

Real-Time Sound Visualization

ECE 445 Design Document

Qian Chen qchen46@illinois.edu

Lin Le linle2@illinois.edu

Xinyue Yu xyu69@illinois.edu

TEAM 43

TA: Dongwei Shi

2/22/2018

Contents

1. Introduction.....	3
1.1 Objective	3
1.2 Background	3
1.3 High-level requirement.....	4
2. Design	4
2.1 Function overview.....	4
2.2 Block Design and verification	5
2.2.1 Sound Input, Video Output	5
2.2.2 Controller.....	7
2.2.3 Other.....	9
2.2.4 Power Supply.....	9
2.3 Circuit.....	11
2.3.1 Circuit Schematics.....	11
2.3.2 Internal circuit and block diagram of FPGA.....	14
2.4 Algorithm.....	16
2.4.1 D/A conversion and FFT.....	16
2.4.2 Data transmission between FPGA and MCU.....	17
2.5 Tolerance Analysis.....	18
2.5.1 Cut-off frequency.....	18
2.5.2 Accuracy of pitch detection.....	19
3. Cost and schedule.....	19
3.1 Cost.....	20
3.1.1 Labor.....	20
3.1.2 Parts.....	20
3.2 Schedule.....	21
4. Ethics and Safety.....	22
5. Reference	23

I. Introduction

1.1 Objective

Have you ever experienced the situation that you want to immediately get the transcription when you listen to piece of music that you are enthusiastic about? The solution that we design for this problem is to create a small and affordable device that listens to a musical instrument or vocal sound and recognizes the notes played with LED effect. These notes can then be sent to a synthesizer in the common MIDI format. This allows people who's not skilled at music to record the music that they want to store as their compositions into musical notes or for those people who do not have music score but want to play piece of music that they heard.

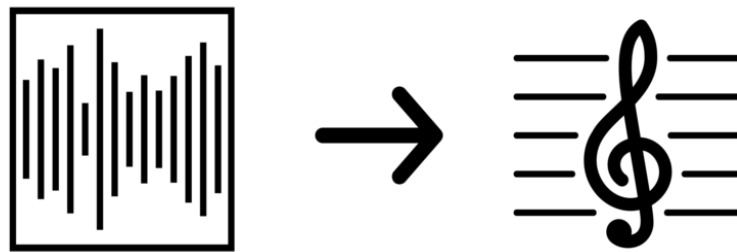


Figure 1. Real-time Sound Visualization

1.2 Background

Nowadays, there are thousands of mobile phone application have the function of pitch detection to recognize pitch like Vocal Pitch Monitor in the Google Play, Pitch Tuner or Tenuto in the Apple Store [1]. But most of pitch detection application do not have the transcription option to show on and few that has the feature is really expensive to purchase in the apple store. Hence it is interesting to have a hardware version to combine all the fantastic features together and make it portable and affordable for musicians or music lovers with a cheap price.

This project creates a small embedded monophonic music transcription system that can transcribe one note at a time when there is one sound. The next step is to store the melody into designed memory chip and mimic piano sound for replaying. There will be LED effect accompanied with piano sound when it is in replay mode. To be specific, a pitch detector in hardware will be used to detect sound in real time at 44k sampling rate. A FFT analysis will be used inside our microcontroller. The detected pitch will be stored into memory for further replaying and displaying. Once the musical notes has been detected, it will automatically display on a VGA display. The musical notes will flow to the right side when one music note is displayed.

1.3 High-level Requirement

- It should detect the correct pitch for notes produced by vocal or other instruments, and display the correct note on display.
- There should be no noticeable delay between incoming audio signal and digital output. The delay should be no larger than 250 ms ($\frac{1}{4}$ second).
- The MCU drives a series of LEDs as a volume unit (VU) meter, which gives the intensity of input sound, like those on the recorder or CD player.

II. Design

2.1.1 function overview

The ultimate goal is to detect which note is playing. To achieve the goal of performing pitch detection, a frequency spectrum analyzation is mandatory. The method is to perform Fast Fourier Transform(FFT) on the sound data collected. To do FFT, first a microphone is connected to the MCU. The microphone translates sound energy to electric energy. Then MCU sample data and digitize all the data and store in memory. Therefore, an ARM based MCU is chosen to do all the calculation. The chosen MCU, has built-in DSP module and capable of utilizing DSP module with DSP library provided by supplier.

All the component will be mounted on a PCB (printed circuit board) except display and light effect LEDs. The MCU will output the detected pitch, then the FPGA will read the data from MCU pins and decode the data and decide what should be displayed on the VGA display. The

FPGA is also responsible to control light effect LEDs. Which works as a volume unit meter; therefore, the use can easily notice if the volume of input sound is too low or too high.

2.1.2 Block Diagram

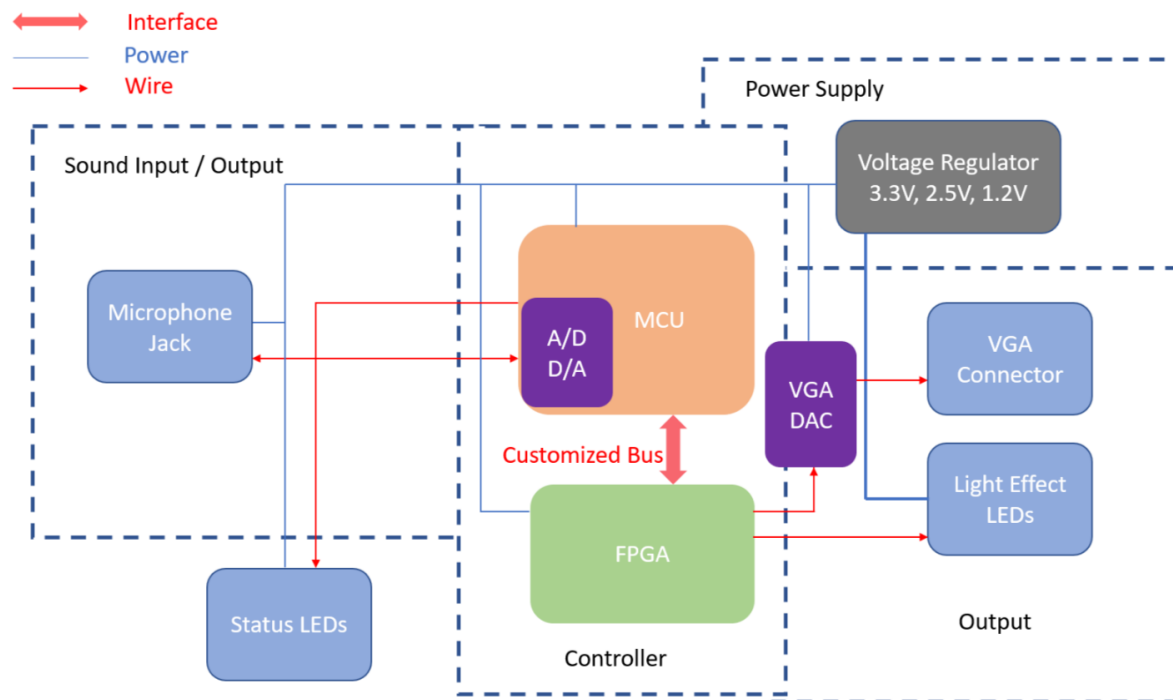


Figure 2. Block Diagram

2.2 Block Design

2.2.1 Sound Input and Output, Video Output

Microphone

The microphone we be connected to the MCU with wires. The microphone have to be sensitive for the A/D module on MCU to read the correct signal. The A/D module in MCU takes voltage

between 0v to 3.3v; therefore, the output of the microphone module must be between 0 to 3.3v.

Requirement	Verification
<ol style="list-style-type: none"> 1. Microphone module outputs voltage between 0 to 3.3v. 2. Microphone is sensitive to input sound. The microphone must have a output for inputs sound of intensity 40dB. The output of microphone must have a signal to noise ratio of at least 10db. 	<ol style="list-style-type: none"> 1. Connect the output of microphone module directly to the oscilloscope, read the output voltage, and ensure is output is between 0 to 3.3v. 2. Connect the microphone to MCU though the GPIO. Use MCU to read data from microphone. <ol style="list-style-type: none"> a. Record without any input sound, record data as noise. b. Record input sound, record data as noise plus signal. The input sound is at 50 db. Calculate the SNR. The SNR should be at least 10 db.

Table 1.

Status LEDs

The status LEDs is controlled by the Microcontroller Unit and will display to the user if the MCU is working properly and whether the microphone and speaker is on or off.

Requirement	Verification
The LED has noticeable light at input voltage 3.3v. Which is provided FPGA and MCU. One of the LED should be red, one of the led should be green.	Assemble the FPGA on PCB, connect all the required capacitors and resistors. Hard code FPGA, set the output GPIO pins to logic 1, to control LED blinking.

Table 2.

VGA port

The VGA port is driven by a VGA DAC. The VGA DAC used is ADV7123 by ANALOG DEVICES. The DAC connect to FPGA by GPIO pins of FPGA. Then the pinout will be assigned through IDE. FPGA controls what to display on the VGA display. The MCU is dedicated on performing FFT algorithm, therefore, FPGA is responsible to display result on a VGA display.

Requirement	Verification
-------------	--------------

Video signal should output with VGA at resolution 640 * 480.	Assemble the FPGA on PCB, connect all the required capacitors and resistors. Connect the VGA DAC to FPGA. Hardcoding testing image in to FPGA, test the output of VGA port. Connect a display through VGA port. The display should have test image on it.
--	---

Table 3.

2.2.2 Controller MCU

The project uses a STM32F207 MCU [2], because we need a powerful MCU to perform FFT in real-time. The current MCU is ARM Cortex M3 based, and capable of doing DSP instruction. The MCU is also responsible to do A/D conversion, the MCU has build-in A/D and D/A convection module. The MCU computation power should be sufficient. Based on our research an even a smaller M2 based MCU can perform FFT.

MCU will collect data at sampling rate 44.1 kHz from A/D port and write data into memory and perform FFT. Before FFT decimation is required to reduce the amount of data. As learned from digital signal processing, decimation will cause aliasing. Thus, a Low-Pass-Filter is needed before performing FFT. A simple averaging should be sufficient. The melody store in memory of MCU will be played back through the build in D/A module.

Requirement	Verification
<ol style="list-style-type: none"> 1. Sampling sound data with A/D module at sampling frequency 44.1kHz. 2. Perform pitch detection, the delay of result is less than 250 ms, and the rate of sample possessed is more than 10k sample/sec. 3. Playback stored sound data with a speaker. 	<ol style="list-style-type: none"> 1. Assemble the MCU on PCB, connect all the required capacitors and resistors. Connect the Wave generator to MCU through audio jack. MCU read data with its A/D module. Set the wave generator at 22kHz. Output the data through serial port, performing FFT with matlab. Ensure the highest peak is at 22kHz. 2. Connect the GPIO pins of MCU to

	<p>wave generator. Read input data from wave generator. Perform the pitch detection algorithm on the collected data. Communicate with MCU by serial port to ensure the result of pitch detection is correct. Record the time takes to perform pitch detection with build in time lib. Ensure the time is less than 250 millisecond.</p> <p>3. Record music with microphone. Store the data on memory. Playback the stored music. Ensure the music can be recognized by human.</p>
--	---

Table 4.

FPGA

One of important module in our control unit is FPGA. One of the important roles FPGA played in our system is to work as a VGA display controller. The MCU is dedicated on performing FFT algorithm, therefore, FPGA is responsible to display result on a desktop monitor though VGA port. Moreover, FPGA will also responsible to control LEDs. The LEDs display the input sound intensity as a VU meter. The FPGA considered is Altera Cyclone III EP3CE144E22.

Requirement	Verification
<ol style="list-style-type: none"> 1. Output the correct video and display it on VGA display. 2. Read the data from MCU, decode the information. 3. Blinks all the light effect LEDs correctly. 	<ol style="list-style-type: none"> 1. <ol style="list-style-type: none"> a. Assemble the FPGA on PCB, connect all the required capacitors and resistors. The FPGA require its dedicated Clock input. Check and make sure the FPGA is running at a desired clock speed by a oscilloscope b. Hard code a test image by quartus II. Upload the synthesised file. Ensure the output VGA signal is 640 * 480 at 60Hz frame per second

	<ol style="list-style-type: none"> 2. Connect the GPIO pins of FPGA to GPIO pins of MCU. Read data to the GPIO pins by MCU. Read data from pins by FPGA. Then display the data through LEDs. Ensure the LED connected to pins of FPGA can be controlled with MCU through FPGA. 3. The LEDs are not directly driven by FPGA, in that case the power consumption and current is too big for FPGA to handle. The pin of FPGA is connected to a LED driver, then the LED driver drives LED. Ensure the LEDs blinks, and the intensity of LEDs are easily recognized by human.
--	---

Table 5.

2.2.3 Other IDE

Eclipse with GCC, STM32 Cube as an IDE for STM32 MCU. Write program with Eclipse, then compile the code with GCC and STM32 Cube. Upload the compiled program.

Quartus II for FPGA. Quartus II is the official IDE realised by altera. The simulation of FPGA will be done by modelsim.

Requirement	Verification
<ol style="list-style-type: none"> 1. Compiles code in C for MCU. 2. Synthesis systemverilog for FPGA. 	<ol style="list-style-type: none"> 1. Upload compiled program to STM32, communicate through serial port. Make sure the status LED of STM32 blinks. 2. Upload synthesised file to FPGA, control LED and VGA. Read data from STM32, make sure the status LED of FPGA blinks.

Table 6.

2.2.4 Power Supply

Since there are a lot of component, a robust power supply circuit will be implement for this project. Power supply consists of several buck converter DC-DC voltage regulators, which will provide different voltage needed by different component. All the power supplies take 5 volt DC as input. The LEDs are going to be power-hungry than the other component. The power supply circuit is separated from the rest. FPGA has a Maximum input voltage of 3.5V. MCU requires 1.7 to 3.5V as power input. The voltage regulator used is LP3964 by texas instrument, the voltage regulator is configabale, capable of outputting voltage between 1.2v to 5v

Requirement	Verification
<ol style="list-style-type: none">1. The power supply need to have output of 1.2v +/- 5%. 3.3V +/- 5%, 5V +/- 5%. The 1.2 volt power supply has the maximum output current at 0 to 500 mA. The 2.5v has a maximum current output of 1A. The 3.3v has a current output up to 800 mA. The 2.5v power supply has a maximum current output of 800mA.2. Maintain the temperature under 100C. The maximum operating temperature for voltage regulator.	<ol style="list-style-type: none">1. Mount all components of power supply to PCB. Connect all the required resistors and capacitors. Making sure the output reading is correct. Connect the output on the PCB to resistors. Read voltage across output and output current with multimeter. Ensure the power supplies meat all the requirements.2. Using infrared thermometer to read temperature. The temperature underload should be under 100 C. Otherwise add heatsink to the voltage regulator.

Table 7.

2.3.1 Circuit Schematics

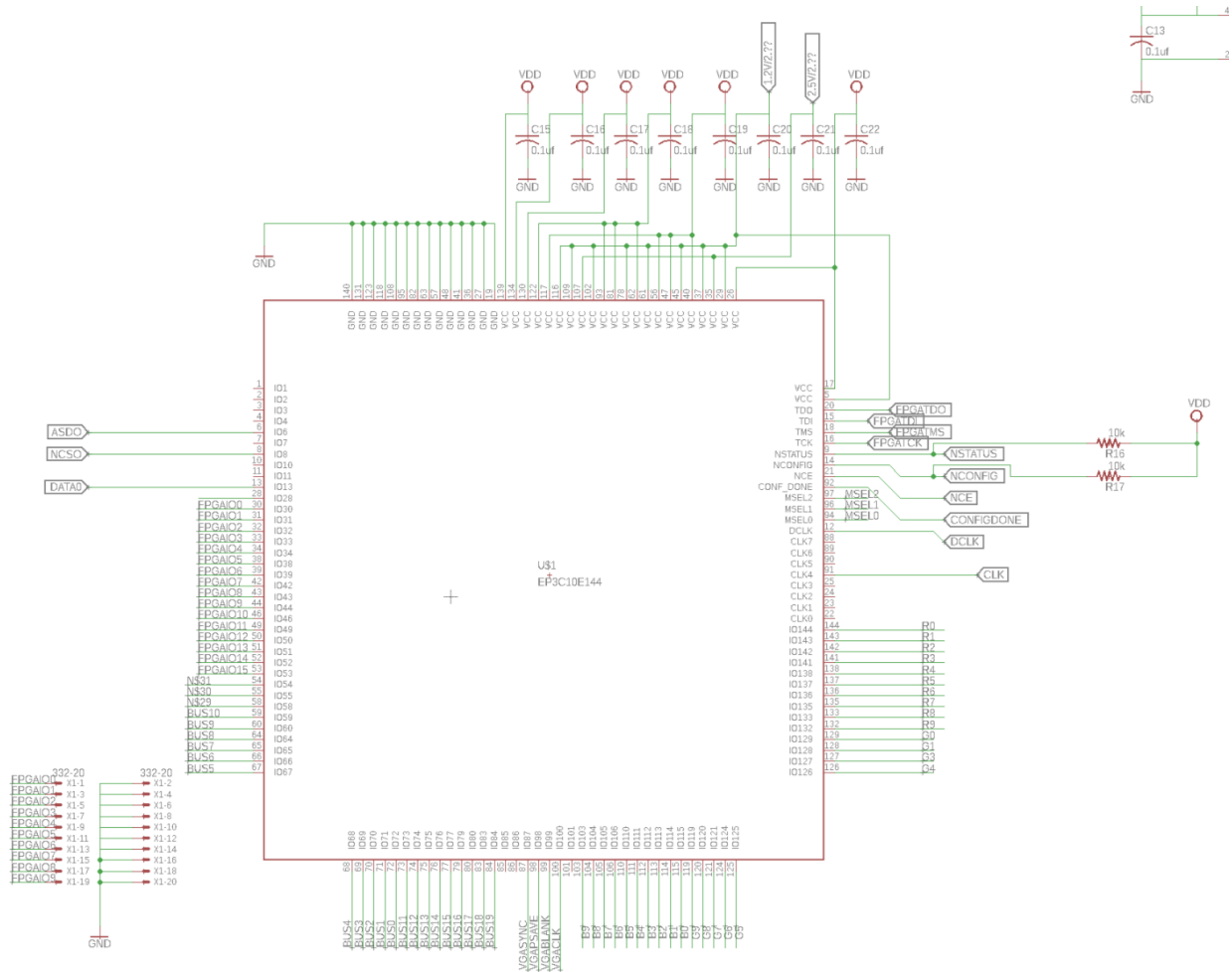


Figure 4. Schematic of FPGA

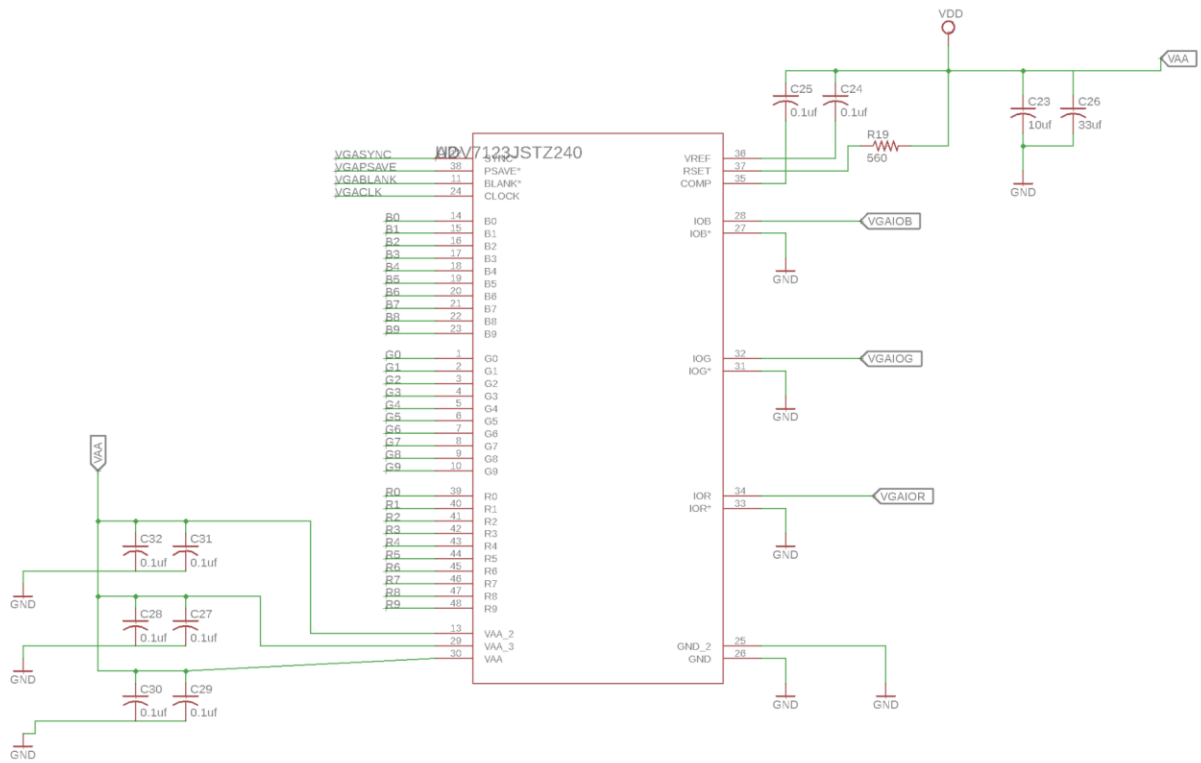


Figure 5. DAC for VGA video output

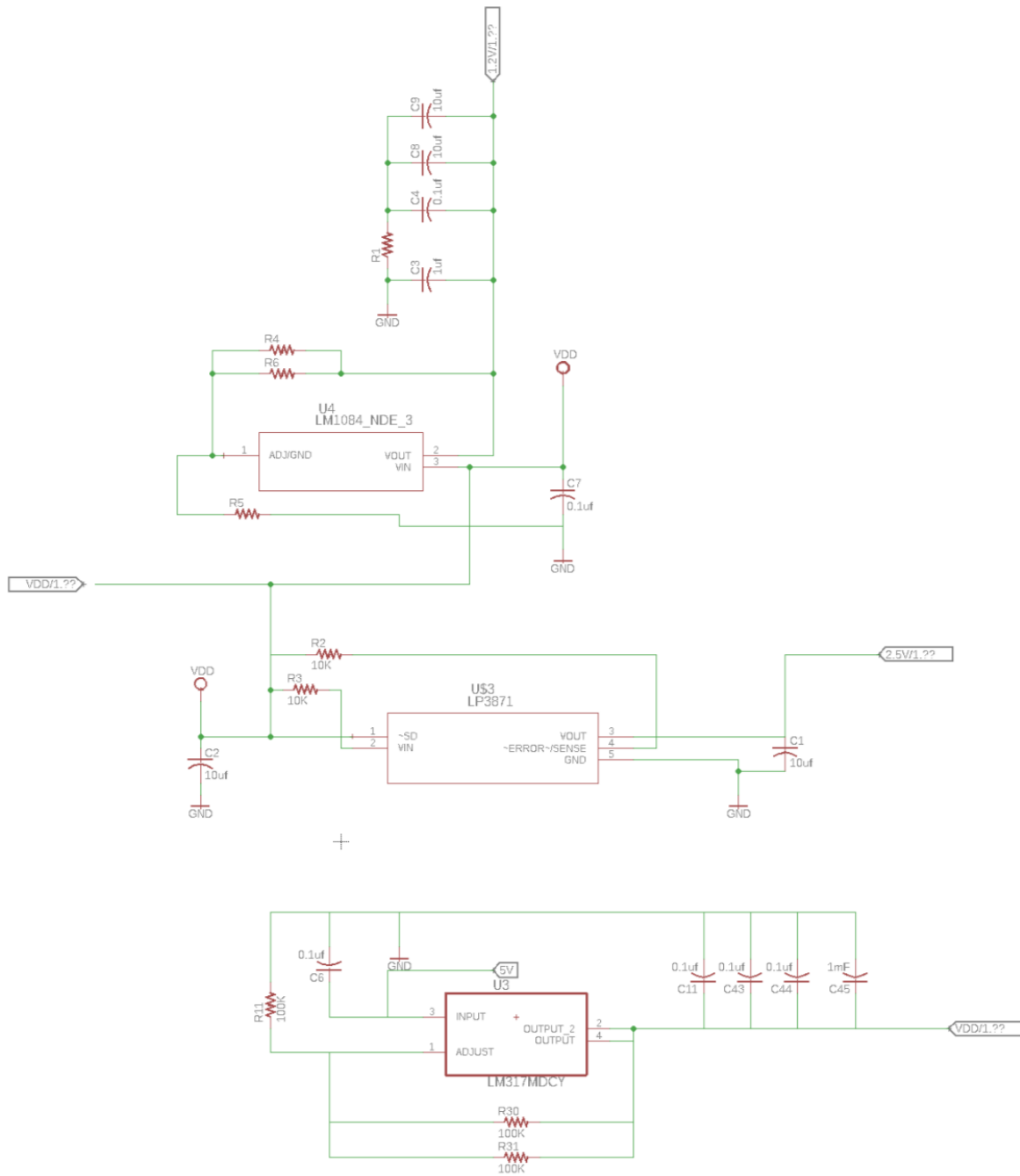


Figure 6. Three voltage regulators. One is 1.2 volt, one is 2.5 volt, one is 3.3 volt.

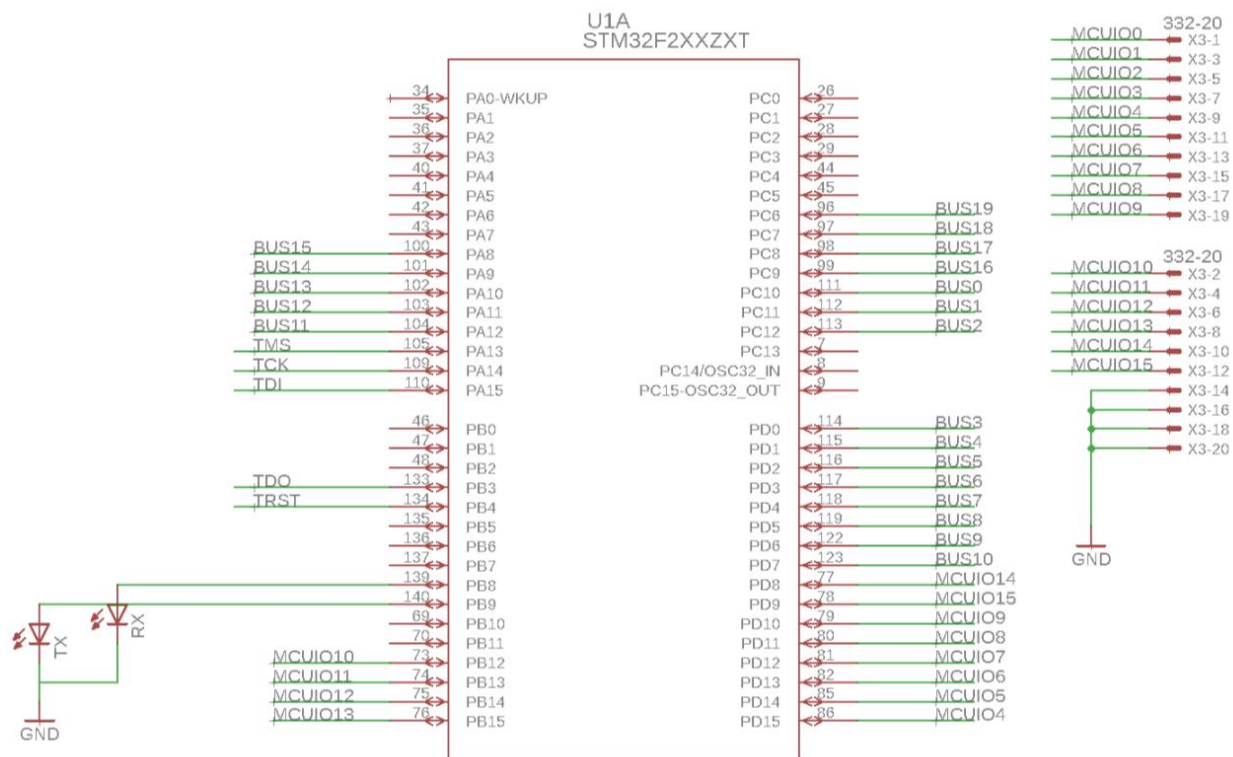


Figure 7. Bank A of MCU, the microphone is connect to the ADC port on MCU. The J-link for uploading is connected to MCU bank A. The BUS between FPGA and MCU is connected to bank A.

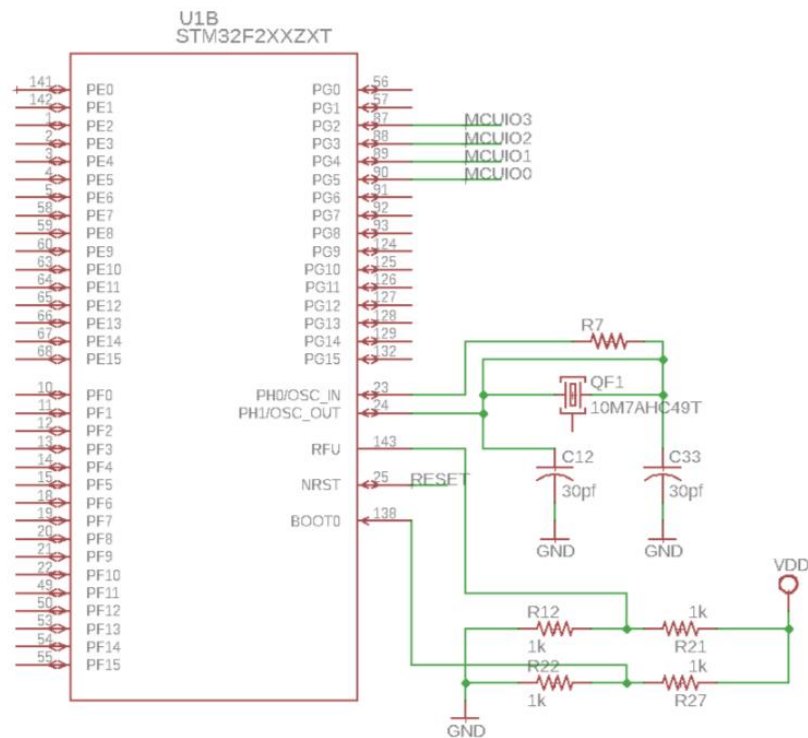


Figure 8. The Bank B of MCU.

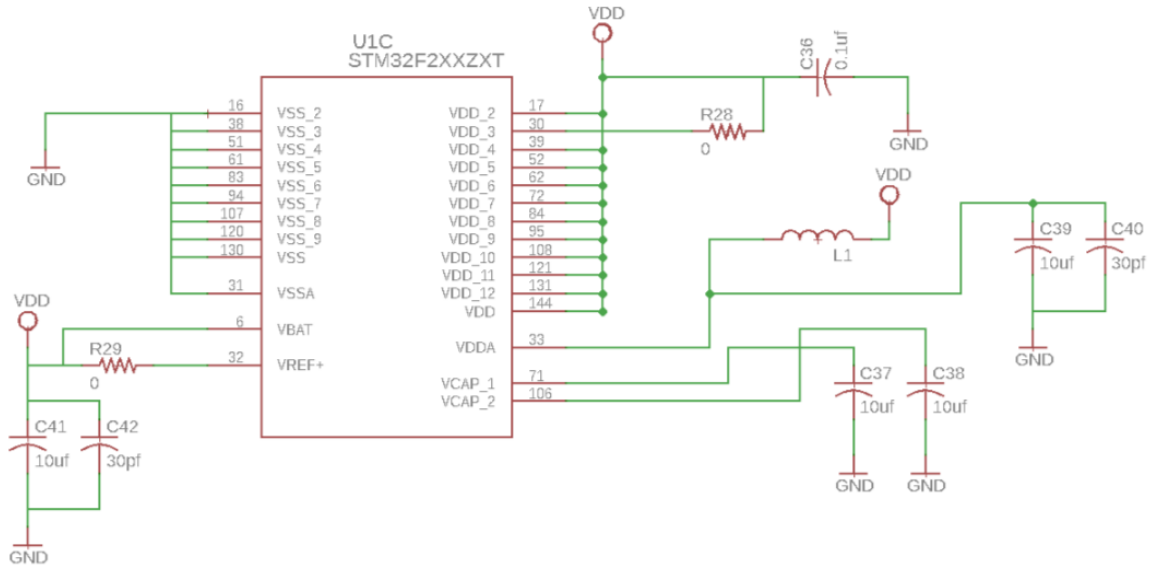


Figure 9. Back C of MCU

2.3.2 Internal circuit and block diagram of FPGA

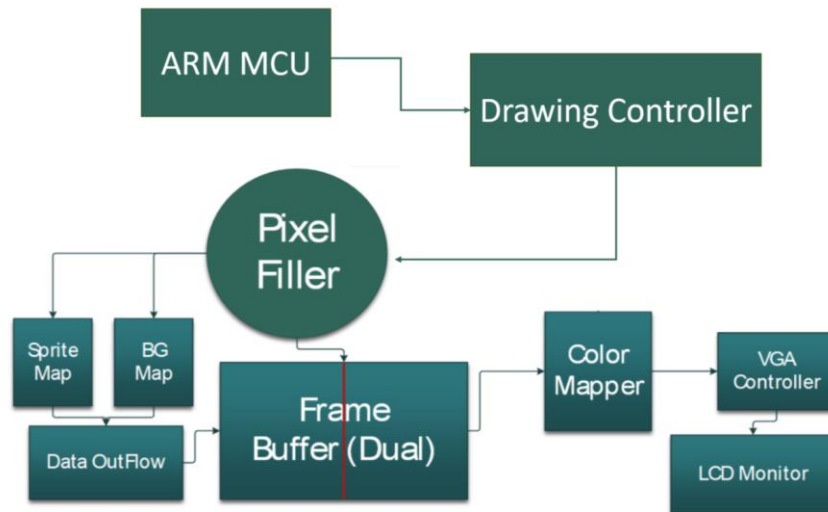


Figure 10. Block diagram of note drawing module in side FPGA. The controller decode the pitch code sent by MCU, and draw the corresponding node on screen.

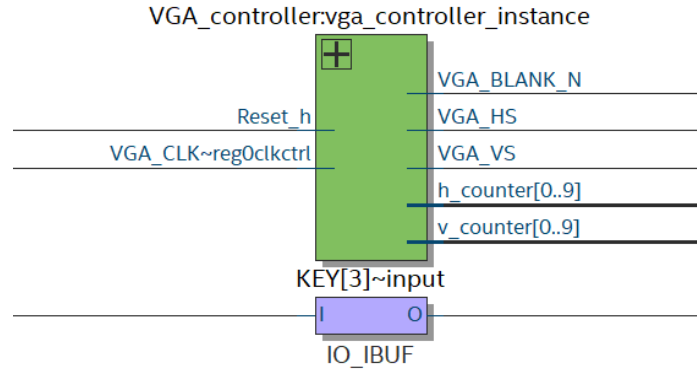


Figure 11. This module aims to control the timing of the VGA port, in which both the monitor vertical and horizontal synchronization signals are generated. Moreover, the 25 MHz VGA_Clk signal is used as the input to this module, which aims to provide a timing standard for VGA output signals.

2.4 Algorithm

The software information in the project divides into two parts. The D/A and A/D conversion (Fig.12) below handle with the pitch detection inside the microcontroller unit. The algorithm will be briefly introduced in the following module. The Data Transmission between FPGA and Display part (Fig.13) deal with displaying final result as we expected like a flowing transcription.

2.4.1 D/A Conversion and FFT

The algorithm for pitch detection on MCU is shown below. MCU collects data from microphone and perform pre-processing and FFT. The result of FFT will be stored in memory. The highest peak in frequency domain will be considered as the fundamental frequency of the input sound.

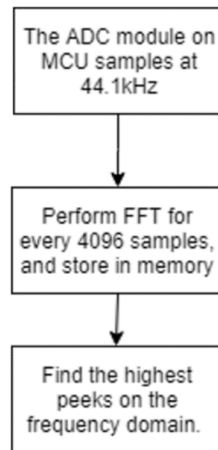


Figure 12. A/D D/A conversion and FFT algorithm

2.4.2 Data Transmission between FPGA to MCU

The algorithm for data transmission between FPGA and MCU is shown below. Once the FPGA receives the signal from MCU, it will show on the display immediately and takes corresponding effect as we expect on display and LED.

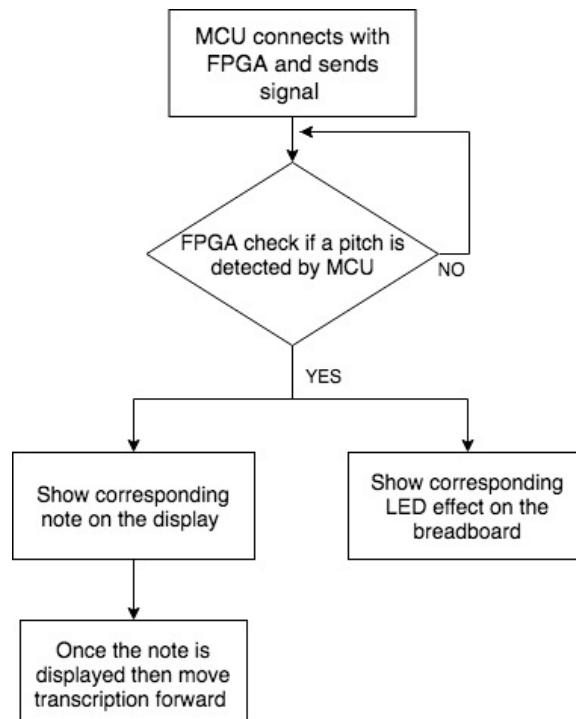


Figure 13. Data Transmission between FPGA and Display Algorithm

2.5 Tolerance Analysis

2.5.1 Cut-off frequency

The frequency response of filter design is a very important tolerance in this project. Before performing the low pass filtering, attenuation of the stop band should be high enough. Meanwhile, the cutoff frequency cannot be too low or too high, otherwise either some information is lost or induced aliasing will make all the data unacceptable.

Therefore, we must find out the preferred cutoff frequency based on the current model. In the current plan, the data collected by A/D converter is sampled at 44kHz, which should be sufficient. Most common instruments do not have a first order harmonic with a frequency exceeds 22kHz. The higher order harmonics are usually not strong. After sampling, the data when through a decimation at down sampling rate 4:1, i.e. Only one of four data are kept, others are discarded. In this case, $U = 4$

$$x_u(m) = \begin{cases} x(n) & m = Un \\ 0 & else \end{cases}$$

It is clear that component in frequency domain with frequency $|\omega| > \omega_0/8 = 5.5rad$ will cause aliasing. If we want to use a averaging filter as a low pass filter with length of 8.

$$H[n] = [1/8, 1/8, 1/8, 1/8, 1/8, 1/8, 1/8, 1/8]$$

The frequency response is as plot.

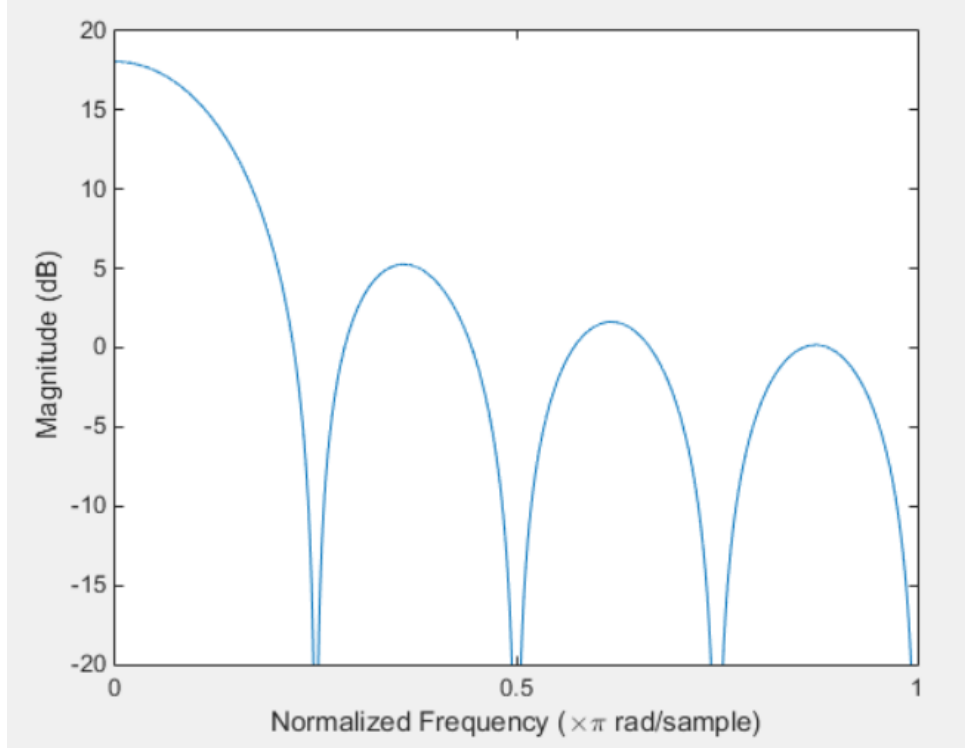


Figure 12. Frequency response of an averaging filter of length 8

From the frequency response, the half width of the main lobe is smaller than $\pi/4$. Therefore the averaging is a good pre-downsampling filter. However, there is still some minor problem, the magnitude sidelobe is a bit too high. The highest value of the side lobe is at 5dB, which may reduce the SNR (signal to noise ratio).

2.5.2 Accuracy of pitch detection

The accuracy of detected pitch is important to our project. According to Wikipedia of piano pitch frequency, “the following equation gives the frequency f of the n th key, as shown in the table[2]:

$$f(n) = \left(\sqrt[12]{2}\right)^{n-49} \times 440 \text{ Hz}$$

Eq. 1

For example, \mathbb{F}_5 has frequency 880.000 Hz, and its next pitch is $\mathbb{F}_{\mathbb{F}_5}$ whose frequency is 932.328 Hz. The tolerance of each pitch will be $\pm 3\%$, that means, pitches in range from 853.836 Hz to 906.164 Hz will all be recognized as \mathbb{F}_5 . Since the highest frequency after FFT will be tested as the pitch, there is 5% error rate saved for background noise.

III. Cost and Schedule

3.1 Cost

3.1.1 Labor

Our development cost for this project is to be \$30/hour, 15 hours/week for three people. There are total 16 weeks for this semester and we consider committing 65% for our final project.

$$\$30/\text{hour} * 15 \text{ hours/week} * 16 \text{ weeks} * 65\% * 3 \text{ people} = \$14,040$$

3.1.2 Parts

Part	Price
Microcontroller: ARM® 32-bit Cortex®-M7 CPU with FPU, adaptive real-time accelerator (ART Accelerator™) and L1-cache (ebay)	\$28.99
FPGA: Altera Cyclone IV EP4CE10E22 (ebay)	\$65
1.8" Color TFT LCD display with MicroSD Card Breakout- ST7735R	\$19.95
Voltage regulators. The voltage regulator is LP3964	\$5
Capacitors and resistors	From lab
Audio jack, J-Link socket, J-Tag soccket	\$20
crystal clock oscillator	\$10
Total	\$148.98

The total cost will be the sum of labor cost and parts cost, which is \$14188.98.

3.2 Schedule

Week	Lin	Qian	Xinyue
2/18/18	Researching on PCB design.	Research and design display	Work on design document
2/25/18	Continue on PCB design, then verify	initiate the FPGA routing protocol	Start connecting LED with side effects

	the design with TAs	programming	
3/4/18	Order PCB Purchase functioning testing board for internet.	Continue FPGA programming and make display visualizable	Study how to program to display
3/11/18	Start working on MCU programming	Begin to connect microcontroller unit to give simple response on the display	Start to use IDE and program on display
3/18/18	Communicate with MCU through serial port. Read data from microphone	Continue working on data transmission from microcontroller to FPGA	Continue working on programming
3/25/18	Start on soldering job (if PCB arrives)	Continue working on data transmission from microcontroller to FPGA and test and debug	Find online instrument source, and test them. Pick three of them
4/1/18	Testing PCB, and component on PCB.	Begin to make "music transcription" moveable on the display as we expected	Connect the LED when replay music.
4/8/18	Complete debugging and testing. Finish build job	Complete the ideal version of our project And fix any bugs during testing	Wrap up the code and test the functionality.
4/15/18	Demo	Demo	Complete the ideal version of our project
4/22/18	Working on final paper	Working on final paper	Fix any bugs during testing
4/29/18	Final touch on lab notebook	Final touch on lab notebook	Prepare for final Presentation

IV. Ethics and Safety

Since it's a sound visualization model, there will be some music pieces and even an entire song involved to test the functionality. It's necessary to follow the IEEE Code of Ethics, #6: "to seek, accept, and offer honest criticism of technical work, to acknowledge and correct errors, and to credit properly the contributions of others" [3]. To respect musician's work, music that has copyright will not be spreaded and will not be saved for commercial purpose neither.

In order to test human sound as input, some people with different voice will be invited to Help and we will record them sing and talk. Therefore, according to ACM code of Ethics and Professional Conduct, #1.7: "Respect the privacy of others." [4], we will always ask if they are willing to share their recording with us, and if it's good to use their recording during our demo. If any one of them says that it's not good to use his/her voice, we will remove his/her recording from the memory.

The main safety concern of this project is the DC power supply. It's mentioned in the DC Power Electrical Safety Guidelines: "Ensure that the polarity of the DC input wiring is correct. Under certain conditions, connections with reversed polarity might trip the primary circuit breaker or damage the equipment." [5] We will carefully mark the polarity before turning on the switch. Furthermore, according to the article, What's the Difference Between AC and DC Electric Shocks: "DC current will make a single continuous contraction of the muscles, and can cause fibrillation of the heart at high enough levels." [6] We will always check if the switch is turned off before we debug our circuit. So that we won't get injured by the power.

V. Reference

- [1] AppCrawlr, 'Best pitch detection apps for ios (Top 100)', [Online]. Available: <http://appcrawlr.com/ios-apps/best-apps-pitch-detection> [Accessed: 20- Feb- 2018]
- [2] Wikipedia, 'Piano Key Frequencies', 2018, [Online]. Available: https://en.wikipedia.org/wiki/Piano_key_frequencies [Accessed: 22- Feb- 2018]
- [3] Open Audio, 'Benchmarking - FFT Speed', 2016. [Online]. Available: <http://openaudio.blogspot.com/2016/09/benchmarking-fft-speed.html> [Accessed: 22- Feb- 2018]
- [4] Life.Augmented, 'System Workbench for STM32: free IDE on Windows, Linux and OS X', 2018. [Online]. Available: <http://www.st.com/en/development-tools/sw4stm32.html> [Accessed: 20- Feb- 2018]
- [5] IEEE, 'IEEE Code of Ethics', Section 7. [Online]. Available: <https://www.ieee.org/about/corporate/governance/p7-8.html> [Accessed: 20- Feb- 2018]
- [6] ACM, 'ACM code of Ethics and Professional Conduct', 1992. [Online]. Available: <https://www.acm.org/about-acm/acm-code-of-ethics-and-professional-conduct#sect4> [Accessed: 22- Feb- 2018]
- [7] Juniper Networks, 'DC Power Electrical Safety Guidelines', 2015. [Online]. Available: https://www.juniper.net/documentation/en_US/release-independent/jsa/topics/reference/safety/dc-power-jsa-electrical-safety-guidelines.html [Accessed: 22- Feb- 2018]
- [8] Bright Hub Engineering, 'What's the Difference Between AC and DC Electric Shocks', 2015. [Online]. Available: <http://www.brighthubengineering.com/power-plants/89792-ac-and-dc-shock-comparison/> [Accessed: 20- Feb- 2018]

[9] Carmine Noviello, 'Build STM32 applications with Eclipse, GCC and STM32Cube' 2015.

[Online]. Available:

<https://www.carminenoviello.com/2015/06/04/stm32-applications-eclipse-gcc-stcube/>

[Accessed: 20- Feb- 2018]