MVO-20 - Fundamentos da teoria de controle

Laboratórios de Controle

Aeropêndulo

Professores:
Guilherme Soares (soaresgss@ita.br)
Flávio Ribeiro (flaviocr@ita.br)

Apoio:

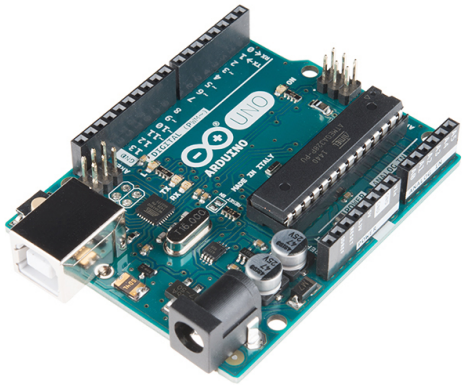
Cronograma do 1o bimestre

- Lab 1: Familialização com o kit de eletrônica: leitura do potenciômetro;
- Lab 2 (04/setembro): Projeto do pêndulo;
- **Lab 3 (11/setembro): Familiarização com o kit de eletrônica: enviar sinal para motor DC;**
- Lab 4 (18/setembro): Montagem do pêndulo e identificação estática;

Apresentação do pêndulo: dia 11 de outubro (quarta-feira da 2a semana/2o bimestre)

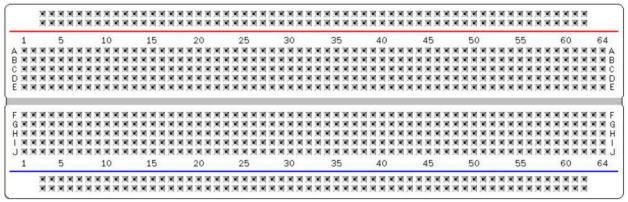
Hoje: comandar motor DC

- Arduino UNO;



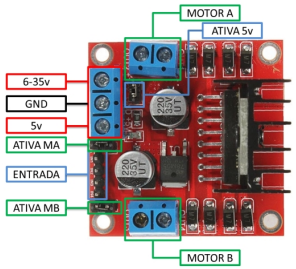
- Motor DC;

- Protoboard;



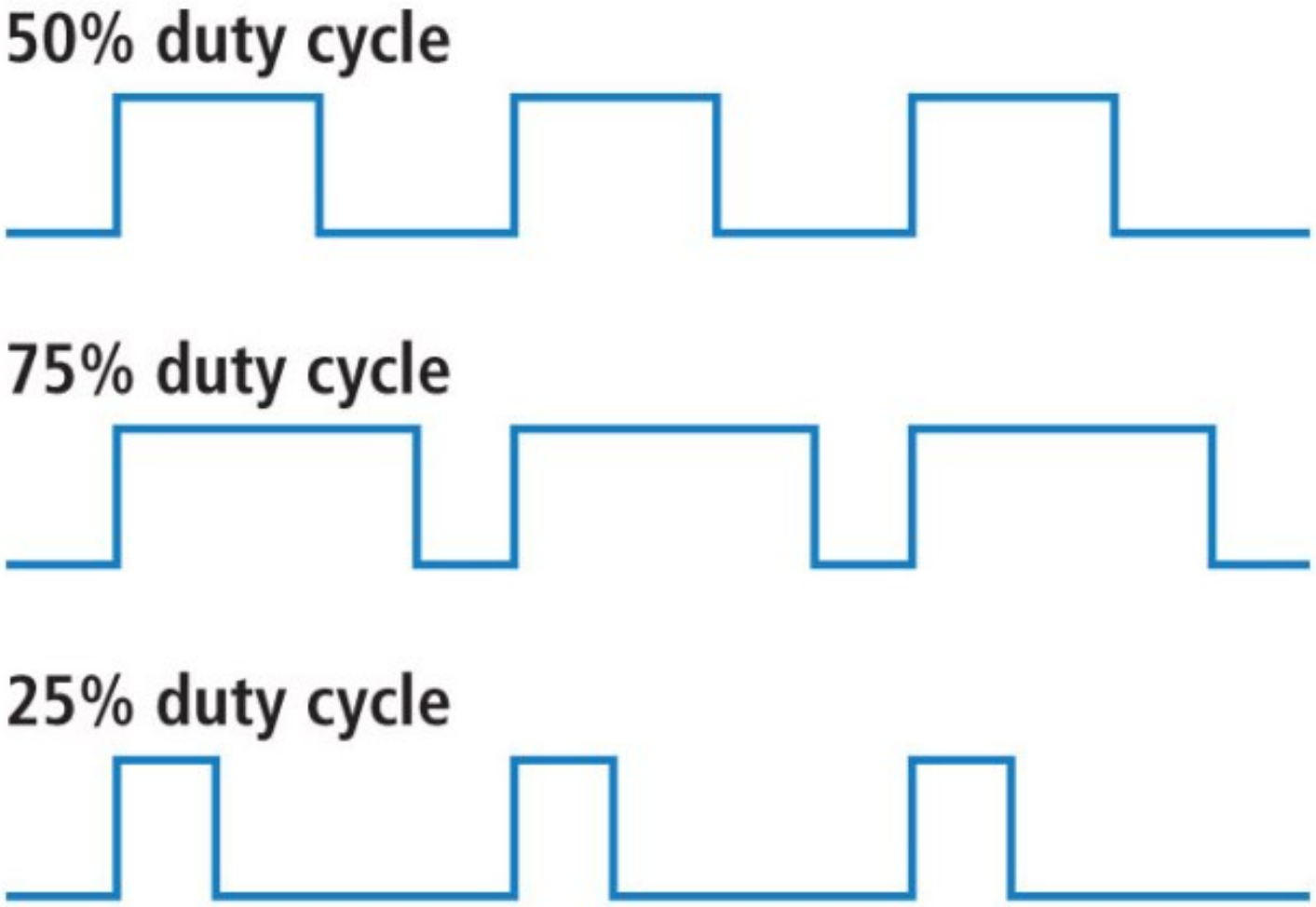
- Fontes de 12 V e conectores DC:

- Driver do motor;



- Botão, fios, etc.

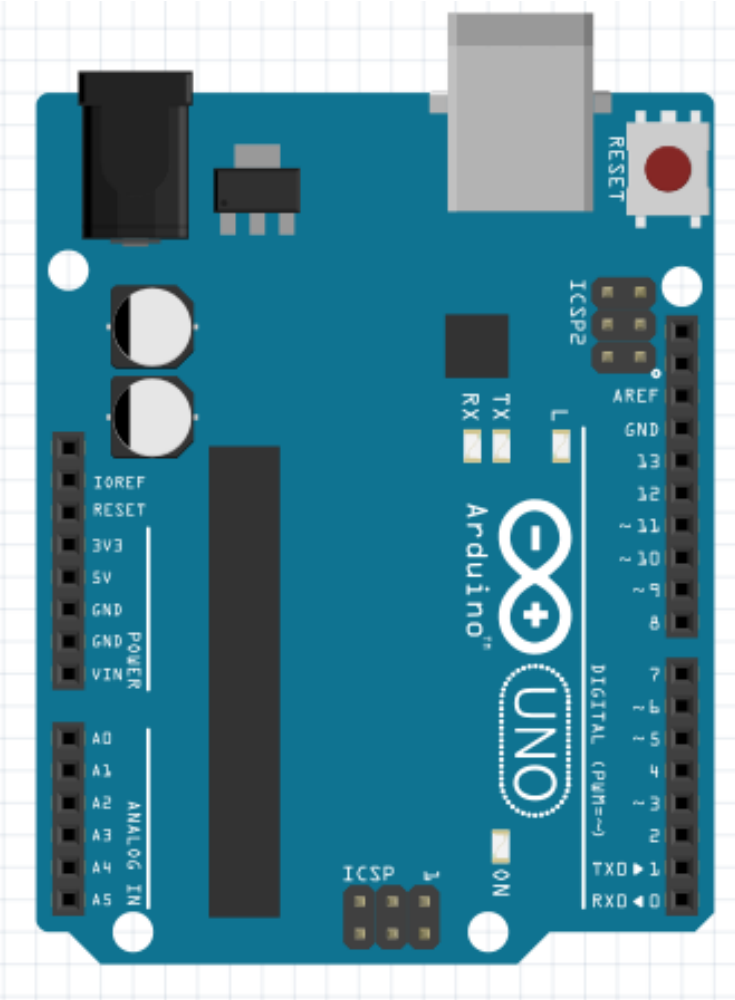
Sinal PWM



https://learn.sparkfun.com/tutorials/pulse-width-modulation

Como gerar um sinal PWM no Arduino

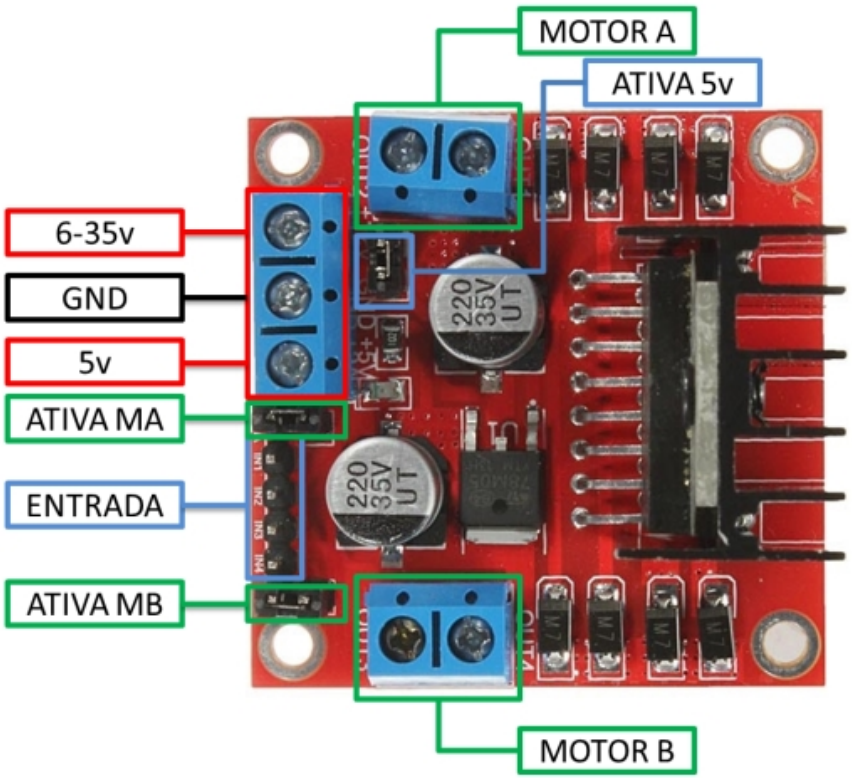
- Usar uma das portas digitais do Arduino precedidas por um ~ (~ 3, ~ 5, ~ 9, ~ 10, ~ 11) como saída;
- Inicializar a porta no “setup”: *pinMode(numeroPIN, OUTPUT)*;
- No “loop”, usar *analogwrite* para especificar o valor do PWM (valor entre 0 e 255): *analogWrite(numeroPIN, valorPWM)*;



Exemplo código de sinal PWM: controle do brilho de um LED

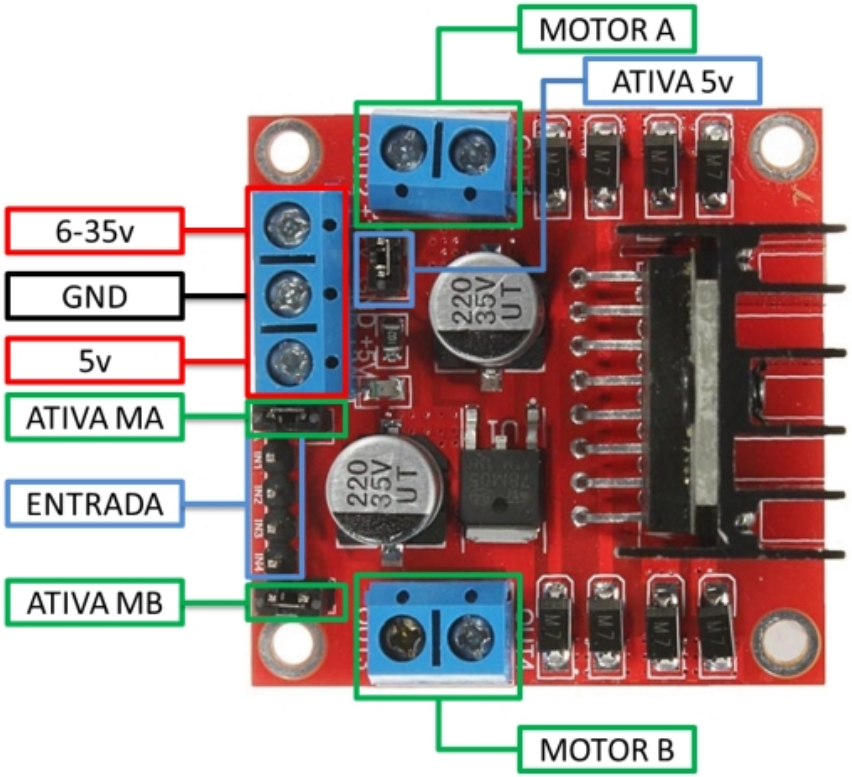
```
int led = 9;           // the pin that the LED is attached to
int brightness = 0;    // how bright the LED is
int fadeAmount = 5;    // how many points to fade the LED by

// the setup routine runs once when you press reset:
void setup()  {
  // declare pin 9 to be an output:
  pinMode(led, OUTPUT);
}

// the loop routine runs over and over again forever:
void loop()  {
  // set the brightness of pin 9:
  analogWrite(led, brightness);

  // change the brightness for next time through the loop:
  brightness = brightness + fadeAmount;

  // reverse the direction of the fading at the ends of the fade:
  if (brightness == 0 || brightness == 255) {
    fadeAmount = -fadeAmount ;
  }
  // wait for 30 milliseconds to see the dimming effect
  delay(30);
}
```

Motor driver / ponte L298N



- 6-35 V: alimentação da placa;
- Ativa MA: quando jumper está ativo, aciona motor A com velocidade máxima;
- Entrada, IN1 e IN2: utilizados para controlar o sentido do motor A;
- Ativa 5v e 5v: quando ativo, a placa utiliza o regulador de tensão para fornecer 5V (na porta 5V);

Motor driver / ponte H L298N



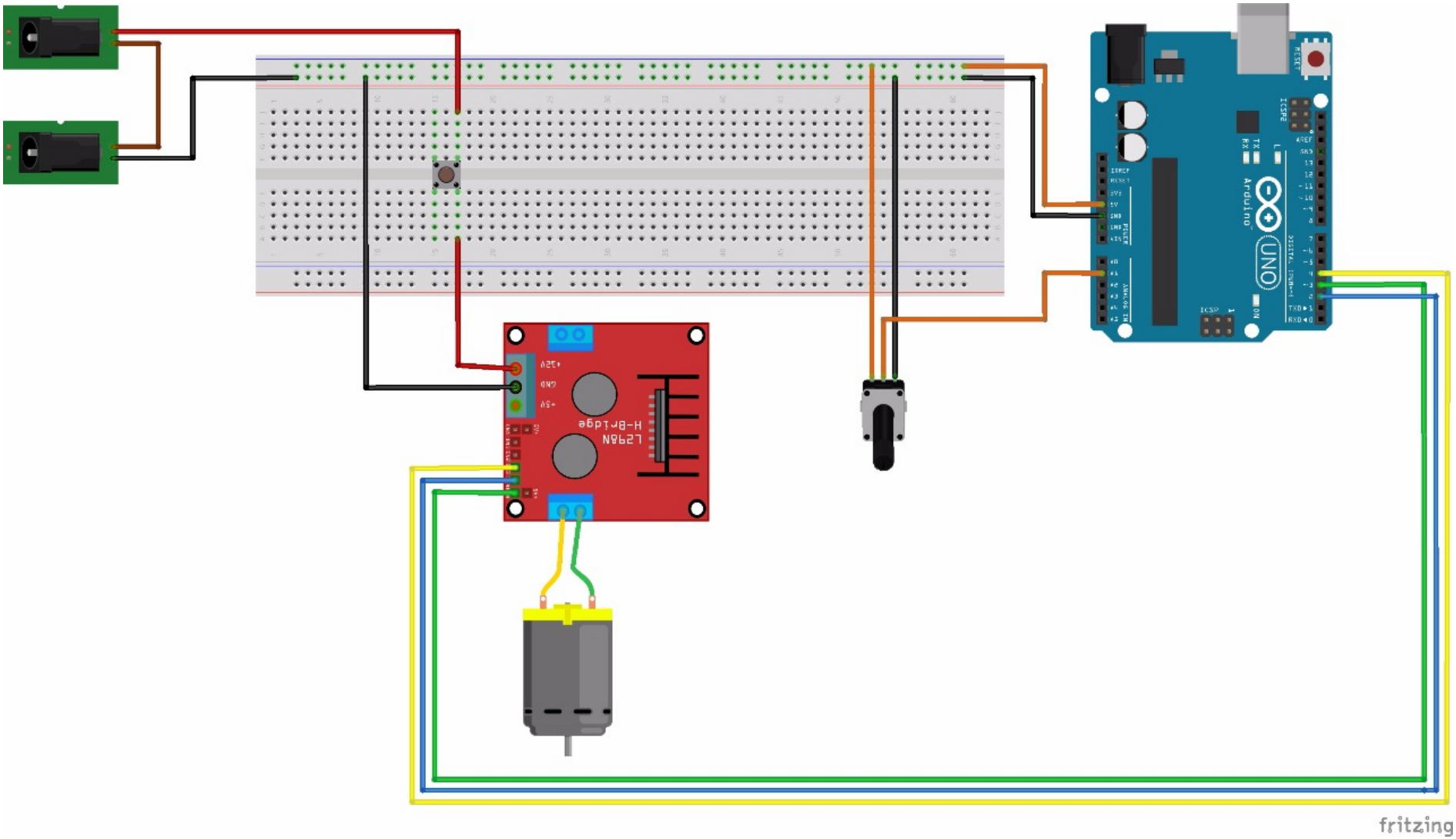
Entradas possíveis para o controle do motor A:

IN1	IN2	Estado
0V	0V	desligado
0V	5V	sentido 1
5V	0V	sentido 2
5V	5V	freio

para o motor B, mesmo procedimento, utilizando portas IN3 e IN4.

Sugestão de tutorial: `www.filipeflop.com/blog/motor-dc-arduino-ponte-h-l298n/`

Exemplo de conexão dos componentes



Exemplo de código que controla o motor

```
int IN1 = 4 ;
int IN2 = 2 ;
int motorpin = 3;              // PWM pin do motor
int throttle = 100;

void setup() {
  pinMode(IN1, OUTPUT);
  pinMode(IN2, OUTPUT);
  pinMode(motorpin, OUTPUT);
  digitalWrite(IN2,LOW);
  digitalWrite(IN1,HIGH);
}

void loop() {
  analogWrite(motorpin, throttle);

  // espera 15 ms antes de enviar outro comando
  delay(15);
}
```

Atividades do dia

- Implementar o código do slide anterior e fazer alguns testes (modificar sentido e velocidade do motor);
- Modificar o código, utilizando o sinal do potenciômetro como entrada para controlar a velocidade do motor.

Algumas idéias adicionais:

- Modificar o sentido de rotação do motor, dependendo da posição do potenciômetro;
- Fazer um código que gire o motor com velocidade proporcional a velocidade de rotação do potenciômetro.