

# Systèmes interconnectés – SINT

## Sujet de TP 3 : SPI

Centrale Nantes

P.-E. Hladik, pehладик@ec-nantes.fr

Version 1.0.1 (2 octobre 2023)

### 1 Objectifs pédagogiques

- écriture d'un driver logiciel pour le SPI
- écriture d'un driver de communication MCP2515

### 2 Ce qu'il y a à rendre

Vous déposez sur hippocampus un document pdf avec les parties pertinente de votre code commenté.

### 3 Ecrire un drive logiciel

Ce TP s'effectue sur la carte Feather M0 WiFi. Vous pouvez piocher dans le code des deux activités précédentes.

Ce TP consiste à programmer une liaison **SPI logicielle** pour communiquer avec le MCP2515. Vous ne devez pas utiliser le périphérique SPI mais uniquement communiquer via les ports d'entrée-sortie (voir le cours). Le datasheet du MCP2525 est disponible sur le serveur pédagogique (a priori ça ne vous servira pas, tout a déjà été dit en cours).

Vous devez produire quatre fonctions :

- **Initialisation** : configuration des ports d'entrée-sortie.  
`void initialiserSPILogiciel (void) ;`
- **Échange de données SPI** : Cette fonction est appelée par les deux fonctions suivantes. Elle sérialise la donnée émise sur MOSI, et retourne la donnée reçue via MISO.  
`byte echangeSPI (const byte inDonneeEmise) ;`
- **Écriture d'un registre** : Affecte inValeur au registre inAdresse.  
`void ecrireRegistreMCP2515 (const byte inAdresse, const byte inValeur) ;`
- **Lecture d'un registre** : Cette fonction retourne la valeur du registre inAdresse.  
`byte lireRegistreMCP2515 (const byte inAdresse) ;`

L'envoi des bits se fait en MSB et il ne faut pas remettre à la ligne SS entre la transmission de 2 octets. Vous pouvez utiliser `delayMicroseconds` ([documentation](#)) pour créer le délai.

**Comment savoir si l'écriture et la lecture fonctionne correctement ?** On va écrire une valeur dans un registre, et relire ce registre et comparer la valeur lue avec la valeur écrite. Il faut sélectionner un registre qui se comporte comme une mémoire : par exemple, le registre TXB0D0, d'adresse 0x36. **Ecrivez votre fonction de test dans le croquis Arduino et faites une capture d'écran avec l'analyseur logique pour montrer les trames.**

#### 4 Utiliser le drive Arduino

Vous allez programmer une liaison SPI matérielle pour communiquer avec le MCP2515. Pour utiliser la librairie Arduino, il faut commencer par configurer les broches d'entrée-sortie convenablement. Pour cela suivez les indications ci-dessous.

**Inclusions :** Deux fichiers d'en-tête sont à inclure :

```
#include <SPI.h>
#include <wiring_private.h>
```

**Variable globale :** Une variable globale mySPI est à déclarer :

```
SPIClass mySPI (&sercom1, 10, 12, 11, SPI_PAD_0_SCK_3, SERCOM_RX_PAD_2) ;
```

**Initialisation :** Écrire une fonction initialiserSPIMateriel dans laquelle :

1. le port correspondant à SS est configuré en sortie, et placé au niveau haut (comme pour le SPI logiciel) ;
2. appeler `mySPI.begin ()`, cet appel configure la liaison SPI avec des paramètres par défaut ;
3. appeler `mySPI.beginTransaction(SPISettings (10 * 1000 * 1000, MSBFIRST, SPI_MODE0))`, cet appel fixe la vitesse ( $10 * 1000 * 1000 = 10 \text{ MHz}$ ), l'ordre de sérialisation des bits, et le mode ;
4. appeler `pinPeripheral (10, PIO_SERCOM)`, cet appel relie le port n°10 au module SERCOM ;
5. appeler `pinPeripheral (11, PIO_SERCOM)`
6. appeler `pinPeripheral (12, PIO_SERCOM)`

**Écriture d'un registre, lecture d'un registre :** Appeler la fonction `mySPI.transfer` à la place de la fonction `echangeSPI` utilisée lors de la liaison SPI logicielle.

**Ecrivez votre fonction de test dans le croquis Arduino et faites une capture d'écran avec l'analyseur logique pour montrer les trames.**