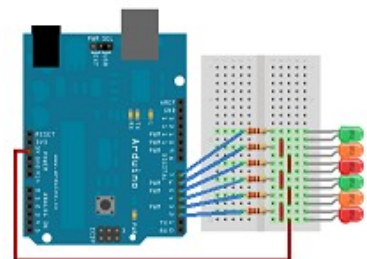


# TP introduction Arduino

## Table des matières

|   |    |
|---|----|
| 1. Installation et configuration de la carte Arduino.....                                   | 2  |
| 1.1. Carte Arduino.....   | 2  |
| 1.2. Structure d'un programme.....  | 3  |
| 1.3. Installation de l'interface de développement.....                                      | 4  |
| 2. Clignotement d'une LED.....  | 4  |
| 3. Utilisation d'un interrupteur switch.....  | 6  |
| 4. Utilisation de l'entrée analogique.....  | 7  |
| 4.1. Variation de la fréquence d'allumage d'une LED en fonction de la tension d'entrée..... | 7  |
| 4.2. Mesure de l'éclairement.....   | 8  |
| 4.3. Mesure de la température.....  | 9  |
| 5. Commande d'un moteur à courant continu en PWM.....                                       | 10 |
| 6. Affichage d'une grandeur sur un afficheur LCD.....                                       | 13 |
| 7. Références.....  | 14 |

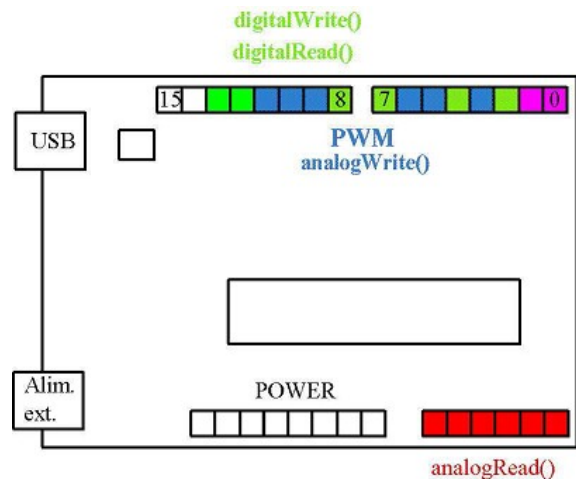
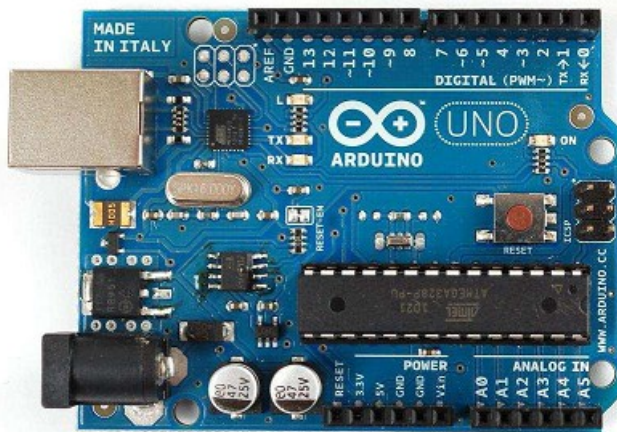
Ce TP de 10 heures est une introduction au projet SI (70 heures jusqu'au mois de mai). Il a pour but d'introduire l'utilisation de la carte Arduino. À l'issue des 10 heures, vous devrez rendre un compte rendu par binôme donnant les sources de vos programmes commentés, les mesures faites, une photo et/ou une vidéo des montages, les explications du fonctionnement des capteurs, de la variation de vitesse d'un moteur à courant continu...



# 1. Installation et configuration de la carte Arduino

## 1.1. Carte Arduino

Nous utiliserons une carte Arduino Uno. Elle emploie un microcontrôleur ATMEGA328P alimenté en 5 V. Il y a 14 entrées/sorties numériques dont 6 sont utilisables en PWM<sup>1</sup>. Il y a 6 entrées analogiques. Le microcontrôleur possède un CAN<sup>2</sup> avec 10 bits de résolution. Sur la carte, il y a un circuit qui permet de gérer facilement l'USB<sup>3</sup> qui peut alimenter la carte.



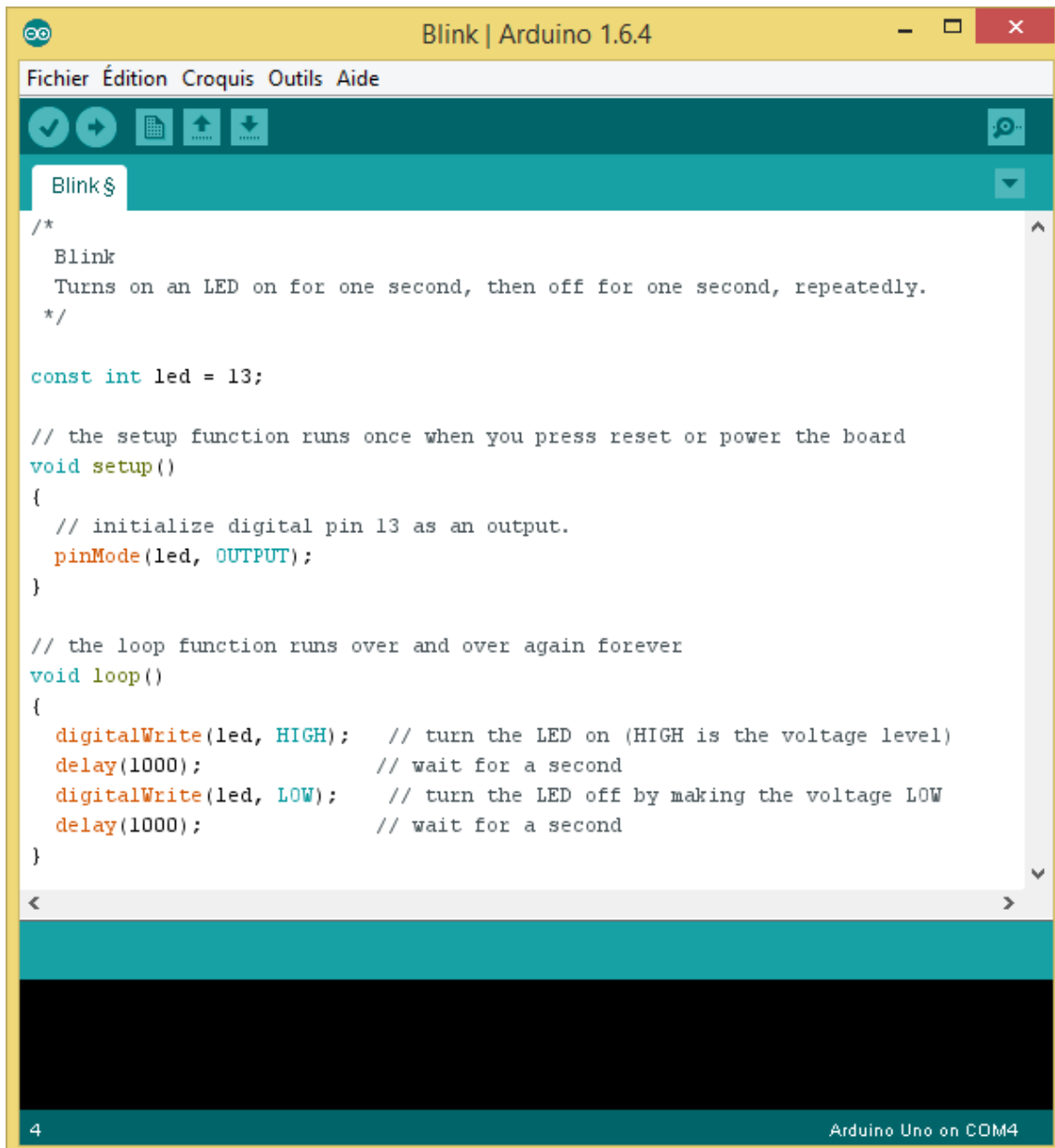
- Mémoire Flash 32 ko
- Mémoire RAM 2 ko
- Mémoire EEPROM 1 ko
- Fréquence d'horloge 16 MHz
- Courant max. E/S 40 mA

Pour en savoir plus, consultez la page : <http://arduino.cc/en/Main/ArduinoBoardUno>

1 Pulse Width Modulation (ou MLI pour modulateur de largeur d'impulsion)  
 2 Convertisseur Analogique Numérique  
 3 Universal Serial Bus

## 1.2. Structure d'un programme

Un exemple de programme est donné ci-dessous.



```
/*
 * Blink
 * Turns on an LED on for one second, then off for one second, repeatedly.
 */

const int led = 13;

// the setup function runs once when you press reset or power the board
void setup()
{
    // initialize digital pin 13 as an output.
    pinMode(led, OUTPUT);
}

// the loop function runs over and over again forever
void loop()
{
    digitalWrite(led, HIGH); // turn the LED on (HIGH is the voltage level)
    delay(1000);             // wait for a second
    digitalWrite(led, LOW);  // turn the LED off by making the voltage LOW
    delay(1000);             // wait for a second
}
```

- Le commentaire de description du programme, appelé sketch, débute par `/*` et se termine par `*/`.
- Puis il est placé la définition des constantes et des variables avec les instructions `const` et `int`.
- Vient ensuite la configuration des entrées / sorties avec l'instruction `void setup()`. La suite d'instructions est précédé de `{` et se termine par `}`.
- On définit par l'instruction `void loop()` la programmation des interactions et comportements. La suite d'instructions est précédé de `{` et se termine par `}`.

## 1.3. Installation de l'interface de développement

Pour utiliser l'interface de développement, [télécharger le logiciel](#) Arduino (version windows).

1. Ouvrir le logiciel Arduino.
2. Vérifier que la carte Arduino Uno est bien prise en compte par : Outils > Type de carte > Arduino Uno.
3. Brancher la carte sur un port USB. Vérifier que le port COM est bien configuré. Outils > Port série > COM X.
4. Vérifier que le port COM X est bien reconnu par windows : Panneau de Configuration > Système et sécurité > Gestionnaire de périphériques > Ports : Arduino Uno (COM X).

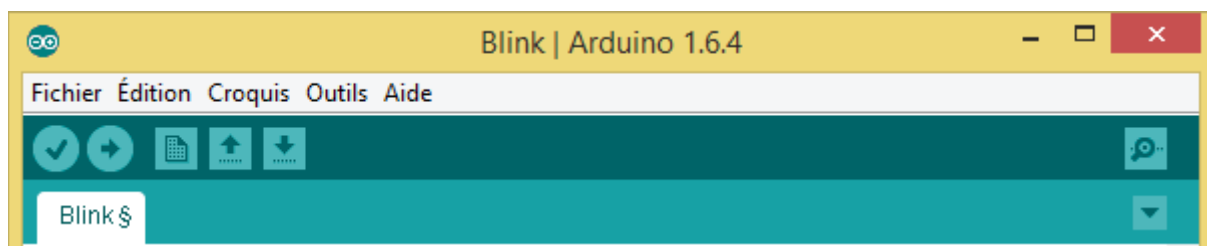
## 2. Clignotement d'une LED

Ouvrir le fichier Blink par : Fichier > Exemples > 01.Basics > Blink

Pour comprendre la syntaxe et le rôle d'une instruction, utiliser la page :

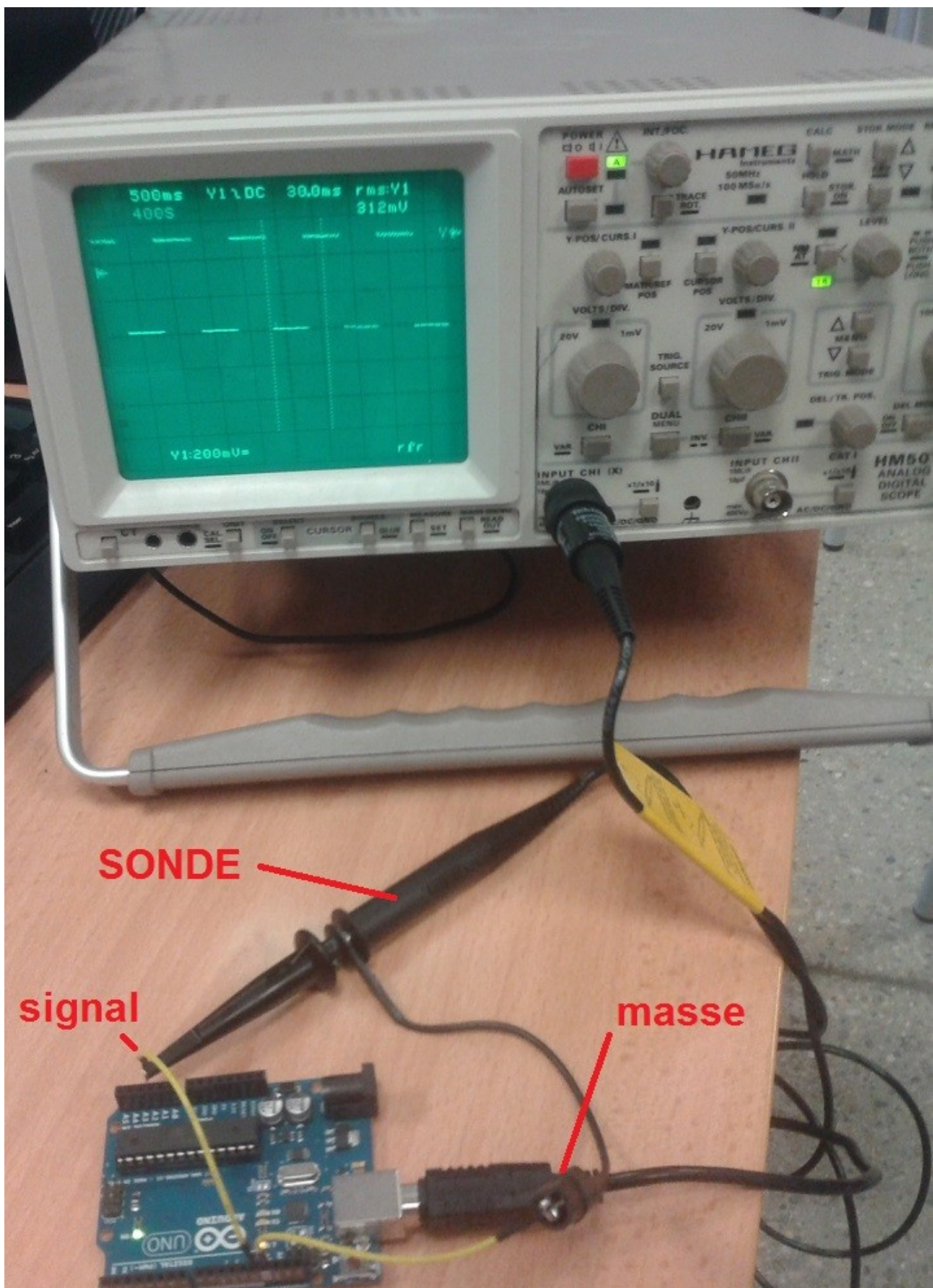
<http://arduino.cc/en/Reference/HomePage>

1. Modifier le programme de telle façon que la sortie sur laquelle on connectera la LED soit la sortie 11. Modifier les instructions pour que la période soit de 1 s et la durée d'allumage de la LED 100 ms.
2. Lancer l'exécution du programme en cliquant sur le bouton en dessous Édition :




3. Connecter une sonde de tension pour mesurer à l'oscilloscope les potentiels sur la sortie 11.
4. Connecter une LED avec une résistance en série sur une plaquette d'essais.
5. Déterminer la valeur de la résistance pour avoir un courant de 10 mA.
6. Réduire la durée d'allumage de la LED à 10 ms. Observer et conclure.
7. Modifier le programme pour que deux LED s'allument en opposition de phase comme dans le tableau suivant :

|       |         |         |
|-------|---------|---------|
| LED 1 | Allumée | Éteinte |
| LED 2 | Éteinte | Allumée |

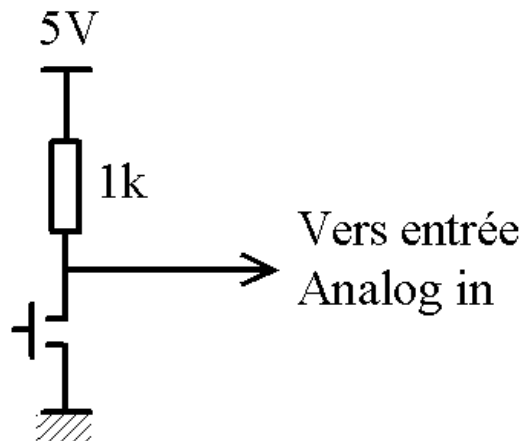




Pour voir le résultat, cliquer sur l'image 

### 3. Utilisation d'un interrupteur switch

Le switch sera connecté en série avec une résistance au + 5 V comme sur le montage suivant :

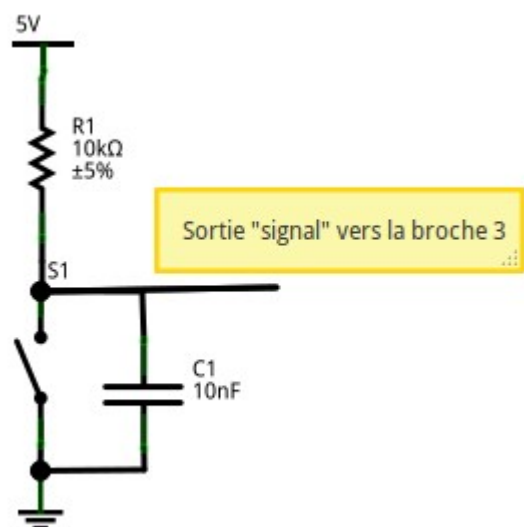


Souvent ils ont 4 pattes (comme sur l'image ci-contre). Si c'est le cas, les broches sont reliées deux à deux. Cela signifie qu'elles fonctionnent par paire. Il faut donc se méfier lorsque vous le branchez sinon vous obtiendrez le même comportement qu'un fil (si vous connectez deux broches reliées). Utilisez un multimètre pour déterminer quels broches sont distinctes. Pour ne pas se tromper, on utilise en général deux broches qui sont opposées sur la diagonale du bouton.

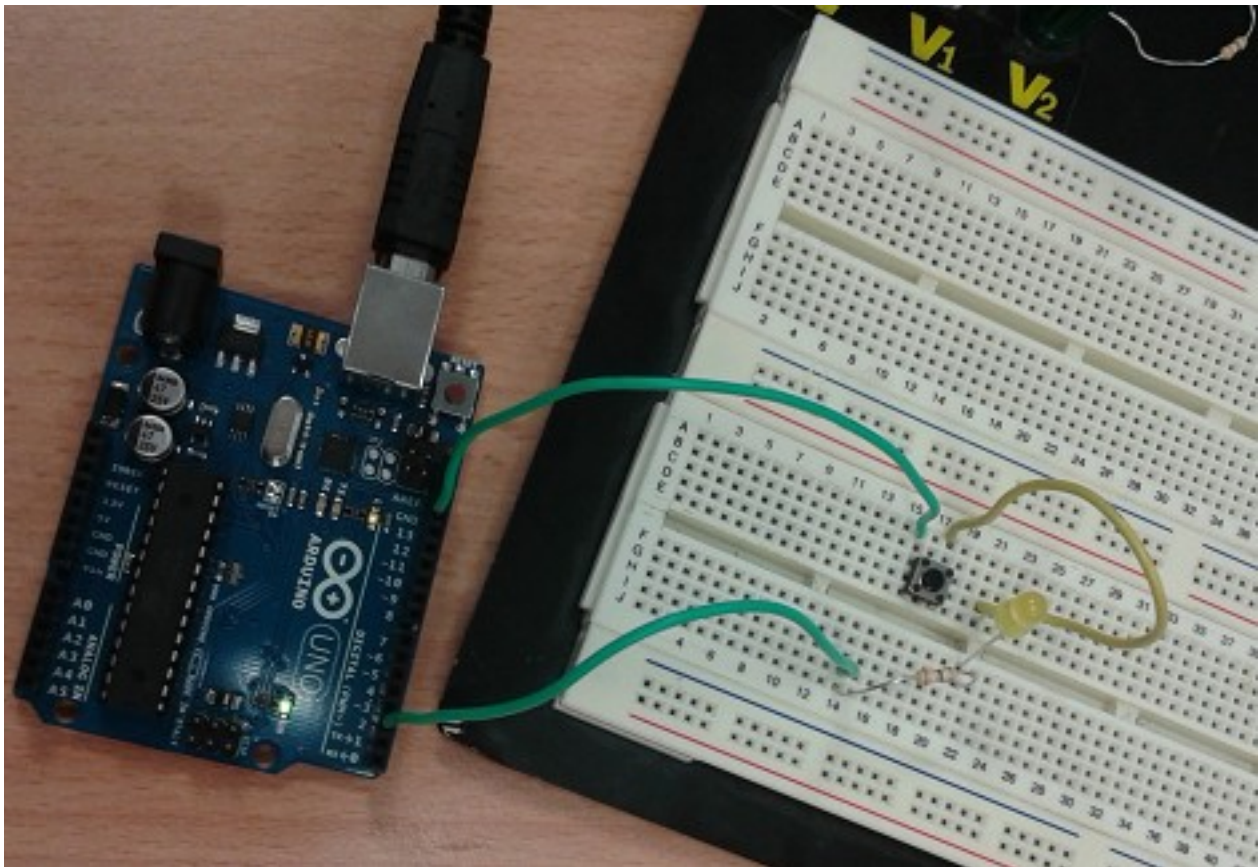


Les boutons ne sont pas des systèmes mécaniques parfaits. Du coup, lorsqu'un appui est fait dessus, le signal ne passe pas immédiatement et proprement de 5V à 0V. En l'espace de quelques millisecondes, le signal va "sauter" entre 5V et 0V plusieurs fois avant de se stabiliser. Il se passe le même phénomène lorsque l'utilisateur relâche le bouton. Ce genre d'effet n'est pas désirable, car il peut engendrer des parasites au sein du programme (si vous voulez détecter un appui, les rebonds vont vous en générer une dizaine en quelques millisecondes, ce qui peut-être très gênant dans le cas d'un compteur par exemple).

Pour atténuer ce phénomène, on utilise un condensateur en parallèle avec le bouton. Ce composant servira ici "d'amortisseur" qui absorbera les rebonds. Le condensateur, initialement chargé, va se décharger lors de l'appui sur le bouton. S'il y a des rebonds, ils seront encaissés par le condensateur durant cette décharge. Il se passera le phénomène inverse (charge du condensateur) lors du relâchement du bouton. Ce principe est illustré ci-contre :



Lorsque l'entrée Analog In sera au niveau 0, la LED sera éteinte et réciproquement. On utilisera les instructions **digitalRead** et **digitalWrite**.



1. Écrire un programme pour que la carte Arduino détecte la pression du bouton poussoir.
2. Modifier ce programme pour que la carte Arduino commande la LED lors de la pression du bouton.

## 4. Utilisation de l'entrée analogique

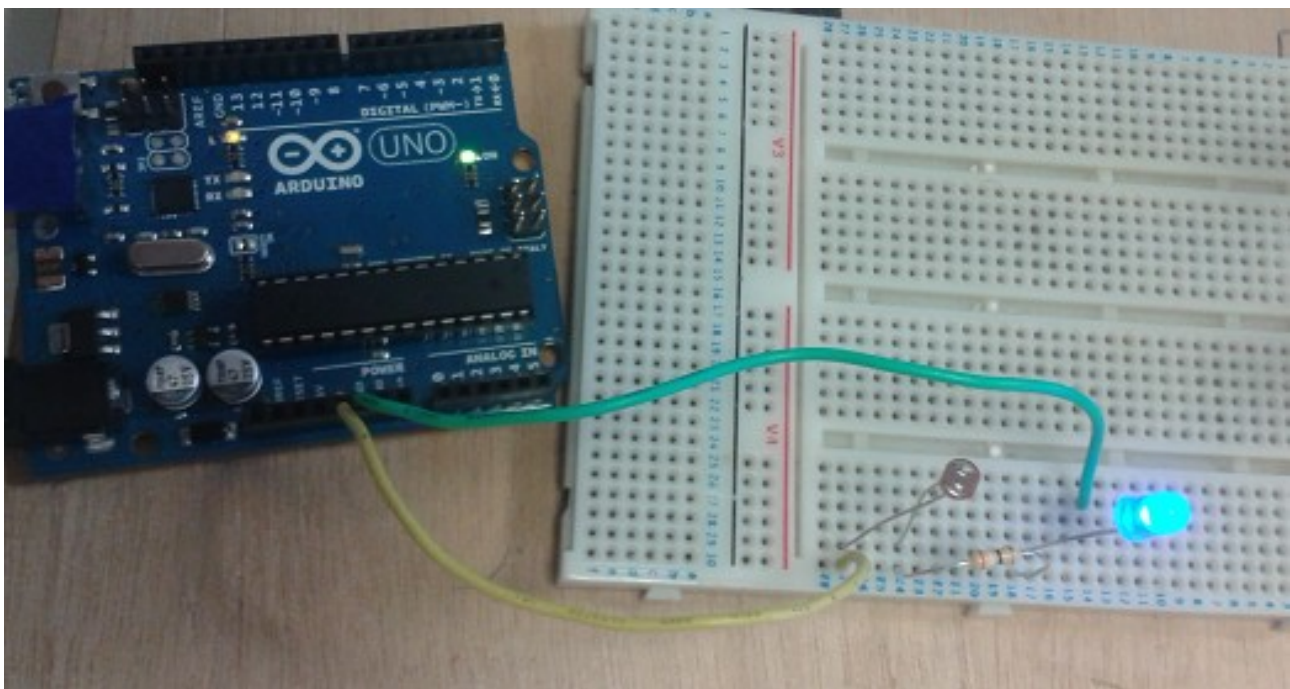
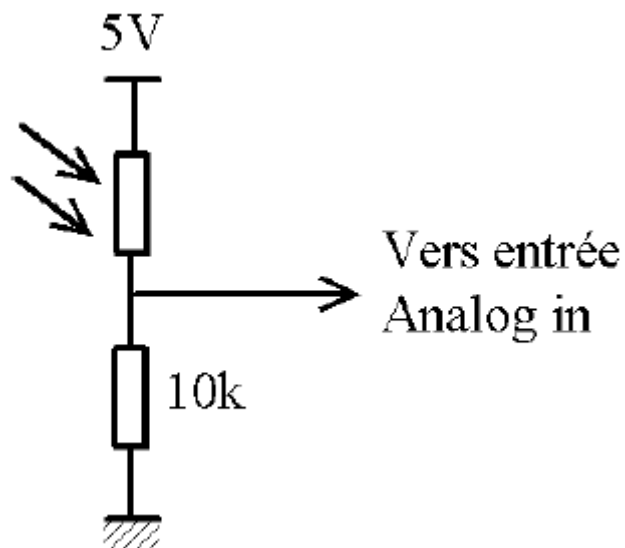
### 4.1. Variation de la fréquence d'allumage d'une LED en fonction de la tension d'entrée

On placera un potentiomètre de 10k entre le +5V et la masse et le curseur sera connecté sur l'entrée Analog IN A0. On utilisera l'instruction **YY = analogRead (XX)**, XX étant la valeur lue sur l'entrée Analog IN A0. L'instruction **delay (YY)** permettra de faire varier la fréquence de commande de la LED.


## 4.2. Mesure de l'éclairement

Le capteur de luminosité utilisé est le LDR VT935G (ou équivalent série VT900).

1. Télécharger la [documentation du capteur](#) et comprendre son fonctionnement.
2. Indiquer le principe de fonctionnement du capteur.
3. Indiquer la tension maximale d'utilisation.
4. Indiquer la signification de la sensibilité.
5. Proposer un montage permettant de détecter le passage d'un objet.
6. Modifier ce montage pour que la détection du passage soit gérée par la carte Arduino (utiliser la liaison série du port USB et l'objet Serial de la bibliothèque Arduino).



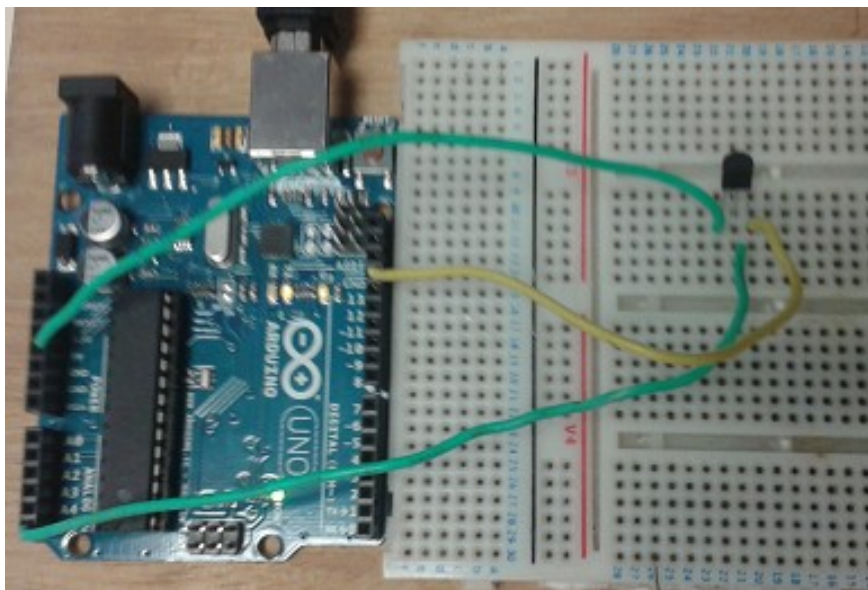
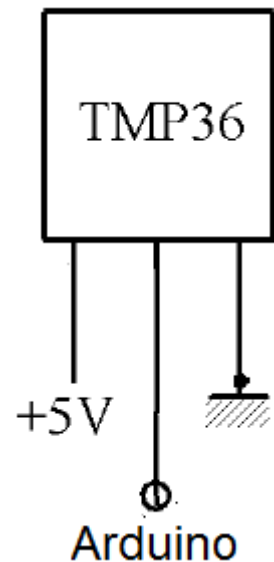


Pour voir le résultat, cliquer sur l'image 

### 4.3. Mesure de la température

Le capteur de température utilisé est le TMP 36.

1. Télécharger la [documentation du capteur](#) et comprendre son fonctionnement.
2. Indiquer le principe de fonctionnement du capteur.
3. Indiquer la tension de sortie pour une température nulle.
4. Donner la sensibilité du capteur.
5. Alimenter le capteur à partir de la carte Arduino.
6. Connecter sa sortie sur une entrée analogique.
7. Afficher sa valeur sur le port série.
8. Vérifier la compatibilité de la valeur numérique lue avec la température et la valeur de la tension continue en sortie du capteur.
9. Dire si la tension de sortie est stable.



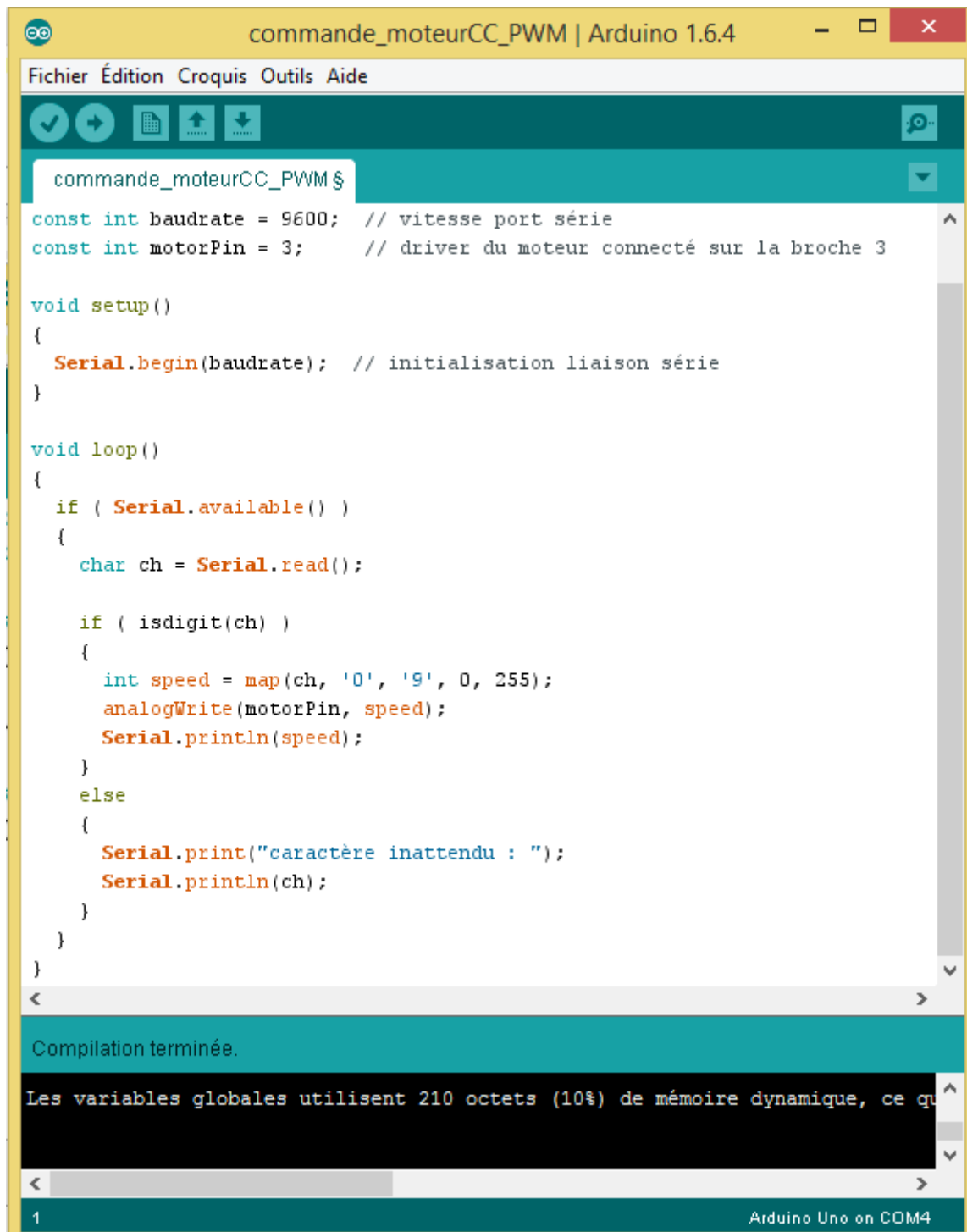
10. Déterminer la valeur de R pour avoir une fréquence de coupure de l'ordre de 10 Hz. Voir le professeur pour le calcul et la ressource cours sur le filtrage.

On peut également calculer la moyenne de plusieurs acquisitions successives : faire 3 acquisitions séparées de 100 ms (avec l'instruction **delay**) puis calculer sa somme divisée par 3.

Ainsi, on obtient : 
$$V_s = \frac{Acq(0) + Acq(100) + Acq(200)}{3}$$

## 5. Commande d'un moteur à courant continu en PWM

Le programme suivant est utilisé pour commander en PWM la vitesse d'un moteur en courant continu. La vitesse est entrée au clavier par un caractère *ch* compris entre 0 et 9 qui est alors converti en valeur de 0 à 255 avec l'instruction *map*. Pour voir la valeur, ouvrir le Moniteur Série par Outils > Moniteur Série.



```

commande_moteurCC_PWM | Arduino 1.6.4
Fichier Édition Croquis Outils Aide

const int baudrate = 9600; // vitesse port série
const int motorPin = 3;    // driver du moteur connecté sur la broche 3

void setup()
{
  Serial.begin(baudrate); // initialisation liaison série
}

void loop()
{
  if ( Serial.available() )
  {
    char ch = Serial.read();

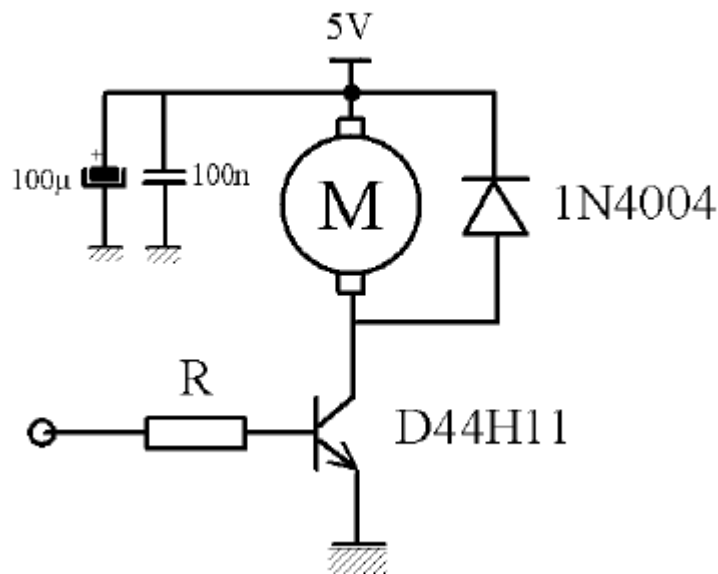
    if ( isdigit(ch) )
    {
      int speed = map(ch, '0', '9', 0, 255);
      analogWrite(motorPin, speed);
      Serial.println(speed);
    }
    else
    {
      Serial.print("caractère inattendu : ");
      Serial.println(ch);
    }
  }
}

Compilation terminée.

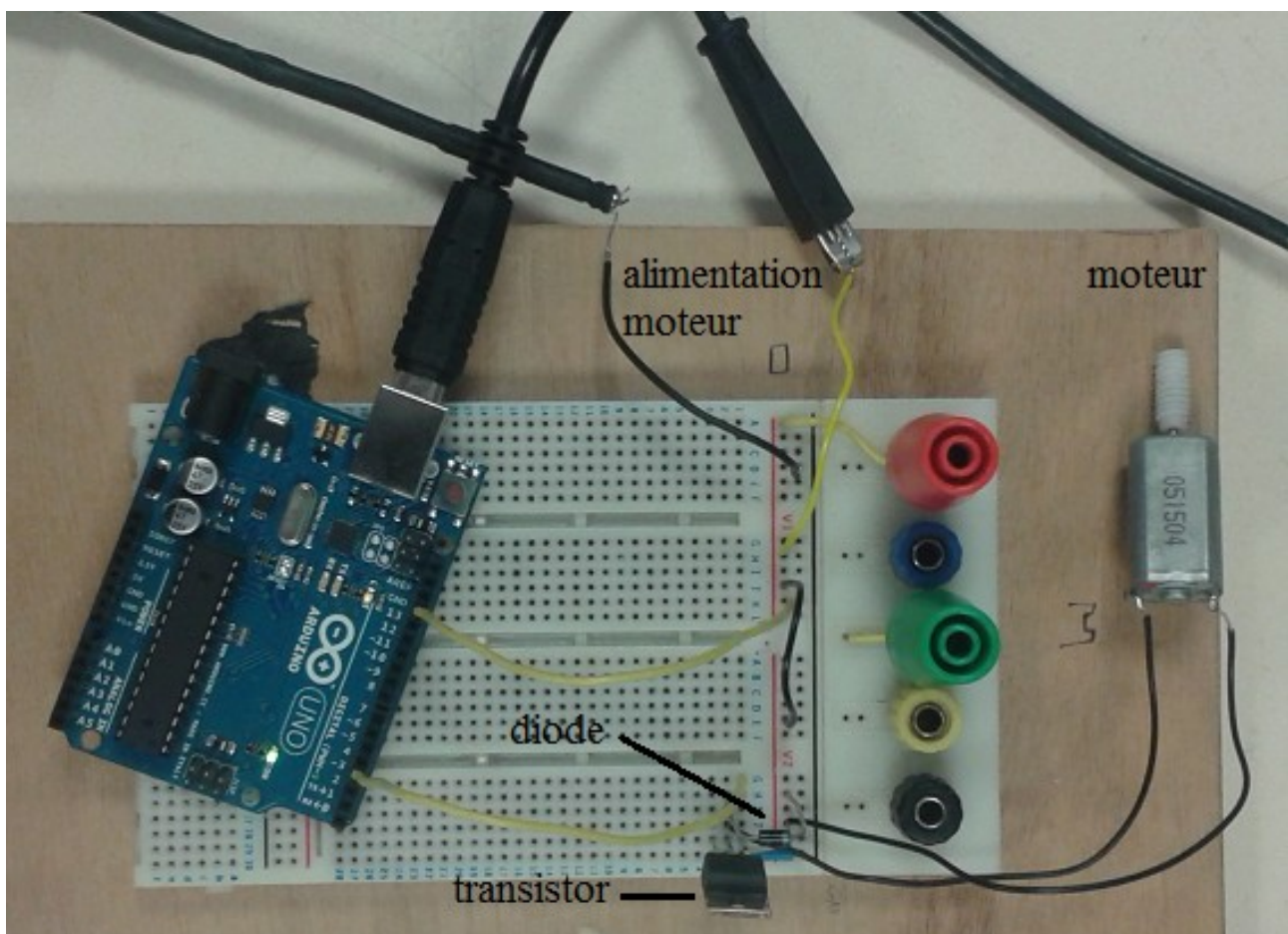
Les variables globales utilisent 210 octets (10%) de mémoire dynamique, ce qu
1
Arduino Uno on COM4

```

On connectera la sortie Arduino utilisée au montage suivant :

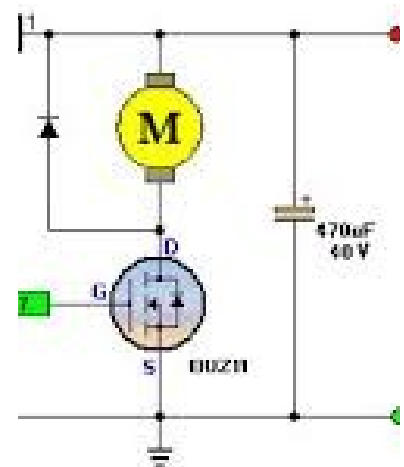
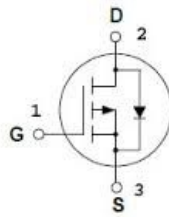
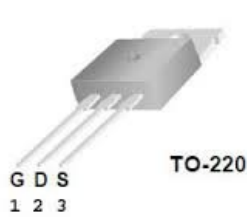


Une alimentation extérieure délivre la tension 5V nécessaire pour alimenter le moteur. Attention au sens de branchement de la capacité de 100  $\mu$ F car elle est polarisée (pole positif et pole négatif). Les masses de l'alimentation et de l'Arduino doivent être reliées.



1. Déterminer la valeur de la résistance R à partir du gain en courant  $\beta_{\min}$  ou  $hFE_{\min}$  du [transistor D44H11](#), ou équivalent, et du courant circulant dans le moteur qu'on estime à 100 mA. Voir le professeur pour le calcul et la ressource cours sur les transistors. Autre solution

utiliser un transistor MOS comme le IRF520N.



2. Indiquer le rôle de la diode et des capacités de découplages ([aide en ligne.](#))
3. Observer à l'oscilloscope Mesurer la tension délivrée par l'Arduino et la tension  $V_{CE}$  du transistor.
4. Mesurer au voltmètre RMS\_True la tension délivrée par l'Arduino et la tension  $V_{CE}$  du transistor.

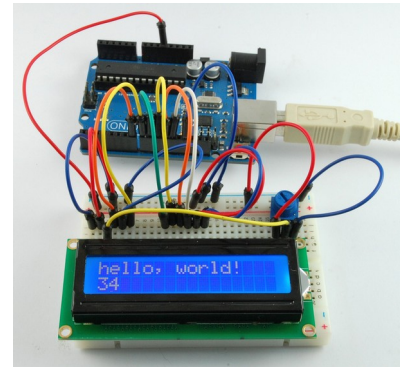
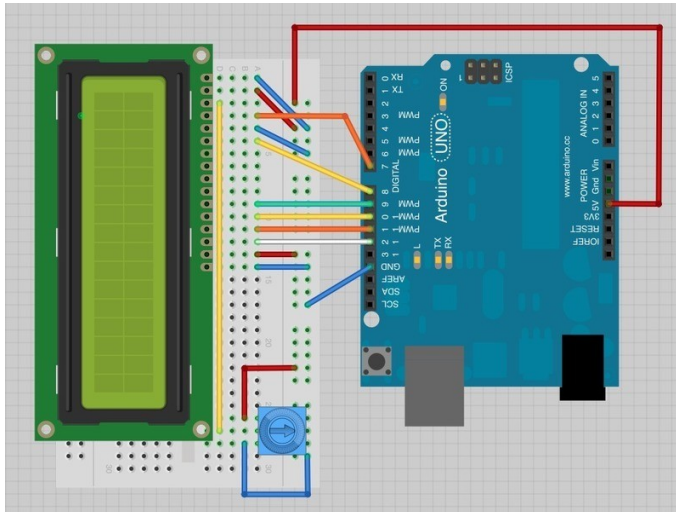
Pour voir le résultat, cliquer sur l'image



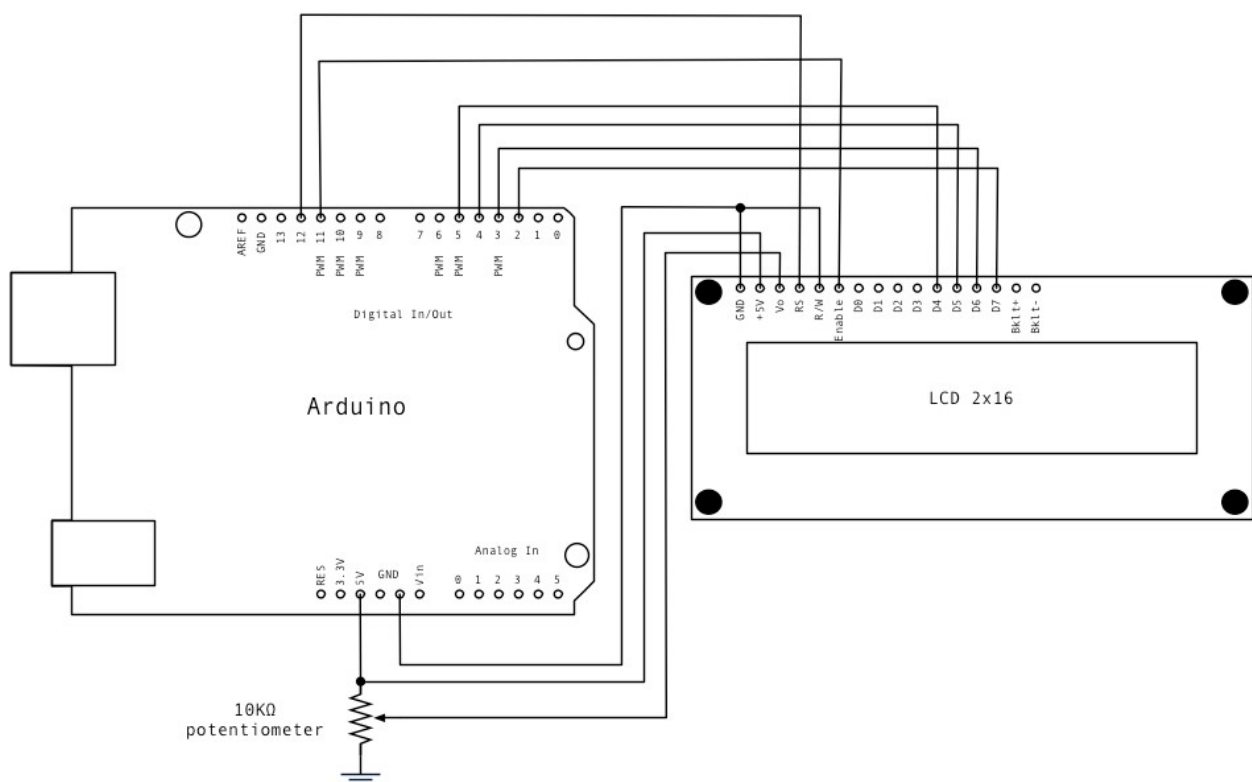
## 6. Affichage d'une grandeur sur un afficheur LCD

L'afficheur utilisé est décrit sur [cette page](#).

1. Ouvrir dans l'interface Arduino ouvrir le programme HelloWorld :  
Fichiers > Exemples > LiquidCrystal > HelloWorld.



2. Câbler l'afficheur LCD en suivant le schéma de câblage.



3. Tester le programme HelloWorld.
4. Connecter le capteur de température à une entrée analogique de l'arduino.

5. Modifier le programme afin d'afficher la valeur numérique de l'entrée analogique lue ?  
Pour convertir la grandeur affichée en degré, on tiendra compte du fait que le convertisseur analogique numérique interne délivre un mot de 10 bits pour une tension variant de 0 à 5 V en entrée.
6. Consulter la [documentation du capteur](#) TMP36 pour afficher la valeur en degré sur l'afficheur.
7. Ajouter un filtre lisseur moyennneur comme indiqué au paragraphe 4.3.

## 7. Références

- [Tutoriel Arduino](#)
- [Manuel de référence Aduino](#)
- [langage C](#)
- [langage C++](#)
- [la résistance](#)
- [la diode](#)
- [le Transistor](#)
- [Travaux Pratique Arduino](#)
- [Référence Langage Arduino](#)
- [les datashhets](#)