

PROFESSOR DANILO

ROBÓTICA – 9 ANO – 17/03/2022

**ENTRADA ANALÓGICA
AULA 04**

MONTAGEM DO CIRCUITO

Nesta aula vamos ver como fazer para o Arduino se comunicar com o computador enviando informações para ele. Monte o circuito apresentado na figura a seguir:

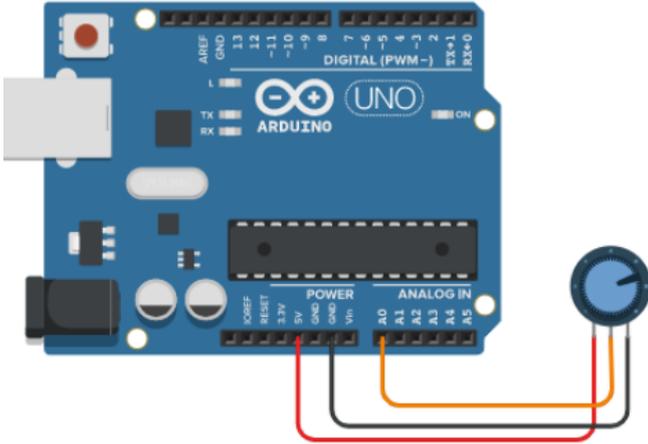


Figura 1: Circuito para entrada analógica

Nessa altura, você já deve ser capaz de montar se circuito mesmo sem ter a placa de ensaio no esquema. Note que num dos terminais do potenciômetro (da esquerda) você conecta na porta 5V e no terminal do outro lado conecta-se no GND. A porta central você conecta na porta analógica A0.

SKETCH

```
void setup() {
  Serial.begin(9600);
}

void loop() {
  int x = 0;
  float y = 0;
  x = analogRead(A0);
  y = map(x, 0, 1023, 0, 500);
  y = y / 100;
  Serial.print("O valor de x é: ");
  Serial.println(x);
  Serial.print("O valor de y é: ");
  Serial.println(y);
  Serial.println("");
  delay(500);
}
```

DISCUSSÃO DO PROGRAMA

Na configuração, vemos uma nova função: Serial.begin(9600).

```
void setup() {
  Serial.begin(9600);
}
```

Esta função serve para iniciar a comunicação entre o Arduino e o computador e o número que vai entre parêntesis é a velocidade de comunicação usada (no caso, 9600 bits por segundo).

```
void setup() {
  Serial.begin(9600);
}
```

Lembre-se que "begin" significa iniciar. No loop, vemos como se declaram as variáveis

```
void loop() {
  int x = 0;
  float y = 0;
```

Aqui, usamos o tipo float. Quando queremos salvar algum dado no computador, este deve saber que tipo de dado está sendo salvo. No caso se números inteiros, usamos `int x = 0;` ou seja, dizemos que a variável `x` somente aceita números inteiros. O tipo `float` aceita números não inteiros, isto é, com vírgulas. Em computação, dizemos que estes números são tipo pontos flutuantes.

No código acima, para trabalharmos com números com algarismos após a vírgula, usamos o tipo float.

Continuando com nosso programa, vemos a função que lê a porta analógica:

```
x = analogRead(A0);
```

Do inglês, "analogical" – analógico, e "read" – leitura. Informa ao Arduino para ler a porta analógica A0 e salva este valor na variável `x`. Conforme você pode ver no vídeo, disponível no site do professor, o valor de entrada nesta porta varia de 0 até 1023, sendo um dado do tipo inteiro. Por isso podemos declarar a variável `x` como inteira.

A função `map` funciona como se fosse uma regra de três. Ela necessita de três argumentos: `map(valor, valor_mínimo, valor_máximo, novo_valor_mínimo, novo_valor_máximo)`.

`valor` é a variável que será lida; `valor_mínimo` é o menor valor que a variável `valor` pode assumir enquanto que `valor_máximo` é o maior valor que ela pode assumir; `novo_valor_mínimo` e `novo_valor_máximo` são os valores mínimos e máximos que a variável `y` receberá. Veja como exemplo o código abaixo:

```
y = map(x, 0, 1023, 0, 500);
```

Imagine que o Arduino lê a porta analógica de modo que `x` seja 0, então `y` vale 0; se `x` valer 1023, então `y` valerá 500; se `x` valer 511, então `y` valerá 250, e assim por diante.

Na linha seguinte, dividimos o valor de `y` por 100 e salvamos novamente na variável `y`.

```
y = y / 100;
```

PROFESSOR DANILO

Note que aqui é diferente da matemática, tal como vemos na sala de aula, por esta razão dizemos que o símbolo “=” é um símbolo de atribuição, ou seja, lemos o seguinte no código acima:

“y recebe o valor de y dividido por 100.”

Se você quiser que o Arduino envie um dado para o PC, você pode usar a função **Serial.print()**.

```
Serial.print("O valor de x é: ");
```

No exemplo acima, a função **Serial.print()** não pula uma linha na tela.

Para ver o que o Arduino envia, você deve abrir o serial monitor clicando no ícone representado abaixo.

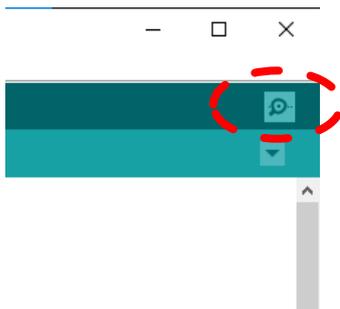


Figura 2: Clicando no botão detalhado acima, você entra no serial monitor.

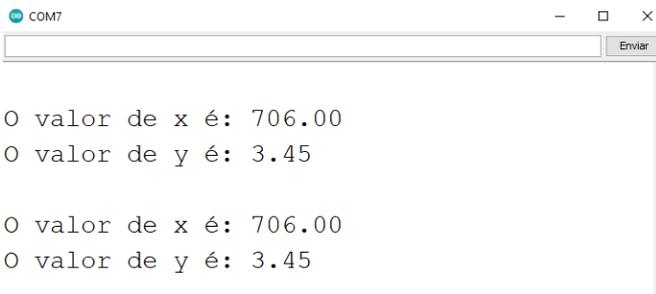


Figura 3: Detalhe do monitor serial.

Você também pode entrar no serial monitor pelo menu superior.

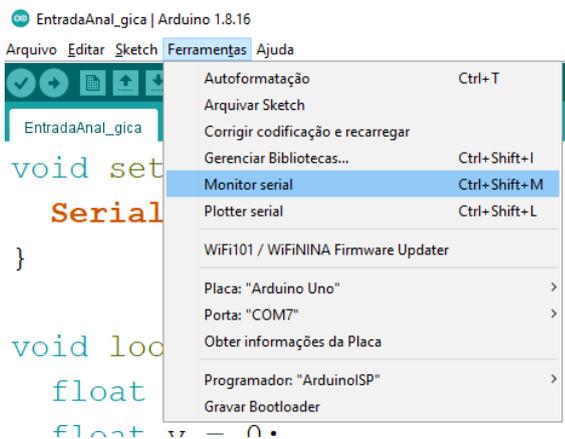


Figura 4: Para acessar o monitor serial você pode usar o atalho Ctrl + Shift + M ou ainda clicar em Ferramentas > Monitor serial.

Se você precisar imprimir uma informação no monitor serial com um enter no final da linha (isto é, que o cursor – onde o computador irá imprimir o próximo caractere – continuará na próxima linha) então você usa a função **Serial.println()** (note um “ln” – “ele ene” – no final da função).

ROBÓTICA – 9 ANO – 17/03/2022

```
Serial.println(x);
```

Neste caso, ele irá imprimir no monitor serial o valor da variável x, que é o valor lido na porta analógica.

Por fim, vejamos as três outras linhas com a função print:

```
Serial.print("O valor de y é: ");
Serial.println(y);
Serial.println("");
```

Na primeira linha, informa que o Arduino deve enviar a frase entre aspas (“O valor de y é: ”), incluindo o espaço, para o computador exibir no monitor serial, sem pular linha no final. Na linha seguinte, o Arduino envia o valor de y, valor da tensão medida pelo Arduino na porta A0, ao moitor serial imprimindo-o com a informação para ir para a linha seguinte (ln). Por fim, colocamos novamente **Serial.println(“”)**, agora com nada entre aspas, para que seja pulada uma linha na e assim os dados fiquem mais organizados.

No final pedimos ao Arduino que ele espera por 500 milissegundos, isto é, por meio segundo, até então voltas ao início do código.

```
delay(500);
}
```

Por fim, você pode acessar o plotter serial:

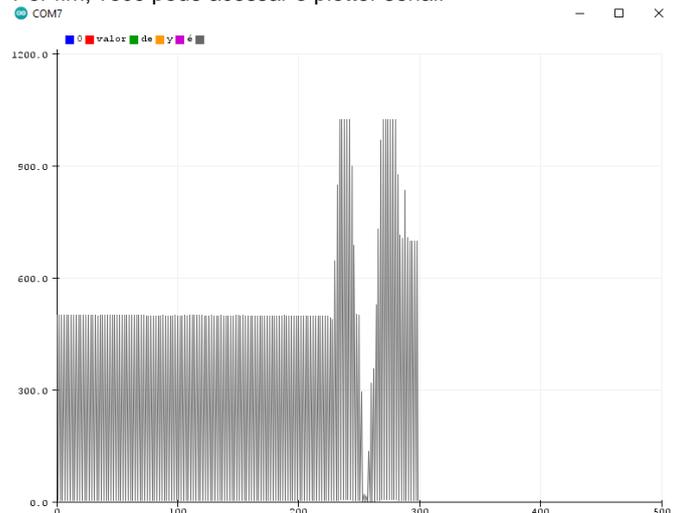


Figura 5: detalhe do plotter serial.

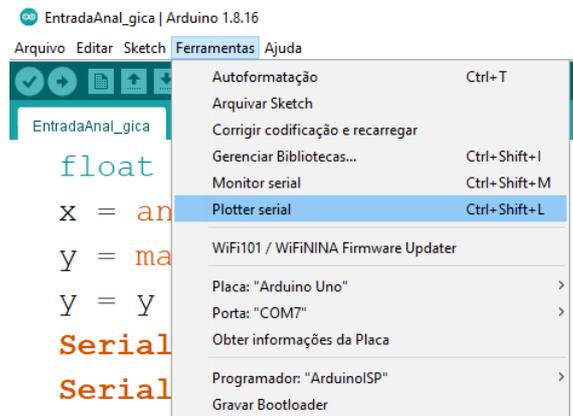


Figura 6: Para acessar o plotter serial você pode usar o atalho Ctrl + Shift + L ou clicando em Ferramentas > Plotter serial.

Finalizamos o assunto desta aula. Na próxima aula vamos entender mais algumas funções da linguagem C.

PROFESSOR DANILO

ROBÓTICA – 9 ANO – 17/03/2022

TABELA VERDADE

Acredite, esse é um assunto bem intuitivo e você pode tentar entender isso brincando. Seu professor, e alguns colegas, criaram um site bem simples no qual você pode aprender um pouco de lógica de programação, mais precisamente, o que são conectivos lógicos, na forma de um jogo. Para ver como é, basta acessar o site abaixo ou ler o QR-code a seguir.

<https://logicando.herokuapp.com/Apresentacao>



Figura 7: Site onde você pode ter alguma ideia sobre lógica de programação na forma de um jogo.

Legal, mas o que é tabela verdade e o que são conectivos lógicos?

Basicamente, conectivos lógicos são símbolos ou palavras que conectam duas ou mais sentenças resultando em apenas dois possíveis resultados: **verdadeiro** ou **falso**.

Em linguagem de programação, temos os seguintes conectivos e seus respectivos símbolos na linguagem C++ (a que estamos usando para controlar o Arduino):

- E: &&
- OU: ||
- NÃO: !

Vamos ver então como usá-los de forma simples:

(O céu é azul) E (A lousa é verde) = VERDADEIRO
(Borboletas voam) OU (Gatos vivem no mar) = VERDADEIRO
(Borboletas voam) E (Cachorros vivem no mar) = FALSO

Agora podemos construir a tabela verdade. Faremos isso usando uma afirmação A, que pode ser verdadeira ou falsa (usaremos V ou F, respectivamente), e uma afirmação B, que também pode ser V ou F. Vamos utilizar os símbolos usados na programação do Arduino.

Tabela 1: tabela verdade. Procure na internet sobre o assunto. Seu professor fez um vídeo para te ajudar: consulte na página da disciplina.

A	B	A && B	A B	!A	!B
V	V				
V	F				
F	V				
F	F				

Pegou a ideia?

Supondo que não haja nenhum problema aqui, vamos ver como usar isso no Arduino fazendo uso do comando *if*, que é traduzido simplesmente como "se". Basicamente, se uma afirmação for verdadeira, ele executa uma ação.

Vamos adicionar o código abaixo no nosso programa e fazer alguns testes. Lembre-se de inseri-lo logo acima do último "}" :

```
//código inserido no final da aula 06
if(3 > 2 && 5 > 1){
    Serial.println("Verdadeiro");
}
```

Veja que no monitor serial, a cada 12 segundos, será impressa a palavra "Verdadeiro.". Tente substituir o que está entre parêntesis por, incluindo os parêntesis, por:

```
(!(2 > 3))
(3 < 1 || 10 > 2)
(!(3 < 1 && 10 > 2))
(!(3 < 1 && 10 > 2) || (10 < 20))
```

Note que em todos os casos acima temos como resultado algo que é verdadeiro.

CONDICIONAIS

Em linguagem de programação, temos os condicionais *if* e *else* que são muito importantes em linguagem C++. A forma de trabalhar aqui é muito simples:

```
if(condição_a_ser_avalizada){
    faça_isso_se_verdadeiro
}
else{
    faça_isso_se_falso
}
```

Uma tradução direta seria:

```
se(condição_a_ser_avalizada){
    faça_isso_se_verdadeiro
}
se não{
    faça_isso_se_falso
}
```

Basicamente o programa executa uma ação se determinada condição for verdadeira e outra ação se tal condição for falsa. Veremos que esta condição pode ser o pressionar ou não um determinado botão.

Não se esqueça de entrar na página do seu professor para praticar um pouco o que você aprender e ter um pequeno spoiler da próxima aula.

Na próxima aula vamos utilizar um botão para entrar com dado digital e, usando os condicionais aprendidos aqui, acenderemos uma lâmpada.