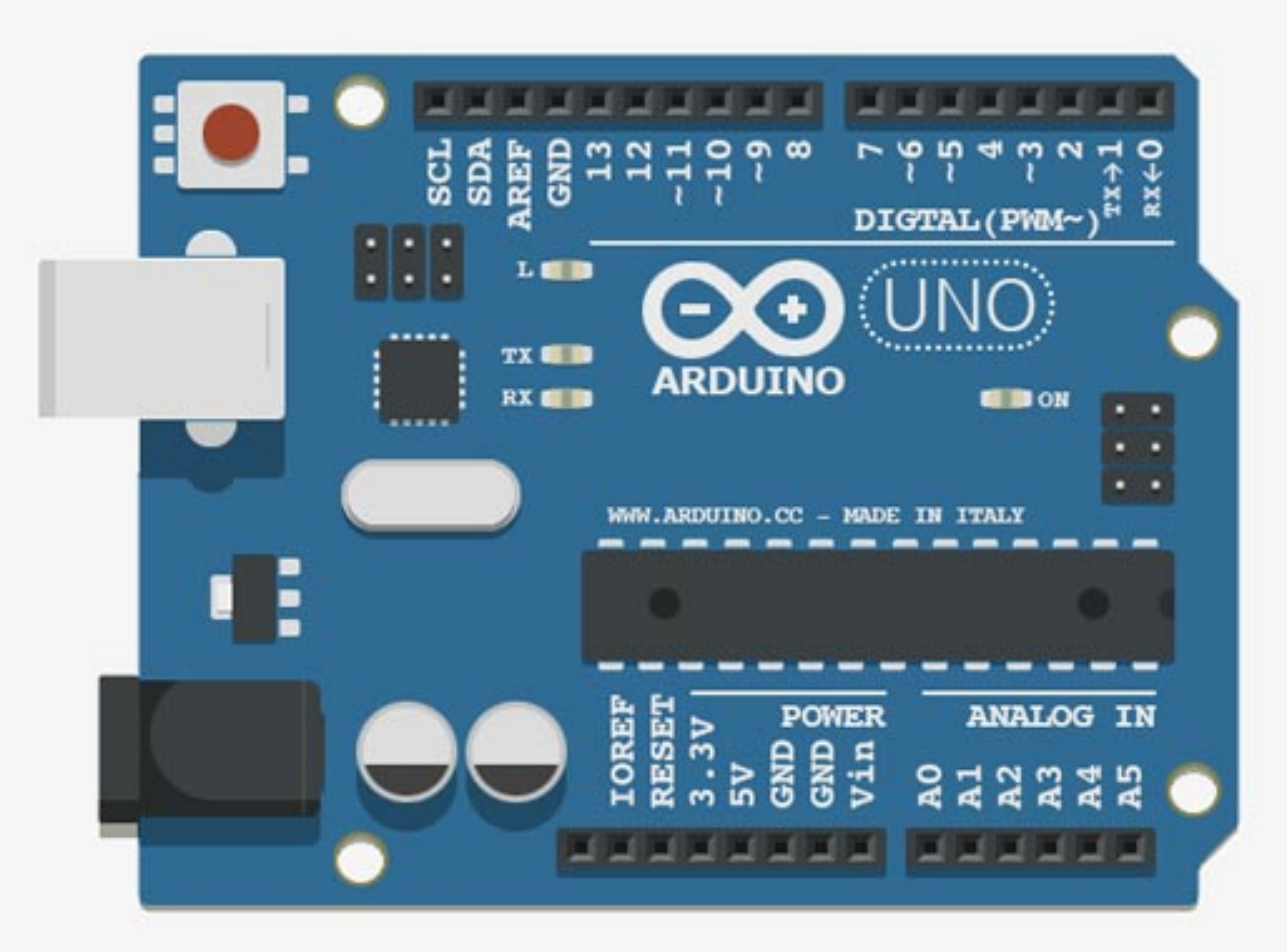
Architecture des ordinateurs : Arduino

Nicolas Garric

Contenu de l'UE

- 2 cours de 2h (N. Garric)
- 11 Tps de 2h
  - 2 TPs : LEDs et Boutons (B. Caillier /L. Brillon)
  - 3 TPs : maquette de feux tricolores (B. Caillier /L. Brillon)
  - 3 TPs : bandeaux de LEDs (B. Caillier / L. Brillon)
  - 3 TPs : matrices de LEDs (B. Caillier /L. Brillon)
- Une évaluation de TP en fin de semestre

Arduino ?


SCL
SDA
AREF
GND
13
12
~11
~10
~9
8
7
~6
~5
4
~3
2
TX→1
RX←0
DIGITAL(PWM~)
L
TX
RX
ARDUINO
UNO
ON
WWW.ARDUINO.CC – MADE IN ITALY
IOREF
RESET
3.3V
5V
GND
GND
Vin
POWER
A0
A1
A2
A3
A4
A5
ANALOG IN

Micro contrôleur

- Le Arduino est basé sur un micro-contrôleur de type Atmega
- Arduino UNO ==> ATmega328
- Il permet :
  - de faire de l'**informatique embarquée,**
  - De réaliser des **objets connectés**

Pour faire quoi ?

- Un robot mobile qui s’adapte à son environnement

Pour faire quoi ?

- Une station météorologique consultable sur le Web
- Une alarme pour votre maison : détecteurs + sirène
- Une commande chauffage : capteurs températures + convecteurs

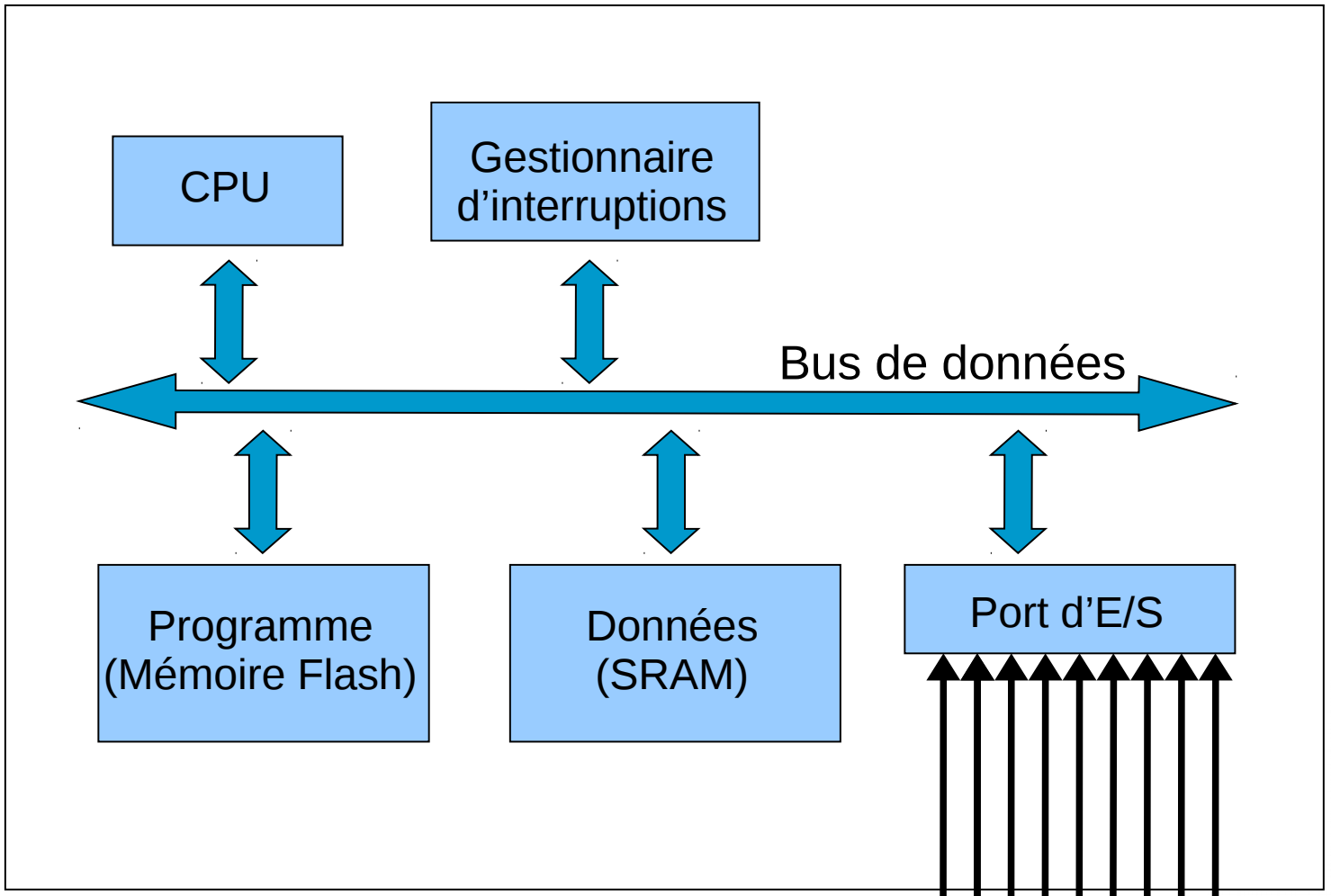
Pour faire quoi ?

- Arduino perçoit l'environnement extérieur à l'aide de capteurs…
- … traite ces données à l'aide d'un programme…
- … puis envoie des commandes à des actionneurs extérieurs.

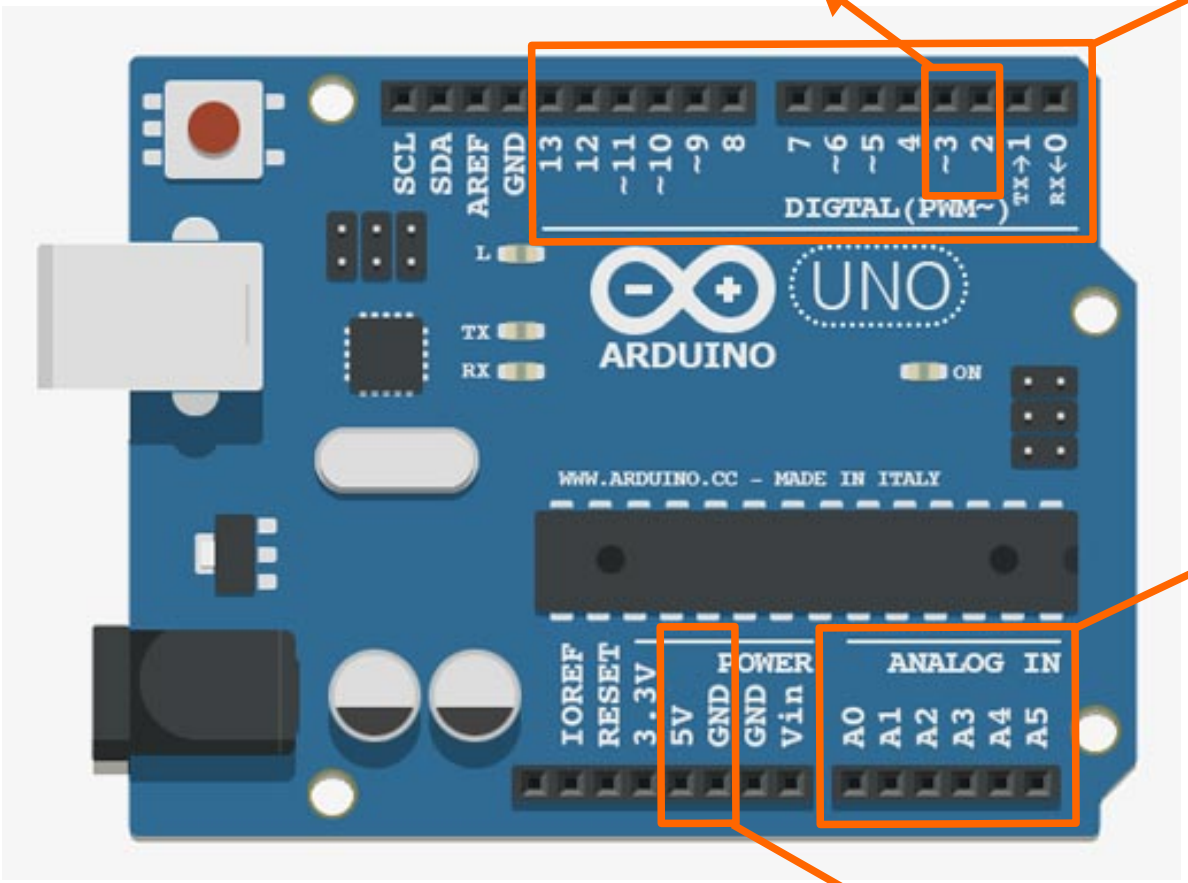
Plusieurs modèles

- Arduino UNO (Genuino) utilisé en TP
  - Mémoire Flash=32Ko, RAM=2ko, E/S numériques=16, E/S analogiques=6 , Fréquence =16MHz
- Arduino MEGA utilisé en TP
  - Mémoire Flash=128Ko, RAM=8ko, E/S numériques=54, E/S analogiques=16, Fréquence =16MHz
- Teensy utilisé en TP (projets plus complexes)
  - Processeur ARM
  - Mémoire Flash=256Ko, RAM=64ko, Fréquence =72MHz
- Arduino Due, Yun, ...

Arduino UNO


CPU
Gestionnaire d'interruptions
Bus de données
Programme (Mémoire Flash)
Données (SRAM)
Port d'E/S

Les entrées/sorties



Les E/S numériques

- 14 broches numériques (0 ou 5V) sélectionnables en entrée ou en sortie
- Parmi elles :
  - 2 entrées (2 et 3) peuvent être utilisées pour les interruptions
  - 5 sorties ~ (3, 5, 6, 10 et 11) peuvent être utilisées en sorties analogiques (pour obtenir 1V on envoie 5V pendant t secondes puis 0V pendant 4t secondes donc en moyenne, si t est très petit, on envoie 1V)

E/S numériques, exemples

- En sortie :
  - `pinMode(13,OUTPUT) ;`
  - `digitalWrite(13,HIGH) ;`
  - `digitalWrite(13,LOW) ;`
- En entrée:
  - `pinMode(12,INPUT) ;`
  - `x=digitalRead(12) ;`

Sortie analogique, exemple

- En sortie :
  - `pinMode(11,OUTPUT) ;`
  - `analogWrite(11,x) ;`
- La tension obtenue en sortie vaut (x/255)*5V

Avec x compris entre 0 et 255.

Les Entrées analogiques

- 6 broches analogiques utilisables en entrée uniquement
- Elles permettent de lire une tension comprise entre 0 et 5V en renvoyant une valeur sur 10 bits.
- Il y a donc 1024 valeurs possibles

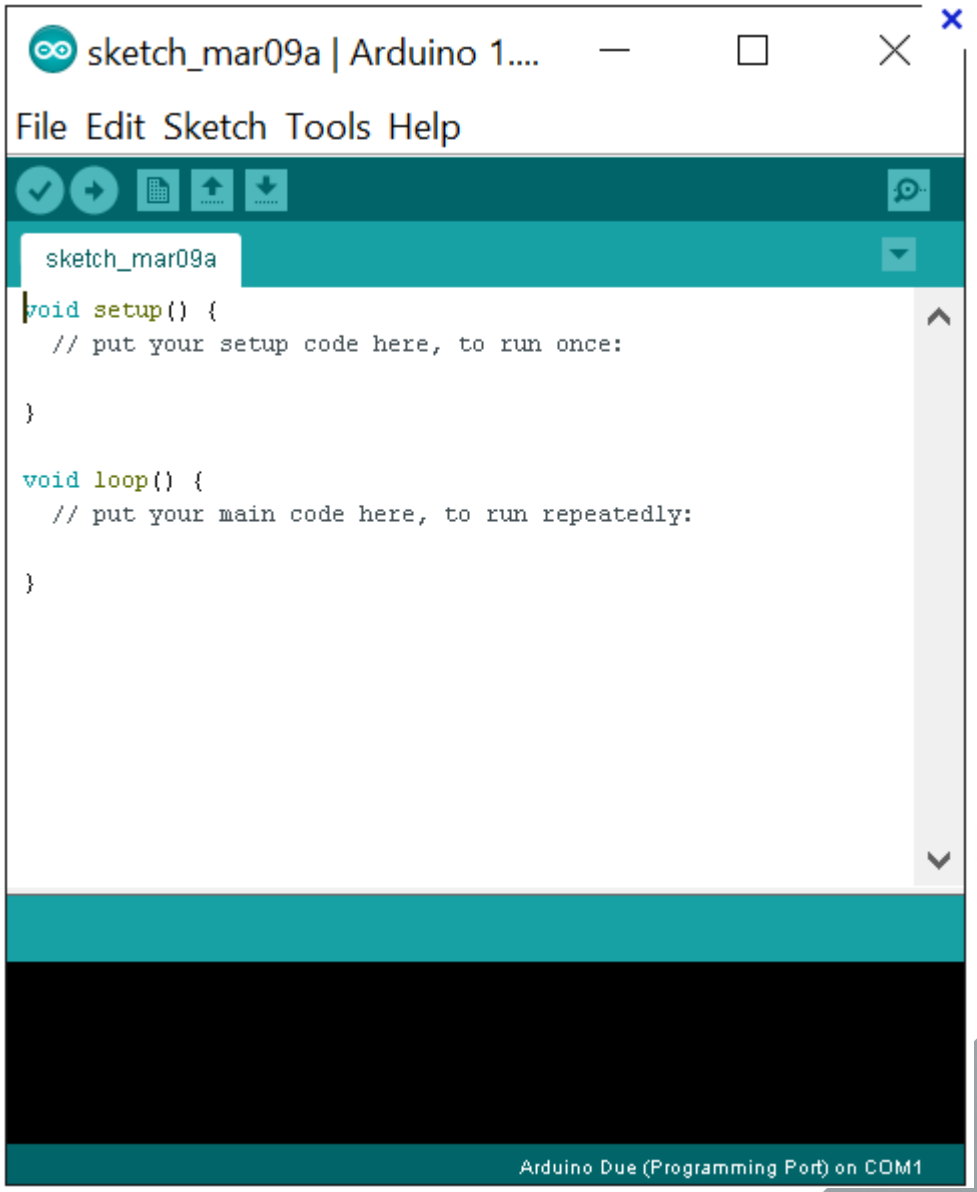
Entrée analogique, exemple

- Pour lire l'entrée analogique A0:
  - `x=analogRead(0) ;`

- **Attention** à ne pas envoyer plus de 5V sur les entrées analogiques

Programmer Arduino

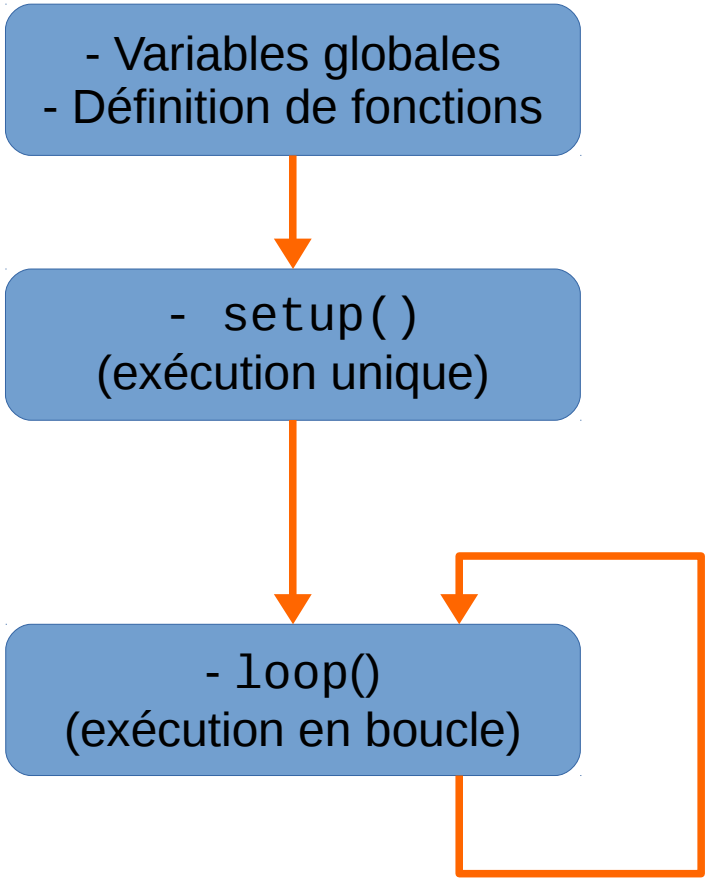
- On installe l'IDE Arduino
- Langage=C/C++



Programmer Arduino

- On compile dans l'IDE Arduino
- On téléverse ensuite vers le microprocesseur via le câble USB
- Tous les modèles Arduino sont disponibles (à choisir avant de compiler)

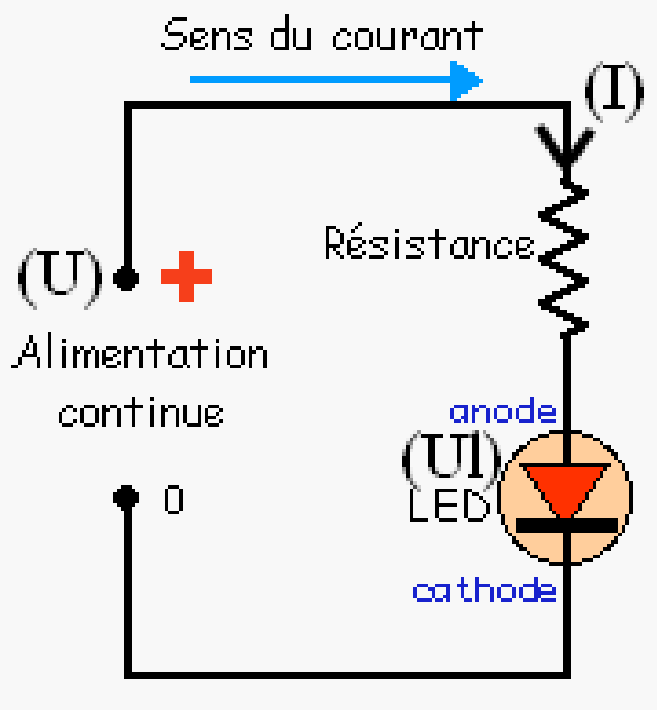
Cycle de vie d'un Sketch



Ex1 : Allumer une LED

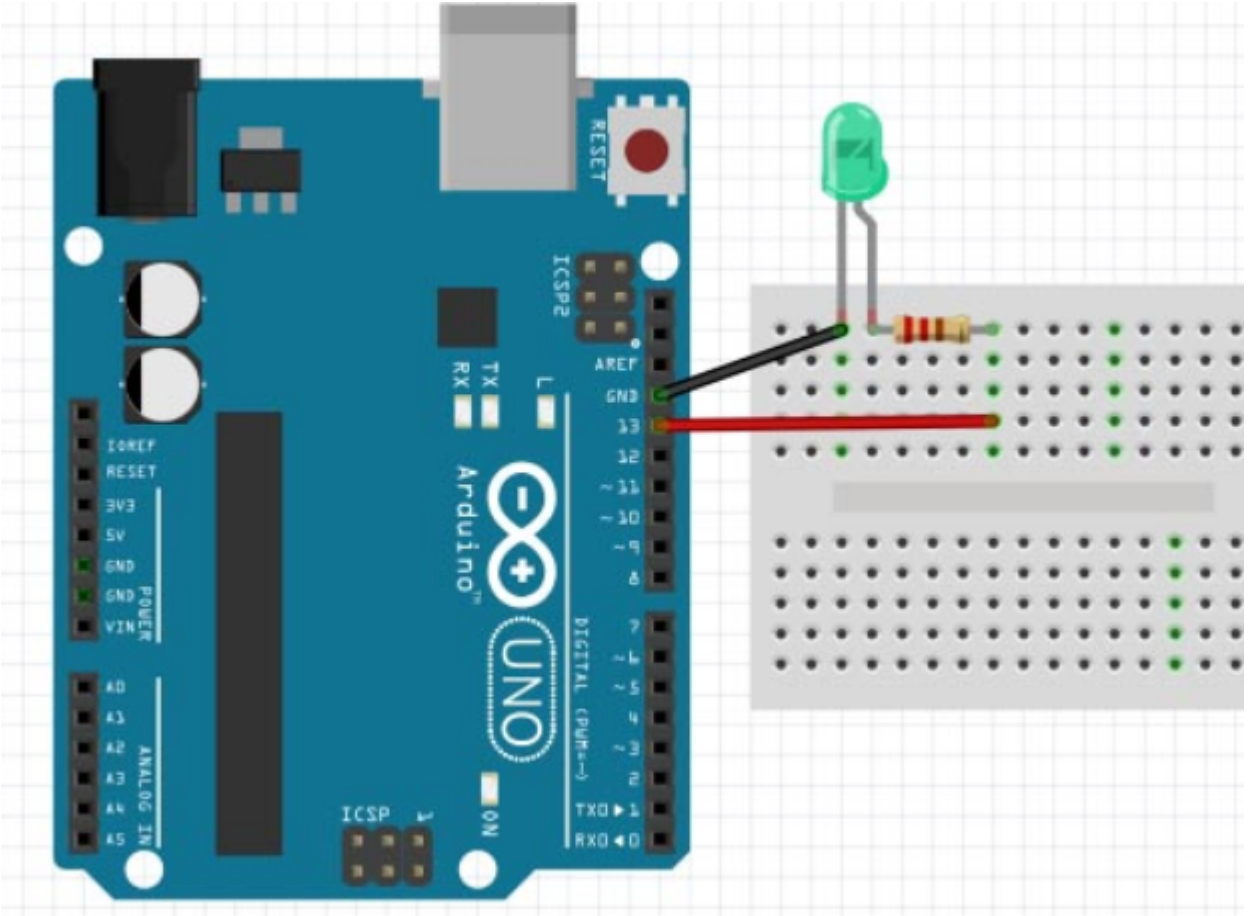
```
// variables globales
int rouge=2;

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

void loop() {
  digitalWrite(rouge,HIGH);
}
```

Ex1 : Schéma électrique



- La LED fonctionne bien si :
  - La tension aux bornes =2V
  - Le courant= 10mA
- La tension=5V en sortie de l'Arduino
- U=3V aux bornes de la résistance
- Donc **R > U/I = 300 Ω**

Ex1 : réalisation en TP



Ex2 : Faire clignoter une LED

```
// variables globales
int rouge=2;
void setup(){
  pinMode(rouge,OUTPUT);
}
void loop() {
  digitalWrite(rouge,HIGH);
  delay(1000) ;
  digitalWrite(rouge,LOW);
  delay(1000) ;
}
```

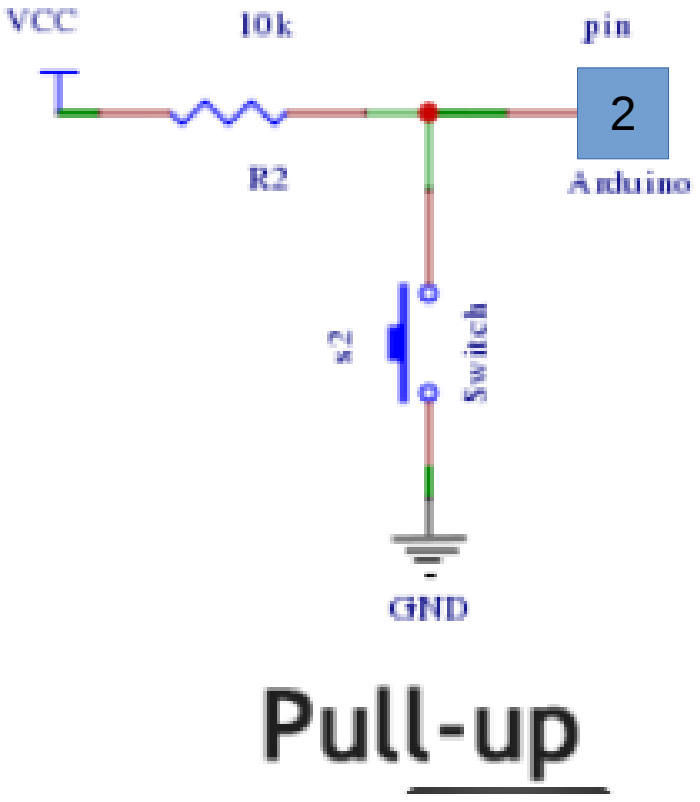
Ex3 : LED et bouton poussoir

- On ajoute un Bouton Poussoir (BP1)...
- … qui allume la LED quand on le presse
- … qui éteint la LED quand on le relâche

Ex3 : programme

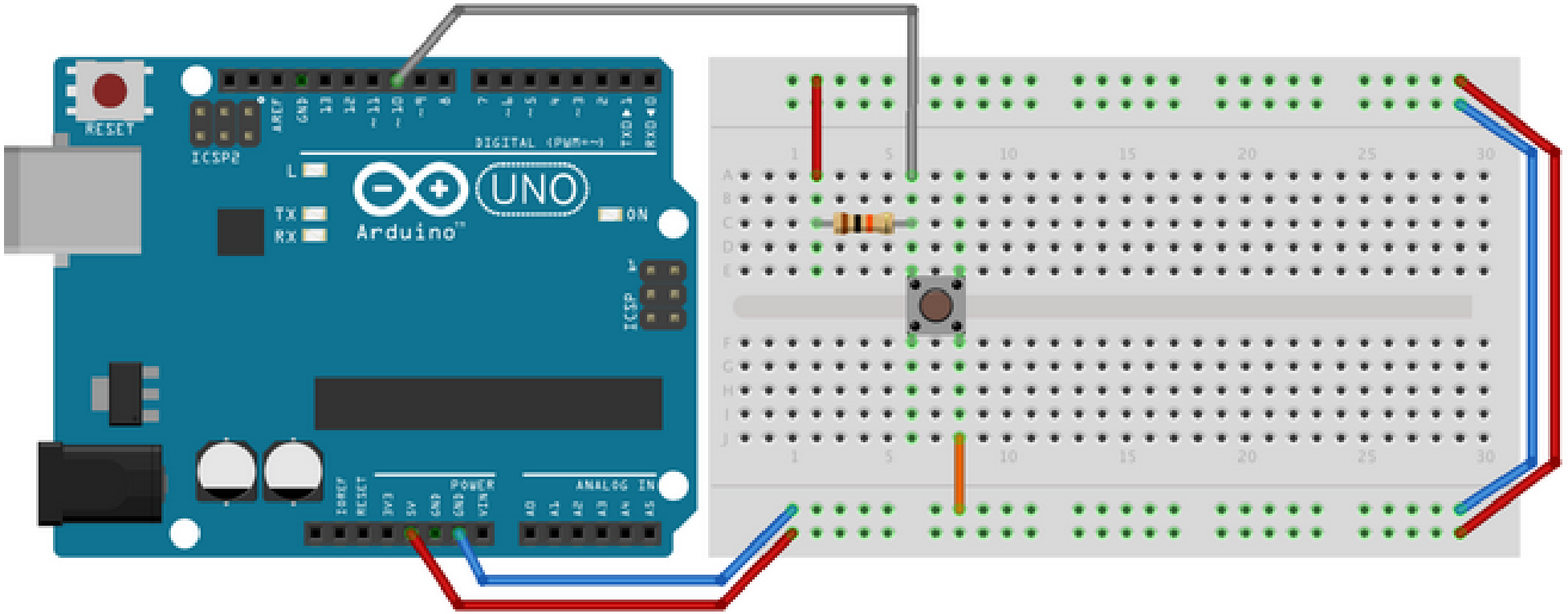
```
// variables globales
int rouge=9;
int bp1=2;
int val=LOW ;
void setup(){
  pinMode(rouge,OUTPUT);
  pinMode(bp1,INPUT);
}
void loop() {
  val=digitalRead(bp1) ;
    if (val==HIGH)
      digitalWrite(rouge,HIGH);
    else
      digitalWrite(rouge,LOW);
}
```

Ex3 : Schéma électrique



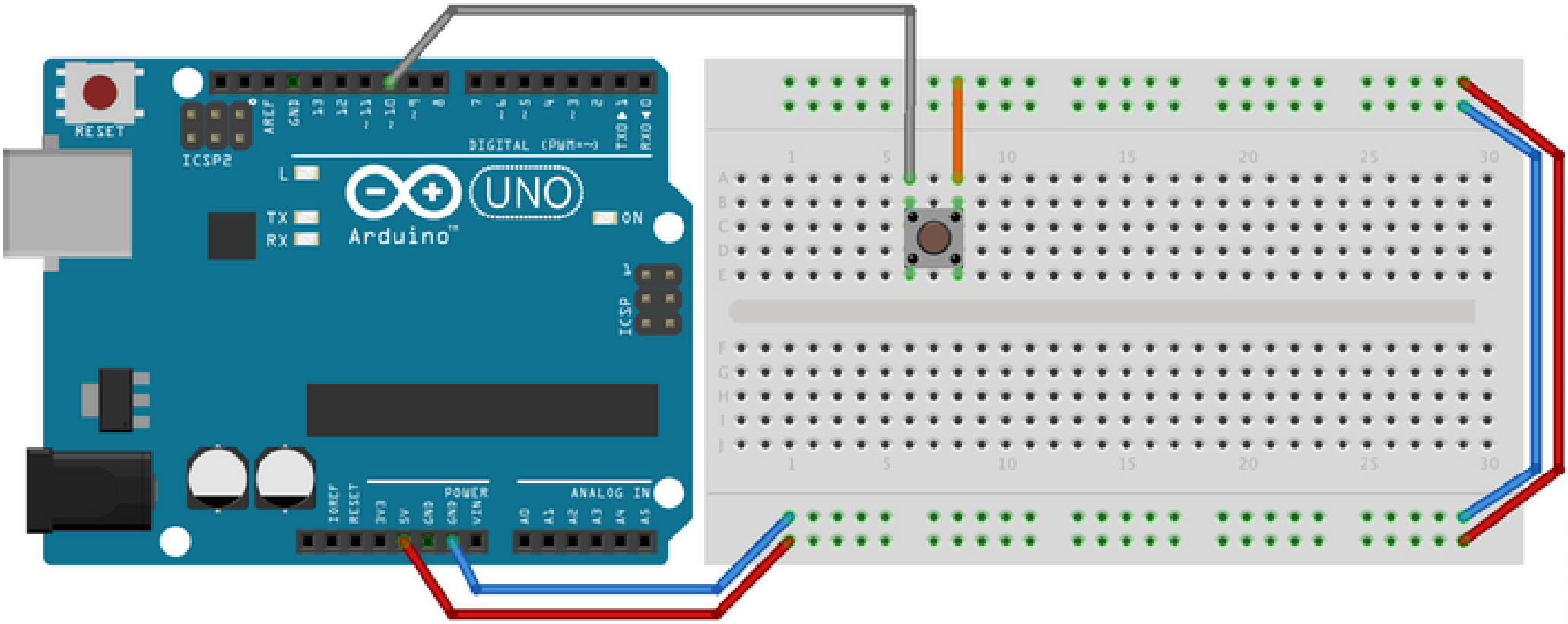
- Si le bouton est appuyé :
  - l'entrée 2 de l'arduino est à 0V
- Si le bouton est relâché:
  - l'entrée 2 est reliée à VCC=5V par la résistance.
  - Comme le courant entrant est très faible, l'entrée 2 est quasiment à 5V
  - On choisit **R=10 k Ω**

Ex3 : réalisation en TP



Ex3 : sans pullup

- On peut utiliser la pullup intégrée à l'arduino :

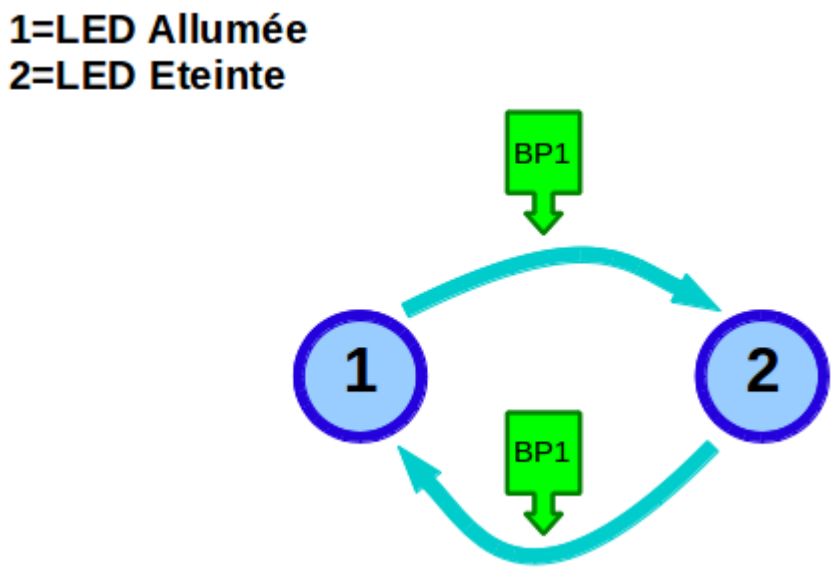


Ex3 : programme

```
// variables globales
int rouge=9;
int bp1=2;
int val=LOW ;
void setup(){
  pinMode(rouge,OUTPUT);
  pinMode(bp1,INPUT_PULLUP);
}
void loop() {
  val=digitalRead(bp1) ;
    if (val==HIGH)
      digitalWrite(rouge,HIGH);
    else
      digitalWrite(rouge,LOW);
}
```

Ex4 : LED et bouton poussoir

- On ajoute un Bouton Poussoir (BP1)...
- … qui change l'état de la LED quand on le presse



Ex4 : programme

```
int rouge=9, bp1=8, etat=0 ;
void setup(){
  pinMode(rouge,OUTPUT);
  pinMode(bp1,INPUT_PULLUP);
}
void loop() {
  int val=digitalRead(bp1);
  //on regarde si on a appuyé sur le bouton
  if (val==HIGH){
    etat=1-etat;
  }
  //on allume en fonction de l'état
  if (etat==1)
    digitalWrite(rouge,HIGH);
  else
    digitalWrite(rouge,LOW);
}
```

Quand on appuie sur le bouton, on modifie **etat** plusieurs fois

Solution = les interruptions

- Au lieu de regarder en permanence l'état du bouton…
- ...on demande au processeur d'exécuter une fonction lorsque le bouton est pressé.
- On associe alors une interruption à la broche du bouton poussoir

Ex4 : avec une interruption

```
int rouge=9;
int bp1=2;
volatile int etat=0; // ajouter volatile aux variables modifiées par interruption

void bouton_presse(){
 etat=1-etat;
}
void setup(){
  pinMode(rouge,OUTPUT);
  pinMode(bp1,INPUT);
  attachInterrupt(0, bouton_presse, RISING);
}
void loop() {
  //on allume en fonction de l'état
  if (etat==1)
    digitalWrite(rouge,HIGH);
  else
    digitalWrite(rouge,LOW);
}
```

Utilisez Fritzing