Projet 1 : Prise en main du Matériel Arduino UNO,

1. Présentation et programmation de la carte Arduino

Arduino est un projet créé par une équipe de développeurs composée de six individus : Massimo Banzi, David Cuartielles, Tom Igoe, Gianluca Martino, David Mellis et Nicholas Zambetti. Cette équipe a créé le "système Arduino". C'est un outil qui va permettre aux débutants, amateurs ou professionnels de créer des systèmes électroniques plus ou moins complexes

1.1 Le matériel : Arduino UNO

C'est un circuit imprimé comportant tous les composants électroniques nécessaires pour faire fonctionner un microcontrôleur (Atmega 328) associé à une interface USB lui permettant de communiquer avec un ordinateur..

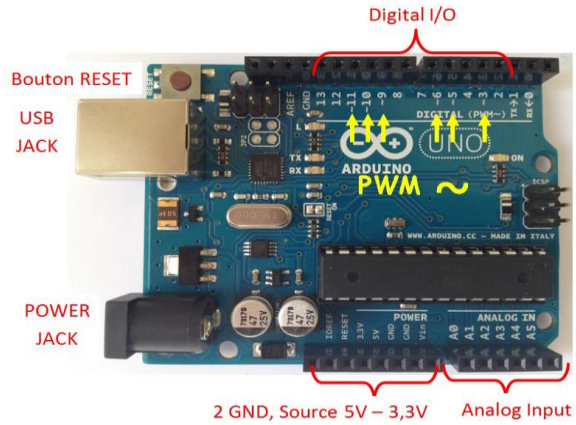


Figure 1 – Description de la Carte Arduino "Uno"

– Microcontroller : ATmega328
– Operating Voltage : 5v
– Input Voltage (recommended) : 7-12 v
– Input Voltage (limits) : 6-20 v
– DC Current per I/O Pin : 40mA
– DC Current for 3.3V Pin :50mA
– Flash Memory :32 KB
– Clock Speed : 16MHz
– Pins assignements :
– Analog read(A0-A5)
– Analog write[PWM] (3,5,6,9,10,11)
– Digital read (2-19)
– Digital write (2-19)

1.2 Le logiciel Arduino

Arduino IDE (Integrated Development Environment). Le logiciel est gratuit et open source dont la simplicité d'utilisation est remarquable .Ce logiciel va nous permettre de programmer la carte Arduino pour :

– Réaliser l'interfaçage avec Matlab/simulink

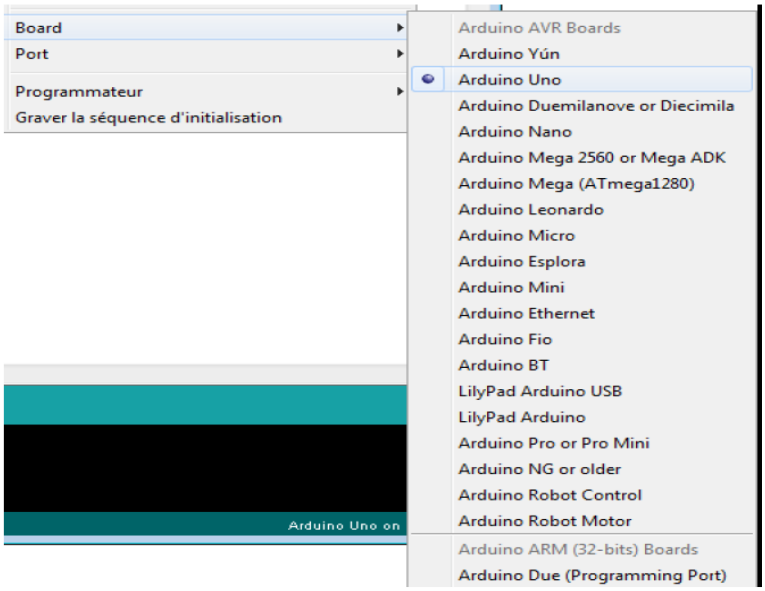
– Implémenter la commande directement sur la carte.

Lancer le logiciel Arduino

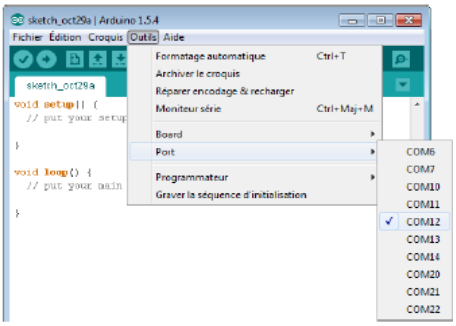
Figure 2 – L'interface du logiciel Arduino

Relier la carte Arduino UNO à votre ordinateur à l'aide du câble US

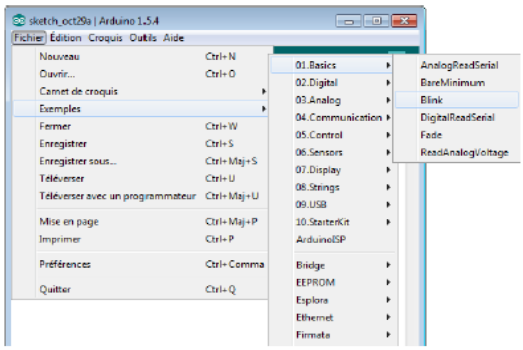
Sélectionner la carte Arduino UNO sur le logiciel Arduino



Sélectionner le port de communication utilisé par votre machine pour dialoguer avec la carte Arduino UNO



Editer le programme blink.(Fichier/Exemples/01 Basics/Blink). Ce programme fait clignoter la DEL **L**



Compiler le programme blink

Télécharger le programme blink dans la carte Arduino UNO

Analyse de programme

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

  This example code is in the public domain.
*/

// Pin 13 has an LED connected on most Arduino boards.
// give it a name:
int led = 13;

// the setup routine runs once when you press reset:
void setup() {
  // initialize the digital pin as an output.
  pinMode(led, OUTPUT);
}

// the loop routine 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
}
```

/* … */ : commentaires — présentation générale du programme

// commentaires

Déclaration de la variable **led** de type int. On donne la valeur **13** à la variable **led**. Ainsi le terme led est équivalent à la valeur 13
Rappelons que la led L est connectée à la broche 13 de la carte

// commentaires

Void setup() Configuration des entrées/sorties

pinMode(**led**, OUTPUT) = pinMode(**13**, OUTPUT) car la variableled=13.
La broche digitale 13 est paramétrée en sortie
http://arduino.cc/en/Reference/pinMode

// commentaires

Void loop(), instructions exécutées en boucle

digitalWrite(**led**, HIGH)= digitalWrite(**13**, HIGH) car la variable led=13 après cette instruction la broche 13 est positionnée au niveau haut de tension la DEL s'allume
http://arduino.cc/en/Reference/DigitalWrite
delay(**1000**); faire une temporisation de **1000**ms la diode restera allumée 1s
http://arduino.cc/en/Reference/Delay
digitalWrite(led, LOW); la broche 13 est positionnée au niveau bas de tension la DEL s'éteint

delay(**1000**); faire une temporisation de 1s la diode s'éteint 1 s

Nous allons maintenant réaliser le schéma suivant :

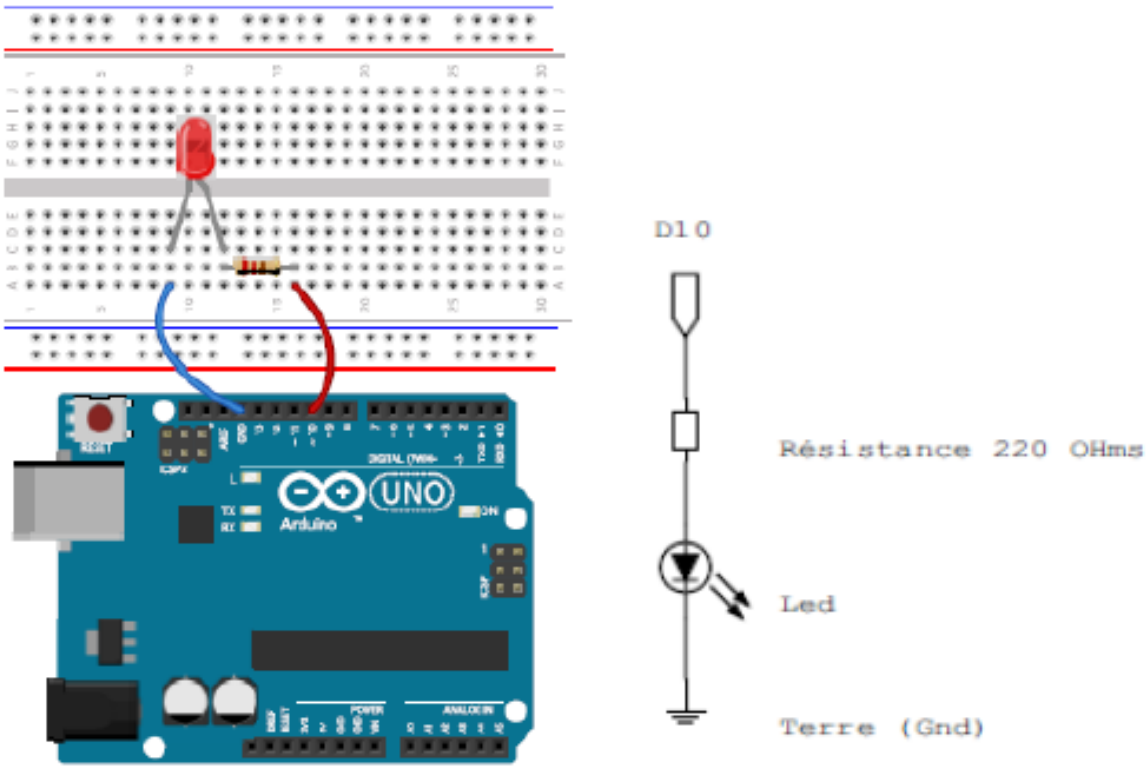


Figure 1 Connections Led et arduino

Compiler le code suivant :

```
void setup() {
// Initialise la broche 10 comme sortie
pinMode(10, OUTPUT);
```

```
// Ouvre le port série à 9600 bauds
Serial.begin(9600);
}
void loop() {
digitalWrite(10, HIGH); // allume la LED
delay(500); // attend 500ms
digitalWrite(10, LOW); // éteint la LED
delay(500); // attend 500ms
}
```

On ajoute deux leds connectées aux branches 11 et 12

a) Modifier le code précédent pour faire clignoter les trois leds en même temps.

b) Modifier le code pour faire clignoter les trois leds en alternance.

c) Dans cet exercice on demande d'écrire un programme pour réaliser un feu tricolore avec trois LED (une verte, une orange, une rouge) qui devront être allumées comme suit :

- Orange allumée pendant 1 seconde

- Rouge allumée pendant 4 secondes

- Verte allumée pendant 4 secondes

d) Ecrire un programme qui permet de clignoter une led 15 fois.

e) Compiler le code suivant et constater le résultat

```
int ledPin = 10;    //On renomme la broche 10 en "ledPin"
int timer = 100;   //On définit une durée de 0,1 seconde pour la variable timer

void setup() { pinMode(ledPin, OUTPUT); }

void loop() {
   // LED à 0%.
  analogWrite(ledPin, 0);
 delay(timer);
   // LED à 19.6%.
   analogWrite(ledPin, 50);
   delay(timer);
   // LED à 39.2%.
   analogWrite(ledPin, 100);
```

```
    delay(timer);
    // LED à 58.8%.
    analogWrite(ledPin, 150);
    delay(timer);
    // LED à 78.4%.
    analogWrite(ledPin, 200);
    delay(timer);
    // LED à 100%.
    analogWrite(ledPin, 255);
    delay(timer); }
```

f) Modifier le code précédent de tel façon :

- la led de branche 10 clignote à 10 %
- la led de branche 11 clignote à 40 %.
- la led de branche 12 clignote à 80 %.

g) Donner le programme qui permet de réaliser un chenillard à 3 led

Projet 2 : Entrées/Sorties analogiques

La carte Arduino Uno dispose de 6 entrées analogiques notées A0, A1,..A5 mais d'un seul convertisseur analogique/numérique, la durée d'une conversion est de l'ordre de 100µs.Il a une résolution de 10 bits. La donnée numérique qu'il fournit après conversion est donc comprise entre 0 et 1024

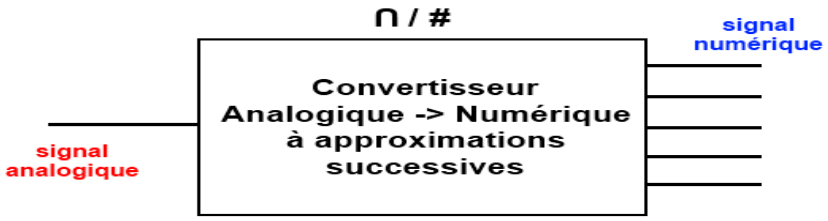


Figure 6 – Type du CAN de la carte Arduino UNO

I) Mesurer une température avec un capteur LM35 et une carte Arduino

Le LM35 fait partie des capteurs de température électroniques de précision en structure intégrée.

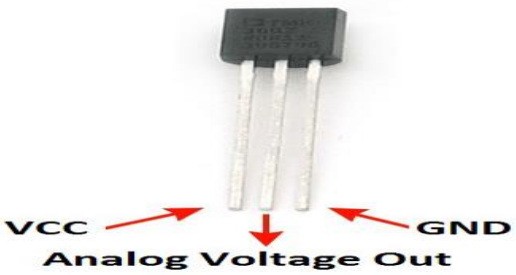


Figure 7 – Capteur de température LM35

D'après la fiche technique :

$$10mV \rightarrow C^o$$

Autrement un volt correspond à 100 degrés Celsius. La lecture analogique d'un signal de 0 à 5V étant codée de 0 à 1023, on a la formule :

$$Temp = Volt * (5/1023) * 100; \qquad Volt.entre.(0et1023)$$

Pour exploiter le capteur LM35, il suffit :

– D'alimenter les pattes VCC et GND
– De brancher la patte centrale à une entrée analogique d'Arduino (A0,...,A5).

La figure 1 montre le Branchement du Capteur LM35 avec Arduino UNO

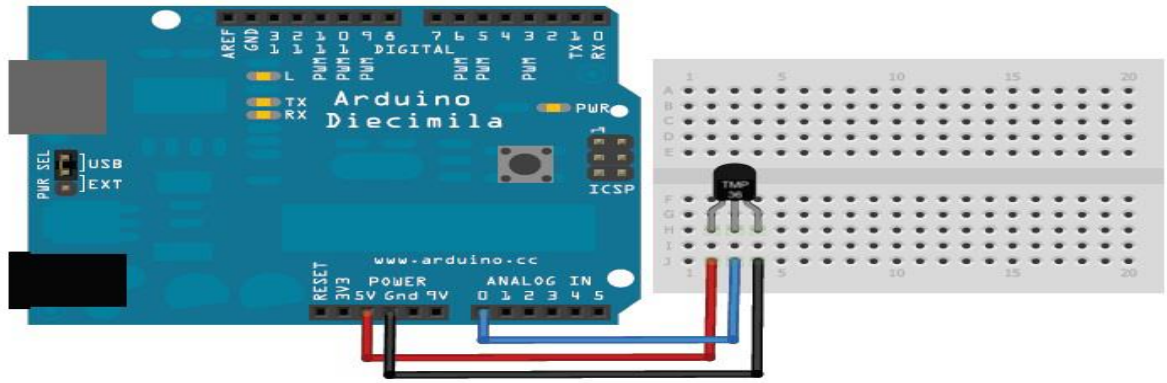


Figure 1 Branchement du Capteur LM35 avec Arduino UNO

a) Compiler le code suivant :

```
void setup()
{ Serial.begin(9600);
}

void loop()
{

 // Mesure la tension sur la broche A0 (sortie capteur)
  int valeur_brute = analogRead(A0);

  // Transforme la mesure (nombre entier) en température via un produit en croix
  float temperature_celcius = (valeur_brute - valeur_offset) * (5.0 / 1023.0 * 100.0);

  // Envoi la mesure au PC pour affichage et attends 250ms
  Serial.println(temperature_celcius);
  delay(250);
}
```

b) Ajouter une led au montage figure 1 et donner un programme qui permet d'allumer cet led si la température dépasse 29 degrés.

II) Mesurer la luminosité avec une photorésistance et une carte Arduino

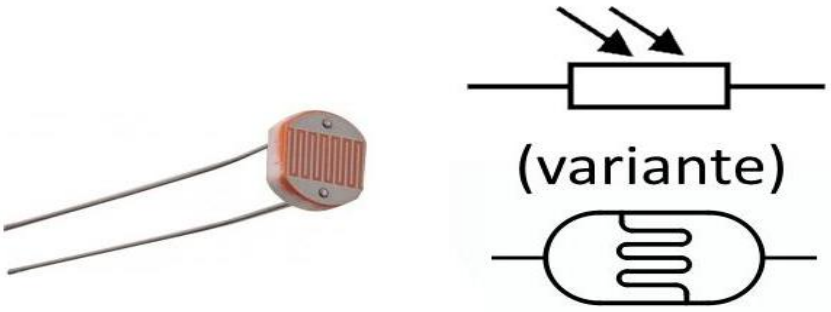


Figure 2 Photorésistance et symbole

Une photorésistance Figure 2 est un composant dont la résistivité dépend de la luminosité ambiante. Pour faire simple, c'est une résistance dont la valeur change en fonction de la lumière qu'elle reçoit.

Il existe différents types de photorésistances, chacune ayant des valeurs de résistance différentes en fonction de la luminosité ambiante. Le type le plus classique de photorésistances est de 1M ohms (obscurité) / 12K ohms (pleine lumière). C'est ce genre de photorésistance qui est employé plus bas dans ce tutoriel.

Qu'importe le diamètre de la photorésistance, sa valeur dans l'ombre ou en pleine lumière, quand une photorésistance est illuminée, sa résistance diminue. On peut donc utiliser une photorésistance pour mesurer la luminosité ambiante.

Sans faire une liste exhaustive, voici quelques exemples d'utilisations très classiques pour une photorésistance :

- Détection jour / nuit,
- Mesure de luminosité ambiante (pour ajuster un éclairage par exemple),
- Suiveur de lumière (pour panneaux solaires, robots, etc)

a) Réaliser le montage suivant :

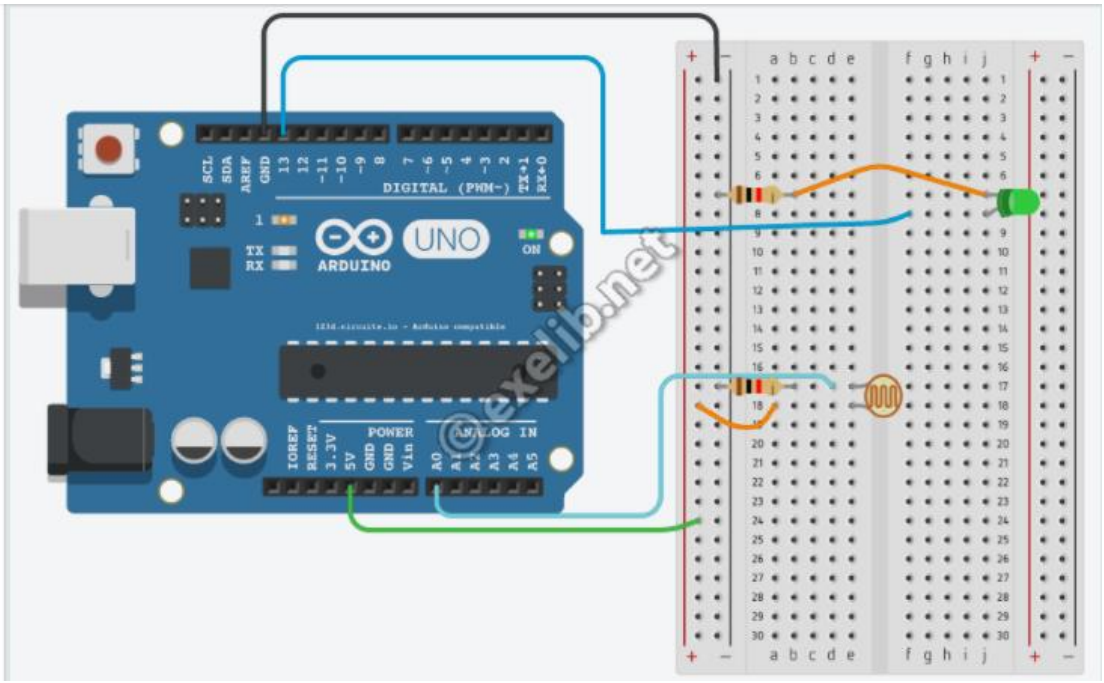


Figure 3 Montage Photorésistance

b) Compiler le code suivant :

```
// Initialisation des constantes :
const int analogInPin = A0;   // Numéro de la broche à laquelle est connectée la photorésistance
const int analogOutPin = 13;   // Numéro de la broche à laquelle est connectée la LED

int sensorValue = 0;          // Valeur lue sur la photorésistance
```

```
int outputValue = 0;          // Valeur envoyée à la LED

void setup() {
  // Initialise la communication avec l'ordinateur
  Serial.begin(9600);

  // Indique que la broche analogOutPin est une sortie :
  pinMode(analogOutPin, OUTPUT);
  // Indique que la broche analogInPin est une entrée :
  pinMode(analogInPin, INPUT);
}

void loop() {
  // lit la valeur de la photorésistance et
  // stocke le résultat dans sensorValue :
  sensorValue = analogRead(analogInPin);
  // change sensorValue vers une intervalle de 0 à 255
  // et stocke le résultat dans outputValue :
  outputValue = map(sensorValue, 0, 1023, 0, 255);
  // envoie de cette nouvelle valeur sur la LED
  analogWrite(analogOutPin, outputValue);

  // envoie tout ça vers l'ordinateur
  Serial.print("sensor = " );
  Serial.print(sensorValue);
  Serial.print("\t output = ");
  Serial.println(outputValue);
}
```

c) Ajouter une led au montage figure 3 et donner un programme qui permet d'allumer cet led si la luminosité mesurée par la photorésistance est trop faible (inférieur à 1%).

III) Variation de la luminosité d'une LED avec un potentiomètre et une carte Arduino

Un potentiomètre (Figure 4) est un bouton qui fournit une résistance variable. Les valeurs des potentiomètres sont envoyées dans l'Arduino sous un signal analogique.

Figure 4 potentiomètres

Le potentiomètre possède 3 broches :
– Une alimentation (généralement, nous utilisons le +5V délivré par l'Arduino)
– Une sortie analogique
– Une masse

En tournant l'axe du potentiomètre, nous modifions la résistance vers l'interface de sortie. L'entrée analogique de l'Arduino Uno est codé sur 10 bits. Quand nous envoyons la tension en sortie du potentiomètre vers l'entrée de l'Arduino, celle-ci va être convertie en un nombre numérique.

Pour une alimentation de 5V :

0V –> 0
5V –> 1023

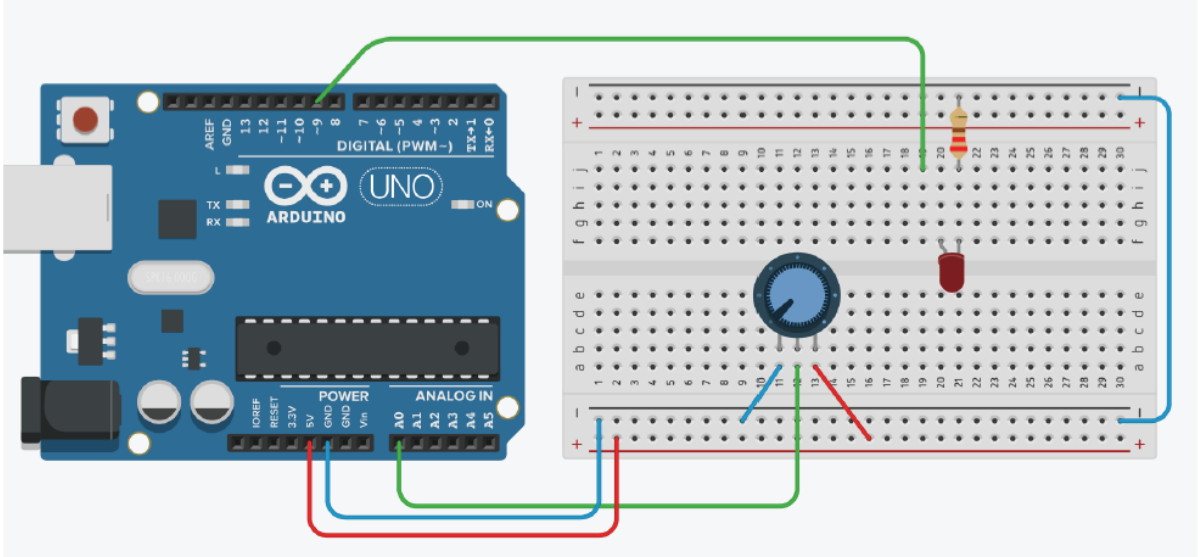


Figure 5 montage potentiomètre avec une led et une carte arduino

a) Réaliser le montage Figure 5
b) Utiliser le code de photorésistance pour écrire un code qui modifie la luminosité de la led avec le potentiomètre
c) Ajouter une led au montage figure 5 et modifier le code précédent de telle sorte que lorsque la luminosité d'une led atteint son maximum l'autre led atteint son minimum c'est-à-dire les luminosités des led varient inversement.