BUS Automation

Bluetooth Enable in PI

Con**figuring Bluetooth as an A2DP Sink**

**sudo apt-get install pulseaudio pulseaudio-module-bluetooth**

**sudo usermod -a -G bluetooth pi**

**sudo reboot**

Let's make our Pi permanently discoverable as an A2DP Sink.

**sudo nano /etc/bluetooth/main.conf**

And add / uncomment / change

**...**

**Class = 0x41C**

**…**

**DiscoverableTimeout = 0**

**…**

**sudo systemctl restart bluetooth**

**Bluetoothctl**

We need to start pulseausio manually now.

**pulseaudio --start**

And now let's check if everything is OK.

**sudo systemctl status bluetooth**

And it is ! Bluetooth is up and running and A2DP is registered.  
  
At this stage, you should be able to see you Pi as an A2DP Source/Sink from your mobile bluetooth menu.  
  
Let's go for a quick test. Try to connect to your Pi. You should be able to pair but you can't connect. Your device must be trusted first. Let's do it manually for the purpose of a quick test

**Bluetoothctl**

Now you should be able to connect. And play Music !!!

**systemctl --user enable pulseaudio**

**sudo apt install python-dbus**

Reference: <https://forums.raspberrypi.com/viewtopic.php?t=235519>

**CODE:**

**File Name :Bluetooth.py**

FM Radio

TEA5767 is a module that uses the I2C serial protocol

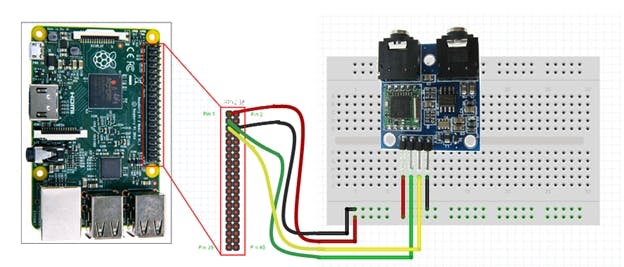
The TEA5767 has 4 IO pins, connect them as follows:1  Connect to ground on the RPi4 (Pin 6)

1. **GND:** Connect to ground on the RPi4 (Pin 6)
2. **VCC:** Connect to 3.3V on the RPi4 (Pin 1)
3. **SDA:** Connect to SDA on the RPi4 (Pin 3). This is the data line for the I2C bus.
4. **SCL:** Connect to SCL on the RPi24(Pin 5). This is the clock line for the I2C bus.

Enable I2c in RPi:

**Sudo Raspi-config**

Goes to interface section and select i2c enable Ok



**CODE:**

**File Name : realfm.py**

**Interface Enable :**

**Sudo raspi-config**

Interface section

I2c enable

Spi enable

Serial 1st prompt no 2nd prompt yes enable

Camera enable

Audio section - audio through earphone

**For Camera Library install**

**sudo apt-get update**

**sudo apt-get upgrade**

**sudo apt-get install libatlas-base-dev**

**sudo apt-get install libjasper-dev**

**sudo apt-get install libqtgui4**

**sudo apt-get install libqt4-test**

**sudo apt-get install libhdf5-dev**

**sudo pip3 install numpy**

**sudo pip3 install opencv-contrib-python**

**sudo pip3 install imutils**

**sudo pip3 install opencv-python**

Nodemcu(HUB)

**Connections :**

**Tx0,Tx2,Rx0,Rx2,32,33**

**Code:**

**File Name :TRY\_ESP32SERIAL /ESP\_Serial\_Dat\_receive**

Distance (HUB)

**Connections :**

**TrigPin are 2,4,6,8,10**

**Echo Pins are 3,5,7,9,11**

**Code:**

**File Name :Distance\_Hub**

Weight (Fire Hydrant)

****Download the "HX711\_ADC" library:****

[https://github.com/olkal/HX711\_ADC](https://github.com/olkal/HX711_ADC" \t "https://circuitjournal.com/_blank)

Click on the green "Clone or download" button and then "Download ZIP."

Extract the downloaded ZIP file and move it into your Arduino "libraries" folder.

On Windows:

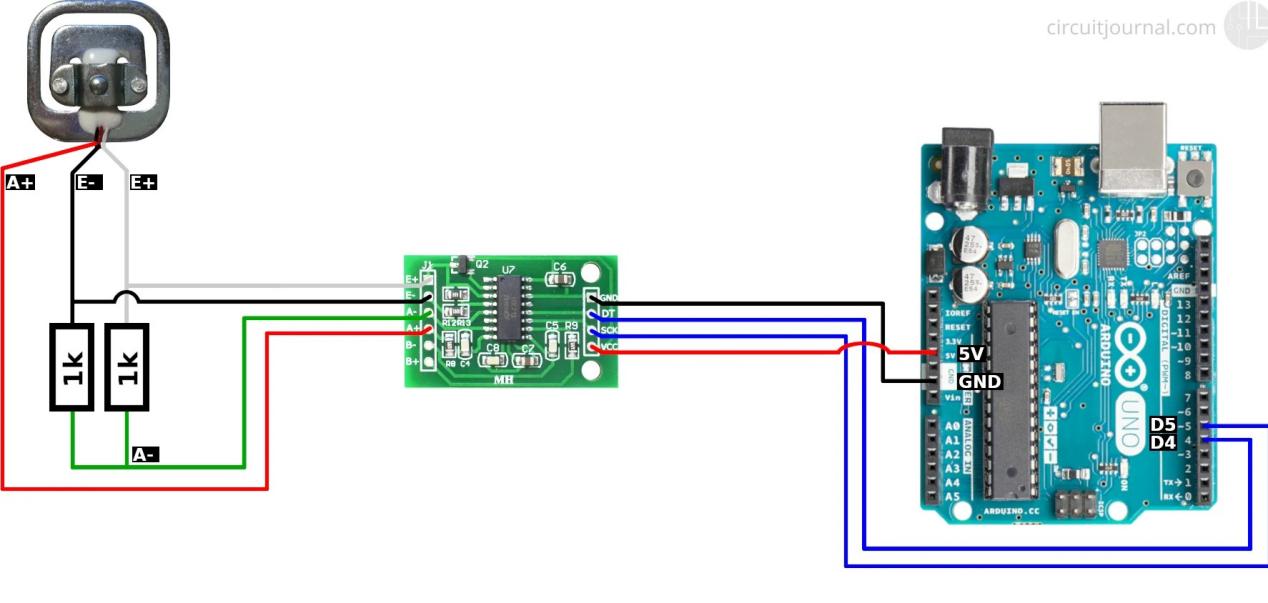
C:\Users\<username>\Documents\Arduino\libraries

****Open the Calibration example that came with the "HX711\_ADC" library.****

Open Arduino IDE and then from the menu:

File->Examples->HX711\_ADC->Calibration

**Calculate the calibration value and store it because it will be needed in main program**

****

**CODE:**

**File Name : weight\_sensor #do some chnges if required**

Reference: https://circuitjournal.com/50kg-load-cells-with-HX711

Fuel, Temp Hub

**Connections :**

**Understandable from code**

**CODE:**

**File Name : fuel\_theft\_gate\_temp**

Feedback Sensor

**Connections :**

**Used push button with resistor in place of feedback sensor**

**CODE:**

**File Name :feedback.py**

Reference:https://raspberrypihq.com/use-a-push-button-with-raspberry-pi-gpio/

LED Strip

**CODE:**

**File Name :LEDSTRIP**

Reference: Json Library of Arduino

Matrix Board

**CODE:**

**File Name :P10\_nodemcu\_display**

Reference: https://medium.com/@tharindu\_peiris/p10-led-display-panel-using-esp8266-wemos-d1-mini-nodemcu-1e463ad60722/

Extra Codes

**CODE:**

Data Receive from esp32 to pi code // type of data changed

**File Name :fuelsensor.py**

MicroPython Fm code if want to add new features

**File Name :FM\_MicroPython.py**

ITPMS

Reference: <https://makersportal.com/blog/2018/10/3/arduino-tachometer-using-a-hall-effect-sensor-to-measure-rotations-from-a-fan>

HTTP esp32

Reference: <https://esp32io.com/tutorials/esp32-http-request>

MAX IC with Arduino Reference:https://diyodemag.com/education/the\_classroom\_rs232\_the\_max232\_ic\_arduino\_uno