# Fuel / Oxidizer Mixing Test Rig[[1]](#footnote-1)

## Some basics concerning working with Raspberry and flow sensor

### Step 1 reading book about Raspberry and C++

Reading a book about raspberry pi and how to write code in C ++

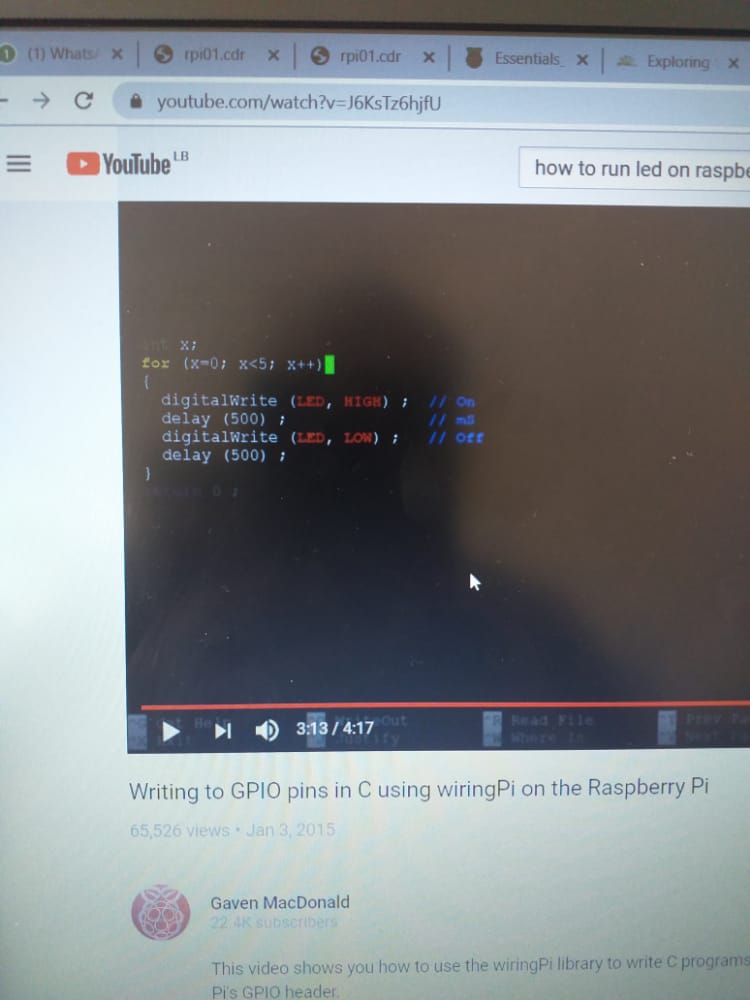
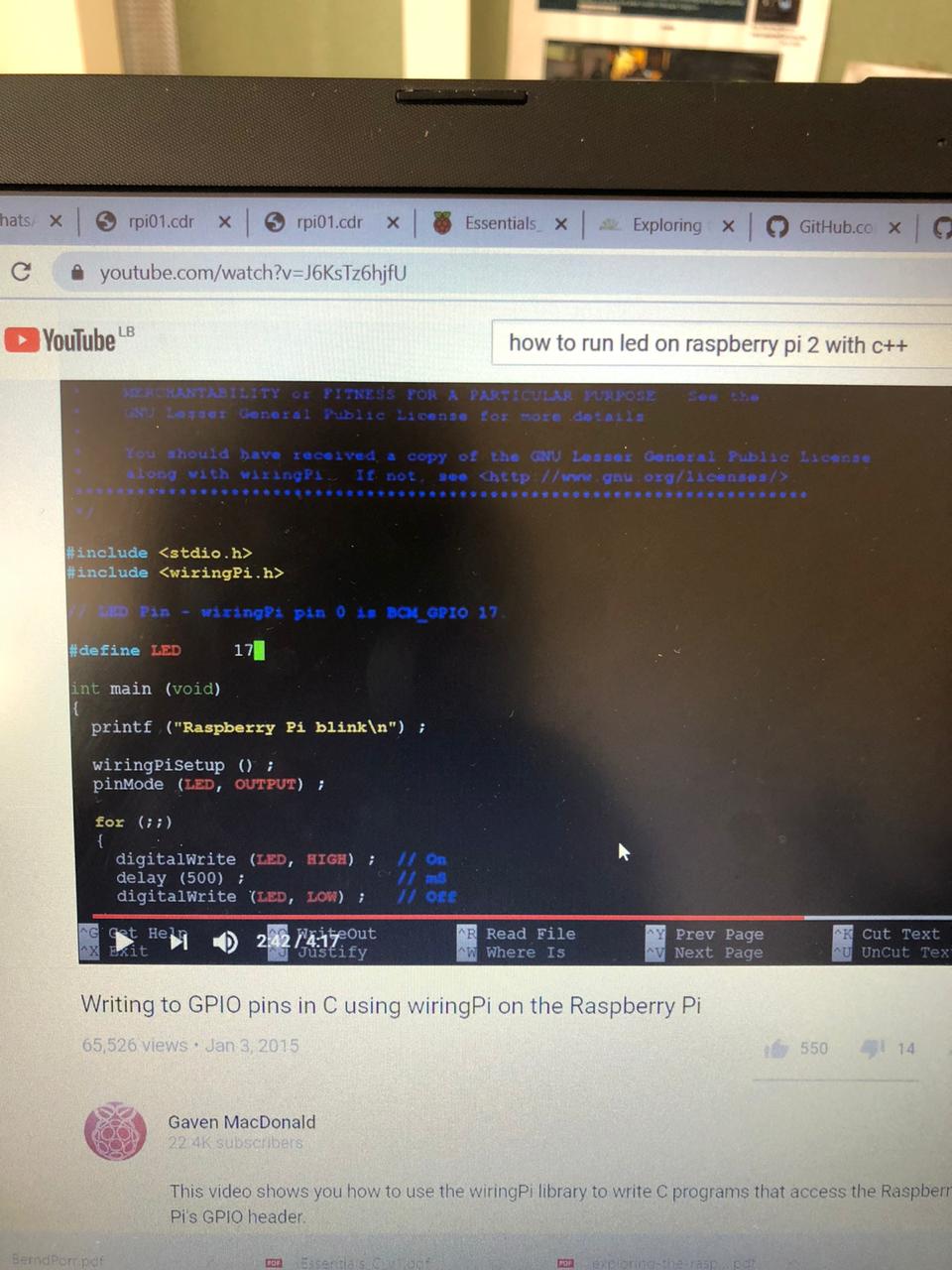
The link of this book:

<https://bbooks.info/b/w/8873f497932991f0a46529d6b98eea3373f9ad81/exploring-the-raspberry-pi-2-with-c.pdf>

### Step 2 Watch on YouTube a video related to LED control

The title of this video: writing to GPIO pins in C using wiringPi on the raspberry Pi

### Step 3 Writing C++ code on Raspberry pi



### Step 4 Hardware integration

We installed a resistor with LED on breadboard and connected it on Raspberry PI



### Step 5 Running

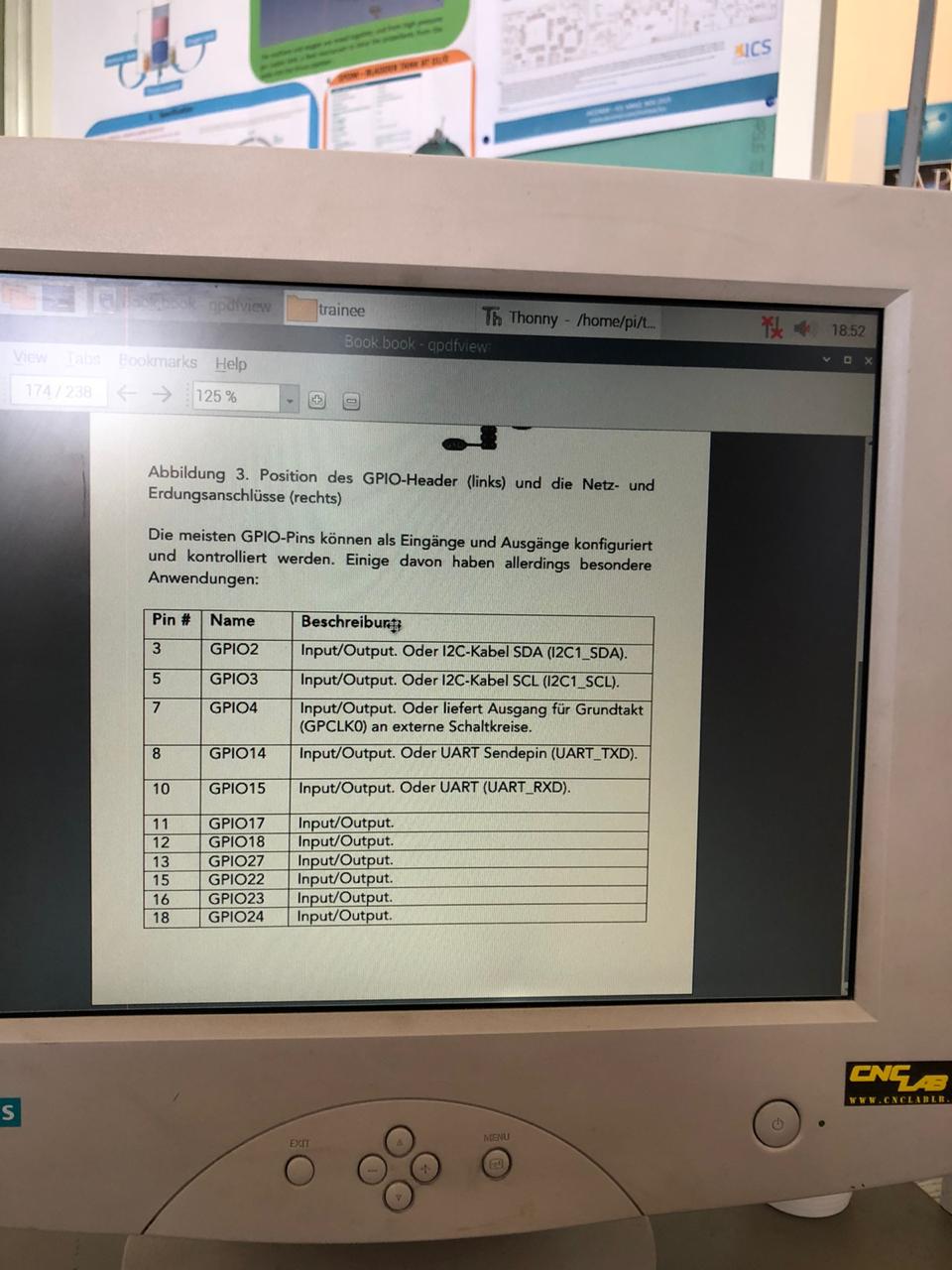
Some pictures while working and controlling the LED:

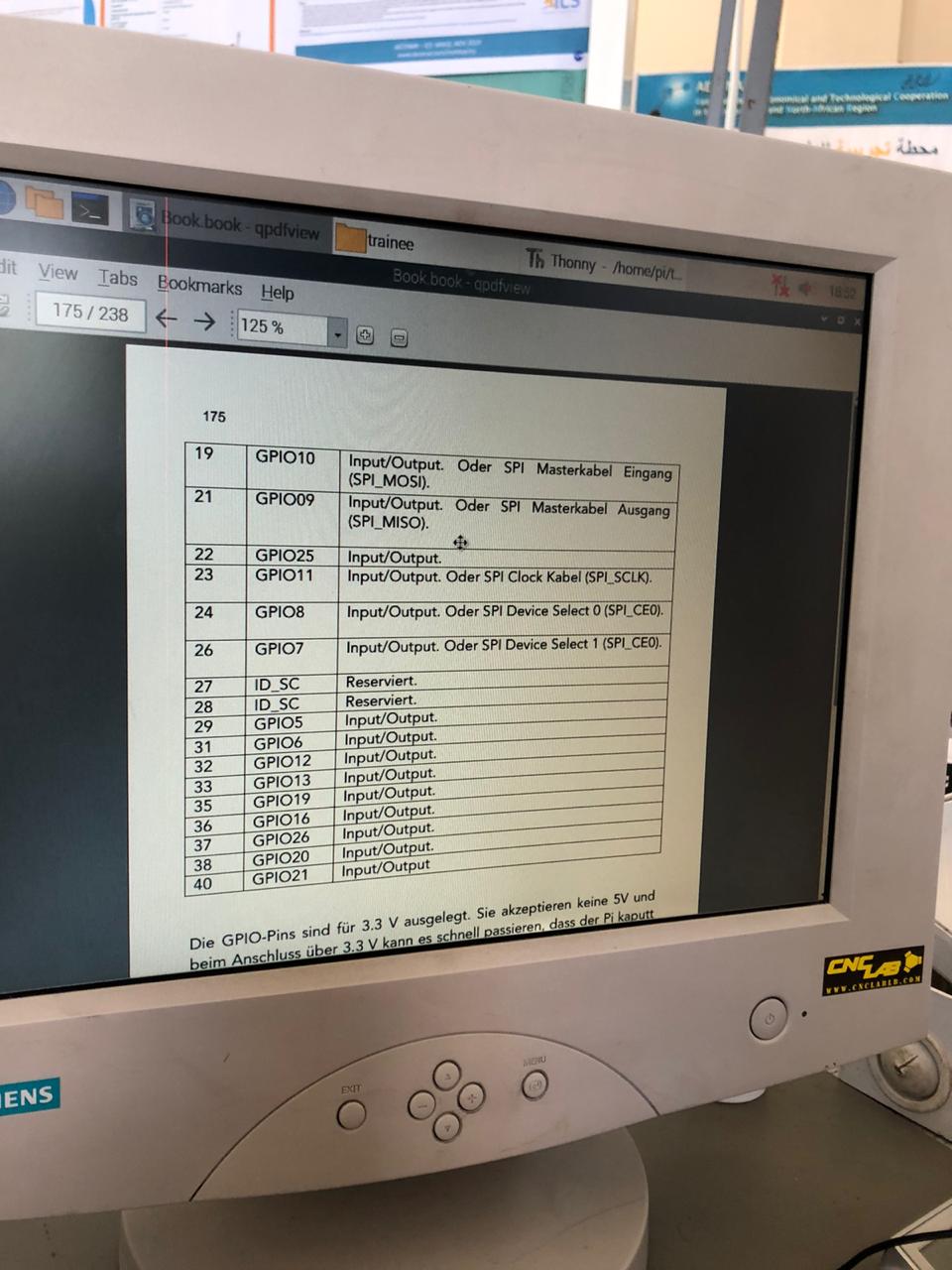


### Flowmeter buyed from CNCLab



### Raspberry GPIO pins



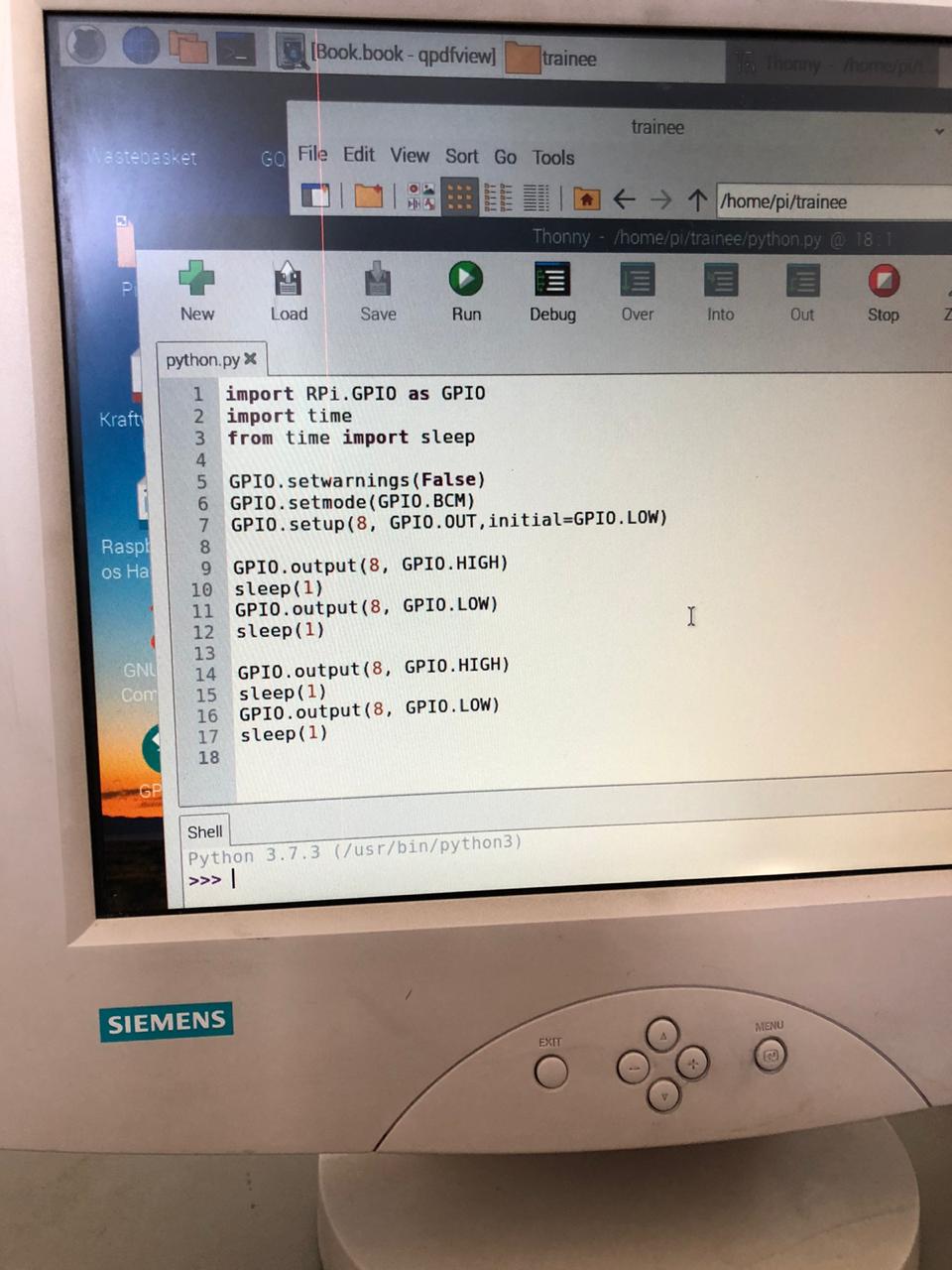


## Some basics connerning Python programming with Raspberry

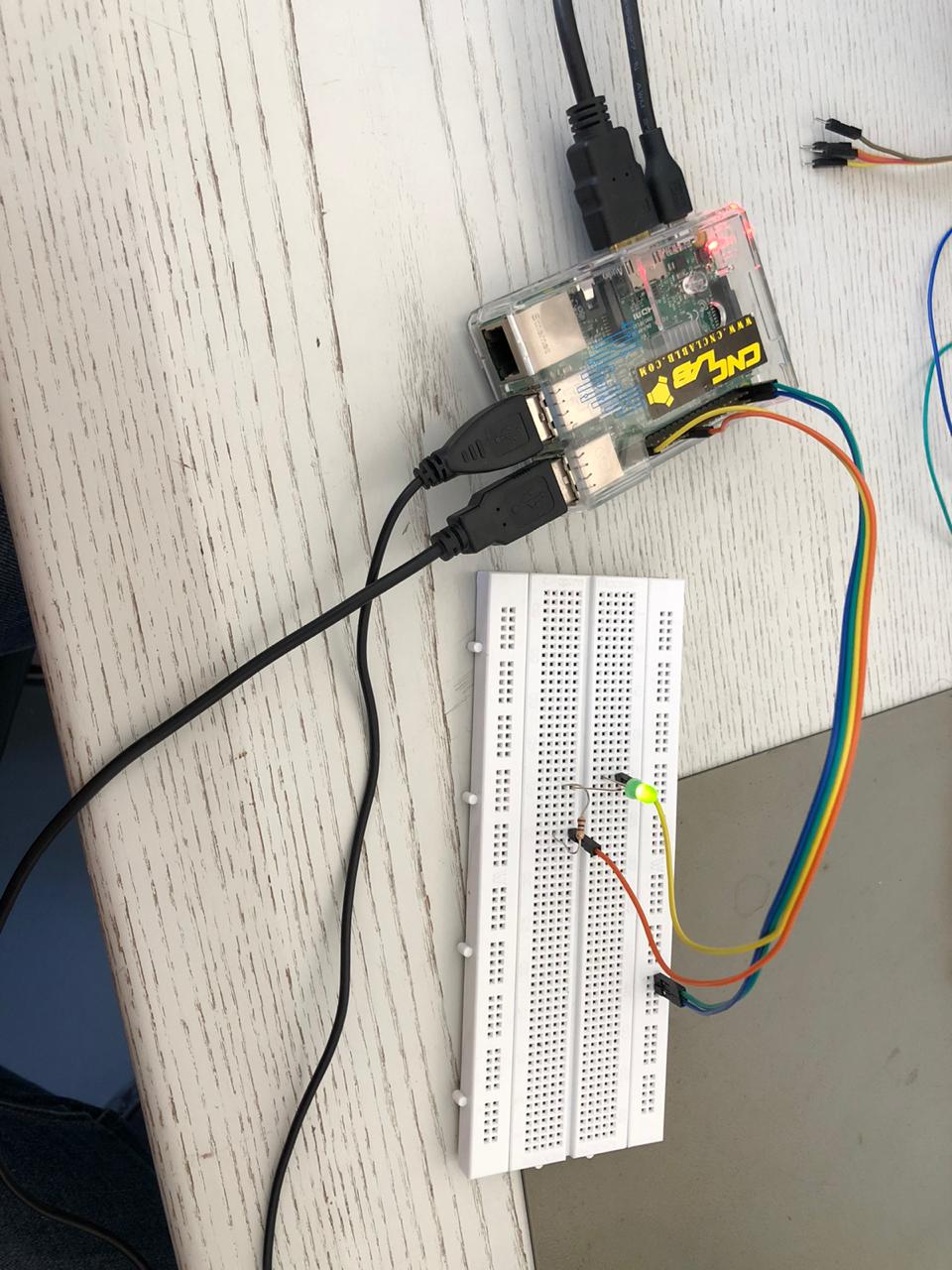
### Step 1: write code in new script to control led

لأنه الكود الاسهل ولقد تم نجاح هذا الاختبار والكود موجود بالأسفلLED قمنا باختبار Python codeكوننا جدد على

Attention: using this raspberry pi you must focus to GPIO.setmode(GPIO.BCM) NOT .BOARD



### Step 2: Hardware



### Testing code for flow meter

The flow sensorاليوم كتبنا كود لنقوم بتشغيل

#!/usr/bin/env python

#flowsensor.py

import RPi.GPIO as GPIO

import time, sys

FLOW\_SENSOR = 23

GPIO.setmode(GPIO.BCM)

GPIO.setup(FLOW\_SENSOR, GPIO.IN, pull\_up\_down = GPIO.PUD\_UP)

#global count

#count = 0

def countpulse(channel,start\_counter):

print ("Bismillah")

print (channel)

# global count

start\_counter = 1

count = 0

if start\_counter == 1:

count += 1

#print count

#flow = count / (60 \* 7.5)

#print(flow)

channel= GPIO.add\_event\_detect(FLOW\_SENSOR, GPIO.FALLING, callback=countpulse)

while True:

try:

start=count - 1

time.sleep(1)

start=count - 1

flow = (count \* 60 \* 2.25 / 1000)

print ("The flow is:" , flow)

#%.3f Liter/min' % (flow)

count = 0

time.sleep(5)

except KeyboardInterrupt:

print ('\ncaught keyboard interrupt!, bye')

print ("No flow")

GPIO.cleanup()

sys.exit()

countpulse (1,2)

New test of flow meter sensor

import RPi.GPIO as GPIO

import time, sys

pulse\_pin = 25

GPIO.setmode(GPIO.BCM)

GPIO.setup(pulse\_pin, GPIO.IN,pull\_up\_down = GPIO.PUD\_UP)

def countPulse1(channel1):

count+=1

print("Number of revolution of wheel of flow sensor:")

print(count)

GPIO.add\_event\_detect(pulse\_pin, GPIO.RISING, callback=countPulse1)

try:

while True:

print("Inside while starting")

time.sleep(10)

print("Inside while ending")

time.sleep(10)

except keyboardInterrupt:

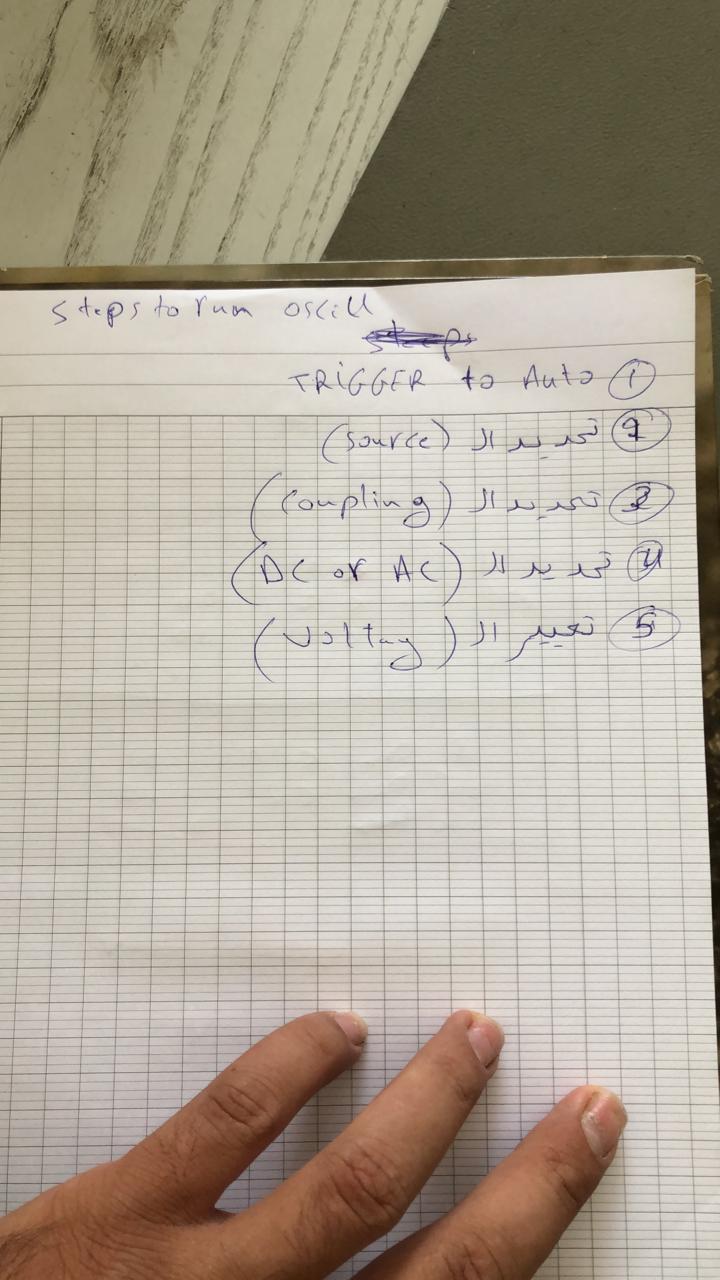
print ('\ncaught keyboard interrupt!, bye')

GPIO.cleanup()

sys.exit()

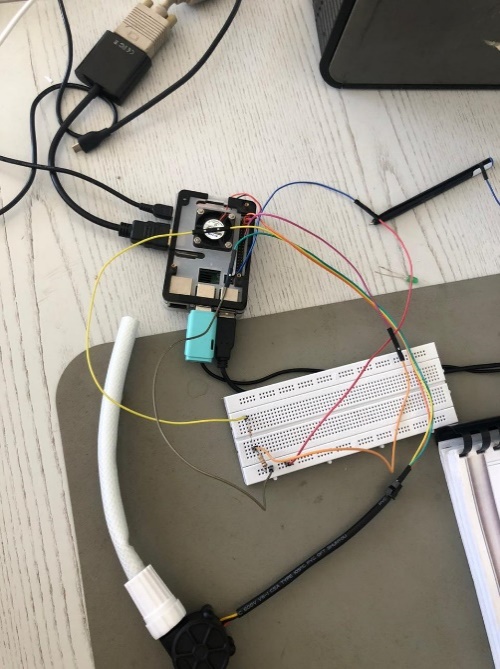
* Today, we're going to modify the code to suit our work with a flow meter sensor
* We will also connect the sensor to the raspberry pi and we will try to reach a result
* We'll also connect the valve to the raspberry pi and we'll control it
* We will write all the writing work on the word

## Working with oscilloscope



## Python program for flow sensor

### Hardware for flowmeter sensor:

### The code of flow meter sensor:

import RPi.GPIO as GPIO

import time, sys

FLOW\_SENSOR = 23

GPIO.setmode(GPIO.BCM)

GPIO.setup(FLOW\_SENSOR, GPIO.IN, pull\_up\_down = GPIO.PUD\_UP)

global count

count = 0

def countPulse(channel):

global count

if start\_counter == 1:

count = count+1

# print count

# flow = count / (60 \* 7.5)

# print(flow)

GPIO.add\_event\_detect(FLOW\_SENSOR, GPIO.FALLING, callback=countPulse)

while True:

try:

start\_counter = 1

time.sleep(1)

start\_counter = 0

flow = (count \* 60 \* 2.25 / 1000)

print "The flow is: %.3f Liter/min" % (flow)

count = 0

time.sleep(5)

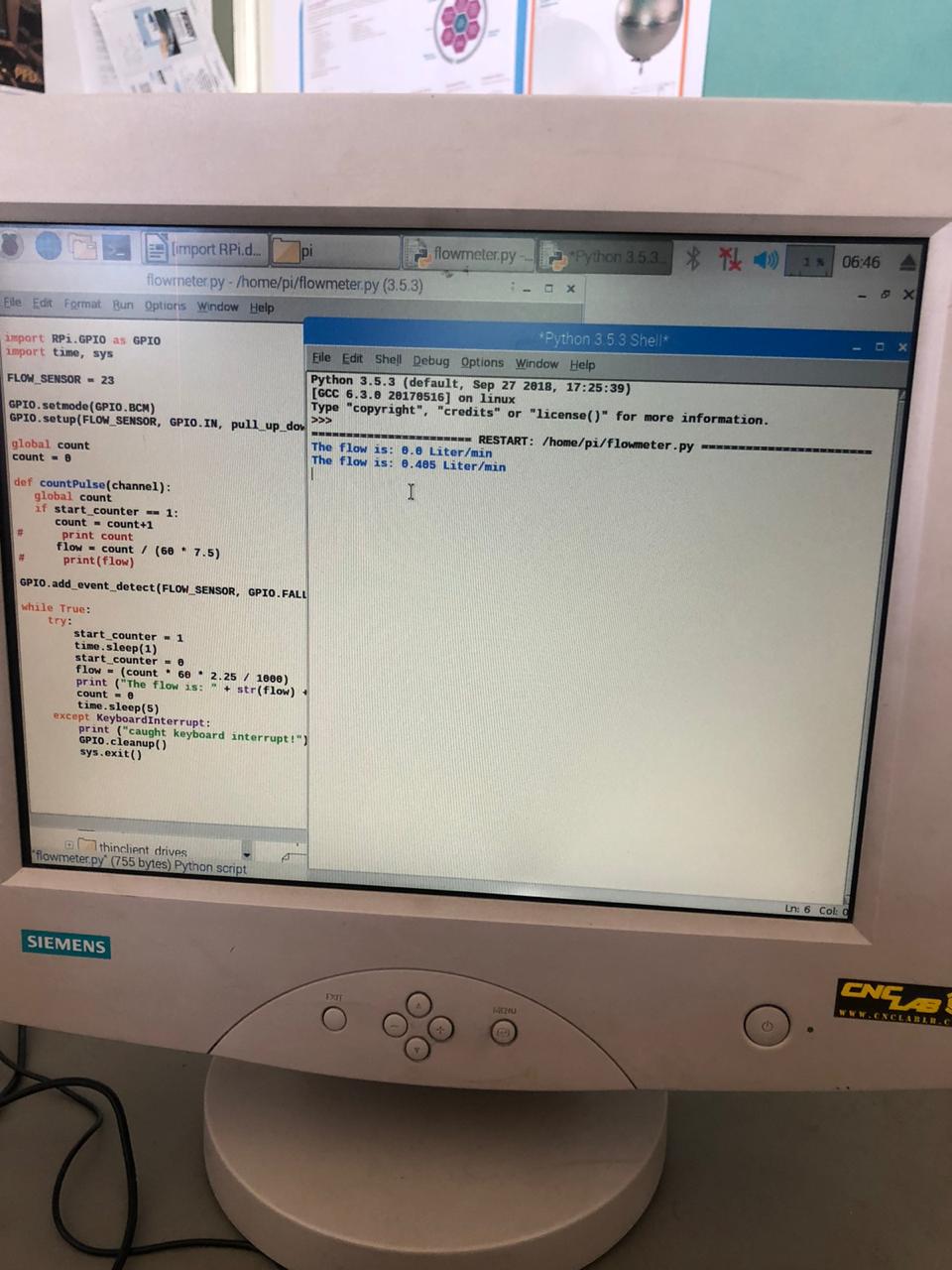
except KeyboardInterrupt:

print '\ncaught keyboard interrupt!, bye'

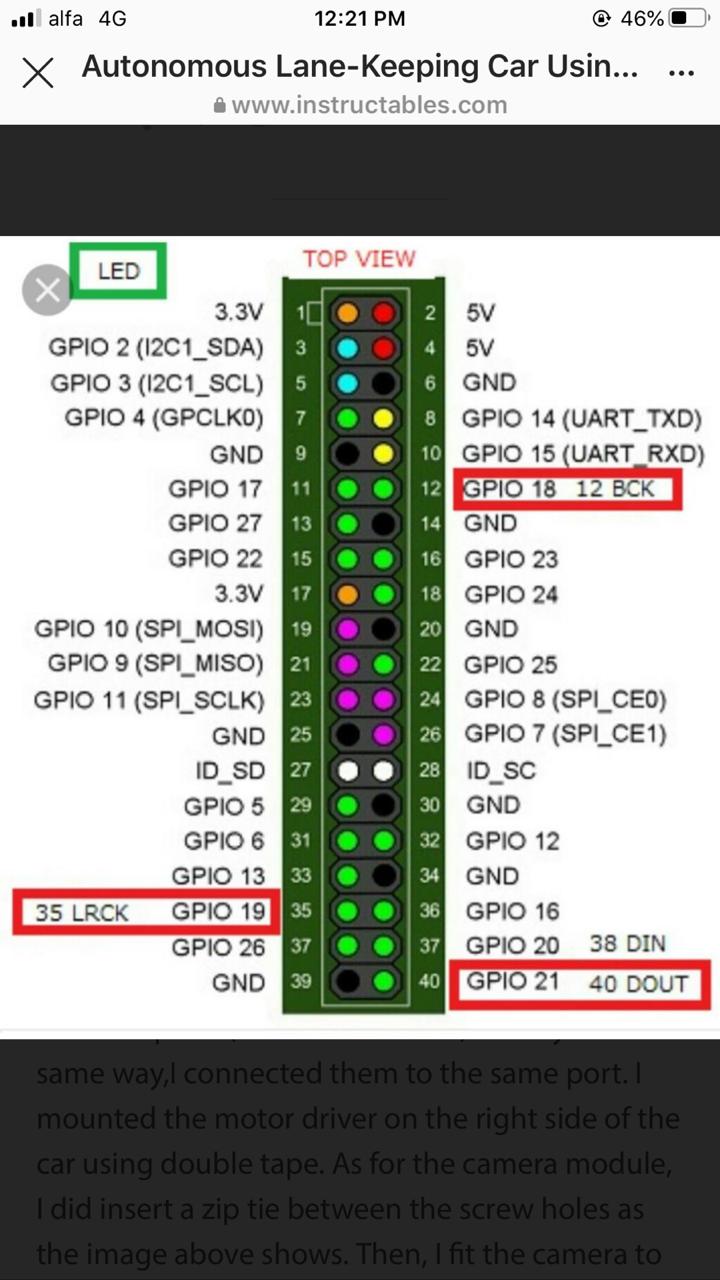
GPIO.cleanup()

sys.exit()

### Running test result



### Pins of raspberry pi 3



## Python code for servo motor (automatic valve, variant 1)

### The code for servor motor

# Import libraries

import RPi.GPIO as GPIO

import time

# Set GPIO numbering mode

GPIO.setmode(GPIO.BOARD)

# Set pin 11 as an output, and set servo1 as pin 11 as PWM

GPIO.setup(11,GPIO.OUT)

servo1 = GPIO.PWM(11,50) # Note 11 is pin, 50 = 50Hz pulse

#start PWM running, but with value of 0 (pulse off)

servo1.start(0)

print ("Waiting for 2 seconds")

time.sleep(2)

#Let's move the servo!

print ("Rotating 180 degrees in 10 steps")

# Define variable duty

duty = 2

# Loop for duty values from 2 to 12 (0 to 180 degrees)

while duty <= 12:

servo1.ChangeDutyCycle(duty)

time.sleep(1)

duty = duty + 1

# Wait a couple of seconds

time.sleep(2)

# Turn back to 90 degrees

print ("Turning back to 90 degrees for 2 seconds")

servo1.ChangeDutyCycle(7)

time.sleep(2)

#turn back to 0 degrees

print ("Turning back to 0 degrees")

servo1.ChangeDutyCycle(2)

time.sleep(0.5)

servo1.ChangeDutyCycle(0)

#Clean things up at the end

servo1.stop()

GPIO.cleanup()

print ("Goodbye")

### Hardware

|  |  |  |
| --- | --- | --- |
|  |  |  |



## The code for the servo motor and flow sensor:

# Import libraries

import RPi.GPIO as GPIO

import time, sys

FLOW\_SENSOR = 23

SERVO = 11

# Set GPIO numbering mode

GPIO.setmode(GPIO.BCM)

# Set pin 23 as input for the flow sensor

# Set pin 11 as an output, and set servo1 as pin 11 as PWM

GPIO.setup(FLOW\_SENSOR, GPIO.IN, pull\_up\_down = GPIO.PUD\_UP)

GPIO.setup(SERVO, GPIO.OUT)

global count

count = 0

start\_counter = 0

def countPulse(channel):

global count

if start\_counter == 1:

count = count+1

# print count

flow = count / (60 \* 7.5)

# print(flow)

servo1 = GPIO.PWM(11,50) # Note 11 is pin, 50 = 50Hz pulse

GPIO.add\_event\_detect(FLOW\_SENSOR, GPIO.FALLING, callback=countPulse)

#start PWM running, but with value of 0 (pulse off)

servo1.start(0)

print ("Waiting for 2 seconds")

time.sleep(2)

#Let's move the servo!

print ("Rotating 180 degrees in 10 steps")

# Define variable duty

duty = 2

# Loop for duty values from 2 to 12 (0 to 180 degrees)

while duty <= 12:

servo1.ChangeDutyCycle(duty)

time.sleep(1)

duty = duty + 1

# Wait a couple of seconds

time.sleep(2)

# Turn back to 90 degrees

print ("Turning back to 90 degrees for 2 seconds")

servo1.ChangeDutyCycle(7)

time.sleep(2)

#turn back to 0 degrees

print ("Turning back to 0 degrees")

servo1.ChangeDutyCycle(2)

time.sleep(0.5)

servo1.ChangeDutyCycle(0)

time.sleep(2)

while True:

try:

start\_counter = 1

time.sleep(1)

start\_counter = 0

flow = (count \* 60 \* 2.25 / 1000)

print ("The flow is: " + str(flow) + " Liter/min" )

count = 0

time.sleep(2)

except KeyboardInterrupt:

servo1.stop()

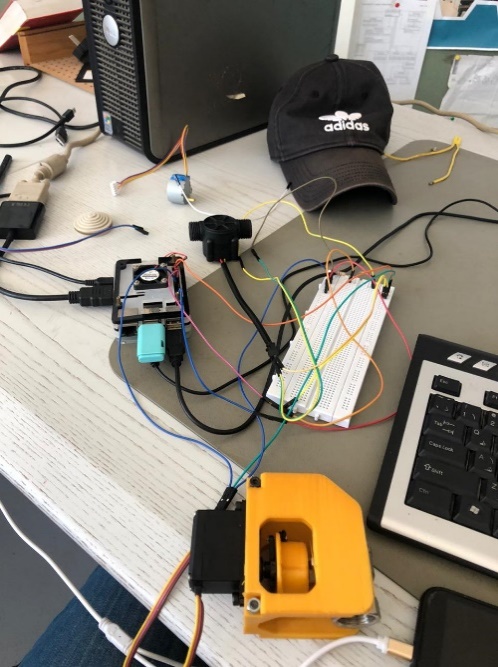
print ("caught keyboard interrupt!")

GPIO.cleanup()

print ("Goodbye")

sys.exit()

### Hardware of servo motor and flow sensor:

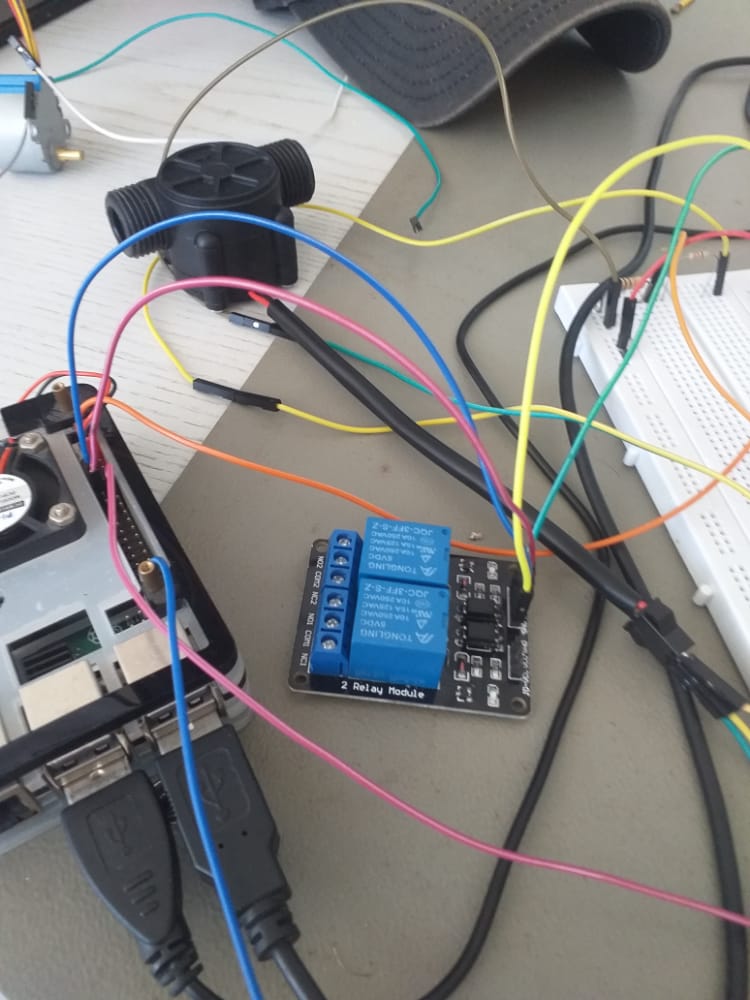


### DC motor for automatic valve (on-off of a relay) (automatic valve, variant 2)

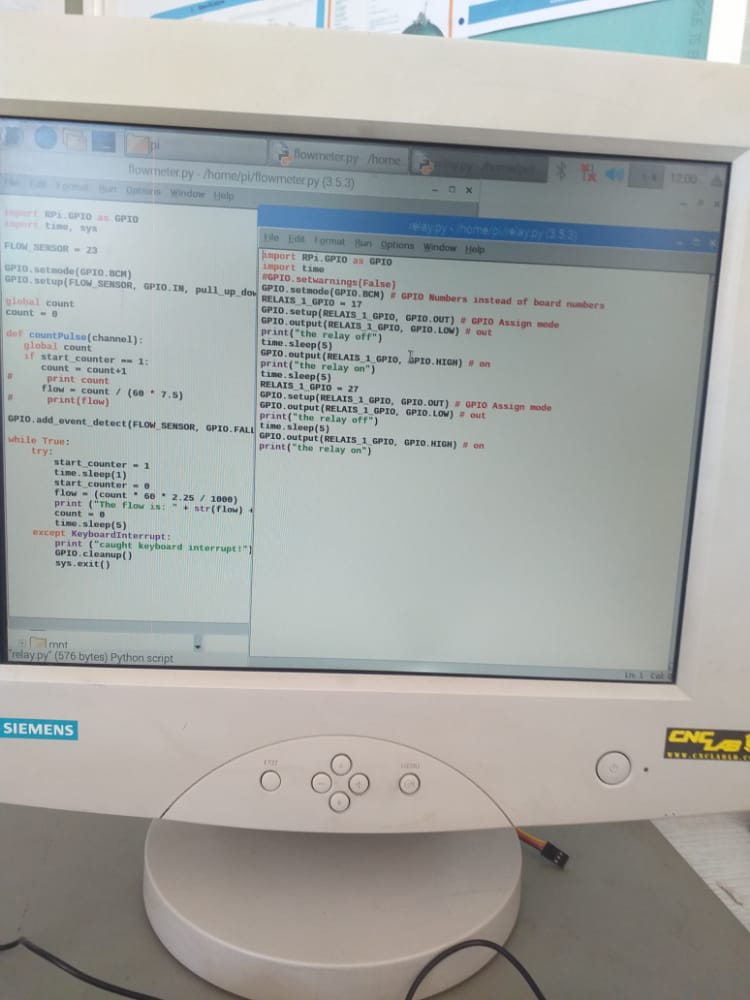
### References of control DC motor:

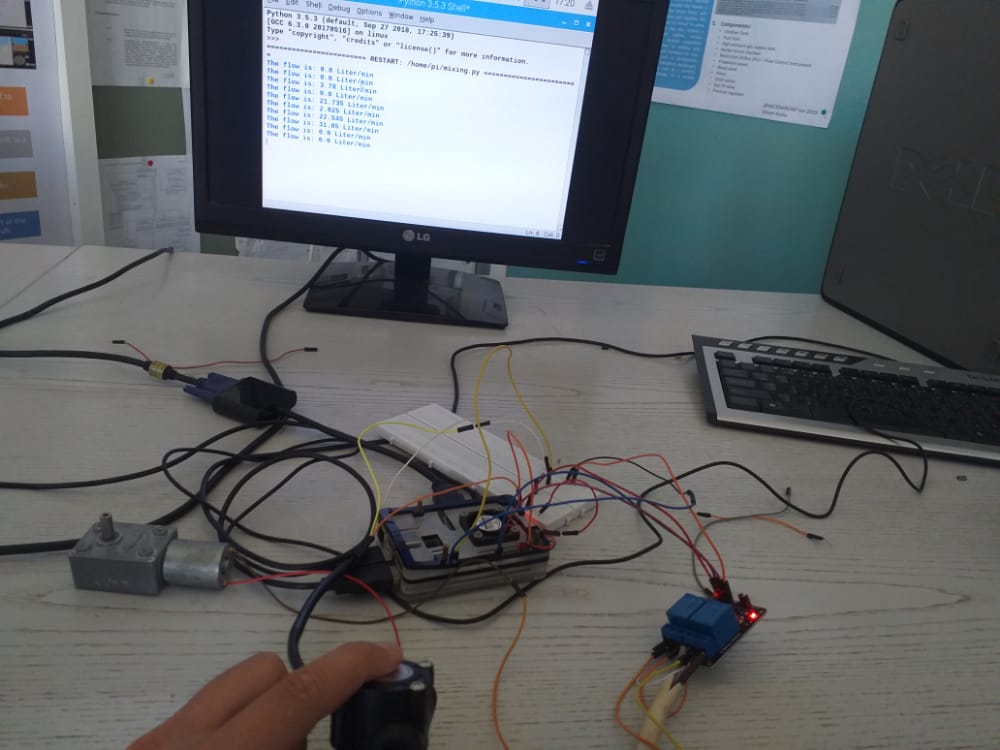
<https://www.aranacorp.com/en/control-a-dc-motor-with-raspberry-pi/>

### Hardware

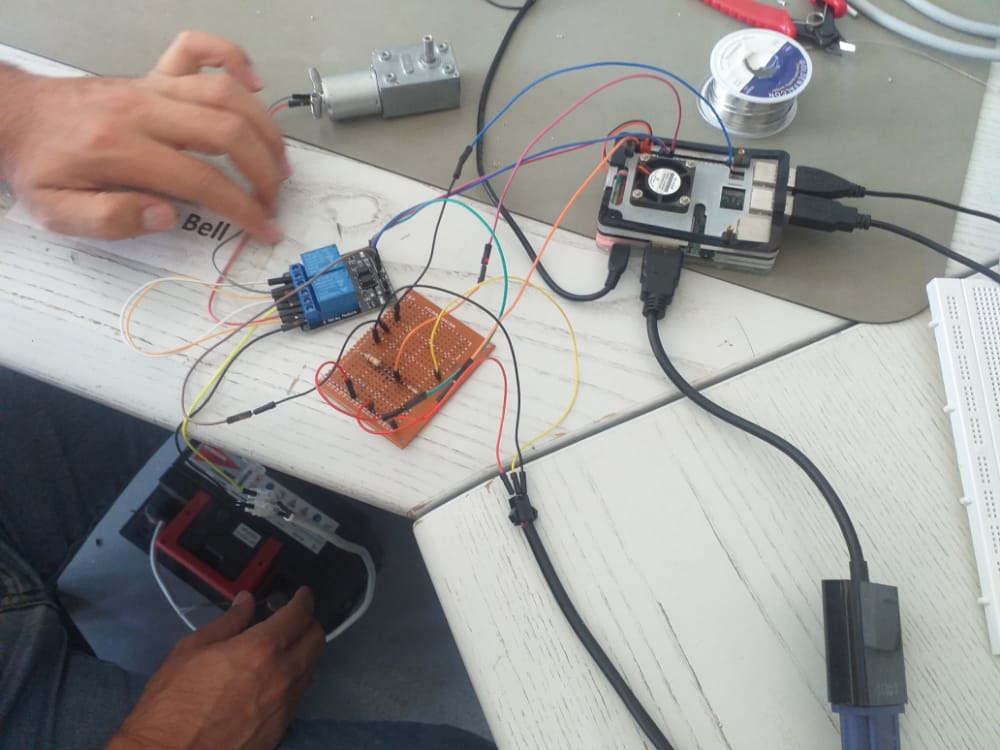
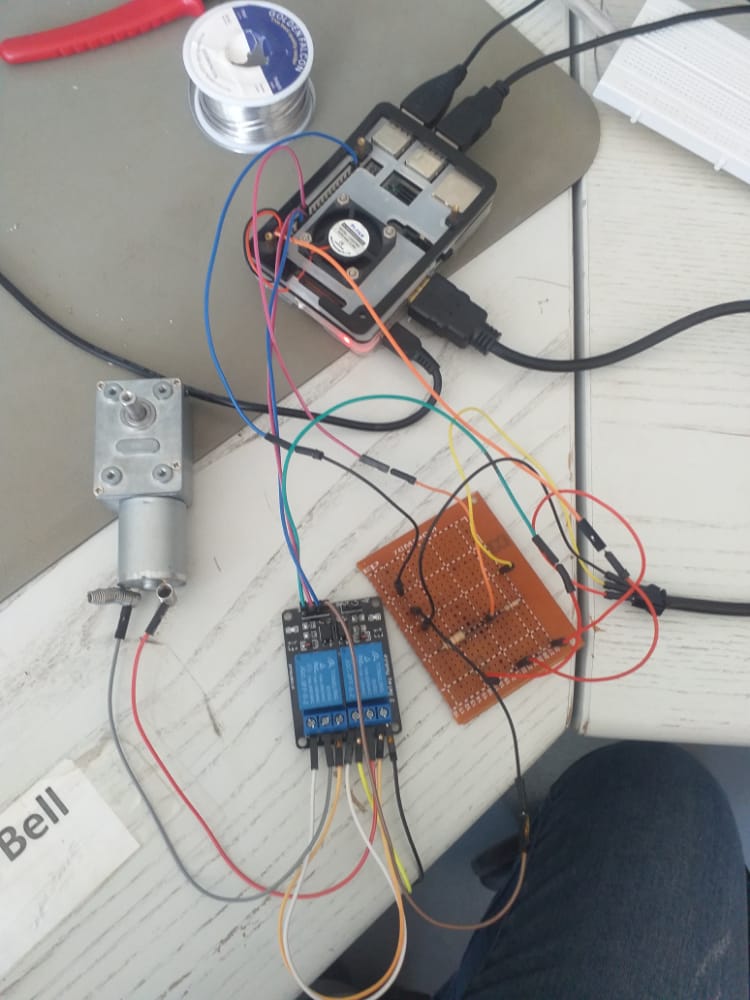
 

### The code

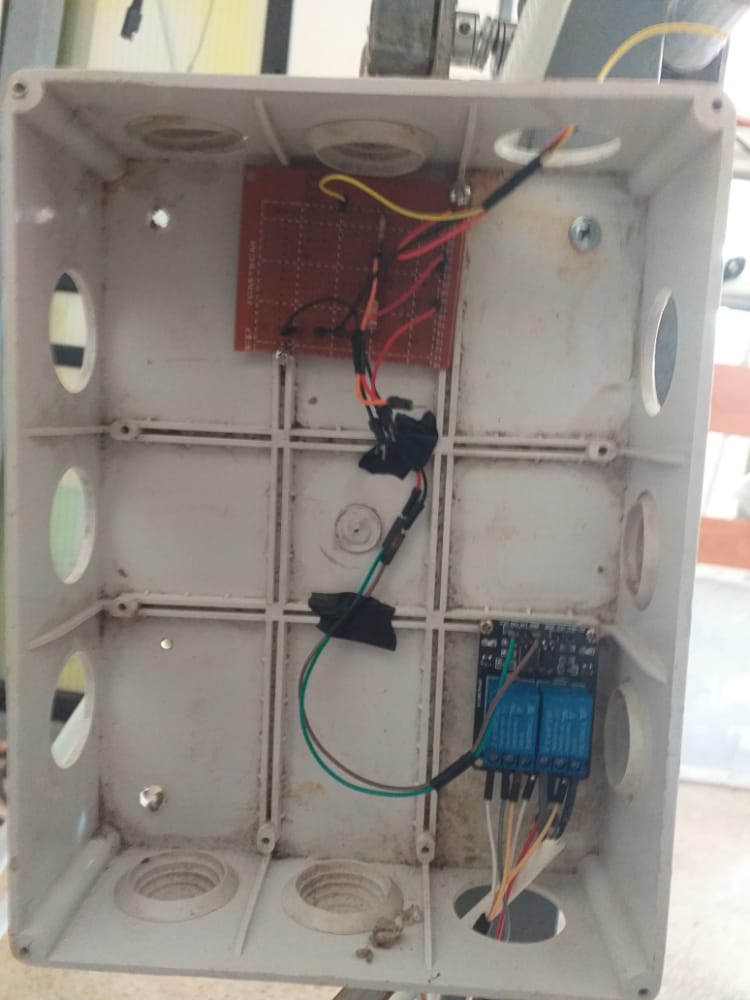




### Testing

## System integration Mixing Control (Mechanical, Hardware & Software)

### Code

import RPi.GPIO as GPIO

import time, sys

RELAY\_2\_GPIO = 27

FLOW\_SENSOR = 23

RELAY\_1\_GPIO = 17

GPIO.setmode(GPIO.BCM)

GPIO.setup(FLOW\_SENSOR, GPIO.IN, pull\_up\_down = GPIO.PUD\_UP)

GPIO.setup(RELAY\_1\_GPIO, GPIO.OUT) # GPIO Assign mode

GPIO.setup(RELAY\_2\_GPIO, GPIO.OUT)

GPIO.output(RELAY\_1\_GPIO, GPIO.LOW)

GPIO.output(RELAY\_2\_GPIO, GPIO.LOW)

global count

count = 0

def countPulse(channel):

global count

if start\_counter == 1:

count = count+1

# print count

flow = count / (60 \* 7.5)

# print(flow)

GPIO.add\_event\_detect(FLOW\_SENSOR, GPIO.FALLING, callback=countPulse)

while True:

try:

start\_counter = 1

time.sleep(1)

start\_counter = 0

flow = (count \* 60 \* 2.25 / 1000)

print ("The flow is: " + str(flow) + " Liter/min" )

if flow == 0: # 0.0

GPIO.output(RELAY\_1\_GPIO, GPIO.LOW)

GPIO.output(RELAY\_2\_GPIO, GPIO.LOW)

elif flow < 15: # 1.9

GPIO.output(RELAY\_1\_GPIO, GPIO.LOW)

GPIO.output(RELAY\_2\_GPIO, GPIO.HIGH)

elif flow < 20: # 2.1

GPIO.output(RELAY\_1\_GPIO, GPIO.LOW)

GPIO.output(RELAY\_2\_GPIO, GPIO.LOW)

else:

GPIO.output(RELAY\_1\_GPIO, GPIO.HIGH)

GPIO.output(RELAY\_2\_GPIO, GPIO.LOW)

count = 0

time.sleep(5)

except KeyboardInterrupt:

print ("caught keyboard interrupt!")

GPIO.cleanup()

sys.exit()

## Mixing control

نحن نريد التحكم بنوعين من السوائل لذلك هذه الخطوة ستكون مشابهة للخطوة الأولى فهي عبارة عن تحكم بكمية السائل الذي نحتاجه

سنحتاج بهذا العمل الى

* 2 relays for DC motors (automatic valves, variant 2)
* Raspberry pi
* Battery 12v
* Wires
* Flow sensor
* 



### The code

import RPi.GPIO as GPIO

import time, sys

RELAY\_2\_GPIO = 27

FLOW\_SENSOR = 23

RELAY\_1\_GPIO = 17

GPIO.setmode(GPIO.BCM)

GPIO.setup(FLOW\_SENSOR, GPIO.IN, pull\_up\_down = GPIO.PUD\_UP)

GPIO.setup(RELAY\_1\_GPIO, GPIO.OUT) # GPIO Assign mode

GPIO.setup(RELAY\_2\_GPIO, GPIO.OUT)

GPIO.output(RELAY\_1\_GPIO, GPIO.LOW)

GPIO.output(RELAY\_2\_GPIO, GPIO.LOW)

global count

count = 0

def countPulse(channel):

global count

if start\_counter == 1:

count = count+1

# print count

flow = count / (60 \* 7.5)

# print(flow)

GPIO.add\_event\_detect(FLOW\_SENSOR, GPIO.FALLING, callback=countPulse)

while True:

try:

start\_counter = 1

time.sleep(1)

start\_counter = 0

flow = (count \* 60 \* 2.25 / 1000)

print ("The flow is: " + str(flow) + " Liter/min" )

if flow == 0: # 0.0

GPIO.output(RELAY\_1\_GPIO, GPIO.LOW)

GPIO.output(RELAY\_2\_GPIO, GPIO.LOW)

elif flow < 15: # 1.9

GPIO.output(RELAY\_1\_GPIO, GPIO.LOW)

GPIO.output(RELAY\_2\_GPIO, GPIO.HIGH)

elif flow < 20: # 2.1

GPIO.output(RELAY\_1\_GPIO, GPIO.LOW)

GPIO.output(RELAY\_2\_GPIO, GPIO.LOW)

else:

GPIO.output(RELAY\_1\_GPIO, GPIO.HIGH)

GPIO.output(RELAY\_2\_GPIO, GPIO.LOW)

count = 0

time.sleep(5)

except KeyboardInterrupt:

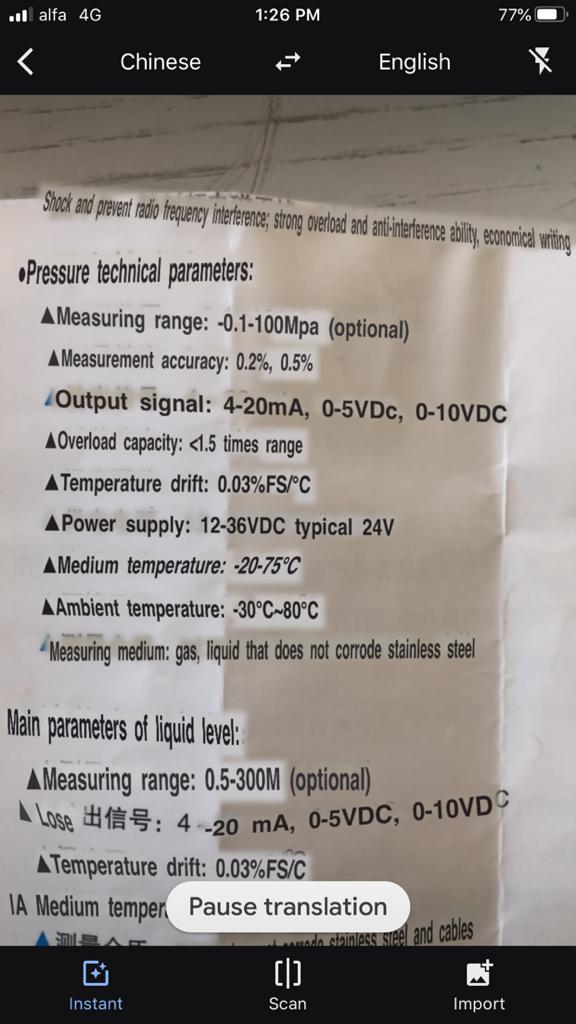
print ("caught keyboard interrupt!")

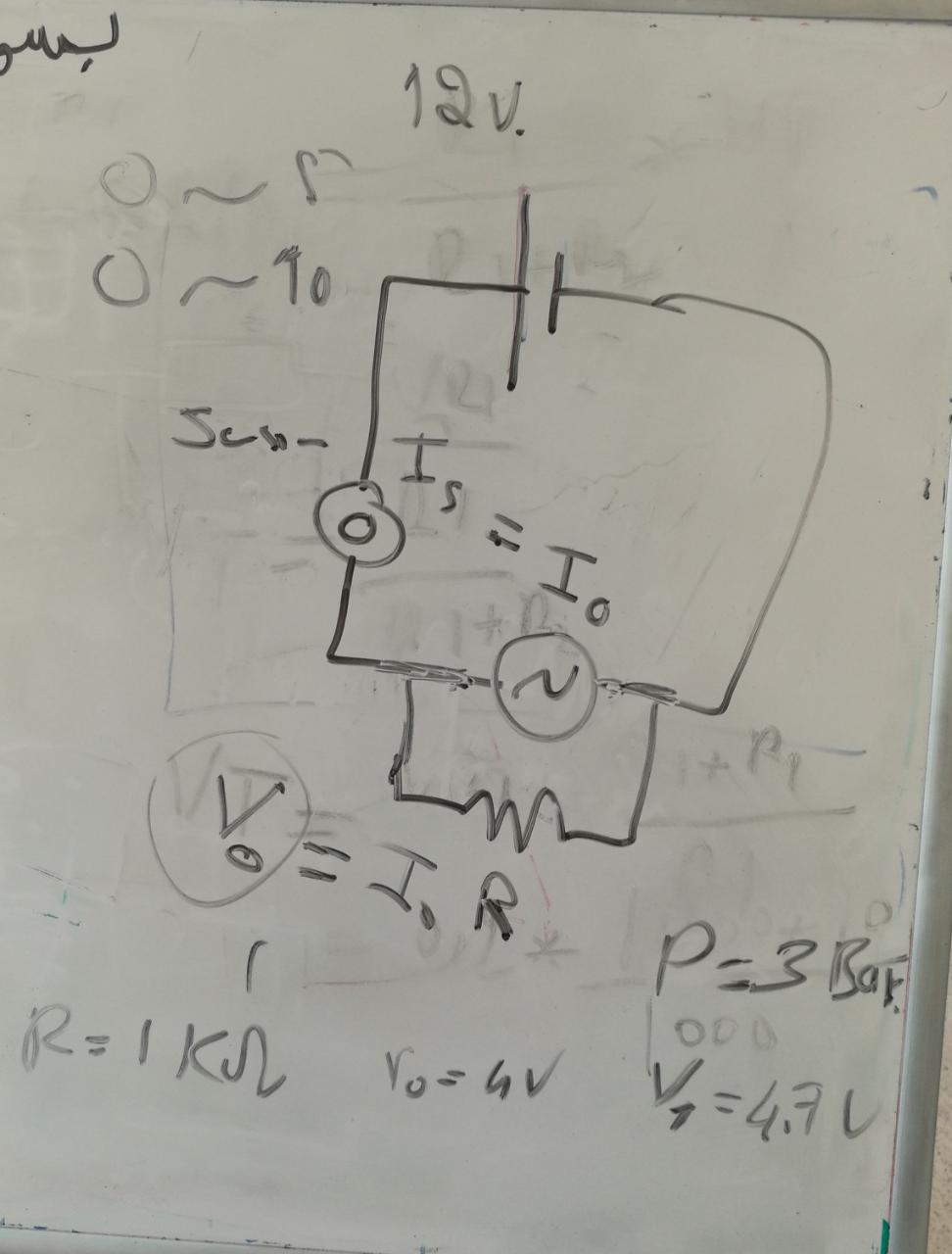
GPIO.cleanup()

sys.exit()

## Pressure Sensor

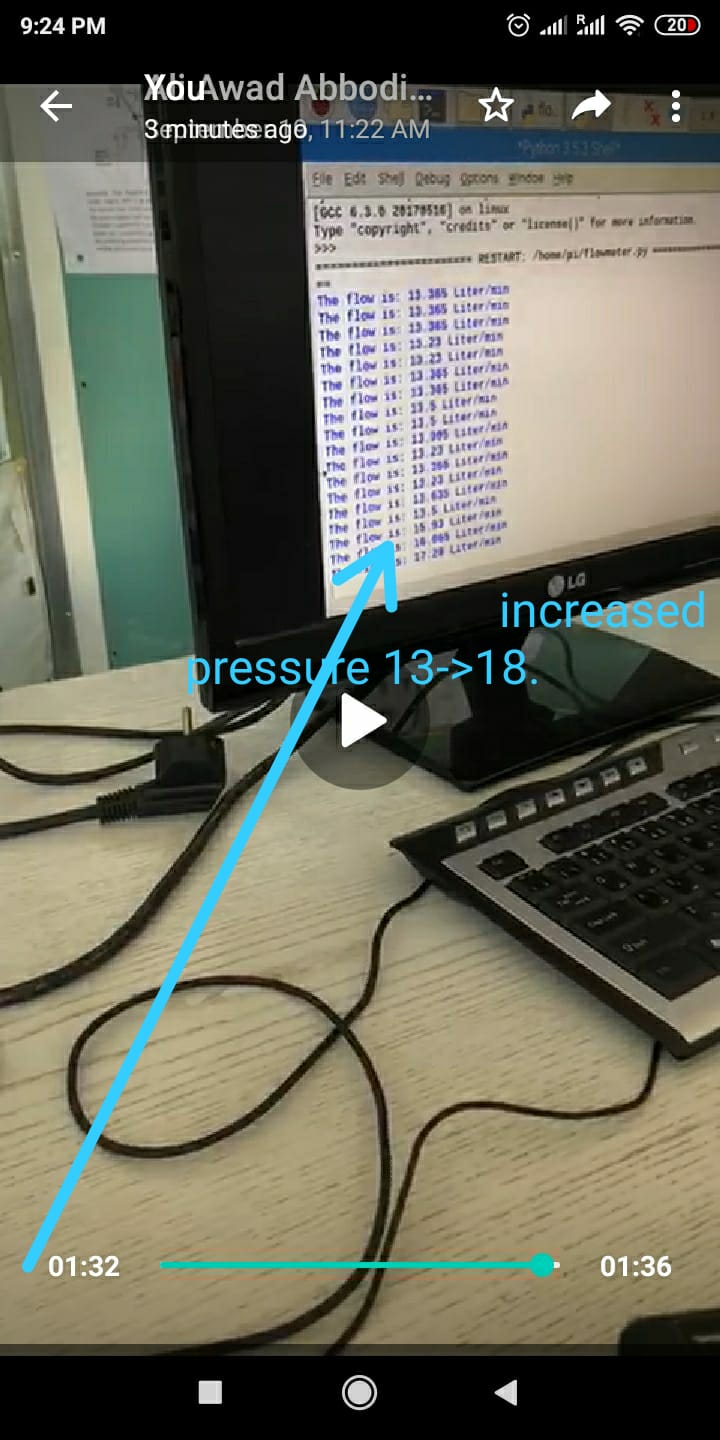






|  |  |
| --- | --- |
|  | High pressure |
|  |  |
|  | Low pressure |





### Code for pressure sensor



#### Pressure

import RPi.GPIO as GPIO

import time, sys

FLOW\_SENSOR = 26

lps = adafruit\_lps35hw.LPS35HW(i2c)

GPIO.setmode(GPIO.BCM)

GPIO.setup(FLOW\_SENSOR, GPIO.IN, pull\_up\_down = GPIO.PUD\_UP)

while True:

print("Pressure: %.2f hPa" % lps.pressure)

print("Temperature: %.2f C" % lps.temperature)

print("")

time.sleep(1)

#### Pressure sensor

import RPi.GPIO as GPIO

import time, sys

FLOW\_SENSOR = 26

GPIO.setmode(GPIO.BCM)

GPIO.setup(FLOW\_SENSOR, GPIO.IN, pull\_up\_down = GPIO.PUD\_UP)

global count

count = 0

global pressure

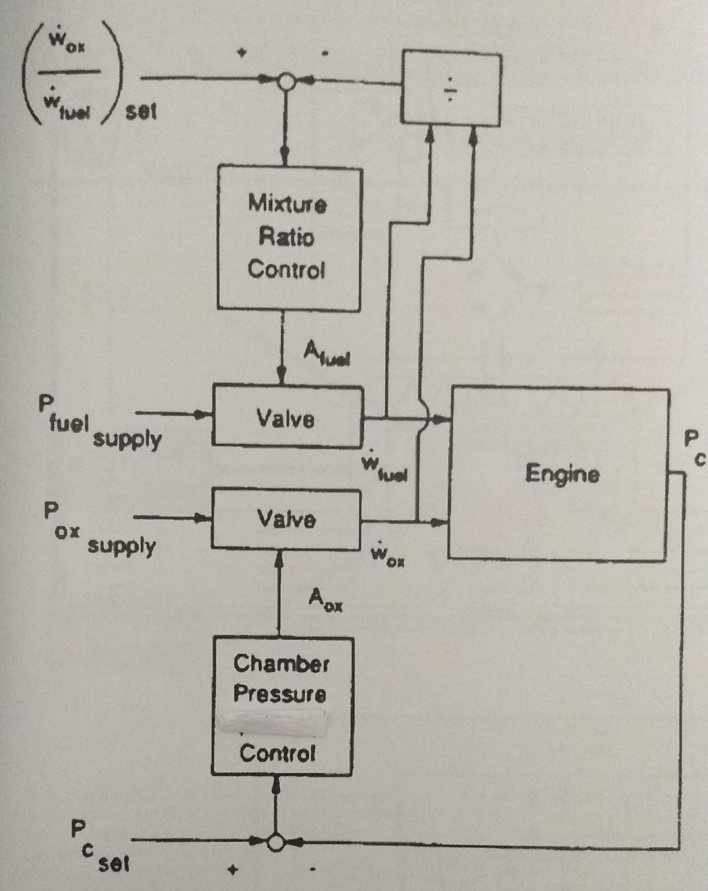
pressure = 0.0001

if pressure >= 0.0001:

print(0.0001)

## Combustion Control

2 flowsensor ودمجه مع ال Pressure sensorالهدف القادم هو إضافة



لقد قمنا اليوم تقريبا بنفس العمل السابق لأننا بحاجة الى تحكمين للوصول الى هدفنا و قد قمنا بتجربة ناجة و الهدف الان تركيب جمبع القطع



Automatic valve with DC motor

Flow sensor

Automatic valve with DC motor

Pressure sensor

1. Based on Trainee Report of Ali Ibrahim and Ali Awad, August 2020 [↑](#footnote-ref-1)