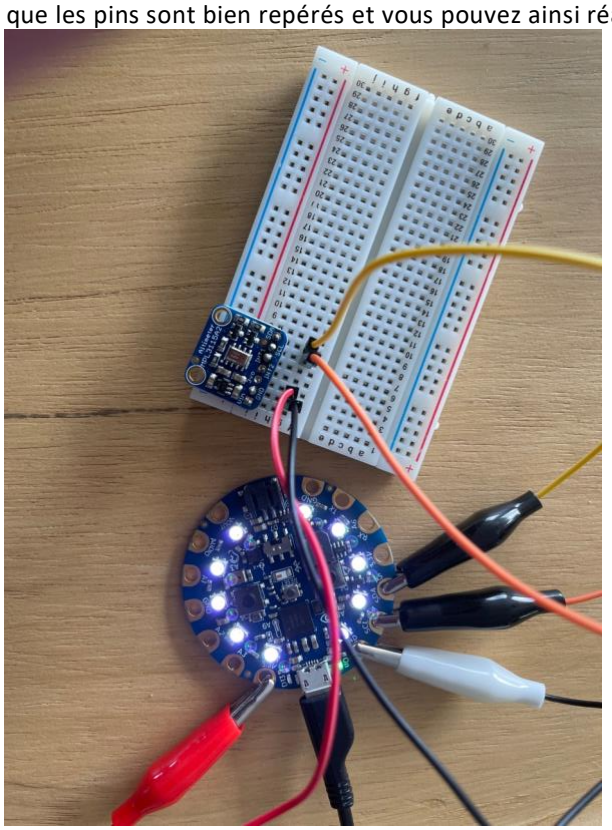
TP connexion d'un capteur et récupération de données

Objectif : connecter un capteur (ici un capteur de pression barométrique/Altimètre/température : MPL3115A2) puis apprendre à stocker ces valeurs sur la mémoire flash.

Vous allez tout d'abord connecter le capteur à la carte.
Nous allons utiliser les power Pads GRD et 3,3V et aussi les pins A4/SCL aussi désignés sous le nom de I2C SCL et A5/SDA désignés aussi sur le nom de I2C SDA.
Remarque :
I2C est un bus informatique (dispositif de transmission de données) devenu très commun. La connexion est réalisée à l'aide de 2 lignes : SDA (Serial Data Line) et SCL (Serial Clock Line).

Sur la carte du capteur vous verrez que les pins sont bien repérés et vous pouvez ainsi réaliser le montage ci-dessous.



Nous allons ensuite initialiser la connexion I2C avec le capteur :
Recopier le code suivant :

```
# initialization I2C
import board
import busio
import adafruit_mpl3115a2
i2c = busio.I2C(board.SCL, board.SDA)
sensor = adafruit_mpl3115a2.MPL3115A2(i2c)

# affichage pression, altitude, temperature
print('Pressure: {0:0.3f} pascals'.format(sensor.pressure))
print('Altitude: {0:0.3f} meters'.format(sensor.altitude))
print('Temperature: {0:0.3f} degrees Celsius'.format(sensor.temperature))
```

Remarque :
Sans rentrer dans les détails, {0:0.3f} permet de demander un affichage de 3 chiffres après la virgule en notation décimale. Si vous remplacez le f par d, vous aurez une notation scientifique.

Et l'enregistrer sur la carte, en le nommant code.py comme d'habitude.
Nous allons pouvoir avoir accès aux données.
La pression est donnée en Pascal (Pa), la température en Celsius (°C) et l'altitude est déduite de la pression atmosphérique mais nous y reviendrons...

Pour paramétrer le calcul de l'altitude, nous avons besoins de la pression atmosphérique actuelle à Grenoble ramenée au niveau de la mer. Vous pouvez aller la chercher ici. Attention de bien remettre la valeur en Pa et non hPa (1hPa=1hectoPascal=100 Pa)!

Une fois trouvée, rentrez le code suivant :

```
sensor.sealevel_pressure = valeur trouvée sur le site
```

Refaites une demande de valeurs pour vérifier que l'altitude est bien étalonnée.

Maintenant, nous allons voir comment stocker puis récupérer ces valeurs afin de les exploiter.

Pour stocker ces valeurs, nous allons utiliser le format .csv (comma-separated values : valeurs séparées par une virgule).

Un petit exemple de csv :

```
Pression, Altitude, température
102 000 000, 325, 22
101 012 020, 327, 24
```

Et ainsi de suite. Les colonnes sont remplacées par les virgules.

Il va tout d'abord falloir rendre notre mémoire flash disponible en écriture. Pour l'instant, le microcontroleur peut uniquement la lire.

Recopier le code suivant et le nommer **boot.py**

```
import board
import digitalio
import storage

switch = digitalio.DigitalInOut(board.D7)

switch.direction = digitalio.Direction.INPUT
switch.pull = digitalio.Pull.UP

storage.remount("/", switch.value)
```

Placer ce code dans la mémoire flash de la carte.
Désormais, si le switch est placé à droite, la carte peut écrire sur la mémoire, et à gauche elle ne peut pas.

```
# Lire la pression atmosphérique et la temperature, sortie en .csv

import time
import board
import digitalio
from adafruit_circuitplayground import cp
import busio
import adafruit_mpl3115a2
i2c = busio.I2C(board.SCL, board.SDA)
sensor = adafruit_mpl3115a2.MPL3115A2(i2c)

# règle la NeoPixel 0 en vert (statut), NeoPixel 1 pour la collecte de
# data
cp.pixels[0] = (0, 90, 0) # codage RGB
cp.pixels[1] = (0, 0, 90) # Pixel 1 en bleu quand il y a collecte de data
num_readings = 50 # parameter le nombre de valeurs voulues

try:
    with open("/temp-pression.csv", "a") as fp:
        while True:
            fp.write('Pression, Temperature\n') # première ligne
            for x in range(0, num_readings):
                pres = sensor.pressure
                fp.write(str(pres) + "," + str(sensor.temperature) + "\n")
```

```
          # Vous pouvez modifier la valeur de l'attente entre 2 mesures :
          # 1 minute=60 sec, 5 mins=300 sec, 1 hour=3600...
               time.sleep(1)
               if cp.button_a:
                   break
      # Done, set NeoPixel 1 to green also
           cp.pixels[1] = (0, 90, 0)
except OSError as e:
    # parametre NeoPixel 1 eteinte and NeoPixel 0 clignotante

#   gestion des erreurs
    cp.pixels[1] = (0, 0, 0) # NeoPixel 1 en blanc
    message_color = (99, 0, 0)   # Rouge pour un probleme
    if e.args[0] == 28: # Plus d'espace
        message_color = (228, 160, 40) # Parametrage en Orange
    elif e.args[0] == 30:  # La carte est en lecture
        message_color = (181, 90, 0) # Parametrage en Jaune
for x in range(1, 10): # Flash 10 fois
    cp.pixels[0] = message_color
    time.sleep(1)
    cp.pixels[0] = (0, 0, 0)
    time.sleep(1)
```

Recopier ce code et l'enregistrer sur la carte sous code.py

Ensuite, débrancher la carte de l'ordinateur (USB), mettre le switch D7 à droite, puis reconnecter la carte.
Un fichier temp_pression.csv se créera et enregistrera les données.
Faire tourner le conde quelques secondes pour avoir assez de valeurs puis récupérez le fichier .csv créé afin de l'ouvrir avec un tableur tel que Libre Office Calc.