Lab #1: Flex, Piezo, and Analog Filters

ECE167: Sensing and Sensor Technology
University of California Santa Cruz

This lab is the introduction to resistive sensors. A **flex sensor** measures the amount of bending–the resistance of the sensor element is varied by bending the sensor. Since the resistance is directly proportional to the amount of bend, it's sometimes called a flexible potentiometer or straight potentiometer. A **piezoelectric sensor,** often shortened to piezo sensor, uses the piezoelectric effect to measure changes in pressure, acceleration, or force. Tapping or vibrating the piezo sensor creates a large voltage spike. Piezo sensor output is a time-domain analog signal that must be captured and digitized. Over the course of this lab you will make a musical instrument where the flex sensor sets the instrument frequency and tapping on the piezo plays a note. You will also be mathematically modeling simple filter stages and then experimentally validating your model to determine how well it matches reality.

Warning: The piezo sensor creates a high enough voltage spike such that it will kill pins on the STM32. DO NOT hook it up without the appropriate circuit in place to snub that voltage spike (see Fig. 2).

Hardware needed

Nucleo-64 + IO shield, speaker, audio amp (you will have to use a potentiometer to change the volume), breadboard, flex sensor (with mount), piezo sensor (with 1 MΩ resistor), resistors, and capacitors.

Part 1: Flex Sensor

Flex sensors are sensors that change in resistance depending on the amount of bend on the sensor. They convert the change in bend to electrical resistance - the more the bend, the more the resistance value. They are usually in the form of a thin strip from 1"-5" long that vary in resistance from approximately 10 to 50 kΩ.

The flex sensor acts as a variable resistor whose resistance changes with the amount of flex (or angle) that the sensor is bent around. One side of the flex sensor is printed with a polymer ink embedded with conducting particles. When flat, the particles are closer together, giving a lower resistance. When the flex sensor is bent, the particles are farther apart from each other, giving a higher resistance. The change is non-linear but monotonic (resistance increases with bend, but not on a straight line).

The resistance of the flex sensor changes when the metal pads are on the outside of the bend. Bending the flex sensor with the metal pads on the inside of the bend will give poor readings and can damage the sensor. Also, be careful with the leads on the sensor, they are fragile and can be easily broken.

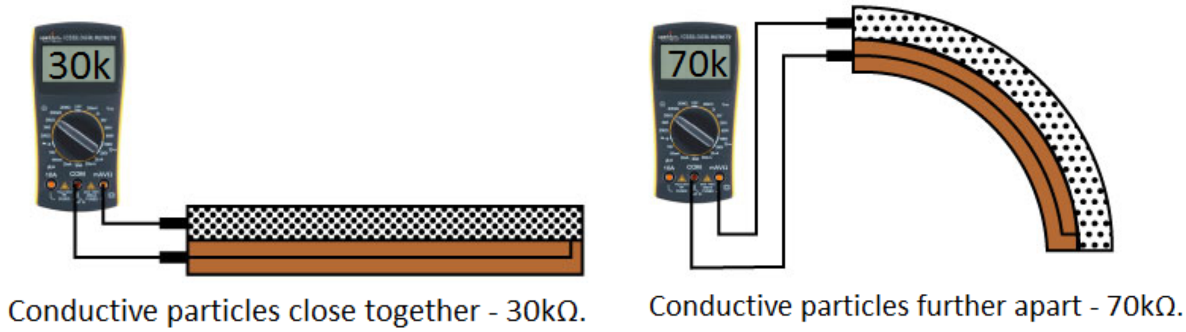

Figure 1: Operating Principle for Flex Sensor

1.1 Flex Sensor Assembly

The flex sensor needs to be assembled. There is a blue clincher connector with two pins that can be placed on a breadboard. Please follow the instructions at learn.sparkfun.com/tutorials/flex-sensor-hookup-guide under the Amphenol CFI Clincher Connector section.

1.2 Flex Sensor Regression Model

You will need to map the bend in the flex sensor to the resulting output. This will involve creating a list of input and corresponding outputs, and using a regression to find the relation between the two. Note that you need lots of points, and that Excel or other spreadsheet software is a good way to come up with least squares regressions.

Hint: you can use a protractor to measure the angle of bend in your flex sensor. These can be found online and printed.

1.3 Speaker Based on Flex

Similar to Lab 0, you will be tasked with reading from an A/D pin on the MCU and using the values to change the tone of the speaker. However, instead of using A/D readings of potentiometer on the IO shield, you will use the flex sensor that you have now found a regression model for. What do you hear when you use the flex sensor readings to change the tone of the speaker? Is your linearization of the flex sensor correct? Is it noisy? Is it smooth? If it's noisy, try smoothing it using the software smoothing you learned about in Lab 0. Make sure to answer these questions in your lab report and explain how you came to those conclusions.

Part 2: Piezoelectric Sensor

Piezoelectric sensors generate (large) voltages when deflected or vibrated. "Piezo," is Greek for "pressure." The piezoelectric effect was discovered by the Curie brothers more than 100 years ago. They found that quartz changed its dimensions when subjected to an electrical field, and conversely, generated electrical charge when mechanically deformed. One of the first practical applications of the technology was made in the 1920's by Langevin, who developed a quartz transmitter and receiver for underwater sound—the first SONAR.

Before World War II, researchers discovered that certain ceramic materials could be made piezoelectric when subjected to a high polarizing voltage, a process analogous to magnetizing a ferrous material. Two main groups of materials are used for piezoelectric sensors: piezoelectric ceramics and single crystal materials. The ceramic materials have a piezoelectric constant/sensitivity that is roughly two orders of magnitude higher than those of the natural single crystal materials and can be produced by inexpensive sintering processes. The piezoelectric effect in piezoceramics is "trained," so their high sensitivity degrades over time. This degradation is highly correlated with increased temperature.

Piezoelectric sensors can be used to measure deflection, acceleration, vibration, and can be shaped into almost any geometric shape as needed for the application. They are very rugged, and immune to electrical, magnetic, and radiation fields—this makes them useful in harsh environments.

The voltages these sensors produce can be very high. Care needs to be taken to ensure that these voltages don't reach the inputs to sensitive electronics without snubbing them to tolerable levels (see Fig. 2).

Warning: If you do not snub the output of the piezo device, ***you will kill the input pin of the MCU****.*

This is <u>very</u> important!

2.1 Capture the taps (analog/digital)

The next sensor to implement is the piezo vibration sensor. By connecting the piezo sensor to a breadboard and using an oscilloscope you can use the trigger feature on the oscilloscope to capture the pulse that is generated upon flicking the piezo sensor. Recall that when using the oscilloscope, one probe is connected to the positive pin, and the alligator/ground cable is connected to the negative pin. Try doing this a number of times and recording the average voltage spike, maximum voltage spike, and minimum voltage spike. Make sure that your measurements are peak-to-peak; otherwise you might be recording voltages that appear to have a lower magnitude than in reality. Look at the STM32 reference manual, which can be found on Canvas. What is the maximum analog input voltage? How does this compare to the values you just measured from the piezo? Include this information in the lab report.

You need to use this voltage spike to trigger your MCU to play a note. But as you have just seen with the oscilloscope, the voltage from the piezo is too high. To snub the high voltage you will use a 1 Mohm resistor in parallel with the sensor. The circuit you will use is shown below in Fig. 2. The bottom of the sensor (labeled as "–") is connected to ground along with the bottom of the resistor. The top of the sensor ("+") is connected to the top of the 1 Mohm resistor and an analog input pin on the MCU.

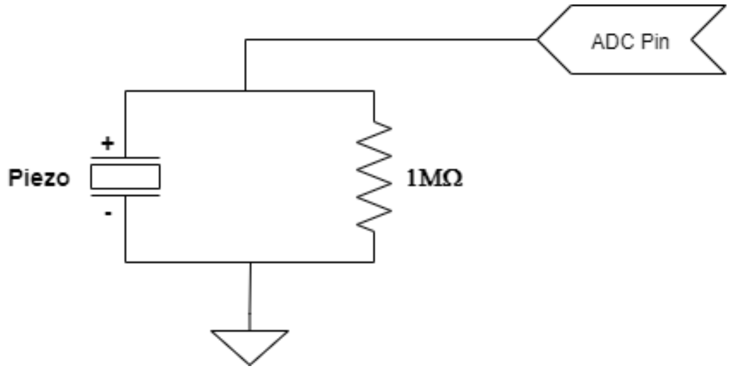

Figure 2: Circuit for snubbing Piezo sensor input to STM32

Part 3: Musical Instrument Redux

You are now going to use both the piezo and flex sensors to create a new musical instrument given the requirements below:

- "Select" a tone based on the flex sensor (that is, once a tone is activated, it is proportional to the amount of flex the sensor has undergone).
- Reliably activate a selected tone with a tap of the piezo sensor. Like a piano, the tone should turn off after a short delay. This duration should be reset if the piezo is tapped again within this period.
- Produce sound that is clear, audible, and smooth.

Part 4: Simple Analog Filtering Analysis

For this part of the lab, you will first be generating a mathematical model of a simple band pass filter; later, you will be experimentally validating your model.

4.1 Low-Pass Filter

In your prerequisite classes, you learned that a low-pass filter (single pole) allows frequencies below the corner frequency to pass unchanged (unity gain or 0dB), but frequencies at the corner see a loss of -3dB, and are further attenuated at a slope of negative 20dB/decade as the frequency increases above the corner frequency. This is often shown in the log-log plot of magnitude vs. frequency.[1]

Low pass filters filter out high frequency noise from a signal, and can be implemented as a voltage divider with a capacitor in the lower leg of the voltage divider. Using complex impedances for the capacitor, and KCL at the V_{out} node, find the transfer function of the single pole low pass. For the frequency, use the complex variable "s" for "jω" once you have the transfer function figured out.

Pick a corner frequency that you can make using capacitors and resistors that you have in the lab, and plot the theoretical magnitude vs. frequency curve from your transfer function. Note that we are not interested in the phase, but only the magnitude; you can solve for this analytically and plot it out.

[1] This is the magnitude half of the bode plot, and the log(ω) on the horizontal axis is used so that the slopes are straight line asymptotes. The magnitude needs to be in a log scale as well (such as dB). These straight line asymptotes means that you can draw the magnitude vs frequency plot easily by hand and even do the convolution via simple addition on log-log paper

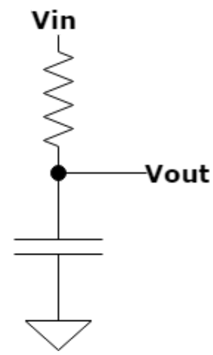

Figure 3: Low-Pass Filter Schematic

4.2 High-Pass Filter

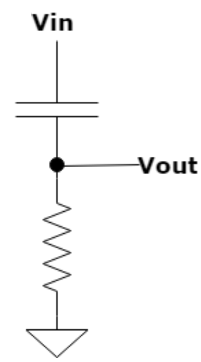

Figure 4: High-Pass Filter Schematic

A high-pass filter (single pole) attenuates frequencies below the corner frequency (-20dB/decade), with a loss of -3dB at the corner frequency, and high frequencies to pass unchanged (unity gain or 0dB). Again this is often shown in the log-log plot of magnitude vs. frequency. High pass filters are often used to filter out bias (DC offset) from a signal. They can be implemented as a voltage divider with a capacitor in the upper leg of the voltage divider. Using complex impedances for the capacitor, and KCL at the V_{out} node, find the transfer function of the single pole high pass. Again, for the frequency, use the variable “s” for “jω” once you have the transfer function figured out.

Plot the theoretical magnitude vs frequency curve from your transfer function, and label the corner frequency.

4.3 Band-Pass Filter

You can build a simple band pass filter by cascading a simple high-pass and a simple low-pass filter together.[2] As a heads up, the math gets messy, so be neat and careful and be sure to check your work. In theory, you can just add the magnitude plots together (they are convolved together, but because they are in the frequency domain they are multiplied, and on log-log multiplication is the same as addition) to get the

[2] Usually the high-pass is first and the low-pass is downstream of the high pass so that you can get rid of any DC bias in the signal before doing any amplification.

combined response. Unfortunately, the downstream loading of the upstream circuit affects the simple analysis you have done above. You are going to verify this experimentally, but we also want you to figure it out. Assume both low-pass and high-pass are independent, and multiply the two transfer functions analytically and once again plot the theoretical magnitude vs. frequency. That is, assume that V_{out} of the first filter is V_{in} of the second. Make sure you call out the natural frequency (ω_n) and the quality factor (Q) of the transfer function.[3]

The actual band pass filter is not a simple convolution of the high pass and low pass, as the downstream loading affects the frequency response of the upstream circuit. As usual, the truth is more complicated than the ideal, and so we have to do yet more math to get this to work. The band pass filter is shown as a more complicated network, and thus must be analyzed more carefully. You will need to use KCL at each of the nodes and solve the equations simultaneously to get the full transfer function. Here you should assume that the resistors and capacitors are each different from each other (R1, R2, C1, C2) and come up with the transfer function.

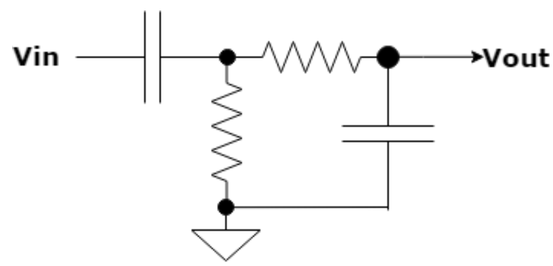

Figure 5: Band-Pass Filter Schematic

Specify both the natural frequency (ω_n) and the quality factor (Q) of the transfer function. Now assume that you have the same R and C for both parts of the filter, and plot the theoretical magnitude vs. frequency and show how it differs from the simple convolution above. Note where the new -3dB point is, and if there was a change in Q.

4.4 Experimental Validation of Analog Filtering

For this part of the lab, you will be validating your magnitude vs frequency experimentally using simple resistors and capacitors. First set up your low pass filter (and make sure you match the resistors and capacitors you chose for your analysis). Use the signal generator and oscilloscope to take a number of measurements and plot these. How does it compare to the theoretical plot?

Do the same with the high pass filter (don't dismantle your low pass). Again, compare to the theoretical. Now hook the high pass to the low pass and generate the magnitude vs. frequency data. Does this match your calculations?

Configure the onboard op-amp as a follower/buffer, and insert it in between the high pass and low pass stages. Regenerate the magnitude vs. frequency data (this should match the original convolution that you did).

---

[3] The quality factor (Q) is defined for a transfer function as $1/(2\zeta)$.

Part 5: Check-Off and Lab Report

Demonstrate your fully functioning instrument to the course staff and be ready to explain your calculations if asked. Add your check-off details and the commit ID to your lab report (which you should be writing as you go). Make sure to answer any questions posed throughout this lab manual. Also discuss:

- What were your transfer functions for the sensors?
- How many measurements did you take? What were they?
- How accurate is your sensor model?
- What is your error margin? If your filters didn't match the theory, why do you think not?

Don't forget that part of your grade will be on code quality and our ability to compile your code.

Acknowledgements

Images in Fig. 1 courtesy of sparkfun https://learn.sparkfun.com/tutorials/flex-sensor-hookup-guide.
Figure 2-6 courtesy of Gabriel Hugh Elkaim

Lab 1 Rubric

- 40% check off (10 point scale)
  - Sensor flexing changes tone (4 points)
  - Audio activates when piezo is tapped (2 points)
  - Audio stops automatically after delay (2 points)
  - Audio keeps playing continuously if you tap the piezo continuously (1 point)
  - Audio quality: sound is clear, audible, and smooth (1 point)
- 15% code quality (comments, readability)
- 15% code compiles
- 30% lab report
  - General format followed; all sections of lab addressed
  - Writing detailed enough for replication
  - Image quality
  - Writing quality (grammar, spelling, appropriate tone)