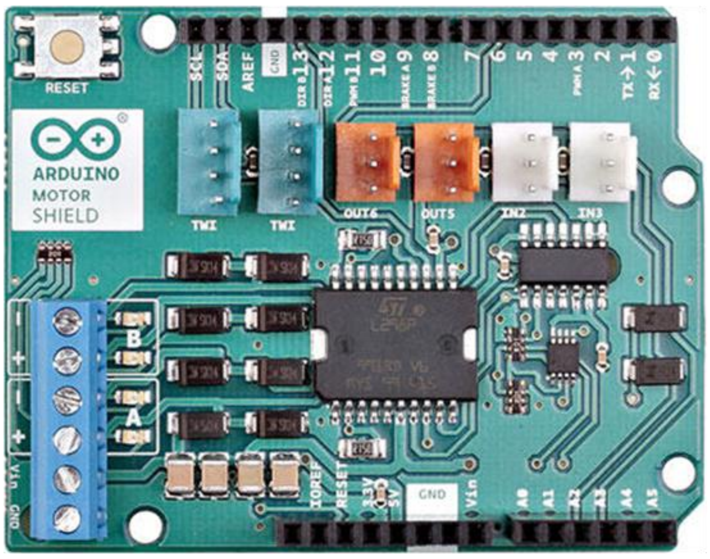
MISE EN OEUVRE DE LA CARTE MOTOR SHIELD ARDUINO

Site de référence: http://www.arduino.org/products/shields/arduino-motor-shield

The Arduino Motor Shield is based on the L298, which is a dual full-bridge driver designed to drive inductive loads such as DC and stepping motors, relays and solenoids. It lets you drive two DC motors with your Arduino board, controlling the speed and direction of each one independently.



Technical specs

Operating Voltage	5V to 12V
Motor controller	L298P, Drives 2 DC motors or 1 stepper motor
Max current	2A per channel or 4A max (with external power supply)
Current sensing	1.65V/A
Free running stop and brake function	

Power

The Arduino Motor Shield must be powered only by an external power supply. Because the L298 IC mounted on the shield has two separate power connections, one for the logic and one for the motor supply driver. The required motor current often exceeds the maximum USB current rating.

External (non-USB) power can come either from an AC-to-DC adapter (wall-wart) or battery. The adapter can be connected by plugging a 2.1mm center-positive plug into the Arduino's board power jack on which the motor shield is mounted or by connecting the wires that lead the power supply to the Vin and GND screw terminals, taking care to respect the polarities.

To avoid possible damage to the Arduino board on which the shield is mounted, we reccomend using an external power supply that provides a voltage between 7 and 12V. If your motor require more than 9V we recommend that you separate the power lines of the shield and the Arduino board on which the shield is mounted. This is possible by cutting the *"Vin Connect"* jumper placed on the back side of the shield. The absolute limit for the Vin at the screw terminals is 18V.

The power pins are as follows:

- Vin on the screw terminal block, is the input voltage to the motor connected to the shield. An external power supply connected to this pin also provide power to the Arduino board on which is mounted. By cutting the *"Vin Connect"* jumper you make this a dedicated power line for the motor.
- GND Ground on the screw terminal block.

The shield can supply 2 amperes per channel, for a total of 4 amperes maximum.

Input and Output

This shield has two separate channels, called A and B, that each use 4 of the Arduino pins to drive or sense the motor. In total there are 8 pins in use on this shield. You can use each channel separately to drive two DC motors or combine them to drive one bipolar stepper motor. The shield's pins, divided by channel are shown in the table below:

Function	pins per Ch. A	pins per Ch. B
Direction	D12	D13
PWM	D3	D11
Brake	D9	D8
Current Sensing	A0	A1

If you don't need the Brake and the Current Sensing and you also need more pins for your application you can disable this features by cutting the respective jumpers on the back side of the shield.
The additional sockets on the shield are described as follow:

- Screw terminal to connect the motors and their power supply.
- 2 TinkerKit connectors for two Analog Inputs (in white), connected to A2 and A3.
- 2 TinkerKit connectors for two Aanlog Outputs (in orange in the middle), connected to PWM outputs on pins D5 and D6.
- 2 TinkerKit connectors for the TWI interface (in white with 4 pins), one for input and the other one for output.

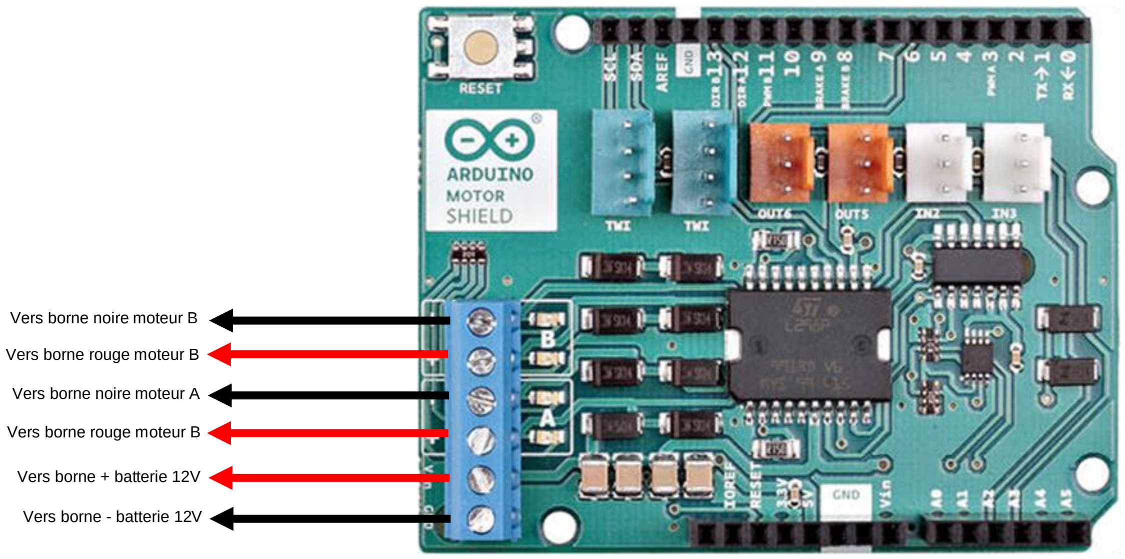
Motors Connection

Brushed DC motor. You can drive two Brushed DC motors by connecting the two wires of each one in the (+) and (-) screw terminals for each channel A and B. In this way you can control its direction by setting HIGH or LOW the DIR A and DIR B pins, you can control the speed by varying the PWM A and PWM B duty cycle values. The Brake A and Brake B pins, if set HIGH, will effectively brake the DC motors rather than let them slow down by cutting the power. You can measure the current going through the DC motor by reading the SNS0 and SNS1 pins. On each channel will be a voltage proportional to the measured current, which can be read as a normal analog input, through the function analogRead() on the analog input A0 and A1. For your convenience it is calibrated to be 3.3V when the channel is delivering its maximum possible current, that is 2A.

Physical Characteristics

The maximum length and width of the Motor Shield PCB are 2.7 and 2.1 inches respectively. Four screw holes allow the board to be attached to a surface or case. Note that the distance between digital pins 7 and 8 is 160 mil (0.16"), not an even multiple of the 100 mil spacing of the other pins.

Exemple de montage


Vers borne noire moteur B
Vers borne rouge moteur B
Vers borne noire moteur A
Vers borne rouge moteur B
Vers borne + batterie 12V
Vers borne - batterie 12V
ARDUINO MOTOR SHIELD
RESET

Progamme associé que vous pouvez trouver sur le site Internet.

```
// Commande 2 moteur CC

// -------Déclaration des broches E/S--------

const int vitmotA=3; // constante de vitesse moteur A pour la broche 3
const int sensmotA=12; // constante de vitesse moteur A pour la broche 12
const int freinmotA=9; // constante freinage moteur A pour la broche 9
const int intensitemotA=A0; // constante lecture intensité moteur A broche A0

const int vitmotB=11; // constante de vitesse moteur B pour la broche 11
const int sensmotB=13; // constante de vitesse moteur B pour la broche 13
const int freinmotB=8; // constante freinage moteur B pour la broche 9
const int intensitemotB=A1; // constante lecture intensité moteur B broche A0


void setup() {
  // put your setup code here, to run once:

Serial.begin(115200); // vitesse lecture

// ----- Broches en sortie numériques ----

pinMode (vitmotA,OUTPUT); // Broche vitesse moteur A en sortie
pinMode (freinmotA,OUTPUT); // Broche frein moteur A en sortie
pinMode (sensmotA,OUTPUT); // Broche sens moteur A en sortie

pinMode (vitmotB,OUTPUT); // Broche vitesse moteur B en sortie
pinMode (freinmotB,OUTPUT); // Broche vitesse moteur B en sortie
pinMode (sensmotB,OUTPUT); // Broche sens moteur B en sortie
// ----  codes initialisation  variables ----

digitalWrite(vitmotA,LOW); // vitesse moteur A à l'arrêt
digitalWrite(sensmotA,LOW);// sens moteur A
digitalWrite(freinmotA,LOW); // frein moteur A off

digitalWrite(vitmotB,LOW); // vitesse moteur B à l'arrêt
digitalWrite(sensmotB,LOW);// sens moteur B
digitalWrite(freinmotB,LOW); // frein moteur B off
}

void loop() {
  // put your main code here, to run repeatedly:


// -------- test initial moteur A -----------


// Sens 1 moteur A
digitalWrite(sensmotA,LOW); // sens 1 moteur A
digitalWrite(vitmotA,HIGH); // vitesse moteur A maximale
delay(5000);  // 5 secondes
digitalWrite(vitmotA,LOW); // vitesse moteur A arrêt

// Sens 2 moteur A
digitalWrite(sensmotA,HIGH); // sens 1 moteur A
digitalWrite(vitmotA,HIGH); // vitesse moteur A maximale
delay(5000);  // 5 secondes
Serial.println(analogRead(intensitemotA));  // lire l'intensité moteur A
delay(1000); // 1 seconde
digitalWrite(vitmotA,LOW); //  vitesse moteur A arrêt


// -------- test initial moteur B -----------


// Sens 1 moteur B
digitalWrite(sensmotB,LOW); // sens 1 moteur B
digitalWrite(vitmotB,HIGH); // vitesse moteur B maximale
delay(5000);  // 5 secondes
digitalWrite(vitmotB,LOW); // vitesse moteur B arrêt

// Sens 2 moteur B
digitalWrite(sensmotB,HIGH); // sens 1 moteur B
digitalWrite(vitmotB,HIGH); // vitesse moteur B maximale
delay(5000);  // 5 secondes
Serial.println(analogRead(intensitemotB));  // lire l'intensité moteur B
delay(1000); // 1 seconde
digitalWrite(vitmotB,LOW); //  vitesse moteur B arrêt


// --------    test vitesse variable moteur A --------

      for (int i=0; i<=255; i++) { // de i=0 à 255
        analogWrite(vitmotA,i); // PWM croissant
        delay(50); // pause 0,05s
        Serial.println(analogRead(intensitemotA));
      }

       for (int i=0; i<=255; i++) { // de i=0 à 255
        analogWrite(vitmotA,255-i); // PWM décroissant
        delay(50); // pause 0,05s
        Serial.println(analogRead(intensitemotA));
      }


// --------    test vitesse variable moteur B --------

      for (int i=0; i<=255; i++) { // de i=0 à 255
        analogWrite(vitmotB,i); // PWM croissant
        delay(50); // pause 0,05s
        Serial.println(analogRead(intensitemotB));
      }

       for (int i=0; i<=255; i++) { // de i=0 à 255
        analogWrite(vitmotB,255-i); // PWM décroissant
        delay(50); // pause 0,05s
        Serial.println(analogRead(intensitemotB));
      }
}
```