

Application Note AN168:

Raspberry Pi to SenseAir S8 CO2 Sensor via UART

Introduction

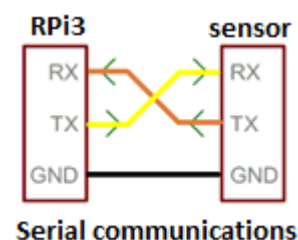
The **Raspberry Pi (RPI)** is a credit card size computer. The RPi 3 used here includes four USB connections plus Wi-Fi and Bluetooth. The HDMI video output requires a HDMI cable and appropriate HDMI capable LCD monitor. A USB keyboard and USB mouse will be necessary.



The **S8 Miniature CO2 Sensor** is one of the smallest non-dispersive infrared (NDIR) sensors on the market. A new optical path design, Enables the S8 to accurately measure carbon dioxide levels up to 5%, yet it has less than a 30x20x10mm footprint and runs on only 30mA. The RPi3 has sufficient 5V current available to power the S8.



For this app note, the connection to the sensor is via the on-board UART. The RPi3 offers a UART TXD-RXD connection for operating a S8 sensor. In order to proceed with this app note, you will need Raspian 8. We recommend the RPi3. The Raspian 8 operating system is used here and was installed using the NOOBS installer. For more details on what is NOOBS, access this link:
<https://www.raspberrypi.org/help/faqs/#generalNoobs>



RPi example code for the S8 sensor is included below.

Setting up your RPi3

1. Connect your Raspberry Pi to a USB keyboard, mouse and HDMI LCD monitor and 5VDC power supply.
2. Power up your RPi3 and LCD monitor. In order to get access to the attached python files, you will need an internet connection. RPi3 offers a built in Wi-Fi device. On your monitor, observe the upper right corner: an icon of a terminal. Click on this, select your available Wi-Fi network and enter your network password.

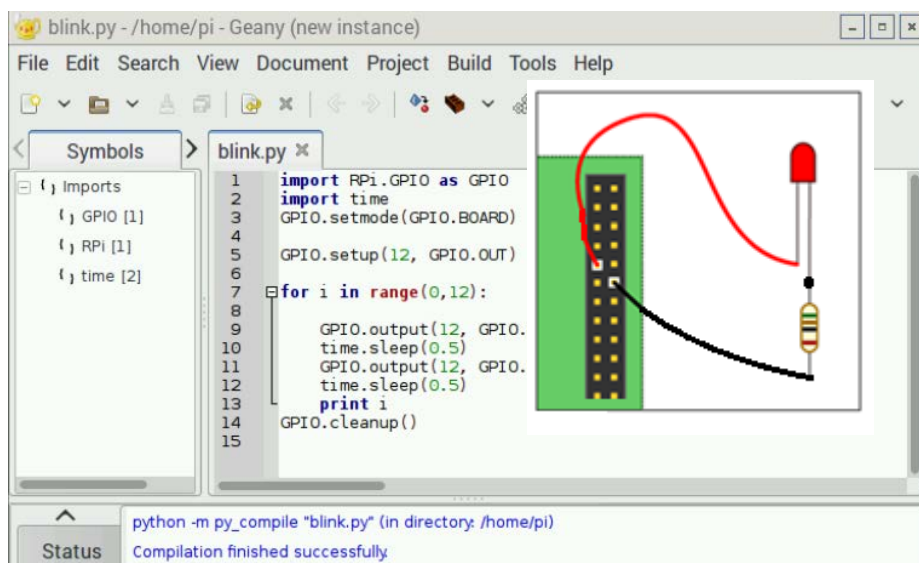
3. Verify that you are Wi-Fi connected: on the monitor, upper left corner is an icon that looks like a *globe*. This is the internet browser. Verify this browser is functioning by opening the browser, then copy the S8.py and Blink.py python code to the RPi3. Both programs are at the end of this app note, and can be copied and pasted into notepad text files with .py extensions or you can download them from AN168 in the App Notes tab at CO2Meter.com.



If you are new to RPi, it is highly advisable to run the Blink example. Even if you are an experienced RPi user, **blink.py** will confirm your connections Raspian operating system and Geany editor are working.

Running the Blink Example

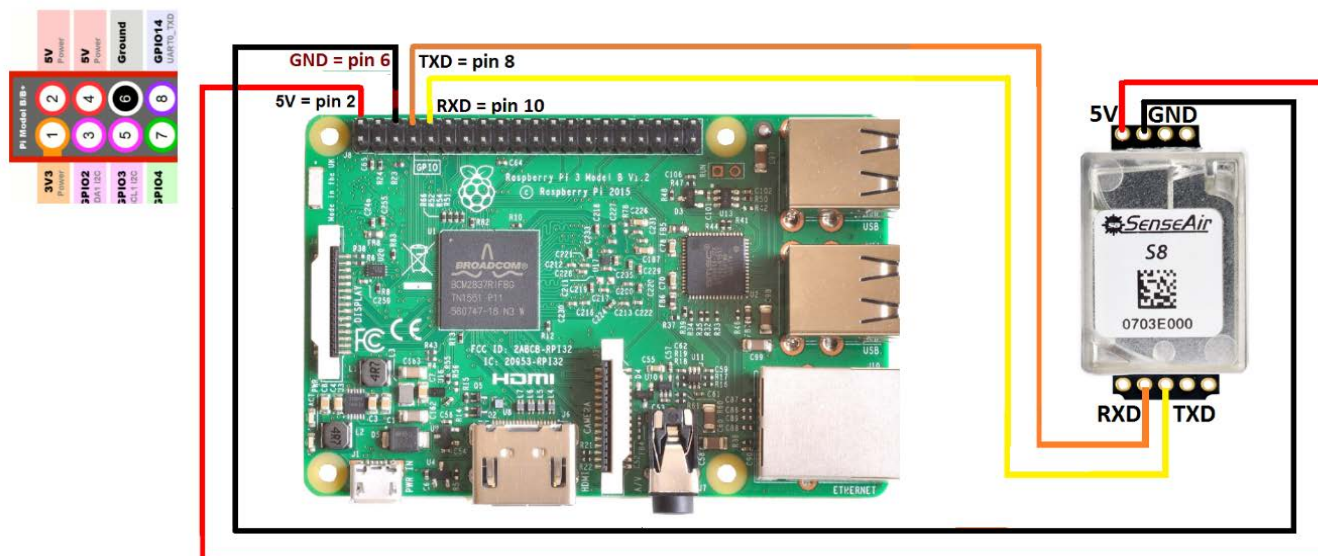
Included in this app note is the blink.py python code. Running the example will confirm your Raspian installation. Blink can be opened with the Raspian editor called Geany. Connect a LED and resistor to the RPi pin header as shown, to GPIO pins 9 and 12. The longer lead on the LED is positive and connects through a 270 ohm ¼ watt resistor to pin 12. The other LED lead connects to pin 9 which is GND.



To compile and run Blink:

1. On the RPI LCD monitor, click on Menu > Programming > Geany. Then click on File> Open and select the blink.py python file you created above. Observe above screen to connect the LED.
2. Click on Build > Compile.
3. Next Build>Execute. Observe that the LED will blink 12 times.

Connecting RPi3 to the S8 Sensor



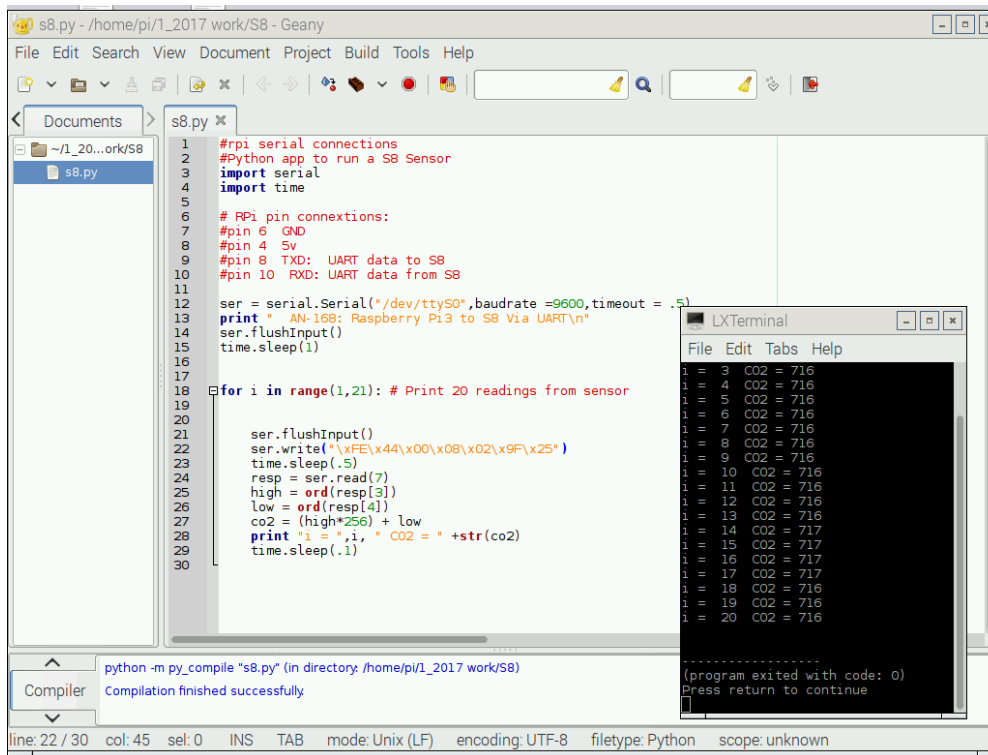
Wire colors match the schematic diagram above.

Creating your RPi3-S8 Project

1. Connect the RPi3 to the S8 sensor as shown. You may use the 5V supply on the Rpi.
2. On the RPI LCD monitor, click on Menu>Programming>Geany. Then click on File > Open and select the s8.py file you've created. Observe the following screen:

Very Important: On line 12, the UART connection is ttyS0. Your tty connection may differ.

This python program has been verified to work properly. Click on Build>compile. Then to run the program, Build>execute. The output is shown here:



The screenshot shows the Geany IDE with the file `s8.py` open. The code in the editor is as follows:

```

1  #!python serial connections
2  #Python app to run a S8 Sensor
3  import serial
4  import time
5
6  # RPi pin connections:
7  #pin 6 GND
8  #pin 4 5v
9  #pin 8 TXD: UART data to S8
10 #pin 10 RXD: UART data from S8
11
12 ser = serial.Serial("/dev/ttyS0",baudrate =9600,timeout = .5)
13 print " AN-168: Raspberry Pi3 to S8 Via UART\n"
14 ser.flushInput()
15 time.sleep(1)
16
17
18 for i in range(1,21): # Print 20 readings from sensor
19
20     ser.flushInput()
21     ser.write("\xFF\x44\x00\x08\x02\x9F\x25")
22     time.sleep(.5)
23     resp = ser.read(7)
24     high = ord(resp[3])
25     low = ord(resp[4])
26     co2 = (high*256) + low
27     print "i = %.1, " CO2 = " +str(co2)
28     time.sleep(.1)
29
30

```

The LXTerminal window shows the output of the program, displaying 20 readings of CO2 concentration:

```

i = 3 CO2 = 716
i = 4 CO2 = 716
i = 5 CO2 = 716
i = 6 CO2 = 716
i = 7 CO2 = 716
i = 8 CO2 = 716
i = 9 CO2 = 716
i = 10 CO2 = 716
i = 11 CO2 = 716
i = 12 CO2 = 716
i = 13 CO2 = 716
i = 14 CO2 = 717
i = 15 CO2 = 717
i = 16 CO2 = 717
i = 17 CO2 = 717
i = 18 CO2 = 716
i = 19 CO2 = 716
i = 20 CO2 = 716

```

The status bar at the bottom of the IDE indicates: line: 22 / 30 col: 45 sel: 0 INS TAB mode: Unix (LF) encoding: UTF-8 filetype: Python scope: unknown.

S8.py

```
#rpi serial connections
#Python app to run a S8 Sensor
import serial
import time

#Rpi pin connections:
#pin 6   GND
#pin 4   5v
#pin 8   TXD:  UART data to S8
#pin 10  RXD:  UART data from S8

ser = serial.Serial("/dev/ttyS0",baudrate =9600,timeout = .5)
print "  AN-168: Raspberry Pi3 to S8 Via UART\n"
ser.flushInput()
time.sleep(1)

for i in range(1,21): # Print 20 readings from sensor

    ser.flushInput()
    ser.write("\xFE\x44\x00\x08\x02\x9F\x25")
    time.sleep(.5)
    resp = ser.read(7)
    high = ord(resp[3])
    low = ord(resp[4])
    co2 = (high*256) + low
    print "i = ",i, " CO2 = " +str(co2)
    time.sleep(.1)
```

Blink.py

```
import RPi.GPIO as GPIO
import time
GPIO.setmode(GPIO.BOARD)

GPIO.setup(12, GPIO.OUT)

for i in range(0,12):

    GPIO.output(12, GPIO.HIGH)
    time.sleep(0.5)
    GPIO.output(12, GPIO.LOW)
    time.sleep(0.5)
    print i
GPIO.cleanup()
```