Contrôle de moteur à transistor

Apprenez à contrôler un moteur à courant continu avec un transistor, en utilisant PWM.

Dernière révision · 07/02/2023

> *Cet article a été révisé le 18/01/2022 par Karl Söderby.*

Les moteurs et les transistors sont des composants électroniques très courants. Cet article vise à fournir quelques notions de base, ainsi qu'un exemple de code fonctionnel, des circuits et des schémas.

Lorsqu'un bouton-poussoir connecté à la broche numérique 2 est enfoncé, l'Arduino contrôlera un transistor via **une modulation de largeur d'impulsion (PWM)** , ce qui augmentera la vitesse du moteur, puis la ralentira.

À propos des transistors

L'Arduino ne peut fournir que 40 mA à 5 V sur ses broches numériques. La plupart des moteurs nécessitent davantage de courant et/ou de tension pour fonctionner. Un transistor peut servir d'interrupteur numérique, permettant à l'Arduino de contrôler des charges nécessitant une puissance plus élevée. Dans cet exemple, le transistor complète le circuit du moteur à la masse. Cet exemple utilise un TIP120, capable de commuter jusqu'à 60 V à 5 A.

La modulation de largeur d'impulsion (PWM) d'un transistor est similaire à l'impulsion d'une LED. Plus la valeur PWM est élevée, plus le moteur tourne vite. Plus la valeur est basse, plus la rotation est lente.

Les transistors possèdent trois broches. Pour les transistors bipolaires à jonction (BJT), comme celui utilisé dans cet exemple, ces broches sont appelées base, collecteur et émetteur. Un faible courant sur la broche de base ferme un circuit entre les broches du collecteur et de l'émetteur. Il existe deux types de BJT : NPN et PNP. Le TIP120 est un transistor de type NPN, ce qui signifie que le collecteur est connecté au moteur et l'émetteur à la masse.

À propos des moteurs

Les moteurs fonctionnent grâce à un processus appelé induction. Lorsqu'une charge électrique traverse un fil, un champ magnétique est créé. Un fil enroulé crée un champ plus fort, tout comme un courant plus fort. Dans un moteur à courant continu, un fil enroulé entoure l'arbre du moteur. Le champ magnétique généré est attiré et repoussé par des aimants situés à l'intérieur du moteur.

Lorsqu'un moteur s'arrête, un léger courant peut être généré tandis que l'arbre continue de tourner. Une diode placée en parallèle avec les fils du moteur empêchera l'électricité générée d'endommager le circuit.

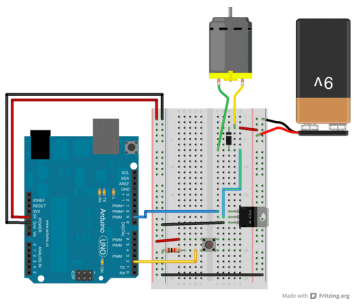
Les moteurs consomment le plus de courant au démarrage ou sous charge. Le courant de calage correspond à la quantité de courant absorbée par un moteur lorsqu'il est arrêté par une force. Lorsqu'un moteur est en marche, il consomme beaucoup moins de courant.

La tension nominale décrit la tension de crête de fonctionnement d'un moteur, lorsqu'il fonctionne à son rendement optimal. Dépasser ou sous-dépasser la tension nominale du moteur réduit sa durée de vie à long terme. Si la tension est inférieure à la tension nominale, le moteur tournera plus lentement. En général, un moteur a besoin d'environ la moitié de sa tension nominale pour fonctionner. Si la tension est inférieure au démarrage, il est probable qu'il ne démarre pas.

Matériel requis

- **Carte Arduino**
- Un interrupteur ou un bouton momentané
- résistance de 10 kohms
- Planche à pain
- Fil de raccordement
- moteur 9 V CC
- Transistor TIP120
- Diode 1N4001
- pile 9V

Circuit



Circuit de commande de moteur à transistor.

Commencez par connecter les fils d'alimentation et de terre. Sur l'illustration, le fil rouge (alimentation) et le fil noir (terre) sont connectés aux deux longues rangées verticales sur le côté de la plaque d'essai. Cela permet d'accéder à l'alimentation 5 V et à la terre.

Placez le bouton-poussoir sur la plaque d'essai, à cheval sur le centre. Un fil relie la broche numérique 2 à une borne du bouton-poussoir. Cette même borne est reliée à la masse via une résistance de rappel de 10 kOhms. La borne du bouton non connectée à l'Arduino doit être câblée à l'alimentation 5 V.

Connectez la broche 9 de l'Arduino à la broche de base du TIP120. Si vous regardez le transistor avec la languette métallique tournée vers l'extérieur, la broche de base se trouve à gauche du transistor. C'est cette broche qui contrôle l'ouverture et la fermeture. Le collecteur du transistor est connecté à une borne du moteur, et l'émetteur à la masse.

L'autre extrémité du moteur se connecte au pôle positif de la pile 9 V. Reliez la masse de la pile à celle de l'Arduino.

Schématique

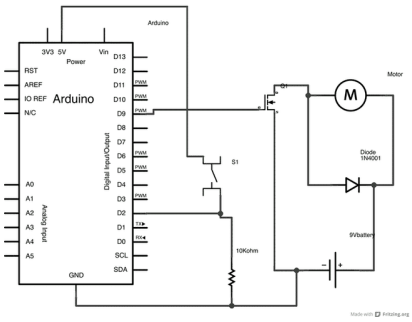


Schéma de contrôle du moteur à transistor.

Exemple de code

Vous trouverez ci-dessous le programme complet permettant de contrôler un moteur à courant continu avec un transistor. Vous trouverez une explication plus détaillée plus loin dans cet article.

```
/*
 Motor Control with a Transistor

 This example shows you how to control a motor's using a transistor.
 When a pushbutton on pin 2 is pressed, the Arduino will control a transistor
 via PWM, which will slowly ramp up the motor's speed, then slow it down.

 The circuit :
 * momentary switch with one end connected to 5V, the other end connected
   to GND through a 10-kilohm resistor, and digital pin 2.
 * TIP120 tranistor, with the Base connected to digital pin 9, the Emitter to ground,
   and the Collector to one lead from a 9V DC motor
 * a 9V battery, with the ground connected to the Arduino's ground, and the power
   connected to the motor
 * 1N4001 diode across the motor's leads, with the striped side connected to the 9V

 The Arduino can only provide 40mA at 5V on its pins. Most motors require
 more current and/or voltage to overcome inertia and run. A transistor
 can act as a digital switch, enabling the Arduino to control loads with
 higher electrical requirements.

 Created on 03 January 2013
 by Scott Fitzgerald

 https://docs.arduino.cc/learn/electronics/transistor-motor-control

 This example code is in the public domain.
 */

// give a name to digital pin 2, which has a pushbutton attached
int pushButton = 2;

// the transistor which controls the motor will be attached to digital pin 9
int motorControl = 9;

// the setup routine runs once when you press reset:
void setup() {
  // make the pushbutton's pin an input:
  pinMode(pushButton, INPUT);

  // make the transistor's pin an output:
  pinMode(motorControl, OUTPUT);
}

// the loop routine runs over and over again forever:
void loop() {

  // read the state of the button and check if it is pressed
  if(digitalRead(pushButton) == HIGH){
    // ramp up the motor speed
    for(int x = 0; x <= 255; x++){
      analogWrite(motorControl, x);
      delay(50);
    }

    // ramp down the motor speed
    for(int x = 255; x >= 0; x--){
      analogWrite(motorControl, x);
      delay(50);
    }
  }

  delay(1);        // delay in between reads for stability
}
```

Procédure pas à pas du code

Dans cette section, vous trouverez une présentation détaillée du programme.

Tout d'abord, créez une paire de variables pour l'état du bouton-poussoir et la broche de contrôle du moteur :

```
int pushButton = 2;
int motorControl = 9;
```

Dans le setup(), déclarez ces broches comme entrée et sortie, respectivement :

```
void setup() {
  pinMode(pushButton, INPUT);
  pinMode(motorControl, OUTPUT);
}
```

Maintenant que votre configuration est terminée, passez à la `loop()`.

```
void loop() {
```

Lisez l'état du bouton-poussoir et vérifiez s'il est à l'état HAUT. Il est possible d'effectuer l'évaluation directement dans votre instruction if(), comme ceci :

```
if(digitalRead(pushButton) == HIGH){
```

Si le bouton est enfoncé, augmentez la vitesse du moteur en augmentant la valeur PWM de la broche MotorControl. Une fois la vitesse maximale atteinte, ralentissez.

```
for(int x = 0; x <= 255; x+=5){
      analogWrite(motorControl, x);
      delay(50);
    }

    for(int x = 255; x >= 0; x-=5){
      analogWrite(motorControl, x);
      delay(50);
    }
```

Fermez l'instruction if() et ajoutez un petit delay() avant de fermer la boucle().

```
}
  delay(1);
}
```

Cet article vous a-t-il été utile ?