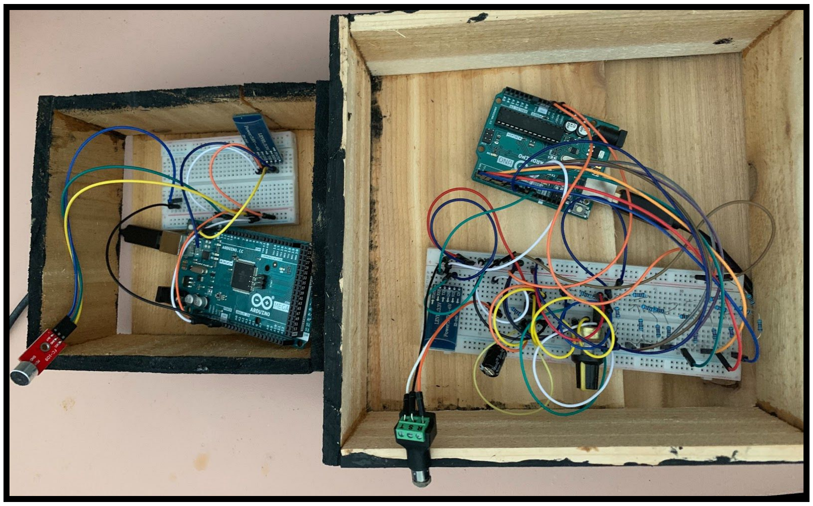
# EE175AB Final Report

# Bluetooth Audio Transmitter



**EE 175AB Final Report**

**Department of Electrical Engineering, UC Riverside**

| | |
|---|---|
| **Project Team Member(s)** | Zohaib Khan |
| **Date Submitted** | 3/18/2019 |
| **Section Professor** | Roman Chomko |
| **Revision** | Version 1 |
| **Permanent Emails of all team members** | Zohaib khan zkhan003@ucr.edu |

## 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, | 06/10//2019 | |

## Table of Contents

# 1 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 BAT's 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 Transmitters do not have a connected speaker and do not connect with all devices that lack Bluetooth functionality. This sets the BAT apart from the rest of devices that are similar. 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.

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 functions with just 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.

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 electronic 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 low risk of failure. When comparing the BAT to audio devices that are more expensive, the more expensive device can lead to complications with sound and become a high investment with a high risk of functionality.

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 to capture the analog signal, deconstruct it, and reconstruct then recreate the signal that was original inputted. Through all these challenges the team was able to keep the idea of a simplistic design, but efficient in quality.

## 2 Introduction

The application of the Bluetooth Audio Transmitter is to allow users to connect a PC to the BAT to produce a sound from afar. The concept of the BAT is different compared to other similar audio devices because this system produces high quality sound from cost efficient parts. The device can connect to any device with a 3.5 mm jack, and allows the user to change the volume using the interchangeable speaker. To accomplish these tasks, the system uses the principles of digital to analog converters, signal regulators, and signal processors.

The BAT is meaningful because it helps strengthen the team's understanding of real time processing and enhance skills such as planning and researching. Moreover, this device is used as a way to cater to both the new and old forms of technology. The combination of the 3.5 mm jack and bluetooth modules allows for more compatible devices. In the market space this project is beneficial, because to a larger audience which allows the device to stand out.

The project is related to electrical engineering because it uses a concept of digital signal processing to enable the signal to successfully be transferred. Both hardware and software is applied to this design's system of electronics. The use of both hard and software correlates to the concept of embedded systems. Embedded systems is a huge component to understanding electrical engineering due to its relationship between real time operating systems and electrical circuits.

The overall goal is to create a system that is a wireless speaker that takes sound from a 3.5mm audio jack. The project is divided into 3 key parts: analog to digital conversion, bluetooth communication, and digital to analog conversion. To accomplish this goal, the system uses a 3.5 mm jack as an input to receive the audio signal from the PC to the transmitter. The transmitter converts the analog signal to a digital signal. The Bluetooth module receives the digital signal and transmits the signal to the receiver. Then the receiver will reconvert the digital values to an analog signal. Lastly, the speaker will play the corresponding signal in real time. Through various obstacles, the accomplishment is to create a BAT to work properly, consistently, and efficiently.

**Design Objectives:**

- Range: 20 Ft
- Sampling Rate: 40 kHz - 80 kHz

- Accuracy: 5 % Error
- Response Time: Microseconds
- Input Frequency: 100 Hz - 40 kHz

**Responsibilities:**
Zohaib Khan
In charge of 8 bit digital to analog converter, power amplifier, and universal asynchronous receiver.

Karen Escareno
In charge of  connection, communication, and operation of bluetooth module

## 2.2 Backgrounds and Prior Art

HiGoing Bluetooth 5.0 Receiver:  The cost of the product is $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, microphone, audio outputs, or headphones[1].

- **Advantage:** The BAT comes with a wired speaker, while this product does not come with any audio device.
- **Disadvantage:** The competitor's product has a rechargeable battery which is more convenient than the BAT product, where the user will have to replace the battery pack.

**Bluetooth Transmitter and Receiver All in 1 Wireless Portable Audio Adapter:** The cost is $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:** The competitor's product does not have a far range for Bluetooth transmission, unlike the BAT is 30 feet range.
- **Disadvantage:** The competitor's product is portable and a smaller device than the BAT.

**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.

- **Advantage**: The product can allow for users to use the 3.5 mm jack in case they do not have a bluetooth device.

- **Disadvantage:** The JBL Flip 4 product has a frequency response of 20 kHz with more bass. The BAT is not able to have bass or at the same frequency.

## 2.3 Development Environment and Tools

**Software:**

The project consists of using a Arduino Uno as the microcontroller due to its 16 MHz processor, low power usage, and user friendly capabilities. The open source programming software from the official Arduino website enables compatibility between software and hardware. This allowed the team to use Windows and Mac to be able to interact with the Arduino. Furthermore, Github was used to upload, change, and download a code with ease. Similarly, Cloud 9 was used as an online cloud based integrated development environment which supports multiple programming languages. The primary programming language was C code. To debug the code, the system uses the Arduino's serial monitor to update and place breakpoints.

**Hardware:**

The testing equipment consist of a multimeter and an oscilloscope. The multimeter's purpose is to see if the voltages values of each component is between 3.3 and 5 volts because each component needs a voltage source of 3.3 or 5 volts. Furthermore, the oscilloscope is used to see the analog signal and how it is manipulated after each component of the system. Moreover, a solder gun is required to solder specific components such as the speaker and wires together. Lastly, a wooden box is implemented to keep the system inclosed to increase its aesthetics and security.

## 2.4 Related Documents and Supporting Materials

- Universal asynchronous receiver/transmitter protocol using the specification stated by IEEE.
- Serial peripheral interface 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.

- ***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.
- **R2R 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** - A series of machine instructions used to activate features on the bluetooth modules.

# 3 Design Considerations

## 3.1 Realistic Constraints

The system's realistic constraints impact the system's performance by reducing wireless range and limiting sound quality. The constraints can be categorized into three categories. The processing and sampling rate of the microcontroller, the wireless range of the Bluetooth modules, and the frequency range of the digital to analog converter.

Processor Speed -

The Arduino has a microcontroller that has a set clock frequency of 16 MHz. This limits the ADC sampling rate because 16 MHz is as fast as the Arduino can process information. The upper limit of the sampling rate is 1 MHz and the lower limit is 19 kHz.

Wireless Range -

The transmission range of the Bluetooth modules is 20 meters. Having the system a part for more than 20 meters will result in lost of connection. Similarly, the Bluetooth module use UART protocol which means that the modules can send information to each other as fast as 1.875 MBaud. However, this limits the wireless range because the further the system is from the user, the more latency.

Frequency Range -

The system uses a sampling rate of 77 kHz. This means that signals higher than 40 kHz will cause aliasing. Similarly, a loss of resolution above a sample rate of 15 kHz due to the sampling rate being faster than the signal takes to be stabilized. Lastly, the system uses a 8 bit resistor ladder which creates 256 values of different voltage levels. This means that the resolution of the analog signal will reach a peak of 256 and acquiring a higher quality signal will cause an error.

## 3.2 System Environment and External Interfaces

The system's hardware interacts 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 system's software interacts with the user's input signal. This means that the software captures the analog signal and successfully outputs the signal back to the user. The system uses UART protocol to send and receive information.

## 3.3 Industry Standards

[1] IEEE, "International Conference on Advanced Computing and Communication Systems." Coimbatore, India, 10-Oct-2016

[2] J. Alveredo, "CERT C Programming Language Secure Coding Standard." Carnegie Mellon University, Pittsburgh,Pennsylvania, 10-Sep-2007.

[3] "UM10204 I2C-bus specifications and user manual." NXP Semiconductors, Eindhoven, Netherlands, 14-April-2014.

These standards describe the asynchronous serial communication in which the data transmission speeds are configured. The system uses 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. Also, the standards show specific symbols that represent specific electrical components, flows, voltage source, etc. The design complies to this standard by using the industry standard symbols in the schematics. This affects the schematic by using specific symbols to represent the system.

## 3.4 Knowledge and Skills

**Zohaib Khan -**

Prior Knowledge -

- 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
- Creating electrical circuits such as lowpass, power amplifier, and op amps,

**Karen Escareno -**

Prior Knowledge -

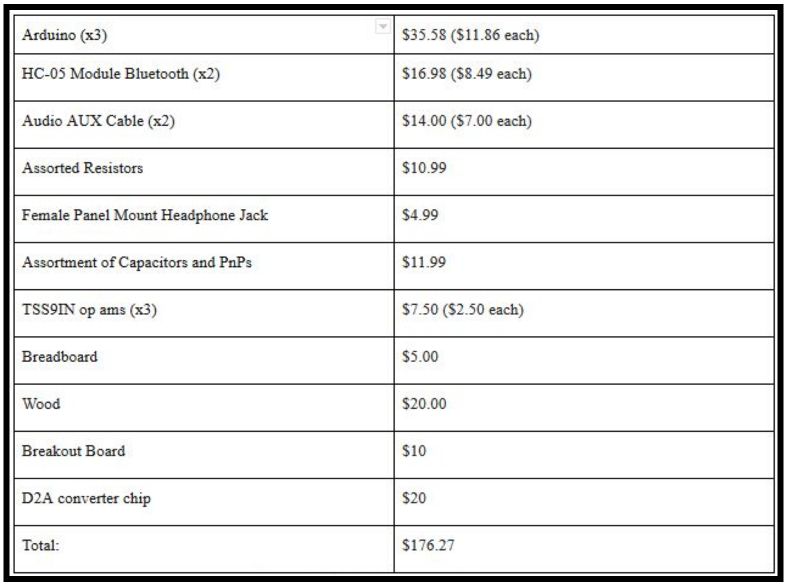
- CS122A

New Skills -

- Using and understanding UART with HC-05 Bluetooth Module
- Register programming for HC-05 .

## 3.5 Budget and Cost Analysis

The system uses very low cost components to produce high quality sound. The system uses two Arduinos, bluetooth modules, and various other electrical components that can be found in Amazon,  DigiKey, and Sparkfun. Table 1 shows the cost of each component and the total cost of the system.



| | |
|---|---|
| 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 |
| Assortment of Capacitors and PnPs | $11.99 |
| TSS9IN op ams (x3) | $7.50 ($2.50 each) |
| Breadboard | $5.00 |
| Wood | $20.00 |
| Breakout Board | $10 |
| D2A converter chip | $20 |
| Total: | $176.27 |

***Table 1*** *- Cost of Components*

## 3.6 Safety

Dealing with low voltages safety protocols are followed to avoid shock or fire. It is important that broken or old equipment are not used to insure a safe environment. Also overcrowding of the breadboard or serial experiments within the same frame are avoided for safety. The appropriate amount of voltage and current throughout the trials helped avoid fire. Necessary precautions are used to make sure there were no exposed wires while working. When soldering, safety protocol were used in using the correct instruments and making sure the surface had time to cool. Overall, the environment did not have outside factors that could potentially create a problem when working on the project.

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

To ensure the BAT is meeting quality and reliability expectations, there are many test trials the system has to pass to make sure the system runs as expected. These test are done in a weekly manner where the components of the circuit are tested to ensure the system is up to date.

The system plays music when an input is plugged into it. The audio can come from any source as long it is connected to the headphone jack, given that it 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 circular speakers will be able to allow more than one person to listen. The input consists of either a mic that has wires connected to the breadboard or a 3.5 mm female jack so it could use a double male aux cable to connect the system to a computer or phone using the common aux cable.

## 3.8 Documentation

The documentation used for 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. A senior design notepad was carried to put ideas, diagrams, and implementation ideas that would be useful for the project. Moreover, a copy of all progress reports are in PDF format and can be found in the Google Drive. The demo of the system is then uploaded to Youtube to show its functionalities.

## 3.9 Risks and Volatile Areas

Some risky areas are seen in the consumers input device. The sound gain depends on the consumer's device used to plug the 3.5 mm jack into, as output sounds vary. For example, a computer will play sound louder than a phone can because the computer can produce more current. To mitigate risk, the system uses a gain amp which increases the current by all input devices.

# 4 Experiment Design and Feasibility Study

## 4.1 Experiment Design

Objective: The objective of this test is to produce a 50 kHz sine wave with a resistor ladder. This will show if the resistor ladder is capable of playing audio with a frequency of 50 kHz.

Setup: Connect the Arduino to the phone. Second, search a 50 kHz audio file to play on the phone. Lastly, connect the oscilloscope to the power source and ground.

Procedure: The procedure is to play the 50 kHz audio file on the phone and observe the sine wave on the oscilloscope.

Expected Result: A sine wave with a frequency of 50 kHz is the expected result.

Objective: The objective of this test is to implement a low pass filter that allows signals of 8 kHz to pass to reduce static noise.

Setup: To set-up the test Matlab has to be open and the Signal Processing Toolbox is downloaded. The built in libraries allow Matlab to graph the behavior of a low pass filter.

Procedure: The procedure is to run the script and open the graph window to start the test.

Expected Results: The expected outcome is that the low pass filter will block signals higher than 8 kHz.

Objective: To see if the ADC is sufficient enough to sample an audio signal of 20 kHz with minimal aliasing.

Setup: To set-up the test connect a potentiometer to the power source, ground, and the A0 pin of the Arduino. Then turn on the ADC to run continuously by programming ADC registers.

Procedure: The procedure is to run the code and open the serial monitor to see the corresponding values of the potentiometer.

Expected Results: The expected results from the test is that the serial monitor will produce values from 0 to 1023. When the potentiometer is turned off the value will be 0, while fully turned on the value will be 1023.

Objective: The objective of this test is to send values from 0 to 255 with the Bluetooth to see if the modules are able to send data

Setup: To set-up the test one Bluetooth module connects to Arduino pins 10,11, and ground. Then another Bluetooth is connected to the same pins on a separate Arduino.

Procedure: To start the test the code has to run and use a serial monitors to see the output of the Arduinos.

Expected Results: The expected result in this test is the serial monitor will output the same value which represents the value being sent and received.

## 4.2 Experiment Results, Data Analysis and Feasibility

Expected Result: The expected result is that the oscilloscope will display a 50 kHz signal .

The system is able to generate various signals from 100 Hz to signals of upwards to 50 kHz. The importance of the experiment is to see if an 8 bit DAC can produce a signal of such frequency. The project's objective is to resemble signals from 20 kHz to 50 kHz compared to most audio from a PC is around 50 kHz. This means that the R2R resistor ladder is capable of producing analog signals with various hertz required for the system, as seen in Figure 4.2. Moreover, a higher bit DAC can produce better solution, however, the 8 bit DAC is enough for our system to produce good quality audio.

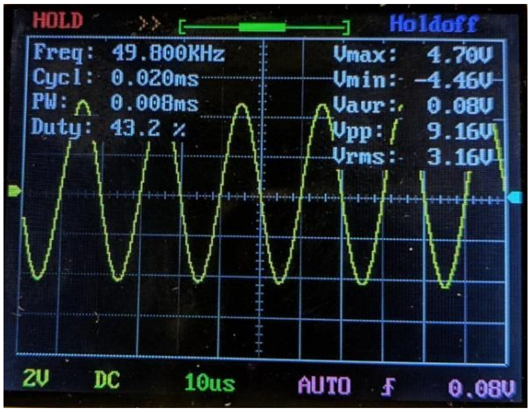
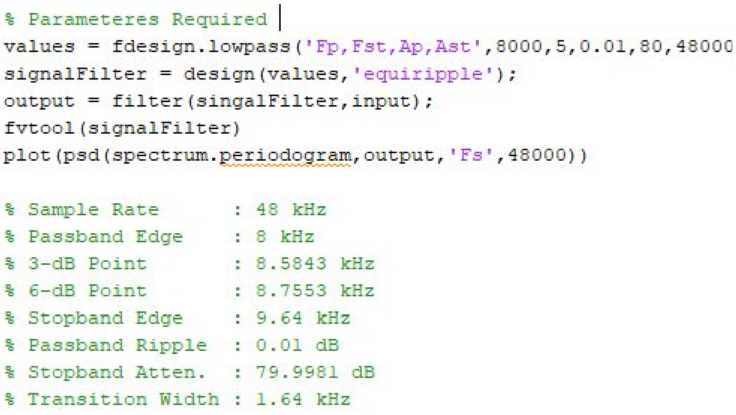


***Figure 4.2 -*** *Oscilloscope of the output of the DAC*

Expected Results: The expected result from this test is the low pass filter will reduce frequencies higher than 8 kHz.

Using the signal processing package through Matlab, the program is able to generate a low pass filter that has a sampling rate of 48 kHz. The low pass filter is designed to block any signals higher than 8 kHz as seen in Figure 4.3. The importance of this experiment is to use a low pass filter to reduce static noise from the audio. Using high pass filters, IIR filters, and many more, the system uses a low pass filter to reduce the unwanted noise and keep the signal within 40 kHz as stated in the objectives. In conclusion, the experiment shows that the low pass filter increases the quality of the audio.



```
% 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
```

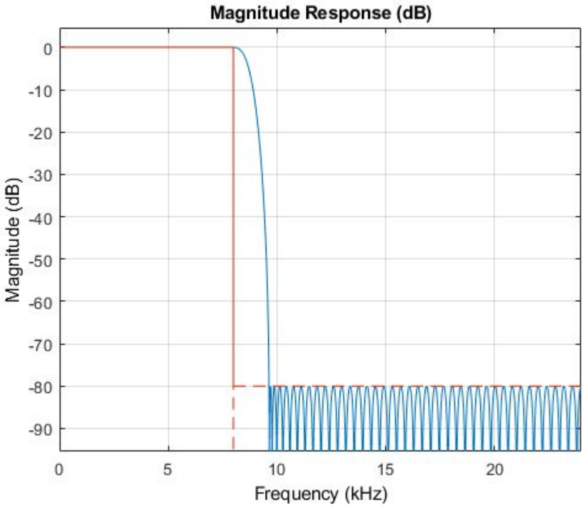
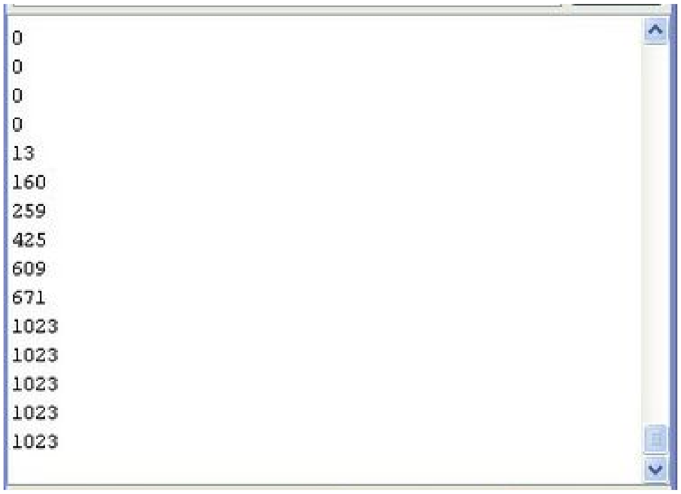


***Figure 4.3 -*** *Representation of a low pass filter through Matlab*

Expected Results: The expected result from this test is that when the potentiometer is off, the monitor will produce 0. On the other hand, when the potentiometer is on, the monitor will produce values from 1 to 1023.

The serial monitor outputs the corresponding values of the potentiometer. When the pointermeter is off, the value is 0. Similarly, turning the potentiometer on, produces values from 1 to 1023 as seen in Figure 4.4. The results are significant because the built in ADC signal can appropriately convert analog signals to digital signals and an external ADC module is not a requirement for the system. The built in ADC programs a sampling rate of 76 kHz, which is optimal for the system's design specification. In conclusion, the ADC is sufficient to produce high frequency audio signals.

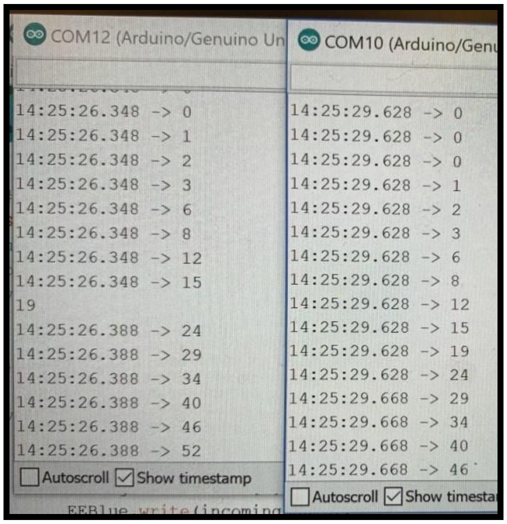


```
0
0
0
0
13
160
259
425
609
671
1023
1023
1023
1023
1023
```

***Figure 4.4 -*** *ADC results*

Expected Result: The expected results from this test is that the char values are sent from one Arduino to another. The Values are 8 bits (0-255) and can be seen on both serial monitors.

The results show that there is a connection between the master and the slave Bluetooth. The results also show when the values are received and when it is receiving to the slave Bluetooth it can be seen in Figure 4.5. The experiments show that the values of the system from the ADC are the same values that outputs to the DAC.



COM12 (Arduino/Genuino Un
□ Autoscroll ☑ Show timestamp
COM10 (Arduino/Genu
□ Autoscroll ☑ Show timesta

```
14:25:26.348 -> 0
14:25:26.348 -> 1
14:25:26.348 -> 2
14:25:26.348 -> 3
14:25:26.348 -> 6
14:25:26.348 -> 8
14:25:26.348 -> 12
14:25:26.348 -> 15
19
14:25:26.388 -> 24
14:25:26.388 -> 29
14:25:26.388 -> 34
14:25:26.388 -> 40
14:25:26.388 -> 46
14:25:26.388 -> 52
```

```
14:25:29.628 -> 0
14:25:29.628 -> 0
14:25:29.628 -> 0
14:25:29.628 -> 1
14:25:29.628 -> 2
14:25:29.628 -> 3
14:25:29.628 -> 6
14:25:29.628 -> 8
14:25:29.628 -> 12
14:25:29.628 -> 15
14:25:29.628 -> 19
14:25:29.628 -> 24
14:25:29.668 -> 29
14:25:29.668 -> 34
14:25:29.668 -> 40
14:25:29.668 -> 46
```

***Figure 4.5*** *Bluetooth modules sending chars*

# 5 Architecture and High Level Design

## 5.1 System Architecture and Design

The BAT is a device made up of multiple components of both input and output functions. The system is broken down into two components, the transmitter, and receiver. The transmitter is in charge of all the input devices. The user can use a PC, phone, or TV as an input device where the audio will be received from. Then the transmitter converts the analog signal to a binary signal to be able to send to the Bluetooth. The Bluetooth module sends the binary values to the other Bluetooth module, which is located on the receiver. Simultaneously, the receiver obtains the audio signal from the transmitter and increases the sound quality of the audio signal. The transmitter produces the audio signal that has been received through a wired speaker. Both the transmitter and the receiver have a Bluetooth module which is in charge of sending and receiving the signal from transmitter to the receiver. The high level design of the system can be seen in Figure 5.1. The composition of the system allows the transmitter to focus on transmitting, the Bluetooth and is incharge of connectivity, and the charge of receiving. The main idea of the layout of the system is to create an effective simple layout to show that low price components can produce high quality sound. Moreover, the simplicity of the structural design allows further improvement and debugging in the system when problems or opportunities present themselves. The system architecture allows the components to be in charge of one specific task. This means that when the system runs, it does not produce much latency and lag, because each component can focus on their task. This dynamic within the structure reduces the finite amount of usable memory, RAM, and processing speed of the Arduino.

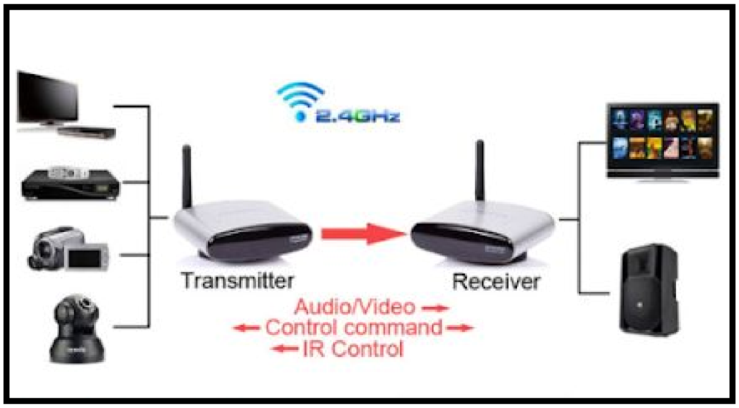


***Figure 5.1 -** High level block diagram interface*

## 5.2 Hardware Architecture

**Responsibilities -**

Zohaib Khan was in charge o f the coupling circuit, power amplifier and DAC.

Karen Escareno was in charge of the Bluetooth.

The 3.5 mm. jack accepts a analog signal from the user through any desired PC. The Arduino accepts the analog signal and output a digital signal to DAC. The DAC sends an analog signal to the low pass filter, which reduces the unwanted background noise. Moreover, the signal goes into a signal amplitude which allows the user to increase the amplitude of the analog signal and the power amplifier increases the current allowing  the signal to have a louder output. Lastly, the signal goes into a speaker or headphone jack which in return outputs the corresponding noise that the user inputted in the beginning of the circuit.

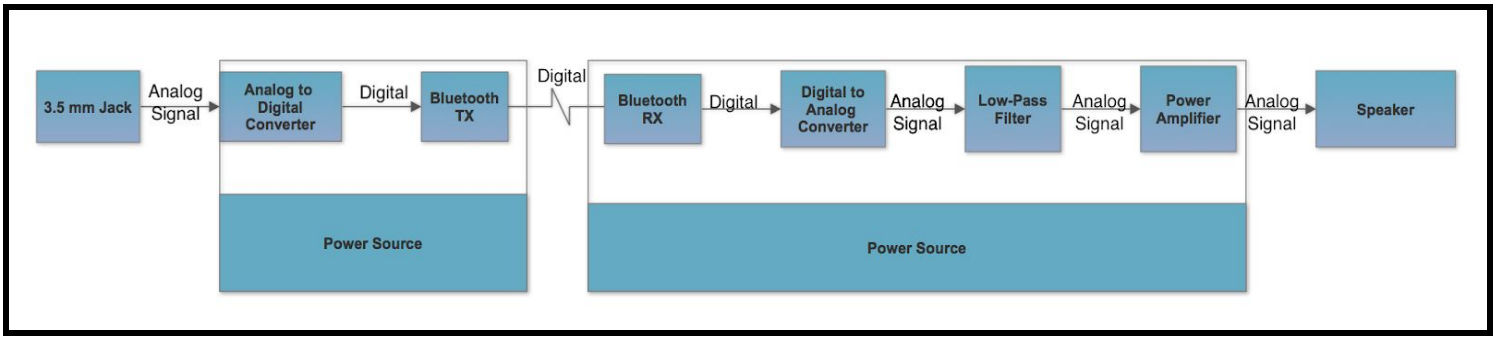


***Figure 5.2 -*** *High level block diagram of hardware*

## 5.3 Software Architecture

**Responsibilities -**

Zohaib Khan was in charge of ADC, analog input, and output.

Karen Escareno was in charge of Bluetooth commands.

The user inputs a signal in which the ADC converts this signal. 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. The Bluetooth module is taking the data that it receives from the ADC and it writes it to the other module. The master Bluetooth then sends it to the slave module. The slave module reads the data it has received and plays the audio signal to the speaker.

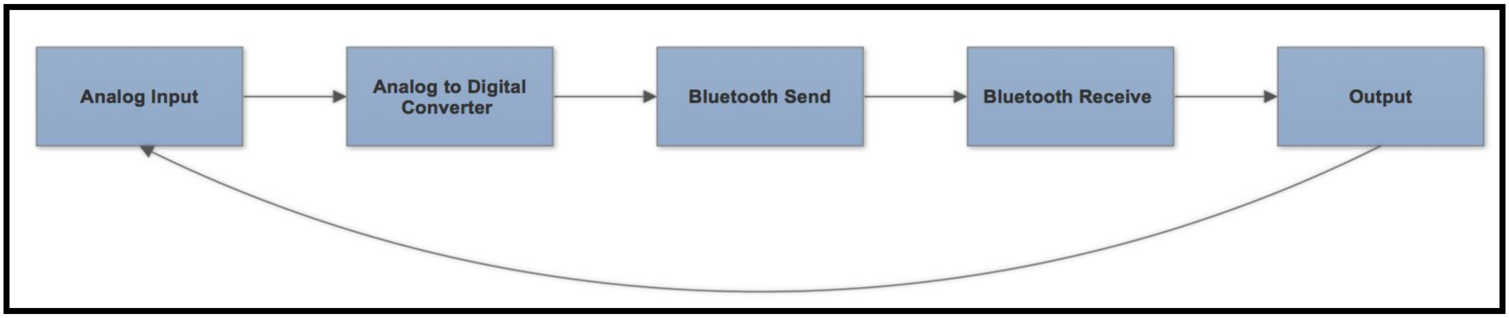


***Figure 5.3 -*** *High level block diagram of software*

## 5.4 Rationale and Alternatives

The project used this architecture because it is the most simplest way to produce quality results for the system. Moreover, each module is created from scratch, thus allowing us to understand each component much better. Software wise, the architecture used is needed to create the sampling speed required and it led to specific values of corresponding 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, resulting to its unique architecture.

The 3.5 mm. jack was chosen for BAT because it is more of a common consumer product compared to the RCA jack. It is most commonly used by consumers since they can play audio from their phones, computer, etc. The approach of using the built in ADC was because of timing issues, but it was convenient because it allows the project to be more efficient and marketable.

Given not a single component could perform DAC it started with a component from DigiKey that was marketed be able to perform the functions needed. The Arduino, which was the most accessible, could perform the functions the best for this project.

# 6. Data Structures

## 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:

### 7.1.1 Processing narrative for *module Input*

This module was constructed due to the Arduino being unable to handle negative voltages. When the 3.5 mm jack is connected to the analog 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 Input* 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 the system that allows the signal to be transmitted and outputted.

### 7.1.3 *Module Input* processing details

This module is created by an op-amp and an AC coupling circuit. The op-amp is the TS922IN. The gain of the op-amp can be calculated with R2/R1. R2 is equal to 1000 Ohms and R1 is equal to 20 Ohms, which resulted in a gain of 50 Ohms. Specifically, with the TS922IN, 20 Ohms is connected to pin 2 and the 1000 Ohm resistor is connected to pin 2 and pin 1. 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, which can be seen in Figure 7.1. To stop the output device from shorting out, a capacitor is used. The limitation of the op-amp is the max current of 80 mA and an operating voltage of 10 volts. Furthermore, the op-amp produces low interference around 9 nV/√ Hz, which is optimal for an audio system.

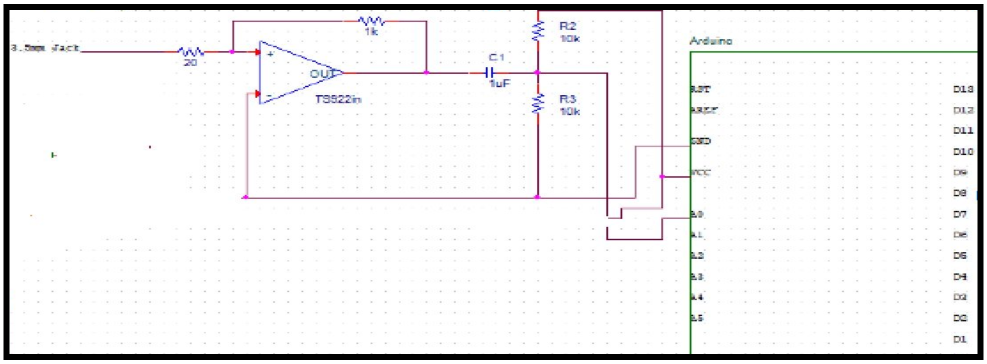


***Figure** 7.1 - Hardware schematic of the input module*

### 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 be change the digital signal to an analog signal.

### *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 and power amplifier, thus 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, PORTD of the Arduino is used which are pins 0-7 and connected 20K resistors on to each pin. Furthermore, 10K resistors were added in a series to create the R2R resistor ladder. The schematic can be seen Figure 7.2. The 8-bit R2R resistor ladder can produce values from 0 to 255 on a scale of 0 to 5 volts. Since this is an 8-bit DAC, it cannot produce a resolution as a 16 bit DAC can, which can lower the quality of sound. However, an 8- bit DAC produces quality audio which is suitable for the system.

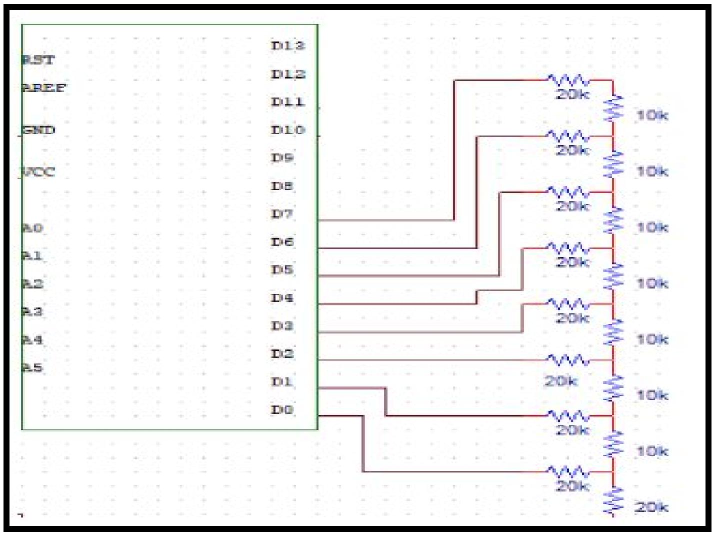


***Figure 7.2 -*** *Hardware schematic of the input module*

### 7.3.1 Processing narrative for *module Output*

This module was constructed due to the DAC, it produces a distorted signal when connected to the speaker directly. The DAC is able to produce the corresponding analog signal, however, the signal listened to was distorted compared to 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 to create a clearer sound.

### 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 goes 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 follower. 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. Then it can connect to a 10k potentiometer that allows the amplitude to be varied by turning the knob which can be seen in Figure 7.3. Since two TS922IN are being used, it can be used to increase the current by 160 mA due to one op-amp increasing the current by 80 mA.The output of the op amp is connected s into a speaker or headphone jack that allows the user to hear the audio.

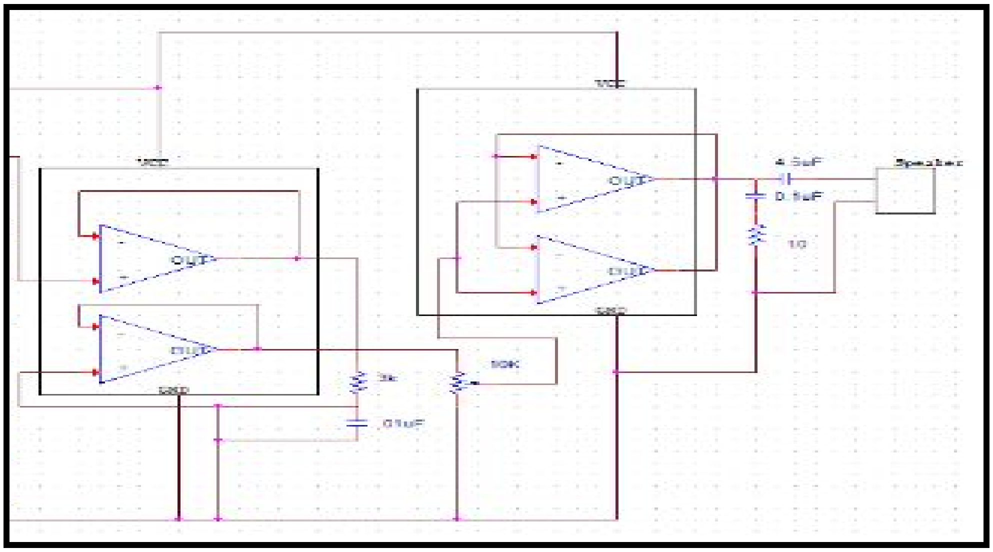


*Figure 7.3 - Hardware schematic of the output module*

### 7.4.1 Processing narrative for *module ADC*

This module was constructed due to the Arduino having a default pre-scaler of 128. This means that the default sampling rate is 10 kHz. The system requires around 80 kHz sampling rate due to most audio through a PC is around 35 kHz.

### *7.4.2 Module ADC Interface* description

This module interacts with the bridge between the input and output. It takes in an analog signal from the user, converts the analog signal to a digital signal, and sends the audio to the speaker.

### *7.4.3 Module ADC* processing details

This module is programmed to have a 76.9 kHz sampling rate and continuous trigger as the analog signal comes in.The ADCSRA register enables the ADC conversions and interrupts. The ADMUX register declares A0 as an input to allow the system to read data from the 3.5 mm jack. 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. When the ADC does a shift bit, both the input and output become 8 bit which results in better audio quality. With these configurations set up, the output to the DAC is reached by calling PORTD = ADCH. Limitations of this ADC is that it is 10-bit. The 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.

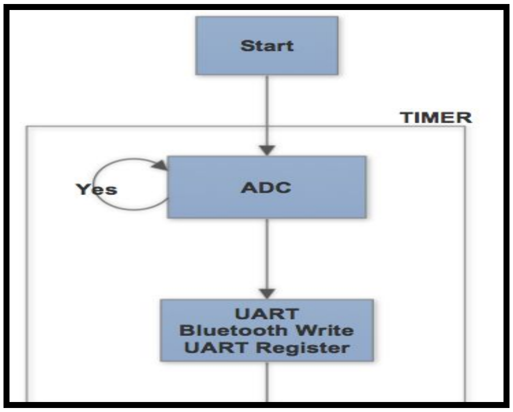


***Figure** 7.4 - Flow chart of ADC module*

### 7.5.1 Processing narrative for *module Bluetooth*

The software module for the Bluetooth was made so that it could get information from the Bluetooth module. This enables the Bluetooth module to write and read from the UART register that is shared between both modules.

### *7.5.2 Module Bluetooth Interface* description

The pins are set to 10 and 11 to be able to read from the Bluetooth module. The Bluetooth writes the char values to a UART register. The slave Bluetooth then reads the values from the UART register and then outputs to PORTD.

### *7.5.3 Module Bluetooth* processing details

The two Arduinos use AT commands in the system. An AT+Role was used to set the master and slave Bluetooth. The master had a role of 1 and the slave had a role of 0. The address is used when the slave is using AT+ADDR and able to connect the slave to the master using AT+BIND=(slave address). A timer was made for the ADC and for the write (Bluetoothname.write()) command for the first Arduino so it can first get a sampling rate of 8 kHz. The Arduino had to give a prescaler of 8 and a compare match register of 249 to get 8 kHz. The timer 0 was set because it was less than a compared match register of 256. Then Arduino is separated so the slave bluetooth is able to read the (Bluetoothname.read()) values from the master Bluetooth, which then output to PORTD as seen in Figure 7.5.

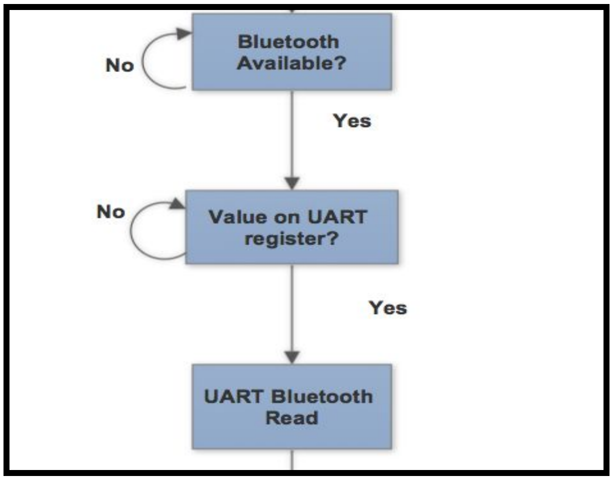


***Figure** 7.5 - Software flow chart of the bluetooth module*

### 7.6.1 Processing narrative for *module Bluetooth*

The hardware for the Bluetooth were the TX, RX, VCC and GND pins. Then the Bluetooths are able to connect to the Arduino so it can communicate through code.

### *7.6.2 Module Bluetooth Interface* description

The Arduino is connected to the TX and RX to help communicate with the Arduino. The Arduino establishes the Bluetooths connection with each other when the lights are in sync. The rest of the communication between them was done in software.

### *7.6.3 Module Bluetooth* processing details

The TX and RX of the Bluetooth module to the Serial connections in pins 0 and 1 were unable to connect because it had too many errors when uploading code and communication between the two Bluetooths. It causes the Bluetooth to receive incorrect information and not the code to be uploaded. The solution to this problem was to change the pins to 10 and 11. Then the Arduino is connected to the TX of the Bluetooth, which can be seen Figure 7.6.

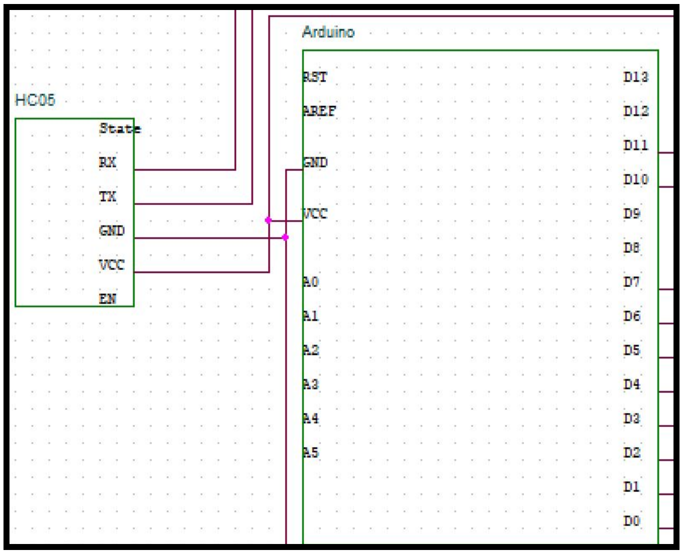


***Figure 7.6*** *- Master and slave bluetooth connections*

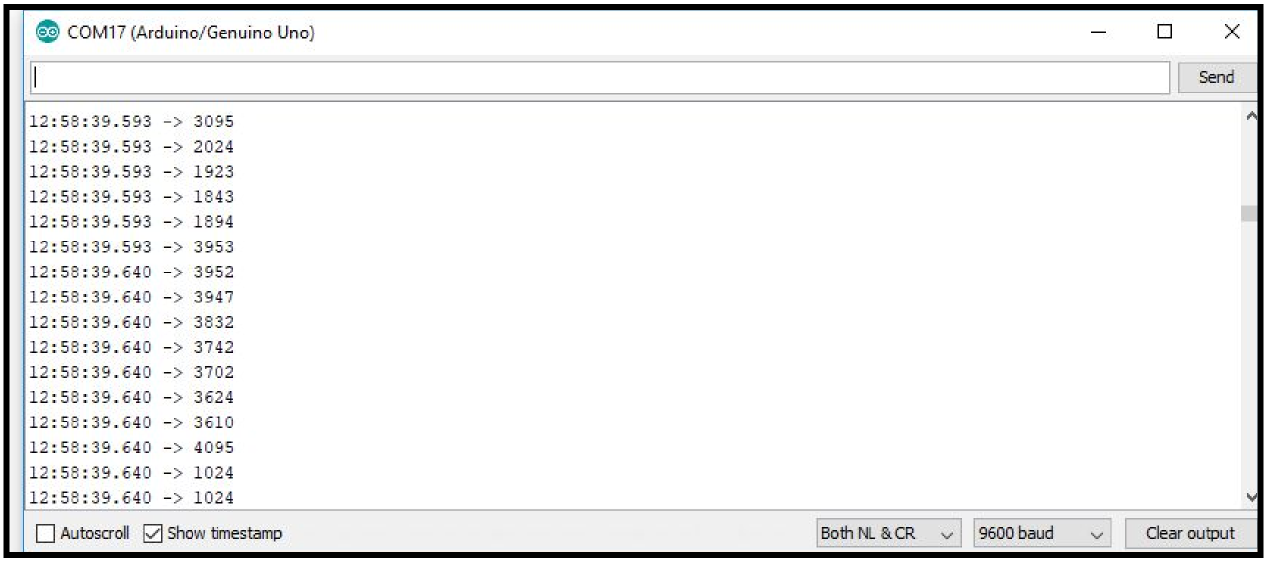
# 8 Technical Problem Solving

## 8.1 SPI Registering Programming

The BAT uses an 8 bit DAC. However, an 8 bit DAC does not have the resolution of a 12 bit DAC. Therefore, the sound quality of the BAT resulted to static and outside noise.

## 8.2 SPI Registering Programming

Due to the sound quality problem, the system tried to use an external DAC, the MCP4725 12 bit DAC. For the DAC to work, SPI programming was needed. As seen in Figure 8.2**,** a successful implementation of SPI was configured where the DAC is converting an analog signal to a digital signal. However, with the addition to the HC-05 bluetooth module, the SPI protocol was malfunctioning. This occurred because when the system was calling a timer interrupt, where it would not successfully send data. The timer function was being called quicker than the I2C function, therefore it caused interference with the data. Thus, the system stuck with the 8 bit DAC due to its flexibility and easy implementation. The new skills that were needed to solve this problem was UART register programming.



```
12:58:39.593 -> 3095
12:58:39.593 -> 2024
12:58:39.593 -> 1923
12:58:39.593 -> 1843
12:58:39.593 -> 1894
12:58:39.593 -> 3953
12:58:39.640 -> 3952
12:58:39.640 -> 3947
12:58:39.640 -> 3832
12:58:39.640 -> 3742
12:58:39.640 -> 3702
12:58:39.640 -> 3624
12:58:39.640 -> 3610
12:58:39.640 -> 4095
12:58:39.640 -> 1024
12:58:39.640 -> 1024
```

☐ Autoscroll ☑ Show timestamp  Both NL & CR  9600 baud

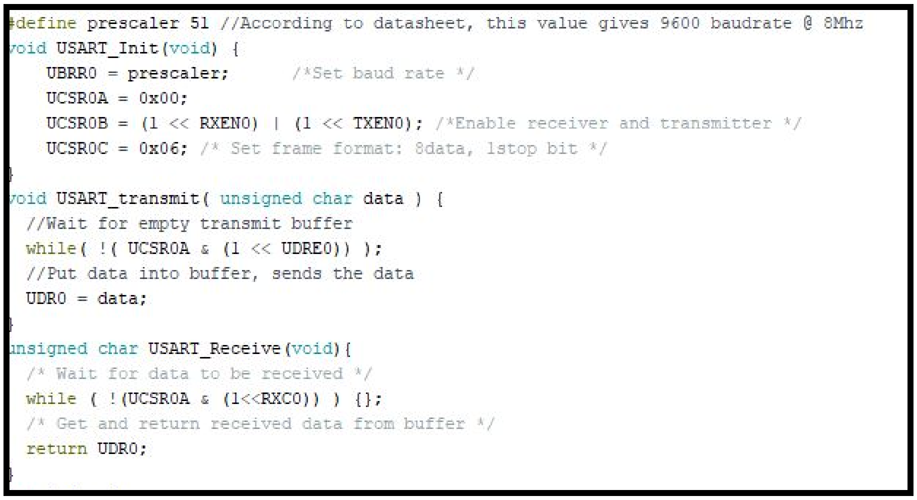
***Figure 8.2.1 -*** *Result of SPI implementation*

## 8.1 UART Register Programming

The BAT uses the HC-05 Bluetooth modules to send data from the transmitter to the receiver. Unfortunately, when running the program for the modules, incorrect data was being sent and received.

## 8.2 UART Register Programming

To tackle the problem of the misread data, the implementation of the UART protocol was required because it allows the system to configure the settings of the Bluetooth modules. As seen in Figure 8.2, a successful implementation of UART was designed. The circuit to test the implementation was one Arduino and an LED. The LED was an output and the RX and TX pins of the Arduino were connected. With the code below, the script sent a char value of 0x08 and recieved it. Once received, the system compared the received values with the sen values. Once both values matched, the program was able to send accurate values using the TX and RX pins of the Arduino. The new skills that were needed to solve this problem was 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;
}
```

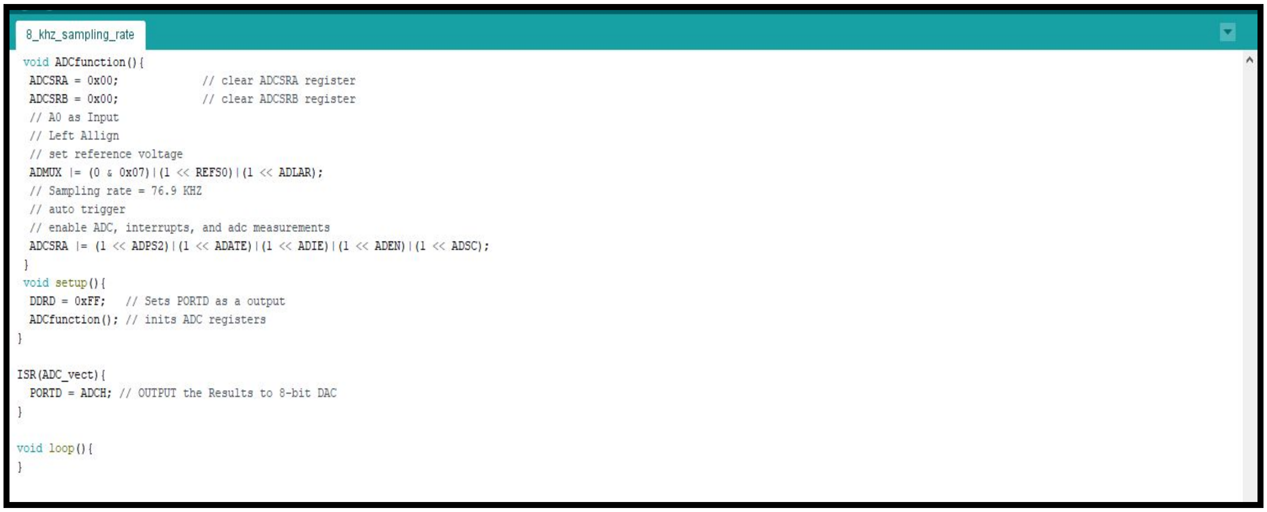
***Figure 8.2.2 -*** *UART implementation*

## 8.1 Bluetooth ADC Timer

Another problem that the system encountered was that the ADC was having a conversion problem. Specifically, when the ADC converted the analog signal to a binary signal, it was not fast enough to keep up with the sending speed of the Bluetooth.

## 8.2 Bluetooth ADC Timer

To combat this problem, a timer function is needed to make sure each function is properly called when the program runs. To be able to increase the sampling speed, the program needed a smaller prescaler to increase the sampling speed. As seen in Figure 8.2, the new implementation led the ADC to run at 8 kHz sampling speed, which allows the ADC to keep up with the speed of the Bluetooth modules. The ADC and time functions were successfully implemented. The new skills needed to solve these problems are timer and ADC register programming.



```
 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
}

ISR(ADC_vect){
  PORTD = ADCH; // OUTPUT the Results to 8-bit DAC
}

void loop(){
}
```

***Figure 8.2.3 -*** *ADC timer implementation*

# 9 User Interaction Features

## 9.1 Application Screens

N/A

## 9.2 User Interface Screens

N/A

# 10 Test Plan

## 10.1 Test Design

Test Case 1 - ADC Sampling Rate

Objective: The objective of this test is to have the ADC sampling at a rate of 80 kHz to be able to produce sound from a phone. This tested the ADC function and measured the technical objective which requires the system to have an output of 40 kHz.

Setup: Look into the atmega328p datasheet to find the equation to calculate sampling rate. The equation to calculate sampling rate is 16 MHz/prescaler/ 13.

Procedure and how data is collected: Run the ADC script and make sure the serial monitor has the correct sampling speed required for the BAT. Moreover, with the equation stated above, the desired sampling rate can be achieved by choosing the correct prescaler.

Expected results: The expected results of the ADC sampling rate is to have the value of 76.9 kHz.

Test Case 2 - Accuracy of the System

Objective: The objective of this test is to measure the accuracy of the input and output audio signal. The experiment measured the technical objective which requires the system to have an accuracy error of 5 percent.

Setup: The setup is consisted of creating the coupling circuit and making sure all the components are connected. Once completed, connect the oscilloscope to the positive and negative ends of the output.

Procedure and how data is collected: Connect the 3.5 mm jack to a PC and play a signal with a specific frequency. The data is collected by observing the data on the oscilloscope.

Expected results: The expected results of the test was 5 percent error to the corresponding to the input signal. This means that if the input signal is 100 Hz, then the output signal must be around 95 Hz.

Test Case 3 - Input Frequency

Objective: The objective of the experiment is to to make sure the Arduino can read the analog values as high as 40 kHz. The experiment measured the technical objective which requires the system to have an input of 40 kHz.

Setup: Have the 3.5 mm jack properly connected and have an oscilloscope connected to the coupling circuit.

Procedure and how data is collected: Play an audio signal with a frequency of 40 kHz and record the data from the oscilloscope to make sure the frequency is as close to the desired results.

Expected results: The expected results of the test is to make sure the input frequency the Audirno received is the same as the frequency of the audio signal played from the phone.

Test Case 4 - Response Time

Objective: The objective of the test is to record the response time of the BAT. The experiment measured the technical objective which requires the system to have an response time of half a second.

Setup: Connected a PC to the BAT and made sure to have a stopwatch on hand.

Procedure and how data is collected: Once the sound is heard on the speaker, stop the stopwatch. The data is collected through the stopwatch.

Expected results: The expected results of the experiment is to make sure the response time is no longer than one second.

Test Case 5 - Bluetooth Sampling Rate

Objective: The objective of the test is to increasing the sampling rate of the Bluetooth modules. The experiment measured the technical objective which requires the system to have Bluetooth capability.

Setup: The setup used to determine the sampling rate was by using a timer interrupt code.

Procedure and how data is collected: The data is collected by the serial monitor which will output the corresponding sampling rate speed of the BAT.

Expected results: The expected result of the experiment is to run the Bluetooth modules at a speed of 8 kHz.

Test Case 6 - Bluetooth Range

Objective: The objective of the test is to make sure the range of the Bluetooth modules are within 20 feet as stated in the technical objectives.

Setup:Have the master and slave Bluetooth modules connected to each other.

Procedure and how data is collected: Play an audio signal from the input, with both Arduinos 1m apart. Started to increase the distance by 1m until the audio signal stopped playing.

Expected results: The expected results of the experiment is that the range is between 15 to 20 feet.

## 10.2 Bug Tracking

A database, shown in Figure 10.2, was used to monitor specific bugs discovered while performing the system's experiments. Once a bug is found, it will be the responsibility of the individuals to investigate the specific defect.

| Test Cases | Defect |
|---|---|
| Case 1 | ADC did not receive the 8 bit signal |
| Case 2 | N/A |
| Case 3 | High frequencies burned the Arduino |
| Case 4 | N/A |
| Case 5 | Bluetooth could not connect to each other |
| Case 6 | Interference by the walls |

***Figure 10.2*** - *Defects of test cases*

## 10.3 Quality Control

The experiments below can either pass or fail when designing a system. The figure below logged all deviation from the expected results which can be seen in Figure 10.3.

| Test Cases | Pass/Fail | Discrepancy |
|---|---|---|
| Test 1 | Fail ( Wed) | Sampling rate too low |
| Test 1 | Pass (Mon) | N/A |
| Test 2 | Fail (Thur) | Higher the frequencies lowers accuracy rates |
| Test 2 | Pass(Thur) | Higher the frequencies lowers accuracy rates |
| Test 3 | Pass(Tues) | N/A |
| Test 4 | Pass (Sat) | N/A |
| Test 5 | Pass (Sun) | Higher baud rates lowers connection speeds |

***Figure 10. 3*** *- Results of the experimental data*

## 10.4 Identification of critical components

1. 3.5mm jack

   - This is a critical component in that it is the input of the entire circuit. The system needed to pay attention to this during testing because based on the audio the user chooses, the frequency can vary therefore the input circuit needed to take account of the different frequencies.

2. Output

   - The output of the system needed extra testing because that was the final destination of the audio signal. The test made sure the output was consistent and could not be affected by the different components in the system.

3. Bluetooth

   - The Bluetooth modules are were a critical component in the testing because if it was not sending or receiving the right values then the entire output of the system would be incorrect.

## 10.5 Items Not Tested by the Experiments

1. Op Amps
   - The op amps used in the output were not tested because the data sheet said it can increase current by 80 mA.
2. Resistors
   - The resistors used in the circuit were not tested for their resistance because the resistance were stated in the data sheet.
3. Capacitors
   - The capacitors were not tested for a specific capacitance due to the value given on the data sheet.

# 11 Test Report

## 11.1 Test 1 - ADC Sampling Rate

Test Results:

For the first test, the system used a prescaler of 32, which resulted in a 38.4 kHz sampling rate.

Comparison with expected results:

The sampling rate acquired had a sampling rate of 38.4 kHz which is less than have the sampling rate of 78 kHz needed to be able to produce good quality sound.

Analysis of Test Results:

The reason the test results yielded 38.4 kHz is due to the prescaler. Using a prescaler of 32 yielded a sampling rate of 38.4 kHz

Corrective Actions Taken:

To correct the mistake the system used a prescaler value of 16 to yield a sampling rate of 78.8 kHz.

**Test 1.2**

Test Results:

For the second test, the prescaler was set to 16 which resulted in the sampling rate becoming 78.6 kHz.

Comparison with expected results:

The test result obtained can be calculated by dividing 16 MHz by the prescaler. The calculations yielded a sampling rate of 78.6 kHz which was exactly the expected results.

Analysis of Test Results:

The sampling rate achieved is the closest the system can get to using the preset prescaler values. For example, using a small prescaler yielded values less than 30 kHz and a prescaler value higher than 16 will double the sampling rate but also increase the aliasing. Therefore, a sampling rate of 78.6 Khz is the most optimal configuration for the BAT.

Corrective Actions Taken:

No corrective actions taken. Objective was successfully obtained.

## 11.2 Test 2 - Accuracy

Test Results:

For the first test, the input signal from the PC to the BAT was 200 Hz. The output signal was at 284 Hz. This resulted in a 58 percent accuracy.

Comparison with expected results:

The theoretical expectation was around 200 Hz, however the calculated results were around 284 Hz.

Analysis of Test Results:

The calculated results were much higher than the theoretical prediction due to the unwanted noise in the background.

Corrective Actions Taken:

Due to a lot of unwanted background noise, an implementation of a low pass filter would greatly reduce the extra noise of the circuit.

**Test 2.2**

Test Results:

With the new implementation of a low pass filter, the input signal was 3 kHz and the output result was oscillating from 2.940 to 3.020 kHz, which can be seen in Figure 11.2.2.

Comparison with expected results:

The calculated results was round 2.9 Khz, while the actual result was at 3 kHz.

Analysis of Test Results:

The percent error was calculated to at 2 percent. The goal of our testing is to achieve a percent error as close to 0 as possible. Unfortunately, 0 percent error is practically impossible therefore the system chose to be run at 2 percent error.

Corrective Actions Taken:

No action were taken due to completing the task.

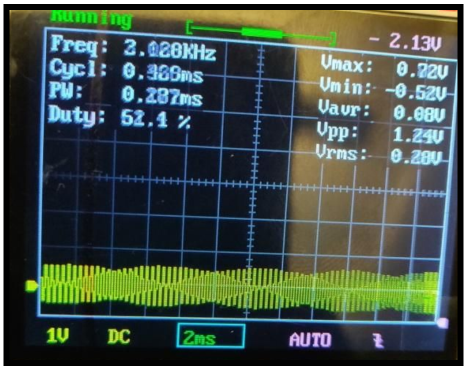


***Figure 11.2.2** - Sine wave signal*

## 11.3 Test 3 - Input Frequency

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 match the theoretical predictions because the upper boundary needed to be 30 kHz and the lower bound needed to be around 500 Hz. The experiment results can be seen in Figure 11.3.2.

Analysis of Test Results:

The input frequencies 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 30 kHz.

Corrective Actions Taken:

No actions were taken because the desired results were obtained.

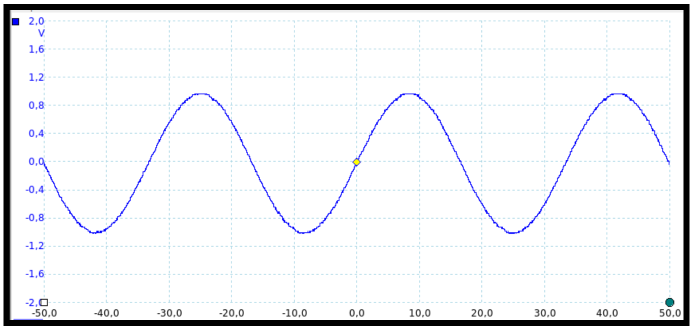


***Figure 11.3.2 -*** *Input frequency of 30 kHz*

## 11.4 Test 4 - Response Time

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 as seen in Figure 11.4.1.

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 this experiment using a stopwatch.

Analysis of Test Results:

The time it took to get for the input to the output was in microseconds which was the expected results.

Corrective Actions Taken:

No further action was taken since the desired response time was met.

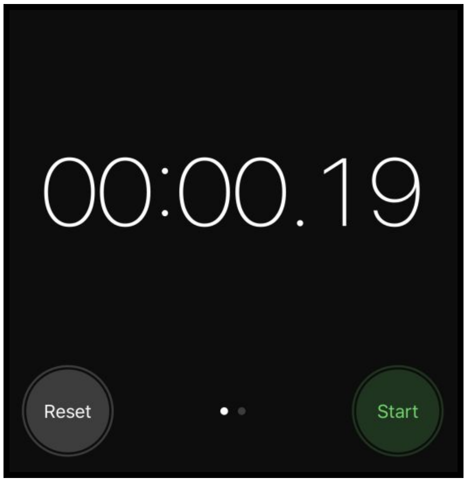


***Figure 11.4.1*** *- Response time*

## 11.5 Test 5 - Bluetooth Sampling

Test Results:

The test results yielded  a 2 kHz interrupt to speed up the ADC

Comparison with expected results:

The calculations results were 5 kHz less was that the frequency expected.

Analysis of Test Results:

The interrupt was at such a small value and was not going as quick as expected. The system needed a larger sampling rate because the sampling rate being used was not taking in as much audio.

Corrective Actions Taken:

To combat lower sampling rates, an increased baud rate was needed to allow the Bluetooth modules to successfully sample data.

**Test 5.2 -**

Test Results:

With an increased baud rate, the test results yielded a sampling rate of 8 kHz.

Comparison with expected results:

The system expected calculations and experimental calculations matched.

Analysis of Test Results:

With a much higher sampling rate, the system is able to pick up signals with frequencies as high as 4 kHz.

Corrective Actions Taken:

No further action was taken since the desired response time was met.

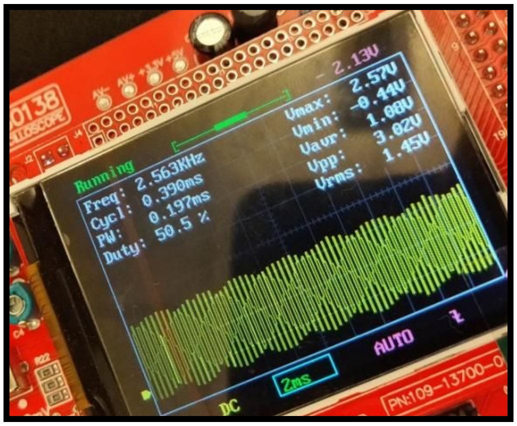


***Figure 11.6.1 -*** *Sine wave signal*

# 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 is 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 is to simulate and create a foundation to develop life-long skills that can be used in the future. 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.

The system meets the overall project goal because the purpose of the project was to create and understand all concepts of an audio playback device. Specifically, the technical design objectives included range: 20 Ft +, sampling rate: 40 kHz - 80 kHz, 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's Responsibilities-**

I was able to successfully complete all of the design objectives, which included sampling rate, response time, input frequency, and accuracy. The responsibilities included creating a working system without the Bluetooth module, so the partners can add the Bluetooth module to the working module. Furthermore, through hardware and software the input, DAC, and output modules stated above This senior design project had many hurdles and obstacles that continued throughout the project, however without them the team would not have learned so much. This project allowed us to learn how to effectively problem solve, regardless how difficult they are in the beginning. These lessons were important as they are skills to learn as engineers. In the project it is encouraged to continue the process and using all the resources that resolve the problems that emerge. Throughout the years as students instructions, tools, and resources are given to complete a project or lab. However in this project this was all up to the team to figure out. The value of the project is to help  create a careful design plan, record progress, and find 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 to learn 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.

**Rishi's Responsibilities- -**

The original goal was to cover everything pertaining to the DAC. I reconstructed the stream from the Bluetooth an analog signal. Then completed the Arduino software to register the input of the Arduino when plugging into a computer. Then using the new signal was created into the system and had information on how it interacts with the rest of the project. Next was marking the parts that are wired to the different power sources and other components the chip wanted based on the datasheet.

In doing these tasks it helped gain information on soldering 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's Responsibilities- -**

Overall the original goal was completed to take in an analog signal which will go over Bluetooth and then go back to analog and sent to the output. The code used with the Bluetooth and timer was insufficient as the audio was poor quality. The limitations of 115,200 baud rate which was only good for 8 kHz of audio caused the poor quality audio. The using this knowledge a proper song was used 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 was learned is that if the baud rate increased, it could have led to many errors because a high baud rate overruns the Bluetooth module. The project helped the team learn about the UART and how it works. The register programming for the arduino was learned so that the speed of the ADC and the Bluetooth function together. 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.

## 12.2 Future Work

The impact of this work is to show how to create a working audio system with efficient yet cost efficient products. For example, during the research stage, many individuals believed that the team would 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 the team was able to create a system with the doubted components used in the device. 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 3-D printed exterior should be produced, 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. These addition to the main component in the system will allow the product to be more optimized. This means that the user 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, the product is a great foundation of an audio signal playback device that the team and anyone who reads this report can replicate.

## 12.3 Acknowledgement

The individuals who provided us helpful feedback and critique with the 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] Wireless Audio Transmitter via Bluetooth.[Online]. Available: https://forum.arduino.cc/index.php?topic=89195.0.[Accessed:07-Jun-2019].

[2] Instructables, “Arduino Audio Output,” *Instructables*, 26-Oct-2017. [Online]. Available:https://www.instructables.com/id/Arduino-Audio-Output/.[Accessed: 07-Jun-2019].

[3] “Wire,” *Arduino*. [Online]. Available: https://www.arduino.cc/en/reference/wire. [Accessed: 07-Jun-2019].

# 14 Appendices

**Appendix A:** Parts List

| Part | QTY. |
|---|---|
| Arduino | 3 |
| Wire speaker | 1 |
| 100 Ohm Resistors | 10 |
| HC-05 Bluetooth module | 3 |
| Capacitors | 10 |
| TSS9IN op amps | 3 |
| Beardboard | 2 |
| PNPS | 4 |
| Mount headphone jack | 2 |

***Table 1A*** *- Parts list*

**Appendix B:** Equipment List

| Equipment | Usage |
|---|---|
| Oscilloscope | Measure analog signals |
| Multimeter | Measure voltage and resistance levels |

***Table 1B*** *- Equipment list*

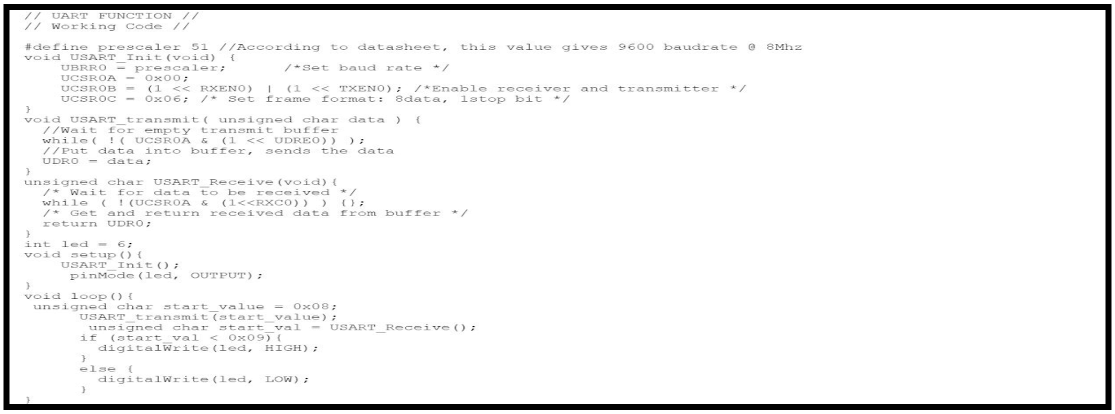
## **Appendix C:** Software List

| Program | Usage | Download |
|---|---|---|
| Arduino Uno | Programming the MCU | https://www.arduino.cc/en/Main/Software |
| MATLAB | Simulating the circuit design | https://www.mathworks.com/downloads |

***Table 1C*** *- Software list*

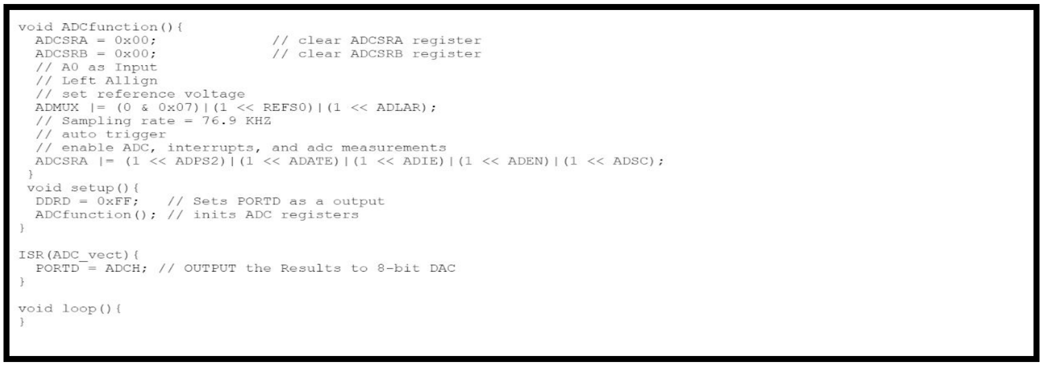
## **Appendix D:** Code

- The code of this project can be seen in the Google Drive link: https://drive.google.com/drive/folders/1bWz7cZo7g91ITM_Y1xnWkes_KrQ4up9p?usp=sharing



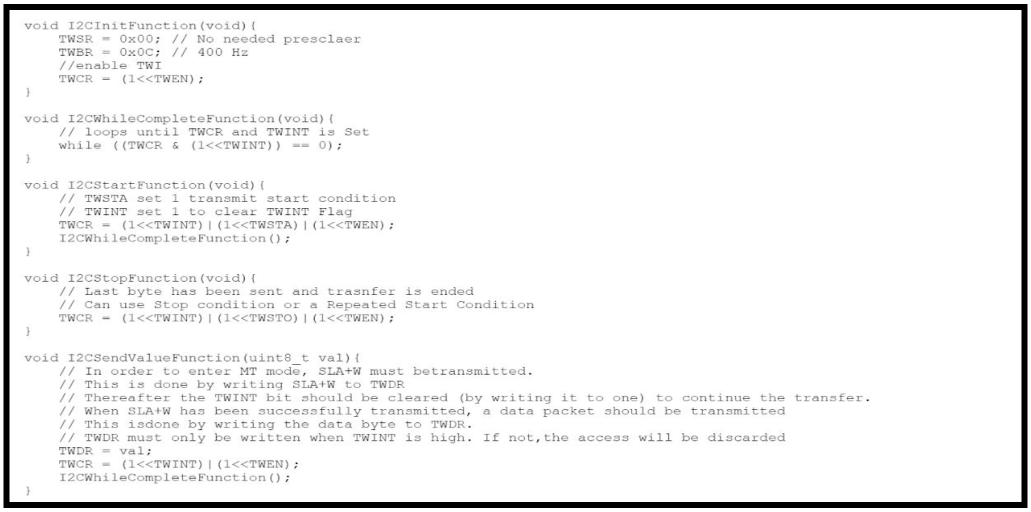
```
// UART FUNCTION //
// Working Code //

#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);
      }
}
```

**Code D1** - *UART implementation*



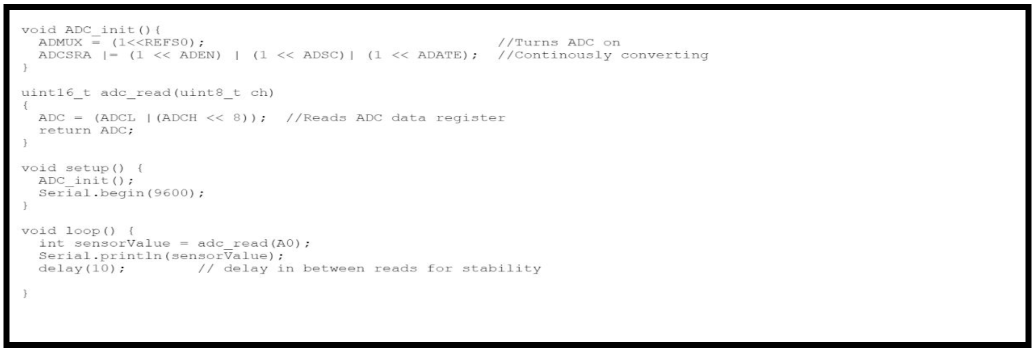
```
void ADCfunction(){
  ADCSRA = 0x00;                // clear ADCSRA register
  ADCSRB = 0x00;                // clear ADCSRB register
  // A0 as Input
  // Left Allign
  // set reference voltage
  ADMUX |= (0 & 0x07)|(1 << REFS0)|(1 << ADLAR);
  // Sampling rate = 76.9 KHZ
  // auto trigger
  // 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
}

ISR(ADC_vect){
  PORTD = ADCH; // OUTPUT the Results to 8-bit DAC
}

void loop(){
}
```

**Code D2** - *ADC implementation*



```
void I2CInitFunction(void){
    TWSR = 0x00; // No needed presclaer
    TWBR = 0x0C; // 400 Hz
    //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);
}

void I2CSendValueFunction(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
    TWDR = val;
    TWCR = (1<<TWINT)|(1<<TWEN);
    I2CWhileCompleteFunction();
}
```

**Code D3** - *I2C implementation*



```
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();
  Serial.begin(9600);
}

void loop() {
  int sensorValue = adc_read(A0);
  Serial.println(sensorValue);
  delay(10);        // delay in between reads for stability

}
```

**Code D4** - *ADC with Bluetooth implementation*

**Appendix E:** Vendors for Components

| Vendors | Components |
|---|---|
| Amazon | All |
| Digikey | All |
| Mouser | All |

***Table 1E*** *- Vendors for Components*