ESSAI04 AVRIL 2021 Vincent HAUTIERE

ESSAI TP CALORIMETRIE ET ACQUISITION DES DONNEES AVEC MICROCONTROLEUR

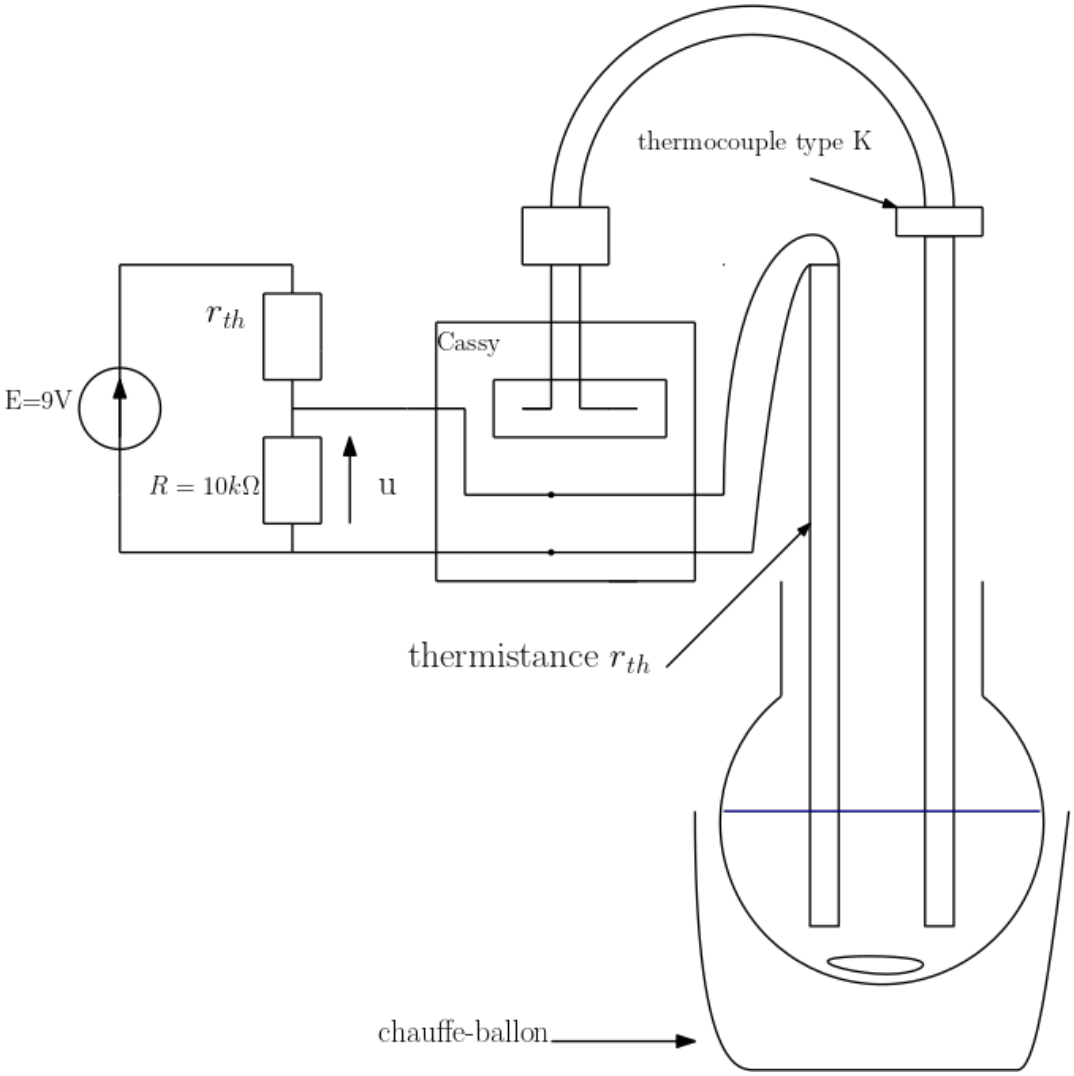
Avant tout, je tiens à remercier **Lionel Desbordes** pour ces conseils avisés sur la carte BBC micro:bit et **Nicolas Faucher** pour son protocole de la détermination expérimentale de la capacité du calorimètre.

- **MANIP01 Etalonnage d'une thermistance et modélisation à l'aide de `curve_fit`**
  Il s'agit ici d'étalonner le capteur de température utilisé, ici, une thermistance. J'ai donc pré-étalonner un thermocouple type K à l'aide de Cassy et d'un thermomètre de référence électronique calibré (modèle testo 925).

  > - Afin de faire l'acquisition de la résistance thermise r_{th}, il est nécessaire de la mettre dans un pont diviseur de tension.
  >
  > Voici donc le montage de la MANIP01 :

In [1]:
```
from IPython import display
display.Image("im01.png",width=500)
```

Out[1]:



- Pour faire cet étalonnage, j'ai plongé le thermocouple et la thermistance dans une eau refroidie à 2°C et j'ai mis en chauffe doucement (durée de la manipulation 1h) jusqu'à 100°C.

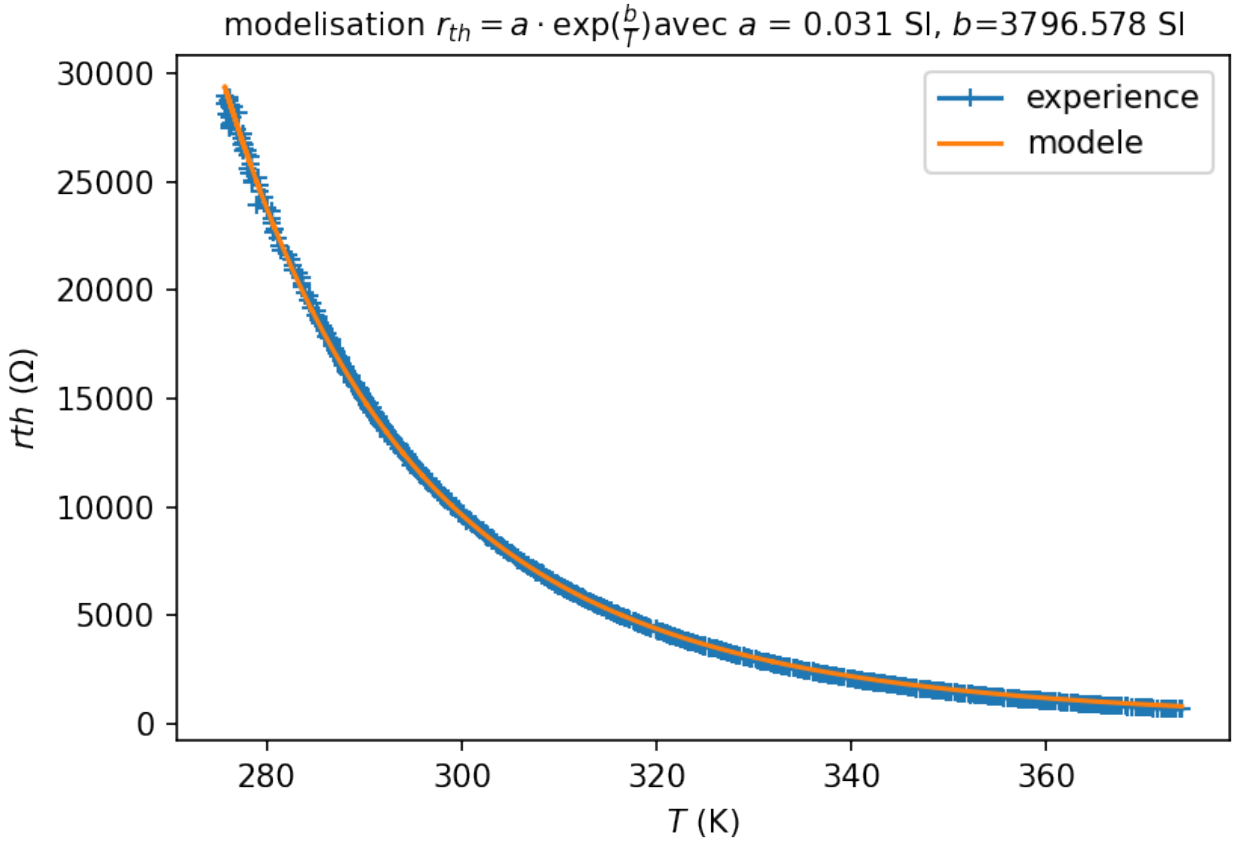
- Pour des raisons d'homogénéisation de la température, j'ai fait cette manip dans un chauffe ballon avec agitation magnétique en enregistrant simultanément toutes les 10 secondes la température donnée par le thermocouple et la tension aux bornes de la résistance R à l'aide de Cassy. j'ai obtenu 721 couples de points (T,u).

- La notice constructeur de la thermistance précise un comportement de r_{th} du type : $r_{th} = a \cdot \exp(\frac{b}{T})$ J'ai donc utilisé `curve_fit` du module `scipy` pour déterminer a et b.

In [2]:
```
#MANIP01 : ETALONNAGE DE LA THERMISTANCE

#IMPORTATIONS DES MODULES ET FONCTION readCSV de LECTURE CSV
import csv
import math
import matplotlib.pyplot as plt
import numpy as np
import scipy as sp
from scipy.optimize import curve_fit



def readCSV(file,sep,n):
    with open(file,"r") as f:
        read=csv.reader(f,delimiter=sep)
        col=[]
        for row in read :
            try:
                col.append(float(row[n].replace(",","-")))
            except:
                pass
    return col

#listes des valeurs experimentales

#Temperature de l'étalon
T=readCSV("essai04_etalonnage_thermistance.csv",";",1)

#Tension aux bornes de R
u=readCSV("essai04_etalonnage_thermistance.csv",";",2)

#Passage de T en kelvin
Tk=[T[i]+273.15 for i in range (len(T))]

#Calcul de rthermique à partir du pont diviseur
rth=[10000*(9-u[i])/u[i] for i in range(len(u))]


#MODELISATION rth=a*exp(-b*T)
# Initialisation des paramètres du modele
a=1
b=1

#definition du modele et optimisation
def func(Tk,a,b):
    return a*np.exp((b/Tk))
(a_opt,b_opt),_=curve_fit(func,Tk,rth,p0=[a,b])

#tracés de la courbe experimentale et du modele Rth en fonction de Tk
print('a','=',a_opt,'b','=',b_opt)
plt.figure(dpi=150)
plt.plot(Tk,rth,'-+',label='experience')
plt.plot(Tk,func(Tk,a_opt,b_opt),label='modele')
plt.xlabel('$T$ (K)')
plt.ylabel('$rth$ ($\Omega$)')
plt.title(r"modelisation $r_{{th}}=a\cdot \exp(\frac{{b}}{{T}})$"
          r"avec $a$ = {:.3f} SI, $b$={:.3f} SI"
```

```
            .format(a_opt, b_opt), fontsize=10)
plt.legend()
plt.show()
```

```
a = 0.030726422201553992 b = 3796.5782015254404
```



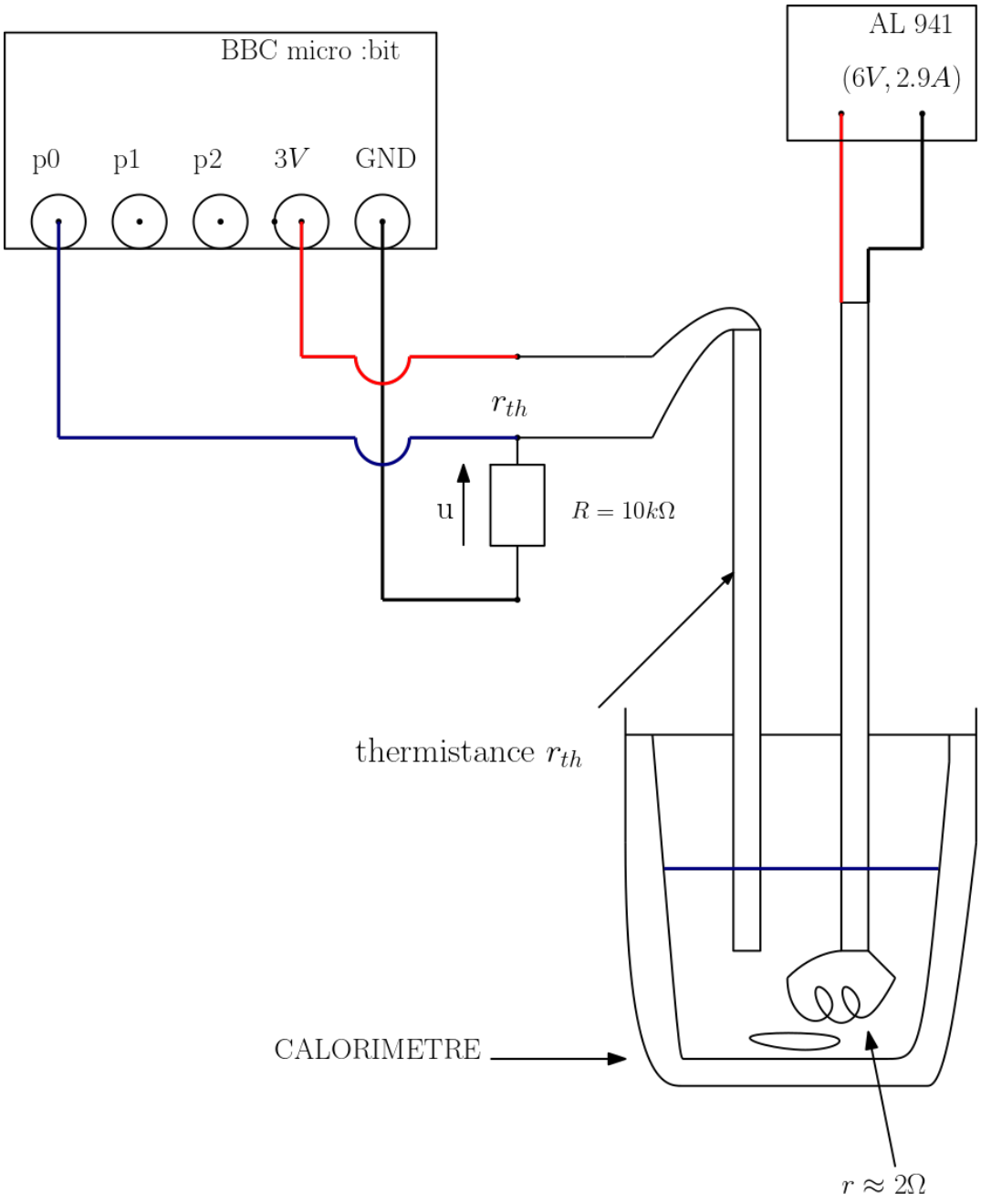
- **MANIP02 Détermination de la capacité du calorimètre étudié**
  Il s'agit de chauffer une certaine masse d'eau connue à l'aide d'une résistance chauffante d'environ 2Ω alimentée par une alimentation stabilisée AL 941. La puissance électrique délivrée est ici $P_{elec} = 6,0(V) \cdot 2,9(A) = 17,4W$. On connait : $m_{eau} = 311,64\ g$, $c_{eau} = 4180\ J.K^{-1}.kg^{-1}$.

  > - On mesure la température à l'aide la thermistance étalonnée au cours du temps en stockant les données (u,t) dans la mémoire flash du micro-controleur BBC micro:bit.
  > - u représente toujours la tension prise dans le montage pont-diviseur. Ce montage est alimenté par l'alimentation (piles $2 \times 1,5V$) du micro-controleur.

> Voici le montage de la MANIP02 :

In [3]:
```
from IPython import display
display.Image("im02.png",width=500)
```

Out[3]:



- **MANIP02 Première étape : acquisition de données par le micro-controleur BBC micro:bit**
  Le programme micropython ci-dessous :

  > - récupère les couples de points bruts `(val,t)` avec `val` une valeur numérique comprise entre 0 et 1023 (échantillonnage 10 bits) et `t` en ms.
  > - convertit la liste de valeurs `val` en tension $u = 3,2 \cdot \frac{val}{1023}$
  > - enregistre les listes `t` et `u` dansle fichier `essai04_capa.csv` dans la mémoire flash du micro-controleur.
  > - ⚠ La mesure de la tension d'alimentation du micro-controleur $(3,2V)$ est cruciale car elle intervient dans le passage de `val` à `u` or les valeurs de `u` permettent de remonter aux valeurs de r_{th} et donc à la température T. J'ai utilisé un voltmètre *de confiance* pour mesurer la tension de la pile de l'alimentation du micro-controleur.

In [ ]:

```
'''
Ce bloc d'instructions est entre commentaire car
il nécessite la présence d'un micro-contrôleur connecté

MANIP02 ACQUISITION DES MESURES (t,u) AVEC MICRO:BIT

import microbit as m

while True:  #boucle "infinie"
    if m.button_a.is_pressed():
        m.display.clear()
        m.pin0.write_digital(0) #mise à 0V du pin0
        m.display.scroll("GO")

        # nombre de points de mesure
        N = 120

        # periode d'échantillonnage en ms
        te = 10000

        t = []
        val = []

        # définition de l'instant initial
        t0 = m.running_time()

        m.sleep(te)
        for i in range(N):

            # instant de mesure
            tn = m.running_time()-t0

            # mesure de la tension sur p0, valeur entre 0 et 1023(10bits)
            valn = m.pin0.read_analog()

            t.append(tn)
            val.append(valn)
            m.display.scroll(str(i))
            m.sleep(te)
        m.display.scroll('FIN')

        #conversion de val en tension u (volt)
        u = [3.2*float(val[i])/1023 for i in range(len(val))]

        with open('essai04_capa.csv', 'w') as f:
            for i in range(len(t)):
                f.write(str(t[i])+";"+str(u[i])+"\n")

    else:
        m.display.scroll("A")
'''
```

- **MANIP02 deuxième étape : exploitation des données et détermination de $C_{calorimètre}$**

  > - Considérations sur la puissance des pertes : On peut "raisonnablement" considérer la puissance des pertes négligeables devant la puissance électrique délivrée (faible variation de température (de 27°C à 35°C) et proche de $\theta_{ambiant}$). Sinon à tester il faudrait refaire l'expérience avec une autre puissance électrique délivrée pour pouvoir éliminer cette puissance de pertes en la supposant constante. (voir le protocole des tp de mpsi et pcsi en calorimétrie).

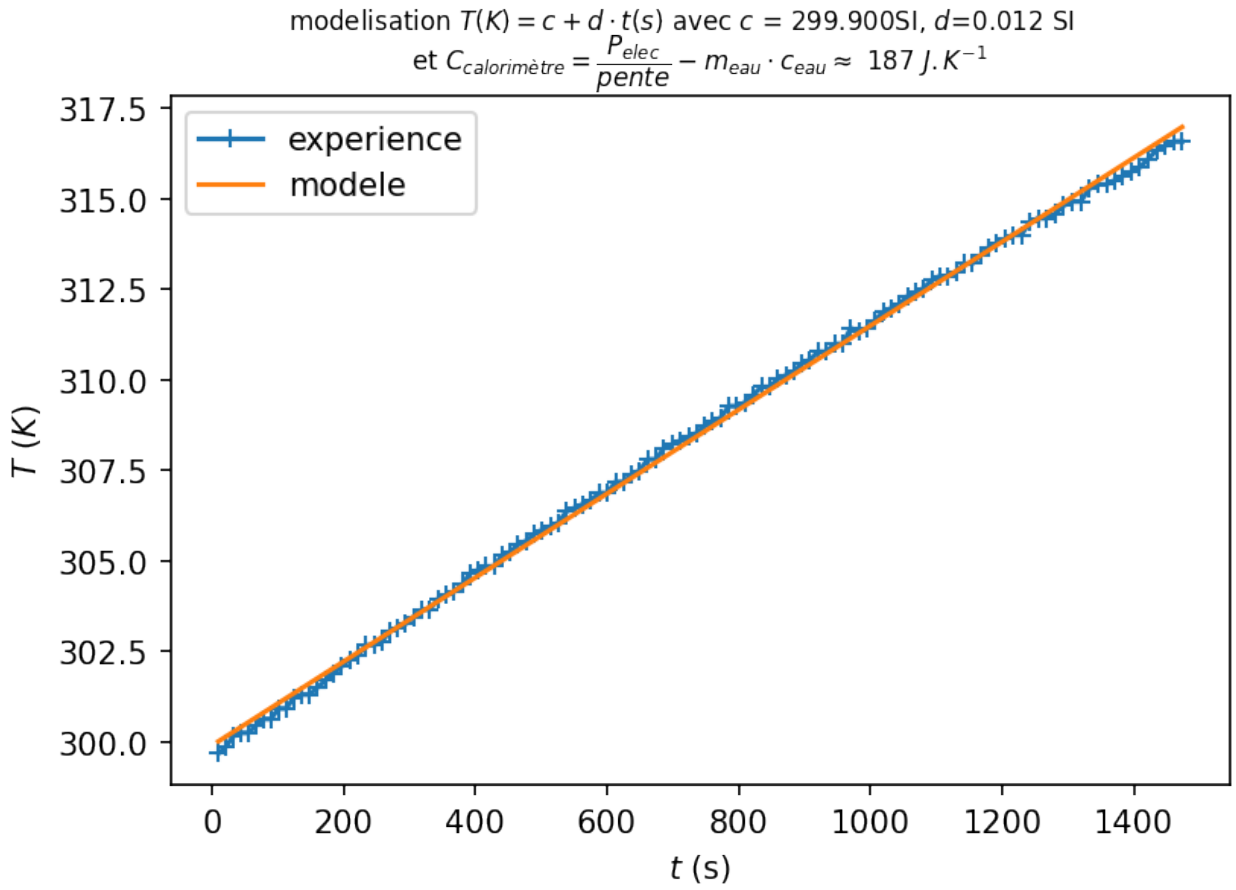
- $\Delta H = \left(C_{calorimètre} + m_{eau}c_{eau}\right) \cdot (\theta(t) - \theta_i(t_i)) = (P_{elec} - P_{pertes}) \cdot (t - t_i)$
  $\Delta H \approx P_{elec} \cdot (t - t_i)$

  > En traçant $\theta(t)$ et par la détermination de la pente, on en déduit
  > $C_{calorimètre} = \dfrac{P_{elec}}{pente} - m_{eau} \cdot c_{eau}$
  > Voici l'exploitation python ci-dessous :

In [5]:

```
#MANIP02 EXPLOITATION DES MESURES ET DETERMINATION
#DE LA CAPACITE DU CALORIMETRE

#listes des valeurs experimentales obtenues avec la carte micro:bit

#temps en ms
tms=readCSV("essai04_capa.csv",";",0)

#tension lue au bornes de R
u=readCSV("essai04_capa.csv",";",1)

#temps en seconde
ts=[tms[i]/1000 for i in range (len(tms))]

#resistance thermique
rth=[10000*(3.2-u[i])/u[i] for i in range(len(u))]

#temperature en kelvin obtenue à l'aide de la courbe d'etalonnage
Tk=[b_opt/(np.log(rth[i]/a_opt)) for i in range (len(rth))]

#definition du modele Tk=c+d*ts
# Initialisation des paramètres du modele:
c=1
d=1

#definition du modele et optimisation
def func(ts,c,d):
    return c+d*np.array(ts)
(c_opt,d_opt),_=curve_fit(func,ts,Tk,p0=[c,d])

print('c','=',c_opt,'d','=',d_opt)
Ccalorimetre=6*2.88/d_opt-0.31164*4180
print('Ccalorimetre','=',Ccalorimetre)
plt.figure(dpi=150)
plt.plot(ts,Tk,'-+',label='experience')
plt.plot(ts,func(ts,c_opt,d_opt),label='modele')
plt.suptitle(r"modelisation $T(K)=c+d\cdot t(s)$ avec $c$ = {:.3f}"
             r"SI, $d$={:.3f} SI ".format(c_opt, d_opt), fontsize=8)
plt.title(r"et $C_{calorimètre}=\dfrac{P_{elec}}{pente}-m_{eau}\cdot$"
          r"$c_{eau}\approx$ $187$ $J.K^{-1}$",fontsize=8)
plt.xlabel('$t$ (s)')
plt.ylabel('$T$ ($K$)')
plt.legend()
plt.show()
```

```
c = 299.89974412038407 d = 0.011597112907657128
Ccalorimetre = 187.3708209237666
```

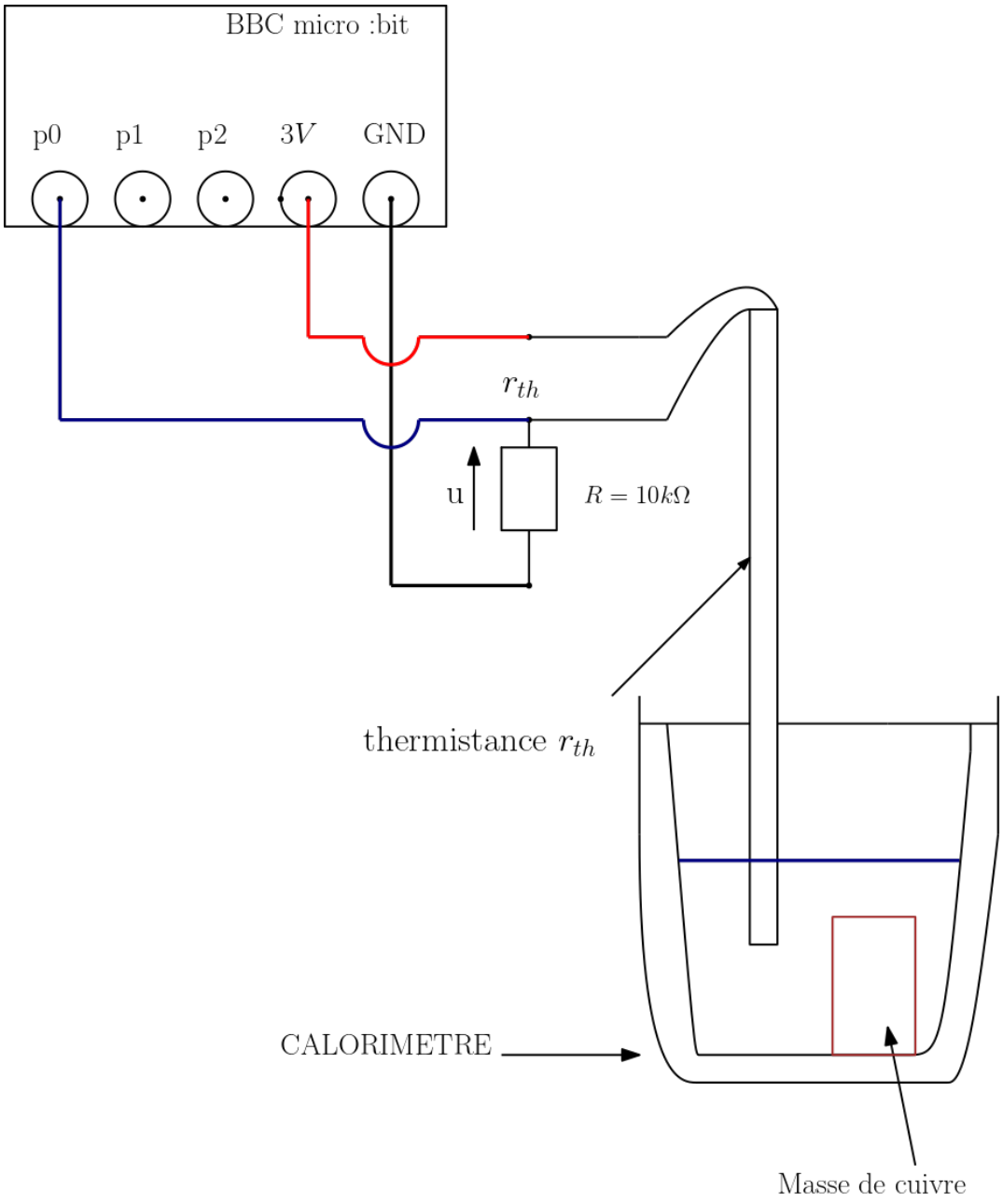


- **MANIP03 Détermination de la capacité thermique massique du cuivre**
  Il s'agit de chauffer à une température proche de $100^\circ C$ une masse de cuivre connue et de le plonger rapidement dans le calorimètre étudié contenant une masse d'eau connue à température ambiante. On mesure la température dans le calorimètre toujours avec la thermistance étalonnée.
  On a $m_{eau} = 311,64\ g$, $c_{eau} = 4180\ J.K^{-1}.kg^{-1}$, $m_{cuivre} = 188,50\ g$, $T_{cuivre} \approx 95^\circ C$ et la capacité du calorimètre déterminé précédemment $C_{calorimètre} \approx 187\ J.K^{-1}$

  > On mesure la température à l'aide de la thermistance étalonnée au cours du temps en stockant les données (u, t) dans la mémoire flash du micro-controleur BBC micro:bit.
  >
  > - u représente toujours la tension prise dans le montage pont-diviseur. Ce montage est alimenté par l'alimentation (piles $2 \times 1,5V$) du micro-controleur.
  >
  > Voici le montage de la MANIP03 :

In [6]:
```
from IPython import display
display.Image("im03.png",width=500)
```

Out[6]:



- **MANIP03 Première étape : acquisition de données par le micro-controleur BBC micro:bit**
  Le programme micropython ci-dessous :

  > - récupère les couples de points bruts `(val,t)` avec `val` une valeur numérique comprise entre 0 et 1023 (échantillonnage 10 bits) et `t` en ms.
  > - convertit la liste de valeurs `val` en tension $u = 3,2 \cdot \frac{val}{1023}$
  > - enregistre les listes `t` et `u` dansle fichier `essai04_acquisition_solide.csv` dans la mémoire flash du micro-controleur.
  > - ⚠ Comme expliqué précédemment, la mesure de la tension d'alimentation du micro-controleur $(3,2V)$ est cruciale car elle intervient dans le passage de `val` à `u` or les valeurs de `u` permettent de remonter aux valeurs de r_{th} et donc à la température T.

In [ ]:
```
'''
Ce bloc d'instructions est entre commentaire car
il nécessite la présence d'un micro-contrôleur connecté

MANIP03 ACQUISITION DES MESURES (t,u) AVEC MICRO:BIT

import microbit as m

while True:  #boucle "infinie"
    if m.button_a.is_pressed():
        m.display.clear()
        m.pin0.write_digital(0) #mise à 0V du pin0
        m.display.scroll("GO")

        # nombre de points de mesure
        N = 240

        # periode d'échantillonnage en ms
        te = 5000

        t = []
        val = []

        # définition de l'instant initial
        t0 = m.running_time()

        m.sleep(te)
        for i in range(N):

            # instant de mesure
            tn = m.running_time()-t0

            # mesure de la tension sur p0, valeur entre 0 et 1023(10bits)
            valn = m.pin0.read_analog()

            t.append(tn)
            val.append(valn)
            m.display.scroll(str(i))
            m.sleep(te)
        m.display.scroll('FIN')

        #conversion de val en tension u (volt)
        u = [3.2*float(val[i])/1023 for i in range(len(val))]

        with open('essai04_acquisition_solide', 'w') as f:
            for i in range(len(t)):
                f.write(str(t[i])+";"+str(u[i])+"\n")

    else:
        m.display.scroll("A")
'''
```

- **MANIP03 deuxième étape : exploitation des données et détermination de c_{cuivre}**

  > On exploite la courbe ci-dessous. (je l'ai fait à la main!!! (honte à moi))

- $\Delta H = \left(C_{calorimètre} + m_{eau} \cdot c_{eau}\right) \cdot \left(T_f - T_i\right)\right) + m_{cuivre} \cdot c_{cuivre} \cdot \left(T_f - T_{cuivre}\right)$
  $\Delta H \approx 0$

  > On en déduit
  >
  > $$c_{cuivre} = \left(C_{calorimètre} + m_{eau} \cdot c_{eau}\right) \cdot \frac{T_i - T_f}{m_{cuivre} \cdot \left(T_f - T_{cuivre}\right)}$$
  >
  > Voici l'exploitation python ci-dessous :

In [8]:
```
#MANIP03 DETERMINATION DE LA CAPACITE THERMIQUE DU CUIVRE

#temps en ms
tms=readCSV("essai04_acquisition_solide.csv",";",0)
#tension lue au bornes de R
u=readCSV("essai04_acquisition_solide.csv",";",1)
#temps en seconde
ts=[tms[i]/1000 for i in range (len(tms))]
#resistance thermique
rth=[10000*(3.2-u[i])/u[i] for i in range(len(u))]
#temperature en kelvin obtenue à l'aide de la courbe d'etalonnage
T=[b_opt/(np.log(rth[i]/a_opt)) for i in range (len(rth))]
ccuivre=(Ccalorimetre+0.31164*4180)*(301.7-305.5)/(0.1885*(305.5-368.15))
print('ccuivre','=',ccuivre)
plt.figure(dpi=150)
plt.plot(ts,T,'-+')
plt.xlabel('$t$ (s)')
plt.ylabel('$T$ ($K$)')
#plt.title(r"trace T en fonction du temps")
plt.grid(True)
plt.title(r"$c_{cuivre}=(C_{calorimètre}+m_{eau}$ $\cdot c_{eau})$"
          r"$\cdot\dfrac{T_i-T_f}{m_{cuivre}\cdot (T_f-T_{cuivre})}$"
          r"$\approx$ $479$ $J.K^{-1}.kg^{-1}$",fontsize=10)
plt.show()
```

```
ccuivre = 479.4518729170167
```

$$c_{cuivre} = (C_{calorimètre} + m_{eau} \cdot c_{eau}) \cdot \frac{T_i - T_f}{m_{cuivre} \cdot (T_f - T_{cuivre})} \approx 479\, J.K^{-1}.kg^{-1}$$