Automatic Piano Tuner - Design Document

By

Joseph Babbo

Colin Wallace

Riley Woodson

Design Document for ECE 445, Senior Design, Spring 2022

TA: Zhicong Fan

24 February 2022

Project No. 49

Contents

1 Introduction . . . 1

1.1 Problem . . . 1

1.2 Solution . . . 1

1.3 Visual Aid . . . 1

1.4 High-Level Requirements List . . . 2

2 Design . . . 3

2.1 Block Diagram . . . 3

2.2 Physical Design . . . 4

2.3 Subsystem Overview . . . 5

2.3.1 Audio Acquisition Subsystem . . . 5

2.3.2 Control System Subsystem . . . 6

2.3.3 Battery Subsystem . . . 7

2.3.4 Power Regulation Subsystem . . . 8

2.3.5 Electric Motor Drive Subsystem . . . 9

2.3.6 Display Subsystem . . . 10

2.3.7 Buttons Subsystem . . . 11

2.4 Tolerance Analysis . . . 13

2.5 Circuit Schematic . . . 14

3 Cost and Schedule . . . 15

3.1 Cost Analysis . . . 15

3.1.1 Labor . . . 15

3.1.2 Parts . . . 15

3.1.3 Sum . . . 16

3.2 Schedule . . . 17

4 Ethics and Safety . . . 19

Reference . . . 20

Appendix A Relevant Code . . . 21

1 Introduction

1.1 Problem

Piano tuning is a time-consuming and expensive process. An average piano tuning will cost in the $100 - $200 range [1] and a piano will have to be re-tuned multiple times to maintain the correct pitch [2]. Due to the strength required to alter the piano pegs it is also something that is difficult for the less physically able to accomplish.

1.2 Solution

We hope to bring piano tuning to the masses by creating an easy to use product which will be able to automatically tune a piano by giving the key as input alongside playing the key to get the pitch differential and automatically turning the piano pegs until they reach the correct note.

1.3 Visual Aid

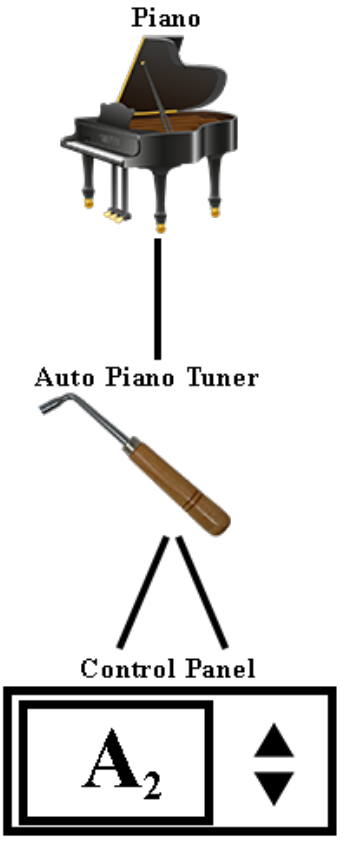

Figure 1: High level visual aid for the 'Automatic Piano Tuner'.

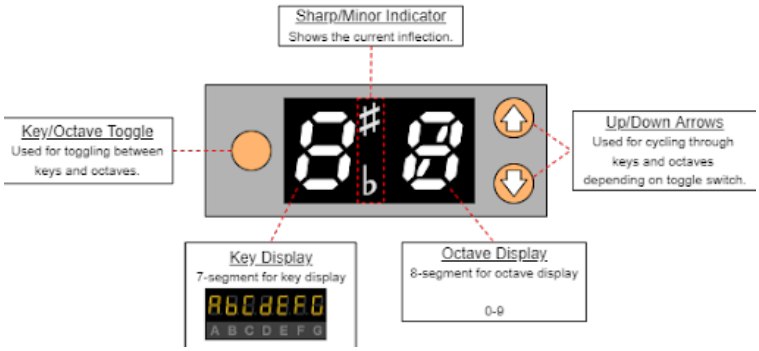

Figure 2: Close up visual aid for the 'Automatic Piano Tuner's' UI display.

Figure 3: A mock-up using a electric driver [3] that would give the correct amount of torque ($\sim 10Nm$).

1.4 High-Level Requirements List

- The 'Automatic Piano Tuner' should be able to separate and tune to frequencies less than 1.6 Hz apart[4].
- The 'Automatic Piano Tuner' should be able to produce 9-14 Nm of torque[5].
- The 'Automatic Piano Tuner' should be able to control and tune all 88 notes of the piano[6].

2 Design

2.1 Block Diagram

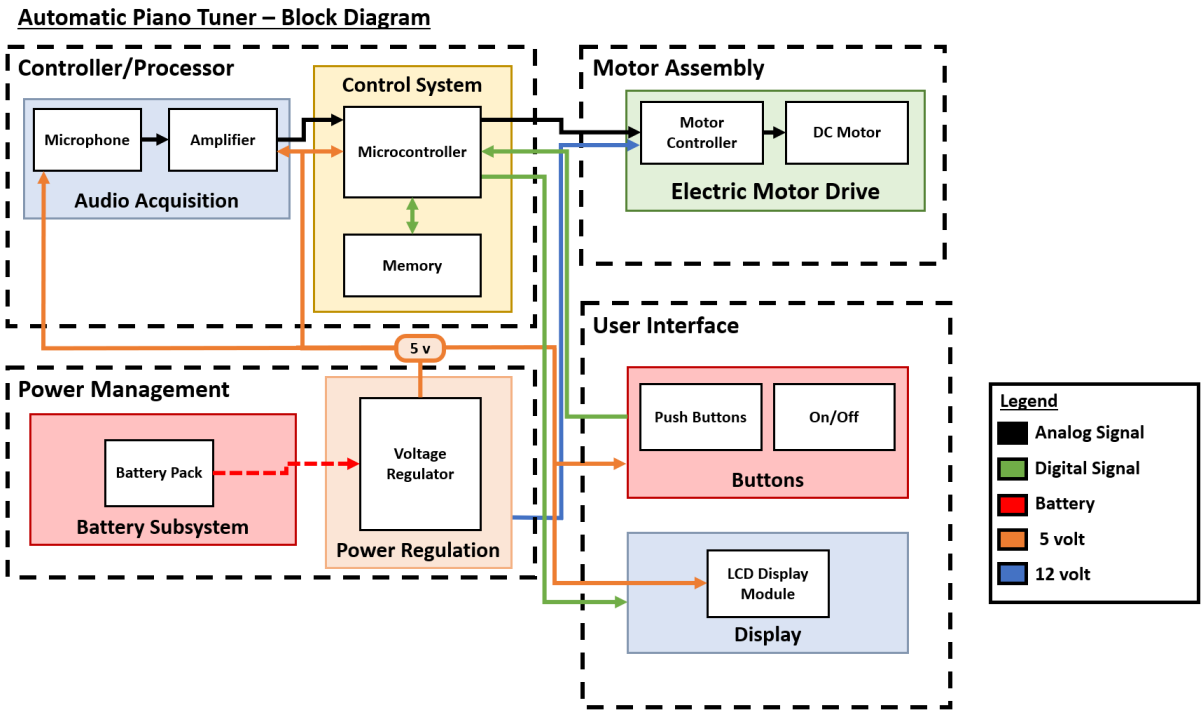

Figure 4: Block Diagram for the 'Automatic Piano Tuner'.

Our design consists of four main areas of focus, three of which we have split further into two subsystems for a total of seven blocks. The power management system, consisting of the battery and power regulation subsystems, help supply the rest of the design with the correct amount of power to run. The user interface system, consisting of the button and display subsystems allows the user to easily control the device and receive critical information on their control. The electric motor subsystem is critical in allowing the piano pegs to be actually moved, allowing tuning to occur. In the controller/processor system, the audio acquisition subsystem allows for the current key's waveform to reach the microcontroller, so that the device knows how much and which way a note has to be tuned. Finally, the control system is able to get the frequency of the played note from the waveform and knowing what frequency the note should be at, tell the electric motor drive to move accordingly. Further detail on each of these subsystems is given in the *Subsystem Overview*.

2.2 Physical Design

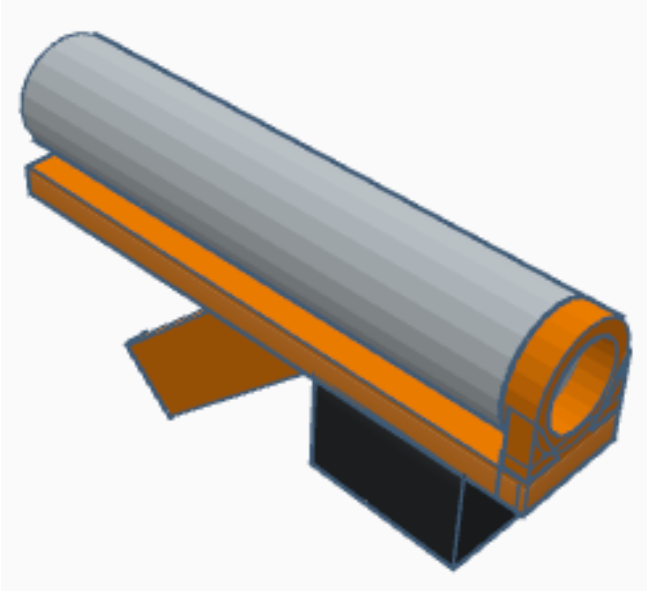

Figure 5: 'CAD mock-up of the 'Automatic Piano Tuner'.

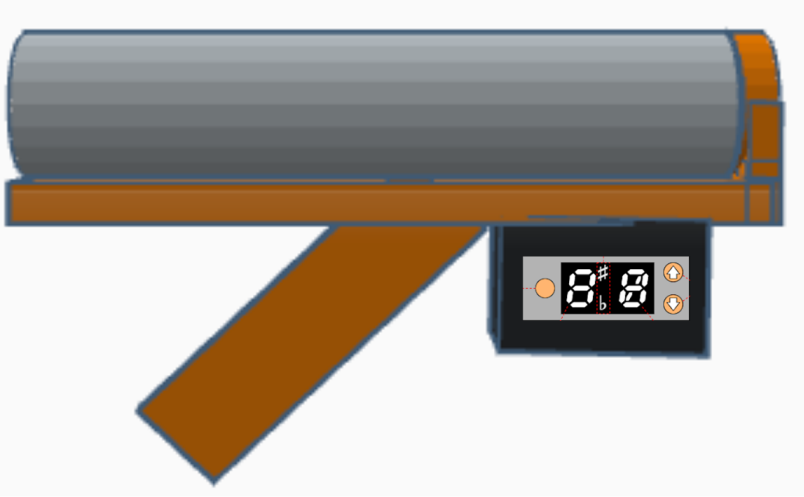

Figure 6: 'CAD mock-up of the side view of the 'Automatic Piano Tuner'.

Our design will try to replicate a cordless hand drill, as consumers are already well versed in the ergonomics and use of such a product. The design has a handle extending from the underside of the motor housing. The motor housing will consist of a metal collar that the motor will be fastened to, and a metal plate that will extend backwards the length of the motor. A black-box, filled with our electronics, will be fastened below this metal plate.

2.3 Subsystem Overview

2.3.1 Audio Acquisition Subsystem

The 'Automatic Piano Tuner' will use a microphone and amplifier to detect and send through the frequency of the piano's non-tuned note. This data will be sent through to the microcontroller which will compare the note given to it by the microphone with what the note should be that is saved in memory to tune the piano correctly. As the note changes from the tightening or loosening of the piano string, the microphone will continually pick up the frequency so that the microcontroller knows when to signal the motor when to stop turning.

Table 1: Audio Acquisition Subsystem Requirements and Verification

Requirements	Verification
1. Pick up frequencies accurately down to 27.5Hz.	1 With an external device play a perfect 27.5Hz tone. 2 Using the audio acquisition subsystem alongside the software frequency analyzer on the microcontroller, determine the frequency (as given by the microphone). 3 Make sure the tone is within ±10% of 27.5Hz to fulfill the accuracy requirement.
2. Pick up frequencies accurately up to 4186Hz.	1 With an external device play a perfect 4186Hz tone. 2 Using the audio acquisition subsystem alongside the software frequency analyzer on the microcontroller, determine the frequency (as given by the microphone). 3 Make sure the tone is within ±10% of 4186Hz to fulfill the accuracy requirement.
Continued on next page	

Table 1 – continued from previous page

Requirements	Verification
3. Differentiate between frequencies 1.6Hz apart.	1 With an external device play a tone starting at 27.5Hz. 2 Sweep up slowly until the software frequency analyzer on the microcontroller notices a frequency change. 3 Make sure the software frequency analyzer on the microcontroller is able to detect a separate frequency at 29.1 Hz or earlier.

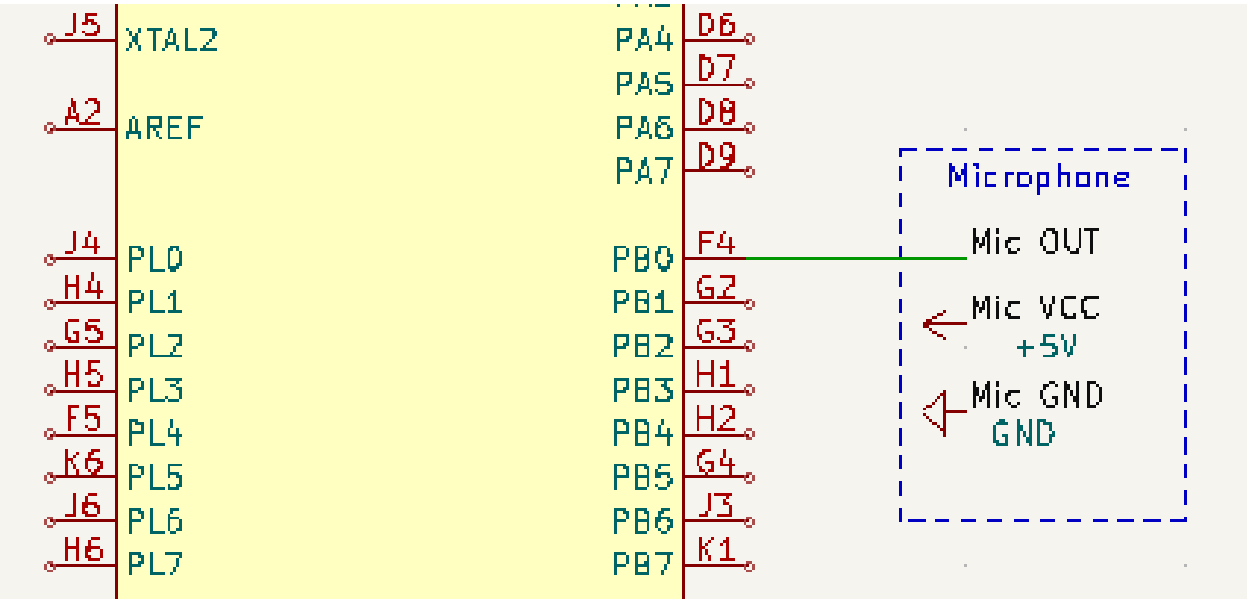

Figure 7: 'Audio Acquisition Subsystem' (connecting to 'Control System') circuit schematic.

2.3.2 Control System Subsystem

The microcontroller acts as the 'command center' of the 'Automatic Piano Tuner'. It takes in frequencies from the audio acquisition subsystem and compares the frequency to what the frequency should be which is saved on memory and indexed by the note pointed to from the button subsystem. If the actual frequency is higher than the indexed frequency, it tells the motor assembly system to move left and loosen the string, while if the actual frequency is lower than the indexed frequency, it tells the motor assembly to move right and tighten the string until the frequency it receives from the audio acquisition system is within a acceptable margin of error with the indexed frequency.

See Appendix A for more detail on the frequency analyzer code used to determine the input frequency based on the sound from the 'Audio Acquisition Subsystem'.

Table 2: Control System Subsystem Requirements and Verification

Requirements	Verification
1. Accurately determine heard frequency in less than 5 seconds.	1 With an external device play a tone (use a tone between 27.5Hz and 4186Hz to insure the 'Audio Acquisition Subsystem' can pick up the frequency). 2 Measure the time it takes for the microcontroller to analyze and determine the tone's frequency. 3 Make sure the time from tone played to frequency determined is less than 5 seconds.
2. Store all 88 standard piano frequencies.	1 Attempt to store table of the 88 piano keys alongside their known frequencies in the microcontroller's memory. 2 Make sure the entire table fits inside memory for later retrieval.

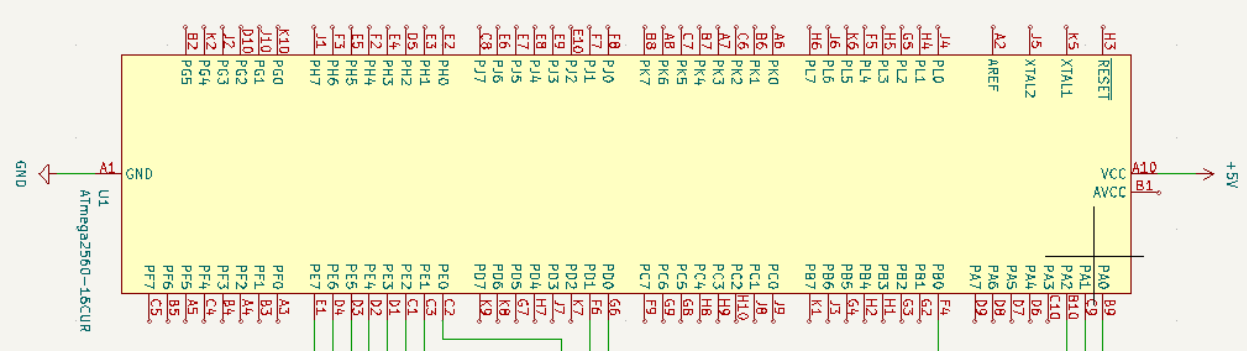

Figure 8: 'Control System Subsystem' circuit schematic.

2.3.3 Battery Subsystem

The 'Automatic Piano Tuner' should be able to be used for an entire piano tuning session without having to be re-charged or plugged in. The battery system is connected to the power regulation system to deliver enough power for the rest of the 'Automatic Piano Tuner' to function. The battery should last for at least 2 hours of use given that a typical piano tuning lasts between 1 and 1.5 hours [2].

Table 3: Battery Subsystem Requirements and Verification

Requirements	Verification
1. Remain powered for 2 hours or greater at load.	1 Turn on device and use at high load and measure time until the battery runs out. 2 Make sure the time until shutdown is greater than 2 hours.

2.3.4 Power Regulation Subsystem

The power regulation system is responsible for delivering stable voltage levels to the various components in the system. It will transform the battery voltage into usable voltage for the microcontroller, user interface, and microphone sensor. It must provide sufficient current to the motor such that it does not saturate before reaching the desired torque output.

Table 4: Power Regulation Subsystem Requirements and Verification

Requirements	Verification
1. Provide 5V to the 'Audio Acquisition', 'Control System', 'Button', and 'Display' subsystems.	1 Plug in battery to power the 'Power Regulation' subsystem. 2 Measure output of the 5V lead. 3 Make sure voltage levels are within ±5% of 5V.
2. Provide 12V to the 'Motor Assembly' subsystem.	1 Plug in battery to power the 'Power Regulation' subsystem. 2 Measure output of the 12V lead. 3 Make sure voltage levels are within ±5% of 12V.

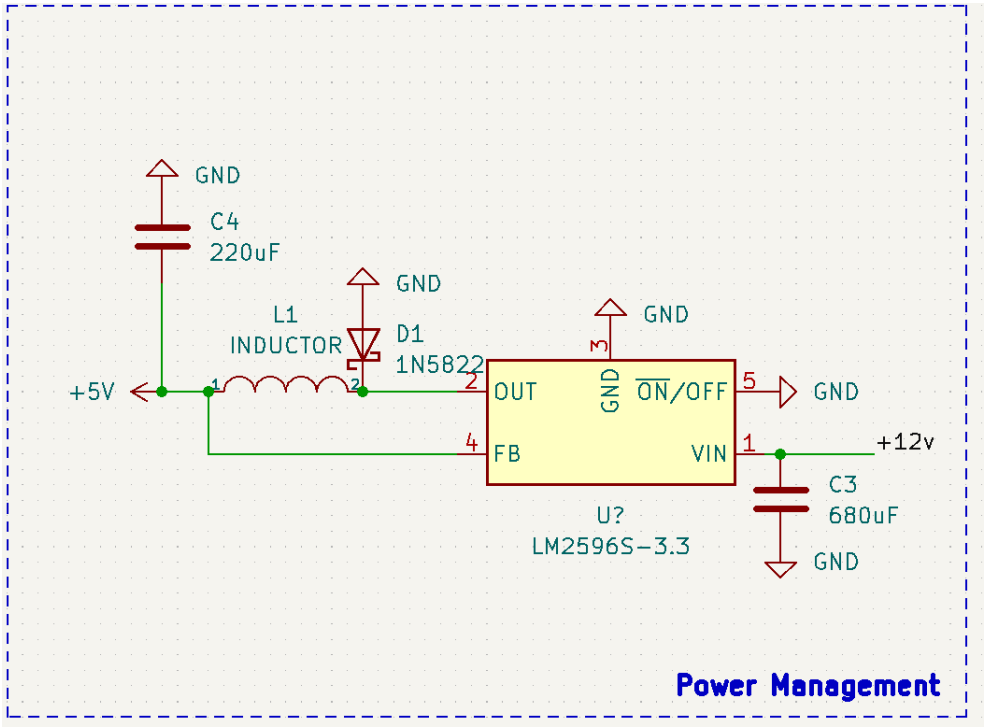

Figure 9: 'Power Regulator Subsystem' circuit schematic.

2.3.5 Electric Motor Drive Subsystem

The drive system will utilize a small torque motor and a gear shaft to produce the required amount of torque. As torque motors can get large, it will be important to create a capable gear shaft that will be able to amplify the torque of a smaller motor.

Table 5: Electric Motor Drive Subsystem Requirements and Verification

Requirements	Verification
1. Able to rotate piano tuning pin.	1 Put piano tuner bit on tuning pin. 2 Power motor manually using leads connected to controller. 3 Make sure that the tuning pin rotates.
Continued on next page	

Table 5 – continued from previous page

Requirements	Verification
2. Able to produce the required 9-14 Nm .	1 Attach one meter lever arm to motor shaft. 2 Place other end of arm onto weight scale. 3 Power motor, with arm rotating into weight scale, and record value. 4 Use equation $1m \cdot 9.81m/s^2 \cdot Akg$ (where A is measurement) to calculate torque in Nm. 5 Make sure torque is within 9-14 Nm range.

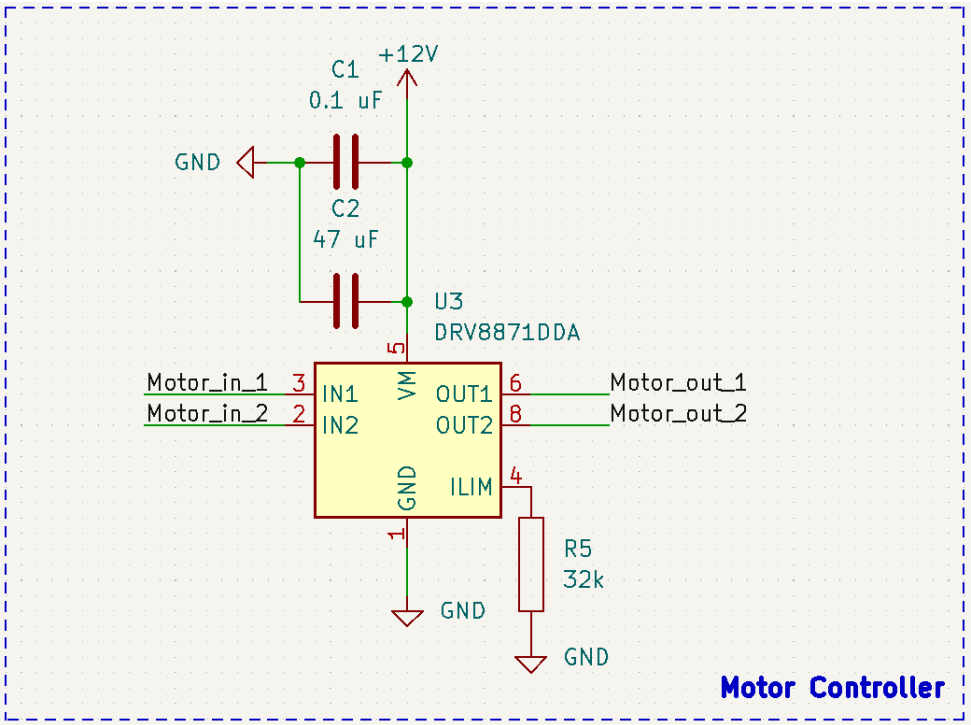

Figure 10: 'Electric Motor Drive Subsystem' circuit schematic.

2.3.6 Display Subsystem

The 'Automatic Piano Tuner' will use an LCD display to show the current note, including if the note is sharp, alongside the current octave number. There will be a 7-segment display to show the note letter: A, b, C, d, E, F, G, an 8-segment display to show the octave number: 0, 1, 2, 3, 4, 5, 6, 7, 8, and an extra display to show if the note is sharp or not. The display will get the current note and octave from the microcontroller which will get the current note from the button subsystem.

Table 6: Display Subsystem Requirements and Verification

Requirements	Verification
1. Display current note and octave at all times.	1 While 'Automatic Piano Tuner' is in use, record the display. 2 Make sure the current note and octave are always displayed on screen for the entire time.
2. Update display within 1 second of a button push.	1 Upon pressing a button track time until the display updates. 2 Make sure the display update time is less than 1 second.

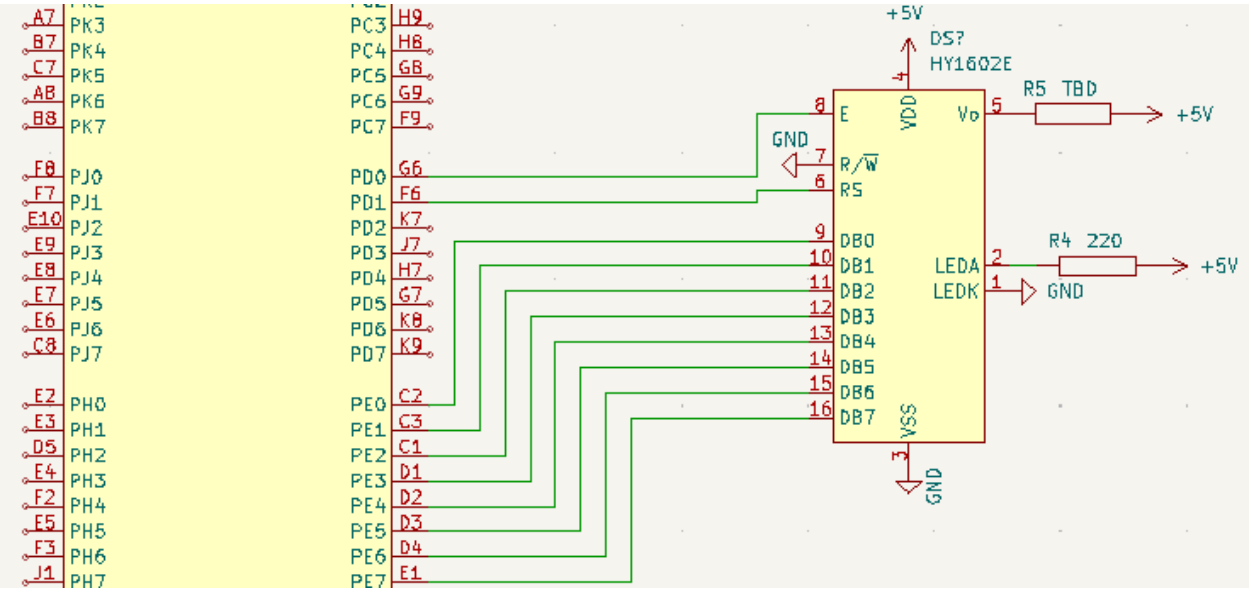

Figure 11: 'Display Subsystem' (connecting to 'Control System') circuit schematic.

2.3.7 Buttons Subsystem

There will be three buttons on the 'Automatic Piano Tuner' associated with moving up and down keys as well as changing octaves. These buttons will send a digital signal to the microcontroller which will use the button press signal to alter the frequency the tuner is tuning to in the software as well as send a signal to the LCD display to correctly show the new key/octave. The up button will move the key up one note in order: C, C♯, D, D♯, E, F, F♯, G, G♯, A, A♯, B. The down button will move the key's in the reverse direction: B, A♯, A, G♯, G, F♯, F, E, D♯, D, C♯, C. If the key is C and the octave is 8 (the last key on a standard

piano), the up button will not change anything. Similarly if the key is A and the octave is 0 (the first key on a standard piano), the down button will not change anything. If the current key is B and the up button is pressed, the key will wrap around to C with the octave increasing by 1. Likewise if the current key is C and the down button is pressed, the key will wrap around to B with the octave decreasing by 1. The octave button will automatically move the octave up by 1 to quickly change keys. For keys: C♯, D, D♯, E, F, F♯, G, G♯, if the octave is 7, the octave will wrap around to 1. For keys: A, A♯, B, if the octave is 7, the octave will wrap around to 0. Finally, if the key is C and the octave is 8, the octave will wrap around to 1.

Table 7: Buttons Subsystem Requirements and Verification

Requirements	**Verification**
1. Pressing the down button should update the frequency listening against in less than 1 second.	1 Press the down button and measure time. 2 Check when the software on the microcontroller changes the frequency it is checking against. 3 Make sure the time between button press and software change is less than 1 second.
2. Pressing the up button should update the frequency listening against in less than 1 second.	1 Press the up button and measure time. 2 Check when the software on the microcontroller changes the frequency it is checking against. 3 Make sure the time between button press and software change is less than 1 second.
3. Pressing both buttons at the same time (within 10ms of each other) should do nothing.	1 Press both the up and down button simultaneously (within 10 ms of each other). 2 Check the software on the microcontroller for any update in the frequency it is checking against. 3 Make sure there is no change in the frequency the software on the microcontroller is checking against.

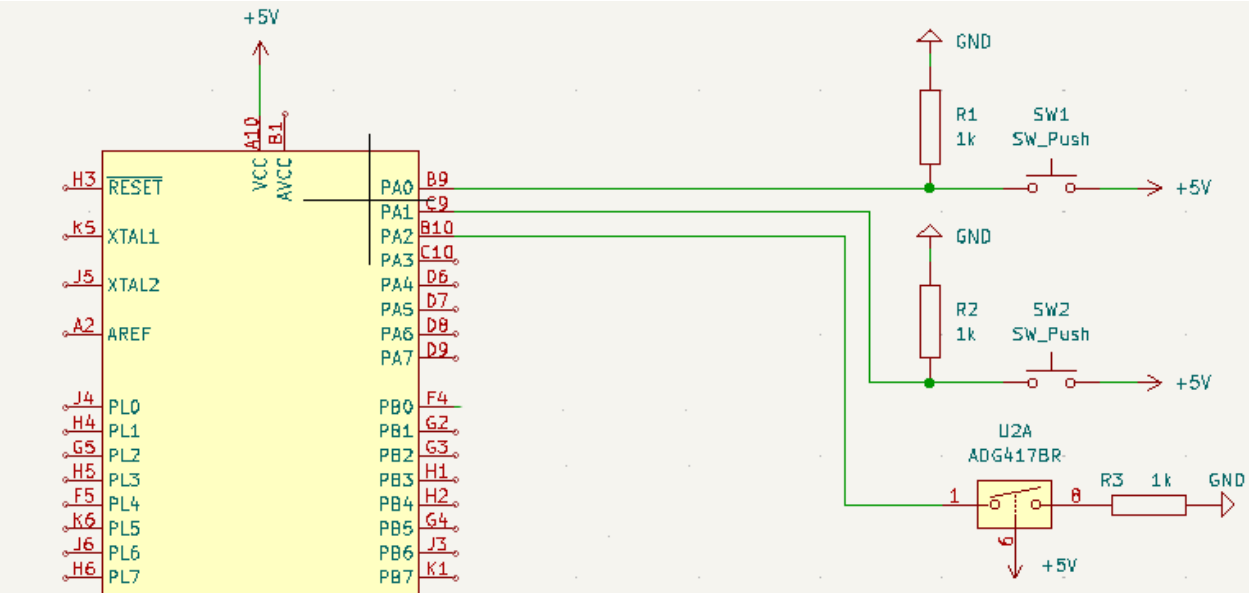

Figure 12: 'Buttons Subsystem' (connecting to 'Control System') circuit schematic.

2.4 Tolerance Analysis

One concern for this project is producing enough torque to be able to tune the pins of the piano. We have already established that this tool will need to produce 9-14 Nm of torque [5]. We have identified a prime candidate for the motor we would use in this tool [7]. This motor produces less torque than required and thus to generate the required torque we will be using a motor gear shaft drive that will amplify the torque of the motor. Gear shafts amplify torque based on the 'gear ratio' which is the ratio of the gear diameters. We can calculate using the following equation and parameters:

Motor Torque: 2.962 Nm, Required Torque: 12 Nm

$$GearRatio = \frac{d2}{d1} = \frac{OutputTorque}{InputTorque} = \frac{12Nm}{2.962Nm} \approx 4 \tag{1}$$

The ratio of the gear diameters must be 4, which we believe will be doable on this gear train. The diameter of the motor shaft is 0.091" [7] which we can stick a 0.25" driver gear on. The driven gear would then have to be 1" in diameter to get the required torque. With the overall diameter of the motor being 1.2", we believe we can use a housing that would be small and ergonomic enough to fit inside the piano. With these considerations in mind we believe we can produce a tool that would have the required amount of torque.

If the torque does not reach 9 Nm, there will not be enough force to adequately turn the tuning pin on a piano, causing the 'Automatic Piano Tuner' to not function correctly. Additionally, if the torque is greater than 14 Nm, the 'Automatic Piano Tuner' will spin the piano's tuning pins too quickly, potentially damaging the piano and making tuning to exact frequencies challenging.

2.5 Circuit Schematic

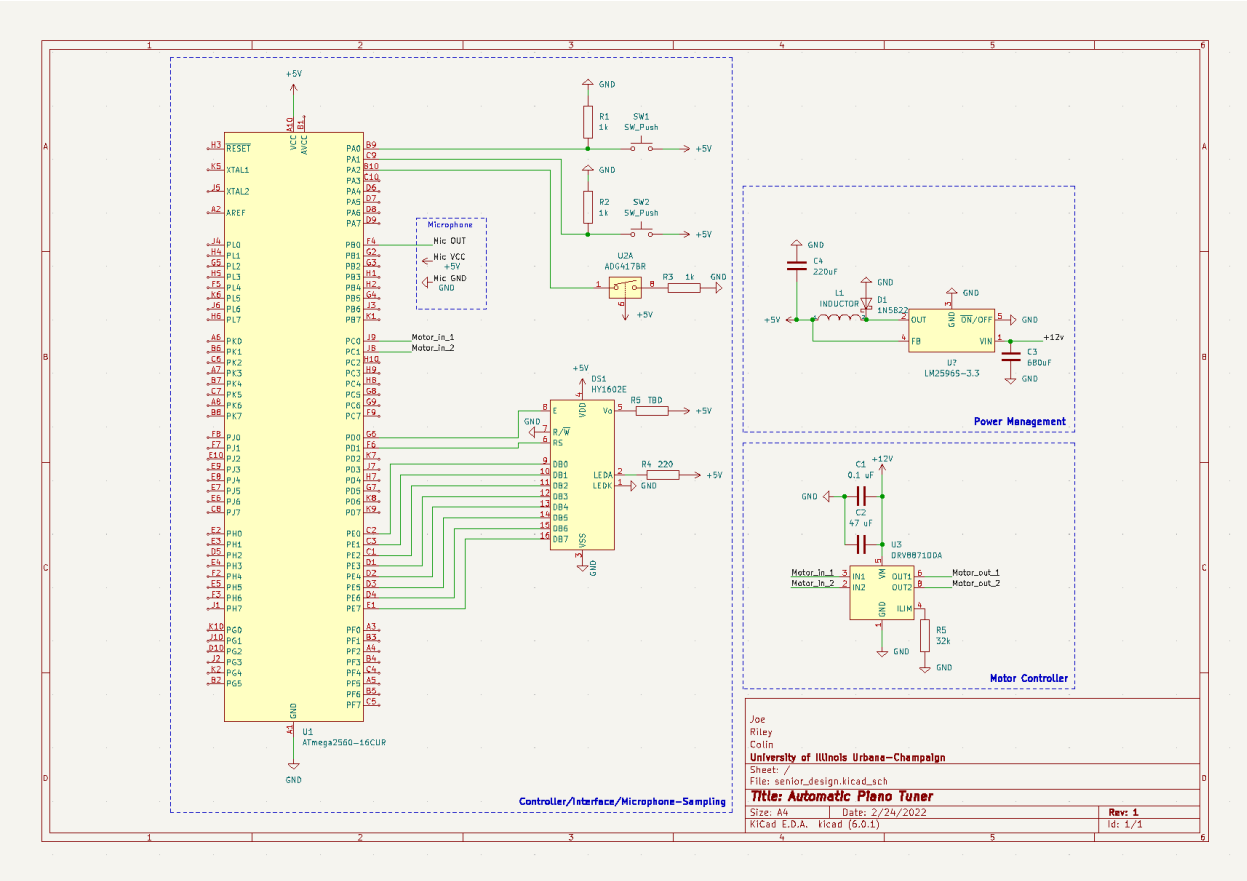

Figure 13: 'Automatic Piano Tuner' circuit schematic.

3 Cost and Schedule

3.1 Cost Analysis

3.1.1 Labor

Taking the average salary of UIUC electrical and computer engineering graduates [8][1] alongside the relative population of the two majors [9][2], the average ECE graduate from UIUC makes an average starting salary of $89,796.

$$Avg.Salary = EEAvgSalary \cdot \frac{\#ofEEMajors}{TotalECEMajors} + CompEAvgSalary \cdot \frac{\#ofCompEMajors}{TotalECEMajors} \tag{2}$$

$$= \$79,714[8] \cdot \frac{870[9]}{2089[9]} + \$96,992[8] \cdot \frac{1219[9]}{2089[9]} \tag{3}$$

$$\approx \$89,796. \tag{4}$$

Assuming an average work week of 40 hours alongside 52 weeks in a year, gives an average hourly rate of $43.17 an hour. Assuming work on the project is done for 15 hours a week over 15 weeks in the semester, the per person labor rate is $\$43.17 \cdot 15 \cdot 15 = \$9713.15$. Multiplying this by a factor of 2.5x overhead multiplier we get $\$9713.15 \cdot 2.5 = \$24,282.88$. With three people working on the 'Automatic Piano Tuner' the labor cost increases to $3 \cdot \$24282.88 = \$72,848.64$.

3.1.2 Parts

Table 8: Parts Costs

Description	Manufacturer	Part #	Quantity	Cost ($)
Microcontroller	Microchip Technology	ATMEGA2560-16CU	1	14.27
Display	Adafruit	1447	1	10.95
Microphone	Adafruit Industries LLC	MAX4466	1	6.95
Button	Omron Electronics Inc-EMC Div	B3F-1020	2	$0.38 \cdot 2 = 0.76$
Power Switch	E-Switch	RA1113112R	1	0.67
Battery	Batteryspace	RA-H2/3A10R2WR	1	32.95
DC-DC Power Converter	Texas Instruments	LM2596	1	4.65
Motor	MidwestMotion	MMP S14-247D-12V GP32-1140	1	78.00
Piano Bit	CRAFTSMAN	CMMT43493	1	2.98

Continued on next page

[1] Using 2018-2019 numbers.
[2] Using undergraduate fall 2021 numbers.

Table 8 – continued from previous page

Description	Manufacturer	Part #	Quantity	Cost ($)
Quoted Machine Shop Labor Hours			20	$36.65 \cdot 20 = 733$
Total				885.18

3.1.3 Sum

Combining the total part costs with the total labor costs we get a final cost analysis of $\$885.18 + \$29,139.75 = \$30,024.93$.

3.2 Schedule

[3]

Table 9: Schedule

Week	Joseph Babbo	Colin Wallace	Riley Woodson
1/17	Jointly brainstorm project ideas	Jointly brainstorm project ideas	Jointly brainstorm project ideas
1/24	Jointly determine project, laboratory safety training, CAD assignment	Jointly determine project, laboratory safety training, CAD assignment	Jointly determine project, laboratory safety training, CAD assignment
1/31	Jointly write RFA, answer RFA questions, get project approved	Jointly write RFA, answer RFA questions, get project approved	Jointly write RFA, answer RFA questions, get project approved
2/7	Project proposal ethics and safety, project proposal subsystem overview, project proposal LaTeX formatting	Project proposal block diagram, project proposal subsystem overview	Project proposal visual aid, project proposal tolerance analysis, initial conversation with machine shop
2/14	Jointly research parts	Jointly research parts	Jointly research parts
2/21	Block diagram paragraph, requirements and verification, cost analysis, schedule, expand ethics and safety, LaTeX formatting	Revisit block diagram, circuit schematic, requirements and verification	Revisit machine shop, revisit tolerance analysis, physical design
2/28	Ordering/sourcing components	Design layout of PCB and estimate size of the box	Finalize machine shop design
3/7	Order the first PCB	Design box to hold the electrical components	Fit motor to the machine design

Continued on next page

[3]Schedule is not strict, with members contributing to other member's tasks.

Table 9 – continued from previous page

Week	Joseph Babbo	Colin Wallace	Riley Woodson
3/21	Test electrical components, specifically power regulation for performance	Solder components to the first PCB design	Test the motor for correct torque vs current settings
3/28	Test the individual components on the PCB to verify functionality	Program and verify that the microcontroller is capable of reading tones and writing to the display	Test functionality of the entire PCB and system
4/4	Create revision of the PCB based on initial testing	Further develop microprocessor code for frequency detection and motor control	Test entire first system on piano pins
4/11	Test audio + control subsystems requirements	Test battery + power subsystems requirements	Test motor + display + buttons subsystem requirements
4/18	Jointly prepare mock demo	Jointly prepare mock demo	Jointly prepare mock demo
4/25	Jointly prepare demo and mock presentation	Jointly prepare demo and mock presentation	Jointly prepare demo and mock presentation
5/24	Jointly prepare presentation and touch-up on final paper	Jointly prepare presentation and touch-up on final paper	Jointly prepare presentation and touch-up on final paper

4 Ethics and Safety

In our project we aim to firmly adhere to both the IEEE and ACM Code of Ethics. While developing 'Automatic Piano Tuner' we will be careful to cite and credit all work that we use or that influences us. Additionally we will pay close attention to and graciously use advice given to use by course TAs and professors as per IEEE Code of Ethics I.3. [10]

Throughout the entire development of the 'Automatic Piano Tuner', we also seek to respect all people we work with. Due to requiring outside people to successfully accomplish our project, we will make sure throughout the process to treat everyone fairly as per IEEE Code of Ethics II [10].

Due to using a battery in our project design, we will insure we are following all safety and regulatory standards regarding preventing fires and explosive injuries from lithium powered devices including keeping the batteries at temperatures below 130° F and above 32° F, and making sure we are not dropping, crushing, or puncturing the 'Automatic Piano Tuner' [11].

References

[1] J. Ross, "How Much Does It Cost To Tune A Piano," Joshua Ross Piano. [Online]. Available: https://joshuarosspiano.com/how-much-does-it-cost-to-tune-a-piano/

[2] "Care and Maintenance of a Piano." [Online]. Available: https://www.yamaha.com/en/musical_instrument_guide/piano/maintenance/maintenance002.html

[3] "Cordless Electric Screwdriver Kit." [Online]. Available: https://nocry.com/product/cordless-screwdriver-kit/

[4] "Keyboard and Frequencies," Sengpiel Audio. [Online]. Available: http://www.sengpielaudio.com/calculator-notenames.htm

[5] J. Harold A. Conklin, "Tuning pin for pianos," U.S. Patent US4 920 847A, 1989. [Online]. Available: https://patents.google.com/patent/US4920847A/en

[6] "How Many Keys On A Piano?" Broughton Pianos. [Online]. Available: https://www.broughtonpianos.co.uk/blog/how-many-keys-on-a-piano

[7] "Ls-00086," Digi-Key. [Online]. Available: https://www.digikey.com/en/products/detail/osepp-electronics-ltd/LS-00086/11198629

[8] "Salary Averages." [Online]. Available: https://ece.illinois.edu/admissions/why-ece/salary-averages

[9] "Rankings and Statistics." [Online]. Available: https://ece.illinois.edu/admissions/why-ece/rankings-and-statistics

[10] "IEEE Code of Ethics," IEEE. [Online]. Available: https://www.ieee.org/about/corporate/governance/p7-8.html

[11] "Preventing Fire and/or Explosion Injury from Small and Wearable Lithium Battery Powered Devices," OSHA. [Online]. Available: https://www.osha.gov/sites/default/files/publications/shib011819.pdf

Appendix A Relevant Code

Frequency analyzer code on the microcontroller used to determine the input frequency based on the sound from the 'Audio Acquisition Subsystem'.

```
void ProcessFrame(sample_buf *dataBuf) {
    // Data is encoded in signed PCM-16, little-endian, mono
    float bufferIn[FRAME_SIZE];
    for (int i = 0; i < FRAME_SIZE; i++) {
        int16_t val = ((uint16_t) dataBuf->buf_[2 * i]) | (((uint16_t) dataBuf->buf_[2 * i + 1]) << 8);
        bufferIn[i] = (float) val;
    }

    // ********************** PITCH DETECTION ************************ //

    // Gather Frame power
    float frame_power = 0;
    for(int i = 0; i < FRAME_SIZE; i++){
      frame_power += bufferIn[i]*bufferIn[i];
    }

    // Test if the frame is voiced
    if(frame_power < VOICED_THRESHOLD){
      lastFreqDetected = -1;
      return;
    }

    // Perform FFT on the input signal
    kiss_fft_cpx *input = new kiss_fft_cpx[FFT_SIZE]; //Temp buffer for data processing
    kiss_fft_cpx *f1 = new kiss_fft_cpx[FFT_SIZE]; //Temp buffer for data processing
    float data_point; //temp variable for storing processed data point

    //Apply Hamming window & zero-pad in same for loop
    for(int i = 0; i < FFT_SIZE; i++){
      //If less than frame size, coeff, else 0
      data_point = bufferIn[i]*getHanningCoef(FRAME_SIZE, i);
      input[i].r = data_point;
      input[i].i = 0;
      f1[i].r = 0;
      f1[i].i = 0;
    }
    // FFT processing beginning
    // config object for kiss fft algorithm
```

```
    kiss_fft_cfg mycfg = kiss_fft_alloc(FFT_SIZE,0,NULL,NULL);
    kiss_fft(mycfg, input, f1); // Perform FFT, store fftOut
    free(mycfg);

    // Copy conjugate f1 and multiply by itself -> input
    for(int i = 0; i < FFT_SIZE; i++){
      //(a + jb)*(c - jd) = ac + j(-ad + bc) + bd
      // comput conjugate and multiply by fft output in same step
      // reuse input for ifft
      input[i].r = f1[i].r*f1[i].r + f1[i].i*f1[i].i;
      input[i].i = 0;
    }

    // Reuse f1 for ifft output
    kiss_fft_cfg newcfg = kiss_fft_alloc(FFT_SIZE,1,NULL,NULL);
    kiss_fft(newcfg, input, f1); // Perform FFT, store fftOut
    free(newcfg);

    // Stand-in for magnitude a^2 + b^2 without the square root
    float autocor_array[FRAME_SIZE];
    for(int i = 0; i < FRAME_SIZE; i++){
      autocor_array[i] = f1[i].r;
    }

    // Get max l
    float l = (float) findMaxArrayIdx(autocor_array, MIN_INDEX, MAX_INDEX);

    // Calculate the frequency
    lastFreqDetected = F_S/l;
}
```