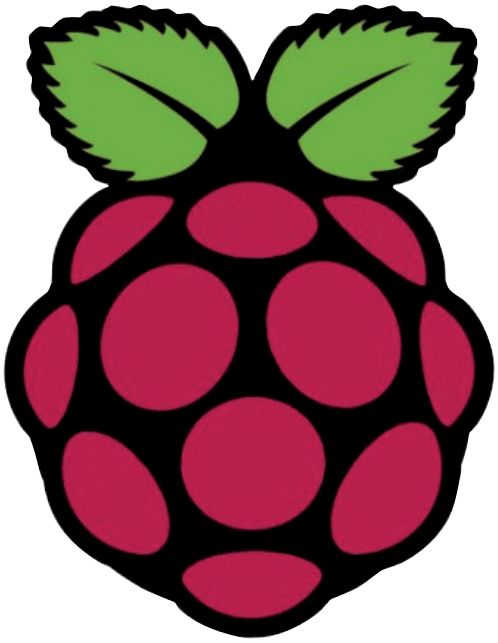
# Raspberry Pi part n° 4

## Hissez le (S)PI



**Tony Dixon**
(Royaume-Uni)

Après avoir vu comment utiliser l'interface série UART présente sur le connecteur d'extension du Raspberry, nous nous intéressons ici à son interface série SPI.

### L'interface SPI

Trois interfaces série sont disponibles depuis le connecteur d'extension du Raspberry : UART (cf. part nº 3), I²C, et SPI (*Serial Peripheral Interface*) [1].

Comme le montre le brochage du connecteur d'extension (**tableau 1)**, l'interface SPI est accessible depuis les broches 19 (MOSI), 21 (MISO), et 23 (SCK). Les deux signaux SCE (*SPI Chip Enable*) sont quant à eux présents sur les broches 24 (CE0) et 26 (CE1).

SPI permet d'établir une interface entre le Raspberry et d'autres périphériques avec un nombre de broches minimal et selon une configuration maître/esclave. Dans nos exemples, le PI sera toujours le maître SPI. D'un point de vue matériel, l'interface SPI a normalement besoin de quatre signaux pour fonctionner correctement : *Master Out*, *Slave In* (MOSI), *Master In*, *Slave Out* (MISO), *Serial Clock* (SCK) et *Chip Enable* (CEx).

### Duplicateur de ports d'E/S

Pour ce projet, nous allons augmenter le nombre de ports GPIO du Raspberry à l'aide du duplicateur de ports MCP23S17 à 16 bits de Microchip.

La **figure 1** montre le schéma d'un MCP23S17 relié à l'interface SPI du Raspberry Pi. Le cavalier J1 permet de choisir entre l'un des deux signaux SPI *chip enable* (sélection de circuit), à savoir CE0 ou CE1.

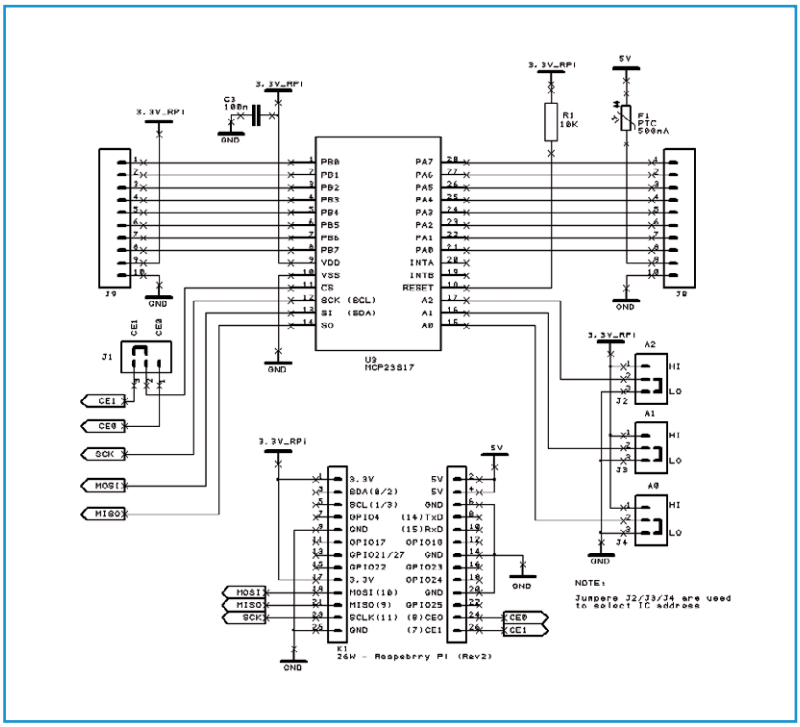



Figure 1.
Schéma de principe du duplicateur MCP23S17 relié au Pi.

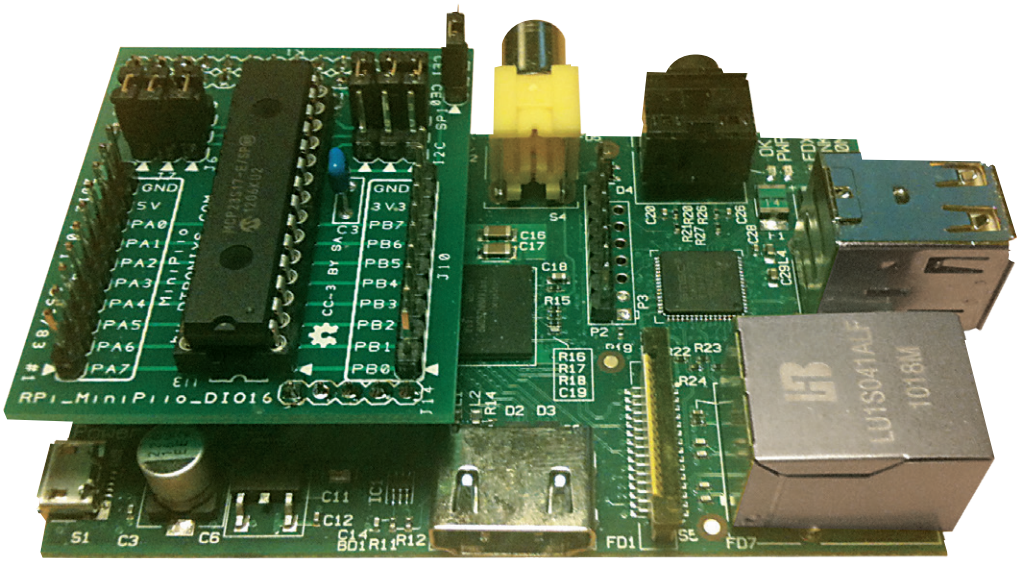



Figure 2.
Pi et le MCP23S17 sur une carte MiniPiio.

La **figure 2** montre notre configuration matérielle. Comme vous le voyez, nous avons assemblé notre interface MC23S17 sur une petite carte d'extension [3].

### Installation du module Python pour SPI

Nous allons de nouveau recourir au langage Python pour programmer nos exemples. L'interpréteur Python est installé par défaut dans la distribution Raspbian, mais aucune bibliothèque Python pour SPI n'est fournie d'avance. Commençons donc par télécharger et installer le module Python pour SPI appelé *py-spidev* [2]. Entrez les commandes suivantes dans une console Lxterminal (**fig. 3**) :

```
cd ~

git clone git://github.com/doceme/
   py-spidev

cd py-spidev/

sudo python setup.py install

(ou :
sudo apt-get install git-core
   python-dev
sudo apt-get install python-pip
sudo pip install spidev)
```

Puisque l'interface SPI matérielle est désactivée par défaut, nous devons l'activer en modifiant le fichier appelé *blacklist*. Ouvrons ce fichier avec par exemple l'éditeur nano :

```
sudo nano /etc/modprobe.d/raspi-
   blacklist.conf
```

Trouvez la ligne qui contient **blacklist spi-bcm2708**, commentez-la en ajoutant le caractère # au début de la ligne, puis sauvegardez le fichier. Pi doit être redémarré pour que le changement soit pris en compte :

```
sudo reboot
```

Lancez une nouvelle session LXTerminal et, pour vérifier que vous avez bien deux fichiers de périphériques SPI (un pour chaque signal SPI *Chip Select*), entrez :

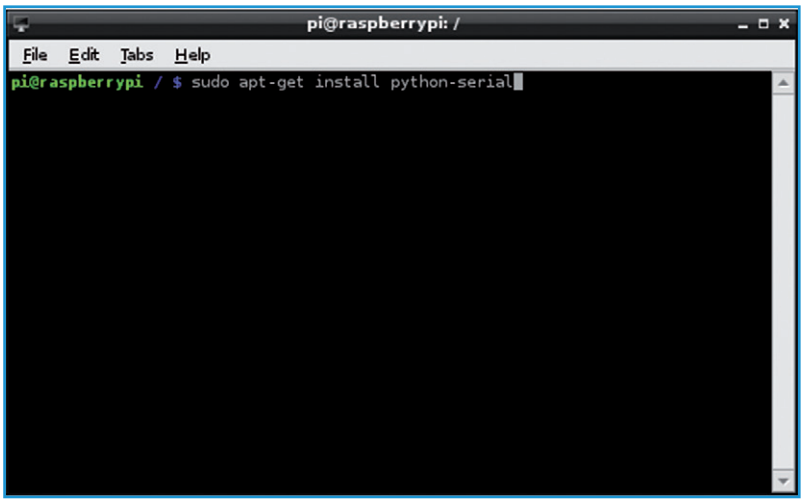



Figure 3.
LXTerminal.

```
ls /dev/spi*
```

La sortie de cette commande devrait être :

```
/dev/spidev0.0
/dev/spidev0.1
```

### Programme d'exemple : mcp23s17.py

Maintenant que spidev est installé, nous pouvons tester l'interface SPI avec un court programme chargé d'allumer des LED reliées au duplicateur de ports GPIO.

Double-cliquez sur l'icône IDLE du bureau pour lancer l'EDI et la console Python (**fig. 4**).

Dans le menu *File*, sélectionnez *New Window* pour créer un nouveau programme dans l'éditeur.

Copiez le programme du **listage** dans l'éditeur (**fig. 5**).

Sauvegardez le programme saisi. Retournez ensuite dans la console Lxterminal, et entrez la commande suivante pour le rendre exécutable :

```
chmod +x mcp23s17.py
```

Et pour exécuter ce programme :

```
sudo ./mcp23s17.py
```

(130211 – version française : Hervé Moreau)

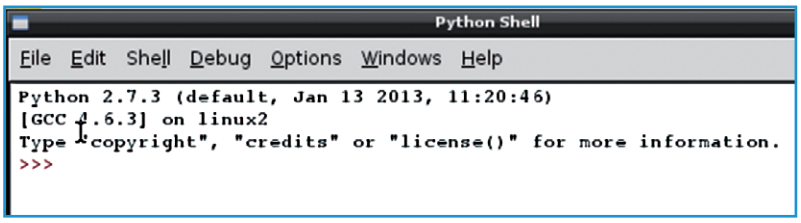



Figure 4. L'environnement IDLE pour Python.

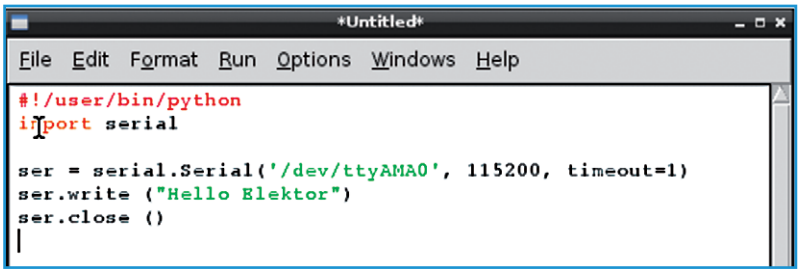



Figure 5. L'éditeur de IDLE.

**Listage**

```
#!/usr/bin/python

import spidev
import time

spi = spidev.SpiDev()
spi.open(0,0)

while True:
    spi.xfer([0,0,0])     # turn all lights off
    time.sleep(1)
    spi.xfer([1,255,254]) # turn all lights on
    time.sleep(1)
```

| commandes spidev | |
|---|---|
| `spi.open (0,0)` | ouvre le bus SPI 0 en utilisant CE0 |
| `spi.open (0,1)` | ouvre le bus SPI 0 en utilisant CE1 |
| `spi.close ()` | déconnecte l'objet de l'interface |
| `spi.writebytes ([tableau d'octets])` | écrit un tableau d'octets dans le périphérique SPI |
| `spi.readbytes (len)` | lit *len* octets reçus du périphérique SPI |
| `spi.xfer2 ([tableau d'octets])` | envoie un tableau d'octets en laissant le signal CEx actif en permanence |
| `spi.xfer ([tableau d'octets])` | envoie un tableau d'octets en désactivant et réactivant le signal CEx à chaque octet transmis |

## Liens

[1] www.raspberrypi.org

[2] www.github.com/doceme/py-spidev

[3] www.dtronixs.com

**Tableau 1. Brochage du connecteur d'extension**

| Nom de la broche | Fonction | Autre fonction | RPi.GPIO |
|---|---|---|---|
| P1-02 | 5,0V | - | - |
| P1-04 | 5,0V | - | - |
| P1-06 | GND | - | - |
| P1-08 | GPIO14 | UART0_TXD | RPi.GPIO8 |
| P1-10 | GPIO15 | UART0_RXD | RPi.GPIO10 |
| P1-12 | GPIO18 | PWM0 | RPi.GPIO12 |
| P1-14 | GND | - | - |
| P1-16 | GPIO23 | | RPi.GPIO16 |
| P1-18 | GPIO24 | | RPi.GPIO18 |
| P1-20 | GND | - | - |
| P1-22 | GPIO25 | | RPi.GPIO22 |
| P1-24 | GPIO8 | **SPI0_CE0_N** | RPi.GPIO24 |
| P1-26 | GPIO7 | **SPI0_CE1_N** | RPi.GPIO26 |

| Nom de la broche | Révision 1 de la carte | | Révision 2 de la carte | |
|---|---|---|---|---|
| | Fonction | Autre fonction | Fonction | Autre fonction |
| P1-01 | 3,3V | - | 3,3V | - |
| P1-03 | GPIO0 | I2C0_SDA | GPIO2 | I2C1_SDA |
| P1-05 | GPIO1 | I2C0_SCL | GPIO3 | I2C1_SCL |
| P1-07 | GPIO4 | GPCLK0 | GPIO4 | GPCLK0 |
| P1-09 | GND | - | GND | - |
| P1-11 | GPIO17 | RTS0 | GPIO17 | RTS0 |
| P1-13 | GPIO21 | | GPIO27 | |
| P1-15 | GPIO22 | | GPIO22 | |
| P1-17 | 3,3V | - | 3,3V | - |
| P1-19 | GPIO10 | SPI0_MOSI | GPIO10 | SPI0_MOSI |
| P1-21 | GPIO9 | SPI0_MISO | GPIO9 | SPI0_MISO |
| P1-23 | GPIO11 | SPI0_SCLK | GPIO11 | SPI0_SCLK |
| P1-25 | GND | - | GND | - |

Note : I2C0_SDA et I2C0_SCL (GPIO0 & GPIO1), ainsi que I2C1_SDA et I2C1_SCL (GPIO2 & GPIO3) sont dotées de résistances de rappel de 1,8 kΩ au 3,3 V.