

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.

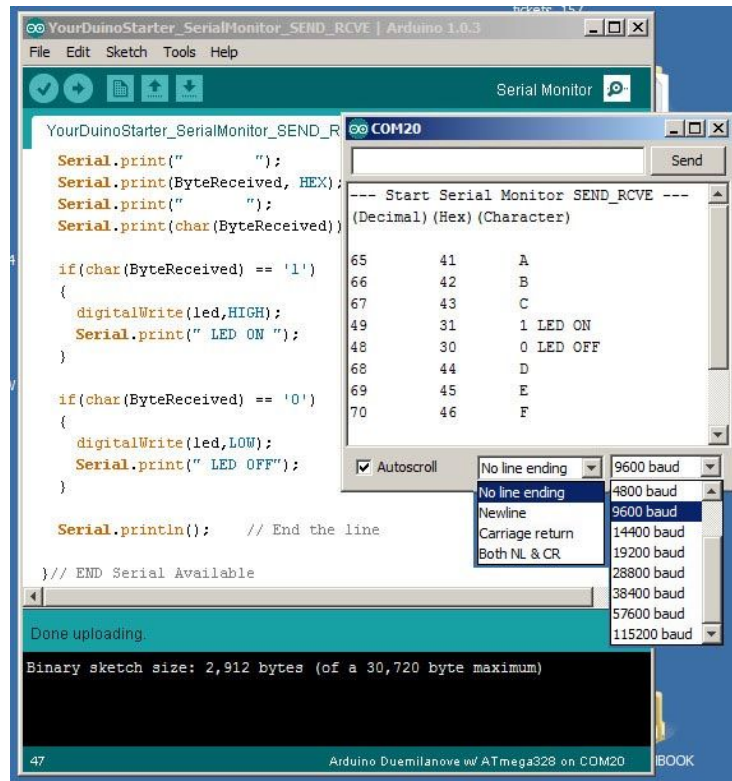


Figura 1: Monitor serial do Arduino IDE (fonte: <https://www.instructables.com/id/HOW-TO-use-the-ARDUINO-SERIAL-MONITOR/>).

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/>

1.3.1 Especificações:

- Modelo: DHT11 (Datasheet: http://img.filipeflop.com/files/download/Datasheet_DHT11.pdf)

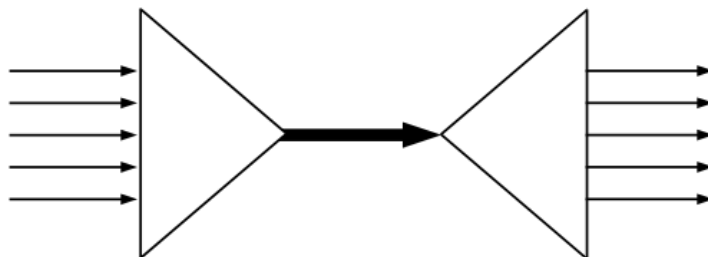


Figura 2: Múltiplos sinais de baixa taxa de dados são multiplexados em um único link de alta taxa de dados e, em seguida, desmultiplexados na outra extremidade (fonte: https://en.wikipedia.org/wiki/Multiplexing#/media/File:Multiplexing_diagram.svg).

- Alimentação: 3,0 a 5,0 VDC (5,5 Vdc máximo)
- 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: 0° a 50°C
- Precisão de umidade de medição: 5,0
- Precisão de medição de temperatura: 2.0 °C
- Tempo de resposta: \leq 5s
- Dimensões: 23mm x 12mm x 5mm (incluindo terminais)

Para facilitar o seu trabalho já existem bibliotecas que pode ser instalada. Duas opções:

Preferível Na aba *Tools* do Arduino IDE vá em *Manage Libraries...* e procure por e instale a **DHT sensor library** da **Adafruit**. Depois, procure por e instale a **Adafruit Unified Sensor**.

Alternativa 1 Na aba *Tools* do Arduino IDE vá em *Manage Libraries...* e procure por e instale a **DHT sensor library for ESPx** de **beegee_tokyo**. Veja exemplo de uso aqui:

https://github.com/beegee-tokyo/DHTesp/blob/master/examples/DHT_ESP8266/DHT_ESP8266.ino

Alternativa 2 Baixar deste link:

<https://github.com/adafruit/DHT-sensor-library>.

Após o download descompacte o arquivo .zip e mova-o para a pasta `libraries/` do arduino e reinicie a IDE. 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. Instale também a biblioteca `Adafruit_Sensor.h` daqui:

https://github.com/adafruit/Adafruit_Sensor

1.3.2 Detalhes

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 MCU e DHT11, pois as leituras do sensor são enviadas usando apenas 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.

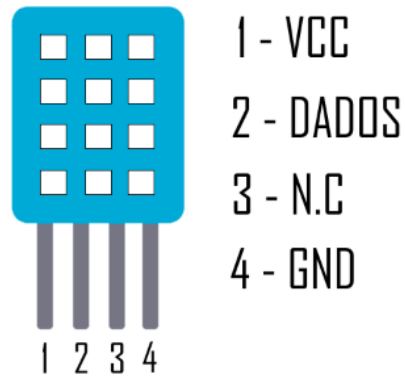


Figura 3: Pinagem DHT11 (fonte: <https://www.filipeflop.com/blog/monitorando-temperatura-e-umidade-com-o-sensor-dht11/>).

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.

Observe que, dos sensores utilizados, iremos multiplexar o potenciômetro e o LDR. O DHT11 é um sensor com leitura digital (serial) e, portanto, não se aplica à porta analógica.

O esquemático de montagem se encontra na Figura 4.

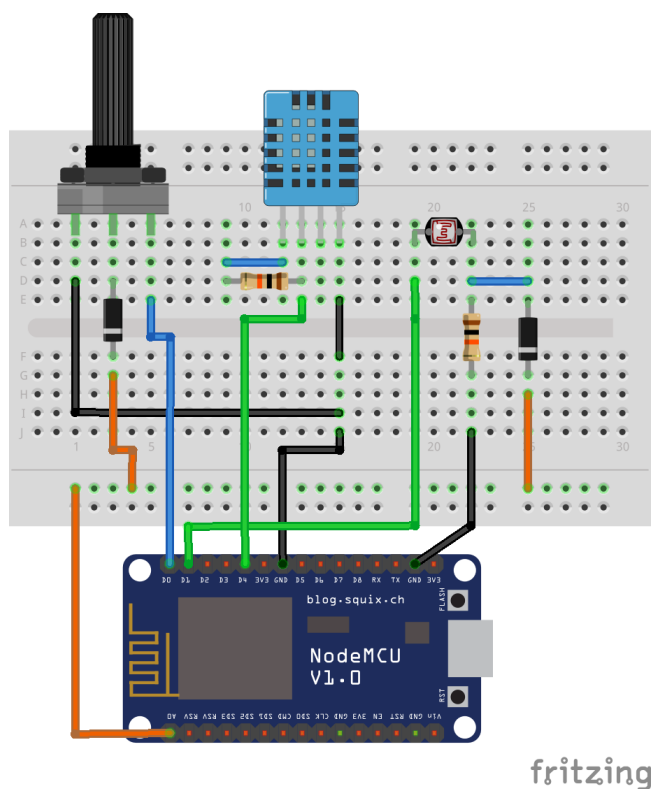


Figura 4: Esquemático para múltiplos sensores

2.1 Material

- 1x Potenciômetro 10k Ω
- 2x Resistores 10k Ω
- 2x Diodos 1N4007
- 1x Sensor Umidade/Temperatura DHT11
- 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 POTPIN A0
4 #define LDRPIN A0
5 #define DHTPIN D4
6 #define DHTTYPE DHT11
7 #define N 100
8
9 int ctrlPOT = D0;
10 int ctrlLDR = D1;
11
12 DHT dht(DHTPIN, DHTTYPE);
13
14 void setup()
15 {
16     pinMode(ctrlPOT, OUTPUT);
17     pinMode(ctrlLDR, OUTPUT);
18     Serial.begin(9600);
19     Serial.println("Sensores");
20     dht.begin();
21 }
22
23 void loop()
24 {
25     static float h = 0.0;
26     static float t = 0.0;
27     static int lum = 0;
28     static int res = 0;
29     static int i = 0;
30
```

```

31     digitalWrite(ctrlPOT, LOW);
32     digitalWrite(ctrlLDR, LOW);
33     h = dht.readHumidity();
34     t = dht.readTemperature();
35     if (isnan(t) || isnan(h))
36     {
37         Serial.print(t);
38         Serial.println("Falha de leitura do DHT");
39     }
40     else
41     {
42         Serial.print("Umidade: ");
43         Serial.print(h);
44         Serial.print("\t");
45         Serial.print("Temperatura: ");
46         Serial.print(t);
47         Serial.println(" °C");
48     }
49     delay(500);
50
51     digitalWrite(ctrlPOT, LOW);
52     digitalWrite(ctrlLDR, HIGH);
53     lum = 0;
54     for(i = 0; i < N; i++)
55         lum += analogRead(LDRPIN);
56     Serial.print("Luminosidade: ");
57     Serial.println(lum/N);
58     delay(500);
59
60     digitalWrite(ctrlPOT, HIGH);
61     digitalWrite(ctrlLDR, LOW);
62     Serial.print("Resistencia: ");
63     res = map(analogRead(POTPIN), 0, 1023, 0, 10000);
64     Serial.print(res);
65     Serial.println(" ohms");
66     delay(500);
67 }

```

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 L^AT_EX (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 L^AT_EX online, além de permitir a colaboração de vários autores para o documento.