ESCOLA SUPERIOR
POLITÉCNICO SETÚBAL

TIAGO MIGUEL ANDRADE CABRITA

Sistema de Gestão de Baterias de Tração: Estudo e Desenvolvimento do Sistema BMS do IPS

Relatório de Dissertação/projeto/estágio/projeto de investigação do Mestrado em Engenharia Eletrotécnica e de Computadores

ORIENTADORES
Victor Manuel Esteves Antunes
Ana Luísa Lopes Antunes

Dezembro 2024

TIAGO MIGUEL ANDRADE CABRITA

Sistema de Gestão de Baterias de Tração: Estudo e Desenvolvimento do Sistema BMS do IPS

JÚRI

Presidente: Doutor Armando José Pinheiro Marques Pires, Instituto Politécnico de Setúbal
Orientador: Doutor Victor Manuel Esteves Antunes, Instituto Politécnico de Setúbal
Vogal: Doutora Ana Luísa Lopes Antunes, Instituto Politécnico de Setúbal
Vogal: Doutor Manuel Mota Ferreira, Instituto Politécnico de Setúbal

Dezembro 2024

Resumo

Nos últimos anos a indústria automóvel tem sofrido uma revolução relativamente ao conceito de mobilidade. O aumento da consciência ambiental e a crescente necessidade de reduzir a dependência de combustíveis fósseis deram origem aos veículos elétricos como uma solução promissora. Nesse contexto, a criação de *Battery Management System* (BMS) desempenha um papel crucial no avanço da transição para a eletrificação da mobilidade.

Os sistemas BMS desempenham um papel crucial na gestão e monitorização das baterias, garantindo a segurança e assegurando uma utilização correta da bateria, o que contribui para aumentar o tempo de vida útil das mesmas.

Esta tese tem como foco explorar a crescente importância de um sistema de gestão de baterias para veículos elétricos, enfatizando o seu valor tanto no mercado como no desenvolvimento tecnológico e na preservação ambiental. Especificamente, centra-se no desenvolvimento e programação da nova versão do sistema BMS do Instituto Politécnico de Setúbal (IPS). Além disso, é analisada a aplicação do Protocolo Manchester nas comunicações por infravermelhos com o propósito de aumentar a velocidade de comunicação entre os componentes do BMS.

Palavras-chave: sistema de gestão de baterias; veículos elétricos; protocolo de comunicação por infravermelhos; BMS IR; baterias de lítio.

Abstract

In recent years, the automotive industry has undergone a revolution regarding the concept of mobility. The increased environmental awareness and the growing need to reduce dependence on fossil fuels have led to electric vehicles as a promising solution. In this context, the creation of Battery Management Systems (BMS) plays a crucial role in advancing the transition to the electrification of mobility.

BMS systems play a key role in managing and monitoring batteries, ensuring safety and proper battery use, which helps increase their lifespan.

This thesis focuses on exploring the growing importance of a battery management system for electric vehicles, emphasizing its value both in the market and in technological development, as well as environmental preservation. Specifically, it focuses on the development and programming of the new version of the BMS system at the Polytechnic Institute of Setúbal (IPS). Additionally, it analyzes the application of the Manchester Protocol in infrared communications to increase the communication speed between BMS components.

Keywords: battery management system; electric vehicles; infrared communication protocol; BMS IR; lithium batteries.

Agradecimentos

Gostaria de expressar os meus sinceros agradecimentos a todas as pessoas e instituições que contribuíram de maneira significativa para a realização deste trabalho.

Primeiramente, gostaria de agradecer ao meu orientador Professor Victor Antunes pela sua orientação excecional, apoio constante e valiosos conhecimentos ao longo deste processo. Também agradeço à minha coorientadora Professora Ana Antunes, pela sua contribuição e apoio neste projeto.

Agradeço também ao Instituto Politécnico de Setúbal e à Transvetra TVT, pelo acesso a recursos e ferramentas que foram essenciais para a conclusão deste projeto. Aos meus colegas e amigos destas organizações que ensinaram lições valiosas e transmitiram os seus conhecimentos.

Aos meus amigos e familiares, expresso minha gratidão pelo apoio emocional, compreensão e encorajamento que me proporcionaram durante toda a minha jornada acadêmica. Gostaria também de expressar o meu agradecimento à Laura Sofia Arrieta Pacheco, pela sua paciência, carinho e orientação.

Por fim, expresso a minha gratidão às inspirações e referências que moldaram a minha jornada académica e pesquisa, inspirando-me a procurar excelência e inovação. Este trabalho não teria sido possível sem o apoio e contribuições de todas essas pessoas e instituições, pelas quais estou profundamente grato.

Índice

Lista de figuras

Lista de tabelas

Lista de siglas e acrónimos

ACK	*Acknowledge*
ADC	*Analog-to-Digital Converter*
AN	*Application Note*
BMS	*Battery Management System*
CAN-BUS	*Controller Area Network Bus*
CCP	*Capture/Compare/PWM (Pulse Width Modulation)*
CS	*Chip Select*
DC	*Direct Current*
EEPROM	*Electrically Erasable Programmable Read-Only Memory*
I2C	*Inter-Integrated Circuit*
ICD	*In-Circuit Debugger*
IDE	*Integrated Development Environment*
IEEE	*Institute of Electrical and Electronics Engineers*
IR	*Infrared*
IO	*Input/Output*
IoT	*Internet of Things*
IPS	*Instituto Politécnico de Setúbal*
MCU	*Microcontroller Unit*
MISO	*Master In Slave Out (SPI bus)*
MOSI	*Master Out Slave In (SPI bus)*
MOSFET	*Metal-Oxide-Semiconductor Field-Effect Transistor*
MSB	*Most Significant Bit*
NVM	*Non-Volatile Memory*
NRZ	*Non-Return to Zero*
PCB	*Printed Circuit Board*
PPS	*Programmable Peripheral Select*
PWM	*Pulse width modulation*
RAM	*Random Access Memory*
RF	*Radio Frequency*
ROM	*Read-Only Memory*
RTC	*Real-Time Clock*
RTOS	*Real-Time Operating System*
SD	*Secure Digital*
SOC	*State of Charge*
SOH	*State of Health*
SOF	*State of Function*
SCL	*Serial Clock (I2C bus)*
SDA	*Serial Data (I2C bus)*
SPI	*Serial Peripheral Interface*
TWI	*Two-Wire Interface, synonymous with I2C*
UART	*Universal Asynchronous Receiver-Transmitter*
USART	*Universal Synchronous/Asynchronous Receiver-Transmitter*
WDT	*Watchdog Timer*

1. Introdução

Esta tese explora o funcionamento de um BMS para veículos elétricos, analisando a implementação de um novo *hardware* e *software* para uma nova versão do sistema desenvolvido pelo Instituto Politécnico de Setúbal (IPS) e patenteado em conjunto com a empresa TVT Transvetra.

O tema desta dissertação foi determinado em conjunto com a TVT Transvetra, empresa de acolhimento especializada na indústria de transformação de veículos elétricos, em ambiente empresarial. Dado o interesse na abordagem do tema de sistemas de gestão de baterias aliado aos trabalhos de conversão de veículos feitos na empresa, concluiu-se que seria vantajoso direcionar este estudo para a pesquisa e desenvolvimento de uma nova versão do sistema BMS, a ser instalado na próxima geração de veículos convertidos em elétricos pela TVT Transvetra.

1.1 Descrição do Trabalho desenvolvido

Este trabalho focou-se no desenvolvimento e programação de um novo sistema BMS, com o intuito de melhorar a gestão e monitorização das baterias dos veículos elétricos da TVT Transvetra. O principal objetivo foi introduzir avanços tecnológicos que aumentassem a eficiência e a fiabilidade do sistema, dando especial atenção à melhoria das comunicações internas entre os vários componentes, através da implementação do Protocolo Manchester por infravermelhos.

O projeto envolveu o desenvolvimento de um novo protocolo de comunicações. A integração do deste protocolo não só procurou aumentar a velocidade de comunicação, como também reforçar a precisão na troca de dados essenciais para o correto funcionamento do sistema.

Adicionalmente, foram realizados testes para validar a eficácia das melhorias introduzidas.

1.2 Estrutura

Esta dissertação está estruturada em vários capítulos que abordam, de forma abrangente, os temas relacionados com os Sistemas de Gestão de Baterias (BMS), o BMS IR do Instituto Politécnico de Setúbal (IPS) e o protocolo Manchester. Estes tópicos são fundamentais para a descrição e implementação do protocolo de comunicações do BMS IR.

A tese está organizada em sete capítulos:

No capítulo 1, é apresentado o sistema BMS, incluindo o funcionamento das baterias, as funções e operações do BMS, bem como os métodos de proteção, monitorização e controlo da temperatura, balanceamento das células e comunicação.

O capítulo 2 foca-se no BMS IR do IPS, abrangendo a introdução ao sistema e a sua arquitetura, redes de comunicação, sensorização, estimativas de autonomia e estado de saúde das baterias, gestão de carregamento e balanceamento, monitorização de dados, diagnóstico de falhas e a aplicação deste BMS em veículos elétricos transformados.

Quanto ao capítulo 3, este introduz os princípios de operação do protocolo Manchester. Descrevem-se os métodos de codificação e descodificação de mensagens, juntamente com as suas aplicações em sistemas embebidos. Também se compara o protocolo Manchester com outros protocolos. São apresentados casos de estudo e exemplos de aplicações industriais. Por fim, são discutidos os formatos de tramas e os métodos de deteção de erros.

Já no capítulo 4 é apresentada a implementação de um novo protocolo de comunicações para o BMS IR, com base no protocolo Manchester. Inicia-se com a introdução ao problema e aos objetivos específicos, seguida de uma análise das soluções existentes conforme as *Application Notes* da

Microchip. A proposta de solução é detalhada, incluindo o desenvolvimento de algoritmos, a implementação prática, os testes realizados e as conclusões derivadas dos resultados.

Relativamente ao capítulo 5, este descreve os testes realizados para validar o novo protocolo e os resultados obtidos. A análise dos resultados serve como base para as conclusões sobre a eficácia e viabilidade do novo protocolo no sistema BMS IR.

Por fim na conclusão oferece-se uma síntese dos principais resultados, discutindo a sua interpretação e as contribuições para a área de estudo. São identificadas as limitações do trabalho e exploradas as suas implicações práticas, além de serem sugeridas direções para o trabalho futuro.

2. Estado da arte: Sistema BMS em veículos elétricos

Este capítulo explora os fundamentos e as inovações tecnológicas dos sistemas de gestão de baterias em veículos elétricos, enfatizando a sua importância para a segurança, desempenho e longevidade das baterias, bem como o seu papel crucial na viabilização e expansão do mercado de veículos elétricos e híbridos.

2.1 Introdução

De dispositivos eletrónicos portáteis a veículos elétricos, as baterias são amplamente utilizadas como fonte principal de energia em várias aplicações. O interesse nas baterias para veículos elétricos remonta ao século *XIX*, quando os primeiros veículos elétricos surgiram [1]. Atualmente, os veículos elétricos têm o potencial de reduzir o consumo de gasolina em até 75%, o que trouxe uma atenção significativa às baterias no mercado automóvel [1].

Nos Estados Unidos, o *U.S. Council for Automotive Research* (USCAR) e o *U.S. Advanced Battery Consortium* (USABC) estabeleceram normas de desempenho para apoiar a comercialização de baterias avançadas em veículos elétricos e híbridos [1]. Contudo, a expansão do mercado destes tipos de veículos depende não só da tecnologia da bateria, mas também do sistema de gestão de bateria. Um BMS fiável é crucial para otimizar o desempenho da bateria e garantir a operação segura destes veículos.

Um BMS abrangente, tal como o sistema de gestão do motor nos veículos a gasolina, é responsável por monitorizar parâmetros críticos, como o estado de saúde da bateria, a utilização, o desempenho e a longevidade. Embora populares, as baterias de iões de lítio apresentam riscos, como a volatilidade e inflamabilidade; o sobrecarregamento pode levar a uma situação de fuga térmica, uma condição que pode provocar incêndios ou explosões [2] [3]. Este risco torna essencial um BMS robusto para mitigar tais perigos, especialmente em aplicações de veículos elétricos [2] [3]. Além disso, a descarga excessiva pode causar uma perda irreversível de capacidade devido a reações químicas dentro da bateria [1].

O rápido crescimento do mercado de veículos elétricos e híbridos sublinhou a importância de tecnologias avançadas de BMS. Os sistemas de BMS são responsáveis por monitorizar parâmetros vitais como a tensão, a corrente, a temperatura e o estado de carga (SoC), enquanto equilibram a distribuição de carga entre as células para garantir uma operação eficiente e prolongar a vida útil da bateria [2] [3] [4]. A tecnologia de BMS sem fios, destacada pelos avanços na indústria, está a ser cada vez mais utilizada para melhorar a fiabilidade e segurança dos pacotes de baterias em VE [4] [5].

Os sistemas de BMS estão em constante evolução, integrando algoritmos mais avançados para a manutenção preditiva, deteção de falhas e comunicação de dados em tempo real através de *internet of things* (IoT) [2] [5] [6]. Estas inovações deverão melhorar o desempenho da bateria, prolongar a sua vida útil e otimizar a eficiência energética, apoiando, assim, a adoção generalizada de veículos elétricos e híbridos [2] [6].

2.2 Funções do BMS

Um BMS monitoriza e controla a bateria com base nos circuitos de aquisição de dados e de segurança incorporados no sistema. Sempre que forem detetadas quaisquer condições anormais, o BMS deve notificar o utilizador e executar o procedimento de correção predefinido [1]. Na Figura 1 pode-se observar o conjunto de funções que um BMS deverá abranger.

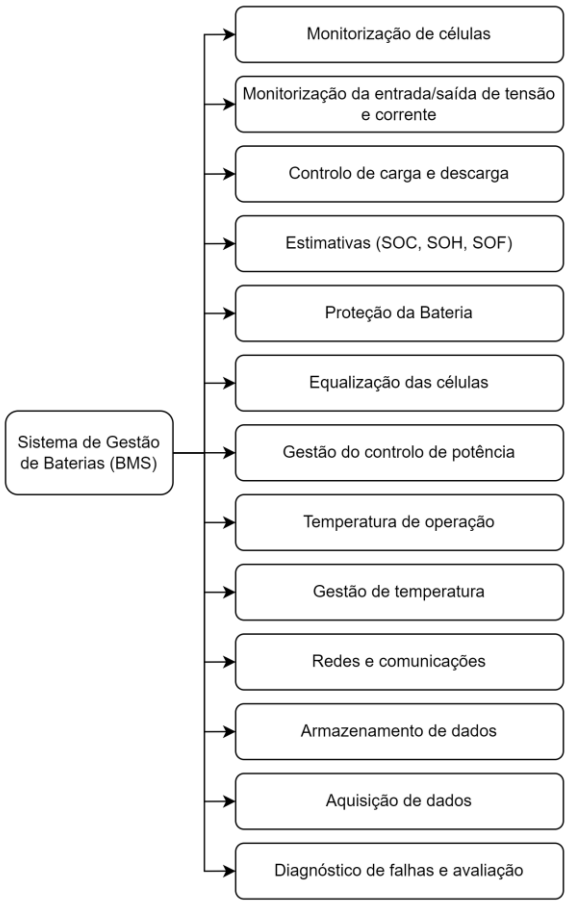


Figura 1 - Funções necessárias de um BMS [5].

O sistema de gestão da bateria é responsável por todos os equipamentos de controlo e gestão relacionados com o armazenamento e transferência de energia nos veículos elétricos. Este sistema inclui o controlo de carga e descarga, monitorização, balanceamento da tensão das células e da bateria, verificação do estado de carga da bateria, corrente de entrada/saída, monitorização de tensão, controlo de temperatura, proteção da bateria, diagnóstico de falhas e avaliação [5].

2.3 Proteção da bateria, balanceamento e equalização das células

As baterias de lítio são utilizadas em sistemas de armazenamento de energia em veículos elétricos, com um número de células conectadas em série e/ou paralelo. A bateria de um veículo elétrico pode ser carregada a partir de uma fonte de alimentação e descarregada para acionar o motor do veículo e outros sistemas [5].

O ciclo consecutivo de carga-descarga pode causar um desequilíbrio na tensão e carga entre a bateria células devido a mudanças nas suas características físicas. Este desequilíbrio acontece devido a defeito de fabrico, temperatura e/ou envelhecimento. Perfis de tensão e carga desequilibrados podem reduzir o desempenho geral e a duração do fornecimento de energia dos sistemas de armazenamento [5]. A Figura 2 mostra um exemplo do desequilíbrio entre as células.

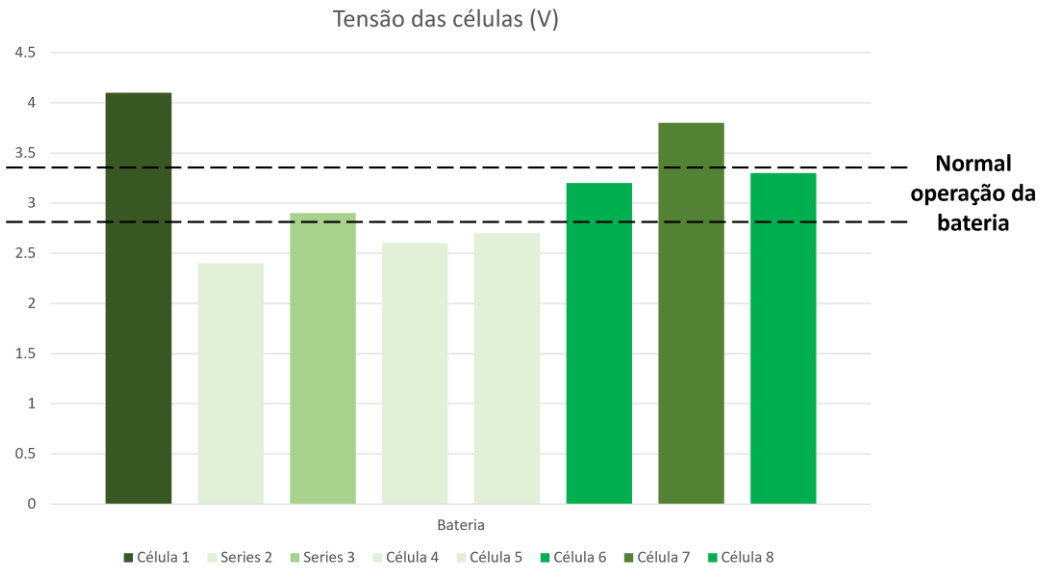


Figura 2 - Exemplo do desequilíbrio de carga entre as células da bateria.

A sobrecarga pode causar a explosão da célula, enquanto a subcarga pode danificar as propriedades químicas da bateria e reduzir a vida útil das células da bateria.

O BMS pode interromper o processo de carregamento e descarregamento da bateria quando uma das células estiver fora dos limites seguros de operação. De maneira a evitar que a bateria possa vir a perder o nível de carga nominal necessário para uma operação ideal. Portanto, um controlo de equilíbrio de carga, é essencial, para proteger as células da bateria. Permitindo assim, preservar ao máximo a capacidade de armazenamento das mesmas.

A monitorização individual e equalização de carga da bateria, é feita pelo BMS. Protege as células da bateria de sobrecarga e subcarga, e melhora o desempenho geral de um sistema de armazenamento de energia de um veículo elétrico de forma eficaz [5]. O BMS monitoriza continuamente a tensão, corrente e temperatura das células, e deteta uma célula desequilibrada através da estimativa do estado de carga da bateria. O processo de equalização é efetuado com uma dissipação do excesso de carga da célula sobrecarregada. Fazendo com que a carga das células equilibre a um nível de tensão dentro dos limites de operação. A Figura 3 mostra um cenário de um conjunto de células predominantemente equilibradas, onde algumas células podem ter a tensão ligeiramente mais altas, mas ainda assim são consideradas carregadas.

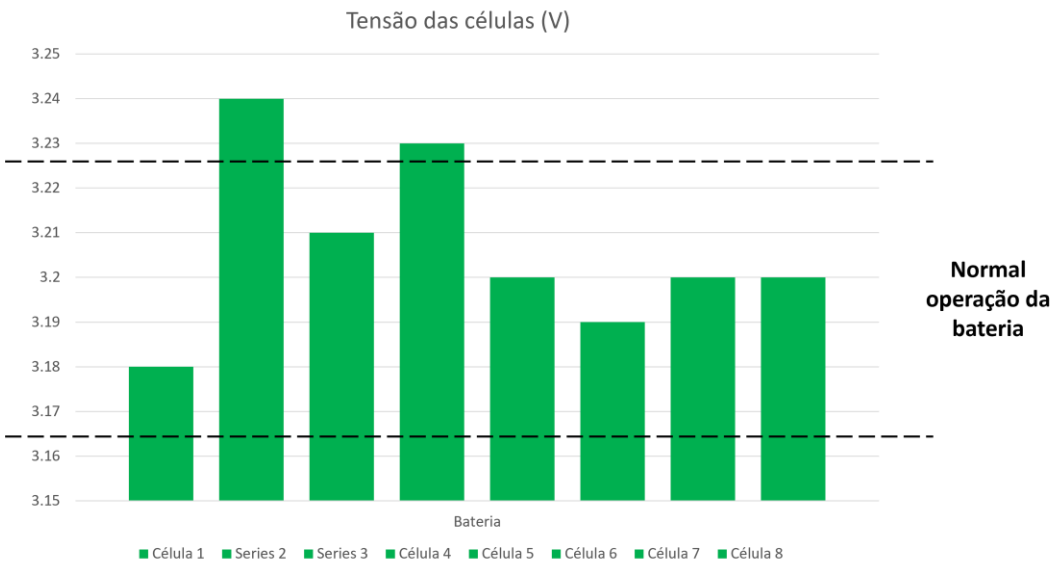


Figura 3 - Exemplo da equalização de carga por célula da bateria.

Um sistema de equilíbrio de carga permite proteger as baterias de danos irreversíveis e melhorar a eficiência e o tempo de vida útil da bateria. No entanto, um veículo elétrico pode utilizar mais de noventa células numa bateria, dificulta a monitorização e a gestão de carga devido à complexidade do sistema. Por isso é necessário desenvolver um BMS com um design simples e inteligente.

2.4 Estimativa do estado da bateria

Os estados da bateria incluem o estado de carga, o estado de saúde e o estado de funcionamento.

Estado de Carga:
O *State of Charge* (SOC) define o estado de carga da bateria e pode ser estimado por diversos métodos, dos quais de destacam três principais:

1. **Método de Coulombimetria**: Este método baseia-se na contagem da carga que entra e sai da bateria. A corrente é integrada ao longo do tempo para determinar a quantidade de carga armazenada. É um método simples, mas suscetível a erros devido ao acumular de desvios ao longo do tempo [7].

2. **Método de tensão de circuito aberto**: A tensão da célula da bateria em circuito aberto tem uma relação não linear com o SOC. Através de tabelas ou modelos, é possível associar a tensão medida ao SOC. No entanto este método mesmo quando a bateria está em repouso apresenta um erro significativo [8].

3. **Filtro de Kalman**: Este método utiliza um modelo do sistema da bateria combinado com medições de tensão e corrente para estimar o SOC. O filtro de Kalman nas suas variações como o *Extentended Kalman Filter* (EKF) ou o *Unscented Kalman Filter* (UKF) consegue lidar com incertezas e ruído nas medições, proporcionando estimativas mais precisas [9].

Estado de Saúde:
O *State of Health* (SOH) reflete a condição atual da bateria em comparação com o seu estado nominal. No caso dos veículos elétricos, o foco principal está na capacidade da bateria, enquanto nos veículos híbridos a principal preocupação é a potência. O SOH pode ser estimado de várias formas:

1. **Método baseado na impedância**: O aumento da impedância interna da bateria indica degradação. A variação da impedância ao longo do tempo pode ser monitorizada para estimar o SOH [10].

2. **Monitorização da capacidade**: A capacidade útil da bateria diminui à medida que envelhece. Comparar a capacidade atual com a capacidade nominal permite fornecer uma estimativa do SOH. Isto pode ser feito através de testes periódicos de descarga completa ou pela análise dos padrões de utilização normais [10].

Estado de funcionamento

O *State of Function* (SOF) define a capacidade real da bateria para atender à procura de energia do sistema do veículo elétrico. O SOF pode ser calculado com base em três fatores principais:

1. **SOC e SOH estimados**: O SOF é diretamente influenciado pelo SOC e pelo SOH. Uma bateria com baixo SOC ou SOH terá um SOF reduzido, uma vez que a sua capacidade de fornecer energia será limitada [10].

2. **Perfil de carga/descarga**: O SOF também depende das condições de utilização, como a taxa de carga e descarga da bateria. Se a bateria for descarregada rapidamente, a sua eficiência e capacidade para responder à procura podem ser afetadas, resultando num SOF mais baixo [10].

3. **Temperatura**: A temperatura da bateria afeta a sua capacidade de fornecer energia. Temperaturas muito elevadas ou muito baixas podem reduzir temporariamente o SOF [10].

Embora o SOC indique a quantidade de carga disponível e o SOH avalie a saúde geral da bateria, o SOF reflete a capacidade real da bateria funcionar adequadamente nas condições atuais de operação, considerando esses dois estados e outros fatores.

Sem gestão, a energia para acionar o sistema de tração e outras cargas, terá um mau aproveitamento [5]. A integração de um sistema de gestão e controlo é a solução para gerir a distribuição de energia e o controlo de carga do veículo de forma eficiente e inteligente.

A determinação dos valores de SOC, SOH e SOF do sistema de armazenamento, permite melhorar o desempenho do sistema, a durabilidade, a segurança e a qualidade de energia, com o objetivo de reduzir o número de falhas e custos de manutenção [5].

2.5 Controlo da temperatura de operação

O sistema de armazenamento de energia num veículo elétrico deve operar preferencialmente dentro de uma gama ideal de temperaturas. O sistema de gestão da bateria desempenha um papel fundamental ao monitorizar a temperatura e garantir que a bateria permanece dentro desta gama ideal de operação, conforme indicado pelo fabricante. Desta forma o BMS assegura que a bateria funcione de forma eficiente e segura, prevenindo condições de operação fora dos limites recomendados.

Com o auxílio de sensores de temperatura no sistema de armazenamento, o sistema de gestão e refrigeração podem atuar de forma a preservar o estado de saúde das células.

Para garantir o bom funcionamento das células da bateria, é necessária uma distribuição uniforme e adequada da temperatura durante a operação do veículo elétrico, tanto durante a carga quanto a descarga. Como ilustra a Figura 4, uma célula de lítio deve operar preferencialmente dentro de uma gama de temperaturas mais restrita do que os extremos entre -40ºC e 60ºC, pois, quando exposta a temperaturas próximas a esses limites, pode ocorrer uma degradação da sua estrutura química. Isso resulta numa variação do SOH, redução dos ciclos de vida e diminuição da capacidade de carga da célula.

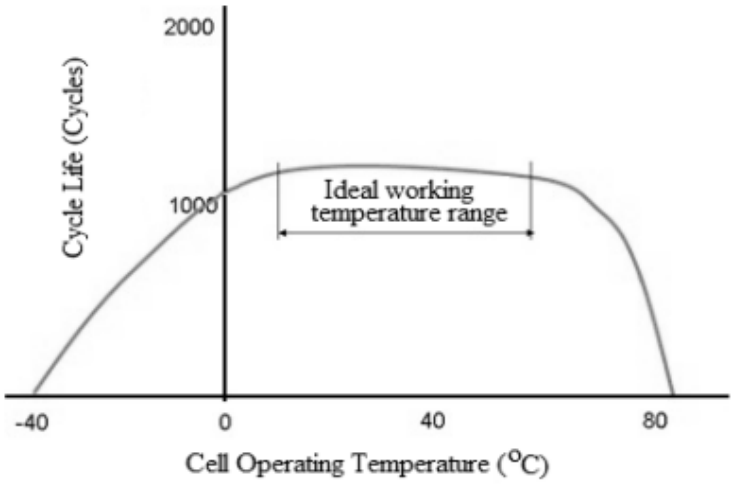


Figura 4 - Gama ideal de operação das células [5].

Com base na parametrização de controlo da temperatura do BMS no sistema de armazenamento do veículo elétrico, aciona-se o sistema de refrigeração ou aquecimento, para regular a temperatura. De forma a manter as condições ideais de operação [5].

2.6 Monitorização e Rede de comunicações

O desempenho do sistema de um veículo elétrico depende fortemente de uma comunicação robusta entre os vários subsistemas. Esta comunicação é crucial para otimização do desempenho do veículo elétrico, facilitar a monitorização, permitir a instalação de *software*/*firmware* e garantir o controlo eficaz das modificações dentro do sistema de gestão de baterias [5].

Internamente, as duas formas de comunicação com os sistemas de monitorização das células são através de conexões com fio e sem fio. As conexões com fio, como o *Controller Area Network* Bus (CAN-BUS), proporcionam uma comunicação robusta e em tempo real entre a unidade de controlo central do BMS e as células individuais da bateria, permitindo a monitorização das tensões, temperaturas e correntes das células, bem como a implementação de estratégias de balanceamento e proteção [4]. Na Figura 5, pode-se observar um sistema de comunicações por CAN-BUS.

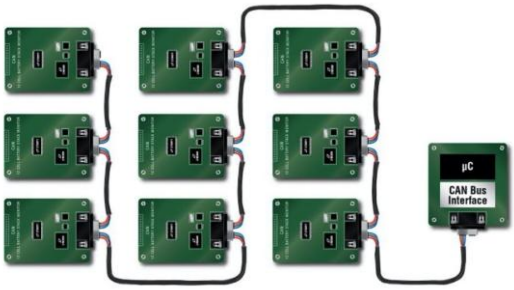


Figura 5 - BMS com comunicações por fios [4].

Por outro lado, tecnologias de comunicações sem fios, como *Bluetooth* ou *WiFi*, podem ser utilizadas para a monitorização remota e diagnóstico das células da bateria. Estas conexões sem fios permitem que técnicos ou operadores acedam aos dados do BMS à distância, realizem atualizações de *firmware* e efetuem diagnósticos sem a necessidade de acesso físico à bateria [4]. Além disso, estas tecnologias apresentam a vantagem adicional de proporcionar, de forma inerente, o isolamento galvânico

necessário entre a bateria e os restantes componentes do veículo, assegurando uma maior segurança no sistema. A Figura 6, ilustra um sistema de gestão de baterias sem fios.

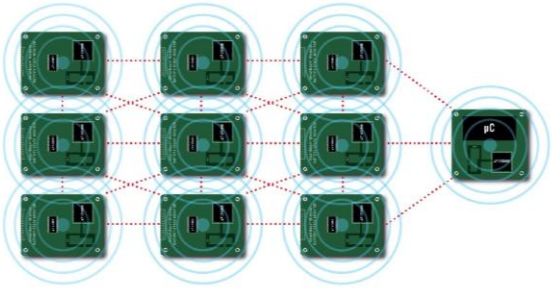


Figura 6 - BMS com comunicações sem fios [4].

Por exemplo, a monitorização em tempo real de parâmetros críticos, como Estado de Carga (SOC) e Estado de Saúde (SOH) do sistema de armazenamento de bateria, é essencial para manter o seu desempenho e longevidade. Esta monitorização é possível através de protocolos avançados de comunicação, como o *CAN-BUS*, que forma um barramento de comunicações internas dentro do veículo.

Adicionalmente, a integração da tecnologia do *Global Positioning System* (GPS) expande ainda mais as capacidades do sistema do veículo elétrico. O GPS não só auxilia na identificação precisa de estações de carregamento de veículo elétrico próximas, mas também desempenha um papel crucial na previsão da autonomia. Ao utilizar dados do GPS em conjunto com a comunicação CAN-BUS, os veículos elétricos podem gerir de forma inteligente o consumo de energia, otimizar estratégias de carregamento e melhorar a sua eficiência em geral [5].

3. BMS IR do Instituto Politécnico de Setúbal

3.1 Introdução

O Sistema de Gestão de Baterias por Infravermelhos (BMS IR), desenvolvido pelo Instituto Politécnico de Setúbal em colaboração com a TVT Transvetra, representa um passo pragmático e inovador no âmbito das tecnologias de veículos elétricos. Este sistema, concebido para uma gestão eficiente de baterias, destaca-se pela sua abordagem única, tendo sido implementado e testado num *tuk-tuk*, em território português. Além disso, é importante salientar que o projeto está protegido por uma patente, reforçando a sua contribuição para o campo das tecnologias de gestão de baterias.

Esta é a segunda versão do BMS, sendo que a primeira versão tinha uma arquitetura diferente da atual. A versão anterior utilizava um barramento SPI entre os diversos módulos, enquanto a versão atual apresenta uma arquitetura mais simples, com uma rede de comunicações por infravermelhos.

3.2 Arquitetura do sistema

Na Figura 7, é apresentada a arquitetura do sistema do BMS IR do Instituto Politécnico de Setúbal.

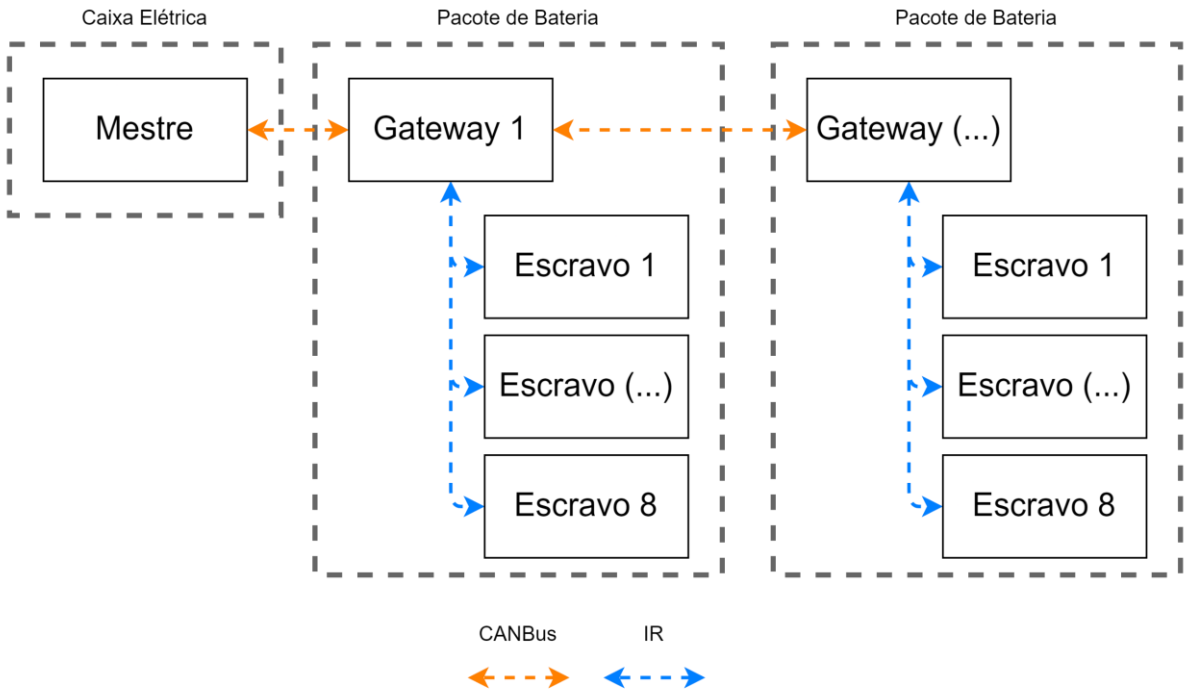


Figura 7 - Arquitetura do BMS IR.

O sistema é composto por um circuito principal de gestão designado de Mestre, ou circuito *Master*, pelos *Gateways* e pelos circuitos de monitorização das células chamados de Escravos (*Slaves*). O Mestre é colocado no exterior da caixa da bateria de tração, enquanto os *Gateways* e os Escravos são colocados no interior. A ligação entre o Mestre e os *Gateways* é feita através do protocolo *Controller Area Network* (CAN-BUS), enquanto a ligação entre os *Gateways* e os Escravos é feita sem recurso a fios, utilizando um protocolo de transmissão de dados por infravermelhos. Os circuitos *Gateway* são responsáveis por encaminhar as mensagens provenientes dos circuitos de monitorização das células (Escravos) para o Mestre e vice-versa. Os circuitos de monitorização de células são alimentados diretamente pelas células da bateria de tração, enquanto o Mestre e os *Gateways* são alimentados pela bateria auxiliar do veículo.

A Figura 8 e a Figura 9 ilustram em mais detalhe, a colocação destes componentes e as suas ligações:

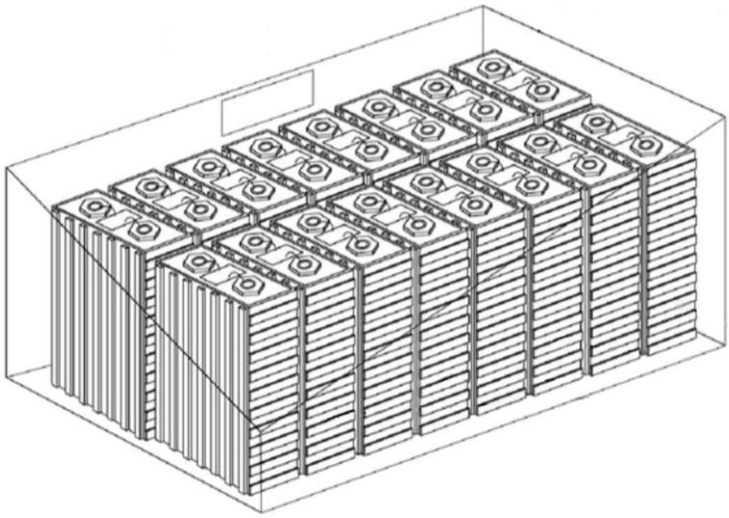


Figura 8 - Esquema de uma parte do sistema do BMS IR dentro do pacote de baterias [11].

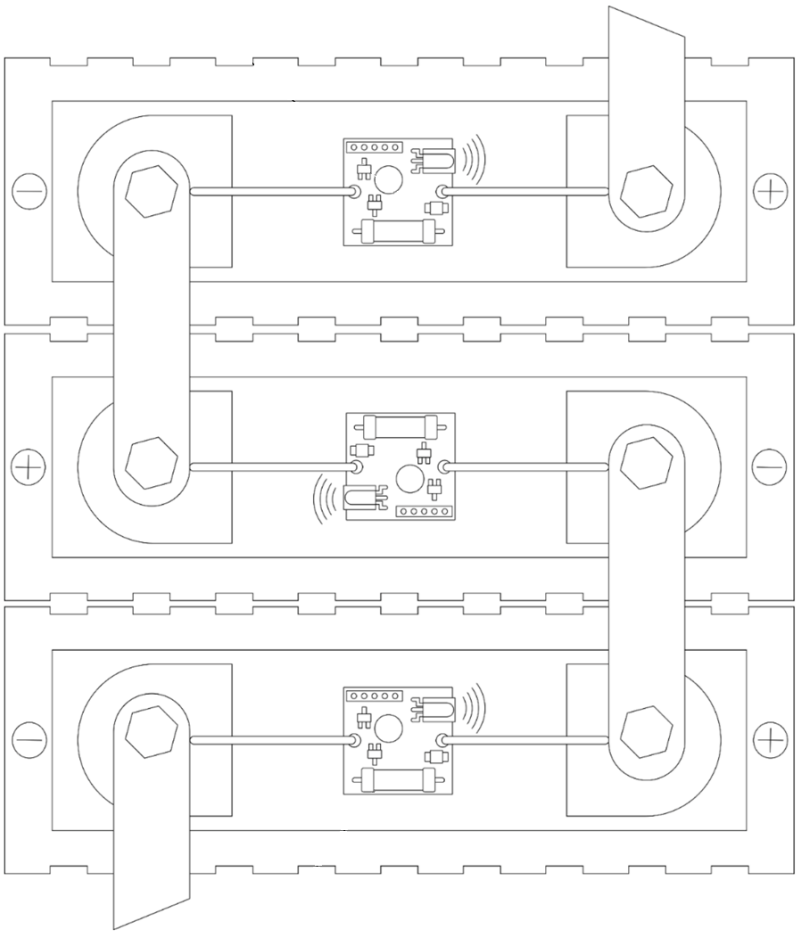


Figura 9 - Detalhe do escravo instalado nas células da bateria [11].

Na Figura 10 é apresentado o desenho da placa do circuito *Mestre*.

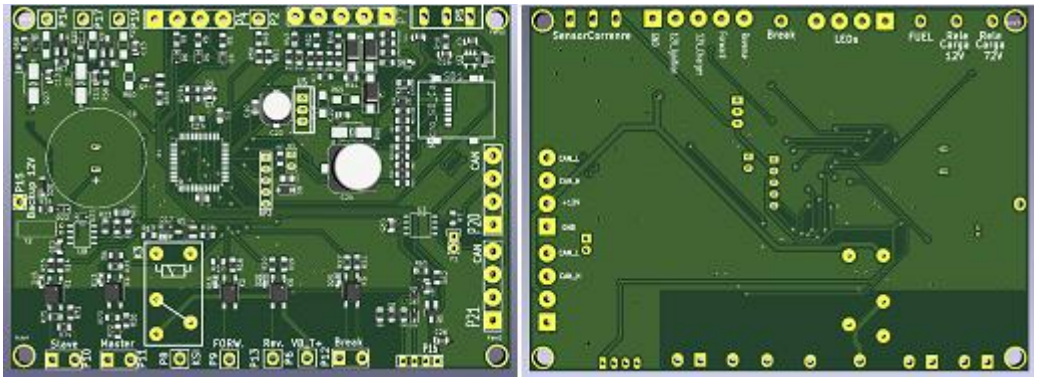


Figura 10 - Modelo tridimensional do BMS IR Mestre.

Já na Figura 11 é apresentada a placa montada.

Figura 11 - BMS IR Mestre.

Quanto ao desenho da placa do Gateway este é apresentado na Figura 12.

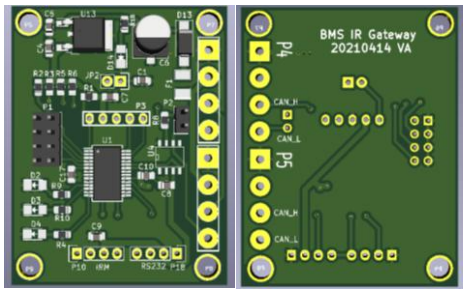


Figura 12 - Modelo tridimensional do BMS IR Gateway.

A placa montada é apresentada na Figura 13.

Figura 13 - BMS IR Gateway.

Na Figura 14 é apresentado o desenho da placa do módulo IR do *Gateway*

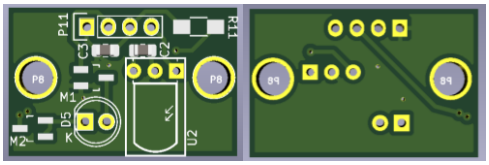


Figura 14 - Modelo tridimensional do módulo IR do Gateway.

A placa montada do módulo é apresentada na Figura 15.

Figura 15 - Módulo IR do Gateway.

O desenho da placa do Escravo é apresentado na Figura 16.

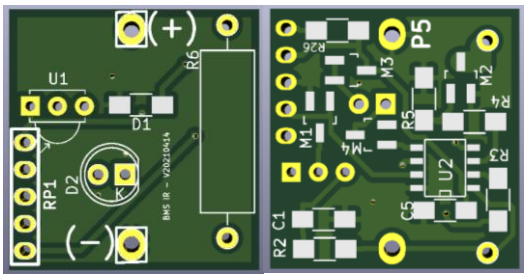


Figura 16 - Modelo tridimensional do BMS IR Escravo.

A placa montada é apresentada na Figura 17.

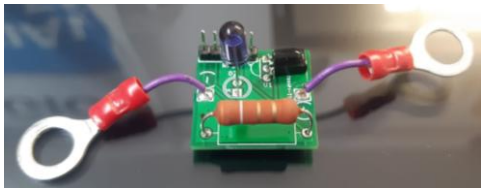


Figura 17 - BMS IR Escravo.

O esquema de alimentação descrito, aliado ao uso da comunicação por infravermelhos, permite minimizar o uso de cablagem, prevenindo assim a perda de contacto devido à vibração do veículo, além de reduzir custos, peso e tempo de montagem [11].

3.3 Característica dos módulos do sistema

A Tabela 1 resume as principais características de processamento, comunicação, entradas/saídas e outros aspetos relevantes de cada módulo do BMS IR.

Tabela 1 - Principais características dos módulos do BMS IR.

Módulo	**Categoria**	**Descrição**
Mestre	Processamento	Microcontrolador PIC18F46K80: microcontrolador de 8 *bits* com 64 Kbytes de memória *Flash* e 4 Kbytes de RAM. Possui um ADC de 12 bits para leituras precisas de sensores. Equipado com módulos *Enhanced Capture/Compare/PWM* (ECCP). [12].
	Comunicações	Interface CAN-BUS: compatível com a norma ISO 11898 para comunicação automotiva robusta e de alta velocidade. Utiliza a tecnologia *Enhanced* CAN (ECAN) com taxas de bits programáveis [12].
		Interface RS-232: comunicação série *full-duplex*, permitindo diagnósticos e configuração de parâmetros. Taxas de *baud rate* ajustáveis para atender os requisitos de equipamentos de diagnóstico externos [12].
	Entradas/Saídas	Seção de Entrada/Saída de 12V: gestão dos sinais da cabine do veículo e controlo do sistema de carregamento AC-DC.
		Outros Relés e Optoacoplador: controlam o variador de velocidade do motor AC usando relés eletromecânicos para corrente alta. Os optoacopladores garantem o isolamento galvânico entre o domínio de baixa tensão do microcontrolador e os circuitos de alta tensão do motor AC.
Gateway	Processamento	Microcontrolador PIC18F25K80: arquitetura de 8 *bits* com 32 Kbytes de memória *Flash* e 2 Kbytes de RAM. Módulo ECAN integrado para comunicação CAN-BUS fiável [13].
	Comunicações	Interface CAN-BUS: projetada para comunicação da indústria automotiva, garantindo a transmissão e receção de mensagens de forma robusta. Oferece grande imunidade a interferências, mantendo a integridade da comunicação [13].
		Interface RS-232: comunicação série para a monitorização do sistema e configuração. Facilita a comunicação bidirecional com sistemas externos que requerem um protocolo série [13].
		Comunicação por infravermelhos (IR): comunicação sem fios com os módulos Escravo, reduzindo conectores físicos o que permite um posicionamento flexível dos componentes.
	Entradas/Saídas	Componentes Infravermelhos: emissor IR: converte sinais elétricos em luz infravermelha (modulação de 38 kHz a 940 nm) para transmitir dados aos módulos Escravo. Recetor IR (38 kHz): Deteta e desmodula sinais IR de 38 kHz, transmitindo o sinal codificado para o microcontrolador do Gateway.
Escravo	Processamento	Microcontrolador PIC16F18313: microcontrolador de ultrabaixo consumo (XLP) com um ADC de 10 *bits* e várias interfaces de comunicação. Design compacto otimizado para baixo consumo de energia, crucial por ser alimentado pelas células que monitoriza [14].
	Comunicações	Comunicação Infravermelha (IR): emissor IR: envia dados dos sensores das células (ex: tensão, temperatura) via sinais infravermelhos para o módulo *Gateway*. Recetor IR (38 kHz): Recebe e decodifica os comandos do *Gateway*, implementando as instruções do Mestre.
	Entradas/Saídas	Resistência de dissipação: resistência de 3 W usada para o equilíbrio da energia das células durante os ciclos de carga, dissipando a energia excedente das células com níveis de carga mais elevados, mantendo o equilíbrio do conjunto de baterias.

3.4 Rede de comunicações

Relativamente às tecnologias utilizadas para a comunicações do sistema BMS IR, o Mestre comunica com o(s) *Gateway*(s) através do protocolo CAN-BUS, tipicamente utilizado na indústria automóvel. O *Gateway*, por sua vez, utiliza um emissor de infravermelhos para comunicar com os vários Escravos da caixa de baterias de tração, formando uma rede coesa que garante um fluxo eficiente de dados.

Vantagens da utilização em sistemas de gestão de baterias de comunicações por infravermelhos [3]:

- Isolamento galvânico entre a bateria e o sistema de gestão de carga;
- Promover a compatibilidade eletromagnética;
- Eliminar o uso de cablagem para comunicação, reduzindo o tempo de montagem, eliminando as falhas relacionadas com a cablagem e diminuindo o peso e os custos de implementação;
- Garantir o um consumo baixo, uma característica crucial para os circuitos do Escravo;
- Garantir tamanho e custo reduzido para os circuitos Escravo.

Neste tipo de aplicação o consumo dos circuitos dos Escravos é crítico, uma vez que são alimentados diretamente pela célula que estão a monitorizar. Outras opções de comunicações sem fios, como *Bluetooth* ou *Wi-Fi*, exigem componentes maiores e mais caros, com consumo de energia significativamente maior, tornando-os pouco recomendados para esta aplicação [3].

3.5 Monitorização e visualização de dados

Estão criadas várias ferramentas de configuração e de análise de dados para o BMS IR. A ferramenta de configuração atual, serve para fazer a configuração inicial dos Escravos e trata-se de um software em *Java* que se conecta à porta de comunicações do Mestre onde é possível enviar e receber informação relativa ao sistema. Esta interface permite a um técnico, realizar uma configuração rápida e automática do sistema acabado de instalar e fazer o respetivo *debug* para a verificação do funcionamento do sistema. Na Figura 18, é apresentada a ferramenta de configuração:

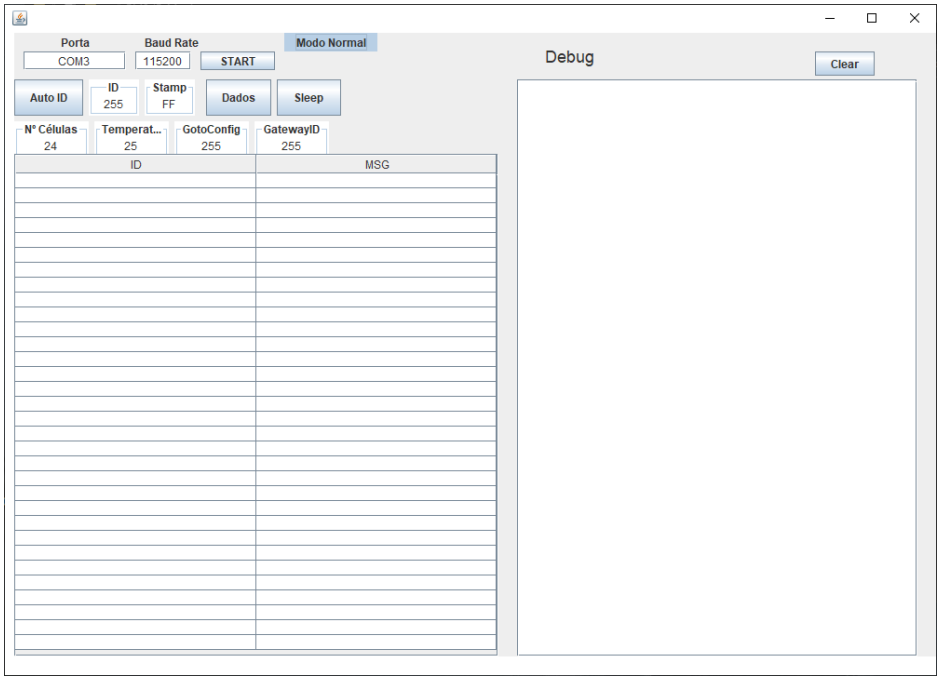


Figura 18 - Software da ferramenta de configuração do BMS IR.

Relativamente aos softwares de monitorização e análise de dados do BMS IR, atualmente existe uma aplicação para a plataforma *Android* que permite ao técnico analisar facilmente todos os dados provenientes do BMS IR. Esta aplicação possibilita visualizar três janelas distintas: a primeira destinada à deteção de defeitos, a segunda para a configuração de parâmetros de carregamento e a última para

a visualização de informações relacionadas com as entradas e saídas, a tensão e temperatura das células da bateria, o estado da bateria, o carregamento e eventuais erros. A última janela mencionada pode ser observada na Figura 19.

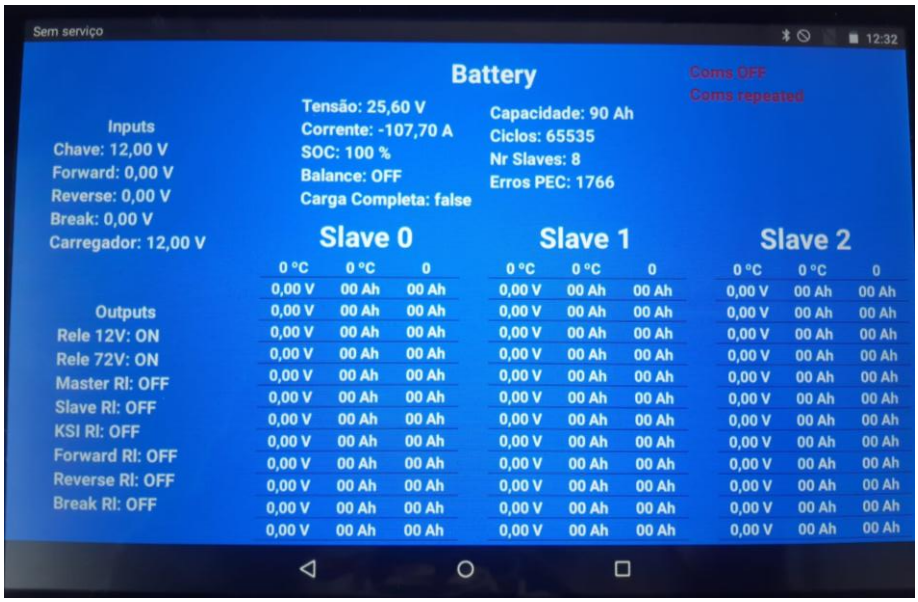


Figura 19 - Aplicação Android para o BMS IR de técnico de manutenção.

A última ferramenta trata-se de um *script* em *Python* capaz de construir gráficos para a visualização de dados em função do tempo. Dados como o SOC, tensão da bateria de tração e auxiliar e estado do BMS IR, entre outros. Para adquirir estes dados, basta apenas descarregar os dados da aplicação *Android* ou do cartão SD normalmente inserido no Mestre. A Figura 20 apresenta um gráfico da tensão de cada uma das células da bateria em função do tempo.

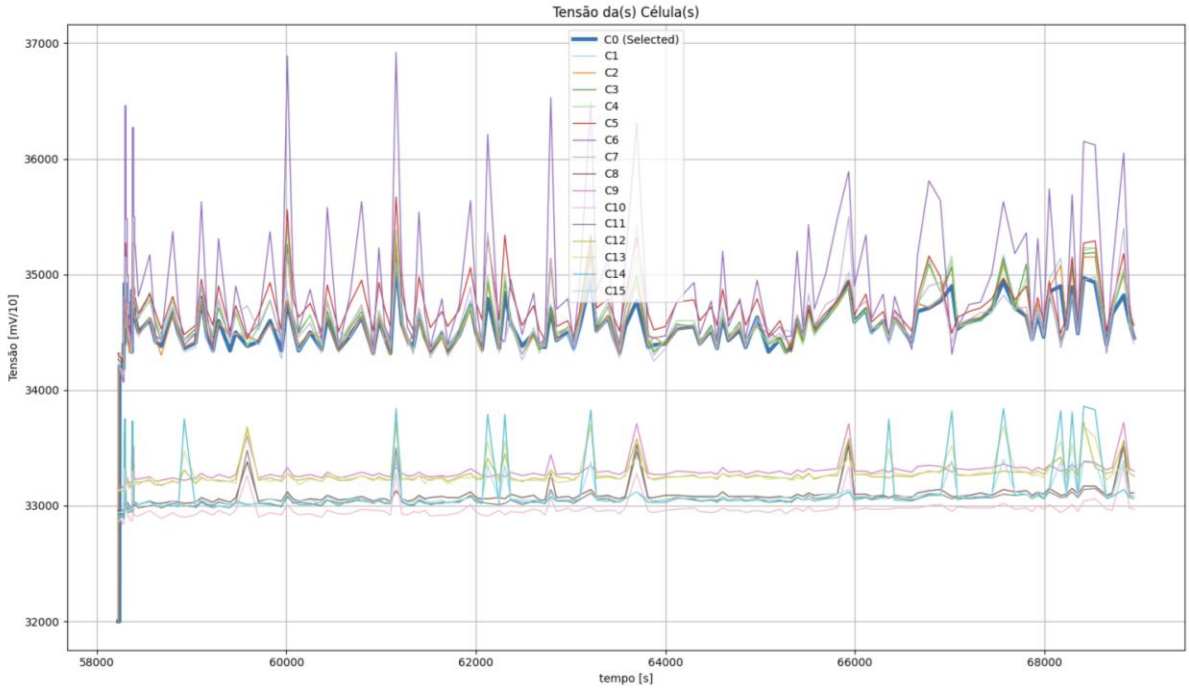


Figura 20 - Gráfico a tensão de cada uma das células da bateria ao longo do tempo.

3.6 Diagnóstico e deteção de problemas

O diagnóstico do BMS IR instalado no veículo, pode ser facilmente realizado com o auxílio das ferramentas anteriormente referidas.

No caso *script* em *Python* permite a visualização de possíveis valores anómalos, os quais poderão ser detetados por um técnico.

Na aplicação *Android* é possível visualizar em tempo real um conjunto de alarmes para diagnosticar um possível problema. Quando um destes alarmes é emitido pelo BMS IR, por segurança, o sistema desliga a bateria de tração do variador de velocidade. Os alarmes emitidos são os seguintes:

- Temperatura Alta: temperatura do sensor interno do microcontrolador do Escravos demasiado alta;
- Sobre tensão: tensão medida nos bordos da célula está demasiado alta;
- Baixa tensão: tensão medida nos bordos da célula está demasiado baixa;
- Falha de comunicações: detetar possíveis falhas de comunicações entre módulos;
- Duplicação de informação: escravos estão mal configurados. Têm o mesmo número de identificação individual.

3.7 Implementação do BMS IR

A primeira implementação feita com o BMS IR, foi no protótipo realizado em dezembro de 2021. Essa implementação foi realizada pelo autor deste documento, no âmbito do projeto de estágio da Licenciatura em Engenharia Eletrotécnica e de Computadores em conjunto com a Transvetra TVT.

Utilizando o Piaggio APE 50 como plataforma, este veículo consistia num sistema de duas caixas de baterias de tração com 8 células cada uma, que se traduz, em termos de *hardware*, num Mestre, dois *Gateways* e dezasseis Escravos. O protótipo foi um sucesso e a prova de que o BMS IR é uma boa solução para este tipo de aplicações. Na Figura 21, apresenta-se uma foto do primeiro teste do veículo.

Figura 21 - Primeiro protótipo com o BMS IR.

O segundo passo foi dado em julho de 2023, quando a Transvetra TVT avançou com a transformação completa de um novo chassis. O cliente deste projeto era a Estufa Fria, pertencente à câmara de Lisboa, que pretendia um moto triciclo silencioso e ágil para movimentar cargas facilmente dentro e fora da estufa. A implementação neste veículo foi também realizada pelo mesmo autor.

É possível observar o veículo na Figura 22.

Figura 22 - Primeiro veículo legalizado com o BMS IR.

4. Protocolo Manchester

Neste capítulo, é explorada a importância do protocolo Manchester no contexto dos sistemas de comunicação digital, com destaque para sua aplicação eficiente na troca de dados. Desenvolvido na Universidade de Manchester no Reino Unido por volta de 1940, o protocolo Manchester representa uma peça fundamental nas comunicações entre sistemas, graças aos seus princípios robustos de codificação e sincronização [15].

Antes de entrar na secção relativa aos métodos de codificação e decodificação no ambiente dos microcontroladores, é essencial estabelecer uma base técnica sobre termos fundamentais utilizados nos sistemas de comunicação, como modulação e codificação. Estes conceitos desempenham papéis distintos na teoria da comunicação, sendo a modulação responsável pela manipulação do sinal portador para a transmissão de informações e a codificação focada na estruturação e representação dos dados para a transmissão e subsequente decodificação.

Este tópico concentra-se principalmente nos aspetos de codificação e decodificação dentro do contexto do protocolo Manchester, explorando como estes processos são implementados e otimizados para garantir uma comunicação robusta e eficiente entre dispositivos baseados em microcontroladores.

Para além disso, são examinadas as aplicações práticas do protocolo Manchester em ambiente de microcontroladores, identificando cenários específicos nos quais o protocolo se destaca pela facilitação da comunicação. Esta descrição inclui uma análise detalhada do funcionamento do protocolo Manchester, desde os princípios básicos até à sua implementação em sistemas embebidos.

4.1 Modulação

A modulação é o processo de adicionar a informação a alguma forma de portadora. A portadora, que é um sinal de frequência mais alta, tem a fase, frequência, amplitude ou uma combinação destes elementos variados em função da informação. Esta variação pode ser detetada e recuperada (desmodulada) no lado da receção. Existem várias formas de modulação mas neste contexto, serão abordadas a Modulação de Amplitude (AM), modulação *on-off keying* (uma variação de AM) e a Modulação de Frequência (FM).

- **Modulação de amplitude:**

Na modulação de amplitude, a amplitude da portadora é ajustada de acordo com o sinal da informação. Isso resulta num "*ripple*" da portadora. A desmodulação desse sinal pode ser realizada através de um detetor de envolvente simples que deteta as variações de amplitude como uma resposta de baixa frequência [16]. Na Figura 23 pode-se observar um exemplo de uma forma de onda deste tipo.

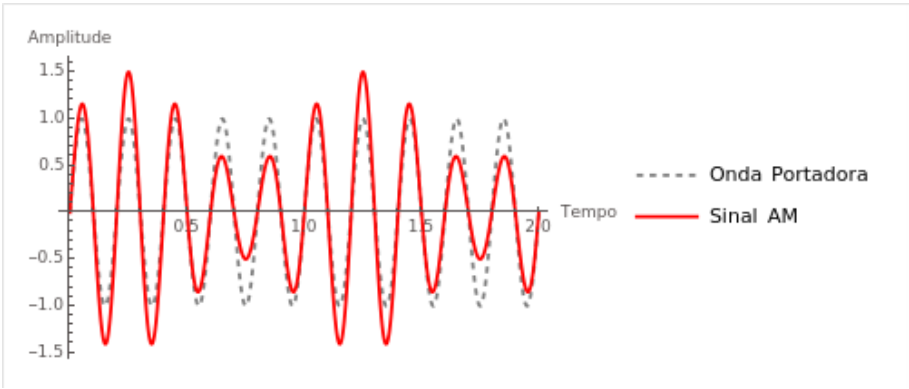


Figura 23 - Formas de onda da modulação de amplitude.

- **Modulação *on-off keying*:**

Esta forma de modulação é uma variação extrema da modulação de amplitude em que há apenas dois estados: a presença da portadora e a ausência da mesma. Este método é especialmente adequado para a transmissão de dados digitais, já que a portadora pode ser transmitida ou não de acordo com a

informação a enviar. A saída é interpretada como alta ou baixa dependendo da presença da portadora [16]. Na Figura 24, é possível observar um sinal com esta modulação.

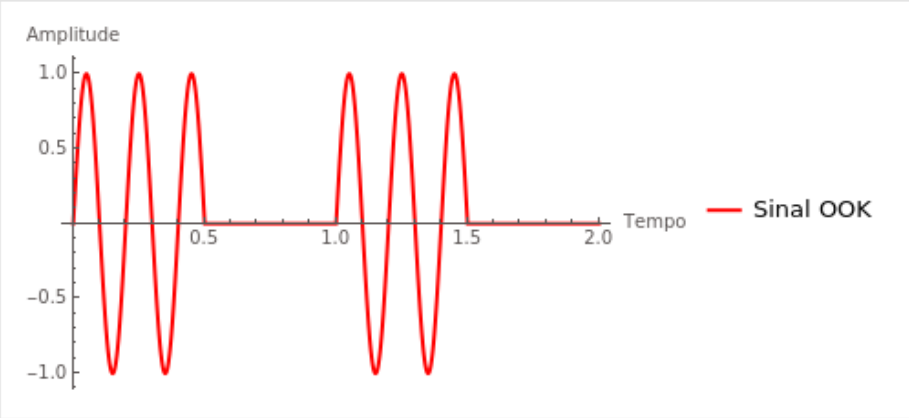


Figura 24 - Forma de onda da modulação on-off keying.

- **Modulação de frequência:**

A modulação de frequência é mais complexa, porém oferece a vantagem de ter uma potência de saída constante independentemente da mensagem transmitida. Nesta técnica a frequência da portadora varia em função do sinal de informação. Isto requer um circuito de desmodulação mais elaborado, geralmente implementado utilizando um *Phase Lock Loop* (PLL) [16]. Na Figura 25, pode-se observar um exemplo de uma mensagem com modulação de frequência.

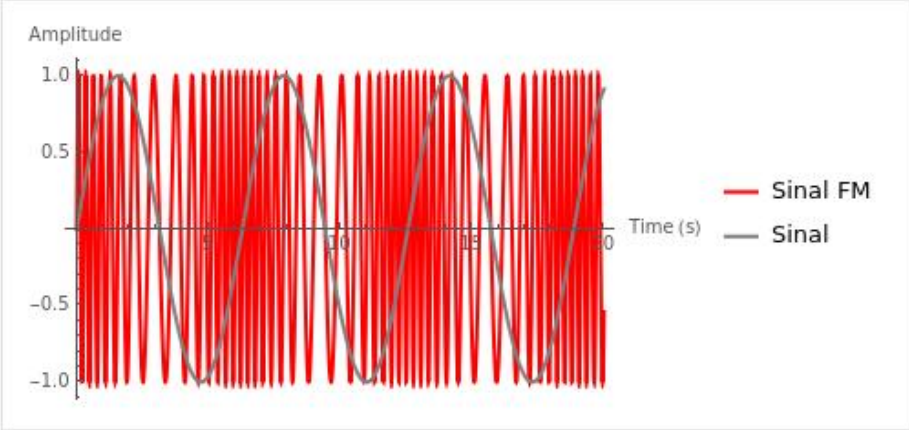


Figura 25 - Formas de onda da modulação de frequência.

- **Modulação por desvio de frequência:**

A relação entre a modulação por desvio de frequência e a modulação de frequência é similar à relação entre a modulação por *on-off keying* e a modulação de amplitude, no sentido de que apenas duas frequências da portadora são utilizadas, cada uma correspondendo a um estado digital. Nesta técnica, os benefícios da modulação de frequência são atingidos com menor complexidade no circuito de desmodulação [16].

4.2 Técnicas de codificação

Assim como na modulação, existe uma grande variedade de técnicas de codificação, cada uma com as suas qualidades e atributos únicos. Estes métodos de codificação podem ser selecionados para otimizar aspetos específicos dentro de um determinado sistema.

Um dos esquemas de codificação mais simples é o *Non-Return to Zero* (NRZ), onde o sinal da mensagem não retorna a zero após cada tempo de *bit*. Por exemplo, uma longa sequência de "1"s resultará num longo período de nível *high* no sinal de mensagem, onde só irá ocorrer uma transição quando houver uma mudança lógica de *bit* (Figura 26).

Implementar a técnica NRZ do lado do codificador é relativamente simples. No entanto é necessário saber o valor preciso da taxa de dados no lado do recetor para a descodificação. Quaisquer discrepâncias nos tempos de relógio de transmissão podem levar a erros de descodificação, exigindo mecanismos de deteção de erro como *checksums* ou *Cyclic-Redundancy-Checks* (CRC). Para além disso os erros provenientes dos canais de comunicação ou das interferências podem passar despercebidos sem verificações adequadas quanto à integridade dos dados transmitidos [16].

A técnica BiPhase introduz alguma complexidade no processo de codificação, mas oferece vantagens, cada *bit* de dados é codificado como uma transição de sinal dentro do período do *bit*. Isto significa que há uma transição de sinal no meio de cada *bit*, garantindo que o recetor possa sincronizar seu relógio com o sinal recebido, tornando a deteção e a interpretação dos dados mais fiáveis.

A codificação Manchester é um dos métodos de codificação de dados mais comuns utilizados atualmente. Semelhante à BiPhase, a codificação Manchester fornece um meio para adicionar o relógio de descodificação de transmissão de dados à mensagem a ser enviada. Para além disso, a codificação Manchester proporciona a vantagem de apresentar sempre um nível médio de corrente contínua de 50%. Tendo implicações positivas no design do circuito do desmodulador.

A codificação Manchester estabelece que há sempre uma transição do sinal no meio do tempo de *bit* dos dados. O que ocorre no início e no final do *bit* depende do estado do *bit*. Um "1" lógico é definido como uma transição do nível lógico baixo para o nível lógico alto e um "0" é uma transição do nível lógico alto para o nível lógico baixo (Figura 26). Uma análise mais detalhada dos métodos para codificar e decodificar dados será mostrada em detalhe nas próximas secções [16].

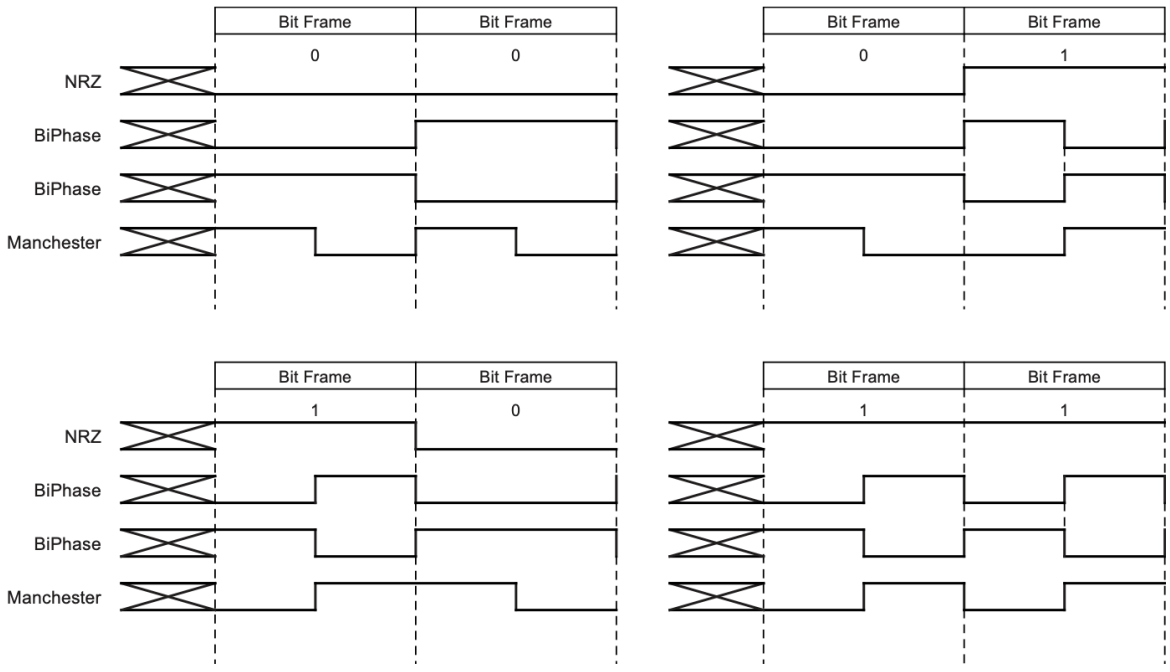


Figura 26 - Exemplos de vários tipos de Codificação [16].

Na Figura 27 pode-se ver que a transição ocorre sempre no meio do período do *bit*.

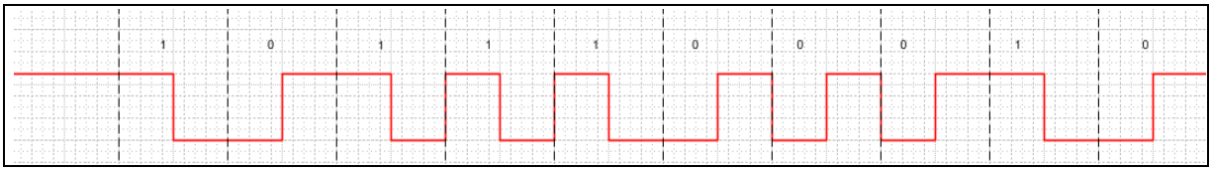


Figura 27 - Codificação no protocolo Manchester [17].

4.3 Métodos de descodificação de mensagens em Manchester

Para a decodificação de uma mensagem codificada em Manchester existem várias abordagens, cada uma com as suas próprias vantagens e desvantagens. Aqui são abordados dois métodos que se destacam por serem as duas opções tipicamente existentes na área de sistemas embebidos. Uma solução em *hardware* e outra por *software*.

Na Figura 28 é apresentado um descodificador utilizando os componentes periféricos de um microcontrolador PIC16F1509 da Microchip, constituindo o primeiro exemplo de uma das possíveis soluções de descodificação com *hardware.* Encontra-se na a*pplication note* AN1470 sobre o tema "*Manchester Decoder Using the CLC and NCO*" [17].

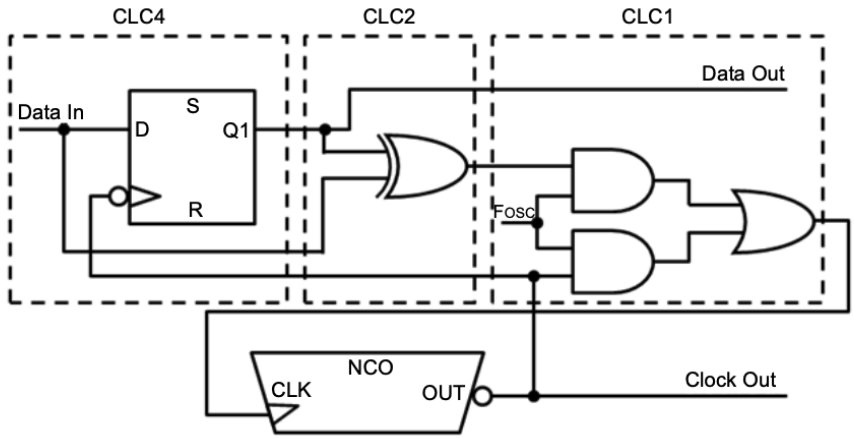


Figura 28 - Hardware de descodificação da mensagem em Manchester [17].

Descrição do Funcionamento:

O microcontrolador PIC16F1509 dispõe de quatro blocos CLC (*Configurable Logic Cell*) para implementar a parte lógica, e um módulo NCO (*Numerically Controlled Oscillator*) para gerar o tempo de *bit* específico. De seguida descreve-se a implementação destes blocos, utilizando os recursos disponíveis no dispositivo.

Existem 3 etapas de descodificação:

- **Etapa 1 – Flip-Flop D (CLC4):**

A primeira etapa serve para a captura de dados utilizando o CLC4, que está configurado como um Flip-Flop D. Esta fase é responsável por capturar os dados Manchester recebidos, fixando-os na transição descendente do sinal de relógio. Os dados capturados permanecem estáveis e podem ser lidos na transição ascendente do relógio, garantindo uma recuperação fiável dos dados. O resultado desta fase torna-se nos dados recuperados que são enviados ao microcontrolador para processamento adicional.

- **Etapa 2 – Porta XOR (CLC2):**

A segunda etapa gere a deteção de transições no meio do *bit* utilizando o CLC2, configurado como uma porta XOR. Como a codificação Manchester garante uma transição no meio de cada *bit*, a porta XOR deteta eficazmente estas transições gerando uma transição ascendente em cada ponto médio do

bit. Esta fase é crucial para manter a sincronização, pois fornece uma referência temporal consistente para o descodificador. O resultado desta fase alimenta a fase final de temporização

- **Etapa 3 – NCO + AND-OR (CLC1):**

A terceira etapa, que é a mais complexa, combina o CLC1 (configurado como uma porta AND-OR) com o NCO para gerar temporização precisa para a amostragem de dados. O módulo NCO gera impulsos de ¾ do tempo de bit enquanto opera no modo de frequência de impulso ativo-baixo. A sua largura de impulso pode ser ajustada utilizando um registo de função especial. Enquanto isso, o CLC1 fornece a fonte de relógio necessária para o NCO e cria impulsos de comprimento fixo a partir dos sinais recebidos da segunda fase. Uma característica importante desta fase é que a saída do NCO realimenta a porta AND-OR, criando um sistema de temporização autossustentável.

A sequência de operação começa com uma configuração inicial onde o sistema produz um único impulso de ¾ do comprimento do bit. Isto coloca o NCO no seu estado ativo-baixo, onde aguarda pelo número configurado de impulsos de relógio. Quando uma fonte de relógio é fornecida, o NCO completa o seu impulso ativo e inicia o próximo ciclo de contagem. Este processo continua para cada bit subsequente, mantendo uma temporização precisa ao longo do processo de descodificação.

A Figura 29 utiliza o exemplo de uma entrada de dados Manchester de 25 kHz. O que traduz num tempo de bit de 40 µs, o que significa que é necessário gerar três quartos do tempo de bit de 30 µs para capturar dados e sincronizar o relógio do NCO com a transição ascendente do relógio extraído.

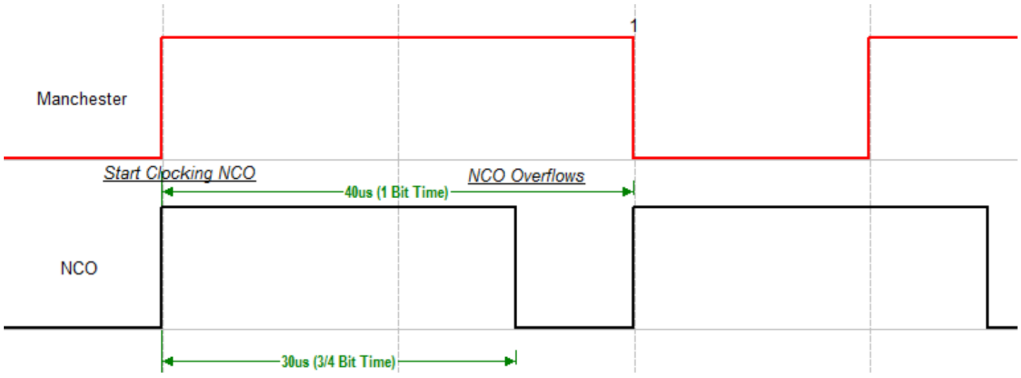


Figura 29 - Processo de descodificação do relógio incluído na mensagem.

Na Figura 30, apresentam-se as formas de onda obtidas nas várias fases de descodificação. Cada uma das fases corresponde ao seguinte sinal:

- Fase 1 - Mensagem recebida em Manchester;
- Fase 2 - Flip-flop D que captura a entrada de informação quando o NCO é ativado;
- Fase 3 - Porta XOR que fornece o tempo inicial ao NCO;
- Fase 4 - Porta AND-OR para fornecer o relógio ao NCO e para garantir que funciona durante três quartos do tempo de bit;
- Fase 5 - NCO para gerar três quartos do tempo de bit, iniciado na transição do meio do bit anterior;

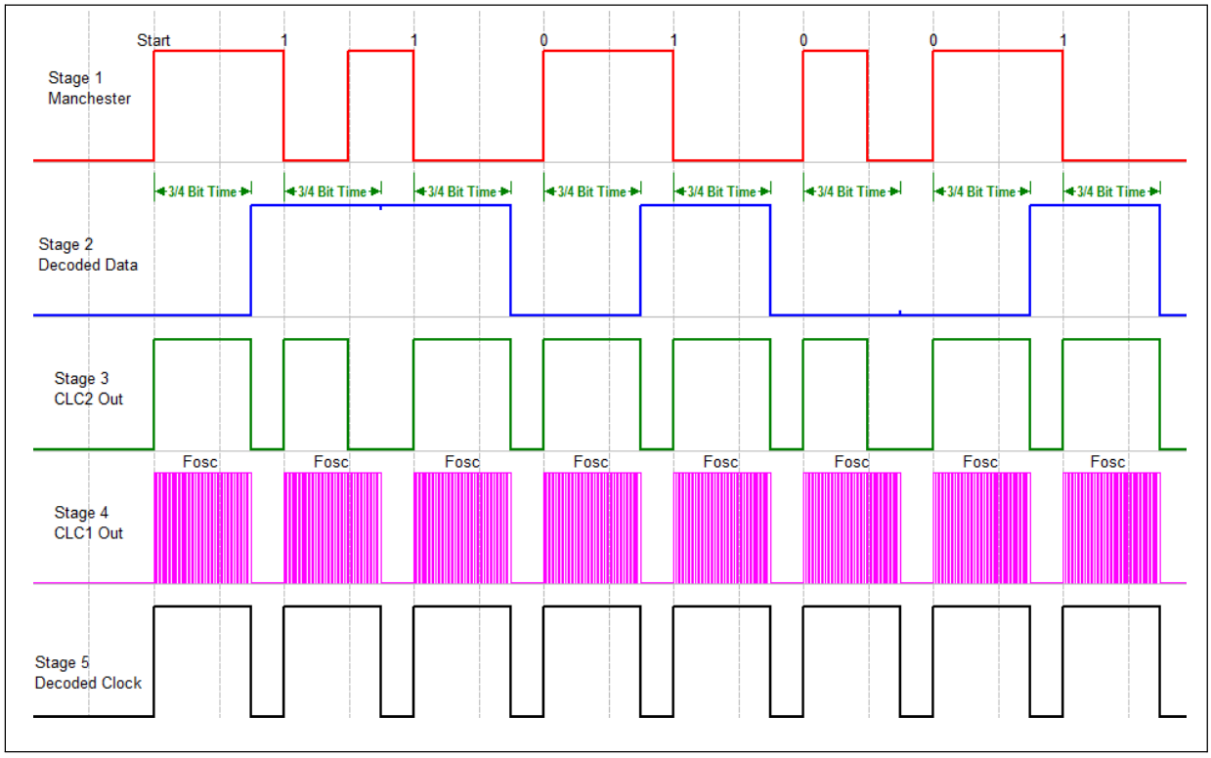


Figura 30 - Processo de descodificada em Manchester (IEEE 802.3) [17].

Esta solução é bastante simples e fiável, pois trata-se de *hardware* configurável que, em contraste com uma solução em código, liberta bastante carga de processamento do microcontrolador. Sendo que todo o processo de descodificação está a ser realizado pela parte periférica do microcontrolador e não está dependente de nenhuma componente de *software*. Permitindo assim, atingir maior velocidade de transmissão do sinal.

Outros métodos recorrem a *software* para realizar a descodificação. Neste caso, utilizam recursos computacionais do processador, o que representa uma desvantagem em comparação com os descodificadores implementados em *hardware*. No entanto, oferecem a vantagem de serem soluções mais universais, podendo ser implementadas na maioria dos microprocessadores disponíveis. No documento "*Application Note Manchester Coding Basics*" da Atmel [8], são discutidas duas formas de fazer a descodificação de sinais Manchester por *software*.

No primeiro método, chamado de "*Timing Based Manchester Decode*" [16], a solução consiste na utilização de dois periféricos, incluídos no microprocessador e com um algoritmo para o processamento do sinal. Neste caso é capturado o tempo entre cada transição do sinal vindo do circuito de desmodulação. O periférico CCP (*Capture/Compare/PWM*) do microcontrolador é ideal, pois gera uma interrupção, permite medições precisas de tempo, e toma decisões baseadas no valor do temporizador.

De seguida faz-se a descrição detalhada de como este método de descodificação funciona, indicando as etapas envolvidas na descodificação da mensagem:

- **Etapa 1 - Sincronização do relógio de dados:**

O descodificador mede o tempo entre as transições do sinal (transições ascendente e descendente). Estas transições ocorrem a intervalos regulares devido à natureza da codificação *Manchester*.

Para sincronizar com o sinal de dados, o descodificador procura um intervalo de tempo igual a 2T, onde T é o tempo correspondente a metade do período de um *bit*. Uma vez este intervalo encontrado, o descodificador está sincronizado com o relógio de dados e pode começar a reconstruir os *bits*.

- **Etapa 2 - Identificação dos *bits*:**

Após a sincronização, o descodificador observa o tempo entre cada transição para determinar o valor de cada *bit*.

Um intervalo de tempo igual a T indica que o próximo *bit* é igual ao *bit* atual, enquanto um intervalo de tempo igual a 2T indica que o próximo *bit* é o oposto do *bit* atual.

- **Etapa 3 - Janela de tolerância:**

Como o valor do temporizador pode não ser exatamente T ou 2T devido a variações no sinal, o descodificador usa uma janela de tolerância para permitir pequenas diferenças nos tempos medidos, garantindo que a informação seja corretamente recuperada.

Etapas da implementação da descodificação:

- **Etapa 1 - Configurar o temporizador e o módulo de captura de entrada:**

O temporizador é configurado para contar continuamente, e o módulo CCP é configurado para detetar transições no sinal e gerar uma interrupção em cada transição (ascendente ou descendente).

- **Etapa 2 - Capturar a primeira transição:**

Quando o descodificador começa a operar, este captura a primeira transição do sinal e descarta-a. Esta transição serve apenas para iniciar a medição do tempo.

- **Etapa 3 - Capturar a próxima transição e sincronizar com o relógio de dados:**

A segunda transição é capturada e o tempo entre esta transição e a anterior é medido. Se o intervalo de tempo for 2T, o descodificador está agora sincronizado com o relógio de dados.

- **Etapa 4 - Ler o nível lógico do pino de entrada:**

Após a sincronização, o descodificador lê o nível lógico do sinal de entrada (1 ou 0) e armazena-o como o valor do *bit* atual.

- **Etapa 5 - Capturar a Próxima Transição e Determinar o Próximo Bit:**

  a. Comparar o tempo decorrido com T:

  - Se o tempo entre as transições for igual a T:

  - O próximo *bit* é igual ao *bit* atual.

  - Captura-se outra transição para garantir que o tempo também seja T. Caso contrário, é gerado um erro.

  b. Se o tempo decorrido for igual a 2T:

  - O próximo *bit* é o oposto do *bit* atual.

  c. Se o tempo não for igual a T ou 2T:

  - O descodificador gera um erro, pois o tempo medido não corresponde aos valores esperados.

- **Etapa 6 - Armazenar o próximo b*it*:**

O próximo *bit* é armazenado num buffer para posterior processamento.

- **Etapa 7 - Repetir o processo até decodificar todos os *bits*:**

O descodificador verifica se o número desejado de *bits* foi recuperado. Se sim, o processo é concluído. Caso contrário, o *bit* atual é atualizado para o próximo *bit*, e o processo continua a partir da etapa 5 até todos os *bits* da mensagem serem descodificados.

Apresenta-se na Figura 31 - Exemplo do método de descodificação "*Timming Based Decode*"., o primeiro processo de descodificação descrito anteriormente numa forma gráfica:

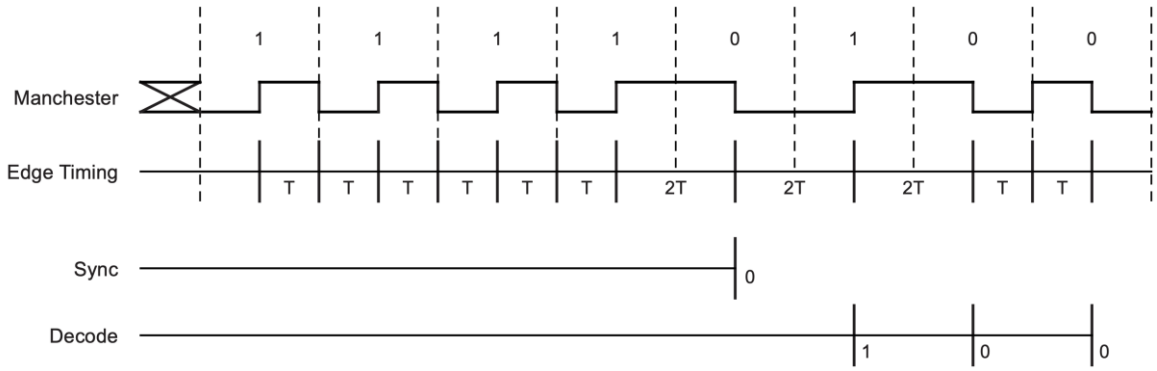


Figura 31 - Exemplo do método de descodificação "Timming Based Decode" [16].

No segundo método, chamado de "*Sampling Based Manchester Decode*" [16], não é necessário capturar ou reconhecer as transições do sinal. Em vez disso o descodificador simplesmente amostra o estado do pino de entrada a uma taxa muito maior do que a taxa de transmissão de dados da mensagem. Esse método requer mais memória, pois os dados amostrados são armazenados num buffer, mas permite que o microcontrolador realize outras tarefas sem corromper a descodificação. A amostragem é feita utilizando um temporizador que interrompe o microcontrolador a intervalos regulares.

De seguida são descritas as várias etapas do funcionamento:

- **Etapa 1 - Amostragem regular do sinal:**

O sinal é amostrado a uma taxa S, que é muito maior do que a taxa de dados da mensagem. Um temporizador é configurado para interromper o microcontrolador a intervalos regulares, e o estado do pino de entrada (1 ou 0) é armazenado em um buffer.

- **Etapa 2 - Processamento posterior do *buffer* de amostras:**

Depois de capturar um número suficiente de amostras o *buffer* é processado. O número de valores consecutivos (uns ou zeros) é contado para identificar as transições e determinar os *bits* da mensagem.

- **Etapa 3 - Decisão com base na amostragem:**

O número de amostras consecutivas é comparado com metade da taxa de amostragem (S/2) para determinar se houve uma mudança de *bit*. Isso permite identificar corretamente os bits, mesmo com variações no sinal.

De seguida são descritas as etapas de implementação deste método:

- **Etapa 1 - Configurar o temporizador para interromper a cada intervalo 2T / S:**

O temporizador é configurado para gerar uma interrupção em intervalos regulares de 2T / S, onde T é metade do tempo do *bit*, e S é a taxa de amostragem, que é muito maior do que a taxa de dados.

- **Etapa 2 – *Interrupt Service Routine* para Armazenar o Estado do Pino de Entrada:**

Na rotina de *Interrupt service routine* (ISR), o estado do pino de entrada (1 ou 0) é lido e armazenado num *buffer* de amostras. Isto é feito continuamente enquanto o temporizador gera interrupções.

- **Etapa 3 - Repetir a etapa 2 para capturar o número desejado de amostras:**

Continuar a amostragem do sinal até que seja capturado o número necessário de amostras, correspondente ao número de *bits* da mensagem.

- **Etapa 4 - Processar o *buffer* de amostras e contar uns ou zeros consecutivos:**

Uma vez que o *buffer* esteja cheio, iniciar o processamento. Contar o número de valores consecutivos iguais (uns ou zeros) à medida que se percorre o *buffer*.

- **Etapa 5 - Quando ocorre uma mudança de valor lógico:**

Quando o valor lógico no *buffer* mudar de 1 para 0 ou de 0 para 1:

a. Verificar se o número de valores consecutivos (uns ou zeros) é maior ou igual a S/2.

b. Se assim for, prosseguir para a próxima etapa. Caso contrário, reiniciar o contador e voltar à etapa 4 para continuar a contar.

- **Etapa 6 - Definir o *bit* atual como o valor lógico no b*uffer*:**

O *bit* atual é determinado pelo valor lógico que o *buffer* está a registar no momento.

- **Etapa 7 - Contar até a próxima mudança de valor lógico:**

Reiniciar o contador e começar a contar até à próxima mudança de valor lógico (de 1 para 0 ou de 0 para 1):

a. Comparar o número de amostras consecutivas com S/2.

b. Se o número de amostras consecutivas for menor que S/2:

- Reiniciar o contador e continuar até a próxima mudança de valor lógico.
- Certificar-se de que a próxima contagem também seja menor que S/2.
- O próximo *bit* será igual ao *bit* atual.
- Armazenar o próximo *bit* no *buffer* de dados.

c. Se o número de amostras consecutivas for maior ou igual a S/2:

- O próximo *bit* será o oposto do *bit* atual.
- Armazenar o próximo *bit* no *buffer* de dados.

d. Se o número de amostras não atender a nenhum dos critérios:

- Um erro é gerado, indicando que a descodificação não foi bem-sucedida.

- **Etapa 8 - Repetir até processar todo o *buffer*:**

Continuar a processar o *buffer* de amostras até que todos os dados capturados tenham sido processados e todos os *bits* da mensagem tenham sido recuperados.

- **Etapa 9 - Sair para processamento adicional dos dados:**

Após descodificar todos os *bits* da mensagem, o descodificador pode sair do ciclo de amostragem e prosseguir para o processamento adicional dos dados, conforme necessário.

Apresenta-se na Figura 32 numa forma gráfica o segundo processo de descodificação descrito anteriormente:

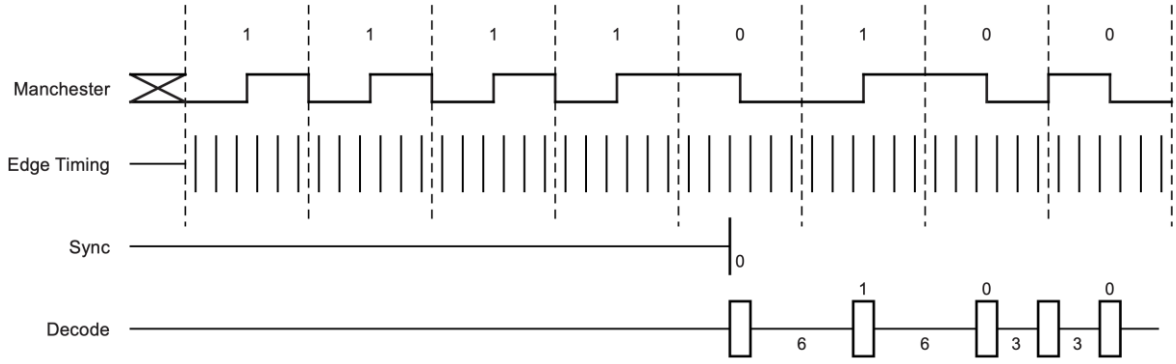


Figura 32 - Exemplo do método de descodificação Sampling Based Decode [16]

Na Tabela 2 é apresentada uma comparação simplificada de cada um dos métodos de descodificação.

Tabela 2 - Comparação de características entre métodos de descodificação.

Características	**Microchip - "*Manchester Decoder Using the CLC and NCO*"**	**Atmel - *"Timing Based Decoding"***	**Atmel - *"Sampling Based Decoding"***
Complexidade de Implementação	Alta (configuração de periféricos)	Moderada (uso de temporizador)	Baixa (simples temporizador e *buffer*)
Uso de unidade de processamento	Muito baixa	Baixa	Alta (*buffer* de amostras)
Precisão Temporal	Muito alta	Alta	Moderada
Dependência de *hardware*	Alta	Baixa	Baixa
Flexibilidade	Baixa	Moderada	Alta

A escolha entre os diferentes métodos de implementação do protocolo *Manchester* deve ser cuidadosamente avaliada com base em várias considerações específicas do projeto. Os principais fatores a serem analisados incluem:

Escolha do microcontrolador: é fundamental selecionar um microcontrolador que suporte as necessidades do projeto, considerando se possui a capacidade de codificação e descodificação *Manchester* em *hardware* ou se dependerá de uma solução em *software*. Microcontroladores com suporte nativo do protocolo oferecem maior eficiência e simplicidade de implementação.

Escolha do *hardware* e/ou método de transmissão e receção de dados em Manchester: a escolha entre uma solução em *hardware* ou *software* para a codificação Manchester depende diretamente do nível de precisão e confiabilidade exigidos pelo projeto. Implementações em *hardware* garantem menor latência e maior estabilidade, enquanto soluções em *software* podem ser utilizadas em casos onde o custo e a flexibilidade são mais importantes.

Latência e tempo de resposta: nas aplicações que exigem baixa latência e tempo de resposta preciso, como sistemas de controlo em tempo real, uma implementação em *hardware* seria a melhor decisão. A escolha do método de transmissão também influencia diretamente esses parâmetros, especialmente em ambientes ruidosos ou com muitos dispositivos.

Consumo de energia: para dispositivos que funcionam com baterias, o consumo de energia é um ponto muito importante. O consumo de energia vai variar conforme o método escolhido, sejam otimizações em *software* ou escolha de *hardware*, e também dependerá do uso de modos de baixo consumo que o microcontrolador possa disponibilizar.

Custo e complexidade de implementação: projetos com restrições orçamentais ou de espaço físico podem exigir soluções em *software*, desde que as exigências de desempenho não sejam comprometidas. A complexidade de implementação também deve ser considerada, uma vez que soluções em *hardware* podem exigir componentes adicionais e um projeto de circuitos mais elaborado.

Soluções em *hardware*: são recomendadas para aplicações críticas, como sistemas automotivos, automação industrial e controlo de motores, onde a robustez e a fiabilidade são essenciais. A codificação Manchester em *hardware* proporciona uma comunicação mais estável e eficiente, especialmente em ambientes com taxas de transmissão elevadas dos ou interferência eletromagnética significativa.

Soluções em s*oftware*: adequadas para dispositivos IoT e sistemas de sensores, onde a flexibilidade e o custo são prioridades. Em projetos onde a carga de processamento do microcontrolador é baixa e

não há exigência de uma taxa de transmissão elevada, a codificação e descodificação via *software* pode ser suficiente.

A escolha do método de implementação depende diretamente das especificações e requisitos da aplicação. Para sistemas que exigem alto desempenho e precisão, a solução em *hardware* é geralmente preferível. No entanto, para aplicações onde o custo, a flexibilidade e o consumo de energia são os principais fatores, uma abordagem em *software* pode ser mais adequada.

Mesmo que a implementação em *hardware* seja, de forma geral, a mais recomendada, a escolha final deve sempre levar em conta o tipo de aplicação e as características do microcontrolador disponível para garantir que todas as necessidades do projeto são atendidas de forma eficiente e económica.

4.4 Deteção de erros

A deteção de erros é uma parte fundamental na transmissão de informação, de forma a garantir a integridade dos dados transmitidos. A implementação de técnicas avançadas de deteção de erros, como o CRC, desempenha um papel vital na fidelidade da transmissão de dados.

O CRC é um método eficaz para detetar erros em tramas de dados transmitidos, utilizando polinómios geradores específicos. Aplicado na verificação da integridade dos *bits* transmitidos e garantir que não ocorreram erros durante a transmissão [18].

O processo de CRC envolve a geração de um valor de verificação (*checksum*) a partir dos dados transmitidos. Este valor é anexado aos dados e enviado juntamente com a mensagem. No recetor, um cálculo semelhante é realizado com os dados recebidos, e o resultado é comparado com o valor de *checksum* recebido. Se houver uma discrepância, isso indica a presença de erros na transmissão.

No contexto específico do protocolo *Manchester*, onde as transições frequentes são utilizadas para codificar os *bits*, a precisão na deteção de erros é essencial. As transições regulares podem ser afetadas por interferências ou ruídos e a utilização de um CRC oferece uma camada adicional de segurança ao verificar a integridade dos dados.

O CRC destaca-se pela sua capacidade de deteção de erros em vários *bits*, proporcionando uma maior cobertura e robustez em comparação com métodos mais simples, como a paridade. A sua eficácia na deteção de erros torna-o uma escolha preferencial em ambientes onde a integridade dos dados é crítica.

4.5 Caso de estudo

Nesta secção é apresentado um caso de estudo detalhado de como o protocolo Manchester é utilizado em aplicações de comunicação em sistemas embebidos.

O protocolo de transmissão IR Philips RC5 utiliza a codificação Manchester para facilitar a comunicação fiável entre telecomandos e dispositivos eletrónicos. Esta secção analisa os aspetos técnicos do protocolo RC5, com foco no esquema de codificação, na estrutura da mensagem e na eficiência de operação [19].

Especificações do protocolo

O protocolo RC5 opera com uma frequência de portadora de 36 kHz, utilizando a codificação Manchester para representar estados lógicos. Neste esquema de codificação cada *bit* é transmitido como uma transição no sinal, proporcionando um mecanismo de sincronização automática que auxilia na sincronização e deteção de erros.

Para a transmissão de *bits* lógicos:

- Um '0' lógico é representado por um conjunto de impulsos (*burst*) de 889 µs, seguido de um intervalo (*space*) de 889µs.
- Em contrapartida, um '1' lógico é codificado como um intervalo de 889 µs, seguido de um impulso de 889 µs.

Esta estrutura resulta numa duração de transmissão uniforme de 1,778 ms para ambos os estados lógicos. A relação impulso/intervalo é aproximadamente 1:3 ou 1:4, o que ajuda a reduzir o consumo de energia, uma consideração importante em telecomandos alimentados a bateria.

Estrutura da trama

Quando um utilizador pressiona uma tecla no telecomando, este transmite uma mensagem estruturada composta por 14 *bits*. A estrutura inclui:

- Dois *bits* de início (S1 e S2): ambos definidos como '1' lógico para indicar o início da transmissão. A presença de um intervalo inicial em S1 introduz um atraso, permitindo que o recetor detete o início da mensagem após 889µs.
- *Bit* de alternância (T): Este *bit* é invertido cada vez que uma tecla é pressionada e libertada. Serve para distinguir entre pressões repetidas e a manutenção de uma tecla pressionada.
- Endereço de 5 *bits*: esta secção identifica o dispositivo recetor específico. É enviada na ordem do bit mais significativo *Most Significant Bit* (MSB) primeiro.
- Comando de 6 *bits*: esta parte codifica o comando específico a ser executado, também na ordem MSB primeiro.

A transmissão dos *bits* de início e alternância demora um total de 5,334 ms. A transmissão do endereço ocupa 8,89 ms, seguida por 10,668 ms para o comando, culminando num tempo total de transmissão de 24,892 ms para a mensagem completa. A Figura 33 ilustra um exemplo de uma estrutura que utiliza o protocolo de transmissão IR Philips RC5. Nesta figura, é apresentado um endereço de 05h (00101b) e um comando de 35h (110101b), detalhando a disposição dos *bits*.

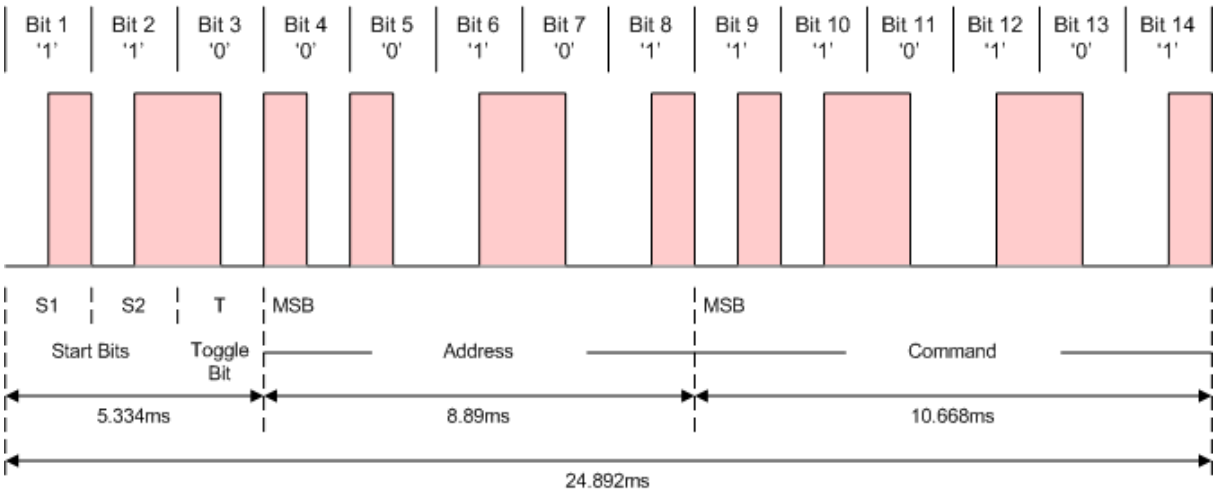


Figura 33 - Exemplo de um comando por infravermelho utilizando o protocolo RC5 [19].

Mecanismo de repetição automática

Uma característica importante do protocolo RC5 é a sua capacidade de repetição automática. Se o utilizador mantiver a tecla pressionada, o telecomando enviará continuamente a mesma mensagem a cada 114 ms. Durante estas transmissões repetidas, o *bit* de alternância mantém-se inalterado, permitindo ao dispositivo recetor interpretar o sinal corretamente. Este *design* facilita a utilização, permitindo ações como o ajuste do volume ou a mudança de canal sem a necessidade de pressionar repetidamente os botões.

Funcionalidade do dispositivo recetor

Após a transmissão, o dispositivo recetor utiliza um desmodulador para captar os sinais IR. O desmodulador utiliza o formato de codificação Manchester para identificar transições que correspondem aos estados lógicos. O *software* do dispositivo está programado para interpretar corretamente os *bits* de início, o *bit* de alternância, o endereço e o comando. Mecanismos de verificação de erros inerentes à codificação Manchester permitem ao recetor detetar e mitigar problemas resultantes de interferências ou perda de sinal, garantindo uma comunicação fiável.

Conclusão

O protocolo de transmissão IR Philips RC5 exemplifica a utilização eficiente da codificação Manchester em aplicações de telecomandos. A estrutura da trama da mensagem, combina técnicas de transmissão energeticamente eficientes e capacidades robustas de deteção de erros.

5. Novo protocolo de comunicações do BMS IR

O desenvolvimento de um novo protocolo entre o *Gateway* e o Escravo do BMS IR visa aumentar a taxa de transmissão de dados do seu sistema de comunicações. Esta seção irá detalhar o processo de desenvolvimento realizado para a implementação do novo protocolo.

5.1 Objetivos

Os objetivos deste trabalho incluem:

- Duplicar a taxa de transmissão do protocolo em relação ao anteriormente utilizado;
- Identificar as vantagens que a nova solução trará para o BMS IR como a fiabilidade na transmissão de dados;
- Encontrar um protocolo compatível com o *hardware* atualmente implementado no BMS IR, explorando as possibilidades de alteração de *hardware*, se possível.

5.2 Protocolo utilizado

O protocolo atualmente utilizado recorre à comunicação série assíncrona conhecida por UART (*Universal Asynchronous Transmitter and Receiver*). Neste caso utiliza-se a linha RX do módulo periférico de USART do microcontrolador para fazer a descodificação do sinal vinda do recetor IR. Quanto à codificação, esta realiza-se via *software* com uma modulação *on-off keying* utilizando o periférico *Pulse Width Modulation* (PWM) com um sinal de portadora de 38 kHz para ser emitida pelo emissor IR conectado a uma porta de saída do microcontrolador.

Na Figura 35 pode-se observar o esquema elétrico do microcontrolador do Escravo com os respetivos pinos mencionados anteriormente.

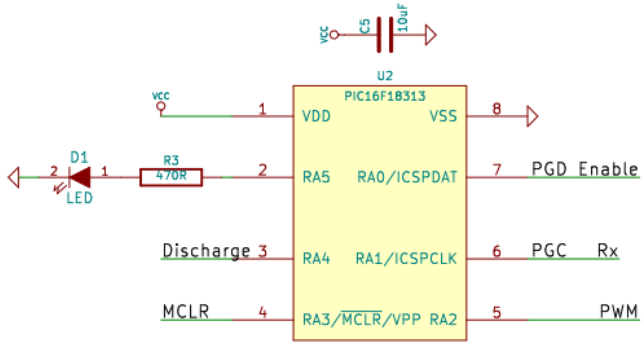


Figura 34 - Esquema elétrico do Microcontrolador do Escravo.

Na Figura 35 é possível observar o pino PWM do microcontrolador conectado à *gate* do transístor para modular o sinal.

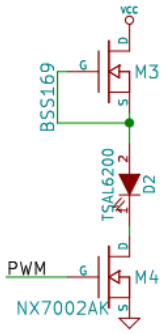


Figura 35 - Emissor infravermelho do Escravo.

E na Figura 36 pode-se observar a saída do recetor de infravermelho conectado ao pino Rx do microcontrolador.

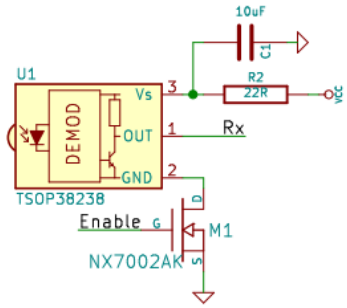


Figura 36 - Recetor infravermelho do Escravo.

De forma a minimizar alterações de *hardware*, sugeriu-se a implementação do protocolo Manchester. Ao analisar o *hardware* instalado no sistema BMS IR, verificou-se que grande parte do sistema era compatível com a alteração.

5.3 Formato das tramas

Nesta seção, é detalhado o formato das tramas para o protocolo de comunicações proposto. Na Tabela 3, apresenta-se a estrutura da mensagem:

Tabela 3 - Formato da trama para o novo protocolo.

Elemento	Descrição
Preamble	Sequência de *bits* que marca o início da transmissão.
Start Bit	*Bit* que indica o início da mensagem.
Mensagem	Dados ou informações a serem transmitidos.
CRC	*Cyclic Redundancy Check*, valor de verificação para garantir a integridade dos dados.

Este formato permite que num ambiente fechado, como uma caixa com baterias de tração de um veículo elétrico, seja garantida a sincronização, a identificação do início da trama e a integridade dos dados transmitidos.

5.4 Processo de codificação e descodificação da mensagem

Nesta seção descrevem-se os benefícios e condicionantes dos métodos estudados para o processo de codificação e descodificação da mensagem. A Tabela 4 apresenta uma comparação entre diferentes métodos de descodificação.

Tabela 4 - Comparação de diferentes métodos de descodificação.

Método de descodificação	**Microchip – "*Decoder using CLC and NCO*"**	**Atmel – "*Timing Based Decoder*"**	**Atmel – "*Sample Based Decoder*"**
Benefícios	- Fácil configuração e programação do microprocessador. - Taxas elevadas de transmissão e não usa a unidade de processamento do microprocessador.	- Fácil implementação com o *hardware* disponível. - Permite maior flexibilidade por ser implementado por *software*.	- Os mesmos benefícios que no "*Timing Based Decoder*".
Condicionantes	- É necessário que o microprocessador contenha todos os periféricos necessários para a implementação desta solução.	- Exigência moderada da parte da unidade de processamento do microcontrolador.	- É necessária mais memória. - Maior exigência de processamento.

Com base nas características dos microcontroladores e nos componentes do sistema do BMS IR, foi escolhido o método de "*Timing Based Decoder*" apresentado pela Atmel.

5.5 Desenvolvimento do algoritmo de descodificação

O algoritmo desenvolvido para codificar e descodificar a mensagem que está a ser transmitida foi desenvolvido de uma forma simples e facilmente ajustável para diferentes microcontroladores. Foram criadas duas versões do algoritmo: com e sem portadora de 38 kHz, que é colocada na codificação da mensagem Manchester. Permitindo assim, enviar os dados via infravermelhos ou utilizando uma ligação física.

Este algoritmo realiza a descodificação de uma mensagem composta por 15 *bits* para sinalizar o início da transmissão da mensagem, 1 *bit* para marcar o início da mensagem, 7 *bytes* para a mensagem e 2 *bytes* para o CRC.

O algoritmo de codificação da mensagem com codificação Manchester, consiste num ciclo que envia a mensagem e cujo fluxograma pode ser observado na Figura 37.

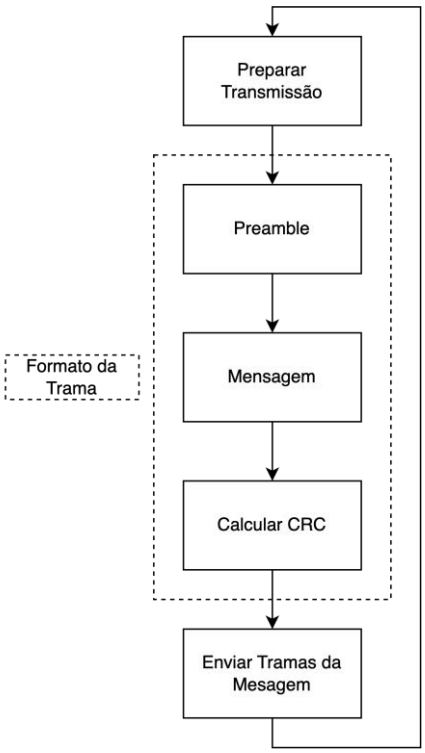


Figura 37 - Fluxograma da codificação das mensagens enviadas em Manchester.

Como referido anteriormente este algoritmo permite enviar os sinais com ou sem portadora, o qual não influência diretamente na estrutura do código. Apenas é feita uma seleção do método de envio na parte do codificador.

O código utilizado para o codificador pode ser consultado no Anexo B. Nesse anexo, pode ver-se observar a função "*CCP1_LoadDutyValue();*" que permite a modulação do sinal a 38 kHz, e as rotinas "*Encoder_SetLow();*" e "*Encoder_SetHigh();*" para o envio do sinal sem portadora.

Ao contrário do que a Atmel sugere, neste caso, optou-se por capturar os períodos entre as transições da mensagem para depois fazer a descodificação ao invés da leitura do sinal de entrada.

Ao nível da organização do algoritmo, este é dividido em três fases:

- Captura de transições e armazenamento das mesmas num *buffer*;
- Descodificação do conteúdo do *buffer* através da utilização de uma máquina de estados;
- Rotina que verifica se a mensagem foi recebida corretamente.

Na primeira fase, são utilizados os periféricos CCP (*Capture/Compare/PWM*) e o TMR1 (*Timer 1*) para capturar uma transição e determinar o tempo decorrido entre transições. Estes valores são armazenados numa variável temporária com a capacidade de guardar transições suficientes para uma trama de 88*bits* (*Preamble* + *Start bit* + Mensagem + CRC) que podem resultar em cerca de 176 transições.

Todas as vezes que existe uma transição, aciona-se uma interrupção, o valor de registo do *Timer 1* é armazenado no *buffer* e após isso, o *Timer 1* reinicia a contagem. Na Figura 38, apresenta-se uma ilustração que indica o momento em que este conjunto de eventos ocorre utilizando o símbolo de uma estrela.

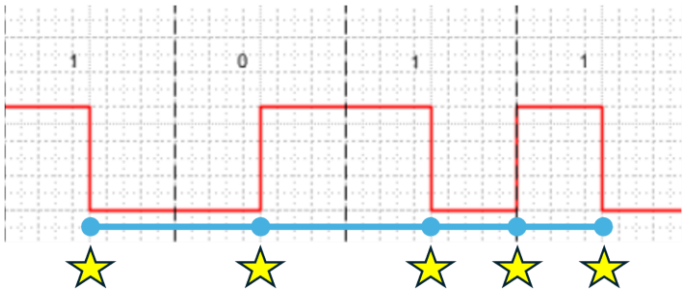


Figura 38 - Momento da captura do tempo de transição entre flancos.

O código desta rotina é apresentado no Anexo D.

Quando o *buffer* se encontra cheio, dá-se início ao processamento da informação recebida. Esta informação vai ser validada pelo algoritmo descodificador de mensagens em Manchester e, se estiver completa, utiliza-se uma rotina de deteção de erros para verificar se a mensagem não foi adulterada. Este processo pode ser visualizado no fluxograma da Figura 39.

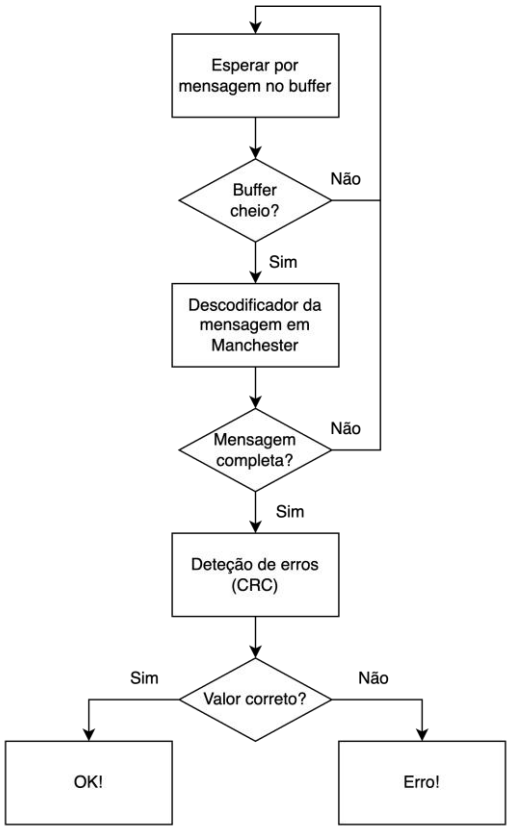


Figura 39 - Fluxograma da descodificação da mensagem recebida em Manchester.

No Anexo C apresenta-se o código desta parte do algoritmo.

Para a descodificação da mensagem, utiliza-se uma máquina de estados capaz de interpretar os valores obtidos no *buffer*. Estes valores serão tratados e validados pelo algoritmo um de cada vez. O fluxograma da máquina de estados é apresentado na Figura 40.

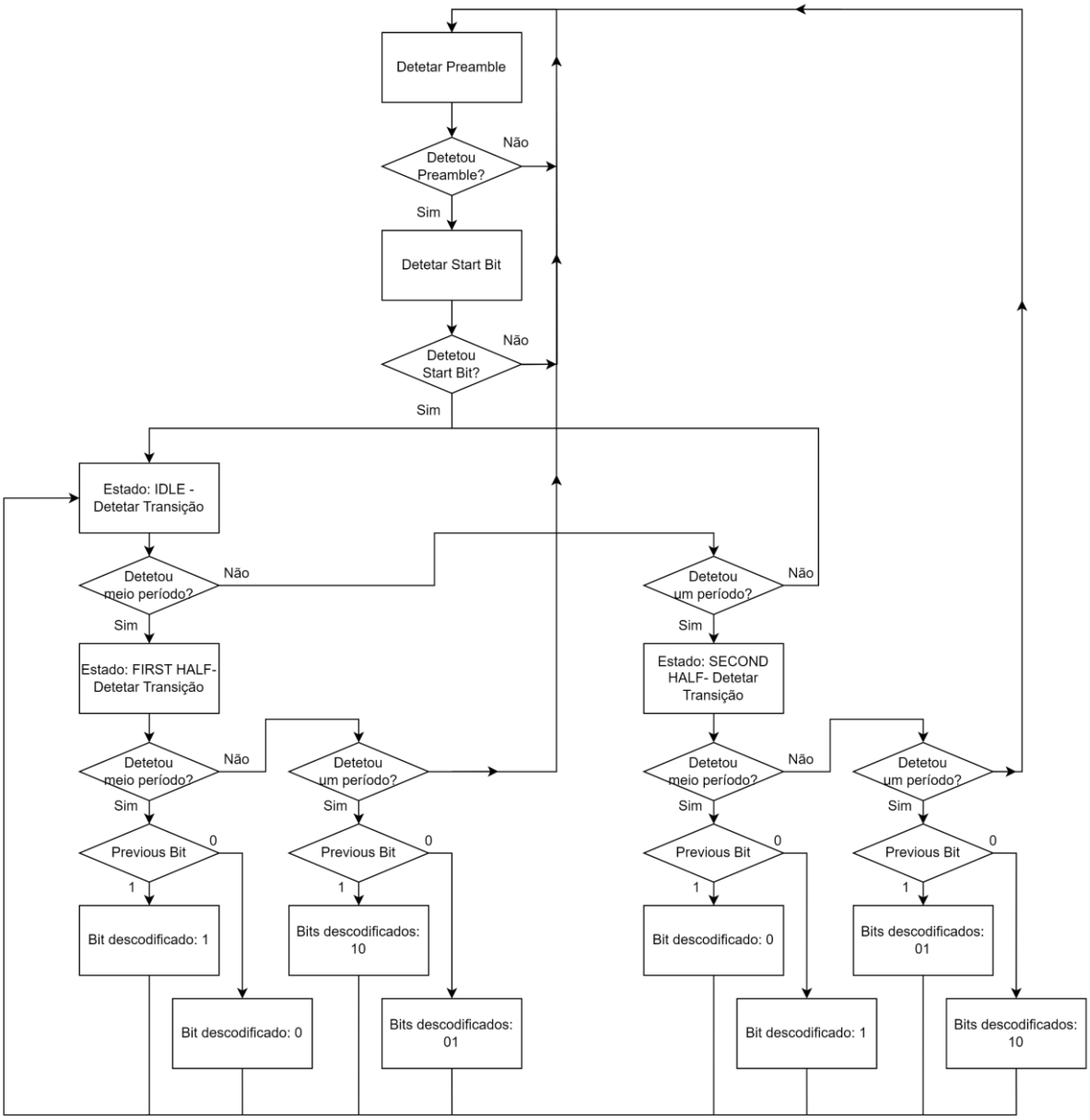


Figura 40 - Fluxograma da máquina de estado para descodificar a mensagem recebida em Manchester.

Um fator muito importante na validação do valor do período de transição é este poder ser ajustado conforme necessário. Considerando um aumento da velocidade da transmissão de dados, é necessário reduzir o valor da janela de validação para permitir a correta interpretação da mensagem.

Os códigos da máquina de estados do descodificador, com as suas funções auxiliares, apresentam-se no Anexo C.

6. Testes e resultados

Criou-se um ambiente controlado de testes baseado nas características e condições de funcionamento das comunicações por infravermelhos no BMS IR.

Foram escolhidas duas placas de desenvolvimento da Microchip, cujos microcontroladores contêm os mesmos periféricos que os do *Gateway* e do Escravo. Estas placas oferecem muitas vantagens a nível de desenvolvimento, por terem um programador e um *debuger* embutido, permitindo assim realizar prototipagens rápidas.

Como emissor utilizou-se a placa de desenvolvimento com o PIC18F16Q20 [20] e para o recetor foi escolhida a placa com o PIC16F18446 [21]. Estas placas estão representadas com um modelo tridimensional na Figura 41.

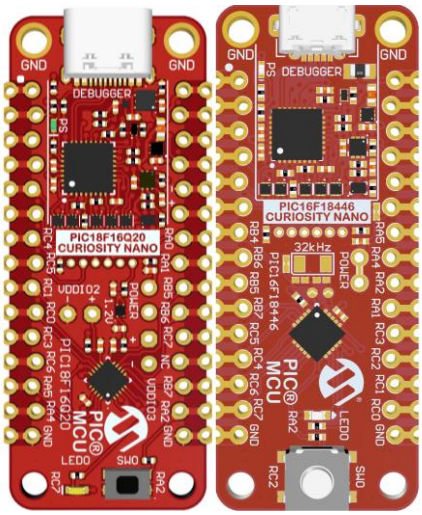


Figura 41 - Placas de desenvolvimento da Microchip (Microchip Technology Inc.).

A montagem do circuito da bancada de testes teve duas versões para o teste e implementação do protocolo de comunicações. Numa primeira fase foi utilizado um fio como meio de transmissão das mensagens entre os dois microcontroladores. Na segunda fase utilizaram-se os componentes infravermelhos para transmitir os dados.

Na Figura 42 apresenta-se uma fotografia da bancada de testes onde foi feito o desenvolvimento do novo protocolo de comunicações.

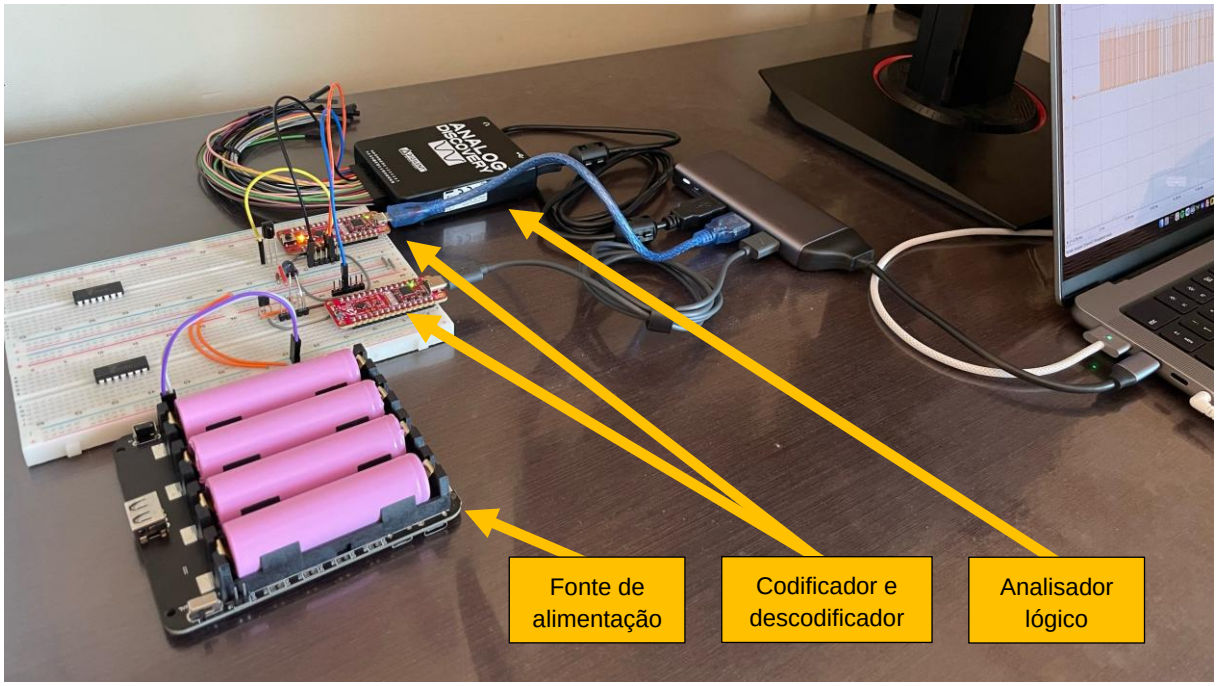


Figura 42 - Bancada de Desenvolvimento com os dois meios de transmissão.

Em termos do esquema elétrico da bancada de desenvolvimento, a primeira versão de testes foi realizada com o circuito da Figura 43.

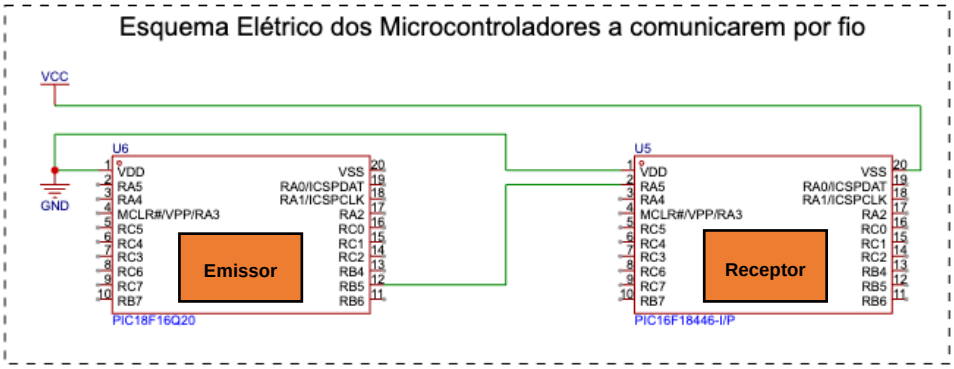


Figura 43 - Esquema elétrico do circuito da bancada de testes com o meio de transmissão por fio [recorte do Anexo A].

Esta versão consiste apenas numa ligação entre o pino RB5 do PIC18F16Q20 que funcionou com emissor e o pino RA5 do PIC16F446 que funcionou como recetor. A alimentação do circuito era feita através do computador, pelo regulador de tensão da placa da Microchip.

A segunda versão foi feita com o circuito representado na Figura 44.

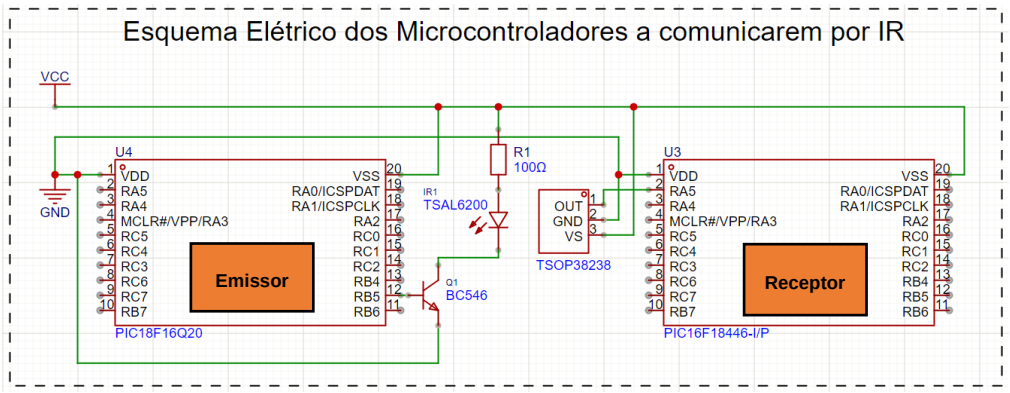


Figura 44 - Esquema elétrico do circuito da bancada de testes com o meio de transmissão por IR [recorte do Anexo A].

Nesta versão foram implementados os componentes necessários para permitir a transmissão de informação pelo ar a partir de infravermelhos na banda de frequência de 38 kHz. O PIC18F16Q20 tem um transístor ligado ao pino RB5 do microcontrolador para controlar o emissor e do lado do PIC16F18446 um recetor ligado ao pino RA5. A alimentação do circuito é realizada por uma fonte externa de 3.2 V para simular a bateria de lítio.

Para a obtenção dos dados apresentados ao longo do capítulo utilizou-se uma ferramenta de análise de ondas lógicas. Esta ferramenta chama-se *Analog Discovery*.

Estrutura do teste

Todos os testes foram realizados com a mesma trama, composta pelos componentes apresentados na Tabela 5.

Tabela 5 - Formato da trama da mensagem de teste.

Preamble	***Start Bit***	**Mensagem**	**CRC**
15 *bits*	1 *bit*	7 *bytes*	2 *bytes*

Os componentes da mensagem podem ser observados na forma de onda gerada pelo codificador na Figura 45.

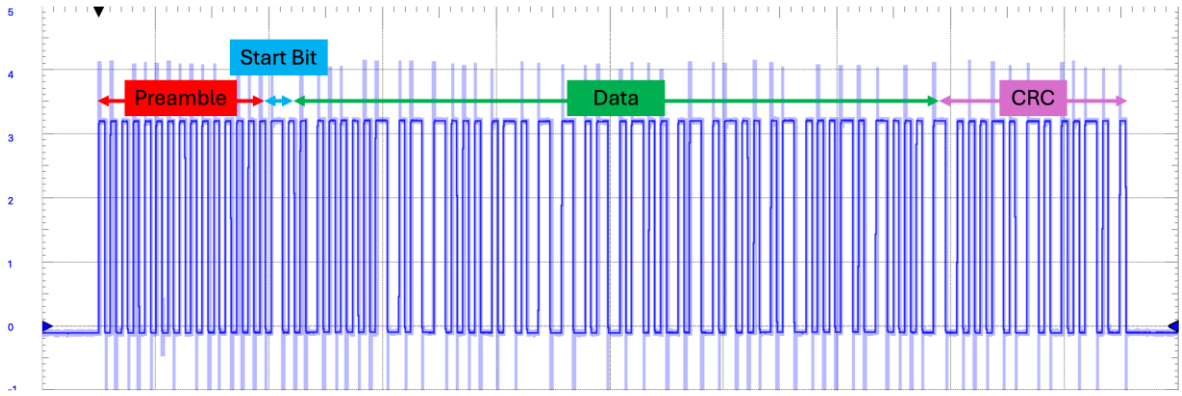


Figura 45 - Forma de onda da mensagem de teste em Manchester.

O conteúdo da mensagem está apresentado na
Tabela 6.

Tabela 6 - Conteúdo da mensagem de teste.

Formato da Mensagem	1 byte	2 byte	3 byte	4 byte	5 byte	6 byte	7 byte
ASCII	?	h	e	l	l	o	!
Binary	00111111	01101000	01100101	01101100	01101100	01101111	00100001
Hex	0x3f	0x68	0x65	0x6c	0x6c	0x6f	0x21

Este conteúdo está codificado em Manchester como se pode observar na Figura 46.

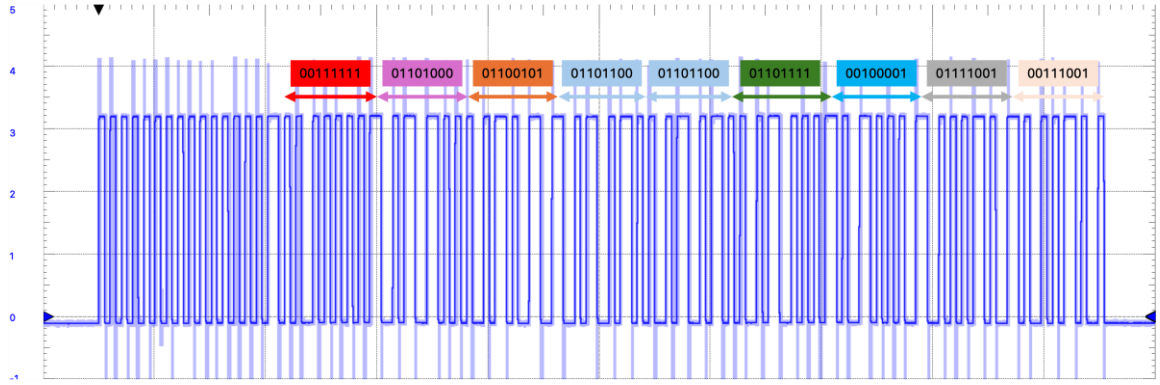


Figura 46 - Forma de onda do conteúdo da mensagem de teste em Manchester.

Testes com comunicações por fio

Começando pelo primeiro teste a transmissão de dados foi realizada utilizando um fio entre os dois microcontroladores.

Os resultados obtidos estão apresentados na Tabela 7.

Tabela 7 - Resultados dos testes de comunicações por fios.

Velocidade dos Testes com comunicações por fio	Resultados
1 kbps	Funciona
2 kbps	Funciona
4 kbps	Funciona

As formas de onda da mensagem codificada com as diferentes velocidades encontram-se nos Anexos E, F e G.

Testes com comunicações por infravermelhos

No segundo teste a transmissão dos dados foi realizada através dos componentes infravermelhos.
Os resultados obtidos estão apresentados na

Tabela 8.

Tabela 8 - Resultados dos testes de comunicações por infravermelhos.

Velocidade dos Testes com comunicações por fio	Resultados
1 kbps	Funciona
2 kbps	Funciona
4 kbps	Não Funciona

As formas de onda da mensagem codificada com as diferentes velocidades encontram-se nos Anexos H, I e J.

Resultados

Para comprovar o sucesso na captura das transições, foram verificados os conteúdos das varáveis do algoritmo com a ferramenta de *debugging* da *Microchip*. Para assim determinar se a captura da mensagem do lado do recetor está igual à mensagem original e se esta é corretamente interpretada e descodificada. A Figura 47 apresenta o conteúdo do *buffer* da mensagem recebida (a mesma da Figura 46), onde é possível observar como o microcontrolador capturou a mensagem.

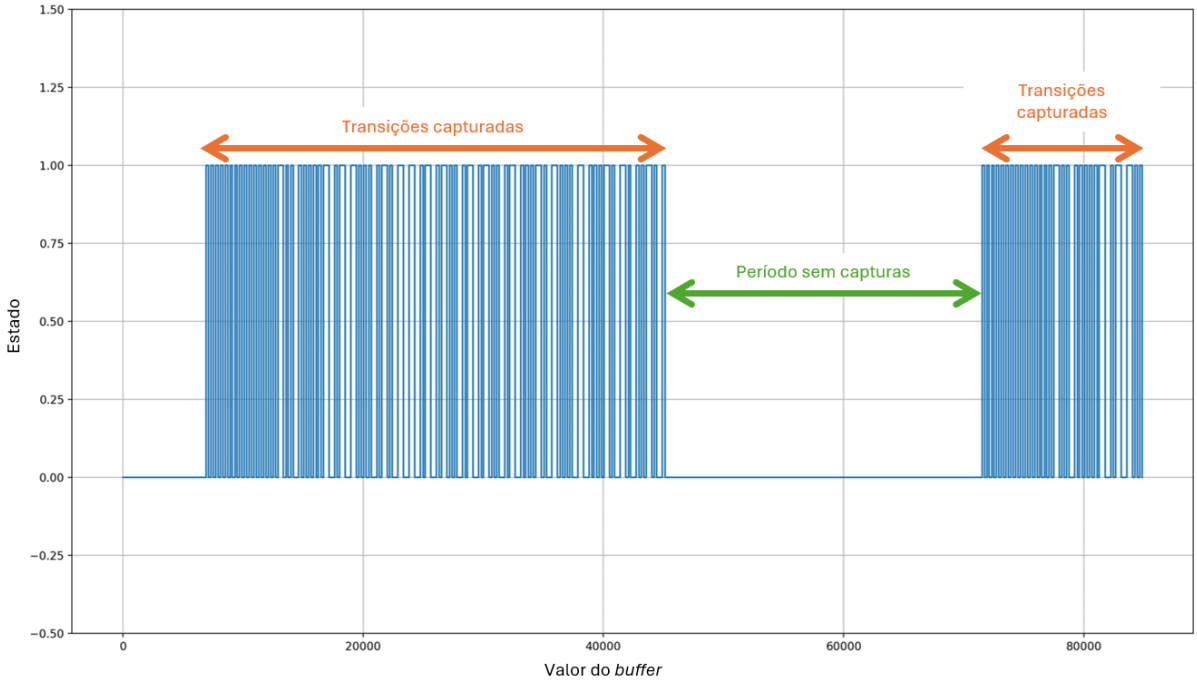


Figura 47 - Transições capturadas no buffer.

Após o processamento do sinal, a mensagem é descodificada e o resultado desse processo é guardado na variável cujo conteúdo se apresenta na Figura 48 a partir da ferramenta de *debbuging* do *integrated development environment* (IDE) da Microchip.

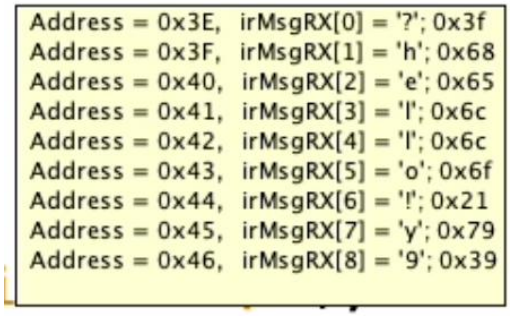


Figura 48 - Conteúdo do buffer.

Problemas e limitações

De acordo com os resultados observados na

Tabela 8, a uma velocidade de transmissão de 4 kbps, a mensagem não é recebida corretamente devido ao pequeno número de impulsos enviados com a portadora de 38 kHz por *bit*. Sendo que o *hardware* interno do recetor só deteta o sinal após um número mínimo de impulsos.

De acordo com o *datasheet* do fabricante [22], o recetor não consegue descriminar corretamente a portadora quando o número de impulsos por *bit* é inferior ao especificado (Anexo J). Isto resulta numa receção de dados errada devido a esta característica do recetor. Na Tabela 9 é possível verificar o número de impulsos enviados pelo emissor, por *bit*, dependendo da velocidade de transmissão.

Tabela 9 - Número de impulsos IR na transmissão da mensagem.

Velocidade de transmissão (kbps)	Número de impulsos de um meio período de *bit*	Número de impulsos de um período inteiro de *bit*
1	20	40
2	10	20
4	5	10

Os dados relativos ao número mínimo de impulsos que o recetor tem de receber, são disponibilizados pelo fabricante e podem ser observados a sublinhado na Tabela 10

Tabela 10 - Características do recetor de infravermelhos [22].

	TSOP382..	TSOP384..
Minimum burst length	10 cycles/burst	10 cycles/burst
After each burst of length a minimum gap time is required of	10 to 70 cycles ≥ 12 cycles	10 to 35 cycles ≥ 12 cycles
For bursts greater than a minimum gap time in the data stream is needed of	70 cycles > 5 x burst length	35 cycles > 15 x burst length
Maximum number of continuous short bursts/second	1700	1700
NEC code	Yes	Preferred
RC5/RC6 code	Yes	Preferred
Thomson RCA 56 kHz code	Yes	Preferred
Sharp code	Yes	Preferred
Sony code	Yes	No
Mitsubishi code	Yes	Preferred
Suppression of interference from fluorescent lamps	Fig. 14	Fig. 14 and Fig. 15

Notes

- For data formats with short bursts please see the datasheet for TSOP383.., TSOP385..
- For Sony 12, 15, and 20 bit IR codes please see the datasheet of TSOP38S40

Proposta de soluções e melhorias

Para atingir o objetivo de uma velocidade de transmissão de 4 kbps por infravermelhos com a codificação de Manchester, seria necessário substituir a componente limitadora que existe no sistema atual do BMS IR, que, neste caso, determinou-se ser o recetor IR.

Uma solução possível, seria implementar um recetor sem o circuito interno de desmodulação, ou seja, um fotodíodo, permitindo que o microcontrolador receba e descodifique o sinal à velocidade pretendida sem grandes alterações adicionais necessárias ao *hardware*.

Esta solução terá as suas desvantagens, nomeadamente a sua suscetibilidade ao ruído, e terá de ser testada num ambiente real para provar a sua eficácia.

7. Conclusões

Este documento apresenta uma análise abrangente do funcionamento de um sistema de gestão de baterias em veículos elétricos no cenário atual, destacando a sua importância para a eficiência e segurança desses veículos. Exploram-se diversos aspetos do BMS IR desenvolvido pelo Instituto Politécnico de Setúbal, incluindo a sua arquitetura, o funcionamento, os componentes e os protocolos de comunicação.

A análise da codificação Manchester apresenta os seus princípios básicos, os métodos de decodificação e as técnicas de deteção de erros evidenciando a sua aplicação em sistemas de comunicação fiáveis e eficientes. Com base nessa análise implementa-se um novo protocolo de comunicações no BMS IR, com o objetivo de melhorar o desempenho e a robustez do sistema.

Na implementação a escolha do método de descodificação mais adequado ao caso de uso recai sobre o método "*Timing Based Decoding*" da Atmel, considerando as condições específicas do BMS IR.

Os testes realizados validam a eficácia do novo protocolo e identificam limitações que impactam a velocidade de transmissão de dados, identificando o recetor IR como fator limitador pelo que se propõe a sua substituição.

Com o novo protocolo, a velocidade de transmissão de dados no BMS melhora significativamente, passando de 1 kbps para 2 kbps. Esta melhoria duplica a taxa de transmissão, aumentando a rapidez na troca de informações cruciais para a gestão das baterias.

Este documento constitui uma base para futuras implementações e desenvolvimentos do BMS IR. As conclusões aqui apresentadas guiam a evolução de novas versões do sistema, contribuindo para a melhoria contínua dos veículos elétricos e a eficiência dos seus sistemas de gestão de baterias. Espera-se que este trabalho inspire avanços tecnológicos que reforcem a transição para uma mobilidade mais sustentável e inovadora.

Referências

[1] Xing, Y., Ma, E. W. M., Tsui, K. L., & Pecht, M. (2011). Battery management systems in electric and hybrid vehicles. Energies, 4(11), 1840–1857. https://doi.org/10.3390/en4111840

[2] Synopsys. (n.d.). What is battery management system in EVs? How does it work? Synopsys. Retrieved October 2024, from https://www.synopsys.com/glossary/what-is-a-battery-management-system.html

[3] Lipu, M. H., Hossain, T. N., & others. (2022). Battery management, key technologies, methods, issues, and future trends of electric vehicles: A pathway toward achieving sustainable development goals.

[4] Zimmer, G. (2024). Wireless battery management systems highlight industry's drive for higher reliability. Linear Technology Corporation (Analog Devices). Retrieved October 2024, from https://www.analog.com/media/en/technical-documentation/technical-articles/s65-wireless_bms_en.pdf

[5] Hannan, M. A., Hoque, M. M., Hussain, A., Yusof, Y., & Ker, P. J. (2018). State-of-the-art and energy management system of lithium-ion batteries in electric vehicle applications: Issues and recommendations. IEEE Access, 6, 19362–19378. https://doi.org/10.1109/ACCESS.2018.2882591

[6] Çeven, S., & Akçay, E. K. (2021). Development of IoT-based battery management system. IEEE Access. https://doi.org/10.1109/IISEC54230.2021.9672346

[7] He, H. X., & Xiong, R. (2012). Online estimation of model parameters and state-of-charge of LiFePO4 batteries in electric vehicles. Applied Energy, 95, 413–420. https://doi.org/10.1016/j.apenergy.2011.08.005

[8] Hu, X. L., & Yao, S. (2012). A comparative study of equivalent circuit models for Li-ion batteries. Journal of Power Sources, 198, 359–367. https://doi.org/10.1016/j.jpowsour.2011.10.013

[9] Plett, G. L. (2004). Extended Kalman filtering for battery management systems of LiPB-based HEV battery packs: Part 2. Modeling and identification. Journal of Power Sources, 134(2), 262–276. https://doi.org/10.1016/j.jpowsour.2004.02.032

[10] Barreras, B., & Chow, M.-Y. (2015). The state-of-the-art approaches to estimate the state of health (SOH) and state of function (SOF) of lithium-ion batteries. In Proceedings of IEEE 13th International Conference on Industrial Informatics (INDIN) (pp. 1302–1307). https://doi.org/10.1109/INDIN.2015.7281923

[11] Antunes, V., Antunes, A., Sousa, J., Maia, J., & Cysneiros, M. A. D. (2021). Boletim da Propriedade Industrial Nº2021/08/11 (Portugal Patente PT 115439 B).

[12] Microchip Technology Inc. (2017). PIC18F46K80 Data Sheet, 8-bit PIC® Microcontrollers. Retrieved October 2024, from https://ww1.microchip.com/downloads/aemDocuments/documents/OTH/ProductDocuments/DataSheets/PIC18F66K80FAMILYEnhancedFlashMCUwithECANXLPTechnology30009977G.pdf

[13] Microchip Technology Inc. (2017). *PIC18F25K80 Data Sheet, 8-bit PIC® Microcontrollers*. Retrieved October 2024, from https://ww1.microchip.com/downloads/aemDocuments/documents/OTH/ProductDocuments/DataSheets/PIC18F66K80FAMILYEnhancedFlashMCUwithECANXLPTechnology30009977G.pdf

[14] Microchip Technology Inc. (2019). *PIC16F18313 Data Sheet, 8-bit PIC® Microcontrollers*. Retrieved October 2024, from https://ww1.microchip.com/downloads/aemDocuments/documents/MCU08/ProductDocuments/DataSheets/40001799F.pdf

[15] Digi-Key Electronics. (2022). Old but still useful: The Manchester code. Retrieved October 2024, from https://www.digikey.pt/en/blog/old-but-still-useful-the-manchester-code

[16] Atmel Corporation. (2015). APPLICATION NOTE Manchester Coding Basics. Retrieved October 2024, from https://ww1.microchip.com/downloads/en/AppNotes/Atmel-9164-Manchester-Coding-Basics_Application-Note.pdf

[17] Microchip Technology Inc. (2012). *AN1470 - Manchester Decoder Using the CLC and NCO*. Retrieved October 2024, from https://ww1.microchip.com/downloads/en/AppNotes/01470A.pdf

[18] Gilbert, H. (2002). *Ethernet Networks (4th edition)*. WILEY.

[19] Altium Limited. (2017). Philips RC5 infrared transmission protocol. Retrieved October 2024, from https://techdocs.altium.com/display/FPGA/Philips+RC5+Infrared+Transmission+Protocol

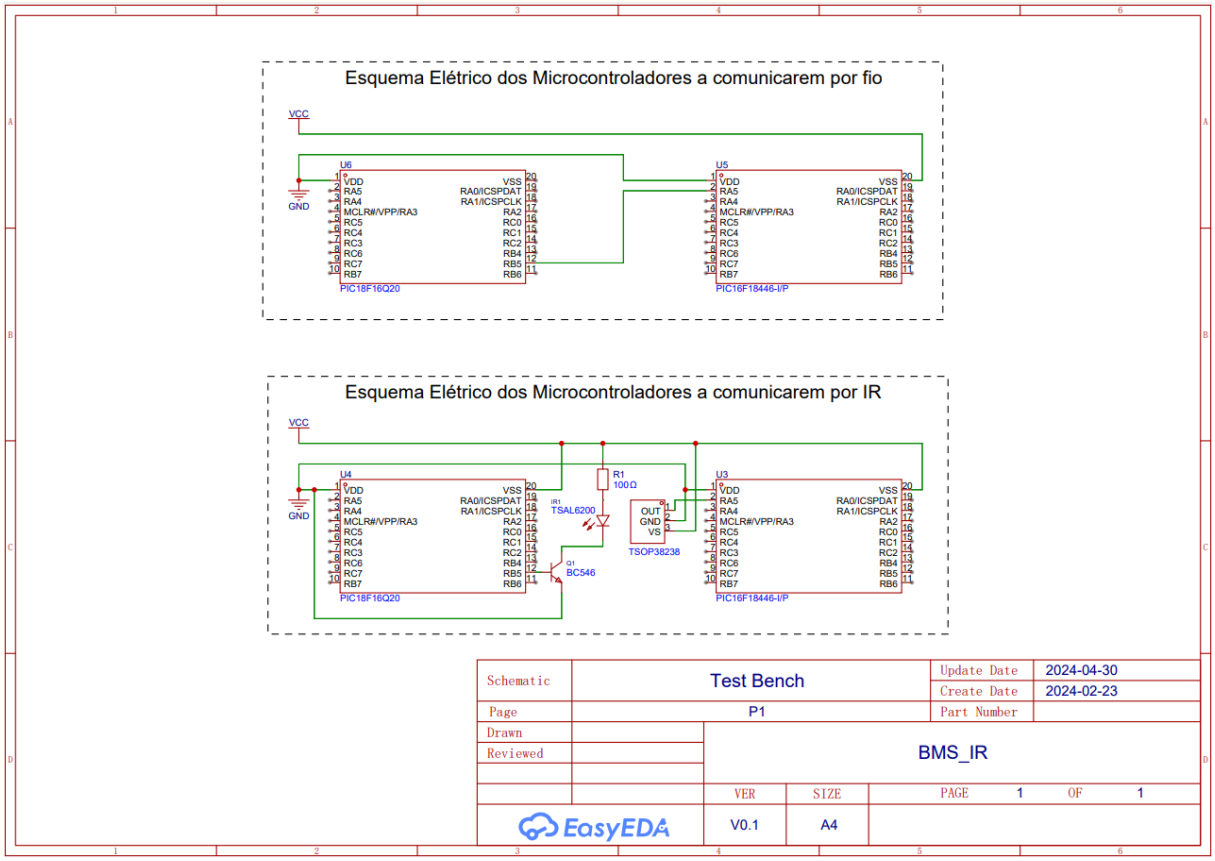
[20] Microchip Technology Inc. (2023). *PIC18F16Q20 Data Sheet, 8-bit PIC® Microcontrollers*. Retrieved October 2024, from https://ww1.microchip.com/downloads/aemDocuments/documents/MCU08/ProductDocuments/DataSheets/PIC18F06-16Q20-Microcontroller-Data-Sheet-DS40002387.pdf

[21] Microchip Technology Inc. (2022). *PIC16F18446 Data Sheet, 8-bit PIC® Microcontrollers*. Retrieved October 2024, from https://ww1.microchip.com/downloads/aemDocuments/documents/MCU08/ProductDocuments/DataSheets/PIC16-L-F18426-46-Micorontrollers-with-XLP-40001985.pdf

[22] Vishay Intertechnology, Inc. (2024). *IR Receiver Modules for Remote Control Systems*. Retrieved October 2024, from https://www.vishay.com/docs/82491/tsop382.pdf

Anexos

ANEXO A – Esquema elétrico da bancada de testes para o protocolo Manchester



ANEXO B – Código do Codificador (main.c)

```
/*
* MAIN Generated Driver File
*
* @file main.c
*
* @defgroup main MAIN
*
* @brief This is the generated driver implementation file for the MAIN driver.
*
* @version MAIN Driver Version 1.0.0
*/

/*
© [2024] Microchip Technology Inc. and its subsidiaries.

Subject to your compliance with these terms, you may use Microchip
software and any derivatives exclusively with Microchip products.
You are responsible for complying with 3rd party license terms
applicable to your use of 3rd party software (including open source
software) that may accompany Microchip software. SOFTWARE IS ?AS IS.?
NO WARRANTIES, WHETHER EXPRESS, IMPLIED OR STATUTORY, APPLY TO THIS
SOFTWARE, INCLUDING ANY IMPLIED WARRANTIES OF NON-INFRINGEMENT,
MERCHANTABILITY, OR FITNESS FOR A PARTICULAR PURPOSE. IN NO EVENT
WILL MICROCHIP BE LIABLE FOR ANY INDIRECT, SPECIAL, PUNITIVE,
INCIDENTAL OR CONSEQUENTIAL LOSS, DAMAGE, COST OR EXPENSE OF ANY
KIND WHATSOEVER RELATED TO THE SOFTWARE, HOWEVER CAUSED, EVEN IF
MICROCHIP HAS BEEN ADVISED OF THE POSSIBILITY OR THE DAMAGES ARE
FORESEEABLE. TO THE FULLEST EXTENT ALLOWED BY LAW, MICROCHIP?S
TOTAL LIABILITY ON ALL CLAIMS RELATED TO THE SOFTWARE WILL NOT
EXCEED AMOUNT OF FEES, IF ANY, YOU PAID DIRECTLY TO MICROCHIP FOR
THIS SOFTWARE.
*/
#include "mcc_generated_files/system/system.h"

/*
Main application
*/

// Manchester Encoding
#define DELAY_HALF_BIT 125
//1kbps - 500us
//2kbps - 250us
//4kbps - 125us
#define RX_BUFFER_SIZE 9

uint8_t IRMsgTx[RX_BUFFER_SIZE];

// ***************************************************************************
```

```
// Routine to Encode a Manchester Data Stream
// ***************************************************************************
void Manchester_Encode(uint8_t data) {
unsigned char mask = 0x80; // Start with the most significant bit

while (mask > 0) {
if (data & mask) {
// Transmit a logical 1 (low-to-high transition)
//CCP1_LoadDutyValue(0); // Set pin low
Encoder_SetLow();
__delay_us(DELAY_HALF_BIT); // Wait for half bit period
//CCP1_LoadDutyValue(103); // Set pin high
Encoder_SetHigh();
} else {
// Transmit a logical 0 (high-to-low transition)
//CCP1_LoadDutyValue(103); // Set pin high
Encoder_SetHigh();
__delay_us(DELAY_HALF_BIT); // Wait for half bit period
//CCP1_LoadDutyValue(0); // Set pin low
Encoder_SetLow();
}

mask >>= 1; // Move to the next bit
__delay_us(DELAY_HALF_BIT); // Wait for half bit period before next transition
}
}

void Manchester_Preamble(){
for(uint8_t i = 0; i<15; i++){
//CCP1_LoadDutyValue(103);
Encoder_SetHigh();
__delay_us(DELAY_HALF_BIT);
//CCP1_LoadDutyValue(0);
Encoder_SetLow();
__delay_us(DELAY_HALF_BIT);
}
//CCP1_LoadDutyValue(103);
Encoder_SetHigh();
__delay_us(DELAY_HALF_BIT*2);
//CCP1_LoadDutyValue(0);
Encoder_SetLow();
__delay_us(DELAY_HALF_BIT);
}

// ***************************************************************************
// CRC Calculation
// ***************************************************************************
uint16_t CalculoCRCSlow(uint8_t *pdata, uint8_t nbytes) {

uint16_t rem;
```

```
uint8_t i, j;

rem = 0;

for (i = 0; i < nbytes; ++i) {
rem ^= (pdata[i] << 7);
for (j = 8; j > 0; --j) {
if (rem & 0x4000)
rem = (rem << 1) ^ 0x4599; // CRC-15
else
rem = (rem << 1);
}
rem &= 0x7FFF; // 15 bit result
}

return rem;
}

// ***********************************************************************
// Main
// ***********************************************************************
int main(void)
{
SYSTEM_Initialize();

printf("\nEncoding Program Starting...\n");

uint16_t crc;

while(1)
{
//Manchester Encoding
IRMsgTx[0] = 0b00111111; // ?
IRMsgTx[1] = 0b01101000; // h
IRMsgTx[2] = 0b01100101; // e
IRMsgTx[3] = 0b01101100; // l
IRMsgTx[4] = 0b01101100; // l
IRMsgTx[5] = 0b01101111; // o
IRMsgTx[6] = 0b00100001; // !
crc = CalculoCRCSlow(&IRMsgTx[0], 7);
IRMsgTx[7] = crc >> 8; // CRC
IRMsgTx[8] = crc & 0x00FF; // CRC

/*IRMsgTx[0] = 0b10110101; // B5
IRMsgTx[1] = 0b00000100; // 4
IRMsgTx[2] = 0b10100101; // A5
IRMsgTx[3] = 0b00000000; // 0
IRMsgTx[4] = 0b11111111; // FF
IRMsgTx[5] = 0b00001111; // F
IRMsgTx[6] = 0b11110000; // F0
```

```
crc = CalculoCRCSlow(&IRMsgTx[0], 7);
IRMsgTx[7] = crc >> 8; // CRC
IRMsgTx[8] = crc & 0x00FF; // CRC
*/

Manchester_Preamble();
Manchester_Encode(IRMsgTx[0]);
Manchester_Encode(IRMsgTx[1]);
Manchester_Encode(IRMsgTx[2]);
Manchester_Encode(IRMsgTx[3]);
Manchester_Encode(IRMsgTx[4]);
Manchester_Encode(IRMsgTx[5]);
Manchester_Encode(IRMsgTx[6]);
Manchester_Encode(IRMsgTx[7]);
Manchester_Encode(IRMsgTx[8]);

LED_Toggle();
//CCP1_LoadDutyValue(0);
Encoder_SetLow();
__delay_ms(1000);
printf("\nMessage Sent: hello!\n");
}
}
```

ANEXO C – Código do Descodificador (main.c)

```
/*
* MAIN Generated Driver File
*
* @file main.c
*
* @defgroup main MAIN
*
* @brief This is the generated driver implementation file for the MAIN driver.
*
* @version MAIN Driver Version 1.0.0
*/

/*
© [2024] Microchip Technology Inc. and its subsidiaries.

Subject to your compliance with these terms, you may use Microchip
software and any derivatives exclusively with Microchip products.
You are responsible for complying with 3rd party license terms
applicable to your use of 3rd party software (including open source
software) that may accompany Microchip software. SOFTWARE IS ?AS IS.?
NO WARRANTIES, WHETHER EXPRESS, IMPLIED OR STATUTORY, APPLY TO THIS
SOFTWARE, INCLUDING ANY IMPLIED WARRANTIES OF NON-INFRINGEMENT,
MERCHANTABILITY, OR FITNESS FOR A PARTICULAR PURPOSE. IN NO EVENT
WILL MICROCHIP BE LIABLE FOR ANY INDIRECT, SPECIAL, PUNITIVE,
INCIDENTAL OR CONSEQUENTIAL LOSS, DAMAGE, COST OR EXPENSE OF ANY
KIND WHATSOEVER RELATED TO THE SOFTWARE, HOWEVER CAUSED, EVEN IF
MICROCHIP HAS BEEN ADVISED OF THE POSSIBILITY OR THE DAMAGES ARE
FORESEEABLE. TO THE FULLEST EXTENT ALLOWED BY LAW, MICROCHIP?S
TOTAL LIABILITY ON ALL CLAIMS RELATED TO THE SOFTWARE WILL NOT
EXCEED AMOUNT OF FEES, IF ANY, YOU PAID DIRECTLY TO MICROCHIP FOR
THIS SOFTWARE.
*/
#include "mcc_generated_files/system/system.h"

// Capture Buffer
#define CCP2BUFFER 200

// Capture Transitions Variables
uint8_t ccp2array = 0;
uint16_t ccp2value[CCP2BUFFER];

// Manchester Decoding Buffer
#define RX_BUFFER_SIZE 9

// Manchester Decoding Variables
bool previousBit = false;
bool preamble_received = false;
bool receivedSyncBit = false;
```

```
uint8_t premable_index = 0;
uint8_t startMsgPosition = 0;
uint8_t receivedByte = 0;
uint8_t bufferIndexBit = 0;
uint8_t bufferIndexByte = 0;
uint8_t irMsgRX[RX_BUFFER_SIZE];

// Acceptable Threshold For a Half Period Transition (Every Edge Mode)
#define EVERY_UPPER_THRESHOLD_HALFT 250
#define EVERY_LOWER_THRESHOLD_HALFT 150

// Acceptable Threshold For a Full Period Transition (Every Edge Mode)
#define EVERY_UPPER_THRESHOLD_FULLT 500
#define EVERY_LOWER_THRESHOLD_FULLT 400

enum DecodeState {
IDLE,
FIRST_HALF_BIT,
SECOND_HALF_BIT
};

volatile enum DecodeState currentState = IDLE;

// ************************************************************************
// Routine to Validate Captured Transitions
// ************************************************************************
bool isCaptureInRangeHP(uint16_t captureValue) {  //Capture half period wave
return ((captureValue >= EVERY_LOWER_THRESHOLD_HALFT) && (captureValue <=
EVERY_UPPER_THRESHOLD_HALFT));
}

bool isCaptureInRangeFP(uint16_t captureValue) {  //Capture full period wave
return ((captureValue >= EVERY_LOWER_THRESHOLD_FULLT) && (captureValue <=
EVERY_UPPER_THRESHOLD_FULLT));
}

// ************************************************************************
// Routine to Process the Captured Transition
// ************************************************************************
void processBit(bool bitValue) {
// Add the bit to the received byte
receivedByte |= (bitValue << (7 - bufferIndexBit));
bufferIndexBit++;

if(bufferIndexBit == 8){
bufferIndexBit = 0;
previousBit = false;

irMsgRX[bufferIndexByte] = receivedByte;
bufferIndexByte++;
```

```
receivedByte = 0;

if(bufferIndexByte == 9){
bufferIndexByte = 0;
ccp2array = 0;
receivedSyncBit = false;
currentState = IDLE;
}
}
}

// ***********************************************************************
// CRC Calculation
// ***********************************************************************
uint16_t CalculoCRCSlow(uint8_t *pdata, uint8_t nbytes) {

uint16_t rem;
uint8_t i, j;

rem = 0;

for (i = 0; i < nbytes; ++i) {
rem ^= (pdata[i] << 7);
for (j = 8; j > 0; --j) {
if (rem & 0x4000)
rem = (rem << 1) ^ 0x4599; // CRC-15
else
rem = (rem << 1);
}
rem &= 0x7FFF; // 15 bit result
}
return rem;
}

/*
Main application
*/

int main(void)
{
SYSTEM_Initialize();

printf("\nDecoder Program Starting...\n");

// If using interrupts in PIC18 High/Low Priority Mode you need to enable the Global High and Low Interrupts
// If using interrupts in PIC Mid-Range Compatibility Mode you need to enable the Global and Peripheral Interrupts
// Use the following macros to:
```

```
// Enable the Global Interrupts
INTERRUPT_GlobalInterruptEnable();

// Disable the Global Interrupts
//INTERRUPT_GlobalInterruptDisable();

// Enable the Peripheral Interrupts
INTERRUPT_PeripheralInterruptEnable();

// Disable the Peripheral Interrupts
//INTERRUPT_PeripheralInterruptDisable();

uint16_t crc_calculado, crc_recebido;

while(1)
{
if(ccp2array == 200){

INTERRUPT_PeripheralInterruptDisable();

if(!preamble_received){
for(uint8_t j = 0; j < 40; j++){
if(isCaptureInRangeHP(ccp2value[j])){
premable_index++;
}

if(premable_index > 15){
if(isCaptureInRangeFP(ccp2value[j])){
preamble_received = true;
receivedSyncBit = true;
premable_index = 0;
startMsgPosition = j+1;
j = 40;
} else {
// If there is no preamble detected, start again
preamble_received = false;
receivedSyncBit = false;
}
}
}
}

if(receivedSyncBit){
for(uint8_t i = 32; i < ccp2array; i++){
//State-Machine Manchester Decoding
switch(currentState){
case IDLE:
if(isCaptureInRangeHP(ccp2value[i])){
currentState = FIRST_HALF_BIT;
```

```
} else if(isCaptureInRangeFP(ccp2value[i])){
currentState = SECOND_HALF_BIT;
} else {
                    // Exit the state machine on unexpected pulse
return;
}
break;

case FIRST_HALF_BIT:
if(isCaptureInRangeHP(ccp2value[i])){
if(previousBit){
previousBit = true;
processBit(true);
currentState = IDLE;
} else {
previousBit = false;
processBit(false);
currentState = IDLE;
}
} else if(isCaptureInRangeFP(ccp2value[i])){
if(previousBit){
previousBit = true;
processBit(true);
previousBit = false;
processBit(false);
currentState = IDLE;
} else {
previousBit = false;
processBit(false);
previousBit = true;
processBit(true);
currentState = IDLE;
}
} else {
                    // Exit the state machine on unexpected pulse
return;
                    }
}
break;

case SECOND_HALF_BIT:
if(isCaptureInRangeHP(ccp2value[i])){
if(previousBit){
previousBit = false;
processBit(false);
currentState = IDLE;
} else {
previousBit = true;
processBit(true);
currentState = IDLE;
```

```
}
} else if(isCaptureInRangeFP(ccp2value[i])){
if(previousBit){
previousBit = false;
processBit(false);
previousBit = true;
processBit(true);
currentState = IDLE;
} else {
previousBit = true;
processBit(true);
previousBit = false;
processBit(false);
currentState = IDLE;
} else {
                    // Exit the state machine on unexpected pulse
return;
                    }
}
break;
}
}
}

LED_Toggle();
printf("\nArray is full!");
printf("\n%u", irMsgRX[0]);
printf("\n%u", irMsgRX[1]);
printf("\n%u", irMsgRX[2]);
printf("\n%u", irMsgRX[3]);
printf("\n%u", irMsgRX[4]);
printf("\n%u", irMsgRX[5]);
printf("\n%u", irMsgRX[6]);
printf("\n%u", irMsgRX[7]);
printf("\n%u", irMsgRX[8]);

crc_recebido = irMsgRX[7] << 8;
crc_recebido |= irMsgRX[8];
crc_calculado = CalculoCRCSlow(&irMsgRX[0], 7);

if (crc_calculado != crc_recebido){
printf("\nERROR");
} else {
printf("\nCRC is correct!");
}
__delay_ms(500);
INTERRUPT_PeripheralInterruptEnable();
}
}
}
```

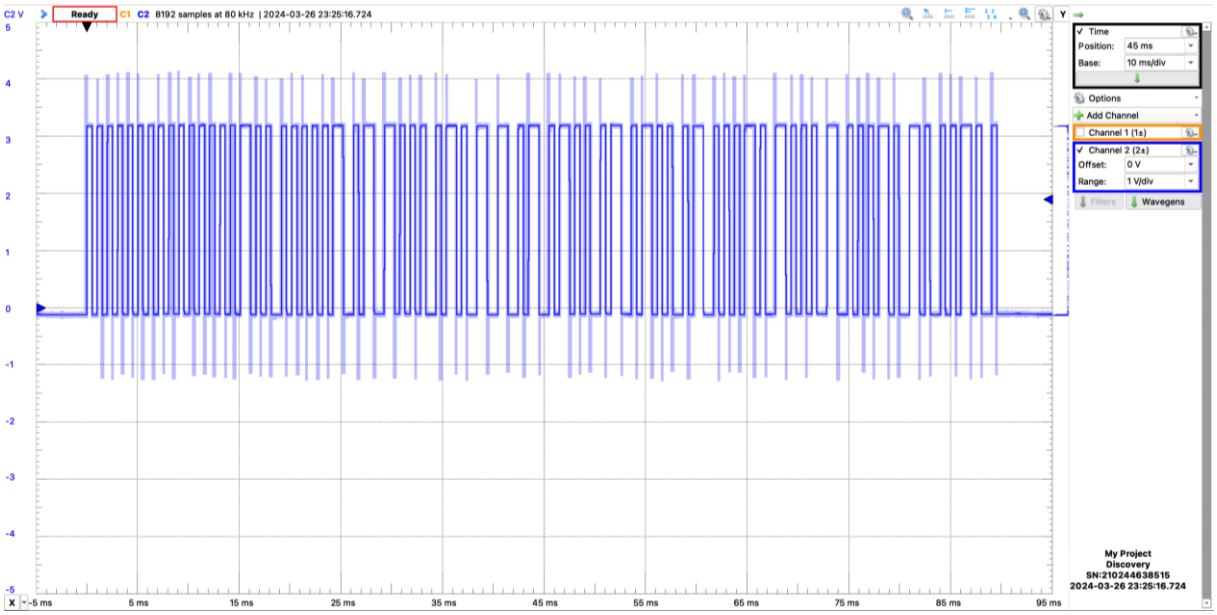
ANEXO D – Código do Descodificador (ccp1.c)

```
/**
* CCP1 Generated Driver File.
*
* @file ccp1.c
*
* @ingroup capture1
*
* @brief This file contains the API implementation for the CCP1 driver.
*
* @version CCP1 Driver Version 2.0.2
*/
/*
© [2024] Microchip Technology Inc. and its subsidiaries.

Subject to your compliance with these terms, you may use Microchip
software and any derivatives exclusively with Microchip products.
You are responsible for complying with 3rd party license terms
applicable to your use of 3rd party software (including open source
software) that may accompany Microchip software. SOFTWARE IS ?AS IS.?
NO WARRANTIES, WHETHER EXPRESS, IMPLIED OR STATUTORY, APPLY TO THIS
SOFTWARE, INCLUDING ANY IMPLIED WARRANTIES OF NON-INFRINGEMENT,
MERCHANTABILITY, OR FITNESS FOR A PARTICULAR PURPOSE. IN NO EVENT
WILL MICROCHIP BE LIABLE FOR ANY INDIRECT, SPECIAL, PUNITIVE,
INCIDENTAL OR CONSEQUENTIAL LOSS, DAMAGE, COST OR EXPENSE OF ANY
KIND WHATSOEVER RELATED TO THE SOFTWARE, HOWEVER CAUSED, EVEN IF
MICROCHIP HAS BEEN ADVISED OF THE POSSIBILITY OR THE DAMAGES ARE
FORESEEABLE. TO THE FULLEST EXTENT ALLOWED BY LAW, MICROCHIP?S
TOTAL LIABILITY ON ALL CLAIMS RELATED TO THE SOFTWARE WILL NOT
EXCEED AMOUNT OF FEES, IF ANY, YOU PAID DIRECTLY TO MICROCHIP FOR
THIS SOFTWARE.
*/

/**
Section: Included Files
*/

#include <xc.h>
#include "../ccp1.h"

extern uint16_t ccp2value[];
extern uint8_t ccp2array;

static void (*CCP1_CallBack)(uint16_t);

/**
Section: Capture Module APIs
*/
```

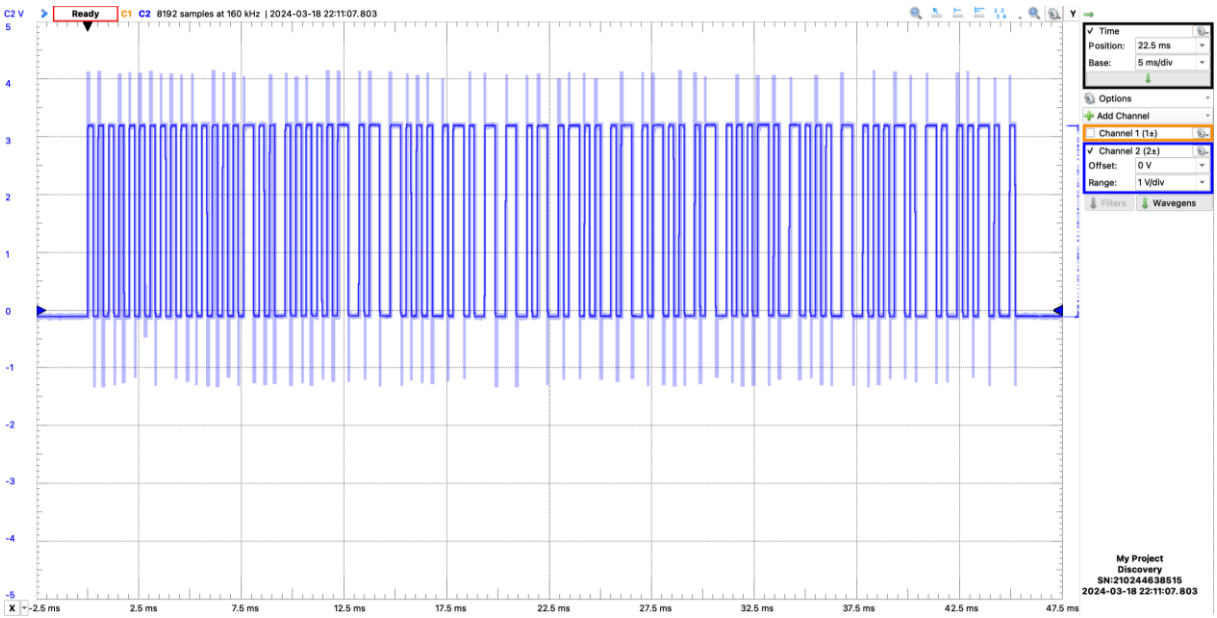
```
/**
* @ingroup capture1
* @brief Default callback function for the capture interrupt events.
* @param capturedValue - 16-bit captured value.
* @return None.
*/
static void CCP1_DefaultCallBack(uint16_t capturedValue) {
// Add your code here
}

void CCP1_Initialize(void)
{
// Set the CCP1 to the options selected in the User Interface

// CCPM Every edge; EN enabled; FMT right_aligned;
CCP1CON = 0x83;

// CTS CCP1 pin;
CCP1CAP = 0x0;

// CCPRH 0;
CCPR1H = 0x0;

// CCPRL 0;
CCPR1L = 0x0;

// Set the default call back function for CCP1
CCP1_SetCallBack(CCP1_DefaultCallBack);

// Selecting Timer 1
CCPTMRS0bits.C1TSEL = 0x1;

// Clear the CCP1 interrupt flag
PIR6bits.CCP1IF = 0;

// Enable the CCP1 interrupt
PIE6bits.CCP1IE = 1;
}

void CCP1_CaptureISR(void)
{
CCPR1_PERIOD_REG_T module;

// Clear the CCP1 interrupt flag
PIR6bits.CCP1IF = 0;

ccp2value[ccp2array] = CCPR1;
ccp2array++;

TMR1H = 0;
```

```
TMR1L = 0;

// Copy captured value.
//module.ccpr1l = CCPR1L;
//module.ccpr1h = CCPR1H;

// Return 16-bit captured value
//CCP1_CallBack(module.ccpr1_16Bit);
}

void CCP1_SetCallBack(void (*customCallBack)(uint16_t)){
CCP1_CallBack = customCallBack;
}
/**
End of File
*/
```

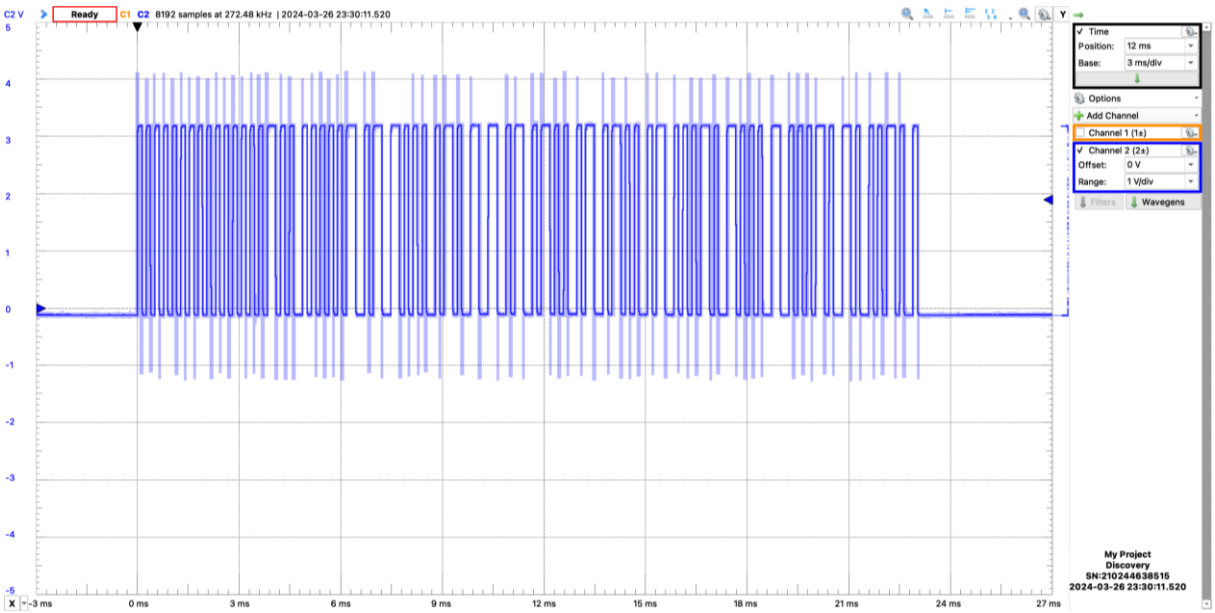
ANEXO E – Formas de onda da transmissão da mensagem de teste com fio a 1 kbps



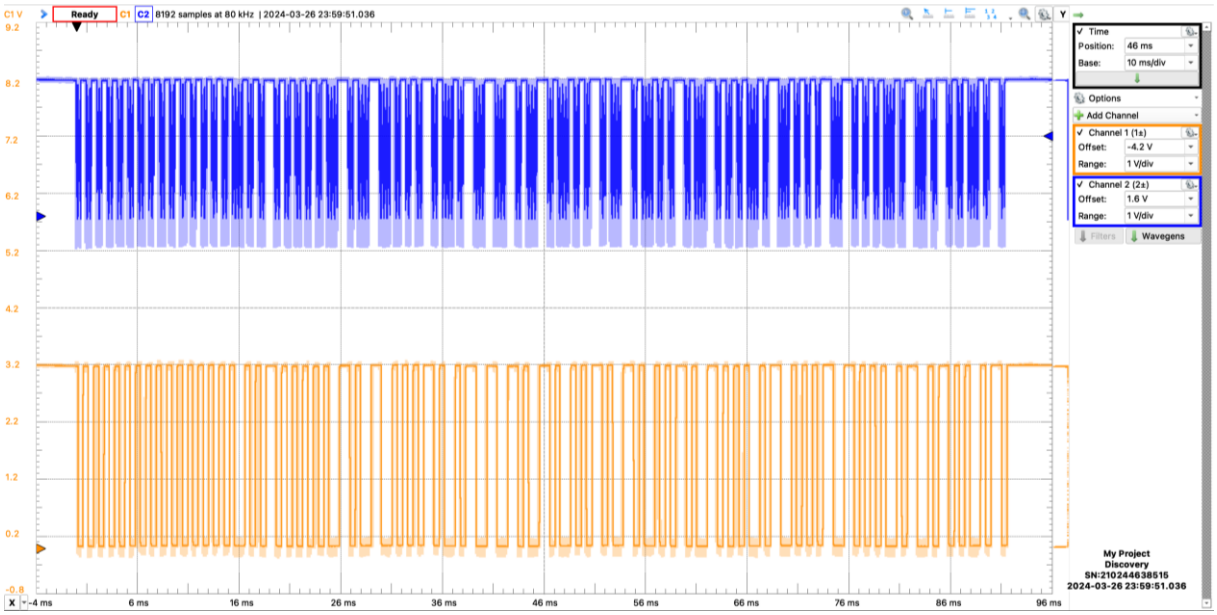
ANEXO F – Formas de onda da transmissão da mensagem de teste com fio a 2 kbps



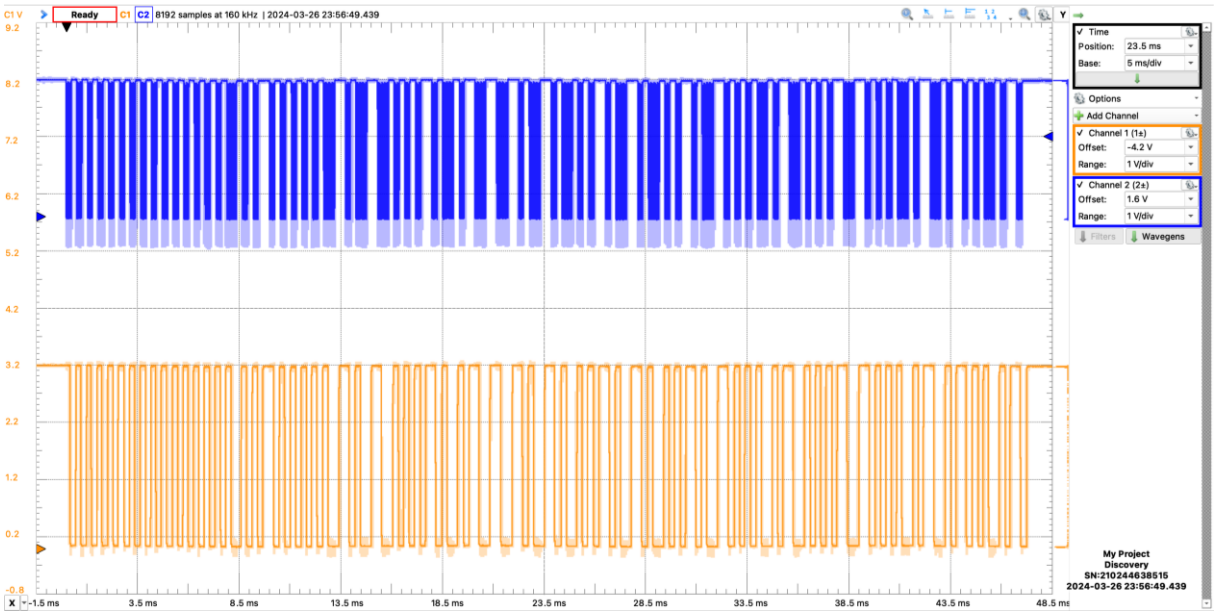
ANEXO G – Formas de onda da transmissão da mensagem de teste com fio a 4 kbps



ANEXO H – Formas de onda da transmissão da mensagem de teste por infravermelhos a 1 kbps



ANEXO I – Formas de onda da transmissão da mensagem de teste por infravermelhos a 2 kbps



ANEXO J – Formas de onda da transmissão da mensagem de teste por infravermelhos a 4 kbps (com maior resolução na segunda imagem)