ASSISTIVE DRUM SET

ECE 445, SENIOR DESIGN

By

Calvin Fisher
Francis Papa
Nicholas Schiesl

Final Report for ECE 445, Spring 2020
TA: Evan Widlowski

08 May 2020

Team No. 63

Abstract

The problem addressed in both projects is that of providing musical education without an instructor. Both solutions aim to provide an educational value to the user with an assistive learning tool. Our solution builds on top of that with functionality value as well. The Assistive Keyboard provides song tutorial through light up LED keys. The Assistive Drum Set provides a hardware and software add on for recording music; while giving uneducated users visual reference to their music through sheet music, showing how performance affects the page.

Contents

1/ Second Project Motivation

1.1/ Problem Statement

Learning how to correctly transcribe and record sheet music is an expensive and tedious task. This is especially true for percussion players. Many drum set players rely on oral translation when learning to play due to the fact that drum lessons can be expensive. This results in the player not getting the opportunity to learn how to read or write percussion notation also referred to as sheet music. Being able to read and write sheet music is an important technical skill that is essential to a performer's success. If a player wanted to bypass learning how to read or write sheet music, they would have to use a piece of software to translate their sound into sheet music. This software can be incredibly expensive for consumers with an average price of about $1 per minute of audio translation. The high cost for the use of this software may result in some drum set players never getting the opportunity to transcribe their music.

1.2/ Solution

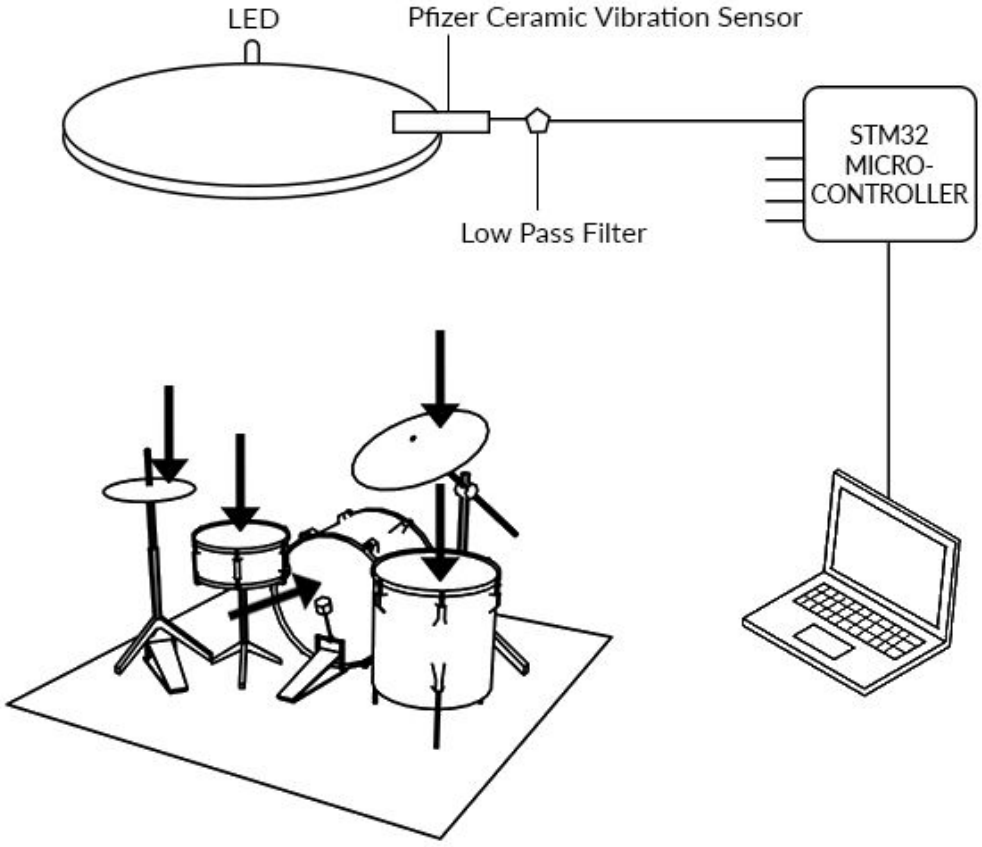
Our solution, the assistive drum set, will provide a much more affordable solution to learn how to read and write sheet music. The assistive drum set will eliminate the need for professional lessons and expensive audio translation software. This product will feature a record mode in which the user will be able to play their drums and their music will then be transcribed into sheet music. During record mode, the system will map each drum hit to its respective percussion notation. This will allow the user to be able to document what they play and create their own sheet music. The ability to see what was played on the drums on transcribed sheet music will be an excellent learning tool for drummers. The record mode will also output a metronome so the user's timing ability is maximized. The device will consist of small modules containing piezoelectric vibration sensors attached to mute pads that sit on each drum head and cymbal. These modules will all be connected to different microcontroller inputs in the control module. When a drum is hit, the microcontroller will receive a voltage signal and then send these signals to the computer, via USB, where it will be mapped to an image as its respective percussion notation. By using low cost components this device will be a more affordable learning tool for percussionists just beginning to read and write sheet music compared to expensive lessons or translation software.

In addition to being an effective learning tool for new drummers, we find that drummers of all experience levels will find our device useful. The sheet music translation element of the record mode provides a new way for musicians to transcribe the music that they play. If an experienced drummer is attempting to write a part for a song and just playing around with possible beats, they are able to save these trials for later use by recording their sessions and having them saved as sheet music to be viewable later.

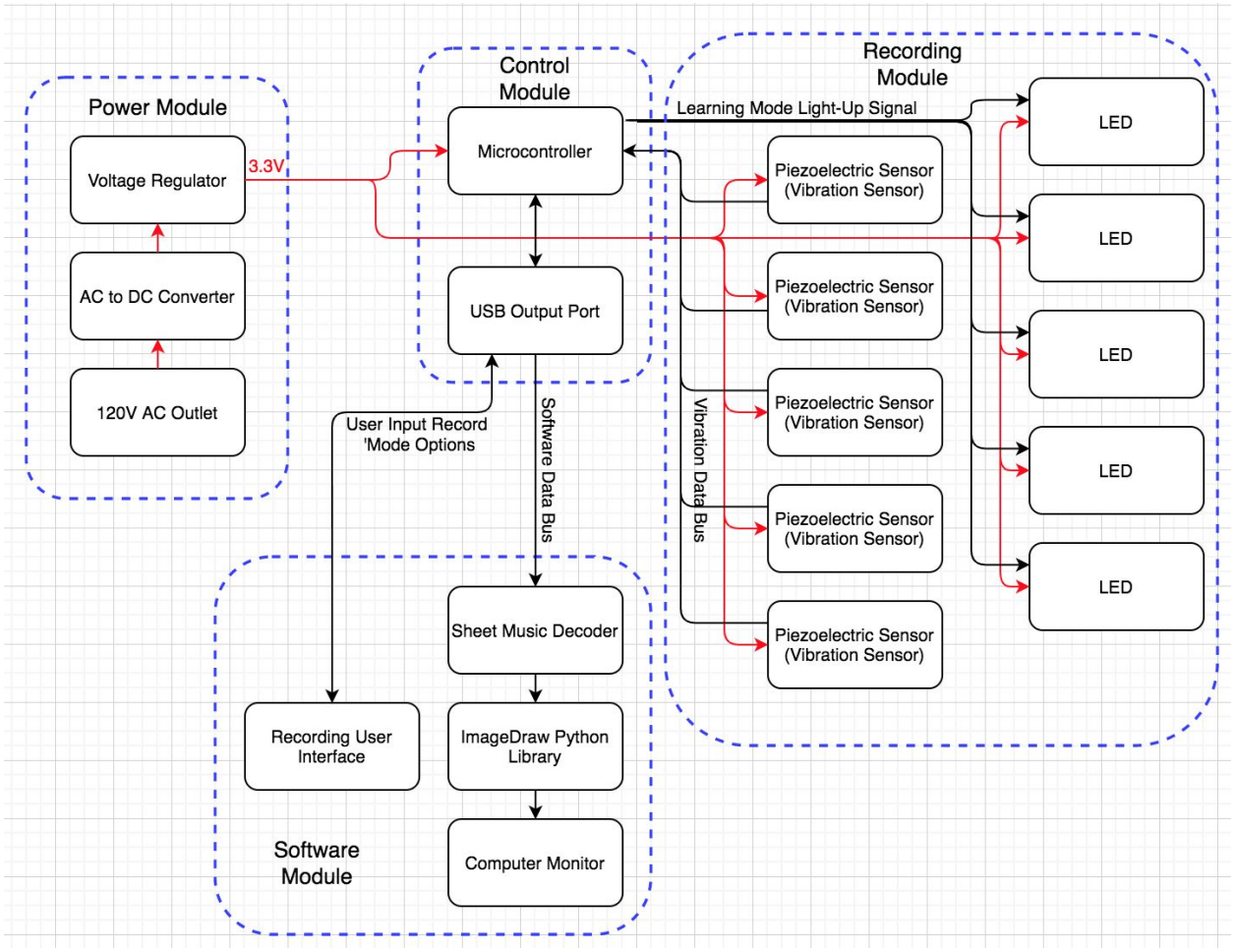
1.3/ High-Level Requirements

1. Serves as an inexpensive add on for drum set owners looking to record percussive music.
2. Able to identify and store which set piece has been hit in real time accompanied by a timestamp respective to the internal clock.
3. Provide a software interface for “Record Mode”, giving pre-record options and translating drum play into percussive notation.

1.4/ Visual Aid



1.5/ Block Diagram



2/ Second Project Implementation

2.1/ Implementation Details & Analysis

2.1.1/ Sheet Music Translator

The biggest piece that we implemented, and were capable of doing at home, was the sheet music translator which took advantage of the ImageDraw library in Python. It takes in a file written by the control module which records drum hits and their timestamps. It also takes in the tempo and which note gets the beat in order to generate a piece of sheet music for the user to save. Along with the Graphical User Interface (GUI) this software is meant to make it easier for a new musician to visualize the music they are playing and a tool that helps older musicians to record their pieces. Since the design review we have added more features which help to make the sheet music more useful to the user. The biggest feature which was suggested that we work on was that it could take in more precise timestamps that were not exact. Based on the tempo our software is able to figure out where it is most likely that the note should be placed in the sheet music. This way if a person is slightly off beat on a note it should still place the note in the correct spot.

Another addition to our software is that it takes in a third value which attempts to measure the force applied in a hit of the drum. In the design review the professor recommended that we take in more data when we record a session. Mainly the professor wanted to see a more meaningful piece of sheet music that is more than just when a note was hit. Unfortunately we did not know what values would be acceptable for a hit on the drum so we chose to use filler values for the time being which approximated what counts as a strong hit versus a weaker hit of the drum. This way we are able to do things like separate notes by marking an X versus a quarter note for the time being.
An example piece of sheet music that may be generated with our program can be seen below.

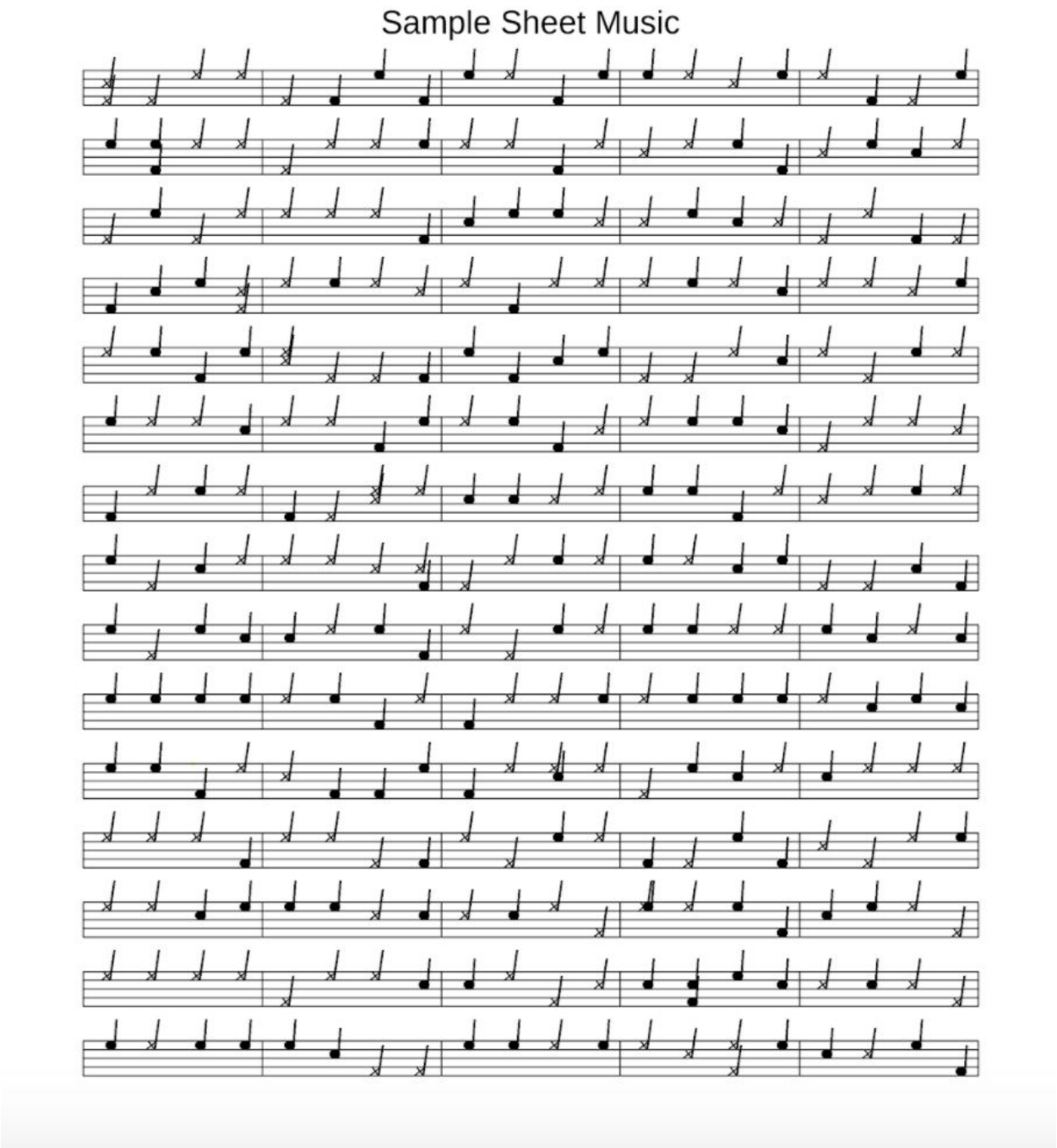


Figure 1. Software Generated Sheet Music

The code can be found in THIS REPOSITORY.

2.1.2/ Tempo Analysis

One important part of our computer UI interface is the tempo analysis feature. Tempo is a critical component of sheet music. Performers rely on a beats per minute indicator to know the speed of the music. A piece of music can totally branch away from the artist's intentions if not played at tempo intended. Our software will have the ability to detect the tempo of the piece being recorded based on notes being played. The UI will include some user pre-determined settings that our program could deduce tempo from; however, we do not want our software to rely on user input.

Our software will sort the packaged note data into a "histogram". The histogram bins would include the distance of time between two notes and the type of note. Using those two data variables, along with the time length of recording sessions, we can calculate a beats per minute value.

The following formula would be used in a calculation with quarter notes.
$BPM = 60000/x$, where x = smallest division between quarter notes

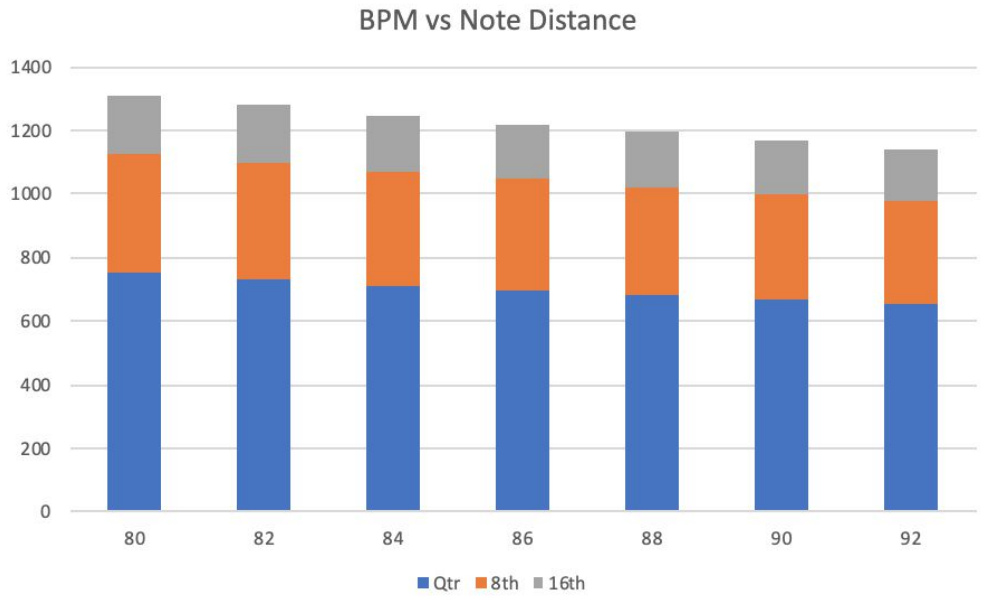
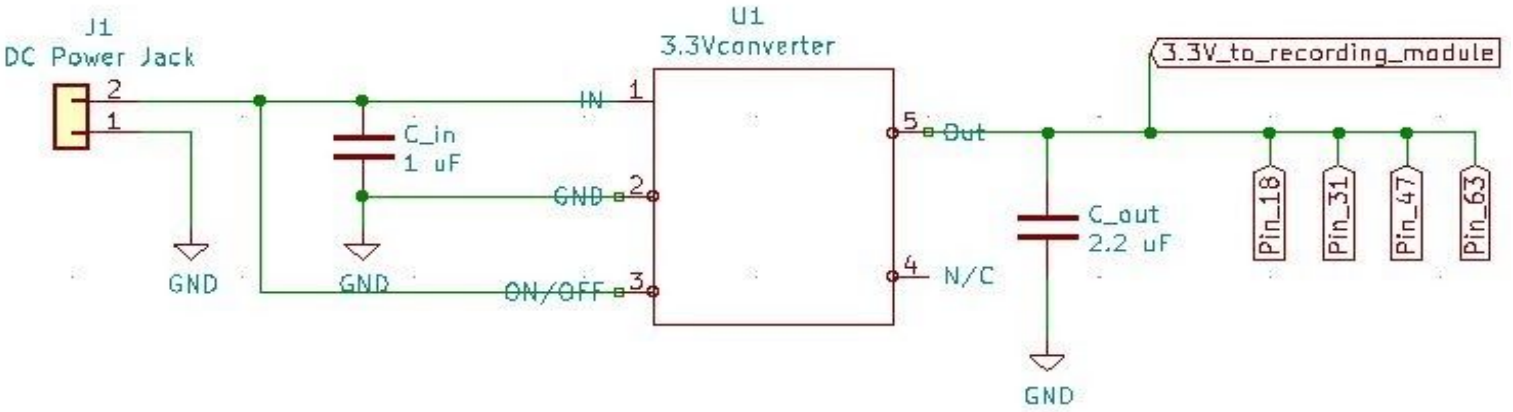


Figure 2. BPM Compared to Note Distance

The above figure compares BPM on the x-axis with the nearest note distance (milliseconds) on the y-axis. This BPM data relies on locating the lowest "note distance" bin in our virtual histogram. Our solution to achieve this is calculating the density of beat divisions (note distances) throughout the session recorded.

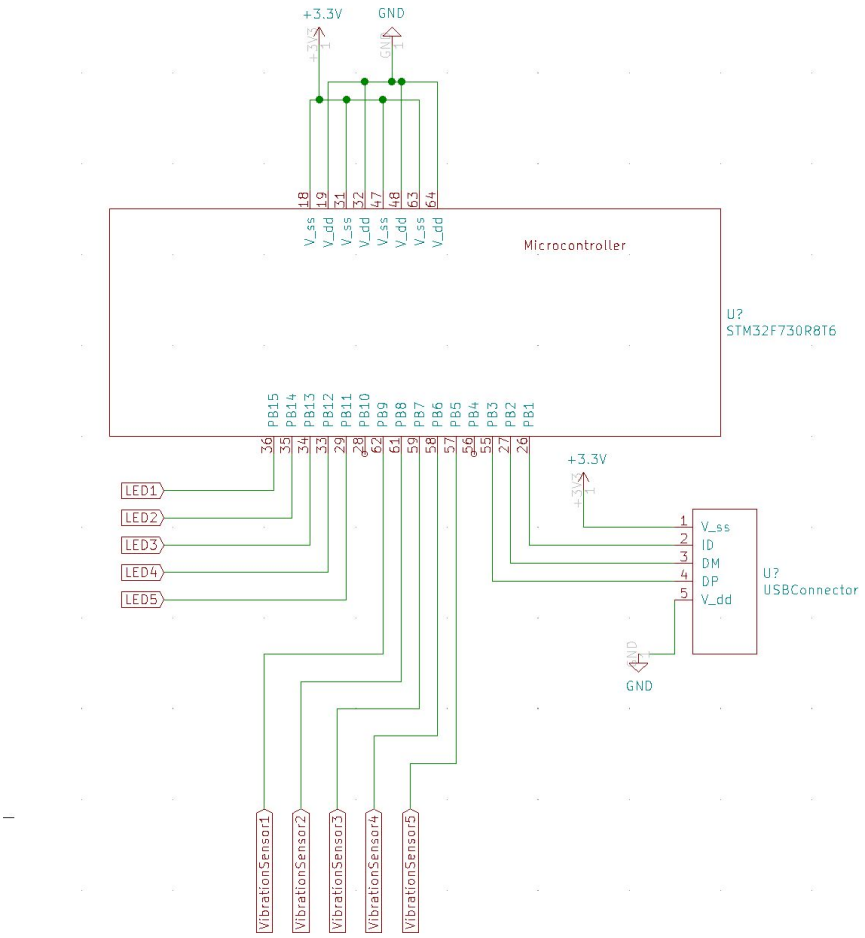
2.1.3 Power Module

The main source of power we will be using for our device will be a 120V AC wall power supply. We will utilize a simple wall power adaptor that will take the 120V AC input and then convert it to a 12V DC output. Using a 3rd party wall power adaptor makes our device a lot less complicated and less expensive to implement. The barrel jack output of the adaptor will be connected to the control module through a barrel jack port. The 12 V input will then be converted to 3.3V using a voltage regulator. The 3.3V voltage regulator requires capacitors to operate so these have been included in the module as well. The output of the 3.3V regulator will be used to power the microcontroller in the control module as well as the piezoelectric vibration sensors in the recording modules.



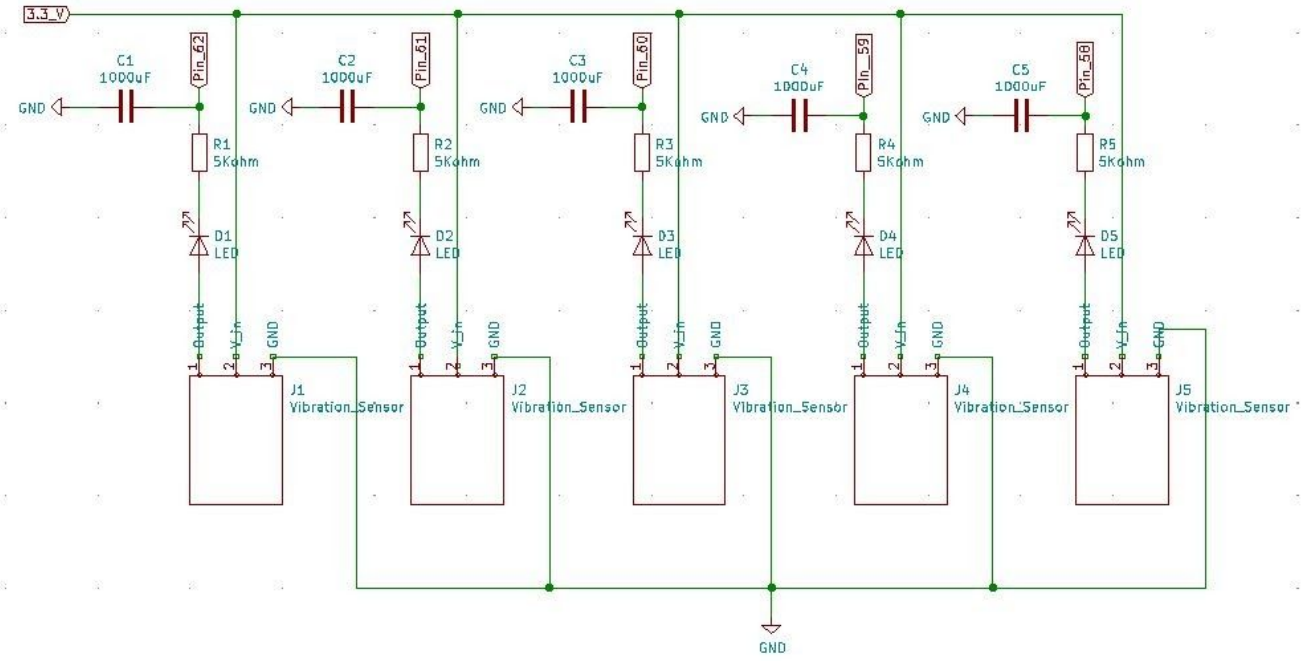
2.1.4 Control Module

The control module in our device is responsible for sending the file with time stamped drum hits to the software module and alerts the user when a hit was recorded with the use of LEDs. The recording module will send an analog signal to the microcontroller which will detect a hit based on a preset sensitivity threshold. The threshold will be set after prior testing; this is in order to avoid false positives and false negatives. It is very important that our device is able to accurately record all hits. Because there are sensors on each drum, the microcontroller will be able to save the drum and timestamp to a file which will be sent to the software module at the end of the recording session. All data to and from the software module will travel through a USB-C connector attached to the computer. When recording starts and ends, the PC will send a signal to the device. All commands sent to the device will be parsed with internal logic and the appropriate actions will take place. The only two components for this module are the microcontroller and the USB connector.



2.1.5 Recording Module

The recording modules consist of several hardware components that communicate with our microcontroller. The main component of the recording modules is a Piezoelectric Ceramic vibration sensor. Each piezoelectric vibration sensor has a tiny computational board along with a ceramic disk that will be placed flush on the drum or cymbal “mute pad” cover that will be used in our design. These mute pads are fluid with any drum shape or size, giving our product a “one size fits all” feature. The ceramic disk utilizes the Piezoelectric effect, which generates electrical charge from an outside mechanical force. [4] The mechanical force in this case is striking the drum or cymbal. The mute pads provide another functional use, they are made from very static material. This way, our sensors will not confuse a vibration from a hard strike as multiple strikes. Once our ceramic disk receives the vibration, a signal will be sent from the sensor to its respective microcontroller pin. When this signal is sent to the microcontroller, it will also illuminate an LED. LEDs will be attached to the outside of each mute pad and serve as a visual aesthetic for the player. These LEDs will light up when the drum or cymbal is struck and will notify the user when their hit has been recorded.


3.3_V
C1
1000uF
C2
1000uF
C3
1000uF
C4
1000uF
C5
1000uF
GND
Pin_62
Pin_61
Pin_60
Pin_59
Pin_58
R1
5Kohm
R2
5Kohm
R3
5Kohm
R4
5Kohm
R5
5Kohm
D1
LED
D2
LED
D3
LED
D4
LED
D5
LED
1 Output
2 V_in
3 GND
J1
Vibration_Sensor
J2
Vibration_Sensor
J3
Vibration_Sensor
J4
Vibration_Sensor
J5
Vibration_Sensor


EVANS
SOUNDOFF

2.1.4/ Bill of Materials

Part Name	Manufacturer	Part #	Quantity	Cost/part
Piezoelectric Ceramic Vibration Sensor Module	KEYESTUDIO	N/A	5	$3.99
USB Connector	CUI Devices	UJ2-MIBH-4-SMT-TR	1	$0.87
DC Power Jack	CUI Devices	PJ-059BH	1	$0.89
Assorted Resistors, Capacitors, ICs, Regulator, LED etc.	Digikey	N/A	1	$5.00
PCB	PCBway	N/A	1	$5.00
12 W Universal Input, Wall Mount Adapter	Kaga Electronics USA	62-1230-ND	1	$12.21
Microcontoller	STMicroelectronics	STM32F730R8T6	1	$4.96
5 Piece Mute Pack	Evans	N/A	1	$23.00
Low Pass Filter Chip	Digikey	LTC1062CN8#PBF	2	$7.97
Total	-	-	-	$79.85

3/ Second Project Conclusion

3.1/ Implementation Summary

In summary, the only part of our project that we were able to implement was the software module. Namely, we spent time working on a basic sheet music translator that would be as ready as possible so that if we had the hardware portion created that we could fine tune the code and finish our project. This ended up being a basic implementation which sets up the framework for future additions to it. For now it has the ability to write a piece of sheet music using quarter notes and best guesses the location based on what beat it is closest to. It currently runs on music played at 60 bpm.

3.2/ Unknowns, Uncertainties, & Testing Needed

Currently, without access to the lab, we are unable to do any implementation, testing, and refining on our hardware modules. We cannot complete these tasks because we do not have access to the parts required or the tools from the lab that would allow us to test and design our product. One big task that needs the lab and equipment to be completed is for us to test the sensitivity of the piezoelectric sensors so that we do not end up with many false positives or false negatives. We plan to complete this task by recording a wide range of drum hits with different forces and methods of hitting and implementing a threshold based on our data. After refining this threshold we could make sure that only one hit is recorded per physical hit of the drum and that any excess frequency is not registered as a hit. Another important component of this testing is our low pass filter implementation. The low pass filter (LPF) is essential in minimizing false positives and false negatives in our data. We would need the lab and multiple testing trials to determine the optimal RC constant for our LPF [6]. Furthermore, our LPF could be customized according to the average frequency of the drum it is paired with. For example, the filter for a cymbal would have a much higher cutoff frequency than the filter for a floor tom. This was each “drum head module” is optimized for the least amount of errors possible.

Another important part that we would need equipment to implement is the transfer of data through the USB connector to and from our microcontroller. To complete this task we would have to find a way to send a stream of data to and from the microcontroller so that it talks with our software implementation. For example, it is important that the graphical user interface (GUI) is able to tell the microcontroller when to begin and end a recording session and for our software to get the proper recording file from the microcontroller.

In addition to working with the transmission of data to and from the microcontroller, another key part of our project relies upon properly programming the microcontroller. Because we do not have all of the supplies to test and work with the STM32 that we picked out, we are unable to program and debug it. If we were to have it, we would implement a program that polls our piezoelectric sensors and records any hits during the recording session. One of the most important pieces of having and testing this is that we can have the microcontroller’s software use a threshold which

would be refined through testing. This would allow us to find out what threshold is required to avoid false positives and false negatives when a drum strike is recorded.

The last thing that we would need lab access to complete is the power module where we would need to test the ability to power our device using the 120V AC outlet. It is important that we do this testing and implementation in the lab as outlined by the dangers of working with such high voltages in general.

3.3/ Ethics & Safety

In order to ethically provide our device to future customers we need to make sure that we follow the guidelines by the IEEE code of ethics. First and foremost, we want to provide safety to our customers when they use our device and this follows in line with the first guideline in the code of ethics, “to hold paramount the safety, health, and welfare of the public, to strive to comply with ethical design and sustainable development practices, and to disclose promptly factors that might endanger the public or the environment” [1]. The main safety concern is that our device takes power from the 120V AC wall outlet. We must make sure that it cannot harm the user and that we regulate this voltage to only what is necessary and safe to use for our parts. We as developers must also make sure to be safe when testing this device in the lab to avoid any injuries due to this high voltage.

The next point of ethics comes from the third guideline, “to be honest and realistic in stating claims or estimates based on available data” [1]. We must be sure to be honest with how well our product is, but more importantly we should provide a well functioning product. It would be highly unethical to create a solution that partially functions and provides the user a lot of hindrances in actually utilizing its functionality.

Another safety concern is that the wires that run from the control module to the sensors are all safe to use. It is important to make sure that there is no electrical current exposed to the users and that our device is always safe to use.

The first issue we plan to mitigate the risk for will be the use of the 120 V AC wall outlet. We plan on using a predesigned 12 V wall adaptor from another company. This will insure that the customer will not have any contact with the 120 volts being supplied, when using our product and as it will be plugged into the wall like any other device. By using a 3rd party adapter that has already been tested for safety, we can avoid issues of directly handling 120 V on our device. Our product will have a warnings and safety section in its instruction manual to warn users the danger of using devices connected to wall outlets. This will include instructions to prevent electrocution such as to avoid contact with water and touching an exposed plug. Another part of our instruction manual will describe our product's functionality and how to use it safely without injuring yourself in any way. To mitigate the risk of exposure to electrical current, all parts of our design will have protective plastic covering around each module and wire that is used.

3.4/ Project Improvements

3.4.1/ Learning Mode

One improvement for our project that was also included in our proposal was the learning mode aspect of the project. The learning mode would be a software application that taught new percussionists how to play the drum set. During learning mode the user would choose a piece of music they would like to learn. This music will be displayed on the graphic user interface as sheet music. The user will then choose what speed or what bpm they want to learn the music at. When the user is ready to begin they will choose to run the program on the user interface and a metronome will start to play similar to record mode. A cursor will follow along with the notes on the sheet music and will show the user where they are at in the song if they get lost. When it is time to hit a specific drum or symbol, a signal will be sent from the microcontroller that will turn on the LED for that specific drum or symbol. The user's drum hit would be recorded in the same way that it is recorded during the record mode. After the song has been played, the recorded sheet music will be compared to the actual sheet music for the song that was played. The program will then provide feedback to the user for which parts that were not played correctly. To implement the record mode we would have to add a significant amount of software to our current design along with potentially adding components to the recording modules so that the LED can have signals sent to it.

3.4.2/ GUI Settings By Drum Input

A really important step in the evolution of this product would be using the piezoelectric sensors as means of user input for options in the GUI. More simply, using drum strikes to determine recording options. This would make our product extremely intuitive and open new feature opportunities. For example, a user can set the desired recording tempo by striking the drum four times instead of directly typing in the GUI. Before the recording session our program could be polling the "tempo set" drum. Once the user makes his first strike, our internal clock starts and ends with strike number four. The average time between these quarter notes can be calculated with division by four, then using our existing tempo analysis capabilities, a BPM can be determined. Another intuitive implementation would be using left and right drum strikes to "scroll" through information or "click" on the GUI. This way, our device turns your drumset into a controller or sorts, allowing the user to minimize time spent on the computer. This makes our product more user and musically friendly.

3.4.3/ Hit Strength Detection

One of the improvements that was suggested in the design review is that we take in more data from drum hits. This includes hit strength as well as a more precise form of hit detection that would allow us to know if the user hit the drum near the center or near the edge. By taking even just these two things into account we can provide the user with a more informative piece of generated sheet music that includes more about their musical style. One way that we have thought about being able to accomplish this is by getting more sensors attached to each drum so that we could record more meaningful data to process.

Another way that we can introduce a more informative piece of sheet music is to introduce dynamics indications. We can attempt to add the dynamics by making educated guesses based on how strong the user's hits are. For example, if a user's hits gradually get stronger it may be that they are following a crescendo. We can try to judge their dynamics by setting a baseline threshold for a mezzo-forte (which is a medium loud indication) and judge the changes in dynamics by the changes in force that the drummer exerts in his hits.

4/ First Project Progress

4.1 Progress Made After Design Review

4.1.1 Charging Capability for Internal Battery

During the first design review it was recommended to us that we should find a way to increase the lifetime of our internal battery. To accomplish this we looked into ways to charge our 3.7 V Lithium ion battery. One solution we found for this was the MCP73831 IC. This IC has a programmable charge current and automatic power down for charging different types of batteries. The inclusion of this IC into the design would have allowed us to charge our internal battery when the device was plugged into a USB power source. This IC and respective components were added to our power module. This improvement fixed the issue of our device only being usable for a few weeks or months. Another improvement that could have been made in the future would have been a visible LED to tell if the battery needed to be charged or not. This improvement would have required additional components and firmware for the microcontroller.

4.1.2 Sleep Mode

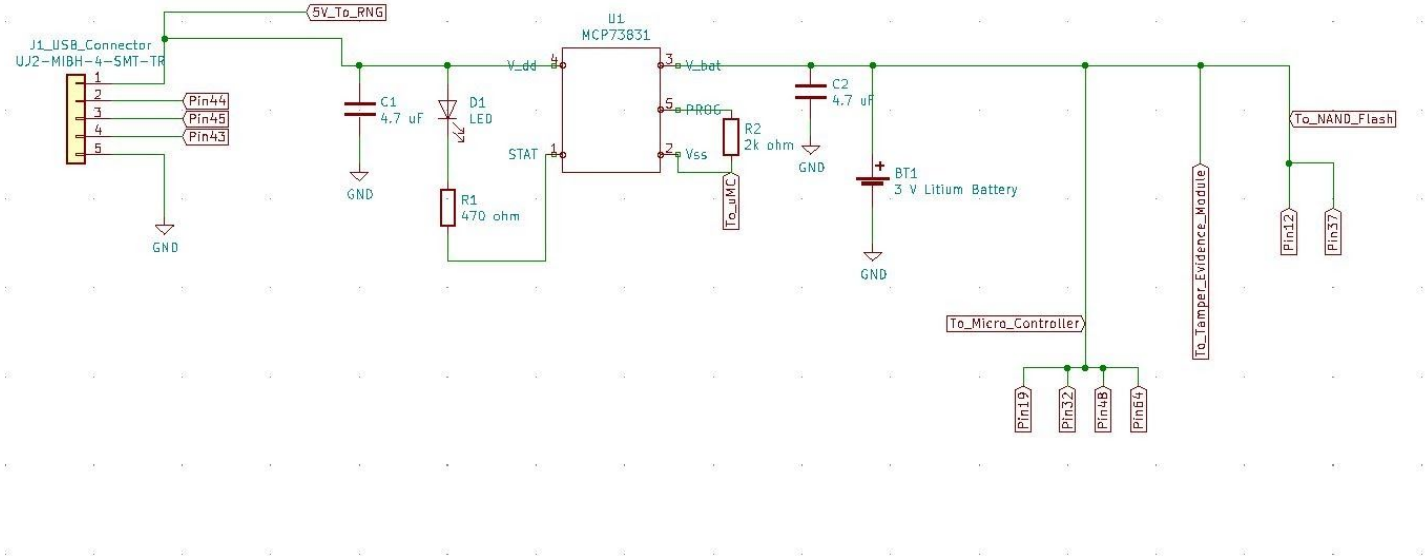
After the design review we realized that battery life was going to be a difficulty in our product. One way we talked about solving this in addition to the battery charging feature was to implement sleep mode on the microcontroller while the device is not in use. Without the lab we were unable to fully implement this in our blue pill dev kit, however I read through the data sheets and found which bits in the internal registers had to be flipped on or off in order to put the microcontroller into sleep mode.

We began basic implementations of this in Arduino which polled a specific pin on the microcontroller which was 1 when the device was powered by the USB connector and 0 otherwise. With this being the case we could then set the internal bits to the desired ones and refrain from using up more power than needed to maintain the tamper evidence module.

4.1.3 AES Implementation

Prior to starting the second project for this class we had finished a basic C implementation of AES that would be used by our device. Following some guides online and an implementation on the FPGA from a different class, we built the base files to be used by the microcontroller for AES-128 encryption and decryption. The files for that can be found in the “445 AES” directory in the same github repository as above.

4.2 Supporting Material



This is an updated schematic for the power module that includes the Li-ion charging IC.

5/ References

[1] "IEEE Code of Ethics," *IEEE*. [Online]. Available: https://www.ieee.org/about/corporate/governance/p7-8.html. [Accessed: 03-Apr-2020].

[2] "Detect Bass Drum Hit," *Forum.arduino.cc*. 2013. [online] Available: <https://forum.arduino.cc/index.php?topic=185207.0> [Accessed 14 April 2020].

[3] E. F. Y. Team, "What Is A Piezoelectric Sensor?," *Electronics For You*, 06-Aug-2019. [Online]. Available: https://www.electronicsforu.com/resources/learn-electronics/piezoelectric-sensor-basics. [Accessed: 15-Apr-2020].

[4] "Piezoelectric Pressure Sensors: The Design Engineer's Guide: Avnet Abacus," *Piezoelectric Pressure Sensors | The Design Engineer's Guide | Avnet Abacus*. [Online]. Available: https://www.avnet.com/wps/portal/abacus/solutions/technologies/sensors/pressure-sensors/core-technologies/piezoelectric/. [Accessed: 17-Apr-2020].

[5] "Calculating Beat Divisions Using Tempo and Time," *Harmony Central*. [Online]. Available: https://www.harmonycentral.com/articles/uncategorized/calculating-beat-divisions-using-tempo-and-time-r168/?tab=comments. [Accessed: 18-Apr-2020].

[6] "Low Pass Filter - Explained," *Learning About Electronics*. [Online]. Available: http://www.learningaboutelectronics.com/Articles/Low-pass-filter.php. [Accessed: 06-May-2020].