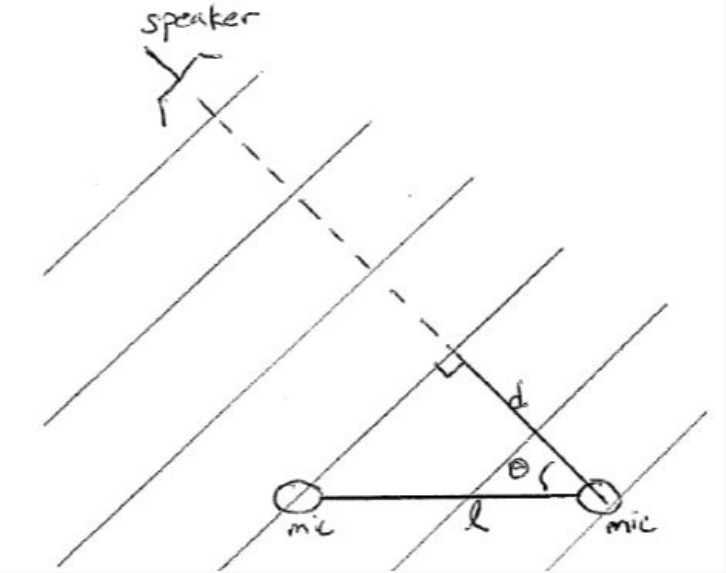
Audio Direction Finding Project

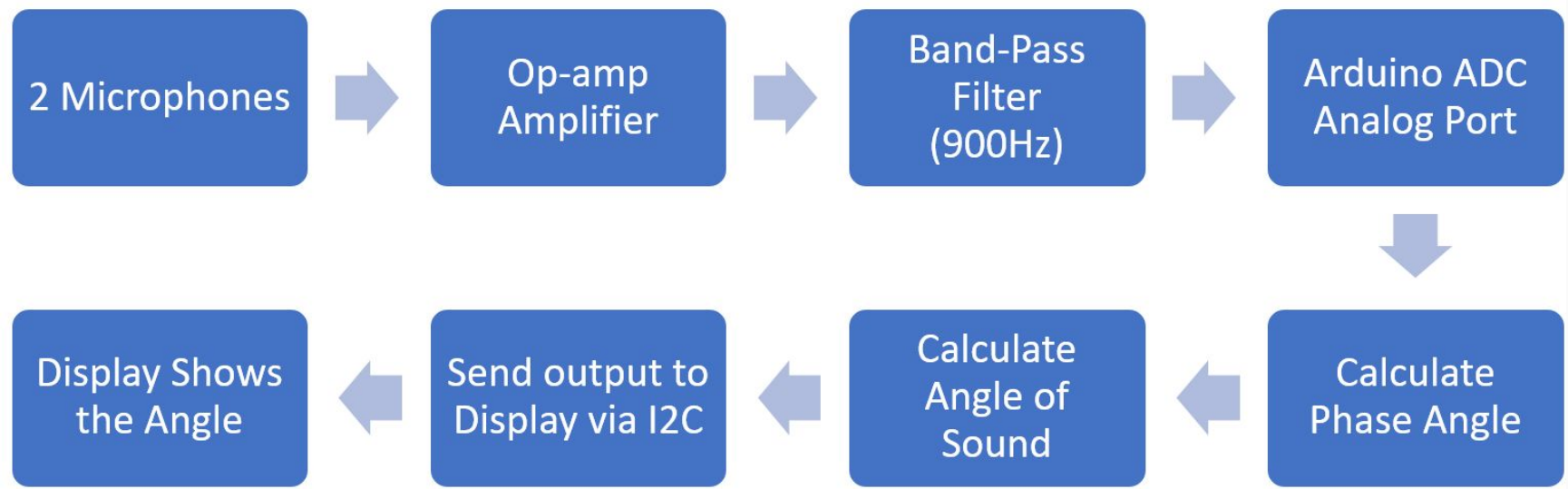
By Saksham Goyal and Jeff Kedda

Introduction

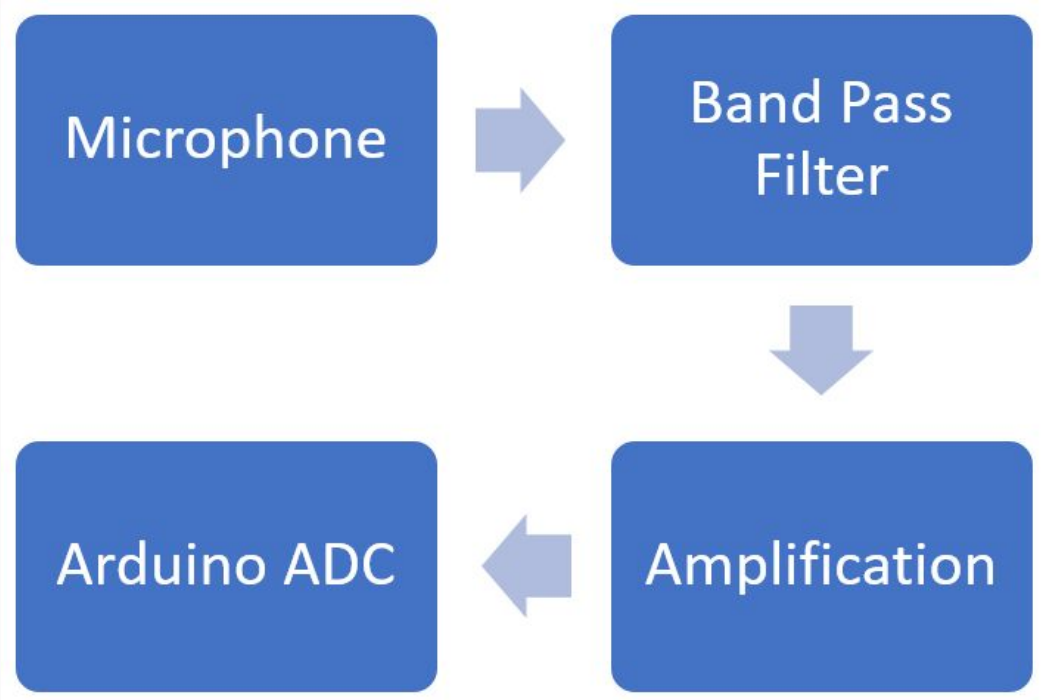
- Our goal was to be able to find the direction from which a signal is emitted.
- We have 2 microphones which we use to find the source.
- The source will be a 900 Hz signal
- We have a screen where we will display the calculated direction



Block Diagram

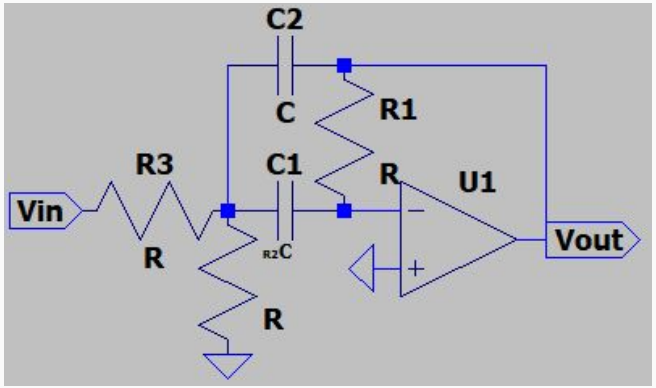


Circuit Design

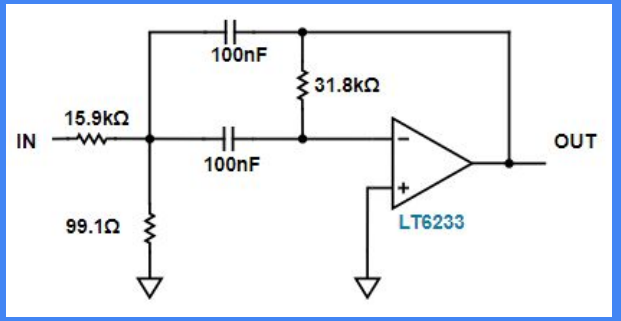


Narrow Band Pass Filter

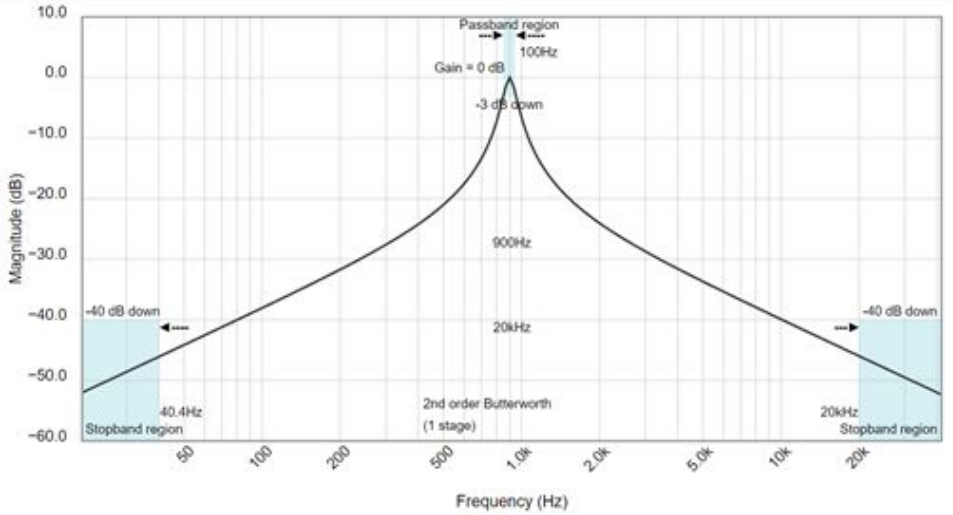
- We used a Multiple Feedback Narrow Band-Pass Filter
- Narrow band pass filters are similar to combining an low-pass and high-pass filter into one
- Narrow band-pass let us set the band to be 800Hz - 1kHz
- We then used another op-amp for amplification
- Filter and Amp are both inverting designs so the output is still positive and circuit is significantly simplified



Choosing Values for Filter



- Needed narrow pass-band to compensate for non-ideal op-amp
- We needed low noise optimisation to make sure the phase is not changed
- Implemented amplification in 2nd stage so amplification here was not needed
- Actual values are different to accommodate resistors that we have in our kit



Passband

Gain: 0 dB

-3 dB 100 Hz

Stopband

-40 dB 20k Hz

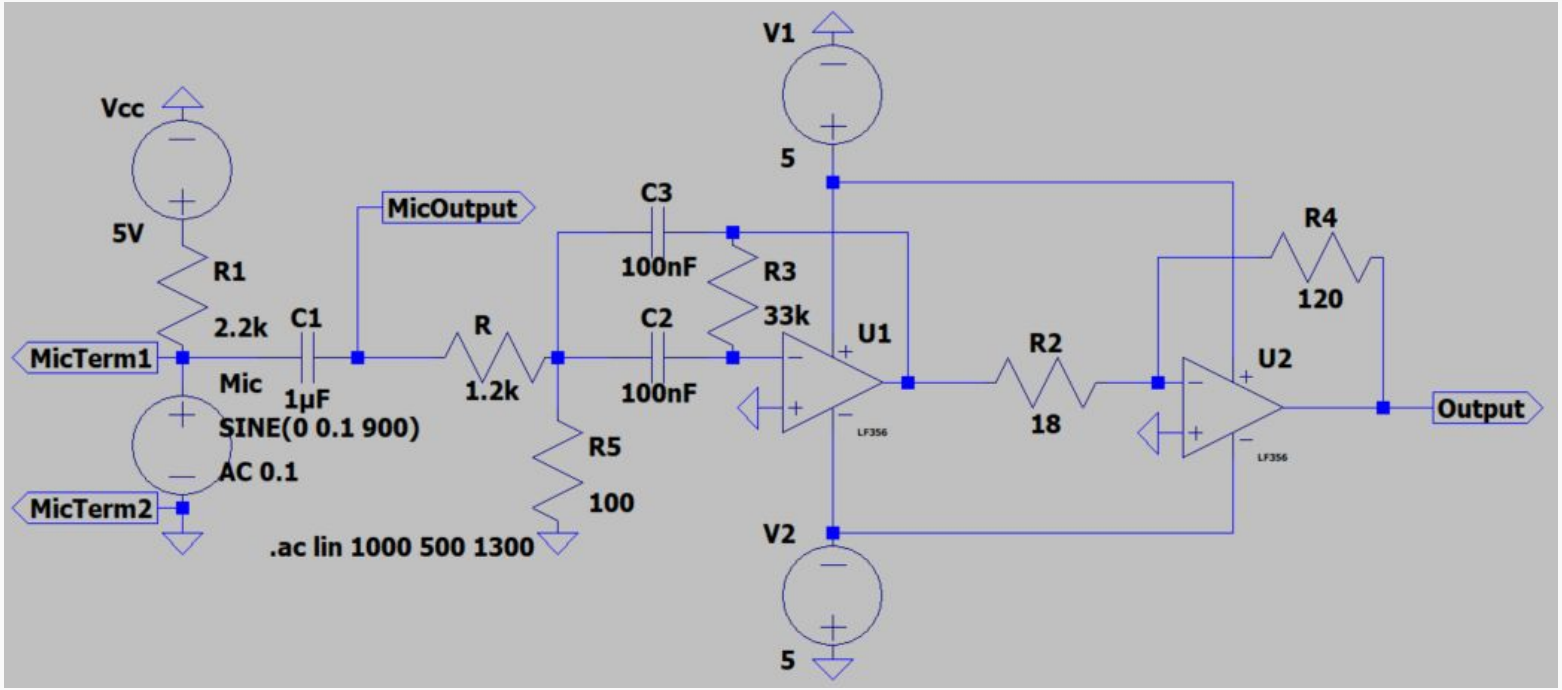
Center Frequency

900 Hz

Optimization

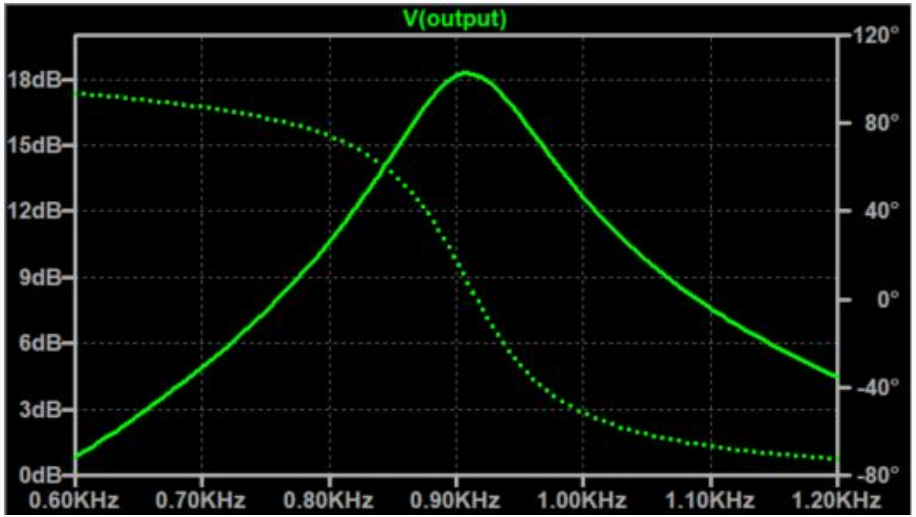
Low Noise

Full Circuit Schematic



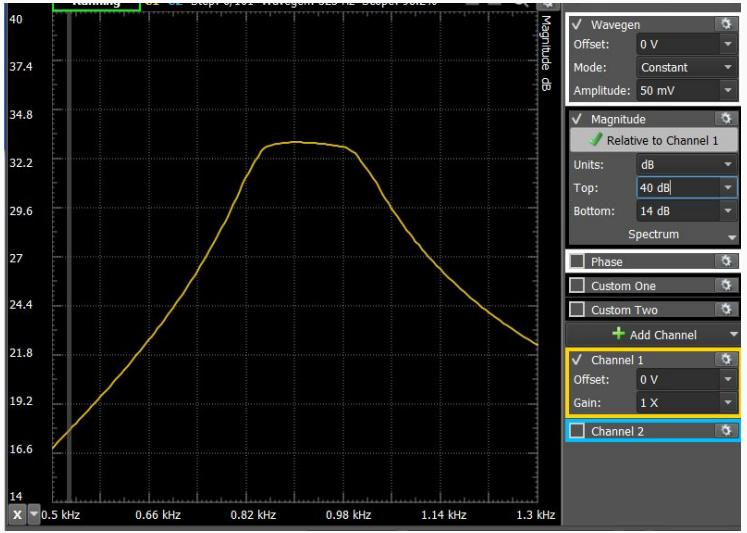
Simulation in LTspice

- Most components are ideal in simulation so it does not match with real scenario
  - Designed circuit to be as close to this with our parts
- Output at 900 Hz is about 2.5x gain compared to 800/1kHz
  - 900Hz@18db vs 800Hz@10db



Testing The Real Circuit

- Output has the correct center
  - Center is ~910 Hz
- The top of the gain is limited by the op-amp power limit
- Passband has correct cutoff points
  - Cut-off at ~810 Hz and at ~1 kHz
- Significantly more gain compared to the simulation in LTspice
  - Calculated peak was 18dB, real is 33dB

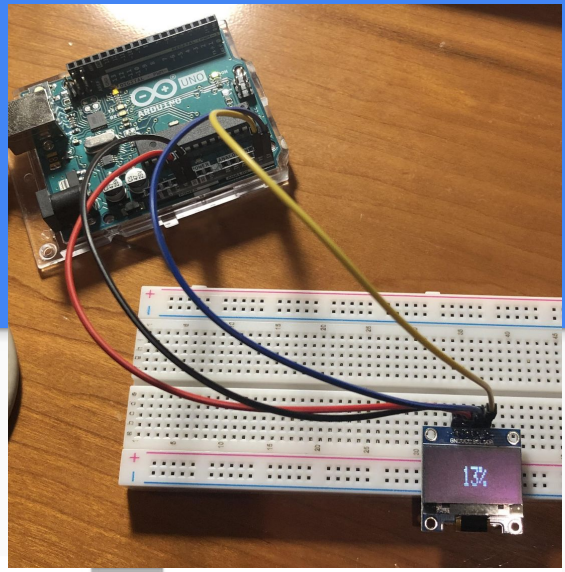


Code

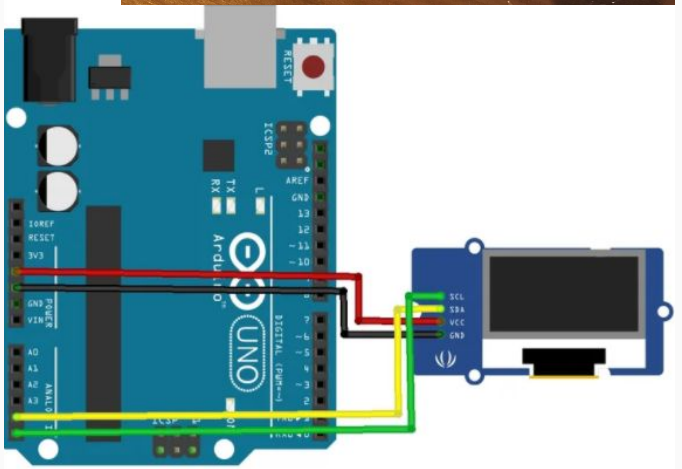
- The code first imports the adafruit libraries for the display
  - These prebuilt libraries significantly shorten the process to code since all of the backend is part of the library.
  - Just the code to display the relevant text/angle is needed
- Then we setup the display by initializing the I2C protocol
- We set up the formatting of the text and print out the text
- Code to calculate the angle has not been implemented yet so currently the display is static

```
#include <SPI.h>
#include <Wire.h>
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>

#define OLED_RESET 4
#define SCREEN_ADDRESS 0x3C
Adafruit_SSD1306 display(OLED_RESET);

void setup() {
  display.begin(SSD1306_SWITCHCAPVCC, SCREEN_ADDRESS);
  display.clearDisplay();
  display.setTextSize(1);
  display.setTextColor(WHITE);
  display.setCursor(0, 0);
  display.println("Find Audio Direction");
  display.display();
  display.setTextSize(1);
  display.setTextColor(WHITE);
  display.setCursor(0, 20);
  display.println("Angle = ");
  display.display();
}
```

OLED Display



- This was the wiring diagram for the display
- The I2C protocol works on the A4 and A5 pins
  - Connect SCL to A5 and SDA to A4
- The display Powered/Grounded via Arduino
  - AD2 cannot supply enough current to drive all 3 circuits (2 for each mic and the display)



Problems We Encountered

- When using the USB Power Supply, the output had an abnormal amount of noise
  - Fixed by changing supply to the AD2
- The top of the bode plot is cut because of the op-amp limits
  - Currently unresolved but we plan to change the microphone biasing to reduce input voltage
- Circuit is hitting AD2 current limits
  - Previously had display using AD2 power, but we are changing that so it uses 5V from the Arduino to avoid reaching AD2 current limits

Timeline

We are currently on schedule and all requirements from milestone 1 and 2 are completed

Week 4:

- Create code that extracts the information from the microphones and calculate the correct sampling rate
- Create and test code that finds the phase shift between the microphone signals
- Compare measured results from expected/calculated results

Week 5:

- Create equation to calculate the angles and program it into the Arduino
- Show output to display and show the angle on the OLED
- Test values from expected results