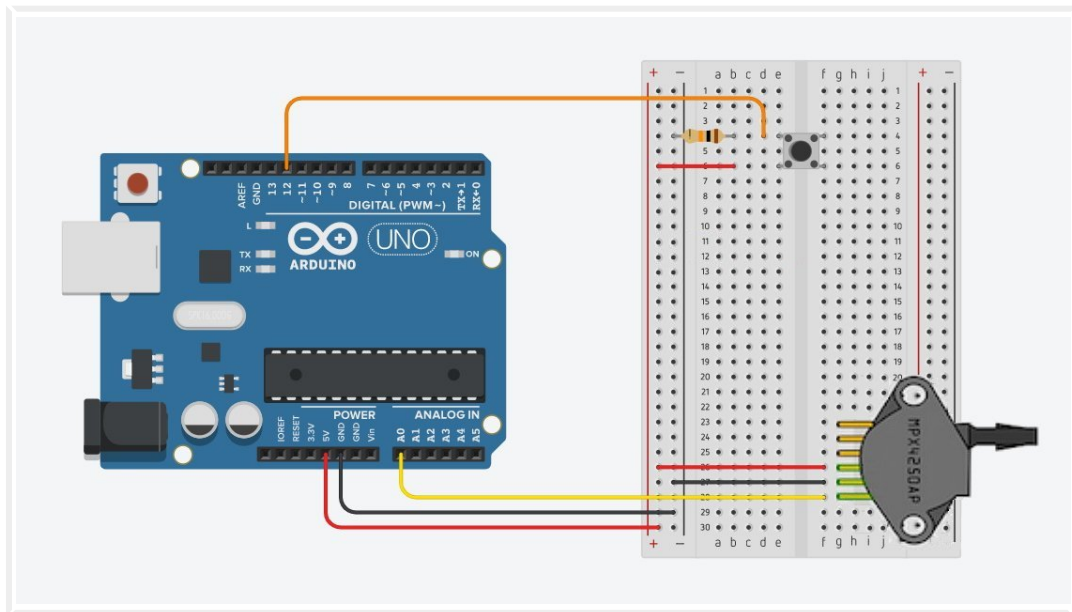


Capteurs de pression

(Mesure d'une pression absolue avec un capteur MPX4250AP)



. Liste des composants :

- . 1 capteur de pression MPX4250AP
- . 1 résistance de 10 k Ω (résistance du circuit du bouton poussoir)
- . 1 bouton poussoir
- . 1 plaque d'essais
- . Fils de connexion

. Objectif

Dans cette activité, nous allons simplement mettre en œuvre l'utilisation d'un capteur de pression dont la sortie est reliée à l'entrée A0 de l'Arduino (Cf. circuit d'étude), pour afficher dans le moniteur série, la tension mesurée et la pression correspondante après un appui sur le bouton poussoir. Un nouvel appui sur le bouton poussoir arrête les mesures.

. Rappels :

L'unité de pression du système international (S.I.) est le pascal (Pa), qui correspond à une force d'1 newton exercée sur une surface d'1 mètre carré (**1 Pa = 1 N/m²**).

Le pascal étant une unité très petite par rapport aux pressions mesurées à la surface du globe (la pression atmosphérique moyenne enregistrée au niveau de la mer vaut 101 325 Pa), les météorologistes et les climatologues préfèrent utiliser l'hectopascal (**1 hPa = 10² Pa**).

D'autres unités ont été préalablement utilisées en météorologie, par exemple le millimètre de mercure (mmHg ou Torr), le millibar (mbar) et l'atmosphère normale (atm) :

- Le millimètre de mercure correspond à la pression exercée à 0°C par une colonne de mercure d'1 millimètre de hauteur (**1 mmHg = 133,322 Pa = 1,333 22 hPa**).
- Le millibar vaut 10⁻³ bars (**1 mbar = 10² Pa = 1 hPa**).
- L'atmosphère normale correspond à la pression atmosphérique moyenne mesurée au niveau moyen de la mer à la latitude de Paris (**1 atm = 101 325 Pa = 1 013,25 hPa = 1 013,25 mbar = 760 mmHg**).

De nombreux types de capteurs de pression peuvent à priori être utilisés avec une carte Arduino sur une entrée analogique. Cependant, outre la gamme de pressions mesurables, un autre paramètre essentiel pour une utilisation avec un Arduino est la sensibilité du capteur souvent exprimé en **mV/kPa**.

En effet, nous avons vu que la résolution par défaut du convertisseur analogique/numérique de l'Arduino était de l'ordre de 5 mV. Hors beaucoup de capteurs de pression ne disposant pas d'amplification ont une résolution de 0,5 mV/kPa, ce qui rend impossible leur utilisation avec notre microcontrôleur.

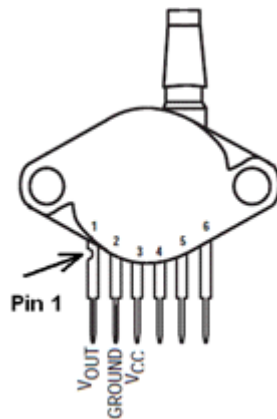
Le capteur **MPX4250AP** a quant à lui une sensibilité de 20 mV/kPa pour une plage de mesure allant de 20 à 250 kPa (soit, une tension de sortie du capteur entre 0,2 V et 4,8 V). Ce capteur est parfaitement adapté à une utilisation avec un Arduino.

. Capteur de pression MPX4250AP :



(Vue de dessus)

Le capteur de pression absolue MPX4250AP dispose de 6 broches, mais seules trois bornes sont utilisées pour le connecter à une carte Arduino :



(Vue de dessous)

VCC : +5 V

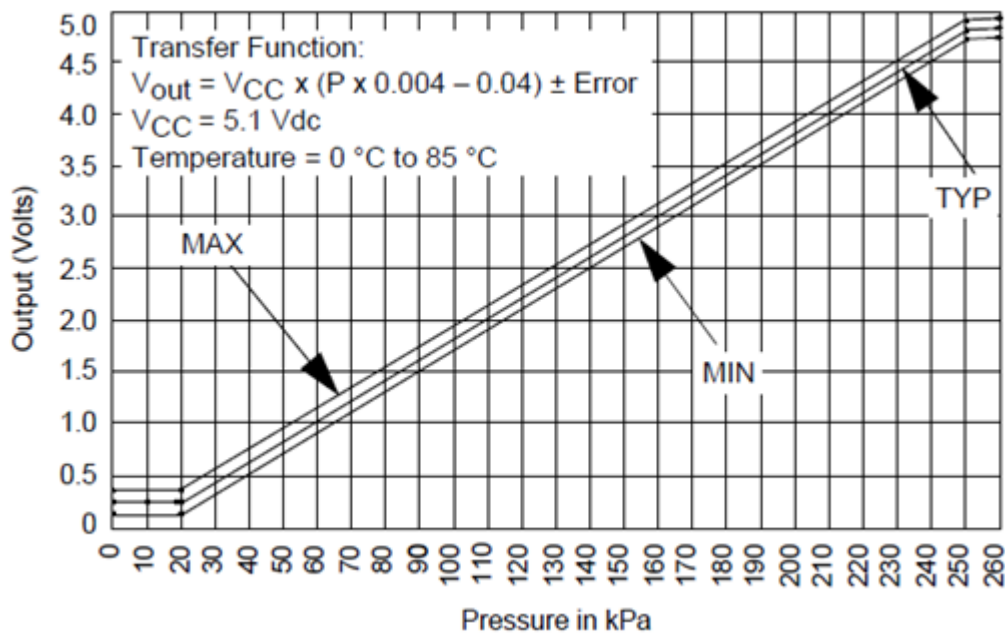
GROUND : masse

Vout : signal de sortie sur une entrée analogique de la carte.

Caractéristiques :

- . Tension de fonctionnement : de 4,85V à 5,35V
- . Courant circuit alimentation maximal : 10 mA
- . Sensibilité : **20 mV / kPa**
- . Plage de mesure : 20 à 250 kPa
- . Tension de sortie (V_{out}) : 0,2 V à 4,8 V
- . Précision : $\pm 1,5\%$ sur V_{out}
- . Courant de charge maximal (signal de sortie) : 0,1 mA
- . Temps de réponse : 1 ms
- . Température de fonctionnement : 0 – 85 °C

La fiche technique du capteur, fourni par le constructeur, donne la courbe de transfert reliant la pression absolue P en kPa à la tension de sortie V_{out} en V :



On a donc :

$$V_{out} = 5,0 \times (P \times 0,004 - 0,04)$$

On obtient alors la pression absolue à partir de la mesure de la tension de sortie par :

$$P \text{ (en kPa)} = \frac{V_{out}}{0,02} + 10$$

Remarques :

Il existe des capteurs de pression relative et de pression absolue.

En pression relative, on mesure une valeur de pression par rapport à la pression ambiante (pression atmosphérique). Ce qui signifie que la pression mesurée peut évoluer quelque peu au gré des variations de pression atmosphérique (climat, altitude par rapport au niveau de la mer).

D'un point de vue pratique, une mesure de pression absolue s'effectue par rapport au vide idéal ou absolu. C'est pourquoi ce type de mesure est indépendant du temps qu'il fait ou de l'altitude.

[. Le programme](#)

Voici le code de l'activité :

Capteurs_pression

```
// Déclaration des constantes et variables
```

```
const int PinSensor=0;
const int PinButton= 12;
```

```
int ValSensor = 0;
float tension = 0.0;
float Pression = 0.0;
float OldPression = 0.0;
```

```
int ValButton = 0;
int OldValButton = 0;
int State = 0;
int OldState = 0;
```

```
// Initialisation des entrées et sorties
```

```
void setup() {
  Serial.begin(9600);
  pinMode(PinButton, INPUT);
  Serial.println("Appuyez sur le bouton poussoir pour commencer les mesures.");
}
```

```

// Fonction principale en boucle

void loop() {
  ValButton = digitalRead(PinButton);
  delay(10);

  if ((ValButton == HIGH) && (OldValButton == LOW))
  {
    State=1-State;
  }
  OldValButton = ValButton;

  if (State==1)
  {
    if (OldState == 0)
    {
      Serial.println("Mesure de la pression en cours.");
      Serial.println("");
      Serial.println ("Tension Entree A0 (en V) ; Pression (en kPa):");
      OldState=1;
    }
    ValSensor = analogRead(PinSensor);
    tension = (ValSensor/1023.0)*5.0;

    Pression = tension / 0.02 + 10;

    if (OldPression != Pression)
    {
      Serial.print(tension);
      Serial.print(" ; ");
      Serial.println(Pression,1);
      OldPression = Pression;
    }
    delay(500);
  }

  else
  {
    if (OldState == 1){
      Serial.println("Fin des mesures.");
      OldState = 0;}
  }
}

```

Déroulement du programme :

– 1. Déclaration des constantes et variables :

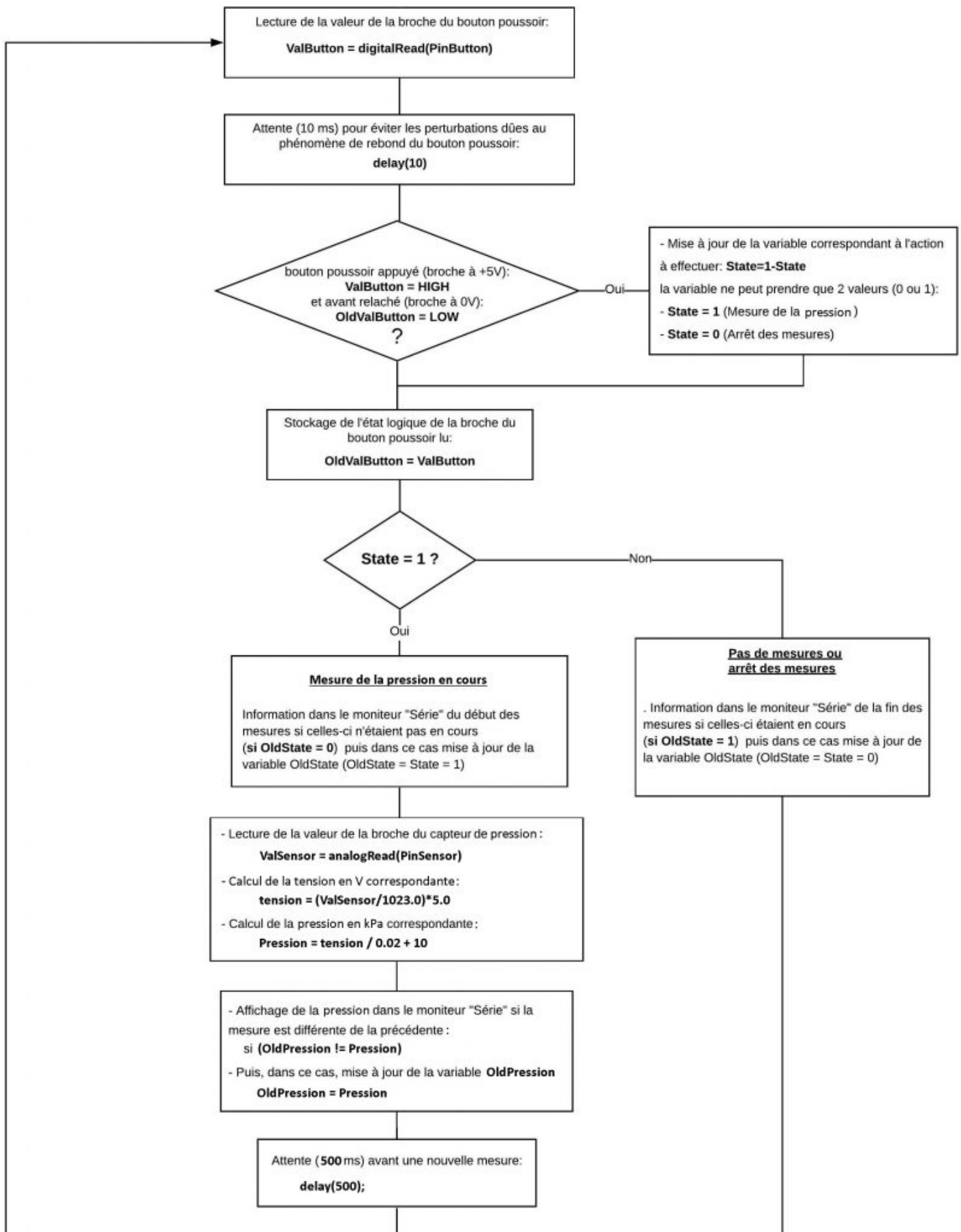
- . **const int PinSensor = 0** (broche du capteur de pression)
- . **const int PinButton = 12** (broche du bouton poussoir)
- . **int ValSensor = 0** (variable nombre entier pour stocker la valeur de la broche du capteur)
- . **float tension = 0.0** (variable nombre décimal pour stocker le résultat du calcul de la tension de la broche du capteur)

- . **float Pression = 0.0** (variable nombre décimal pour stocker le résultat du calcul de la pression)
- . **float OldTPression = 0.0** (variable nombre décimal pour stocker le résultat du calcul de la pression précédent)
- . **int ValButton = 0** (variable nombre entier pour stocker la valeur de la broche du bouton poussoir)
- . **int OldValButton = 0** (variable nombre entier pour stocker la valeur précédente de la broche du bouton poussoir)
- . **int State = 0** (variable nombre entier correspondant à l'action à effectuer)
- . **int OldState = 0** (variable nombre entier correspondant à l'action effectuée précédemment)

– 2. Initialisation des entrées et sorties :

- . **Initialisation de la liaison série à un débit de 9600 bauds**
- . **Initialisation de la broche du bouton poussoir en entrée**

– 3. Fonction principale en boucle :



Résultats dans le moniteur série :

