SI4432 Raspberry

Socket Server

run

sudo python socketserver.py :8889

located in

pi@raspberrypi ~/web2py/applications/SolarPower_Molenhoek/modules $

Define IO-pins

```python
# Raspberry
from time import sleep
import RPi.GPIO as GPIO
# to use Raspberry Pi board pin numbers
#GPIO.setmode ( GPIO.BOARD )
# to use Broadcom pin numbers
GPIO.setmode ( GPIO.BCM )

SI4432_Shutdown_Pin = 24
SI4432_GPIO2_Pin    = 25
ShutDown_Available  = True

GPIO.setup ( SI4432_GPIO2_Pin   , GPIO.IN  )
GPIO.setup ( SI4432_Shutdown_Pin, GPIO.OUT )

print 'Setup Done'
```

```
Setup Done

..... Raspberry process has finished .....
```

Define SPI

```python
# Raspberry
import spidev

SI4432 = spidev.SpiDev ()
SI4432.open ( 0, 0 )         # otherwise parameters will not be set

#spi.lsbfirst = True      # can't be changed
#spi.cshigh = True
#spi.mode = 3
SI4432.max_speed_hz = 4900000
print 'Bits per word :'  , SI4432.bits_per_word
print 'CSHigh :'         , SI4432.cshigh
print 'Loop :'           , SI4432.loop
print 'Max Speed [Hz] :' , SI4432.max_speed_hz
print 'ThreeWire :'      , SI4432.threewire
print 'Mode CPOL / CPHA :', SI4432.mode
print 'LSB first :'      , SI4432.lsbfirst
print dir(SI4432)
```

```
Bits per word : 8
CSHigh : False
Loop : False
Max Speed [Hz] : 4900000
ThreeWire : False
Mode CPOL / CPHA : 0
LSB first : False
['__class__', '__delattr__', '__doc__', '__format__', '__getattribute__', '__hash__', '__init__',
'__new__', '__reduce__', '__reduce_ex__', '__repr__', '__setattr__', '__sizeof__', '__str__',
'__subclasshook__', 'bits_per_word', 'close', 'cshigh', 'loop', 'lsbfirst', 'max_speed_hz', 'mode',
'open', 'readbytes', 'threewire', 'writebytes', 'xfer', 'xfer2']

..... Raspberry process has finished .....
```

Read SI4432

```
# Raspberry
def SI4432_Read ( Address ) :
  Response = SI4432.xfer ( [ Address, 0 ] )
  #print 'Response', Response
  return Response [1]

print hex ( SI4432_Read ( 0x30 ))
```

```
0x8d

..... Raspberry process has finished .....
```

Write SI4432

```
# Raspberry
procedure SI4432_Write ( byte in reg, byte in data ) is
  CSN = Low
  SPI_RW ( reg | 0x80 )
  SPI_RW ( data )
  CSN = high

```

Reset SI4432

```
# Raspberry
def SI4432_Reset () :
  if ShutDown_Available :
    GPIO.output ( SI4432_Shutdown_Pin, GPIO.HIGH )
    sleep ( 0.1 )                                # datasheet: minimal 20 msec
    GPIO.output ( SI4432_Shutdown_Pin, GPIO.LOW )
    sleep ( 0.04 )                               # datasheet: minimal 20 msec
                                                 # tested: 20 is too small, 30 seems
  else :
    # always perform a system reset (don't send 0x87)
    ##SI4432_Write( 0x07, 0x80 )    -- <<<< CRITICAL

    # here we have a contradiction
    #   1, we're not allowed to do any SPI commnunication before the reset is complete
    #   2, we must check CHIPRDY before doing any SPI communication
    # So we implement a wait here
    sleep ( 1 )

    # wait for chiprdy bit
    #while ( SI4432_Read ( 0x04 ) & 0x02 ) == 0 :
    #  pass
SI4432_Reset()
print 'reset done'
```

```
reset done

..... Raspberry process has finished .....
```

Dump

```
# Raspberry
def SI4432_Dump_Registers () :
  Line = []
  for i in range ( 128 ) :
    Line.append ( SI4432_Read ( i ) )
    if ( i % 16 ) == 0 :
      print Line
      Line = []
  print Line

  #SI4432_Hangup_Test_Serial ()
  #serial_hw_write ( si4432_support_version )

SI4432_Dump_Registers ()
```

```
[8]
[6, 32, 32, 0, 0, 3, 1, 0, 127, 6, 0, 0, 0, 0, 0, 0]
[0, 32, 0, 3, 0, 1, 0, 0, 1, 20, 0, 1, 68, 10, 3, 100]
[1, 71, 174, 2, 143, 0, 30, 0, 0, 0, 0, 24, 188, 38, 8, 141]
[0, 12, 34, 8, 42, 45, 212, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 255, 255, 255, 255, 0, 0, 0, 0, 255, 8, 8, 8, 16, 0]
[0, 223, 82, 32, 100, 0, 1, 135, 0, 1, 0, 14, 0, 0, 0, 160]
[0, 36, 0, 0, 130, 66, 31, 3, 32, 157, 0, 1, 24, 10, 61, 12]
[0, 32, 0, 0, 117, 187, 128, 25, 0, 0, 3, 55, 4, 55, 131]

..... Raspberry process has finished .....
..... Raspberry process has finished .....
```