Projet Oxymètre de pouls

20h

➢ Objectifs et déroulement :

L'objectif du projet est de réaliser en binôme un oxymètre de pouls communicant sans fils en utilisant les différents éléments vus en cours et TP :

- Capteur communicant en liaison série
- Microcontrôleur
- Module radio

Le projet se déroule en 5 séances de 4h chacune. Le projet comporte des points de validation. Certains points de validation sont guidés comme dans un TP. D'autres sont plus libres et peuvent être réalisés de plusieurs façons. Dans tous les cas vous devez expliquer votre code pour valider un point. Vous commenterez vos codes et les déposerez dans un dossier seafile partagé dont le lien est sur Moodle. Le dépôt sera un fichier .zip à vos noms contenant les codes de tous les points de validation réalisés.

➢ Information sur le fonctionnement d'un oxymètre de pouls :

Le capteur mesure la concentration en oxygène dans le sang grâce à une mesure d'absorption de la lumière traversant un doigt. On les trouve par exemple dans les hôpitaux en service réanimation ou comme équipement pour diagnostiquer l'apnée du sommeil.

Figure 1 : Oxymètre de pouls sur un doigt (l'oxymètre pince le doigt)

Un oxymètre de pouls mesure principalement le pourcentage d'hémoglobine qui transporte de l'oxygène (l'oxyhémoglobine $Hb0_2$). Ce taux est ce qu'on appelle la saturation en oxygène ou SpO_2 lorsqu'il est mesuré avec un oxymètre. Il permet aussi de mesurer le pouls de la personne car la saturation en oxygène est modulée par le pouls.

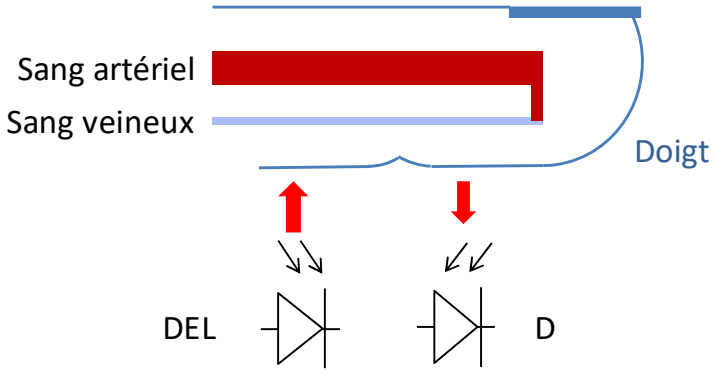

Figure 2 : Schéma de principe de la mesure d'absorption par diffusion dans le doigt

Une grande partie de l'absorption du doigt est constante. Cependant une partie dépend du sang circulant dans les veines. Une petite partie de l'absorption due au sang artériel est modulée par le pouls (figure 3). C'est cette mesure d'absorption qu'on souhaite réaliser. Elle représente moins de 2% de l'absorbance globale. L'absorbance A est définie comme suit :

$A = -log_{10}(\frac{I}{I_0})$ avec I, l'intensité de la lumière transmise et I_0, l'intensité incidente.

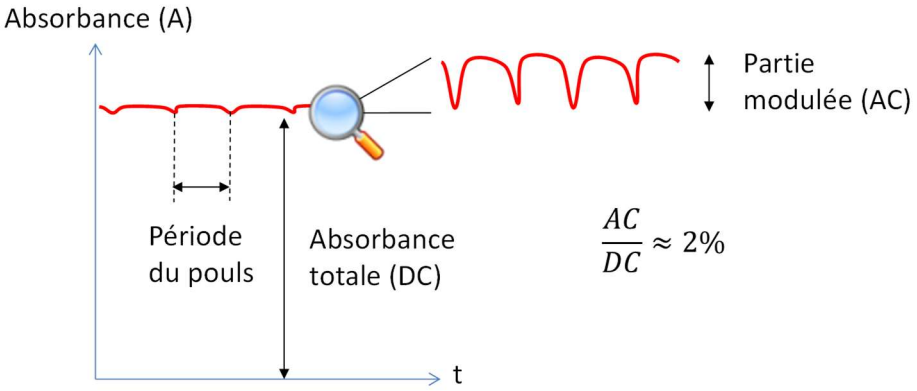

Figure 3 : Schéma de principe de la mesure d'absorption à travers le doigt

L'absorption dépend du coefficient d'absorption (en cm^{-1}). La partie pulsée dépend en particulier du coefficient d'absorption de l'hémoglobine non oxygéné et de l'hémoglobine oxygéné. Pour un sang non saturé en oxygène le profil du coefficient d'absorption en fonction de la longueur d'onde sera donc intermédiaire (voir figure 4). Deux mesures (une à chaque longueur d'onde) sont nécessaires pour remonter à l'information de saturation $Sp0_2$. C'est l'amplitude des pulsations mesurées par la diode photoréceptrice qui est importante mais le pouls peut aussi être mesuré en même temps.

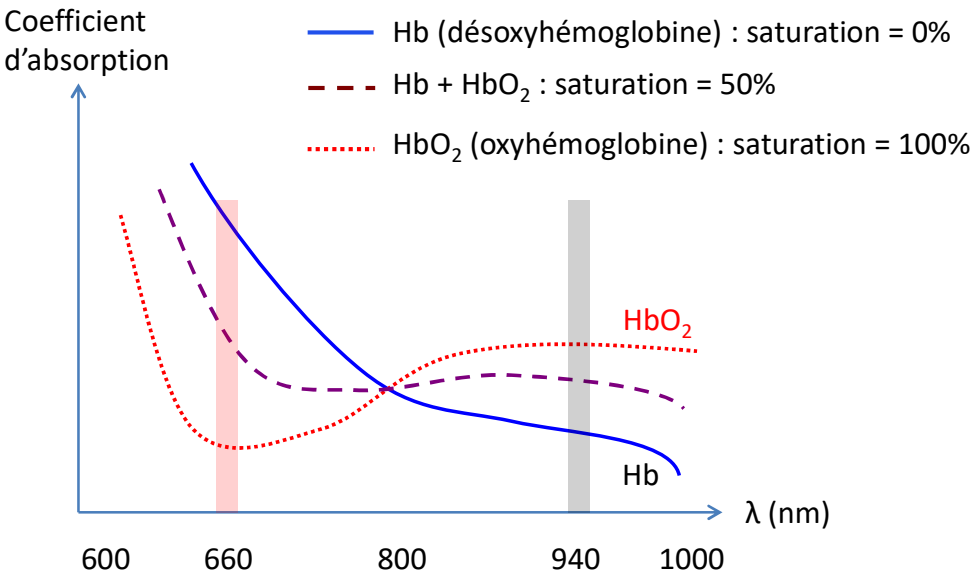

Figure 4 : Absorption en fonction de la longueur d'onde

Vous allez d'abord récupérer le signal du capteur le traiter avec le premier arduino et l'envoyer par transmission sans fils à un autre module Xbee. Ce dernier est connecté à un second arduino que s'occupe simplement de transférer les informations reçues au PC pour qu'elles soient affichées sur le traceur et le moniteur.

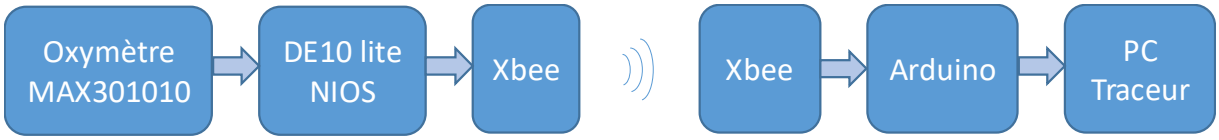

Figure 5 : Chaîne complète avec liaison sans fil

A chaque étape du projet vous allez créer une nouvelle version de votre code arduino dans un fichier .ino qui contiendra le numéro du point de validation. Chaque fichier sera dans un dossier avec le même nom comme l'exige Arduino IDE. Vous pouvez créer plusieurs versions pour la même étape pour pouvoir revenir en arrière.

1. Prendre en main l'arduino et l'adaptateur « shield »

Brancher l'adaptateur de type « shield » arduino et faire un programme qui allume la LED verte du shield quand on appuie sur le bouton et l'éteint si on appuie une seconde fois. Le « shield » se branche sur les PIN qui « power » de l'arduino d'un côté et des pins 8 à SCL de l'autre. Le brochage et le schéma de la carte se trouve dans la rubrique « Adaptateur » sur Moodle.

Remarques :

- vous devrez peut-être installer les cartes Arduino Uno R4 sur Arduino IDE. Pour cela aller dans l'onglet Tool > Boards > Boards Manager…
- cette fonction ne sert pas au projet global et permet juste de prendre en main le « shield » pour interagir avec l'arduino.

2. Vérifier que le max MAX30101 est bien connecté

En s'aidant de la biblothèque « wire » pour la communication série I2C, faire une fonction qui lit le « Part ID » de la puce et vérifier que la puce renvoie bien la valeur par défaut au POR. Cette fonction sera gardée dans le setup de votre programme final pour vérifier que la puce est bien connectée.

Les informations nécessaires pour communiquer avec le capteur se trouve dans la doc technique du MAX30101 qui est disponible sur Moodle. Il faut connaître l'adresse I2C du composant (voir les instructions pour lire un registre, figure 10, page 28, « slave ID ») et l'adresse du registre dans lequel se trouve le PART ID (regarder dans le « register map » de la datasheet, page 12 et 13). Après une lecture, afficher le résultat sur le moniteur et un message d'erreur si la communication se passe mal. Il est bien de faire une fonction pour lire un registre du MAX30101 au choix et une fonction pour lire le PART ID. La première pourra être réutilisée pour récupérer d'autres informations du capteur et la seconde pourra être utilisée à l'initialisation du microcontrôleur dans tous les programmes réalisés pour vérifier que le capteur est bien branché.

Remarques :
- On utilisera l'adaptateur de type « shield » arduino pour brancher le capteur. Faîtes attention à l'orientation. Le dessus est indiqué sur la nappe reliée au capteur.
- Le part ID est 0x15, c'est-à-dire 21 en décimal (valeur POR : « power on reset »). Si vous avez une réponse différente alors que la communication se passe bien, vérifiez que vous affichez correctement le résultat (lecture des bits dans le bon ordre par exemple).

Les fonctions de la bibliothèque « wire » sont décrites ici : https://docs.arduino.cc/language-reference/en/functions/communication/wire/

Utiliser pour reproduire les signaux de la figure 10, page 28, de la datasheet :
Wire.beginTransmission(address) : pour envoyer le « slave ID »
Wire.write(data) : pour envoyer l'adresse du registre
Wire.requestFrom(address, quantity) : pour lire le nombre d'octet indiqué. L' « address » est le slave ID
Wire.read() : renvoie le prochain octet demandé à l'adresse indiquée par les fonctions précédentes

Attention, il est nécessaire de lancer la communication I2C par la fonction Wire.begin() qui peut être placée dans le setup().

Wire.available() : peut être utiliser pour savoir si les données sont disponibles après l'utilisation de Wire.requestFrom()
Wire.endTransmission(stop) : Renvoie 0 s'il n'y a aucun problème ou un chiffre indiquant l'erreur de transmission s'il y en a une. Afficher l'erreur sur le moniteur série permet de débugger. « stop » permet de mettre un condition de stop et libérer le bus I2C si « stop » est true. Si « stop » est false la communication I2C peut continuer.

3. Allumer la LED rouge du MAX30101

Configurer le MAX de façon à allumer la LED rouge du MAX30101 en écrivant dans les bons registres. Le registre « LED Pulse Amplitude » est décrit à la page 20. Il faut aussi bien configurer le registre Mode

Configuration. On pourra vérifier que les bonnes valeurs ont été écrites dans les registres en les lisant et les affichant sur le moniteur série. L'écriture dans un registre est décrite sur la figure 9 (page 27).

Remarque : En annexe sur Moodle vous trouverez la configuration des registres de configuration du capteur conseillé dans le document « ANNEXE oxymètre » . Il sera possible de le changer pour améliorer l'oxymètre.

4. Récupérer les données du MAX30101

Coder et utiliser des fonctions pour lire les valeurs rouges et infrarouges de la photodiode. Les afficher sur le traceur série.

Pour lire en continue placer un délai de 10ms dans la boucle infinie. L'échantillonnage réalisé ne sera pas fait à une fréquence très précise mais cela devrait suffire pour reconnaitre le signal de pouls Attention toutefois aux affichages sur le port série qui peuvent ralentir la boucle, surtout si la transmission se fait avec une vitesse lente. Il est préférable d'augmenter cette vitesse de transmission sur le port série pour pouvoir débugger avec le moniteur sans trop ralentir la boucle.

La lecture de la valeur des capteurs se fait sur la pile FIFO. La figure 2.b à la page 16 résume le fonctionnement de la FIFO dans le mode conseillé. Avant chaque lecture, le pointeur de lecture de la FIFO sera réinitialisé à 0 pour lire toujours le même échantillon. Après chaque lecture le pointeur de lecture de la FIFO sera réinitialisé à 0 pour ne pas que la FIFO déborde. Puisque la période d'échantillonnage du MAX30101 est de 1ms et que celle du NIOS est de 10ms environ, 9 échantillons seront perdus. Une amélioration consiste en lire toutes les valeurs enregistrées dans la pile FIFO pour les moyenner et en faire une seule moins bruitée.

Pour enregistrer un tracé du signal de pouls il faut copier suffisamment de données de la console du moniteur pour avoir plusieurs battements de pouls et aller dans Excel. Attention le signal alternatif de pouls est petit par rapport à l'offset. Faire ces tracés en rouge puis en infrarouge. En temps réel vous pouvez utiliser le traceur série de l'arduino IDE (Tools > serial plotter…). Sur le traceur le signal d'une seule LED pourra être observé.

5. Echantillonnage des données du MAX30101

En utilisant une interruption sur timer, échantillonner les données avec une période de 2ms. Afficher les valeurs sur le traceur série de la même façon qu'au point précédent. Moyenner les valeurs si la FIFO contient plusieurs échantillons.

6. Récupérer les valeurs DC

Pour chaque longueur d'onde, faire une fonction qui récupère la valeur moyenne du signal de photodiode sur une durée réglable suffisamment grande pour moyenner plusieurs battements de pouls. La durée sera comptée en nombre d'interruption du TIMER. Cette valeur servira au calcul du SPO2. Afficher cette valeur dans la console pour valider.

7. Récupérer le signal AC

Pour chaque longueur d'onde, récupérer le signal AC des photodiodes R et IR. L'afficher sur le traceur série pour voir votre signal de pouls. Voir l'annexe pour la fonction de transfert du filtre passe haut. Faire aussi un filtre passer bas pour améliorer le signal.

8. Afficher le pouls

Faire clignoter la LED verte du shield au rythme du pouls.

9. Mesurer le BPM

Faire une fonction qui mesure le BPM et l'affiche sur le moniteur série.

10. Mesurer les amplitudes AC

Pour chaque longueur d'onde, faire une fonction qui mesure l'amplitude du signal AC.

11. Mesurer le S_pO_2

Pour la mesure du taux de saturation en fonction des valeurs de chaque photodiode, on utilisera la formule empirique suivante basée sur un polynôme second degré :

S_PO_2 (%) = 108 - 20R - 0,375R^2

Où R est le rapport des amplitudes relatives des signaux rouge et infrarouge : R=(ACr/DCr)/(ACi/DCi)

ACr et ACi sont les amplitudes totales (Max(OUT_AC) – Min(OUT_DC)) pour respectivement le rouge et l'infrarouge. DCr et DCi les valeurs moyennes de OUT_DC précédemment calculées.

12. Faire un menu

L'arduino devra se trouver dans 3 configurations possible et changer de configuration à l'appui sur le bouton :

1- Affichage du BPM et clignotement LED au rythme du pouls
2- Afficher les données du capteur brutes pouvant être lues sur le traceur série
3- Afficher le BPM et le S_p0_2 sur le moniteur série

13. Prendre en main les modules radio Xbee

Configurer les modules Xbee avec XCTU et envoyer les données lues sur le port analogique A0 de l'arduino vers un moniteur XCTU sur le PC.

Remarque :

- les Xbee sont configurés en mode transparent et s'utilise avec l'UART.
- Un module Xbee se branche directement sur le shield avec un petite carte adaptatrice fournie. L'antenne est orientée vers l'extérieur du shield.
- XBEE_TX et XBEE_RX sont reliés aux ports 12 et 11 de l'arduino comme indiqué sur le schéma du shield.
- comme il ne s'agit pas des ports de l'UART de l'arduino, il faudra utiliser un objet SoftwareSerial dans le code arduino comme dans l'exemple donné sur Moodle.

14. Afficher des données sur Matlab

Utiliser le code exemple donné pour afficher les données lues sur le port analogique A0 de l'arduino sur Matlab.

Remarque :

- On utilisera un arduino UNO R3 qui sera le second arduino.
- On peut les afficher sous forme de graphique ou de texte.
- Attention au format des données.

15. Afficher les données du MAX sur Matlab

Reprenez le code arduino avec des menus et envoyer les données destinées au moniteur série vers Matlab via les modules radio. Pour les données brutes des capteurs afficher dans un graphique.

Remarques :

- Utiliser une platine d'essai pour connecter le module Xbee au second arduino.
- Attention le TX du Xbee doit être connecté au RX de l'arduino.