# Imperial College RADAR & Sensing Project

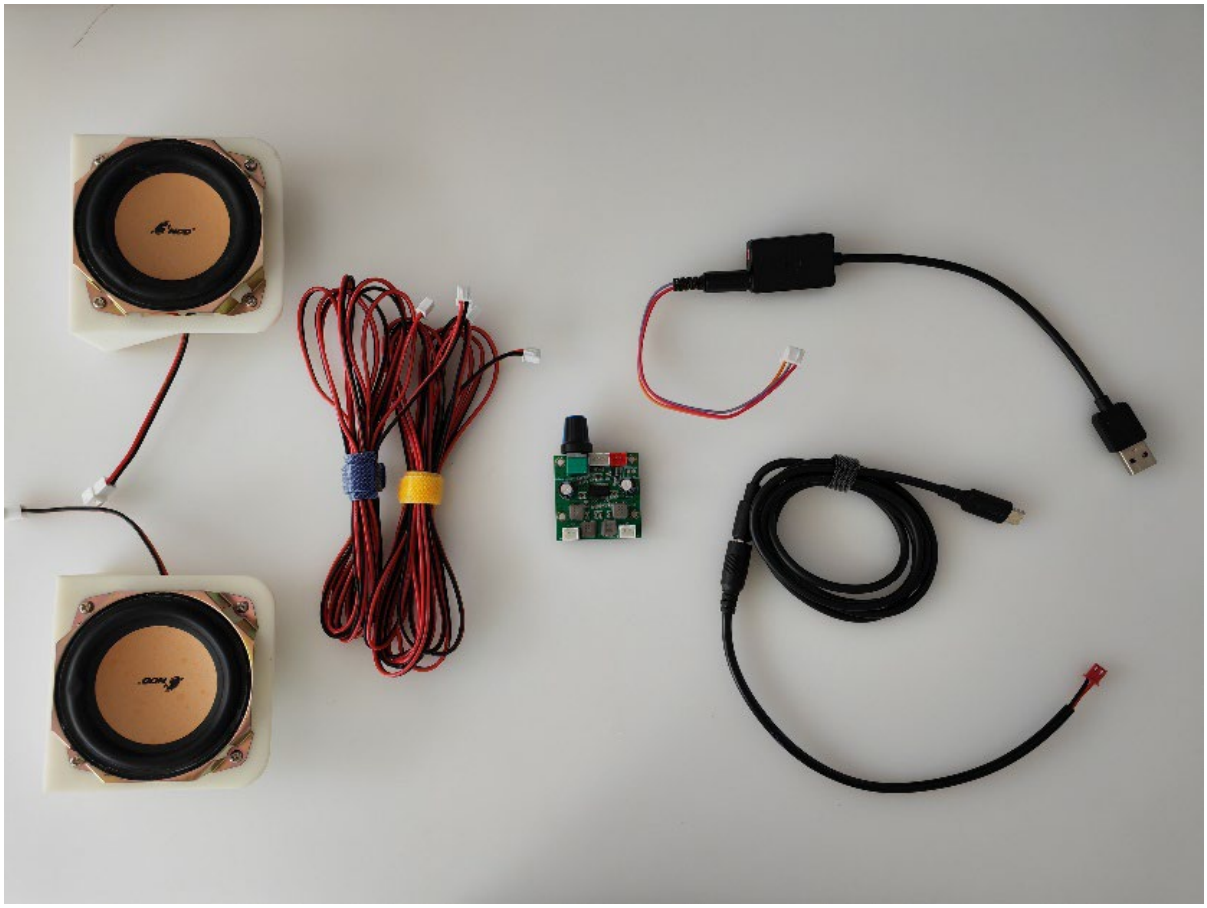
This document details the setup of an eight-channel microphone and speaker to record and play sounds. An additional software demonstration and code template is included.

**Hardware Details:**

Microphone: Yundea USB 8MIC Array v1.5

Speaker: 2x 4Ω 10W drivers with TPA3110D2 amplifier board

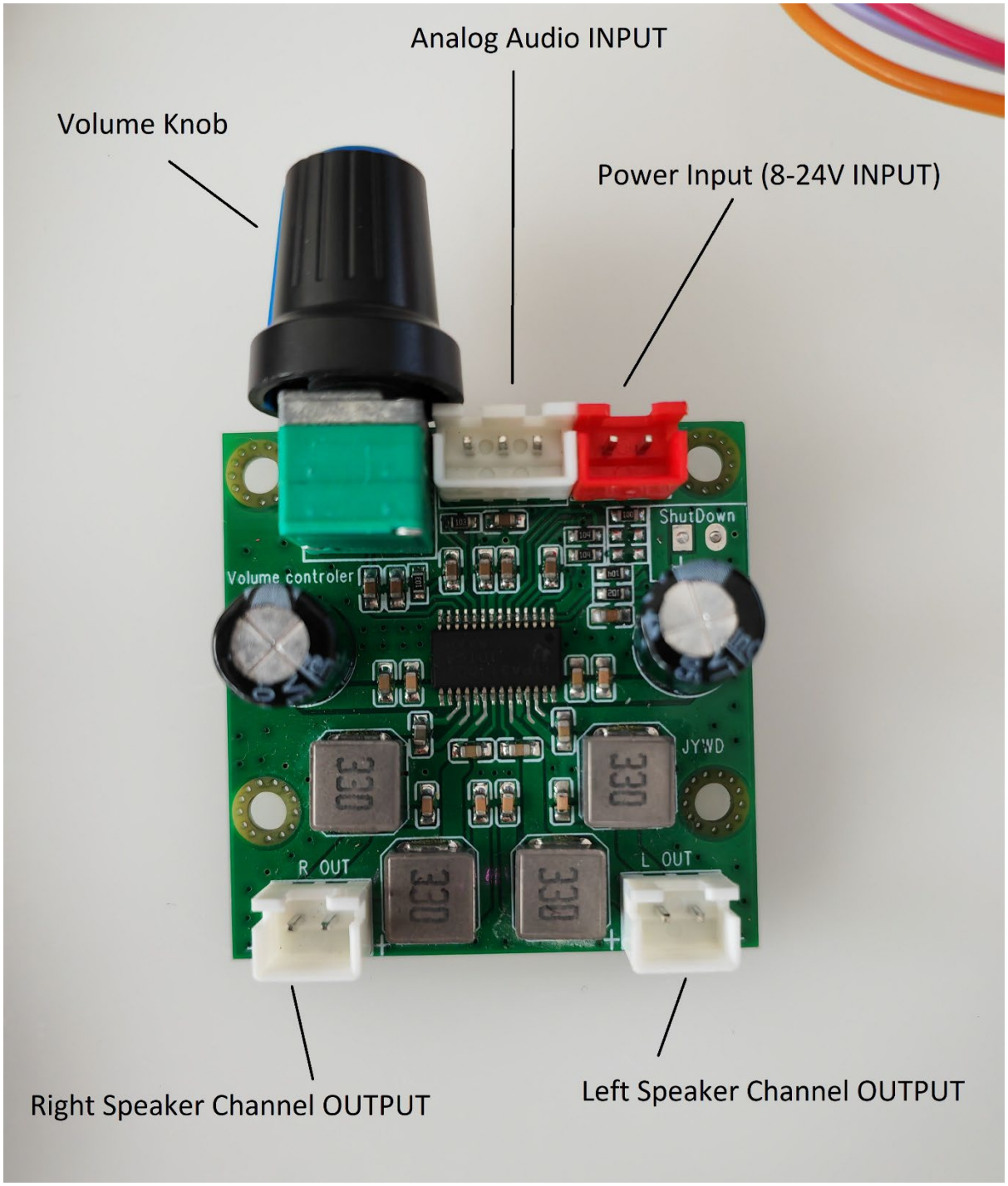
## Speaker Hardware Setup



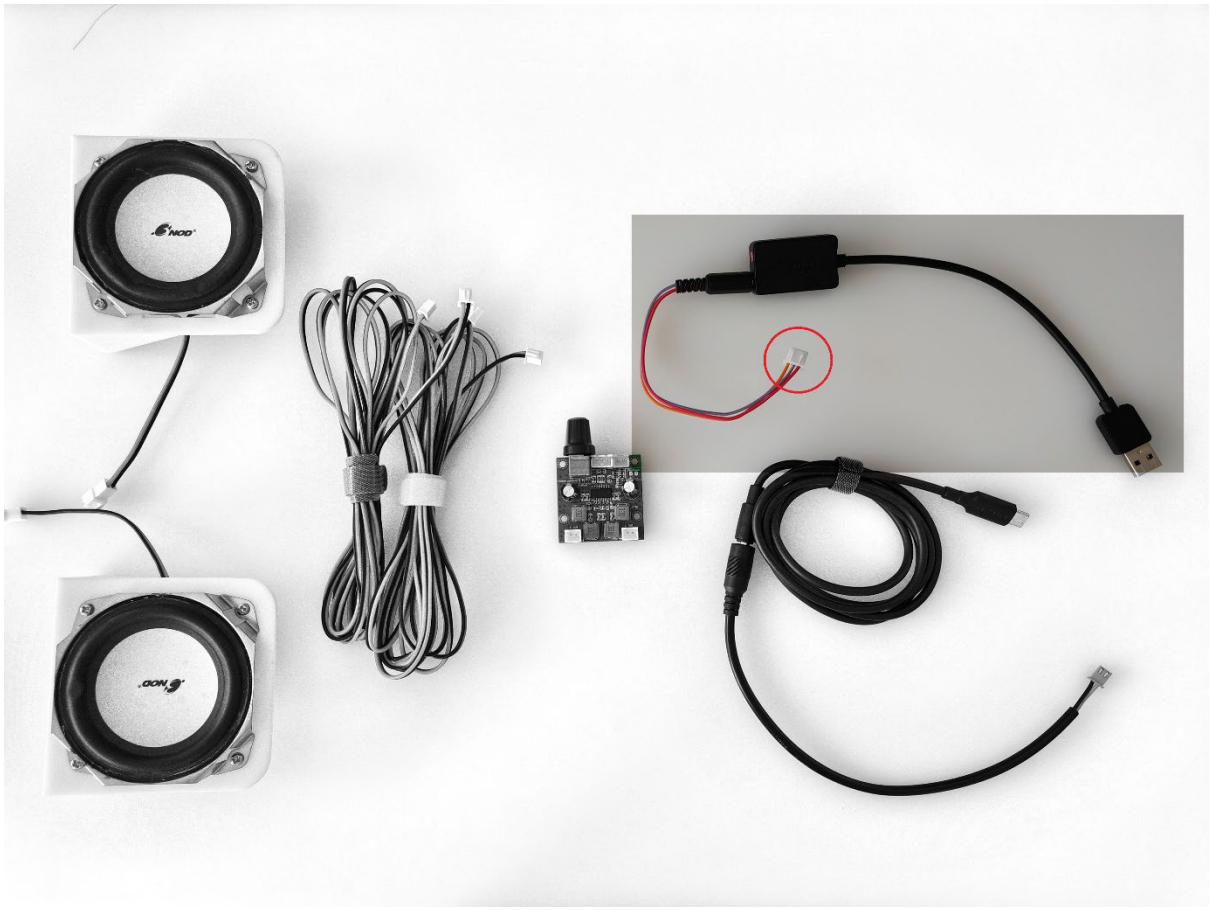
The Speaker Setup can be split into subsections. The heart of this system lies within the circuit board, which provides a bridge between the PC input interface (USB) and the speaker output interface (analog).

To the right of this board are connections to the PC, and to the left are connections to the Speakers.

The pins on this board can also be labelled.

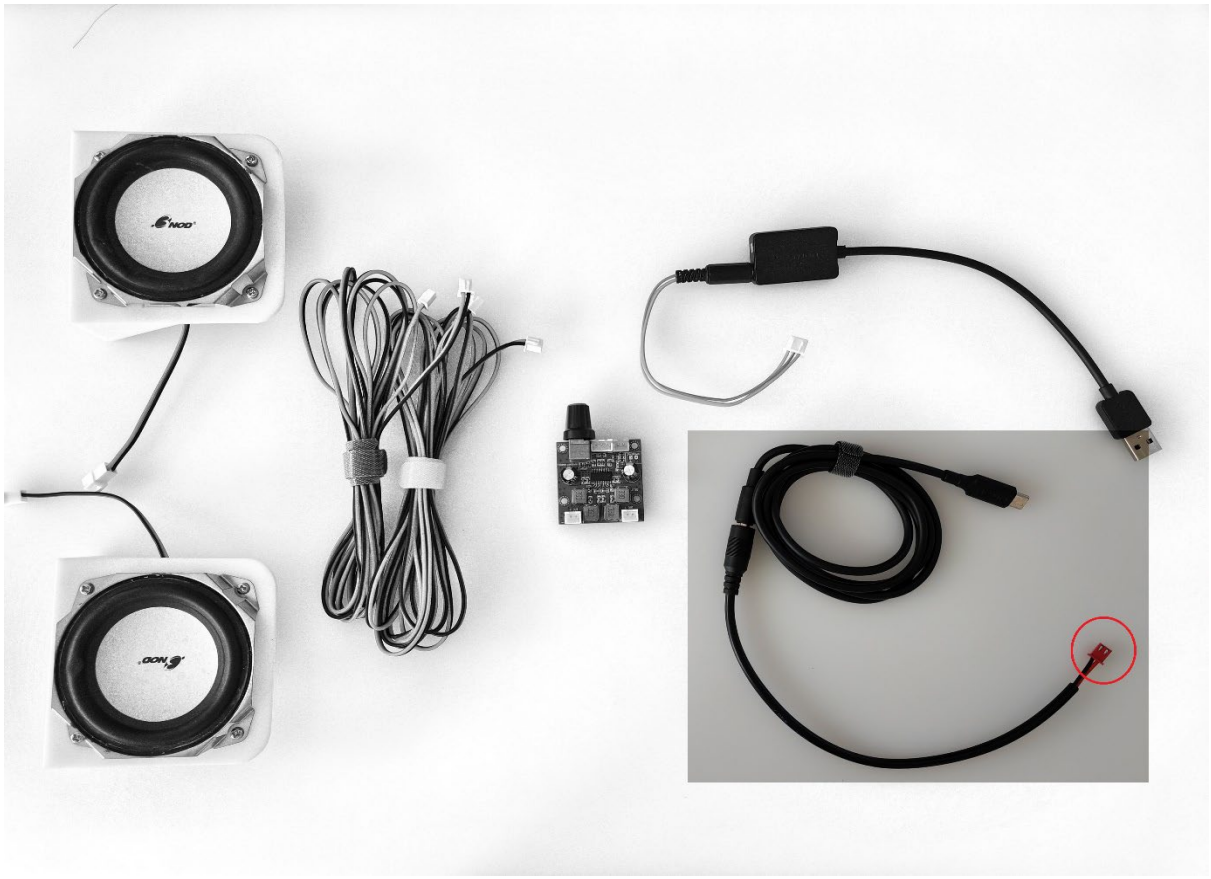



The volume knob is self-explanatory – twisting it changes the volume. To the right of this knob are two input connectors. The white three pin substitutes for an audio (AUX) jack. To connect this to the PC, the top right cable is employed.



The circled connector plugs into the connector on the main board labelled ‘Analog Audio INPUT’. The other end connects to the PC.

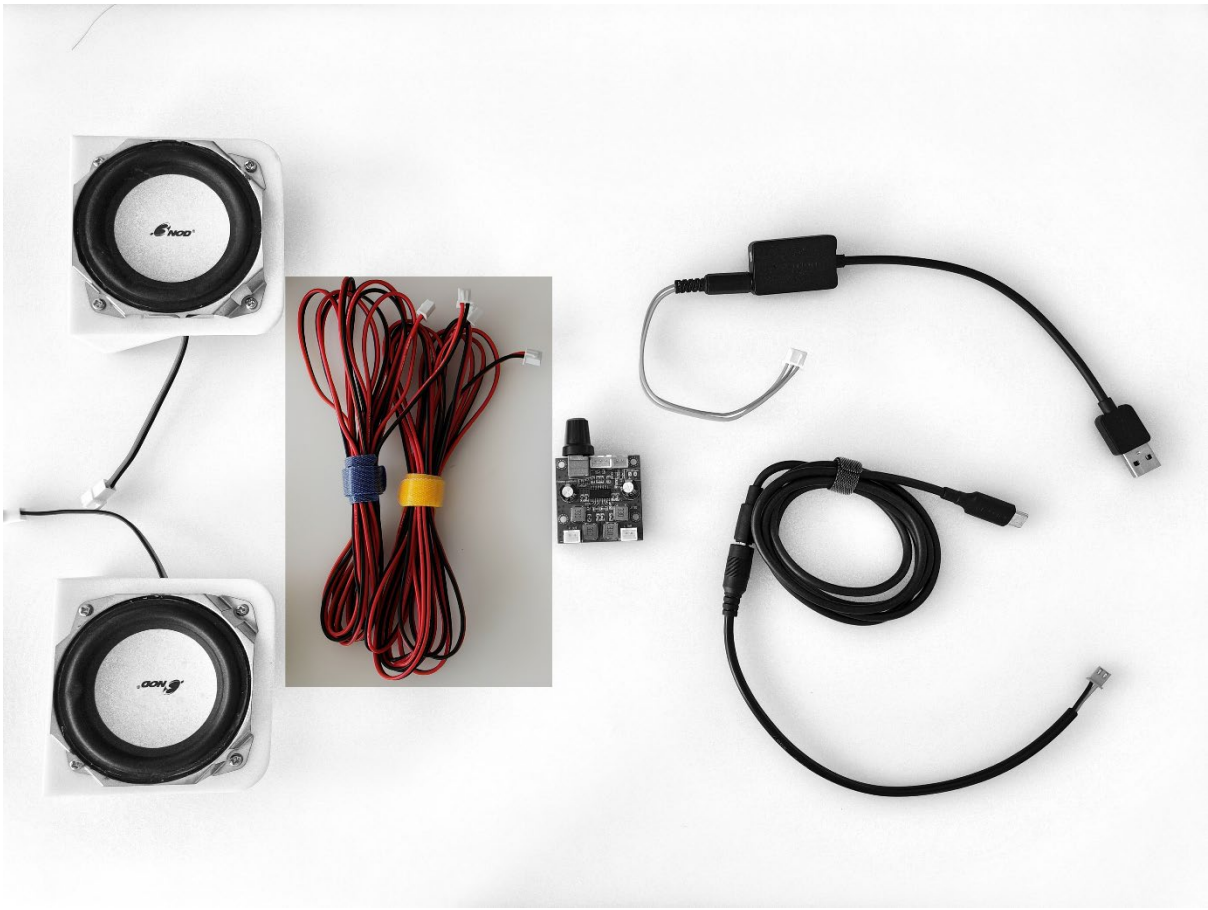
The cable below provides power to the board. On one end is a normal type-C connector (non-circled). This plugs into the supplied power bank (not shown in the picture). The other end (circled) plugs into the connector on the main board labelled ‘Power Input (8-24V INPUT)’.



Not every mechanically compatible USB-C port supports the 12V USB-PD specification required for this device to function. Ensure third party power sources support 12V PD with minimum 1.5A current. Misuse could result in a damaged main board.

If the power bank doesn't light up with a number (indicating percentage of battery left) within a few seconds of plugging the board in, please press the silver button on the side. If the power bank still does not light up, please charge it.

Following, one of the red/black speaker cables (highlighted below) should be plugged into each speaker channel ('Right Speaker Channel OUTPUT' and 'Left Speaker Channel OUTPUT'). The other end should be plugged into the cable coming out of the speaker. Both ends of the red/black speaker cable are the same, it does not matter which end is plugged into which device.

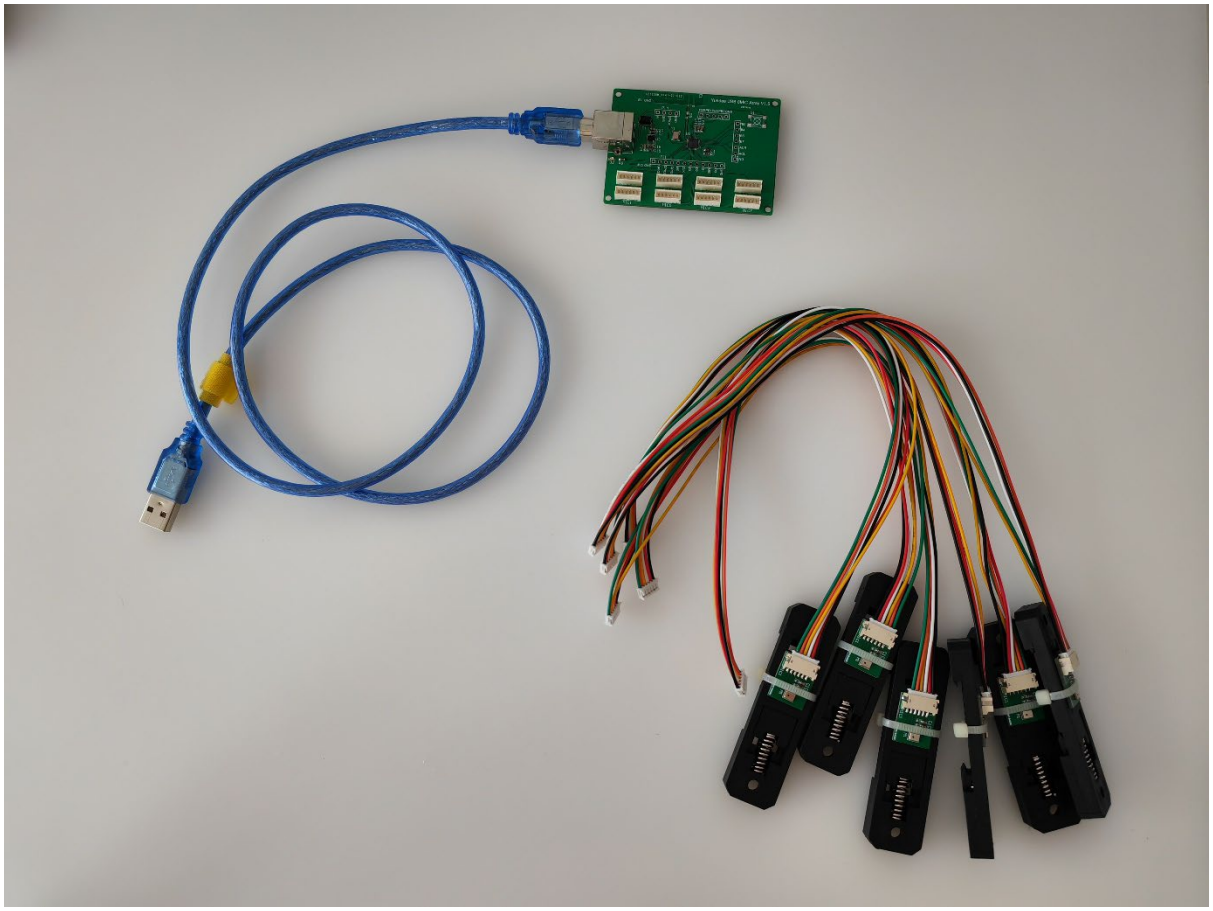


To review this section: Both speakers should be plugged into the main board using their red/black cables. One USB connection should be established to the PC. One power connection should be established to the power bank.

Windows will automatically detect this device as a normal speaker/headphone.

## Microphone Hardware Setup

Mechanically, the setup for the microphones is much easier. The equipment consists of only a driver board that supports up to eight individual microphones and the accompanying microphone modules.



Once again, this equipment connects to the PC with a USB cable.

The microphone connectors are positioned in such an array:

| *<- USB Connector* | *Text this way up* | | |
|---|---|---|---|
| 2 | 6 | 4 | 8 |
| 1 | 5 | 3 | 7 |

In software these are mapped accordingly. If using less than eight microphones, please fill up the connectors in their ascending numerical value.

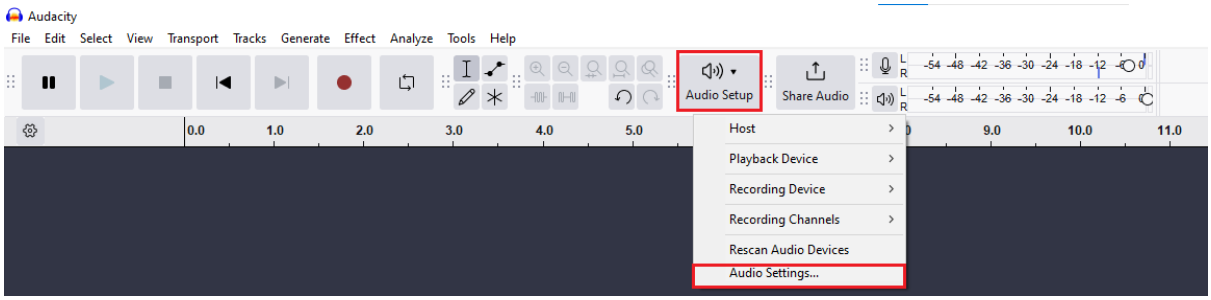
To review this section: each microphone should be connected to the driver board using its accompanying cable and the driver board should be in turn connected to the PC over USB.

Windows will automatically detect this device as an 8-channel microphone.

## Software Setup

The speakers do not require any extensive software testing. Play any audio file from the PC and ensure both speakers are outputting sound.

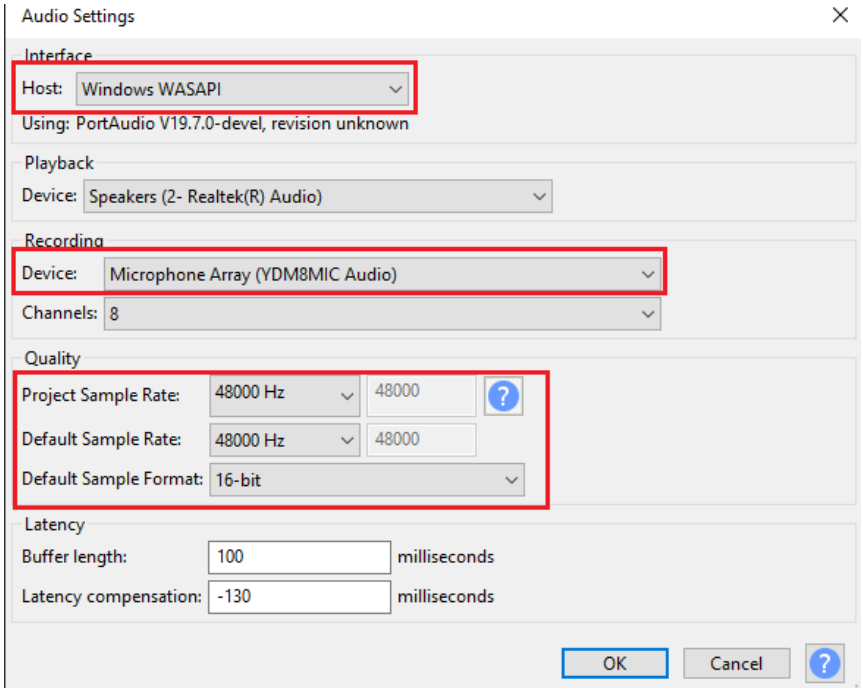
Audacity can be used to test the functionality of each microphone. To setup the microphones in the software, click the 'Audio Setup' in the top bar and select 'Audio Settings' at the bottom.




In the window that opens, select the following:

- Host: 'Windows WASAPI'
- Device: 'YDM8MIC Audio'
- Project Sample Rate: 48000 Hz
- Default Sample Rate: 48000 Hz
- Default Sample Format: 16-bit

Set the 'Channels' accordingly to how many microphones are plugged in.




Click OK to exit the window, then the red circle to start recording. *If an error is thrown, try changing the playback device.*

## Programming Example

All files mentioned can be found at https://github.com/jinghuahe123/Sensing-Project.

For this demonstration, the focus will be on these files:

- main_eight_channel.py
- main_playback.py
- demo_eight_channel.py
- demo_playback.py
- demo_record_and_play.py

The structure of these files goes as follows:

- "main_eight_channel.py" and "main_playback.py" are the files that contain the main code for processing the audio. "main_eight_channel.py" contains the functions for recording eight-channel audio and "main_playback.py" contains the functions for playing chirp signals. These files do not support being run by themselves.
- "demo_eight_channel.py", "demo_playback.py" and "demo_record_and_play.py" are the files that reference the main files to either record or play audio. These files handle the user inputs which then get passed on to the main files. Run these files to use the program.

For these programs, the 'sounddevice' and 'scipy' libraries for handling the audio data. Additional libraries used for plotting data and mathematical calculations include 'numpy' and 'matplotlib'.

### Recording

For recording audio, run "demo_eight_channel.py". This program takes two inputs – a duration to record for (in seconds) and a filename to output the file to (in .wav format). The program will record for the specified duration, save the recording to the .wav file and plot the soundwave on a graph.

Optionally, uncomment line 15 and comment out line 14 if the graph is not desired. For a more professional look at the soundwave, load the .wav file into Audacity.

### Playback

For playing chirp signals, run "demo_playback.py". This program takes three inputs – a duration to play for, a starting frequency and an end frequency. It will play a chirp signal based on the parameters given, then save that chirp signal to a .wav file (filename auto generated).

Optionally, comment out line 5 if saving the audio file is undesirable.

## Recording a Chirp

For simultaneous recording and playing, run “demo_record_and_play.py”. This program takes three inputs: a duration to run for, and a start and end frequency for the chirp signal. It will record the chirp signal generated, then save both the chirp signal and the recorded audio to separate wav files.

This program leverages the “threading” library for simultaneous operation of both functions.

Optionally, comment out line 18 if saving the chirp signal is undesirable.