In-Ear Microphone Array Design for Ear Health Monitoring

A Major Qualifying Project report

submitted to the faculty of

WORCESTER POLYTECHNIC INSTITUTE

in partial fulfillment of the requirements for the

degree of Bachelor of Science

By:

Haoyu Fu [1]

Advisor:

Prof. Bashima Islam [1,**]

Date:

2023/3/6

[1] Worcester Polytechnic Institute, Worcester, MA, United States of America

Table of Contents

ABSTRACT:

In this project, we study the design challenges of developing a microphone array for ear health monitoring. We investigate the effects of sampling rate, distance between microphones, and the number of microphones on the design. Literature[28] shows that the distance between two microphones needs to be half of the wavelength to have beneficial phase difference.Therefore, we found that the minimum sampling rate of such a microphone array needs to be at least 8000Hz with a minimum of 1.5cm distance between the microphones. Then we investigate various data reading and processing protocols available in common of the shelf embedded platforms with the goal to identify the suitable protocol and platform. We looked into three designs, where we used Raspberry Pi with I2S protocol in the first design, Arduino and Raspberry Pi (with external ADC) with SPI protocol in the second design, and Raspberry Pi with 4-channel ADC (supports I2S interface) in the third design. Our first design achieved the desired sampling rate but failed to support more than two microphones. However, literature shows that having a three-microphone array is important to remove front-back ambiguity[25]. Our second design acquires data from four microphones but fails to achieve the desired sampling rate. In the third design, we used Raspberry Pi and 4-channel ADCs with I2S to support four microphones with minimum sampling rate of 8000Hz.

1. Introduction

Earbuds or in-ear headphones have become increasingly popular over the past decay due to their compact size and convenience. For instance, Apple sold 19.3 million units of AirPods in Q1 2022 alone. All true wireless (TWS) headphones shipments in the third quarter of 2022 are 76.9 million units, with 4.2 million units representing the latest Apple AirPods Pro 2[29]. These small earpieces fit inside the ear canal and wirelessly connect to smartphones, tablets, or other computing devices. They are a popular alternative to over-ear headphones for listening to music, podcasts, and other audio content on the go.

There are earbuds with various health applications beyond just listening to music or audio content. Some earbuds come with built-in sensors that can monitor various health metrics such as heart rate from Bose Soundsport Pulse wireless[1] and blood pressure from Valencell[17]. These features can be particularly useful for individuals who are looking to track their fitness goals or manage certain health conditions. Therefore, earbuds have the potential to offer a range of health benefits beyond just entertainment, making them an increasingly popular choice for health-conscious individuals[18].

Ear health is essential for maintaining overall well-being and quality of life. The ears play a crucial role in our ability to communicate, maintain balance, and process sound, making it important to take care of them. Issues with hearing, balance, or other ear-related problems can significantly impact our daily lives, causing difficulties with communication, mobility, and overall health.

The location of these earbuds at the entrance of ear canals gives us a unique opportunity to monitor ear health, which is expensive and cumbersome to monitor. It enables individuals to monitor their ear health and identify potential issues at home, although it is important to note that these devices are not a substitute for professional medical care and advice. The current ear health parameters that can be monitored with earbuds include ruptured eardrum, earwax buildup and blockage, and otitis media[18]. Previous works[20] has shown the benefits of using the existing sensors – microphones and inertial measurement units (IMU) of earbuds for monitoring ear health. These works only focus on the single in-ear microphone present in commercial earbuds. Microphone arrays have shown promising success in many acoustic health applications[21] and can suppose complex acoustic signal processing algorithms, e.g., beamforming, sound source localization, and source separation.

MEMS microphones are most commonly used for earphones due to their omnidirectional pickup responses. This property allows them to respond equally to sounds in different directions. However, to focus on different parts of the environment dynamically, multiple microphones can be arranged in an array and create a directional response or a beam pattern. This allows for the isolation of sounds arriving from specific directions while suppressing noise and interference from other directions. This makes it a valuable tool in applications such as speech recognition, acoustic sensing, noise reduction and spatial information sensing in various environments; some of these applications can't be achieved with one single microphone.

However, none of the existing earbuds are equipped with in-ear microphone array. To illustrate, the AirPods Pro from Apple (Figure 2.2.1) has one microphone facing inside of the ear in each piece of earbuds for noise cancellations and ear tip fit test. Thus, to study the opportunities presented by an in-ear microphone array, we first investigate the challenges of designing such array. In this study, we focus on analyzing different design choices of an in-ear microphone array.

1.1 Goal

This project aims to design the new hardware that solves the limitations of the current earables for spatial acoustic sensing on a smaller scale. In particular, the hardware design should satisfy the following requirements:

R1. The microphone array needs to be small enough to fit in an in-ear microphone.

R2. The distance between two microphones in the array needs to be sufficient to preserve diversity in spatial information among the mic-array.

R3. The microphones arrangement must have both central symmetry and axial symmetry. For instance, the arrangement of microphones can be in a square, a circle, or an equilateral triangle.

R4. The sampling rate from multiple microphones needs to be sufficiently high during dada read using an embedded system.

In this project, we present three hardware designs which aims to satisfy these requirements with various advantages and limitations. By providing a comparative analysis, we show the shortcomings of current components and describe the necessary steps required to develop the target platform in the future.

2. Related Works

This section presents the state-of-the-art for earables and the custom designs that are used in health applications.

2.1 Earable for Ear-Health

One of the works[15] that has been done on ear-health with earphones is EarEcho, which is using ear canal for wearable authentication. The EarEcho takes advantages of the unique physical and geometrical characteristics of human ear canal and assesses the content-free acoustic features of in-ear sound waves for user authentication in a wearable and mobile manner.

Another work[16] that has been done is TeethPass, which is Dental Occlusion-based User Authentication via In-ear Acoustic Sensing. The work investigates novel biometric traits based on dental occlusion and discovers that the bone-conducted sound of dental occlusion recorded in the canals of the ears contains distinctive characteristics of various bones and teeth. The TeethPass collects occlusal noises in binaural canals using headphones. To identify bone-conducted noises, the work develops an event detection technique based on spectrum variance and double thresholds. After filtering out motion noises from the sounds using time-frequency analysis, three distinct user characteristics can be derived: bone structure, occlusal position, and occlusal sound. Finally, they create a Siamese network based on incremental learning to build the classifier.2.2 Current Earable Mic Design.

Most flagship earbuds in the industry, e.g., AirPods Pro, are equipped with multiple microphones to support active noise cancellation. Active noise cancellation uses a microphone on the outside of the earbud that listens to the background noises and block them out to create an immersive listening experience. To illustrate, AirPods Pro consists of three microphones, one on the tip of the stem focusing on the human speech, one on the base of the stem listening to environmental noise for noise cancellation, and one in-ward facing microphone listening inside the ear.

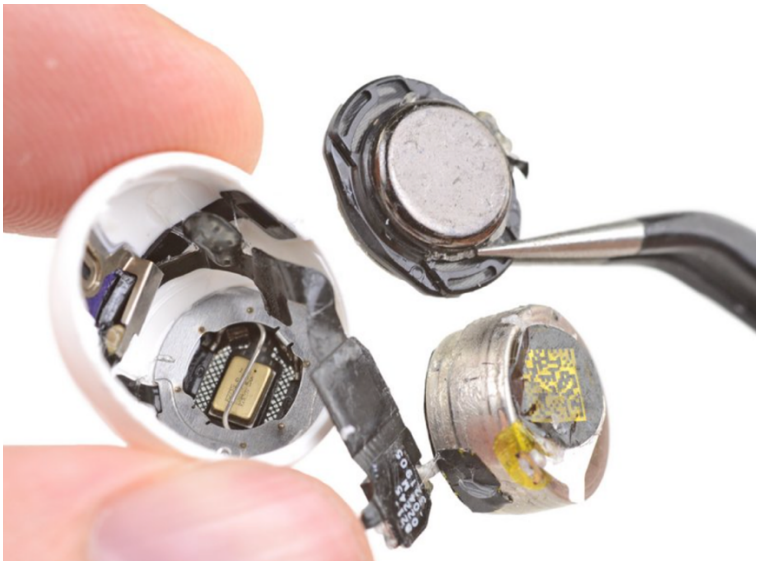


Figure 2.2.1: AirPods Pro Teardown[2]

In Figure 2.2.1, there is one microphone facing inside of the ear for noise cancellation in the AirPods Pro. The other two microphones are used for speech.

For custom earphone design, one of the examples is ClearBuds[3] from arXivLabs. What is unique about ClearBuds is that it is the first hardware and software system to enhance speech with neural networks. ClearBuds bridges state-of-the-art deep learning for blind audio source separation and in-ear mobile systems by making two key technical contributions, including a new wireless earbud design capable of operating as a synchronized, binaural microphone array, and a lightweight dual-channel speech enhancement neural network that runs on a mobile device.

As Apple does not allow the users to record using the in-ward facing microphone, most ear-health works use a plug in in-ear microphone[19] in conjunction with off-the-shelf earbuds to collect in-ear data[18].

2.2 Current Mic Array

The most common MEMS microphone array used for research is the ReSpeaker 4-Mic Array[4] and ReSpeaker 6-Mic Array[5]. The ReSpeaker 4-Mic Array (shown in Figure 2.3.1 and 2.3.2) has four analog MEMS microphones and AC108 Audio codec, specifically designed for the voice interface for Raspberry Pi.

As shown in Figure 2.3.1, an analog MEMS microphone is mounted on each corner of the ReSpeaker to receive audio data around then transmit them to the codec. The codec integrates synchronized ADCs with the mic boost amplifier, to deliver valid channel data to the Raspberry Pi/ Pi Zero, where it transits them over I2S ports by standard I2S or PCM format, along with a single port in TDM format.

AC108 Audio codec is designed for the multi-microphone array in voice capture applications. It is an integrated quad-channel ADC with I2S/TDM transition output, which means it can transit four-channel output data over two I2S ports with standard I2S or PCM format. The device also has DSP features, including a high-pass filter, mixer, and volume control.

However, the ReSpeaker is not enough to satisfy our requirements because the distance between every two microphones is 5.9cm, which means the size of the ReSpeaker is too large that it will not fit in ears. Therefore, we need to design our own microphone array with smaller sizes.

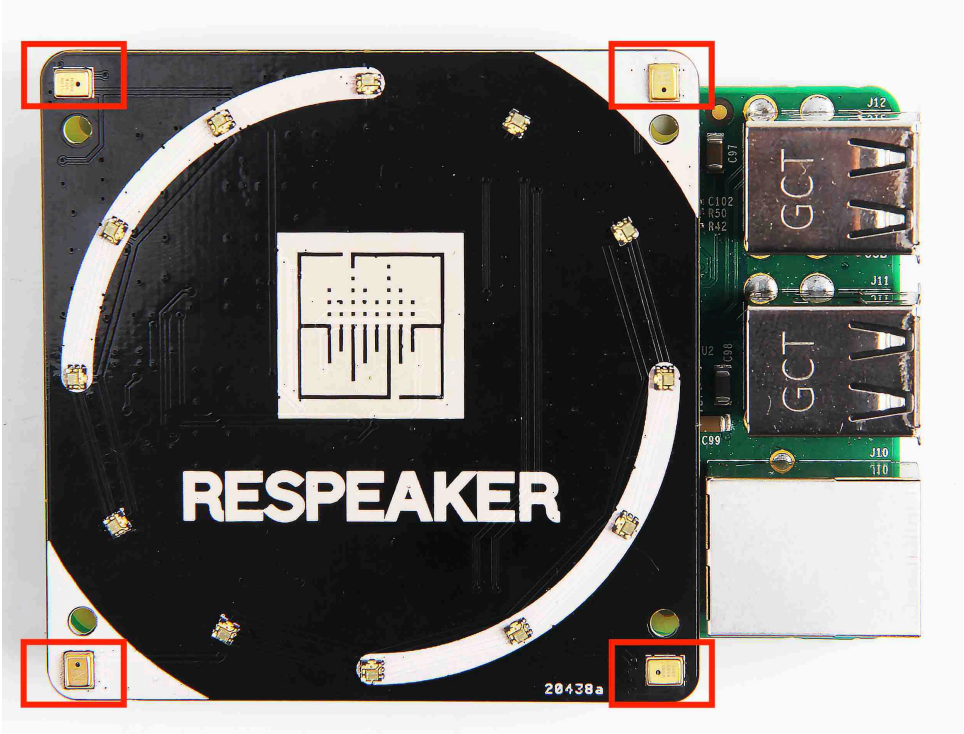


Figure 2.3.1: Top of ReSpeaker 4-Mic Array with MEMS Microphone on the Corners[4]

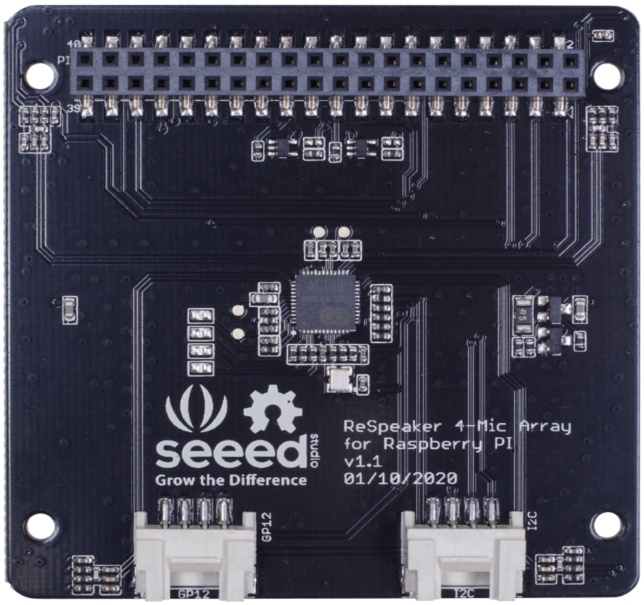


Figure 2.3.2: Bottom of the ReSpeaker 4-Mic Array with AC108[4]

3. Design Questions

Developing an in-ear microphone array may have interdisciplinary challenges and new areas to discover. In particular, the following design questions need to be answered.

Q1. How many microphones are needed?

In general, the more microphones that are used in an audio beamforming system, the more accurate the system will be. The number of microphones used in a beamforming array can vary depending on the application and the desired performance. Regarding the minimum number of microphones required for audio beamforming, it is generally accepted that at least two microphones are required to achieve directional sensitivity. However, a two-microphone array has limitations in terms of understanding the degree of arrival of sound and cannot resolve front-back ambiguity. To address this issue, a minimum of three microphones is typically required[31].

Q2. What is the maximum size of the platform to be able to collect in-ear information?

Based on the actual size of AirPods Pro and Bose Soundsport Pulse wireless[1], the maximum size of the platform to collect in-ear information is about 1.5cm or 1.7cm.

Q3. What is the minimum required distance between 2 microphones to preserve the phase information?

In audio beamforming applications, the distance between microphones plays a crucial role in achieving accurate and effective sound localization. The distance between microphones determines the spatial resolution of the beamforming system, where a shorter distance results in higher resolution but a smaller coverage area, while a longer distance provides wider coverage but lower resolution. Typically, the distance between microphones is based on the wavelength of the sound being detected, with a distance of half the wavelength providing optimal performance[25]. However, other factors such as the frequency range, ambient noise levels, and the desired beamforming algorithm also play a role in determining the optimal microphone spacing.

In a microphone array with a sampling rate of 44100Hz, the minimum required distance between two microphones can be approximated using the following equation:

$$V = f \times \lambda \tag{1}$$

Here, V is the velocity of sound, f is the sampling frequency and λ is the wavelength. As we know that to have phase difference between two receivers, the distance between them needs to be half of the wavelength[28], we can write the distance d as following.

$$d = \frac{\lambda}{2} \tag{2}$$

From Equation (1) and (2) we can write the following

$$V = 2 \times f \times d$$
$$d = \frac{V}{2f} \tag{3}$$

Using Equation (3) we can determine the minimum distance between two microphones for a sampling frequency of 44100 Hz.

$$d = \frac{340\ ms^{-1}}{2 \times 44100} = \ 0.00385\ m = 0.385\ cm$$

As 0.385 cm is smaller than the size of the in-ear mic, we can use this sampling rate and distance combination.

Q4. What is the minimum required sampling rate?

The size of the platform is about 1.5cm to 1.7cm. Any lower sampling rate results in an even larger distance between the microphones. Thus, following the answer of Q3 and using Equation (3), the minimum sampling frequency needs to be 8000Hz.

Q5. Which embedded platforms and protocol supports the sampling rate for multiple microphones?

To support the sampling rate for multiple microphones, the multi-channel ADC with I2S protocol supports the sampling rate for multiple microphones, including AC108 and PCM1865, is required. The highest sampling rate of I2S is 192 kHz[30]. The embedded platforms that can support the sampling rate include Raspberry Pi and ARM Cortex M4.

Q6. What should be the arrangement of the microphones?
The arrangement of microphones for audio beamforming depends on the specific application and the desired beamforming algorithm. However, there are a few common arrangements that are commonly used[25]:

1. Linear array: In this arrangement, microphones are placed in a line. This is a simple and commonly used arrangement. The beamforming algorithm can use the time differences of arrival (TDOA) between the microphones to determine the direction of the sound source.

2. Circular array: In this arrangement, microphones are placed in a circle. This arrangement is useful when the sound source is omnidirectional or when the direction of the source is not known. The beamforming algorithm can use the phase differences between the microphones to determine the direction of the sound source.

3. Planar array: In this arrangement, microphones are placed on a planar surface, such as a flat board or a wall. This arrangement is useful when the sound source is located in a specific plane, such as a person speaking in front of a podium. The beamforming algorithm can use the time and phase differences between the microphones to determine the direction of the sound source.

4. 3D array: In this arrangement, microphones are placed in a 3D space. This arrangement is useful when the sound source can be located in any direction. The beamforming algorithm can use the time and phase differences between the microphones to determine the direction of the sound source.

In general, the number and spacing of microphones in the array will affect the resolution and accuracy of the beamforming. A larger array with smaller microphone spacing will provide higher resolution and accuracy but will also require more processing power.

3.1 Design Variables

Below includes the design variables that need to be considered in the hardware design because these parameters are deeply related to the spatial information important for beamforming or sound source localization, and they determine the specs of final hardware design:

1. d: distance
2. c: speed of sound in air(343 m/sec at 20°C)
3. t: time

4. n: number of samples of delay in DSP.
5. t_D: time delay
6. f_S: sampling frequency
7. f_{NULL}: frequency of the null

The relationships between the variables include:

1. $d=c\times t$ (Distance Sound Travels in a Specified Time)
2. $d = n \times c/f_S$ (Microphone Spacing to Match an n-Sample Delay)
3. $t_D = n/f_S$ (Time Delay for an n-Sample Delay)
4. $f_{NULL} = 1/2 \times c/d$ (On-Axis Null Frequency in a Differential Array)

4 Hardware Design and Prototyping

This section presents three hardware designs, data acquisition processes, and design challenges.

4.1 First Hardware Design

The first hardware design started with the breadboard for prototyping. Based on the original goal, there are four Digital MEMS microphones, and each Digital MEMS microphone has an individual PCB instead of being integrated into the same PCB. Due to time constraints, the hardware design team aims to make fast prototyping.

4.1.1 Component Overview

Communication Protocol. Communication protocol refers to the set of rules and standards that govern the exchange of information between two or more devices. It defines how data is transmitted over a network, including the format of messages, the sequencing of message exchange, error checking, and other parameters. Communication protocols are essential in enabling different devices to communicate with each other and exchange data seamlessly. They are widely used in various industries, including telecommunications, computer networking, and the internet. Understanding communication protocols is crucial for ensuring efficient and effective communication between devices. The main communication protocol we are going to use is I2S.

I2S (Inter-IC Sound) provides a simple and efficient way to transmit high-quality digital audio signals between devices. It is a communication protocol used to transmit digital audio data between integrated circuits.

I2S is a widely used standard for transmitting digital audio signals in consumer electronics. It uses a three-wire interface consisting of a data line, a word clock, and a bit clock to transmit digital audio data. The data line carries the audio samples, the word clock synchronizes the transmission of each audio sample, and the bit clock provides the timing for the transmission of each bit of the audio data. I2S is also capable of transmitting audio data with up to 24 bits of resolution and at sample rates up to 192 kHz. I2S is commonly used in audio applications such as

digital audio interfaces, digital signal processors, and audio codecs. It is also used in many microcontroller-based audio applications, including audio playback and recording systems.

Microphone: The main component of the first prototype is the ICS-43432 I2S Digital MEMS Microphone. According to the ICS-43432 Datasheet, the ICS-43432 has a high SNR of 65 dBA and a wideband frequency response from 50Hz to 20kHz; the sensitivity tolerance of the ICS-43432 is ±1 dB, which enables high-performance microphone arrays without the need for system calibration; the surface mount package is in 4 mm × 3 mm × 1 mm.

Embedded System. The second component, Raspberry Pi 4[11], is used for data acquisition from digital MEMS microphone. The Raspberry Pi 4 is a powerful and versatile single-board computer developed by the Raspberry Pi Foundation. The Raspberry Pi 4 is powered by a Broadcom BCM2711 quad-core processor with speeds up to 1.5 GHz, and it comes with up to 8 GB of RAM, making it more than capable of running a variety of applications and projects. It also features dual HDMI ports, which allow for dual display output, and supports 4K video playback. In addition to its powerful hardware, the Raspberry Pi 4 is also known for its wide range of connectivity options. It includes Gigabit Ethernet, dual-band 802.11ac wireless, Bluetooth 5.0, and two USB 3.0 ports, as well as two USB 2.0 ports, a 40-pin GPIO header, and a microSD card slot.

The Raspberry Pi 4 includes support for the I2S (Inter-IC Sound) bus, a communication protocol used to transmit digital audio data between integrated circuits. This makes it possible to use the Raspberry Pi 4 as a platform for audio applications, such as audio playback, audio recording, and audio processing.

To use the I2S bus on the Raspberry Pi 4, users must first enable the I2S driver and configure the GPIO pins for the I2S interface. This can be done through the Raspberry Pi configuration tool or by modifying the device tree overlay file.

Once the I2S interface is configured, users can connect compatible audio devices, such as audio codecs, digital signal processors, and audio amplifiers, to the Raspberry Pi 4 and begin transmitting digital audio data. The Raspberry Pi 4 supports I2S audio data with up to 32 bits of resolution and at sample rates up to 192 kHz. Because of the high sample rates and high resolution in the I2S interface, Raspberry Pi 4 is one of the best choices for the in-ear microphone array design.

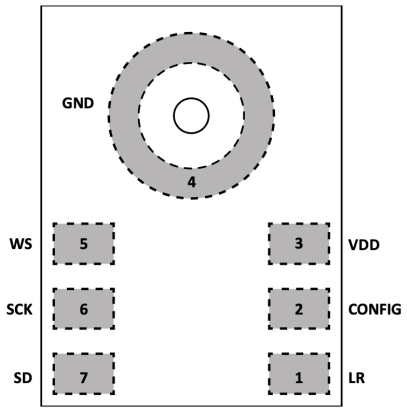


Figure 4.1.1.1: Pin Configuration of ICS-43432 (Top View, Terminal Side Down)[7]

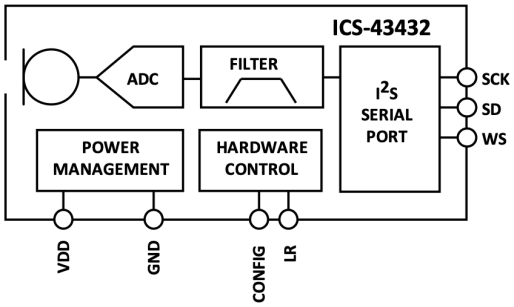


Figure 4.1.1.2: Functional Block Diagram of ICS-43432[7]

PIN	NAME	TYPE	FUNCTION
1	LR	Input	Left/Right channel select. When set low, the microphone outputs its signal in the left channel of the I2S frame. When set high, the microphone outputs its signal in the right channel.
2	CONFIG	Input	Pull to ground. The state of this pin is used at power-up.
3	VDD	Power	Power, 1.62 to 3.63 V. This pin should be decoupled to GND with a 0.1 µF capacitor.
4	GND	Ground	Ground. Connect to ground on the PCB.
5	WS	Input	Serial Data-Word Select for I2S Interface
6	SCK	Input	Serial Data Clock for I2S Interface
7	SD	Output	Serial Data Output for I2S Interface. This pin tri-states when not actively driving the appropriate output channel. The SD trace should have a 100 kΩ pulldown resistor to discharge the line during the time that all microphones on the bus have tri-stated their outputs.

Table 4.1.1.3: Pin Function Descriptions[7]

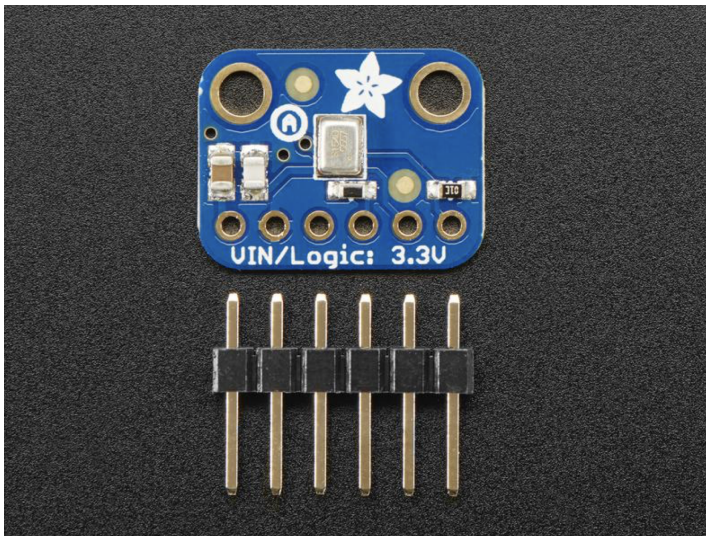


Figure 4.1.1.4: Single ICS-43432 I2S MEMS Microphone in PCB[8]

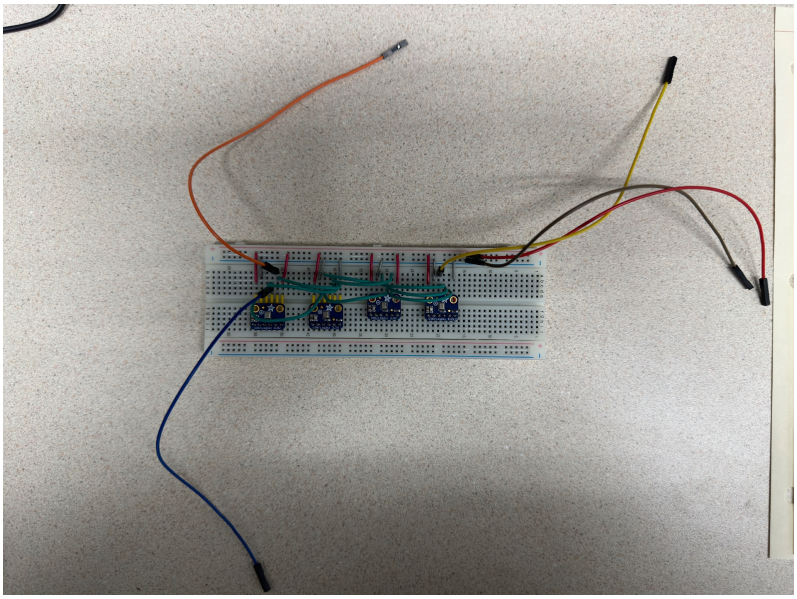


Figure 4.1.1.5: Breadboard Prototype with Four I2S MEMS Microphones

4.1.2 Prototype and Data Acquisition

Prototype design, shown in Figure 4.1.1.5, will also be modified based on the test. The following shows the procedures to record data from the Adafruit I2S MEMS Microphones[8] with Raspberry Pi.

Materials Needed:

- Raspberry Pi board
- Adafruit I2S MEMS Microphone Breakout
- Jumper wires

- MicroSD card with Raspbian installed
- USB microphone or speakers with 3.5mm jack (optional)

We have performed the following steps to collect data from two microphones using a Raspberry Pi[8]:

1. Install Python 3 and Pip on Raspberry Pi with command “sudo apt install python3-pip”
2. Install scripts with the following commands:

```
cd ~
sudo pip3 install --upgrade adafruit-python-shell
wget https://raw.githubusercontent.com/adafruit/Raspberry-Pi-Installer-Scripts/master/i2smic.py
sudo python3 i2smic.py
```

3. Connect the Adafruit I2S MEMS Microphone Breakout to your Raspberry Pi board using the jumper wires. The connections, shown in Figure 4.1.2.1, should be made as follows:

    - VIN (microphone) -> 3.3V (Raspberry Pi)
    - GND (microphone) -> GND (Raspberry Pi)
    - BCLK (microphone) -> BCM 18 (Raspberry Pi)
    - LRCL (microphone) -> BCM 19 (Raspberry Pi)
    - DOUT (microphone) -> BCM 21 (Raspberry Pi)
    - SEL (microphone) -> BCM 12 (Raspberry Pi)

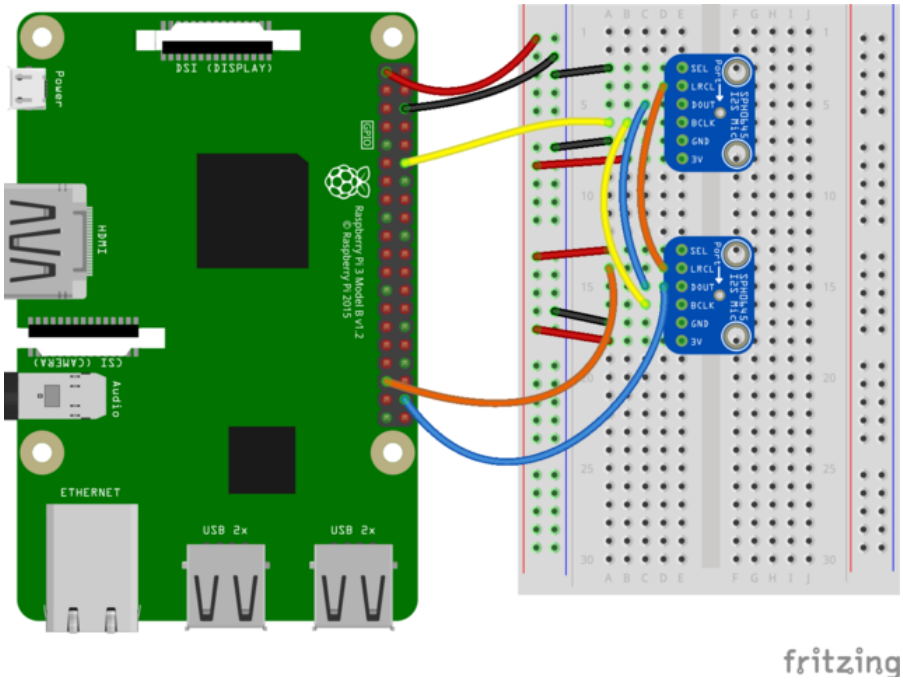


Figure 4.1.2.1: Circuit Diagram for Raspberry Pi and Adafruit I2S MEMS Microphone[8]

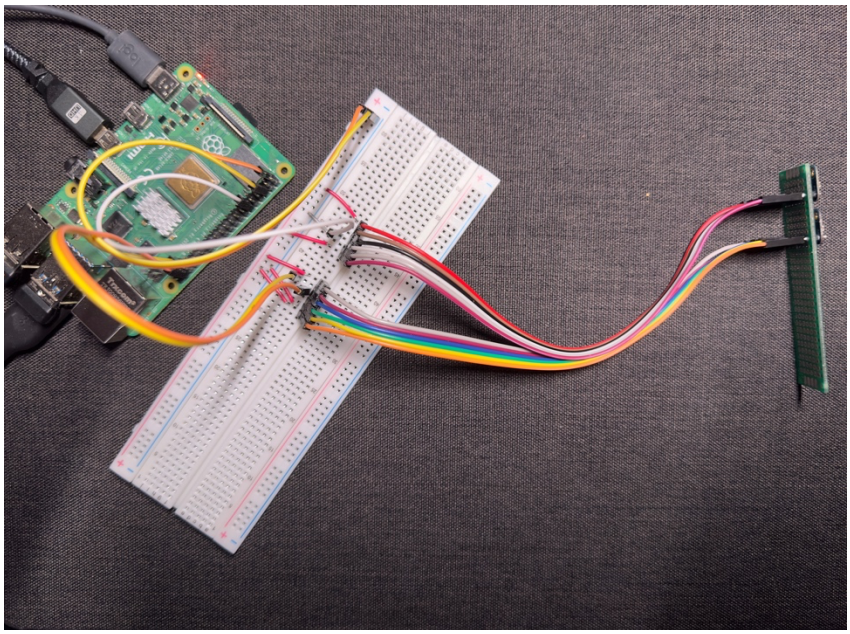


Figure 4.1.2.2: Two I2S MEMS Microphone to Raspberry Pi

```
pi@raspberrypi:~ $ arecord -l
**** List of CAPTURE Hardware Devices ****
card 0: sndrpii2scard [snd_rpi_i2s_card], device 0: simple-card_codec_link snd-soc-dummy-dai-0 [simple-card_codec_link sn
d-soc-dummy-dai-0]
  Subdevices: 1/1
  Subdevice #0: subdevice #0
pi@raspberrypi:~ $ arecord -D plughw:0 -c2 -r 48000 -f S32_LE -t wav -V stereo -v file_stereo.wav
Recording WAVE 'file_stereo.wav' : Signed 32 bit Little Endian, Rate 48000 Hz, Stereo
Plug PCM: Hardware PCM card 0 'snd_rpi_i2s_card' device 0 subdevice 0
Its setup is:
  stream       : CAPTURE
  access       : RW_INTERLEAVED
  format       : S32_LE
  subformat    : STD
  channels     : 2
  rate         : 48000
  exact rate   : 48000 (48000/1)
  msbits       : 32
  buffer_size  : 24000
  period_size  : 6000
  period_time  : 125000
  tstamp_mode  : NONE
  tstamp_type  : MONOTONIC
  period_step  : 1
  avail_min    : 6000
  period_event : 0
  start_threshold  : 1
  stop_threshold   : 24000
  silence_threshold: 0
  silence_size : 0
  boundary     : 1572864000
  appl_ptr     : 0
  hw_ptr       : 0
                                + 05%|05%#+                                ^CAborted by signal Interrupt...
                                + 05%|05%#+                                pi@raspberrypi:~ $ arecord -D plughw:0 -c2
-r 48000 -f S32_LE aplay file_stereo.wav
Playing WAVE 'file_stereo.wav' : Signed 32 bit Little Endian, Rate 48000 Hz, Stereo
```

Figure 4.1.2.3: Audio Recording Procedures

4. In the audio recording procedures shown in Figure 4.1.2.3, the command “arecord -l” is for listing available devices. Based on the card number “0”, mono audio can be recorded with the command “arecord -D plughw:0 -c1 -r 48000 -f S32_LE -t wav -V mono -v file.wav”, and stereo audio with command “arecord -D plughw:0 -c2 -r 48000 -f S32_LE -t wav -V stereo -v file_stereo.wav”. The sampling rate is 48000Hz, and the data is in 32 bit. Figure 3.3.5 shows an example recording command.

5. The command “aplay file_stereo.wav” is for playing the file back directly on the device. The wav file generated can also be saved in the Raspberry Pi with command “scp pi@raspberrypi:file.wav ~/Desktop/file.wav”. Figure 4.1.2.4 shows an example recording.

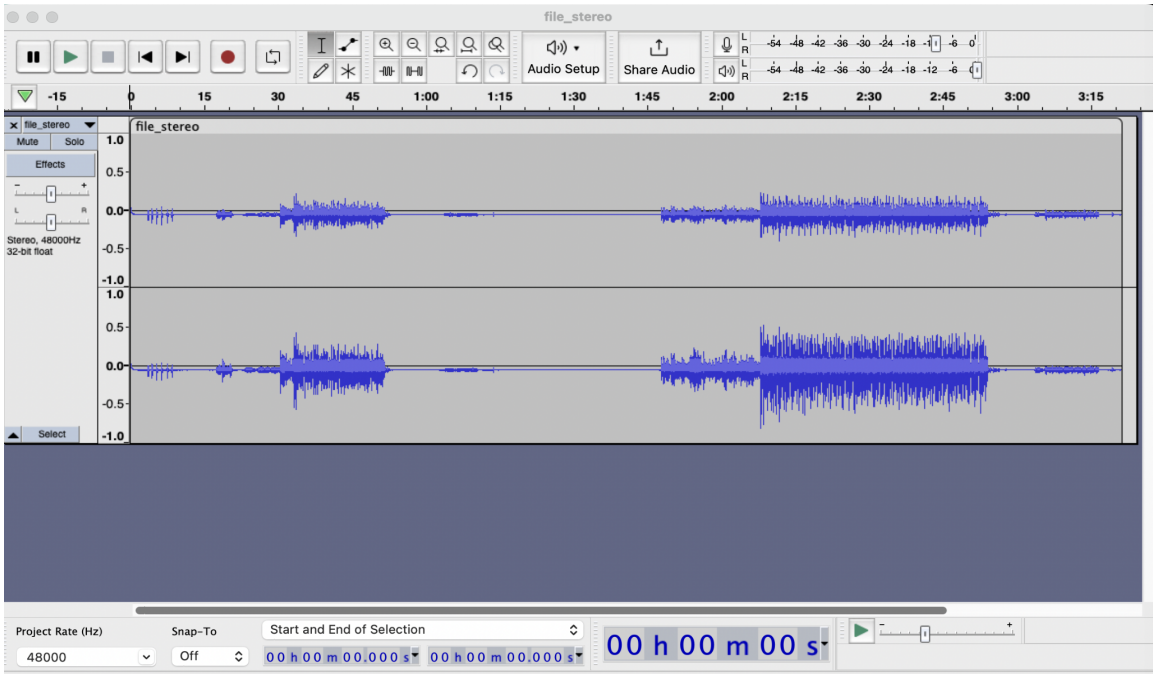


Figure 4.1.2.3: Recorded Stereo Audio WAV File in Audacity

4.1.3 Design Challenge

Limited I2S in Raspberry Pi. The first design challenge for the first hardware design is recording data from four microphones instead of two microphones, which is due to the pin configuration of digital MEMS microphone, which has the pin for Left/Right channel select. When set low, the microphone outputs its signal in the left channel of the I2S frame. When set high, the microphone outputs its signal in the right channel. However, there is only one I2S bus on Raspberry Pi, which sets barriers to transferring data from 4 I2S digital MEMS microphones to Raspberry Pi.

Breadboard Noise. The second design challenge is reducing noise from the prototype based on the breadboard; the mounting holes on the breadboard are too loose, which will cause noises in the circuit prototype. This issue can be solved by building the circuit on the protoboard.

Larger Size. The third challenge is the size of the prototype. Due to timing constraints, we only developed a breadboard version and the distance between the microphones is 1.5cm and the total mic-array size is 2.8cm × 1.6cm. Though we have only answered Q4 and Q5 in this question this is the first step towards developing the target prototype.

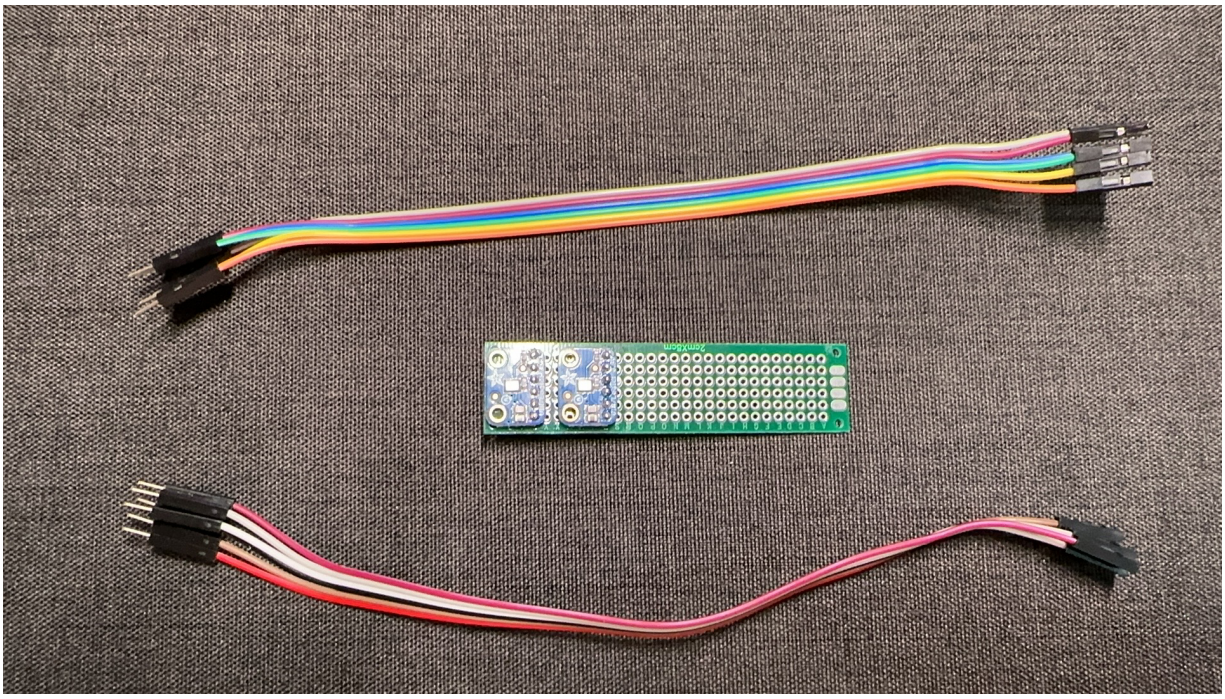


Figure 4.1.3.1: Two I2S Digital MEMS Microphones on Protoboard

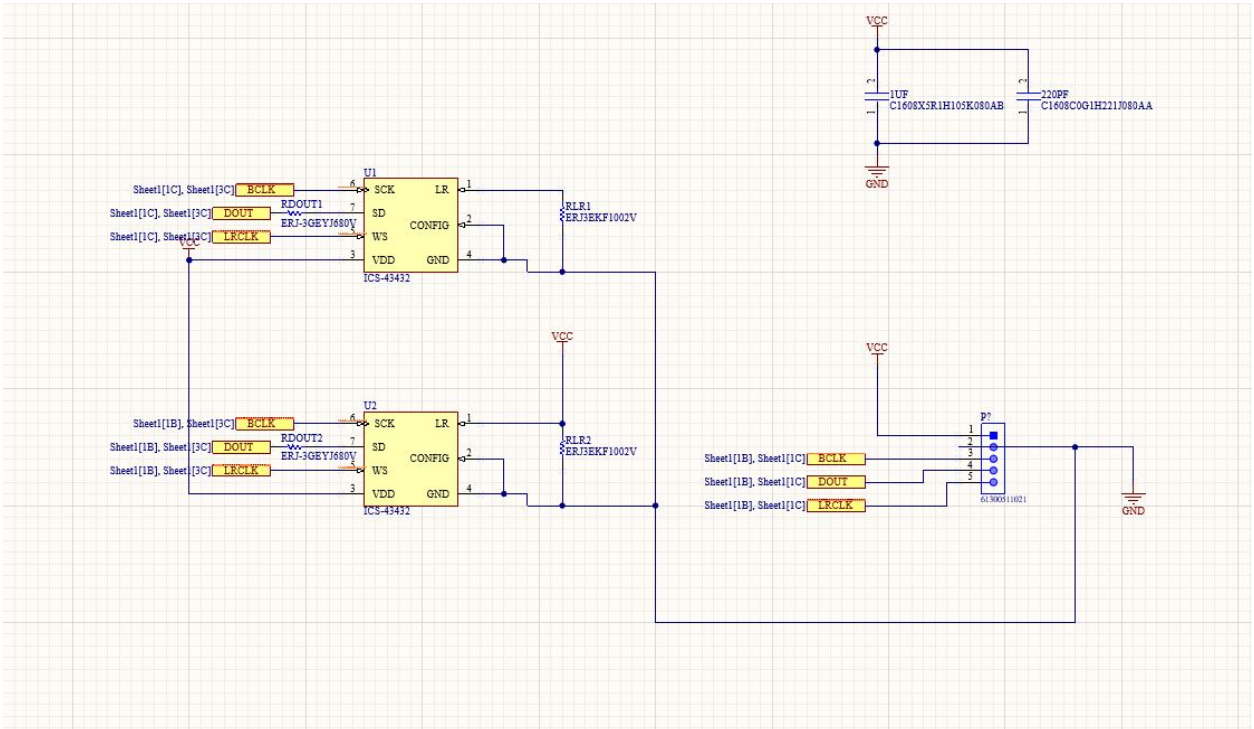


Figure 4.1.3.2: First PCB Schematic (Digital 2-Mic)

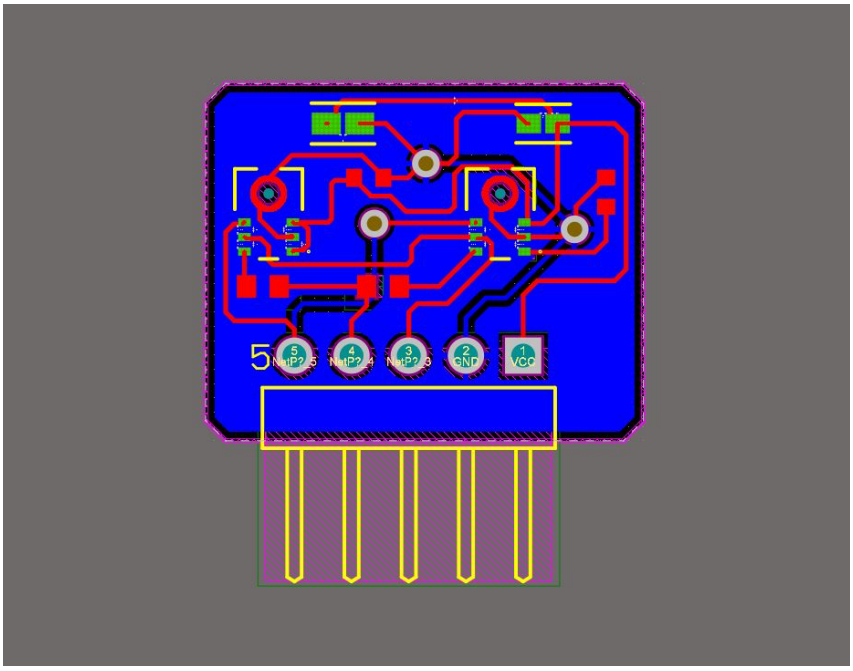


Figure 4.1.3.3: PCB Design (Digital 2-Mic)

Figure 4.1.3.2 shows the PCB schematic for digital 2-mic array, and Figure 4.1.3.3 shows the PCB layout design. Both the schematic and the layout are designed by our own with Altium Designer.

4.2 Second Hardware Design

Considering the time constraints, the second hardware design aims to solve the problem of limitation in the number of microphones in the first design. Different from the first design with digital I2S microphones, the second design of hardware is based on analog microphones. Every single module used in all prototypes has integrated both a MAX4466 preamplifier and an analog MEMS microphone. The second hardware started from a prototype from the breadboard.

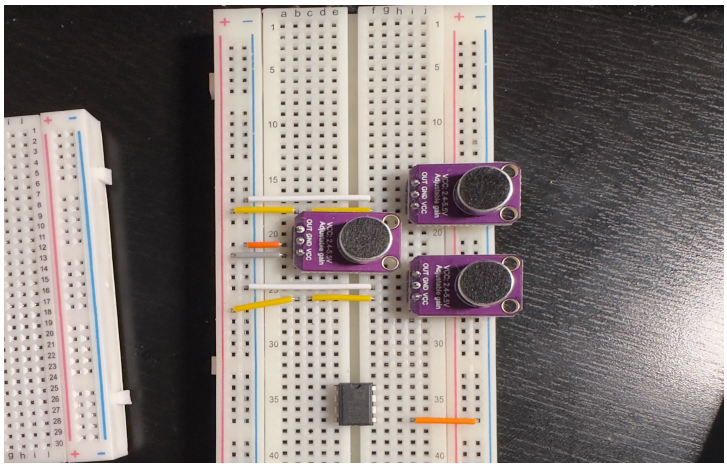


Figure 4.2.1: Breadboard Prototype with Three Analog MEMS Microphones

4.2.1 Component Overview

Microphone. The MAX4465-MAX4469[9] is a series of microphone preamplifiers designed for use in low-voltage applications. These preamplifiers can amplify small signals from microphones to a level suitable for further processing. The preamplifiers feature a low-noise input stage with a fixed gain of 20dB or 26dB, depending on the specific model. They have a wide frequency response range of 20Hz to 20kHz, making them suitable for a variety of audio applications. The MAX4465-MAX4469 preamplifiers operate from a single supply voltage between 2.4V and 5.5V, which makes them suitable for battery-powered applications. They also have a low quiescent current of only 20µA. The preamplifiers are available in small packages that are easy to integrate into a wide range of electronic devices. They also have a shutdown pin that can be used to turn off the device when it is not in use, further reducing power consumption. Therefore, the MAX4465-MAX4469 microphone preamplifiers are a versatile and low-power solution for amplifying microphone signals in a variety of low-voltage applications.

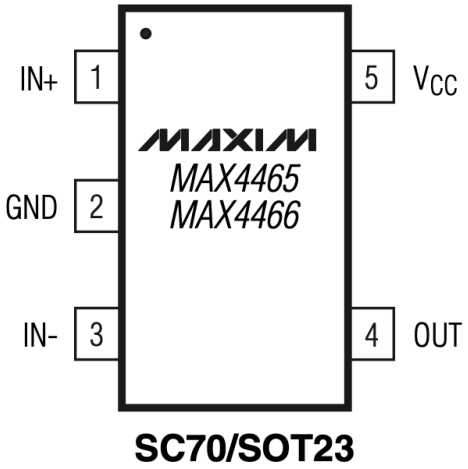


Figure 4.2.1.1: MAX4465/MAX4466 Pin Configurations[9]

Pin	Name	Function
1	IN+	Noninverting Amplifier Input
2	GND	Ground
3	IN-	Inverting Amplifier Input
4	OUT	Amplifier Output
5	Vcc	Supply Voltage

Table 4.2.1.1: Pin Function Descriptions[9]

4.2.2 Prototype and Data Acquisition

The development boards used for data acquisition are Arduino Uno and Raspberry Pi 4 because both hardware and software teams only have these two development kits. Arduino Uno has six on-board ADC channels which can be used to read analog signals, and Raspberry Pi has both SPI and I2S interfaces.

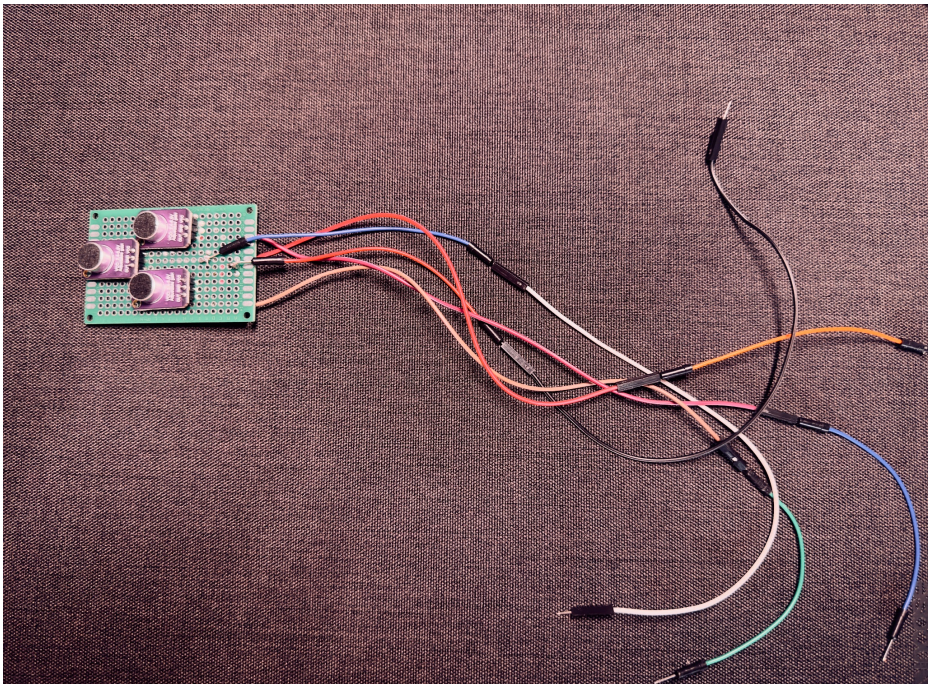


Figure 4.2.2.1: Three Analog MEMS Microphones on Protoboard (Analog 3-Mic Array)

Test and Data Acquisition with Arduino Uno

The first test method is based on Analog 3-Mic Array, which is shown above, and Arduino Uno[10]. There are five analog pins for ADC, 3 power pins, and three ground pins on Arduino Uno. Three analog pins, one 3V3 pin, and one ground pin are used to gather data.

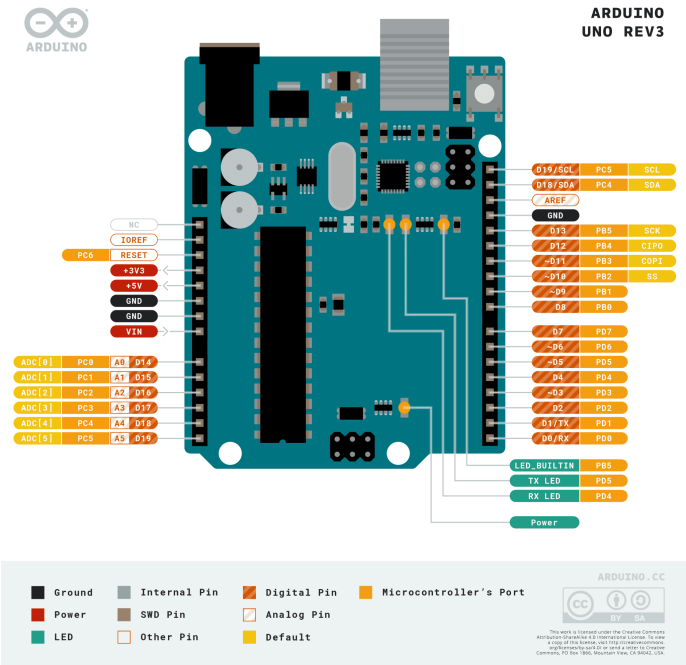


Figure 4.2.2.2: Arduino Uno Pinout[10]

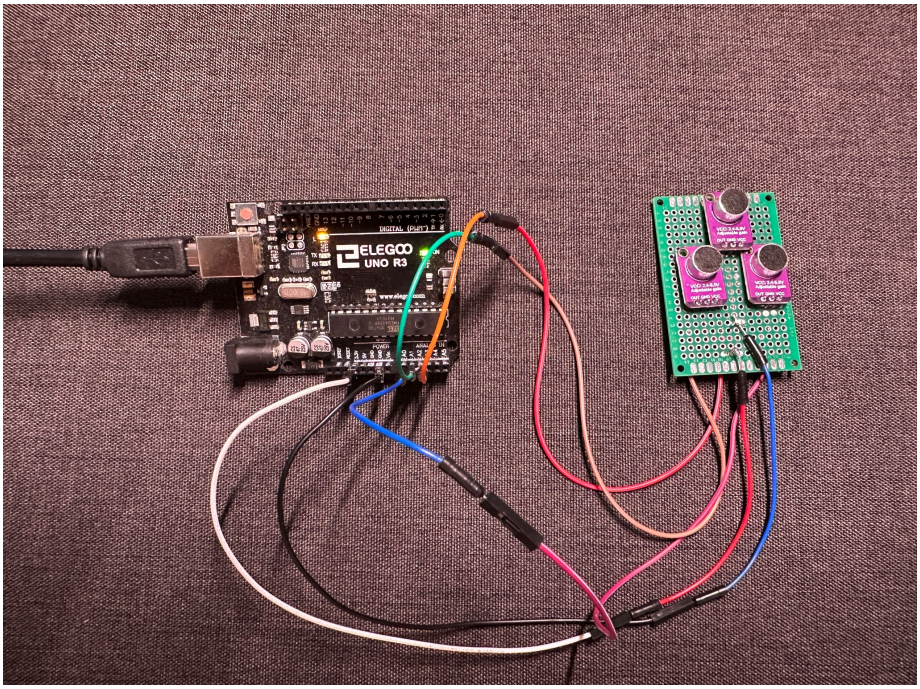


Figure 4.2.2.3: Test with Analog 3-Mic Array and Arduino Uno

```
/*Read the analog input of three microphone */
unsigned long myTime;

int micIN1= A0; //Analog input from Pin A0 on Arduino
int micIN2= A1; //Analog input from Pin A1 on Arduino
int micIN3= A2; //Analog input from Pin A0 on Arduino
int audioVal1 = 0; //Initialize value gathered from A0
int audioVal2 = 0; //Initialize value gathered from A1
int audioVal3 = 0; //Initialize value gathered from A2
void setup() {
  Serial.begin(9600);
}
void loop() {
  audioVal1 = analogRead(micIN1);
  audioVal2 = analogRead(micIN2);
  audioVal3 = analogRead(micIN3);

  myTime = millis();

  Serial.print(audioVal1);
  Serial.print(",");
  Serial.print(audioVal2);
  Serial.print(",");
  Serial.print(audioVal3);
  Serial.print(",");
  Serial.print("Time:");
  Serial.println(myTime);
}

```

Figure 4.2.2.4: Sketch and Code for Data Acquisition on Arduino IDE

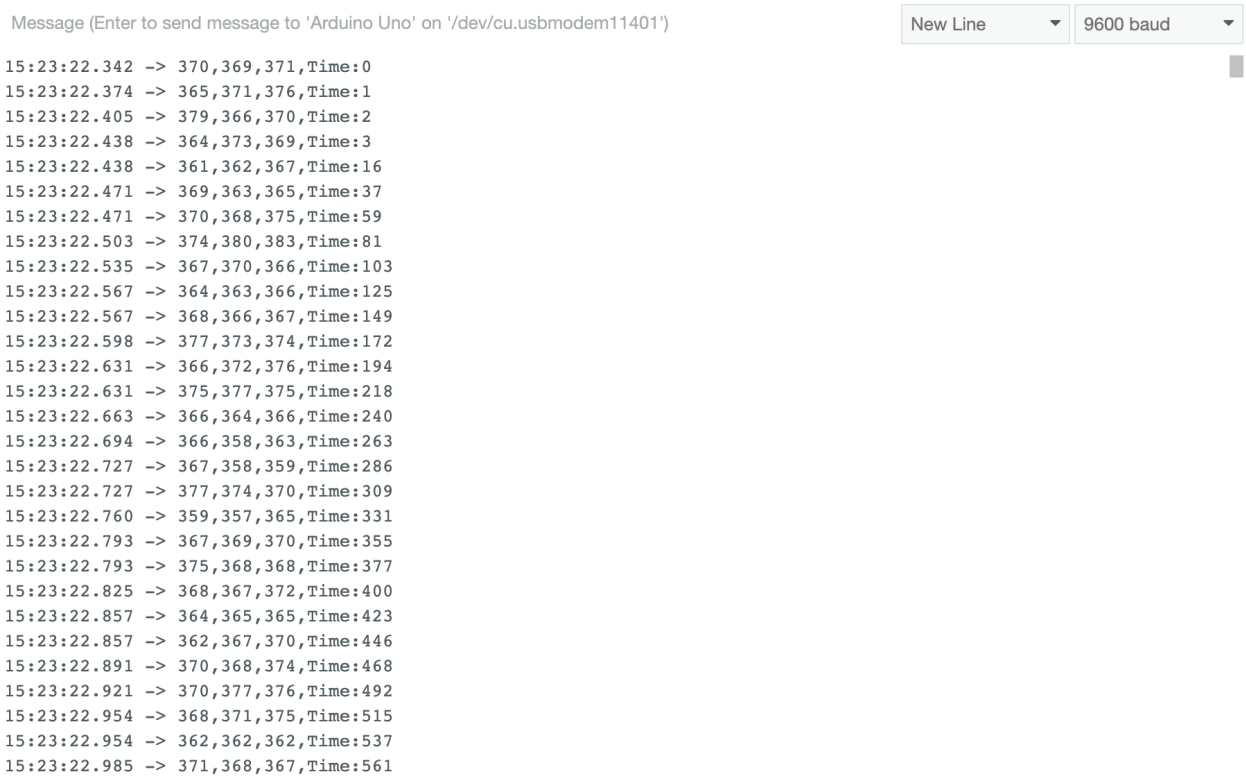


Figure 4.2.2.5: Data Acquisition with Serial Monitor on Arduino IDE

In Figure 4.2.2.5, there are data gathered from the analog 3-mic array, and time shows the number of milliseconds at the time since the program is uploaded to Arduino Uno.

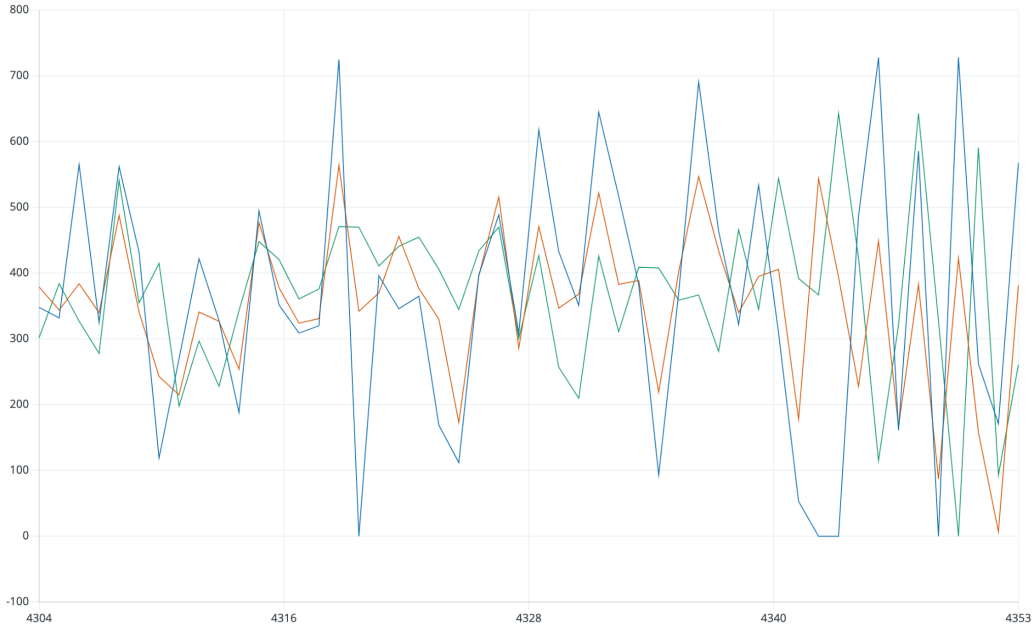


Figure 4.2.2.6: Data Acquisition with Serial Plotter on Arduino IDE

Based on Data Acquisition, from the time of 16 milliseconds to 218 milliseconds, there are 10 samples gathered from the analog 3-mic array. Below is the process to calculate the sampling rate:

$$t1 = 16\ msec,\ t2 = 218\ msec,\ n = 10\ samples$$
$$\Delta t = t2 - t1 = 218\ msec - 16\ msec = 202\ msec$$
$$fs = n/\Delta t = 10\ samples/202\ msec \approx 49\ Hz$$

The sampling rate of 49 Hz is extremely low for data acquisition in audio applications, which is also displayed in the waveform with sharp edges from Figure 3.4.7. Therefore, we chose Raspberry Pi to solve the issue of the low sampling rate.

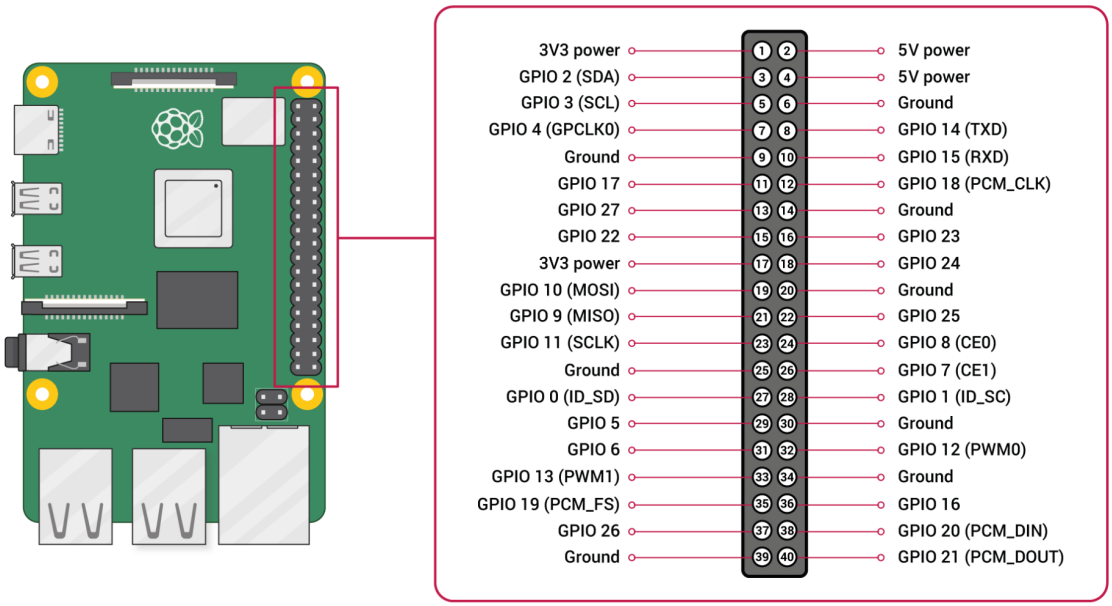


Figure 4.2.2.7: Raspberry Pi 4 Pinout[11]

Communication Protocol. Based on Figure 4.2.2.7, there is only one I2S bus on Raspberry Pi[11]. Therefore, the second attempt for test and data acquisition with the analog 3-mic array is based on the SPI communication interface on Raspberry Pi due to time constraints. The effect of time constraints will be described in the next sections.

Analog to Digital Converter. To transfer data from the analog 3-mic array to Raspberry Pi with both ADC (analog-to-digital converter) and SPI interface, MCP3008 is selected.

MCP3008[12] is a popular 10-bit ADC chip that can be used to convert analog signals into digital signals for processing by a microcontroller or computer. It offers 8 channels for analog input, which satisfies the requirement to convert and transfer data from the analog 3-mic array. Besides, MCP3008 has a simple SPI (Serial Peripheral Interface) communication protocol, which makes it easy to integrate with a wide range of microcontrollers and computers. It also offers a wide input voltage range from 0V to 5V, which makes it suitable for a variety of applications, including sensor data acquisition, and audio signal processing.

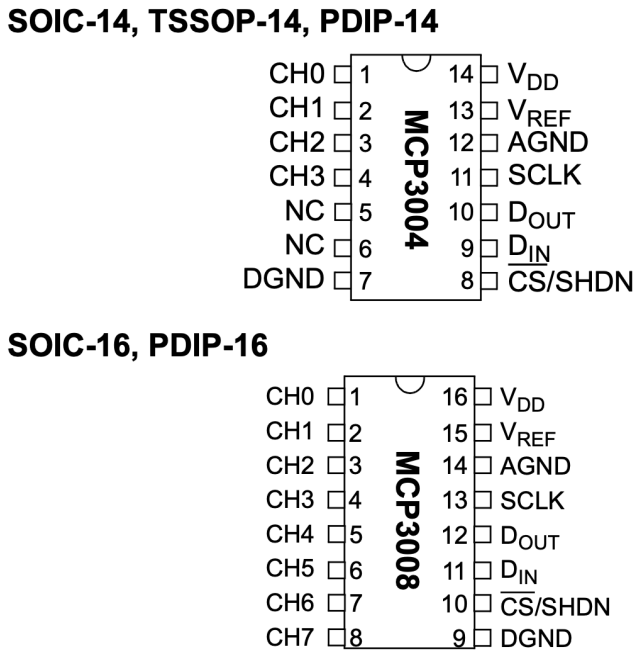


Figure 4.2.2.8: Pinout and Package for MCP3004/3008[12]

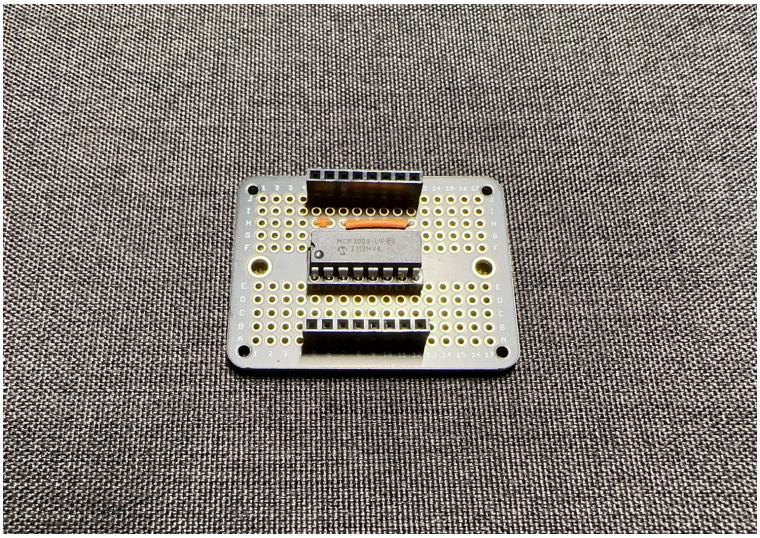


Figure 4.2.2.9: MCP3008 in Protoboard

```python
#!/usr/bin/python
import spidev
import os
import time
import numpy as np
from collections import deque
import wave

# Define Audio Channels (channel 3 to 7 can be assigned for more sensors)
micone_channel = 0
mictwo_channel = 1
micthree_channel = 2
#Time delay, which tells how many seconds the value is read out, for prior testing
#delay = 0.1

# Spi open
spi = spidev.SpiDev()
spi.open(0,0)
spi.max_speed_hz=1000000
# Function for reading the MCP3008 channel between 0 and 7
def readChannel(channel):
  val = spi.xfer2([1,(8+channel)<<4,0])
  data = ((val[1]&3) << 8) + val[2]
  return data

start = time.time()

micone_val_list = []

# endless loop
while len(micone_val_list) <= 1000:
  # Determine value
  micone_val = readChannel(micone_channel)
```

Shell

```
13.067371606826782
micOne : 129  micTwo : 129  micThree : 129
<class 'int'> <class 'int'> <class 'int'>
926
13.070920944213867
micOne : 128  micTwo : 129  micThree : 125
```

Figure 4.2.2.10: Data Acquisition with Python in Raspberry Pi OS

Shell

```
micOne : 122  micTwo : 126  micThree : 126
<class 'int'> <class 'int'> <class 'int'>
754
10.914814949035645
micOne : 129  micTwo : 128  micThree : 128
<class 'int'> <class 'int'> <class 'int'>
755
10.918060302734375
micOne : 127  micTwo : 129  micThree : 127
<class 'int'> <class 'int'> <class 'int'>
756
10.921848773956299
micOne : 128  micTwo : 128  micThree : 127
<class 'int'> <class 'int'> <class 'int'>
757
10.926247358322144
micOne : 127  micTwo : 128  micThree : 127
<class 'int'> <class 'int'> <class 'int'>
758
11.023091077804565
micOne : 128  micTwo : 129  micThree : 128
<class 'int'> <class 'int'> <class 'int'>
759
11.026307344436646
micOne : 127  micTwo : 129  micThree : 129
<class 'int'> <class 'int'> <class 'int'>
760
11.029302597045898
micOne : 127  micTwo : 129  micThree : 127
<class 'int'> <class 'int'> <class 'int'>
761
11.03279423713684
micOne : 128  micTwo : 127  micThree : 128
<class 'int'> <class 'int'> <class 'int'>
```

Figure 4.2.2.11: Data Acquisition in Python Shell

In Figure 3.4.12 shown above, there are 754 data collected in the time of 10.914814949035645 seconds and 761 data collected in the time of 11.03279423713648 seconds. Below is the process to calculate the sampling rate:

$$t1 = 10.914814949035645 \text{ sec}, t2 = 11.03279423713648 \text{ sec}, n1 = 754 \text{ samples}, n2 = 761 \text{ samples}$$
$$\Delta t = t2 - t1 = 11.03279423713648 \text{ sec} - 10.914814949035645 \text{ sec} = 0.1179792881 \text{ sec}$$
$$\Delta n = n2 - n1 = 761 \text{ samples} - 754 \text{ samples} = 7 \text{ samples}$$
$$fs = \Delta n/\Delta t = 7\text{samples}/ 0.1179792881 \text{ sec} \approx 59 \text{ Hz}$$

Based on calculations shown above, the sampling rate of 59 Hz is extremely low for data acquisition in audio applications. Therefore, a new design is required to achieve the goal and requirements of high sampling rates and data acquisition from multiple microphones. This issue happens because Raspberry Pi does not support SPI of up to 8000 Hz, which is the frequency for low-quality audio[23]; and there are hardware limitations in Arduino Uno because the ATmega328P microcontroller in Arduino Uno has a limited processing power and memory, which limits the ADC sampling rate. The ADC conversion takes several clock cycles to complete, and the microcontroller must also handle other tasks such as data processing, communication, and input/output operations[32].

4.2.3 Design Challenge

The design challenge for the second design of hardware is an extremely low sampling rate with both Arduino Uno and Raspberry Pi with the SPI communication interface. Therefore, the second design is unable to solve the issues in the first design. To solve the issue of a low sampling rate for data acquisition from three or more microphones, the hardware with both multiple ADC channels and I2S serial bus interface is required. However, most of the ICs (Integrated Circuits) with both multiple ADC channels and I2S are in small packages such as QFN (Flat no-leads package). The hardware design with IC in QFN must be based on PCB (Printed Circuit Board) and receiving the final hardware design product after sending PCB design files takes a huge amount of time, which is a huge challenge with time constraints.

4.3 Analysis of the First and Second Designs of Hardware Design

Based on Section 3.3 and Section 3.4, each type of hardware design has different issues and limitations. Below is the table with the analysis:

Pros (First Design)	Cons (First Design)
The sampling rate of the data is fast, which satisfies the requirement for audio applications and machine learning tasks with model training.	The number of microphones is limited. To achieve more accurate machine learning and model training, three or more microphones are required.
Pros (Second Design)	Cons (Second Design)

The data from three or more microphones can be transferred to the computer.	The sampling rate is extremely low, which is unable to satisfy the software team's requirements of high sample rates and model training with WAV files

Table 4.3.1: Analysis of the First Design and the Second Design

4.4 Third Hardware Design

Based on Section 4.2 and 4.3, the main requirements are data acquisition from three or more microphones and a high sampling rate for model training with WAV files, the design for the third design of hardware is implemented to solve all the issues from both the first and the second hardware designs.

4.4.1 Component Overview

Considering the issues of time constraints, AC108[6], which has a brief introduction in Section 3.2, is integrated into the third hardware design. AC108 has both quad-channel ADC and I2S/TDM interfaces. In this way, the software team can collect data into WAV files in the new compact hardware with the current driver used in their workflow. The third design will be based on Figure 3.6.1. AC108 is in QFN package, which means it is impossible to build the prototypes on both the breadboard and the protoboard because QFN package is too small.

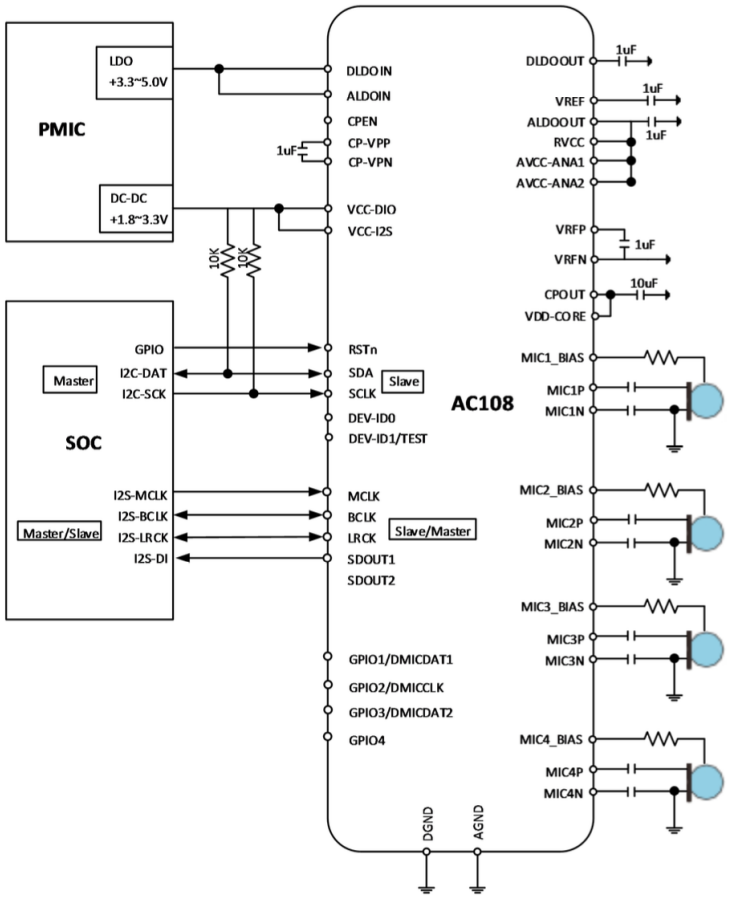


Figure 4.4.1.1: AC108 Typical Application Diagram from AC108 Datasheet[6]

4.4.2 PCB Library Documents

One of the main challenges in PCB design is creating both the schematic and the layout for AC108. The AC108 is in the QFN package with 48 pins, and there is no PCB layout data in the PCB design software. QFN[22] stands for Quad Flat No-Lead, which is a type of surface-mount packaging for integrated circuits. It is a low-profile package with a square or rectangular shape that has contacts on all four sides, allowing for a high density of connections in a small area. QFN packages are widely used in a variety of electronic devices, including mobile devices and automotive electronics. The main reason most of the ICs for audio applications with both multi-channel ADCs and I2S are in the small QFN package is that the size of the circuitry needs to be reduced to minimize noises and signal distortions from parasitic capacitance, resistance, and inductance. Parasitic capacitance, resistance, and inductance are all undesired effects that can occur in electronic circuits. Parasitic capacitance refers to the capacitance that exists between conductive elements in a circuit that was not intentionally designed to be a capacitor. It can result from the physical proximity of two conductors, such as the leads of a resistor or the traces on a printed circuit board. Parasitic resistance is a similar phenomenon, where resistance is present in a component or connection that was not intended to have resistance. Similarly, parasitic inductance refers to the unintended inductance that exists in a circuit, which can cause issues

such as signal distortion or unwanted oscillation. These parasitic effects can negatively impact the performance of electronic circuits, particularly at high frequencies, and need to be taken into account during circuit design and optimization.

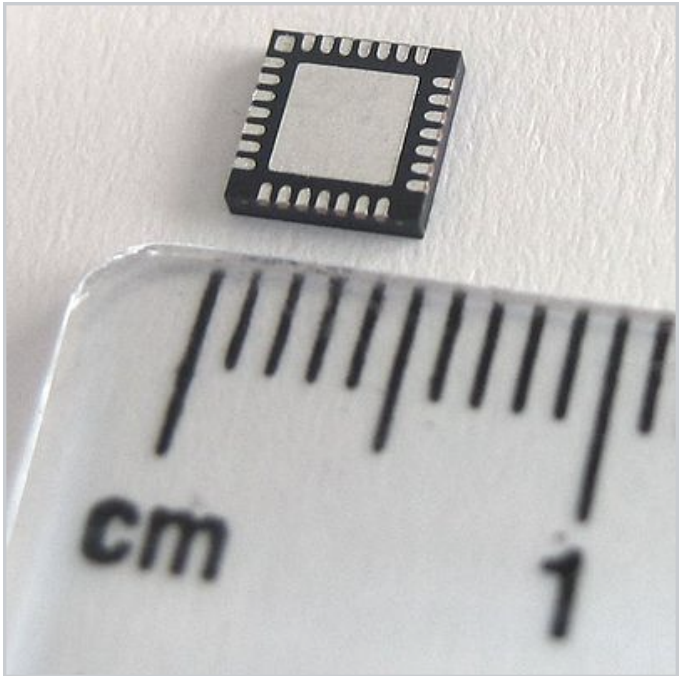
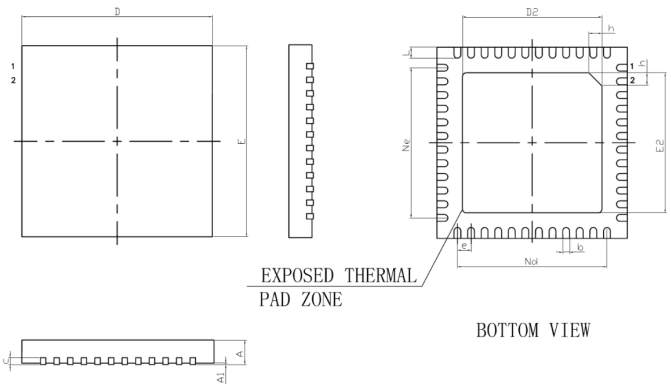


Figure 4.4.2.2: 28-Pin QFN package[13]



SYMBOL	MILLIMETER		
	MIN	NOM	MAX
A	0. 70	0.75	0. 80
A1	—	0.02	0. 05
b	0. 15	0.20	0. 25
c	0. 18	0.20	0. 23
D	5. 90	6. 00	6. 10
D2	4. 10	4.20	4. 30
e	0. 40BSC		
Ne	4. 40BSC		
Nd	4. 40BSC		
E	5. 90	6. 00	6. 10
E2	4. 10	4.20	4. 30
L	0. 35	0.40	0. 45
h	0. 30	0. 35	0. 40
L/F载体尺寸 (MIL)	177*177		

Figure 4.4.2.3: Package Dimension of AC108[6]

The creation of the PCB library document for AC108 is based on the package dimension shown in Figure 4.4.2.3. The package of AC108 has both a small dimension of 6mm × 6mm × 0.75mm and 48 tiny pins with a dimension of 0.2mm × 0.4mm × 0.2mm, which makes it even more challenging to make the hardware. Besides, the creation of the PCB library for AC108 requires information from IC manufacturers and manufacturing technology from PCB companies. Some PCB companies are unable to support the manufacturing of PCBs with such small packages; therefore, selecting manufactures to make the final PCB also takes a huge amount of time.

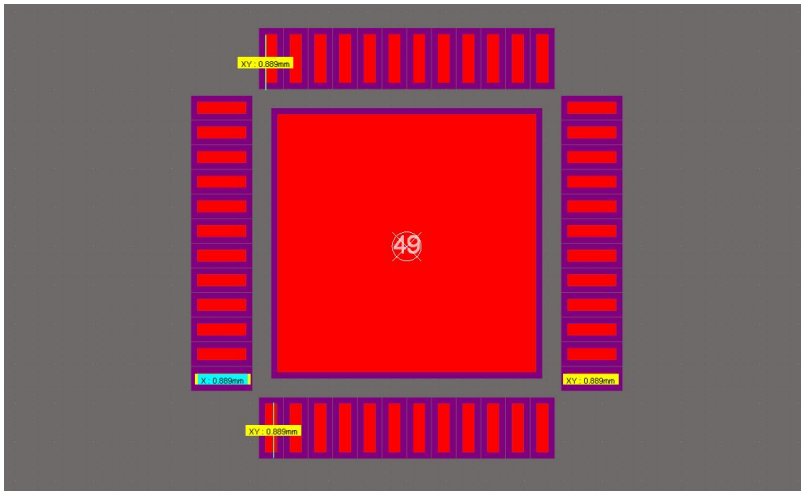


Figure 4.4.2.4: AC108 Pin Pad Layout Design for PCB

Figure 4.4.2.4 shows the AC108 pin layout for PCB design. Some of the details in the design are extremely important because ignorance of these details will lead to failure in the third design. The first detail is the length of the smallest pin pad in the pin layout design, which is about 0.889mm and longer than pin length L in Figure 4.4.2.3; such kind of design considers the sizes of pins and the IC package to make sure that IC won't peel away after being soldered on PCB. Making the pin pad longer is one of the main techniques to solder small IC packages firmly on the PCBs. Besides, the pad with label number 49 is the digital ground of AC108, which needs to be designed carefully in the final layout because there should be a specific distance between the analog ground and the digital ground to reduce signal distortion and crosstalk.

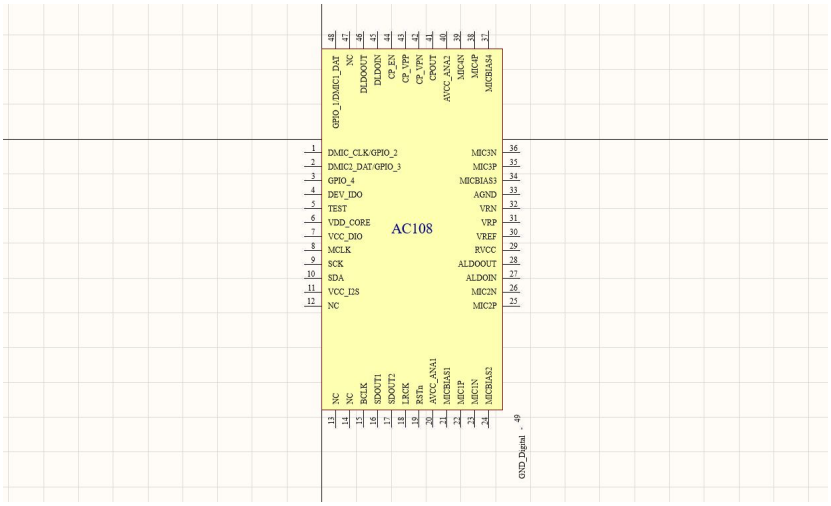


Figure 4.4.2.5: AC108 Label for Schematics

4.4.3 Schematic Design

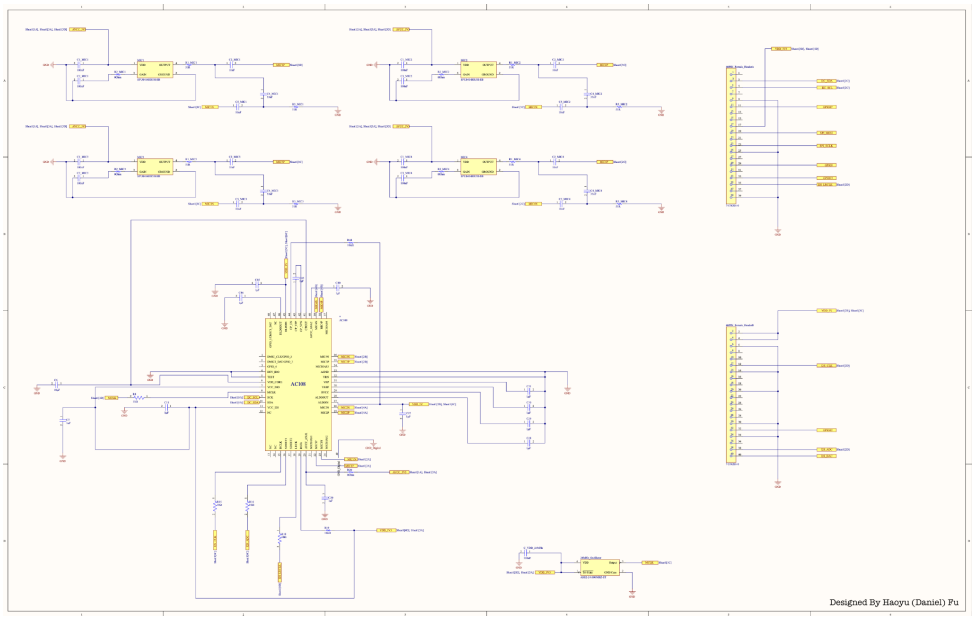


Figure 4.4.3.6: Third Design (4-Mic Array) Schematic

In Figure 4.4.3.6, there are four types of blocks, including the AC108, analog MEMS microphone, 24MHz CMOS compatible SMD crystal oscillator, and 40-pin female headers.

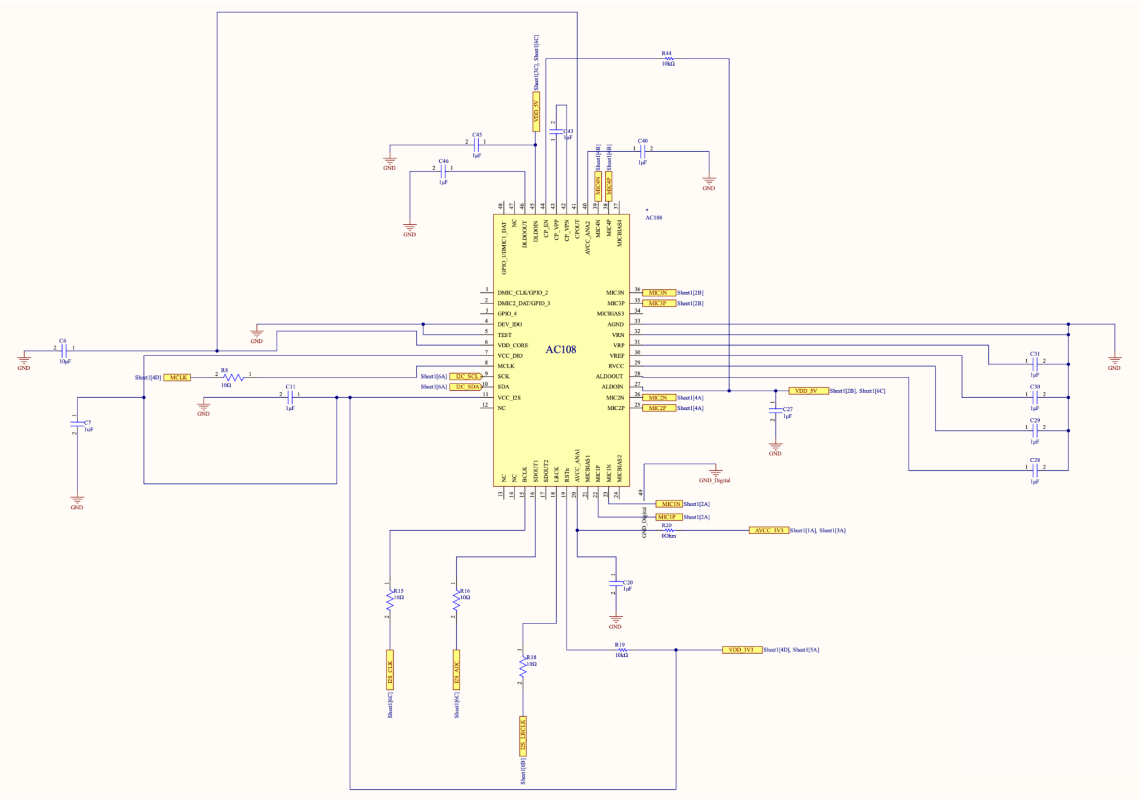


Figure 4.4.3.7: AC108 Block in Schematic

In Figure 4.4.3.7, some capacitors are connected between power supplies and ground to prevent noise propagation to the subsequent circuit by transmitting the noise to the grounded side. The analog ground and the digital ground are separated to avoid crosstalk between signal paths. Similar methods will be used in analog MEMS microphone blocks and the 24MHz crystal oscillator block, which will be shown in Figure 4.4.2.8 and Figure 4.4.2.9.

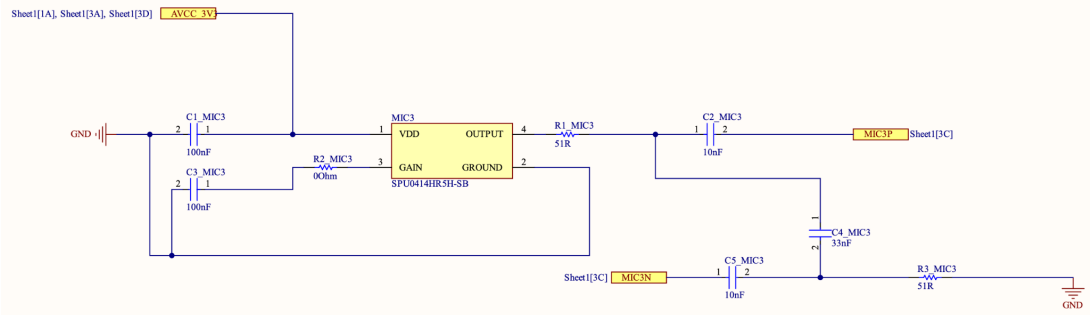


Figure 4.4.3.8: Analog MEMS Microphone Block in Schematic

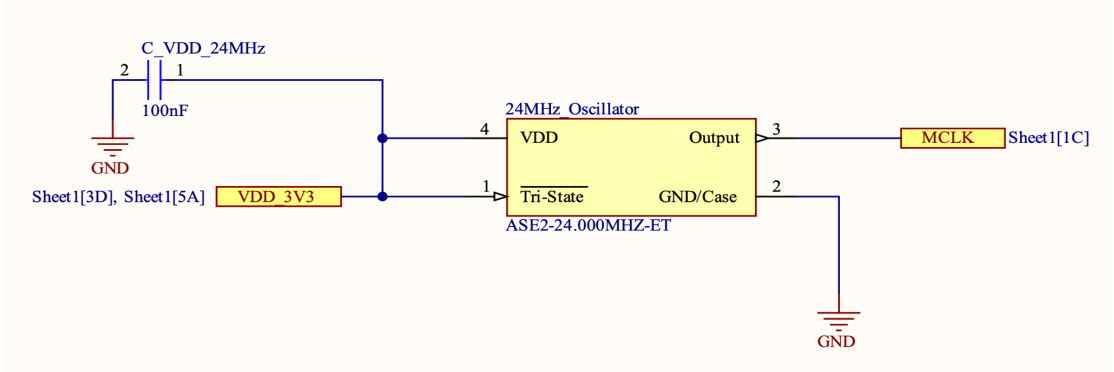


Figure 4.4.3.9: 24MHz Crystal Oscillator Block in Schematic.

A 24MHz crystal oscillator, which is in a compact package and low in height at 1.2mm, is used in the third design. It has low current consumption and a tri-state function to reduce power. VDD connects to a 100nF capacitor to reduce noises for stable output frequency.

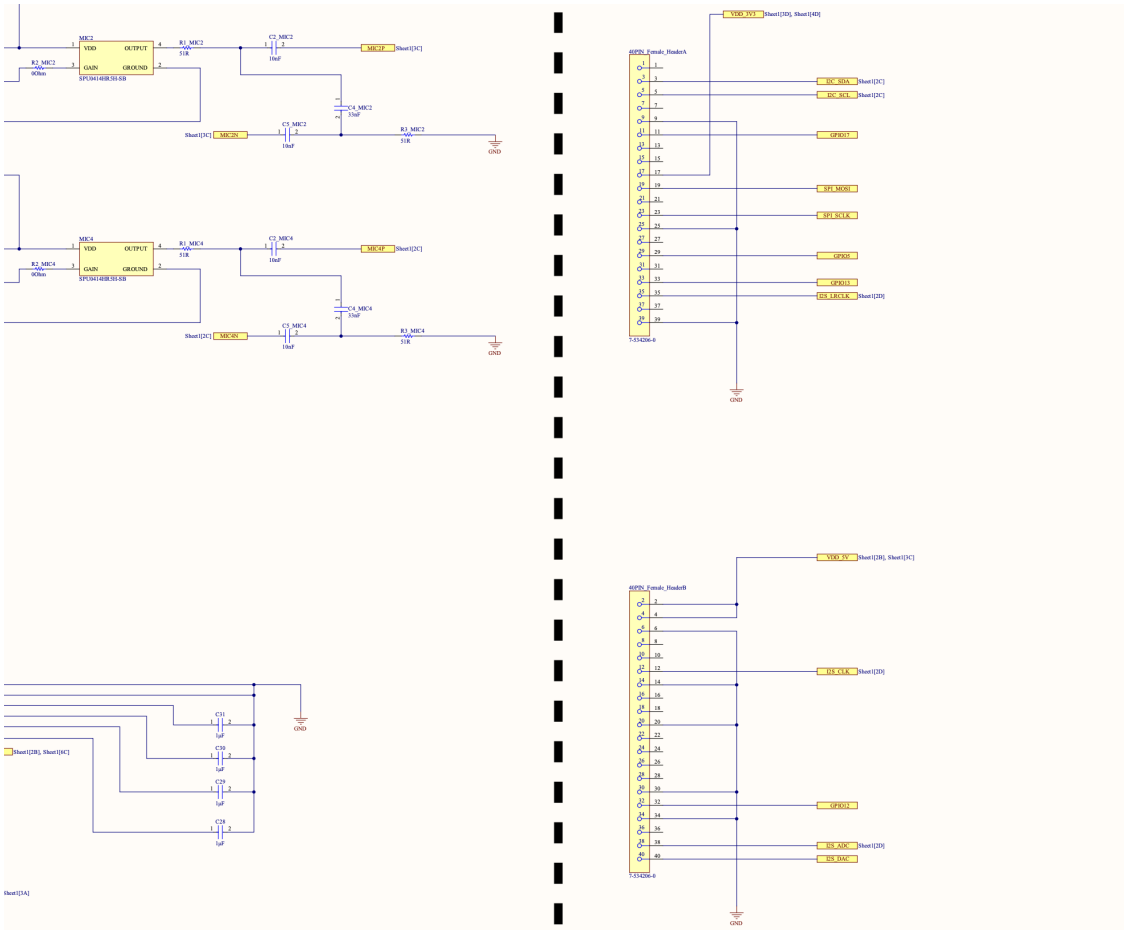


Figure 4.4.3.10: 40-Pin Female Headers in Schematic (On the Right Side of the Dotted line)

4.4.4 Layout Design

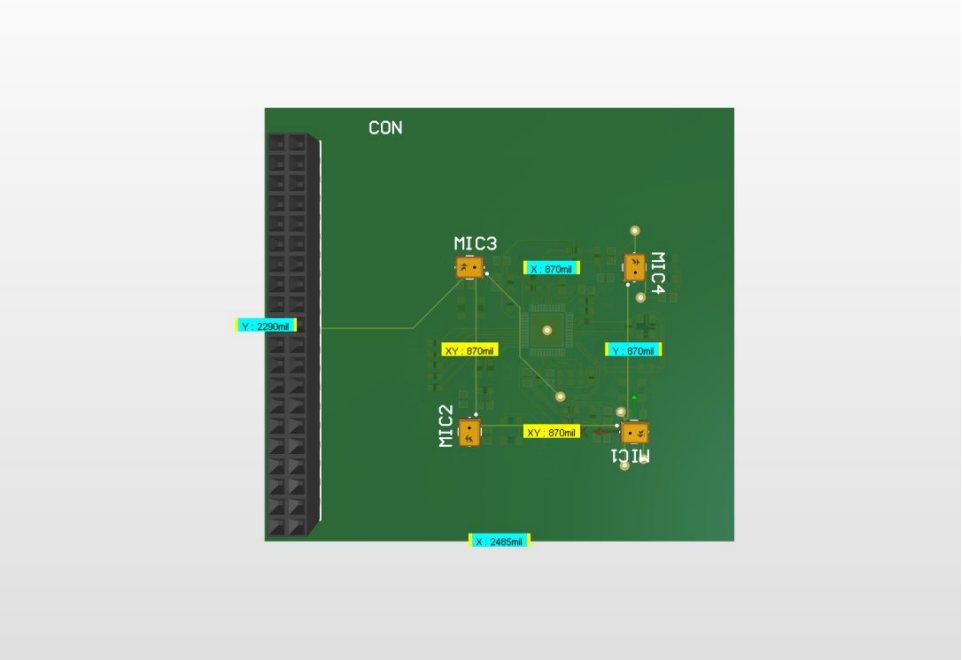


Figure 4.4.4.1: Bottom Side of Analog 4-Mic Array PCB Layout in 3D View

All the PCB schematics and PCB layout in section 4.4 are designed on our own. Figure 4.4.4.1 shows the layout of analog 4-mic array PCB design; the dimension of whole PCB is 78.359mm × 66.802mm, and the distance between every two microphones is 870 mil, which is about 2.21cm for a sampling rate of a minimum of 8000Hz. Both PCB schematic and PCB layout takes a huge amount of time; the information of IC package needs to be checked to make a stable final product and the routing of circuits requires information from different PCB manufacturers so that the PCB layout can pass DRC (Design Rule Check). The size of the 4-mic array in the third design is significantly smaller than the one in the ReSpeaker 4-Mic Array, which will be a better option for the software team to work on audio beamforming and model training in smaller spaces. However, it will take some time to get the hardware delivered to the software team in the next term because of the time consumed in manufacturing.

4.5 Design Decision

There are three hardware designs in total. The third hardware design is able to solve the issues from the first and second designs, including the limited number of microphones and the low sampling rate for audio and machine learning tasks; both of the issues are introduced in Section 3.5. Therefore, the final decision is the third microphone array design to record data from four MEMS microphones in WAV format. However, it takes some time to receive the product of the design from manufacturers.

There are three types of hardware design in total. The first design is usable but may not be accurate in model training. The second design is unable to solve the problems from the first design. The third design is able to achieve the requirements of the software team, but it will take a longer time for the software team to get the final version of the microphone array because the hardware design team needs time to find and check every single detail and manufacturer spends time to make the final product.

5 Discussion and Future Plan

Designing microphone array hardware is challenging because it not only needs to design the hardware itself but also work on the software. There were so many mistakes made during the design process, including the selection of hardware parts and the direction of development. For instance, when designing the first hardware, a huge amount of time has been wasted on the PCB design with two digital MEMS microphones, and the prototype of the audio amplifier circuit on the breadboard, which was not the initial steps planned for the project.
The following are some of the technical challenges we have experienced.

5.1 Sensor Data Collection Protocol and Sampling Rate

The sampling rate is deeply related to data transfer protocols. There are two data transfer protocols in the three hardware designs, including I2S and SPI. Raspberry Pi has both I2S and SPI transfer protocols. With MCP3008, Raspberry Pi can achieve data acquisition from four microphones. However, the sampling rate of SPI protocol on Raspberry Pi is too low. I2S supports the required sampling rate, but there is only one I2S bus on Raspberry Pi, which means that I2S on Raspberry Pi can only support data acquisition from two I2S digital microphones. One of the platforms that has two I2S buses is STM32F413/423[27].

5.2 Number of Microphones

The number of microphones that can be used in an audio beamforming system is often limited by the physical space available for the microphones and the capabilities of the embedded system used to process the data.

Firstly, the physical space available for the microphones can limit the number of microphones that can be used in an audio beamforming system. For example, if the system is designed to be compact or portable, it may only be possible to accommodate a limited number of microphones. Similarly, if the system is designed for use in a specific environment where space is limited, the number of microphones that can be used may be limited by the available space. The distance between the microphones is also determined by the sampling rate.

Secondly, the embedded system used to process the data can also limit the number of microphones that can be used in an audio beamforming system. The processing power and memory capacity of the embedded system will determine how many microphones can be read and processed in real-time. If the system is designed to use a simple protocol to read data from the microphones, such as I2S or SPI, the number of microphones that can be used may be limited by the bandwidth of the protocol.

5.3 Potential Integrated Chip Hardware Design Opportunities

The ultimate goal of this project is to achieve audio beamforming in the earable applications, which has new requirements and challenges on both the hardware and software sides, including reducing the size of the hardware. Therefore, a customized design for the IC or a more integrated PCB design has the possibilities to create a smaller for earable applications and the new software needs to consider the effects of the sampling rate and distance between the integrated microphones in the custom IC, which may have the intersection among hardware, software, and biomedical engineering. In the future, the project has the possibility to become interdisciplinary research, which makes me feel excited.

5.4 Lessons Learned

In the development process, the design for both hardware and software should be inseparable. I learned a lot during the development process and realized how important teamwork and communication are to achieve a mutual goal. Besides, I realized that designing good hardware is challenging because it is not only about designing the hardware itself but also about designing the whole system which includes both hardware and software.

6 Conclusion

Based on the three designs, we noticed that there are several key aspects in the hardware design, including the sampling rate for data from microphones, the distance between every two microphones, and the number of microphones.

The number of microphones used in beamforming affects the beamforming resolution, accuracy, and sensitivity. Typically, more microphones result in higher resolution and accuracy, but come with higher costs and increased complexity. A common configuration for microphone arrays is a linear or circular arrangement, which can be used to enhance the directionality of sound capture in one or more directions.

The distance between microphones in a beamforming system is also an important factor in determining its performance. The spacing between microphones determines the directionality of the beamformer, where a larger spacing results in a narrower beam and higher directivity. However, larger spacing can also result in grating lobes, which are false peaks in the beam

pattern. Optimal spacing is determined by the wavelength of the sound being captured and the desired beam width.

Finally, the sampling rate of the microphones determines the frequency range and resolution of the captured signal. Higher sampling rates are beneficial in capturing higher frequency components and providing better time resolution but come with higher data storage requirements and processing demands.

In conclusion, the effectiveness of audio beamforming is dependent on the number and arrangement of microphones, the distance between them, and the sampling rate of the microphones. These factors must be carefully considered in designing a beamforming system for a particular application.

Reference

[1] “Soundsport Pulse Wireless Headphones.” *Bose*, https://www.bose.com/en_us/support/products/bose_headphones_support/bose_in_ear_headphones_support/soundsport-wireless-pulse.html.

[2] “Airpods Pro Teardown.” *IFixit*, 28 Apr. 2022, https://www.ifixit.com/Teardown/AirPods+Pro+Teardown/127551.

[3] Chatterjee, Ishan, et al. “ClearBuds: Wireless Binaural Earbuds for Learning-Based Speech Enhancement.” *ArXiv.org*, 27 June 2022, https://arxiv.org/abs/2206.13611.

[4] “Respeaker 4-MIC Array for Raspberry Pi: Seeed Studio Wiki.” *Seeed Studio Wiki RSS*, https://wiki.seeedstudio.com/ReSpeaker_4_Mic_Array_for_Raspberry_Pi/.

[5] “Respeaker 6-Mic Circular Array Kit for Raspberry Pi: Seeed Studio Wiki.” *Seeed Studio Wiki RSS*, https://wiki.seeedstudio.com/ReSpeaker_6-Mic_Circular_Array_kit_for_Raspberry_Pi/.

[6] “Product.” *Powers*, http://www.x-powers.com/en.php/Info/product_detail/article_id/41.

[7] “ICS-43432 - TDK.” *Low-Noise Microphone with I2S Digital Output*, TDK Corporation, Feb. 2015, https://invensense.tdk.com/wp-content/uploads/2015/02/ICS-43432_DS.pdf.

[8] Ada, Lady. “Adafruit I2S MEMS Microphone Breakout.” *Adafruit Learning System*, https://learn.adafruit.com/adafruit-i2s-mems-microphone-breakout.

[9] “MAX4466.” *Low-Cost, Micropower, SC70/SOT23-8, Microphone Preamplifiers with Complete Shutdown | Analog Devices*, 5 July 2012, https://www.analog.com/en/products/max4466.html#product-overview.

[10] Team, The Arduino. “Arduino Uno Rev3 with Long Pins: Arduino Documentation.” *Arduino Documentation | Arduino Documentation*, https://docs.arduino.cc/retired/boards/arduino-uno-rev3-with-long-pins.

[11] “Raspberry Pi Documentation.” *Raspberry Pi Hardware*, Raspberry Pi Ltd, https://www.raspberrypi.com/documentation/computers/raspberry-pi.html.

[12] “MCP3008 | Microchip Technology.” *MCP3008*, Microchip Technology Inc., https://www.microchip.com/en-us/product/MCP3008.

[13] “File:28 Pin MLP Integrated Circuit.jpg - Wikimedia Commons.” *File:28 Pin MLP Integrated Circuit.jpg*, Wikipedia, https://commons.wikimedia.org/wiki/File:28_pin_MLP_integrated_circuit.jpg.

[14] Voss, Susan E, et al. “Measurements of EAR-Canal Cross-Sectional Areas from Live Human Ears with Implications for Wideband Acoustic Immittance Measurements.” *The Journal of the Acoustical Society of America*, U.S. National Library of Medicine, Nov. 2020, https://www.ncbi.nlm.nih.gov/pmc/articles/PMC7791892/.

[15] Gao, Yang, et al. “Earecho: Using Ear Canal Echo for Wearable Authentication: Proceedings of the ACM ON Interactive, Mobile, Wearable and Ubiquitous Technologies: Vol 3, No 3.” *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, 1 Sept. 2019, https://dl.acm.org/doi/10.1145/3351239.

[16] Xie, Yadong. “Teethpass: Dental Occlusion-Based User Authentication via ... - IEEE Xplore.” *TeethPass: Dental Occlusion-Based User Authentication via In-Ear Acoustic Sensing*, IEEE Xplore, 2 May 2022, https://ieeexplore.ieee.org/document/9796951/.

[17] Saracco, Roberto. “Earbuds to Check Your Blood Pressure.” *IEEE Future Directions*, 13 Jan. 2020, https://cmte.ieee.org/futuredirections/2020/01/13/earbuds-to-check-your-blood-pressure/.

[18] Jin, Yincheng, et al. “EarHealth: Proceedings of the 20th Annual International Conference on Mobile Systems, Applications and Services.” *ACM Conferences*, 1 June 2022, https://dl.acm.org/doi/10.1145/3498361.3538935.

[19] *Sound Professionals Low Noise in-Ear Binaural Microphones - Amazon.com*. https://www.amazon.com/Sound-Professionals-NOISE-BINAURAL-MICROPHONES/product-reviews/B002N6PAMM.

[20] Röddiger, Tobias, et al. “Towards Respiration Rate Monitoring Using an in-Ear Headphone Inertial Measurement Unit: Proceedings of the 1st International Workshop on Earable Computing.” *ACM Conferences*, 1 Sept. 2019, https://dl.acm.org/doi/10.1145/3345615.3361130.

[21] Müller-Trapet, Markus, et al. “Acoustic Source Localization with Microphone Arrays for Remote Noise Monitoring in an Intensive Care Unit.” *Applied Acoustics. Acoustique Applique. Angewandte Akustik*, U.S. National Library of Medicine, Oct. 2018, https://www.ncbi.nlm.nih.gov/pmc/articles/PMC6039849/.

[22] “Flat No-Leads Package.” *Wikipedia*, Wikimedia Foundation, 3 Mar. 2023, https://en.wikipedia.org/wiki/Flat_no-leads_package.

[23] “WAV.” *Wikipedia*, Wikimedia Foundation, 28 Feb. 2023, https://en.wikipedia.org/wiki/WAV.

[24] Hu, Jwu-Sheng, et al. “Geometrical Arrangement of Microphone Array for Accuracy Enhancement in ...” *Geometrical Arrangement of Microphone Array for Accuracy Enhancement in Sound Source Localization*, IEEE, 20 June 2011, https://ieeexplore.ieee.org/document/5899088.

[25] Benesty, Jacob, et al. *Microphone Array Signal Processing*. Springer, 2008.

[26] Nagimov. “Nagimov/mcp3008hwspi: Fast MCP3008 Reader for Raspberry Pi.” *GitHub*, https://github.com/nagimov/mcp3008hwspi.

[27] “STM32F413/423.” *STMicroelectronics*, https://www.st.com/en/microcontrollers-microprocessors/stm32f413-423.html.

[28] “Phase Difference and Path Difference.” *Physics Things*, 9 Apr. 2014, https://physicsthings.wordpress.com/as-physics/unit-2-mechanics-materials-and-waves/phase-difference-and-path-difference-2/.

[29] HeadphonesAddict, Matija Ferjan. “Airpods Facts 2023: Airpods Revenue, Release Date, Units Sold.” *HeadphonesAddict*, HeadphonesAddict, 21 Feb. 2023, https://headphonesaddict.com/airpods-facts-revenue/.

[30] Lewis, Jerad. “Common Inter-IC Digital Interfaces for Audio Data Transfer - Analog Devices.” *Analog Devices*, Analog Devices, Jan. 2012, https://www.analog.com/media/en/technical-documentation/technical-articles/MS-2275.pdf.

[31] Benesty, Jacob, et al. “Microphone Array Signal Processing.” *Scitation*, Acoustical Society of AmericaASA, 1 Jan. 1970, https://asa.scitation.org/doi/10.1121/1.3124775.

[32] “Arduino Uno REV3.” *Arduino Online Shop*, https://store-usa.arduino.cc/products/arduino-uno-rev3.

[33] Voss, Susan E., et al. “Measurements of Ear-Canal Cross-Sectional Areas ... - Smith Scholarworks.” *Smith ScholarWorks*, Smith College, 24 Nov. 2020, https://scholarworks.smith.edu/cgi/viewcontent.cgi?article=1078&context=egr_facpubs.