

Systèmes interconnectés – SINT

Sujet de TP 1 : liaison série asynchrone

Centrale Nantes

P.-E. Hladik, pehladik@ec-nantes.fr

Version 1.0.0 (2 octobre 2023)

1 Objectifs pédagogiques

- utiliser un analyseur logique
- mettre en oeuvre une communication par liaison série asynchrone
- comprendre la structuration en couche
- structurer un code

2 Ce qu'il y a à faire

Ce premier TP s'effectue sur la carte Feather M0 WiFi et utilise l'analyseur logique LHT00SU1 avec LogicView.

Un compte rendu est à rendre (sur hippocampus) sous **forme d'un pdf** avec les réponses aux questions et les extraits de code nécessaires directement dans le document. Pensez à commenter votre code au fur et à mesure et répondez aux questions directement dans les fichiers sous forme de commentaire.

Sous Arduino la liaison série entre le PC et la carte Feather est nommée **Serial** et la liaison entre le PIC et la carte Feather est nommée **Serial1**.

Activité 1 : Retrouver un format de données

La documentation décrivant le format des données échangées entre la carte Feather M0 et le PIC a été partiellement perdue. Seul le format des données émises depuis la Feather M0 pour contrôler les leds est connu et rappelé dans le tableau ci-dessous.

Numéro du bit	7	6	5	4	3	2	1	0
Valeur	0	0	0	0	0	0 : éteindre 1 : allumer	00 : led RED 01 : led GREEN 10 : led YELLOW 11 : led BLUE	

TAB. 1: Description de l'octet de contrôle des leds

Votre première mission sera de retrouver la configuration de la liaison série et le format des données transmises par le PIC. Tout ce que nous savons c'est qu'à chaque fois que l'état d'un poussoir change, ou que l'encodeur est tourné ou appuyé, un octet est envoyé à la carte Feather M0 WiFi.

Pour vous aider dans cette mission, vous disposez de la carte Feather M0 et d'un analyseur logique (un guide sur l'utilisation de PulseView est disponible sur hippocampus). Attention avec PulseView il vous faudra choisir un **sample rate** cohérent avec les fréquences possible d'une liaison série. Il vous faudra aussi identifier les broches sur lesquelles sont connectées

Vous disposer d'un projet initial sur [hippocampus](#) pour pouvoir flasher la carte. Même si le code sur la Feather ne fait rien, des messages sont envoyés sur la liaison série dès qu'un bouton est pressé ou que l'encodeur est manipulé.

Activité 2 : Driver Arduino

Trouver le fichier `Uart.cpp` parmi la bibliothèque Arduino dédiée à l'architecture samd de adafruit. Quelle est la méthode de la classe `Uart` pour configurer le baudrate de la liaison série ?

A quelle autre classe la méthode de `Uart` fait-elle appel pour fixer le baudrate ? Retrouver le code de celle ci.

Une fois retrouver la nouvelle méthode, examiner son code pour faire le lien avec la section 34 de la datasheet du micro-contrôleur (voir hippocampus) et trouver le registre qui permet de fixer le baud rate.

Activité 3 : Contrôle des leds et affichage de l'encodeur

En utilisant le [projet de départ](#), créer un fichier `io.cpp` pour implémenter les fonctions déclarées dans `io.h` :

- `beginIO` configure la liaison série 1 (entre le PIC et le Feather)
- `ledWrite` allume ou éteint individuellement chaque led
- `pushButton` permet de savoir si un poussoir est appuyé ou non
- `encoderValue` retourne la valeur cumulée de l'encodeur, limitée entre 0 et 63

En utilisant ces fonctions, écrivez un programme permettant de :

- faire clignoter la led BUILTIN à 1 Hz ;
- allumer la led BLUE si on appuie sur le CLIC de l'encodeur et l'éteint quand il est relâché
- allumer la led YELLOW si on appuie sur le poussoir P0 et l'éteint quand il est relâché
- allumer la led GREEN si on appuie sur le poussoir P1 et l'éteint quand il est relâché
- allumer la led RED si on appuie sur le poussoir P2 et l'éteint quand il est relâché
- afficher sur la 4e ligne de l'écran la valeur cumulée de l'encodeur (voir annexe)

3 Annexe : bibliothèque LCD

La bibliothèque pour utiliser l'afficheur LCD est décrite sur :

<https://www.arduino.cc/en/Reference/LiquidCrystal>

Un exemple d'utilisation est fourni ci-dessous.

```
#include <LiquidCrystal.h>

LiquidCrystal lcd(18, 17, 16, 15, 14, 19) ;

void setup () {
  lcd.begin(20, 4); // 20 colonnes et 4 lignes
  lcd.setCursor(0, 3);
  lcd.print("hello, world!");
}
```

FIG. 1: Exemple de code utilisant la bibliothèque LCD Arduino