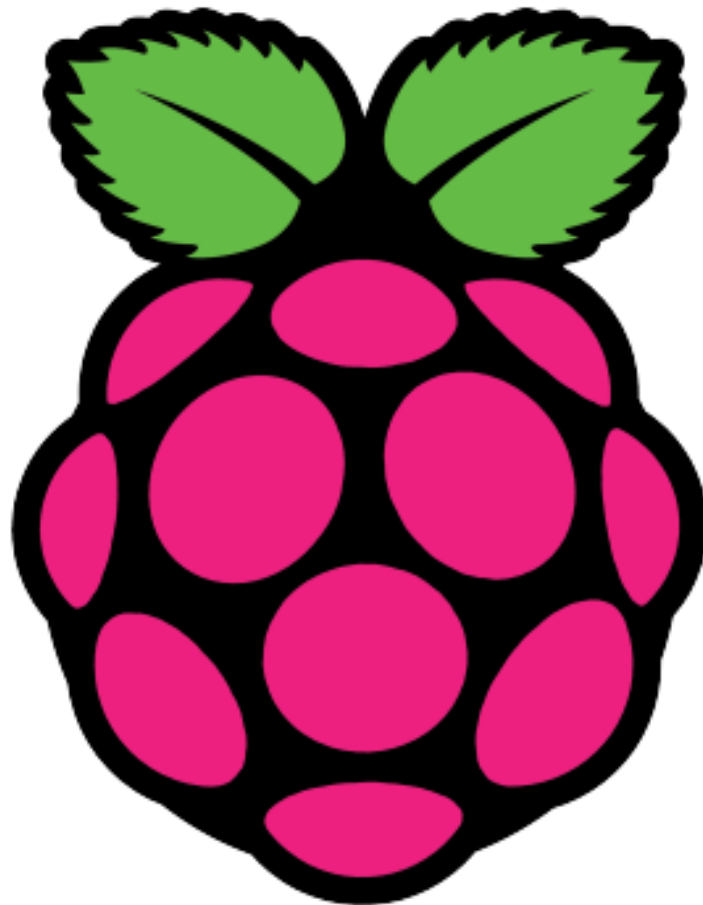


“Raspberry pi course”

ENG: AHMED MUBARAK

01032414034



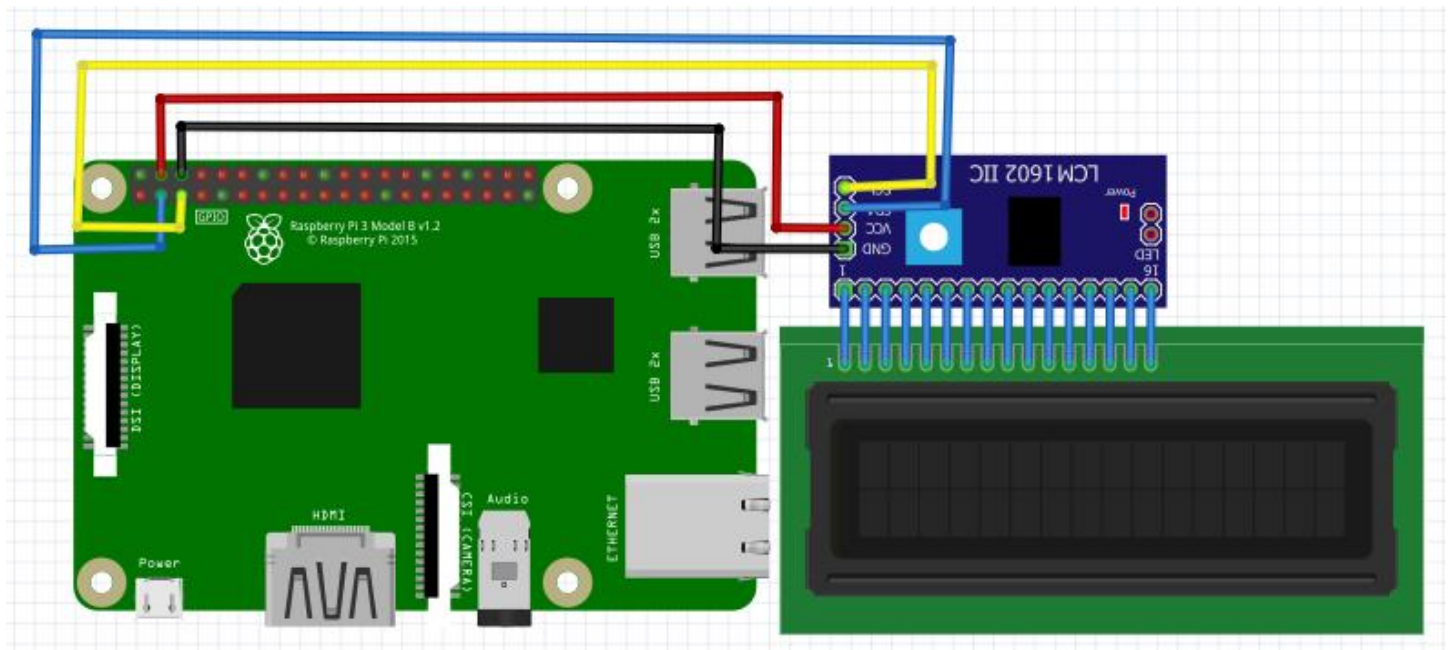
## SESSION NO. “8”

- 16\*2 I2C LCD
- CLOUD
- UPIDOTS

ENG.AHMED MUBARAK

01032414034

## 16\*2 I2C LCD



### • TO CHECK I2C ADDRESS : `i2cdetect -y 1` FIRST AND EASIEST WAY :

- `sudo pip3 install rpi_lcd`
- `sudo find /usr/local -name rpi_lcd 2> /dev/null`
- `/usr/local/lib/python3.7/dist-packages/rpi_lcd`
- `cd /usr/local/lib/python3.7/dist-packages/rpi_lcd`
- `sudo nano lcd.py` ( write your code here )

```
from signal import signal, SIGTERM, SIGHUP, pause
from rpi_lcd import LCD
lcd = LCD()
def safe_exit(signum, frame):
    exit(1)
try:
    signal(SIGTERM, safe_exit)
    signal(SIGHUP, safe_exit)
```

```

lcd.text("Hello,", 1)
lcd.text("Raspberry Pi!", 2)
pause()
except KeyboardInterrupt:
    pass
finally:
    lcd.clear()

```

- `sudo python3 lcd.py`

## THE SECOND WAY USING CODE:

### EXAMPLE CODE :

```

import smbus
import time

# Define some device parameters
I2C_ADDR = 0x27 # I2C device address
LCD_WIDTH = 16 # Maximum characters per line

# Define some device constants
LCD_CHR = 1 # Mode - Sending data
LCD_CMD = 0 # Mode - Sending command

LCD_LINE_1 = 0x80 # LCD RAM address for the 1st line
LCD_LINE_2 = 0xC0 # LCD RAM address for the 2nd line

```

```
LCD_LINE_3 = 0x94 # LCD RAM address for the 3rd line
```

```
LCD_LINE_4 = 0xD4 # LCD RAM address for the 4th line
```

```
LCD_BACKLIGHT = 0x08 # On
```

```
#LCD_BACKLIGHT = 0x00 # Off
```

```
ENABLE = 0b00000100 # Enable bit
```

```
# Timing constants
```

```
E_PULSE = 0.0005
```

```
E_DELAY = 0.0005
```

```
#Open I2C interface
```

```
#bus = smbus.SMBus(0) # Rev 1 Pi uses 0
```

```
bus = smbus.SMBus(1) # Rev 2 Pi uses 1
```

```
def lcd_init():
```

```
    # Initialise display
```

```
    lcd_byte(0x33,LCD_CMD) # 110011 Initialise
```

```
    lcd_byte(0x32,LCD_CMD) # 110010 Initialise
```

```
    lcd_byte(0x06,LCD_CMD) # 000110 Cursor move direction
```

```
    lcd_byte(0x0C,LCD_CMD) # 001100 Display On, Cursor Off, Blink Off
```

```
    lcd_byte(0x28,LCD_CMD) # 101000 Data Length, number of lines, font size
```

```
lcd_byte(0x01, LCD_CMD) # 000001 Clear display  
time.sleep(E_DELAY)
```

```
def lcd_byte(bits, mode):
```

```
    # Send byte to data pins
```

```
    # bits = the data
```

```
    # mode = 1 for data
```

```
    #           0 for command
```

```
    bits_high = mode | (bits & 0xF0) | LCD_BACKLIGH
```

T

```
    bits_low = mode | ((bits<<4) & 0xF0) | LCD_BACK
```

LIGHT

```
    # High bits
```

```
    bus.write_byte(I2C_ADDR, bits_high)
```

```
    lcd_toggle_enable(bits_high)
```

```
    # Low bits
```

```
    bus.write_byte(I2C_ADDR, bits_low)
```

```
    lcd_toggle_enable(bits_low)
```

```
def lcd_toggle_enable(bits):
```

```
    # Toggle enable
```

```
    time.sleep(E_DELAY)
```

```
    bus.write_byte(I2C_ADDR, (bits | ENABLE))
```

```
    time.sleep(E_PULSE)
```

```
    bus.write_byte(I2C_ADDR, (bits & ~ENABLE))
```

```
time.sleep(E_DELAY)

def lcd_string(message,line):
    # Send string to display

    message = message.ljust(LCD_WIDTH," ")

    lcd_byte(line, LCD_CMD)

    for i in range(LCD_WIDTH):
        lcd_byte(ord(message[i]),LCD_CHR)

def main():
    # Main program block

    # Initialise display
    lcd_init()

    while True:

        # Send some test
        lcd_string("AHMED",LCD_LINE_1)
        lcd_string("MUBARAK",LCD_LINE_2)

        time.sleep(4)

        # Send some more text
        lcd_string("Raspberry Pi",LCD_LINE_1)
```

```

lcd_string("Tutorial",LCD_LINE_2)

time.sleep(3)
lcd_string("",LCD_LINE_1)
lcd_string("",LCD_LINE_2)
time.sleep(4)

if __name__ == '__main__':

    try:
        main()
    except KeyboardInterrupt:
        pass
    finally:
        lcd_byte(0x01, LCD_CMD)

```

## CLOUD

“I WILL SEND AN EMAIL TO MYSELF”

- USING “ <https://ifttt.com/> ” .
- MAKE AN ACCOUNT ON “IFTTT”.
- CLICK “ CREATE ”.
- CLICK “ IF THIS ”.
- SEARCH FOR “ WEBHOOKS ” AND CHOOSE IT.
- CHOOSE RECEIVE A WEB REQUEST.
- NAME THE EVENT EMAIL AND CLICK “ CREATE TRIGGER ”.
- CLICK “ THEN THAT ”.
- SEARCH FOR “ GMAIL ” AND CHOOSE IT.
- CHOOSE SEND YOURSELF AN EMAIL.
- TYPE THE SUBJECT AN THE BODY OF THE EMAIL AND CLICK “ CREATE ACTION ”.
- CLICK CONTINUE AND CLICK FINISH.
- NOW, GO TO “ MY SERVICES ” AND CHOOSE “ WEB HOOKS ” THEN CLICK ON DOCUMENTATION TO GET THE LINK OF THE EVENT.
- NOW, GO TO RASPBERRY PI TO FIRE THE EVENT .



- **1.** As we may need to install some additional packages, let's start by ensuring our Raspberry Pi is running the latest available software.
- You can do this by running the following two commands.

```
•sudo apt update
```

- `sudo apt upgrade`

- **2.** With the packages up to date, we can now make sure Python 3 and some of its dependencies are installed.
- We will be using Python to interact with our IFTTT webhook.
- To ensure Python is installed, run the following command.

```
•sudo apt install python3 python3-pip
```

- **3.** Now with Python 3 installed, let's ensure that we have access to the Python Requests library.
- Run the following command to use `pip` to install the requests module.
- `sudo pip3 install requests`
- You can now proceed to write a Python script that will interact with your IFTTT webhook from your Raspberry Pi.

---

## • Interacting with IFTTT from your Raspberry Pi

---

- **1.** Let's begin the process of writing our Python script by using the following command.
- This first script will show you the very basics of how to use an IFTTT webhook.

```
•nano ~/ifttt.py
```

- **2.** We should start this script by importing the `requests` library.

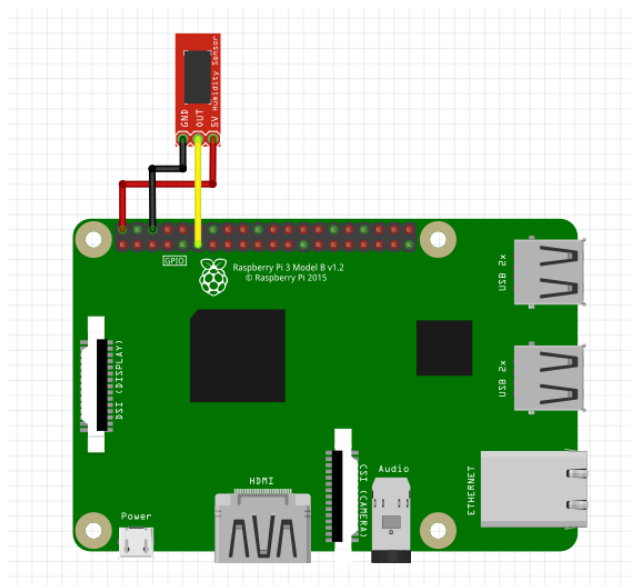
```
•import requests
```

- **3.** Our next step is to make a post request to the webhook URL provided by IFTTT.
- To make a request there are two things you will need to know.
- The first is the **event name** that you defined for your webhook. In our example, we used the event name "**motion\_detected**".
- The second thing you will need is the webhook API key. You should have found this earlier on in the guide when setting up your IFTTT action.

- With both of these values in hand, we need to enter the following line. Make sure to replace both "{EVENT\_NAME}" and "{YOURAPIKEY}" with the relevant pieces of information.
- ```
requests.post('https://maker.ifttt.com/trigger/{EVENT_NAME}/with/key/{YOURAPIKEY}')
```
- **4.** That is all we need to do to make a webhook request in Python.
- Save the file by pressing **CTRL + X**, then **Y**, followed by **ENTER**.
- **5.** Before we test this script, make sure that you have the IFTTT app installed on your phone.
- You can install the IFTTT app through the [Google Play Store for Android](#) or the [App Store for iPhones](#).
- Without this app, you will never receive the notification.
- **6.** Once you have the IFTTT app installed, test the script by running the command below.
- ```
python3 ~/ifttt.py
```
- If everything is working correctly, you should see a notification pop up on your device.

**UPIDOTS**

## SHOW TEMP AND HUMIDITY ON UPIDOTS



```
import time
import requests
import math
import random
import sys
import Adafruit_DHT

TOKEN = "BBFF-
fVw8lRQ2f334KlkLQA784UDLRACpQG" # Put your TOKEN h
ere
DEVICE_LABEL = "raspberry" # Put your device label
here
VARIABLE_LABEL_1 = "temperature" # Put your first
variable label here
VARIABLE_LABEL_2 = "humidity" # Put your second va
riable label here

def build_payload(variable_1, variable_2):
    # Creates two random values for sending data
    humidity, temperature = Adafruit_DHT.read_retry
(11, 14)

    # Creates a random gps coordinates
    lat = random.randrange(34, 36, 1) + \
        random.randrange(1, 1000, 1) / 1000.0
    lng = random.randrange(-83, -87, -1) + \
        random.randrange(1, 1000, 1) / 1000.0
```

```
payload = {variable_1: temperature,  
           variable_2: humidity}
```

```
return payload
```

```
def post_request(payload):
```

```
    # Creates the headers for the HTTP requests
```

```
    url = "http://industrial.api.ubidots.com"
```

```
    url = "{} /api/v1.6/devices/{}".format(url, DEVICE_LABEL)
```

```
    headers = {"X-Auth-Token": TOKEN, "Content-Type": "application/json"}
```

```
    # Makes the HTTP requests
```

```
    status = 400
```

```
    attempts = 0
```

```
    while status >= 400 and attempts <= 5:
```

```
        req = requests.post(url=url, headers=headers, json=payload)
```

```
        status = req.status_code
```

```
        attempts += 1
```

```
        time.sleep(1)
```

```
    # Processes results
```

```
    if status >= 400:
```

```
        print("[ERROR] Could not send data after 5 attempts, please check \
```

```
        your token credentials and internet con
nection")
        return False

    print("[INFO] request made properly, your devic
e is updated")
    return True

def main():
    payload = build_payload(
        VARIABLE_LABEL_1, VARIABLE_LABEL_2)

    print("[INFO] Attempting to send data")
    post_request(payload)
    print("[INFO] finished")

if __name__ == '__main__':
    while (True):
        main()
        time.sleep(1)
```

---

*With my best wishes:*

*ENG : AHMED MUBARAK*

---