FEELT31109 - Enriquecimento Instrumental
Roteiro Lab.06

Prof. Igor Peretta

21-Maio-2019

1 Introdução

1.1 Serial Monitor

Você pode usar o *Serial Monitor* do Arduino IDE para "debugar" seus códigos ou para visualizar os dados enviados por um código em execução. Você deve ter uma placa conectada por USB ao seu computador para poder ativar o *Serial Monitor*.

O *Serial Monitor* (Figura 1) é uma janela *pop-up* separada (acesse via `Tools/Serial Monitor` ou `CTRL+SHIFT+M`) que atua como um terminal separado que se comunica recebendo e enviando dados seriais. Dados seriais são enviados através de um único fio (através do USB no nosso caso) e consiste em uma série de 1's e 0's enviados através do fio. Os dados podem ser enviados em ambas as direções.

1.2 Multiplex

Texto adaptado de `https://en.wikipedia.org/wiki/Multiplexing`.

Nas redes de telecomunicações e de computadores, a multiplexação (às vezes contratada para muxing) é um método pelo qual múltiplos sinais analógicos ou digitais são combinados em um sinal em um meio compartilhado. O objetivo é compartilhar um recurso escasso. Por exemplo, nas telecomunicações, várias chamadas telefônicas podem ser realizadas usando um fio. A multiplexação originou-se na telegrafia nos anos 1870 e é agora amplamente aplicada nas comunicações. Na telefonia, George Owen Squier é creditado com o desenvolvimento da multiplexação de operadoras de telefonia em 1910.

O sinal multiplexado é transmitido através de um canal de comunicação, como um cabo. A multiplexação divide a capacidade do canal de comunicação em vários canais lógicos, um para cada sinal de mensagem ou fluxo de dados a ser transferido. Um processo inverso, conhecido como demultiplexação, extrai os canais originais no final do receptor.

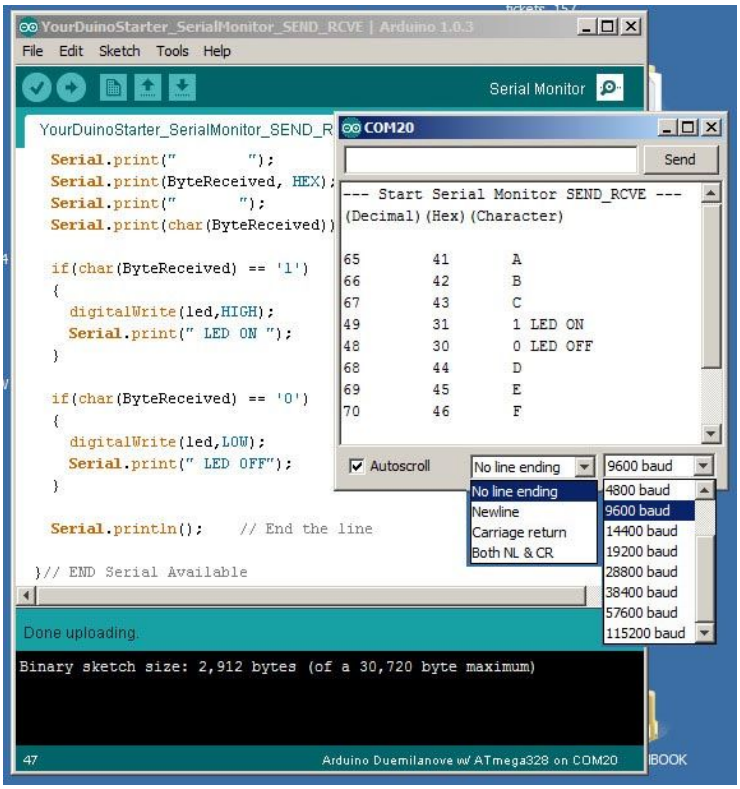

Figura 1: Monitor serial do Arduino IDE (fonte: `https://www.instructables.com/id/HOW-TO-use-the-ARDUINO-SERIAL-MONITOR/`).

Um dispositivo que executa a multiplexação é chamado de multiplexador (MUX), e um dispositivo que executa o processo inverso é chamado de demultiplexador (DEMUX ou DMX). Veja Figura 2.

Na computação, a multiplexação de E/S também pode ser usada para se referir ao conceito de processar múltiplos eventos de entrada/saída a partir de um único laço de eventos.

1.3 DHT11

Texto de:
`https://www.filipeflop.com/blog/monitorando-temperatura-e-umidade-com-o-sensor-dht11/`

Especificações:

- Modelo: DHT11 (Datasheet: `http://img.filipeflop.com/files/download/Datasheet_DHT11.pdf`)
- Alimentação: 3,0 a 5,0 VDC (5,5 Vdc máximo)

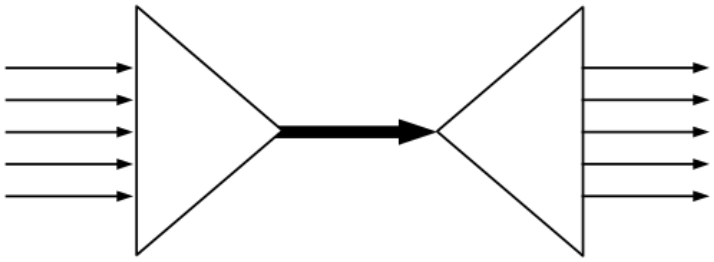

Figura 2: Múltiplos sinais de baixa taxa de dados são multiplexados em um único link de alta taxa de dados e, em seguida, desmultiplexados na outra extremidade (fonte: `https://en.wikipedia.org/wiki/Multiplexing#/media/File:Multiplexing_diagram.svg`).

- Corrente: 200uA a 500mA, em stand by de 100uA a 150 uA
- Faixa de medição de umidade: 20 a 90
- Faixa de medição de temperatura: 0º a 50ºC
- Precisão de umidade de medição: 5,0
- Precisão de medição de temperatura: 2.0 ºC
- Tempo de resposta: ¡ 5s
- Dimensões: 23mm x 12mm x 5mm (incluindo terminais)

Para facilitar o seu trabalho já existe uma bilioteca que pode ser baixada neste link: `https://github.com/adafruit/DHT-sensor-library`. Após o download descompacte o arquivo `.zip` e mova-o para a pasta `arduinosketchfolder/libraries/` e reinicie a IDE do Arduino. Não retire o arquivo `dht.cpp` e não esqueça de renomear a pasta para "DHT". Talvez será necessário criar uma sub-pasta da biblioteca caso não exista.

Este sensor inclui um componente medidor de umidade e um componente NTC para temperatura, ambos conectados a um controlador de 8-bits. O interessante neste componente é o protocolo usado para transferir dados entre o MCDU e DHT11, pois as leituras do sensor são enviadas usando apena um único fio de barramento.

Formato dos dados: 8bit integral RH data + 8bit decimal RH data + 8bit integral T data + 8bit decimal T data + 8bit check sum = 40 bits.

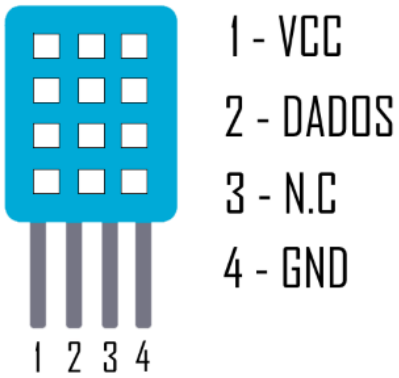

Figura 3: Pinagem DHT11 (fonte: `https://www.filipeflop.com/blog/monitorando-temperatura-e-umidade-com-o-sensor-dht11/`).

O DHT11 possui 4 terminais (Figura 3) sendo que somente 3 são usados: GND, VCC e Dados. Se desejar, pode-se adicionar um resistor pull up de 10K entre o VCC e o pino de dados.

A leitura da temperatura e umidade pode levar 250ms. O atraso do sensor pode chegar a 2 segundos.

2 Experimento 6A: Vários sensores analógicos

Este projeto foi adaptado de:
`https://www.instructables.com/id/ESP8266-with-Multiple-Analog-Sensors/`

O projeto consiste em usar vários sensores analógicos na única entrada analógica do NodeMCU ESP8266.

Embora o NodeMCU ESP8266 tenha apenas um pino ADC (Conversor Analógico Digital), isso não significa que você está limitado a apenas um sensor analógico por módulo. Você pode usar muitos!

No entanto, para usar vários sensores, você precisará "multiplexar"os sensores. Multiplexação significa simplesmente que você ligará um sensor, lerá o sensor, desligará o sensor e então passará para o próximo sensor.

O esquemático de montagem se encontra na Figura 4.

2.1 Material

- 2x Resistor 10k Ω
- 2x Diodos 1N4007
- 1x Sensor Umidade/Temperatura DHT11

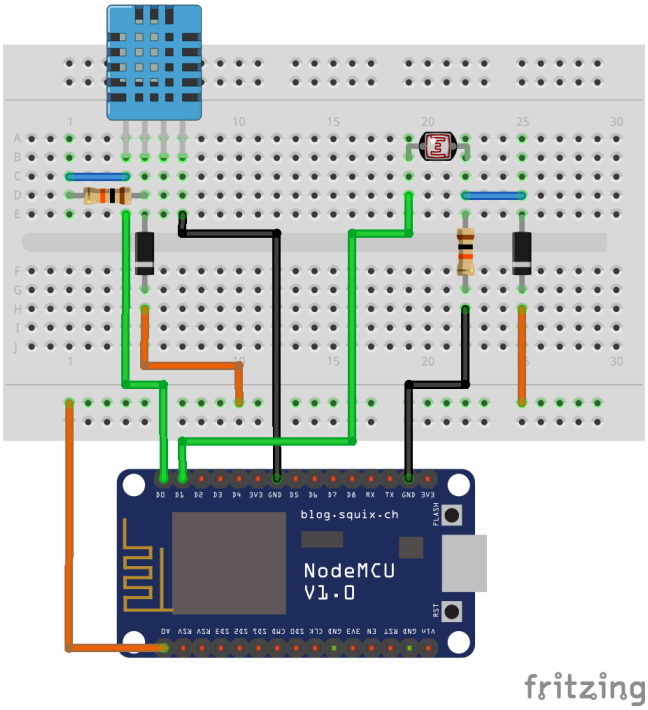

Figura 4: Esquemático para múltiplos sensores

- 1x Sensor de luminosidade LDR

Mais: *Protoboard*, *jumpers*, placa **NodeMCU ESP8266**

2.2 Código

Código-Fonte 1: Código-fonte para acionamento do servomotor

```
1  #include "DHT.h"
2
3  #define DHTPIN A0
4  #define LDRPIN A0
5  #define DHTTYPE DHT11
6  #define N 100
7
8  int ctrlDHT = D0;
9  int ctrlLDR = D1;
10
11 DHT dht(DHTPIN, DHTTYPE);
12
```

```
13 void setup()
14 {
15     Serial.begin(9600);
16     Serial.println("Sensores Umidade, Temperatura, Luminosidade");
17     dht.begin();
18     pinMode(ctrlDHT, OUTPUT);
19     pinMode(ctrlLDR, OUTPUT);
20 }
21
22 void loop()
23 {
24     static float h = 0.0;
25     static float t = 0.0;
26     static int lum = 0;
27     static int i = 0;
28
29     digitalWrite(DHTPIN, HIGH);
30     digitalWrite(LDRPIN, LOW);
31     h = dht.readHumidity();
32     t = dht.readTemperature();
33     if (isnan(t) || isnan(h))
34     {
35         Serial.println("Falha de leitura do DHT");
36     }
37     else
38     {
39         Serial.print("Umidade: ");
40         Serial.print(h);
41         Serial.print(" %t");
42         Serial.print("Temperatura: ");
43         Serial.print(t);
44         Serial.println(" *C");
45     }
46
47     digitalWrite(DHTPIN, LOW);
48     digitalWrite(LDRPIN, HIGH);
49     lum = 0
50     for(i = 0; i < N; i++)
51         lum += analogRead(LDRPIN);
52     Serial.print("Luminosidade: ");
53     Serial.println(lum/N);
54
55 }
```

Para saber o resultado do seu experimento, você deve acompanhar as saídas impressas via *Serial Monitor*.

3 (Desafio) Experimento 6B: ESP8266 VGA Pong

Faça o projeto em:
`https://www.instructables.com/id/ESP8266-VGA-Pong/`

4 Relatório

Redigir um relatório em LaTeX (de 1 a 3 páginas) contendo, para cada projeto:

- Identificação da disciplina e do laboratório
- Identificação do(s) discente(s)
- Breve descrição do que era esperado
- Relato das montagens e das dificuldades encontradas
- Relato dos resultados obtidos
- (Opcional) Descrição das modificações feitas ao(s) projeto(s) e resultados obtidos a partir das mesmas

Publicar em PDF e enviar anexo para o e-mail: `iperetta@ufu.br` com o assunto: **FEELT31109 2019-1 Relatório do Lab XX**, onde **XX** é o número identificador deste laboratório.

Dica: o site `http://www.overleaf.com/` é um dos sites em que se pode editar documentos LaTeX online, além de permitir a colaboração de vários autores para o documento.