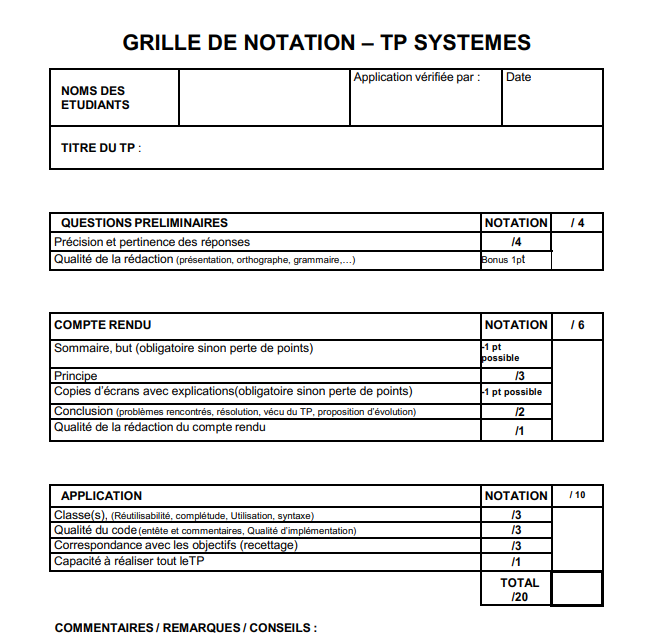
TP DMX

Sommaire

1. Grille d’évaluation (Page 3)
2. But du projet (Page 4)
3. Principe du projet (Page 5)
4. Question Préliminaire (Page 6 à 9)
5. Algorithme (Page 10)
6. Code (Page 11 à 12)
7. Problème rencontré (Page 13)
8. Recette (Page 14)
9. Conclusion (Page 15)
10. Lien github (Page 16)



But du Projet

Le but du projet est de gérer les couleurs et la luminosité des lampes SABER.

Principe du Projet

Le principe du projet est de commander un protocole utilisé sur un réseau de type BUS via une carte contrôleur DMX.

Question Préliminaire

Question 1 :

Les différents canaux que cette lampe propose sont :

Canal 3 qui permet de mélanger les couleurs HSI.

Canal 4 qui permet de mélanger RGBW

Canal 4 qui permet de mélanger les couleurs HSI+

Canal 5 permet de mélanger les couleurs RGBW+.

Canal 6 permet d’avoir fonction Stroboscope.

Canal 7

Canal 8 RGBW À GRADATION 16bit

Canal 9

Canal 11

Canal 12

Question 2 :

Si on possède 12 canaux de propriété, on peut piloter 8 canaux en même temps sur un bus DMX 512.

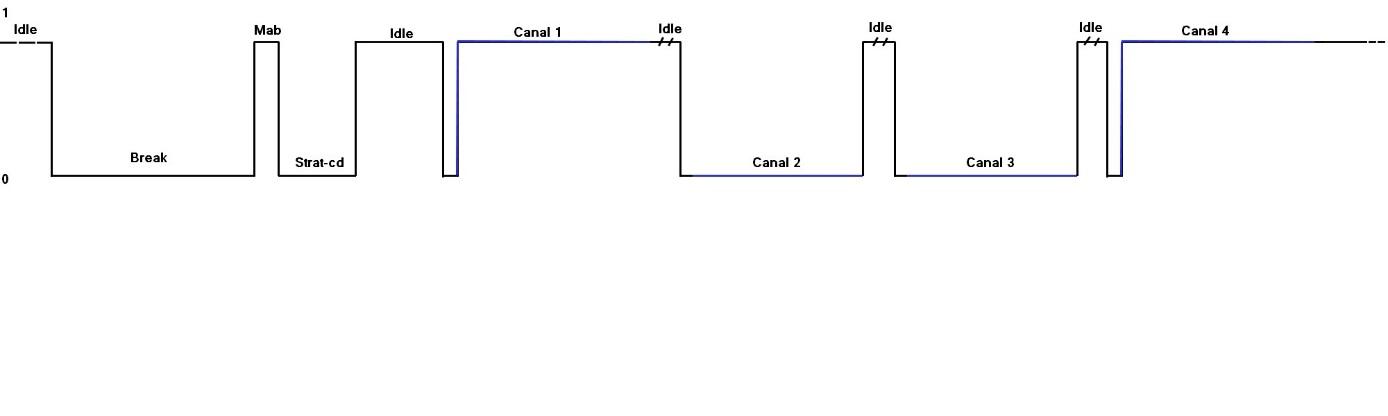
Question 3 :

Il est possible de mettre 6 canaux pour avoir un rouge qui clignote avec la SABER qui permet d’éteindre et d’allumer la lampe et elle peut clignoter.

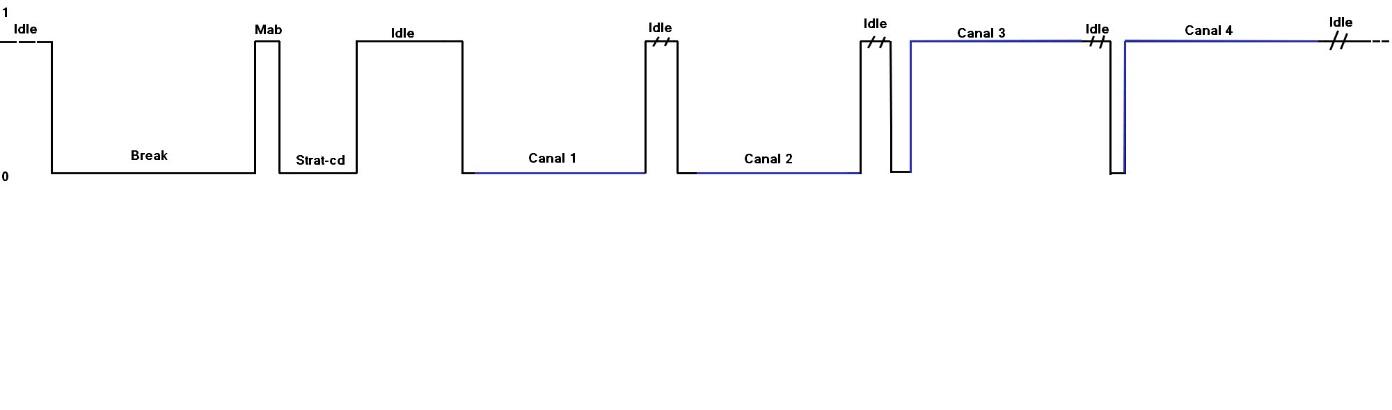
Question 4 :

Les trames à envoyer pour piloter les 3 lampes SABER :

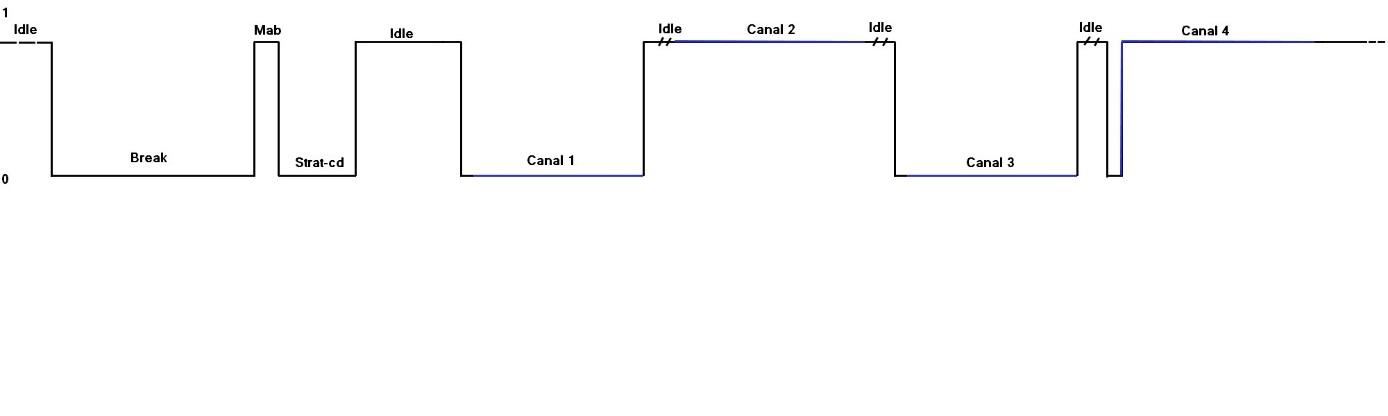
Trame rouge :



Trame bleu :



Trame verte :

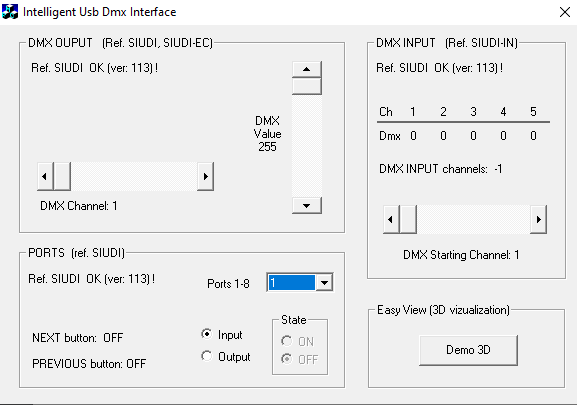


Question 5 :

On doit modifier comme information sur la lampe pour la différencier des autres est l’adresse.

Question 6 :

Pour avoir en rouge, on paramètre une lampe SABER sur un channel ici, le channel 1 puis on met la valeur DMX Value 255 pour avoir le rouge.



Question 7 :

Pour vérifier la librairie : (DASHARDCOMMAND)::GetProcAddress((HMODULE)g\_dasusbdll, "DasUsbCom mand");

Intialiser la DLL : =DasUsbCommand(DHC\_INIT,0, NULL);

Ouverture de la liaison avec l’interface USB : DasUsbCommand(DHC\_OPEN,0,0);

Question 8 :

La méthode C++ à utiliser pour envoyer une trame sur le bus DMX est DasUsbCommand.

Algorithme

Allume( entier valeur)

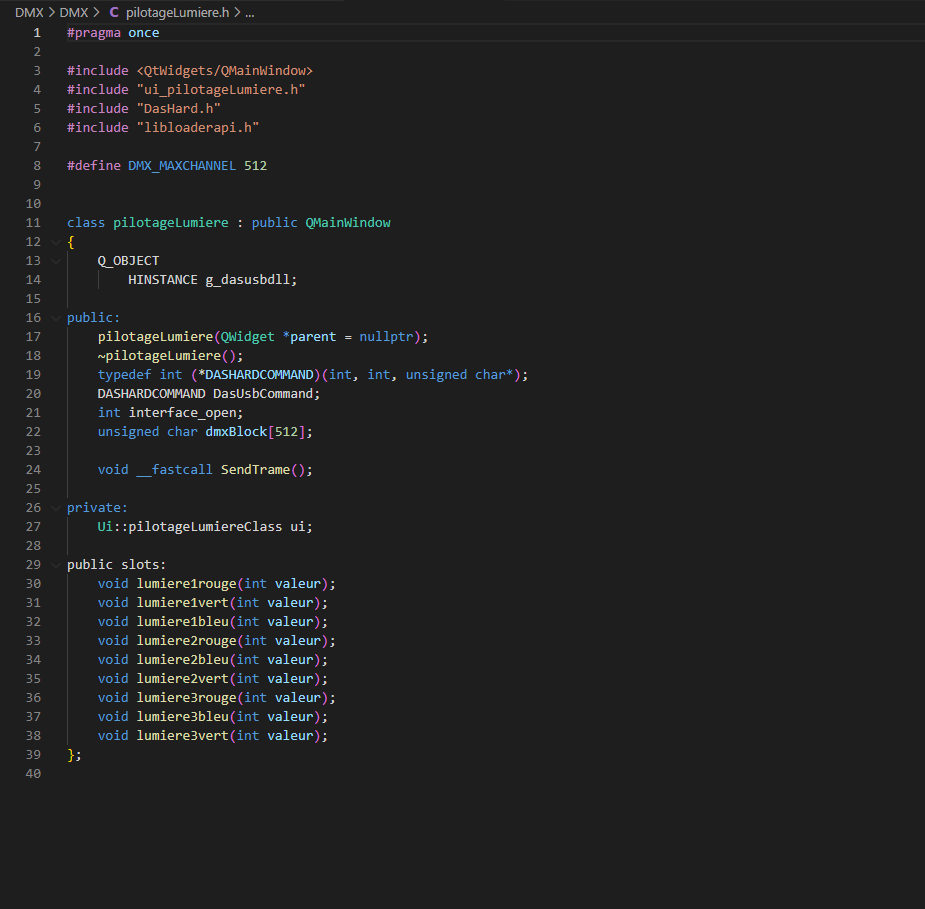
{

dmxBlock[valeur de l’adresse] prend la valeur

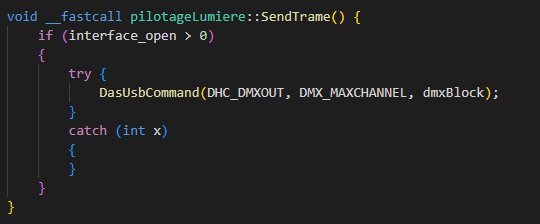
on appelle la méthode SendTrame()

}

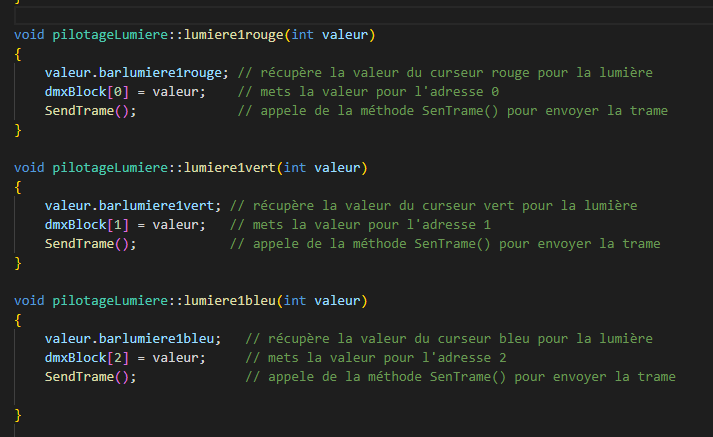
Code



Voici ci-dessus le code de la class “pilotageLumière” dans le partie public de la classe, les méthodes initialise la DLL qui va permettre de contrôler les lumières. Dans public slots les méthodes vont gérer les trois lumières avec pour chaque lumière trois couleurs différentes, bleu, rouge et vert.



Voici ci-dessus le code d’envoie de la trame, dès que l’interface sera supérieur à 0, la trame sera envoyée avec les valeurs des lumières.



Voici ci-dessus le code de contrôle des lumières, cela va récupérer la valeur du curseur de la lumière, cela va mettre cette valeur a l’adresse 0 et ensuite appeler la méthode d’envoie de la trame, ici nous ne vous avons mis un exemple pour une lumière, ce code se répète trois fois pour chaque lumières.

Problème rencontré

Nous avons rencontré un problème avec le fichier dll pour l'utiliser dans notre code ainsi que pour récupérer les informations du curseur et à le stocker dans une variable pour l’envoyer avec la trame pour afficher la lumière.

Recette

| ID | Test | OK/NOT OK |
| --- | --- | --- |
| 1 | allume les lampes |  |
| 2 | change les couleurs des lampes |  |
| 3 |  |  |
| 4 |  |  |
| 5 |  |  |

Conclusion

Pour conclure, durant ce tp nous avons été confrontés à des problèmes avec le fichier dll ainsi que dans l’interprétation des rames des bus. Ce projet a été particulièrement difficile et nous a permis de développer nos connaissances sur le sujet.

Lien github

Lien : https://github.com/charlesdesenclos/TP-Systeme