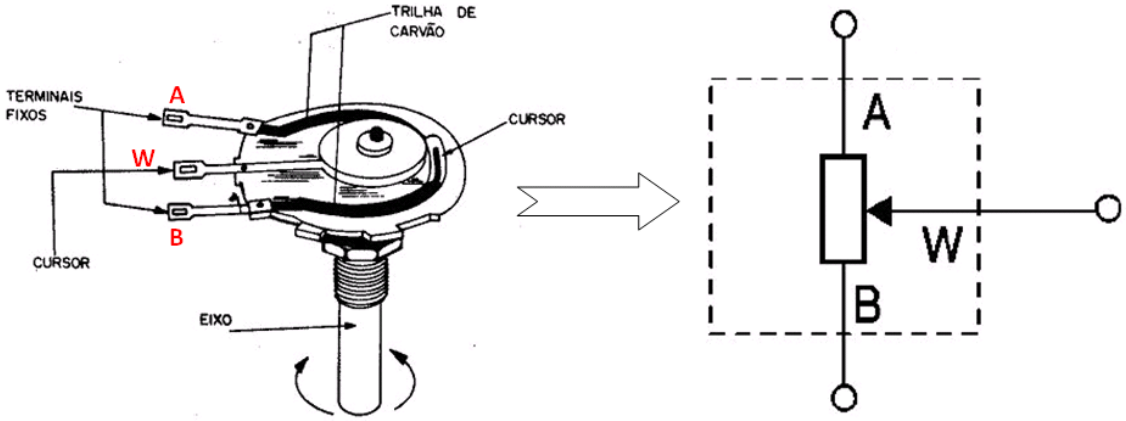
Atividade Prática 4

Introdução

Potenciômetro

Potenciômetros são resistores variáveis e ajustáveis, e por isso são usados para controle analógico de funcionalidades de alguns aparelhos eletrônicos, tal como o volume de um aparelho de som.
Eles costumam possuir três terminais. Dois são ligados às extremidades da resistência (A e B) e a terceira a um cursor que anda de ponta a ponta da resistência (W).



Leitura de Entrada Analógica

A leitura da entrada analógica é feita com a função analogRead, que recebe como parâmetro o pino analógico a ser lido e retorna o valor digital que representa a tensão no pino. Como o conversor analógico-digital do Arduino possui uma resolução de 10 bits, o intervalo de tensão de referência, que no nosso caso é 5 V, será dividido em 1024 pedaços (2^10) e o valor retornado pela função será o valor discreto mais próximo da tensão no pino.

Exemplo:

```
unsigned int valorLido = analogRead(A0);
```

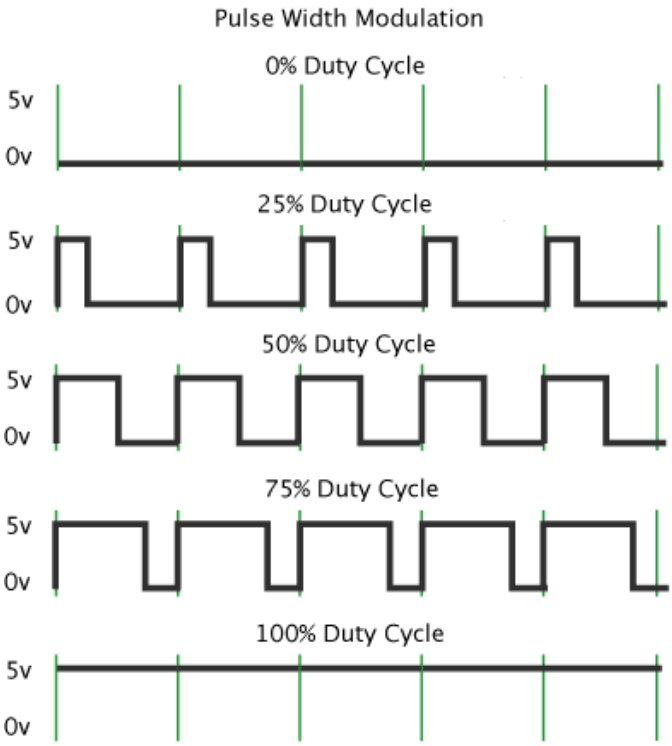
O código acima lê o valor analógico de tensão no pino A0 e guarda o valor digital na variável valorLido. Supondo que o pino está com uma tensão de 2V, o valor retornado pela conversão será:

2 x 1024 / 5 = 409,6

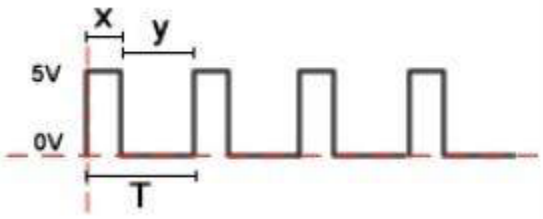
PWM

PWM (Pulse Width Modulation – Modulação por Largura de Pulso) é uma técnica para obter resultados analógicos por meios digitais. Essa técnica consiste na geração de uma onda quadrada em uma frequência muita alta em que pode ser controlada a porcentagem do tempo em que a onda permanece em nível lógico alto. Esse tempo é chamado de Duty Cycle e sua alteração

provoca mudança no valor médio da onda, indo desde 0V (0% de Duty Cycle) a 5V (100% de Duty Cycle) no caso do Arduino.



O duty cycle é a razão do tempo em que o sinal permanece na tensão máxima (5V no Arduino) sobre o tempo total de oscilação, como está ilustrado na figura abaixo:



O valor do Duty Cycle usado pelo Arduino é um inteiro armazenado em 8 bits, de forma que seu valor vai de 0 (0%) a 255 (100%).
Para escrever um sinal na saída PWM utiliza-se a função analogWrite, que recebe como parâmetros o pino PWM e o valor do duty cycle, respectivamente. Esse último parâmetro é guardado em 8 bits, de modo que esse valor deve estar entre 0 (0% de duty cycle) e 255 (100% de duty cycle).

```
analogWrite(9,127);//Escreve no pino 9 um sinal PWM com 50% de duty cycle //(50% de 255=127)
```

```
analogWrite(10,64);//Escreve no pino 10 um sinal PWM com 25% de duty cycle //(25% de 255=64)
```

Variando o duty cycle altera-se também o valor médio da onda, de modo que o efeito prático obtido com o PWM em algumas aplicações é um sinal com

amplitude constante e de valor igual ao valor médio da onda. Isso permite que se possa, por exemplo, controlar a intensidade do brilho de um LED, ou a velocidade de um motor de corrente contínua.

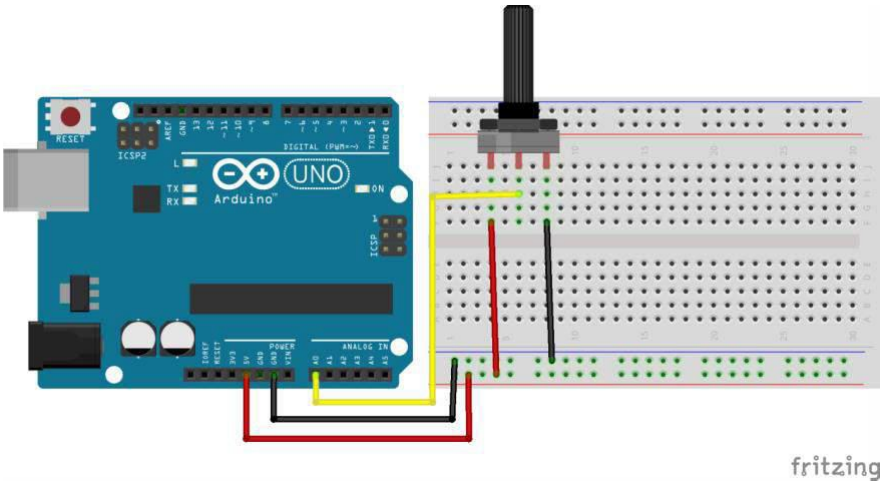
Para variarmos o PWM, usamos o valor analógico lido no potenciômetro e armazenamos na variável valorLido, que armazena valores entre 0 e 1023. Porém, para que seja possível usar essa variável para controlar o PWM, devemos mudar sua escala para 0 a 255. Para isso usaremos uma função. Tal função recebe uma variável e muda sua escala.

```
pwm = map(valorLido, 0, 1023, 0, 255); // Mudança de escala

analogWrite(led,pwm);//Escreve no led um sinal PWM
proporcional ao valorLido
```

Roteiro

1- Monte o circuito conforme a figura a seguir:



2- Crie um código para mostrar a cada 1 segundo (no monitor serial) o valor em bits da tensão de acordo com o ajuste do potenciômetro.
3- Altere o código anterior e inclua um LED no circuito para variar a intensidade do LED através do potenciômetro.