UNIVERSITY OF CALIFORNIA, SANTA CRUZ
BOARD OF STUDIES IN COMPUTER ENGINEERING

CMPE167/L: INTRODUCTION TO MECHATRONICS

LAB 1: FLEX, PIEZO, AND (SIMPLE) ANALOG FILTERS

PURPOSE:

This lab is the introduction to resistive sensors. A flex sensor is a straight potentiometer, changing resistance with deflection. It is a nonlinear sensor and you will need to accommodate that nonlinearity. A piezo sensor is a time based analog signal that must be captured. Each tap generates a voltage spike that must be clipped to prevent damage to systems. Students over the course of this lab will make a musical instrument where the flex sensor sets tone and each tap plays said note.

Lastly, you will be mathematically modelling 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 uno32. DO NOT hook it up without the appropriate circuit in place to snub that voltage spike.

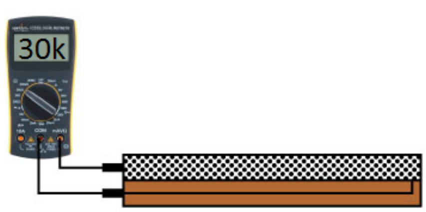
HARDWARE PROVIDED AND INTENDED USE:

Uno32, Speaker, Audio AMP (you will have to use a potentiometer to change the volume), Protoboard (bread board), Flex Sensor (with board mount), Piezo tap sensor (with 1Mohm resistor and diode). MCP6004 Quad OpAmp, Resistors, and Capacitors.

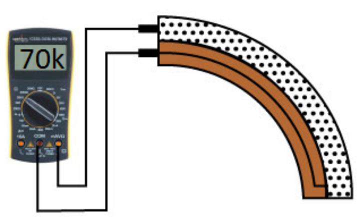
BACKGROUND FLEXSENSOR:

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 kilohms. They are often used in gloves to sense finger movement.

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).



Conductive particles close together - 30kΩ.



Conductive particles further apart - 70kΩ.

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 with give poor readings and can damage the sensor (don't do it). Also, be very careful with the leads on the sensor, they are fragile and can be easily broken.

Open MPLAB-X and Create a New Project:

Follow the instructions outlined in CMPE167_MPLABXNewProjectInstructions.pdf. Be sure to understand the difference between absolute and relative paths when making the project, or you will risk deleting your project when running the code updater.

Part 1 - Linearize the flex sensor

The flex sensor is nonlinear. You have to linearize it. The basic idea behind doing this is to find a transfer function that describes how the input relates to the output. This can involve creating a list of input and corresponding outputs, and using regression to find the relation between the two. Refer to your book for more information on linearization and regression.[†] Note that you need lots of points, and that Excel can be a very good resource for coming up with regressions.

The flex sensor needs to be assembled. There is a blue clincher connector with 2 pins that can be placed on a breadboard. You need to connect this to the flex sensor. If you are familiar with crimping, then using the clincher should make sense. However, to mitigate the possibility of breaking the flex sensor you will need to **consult with the TA** during lab section to learn how to properly connect the clincher to the flex sensor. There are actually not many good videos online showing how to use these, but if you find one, please post it on piazza for others to see.

Background PiezoElectric Sensor:

Piezoelectric sensors generate (large) voltages when deflected or vibrated. "Piezo," Greek for "pressure," electricity 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 another Frenchman, 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 (such as PZT ceramic) 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 piezoeffect 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 piezos useful in harsh environments.

Due to the nature of the response vs frequency, most piezo sensors are used in the flat region away from their resonant peaks, and the DC response is almost zero (that is, they produce a signal to a *change* in deflection). The output of the piezo element can be increased by laminating strips together in clever ways.

[†] If you have the inputs and outputs, you can use a LEAST SQUARES to find the relationship between the two. More precisely, you can find the best parameters of the fitting function you have to match input to output. A decent discussion of least squares can be found at https://en.wikipedia.org/wiki/Least_squares. It is a useful tool and will be used in the class in a later lab.

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.

PART 2 - CAPTURE THE TAPS (ANALOG/DIGITAL)

The next sensor to implement is the Piezo vibration sensor. By connecting the piezo sensor to a bread board 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. Include this information in the lab report. You will use this voltage spike to play a note using your Uno32; you will need to be able to reliably capture the taps/flicks of this sensor to trigger the note.

PART 3 - TONE OUT SPEAKER BASED ON FLEX

Similar to Lab 0, you will be tasked with reading from an A/D pin on the micro 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 linearized (and probably filtered). What do you observe (hear) when you use the flex sensor readings to change the tone of the speaker? Is it noisy? Is it smooth? Is your linearization of the flex sensor correct? How could you improve the sound? Make sure to answer these questions in your lab report and explain how you came to those conclusions.

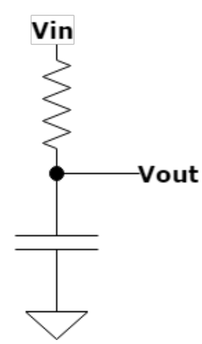
PART 4 - TONE OUT BASED ON TAP

For this part you will combine the previous two parts. You will be tasked with programming your micro such that you can:

1. "**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).
2. Reliably **activate** a selected tone with a tap of the piezo sensor. The tone should turn off after a short duration. This duration should be reset if the piezo is tapped again within this period.
3. Produce sound that is **clear**, **audible** (hopefully not too loud, but definitely not too quiet, because you are using an audio amp), and **smooth**.

PART 5 – SIMPLE ANALOG FILTERING (THE MATH)

For this part of the lab, you will be first generating a mathematical model of a simple band pass filter; later, you will be experimentally validating your model. We will then have you do a more complicated active filter, and come up with the mathematical model for that as well.



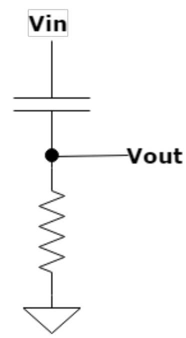
<u>Low-Pass Filter</u>: From your EE101 class, you should know that a low-pass filter (single pole) can be shown to allow frequencies below the corner frequency to pass unchanged (unity gain or 0dB), a loss of -3dB at the corner frequency, and a ramp down (slope of negative 20dB/decade) as the frequency increases above the corner frequency. This is often shown in the log-log plot of magnitude vs. frequency.[‡]

Low pass filters are used extensively to 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 variable "s" for "jω" once you have the transfer function figured out.

[‡] 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.

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 that you have from your transfer function.

High-Pass Filter: Again, from your EE101 class, you should know that a high-pass filter (single pole) can be shown to attenuate frequencies below the corner frequency (-20dB/decade), 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.

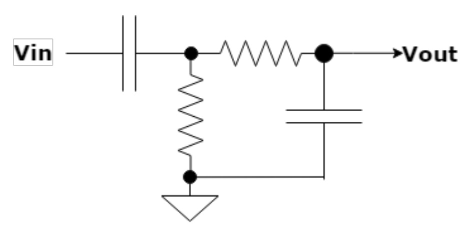


High pass filters are used extensively to filter out bias (DC noise) from a signal, and can be implemented as a voltage divider with a capacitor in the upper leg of the voltage divider. Using complex impedances for the capacitor, and KCL at the V_{out} node, find the transfer function of the single pole high pass. Again, for the frequency, use the variable "s" for "jω" once you have the transfer function figured out.

Using the same corner frequency you used for the low pass filter, plot the theoretical magnitude vs frequency curve from your transfer function.

Band-Pass Filter: You can build a simple band pass filter by cascading a simple high-pass and a simple low-pass filter together.[§] 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 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.[**]

(1) 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. Yes, the math gets messy and we know that. Make sure you call out the natural frequency (ω_n) and the quality factor (Q) of the transfer function.[††]



(2) 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.

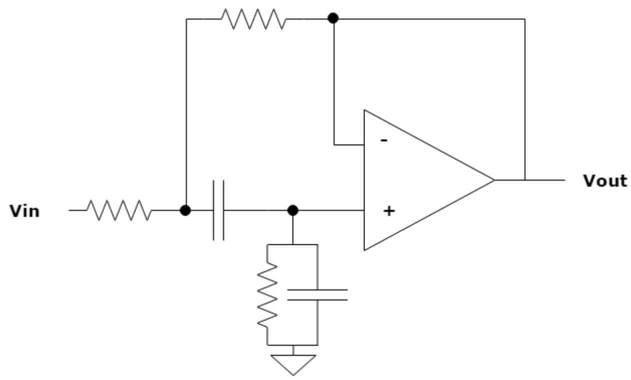
It should be in the form of: $\frac{\frac{\omega_n}{Q}s}{s^2+\frac{\omega_n}{Q}s+\omega_n^2}$. Call out both the natural frequency (ω_n) and the quality factor (Q) of the transfer function. Now assume that you have the same R and C for both parts of the filter, and plot the theoretical magnitude vs frequency 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.

[§] 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.

[**] The math is going to get messy. Make sure you do this neatly and carefully. And check your work.

[††] The quality factor (Q) is defined for a transfer function as 1/(2ζ).

(3) The last heavy math thing you are doing to do is to analyze a simple Sallen-Key active band pass filter and see what you get in terms of the transfer function. The Sallen-Key is a variant of the VCVS filter, and has the advantage that each 2nd order pair can be cascaded without loading issues due to the buffer of the opAmp (and the fact that it is formulated as a non-inverting gain amplifier with infinite input impedance).‡‡



In order to analyze the filter above, you will need to solve for the voltage at each of the nodes using KCL and complex impedance and then substitute back to get the final equations in the form of a transfer function.

Again, you should get a transfer function of the same form, and be able to pick off the natural frequency and quality factor of the filter from the equations in terms of the resistors and capacitors. How is this different from the passive one above? Is it possible to set Q and ω_n independently? Plot the magnitude vs frequency for the theoretical response assuming the same R and C. Again compare it to the others you plotted and see how it is different.

PART 6 – EXPERIMENTAL VALIDATION OF ANALOG FILTERING

For this part of the lab, you will be validating your magnitude vs phase 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 scope 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 MCP6004 OpAmp 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).

‡‡ You will note if you look closely at the Sallen-Keys band pass filter you will see that it is topologically different than the high pass or low pass variants in that it has a capacitor and resistor in parallel in one of the legs that is usually just a single element (resistor or capacitor). This is not the only version of Sallen Keys bandpass filter, but that all of the single opAmp ones will have 3 resistors and 2 capacitors. They are usually not set as unity gain on the opAmp, as there is significant attenuation at the peak, and thus the opAmp is used to amplify the signal to some desired point at the peak. The Wikipedia entry found at https://en.wikipedia.org/wiki/Sallen-Key_topology is worth reading. The classic TI ApNote in 1999 (http://www.ti.com/lit/an/sloa024b/sloa024b.pdf) goes through the Sallen-Keys topology including non-idealities; if you get into designing filters, it is a good read.

Extra Credit: Hook up the Sallen-Keys filter you designed/analyzed and run it with the signal generator as an input and measure the output. Compare this to your theoretical plots.

Part 7 - Your Lab Report:

Congratulations! You have now completed lab 1. By now you have very accurate models of your flex and piezo sensors that are linear and produce good sounds on the speaker. You also have a good theoretical understanding of simple analog filtering and some experimental validation behind you. Explain how you did this lab, what challenges you observed and overcame, and make sure to answer any questions posed throughout this lab manual. 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?