PREMIERE GENERALE

NUMERIQUE ET SCIENCES INFORMATIQUES

CARTE RASPBERRY PICO : COMMUNICATION SPI

1 – INTERFACES SPI DE LA CARTE RASPBERRY PICO

La carte Raspberry PICO intègre 2 interfaces SPI : **SPI0**, **SPI1** indépendantes l'une de l'autre. Plusieurs broches GPIO permettent d'accéder aux lignes des interfaces SPI.

Interface	MOSI	MISO	SCK	CS
SPI0	GPIO3, GPIO7, GPIO19	GPIO4, GPIO7, GPIO19	GPIO2, GPIO6, GPIO18	GPIO1, GPIO5, GPIO17
SPI1	GPIO11, GPIO15	GPIO8, GPIO12	GPIO10, GPIO14	GPIO9, GPIO13

Les interfaces SPI sont gérées par la classe **SPI** de la librairie **machine**. Il faut créer un objet de type **SPI**, sur lequel seront appliqués les différentes méthodes, en spécifiant le numéro de l'interface, la vitesse de communication (**baudrate**) et éventuellement l'état de repos des lignes du bus SPI (**polarity**), le front sur lequel les signaux changent d'état (**phase**). Il faut également définir les numéros des broches utilisées par l'interface.

```
from machine import SPI
nom_SPI = SPI(num_interface, bauderate = vitesse, polarity = etat_repos, phase = etat_front,
sck = Pin(num_pin), mosi = Pin(num_pin), miso = Pin(num_pin))
```

Méthode « read »

Cette méthode lit le nombre d'octets **nbytes** sur la ligne MISO, tout en écrivant l'octet spécifié par l'argument **write** sur la ligne MOSI.

```
nom_SPI.read(nbytes, write = 0x00);
```

Méthode « Write »

Cette méthode écrit les octets stockés dans un objet **data** sur la ligne MOSI. Il ne renvoie rien.

```
nom_SPI.write(data);
```

2 – AFFICHAGE DE LA PRESSION MESUREE PAR LE CAPTEUR MPL115A1

La mesure de la pression atmosphérique est réalisée au moyen du composant **MPL115A1**. Il s'agit d'un **baromètre** possédant une **interface SPI**. Sparkfun propose une petite carte équipée d'un **MPL115A1**.

- Capteur MPL115A1 -

La pression atmosphérique sera déterminée à partir de la pression compensée Pcomp :

$$\textbf{Pression (kPa)} = \textbf{Pcomp} \times \frac{115 - 50}{1023} + 50$$

La pression compensée P_{comp} peut être calculée à partir des données transmises par le capteur MPL115A1 :

$$\mathbf{Pcomp} = \mathbf{a0} + (\mathbf{b1} + \mathbf{c12} \times \mathbf{Tadc}) \times \mathbf{Padc} + \mathbf{b2} \times \mathbf{Tadc}$$

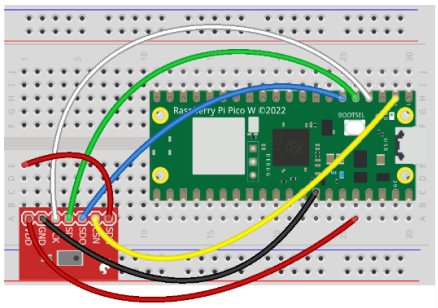
La valeur des coefficients a0, b1, b2, c11, c12 sont spécifiques à chaque capteur. Ces valeurs restent constantes et sont enregistrées dans les registres internes du capteur. Il s'agit de nombres signés à virgules codés sur 16 ou 14 bits.

Padc et Tadc correspondent aux valeurs brutes de la pression et de la température mesurées par le capteur. Ces valeurs codées sur 10 bits sont également mémorisées dans les registres internes du capteur.

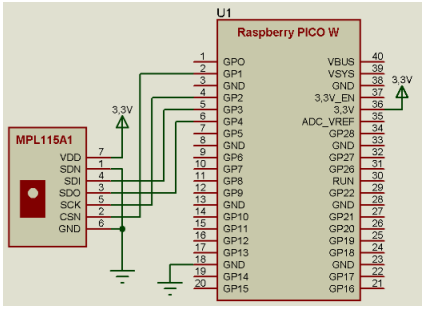
Le programme ci-dessous permet de récupérer la pression mesurée par le capteur MPL115A1 et de l'afficher sur la console :

```
from machine import SPI, Pin
import time

MPL115A1_SPI = SPI(0, baudrate = 200000, sck = Pin(2), mosi = Pin(3), miso = Pin(4))
pin_CS = Pin(1, mode = Pin.OUT)

# Lecture et calcul coefficients
pin_CS.value(0)                          # Sélection périphérique
MPL115A1_SPI.write(bytearray([0x88]))
a0Msb = int.from_bytes(MPL115A1_SPI.read(1), "little")
MPL115A1_SPI.write(bytearray([0x8A]))
a0Lsb = int.from_bytes(MPL115A1_SPI.read(1), "little")
MPL115A1_SPI.write(bytearray([0x8C]))
b1Msb = int.from_bytes(MPL115A1_SPI.read(1), "little")
MPL115A1_SPI.write(bytearray([0x8E]))
b1Lsb = int.from_bytes(MPL115A1_SPI.read(1), "little")
MPL115A1_SPI.write(bytearray([0x90]))
b2Msb = int.from_bytes(MPL115A1_SPI.read(1), "little")
MPL115A1_SPI.write(bytearray([0x92]))
b2Lsb = int.from_bytes(MPL115A1_SPI.read(1), "little")
MPL115A1_SPI.write(bytearray([0x94]))
c12Msb = int.from_bytes(MPL115A1_SPI.read(1), "little")
MPL115A1_SPI.write(bytearray([0x96]))
c12Lsb = int.from_bytes(MPL115A1_SPI.read(1), "little")
pin_CS.value(1)                          # Désélection périphérique
time.sleep_ms(3)

a0 = (a0Msb << 8 | a0Lsb) / 2**3
b1 = -(((~(b1Msb << 8 | b1Lsb)) & 0x7FFF)+1) / 2**13
b2 = -(((~(b2Msb << 8 | b2Lsb)) & 0x7FFF)+1) / 2**14
c12 = ((c12Msb << 8 |  c12Lsb) >> 2) / 2**22

while True:
    # Lire la pression et la température
    pin_CS.value(0)              # Sélection périphérique
    MPL115A1_SPI.write(bytearray([0x24])) # Lancer la conversion
    MPL115A1_SPI.write(bytearray([0x00]))
    pin_CS.value(1)              # Désélection périphérique
    time.sleep_ms(3)

    pin_CS.value(0)              # Sélection périphérique
    MPL115A1_SPI.write(bytearray([0x80]))
    padcMsb = int.from_bytes(MPL115A1_SPI.read(1), "little")
    MPL115A1_SPI.write(bytearray([0x82]))
    padcLsb = int.from_bytes(MPL115A1_SPI.read(1), "little")
    MPL115A1_SPI.write(bytearray([0x84]))
    tadcMsb = int.from_bytes(MPL115A1_SPI.read(1), "little")
    MPL115A1_SPI.write(bytearray([0x86]))
```

```
    tadcLsb = int.from_bytes(MPL115A1_SPI.read(1), "little")
    pin_CS.value(1) )              # Désélection périphérique
    time.sleep_ms(3)

    Pression = (padcMsb << 8 | padcLsb) >> 6
    Temperature = (tadcMsb << 8 | tadcLsb) >> 6

    #  Calcul de la pression compensée
    Pcomp = a0 + (b1 + c12 * Temperature) * Pression + b2 * Temperature

    # Calcul et affichage de la pression absolue
    Pabs = Pcomp * (115 - 50) / 1023.0 + 50
    Phecto = Pabs * 10;
    print("Pression : ", Phecto, " hecto Pascal")
    time.sleep(10)
```

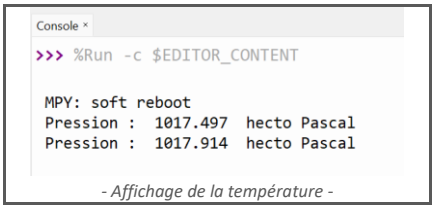


- Schéma de cablâge -



- Schéma struturel -

Après avoir exécuté le programme, la pression atmosphérique mesurée par le capteur MPL115A1 est affichée toutes les secondes sur la console.



- Affichage de la température -