INF1900

Projet initial de système embarqué
Travail pratique No. 6
Capteurs et conversion analogique/numérique

Objectif: Introduire les interfaces et les capteurs
Date de remise: vendredi 22 février 23h00 au plus tard
Travail préparatoire: Aucun
Documents à remettre: Le code écrit pour résoudre les deux problèmes sera placé sous Git et sera corrigé par les chargés de laboratoire.
Présentation en classe: fichier LibreOffice Impress

> “It would appear that we have reached the limits of what it is possible to achieve with computer technology, although one should be careful with such statements, as they tend to sound pretty silly in 5 years”
>
> *-John von Neumann, 1903-1957*

Introduction

Le but de ce travail pratique est de prendre connaissance du monde des capteurs et de voir la façon de les interfacer au microcontrôleur. Dans un premier temps, l'attention sera portée sur les capteurs avec une interface numérique. Par la suite, nous verrons les capteurs analogiques et la manière particulière de les interfacer au système. Il convient tout de même de limiter notre étude puisque le domaine des capteurs est extrêmement vaste. On trouvera de très bonnes sources d'information sur ce sujet sur Internet (Wikipédia, Sensors On Line, TE, Keyence, etc.) pour comprendre les possibilités, le fonctionnement et les modèles de capteurs.

Dans les travaux pratiques précédents, l'accent a été placé sur le traitement automatique de l'information et sur la mécanique du microcontrôleur. Par contre, peu d'effort a été fait sur l'acquisition de l'information à traiter et les entrées en général. Il est maintenant temps de voir comment des capteurs peuvent évaluer des paramètres du monde extérieur et les traduire sous une forme accessible au microcontrôleur.

Il est important de garder en tête que le microcontrôleur traite des signaux numériques, mais que les phénomènes physiques dans l'entourage du robot ont une nature profondément analogique. Par analogique, on entend une grandeur dans un domaine continu, et donc dans

lequel il existe une infinité de valeurs possible entre deux points. La température et la vitesse peuvent être classées dans cette catégorie. Cependant, un système numérique, par définition, traite des valeurs discrètes. Un capteur dans notre système devra donc fournir un signal directement sous forme numérique ou encore sous forme de valeurs de tension. Dans ce dernier cas, cette possibilité existe puisque le microcontrôleur possède un convertisseur analogique/numérique intégré pour passer du monde analogique à un monde basé sur des valeurs représentées par des zéros et des uns.

Certains capteurs pourraient être qualifiés de «simples» puisqu'ils se connectent plus ou moins directement au microcontrôleur. Ces capteurs peuvent être des interrupteurs, des détecteurs de niveau, des encodeurs de priorité, des encodeurs de position (comme la roulette d'une souris), etc. Leur caractéristique commune est qu'ils fournissent des valeurs binaires déjà décodées ce qui simplifie l'interface au microcontrôleur.

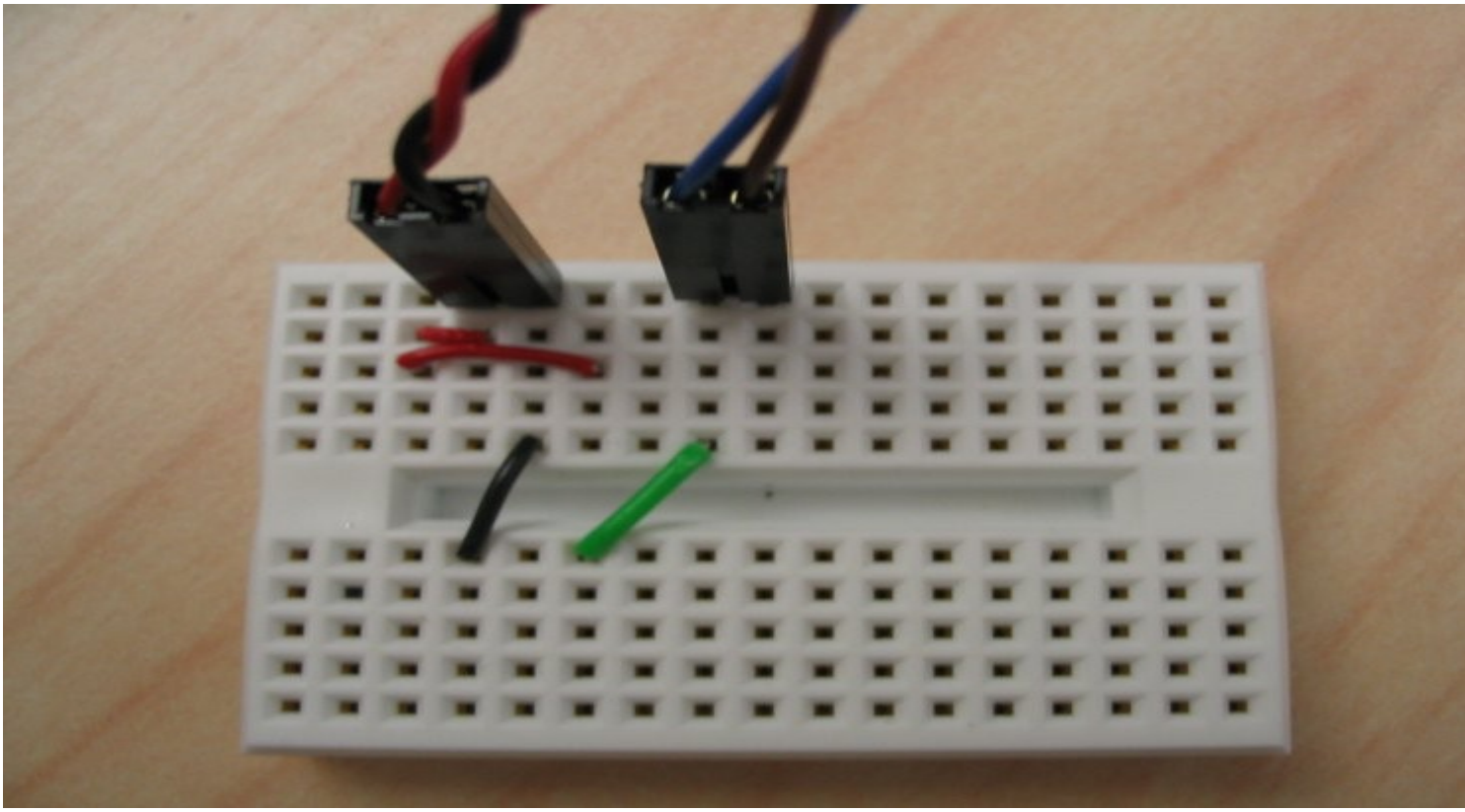
D'autres capteurs peuvent avoir des interfaces beaucoup plus sophistiquées. Un capteur avec une interface numérique avec un protocole est souvent capable de prendre des données plus complexes et précises. Ils peuvent utiliser des bus de tout type tel SPI (Serial Peripheral Interface), I^2C (Inter Integrated Circuit), ou encore une communication série asynchrone ou parallèle. Ils sont souvent remarquablement autonomes et fiables, surtout s'ils possèdent des mécanismes intégrés de correction d'erreurs. Étant donné leur système de contrôle numérique, ces capteurs ont une tendance à coûter plus chers que les capteurs simples. Les exemples de tels capteurs sont nombreux: sonars, caméras spécialisées, thermomètres, détecteurs de fumée, accéléromètres, etc. Ces capteurs peuvent souvent être vus comme des systèmes informatiques complets en eux-mêmes. Il est probable que vous utiliserez de tels capteurs dans quelques semaines.

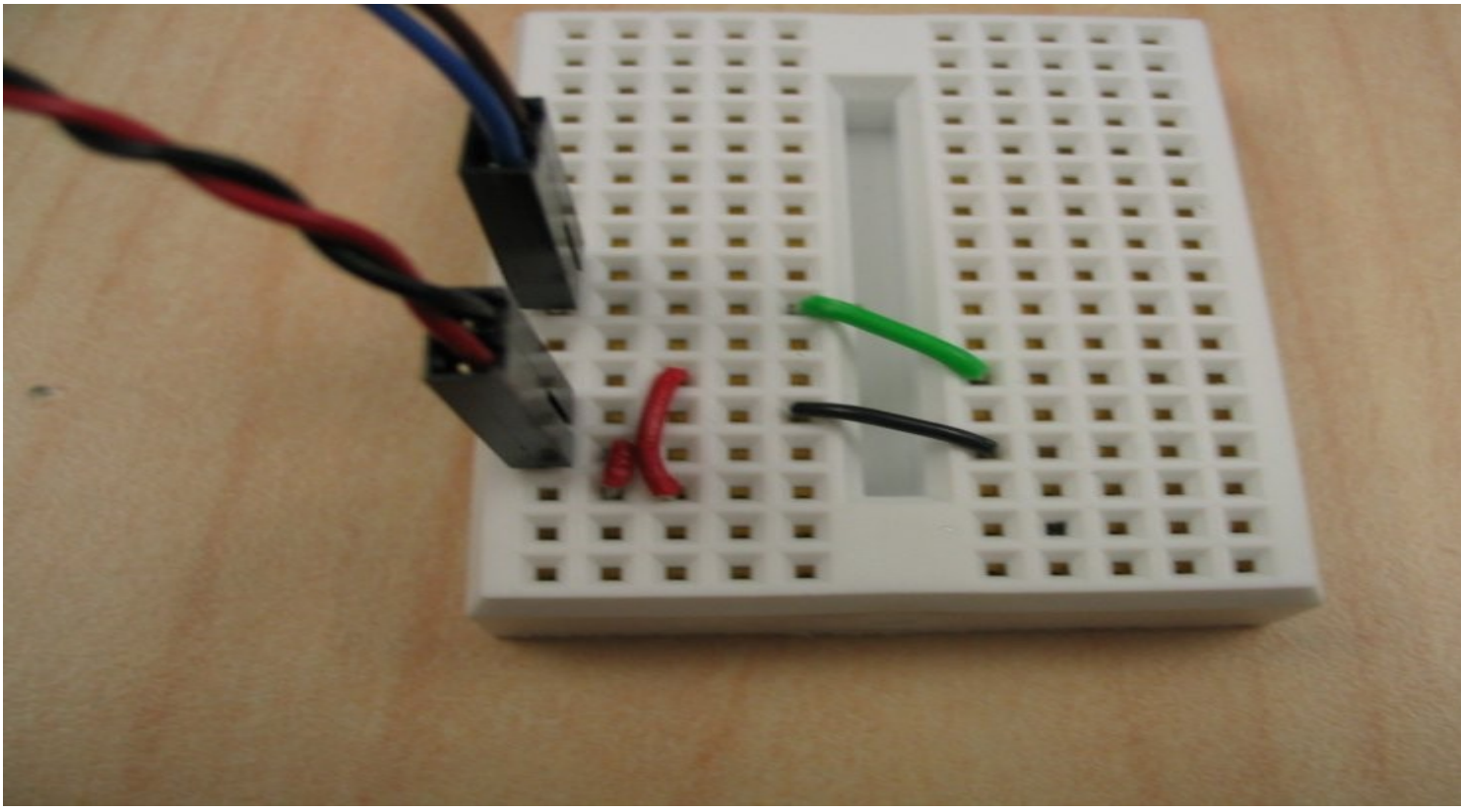
Un ATmega324PA est doté de ports capables de convertir un signal analogique (voltage) en données numériques. Il le fait grâce à un convertisseur analogique numérique (CAN). Le microcontrôleur peut effectivement lire jusqu'à 8 signaux du port A. Le CAN fonctionne en graduant une plage délimitée par deux voltages en 2^{10} valeurs. La valeur la plus basse pouvant être représentée est 0 volt et la plus haute est la référence analogique (5 volts ou moins). La valeur numérique fournie en sortie est proportionnelle à celle de la tension d'entrée. L'utilisation du CAN est nécessaire lorsqu'un système a recours à des capteurs dépourvus de capacités de communication numérique, même les plus simples. On peut donc placer ces capteurs dans une troisième catégorie, celle des capteurs avec interface analogique. Le deuxième problème de ce laboratoire montrera comment utiliser une photorésistance comme capteur avec interface analogique.

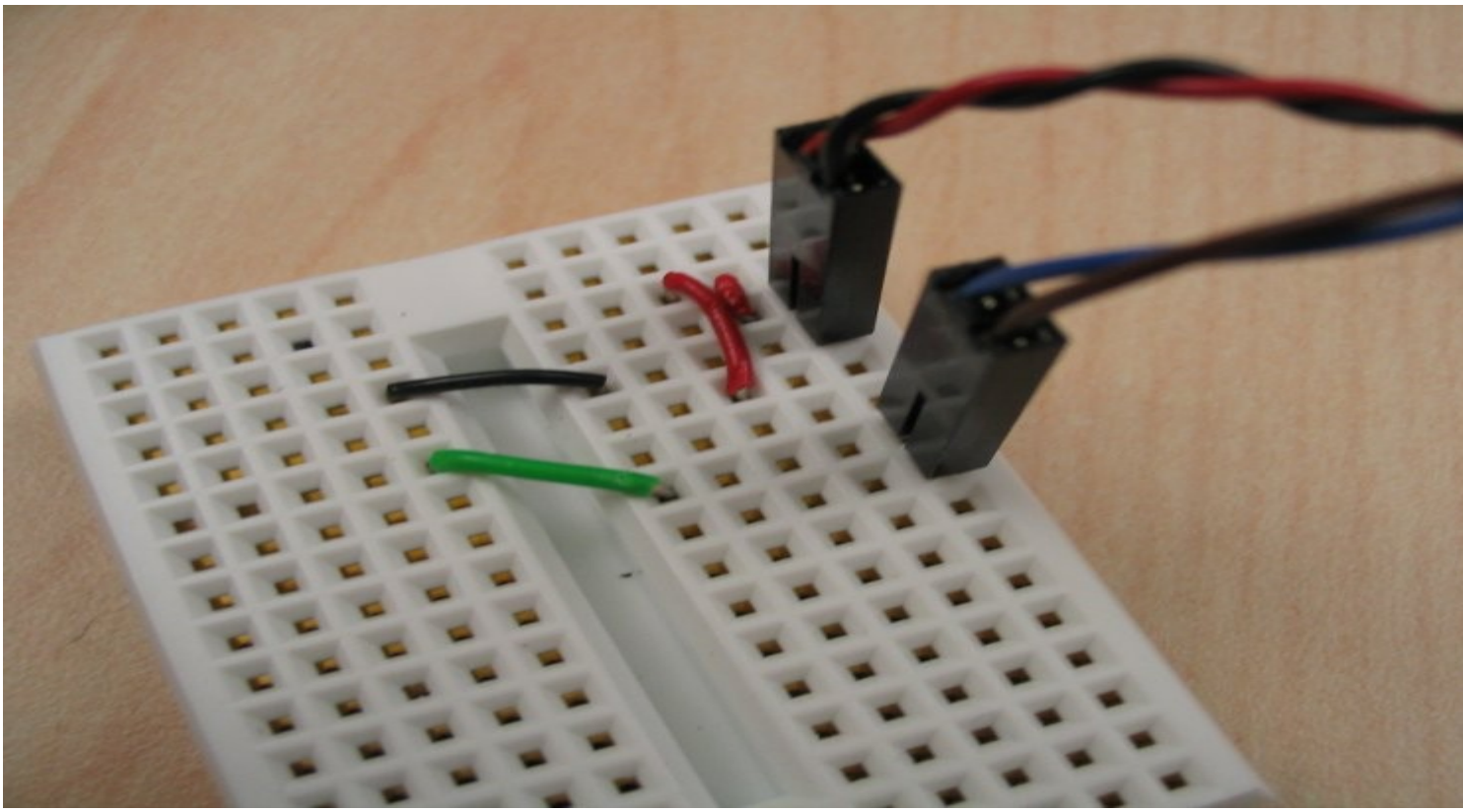
Pour bien comprendre la nature d'un capteur simple, il faut avoir quelques notions de circuits électriques de base. La lecture de cette section du site web est importante.

Problèmes

Avant de passer aux deux problèmes plus bas, il faut placer quelques fils sur un *breadboard*. Ces fils seront utilisés pour les deux problèmes. Par contre, chaque problème complétera le circuit en rajoutant quelques éléments différents. Trois images montrent comment placer les fils de base. Les fils rouges devront aller à la tension d'alimentation (Vcc) et les noirs à la masse (GND). On reliera donc le câble formé de la paire rouge et noir à une paire de broches appropriées sur la carte mère. Le fil de données (en bleu) du câble brun-bleu sera relié à des broches différentes en fonction du problème.



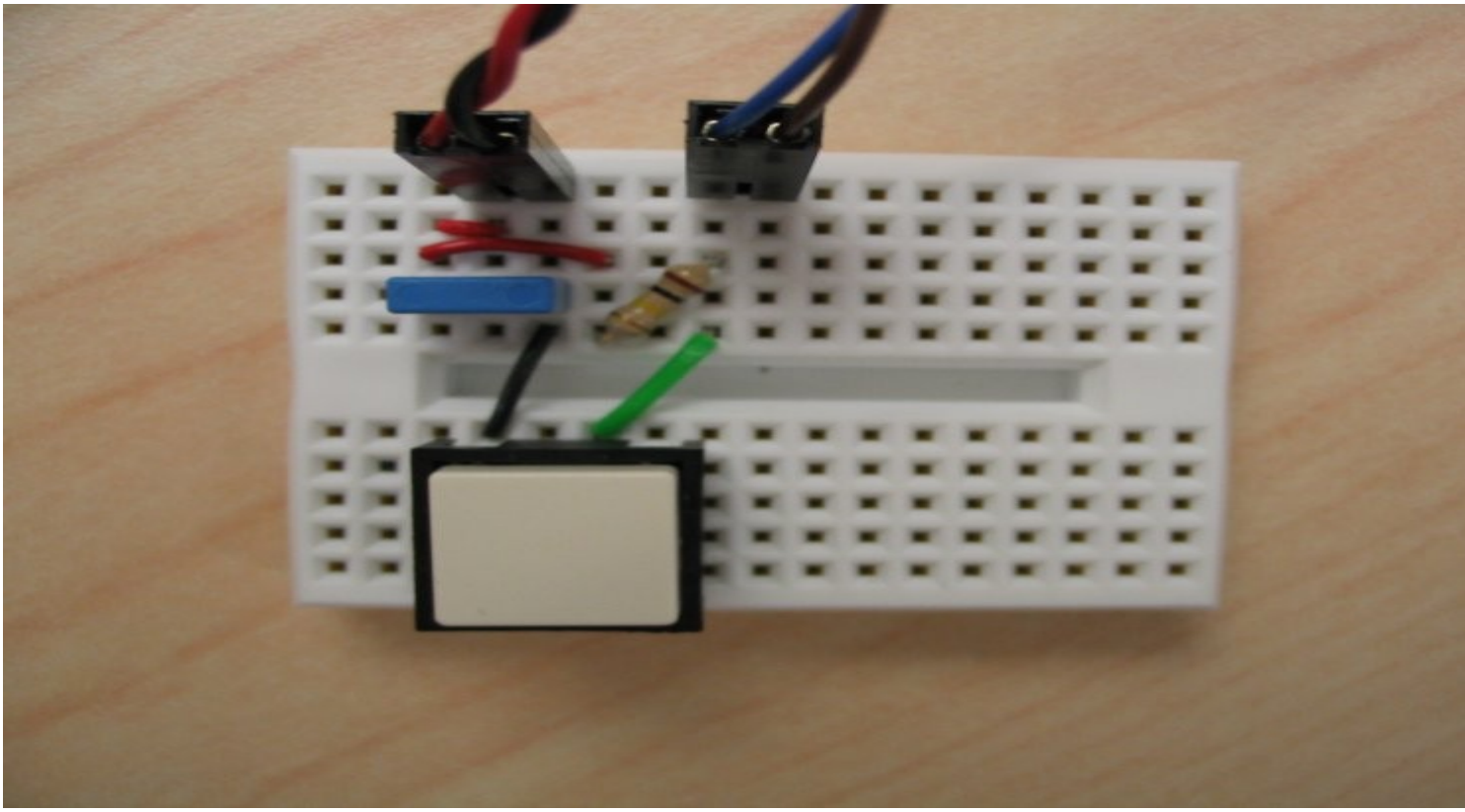


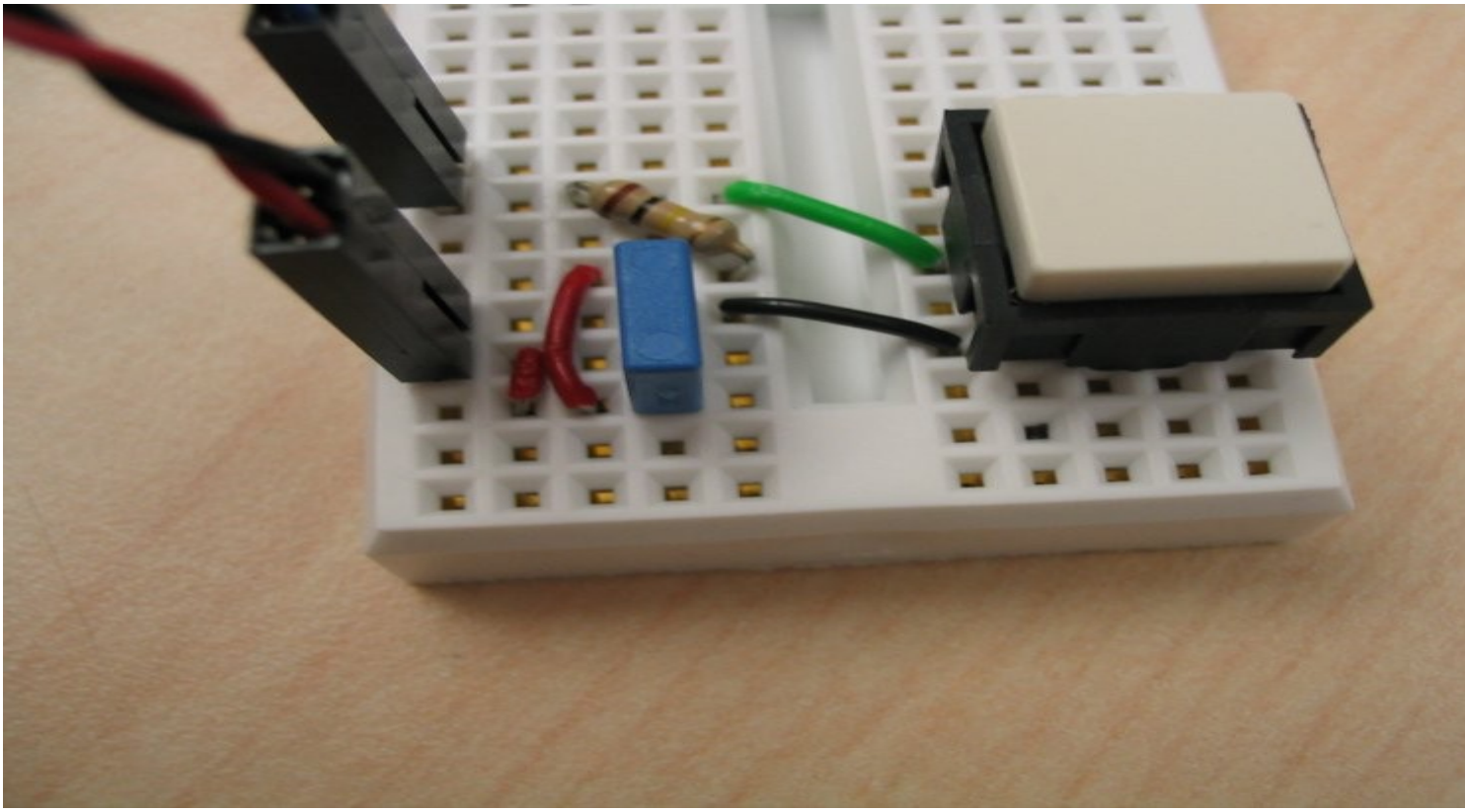


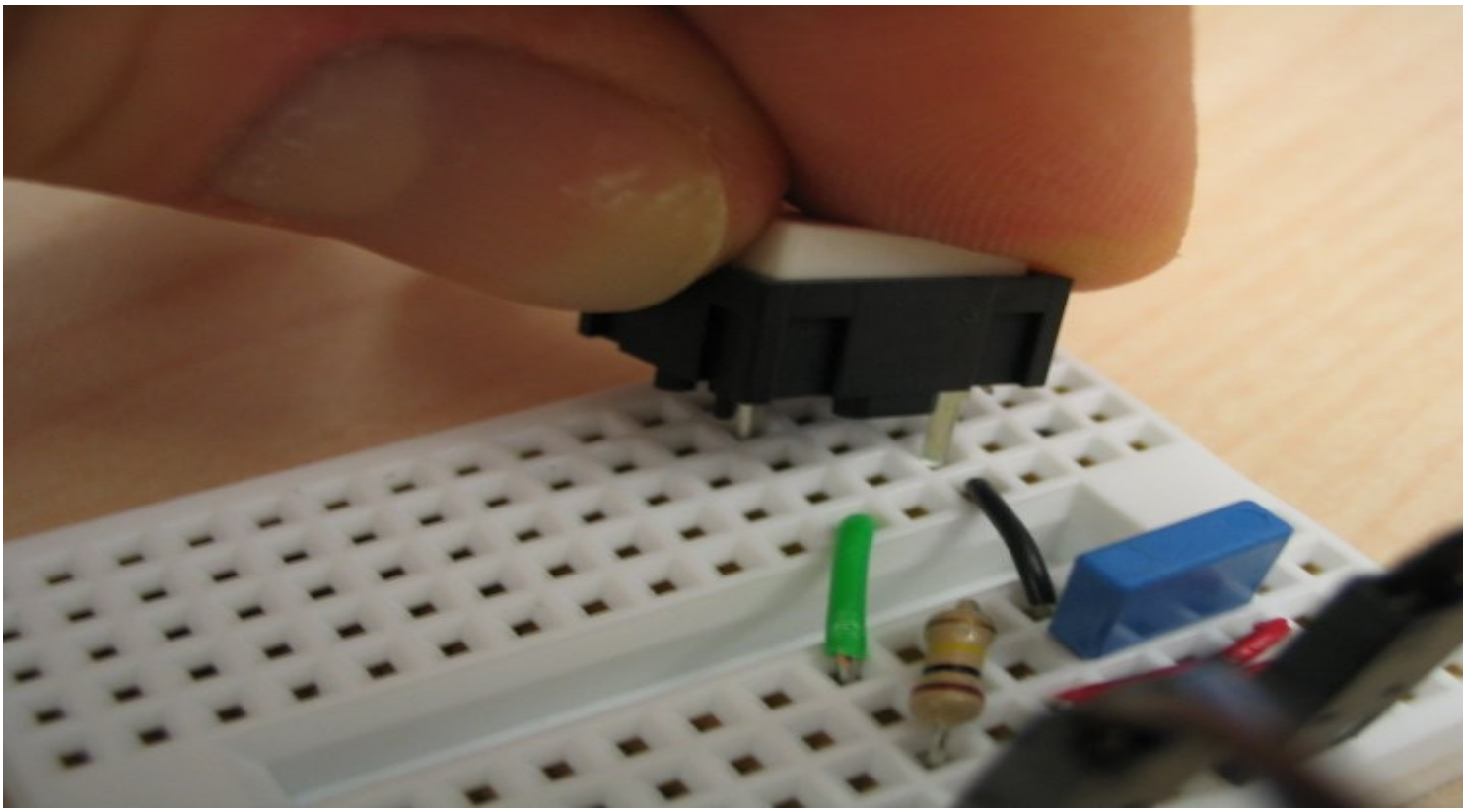
Problème 1

Connectez un bouton-poussoir blanc (Digi-Key: EG1328-ND) au *breadboard*. Bien suivre le détail des photos pour une insertion correcte. Placez également un petit condensateur (gris ou bleu) de 0.1 µF (Digi-Key: BC1621-ND) pour réguler l'alimentation au haut, à gauche. Enfin, insérer une résistance de 100K (brun-noir-jaune) (Digi-Key: S100KQTR-ND) un peu plus à droite.

Dans ce circuit, lorsque le bouton-poussoir est relâché, aucun courant ne peut circuler et la tension sur le signal de sortie (bleu ou vert)) est Vcc. Lorsqu'on appuie sur le bouton, le circuit est fermé et la résistance de 100K limite le courant. Le signal de sortie passe à 0 volt. Connectez le signal de sortie au port PD2 de la carte mère. Attention, le cavalier *IntEN* doit donc être retiré pour éviter que le bouton-poussoir *Interupt* de la carte mère vienne créer un conflit. Il est absolument nécessaire d'utiliser ce bouton-poussoir blanc sur le *breadboard* pour résoudre le problème. Celui sur la carte mère doit demeurer inactif. On vous demande aussi de relier votre DEL libre à B0-B1.







Quand le bouton est enfoncé, un compteur qui incrémente 10 fois par seconde est activé.
Quand le bouton est relâché ou lorsque le compteur atteint 120, la lumière clignote vert

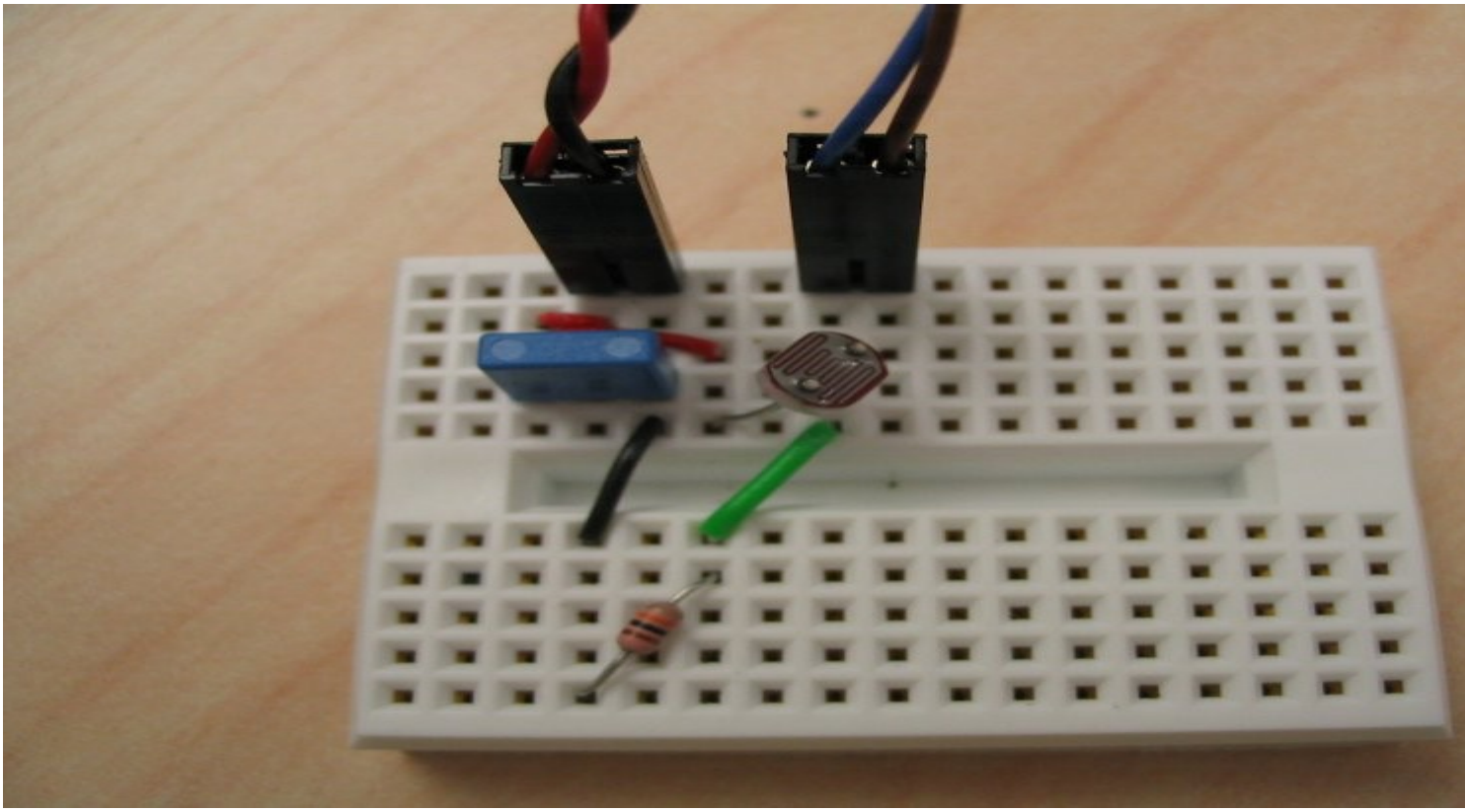
pendant 1/2 seconde. Ensuite, la carte mère ne fait rien. Puis, deux secondes plus tard, la lumière rouge s'allume. Elle devra clignoter (compteur / 2) fois au rythme de 2 fois par seconde. Ensuite, la lumière tourne au vert pendant une seconde. Finalement, elle s'éteint et le robot revient à son état original.

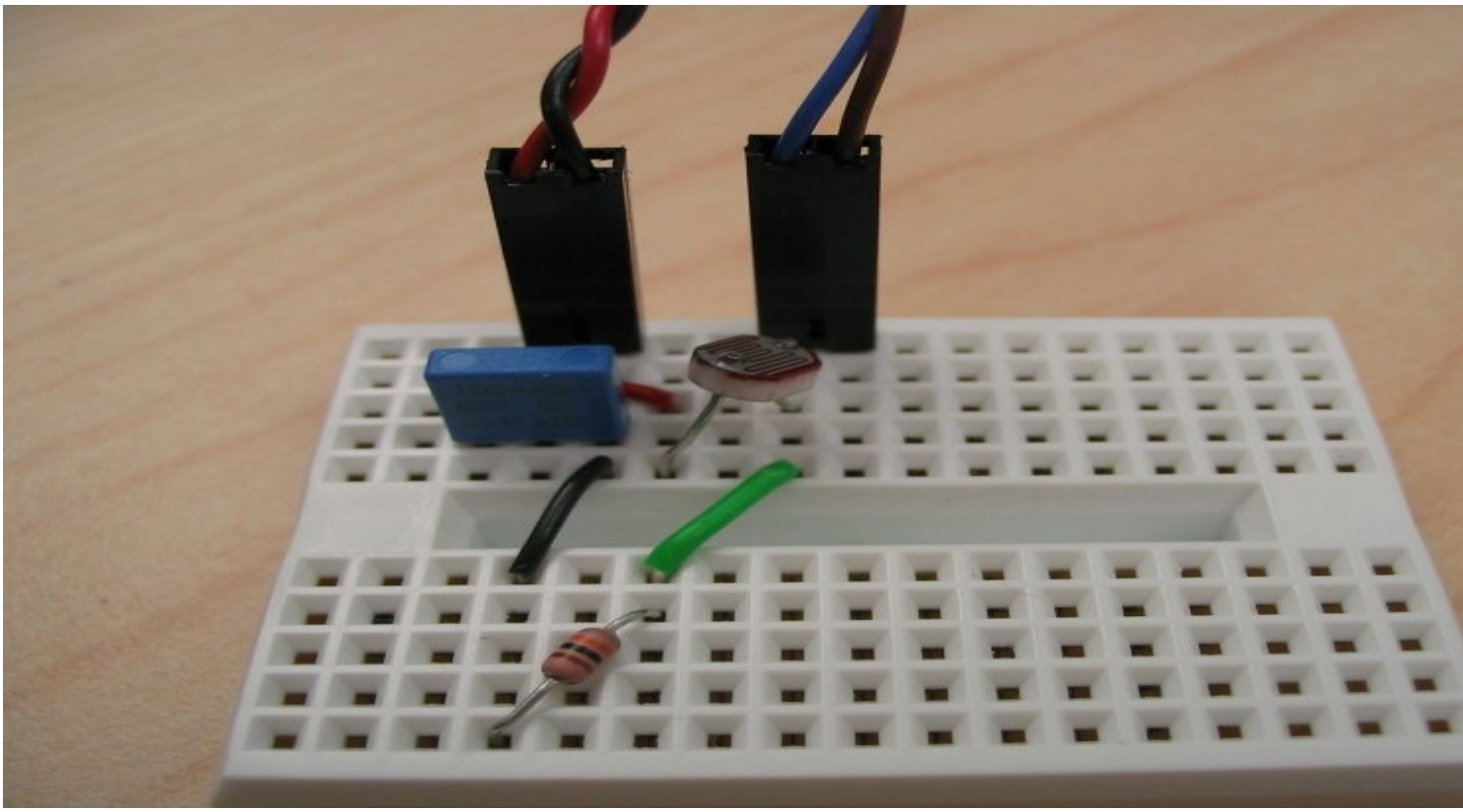
Problème 2

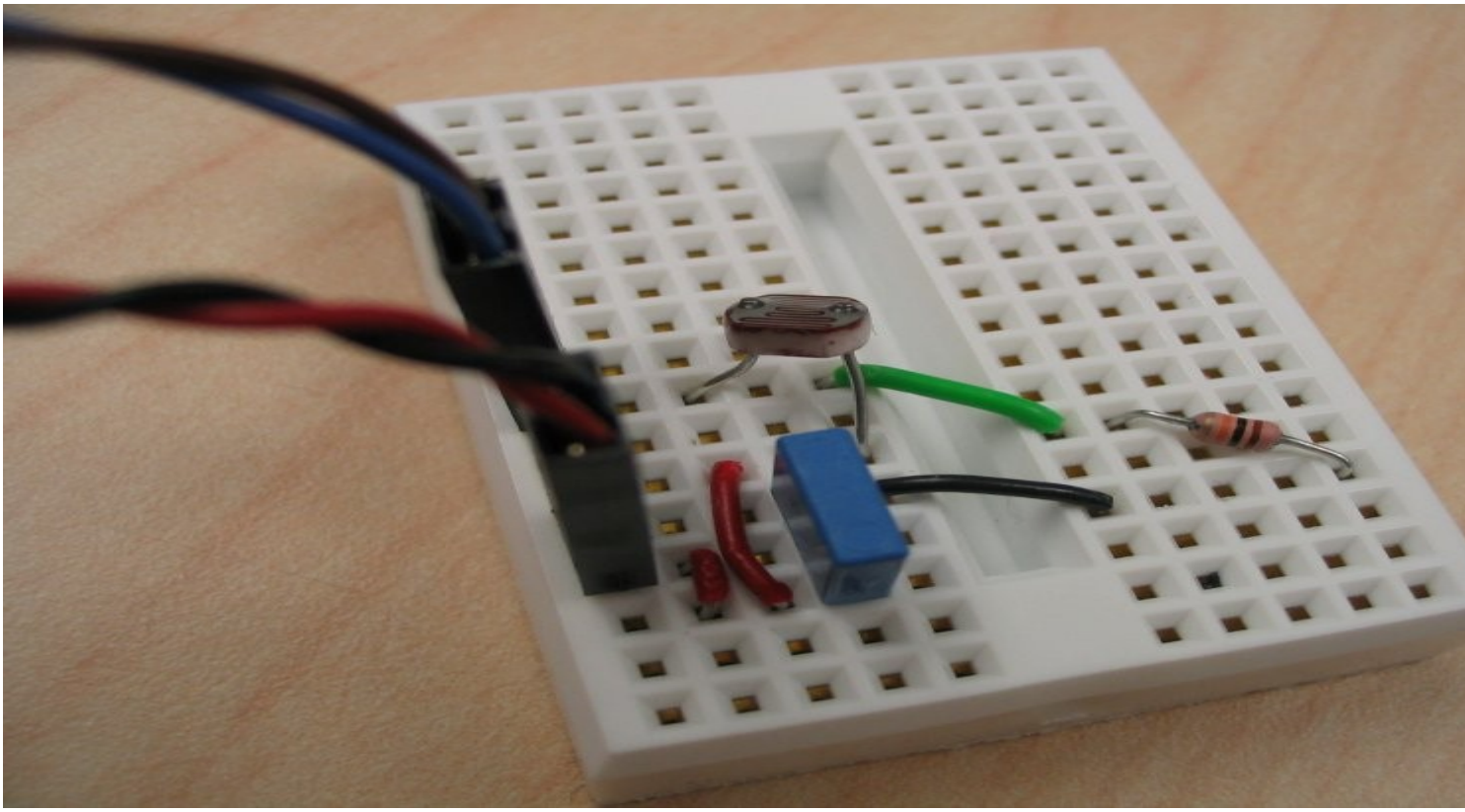
Retirez le bouton-poussoir et la résistance de 100K du montage du problème précédent. Insérez une résistance de 10K (brun-noir-orange) (Digi-Key: S10KQTR-ND) à l'endroit où le bouton-poussoir était. Placez la photorésistance (Digi-Key: PDV-P8101-ND) où la résistance de 100K était précédemment. Reliez le fil de données à un port analogique de votre choix (de PA0 à PA7).

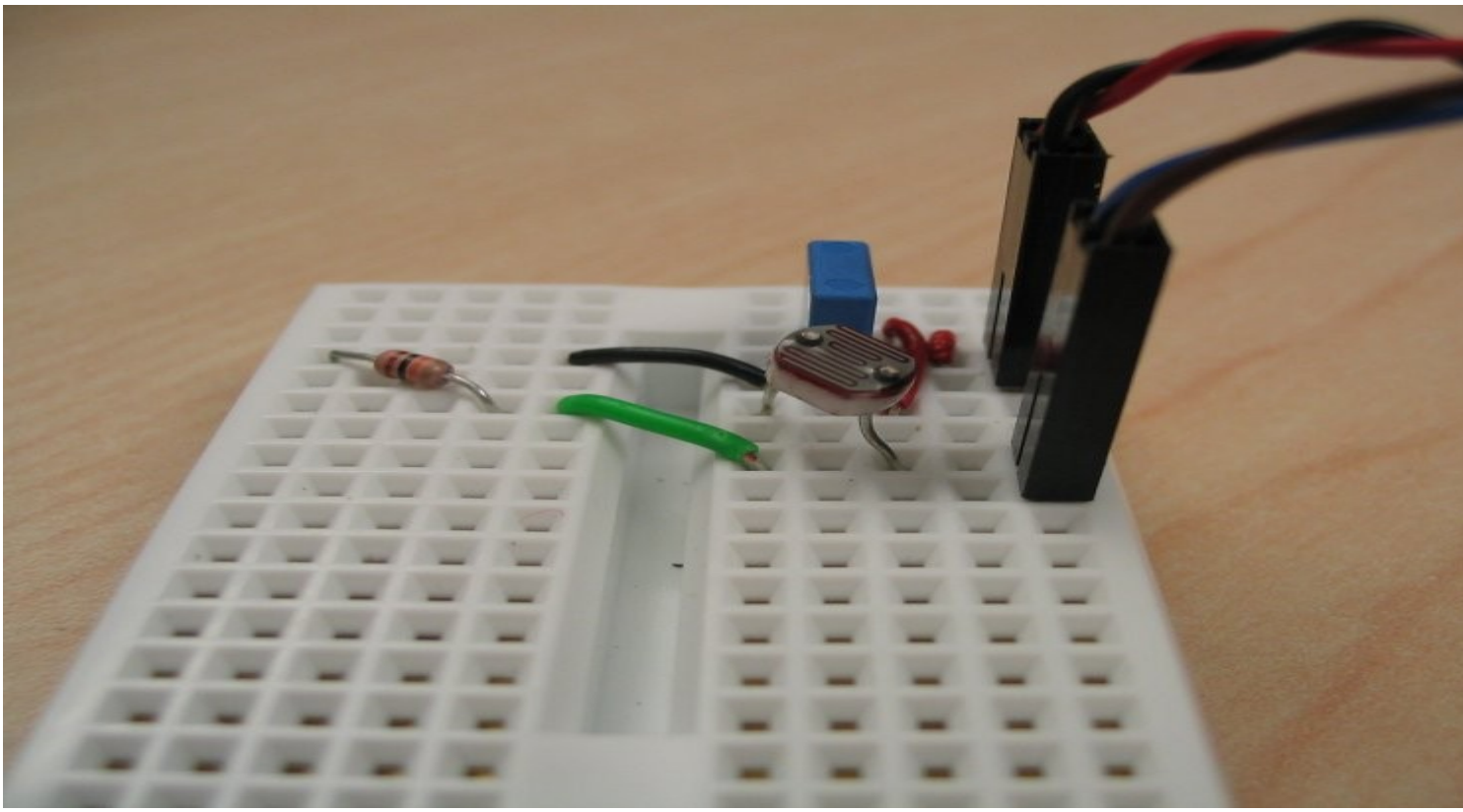
La photorésistance est une résistance qui varie (de 4K à 11K dans le présent) en fonction de l'intensité lumineuse. On l'a placée ici en série avec une résistance de 10K. La tension au point de contact entre ces deux résistances variera donc en fonction de la lumière. On peut mesurer cette tension avec le convertisseur analogique/numérique du microcontrôleur. On peut aussi l'évaluer avec le multimètre lors de la mise au point pour être bien certain.

La valeur retournée par le convertisseur est de 10 bits. Par contre, les derniers 2 bits sont sûrement sans signification et peuvent donc être négligés ce qui permet de réduire le format des données à 8 bits. Le traitement des données s'en trouvera simplifié. Utiliser le port A0 pour effectuer vos conversions. On vous demande aussi de relier votre DEL libre à B0-B1.



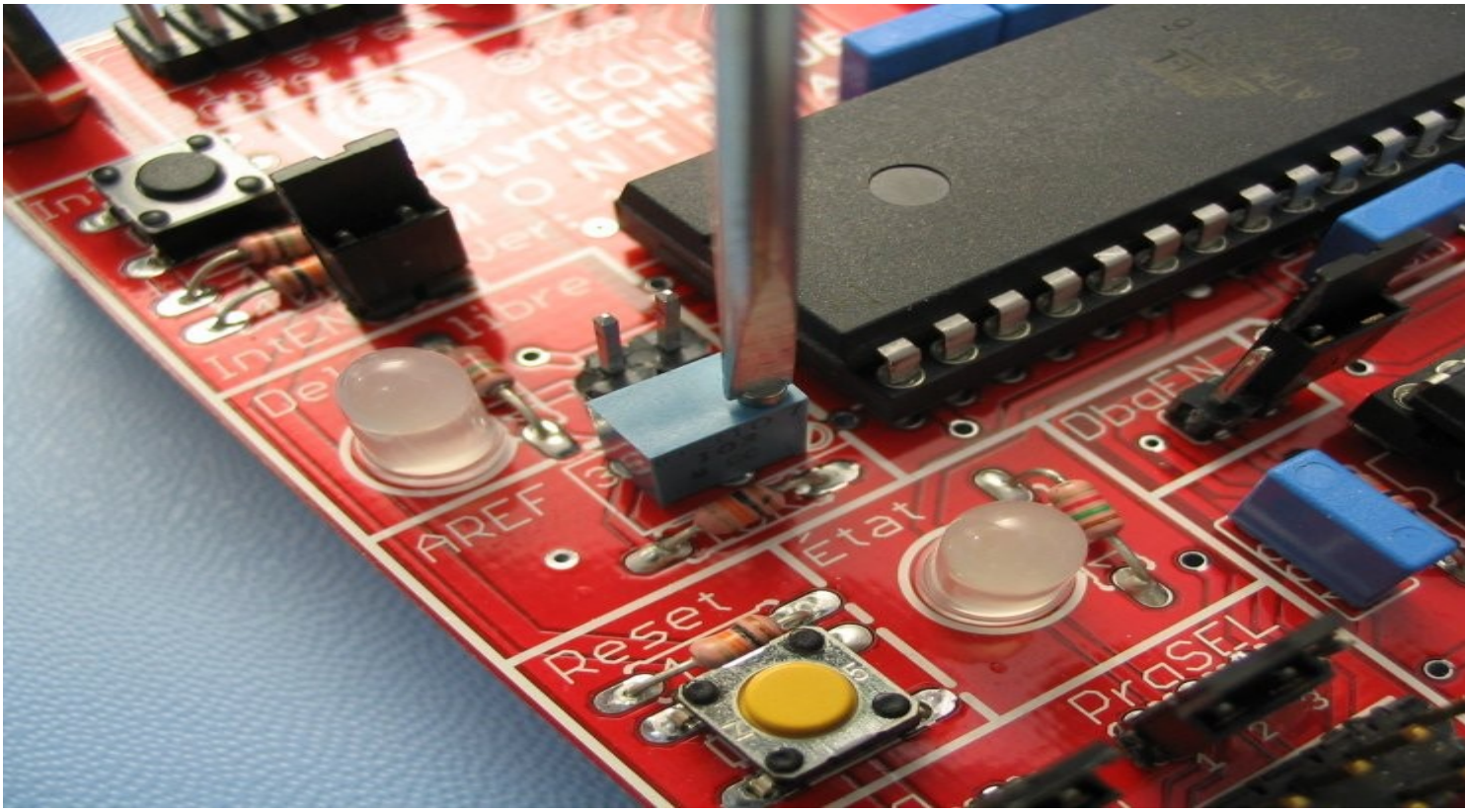






Une classe (can.h et can.cpp) permet le contrôle du convertisseur analogique/numérique. Référez à la documentation d'Atmel pour bien comprendre le fonctionnement de ce périphérique interne. De plus, vous devrez ajuster la référence analogique à +5.0 volts pour

cet exercice. Concrètement, il faut ajuster le potentiomètre de manière à régler le voltage de sortie du diviseur de tension sur la carte mère pour que la broche 32 (AREF) du ATmega324PA soit à 5 volts. Il vous faudra donc utiliser le multimètre et votre petit tournevis à tête plate pour y arriver. L'alimentation de la carte peut difficilement se faire par le câble USB dans ce cas car la tension fournie par l'USB est inférieure à 5 volts. Il vaut mieux utiliser la source de tension au laboratoire (ou, au pire, une pile rectangulaire 9 volts) pour alimenter la carte.



Faire un robot pouvant se faire bronzer...
Si la lumière est basse (en cachant la photorésistance), la DEL prendra la couleur verte.
Si la lumière est à un bon niveau (lumière ambiante), la DEL tournera à l'ambré.
Si la lumière est trop forte (photorésistance sous une lampe de poche), la DEL devient rouge.

Questions intéressantes à vous demander...

- Avec un CAN de 10 bits, combien de valeurs possibles peut-on représenter?
- Si Vref = 5 volts et Vgnd = 0 volt, que donnera le CAN à 2.65 volts?
- Proposez une architecture logicielle où l'on prend des valeurs d'un CAN et où on les enregistre en mémoire. Comment enregistrerait-on des valeurs de 10 bits dans une mémoire de 8 bits?
- Proposez des méthodes permettant d'enregistrer plus de données de 10 bits dans les 64K de mémoire.

Soumission du programme

Ces deux programmes seront corrigés. Ils devront être placés dans votre entrepôt Git. Pour faciliter la correction, les programmes devront être dans un répertoire précis et être accompagnés d'un Makefile. *XY* est votre numéro d'équipe. Vous pouvez nommer les fichiers de la façon que vous le désirez à l'intérieur des deux répertoires suivant, tant que le code peut être compilé avec la commande «make»:

Il faut se souvenir que Linux est sensible à la casse (*case sensitive*) et que les chemins précédents doivent être exacts. De plus, il faudra placer sous Git les fichiers `can.h` et `can.cpp` pour que le code puisse être compilé correctement. Il est possible que vos fichiers soient directement annotés par les chargés de laboratoire durant la correction. Un fichier `correction.txt` sera placé dans le répertoire tp6 lorsque la correction sera terminée. Les programmes seront corrigés selon le barème de la section évaluation du site web.