Instituto Politécnico
de Castelo Branco
Escola Superior
de Tecnologia

E-Scooter
Desenvolvimento de uma Trotinete Elétrica

Keven Lourenço M. de Carvalho Nº20210104

Orientador

Professor José António Barros Vieira

Trabalho de Projeto apresentado à Escola Superior de Tecnologia do Instituto Politécnico de Castelo Branco para cumprimento dos requisitos necessários à obtenção do grau de Licenciado em Engenharia Eletrotécnica e das Telecomunicações, realizada sob a orientação científica do Professor Doutor José Vieira, do Instituto Politécnico de Castelo Branco.

Outubro 2024

Composição do júri

Presidente do júri

Doutor José António Barros Vieira

Professor Adjunto da Escola Superior de Tecnologia, Instituto Politécnico de Castelo Branco

Vogais

Doutor António José Cerejo da Silva

Professor Adjunto da Escola Superior de Tecnologia, Instituto Politécnico de Castelo Branco

Doutora Paula Cristina Alves Pereira

Professora Adjunta da Escola Superior de Tecnologia, Instituto Politécnico de Castelo Branco

Dedicatória

Dedico este trabalho à minha querida família, especialmente aos meus pais, por todo o amor, apoio e por acreditarem em mim em todos os momentos. Sem a vossa paciência, incentivo e sacrifícios, esta conquista não seria possível. À minha irmã, pela constante inspiração e por ser sempre uma fonte de motivação e alegria.

Dedico também aos meus professores, em particular ao Prof. José Vieira, cuja orientação e sabedoria foram essenciais para o desenvolvimento deste projeto. O vosso compromisso com o ensino e o apoio contínuo foram fundamentais para que eu chegasse até aqui.

Agradecimentos

Gostaria de expressar minha sincera gratidão a todos aqueles que, de uma forma ou outra, contribuíram para que eu pudesse alcançar este objetivo. Primeiramente, gostaria de agradecer ao meu orientador, Prof. José Vieira, por toda disponibilidade mostrada e pelos seus esforços para solucionar todas as questões e obstáculos que surgiram pelo caminho.

Agradeço à minha família, em especial aos meus pais e à minha irmã, por todas as oportunidades que me têm dado, sempre me apoiando e motivando para fazer melhor ao longo de todo o curso, mesmo quando tudo parecia mais difícil, e pensei em desistir.

A todos os meus amigos, dentro e fora do IPCB, agradeço pelos momentos passados e partilhados e pelo incentivo constante. Em especial, aos meus colegas Denílson Miranda, Máxime Gonçalves e Miguel Dunge que estiveram presentes durante grande parte da realização deste projeto, dando seu suporte.

A todos, muito obrigado!

Resumo

A trotinete elétrica tornou-se uma solução de mobilidade urbana prática e sustentável, oferecendo uma alternativa eficiente para deslocamentos de curta e média distância.

Este projeto visa o desenvolvimento de uma trotinete elétrica utilizando uma estrutura mecânica construída a partir de bicicletas infantis, com o objetivo de criar um veículo econômico e sustentável, fazendo a adaptação de um motor elétrico de 250W, baterias LiPo, um controlador, um sensor de efeito hall, etc.

Facto é que protótipo foi concretizado, onde sistema de controle de velocidade foi implementado com base em um controlador PWM. Testes feito mostram que a trotinete pode atingir velocidades de 18 km/h, com uma autonomia superior a 20 km, para além de implementar funcionalidades, como o *cruise control, e monitorização da bateria,* que oferecem mais conforto e acessibilidade.

Palavras-chave

Trotinete elétrica;

Sustentabilidade;

Cruise control;

Microcontrolador;

Sistema.

Abstract

The electric scooter has become a practical and sustainable urban mobility solution, offering an efficient alternative for short and medium-distance commutes.

This project aims to develop an electric scooter using a mechanical structure built from children's bicycles, with the goal of creating an economical and sustainable vehicle. The adaptation includes a 250W electric motor, LiPo batteries, a controller, a Hall effect sensor, and more.

The prototype was successfully realized, with a speed control system implemented based on a PWM controller. Tests show that the scooter can reach speeds of 18 km/h, with a range of over 20 km, and it also includes features like cruise control and battery monitoring, providing more comfort and accessibility.

Keywords

Electric scooter;

Sustainability;

Cruise control;

Microcontroller;

System.

Índice geral

Índice de figuras

Lista de tabelas

Lista de Abreviaturas, sigla e acrónimos

mA/h – miliampere/hora, determina a corrente elétrica fornecida numa hora;

LED - *Light Emitting Diode* (Diodo Emissor de Luz);

LCD - *Liquid Crystal Display* (Display de Cristal Líquido);

V – Volt, unidade de medida da tensão;

W – Watt, unidade de medida de potência;

μC – microcontrolador;

DC - Corrente Contínua (*Direct Current*);

PWM - Modulação por Largura de Pulso (*Pulse Width Modulation*);

DPST – *Double pole, Single Throw;*

SPP – *Bluetooth Serial Port Profile;*

1. Introdução

1.1 Motivação

Em meio ao aumento das preocupações com as mudanças climáticas, a poluição do ar e a necessidade de reduzir a dependência de combustíveis fósseis, desenvolveu-se uma consciência global em relação à sustentabilidade e à busca por soluções de transporte alternativas e ecológicas, os veículos elétricos. Em meio a todo esse caos, as trotinetes elétricas emergiram como uma opção prática, acessível e amiga do ambiente.

Além de oferecerem uma alternativa ao transporte público e aos automóveis, as trotinetes elétricas proporcionam uma forma de deslocamento ágil em áreas urbanas congestionadas, contribuindo para a redução do tráfego e das emissões de gases de para a atmosfera.

Associando o facto de a razão primordial residir no compromisso com a sustentabilidade ambiental, com o facto das trotinetes comerciais apresentarem um custo elevado e uma vida útil limitada, propôs-se neste projeto o desenvolvimento de uma trotinete elétrica, numa estrutura mecânica construída a partir de peças reutilizadas de bicicletas infantis, voltando assim a primitiva da questão.

1.2 Objetivos

Com a realização deste projeto, pretende-se alcançar alguns objetivos de diferentes naturezas.

Na ótica pessoal, o objetivo é aprofundar e aplicar os conhecimentos adquiridos na área de eletrónica e programação. Não só, pretende-se também, adquirir e aprimorar habilidades práticas em setores desconhecidos, ou pouco aprimorados como a soldagem.

A nível do projeto em si, a meta é criar um sistema eletrónico estável e funcional, que nos permita locomover, através da propulsão dada por um motor DC alimentado por baterias de lítio, onde podemos controlar e visualizar através de uma interface gráfica (um telemóvel), parâmetros como a velocidade, a carga das baterias, e mais.

Ao montar este protótipo, pretende-se provar que, usando materiais recicláveis, de baixo custo, ou provenientes de outro veículo, acompanhados de componentes eletrónicos e de um certo conhecimento na área, é possível construir uma trotinete elétrica com um desempenho aceitável, embora compreensível se for inferior ao das comerciais, sendo totalmente sustentável, económica e eficaz.

1.3 Estrutura do relatório

Este relatório é composto por 8 capítulos, incluindo o da Introdução, da qual esta secção faz parte. Neste primeiro capítulo, são citadas as motivações que levaram a escolha deste projeto, e os objetivos que se pretendem alcançar. E como neste projeto utilizou-se uma estrutura mecânica feita por um colega um ano atras, é feita a descrição dessa estrutura.

O segundo capítulo aborda o estado da arte, onde é descrita a situação atual das trotinetes elétricas, mencionando os últimos avanços, as leis aplicadas, e o seu impacto na sociedade e no ambiente.

No terceiro e quarto capítulo, são listados todos os recursos de hardware e ferramentas de software, respetivamente, utilizadas para a implementação e desenvolvimento deste projeto, e seu papel nele.

O quinto capítulo se refere a arquitetura do sistema, citando e ilustrando as ligações que se fizeram essenciais, para o funcionamento de todo o sistema.

Já no sexto capítulo denominado Operação e Desempenho Tecnológico, se explica as funcionalidades aplicadas, os modos de funcionamento, e se apresentam estudos feitos de modo a compreender e classificar o desempenho do sistema

Por último temos as conclusões finais do trabalho realizado e sugestões de ideias de melhoramento para trabalhos futuros.

1.4 Estrutura mecânica base

Como já foi mencionado, a implementação deste projeto, foi realizada tendo como referência uma estrutura mecânica já elaborada. Mas antes de começar a projetar o nosso sistema é necessário conhecer as partes que compõem essa estrutura, e obviamente compreendê-la como um todo.

Figura 1 - Trotinete Elétrica

De toda a estrutura física da trotinete, existem algumas partes que são mais essenciais a sua compreensão, do que outras. E é nessas partes que vamos focar.

1.4.1 Quadro da trotinete

O quadro da trotinete, consiste numa estrutura no formato de um paralelepípedo feita com vigas angulares metálicas (ferro), que ao ser fechado no fundo e nas laterais, se assemelhou a uma caixa, onde foram depositadas a maioria das componentes essenciais do protótipo.

Esse quadro apresenta uma estrutura firme e robusta, pois não só armazena as componentes *core* do sistema, como, é sobre ela que o utilizador mantém o seu apoio, incidindo diretamente nele todo o seu peso.

Figura 2 - Quadro da Trotinete

1.4.2 Rodas da trotinete

As rodas presentes na estrutura da trotinete, têm origem de bicicletas, porém infantis, o que significa que não tem uma dimensão extrema, mas ainda sim, considerável.

Figura 3 - Roda da trotinete

Cada uma das rodas tem um raio (r) de aproximadamente 17 cm (0.17m).

(1.1) $$Perimetro\ da\ roda = 2\pi * r$$

$$\Leftrightarrow \quad Perimetro\ da\ roda = 2\pi * 0.17$$

$$\Leftrightarrow \quad Perimetro\ da\ roda \cong 1.07\ metros$$

1.4.3 Posição do Motor

O motor usado para esse protótipo se encontra soldado, na extremidade traseira da parte inferior do quadro central, onde se encontram as componentes. Essa é uma localização estratégica, pois fica posicionada convenientemente próxima a roda traseira, na qual é dada a tração.

Figura 4 - Localização do motor

2. Estado da Arte

Nos últimos anos, as trotinetes elétricas emergiram como uma solução de mobilidade urbana inovadora e sustentável. Com o aumento da preocupação com as emissões de carbono e a necessidade de alternativas de transporte mais eficientes, a popularidade das trotinetes elétricas cresceu exponencialmente. Este capítulo explora o estado da arte das trotinetes elétricas, abordando pontos como: avanços tecnológicos, as questões de regulamentação e os impactos sociais e ambientais dessa forma de transporte, e o custo de aquisição deste bem.

2.1 Avanços Tecnológicos

As trotinetes elétricas passaram por diversos avanços desde o seu surgimento até os dias atuais, refletindo inovações tecnológicas, melhorias na eficiência e um aumento na popularidade como meio de transporte urbano. No entanto têm áreas onde esses avanços são mais evidentes, como:

- O desenvolvimento de baterias de alto desempenho, de lítio, mais leves e de maior capacidade, que permitiu que as trotinetes elétricas aumentassem sua autonomia. Modelos recentes oferecem uma autonomia na faixa de 20 a 80 km, dependendo do peso do utilizador e do terreno.

- Os motores elétricos passaram por melhorias significativas em eficiência e potência. Motores de 250 a 800 watts são comuns, permitindo velocidades de até 25 km/h em áreas urbanas. Além disso, inovações em tecnologia de controle de motor têm melhorado a resposta e a experiência de condução.

- A incorporação de tecnologias IoT (*Internet of Things*), permite que as trotinetes elétricas se conectem a smartphones, oferecendo funcionalidades como rastreamento GPS, monitoramento de saúde da bateria e compartilhamento de dados de uso.

Estes são alguns dos avanços mais significativos, que as trotinetes elétricas tiveram nos últimos tempos, mas existem ainda muitas outras, tanto a nível de Design como de Segurança. Então podemos constatar que, com o contínuo desenvolvimento de tecnologias e a crescente aceitação em ambientes urbanos, as trotinetes elétricas prometem se tornar uma parte cada vez mais importante da mobilidade moderna.

2.2 Impacto Social e Ambiental

O uso de trotinetes elétricas tem implicações significativas tanto sociais quanto ambientais, podendo estas ser divididas em três áreas:

- Sustentabilidade;
- Desafios Urbanos;
- Acessibilidades e Mobilidade.

a) Sustentabilidade

As trotinetes elétricas oferecem uma alternativa de transporte que pode reduzir a dependência de veículos movidos a combustíveis fósseis. Estudos demonstram que a adoção generalizada de trotinetes elétricas tem contribuído significativamente para a redução das emissões de carbono nas cidades.

b) Acessibilidade e Mobilidade

As trotinetes elétricas proporcionam uma opção de transporte acessível para muitas pessoas, especialmente em áreas urbanas congestionadas. Elas oferecem uma solução eficaz para percursos curtos e complementam o transporte público, facilitando a mobilidade.

c) Desafios urbanos

Apesar dos inúmeros benefícios, as trotinetes elétricas também apresentam desafios, como o aumento do congestionamento em calçadas e a necessidade de infraestrutura adequada. As cidades precisam se adaptar, implementando faixas exclusivas e estacionamentos para trotinetes para garantir a segurança de todos os utilizadores, e pedestres.

2.3 Regulamentação e Segurança

Com o aumento do uso de trotinetes elétricas, surgiram questões relacionadas à regulamentação e segurança destas. Com isso deu-se o surgimento de normas e regulamentações para o uso de trotinetes elétricas em áreas urbanas, incluindo restrições que contribuem para a segurança e aceitação pública.

2.3.1 Legislação

Diversas cidades ao redor do mundo estão implementando regulamentações específicas para o uso de trotinetes elétricas. Mas antes de apresentar essas regras, temos de enunciar como a legislação define uma trotinete.

Pelo Código da Estrada de 8 janeiro 2021, no Nº4 do artigo 112˚, «considera-se trotineta o veículo constituído por duas rodas em série, que sustentam uma base onde o condutor apoia os pés, conduzida em pé e dirigida através de um guiador que se eleva até a altura da cintura». Então um veículo com as características mencionadas, e que possua um motor de até 250 W, e que atinja a velocidade máxima num patamar não acima de 25 km/h, está sancionado pelas regulamentações estipuladas pelo Código de Estrada (Artigo 112˚).

Agora serão enumeradas algumas regulamentações estipuladas as trotinetes elétricas pela legislação em vigor, onde:

- Nas ciclovias não é permitida a circulação a mais do que 25 Km/h, nem é permitida a circulação de trotinetes com potência superior a 250W, capazes de ultrapassar a velocidade máxima em patamar de 25 Km/h;
- Trotinetes com potência inferior a 250W e incapazes de ultrapassar a velocidade máxima de 25 km/h, só podem circular na estrada quando não exista uma ciclovia adjacente;
- As trotinetes com motor não podem circular na autoestrada, pois sendo equiparadas a velocípedes, lhe é proibido circular na autoestrada;
- Não é permitido circular a esquerda da via, pois o Artigo 90˚ do Código de Estrada estipula que «os condutores de velocípedes devem transitar pelo lado direito da via conservando das bermas ou passeios uma distância que permita evitar acidentes».
- É obrigatório o uso de luzes de presença a noite.

2.3.2 Sistemas de Segurança

Inovações em segurança, como sistemas de travagem regenerativa e faróis LED, têm sido integradas para melhorar a visibilidade e a segurança dos utilizadores. Alguns modelos também incluem recursos de rastreamento para auxiliar na recuperação em caso de roubo.

3. Recursos de *Hardware*

Neste capítulo, serão apresentados todos os dispositivos, componentes de *hardware* e ferramentas utilizadas durante o desenvolvimento deste projeto, onde a seleção dos materiais foi realizada com base nas necessidades específicas do sistema projetado, garantindo o desempenho adequado, a eficiência energética e a segurança da estrutura.

A escolha criteriosa dos componentes de *hardware*, como o motor, as baterias e o controlador, foi essencial para alcançar os objetivos de funcionalidade e confiabilidade. A seguir, serão descritos detalhadamente todos os dispositivos envolvidos na execução do projeto, destacando suas especificações técnicas e suas respetivas funções no sistema.

3.1 Placa de demonstração PICDEM 2 Plus

A PICDEM 2 Plus é uma placa de demonstração (*demo board*) desenvolvida pela Microchip Technology para facilitar o desenvolvimento, teste e prototipagem de aplicações baseadas em microcontroladores PIC. Ela foi projetada especificamente para desenvolvedores que desejam experimentar e aprender a programar microcontroladores PIC da família de 8 bits, incluindo PIC16F e PIC18F.

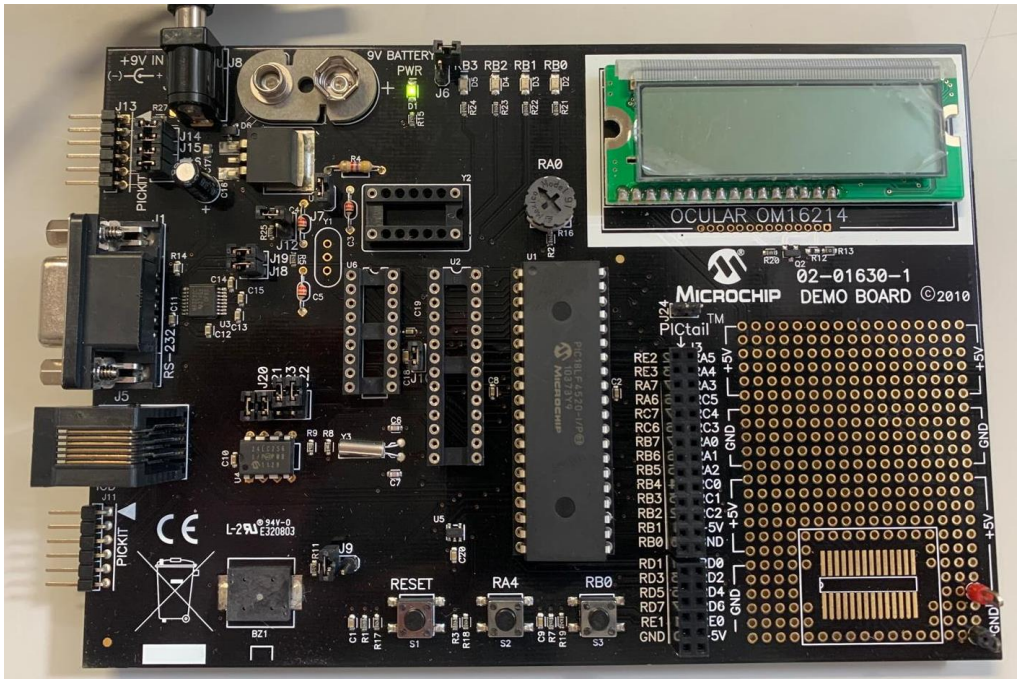


Figura 5 - Placa de demonstração PIKDEM2 Plus

Como podemos observar pela figura, a placa vem equipada com uma série de periféricos e componentes que facilitam o desenvolvimento de projetos com microcontroladores PIC, como é o caso deste. É importante realçar a relevância destes periféricos e componentes disponibilizados pela placa, onde a maioria destes desempenha um papel ativo e essencial para o sistema.

Na figura abaixo encontrasse representada a estrutura da placa, realçando os recursos de hardware disponibilizados por esta, seguido da legenda dos mesmos.

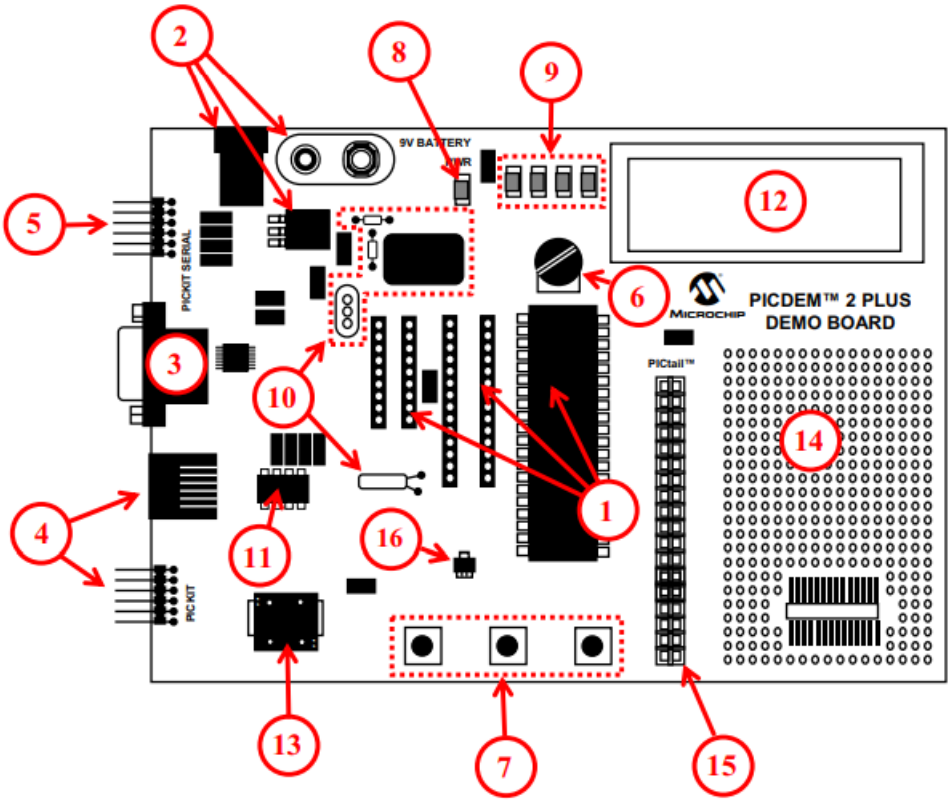


Figura 6 - Recursos de hardware do PIKDEM2 Plus

Recursos de *hardware* presentes na placa:

1. DIP *sockets* de 18, 28 e 40 pinos (apenas 1 *socket* pode ser usado por vez);
2. Adaptador para uma fonte AC/DC de 9V e 100 mA, ou uma bateria de 9V;
3. Porta de comunicação RS-232;
4. Portos para programador/depurador(debugger), como o PICkit2;
5. Conector PICkit série, para analise de periféricos de comunicação série;
6. Potenciómetro de 5 kΩ;
7. Três botões de pressão, sendo um para reiniciar a placa, e outros dois para interação externa;
8. LED indicador de energia;
9. Quatro LEDs ligados ao Porto B;
10. Osciladores internos integrados;
11. Memoria EEPROM série 32K x 8;

12. *Display LCD;*
13. *Buzzer* piezoelétrico;
14. Área de protótipo, para *hardware* do utilizador;
15. Portos de acesso aos pinos do microcontrolador;
16. Sensor térmico Microchip Tc74.

3.2 Microcontrolador PIC18F4520 I/P

O PIC (*Peripheral Interface Controller*) é uma família de microcontroladores desenvolvidos pela Microchip Technology. Eles são componentes integrados amplamente utilizados em projetos de eletrónica devido à sua versatilidade, eficiência e facilidade de programação.

O PIC18F4520 I/P é um microcontrolador de 8 bits da família PIC18, sendo uma versão do microcontrolador PIC18F4520 da Microchip Technology, que vem em um encapsulamento PDIP (*Plastic Dual In-line Package*). O sufixo I/P geralmente indica que o dispositivo está disponível em um pacote de 40 pinos, numa configuração de montagem através de furos, comum em aplicações de protótipos e desenvolvimento.

Ele utiliza uma arquitetura RISC (*Reduced Instruction Set Computing*), permitindo uma execução rápida e eficiente das instruções.

Segue abaixo algumas características relevantes do microcontrolador em questão, seguido da sua ilustração e do seu *pinout.*

- Arquitetura RISC otimizada;
- Pode operar com uma frequência de *clock* de até 40 MHz;
- Memória Flash: 32 KB;
- Memória SRAM: 1.536 bytes;
- Memória EEPROM: 256 bytes;
- Possui 33 pinos de I/O configuráveis,
- Resolução ADC de 10 bits;
- Suporta 13 canais ADC;
- Tensão de operação típica entre 2.0V e 5.5V;
- Temperatura de operação: de -40°C a +85°C, permitindo o uso em ambientes exigentes;

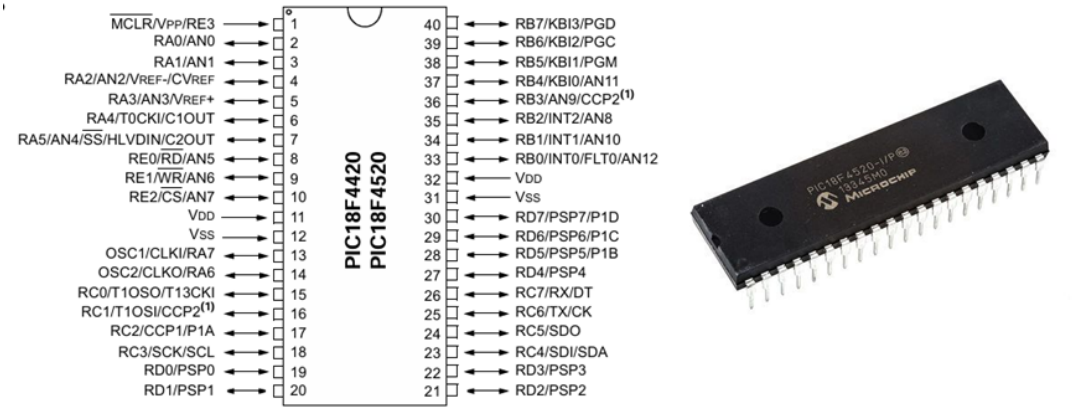


Figura 7 - PIC18F4520 I/P e seu Pinout

3.3 Baterias LiPo

Baterias LiPo (*Lithium Polymer*) são um tipo de bateria recarregável, composta por polímeros de lítio, amplamente usada em aplicações que exigem alta densidade de energia, baixo peso e formatos compactos.

Para a realização deste projeto, foram usadas seis baterias LiPo, da marca TopFuel, como fonte de energia para o nosso sistema. Onde cada uma delas apresenta as seguintes características/especificações:

- Tensão nominal 14,8 V – que é tensão elétrica média que ela fornece durante a maior parte do seu ciclo de descarga;
- Capacidade 2400 mA/h;
- 4S – composto por quatro células de lítio, ligadas em série;
- Taxa de descarga 20 C – pode fornecer até 20 vezes a sua capacidade nominal de uma vez (20*2.4A=48A);
- 35,5 Wh – capacidade de energia que ela pode fornecer;

Tendo em vista as características da bateria, alas podem ser unidas em série ou em paralelo, de modo a ajustar a tensão ou a corrente, respetivamente, de acordo com a demanda. Só que, é importante frisar que cada célula da bateria, quando totalmente carregada, apresenta uma tensão de 4.2 V, então a tensão máxima real da bateria é de 16,8 V (4 * 4.2 V = 16.8V).

No caso da trotinete, o motor precisa de uma tensão igual ou superior a 24 V, e para isso, uniu-se dois conjuntos de baterias em série, duplicando a tensão, e três em paralelo, obtendo maior capacidade da corrente, mais especificamente o triplo dela.

(3.1) $T_{TOTAL} = 16.8\,V * 2$

$\Leftrightarrow \quad T_{TOTAL} = 33{,}6\,V$

(3.2) $I_{TOTAL} = 2400\,mA * 3$

$\Leftrightarrow \quad I_{TOTAL} = 7200\,mA$

Esta é a descrição das ligações feitas entre as baterias, e os parâmetros que essa montagem oferece. Segue nas figuras abaixo uma ilustração das baterias usadas, e da montagem feita.

Figura 9 - Baterias LiPo

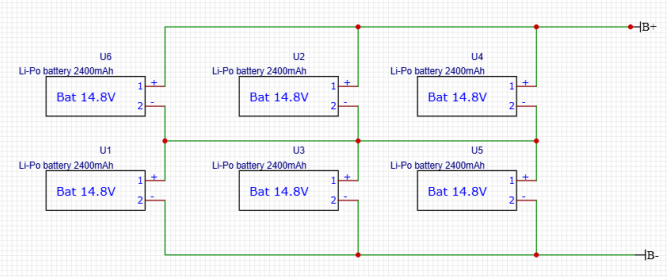


Figura 8 - Esquema de Ligação das baterias

3.4 Módulo de sensor de Hall

É um dispositivo que integra um sensor de campo magnético hall, que usa o efeito Hall para detetar a aproximação de um campo magnético do sensor, gerando em sua saída um valor analógico dependendo da força do campo, e do tipo de construção do modulo. Ele pode ser usado para detetar a velocidade de um íman, como é este o caso, mas falaremos disto mais a frente.

Especificações do sensor:

Tabela 1 *- Especificações do Sensor de efeito hall*

Tensão de operação	4.5 a 6 Volts contínuos
Corrente de operação	4.2 mA
Tipo de saida	linear
Temperatura de operação	-40^0 a 85^0 celsius
Peso	1g

Quanto aos pinos, o modulo só possui três, sendo bem simples de introduzi-lo no seu sistema. Abaixo temos uma figura do modulo, especificando os seus pinos.

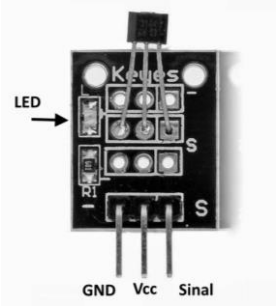


Figura 10 - Módulo do sensor de efeito Hall

3.5 Módulo Bluetooth

Foi utlizado um modulo Bluetooth HC-05, que é um dispositivo amplamente utilizado para a comunicação sem fio em projetos de eletrônica, que envolvam a comunicação com/entre microcontroladores, como PIC, Arduino e Raspberry Pi.

No projeto ele conseguiu desempenhar o papel pretendido, estabelecendo a comunicação com um dispositivo movel, pela App BlueSPP, e comunicando com o PIC pelos pinos Tx e Rx (*Port_C7* e *Port_C6*).

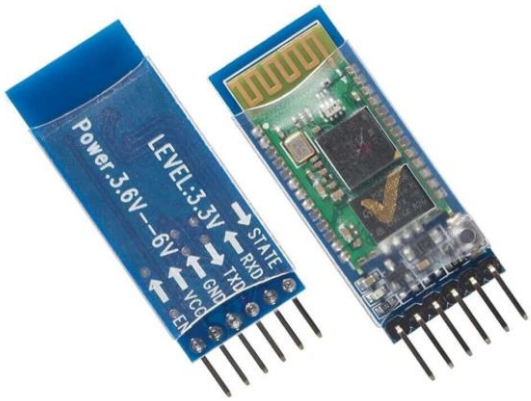


Figura 11 - Módulo Bluetooth HC-05

Dentre as características do modulo, podemos destacar algumas, como:

- Tipo de Módulo: Bluetooth Classe 2, com um alcance de até 10 metros (dependendo do ambiente);
- *Interface*: Série UART (Universal Asynchronous Receiver-Transmitter), permitindo comunicação simples com microcontroladores;
- Tensão de Operação: 3,3 V a 5 V, com um consumo de corrente variando de 30 mA a 40 mA;
- Configuração: Pode ser configurado como mestre ou escravo;
- Taxa de Transferência: Suporta taxas de até 115200 bps;
- Compatibilidade: Funciona com a maioria dos microcontroladores, incluindo Arduino, PIC, Raspberry Pi, entre outros.

Apesar do modulo ser alimentado com 5 V, nos pinos de transmissão ele opera até 3,3V. Devido a isso, no pino Rx onde ele recebe os dados provenientes do PIC, teve de se implementar um divisor de tensão, baixando-a de 5 V, para 3,3 V.

Na figura abaixo, que representa o esquema do modulo com as suas respetivas ligações, pode-se visualizar o divisor de tensão aplicado, e as devidas resistências utilizadas.

Tabela 2 *- Pinos do modulo bluetooth*

Pinos	Função
VCC	Alimentação (3,3V a 5V)
GND	Terra (GND)
TX	Transmissão de Dados (UART)
RX	Receção de Dados (UART)
STATE	Indica o estado do Bluetooth (*ON/OFF*)
EN	Modo de configuração

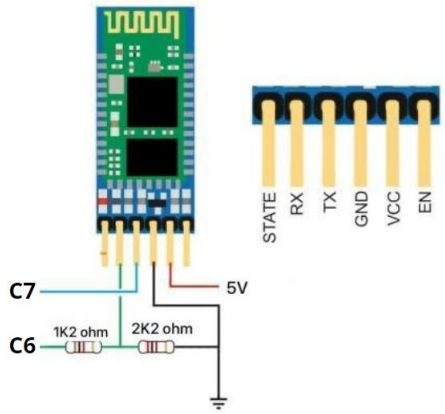


Figura 12 - Esquema de Ligação do módulo Bluetooth HC-05

3.6 Conversor DC-DC

No projeto foi usado um módulo regulador de tensão LM2596, que trabalha como um conversor DC-DC no modo *Step Down,* onde ele recebe a entrada uma tensão contínua, e converte-a à saída numa de valor menor. Sendo mais específico, o seu papel no nosso sistema é pegar na tensão gerada pelas baterias (a volta dos 30 V), e baixá-la para 9 V contínuos, nos disponibilizando assim, uma tensão que nos permite alimentar a placa PICDEM2 Plus, e as demais componentes.

Figura 13 - Módulo conversor DC-DC step down

Pela figura, podemos observar que no módulo constam diversos componentes, como o *chip* regulador LM2596S, condensadores (de entrada e saída), um potenciómetro ajustável, LED indicador de funcionamento, entre outros. Estes componentes possibilitam que o modulo desempenhe o seu papel, com as seguintes propriedades:

- A tensão de entrada: 4 v - 35 V;
- - Tensão de saída: 1,23 V - 30 V;
- - Corrente de saída: 2A corrente nominal, 3A máxima;
- - Eficiência de conversão: 92% (quanto maior a tensão de saída maior a eficiência);
- - Frequência de comutação, 150 KHZ;
- - A ondulação da saída: 30 mv (máximo);
- - Regulação de carga: mais ou menos 0,5%;
- - Temperatura de trabalho: - 40 °C a + 85 °C

3.7 Controlador BTS7960

O Módulo BTS7960 é um *driver* de motor de corrente contínua (DC) de alta potência, projetado para controle de motores em aplicações de robótica, automação e outros projetos que requerem controle preciso de motores. Basicamente ele é um dispositivo que regula a potencia entregue ao motor, através de um sinal de controlo PWM, sendo este gerido pelo microcontrolador PIC18F4520, que se encontra afixado numa placa PICDEM2 Plus.

Apesar de neste protótipo, impulsionarmos o motor a rodar apenas num sentido, o BTS7960 é um *driver H-Bridge*, o que significa que ele pode controlar a direção de um motor DC, permitindo que ele gire em ambos os sentidos (horário e anti-horário).

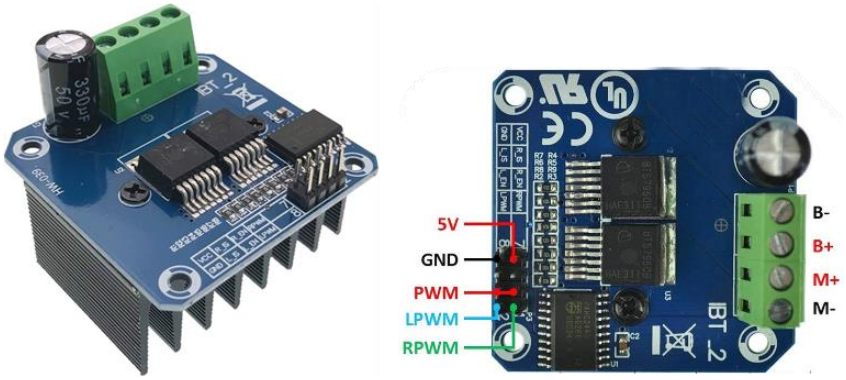


Figura 14 - Módulo do controlador BTS7960 e pinos usados

Como podemos ver pela figura, o módulo possui alguns periféricos de entada e saída, os quais são essências para o seu bom funcionamento, e a sua compreensão é imprescindível para uma boa implementação. Na tabela abaixo, estão listados os pinos presentes no modulo, e as suas funções.

Tabela 3 - *Funções dos pinos do controlador*

Pinos	Finalidade	
VCC	Alimentação do circuito de controlo (geralmente 5V)	
GND	Massa	
RPWM e LPWM	Pinos de entrada PWM, para controlar o sentido (*Right e Left) e intensidade* de rotação do motor	
R_EN e L_EN	Pinos de habilitação (*enable*) para controlo de direção (*Right Enable e Feft Enable)*	
R_IS e L_IS	Pinos de saída de corrente para monitoramento	
B+	Pino onde é conectado o terminal positivo das baterias (alimentação). Fornece a tensão de operação	
B-	Pino onde é conectado o terminal negativo das baterias. Serve como referência de massa	
M+	Pino conectado ao terminal positivo do motor	Responsáveis por impulsionar o motor, com a velocidade e direção determinada
M-	Pino conectado ao terminal negativo do motor	

3.8 Motor

No protótipo do projeto, foi anexado um motor de corrente contínua (DC) com escovas, que opera a uma tensão de 24 V, e uma potência de 250 W. Neste caso em específico temos o motor a operar ligeiramente acima da tensão recomendada, pois a configuração das baterias disponibiliza, quando totalmente carregadas, 33,6 V.

Ele é um motor de 3000rpm (rotações por minuto), porem apresenta uma relação de redução de 9.78:1, ou seja, para engrenagem externa completar uma volta, a interna tem de girar aproximadamente dez vezes. Posto isso, podemos listar algumas características do nosso motor:

- Velocidade nominal: 3000 rpm/min;
- Velocidade real: 300rpm (diretamente no pinhão, após a redução);
- Corrente nominal: 13.4 A;
- Eficiência: 78%;
- Relação de redução: 9.78: 1;
- Roda dentada de 9 dentes.
- Peso: 2.6 kg

Segue abaixo uma ilustração de um motor elétrico, semelhante ao usado no protótipo do projeto.

Figura 15 Motor Elétrico 24V DC

3.9 Acelerador ("manopla")

Foi utilizado um acelerador de torção, que consiste num dispositivo que permite controlar a velocidade da trotinete, através de uma variação de tensão. Essa variação de tensão ocorre devido a um potenciómetro presente no acelerador, que tem seus terminais conectados a um circuito elétrico (5V, GND, *Output Signal*), onde a medida que o utilizador gira o acelerador, o potenciómetro altera sua resistência. Isso resulta em uma variação da tensão de saída que é proporcional à posição do acelerador, que está limitada a uma gama de valores entre 0.8V e 4.2V.

O acelerador que geralmente está localizado no guidão da trotinete, próximo das mãos do utilizador, tornando-o facilmente acessível, envia a diferença de tensão para o microcontrolador, que trata o sinal (transforma em PWM), usando-o para controlar a velocidade da trotinete.

Figura 16 - Manopla de aceleração

3.10 Interruptor *ON/OFF*

Foi usado um interruptor DPST (*Double Pole, Single Throw*), do tipo gangorra entre a bateria e o restante do sistema, de modo a poder estabelecer e interromper o fornecimento de energia ao sistema através de um simples clique.

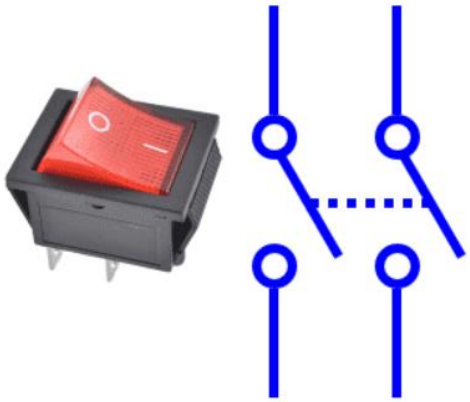


Figura 17 - Interruptor ON/OFF

3.11 Botão de Pressão e *Microswitch*

No projeto, foram utilizados um botão de pressão e um *microswitch* como componentes essenciais para o controle do sistema. Esses dispositivos são bastante simples e eficientes, proporcionando uma maneira prática de interagir com o circuito, e por isso são bastantes comuns em qualquer circuito eletrónico.

Neste momento, não aprofundarei nos detalhes técnicos sobre o funcionamento e as aplicações desses componentes, pois pretendo abordá-los de forma mais completa em um capítulo futuro. Nesse próximo capítulo, exploraremos suas características, especificações e a importância que desempenham no contexto do projeto, permitindo uma compreensão mais ampla de suas funções e contribuições para o sistema como um todo.

Figura 18 - Botão de pressão e microswitch

4. Ferramentas de *Software* utilizadas

No desenvolvimento deste projeto, a escolha e utilização de ferramentas de software adequadas desempenharam um papel crucial para garantir o sucesso na implementação do sistema. Cada um dos aplicativos de *software* selecionados foi fundamental para as diferentes etapas do processo, desde a programação do microcontrolador até a interface de comunicação com o dispositivo final.

A utilização de um compilador robusto e eficiente foi essencial para assegurar que o código desenvolvido fosse compatível com o microcontrolador escolhido, assegurando não só a tradução correta da logica do código para a máquina, como também a otimização do desempenho do microcontrolador em situações adversas. Adicionalmente, o uso de uma ferramenta de gravação e depuração foi indispensável para a fase de implementação do *firmware* no microcontrolador, permitindo não só a transferência do código compilado para o dispositivo0 PIC escolhido, como a correção de falhas de maneira ágil.

Outro ponto chave do projeto foi a necessidade de comunicação entre o sistema desenvolvido e dispositivos móveis, onde um aplicativo de comunicação série via Bluetooth, foi essencial para o monitoramento de parâmetros e desempenho em tempo real, permitindo até o controle remoto.

As ferramentas/aplicativos de *software* utilizados para prover todas as funcionalidades acima mencionada foram, respetivamente, o PIC C Compiler, o PICKIT2 *Programmer* e o BlueSPP. E estes serão abordados de forma singular e mais aprofundada neste capítulo.

4.1 PIC C Compiler

O PIC C *Compiler* é uma ferramenta de software desenvolvida pela CCS, Inc (*Custom Computer Services*), especialmente para a programação de microcontroladores PIC, como o PIC18F4520, em linguagem C. Sendo mais específico ele é um compilador inteligente e altamente otimizado, que contem operadores padrão de funções da linguagem C construídas em bibliotecas que são especificas para registos de PIC.

Os microcontroladores apenas podem ser programados através de um ficheiro de um ficheiro hexadecimal (.hex). Como é de se esperar o homem não escreve os seus programas em hexadecimal, então o compilador traduz o código escrito em linguagem C para código de máquina (.hex), que pode ser executado por microcontroladores PIC.

A escolha do PIC C *Compiler* para este projeto não foi feita de forma inconsciente, foi justificada por diversas razões, tais como:

- Facilidade de Programação: A linguagem C, que é uma linguagem de alto nível, é mais intuitiva e fácil de aprender em comparação com o *Assembly*, que é uma das outras linguagens que podem ser usadas para programar µC PIC;
- Compatibilidade: o PIC C *Compiler* é compatível com a maioria dos microcontroladores PIC, incluindo o PIC18F4520;
- Ferramentas de depuração (*DEBUG*): ele inclui ferramentas que ajuda a identificar e corrigir erros no código de forma mais eficiente.

Apesar de não termos programado em *Assembly devido a sua grande complexidade*, o nosso código em C escrito no PCWHD IDE CCS, ao ser compilado é traduzido em *Assembly*, e só apos isso, na montagem que ele é convertido em código máquina, que vai gerar um arquivo final em formato hexadecimal (.hex). Na **figura 19** abaixo, pode se observar o *layout* do IDE do compilador utilizado.

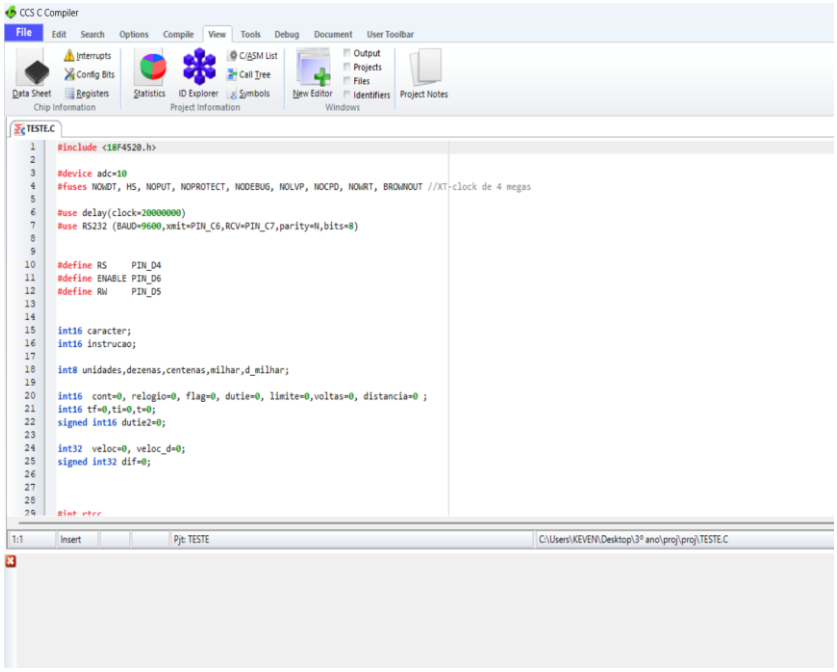


Figura 19- PIC C compiler

4.2 PICkit2 *Programmer*

O *software* PICkit2 *Programmer* e um programa usado para controlar o programador PICkit 2, representado na **fig. 21**, permitindo assim a programação e depuração de microcontroladores PIC e outras memorias compatíveis. Esse software foi desenvolvido pela Microchip e oferece uma interface, ilustrada na **fig. 20,** simples e fácil de usar para gravar código no microcontrolador e verificar o seu funcionamento.

Dentre as funcionalidades oferecidas por este *software*, podemos destacar algumas, como:

1. Gravação de Código no Microcontrolador: O *software* permite gravar o arquivo HEX no microcontrolador;
2. Leitura de Microcontroladores: Além de programar, o PICkit 2 pode ler o conteúdo de um microcontrolador. Isso é útil para verificar qual código ou dados estão armazenados na memória;
3. Controle de Tensão (Vdd): O *software* pode fornecer a tensão de alimentação (Vdd) para o microcontrolador diretamente a partir do programador PICkit 2, o que facilita o teste de pequenos circuitos sem a necessidade de uma fonte de alimentação externa.

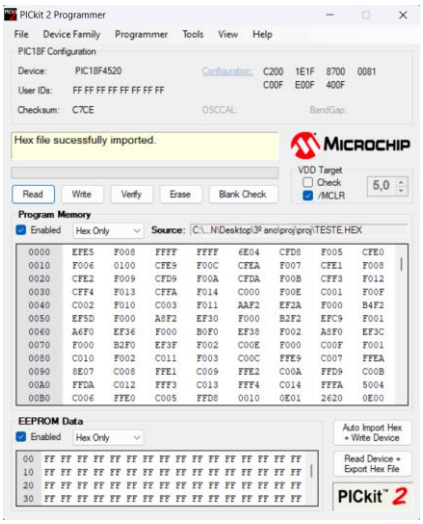


Figura 20 - Interface do PICKit2 Programmer

4.2.1 Programador PICkit 2

Como podemos notar acima, é praticamente impossível falar do PICkit2 *Programmer* (aplicativo de *software*), sem falar do programador PICkit 2 (dispositivo

de *hardware*), pois a relação entre estes é intrínseca, então vamos entender um pouco mais sobre este último.

O PICkit 2 e um programador físico (hardware) que se conecta ao µC e ao computador via USB. Ele lê os comandos do software de programação e grava o *firmware* no microcontrolador. Em suma, podemos afirmar que ele é o canal físico que permite ao PICkit 2 *Programmer* desempenhar todas as suas funções, incluindo as acima citadas.

O PICkit 2 se conecta ao microcontrolador através de um conector ICSP (*In-Circuit Serial Programming*), que geralmente tem 5 ou 6 pinos. O ICSP permite que você programe o microcontrolador sem removê-lo do circuito. O conector geralmente inclui os pinos destacados na **fig. 22**.

Figura 21 - Programador PICKit2

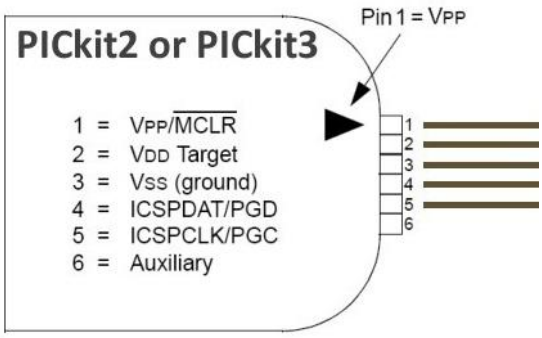


Figura 22 - Conector ICSP

A pinagem do PICkit 2 é a seguinte:

- Pin 1: Vpp (Tensão de programação).
- Pin 2: Vdd (Alimentação do circuito).
- Pin 3: GND (Terra).
- Pin 4: PGC (*Clock* de programação).
- Pin 5: PGD (Dados de programação).
- Pin 6: LVP (opcional, raramente usado).

Podemos constatar que o *software* PICkit2, junto com o dispositivo de *hardware* PICkit2, são indispensáveis e essências para trans formar o *firmware* desenvolvido em formato .hex em um programa executável no microcontrolador PIC.

4.3 Aplicativo BlueSPP

O BlueSPP é um aplicativo para dispositivos móveis que oferece comunicação via *Bluetooth Serial Port Profile* (SPP), um protocolo que permite a comunicação série sem fio entre dispositivos. Esse tipo de comunicação é amplamente utilizado em dispositivos embarcados, módulos Bluetooth, sensores e microcontroladores que precisam se comunicar com um smartphone ou outro dispositivo, sem a necessidade de cabos.

O SPP é um perfil de bluetooth que emula uma conexão série RS-232, permitindo a troca de dados entre dispositivos de forma simples e eficiente, e o BlueSPP usa esse perfil para permitir que dispositivos (*smartphones Android*) se conectem a dispositivos como microcontroladores (modulo bluetooth HC-05).

A interface de interação do aplicativo permite que o utilizador ao se conectar, envie comandos (geralmente em formato de texto ou binário), através do teclado como mostrado na **fig. 23**, e/ou visualize respostas de dispositivos *bluetooth* pelo terminal, **fig. 24**.

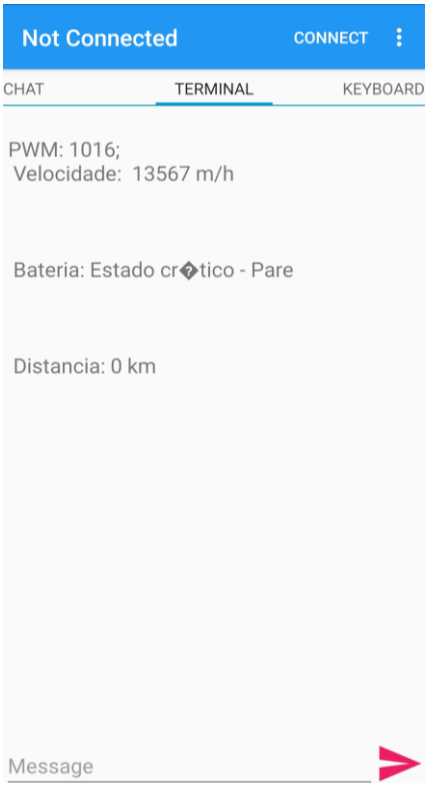


Figura 24 - Terminal da App BlueSPP

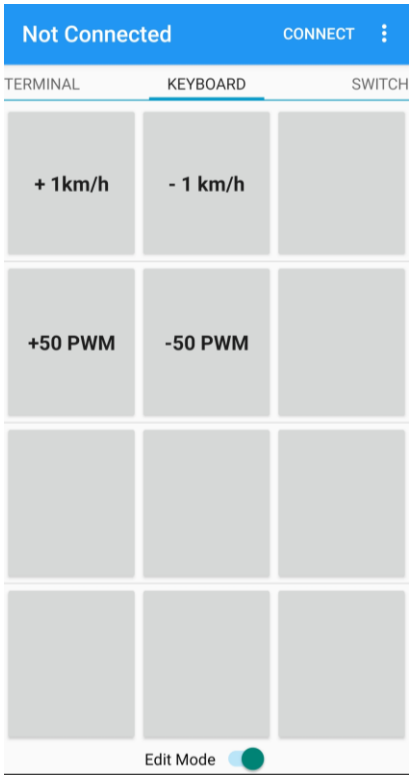


Figura 23 - Teclado do BlueSPP

5. Arquitetura do Sistema

Neste capítulo será apresentado, e explorado a arquitetura e estrutura do sistema por detrás do funcionamento da trotinete elétrica desenvolvida, destacando a organização e interconexão dos principais sistemas que compõem este veículo (protótipo).

A arquitetura do nosso sistema, é composta por diferentes módulos, incluindo o sistema de propulsão (aceleração), o sistema de controlo, monitoramento da velocidade, deteção de travagem, entre outros, que trabalham de forma integrada para garantir a operação eficiente e confiável da trotinete.

O diagrama de blocos representado abaixo, oferece uma visão geral de como os diferentes componentes e subsistemas se conectam e interagem.

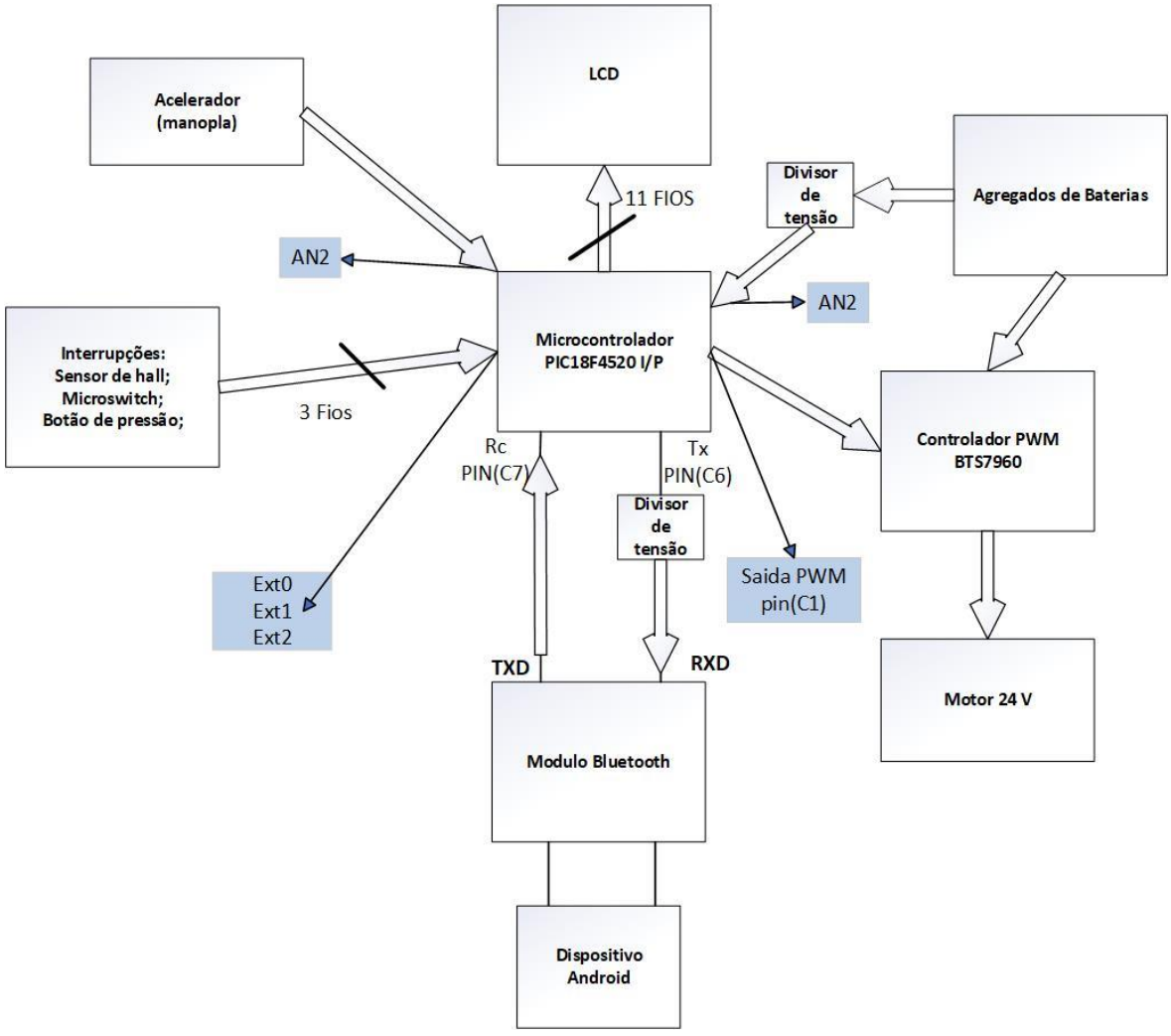


Figura 25 Diagrama de Blocos do Sistema

Com o auxílio da Figura 25, podemos observar que o microcontrolador está no centro de todo o sistema, permitindo a interação entre os diferentes módulos. Ele desempenha diferentes papeis, tais como:

- Receber um sinal um sinal analógico de um acelerador de torção e convertê-lo num sinal PWM;
- Enviar esse sinal PWM para um controlador, que regula a potência fornecida ao motor (proveniente das baterias);
- Recebe sinais de um sensor de Hall, um *microswitch* e um botão de pressão, os quais são utilizados, respetivamente, para determinar a velocidade, detetar o travão, e entrar no modo *Cruise Control;*
- Receber um sinal analógico proveniente de um divisor de tensão relacionado às baterias, que é utilizado para determinar o percentual de carga das mesmas;
- Ele também envia, todas as informações relevantes ao utilizador, tanto para um LCD quanto para um dispositivo movel (telemóvel), por meio de um módulo bluetooth.

5.1 Alimentação do Sistema

No nosso sistema foi implementado um interruptor DPST (*Double Pole, Single Throw*), que ao ser conectado ao terminal do conjunto das baterias, nos proporciona o controlo manual da alimentação de toda a nossa rede elétrica e eletrónica.

Basicamente ele abre ou fecha o circuito elétrico, assim podendo ligar ou desligar o circuito que alimenta o motor e os sistemas eletrónicos que liga as baterias ao restante do sistema.

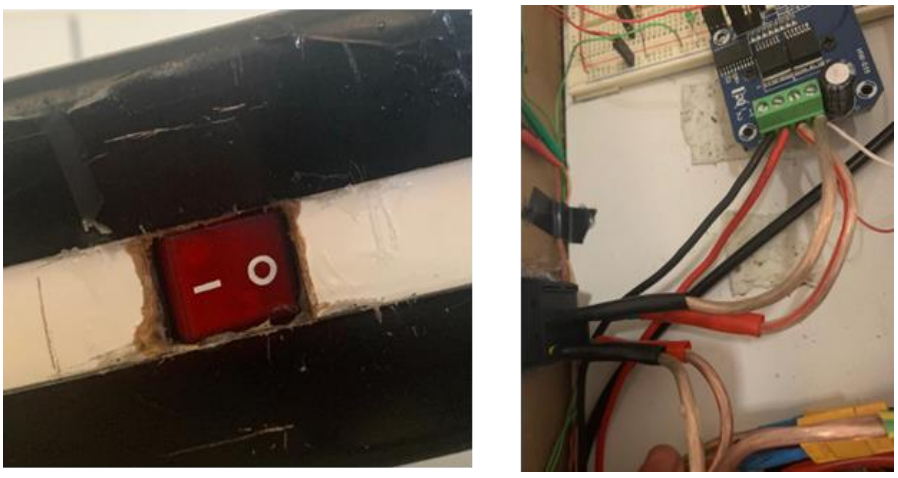


Figura 26 - Interruptor de alimentação do circuito

5.1.1 Fornecimento de energia a PICDEM2 Plus

Para os que já tiveram algum contacto com a placa de demonstração, quando se menciona a ideia de alimentá-la, a tendência regular é pensar numa fonte de alimentação externa (adaptador de energia), ligada ao seu conector de alimentação, que lhe permite ser energizada com uma tensão de 9 a 15 V DC.

Como, nessas condições, alimentar a placa com um cabo de alimentação é uma fantasia, optou-se por energizá-la através do conector de pilha/bateria de 9 V. Onde se utilizou do regulador de tensão, para baixar a tensão disponibilizada pelas baterias, e conectou-se a um terminal de 9 V, que por seu fim veio a ser ligado ao conector de pilha 9 V da placa, alimentando-a. Segue uma ilustração disso, abaixo

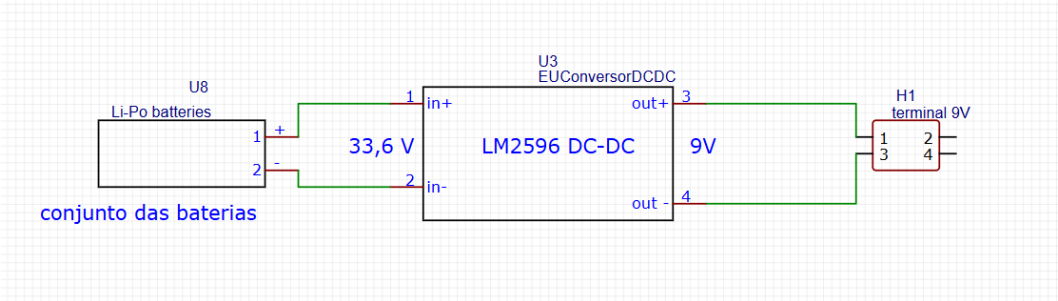


Figura 27 - Esquema de alimentação da Placa

E foi dessa forma que se conseguiu alimentar a placa de demonstração, que seria o "coração" do nosso sistema (não o cérebro, pois o cérebro seria o nosso µC).

Nas figuras podemos observar o terminal e o conector utilizados, bem como o resultado da montagem desse sistema.

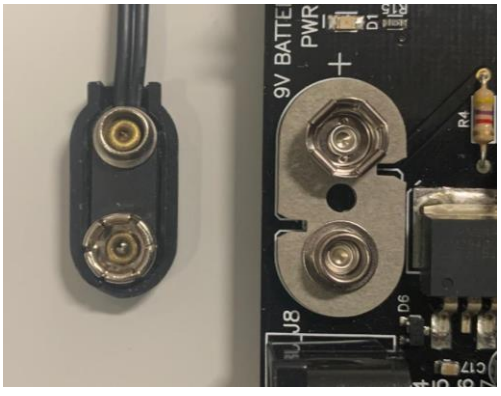


Figura 29 - Terminal e conector de bateria 9V

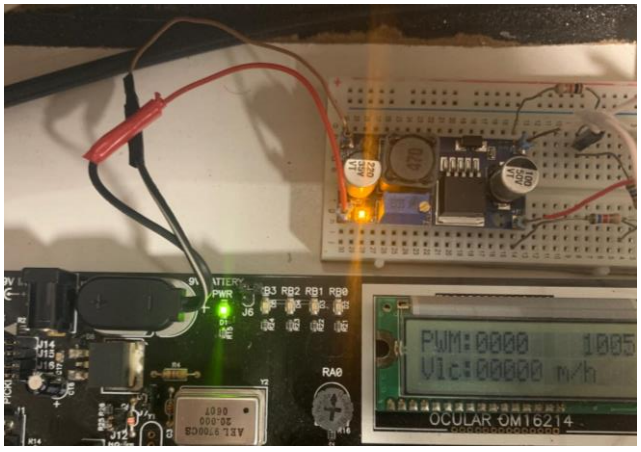


Figura 28 - Sistema de alimentação da placa

5.2 Medição de velocidade

A análise do sistema de medição de velocidade apresenta um desafio interessante. Embora os conceitos fundamentais e a metodologia envolvida na sua implementação, assim como a montagem física e elétrica, sejam relativamente simples e de fácil compreensão, a complexidade aumenta significativamente ao abordar o desenvolvimento do algoritmo de controlo nos microcontroladores PIC.

O desenvolvimento deste algoritmo requer uma abordagem cuidadosa na definição de parâmetros, funções e interrupções, essenciais para o funcionamento eficiente do sistema.

Portanto, embora os aspetos básicos do sistema sejam acessíveis, a verdadeira complexidade reside na integração dos componentes no algoritmo de controlo, que demanda um entendimento mais profundo das interações entre hardware e software.

Devido a esses fatores, a analise desse sistema, será feita em duas secções distintas, abordando cada um dos aspetos.

5.2.1 Conceito e montagem

Relembrando as aulas de física, sabemos que a velocidade é uma grandeza associada ao movimento, que é definida pela razão entre o espaço percorrido e o tempo gasto para tal.

Tendo noção desse princípio, posso afirmar que para determinar a velocidade da trotinete, preciso apenas saber o quanto a trotinete andou num certo intervalo de tempo. Mas como o objetivo é chegar o mais próximo da velocidade instantânea, esse intervalo de tempo tem de ser muito curto.

(5.1) $$V = \frac{\Delta s}{\Delta t}$$

Onde:

V- Velocidade;

Δs- Variação de posição;

Δt- Variação de posição.

Para a montagem de um circuito que possibilitasse essa medição usou-se o sensor de Hall, acoplado no mecanismo de suporte da roda traseira, e se afixou um íman num dos raios da roda, como ilustrado na figura abaixo.

Figura 30 - Sensor de hall e íman

Esse mecanismo gera uma variação de tensão sempre que a roda der uma volta (íman passa pelo sensor), nos informando que ele já andou 1,07 metros, que é o perímetro da roda (calculado no Capítulo 1). Agora que já conseguimos mensurar o seu deslocamento, temos de medir o tempo que ele leva a fazer isso. Essa função será desempenhada pelo µC, através do algoritmo de controlo.

5.2.2 Algoritmo de controlo de velocidade

Sabendo que o essencial no momento é o tempo, nada melhor do que começar falando pela interrupção que permite contar o tempo, o *Timer_0*.

O Timer_0 e um contador de 8 bits que incrementa a cada 0.4µs, sabendo que 8 bits permitem a contagem apenas até 255, temos:

(5.2) $$255 * 0.4\mu s = 104\ \mu s$$

Dessa forma, temos que 104µs é o tempo de *overflow*, do Timer_0. Após isso, criou-se uma interrupção interna, ativada pelo *overflow* do Timer_0, onde dentro dela vai sendo incrementada uma variável de 16 bits (cont).

```
#int_rtcc
void rtcc_interrupt(void)
{
cont=cont+1;
relogio++;
}
```

$$cont \rightarrow int\ 16 \rightarrow 65535$$

(5.3) $$Overflow = 65535 * 104\mu s$$

$$Overflow \cong 6.8\ segundos$$

Figura 31 - Interrupção interna (contador)

Tendo um temporizador tão longo quanto o implementado pela variável 'cont', tornou-se possível medir a velocidade.

A medição da velocidade é feita da seguinte forma:

1. Assim que a roda der uma volta, ela ativa uma interrupção externa (int ext), que de imediato lê e guarda na variável tf o valor de cont;
2. De seguida faz a diferença entre o instante de tempo que se completou essa volta(tf), e o instante de tempo que se completou a volta anterior(ti);
3. Tendo essa variação de tempo, é só dividir o perímetro por ela, e se chega na velocidade.

```
#int_ext
void ext_interrupt()
{
tf= cont;

if(tf>ti){ // se a volta ocorreu toda dentro de um unico ciclo do cont
t= tf - ti; //calcula a variação do tempo entre uma volta e outra
}

else{// se a volta inicioum num ciclo do cont, porem acabou em outro
t=(tf+65535)-ti;
}
veloc=(1.07*3600*10000)/(t*1.04); // calcula a velocidade
ti=tf; //o tempo final se torna inicial para a proxima volta
}
```

Figura 32 - Interrupção que mede a velocidade

Na figura acima está representada a interrupção que mede a velocidade. Pode-se observar que nela constam mais instruções dos que as que eu enumerei como sendo necessárias para fazer a medição. Porém na figura elas se encontram explicadas, e têm mais a função de evitar erros na medição.

OBS: O cálculo da velocidade tem uma configuração estranha, pois foi modificada para apresentar o resultado em m/h, pois:

(5.4) $$velocidade(m/h) = \frac{1.07}{T_s/3600} \quad \text{porém,} \quad T_S = t * 104 * 10^{-6}$$

$$\Leftrightarrow \quad velocidade = \frac{1.07*3600}{t * 1.04 * 10^{-4}} \quad \Leftrightarrow \quad velocidade = \frac{1.07*3600*10^4}{t * 1.04 *}$$

Onde:

T_S – tempo em segundos;

t – tempo medido no sistema (1t = 104µs).

5.3 Esquema Elétrico

Nesta secção, será apresenta o esquema elétrico do sistema desenvolvido para o projeto, onde foi feita toda a representação visual essencial para ilustrar as conexões entre os diversos componentes eletrónicos utilizado, incluindo o microcontrolador, driver de motor, sensores e fontes de alimentação. Aproveito também para avisar que, exceto pelo LCD, que se achou necessário devido ao seu papel, no esquema elétrico não constam as ligações e componentes presentes na placa PICDEM2 Plus, mas esta já possui o seu próprio esquema elétrico.

Esta representação é importante para compreender como cada parte do sistema interage, e não apenas facilita a análise e o diagnóstico de problemas potenciais, mas também serve como uma ferramenta valiosa para futuras expansões e melhorias do sistema.

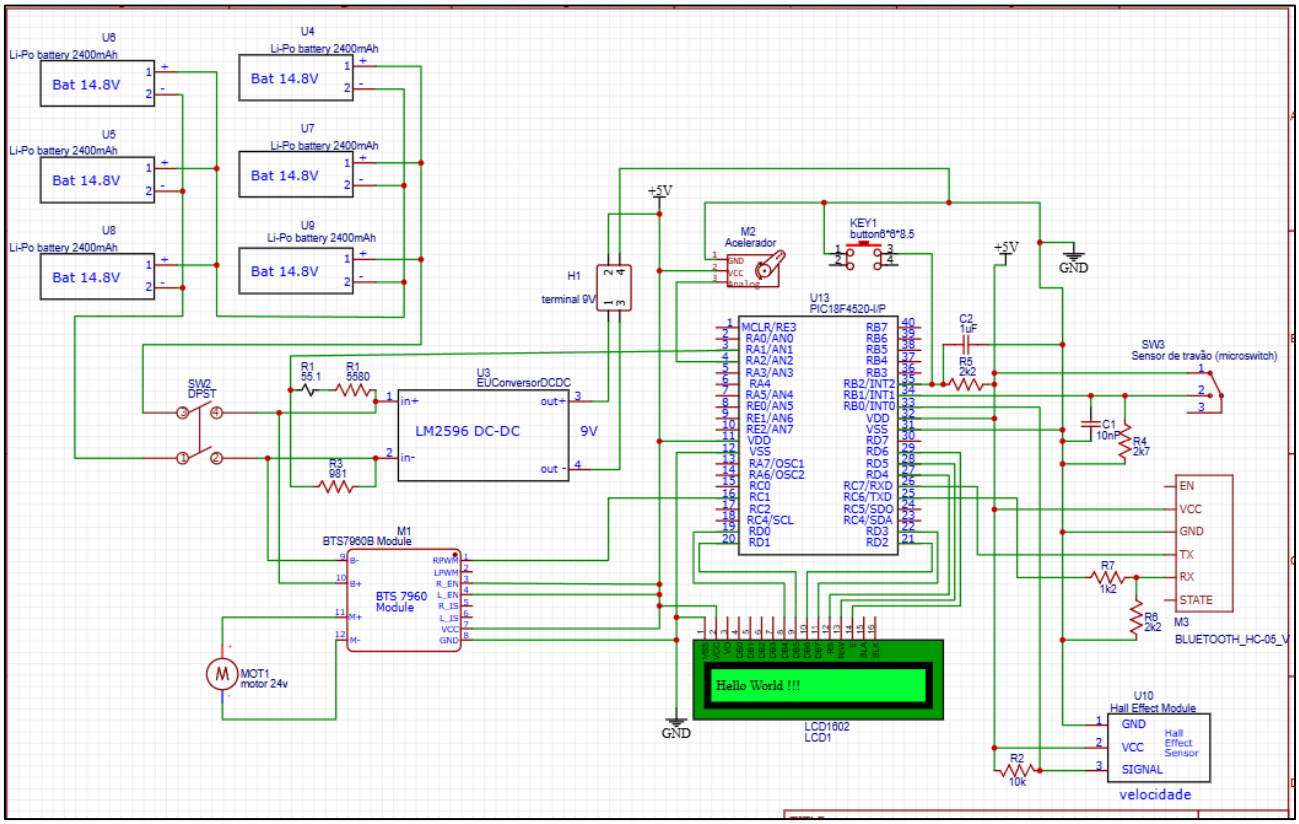


Figura 33 - Esquema Elétrico

6. Operação e Desempenho Tecnológico

Neste capítulo, será apresentada uma análise detalhada das funcionalidades operacionais e do desempenho tecnológico do protótipo desenvolvido neste projeto. Aqui, exploraremos como o sistema funciona na prática, destacando os seus modos de operação, bem como as tecnologias integradas que visam aprimorar a experiência do utilizador.

Dentre os tópicos abordados, está o funcionamento do sistema de *cruise control*, que oferece maior conveniência ao permitir que o utilizador mantenha uma velocidade constante sem precisar acionar continuamente o acelerador. Também será discutido o processo de monitorização da carga das baterias, um componente fundamental para garantir que o utilizador tenha acesso contínuo às informações sobre a autonomia e o estado de carga, proporcionando um uso mais eficiente do veículo.

6.1 Funcionamento do sistema

O Modo padrão de funcionamento do protótipo, foi implementado na sua totalidade, e é no seu todo semelhante ao funcionamento das trotinetes elétricas convencionais.

Basicamente nesse modo, o utilizador dá impulsão ao motor, através da manopla de aceleração, onde esta gera um sinal analógico, que ao chegar na placa é convertida em um sinal PWM, e enviada para o controlador, onde este sim alimenta o motor, com uma corrente proporcional ao *duty cycle* do PWM que nele é injetado.

Na figura abaixo esta ilustrado o algoritmo de funcionamento da trotinete no seu modo convencional.

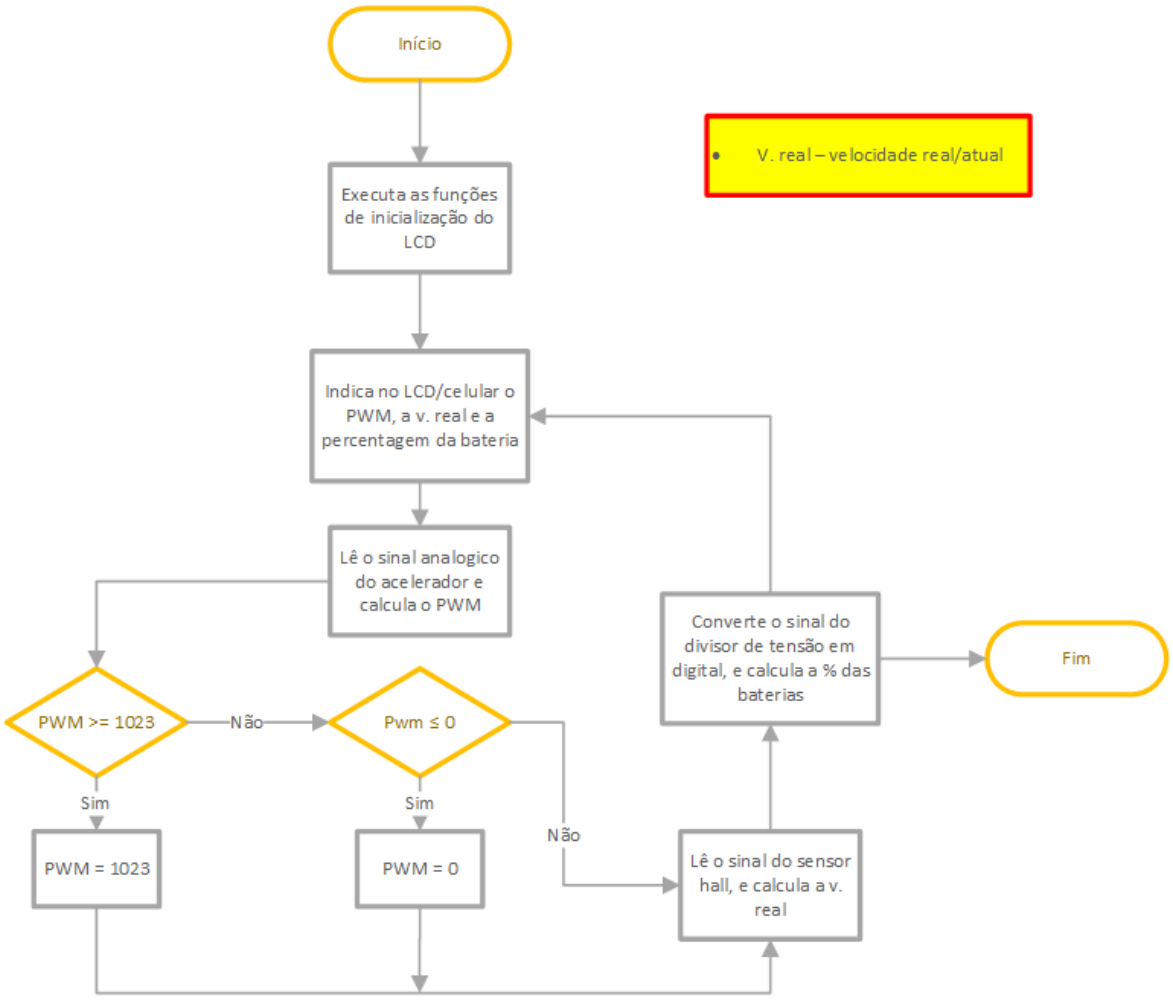


Figura 34 - Fluxograma do programa principal

6.2 Desempenho do Motor e Acelerador

Como já foi dito anteriormente, o motor utilizado e um motor de 24 V, 250W e 3000rpm. Porém ele possui uma relação de redução de 9,78:1, o que significa que para cada 9,78 voltas que o motor realiza, a roda dá uma volta. Portanto a rotação da roda será menor do que a do motor.

$$(6.1) \quad rpm\ da\ roda = \frac{rpm\ do\ motor}{relação\ de\ redução} \quad \Leftrightarrow \quad rpm\ da\ roda = \frac{3000}{9{,}78} \cong 306{,}75\ rpm$$

Ao fazer isso, podemos calcular qual a velocidade máxima que a trotinete pode atingir, com base na rotação do motor. Primeiro convertemos as rotações por minuto (rpm) para rotações por segundos (rps), dividindo os 306,75 rpm, por 60 segundos, obtendo assim 5,11 rps (rotações por segundo)

Feito tudo isso, a velocidade máxima será dada pelo produto das rotações por segundo pelo perímetro da roda.

(6.2) $$Veloc\ (m/s) = rps\ da\ roda * \ perímetro$$

$$Veloc\ (m/s) = 5.11 * 1.07$$

$$Veloc\ (m/s) \cong 5.47\ m/s\ =\ 19.7km/h$$

Devido a relação de redução da engrenagem do motor ser um pouco elevada, a velocidade máxima é limitada, porém a relação de força na engrenagem (torque) e maior, dando mais força para superas obstáculo e atrito.

Fez-se um teste, injetando um sinal PWM constante no sistema, por um certo período, depois aumentando numa escala de 20% em 20%, para ver como o comportamento da velocidade. Na figura seguinte, o eixo da direita é a escala para os valores do PWM, e os da direita são da velocidade em m/h.

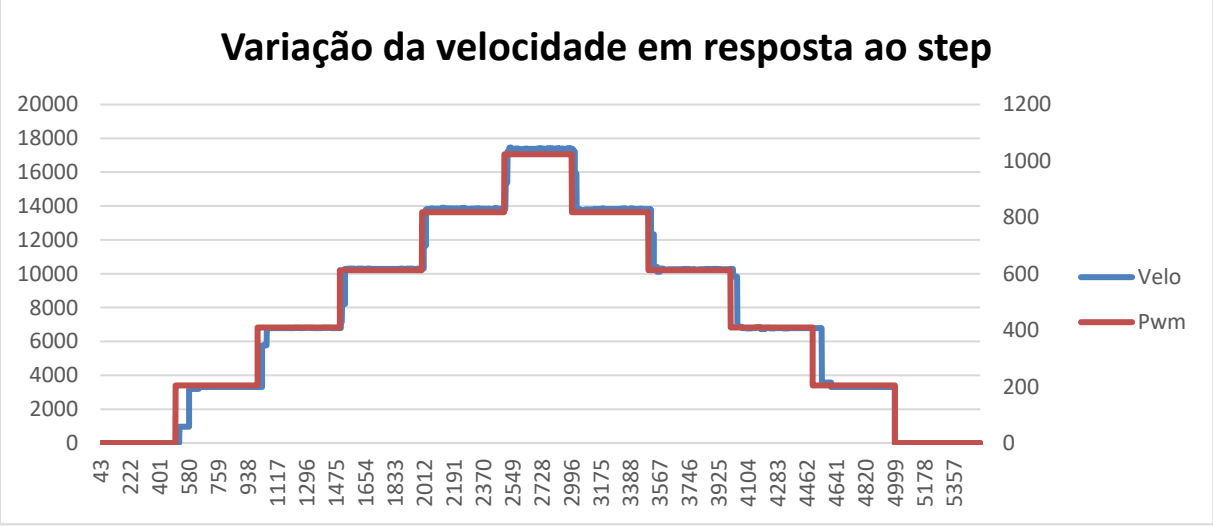


Figura 35 - Velocidade em resposta ao "step" de PWM

Objetivo principal era usar o gráfico para fazer o estudo do sistema determinando o seu ganho, mas o modo no qual estamos medindo a velocidade não nos permite observar a sua progressão gradual, pois os dados da velocidade só são atualizados quando a roda completa um giro, o que ocorre a cada 20 m/s, quando o sistema está operando na velocidade máxima.

Em contrapartida, diagrama nos confirmou que realmente o motor limita a velocidade máxima, que mesmo após um bom tempo de uso das baterias, apresenta quando acelerado ao máximo uma velocidade por volta dos 18000 m/h (18km/h), valor perto dos 19 km/h calculados.

Outro parâmetro que os dados recolhidos nessa amostragem nos permitiram definir foi o tempo de resposta do sistema, pois se repararem bem no gráfico, há um espaço de tempo entre a injeção do PWM, e a reação do motor (roda no caso).

No instante de arranque onde o sistema sai do repouso, é maior do que um segundo (1s), mas pecando nos outros instantes de transição, o sistema tem um atraso/tempo de resposta medio de 0.89 segundos.

6.3 Sistema de *Cruise Control*

Desde que se idealizou este projeto, o desenvolvimento de um sistema de *cruise control*, foi uma das principais metas a serem alcançadas. Sendo uma funcionalidade projetada para proporcionar maior conforto e praticidade ao utilizador da trotinete elétrica, esse sistema permite que o veículo mantenha uma velocidade constante automaticamente, sem que seja necessário manter o acelerador "rotacionado". O que é extremamente útil em trajetos mais longos ou em terrenos planos, onde o utilizador pode desfrutar de uma condução mais relaxada, economizando esforço e, em muitos casos, otimizando o consumo de energia.

6.3.1 Implementação

Relativamente semelhante ao sistema de medição de velocidade, nos aspetos relativos a conceito, projeção e montagem elétrica, o sistema é bem simples e de fácil compreensão, a complexidade reside, na sua maior parte, na programação do algoritmo, processamento de informações e segurança.

Como mecanismo/interface de ativação do *cruise control*, foi utilizado um botão de pressão, que ao ser pressionado, ativa uma interrupção (ext_2), que interrompe o algoritmo do modo de funcionamento padrão, pulando para a sub-rotina equivalente a esse sistema.

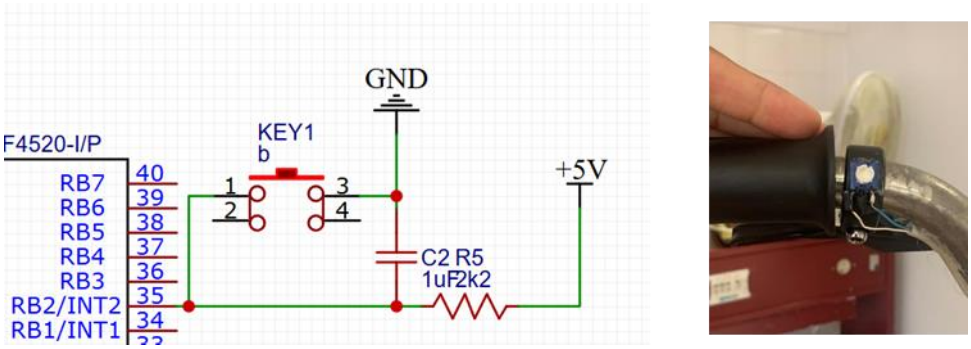


Figura 36 - Esquema e Botão de ativação do *cruise control*

Foi aplicado um filtro passa baixo (RC), pois o sinal chegava a placa com perturbações, o que era um problema, então projetei um filtro passa baixo de modo a proteger o sinal de interferências eletrónicas ou eletromagnéticas do motor ou de outras componentes.

(6.3) $$f_c = \frac{1}{2*\pi*rc} \quad \Leftrightarrow \quad f_c = \frac{1}{2*\pi*2200*10^{-6}} \quad \Leftrightarrow \quad f_c \approx 75\ Hz$$

Vale a pena também realçar que a afirmação de que ao pressionar o botão se ativa a interrupção, do *cruise control*, não quer dizer que o *cruise control* em si será ativado, pois existe uma condição que impede a ativação deste sistema se a trotinete não estiver em movimento.

- **"if (veloc != 0) {Ativação do *cruise control*}"** – essa simples linha de código condiciona a ativação do *cruise control,* a ela velocidade ser diferente de zero.

6.3.2 Operação do Sistema

O sistema foi projetado para monitorar a velocidade da trotinete e ajustar automaticamente a entrega de potência ao motor para manter essa velocidade constante.

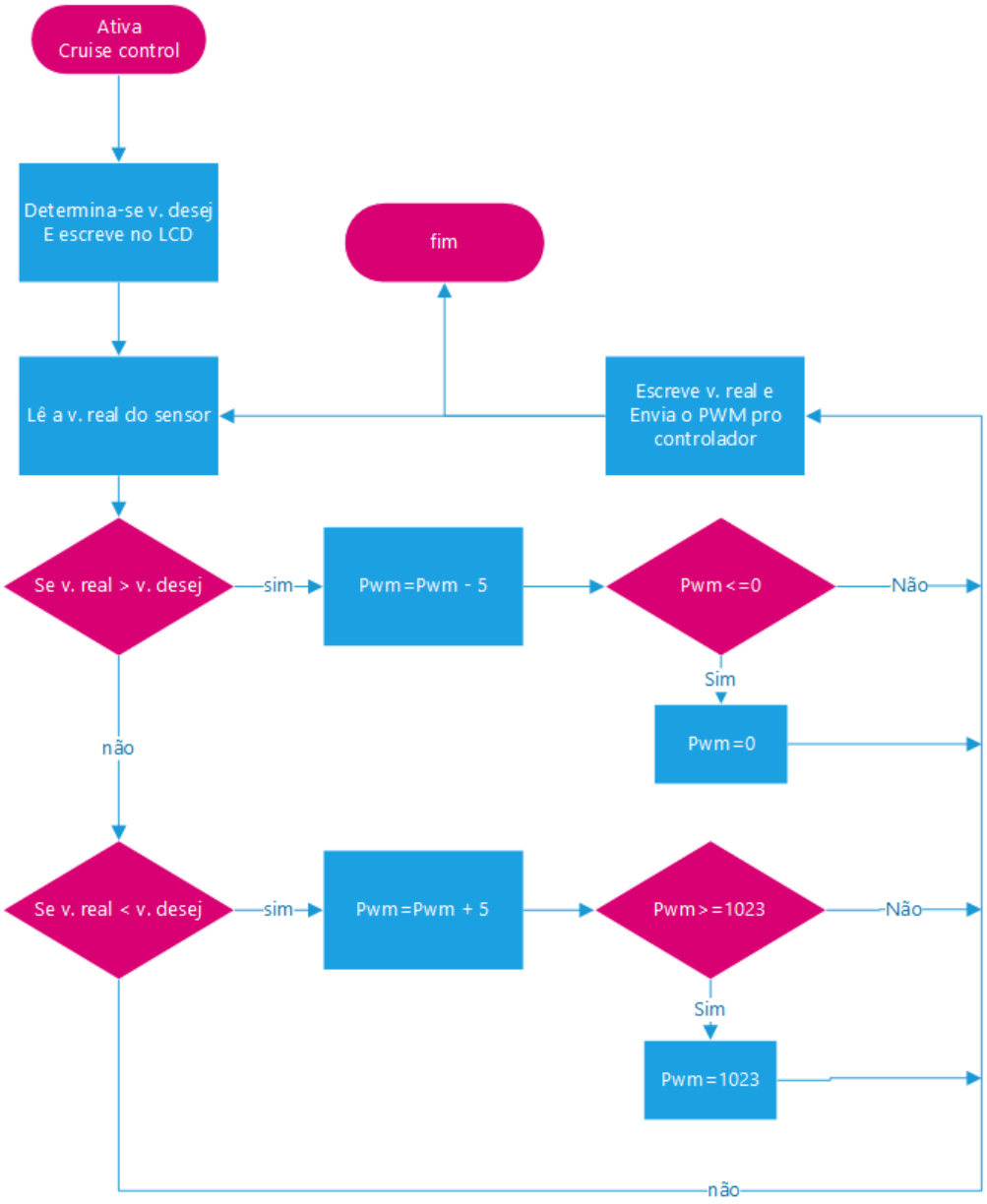


Figura 37 - Fluxograma do *cruise control*

O fluxograma acima, ilustra o modo de funcionamento deste sistema, que só foi possível, devido as componentes que o integram, como:

- O sensor de Hall - que em colaboração com o µC, se torna um sensor de velocidade;
- O botão de pressão que permite ao utilizador entrar no *cruise control*;
- O µC que além do seu papel a medir a velocidade, ajusta o PWM, por consequência a potência, de modo a manter a velocidade desejada pelo utilizador.

6.3.3 Desempenho do sistema

Com o sistema funcionando veio a necessidade de analisar o comportamento do sistema em diferentes situações. Para isso realizaram-se alguns testes, submetendo o sistema de *cruise control* a diferentes ambientes.

O primeiro teste foi realizado com o intuito de compreender como, que o sistema se comportava com a variação das características do ambiente, adaptando-se para manter a velocidade desejada pelo utilizador. Na amostragem ilustrada pelo diagrama abaixo, o fator de incrementação do sistema era de uma unidade (aumentava/diminuía o PWM de 1 em 1), e a amostragem era feita a cada meio segundo, dessa forma permitindo uma melhor visualização do seu comportamento.

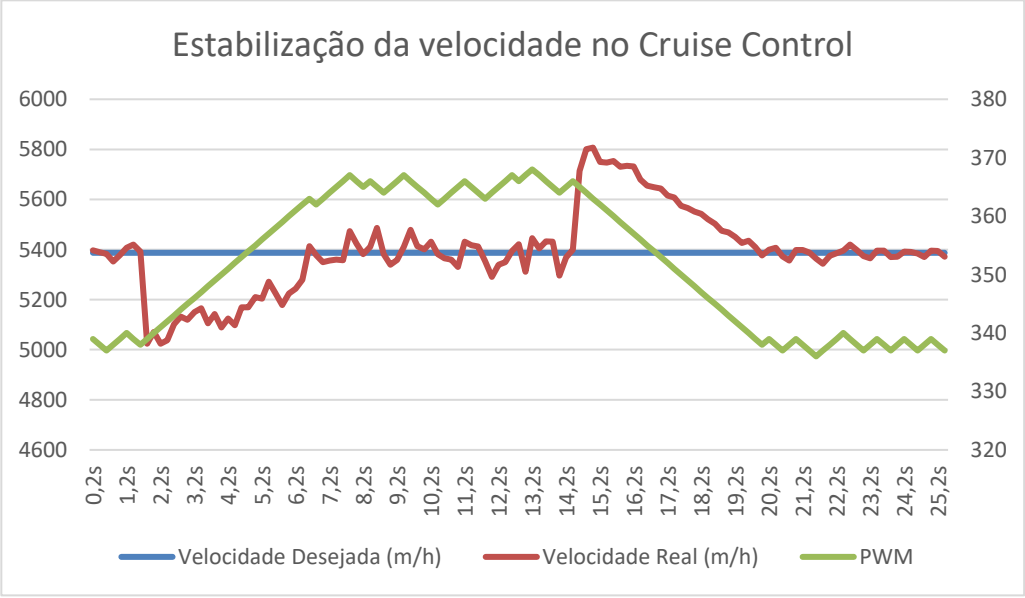


Figura 38 - Velocidade no *cruise control*

O diagrama acima, ilustra uma situação onde a trotinete já estava no modo *cruise control*, e por volta do instante 1,6s se deparou com um obstáculo (uma subida), que fez a velocidade real diminuir, e o sistema ao identificar isso, aumenta o PWM, aumentando a potencia dada ao motor. O sistema demora cerca de cinco segundos, ou 10 ciclos do sistema, para conseguir estabilizar o sistema, voltando a funcionar numa gama de velocidades a volta da velocidade desejada.

Podemos também concluir, que apesar do ruido causado pela irregularidade do ambiente, o aumento da velocidade é proporcional ao aumento do PWM. Isso é evidenciado de uma forma melhor, no instante 15s, quando é removido o obstáculo (acaba a subida), e a velocidade real tem um pico, pois o motor estava a aplicar uma potência superior para poder compensar o sistema de forças, é então que o PWM começa a descer, e junto consigo a velocidade real, estabilizando após 10 ciclos do sistema (5 segundos).

Através de inúmeros testes deste género, podemos fazer algumas constatações, como:

- O tempo que o sistema demora para se restabelecer voltar a entregar a velocidade desejada após a mudança das condições do ambiente, está diretamente relacionada com o fator de incrementação e o tempo do ciclo de atualização do sistema;
- Quanto maior for o fator de incrementação, menor será o tempo que o sistema levará para se estabilizar, porem maior será a oscilação da velocidade em ambientes adversos
- Fatores de incrementação muito baixos, oferecem maior estabilidade da velocidade, no entanto prolongam o tempo estabilização, principalmente em subidas muito inclinadas;
- Quanto ao ciclo do sistema, quanto menor ele for, mais ágil será o seu sistema, porém o sistema possui um limite de tempo por ciclo, o qual ele necessita para realizar todas as suas tarefas.

Em suma para devido a esses fatores, decidiu-se implementar no protótipo, um fator de incrementação de 5 unidades, que apresenta uma oscilação de $\pm 5\%$ em ambientes adversos, e o ciclo do programa é de 250ms, pois alguns parâmetros, como a velocidade, só se atualizam numa frequência acima de 200ms.

Há situações onde o obstáculo exerce uma força de atrito (contra o sistema), constante e de uma magnitude que, mesmo após dar toda a sua potencia o sistema não conseguem voltar a velocidade desejada, que ele se encontrava, então o sistema fica aplicando um sinal PWM, com um *duty cycle* contante de 1023 (máximo).

Figura 39 - Diagrama de resistência constante

6.4 Monitorização da carga das baterias

As baterias são um dos elementos mais importantes em sistemas de mobilidade elétrica, como o caso de uma trotinete, já que ela determina a autonomia e o tempo de uso do veículo. A monitorização da carga das baterias não só informa o nível de energia restante, como, mas também previne sobrecarga e descarga excessiva, que são fatores que podem danificar as baterias.

Para montar um sistema que nos permitisse visualizar a carga das baterias, tivemos de as estudar a fundo, compreendendo as suas especificações, que estão enunciadas no capítulo 3.3, e compreender como varia a carga de cada bateria, e em que parâmetros podemos evidenciar essa variação.

6.4.1 Variação da carga das baterias

A medida que uma bateria vai se descarregando, a tensão fornecida por ela, vai diminuindo junto com a carga.

Uma bateria de LiPo 4S, que é o tipo das presentes no protótipo, e composta por quatro células de lítio ligadas em série. Quando carregada totalmente, cada célula pode chegar a 4,2 V, o que dá uma tensão total de 16,8 V. Então 16,8 V será a nossa tensão máxima (100%).

E de acordo com as especificações, descarregada cada célula não pode ficar abaixo de 3,2 V, dando uma tensão total de 12,8 V. que seria o nosso mínimo(crítico).

Com relação a percentuais de cargas intermediarias (ex:50%, 30%, 70%), não havia especificações no *datasheet*. Porém as pesquisas dizem, que nesse aspeto as baterias LiPo em geral têm comportamento análogo.

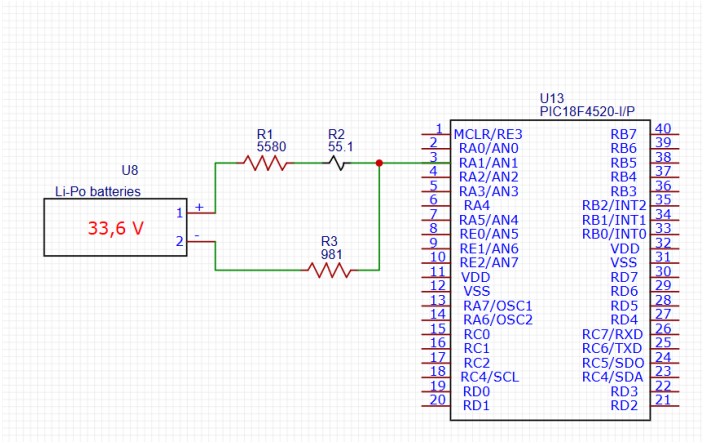
6.4.2 Esquema e sistema de monitorização

Para podermos monitorizar a bateria, definiram – se níveis de carga:

- 100%;
- 60%
- 30%
- *Low*;
- Crítico;

Escolheu-se esses níveis pois as baterias usadas na trotinete, possuem LEDs incorporados, que informam o nível de carga, e esses são os níveis presentes. Escolhendo os mesmos níveis, conseguimos fazer o sistema teórico, baseado nas especificações e dados de estudo, e posteriormente comparar com "real" ilustrada pelos LEDs.

Para monitorizar as baterias é necessário, ler e processar a tensão que elas entregam, mas como a tensão máxima o microcontrolador converte nos seus portos ADC é de 5 V, teremos de aplicar um divisor de tensão, onde a tensão máxima do conjunto, 33,6V, caia para 5V.



$$6.4) \quad V_{A1} = \frac{981}{981+(5580+55.1} * T_{Bateria}$$

$$\Leftrightarrow \quad V_{A1} = \frac{981}{981+(5580+55.1} * 33.6V$$

$$\Leftrightarrow \quad V_{A1} = 5V$$

Figura 40 - Esquema de monitorização da bateria

Na tabela abaixo faz-se a relação teórica entre a tensão e a percentagem de carga:

Tabela 4 - *Relação teórica tensão - percentagem de carga*

Bateria (%)	Tensão p/ célula (V)	Tensão p/ bateria (V)	Tensão do conjunto (V)	Divisor de Tensão (V)	Digital Hex
100%	4,2	16,8	33,6	5	1023
60%	3,9	15,6	31,2	4,63	948
30%	3,7	14,8	29,6	4,38	896
low	3,5	14	28	4,15	849
crítico	3,2	12,8	25,6	3,8	777

Ao aplicar esses intervalos no algoritmo de monitorização desenvolvido através µC PIC18F4520, na placa PICDEM2 Plus, observamos que o sistema estava operacional, mas os status da bateria visualizado no telemóvel, que era gerado pelo algoritmo, não coincidia com os mostrados pelos LEDs da bateria. A tabela de relação entre a tensão e a percentagem de carga foi refeita, agora baseando-se em dados empíricos, tendo algumas discrepâncias com a teórica, como podem observar.

Tabela 5 - *Relação empírica tensão - percentagem de carga*

Bateria (%)	Tensão p/ célula (V)	Tensão p/ bateria (V)	Tensão do conjunto (V)	Divisor de Tensão (V)	Digital Hex
100%	4,2	16,79	33,57	5	1023
60%	4,09	16,38	32,75	4,88	998
30%	3,94	15,77	31,54	4,71	963
low	3,65	14,59	29,17	4,36	893
crítico	3,3	13,2	26,4	3,92	802

O algoritmo usado para monitorizar a carga das baterias, basicamente consiste em ler e converter para digital o valor gerado pelo divisor de tensão, e submetê-lo a uma análise determinando a que intervalo pertence, e baseado nisso fazer um *"print"*, específico desse *status*, acompanhado de outras informações necessárias.

```
if( (bateria <= 1023)  && (bateria > 998)){
status=100;
printf("PWM: %ld;\n  Velocidade:  %ld m/h \n\n\n\n\r Bateria: %ld %% \n\n\n\n\r Distancia: %ld km\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\r",dutie,veloc,status,distancia);
}
if( (bateria <= 998)  && (bateria > 963)){
status=60;
printf("PWM: %ld;\n  Velocidade:  %ld m/h \n\n\n\n\r Bateria: %ld %% \n\n\n\n\r Distancia: %ld km\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\r",dutie,veloc,status,distancia);
}
if( (bateria <= 963)  && (bateria > 893)){
status=30;
printf("PWM: %ld;\n  Velocidade:  %ld m/h \n\n\n\n\r Bateria: %ld %% \n\n\n\n\r Distancia: %ld km\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\r",dutie,veloc,status,distancia);
}
if( (bateria <= 893)  && (bateria > 802)){
printf("PWM: %ld;\n  Velocidade:  %ld m/h \n\n\n\n\r Bateria: Low - Recarregar\n\n\n\n\r Distancia: %ld km\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\r",dutie,veloc,distancia);
}
if(bateria <= 802){
printf("PWM: %ld;\n  Velocidade:  %ld m/h \n\n\n\n\r Bateria: Estado crítico - Pare\n\n\n\n\r Distancia: %ld km\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\r",dutie,veloc,distancia);
}
```

Figura 41 - Algoritmo de determinação do status

6.5 Interface de visualização

Para este projeto, a presença de uma interface de visualização mostrou-se indispensável, pois foi essencial durante o desenvolvimento de todo o sistema. Ela permitiu a visualização tanto dos erros quanto dos avanços em cada etapa do desenvolvimento, além de desempenhar um papel fundamental no uso e na manutenção do protótipo como um produto final.

Inicialmente, cogitou-se a implementação de um LCD para a visualização de informações do veículo. No entanto, essa opção já havia sido abordada há algum tempo, e não mostrou ser uma solução estável e eficaz, devido a distância que o *display* precisaria de estar da placa e à quantidade de cabos necessários para a conexão. Além desses fatores, há também o facto de terem sido implementadas algumas funcionalidades voltadas a eficiência, praticidade e segurança do protótipo, o que aumentou a quantidade de informações a serem transmitidas. Portanto, o LCD, por ter espaço limitado não seria uma opção viável.

Diante desses fatores, decidiu-se pela utilização de um módulo Bluetooth como alternativa para a transmissão dos dados, onde esta envia os dados para um dispositivo movel (telemóvel). Essa solução permitiu superar as limitações identificadas com o uso do *display LCD*, eliminando a necessidade de extensos cabos e a dependência de uma conexão física direta com a placa. Além disso, o uso do módulo Bluetooth proporcionou maior flexibilidade, permitindo que todas as informações do veículo fossem transmitidas de forma eficiente e em tempo real para um telemóvel, garantindo uma interface maior, mais prática e acessível para o monitoramento do protótipo.

Para se conectar com o modulo, é necessário instalar uma aplicação chamada BlueSPP, disponíveis apenas para dispositivos Android.

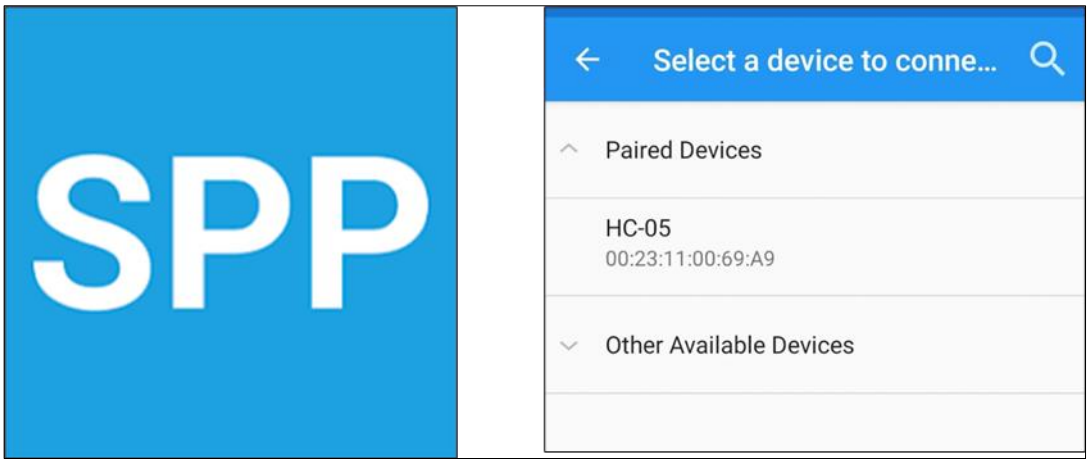


Figura 42 - Logo e janela de conexão do BlueSPP

Os parâmetros em estudo, como a velocidade real, a velocidade desejada, a percentagem da bateria e a distância percorrida, podem ser visualizadas na aba "Terminal" na aplicação.

O *layout* de exibição dos parâmetros no terminal da aplicação é adaptável, onde este varia de acordo com o modo de funcionamento em que a trotinete se encontra. A interface foi projetada para ajustar automaticamente as informações apresentadas, de

modo a refletir as condições e parâmetros específicos de cada modo operacional (convencional ou *Cruise control).*

Nas duas figuras abaixo estão representados os layouts correspondentes aos dois modos de funcionamento, onde se pode lhes comparar, e evidenciar as diferenças entre elas.

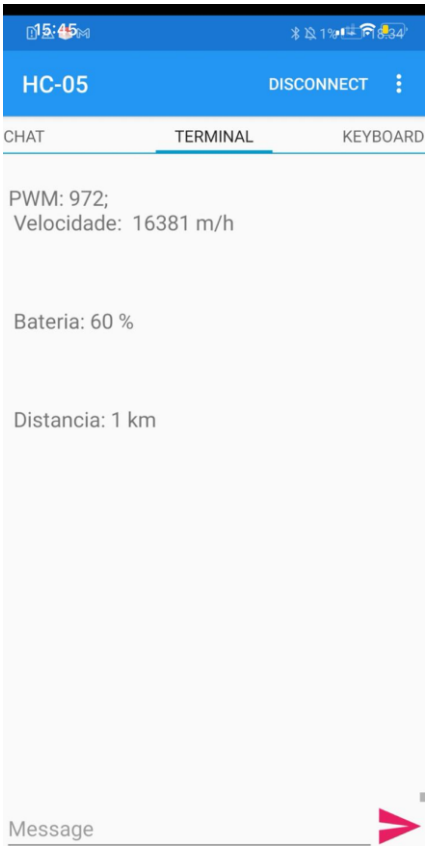


Figura 44 - Layout Modo Convencional

Figura 43 - Layout *Cruise Control*

Podemos observar que no modo convencional, as informações exibidas são o *duty cycle* do sinal PWM, a velocidade, a percentagem das baterias e a distância percorrida até o momento. Porém quando a trotinete entra no modo *cruise control*, esse layout muda um pouco, onde se indica que a trotinete se encontra nesse modo, e os parâmetros exibidos são o *duty cycle*, as velocidades (real e desejada), a percentagem das baterias e a distância percorrida.

Outra secção da aplicação BlueSPP que foi trabalhada numa outra ocasião foi o teclado (*Keyboard*), onde esta era usada para enviar instruções para a placa. Numa primeira fase, onde a trotinete se encontrava suspenso em cima da mesa no laboratório, esta ideia parecia viável, porém ao colocá-la no chão para um teste real, esta funcionalidade não se mostrou pratica nem segura, pois controlar a velocidade

de uma trotinete através de um botão no celular, enquanto esta se encontra a 14/15 km/h, não é tão fácil.

Devido a esses fatores, desistiu-se da ideia de usar um *smartphone* (telemóvel) para interação com a placa, passando este a ser usado apenas para visualização. Contudo segue abaixo uma imagem da secção "*Keyboard",* onde foram criados alguns botões, com o intuito de enviarem dados para a placa, porém acabando por não serem utilizados

Figura 45 - Teclado do BlueSPP

.

7. Conclusões

Este projeto tinha como âmbito e objetivos principais, o desenvolvimento de uma trotinete elétrica estável e funcional, o que se conseguiu alcançar, apresentando um protótipo simples, mas desenvolvido com técnicas e sistemas que implementando um conjunto de funcionalidades que combinam eficiência, praticidade e segurança.

Um dos intuitos também, era provar que com os materiais necessários, dedicação e um determinado conhecimento em eletrónica era possível, pegar numa estrutura mecânica montada a partir de peças reutilizadas de bicicletas, e transformá-la numa trotinete elétrica, com um bom desempenho, de modo a encorajar o desenvolvimento de projetos unam o avanço tecnológico com a sustentabilidade.

O protótipo construído apresentou um desempenho aceitável, conseguindo atingir velocidades na casa dos 18 km/h, sendo um pouco abaixo do que alcançado pelos modelos comerciais, muito devido ao motor, que apresenta uma relação de que limita ligeiramente a capacidade do sistema. Porém, de modo surpreendente, o protótipo apresentou uma autonomia razoavelmente alta, percorrendo cerca de 24 km, até as baterias, chegarem no estado *LOW,* podendo ainda percorrer mais alguns km.

O *cruise control,* desenvolvido mostrou-se totalmente funcional e eficaz, podendo ter como única desvantagem, o facto de só possuir um sensor de travagem, e caso este falhe, se perca a opção de sair do *cruise control*, o que pode até causar um acidente. Sendo este um dos pontos de aprimoramento do projeto.

O sistema de monitorização da carga das baterias, foi um sucesso, permitindo ao utilizador saber a carga das baterias através do telemóvel, sem precisar abrir o quadro para verificar os LEDs das baterias.

Falando do telemóvel, a comunicação entre o dispositivo movel e a placa só foi desenvolvida em um sentido, servindo só como interface de observação. Durante a elaboração do projeto, chegou-se a realizar interações do telemóvel para a placa, controlando o PWM e outras coisas, tanto que a fig. 23, do teclado do BlueSPP, apresenta alguns botões. O problema foi que ao colocar este, não me pareceu viável, estar a mexer no celular enquanto te equilibras em cima de uma trotinete a mais de 15 km/h.

Outra funcionalidade que pode ser desenvolvida, é um sistema de carga para as baterias, permitindo assim carregar as baterias de forma fácil e eficiente.

8. Referencias Bibliográficas

Na grande parte, serviram com referência bibliográficas, os datasheets das componentes, disponibilizadas pelo orientador, ou encontrados na internet.

Manual de utilização da placa PICDEM2 PLUS:

https://ww1.microchip.com/downloads/aemDocuments/documents/OTH/ProductDocuments/UserGuides/40001584C.pdf

Datasheet das baterias:

https://www.hacker-motor.com/daten/anleitungen/akkus/Anleitung_Hacker_LiXx_2021-25112020.pdf

Informações do módulo regulador de tensão:

https://www.makerhero.com/produto/regulador-de-tensao-lm2596-conversor-dc-dc-step-down/

Legislações das trotinetes elétricas:

https://trotinetesportugal.com/legislacao/

Vídeo aula das baterias LiPo:

https://www.youtube.com/watch?v=pf7gG8ENQqE

9. Anexos

9.1 Código do Programa:

```
#include <18F4520.h>
#device adc=10
#fuses NOWDT, HS, NOPUT, NOPROTECT, NODEBUG, NOLVP, NOCPD, NOWRT, BROWNOUT //XT-clock de 4 megas
#use delay(clock=20000000)
#use RS232 (BAUD=9600,xmit=PIN_C6,RCV=PIN_C7,parity=N,bits=8)
#define RS     PIN_D4
#define ENABLE PIN_D6
#define RW     PIN_D5

int16 caracter;
int16 instrucao;
int8 unidades,dezenas,centenas,milhar,d_milhar;
int16 cont=0, relogio=0, flag=0, dutie=0, limite=0, voltas=0, distancia=0;
int16 tf=0, ti=0, t=0;
signed int16 dutie2=0;
int32 veloc=0, veloc_d=0;
signed int32 dif=0;

#int_rtcc
void rtcc_interrupt(void)
{
//output_toggle(Pin_C0);
cont=cont+1;
relogio++;
}
```

```
#int_ext
void ext_interrupt()
{
tf= cont;
if(tf>ti){
t= tf - ti; //calcula a variação do tempo entre uma volta e outra
}
else{
t=(tf+65535) -ti;
}
veloc=(1.07*3600*10000)/(t*1.04);
//!ts = t * 104;
ti=tf;
limite=0;
voltas++;
if (voltas >= 935){ // 1000 m (1km) a dividir por 1.07 (perimetro da roda) dá 935 voltas
distancia++; //adiciona 1km a distancia
voltas=0;
}
}
#int_ext1
void ext1_isr(void){
flag=0;
output_toggle(pin_c4);
}
#int_ext2
void ext2_isr(void){
output_toggle(pin_c4);
if (veloc!=0){
flag=1;
dutie2=dutie;
```

```
veloc_d=veloc;}
}
void activa()
{
 output_high(ENABLE);
 delay_ms(10);
 output_low(ENABLE);
}
void transfere_carac(int8 carac)
{
 output_high(RS);
 output_bit(PIN_D0,(carac & 0x10));
 output_bit(PIN_D1,(carac & 0x20));
 output_bit(PIN_D2,(carac & 0x40));
 output_bit(PIN_D3,(carac & 0x80));
 //output_d(carac);
 activa();
 output_bit(PIN_D0,(carac & 0x01));
 output_bit(PIN_D1,(carac & 0x02));
 output_bit(PIN_D2,(carac & 0x04));
 output_bit(PIN_D3,(carac & 0x08));
 //output_d(carac);
 activa();
}
void transfere_inst(int8 inst)
{
 output_low(RS);
 output_bit(PIN_D0,(inst & 0x10));
 output_bit(PIN_D1,(inst & 0x20));
 output_bit(PIN_D2,(inst & 0x40));
 output_bit(PIN_D3,(inst & 0x80));
```

```
 //output_d(carac);
 activa();
 output_bit(PIN_D0,(inst & 0x01));
 output_bit(PIN_D1,(inst & 0x02));
 output_bit(PIN_D2,(inst & 0x04));
 output_bit(PIN_D3,(inst & 0x08));
 //output_d(carac);
 activa();
}
void ini_lcd()
{
 instrucao=56;
 transfere_inst(instrucao);
 instrucao=1;  //limpa display
 transfere_inst(instrucao);
 instrucao=12; // cursor invivivel
 transfere_inst(instrucao);
 instrucao=2;            // faz clear e colocação do cursor na 1º linha
 transfere_inst(instrucao);
}
void pwm (int16 i)
{
   milhar=i/1000;
   caracter =milhar+48;
   transfere_carac(caracter);
   delay_ms(5);
   centenas=(i%1000)/100;
   caracter =centenas+48;
   transfere_carac(caracter);
   delay_ms(5);
   dezenas=((i%1000)%100)/10;
```

```
    caracter =dezenas+48;
    transfere_carac(caracter);
    delay_ms(5);
    unidades=((i%1000)%100)%10;
    caracter =unidades+48;
    transfere_carac(caracter);
    delay_ms(5);
    caracter = 32;
    transfere_carac(caracter);
}
void velocidade (int32 vel)
{
  d_milhar=vel/10000;
  caracter =d_milhar+48;
  transfere_carac(caracter);
  delay_ms(5);
  milhar=(vel%10000)/1000;
  caracter =milhar+48;
  transfere_carac(caracter);
  delay_ms(5);
  centenas=(vel%1000)/100;
  caracter =centenas+48;
  transfere_carac(caracter);
  delay_ms(5);
  dezenas=((vel%1000)%100)/10;
  caracter =dezenas+48;
  transfere_carac(caracter);
  delay_ms(5);
  unidades=((vel%1000)%100)%10/1;
  caracter =unidades+48;
  transfere_carac(caracter);
```

```
 delay_ms(5);
 caracter = 32;
 transfere_carac(caracter);
 caracter = 'm';
 transfere_carac(caracter);
 caracter = '/';
 transfere_carac(caracter);
 caracter = 'h';
 transfere_carac(caracter);
}

void main()
{
  unsigned int16 i=0, bateria=0, status=0;
  setup_adc_ports (AN0_TO_AN3);
  setup_adc(ADC_CLOCK_INTERNAL);
  //setup_ccp1(CCP_PWM); //ccp1 pin c2
  setup_ccp2(CCP_PWM); //ccp2 pin c1
  set_pwm2_duty(0);
  setup_timer_2(T2_DIV_BY_16,255,1); //definir o tempo
  set_tris_d(0x00); //tudo saidas

  setup_timer_0(RTCC_Internal|RTCC_8_BIT);//cria um timer de 8 bit que demora 104 us p/
overflow
  enable_interrupts(int_rtcc);
  enable_interrupts(int_ext);// ativa interrupção no pin B0
  enable_interrupts(int_ext1);// ativa interrupção no pin B1
  enable_interrupts(int_ext2);// ativa interrupção no pin B2
  enable_interrupts(global);
  output_low(RW);
  ini_lcd();
  veloc=0;
```

```
  flag=0;


while (1)
  {
    if (flag == 0){
      transfere_inst(1); // limpa lcd
      instrucao=128;          //escreve na primeira linha
      transfere_inst(instrucao);
      caracter = 'P';
      transfere_carac(caracter);
      caracter = 'W';
      transfere_carac(caracter);
      caracter = 'M';
      transfere_carac(caracter);
      caracter = ':';
      transfere_carac(caracter);
      instrucao=192;          //escreve na segunda linha
      transfere_inst(instrucao);
      caracter = 'V';
      transfere_carac(caracter);
      caracter = 'l';
      transfere_carac(caracter);
      caracter = 'c';
      transfere_carac(caracter);
      caracter = ':';
      transfere_carac(caracter);
      while(flag==0){
      if (relogio >= 2404){ //atualiza os dados a cada 1/4 de segundo; timer de 104us (104us * 2404=
250ms)
        relogio=0;
        limite++;
        instrucao=132; // coloca o cursor na 5ª posição da 1ª linha
```

```
transfere_inst(instrucao);
set_adc_channel(2);// lê a ADC no A2
delay_us(60);
i=read_adc();     //lê da ADC
i=i-182;
dutie=i*1.482;
if (dutie>=1023){
  dutie=1023; }
if (dutie<=0){
 dutie=0;
pwm(dutie);
set_pwm2_duty(dutie);
set_adc_channel(1);// lê a ADC no A1
delay_us(60);
bateria=read_adc();     //lê da ADC

instrucao=140; // coloca o cursor na 5ª posição da 1ª linha
transfere_inst(instrucao);
pwm(bateria);

if((limite>=3)&&(dutie<30)){
  veloc=0;
  }

instrucao=196; // coloca o cursor na 5ª posição da 2ª
transfere_inst(instrucao);
velocidade(veloc);



if( (bateria <= 1023)  && (bateria > 998)){
status=100;
```

```
printf("PWM: %ld;\n  Velocidade:  %ld m/h \n\n\n\n\r Bateria: %ld %% \n\n\n\n\r
Distancia: %ld km\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\r",dutie,veloc,status,distancia);
}
if( (bateria <= 998)  && (bateria > 963)){
status=60;
printf("PWM: %ld;\n  Velocidade:  %ld m/h \n\n\n\n\r Bateria: %ld %% \n\n\n\n\r
Distancia: %ld km\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\r",dutie,veloc,status,distancia);
}
if( (bateria <= 963)  && (bateria > 893)){
status=30;
printf("PWM: %ld;\n  Velocidade:  %ld m/h \n\n\n\n\r Bateria: %ld %% \n\n\n\n\r
Distancia: %ld km\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\r",dutie,veloc,status,distancia);
}
if( (bateria <= 893)  && (bateria > 802)){
printf("PWM: %ld;\n  Velocidade:  %ld m/h \n\n\n\n\r Bateria: Low - Recarregar\n\n\n\n\r
Distancia: %ld km\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\r",dutie,veloc,distancia);
}
if(bateria <= 802){
printf("PWM: %ld;\n  Velocidade:  %ld m/h \n\n\n\n\r Bateria: Estado crítico -
Pare\n\n\n\n\r Distancia: %ld km\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\r",dutie,veloc,distancia);
}
}//fim do if do relogio
}// fim do while flag 0
}//fim da flag 0


if (flag == 1){

transfere_inst(1); // limpa lcd
instrucao=128;          //escreve na primeira linha
transfere_inst(instrucao);
caracter = 'P';
transfere_carac(caracter);
```

```
caracter = ':';
transfere_carac(caracter);
instrucao=135;          //escreve na 10ª posição da primeira linha
transfere_inst(instrucao);
caracter = 'V';
transfere_carac(caracter);
caracter = 'D';
transfere_carac(caracter);

instrucao=192;          //escreve na segunda linha
transfere_inst(instrucao);
caracter = 'V';
transfere_carac(caracter);
caracter = 'l';
transfere_carac(caracter);
caracter = 'c';
transfere_carac(caracter);
caracter = ':';
transfere_carac(caracter);

while(flag == 1){

if (relogio >= 2404){ //atualiza os dados a cada 1/4 de segundo; timer de 104us (104us * 2404= 250ms)

 relogio=0;
 limite++;
 instrucao=130; // coloca o cursor na 3ª posição da 1ª linha
 transfere_inst(instrucao);

 dif= veloc_d-veloc;
```

```
if (dif > 0){
  dutie2=dutie2+2;
  }
 else if (dif < 0){
  dutie2=dutie2-2;
  }
 else{
  dutie2=dutie2;
  }

 if (dutie2>=1023){
  dutie2=1023; }
 if (dutie2<=0){
  dutie2=0; }
 pwm(dutie2);
 set_pwm2_duty(dutie2);
 instrucao=137;          //escreve na 12ª posição da primeira linha
transfere_inst(instrucao);
velocidade(veloc_d);
 if((limite>=3)&&(dutie2<30)){
  veloc=0;
  }
 instrucao=196; // coloca o cursor na 5ª posição da 2ª
 transfere_inst(instrucao);
 velocidade(veloc);
 instrucao=207; // coloca o cursor na ª posição da 1ª linha
 transfere_inst(instrucao);
 transfere_carac(flag+48);
 set_adc_channel(1);// lê a ADC no A1
 delay_us(60);
 bateria=read_adc();     //lê da ADC
```

```
        if( (bateria <= 1023)  && (bateria > 998)){
        status=100;
         printf("Modo Cruise Control:\n\n\r PWM: %ld;\n\n\r  Velocidade desejada:  %ld
m/h;\n\n\n\r  Velocidade Real: %ld m/h \n\n\n\n\r Bateria: %ld %% \n\n\n\n\r Distancia: %ld
km\n\n\n\n\n\n\n\n\n\n\r",dutie2, veloc_d,veloc,status,distancia);
        }
        if( (bateria <= 998)  && (bateria > 963)){
        status=60;
         printf("Modo Cruise Control:\n\n\r PWM: %ld;\n\n\r  Velocidade desejada:  %ld
m/h;\n\n\n\r  Velocidade Real: %ld m/h \n\n\n\n\r Bateria: %ld %% \n\n\n\n\r Distancia: %ld
km\n\n\n\n\n\n\n\n\n\n\r",dutie2, veloc_d,veloc,status,distancia);
        }
        if( (bateria <= 963)  && (bateria > 893)){
        status=30;
         printf("Modo Cruise Control:\n\n\r PWM: %ld;\n\n\r  Velocidade desejada:  %ld
m/h;\n\n\n\r  Velocidade Real: %ld m/h \n\n\n\n\r Bateria: %ld %% \n\n\n\n\r Distancia: %ld
km\n\n\n\n\n\n\n\n\n\n\r",dutie2, veloc_d,veloc,status,distancia);
        }
        if( (bateria <= 893)  && (bateria > 802)){
         printf("Modo Cruise Control:\n\n\r PWM: %ld;\n\n\r  Velocidade desejada:  %ld
m/h;\n\n\n\r  Velocidade Real: %ld m/h \n\n\n\n\r Bateria: Low - Recarregar\n\n\n\n\r
Distancia: %ld km\n\n\n\n\n\n\n\n\n\n\r",dutie2, veloc_d,veloc,distancia);
        }
        if(bateria <= 802){
        printf("Modo Cruise Control:\n\n\r PWM: %ld;\n\n\r  Velocidade desejada:  %ld
m/h;\n\n\n\r  Velocidade Real: %ld m/h \n\n\n\n\r Bateria: Estado crítico - Pare\n\n\n\n\r
Distancia: %ld km\n\n\n\n\n\n\n\n\n\n\r",dutie2, veloc_d,veloc,distancia);
        }
       }//fim do if do relogio
       }// fim do while flag 1
     }// fim da flag 1
   }// fim do while(1)
}// fim do void main()
```