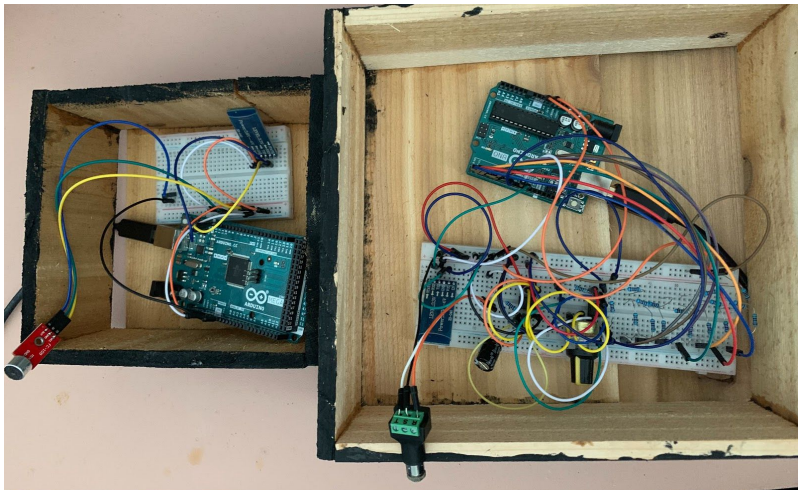
EE175AB Final Report

Bluetooth Audio Transmitter



EE 175AB Final Report

Department of Electrical Engineering, UC Riverside

Project Team Member(s)	Zohaib Khan, Rishi Kodukula, Karen Escareno
Date Submitted	3/18/2019
Section Professor	Roman Chomko
Revision	Version 1
Permanent Emails of all team members	Zohaib khan zkhan003@ucr.edu Karen Escareno kesca004@ucr.edu Rishi Kodukula: rkodu001@ucr.edu rishi.kodukula@gmail.com

Revisions

Version	Description of Version	Author(s)	Date Completed	Approval
Version Number	Information about the revision. This table does not need to be filled in whenever a document is touched, only when the version is being upgraded.	Full Name	00/00/00	
1	Final Draft	Zohaib Khan, Rishi Kodukula, Karen Escareno	03/18/2019	

Table of Contents

* Executive Summary

The Bluetooth Audio Transmitter is an input and output audio device and allows users to play music with a 3.5 mm jack. The user is able to connect their device to the transmitter and select a song of their choice and it will output in a connected speaker via bluetooth connection. The intended applications for this project is to connect any form of sound through a 3.5mm jack and be able to hear the corresponding real time audio.The overall goal for this device is for an accurate, clear, audio output so that the user is able to use this device to its fullest potential. The main objective as well is to create a device able to be compatible with most devices in the world regardless of what type of audio device the user is using for audio connection.

The B.A.T key features are a volume control dial to adjust volume and incorporates a speaker within the device. Moreover, it connects with any device that has a 3.5mm jack. Many Bluetooth Audio Transmitter do not have a connected speaker and does not connect with all devices that lack bluetooth functionality which sets our device apart from the rest of devices that are similar. Another component to the design is not only to have efficient sound quality, but to also make it affordable for the user. Speakers similar to this design in retail stores are fairly costly and can be complicated to use. The simplistic design of the bluetooth audio transmitter has allowed for the device to be cost effective totaling to less than twenty dollars. With a simplistic reliable design and sound users are guaranteed a reliable device with a lower risk of failure. This is in comparison to other devices in the market similar to this one with a higher price tag, which can bring complications with sound and become a high investment with a high risk of functionality.

This project is significant in today's world because it is able to do real time signal processing which leads to less delays in sound and creating an efficient sound. Furthermore, many devices have delay in sound, which can cause inaccuracy and frustrations with devices. This device uses reliably and cost effective effective components that allows this device to be so accessible to many individuals. Our system is cost effective, user friendly, and has very minimal latency.

The design itself is able to carry out all its functions with just needing an audio device from the user. Furthermore, the system is encapsulated in a wooden frame for an aesthetically pleasing device case. The wooden frame's purpose is so it can encompass the whole device and also protect it from damage. The wooden frame will allow the device to be portable and still weigh less than a pound. The goal is to allow this device to be durable and portable. The user will only have to input their device to the 3.5 mm jack and play their audio. This allows for the device to be used by any user regardless of prior knowledge with this technology.

To create a simple, but efficient design, the system uses a integrated software and hardware that does not diminish the quality of sound. The overall achievement was being to capture the analog signal, deconstruct it, and reconstruct it after to recreate the signal that was original inputted. Through all these challenges we were able to keep our idea of a simplistic design, but efficient in quality.

* Introduction

2.1 * Design Objectives and System Overview

The Bluetooth Audio Transmitter is a wireless speaker that takes sound from a normal audio jack. The output is the music that was selected and due to the bluetooth connection the speaker can be placed away from the device. The project is split into 3 major parts: Analog to Digital Conversion, Bluetooth Sending/Receiving, and Digital to Analog Conversion. The audio can come from a bluetooth device or using an aux cord.

For Analog to digital conversion we were able to use register programing which allowed us to manipulate the ADC. When looking at Bluetooth sending/Receiving through AT commands. Finally looking at the digital to analog conversion we used a DAC that will take care of the digital signal that is received from the bluetooth modules. This is connected to each of digital pins 0-7 to each of the 8 junctions in the DAC. Through these three major parts we were able to successfully create a Bluetooth Audio Transmitter with an adjustable speaker.

The purpose of this project was to simulate and create a foundation to develop life long skills we will continue to use. The importance of this project is to further understand and become familiarized with Digital signal processing,Register Programming,Sampling rates, Analog and Digital Signals, Aliasing & Quantization, UART, SPI, ADC, DAC, Debugging, Peer Communication and Collaboration. This project is meaningful in that we were able to use a full range on engineering skills and concepts to create this project. It also helped strengthen our understanding of real time processing and enhance our planning and researching. The project is related to electrical engineering in that it uses both electricity and electronics.We applied both hardware and software to design a system of electronics.There were many trial and error within this project that we were able to overcome, but it took patience and understanding on what was occurring. In world perspective this project is meaningful in that it is able to use a larger set of devices both new and old. This is helpful in the market space that it can be favorable to a larger audience. The overall goal for the projects is to simulate analog signals through digitize signals and vice versa. As well as we wanted our Bluetooth Audio Transmitter to work properly, consistently and efficiently.

We are acquiring that our technical design objectives is a system that requires an audio signal that will be digitized by the transmitter before being sent to the receiver to output the sound. The intended applications for this project is to connect any form of sound through a 3.5mm jack and be able to hear the corresponding real time audio. Real-time processes apply to all core concepts of technology, such as digital communication, digital signal processing and real time processing.

-written by Zohaib

Design Objectives:

- Range: 20 Ft +
- Sampling Rate: 40KHZ - 80KHZ
- Accuracy: 5 % Error
- Response Time: Microseconds
- Input Frequency (Depends on Signal): 100 HZ to 40 KHZ

Responsibilities:
Zohaib Khan -
Hardware: 8 Bit Dac, Lowpass, Power Amplifier, 3.5 mm jack, signal amplitude, coupling circuit and dc offset
Software: ADC,Uart, and SPI Register Programming(not all were implemented in final design)

Karen Escareno -
Hardware: Bluetooth
Software: Bluetooth connection, Bluetooth communication, Timer for Bluetooth operations

Rishi Kodukula -
Hardware: D2A conversion over a chip, soldering
Software: Arduino code

2.2 * Backgrounds and Prior Art

HiGoing Bluetooth 5.0 transmitter Receiver: Cost $36.99 on Amazon. It is a Bluetooth transmitter/receiver that can stream stable audio and sound wirelessly over bluetooth connections. The Bluetooth transmitter receiver has a 24 hour battery life that can be recharged for three hours. The device can be connected to wired stereo, speakers, audio outputs, or headphones.

- **Advantages of our product:** Our product comes with a wired speaker, while this product does not come with any audio device.
- **Disadvantage of our product:** Their product has a rechargeable battery which is more convenient than our product, where the user will have to replace the battery pack.

Bluetooth Transmitter and Receiver All in 1 Wireless Portable Audio Adapter: Cost $27.99 at Walmart. All in One Transceiver can be connected to a speaker to receive audio signal wirelessly. As a transmitter it can transfer the music from your mobile phone, TV or computer wirelessly. It can pair and connect to two devices simultaneously.

- **Advantage of our product:** Their product does not have a far range for bluetooth transmission, ours is 30 feet range.
- **Disadvantage:** Their product is portable and a smaller device, as ours is not.

JBL Flip 4: Cost $56.99 at Harman site. JBL Flip 4 is a portable bluetooth speaker powered by a rechargeable battery. Built in noise and echo cancelling speakerphone for conference calls. Can wirelessly connect with other JBL speakers to amplify more sound.

- **Advantages of our product**: Our product can allow for users to use the 3.5 mm jack in case they do not have a bluetooth device.
- **Disadvantages of our product:** Their product has a frequency response of 20 KHZ with more bass. Our product is not able to have bass or at the same frequency.

-written by Zohaib

2.3 * Development Environment and Tools

Software:

The project consist of using a Arduino UNO as the microcontroller for this project due to its 16 MHZ processor, low power usage, and user friendly capabilities. We used the open source programming software from the official Arduino website. This allows us to use Windows and Mac to be able to interact with the Uno. Furthermore, we used Github which allowed us to be able to upload, change, and download the code with ease. Similarly, we used Cloud 9 which is an online cloud based integrated development environment which supports multiple programming languages. Our primary programming language was C code. Lastly, to debug our code we used Visual Micro Serial Debugger for Arduino which allowed us to see the values of the variables, allowed us to update variables, and place breakpoints.

Hardware:

The testing equipment we used consisted of a multimeter and an oscilloscope. We used a multimeter to see if the voltages values we were obtaining after each component due to each component needing 3.3 or 5 volts. Furthermore, the oscilloscope is used to see the analog signal and how it is manipulated after each component. Moreover, a solder gun is required to solder specific components such as the speaker and wires. Lastly, we implemented a wooden box to keep our system inside to increase its aesthetics and security.

-written by Zohaib

2.4 * Related Documents and Supporting Materials

- UART protocol using the specification stated by IEEE.
- SPI protocol using the specification stated by IEEE.

2.5 * Definitions and Acronyms

- DAC - (Digital-to-Analog Converter) is a device that converts digital audio information, comprised of series of 0s and 1s, into an analog audio signal that can be sent to a headphone amp.
- ADC - (Analog-to-Digital converter) is a system that converts an analog signal, such as a sound picked up by a microphone or light entering a digital camera, into a digital signal. An ADC also provides an isolated measurement such as an electronic device that converts an input analog voltage or current to a digital number representing the magnitude of the voltage or current.
- 8 Bit Resistor Ladder - is an electrical circuit made from repeating units of resistors. Converts a parallel digital data to analog voltages.
- Register Programming - manipulating registers (small set of data) to do a specific task.
- Low Pass Filter - Is a filter that passes signals with a frequency lower than a selected cutoff frequency and attenuates signals with frequencies higher than the cutoff frequency. The exact frequency response of the filter depends on the filter design.
- Analog/ Digital Signals - An analog signal is a continuous wave that changes over a time period. A digital signal is a discrete wave that carries information in binary form. An analog signal is represented by a sine wave. A digital signal is represented by square waves.
- Power Amplifier - An audio power amplifier (or power amp) is an electronic amplifier that amplifies low-power electronic audio signals such as the signal from radio receiver or electric guitar pickup to a level that is high enough for driving loudspeakers or headphones.
- AT commands - “ATtention commands”.A series of machine instructions used to activate features on the bluetooth modules..
- RX/TX- (Transmit and Receive) The Tx level is the power in decibels per milliwatt (dBm) at which a modem transmits its signal. The Rx level is the power in dBm of the received signal.

  -written by Zohaib

* Design Considerations

3.1 * Realistic Constraints

- 16 MHZ clock frequency and AVR takes 13 clock cycles per conversion. This limits ADC sampling rate because ADC uses successive approximation type.
- 8 Bit R2R Resistor Ladder - a resolution of 256 (2^8) different voltage levels between 0 and 5v. This limits the resolution of the analog signal.
- Obstruction blocking the bluetooth modules will result in connection problems. Moreover, transmission range is around 9 meters.
- Power: 3.3 volts. @ 8GHZ = .004 milliamps, resulting in > 1 watt. Resulting in a limited amount of power.
- Loss of resolution above a sample rate of 15 kHz due to the sampling rate being faster than the signal takes to be stabilized.
- Price cost is around $120, which is not an optimal prices for a consumer product.
- Sampling rate at 77 KHZ, therefore, signals higher than 40 KHZ will cause aliasing.
- The highest sampling rate is limited to 16 MHZ /1 prescaler /13 cycles.
- Lack of ram limits sample buffer size to just a fraction of a second.

-written by Zohaib

3.2 System Environment and External Interfaces

The system's hardware will interact with the user's desired input. The input can be a pc that can play sound, for example a phone, pc, and/or mic.

The systems' software will interact with the user's input signal. This means that the software will capture the analog signal and be successfully output the signal back to the user.

-written by Zohaib

3.3 * Industry Standards(zk)

- UART - IEEE 1451.0, Accession # 16375333, Date: October 10, 2016 : This standard describes for asynchronous serial communication in which the data format and transmission speeds are configurable. Our system use Tx and Rx to send/receive data amongst each other. UART limits to 1 byte at a time, so having a higher baud rate will be more optimal for audio signals.

- Standard for Graphic Symbols - IEEE 315-1975 Date: September 4, 1975 : This standards shows specific symbols that represent specific electrical components, flows, voltage source, etc.Our design complies to this standard by using the industry standard symbols in our schematics. This affected our schematic by using specific symbols to represent the system.
- SPI - IEEE, Accession #10459541 Date: March 26, 2018: This standard describes the protocol of SPI. The standards describes how to implement SPI and define a way to communicate between digital devices. Our old design implemented SPI control registers declared in the standard that has influence on SCLK, SSn, MISO,MOSI. SPI with the atmega328 can transfer about 10000 byte in 16ms. This limits our byte transfer abilities .
- ISO/IEC 9899: Programming Language C. Dec 1999: This standard describes syntax, constraints, representation of input and out data, and semantic rules of C programming. Our code uses syntax such as #include and 0xFF that ISO expressed as industry standard. The limitation of this standard restrict us to only use C. If you java for example, we will not be under the ISO C programming industry standard.

-written by Zohaib

3.4 * Knowledge and Skills

Zohaib Khan

- EE 128 - Signal Processing
- EE 120B - Embedded Systems
- EE 100A & B - Electrical Circuits

New Skills -

- Knowledge and implementation of ADC/DAC
- Real Time Signal Processing
- Register Programming (ADC, UART, SPI, TIMER)
- Creating electrical circuits such as lowpass, power amplifier, op amps, 8 bit dac.

Rishi Kodukula

- EE100A, EE100B, EE110A, EE110B, EE120A, EE120B
- EE 115, EE128, EE 141
- ENGR180W
- CS10, CS13, CS61

New Skills -

- Knowledge of ADC
- Understanding a datasheet to gain relevant information to incorporate it to an existing system

- Soldering without supervision
- Soldering a wire directly to a part vs standard pins sticking out
- Who to ask if I need to have a smaller component soldered

Karen Escareno -

- CS122A

New Skills -

- Using and understanding UART with HC-05 Bluetooth Module
- Register Programming (Timer) for HC-05 .
- Adding single components to an already built working circuit.

3.5 * Budget and Cost Analysis

Arduino (x3)	$35.58 ($11.86 each)
HC-05 Module Bluetooth (x2)	$16.98 ($8.49 each)
Audio AUX Cable (x2)	$14.00 ($7.00 each)
Assorted Resistors	$10.99
Female Panel Mount Headphone Jack	$4.99
10k potentiometer	$1.25
3.5 mm jack	$3.50
Assortment of Capacitors and PnPs	$11.99
TSS9IN op ams (x3)	$7.50 ($2.50 each)
Breadboard	$5.00
jumper wires	$8.49

Speaker	$6.00
Wood	$20.00
Breakout Board	$10
D2A converter chip	$20
Total:	$176.27

-witten by Rishi

3.6 * Safety

Our Safety protocols were to be in a clean flat area with no food or water that could fall on our equipment. Since we are dealing with low voltages we followed safety precautions that would avoid shock or fire. As well we never used broken or old equipment to insure a safe environment. Also we avoided overcrowding of the breadboard or serial experiments within the same frame. We also made sure we used the appropriate amount of voltage and current through out our trials. We took necessary precautions to make sure there were no exposed wires while working. When soldering we followed safety protocol in using the correct instruments and making sure the surface had time to cool. Overall we made sure that our environment did not have outside factors that could potentially create a problem when working on our project.

-written by Zohaib

3.7 Performance, Security, Quality, Reliability, Aesthetics etc.

We were able to work independently on each component of this project, but if needed we would work in the SAE room in Chung Hall. When working in that room we could help each other out and could collaborate face to face versus the usual email or text message delay. We also meet up during lab time once a week and could pass information knowing that for sure the other two would be there.

Security was not really important given the scale of this project. The door to most lab rooms including the SAE room was locked and those with card access could enter, These meant that only students of the EE/CE departments could enter and if any others were interested they would have to request access. A flaw arises if someone knocks at the door, due to common courtesy we let them in. This is not detrimental to our work since we don't have classified information pertaining to the project.

Quality Control was a issue since there was a chance a specific part could fail leading to a set back. To counteract this we ordered extra parts and would just replace it with a different part was not working. This would allow us to continue working with the new part but not have to worry about replanning a completely different part.

The system plays music when a input is plugged into it. The audio can come from any source as long it connects to the headphone jack, given that is the only form of input the system can take to have a proper conversion.

The system is placed inside a black wooden box with the input and output cables easily accessible. The output has a 3.5 mm jack output so standard headphones or a circular speakers that would be able to allow more than one person to listen. The input consisted of either a mic that had wires connected to the breadboard or a 3.5 mm female jack so we could use a double male aux cable to connect the system to a computer or phone using the common aux cable.

-written by Rishi

3.8 * Documentation

All code of all team members can be found here :

https://drive.google.com/drive/folders/1bWz7cZo7g91ITM_Y1xnWkes_KrQ4up9p?usp=sharing

Zohaib Khan

The documentation I used to safety keep block diagrams, circuit designs, and code was mainly through google drive. Furthermore, after completing this project, the documentation of this project will be uploaded to github for safe keeping. Furthermore, I carried a senior design notepad where I put my ideas, diagrams, and implementation ideas that would be useful for the project.

My responsibilities/ design objectives : https://www.youtube.com/watch?v=dQPvpXkmBOw

Rishi Kodukula

I placed most of my material pertaining to the project here in a drive, and I had us all saved in a folder marked as such. All my progress reports are PDFs that I submitted and all information related to my project. I think the PDF I used to look at the datasheet is not included

but that is public knowledge on the manufacturer's website. I annotated a page that was my idea of how the chip would work wired up, but testing was fruitless due to usb related error.

Karen Escareno

The documentation I used to keep everything together was google drive since it is more reliable. It was easy and reliable since it was able to save folder of arduino code since our code was always changing.

My responsibilities/ design objectives : https://youtu.be/Y9HJFjz9YLc

3.9 Risks and Volatile Areas

Some risky areas is related towards the consumers input device. The gain in our device is around 50, but depending on the device the consumer uses to plug the 3.5 mm jack into, the output sound will vary. For example, a computer will output sound louder than a phone can due to its manufacturing designs. To mitigate risk, we chose the value of 50 because the sound is around 60 decibels and the max it plays from a phone is around 40 decibels.

-written by Zohaib

* Experiment Design and Feasibility Study

4.1 Experiment Design

Zohaib Khan

Objective: To produce a desired analog signal with a 8 - Bit R2R resistor ladder

Setup: With the use of software, generate a signal and send it to the Dac

Procedure: Hook up the Dac to PORTD and use a oscilloscope to see the result. One positive side to the signal output of the DAC and negative to ground

Expected Result: Plus or minus 5 percent of expected signal.

Objective To understand how low pass filter work and how they manipulate a signal.

Setup: Using a matlab script and downloading the Signal Processing Toolbox, we can generate a signal with the desired cutoff and see how the desired signal will look like.

Procedure: Run the script with the correct parameters such as the sampling rate, cutoff, etc

Expected Results: Pass signals of 8 KHZ and lower

Objective: To see if the built in ADC in the atmega328p is sufficient enough to sample an audio signal with minimal aliasing.

Setup: Register program ADC and use a potentiometer connected to 5 volts, ground, and A0

Procedure: To run the code and open serial monitor to see the results

Expected Results: Values corresponding on the turn of the potentiometer will increase or decrease values on the serial monitor

Karen Escareno -

Objective: Bluetooth should send data at a fast speed.

Setup: One Bluetooth module connected to Arduino pins 10,11, ground and 5V. Another Bluetooth connected to the same pins on a seperate arduino.

Procedure: Run code. Use Serial.write and Serial.read to see the char outputs of the bluetooths. Use the Serial monitors to see the outputs of both arduinos.

Expected Results: The outputs should be the same for both arduinos. Using timestamps the values (0-255) should be close in time for the master and slave.

4.2 * Experiment Results, Data Analysis and Feasibility

Zohaib Khan

Expected Result: Plus or minus 5 percent of expected signal.

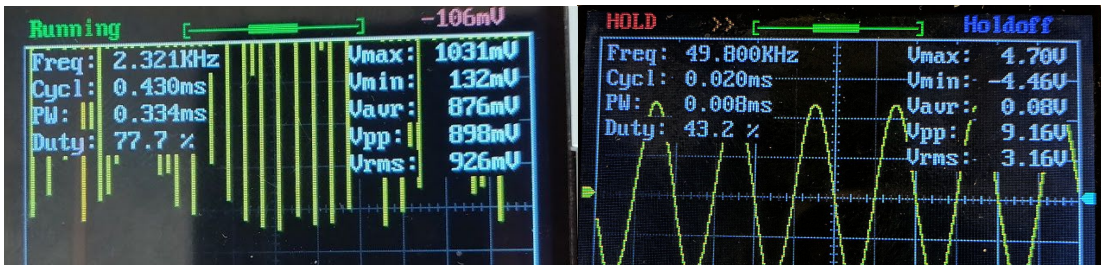


Figure 4.2.1 - Oscilloscope of the output of the DAC

```
void twoKILOhertz(){
 for (int t=0;t<100;t++){
    PORTD = sine[t];  // Outputs to PORTS 0-7 of the 8 Bit DAC
    delayMicroseconds(.0005);
 }
 }
//period = (duration of each step) * (number of steps)
//period = .05us * 100 = 5us = 0.00005s
//so the frequency should be:
//frequency of ramp = 1/0.00005s = 20000 Hz

 void onehundredhertz(){
   for (int t=0;t<100;t++){
    PORTD = sine[t];
    delayMicroseconds(100);
   }
 }

 void threehundredhertz(){
 for (int t=0;t<100;t++){
    PORTD = sine[t];
    delayMicroseconds(33);
 }
```

Figure 4.2.2 -Arduino Code for a digital signal

As seen in **figures 4.2,** I was able to generate various signals from 100 Hz to signals of upwards to 50 KHZ. The importance of the experiment was to see if an 8 bit dac can produce a signal of such amplitude. The project's objective was to resemble signals of value around 20 KHZ due to most audio from a pc is such value. This means that the R2R resistor ladder is capable of producing analog signals with various HZ required for the system. Moreover, a higher

bit DAC can produce better solution however, the 8 bit DAC is enough for our system and can be produced very cheaply thus being better for consumers.

Expected Results: Pass signals of 8 KHZ and lower

```
% Parameteres Required
values = fdesign.lowpass('Fp,Fst,Ap,Ast',8000,5,0.01,80,48000);
signalFilter = design(values,'equiripple');
output = filter(singalFilter,input);
fvtool(signalFilter)
plot(psd(spectrum.periodogram,output,'Fs',48000))

% Sample Rate      : 48 kHz
% Passband Edge    : 8 kHz
% 3-dB Point       : 8.5843 kHz
% 6-dB Point       : 8.7553 kHz
% Stopband Edge    : 9.64 kHz
% Passband Ripple  : 0.01 dB
% Stopband Atten.  : 79.9981 dB
% Transition Width : 1.64 kHz
```

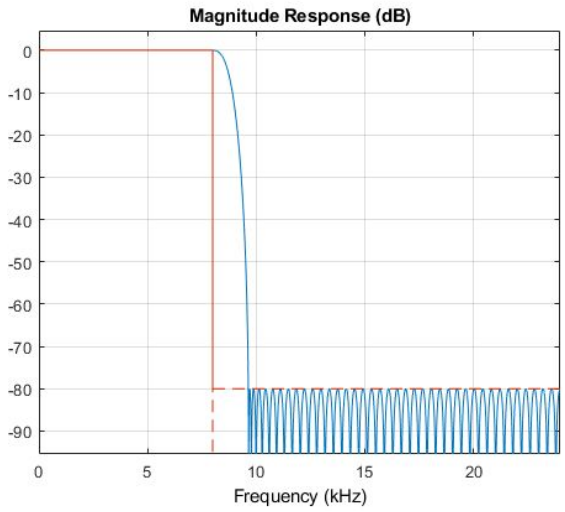


Figure 4.3 - Representation of a low pass filter through matlab

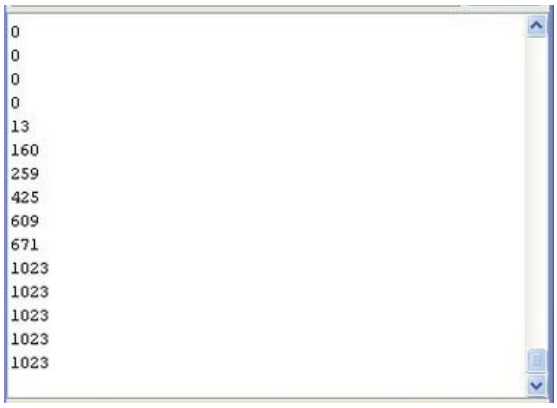
As seen in **figure 4.3,** using the signal processing package through matlab, I was able to generate a signal with a specific characteristics such as the sampling rate of 48 KHZ and a frequency required of 8 KHZ. This is important to our design because understanding what filter is needed for the system is vitale due to having a lot of unwanted noise from specific components. Using high pass filters, IIR filters, and many more, we decided using a low pass filter to it can reduce the unwanted noise and keep the signal within 40 KHZ as stated in the objectives. A low pass filter passes signals with a frequencies lower than the selected cutoff frequency. In conclusion, a low pass filter will be a good fit for our audio system.

Expected Results: Values corresponding on the turn of the potentiometer will increase or decrease values on the serial monitor

```
void ADC_init(){
  ADMUX = (1<<REFS0);                                 //Turns ADC on
  ADCSRA |= (1 << ADEN) | (1 << ADSC)| (1 << ADATE);  //Continously converting
}

uint16_t adc_read(uint8_t ch)
{
  ADC = (ADCL |(ADCH << 8));  //Reads ADC data register
  return ADC;
}

void setup() {
  ADC_init();
```



Figure

4.4 - ADC Register Programming Results

As seen in figure 4.4, I was able to successfully program ADC registers. Furthermore, the results are significant because the built in ADC signal can apporately convert analog signals to digital signals and an external ADC module is not required for the system. Furthermore, the built in ADC can have a sampling rate of 76 KHZ which is optimal for our design specifications. This means that we can manipulate the built in 10 bit ADC to sample quicker and do a shift bit to allow 8 bit to 8 bit transfer. In calculations the adc will be sufficient to carry out our design objectives and tasks stated above.

Karen Escareno:

Expected Result: Char values are sent from one Arduino to another. Values are 8 bits (0-255) and can be seen on both serial monitors.

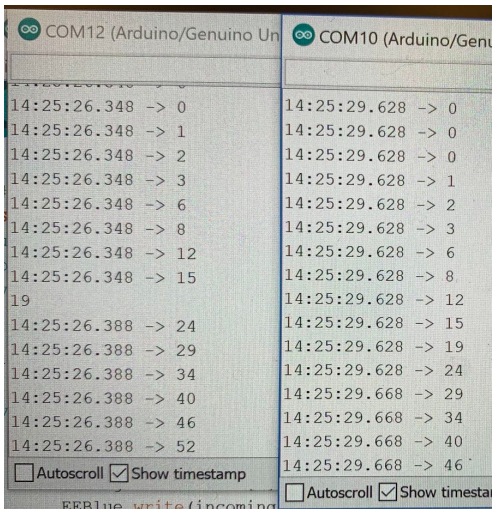


Figure 4.5 Bluetooth modules sending chars

The results show that there is a connection between the master and the slave bluetooth. It also shows that the values are being received 3 seconds after and it is receiving the right values that it is sending. It also confirms that the values we are getting from the ADC are the same values that are being outputted to the DAC.

* Architecture and High Level Design

5.1 * System Architecture and Design

Zohaib Khan

Hardware -

Figure 5.2 points out the interaction between each hardware component. The 3.5 mm jack will accept a analog signal from the user through any desired pc. From there, the coupling circuit will shift the DC value due to Arduino not able to accept negative voltages. The Arduino will accept the analog signal and output a digital signal to the 8 bit R2R resistor ladder. The DAC will send an analog signal to the low pass filter which will reduce the unwanted background noise. Moreover, the signal will going into a signal amplitude will allows the user to increase the amplitude of the analog signal and the power amplifier will increase the current allowing the signal to have a louder output. Lastly, the signal will go into a speaker or headphone jack that will output the corresponding noise that the user inputted in the beginning of the circuit.

Software -

Figure 5.3 points out the interaction the ADC has with the user. The user will input a signal in which the ADC will convert this signal and store it into the variable. The ADC was done through register programming due to the default ADC function not properly working. By calling the register to their desired states and calling a ADC timer to interrupt while these actions are taking place. Once the ADC converted the analog signal into a digital signal, it was stored into a variable to allow the Hc-05 bluetooth module to obtain those values and send those values to the other module.

Karen Escareno

Hardware-

Figure 5.2 points out the interaction the bluetooth modules have with the system. There are two Hc-05 Bluetooth modules, one that sends the data and one that receives the data. The transmitter bluetooth module will receive a digitized signal from the ADC module, which it will send the the receiver. The receiver will obtain those values and output them to the DAC where the signal will be ready to be heard on the speaker.

Software -

Figure 5.3 shows that the bluetooth module is taking the data that it receives from the ADC and it writes it to the master bluetooth. The master bluetooth then sends it to the slave module. The slave module reads the data it has received and outputs to PORTD which is where the DAC is.

5.2 * Hardware Architecture

Responsibilities -

Zohaib Khan - In charge of the 3.5mm jack, coupling circuit, 8- Bit R2R Ladder-DAC, Signal amplitude, Power Amplifier, DC offset, Low Pass Filter, and Output.

Karen Escareno - In charge of Bluetooth.

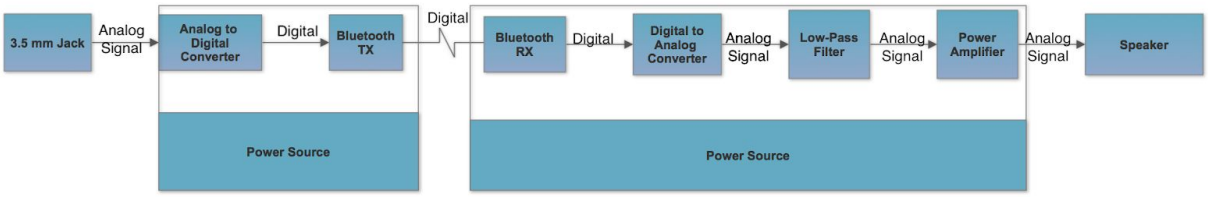


Figure 5.2 - High Level Block Diagram of Hardware

5.3 * Software Architecture (only required if your design includes software)

Responsibilities -

Zohaib Khan - ADC & Output

Karen Escareno - Bluetooth

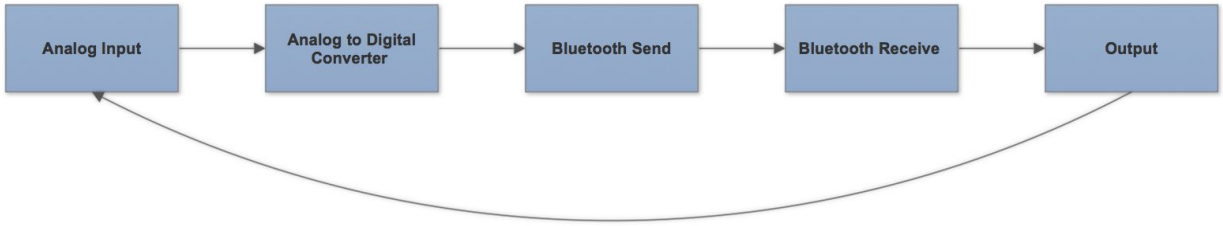


Figure 5.3 - High Level Block Diagram of Software

5.4 * Rationale and Alternatives

We used this architecture because it was the most simplest way to produce quality results for our system. Moreover, each module we created, we created from scratch thus allowing us to understand each component much better. Software wise, the architecture we used is needed to create the sampling speed we needed and also allowed us to choose specific values of specific registers to do tasks such as sampling and timer functions. The modules that took the input, digitized it , and output the signal is needed in any audio system thus we have the architecture we have.

The 3.5mm jack was the input we decided to use because it is more of a common consumer product compared to the RCA jack. It is more commonly used by consumers since they would be able to play audio from their phones, computer, etc. The approach of using the built in ADC was because of timing issues but was convenient because we were able to.

Given not a single component could perform DAC we started with a component from DigiKey that was marketed to do this. The Arduino, a part that I had access, could pull this off, but the project would gain a 3rd unit leading to less work that the of the system and another part using a system that already existed, unfortunately, this this not work, thus we kept the 8 bit DAC we used.

6. Data Structures (include if used)

6.1 Internal software data structure

N/A

6.2 Global data structure

N/A

6.3 Temporary data structure

N/A

6.4 Database descriptions

N/A

7 * Low Level Design Hardware:

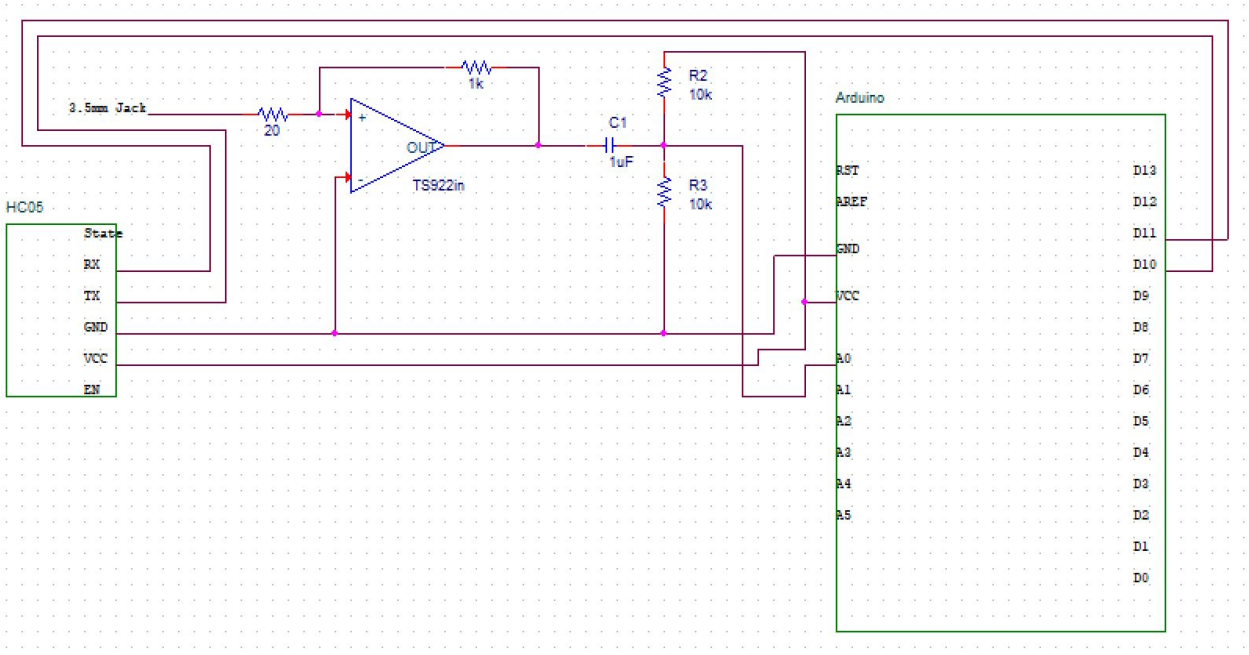


Figure 7 - Hardware Schematic of Transmitter

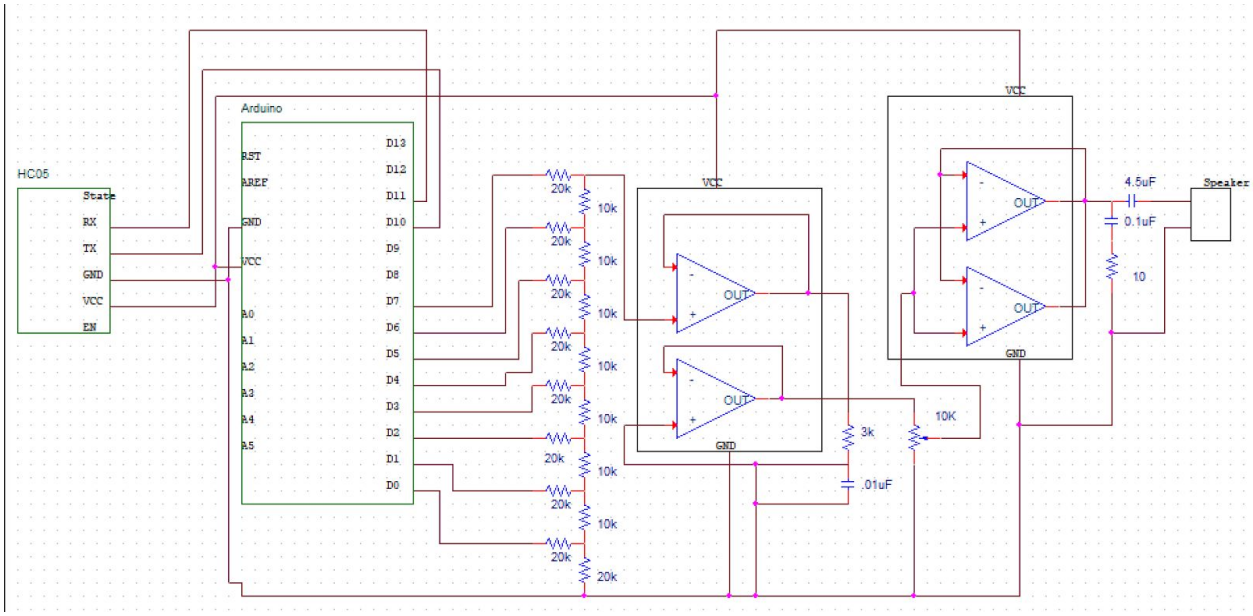


Figure 7 - Hardware Schematic of Receiver

Software :

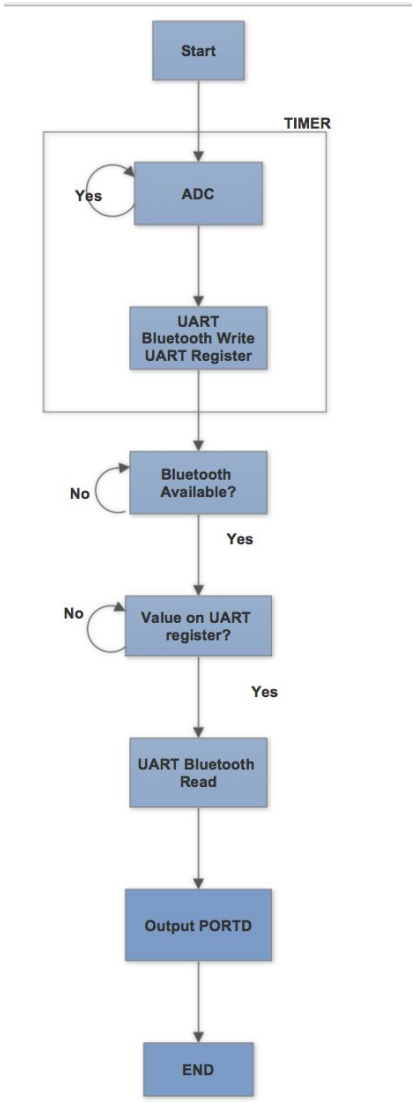


Figure 7 - Software System Flow Chart

Zohaib Khan -

Module of the 3.5 mm jack

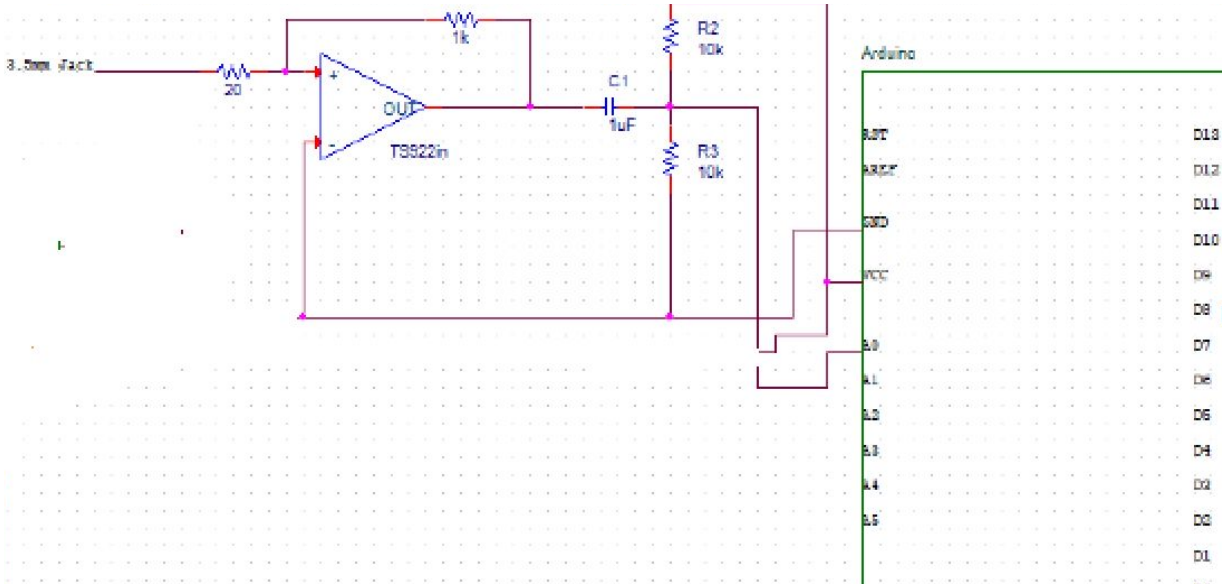


Figure 7.1 - Hardware schematic of the Input module

7.1.1 Processing narrative for *module 3.5mm jack* This module was constructed due to the arduino unable to handle negative voltages. When I took the 3.5 mm jack and connected it to analog in of the arduino, the values outputting to the serial monitor were unexpected and did not match when the signal increased or decreased.

7.1.2 ***Module 3.5mm jack*** **interface description**

This module interacts with the ADC timer function which converts the analog signal to a digital signal. Moreover, this allows the user to plug a pc signal into our system that allows the signal to be transmitted and outputted.

7.1.3 ***Module 3.5 mm jack*** **processing details**

This module is created by an op-amp and an AC coupling circuit is required The op-amp is the TS922IN which the gain of the op amp can be calculated with R2/R1. R2 is equal to 1000 ohms and R1 was 20 ohms which resulted in a gain of 50. Specifically, with the TS922IN, a 20 ohms is connected to pin 2 and the 1000 ohm resistor is connected to pin 2 and pin 1 (output pin). Moreover, VCC+ and VCC- is connected to the power supply and ground respectively. The op- amp is connected to the coupling circuit. The coupling circuit uses a 1uf capacitor, and two

10k resistors to ground and power respectively to shifts up the D.C. value to half the supply rail with resistors. Then to stop the output device shorting out this bias we had to use a capacitor. The limitation of the op- amp is the max current is 80 mA and an operating voltage of 10 volts. Furthermore, they produce low noise around 9 nV/√ Hz, which is optimal for an audio system.

**Module of the R2R Resistor Ladder (DAC)*

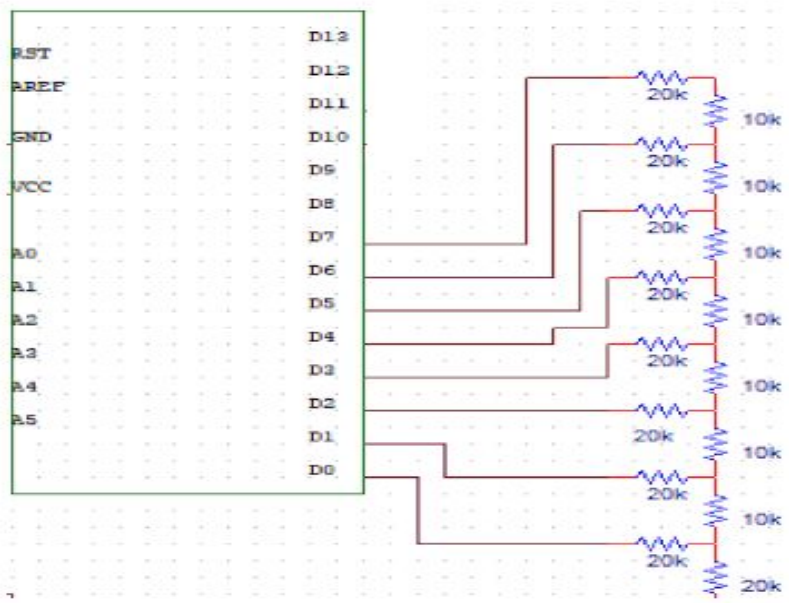


Figure 7.2 - Hardware schematic of the Input module

7.2.1 Processing narrative for *module R2R Ladder*

This module was constructed due to the arduino not having a built in digital to analog converter. Thus, an external DAC was needed to changed the digital signal to an analog signal. Moreover, we used a R2R ladder because we were able to make it from scratch, they can be made very cheaply, and can produce the results we were looking for as mentioned in the feasibility study above.

7.2.2 Module R2R Ladder Interface description

This module interacts with the output and the arduino. The arduino sends a digitized signal from the ADC to the DAC. The analog signal from the corresponding digital signal is outputted to the output circuit where the analog circuit is passed through a low pass filter, power amplifier, and able to be outputted.

7.2.3 *Module R2R Ladder* processing details

This module is created by using nine 20K resistors and seven 10K resistors. To create a R2R resistor ladder, I used PORTD of the arduino which are pins 0-7 and connected 20K resistors on to each pin. Furthermore, added 10K resistors in series. The resistors connected to the arduino pins in parallel should be the value of 2R. That's why I used 10K and 2(10K) = 20K resistors. The 8- bit R2R resistor ladder can produce values from 0 to 255 on a scale of 0 to 5 volts. This means that the DAC has a step of 5 volts / 255 = .019 volts per step. We can figure the corresponding value by multiplying an input of 100 which corresponds to 100 * .019 = 1.9 volts. Because this is an 8 bit DAC, it can not produce a resolution as a 16 bit DAC can which can allow the system to sound less smoother.

***Module of the Output**

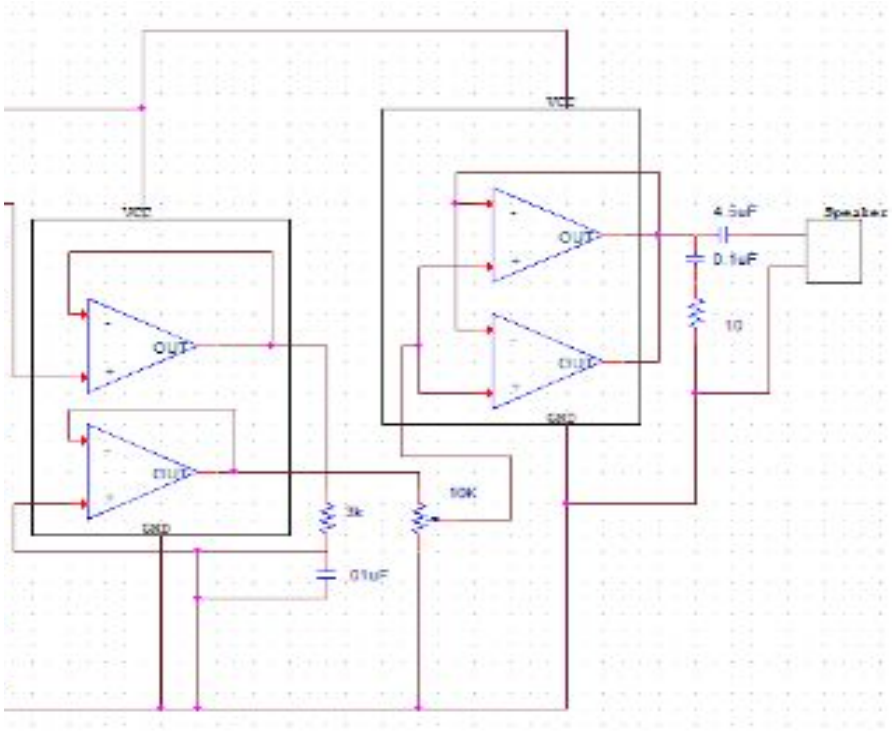


Figure 7.3 - Hardware schematic of the Output module

7.3.1 Processing narrative for *module Output*

This module was constructed due to the DAC producing very distorted signal when hooking up the speaker directly. The DAC was able to produce the corresponding analog signal, however, the signal I listened to was very manipulated from the original. Thus a low pass filter, power amplifier, and buffer are required to get rid of unwanted noise and increase the original analog signal we were trying to hear.

7.3.2 *Module Output Interface* description

This module interacts with the R2R ladder. The R2R ladder outputs the analog signal to the output module where the analog signal is manipulated to give the user a better sound quality.

7.3.3 *Module Output* processing details

This module is consisted of a few electrical components. First, the analog signal will go into a buffer to keep the original signal from being distorted. This is done by using a rail to rail dual op amp (TSS29IN) as a voltage flower. The output pin of the R2R ladder is connected to pin 3. pin 3 is connected to pin 4 and pin 4 is connected to the output. The buffer's output is connected to a low pass filter. The low pass filter is designed as any RC circuit. With a resistor and capacitors that are connected in series to ground. The values corresponding to the resistor is 1K ohm and the capacitor is 4.5 nF. The signal from the low pass filter is connected to the second voltage follower of the same op amp due to the T9SSIN being a dual op amp.This means that the output from the low pass filter is connected to pin 3, pin 3 is connected to pin 2, and pin 2 is connected to output. Then we can connect a 10k potentiometer that will allow the amplitude to be varied by turning the knob. The 10K potentiometer is connected to power and ground respectively. The output from the 2nd voltage follower is connected to the middle pin of the potentiometer that will allow one to vary the signal from 0 to 2.5 volts. Since we are using two TS922IN, we can use them to increase the current by 160 mA of current due to one op amp increasing the current by 80 mA. From the output of the potentiometer, the output is connected to the non-inverting inputs of both op amps. Lastly, by implementing a capacitor with the value of 200 uF, we can create a DC offset. The offset allows the signal to be oscillating from 0 volts as typically seen with an analog signal in most audio systems The output of the op amp is connected to the capacitor and the final filtered analog signal goes into a speaker or headphone jack that allows the user to hear the corresponding sound.

**Module of the ADC*

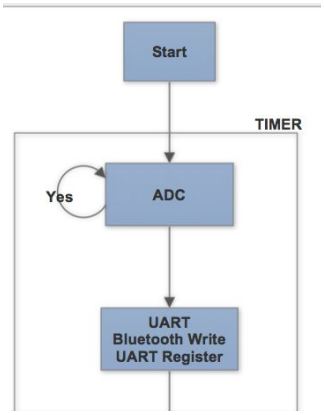


Figure 7.4 - Flow Diagram of the ADC

7.4.1 Processing narrative for *module ADC*

This module was constructed due to the analogread function of the Arduino having a default prescaler of 128. This means that the default sampling rate is 16 MHZ/128/13. We needed around 80 KHZ sampling rate due to most audio through a pc is around 35 KHZ.Moreover, the ADC is 10 bit while our DAC is 8 bit. This means we need to do a but shift so we get 8 bits for both the ADC and DAC. Therefore, to accomplish this goal we had to program the adc registers.

7.4.2 Module ADC Interface description

This module interacts is the bridge between the input and output. It takes in an analog signal from the user, converts the analog signal to digital signal and sends that to the output.

7.4.3 Module ADC processing details

```
void ADCfunction(){
 ADCSRA = 0x00;             // clear ADCSRA register
 ADCSRB = 0x00;             // clear ADCSRB register
 // A0 as Input
 // Left Allign
 // set reference voltage
 ADMUX |= (0 & 0x07)|(1 << REFS0)|(1 << ADLAR);
 // Sampling rate = 76.9 KHZ
 // auto trigger
 // enable ADC, interrupts, and adc measurements
 ADCSRA |= (1 << ADPS2)|(1 << ADATE)|(1 << ADIE)|(1 << ADEN)|(1 << ADSC);
}
void setup(){
 DDRD = 0xFF;   // Sets PORTD as a output
 ADCfunction(); // inits ADC registers
}
```

Figure 7.4.3 - ADC Register Declarations

As seen in figure 7.4.3, this module is programmed to have a 76.9 KHZ sampling rate and continuous trigger as the analog signal comes in. This is achieved by having a prescaler of 16. 16 MHZ of the cpu speed is divided by the prescaler si 16MHZ/16 = 1. 1 is divided by 13 bacuse in contounis mode, the atmega takers 13 cycles per conversion. Therefore, 1/ 13 = 76.9 KHZ. Moreover, ADCSRA register enables the adc conversions, interrupts, and enables the adc. The ADMUX register declares A0 as an input which is called by ADMUX || = (0 & 0x07). Similraly, ADMUX || = (1 << REFS0) enables reference voltage and lastly enabling the ADLAR register does a shift bit. The shift is due to the ADC being 10 bit and the DAC being 8 bit. If we shift the ADC we can get better results because they both will output 8 bit results. With these configurations set up, we can output to the DAC by calling PORTD = ADCH; without the HC-05 bluetooth module and it will output the sound really well on the speak. Limitations of this ADC is that it is 10 bit. 10 bit adc can have values of 2^10. This means that an external adc module of 16 bit can produce better resolution of the analog signal.

**Module of the Bluetooth (Software)*

7.5.1 Processing narrative for *module Bluetooth*

The software module for the bluetooth was made so that we could get information from the bluetooth module and be able to write and read from the UART register that is shared between the two bluetooth modules.

7.5.2 Module Bluetooth Interface description

We set the pins to 10 and 11 to be able to read the write from the bluetooth module. The bluetooth writes the char values to a UART register. The slave bluetooth would then read the values from the UART register and would then output to PORTD.

7.5.3 Module Bluetooth processing details

```
#include <SoftwareSerial.h>
SoftwareSerial EEBlue(10, 11); // RX | TX

void setup()
{
    // start th serial communication with the host computer
    Serial.begin(9600);
    //Serial.println("Arduino with HC-05 is ready");

    // start communication with the HC-05 using 38400
    EEBlue.begin(38400);
    //Serial.println("BTserial started at 38400");
}

void loop()
{

     // Keep reading from HC-05 and send to Arduino Serial Monitor
    if (EEBlue.available())
    {
        Serial.write(EEBlue.read());
    }

    // Keep reading from Arduino Serial Monitor and send to HC-05
    if (Serial.available())
    {
        EEBlue.write(Serial.read());

    }

}
```

Figure 7.5.3-1 Bluetooth AT commands

```
#include <SoftwareSerial.h>
SoftwareSerial EEBlue(10, 11); // RX | TX

unsigned char incomingAudio = 0;

void setup(){
  //Serial.begin(115200);
  cli();
  TCCR0A = 0;// set entire TCCR1A register to 0
    TCCR0B = 0;// same for TCCR1B
    TCNT0  = 0;//initialize counter value to 0
    OCR0A = 249;
    // turn on CTC mode
    TCCR0A |= (1 << WGM01);
    // Set CS11 bit for 8 prescaler.
    TCCR0B |= (1 << CS11);
    // enable timer compare interrupt
    TIMSK0 |= (1 << OCIE0A);
    sei();
  EEBlue.begin(115200);  //Baud Rate for command Mode.
}

void loop(){

   }

ISR(TIMER0_COMPA_vect)
{
  incomingAudio = (analogRead(A0));//read voltage at A0
    EEBlue.write(incomingAudio>>2);
  //insert your code here that you want to run every time
```

```
#include <SoftwareSerial.h>
SoftwareSerial EEBlue(10, 11); // RX | TX
volatile unsigned char data = 0;
void setup() {
  // Serial.begin(9600);

    DDRD = 0xFF;
   EEBlue.begin(115200);
}

void loop() {
    if (EEBlue.available() > 0)
    {
        data = EEBlue.read();
        PORTD = data;
      //Serial.println(data);
    }
}
```

Figure 7.5.3-2 Master/Slave Bluetooth Code

We first connected the two arduinos using AT commands as seen in Figure 7.5.3-1. We used AT+Role to set our master and slave bluetooth. Our master had a role of 1 and the slave had a role of 0. We were able to find the address if the slave using AT+ADDR? and were able to connect the slave to the master using AT+BIND=(slave address). We had to make a timer for the ADC and for the write (Bluetoothname.write()) command for the first arduino so we could at first get a sampling rate of 8 kHz as seen in Figure 7.5.3-2. We had to give a prescaler of 8 and a compare match register of 249 to get 8 kHz. We used Timer 0 as our timer because it was less than a compare match register of 256. We then had a seperate arduino which had the slave bluetooth which would use read (Bluetoothname.read()) to get values from the master bluetooth which we would then output to PORTD.

**Module of the Bluetooth (Hardware)*

7.6.1 Processing narrative for *module Bluetooth*

The hardware for the bluetooth were the TX, RX, VCC and GND pins that we used. We were able to connect the bluetooths to the arduino which we were then able to communicate through code with.

7.6.2 Module Bluetooth Interface description

We set the pins to 10 and 11 to be our TX and RX to help communicate with the Arduino. We set the GND to ground on our arduino and VCC to our 5V. We established that the bluetooths were communicating with each other when their lights were in synch. The rest of the communication between them was done in software.

7.6.3 Module Bluetooth processing details

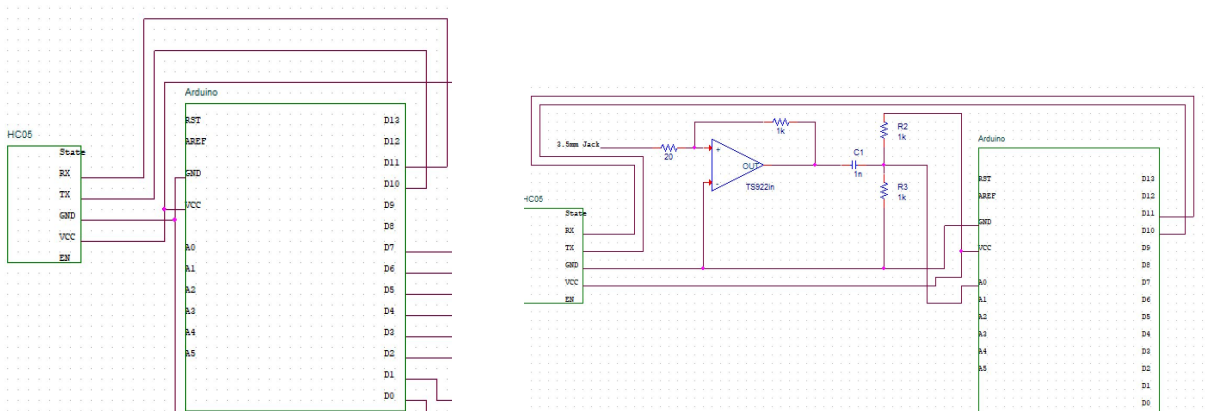


Figure 7.6.3 Master and Slave Bluetooth Connections

In figure 7.6.3 we are able to see how they are connected to the arduino. We were unable to connect the TX and RX of the bluetooth module to the Serial connections in pins 0 and 1 because it would have too many errors uploading code and communication between the two bluetooths. It would cause the bluetooth to receive incorrect information and would not like the code to be uploaded. What we did was we changed the pins to 10 and 11. We connected the TX of the bluetooth to the RX in the arduino and vice versa. It worked because when we were using the AT commands it would blink at a slower pace and hold the light for a few seconds but when it was connected to another bluetooth module they would both blink the light at the same pace.

8* Technical Problem Solving

Zohaib -

8.1 SPI Registering Programming

```
#define Address_Value 0x60
// MCP4725A0 address is 0x60 or 0x61
// MCP4725A1 address is 0x62 or 0x63
// MCP4725A2 address is 0x64 or 0x65
// MCP4725A3 address is 0x66 or 0x67

void I2CInitFunction(void){
    TWSR = 0x00; // No needed presclaer
    TWBR = 0x0C; // 400 KHz
    //enable TWI
    TWCR = (1<<TWEN);
}

void I2CWhileCompleteFunction(void){
    // loops until TWCR and TWINT is Set
    while ((TWCR & (1<<TWINT)) == 0);
}

void I2CStartFunction(void){
    // TWSTA set 1 transmit start condition
    // TWINT set 1 to clear TWINT Flag
    TWCR = (1<<TWINT)|(1<<TWSTA)|(1<<TWEN);
    I2CWhileCompleteFunction();
}

void I2CStopFunction(void){
    // Last byte has been sent and trasnfer is ended
    // Can use Stop condition or a Repeated Start Condition
    TWCR = (1<<TWINT)|(1<<TWSTO)|(1<<TWEN);
```

```
CSendValueFunction(uint8_t val){
In order to enter MT mode, SLA+W must betransmitted.
This is done by writing SLA+W to TWDR
Thereafter the TWINT bit should be cleared (by writing it to one) to continue the transfer.
When SLA+W has been successfully transmitted, a data packet should be transmitted
This isdone by writing the data byte to TWDR.
TWDR must only be written when TWINT is high. If not,the access will be discarded
R = val;
R = (1<<TWINT)|(1<<TWEN);
WhileCompleteFunction();


sorValue = 0;
tup() {
itlazies I2C
itFunction();


op() {
rValue = analogRead(A0);
ad to Transmit
artFunction();
nds with #define adress value
ndValueFunction(Address_Value);
: Register ( c0 = 1, c1 = 1, c2 = 0)
ndValueFunction(0b01100000);
ndValueFunction(sensorValue);
opFunction();
```

Figure 8.2.1 SPI code implementation

Figure 8.2.2 - Result of SPI Implementation

The R2R Ladder we are using is 8 bit dac. However, an 8 bit dac does not have the resolution of a 12 bit dac. This is because a 8 bit dac has values of 2^8 while 12 bit dac has values up to 2^12. Therefore, we had an external dac, the MCP4725 Breakout Board - 12-Bit DAC w/I2C interface. As seen in figures 8.2, I was able to successful program SPI without the bluetooth. However, with the addition to the HC-05 bluetooth module, we couldn't successfully implement the DAC into our system. This is because when we were calling a timer interrupt, it would not successful send data.We think that this was due to the I2C function not being able to be called correctly in the timer interrupt.The timer function was being called quicker than the I2C function, therefore it caused interference with the data. Thus, we stuck with the 8 bit DAC due to its flexibility and easy implementation.

8.2 UART Register Programming

```
#define prescaler 51 //According to datasheet, this value gives 9600 baudrate @ 8Mhz
void USART_Init(void) {
    UBRR0 = prescaler;      /*Set baud rate */
    UCSR0A = 0x00;
    UCSR0B = (1 << RXEN0) | (1 << TXEN0); /*Enable receiver and transmitter */
    UCSR0C = 0x06; /* Set frame format: 8data, 1stop bit */
}
void USART_transmit( unsigned char data ) {
  //Wait for empty transmit buffer
  while( !( UCSR0A & (1 << UDRE0)) );
  //Put data into buffer, sends the data
  UDR0 = data;
}
unsigned char USART_Receive(void){
  /* Wait for data to be received */
  while ( !(UCSR0A & (1<<RXC0)) ) {};
  /* Get and return received data from buffer */
  return UDR0;
}
```

```
int led = 6;
void setup(){
    USART_Init();
     pinMode(led, OUTPUT);
}
void loop(){
 unsigned char start_value = 0x08;
      USART_transmit(start_value);
       unsigned char start_val = USART_Receive();
      if (start_val < 0x09){
        digitalWrite(led, HIGH);
      }
      else {
        digitalWrite(led, LOW);
      }
}
```

Figure 8.3 - UART Implementation

As seen in figure 8.3, I was able to successfully program UART. The circuit for testing this implementation was one arduino and an led. The led was a output and the rx and tx pins of the arduino were connected. With a simple arduino code, i sent a char value of 0x08 and recieved it. Once received, I compared the received values with lower/higher values to see the value i was obtaining was correct. The program worked successfully and I was able to send accurate values using the tx and rx pins of the arduino. Unfortunately, the bluetooth module was having trouble sending this code once I gave it to my partner to implement to the bluetooth module. This is due to the bluetooth modules sending a value that differed from what they were expecting. For example, they expected the value of 200, however the value was around 300. They believe that this is due to the bluetooth module change some values internally before sending it.

Rishi -

8.3 A2D conversion using I2C

I was working on this component by myself. I had the part but could not find a way to have the chip solely convert the data. I wanted it so a 3rd computer is not needed to have the entire system to work. I marked important on the datasheet for the least amount of connections that the chip needed in order to function but after a few weeks works of pseudo testing, I was told to pivot to have the conversation over Arduino.

8.4 A2D conversion using Arduino

This problem might have been a underlying reason why 8.2 did not work as well. After connecting an Arduino to my computer and running any form of complex code my computer, the usb part would disconnect within the software. I knew of this problem existed at the end of December 2018 since it showed up near the end of my 128 labs, leading to all code to be ran on my partners computer for the last few labs and project. Given time has passed and I cleaned my computer to free up more move, I assumed that the problem was fixed. The failure here on even the basic code, makes me think that the usb ports on my computer are faulty. It is not practical for me to buy a new laptop for a single project.

Karen Escareno -

8.5 Bluetooth/ADC Timer

The Figure 8.5 has the code we used for the Bluetooth and ADC timer. The problem with the bluetooth communication is that since the ADC was not going fast enough we were having trouble getting the decent audio since the ADC was not going fast enough. With the ADC going faster we had to make the write function for the bluetooth also go faster. What we did was we set a timer interrupt for the ADC and bluetooth write function so that it would have a sampling rate of 8 kHz. This would help speed up the ADC so our audio would be better and we would get values from the ADC at a much quicker rate and output them quicker to the slave bluetooth. It helped us get decent audio compared to what we were getting before the timer.

9 User Interaction Features

9.1 User Interaction with the System

In terms of our audio system, the individual can interact with the input of the system. For example, the user can decide what type of pc they are going to use to play their music from. For example, the user can choose a laptop, tv, mp3, and phone and plug the 3.5 mm jack to use in the audio system. Moreover, a mic perfectly works with out system, and will take the user's voice and output the sound from the speaker. Moreover, the pointemeter acts like a volume up/down button where the user can turn off the sound when the individual rotates the potentiometer to the left. This means the user is in control of the "volume" and as well as the input device they decide on

9.2 User Interface Screens

N/A

10* Test Plan

10.1 * Test Design

Zohaib

Test Case 1 - ADC Sampling Rate of 80KHZ

Objective: To have the ADC sampling at a rate of 80 KHZ to be able to produce a signal of 0- 40KHZ(most audio ranges between those values)

Setup: Run a sampling rate code & look into the atmega328p datasheet to find the equation to calculate sampling rate. For the equation to calculate sampling rate which is (cpu/prescaler/13 cycles).

Procedure and how data is collected: Run the script & do the calculation. The CPU speed is 16 MHZ and the prescaler with a value of 2 to the power value. (1,2,4,8,16,32,64)

Expected results: The expected results is 76.9 KHZ

Test Case 2 - Accuracy of the system

Objective: The accuracy of the output signal is 95% compared to the input signal.

Setup: Have the system created. Make sure the input and output modules work correctly.

Procedure and how data is collected: Connect the 3.5 mm jack to a pc and play a signal with a specific hertz.(200 HZ to 20KHZ) Then use an oscilloscope to see the output signal.

Expected results: The expected results will be within 5 percent error corresponding to the input signal. (200 to 20KHZ)

Test Case 3 - Input Frequency

Objective: To make sure the Arduino can read analog values up to 40KHZ

Setup: Have the 3.5 mm jack properly connected and have an oscilloscope on hand.

Procedure and how data is collected: Play a sound with different values and with an oscilloscope, to see the input frequencies of the 3.5 mm jack.

Expected results: The expected results will be that the 3.5 mm jack will have the corresponding value as the input.

Test Case 4 - Response Time

Objective: The response time of the circuit should be in microseconds.

Setup: Be sure to have a stopwatch, and a full functioning input and output of your circuit.

Procedure and how data is collected: Have a stopwatch at hand and start the stopwatch the second you place an audio signal into the input of the circuit. Once the audio is heard on the output you can stop the stopwatch.

Expected results: The expected results will be that the response time will be in microseconds which in this case the results using this testing method was also in microseconds.

Karen Escareno -

Test Case 5 - Bluetooth sampling rate of 8kHz

Objective: To have an 8kHz for the ADC and write function for the bluetooth so we can get a quicker output from the ADC and write to the bluetooth.

Setup: The setup used for determining the sampling rate was by using timer interrupt equations to get the desired 8kHz.

Procedure and how data is collected: Using the timer speed we wanted of 2M and so that we would get a prescaler of 8. We would then use that the interrupt frequency of 8 kHz which would give us our compare match register of 249 to ensure that we have an interrupt of 8 kHz.

Expected results: Our expected results are that we have around 8 kHz for our interrupt.

Test Case 6 - Accuracy of circuits [Bluetooth module]

Objective: To have a high accuracy so that we get the same input as our output.

Setup: Have our two circuits, one for the master and one for the slave. Make sure that the input and the output are both working correctly.

Procedure and how data is collected: Connect an oscilloscope to the input of the circuit which is this case could be a 3.5mm jack or a microphone. Play a 200 Hz sine wave and make sure that the input plays a 200 Hz sine wave on the oscilloscope. Then connect an oscilloscope to the output of the circuit, it should have an output of a 200 Hz sine wave.

Expected results: It had a very low accuracy rate in that instead of a sine wave we would get a square wave of about 200-300 Hz.

Test Case 7 - Bluetooth Range

Objective: The range between the bluetooths should be between 10 to 15m.

Setup: Have our two circuits, one for the master and one for the slave. Make sure that the bluetooths are connected and are sending the correct data from the master to the slave using the serial monitor output.

Procedure and how data is collected: Play sound from the input, with both arduinos 1m apart. Make sure that the output is what is expected. Start to increase the distance by 1m until the audio is no longer what you expected.

Expected results: The expected results is that the range between the two bluetooth is about 10 to 15m.

Rishi-

Test case 8 - Testing the CPC5751 Chip

After soldering it up, I checked the input pins and placed wires for import points of the chip and placed them in destinations based on other parts that needed for it to run. I marked the left power rail for input power supply, and the right power rail for right power supply. I needed to take into account the chance they were not the same voltage and current if the chip was in use. I even got wires of different colors and had the connection on the board plugged in and had the color wire marked on paper where it would be connected to. I could not collect any data due to an error with my computer not seeing the Arduino when connected. I ran into a hardware problem with the arduino. My computer would not register it an input. I tested the usb input in all 3 of my usable usb ports but the program would not send even the simplest of code. The most effective way to fix this problem would be to send my computer to a shop or buy a brand new one. But would be

unreasonable since a shop would take a week or more to look for a problem, get the parts for replacement, and return it to me. Buying a new computer would increase the budget for no good reason and would be over doing it if it was a very small problem.

10.2 * Bug Tracking

A database will be used to track defects found while performing the test cases. All defects will be logged as they are discovered. Defects will be assigned to Person A to fix, or to Person B to investigate.

10.3 * Quality Control

Rishi

We knew going in that the bluetooth parts had a chance of failure so we ordered spares allowing us to be able to test when one chip does not work. The chips that were defective were not placed with the ones we have not tested so there was no cross contamination. It was economical since buying in bulk of 6 units was much better than buying a single one and ordering and waiting for shipping before the replacement to show up.

I also bought a few extra boards for soldering incase my first attempt connected pins that should not he touching. This was close to happen when I was soldering and the metal connected two pins together. The IEEE room had proper equipment so I was able to remove the excess solder so the two pins would not touch. If this failed I had spare boards and parts to restart from the beginning. This would have had a 3-4 day delay to get the new part soldered to the new breakout board.

10.4 * Identification of critical components

- 3.5mm jack - This is a critical component in that it is the input of the entire circuit. We need to pay attention to this during testing because based on the audio we choose the frequency of the signal will never be the same since all audio signals are different.
- Output - It is a critical component in that we are looking to get the right audio output. The point of the project is to reproduce the input audio to the output.
- Bluetooth - this is a critical component because if it was not sending or receiving the right values then the entire output would be wrong.

10.5 * Items Not Tested by the Experiments

- Op Amps - used in the output were not tested, in the datasheet they were said to have an 80mA power amplifier which was assumed from the data sheet and not tested.

11 * Test Report

11.1 * Test 1 - ADC Sampling Rate

Zohaib

Test Results: With a prescaler of 32, the sampling rate became 38.4 Khz

Comparison with expected results: The sampling rate acquired is less than have the sampling rate needed to be able to produce good quality sound.

Analysis of Test Results: The reason we got 38.4 khz is due to the prescaler we chose. Using a prescaler of 32 would yield results of (16MHZ/32/13 cycle) = 38.4 Khz

Corrective Actions Taken: By changing the prescaler value to 16, we are expecting our sampling speed to be 78.6 Khz

Test 1.2

Test Results: With a prescaler of 16, the sampling rate became 78.6 Khz

Comparison with expected results: The sampling rate is the closest we can get to using the prescaler given. For example using a small prescaler will give us values less that 30khz and prescaler higher than 16 will double the sampling rate but also increase the aliasing. Therefore. 78.6 Khz is the most optimal configuration we can develop.

Analysis of Test Results: The test result obtained can be obtained by calculating sampling speed which is (16MHz/16/13) = 78.6Khz.

Corrective Actions Taken: No corrective actions taken. Objective was successfully obtained.

11.2 * Test 2 - Accuracy

Zohaib

Test Results: Signal sent was 200 Hz. The output Result was 284 Hz

Comparison with expected results: The result was much higher than expected due to the unwanted noise in the background.

Analysis of Test Results: I calculated the percent error to be 42% (284-200/200)*100

Corrective Actions Taken:Due to a lot of unwanted background noise we implemented a low pass filter to reduce the extra unwanted noise of the circuit.

2.1

Test Results: Signal sent was 3 KHZ and the output result was oscillating from 2.940 to 3.020 Khz

Comparison with expected results: We were able to get a solid output result. Although the result was not identical we got as close as we can.

Analysis of Test Results: The percent error was calculated to (2,940 Hz - 3000 Hz)/3000 Hz *100 is about 2 percent.

Corrective Actions Taken: No action were taken due to completing the task.

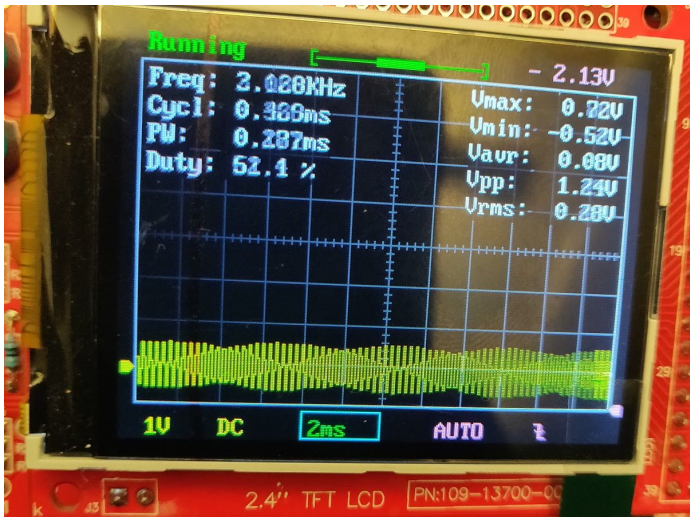


Figure 11.2.1 - Sine wave signal

11.3 * Test 3 - Input Frequency

Zohaib

Test Results: The input frequency was between 200 Hz and 40 kHz depending on the audio signal that was inputted to the circuit.

Comparison with expected results: The expected results were that the input signal frequency would vary between 200 Hz and 40 kHz since all different types of audio have a different frequency.

Analysis of Test Results: The input frequencies would vary based on the different types of audio that were played into the input of the circuit. Using an oscilloscope it was determined that the values were from 200 Hz to 40 kHz.

Corrective Actions Taken: No action was taken because the desired results were obtained.

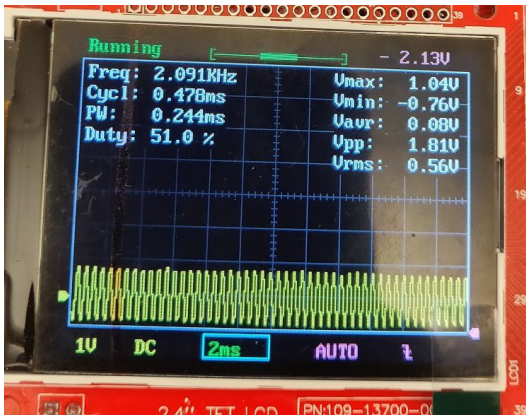


Figure 11.3.1 - Input frequency

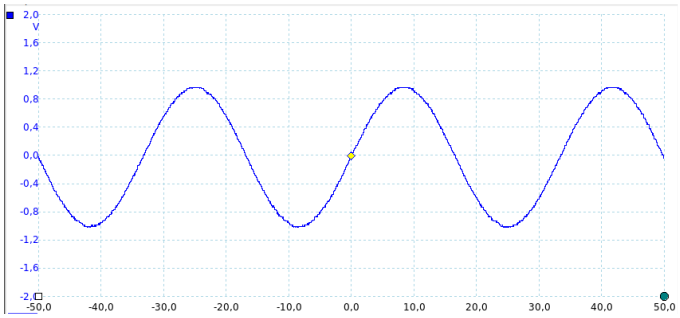


Figure 11.3.2 - Input frequency of 30 Khz

11.4 * Test 4 - Response Time

Zohaib

Test Results: The response time of the circuit was in microseconds. From the second the audio went into the input of the circuit to the output it was only a few microseconds.

Comparison with expected results: The expected results for the input audio to be outputted was expected to be in microseconds which was the case in these test cases using a stopwatch.

Analysis of Test Results: The time it took to get for the input to the output was in microseconds which is what was the expected results.

Corrective Actions Taken: No further action was taken since the desired response time was met.

Figure 11.4.1 - Response Time

11.5 * Test 5 - Bluetooth Sampling

Karen

11.5.1

Test Results: We obtained a 2 kHz interrupt to speed up the ADC and bluetooth commands.

Comparison with expected results: The expected results was that the audio was not as good as we expected it to be.

Analysis of Test Results: The interrupt was at such a small value and was not going as quick as we expected it to be. We also needed a much larger sampling rate because the sampling rate being used was not taking in as much audio as we expected.

Corrective Actions Taken: We decided to increase the sampling rate and increase the baud rate of our bluetooth modules.

11.5.2

Test Results: To get a interrupt every 125ms we obtained a sampling rate of 8 kHz.

Comparison with expected results: We were able to read in audio that has no voices. We were able to hear the beat of the audio which is more than we heard in the previous test case.

Analysis of Test Results: With such a small sampling rate it was hard to hear voices but we were able to detect audio that was around 4 kHz. With a much higher sampling rate there would have been errors that would have affected our data and output.

Corrective Actions Taken: We were not able to increase the sampling rate to due the baud rate of the bluetooth modules and how it would cause error when the baud rate and sampling rate were increased.

11.6 * Test 6 - Accuracy of circuit [Bluetooth]

Karen

11.6.1

Test Results: For the accuracy we obtained a very low accuracy due to the sampling rate being so low.

Comparison with expected results: We were suppose to get an output of a sine wave but instead we got a square wave which had to do with the delay of it having such a low sampling rate.

Analysis of Test Results: By increasing the sampling we would be able to fix the delay and would increase the accuracy so the signal we got had a frequency of about 200-300 Hz.

Corrective Actions Taken: Increasing the sampling rate so there would not be a delay on the ADC and on the bluetooth write function.

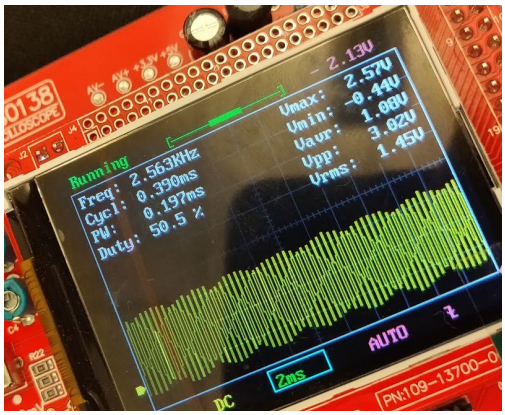


Figure 11.6.1 - Sine wave signal

11.7 * Test 7 - Bluetooth range

Karen

11.7.1

Test Results: We were able to go from 10-15m since it was never a constant value.

Comparison with expected results: The expected results were that it would have a range between the bluetooths of about 10m which was true in our case but at time would go as far as 15m.

Analysis of Test Results: The values varied because of the surroundings. When it detected 10m compared to 15m they were in different surrounding which may have affected the range between the two bluetooth.

Corrective Actions Taken: There is not action to be taken since it was what we expected the range to be in the first place.

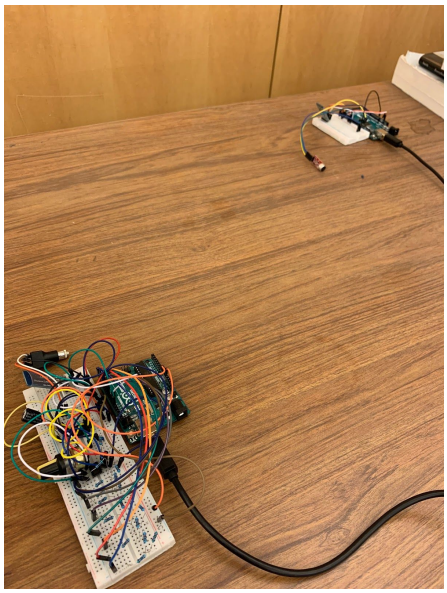


Figure 11.7.1 - Bluetooth Range

12* Conclusion and Future Work

12.1 * Conclusion

The Bluetooth Audio Transmitter is a wireless speaker that takes sound from a normal audio jack. The output is the music that was selected and due to the bluetooth connection the speaker can be placed away from the device. The project is split into 3 major parts:analog to digital conversion, bluetooth Sending/receiving, and digital to analog conversion. The system requires an audio signal that will be digitized by the transmitter before being sent to the receiver to output the sound. The intended applications for this project is to connect any form of sound through a 3.5mm jack and be able to hear the corresponding real time audio.

The purpose of this project was to simulate and create a foundation to develop life long skills we will continue to use. The importance of this project is to further understand and become

familiarized with digital signal processing, register programming,sampling rates,sampling analog and digital signals, aliasing & quantization, debugging, peer communication and collaboration.

Out system meet the overall project goal because the purpose of our project was to create and understand all concepts of an audio playback device. Specifically, our technical design objectives included range: 20 Ft +, sampling rate: 40KHZ - 80KHZ, accuracy: 5 % error, response time: microseconds, and input frequency: 100 HZ to 40 KHZ. some individuals had their ups and downs in their contribution and meeting technical design objectives.

Zohaib -

I was able to successfully complete all of my design objectives which included sampling rate, response time, input frequency, and accuracy. My responsibilities included creating a working system without the bluetooth module, so my partners can add the bluetooth module to my working module. Furthermore, through hardware and software I was able to create the input,dac, and output modules stated above which ran through an adc timer interrupt. This senior design project had many hurdles and obstacles that continued throughout the project, however without them we as a team would not have learned so much. This project allowed us to learn how to effectively problem solve, regardless how difficult they were in the beginning. These lessons were important as they are important skills to learn as engineers. In the project I was encouraged to continue the process and using all the resources that I had to resolve the problems that emerged. Throughout the years as students we are given instructions, tools, and resources to complete a project or lab, however in this project this was all up to us as a team to figure out. The value of this is that it helped to create a careful design plan, record our progress, and find our resources. This careful planning is what contributes to lifelong skill that can be used as a student and as an engineer in the work field.

The project was a new experience as a student to use everything we learned as students and put it all in one project. The use of having leadership skills within a team was also a lifelong lesson that was different from any other. The skills learned allowed for the project to succeed as well as recognize all the valuable resources that technology has offered.

Rishi -

The original goal I was talked to do was cover everything pertaining to DAC. I would have the stream from the bluetooth and I had to reconstruct it to an analog signal in completely. I was not able to complete the part I set off to due do the Arduino software did not register the input of the Arduino when plugging into my computer. On the hardware side of things I mapped out how my part would fit with the system and had information on how it would interact with the rest of the project by marking the parts that would be wired to the different power sources and other components the chip wanted based on the datasheet.

I also gained information on soldering by myself and know how to use a few of the basic techniques based on the parts provided: nodes that make mountains, having the wire connected straight to another metal, and resources and equipment to have a smaller chip be integrated into a system using a breakout board.

Karen Escareno -

Overall the original goal was completed in that we should be able to take in an analog signal which would go over bluetooth and then would go back to analog and sent to the output. For the code that I used with the bluetooth and timer that I created I was able to get this done but the audio is not a good as it should be because of the limitations of 115,200 baud rate which was only good for 8 kHz of audio. I was able to get all this done with a song with just a beat since it was able to detect highs and lows of a song and output to the user.

It failed in the way that it was not able to detect voices but just beats of a song which is due to a small sampling rate because of the bluetooth modules. What I learned is that we could have increased the baud rate to 1M but it could have led to many errors which such a high baud rate for the bluetooth module. What I learned from the project is how to use UART and how it works. I was also able to learn register programming for the arduino so that we would speed the ADC and also the bluetooth write function. I was also able to learn how to add single components to a built circuit and fix any errors that can come up with it. I was able to learn how to work with others and use my learned engineering skills in this project.

12.2 Future Work

The impact of this work is to show how to create a working audio system with efficient yet cheap products. For example, during the research stage, we found that many individuals believed that we can not produce any quality sound with a 8 bit adc and R2R ladder Moreover, many individuals online said the arduino is not suitable for an application such as audio signal processing, but with endless hours of designing we were able to create a system many individuals wouldn't think would be possible to do with the components we used. The system can improve in sound quality by using external adc and dac. Preferably with audio, a 16 bit adc and dac are preferred to produce a finer resolution of the signal. In terms of marketability, a 3d printed exterior should be produced and moreover can connect over bluetooth rather than just a 3.5mm jack. This increases the enjoyment of the consumer which in turn, increases the value of the product. Although the main component are in our system, we think that the system can become more optimized. This means that one can find ways to decrease aliasing and background noise while increase the analog signal so there is not as much distortion for the listener. All in all,

we have set a great foundation of an audio signal playback device that our group and anyone who reads this report can replicate. And we hope to work on this system to improve its technical objectives so we can have a more fine tuned commercial product.
-written by Zohaib

12.3 * Acknowledgement

The individuals who provided us helpful feedback and critique with our project are:

Dr. Roman Chomko, Dept. of Electrical & Computer Engineering,
Shaoyi Peng, Dept. of Electrical & Computer Engineering
Manglai Zhou, Dept. of Electrical & Computer Engineering

13* References

[1] Arduino, "Wire Library", Arduino Reference, https://www.arduino.cc/en/reference/wire [Accessed January 20, 2019]

[2] GM Electronic, "HC-05 Bluetooth Module", https://www.gme.cz/data/attachments/dsh.772-148.1.pdf [Accessed October 28, 2018]

[3] GM Electronic, "Arduino with HC-05 (ZS-040) Bluetooth module – AT MODE" https://www.gme.cz/data/attachments/dsh.772-148.2.pdf [Accessed November 5, 2018]

[4]Mouser Electronics, "CPC5750U", www.mouser.com https://www.mouser.com/datasheet/2/205/CPC5750_R00F-1480157.pdf [Accessed Nov 15, 2018]

[5] STMicroelectronics, "TS922, TS922A Datasheet" https://www.st.com/resource/en/datasheet/ts922.pdf [Accessed January 10, 2019]

14* Appendices

*** Appendix A:** Parts List

- Arduino (x3)
- HC-05 Module Bluetooth (x2)
- Assorted Resistors
- Female Panel Mount Headphone Jack
- 10k potentiometer
- Assortment of Capacitors and PnPs
- TSS9IN op ams (x3)
- Breadboard
- jumper wires
- Speaker

*** Appendix B:** Equipment List (for testing)

- Oscilloscope - check if analog signal to make sure it behaves correctly
- Multimeter - to check make sure each component are getting their desired voltages and currents

*** Appendix C:** Software List

- Arduino Uno (https://www.arduino.cc/en/Main/Software)
- Matlab - Not needed but helpful to understand the characteristics signals (https://www.mathworks.com/downloads/)

Appendix D: Code

- https://drive.google.com/drive/folders/1bWz7cZo7g91ITM_Y1xnWkes_KrQ4up9p?usp=sharing

Appendix E: Vendors for Components

- Amazon
- Digikey
- Mouser
- Arrow