



B2 - Langage C++ - Microcontrôleur

TP Noté Arduino - Communicateur Morse

Par Julien SOSTHENE - 2021

Sommaire

1. Présentation du TP
2.  Etape 1 : Afficher un message en Morse
3.  Etape 2 : Interpréter et envoyer du Morse

1. Présentation du TP

1.1 But du TP:

Le but de ce TP est de pouvoir communiquer en code Morse entre le moniteur série et un montage Arduino.

On utilisera pour cela :

- La liaison série UART
- Les entrées numériques
- Les sorties numériques
- La programmation orientée objet en C++ et les bibliothèques Arduino communes.

Code Morse

💡 Pour rappel, le code Morse est un code utilisé dans l'aéronautique et l'aviation, mais qui a été inventé en 1832 **pour la télégraphie**.

Il associe aux lettres et aux chiffres des combinaisons de signaux longs et courts séparés par des interruptions de signal.

👉😊 On considère le code Morse comme l'ancêtre du numérique !

On alternera alors entre

- Un appui court ■
- Un appui long ≈ dont la longueur est environ 3 fois celle d'un appui court
- Une pause (aussi longue qu'un appui court) ■

Exemple:

- La lettre **s** est encodée par ■ ■ ■ ■ ■
- La lettre **o** est encodée par ≈ ■ ≈ ■ ≈

On écrit alors **sos** de la manière suivante:

■ ■ ■ ■ ■ ≈ ■ ≈ ■ ≈ ■ ■ ■ ■ ■ ■ ■ ■ ■

💡 Notez l'espacement des lettres de 3 ■ !

Qu'on peut écrire plus facilement ... --- ...

💡 🙌 😊 En réalité, dans le cas de SOS, on peut ne pas séparer les lettres dans le code Morse international, mais n'allons pas trop compliquer notre tâche avec des cas particuliers!

💡 Un espace entre deux mots sera obtenu grâce à une attente d'une durée de 5 à 7 appuis courts.

Parties du TP

👉😊 On aura donc deux parties :

1. Envoyer du texte depuis la liaison série et l'émettre en code morse lumineux (*Scott*) à travers l'affichage d'une LED
2. Entrer du Morse à l'aide d'un bouton et afficher le texte correspondant sur le moniteur série.

1.2 Conditions de réalisation

Le TP sera à faire par groupes de 2: vous disposez pour le faire d'une partie suivie et d'un temps d'une semaine pour le terminer en autonomie.

1.3 Livrables

A l'issue de ce TP, il faudra rendre, sur Moodle (Aix)/Teams (Nantes):

- Le fichier CPP (croquis) produit, éventuellement séparé en plusieurs fichiers pour la facilité de lecture
- Une capture d'écran de votre montage sur TinkerCAD
- L'URL de votre montage sur TinkerCAD (pensez à le rendre public!)

💡 **A compter de cette session, vous aurez une semaine supplémentaire pour finir le TP.**

1.4 Contraintes

- Vous devrez n'utiliser que les bibliothèques déjà pré-incluses avec l'IDE Arduino (et disponibles dans TinkerCAD)
- Vous avez le droit d'utiliser les classes développées en cours
- Vous devrez organiser votre code de manière orientée objet en créant des classes et des méthodes.

1.5 Evaluation

Vous serez évalués sur:

- La propreté du rendu (Montage TinkerCAD, propreté et lisibilité du code)
- La qualité du montage
- L'aspect fonctionnel de la production et le respect des consignes
- La qualité du code C++ et le respect de l'organisation POO, le respect des principes **KISS** et **DRY**

2. Etape 1 : Afficher un message en Morse

2.1 Montage

Créez, dans TinkerCAD/en physique, un montage permettant d'allumer une LED.

⚠ Attention aux branchements! Prêtez attention à la masse (GND) et aux GPIO choisies en fonction du type de signal recherché.



2.2 Une classe MorseConverter

Créez une classe `MorseConverter` qui contiendra les méthodes nécessaires pour:

- Lire des caractères un à un sur la liaison série UART jusqu'à ce qu'il n'y ait plus rien à lire.
- Convertir un caractère ASCII en sa représentation en Morse dans le format de votre choix (il faudra inclure cette correspondance dans votre fichier!)
- Afficher une chaîne de caractères en Morse en faisant clignoter la LED selon une cadence donnée en propriété.

Slide suivant pour les détails 

💡 On ne gèrera ici que les lettres de A à Z et on ne considèrera ni chiffres, ni caractères spéciaux, qui correspondent aux codes ASCII 65 à 90. Utilisez le code fourni [sur Wikipedia!](#)

💡 Le code Morse ne considère pas la casse. Si vous voulez éviter les erreurs, sachez qu'Arduino dispose d'une fonction `toUpperCase()` sur son type `String`

⚠️ En cas de caractère non supporté, on fera clignoter très rapidement la LED 8 fois.

- Lancez votre programme, et faites en sorte d'afficher les chaînes entrées sur le moniteur série.
- Testez le programme sur plusieurs entrées

Indices

💡 Créez un tableau allant des index 0 à 25 et faites correspondre à chaque lettre aux indices ASCII 65 à 90!

💡 Rappelez-vous que `char` est un type numérique en C++ et qu'il peut être utilisé en lieu et place d'un `int` pour des valeurs de 0 à 255!

💡 Rappelez-vous qu'une chaîne de caractères "standard" C est un tableau de `char` terminé par le caractère nul `\0` qui peut donc être parcourue avec un `while` 😊

💡 ****Cadence d'affichage et de lecture **** : Vous pouvez choisir la cadence (le temps d'un clignotement et la limite de ce qui est considéré un appui court) de manière arbitraire. Si vous utilisez un dispositif analogique permettant de la régler (comme un potentiomètre), des points bonus seront attribués.

3. Etape 2 : Interpréter et envoyer du Morse

3.1 Montage

Ajoutez au montage précédent un bouton, et connectez-le aux broches pertinentes!

3.2 Extension de la classe MorseConverter

Ajoutez les méthodes à la classe MorseConverter pour lire du code Morse selon l'appui sur le bouton, selon la cadence choisie, et envoyer le résultat sur le moniteur série sous forme de texte lisible ASCII.

- Un appui inférieur à deux fois la cadence comptera comme un appui court
- Un appui supérieur à deux fois la cadence comptera comme un appui long

Détails sur le slide suivant 

- Une pause entre 3 fois la cadence et 7 fois la cadence comptera comme un espace
- Une pause supérieure à 7 fois la cadence signalera la fin du message.

Chaque lettre sera envoyée au fur et à mesure sur la liaison série.

Lorsque le message est terminé, on enverra un caractère de fin de ligne sur la liaison série.

💡 Appuyer sur le bouton allumera la LED pour suivre visuellement l'envoi du code.

⚠ Entrer un caractère invalide enverra sur la liaison série un passage à la ligne puis la chaîne `Caractère non reconnu\n`

Conseils

💡 Vous pouvez boucler sur le tableau de la première partie pour trouver le caractère correspondant au code entré par appui sur le bouton. Le caractère ASCII correspondra à $65 + \text{l'index du tableau}$!