

# Terminale SI

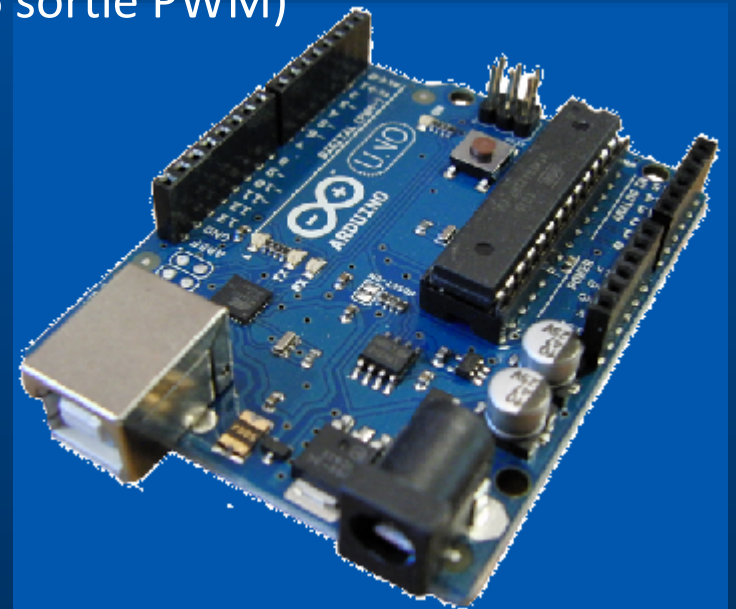
## Initiation Arduino



### Nano automate Arduino

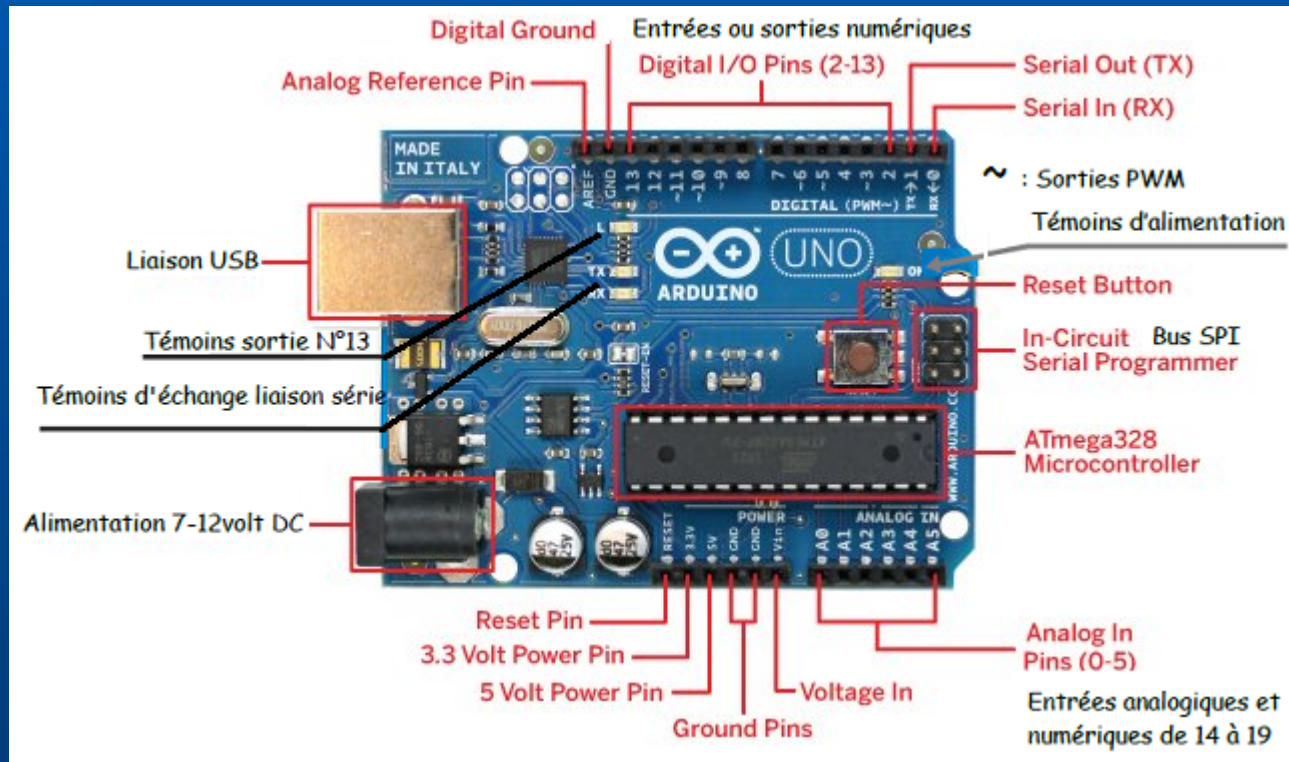
#### Modèle UNO

- Microcontrôleur : ATmega328
- Tension de fonctionnement : 5V
- Broches E/S numériques : 14 (dont 6 sortie PWM)
- Broches d'entrées analogiques 6
- Intensité maxi : 40 mA
- Mémoire Programme Flash : 32 KB
- Mémoire SRAM : 2 KB
- Mémoire EEPROM : 1 KB
- Vitesse d'horloge : 16 MHz



# Terminale SI

## Initiation Arduino



# Terminale SI

## Initiation Arduino



### Entrées et sorties numériques

Les 14 broches numériques de la carte UNO (numérotées des 0 à 13) peut être utilisée comme entrée ou sortie numérique :

instructions `pinMode()`, `digitalWrite()` et `digitalRead()`

Ces broches fonctionnent en 5V avec un maximum de 40mA.

Les résistances internes des broches :

- s'**activent** avec l'instruction `digitalWrite(broche, HIGH)`
- se **désactivent** avec l'instruction `digitalWrite(broche, LOW)`.

# Terminale SI

## Initiation Arduino



### Broches spécialisées

- Communication Série: Broches 0 (RX) et 1 (TX) de niveau TTL.
- Interruptions Externes: Broches 2 et 3 - voir instruction `attachInterrupt()`.
- 6 sorties PWM 8 bits : Broches 3, 5, 6, 9, 10, et 11 - instruction `analogWrite()`.
- SPI : Broches 10 (SS), 11 (MOSI), 12 (MISO), 13 (SCK).
- I2C : Broches 4 (SDA) et 5 (SCL).
- 6 entrées analogiques (numérotées de 0 à 5), avec une résolution de 10 bits

# Terminale SI

## Initiation Arduino



### Interface Arduino

- 1) un menu
- 2) une barre d'actions
- 3) un ou plusieurs onglets correspondant aux "sketchs"
- 4) une fenêtre de programmation
- 5) une console qui affiche les informations et erreurs de compilation et de téléversement du programme



# Terminale SI

## Initiation Arduino



### Interface de programmation

//Commentaires (1 ligne)  
/\* zone de commentaire\*/

- 1) la partie déclarative, variables et constantes (optionnelle).
- 2) la partie initialisation et configuration .  
Déclaration des entrées/sorties.
  - C'est la fonction **setup () {}**
- 3) la partie principale qui s'exécute en boucle :
  - C'est la fonction **loop {}**

```
Button | Arduino 1.0.4
Fichier Édition Croquis Outils Aide

Button

// constants won't change. They're used here to
// set pin numbers:
const int buttonPin = 2;    // the number of the pushbutton pin
const int ledPin = 13;     // the number of the LED pin

// variables will change:
int buttonState = 0;        // variable for reading the pushbutton status

void setup() {
  // initialize the LED pin as an output:
  pinMode(ledPin, OUTPUT);
  // initialize the pushbutton pin as an input:
  pinMode(buttonPin, INPUT);
}

void loop(){
  // read the state of the pushbutton value:
  buttonState = digitalRead(buttonPin);

  // check if the pushbutton is pressed.
  // if it is, the buttonState is HIGH:
  if (buttonState == HIGH) {
    // turn LED on:
    digitalWrite(ledPin, HIGH);
  }
  else {
    // turn LED off:
    digitalWrite(ledPin, LOW);
  }
}
```

# Terminale SI

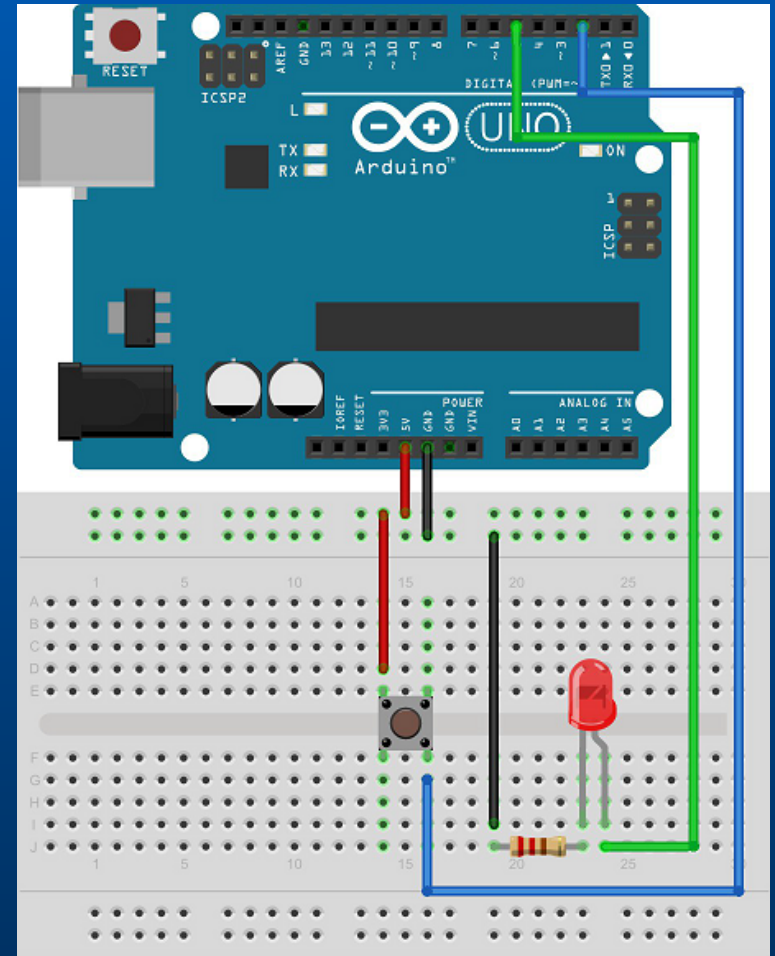
## Initiation Arduino



### Exemple 1 : lecture numérique

```
// Allumage LED par bouton
void setup()
{
    pinMode(2 , INPUT); // bouton en entrée 2
    pinMode(5 , OUTPUT); // LED en sortie 5
}

void loop()
{
    If ( digitalRead(2) ) // capteur TOR
        digitalWrite(5 , HIGH); // actionneur
    else
        digitalWrite(5 , LOW); // éteindre
    delay(100); // attente 100 ms
}
```



# Terminale SI

## Initiation Arduino



### La diode

Une diode admet un courant maximum. Au delà d'une certaine limite le composant est détruit. On protège la diode à l'aide d'une résistance qui limite l'intensité du courant.

Soit le schéma ci-contre, on donne :

- $V_e = +5V$
- $V_f = 1,6 V$
- $I_f = 10 \text{ mA}$

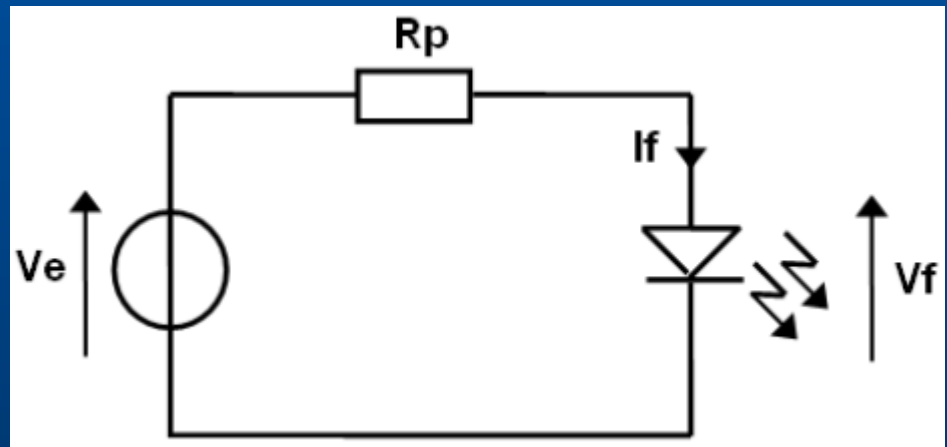
Calculer  $R_p$  :

$$V_e = U_{R_p} + V_f$$

$$U_{R_p} = V_e - V_f = 5,0 - 1,6 = 3,4 \text{ V}$$

$$\text{Loi d'Ohm : } U_{R_p} = R_p \cdot I_f$$

$$R_p = \frac{U_{R_p}}{I_f} = \frac{3,4}{10 \cdot 10^{-3}} = 340 \Omega$$





# Terminale SI

## Initiation Arduino

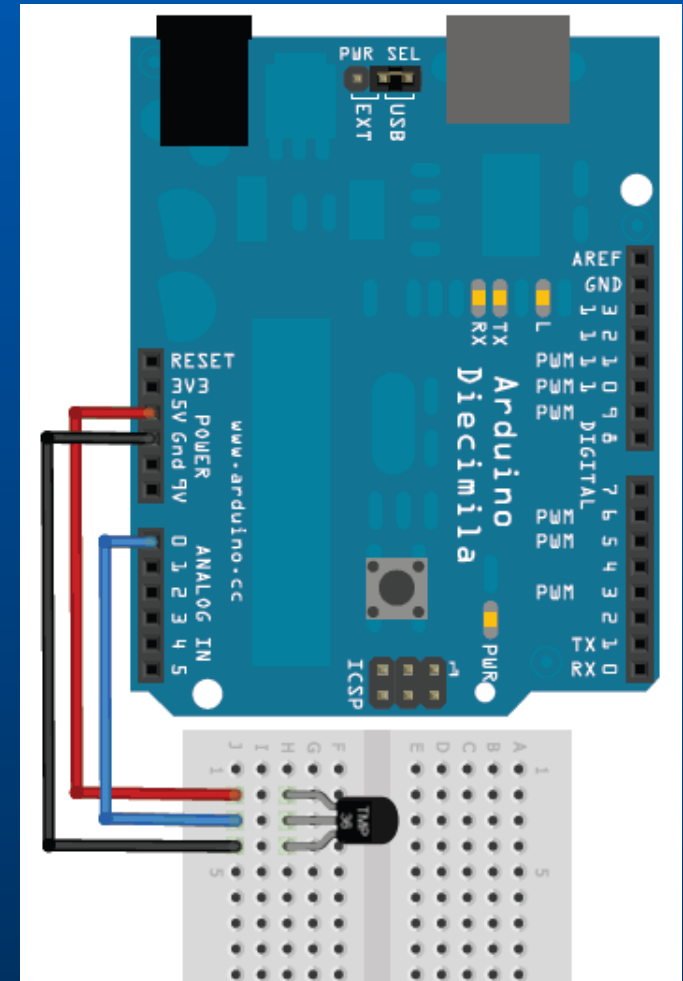


### Exemple 2 : lecture analogique

```
// lecture de température
void setup()
{
    //initialisation vitesse liaison série à 9600 bauds
    Serial.begin(9600) ;
}

void loop()
{
    int t = analogRead(A0) ;    // lecture entrée analogique

    Serial.println(t);          // affichage valeur numérique
    delay(100);                // attente 100 ms
}
```



# Terminale SI

## Initiation Arduino

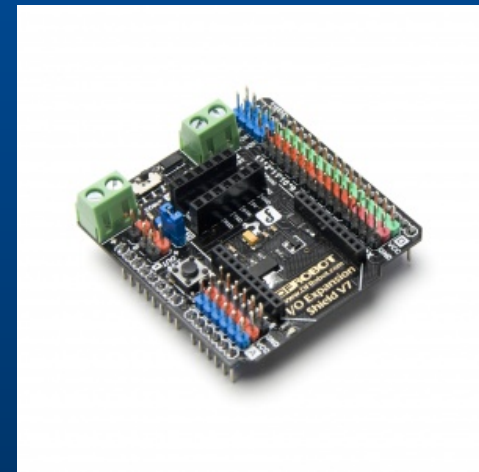


### IO Expansion Shield for Arduino V7 SKU:DFR0265

Cette carte d'extension permet de connecter plus rapidement et plus facilement les composants externes avec des connecteurs adaptés.

Des couleurs sur les fils permettent de connecter correctement les composants à la carte Arduino.

- Fil rouge correspond au +5V
- fil noir au 0V (ou masse)
- fil vert à une donnée numérique ou logique
- fil bleu à une donnée analogique



# Terminale SI

## Initiation Arduino

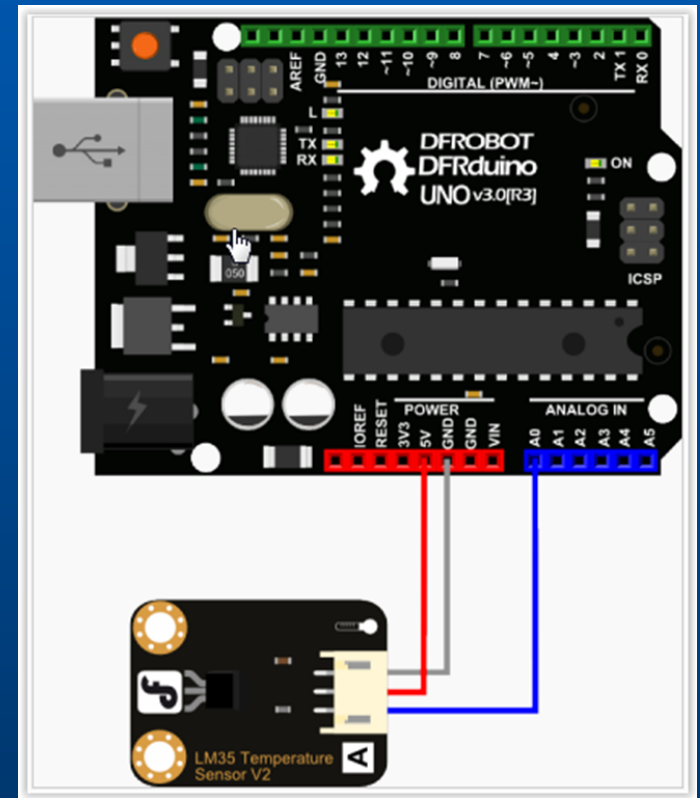


### Lecture analogique avec carte DFRobot

```
// lecture de température
void setup()
{
    //initialisation vitesse liaison série à 9600 bauds
    Serial.begin(9600) ;
}

void loop()
{
    int t = analogRead(A0) ;// lecture entrée analogique

    Serial.println(t); // affichage valeur numérique
    delay(100); // attente 100 ms
}
```



# Terminale SI

## Initiation Arduino

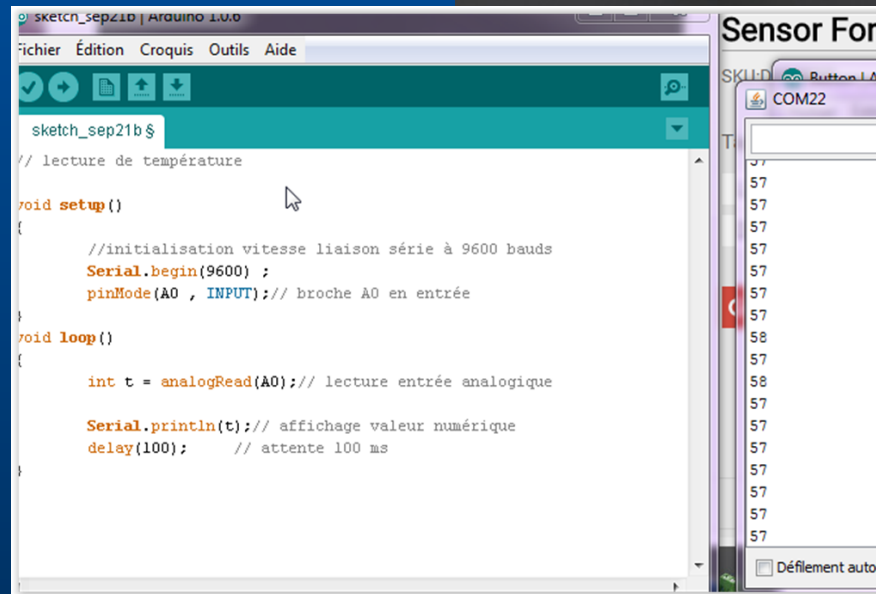
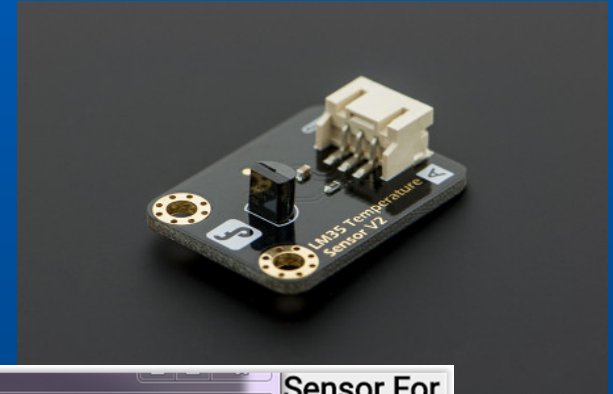


### Lecture analogique avec capteur intégré

```
// lecture de température
void setup()
{
    //initialisation vitesse liaison série à 9600 bauds
    Serial.begin(9600) ;
}

void loop()
{
    int t = analogRead(A0) ;

    Serial.println(t);
    delay(100);
}
```



# Terminale SI

## Initiation Arduino



### Sensibilité

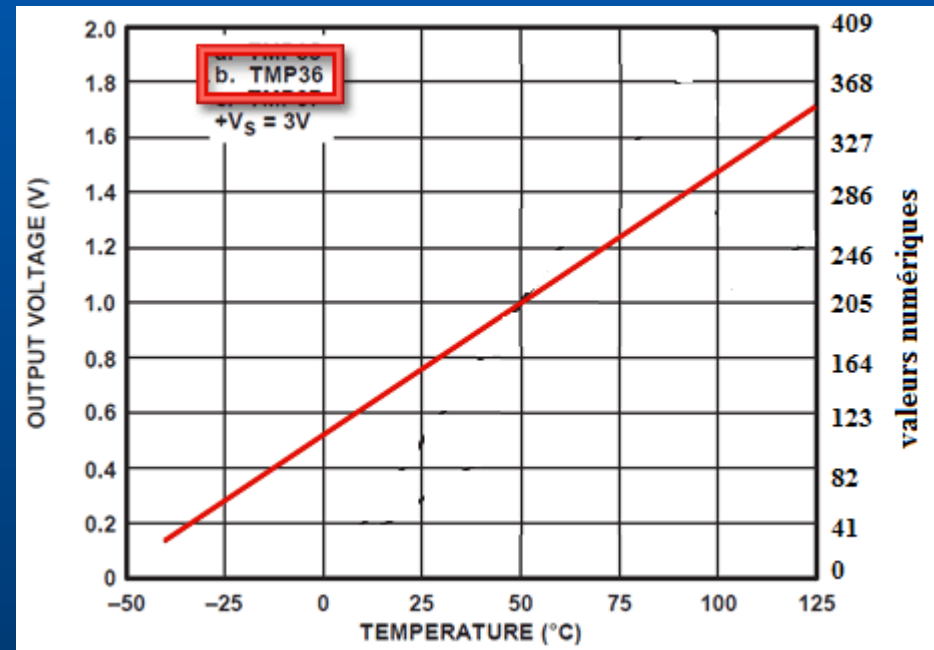
A une tension de sortie du capteur de  $t^\circ$  correspond une température

Le capteur mesure une grandeur physique sur une plage déterminée

On a donc une correspondance entre :  
un intervalle d'une grandeur physique  $\theta$   
vers un intervalle de tension  $U = [0 ; 5] \text{ V}$

$$s = \frac{\Delta U}{\Delta \text{mesurande}} = \frac{\Delta U}{\Delta \theta}$$

La sensibilité d'un capteur est le paramètre qui exprime la variation du signal de sortie (tension) en fonction de la variation du signal d'entrée.



# Terminale SI

## Initiation Arduino



### Étalonnage

A une tension de sortie du capteur de  $t^\circ$  correspond une valeur numérique sur 10 bits

Le CAN de l'arduino discrétise une grandeur analogique  $[0..5]_V$  d'un intervalle sur R vers un intervalle  $[0 ; 1023]_{CAN}$  sur N

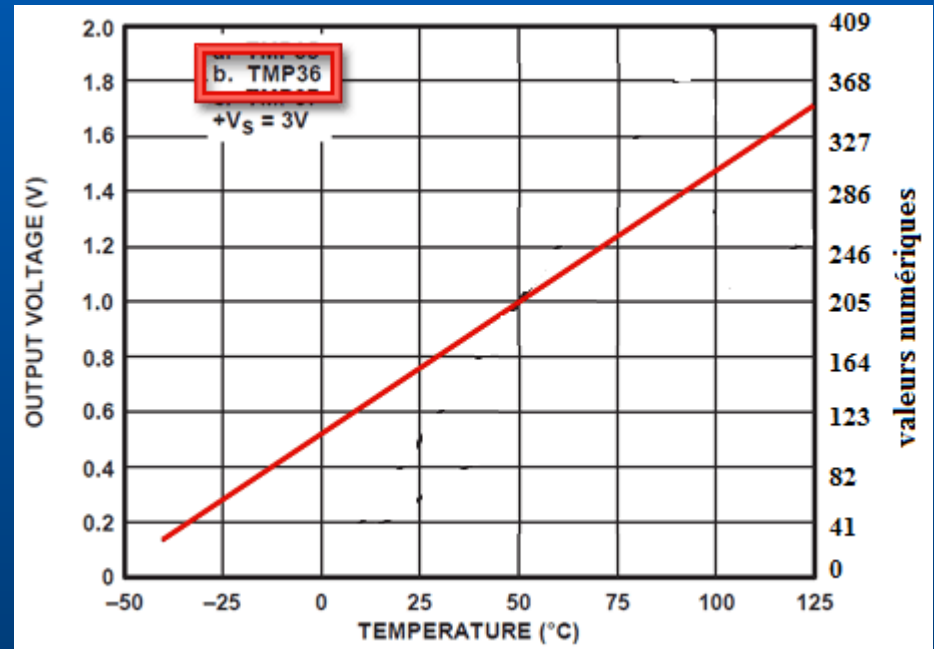
Pour afficher la valeur mesurée, il faut appliquer une fonction de conversion f.

$$f : x \rightarrow f(x) = a.x + b$$

Pour cela, il faut 2 points de mesure : c'est l'étalonnage

Le coefficient a s'appelle le **quantum**

C'est la plus petite tension analogique mesurable par le CAN



# Terminale SI

## Initiation Arduino



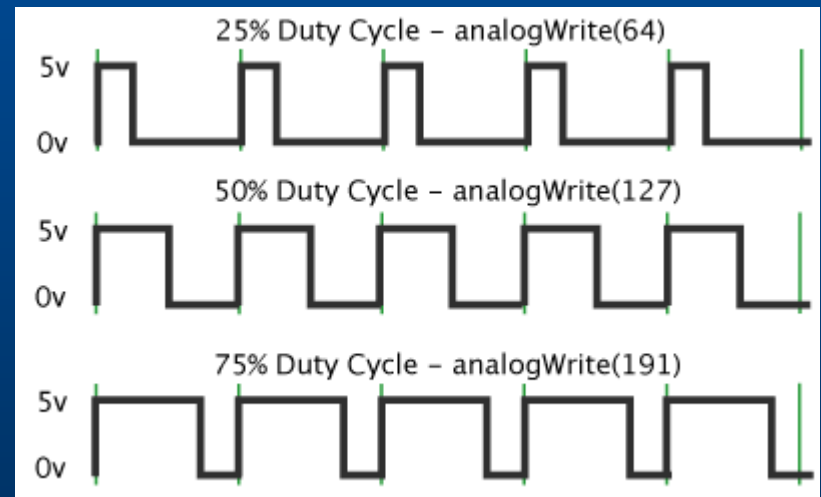
### PWM

La PWM (**Pulse Width Modulation**) génère un signal logique (valant 0 ou 1), à fréquence fixe mais dont le rapport cyclique  $\alpha$  est contrôlé numériquement.

La moyenne du signal de sortie est égale au rapport cyclique x signal d'entrée.

$$\hat{U}_s = U_e \times \alpha$$

Selon la moyenne obtenue, un moteur CC tournera plus ou moins vite.



# Terminale SI

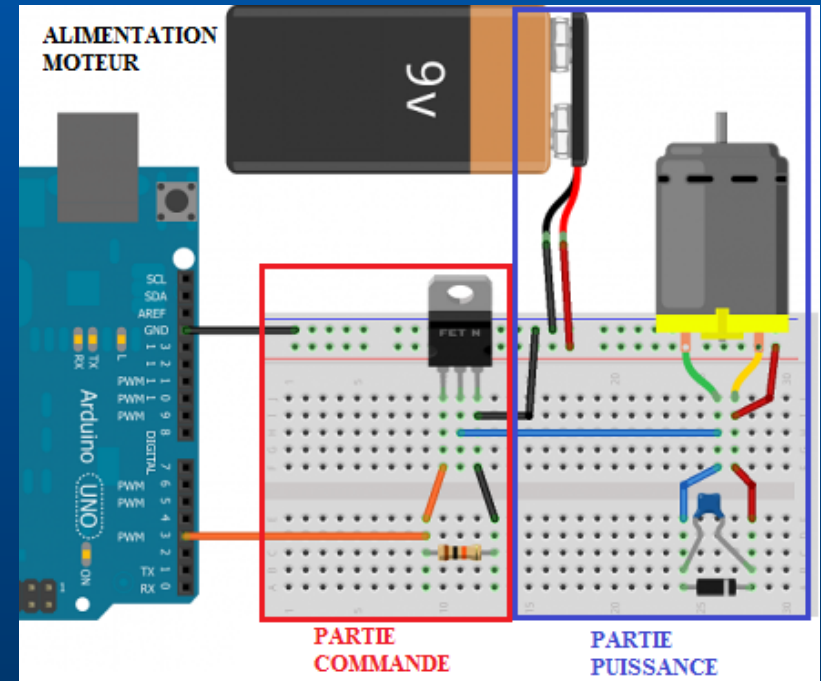
## Initiation Arduino



### Exemple 3 : PWM

```
void setup()
{
}

void loop()
{
    for (int i(0) ; i < 256 ; i++) { // accélération
        analogWrite(3, i) ;
        delay(50); // attente 50 ms
    }
    for (int i(0) ; i < 256 ; i++) { // décélération
        analogWrite(3, 255 - i) ;
        delay(50); // attente 50 ms
    }
}
```





# Terminale SI

## Initiation Arduino



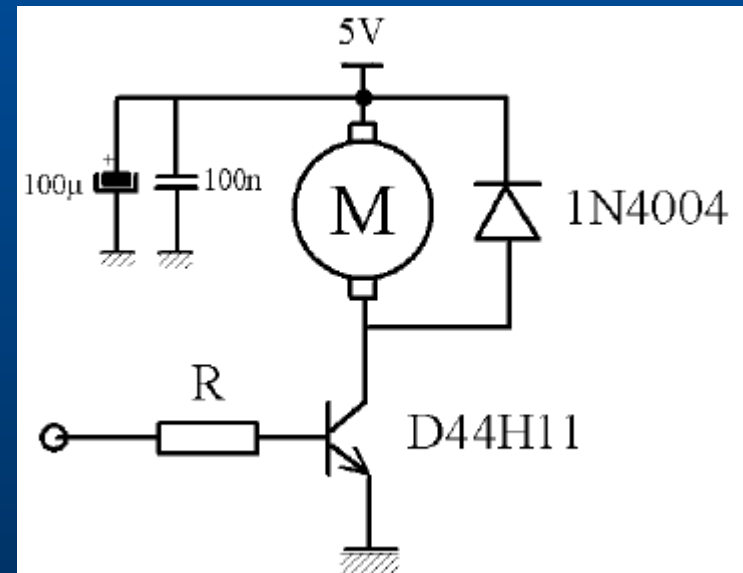
### Interface de puissance

La carte Arduino fait partie de la chaîne d'information et non d'énergie. A ce titre, elle ne possède pas de circuit de puissance et il faut utiliser une interface pour ne pas l'endommager.

On utilise alors le montage ci-contre.

La carte Arduino pilote le transistor qui commande à son tour le moteur.

Une diode de roue libre, protège le transistor des risques de surtension quand le moteur s'arrête.



# Terminale SI

## Initiation Arduino



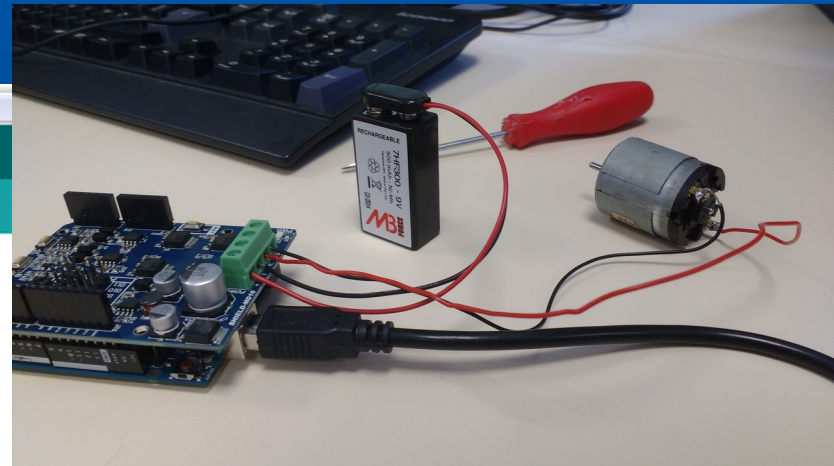
### Utilisation d'une carte d'extension à moteur DC

```
Fichier  Édition  Croquis  Outils  Aide
motor$

const int motorPin = 3;
void setup() {
  Serial.begin(9600);
}

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

    if( isdigit(ch) ) {
      int speed = map(ch, '0', '9', 0, 255); // map = convertisseur de valeur
      analogWrite(motorPin, speed); // on envoie la valeur au pin 3
      // PWM avec le pourcentage (/255) ou la sortie est à l'état haut
      Serial.println(speed);
    }
    else {
      Serial.print("caractère inattendu : ");
      Serial.print(ch);
    }
  }
}
```

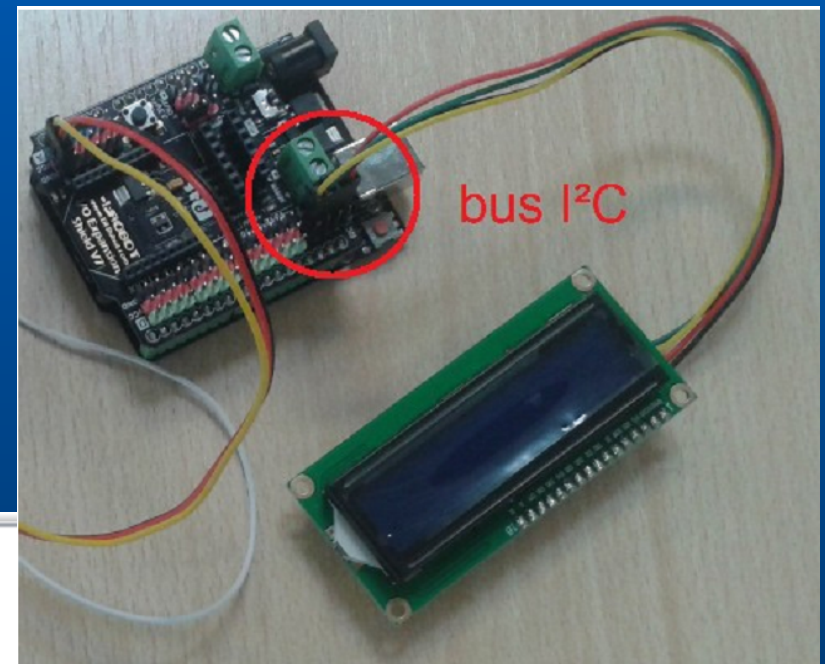


# Terminale SI

## Initiation Arduino



### Utilisation d'un afficheur LCD



```
I2C Scanner  
Scanning...  
dispositif I2C detecté Ã  l'adresse : 0x20  
dispositif I2C detecté Ã  l'adresse : 0x7C  
terminé
```