Pression : Mesurer & Exploiter

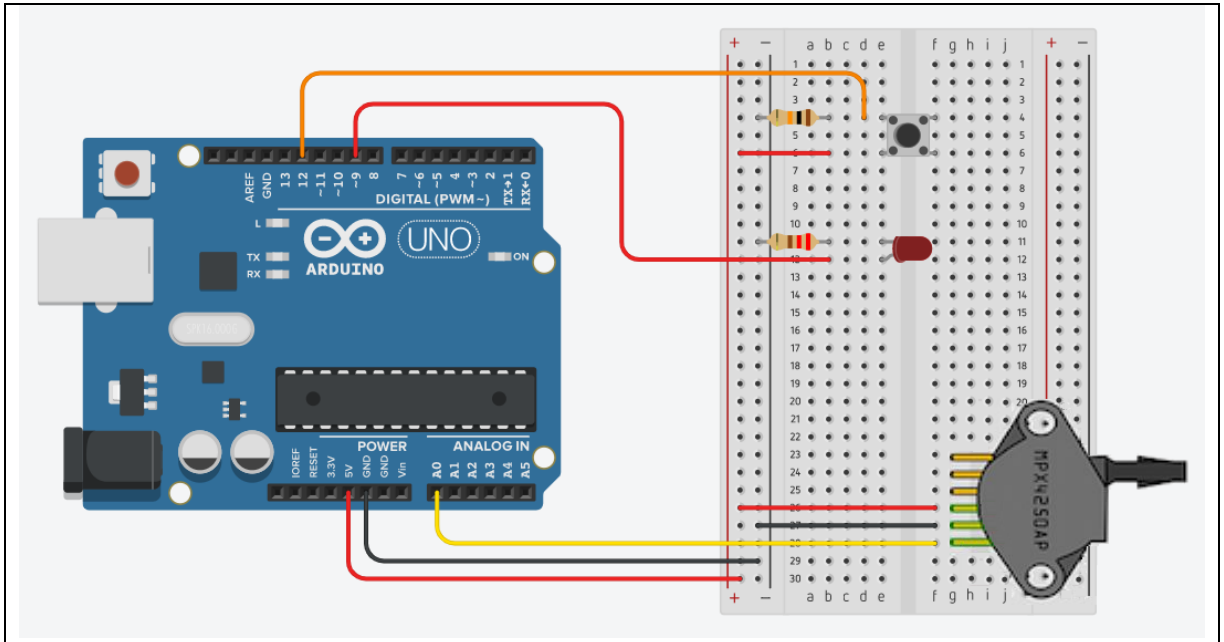
Dans le domaine des sciences, une autre grandeur physique fréquemment utilisée est la pression exprimée en Pascal (Pa).

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.

Toutes les activités de mesure de pression et d'exploitation seront réalisées avec le circuit suivant :



. Liste des composants :

. 1 capteur de pression MPX4250AP

. 1 DEL rouge

. 1 résistance de 220 Ω (résistance de protection de la DEL)

. 1 résistance de 10 kΩ (résistance du bouton poussoir)

. 1 bouton poussoir

. 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 = 10^2 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 ^{0}C par une colonne de mercure d'1 millimètre de hauteur (**1 mmHg = 133,322 Pa = 1,333 22 hPa**).

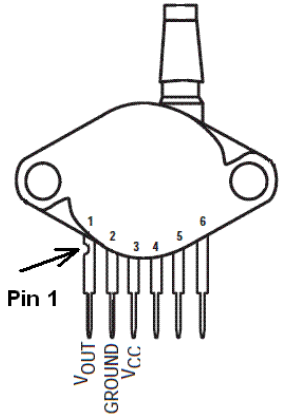
- Le millibar vaut 10^{-3} bars (**1 mbar = 10^2 Pa = 1 hPa**).

- 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**).

. 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 (Vout) : 0,2 V à 4,8 V

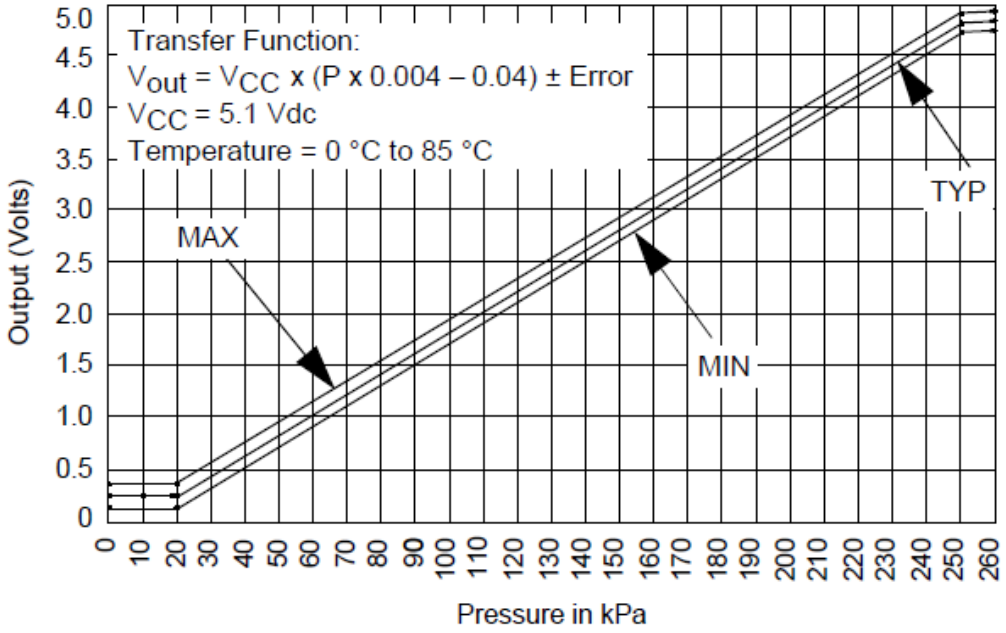
. Précision : ± 1,5% sur Vout

. 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 Vout 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 :

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

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.

Activité 1 : Mesure d'une pression absolue avec un capteur MPX4250AP

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, 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.

. Le programme

Voici le code de l'activité (**"Activity1.ino" dans le dossier "Codes/Pression"**) :

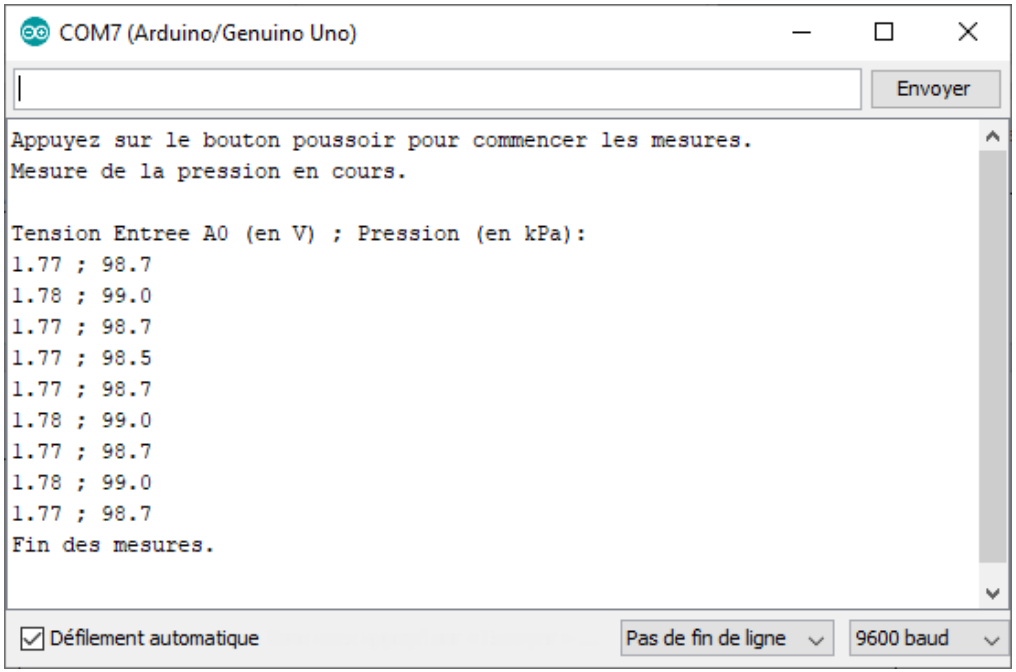
Activity1

```
// 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;}
    }
}
```

Résultats dans le moniteur série :



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)

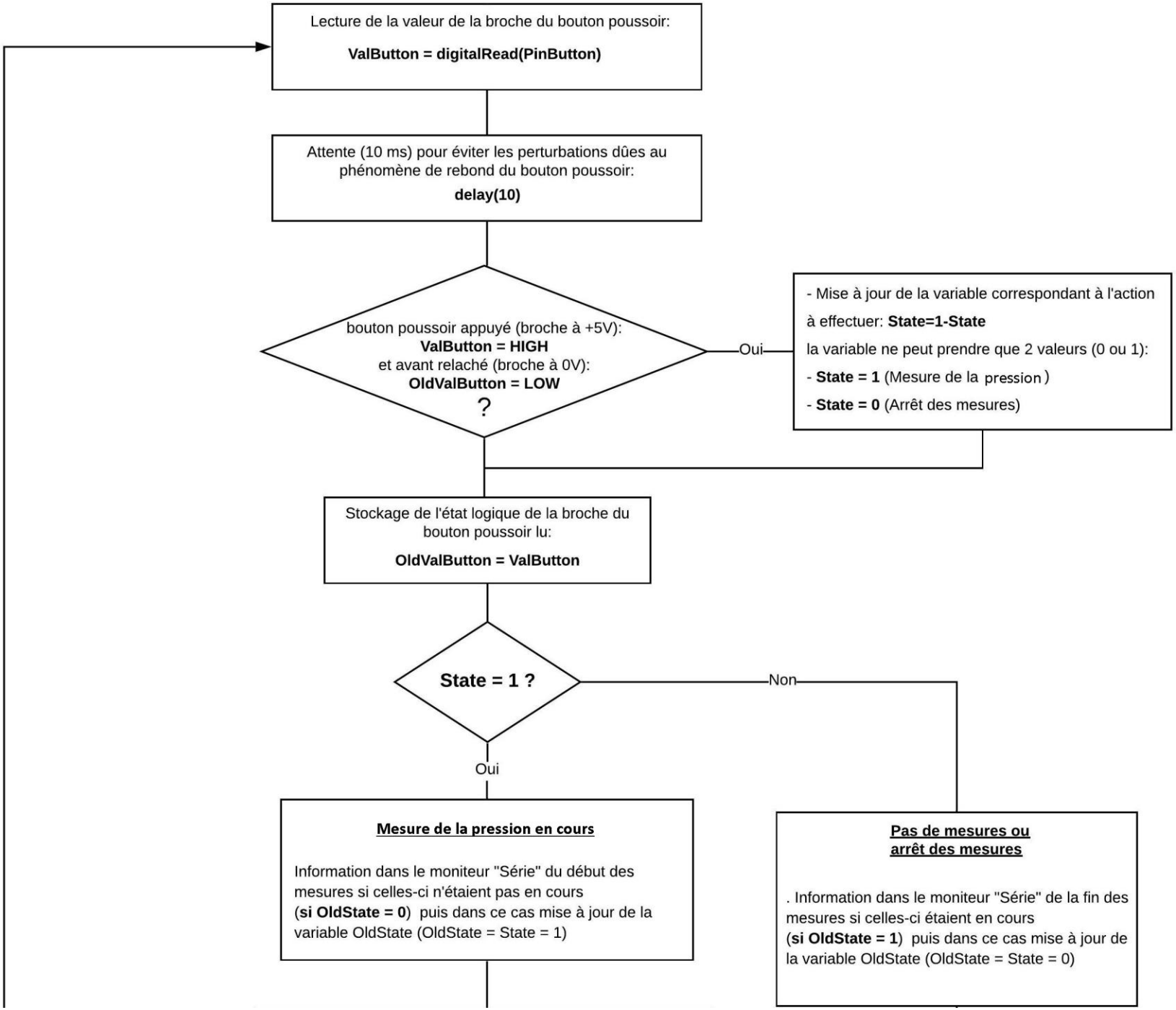
. **int ValSensor = 0** (variable nombre entier valeur broche du capteur)
. **float tension = 0.0** (variable nombre décimal calcul tension broche capteur)
. float Pression = 0.0 (variable nombre décimal calcul pression)
. float OldTPression = 0.0 (variable nombre décimal ancien calcul pression)

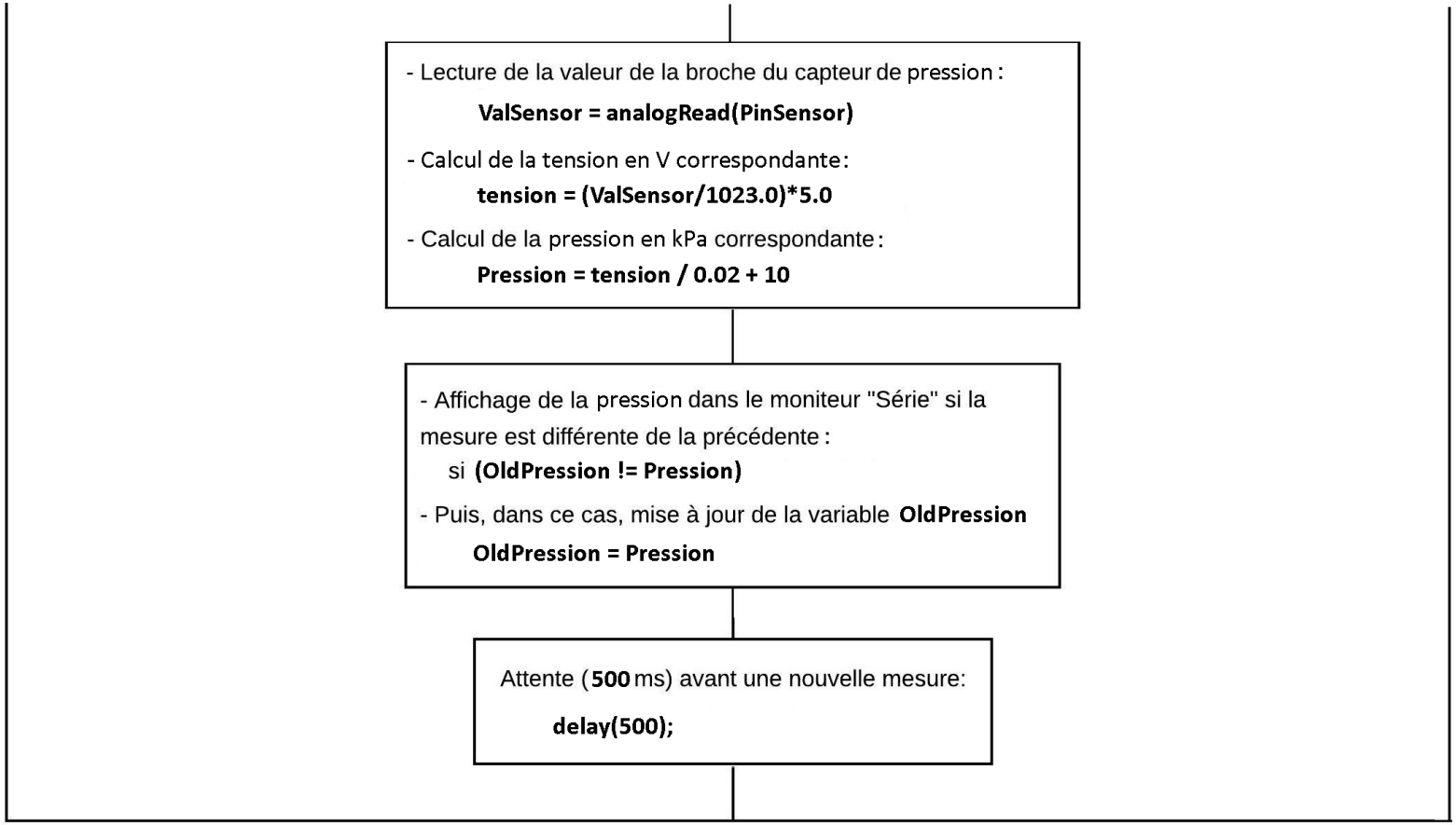
. int ValButton = 0 (variable nombre entier valeur broche bouton poussoir)
. int OldValButton = 0; (variable nbr entier ancienne valeur broche btn poussoir)
. int State = 0; (variable nombre entier pour action à effectuer)
. int OldState = 0; (variable nombre entier pour action effectuée précédemment)

- 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 :

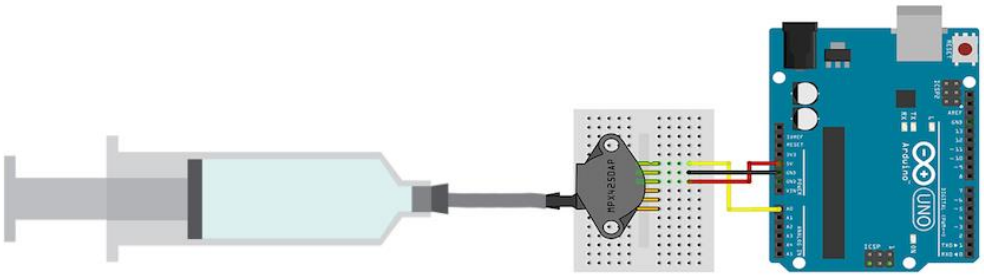

Lecture de la valeur de la broche du bouton poussoir:
ValButton = digitalRead(PinButton)
Attente (10 ms) pour éviter les perturbations dûes au phénomène de rebond du bouton poussoir:
delay(10)
bouton poussoir appuyé (broche à +5V):
ValButton = HIGH
et avant relaché (broche à 0V):
OldValButton = LOW
?
Oui
- Mise à jour de la variable correspondant à l'action à effectuer: State=1-State
la variable ne peut prendre que 2 valeurs (0 ou 1):
- State = 1 (Mesure de la pression)
- State = 0 (Arrêt des mesures)
Stockage de l'état logique de la broche du bouton poussoir lu:
OldValButton = ValButton
State = 1 ?
Non
Oui
Mesure de la pression en cours
Information dans le moniteur "Série" du début des mesures si celles-ci n'étaient pas en cours
(si OldState = 0) puis dans ce cas mise à jour de la variable OldState (OldState = State = 1)
Pas de mesures ou arrêt des mesures
. Information dans le moniteur "Série" de la fin des mesures si celles-ci étaient en cours
(si OldState = 1) puis dans ce cas mise à jour de la variable OldState (OldState = State = 0)


- Lecture de la valeur de la broche du capteur de pression :
ValSensor = analogRead(PinSensor)
- Calcul de la tension en V correspondante :
tension = (ValSensor/1023.0)*5.0
- Calcul de la pression en kPa correspondante :
Pression = tension / 0.02 + 10
- Affichage de la pression dans le moniteur "Série" si la mesure est différente de la précédente :
si (OldPression != Pression)
- Puis, dans ce cas, mise à jour de la variable OldPression
OldPression = Pression
Attente (500 ms) avant une nouvelle mesure:
delay(500);

Activité 2 : Indicateur de pression

De façon à s'assurer que la pression mesurée par notre capteur MPX4250AP ne soit pas supérieure ou inférieure à la pression maximale (250 kPa) ou minimale (20 kPa) admissible, nous allons dans cette activité utiliser une DEL rouge qui sera allumée quand la pression est supérieure ou inférieure à des seuils à définir afin de prévenir de leurs dépassements.

La pression mesurée sera modifiée avec une seringue d'un volume utile de 60 mL fixée au capteur par l'intermédiaire d'un tuyau suivant le montage ci-dessous :



En déplaçant le piston, initialement placé sur la graduation 30 mL, on fait varier le volume de l'air enfermé dans le corps de la seringue et donc la pression appliquée sur le capteur.

. Le programme

Voici le code de l'activité (**"Activity2.ino" dans le dossier "Codes/Pression"**) :

Activity2

```
// Déclaration des constantes et variables

const int PinSensor = 0;
const int PinButton = 12;
const int PinLed = 9;
const int PMax = 200;
const int PMin = 55;

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);
  pinMode(PinLed,OUTPUT);
  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 ("Pression (en kPa):");
   OldState=1;
  }
  ValSensor = analogRead(PinSensor);
  tension = (ValSensor/1023.0)*5.0;

  Pression = tension / 0.02 + 10;

  if (OldPression != Pression)
  {
    Serial.println(Pression,1);
    OldPression = Pression;
  }

  if (Pression>PMax or Pression<PMin)
  {
    digitalWrite(PinLed,HIGH);
  }
  else {
    digitalWrite(PinLed,LOW);
  }
  delay(500);
}
else
  {
    if (OldState == 1){
    Serial.println("Fin des mesures.");
    digitalWrite(PinLed,LOW);
    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)
. **const int PinLed = 9** (broche de la DEL)
. const int PMax = 200 (constante nombre entier valeur pression max en kPa)
. **const int PMin = 55** (constante nombre entier valeur pression min en kPa)

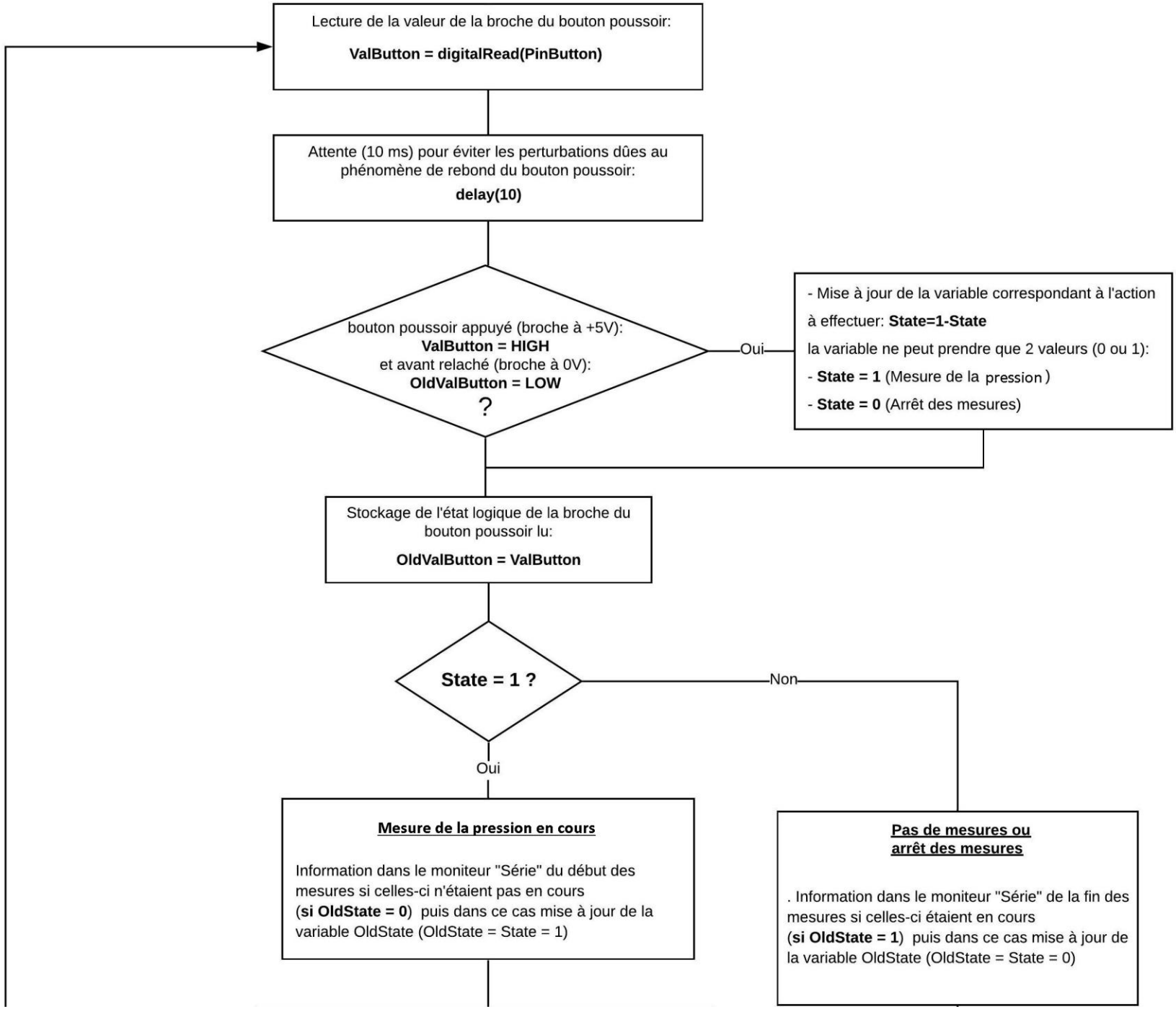
. **int ValSensor = 0** (variable nombre entier valeur broche du capteur)
. **float tension = 0.0** (variable nombre décimal calcul tension broche capteur)
. float Pression = 0.0 (variable nombre décimal calcul pression)
. float OldPression = 0.0 (variable nombre décimal ancien calcul pression)

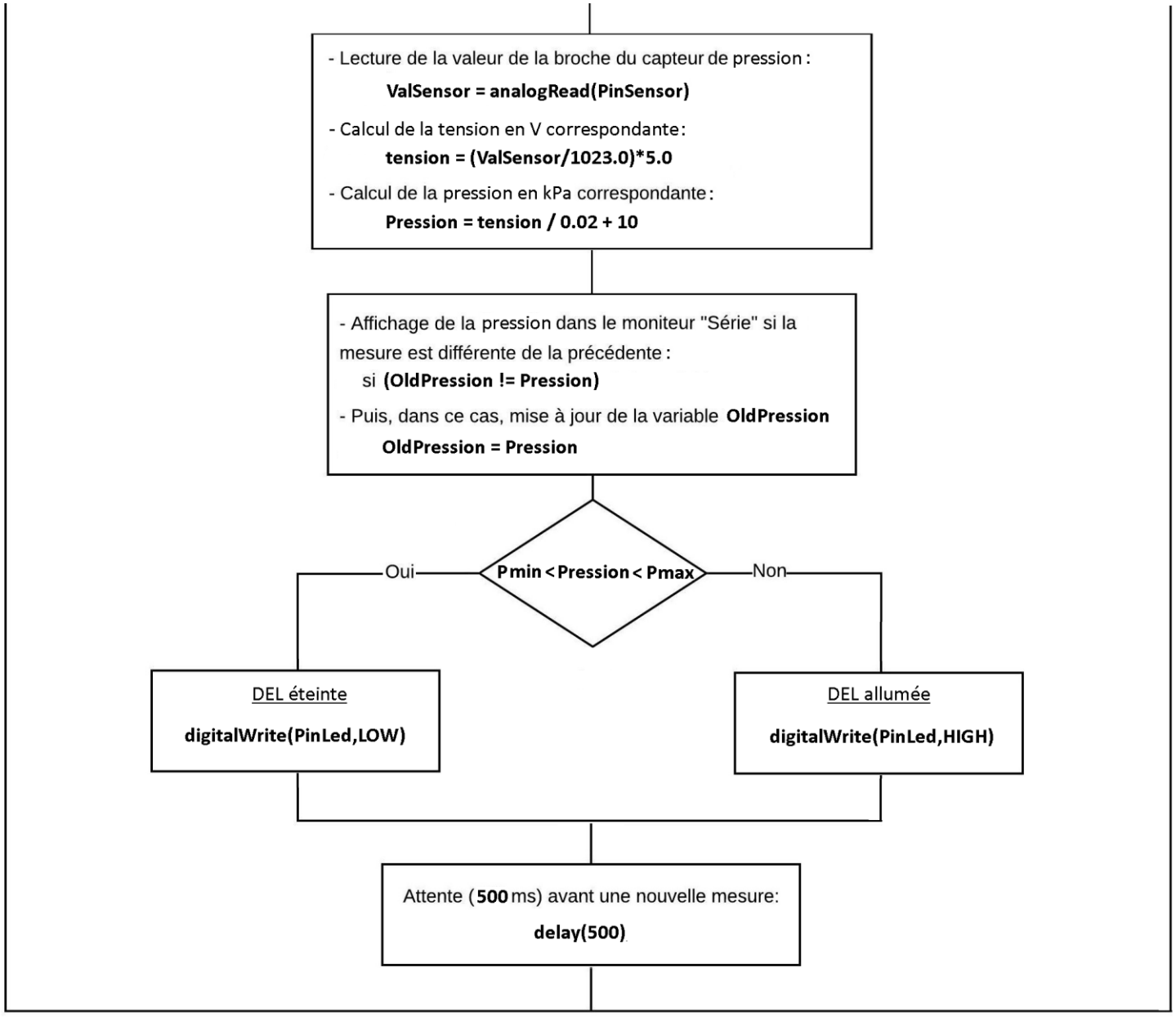
. int ValButton = 0 (variable nombre entier valeur broche bouton poussoir)
. int OldValButton = 0; (variable nbr entier ancienne valeur broche btn poussoir)
. int State = 0; (variable nombre entier pour action à effectuer)
. int OldState = 0; (variable nombre entier pour action effectuée précédemment)

- 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
. Initialisation de la broche de la DEL en sortie

- 3. Fonction principale en boucle


Lecture de la valeur de la broche du bouton poussoir:
ValButton = digitalRead(PinButton)
Attente (10 ms) pour éviter les perturbations dûes au phénomène de rebond du bouton poussoir:
delay(10)
bouton poussoir appuyé (broche à +5V):
ValButton = HIGH
et avant relaché (broche à 0V):
OldValButton = LOW
?
Oui
- Mise à jour de la variable correspondant à l'action à effectuer: State=1-State
la variable ne peut prendre que 2 valeurs (0 ou 1):
- State = 1 (Mesure de la pression)
- State = 0 (Arrêt des mesures)
Stockage de l'état logique de la broche du bouton poussoir lu:
OldValButton = ValButton
State = 1 ?
Non
Oui
Mesure de la pression en cours
Information dans le moniteur "Série" du début des mesures si celles-ci n'étaient pas en cours
(si OldState = 0) puis dans ce cas mise à jour de la variable OldState (OldState = State = 1)
Pas de mesures ou arrêt des mesures
. Information dans le moniteur "Série" de la fin des mesures si celles-ci étaient en cours
(si OldState = 1) puis dans ce cas mise à jour de la variable OldState (OldState = State = 0)


- Lecture de la valeur de la broche du capteur de pression :
ValSensor = analogRead(PinSensor)
- Calcul de la tension en V correspondante :
tension = (ValSensor/1023.0)*5.0
- Calcul de la pression en kPa correspondante :
Pression = tension / 0.02 + 10
- Affichage de la pression dans le moniteur "Série" si la mesure est différente de la précédente :
si (OldPression != Pression)
- Puis, dans ce cas, mise à jour de la variable OldPression
OldPression = Pression
Pmin < Pression < Pmax
Oui
Non
DEL éteinte
digitalWrite(PinLed,LOW)
DEL allumée
digitalWrite(PinLed,HIGH)
Attente (500 ms) avant une nouvelle mesure:
delay(500)

Activité 3 : Vérification de la loi de Boyle-Mariotte

L'objectif de cette activité est de vérifier la loi de Boyle-Mariotte à l'aide de notre capteur de pression MPX4250AP et de la seringue d'un volume utile de 60 mL suivant le même montage que l'activité précédente.

. Énoncé de la loi de Boyle-Mariotte :

À température constante, pour une quantité de matière donnée de gaz, le produit de la pression P par le volume V de ce gaz ne varie pas :

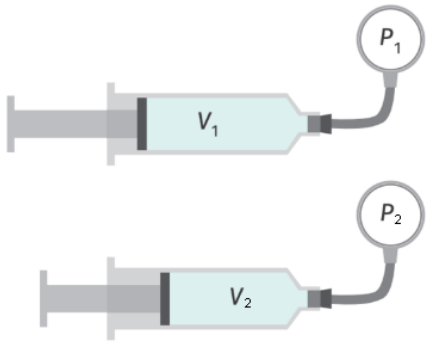
$$P \times V = \text{constante}$$

Si un gaz subit une transformation qui l'amène d'un état n°1 (son volume est V1 et sa pression P1) à un état n°2 (son volume est V2 et sa pression P2) alors la loi de Boyle-Mariotte peut s'écrire:

$$P1.V1 = P2.V2$$

Cette relation est valable à condition que:

- La transformation soit à température constante,
- Les deux pressions P1 et P2 soient exprimées dans la même unité de pression qui peut être le pascal, l'hectopascal, le bar, etc…,
- Les deux volumes soient exprimés dans la même unité de volume qui peut être le litre, le mètre cube, le centimètre cube, etc.



. Descriptif de l'activité

En déplaçant le piston, initialement placé sur la graduation 30 mL, on fait varier le volume de l'air enfermé dans le corps de la seringue reliée au capteur de pression.

Après avoir appuyé sur le bouton poussoir, le volume lu (entre 10 et 60 mL) sur le corps de la seringue est saisi au clavier dans le moniteur série.
La pression est ensuite mesurée par le capteur qui délivre une tension électrique proportionnelle à la pression. Une moyenne est réalisée sur dix mesures et le résultat de la pression moyenne (p en kPa) pour le volume (V en mL) est affiché dans le moniteur série.

Les mesures pour le volume V sont arrêtées en appuyant sur le bouton poussoir.
Une nouvelle mesure de la pression pour un volume différent est réalisable en appuyant de nouveau sur le bouton poussoir.

Il est donc possible d'acquérir des couples de données (p, V) afin de vérifier la loi de Boyle-Mariotte.

. Le programme

Voici le code de l'activité (**"Activity3.ino" dans le dossier "Codes/Pression"**) :

Activity3

```
// Déclaration des constantes et variables

const int PinSensor = 0;
const int PinButton = 12;
const int PinLed = 9;
const int PMax = 240;
const int PMin = 50;

int ValSensor = 0;
float tension = 0.0;
float Pression = 0.0;
float OldPression = 0.0;
int Vol = 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);
  pinMode(PinLed,OUTPUT);
  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.print("Veuillez saisir le volume V de la seringue en mL ");
    Serial.println("(valeur entre 10 et 60 mL).");
    Vol = 0;
    while(Vol<10 || Vol>60)
    {
    Vol=Serial.parseInt();
    }
    Serial.println("Mesure de la pression en cours.");
    Serial.println("");
    Serial.println ("Volume (mL);Pression (kPa):");
    OldState=1;
  }
  Pression = 0;
  for(int i = 0 ; i < 10 ; i++){
    ValSensor = analogRead(PinSensor);
    tension = (ValSensor/1023.0)*5.0;
    Pression = Pression + (tension / 0.02 + 10);
  }
  Pression = Pression / 10;
  if (OldPression != Pression)
  {
    Serial.print(Vol);
    Serial.print(";");
    Serial.println(Pression,1);
    OldPression = Pression;
  }
```

```
  if (Pression>PMax or Pression<PMin)
  {
    digitalWrite(PinLed,HIGH);
  }
  else {
    digitalWrite(PinLed,LOW);
  }
  delay(500);
  }
  else
  {
    if (OldState == 1){
    Serial.println("Fin des mesures.");
    Serial.println("");
    digitalWrite(PinLed,LOW);
    OldState = 0;}
  }
}
```

Remarque :

La DEL rouge est allumée si la pression est inférieure ou supérieure aux seuils de pression correspondant à un volume de 10 ou de 60 mL.

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)
. **const int PinLed = 9** (broche de la DEL)
. const int PMax = 200 (constante nombre entier valeur pression max en kPa)
. **const int PMin = 55** (constante nombre entier valeur pression min en kPa)

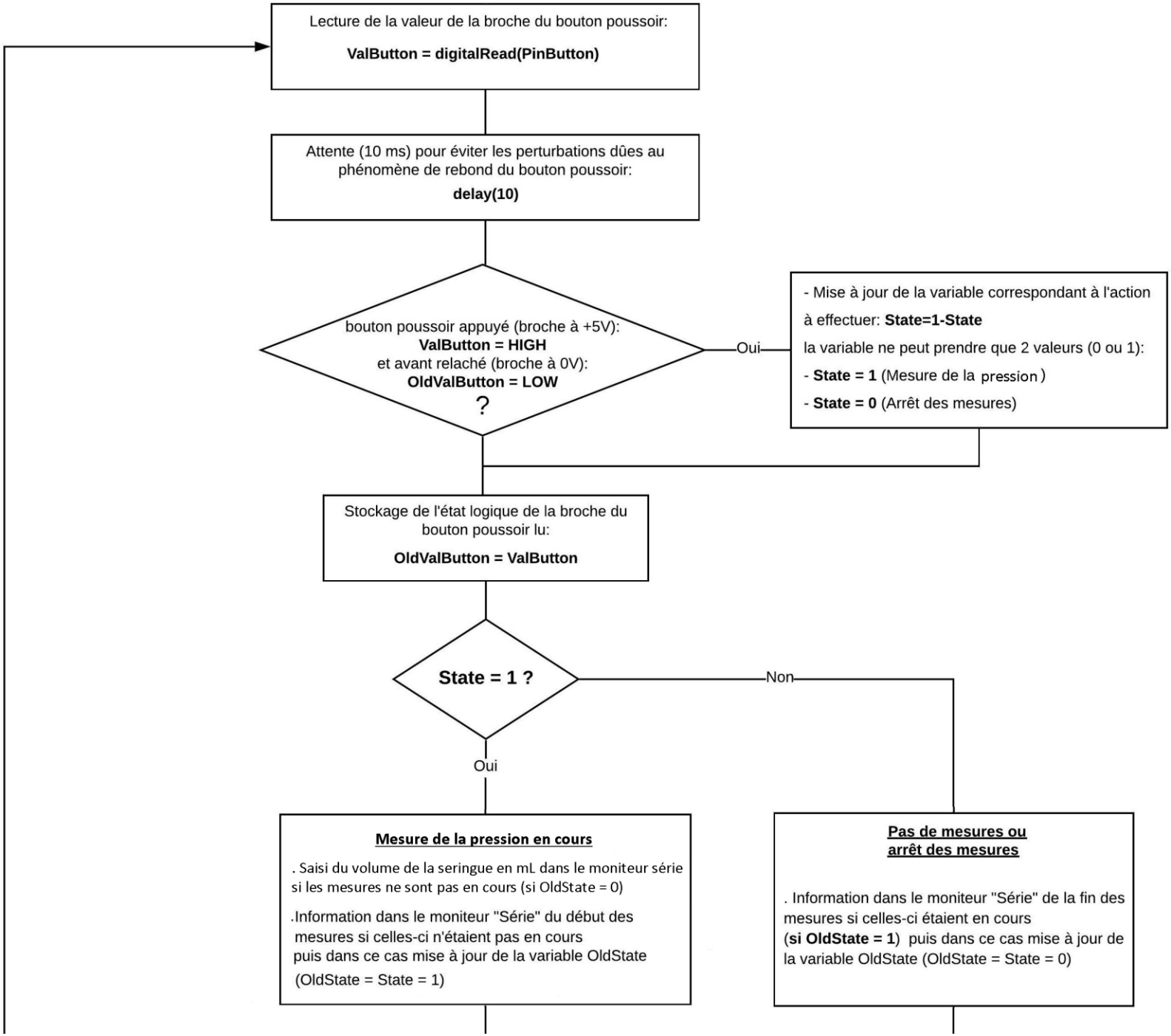
. **int ValSensor = 0** (variable nombre entier valeur broche du capteur)
. **float tension = 0.0** (variable nombre décimal calcul tension broche capteur)
. float Pression = 0.0 (variable nombre décimal calcul pression)
. float OldPression = 0.0 (variable nombre décimal ancien calcul pression)
. int Vol = 0 (variable nombre entier valeur du volume en mL)

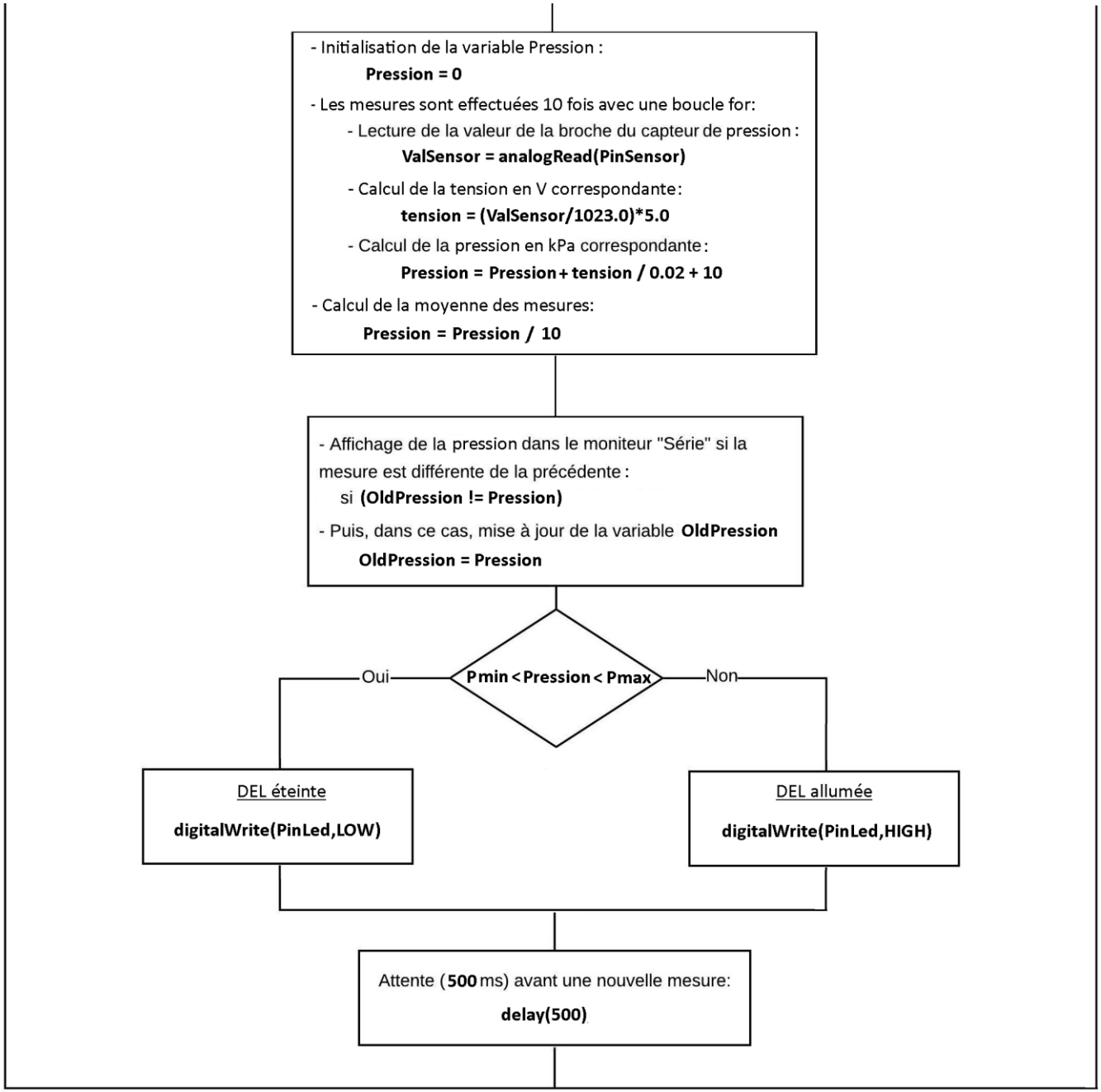
. int ValButton = 0 (variable nombre entier valeur broche bouton poussoir)
. int OldValButton = 0; (variable nbr entier ancienne valeur broche btn poussoir)
. int State = 0; (variable nombre entier pour action à effectuer)
. int OldState = 0; (variable nombre entier pour action effectuée précédemment)

- 2. Initialisation des entrées et sorties :

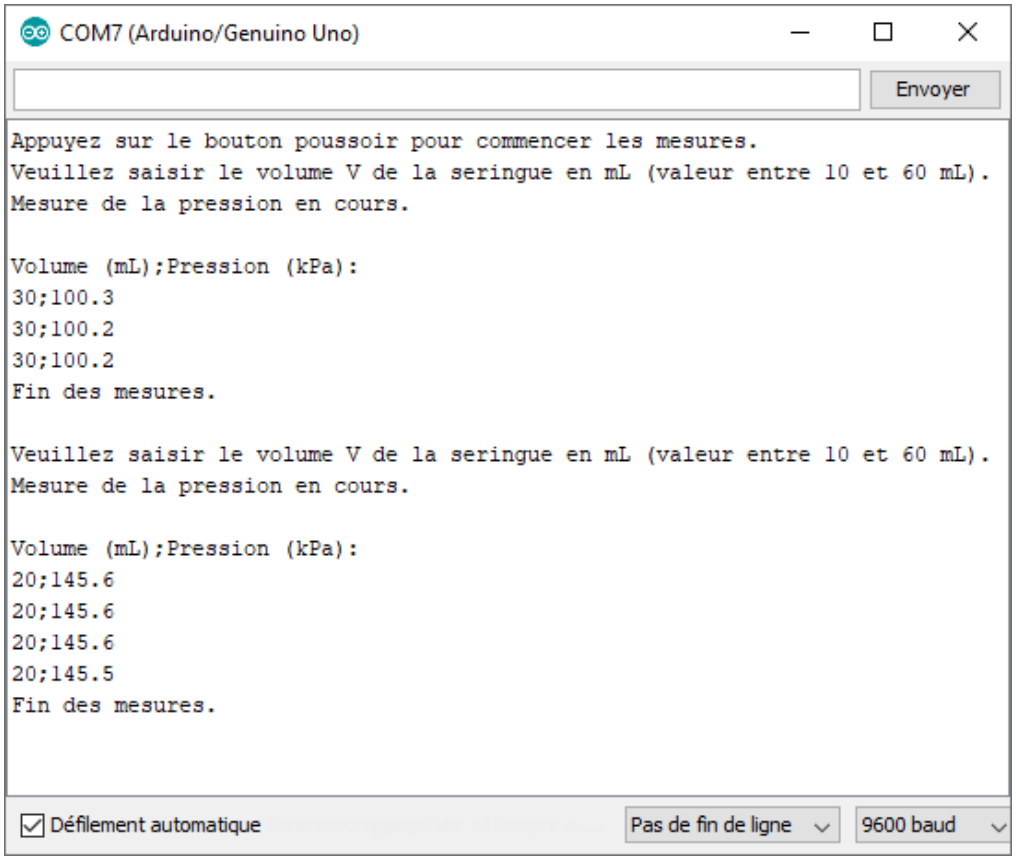
. Initialisation de la liaison série à un débit de 9600 bauds
. Initialisation de la broche du bouton poussoir en entrée
. Initialisation de la broche de la DEL en sortie

- 3. Fonction principale en boucle :


Lecture de la valeur de la broche du bouton poussoir:
ValButton = digitalRead(PinButton)
Attente (10 ms) pour éviter les perturbations dûes au phénomène de rebond du bouton poussoir:
delay(10)
bouton poussoir appuyé (broche à +5V):
ValButton = HIGH
et avant relaché (broche à 0V):
OldValButton = LOW
?
Oui
- Mise à jour de la variable correspondant à l'action à effectuer: State=1-State
la variable ne peut prendre que 2 valeurs (0 ou 1):
- State = 1 (Mesure de la pression)
- State = 0 (Arrêt des mesures)
Stockage de l'état logique de la broche du bouton poussoir lu:
OldValButton = ValButton
State = 1 ?
Non
Oui
Mesure de la pression en cours
. Saisi du volume de la seringue en mL dans le moniteur série si les mesures ne sont pas en cours (si OldState = 0)
.Information dans le moniteur "Série" du début des mesures si celles-ci n'étaient pas en cours puis dans ce cas mise à jour de la variable OldState (OldState = State = 1)
Pas de mesures ou arrêt des mesures
. Information dans le moniteur "Série" de la fin des mesures si celles-ci étaient en cours (si OldState = 1) puis dans ce cas mise à jour de la variable OldState (OldState = State = 0)

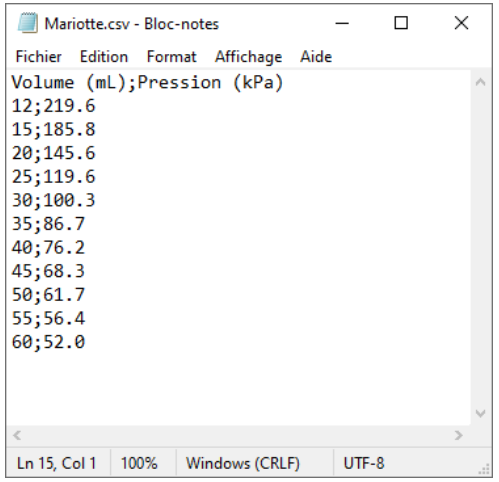

- Initialisation de la variable Pression :
Pression = 0
- Les mesures sont effectuées 10 fois avec une boucle for:
- Lecture de la valeur de la broche du capteur de pression :
ValSensor = analogRead(PinSensor)
- Calcul de la tension en V correspondante :
tension = (ValSensor/1023.0)*5.0
- Calcul de la pression en kPa correspondante :
Pression = Pression + tension / 0.02 + 10
- Calcul de la moyenne des mesures:
Pression = Pression / 10
- Affichage de la pression dans le moniteur "Série" si la mesure est différente de la précédente :
si (OldPression != Pression)
- Puis, dans ce cas, mise à jour de la variable OldPression
OldPression = Pression
Pmin < Pression < Pmax
Oui
Non
DEL éteinte
digitalWrite(PinLed,LOW)
DEL allumée
digitalWrite(PinLed,HIGH)
Attente (500 ms) avant une nouvelle mesure:
delay(500)

Résultats dans le moniteur série :

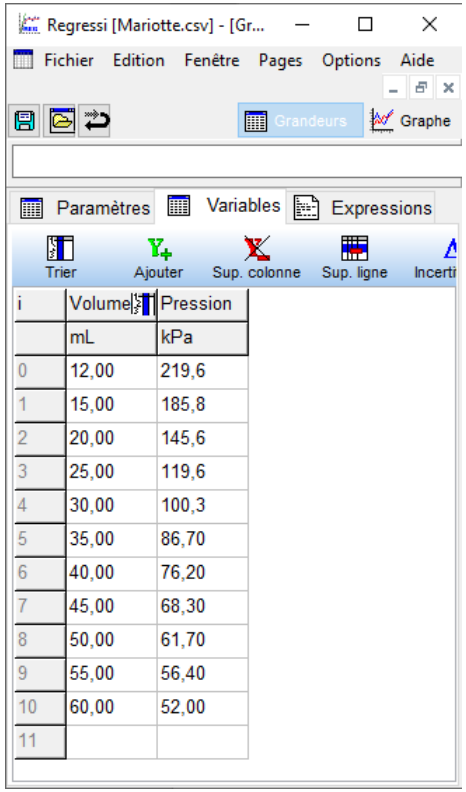


Exploitation des résultats :

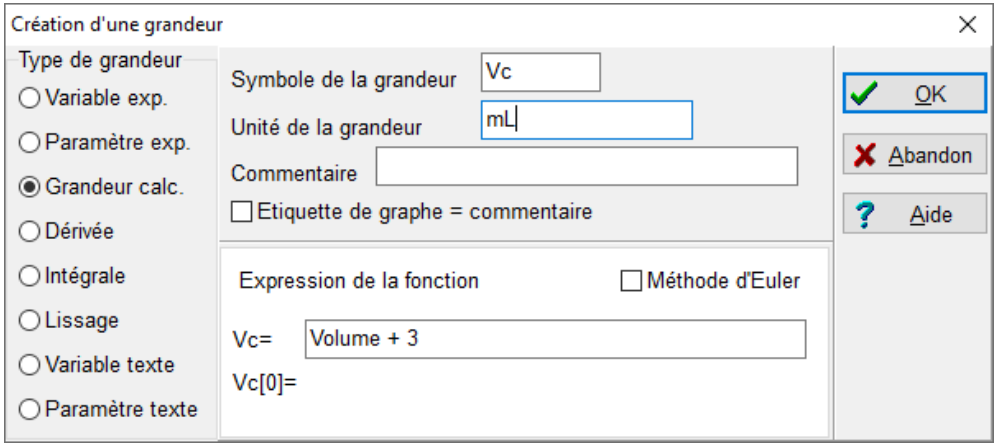
Pour exploiter les mesures, on peut dans un premier temps, copier/coller les résultats dans un fichier **"csv"** créé avec le bloc-notes de Windows, comme ci-dessous :



Puis ouvrir le fichier avec un tableur comme Regressi :



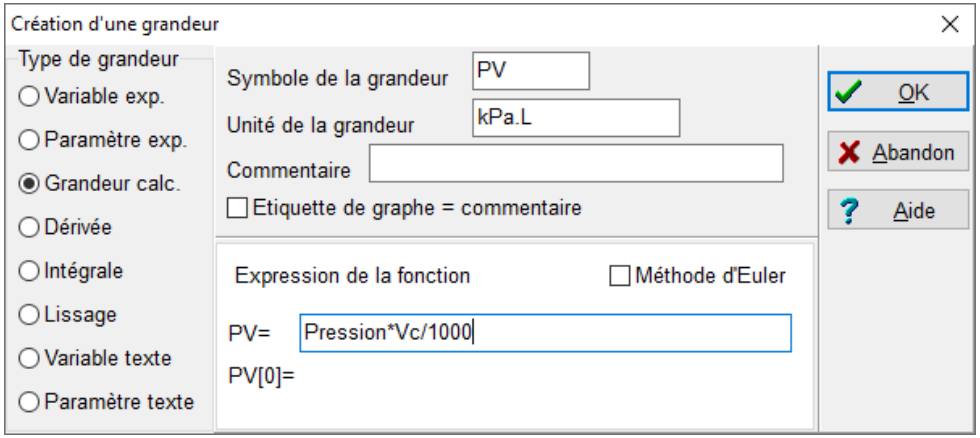
Pour les calculs, il ne faut pas oublier d'ajouter le volume d'air contenu dans le tuyau, qui relie la seringue et le capteur, au volume d'air de la seringue.
Pour cela, on va créer une grandeur, appelée **Vc** en mL, pour volume corrigé :

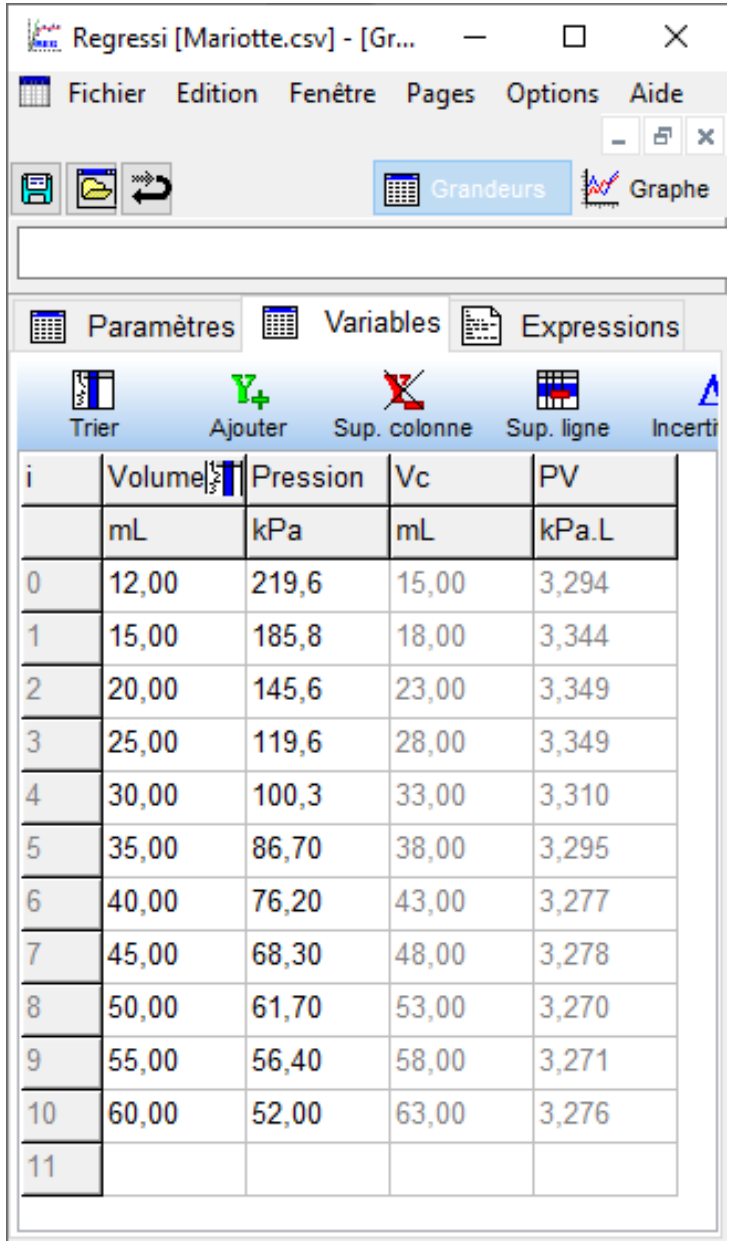


Le tuyau a un diamètre de 5 mm pour une longueur de 15 cm :

$$\mathbf{V_{air\ tuyau} \cong 3\ mL}$$

Puis on crée une grandeur, appelée **PV**, de façon à déterminer la valeur de la constante de la loi de Boyle-Mariotte (en kPa.L) :

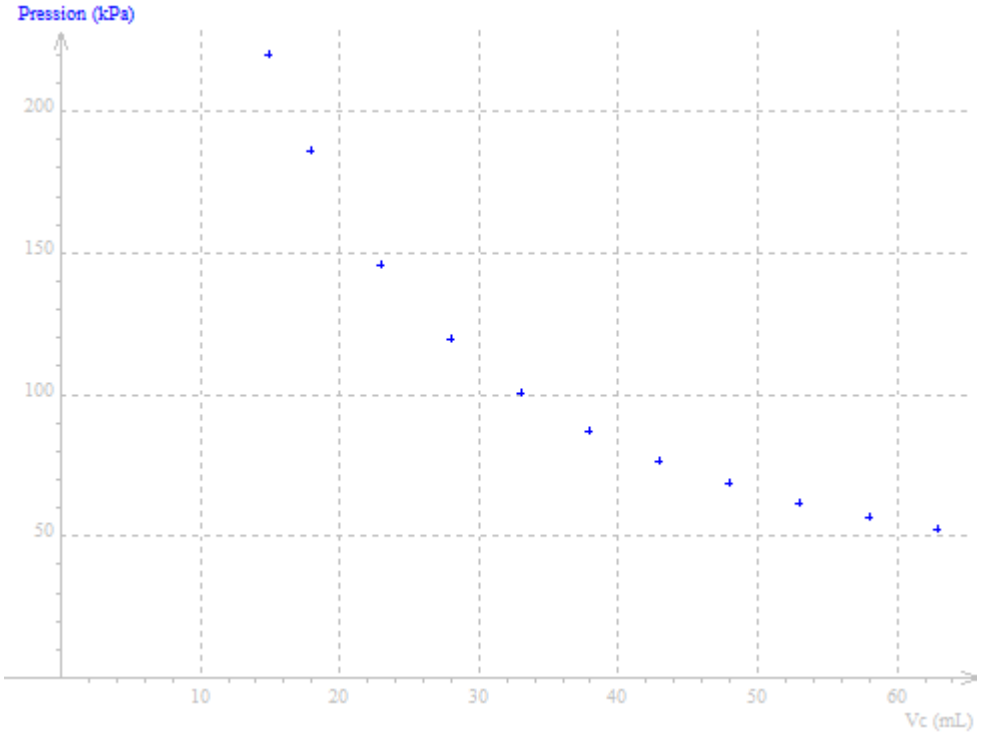




i	Volume	Pression	Vc	PV
	mL	kPa	mL	kPa.L
0	12,00	219,6	15,00	3,294
1	15,00	185,8	18,00	3,344
2	20,00	145,6	23,00	3,349
3	25,00	119,6	28,00	3,349
4	30,00	100,3	33,00	3,310
5	35,00	86,70	38,00	3,295
6	40,00	76,20	43,00	3,277
7	45,00	68,30	48,00	3,278
8	50,00	61,70	53,00	3,270
9	55,00	56,40	58,00	3,271
10	60,00	52,00	63,00	3,276
11				

La constante de la loi de Boyle-Mariotte est d'environ **3,3** kPa.L

On peut tracer le graphe représentant la pression en fonction du volume corrigé :

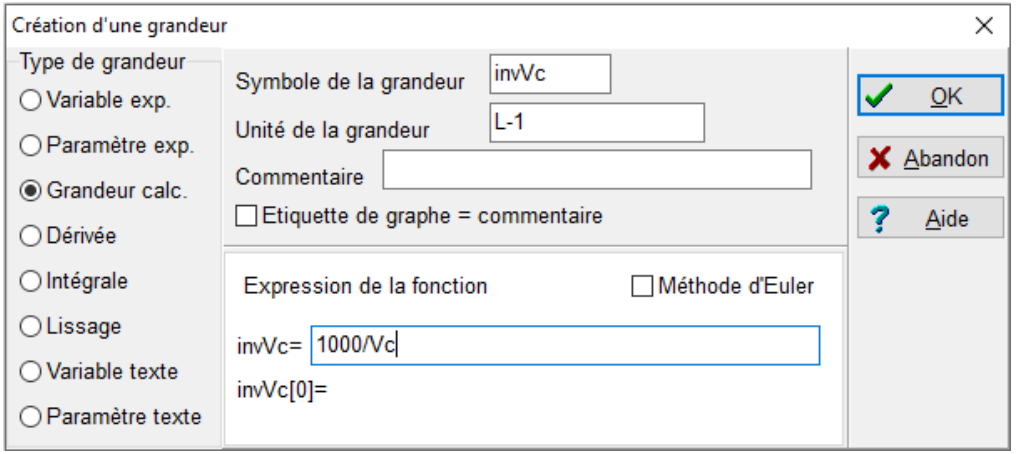


Mais comme d'après la loi de Boyle-Mariotte :

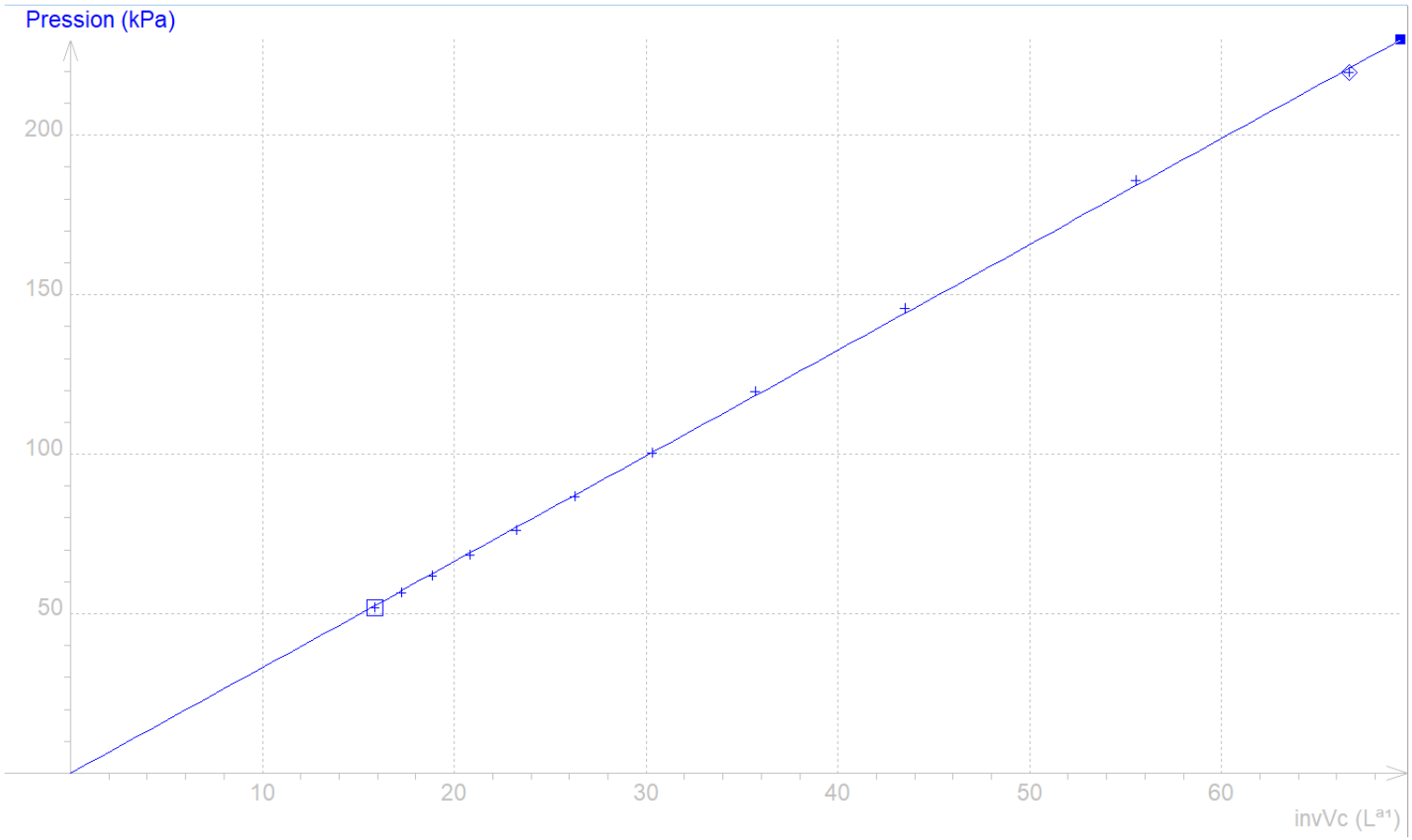
P = k / V (où k est une constante en kPa.L),

Il est préférable de tracer le graphe représentant **P = f (1/Vc)**.

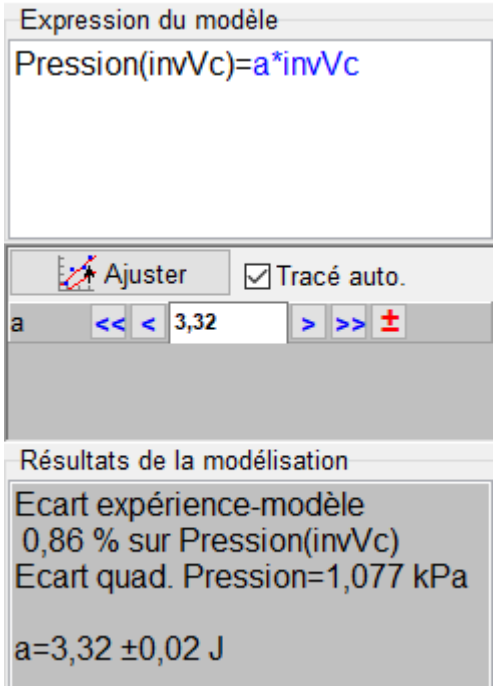
Pour cela, on crée une grandeur, appelée invVc (en L^{-1}), représentant l'inverse de Vc en L :



La représentation graphique de **P = f(1/Vc)** est alors :



La représentation graphique de **P = f(1/Vc)** peut être modélisée par une fonction linéaire :



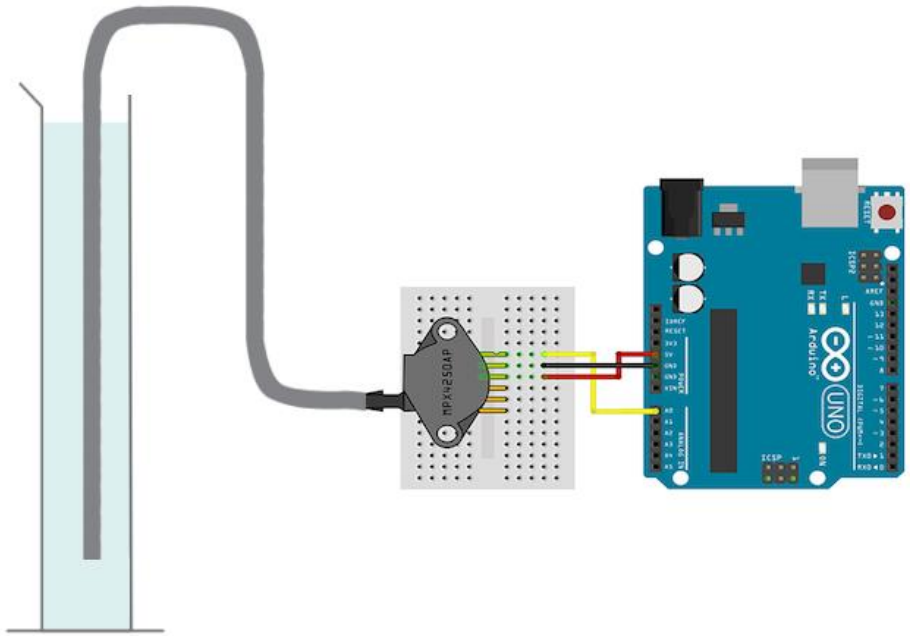
On retrouve bien la valeur de la constante de la loi de Boyle-Mariotte d'environ **3,3** kPa.L approximée précédemment.

Remarques :

- Les résultats obtenus sont compatibles avec la loi de Boyle-Mariotte
- Les incertitudes dans les mesures sont dues à :
  - . la précision du capteur,
  - . la valeur de la tension de référence de l'Arduino qui n'est pas strictement égale à 5,0 V,
  - . la précision de la mesure du volume d'air dans la seringue et le tuyau de connexion.

Activité 4 : Principe fondamental de la statique des fluides

L'objectif de cette activité est de déterminer la profondeur d'immersion dans une colonne d'eau d'un tuyau relié à un capteur de pression MPX4250AP en appliquant le principe fondamental de la statique des fluides au montage suivant :



. Énoncé du principe fondamental de la statique des fluides

La statique des fluides constitue l'étude des fluides au repos.
Les fluides se déforment sous l'effet de forces très faibles, un fluide n'a pas de forme propre.

On distingue les liquides et les gaz :

- Le liquide prend la forme du récipient qui le contient, mais il est incompressible (ρ varie peu avec P et T).
- Le gaz occupe tout le volume mis à sa disposition et il est compressible (ρ varie beaucoup avec P et T)

La pression est la même en tout point d'un même plan horizontal d'un fluide homogène au repos.

La différence de pression (en Pa) entre deux points A et B (Cf. schéma ci-dessous) d'un fluide homogène au repos est égale à :

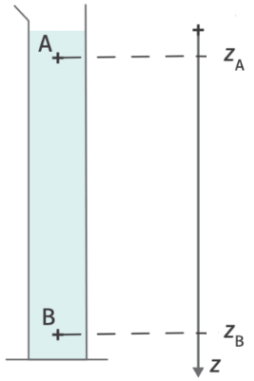
$$P_B - P_A = \rho\, g\, (Z_B - Z_A)$$

Où : . B est en-dessous de A,

. ρ = masse volumique du fluide en $kg.m^{-3}$,

. g = champ de pesanteur ($g = 9{,}81\ m.s^{-2}$),

. $Z_B - Z_A$ = distance entre les plans horizontaux passant par A et B en m.



. Descriptif de l'activité

Après avoir positionné le tuyau au niveau du point A, constituant l'origine du repère (Z_A=0), et mesuré la pression P_A de ce point, on déplacera l'extrémité du tuyau (Point B) dans la colonne d'eau. Le microcontrôleur mesurera en continu la pression du point B et calculera la distance (en m) entre les 2 points :

$$Z_B = \frac{(P_B - P_A)}{\rho g}$$

Avec ρ_{eau} = 1000 kg/m^3

On a donc :

$$Z_B = \frac{1000\ \Delta P(en\ kPa)}{9{,}81\ x\ 1000} = \frac{\Delta P(en\ kPa)}{9{,}81}$$

Remarques :

La sensibilité du capteur de pression MPX4250AP est de 20 mv/kPa. La résolution par défaut du convertisseur analogique/numérique de l'Arduino étant de l'ordre de 5 mV, il est possible de mesurer des différences de pression de 0,25 kPa, soit une différence de profondeur d'immersion (Δz) d'environ 0,025 m (2,5 cm).

La précision de la mesure de Δz (Z_B - Z_A)sera donc de ± 2,5 cm.

. Le programme

Voici le code de l'activité (**"Activity4.ino" dans le dossier "Codes/Pression"**) :

Activity4

```
// 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 PRef = 0.0;
float Dz=0.0;
float OldDz = 0.0;

int ValButton = 0;
int OldValButton = 0;
int State = 0;
int OldState = 0;
boolean BtnAppui = false;


// Initialisation des entrées et sorties

void setup() {
  Serial.begin(9600);
  pinMode(PinButton,INPUT);
  Serial.print("Placer le tuyau a la position de reference ");
  Serial.println("et appuyez sur le bouton poussoir.");
  while (BtnAppui==false){
    ValButton = digitalRead(PinButton);
    if ((ValButton == HIGH)&&(OldValButton == LOW)){
       BtnAppui=true;
    delay (200);
    }
  }
  OldValButton = ValButton;
  ValSensor = analogRead(PinSensor);
  tension = (ValSensor/1023.0)*5.0;
  PRef = (tension / 0.02) + 10;
  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 ("Pression (en kPa), Profondeur (cm):");
   OldState=1;
  }
  ValSensor = analogRead(PinSensor);
  tension = (ValSensor/1023.0)*5.0;
  Pression = (tension / 0.02) + 10;
  Dz = 100*(Pression-PRef)/9.81;

  if (abs(OldDz - Dz)>2)
  {
    Serial.print(Pression,1);
    Serial.print(" ; ");
    Serial.println(Dz,0);
    OldDz = Dz;
  }
  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)

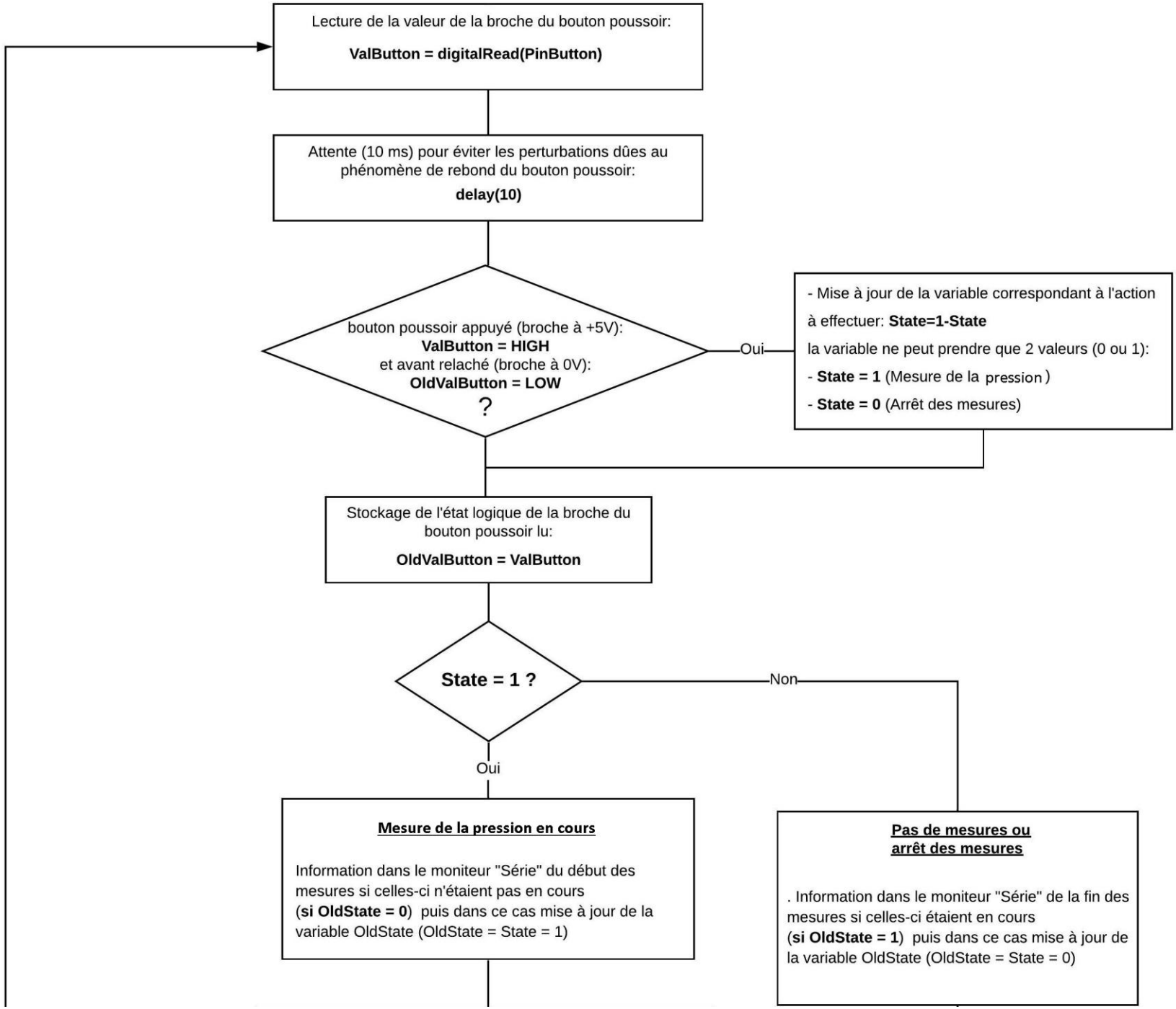
. **int ValSensor = 0** (variable nombre entier valeur broche du capteur)
. **float tension = 0.0** (variable nombre décimal calcul tension broche capteur)
. **float Pression = 0.0** (variable nombre décimal calcul pression)
. **float Pref = 0.0** (variable nombre décimal pression point référence)
. **float Dz=0.0** (variable nombre décimal profondeur en cm)
. **float OldDz = 0.0** (variable nombre décimal ancienne valeur profondeur en cm)

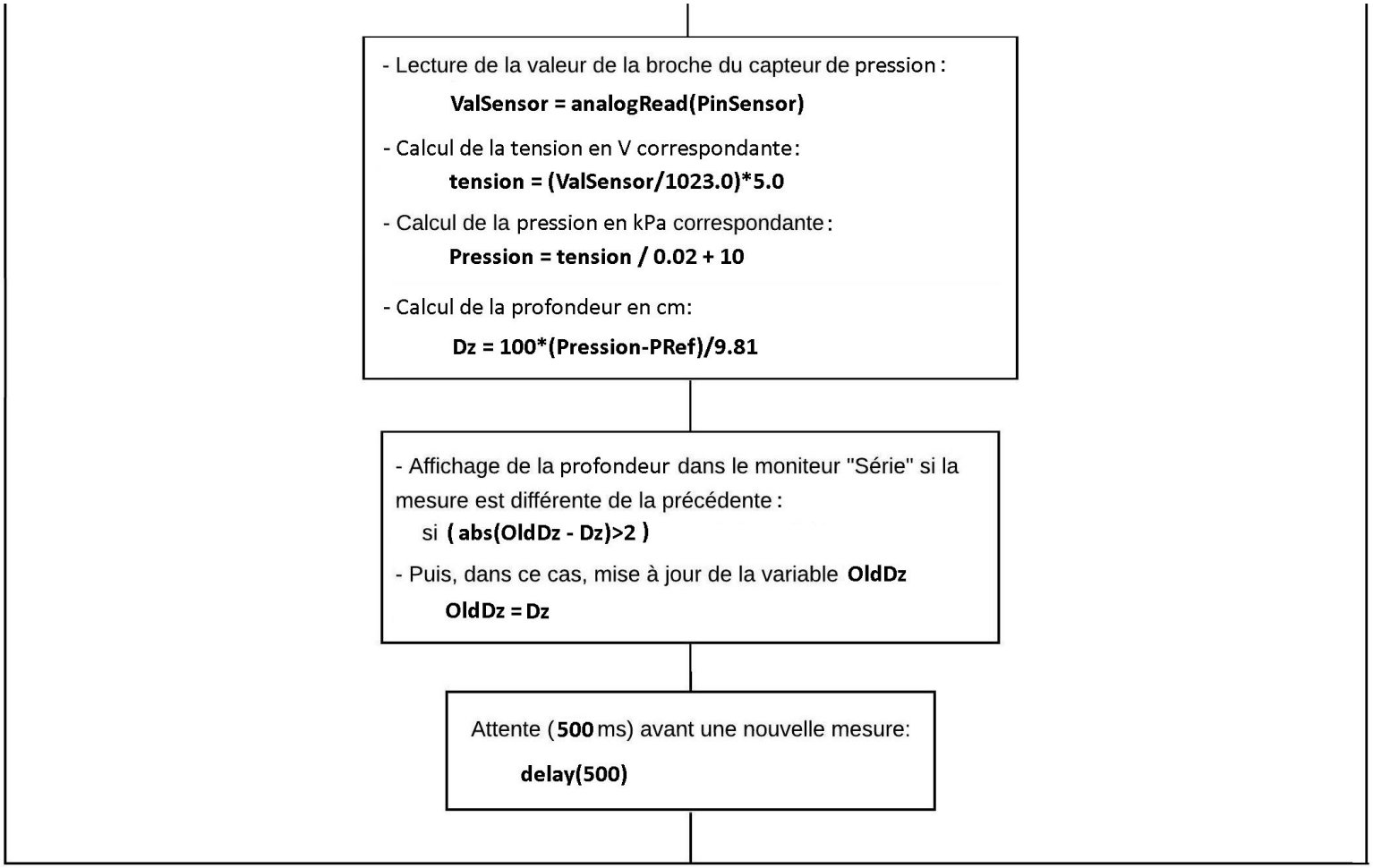
. **int ValButton = 0** (variable nombre entier valeur broche bouton poussoir)
. **int OldValButton = 0** (variable nbr entier ancienne valeur broche btn poussoir)
. **int State = 0** (variable nombre entier pour action à effectuer)
. **int OldState = 0** (variable nombre entier pour action effectuée précédemment)
. **boolean BtnAppui = false** (variable booléenne indiquant appui sur btn poussoir)

- 2. Initialisation des entrées et sorties :

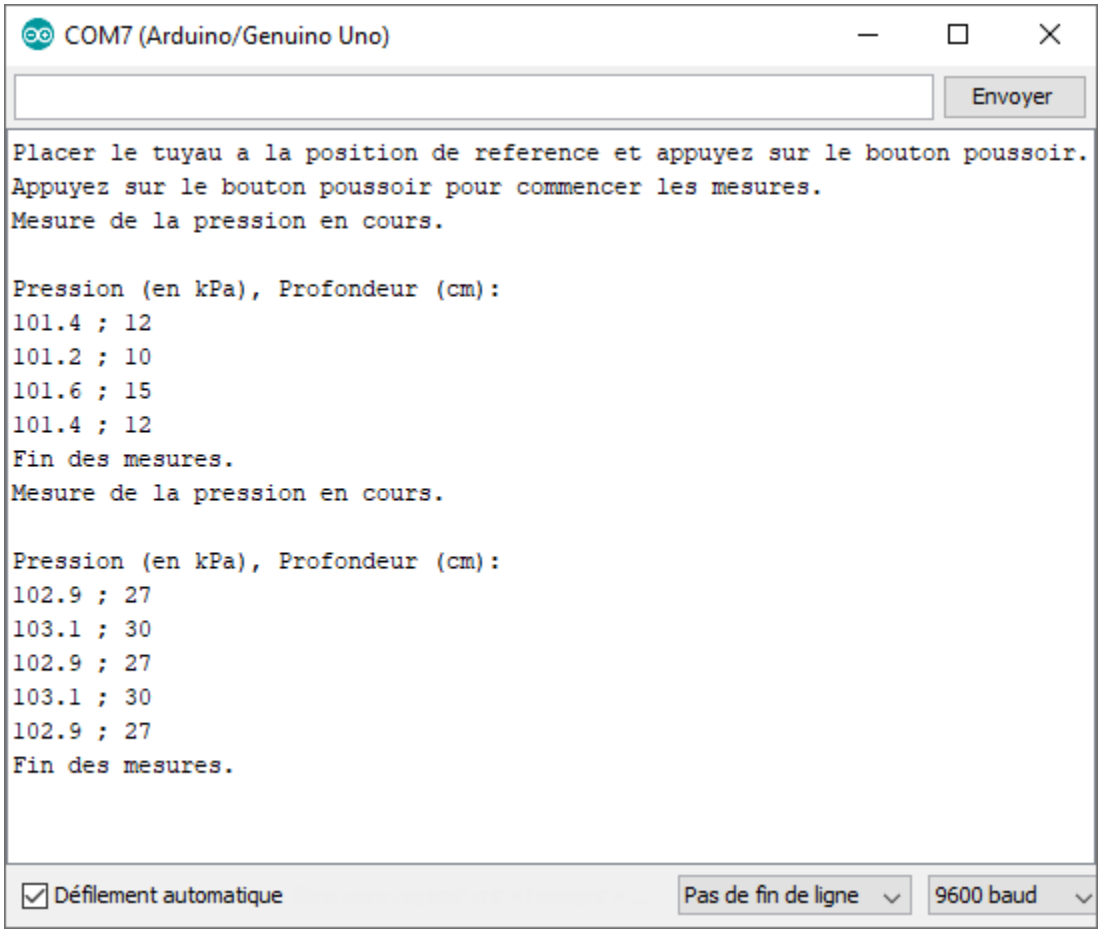
. **Initialisation de la liaison série à un débit de 9600 bauds**
. **Initialisation de la broche du bouton poussoir en entrée**
. **Mesure de la pression au point de référence (Z_A=0)**

- 3. Fonction principale en boucle


Lecture de la valeur de la broche du bouton poussoir:
ValButton = digitalRead(PinButton)
Attente (10 ms) pour éviter les perturbations dûes au phénomène de rebond du bouton poussoir:
delay(10)
bouton poussoir appuyé (broche à +5V):
ValButton = HIGH
et avant relaché (broche à 0V):
OldValButton = LOW
?
Oui
- Mise à jour de la variable correspondant à l'action à effectuer: State=1-State
la variable ne peut prendre que 2 valeurs (0 ou 1):
- State = 1 (Mesure de la pression)
- State = 0 (Arrêt des mesures)
Stockage de l'état logique de la broche du bouton poussoir lu:
OldValButton = ValButton
State = 1 ?
Non
Oui
Mesure de la pression en cours
Information dans le moniteur "Série" du début des mesures si celles-ci n'étaient pas en cours
(si OldState = 0) puis dans ce cas mise à jour de la variable OldState (OldState = State = 1)
Pas de mesures ou arrêt des mesures
. Information dans le moniteur "Série" de la fin des mesures si celles-ci étaient en cours
(si OldState = 1) puis dans ce cas mise à jour de la variable OldState (OldState = State = 0)


- Lecture de la valeur de la broche du capteur de pression :
ValSensor = analogRead(PinSensor)
- Calcul de la tension en V correspondante :
tension = (ValSensor/1023.0)*5.0
- Calcul de la pression en kPa correspondante :
Pression = tension / 0.02 + 10
- Calcul de la profondeur en cm:
Dz = 100*(Pression-PRef)/9.81
- Affichage de la profondeur dans le moniteur "Série" si la mesure est différente de la précédente :
si (abs(OldDz - Dz)>2)
- Puis, dans ce cas, mise à jour de la variable OldDz
OldDz = Dz
Attente (500 ms) avant une nouvelle mesure:
delay(500)

Résultats dans le moniteur série :



Pour la première mesure, la profondeur réelle par rapport à l'origine du repère était de 12 cm.
Et pour la deuxième, elle était de 30 cm.

Les mesures ne sont donc pas très précises mais l'ordre de grandeur est respecté.