Laser-wave

2 Av. Augustin Fresnel,
91120 Palaiseau
(33) 01 23 45 67 89

Conversion MIDI-DMX

08 avril 2022

Vue d'ensemble

Le Laser Wave (LW) est une association étudiante de l'Institut d'Optique Graduate School (IOGS). Son rôle est de gérer les lumières d'ambiance en soirée (i.e. "lighter"). Son influence s'étend au-delà des murs de l'IOGS tant la lumière est notre spécialité. Les moyens dont dispose le LW pour lighter sont pour l'instant limités. Ils utilisent des ordinateurs personnels qui font le liens entre leur table de contrôle (qui envoie des informations MIDI) et leur lumière (pilotée en DMX) Le souci c'est qu'il faut reprogrammer les liaisons à chaque fois, ce qui est chronophage et pénible. Le LW a donc fait appel à nous pour régler ce souci afin de pouvoir plus facilement s'installer en soirée quel que soit le type de lampes qu'ils utilisent.

Le but de notre projet est en général de simplifier la vie aux personnes du LW.
Notre cahier des charges est constitué des contraintes suivante :

- Le système doit réaliser une conversion MIDI-DMX.
- Les conversions fonctionnent quelle que soit la lampe utilisée.
- Le système que nous mettons en place est simple à prendre en main par les lighters.
- commande le plus en temps réel possible

La solution technique que nous proposons est d'utiliser une carte nucléo pour réaliser cette conversion et que les lighters n'aient qu'à s'installer derrière leur système de commande. Pour ceux qui préfèrent encore utiliser leur ordinateur, nous souhaitons que cette option leur reste toujours disponible via une interface graphique téléchargeable grâce à un QR code.

Découpage fonctionnel

1. Description du matériel

Le système de conversion sera assuré par une carte nucleo.
Le contrôleur MIDI que nous utiliserons pour nos tests sera un PAD beaucoup utilisé par les membres du LW : un MPD-26. Ce dernier est composé de 16 pads envoyant comme information quel pad a été touché, à quelle vitesse (d'impact) et quelle pression y est maintenue (sustain). Il y a également 12 contrôleurs : 6 boutons que l'on peut translater et 6 que l'on peut tourner. Ils renvoient 2 information : quel contrôleur est manipulé et sa valeur (que l'on modifie en tournant/ poussant le contrôleur)
Leur lampe sont des bancs de lyres/ LEDS où l'on peut contrôler, la luminosité, la couleur (2 degrés de rotation pour les lyres, zéro pour les leds)… Elles sont pilotées en MIDI.

1. Graphique synthétique des différentes fonctions réalisées:

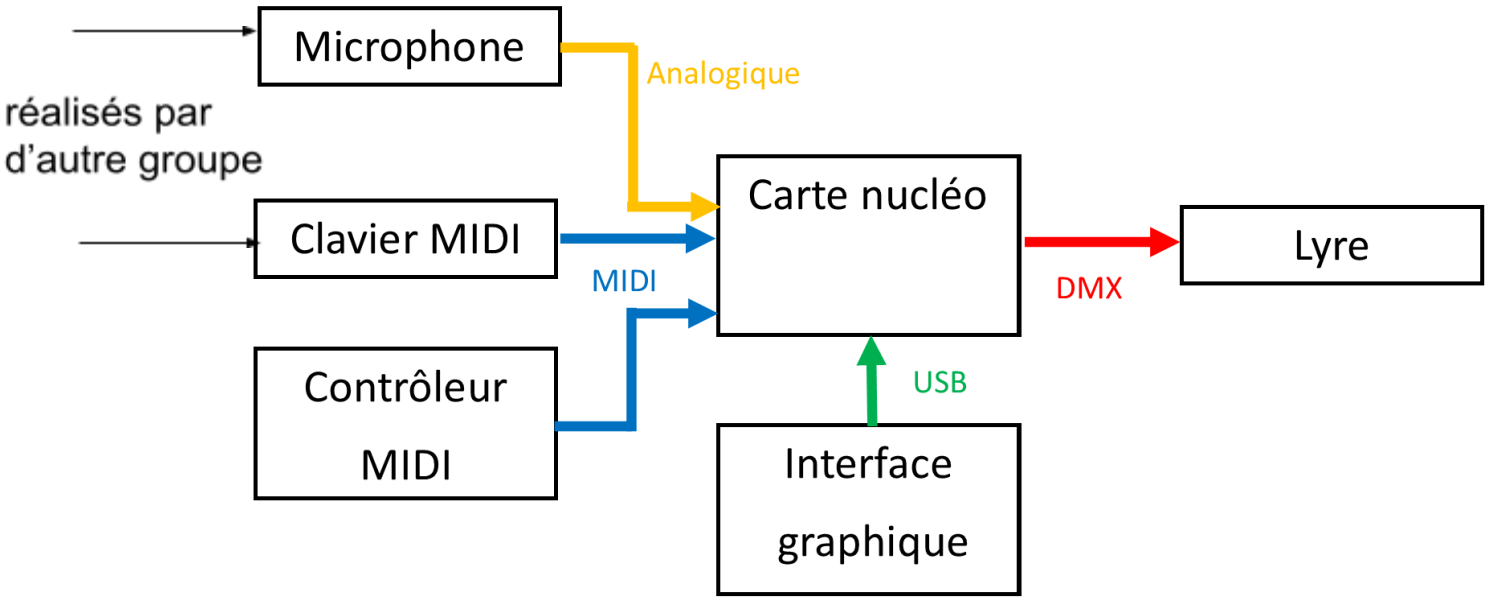

Figure 1. Schéma d'un nouveau contrôle des lumières

2. Descriptif des fonctionnalités :

Pour pouvoir brancher simplement l'entrée MIDI et la sortie DMX sans avoir à s'encombrer de branchements compliqués, notre carte Nucleo sera montée par un multiplexeur. Ainsi le LW n'aura qu'à brancher ses lyres à la sortie DMX et son PAD à l'entrée MIDI prévues à cet effet.

La fonctionnalité principale de notre système est de prendre le signal envoyé en MIDI et de le traduire en commande DMX lisible par les lampes. Les applications de cette fonctionnalité peuvent permettre de contrôler la fréquence des flashs colorés pour se synchroniser sur des musiques, le choix de la couleur en temps réel, où même de lancer des séquences automatiques; et ce, avec plusieurs éléments différents du pad.

Nous avons également mis en place une interface graphique utilisable depuis un ordinateur. Grâce à la Nucleo, il n'y a plus de longue installation compliquée. Il est possible d'utiliser directement l'interface graphique codée sur matlab permettant de choisir quelle couleur appliquée à quelle LED, Il suffira simplement de rentrer le modèle de la LEDs et son adresse. Il sera ainsi possible de lancer des séquences prédéfinies ou de piloter d'autres types de LEDs à partir d'un catalogue interne.

Ainsi, quelles que soient les habitudes des lighters, ils pourront se familiariser facilement avec le système mis en place et profiter aux maximum des possibilités offertes par les lampes, sans avoir à se familiariser avec l'installation des jours à l'avance.

Réalisation du prototype.

1. Partie électronique

Notre prototype est donc simplement constitué d'une carte Nucleo sur laquelle nous avons monté et câblé des ports pour pouvoir les brancher une entrée MIDI et une sortie DMX.

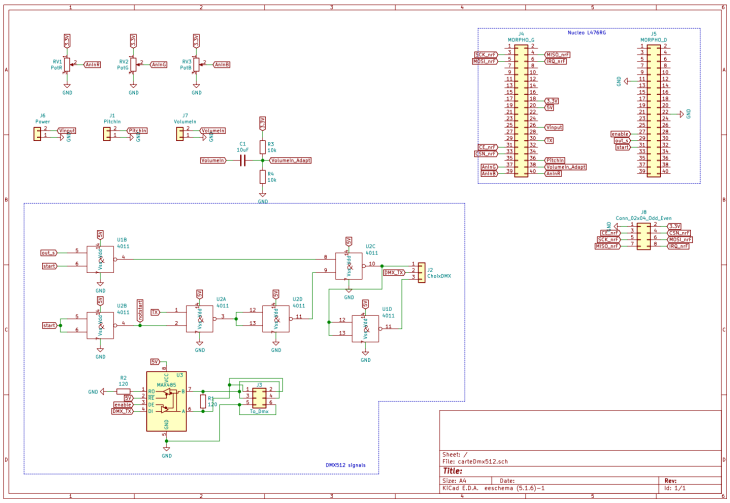

Figure 2. schéma électrique carte DMX

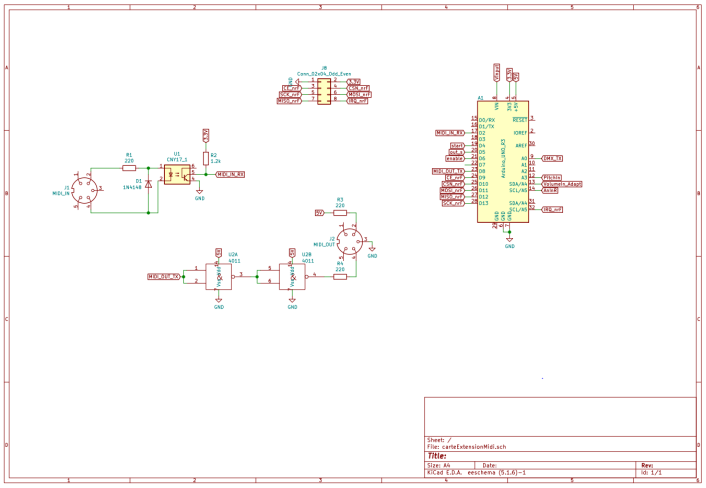

Figure 2 bis. schéma carte nucléo MIDI-DMX

2. Partie algorithmique.

C'est là la plus grosse partie de notre travail :

-Code Mbed

Le code sous Mbed permet en général de faire la conversion des signaux MIDI (qui envoie des bus de bit) en DMX. Il faut faire attention à bien gérer les indices des chaînes de caractères qui constituent les signaux reçus (liste de liste) et les signaux envoyés (liste simple de 512 bits où chaque adresse correspond à une voie d'une Led). Les fonctionnalités mises en place pour illustrer les possibilités de nos conversion midi dmx sont expliquées dans une version très commentée de notre code (C.F. annexe).

Un autre code a été fait afin de recevoir les messages envoyés par l'interface graphique et les transcrire en contrôle DMX. La partie d'actualisation des commandes DMX fonctionne exactement pareil que pour la conversion MIDI-DMX

-Interface graphique Matlab

A l'aide de la fonction MATLAB App, nous avons décidé de coder une interface graphique permettant de contrôler les LEDs à distance via un ordinateur. Ce programme permet :

- d'établir la communication ordinateur-carte Nucléo via la liaison USB
- de définir et supprimer les LEDS (nom, adresse, type)
- de paramétrer les LEDs (contrôler l'intensité lumineuse, les couleurs des LEDs, vitesse d'affichage, rotation des lyres…)
- BONUS: de programmer et de lancer des séquences ne nécessitant pas d'intervention une fois lancées.

Notre travail s'est divisé en plusieurs sous parties :

- Le design de l'interface graphique: comment organiser les fenêtres graphiques, quelles fonctions nous avions besoin, comment optimiser l'utilisation de l'interface pour un utilisateur lambda
- Ensuite la partie la plus longue et fastidieuse était de coder les boutons, c'est-à-dire

coder les fonctions derrière les fenêtres graphiques pour que les boutons répondent aux commandes qu'on renseigne .

- Enfin nous avons ajouté et codé une fenêtre graphique pour établir la communication avec la carte Nucleo

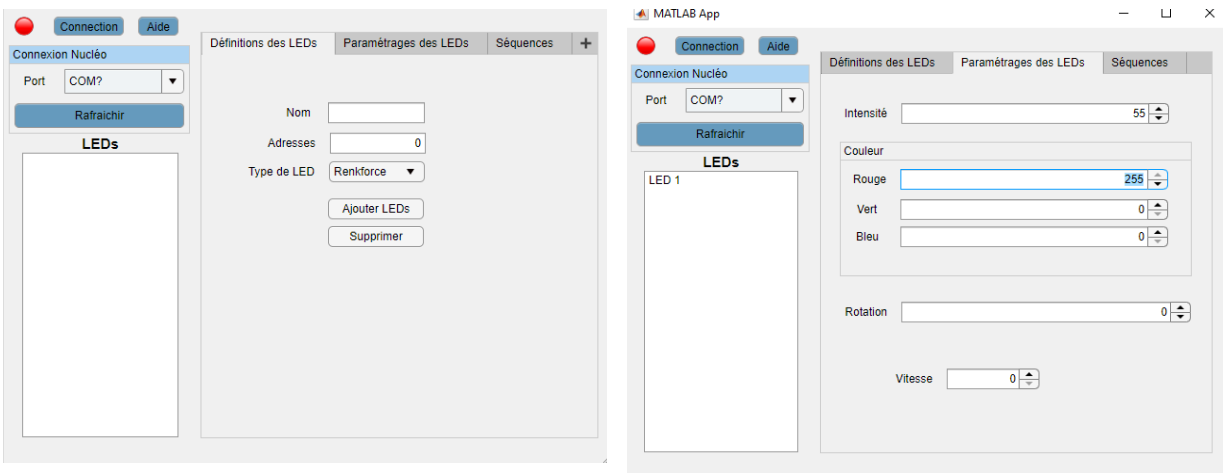

Figure 3. Interface graphique.

En haut à gauche, on trouve les boutons servant à la connexion avec la Nucléo. En dessous, on trouve la liste des LEDs définies. Celles-ci le sont dans l'onglet "définitions des LEDs" et commandées dans l'onglet "Paramétrages des LEDs".

Principe et explication des différentes fonctions:

Vous pouvez retrouver le code de l'interface dans le fichier suivant:
https://drive.google.com/file/d/1EwLykHkAeU2K3pMk5tnqZvhtsOaPtt2p/view?usp=sharing

Définitions des LEDS:

- AddLED: Nous avons codé cette fonction afin d'ajouter une nouvelle LED à la liste des LED crée; elle se déclenche quand on appuie sur le bouton AjouterLEDs. Le nom est stocké dans la liste app.LEDsList.Items, ce qui correspond à une propriété de la ListBox LEDs (à gauche). L'adresse et le type sont stockés dans deux listes (app.adresseLED, app.typeLED). La clé primaire pour les données est donc l'indice dans chaque liste. La fin de la fonction remet à 0 les différents champs de l'onglet.
- supprimerLED: Nous avons codé cette fonction afin de supprimer les LEDs sélectionnées dans la ListBox LEDs , si l'utilisateur a fait un erreur dans sa définition. Elle se déclenche quand on appuie sur le bouton Supprimer.

Paramètres des LEDS:

- Lors de la modification d'une des valeurs de couleurs, d'intensité ou d'angle, une fonction permet de récupérer la valeur à envoyer, déterminer le canal sur lequel l'envoyer, mettre en forme et envoyer le message.
    a. Les indices des LEDs sélectionnées sont récupérés (ismember). On balaye ensuite LED par LED.
    b. A l'aide de l'adresse et du type, on calcule le port sur lequel envoyer l'information grâce à la fonction findcanal.
    c. On envoie le message: une chaîne de 7 caractères au format ASCII 'a001002'. Le 'a' permet de marquer le début d'une séquence. Les trois chiffres suivants codent le canal sur lequel doit être imposée la valeur donnée par les trois derniers chiffres. La fonction writeline permet d'envoyer le message via le port USB.
    d. Remarque: pour certaines LEDs, il est nécessaire d'imposer le mode de commande en 'manuel' sur le canal 1. C'est ce qui est fait dans le 'if'.

- findcanal: Le numéro du canal correspondant à une commande (niveau de rouge par exemple), varie selon le type de LED. La fonction findcanal renvoie donc le numéro de canal correspondant à la commande souhaitée (CC est un caractère codant la commande selon le tableau suivant) en fonction du type de Led (TypedeLED est une chaîne de caractère). Pour rajouter un nouveau type de LED, il suffit de copier-coller un type existant, de modifier le nom et de noter les canaux correspondants à chaque commande (cf doc technique de la LED). **Attention: tous les noms doivent faire le même nombre de caractères (11 ici)**. Il faut ensuite rajouter ce nom dans les valeurs possibles de app.TypedeLED.

Commande	CC
Niveau de Rouge	R
Niveau de Bleu	B
Niveau de Vert	V
Niveau de Blanc	W
Intensité	I
Tilt (rotation autour de l'axe horizontale)	T
Pan (rotation autour de l'axe verticale)	P
Vitesse de rotation	S

Figure 4. tableau de traduction commande <-> caractère codant

Connexion de l'interface à la carte Nucléo:

- connectionpush: Nous avons codé cette fonction afin de connecter la carte Nucléo à l'interface graphique à l'aide du port sélectionné dans ListBox. La fonction matlab configureTerminator permet cela, le reste étant globalement esthétique.

- rafraichirpush: permet de mettre à jour la liste des ports disponibles.

Validation du système final

1. test du MPD 26

a. test des pads

Les pads sont le contrôle le plus basique du MPD 26. Lorsque l'on appuie sur un des 16 pads disponibles, l'information MIDI envoyée contient la vitesse à l'impact du premier appuie, la pression que l'on met ensuite (sustain) et le numéro du pad (la note produite)

On teste séparément ces trois contrôles. On distingue donc dans notre programme 3 actions différentes : la pression correspond à l'intensité de la lumière, la vitesse permet de choisir le set de couleur parmi lesquelles choisir et la note la couleur choisie.

b. test des contrôleurs

Le contrôleur consiste en un bouton à tourner où à translater: il renvoie une note (qui correspond à quel contrôleur on tourne) et une valeur de contrôle.

On teste séparément en faisant en sorte qu'un contrôleur fasse varier la vitesse de clignotement des Leds et un autre fasse varier les couleurs.

Dans notre test, on a utilisé 4 contrôleurs en translation pour gérer respectivement l'intensité moyenne, et les poids de vert, de rouge et de bleu (la modélisation est assez visuelle donc instinctive pour les lighters).

En parallèle un contrôleur peut être utilisé pour faire varier les lumières aléatoirement au rythme d'une musique. C'est donc la fréquence du changement de couleur qu'on contrôle (C.F. code en annexe).

2. Test de l'interface graphique

Le test de l'interface a été assez rapide. Nous avons adressé 4 LED de marque Renkforce à 4 adresses différentes puis nous avons rajouté une lyre Leuchtkraft. Nous avons ensuite essayé d'envoyer différentes commandes en sélectionnant une ou plusieurs LED. Le système exécute bien les commandes mais le temps d'exécution peut sembler un peu long parfois mais n'a pas été chronométré. La lyre se bloque parfois pendant quelques secondes, sans répondre aux commandes, puis après un instant sans nouvelle commande elle remarche. La raison de ces blocages n'est pas connue.

Vous pouvez retrouver une vidéo de démonstration au lien suivant:
https://drive.google.com/file/d/1O1B2TMjoIeyfc8YURGbzhvxH-WnZWm1E/view?usp=sharing

Gestion d’équipe du projet

1. Tenue des deadlines

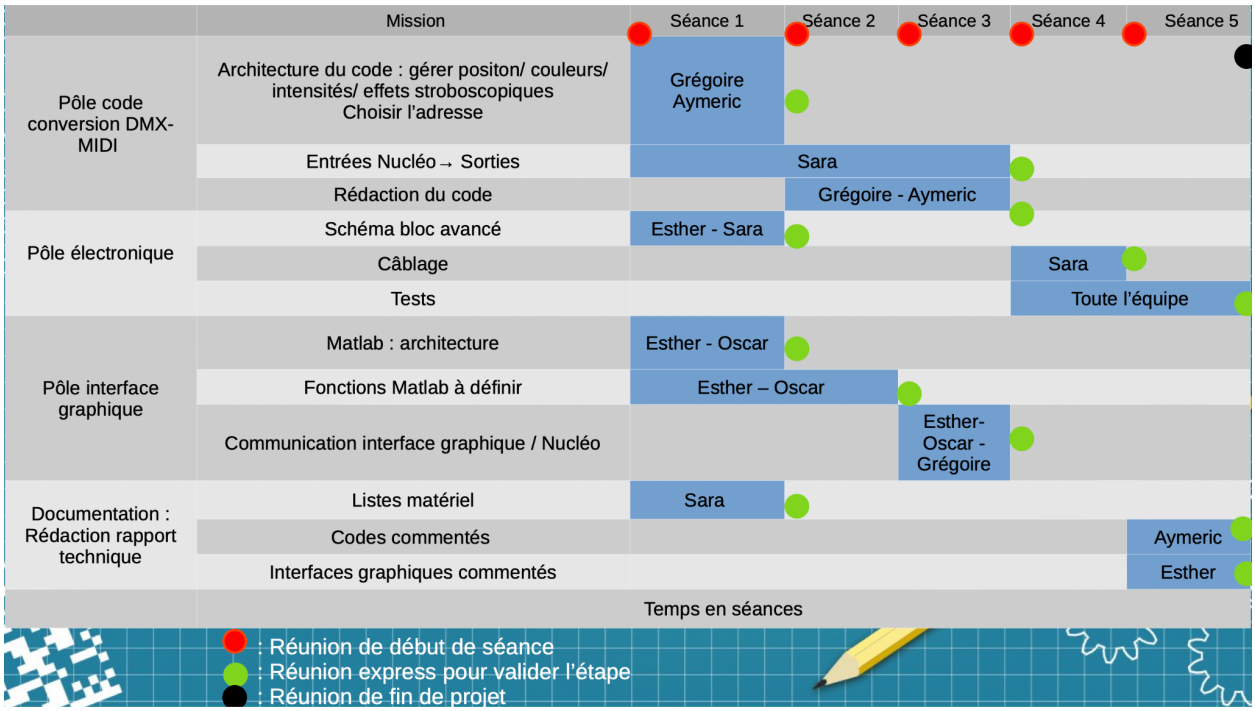

Figure 5. planning et rétroplanning de notre projet.

Avant de se lancer dans notre projet, nous avons réalisé un planning prévisionnel (ci-dessus). Par ailleurs, nous avons tenu un cahier de bord du travail réalisé avant chaque séance. Il s’avère que le planning a été tenu, ce dont nous pouvons nous féliciter. Cependant, le départ des CFA (3 membres sur 5) n'a pas été bien anticipé provoquant une désorganisation pour la rédaction des livrables. Nous n’avons aussi pas eu le temps de réaliser la partie ‘bonus’ de l’interface graphique consistant à préprogrammer des séquences.

La gestion des tâches en outre à été bien réalisés car chaque personne avait un travail qui correspondait à ses envies et à ses compétences, ainsi le projet a pu avancer le plus efficacement possible. (Cela a nécessité de fréquentes réunions surtout en début de projet).

difficultés rencontrées

1) Les difficultés que nous avons rencontré étaient d'abord matérielles :

-En effet les branchements d'une entrée midi pour une sortie en DMX sur une carte nucléo n'est pas une mince affaire, mais SOLEC nous a gentiment pourvu d'un multiplexeur adapté de telle sorte que nous n'avions à nous concentrer que sur la partie software du projet.

2) Ensuite on a eu des difficultés sur le code MBED :

-Le format des données envoyées en MIDI est différent de celui utilisé en DMX. Ainsi notre première grosse difficulté a été la conversion de l'un vers l'autre. Chaque séquence MIDI devait être récupérée et stockée, chaque bit devait piloter le bon canal DMX et il ne fallait surtout pas se décaler dans les listes. Un autre problème était le fait que chaque type de lampe se pilote différemment, par exemple les LEDS du banc de LEDS envoie sur les trois premiers canaux les signaux rouge puis vert puis bleu alors que sur la lyre se sont les trois derniers dans un autre ordre. Ainsi il faut faire attention à comment évoluent les indices des listes dans lesquels nous avions stocké les bus d'informations envoyés en MIDI pour bien les diriger dans les canaux DMX. Ensuite chaque contrôle midi n'envoie pas qu'une seule information, ainsi au sein même d'une commande midi il faut faire attention à l'indice à prélever en fonction du contrôle souhaité (exemple pour le pad: pression, vitesse ou note).

-Il fallait aussi faire attention aux types de variables utilisées, contre instinctivement il fallait traiter les bus de bits comme des séquences de caractères (char) sinon Mbed n'arrivait pas à réaliser les commandes demandées.

-Ces difficultés nous ont posés quelques soucis de diagnostic au débogage, mais rien d'insurmontable.

3) Enfin l'interface graphique et sa communication avec la carte nucléo a eu son lot de problème:

-Le problème principal au niveau de l'interface graphique a été la communication avec la carte nucléo. L'envoie et la lecture d'un caractère a tout de suite fonctionné (sous réserve de la

conversion depuis le format ASCII à la réception) mais pas l'envoie de chaine de caractère. Nous avons donc opté pour une lecture caractère par caractère d'un message de longueur fixé. Le numéro de canal et la valeur à envoyer sont ensuite convertie en entier d'après la décomposition en base 10.

-L'autre problème a été de maintenir allumé les LEDs, ce qui nécessite d'actualiser la commande régulièrement, ce qui a été fait via une commande d'interruption.

2. Compétences acquises

Les compétences que nous avons pu développer au cours de ce projet ont été nombreuses.

Compétences transverses :

La première étape de ce projet a été de traduire les attentes du client en contraintes concrètes (partie de dialogue). Ensuite nous avons dû extraire de ces contraintes un cahier des charges technique (synthétisation) sur lequel nous pourrions nous baser lorsque nous passions au projet concret.

Toujours sur les soft skills, nous avons dû gérer notre équipe du mieux possible: chaque personne ayant un avis différent il fallait que chacun se sente à l'aise pour s'exprimer librement. Et toujours faire attention à ce que la parole circule bien. Pour ce faire nous réalisons fréquemment des réunions tous ensemble. Ensuite, n'ayant pas tous les mêmes formations et capacités, nous avons adapté le travail demandé à chacun. Par exemple Esther avait déjà travaillé sur des interfaces graphiques, il était plus logique de lui confier cette tâche là où Grégoire était plus rodé en C++ par exemple..;

Compétences techniques :

La première partie du travail technique a été de réaliser un prototype qui répondait positivement au cahier des charges que nous avions établi. Le prototypage nous propulse directement dans le concret : les idées abstraites deviennent des réalités et on se rend compte de ce qui est réaliste, ce qui est plus simple que prévu etc …

Quel que soit notre rôle dans l'équipe nous avons tous eu à nous confronter à une partie software : Soit via MBed soit via matlab. La partie hardware était quasi uniquement liée au multiplexeur et ne nous a pas posé trop de soucis. En tout cas, étant plusieurs à travailler au codage et au débogage des mêmes programmes, il fallait avoir une bonne communication entre nous et apprendre à rédiger des codes clairs avec des variables qui ne prêtent pas à confusion.