Electronics
Serial Communication-SPI

Terry Sturtevant

Wilfrid Laurier University

November 14, 2017

Serial Communication -SPI

Serial Communication -SPI

- Serial Peripheral Interface

Serial Communication -SPI

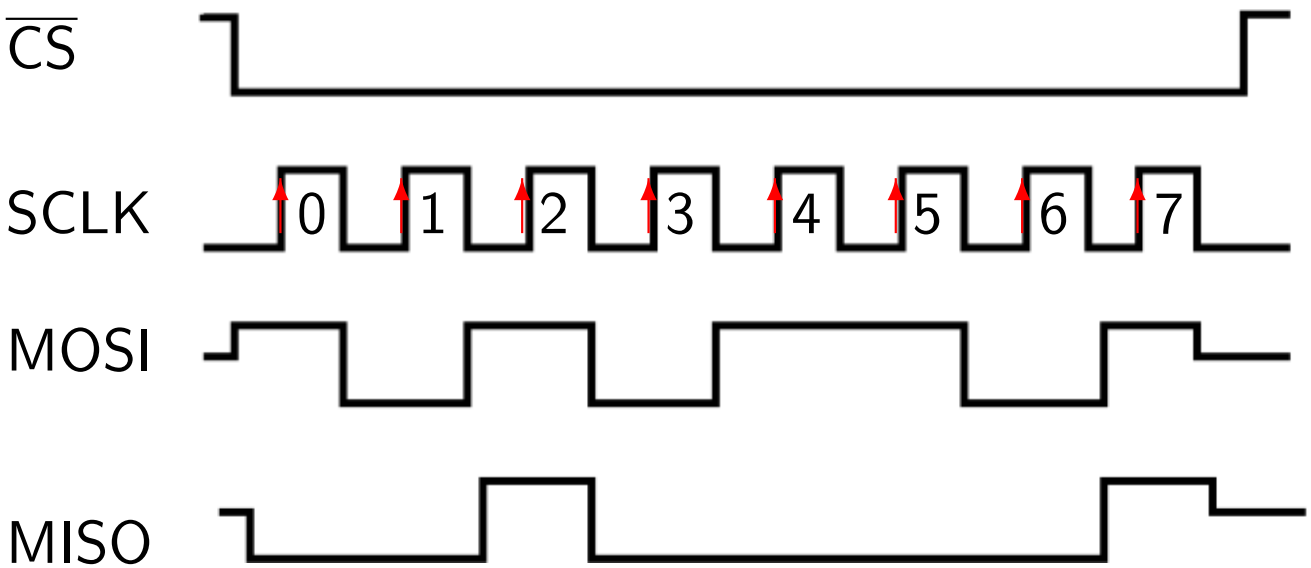
- Serial Peripheral Interface
- Master/slave communication

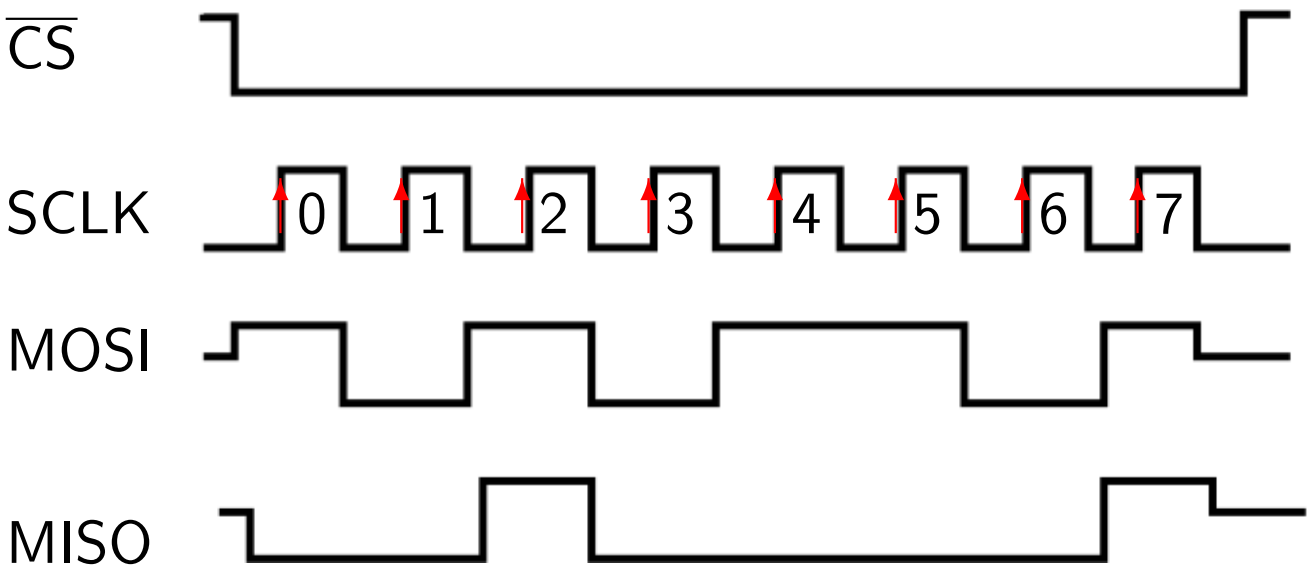
Serial Communication -SPI

- 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


$\overline{CS}$
SCLK
0
1
2
3
4
5
6
7
MOSI
MISO



SPI transfers can happen in both directions simultaneously.

PySpidev

PySpidev

- **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**

  CE**0** and CE**1**

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**

  CE**0** and CE**1**
- So, **spi.open(0,0)** means to

  connect to the device using CE0 on SPI0

PySpidev

(continued)

PySpidev

(continued)

- **response = spi.xfer2([0xAA])**

  transfer one byte

  CS held active between blocks

PySpidev

(continued)

- **response = spi.xfer2([0xAA])**

  transfer one byte

  CS held active between blocks

- **response = spi.xfer([values])**

  transfer bytes

  CS realeased and reactivated between blocks

PySpidev (continued)

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

PySpidev sample code

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