ECE 445 - Senior Design Project Laboratory

Final Report

May 1st, 2024

Self Tuning Violin

Team No. 9

Kevin Lyvers (klyver2)

Ginny Bytnar (bytnar2)

Erik Kwiatkowski (erikk3)

TA: Tianxiang Zheng

Abstract

This final paper documents the process that Team 9 took to create a *self tuning violin*. The paper provides an overview of the goals and objectives to be achieved as well as how each objective was completed. Specifically, we discuss our design choices, challenges faced, solutions, overall functionality, and cost.

1. Introduction 1
2. Design 3
3. Verification 8
4. Costs and Schedule 11
4.1 Costs 11
4.2 Schedule 13
5. Conclusions 15
6. References 16
Appendix A - Requirements and Verification Tables 17

1. Introduction

A common issue faced among beginner string players is tuning their instrument with the tuning pegs. These pegs require a certain level of finesse and muscle to turn without damaging the instrument or strings. This led to our project's creation - a self tuning violin. This violin detects the pitch of the open string's sound and is able to turn mechanical tuning pegs to change the pitch correctly. This works by using several submodules consisting of a microphone module, a microcontroller module, a motor module, and a power module, as shown in the block diagram below. The particular connections are laid out in Figure 1.

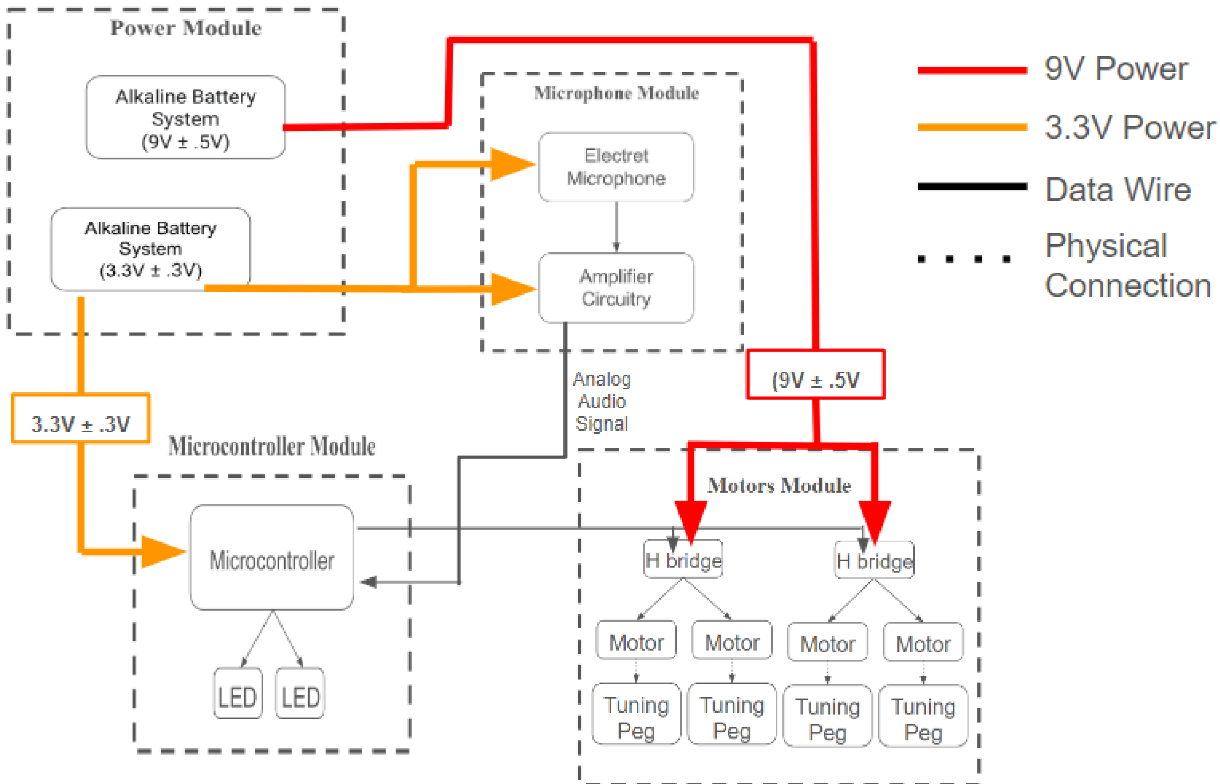

Figure 1: Finalized Block Diagram

Figure 1 went through multiple revisions, with the final changes being alterations to the microcontroller module, power module, and motor module. In the microcontroller module we decreased the number of LEDs from our original four to two, due to a limited number of outputs and current draw of the microcontroller. The power module originally consisted of one high voltage battery that was then fed into a voltage regulator to control the several outputs needed for the module's function. However due to issues in shipping and part ordering we did not receive the necessary regulator in time, so the change was made to use two individual battery clusters, supplying necessary voltage levels of 9 V and 3.3 V. Finally the motor module was changed to use only two dual channel H-bridges rather than four single channel H-bridges. This was due to changes in our model of H-bridges midway through the manufacturing process. This change led us to have each H-bridge control two motors rather than a single motor, meaning we could reduce the total number of H-bridges.

Looking at the microphone module, its aim is to produce an amplified analog output signal to the microcontroller. This was accomplished by thorough research of which microphone we purchased. Microphones come in various forms, such as active and passive outputs along with various different line-level voltages. We found our ideal microphone of a cheap low-current active instrument-level output microphone. Once the product arrived we verified its accuracy by observing its output with an oscilloscope. This was so the primary frequency of the sound that could be verified to ensure the microphone had high enough fidelity to correctly pick up the sounds necessary.

The microcontroller module consists of a ATMega 328 microcontroller that takes in the analog input from the microphone module and processes the sound into its composite frequencies. The frequencies are then analyzed for an accurate response to be sent to the module modules to turn the strings correctly. The correctness of this module is verified in several ways. First the decomposition of frequencies is checked against a verified decomposition of the sound that will be touched more in depth later, however the fundamental frequencies should differ no more than 2.5%. The microcontroller also needs to control the two LED lights the project has. As a simple verification, we needed to ensure the lights would turn on or not. Since the LEDs are so simple and consistent, we consider any failure to control these lights as a failed verification. Lastly, the microcontroller must send accurate signals to the motor modules to within 5º. This is verified with a clever equivalent of determining the time accuracy of which we control the motors. Due to the fact the motors spin very consistently at a set RPM it can be determined that 166 milliseconds is our time equivalent for 5º for our motors and is accurate enough for the criteria. The microcontroller also needs to calculate if the note is flat or sharp compared to the ideal frequencies of the strings[1] which is trivial and no more than subtraction.

The motor module consists of the H-bridges and motors which work together to take input from the low-current microcontroller and turn the high-current draw motors. To verify the correct operation, it is checked once again that the motors can be turned within 5º or 166ms of precision. Additionally, when the motors first arrived, we verified that they turn constantly at 5 RPM with varying levels of voltages in case the H-bridges become inconsistent.

The purpose of the power module is to simply output the two consistent voltages accurately. The module outputs a high 9V output to the H-bridges used to turn the motors, and a lower 3.3V for the microcontroller and microphone input voltages. The module's implementation is little more than an array of batteries and resistors. The module's success is determined how accurately the module can output voltages, in our specific case 9V should be within a tolerance of 0.5V and the 3.3V line should be within 0.3V of the expected value.

2. Design

In this section, we decided to combine the Design Procedure and Design Details, as we believed it would be very repetitive to write about these two sections separately.

Our design process for the microphone module took place mainly in the planning phase of our project. We do not have the materials or knowledge to design or troubleshoot our own microphone so we knew we had to acquire one that met our specifications. One of our team members had in-depth knowledge of different kinds of microphones and the industry standard for output lines. Some microphones output a high voltage output to drive speakers, but most output a very small “microphone-line” signal that is extremely weak, usually fed into an amplifier to strengthen the signal. We did not need our microphone to produce a signal strong enough to power a passive speaker, but we needed a sufficient strength to assist the microcontroller’s digital signal processing of the signal. We also prioritized shipping time into our search. We were fearful of shipping delays with our tight deadline of our project. A microphone was found that fit these needs. It was going to ship fast, had the audio resolution and frequency range we needed, and had a potentiometer to control the audio output’s gain or amplification. This met our criteria perfectly and we did not consider other microphones or other ways of measuring the string’s frequency beyond our preliminary research and ordering of the microphone that was used in our final project. The pinout of the microphone is shown in Figure 2.

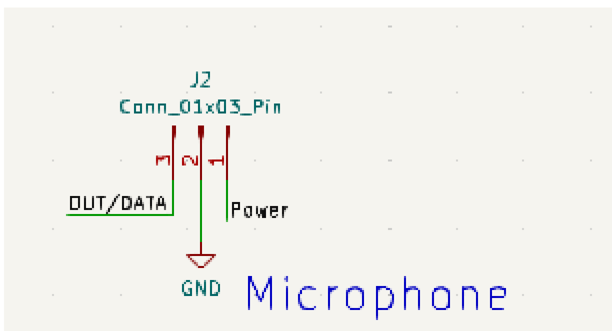

Figure 2: Microphone Subsystem Schematic

The microcontroller module had a similar design path as the microphone module. Majority of the design process happened in the planning stage and very little was changed once the parts were received. Almost all microcontrollers can output digital signals that we were planning on sending to the motor modules, so that was of very little concern. Our two main considerations were the ability for analog input and the necessary computation power to run the frequency decomposition. One of our teammates had good knowledge of the capabilities of Arduinos and knew an Arduino Uno can sufficiently do both of those tasks. We then choose to use the microcontroller from the Arduino Uno board, which is an ATMega 328, for several reasons. It had multiple analog input pins that could also be used as analog outputs if it were needed. The microcontroller had many digital I/O pins that could be used for all the different functions necessary for the project. The pinout of the microcontroller is shown in Figure 3. The Arduino foundation also has a library built for Fast Fourier Transforms (FFT) which we would later use and modify to find the frequency information from the analog input[3]. Another major advantage from using this microcontroller was the ease of prototyping. Since the microcontroller works natively with the arduino board we can use it while we develop and wait for the PCB shipments. The necessary hardware to make the ATMega run without a development board is also very easy in the case our PCB did not function properly.

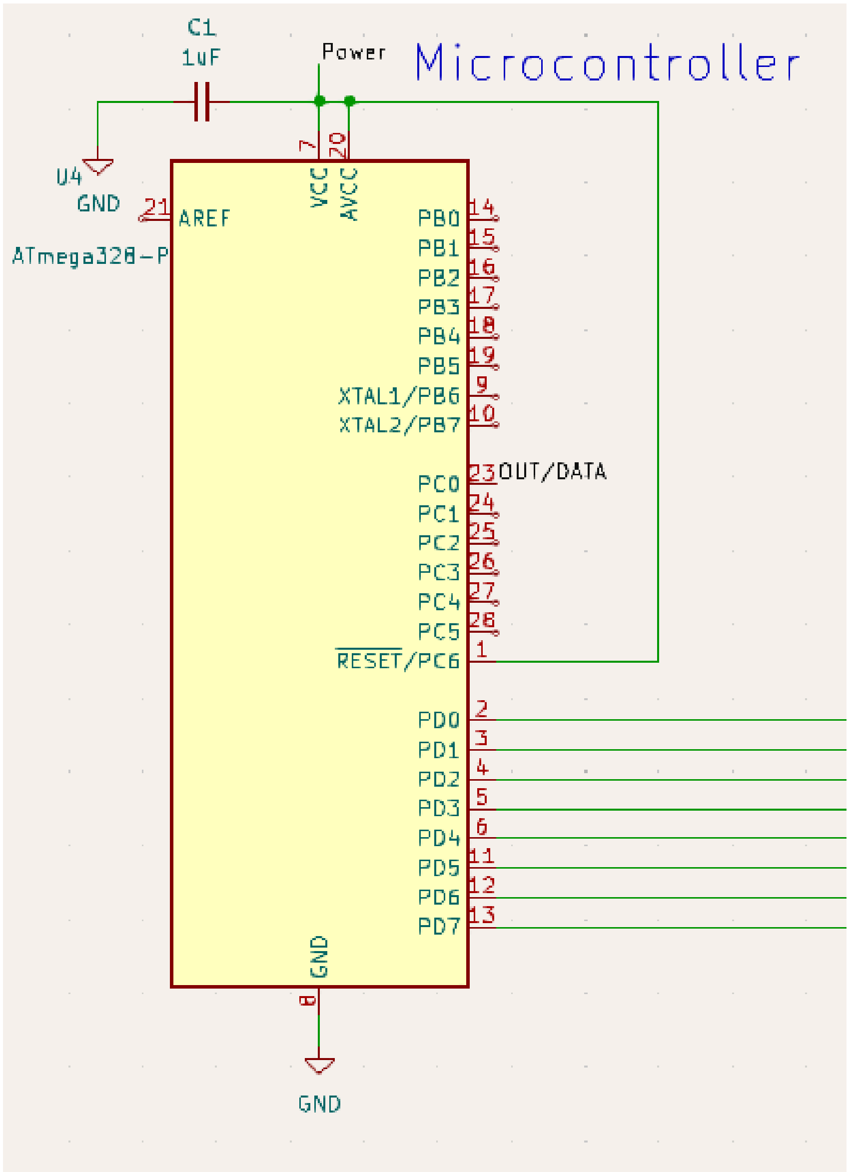

Figure 3: Microcontroller Subsystem Schematic

The motor module had us acquiring multiple motors and H-bridges. Our team was relatively naive in what motors were necessary and tried to do sufficient research to get the best fit for our project. However, we got motors that did not have enough torque and H-bridges that were underpowered. Luckily we talked to the machine shop for help with attached motors to the instrument and they were able to provide key insights onto what motors we need. Initially our team believed stepper motors would be best for the control and precision they bring in how they turn, however the torque put on the motors by the string seemed too great for the motors. We could have purchased higher-torque motors, but those came at a hefty cost. The machine shop recommended we use gear-box motors rated for a low RPM. This provided the motors great mechanical advantage against the tension of the string pulling back on it. The downside to the gear box motors was a higher current needed. This led us to acquire new H-bridges that could supply the motors with enough current to turn. To find new H-bridges that met this new need was easy. H-bridges are plentiful, well documented, and mostly do the same task. Once we founded a H-bridge with the current output needed we knew with good certainty that they would work with little troubleshooting. A change was that the new H-bridges could control two motors with one submodule rather than our initial plan of four H-bridges one for each motor. The final layout of two H-bridges and four motors is shown in Figure 4.

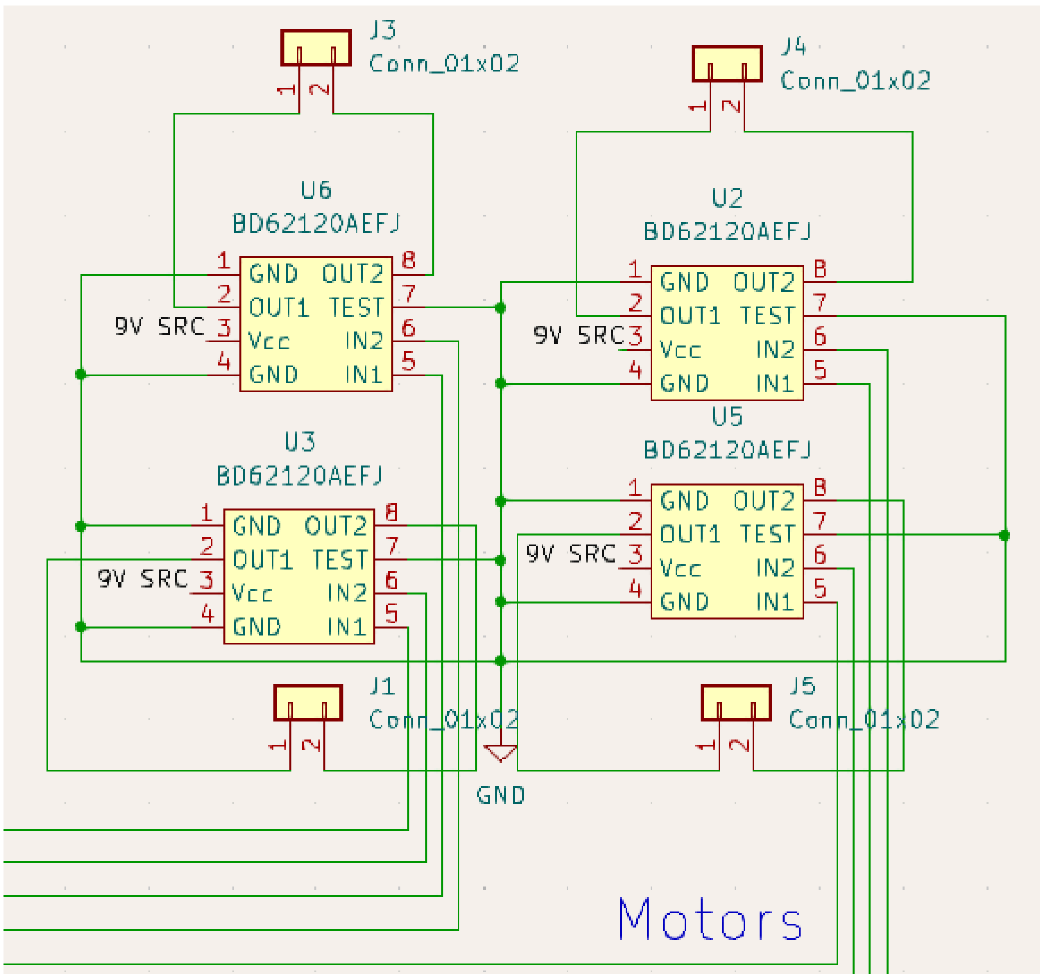

Figure 4: Motor Subsystem Schematic

Lastly the power module went through several interactions throughout the project. There was constant debate and deliberation about what type of batteries and how to implement them. We consider alkaline batteries and lithium batteries. Lithium batteries were lighter and smaller, however not as available as alkaline batteries. We were planning on using lithium batteries until very close to the deadline when we decided that since we used alkaline for testing and prototyping we would eliminate another unknown by sticking with our current type of battery. There was also thought of whether we should use one high voltage battery with different step-down circuitry to distribute the different voltages, or rather use multiple batteries each with the necessary voltages. The decision was to go with multiple batteries because of the current draw of the motors. The original design of the Power Module is shown in Figure 5. The motors quickly drained our 9V alkaline battery and adding the additional current draw of the microphone and microcontroller would only speed the process up. While convenient to only have one battery on the instrument, the inconvenience of frequency replacing the battery would outway the positives. After some brief testing we found the batteries produced the output needed with good consistency so we decided against a voltage regulator. While this would have added small consistency advantages to our project we decided against the added complexity of more components on our board.

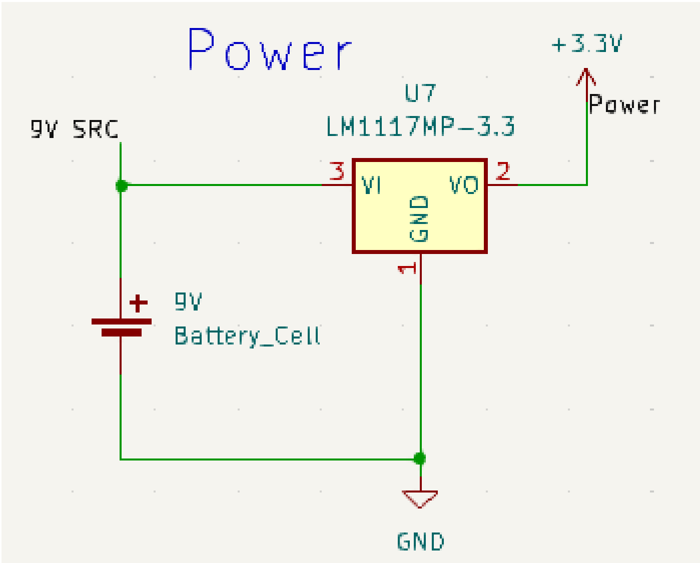

Figure 5: Power Subsystem Schematic

Our project did not require any major design equations, instead requiring the Fast Fourier Transform algorithm. Fast Fourier Transform was used to convert the incoming audio signal from the violin into a single spectral component which we are then able to use to get the frequency. Our project required the use of KiCad in creating our PCB. On KiCad, we assembled all of the required components and placed them on the schematic, which then required us to find the correct footprints for each component. The next step of creating the PCB was creating the layout, which required us to connect all of the components to each other where necessary, as well as to power and ground. We then ordered our PCB through PCBWay through the course staff. The final CAD circuit is shown in Figure 6.

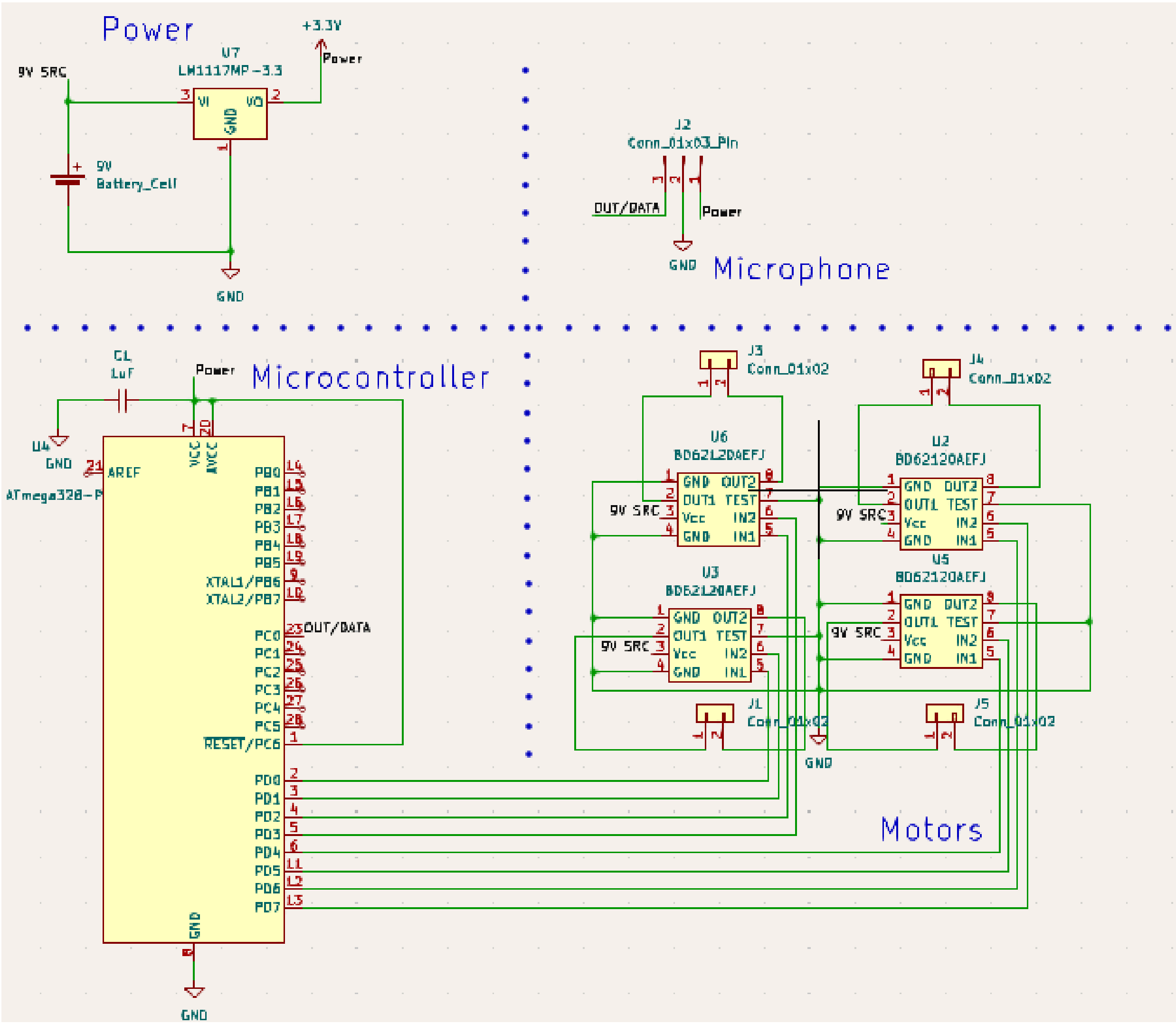

Figure 6: Entire Circuit Design/Schematic

3. Verification

Once our chosen microphone arrived we immediately tested the device to verify it could accurately receive the audio signal. We tested the microphone with aural sine waves and sawtooth waves. Both signals were received accurately enough for our purposes. The sine waves were recorded with very high accuracy, however sine waves are the simplest audio signal possible. Using a slightly more complicated wave such as a sawtooth lacked some of the higher fidelity seen in the sine waves. These results are shown in figures 6 and 7. However the project is focused on determining the fundamental frequency of the pitch which is the frequency of the first term of the sawtooth fourier series. This correlates to the overall long period frequency of the signal, which the microphone accurately encamplated[2]. This limitation actually became advantageous because it filtered out the higher frequencies initially. This saves processing time later on in the signal chain, because the microcontroller has to filter out less higher frequency noise from the audio signal.

Figure 7: Readout of Sine Wave from the Microphone

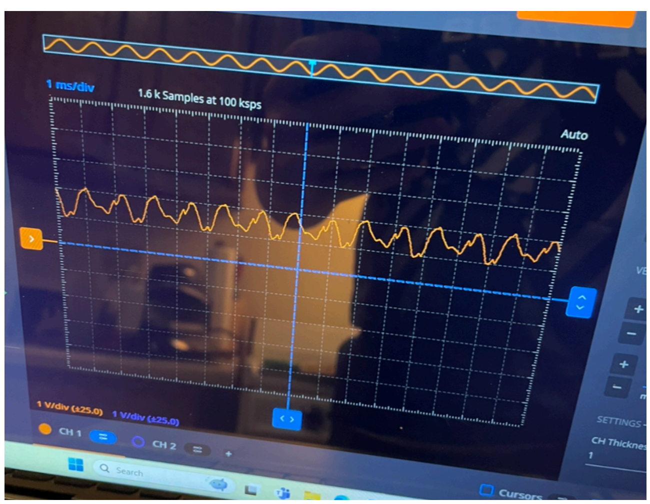

Figure 8: Readout of Sawtooth Wave from the Microphone

Due to the microcontroller module having the most inputs and outputs it has the most verification tests. The first test is the simplest, which is to verify the control of the LEDs. As long as the microcontroller can turn on and off the LEDs and they function, it passes the verification. There is nothing beyond them lighting up so a visual assessment will suffice. Secondly it is necessary to verify the degree of control the microcontroller has on the motors shown in Table 1. We desired 5º of precision in controlling the motors, however as briefly discussed in the introduction we have decided on using the equivalent 166 milliseconds(ms) of precision. These are equivalent measurements because the motors spin consistently at 5 rotations per minute (RPM) and at that rotational speed 5º and 166 ms are equal. This precision is measured by a test program that turns the motors in different directions every 166 ms. This change creates an audible click that we can track. Knowing the amount of clicks in a minute can be correlated to milliseconds with a simple equation:

$$Time\ precision\ =\ \frac{60000}{(BPM * 2)}$$

This allows one to see the upper limit of control the microcontroller has over the motors. This test is also included in the motor module because the microcontroller does not have enough current to turn the motors on its own. This means the test must be done in conjunction with the H-bridges. In order for this test to work both the microcontroller module and the motor module must pass the test.

Clicks per minute	Tme Threshold calculation
181 clicks	165.746 ms
182 clicks	164.835 ms
181 clicks	165.746 ms
184 clicks	165.746 ms
180 clicks	163.043 ms

Table 1: Displays precision between clicks (BPM) and time

The final verification for the microcontroller module is to test the decomposition of analog audio signal. This decomposition is done by a series of digital filtering and Fast Fourier Transforms. The output of the function is the fundamental frequency of the pitch. This calculated output value is checked against physical musical tuners - which are designed to determine the fundamental pitch of an instrument. The frequencies of these two are compared and if within a 2.5% threshold they are considered accurate. Due to this calculation running many times there is also a 95% accuracy limit on this value. Out of 20 decompositions the program should accurately assess the sound at least 19 of the times. In order to calculate the frequencies found in Table 2, we use the simple percent error formula as follows:

$$relative\ error\ =\ 100\ *\ \left|\frac{(Estimated\,Value - Actual\,Value)}{Actual\,Value}\right|$$

Actual Frequency	Perceived Frequency	Relative Error
440.0 Hz	445 Hz	%1.13
196.0 Hz	200 Hz	%2.04
293.7 Hz	295 Hz	%0.443
659.3 Hz	665 Hz	%0.865

Table 2: Relative error calculations

The motor module's verification test is the same as the microcontroller test to measure the degree of precision the motors can be controlled by. A test program is run to change the motors' direction quickly in order to see how fast and precise the motors can be controlled. The output from the microcontroller is fed into the H-bridges which amplify the current. If the H-bridges work as intended the microcontroller program will run correctly. If the microcontroller test passes the motor module test must also pass. Regardless, it's important to add the emphasis on the H-bridges as they have an effect on motor behavior as shown by Table 3. When the input values of the H-bridges are the same the go into a halted state, denoted as either stop or brake. The difference between these two states aside from both inputs reading as low or high respectively is that in the stop state, the H-bridges act as an open circuit, slowing the motors down to a halt. When transitioning into the brake state, the H-bridges act as if it were a low resistor circuit, halting the motors immediately or with more abruptness. Another behavior demonstrated by the motors is the forward spin which occurs when the terminal IN1 is high and terminal IN2 is low. In the case that pin IN1 is low and pin IN2 is high, the motors will spin in reverse.

Input		Output		State
IN1	IN2	OUT1	OUT2	
L	L	OPEN	OPEN	STOP
H	L	H	L	FORWARD
L	H	L	H	REVERSE
H	H	L	L	BRAKE

Table 3: H-bridge truth table

The power module was the easiest to verify its functionality. Using fine tip multimeter probes the output voltage is able to be rad. These values can then be checked to our desired values without any further calculations or observations. The only particular thing to note is that since two separate batteries are used it is necessary to check the output voltage in two separate areas of the submodule.

4. Costs and Schedule

4.1 Costs

Total Labor Cost:

($45/hour) x 2.5 x 110 hours = $12,375 labor cost per person

Three lab partners for a total labor cost = 3 x $2250 = $37,125

Final Parts List:

Description	**Manufacturer**	**Part #**	**Quantity**	**Per Item Cost**	**Total Cost**	**Purchase Link**
H-bridge	Rohm Semiconductor	BD62120AEFJ	4	$2.50	$10.00	https://www.digikey.com/en/products/detail/rohm-semiconductor/BD62120AEFJ-E2/10233249?utm_adgroup=&utm_source=google&utm_medium=cpc&utm_campaign=Pmax%20Shopppping_Supplier_Rohm_0846_Co-op&utm_term=&utm_content=&utm_id=go_cmp-20747727598_adg-_ad-__dev-c_ext-_prd-_sig-CjwKCAiA_tuuBhAUEiwAvxkgTgX5syiwehMVOQFw6puBaX-QF2m685E9IchXzkguQuwkXbqHCF41kRoCEjMQAvD_BwE&gad_source=1&gclid=CjwKCAiA_tuuBhAUEiwAvxkgTgX5syiwehMVOQFw6puBaX-QF2m685E9IchXzkguQuwkXbqHCF41kRoCEjMQAvD_BwE
Microcontroller	Microchip Technology	ATMEGA328P-PU	1	$2.89	$2.89	https://www.microchip.com/en-us/product/atmega328p
Electret Microphone Amplifier	MAXIM	MAX4466	1	$6.95	$6.95	https://www.adafruit.com/product/1063?gad_source=1&gclid=CjwKCAiA_tuuBhAUEiwAvxkgTi7vZAKYCRdVOlCNYvX4cyT80aqFMCULmO3u1XPQK5R9G0Bh7yHnrxoCU60QAvD_BwE
Linear Regulator	Texas Instruments	LM1117 - MPX	1	$1.69	$1.69	https://www.digikey.com/en/products/detail/texas-instruments/LM1117T-ADJ%2FNOPB/363595?utm_adgroup=Texas%20Instruments&utm_source=google&utm_medium=cpc&utm_campaign=PMax%20Shopping_Supplier_Texas%20Instruments&utm_term=&utm_content=Texas%20Instruments&utm_id=go_cmp-17816159938_adg-_ad-__dev-c_ext-_prd-363595_sig-CjwKCAiA_tuuBhAUEiwAvxkgTookIygA9PJ2FDsRDDCLVcRAdPSv4CglVPE-eKv3O_nfyy9iugZ1WBoCkEAQAvD_BwE&gad_source=1&gclid=CjwKCAiA_tuuBhAUEiwAvxkgTookIygA9PJ2FDsRDDCLVcRAdPSv4CglVPE-eKv3O_nfyy9iugZ1WBoCkEAQAvD_BwE
5 RPM Motors	Walfront	25GA-370	4	$12.11	$48.44	https://www.amazon.com/25GA-370-Micro-Reduction-Geared-Electronic/dp/B07JWBSF6B
Violin	Yasisid	VN-210	1	$59.99	$59.99	https://www.amazon.com/Yasisid-Stringed-Instruments-Beginner-Shoulder/dp/B0C431T49B

Total Parts Cost: $129.96

This final parts list does not include other materials that were not used in the final project such as underpowered motors or unused H-bridges because those materials were returned for a refund. We utilized the kind and amazing help from the ECE machine shop for no cost because the work was minimal but necessary. They provided us with metal brackets to attach the motors to the instrument and custom metal tuning rods for the motors to encase. A reasonable estimate if we were charged from an outside manufacturer would be $50.

We also tested with an Arduino Uno board and an ADALM kit from ECE 110. If one were to purchase these, it would be around $250, however we already had these devices from previous courses and are not accounted into our gross cost.

Total cost: $37,254.96

4.2 Schedule

	M	T	W	Th	F
Week 1 (2/19 - 2/23)				Design Doc - Team Picked out exact parts - Team	Ordered Main Parts - Team
Week 2 (2/26 - 3/1)	Checked that motors work with the strings - Kevin		Design Review 1pm - Team	PCB Outline - Ginny	Very Rough Code - Kevin
Week 3 (3/4 - 3/8)	Began refining code throughout the week - Kevin				
Week 4 (Spring Break)	Continued to refine code to a finished product - Kevin				
Week 5 (3/18 - 3/22)			Test power module - Team	Continue testing previous or begin testing new PCB - Ginny and Erik	Have a finished code - Kevin
Week 6 (3/25 - 3/29)			Test microphone module - Team		Testing motor - Team
Week 7 (4/1 - 4/5)	Made revisions to PCB design - Ginny		Verified requirements are met for modules - Team		
Week 8 (4/8 - 4/12)	Made revisions to PCB design - Ginny	PCB Ordered - Ginny			
Week 9 (4/15 - 4/19)		Mock Demo - Team		Began preparing for mock presentation - Team	

Week 10 (4/22 - 4/26)	Final work on project - Team	Final Demo - Team		Mock presentation - Team	
Week 11 (4/29-5/3)	Final Presentation -Team		Final Paper due - Team	Lab Checkout - Team	

5. Conclusions

At the end of our project, the violin could tune itself correctly. There were errors in the PCB design so the project relies solely on handwired breadboards, but given enough time and attention this could be accomplished with a mass-produced PCB. Although the project did work there could certainly be improvements in the future. The biggest improvement could be made by using a more powerful microcontroller. The ATMega 328 accomplished what it needed to for the project, but slowly and loosely. A faster microcontroller with more memory and a much higher degree of accuracy could have been accomplished with fewer errors. Outside of that, the other modules worked as intended with consistency.

If we were to work further on this project, we would attempt to make the divide a lot smaller and more compact, with more emphasis on being able to actually play the violin without any issues. Our current iteration of the project leaves a lot to be desired in terms of playability as wires are going all around the strings, with no room to hold the violin normally and use the bow on it. We would like to hide all of this exposed circuitry under some kind of boxes or along the back of the violin. We would also like to give the device a single on/off switch, as it currently has multiple, and we would like to have a microcontroller that is both faster and has more memory for there to be less error as mentioned above. The last thing we would like to improve is a better user interface, as currently it is hard to understand what is happening unless you learn what each button on the device does.

As for ethical and safety concerns for the project we have very few. The device does little more than fine adjustments to a fragile part of an instrument. We will adhere to the IEEE code of ethics. According to the IEEE code of ethics, we will avoid any injuries with our project. The safety concern of the high tension strings should be noted. Violin strings are under high tension when correctly tuned and inevitably strings pop under the tension. This is not necessarily an error; occasionally a string can pop with little cause or mishandling. Although, we can mitigate the chances of popping by designing the product to tune slowly and prevent significant overshooting of the target pitch. Both of these precautions will put less sudden stress on the strings, and help to limit the possibility. In the case a string does break, it is no more dangerous than playing a normal instrument. No extra care or worry needs to be placed than when normally playing a string instrument. Both are safe within reasonable limits, but it's on us to do what we can for the safety of the instrument and players. Another safety concern to consider is uncovered pieces of electrical components as it could cause shocks to users. To prevent this, all electrical components will be covered appropriately to avoid exposed electrical conductors.

If this project were to become widely adopted there would be few negative economic, environmental, or societal factors. The project saves time for the precious lesson time of a beginner student and their teacher. This device would not take away any jobs or negatively affect a certain party. It is solely saving time. The device does not use many dangerous materials outside of batteries which are commonplace and do not pose an excess danger in this situation as in any other common context. [4]

6. References

[1] “Violin,” *Gsu.edu*, 2020. http://hyperphysics.phy-astr.gsu.edu/hbase/Music/violin.html

[2] R. E. Berg, “Sound - Overtones,” *Encyclopædia Britannica*. Mar. 06, 2019. Available: https://www.britannica.com/science/sound-physics/Overtones#ref527272

[3] E. W. Weisstein, “Discrete Fourier Transform,” *mathworld.wolfram.com*. https://mathworld.wolfram.com/DiscreteFourierTransform.html

[4] IEEE, “IEEE Code of Ethics,” *ieee.org*, Jun. 2020. https://www.ieee.org/about/corporate/governance/p7-8.html

Appendix A - Requirements and Verification Tables

Power Module

Requirement	Verification
Provide consistent and accurate voltage to the other components, primarily a 9V ± 0.5V line to turn the motors and a 3.3V ± 0.3V line for most other items.	1. Connect all subsystems to power, either directly to the 9V line or to the 3.3V line from the batteries or voltage regulator respectively 2. Use a multimeter to verify that the 9V line is within the tolerance range, as well as verify the 3.3V line is within its own tolerance range
Able to keep the motors running under 5 rpm to avoid damage to the violin	1. Connect power to the motors 2. Check if motor speeds are under 5 rpm by marking one side of the motor shaft and calculating the speed of a rotation to make sure the violin will not be damaged

Microphone Module

Requirement	Verification
Accurately transceive sounds ranging from a minimum span of 100 Hz to 1000 Hz.	1. Set up the microphone to receive sound and send the signal to an oscilloscope. 2. Play sounds with verified frequencies 3. Check that the correct frequency is displayed on the oscilloscope from the microphone input

Microcontroller Module

Requirement	Verification
Be able to control the 2 LEDs to show the current string being worked on and do this without skipping to another string or turning off completely	1. Once the power switch is turned on, the first LED should turn on for the first string to begin being tuned 2. Once the string is played, the string should be tuned by the rest of the system 3. Once this is completed the first LED will turn off, and the second will turn on to continue tuning the next string.

Accurately determine the loudest pitch in the given analog signal using a FFT with more than 95% accuracy	1. Set up the microphone to receive sound and send the signal to the microcontroller 2. Play sounds with verified frequencies 3. Run the signal through the FFT 4. Check that the frequency is within the given range of frequencies
Send consistent signals to the motor module so the motor spins within 5 degrees of the correct location	1. Connect the microcontroller to the motors and to the power system 2. Send a command from the microcontroller to the motors to spin a certain amount of degrees 3. Check that the motor spins within 5 degrees of the correct amount

Motors Module

Requirement	Verification
Accurately turn the tuning peg to within 5 degrees as directed forwards and backwards through the use of H-bridges.	1. Connect the motors to the H-bridges 2. Mark the tuning peg to see where you began 3. Input a command to turn the peg a specific number of degrees. 4. Check the difference between the mark and where it began and compare this to the inputted command and make sure it is within 5 degrees.