Summary of operation of 4x4 MOX sensor board with Arduino

Jordi Fonollosa; July 2017

Connections:

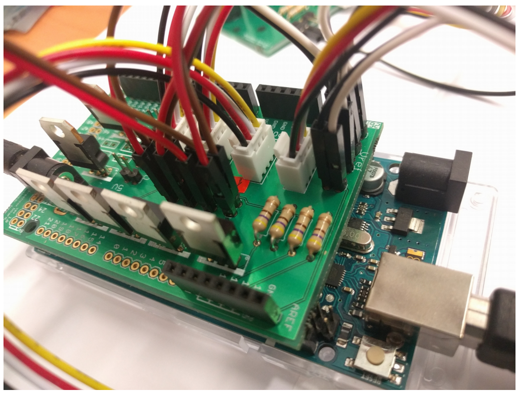
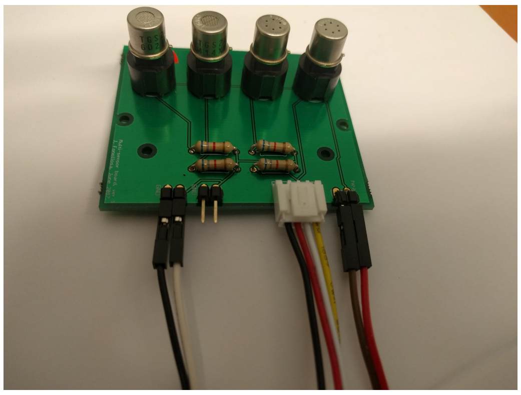
a) Connect the Arduino to the laptop using USB cable. This is used to power the Arduino, sample the sensors, and also to send the signals that drive the sensor heaters. **It should be always connected**. If you are not measuring, you can plug the USB cable to any power outlet (or leave it connected to your PC). If you unplug the USB cable, the sensors will be powered off (which you don't want).

b) Connect the power of the shield to the power source (power outlet). This is from where most of the power is supplied. Make sure to use a source that can deliver, at least 1A at 9V. Typically, the board consumes 700-800mA.

c) The power connector of the Arduino should be always unconnected.

d) I added red stickers with numbers. The are used to id the four boards. A red sticker with 1234 is used to reconnect the cables. The brown-red wires and the four-pin cables should be connected to the board that is indicated in the red label. **Do not connect the wires the other way around (especially the white-black cables)**. That would invert Vcc-Gnd and we could burn the board.

These images can be helpful if you have to reconnect:

Arduino control:

We use *MEGA_16ch_serial_v0.ino*. We can change the code using Arduino IDE.

Default settings are:
A0-A3: PWM 128 (50% @500Hz). Vheater (mean): 3.75V
A4-A7: PWM 150 (62.5% @500Hz). Vheater (mean): 4.39V
A8-A11: PWM 171 (66.8% @500Hz). Vheater (mean): 5.01V
A12-A15: PWM 192 (75% @500Hz). Vheater (mean): 5.62V

Data Acquisition:

You can use any code that communicates over the serial port.
For example, *MEGA_pc_communication_v0.py*, in Python.

If you use the code *MEGA_pc_communication_v0.py*, make sure to specify the correct working directory. Parameters *sampling_time* and *total_time* detail the sampling period and the duration of the experiment, respectively. Change that according to your needs.
Make sure that serial port is the right one (default ttyACM0).

You can run the code by opening a terminal console in the folder where you have the code. Then type:

```
python MEGA_pc_communication_v0.py
```

The code will generate a file with the filename as *date_time*. After the header (first line), data is organized as follows: First column is local (laptop) time, second column is Arduino board time, next 16 columns are sensor readings A0-A15.

A0: TGS2600 @3.75V
A1: TGS2602 @3.75V
A2: TGS2610 @3.75V
A3: TGS2611 @3.75V

A4: TGS2600 @4.39V
A5: TGS2602 @4.39V
A6: TGS2610 @4.39V
A7: TGS2611 @4.39V

A8: TGS2600 @5.01V
A9: TGS2602 @5.01V
A10: TGS2610 @5.01V
A11: TGS2611 @5.01V

A12: TGS2600 @5.62V
A13: TGS2602 @5.62V
A14: TGS2610 @5.62V
A15: TGS2611 @5.62V

Plot Data:

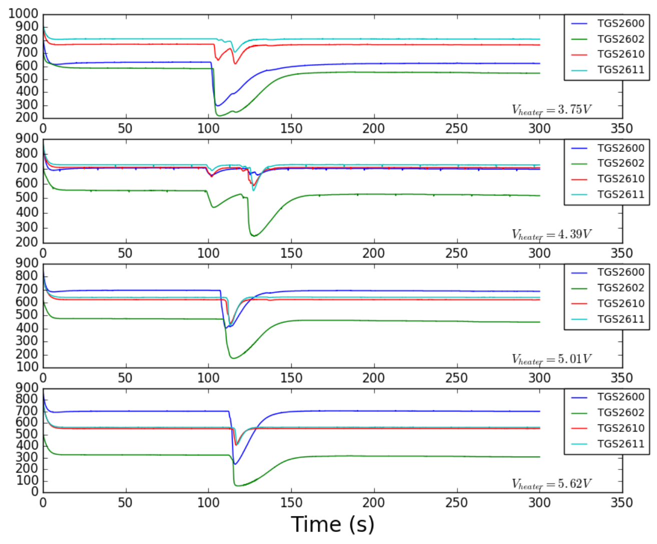
You can use any code to plot the data. I prepared a Python code to plot it easily: *plot_data_v0.py*
I recommend using local (laptop) time. It seems more stable.
The code will generate a figure.

Example of operation:

Locate the port at which the Arduino is connected. (Open terminal and run *dmesg | grep tty*).
Open *MEGA_pc_communication_v0.py* and set your parameters (port, typically /dev/ttyACM0 in Linux). Set the parameters for your measurement (sampling_time=0.1 and total_time=300): 5 minutes at 100ms.
Open your terminal and run *python MEGA_pc_communication_v0.py*
Open the folder and locate your new file.

You can use any code to plot the data. I prepared a Python code to plot it easily: *plot_data_v0.py*

In this example, I started the measurement. At 100s I presented some ethanol to the sensors (non-controlled presentation). After few seconds, I removed the ethanol. Sensors recover the baseline after a minute or so. This figure is what .py code will generate:



Recommendations:

Keep the sensors powered all the time. If the sensors have been unpowered for some time, you'll need to heat them again for some time (this depends on the number of days that the sensors were off). So, when you receive them, power them. You can do tests as soon as you power them, but I'd suggest to wait for a couple of days until you perform your final measurements.

You can discard the first 10 seconds every time you start You can use any code to plot the data. I prepared a Python code to plot it easily: *plot_data_v0.py*