

# UNIVERSITY OF CAPE TOWN

## Department of Electrical Engineering



### **EEE3097S – Electrical and Computer Engineering Design Project 2018 Report for Milestone 3**

#### **Group 4**

<b>Tapuwa Dhliwayo</b>	<b>DHLTAP001</b>
<b>David Fransch</b>	<b>FRNDV011</b>
<b>Heng Rui Guo</b>	<b>HNGGUO001</b>
<b>Terri Lee</b>	<b>LXXDOH001</b>

**<14 October 2018>**

## Declaration

1. I know that plagiarism is wrong. Plagiarism is to use another's work and pretend that it is one's own.
2. I have used the IEEE convention for citation and referencing. Each contribution to, and quotation in, this final year project report from the work(s) of other people, has been attributed and has been cited and referenced.
3. This project report is the work of the team members of my group.
4. I have not allowed, and will not allow, anyone to copy my work with the intention of passing it off as their own work or part thereof.

**Name 1:** Terri Lee

**Signature:**  \_\_\_\_\_


**Date:** 14/10/2018

**Name 2:** Heng Rui Guo

**Signature:**  \_\_\_\_\_


**Date:** 14/10/2018

**Name 3:** David Fransch

**Signature:**  \_\_\_\_\_

**Date:** 14/10/2018

**Name 4:** Tapuwa Dhilwayo

**Signature:**  \_\_\_\_\_

**Date:** 14/10/2018

## Contents

**Declaration**

19

<b>1. Introduction and Project Plan</b>	<b>5</b>
<b>2. System Requirements</b>	<b>6</b>
2.1. User requirements	7
2.1. Technical specifications of the system	9
<b>3. Subsystem Description</b>	<b>12</b>
3.1. Use Case diagram	12
3.2. Sequence diagram	13
3.3. Activity diagram	13
<b>4. System Descriptions</b>	<b>15</b>
4.1. System Block Diagram	15
4.2. System Descriptions	16
4.3. Interfaces	16
<b>5. Identification of different approaches to problem solving and final selection</b>	<b>18</b>
5.1. Hardware design and list of components	20
5.2. Software tools for App Development	26
<b>6. Identification of sub-systems and sub-system requirements</b>	<b>29</b>
6.1. Mobile App Subsystem	29
6.1.1. Technical requirements for App communicating with RPi	29
6.1.2. UML diagrams	30
6.1.3. User interface	32
6.2.1 Technical requirements for Transmitter Circuit	38
6.2.2. Calculations, circuit diagrams and simulations	38
6.2.3. Build and Testing	39
<b>8. Logical integration of sub-systems and testing</b>	<b>52</b>
8.1. App and Tx RPi	52
8.2. Tx RPi and Rx RPi	53
8.3. Audio amplifier and Rx RPi	53
<b>9. System Level Testing</b>	<b>54</b>
9. 1. Testing	54
9.2. Bill of Materials	55
<b>10. Summary of contribution by each team member</b>	<b>55</b>

**11. Soft skills related to milestone 1**

**Error! Bookmark not defined.**

**References**

56

**Appendix A: Minutes of team meetings**

58

## 1. Introduction and Project Plan

The objective of the project is to design a Wireless Walkie-Talkie Melody Player. This system consists of two cellphones and two Raspberry Pis (RPi) that provide the following functionality:

- Each cellphone is associated with a RPi.
- A mobile app installed on the cellphones must allow the client to select a tune which will send the information relating to the tune to the associated RPi via Bluetooth.
- The associated RPi (i.e. transmitting RPi) will then encode and transmit the tune to the other RPi (receiving RPi) using infrared light.
- The receiving RPi must receive, decode the signal and play the tune via a speaker in real-time, with minimum delay.

Our goals for this project are to:

- produce a satisfactory product by the deadline (15th October 2018).
- submit all required deliverables/milestones by their respective due dates.
- meet all user requirements and technical specifications.

The project is limited by the following constraints:

- funding limited to R800
- total project duration of 10 weeks
- members of the project fixed to 4 people
- source of electrical components is limited to RS components and White Lab
- maximum transmission rate limited by the toggle rate of LEDs available at the supplier
- maximum transmission distance limited by the maximum power of the LED

The Wireless Walkie-Talkie Melody Player needs to be designed to maximise/optimize the following two properties:

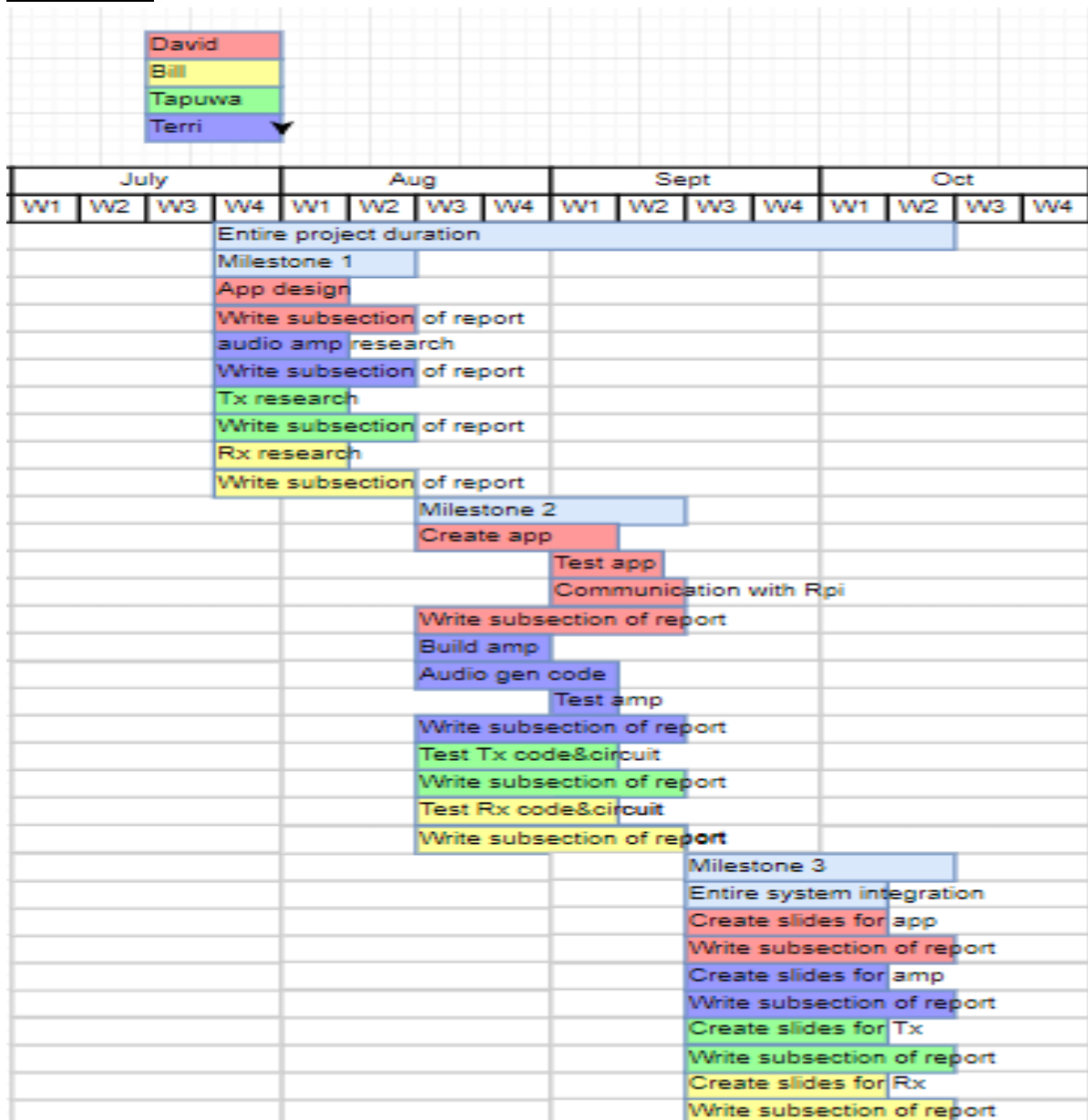
- Speed of data transfer or data rate between two RPis
- The distance between the two RPis.

Required for milestone 3 is the demonstration, the report and a supporting video. The demonstration requires the testing of the whole system (system level testing). The results of these tests should confirm whether the system requirements and technical specifications were met, partially met or unsuccessfully met.

Reflections and conclusions will then be drawn from these results. The report documents the entire design process and requires the following: user and refined requirements, UML diagrams, interfaces, subsystem designs, logical integration of sub-systems and testing, and finally system level testing. The supporting video recap the objectives of the project and demonstrate if the system requirements were met or not.

The report documents the entire design process of the project. It begins by listing user requirements and then refining these into technical specifications. From there, UML diagrams further describes the project system graphically. Afterwards, the hardware design and software design of the sub-systems are discussed in detail. This is followed by integration of these sub-systems and system level testing.

## Gantt chart



## 2. System Requirements

### 2.1. User requirements

<b><u>Key:</u></b>	
UR#	User Requirement Number
T#	Test Number

#### **Mobile App:**

- UR1: Compatibility with android devices.
  - T1: Run application on android emulator and android mobile device.
  -
- UR2: Setup bluetooth and connect to device.
  - T2: Connect to Transmitting (Tx) RPi using Bluetooth from App.
- UR3: Store the note data parameters.
  - T3: Verify within App.
- UR4: Display multiple notes to user.
  - T4: View within App.
- UR5: Select notes using tap gesture.
  - T5: TextView is used to verify selection of note or melody.
- UR6: Play a single note at a specified volume and duration.
  - T6: Verify parameters on Tx RPi Serial Port console.
- UR7: Play melodies consisting of notes at a specified volume and duration.
  - T7: Verify parameters on Tx RPi Serial Port console.

### **Transmitter Circuit:**

- UR8: Play melodies controlled by mobile app
  - T8: Test to see if the melody is on the Raspberry Pi after it is sent by Bluetooth by connecting it to a monitor to check
- UR9: Change volume, pitch and add effects e.g. reverb controlled by mobile app
  - T9: Test by connecting Raspberry Pi to a monitor and checking to see if code manipulates melodies in the required manner
- UR10: Melodies must be encoded before transmission
  - T10: Test by checking that signal transmitted when connected to a speaker won't play melody
- UR11: Transmit encoded melodies using IR Leds
  - T11: Check to see if a binary signal can be seen at the receiving end using an oscilloscope

### **Receiver Circuit:**

- UR12: Receive the encoded tune via visible light or infrared light.
  - T12: Check encoded data is received via IR or visible light by detecting IR using a smartphone camera which displays IR light as a purple glow or in the case of visible light, just visually check. Check received data using oscilloscopes.
- UR13: Decode the encoded signal.
  - T13: Check the original signal is the same as the decoded signal.
- UR14: Play the tune via a speaker in real-time, with minimum delay as possible.
  - T14: Test by measuring how long it takes for tune to play after it is selected.

### **Speaker & Amplifier:**

- UR15: Amplifier should be able to be interfaced with the Receiving (Rx) RPi with an aux cable
  - T15: Connect the amplifier to the RPi with the aux cable and play an mp3 file and listen to the sound from the speaker (test if aux cable works)
- UR16: Speaker is clearly audible of upto 2m away
  - T16: generates a note at maximum volume. Stand 2m away from the speaker and decide if it is loud and clear.
- UR17: Speaker has no distinct noise that interferes with the sound it is playing
  - T17: Probe the speaker with the oscilloscope and use the function generator to generate a signal at a specific frequency (e.g 261Hz) and connect it to the input of the amp. The output waveform should be a smooth sine wave, if not, there is noise present. (reason for using the function generator and not the RPi is that the RPi superimposes the input signal with a carrier signal for improved transmission)



## 2.1. Technical specifications of the system

The user requirements are translated into technical specification. The technical specifications of the system describe the functions of the system and how well the system performs these functions, using technical terminology.

<b><u>Key:</u></b>	
TS#	Technical Requirement Number
T#	Test Number

### **Mobile App:**

- TS1: The App should be compatible with android devices.
  - T1: Android compatibility will be tested on the android emulator Nexus 5 API 23 as well as two different android mobile devices namely; the Samsung J5 prime and the Sony Xperia M5.
- TS2: The App should be able to connect to the Tx RPi via Bluetooth v4.1 Low Energy (LE) standard.
  - T2: Verify connection by connecting to Tx RPi and sending a simple message.
  - T2: Expect throughput around 100kbs. Maximum throughput is 230kbs.
  - T2: Range of about 10m. Walking distance test to verify.
- TS3: Note parameters which consist of the note, volume and duration associated with each selected note are to be read in as Strings and sent via a byte array to the Tx RPi which will read in the parameters as Strings.
  - T3: The serial port console on the Tx RPi will be used to verify correctness.
- TS4: Notes are to be displayed in a user-friendly manner which minimizes the user effort involved in selection.
  - T4: The user interface will be tested by all four team members as well as two non-members to get feedback for improvement and verify that the TS3 is met.
- TS5: Note selection will be done using simple tap gesture. A conventional Onclick method approach is used to wait for user gesture and on user gesture store and send note parameters to the Tx RPi.
  - T5: TS4 will be tested using the android studio environment (specifically the Logcat window).
- .TS6: Send and play single note on the output speaker in a full-duplex manner.

- T6: Verify parameters on Tx RPi Serial Port console and check frequency of note on output of speaker using an oscilloscope.
- T6: Play note from each device concurrently.
- TS7: Send and play a melody consisting of notes in a full-duplex manner.
  - T7: Same as T6.

### **Transmitter Circuit:**

#### Raspberry PI

- TS8: enabled to receive instructions from a mobile app
- TS9: encode melodies through digital modulation
- TS10: include error detection within encoded data
- TS11: include error correction should errors be detected
- TS12: encoded data is relayed through GPIO pins connected to the physical

#### Transmitting Circuit

- TS13: send encoded data from Raspberry pi using an IR Led configuration
- TS14: must transmit between a reasonable distance of 2m - 5m approximately
- TS15: should be able to transmit over a short distance 50cm through a glass screen
- TS16: should be a low voltage system using 3.3V to 5V to power the circuit

### **Receiver Circuit:**

#### Raspberry PI

- TS17: Able to detect and receive transmitting signal from transmitting circuit.
  - T15: Verify signal received by attaching output of IR receiver and output of transmitting IR LED on a oscilloscope.
- TS18: The receiving peripheral circuit connects to RPI through GPIO pins.
  - T16: Check whether any GPIO pins were used.
- TS19: Decoded signal must be amplified before played through a speaker.
  - T17: Check whether an amplifier is set up before the speaker.

#### Receiving Circuit

- TS20: Use an IR receiver component similar to Vishay 57236.
  - T18: Check receiving circuit component used.
- TS21: Decode the transmitting signal.
  - T19: Check whether decoded signal relates to the signal before it was encoded.
- TS22: Filter out any unwanted signals and noise.
  - T20: Check whether terminal of receiving RPi is detecting any noise or unwanted signals.

### **Speaker & amplifier:**

- TS23: lowest Vpp that the amplifier circuit can amplify without distortion is lower than the maximum output Vpp of the Raspberry pi
  - T23: Generate audio signal by the audio code at maximum volume and connect it to the amplifier. Probe the speaker (output) and input to the oscilloscope and check whether the amplifier is able to detect the input signal and create a non-distorted output waveform
- TS24: noise should be as minimal as possible
  - T24: probe the speaker (output) to the oscilloscope and check the waveform

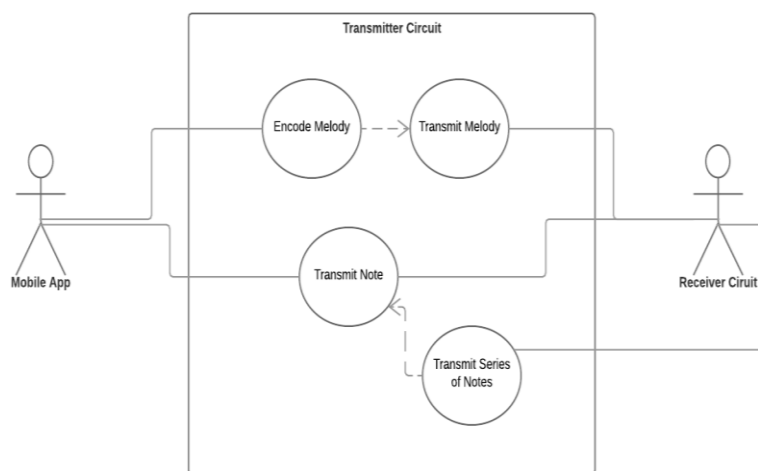
### 3. Subsystem Description

This section describes the various subsystems using a use case diagrams, sequence diagrams, activity diagrams and block diagrams. The interactions of the various subsystems and a state diagram is also described.

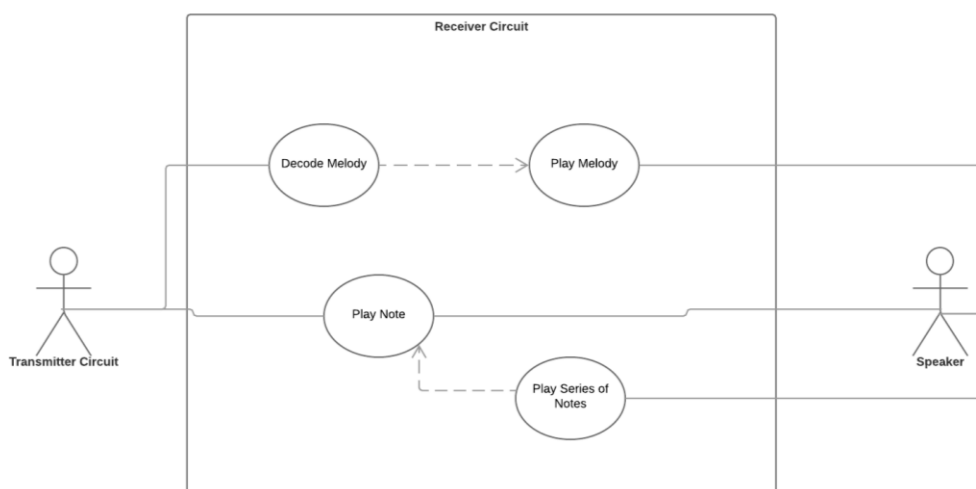
#### 3.1. Use Case diagram

This a non-technical UML diagram that describes the possible actions between various sub-systems and users to ultimately produce an output. It is a high level representation of the full system, that can be broken up to describe the actions of each sub-system with respect to its inputs and outputs, showing how it possibly connects to the next system.

##### Transmitter to Receiver to Speaker Use Case Diagrams



This use case describes the interaction between the user and the transmitter side and the



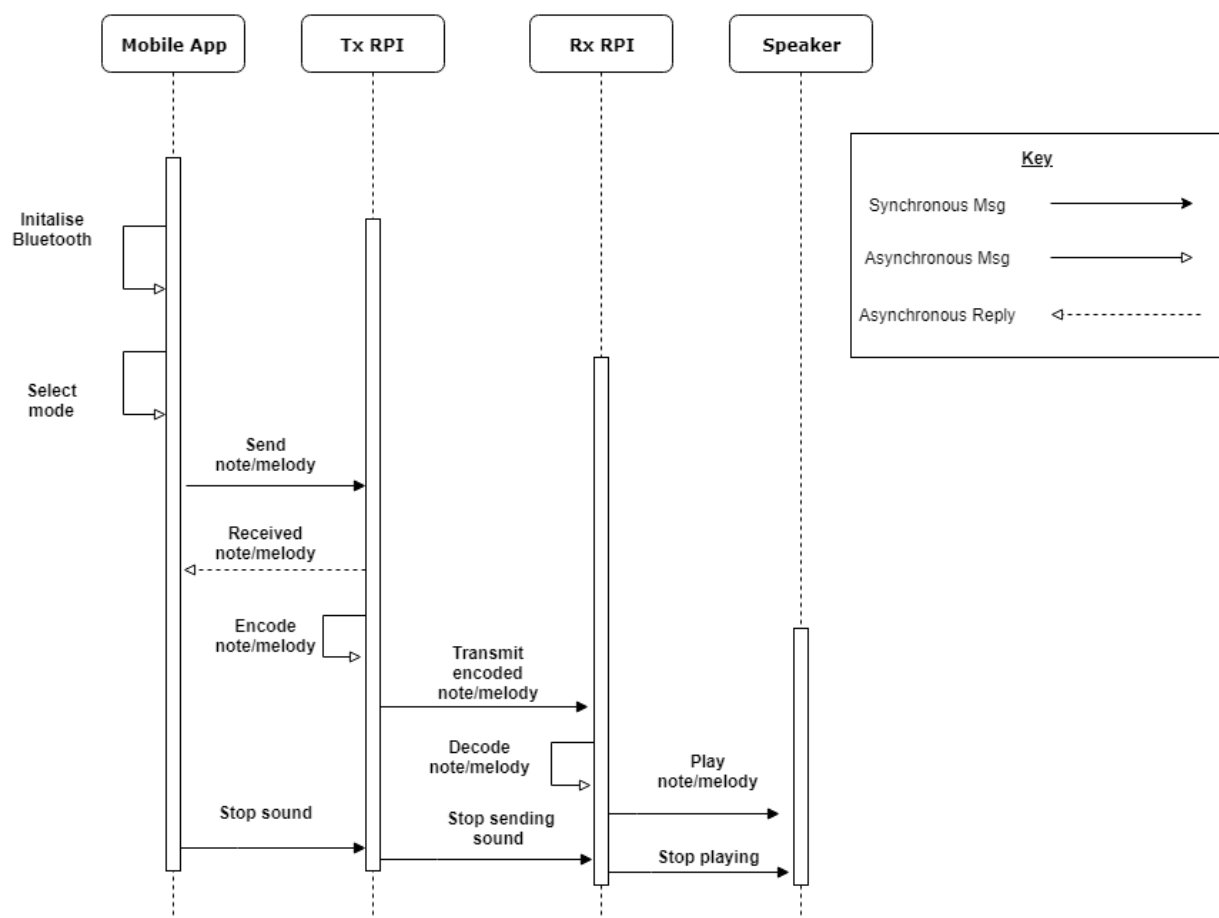
This use case diagram describes the actions that the receiver performs when interacted to by the transmitter side

## 3.2. Sequence diagram

This is a simple UML diagram that focuses on the sequence of events that occur in the system when an action is initiated by either the user or a given input to produce a set of outputs. It also shows how a set of concurrent may possible interact depending on how many inputs can the system be given at any one time.

### Description:

The following sequence diagram shows the interaction between all the subsystems which make up the system. The mobile App



### 3.3. Activity diagram

An activity diagram is a form of UML diagram that describes the dynamic behaviour of a system. It is used to show activity flow as well as describe the sequence from one activity to another. The flow is described as parallel, branched or concurrent.

The activity diagram below shows the sequence of events of the Infrared walkie-talkie. The system is able to send, receive and play tunes concurrently in a full duplex manner using two cell phone devices, two RPi3Bs and two output speaker systems.

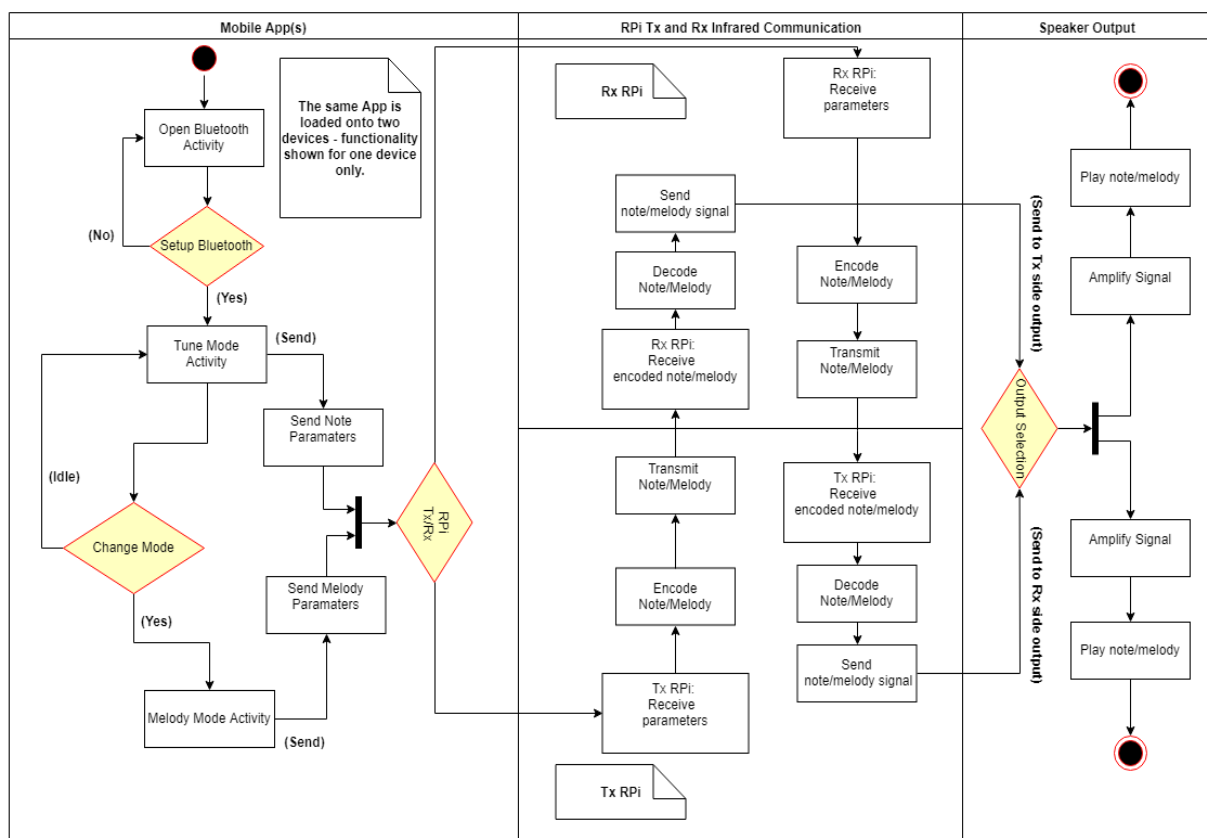
#### Description:

The mobile App section shows that the App initially opens on the Bluetooth Activity where the user is prompted to setup and connect via Bluetooth to the Tx RPi.

Once connected, the user can go to Tune Mode whereby they can send/play a single note to the output speaker. In Tune Mode the user can also move to Melody Mode where they can select multiple notes and send/play through the speaker output.

The RPi Tx and Rx Infrared Communication section shows how notes or melodies are sent in a full-duplex manner both to and from the transmitting and receiving side RPi's. This section is separated into the receiving and transmitting side (labelled Rx RPi and Tx RPi respectively).

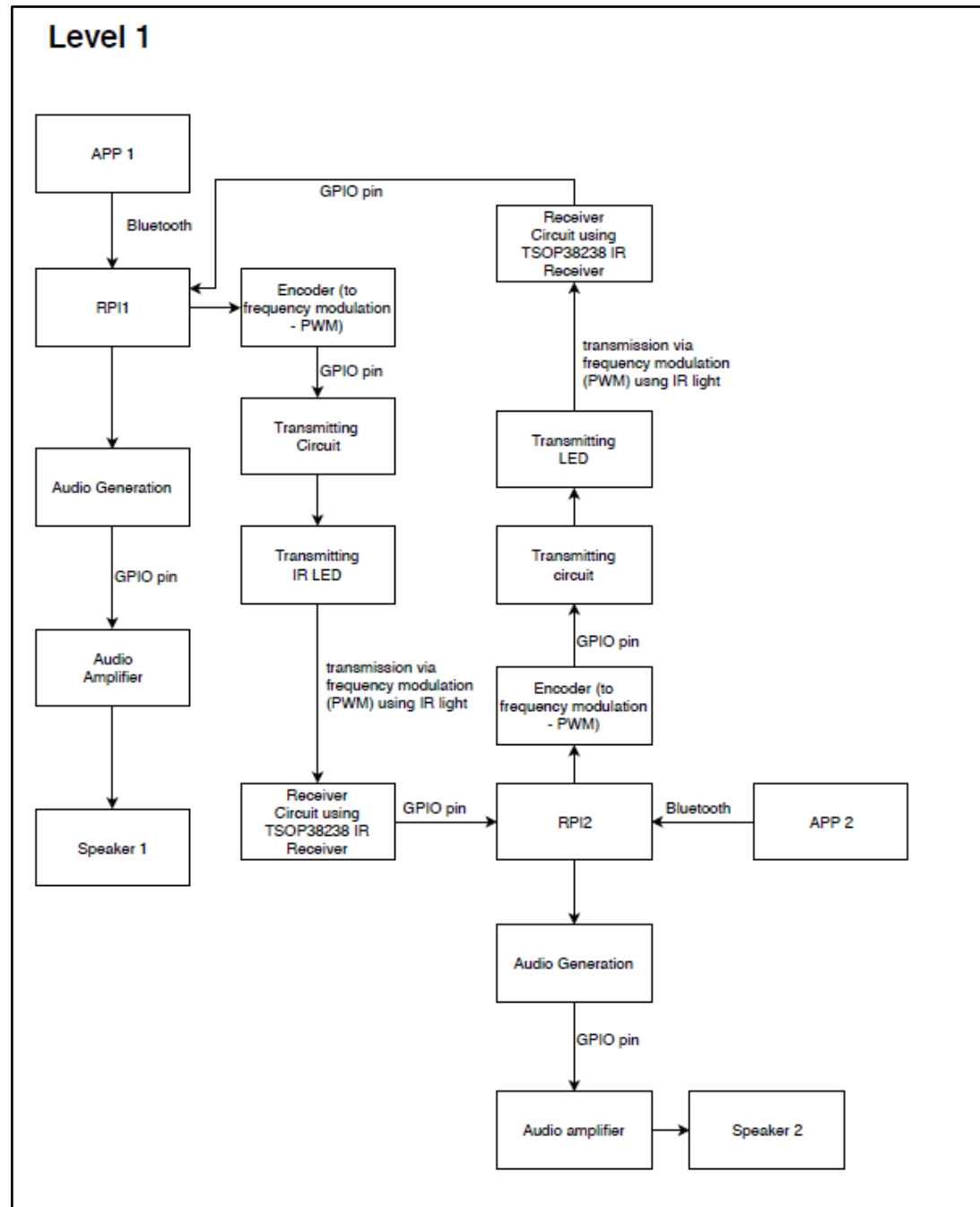
The speaker output section shows how notes and melodies will be played.



## 4. System Descriptions

### 4.1. System Block Diagram

A block diagram is a UML that describes the functional units of a system and the interfaces between the units. It focuses on the actual hardware as opposed to processes/activities and this makes block diagrams especially useful for debugging of systems. Block diagrams have different levels that determines at what magnification the system is observed at. Level 0 block diagrams looks at the external inputs and outputs, level 1 looks at the subsystems and level 2 looks at each subsystem.



The system works in a full duplex manner, which means that both the RPIs can transmit and receive data. The UI of the app allows user to choose a mode and a tune or selection of tunes depending on the mode. The data is sent via bluetooth to the “transmitting” RPI where the data is encoded into a discrete signal of a specific frequency that is amplified in the transmitting circuit and toggles the LED. The IR receiver detects the transmitting signal and just passes the raw digital data to the “receiving” RPI. Any noise is filtered by the IR receiver component. The receiving RPI decodes the signal and then synthesises the required audio using code for audio generation. This is then sent to the amplifier via the audio jack on the RPI where the audio signal is amplified before it is played by the speaker. The full duplex is achieved by repeating the same circuitry in the opposite direction.

## 4.2. System Descriptions

**Table showing description of each sub-system**

Sub-system	Description
Mobile app	Send data in the form of a byte array to indicate selection of single note or selection of multiple melodies. The App should send data using Bluetooth protocol.
Tx RPi	Receive data from mobile app via Bluetooth and send relevant encoded data to Tx Circuit for audio generation.
Tx Circuit	Broadcast signal received from RPi using frequency modulation (PWM) via IR.
Rx Circuit	Detect IR signal from Tx circuit and send the ‘raw data’ detected directly to Rx RPi.
Rx RPi	Decodes the received signal from Rx circuit to obtain the correct information to generate an audio signal.
Audio generation	Use code to generate sound, obtain inputs from Rx RPi and output generated sound to amplifier.
Audio amplification	Audio signal from the Rx RPi is amplified for the speaker.
Speaker	Transforms amplified audio signal into actual sound.



### 4.3. Interfaces

Characteristics	Interface 1: Mobile app to Tx RPi	Interface 2: Tx circuit to Rx circuit	Interface 3: Receiving RPi to Audio amplifier
What data is exchanged?	A byte array associated to the selected tune or melody.	Binary signal	Audio signal
Wired/wireless connection?	Wireless connection using Bluetooth protocol.	Wireless infrared connection	Connected via aux cable
Speed of data transfer (e.g. kbits/sec)	50-200kb/s	Not calculated yet but using timer library to record and sum up the time it took to receive an array of binary signals.	
Format of data	Selected as string sent as byte array.	Binary 1s and 0s in a binary digit	Pulse Width modulation
What triggers the exchange event?	The configuring of bluetooth and pairing of devices and finally the press of the play button.	Running a C compiled code on the RPi	Rx RPi executes audio generating code when signal is received
When does data exchange stop?	When the pause button is pressed or the duration of the tune or melody is sent.	Stops after the short binary pulse is sent	Exchange stops when audio generating code terminates
Any error checking?	Unit Tests will be used to verify correctness of functions as while debugging loops to ensure accuracy of App itself.	There was no error checking at this stage of development	No error checking

#### Table showing all the interfaces with their characteristics

Tests that will be done to assess if the interface is working as expected:

#### Interface 1 testing:

##### Button Tests:

1. On/Off
2. Enable Discoverability
3. Discover
4. Pair

## 5. Send

### Bluetooth Connectivity

The connectivity of Bluetooth is tested using the logcat window within android studio the data log messages are placed in critical sections of the code.

Further verification of connectivity is assessed on the RPi Tx side, this is done using VNC and the serial port console shown in the terminal.

### Sending Data

The sending of data is verified both in android studio logcat window and the RPi terminal. This is verified by having the actually message sent displayed on the serial port console window as well as a verification message on the App.

### **Interface 2 testing:**

By connecting both the transmitting signal and received signal to the same oscilloscope, we can compare whether the transmitting signal correspond to the received signal in any way. To verify the IR LED is sending any IR light, a smartphone camera can be used to check whether the IR LED emits a purple glow.

### **Interface 3 testing:**

The interface between the RPi and the amplifier can be checked by comparing the generated signal from the RPi and the input signal to the amplifier.

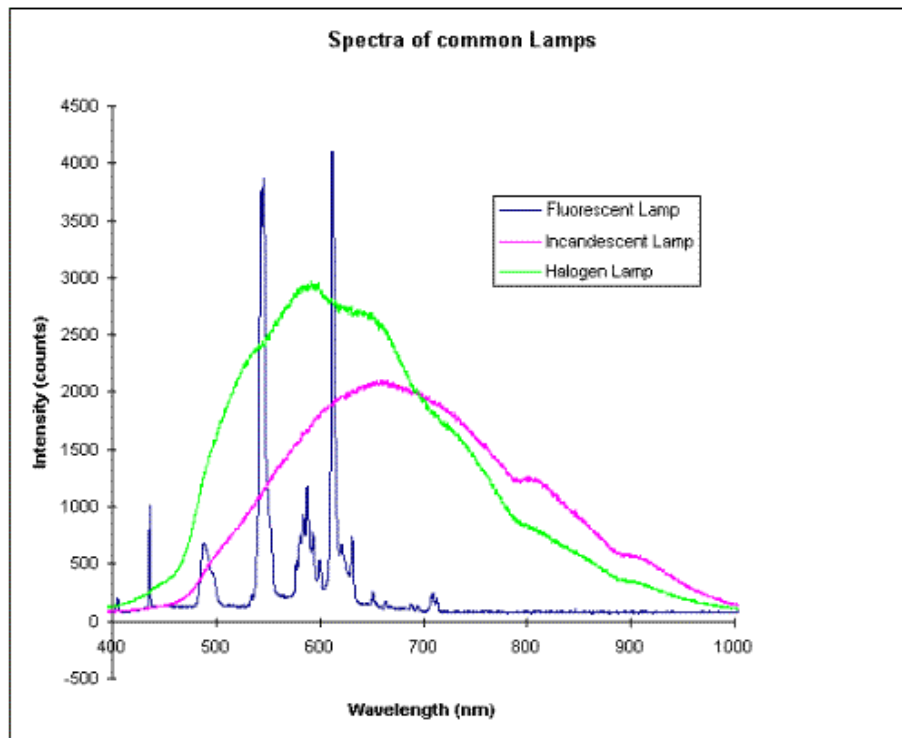
If the signal at the input point of the amplifier is the same as the generated signal for the various frequencies and amplitude, the interface passes the test.

## 5. Identification of different approaches to problem solving and final selection

### **Different Approaches for transmitting signals wirelessly**

Implementation	Advantages	Disadvantages
Infrared light	<ul style="list-style-type: none"><li>● High directionality</li><li>● Safe for humans</li><li>● High speed communication – data rate at 1Gbps</li><li>● High security – no risk of information being dispersed</li></ul>	<ul style="list-style-type: none"><li>● Requires to be line of sight for communication between receiver and transmitter</li><li>● Devices can't move around during transmission</li><li>● Used for very short distances</li></ul>
Visible light	<ul style="list-style-type: none"><li>● Fast-switching</li><li>● Safe for human beings</li><li>● Supports a large bandwidth</li><li>● Low power</li></ul>	<ul style="list-style-type: none"><li>● Has interference issues from ambient light</li><li>● Supports a short coverage range</li><li>● Both should be line of</li></ul>

	consumption	sight there is no line of sight communication <ul style="list-style-type: none"> <li>Other drawbacks – atmospheric absorption, shadowing and beam dispersion</li> </ul>
Radio Waves	<ul style="list-style-type: none"> <li>Has different penetration through objects based on frequency</li> </ul>	<ul style="list-style-type: none"> <li>Unsafe if radiation is uncontrolled</li> <li>Not very secure signals can easily be detected by other sources</li> </ul>



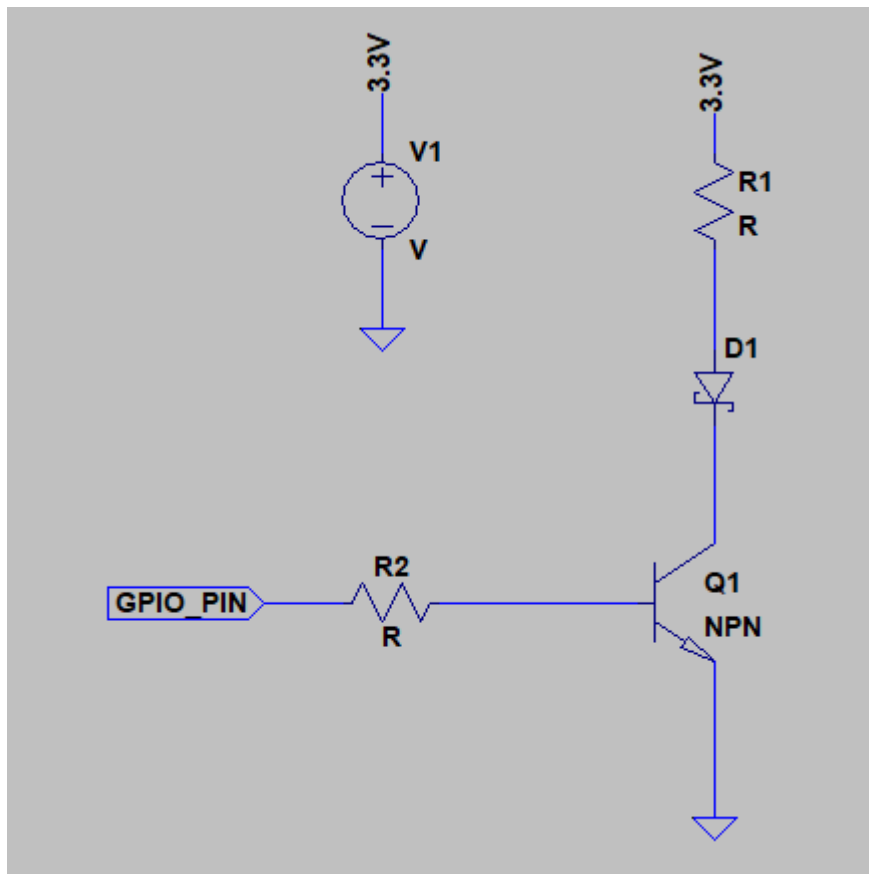
Wavelength of infrared extends from the nominal red range at 700 nm to 1 mm. Given that the environment, where the transmission and reception will take place, is indoors (lit with artificial lights with the spectrum characteristics above) it makes sense to use infrared over the visible light since the intensity of various lamps die down at greater wavelengths.

For fluorescent lamps, the intensity almost completely dies down at the infrared range. Since most of the indoor light systems are fluorescent lamps, there should be very little interference/noise at the receiver.

From these advantages and disadvantages we selected the use of IR light to transfer our data due to it being very direct and secure with a high rate of data transfer. Even though the distance range is not the best, this is the compromise we are willing to make.

## 5.1. Hardware design and list of components

### Transmitting Circuit:



This is the standard transmitting circuit to be used we will be using 2 of these circuits. First circuit for continuous data transmission and second circuit as a remote control for sending instructions for effects, volume and itch manipulation.

Below is a table comparing different IR LEDs and the required R1 resistor for each one, table also contains the NPN transistor to be used due to its high speed switching capability (works well if the base pin is being switched from high to low at high frequencies).

IR LED TYPE	Wavelength	Maximum Forward Voltage Drop / V	Maximum Rating, Forward Current / A	Biassing resistor R1 needed(minimum value) / ohms	Cost / Per individua l	Cost / Standard Bundle
TSHG5410	850nm	1,8	0,1	18	R4,65	R93,02
TSAL6200	940nm	1,6	0,1	16	R2,99	R74,80
SFH 4554	860nm	1,7	0,1	17	R4,57	R114,25
Transistor						
NPN TIP50					R3,63	R181,30

Note\* The Maximum Forward Voltage and Maximum Rating Current was taken from the datasheets of the LEDs to be considered - the governing equation to find the value for Biassing resistor R1

is as follows:

$$R1 = \frac{\text{Maximum Forward Voltage Drop}}{\text{Maximum Forward Current rating}}$$

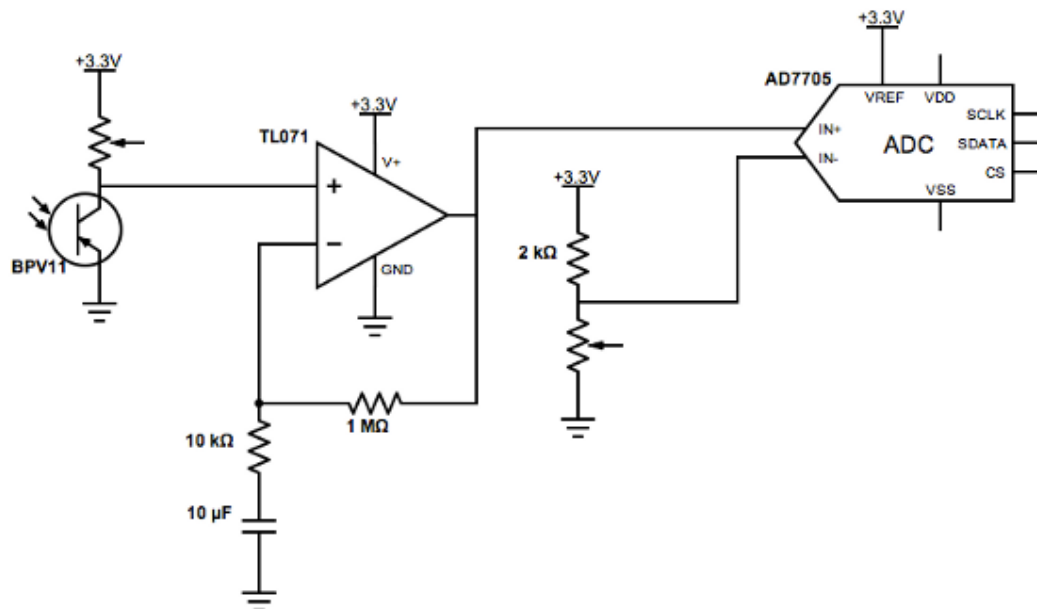
### Receiving Circuit:

This receiving circuit has the following requirements:

- All components must fit onto veroboard.
- Must be able to receive signals from IR LED (transmitting circuit) and relate the “raw” data to receiving RPI.
- Must operate from RPI GPIO pin of 5V or 3V3.

Three different circuits that satisfy the above requirements will be described below:

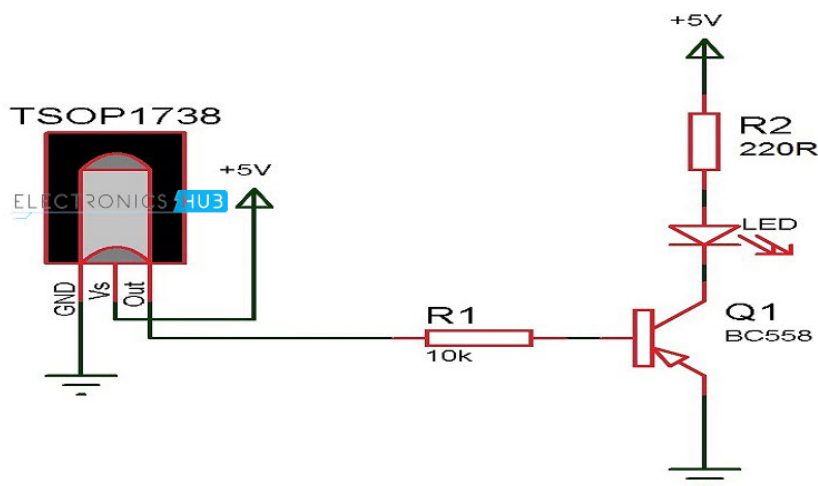
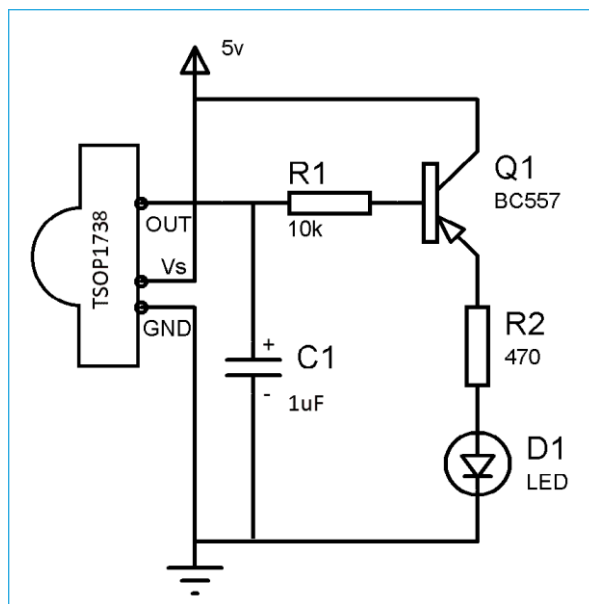
1.



Suggested receiver circuit (based on <http://web.eng.gla.ac.uk/RPI/2013/4/>)

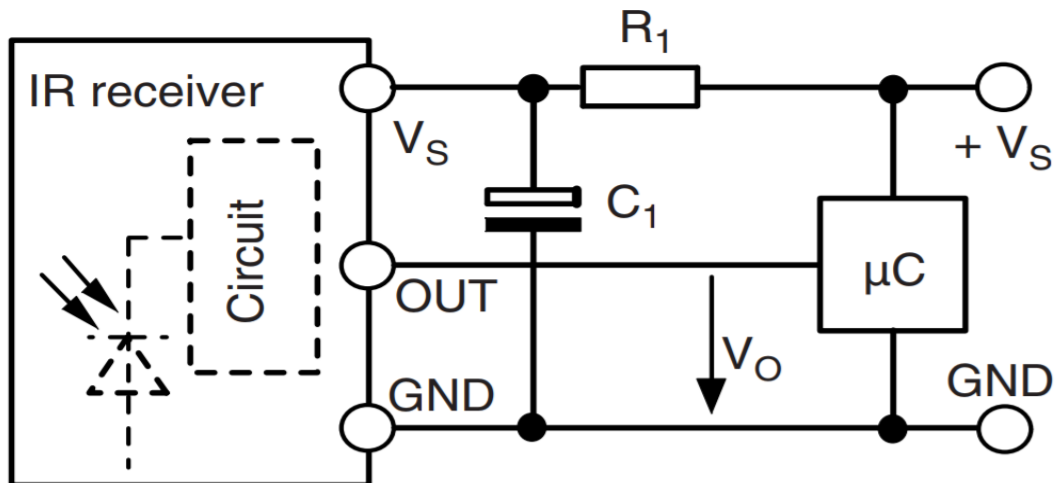
This circuit has a phototransistor to sense the transmitting signal. It will then send the signal through a AC-coupled non-inverting op-amp configuration which will only amplify signals that change with time. This not only makes the circuit easier to bias, it also reduces any noise which might exist in the frequencies below those of interest. From there, it will be sent to an ADC to convert the analogue signal to a digital signal for the RPI to process.

2.



Instead of using the TSOP1738, we will be using the TSOP38238. This IR receiver is tuned at 38kHz. It runs from 2.5-5.5V so it can be powered from a RPI. The above circuit configurations are optional add-ons to help test the TSOP38238 receiver by connecting an LED at the output.

The TSOP38238 is a through hole component so compatible to use on a veroboard. It has a maximum transmission distance of 45m so the IR communication will be limited by the transmitting LED. To use, connect pin 3 (all the way to the right) to 5V power, pin 2 (middle) to ground and listen on pin 1.



This is the application circuit from the TSOP38238 datasheet, “R<sub>1</sub> and C<sub>1</sub> are recommended for protection against EOS. Components should be in the range of  $33\ \Omega < R_1 < 1\ \text{k}\Omega$ ,  $C_1 > 0.1\ \mu\text{F}$ .”

3.



<https://za.rs-online.com/web/p/ir-transceivers/7103850/>

The use of an IR transceiver like the TFBS4711-TT1. Since both our RPI needs to be able to send and receive tunes, an IR transceiver component is an option.

Circuit	Advantages	Disadvantages
1	Not expensive, lots of phototransistors to choose from.	Most amount of components used, need to make use of ADC as components are mainly passive, more complex as many places where circuit problems can occur. Difficult to debug.
2	Easy and simple to use. Can use component alone. Has immunity against HF and RF noise. Low supply current. Photo detector and preamplifier in one package Internal filter for PCM frequency	It does not do any decoding of the signal. Slightly expensive.
3	All packaged into one component.	Most expensive component, complex to use a transceiver, no through hole components found on RS.

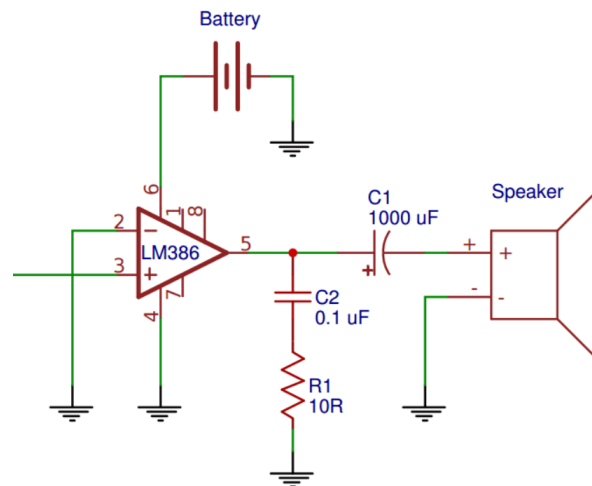
Component	Quantity	Price	Total
TSOP38238	5	7,954	39,77
TFBS4711-TT1	1	55,52	55,52

After analysing the different circuit, we have chosen **circuit option 2** to implement. It meets all the sub-system requirements and is suitable for our design optimisation criteria.



## Audio amplifier circuits:

### 1.) simple audio amplification using LM386 op amp



### 2.) amplification using a BJT

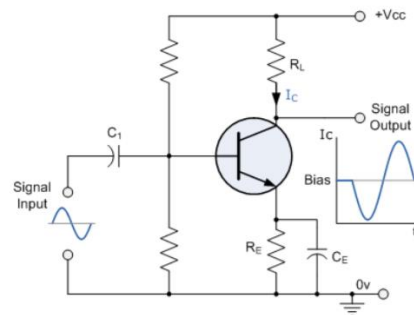
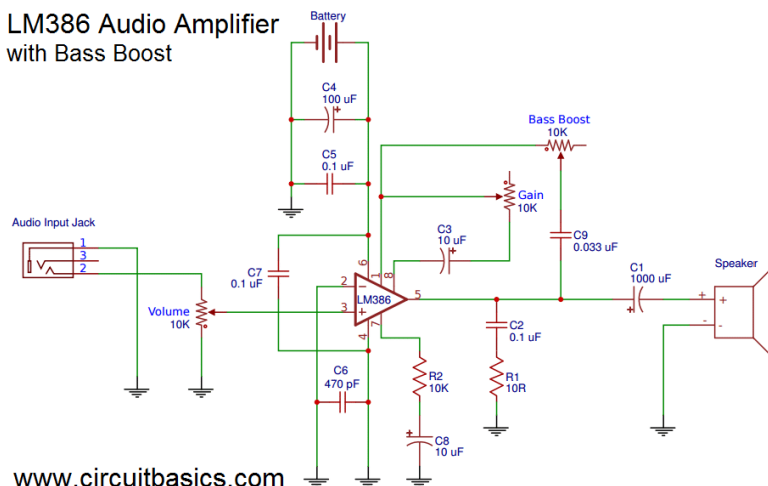


Figure 4: Sound amplifier for connecting to a speaker

### 3.) audio amplifier with a bass boost

LM386 Audio Amplifier  
with Bass Boost



The audio amplifier needs to amplify the audio signal from the RPI with minimal distortion and meet the specification (ie. power rating) of the speaker.

The BJT amplifier has little advantage over the op amp amplifiers. BJTs, if not biased properly, can undergo saturation and cut-off when the audio signal fluctuate slightly high.

Audio amplifier with a bass boost is a good use of the additional features of the LM386 which allows the bass boost functionality. However, bass boost is not necessary for the scope of the project and the circuit requires more complexity and cost.

Therefore, the first implementation: the simple op amp audio amplifier will be used.

## 5.2. Software tools for App Development

Initially the advantages and disadvantages of Web Apps and Native Apps are compared.

### Web App:

#### Advantages:

- Easy to build as it uses simple web languages such as HTML, CSS and JavaScript.
- It is easy to maintain the application with necessary updates and latest features.
- Any technology/language can be used to integrate with the Web App (i.e. Python, Java etc)
- Developers are generally cheaper to hire, thus making the project cheaper.
- A single app may be used across all platforms.

#### Disadvantages:

- Needs to run in browser.
- Slower than a native app due to possible internet connection issues.
- Less interactive and intuitive than native app.
- Poor interaction with device utilities (i.e. camera etc).
- Cannot be submitted to app stores easily. Would require native integration.

### Native App:

#### Advantages:

- Very fast.
- Interactive and intuitive.
- Good interaction with device utilities.
- Can easily be distributed in app stores.

#### Disadvantages:

- Single platform.
- Written using difficult languages such as Java and Swift.
- Require the hiring of expensive developers.
- Harder to maintain.

Further research into the most effective frameworks was done and the results are summarised below.

## **Mobile Development Frameworks**

### **1. Ionic**

#### **Advantages:**

- Easy to get up and running.
- Cross-platform.
- Based on Angular, main development in HTML, CSS and JavaScript.
- Good looking user interface.

#### **Disadvantages:**

- Less performance as it is not completely native.
- Not necessarily familiar to ECE students.

### **2. Android Studio**

#### **Advantages:**

- Tight integration with all android devices.
- Familiar to E.C.E students as it is introduced in CSC2002S.
- Native development (benefits of this are discussed above).
- Uses Java language which group members are most familiar with.
- Many helpful resources online.
- Open source and free.

#### **Disadvantages:**

- Single platform can only develop android applications.
- Official google IDE.

### **3. React Native**

#### **Advantages**

- Cross-platform.
- Hot reloading which means the developer can keep the app running while making changes. This increases productivity.
- Good looking user interface
- Very good performance for a hybrid development tool.
- Backed by big companies such as Facebook, Airbnb etc.

#### **Disadvantages:**

- Less smooth navigation than in native navigation.
- Requires native developers.
- Lack of some custom models.

## **Conclusion:**

For the purposes of this project, many of the disadvantages associated with building a native app may be negated:

- Single platform.

The project description only requires the app to be built for android devices.

- Written using difficult languages such as Java and Swift.

Third year students have been introduced to Java and arguably have the most experience in this language, for this reason the difficulty associated with coding in Java can be negated.

- Require the hiring of expensive developers.

This project requires no extra developers and the developers working on it are all well versed in Java.

- Harder to maintain.

Finally, the maintenance involved with the app should be minimal as its functionality is quite limited.

For these reasons the application will be developed in a native environment.

In evaluating the different mobile frameworks, it was decided that Android Studio would be the chosen framework. This is because it is a native platform and the benefits of this have been discussed above, it is furthermore familiar to ECE students and has a wide range of resources to guide the project.

## 6. Identification of sub-systems and sub-system requirements

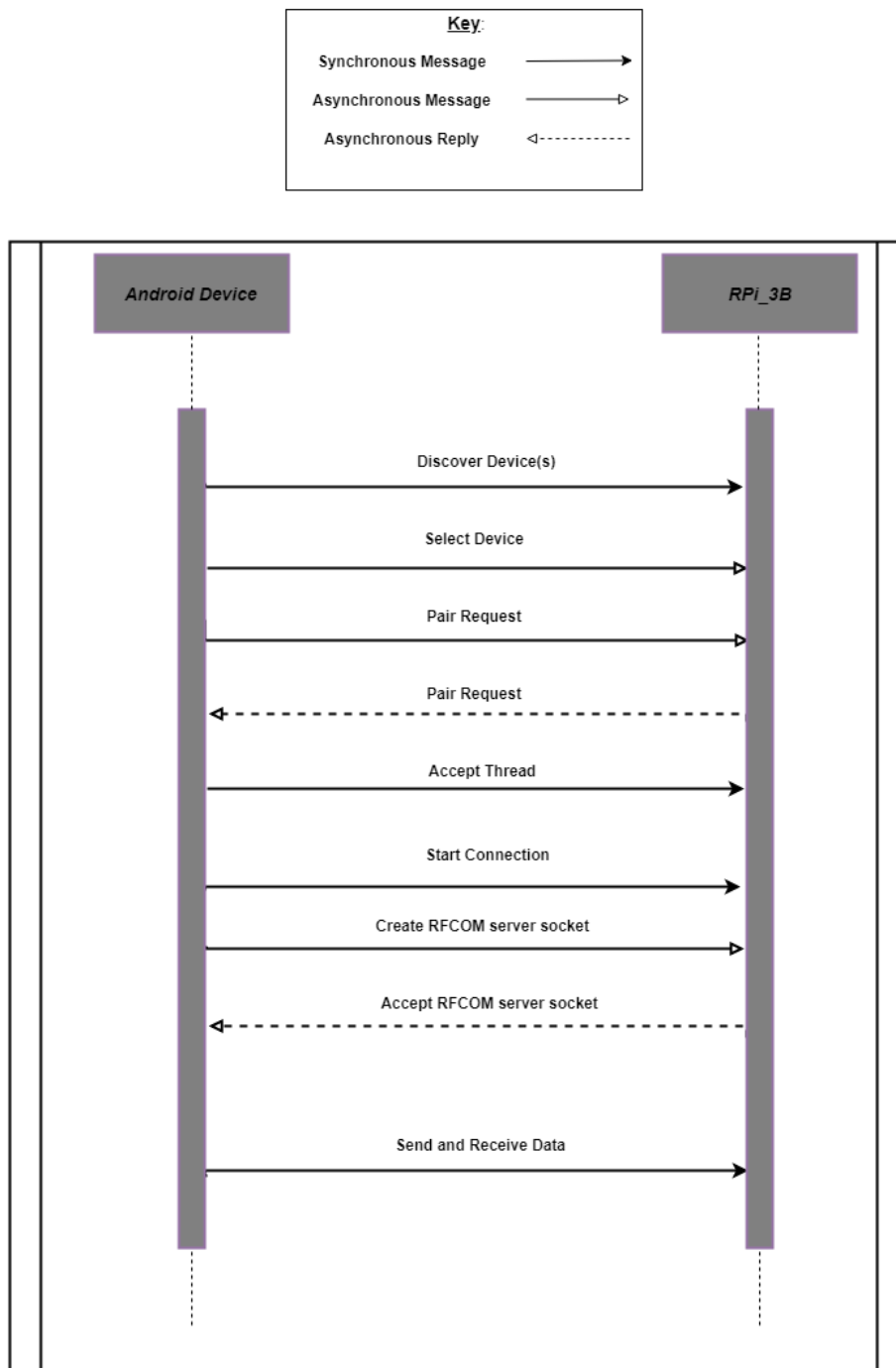
### 6.1. Mobile App Subsystem

#### 6.1.1. Technical requirements for App communicating with RPi

- **TS1:** The App should be compatible with android devices.
  - **Test\_1:** Build and run on an android device (J5 prime) and emulator (Nexus 5 API 23).
- **TS2:** The App should switch on and off Bluetooth.
  - **Test\_2:** Pressing on/off button turns on/off Bluetooth verified on device and in android studio Logcat output box (via log message).
- **TS3:** The App should enable the device to be in discoverable mode.
  - **Test\_3:** Verified in android studio Logcat output window.
- **TS4:** The App should discover the Tx RPi.
  - **Test\_4:** Verified on App Interface (TextView).
- **TS5:** The App should pair with the Tx RPi.
  - **Test\_5:**
    - Clicking start connection button on App with RPi3 initiates dialog box with pair request on both the App and RPi.
    - Message dialog box reads paired successful.
- **TS6:** The App should send data to the Tx RPi.
  - **Test\_6:**
    - Verified in App Logcat output box.
    - Verified in terminal using Serial Port console. This is implemented in the RfComm server python script that is used on the Tx RPi.
- **TS7:** App should be able to send note and melody parameters to Tx RPi
  - **Test\_7:** Verified in terminal using Serial Port console window.

## 6.1.2. UML diagrams

### Bluetooth Sequence Diagram:



## Mobile App Activity Diagram:

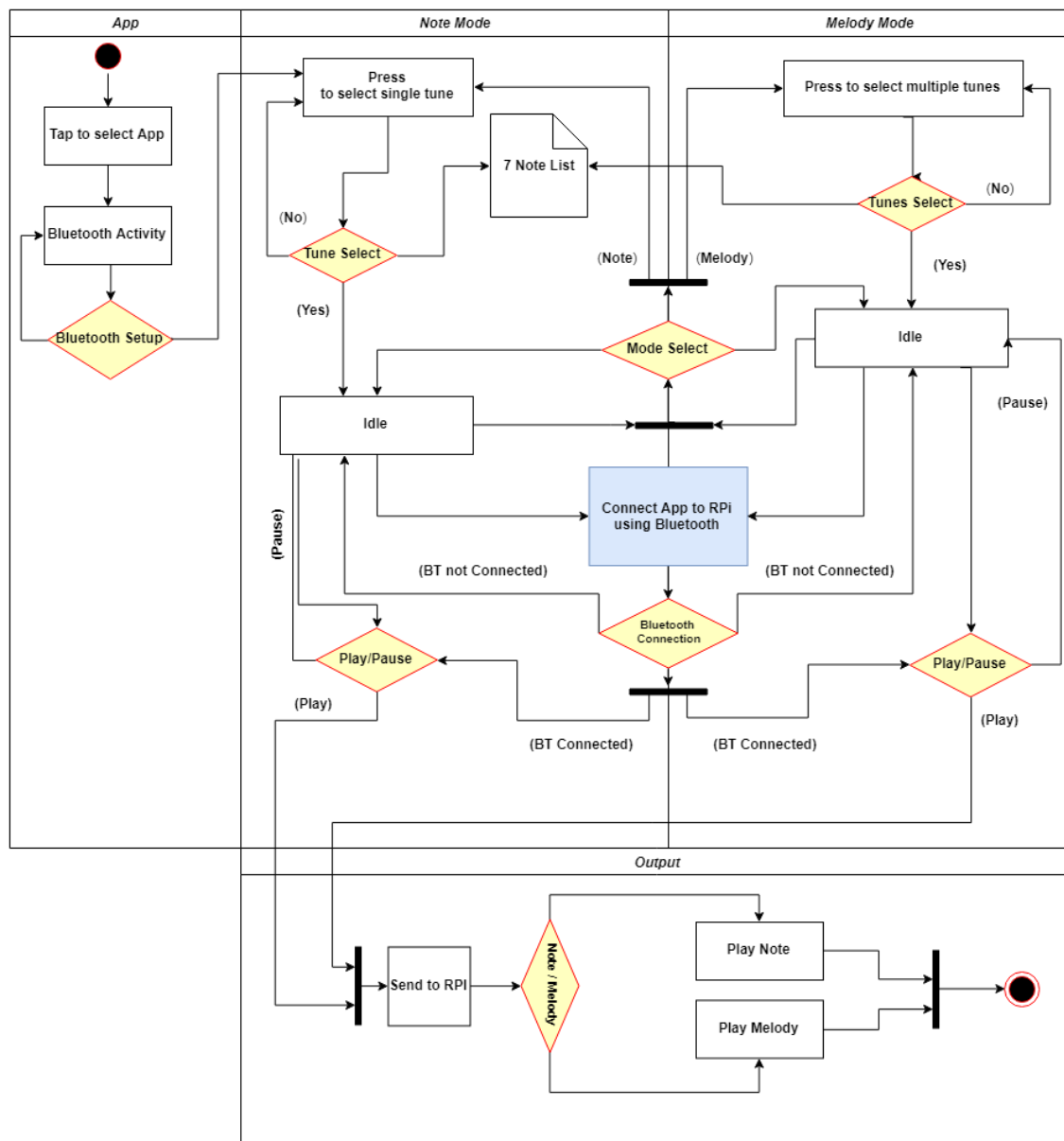
### Description:

The diagram below consists of three sections namely the App (Bluetooth), Note Mode and Melody Mode. The first section shows that the App opens on the Bluetooth Activity screen where the user is prompted to setup and connect to the Tx RPi via Bluetooth.

Once Bluetooth is setup the user can go to Note Mode where the user can select and play a single note from a selection of 7 notes (A to G).

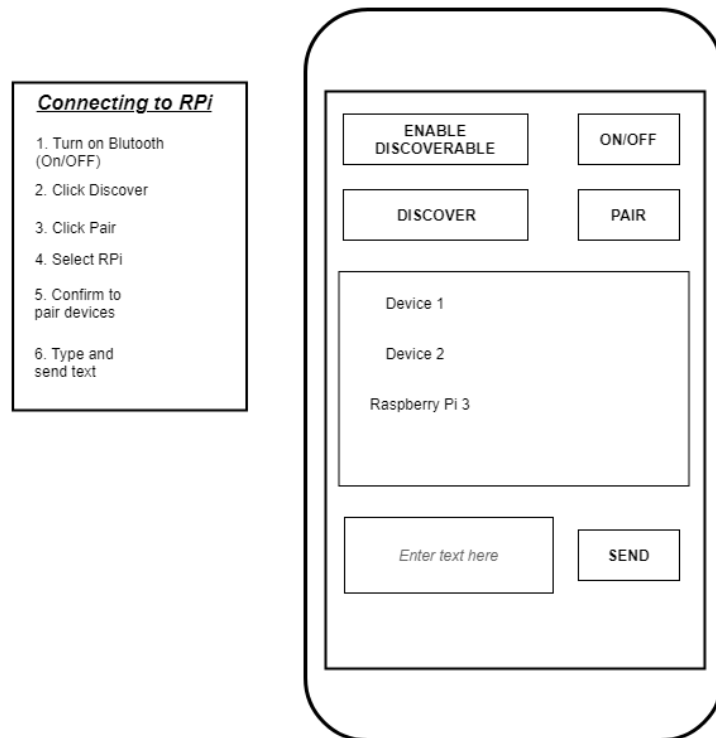
The user can also change modes to Melody Mode where they can select multiple notes and play a melody.

At any stage the user can switch between modes or disconnect/reconnect the bluetooth connection.



### 6.1.3. User interface

This is the paper design for the App. This user interface is used to send data from the android device to the Tx RPi. This data will be sent as byte array. For the purposes of testing a text message will be sent from the App to the Tx RPi .



Once the base design was developed, further progress towards the final android App design could be made. Additions to this design are the mode selection, tune selection and play/pause button.

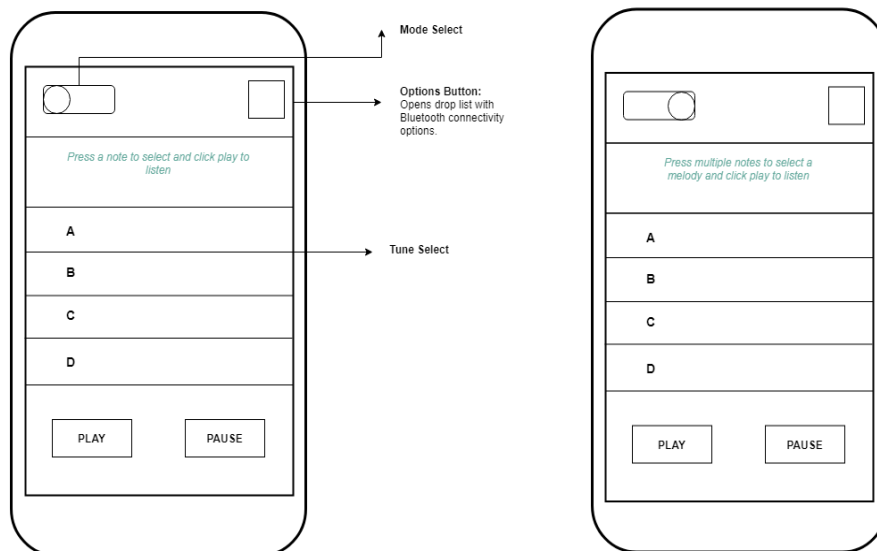
Mode Selection: Switch between note and melody mode.

Tune Selection: Depending on mode select note or melody to play.

Bluetooth Button: Bluetooth configuration/reconfiguration.

Play/Pause: To stop or start music.





Further adaptations were made to the design above which lead to the final design shown below.

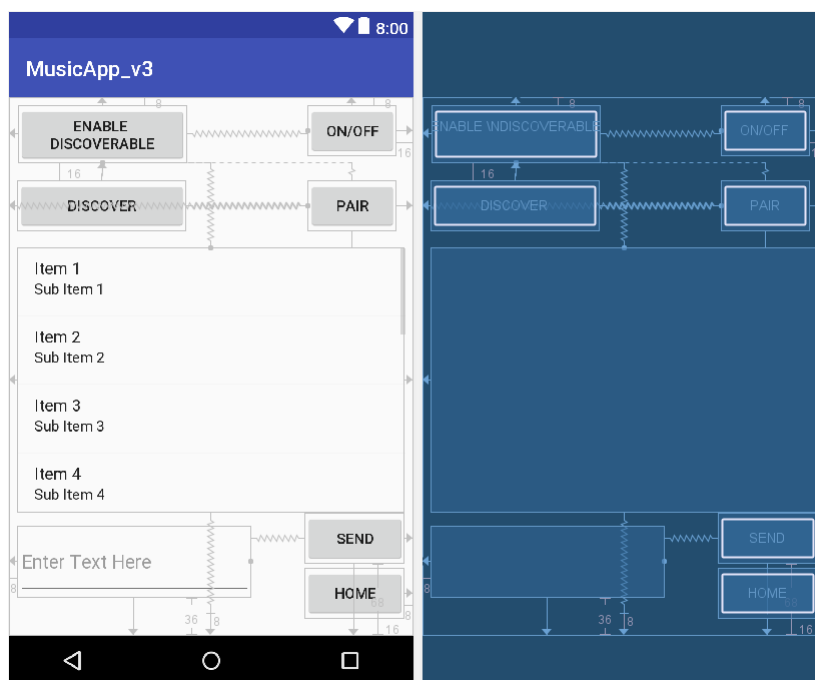
The notable changes from the design above were the mode select button which was changed from a switch to a button for simpler use.

The options button was removed, and a Bluetooth button was added to both Note and Melody Modes.

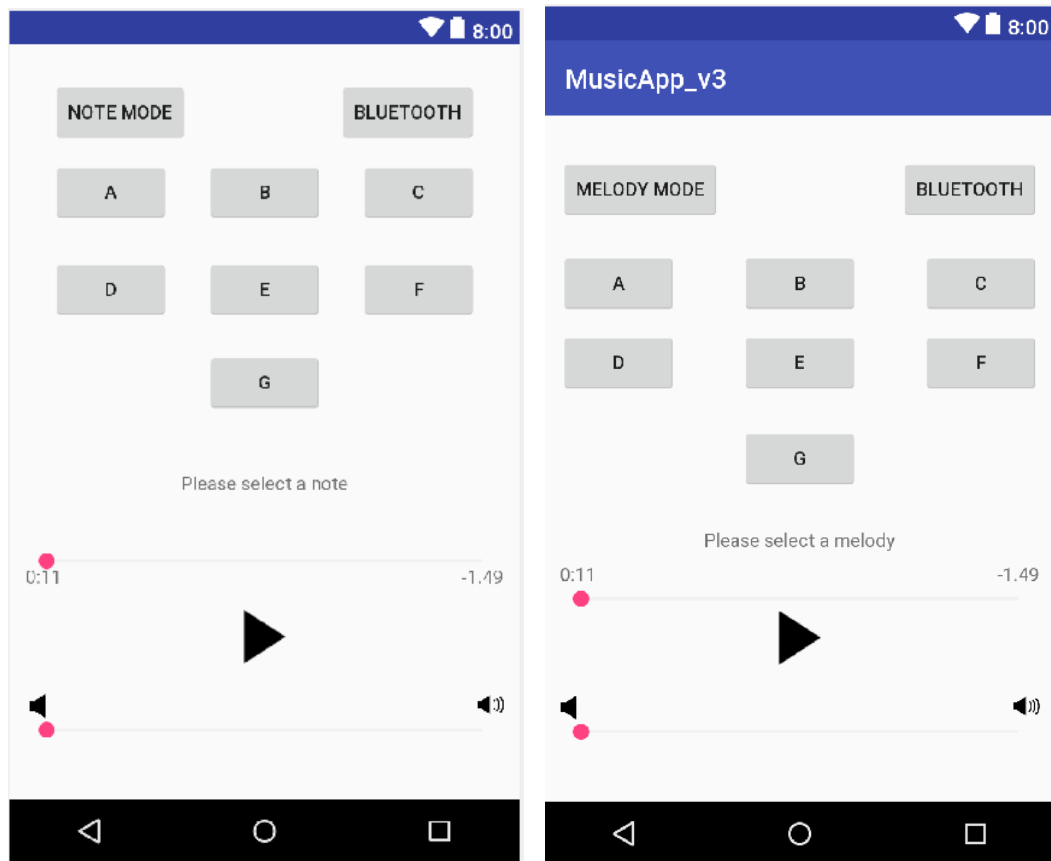
The user interface was updated to display the notes in a manner not requiring a scroll view. Selection of notes in both modes is now displayed to the user via a TextView format.

Finally, seekbars were added for the user to be able to select duration and volume for the notes or melodies they select to play.

### **Bluetooth Activity:**



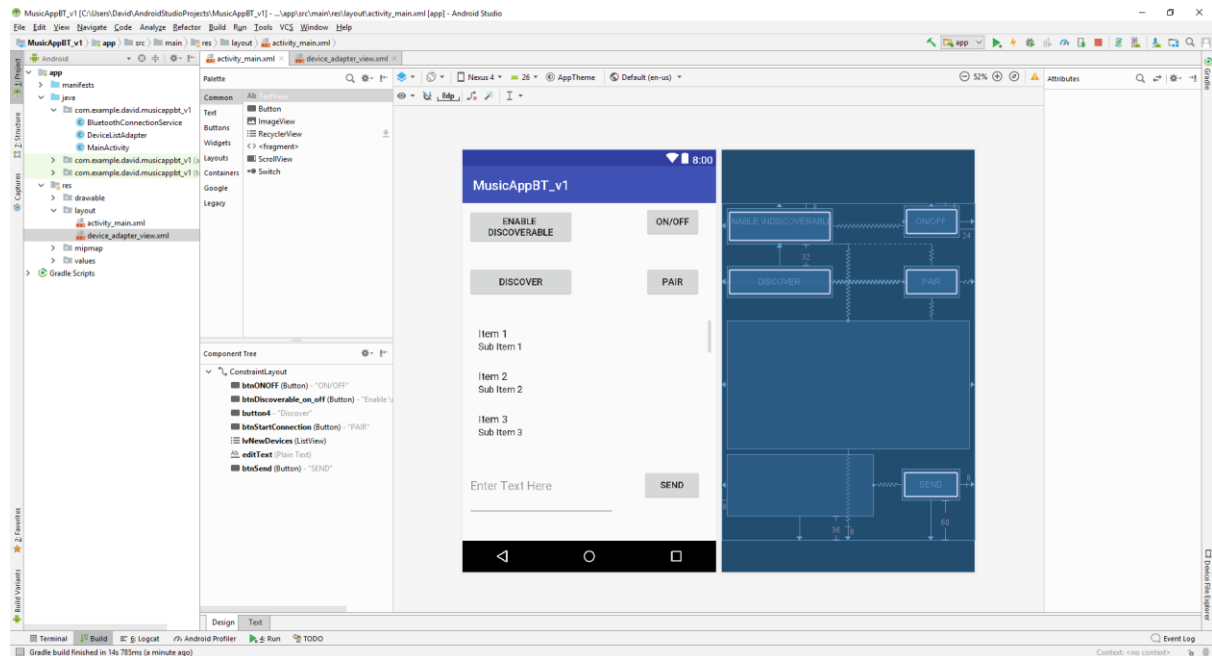
### Note and Melody Activities:



## 6.1.4. Build and Testing

### User Interface Designed in software:

Below is an illustration of the user interface designed in software.



### How user interface was tested

The Android Studio App was tested using the android J5 prime mobile device and Android Emulator, Nexus 5 API 23.

The App was configured to use Bluetooth protocol to connect with a Tx RPi and send data in the form of a byte array.

To test the bluetooth protocol; bluetooth is switched on for both devices, enable discoverability is selected on the Tx RPi at which time the discover button on the android App can be clicked.

Within a few seconds the Tx RPi should show up in the TextView section of the Apps GUI. Once user clicks on the specific Tx RPi device, pressing the pair button will ensure pairing on both devices takes place.

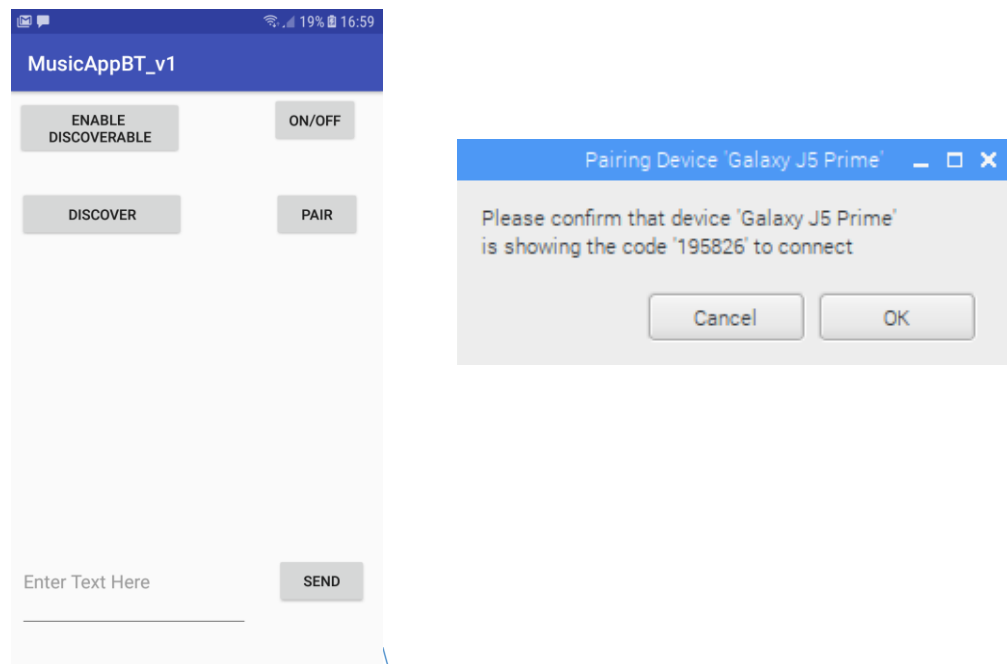
Once both devices are paired a text message can be sent from the Tx RPi using the Text Field and send button shown in the design above.

To verify that the data has been sent across devices. A log message within the send method on the App side can be seen in Android Studio.

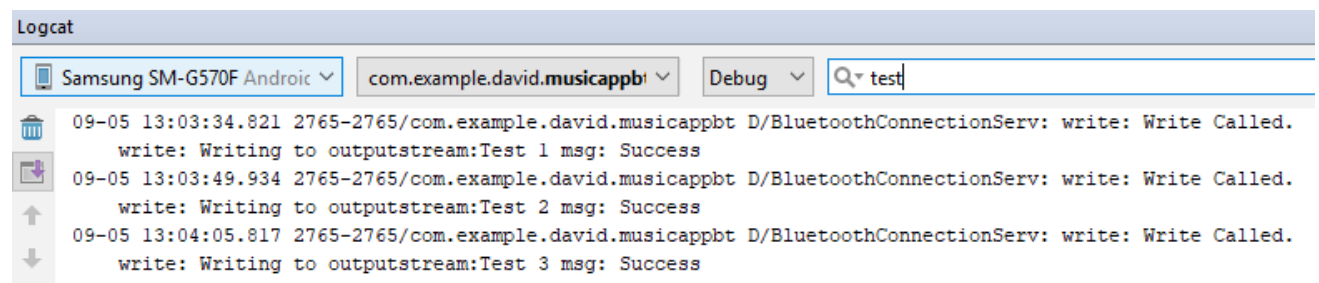
On the Tx RPi a serial profile port console is used to display the data sent from the App this is shown in the terminal screenshots below.

## Screenshots of Testing

The App screen and message dialog box below show the App connecting to the Tx RPi. This meets the test specifications for TS\_1 to TS\_5 and validates Test\_1 to Test\_5.



The Android Studio Logcat window and Linux terminal below, show that a message is successfully sent from the App to the Tx RPi using Bluetooth. This satisfies TS\_6 and Test\_6.



```
pi@raspberrypi:~/Desktop $ sudo hciconfig hci0 piscan
pi@raspberrypi:~/Desktop $ sudo python rfcomm-server.py
Waiting for connection on RFCOMM channel 1
('Accepted connection from ', ('58:C5:CB:17:33:98', 1))
received [Test 1 msg: Success ]
received [Test 2 msg: Success ]
received [Test 3 msg: Success ]
disconnected
all done
pi@raspberrypi:~/Desktop $
```

The final screenshot below satisfies TS\_7 and successfully validates Test\_7. The screenshot shows the sending of both single note parameters and melody parameters.

The array which is shown to be received in the console window consists of two boolean flags to notify the Tx RPi of sending and the respective mode. These flags are followed by the actual notes, duration and volume parameters.

```
pi@RPiDavid:~ $ cd Desktop/
pi@RPiDavid:~/Desktop $ sudo python rfcomm-server1.py
waiting for connection on RFCOMM channel 1
('Accepted connection from ', ('58:C5:CB:17:33:98', 1))
received [true true A 0:56 46]
received [true false ABCDEFGAAA0 0:08 9]
```

### **Summary of Results:**

Test	Success	
Test_1: Android compatibility	Yes	
Test_2: Bluetooth on/off button	Yes	
Test_3: Discoverability button	Yes	
Test_4: Discover button	Yes	
Test_5: Pair button	Yes	
Test_6: Able to send data from the bluetooth App to the Tx RPi/	Yes	
Test_7: Able to send note and melody parameters to Tx RPi.	Yes	

All the tests that were set out for the App subsystem were met, namely Test\_1 to Test\_7 . The results confirm that the technical specifications (TS\_1 to TS\_7) of the software mentioned above were met.

Further additions to the GUI and ease of use will be looked into.

The next major step is full integration with the Tx RPi, this will require the Tx RPi to automatically run the RFCOMM python script which is used to accept a connection between the App and Tx RPi.

Re-connectivity features will also be investigated.

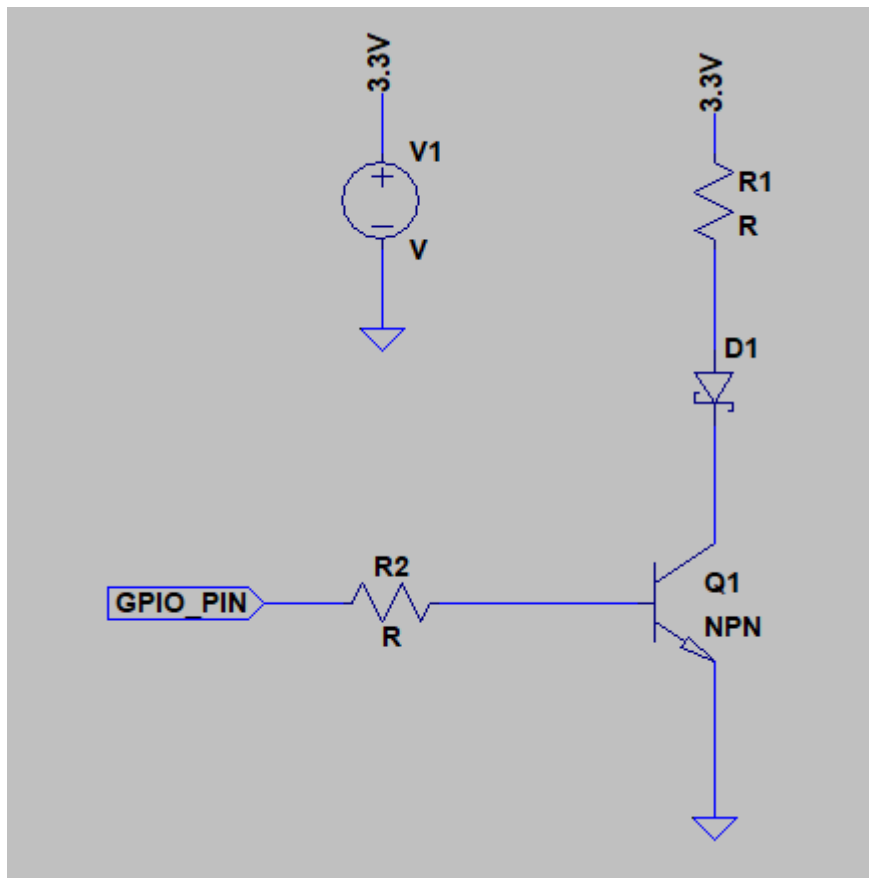
## 6.2.1 Technical requirements for Transmitter Circuit

### Technical Requirements for Transmitting Circuit:

- **TS1:** send encoded data(binary 1s and 0s) from Raspberry pi using an IR Led configuration
  - **Test\_1:** pwm wave at a fixed frequency alternating is sent from gpio pin of the Pi and if a square wave at same frequency is observed from an oscilloscope at pins of the IR LED then this specification is met.
- **TS2:** must transmit between a reasonable distance of 2m or more approximately.
  - **Test\_2:** The IR transmitter and the receiver will be placed 2 m apart (measured) and a success is if the signal (arbitrary highs and lows sent) are observed at the receiver end.
- **TS3:** should be a low voltage system using 3.3V to 5V to power the circuit.
  - **Test\_3:** If the transmitter circuit can be run by the RPi3 this requirement would have been met.

## 6.2.2. Calculations, circuit diagrams and simulations

### Circuit Diagram



The change made to this practical circuit is that it is driven by a 5V DC line.

### Components used:

Component in Diagram	Specifications
R1	22 ohms
R2	1000 ohms
D1	TSHG5410 IR Led
Q1	TIP50 NPN (High speed switching transistor

Only calculation done was to find the limiting resistance needed to safely work the IR LED.

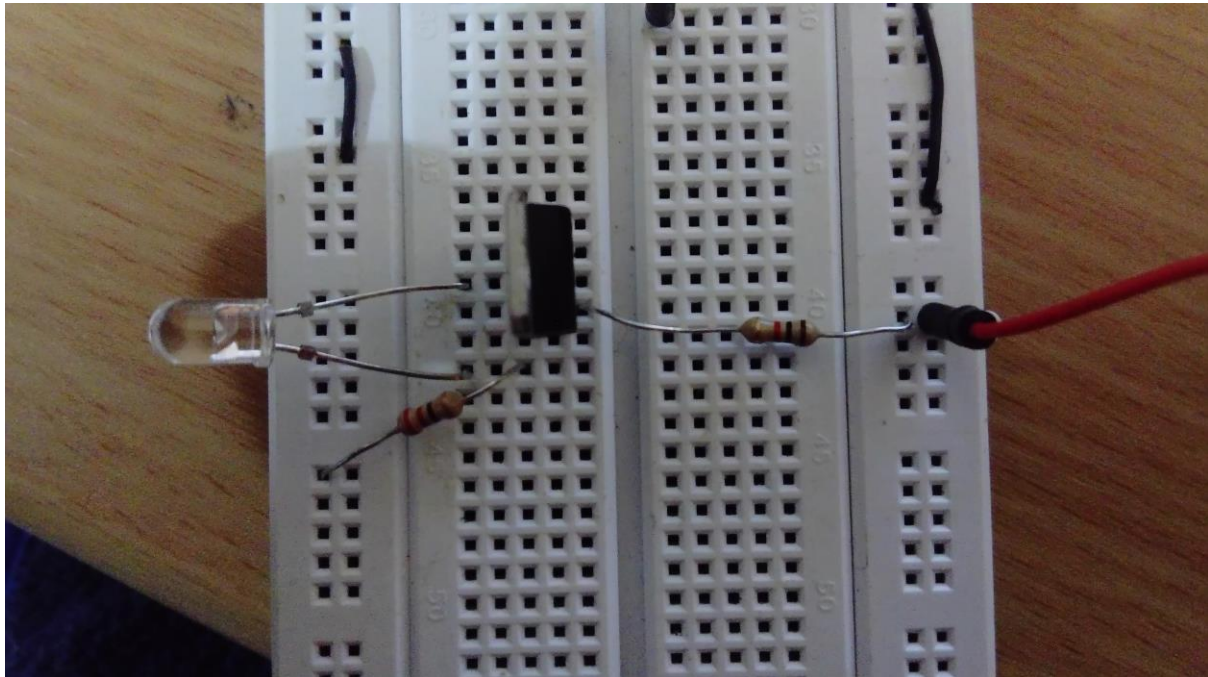
$$R1_{\text{ minimum }} = \frac{\text{Maximum Voltage rating of the IR LED}}{\text{Maximum Current rating of the IR LED}}$$

For this particular IR LED, Max Voltage = 1.8 V, Max Current = 0.1A thus minimum required R1 = 18 ohms. Thus we chose 22 ohms to give maximum power but with a little more protection.

These rating are all found in the datasheet of the IR LED and have been used to calculating the current limiting resistor necessary to safely run this LED. The value found was 18 ohms was found but a value resistor of 22 ohms is used instead to factor in tolerances of resistors such that we remain within the safety range will still operating at maximum power.

### 6.2.3. Build and Testing

#### Physical Diagram



#### Tests Performed:

These tests were performed to capture all the requirements of the transmitting circuit while also observing for any limitations in the hardware for developing of the protocol to be used.

The 2 tests performed for transmitting circuit were as follows:

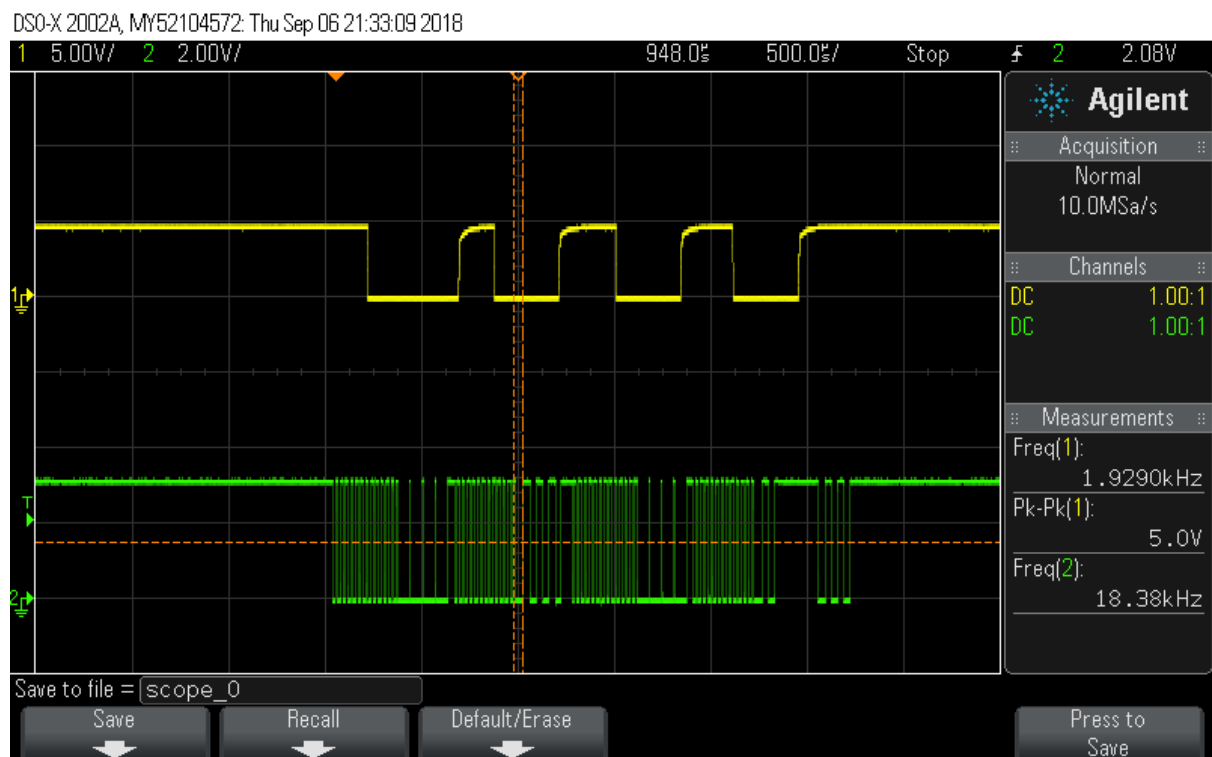
1. Whether or not transmitting circuit sent a binary signal
2. Whether it still sent the same signal over a range of distances

To check for test number 1 an oscilloscope was connected to the leg of IR Led and collector pin of the transistor. And a second oscilloscope was connected to output pin of the receiver and signals were recorded after the binary message was sent from the receiver.

This same test was then repeated with a different binary string. Then for Test 2, Test 1 was done over a range of distances.

**KEYNOTE\***The signal was transmitted at 38 kHz due to the receiver having a set observing frequency 38 kHz thus the transmitting side for the signal to be accurately translated by the receiver, it should be sending at this frequency and this was verified during testing.

### Results:



**Yellow line shows receiver end binary string**

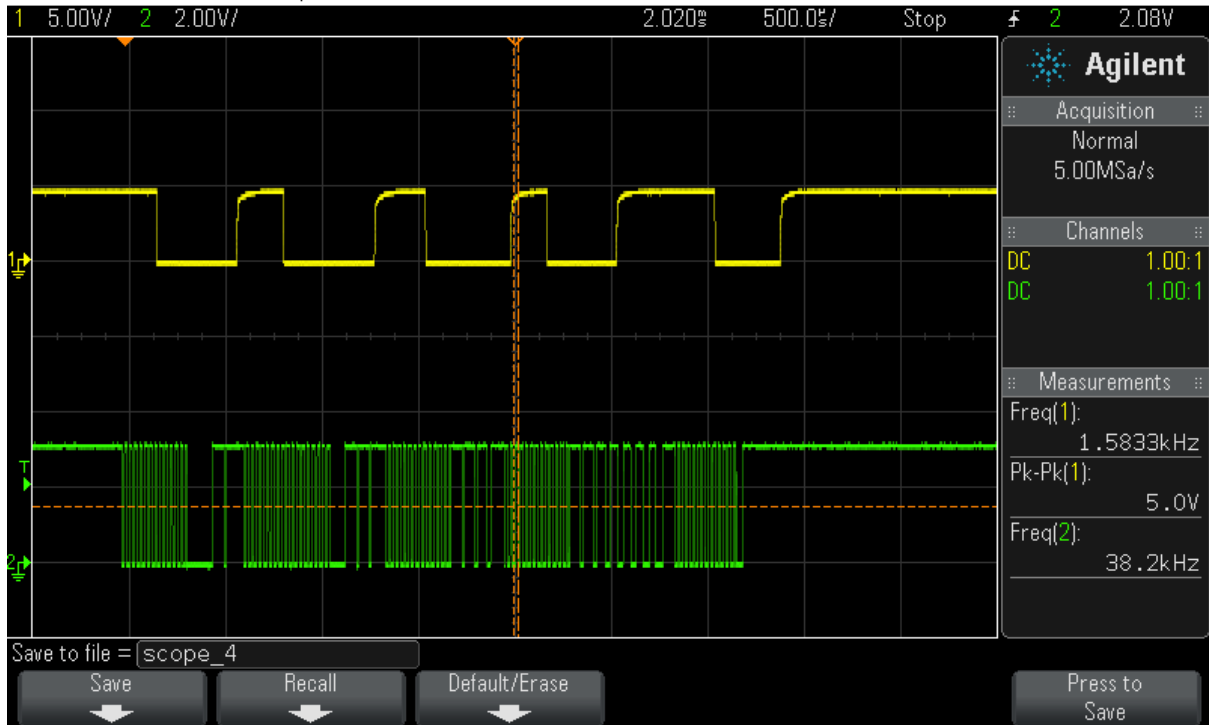
**Green line shows transmitting end binary string(This is a pulse modulated signal) Note\*This was scale doesn't show the actual frequency of the transmitting pulse this is observed in second test**

**The binary string sent was 1010101**

From this it shows that the receiver end interprets the sent binary string in an inverted form. Thus a string of **1010101** is seen as **0101010**.

This due to the receiver's default state being constantly high thus when it views and equivalent 1 it goes down to active low.





Yellow line shows receiver end binary string

Green line shows transmitting end binary string(This is a pulse modulated signal) Note\* The pulse of the transmitting signal was accurately measured to see if indeed it was transmitting at the set freq of 38kHz.

The binary string sent was 10101001

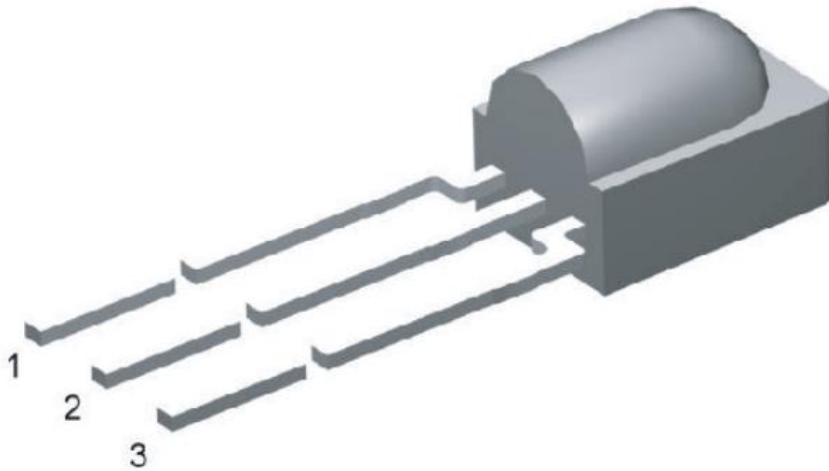
## 6.3 Receiver Circuit Subsystem

### 6.3.1 Technical requirements for Receiver Circuit

#### Technical Requirements for Receiving Circuit:

- **TS1:** Receive IR signal from IR LED and should correspond with what was sent
  - **Test\_1:** Attach output of IR Receiver to oscilloscope to verify received signal correspond to transmitted signal
- **TS2:** Send received signal to Receiving RPi through GPIO pin
  - **Test\_2:** Check RPi is using a GPIO pin to read in data from IR receiver
- **TS3:** Be able to comfortably receive IR signal at transmission distances of 2m or more
  - **Test\_3:** Do Bit-error-rate (BER) tests at different distances

### 6.3.2 Calculations, circuit diagrams and simulations



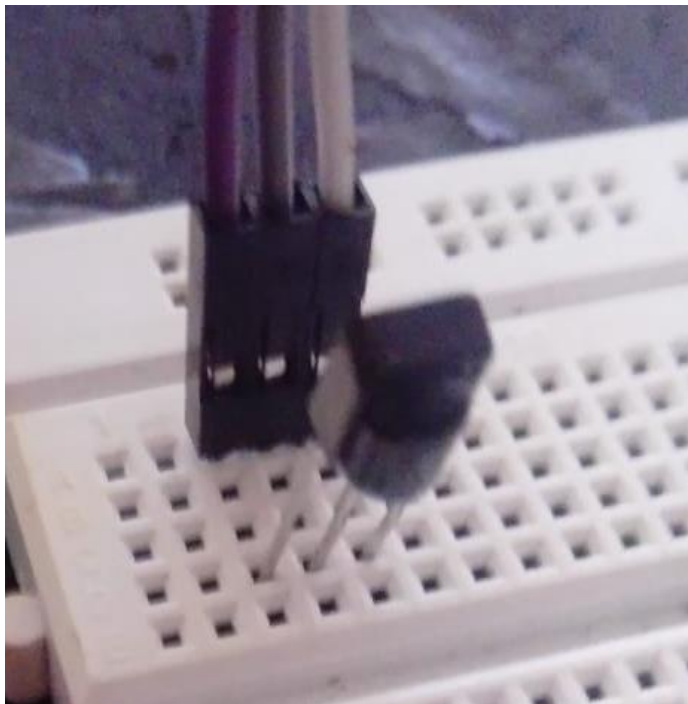
**1 = GPIO PIN (Output of TSOP38238), 2 = GND, 3 = 5V**

The TSOP38238 IR Receiver is directly connected to the RPi. Using the physical numbering of the 40 pins on the RPi, 1 will be connected to pin 11, 2 will be connected to pin 9 and 3 will be connected to pin 2.

No calculations needed and the TSOP38238 component does not do any decoding of the signal, it just passes the 'raw data' along.

### 6.3.3 Build and Testing

#### Physical Diagram



#### Tests Performed:

The 2 tests performed for receiving circuit were as follows:

1. Detect a response output from Rx circuit from an infrared input with carrier frequency of 38 kHz
2. Determine whether the received signal correspond to the signal sent by Tx circuit

To check for test number 1, a DSTV remote sending IR signals was used. The “XMP1.4” protocol is transmitted at a carrier frequency of 38 kHz. Receiving python code was used to detect whether the IR receiver detected anything. Then for Test 2, this was in collaboration with the Tx circuit. The outputs from both circuits was attached to an oscilloscope to see if the received signal correspond in any way to the signal sent by Tx circuit.

### Results:

#### Test 1:

```
0 8982
1 4399

0 631
1 497

0 628
1 1607

0 635
1 490

0 629
1 509

0 627
1 500

0 633
1 500

0 625
1 500

0 624
1 1623

0 629
1 1605

0 622
1 501
```

#### Python receiving program detecting IR signal from DSTV remote to test IR receiver component TSOP38238

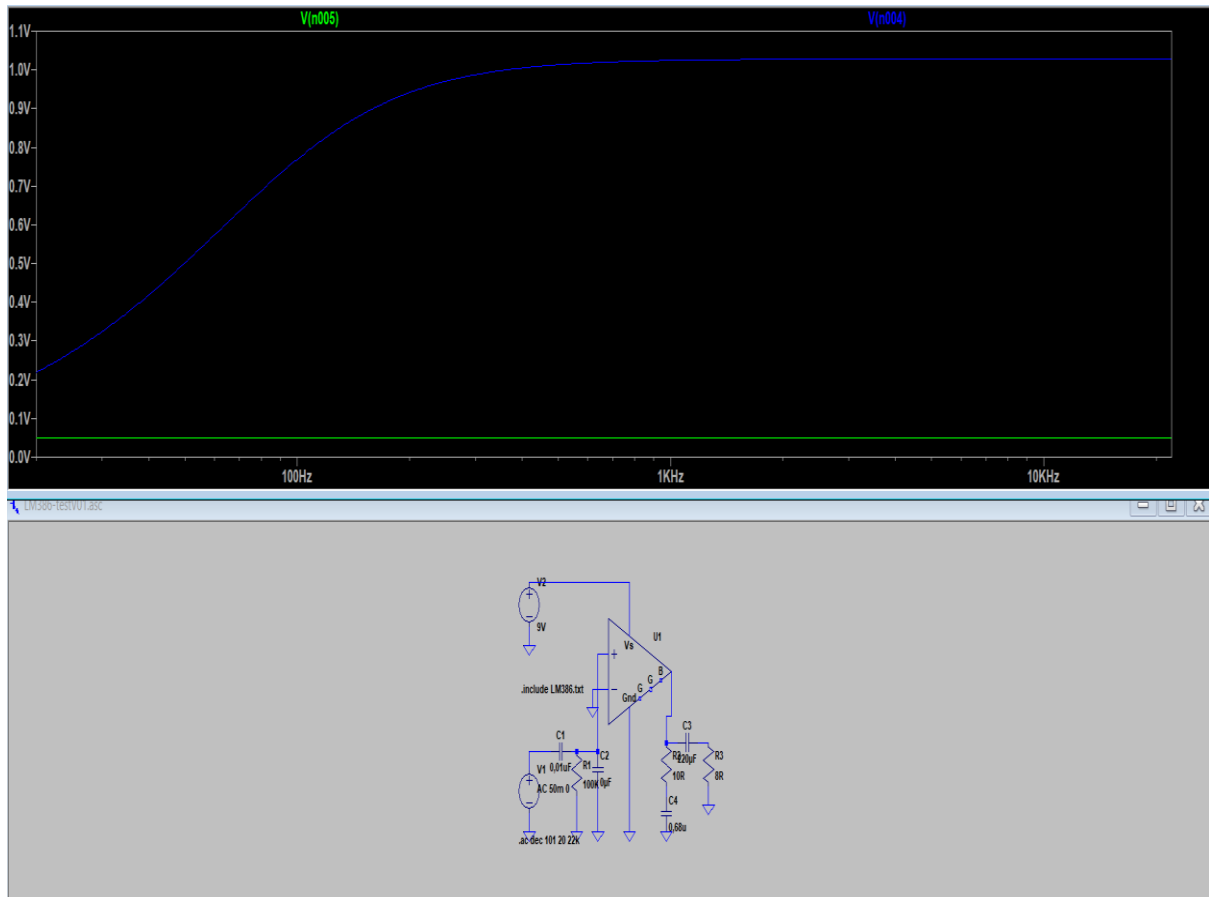
The python test code shows the binary values detected by the IR receiver and the duration that the component sees the signal for in nanoseconds.

#### Test 2:

Test 2 results were done in collaboration with the Tx circuit therefore the oscilloscope results were recorded under the transmitter circuit section.

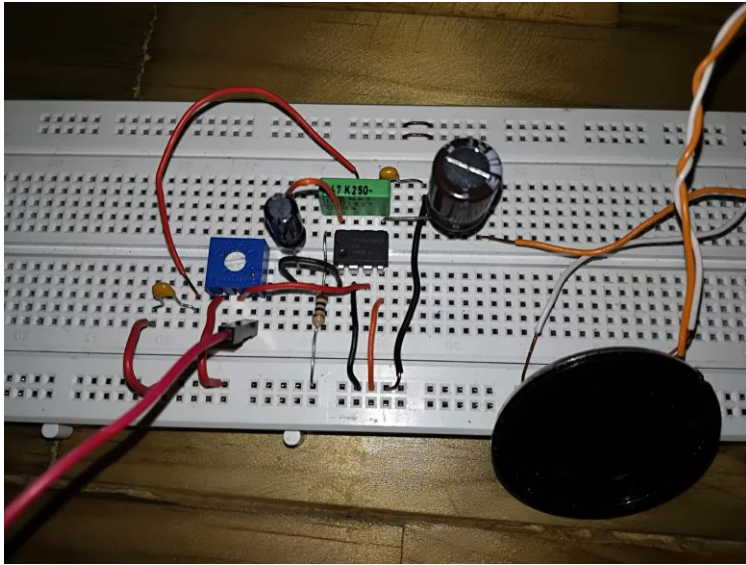
## 6.4 Audio amplifier & Speaker

### Audio Amplifier

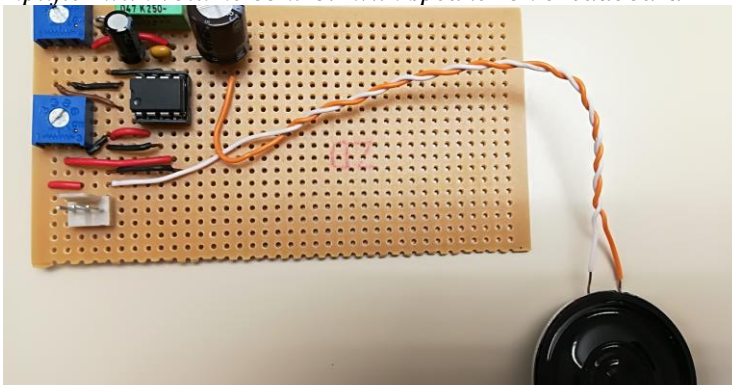


frequency sweep simulation of the lm386 amplifier circuit in LTSpice

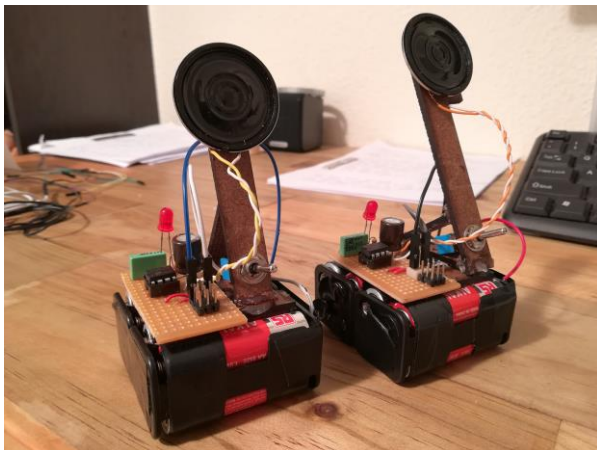
Gain: gain is set by adjusting the impedance of the pin 1 and 8. Open pins (infinite impedance) sets the gain at 20dB



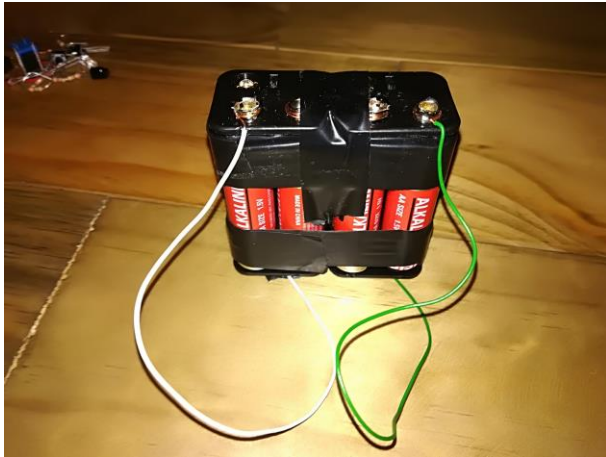
*Audio amplifier with volume control with speaker on breadboard*



*Audio amplifier with volume and gain control and molex connector soldered onto veroboard*

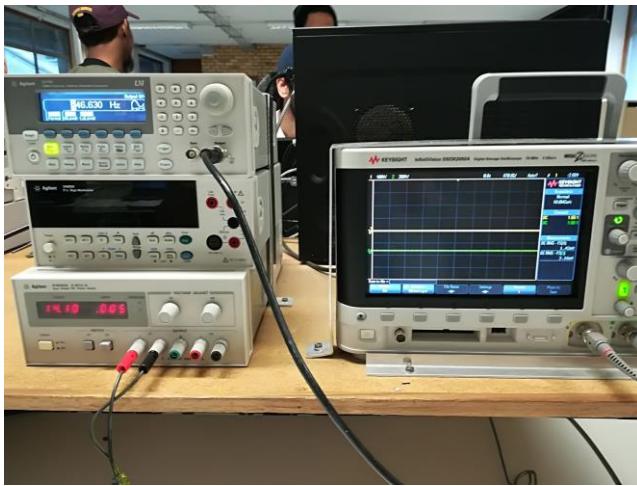


*Final audio amplifiers with speaker*



*Battery pack at approx. 12V*

### **Testing equipment:**



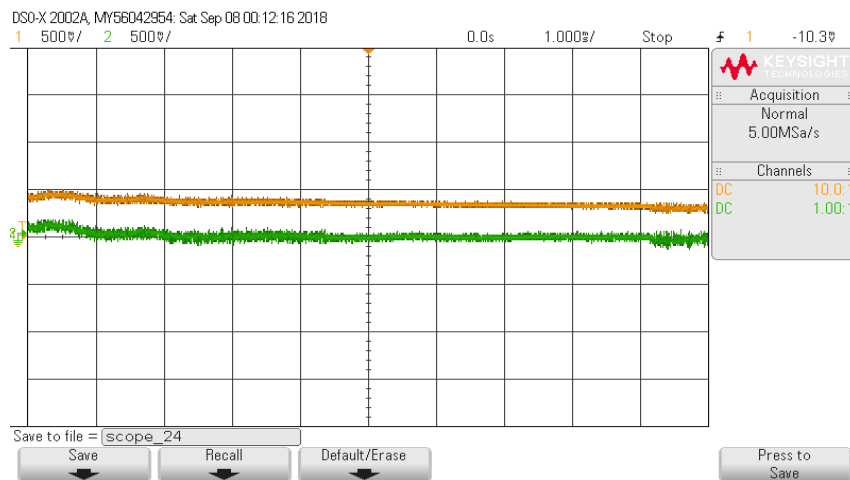
*Signal generator, dc power supply and the oscilloscope*

### **Testing conditions:**

- input signal was generated by the function generator to experiment with not only frequency but also the effect of signal  $V_{pp}$  on the output waveform
- input signal frequency was experimented for frequencies (Hz): 200, 300, 400, 500, 600, 700. Frequencies higher was not experimented since a clear pattern emerged and it could be extrapolated
- The amplifier was set at a gain of 20. This is achieved by opening pin 1 and 8.
- The output and the input of the amplifier was connected to the oscilloscope for side-by-side analysis
- The amplifier was powered by the DC power supply and set at 12V since the battery pack produces 12V
- **green line:** output signal (probed at speaker terminals)
- **orange line:** input signal (probed at the signal generator clips)

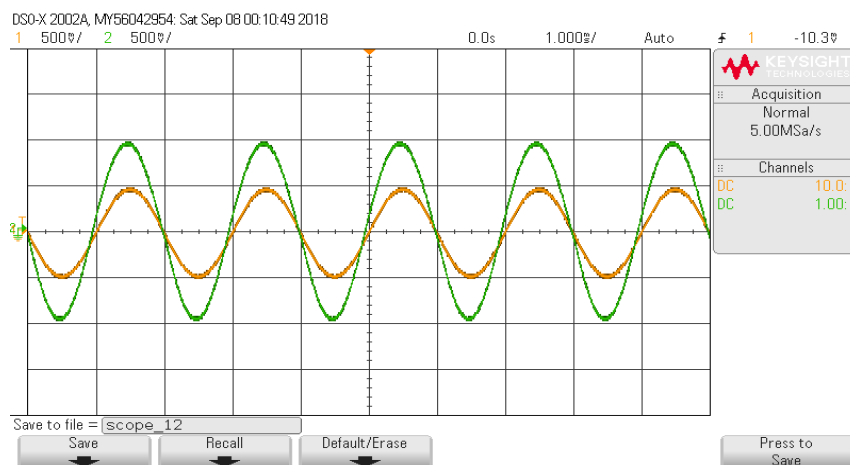
### **Results & Analysis:**

- noise when the amplifier is powered but has no input signal:



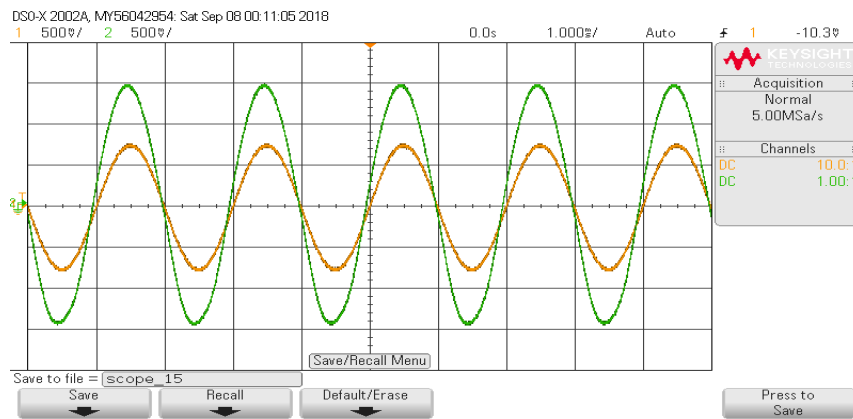
the noise is large enough that it is audible in a manner typically described as “static noise”, however, it is very minimal and only noticeable in a quiet environment at very close proximity to the speaker. Furthermore, it can be seen that the output waveform closely mirrors the input noise. This suggests that there is no significant additional noise that occurs within the circuit

### - 500Hz; 50mVpp



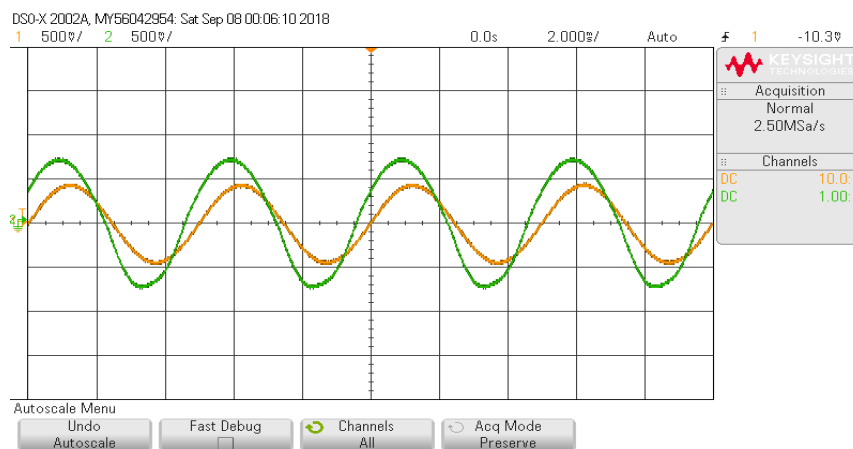
output waveform (green) looks undistorted and the smooth lines indicate minimal noise

### - at higher input voltage: 500Hz; 80mVpp



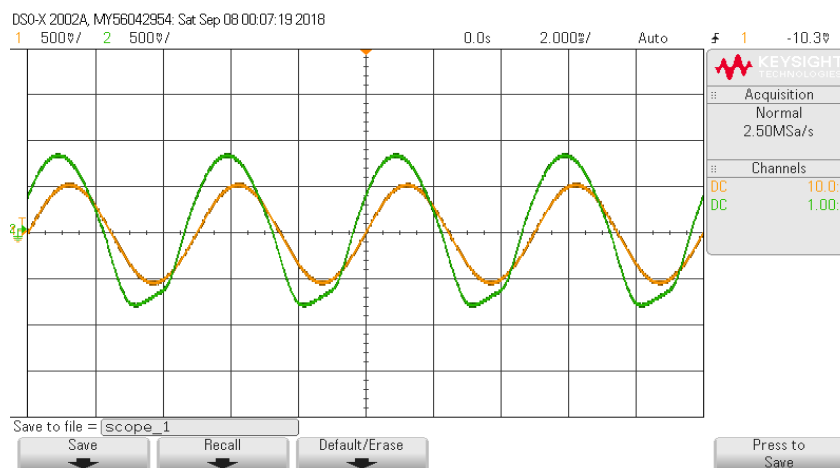
even at high input voltage, the output waveform is undistorted

- at lower frequency: **200 Hz; 50mVpp**



output shows slight distortion at the convex portion of the wave. The bottom peaks are slanted slightly to the left

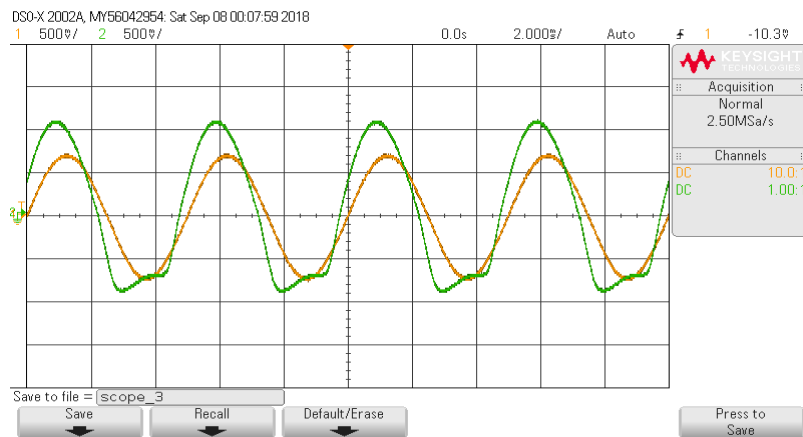
- **200 Hz; 60mVpp**



at a slightly higher input signal voltage, the distortion is more pronounced and it starts to be clipped

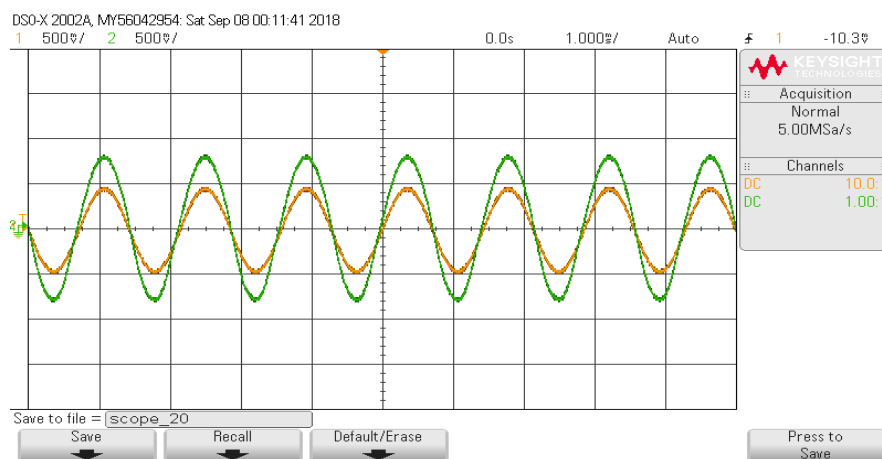
- at high input voltage: **200Hz; 80mVpp**





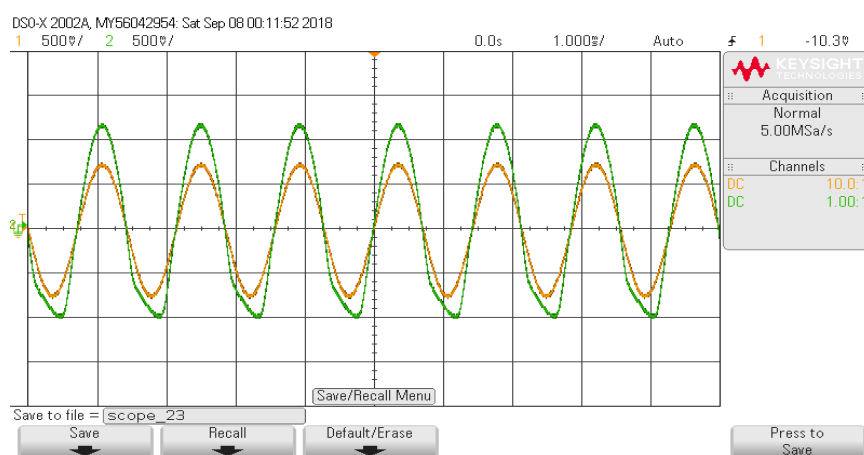
significant distortion of the output waveform at higher input signal voltage

- at high frequency: **700Hz; 50mVpp**



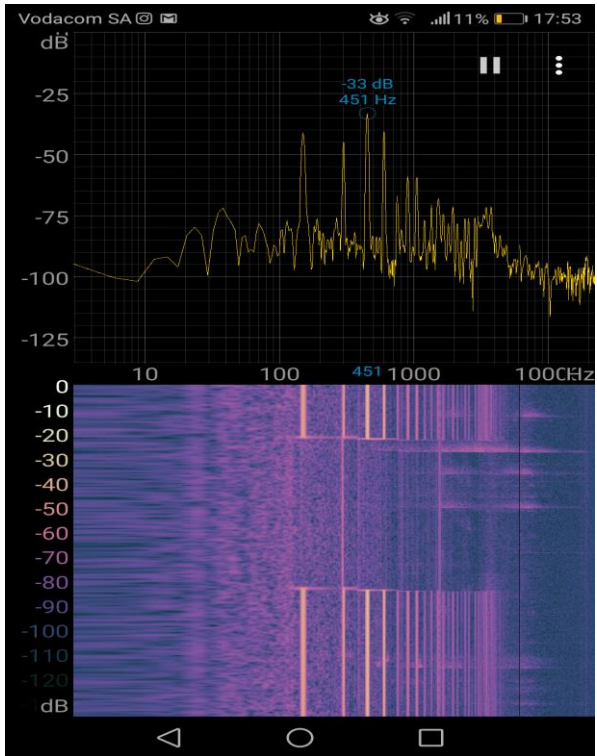
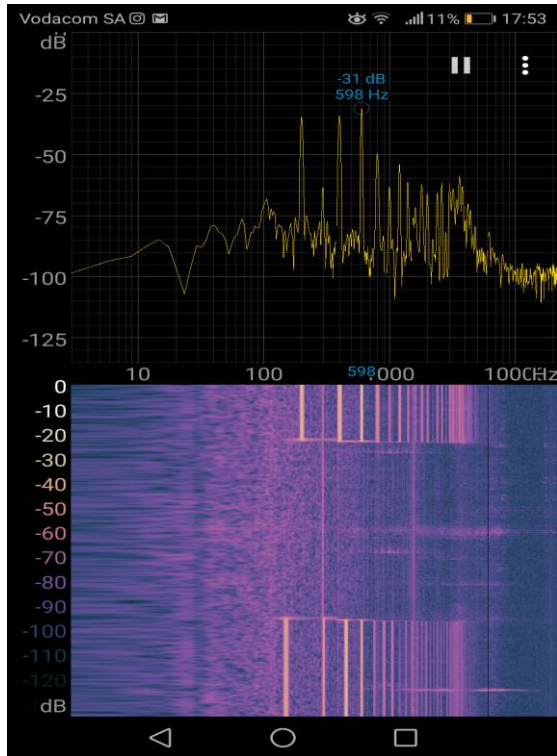
the output waveform is slightly distorted at the convex protino of the wave but it is minimal

- at higher input voltage: **700Hz; 80mVpp**



output waveform is distorted but, in contrast to 200Hz input signal, the wave is slanted to the right (as opposed to left). The distortion is not as significant as the 200Hz input signal

**Audio spectrum analyser:**



The audio amplifier was connected directly to the function generator that created a pure sine wave at 200Hz and 150Hz respectively.

What is interesting to note is that the speaker is generating frequencies at not only the input frequency but also at  $2^n \times \text{input\_frequency}$  ( $n = 1, 2, 3, \dots$ )

The frequencies at  $2^n \times$  input frequency are in musical terms, the same note but at higher octaves.

### **Summary of result:**

The most stable input signal frequency seems to be roughly around 500Hz. It is at this frequency that the increases in input signal  $V_{pp}$  does not distort the output waveform as it does in other frequencies. The further the frequencies are from 500Hz the more significant the distortion seems to become. Higher frequencies suffer distortion but not as much as the lower frequencies.

From the results, it would make sense to choose the octave scale such that it stays around 500Hz. The C5 note is 523.3Hz and ends at B5 of 987.8Hz. Since the C4 is 261Hz and ends at B4 of roughly 450Hz, and higher frequencies suffer less distortion than lower frequencies, the 5th scale will be used. Furthermore, the input  $V_{pp}$  should be kept as close to 50mV $_{pp}$  as possible without making the speaker inaudible.

The speaker generates the correct note but at various octaves. Since this is not an error on the hardware part, this will be accepted as a correct implementation and this property won't declare the subsystem faulty in later tests.

## 8. Logical integration of sub-systems and testing

The system was integrated by creating a single driving program that would be used to run the bluetooth connector, transmitting program, receiving program and the audio generation programs.

To achieve this multi-threading programming was used by creating concurrent threads that would run the programs independently while sharing specific files only if it was necessary for the two threads to interact.

### 8.1. App and Tx RPi

To test this connection the terminal was used and a shared Read file was created as a way to interact between the transmitting code and the application.

To connect the Tx RPi and the application a python script was developed that would use the bluetooth interface to send information to the RPi from the application. The application would then send instructions to the RPi via a READ file that was changed to match the instructions conveyed by the Application.

The instructions were in the following format : [ Boolean(Has a message been sent), Boolean(Is it melody or tune mode), String(single note or line of notes), String(duration of note or notes), String(volume of notes) ].

To test this a message is displayed on terminal when a specific instruction is sent from the App, the Read file is then checked to see if the instructions match the READ file contents.

This test was a success with messages being successfully sent in the agreed format with READ file being changed to match.

The transmitting program was adjusted to read this file at every millisecond intervals and run a tune transmission or a note transmission whenever The TRUE state appeared in the READ file. The code would then read the next BOOLEAN message if it was TRUE it should transmit the a single note specified by the following parameters in the instructions sent by the App. If the message BOOLEAN is FALSE then the transmission code sends the tune specified at the specified time interval and volume.

This test was also a success with with messages being accurately Read and the appropriate signal being sent.

Thus integration between App and Tx RPi was a success.

## 8.2. Tx RPi and Rx RPi

The integration between the two Pis was done by developing an asynchronous communication protocol that uses Infrared transmitter and receiver and counting for the physical drawbacks of these physical components.

The first drawback that was addressed was the timing of each binary signal during to not sharing a synchronised clock the time for each signal had to be agreed upon.

The biggest issue being that the time for each signal is not consistent thus the time had to be large enough that error between the timing would be enough to cover the boundary conditions.

Thus the digital signal used was decided to be 8 times the time for a normal high which a single high was registered at 400 nanoseconds. This produced the most accurate results and accurately performed the BER test every time. Thus when testing if message was accurately translated for signal note mode by reading the message sent via a scripting code that interpreted the sent bytes the code was able to successfully and accurately send and read the delivered message.

But this did not alleviate the errors during tune mode where a long binary message was being sent.

To address this a buffer was sent in-between the data to alleviate errors thus the following message structure was developed - Buffer(20 digital highs) - Data( 10 bit binary string for either note volume or duration starting and ending with a digital 0) - Buffer.

This fixed data being able to be differentiated from and read as separate bytes by the receiver. The next issue was slightly corruption during transmission at the beginning and end of each data bit string towards the end of the chain - this was fixed by an interrupter code that actively corrected the notes (due to them being part of an agreed table of known bytes) by matching the bit string to its closest known counterpart. This code performed extremely well able to correct every 9 out of 10 corrupted bit strings.

But to bring this to 100 percent error free transmission we added buffer away bit strings to lengthen the tune sent and this corrected the error.

Thus the communication protocol was developed through systematic trial and error until a robust protocol was developed. Thus providing a successful test when the final protocol was run through all notes and durations sent by the transmitter being accurately read and played.

## 8.3. Audio amplifier and Rx RPi

The integration of the Rx RPi and the audio amplifier was done through the following procedures:

### 1). frequency testing

the Rx RPi generated a series of notes and sent the signal to the audio amplifier. The speaker (output of the amplifier) was probed with the oscilloscope and the output signal as well as the FFT was measured and performed.

The purpose of the above procedure is to compare the apparent signal that the Rx RPi wants to play to the actual note that comes out of the speaker.

The test was successful: all the notes in a scale was generated by the RPi and was compared with the oscilloscope reading. All the frequency readings were accurate to the frequencies of the wanted note.

### 2). volume testing

The Rx RPi receives 10 different levels of volume from the Tx RPi and to test how the Rx RPi is relaying this information to the audio amplifier, the RX RPi generated a series of signals at the same

note at 10 different volume levels. Then the speaker was probed (output signal of amplifier) with the oscilloscope and the amplitude of the signals was measured.

The test was successful: 10 different signals (of same note and duration but different volume) manifested in creating an output that had different relative amplitudes. i.e volume of 0 resulted in no signal and volume of 10 resulted in the greatest amplitude.

### 3). duration testing

The duration for which the note should be played versus the actual duration of the sound was tested by producing a series of signals at the same volume and note at different durations. The duration for which the note should be played and the actual duration was timed using a stopwatch.

The test was successful: various signals with different durations was generated and the time specified resulted directly in the output signal lasting for the same duration. The result of this test was unsurprising since the audio generation code was known to work properly (from sub system tests) and duration is directly linked to how long the RPi executes the code for and not affected by hardware of the amplifier.

## 9. System Level Testing

List the test that were done on the system. For each 'system level test' explain the following

- What equipment and process was used in the test.
  - Compare the expected results from each test, to the actual results obtained.
  - State whether the test was successful or unsuccessful
  - If unsuccessful, explain reasons for this and what could be improved to obtain better results.
  - Based on the results of the test, confirm which system technical specification and user requirement can be confirmed to be met or not met
- 

### 9. 1. Testing

The final system test was done by examining the data that was being inputted to the first subsystem (App) and the data that was being outputted by the last subsystem (audio amplifier/speaker).

The following test was performed:

2 random values from each of the 3 data variables (note, duration, volume) was chosen and all possible permutation of the 3 variable values was generated and tested

i.e

values from note: A and D

values from volume: 2 and 6

values from duration: 2 and 10

inputted data values for testing are: A,2,2 ; A,2,10 ; A,6,2 etc.

It was necessary to test the permutation of the variable values because it tests how certain variable values interact with one another to possibly produce errors. In this regard, the above test is not complete since it only tests 2 values per variable but this was necessary otherwise, there are too many tests.

The output from the last subsystem (speaker) was analysed using the oscilloscope and a stopwatch and compared with the inputted variable values.

As expected from the success of the subsystem tests and the integration tests, the final system test was successful:

- 1). all the inputs matched the output correctly with in terms of the frequency, the relative amplitude and the duration when compared with the oscilloscope as well as the audio spectrum analyser.
- 2). the speaker was audible from 2m away from volume of greater than 4 units

## 9.2. Bill of Materials

Group 4: RS Components				
Part Number	Manufacturer Part Number	Price	Quantity	Goods value
7085086	TSOP38238	7.954	5	39.77
6997676	TSHG5410	4.651	6	27.906
4859556	TIP50	4.238	6	25.428
7542090	Mini speaker	32.55	2	65.1
5342911	LM386N-4/NOPB	16.87	2	33.74
1004081	Veroboard	117.58	1	117.58
9176588	LR6 AA 20s Batteries	5.483	20	109.66
594628	SN341 Battery Holder	9.99	4	39.96
			<b>Total</b>	459.144
			<b>Budget</b>	800

## 10. Summary of contribution by each team member

### Tasks completed for milestone 3 by each group member

#### Bill:

- Introduction & Revised Project Plan
- Revised System Block Diagram & Description of Rx RPi and Rx Circuit

- Receiver Circuit Subsystem
- Embedded Software Receiving Code & Activity Diagram & Testing it
- BOM & Soft skills questions
- Appendix A: Minutes of team meetings
- Video Editing

**Tapuwa:**

- Description of Tx RPi and Tx Circuit
- Interface 2
- Transmitter Circuit Subsystem
- Embedded Software Transmitting Code & Sequence Diagram
- Logical Integration and Testing

**Terri:**

- audio synthesis using python code
- building and testing of audio amplifier
- Interface 3
- Logical Integration and Testing

**David:**

- Project Plan
- Interface 1
- Mobile App Subsystem
- Appendix A: Minutes of team meetings
- UML Diagrams

Summary of each team members contribution to the work done upto milestone 1

	<b>David</b>	<b>Tapuwa</b>	<b>Bill</b>	<b>Terri</b>
<b>David contribution</b>	25%	25%	25%	25%
<b>Tapuwa contribution</b>	25%	25%	25%	25%
<b>Bill contribution</b>	25%	25%	25%	25%
<b>Terri contribution</b>	25%	25%	25%	25%

## References

tutorialspoint.com. 2018. *Digital Communication Frequency Shift Keying*. [ONLINE] Available at: [https://www.tutorialspoint.com/digital\\_communication/digital\\_communication\\_frequency\\_shift\\_keying.htm](https://www.tutorialspoint.com/digital_communication/digital_communication_frequency_shift_keying.htm). [Accessed 10 August 2018].

Advantages of VLC | Disadvantages of VLC communication. 2018. [ONLINE] Available at: <http://www.rfwireless-world.com/Terminology/Advantages-and-Disadvantages-of-VLC-Visible-Light-Communication.html>. [Accessed 10 August 2018].



Advantages of RF | Disadvantages of RF | Radio Frequency. 2018. [ONLINE] Available at: <http://www.rfwireless-world.com/Terminology/Advantages-and-Disadvantages-of-RF.html>. [Accessed 10 August 2018].

Electronics Hub. 2018. *IR transmitter and receiver circuits*. [ONLINE] Available at: <https://www.electronicshub.org/ir-transmitter-receiver-circuits/>. [Accessed 10 August 2018].

za.rs-online.com. 2018. *IR Receivers*. [ONLINE] <https://za.rs-online.com/web/p/ir-receivers/7085086/>. [Accessed 10 August 2018].

IR Transmitter and Receiver Circuit Diagram. 2018. *IR Transmitter and Receiver Circuit Diagram*. [ONLINE] Available at: <https://circuitdigest.com/electronic-circuits/ir-transmitter-and-receiver-circuit>. [Accessed 10 August 2018].

blog.bschwind.com. 2018. *Sending Infrared Commands From a Raspberry Pi Without LIRC*. [ONLINE] Available at: <https://blog.bschwind.com/2016/05/29/sending-infrared-commands-from-a-raspberry-pi-without-lirc/>. [Accessed 09 September 2018].

raspberrypi-blutetooth-demo[ONLINE] Available at <https://github.com/EnableTech/raspberry-blutetooth-demo> [Accessed 03 September 2018]

Bluetooth Tutorial - Sending/Receiving Data with Bluetooth [ONLINE] Available at <https://www.youtube.com/watch?v=sifzY2SA1XU&t=600s> [Accessed 28 August 2018]

github.com. 2018. *EnableTech/raspberry-blutetooth-demo*. Available at: <https://github.com/EnableTech/raspberry-blutetooth-demo> [Accessed 05 September 2018].

upenn. 2018. *Audio Data*. Available at: [https://www.ling.upenn.edu/courses/Spring\\_2003/ling538/Lecnotes/AudioData.html#mp3](https://www.ling.upenn.edu/courses/Spring_2003/ling538/Lecnotes/AudioData.html#mp3) [Accessed 01 June 2018]

android developer. 2018. *Bluetooth Overview*. Available at: <https://developer.android.com/guide/topics/connectivity/bluetooth> [Accessed 13 August 2018]

## Appendix A: Minutes of team meetings

### Milestone 1:

#### Meeting 1:

Date: 25 July 2018

Time: 15:00

Duration: 20 minutes

Location: Menzies 3<sup>rd</sup> floor (glasshouse)

Team Members Present: Heng Rui (Bill) Guo, Tapuwa Dhliwayo, David Fransch

Team Members Absent: Terri Lee (late registration, not part of group yet)

Discussion:

- Align understanding of the project between team members
- Divide project into sub-systems and assign group members to be responsible for specific subsystems (Bill – Receiver; Taps – Transmitter; David – Mobile App, Amplifier and Speaker)
- Set dates for certain tasks to be done:
- All report sections (8 August)
- Final formatting + final report for milestone 1 (10 August)
- Decided on project roles: Bill – Team Secretary (Note Taker); Taps – Project Leader; David Project Manager
- Set meeting times for every week (Tuesday and Thursday @ 10am)
- LATEX (Overleaf)

Tasks (Action Items):

- Do research on sub-systems, each member in charge of research for own sub-system
- Clarify with TA on how tune must flow from beginning (Mobile App) to end (Speaker) – Bill responsible to ask and report back
- Set up WhatsApp group and Google Drive – Bill

#### Meeting 2:

Date: 31 July 2018

Time: 10:00

Duration: 25 minutes

Location: Menzies 13

Team Members Present: Heng Rui (Bill) Guo, Tapuwa Dhliwayo, David Fransch, Terri Lee

Discussion:

- Met Terri and finalised decision on which option to increase the workload of our project (chose option 3)

- Reassessed and aligned understanding of our project.
- Identified report sections for milestone 1 and set date for User Requirements, Technical Specifications for each subsystem to be done (3/4 August)
- Use-case sequence diagrams will be completed by Taps.
- Assigned Terri to help Taps with research on subsystem (Transmitter).

Tasks (Action Items):

- Complete User Requirements, Technical Specifications – All
- Do further research about own sub-system – All

### Meeting 3:

Date: 02 Aug 2018

Time: 10:00

Duration: 25 minutes

Location: RW James Foyer

Team Members Present: Heng Rui (Bill) Guo, Tapuwa Dhliwayo, David Fransch, Terri Lee

Discussion:

- Progress check-up on how everyone is doing with their research and report for their section.
- Different modulation and demodulation techniques (frequency, amplitude and phase) were discussed. Probably going to implement frequency modulation.
- Favouring Bluetooth communication between phone and RPI.

Tasks (Action Items):

- Find out how music notes are converted to signals. - All
- Can infrared LEDs communicate through glass? – Taps

### Meeting 4:

Date: 07 Aug 2018

Time: 10:00

Duration: 30 minutes

Location: RW James Foyer

Team Members Present: Heng Rui (Bill) Guo, Tapuwa Dhliwayo, David Fransch

Team Members Absent: Terri Lee

Discussion:

- Receiver circuit approaches: IR receiver components (TSOP38238), IR transceiver components and the given figure

- Displayed how to you can create wav files on the site  
<https://www.wavtones.com/functiongenerator.php>
- Reiterate deadlines for report sections 2018/08/09 17:00.
- IR communication can go through glass but not below a certain frequency

Tasks (Action Items):

- Email Jason about wav files and IR transceiver components – Bill
- Complete and compile file report. – All

## **Milestone 2:**

### **Meeting 5:**

Date: 14 Aug 2018

Time: 10:00

Duration: 20 minutes

Location: RW James Foyer

Team Members Present: Heng Rui (Bill) Guo, Tapuwa Dhliwayo, David Fransch

Team Members Absent: Terri Lee

Discussion:

- Told David what he missed on 13 August lecture, basically ran through report template for milestone 2.
- Talked about issues and conflicts within the team.
- Need to establish safety net to make sure we meet deliverable deadlines.
- Talked about inputs and outputs for each subsystem and who must work with who to make sure subsystems are integrable.

Tasks (Action Items):

- Test out receiver raspberry pi to make sure it can receive and detect a square wave. - Bill
- Send email to Jason to mark specific report as 3 submissions was given – Bill
- List your responsibilities here – other group members if you can

### **Meeting 6:**

Date: 16 Aug 2018

Time: 10:40

Duration: 15 minutes

Location: RW James Foyer

Team Members Present: Heng Rui (Bill) Guo, Tapuwa Dhliwayo, David Fransch, Terri Lee

Discussion:

- Talk about using Sketch and Invision to show android app interface prototype.
- Due dates got extended by a week.
- Split into groups of two: Receiver end (Bill and Terri), Mobile app + Transmitter (David and Taps)

Tasks (Action Items):

- Receiver end: Bill decide how to go about it
- Transmitter end: Taps decide how to go about it
- What and how sections all must be done by 23 August

### Meeting 7:

Date: 21 Aug 2018

Time: 10:00

Duration: 30 minutes

Location: RW James Foyer

Team Members Present: Heng Rui (Bill) Guo, Tapuwa Dhliwayo, David Fransch, Terri Lee

Discussion:

- Talk about using Sketch and Invision to show android app interface prototype.
- Due dates got extended by a week.
- Split into groups of two: Receiver end (Bill and Terri), Mobile app + Transmitter (David and Taps)

Tasks (Action Items):

- Receiver end: Bill decide how to go about it
- Transmitter end: Taps decide how to go about it
- What and how sections all must be done by 23 August

### Meeting 8:

Date: 23 Aug 2018

Time: 10:00

Team Members Present: David Fransch, Terry Lee

Team Members Absent: Heng Rui(Bill) Guo, Tapuwa Dhilwayo

Discussion:

- Terri updated David on previous meeting- where modulation and demodulation techniques were discussed.
- David updated Terri on progress of App
- Discussed meeting structure and milestones.

Tasks (Action Items):

- Clarify modulation/demodulation with Jason
- Make meetings more structured to get things done
- Have shorter meeting for simple update sessions.

### Meeting 9:

Date: 28 Aug 2018

Time: 10:00

Team Members Present: David Fransch, Terry Lee, Heng Rui(Bill) Guo

Team Members Absent: Tapuwa Dhilwayo

Discussion:

- Receiver working with DSTV remote
- Issues with transmitter
- Set deadlines ahead of milestone 2 deadline

Tasks (Action Items)

- Have sub-systems working by Sat (01-09-2018)
- Report Tue (04-09-2018)
- Meeting on 05-09-2018 to finalise report & subsystems

- Stretch goal: Connect subsystems (07-09-2018)

### **Milestone 3:**

Meeting 10:

**Date:** 20-09-18

**Time:** 10:10am

**Duration:** 35min

**Location:** RW James

Team Members Present: David, Bill and Tapuwa

Team Members Absent: Terri

### **Feedback on Milestone 2:**

- Better meeting attendance.
- Improve communication.
- Have more milestones.
- Understand one another's strengths and weaknesses.
- Make one another more accountable.

### **Discussion**

Decided on having a half duplex demo where all subsystems could be tested on **27-10-2018**. This will be followed by a review and feedback of the demo. Using this we can build onto progress and set a new milestone.

### **Tasks (Action Items)**

#### **Milestone A:**

1. App:
  - a. Update GUI
    - i. Melody mode update
  - b. Send note parameters
2. Transmitter:
  - a. Take in note parameters
  - b. Send to TX
    - i. Not yet robust - add padding/buffer
3. Receiver:
  - a. Change to C code. - done
  - b. Make more robust (add tolerances) - same as above
  - c. Audio generation code (Bill and Terri) - link Rx to Audio side (possibly change to C code)
4. Speaker & Amp:
  - a. Put on veroboard - done

- i. Build another
- b. Fix any bugs - done
- c. Audio generation code (Bill and Terry)

TEST: Send a note through system and play it on speaker  
SUCCESS YES/NO

**[DUE: 27-10-2018]**

#### Meeting 11:

**Date:** 27-09-18

**Time:** 10:10am

**Duration:** 35min

**Location:** RW James

Team Members Present: David, Bill, Terri and Tapuwa

#### Discussion:

This meeting was held to integrate the subsystems in order to test the half duplex implementation (Milestone A). Set dates and objectives for next milestone B. General

#### **Tasks (Action Items):**

(Results of Milestone A and tasks TODO)

#### **App:**

##### Completed

GUI for note sending

Sending note parameters

##### TODO

Implement GUI for melody

Test sending data by verifying on Tx side

#### **Tx:**

##### Completed

Receiving note identifier

Sending to Rx

##### TODO

Implement for parameters duration and volume

Make more robust – add padding/buffer

Error checking

#### **Rx:**

### Completed

Change to C code

Receive from Tx

### TODO

Make more robust – add padding/buffer

Connect to audio side

### **Speaker & Amp:**

#### Completed

Put on Veroboard

Fixed bugs

### TODO

Build second circuit (full duplex)

Audio generation code – should work with Rx RPi

### Meeting 12:

**Date:** 8-10-18

**Time:** 2pm

**Duration:** 2 hours

**Location:** RW James

Team Members Present: David, Bill, Terri and Tapuwa

### Discussion:

- Tested individual subsystems
- Tested RPi Rx with speaker & amp
- Fixed problems with App
- Planned integration with App and RPi Tx

### Tasks (Action Items):

- Integrate RPi and App
- Plan testing for full duplex
- Fix and neaten up code

### Meeting 13:

**Date:** 09-10-18

**Time:** 2pm

**Duration:** 3 hours

**Location:** RW James

Team Members Present: David, Bill, Terri and Tapuwa



**Discussion:**

- Reviewed progress of last meeting
- Found new issue with App (cross-compatibility)
- Designate sections of report
- Integrate App and RPi Tx (not tested)
- Melody mode

**Tasks (Action Items):**

- Fix strange bug in App
- Test App and RPi
- Complete issues with melody mode

**Meeting 14:**

**Date:** 11-10-18

**Time:** 2pm

**Duration:** 4 hours

**Location:** RW James

Team Members Present: David, Bill, Terri and Tapuwa

**Discussion:**

- Solve outstanding issues from yesterday
- Any uncertainties that members may have
- Fixed and tested App with RPi Tx

**Tasks (Action Items):**

- Test full system
- Verify that melody mode works across subsystems

**Meeting 15:**

**Date:** 11-10-18

**Time:** 2pm

**Duration:** 2 hours

**Location:** RW James

Team Members Present: David, Bill, Terri and Tapuwa

**Discussion:**

- Test full system
- Finalise full duplex
- Preparation for demo

**Tasks (Action Items):**

- Demo
- Report
- Video