ENSC 894: ADVANCED ANALOG ELECTRONICS

Project – PDM Audio with D-Flipflop Trigger

Group 13

April 14, 2024

(Gene) Jun Li : 301152342

Prepared for:

Prof. Patrick Palmer

SCHOOL OF MECHATRONIC SYSTEMS ENGINEERING

SIMON FRASER UNIVERSITY

Table of Contents

Table of Figures

1. Introduction

This report discusses processes, and methods developed to measure the signal delay between two MEMS microphones. The application is for finding the time of flight (TOF) of the source signal, hence providing a quick, accurate and cost-effective way of making an anemometer. We are using a 39kHz ultrasonic generator as our source of signal and MEMS microphone(s), analog and digital as the receiving sensor.

2. Methods

We have examined several different methods on how to measure the phase shift between the two sinewave signals whether between the ultrasonic source and a MEMS mic or between the two MEMS mics.

2.1. Method 1 – Ultrasonic Generator to Analog MEMS Mic

Initially, we want to use an analog MEMS microphone to pick up the signal that is generated from the ultrasonic generator. We put the two in series with a fixed distance (200mm), use a function generator to excite the ultrasonic generator, and then plot this waveform against the output waveform from the analog microphone (Fig1). We then use an analog multiplier to get rid of the high frequency and a low-pass filter to get the DC signal. By measuring the phase difference between the two, we can find the delay (Time of Flight) of the two signals. With fixed distance, the velocity is the distance over TOF.

$$V_0 = f\lambda_0 \text{ , speed of sound} \tag{1}$$

$$\varphi_d = \left(A_1 \cos\left(\frac{2\pi * x_1}{\lambda} - 2\pi f t\right)\right) \times \left(A_2 \cos\left(\frac{2\pi * x_2}{\lambda} - 2\pi f t\right)\right) = \frac{2\pi\,(x_2 - x_1)}{\lambda} \tag{2}$$

$$V_0 = \frac{2\pi\,(x_2 - x_1) f}{\varphi_{d0}} \tag{3}$$

However, this method may not work as expected since the amplitude of the two signals is different due to the loss in the medium of travel. We will discuss the detail in the Result Section.

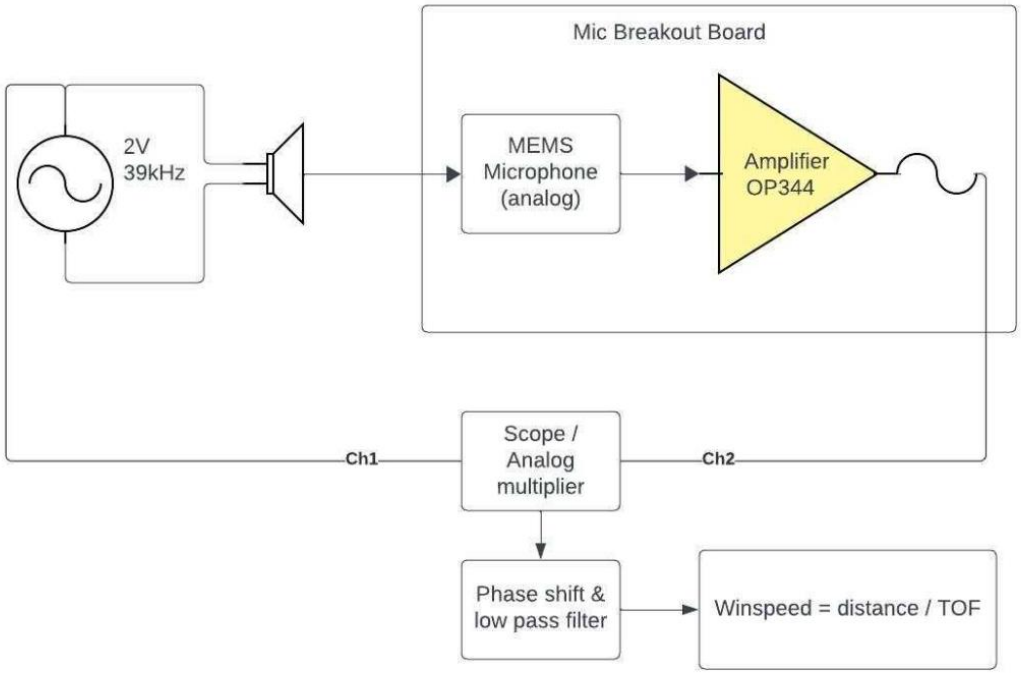


FIGURE 1: METHOD 1 – ANALOG MULTIPLIER

2.2. Method 2 – Ultrasonic Generator to Digital MEMS Mic

This method uses a digital MEMS mic with Pulse Density Modulation (PDM) output. Same physical layout as Method 1, we feed the two analog signals to two ADCs in an RPi Pico microcontroller. Then, we find the timestamp of the zero crossing of the two signals, hence the delay. This delay will be out TOF (see figure below).

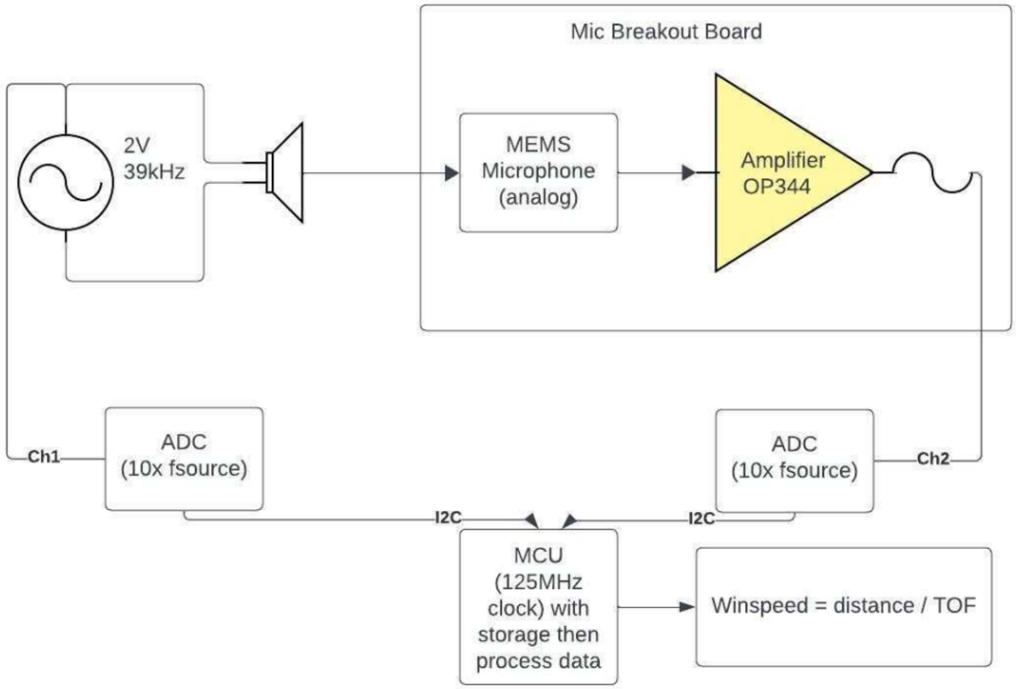


FIGURE 2: METHOD 2 – PDM MIC WITH ADC

2.3. Method 3 – Use a Microcontroller to excite the speaker and provide the clock for the MEMS PDM

This way, we are generating a 39kHz sinewave from an RPi Pico MCU via the DAC method. Also, we will be providing the 3.072Mhz clock to the digital mic. We then read the PDM data back to the Pico MCU. Then implement a Phase Lock Loop (PLL) to find the delay. However, a PLL might be too slow. If the signals are very stable and we do not need real-time output, this would not be an issue.

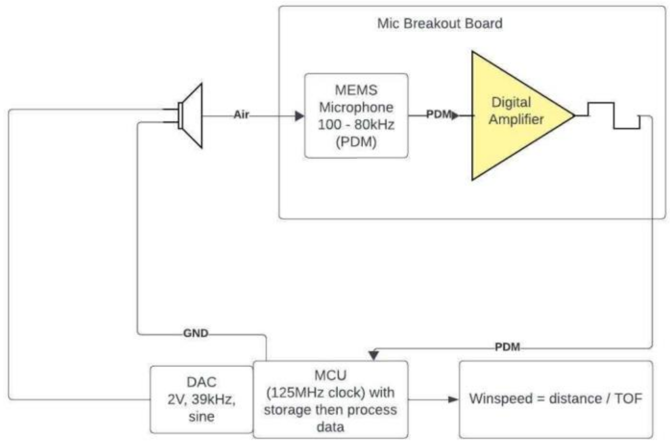


FIGURE 3: METHOD 3 – PDM MIC WITH PLL

2.4. Method 4 – Use two MEMS digital PDM mics and JK-Flipflops to find the zero crossing PDM signals

In this approach, we will be using two digital mics that produce two PDM signals for the Pico microcontroller. Firstly, we will generate a 39kHz sinewave from the Pico MCU via a Digital Analog converter. At the same time, we use Pico to generate a 3.072Mhz clock for the two digital microphones.

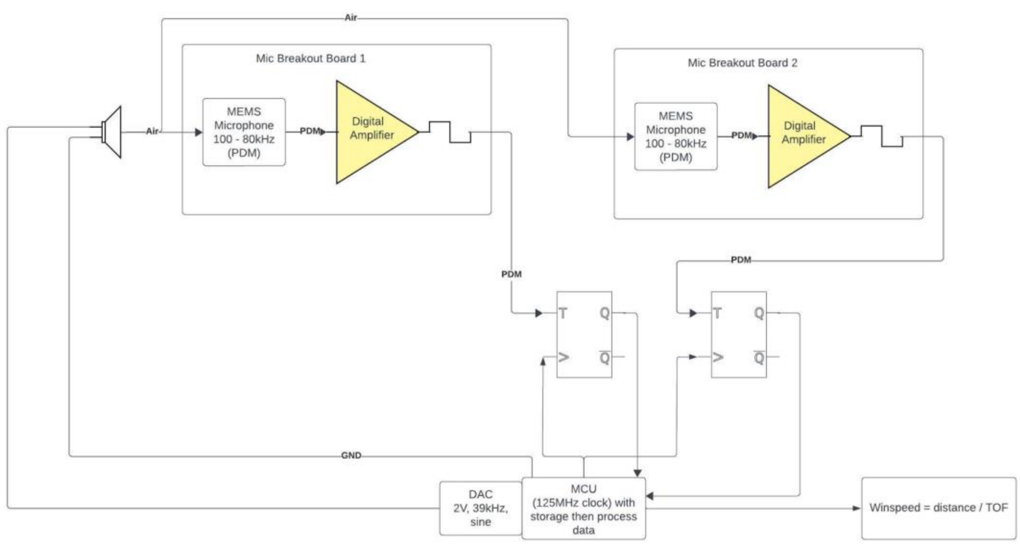


FIGURE 4: METHOD 4 – PDM MIC WITH JK FLIP FLOP

The figure below shows how the zero crossing of the PDM signals sync with the clock and gives a pulse when that happens. Signal 1 is the PDM output from mic1, and signal 2 is the PDM output from mic 2. At 50% duty cycle is the sinewave's zero crossing point. We use a logic gate to pulse that input and log the clock counter value in variable 1. Then, we keep the global clock running and wait for signal 2's zero crossing pulse. Once this pulse goes high, we store the global clock value in variable 2. The difference between variables 1 and 2 is our delay.

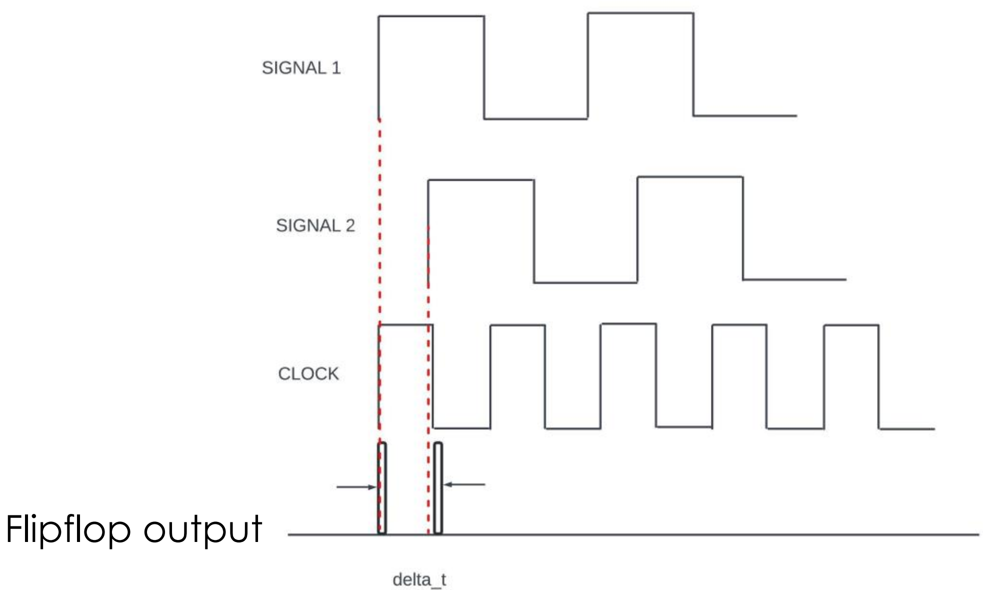


FIGURE 5: METHOD 4 – PDM SIGNALS WITH JK FLIP-FLOP OUTPUT

This is our preferred method and we will set this up to test the idea and compare it with the analog version (Method 1).

3. Results

3.1. Method 1 – Initial Testing

The following shows the initial test setup and results. We activate the ultrasonic generator using a function generator, 39kHz sine. Then we power the analog microphone and measure the audio signal from it.

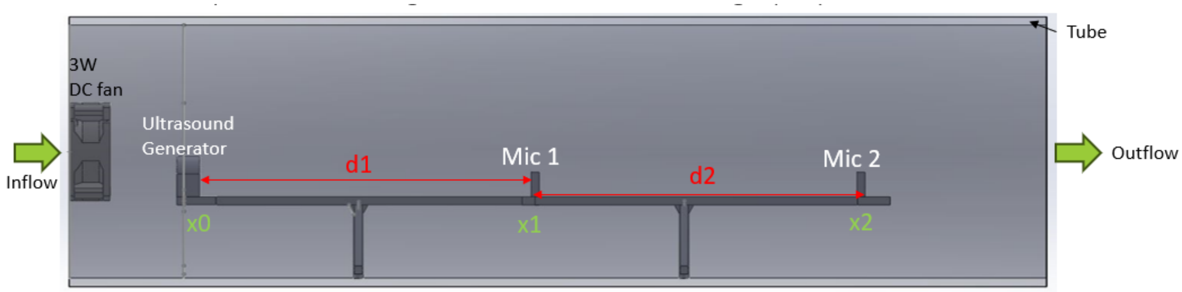


FIGURE 6: METHOD 1 – PROTOTYPE

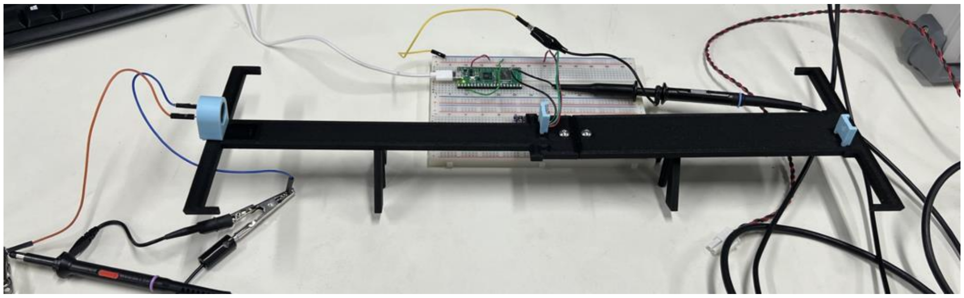


FIGURE 7: METHOD 1 – TEST

Figure 8 depicts signal 1 (yellow) going into the ultrasonic speaker and signal 2 (green) coming out from the analog mic. We can see the expected phase lag on signal 2. However, the signal has a lot of jittering. It is difficult to make a good measurement of the phase difference. Also, the amplitude dips on signal 2 due to the energy loss through the air. We later discovered that the ultrasonic speaker works at a minimum frequency of 39kHz. Conversely, the analog mic has a max range of 36kHz and very poor high gain.

We replaced the two feedback resistors on both differential outputs of the op-amp (OPA344) and still did not fix the jittering issue.

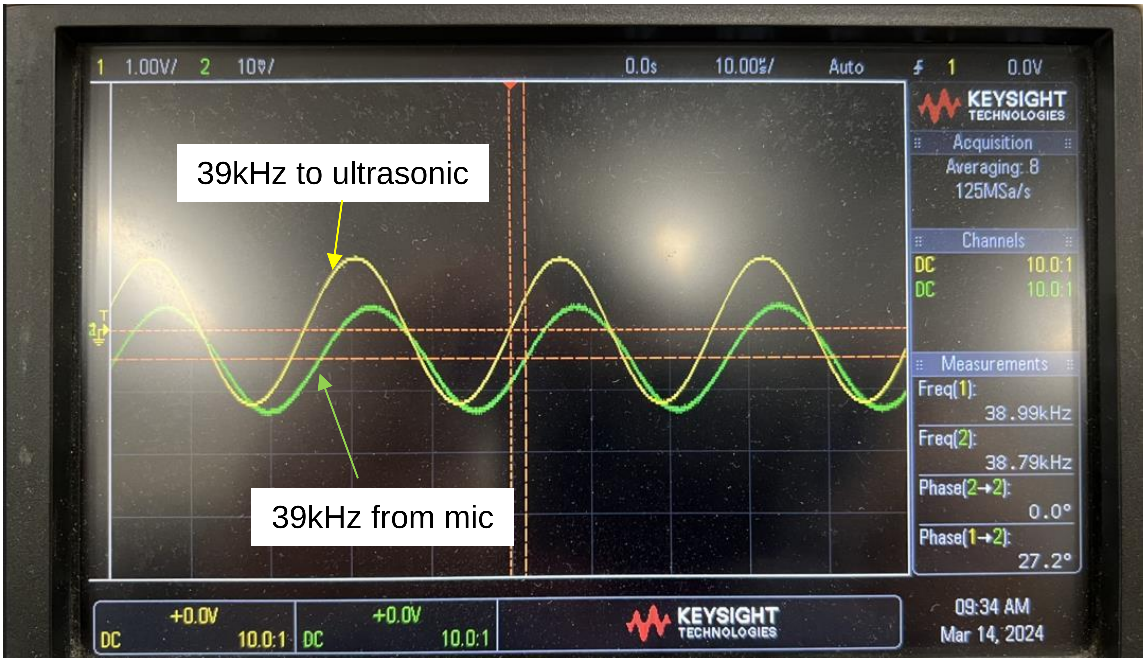


FIGURE 8: METHOD 1 – RESULT

3.2. Method 4 – Simulation and Prototyping

To test this concept, we have to make a breakout board to hold the digital sensors with the necessary filtering components added (see schematic below).

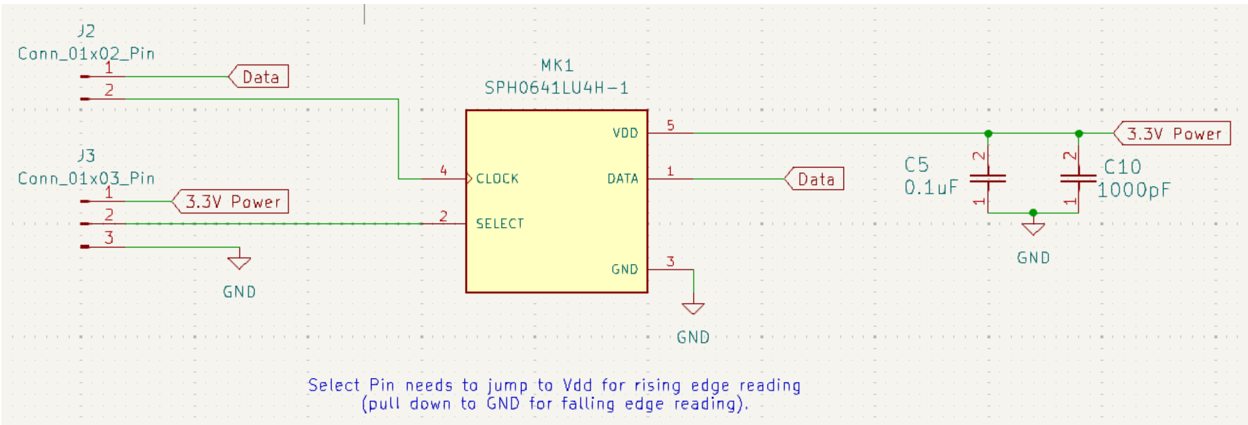


FIGURE 9: METHOD 4 – PDM RECEIVER SCHEMATIC

Figure 10 is the custom-made breakout board to host the digital mic.

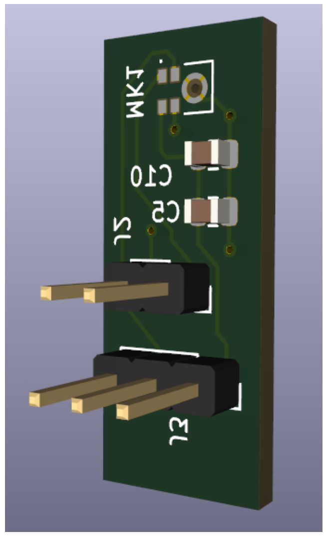


FIGURE 10: METHOD 4 – PDM RECEIVER BOARD

We simulate the PDM signals in LTSpice with the 3Mhz clock and use the logic gate to AND it with the signals at 50% duty cycle. Both PDM signals are 3.072Mhz and are converted from two 39kHz sine waves that are 120 degrees apart.

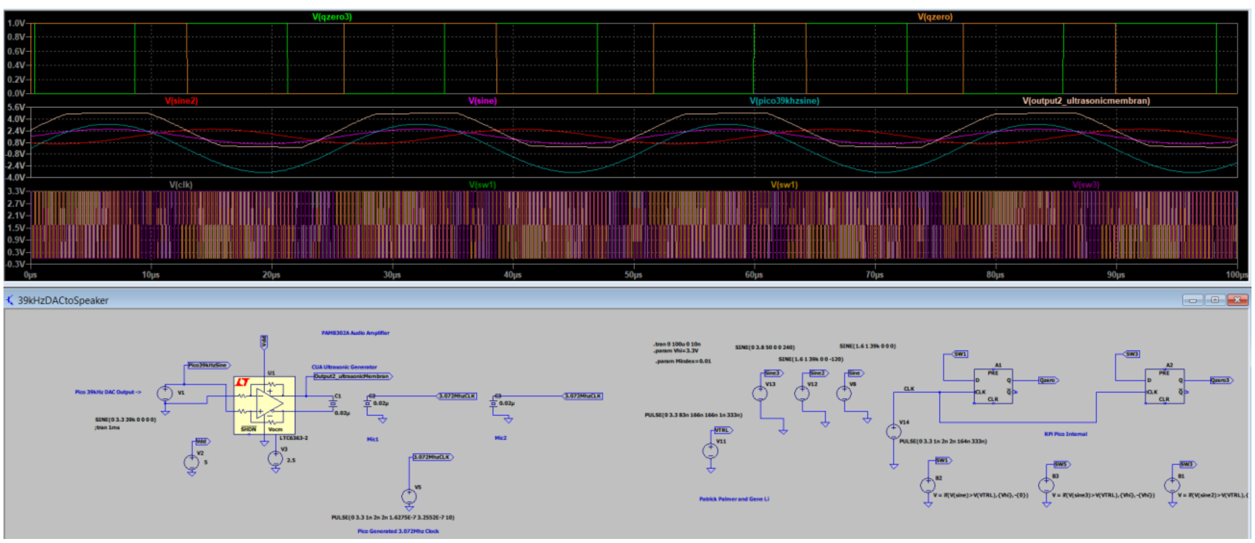


FIGURE 11: METHOD 4 – SIMULATION [1]

We have the two rising edges of the pulses and the measured time difference is 8.58us, which corresponds to a 120.56° phase shift which is close to what we have entered (120 degrees between sine and sine2). Frequency is 39kHz.

$$\Delta t = \frac{\varphi^{\circ}}{360 * f} \tag{5}$$

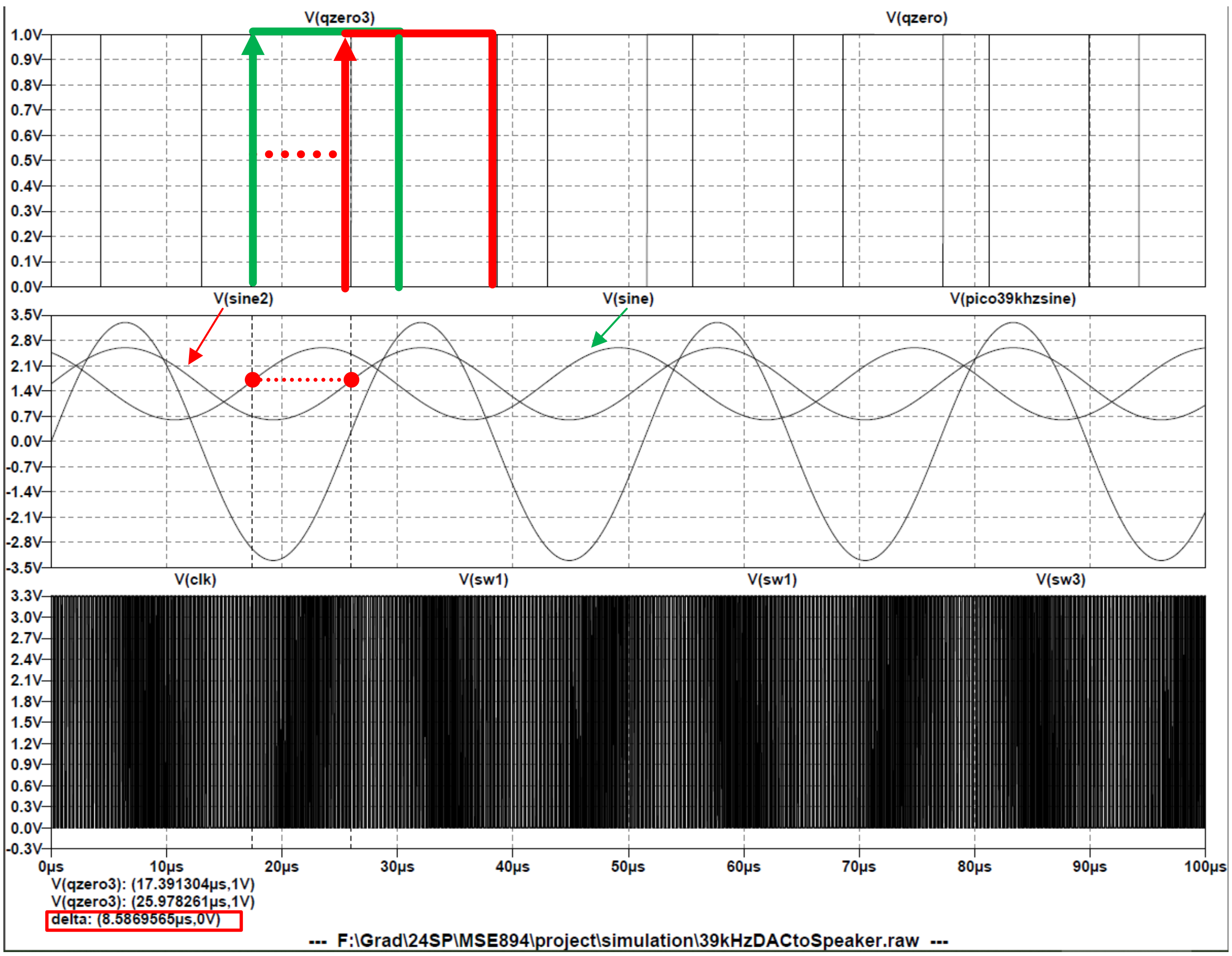
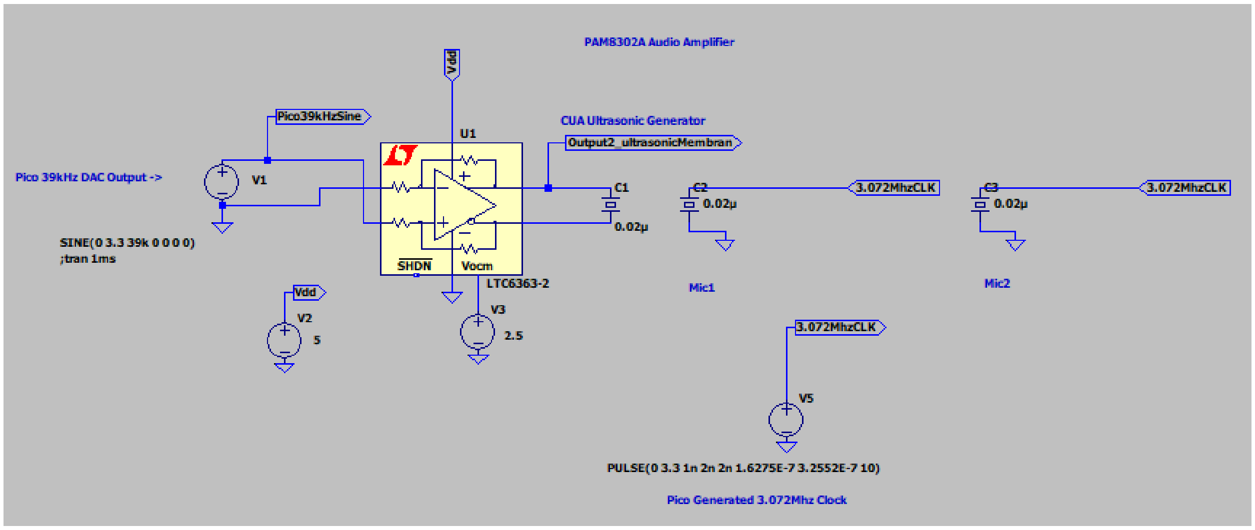


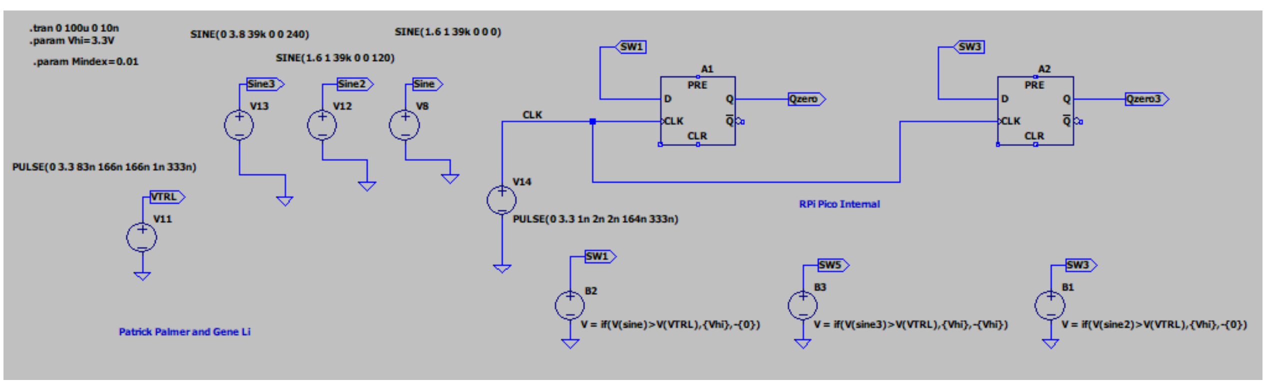
FIGURE 12: METHOD 4 – SIMULATION TIME DELAY

As we can see the simulation shows that this method would work. The simulation has two parts. The first part is the 39kHz generated from the RPi Pico to the digital amplifier and then to the ultrasonic speaker. The second part is the mechanical sinewave converted into PDM signals from the digital mic and then going to the logic gates which could potentially be inside the RPi Pico (see the two figures below).



Bitmap Image

FIGURE 13: METHOD 4 – SIMULATION PICO TO ULTRASONIC SPEAKER

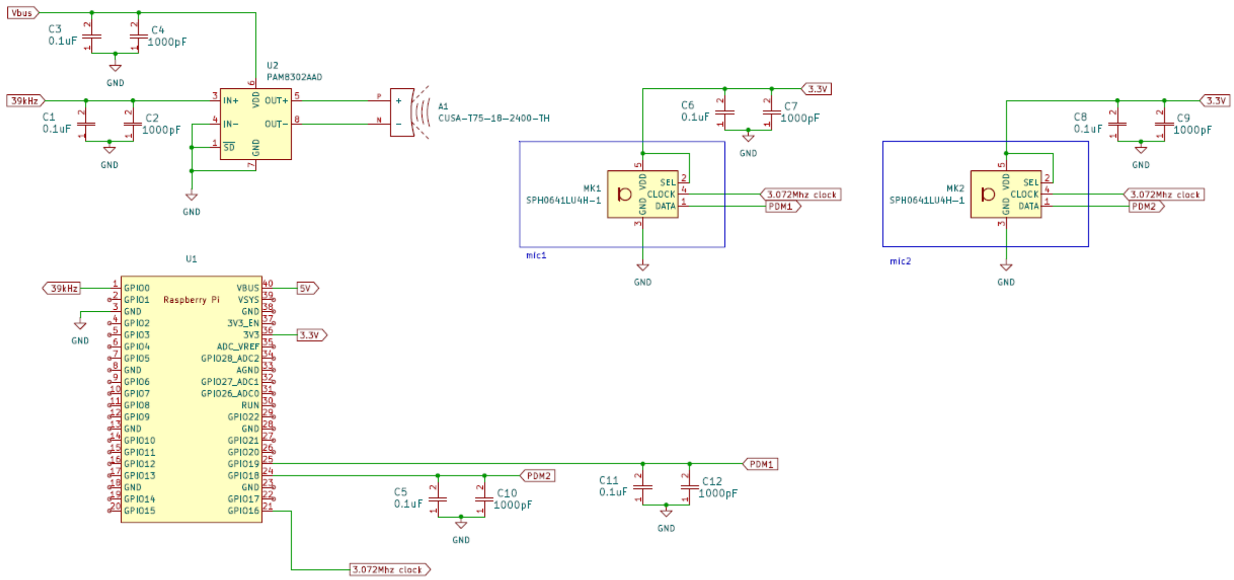


Bitmap Image

FIGURE 14: METHOD 4 – SIMULATION CIRCUIT ON PDM TO LOGIC GATE [1]

4. Future Work

The next step is to build up the circuit for the main board and assemble the digital mic receiver board to test. The circuit for the whole system is below.



Adobe Acrobat Document

*double click icon for full PDF

FIGURE 15: METHOD 4 – SYSTEM CIRCUIT DIAGRAM

Parallelly, we will develop a state machine in the RPi MCU to log the two-timing pulses. The figure below shows the state flow diagram of the state machine. On power-up, it goes to State 0, where the initialization of the clocks and GPIOs happens. A 1Mhz clock is sent to the digital mic first to bring it into normal mode. It is not recommended to start the mic with ultrasonic mode. A global clock (clock 0) will start outside of the state sequence. State 1 starts when the initiation is completed. In State 1, 39khz sine DAC is generated for the ultrasonic speaker. Also, a 3.072Mhz clock (clock1) is generated and sent to the two mics. The system waits here until the first rising edge is triggered (Pulse1 = true), meaning clock 1 is equal to PDM1 rising. When the first Pulse 1 = true, we store the global clock value into timeVal_1 and the system enters State 2. During State 2, it waits for the second rising edge (Pulse2 = true) which is when clock1 is equal to PDM2 rising. Once, Pulse2 = true, it saves the global clock value in timeVal_2 and goes to State 3. In State 3, the final state, the system computes the time difference between timeVal_1 and timeVal_2. Ideally, all the clocks are synchronized since it is coming from one source (RPi Pico MCU). However, if it is too computationally heavy for it, we can always implement an external clock using a relaxation oscillator, or a PIC12 controller, and add JK flip-flop logic gate hardware.

$$TOF = timeVal2 - timeVal1 \tag{6}$$

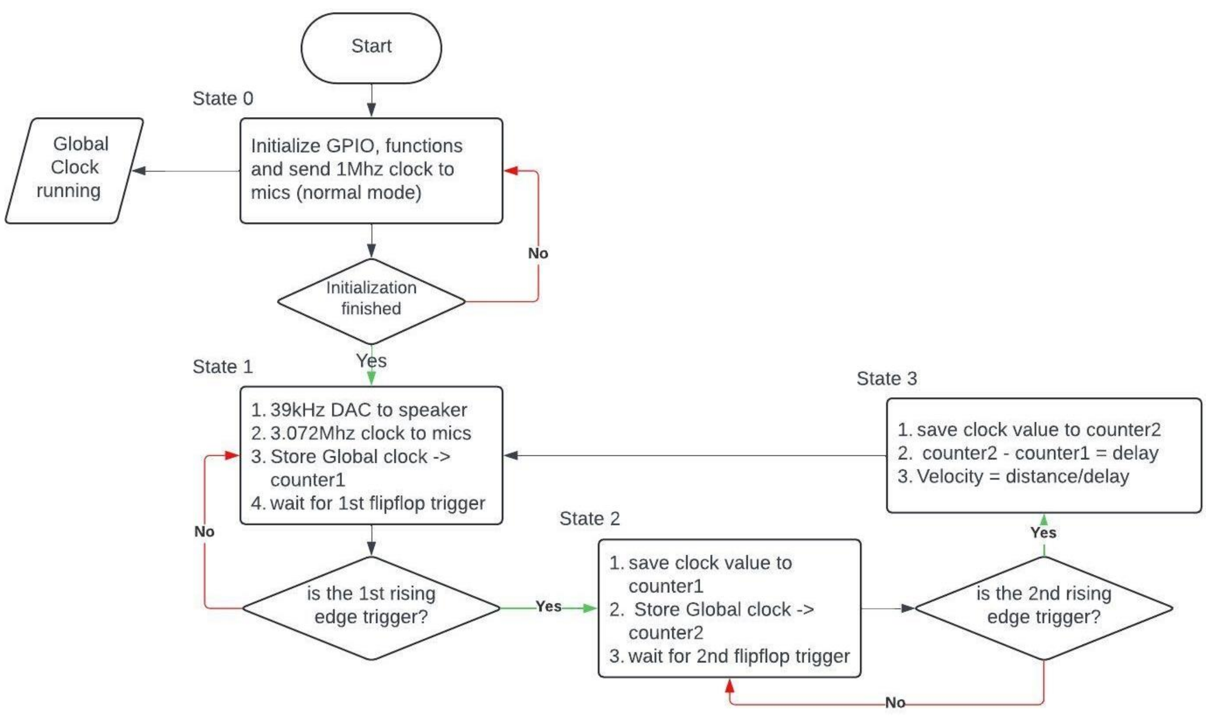


FIGURE 16: METHOD 4 – SYSTEM STATE FLOW DIAGRAM

Once we have the delta time, we can quickly calculate the wind velocity.

$$Windspeed = V_w = \frac{d=0.2m}{TOF} = 331.5 + 0.607T, \tag{7}$$

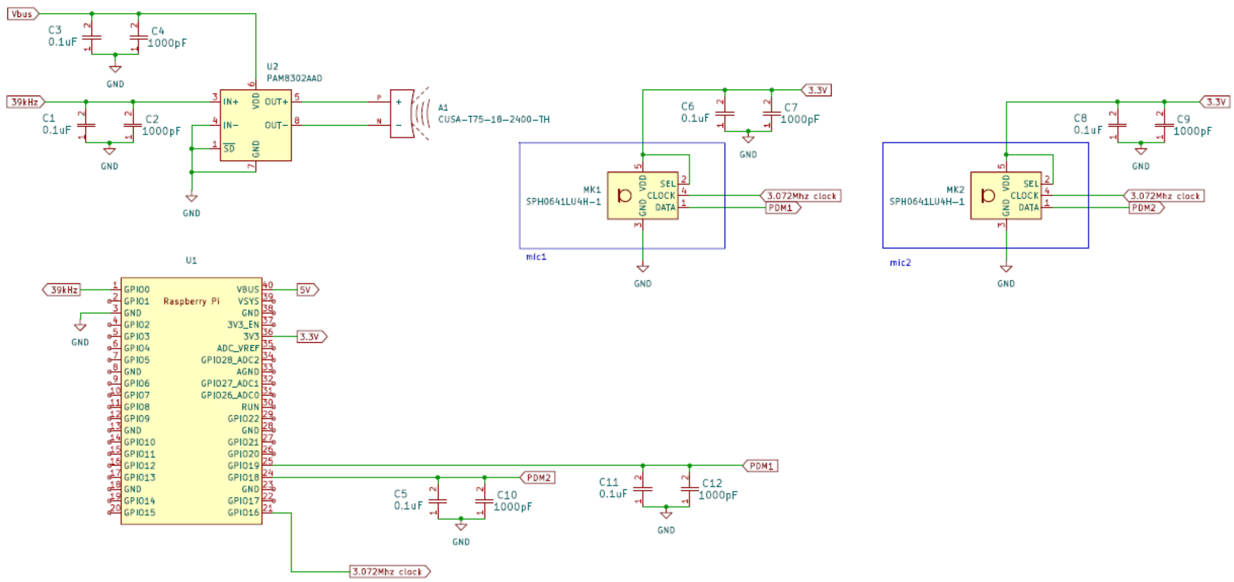
$$d = distance\ between\ the\ two\ sensors,$$

$$T = absolute\ temperature\ of\ the\ air$$

5. Summary

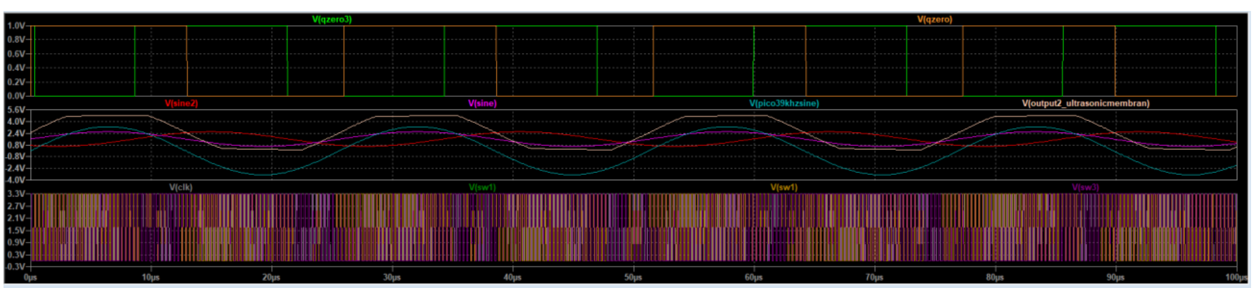
In summary, with the tremendous help from Dr. Palmer, we were able to conclude on a plausible digital approach for this project. We have tested the analog approach and found that the receiving signals from the analog mic have too much jittering. This is perhaps due to its maximum limited frequency being only 36kHz. We have also searched for other alternative devices for this reason, but most off-the-shelf microphones are only rated up to 20khz or 36khz signal. The one that is suitable for higher-frequency applications is a digital mic with a PDM output signal. It is rated between 100Hz to 80kHz. The simulation seems very promising. With some tweaking on the software side, we should be able to find the zero crossing of the two signals cleanly. This project has been quite a fun journey - from the initial validation and brainstorming of filtering and analog to digital conversion to ordering new parts and creating new PCBs. If all the clocking and DAC conversions are too computationally heavy for the RPi Pico, we will add an external clock for the digital mics and ultrasonic generator. Lastly, we have ordered the components and PCB for the digital mic and the State Machine and sinewave generator are under work.

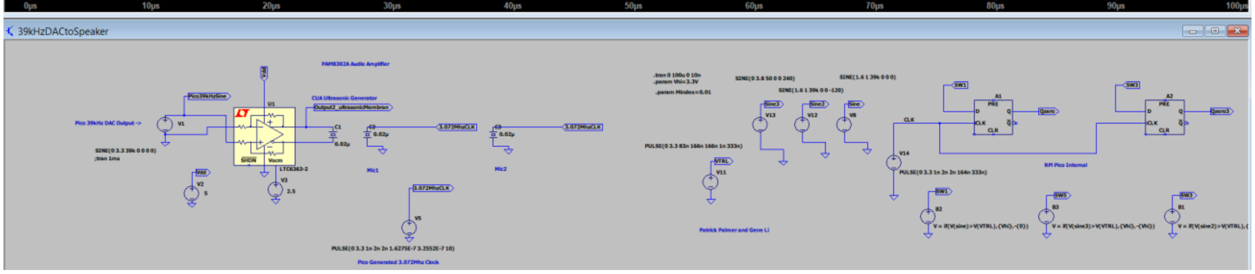
6. Appendix



Adobe Acrobat Document

*double click icon for full PDF





Bitmap Image

7. Acknowledgement

Mr. Jordan Hughes and Prof. Palmer have helped out in this greatly, especially Prof. Palmer. This is mostly his ideas and recommendations.

8. References

[1] P. P. Palmer, *Experiment 5,* Surrey, 2024.

[2] "Raspberry Pi Pico W Pinout," [Online]. Available: https://www.raspberrypi.com/documentation/microcontrollers/images/picow-pinout.svg. [Accessed 09 February 2024].

[3] "LF147/LF347 Wide Bandwidth Quad JFET Input Operational Amplifiers".

[4] "LF347, LF347B JFET-Input Quad Operational Amplifiers datasheet (Rev. C)," [Online]. Available: https://www.ti.com/lit/ds/symlink/lf347.pdf?ts=1708383698297. [Accessed 19 Feb 2024].

[5] "24-Bit Analog-to-Digital Converter (ADC) for Weigh Scales," AVIA Semiconductor.

[6] "SergeyPiskunov/micropython-hx711," [Online]. Available: https://github.com/SergeyPiskunov/micropython-hx711. [Accessed 19 Feb 2024].