

# ASSISTIVE DIGITAL PIANO

Final Report  
ECE 445: Senior Design

Shruti Chanumolu (shrutic2),  
Anna Shewell (shewell2),  
Jae Kwak (jaekwak2)

---

May 2, 2018  
TEAM 45, TA: Zipeng Wang

## Abstract

We created a digital piano that is capable of identifying which finger was used to press which key. This piano helps users learn how to play correctly by forcing them to use correct fingering. The piano uses photointerrupters for velocity sensitive key input, color sensors and colored finger sleeves for finger identification, and RGB LEDs to indicate which key should be pressed by which finger next.

# Contents

1.Introduction.....	4
1.1 Objective.....	4
1.2 Background.....	4
1.3 High-Level Requirements.....	4
2.Design.....	5
2.1 Block Diagram.....	5
2.2 Physical Design .....	5
2.3 Functional Overview of Blocks .....	7
3.Design	
Verification.....	13
3.1 Power Module.....	13
3.2 Control Module.....	14
3.3 Input Module.....	14
3.4 Indicator Module.....	15
3.5 Software Module.....	15
4.Cost and Schedule.....	16
4.1 Schedule.....	16
4.2 Cost.....	16
5.Conclusion and Further Resources.....	18
5.1 Accomplishments.....	18
5.2 Uncertainties.....	18
5.3 Ethics.....	19
5.4 Future work.....	20
6.References .....	21
Appendix A: Requirements and Verification Table .....	22

# 1 Introduction

## 1.1 Objective

Our project aims to develop an assistive digital piano for beginners who wish to teach themselves how to play. Many self-taught players lack the technical skills that other players possess. For example, self-taught players might not play at the right tempo or strength or they might rely too much on their index fingers. Therefore, we want to make an assistive digital piano that can teach the user these skills. Our piano will implement many novel features: it will use LEDs and colored finger sleeves to highlight correct fingering; after a song has been played, it will report the timing of the notes, the speed at which the keys were hit, and the number and types of mistakes so that users can identify what they need to improve; and it will provide a strict learning mode where the piano will wait for the correct finger to press the correct key before it plays the note.

## 1.2 Background

Many people are interested in learning how to play the piano. However, taking lessons requires money and the availability of a good teacher. The average cost of piano lessons in the U.S. is between \$15 and \$40 for 30 minutes of instruction. If a student takes 1 lesson per week, that adds up to approximately \$720 to \$2160 per year. In addition, it's been found that many beginning students lose their motivation to play in part because they don't like their piano teacher [1]. Thus, we want to design an assistive piano that would cost little more than a regular piano, but would allow the user to move at their own pace and improve their technical skills. In order to differentiate ourselves from the many other assistive pianos on the market today, our piano will teach fingering and give the user detailed feedback on metrics like rhythm accuracy.

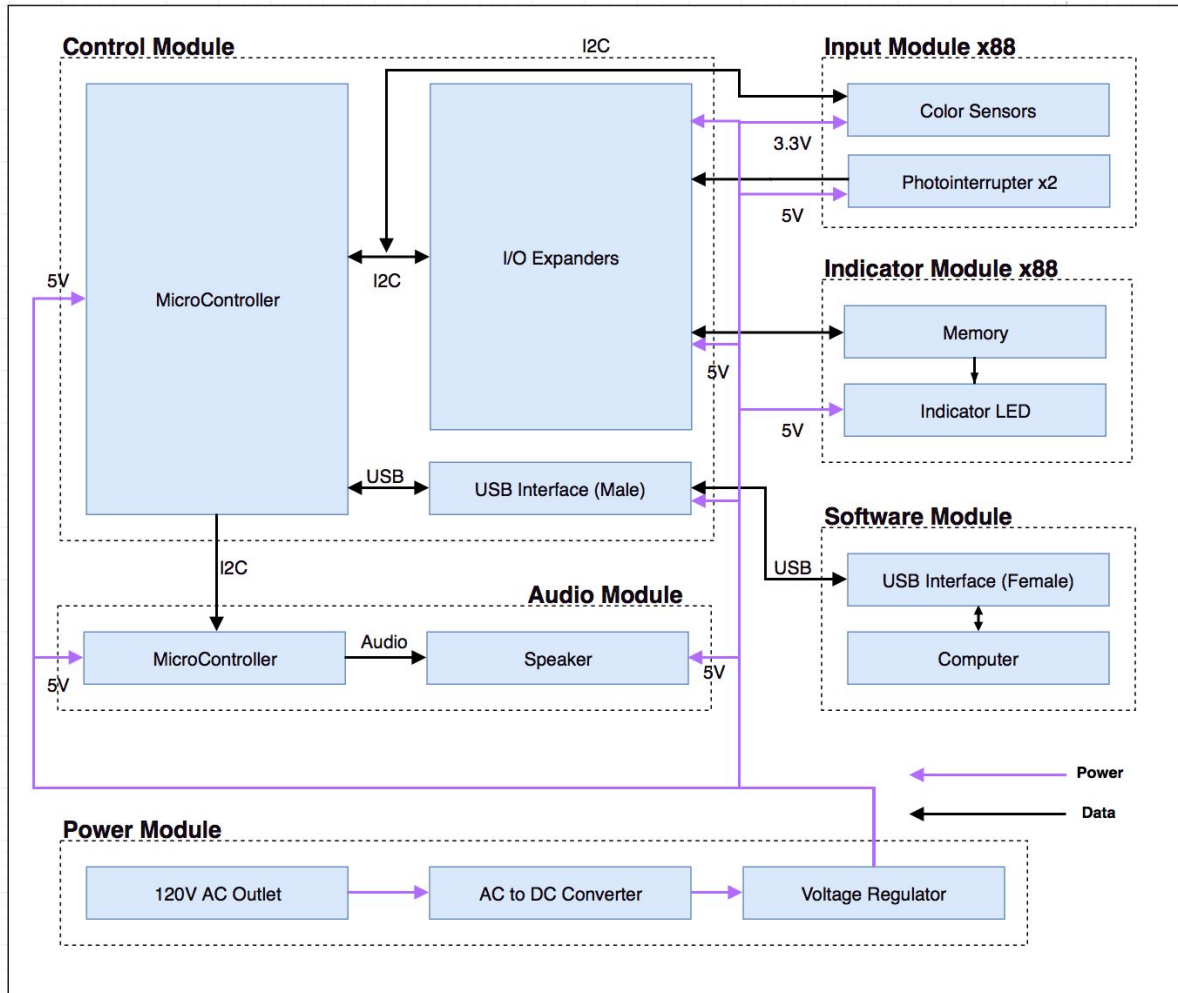
## 1.3 High-Level Requirements

- The piano must be able to detect key presses and play the corresponding notes from the speakers.
- The piano must be able to restrict key input to force the user to play along with a programmed song.
- The software must be able to read rhythm and keypress data from the piano and then evaluate and display timing and keypress accuracy to the user in a graphical interface.

## 2 Design

### 2.1 Block Diagram

Figure 2.1: Block Diagram



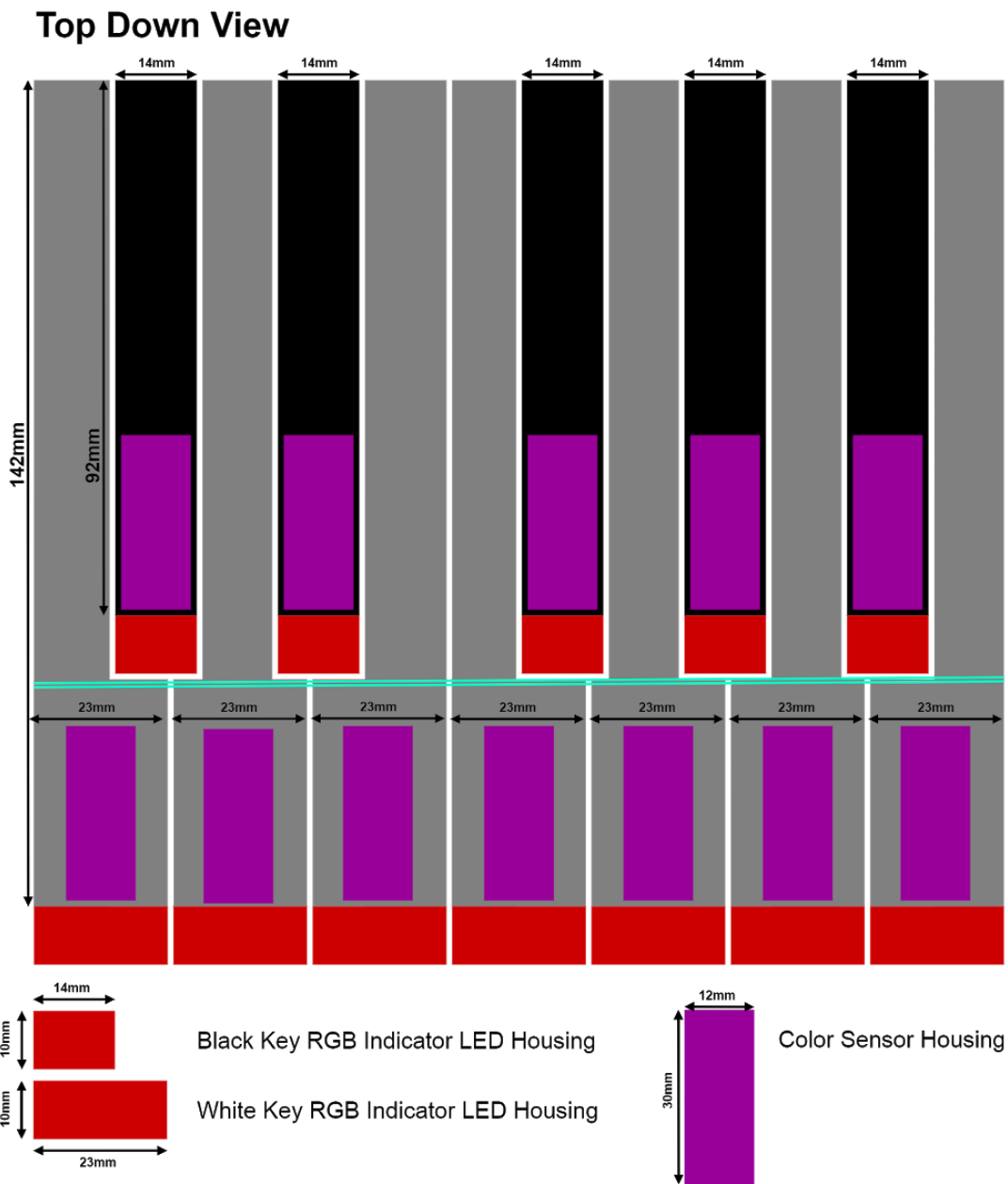
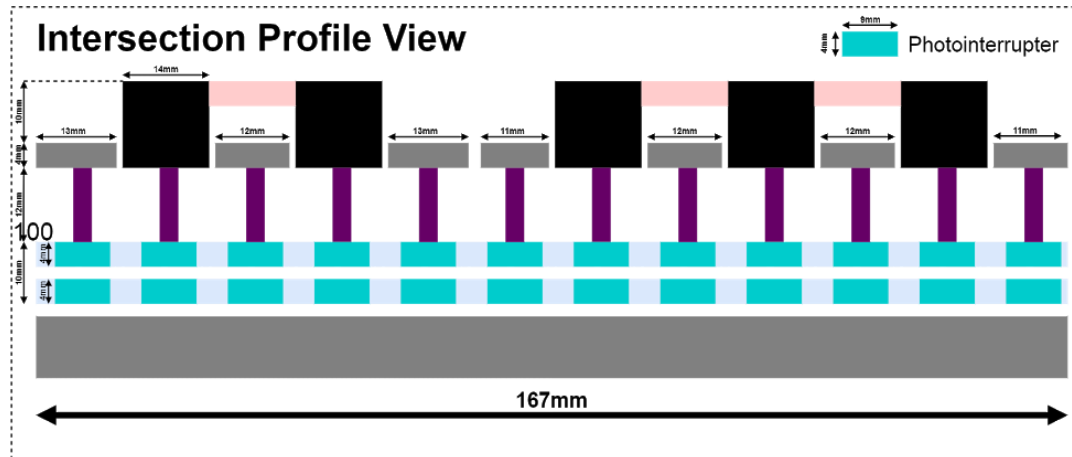
### 2.2 Physical Design

#### Keys

Two photointerrupters are placed beneath the keys. When the key is pressed, a thin (3 mm) protrusion on the bottom of the key triggers them sequentially (see section 2.3.3 for more information on the photointerrupters).

If the piano is in “strict” mode, our microcontroller simultaneously reads RGB values from the color sensors. As shown in the diagram below, the color sensor is located around where a user’s finger will be.

Figure 2.2: Key Dimensions and Sensor Layout



## 2.3 Functional Overview of Blocks

### 2.3.1 Power Module

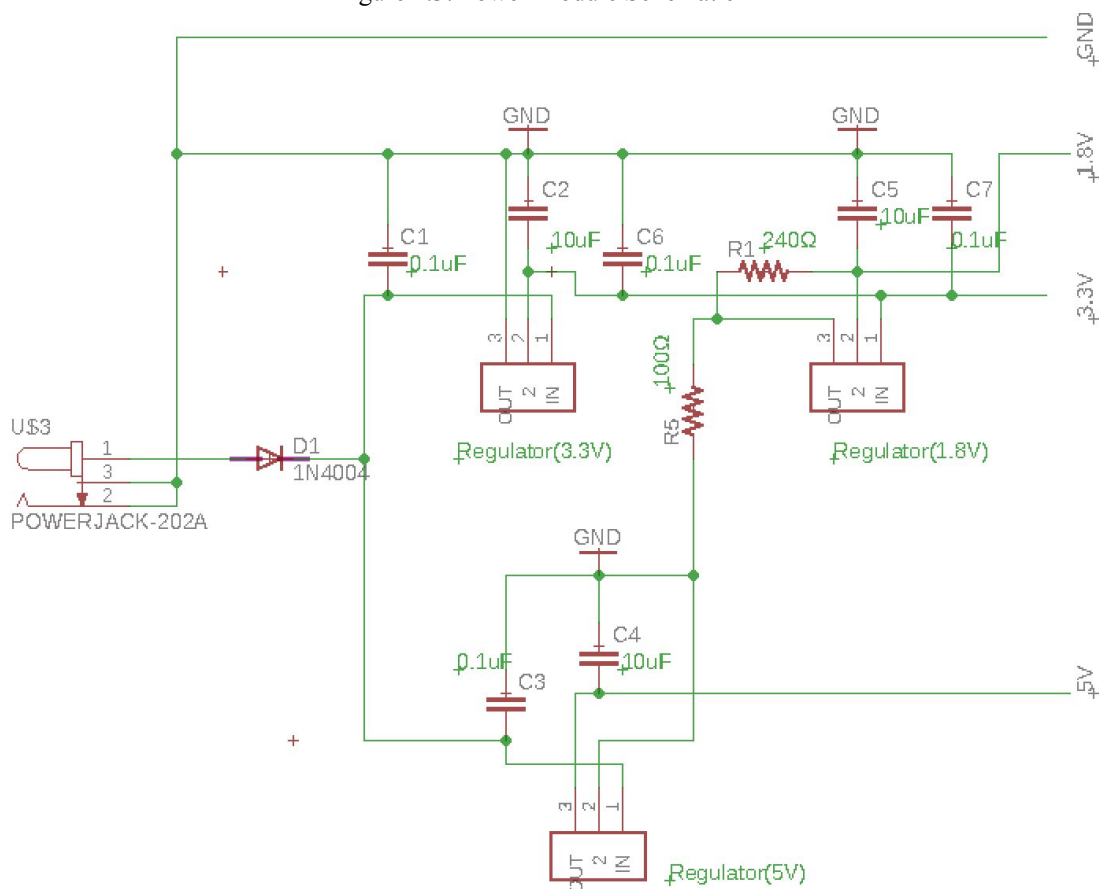
A 120-130 V AC wall outlet is used to power the system. The AC input is sent through an AC to DC converter to convert the 120 VAC to a steady 9 VDC power supply. Voltage regulators are used to produce 3.3 V, 5 V, and 1.8 V signals to power different modules of the system.

### Calculations

1.8 V generation from LM558 adjustable voltage regulator:

$$V_{out} = 1.8 \text{ V} = 1.25 (1 + R5 / R1)$$
$$R5 = 100 \, \Omega, R1 = 240 \, \Omega$$

Figure 2.3: Power Module Schematic



### 2.3.2 Control Module

The control module is responsible for handling the control logic and I/O (input/output) of the piano. This includes periodically scanning the photointerrupter inputs, reading color sensor values on full keypress, outputting keypress data to the software module, and outputting keypress data to the audio module. Due to the high number of I/O requirements (88 keys in theory, each with two photointerrupters and a color sensor), we plan to use port expanders.

If the piano is in “strict” mode, our microcontroller reads in RGB values from the color sensors to determine which finger has been used to press the key. If the fingering or keypress of the input is wrong while in this mode, the microcontroller will not send anything to the audio module.

Figure 2.4: Microcontroller Program Flow Chart

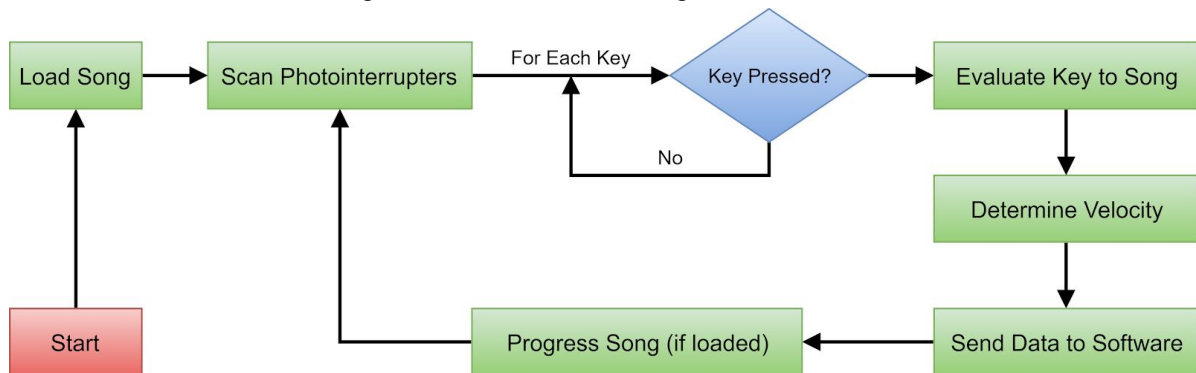
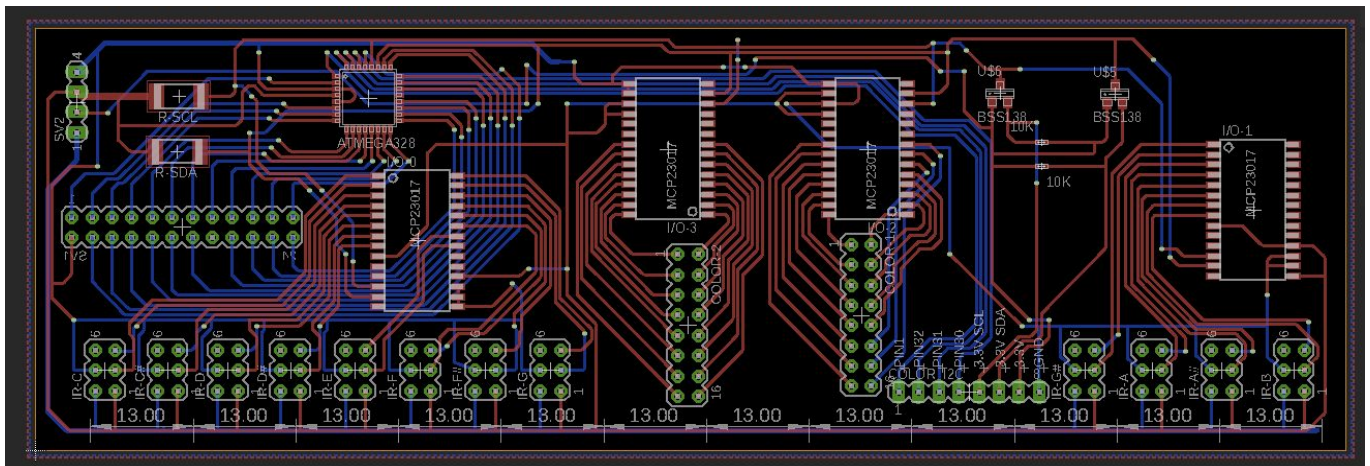


Figure 2.5: Control Module PCB design





### 2.3.3 Input Module

The input module handles receiving input from the user. It consists of two photointerrupters that get triggered sequentially when a key is pressed for the purpose of velocity-sensitive keypress detection and a color sensor to read the color of the finger that is over the key.

#### Photointerrupters

Two photointerrupters are attached sideways underneath each key that will get triggered sequentially when the key is pressed. The purpose of having two is for velocity sensing. When the first photointerrupter is triggered, the microcontroller will start timing the keypress. When the second photointerrupter is triggered, the microcontroller will stop timing the keypress. Since the two sensors are stationary and spaced a known distance apart, velocity can be computed. The computed velocity will then be used to determine the volume of the sound output.

#### Color Sensor

Color Sensors (TCS3471) are placed beneath the transparent keys to detect if the correct finger is used to press the key. Once a key is pressed, the control module identifies the key and outputs select bits to an I2C multiplexer (TCA9548A), which polls RGB data from the corresponding color sensor. These sensors read the color on the fingers of a set of included finger sleeves and send this data to the control module using an I2C bus. The control module will then compare the RGB values of the signal sent by the color sensor to the color of the finger that was supposed to be used. Thus, it can determine if correct fingering is being used to press the key or not. Moreover a Bidirectional I2C level shifter has been used to shift the SDA and SCL lines between the 3.3 V color sensor and 5 V microcontroller.

#### Calculations

---

##### Color Sensor Read Time

Device Address: 8 bits per color sensor

Register Address: 8 bits per RGB component

Color Output: 16 bits per RGB component

$(\text{Device Address} + \text{Register Address} + \text{Color Output}) * 3$

$(8 + 8 + 16) * 3 = \mathbf{96 \text{ bits}}$  for full RGB signal

At 400 kbps:  $96 \text{ bits} / 400\text{k} = \mathbf{0.24 \text{ ms}}$

Assuming the slower data transfer rate of 400 kbps, ten color sensors can be read one after the other in approximately:

### 2.3.4 Indicator Module

On each key, one RGB LED (instead of ten regular LEDs per key) guides users through a preprogrammed song. The keys that need to be played next light up in 1 of 10 colors corresponding to the colors on the set of included finger sleeves.

We decided on a simpler final design for this module after we discovered interference issues with the PCB (Printed Circuit Board) of the original design. Instead of using a 50% PWM (Pulse Width Modulation) signal, we used resistor combinations to create the correct power signals. Instead of storing the color codes in memory, we use port expanders to directly connect the indicator module to the control module. With this setup, each key requires six I/O pins, two for each RGB color component. Those two pins correspond to setting that RGB component to approximately 255 or 128. Setting neither corresponds to setting that RGB component to 0. In this way, we can produce ten colors.

Table 2.1: RGB Color Values

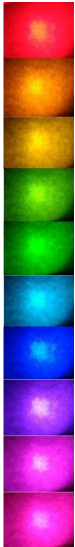
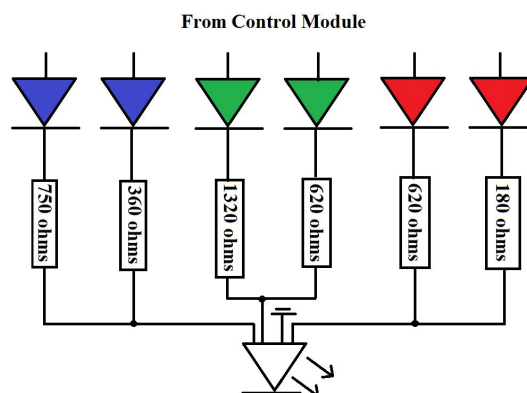
Finger	Color	RGB	Resulting LED Colors
Left Pinky	Red	(255, 0, 0)	
Left Ring Finger	Orange	(255, 128, 0)	
Left Middle Finger	Yellow	(255, 255, 0)	
Left Index Finger	Yellow Green	(128, 255, 0)	
Left Thumb	Green	(0, 255, 0)	
Right Thumb	Sky Blue	(0, 255, 255)	
Right Index Finger	Dark Blue	(0, 0, 255)	
Right Middle Finger	Purple	(128, 0, 255)	
Right Ring Finger	Magenta	(255, 0, 255)	
Right Pinky	Pink	(255, 0, 128)	

Figure 2.8: Indicator Module Schematic



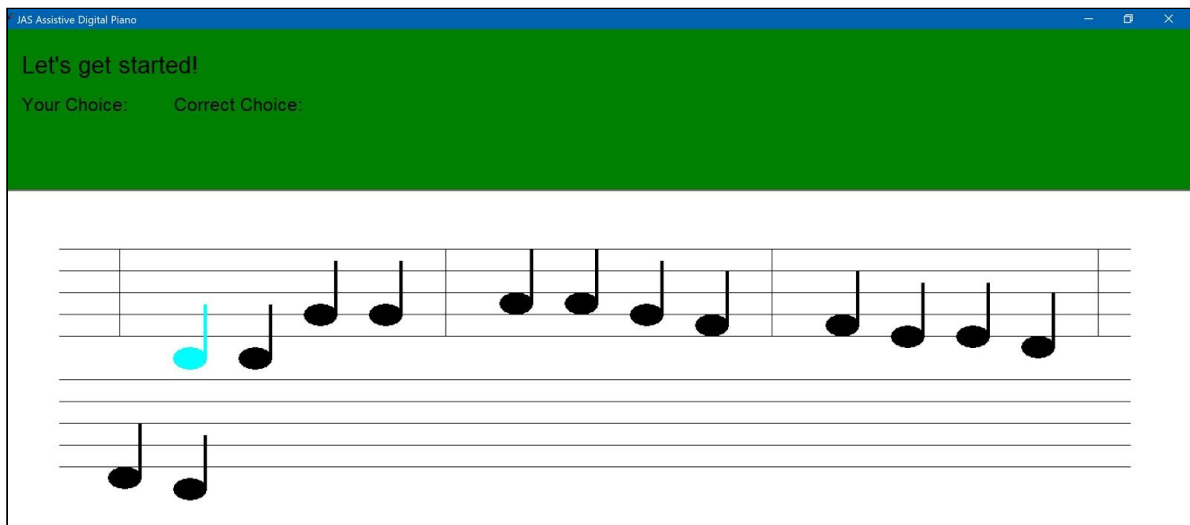
### 2.3.5 Audio Module

The purpose of this module is to produce the sounds corresponding to the keys pressed. We implemented this module through the computer. The microcontroller sends a couple bytes identifying the key and volume to the computer, and the computer generates the sound.

### 2.3.6 Software Module

This module consists of two pieces of software: a GUI (Graphical User Interface) and the interface between the piano and the computer. The GUI displays sheet music with notes that light up in the color matching the finger the user should use to play the note. The GUI also gives immediate feedback to the user when a correct or incorrect note is played. The interface between the piano and the computer records all keypresses.

Figure 2.9: GUI



## 3 Design Verification

### 3.1 Power Module

This module was verified by connecting the the input, indicator, and control modules to the power circuit and measuring the voltages across a component of each module using a voltmeter. The voltmeter reading showed the voltage input of the microcontroller, photointerrupter, and indicator module to be 5.0V. The voltage across each color sensor was measured to be 3.3V which satisfies our power requirements.

The 1.8 V voltage regulator of the power module module failed to supply the required 350 mA to power all the photointerrupters. It was only able to supply 250 mA of current at 1.67V. Power was dissipated in the form of heat due to the large voltage difference between the input and output terminal of the 3.3 V voltage regulator. Connecting the input of this regulator to 5V supply rather than 9V supply could have helped.

### 3.2 Control Module

#### 3.2.1 I/O Expanders

The I/O Expanders were tested for compatibility and reliability when used to communicate with the microcontroller. The test was done by setting the I/O Expander ports to output mode and setting I<sup>2</sup>C speed to 1.7MHz. A byte of data was written to port A, and then the same port was read back. The data did not change, confirming that I<sup>2</sup>C communication was working perfectly between the two components.

#### 3.2.2 Microcontroller

The microcontroller needed to output the right signals to the LEDs and be able to identify 10 keypresses simultaneously. The test for the LED signalling was not possible, since the indicator module was not implemented the way that we had planned, but a simple test to light up the proper LEDs was conducted. The signals were sent by setting I/O expander ports to inputs or outputs depending on which keys needed to be pressed. We confirmed visually that the correct LEDs were lit.

The test for 10 simultaneous keypresses was done using a power supply. We connected 10 of the key inputs to a power supply. We set the power supply to 5V to signify that all keys were not being pressed. Then, we turned the power supply off to simulate 10 simultaneous keypresses. The microcontroller was able to detect them all within a single loop and output the acknowledgement of key detection to the serial output on the computer.

## 3.3 Input Module

### 3.3.1 Photointerrupters

Each PCB had two photointerrupters. They were tested by linking a 5V power supply and 1.8V power supply to the PCB and measuring the voltage at the two output terminals with a voltmeter. The circuit needed to output  $5 \pm .3V$  when the slot was empty and  $0 \pm .3V$  when blocked, independently of each other. They each passed the verification with  $\sim 4.82V$  for non-blocking and  $\sim .02V$  for blocking the slot.

### 3.3.2 Color Sensors

The color sensor needed to identify ten colors corresponding to ten fingers. Table 3.1 gives the RGB values read by the color sensor when calibrated as detailed in section 2.3.3 and programmed using an Arduino. Each color was tested approximately 15 to 20 times to get the below mentioned results. These RGB values distinctly classify all 10 colors.

To check for correct fingering, 10 color sensors were connected to an I2C multiplexer and each sensor was programmed using an Arduino to identify a particular color on the finger sleeve. When a finger sleeve was placed on a particular color sensor such that incident light value is less than 5000 (The finger sleeve is less than 50 mm from the color sensor). The color sensor outputted the finger corresponding to that color.

The recorded RGB values fluctuated whenever the color sensor was moved and the ambient light changed. In order to maintain consistency in the future, a clear LED should be added on the color sensor PCB. Moreover, the sensor often incorrectly identified similar colors. For example, red and pink were often too similar. Going forward, finger sleeve colors need to be carefully chosen with the color sensor in mind.

Table 3.1: RGB Results for 10 Colors

COLOR	RED	GREEN	BLUE	FINGERING
RED	1.60-1.66	0.23-0.25	0.31-0.34	Left Pinky
ORANGE	1.62-1.68	0.25-0.3	0.26-0.28	Left Ring Finger
YELLOW	1.6-1.69	0.55-0.6	0.20-0.22	Left Middle Finger
YELLOW-GREEN	1.2-1.22	0.53-0.6	0.31-0.35	Left Index Finger
GREEN	0.8-1.10	0.5-0.6	0.22-0.25	Left Thumb
SKYBLUE	1.07-1.10	0.5-0.61	0.43-0.46	Right Thumb
DARK-BLUE	0.86-1.0	0.43-0.56	0.65-1	Right Index Finger
PURPLE	1.2-1.23	0.48-0.53	0.5-0.8	Right Middle Finger
MAGENTA	1.4-1.55	0.33-0.35	0.41-0.44	Right Ring Finger
PINK	1.57-1.62	0.23-0.26	0.33-0.36	Right Pinky
WHITE	1.7-1.9	0.7-0.9	0.7-1	

### 3.4 Indicator Module

The indicator module had two requirements: the LEDs needed to light up in ten distinct colors and the latency between the microcontroller and the LEDs needed to be imperceivable. Both requirements were met through a visual test.

### 3.5 Software Module

The software module was required to send and receive data through a serial connection to the microcontroller. It was able to receive keypress data perfectly, but was not able to send data to the microcontroller.

## 4 Costs and Schedule

### 4.1 Schedule

Table 4.1: Individual Task Breakdown

		<b>Shruti Chanumolu</b>	<b>Anna Shewell</b>	<b>Jae Kwak</b>	<b>Expected Time Per Person</b>
<b>1</b>	<b>2/26</b>	Incorporate feedback, purchase parts, finish PCB design.			12 hrs
<b>2</b>	<b>3/5</b>	Design data analysis algorithm.	Design GUI.	Construct keys.	10 hrs
<b>3</b>	<b>3/12</b>	Connect data analysis algorithm to GUI, finish/edit PCB design.		Construct case.	7 hrs
<b>4</b>	<b>3/19</b>	Spring Break			-
<b>5</b>	<b>3/26</b>	Solder/test power and audio circuitry.	Solder/test indicator circuitry.	Solder/test sensor circuitry.	15 hrs
<b>6</b>	<b>4/2</b>	Solder control circuitry and connect to all other modules.		Test color sensor	12 hrs
<b>7</b>	<b>4/9</b>	Program and test microcontroller.			10 hrs
<b>8</b>	<b>4/16</b>	Mount all circuitry and re-test connections.			7 hrs
<b>9</b>	<b>4/23</b>	Test final product, write final paper and presentation.			10 hrs

### 4.2 Cost

#### 4.2.1 Cost of Labor

In order to estimate a reasonable salary for our group members, we looked up the average yearly salary data for ECE graduates collected by Engineering Career Services (ECS) [6]. We chose the averages from 2013 - 2014, the most recent data available (not counting data from 2014 - 2015 which, according to ECS, is still subject to change). This data is self-reported and thus, may be subject to bias. From this average salary information, we applied the standard used by the U.S. Office of Personnel Management [7] to estimate hourly salaries.

Table 4.2: Summary of Salary Data

	<b>Average Yearly Salary</b>	<b>Average Hourly Salary</b>
<b>CompE</b>	\$77,653	$\$77,653 / 2,087 \text{ hrs} = \text{\$37.21 per hr}$
<b>EE</b>	\$69,342	$\$69,342 / 2,087 \text{ hrs} = \text{\$33.23 per hr}$



Based on our schedule, we estimated that each member contributed approximately 85 hours total over the course of the project. Thus, the total cost of our labor was:

Table 4.3: Labor Costs

Shruti Chanumolu	EE	$\$33.23 * 85 = \$2,824.55$
Jae Kwak	EE	$\$33.23 * 85 = \$2,824.55$
Anna Shewell	CompE	$\$37.21 * 85 = \$3,162.85$
<b>Total</b>		<b>\$8,811.95</b>

#### 4.2.2 Cost of Parts

Table 4.4: Total Parts Costs

Power Module	\$2.55
Control Module	\$12.24
Input Module	\$90.48
Indicator Module	\$12.29
Audio Module	\$3.75
Miscellaneous	\$92.00
<b>Total</b>	<b>\$213.31</b>

#### 4.2.3 Total Costs (Labor + Parts)

Table 4.5: Total Costs

Labor	\$8,811.95
Parts	\$213.31
<b>Total</b>	<b>\$9,025.26</b>

#### 4.2.4 Total Costs For Full-Size Piano

Our budget calculations apply only for the single octave (12 keys) that we created. For a full piano with ~7 octaves (88 keys), the number of sensors and LEDs we'd need increases the cost significantly (see tables 3.11 and 3.12). This estimate may be a little high due to the cost savings of bulk ordering. In any case, our estimate falls within the average cost range of commercial full-size digital pianos (based on the products listed by [8], we estimate a range of ~\$200 - ~\$1,500).

Table 4.6: Total Costs (Labor + Parts) For Full Piano

Labor	\$8,811.95
Parts	\$864.19
<b>Total</b>	<b>\$9,676.14</b>

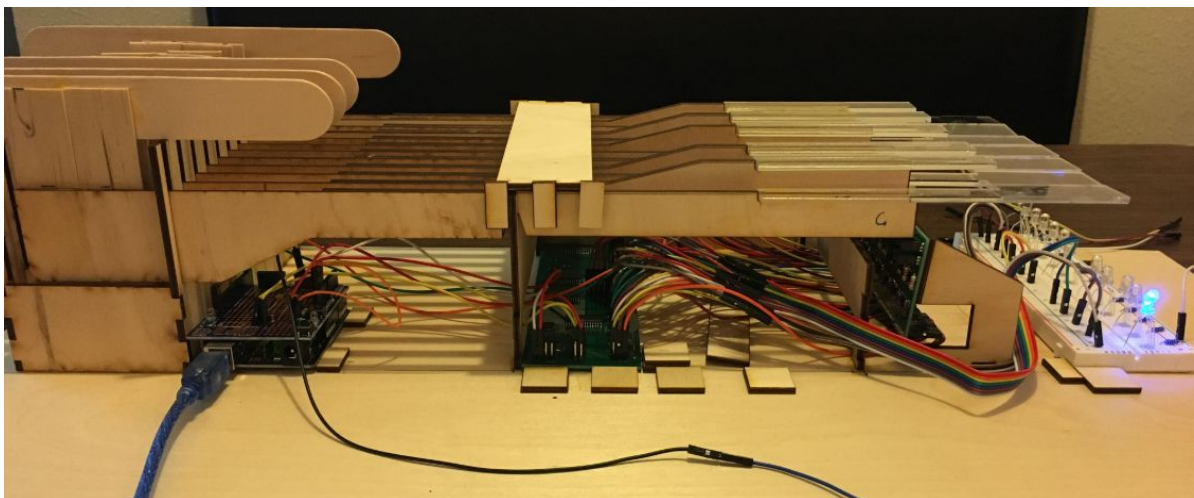
## 5 Conclusion and Future Work

### 5.1 Accomplishments

Although most of our requirements met the verifications, we were only able to integrate a select few components into the final product. These included a fully functional control module, an input module with photointerrupters, a simple LED indicator, and a note verification system (without finger verification).

To play the piano, a laptop needs to be plugged in using a USB connector. A programmed song will be loaded upon startup. The user then loads a python program that will begin listening to incoming data from the piano. The indicator LEDs will light up in front of the keys that need to be pressed. Once the correct keys are pressed, the piano will move onto the next notes in the programmed song. When the song is finished, all LEDs will turn on and the piano can then be played like a normal piano.

Figure 5.1 Final Product



One of the major successes of our project was implementing velocity sensitive keys. Although some of the keys were misaligned and caused issues with note detection, we think that the system could be used in a real piano as a reliable note detection mechanism. A small issue was that a note could not be played unless the key was fully released. This is not how a real piano works, but could be easily fixed by changing the software or using a better physical mechanism

### 5.2 Uncertainties

In our final product, we had a couple issues. One involved the photointerrupters. Of the 12 keys that we created, only 11 were fully functional and one was unreliable. The

broken key seemed to have a malfunctioning photointerrupter. This meant that it was impossible for the microcontroller to detect a keypress. The unreliable key had photointerrupters working perfectly. Upon testing, however, we confirmed that the issue was due to the photointerrupter PCB being misaligned with the key. The key simply needed to be pulled up fully before being pressed again. The other issue involved the serial connection to the computer. Although data was being sent from the microcontroller to the Python program, data was not being sent reliably back to the microcontroller. This was caused by the software being on a 64 bit operating system. The data was not being sent until the serial buffer was filled with 64 bits of data. This was not resolved in time for the demo.

Though we met the verifications for many of our other components, we could not fully integrate them into the system for the demo. For example, although the color sensors met the verification that we specified, they were very unreliable and shut down periodically. Moreover, we had difficulty physically mounting it to the bottom of the keys in such a way that it would actually read the finger sleeves. In the end, we decided to forego physical integration. Another example is that although a few indicator LEDs were built, we did not have enough for a full octave. Since it would have been painful to write a microcontroller program that distinguished between working PWM LEDs for a few keys and simple LEDs for others, we decided to leave the few finished indicator LEDs out of the final product and use a makeshift breadboard LED circuit instead. This allowed us to properly indicate which key to press for all 12 keys, rather than 2 or 3.

### 5.3 Ethical Considerations

As with anything that connects to a wall outlet, there's a chance of electrocution or fire if the plug/circuitry is tampered with, misused, or accidentally compromised. To ensure the safety of the operator, as well as to comply with the IEEE [6] and ACM Code 20 of Ethics [7] we designed our piano with proper circuit protections. Moreover we added heat sinks to our voltage regulator to dissipate excess power.

We have accurately reported the capabilities of our project in addition to discussing all its shortcomings. Doing so we maintained the integrity of our work in keeping with the IEEE Code of Ethics #3, "...to be honest and realistic in stating claims or estimates based on available data," [2].

Also of great importance to our project is the IEEE Code of Ethics #7, "...to seek, accept, and offer honest criticism of technical work, to acknowledge and correct errors, and to credit properly the contributions of others," [2]. We will welcome criticism of our work and do our best to fix any errors in our project. In keeping both with the aforementioned IEEE policy and also with one of the University of Illinois's most important policies, we will not plagiarize anyone else's work and will give credit to outside sources whenever necessary.

Lastly, in keeping with the IEEE Code of Ethics #10, “...to assist colleagues and co-workers in their professional development and to support them in following this code of ethics,” [2], we will offer our aid to any of our fellow peers in their projects should they need it given we have the requisite knowledge and resources to do so.

## 5.4 Future Work

We plan to continue work on this project and to make improvements in the future. The first thing we will do is fix the components that we could not finish in time for the demo in the class - namely the indicator module and the power module. Once these are implemented, the project could be used as a base for improvements.

The next thing we would do is add the piano functions that were omitted for simplicity in the project. These functions include physical buttons for controlling the power and mode, pedal integration, onboard speakers, audio synthesizing, and metronoming. In addition, a sturdy, reliable, and well designed physical mechanism would be necessary to create a piano that is worth playing on.

Finally, we would consider alternatives to the solutions we found in this design to make this piano a better and more attractive learning tool than what is on the market today. An alternative idea for tracking fingers is to use image recognition across the breadth of the keys. Though it might be difficult to analyze fingering in real time with this method, it could be combined with a tracking technology that is placed on the finger to aid in recognition or be used after a song is played for analysis on which fingers were used. Since there are only 10 fingers on two hands, the recognition software could potentially be very accurate using a process of elimination as an aid to recognize fingers.

## 6 References

- [1] A. Mazzocchi, “Why Students Really Quit Their Musical Instrument (and How Parents Can Prevent It),” *The Music Parents’ Guide*, 17-Feb-2015. [Online]. Available: <http://www.musicparentsguide.com/2015/02/17/students-really-quit-musical-instrument-parents-can-preven>. [Accessed: 08-Feb-2018].
- [2] “IEEE Code of Ethics,” *IEEE*. [Online]. Available: <http://www.ieee.org/about/corporate/governance/p7-8.html>. [Accessed: 08-Feb-2018].
- [3] “How Colour Changing LEDs Work,” *Kitronik*. [Online]. Available: <http://www.kitronik.co.uk/blog/how-colour-changing-leds-work>. [Accessed: 08-Feb-2018].
- [4] “Tempo,” *Wikipedia*. [Online]. Available: [en.wikipedia.org/wiki/Tempo#Basic\\_tempo\\_markings](http://en.wikipedia.org/wiki/Tempo#Basic_tempo_markings). [Accessed: 08-Feb-2018].
- [5] “Musical Keyboard,” *Wikipedia*. [Online]. Available: [en.wikipedia.org/wiki/Musical\\_Keyboard](http://en.wikipedia.org/wiki/Musical_Keyboard). [Accessed: 08-Feb-2018].
- [6] “Salary Averages”, ECE Illinois. [Online]. Available: <https://ece.illinois.edu/admissions/why-ece/salary-averages.asp>. [Accessed 22-Feb-2018].
- [7] “Computing Hourly Rates of Pay Using the 2,087-Hour Divisor”, *OPM.GOV*. [Online]. Available: <https://www.opm.gov/policy-data-oversight/pay-leave/pay-administration/fact-sheets/computing-hourly-rates-of-pay-using-the-2087-hour-divisor/>. [Accessed 22-Feb-2018].
- [8] “Digital Pianos”, Guitar Center. [Online]. Available: <http://www.guitarcenter.com/Digital-Pianos.gc>. [Accessed 26-Feb-2018].

## Appendix A: Requirements and Verification Table

COMPONENT	REQUIREMENTS	VERIFICATION	VERIFIED
AC to DC Converter (Power Module)	Power from 120V AC and output a $9 \pm 0.5V$ DC	<p>A. Connect one end of the AC-DC adapter to a 120V AC source and the other end to a DC Jack mounted on a breadboard</p> <p>B. Attach the Voltmeter to the DC Jack output</p> <p>C. Voltmeter Reading: <math>9 \pm 0.5V</math></p>	Y
Voltage Regulator 1 (3.3V) (Power Module)	The voltage across the output terminal and ground must be $3.3 \pm 0.5 V$ for a current load up to $3.4 \pm 0.5 mA$ when connected to a DC supply of 9V.	<p>A. Connect a voltmeter across the VCC pin of the the color sensor and ground. The Voltmeter reading (measured voltage) should be around <math>3.3 \pm 0.5 V</math>.</p> <p>B. Connect a multimeter across the VCC pin of the the color sensor and ground. The Multimeter reading (measure current )should be around <math>3.4 \pm 0.5 mA</math></p>	Y
Voltage Regulator 2 (5V) (Power Module)	The voltage across the output terminal and ground must be $5 \pm 0.5 V$ when connected to a DC supply of 9V.	<p>A. Connect a voltmeter across the VCC pin of the the Photointerrupters and ground. The measured voltage should be <math>5 \pm 0.5 V</math>. The Voltmeter reading (measured voltage) should be around <math>5 \pm 0.5 V</math>.</p> <p>B. Connect a voltmeter across the anode and cathode pins of the the LED . The Voltmeter reading (measure voltage) should be around <math>5 \pm 0.5 V</math>.</p>	Y

COMPONENT	REQUIREMENTS	VERIFICATION	VERIFIED
Color Sensor (Input Module)	Must be able to identify 10 distinct colors, in the visible range	<p>A. Take ten samples of different colors each in the visible range</p> <p>B. Connect the color sensor to arduino board. (VCC-3.3V, GND-GND, SDL-SDL,SCL-SCL)</p> <p>C. ADD the color sensor library into /arduino/libraries</p> <p>D. Read RGB data through the library function</p> <p>E. The intensities of these RGB data should be different for all the 10 colors</p>	Y
	Must not be able to identify a color beyond the area of the key (14±2 mm X 92 ±2 mm )	<p>A. Take a color sample (red paper) of dimension 14±2 mm X 92 ±2 mm . Place it on a different color sample (blue paper) of dimension greater than 14±2 mm X 92 ±2 mm</p> <p>B. Repeat B to D from above by placing the color sensor 10mm above the centre color strip (red paper)</p> <p>The color sensor reads red color and not the blue color</p>	Y
	Must be able to register color changes in atmost 146 ms.	<p>A. Make a color wheel of 10 distinct colors and of radius 10cm.</p> <p>B. Spin the wheel such that colors are changing at 145ms form a point where the color sensor is placed.</p> <p>C. Repeat steps 1-B to 1-D</p> <p>D. The intensities of the RGB data should change</p>	Y

COMPONENT	REQUIREMENTS	VERIFICATION	VERIFIED
Photointerrupter (Input Module)	Top sensor must be able to detect key press when a key is pressed 3 mm from its resting position, measured from above the sensor.	<p>A. Supply 5V and 1.8V DC power to the power inputs of the photointerrupter PCB.</p> <p>B. Connect a digital multimeter across the output pin 2 of the photointerrupter and GND.</p> <p>C. The multimeter must show <math>0\pm.3V</math> if the key is pressed by more than 3mm. It should show <math>5V\pm3V</math> if the key is pressed any shallower.</p>	Y
	Bottom sensor must be able to detect key press when a key is pressed 9mm from its resting position, measured from above the sensor.	<p>A. Supply 5V and 1.8V DC power to the power inputs of the photointerrupter PCB.</p> <p>B. Connect a digital multimeter across the output pin 1 of the photointerrupter and GND.</p> <p>C. The multimeter must show <math>0V\pm.3V</math> if the key is pressed by 9mm .It should show <math>5V\pm.3\%</math> if the key is pressed any shallower.</p>	Y

COMPONENT	REQUIREMENTS	VERIFICATION	VERIFIED
I/O Expander (Control Module)	Processes I <sup>2</sup> C to the I/O expanders at a rate of 1.7 Mbps.	<p>A. Connect two of the pins to a I<sup>2</sup>C bus that is connected to a 16 bit I/O expander.</p> <p>B. Write 8 bits of data to port A of the I/O expander set at a rate of 1.7 Mbps.</p> <p>C. Read 8 bits of data from port A of the I/O expander set at a rate of 1.7 Mbps.</p> <p>D. Verify that the data does not change from reading/writing.</p>	Y



COMPONENT	REQUIREMENTS	VERIFICATION	VERIFIED
Microcontroller (Control Module)	Outputs signals to correct LEDs with 100% accuracy.	<p>A. Connect VCC pin of the microcontroller to <math>5 \pm 5\%</math> V.</p> <p>B. Set the output bits of the microcontroller to identify all keys of the piano, one at a time.</p> <p>C. Ensure that the correct LEDs turn on.</p>	Y
	Outputs correct color codes to LEDs with 100% accuracy.	<p>A. Connect VCC pin of the microcontroller to <math>5 \pm 5\%</math> V.</p> <p>B. Set the output bits of the microcontroller to identify one of the piano keys.</p> <p>C. Send all possible color codes to the LED.</p> <p>C. Ensure that the LED lights up as expected.</p>	Y
	Must be able to correctly identify at least 10 keys pressed within 1 ms of each other.	<p>A. Connect the microcontroller to a computer through the USB interface.</p> <p>B. Press any 10 keys at the same time.</p> <p>C. Echo key data to the computer and display that information on the screen.</p> <p>D. Ensure that 10 key presses have been identified and that they correspond to the respective keys pressed.</p>	Y

COMPONENT	REQUIREMENTS	VERIFICATION	VERIFIED
Indicator Module	Must be able to produce 10 distinct colors matching the colors on the provided finger sleeves such that the RGB values are within 50 degrees of each other (per RGB component).	<p>A. Take photos of one of the LEDs, one photo for each of the ten colors.</p> <p>B. Using a computer, find the RGB values of the color produced.</p> <p>C. Compare each component individually.</p>	Y
	Must be able to respond to microcontroller's signal and change which keys are lit within $10 \pm 0.5$ ms.	<p>A. Store two colors in the indicator module pcb by mimicking the microcontroller's signals with an Arduino.</p> <p>B. Set the indicator module to the address of the first color stored.</p> <p>C. Use the Arduino to increment the address and time the delay between sending the address and the output of the circuit changing.</p>	

COMPONENT	REQUIREMENTS	VERIFICATION	VERIFIED
Software Module	Must be able to send a set of "next keys and fingers" to the microcontroller.	<p>A. Send keypress data to the microcontroller.</p> <p>B. Verify that the microcontroller is processing the next note correctly with a rate of 99% accuracy.</p>	N
	Must be able to download key press data from the control module.	<p>A. Send a sequence of keypress inputs from the microcontroller to the software.</p> <p>B. Verify that the software detected at least 99% of the keypress inputs.</p>	Y

COMPONENT	REQUIREMENTS	VERIFICATION	VERIFIED
Audio Module	Must be able to convert the key press code into the corresponding waveform.	<p>A. Connect the output of the audio controller to an oscilloscope and press all keys on the piano, one at a time.</p> <p>B. Ensure that the frequency of the waveform matches the key pressed.</p>	Y
	Must be able to produce sound at volumes between 65 and 95 dB. The sound should be clearly audible from a distance of 1m.	<p>A. Connect the speaker to a computer and play a song.</p> <p>B. Check manually by hearing the song if it is in the audible range of human hearing from at least 1m away from the Piano.</p>	Y