Raspberry Pi Chemistry Project

Worksheet 1

Contacts: Scott Morgan, Dan Baker, Emma Yhnell

####Checking your kit:

We will be developing code and building the structure for a Lego spectrophotometer, which we will use to conduct a simple experiment.

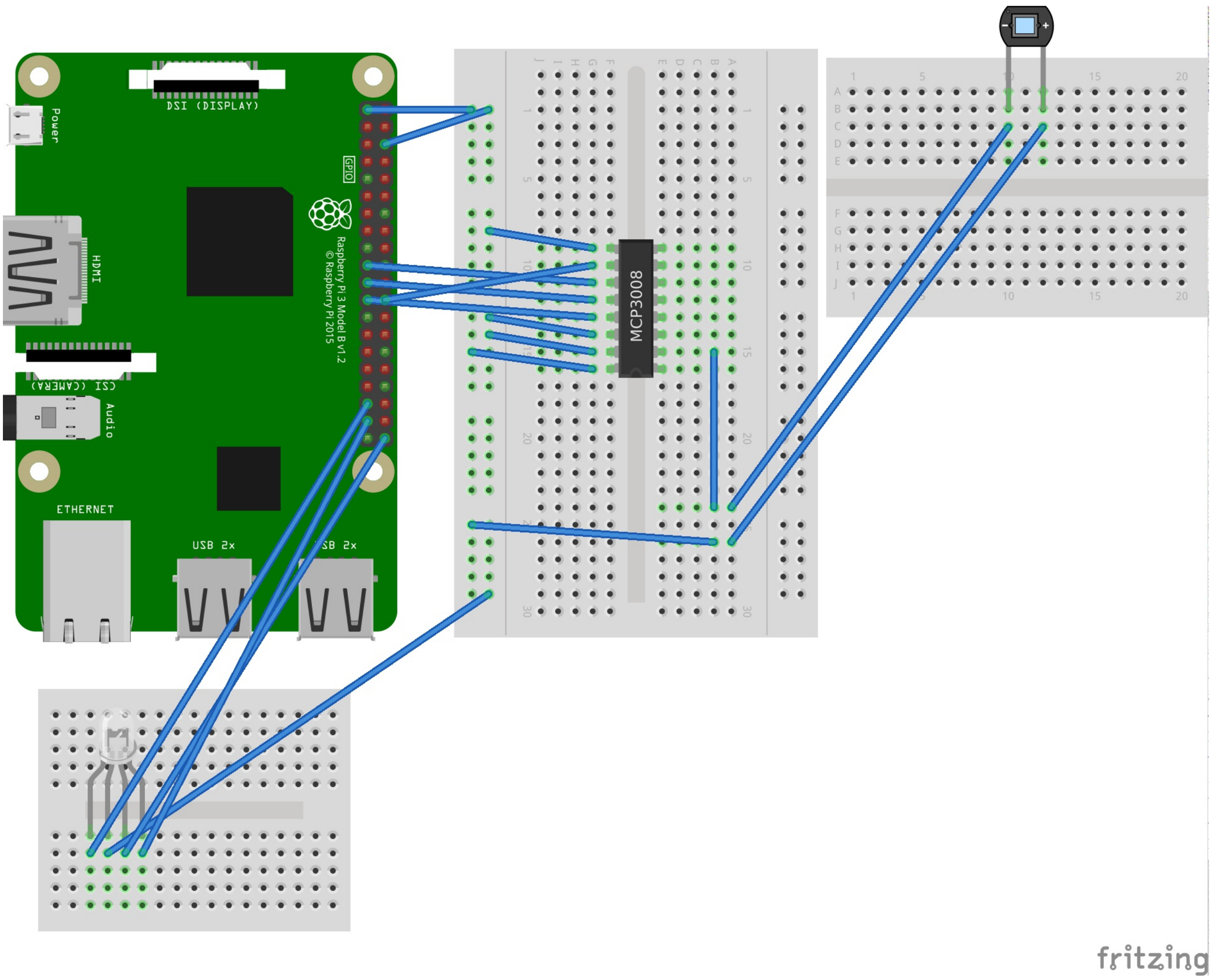
In your pack you should have:

- Raspberry Pi x1
- SD card x1
- HDMI cable x1
- MicroUSB cable x1
- Photodiode x1
- RGB LED x1
- MCP3208 analogue-digital converter x1
- Large breadboard x1
- Mini breadboard x2
- Breadboard jumper wires x18

You should also have access to a **mouse**, **keyboard** and **monitor**.

Assembling the circuit:

Connect your components like the following diagram:



Booting up the Pi

Connect your Raspberry Pi to the power cable, the monitor using the HDMI lead, the mouse and keyboard. Insert the SD card in the rear of the pi.

When the Pi has loaded up, click on the red raspberry in the top left corner and select:

`Programming -> Python3 Idle`

Click:

`File -> New script`

and type the following code.

NB: Python is case-sensitive and requires the correct indentation. You MUST type these exactly.

```
from gpiozero import MCP3208, RGBLED
```

```
led = RGBLED(red=19,green=21,blue=26)
led.color = (0,178/255,1)

adc = MCP3208(channel=1)

while True:
    voltage = 3.3*adc.value
    print(voltage)
```

Press `F5` to run the code. The console will ask you to save the file. Pick an appropriate filename and save the file. You should see a continuous stream of voltages output in the shell. Try covering the photodiode - do they change?

Assembling the Lego:

You will need to create the following setup:

Stage 1:

Stage 2:

Stage 3:

Stage 4:

Assembling the code:

In Python Idle, create a new file and copy the code from the test file into it. Your file should now look like this:

```
from gpiozero import MCP3208, RGBLED

led = RGBLED(red=19,green=21,blue=26)
led.color = (0,178/255,1)

adc = MCP3208(channel=1)

while True:
    voltage = 3.3*adc.value
    print(voltage)
```

We will need to import some new packages to be able to run our experiment. Change the top of the document so that it looks like this:

```
from gpiozero import MCP3208, RGBLED
from time import *
import numpy as np
```

We will also need to define some new variables so we can track an average measurement across time.

Just before `while True:`, add the following:

```
secs = 5

voltarray = []
voltavgarray = []
narray = []
```

Edit the `while` loop so that it looks like this:

```
t_end = time.time() + secs
while time.time() < t_end:
    voltage = 3.3*adc.value
    voltarray.append(voltage)
    print(voltage)
```

Finally, after the loop, add the following:

```
vmean = sum(voltarray)/float(len(voltarray))
```

```
vsd = np.std(voltarray)
rsd = vsd/vmean*100

print(' ')
print(vmean)
print(' ')
print(vsd)
print(' ')
print(rsd)
```

Your final file should look like this:

```
from gpiozero import MCP3208, RGBLED
from time import *
import numpy as np

led = RGBLED(red=19,green=21,blue=26)
led.color = (0,178/255,1)

adc = MCP3208(channel=1)

secs = 5
voltarray = []
voltavgarray = []
narray = []

t_end = time.time() + secs
while time.time() < t_end:
    voltage = 3.3*adc.value
    voltarray.append(voltage)
    print(voltage)

vmean = sum(voltarray)/float(len(voltarray))
vsd = np.std(voltarray)
rsd = vsd/vmean*100

print(' ')
print(vmean)
print(' ')
print(vsd)
print(' ')
print(rsd)
```

Plotting the data

Run your experiment for the four samples given to you.

Write a Python script to plot a graph of the samples you've been given. We will use `matplotlib` to do this. Open a new file and type the following code.

```
import matplotlib.pyplot as plt

xarray = #insert x data in square brackets: e.g. [0,1,2,3,4,5]
yarray = #insert x data in square brackets: e.g. [0.40,0.43,0.41,0.42,0.45]

plt.plot(xarray,yarray,color='k',linewidth=2)
plt.xlabel('Millileters (ml)')
plt.ylabel('Voltage (V)')
plt.xlim(0,max(xarray))
plt.show()
```

Run the final sample and record the data. Edit your plotting file so that it looks like this:

```
import matplotlib.pyplot as plt

xarray = #insert x data in square brackets: e.g. [0,1,2,3,4,5]
yarray = #insert y data in square brackets: e.g. [0.40,0.43,0.41,0.42,0.45]

ynew = ##insert your new y data here, in square brackets: e.g. [0.4014]

plt.plot(xarray,yarray,color='k',linewidth=2)
plt.line([min(xarray),max(xarray)],[ynew,ynew],color='r',linewidth=2)
plt.xlabel('Millileters (ml)')
plt.ylabel('Voltage (V)')
plt.xlim(0,max(xarray))
plt.show()
```

Read off the approximate value of `x` and record it.