# Projet ProTIS
## SupopLED

**Résumé**

Les systèmes à LED deviennent omniprésents dans les éclairages industriels et particuliers. Notre projet “Smart LED” que nous avons renommé “SupopLED” est une solution d’éclairage intelligent qui permet d’asservir l’intensité des LEDs en fonction des cas d’utilisations et de la luminosité ambiante. Le choix du mode, de la couleur et de la largeur du faisceau par exemple pourront être contrôlés par l’utilisateur, tandis la régulation de l’intensité lumineuse est automatique.

# 1 Introduction

## 1.1 Nos objectifs

Les objectifs que nous avons atteints sont l’adaptation de l’intensité lumineuse de notre système d’éclairage en fonction de l’éclairage ambiant, le choix de la couleur et de la largeur du faisceau, et enfin du nombre de LEDs allumées ou éteintes sur le bandeau. L’objectif que nous avons partiellement complété est l’utilisation d’une interface mobile pour pouvoir changer de mode d’utilisation.

## 1.2 Notice d’utilisation

### Matériel requis

- Carte Nucléo
- Câbles et fils
- Boîtier d’alimentation
- Photodiode
- Résistance

Pour pouvoir utiliser le bandeau de LEDs, il est nécessaire de brancher les 3 fils en sortie du bandeau avec la carte Nucléo suivant les branchements suivants :

- Fil rouge : alimentation 5V (borne +)
- Fil blanc (doublé d’un fil noir) : port D7 de la carte Nucléo
- Fil noir : alimentation 5V (borne -) relié au port GND de la carte Nucléo

puis de téléverser le code depuis l’interface MBED vers la carte Nucléo. Il faut ensuite brancher la photodiode sur cette même carte Nucléo. Pour ce faire, il suffit de brancher la sortie de la photodiode sur le port d’entrée analogue A0 et de relier la terre de la photodiode à celle de la Nucléo.

## 1.3 Cahier des charges

Notre système d'éclairage doit répondre au cahier des charges suivant :

- Tenir quelques heures en autonomie
- Atteindre un flux lumineux d'au moins 500 lm
- Utilisation possible de plusieurs LEDs pour diversifier les modes d'éclairage (large ou focalisé)
- Proposer plusieurs modes d'éclairage
- Si possible, programmer à distance ces derniers modes d'éclairage

Nous avons mesuré à l'aide d'un luminancemètre un flux lumineux allant jusqu'à 563.3 lm : nous avons donc satisfait cette performance. De plus, nous avons utilisé un bandeau de LEDs pour pouvoir varier les modes d'éclairage. L'utilisateur possède ainsi le choix entre différents modes préconçus : "Classique" (tout le bandeau, violet), "Focus" (faisceau focalisé, blanc), "Réveil" (1 led sur 2, orange), "Soir" (1 led sur 2, bleu ciel), "Ambiance" (tout le bandeau, rouge), et le mode manuel qui lui permet de choisir ses propres paramètres tels que la couleur du bandeau, la largeur du faisceau et le nombre de LEDs allumées (toutes les LEDs ou bien une LED sur deux). Nous n'avons malheureusement pas eu le temps de prendre en compte la contrainte d'autonomie ; notre bandeau de LED doit être constamment alimenté. Enfin, nous n'avons pas fini d'établir la programmation des différents modes à distance : nous avons opté pour une application mobile créée à partir de MIT App Inventor, connectée au bandeau de LEDs par un module Bluetooth de type RN-42. Cependant, nous n'avons pas eu le temps d'établir l'ensemble des liaisons entre notre application et le module.

## 1.4 Schéma fonctionnel

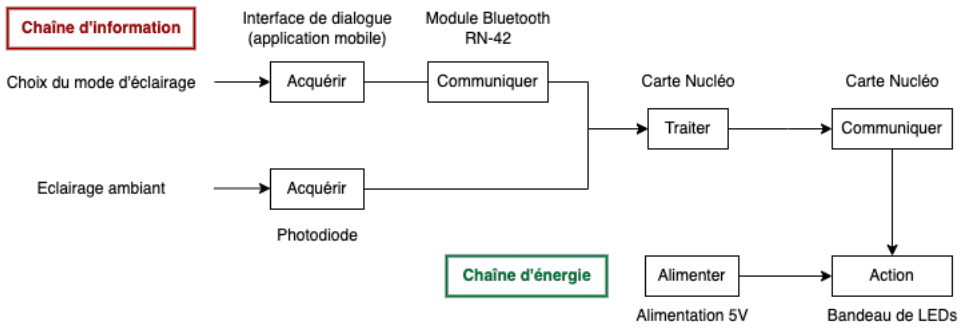


FIGURE 1 – Schéma fonctionnel

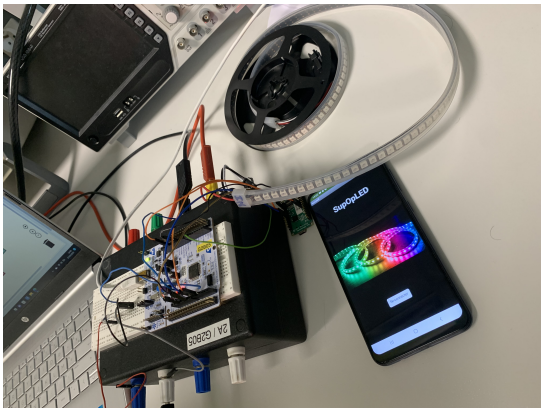

FIGURE 2 – Montage expérimental

# 2 Description du projet

## 2.1 Détail des fonctions

Afin de réaliser notre projet, nous avons défini et utilisé 6 fonctions sur MBED :

```
void toggle();
double moyenne_tension_phd();
int calcule_intensite_led();
void change_intensite_couleur_et_largeur_led(int intensite, int couleur,
int largeur, int mode);
void ISR_get_data_pc(void);
void ISR_get_data_device(void);
```

**Moyenne_tension_phd**

La fonction moyenne_tension_phd permet de calculer la valeur moyenne de la tension mesurée aux bornes de la résistance.

Notre photodiode est alimentée par une tension de 3.3V (tension maximale pour la Nucléo) et reliée à une résistance de protection de $100k\Omega$. La photodiode que nous avons utilisée est sensible à de très faibles variations d'éclairement telles que celles des lampes de la salle de TP qui clignotent à une fréquence invisible à l'œil nu. En effet, le signal reçu par la photodiode varie sinusoïdalement avec le temps, d'après nos observations à l'oscilloscope numérique. Nous avons donc cherché à calculer la valeur moyenne de la tension mesurée aux bornes de la résistance afin d'éliminer ces variations parasites et obtenir une valeur proportionnelle à l'éclairage ambiant. Pour cela, nous avons établi une moyenne sur 100 mesures. De plus, nous voulions augmenter la sensibilité de notre photodiode égale à 0.5 A/W (données constructeur) afin de mieux détecter les variations de l'éclairage ambiant. Pour cela, nous avons essayé d'utiliser plusieurs photodiodes en série et en parallèle, mais le résultat n'était pas convaincant. Avec un tel montage, la tension mesurée varie sinusoïdalement entre la valeur 0 et sa valeur maximale alors que l'éclairement reste inchangé. Nous avons donc finalement décidé de nous contenter d'une seule photodiode.

Pour tester cette fonction, nous avons ajouté une ligne de code permettant d'afficher nos résultats en temps réel sur TeraTerm. Nous pouvons ainsi vérifier nos calculs et observer leur variation avec l'éclairage ambiant. C'est notamment grâce à ce test intermédiaire que nous avons conclu qu'il fallait établir une moyenne sur 100 mesures au moins.

**Calcule_intensite_led**

La fonction calcule_intensite_led renvoie l'intensité (sous forme de rapport cyclique nommé rc) à appliquer à chaque LED du bandeau en fonction de l'intensité reçue par la photodiode, et donc du résultat de la fonction Moyenne_tension_phd.

Pour cela, nous avons arbitrairement (suite à une série de tests) associé une valeur de rc à chacune des 16 plages de tension mesurée aux bornes de la résistance afin d'obtenir un changement d'intensité progressif du bandeau. Nous avons déterminé les différentes plages de tension en relevant sur TeraTerm la tension mesurée d'un éclairage nul (noir complet) à un éclairage maximal (éblouissement de la photodiode par une lampe).

Pour vérifier le bon fonctionnement de cette fonction, nous avons ajouté une ligne de code permettant d'afficher le rapport cyclique rc sur TeraTerm. En combinant ceci au résultat de la fonction précédente, nous pouvons voir si la valeur de rc correspond bien aux seuils définis en fonction de l'éclairage ambiant.

**Toggle**

La fonction Toggle est une fonction d'interruption permettant d'appeler chaque seconde la fonction calcule_intensite_led. Elle est utilisée dans le Main de manière à ce que le calcul calcule_intensite_led soit effectué chaque seconde et donc que l'intensité du bandeau s'adapte au même rythme.

Pour cela, nous avons défini en début de code un Ticker nommé toggle_ticker.

Pour vérifier le bon fonctionnement de cette fonction, nous avons affiché sur TeraTerm la tension mesurée aux bornes de la résistance grâce à la fonction Moyenne_tension_phd qui est appelée dans calcule_intensite_led, ainsi que le rapport cyclique rc renvoyant l'intensité à appliquer au bandeau de LEDs calculée par cette dernière fonction. À cette étape, lorsque nous compilons notre code, nous pouvons lire sur TeraTerm les valeurs de tension mesurée et d'intensité calculée en temps réel (chaque seconde).

### change_intensite_couleur_et_largeur_led

Cette fonction transmet au bandeau de LEDs les informations suivantes : son intensité lumineuse, sa couleur, la largeur du faisceau (nombre de LEDs allumées pour un faisceau centré) ainsi que le mode d'éclairage (toutes les LEDs allumées ou une LED sur deux). Nous avons défini ces variables globales au début de notre code et les avons nommées color, l et m respectivement.

La fonction est appelée dans le While(1). Nous avons fait ce choix, car nous nous sommes rendues compte après une série de tests que notre code ne fonctionnait pas lorsque nous mettions cette instruction en dehors de la boucle While. Cette fonction comporte une boucle If permettant de définir la largeur du faisceau (traduit en nombre de LEDs allumées) en fonction des paramètres en entrée. Afin de transmettre ces informations au bandeau, nous utilisons la commande .write.

Afin de tester cette fonction, nous avons entré manuellement différents paramètres puis compilé le code ; ces derniers étaient bien retransmis par notre système (**Figure 3**). Nous avons par la suite fixé des valeurs d'initialisation du bandeau qui nous permettent de vérifier son bon fonctionnement lors de chaque utilisation.

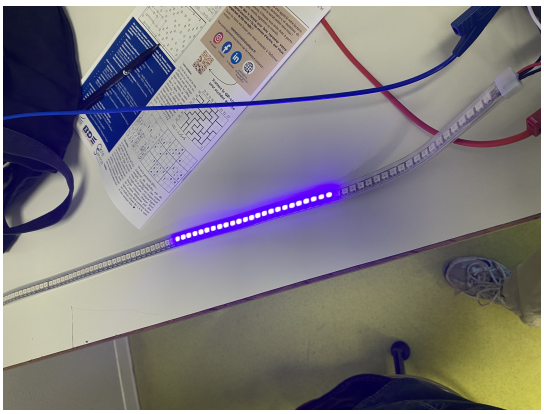

FIGURE 3 – Utilisation manuelle du bandeau de LEDs

### ISR_get_data_pc

La fonction ISR_get_data_pc est une fonction d'initialisation permettant de connecter le module Bluetooth au code MBED.

On initialise la liaison série avec le port USB grâce à UnbufferedSerial que l'on nomme dans notre code debug_pc.

### ISR_get_data_device

La fonction ISR_get_data_device permet de recevoir les commandes indiquées par l'utilisateur et

transmises par le module Bluetooth sous la forme d'un unique caractère (Exemple : 'a' pour le Case 'a'). Chaque caractère (Case) correspond à l'un des modes d'éclairage prédéfinis ou bien à une couleur, largeur de faisceau ou option lorsque le mode manuel est activé. Cette fonction modifie, en fonction du caractère reçu, les paramètres (color, l et m) à appliquer au bandeau de LED en entrée de la fonction change_intensite_couleur_et_largeur_led.

On initialise la liaison série avec la carte Nucléo grâce à UnbufferedSerial que l'on nomme device_rs.

Pour la réception des données par le module Bluetooth, nous avons également utilisé l'affichage sur TeraTerm. Ainsi, lorsque l'application était connectée au module, TeraTerm renvoyait le message "CONNECTED". Pour vérifier que la commande indiquée par l'utilisateur était également transmise par le module, nous avons affiché sur TeraTerm le caractère correspondant, ce qui s'est avéré concluant. Cependant, le bandeau de LEDs n'était pas modifié. Ceci est dû au fait que la fonction ayant pour paramètres color, l et m est une fonction void. Ces trois valeurs étant des variables globales, nous aurions pu les afficher sur TeraTerm pour vérifier leur modification à l'envoi d'une commande (mais ne l'avons pas fait par manque de temps).

#### Main

Dans le Main, nous commençons par initialiser la liaison série USB et Nucléo de notre module Bluetooth. Tant que l'application n'est pas connectée au Bluetooth, il s'affiche le message "NOT YET" sur TeraTerm.

Lorsque la connexion est établie, on initialise la matrice du bandeau de LEDs à une intensité nulle. L'objet "bandeau" défini en début de code permet de piloter notre bandeau de NOMBRE_LED = 144 LEDs (défini également en début de code) sur la broche D9 de la carte Nucléo. Le bandeau est défini de manière à respecter les contraintes temporelles imposées lors de la fabrication de la carte Nucléo.

On appelle ensuite la fonction d'interruption. La fonction change_intensite_couleur_et_largeur_led est appelée dans une boucle While (validation par l'équipe de Solec) car elle ne fonctionne pas en dehors de celle-ci.

### 2.2 Application mobile

Nous avons créé notre application sur MIT App Inventor afin d'obtenir une interface entre l'utilisateur et le bandeau de LEDs. Cette application permet de se connecter au module Bluetooth afin d'appliquer à notre système l'un des modes prédéfinis dans le cahier des charges ou bien une caractéristique spécifique (color, l, m).

#### Création de l'application

Le site pour créer l'application est structuré en deux catégories, l'une pour designer l'application, l'autre pour créer des schémas blocs.

Nous avons commencé par designer notre application. La première page est une page d'accueil qui permet de se connecter au module Bluetooth. La seconde page permet à l'utilisateur de sélectionner le mode automatique (modes prédéfinis) ou le mode manuel (choix manuel des différents paramètres). La troisième page permet de sélectionner ces modes et paramètres.

Nous avons ensuite utilisé la fonctionnalité bloc du site. Pour chaque page que nous avons designé, il existe une interface bloc. Par exemple, nous avons créé pour la première page de notre application un bloc permettant d'afficher sur le mobile de l'utilisateur le message "Connecté" lorsque la connexion Bluetooth est établie. Ce bloc permet également d'accéder à la page suivante une fois la connexion effectuée. Nous avons ajouté entre chaque page une option permettant de revenir à la page précédente.

#### Utilisation de l'application

Pour pouvoir utiliser notre application, il faut d'abord télécharger l'application Android associée au site MIT App Inventor. Il faut aussi que le téléphone de l'utilisateur et l'ordinateur de commande soient connectés au même réseau Wi-Fi. Avant chaque utilisation, il faut scanner un QR-code généré sur le site depuis l'application Android.

Nous avons ainsi réussi à établir la connexion avec le Wi-Fi ainsi qu'à envoyer un caractère via le module Bluetooth depuis la première page de l'application. Nous avons cependant rencontré des problèmes liés à la déconnexion soudaine du module lorsque nous changions de page.

# 3 Bilan

## 3.1 Avancement final

Finalement, nous avons réussi à réaliser les objectifs suivants :

- Adapter l'intensité du bandeau de LED à l'intensité lumineuse ambiante
- Créer différents modes d'éclairage ainsi qu'un mode de réglage manuel
- Contrôler le bandeau manuellement via le code
- Créer une application permettant de contrôler ces fonctionnalités avec MIT App Inventor
- Connecter l'application au module Bluetooth

Cependant, nous n'avons pas réussi à :

- Transmettre les informations de l'application vers le module Bluetooth puis le code
- Contrôler le bandeau à distance via l'application

En effet, nous avons réussi à coder les différentes fonctionnalités du bandeau de LEDs qui sont modifiables manuellement sur MBED. Nous avons de plus réussi à développer une interface avec l'utilisateur, sa connexion au module Bluetooth ainsi que l'envoie de données récupérées par le code et lisibles sur TeraTerm.

Cependant, nous avons rencontré un problème de déconnexion du module Bluetooth lorsque nous changions de page sur l'application, ou bien au bout de quelques secondes. Ce problème vient probablement du code de l'application, mais nous n'avons pas eu le temps de comprendre exactement sa provenance ni de trouver des solutions. C'est pour cela que nous n'avons pas pu transmettre les informations depuis l'application vers le code pour ainsi contrôler le bandeau à distance.

Par ailleurs, nous aurions pu rajouter une étape intermédiaire de conception d'interface via TeraTerm nous permettant de contrôler le bandeau à distance et non directement du code avant de lancer notre application. Ceci aurait permis contrôle simplifié du bandeau de LEDs.

## 3.2 Retour d'expérience

Finalement, nous sommes contentes de notre projet, mais tout de même déçues de ne pas avoir pu finir totalement ce que nous souhaitions faire avec l'application Bluetooth.

Pour mener à bien notre projet, nous avons utilisé la plateforme Teams qui nous a permis de centraliser les informations et de prendre en notes nos progressions respectives, séance après séance. Nous avons commencé par définir les différentes tâches à réaliser que nous nous sommes réparties sur l'ensemble du planning des séances. Nous avons aussi décidé de travailler par binômes. Cet emploi du temps a été mis en page sur le diagramme de Gantt (**Figure 4**) que nous avons mis à disposition sur notre Teams. À l'issue de chaque séance, nous y indiquions l'avancement des différentes tâches que nous reprogrammions au besoin à la séance suivante. Le travail en binôme nous a permis de progresser

efficacement dans notre projet et de surmonter plus facilement les difficultés rencontrées. Par ailleurs, lorsque l'un des binômes rencontrait un problème majeur, il pouvait demander de l'aide à l'autre binôme afin de débloquer la situation. Enfin, nous nous réunissions à la fin de chaque séance pour mettre en commun nos avancées et les nouvelles connaissances acquises.

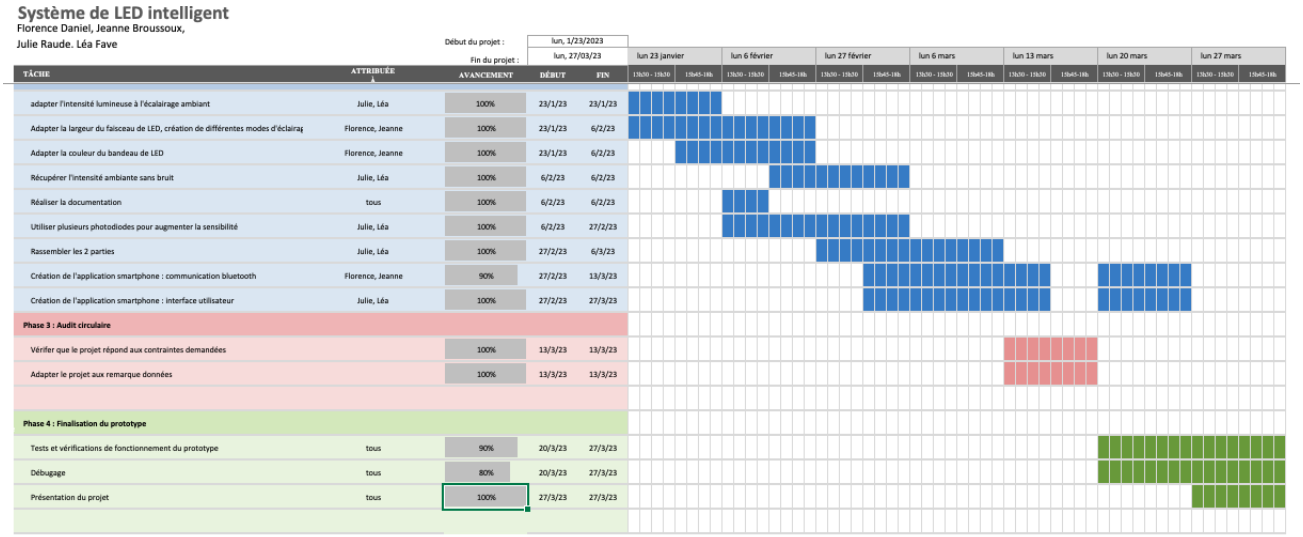


FIGURE 4 – Diagramme de Gantt