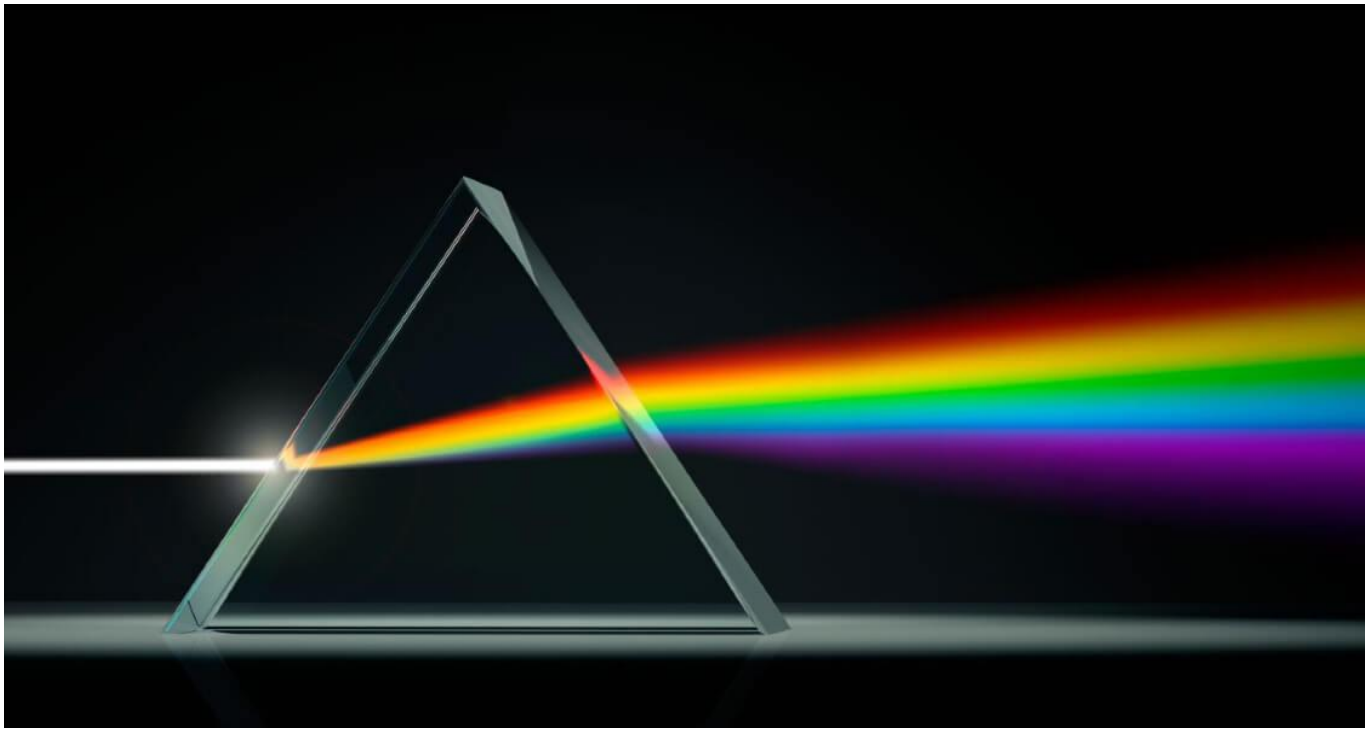
Safa BIRYG
Loic Clarot
1A G4

RAPPORT TECHNIQUE

Projet d'Electronique de Première Année à l'Institut d'Optique

Spectromètre à réseau



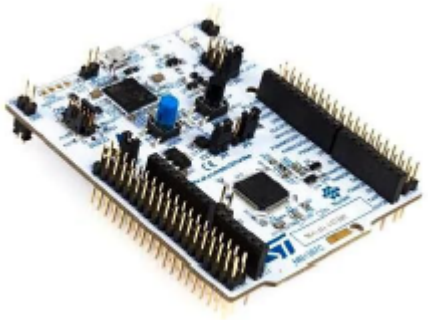
Présentation :

Nous présentons un système qui permet d'afficher le spectre d'une source lumineuse.
On utilise pour cela un goniomètre qui renvoie le spectre physique vers une barrette CCD connecté à l'ordinateur par une carte nucleo.
Une application programmée sur Matlab permettra alors d'afficher le spectre directement et modifier les paramètres de l'acquisition.

Les exigences du système :

- Utiliser un réseau dans un goniomètre pour diffracter la source lumineuse .
- Capter le flux lumineux à la sortie du goniomètre avec une caméra CCD 64 pixels TSL 201.
- Faire l'acquisition de 64 valeurs reçues depuis les 64 photosites de façon synchronisée et cyclique grâce à une carte nucleo .
- Récupérer le tableau des valeurs qui constituent le spectre sur PC grâce à une communication RS232.
- Permettre d'afficher le spectre directement sur une application codée sur AppDesign sur Matlab.

Barrette CCD 64 pixels TSL 201



Carte Nucléo

Le système présente des contraintes au niveau électronique surtout la synchronisation des horloges par la carte nucleo pour acquérir les valeurs captées par les 64 photodiodes pour chaque cycle de manière séparées.La communication entre la carte nucleo et le pc pose des défis sur le niveau informatique car nécessite d'utiliser les deux langages de programmation C/C++ et Matlab pour réaliser la communication RS232 avec une vitesse de 115200 baudes qui permettra de récupérer le tableau qui contient le spectre par le PC et d'envoyer la valeur du temps d'exposition pour adapter la sensibilité du spectromètre à l'intensité de la source et éviter de saturer le spectre afficher.

Découpage fonctionnel :

Voici le schéma de toutes les fonctionnalités du système et la fonction de chaque composant .

schéma fonctionnel :

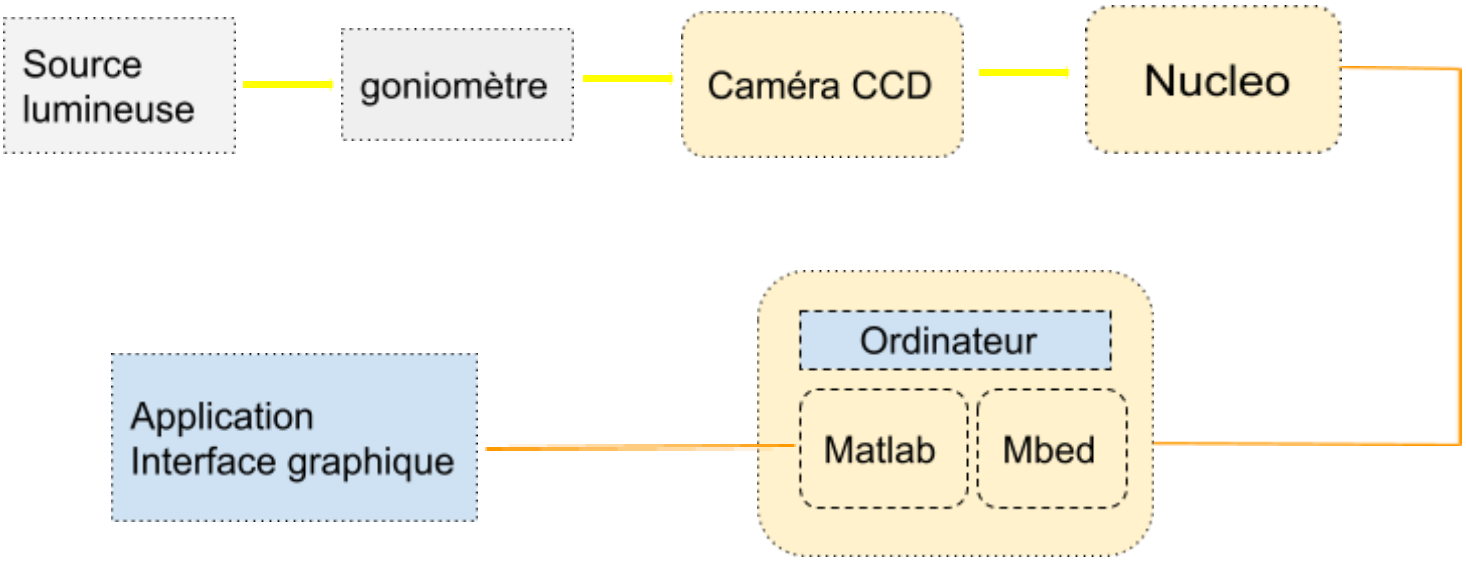


Schéma électrique :

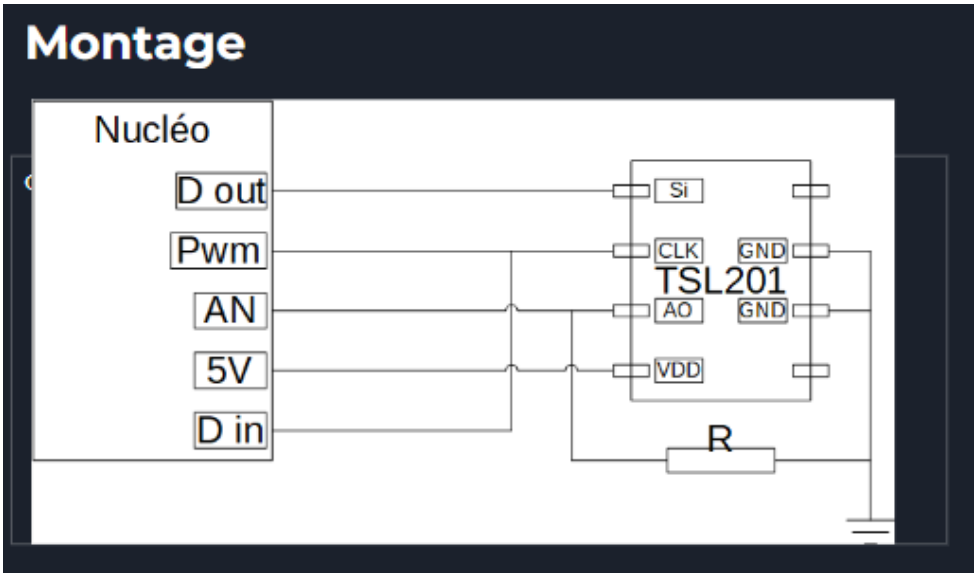


Schéma du circuit électronique réalisée sur la carte enfichable

Fonctionnalitées réalisées :

Voici les étapes que nous avons réalisé pour arriver au projet final et répondre aux exigences citées précédemment :

Partie Optique et électronique :

- Capter le spectre à la sortie du goniomètre par la caméra CCD
- Brancher la caméra CCD à la carte Nucleo

Partie matérielle :

- Gérer et synchroniser les mesures par la carte nucleo qu'on programme sur Mbed
- Transférer les données vers l'ordinateur par le port USB - Port série virtuel

Partie interface :

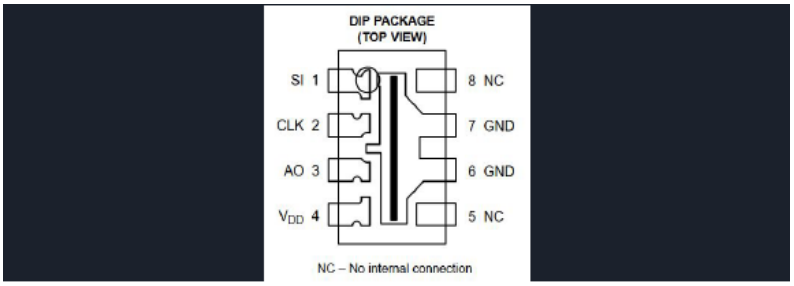
- Etablir une communication RS232 entre la carte nucleo et l'ordi avec une vitesse de 115000 baudes en utilisant le port USB - Port série virtuel.
- Cette communication permettra de renvoyer le spectre vers le PC et de récupérer la valeur du temps d'exposition.

Partie logiciel / Interface graphique :

- Réaliser une application sur App Design par Matlab qui communique avec le programme de la carte Nucleo réalisé sur Mbed.
- L'application doit permettre d'afficher le spectre en temps réel, de pouvoir le capturer et de renvoyer la valeur du temps d'exposition adapté à l'intensité de la source.

Utilisation du capteur CCD :

Le capteur utilisé est un capteur 64 pixels TSL201.

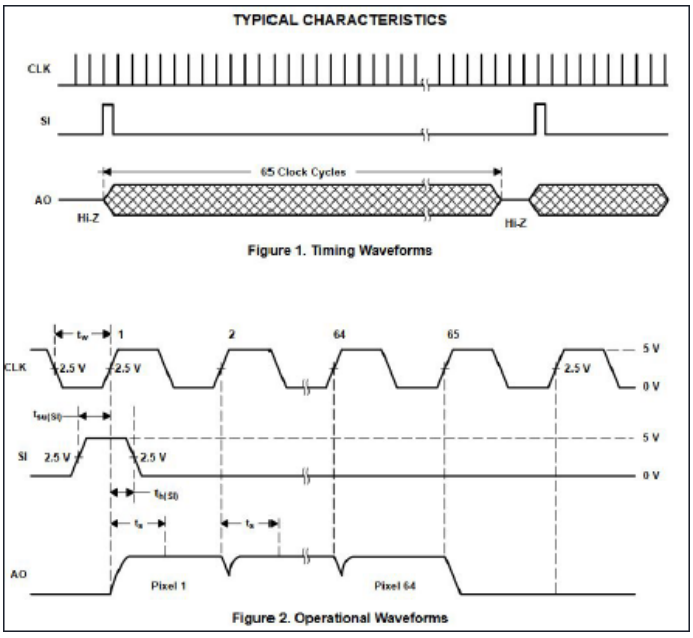


$$V_{out} = V_{drk} + (R_e)(E_e)(t_{int})$$

where:

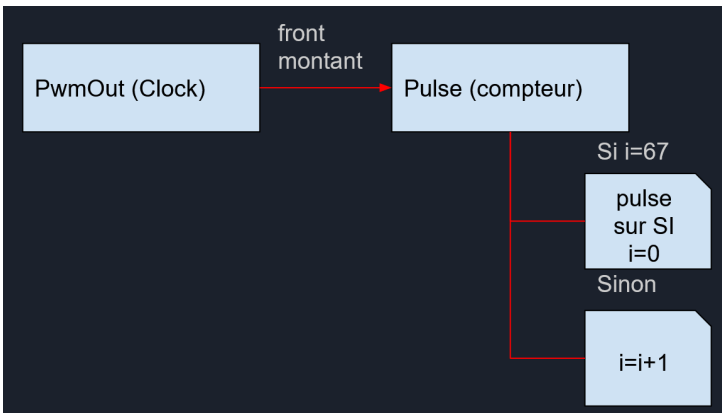
V_{out}	is the analog output voltage for white condition
V_{drk}	is the analog output voltage for dark condition
R_e	is the device responsivity for a given wavelength of light given in V/(μJ/cm²)
E_e	is the incident irradiance in μW/cm²
t_{int}	is integration time in seconds

Il a 8 broches, mais en réalité, seulement 5 broches sont utiles. Deux broches ne sont pas utilisées, et deux sont associées à la masse. C'est un composant actif : il doit être alimenté en 5 volt sur la broche V_{DD}. Sur les trois dernières broches, deux doivent recevoir un signal pour fonctionner et la dernière, la broche "Analog Output", renvoie le spectre capté sur les 64 pixels. Le signal "clock" donne la fréquence de lecture de chaque pixel, et le signal "SI" donne le départ pour la lecture du premier pixel.

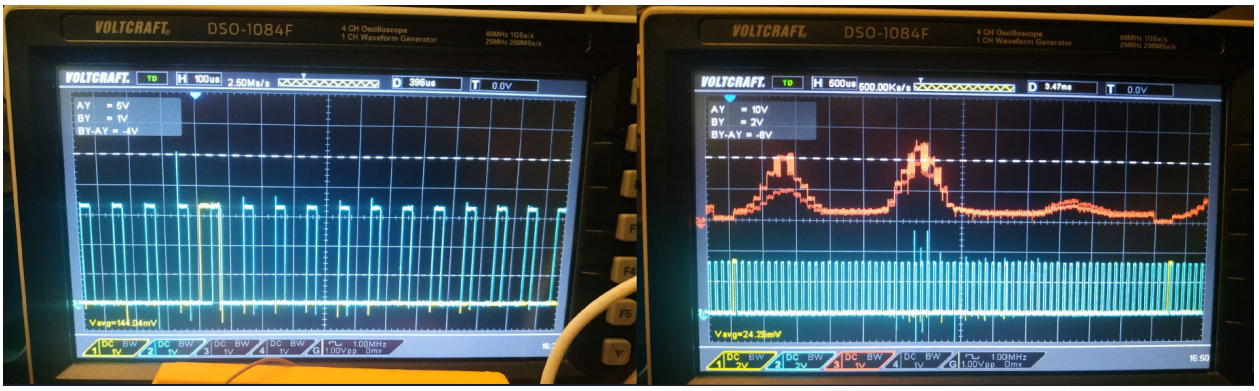


Pour la génération du signal clock, on utilise une sortie PWM pour générer un signal stable qui ne sera pas perturbé par les autres tâches que l'on demandera à la carte Nucléo. Ce signal fonctionnera en permanence. Pour la génération du signal SI, on envoie le signal Clock sur une entrée D# à laquelle on associe une fonction interruption. Cette fonction a pour rôle, pour l'instant, de compter le nombre de front montant du signal clock, et d'envoyer

une impulsion synchronisée avec le signal clock, toutes les 67 impulsions du signal clock sur le front montant qui suit.



Génération et synchronisation des signaux
Résultats expérimentaux



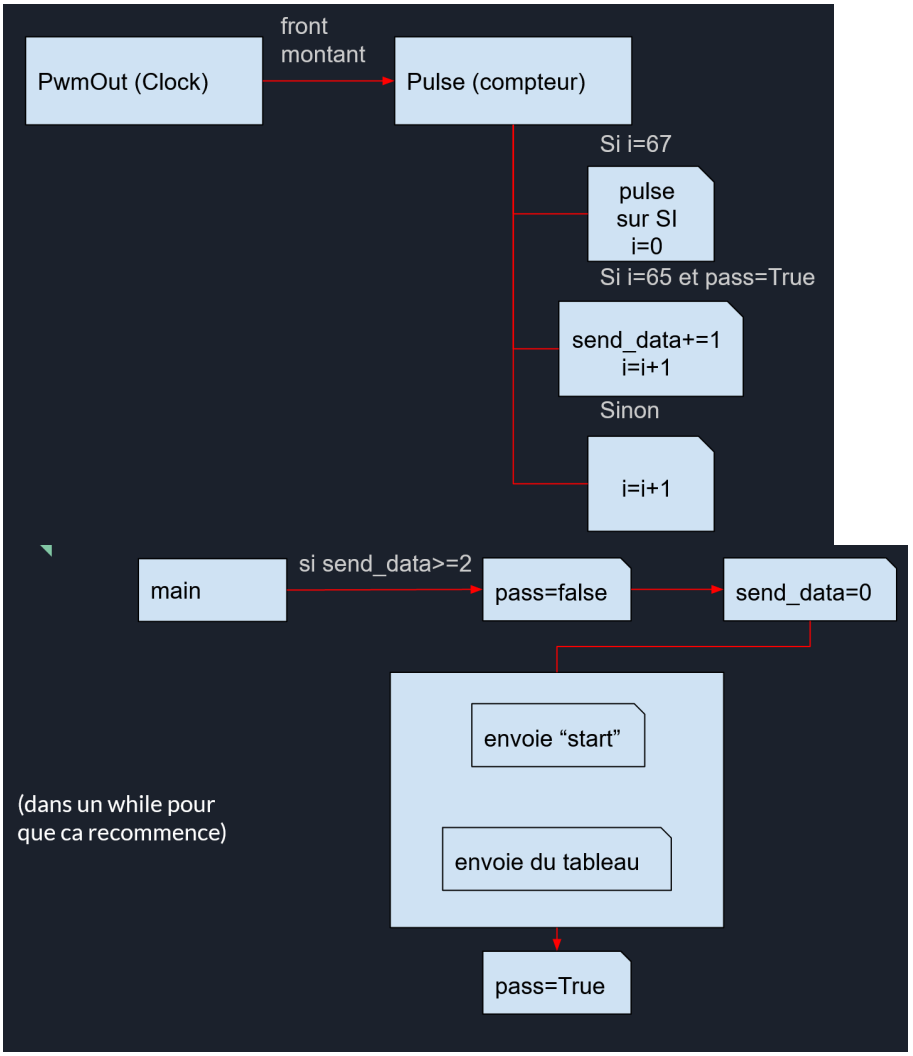
Pour acquérir le signal, il faut aussi synchroniser l'acquisition avec la clock. Pour cela, on peut le faire avec une fonction interruption, associée à un front descendant du signal clock. Le choix du rapport cyclique nous permet donc de choisir quand on fait l'acquisition. On a choisi un rapport cyclique de 0.3.
Pour savoir où écrire dans le tableau, on utilise le compteur utilisé dans la première fonction interruption.

Enregistrer le spectre et l'envoyer :

Le spectre est pour l'instant enregistré dans un tableau de 64 cases. On va mettre une variable dans la fonction interruption qui va compter le nombre de cycles que le compteur a fait. Dans la fonction main, on va faire un test sur ce compteur, qui lancera l'envoie du spectre à l'ordinateur et réinitialisera la variable. Mais nous avons quelques considérations à prendre en compte :

- Le tableau ne doit pas être modifié durant son envoi, pareil pour la variable. Il faut donc ajouter une condition au test pour ces actions, qui sera une variable booléenne qu'on changera avant et après l'envoi du tableau.
- On veut envoyer un tableau complètement modifié : lorsque nous autorisons la modification du tableau, nous sommes peut-être au milieu du spectre. Nous lancerons l'envoi du tableau lorsque nous aurons fait deux cycles.

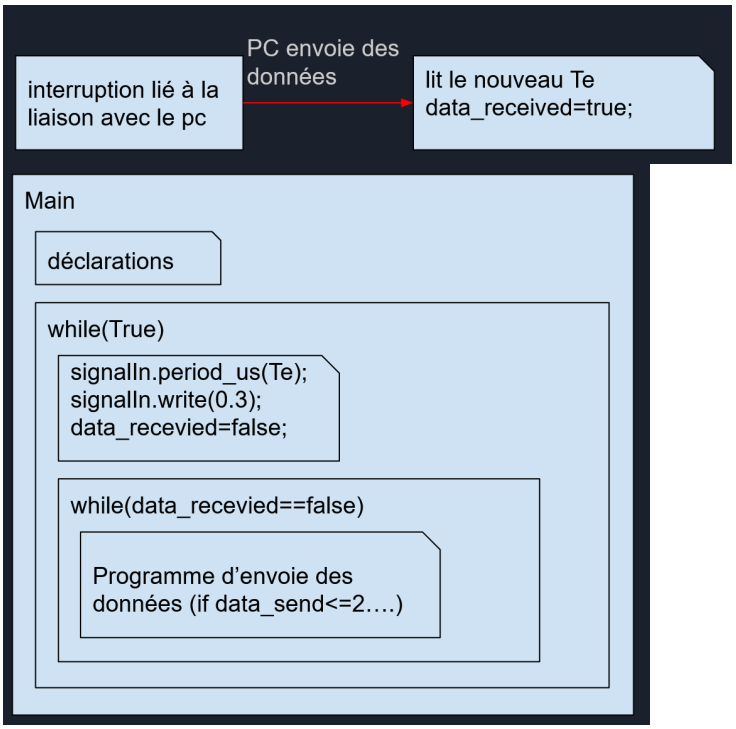
On nomme la variable qui compte le nombre de cycle "send_data" (c'est elle qui décide de l'envoi des données), et la variable booléenne qui autorise la modification du tableau et de send_data "pass".



Modifier le temps d'intégration :

La réception de données de la part de l'ordinateur n'est pas prévisible, nous associons donc une fonction interruption à la réception de données sur la liaison série. Elle modifiera aussi une variable booléenne qui informera le reste du programme que des données ont été reçues. Le temps d'intégration reçu de l'ordinateur est écrit dans une variable globale "Te". Dans le main, nous avions initialisé la sortie PWM. Nous créons une boucle while qui teste

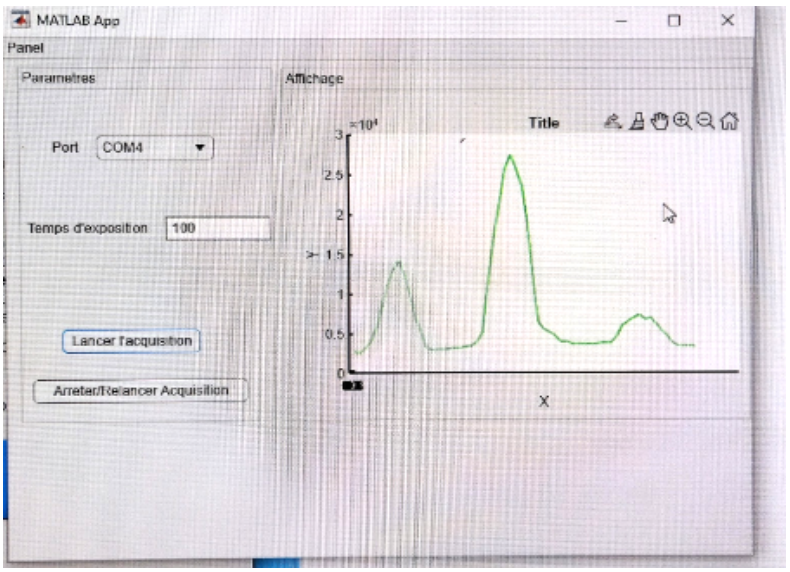
la variable booléenne, et dans laquelle on met la majorité du programme main. Seule une partie de l'initialisation de cette sortie PWM n'est pas dedans. Du coup quand la variable booléenne change, on sort de la boucle, et avec une autre boucle while, on refait l'initialisation de la sortie, et on change ainsi le temps d'intégration.



Programme Matlab

Le programme Matlab est censé recevoir les données du tableau et de l'afficher mais aussi de préciser la valeur du temps d'exposition pour le programme Mbed. La première chose est de préciser le port sur lequel il y aura le transfert des données, la fonction ***serialportlist()*** on importe les ports disponibles et l'utilisateur pourra donc choisir l'un d'entre eux. Une fois la connexion établie, les données sont transmises à une vitesse de 115200 bauds précisée par la fonction ***serialport*** .Pour renvoyer la valeur du temps d'exposition on utilise la fonction ***writeline*** et pour récupérer les valeurs du tableau on utilise la fonction ***readline*** .Enfin , l'affichage du tableau se fait par la fonction ***plot***.

L'interface graphique est faite de sorte à faciliter la lecture du spectre.



Organisation de travail :

séance 1 :

- Découvrir les différents composants utilisés notamment la caméra CCD à l'aide de la fiche technique.
- Se diviser les tâches.
- Réaliser le circuit électrique et visualiser le spectre sur l'oscilloscope .

séance 2 :

- Programmer la carte nucleo.
- Générer les horloges sur Mbed.

séance 3 :

- Établir une communication entre la carte nucleo et l'ordinateur en utilisant Tera Term.
- Résoudre les problèmes de synchronisation.

séance 4 :

- Utiliser un programme sur Matlab pour communiquer avec la carte nucleo (Annexe 2)
- Réadapter le programme Mbed pour communiquer avec le programme Matlab

séance 5 :

- Découvrir l'option de App Design sur Matlab pour créer l'interface graphique.
- S'assurer de la communication entre matlab et la carte nucleo pour bien récupérer le spectre.

séance 6 :

- Réaliser l'application sur App Design pour afficher le spectre de la source en temps réel .

séance 7 :

- Améliorer l'interface graphique pour pouvoir stopper l'affichage et puis le relancer grâce à un bouton.
- Ajouter la fonctionnalité sur l'application pour pouvoir entrer le temps d'exposition.

Difficultées rencontrées :

Les principales difficultés résident pour la partie d'acquisition dans la synchronisation des horloges , donc au niveau de la programmation sur Mbed il a fallu à chaque fois modifier le programme pour la bonne récupération du spectre.

Pour la partie programmation sur Matlab, il était plus difficile de travailler sur AppDesign comme il est nouveau pour nous et on a pas pu ajouter sur les axes plus d'informations comme l'ordre de grandeur des fréquences spectrales associées à chaque peak .

Analyse du travail en groupe :

La division des tâches nous a permis d'avancer plus vite car chacun était concentré sur sa partie, mais en même temps on s'entendait si nécessaire .