Electronics
Serial Communication-SPI

Terry Sturtevant

Wilfrid Laurier University

October 3, 2018
Serial Communication -SPI

Serial Peripheral Interface
Master/slave communication
Uses 3 signals (and Ground), MISO, MOSI, SCLK and chip selects for each slave device
Synchronous, so master controls clock rate

Terry Sturtevant
Electronics Serial Communication-SPI
Serial Communication - SPI

- Serial Peripheral Interface
Serial Communication -SPI

- Serial Peripheral Interface
- Master/slave communication
Serial Communication - SPI

- Serial Peripheral Interface
- Master/slave communication
- Uses 3 signals (and Ground), MISO, MOSI, SCLK
  and chip selects for each slave device
Serial Communication -SPI

- Serial Peripheral Interface
- Master/slave communication
- Uses 3 signals (and Ground), MISO, MOSI, SCLK
  and chip selects for each slave device
- Synchronous, so master controls clock rate
SPI transfers can happen in both directions simultaneously.

Terry Sturtevant
SPI transfers can happen in both directions simultaneously.
Serial Communication - SPI
Raspberry Pi Python SPI Library

PySpidev

```
spi = spidev.SpiDev()
create object
spi.open(0,0)
open port, device
There is normally one SPI port on the Raspberry Pi.
SPI0 has two associated devices, selected by chip selects CE0 and CE1.
So, spi.open(0,0) means to connect to the device using CE0 on SPI0.
```
PySpidev

- \texttt{spi = spidev.SpiDev()}
  create object
PySpidev

- `spi = spidev.SpiDev()`
  create object
- `spi.open(0,0)`
  open port, device
PySpidev

- `spi = spidev.SpiDev()`
  - create object
- `spi.open(0,0)`
  - open port, device
There is normally one SPI port on the Raspberry Pi.
PySpidev

- `spi = spidev.SpiDev()`
  - create object
- `spi.open(0,0)`
  - open port, device
  - There is normally one SPI port on the Raspberry Pi.
  - It is designated as port 0
PySpidev

- `spi = spidev.SpiDev()`  
  create object

- `spi.open(0,0)`  
  open port, device
  There is normally one SPI port on the Raspberry Pi.
  It is designated as port 0
  SPI0 has two associated devices, selected by chip selects
  CE0 and CE1
PySpidev

- `spi = spidev.SpiDev()`
  - create object
- `spi.open(0,0)`
  - open port, device
  
  There is normally one SPI **port** on the Raspberry Pi.
  
  It is designated as port **0**
  
  SPI0 has two associated **devices**, selected by **chip selects**
  
  CE0 and CE1

- **So, `spi.open(0,0)` means to**
  - connect to the device using CE0 on SPI0
(continued)
(continued)

- So, `spi.max_speed_hz = 5000` means to limit the SPI clock speed to about 5kHz.
(continued)

- So, `spi.max_speed_hz = 5000` means to limit the SPI clock speed to about 5kHz

- `response = spi.xfer2([0xAA])`
  - transfer one byte
  - CS held active between blocks
(continued)

- So, `spi.max_speed_hz = 5000` means to limit the SPI clock speed to about 5kHz
- `response = spi.xfer2([0xAA])`
  transfer one byte
  CS held active between blocks
- `response = spi.xfer([values])`
  transfer bytes
  CS released and reactivated between blocks
PySpidev (continued)

```python
spi.writebytes([values])
write bytes
spi.readbytes(len)
read len bytes
spi.cshigh
get or set active state of CS
spi.close()
close port
```
PySpidev (continued)

- `spi.writebytes([values])`
  
  write bytes
PySpidev (continued)

- `spi.writebytes([values])`
  write bytes

- `spi.readbytes(len)`
  read \( len \) bytes
PySpidev (continued)

- `spi.writebytes([values])`
  write bytes

- `spi.readbytes(len)`
  read \( len \) bytes

- `spi.cshigh`
  get or set active state of CS
PySpidev (continued)

- `spi.writebytes([values])`
  write bytes
- `spi.readbytes(len)`
  read `len` bytes
- `spi.cshigh`
  get or set active state of CS
- `spi.close()`
  close port
PySpidev sample code

```python
import spidev
spi = spidev.SpiDev()
spi.open(0, 0)  # port, device
# use port 0, chip select CE0
spi.max_speed_hz = 5000
while True:
    str_val = input("val (0...255, q=quit) :")
    if str_val == 'q':
        break
    else:
        value = int(str_val)
dummy = spi.xfer2([49, value])
spi.close()
```
PySpidev sample code

```python
import spidev
spi = spidev.SpiDev()
spi.open(0,0)  #port,device
# use port 0, chip select CE0
spi.max_speed_hz = 5000
while True:
    strval = input("val (0...255, q=quit):")
    if strval == 'q':
        break
    else:
        value = int(strval)
        dummy = spi.xfer2([49,value])
spi.close()
```

Terry Sturtevant
Electronics Serial Communication-SPI