Introduction

The goal of the experiment was to design and build a circuit that samples audio using a microphone and to obtain an intelligible ".wav" file that matches the audio input the microphone picked up.To do so, A Band pass audio amplifier circuit was designed using the MCP 6272 Op-Amp and various values of resistors and capacitors chosen to meet an appropriate and pass band which approximated the human voice range of 300 to 3000 Hz.An STM 32 Nucleo Board was used to power the audio amp and interpret and process the output AC voltage of the amplifier circuit. The STM board ran a C programme that took the analog AC values and used the onboard ADC to convert it to values it printed out using a terminal application called PuTTY. Finally the values obtained on Putty were saved in a log file which was uploaded to MATLAB for post-processing and to reconstruct the audio signal obtaining a ".wav" file.

Choosing Circuit values

Choosing R0 and C0 and R3 , R4: According to a Circuit diagram in the Microphone Datasheet[1.B] we know our R0 and C0 values are meant to be 2.2kΩ and $1\mu F$ respectively. We use the values, R3 and R4, to form a voltage divider to act at the Non-inverting input of the Op-Amp so we arbitrarily choose their values and make sure they are equal. In this case we chose their values to be 5.6kΩ each.

Calculating the DC Input: From the microphone datasheet [1.B] we know that the DC voltage input was 0.5 mA[1.B] and we assumed the AC output of the mic to be about $3\mu A$ [1.D]. To use the Op Amp we need an idea of the voltage produced by this AC signal to know our Vin. Since we know the mic is attached to a 2.2k resistor [1.B] we use Ohm's law ,V = IR. $V_{in} = 3\mu A * 2.2k\Omega = 6.6mV$.

Calculating Gain and picking R1,R2 : Next we need to establish an upper and lower bound for our gain. We know our $V_{min\ gain} = 1 / 2 = 0.5V$ (since we know we need at least 1 Volt peak to peak) and our $V_{max\ gain} = 3.3 / 2 = 1.65V$(since the op amp is rated for 3.3V) thus our min gain is : $\frac{V_{min\ gain}}{V_{in}} = \frac{0.5V}{6.6mV} = 83$ and $\frac{V_{max\ gain}}{V_{in}} = \frac{1.65V}{6.6mV} = 250$ we thus choose the gain to be 88
We pick R1 = 2.2kΩ and R2 = 390kΩ and we thus calculate:
$gain = \frac{R_2}{R_1 + R_0} = \frac{390{,}000\Omega}{2200\Omega + 2200\Omega} = 88\ approx.$

Choosing c1 and calculating cutoffs : C0 Forms a High pass filter with R1 so using the formula $f_c = \frac{1}{2*\pi*R\ *C}$ we get $f_c = \frac{1}{2*\pi*R_1*C_0} = approx\ 80\ Hz$ to get a lower cutoff frequency. We choose a C1 = 150pF resistor to be paired with R2 to form the low pass filter, using the formula we get $f_c = \frac{1}{2*\pi*R_2*C_1} = approx\ 2400Hz$. This does a decent job of staying within the human voice frequency range of 300 to 3000 Hz which is what we want[1.D].

No changes were made to the component values in the final amplifier circuit.
Additional smaller circuits added to the Breadboard: a set of decoupling capacitors rated 10 nF and 1 μF, to decrease the noise caused by digital signals of the Nucleo Board.
And a 4.7 kΩ resistance and A Green LED to indicate whether the breadboard is receiving power.

Circuit testing

Testing the Op-Amp DC Level: Before connecting the microphone the Amplifier circuit was first tested to see if it was operational. The Board was supplied with 3.3 Volts and an Oscilloscope Probe set to DC Coupling was attached to the output of the Op-Amp. The probe showed to settle at 1.6V which is approximately half the DC input (3.3V/2 = 1.65V) we provided and is consistent with the Voltage divider at the Non-inverting input of the Op-amp.

Testing the Op-Amp gain and Cutoff frequency :A waveform generator was used as an input to the circuit with a 100 mV Peak to Peak Sine wave.The input must be at least 100mV inorder to achieve a clean looking output. Since the designed gain would cause clipping out of the 3.3V rating of the MCP6272 Op-Amp for the given input[1.A], a voltage divider was designed to step the voltage down. The voltage divider ratio was designed to be 1:15 and it was assembled at the inverting input Op Amp. This input was supplied with the output of the waveform generator and an Oscilloscope probe was placed at the output of the Op-Amp. We observed an output sine wave of 610mV Vout using the oscilloscope. To calculate the gain we use the formula $\frac{V\ out}{V\ in} = Gain$. To obtain Vin we divide 100 mV by 15 due to the Voltage divider and obtain 6.6 mV. We know our V out is 610mV. Thus our gain is $\frac{610mv}{6.6mv} = approx$ 90 which is a little higher than the designed gain of 80.

The cutoff frequency was experimentally verified by generating a Bode plot using an Oscilloscope. A T splitter was attached to the Gen-out port(Voltage generator of the oscilloscope). One end was connected with a cable with crocodile clips at the end (This will be attached to th einput_and the other was connected in place of the Probe 1 port.using a BnC to BnC cable. Finally a BNC to crocodile clip cable was connected to probe port 2 and attached to the output of the amplifier circuit. The oscilloscope was then used to generate a Bode plot from the range of 10 Hz to 5000 Hz. The band pass was obtained and we can see that the gain is attenuated at approximately 70 Hz and at approximately 2400 Hetz. This matches our theoretical calculation.[7]

Testing the microphone: The microphone was added after the voltage divider was removed according to the LT spice Diagram. Now that the circuit is functional we set up the tools we will use to process the data. We used the STM Cube IDE to write the code in C that gets uploaded to the STM Nucleo Board. The code will read the values on an input pin , send it to the inbuilt Analogue to Digital Converter which outputs values between 0-4095 that map to values between 0 - 3.3 V. These output values are printed onto a terminal application called PuTTY. The PuTTY console is set to 500000 Baud rate under the serial data logging option. Appendix Image 4 Contains the C Code that was uploaded to the Nucleo Board and executes this functionality[4].
The STM Nucleo was connected to a Laptop to upload the code and wires were used to connect the 3.3 Volts and GND pins of the Nucleo to the Breadboard. The debugging LED lit up indicating the amplifier has power and the circuit is active. A final 3rd wire was used to connect to the ADC input pin defined in the IOC file in the STM Cube IDE that defines which pins are active and where the signal goes to. Once we obtained the logged values we imported them into MATLAB and the data was converted in column vectors between 1 and -1 so that it can be converted to an intelligible audio sample. We use the audiowrite function on this data and we obtain the audio signal as a .wav file and as a plot. [5],[6]

Conclusion :The amplifier had a gain of 90 and was able to record at least 24000 signal samples in the 80 Hz to 2400 Hz range, which is roughly around the human speech range. After processing the data, we managed to get clear ".wav" audio files and their frequency plots. As a possible improvement we could have chosen resistors and capacitors that matched the range closer.

Appendixes:

1:

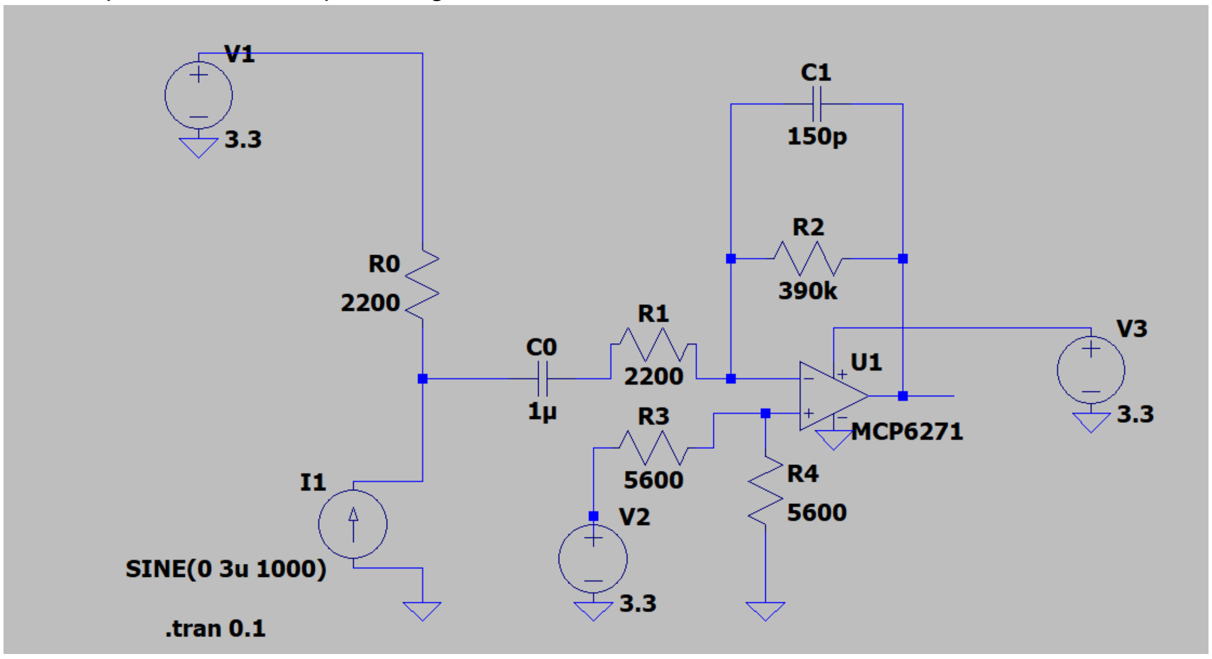
A - Microchip Technology Inc. (2004). M CP6271/2/3/4/5. In *Microchip Technology*

B - *CUI Devices. (2020). ELECTRET CONDENSER MICROPHONE (pp. 1–4).*

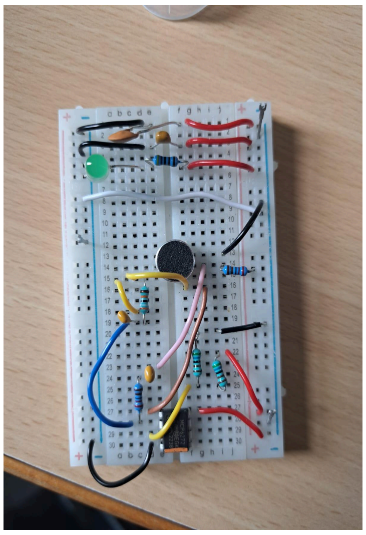
C - *J. Thompson. (2023). SES Lab 2: Amplifier Design*

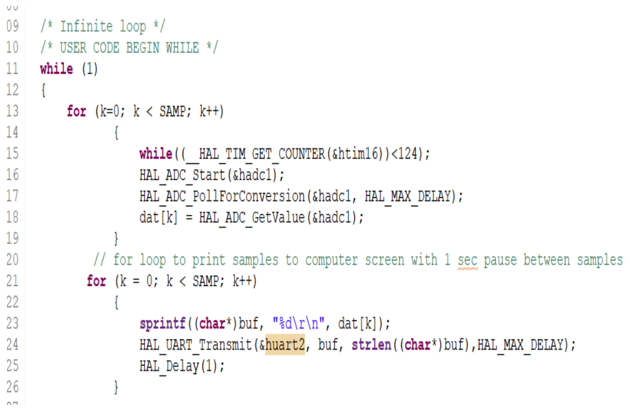
D - *J. Thompson. (2023). SES Lab 2: STM32 Audio Recording.*

2: Microphone circuit LTspice Diagram



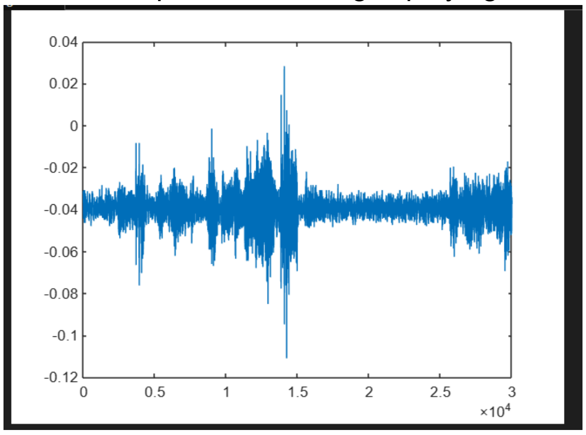
3: Breadboarded Microphone circuit: **4:** Code Uploaded on the STM Nucleo Board

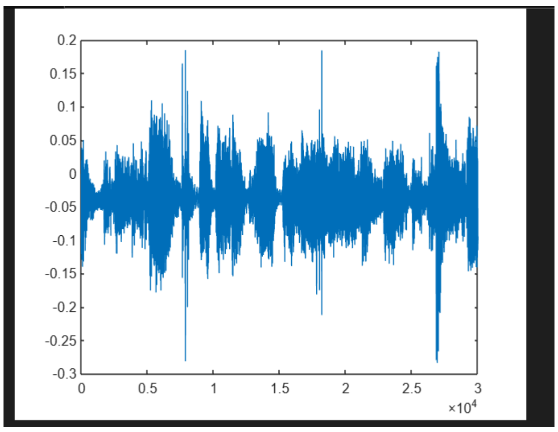




```
/* Infinite loop */
/* USER CODE BEGIN WHILE */
while (1)
{
    for (k=0; k < SAMP; k++)
          {
              while((__HAL_TIM_GET_COUNTER(&htim16))<124);
              HAL_ADC_Start(&hadc1);
              HAL_ADC_PollForConversion(&hadc1, HAL_MAX_DELAY);
              dat[k] = HAL_ADC_GetValue(&hadc1);
          }
        // for loop to print samples to computer screen with 1 sec pause between samples
      for (k = 0; k < SAMP; k++)
          {
              sprintf((char*)buf, "%d\r\n", dat[k]);
              HAL_UART_Transmit(&huart2, buf, strlen((char*)buf),HAL_MAX_DELAY);
              HAL_Delay(1);
          }
```

<u>5.</u> Audio Plot from a different part of the song but moving the source away after a few seconds(left)
<u>6.</u> Audio output while a song is playing continuously: (right)





<u>7.</u> Bode plot to verify cutoff frequency