

DÉFI

TESTEZ VOTRE CARTE

Réussir à faire clignoter la carte pour vérifier que tout marche !

MATÉRIEL NÉCESSAIRE

- une carte Arduino
- un câble USB
- un ordinateur
(nécessaire pour tous les défis)



FICHES ASSOCIÉES

carte Arduino
logiciel Arduino IDE



FACILE

INSTALLEZ LE LOGICIEL ARDUINO

Il faut un logiciel qui permette le dialogue entre votre ordinateur et votre carte Arduino. Pour cela, allez sur le site de référence www.arduino.cc, et cliquez sur l'onglet « Software ». Le programme Arduino IDE est libre et gratuit, installez-le en choisissant l'installateur qui correspond à votre ordinateur.

BRANCHEZ VOTRE CARTE ARDUINO

Une fois l'installation terminée, utilisez le câble USB pour brancher votre carte Arduino à votre ordinateur. La carte doit être reconnue par l'ordinateur. Si c'est le cas, le logiciel Arduino IDE doit détecter votre carte. Pour le vérifier, ouvrez ce logiciel et déroulez le menu Outils. Il y a deux paramètres qu'il faut vérifier dans ce menu : le type de carte et le port. Le paramètre « type de carte » doit correspondre à la carte que vous utilisez (Uno). Le paramètre « port » doit correspondre au port de votre ordinateur sur lequel est connectée la carte (par exemple « COM 11 : Arduino/Genuino Uno » si vous utilisez Windows, « \dev\tty.usbmodem ... (Arduino Uno) » sur un ordinateur Apple). Si ces deux paramètres ne sont pas correctement configurés, la communication ne pourra pas s'établir.

ENVOYEZ UN PROGRAMME SUR LA CARTE

Les programmes sont écrits sur l'ordinateur, puis envoyés sur la carte Arduino. Pour vérifier la connexion avec votre carte, vous allez envoyer un programme de test. Ouvrez le menu Fichier, choisissez Exemples, Basics, et le programme « Blink ».

DÉFI – TESTEZ VOTRE CARTE

```
// the setup function runs once when you press reset or power the board
void setup() {

  // initialize digital pin LED_BUILTIN as an output.
  pinMode(LED_BUILTIN, OUTPUT);
}

// the loop function runs over and over again forever
void loop() {
  digitalWrite(LED_BUILTIN, HIGH); // turn the LED on (HIGH is the voltage level)
  delay(1000); // wait for a second
  digitalWrite(LED_BUILTIN, LOW); // turn the LED off by making the voltage LOW
  delay(1000); // wait for a second
}
```

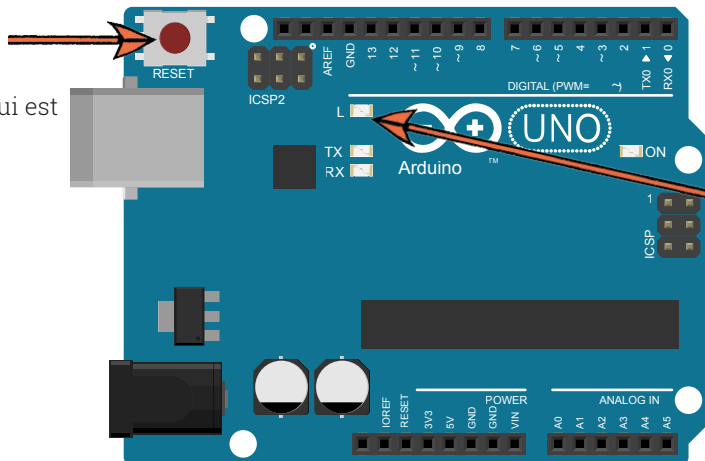
TÉLÉVERSEZ



Envoyez ce programme sur la carte Arduino en cliquant sur l'icône « Téléverser » du logiciel. Votre ordinateur va transformer le programme en instructions compréhensibles par votre carte et lui envoyer par le câble USB.

Une fois installé sur votre carte (cela peut prendre quelques secondes), ce programme va s'exécuter en boucle.

Bouton reset de la carte. Il réinitialise la carte et relance le programme qui est installé dessus.



Petite diode lumineuse (LED) de test. Cette LED est reliée au port numérique 13 sur la plupart des cartes Arduino.

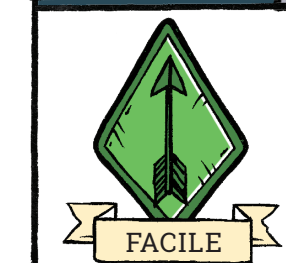
Ce programme fait clignoter la LED de test. Si la LED de votre carte clignote : bravo ! Tout va bien, vous avez réussi à communiquer avec votre carte.

EN CAS DE PROBLÈME

- Vérifiez que vous avez bien configuré le type de carte Arduino et le port COM utilisé par le logiciel (menu Outils) ;
- Débranchez votre carte, arrêtez le logiciel Arduino, puis rebranchez et relancez le programme. Si ça ne fonctionne toujours pas, éteignez votre ordinateur puis relancez tout.



DÉFI



FACILE

ALLUMEZ UNE LED

Vous allez contrôler la lumière !

MATÉRIEL NÉCESSAIRE

- une LED rouge
- une résistance de 220 ohms
- un breadboard
- des petits fils électriques



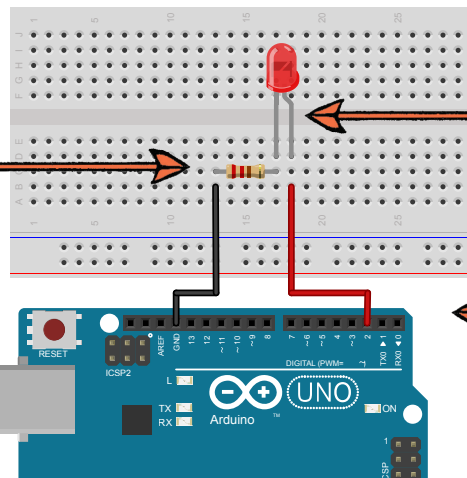
FICHES ASSOCIÉES

testez votre carte
sorties numériques

RÉALISEZ CE MONTAGE ÉLECTRIQUE

Enfoncez bien les différents composants et fils dans le breadboard pour avoir un bon contact électrique.

Cette résistance évite d'envoyer trop de courant dans la LED et de la cramer.



Attention au sens pour brancher la LED : le fil le plus long est connecté avec le fil rouge (vers la tension +5 V).

Les connecteurs numérotés de 0 à 13 sont des ports numériques : vous pouvez imposer une tension de 5 V ou l'annuler sur chacun des ports.

RECOPIEZ CE PROGRAMME

HIGH : ça veut dire qu'on envoie 5 V dans le fil branché sur le port 2.

LOW : ça veut dire qu'on envoie zéro volt.

```

// setup pour initialiser la carte
void setup() { // début du setup
  pinMode(2, OUTPUT) ; // initialise le port numérique 2 comme sortie
} // fin du setup

// cette boucle va se répéter sans arrêt
void loop() { // début de la boucle
  digitalWrite(2, HIGH) ; // allume la LED
  delay(1000) ; // attend 1 seconde (soit 1000 msec)
  digitalWrite(2, LOW) ; // éteint la LED
  delay(1000) ; // attend 1 seconde (soit 1000 msec)
} // fin de la boucle

```

Le texte en gris est un texte de commentaire, ignoré par l'ordinateur. Ce n'est pas nécessaire de le recopier mais il permet de mieux comprendre le programme.

Un programme similaire est facilement accessible à travers le menu du logiciel (menu Fichier, Exemples, Basics, programme Blink).

DÉFI – ALLUMEZ UNE LED

TÉLÉVERSEZ



S'il ne se passe rien, c'est qu'il y a un problème ! Vérifiez votre montage électrique et la connexion entre votre ordinateur et la carte Arduino.

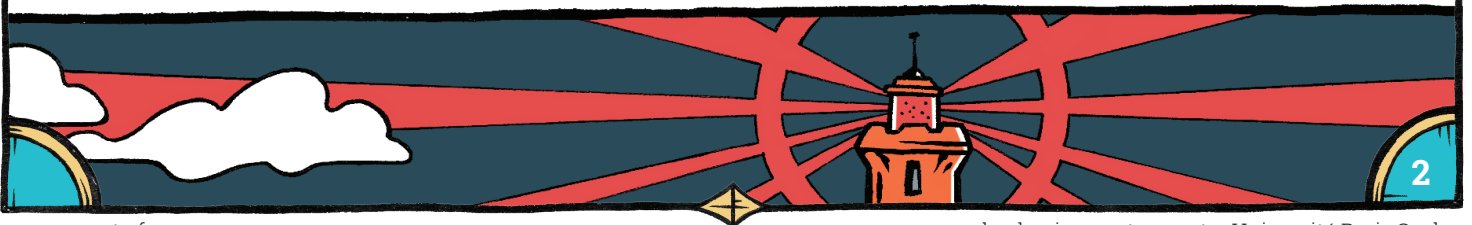
POUR ALLER PLUS LOIN

Essayez de changer la vitesse du clignotement. Pour cela, modifiez dans votre programme les valeurs des instructions `delay()`, et observez ce qui se passe quand vous téléversez votre nouveau programme. Essayez d'allumer une autre LED branchée cette fois sur le port 3.



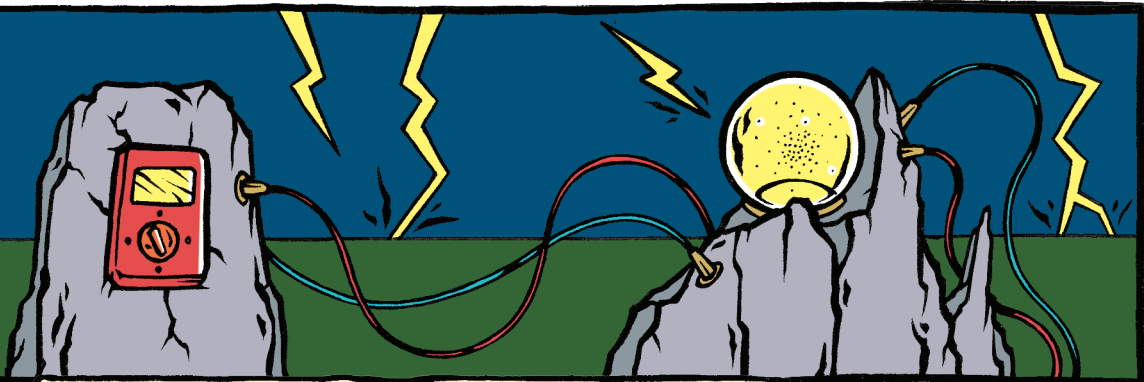
L'ULTIME DÉFI !

Modifiez votre programme pour que votre lumière lance un SOS en morse !





DÉFI



MESUREZ UNE TENSION

Vous allez mesurer une tension avec votre carte !

MATÉRIEL NÉCESSAIRE

- un potentiomètre de 10 000 ohms
- un breadboard
- des petits fils électriques
- éventuellement un multimètre



FICHES ASSOCIÉES

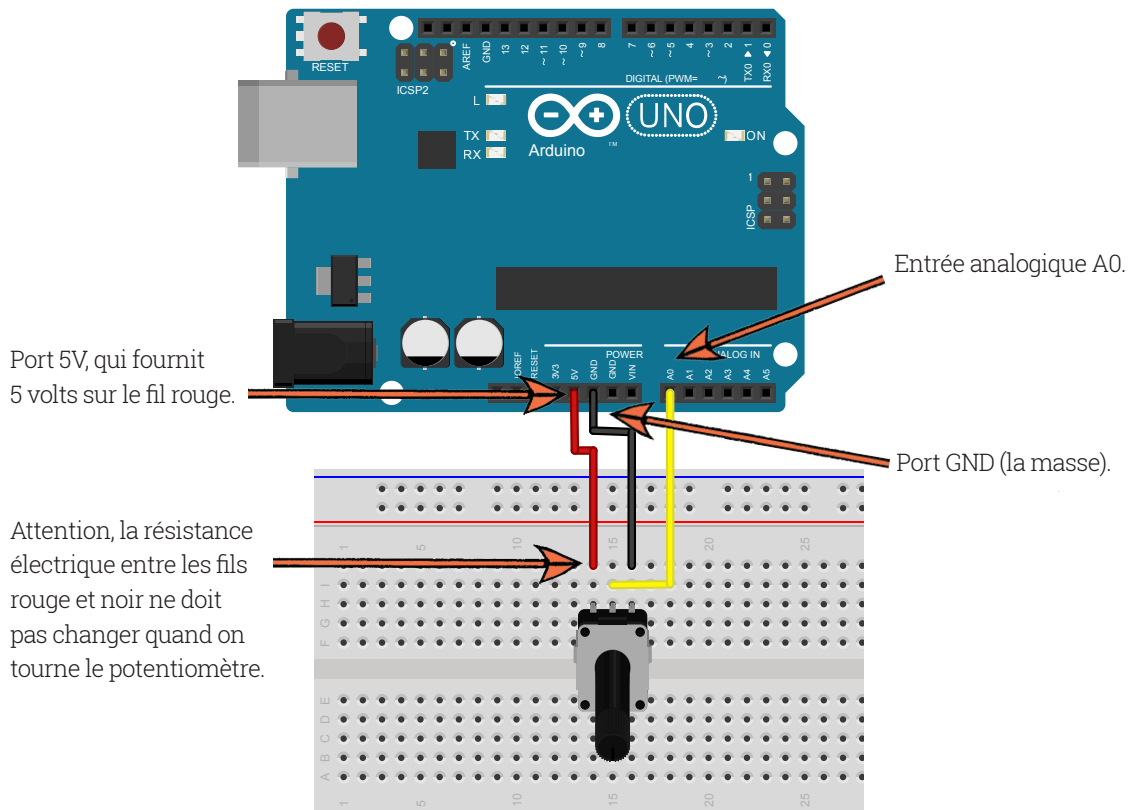
testez votre carte
entrées analogiques



FACILE

RÉALISEZ CE MONTAGE ÉLECTRIQUE

Un potentiomètre possède trois contacts. La résistance électrique entre deux de ces contacts ne varie pas quand on tourne le potentiomètre : ce sont ces contacts qui doivent être reliés au port 5V et au port GND. (Si vous avez un multimètre, vous pouvez vérifier où se trouvent ces contacts, sinon faites confiance au montage ci-dessous.)



DÉFI – MESUREZ UNE TENSION

RECOPIEZ CE PROGRAMME

```
int MesureTension ;           // on définit la variable MesureTension
                               // cette variable peut stocker des valeurs

// setup pour initialiser la carte
void setup() {                 // début du setup
  Serial.begin(9600) ;         // initialise la communication série avec l'ordinateur
}                               // fin du setup

// cette boucle va se répéter sans arrêt
void loop() {                 // début de la boucle
  MesureTension = analogRead(A0) ; // on mesure la tension sur le port A0
                               // et on l'attribue à la variable MesureTension
  Serial.println(MesureTension) // on envoie la valeur mesurée à l'ordinateur
  delay(100) ;                // attend 0.1 seconde (100 millisecondes)
}                               // fin de la boucle
```

Une variable permet de stocker une valeur dans un programme. Il faut les déclarer avant de les utiliser ; int est le type de la variable (un entier), MesureTension est le nom de la variable.

Petit temps d'attente entre deux mesures consécutives.

C'est cette instruction qui déclenche la mesure de la tension sur le port A0.

Un programme similaire est facilement accessible à travers le menu du logiciel (menu Fichier, Exemples, Basics, programme AnalogReadSerial).

TÉLÉVERSEZ

La carte Arduino va mesurer la tension sur le port A0 puis envoyer le résultat vers l'ordinateur. Pour les lire en direct depuis l'ordinateur, utilisez le Moniteur série (dans le menu Outils du logiciel Arduino). Si des caractères bizarres apparaissent, vérifiez la vitesse de connexion du Moniteur série, qui doit être la même que celle utilisée pour initialiser la communication dans le programme (ici 9600 bauds).

Tournez le potentiomètre, et regardez ce qui se passe sur le Moniteur série !

POUR ALLER PLUS LOIN

Branchez le fil rouge sur le port 3V3, et observez ce qui se passe (ce port délivre 3,3 volts). Trouvez la relation qui permet de traduire en volts ce qui est affiché dans le Moniteur série.



L'ULTIME DÉFI !

Modifiez votre programme pour que la LED de test sur la carte Arduino s'allume quand la tension mesurée dépasse 3,3 V (il faudra utiliser l'instruction « if... else... »).

TÉLÉVERSEZ



S'il ne se passe rien, c'est qu'il y a un problème ! Vérifiez votre montage électrique et la connexion entre votre ordinateur et la carte Arduino.

POUR ALLER PLUS LOIN

Modifier le programme du défi « allumer une led » de façon à allumer, alternativement toutes les secondes, la led rouge puis verte.



L'ULTIME DÉFI !

Modifiez votre programme pour que votre lumière lance un SOS en morse !



DÉFI



CRÉER UN BUZZER LUMINEUX

MATÉRIEL NÉCESSAIRE

- une LED rouge et une LED verte
- une résistance de 220 ohms
- un breadboard
- des petits fils électriques



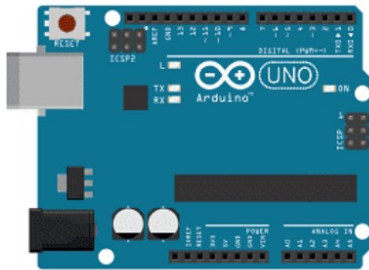
FICHES ASSOCIÉES

testez votre carte sorties numériques

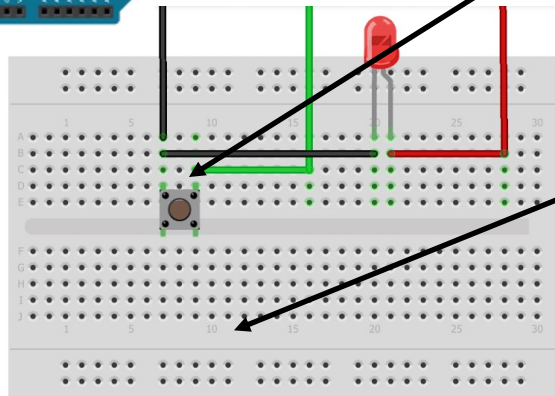


FACILE

RÉALISEZ CE MONTAGE ÉLECTRIQUE



Réaliser la connexion vers la carte arduino



Si on appuie sur le bouton poussoir, la led rouge s'allume (qui symbolise une réponse négative pour l'élève)

Refaire un autre circuit pour une led verte (qui symbolise une réponse positive pour l'élève)

Créer un programme

En vous inspirant des programmes ci-contre, programmer la carte qui permet d'allumer la led verte ou rouge en appuyant sur le l'un des deux buzzer.

```

// setup pour initialiser la carte

void setup() {
  pinMode(2, OUTPUT); // début du setup // initialise le port numérique 2 comme sortie
} // fin du setup

// cette boucle va se répéter sans arrêt
void loop() {
  digitalWrite(2, HIGH); // début de la boucle // allume la LED
  delay(1000); // attend 1 seconde (soit 1000 msec)
  digitalWrite(2, LOW); // éteint la LED
  delay(1000); // attend 1 seconde (soit 1000 msec)
} // fin de la boucle

if (water > 100) { digitalWrite(12, HIGH); digitalWrite(10, LOW); }
else { digitalWrite(12, LOW); digitalWrite(10, HIGH); }

```


Travail Pratique de PHYSIQUE Première Générale

TP CH12 Modifier des signaux électriques

NOM :

Prénom :

ÉVALUATION				
Compétences	Niveaux validés			
	A	B	C	D
s'APProprier				
ANALyser				
RÉALiser				
Note :		/20		

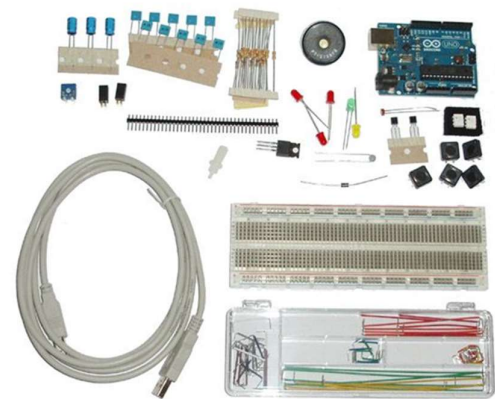
OBJECTIFS DU SUJET

Étudier la réponse d'un dispositif modélisé par un dipôle RC.

Modifier la réponse d'un dipôle RC à l'aide d'une carte d'acquisition arduino et en mesurer les effets. .

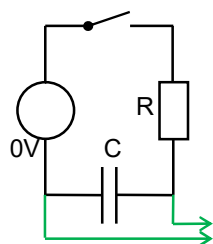
Pour chaque poste

- 1 carte Arduino Uno + 1 câble USB
- 1 boîte de connexions 840 points
- ponts de connexions
- jumpers
- 3 résistances 10 kΩ 1/4W
- 3 résistances 2,2 kΩ 1/4W
- 3 résistances 220 kΩ 1/4W
- 3 résistances 330 kΩ 1/4W
- 2 condensateurs 100 μF/25 Vcc
- 1 bouton-poussoir 12x12mm
- Un ordinateur muni d'un logiciel UNO et d'un tableur –grapheur Regressi
- Un multimètre.

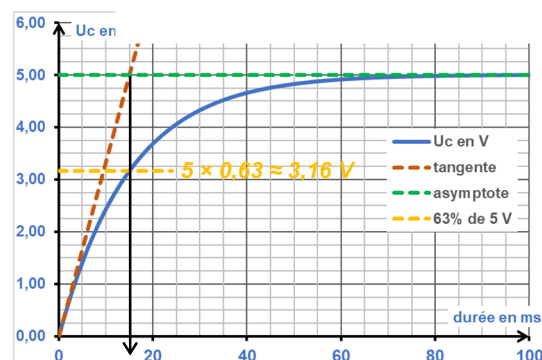


Données :

Schéma du circuit de charge d'un condensateur :



Branchement vers une carte d'acquisition, un microcontrôleur ou un oscilloscope



Si l'on suit l'évolution de la tension aux bornes du condensateur en fonction du temps, on obtient une courbe similaire à la courbe bleue ci-dessus.

Travail à effectuer :

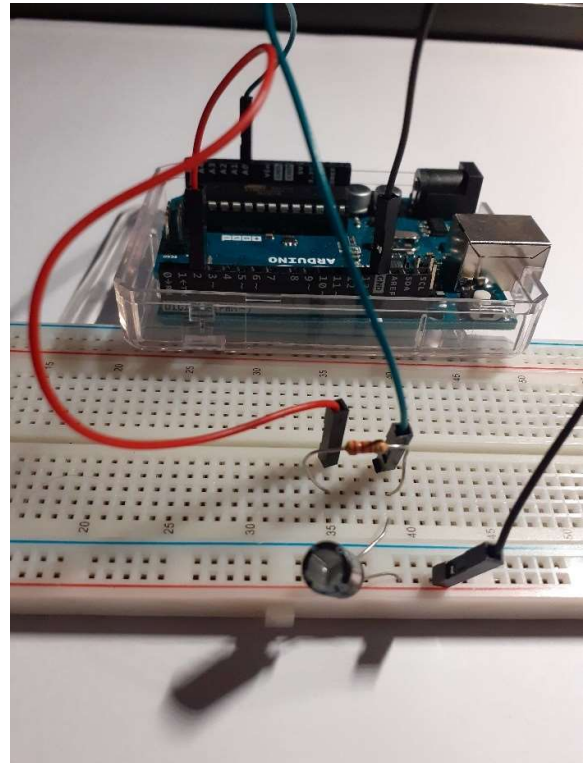
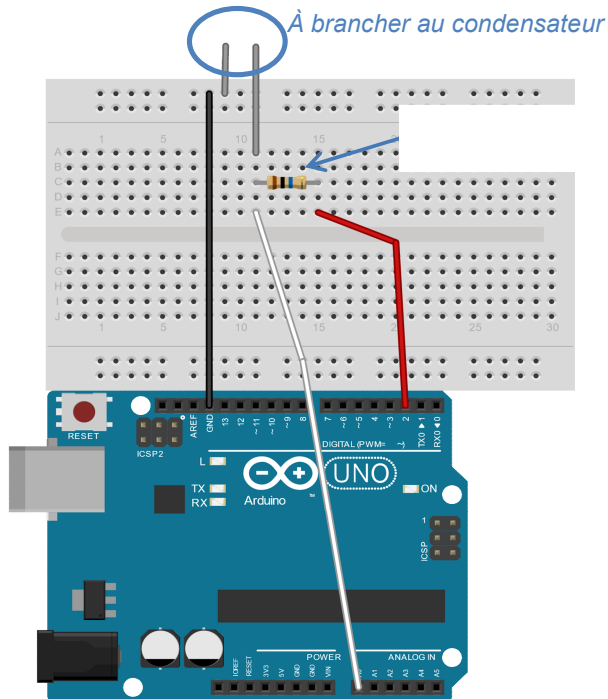
1. Réalisation du circuit

1.1. Mesure et réglages préliminaires

1.1.a. On utilise deux condensateurs de capacité $C = 100 \mu\text{F}$ et des conducteurs ohmiques de résistance R pour réaliser le circuit identique schématisé ci-dessous.

Avec un multimètre, relever la valeur des résistances R des deux conducteurs ohmiques que vous avez choisis.

1.2. Câblage du circuit



APPEL N°1



Appeler le professeur pour lui présenter le montage ou en cas de difficulté

Réaliser, sur la boîte de connexion le montage schématisé ci-dessus.

Ce circuit correspond à celui schématisé ci-dessus sans interrupteur.

La borne 2 peut être portée à +5V pour charger le condensateur, ou à 0V pour le décharger.

La borne A0 permet de mesurer la tension aux bornes du condensateur.

1.3. Téléversement du programme

A l'aide du logiciel Arduino IDE, et en branchant la carte UNO au port USB de l'ordinateur, téléverser le programme suivant sur la carte UNO (il se trouve dans commun classe). **Vous expliquerez l'intérêt de chaque groupe de lignes de ce programme.**

Programme :

```
1 // broches
2 constint(Alim) =2;
3 constint(mesure) = A0;
4 // grandeurs
5 unsignedlong Temps =0;
6 unsignedlong Duree =0;
7 float Uc;
8 int IntervalMesure =10; //mesures toutes les 10 ms
9
10 voidsetup(){
11   Serial.begin(9600); // initialisation port série
12   // décharge préalable de C
13   pinMode(Alim, OUTPUT);
14   digitalWrite(Alim, LOW);
15   delay(2000); //attente de 2s
16   // charge de C
17   int i =0;
18   digitalWrite(Alim, HIGH);
19   while(i<=100){
20     if (millis()-Temps >= IntervalMesure){
21       Temps=millis();
22       Duree = millis()-1999;
23       Uc = (float(analogRead(mesure)) *5/1023);
24       Serial.print(Duree);
25       Serial.print("\t");
26       Serial.print(Uc);
27       Serial.print("\n");
28     i++;
29   }
30 }
31 // décharge de C
32 i =0;
33 Temps = millis();
34 Temps = Temps - Temps; // mise à 0 du chrono
35 digitalWrite(Alim, LOW);
36 while(i<=100){
37   if (millis()-Temps >= IntervalMesure){
38     Temps = millis();
39     Duree = millis()-1999;
40     Uc = (float(analogRead(mesure)) *5/1023);
41     Serial.print(Duree);
42     Serial.print("\t");
43     Serial.print(Uc);
44     Serial.print("\n");
45   i++;
46 }
47 }
48 }
49 voidloop(){
50 }
```

On rappelle que la borne A0 donne une valeur numérique de la tension. Elle peut prendre $2^{10} = 1024$ valeurs différentes comprises entre 0 (qui correspond à 0 V) et 1023 (qui correspond à 5 V).

Les lignes 1 à 9 définissent les variables qui seront utilisées par le programme.

La ligne 11 prépare l'écriture des résultats dans le moniteur série.

Les lignes 13 à 15 organisent la décharge du condensateur.

1. Que se passe-t-il lors de l'exécution de la ligne 18 ?

.....

2. Quelle est la différence entre la variable Temps et la variable Durée ?

.....

3. Quel est le rôle de la ligne 23 ?

.....

4. À quoi peut-on voir que la charge du condensateur va durer 1s ?

.....

5. Quelle ligne déclenche la décharge du condensateur ?

.....

2. Acquisition

Ouvrir le « moniteur série » situé en haut à droite du logiciel Arduino IDE.

Le choix a été fait de mettre le programme dans le setup, ainsi, une mesure est faite au lancement du programme et non en permanence.

Pour faire une autre acquisition, il suffit de relancer le programme en appuyant sur le bouton reset de la carte.

Ce programme affiche les couples (temps ; tension aux bornes du condensateur) toutes les 10 ms dans le moniteur série.

Un copier-coller permet de les récupérer dans un tableur.

(Si besoin, dans un tableur, "ctrl +H" permet d'ouvrir une fenêtre pour, par exemple, remplacer les points pas de virgules.)

3. Affichage des courbes

Deux possibilités pour créer les graphes représentant la tension aux bornes du condensateur en fonction du temps.

Quelle option prendre ? Essayer les deux méthodes !

1) Utiliser le tableur Regressi.

Ouvrir le logiciel Regressi et dans le menu « Edition » sélectionner « coller document » : l'ensemble des données sont alors visibles sous forme de graphes que vous allez exploiter. Ne pas oublier de sauvegarder ce fichier regressi !

2) Utiliser la programmation Python.

Cela nécessite que les données soient sauvegardées dans un fichier csv appelé 'valeurs.csv' (utiliser le logiciel excell) dans le même dossier que ce programme python™.

Ouvrir le logiciel Pyzo puis Copier-coller le programme suivant dans un fichier py


```
1 import matplotlib.pyplot as plt
2
3 t = []
4 U = []
5 #ouverture du fichier csv en lecture
6 fichier = open('valeurs.csv', 'r')
```



```

7 fichier.readline()
8
9 #lecture du tableau ligne par ligne pour remplir progressivement
10 # les variables en remplaçant les , par des .
11 for ligne in fichier:
12     ligne_lue=ligne.split(';')
13     t.append(float(ligne_lue[0].replace(',','.')))
14     U.append(float(ligne_lue[1].replace(',','.')))
15
16 fichier.close()
17
18 plt.scatter(t,U, marker="+", label='U=f(t)')
19 plt.xlabel('temps en ms')
20 plt.ylabel('U en V')
21
22
23 plt.grid()
24 plt.legend()
25 plt.savefig("charge_decharge.png")
26 plt.show()

```

APPEL N°2	Appeler le professeur pour lui présenter les résultats ou en cas de difficulté
	

4. Défi supplémentaire

- 1) Proposer et réaliser un protocole qui permette d'obtenir un signal carré puis un signal triangle en modifiant un constituant du circuit électronique. Visualiser les courbes obtenues.
- 2) Proposer et réaliser un protocole qui permette d'écouter les sons générés par ces signaux triangle et carré.

Défaire le montage et ranger le matériel sur la pailasse.