Estação Meteorológica Microcontrolada

Autor: Rodrigo Magalhães Caires

Resumo:

Esta estação meteorológica faz a medição de temperatura, pressão atmosférica, umidade e velocidade do vento através de um Arduino Nano e dos sensores DHT11 (umidade e temperatura), BMP180 (pressão e temperatura) e KY-025 (interruptor magnético). Os dados podem ser enviados a um computador via USB ou Bluetooth e são armazenados em txt e processados pelo MATLAB.

Projeto:

Microcontrolador, sensores e periféricos:

O microcontrolador escolhido foi o Arduino Nano e uma outra opção seria o Arduino Mini Pro. Com algumas pequenas modificações estruturais e mais algumas linhas de código, pode-se jogar ambos microcontroladores em modo de baixo consumo, sendo capazes de durar longos períodos com uma bateria. O microcontrolador precisa de dois pinos digitais de propósito comum e de pinos com as funcionalidades TX, RX, SDA e SCL fora a alimentação de 5V para todos os componentes.

A escolha dos sensores levou em conta a compatibilidade com o Arduino (conectores e bibliotecas) e o preço. Logo, os sensores DHT11, BMP180 e KY-025 foram os selecionados para este projeto.

O projeto ainda oferece a opção de um módulo Bluetooth (HC-05) e conta com um anemômetro, dispositivo que quando somado a um ímã e ao sensor KY-025 fornecerá uma estimativa da velocidade do vento.

A melhor opção para o anemômetro seria uma construção caseira em razão do custo. Um ímã em um dos copos do anemômetro alinhado com o interruptor magnético fará a marcação dos giros do anemômetro, sendo este um parâmetro fundamental no cálculo da velocidade como visto mais a frente.

Funcionamento do código:

O código do Arduino, apesar de pronto, necessita ser ajustado de acordo com o projeto antes de ser carregado na placa. Primeiramente, deve-se decidir se a transmissão de dados será via USB ou Bluetooth. Se optado pela conexão à cabo, descomentar as linhas 123 e 133 e, se optado pela conexão WiFi, descomentar as linhas 46, 112 e 120.

A segunda modificação a ser feita no código se apresenta na linha 27. Para uma leitura correta da velocidade do ar, é necessário que se saiba o raio do anemômetro, pois a mesma é definida como sendo (Voltas*2*π*Raio)/Intervalo.

Além da leitura e envio de dados, o código também pré-processa os valores obtidos por meio de um filtro de Kalman que reduz consideravelmente o ruído. A

leitura da temperatura também é feita pelos dois sensores (DHT11 e BMP180), somados e divididos por 2 para uma precisão maior. O código se repete em períodos de 5 segundos.

Recebimento de dados no computador:

Como mencionado anteriormente, este projeto é capaz de transmitir dados tanto via USB, quanto Bluetooth.

A tecnologia do Arduino não permite que o microcontrolador, e nem seu compilador, acesse certas funções do computador como a criação, modificação ou leitura de arquivos. Para tanto, utiliza-se o software PuTTY (outra opção sendo o GoBetwino) que acessa a porta de comunicação conectada ao Arduino ou Bluetooth e escreve as informações recebidas em um terminal. O PuTTY também é capaz de armazenar as sessões em um arquivo de formato livre (txt, dat, etc.).

Os procedimentos para cada um dos métodos é explicado abaixo:

Primeiramente, é necessário encontrar a porta onde se conecta o Arduino. Para isso, deve-se acessar o Gerenciador de Dispositivos > Portas (COM e LPT):

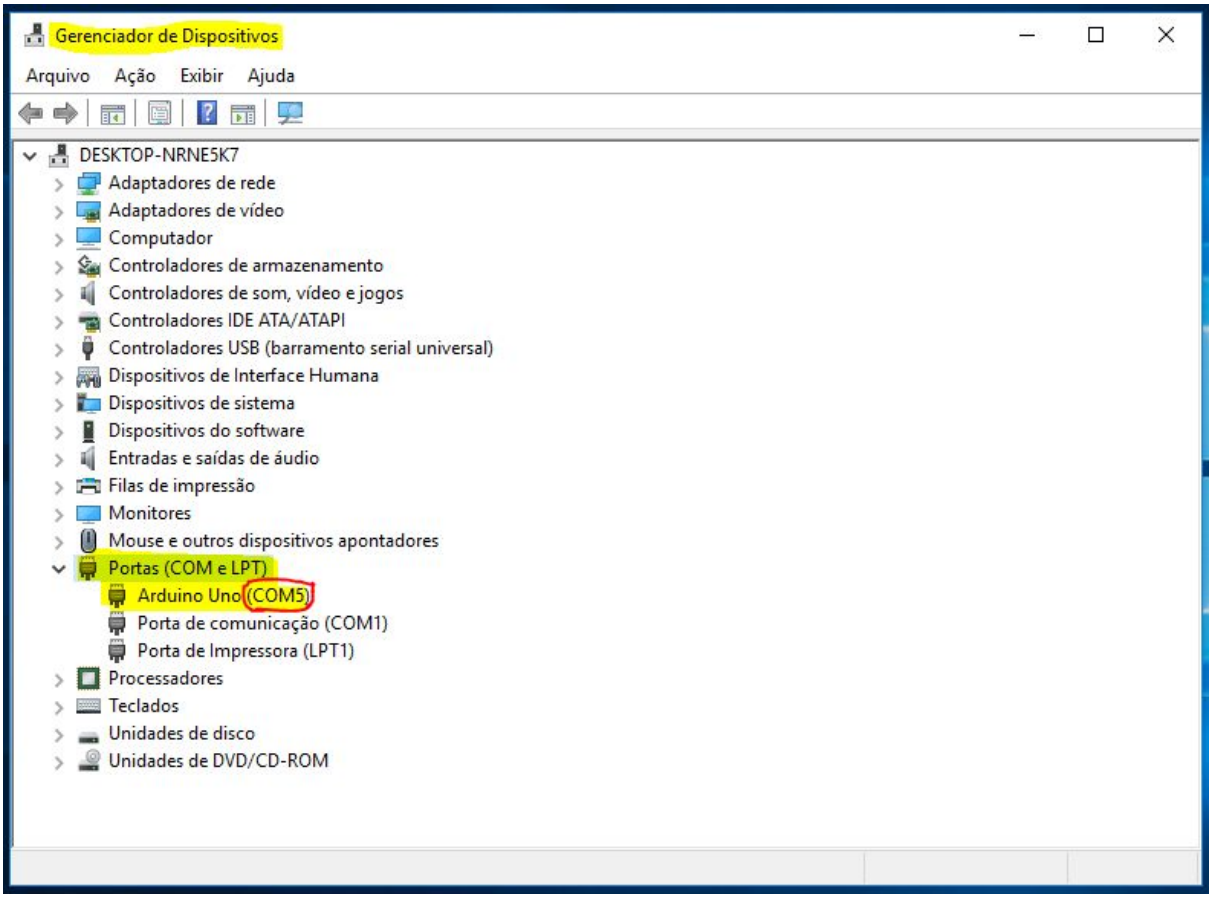

Figura 1. Encontrando canal de comunicação via USB.

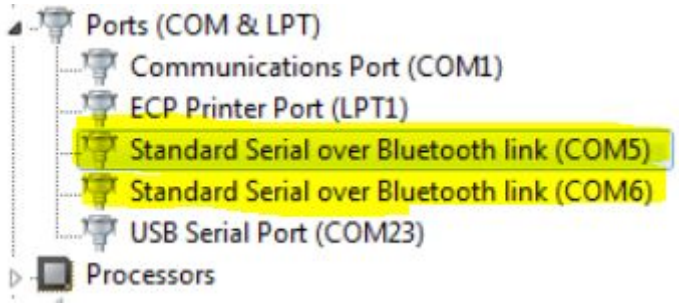

Figura 2. Encontrando canal de comunicação via Bluetooth.

No caso do Bluetooth, duas portas de nome igual aparecerão no lugar do Arduino, uma sendo a TX e a outra a RX. É necessário testa-lás no PuTTY enviando qualquer informação para descobrir qual é a RX. Se for a RX, os dados enviados serão escritos no terminal e se for a TX nada acontecerá.

Em seguida, configura-se o PuTTY para observar a porta encontrada:

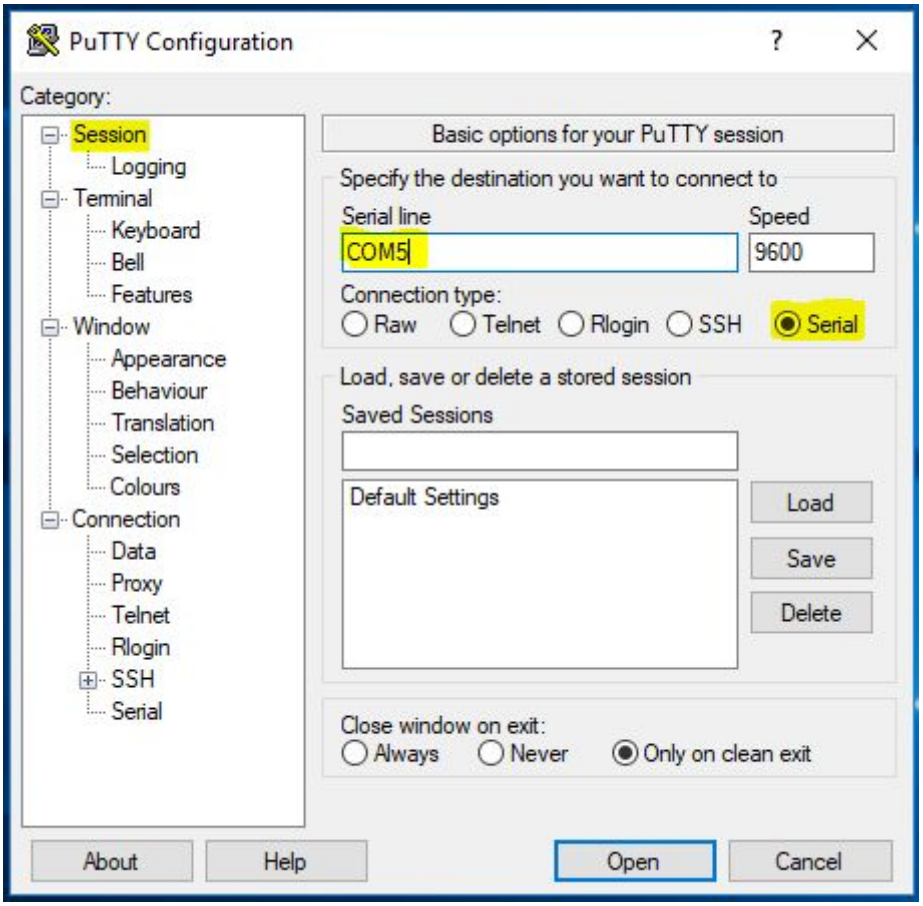

Figura 3. Configurando a conexão do PuTTY.

Por fim, configura-se o armazenamento dos dados. Antes, criar o arquivo WeatherData.txt:

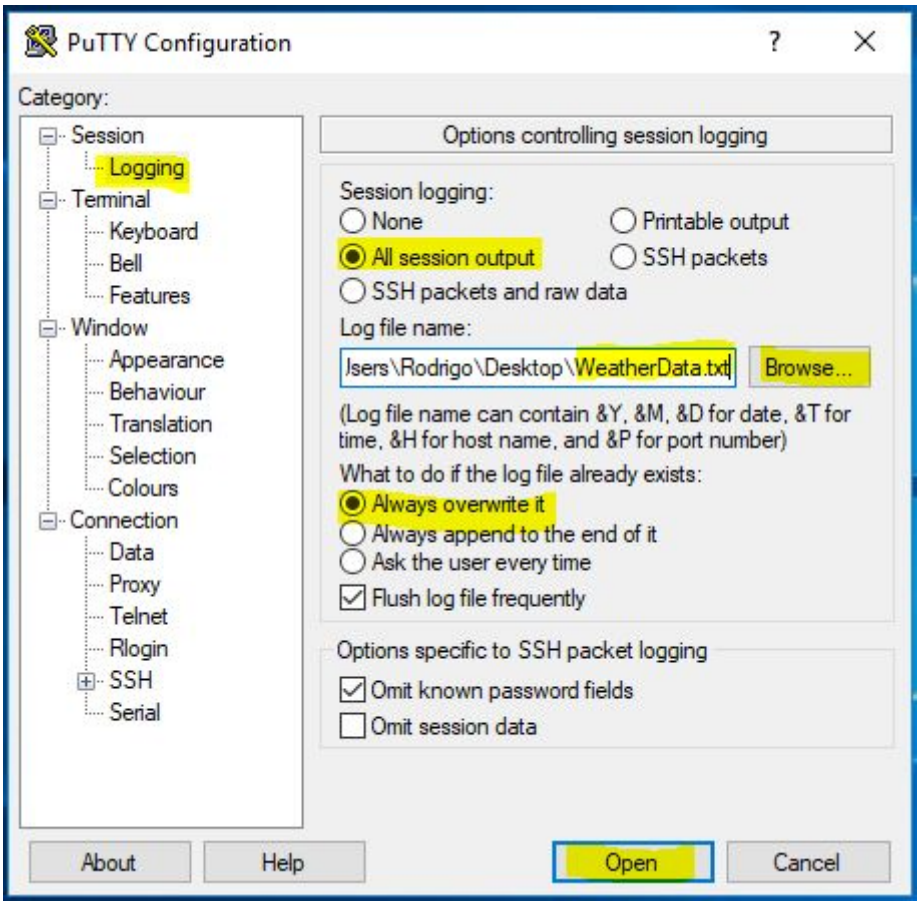

Figura 4. Configurando o armazenamento de dados.

A preferência é pela opção de overwrite ao invés de append pois sempre que o PuTTY abre um arquivo para escrita, a primeira linha anotada contém um log com a data de entrada e pode interferir com o processamento dos dados pelo MATLAB.

Processamento dos dados:

O arquivo texto gerado no procedimento anterior é aberto no MATLAB. Devido à algumas dificuldades encontradas durante a programação, o arquivo não é lido todo de uma vez, mas uma linha a cada 5.2 segundos. Isso dá uma margem para o Arduino atualizar os dados, que são salvos instantaneamente no txt e, deste modo, o programa MATLAB pode continuar funcionando sem causar erro.

A interface plota os 12 últimos valores lidos (último minuto) com um gráfico para cada parâmetro e ainda fornece o valor exato do instante atual numa linha de texto como demonstrado abaixo:

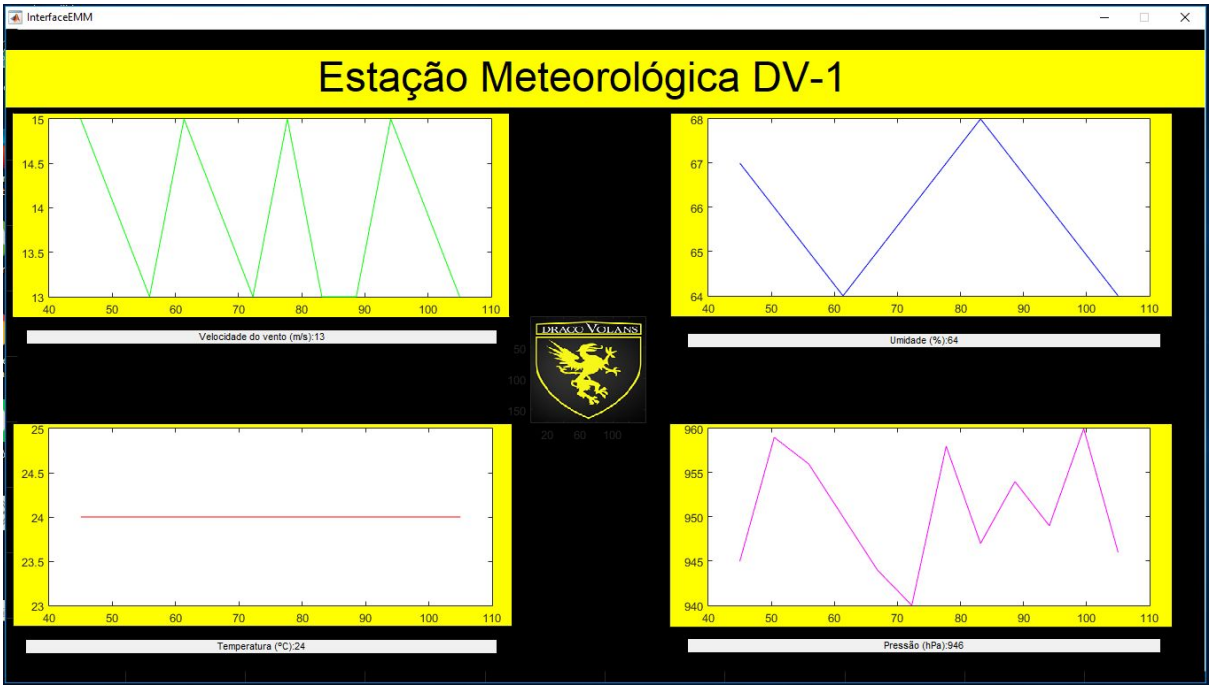

Figura 5. Interface EMM.

Esquemáticos:

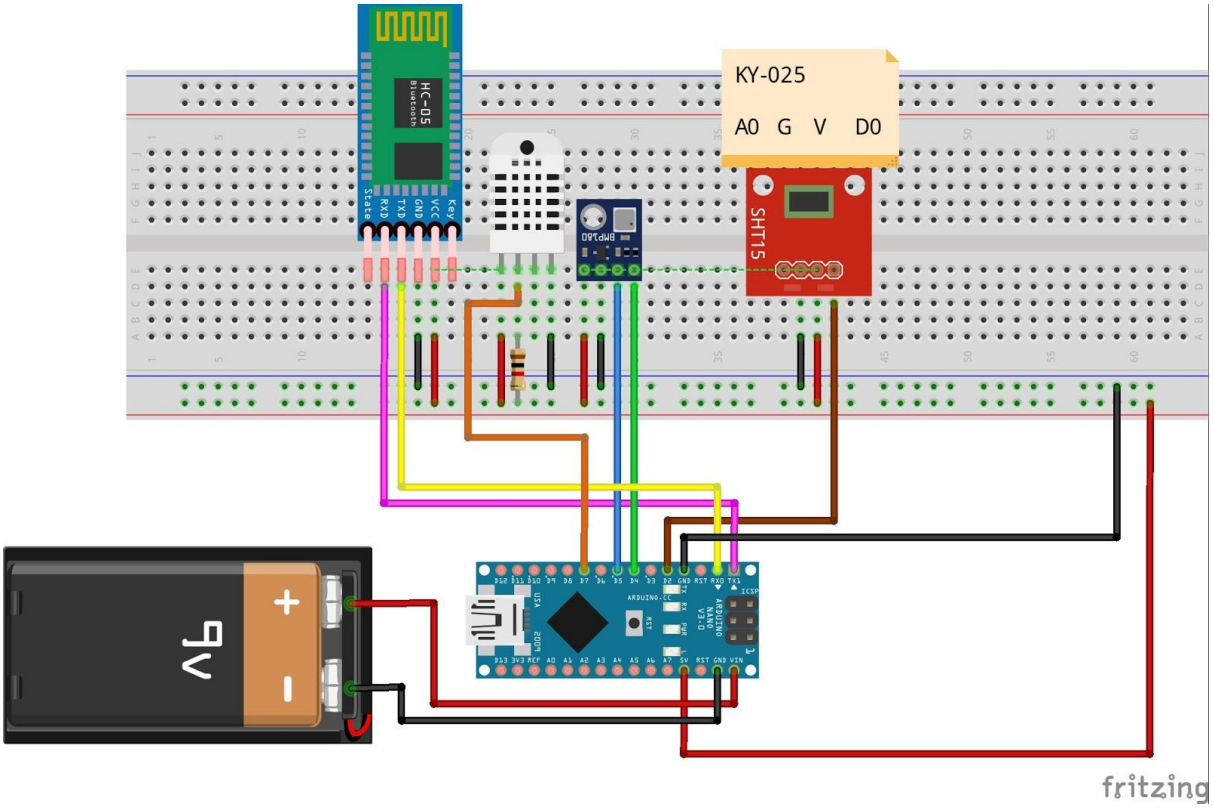

Figura 6. Esquemático do circuito.

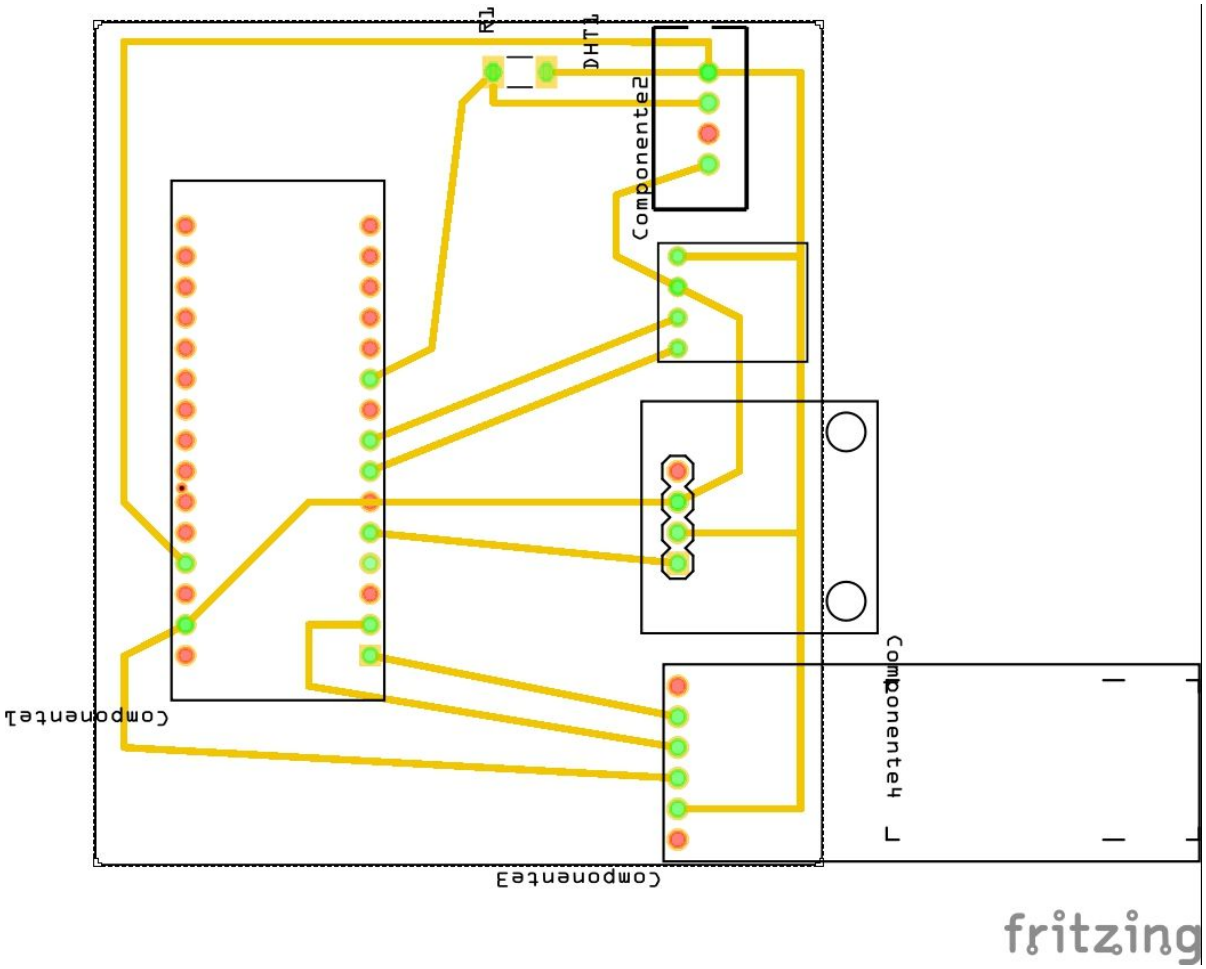

Figura 7. Esquemático da PCB.

Códigos:

https://github.com/RodrigoMac/Draco-Volans/tree/master/Estacao%20Meteorologica%20Microcontrolada

Referências:

Arduino Low Power :
http://www.home-automation-community.com/arduino-low-power-how-to-run-atmega328p-for-a-year-on-coin-cell-battery/
Estação Meteorológica com Arduino:
https://www.filipeflop.com/blog/estacao-meteorologica-com-arduino/
Como fazer uma mini estação meteorológica com Arduino:
http://www.artilhariadigital.com/2015/10/Como-fazer-uma-mini-estacao-meteorologica-com-arduino.html