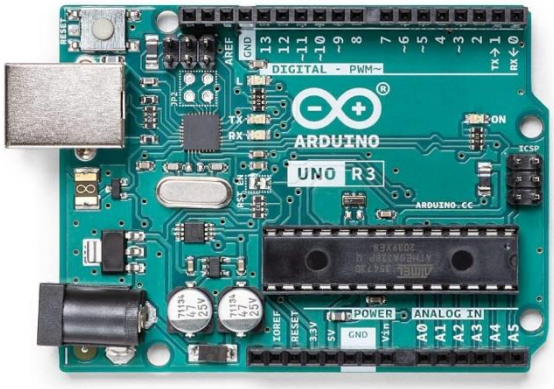
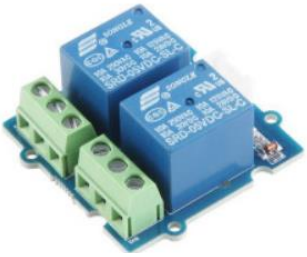
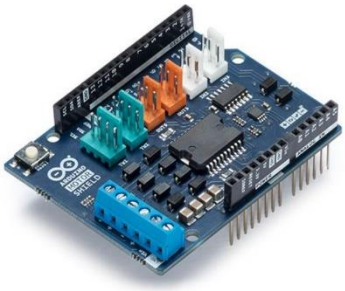
MOTEUR A COURANT CONTINU PILOTE PAR ARDUINO

Introduction

Une machine à courant continu est une machine électrique. Il s'agit d'un convertisseur électromécanique permettant la conversion bidirectionnelle d'énergie entre une installation électrique parcourue par un courant continu et un dispositif mécanique ; selon la source d'énergie.

1. Analyse des composants

Q1 . Quelle est la tension de sortie de la carte arduino UNO et son courant max ?

Q2. Quelle est la tension du moteur (cf. annexe) ?

Q3. Proposer un schéma de câblage pour trouver le courant nécessaire au moteur.

Hors tension , relier le moteur à l'alimentation continu.

STOP	**FAIRE VÉRIFIER PAR LE PROFESSEUR**

Q4. Quelle est le courant nécessaire au moteur ?

Q5. Dans quel sens tourne le moteur ?

Q6. Comment inverser le sens de rotation du moteur ? Proposer un schéma de câblage.

Hors tension , relier le moteur à l'alimentation continu.

STOP	**FAIRE VÉRIFIER PAR LE PROFESSEUR**

Q7. Quelle est le courant nécessaire au moteur ?

Q8. Dans quel sens tourne le moteur ?

Q9. Est-ce que l'arduino peut faire tourner le moteur ?

Q10. A quoi sert un relais ?

2. MCC 1 sens de marche Relais et Arduino

On désire à présent commander le moteur depuis une carte Arduino et un bouton poussoir

Hardware Overview

Pin Map

- 4 GND: connect this module to the system GND
- 3 VCC: you can use 5V for this module
- 2 SIG2: contral signal of swith2, high-NO2/low-NC2
- 1 SIG1: contral signal of swith1, high-NO1/low-NC1
- 5 NC1: one throw, connected to COM1 by default
- 6 COM1: controlled by SIG1, connected to NC1 or NO1
- 7 NO1: the other throw of swicth1
- 8 NC2: one throw, connected to COM2 by default
- 9 COM2: controlled by SIG2, connected to NC2 or NO2
- 0 NO2: the other throw of swicth2

Q11. Proposer un schéma de câblage pour piloter le moteur (1 sens de marche) grâce à une carte Arduino et un bouton poussoir.

Q12. Faire la table d'adressage des entrées sorties

Q13 Ecrire l'algorigramme permettant la commande du moteur lors d'un appuie sur le bouton poussoir.

Q14 Faire le programme MathLab

Hors tension , faire le montage

STOP	**FAIRE VÉRIFIER PAR LE PROFESSEUR**

3. MCC 2 sens de marche shield moteur

Pour simplifier l'utilisation du moteur on peut utiliser un shield arduino spécialement conçu pour cela.

Consulter l'annexe du shield moteur pour comprendre son fonctionnement

Q15. Proposer un schéma de câblage pour piloter le moteur (2 sens de marche) grâce à une carte Arduino, un shield moteur, un bouton poussoir et interrupteur.

Le bouton sert à faire tourner le moteur, l'interrupteur permet de choisir le sens.

Q16. Faire la table d'adressage des entrées sorties

Q17 Faire le programme MathLab qui réponds à la demande énoncée plus haut.

Hors tension , faire le montage

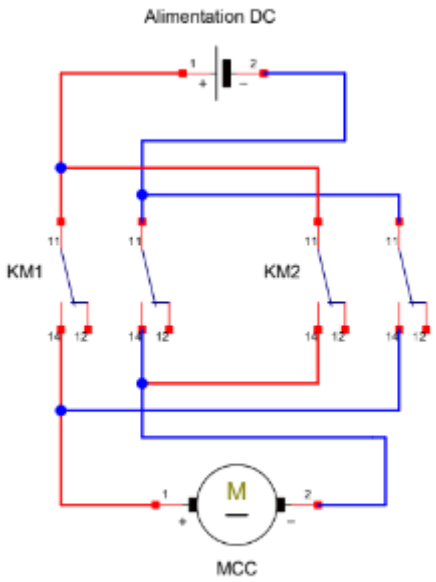
STOP	**FAIRE VÉRIFIER PAR LE PROFESSEUR**

4. MCC 2 sens de marche Relais et Arduino

Q18. A partir du schéma ci-contre, proposer un schéma de câblage pour piloter le moteur (2 sens de marche) grâce à une carte Arduino, 4 relais et un bouton poussoir et interrupteur.

Q19. Faire la table d'adressage des entrées sorties

Q20. Modifier si besoin votre programme MathLab



Hors tension , faire le montage

STOP	**FAIRE VÉRIFIER PAR LE PROFESSEUR**

Gear Motor

Technical Specifications

Model	Gear Motor
Article No.	Com-Motor04
Voltage Supply	3-9V DC
Rounds per Minute	Approx. 190 RPM at 6V
Motor Size	Approx. 12 x 24mm
Wheel Diameter	Approx. 67mm
Dimensions (Motor)	45 x 42mm
Scope of delivery	Motor, Wheel
EAN	4250236817248

Controlling a DC Motor with Motor Shield Rev3

Learn how to connect a DC motor to the shield, and how to control the speed and direction of the motor.

AUTHOR: **Karl Söderby**

Introduction

In this tutorial, we will learn how to control a DC motor, using the Motor Shield Rev3, a shield compatible with the Arduino UNO. We will take a look at three different pins: **brake, pwm & direction**, where we will create a simple sketch that uses all three of them. In addition we will also take a look at how we can power our project, using an external power source.

Goals

The goals of this project are:

- Set up your Motor Shield Rev3 to control a DC motor.
- Control the brake, pwm and direction of the motor.
- How to connect an external power source to the shield.

Hardware & Software Needed

- Arduino IDE (online or offline).
- Arduino Motor Shield Rev3(link to store)
- Arduino UNO (link to store)
- DC motor (6-12V)
- Power source (this tutorial uses 2x 3.7V Li-Ion 18650 batteries).

Controlling a DC Motor

There are several ways we can control a DC motor, perhaps the easiest one is just by applying power to it. Very early inventions using the DC motor simply worked like that: add a power source and the motor will start rotating, switch the polarity and you switch the direction.

But if we want to do a bit more than just making a motor spin full speed in two directions, we need a **motor control circuit.** More specifically, the dual full-bridge driver L298P, which we can find on the Motor Shield Rev3.

With this IC, we can set the work duty (0-100), enable brakes (HIGH, or LOW), and set the direction (HIGH or LOW). Each of these features can be controlled using a different set of pins. As we are going to control a DC motor in this tutorial, let's take a look at the pins that are used:

Channel A:

- **D12** - Direction
- **D3** - PWM (work duty)
- **D9** - Brake
- **A0** - current sensing.

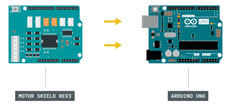
Channel B:

- **D13** - Direction
- **D11** - PWM (work duty)
- **D8** - Brake
- **A1** - current sensing.

Now if we were to use just the bare chip, it would involve a bit more complex circuitry. Fortunately, all we have to worry about is connecting the DC motor to one of the channels, connect an external power source, and we are good to go!

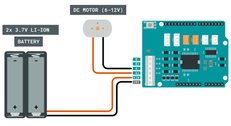
Circuit

Let's begin by mounting our Arduino Motor Shield Rev3 on top of an Arduino UNO.



Mounting the shield.

Now, let's connect the motor to the A channel, following the image below. The channels are marked next to the screw terminals on the shield.



Connecting a power source and DC motor.

Finally, we can connect the USB cable to the computer.

Programming the Board

We will now get to the programming part of this tutorial.

First, let's take a look at some key commands in the code. We are actually not using a library, as the operation is very basic.

- `int directionPin = 12;` - assign direction pin.
- `int pwmPin = 3;` - assign PWM (work duty) pin.
- `int brakePin = 9;` - assign brake pin.
- `digitalWrite(directionPin, state)` - sets the direction of the pin by using HIGH or LOW states.
- `digitalWrite(brakePin, state)` - release or activate brakes, using HIGH or LOW states.
- `analogWrite(pwmPin, 30)` - write a value between 0-100 to set the work duty.
- `directionState = !directionState` - a boolean that switches every time the loop is run.

The sketch can be found in the snippet below. Upload the sketch to the board.

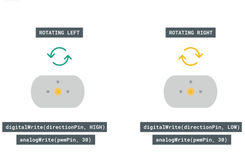
COPY

```
int directionPin = 12;
int pwmPin = 3;
int brakePin = 9;

//uncomment if using channel B, and remove above definitions
//int directionPin = 13;
//int pwmPin = 11;
//int brakePin = 8;

//boolean to switch direction
bool directionState;

void setup() {

//define pins
pinMode(directionPin, OUTPUT);
pinMode(pwmPin, OUTPUT);
pinMode(brakePin, OUTPUT);

}

void loop() {

//change direction every loop()
directionState = !directionState;

//write a low state to the direction pin (13)
if(directionState == false){
  digitalWrite(directionPin, LOW);
}

//write a high state to the direction pin (13)
else{
  digitalWrite(directionPin, HIGH);
```

Testing It Out

After we have uploaded the code, the program will start running immediately. If everything is working correctly, the motor should start spinning as soon as the program finishes uploading.

The expected outcome is that the motor spins in one direction for 2 seconds, with the work duty set to `30` (quite low), and brakes disengaged.



Switching the direction.

When 2 seconds have passed, the brakes are activated and work duty is set to 0. This means the motor comes to a full stop.

Motor at full stop.

Troubleshoot

If the code is not working, there are some common issues we can troubleshoot:

- We have not connected the circuit properly. We can double check this by going back to the circuit at the top of this page.
- We have connected the motor to the wrong channel. Remember that the pin definitions in the code need to match the channel you connect it to.
- The external power source is not working: double check that your batteries are working, and re-charge them if needed.

Conclusion

In this tutorial, we have learned how to control a DC motor using the Motor Shield Rev3. With the **brake, pwm and direction**, you now have a lot more options when it comes to motor control. Having this type of control allows you to create many different cool projects, without having to create complicated programs or advanced circuits!