Universidade do Minho

Microdispositivos de RF para Comunicação sem Fios

MIEEICOM /MIETI

Projeto Prático 2 - Bluetooth

2019/2020

Grupo 11

Grupo de Trabalho

Nome completo: André Filipe Marques da Silva Machado

E-mail: a66693@alunos.uminho.pt

Nome completo: Pedro Rafael Gonçalves Dourado

E-mail: a77973@alunos.uminho.pt

Nome completo: Pedro Rafael Carvalho Miranda

E-mail: a79662@alunos.uminho.pt

Nome completo: Pedro Alexandre Morais Alves

E-mail: a61893@alunos.uminho.pt

Índice

Introdução 4
Estado da arte 5
Módulo Bluetooth Slave HC06 5
Arduíno UNO 6
Bluetooth 7
Implementação 9
Arquitetura 9
Arduíno 10
MATLAB 11
Análise de Resultados 13
Conclusão 15
Referências 16

Índice de figuras

Figura 1 - Comunicação Arduíno - Bluetooth. 9
Figura 2 - Código Arduíno. 10
Figura 3 - Conexão entre o MATLAB e o módulo Bluetooth. 11
Figura 4 - Receção e tratamento de dados 12
Figura 5 - Representação gráfica doa 10 primeiros valores. 13
Figura 6 - Atualização da representação gráfica ao longo da receção de dados. 14

Introdução

No âmbito da Unidade Curricular de "Microdispositivos RF para Comunicação Sem Fios", inserida no Mestrado Integrado em Engenharia de Telecomunicações e Informática, foi proposta a elaboração de um sistema de comunicações sem fios com o qual se pretende transmitir dados, que são gerados por um Arduíno conectado a um módulo Bluetooth *slave* HC06 que permite enviar dados através de uma ligação por porta série, e na qual posteriormente são enviados para um computador com ligação Bluetooth, para serem posteriormente processados e representados em tempo real.

Este projeto introduz uma componente importante uma vez que o Bluetooth se trata de uma das tecnologias abordadas no desenrolar desta unidade curricular.

A implementação deste sistema leva a que seja necessário a programação de um Arduíno, de forma a que este seja capaz de gerar uma *frame* na qual pode ser por exemplo uma sequência de inteiros. Para tal, a sua programação é realizada com recurso ao Arduíno IDE *(Integrated Development Environment),* que se trata de uma aplicação para escrita de código de linguagem C e C++ na qual se pode efetuar *upload* para placas compatíveis com Arduíno. De forma a configurar a ligação, cada grupo possui valores específicos, no qual ao nosso foi atribuído o valor de 4800 ao BaudRate. Ainda assim devem ser efetuadas outras configurações, como a alteração do PIN e ainda o seu nome. Após a receção dos dados pelo computador, escolheu-se representá-los através do uso do MATLAB com recurso a um gráfico.

A realização deste relatório visa explicar todas as decisões elaboradas ao longo da conceção do projeto, assim como foi solucionado e as suas diferentes configurações. No final são ainda apresentados os resultados e respetiva análise aos mesmos.

Estado da arte

Neste capítulo procura-se introduzir os diferentes componentes do sistema para posteriormente ser possível implementar o sistema em questão.

Módulo Bluetooth Slave HC06

O módulo Bluetooth utilizado neste projeto é o HC06 Slave, que permite a transmissão e receção de dados através do uso da tecnologia *wireless* Bluetooth. A transmissão de dados pode ser efetuada até uma distância de 10 metros se não existirem obstáculos entre o emissor e o recetor, o que se pode concluir que opera sobre a classe 2 de potência Bluetooth, na qual foi abordada ao longo da unidade curricular.

Este módulo é composto por 6 pinos, porém só pode ser efetuado o uso de 4 destes diretamente com a interface sem a necessidade de mais ações como solda ou outro procedimento alternativo.

Para a conexão com o Arduíno, este módulo possui 2 pinos, o pino 4 TxD para a transmissão e o pino 5 RxD para a receção, permitindo assim a comunicação com outros dispositivos através de redes sem fios. [1] O pino 1 chamado de "key", permite definir se o módulo opera sobre o modo High, ou seja, "AT command", ou no modo Low classificado como "Normal mode". O pino 2 é o VCC do módulo na qual deve ser alimentado por uma tensão de 5 Volts, e similarmente o pino 3 é o GND do módulo na qual deve ser ligado ao *ground*. Por fim, o pino 6 denominado de "State" na qual está conectado ao LED do módulo que representa o estado da placa.

Como este módulo se trata do módulo *slave*, permite apenas que outros dispositivos se conectem a este.

A configuração deste módulo, é realizada através do comando AT. Com os seguintes comandos é permitido:

- AT: Teste da comunicação;
- AT+BAUDX: Alteração do *baudrate*, onde X representa o *baudrate* desejado. X pode assumir um valor dos abaixo indicados:
    - 1 - 1200;

    - 2 - 2400;
    - 3 – 4800;
    - 4 – 9600 (*Default*);
    - 5 – 19200;
    - 6 – 38400;
    - 7 – 57600;
    - 8 – 115200;
    - 9 – 230400;
    - A – 460800;
    - B – 921600;
    - C – 1382400.
- AT+NAMEX: Alteração do nome, onde X representa o nome desejado;
- AT+PINXXXX: Alteração da *password*, onde XXXX representa a *password* desejada;
- AT+PN: Desabilitar a verificação de paridade;
- AT+PE: Habilitar a verificação de paridade;
- AT+PO: Habilitar a verificação de paridade ímpar;
- AT+VERSION: Obter a versão do AT. [2]

Arduíno UNO

É uma plataforma *open-source*, o que significa que qualquer pessoa pode modificar e otimizar a placa com base no que se deseja conceber. O *software* utilizado é o Arduíno IDE *(Integrated Development Environment)* que é grátis e suporta as linguagens de programação C e C++. O Arduíno UNO é uma das muitas variantes existentes, que tem como base o microcontrolador ATmega328. O Arduíno UNO possui uma interface USB, que permite conectar diretamente um computador, com o objetivo de transferir código através do IDE. Este possui 6 pinos analógicos e 14 portas digitais para *input* e *output* que permitem a conexão de dispositivos externos. De forma a fornecer energia à placa, é necessária uma ligação de 5 volts que é fornecida através da entrada USB. Contudo, este suporta fontes de energia até um máximo de 12 volts. A existência da ligação à terra (GND) permite a existência de um ponto de referência para o controlo das voltagens. [3]

Bluetooth

O Bluetooth é uma especificação das redes WPAN *(Wireless Personal Area Networks)*. Esta tecnologia possui diversas versões, e procura encontrar maneira de trocar informações entre dispositivos num determinado raio de alcance. A transmissão é efetuada por meio de radiofrequências o que facilita o facto de um dispositivo poder detetar outro independentemente da localização deste. Este opera sobre a faixa *Industrial, Scientific, Medical* (ISM) que opera à frequência de 2,4 GHz, e para que evite interferências de outros dispositivos que utilizem esta frequência. O esquema de comunicação segue o Frequency Hopping – Code-Division Multiple Access (FH-CDMA). Este esquema de comunicação tem por base múltiplos saltos da portadora sobre canais de frequência usando uma sequência pseudoaleatória conhecida pelo transmissor e recetor. Cada frequência é dividida em sub-frequências e o sinal rapidamente muda de "Hop" de forma a que quando ocorra uma interferência, interfira com o sinal só num curto período. A comunicação Bluetooth funciona sobre o modo *full-duplex,* dessa forma os canais podem alternar entre slots para transmitir e receber com o esquema *Frequency Hopping / Time Division Duplex* (FH/TDD) sendo cada slot dividido em períodos de 625 µs, ou seja, por segundo existe 1600 saltos. No Bluetooth pode-se utilizar até 79 frequências ou 23 dependendo do país e cada portadora encontra-se espaçada por intervalos de 1 MHz.

O Bluetooth faz uso de dois padrões para *paring*:

- Synchronous Connection-Oriented (SCO) – utilizado sobretudo para envio de dados contínuos;
- Asychronous Connection-Less (ACL) – utilizada sobretudo para link de dados.

Quando dois ou mais dispositivos se conectam por Bluetooth forma-se uma rede denominada de Piconet. O dispositivo que iniciou a conexão é denominado de *master*, na qual é responsável por regular a transmissão de dados, padrão de hoping e sincronismo dos *slaves* que podem ser no máximo 7. O *master* fornece o seu *clock* aos dispositivos (BD_ADDR), para determinar também a fase do padrão *hopping.* Quando várias *piconets* se ligam forma-se uma *scatternet,* e é de salientar que os dispositivos podem saltar entre Piconets.

O Bluetooth sofreu assim várias iterações, sendo a primeira versão a 1.0 e de seguida 1.0B e devido a isso os fabricantes encontravam problemas que dificultavam a sua implementação e interoperabilidade entre dispositivos. A versão 1.1 foi sobretudo para correção de erros e a versão 1.2 introduziu dois conceitos:

- **Adaptive Frequency-hopping spread spectrum (AFH)-improves -** Melhora a robustez à interferência de radiofrequência, evitando o uso de frequências lotadas na sequência de salto;
- **Extended Synchronous Connections (eSCO) -** melhora a voz qualidade dos links de áudio com retransmissões de pacotes corrompidos.

A versão 2.0 introduziu o Enhanced Data Rate (EDR) alcançando melhores esquemas de modulação e de velocidade mais alta para dados úteis, porém esta versão não necessita desta tecnologia para operar.

A versão 3.0 apresentou melhorias sobre a velocidade em dispositivos compatíveis com as instruções High Speed (HS) e o consumo de energia também foi melhorado. Com a versão 4 foi incluído o BLE na qual a eficiência de energia foi o seu foco. Por fim a versão 5 permite maiores distâncias e velocidades assim como usam tecnologias que diminuem o risco de interferências.

Este está dividido em classes de potência distintas como se pode observar na seguinte tabela de forma a atender os vários tipos de dispositivos.

Tabela 1 - Classes de potência Bluetooth.

Classe	Potência máxima permitida	Alcance
Classe 1	100 mW (20 dBm)	Até 100 metros
Classe 2	2.5 mW (4dBm)	Até 10 metros
Classe 3	1 mW (0dBm)	Até aproximadamente 1 metro

As velocidades de transmissão também têm vindo a aumentar as diferentes versões que surgiram. [4]

Tabela 2 - Velocidades de transmissão das diferentes versões Bluetooth.

Versão	Taxa de transmissão
1.2	1 Mbps
2.0 + EDR	3 Mbps
3.0	24 Mbps
4.0	25 Mbps
5.0	50 Mbps

Implementação

Neste capítulo procura-se abordar as diferentes decisões, assim como retratar como se implementou o sistema.

Arquitetura

Na figura seguinte encontra-se representada a comunicação entre Arduíno e módulo Bluetooth Slave HC06. Para a elaboração deste esquema foi utilizado o Fritzing que é um *software* que permite o *design* de circuitos eletrónicos.

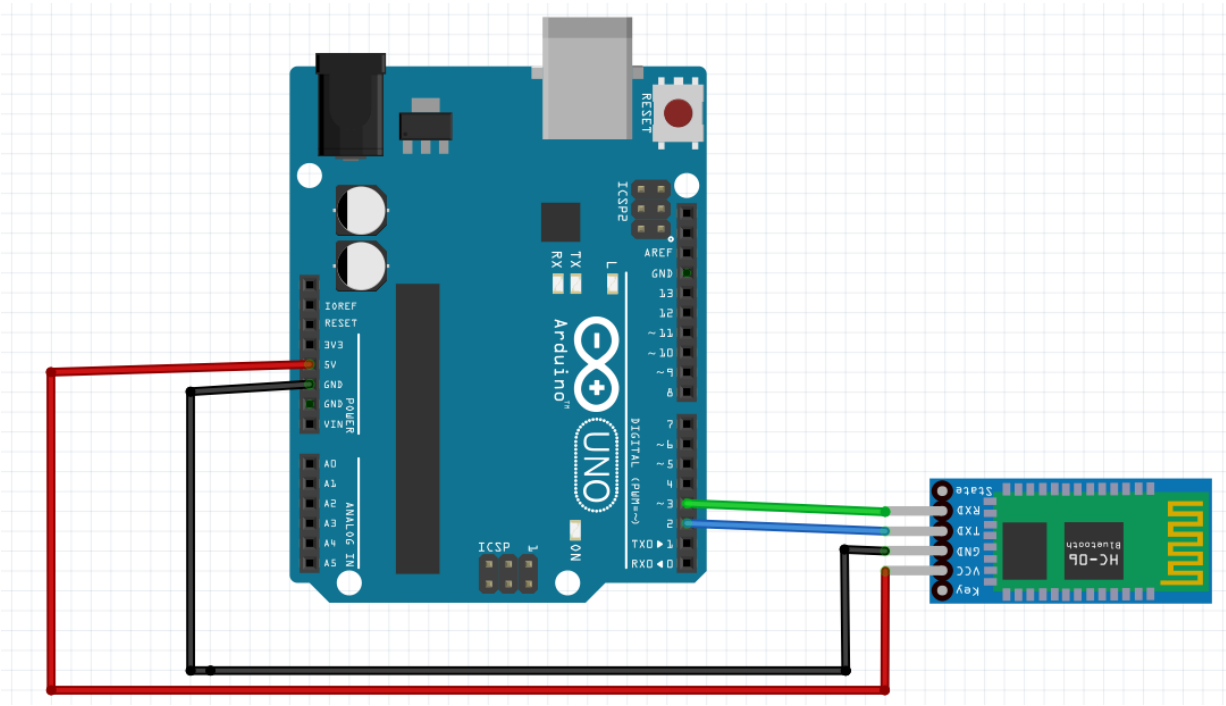

Figura 1 - Comunicação Arduíno - Bluetooth.

O módulo HC06 possui seis pinos, como já foi abordado numa das secções do capítulo anterior, porém na ligação foi apenas utilizado 4 pinos para estabelecimento da comunicação, uma vez que não se pretende alterar o modo do módulo nem o LED do módulo. Relativamente à alimentação do sistema o pino VCC, pino de alimentação é especificado no datasheet que deve se encontrar ligado aos 5 volts fornecidos pelo Arduíno, de seguida o pino 4 “GND”, ponto de referência a partir do qual todas as voltagens são medidas, este é conectado ao pino GND do Arduíno. Relativamente à comunicação o RxD, pino de receção foi conectado ao pino 3 e o TxD do modulo, pino de transmissão, encontra-se conectado ao pino 2 do Arduino.

Arduíno

Na figura seguinte encontra-se exposto o código referente ao Arduíno.

```
#include <SoftwareSerial.h>

SoftwareSerial hc06(2,3);

void setup(){
  //Initialize Serial Monitor
  Serial.begin(4800);
  Serial.println("ENTER AT Commands:");
  //Initialize Bluetooth Serial Port
  hc06.begin(4800);
}

void loop(){
  //Write from Serial Monitor to HC06
    for(int i=1;i<=10;i++){
      hc06.println(i);
    delay(3000);
  }
}
```

Figura 2 - Código Arduíno.

Podemos dividir este código em duas partes distintas. Na primeira delas é realizada a configuração da ligação, definindo-se o valor do Baud Rate e os pinos utilizados na comunicação série. Para tal são utilizados os seguintes comandos:

- SoftwareSerial(Rx,Tx) – definição dos pinos da porta série;
- Serial.begin(4800) – configuração do Baud Rate utilizado para a porta série;
- Hc06.begin(4800) –inicialização do módulo Bluetooth com o mesmo valor do Baud Rate utilizado na configuração da porta série.

Os dados gerados pelo Arduíno são enviados, com recurso ao módulo Bluetooth, a cada 3 segundos para que não ocorra *overflow* no buffer do recetor, para tal foi utilizada a instrução "delay". A sequência enviada é um conjunto de 10 inteiros, que representam os números de 1

até 10. Para tal é utilizado um ciclo “for” de forma a enviar um valor de cada vez. Para o envio são utilizados os seguintes comandos:

- Hc06.println(i) – envio de uma string no formato ASCII;
- delay(3000) – causa um delay durante o intervalo de 3 segundos.

MATLAB

No MATLAB foram desenvolvidas 2 *scripts* distintas, uma primeira para estabelecimento da conexão, e outra para a receção e para o tratamento de dados enviados pelo módulo HC06.

O código que garante a conexão com o módulo HC06 encontra-se exposto na figura 3.

```
%Script de conecção com o HC06
b = Bluetooth('HC06dobem',1);
fopen(b);
```

Figura 3 - Conexão entre o MATLAB e o módulo Bluetooth.

Assim sendo, passamos à explicação das ações que são efetuadas para conectar o MATLAB ao módulo HC06 utilizando as seguintes funções:

- Bluetooth(‘HC06dobem’,1) – Cria um objeto *Bluetooth* associando o nome do módulo ao canal que este está a utilizar;
- fopen() – Faz a ligação do objeto *Bluetooth* ao MATLAB.

Para receção e tratamento de dados é utilizado o código exposto na figura 4.

```
%Script de comunicação com o HC06
v = []; %inicialização de vector vazio
valor = b.BytesAvailable; %obtenção do número de bytes disponíveis para ler
nan = isnan(valor); %verificar se valor é NaN (Not a Number)

while (valor > 2) %enquanto houver mais de 2 bytes para ler
    if (valor ~= nan)
        ler = fscanf(b); %ler valor enviado por bluetooth
        j = str2double(ler); %converter de string para número
        disp(j) %escrever esse valor
        v = [v j]; %preencher o vector

        %desenhar o gráfico
        P = plot(v, '*');
        set(P,'color','r');
        xlabel('Tempo');
        ylabel('Valores');
        title('Valores obtidos por Bluetooth');
        drawnow

        %se o tamanho do que foi lido for igual ao número de bytes
        %disponíveis -> leitura completa
        if (strlength(ler)==valor)
            valor = 0;
            ler = 0;
        end
    end
end
```

Figura 4 - Receção e tratamento de dados

A receção e tratamento dos dados, tem 4 etapas. Inicialmente é obtido o número de bytes de dados disponíveis para ler, acedendo à propriedade do *buffer* BytesAvailable. A função isnan() permite verificar se esse valor é um número.

O processo de leitura, e tratamento dos dados, é realizado enquanto o número de *bytes* a ler for superior a 2.

A função fscanf() permite ler os valores para uma string. Utilizando a função str2double() é realizada a conversão da string para um double com o objetivo de inserir cada um dos valores num vetor, v, de forma a elaborar a representação gráfica dos valores obtidos. Par a representação dos valores, recorre-se à utilização da função plot(v, '*') que permite criar um gráfico em "*" cujos valores são os de v.

No final desta etapa é invocada a função "drawnow" para elaborar o desenho do gráfico.

Análise de Resultados

Como foi dito anteriormente, a representação dos dados é realizada com recurso à geração de um gráfico, no MATLAB, permitindo assim a análise dos resultados obtidos.

Na figura seguinte encontra-se a representação gráfica dos primeiros 10 valores recebidos.

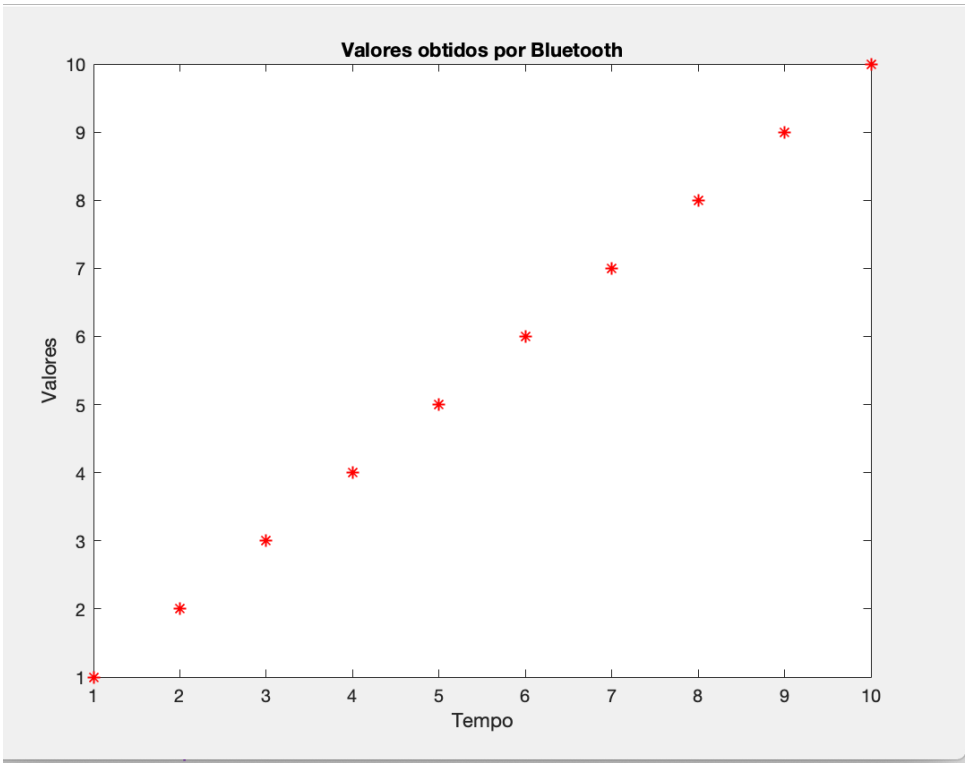

Figura 5 - Representação gráfica doa 10 primeiros valores.

À medida que são recebidos novos valores numéricos, o gráfico é atualizado adicionando assim novos valores. A atualização dos dados do gráfico pode ser verificada na figura 6.

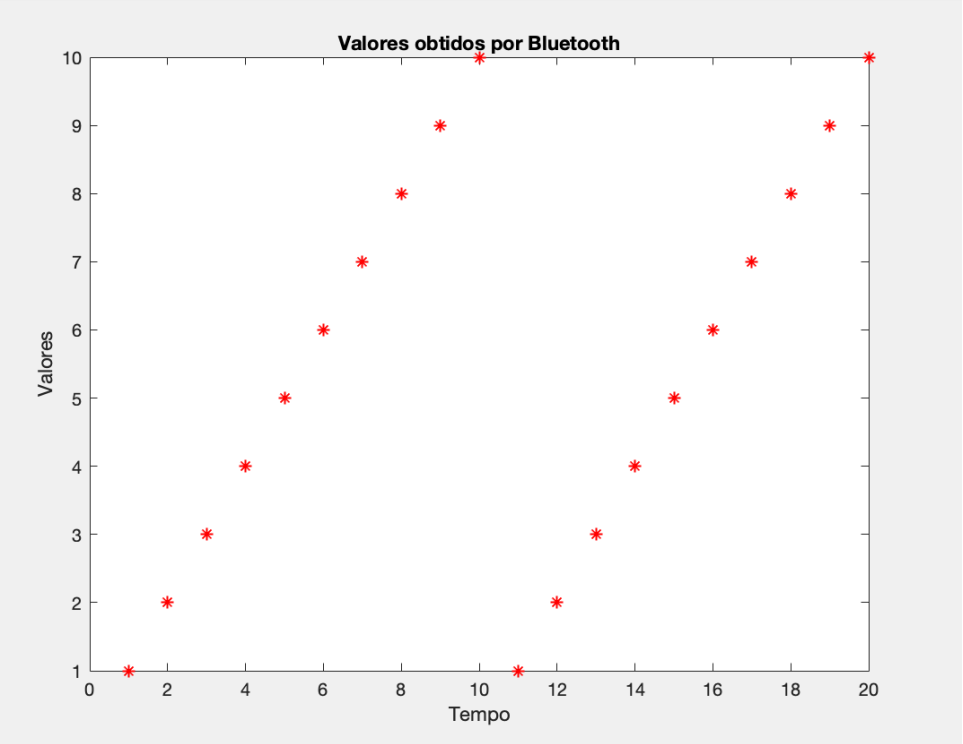

Figura 6 - Atualização da representação gráfica ao longo da receção de dados.

Ao longo do tempo, o código desenvolvido permite gerar, enviar e representar dados referentes a 10 valores que são sempre constantes. Analisando o gráfico anterior, afirmamos que a sua representação está correta, uma vez que os dados obtidos são sempre constantes, levando apenas à variação da escala temporal.

Conclusão

Durante a elaboração deste projeto, o primeiro desafio com o qual o grupo se deparou foi a configuração do módulo Bluetooth, com os parâmetros corretos (nome, pin e Baud Rate), nomeadamente o valor do Baud Rate a utilizar, devido ao facto de não conseguirmos perceber, inicialmente, que o valor a utilizar, no código Arduíno, deveria ser igual aquele com que o módulo Bluetooth tinha sido configurado.

O desenvolvimento do programa, para a receção e representação dos dados enviados pelo módulo HC06 levaram ao surgimento de novos problemas. O principal problema foi o facto de apenas um dos elementos do grupo possuir uma versão do MATLAB capaz de usar as funções de comunicação com a interface Bluetooth. Para além desta dificuldade, surgiu também o dilema da conversão dos valores obtidos, através da função “fscanf()”, para valores que pudessem ser representados graficamente.

Por fim, analisando os resultados obtidos verificamos que o objetivo final do projeto foi alcançado, conseguindo representar graficamente os dados enviados através do módulo Bluetooth HC06. Assim sendo, estamos satisfeitos com tudo o que conseguimos desenvolver no decurso deste trabalho prático.

Referências

[1] "Módulo Bluetooth HC06." [Online]. Available: https://www.botnroll.com/pt/bluetooth/2583-m-dulo-bluetooth-hc06.html.

[2] L. Guangzhou HC Information Technology Co ., "HC06 Datasheet," *GuangZhou, China*, no. 13, pp. 1–17, 2011.

[3] J. A. Langbridge, "Introduction to Arduino," *Arduino™ Sketches*, pp. 01–24, 2015.

[4] Emerson Alecrim, "Tecnologia Bluetooth: o que é e como funciona?," *@InfoWester 2001-2018*, 2018.