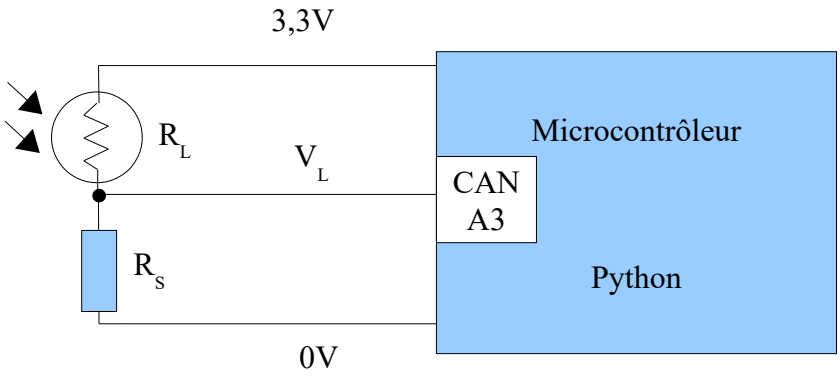
Activité : allumage automatique d'un lampadaire

Auteur : Christophe LIGERET

I / Test du capteur de lumière

Le capteur de lumière est une photorésistance connectée avec une résistance sur une entrée analogique A3 du microcontrôleur : cet ensemble {photorésistance ; résistance} envoie une tension électrique comprise entre 0V et 3,3V sur A3. Ci-dessous, un shémat simplifié de l'installation

On considère le montage ci-dessous



La fonction informatique `lireCapteurLumiere()` présente dans la bibliothèque `CapteurLumiere` permet de lire les données provenant du capteur de lumière.

Une fois appelée, cette fonction renvoie un nombre entier (de type `int`) entre 0 et 1023 qui correspond à l'intensité lumineuse reçue : par exemple :

- si le capteur photoélectrique est dans l'obscurité totale, la fonction renvoie 0
- si le capteur est exposé à une lumière très intense, la fonction renvoie 1023
- etc... (cette fonction renvoie toutes les valeurs entières possibles entre 0 et 1023).

Pour information, on donne ci-dessous son code source présent dans la bibliothèque `CapteurLumiere`.

```
import board
from analogio import AnalogIn
analog_in = AnalogIn(board.A3)

def lireCapteurLumiere(): # Définition de lireCapteurLumiere
    return analog_in.value # On renvoie le résultat
```

2°) a) Combien de niveaux de luminosité différents le capteur de lumière peut-il renvoyer ?

b) Combien de bits au minimum sont-ils nécessaires pour coder tous ces niveaux de lumière ?

c) La fonction `lireCapteurLumiere()` renvoie un résultat représenté sur un `int`, est-ce compatible avec le résultat trouvé dans la question précédente ? Justifier.

f) On considère le programme ci-dessous :

```
from CapteurLumiere import lireCapteurLumiere
import time


while True:
    valeur = lireCapteurLumiere()
    print(valeur)
    time.sleep(1.0)
```

Commenter chacune des lignes de ce programme puis dire ce que fait ce programme.

Que fait l'instruction `print(valeur)` ?

g) En vous inspirant du programme ci-dessus, écrire un nouveau programme qui affiche la tension électrique (en volts, avec une résolution de 0,01 V) présente sur l'entrée analogique A3.

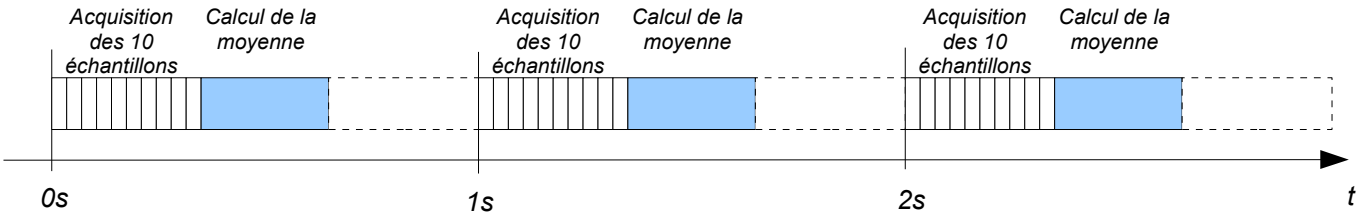
II / Moyenne de 10 échantillons

On rappelle que la fonction informatique `lireCapteurLumiere()` permet de lire les données provenant du capteur de lumière sur 1024 niveaux différents.

Il peut toutefois arriver parfois que certaines valeurs soient abérantes : perturbation électromagnétique, problèmes de contacts électriques, surtensions dues aux moteurs... Ces perturbations risqueraient de perturber le fonctionnement du système d'allumage automatique du lampadaire.

Aussi, pour réduire ces effets, nous allons calculer la moyenne de 10 échantillons et travailler ensuite uniquement sur les moyennes de 10 échantillons.

Ci-dessous, on illustre le fonctionnement du calcul de la moyenne des 10 échantillons.
On calcule les moyennes toutes les secondes, ce qui veut dire qu'on effectue 10 mesures d'intensité lumineuse chaque seconde.



Au moyen de la fonction `lireCapteurLumiere()` on fait une acquisition de 10 échantillons qu'on stocke dans un tableau, puis on calcule la moyenne de ces 10 échantillons.

3°) a) On propose le programme ci-dessous :

```
import board
import time
from analogio import AnalogIn
analog_in = AnalogIn(board.A3)

def lireCapteurLumiere(): # Définition de lireCapteurLumiere
    return analog_in.value # On renvoie le résultat


def lireMoyenneCapteurLumiere(): #Définition de  lireMoyenneCapteurLumiere
  resultat = 0
  for i in range (0,10):
     resultat = resultat + lireCapteurLumiere()
     time.sleep(0.1)
  return resultat/10.0
```

b) Quel est le type de résultat renvoyé par la fonction `lireMoyenneCapteurLumiere` ?

c) Pourquoi a-t-on placé l'instruction `time.sleep(0.1)` et en combien de temps environ s'exécute la fonction `lireMoyenneCapteurLumiere()` ?

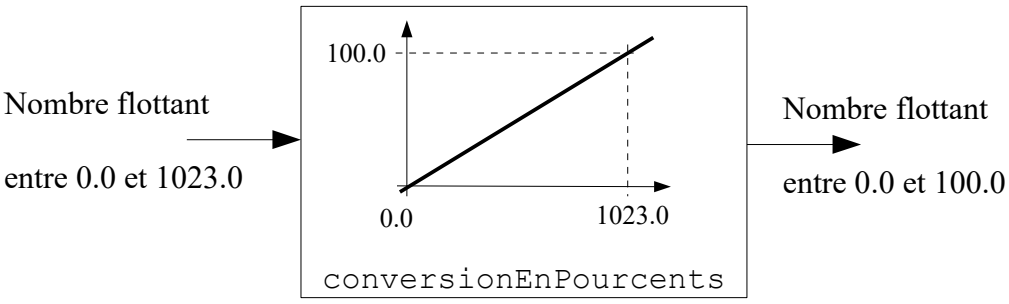
d) Proposer un programme en langage Python qui affiche sur la console de l'ordinateur toutes les 2 secondes l'intensité lumineuse moyenne.

III / Normalisation des valeurs du capteur de lumière entre 0% et 100%

Les nombre générés par la fonction `lireMoyenneCapteurLumiere()` sont des nombres décimaux (flottants) dont les valeurs entre 0.0 et 1023.0 ne sont pas très parlantes.

Aussi, on souhaite construire un nouvel indicateur qui soit un pourcentage d'éclairage maximum.

On souhaite donc réaliser une fonction informatique, nommée `conversionEnPourcents` et basée sur une fonction affine, qui convertisse les résultats (décimaux entre 0.0 et 1023.0) en un nombre décimal entre 0.0 et 100.0 comme décrit sur le schéma ci-dessous :



a) Proposer une fonction informatique nommée `conversionEnPourcents` basée sur une fonction affine, qui, à la valeur générée par `lireMoyenneCapteurLumiere` renvoie une valeur entre 0.0 et 100.0

b) Compléter ce programme pour afficher à l'écran de l'ordinateur, chaque secondes, l'intensité lumineuse mesurée sous forme d'un pourcentage (nombre décimal entre 0.0 et 100.0).
On utilisera la communication série USB pour vérifier le fonctionnement correct du programme.

c) Commentaires

IV / Commande à seuil du lampadaire

a) En exploitant les programmes précédents et les résultats affichés à l'écran de l'ordinateur, déterminer :

- la valeur affichée en pourcents lorsque le capteur est exposé directement à l'éclairage ambiant.
- la valeur affichée en pourcents lorsqu'on place la main 5 cm au dessus du capteur

b) Déterminer alors un niveau de luminosité (en pourcents) qui permette à la fois :

- d'allumer le lampadaire lorsqu'on place la main 5 cm au dessus du capteur.
- d'éteindre le lampadaire lorsqu'on le capteur est exposé directement à l'éclairage ambiant.

Ce niveau de luminosité pourra être défini dans le programme comme suit :

```
NIVEAULUMINOSITE 60.0 // Exemple où le niveau pour allumer ou éteindre est 60%
```

c) A partir des programmes précédents, compléter le programmeci-dessous de manière à allumer le lampadaire lorsqu'on place la main 5 cm au dessus du capteur.

```
from  CapteurLumiere import conversionEnPourcents
import board
from digitalio import DigitalInOut, Direction
NIVEAULUMINOSITE 60.0

lampadaire = DigitalInOut(board.D13) # Le lampadaire est branché sur PIN13
lampadaire.direction = Direction.OUTPUT

while True:
  niveauLuminositePourcents = conversionEnPourcents()
  .....( niveauLuminositePourcents > .....):
    lampadaire.value = .....
    print("Le lampadaire est........")
  .....:
    lampadaire.value = .....
    print("Le lampadaire est ........")
```

d) Lorsqu'on place ou retire la main du capteur, on observe un petit retard. A quoi est-il dû ?
Proposer puis implémenter une solution simple pour diminuer ce retard.

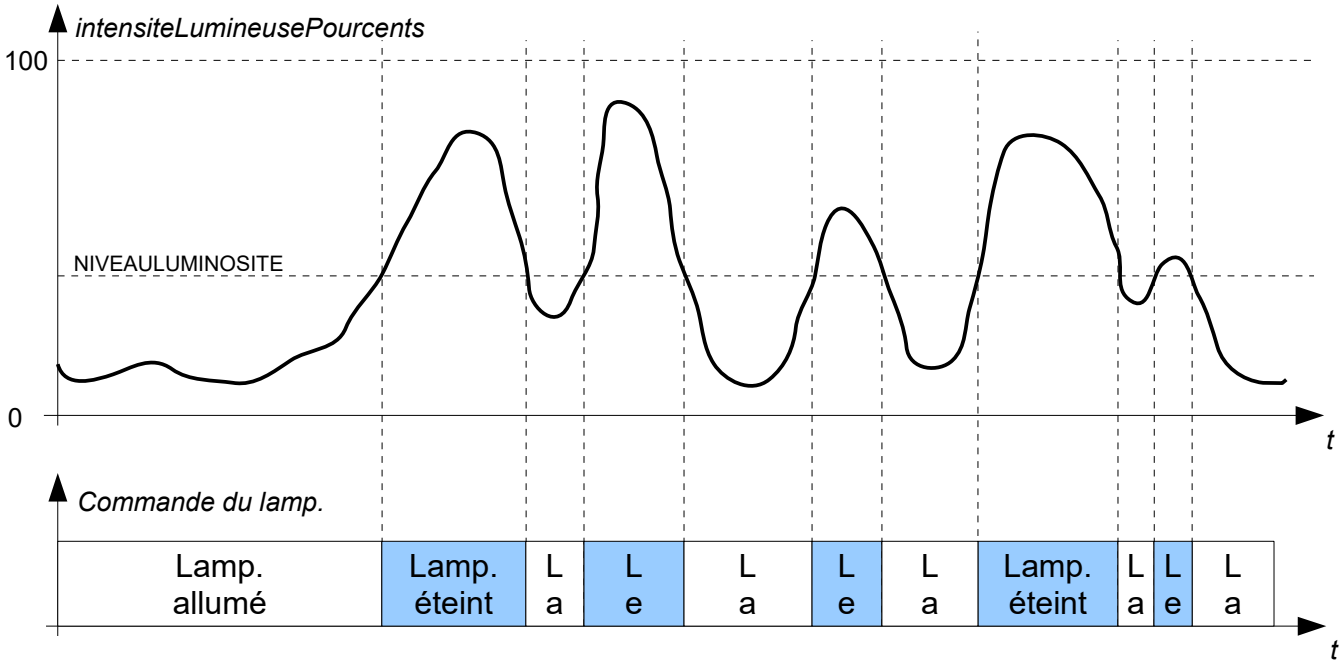
e) Placer la main au dessus du capteur de manière à ce que la valeur de l'intensité lumineuse en pourcent soit proche de NIVEAULUMINOSITE.
Bouger très légèrement la main. Qu'observez vous ? Est-ce acceptable ?

V / Commande à hystérésis du lampadaire

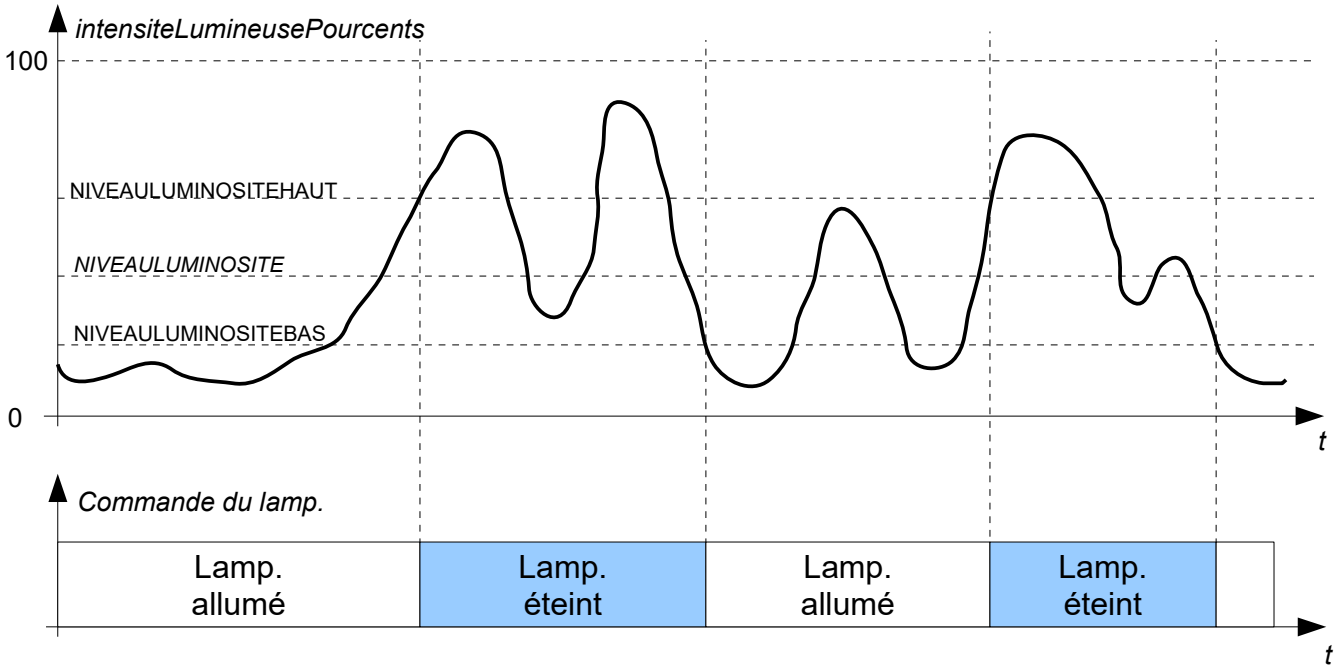
On a observé ci-dessous, que dans certaines conditions, lorsque la luminosité est très proche du niveau d'activation / désactivation du lampadaire, ce dernier pouvait clignoter.

Pour résoudre ce problème, on ne va plus utiliser un seul niveau d'activation / désactivation, mais deux niveaux : NIVEAULUMINOSITEBAS et NIVEAULUMINOSITEHAUT (avec NIVEAULUMINOSITEBAS ≤ NIVEAULUMINOSITEHAUT)

Sur le chronogramme ci-dessous, on rappelle le fonctionnement du lampadaire avec l'unique seuil NIVEAULUMINOSITE. On observe l'instabilité du pilotage du lampadaire qui "clignote".



Sur le chronogramme ci-dessous, on donne les spécifications du fonctionnement du lampadaire avec la commande à hystérésis. On observe que la commande du lampadaire est plus stable.
On donne deux seuils de déclenchements du lampadaire : NIVEAULUMINOSITEBAS et NIVEAULUMINOSITEHAUT
remarque : ici le seuil NIVEAULUMINOSITE est donné à titre comparatif (plus utilisé).



6°) Proposez un code en langage Python correspondant à la commande à hystérésis.

VI / Commande à hystérésis avec temporisation du lampadaire

On observe avec le programme n°6 qu'il persiste tout de même une petite instabilité et que dans certaines conditions, le lampadaire a encore tendance à clignoter (mais moins qu'avec le programme n°6).

Pour améliorer la commande par hystérésis, on va lui rajouter des temporisation :

- on impose que le lampadaire reste allumé au minimum 5 secondes
- on impose que le lampadaire reste éteint au minimum 2 secondes

Redit autrement :

- lorsque le lampadaire vient d'être allumé, on interdit de l'éteindre avant 5 secondes.
- lorsque le lampadaire vient d'être éteint, on interdit de l'allumer avant 2 secondes.

a) Proposez un code en langage Python correspondant à la commande à hystérésis avec temporisation.

Indices : utiliser une variable entière qui servira de compteur
On comptera en centièmes de secondes : rajouter delay(10); à la fin de la fonction loop().

b) Commentaires