         digitalWrite(rele3, HIGH);
         digitalWrite(rele4, HIGH);
         Serial.println("iluminação desligada");
      }
      if(&len, buf[1] == 0xA0){
         piscal();
         digitalWrite(rele2, HIGH);
         digitalWrite(rele3, HIGH);
         digitalWrite(rele4, HIGH);
         Serial.println("Seta esquerda");
      }
      if(&len, buf[1] == 0x60){
         digitalWrite(relel, HIGH);
         pisca2();
         digitalWrite(rele3, HIGH);
         digitalWrite(rele4, HIGH);
         Serial.println("Seta direita");
      }
```

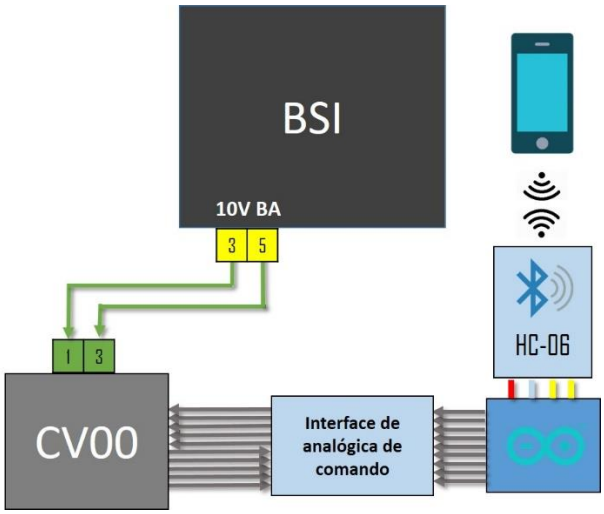
As sub-rotinas *void pisca1* e *void pisca2* são responsáveis por criar o efeito de intermitência (*blink*, piscada) dos indicadores de direção esquerdo e direito, sendo “chamadas” na sub-rotina anterior quando do acionamento dos indicadores de direção:

```
//Subrotinas de blink para os indicadores de direção esquerdo e direito:
void pisca1(){
 digitalWrite(rele1, LOW);
 delay(320);
 digitalWrite(rele1, HIGH);
}

void pisca2(){
 digitalWrite(rele2, LOW);
 delay(320);
 digitalWrite(rele2, HIGH);
}

//*************************************************FIM DO SKETCH*************************************************\\
```

3.3 SISTEMA DE CONTROLE REMOTO

O dispositivo de controle remoto desenvolvido visa substituir a necessidade de atuação dos comandos manuais das alavancas do painel por comandos de voz, reduzindo a exigência dos membros superiores do condutor na operação dos sistemas de sinalização e iluminação. Para tanto, desenvolveu-se um circuito integrado à rede CAN veicular, acoplado fisicamente ao conjunto de alavancas do veículo, atuado remotamente via comunicação *bluetooth* por aplicativo instalado em um dispositivo móvel (Figura 42).

Figura 42. Diagrama de blocos da versão final do controle remoto.



Fonte: Autoria própria.

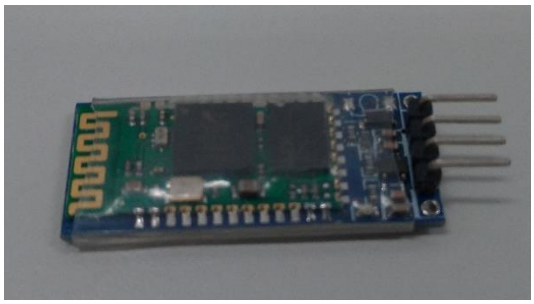
3.3.1 Estrutura eletroeletrônica

Os componentes utilizados no desenvolvimento do sistema de controle remoto são:

- Plataforma de prototipagem eletrônica Arduino UNO;
- Módulo de comunicação *bluetooth* JY-MCU HC-05;
- Interface analógica de comando por transistores;

O módulo *bluetooth* é o dispositivo pelo qual se realiza a comunicação sem fio entre o sistema de controle remoto e o dispositivo móvel. O modelo adotado estabelece comunicação com o Arduino por meio de interface serial, normalmente a um *baud rate* de 9600 bps, podendo ser reconfigurado de acordo com a aplicação (Figura 43).

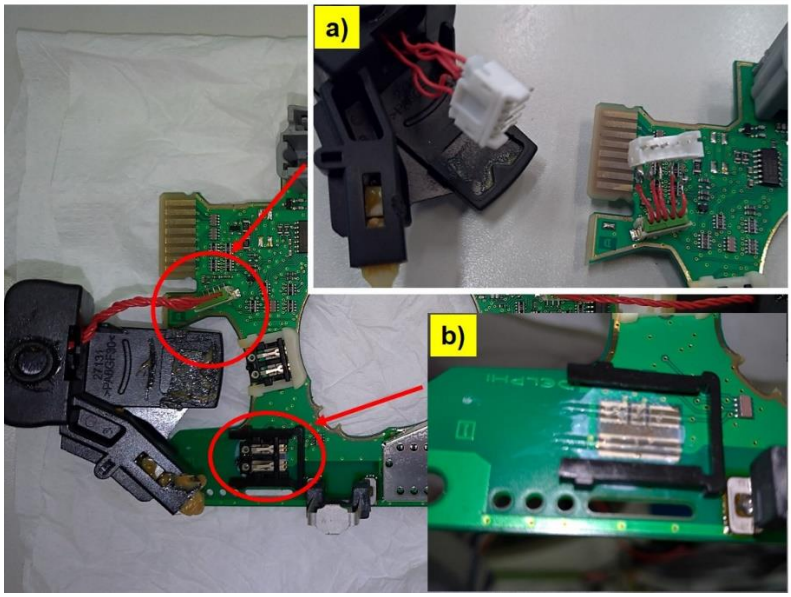
Figura 43. Módulo de comunicação *bluetooth* HC-05.



Fonte: Autoria própria.

A integração do controle remoto com o sistema veicular se dá através da interface analógica de comando, dentro do conjunto de alavancas. Este dispositivo é instalado entre a alavanca de comando das luzes e indicadores de direção e a placa do circuito. É através dele que se realizam os comandos analógicos internos ao conjunto da alavanca sem a necessidade da atuação da mesma, Figura 44.

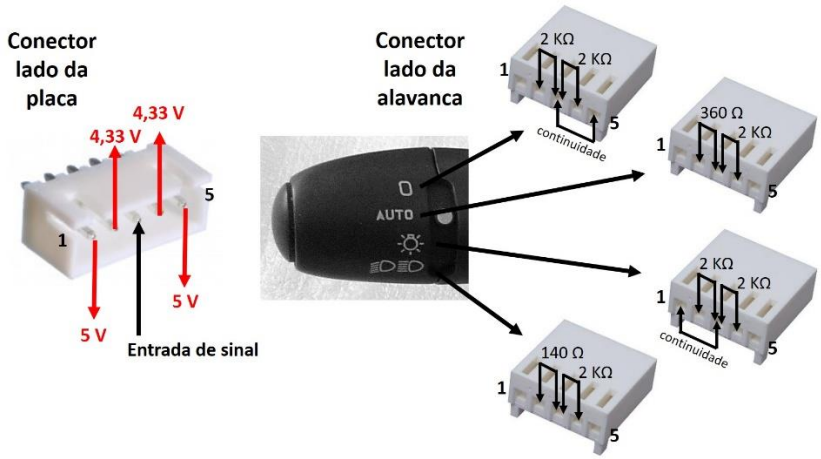
Figura 44. Pontos do circuito do conjunto de alavancas onde a interface é ligada.



Fonte: Autoria própria.

Para determinar quais os comandos enviados pela alavanca, realizaram-se medições elétricas no conector da mesma e no conector da placa, chegando-se aos resultados da Figura 45:

Figura 45. Resultados das medições no conector da alavanca de luzes.



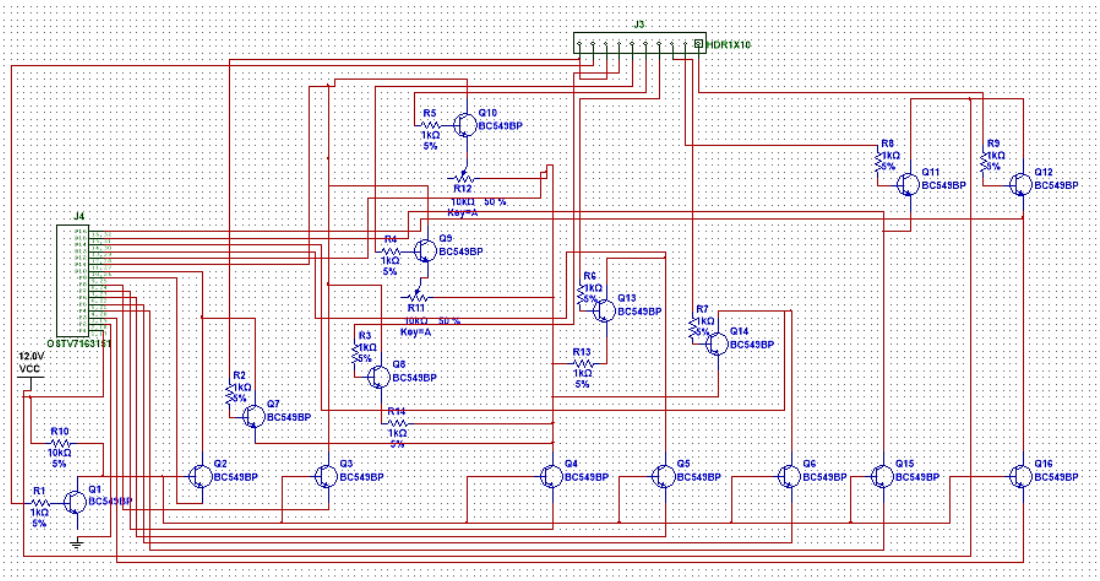
Fonte: Autoria própria.

Conforme se observa na Figura 44, há dois pontos do circuito interno do conjunto de alavancas onde a interface será conectada. No ponto onde o chicote da alavanca é conectado à placa do circuito (a), foram inseridos dois conectores por meio dos quais será ligada a interface de controle de luzes. No ponto onde o interruptor

deslizante realiza o acionamento dos indicadores de direção (b) foram soldados três conectores, que também serão ligados na interface de comando.

As medições elétricas realizadas permitiram determinar que a alavanca interage com a placa como uma combinação de divisores de tensão e chaveamentos, circuito este reproduzido na interface analógica de comando, que será controlado pelo Arduino, Figura 46.

Figura 46. Diagrama da interface analógica de comando.



Fonte: Autoria própria.

Na interface há um conjunto de componentes (oito transistores da parte inferior do diagrama) responsável pelo chaveamento do circuito, conectando e desconectando eletronicamente o sistema de controle remoto do circuito veicular, substituindo a alavanca original do sistema. Este chaveamento é feito também por comando de voz, e executado por um pino de saída do Arduino, assim como os demais comandos responsáveis pela atuação do sistema de sinalização e iluminação. Para estes, cada transistor do circuito da interface realiza uma conexão entre determinados pinos do chicote sendo que, para cada posição da alavanca há uma combinação específica de acionamentos, que gerará o mesmo comportamento do circuito original, Figura 47.

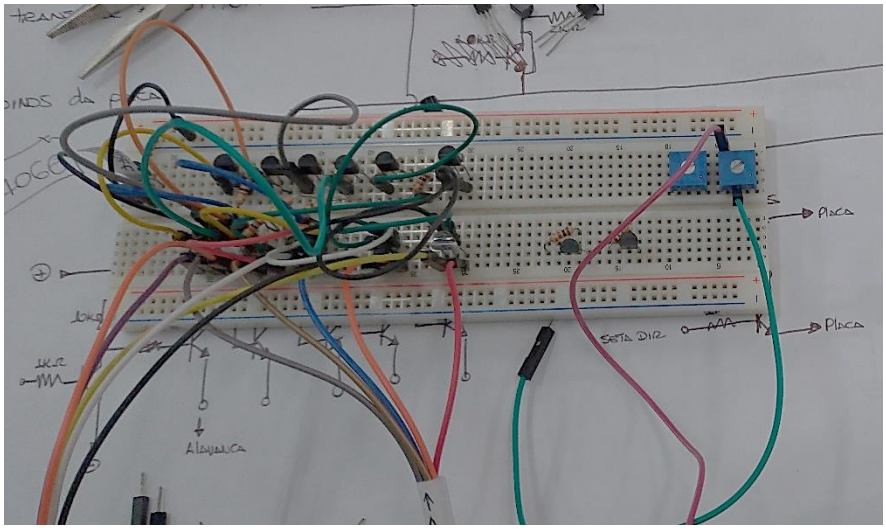
Figura 47. Combinações de acionamentos dos transistores para cada comando do sistema.

		Saídas digitais do Arduino								
		3	4	5	6	7	8	9	10	11
Função	Comando do sistema	Estado dos Pinos								
Controle Remoto	ON/OFF	1	0		0	0	0	0	0	0
Luzes	0	1			1	1	0	0	1	0
	Auto	1			1	0	1	0	0	0
	Farolete	1			1	1	0	0	0	1
	Luz baixa/alta	1			1	0	0	1	0	0
Indicador de direção	Esquerdo	1	1	0	0					
	Direito	1	0	1						

Fonte: Autoria própria.

Para os testes iniciais, o circuito foi montado em matriz de contatos (*protoboard)*, possibilitando medições e ajustes durante a fase de teste e validação da plataforma, Figura 48.

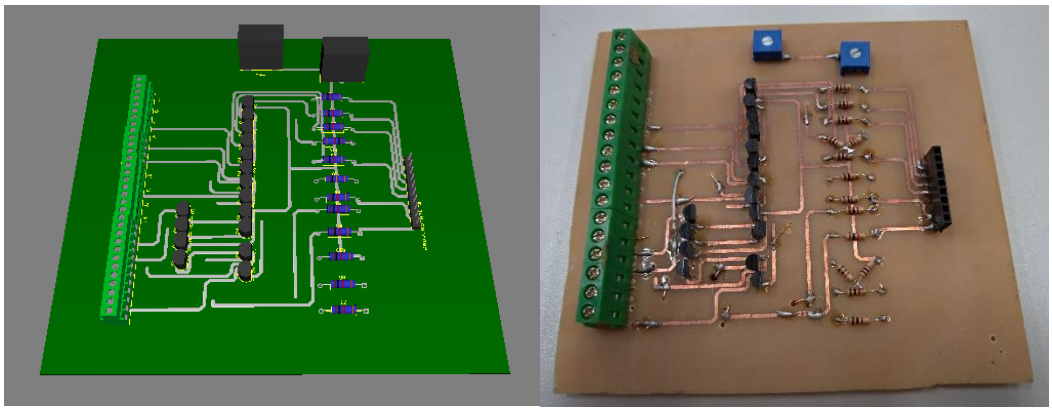
Figura 48. Montagem de teste da interface de comando do controle remoto.



Fonte: Autoria própria.

Após testes e ratificação do modelo eletrônico do sistema, desenvolveu-se um projeto de roteamento em PCB em software dedicado, possibilitando a montagem e soldagem dos componentes numa placa mais apropriada, livre de problemas de mau contato e fisicamente mais próxima de um projeto real, Figura 49.

Figura 49. Projeto de PCB desenvolvido em *software* (esq.) e placa pronta (dir.).



Fonte: Autoria própria.

3.3.2 *Firmware* do Sistema de Controle Remoto

A programação do sistema de controle remoto, se comparada com a do módulo de controle da bancada, é muito mais simples, uma vez que não são necessárias bibliotecas para dispositivos ou funções específicas. Na primeira parte do *sketch* são definidas as saídas (pinos) do Arduino que serão utilizadas para comandar a interface analógica, bem como a variável que receberá os dados transmitidos pelo dispositivo móvel via conexão *bluetooth*:

```
  //*************************************************************************************************************\\
 //*         --> Dispositivo de Controle Remoto de Sistemas de Sinalização e Iluminação Veicular               *\\
//*          --> Dispositivo de entrada: Módulo Bluetooth HC-05 (+ App dedicado)                                *\\
//*          --> Dispositivo de saída: Interface Analógica de Comando integrada à Rede CAN                      *//
 //*         --> Data:26/10/2018                                                                               *//
  //*************************************************************************************************************//

int Bluetooth; //Variável que irá receber o comando enviado do Android
int onoff    = 3;
int setaesq  = 4;
int setadir  = 5;
int comandoA = 6;
int comandoB = 7;
int comandoC = 8;
int comandoD = 9;
int comandoE = 10;
int comandoF = 11;
```

Em seguida, na rotina de inicialização do programa *(void setup)*, é definida a taxa de transmissão de dados em 9600 bps, que é um valor padrão para comunicação *bluetooth*. Também as condições dos pinos de controle são definidas como saída de sinal, e ajustados os estados iniciais dos mesmos como nível baixo de sinal (LOW), ou seja, todas as saídas estarão desligadas ao iniciar o sistema:

```
void setup() {
 Serial.begin(9600);//Inicia comunicação serial
 pinMode(onoff,OUTPUT);    //Definindo o pino 3 como saída
 pinMode(setaesq,OUTPUT);  //Definindo o pino 4 como saída
 pinMode(setadir,OUTPUT);  //Definindo o pino 5 como saída
 pinMode(comandoA,OUTPUT); //Definindo o pino 6 como saída
 pinMode(comandoB,OUTPUT); //Definindo o pino 7 como saída
 pinMode(comandoC,OUTPUT); //Definindo o pino 8 como saída
 pinMode(comandoD,OUTPUT); //Definindo o pino 9 como saída
 pinMode(comandoE,OUTPUT); //Definindo o pino 10 como saída
 pinMode(comandoF,OUTPUT); //Definindo o pino 11 como saída
```

Na rotina cíclica do *firmware* (*void loop)*, estão as funções principais: leitura dos dados do módulo *bluetooth* através de protocolo serial; e um switch, que determina, de acordo com o dado lido, qual será o comando realizado na interface analógica, através de uma combinação de sinais nos transistores de chaveamento da mesma:

```
void loop() {
  if(Serial.available()>0){
    Bluetooth=Serial.read();
    Serial.println(Bluetooth);
  }
    switch(Bluetooth){
```

Os casos (*cases)* ou opções da função *switch* são condicionados a determinados caracteres enviados via *bluetooth*, que são transmitidos ao controlador como seus correspondentes na tabela ASCII (*American Standard Code for Information Interchange*, Código Padrão Norte-Americano para Intercâmbio de Informações):

- Case 49 – número 1 – comando *bluetooth* para ligar ou desligar o sistema de controle remoto;
- Case 50 – número 2 – comando para ligar ou desligar o indicador de direção esquerdo;
- Case 51 – número 3 – comando para ligar ou desligar o indicador de direção direito;
- Case 65 – letra A – comando que corresponde à posição 0 (zero) da alavanca de acionamento de luzes no veículo;
- Case 66 – letra B – comando que corresponde à posição AUTO (automático) da alavanca de acionamento de luzes. Nesta posição está habilitado o sensor crepuscular do veículo;
- Case 67 – letra C – comando que corresponde à posição de farolete (meia-luz) da alavanca de acionamento de luzes.

- Case 68 – letra D – comando que corresponde à posição de luz baixa/luz alta da alavanca de acionamento de luzes.

```
  case 49:
    Serial.println("1 ok");
    digitalWrite(onoff,!digitalRead(onoff));//Escreve na porta 3 o inverso do valor lido
    digitalWrite(setaesq,LOW);//Reinicia a porta 4 (desliga)
    digitalWrite(setadir,LOW);//Reinicia a porta 5 (desliga)
    digitalWrite(comandoA,LOW);//Reinicia a porta 6 (desliga)
    digitalWrite(comandoB,LOW);//Reinicia a porta 7 (desliga)
    digitalWrite(comandoC,LOW);//Reinicia a porta 8 (desliga)
    digitalWrite(comandoD,LOW);//Reinicia a porta 9 (desliga)
    digitalWrite(comandoE,LOW);//Reinicia a porta 10 (desliga)
    digitalWrite(comandoF,LOW);//Reinicia a porta 11 (desliga)
    Serial.println(digitalRead(onoff));
    Bluetooth=0;
  break;
  case 50:
     if(digitalRead(onoff)){
     Serial.println("2 ok");
     digitalWrite(setadir,LOW);//Reinicia a porta 5 (desliga)
     digitalWrite(setaesq,!digitalRead(setaesq));//Escreve na porta 4 o inverso do valor lido
     }
     Bluetooth=0;
  break;


   case 68:
      if(digitalRead(onoff)){
      Serial.println("D ok");
      digitalWrite(comandoA,!digitalRead(comandoA));//Escreve na porta 6 o inverso do valor lido
      digitalWrite(comandoB,LOW);//Reinicia a porta 7 (desliga)
      digitalWrite(comandoC,LOW);//Reinicia a porta 8 (desliga)
      digitalWrite(comandoD,!digitalRead(comandoD));//Escreve na porta 9 o inverso do valor lido
      digitalWrite(comandoE,LOW);//Reinicia a porta 10 (desliga)
      digitalWrite(comandoF,LOW);//Reinicia a porta 11 (desliga)
      }
      Bluetooth=0;
   break;
   }
}

\\*********************************************FIM DO SKETCH*****************************************************\\
```

3.4 APLICATIVO DE COMANDO DE VOZ

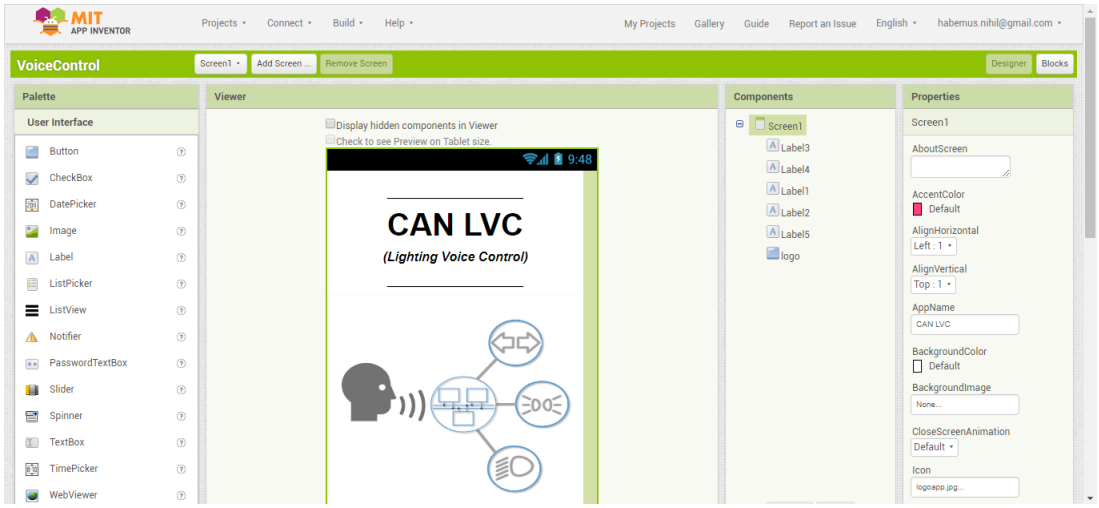
Uma vez pronto o sistema de controle remoto, faz-se necessário o desenvolvimento de um aplicativo para dispositivos móveis com reconhecimento de informações por comando de voz. Para o desenvolvimento do aplicativo foi utilizada a plataforma online *App Inventor*, do MIT (*Massachusetts Institute of Technology*, Instituto de Tecnologia de Massachusetts).

3.4.1 Programação do Aplicativo

A *App Inventor* é uma plataforma de programação por linguagem de texto estruturado de alto nível, subdividida em duas interfaces: a interface de programação

gráfica (*Designer)* e a de programação em blocos (*Blocks)*. A programação gráfica é utilizada para se determinar as telas do aplicativo e desenvolver o aspecto estético, onde se confeccionam os estilos das telas (cor dos itens, cor ou imagem de fundo, etc.) e são definidos os elementos nelas presentes, como botões, caixas de texto, cabeçalhos, links, uso de sensores, etc. Já na interface de blocos é feita a programação do aplicativo em si, estabelecendo as relações lógicas e funcionais entre os itens presentes nas telas, definidos anteriormente na programação gráfica (Figura 50).

Figura 50. Interface de programação gráfica da plataforma *App Inventor.*

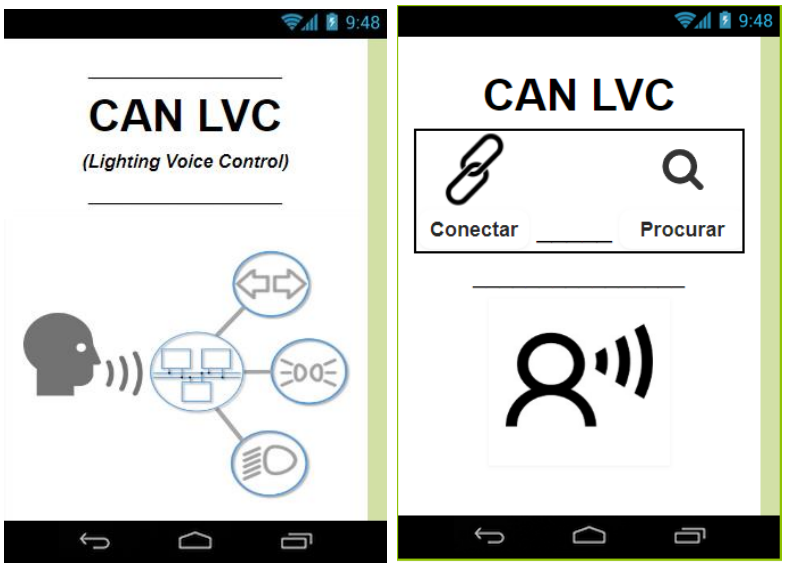


Fonte: Autoria própria.

Dada a premissa de que o uso do aplicativo em um dispositivo móvel, seja ele um *smartphone* ou um *tablet*, deve substituir a operação de controles remotos manuais por meio do uso da voz, tem-se que a sua interface deve ser simplificada e intuitiva, além de funcional. Sendo assim, o projeto desenvolvido possui apenas duas telas (*screens*):

- Tela inicial – contendo título e logo do aplicativo;
- Tela principal – interface com botões de busca de dispositivos, ativação da conexão *bluetooth* e comando do controle remoto, vide Figura 51.

Figura 51. Tela inicial (à esquerda) e tela principal do aplicativo (à direita).



Fonte: Autoria própria.

Ao ativar o aplicativo, a tela inicial é apresentada. A transição para a tela principal é feita com um toque simples sobre a logo. A programação desta transição de telas é simples, sendo necessária apenas uma função *when.Click* (“quando clicar em...”). Esta função aguarda um clique na logo para abrir a tela principal, Figura 52.

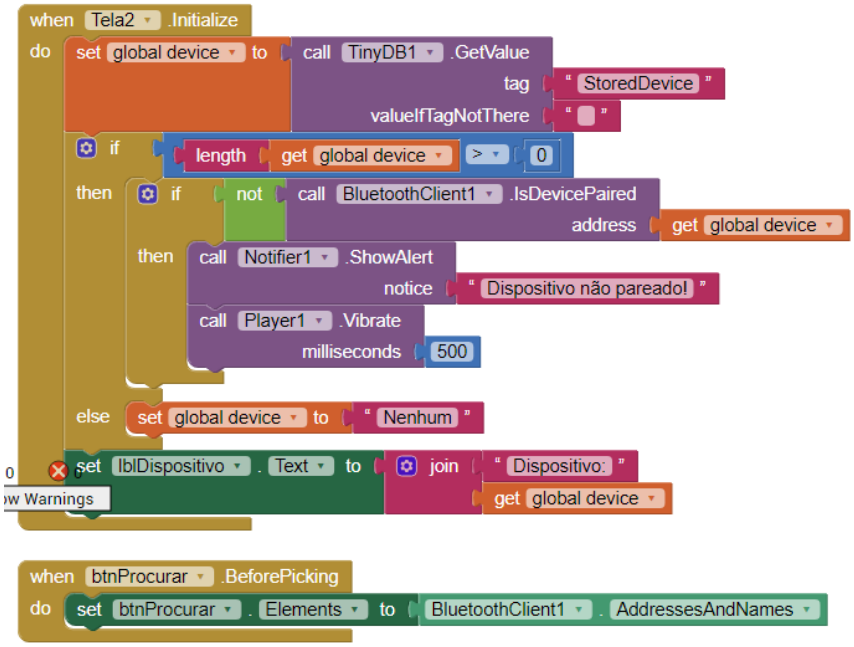
Figura 52. Programação da transição de telas em linguagem de blocos.

Fonte: Autoria própria.

Na tela principal há três funcionalidades: busca de dispositivos, conexão com dispositivos via *bluetooth* e comando de voz, todas condicionadas à atuação dos botões nela presentes. Ao ser inicializada a tela principal, algumas ações são executadas antes mesmo do acionamento dos botões, a fim de criar os objetos e lógicas de conexão com dispositivos *bluetooth*. Estas configurações iniciais estão contidas em duas funções *when:* uma delas estabelece, logo após a inicialização da tela, as configurações da comunicação *bluetooth* no dispositivo móvel, a mensagens de não pareamento e realiza a busca por outros dispositivos em segundo plano; a outra função, faz a leitura da lista de dispositivos já encontrados antes de soltar o

clique no botão ‘Procurar’, uma vez que a rotina anterior já realizou a busca e armazenou os dispositivos encontrados, Figura 53.

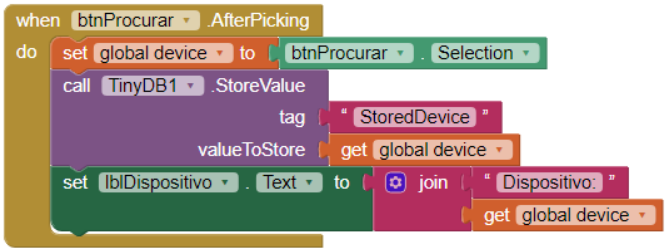
Figura 53. Funções iniciais da tela principal do aplicativo.



Fonte: Autoria própria.

Uma vez liberado o botão ‘Procurar’, o aplicativo apresentará uma lista com todos os resultados previamente encontrados, Figura 54.

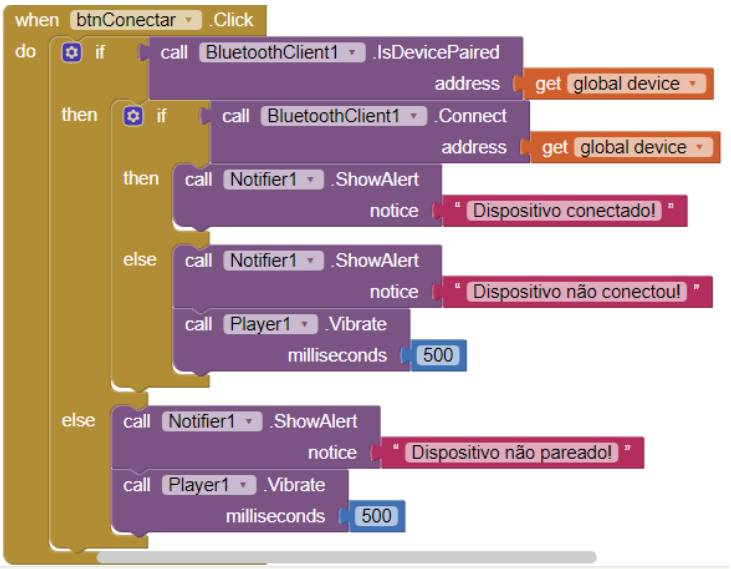
Figura 54. Funções atreladas ao botão ‘Procurar’.



Fonte: Autoria própria.

Ao clicar no botão ‘Conectar’, a rotina inicialmente verifica se o dispositivo está pareado e, em caso afirmativo, notifica o usuário com a mensagem “Dispositivo conectado!”. Caso contrário, a notificação “Dispositivo não conectou!”. Se o dispositivo nem tiver sido pareado com o aparelho, a mensagem “Dispositivo não pareou!” será apresentada e o aparelho vibrará, Figura 55.

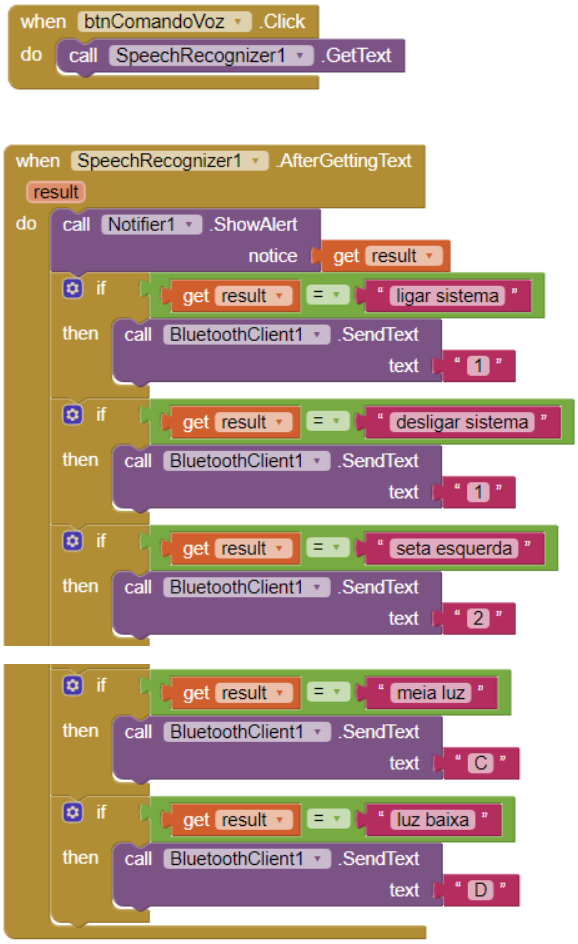
Figura 55. Funções atreladas ao botão “Conectar”.



Fonte: Autoria própria.

Uma vez conectados via *bluetooth* o dispositivo móvel e o controle remoto do veículo, é possível utilizar o comandos de voz do aplicativo para enviar informações para o sistema de sinalização e iluminação do veículo. A primeira etapa é a função *when* relacionada ao clique do ícone da tela principal que corresponde ao comando de voz (figura de um boneco emitindo ondas). Esta função necessita de conexão com a internet para funcionar, uma vez que captura o aúdio no microfone do dispositivo móvel e utiliza a plataforma de reconhecimento de voz da *Google* na nuvem. Depois de reconhecido o comando de voz, a próxima função *when* recebe o resultado e, de acordo com a solicitação do condutor, envia um comando específico via *bluetooth* para o sistema de controle remoto, conforme Figura 56.

Figura 56. Funções atreladas ao botão ‘Comando de Voz’.



Fonte: Autoria própria.

4 APRESENTAÇÃO E ANÁLISE DOS RESULTADOS

Sempre que se desenvolve, quer seja um projeto, um sistema ou um equipamento, é inevitável o surgimento de elementos não programados e/ou dificuldades imprevistas, como falta de insumos, ferramentas e/ou conhecimentos necessários ao término da tarefa. Tais desafios são inerentes ao desenvolvimento de atividades num âmbito geral, têm níveis de impacto diferentes, e podem ter sua fonte interna ou externa ao processo. A partir deste ponto de vista, o presente projeto apresentou alguns desafios, conforme exposto a seguir.

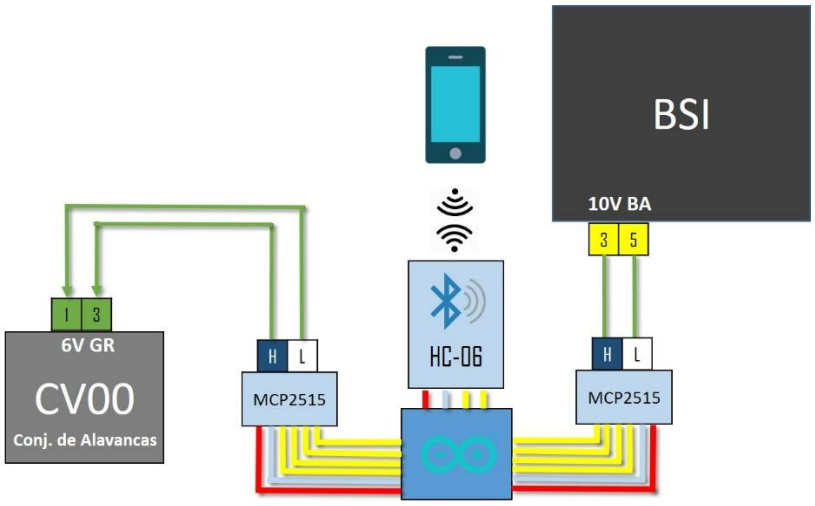
4.1 BANCADA DE TESTES

A operacionalização da bancada apresenta algumas dificuldades no que diz respeito ao funcionamento dos componentes automotivos fora do ambiente do veículo. A linha PSA (Peugeot e Citroën) possui proteções de rede em nível de aplicação, que geram operação dos módulos em modo de emergência quando da ausência de outros módulos, ou mesmo a presença de dispositivos não reconhecidos pela rede. Por conta disso, o painel de instrumentos da bancada de testes opera em modo de emergência, o que restringe a atuação e sinalização de algumas funções. Um exemplo desta restrição é a operação do indicador de combustível, que só atualiza a posição do ponteiro se a bancada for desligada e ligada novamente com o sensor de nível em uma nova posição. Isso se deve à ausência da PSF1 e do controlador do sistema motopropulsor, que fornece informações complementares ao sensor de nível para que o indicador atualize sua posição em tempo real. Um outro exemplo são os indicadores das luzes (farolete, baixa e alta), que permanece sempre na posição de luz baixa, mesmo recebendo os sinais de outras posições de funcionamento das alavancas de comando de luzes via rede CAN.

4.2 CONTROLE REMOTO

Desde a ideação do projeto, o dispositivo de controle remoto passou por três versões, até que fosse possível gerar resultados próximos da expectativa inicial. O primeiro esboço da solução atuava somente no barramento CAN, e possuía como função filtrar a comunicação CAN entre a BSI e o conjunto de alavancas, comportando-se como se fosse o conjunto de alavancas, enviando dados correspondentes a cada comando de voz dado, sobrepostos aos dados originais, Figura 57.

Figura 57. Diagrama de blocos da primeira versão do controle remoto.



Fonte: Autoria própria.

Para este dispositivo a estrutura eletroeletrônica era composta de:

- Plataforma de prototipagem eletrônica Arduino UNO;
- Módulo de comunicação *bluetooth* JY-MCU HC-05;
- 2 Módulos CAN Bus MCP2515;

Neste modelo de solução, quando o controle remoto estava desligado, a comunicação entre conjunto de alavancas e a BSI ocorria normalmente, passando pelo microcontrolador sem interferências. Quando acionado o dispositivo por comando de voz via *bluetooth*, a intenção era de que os dados fossem filtrados pelo Arduino, e a interface do controle remoto passava a enviar dados CAN sobrepostos aos originais nos mesmos pinos da BSI em lugar do conjunto de alavancas.

Porém, durante o desenvolvimento deste modelo, surgiram alguns entraves:

1. O sistema microcontrolado adotado no projeto não opera em tempo real, o que atrasa a comunicação entre o mesmo e os dispositivos automotivos;
2. Uma vez que os módulos CAN Bus operam através de interface serial em conjunto com o Arduino, assim como o módulo *bluetooth*, não existe uma biblioteca que habilite o uso de 2 módulos CAN Bus simultaneamente na mesma plataforma microcontrolada, sendo necessário desenvolver tal biblioteca;

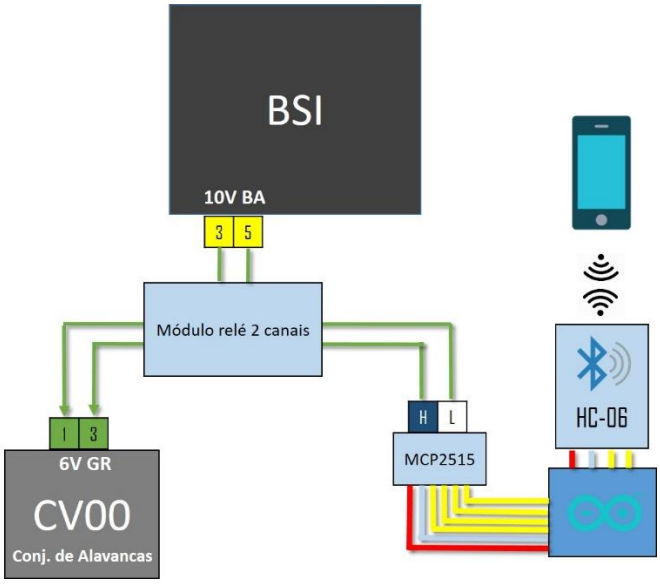
Por esses motivos, esta solução apresentou-se inviável, levando em consideração o tempo disponível para a execução do projeto e as limitações de software e hardware.

Uma segunda possível solução foi proposta, ainda na tentativa de utilizar o barramento CAN como meio de integração do controle remoto ao sistema veicular, contendo:

- Plataforma de prototipagem eletrônica Arduino UNO;
- Módulo de comunicação *bluetooth* JY-MCU HC-05;
- Módulo CAN Bus MCP2515;
- Módulo relé 2 canais;

Neste modelo, o barramento da rede CAN entre BSI e conjunto de alavancas recebeu um conjunto de relés de chaveamento, com a finalidade de abrir o circuito entre os componentes quando do acionamento do controle remoto. Este novo modelo pretendia resolver os problemas do sistema anterior: a capacidade de transferência e filtragem dos dados em tempo real por parte do Arduino e a ausência de biblioteca para operação de módulos CAN Bus simultâneos. O acionamento do dispositivo por comando de voz via *bluetooth* substituía o conjunto de alavancas pelo controle remoto, que enviava os dados CAN correspondentes aos comandos de voz, permitindo o uso de apenas um módulo CAN Bus, conforme Figura 58.

Figura 58. Diagrama de blocos da segunda versão do controle remoto.



Fonte: Autoria própria.

Porém, uma nova limitação surgiu. Como a plataforma Peugeot atua na camada de aplicação do protocolo CAN, reconhecendo cada módulo do veículo, a tentativa de transmitir dados para a rede a partir do controle remoto não era reconhecida como mensagem CAN válida, alarmando uma falha de sistema. Ou seja, não é possível escrever dados CAN na rede original em substituição aos dos módulos proprietários. Portanto, dadas as limitações e objetivos do projeto, desenvolveu-se uma terceira versão do sistema de controle remoto, que não atua necessariamente no barramento CAN, mas no nível de comando analógico interpretado pelo controlador do conjunto de alavancas, que converte os sinais enviados pelo controle remoto em dados CAN válidos para a rede veicular.

4.3 INTEGRAÇÃO DOS SISTEMAS

Uma vez realizados todos os testes e ajustes dos sistemas individualmente, a fase de integração do projeto contemplou a junção dos três dispositivos desenvolvidos: a bancada de testes, o sistema de controle remoto e o aplicativo de comando de voz. Nesta etapa a atenção principal é nas interfaces, que entre a bancada e o sistema de controle remoto é física, por meio de chicote elétrico e entre o controle remoto e o aplicativo virtual, através da comunicação *bluetooth*.

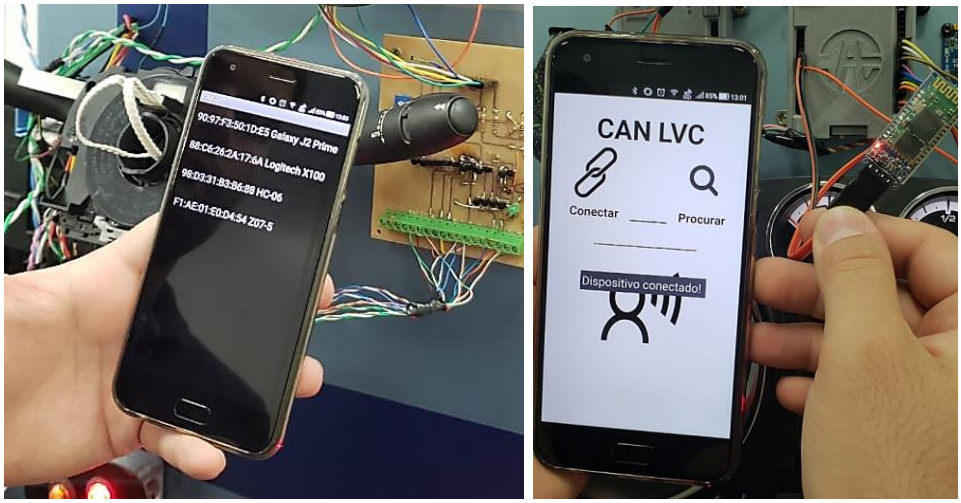
A maior dificuldade encontrada nesta fase foi encapsular os conectores dentro do conjunto de alavancas e resolver os problemas de mau contato entre os mesmos. Uma vez resolvido este problema, foi testado o comando de voz com o aplicativo atuando no controle remoto, tendo funcionado corretamente. Ao se enviar um comando qualquer, a interface analógica atua como se fosse a alavanca e a placa do conjunto de alavancas envia à BSI os dados CAN correspondentes. Uma vez que o painel de instrumentos está operando em modo de emergência, alguns destes comandos não são verificados no mesmo, porém as mensagens CAN podem ser verificadas no módulo MCP2515 da bancada ou mesmo com auxílio de osciloscópio por meio dos conectores de inspeção elétrica disponíveis na bancada.

4.4 OPERAÇÃO DO CONTROLE REMOTO

A operação do conjunto integrado – bancada e controle remoto – tem início na energização da bancada de testes, através do acionamento do interruptor geral de alimentação, apresentado nas Figuras 35 e 36. Uma vez acionado, o mesmo energiza o sistema automotivo e também o sistema de controle remoto, que permanece funcionando em modo espera (*stand-by*) enquanto o comando de voz de ativação não é recebido, estado evidenciado pelo piscar do LED do módulo *bluetooth*. Neste estágio, o conjunto de alavancas opera normalmente. Em seguida, o condutor deve ativar o *bluetooth* e a internet do aparelho móvel, abrir o aplicativo previamente instalado e proceder à tela principal do mesmo, conforme Figura 51.

Uma vez na tela principal, ao clicar no botão ‘Procurar’, aparecerão listados os dispositivos *bluetooth* pareados com o aparelho, incluindo o receptor do sistema de controle remoto, identificado como HC-06. Uma vez detectado o receptor do sistema, retorna-se à tela principal e clica-se no botão ‘Conectar’, que estabelecerá a conexão entre o emissor (dispositivo móvel) e o receptor (sistema de controle remoto). O estado da conexão é verificado através do LED indicador do módulo *bluetooth* que permanecerá continuamente aceso, e também pela mensagem ‘Dispositivo conectado!’ apresentada no aplicativo. Se alguma falha ocorrer no momento da conexão, a mensagem ‘Dispositivo não pareado!’ será exibida, e o LED continuará piscando (Figura 59).

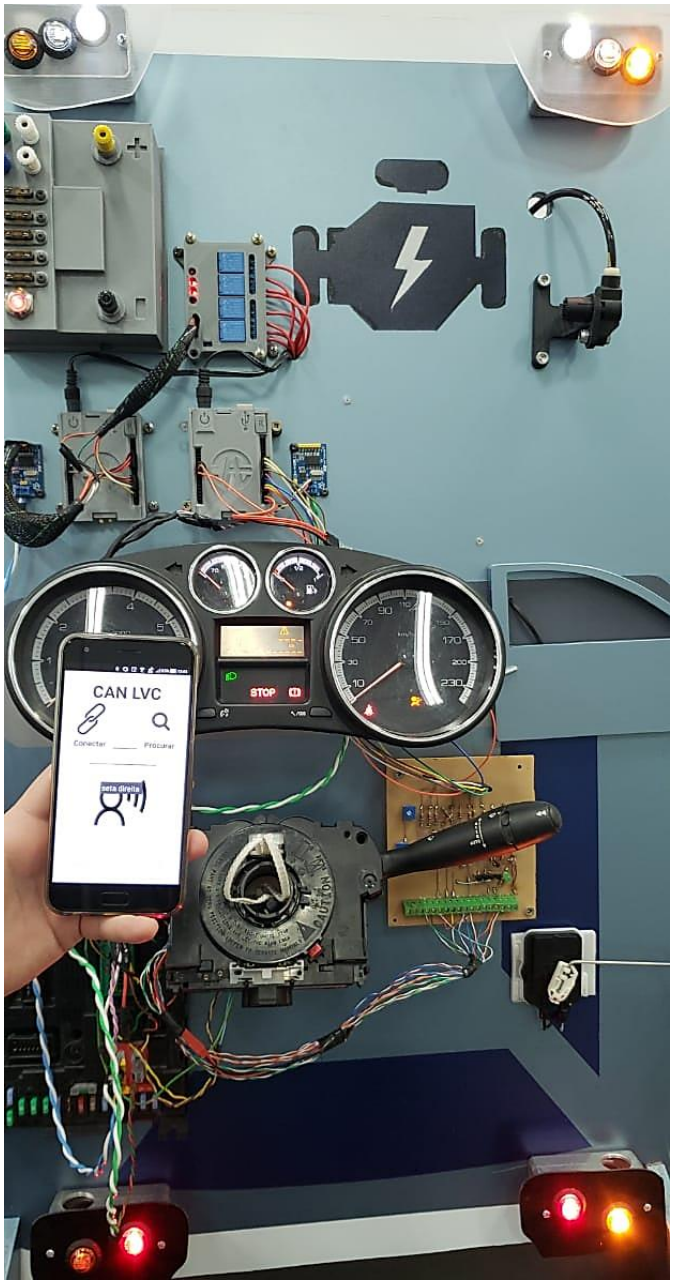
Figura 59. Lista de dispositivos *bluetooth* detectados pelo aplicativo (esq.) e mensagem de confirmação da conexão com LED indicador (dir.).



Fonte: Autoria própria.

Após a conexão do dispositivo é necessário clicar no botão de comando de voz (ícone principal da tela) e dar o comando ‘Ligar sistema’ para que o controle remoto seja habilitado. A partir deste comando, para cada função de sistema de sinalização e iluminação que se deseja atuar, deve-se clicar no botão de comando de voz e em seguida dar o comando correspondente. A identificação do comando de voz é feita, a frase (*string*) exibida na tela do aplicativo e o sinal bluetooth emitido para o receptor do controle remoto (Figura 60).

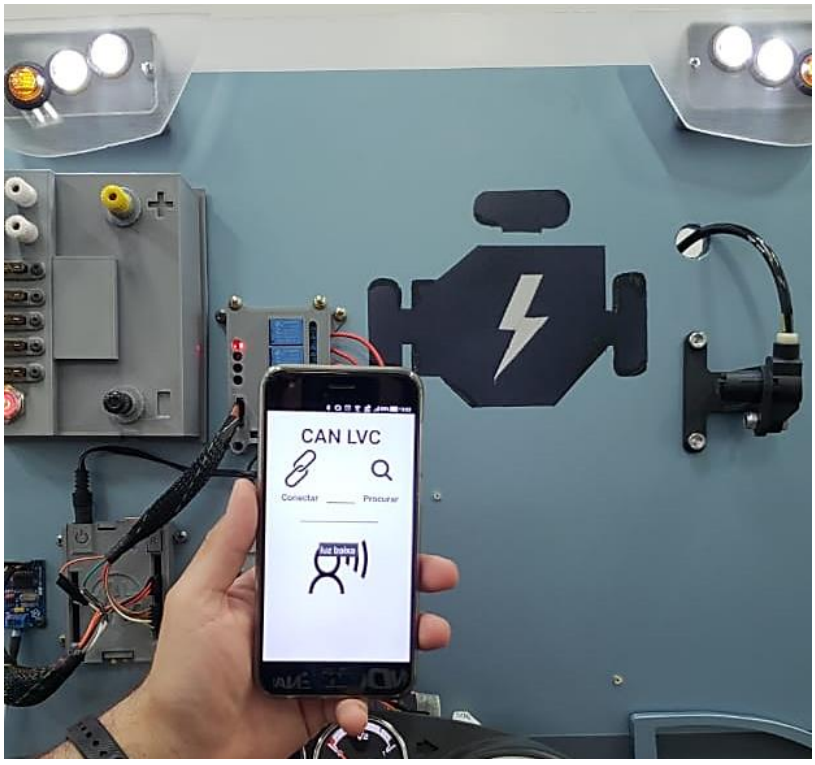
Figura 60. Comando de voz ‘seta direita’ (detalhe para a mensagem na tela do aplicativo).



Fonte: Autoria própria.

Para simplificar a operação do sistema e tornar a utilização mais intuitiva, o mesmo comando de voz ativa e desativa uma dada função veicular. Por exemplo, uma vez as luzes desligadas, se o comando ‘luz baixa’ é dado, as luzes baixas são ativadas. Ao repetir o mesmo as luzes são desligadas, bastando memorizar apenas um comando para cada função veicular. Uma outra característica é a independência e simultaneidade de funcionamento, podendo-se ativar e desativar um dado sistema com um determinado comando mesmo outro estando em funcionamento, sem que haja interferência ou falha de funcionamento (Figura 60).

Figura 61. Funcionamento simultâneo das funções meia-luz e luz baixa.



Fonte: Autoria própria.

A desativação do sistema se dá por meio do comando de voz ‘desligar sistema’, desligando todos os sistemas em operação por meio do controle remoto e também reativando o uso do conjunto de alavancas.

5 CONSIDERAÇÕES FINAIS

A aplicação de protocolos de comunicação, mais especificamente a rede CAN, nas arquiteturas veiculares figura como uma das tecnologias mais importantes do ponto de vista da eletrônica, ao integrar diversos sistemas por meio de redes embarcadas distribuídas, utilizando-se de estruturas físicas relativamente simples e ao mesmo tempo robustas, como é o caso dos barramentos CAN de par trançado, que garantem confiabilidade e resistência à aplicação automotiva. Dentre as contribuições das redes veiculares, tornou-se possível o desenvolvimento e a inserção de novos acessórios, sistemas e dispositivos nos automóveis com mais recursos e interatividade, trazendo soluções e respondendo a necessidades específicas, tanto de segurança quanto de conforto.

Esta grande versatilidade da aplicação de sistemas embarcados interconectados via redes possibilita o desenvolvimento de soluções voltadas para públicos que demandam adaptações e melhorias personalizadas, como é o caso dos condutores idosos e deficientes físicos. Para cada condição ou necessidade pontual do condutor, como por exemplo, amputamento total ou parcial de membro superior ou inferior, paraplegia, incapacidade de realizar força física, dentre outros, faz-se necessária uma adaptação diferente do veículo, tornando possível a condução segura de um carro, realizando todas as tarefas necessárias.

Considerando tais aspectos, a solução desenvolvida ao longo deste trabalho é um controle remoto integrado à rede veicular via comando de voz, para atuação dos sistemas de sinalização e iluminação veicular, possibilitando ao condutor realizar tais tarefas com o mínimo de esforço possível. O protótipo desenvolvido constitui um MVP (*Minimum Viable Product*, Produto Mínimo Viável), uma versão simplificada de um dispositivo possível de ser desenvolvido para aplicações veiculares. Dito isto, vale ressaltar que o foco do projeto não está em atender pré-requisitos estruturais e/ou de caráter estético, dado o nível de simplicidade e uso de dispositivos didáticos na sua confecção, portanto a solução desenvolvida atende aos objetivos iniciais, de possibilitar a integração de um a dispositivo de controle remoto via comando de voz com a transmissão de dados na rede CAN veicular, mesmo que não esteja em etapa final de prototipação.

Dentre as dificuldades e limitações que, de certa forma, também restringiram o potencial do protótipo, há algumas opções de aprimoramento, como por exemplo a

substituição do Arduino UNO como plataforma de controle por outra com maior capacidade de processamento e versatilidade, como é o caso do Raspberry Pi ou mesmo do Arduino DUE, que já possui duas portas CAN Bus embarcadas na placa e processador de 32 bits.

Uma vez que o presente projeto provou ser possível desenvolver um sistema de baixo custo que facilita a operação de comandos veiculares via comando de voz, as possibilidades de aplicação não se restringem somente à sinalização e iluminação, mas a inúmeros dispositivos e sistemas presentes no veículo integrados via rede ou controlados eletronicamente.

REFERÊNCIAS

ARDUINO. **About Us.** [S.l]: Arduino, 2018. Disponível em: < https://www.arduino.cc/en/Main/AboutUs>. Acesso em: 18 jun. 2018.

_____. **ARDUINO UNO REV3.** [S.l]: Arduino, 2018. Disponível em: < https://store.arduino.cc/usa/arduino-uno-rev3>. Acesso em: 18 jun. 2018.

ATMEL. **Atmel 8-bit Microcontroller**. p. 670, 2013. Disponível em: <http://www.atmel.com/Images/Atmel-8271-8-bit-AVR-Microcontroller-ATmega48A-48PA-88A-88PA-168A-168PA-328-328P_datasheet.pdf>.

BARRAGÁN, Hernando. **The Untold History of Arduino**. Disponível em: <https://arduinohistory.github.io/>.

BENCH, Henry's. **Arduino CAN Bus Module Pin outs and Schematics**. [S.l.]: [S.d.]. Disponível em: <http://henrysbench.capnfatz.com/henrys-bench/arduino-projects-tips-and-more/arduino-can-bus-module-pin-outs-and-schematics/>.

BEVER. **Smartsteer – Driving without limitations.** [S.n]. Disponível em: < https://www.bevercarproducts.com/en/uploads/producten/producten_product_taalspecifiek/Brochure_SmartSteer_English_1.pdf >. Acesso em: 18 jun. 2018.

BOSCH, Robert. **Manual de Tecnologia Automotiva**. Tradução de Helga Madjderey, Gunter W. Prokesch, Euryale de Jesus Zerbini, Sueli Pfeferman. 25° ed. São Paulo: Edgar Blücher, 2005.

CAVENAGHI. **Controle de comandos ERGON – ERGON CE.** 2018. Disponível em: < https://www.cavenaghi.com.br/controle-de-comandos-ergon-ergon-ce-363/p >. Acesso em: 18 jun. 2018.

CHADE, Jamil. Brasil tem maior expansão de produção de veículos entre principais mercados. **O Estado de São Paulo.** Mar. 2018. Disponível em: < https://economia.estadao.com.br/noticias/geral,brasil-tem-maior-expansao-de-producao-de-veiculos-entre-principais-mercados,70002217301>. Acesso em: 20 abr. 2018.

CIRCUITS TODAY. **Story and History of Development of Arduino.** [S.l]: Circuits Today, 2018. Disponível em:< http://www.circuitstoday.com/story-and-history-of-development-of-arduino>.

CONCEIÇÃO, César Stallbaum. **Da Revolução Industrial à Revolução da Informação: Uma análise evolucionária da Industrialização da América Latina**. 2012.

CORRIGAN, Steve. **Introduction to the Controller Area Network (CAN) – Application Report.** Rev. Ed. Dallas: Texas Instruments, 2016. Disponível em: < http://www.ti.com/lit/an/sloa101b/sloa101b.pdf>. Acesso em: 13 fev. 2018.

CUETO, Bruno. **Protocolo de Comunicação CAN Aplicado à Veículos: Estudo, Projeto e Implementação de um Equipamento com Finalidade Didática.** 2018. 55 f. Trabalho de Conclusão e Curso (Bacharel em Engenharia Elétrica) – Universidade Federal do Paraná, 2018. Disponível em:< http://www.eletrica.ufpr.br/tcc/2018/1s/BRUNO%20CUETO/RELATORIO%20TCC%20BRUNO%20CUETO%202018-1.pdf >. Acesso em: 16 set. 2018.

CURVELLO, André. **Intel Edison - Lançamento do módulo IoT da intel**. [S.l], 2014. Disponível em: <https://www.embarcados.com.br/intel-edison-modulo-iot-da-intel/>.

ESPRESSIF SYSTEMS. **About Espressif.** Shanghai: Espressif Systems, 2018. Disponível em:< https://www.espressif.com/en/company/about-us/who-we-are>.

FILIPEFLOP. **CNC Shield V3 para Arduino Impressora 3D**. [S.l.], 2018. Disponível em: < https://www.filipeflop.com/produto/cnc-shield-v3-para-arduino-impressora-3d/>.

_____. **Display LCD Shield com Teclado para Arduino**. [S.l.], 2018. Disponível em: <https://www.filipeflop.com/produto/display-lcd-shield-com-teclado-para-arduino/>.

_____. **Módulo CAN BUS MCP2515 TJA1050**. [S.l.], 2018. Disponível em: < https://www.filipeflop.com/produto/modulo-can-bus-mcp2515-tja1050/>.

_____. **Módulo Relé 5V 2 canais**. [S.l.], 2018. Disponível em: < https://www.filipeflop.com/produto/modulo-rele-5v-2-canais/>.

_____. **Motor Shield L293D Driver Ponte H para Arduino**. [S.l.], 2018. Disponível em: < https://www.filipeflop.com/produto/motor-shield-l293d-driver-ponte-h-para-arduino/>.

GARCIA, Vera. por NOGUEIRA, Sergio A. G. Blog da inclusão e Cidadania. **Veja os vários tipos de adaptações veiculares.** [S.l]: [S.n], 2009. Disponível em:< https://www.deficienteciente.com.br/adaptacao-de-veiculos-parte-1.html>. Acesso em: 17 jun. 2018.

GUIDOSIMPLEX. Manuale Utente. **Telecomando a Infrarossi 1096.** Roma: Guidosimplex, 2018. Disponível em: <https://www.guidosimplex.it/pdf/manualiutenteita/MnUT_1096%20-%20Rev1.pdf >. Acesso em: 17 jun. 2018.

GUIMARÃES, Alexandre de Almeida. **Eletrônica Embarcada Automotiva.** 1ªed. São Paulo: Érica, 2012.

HELIO. **Comandos elétricos no volante Asaflex.** 2018. Disponível em: < https://www.heliomobilidade.com.br/direcao/comandos-eletricos%20/comandos-eletricos-de-volante-asaflex-final >. Acesso em: 18 jun. 2018.

INSTITUTO BRASILEIRO DE GEOGRAFIA E ESTATÍSTICA. **Censo Demográfico 2010: Características gerais da população, religião e pessoas com deficiência.**

Disponível em: <https://ww2.ibge.gov.br/home/estatistica/populacao/censo2010/caracteristicas_religiao_deficiencia/caracteristicas_religiao_deficiencia_tab_uf_xls.shtm. Acesso em: 17 jun. 2018.

INTEL. **MCS-48 Family of Single Chip Microcomputer - User's Manual**. Santa Clara: Intel Corporation, 1978. Disponível em: <http://bitsavers.informatik.uni-stuttgart.de/components/intel/8048/9800270D_MCS-48_Family_Users_Manual_Jul78.pdf>.

KALOGIANNI, Alexander. Top Car Trends of CES 2018. **Digital Trends.** 22 jan. 2018. [S.l.]: [S.n.]. Disponível em: < https://www.digitaltrends.com/cars/top-car-trends-of-ces/>. Acesso em: 27 fev. 2018.

KIVI. Encarte do Produto. **PV3000. Central de Comandos Auxiliares.** Belo Horizonte: KIVI, 2018. Disponível em: <http://www.kivi.com.br/media/prodotti/central_de_comandos_auxiliares_pv3000/download/06B%20-%20PV3000%20kivi-brasil.pdf >. Acesso em: 17 jun. 2018.

LINDSAY, Jon Williams; Jeff Martin; Ken Gracey; Aristides Alvarez; Stephanie. **BASIC Stamp Syntax and Reference Manual**. v 2.2 ed. [S.l.]: Parallax Inc., 2005. Disponível em: <https://www.parallax.com/sites/default/files/downloads/27218-Web-BASICStampManual-v2.2.pdf>.

LODGESONS. **R200 Keypad Controls.** 2018. Disponível em: < http://www.lodgesons.co.uk/products/R200-keypad-controls >. Acesso em: 18 jun. 2018.

LOSCHI, Marília. Pessoas com deficiência: adaptando espaços e atitudes. **Agência IBGE Notícias.** Set. 2017. Disponível em: < https://agenciadenoticias.ibge.gov.br/agencia-noticias/2012-agencia-de-noticias/noticias/16794-pessoas-com-deficiencia-adaptando-espacos-e-atitudes.html>. Acesso em 17 jun. 2018.

MICROCHIP. **MCP2515 – Stand-Alone CAN Controller With SPI™ Interface**. [S.l]: Microchip Technology Inc., 2005. Disponível em: http://ww1.microchip.com/downloads/en/DeviceDoc/21801d.pdf. Acesso em: 27 fev. 2018.

MOTABILITY. **Adaptations Avaiable on the Scheme.** [S.l.]: Motability Scheme, 2018. Disponível em: < https://www.motability.co.uk/cars-scooters-and-powerchairs/adaptations-overview/adaptations-available-on-the-scheme/ >. Disponível em: 17 jun. 2018.

NOERGAARD, Tammy. **Embedded Systems Architecture: A Comprehensive Guide for Engineers and Programmers**. [S.l: s.n.], 2013.

ORGANISATION INTERNATIONALE DES CONSTRUCTEURS D'AUTOMOBILES. **2017 Production Statistics.** Disponível em:

<http://www.oica.net/category/production-statistics/2017-statistics/>. Acesso em: 20 abr. 2018.

PARAVAN GmbH. Paravan Voice Control: The new voice control for the secondary functions. [S.l]: Paravan GmbH, 2018. Disponível em: < https://www.paravan.de/en/product-solutions/the-voice-control-for-secondary-vehicle-functions/ >. Acesso em: 17 jun. 2018.

PARAVAN GmbH. **Die neue Sprachsteuerung von PARAVAN Voice Control – Mit Sprache zaubern.** 2015. (3m09s). Disponível em:< https://www.youtube.com/watch?v=-anLHhgL8sA >. Acesso em: 17 jun. 2018.

PENIDO, Édilus de C. C; TRINDADE, Ronaldo S. **Microcontroladores.** Ouro Preto: Rede e-Tec Brasil, 2013.

PHILIPS. **TJA1050.** High speed CAN transceiver. [S.l.]: Philips Electronics, 2003.

RASPBERRY PI FOUNDATION. About Us. [S.l]: Raspberry Pi Foundation, 2018. Disponível em:< https://www.raspberrypi.org/about/>.

SILVA, Edna Lúcia da; MENEZES, Estera Muskat. **Metodologia da Pesquisa e Elaboração de Dissertação.** 4° ed. rev. atual. Florianópolis: UFSC, 2005.

STMICROELECTRONICS. **STM32 MCU Nucleo.** [S.l]: STMicroelectronics, 2018. Disponível em:< https://www.st.com/en/evaluation-tools/stm32-mcu-nucleo.html?querycriteria=productId=LN1847>.

TANENBAUM, Andrew S. **Computer Networks.** 5th ed. Boston: Pearson, 2011.

TEXAS INSTRUMENTS. **BeagleBone Development Board.** [S.l]: Texas Instruments Incorporated, 2018. Disponível em: < http://www.ti.com/tool/BEAGLEBN>.

TODAY, Circuits. **Story and History of Development of Arduino**. Disponível em: <http://www.circuitstoday.com/story-and-history-of-development-of-arduino>.