K-Series / Cozir / GSS
Raspberry Pi Communication Guide

Foreword:

This is a guide to getting a Co2meter.com sensor working with a Raspberry Pi computer. In this example I will show scripts for both GSS/Cozir sensors as well as the K-series sensors, but It can easily be adapted to any sensor with UART (Tx/Rx) communication

Outline of Steps:

Assuming you already have a Raspberry Pi set up and running, there are only five steps you need to follow to start using it to read from a CO2meter.com sensor. If you have not yet set it up, There is a quick start guide here: Raspberry Pi Quick start guide

1) Change Keyboard Layout
2) Set up UART on the Raspberry Pi
3) Install python-serial
4)Connect Sensor
5) Run script

1. Change keyboard Layout:

By default the Raspberry Pi uses the british keyboard layout. If you are using a U.S. Keyboard layout you will need to change the settings so the symbols are mapped correctly.

There are a few way to do this. The easiest is to open a terminal Start->Accessories->Root Terminal.
Now type:

```
sudo nano /etc/default/keyboard
```

this will open up a text file used to configure the keyboard layout. There will be a line that says :

```
“XKBLAYOUT=”gb”
```

using the arrow keys, move the cursor and change the **“gb”** to **“us”**
now press **ctrl+x** and it will ask if you want to save the changes. Press **'y'** to save, and then **[enter]** to keep the same name. Now you need to reboot the Pi. Do this by typing in:

```
sudo reboot
```

2. Set up the UART lines:

Now that are keyboard is mapped correctly we need to set up the UART lines. The Raspberry Pi has a breakout header that has two pins for Tx/Rx but they are set by default to send out debug information about the device. We need to disable that, so that we can use it to talk to a sensor.

To do this we need to edit two more configuration files:

/boot/cmdline.txt
/etc/inittab

Open up another Root Terminal (Start->Accessories->Root Terminal). Although theses files are easily replaced its always a good idea to make a backup. So type in :

```
cp /boot/cmdline.txt /boot/cmdline.bak
cp /etc/inittab /etc/inittab.bak
```

Now that they are backed up we can edit them using the nano editor. First we'll edit cmdline.txt

```
nano /boot/cmdline.txt
```

Delete **"console=ttyAMA0,115200"** and **"kgdboc=ttyAMA0,115200"** and use **ctrl+x** to save and exit.

Next edit inittab :

```
nano /etc/inittab
```

For this file we just need to Comment out the last line of the file. It should read "T0:23:respawn....". Just place a '#' at the beginning so it reads "#T0:23:respawn...." , then use ctrl+x to save and exit.

3. Install Python-Serial:

The script that we will be a python script, and it relies on a module called python-serial. In the terminal type:

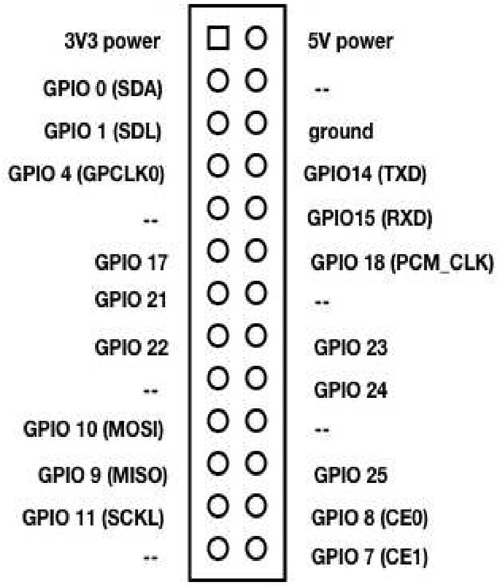
```
sudo apt-get install python-serial
```

this will go out and get the module and then install it for you

4. Connect Sensor:

Now we are ready to connect the Sensor to the Raspberry Pi. Check the Pinout of your sensor in the data sheet to make sure you have the right connections. Some sensors will be able to get their power from the 5v Pin on the break out, but it might not be enough for current for others. In the example we will use a power supply.

Here is the pinout for the Raspberry Pi breakout header, and the connections needed to communicate:



Raspberry Pi	Sensor	Power Supply
GPIO14 (TXD)	Rx	N/a
GPIO15 (RXD)	Tx	N/A
Ground	Ground	Ground
	5V+	5V+

5. Run the Script:

Now that everything is connected you just need to run the python scripts included with this note, or you can write your own. The scripts are very basic and just report the data back to the console, but they provide a good building block for those who wish to code more elaborate applications.

To run the script, just navigate to where they are downloaded in the terminal and type:

```
python [script name]
```

so for the cozir sensor you would type :

```
python cozir.py
```

to view the source of the script just open it in any text editor.