

Nom : .....  
 Prénom : .....  
 Classe : .....  
 Date : .....

**Activité 5 – Détecteur de niveau d'un liquide dans une cuve**

→ **Comment tester le principe du fonctionnement d'un détecteur capacitif de niveau de liquide ?**

**1. Réaliser**

a. Mettre en œuvre le protocole expérimental mis à disposition par le professeur permettant de représenter et de modéliser le volume  $V$ , proportionnel à la hauteur de liquide dans l'éprouvette graduée, en fonction du temps caractéristique  $\tau$  du dipôle RC, constitué de la résistance  $R$  et de la sonde de capacité  $C$ .

**Protocole :**

- Réaliser le montage complet du **DOC. 1** correspondant au montage électrique du **DOC. 3** permettant de mesurer le temps caractéristique du capteur capacitif avec un microcontrôleur ;
- Régler la position du capteur capacitif verticalement en laissant à l'éprouvette graduée la possibilité de glisser horizontalement par rapport au capteur : celui-ci ne doit pas bloquer le fond de l'éprouvette car le liquide doit pouvoir pénétrer à l'intérieur ;
- Ouvrir l'IDE Arduino, charger le code source **ch21\_486\_1b.ino** puis le téléverser dans le microcontrôleur ;
- Ouvrir le moniteur série sur l'ordinateur : toutes les 10 s environ, 20 valeurs de temps caractéristiques ainsi que la moyenne (nulle pour l'instant) s'affichent sur l'écran de l'ordinateur ;
- Ajouter un volume d'éthanol absolu compris entre 4 et 8 mL dans l'éprouvette graduée de 50 mL, attendre que la valeur moyenne du temps caractéristique se stabilise puis la relever.
- Ajouter à nouveau de l'éthanol pour réaliser une dizaine de mesures entre 0 et 50,0 mL et remplir le tableau suivant :

<b>V (en mL)</b>	0										50,0
<b><math>\tau</math> (en <math>\mu</math>s)</b>	0										

- Saisir les mesures dans un tableur grapheur incluant des fonctions de modélisation ;
- Choisir un modèle et modéliser la courbe d'étalonnage  **$V = f(\tau)$** ;
- Relever l'expression du modèle :

$$V = \dots \times \tau + \dots \times \tau^2, \text{ avec } V \text{ en mL et } \tau \text{ en } \mu\text{s}.$$

b. En utilisant les résultats de la modélisation de  $V = f(\tau)$ , modifier le code source du microcontrôleur afin d'obtenir la mesure du volume  $V$  de liquide dans le récipient sur l'écran de l'ordinateur.

Le listing du code source **1. ch21\_486\_1b.ino** est le suivant :

Nom : .....  
 Prénom : .....  
 Classe : .....  
 Date : .....

```

1 long i=0; // variable pour la durée de la charge en us (10-6 s).
2 int k=100; // variable pour la durée de charge.
3 long somi=0; // variable pour la moyenne des temps tau.
4 int n=0; // nombre de mesures par tau moyen.
5
6 void setup() {
7     pinMode(11,OUTPUT); // broche 11 pour la charge et la décharge.
8     Serial.begin(9600); // vitesse des données dans le port série.
9 }
10
11 void loop() {
12     digitalWrite(11,LOW); // décharge du capteur capacitif.
13     while(analogRead(A0)>1){ /*le capteur se décharge...*/ }
14     digitalWrite(11,HIGH); // charge du capteur capacitif.
15     delayMicroseconds(i); // durée de la charge du capteur.
16     if(analogRead(A0)<647){ // test uc<0,632*E ?
17         i=i+k; // si oui alors on augmente la durée de charge de k us.
18     }
19     else if(k==100){ // si non alors on diminue la durée de charge de
20         i=i-k; // k=100 us pour approcher la durée de charge
21         k=10; // à k=10 us près.
22     }
23     else if(k==10){ // si uc>=0,632*E et si k=10 us alors on diminue la
24         i=i-10; // durée de charge de k=10 us pour approcher la durée
25         k=1; // de charge à la microseconde près.
26         if(i<0)i=0; // tau ne peut pas être négatif, au minimum nul.
27     }
28
29     else{ // la mesure de tau de n°n est stockée.
30         n++; // on passe à la mesure suivante n°n+1.
31         somi=somi+i; // on fait la somme des mesures de tau.
32         Serial.print(i); Serial.print("/"); // affiche les valeurs mesurées.
33         i=0; k=100;
34         if(n==20){ // affichage de la moyenne des temps caractéristiques.
35             Serial.println(); Serial.print("Moyenne des 20 valeurs : tmoy=");
36             Serial.print(somi/n); Serial.println(" us");
37             /* ..... Ecrire ici les lignes .....
38              * ..... de code pour afficher .....
39              * ..... le volume mesuré, calculé .....
40              * ..... à partir du modèle obtenu .....
41              * ..... avec le tableur grapheur .....*/
42             somi=0; n=0; // réinitialisation des variables.
43             delay(2000); // pause de 2s pour la lecture.
44         }
45     }
46 }

```

**2. Valider**

- a. Mettre un volume V de liquide dans l'éprouvette et, par une approche statistique de mesures effectuées par la sonde :
  - évaluer l'incertitude-type  $u(\bar{V})$ .
  - écrire, avec un nombre adapté de chiffres significatifs, le résultat de la mesure  $V_{mes}$  du volume de liquide.

Nom : .....  
Prénom : .....  
Classe : .....  
Date : .....

b. Comparer et discuter le résultat de la mesure  $V_{\text{mes}}$  effectuée par la sonde à la valeur de référence  $V_{\text{ref}}$  lue à l'aide des graduations de l'éprouvette en utilisant le quotient  $|V_{\text{mes}} - V_{\text{ref}}| / u(\bar{V})$ .

Partie utilisant un traitement statistique des mesures avec le microcontrôleur de type Arduino, à supprimer si on ne souhaite utiliser cette méthode.

Le listing du code source **1. ch21\_486\_2a.ino** est le suivant :

Nom : .....  
 Prénom : .....  
 Classe : .....  
 Date : .....

```

1 #include "Statistic.h" // importe la bibliothèque Statistic.
2 Statistic stat;      // créer un objet stat pour faire des statistiques.
3
4 long i=0;    // variable pour la durée de la charge en us (10-6 s).
5 int k=100;   // variable pour la durée de charge.
6 int n=0;    // nombre de mesures par tau moyen.
7
8 void setup() {
9     pinMode(11,OUTPUT); // broche 11 pour la charge et la décharge.
10    Serial.begin(9600); // vitesse des données dans le port série.
11    stat.clear();      // réinitialise les données statistiques
12 }
13
14 void loop() {
15     digitalWrite(11,LOW); // décharge du capteur capacitif.
16     while(analogRead(A0)>1){ /*le capteur se décharge...*/ }
17     digitalWrite(11,HIGH); // charge du capteur capacitif.
18     delayMicroseconds(i); // durée de la charge du capteur.
19     if(analogRead(A0)<647){ // test uc<0,632*E ?
20         i=i+k;          // si oui alors on augmente la durée de charge de k us.
21     }
22     else if(k==100){ // si non alors on diminue la durée de charge de
23         i=i-k;        // k=100 us pour approcher la durée de charge
24         k=10;         // à k=10 us près.
25     }
26     else if(k==10){ // si uc>=0,632*E et si k=10 us alors on diminue la
27         i=i-10;       // durée de charge de k=10 us pour approcher la durée
28         k=1;          // de charge à la microseconde près.
29         if(i<0)i=0;   // tau ne peut pas être négatif, au minimum nul.
30     }
31     else{             // la mesure de tau de n°n est stockée.
32         n++;          // on passe à la mesure suivante n°n+1.
33         stat.add(i);  // ajoute la valeur dans les données statistiques.
34         Serial.print(i); Serial.print("/"); // affiche les valeurs mesurées.
35         i=0; k=100;
36         if(n==20){ // affichage de la moyenne des temps caractéristiques.
37             Serial.println();
38             Serial.println("Toutes les mesures sont exprimées en us");
39             Serial.println("tmin\t|   tmax\t|   tmoyen\t|   u(tmoyen)");
40             Serial.print(stat.minimum()); Serial.print("\t|   "); // minimum.
41
42             Serial.print(stat.maximum()); Serial.print("\t|   "); // maximum.
43             Serial.print(stat.average()); Serial.print("\t|   "); // moyenne.
44             Serial.println(stat.pop_stddev()/sqrt(20)); // incertitude-type.
45             stat.clear(); // réinitialise les données statistiques.
46             n=0; // réinitialisation des variables.
47             delay(2000); // pause de 2s pour la lecture.
48         }
49     }
50 }

```