Alpha Hearing

PHT

Portable Hearing Test

Created By:
Nizar Abdul-Halim
Sami Abdelhalem
Mustafa Sheikh

SCHULICH
School of Engineering
UNIVERSITY OF CALGARY

DEPARTMENT OF ELECTRICAL
AND COMPUTER ENGINEERING

Table of Contents

Introduction:

The purpose of this project was to utilize the materials learnt from the courses taught in the second year of the electrical engineering program at the University of Calgary, to build a audio playing device to fulfill a specific need or perform a specific task. Throughout the building process of this project, there have been a lot of different constraints that needed to be considered. To tackle different obstacles that came in our path our project was split up into different smaller sections that were then divided amongst our group.

Target Market and Competitors:

This product is labeled under “Medical Instruments” which makes it portray the image of it being used only by professionals, but in fact it is not the case. The objective of building this product is to make hearing tests more accessible for people in need of medical help but are facing financial or circumstantial problems. The market value of the medical device industry is estimated to be around US$173 this year (2019) [1], this brings a lot of opportunity for our device to contribute to this growing market since there is not much competition in this field. The technology used to make the device may be outdated and simple, but after testing and getting feedback from people of interest, the device has proven to be successful and beneficial.

Persona Description:

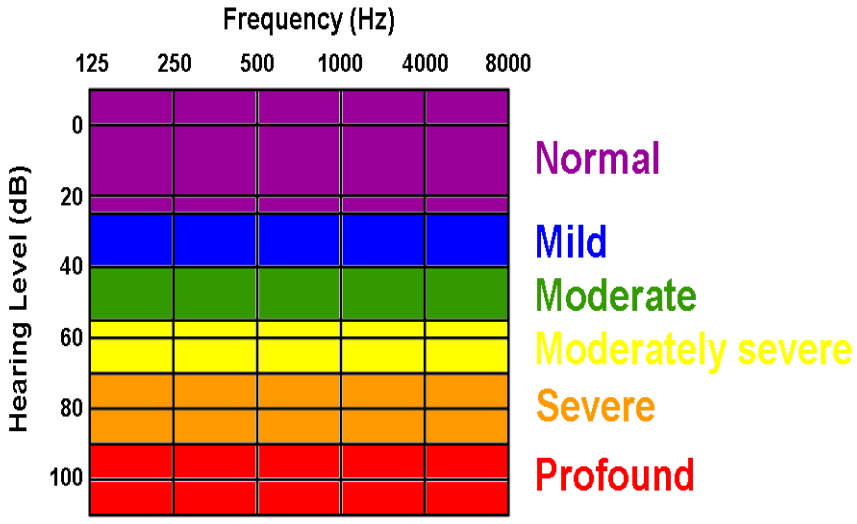
The PHT was developed to help those suffering from hearing loss and trying to make it accessible for everyone for an affordable price and from the comfort of their own home. Also, one of our development team’s goals is to supply the PHT to the military to help in active war zones. Having our device as a tool while in an isolated area will allow soldiers to know whether they are suffering from hearing loss. Knowing the state of your hearing can help save lives as a soldier inflicted with hearing loss will not be as perceptive of quiet sounds in a tactical situation and will not be able to warn their comrades if danger is imminent. Below is the description of the persona of a user of the PHT:

Malcolm is twenty-four years old and was deployed to Iraq two months ago. His father is a general and his brother is in the navy. His family has been enlisting to serve their country for decades. When Malcolm arrived at home base in Iraq he got introduced to his squad and as time passed, they grew closer. A month into his deployment Malcolm and his team got dispatched to verify if a bomb strike on a terrorist cell eliminated all threats. Upon arrival they spotted a bloodied man standing near the wreckage of the building. The team remained on guard and instructed the man to stay put. The man didn’t seem to hear their instructions and charged at the

soldiers. The soldiers took cover behind their armored vehicle and fired shots at the man. Unfortunately, Malcolm lost his footing on some loose debris and was unable to find cover. The man self-detonated thirty feet away from Malcolm. Malcolm dropped to the ground and covered his head as he heard a large BOOM...then silence. He floated in and out of consciousness as he saw his team crowd around him. He could see them yelling but he could not hear anything they said. He was loaded into the rear of the vehicle as his squad rushed him back to home base for treatment. After a couple minutes Malcolm started hearing a high-pitched ringing in his ears which eventually faded away. After the medic treated his wounds and removed shrapnel from his back and legs he was sent to the infirmary to rest. Soon Malcolm was moving around comfortably and felt he was back to normal. After the on-site medic saw Malcolm recovering, he used the PHT to administer a hearing test on-site. The results showed Malcolm had severely impaired hearing (RED) and the medic was unable to sign off for him to report back to his post. A day later Malcolm was flown to a nearby ENT specialist to determine what can be done to restore his hearing. The specialist concluded that Malcolm had burst an eardrum and if it was left untreated his ear could have become infected and never fully heal. After administering eardrops, a recovery period of two weeks was suggested, and Malcolm was discharged. The assessment of Malcolm's hearing on site using our PHT could have potentially saved his or his comrade's lives since his hearing loss imposes a liability upon his crew.

Concept Explanation:

The idea behind this project is to simulate an audiogram test and make it portable and simple to use. Typical hearing ranges of healthy humans varies with age and exposure to high intensity noises but can be generally summarized through the following graph.



[2]

With the PHT our idea is to create an inexpensive self-diagnostic hearing test, using common frequencies and introducing a gradual increase in volume as a way to test at what point on the above graph one is able to detect a sound. The following frequencies are tested:

Frequency	Decibel Range
250 Hz	0-80 dB
500 Hz	0-80 dB
750 Hz	0-80 dB
1000 Hz	0-80 dB
2000 Hz	0-80 dB
4000 Hz	0-80 dB
6000 Hz	0-80 dB
8000 Hz	0-80 dB

While each frequency is playing, the program waits for a response to log the decibel level that was detected. After all the frequencies are played a calculation is made using the average decibel range that was heard and the result of how well the individual can hear is displayed at the end.

Project Requirements:

The process of developing this product was guided by a set of requirements and constraints that were set by both the Schulich School of Engineering and the Alpha Hearing team. These are listed below.

Design Constraints:

During the development of this project, there were some technical and design constraints that have been imposed to better develop our device, which includes:

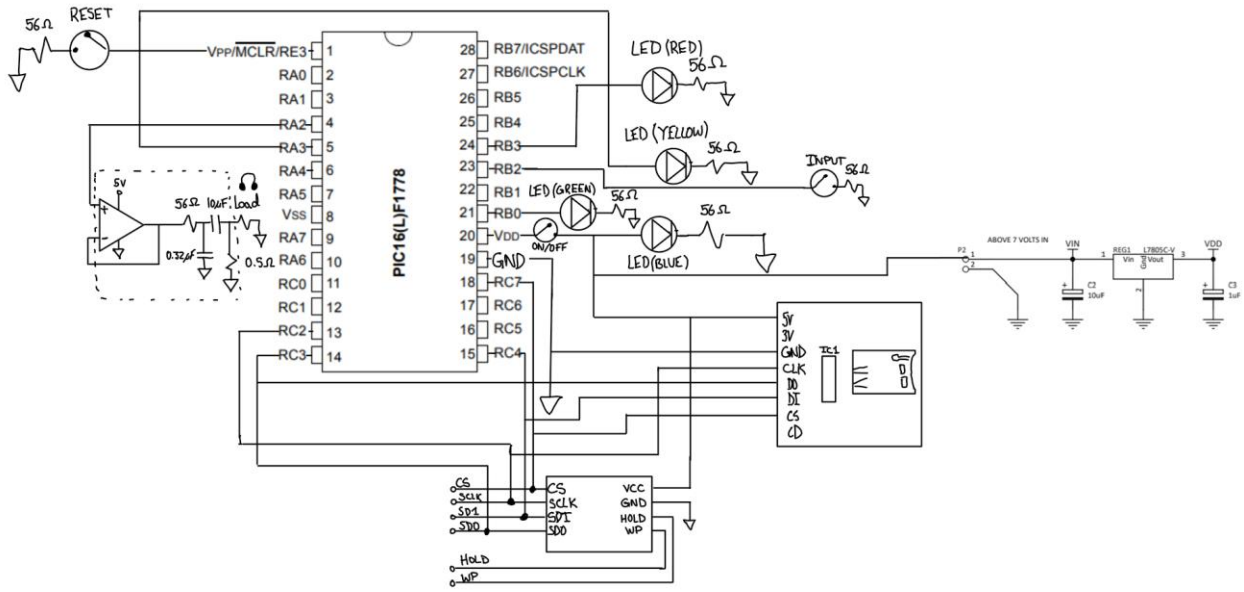
1- Must implement the PIC16F1778.
2- Must include sections from course material.
3- Must play tones from the SD card once the button is pressed.
4- Must work with headphones plugged into AUX.
5- Must use parts authorized by the Schulich School of Engineering.
6- Must be portable and hand-held.
7- Must have a purposeful use.

Financial Constraints:

For the design and implementation of our design our spending was limited to $60. To the relatively low cost of parts we were able to complete our project well below budget and are happy with our final product.

Circuit Diagrams:

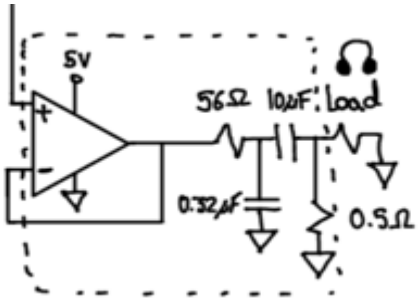
Overall Diagram:



Description:

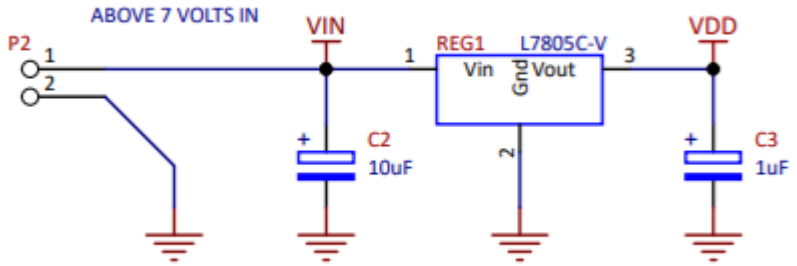
Our circuit involved communication between a micro-SD card and the PIC16(L)F1778. The digital audio signals received from the micro-SD card are processed by the Serial Peripheral Interface through a series of linking and communication by the PIC and the SD card. It is then sent to the DAC (digital-to-analog converter) and outputted to headphones through our buffer amplifier and then through our bandpass filter. The audio output is controlled by one button connected to the RB2 pin of the PIC and VDD which acts as an input. There is a reset button connected to the MCLR/RE3 pin it is also connected to ground and that pin is configured to reset the PIC whenever it is grounded. There are output green, yellow and red LED's connected to pins RB0, RA3 and RB3 respectively, these are used to show the user the result of the hearing test. The bits for SD1 and SD0 that are read from the SD card are inputted into the PIC through pins RC4 and RC3 respectively.

Amplifier and Bandpass Filter Circuit:



The amplifier setup used here is a buffer amplifier which does not amplify the voltage it is supplied, but it prevents the outputted signal to interfere with the supplied signal. The first RC low-pass filter consists of a 56-ohm resister and a 0.32 micro-farad capacitor, the purpose of this filter is to prevent frequencies greater than 8kHz from being outputted to the user. The second CR high-pass filter consists of a 0.5-ohm resistor and a 10 micro-farad capacitor, the purpose of this filter is to prevent frequencies lower than 240Hz from being outputted to the user to further eliminate any noise in the circuit.

Voltage Regulator Circuit:



[3]

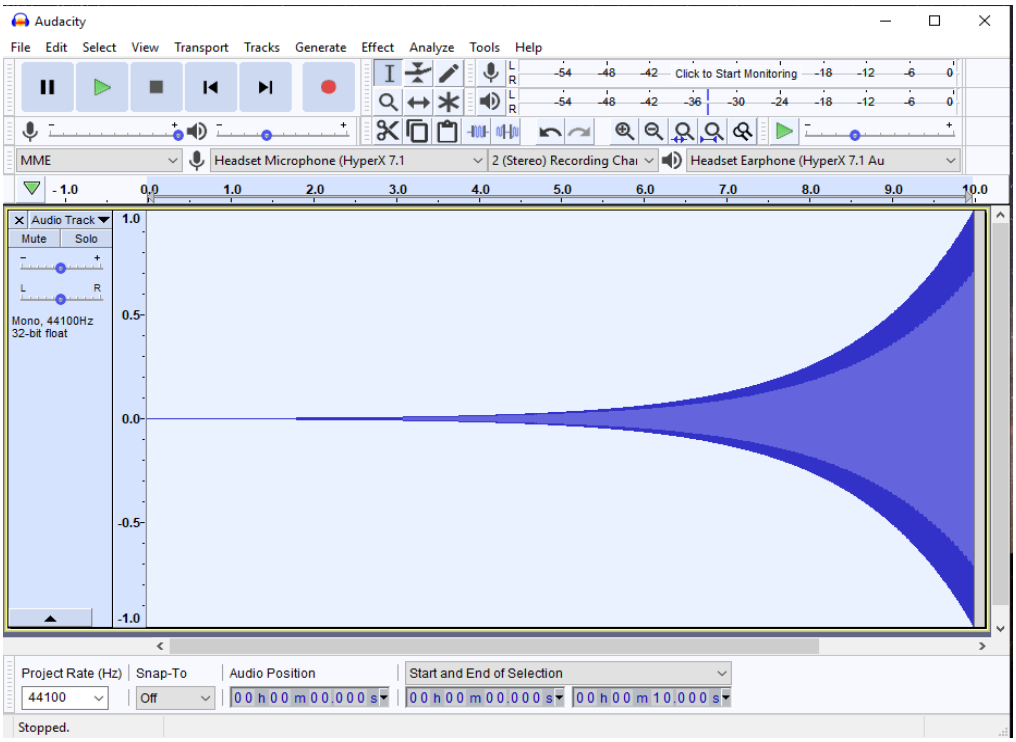
The PIC microcontroller can only handle 5 volts, anything more than that can damage the PIC and potentially the entire circuit. To solve this problem a voltage regulator circuit was introduced, where it connects to any power source above 7 volts accompanied by a 10 micro-farad capacitor from one end, and then outputs 5 volts accompanied by a 1 micro-farad capacitor. The output of this circuit is connected to the VDD of the PIC microcontroller.

PIC Programming:

The programming of the 8-bit PICF1778 microchip was a semester long effort. Beginning with our labs which was meant for us to get used to the coding and debugging process using the MPLAB IDE. After developing the basic code of reading and writing to the SD card and outputting an audio file from it out of a speaker, we decided to actually use the SD card to preload frequencies and read it from the PIC.

Tone Generation:

We used audacity to generate tones and used effects to simulate a linear increase in volume



Preparing Audio File for PIC Reading:

To achieve a clear sin wave, a sampling rate of 44.1 kHz had to be chosen, then it had to be converted into binary files and manually copied into the SD card using the HEX editor software.

Playing Frequencies:

The code that was used to output the files from the memory card to output essentially begins a series of commands and once the PIC and memory card is ready begin the transfer of data. Our SD card works in a way where it sends chunks of 512 bytes of data at a time. We programmed it

to continuously read one byte from the memory card and output it to the digital to analog converter at the same sampling rate that was exported from the audacity (44.1kHz). We used the PIC's built in timer module to create the effect of outputting a continuous wave by holding a varying 8-bit value to be outputted

]

Files Description:

Main:

The main file initiates many startup functions and contains the prototypes that are required to begin our program. From the beginning when the main is launched, the functions to configure the SPI and the DAC are declared. Then the functions for the two timers timer2 and timer4 are also declared and then the functions that are used to control interactive peripherals such as the LED's and inputs are declared.

Prototypes:

```
#include <pic.h>
#include "Lab3_Config.h"
#include "Lab3_SPI.h"
#include "Lab3_SD.h"
#include <xc.h>
#include <stdio.h>
#include "Lab2_Library.h"


//0x00,0x01,0x5F,0x90) HORSES IN THE BACK SONG SECTOR LOCATION


void Lab2_WriteDAQ(char WriteValue);
void initP(void);
void initDAC(void);
void initB(void);
void resetaudio(void);
void setsector(unsigned char a3, unsigned char a2, unsigned char a1, unsigned char a0);
void Pulse(void);
void Result(void);
void LED(char x);
void Lab2_ConfigureTimer2(char TimerPeriod_us);
void ConfigureTimer4(char TimerPeriod_us);
void wait(unsigned char s);
```

Variables:

All the variables used are initialized to zero except the arrays for time tracking. RA2 will be set in the DAC initialization to be set as an analog output.

```
28
29   unsigned short counter = 0; //setting a counter to keep track of where the pointer is in the memory of the SD card.
30
31   unsigned char x3 = 0, x2 = 0x00, x1 = 0x00, x0 = 0x00, b = 0, c=0,d=0; //44.1 KHz 250Hz sin
32
33   int e = 0;// intializing to sector 0 and c counter
34
35   int sec = 0; //counter for seconds passed
36
37
38   long unsigned int seconds [8]; //records how many full seconds each recording took
39
40   long unsigned int ticks [8];  // record how many ticks/242 after every second, 2 <<
41
42   double result [8];
43
44
45
46  #define INPUT PORTBbits.RB2
47
48  #define GLED PORTBbits.RB0
49
50  #define YLED PORTAbits.RA3
51
52  #define RLED PORTBbits.RB3
53
54  #define SECONDFLAG PIR4bits.TMR4IF
55
```

Function Declarations and Pausing:

Prior to the main while(1) loop, the frequency of the clock is set to the highest frequency (32MHz) , all functions for peripherals that are required to be initially set up are called, and timer2 is set to repeat every 23 microseconds. That is meant for the 44.1KHz playback speed of the audio files. Our method of waiting for a button click is to create two back to back while loops waiting for INPUT to be 1 and then 0. A wait function was created to counter the bounce issues that are intrinsic with buttons.

```
78   void main(void)
79  {
80
81      //Set the system clock speed to 32MHz and wait for the ready flag.
82      OSCCON = 0xF4;
83      while(OSCSTATbits.HFIOFR == 0);
84
85      //Initialize all B port peripherals
86      initB();
87
88      //Initialize all required peripherals
89      initP();
90
91      // Initialize the Digital to Analog Converter
92      initDAC();
93
94      // Initialize Timer2 at ~44100Hz
95      Lab2_ConfigureTimer2(23);
96
97      //Lights off
98      LED(0);
99
100     //Wait for Input to begin program
101     while(INPUT==0)
102         wait(50);
103
104     while(INPUT==1)
105         wait(50);
106
107     //Start timer4 set to 1 second and begin timer
108     ConfigureTimer4(242);
109
110     //lights on
111     LED(1);
112
113
114     while(1)
115     {
```

Byte Read/Output and Timer/Button Check:

In the main while(1) loop the 512 byte sectors of the SD card are constantly being read from the SPI and outputted to the DAC. There are two functions that are implemented at the beginning, TimerCheck() which is meant to increment the variable sec every time a second has elapsed using the flag from timer4 allowing for the tracking of playback time. CheckButton() is used to track the INPUT tied to RB2 which is a digital input high active sensor. It increments c every time C the button is pressed.

```
114     while(1)
115     {
116     TimerCheck();
117     CheckButton();
118
119         if (counter==0)
120          Start_Read(x3,x2,x1,x0); //beginning the read process every time the counter finishes a sector
121
122         Lab2_WriteDAQ(SPI_Read()); //using pre made functions to write each byte through SPI to the DAC
123
124         counter++;
125
126         if(counter==512) //when the counter reaches the end of a sector this is required to skip the weird bit between.
127          {
128             SPI_Read(); //reading three times because after each sector there is 12 instructions that occur to load
129                         //the next block of 512 bytes
130             SPI_Read();
131             SPI_Read();
132
133             counter = 0; //reset the counter
134
135             x0++; //increment the sector address
136
137             //Increment the sector address
138             if(x0==0)
139             {
140                 x1++;
141                 if(x1==0)
142                 {
143                     x2++;
144                     if(x2==0)
145                     {
146                         x3++;
```

Timer4 Function:

Timer4 (1 second timer) function was created in order to be able to keep track of the length of time that each frequency was played for. T4CLKCON which is the register controlling the frequency of the timer's ticks is set 31KHz (the lowest frequency) and a pre scaler of 1:128 is used with the T4CON register that allows the T4TMR register to increment once every 4 milliseconds. Timerr2 is identical to this with the exception of using the 32MHz system clock and being pre scaled by 1:32 instead.

```
482  void ConfigureTimer4(char TimerPeriod_us)
483  {
484      // Set the system clock speed to 32MHz.
485
486      OSCCON = 0xF4;
487
488
489
490      //Wait for the oscillator to switch to its new speed.
491      while(OSCSTATbits.HFIOFR == 0);
492
493      // T2PR: TMR4 PERIOD REGISTER (Page 287)
494      // The timer runs at ~4 ms per tick 242.187 ticks/second after using 31 kHz and pre scale 1:128
495      T4PR = TimerPeriod_us;
496
497      // T2CLKCON: TIMER4 CLOCK SELECTION REGISTER (Page 306))
498      // Bits 8:4 = 0b00000   These bits are not implemented
499      // Bits 3:0 = 0b0100    Clock source is LFINTOSC 31 khz
500      T4CLKCON = 0x04;
501
502      // T2CON: TIMER4 CONTROL REGISTER (Page 307)
503      // Bit  7   = 0b1       Timer is on.
504      // Bits 6:4 = 0b101     Prescaler is 1:128 (Timer increments every 128 cycles)
505      // Bits 3:0 = 0b0000    Postscaler is 1:1
506      T4CON = 0xF0;
507
508      T4TMR = 0;
509      // TMR4IF is an interrupt flag, and is set whenever Timer2 expires.
510      // We are not using interrupts, but it can still be checked manually.
511      // It is cleared here as we start the timer.
512      PIR4bits.TMR4IF = 0;
513
514
515      return;
516  }
```

DAC Function:

This function is used every 23 microseconds to output out of RA2 the byte value specified by the frequency file which is read from the SPI.

```
316  //function to write bytes from SPI into DAC
317  void Lab2_WriteDAQ(char WriteValue)
318    {
319      DAC1REFH = 0;
320      DAC1REFL = WriteValue; //the 8 bit/0-256/1byte data from spi
321      DACLD = 1; //right justified
322      return;
323    }
```

State Checking and Pausing:

Every 512 bytes a state check is performed and the position in the memory is tracked with a c variable or the last sector address of each audio file. If the state is changing a series of functions begin,

```
152         //*****************************
153         // When a frequency file is playing, if it reaches the end of the file or an input is detected
154         // the following instructions occur:
155         //
156         // 1. Turn lights off
157         // 2. Record time
158         // 3. Reset the timer
159         // 4. Wait for input from user
160         // 5. Set the address of the next frequency file
161         // 6. Reset all constants
162         // 7. Turn lights on
163         // *****************************
164
165         if(x3==0x00 && x2==0x00 && x1==0x03 && x0==0xA9 || c==1)//250Hz end
166         {
167             LED(0);
168             seconds[0] = sec; // populating array with seconds elapsed
169             ticks[0] = (int)T2PR; // populating with ticks after whole number
170             SECONDFLAG = 0; //reset second timer and set flag to zero
171             while(INPUT==0)
172                         wait(50);
173
174             while(INPUT==1)
175                         wait(50);
176             setsector(0x00,0x00,0x27,0x10); //500Hz beginning
177             counter = 0;
178             c=2;
179             sec = 0;
180             SECONDFLAG = 0; //reset second timer and set flag to zero
181             LED(1);
182         }
183         else if((x3==0x00 && x2==0x00 && x1==0x2A && x0==0xB9) || c==3)//500hz end
184         {
185             LED(0);
```

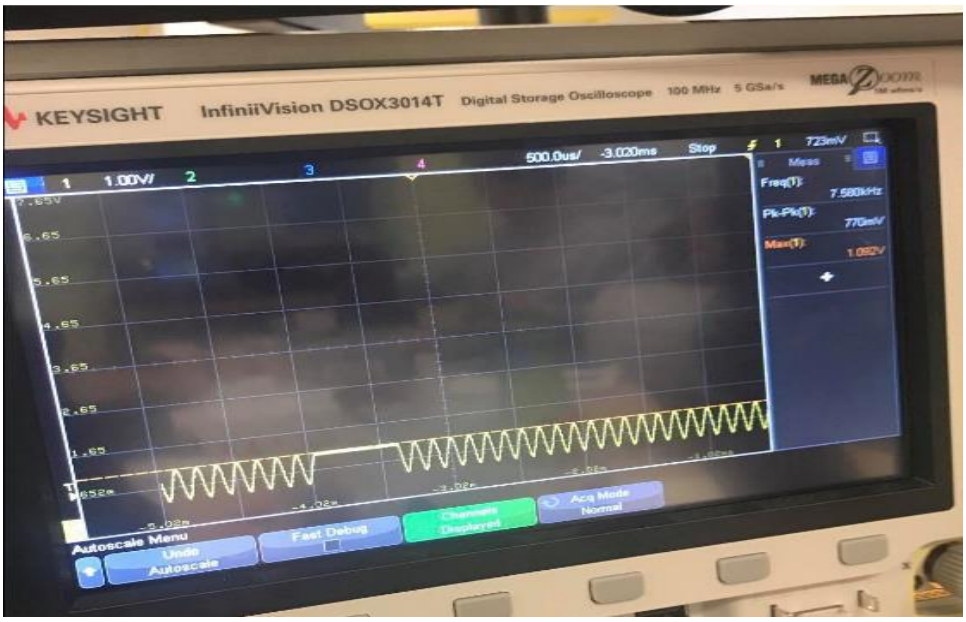
Result:

The Result() function is called after the last audio file is played and uses the recorded times, begins a calculation of the average of seconds recorded for each frequency. Since we mapped the times with decibels, we have a pretty clear idea of how many seconds into the file no/mild/severe hearing loss would be.

```
387     void Result()
388     {
389         double average= 0;
390         for (int i = 0; i<8; i++)
391          {
392          result[i] = seconds[i]+(ticks[i]/244);
393          average += result[i];
394          }
395         average = average/8;
396
397         if(average<=4) //excellent hearing greenlight flash
398          {
399             while(1)
400             {
401                 TimerCheck();
402                 if(sec%2==0)
403                 GLED = 1;
404
405                 if(sec%2==1)
406                 GLED = 0;
407
408                 /*if(INPUT ==1)
409                  break;*/
410             }
411
412          }
413         else if (average>=4 && average<=7) //slight hearing loss
414          {
415             while(1)
416             {
417                 TimerCheck();
418                 if(sec%2==0)
419                 YLED = 1;
420
421                 if(sec%2==1)
422                 YLED = 0;
```

Coding Constraints:

Due to the limitations of this processor, playing a back a continuous audio file like a single frequency tone proves to be slightly difficult because of the relatively slow speed of the clock. Even though we are using 32 MHz, since the processor uses 4 clock cycles per instruction of assembly, the clock speed is effectively 8 MHz Using 44.1 KHz for the sampling rate that would only give us 180 instruction cycles space to execute all of our code per byte loaded onto the DAC.



Due that constraint our sin waves produce slight gaps periodically slightly changing the sound output but does not change the frequency too drastically.

Functionality and Usability Changes:

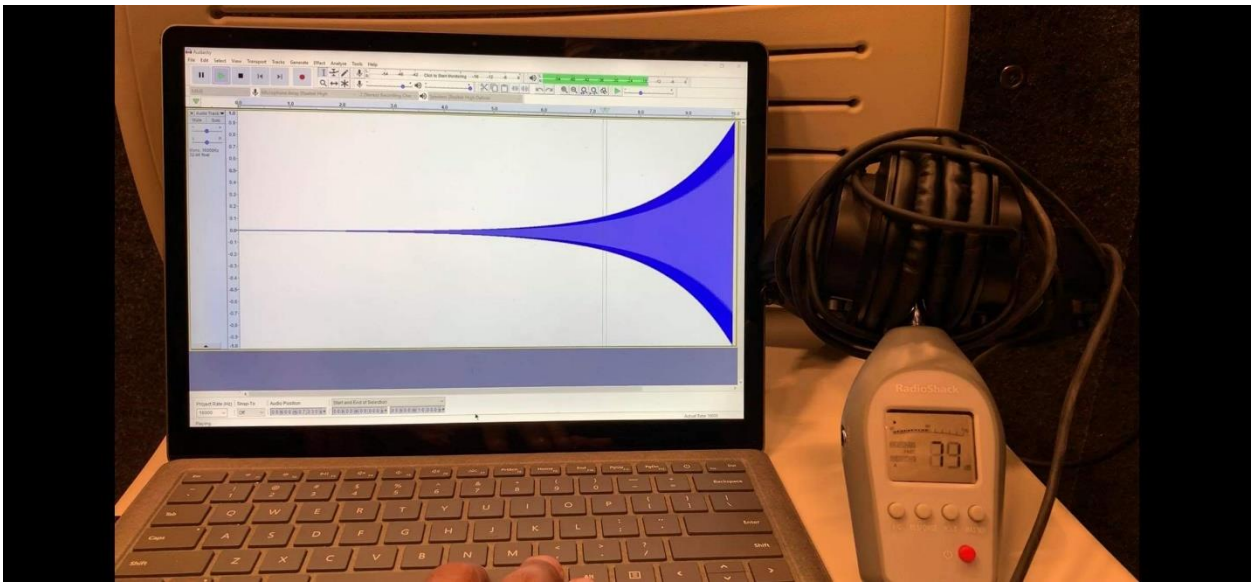
The device has gone through a lot of different design iterations to better improve the usability of the device to insure customer satisfaction. Initially we had introduced our device having one input button to control the device and go through the hearing test, this design had proven to be inconvenient since the user does not have the option of resetting the device without switching it off and on. Therefore, to improve the user experience we decided to add a reset button therefore if the user is distracted or does not press the button, they can go through the test again so that they get accurate results. Another change that was done was the replacement of the LCD screen in the first design with the colored LED's in the second and third iteration, this has helped us in making the form factor smaller, and thus making it more convenient for the user to carry it around. A power indicating LED was also added to inform the user whether the device is on or off thus preventing battery loss.

The amount and range of frequencies being tested has been changed as well, in the initial design the test involved testing frequencies between 200Hz and 20kHz, the test seemed to take too long, and the user was having difficulty hearing the frequencies over 10kHz. After doing some research our team decided to change the way the hearing test is done, the final product does 8 quick tests with frequencies between 250Hz and 8kHz since our research has shown that this range of frequencies is closely associated with the range of a human's voice. This helps the user enjoy the experience of having their hearing tested since it is more efficient and accurate.

Testing Methods:

Circuitry Testing:

To test our circuitry, we employed the use of the oscilloscope. To ensure all the connections and the resistance/capacitance values are proper, we used a decibel meter inserted inside the cups of the headphones, inside a soundproof recording studio, and began to output the desired frequencies through our filtered and amplified circuit to insure that there is no clipping and that the corner frequencies of our filters are proper. While testing to see if desired voltage is being outputted from a certain circuit element, we used a LED and plugged it into any part of the circuit to check and see if the desired voltage is present or not. Below is an example of one of the frequency outputs being tested.



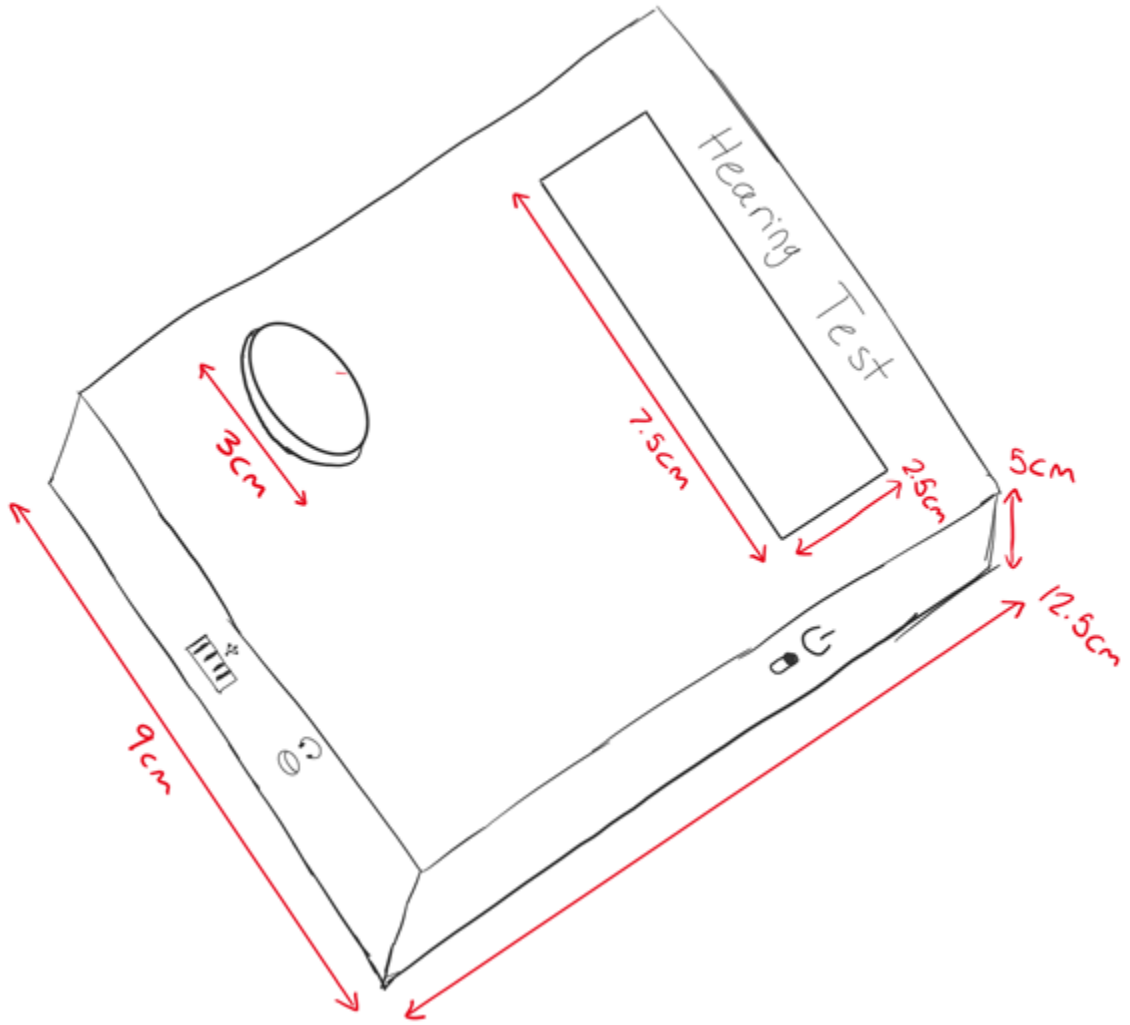
Programming Testing:

While programming our microprocessor we also used the oscilloscope. While frequencies were being outputted, we checked to make sure that there was an uninterrupted, clear sine wave. We implemented the use of a pulse signal that was outputted to the desired portion of our project, by doing this we can write the necessary code accompanied with the pulse function to ensure that the correct voltage is being outputted/inputted at the correct frequency and amplitude as we need it to be. Below is a snapshot of the pulse function definition.

```
361   void Pulse()
362   {
363    Lab2_WriteDAQ(0xFF);
364    wait(244);
365    Lab2_WriteDAQ(0x00);
366   }
```

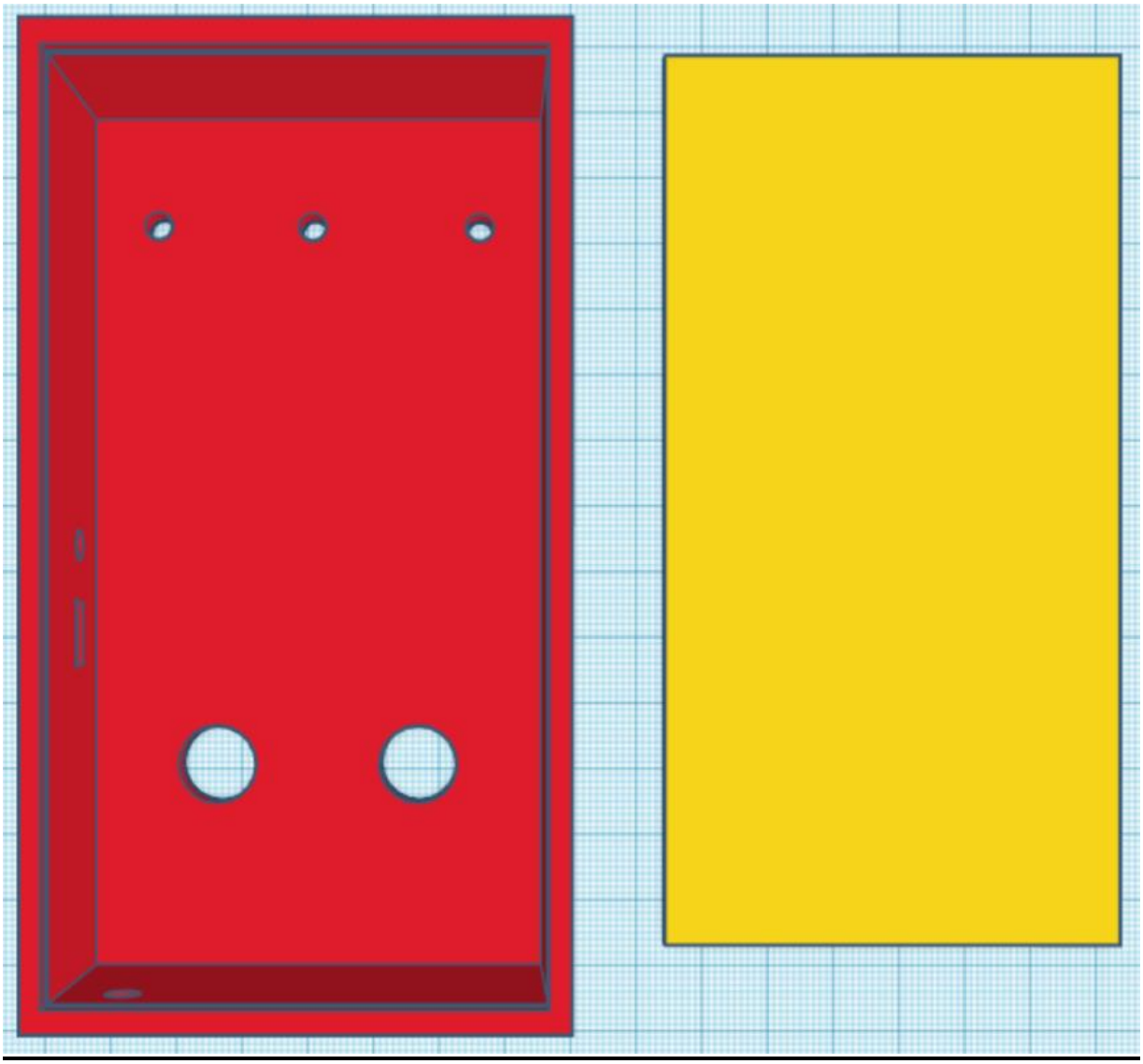
Case Design Development and Changes:

First Design Schematic:



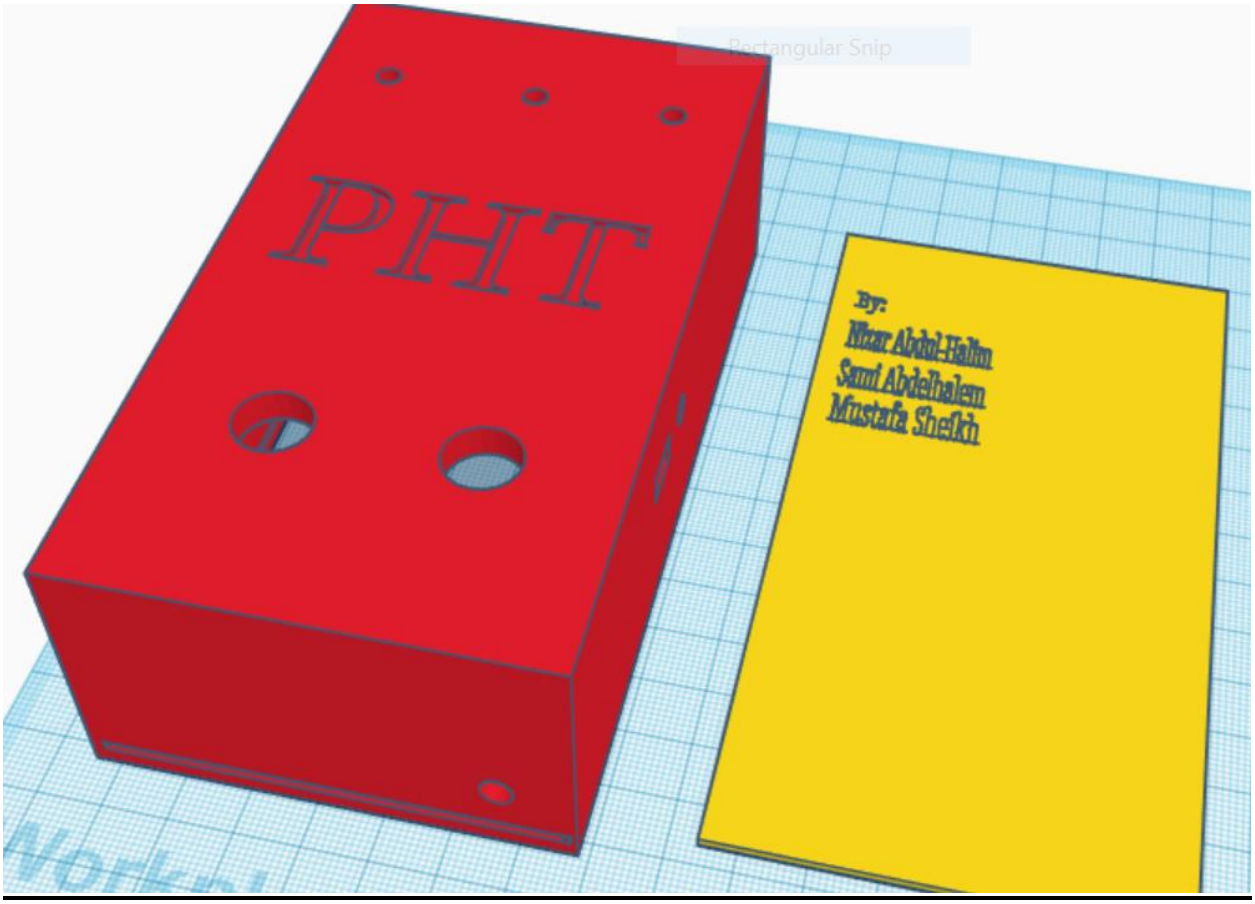
The initial design idea involved using one button to switch between consequently increasing frequencies along with an LCD screen that will display the results of the test. The device was to be powered by a lithium-ion battery that can be recharged using a micro USB cable.

Second Design Iteration:



The Second iteration to our design involved changing the power supply to exchangeable 9V batteries, this was done to extend the shelf life of our product and prevent any battery defects. The back panel of the casing was designed to slide in and out for easy battery replacement, and the LCD was replaced with 3 LED's (red, yellow, green) to indicate the results of the hearing test. This was done to reduce the form factor of the device and make it easier to carry around. Another button was added to the design to reset the test to make it more user friendly.

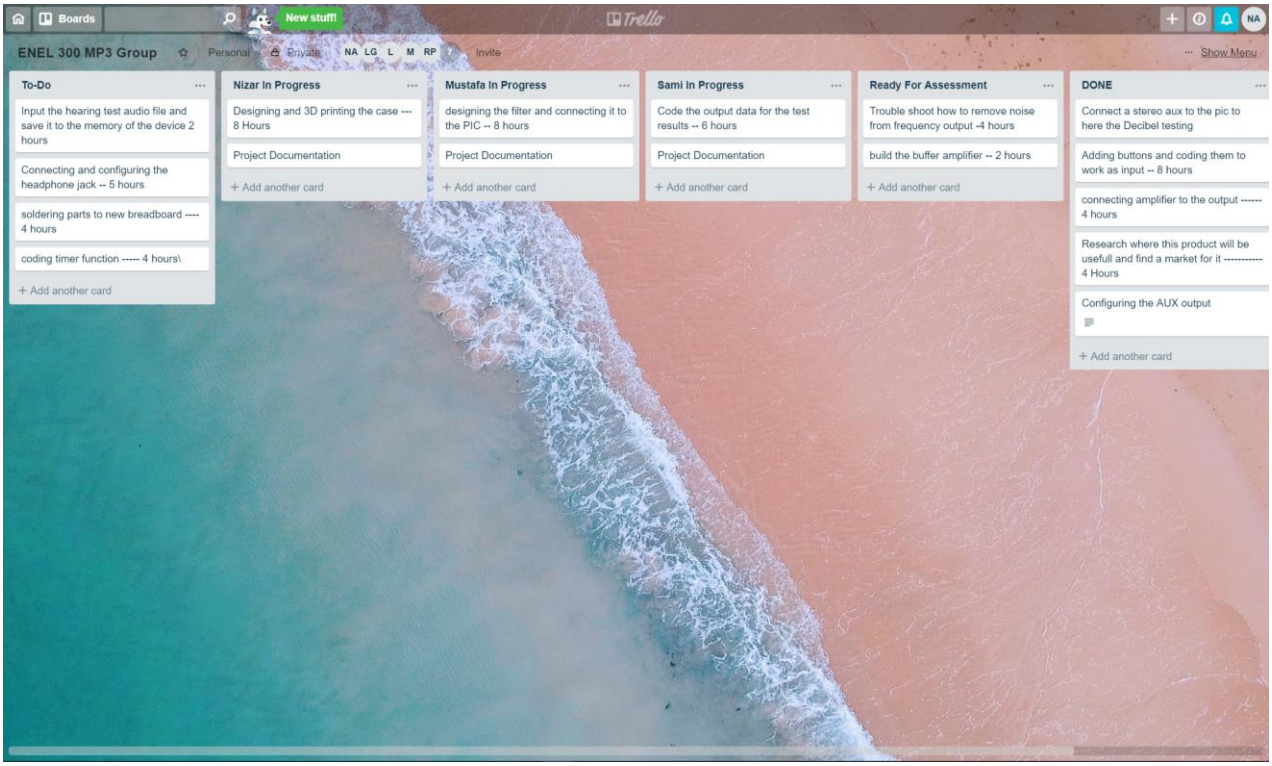
Third and Final Design Iteration:



The third and final iteration of the PHT involved a slightly bigger casing to help fit all the components inside, along with branding on the front and back of the device. The headphone jack was also moved lower to the bottom of the device for better usability. Also, a LED indicating whether power was on or off has been added beside the switch to improve usability.

Teamwork:

There have been a lot of work hours put into the building of this project. The completion of this project would not have been possible without the combined efforts of all team members making sure everything is done by the given due date. Ensuring the proper function of the device at the best possible state it can, required many hours of combined debugging. To divide the work amongst each other we made use of a free software called Trello which allowed us to list the tasks that needed to be done with assorted columns that organized our work as such: 'In progress', 'Ready for Assessment' and 'Completed' (example screenshot attached below). The workload was split up into 3 different categories where one member worked on the coding of the device and optimizing the performance, another member was in charge of designing and building the physical components and the third member was in charge of the documentation and a quality assessment of each component of the device. Team members helped each other in fulfilling tasks if help was needed, ensuring the device meets the highest standards possible.



User Interviews:

After getting a working prototype, we conducted interviews with some potential users that our team thought fit the profile of our persona and our target audience. The users were asked questions about their experience using our device.

Interview 1: Khaled – Diagnosed with hearing loss, works in a loud environment. Has an appointment with a specialist scheduled in August.

1- What do you like about our product?
   - I like how easy and simple the test is, and the time convenience.
2- What can we do to improve our product?
   - If you make the case slimmer with softer edges it will feel better in the hand.
3- Would you say the PHT gave you a good idea of how severe your hearing is? What do you think is an adequate price point for this device?
   - Yes. I think my result (RED LED flashing) shows my current hearing ability as I do feel that I am missing a lot of things said to me. I think a price range of $50-$80 is good for the provided functionality.

Interview 2: Safwan – Engineering student, claims to have perfect hearing.

1- What do you think about the PHT?
   - I like how small and portable the device is.
2- Do you think the result you got depicts your current hearing ability?
   - Yes, the GREEN LED flashing proves that I have perfect hearing.
3- What can we do to improve our product?
   - I think you should have a clear-instructed user's manual to help make the test go smoother.

The feedback collected in the above interviews has been taken into consideration in the process of developing the final prototype, to make sure that maximum functionality and user satisfaction is achieved.

User Manual

Thank you for selecting the PHT (Portable Hearing Test) as your preferred hearing test kit. This device delivers to you a quick and easy way to check your hearing health and lets you decide whether medical aid is necessary.

The PHT features a 3.5mm audio jack through which your preferred set of headphones are connected. **NOTE: For an accurate result please use a high-quality headset with a frequency range of at least 100 Hz – 10000Hz.**

WARNING:
IF ANY DISCOMFORT IS EXPERIENCED WHILE USING THE DEVICE PLEASE POWER OFF IMMEDIATELY OR PRESS THE 'RESET' BUTTON. PLEASE ONLY USE THIS PRODUCT FOR ITS INTENDED PURPOSE.

STARTING HEARING TEST

1. Plug in your choice of headset into the auxiliary input located on the front end of the device.
2. Power on device by moving the power switch on the right side to **ON.** The blue LED should turn on indicating the device in powered on.
3. Press the **'SELECT'** button located on top of the device. When all LED's are powered on the test has begun and the frequency has started playing.
4. As soon as the frequency is heard please press the **'SELECT'** button again. If all LED's power off, your input has been registered.
5. Repeat steps 3 and 4 for the remainder of the test.
6. Results will be displayed via the LED's on top of the device.

NOTE: If any errors are made please press the '**RESET'** button to begin test again.

Result (What does it mean?)

Please ensure you have completed the test before checking your results. The LED with your result should pulse on and off until the device is either powered off or the '**RESET'** button is pressed.

NOTE: This product is not meant to replace a hearing test conducted by a medical professional. Please consult your family doctor or ENT specialist for advice on how to proceed.

GREEN: Excellent Hearing. Whew! You have nothing to worry about. Your test results show that you were able to detect all frequencies at normal dB levels.

YELLOW: Moderate Hearing. While you did detect most frequencies at an appropriate dB level, some of your hearing range is poor. We recommend booking an appointment with your specialist to determine the appropriate treatment.

RED: Poor Hearing. Your results show that you were unable to detect or detected sounds at high dB levels. For further action please visit a licensed medical professional.

References:

[1] “USA – Overview of medical device industry and healthcare statistics,” *Emergo,* 23-Oct-2018. [Online]. Available: https://www.emergobyul.com/resources/market-united-states. [Accessed: 18-Apr-2019].

[2] “Communication Status - Audiologic Rehabilitation,” *Google Sites*. [Online]. Available: https://sites.google.com/site/coreandcare/home/communication-status. [Accessed: 18-Apr-2019].

[3]” PIC16F1778 Development Board Documentation” Schulich School of Engineering. [Accessed: 18-Apr-2019].

[4] K. Murari, ‘ENEL 361 Professor’, University of Calgary, 2019.

[5] P. Kathol, ‘Teaching Assistant’, University of Calgary, 2019.

[6] J. Long, ‘Teaching Assistant’, University of Calgary, 2019.