Physique et informatique

**Mesure de résistances, de diodes, de capacités,
d'un générateur et d'un moteur
avec Raspberrypi et INA219**

Guyot Vincent

4 février 2023

Table des matières

1. But

Il s'agit d'illustrer une chaîne de traitement de mesures physiques d'éléments électroniques par l'intermédiaire du multimètre INA219 branché sur un raspberrypi 4, le traitement des mesures obtenues dans un simple fichier texte étant alors fait directement dans LaTeX à l'aide du logiciel Gnuplot.

2. Théorie

Plusieurs éléments ont été étudiés :
— des résistances,
— des diodes,
— des condensateurs,
— un générateur et
— un moteur électrique.

2.1. Résistance

Pour les mesures de résistance, la physique est très simple. Il s'agit de la loi d'Ohm.

$$U = R \cdot I \tag{1}$$

Si la physique est simple, le dispositif expérimental nécessaire pour en faire la vérification sera complexe à deux points de vue :
— la gestion des incertitudes et
— le transfert des données sur l'ordinateur.
Le rôle de cette expérience au lycée n'est pas dans un premier temps d'explorer une physique complexe. Il s'agit plutôt de montrer que mettre en place une expérience liant physique et informatique est complexe et demande des compétences multiples.

Ces compétences vont être amenées par des étudiants en physique et en informatique qui devront collaborer pour mettre en place le dispositif expérimental et les outils d'analyse informatique.

2.2. Diode

Pour les mesures de caractéristique des diodes, la physique ne sera pas abordée autrement que pour indiquer l'existence d'une valeur limite de conduction, l'objectif étant de la visualiser par le diagramme Tension-courant.

2.3. Condensateur

Pour les condensateurs, la physique de la décharge d'un condensateur est étudiée au cours et la constante de temps $\tau = R \cdot C$ supposée connue.

L'INA219 va permettre de réaliser des mesures que l'humain n'est pas capable de faire en raison de la vitesse du phénomène étudié. Ainsi, la décharge de condensateurs de grande capacité pourra être réalisée automatiquement, malgré le temps long nécessaire, mais surtout la décharge de condensateurs de faible capacité deviendra possible en raison de la possibilité de faire des mesures allant jusqu'à moins d'une milliseconde.

L'équation au cœur du phénomène est :

$$U(t) = U_0 \cdot e^{t/R \cdot C} \tag{2}$$

Une application intéressante est présentée sous la forme d'un dispositif de détection de touché. L'INA219 n'y est pas utilisé, mais la charge d'un condensateur, sa constante de temps et sa relation à la résistance, soit $\tau = R \cdot C$ le sont.

2.4. Générateur

Pour le générateur, c'est la résistance interne qui va nous intéresser [1]. Contrairement à un générateur à courant constant [2], qui voit la tension à ses bornes augmenter quand la charge (une résistance par exemple) augmente, un générateur à tension constante [3], comme une pile ou un accumulateur, voit la tension à ses bornes diminuer avec la charge. En effet, pour un résistance externe qui diminue, si la tension est constante la loi d'Ohm implique que le courant augmente. Or, si le courant augmente, la résistance interne de la pile augmente et donc la tension aux bornes de la pile diminue.

La raison de la décroissance de la tension aux bornes d'un accumulateur ou d'une pile est donc sa résistance interne. Or, celle-ci peut changer non seulement au cours de la décharge de l'accumulateur, mais au cours du temps de vie de la batterie [4].

2.5. Moteur

Nous allons réaliser ici l'étude d'un moteur CC, c'est-à-dire d'un moteur à Courant Continu. En substance, il existe quatre types de moteurs : les moteurs à courant continu (voir [1, p. 96]), les servo-moteurs (voir [1, p. 156]), les moteurs pas-à-pas (voir [1, pp. 178,179]) et les moteur brushless. Pour une description des trois premiers moteurs en relation avec le Raspberry pi, voyez l'excellent ouvrage [1] que je recommande par ailleurs pour toute une série d'autres montages extrêmement intéressants.

D'un point de vue électrique, un moteur CC est constitué d'un bobinage permettant de créer un champ magnétique. Or, ce bobinage constitue une résistance électrique qu'il est possible de mesurer. En effet, d'un point de vue théorique, on peut séparer un moteur CC en deux parties. Une partie constituée d'un moteur idéal sans résistance électrique

1. Voir : `http://www.learningaboutelectronics.com/Articles/Battery-internal-resistance`.
2. Voir : `http://www.zpag.net/Electroniques/Alimentation/GenCourConstant.htm`.
3. Voir : `http://www.ile-reunion.org/louispayen/cours/generateur-courant-tension.htm`.
4. Voir : `https://www.powerstream.com/internal-resistance.htm`.

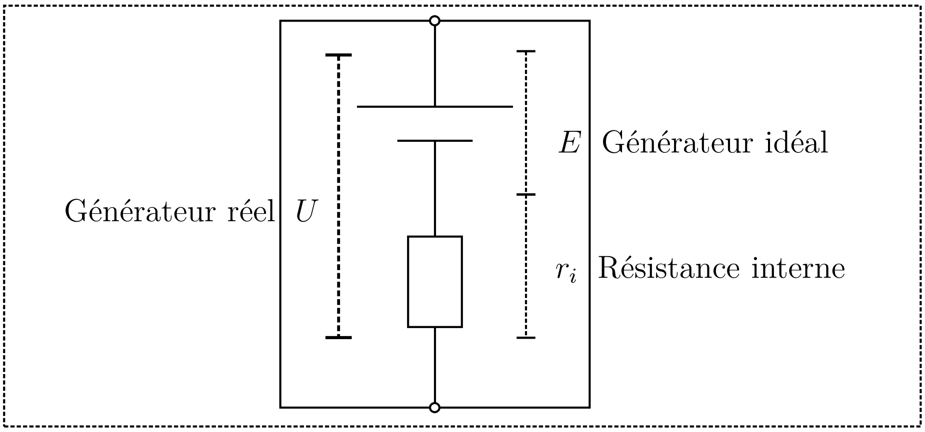


FIGURE 1 – Générateur réel

et une autre constituée d'une résistance représentant la résistance interne du moteur. Ainsi, si on considère la tension aux bornes du moteur réel U, on peut écrire l'équation 3.

$$U = E' + r_i \cdot I \tag{3}$$

où E' est la tension, dite contre-électromotrice, au bornes du moteur idéal, r_i la résistance interne et I le courant dans le moteur.

Par ailleurs, un moteur CC a la particularité de produire une vitesse de rotation proportionnelle à sa tension contre-électromotrice selon l'équation 4.

$$E' = k \cdot \omega \tag{4}$$

En multipliant l'équation 3 par le courant, on obtient l'équation 5 qui nous permet de déterminer avec l'équation 6 la puissance mécanique théorique du moteur.

$$U \cdot I = E' \cdot I + r_i \cdot I^2$$
$$P_{elec.} = P_{mec.} + r_i \cdot I^2 \tag{5}$$
$$P_{mec.} = U \cdot I - r_i \cdot I^2 \tag{6}$$

On voit que la puissance mécanique est une fonction quadratique du courant. On peut donc étudier cette fonction et par dérivation déterminer le courant permettant une puissance maximale, comme le montre l'équation 7.

$$\frac{dP_{mec.}}{dI} = U - r_i \cdot I = 0 \;\Rightarrow\; I = \frac{U}{2 \cdot r_i} \tag{7}$$

Enfin, il est possible de déterminer la puissance mécanique développée par le moteur.

$$P_{mec.} = F \cdot v = F \cdot \frac{L}{t} = F \cdot \frac{R \cdot \alpha}{t} = F \cdot R \cdot \frac{\alpha}{t} = F \cdot R \cdot \omega = M \cdot \omega \tag{8}$$

L'équation 8 et le schéma 2 permettent d'obtenir la puissance mécanique en sachant que c'est soit la force multipliée par la vitesse linéraire, soit le moment de force multiplié par

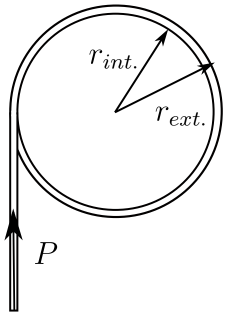


FIGURE 2 – Axe moteur

la vitesse angulaire. On peut donc en connaissant la force exercée par le moteur, qui a la valeur du poids suspendu si la vitesse de celui-ci est constante, sa vitesse de rotation et le bras de levier, c'est-à-dire la distance du centre de l'axe au point d'application de la force, calculer la puissance à partir de mesures simples. On comprends bien que mesurer le poids est simple. Pour la vitesse de rotation et le bras nous verrons dans la description de l'expérience comment le faire.

3. Dispositif expérimental

Le dispositif expérimental est présenté la figure 3. On y voit à gauche le GPIO du Raspberrypi4, au centre l'INA219, la résistance R et le générateur externe à droite.

La résistance de shunt permettant à l'INA de mesurer le courant est représentée en noir contre les ports de mesure. Celle-ci doit se placer en mode « high side », c'est-à-dire entre la tension positive du générateur continu (Vin+) et le pôle de tension la plus élevée de la résistance R (Vin-).

Ainsi, en reliant la masse du générateur, du pôle de tension la plus basse de la résistance R et le Ground (GND) du GPIO, étant donné la faible valeur de la résistance de schunt (Rs = 0,1 Ω), la précision de la mesure de la tension aux bornes de R va peu fluctuer et sera de meilleure qualité qu'avec une mesure « low side ».

Du côté du bus I2C, à part l'alimentation (VCC 3,3V) et la terre (GND) de l'INA, les données sont transmises par la broche SDA et l'horloge par la broche SCL.

La figure 4 présente le dispositif expérimental concrètement. L'INA ainsi que les branchements sont montés sur une « breadboard ». On ne voit pas le générateur qui se trouve sur la gauche de l'image.

La figure 5 presente le dispositif expérimental concrètement. La différence avec la figure 4 est la présence du T-Cobbler plus, permettant une lecture des fonctions des ports du GPIO et un branchement facilité sur le RPI4.

Figure 3 – Le RPI4, l'INA219 en mode high side et la résistance R

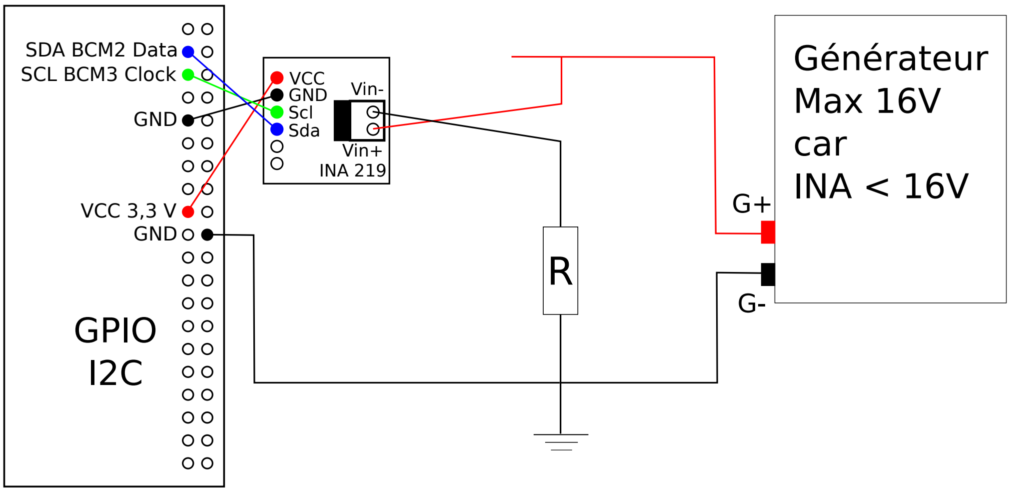
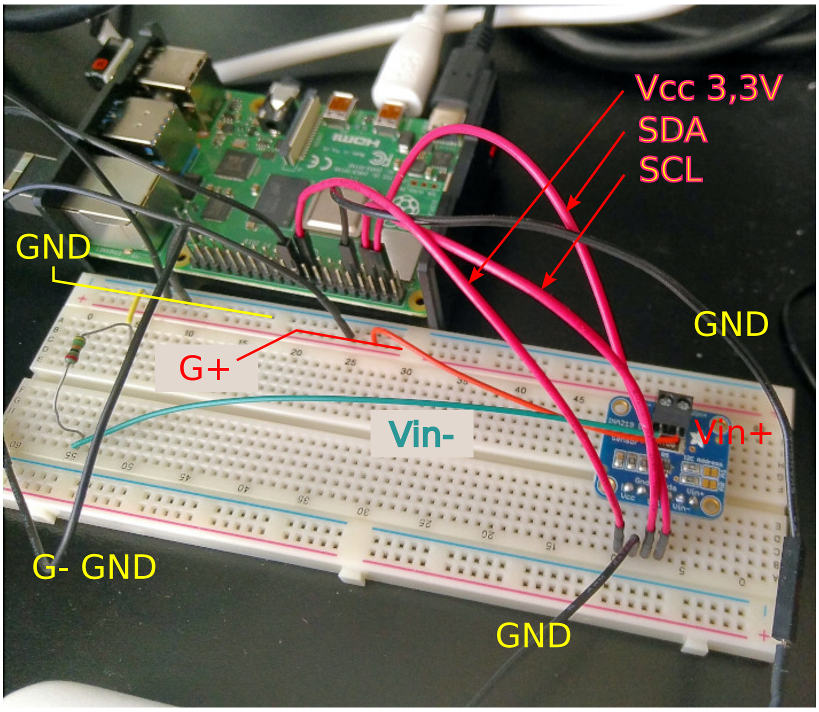


Figure 4 – Le RPI4, l'INA219 et la résistance R



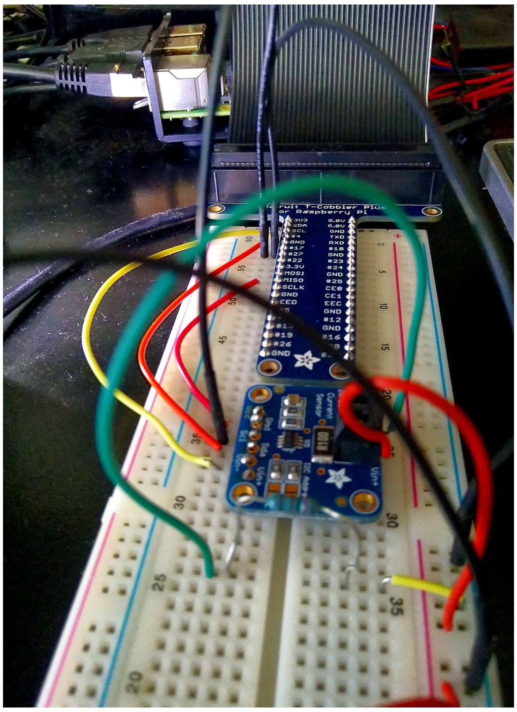


FIGURE 5 – INA, résistance et T-Cobbler plus

3.1. Résistances

La procédure expérimentale a consisté en cinq séries de mesures successives sur différentes résistances. Les données sont récupérées sur le bus I2C (voir annexe A, page 42) à l'aide du programme Python du listing 1.

Il faut relever que le programme enregistre la mesure de la puissance dissipée dans la résistance et suivant la valeur du maximum possible, ici les résistances utilisées étaient des ¼W, annonce le dépassement. Le réglage de la tension se fait en tournant le potentiomètre du générateur et chaque mesure est effectuée en pressant la touche ENTER.

La figure 6 présente en arrière plan le code python permettant les mesures de tension en fonction du courant pour une résistance, l'affichage des mesures faites par ce code et le graphe gnuplot correspondant.

Par ailleurs, Gnuplot permettant la réalisation d'un graphe comportant plusieurs courbes pour chaque résistance à partir du même fichier, pour autant que chaque série de données soit séparées par deux lignes vides, le programme utilisé (listing 1) permet de demander une nouvelle série de mesures en pressant la touche n (pour nouvelle) et cela autant de fois qu'on le désire jusqu'à la fin déclarée par la touche q (pour quitter).

3.2. Diodes

Pour les diodes, le schéma de principe est donné dans la figure 7. On voit que la diode est placée en série avec une résistance R comme charge. Ainsi, pour les mesures de la tension à ses bornes (U_{diode}), il a été nécessaire de soustraire à la tension du bus

Listing 1 – Le code de mesure avec l'INA.

```python
from ina219 import INA219, DeviceRangeError
from time import sleep
SHUNT_OHMS = 0.1
#MAX_EXPECTED_AMPS = 2.0
MAX_EXPECTED_AMPS = 0.3
ina = INA219(SHUNT_OHMS, MAX_EXPECTED_AMPS)
#ina.configure(ina.RANGE_16V, ina.GAIN_1_40MV)
ina.configure(ina.RANGE_16V)
max_power_R_mW = 230
marchearret = ''
def read_ina219():
    try:
        print('Bus Voltage: {0:0.3f}V'.format(ina.voltage()))
        print('Bus Current: {0:0.3f}mA'.format(ina.current()))
        if ina.power() < max_power_R_mW:
            print('Power: {0:0.3f}mW'.format(ina.power()))
        else:
            print('!!!! Dépassement de puissance !!!!
        Coupez tout !')
        print('Shunt Voltage: {0:0.3f}mV\n'.format(ina.
           shunt_voltage()))
        fichier = open("data.txt", "a")
        fichier.write('{0:0.3f}'.format(ina.voltage()))
        fichier.write(' {0:0.3f}'.format(ina.current()))
        iv = ina.voltage()*0.005
        fichier.write(' {0:0.3f}'.format(iv))
        ia = ina.current()*0.005
        fichier.write(' {0:0.3f}\n'.format(ia))
        fichier.close()
    except DeviceRangeError as e:
        print(e)
max_power_R_mW = int(input('Entrez la puissance max.
  de la résistace en mW : '))
while marchearret != 'q':
    marchearret = input('Tapez ENTER pour enregistrer une
      mesure, q pour quitter et n pour une nouvelle série de
         mesure : ')
    read_ina219()
    if marchearret == 'n':
        max_power_R_mW = int(input('Entrez la puissance max.
          de la résistace en mW : '))
        fichier = open("data.txt", "a")
        fichier.write('\n\n')
        fichier.close()
```

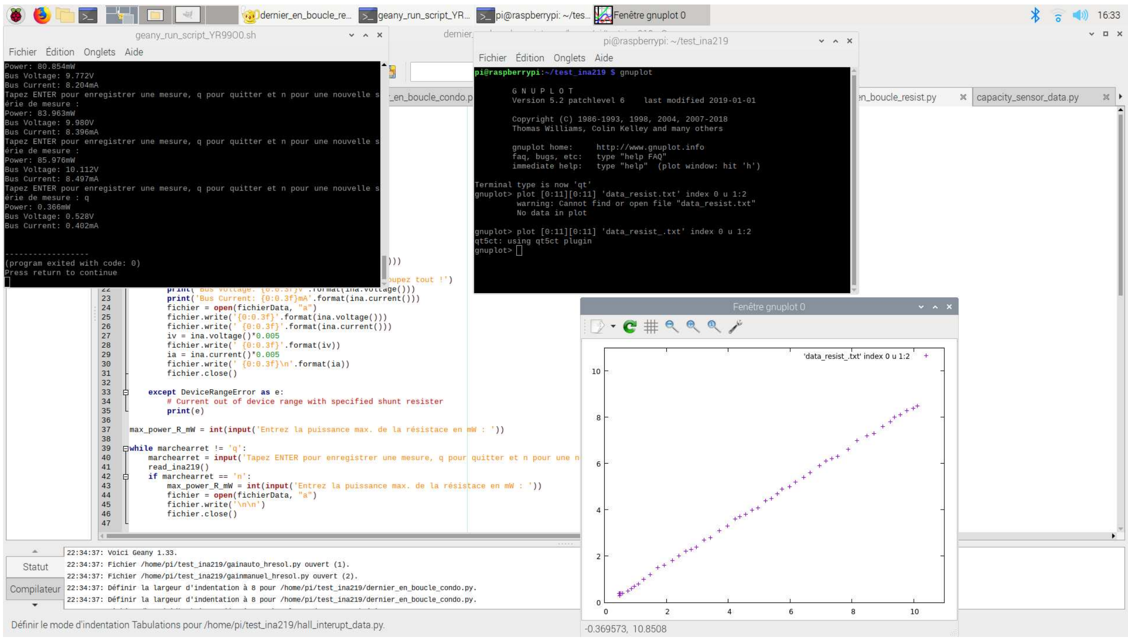


FIGURE 6 – Code, fonctionnement et graphe des mesures de résistances

(U_{bus}) la tension de la résistance (U_R), obtenue par la multiplication de la valeur de R précédemment mesurée avec le courant fourni par l'INA.

La figure 8 présente le dispositif expérimental concrètement. L'INA ainsi que les branchements sont montés sur une « breadboard ». On ne voit pas le générateur qui se trouve sur la gauche de l'image. Relevez qu'une diode classique et quatre diodes électroluminescentes (LED) ont été utilisées. Sur l'image, on voit l'une des deux diodes vertes sur la breadboard.

Le code utilisé pour la mesure des caractéristiques des diodes est celui du listing 1 auquel on a retiré les lignes 9, 15-19, 32 et 39. En effet, en se limitant à 30 mA, il n'était pas nécessaire de faire attention à la puissance dissipée dans la résistance, puisque avec 187 Ω, celle-ci se monte à $P = R \cdot I^2 = 187 \cdot (30 \cdot 10^{-3})^2 = 168{,}3\,\mathrm{mW}$, soit moins d'un quart de watt.

3.3. Condensateurs

Pour les condensateurs, le schéma de principe est donné dans la figure 9. On voit que le condensateur est placé en parallèle avec une résistance R comme charge. Comme cette résistance était très grande relativement à celle du shunt de l'INA, on peut dire que la tension du bus correspondait à celle aux bornes du condenstaeur.

La figure 10 présente le dispositif expérimental concrètement. L'INA ainsi que les branchements sont montés sur une « breadboard ». On ne voit pas le générateur qui se trouve sur la gauche de l'image. Relevez que le condensateur électrolytique présent sur l'image est branché entre la borne positive du générateur, pour la borne positive du condensateur, et la terre, borne négative du condensateur.

Figure 7 – Le RPI4, l'INA219 en mode high side, la diode et la résistance R

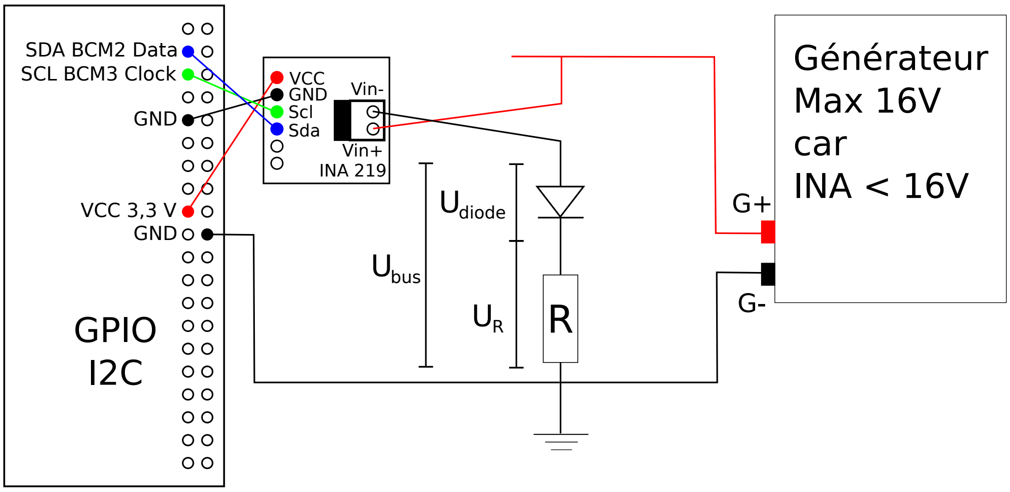


Figure 8 – Le RPI4, l'INA219, les diodes et la résistance R

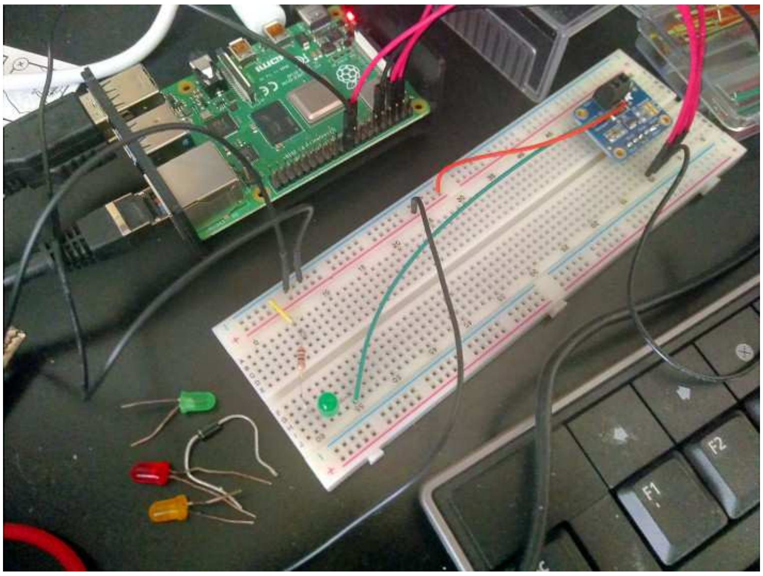


Figure 9 – Le RPI4, l'INA219 en mode high side, la diode et la résistance R

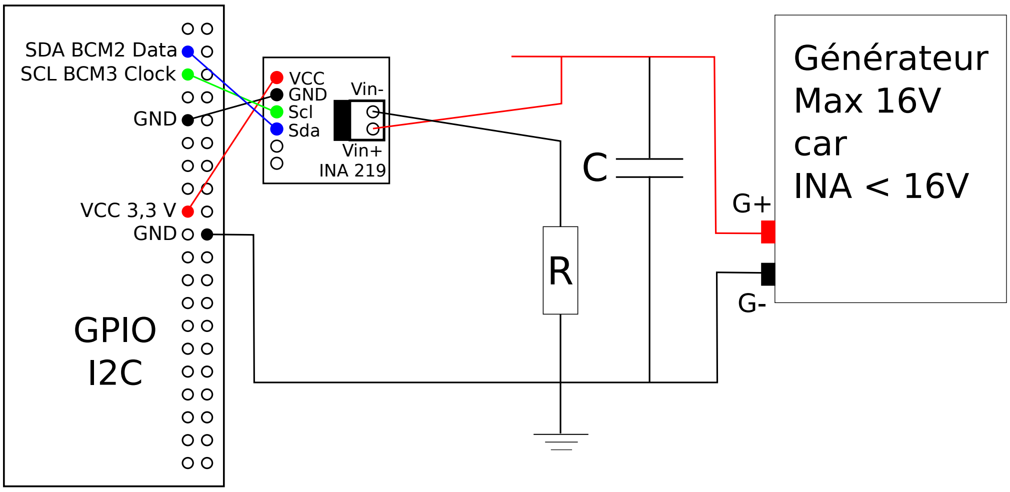
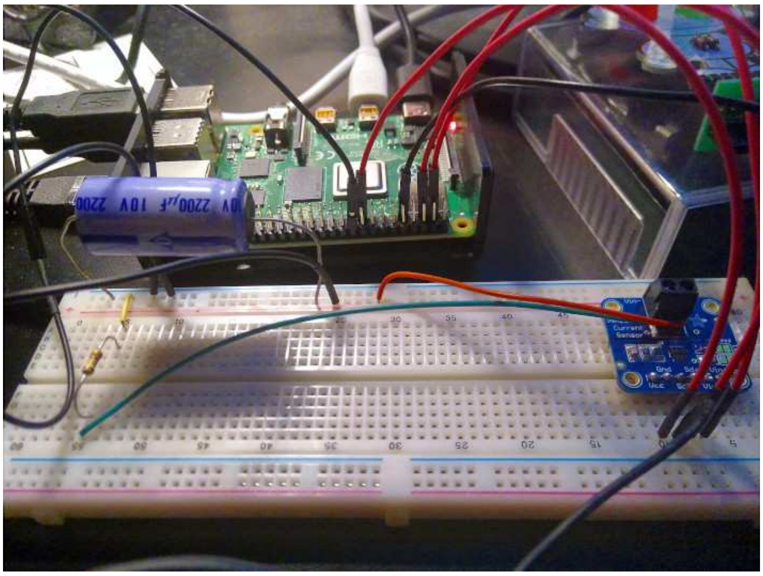


Figure 10 – Le RPI4, l'INA219, le condensateur et la résistance R



Le code utilisé pour la mesure de la décharge est celui donné au listing 2.

Dans ce cas, il n'est plus possible d'utiliser la touche `Enter` pour effectuer la mesure. L'idée est donc de lancer une boucle `while` dans laquelle on effectue une pause, donnée par la variable `intevalle_temps` de la ligne 12, juste après la prise des mesures avec l'INA, comme on le voit à la ligne 42.

Par ailleurs, attendre un certain intervalle de temps sur un système qui n'est pas en temps réel, implique que des appels système peuvent faire varier cet intervalle. C'est pourquoi, il est préférable d'horodater chaque mesure, plutôt que de postuler que chacune d'entre-elle se fait après un intervalle de temps fixe. C'est la raison de la méthode `time()` du module éponyme. À la ligne 13, on horodate (par un nombre de seconde sur Raspbian) le lancement du programme. Puis, à la ligne 32, on horodate chaque mesure et fait la différence avec l'heure de lancement du programme. Ainsi, chaque mesure apparaîtra au bon instant sur le graphe.

3.3.1. Détection de toucher

L'idée est de charger l'une des broches numérique (puisque le raspberrypi ne possède pas d'entrée analogique) jusqu'à ce qu'elle parvienne à l'état haut (GPIO.HEIGH). Le temps pour y parvenir dépend d'un condensateur et d'une résistance externe qui va limiter le temps de charge selon la relation $\tau = R \cdot C$. Ainsi, plus la résistance est grande, plus le temps sera long jusqu'à ce que l'on parvienne à l'état haut. Pendant la charge, on enclenche un compteur et on répète plusieurs fois l'opération pendant un temps donné. Si on approche le doigt de cette broche pendant sa charge, elle va se décharger à travers celui-ci et le processus de charge va recommencer. Ainsi, le temps de charge sera plus long, car la charge se fera tout de même, mais plus difficilement.

Pratiquement, on va donc utiliser deux broches.

L'une pour y placer des charges en la mettant au niveau haut avant de compter. Celle-ci sera placée relativement près de la seconde, celle qui va se charger progressivement, pour fournir lui fournir des charges. Y sera raccordée une résistance évitant un courant de décharge important dans la broche. Une valeur de $10\,\mathrm{k\Omega}$ est utilisée.

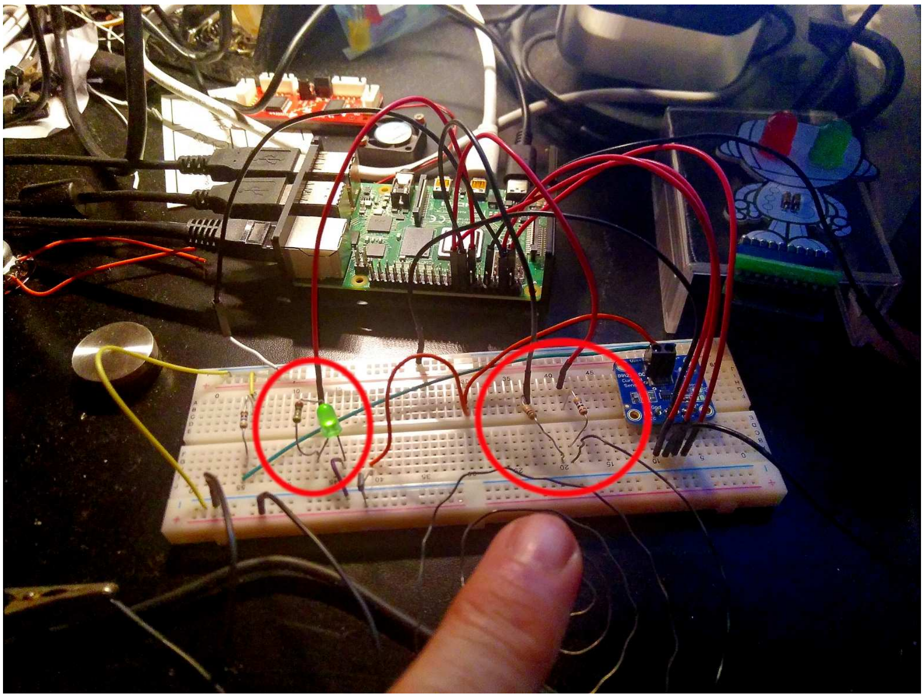
L'autre pour réaliser la charge. Une grande valeur de résistance permet de bonnes mesures. Une valeur de $10\,\mathrm{M\Omega}$ est utilisée et y est raccordé un fil conducteur enroulé en spirale pour réaliser le condensateur et permettre une bonne décharge avec le doigt. La figure 11a présente le dispositif. On y voit à droite les deux résistance dont celle de droite, la résistance réceptrice, est reliée au fil métallique enroulé en spiral pour le toucher et celle de gauche, la résistance émettrice, n'est relié à rien. On y voit aussi à gauche la résistance et la LED permettant de visualiser le toucher.

Les deux résistances sont placées l'une près de l'autre pour permettre une charge plus rapide. On peut les éloigner considérablement plus. Des tests parfaitement fonctionnels ont été fait en plaçant la résistance émettrice à l'extrémité gauche de la « breadboard », la plaque permettant de relier les composants. Mais la charge est alors bien plus lente.

Sur la figure 11b, on voit les trois « canaux » émetteur, récepteur et LED ainsi que les broches correspondant à chacun d'eux, notés avec la notation choisie dans le code du listing 3 : GPIO.BOARD.

Listing 2 – Le code de mesure avec l'INA.

```
from ina219 import INA219, DeviceRangeError
from time import sleep
import time

SHUNT_OHMS = 0.1
#MAX_EXPECTED_AMPS = 2.0
MAX_EXPECTED_AMPS = 0.3
ina = INA219(SHUNT_OHMS, MAX_EXPECTED_AMPS)
#ina.configure(ina.RANGE_16V, ina.GAIN_1_40MV)
ina.configure(ina.RANGE_16V)
marchearret = ''
intervalle_temps = 0.5 # en seconde
temps0 = time.time()
temps = 0

def read_ina219():
    global temps, intervalle_temps
    try:
        print('Bus Voltage: {0:0.3f}V'.format(ina.voltage()))
        print('Bus Current: {0:0.3f}mA'.format(ina.current()))
        print('Shunt Voltage: {0:0.3f}mV\n'.format(ina.
           shunt_voltage()))
        fichier = open("data_condo.txt", "a")
        fichier.write('{0:0.3f}'.format(temps))
        fichier.write(' {0:0.3f}'.format(ina.voltage()))
        fichier.write(' {0:0.3f}'.format(ina.current()))
        iv = ina.voltage()*0.005
        fichier.write(' {0:0.3f}'.format(iv))
        ia = ina.current()*0.005
        fichier.write(' {0:0.3f}\n'.format(ia))
        fichier.close()
        #temps = temps + intervalle_temps
        temps = time.time()-temps0

    except DeviceRangeError as e:
        # Current out of device range with specified shunt
           resister
        print(e)

input('Lancez les mesures en pressant ENTER.')

while marchearret != 'q':
    read_ina219()
    sleep(intervalle_temps) # en secondes
```



(a) Dispositif réel

(b) Schéma de principe

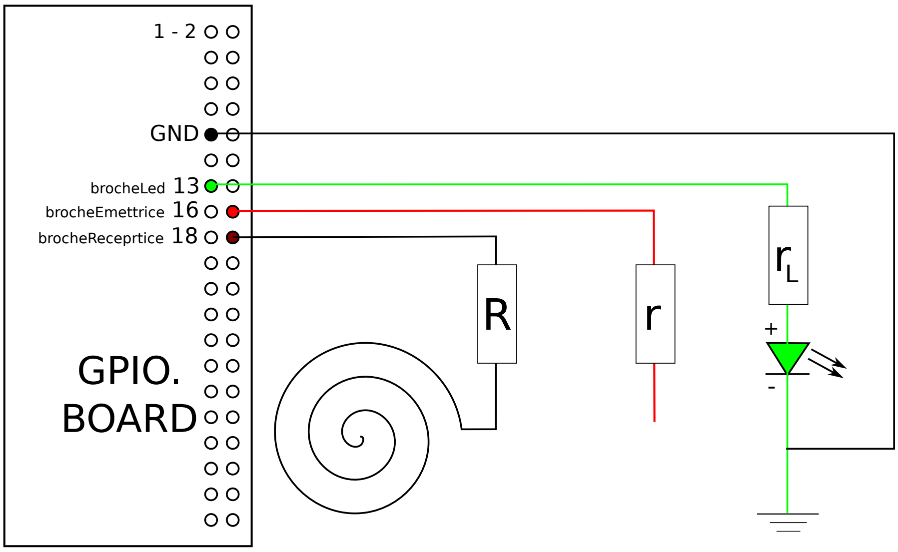


FIGURE 11 – Dispositif de détection de toucher

Le mode GPIO.BOARD se réfère à une numérotation continue des pins depuis le pin le plus proche de la carte sd (pin 1), puis celui à ses côté (pin 2), puis celui en dessous dans l'axe du GPIO du pin 1 (pin 3), etc.

Le numéro des broches du GPIO utilisées dans le code de la figure 3 correspond donc à celui indiqué à la figure 11b. Dans ce code, on voit l'initialisation du GPIO jusqu'à la ligne 6, puis aux lignes 7, 12 et 13, la déclaration des broches utilisées, suivie de leur mode d'entrée ou sortie et de leur initialisation.

Le cœur du programme se trouve dans la boucle *for* qui suit. La description est donnée en commentaires dans le code : on va incrémenter la variable *total* pendant que la broche de réception est en train de se charger, mais ne l'est pas encore, c'est-à-dire n'est pas au niveau haut. Plus on éloigne la broche d'émission, plus ce nombre est grand. Si on touche, on empêche la broche d'émission de charger la broche de réception et ce nombre augmente.

Grâce à la boucle *for* et à la variable *count*, on répète l'opération trente fois, toute les centièmes de seconde. À la ligne 32, on récupère le nombre *total* et on teste s'il est supérieur à un seuil défini par l'absence de touché. Dans le cas présent, sans touché, ce nombré était systématiquement inférieur à 5000 (il ne s'agit pas de centième de seconde). En s'approchant, il augmente pour dépasser 5000. Il n'est donc même pas nécessaire de toucher réellement pour détecter la présence du doigt.

Sur la figure 11a, on voit la LED allumée alors que le doigt ne touche pas la spirale métallique.

L'idée de ce dispositif était de tester si des modifications de la longueur de la spirale métallique ou son remplacement pas une feuille de papier d'aluminium peut conduire à des capacités et donc des temps de chargement différents.

3.4. Générateur

L'étude du générateur est classique. Nous considérerons ici la mesure de la résistance interne du générateur selon le modèle :

$$U_G - r \cdot I = R \cdot I \tag{9}$$

où r est la résistance interne du générateur.

Pour cela, nous allons utiliser exactement le même dispositif expérimental que celui de la mesure des résistances. Mais, ici, ce n'est plus la tension du bus qui nous intéresse, mais la somme de celle-ci avec la tension de shunt.

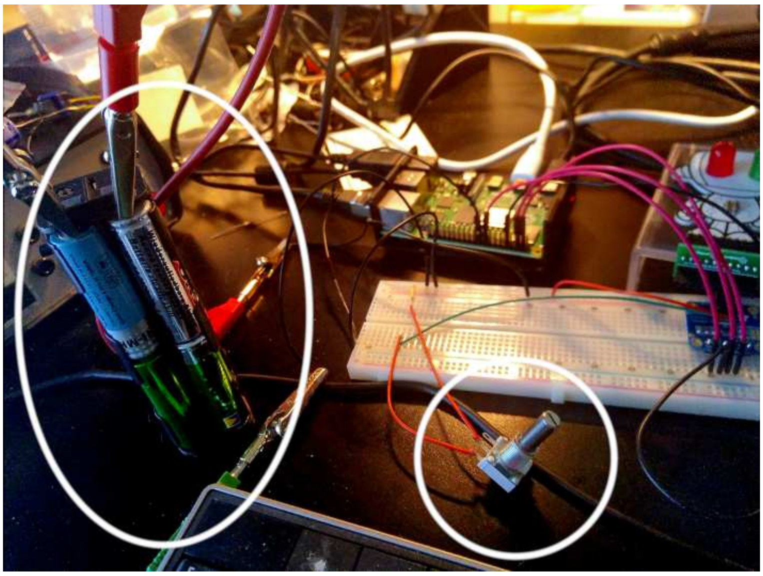
L'idée est donc de tracer le graphe de la tension au bornes du générateur en fonction du courant mesurée par l'INA, cela pour différentes charges, ici avec différentes résistances. On peut soit changer à chaque fois de résistances, soit utiliser un potentiomètre, c'est-à-dire une résistance variable.

On commence par placer le potentiomètre au maximum de sa résistance. Idéalement, si elle était infinie, aucun courant ne circulerait et la tension mesurée aux bornes du générateur serait la tension idéale ou tension à vide U_G de l'équation 9. Ensuite, on diminue progressivement la résistance. Le courant augmente donc et parallèlement la

Listing 3 – Le code de détection du toucher.

```python
import time
try:
    import RPi.GPIO as GPIO
except RuntimeError:
    print("Error importing RPi.GPIO!")
GPIO.setmode(GPIO.BOARD)
brocheLed = 13
def mesure(count):
  """ Compte les coups """
  global brocheLed
  # Définition des broches utilisées en mode GPIO.BOARD
  brocheEmettrice = 16
  brocheReceptrice = 18
  GPIO.setup(brocheEmettrice, GPIO.OUT)
  GPIO.setup(brocheReceptrice, GPIO.IN)
  GPIO.setup(brocheLed, GPIO.OUT)
  # Initialisation de la broche d'envoi du signal, en mode bas
  GPIO.output(brocheEmettrice, GPIO.LOW)
  GPIO.output(brocheLed, GPIO.LOW)
  total = 0
  for x in range(1,count):
    # On répète la mesure un nombre count de fois
    time.sleep(0.01)
    GPIO.output(brocheEmettrice, GPIO.HIGH)
    while(GPIO.input(brocheReceptrice, == False):
      # Compte le temps (en centième de seconde) pendant lequel la broche de réception ne se charge pas. Plus on éloigne la broche d'émission, plus ce nombre est grand. Si on touche, on empêche la broche d'émission de charger la broche de réception et ce nombre augmente.
      total += 1
    GPIO.output(brocheEmettrice, GPIO.LOW)
  return total
def main():
  while True:
    touch = mesure(30)
    print("Touché: %d" % touch)
    if(touch > 5000):
      GPIO.output(brocheLed, GPIO.HIGH)
    else:
      GPIO.output(brocheLed, GPIO.LOW)
    time.sleep(0.5)
if __name__ == "__main__":
  main()
```

Figure 12 – Le RPI4, l'INA219, le générateur et le potentiomètre



partie $r \cdot I$ de l'équation 9, diminuant d'autant la tension à vide. La tension mesurée diminue donc.

On voit sur l'image de la figure 12 le générateur constitué de huit accumulateurs de 1,2 V à gauche et à droite le potentiomètre.

Le code nécessaire est donc exactement le même que celui utilisé pour les résistances, à l'exception ...

3.5. Moteur

Au paragraphe 2.5, page 4, nous avons vu les principes généraux de l'étude d'un moteur CC. Voyons-en maintenant la réalisation concrète. Trois expériences vont être réalisées.

1. La détermination de la résistance interne du moteur. Pour cela, il faut bloquer le moteur. En effet, si la puissance mécanique qu'il développe est nulle, et c'est le cas si le moteur ne tourne pas, comme $P_{mec.} = E \cdot I$, comme le courant n'est pas nul, cela signifie que la tension contre-électromotrice E est nulle. L'équation 3, page 5, se résume alors à l'expression donnée par l'équation 10.

$$U = r_i \cdot I \tag{10}$$

   Il suffit donc de mesurer la tension et le courant dans le moteur en mode bloqué. Évidemment, il faut être prudent, car en bloquant le moteur, la résistance interne étant généralement faible, le courant peut devenir très important et endommager le moteur.

   La réalisation des mesures est ici très simple, puisqu'on se trouve dans le même cas que la mesure de résistances avec l'INA219. Il suffit de mettre le moteur à leur place en se référant à la figure 3, page 7, tout en veillant à ne pas dépasser la tension maximale admissible pour le moteur (suivant les cas de 4 à 12 V).

2. La détermination de la constante k du moteur dans la relation 4 de la page 5.

   Connaissant la résistance interne du moteur, la mesure de la tension U qui lui est appliquée et, simultanément, du courant I qui le traverse, permet d'obtenir l'équation 11 donnant la tension contre-électromotrice E' grâce à l'équation 3, page 3.

$$E' = U - r_i \cdot I \tag{11}$$

   La tension et le courant sont mesurés par directement par l'INA219 comme décrit au point précédent, sauf que le moteur n'est alors plus en mode bloqué.

   Reste à mesurer la vitesse de rotation angulaire ω. Le principe est simple. On va utiliser un compte tour. On compte un nombre donné de tours et le temps pour les faire. Par division, on obtient la fréquence en nombre de tours par seconde et en multipliant par $2 \cdot \pi$, on obtient la vitesse angulaire ω.

   Pour faire cela, il faut pouvoir compter compter le nombre de tours. Cela se fait en utilisant un aimant qui passe devant une sonde de Hall numérique. Une telle sonde ne mesure pas le champ magnétique en continu. Où plutôt, elle le fait, mais renvoie une valeur binaire quand le champ est supérieur à une valeur donnée et la valeur binaire opposée quand celui-ci est inférieur à cette valeur. Dans notre cas, la sonde LittleFuse 55100 [5] qui est un capteur à effet Hall, renvoie une valeur bas quand la sonde est à plus de 130 Gauss ou à moins de 12,5 mm de l'aimant de référence LittleFuse donné dans la documentation. Évidemment, on pourrait utiliser une sonde de Hall analoqique. Nous le ferons plus tard. Mais ici, il faut comprendre l'avantage d'une sonde numérique quand on sait que si un Arduino, par exemple, dispose d'entrées analogiques, Raspberry ne dispose que d'entrées numériques. Ce type de sondes permet donc de se passer d'un convertisseur analogique-numérique et peut être utilisée directement sur les broches du Raspberry.

   Deux problèmes se sont présentés, cependant. Le premier dû au problème bien connu des instabilités des broches laissées en l'air, généralement résolu par l'utilisation de résistances pull-up ou pull-down. Le second dû au temps de passage de l'aimant devant la sonde.

   Avant de présenter ces deux problèmes, présentons le dispositif complet. La figure 13 est le schéma de principe correspondant à la figure 14 qui présente le montage. On y voit à gauche le moteur, au-dessus à gauche l'aimant et à droite le compte tour magnétique, scotché au montant du support de l'écran.

   Il faut noter, comme on le voit partiellement sur la figure 14, que le compte tour magnétique, soit la sonde de Hall LittleFuse, est équipée de trois longs fils permettant de l'approcher facilement du moteur. De plus, le couleurs de ces fils sont très claires, puisque le fil rouge correspond à la tension d'alimentation de 5 V (La tension minimale de 3,8 V nécessitait l'utlisation de la broche GPIO numéro deux,

5. Voir `https://www.distrelec.ch/Web/Downloads/_t/ds/littelfuse_55100_eng_tds.pdf?pid=\protect\T1\textdollarproduct.code` pour la sonde 3M à trois fils.

Figure 13 – Le moteur, l'aimant et le compte tour magnétique

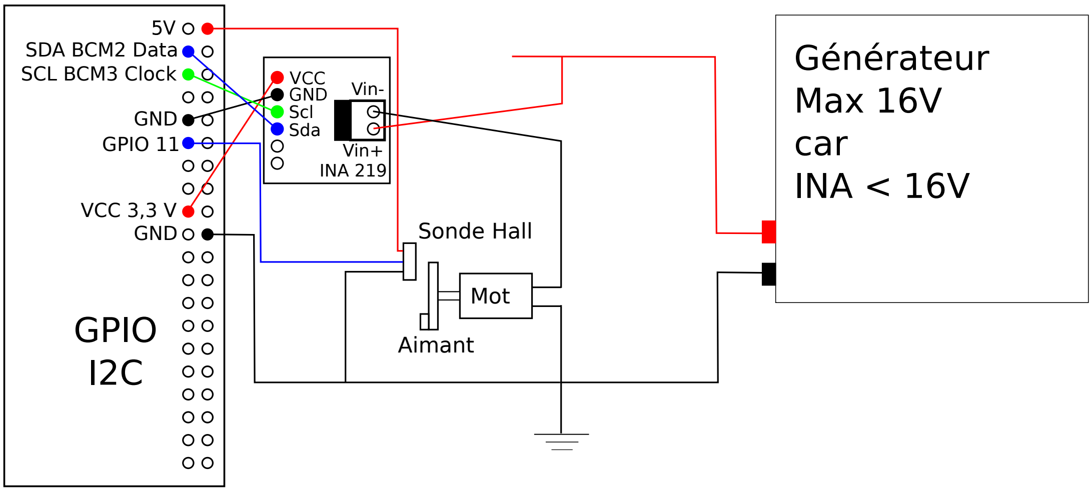
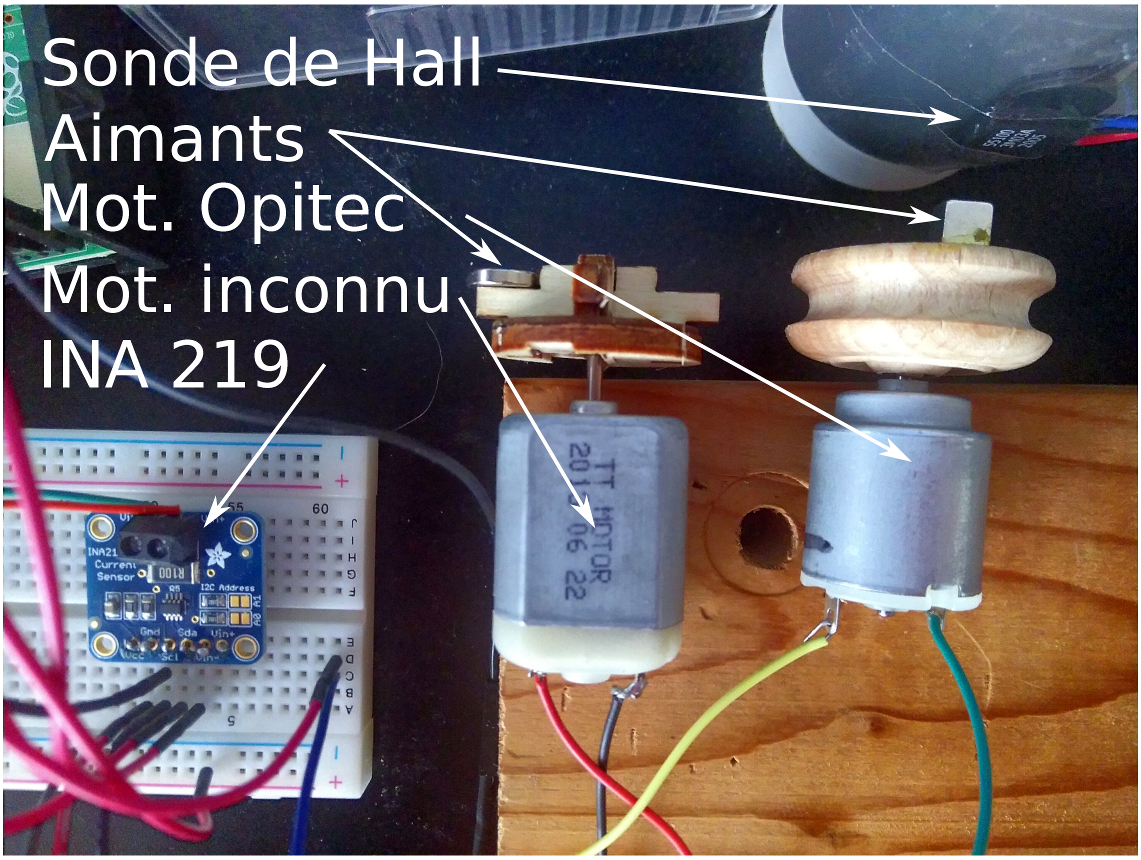


Figure 14 – Les moteurs, les aimants et le compte tour magnétique ou sonde de Hall



puis que la première broche ne délivre que 3,3 V), le fil noir à la masse et le fil bleu aux données, comme indiqué sur le schéma 13 de principe.

Le code correspondant est donné dans le listing 4.

Listing 4 – Le code de mesure avec le moteur, l'INA et la sonde de Hall.

```python
from ina219 import INA219, DeviceRangeError
from time import sleep
import time
import math
try:
    import RPi.GPIO as GPIO
except RuntimeError:
    print("Error importing RPi.GPIO!")
GPIO.setmode(GPIO.BOARD)
SHUNT_OHMS = 0.1
MAX_EXPECTED_AMPS = 0.3
ina = INA219(SHUNT_OHMS, MAX_EXPECTED_AMPS)
ina.configure(ina.RANGE_16V)
marchearret = ''
r = 3.17 # résistance interne du moteur (ohm)
temps = 0
def read_ina219():
    global r
    brocheHall = 11
    # On est obligé de mettre à up la broche de Hall, car sinon, elle
    # est aléatoirement fixée.
    GPIO.setup(brocheHall, GPIO.IN, pull_up_down=GPIO.PUD_UP)
    nbtours = 100
    n = 0
    drapeau = 'declanche'
    temps0 = time.time()
    while (n <= nbtours):
        etatBrocheHall = GPIO.input(brocheHall)
        if (etatBrocheHall == GPIO.LOW):
            if (drapeau != 'enclanche'):
                n = n + 1
                #print('Enclanché')
                drapeau = 'enclanche'
        else:
            #print('-')
            drapeau = 'declanche'
        time.sleep(0.001)
    temps = time.time()-temps0
    frequence = nbtours/temps
```

```
    omega = 2*math.pi*frequence
    print(temps)
    print('Fréquence mot. : {0:0.3f}trs/s'.format(frequence))
    print('Vit. angulaire : {0:0.3f}rad/s'.format(omega))
    try:
        print('Bus Voltage: {0:0.3f}V'.format(ina.voltage()))
        print('Shunt Voltage: {0:0.3f}mV'.format(ina.shunt_voltage()))
        print('Bus Current: {0:0.3f}mA\n'.format(ina.current()))
        eprime = ina.voltage()-r*(ina.current()/1000)
        fichier = open("data_mot_hall.txt", "a")
        fichier.write('{0:0.3f}'.format(omega))
        fichier.write(' {0:0.3f}'.format(eprime))
        fichier.write(' {0:0.3f}'.format(ina.voltage()))
        fichier.write(' {0:0.3f}'.format(ina.current()))
        iw = frequence * 0.001/temps
        fichier.write(' {0:0.3f}'.format(iw))
        iv = ina.voltage()*0.005
        ia = ina.current()*0.005
        ieprime = iv + r*(ina.current()/1000)*(0.1/r+ia/(ina.current()/1000))
        fichier.write(' {0:0.3f}'.format(ieprime))
        fichier.write(' {0:0.3f}'.format(iv))
        fichier.write(' {0:0.3f}\n'.format(ia))
        fichier.close()
    except DeviceRangeError as e:
        # Current out of device range with specified shunt resister
        print(e)
while marchearret != 'q':
    marchearret = input('Tapez ENTER pour enregistrer une mesure, q pour quitter et n pour une nouvelle série de mesure : ')
    read_ina219()
    if marchearret == 'n':
        fichier = open("data_mot_hall.txt", "a")
        fichier.write('\n\n')
        fichier.close()
```

Le premier problème a été que la tension de sortie haute de la sonde de hall n'est pas fixée. La documentation précise « sinking output » pour la sortie haute, alors que pour la sortie basse, il est dit « 0,4 V @ 20 mA ». Il faut donc utiliser une résistance pull-up, c'est-à-dire une résistance branchée sur l'alimentation de 5 V. En effet, sans courant, la tension aux bornes de la résistance est nulle est la tension de la broche est à son niveau haut de 5 V. On peut le faire avec une résistance réelle, mais ce n'est pas nécessaire ici, car la broche utilisée peut être maintenue à son niveau haut de manière logicielle grâce à l'instruction de la ligne 22 du code du listing 4 où on voit la configuration de la *brocheHall* avec `pull_up_down=GPIO.PUD_UP`.

Le second problème ne fut pas évident à détecter, car relativement aléatoire. Ce qu'il faut savoir, c'est que la fréquence de rafraîchissement de la sonde de Hall est de 10 kHz, ce qui signifie un intervalle temporel d'un dixième de milliseconde. Or, le temps de passage de l'aimant devant la sonde est bien supérieur. Dans l'idéal, il faudrait donc faire un pause assez longue entre deux mesures pour permettre à l'aimant de ne plus solliciter la sonde de façon à ne compter qu'une seule fois chaque passage de celui-ci devant celle-ci. Or, comme il était prévu de faire varier la vitesse de rotation du moteur, il n'était pas envisageable de recalculer au mieux ce temps à chaque fois.

La solution est venu du constat mis en évidence par l'affichage de l'état de la sonde à chaque pas de la boucle de mesure : pendant le passage devant la sonde, si dix mesures étaient faites, elles étaient systématiquement toujours dans l'état bas. Ainsi, il suffisait de lever un drapeau au premier passage au niveau bas devant la sonde pour incrémenter l'indice du nombre de tours et d'interdire cette incrémentation tant que le niveau de la sonde n'était pas retourné au niveau haut, signifiant que l'aimant n'était plus devant la sonde.

Le code correspondant est celui des lignes 23 à 40 du listing 4. À la ligne 23 est défini le nombre de tours pendant lesquels on va faire la mesure de temps, initialisée à la ligne 26 et enregistrée à la ligne 38. L'état de la sonde est relevée à la ligne 28 et on bloque l'augmentation du nombre de tours pendant le passage de l'aimant devant la sonde aux lignes 29 à 36. À la ligne 37, on attends ensuite une fraction de seconde avant de relancer la mesure (mais cela ne doit pas être absolument nécessaire). Enfin, on calcule la fréquence et la vitesse angulaire.

La suite du code permet d'enregistrer les différentes grandeurs de l'équation 11, page 19, que sont la vitesse angulaire ω, la tension contre-électromotrice E', la tension aux bornes du moteur U, le courant I et leurs incertitudes, en relevant que le courant et son incertitude sont mesurés en mA.

La procédure de mesure est ainsi la suivante. On lance le programme, puis on met le moteur à sa vitesse maximale et en appuyant sur la touche *Enter*, on lance le programme. Une mesure est immédiatement faite. À l'aide du potentiomètre du générateur, on diminue un peu la tension et donc la vitesse du moteur et on appuie à nouveau sur *Enter* pour une nouvelle mesure. On procède ainsi jusqu'à l'arrêt du moteur.

3. La détermination de la puissance maximale que peut fournir le moteur.

   Rappelons dans un premier temps que l'équation 7, page 5, nous donne une expression théorique nous permettant de calculer le courant dans le moteur pour la puissance maximale qu'il peut fournir. Ce courant dépend de la tension appliquée au moteur et de sa résistance interne. Avec la mesure de ces deux grandeurs, on peut donc prévoir le courant et à l'aide de l'équation 6, page 5, déterminer la puissance maximale.

Une autre manière d'obtenir ce courant est de tracer le graphe de la puissance mécanique du moteur en fonction du courant. Selon l'équation 6, il s'agit d'une fonction quadratique et on devrait obtenir une parabole dont le maximum correspond à la puissance maximale du moteur pour la tension donnée.

Deux manières de déterminer cette courbe se présentent alors :

a) en calculant la puissance à l'aide des mesures de la tension aux bornes du moteur, du courant et de la résistance, détermination qu'on nommera « électrique » et

b) en calculant la puissance « mécanique » du moteur via son couple C et sa vitesse de rotation angulaire ω. En effet, on a que :

$$P_{mec} = C \cdot \omega = F \cdot r \cdot \omega \tag{12}$$

où, F est la force exercée sur l'axe de rotation et r la distance du centre de rotation à la position à laquelle s'exerce la force.

La mesure du rayon se fait avec un pied à coulisse et celle de la force en enroulant une ficelle sur une petite poulie attachée à l'axe du moteur à laquelle on met un poids connu pendant dans le vide d'un côté et de l'autre en reliant celle-ci à un dynamomètre. Ainsi, par différence du poids et de la mesure du dynamomètre, on obtient la force totale de frottement exercée sur la poulie par la ficelle. Le bras de levier du moment de force est alors le rayon de la poulie.

4. Résultats

4.1. Résistances

Le résultat des mesures effectuées sur cinq résistance de 50, 100, 180 , 500 et 1200 Ω est donné à la figure 15.

Les résistances sont identifiées par leur code de couleur, puis mesurées à l'aide d'un ohmmètre dont la précision est de 1% ± 2 digits, valeurs reportées sur le graphe de la figure 15.

L'écart entre la valeur mesurée par l'ohmmètre et l'INA est reporté sous $e_{mes.}$ dans la figure 15.

Les incertitudes sur les mesures de la tension et du courant ont été choisies selon les spécifications de l'INA219 qui sont données à 0,5% de la valeur (voir l'annexe A, page 42), tant pour le courant que pour la tension. Mais d'autres sources mentionnent 5%, ce qui me paraît beaucoup au vu des écarts obtenus. Sur le graphe de la figure 15, on ne voit pas les barres d'incertitudes qui sont trop petites. Quoi qu'il en soit, l'objectif n'est pas ici d'analyser finement l'incertitude de façon à déterminer la meilleures valeur de résistance possible.

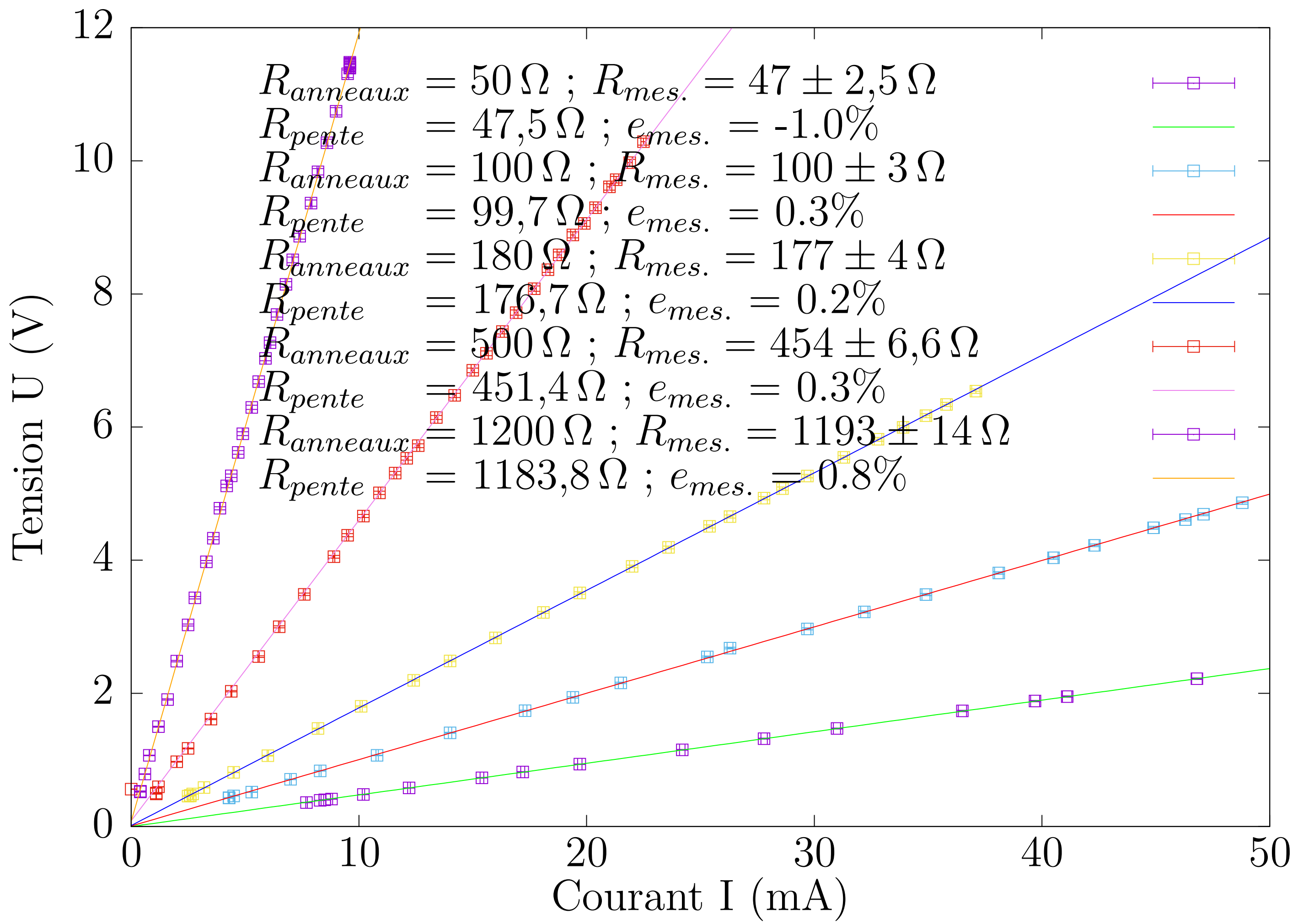


FIGURE 15 – Mesure de résistance avec l'INA219

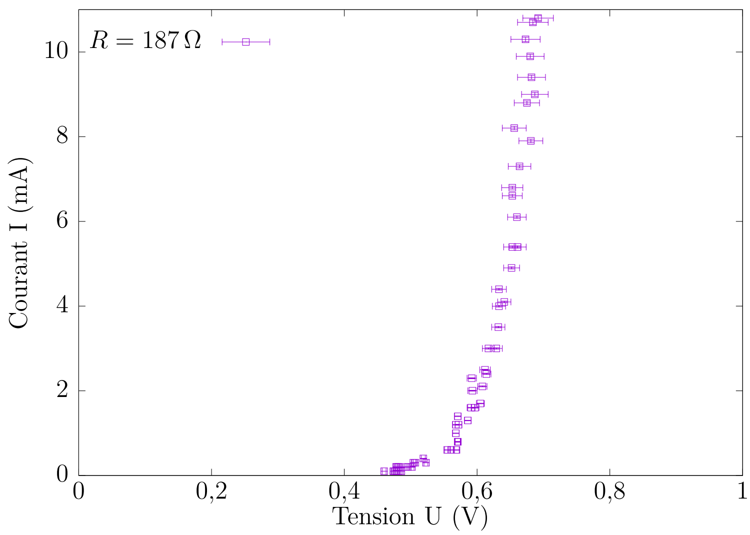


FIGURE 16 – Caractéristique courant-tension d'une diode avec l'INA219

4.2. Diodes

La figure 16 présente la première série de mesure faite avec une diode en mode direct. L'absence de mesures avant 0,5 V est due à mon générateur fait maison, bien stabilisé, mais présentant une tension minimale d'un demi volt.

La tension de seuil à 0,6 V est très bien marquée.

Sur la figure 17, on voit les caractéristiques de plusieurs diodes électroluminescentes (Light Emiting Diode : LED) en mode direct. La résistance donnée est en série avec la diode pour limiter le courant.

4.3. Condensateurs

La figure 18b est un test de décroissance de la tension au bornes de la résistance seule quand on coupe le courant. Aucun condensateur n'a été placé en parallèle des résistances et pourtant on observe bien une décroissance avec cependant une résonance. Cette décroissance est indépendante de la valeur de la résistance. Elle vient d'un condensateur du générateur. En effet, la manière de procéder pour couper la tension ayant simplement été de couper le générateur, le circuit passant par le générateur restait branché.

Ainsi, il est important de poser un interrupteur permettant de découpler entièrement le générateur du circuit. En procédant de cette manière, la décroissance n'a plus présenté cette résonance comme on peut le voir sur la figure 19a.

On voit par comparaison des figures 19b et 19c que la précision apportée par la méthode `time()` du module `time` n'est pas sans qualité au vu de la qualité de la courbe de régression. Mais, l'écart entre la valeur mesurée de la capacité et la valeur annoncée reste grand.

Cela peut avoir plusieurs explications.

— L'incertitude sur la valeur donnée sur la résistance. Il faut savoir que l'incertitude

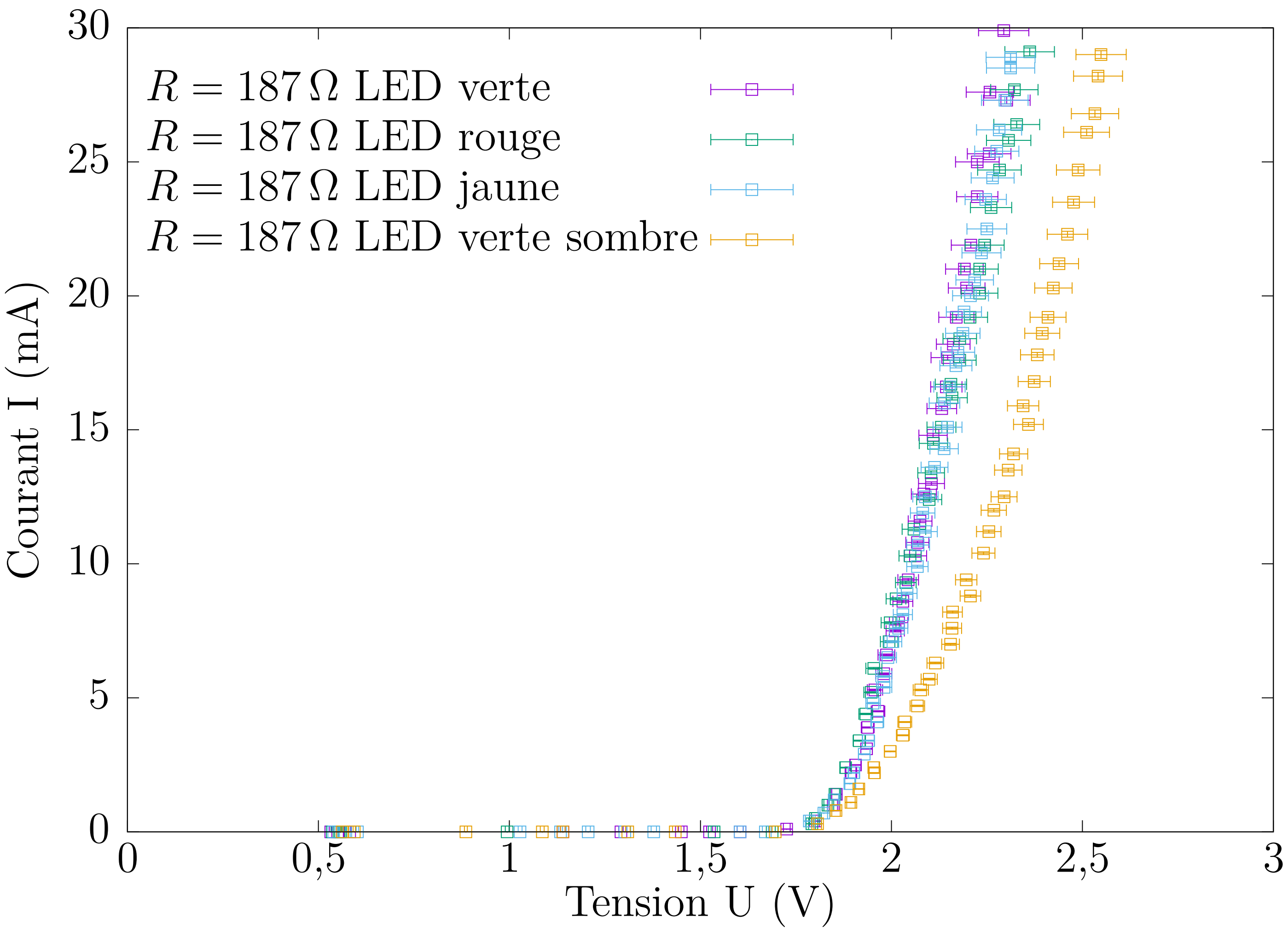
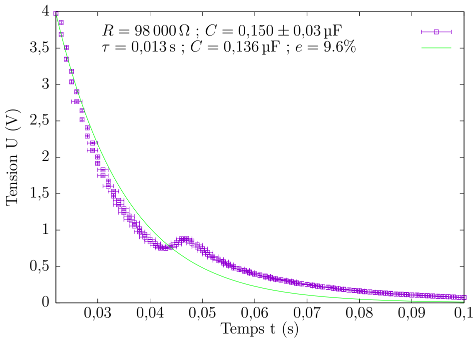
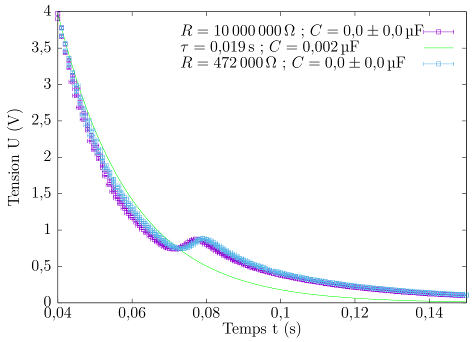


FIGURE 17 – Caractéristique courant-tension d'une LED avec l'INA219



(a) Mesure de chute de tension capacitive avec l'INA219



(b) Mesure de chute de tension capacitive avec l'INA219

FIGURE 18 – L'influence d'un générateur.

minimale sur la valeur de la capacité d'un condensateur est de 20%, mais on trouve des condensateurs à 30, 40 voir 50%. Or, il n'a pas été possible de déterminer la valeur de cette incertitude pour les condensateurs utilisés. Elle a donc été annoncée à 20%.

— Si les condensateurs utilisés l'ont été sous des conditions normales de température, tension, ... il n'avaient pas été utilisés depuis plus de vingt ans. Cela peut-il avoir eu une influence ?

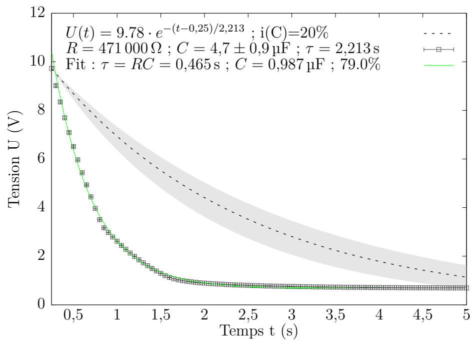
Par ailleurs, il est très important que le courant de décharge ne soit pas trop faible. Si c'est le cas, on se retrouve dans une situation où la précision de la mesure sur le courant implique l'impression d'une tension résiduelle mesurée pas l'INA.

Pour maximiser le courant, avec des résistances de ¼W, la résistance à utiliser avec une tension de 9 V est de :

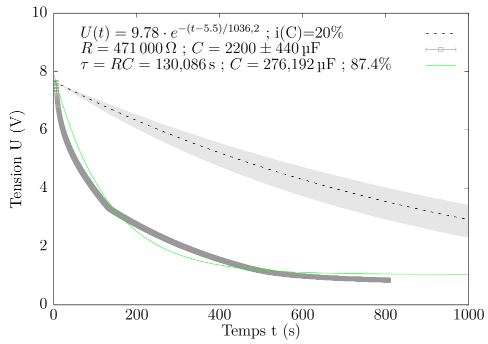
$$P = \frac{U^2}{R} \rightarrow R = \frac{U^2}{P} = \frac{9^2}{0,25} = 324\,\Omega$$

au minimum. Sur la figure 22, la résistance était de 455 Ω pour une capacité de 2200 µF et, pour une capacité, l'écart entre la valeur de capacité mesurée pas décroissance et la valeur annoncée sur la résistance, soit environ 40% est correct. Alors que sur la figure 19b, on voit qu'avec une résistance de 471 kΩ, non seulement l'écart est de plus de 80%, mais une tension résiduelle autour de 1 V est présente. Avec une tension initiale d'environ 8 V et cette résistance, le courant était au maximum de 0,02 mA, soit au-delà de ce que l'INA peut honnêtement mesurer. On peut donc imaginer que le courant effectivement mesuré par l'INA ait été trop important, donc aussi la tension de shunt et par conséquent la tension de bus systématiquement trop faible. Ce qui est compatible avec ce qu'on voit sur la figure 19b.

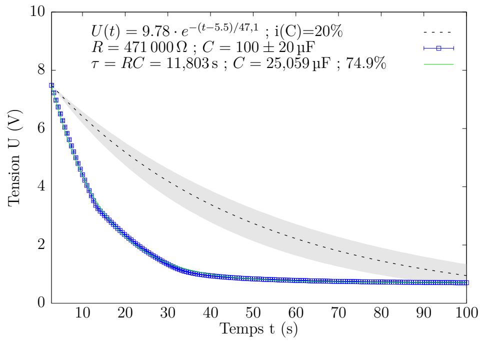
On voit sur la figure 21 une série de mesures réalisées avec une résistance de 455 Ω et un condensateur de 100 µF, puis en remplacement de cette résistance, avec un potentiomètre réglable dont on a changé la résistance de plus de 1 kΩ à 100 Ω. Dans cette plage de résistances, les mesures sont cohérentes, même si l'écart est grand.



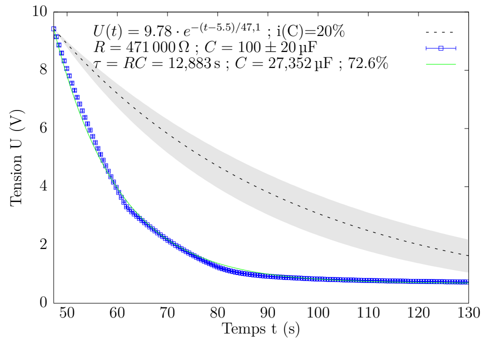
(a) Chute de tension capacitive avec l'INA219
$R = 471\,\mathrm{k\Omega}$; $C = 4{,}7\,\mathrm{\mu F}$



(b) Mesure de chute de tension capacitive avec l'INA219
$R = 471\,\mathrm{k\Omega}$; $C = 2200\,\mathrm{\mu F}$



(c) Mesure de chute de tension capacitive avec l'INA219
$R = 471\,\mathrm{k\Omega}$; $C = 100\,\mathrm{\mu F}$; python *time.time* est utilisé ici



(d) Mesure de chute de tension capacitive avec l'INA219
$R = 471\,\mathrm{k\Omega}$; $C = 100\,\mathrm{\mu F}$; python *time.time* est utilisé ici

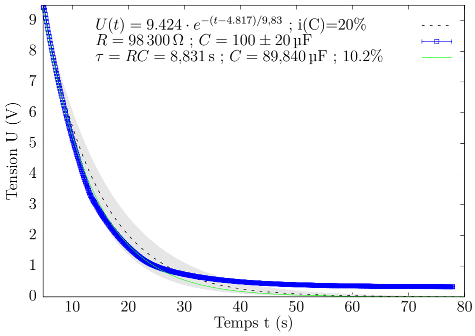
FIGURE 19 – Premières séries de mesures.

En ligne traitillée se trouve la courbe correspondant à $455\,\Omega$ et $100\,\mathrm{\mu F}$. L'écart à la valeur de la capacité calculée sur la base des mesures pour cette résistance et donnée dans la légende. La zone grisée correspond à une tolérance de ±20%. L'écart de plus de 60% est grand mais du bon ordre de grandeur.

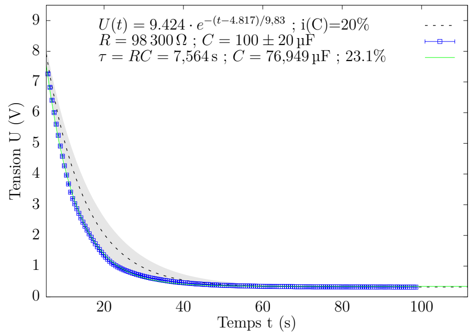
4.4. Générateur

Les mesures présentés à la figure 24a montrent la décroissance de la tension aux bornes du générateur constitué par huit accumulateurs NiMH d'environ 1,2 V. Le courant étant affiché en mA, la pente de la courbe a été multipliée par mille et c'est la valeur en Ω qu'on lit dans l'équation de la droite, soit 6,5 Ω, comme résistance interne.

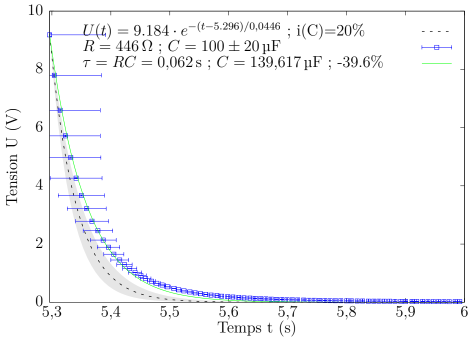
Les mesures présentés à la figure 24b montrent la diminution de la tension aux bornes d'un accumulateur NiMH de 9 V. La résistance interne de 214,6 Ω est importante comparativement à celle de l'ensemble des huit accumulateurs de la figure 24a. Les raisons



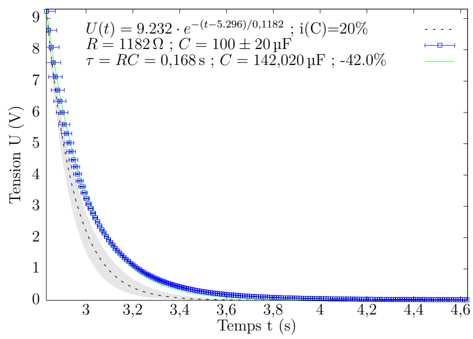
(a) Mesure de chute de tension capacitive avec $R = 98{,}3\,k\Omega$



(b) Mesure de chute de tension capacitive avec $R = 98{,}3\,k\Omega$; déchargé



(c) Mesure de chute de tension capacitive avec $R = 446\,\Omega$



(d) Mesure de chute de tension capacitive avec $R = 1182\,\Omega$

Figure 20 – Un condensateur caractéristique $C = 100\,\mu F$ et python *time.time*.

de cette différence sont encore à déterminer.

Sur la figure 25, on voit les résultats pour la mesure de quatre accumulateurs NiMH de 9 V, le même ensemble de huit accu. NiMH 1,2 V que pour les résultats de la figure 24a, et une pile AAA R03 1,5 V. Les accumulateurs de 9 V n'étaient pas entièrement chargés. Le premier présentait 3,58 V à vide, le second 6,13 V, le troisième 6,89 V et le dernier 1,23 V.

Sur la figure 26a sont présentées les mesures de décroissance de la tension de six générateurs de 9 V préalablement chargés. Malheureusement, on ne peut comparer avec ceux de la figure 25, car tous n'étaient pas présents et je n'ai pas pris note du nom de chaque générateur. Mais on voit évidemment l'effet de la charge de chacun.

4.5. Moteur

4.5.1. Résistance interne

Commençons par la détermination de la résistance interne du moteur. Deux moteurs CC vont être utilisés. Un petit moteur très bon marché acheté chez Opitec, dont la documentation technique mentionne :

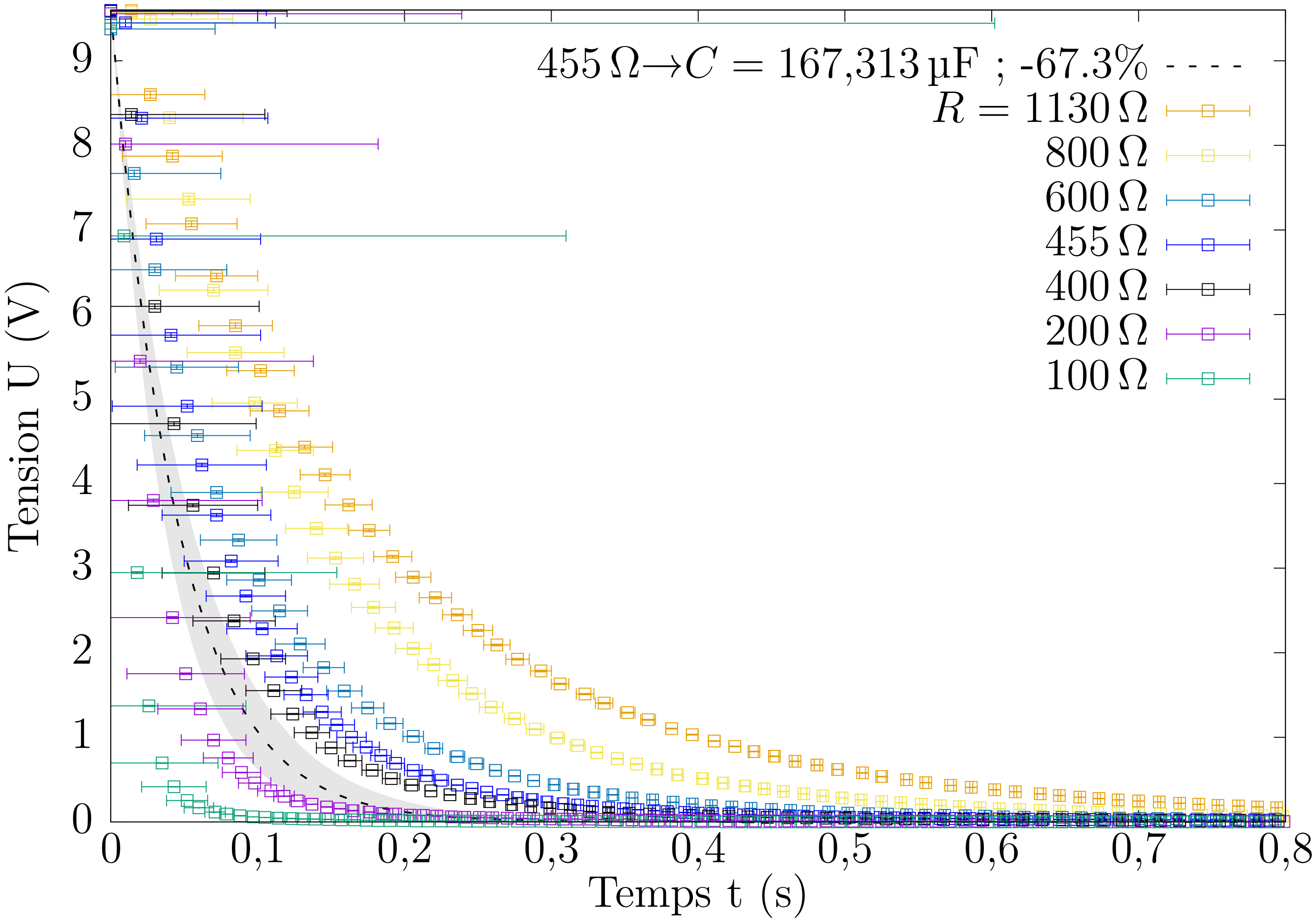


FIGURE 21 – Chute de tension d'un condensateur $C = 100 \pm 20\,\mu\text{F}$ pour de multiples résistances

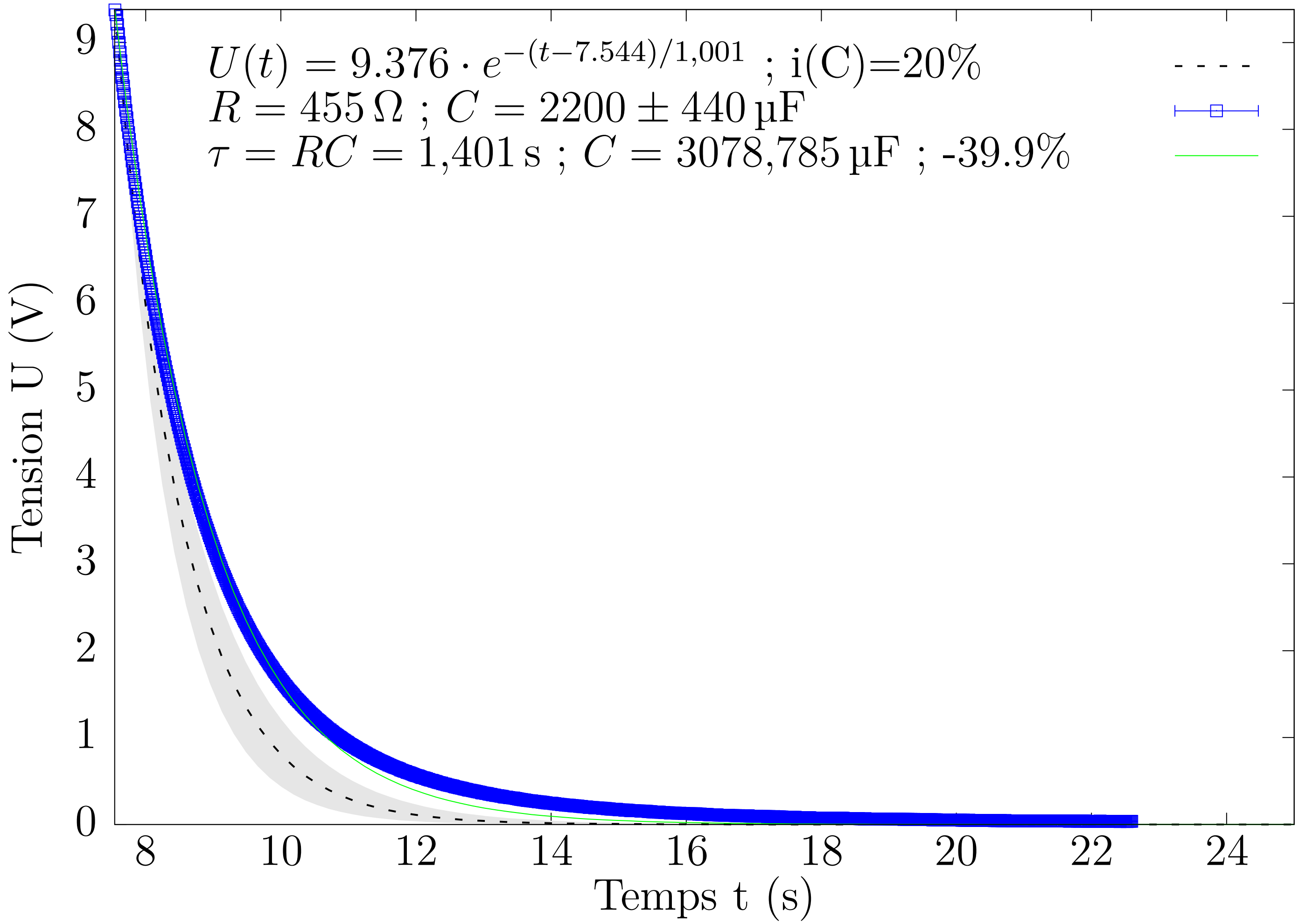
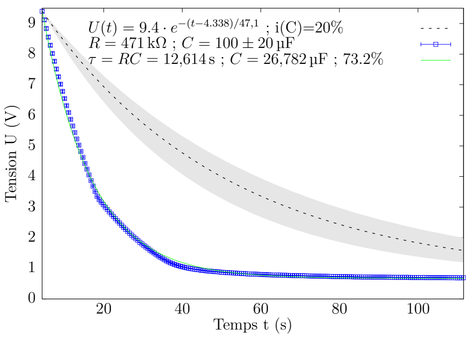
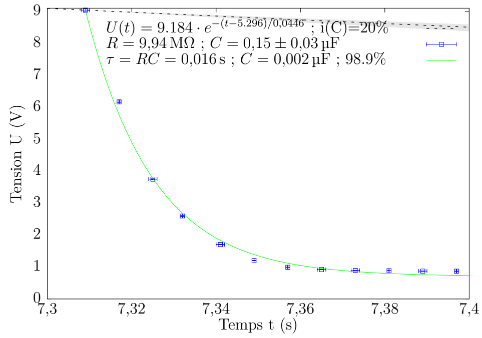


Figure 22 – Mesure de chute de tension capacitive avec l'INA219
$R = 455\,\Omega$; $C = 2200\,\mu$F ; python *time.time* ; déchargé

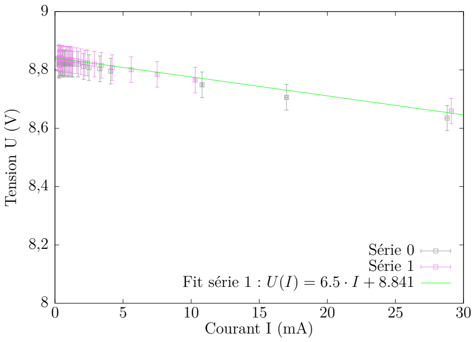


(a) Mesure de chute de tension capacitive. Pour $R = 471\,\mathrm{k\Omega}$ $C = 100\,\mu\mathrm{F}$ et *time.time* déchargé

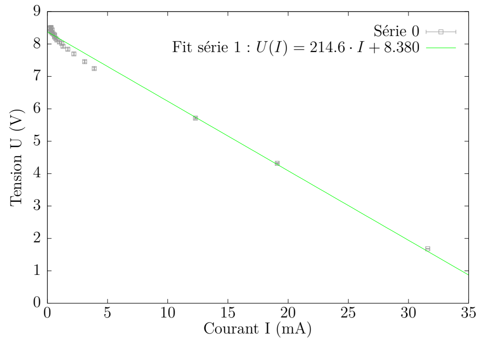


(b) Mesure de chute de tension capacitive avec l'INA219
$R = 9{,}94\,\mathrm{M\Omega}$; $C = 0{,}15\,\mu\mathrm{F}$; python *time.time* est utilisé ici

FIGURE 23 – Cas problématiques.



(a) Mesure de chute de tension avec huit piles 1,2 V, potentiomètre 46 kΩ 1 W



(b) Mesure de chute de tension avec un accu. de 9 V, potentiomètre 46 kΩ 1 W

FIGURE 24 – Premières mesures.

```
Moteur R20 ou R140
1,5 - 4,5V DC
20g diam 21 mm, 25 mm long, diam 2 mm
1,5V - 8400 trs/min - 230 mA
3V - 14200 trs/min - 280 mA
```

et un moteur d'origine inconnue et sans documentation.

Après plusieurs essais de mesure de la tension et du courant avec l'INA en mode de mesure du courant maximum de 300 mA en augmentant la tension de 0 à moins de 2 V de façon à ne pas dépasser le courant maximum de l'INA et avec le moteur inconnu en mode bloqué, le constat est que les mesures sont très peu linéaires au début est à la fin. Il existe bien une zone où une certaine linéarité est présente, mais ce n'est pas évident.

En travaillant le problème, on remarque qu'en mode non bloqué, ce moteur ne démarre pas immédiatement. Par ailleurs, il semble non polarisé, c'est-à-dire qu'on peut inverser ses pôles et il fonctionne toujours correctement.

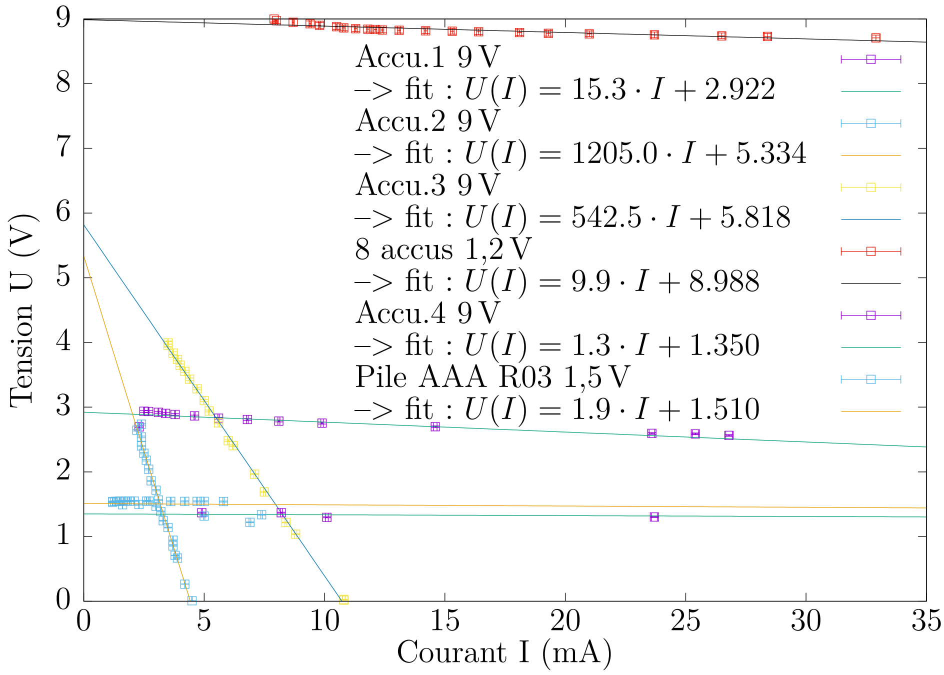


Figure 25 – Mesure de chute de tension avec plusieurs accu., une pile et un potentiomètre 1,2 kΩ 1 W

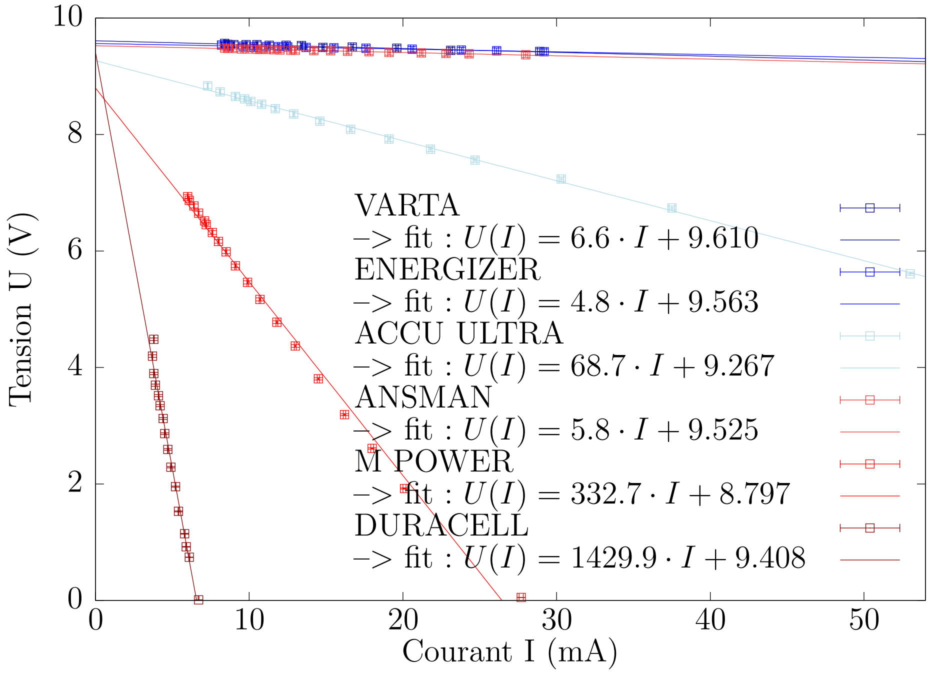
Pour le moteur d'Opitec, bizarrement, il semble polarisé. Directement branché sur le générateur, il ne fonctionne que dans un sens. Pourtant aucune indication de polarité n'est présente sur le moteur.

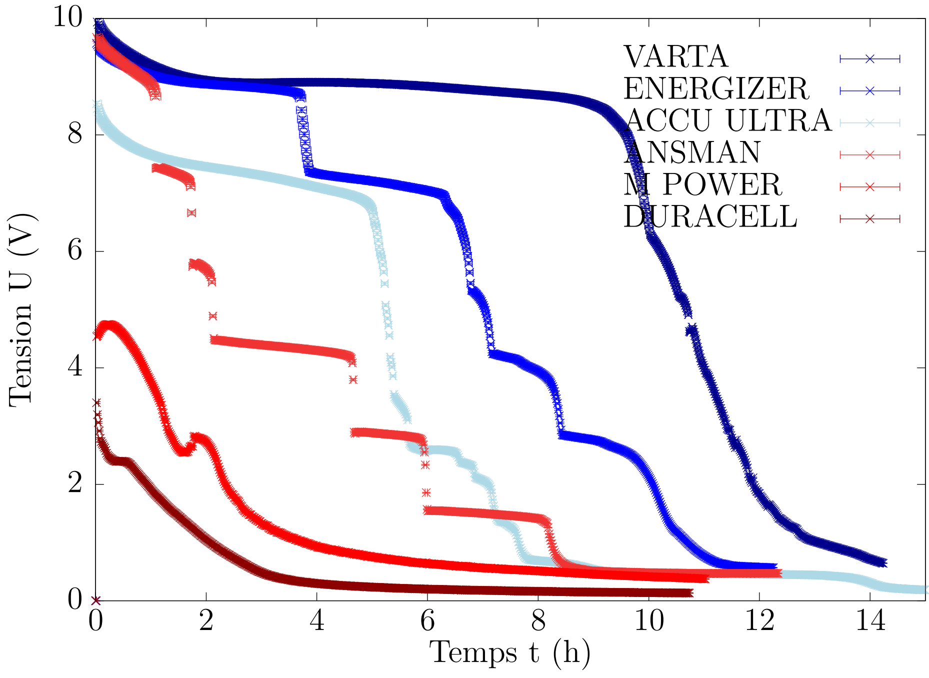
Le constat de mesures faiblement linéaires en mode bloqué, m'a fait tester sur le moteur d'Opitec une capacité en parallèle avec la même polarisation que le moteur et sur le moteur inconnu, une diode de retour de courant classiquement utilisée pour protéger les moteurs. La figure 27 présente la situation. Sur la figure 27a la capacité est clairement visible, puisqu'elle est reliée directement aux bornes du moteur. Par contre, sur la figure 27b, c'est en arrière plan, à côté de la LED qu'on voit la diode. On voit aussi ce deux moteurs non bloqués. En mode bloqués, une pince crocodile se trouvait attachée à leur axe. Pour le moteur inconnu, on voit une structure en bois sur laquelle on peut distinguer la présence d'un petit aimant qui va servir à déterminer sa vitesse de rotation.

Les multiples essais opérés en augmentant la tension en mode bloqué s'étant avéré de mauvaise qualité, l'idée de partir de la tension maximale et de la diminuer est venue. La procédure est la suivante : On fait tourner le moteur pour que le courant tourne autour de 250 mA. Puis, brutalement on le bloque et rapidement on descend la tension en effectuant des mesures très régulièrement jus qu'à zéro. Si, à priori le résultat devait être le même, ce ne fut pas le cas. Les mesures présentées sur le graphe de la figure 28 le montrent.

Il faut mentionner que la mesure directe avec un ohmmètre de la résistance interne du moteur Opitec a donné 1 Ω et que celle du moteur inconnu a donné environ 50 Ω.

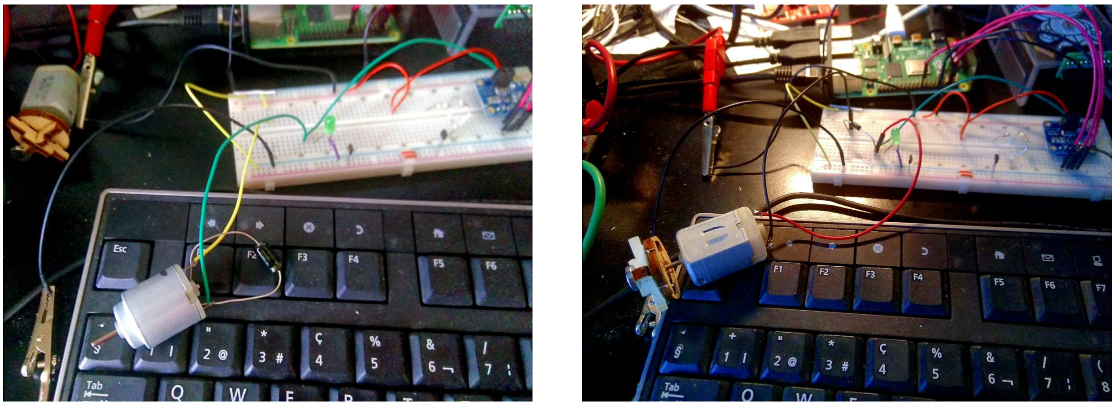
(a) Mesure de chute de tension





(b) Chute de tension vs temps

FIGURE 26 – Plusieurs accu. 9 V et une résistance 455 Ω 0,25 W



(a) Le moteur d'Opitec (b) Le moteur inconnu

FIGURE 27 – Résistance interne de deux moteurs

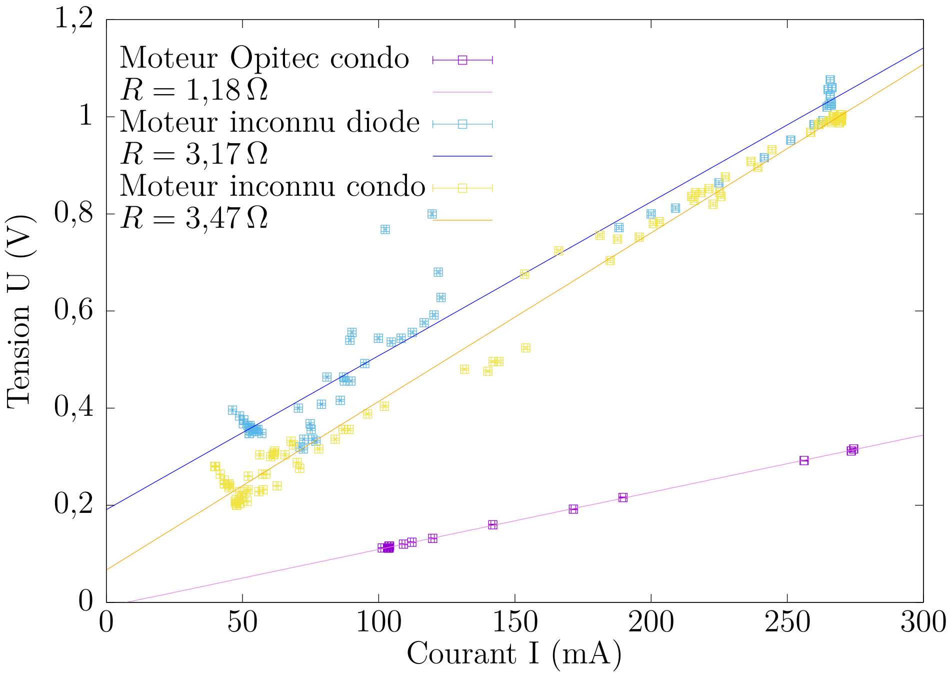


FIGURE 28 – Résistance interne de deux moteurs

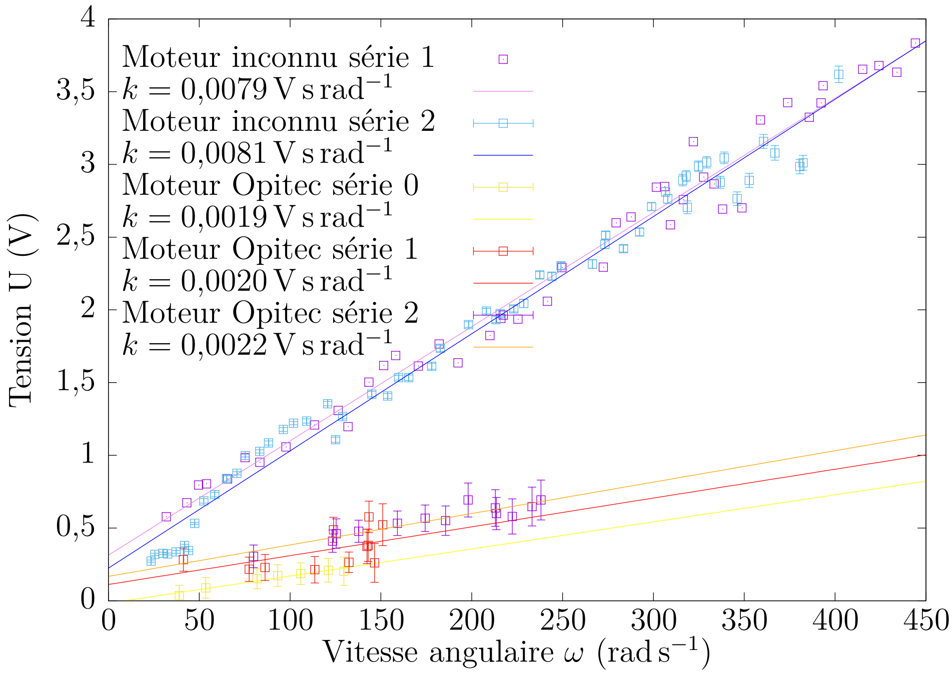


FIGURE 29 – Constante k du moteur inconnu

Visiblement la qualité des mesures pour le moteur inconnu dépend surtout de la manière de diminuer la tension. L'utilisation d'un condensateur à la place d'une diode ne change rien.

Relevons encore que le blocage des deux moteurs n'a pas posé de problème d'échauffement. Aucun des deux moteurs n'a été endommagé.

4.5.2. Constante de proportionnalité

Pour la détermination de la constante de proportionnalité, il a fallu mélanger les trois système que sont le moteur, l'INA et le compteur de tours magnétique dans la même expérience.

Le résultat est présenté à la figure 29, page 37.

Les deux séries de mesures pour le moteur inconnu ont été faites en commençant avec un moteur en rotation maximale. L'abaissement de la tension a permis d'obtenir des vitesses de rotations plus faibles que si on avait démarré avec une tension nulle et élevé celle-ci progressivement. En effet, les frottements entravaient tellement le moteur à faible vitesse de rotation qu'il ne démarrait que par à coup, alors qu'en diminuant sa vitesse, il s'arrêtait avec une vitesse plus faible qu'il ne démarrait.

Sur le graphe de la figure 29, on voit deux séries de mesures prises dans les mêmes conditions pour le moteur inconnu. Ce sont celles dont les pentes sont les plus importantes. La différence est que sur la première les incertitudes ne sont pas présentes. Si elles le sont sur la seconde, elles sont clairement trop faibles. Cela est certainement dû à une sous estimation de l'incertitude de la résistance interne prise à $0{,}1\,\Omega$, pour une

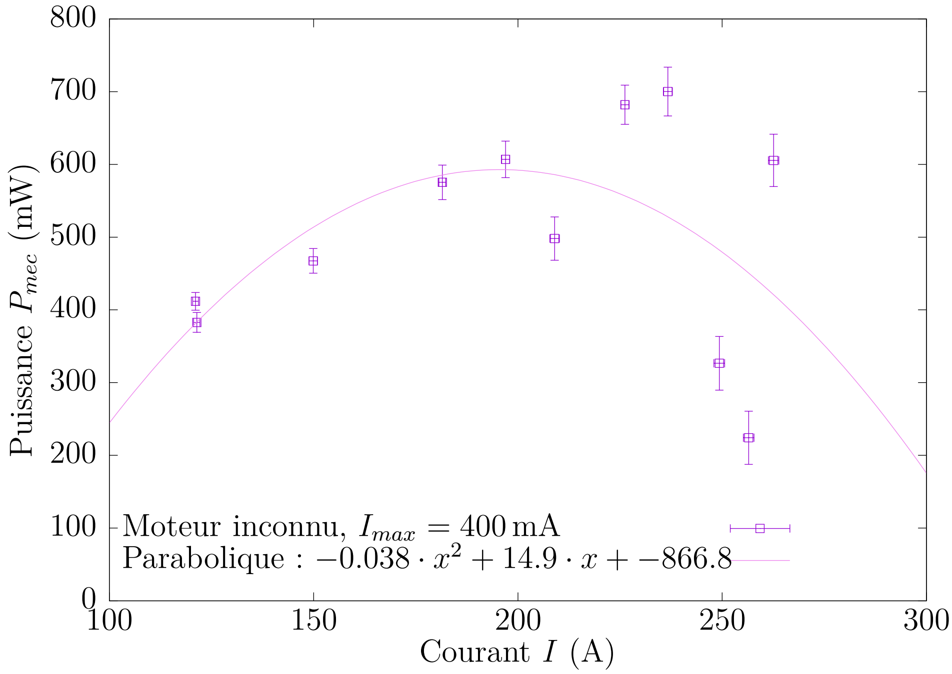


FIGURE 30 – Puissance maximale du moteur

valeur de celle-ci à 3,17 Ω, soit 3,2% d'incertitude alors que la figure 28 présente deux mesures de la résistance interne de ce moteur à 3,17 et 3,47 Ω, soit un écart de 0,3 Ω supplémentaire qui représente presque 10% supplémentaires.

On voit aussi trois autres séries de mesures pour le moteur Opitec. Ce sont les trois séries de plus faible pente. Pour ces mesures, la résistance interne utilisée est de 1,18 Ω. Mais, cette fois, son incertitude à été portée à 0,5 Ω, ce qui représente une incertitude relative de 42% tout de même. De plus, pour des raisons pratiques, le moteur cylindrique étant mal fixée sur la planche, il a fallu augmenter la tension plutôt que de la diminuer dans les deux premiers cas. Seul le dernier correspond à des mesures faites en diminuant la tension. En relevant que le meilleur moyen de faire varier progressivement la tension était simplement d'écouter le moteur dont le bruit changeait clairement avec elle, il faut souligner que le moteur Opitec avec de nombreuses variation de vitesse de rotation pendant la mesure. Enfin, il suffisait d'appuyer sur celui-ci ou de le faire basculer en avant ou arrière pour que la vitesse de rotation change. Les deux dernières série de mesures pour ce moteur ayant été faites en tentant de stabiliser le moteur par appui sur celui-ci pendant la rotation, au son, on a pu constater une légère augmentation de la vitesse qui se traduit sur la figure 29 par deux droites légèrement déplacées vers le haut.

4.5.3. Puissance maximale

À l'aide du dispositif expérimental de la figure

5. Discussion

5.1. Résistances

Dans un premier temps les mesures faites n'étaient pas très bonnes, puisqu'un écart d'environ 45% a été obtenu sur une résistance de 100 kΩ. Or, avec une précision sur le courant de l'ordre de 0,1 mA et une tension maximum du générateur de 11 V, comme le courant maximum était :

$$I_{max.} = U_{max.}/R = 11/100'000 = 0{,}11\,\mathrm{mA}$$

la précision était insuffisante.

En mode 16 V maximum, soit pour une résolution de 0,1 mA, la mesure de résistance de valeur aussi importante n'est clairement pas possible. Je n'ai pas testé la mesure en mode 32 V, avec une résolution annoncée de 0,8 mA, car mon générateur ne dépassait pas 11 V. De plus, les générateurs du lycée ne dépassant pas 20 V, cela m'a semblé inutile.

Avec des valeurs de résistance plus basse, le principal problème devenait de ne pas dépasser la valeur maximale de puissance de ¼W, avec les résistances que j'avais à disposition. Une modification du programme s'est donc avérée nécessaire pour m'avertir d'un éventuel dépassement.

Ce dépassement à pourtant été réalisé avec une résistance de 50 Ω soumise à une tension de 11 V, soit une puissance de 2,42 W ! Elle a bien noirci, mais la mesure par la pente du graphe $U = R \cdot I$ n'a donné qu'un écart de 12% et c'est cette résistance qui a été utilisée par la suite et à donné les mesures du graphe 15, dans un domaine de puissance correct cependant. Cela pour dire que ces résistances sont solides.

Dans l'ensemble les mesures sont de bonne qualité, pour autant qu'on s'en tienne à un domaine de puissance correct ne dépassant pas la valeur maximale admise par chaque résistance.

Pour cela, une modification du programme a été réalisée demandant cette puissance (en mW) à chaque nouvelle série de mesure.

Les résultats sont excellents. Les écarts sont inférieurs ou égaux à 1% et cela en raison du grand nombre de mesures réalisées. En effet, avec une dizaine de mesures, certains résultats étaient supérieurs à 1%. On peut même se demander si l'incertitude de l'ohmmètre n'est pas surestimée.

5.2. Générateur

La mesure de la décroissance de la tension a été difficile à réaliser, car la valeur des potentiomètre utilisés était soit trop importante (46 kΩ) mais avec une puissance maximale de 1 W, soit avec une puissance inférieure à 100 mW qui fait que deux d'entre eux ont grillés.

De plus, quand la résistance devient faible, la linéarité n'est plus bonne et il faut faire très attention, car le courant monte soudainement très vite.

La comparaison des deux graphes des figures 24a et 25, montre pour les huit accumulateurs NiMH de 1,2 V la mesure d'une résistance interne de 6,5 Ω et 9,9 Ω.

Le graphe de la figure 26a, qui présente la décroissance de la tension avec le courant pour des générateurs chargés au maximum est intéressant. On voit que, chargés, tous présentent une tension à vide de 9 V si on suit la régression linéaire. Mais pour une même charge initiale (environ 1,2 kΩ), la tension initiale n'est absolument pas la même.

De plus, leur résistance interne est totalement différente. Pour trois d'entre eux (Accu Ultra, Mpower et Duracell) elle est importante, voire très importante. On peut supposer que c'est là le signe d'accumulateurs en fin de vie. Les trois autres (Varta, Energizer et Ansman) présentent une tensions élevée quasiment constante et donc une très faible résistance interne.

Sur la figure 26b, où les accumulateurs sont identiques à ceux de la figure 26a, les couleurs étant respectées, on voit que la décharge au cours du temps n'est pas toujours continue et varie beaucoup selon les générateurs. Cependant, si on compare les deux graphes, on voit qu'à part le couple Accu Ultra et Ansman qui sont inversés, plus la résistance interne est grande, plus la décroissance de la tension se fait rapidement.

5.3. Moteur

5.3.1. Résistance interne

5.3.2. Constante de proportionnalité

On constate une valeur de la constante k quatre fois inférieure pour le moteur Opitec, par rapport au moteur inconnu : env. $0{,}002\,\mathrm{V\,s\,rad^{-1}}$ pour le premier, contre $0{,}008\,\mathrm{V\,s\,rad^{-1}}$ pour le second.

6. Conclusion

L'INA219 est vraiment un multimètre très performant. Il est particulièrement adapté à des mesures réalisées avec des élèves d'option spécifiques de physique soutenus par des élèves d'option complémentaire informatique.

Références

[1] S. Monk, *Mouvement lumière et son avec ARDUINO et RASPBERRY PI*, Eyroles, 2016.

[2] T. Karvinen, K. Karvinen et V. Baltokari, *Les capteurs pour Arduino et Raspberry Pi*, Dunod, 2104.

7. Annexes

A. Drivers INA219

Cette annexe présente l'installation des drivers pour l'INA219 sur raspberrypi 4. Deux possibilités sont offertes pour utiliser l'INA219 sur cette plateforme :

pi-ina219 une librairie python « for Voltage and Current Sensors Using the INA219 » disponible à l'adresse `https://github.com/chrisb2/pi_ina219` et compatible python2 et python3.

CircuitPhyton un ensemble de librairies permettant d'utiliser l'INA219 mais aussi l'INA260, non utilisé ici, mais uniquement avec python3. La référence est : `https://cdn-learn.adafruit.com/downloads/pdf/adafruit-ina219-current-sensor-breakout.pdf?timestamp=1583432707`.

La seconde solution est développée plus bas, mais sans exemples d'utilisation, puisque l'ensemble de ce travail a été réalisé avec l'INA219.

A.1. INA219 : installation

Conformément à la réfécence donnée ci-dessus au point 1, l'installation se fait avec PIP pour python 3 par :

Listing 5 – Installation de la librairie de l'INA.

```
sudo apt-get install python3-pip
sudo pip3 install pi-ina219
```

Il ne faut pas oublier d'activer le bus I2C dans les options avancées de l'outil de configuration du raspberry : *raspi-config* et, toujours selon cette référence, l'utilisation est alors simplement celle donnée par le listing 6.

On y voit que la déclaration de la librairie est très simple et que l'objet ina créé et configuré, quatre instructions permettent de récupérer le courant, la tension, la puissance et la tension de shunt. Chacune est une méthode de cet objet (*ina.voltage()*, *ina.current()*, *ina.power()* et *ina.shunt_voltage()*). Il faut relever que la tension de shunt est la tension aux bornes de la résistance éponyme qui est de $0{,}1\,\Omega$, que la tension de bus, ou tension de charge, est la tension entre la terre et la broche Vin- de l'INA219 (voir figure 3, page 7) et donc que la tension entre la borne Vin+ de l'INA219 est la somme des tensions de bus et de shunt. Ces tension sont mesurées en V, alors que le courant est lui mesuré en mA.

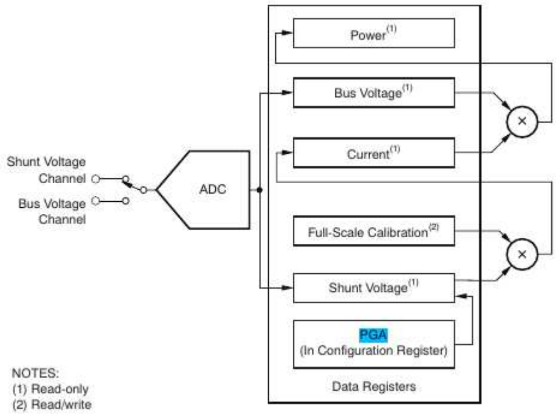
L'INA219 est constitué d'un élément permettant de mesurer physiquement la tension aux bornes de la résistance de shunt et celle du bus et d'un convertisseur analogique-logique (ADC : Analogic Digital Converter). La documentation de référence se trouve à l'adresse : `https://www.ti.com/lit/ds/symlink/ina219.pdf` et dit ceci à la page 10 :

> *"The INA219 can be used without any programming if it is only necessary to read a shunt voltage drop and bus voltage with the default 12-bit resolution,*

Listing 6 – Utilisation de la librairie de l'INA.

```python
#!/usr/bin/env python
from ina219 import INA219
from ina219 import DeviceRangeError

SHUNT_OHMS = 0.1

# MAX_EXPECTED_AMPS = 2.0
# MAX_EXPECTED_AMPS = 0.3
# ina = INA219(SHUNT_OHMS, MAX_EXPECTED_AMPS)
# ina.configure(ina.RANGE_16V, ina.GAIN_1_40MV)
# ina.configure(ina.RANGE_16V)

def read():
    ina = INA219(SHUNT_OHMS)
    ina.configure()

    print("Bus Voltage: %.3f V" % ina.voltage())
    try:
        print("Bus Current: %.3f mA" % ina.current())
        print("Power: %.3f mW" % ina.power())
        print("Shunt voltage: %.3f mV" % ina.shunt_voltage())
    except DeviceRangeError as e:
        # Current out of device range with specified shunt
           resistor
        print(e)

if __name__ == "__main__":
    read()
```

Figure 31 – Diagramme fonctionnel de l'INA



320-mV shunt full-scale range (PGA = /8), 32-V bus full-scale range, and continuous conversion of shunt and bus voltage.

Without programming, current is measured by reading the shunt voltage. The Current register and Power register are only available if the Calibration register contains a programmed value."

Conformément au diagramme fonctionnel présenté au paragraphe 8.2 de la page 9 de la documentation de référence et reproduit ici à la figure 31, ce sont les tensions de bus et de shunt qui sont mesurées et les courant et puissance sont dérivés.

Plusieurs modes de fonctionnement sont possibles. Ils sont donnés dans le listing 7 qui est tiré du code source utilisé dans le module ina219.py : `https://github.com/chrisb2/pi_ina219/blob/master/ina219.py`. La description des fonctionnalités et des exemples d'utilisation du module sont à l'adresse : `https://github.com/chrisb2/pi_ina219`.

On constate la possibilité de configurer deux domaines de tension : 16 V et 32 V (mode par défaut) et différents gains au niveau du shunt qui permettent une tension maximale de shunt de 320 mV. Avec une résistance de shunt de 0,1 Ω, cela représente un courant de 3,2 A. Avec ce mode par défaut, soit avec une amplification de gain programmée (PGA) de 8 fois ($8 \cdot 40 = 320$ mV) et une résolution par défaut de 12 bits, soit $2^{12} = 4096$ niveaux, le courant par bit, soit la résolution de la mesure de courant vaut $3,2/4096 = 0{,}8$ mA. Or, la précision de la mesure, tant en tension qu'en courant est annoncée dans la documentation (p. 5), typiquement à 0,2% et au maximum à 0,5%, ce qui représente au maximum 16 mA. On comprends ainsi que le choix du nombre de bits de conversion analogique-digital est surtout important pour les temps de conversion, comme par exemple pour la mesure de la décroissance de condensateurs à faible capacités.

La lecture du document de référence cité ci-dessus est intéressante, mais complexe. Ce rapport étant essentiellement tourné vers l'utilisation de l'INA219 dans le cadre de composants électroniques, il suffira de dire qu'à la création de l'objet ina, la valeur de la résistance de shunt (0,1 Ω) et le courant attendu maximal peuvent lui être passés. Ces indications serviront à déterminer automatiquement le gain. On peut aussi spécifier

Listing 7 – Modes de fonctionnement de l'INA219.

```
RANGE_16V = 0  # Range 0-16 volts
RANGE_32V = 1  # Range 0-32 volts

GAIN_1_40MV = 0  # Maximum shunt voltage 40mV
GAIN_2_80MV = 1  # Maximum shunt voltage 80mV
GAIN_4_160MV = 2  # Maximum shunt voltage 160mV
GAIN_8_320MV = 3  # Maximum shunt voltage 320mV
GAIN_AUTO = -1  # Determine gain automatically

ADC_9BIT = 0  # 9-bit conversion time  84us
ADC_10BIT = 1  # 10-bit conversion time 148us
ADC_11BIT = 2  # 11-bit conversion time 2766us
ADC_12BIT = 3  # 12-bit conversion time 532us
ADC_2SAMP = 9  # 2 samples at 12-bit, conversion time 1.06ms
ADC_4SAMP = 10  # 4 samples at 12-bit, conversion time 2.13ms
ADC_8SAMP = 11  # 8 samples at 12-bit, conversion time 4.26ms
ADC_16SAMP = 12  # 16 samples at 12-bit,conversion time 8.51ms
ADC_32SAMP = 13  # 32 samples at 12-bit, conversion time 17.02ms
ADC_64SAMP = 14  # 64 samples at 12-bit, conversion time 34.05ms
ADC_128SAMP = 15  # 128 samples at 12-bit, conversion time 68.10
   ms
```

la tension maximale admise pour les même raisons, mais au niveau de la procédure de configuration. Pour les mesures effectuées (voir listing 1, page 9), 16 V au maximum ont été prévus et un courant maximum de 300 mA.

La documentation de l'INA219 précise que la précision des mesures, tant sur la tension que sur le courant, est de 0.5% au maximum. Cette indication est importante, mais les conditions de configuration sous lesquelles elles pourrait être plus faible, sont peu claires. Nous en resterons donc là dans ce travail.

A.2. INA260 : installation

Blabla ...

B. Introduction aux LEDs

B.1. Les LEDs

Les « Light Emitting Diodes » ou LED sont d'abord des diodes. Une diode est un élément électronique polarisé, c'est-à-dire qu'il a un comportement dans un sens qu'il n'a pas dans l'autre.

B.1.1. Fonctionnement inverse

Dans ce sens, une diode ne laisse pas passer le courant. On peut élever la tension, aucun courant ne circulera. Pour les diodes classiques, on peut fortement élever la tension, dite inverse, avant d'endommager le composant. Pour une LED, une tension inverse maximale se situe de 3 à 5 V.

B.1.2. Fonctionnement direct

Dans ce sens, une diode présente un seuil à partir duquel le courant passe. Pour une diode classique, une valeur typique de seuil est de 0,7 V. Pour une LED, une valeur typique de seuil est de 2 V. Il ne s'agit pas d'une transition très franche, mais on passe sur quelques dixièmes de volts d'un courant nul, à des valeurs de quelques dizaines de milli-ampères.

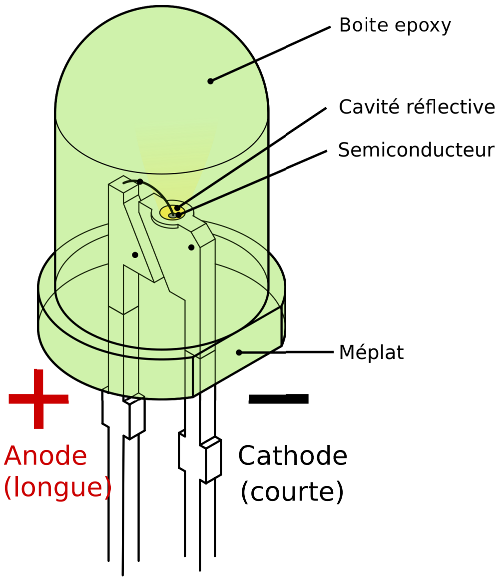
B.1.3. Rayonnement

Une diode classique ne rayonne pas. Une LED émet un rayonnement qui peut être de couleur différente. Il existe des LEDs rouges, jaunes, vertes, bleues, ...et même infra-rouges ou ultraviolettes.

B.1.4. Physique des semi-conducteurs

Une diode est un semi-conducteur électroluminescent. De quoi s'agit-il ? À vous de vous renseigner.

Figure 32 – Le composant LED



B.1.5. Le composant

Le composant est simple. Le schéma de la figure 32 le présente [6]. On y voit que celui-ci est composé d'une anode et d'une cathode. Physiquement, elles se distinguent par le fait que l'anode a une longueur plus important que la cathode. Or, l'anode est la partie de la LED qui doit se trouver au potentiel le plus haut, soit sur la broche de 3,3 V du raspberrypi. On constate aussi que la cathode, la broche négative se trouve indiquée par un méplat sur le composant.

B.2. Le GPIO

On parle de ports GPIO pour désigner les entrées/sorties numériques à usage général (General Purpose digital Input/Output) du raspberry (ou d'un autre microordinateur). Il s'agit de broches lui permettant de communiquer avec l'extérieur. La figure 33 présente l'ensemble des broches avec quelques indications de fonctionnement que nous allons reprendre ci-dessous [7].

Il existe deux notations possibles pour les broches du GPIO. L'une est la numérotation dite physique. L'autre est la numérotation Broadcom (BCM ou GPIO) qui permet d'utiliser la librairie RPI.GPIO et dont les numéros correspondent aux registres de la puce ARM BCM.

6. Voir Wikimedia Commons : `https://commons.wikimedia.org/wiki/File:LED,_5mm,_green_(en).svg`

7. Il faut savoir que les fonctions supplémentaires au fonctionnement GPIO des broches sont en réalité bien plus nombreuses que le montre la figure 33. Pour en avoir une idée plus précise, voyez la référence suivante `https://microcontrollerslab.com/raspberry-pi-4-pinout-description-features-peripherals-applications/`

Listing 8 – Le code de clignotement de la LED.

```
import RPi.GPIO as GPIO
# Définition du mode de nommage des broches
GPIO.setmode(GPIO.BOARD)
```

Listing 9 – Le code de clignotement de la LED.

```
import RPi.GPIO as GPIO
# Définition du mode de nommage des broches
GPIO.setmode(GPIO.BCM)
```

Du point de vue logiciel, chacun de ces deux mode de numérotation est accessible par configuration du GPIO. Ainsi, pour la numérotation physique on la déclare grâce aux lignes du listing 8. Pour la numérotation BCM, on la déclare grâce aux lignes du listing 9.

Évidemment, comme ce sont des entrées sorties, les broches GPIO peuvent être utilisées soit en entrée, soit en sortie [8]. Quelque soit le mode, ce sont des entrées/sorties numériques. Elles ne délivrent ou ne reçoivent que des niveaux bas (0 V) ou haut (3,3 V). Le raspberry ne dispose pas d'entrée analogique, mais peut simuler des sorties analogiques grâce notamment au mode PWM (Pulse Width Modulation).

B.3. GPIO en direct

Avec au plus 20 mA par broche du GPIO, une connection directe de la LED est envisageable. Mais le courant de fonctionnement normal était généralement de 10 mA, mettre la LED en série avec une résistance est plus prudent.

Les broches de contrôle en sortie du raspberry délivrant une tension de 3,3 V et la tension de fonctionnement des LEDs utilisées se situant aux alentours de 2 V, la tension couverte par la résistance constitue la différence, soit 1,3 V. Pour qu'un courant de 10 mA la traverse, il faut une résistance de :

$$R = \frac{U}{I} = \frac{1,3}{10 \cdot 10^{-3}} = 130\,\Omega$$

Le branchement se fait donc entre la broche 17 du GPIO (11ebroche) et l'une des broches GND (6ebroche), qui est la terre, avec une résistance normalisée à 180 Ω en série, comme le montre la figure 35a et le schéma de la figure 34.

Pour comparaison, la figure 35b montre la luminosité obtenue avec une résistance de 50 Ω. Avec une telle résistance, le courant traversant la résistance change. Pour comparaison, on a :

$$I_{180\,\Omega} = \frac{1,3}{180} = 7{,}2\,\text{mA} \ ; I_{50\,\Omega} = \frac{1,3}{50} = 26\,\text{mA}$$

8. Voir le meilleurs site de référence : `https://microcontrollerslab.com/raspberry-pi-4-pinout-description-features-peripherals-applications/`.

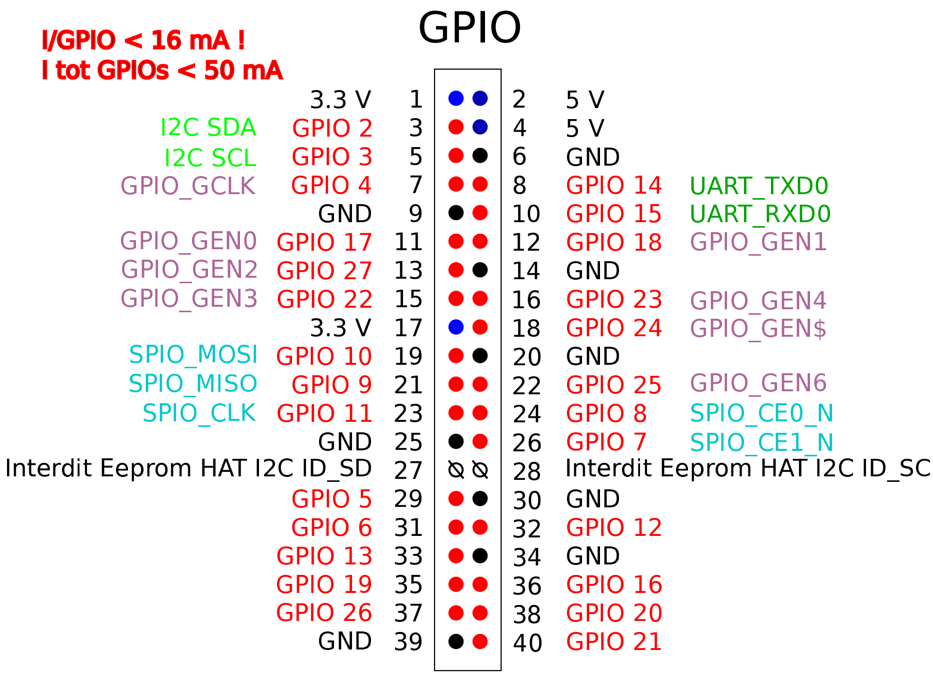


FIGURE 33 – Le port GPIO

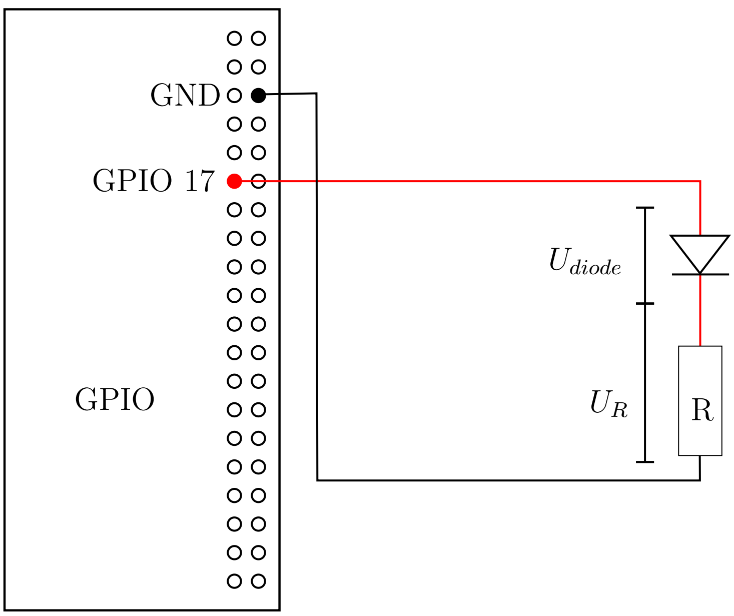
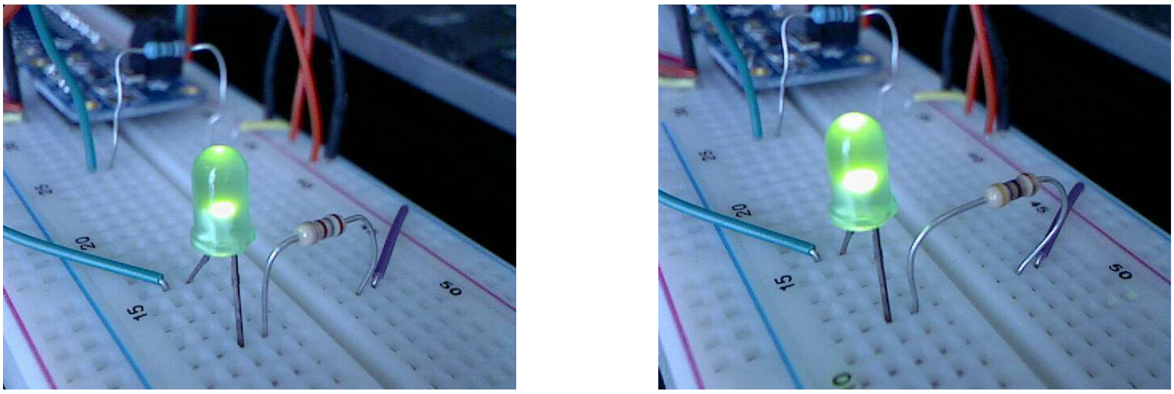


FIGURE 34 – Diode commandée à travers une résistance



(a) Résistance de 180 Ω　　(b) Résistance de 50 Ω

FIGURE 35 – LED et résistances

Le code nécessaire pour faire clignoter cette LED est simple. Il est donné au listing 10.

Le code 10 se comprend comme suit :

1. ligne 1 : on déclare la bibliothèque nécessaire pour la gestion du GPIO et on lui donne le nom GPIO,
2. ligne 2 : on déclare la bibliothèque nécessaire pour faire attendre le programme via time.sleep,
3. ligne 4 : on déclare comment on va nommer les pins, ici selon BroadCoM, c'est-à-dire selon les numéros GPIO et pas le numéro des broches,
4. ligne 6 : on met la valeur de la broche choisie pour alimenter la LED dans la variable control_pin,
5. ligne 8 : on définit le mode sortie pour cette broche,
6. ligne 10 : on lance la boule principale du programme,
7. ligne 13 : on éteint la LED,
8. ligne 14 : on patiente,
9. ligne 16 : on allume la LED et
10. ligne 22 : si un problème est constaté, on met toutes les broches en mode entrée, ce qui évite les courts-circuits. En effet, si on met accidentellement le câble d'une broche à la terre et que celle-ci est en mode sortie haute, un court-circuit se produit[9].

Pour terminer le programme, il suffit de presser Ctrl C.

9. Voir « Why cleanup the GPIO pins » : `https://raspberrypi.stackexchange.com/questions/34363/why-cleanup-the-gpio-pins#34370` et le site de référence, section « Digital Input/Output Pins » : `https://microcontrollerslab.com/raspberry-pi-4-pinout-description-features-peripherals-applications/`.

Listing 10 – Le code de clignotement de la LED.

```
import RPi.GPIO as GPIO
import time
# Définition du mode de nommage des broches
GPIO.setmode(GPIO.BCM)
# Définition de la broche utilisée
control_pin = 17
# Déclaration du mode de sortie pour cette broche
GPIO.setup(control_pin, GPIO.OUT)
try:
    while True:
        # Boucle ininterrompue
        # Extinction
        GPIO.output(control_pin, False)
        time.sleep(5)
        # Allumage
        GPIO.output(control_pin, True)
        time.sleep(2)
finally:
    # Si problème, remise à entrée de toutes
    # les broches évitant des cours circuit
    print('Nettoyage')
    GPIO.cleanup()
```

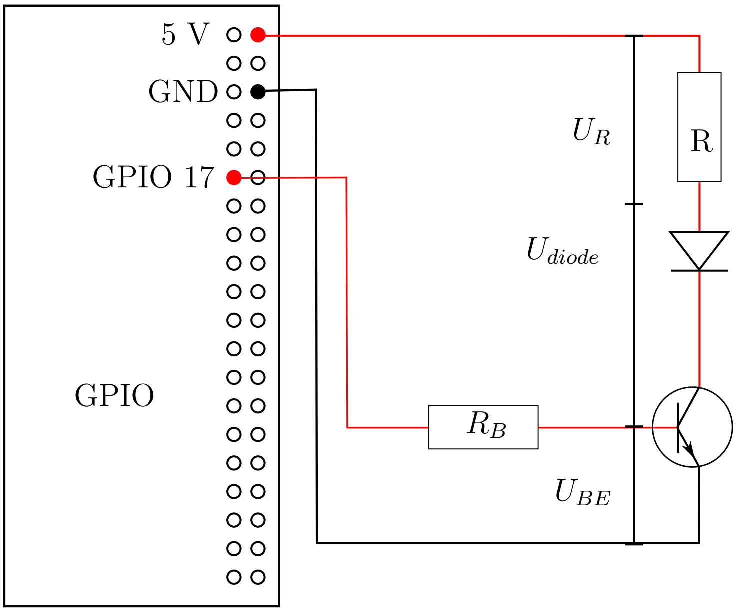


FIGURE 36 – LED, résistances et transistor

B.4. Via un transistor

Nous allons ici utiliser un transistor pour commander l'alimentation d'une led. Le schéma de l'expérience est celui présenté à la figure 36.

Le calcul des valeur des composant du circuit de la figure 36 débute avec le choix du courant qui va traverser la LED. Une broche du GPIO ne peut dépasser un courant maximum de 20 mA. Nous allons choisir 15 mA. Il faut savoir que la tension entre la base et l'émetteur d'un transistor comme le TO-92 ou 2N2222A qui a été choisi vaut environ 0,5 V. Ainsi, entre la broche à 5 V et la terre, les tensions sont de 0,5 V au niveau du transistor, de 2 V au bornes de la LED et, en conséquence de 2,5 V au niveau de la résistance R. Comme le courant choisi est de 15 mA, la résistance R nécessaire vaut :

$$R = \frac{U}{I} = \frac{2,5}{15 \cdot 10^{-3}} = 167\,\Omega$$

Par ailleurs, la documentation du TO-92 spécifie que le facteur de gain (hFE) est de 75 pour un courant de collecteur de 10 mA. Ainsi, pour avoir un tel courant du collecteur à l'émetteur, il faut que la base délivre un courant de :

$$I_{base} = \frac{10}{75} = 0{,}13\,\text{mA}$$

Ainsi, en considérant que la broche de commande GPIO 17 délivre une tension de 3,3 V, à laquelle il faut soustraire les 0,5 V du transistor, on a une tension de 2,8 V aux bornes de la résistance R_{base}. Sa valeur pour un courant de 0,13 mA vaut donc :

$$R_{base} = \frac{2,8}{0,13 \cdot 10^{-3}} = 22\,\text{k}\Omega$$

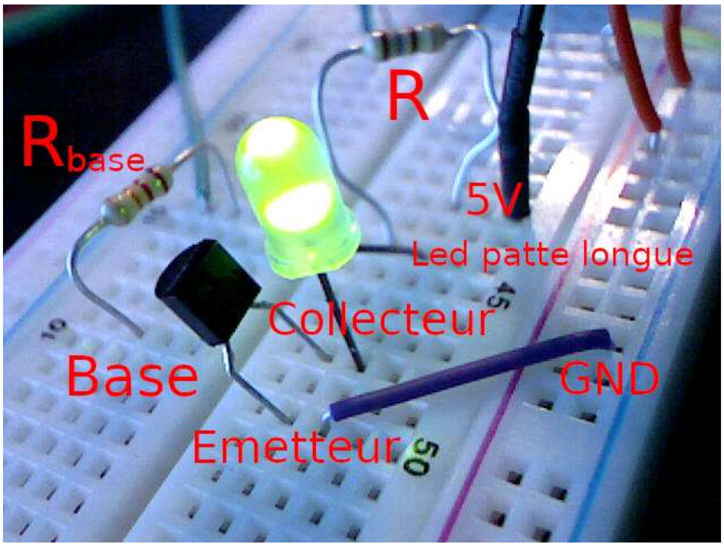


FIGURE 37 – LED avec un transistor

Il n'est pas possible en réalité de disposer de résistances de valeurs rigoureusement exactes. Il faut utiliser les valeurs normalisées les plus proches. Ainsi, on a utilisé $R = 180\,\Omega$ et $R_{base} = 22\,\mathrm{k}\Omega$. La figure 37 présente le résultat, parfaitement fonctionnel avec exactement le même code que celui du listing 10.

B.5. Le T-Cobbler Plus

Pour éviter les contacts avec les pins du GPIO, Adafruit propose un dispositif physique qui permet de relier, via une nappe, le GPIO à la breadboard (voir la figure 38). De plus, l'extrémité du côté de la breadbord présente un T ou est reporté l'ensemble des numéros des pins du GPIO (voir la figure 39). Cela permet de se connecter aux pins facilement avec des câbles mâle-mâle et de situer facilement les pins désirés. Il s'agit du T-Cobbler Plus. Le Plus traduit un T-Cobbler permettant une connection aux GPIO avec 40 pins. Il existe aussi le T-Cobbler pour des GPIO à 26 broches. Évidemment ce type de dispositif sécurise l'accès au GPIO, mais prend de la place sur la breadboard.

Sur les figures 38 et 39, on voit aussi à droite l'INA219, ainsi que comment le brancher sur le T-Cobbler Plus.

B.6. Arduino

La même réalisation peut être faite avec un arduino. Ici nous allons utiliser une « Red-Board » de Sparkfun qui est une carte équivalente à la carte Arduino Uno. L'éditeur de code prévu pour Arduino est parfaitement utilisable pour cette carte et les pins sont les mêmes. On voit cette carte faisant fonctionner une led sur la figure 40.

Pour faire fonctionner cette carte, il faut d'abord la brancher sur un ordinateur. Puis y installer « Arduino IDE », le lancer et dans le menu outil, sous type de carte, choisir

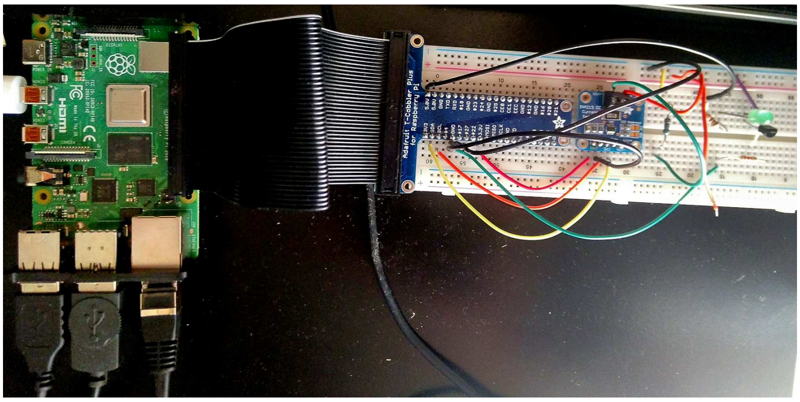


Figure 38 – Le T-Cobbler et le RPI

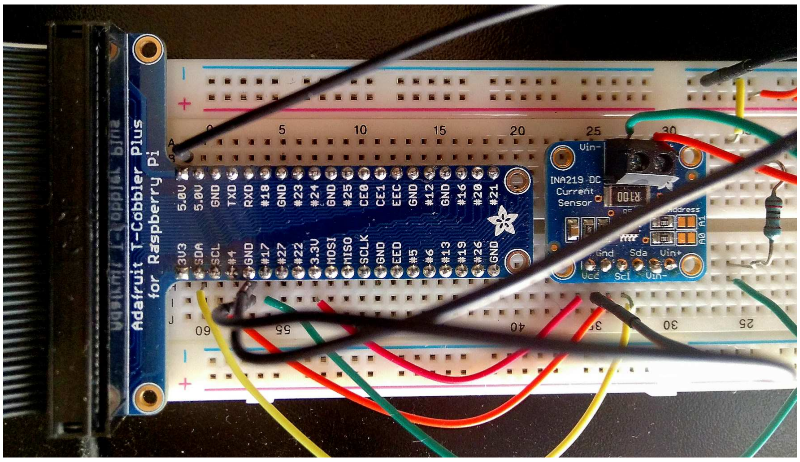


Figure 39 – Le T-Cobbler et son affichage

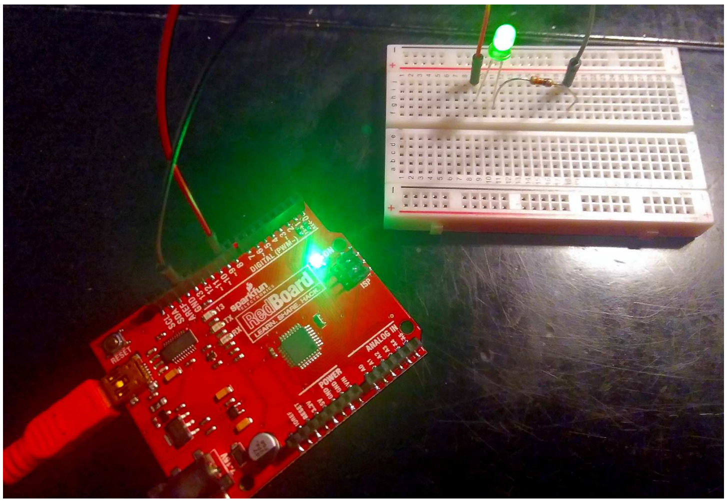


FIGURE 40 – La RedBoard avec une LED verte.

Arduino uno, sous port série, choisir */dev/ttyUSB0*. La carte est alors connectée à l'ordinateur et on va pouvoir y verser le code en C permettant de gérer la LED. Celui-ci est donné dans le listing 11.

À l'aide d'Arduino IDE, on compile ce code (bouton vu vert) et on l'envoie sur la RedBoard avec le bouton flèche droite.

Le montage physique est extrêmement simple, puisque la longue patte de la LED est directement branchée sur le pin 9, sa patte la plus courte sur la résistance est l'autre extrémité de la résistance sur le pin GND. Avec une résistance de 330 Ω, la LED éclaire raisonnablement, comme le montre la figure 40. Attention, la LED verte qui se trouve sur la RedBoard est le témoin de fonctionnement de la carte. La LED pilotée par la broche 9 se trouve en haut à droite de la figure.

C. Introduction aux mesures avec l'INA 219

C.1. Le dispositif expérimental

En supposant les drivers de l'INA déjà installé et l'INA 219 parfaitement fonctionnel, voici la procédure pour bien démarrer la série de mesures permettant d'obtenir le graphe tension-courant pour des résistances ohmiques.

Le dispositif expérimental est présenté la figure 41. On y voit à gauche le GPIO du Raspberrypi4, au centre l'INA219, la résistance R et le générateur externe à droite.

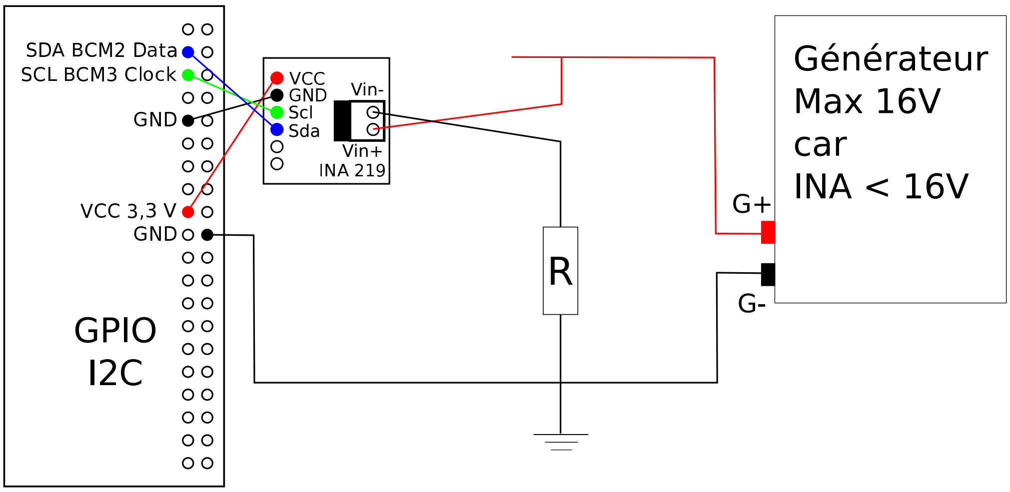
La résistance de shunt permettant à l'INA de mesurer le courant est représentée en noir contre les ports de mesure. Celle-ci doit se placer en mode « high side », c'est-à-dire entre la tension positive du générateur continu (Vin+) et le pôle de tension la plus élevée de la résistance R (Vin-).

Ainsi, en reliant la masse du générateur, du pôle de tension la plus basse de la résistance R et le Ground (GND) du GPIO, étant donné la faible valeur de la résistance de schunt

Listing 11 – Le code de clignotement de la LED.

```
/*
  Clignottement d'une LED verte
  Fait clignoter une LED toute les demi-secondes.
*/

// On utilise le pin 9 (digital)
int led = 9;

// La fonction d'initialisation lancée par le reset
void setup() {
  // initialise le pin led digital comme une sortie
  pinMode(led, OUTPUT);
}

// La boucle principale
void loop() {
  digitalWrite(led, HIGH);  // enclanche la LED (HIGH est le
     niveau de tension)
  delay(500);               // attend une demi-seconde
  digitalWrite(led, LOW);   // éteint la LED en mettant sa
     tension à LOW
  delay(500);               // attend une demi-seconde
}
```

Figure 41 – Le RPI4, l'INA219 en mode high side et la résistance R



(Rs = 0,1 Ω), la précision de la mesure de la tension aux bornes de R va peu fluctuer et sera de meilleure qualité qu'avec une mesure « low side ».

Du côté du bus I2C, à part l'alimentation (VCC 3,3V) et la terre (GND) de l'INA, les données sont transmises par la broche SDA et l'horloge par la broche SCL.

La figure 42 presente le dispositif expérimental concrètement. On y remarque la présence du T-Cobbler plus, permettant une lecture des fonctions des ports du GPIO et un branchement facilité sur le RPI4.

C.2. Les mesures de résistance

La procédure expérimentale consiste en séries de mesures successives sur différentes résistances. Les données sont récupérées sur le bus I2C (voir annexe A, page 42) à l'aide du programme Python du listing 12.

Il faut relever que le programme enregistre la mesure de la puissance dissipée dans la résistance et suivant la valeur du maximum possible, ici les résistances utilisées étaient des ¼W, annonce le dépassement. Le réglage de la tension se fait en tournant le potentiomètre du générateur et chaque mesure est effectuée en pressant la touche ENTER.

La figure 43 présente en arrière plan le code python permettant les mesures de tension en fonction du courant pour une résistance, l'affichage des mesures faites par ce code et le graphe gnuplot correspondant.

Par ailleurs, Gnuplot permettant la réalisation d'un graphe comportant plusieurs courbes pour chaque résistance à partir du même fichier, pour autant que chaque série de données soit séparées par deux lignes vides, le programme utilisé (listing 12) permet de demander une nouvelle série de mesures en pressant la touche n (pour nouvelle) et cela autant de fois qu'on le désire jusqu'à la fin déclarée par la touche q (pour quitter).

Détaillons ce code.

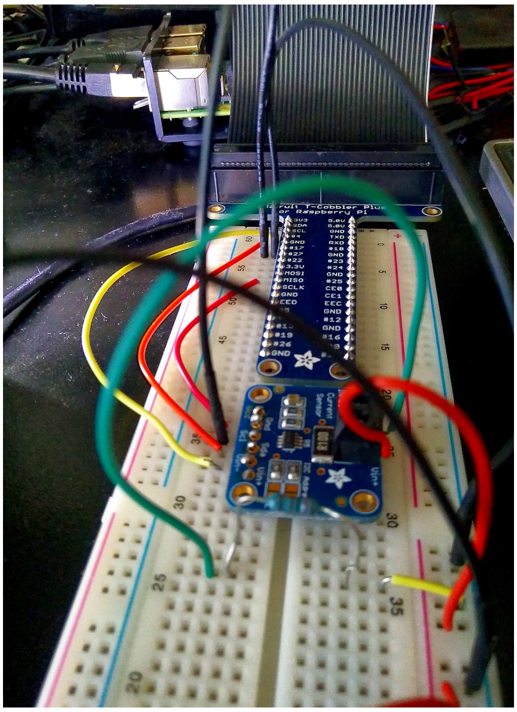


FIGURE 42 – INA, résistance et T-Cobbler plus

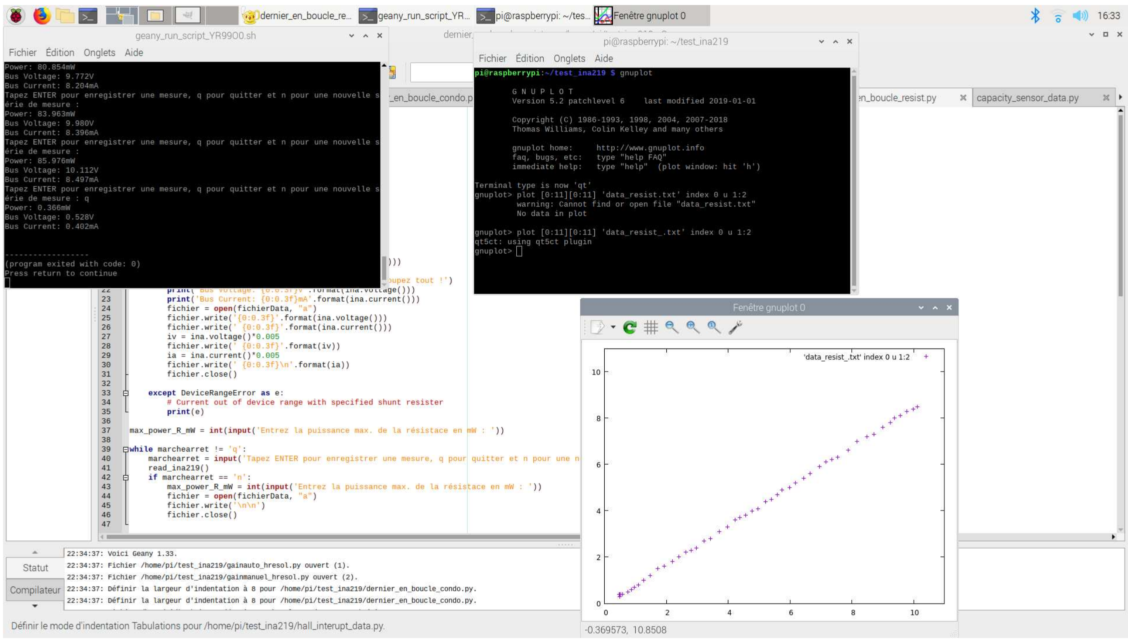


FIGURE 43 – Code, fonctionnement et graphe des mesures de résistances

Les trois modules importés sont celui de l'`INA219`, `DeviceRangeError` et `sleep`. Nous verrons pourquoi par la suite. En majuscule figurent ensuite la déclaration des constantes qui sont la valeur du shunt de l'INA, obligatoirement celle-ci, et la valeur du courant maximal qu'on s'autorise. À la ligne 6, on instancie, c'est-à-dire on crée, un objet qu'on nomme `ina`, grace au constructeur d'objet INA219(). Puis, on configure cet objet avec la méthode qui lui est liée `.configure()`, selon les variables spécifiées plus haut. À la ligne 9, on spécifie la limite de puissance en mW qu'on s'impose pour ne pas dépasser le quart de watt des résistances classiques utilisées. On définit enfin une variable `marchearret` qui va nous permettre d'interrompre ou de poursuivre le programme.

Vient ensuite la fonction chargée des mesures, nommée `read_ina219()`. Elle est constituée d'un try-except basé sur le module `DeviceRangeError` permettant d'interrompre le programme à la moindre erreur.

Les quatre instructions `print` qui suivent permettent d'afficher dans la console les mesure de tension aux bornes de la résistance, de courant à travers elle, de puissance dans la résistance, et si celle-ci dépasse la valeur qu'on s'est autorisée (ligne 15, de le signaler clairement, et enfin de tension aux bornes de la résistance de shunt. Cette dernière étant monitorée pour information.

Puis, à la ligne 21, on ouvre un fichier *data.txt* en mode ajout (a) de ligne (et non en mode remplacement) pour y écrire les mesures. On y met la tension aux bornes de la résistance, le courant qui la traverse, l'incertitude sur la tension et l'incertitude sur le courant (les deux incertitudes étant calculée dans les variables iv et ia). Enfin, on ferme le fichier.

Remarquons que l'écriture des mesures `ina.voltage()`, `ina.current()`, ...est formatée par la méthode `.format()` qui s'applique à la chaine de caractère qui la précède. Cette méthode remplace dans la chaine les accolades par un nombre suivi d'un nombre de chiffres après la virgule spécifié avant la lettre f.

Enfin vient le programme principal constitué d'une boucle dont l'arrêt est spécifié par la présence de la lettre q dans la variable `marchearret` récupérée à chaque itération de la boucle par un `input()`. À la ligne 36, on lance la fonction de mesure.

Sans le test suivant, à chaque pression de la touche *Enter*, une nouvelle mesure est affichée et enregistrée dans le fichier. Le test de la ligne 37, permet de tester si, après avoir fait une série de mesures sur une résistance, on veut en faire une autre. Auquel cas, on redemande la puissance maximale et on réouvre le fichier pour y écrire deux lignes vides (`\n\n`). En effet, grace à ces lignes, Gnuplot saura qu'il a à faire avec une nouvelle série de mesures.

C.3. Graphes

Pour réaliser les graphes des mesures, on commence par utiliser Gnuplot en mode direct. Il faut pour cela ouvrir une console et entrer *gnuplot* à l'invite de commande. Ensuite, à l'invite de commande de Gnuplot (`gnuplot>`), il faut demander un graphe :

```
plot [0:11][0:11] 'data.txt' index 0 using 1:2
```

Listing 12 – Le code de mesure avec l'INA.

```
from ina219 import INA219, DeviceRangeError
from time import sleep
SHUNT_OHMS = 0.1
#MAX_EXPECTED_AMPS = 2.0
MAX_EXPECTED_AMPS = 0.3
ina = INA219(SHUNT_OHMS, MAX_EXPECTED_AMPS)
#ina.configure(ina.RANGE_16V, ina.GAIN_1_40MV)
ina.configure(ina.RANGE_16V)
max_power_R_mW = 230
marchearret = ''
def read_ina219():
    try:
        print('Bus Voltage: {0:0.3f}V'.format(ina.voltage()))
        print('Bus Current: {0:0.3f}mA'.format(ina.current()))
        if ina.power() < max_power_R_mW:
            print('Power: {0:0.3f}mW'.format(ina.power()))
        else:
            print('!!!! Dépassement de puissance !!!!
        Coupez tout !')
        print('Shunt Voltage: {0:0.3f}mV\n'.format(ina.
           shunt_voltage()))
  fichier = open("data.txt", "a")
        fichier.write('{0:0.3f}'.format(ina.voltage()))
        fichier.write(' {0:0.3f}'.format(ina.current()))
        iv = ina.voltage()*0.005
        fichier.write(' {0:0.3f}'.format(iv))
        ia = ina.current()*0.005
        fichier.write(' {0:0.3f}\n'.format(ia))
        fichier.close()
    except DeviceRangeError as e:
        print(e)
max_power_R_mW = int(input('Entrez la puissance max.
  de la résistace en mW : '))
while marchearret != 'q':
    marchearret = input('Tapez ENTER pour enregistrer une
      mesure, q pour quitter et n pour une nouvelle série de
         mesure : ')
  read_ina219()
  if marchearret == 'n':
        max_power_R_mW = int(input('Entrez la puissance max.
          de la résistace en mW : '))
        fichier = open("data.txt", "a")
        fichier.write('\n\n')
        fichier.close()
```

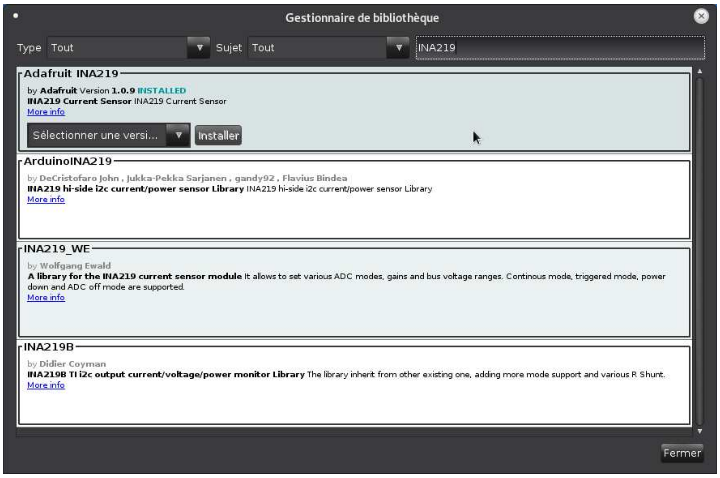


FIGURE 44 – Le gestionnaire de bibliothèques

pour autant que cette commande soit lancée dans le même répertoire que le fichier de données.

Ainsi, un graphe sommaire sera réalisé qui permettra de se faire un idée des résultats et notamment de la pente du graphe pour la comparer avec la résistance.

Ensuite, il faut bien conserver, archiver, nommer les fichiers de mesures pour pouvoir ensuite les utiliser directement dans LaTeX. Pour cela, on se référera à ce qui a été vu dans les précédents rapports.

C.4. Arduino

Pour comparaison, nous allons ici utiliser l'Arduino pour réaliser les mêmes mesures. La différence dans l'utilsation de l'Arduino par rapport au paragraphe B.6 est que l'IDE utilisé sera mis-à-jour. En effet, l'INA nécessite plusieurs librairies qui ne sont pas installées par défaut et l'installation de celles-ci avec la dernière version de l'IDE d'Arduino est grandement facilitée. Néanmoins, cela pose le problème de l'installation hors du cadre des paquets se trouvant dans les dépôts de Debian, car cette nouvelle version n'y figure pas. On peut donc penser soit à rétroporter le paquet de la version *Testing* de Debian, soit à l'installer avec le script d'installation d'Arduino. La page dédiée à cette installation sous linux est la suivante : `https://www.arduino.cc/en/Guide/Linux` et celle permettant le téléchargement est : `https://www.arduino.cc/en/software`. En suivant cette procédure et éventuellement en réglant quelques problèmes de droits mentionné à la fin de la première référence, on se retrouve avec un IDE fonctionnel. Il faut alors comme précédemment y sélectionner la carte et le port utilisés avec par exemple `/dev/ACM0`.

Il faut ensuite installer la librairie de l'INA219. Pour cela, la nouveauté de cette version de l'IDE est qu'elle permet un installation en ligne. En effet, en se rendant dans le menu `Croquis` et en y sélectionnant l'entrée `Inclure une bibliothèque`, on peut ouvrir un gestionnaire de bibliothèques en choisissant `Gérer les bibliothèques`. La figure 44 montre le gestionnaire qui est lancé. On y voit qu'il existe quatre bibliothèques pour

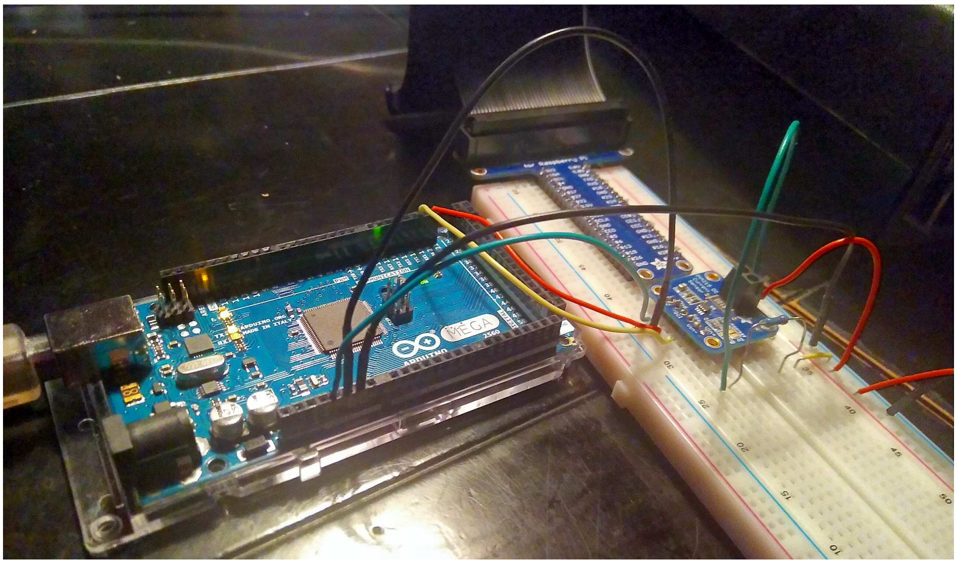


Figure 45 – Le montage physique de l'INA sur l'Arduino.

l'INA219, dont la première est celle d'Adafruit. C'est celle-ci que nous avons choisie et dont on a demandé l'installation avec le bouton apparaissant sur la figure 44. Il est alors demandé d'installer quatre autres librairies, ce qu'il faut impérativement faire. L'installateur met alors en place tout ce qui est nécessaire au fonctionnement de l'INA et notamment un script permettant de vérifier le fonctionnement de l'INA, disponible dans les `Exemples` du menu `Fichier` sous `Adafruit INA219`, `getcurrent`.

Pour le montage de l'INA sur l'Arduino, la figure 45 présente la situation. On y voit que le bus I2C est câblé avec la broche Vcc 5 V de l'Arduino au premier plan à gauche, fil bleu-clair, une terre juste à côté de la tension d'alimentation, le SCL en arrière plan avec un fil orange et le SDA à côté en jaune. On y voit aussi le montage de l'INA avec la résistance et le générateur qu'on devine tout à droite alimentant les deux fils en bas à droite. Le schéma du montage est identique à celui de la figure 41 de la page 57.

Le code permettant la mesure de résistances est celui présenté à la figure ...

C.5. Travail pratique LED

C.5.1. Objectif

Pour apprendre à utiliser le raspberrypi et son port GPIO, vous allez l'utiliser pour faire clignotter une LED. Plus précisément, il faudra lui faire émettre un SOS en morse.

C.5.2. Préalable

Pour faire fonctionner un raspberrypi, il faut le microordinateur lui-même, une alimentation 5 V, un clavier, une souris et un câble HDMI qu'il faut brancher sur l'écran. Vous utiliserez un raspberrypi 3. C'est une machine lente, mais parfaitement adaptée pour son utilisation en physique.

Le système d'exploitation est Raspbian OS, un dérivé de la célèbre distribution Debian, avec une interface graphique très légère. Ce système est installé sur la carte SD qui se trouve sous le raspberrypi.

Pour relier les composants électroniques au raspberrypi, il faut utiliser une interface nommée GPIO et une plaque permettant de les enficher facilement : la breakboard.

Pour programmer le GPIO, nous allons utiliser le langage Python et l'interface de programmation Geany.

Attention, au préalable de tout branchement sur le GPIO, il faut impérativement savoir ce qu'on fait, sous peine d'une destruction du raspberrypi par court-circuit. Ne faites donc rien sans avoir au préalable la CERTITUDE qu'aucun problème ne survienne. Pour cela, il est nécessaire de très bien se documenter.

C.5.3. À réaliser

Commencez donc par vous documenter sur les éléments suivants :

- l'ensemble des broches du GPIO et, notamment, les deux systèmes notation pour la numérotation de celles-ci, mais aussi le courant maximum qu'elles peuvent fournir,
- ce qu'est une LED, comment elle fonctionne, comment elle se branche dans un circuit, quelle est la tension et le courant maximal qu'elle admet et pourquoi il est nécessaire de l'utiliser avec une résistance,
- comment se font les connexions sur la breakboard, c'est-à-dire comment sont reliés les connecteurs qui la composent,
- quel est la tension et le courant maximal ou la puissance maximale qu'une résistance peut supporter,
- quel sont les modules python nécessaire pour gérer le GPIO et faire clignoter la LED.

C.5.4. Fonctionnement continu

Il s'agit de dessiner et de réaliser un circuit permettant d'allumer en continu la LED grâce au GPIO. Dessinez et montez le circuit avec le raspberrypi éteint. Avant toute chose, faite vérifier votre circuit par l'enseignant.

C.5.5. Fonctionnement clignottant

Une fois le fonctionnement continu réalisé, écrivez avec Geany le code nécessaire pour faire clignoter indéfiniment la LED avec un intervalle de 3 secondes allumée et 2 secondes éteinte. Mais ne lancez pas le code sans vérification de l'enseignant.

C.5.6. Le SOS

Réalisez enfin par vous-même le SOS demandé.

C.5.7. Documentation

Dans l'optique d'un rapport de physique sur un composant électronique, commencez un rapport sous LaTeXen rédigeant une annexe comprenant touts les éléments importants de votre expérience sur l'émission d'un SOS. L'idée est de documenter au mieux vos connaissances sur les LEDs et sur le GPIO.

C.6. INA219, mesures de résistances

Le raspberrypi ne dispose pas d'entrée analogiques. La mesure directe de tension et/ou courant n'est donc pas possible. Pour la réaliser, il est donc nécessaire d'utiliser un convertisseur analogique-numérique. L'INA219 est un voltmètre-ampèremètre muni d'un tel convertisseur. C'est donc lui que nous allons utiliser.

C.6.1. Documentation

Documentez-vous sur ce circuit intégré. Trouvez notamment sur quel bus il fonctionne, comment en réaliser le branchement sur le raspberrypi et comment réaliser le circuit high-side lui permettant de faire la mesure du courant et de la tension au bornes d'une simple résistance alimentée par un générateur.

Trouvez ensuite sa plage de fonctionnement en tension et courant, sa précision pour ces deux grandeurs et trouvez la puissance maximale que peut supporter une résistance classique.

À quoi doit-on faire attention au sujet de la terre des instruments ?

C.6.2. Schéma du dispositif

Réalisez un schéma de branchement et faites-le vérifier par votre enseignant.

C.6.3. Programmation

Déterminez la/les bibliothèques nécessaire à une utilisation de l'INA219 avec Python3.

Préparez un code permettant de faire une première mesure de tension-courant pour la résistance. Faites valider ce code par votre enseignant.

C.6.4. Réalisation

Réalisez le montage, enclanchez les appareils et effectuez une mesure.