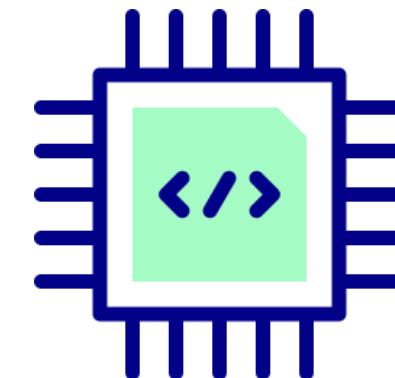
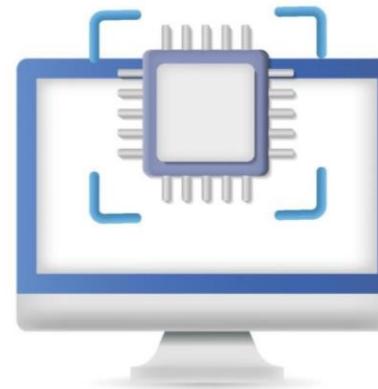
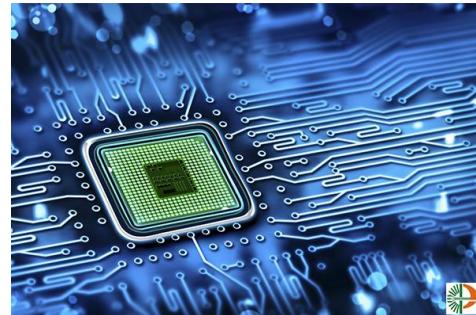




# Travaux Pratiques de Systèmes Embarqués

Filière: Sécurité IT et confiance numérique (S2)



# TP1: Initiation à l'Arduino

## Objectif du TP:

*Ce TP a pour objectif d'initiation à l'arduino. Il s'agit de manipuler les entrées et les sorties d'une carte arduino en et de se familiariser avec le langage de programmation embarqué.*

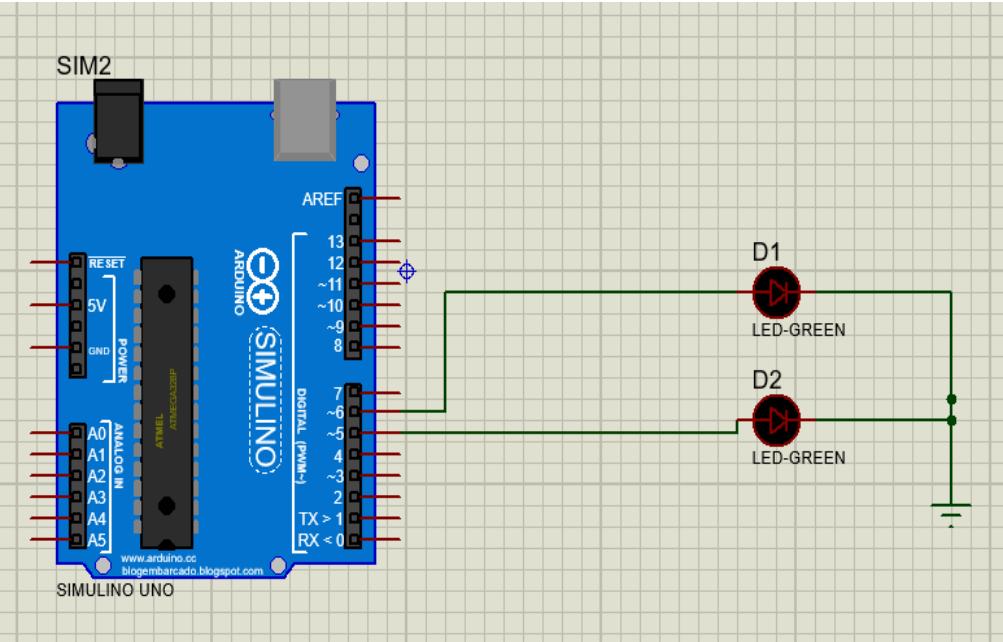
*Ce TP est composé des parties suivantes:*

- ❖ *Installation et utilisation du logiciel ISIS Proteuse 8*
- ❖ *Réalisation d'un chenillard de LEDs en manipulant les sorties numériques d'une carte Arduino ;*
- ❖ *Acquisition et traitement des informations issues d'un capteur analogique numérique;*
- ❖ *Acquisition des informations issues d'un capteur Ultrason*

## Activité 1:

Réaliser le montage suivant sous le logiciel ISIS proteus

Développer le code Arduino suivant, télécharger le dans la carte Arduino. Exécuter et vérifier le fonctionnement



```
#define led1 5 // led1 branchée sur la broche 2
#define led2 6 // led2 branchée sur la broche 3

void setup() // setup est déroulé une seule fois après la remise à zéro
{
    pinMode(led1, OUTPUT); // la broche led1 (2) est initialisée en sortie
    pinMode(led2, OUTPUT); // la broche led2 (3) est initialisée en sortie
}

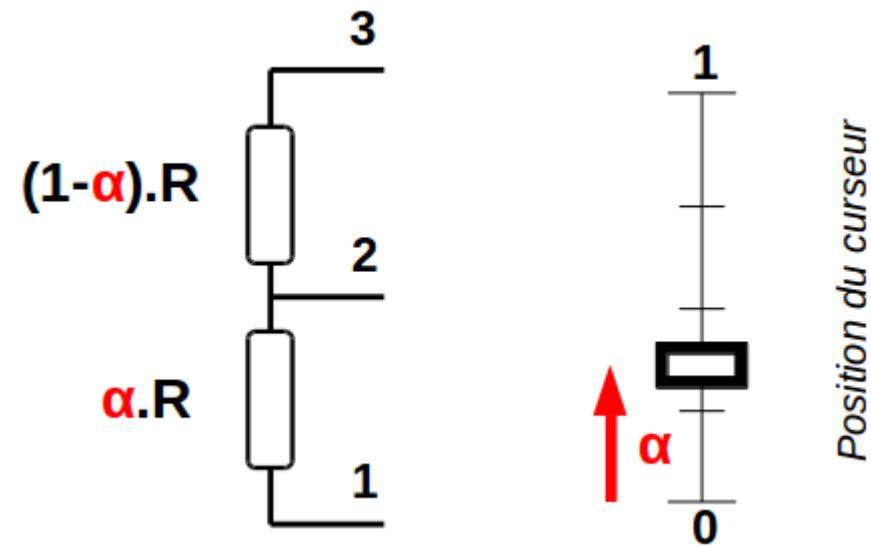
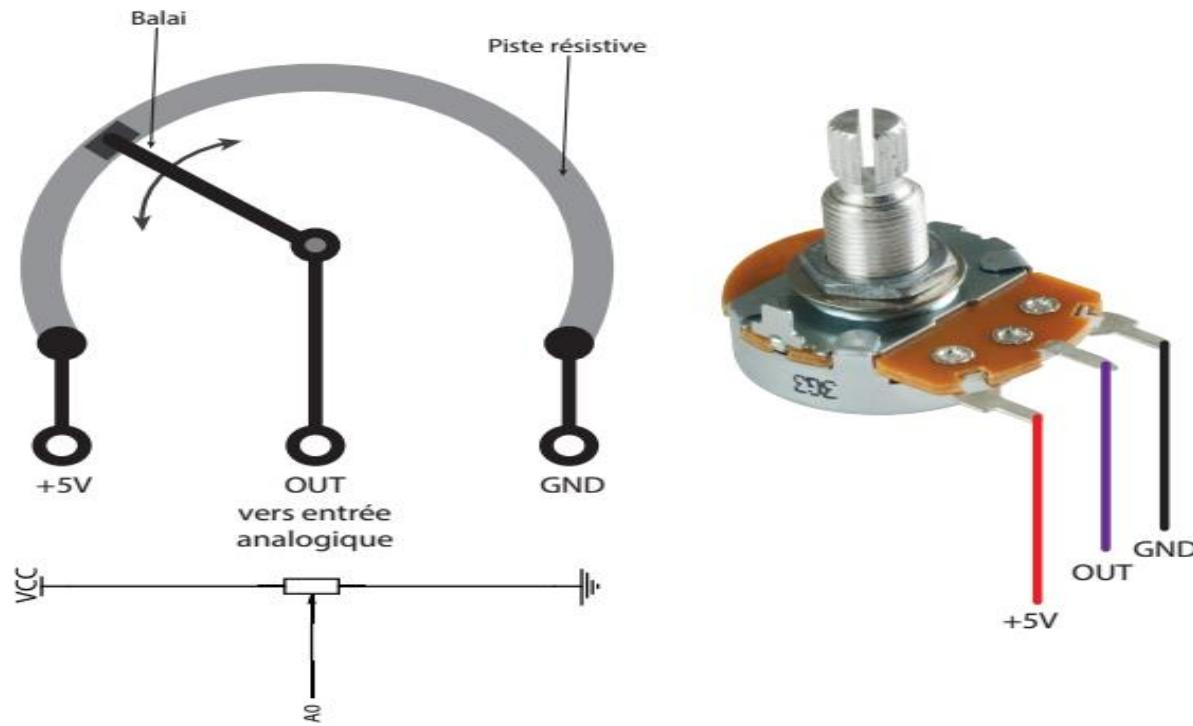
void loop() // loop est déroulé indéfiniment
{
    digitalWrite(led1, HIGH); // allume la LED1 on aurait aussi pu écrire 1 à la place de HIGH
    digitalWrite(led2, LOW); // éteint la LED2 on aurait aussi pu écrire 0 à la place de LOW
    delay(500); // attente de 1/2 seconde
    digitalWrite(led1, LOW); // éteint la LED1
    digitalWrite(led2, HIGH); // allume la LED2
    delay(500); // attente de 1/2 seconde
    digitalWrite(led1, LOW); // éteint la LED1
    digitalWrite(led2, LOW); // éteint la LED2
    delay(500); // attente de 1/2 seconde
    digitalWrite(led1, LOW); // éteint la LED1
    digitalWrite(led2, LOW); // éteint la LED2
    delay(500); // attente de 1/2 seconde
}
```

## Challenge 1:

Modifier le programme et le montage de manière à réaliser un chenillard de 5 LEDS

## ❑ Entrée analogique

**Potentiomètre** : Il s'agit d'une résistance variable, il délivre sur sa borne de sortie (curseur) une tension entre 0 et VCC, il joue donc le rôle d'un capteur analogique.



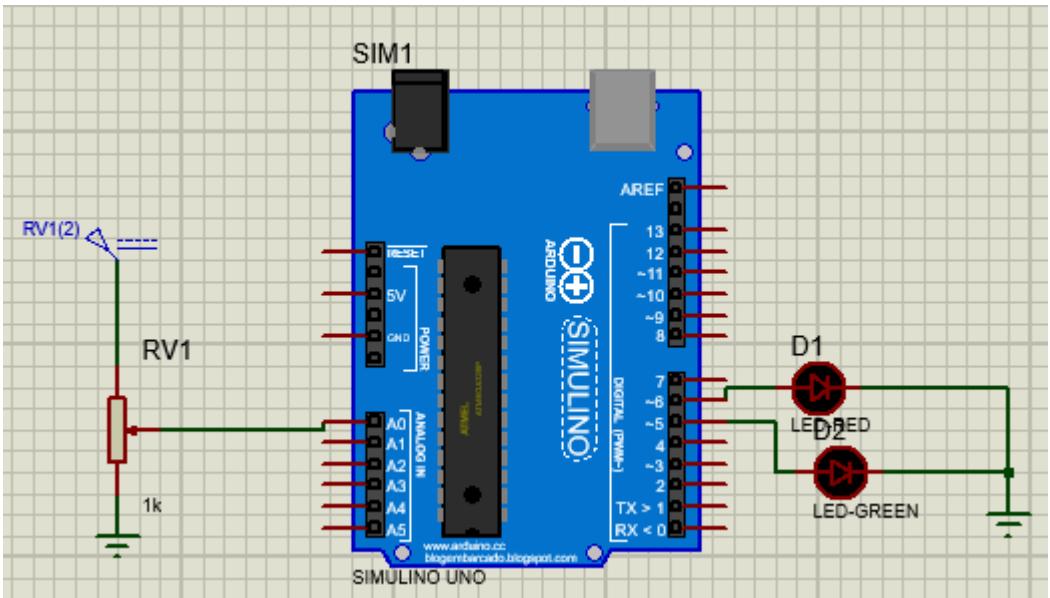
La carte arduino contient une Convertisseur analogique numérique de 10bits, en conséquence lorsqu'une valeur analogique (0 à Vcc) apparait en entrée, le programme affiche une valeur image entre 0 et 2047, pour afficher la valeur réelle il suffit d'utiliser la règle de trois (3).

## Entrée analogique

## Activité 2:

*Réaliser le montage suivant sous le logiciel ISIS proteuse*

*Développer le code Arduino suivant, télécharger le dans la carte Arduino. Exécuter et vérifier le fonctionnement*



```
float Pot=A0;
float val=0.0;
int L1=5;
int L2=6;
void setup() {
  pinMode(L1, OUTPUT);
  pinMode(L2, OUTPUT);
  pinMode(Pot, INPUT);
}
void loop() {
  val = analogRead(Pot); // Lire la valeur du potentiomètre
  if (val < 100) {
    digitalWrite(L1, HIGH);
    digitalWrite(L2, LOW);
  } else {
    digitalWrite(L1, LOW); // Correction de la logique pour L1
    digitalWrite(L2, HIGH); // Correction de la logique pour L2
  }
  delay(500); // Attendre 500 ms
}
```

## Challenge 1:

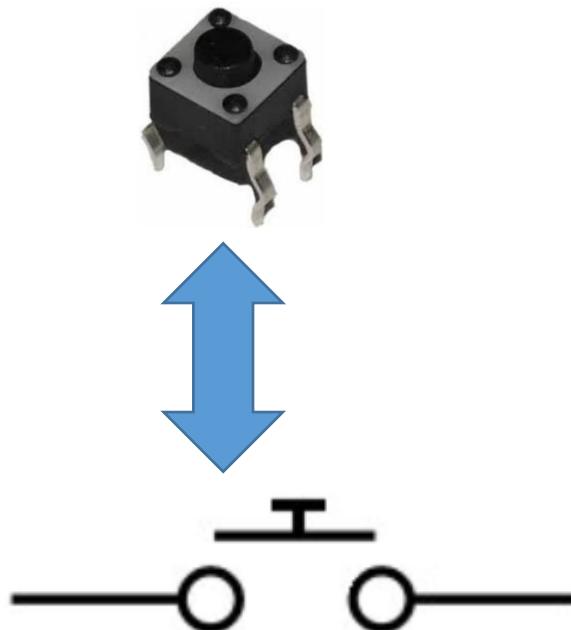
*Modifier le programme et le montage pour utiliser trois LEDs. Proposer un scénario pour clignoter ces LEDs en fonction de la valeur du capteur.*

## Activité 3: LED et bouton poussoir

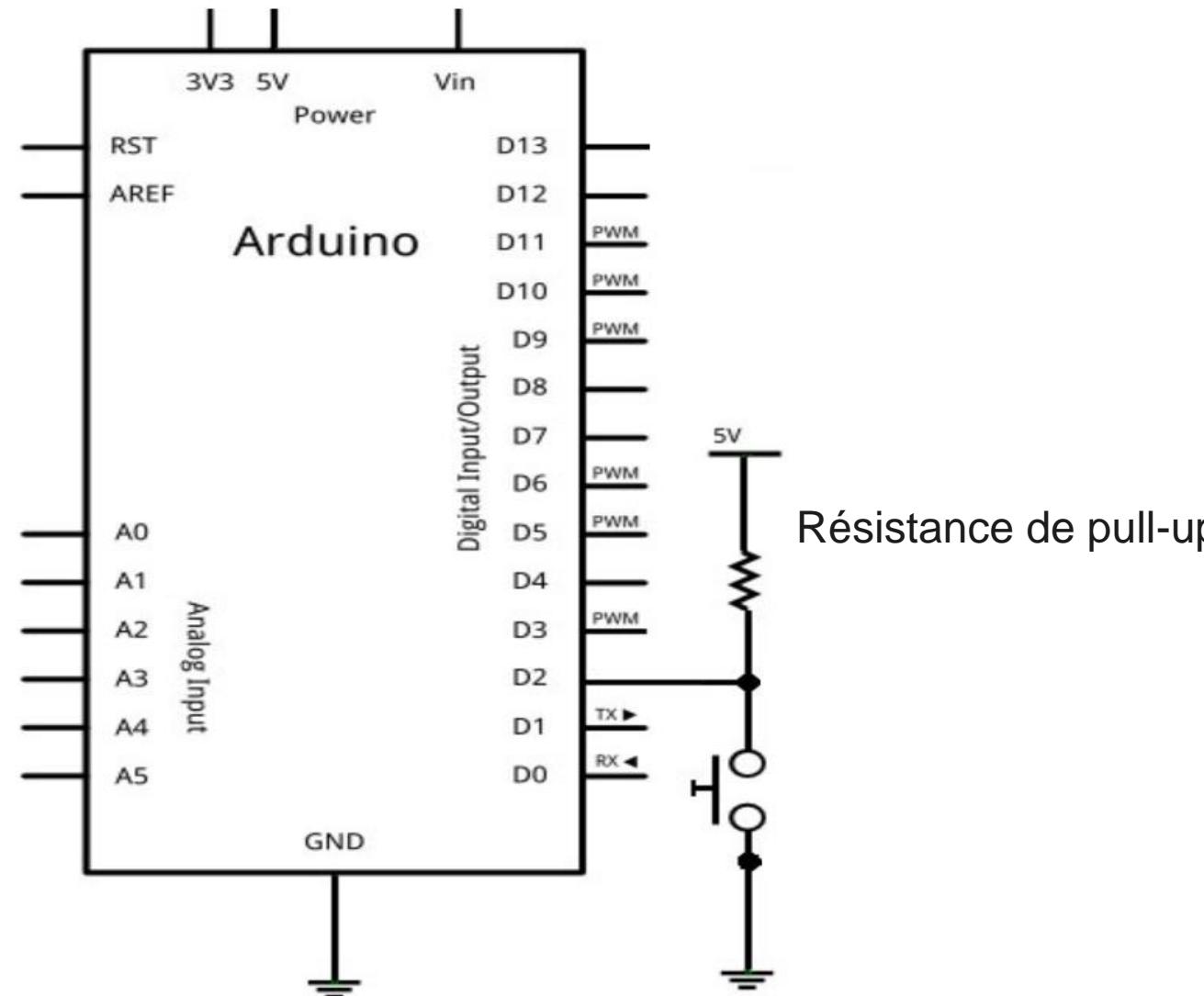
*Développez un programme qui permet de contrôler l'état d'une LED à l'aide d'un bouton poussoir*

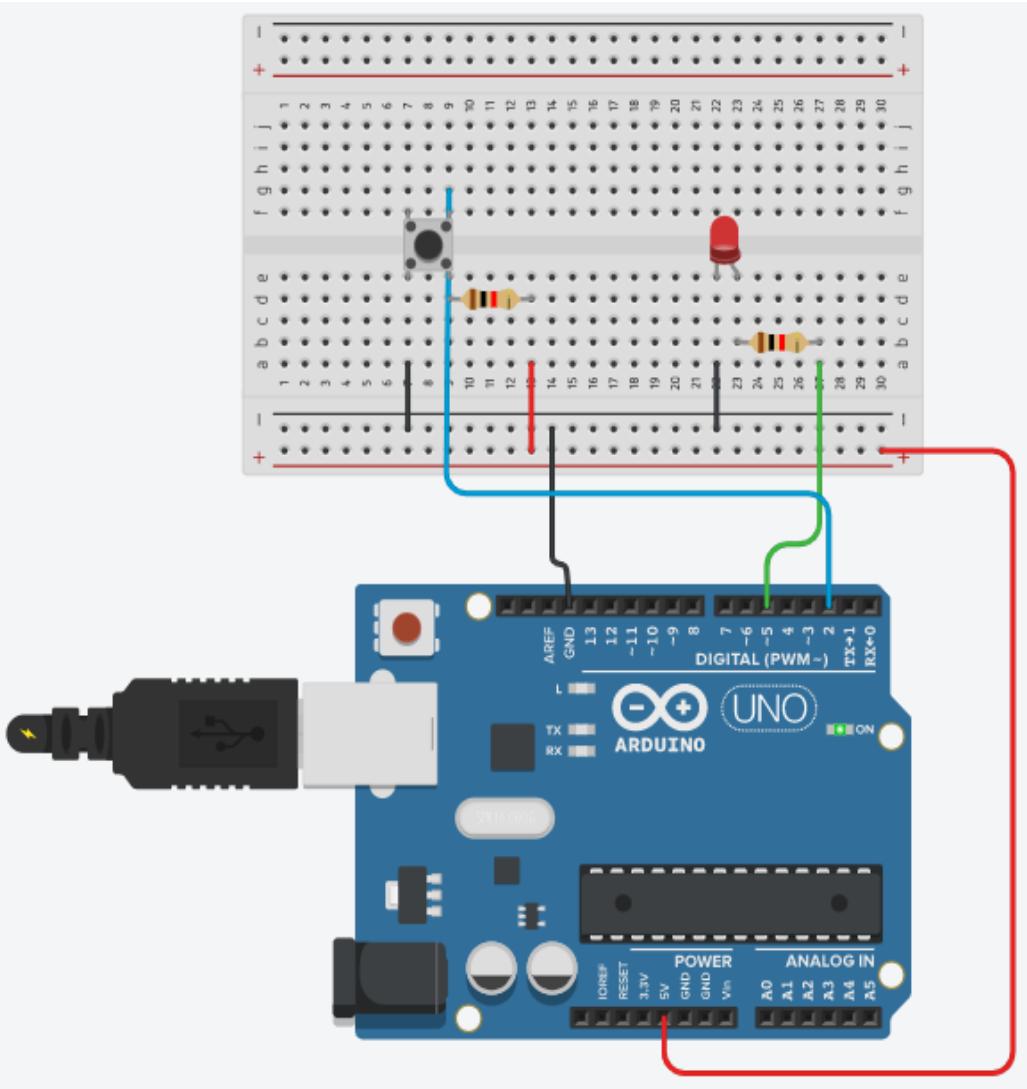
- *allumer la led si le bouton est appuyé*
- *Eteindre la led si le bouton est relâché*

Fonctionnement: bouton poussoir (BP)



Le BP est un interrupteur fermé lorsque il est appuyé et ouvert lorsque il est relâché.





```

const int bouttonPin = 2; // crée un identifiant pour la broche utilisée avec le BP
const int ledPin = 5; // crée un identifiant pour la broche utilisée avec la LED
int bouttonEtat = 0; // variable pour mémoriser l'état du bouton
void setup() {
    // configure la broche numérique en SORTIE
    pinMode(ledPin, OUTPUT);
    // configure la broche numérique en SORTIE
    pinMode(bouttonPin, INPUT);
}
void loop(){
    // lit la valeur de l'état du bouton et la mémorise dans la variable
    bouttonEtat = digitalRead(bouttonPin);

    // Teste si le bouton est appuyé
    // c'est à dire si la variable bouttonEtat est à 1

    if (bouttonEtat == LOW) {
        // allume la LED
        digitalWrite(ledPin, HIGH);
    }
    else { // sinon
        // éteint la LED
        digitalWrite(ledPin, LOW);
    }
}

```

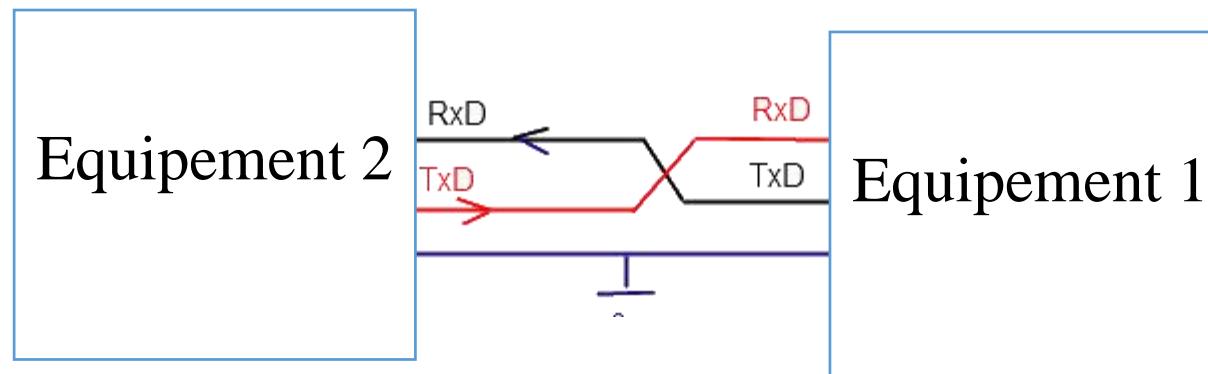
**Challenge 3: Développer un programme qui permet d'utiliser deux boutons pour contrôler les états de deux LEDs**

## TP2: Manipulation de la communication série

### Objectif du TP:

Le but de ce TP est de se familiariser avec les protocoles de communication série à savoir UART (USART) et I2C. Ces outils de communication reposent sur une programmation embarquée appropriée et ils servent à interconnecter les objets.

### Principe de la communication série

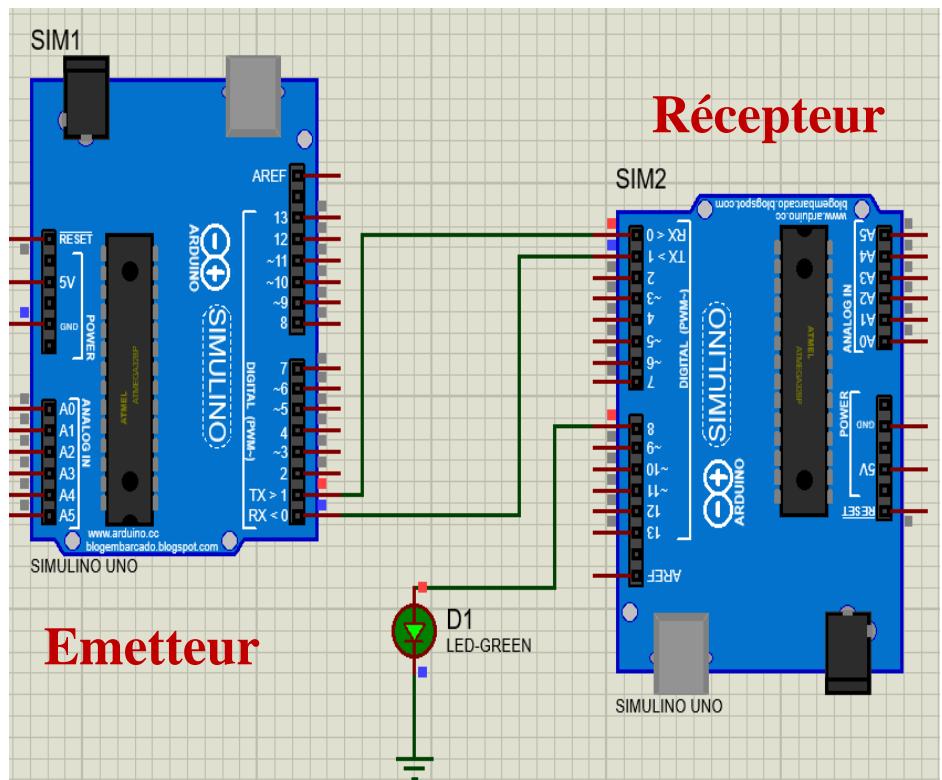


## Activité 1: Communication des données entre deux cartes Arduino

Il s'agit de réaliser la communication entre deux objets, un émetteur et un récepteur, via une liaison série (filaire). Dans cet activité, l'émetteur va envoyer une information de commande d'allumer ou éteindre d'une LED. Le récepteur va recevoir l'information et va actionner la sortie (LED).

**Réaliser le montage suivant et exécuter les codes correspondants**

**Observer les résultats de la simulation**



### Emetteur

```
void setup() {  
    Serial.begin(9600); // Initialisation de la communication série à 9600 bauds  
}  
void loop() {  
    Serial.print('0');  
    delay(500); // Attendre 500ms avant la prochaine lecture  
    Serial.print('1');  
    delay(500);  
}
```

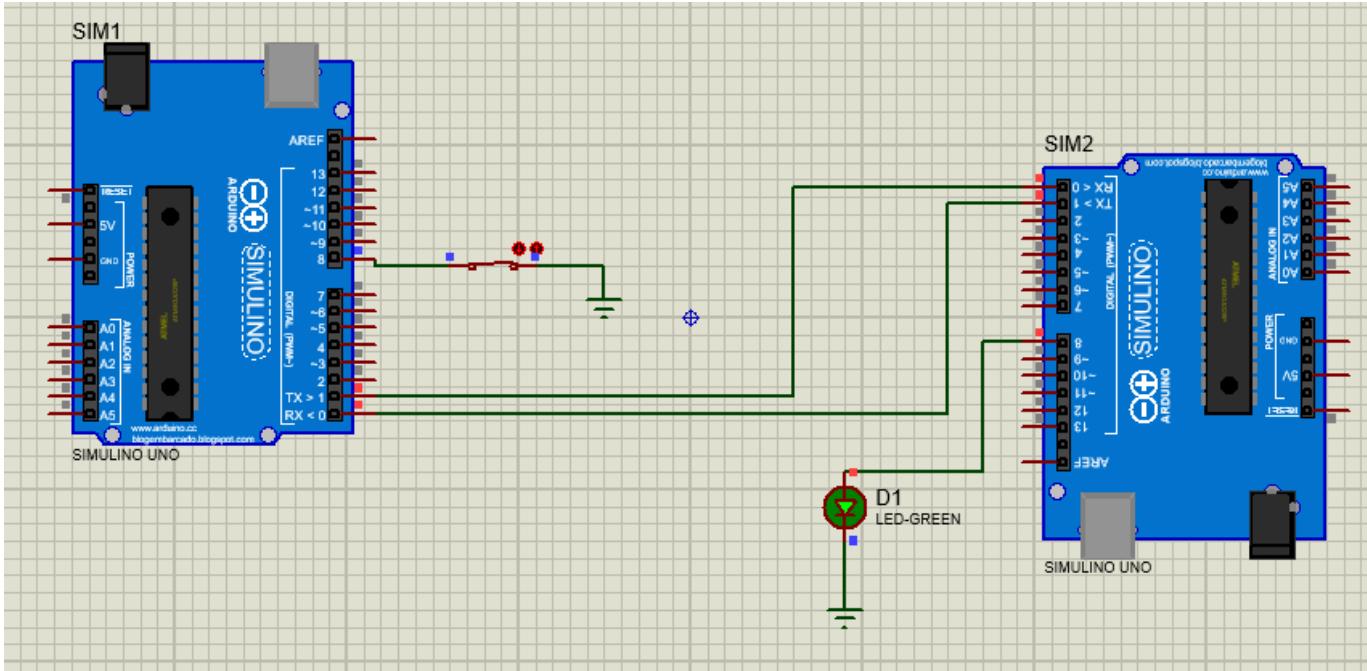
### Récepteur

```
#define LED_PIN 8 // Broche où est connectée la LED  
void setup() {  
    pinMode(LED_PIN, OUTPUT);  
    Serial.begin(9600);  
}  
void loop() {  
    if (Serial.available()) {  
        char commande = Serial.read(); // Lire la valeur reçue  
        if (commande == '1') {  
            digitalWrite(LED_PIN, HIGH); // Allumer la LED  
            Serial.println("LED allumée");  
        }  
        else if (commande == '0') {  
            digitalWrite(LED_PIN, LOW); // Éteindre la LED  
            Serial.println("LED éteinte");  
        }  
    }  
}
```

## TP2: Manipulation de la communication série

### Exercice 1:

1. Modifier le programme et le montage pour commander la LED par un interrupteur au niveau de l'émetteur.
2. Modifier le programme et le montage pour commander trois LEDs en utilisant trois interrupteurs.



## TP2: Manipulation de la communication série

### Exercice 2:

1. Exécuter le code suivant et réaliser la communication entre une carte Arduino et votre Ordinateur via un câble USB.
2. Modifier le code de manière à envoyer un commande ou accusé de votre PV vers la Carte

```
void setup() {  
    // put your setup code here, to run once:  
    Serial.begin(9600);  
}  
  
void loop() {  
  
    Serial.println("Bonjour SITCN_S2");  
    delay(1000);  
}
```

