Ministério da Educação
Universidade Tecnológica Federal do Paraná
Coordenação de Engenharia de Computação
Materiais e Equipamentos Elétricos
Campus Apucarana

Laboratório de Materiais e Equipamentos Elétricos

Roteiro 12: Motor de Passo

1. Objetivos

- Controle de um motor de passo
- Utilização da biblioteca <Stepper.h>
- Ligação elétrica de um motor de passo em um driver

2. Material utilizado

- Arduino UNO;
- Matriz de contato (protoboard);
- Driver ULN2003
- Motor de Passo 28BYJ-48
- Potenciômetro de 10 kOhms

2.1. Motor de Passo

Os motores de passo usam uma roda dentada (com 32 dentes) e quatro eletroímãs que formam um anel ao redor da roda. Cada pulso alto enviado energiza a bobina, atraindo os dentes mais próximos da roda dentada e girando o motor em incrementos de ângulo precisos e fixos conhecidos como **passos.**

A imagem abaixo (Figura 1) mostra um exemplo ilustrativo de um processo de 4 bobinas sendo alimentadas de forma alternada para reproduzirem o giro

controlado do motor de passo[1].

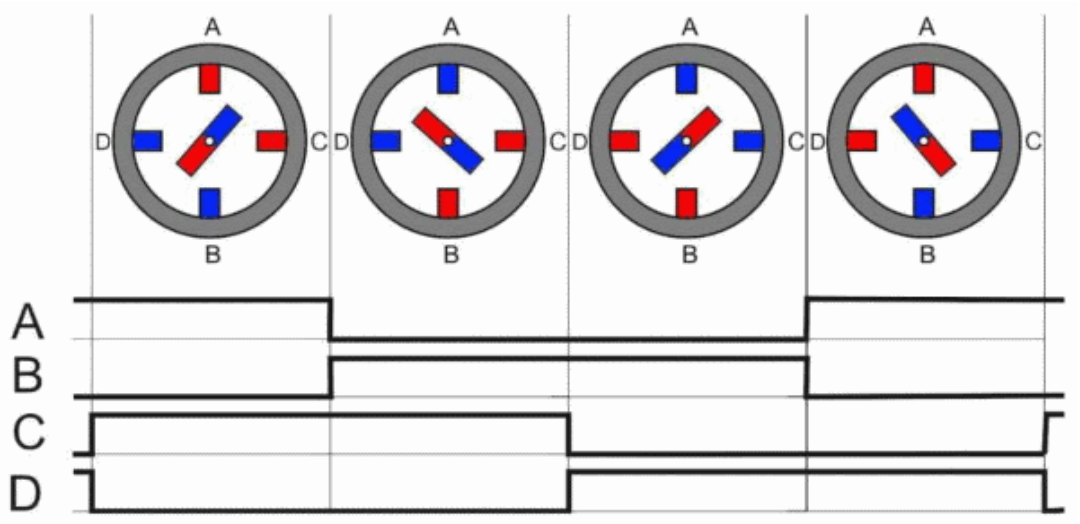

Figura 1: Bobinas de um motor de passo.

A maneira como você pulsa essas bobinas afeta muito o comportamento do motor.

- A sequência de pulsos determina a direção de rotação do motor.
- A frequência dos pulsos determina a velocidade do motor.
- O número de pulsos determina até que ponto o motor irá girar.

O 28BYJ-48 (Figura 2) é um motor de passo unipolar de 5 fios que funciona com 5 V. A melhor parte deste motor é que ele pode ser posicionado com precisão um 'passo' de cada vez. Ele se sai bem em projetos que exigem posicionamento preciso, como abrir e fechar uma válvula.

Considerando seu tamanho, o motor entrega um torque decente de 34,3 mN.m a uma velocidade de cerca de 15 RPM. Ele fornece um bom torque mesmo em estado parado, que é mantido enquanto a energia é fornecida ao motor. A única desvantagem é que é um pouco faminto de energia e consome energia mesmo quando não está em movimento.

[1] Sugestão de vídeo: How Stepper Motors Work - Electric motor

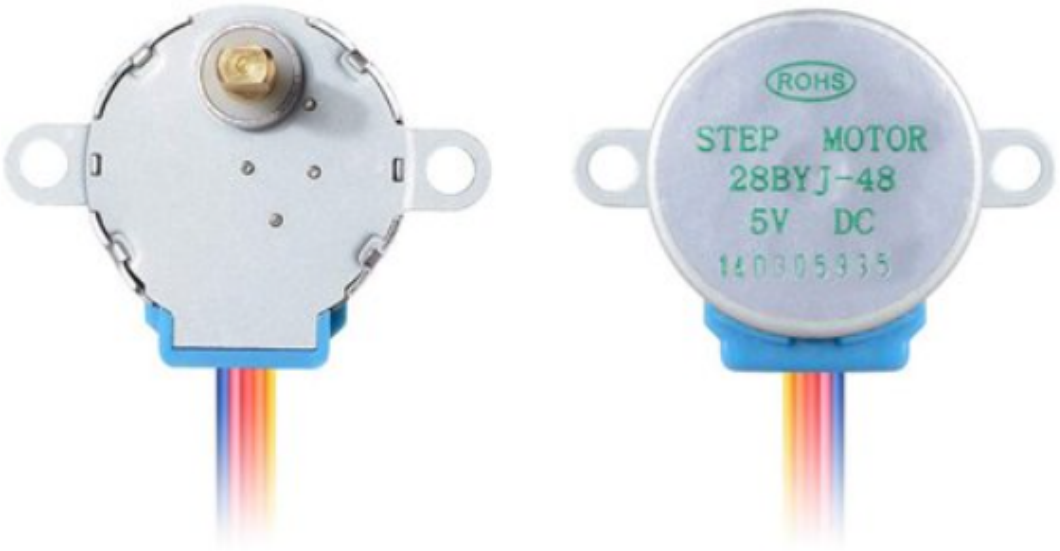

Figura 2: Motor de passo 28BYJ-48

A pinagem desse motor é mostrado abaixo (Figura 3). O 28BYJ-48 consiste em duas bobinas e cada bobina possui uma derivação (tap) central. As duas derivações centrais são conectadas internamente e fornecidas pelo 5º fio (fio vermelho). Uma extremidade da bobina e um tap central juntas formam uma fase. Portanto, 28BYJ-48 tem um total de 4 fases.

O fio vermelho é sempre alimentado com nível ALTO (5 V). Quando outro fio é alimentado com nível BAIXO (0 V), a fase é energizada. O motor de passo gira apenas quando as fases são energizadas em uma sequência lógica conhecida como sequência de passo.

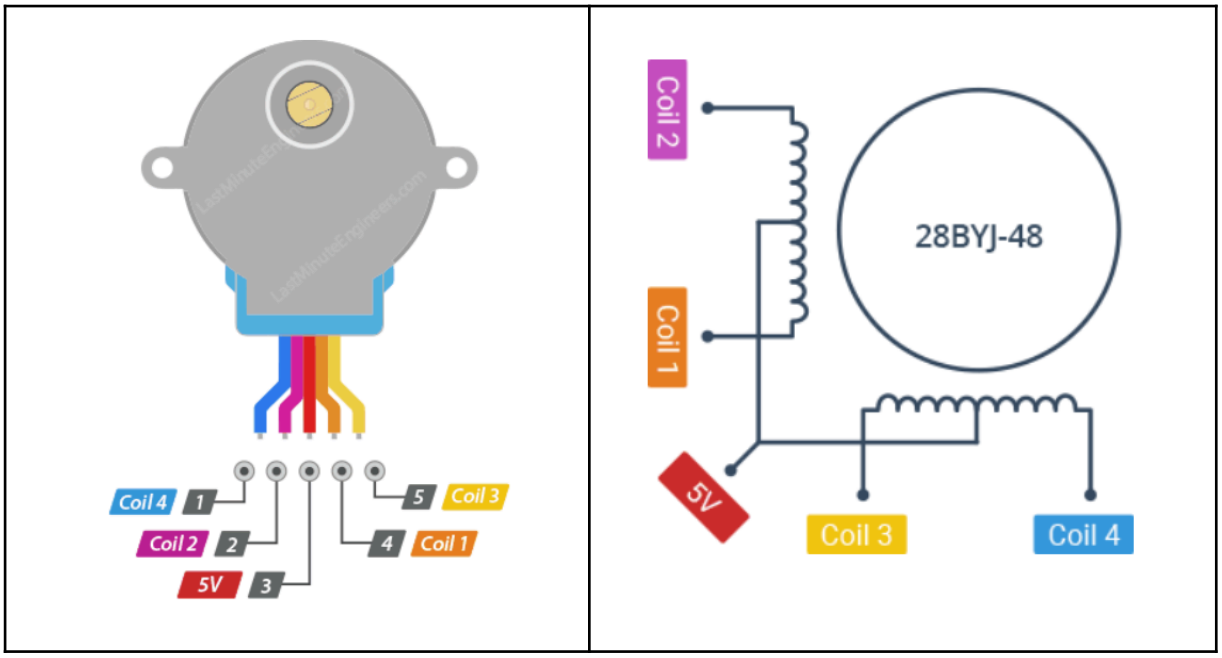

Figura 3: Pinagem do motor de passo 28BYJ-48

Quando o motor 28BYJ-48 é operado em modo *full step* cada passo corresponde a uma rotação de 11,25°. Isso significa que há 32 passos por revolução (360°/11,25° = 32).

Além disso, o motor possui conjunto de redução de 1/64 (na verdade, é 1/63.68395, mas para a maioria dos propósitos 1/64 é uma aproximação boa o suficiente). Isso significa que existem, na verdade, 2038 passos (32*63,68395 passos por revolução = 2037,8864 aproximadamente 2038 passos). O esquema de redução do motor é mostrado na Figura 4.

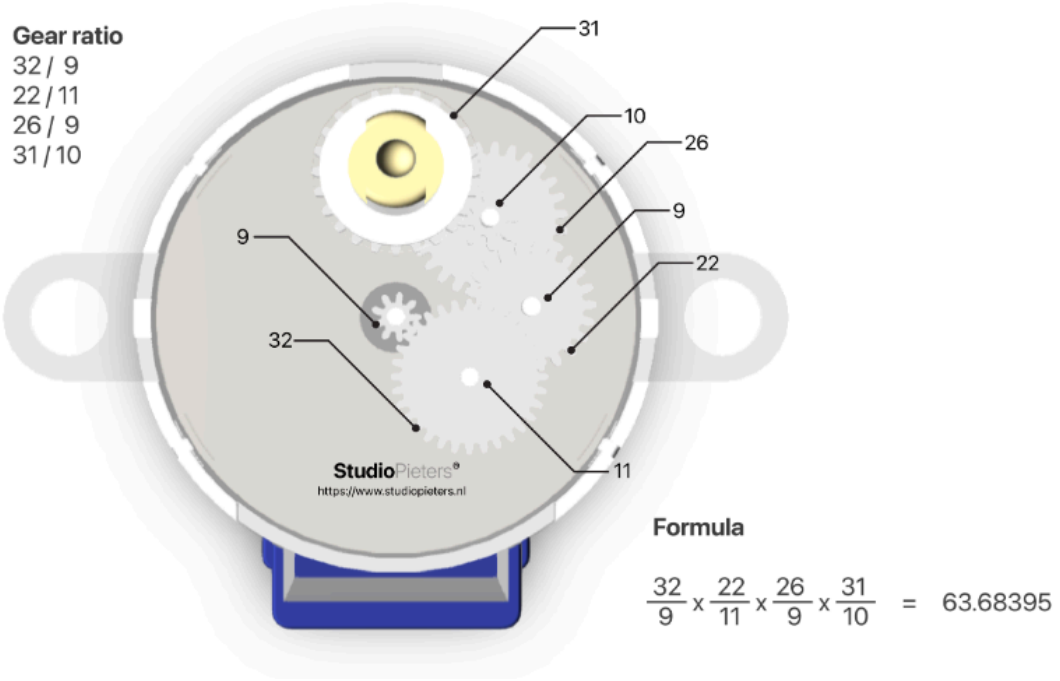

Figura 4: Redução mecânica do motor de passo 28BYJ-48

O 28BYJ-48 normalmente consome cerca de 240mA. Como o motor consome muita energia, é melhor alimentá-lo diretamente de uma fonte de alimentação externa de 5V, em vez de extrair essa energia do Arduino. Observe que o motor consome energia mesmo em estado parado para manter sua posição.

2.2. Driver

Como o motor de passo 28BYJ-48 consome muita corrente, um microcontrolador como o Arduino não pode controlar o motor diretamente. Ele requer um circuito integrado (CI) de *driver* como ULN2003 para controlar o motor (Figura 5). O motor de passo 28BYJ-48 é muitas vezes vendido junto com o driver de motor ULN2003.

Conhecido por sua capacidade de alta corrente e alta tensão, o ULN2003 oferece um ganho de corrente mais alto do que um único transistor e permite que a saída de baixa corrente de baixa tensão de um microcontrolador acione um motor de passo de alta corrente.

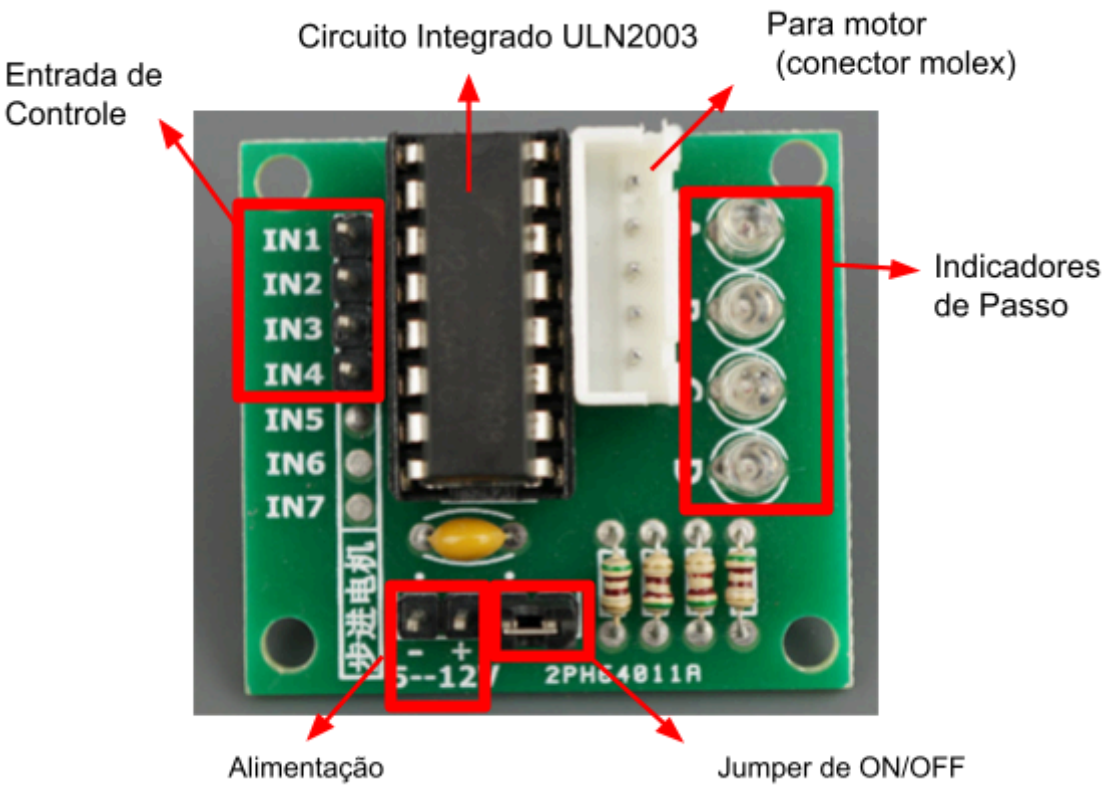

Figura 5: *Driver* ULN2003 para controle do motor de passo.

A placa possui um conector *MOLEX* que conecta perfeitamente os fios do motor tornando muito fácil conectar o motor à placa. Existem quatro entradas de controle, bem como uma conexão de alimentação.

A placa possui quatro LEDs que mostram atividade nas quatro linhas de entrada de controle (para indicar o estado de passo). Eles fornecem uma boa noção visual do passo do motor. A placa também vem com um jumper ON/OFF para isolar a alimentação do motor de passo.

A pinagem do driver é mostrada abaixo:

- IN1 – IN4 são usados para acionar o motor. Conecte-os na saída digital no Arduino.
- - é o pino e referência (GND). A referência é conectada tanto no Arduino quanto na fonte de alimentação para garantir que todo o circuito esteja na mesma referência.

- + é a alimentação do motor. Conecte em uma fonte externa de 5 V. Pelo fato do moto consumir muita potência, você NUNCA deve usar os 5 V da placa do Arduino.
- Conector do motor é ligado no motor por um conector molex, que possui uma saliência para evitar a conexão incorreta.

2.3. Configuração

Considere o exemplo abaixo ilustrado na Figura 6 mostrando o esquema de ligação do motor de passo e driver no Arduino. Desejamos fazer um motor de passo dar um giro completo em um sentido, esperar 1 segundo, dar um giro no sentido anti-horário, e esperar 1 segundo, em loop.

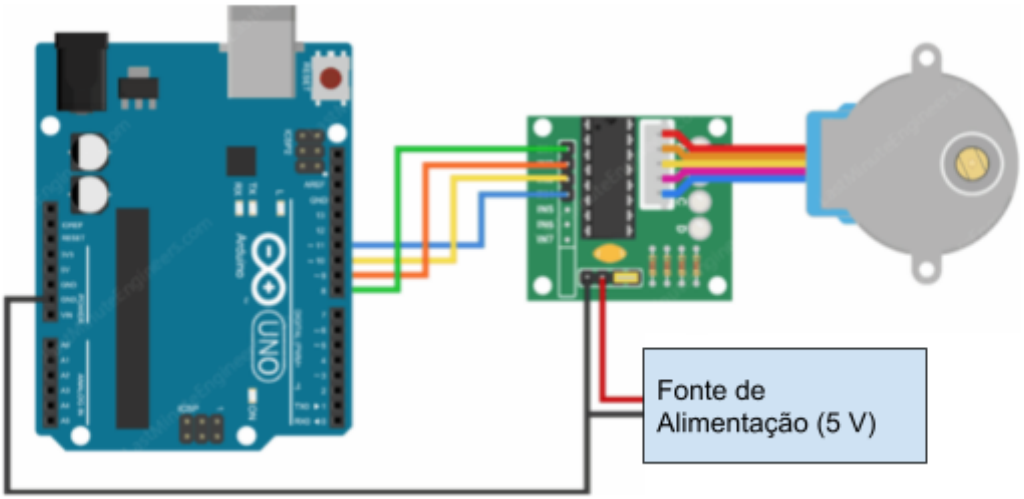

Figura 6: Esquema de ligação do Driver e motor de passo no Arduino.

Iniciamos a configuração incluindo a biblioteca **<Stepper.h>** que irá possibilitar a criação de objetivos do tipo Stepper. Para criar uma instância do objeto Stepper, precisamos primeiro definir, a partir dos dados do fabricante do motor, quantos passos o motor possui por revolução (por giro). Neste caso específico vimos que são 2038 passos por revolução. A partir dessa informação podemos instanciar um objeto do tipo “servo”, definindo os pinos em que o servo irá estar conectado

(seguindo a ordem IN1-IN3-IN2-IN4).

```
//Incluem a biblioteca do motor de passo
#include <Stepper.h>

// Define o número de passos por revolução (steps per
revolution)
const int stepsPerRevolution = 2038;

// Cria uma instância da classe "Stepper"
// *Motor de passo em inglês = step motor
// Pinos são declarados na sequência IN1-IN3-IN2-IN4
Stepper myStepper = Stepper(stepsPerRevolution, 8, 10, 9, 11);
```

Após instanciar o servo, definimos a velocidade do servo (em RPM) utilizando o método **.setSpeed().** Veja que esse método não faz o motor se movimentar, apenas define a velocidade com que ele vai rodar. É possível mudar a velocidade no servo no loop também.

```
void setup() {
      // A biblioteca já ajusta os pinos como saída

      // 1 RPM = 1 revolução por minuto
      myStepper.setSpeed(5);
}
```

No loop, utilizando o método **.step()** para fazer o motor se movimentar. A função recebe quantos passos o motor irá executar. Lembrando que o uma volta completa possui 2038 (que está gravado na variável stepPerRevolution). É possível atribuir um passo negativo para que o motor gire em sentido contrário. Por exemplo, se o comando fosse **myStepper.step**(1019), o motor irá girar meia volta apenas.

```
void loop() {
      // Roda uma volta em sentido horário em 5 RPM
      myStepper.step(stepsPerRevolution);
      delay(1000);

      // Roda uma volta em sentido anti-horário em 5 RPM
      myStepper.step(-stepsPerRevolution);
      delay(1000);
}
```

3. Parte prática

A. Faça com que o motor dê uma volta completa com uma velocidade de 3 RPM em um sentido, aguarde 1 segundo e depois gire uma volta completa com uma velocidade de 10 RPM no sentido contrário e depois aguarde 1 segundo.

B. Faça com que motor gire 180 graus no sentido horário e depois 90 graus no sentido anti-horário a uma velocide de 5 RPM.

C. Elabore um código que irá fazer a leitura de um potenciômetro em uma entrada analógica e irá controlar a velocidade do motor de 0 - 10 RPM que deverá girar continuamente. Dica: o código irá travar caso seja colocado velocidade = 0 como parâmetro, então seu código deverá identificar valores pequenos de velocidade para não utilizar a conversão direta do A/D como entrada da velocidade.

D. **Desafio:** Use dois botões de forma que toda vez que o botão for pressionado, o motor gire 45 graus (por exemplo, para dar uma volta completa, será preciso pressionar o botão 8 vezes). Um botão irá fazer o motor girar no sentido horário e o outro botão no sentido anti-horário. Caso algum botão seja pressionado por mais do que 2 segundos, o motor irá girar continuamente.