BIR Group G: Human-Robot Team Communication Using Audible Pitch Detection

Iwan Cavil, Peter Alexander, Bruno Colato, John Hawk, Mario Pulze

1 Introduction

This project has produced two robots, and a base station. The robots are able to accurately and reliably detect different frequencies and modify their behaviour accordingly. It was inspired by the Vervet Monkeys as a novel method of communication, with the aim of enabling cohesive teams of humans and robots, so audible tones from C3 to A4 were used.

2 Project Inspiration

2.1 Vervet Monkeys

This project was inspired by Vervet monkey alarm calls. This call is used by the monkeys to share information about specific threats in the area. The calls are very specific, which allows other monkeys to differentiate and react appropriately [1].

2.2 Music Theory

As stated in [1], it takes infant Vervet monkeys many months to learn Starling alarm calls. To minimise the training required for humans to learn a similar system, music theory was used to guide the design of the robot calls. Musicians and non-musicians alike are able to identify emotions such as "happy" and "sad" from major and minor musical chords of two or three notes[2][3]. With no training, an operator can interpret different calls, ignoring "happy" and "pleasant" calls whilst instinctively understanding "sad" or "unpleasant" calls as the result of an issue. A human operator may also be able to evaluate the status of the robot swarm by the number of "happy" and "sad" calls, similar to the emotion of a crowd[4]. These properties can replace or augment the equipment needed to convey this information digitally.

3 Project Overview

The goal of this project was to create mobile robots that communicate with each other and a base station using audible frequencies. Two working robots were created that could emit and detect different tones and modify their behaviour based on this. The base station emits a C4 note, which the robots' emitted tones overlay to create dissonant or harmonic tones, so humans can instantly hear if there is a problem.

An application of this project would be in automated warehouses. The robots would be automated item movers. When an issue is detected, they will emit a tone that warns other movers of the issue. The human workers would then be able to hear this. Once resolved, they would reset the area and the robots would return to work. Humans can also turn off the base station, stopping the robots from working. A block diagram of the project overview is shown in Figure 1, with final product images illustrated in Appendix A, Figures (4-6). The project specification is detailed in Appendix G.

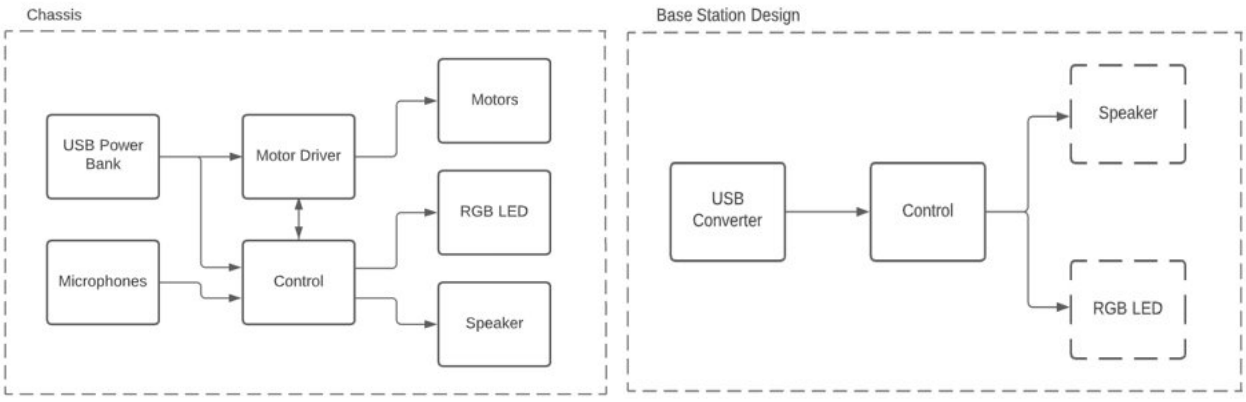

Figure 1: Block Diagram Overview of Project

4 Development

4.1 Motors

In this project, the motor selection was given particular attention as it is the most important factor in determining the battery life of the robots. There were several requirements; low current draw, cheap, small, ease to use, speed control, and reliability. Due to these factors, a simple hobbyist motor was selected with an inbuilt gearbox. These were chosen as they were very easy to use, being controlled with a PWM input, and the axle shaft was mounted at 90° to the body of the motor, meaning that it wouldn't limit the space across the width of the robots during assembly.

4.2 Speakers

4.2.1 Robot Speakers

The purpose of the speaker on the robots is to communicate it's current state to other robots and humans nearby. This is achieved by emitting a range of tones (C3-A4) depending on its state.

Due to the robots being battery powered, small 0.25W speakers with 5V PAM8403 audio amplifiers were used to provide long battery life and sufficient volume. To improve the audio quality, digital-to-analogue converters (DAC) controlled by the serial peripheral interface (SPI) communication protocol, were chosen. A low-pass filter was added to remove the high frequencies created by the DAC. The circuit diagram is available in Appendix B, Figure 8.

4.2.2 Base Station Speakers

When on, the base station emits a continuous C4 tone to bring the robots out of standby mode. More importantly, the base station tone provides a reference note for humans, which, combined with the robot tones, produce a pleasant or unpleasant sound.

The base station is powered by USB C, therefore a 5W speaker and a 12V LM384N audio amplifier were used to provide the tone at a higher volume for greater communication distance. Similarly, an SPI DAC was used to produce the tone and a low-pass filter used to remove high frequencies. The circuit diagram is available in Appendix B, Figure 9.

4.3 Chassis

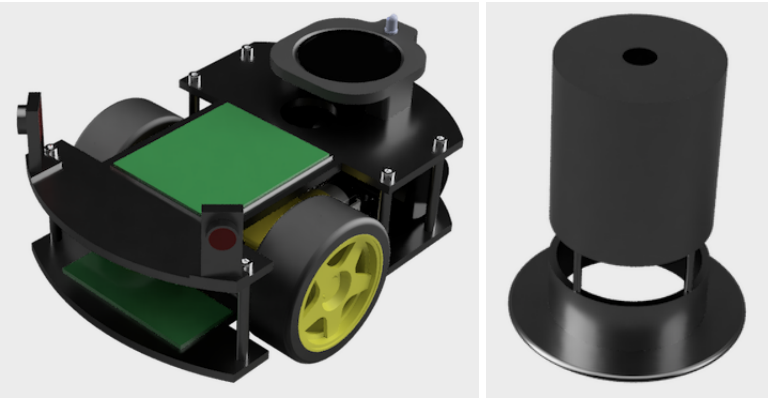

Figure 2: CAD Rendered Models of the robot (left) and the base station (right)

4.3.1 Robot Chassis

The robot chassis was designed using Autodesk Fusion 360 and printed on a 3D printer. It's design needed to accommodate all of the other components and therefore would be used extensively. With that in mind the design needed to allow easy access to all components. To minimise interference the microphones needed to be positioned away from the motors and speaker. To achieve this the chassis incorporates a two level design, separated by standoffs, illustrated in Figure 2.

Components located on the lower level are; motors, wheels, motor driver protoboard, power bank and its mount. A caster wheel is fitted on the underside to provide three point floor contact. The motors are mounted with 3M screws on raised platforms to ensure a level chassis. Space was allocated to mount the motor driver protoboard, and a bespoke clip was created for the power bank.

The upper level has a 20mm opening for cables and was initially designed as a flat surface. This would allow models to be redesigned and printed once parts had been ordered and changes made. For example, the speaker and microphone mounts were attached with hot glue onto the chassis. This is illustrated in Appenidx C, Figure 11.

4.3.2 Base Station

Similarly, the base station was designed to be accessible, consisting of three interlocking parts. The lower part includes a wide base for stability, and openings for the sound to be emitted. The middle section has interlocking feet and a mount to hold the speaker. This has a large space to accommodate the base station electronics. Lastly, the final section, which includes an RGB LED and power lead cutout, interlocks into the middle section using an inner grove, as shown in Figure 2. This is also shown in Appendix C, Figure 12.

4.4 Microphones

To facilitate rapid development, pre-made microphone amplifier boards were used. Initially boards based on the Max9814 amplifier were purchased, however these were later replaced with Max4466 boards due to their lower cost. Testing was carried out in an attempt to block out sounds from behind using 3D-printed audio shields, however these tests were unsuccessful. Instead, the microphones on the amplifier boards were replaced with unidirectional condenser microphones with a cardioid pickup pattern, see Appendix D, Figure 14.

To reduce the requirements for data collection from the microphones, an LTC1062/LTC1069 low pass filter was used to remove the high frequencies from the incoming audio. The cut-off frequency was set at 500Hz as the highest frequency note produced is an A4 (440Hz), to provide a margin for error.

4.5 Control

4.5.1 Frequency Detection

Each robot must be able to parse incoming sound samples to detect the presence of certain frequencies in real time. The Discrete Fourier Transform (DFT) family of algorithms was chosen as it is a well-documented method of detecting frequency information in embedded environments. Both the Goertzel Filter[5] and Fast Fourier Transform (FFT) were investigated. Both rely on grouping frequency data into N bins, each collecting the energy of a small part of the range of frequencies from 0 Hz to the sampling frequency. More bins results in higher precision, but an increased computational load, as each bin covers a smaller frequency range.

By restricting maximum detected frequency to 500 Hz, the sampling frequency can be restricted to 1 kHz, much less than the 44.1 kHz commonly used to sample the full range of human hearing. The computing load was reduced further by choosing a lower bound for frequency detection as well. Lower-pitches are closer together than higher pitches (D1 - C1 = 4.01 Hz, D3 - C3 = 32.03 Hz). The frequency range of C3 to A4 provided a selection of notes to work with whilst also limiting the precision required to 128 bins. Details on the pitch selection can be found in Appendix E.1. These choices meant that real-time frequency detection could occur entirely within software on low-cost, low-power microcontrollers.

4.5.2 Application Code

It was decided that the code manipulating the robot's actuators would be run on the same processor as frequency detection. This central core had to balance hard real-time tasks such as audio sampling with motor control and sound generation. For this reason, the central core required a multi-tasking environment.

One set of tasks is responsible for frequency detection, and the other uses the frequency data to make decisions about system state and actuate the on-board motors, RGB LED, and speaker. For this demonstration, the application code was kept simple. A state machine was implemented which used processed frequency data as inputs. The state machine allows the robots to exhibit complex behaviour in response to frequency data without large computational requirements. Details on the robot behaviour can be found in Appendix E.2.

5 Implementation & Testing

5.1 Motor Driver Board

The drivers chosen to control the motors were the LB1930MC. As these were surface mount components they were soldered to breakout boards allowing a protoboard to be designed, shown in Appendix B, Figure 7. Female pin headers were utilised for the input terminals and JST connectors were used as the outputs for the motors.

One board was needed for each robot. Testing was conducted using an Arduino UNO with code written on the Arduino IDE. Various speeds were tested successfully to provide variation in movement whilst the robot is in different operating states. After the boards were operating reliably they were mounted to the robot.

5.2 Speakers

Testing found that both the robot and base station speakers produced the required tones at sufficient volume and quality. However, the robot speaker did not function well at low frequencies (< 260Hz, C4), but this could be avoided by using slightly higher frequencies.

5.3 Microphones

Preliminary testing for the microphones was undertaken using a simple embedded program to read the voltage levels and plot it on a display. Both the omni-directional, and the replacement unidirectional microphones were tested. After the functionality of these amplifiers and microphones was verified, they were connected to the FFT and filters. This showed the required frequencies could be detected on the incoming signal when using a known frequency from a tone generator. This test is described in Appendix D.

5.4 Control

Initial investigations into robot frequency detection focused on identifying a single frequency bin using a Goertzel filter on an ATMega328P. The team had experience working with AVR devices both bare-metal and using FreeRTOS[6]. This showed early success in analysing a set of test samples generated in MATLAB within a FreeRTOS task. However, adding multiple Goertzel filters reached the memory limitations of this chip so other solutions were explored.

MicroPython was used for this project as it is more powerful and flexible than AVR/FreeRTOS. An ESP32 development board was selected for the hardware as it is commonly available and supports a port of MicroPython. Due to the increased performance of this platform, it was decided that a full FFT should be implemented. A major benefit of using MicroPython over AVR/FreeRTOS is the wide range of public, open-source user libraries that have been written to augment it. The micropython-ulab library[7] implements a subset of the CPython libraries NumPy[8] and SciPy[9], including SciPy's `spectrogram` function. This serves as the core of the frequency detection system. Details on the Micropython build for this project can be found in Appendix E.3.

A downside to MicroPython is its cooperative multitasking system. This, combined with the ESP32 implementation of MicroPython makes audio sampling directly on the ESP challenging. An ATTiny1604 running Arduino was connected to the ESP32 over I2C to handle the sampling. Initially, this device responded to a read request from the ESP by sending over two samples, one for the left mic, the other for the right. A MicroPython software timer was used to regulate the sampling frequency, but this proved inaccurate when the ESP was under heavy load. This motivated the creation of a real-time border between the ESP and the ATTiny. The ATTiny handles all audio sampling using hardware timers. Samples are buffered locally before being sent over I2C in 32-byte packets at the ESP's request, relieving it of all hard real-time duties.

The ESP requests a new block of samples 20 times a second. This rate was chosen to limit the number of samples the ATTiny has to store at any given time. These samples are placed in to circular buffers to await processing by the FFT tasks. Sample retrieval, FFTs, and application code all execute on asynchronous tasks and are coordinated by a system of events and locks. FFT outputs are placed on a global Python dictionary where each bin can be accessed by referencing the note name (e.g. `output_dict['C3']`) for ease of use and upgrade potential. A signal flow diagram for the finished frequency detection and control system can be found in Figure 3. A schematic for the frequency detection and control circuits can be found in Figure 10. The application code performs a running average on the FFT data to further remove noise. The FFT data is converted to state-machine inputs at each new FFT result as follows: when the amplitude of a particular note is above a certain threshold, a counter is incremented to an upper limit; if the note is not present, the counter is decremented to a lower limit; if the counter is above a set value it is considered to be 'On' otherwise it is considered to be 'Off'. These inputs are then fed into a traditional state machine, more details can be found in Appendix E.2.

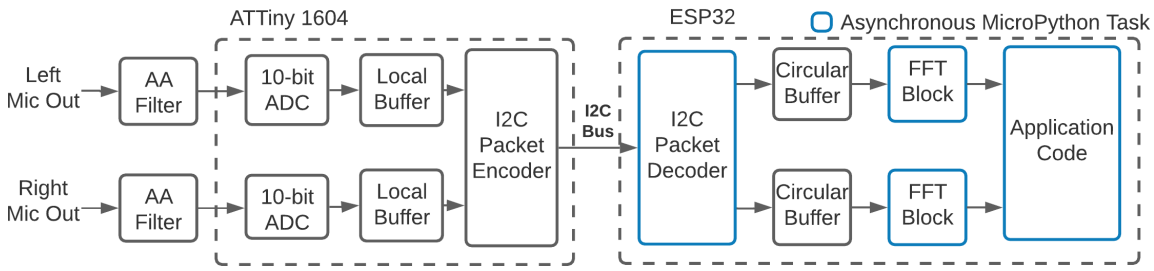

Figure 3: Control Block Diagram

6 Budget Management and Risk Assessment

The budget for this project was £200, of which we spent £200.25 including VAT and shipping costs. A full list of the components ordered and suppliers used is shown in Appendix F.1, Table 2.

This project involved working on a number of different aspects of electronics which has risks, particularly during a global pandemic. An exhaustive risk assessment is illustrated in Appendix F.2, Table 3.

7 Future Works

From our specification, future work on this project would be to implement a more complex control scheme to take full advantage of the directionality of the microphones. This would allow for more advanced behaviour to be exhibited by the robots. Then, more sensors would be added to the robots to detect nearby dangers. After this has been done, adapting the wheels to make them able to cope with rougher terrain would be a simple next task. Additionally, the ESP32 should be replaced with an STM32 based microcontroller, as this supports micropython natively and would remove the requirement for the ATTiny I2C slave devices.

8 Conclusion

This project has accomplished nearly all of the specification goals, including the stretch goals that were set. The robots are able to put themselves into different states and modify their behaviour based on the sounds that they hear. They have a long battery life and are able to move around freely. This project has developed many new approaches in hardware and software which are applicable to further research in this project and others.

References

[1] M. D. Hauser, "How infant vervet monkeys learn to recognize starling alarm calls: the role of experience," *Behaviour*, vol. 105, no. 3-4, pp. 187 – 201, 1988. [Online]. Available: https://brill.com/view/journals/beh/105/3-4/article-p187_1.xml

[2] K. J. Pallesen, E. Brattico, C. Bailey, A. Korvenoja, J. Koivisto, A. Gjedde, and S. Carlson, "Emotion processing of major, minor, and dissonant chords: a functional magnetic resonance imaging study," *Annals of the New York Academy of Sciences*, vol. 1060, no. 1, pp. 450–453, 2005.

[3] R. G. Crowder, "Perception of the major/minor distinction: Iii. hedonic, musical, and affective discriminations," *Bulletin of the Psychonomic Society*, vol. 23, no. 4, pp. 314–316, 1985.

[4] V. Franzoni, G. Biondi, and A. Milani, "Crowd emotional sounds: spectrogram-based analysis using convolutional neural network." in *SAT@ SMC*, 2019, pp. 32–36.

[5] G. Goertzel, "An algorithm for the evaluation of finite trigonometric series," *American Mathematical Monthly*, vol. 65, p. 34, 1958.

[6] F. Guan, L. Peng, L. Perneel, and M. Timmerman, "Open source freertos as a case study in real-time operating system evolution," *Journal of Systems and Software*, vol. 118, pp. 19–35, 2016.

[7] Z. Vörös, "micropython-ulab module, release - 2.6.0," GitHub Repo, 2021. [Online]. Available: https://github.com/v923z/micropython-ulab

[8] C. R. Harris and K. J. Millman, "Array programming with NumPy," *Nature*, vol. 585, p. 357–362, 2020.

[9] P. Virtanen, R. Gommers, and T. E. Oliphant, "SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python," *Nature Methods*, vol. 17, pp. 261–272, 2020.

A Final Product

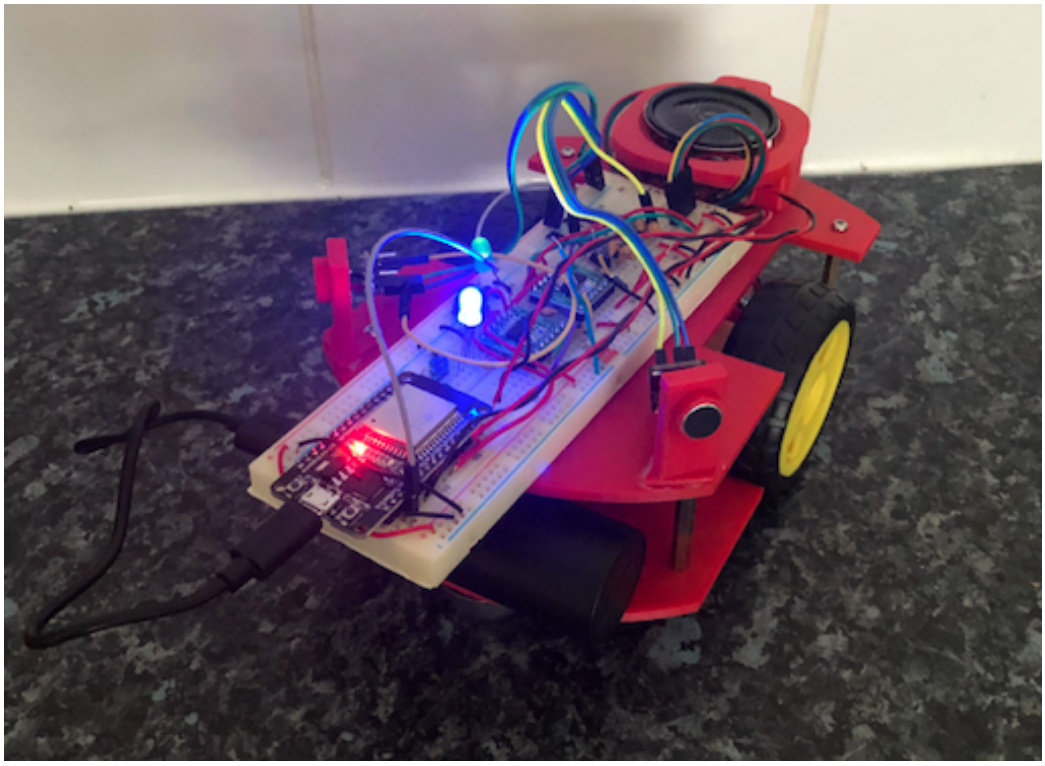

Figure 4: Knuckles

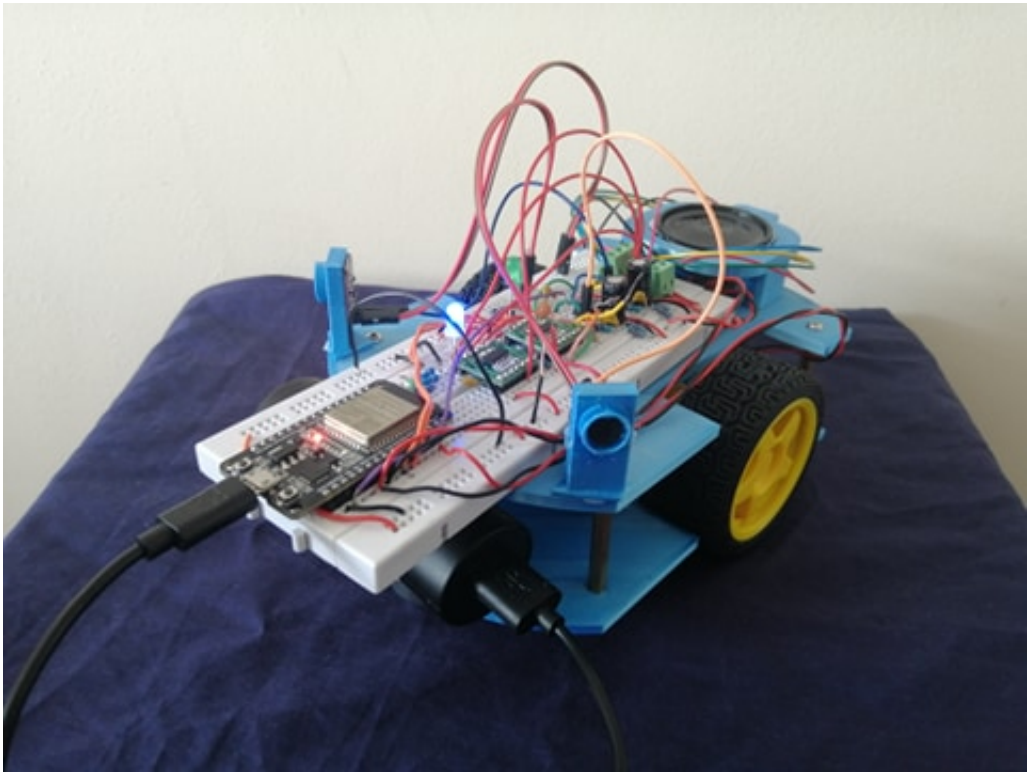

Figure 5: Sonic

Figure 6: Tails

B Circuit Diagrams

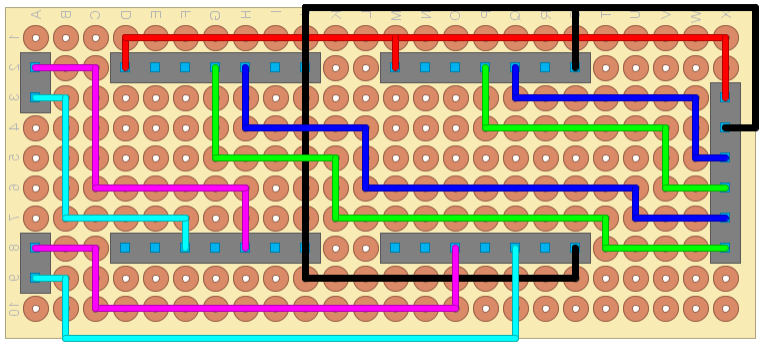

Figure 7: Motor Driver Board

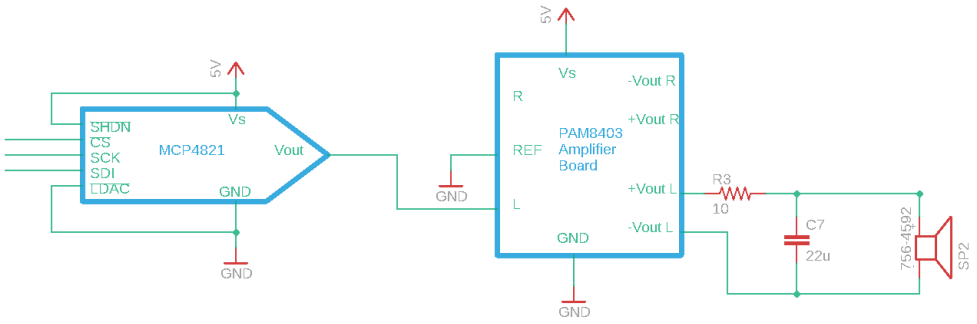

Figure 8: Robot speaker circuit diagram

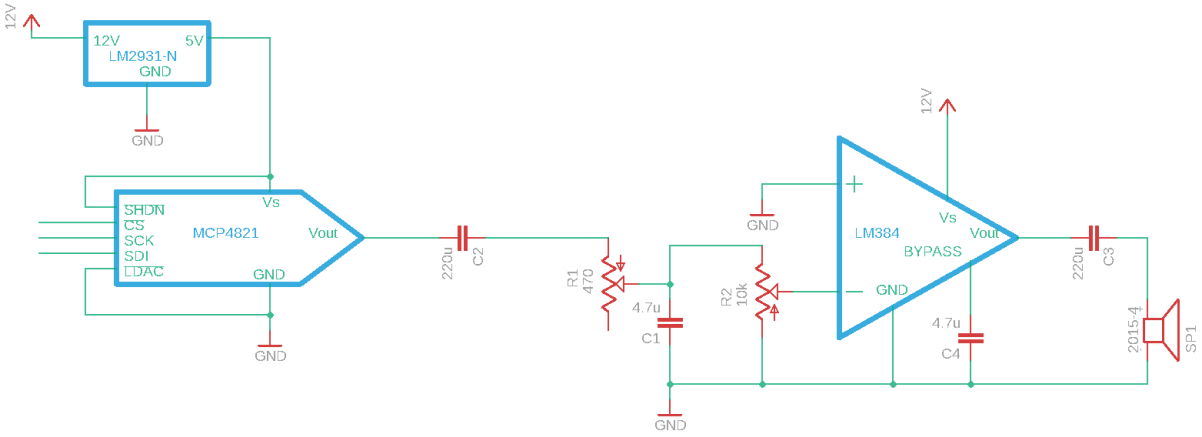

Figure 9: Base station speaker circuit diagram

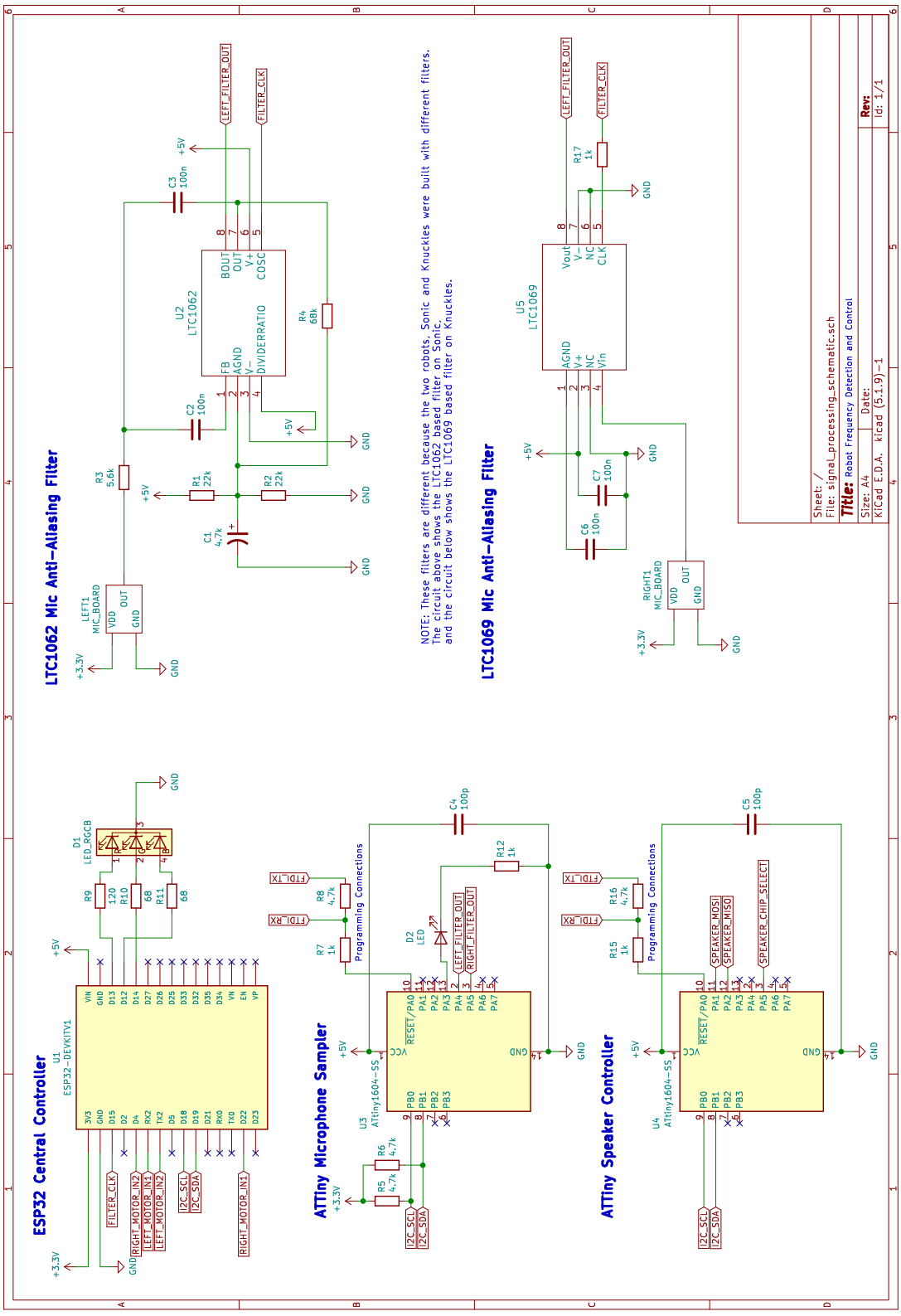

Figure 10: The schematic for robot the control and frequency detection circuits

C Chassis

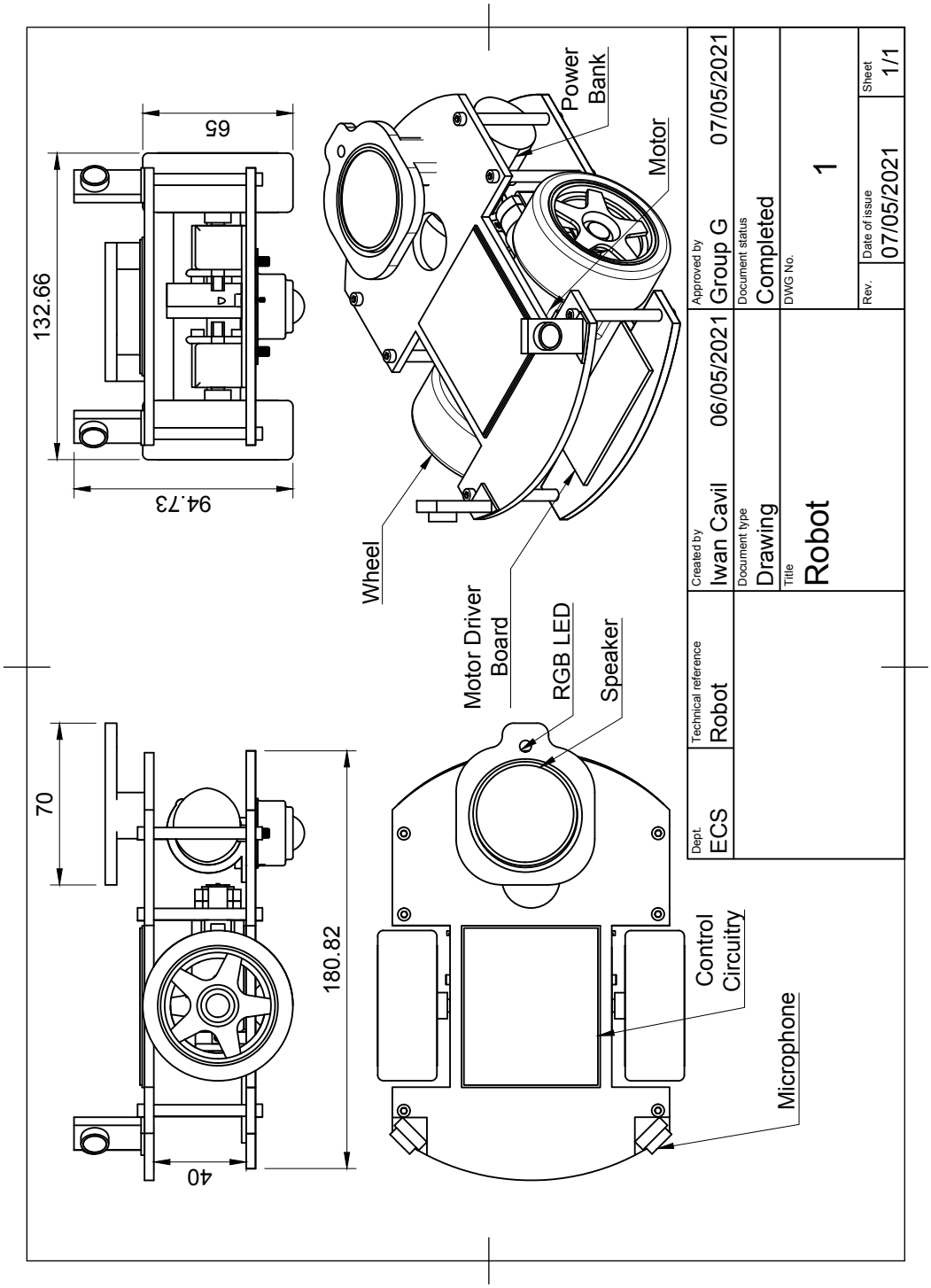

Figure 11: Engineering Drawing of the Robot Chassis

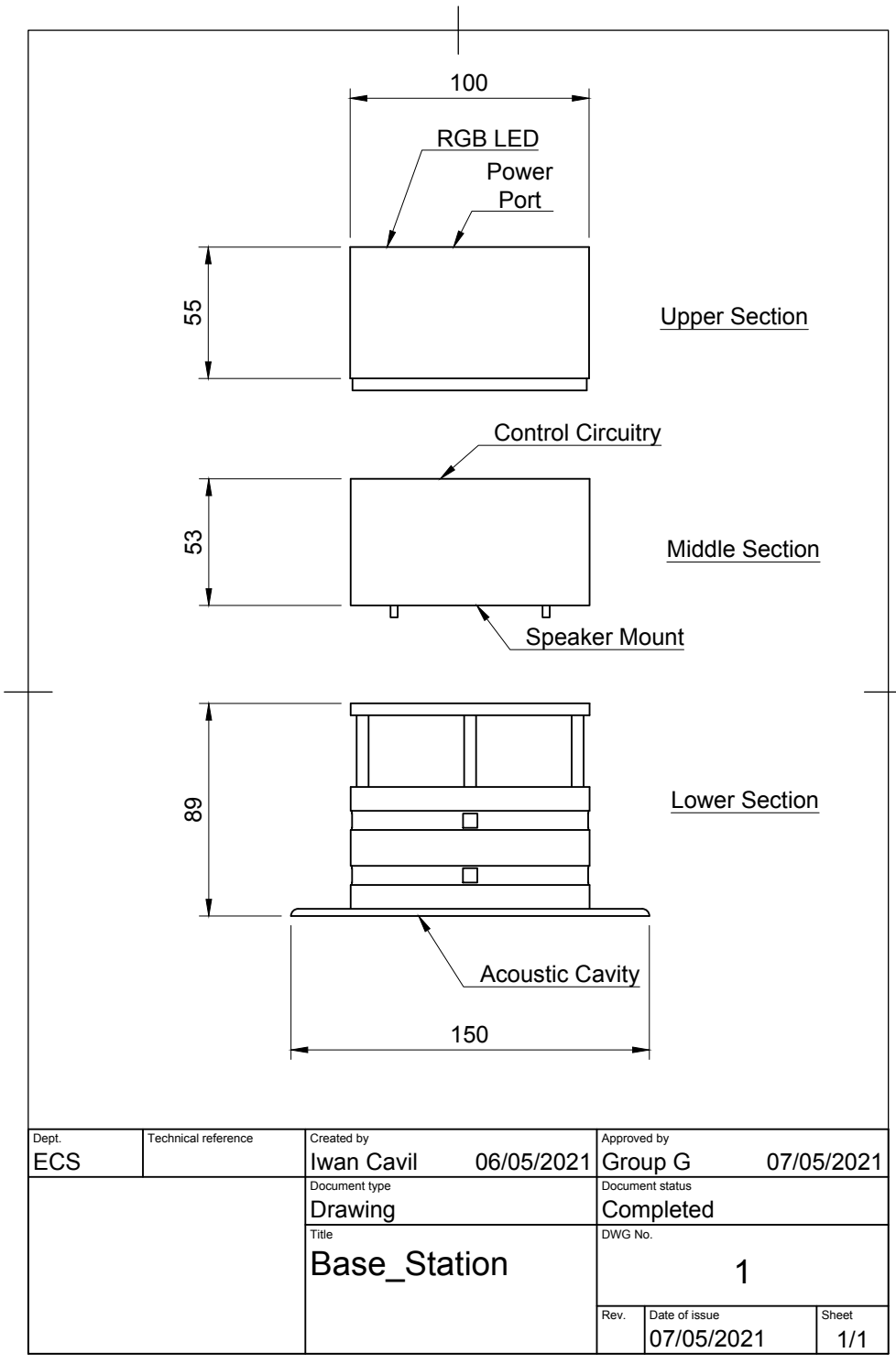

Figure 12: Engineering Drawing of the Base Station

D Microphone

Figure 13 shows a plot of of the MicroPython `spectrogram` output read over serial connection from the ESP32. It was produced to demonstrate the correct identification of pitches using the microphones working and the hardware discussed in Section 5.4. To produce this plot, the notes C3, E3, Fsh3 and A3 were sung in succession by the tester with gaps left between each note. It can be clearly seen that the FFT registers these notes, along with their respective, higher pitched counterparts C4, E4. Fsh4, and A4. This demonstrates the inclusion of extra hidden frequencies, or overtones, in what a human would perceive as a single tone (discussed further in Appendix E.1. It is thought that the higher pitched notes register more strongly because the cheap, electret microphones used in this project are attenuating the lower frequencies. More expensive microphones with a broader frequency range could be used in the future to improve the pickup of these notes.

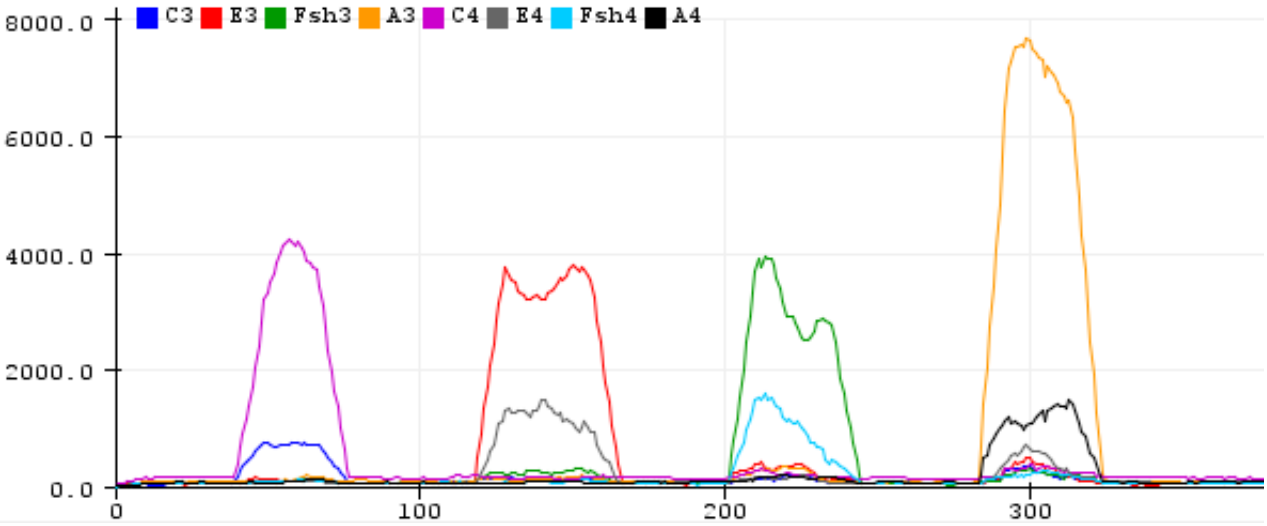

Figure 13: Micropython FFT outputs. y-axis is FFT amplitude, x-axis is samples (approx. 20 samples per second)

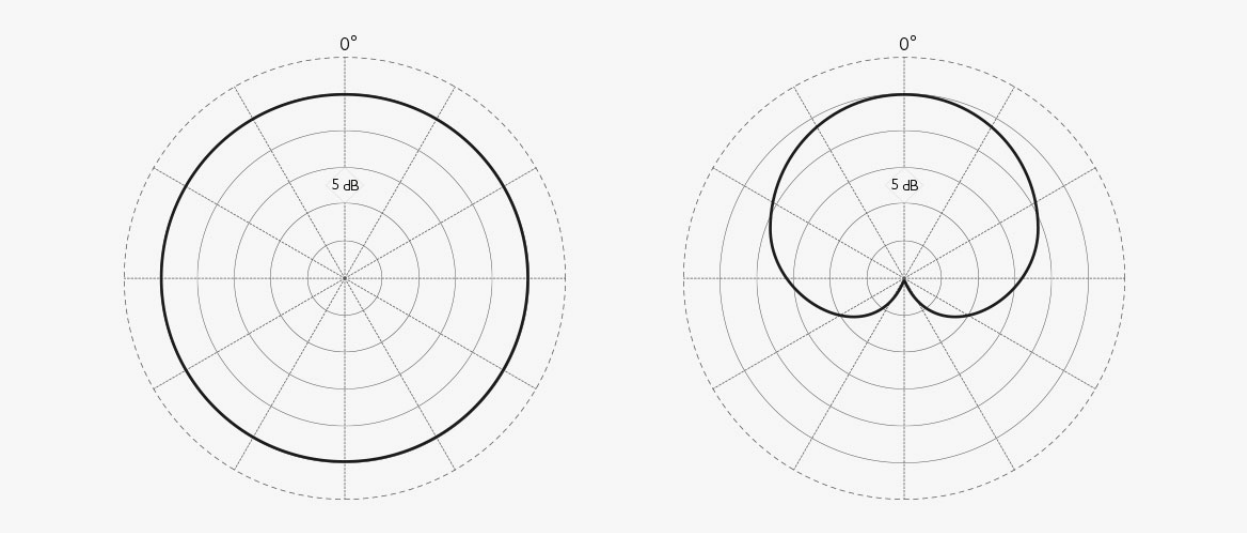

Figure 14: Omni-directional (left) vs Cardioid (right) microphone pickup patters

E Control

E.1 Pitch Selection

The pitches selected to convey information were chosen from the 12-tone equal temperament system with A4 = 440 Hz. The vast majority of Western music genres are built on this system and it is the system the team are most familiar with. Further investigation is needed to determine whether different pitches should be selected if the system were used by people with different musical upbringings.

Table 1 shows the the notes that the robot application code was set to interpret along with their basic meaning.

Note	**Frequency (Hz)**	**Message**
C3/C4	130.81/261.63	Base Station Note/Begin Working
E3/E4	164.81/329.63	All Clear
F#3/F#4	185.00/369.99	Danger Found
A3/A4	220.00/440.00	Caution Required

Table 1: A table showing the notes the robots can interpret and their basic meaning.

Note that each message is conveyed by two notes placed exactly an octave apart. Human pitch perception is more complex than frequency perception. What a human hears as a single tone may be made up of multiple frequencies, each of them detectable by an FFT. It is especially common to find frequencies of the same note in the octaves above (A3 transmitted with A4, A5, ...). Different sound generators have different overtone properties. The robot has been set to listen to notes from both detectable octaves to allow it to detect sounds from a variety of sources. In the future, robots could be tuned to respond only to the overtone 'fingerprint' generated by the Base Station and other robots.

Below is a justification for each note selection:

- **Base Station Note C3/C4** - This note is important as it will set the musical 'key' for the robot noises. This is a reference point against which all other pitches are compared. Most humans can only accurately detect relative pitch, i.e. the difference between two notes, so all other notes must be heard alongside this one for a human to interpret them.

  The notes C3/C4 were chosen as this puts the robot noises into the key of C major. This is the key "Twinkle twinkle little star..." and other Western nursery rhymes are often first hear in, so this note should feel familiar and "safe" to anyone brought up on such songs.

- **All Clear E3/E4** - When played against the Base Station C3/C4, this will form the C major chord, which represents a "happy" emotion when heard both on its own and as part of a song[2][3]. This was chosen to convey that a "good" thing has happened. This tone could also be used to convey other successes, such as a successfully completed task.

- **Danger Found F#3/F#4** - When played against the Base Station C3/C4, this sounds very discordant and "unpleasant". It was chosen to communicate that a bad thing has happened and to catch a human operators attention as something they need to focus on immediately.

- **Caution A3/A4** - This sound will be played when a known hazard enters the working area. As this is expected it may not require the immediate attention of an operator so a note that elicited an "anxious" or "sad" emotion against the Base Station note was chosen, rather than one that is discordant or "unpleasant" to hear. A3/A4 was chosen as it will form the chord of A minor, a"sad" sound alongside the C of the base station and the E heard regularly as the All Clear signal.

E.2 Demo Robot Behaviour

Figure 15 shows a state transition diagram describing the behaviour of the robot in response to the pitches heard. The behaviour of each state is as follows:

- **Standby** - The starting state of the robot. Do nothing. Slow flashing Blue LED.

- **Working** - The normal working state of the robot, move around randomly. Solid Green LED.

- **In Danger** - Stops in place, tries to identify the source of the triggering sound and moves away from it. Flashing Red LED.

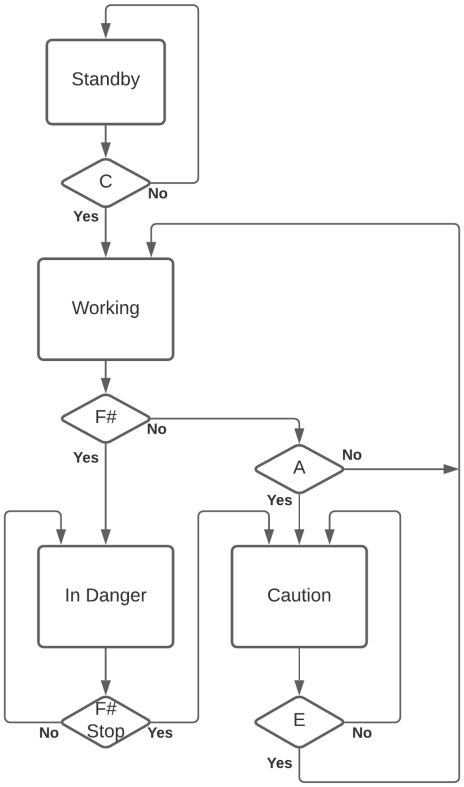

Figure 15: Robot state transition diagram

- **Caution** - Normal working with a slower speed. Solid Amber LED.

When the robot enters a new state, it starts one or more tasks to implement the behaviour of said state. It also cancels the tasks started by the previous state by injecting a `CancelledError` into that task. The task has a chance to react to that error before closing. For instance, any child tasks created can be closed.

This system can be easily extended with more states or more elaborate tasks implementing their features.

E.3 MicroPython Build

As discussed in Section 5.4, the central robot control was written in MicroPython on an ESP32. The version of MicroPython used was built from source specifically for this project. The changes made to the source are as follows:

1. Add the third party module micropython-ulab which only supports up to MicroPython v1.14 on the ESP32.
2. Add support for the newly added ThreadSafeFlag utility only available in MicroPython v1.15 onwards.

E.3.1 micropython-ulab

This library was added to make simplify the implementation of the FFTs. As mentioned in Section 5.4, it implements a subset of NumPy and SciPy, including the vector data-type Ndarray. Several functions are included that perform FFT-like operations on vectors of input samples. The `spectrogram` function was used as it returns the magnitudes as real numbers, rather than complex form produced by basic FFTs. The Ndarray type was also used to implement the circular buffers and a hamming window.

micropython-ulab is implemented in hardware independent C and can be built as a user module for any variety of MicroPython. However, the build system for the ESP32 port of MicroPython has adopted a new build system as of the v1.15 release, and the ulab build system has not caught up. This restricted the team to using MicroPython v1.14 for this project.

E.3.2 ThreadSafeFlag

This is an extension to MicroPython's uasyncio multi-tasking framework. It allows uasyncio tasks to be triggered from code outside the uasyncio system, such as External or Timer Interrupt Service Routines (ISR). This is essential functionality to this project as a MicroPython timer is is used to ensure samples are retrieved from the ATTiny regularly enough to stop the ATTiny buffers overflowing and dropping samples.

This code was added on Feb 6th 2021 (commit 5e96e89), after the ESP32 build system was upgraded, so it could not be easily imported into the existing project. A local copy of the MicroPython source was created, the repository was reverted back to v1.14, and the code implementing ThreadSafeFlag was copied over manually. Then, following instructions listed here[1], the ulab code was integrated into the core MicroPython source and built for ESP32. Reading issue discussions on the micropython-ulab repository, it seems that updating ulab to support the new ESP32 build system was a priority, so it is likely that the aforementioned versioning issue has been resolved.

[1] https://github.com/v923z/micropython-ulab#esp32-based-boards

F Management

F.1 Budget Management

Product	Source	Individual Cost	Quantity	Total Cost
Robot Wheels (Pack of 5)	Amazon	£13.99	1	£13.99
Microphones (Pack of 3)	Amazon	£8.99	2	£17.98
Battery Pack	Amazon	£5.99	3	£17.97
Robot Amplifier (Pack of 5)	Amazon	£5.99	1	£5.99
Castor Wheels (Pack of 5)	Amazon	£5.99	1	£5.99
ESP Microcontroller (Pack of 3)	Amazon	£24.39	1	£24.39
12-bit SPI DAC	RS	£1.818	6	£10.91
Motor Driver	RS	£1.02	7	£7.15
Active Low Pass Filter	RS	£6.792	5	£33.96
RGB LED	RS	£0.978	5	£4.89
Uni Directional Microphone	RS	£2.66	6	£15.96
Robot Speaker	RS	£1.09	3	£3.27
Base Station Amplifier	RS	£1.49	5	7.45
100uF Electrolytic Capacitor	RS	£0.176	5	£0.88
100nF Electrolytic Capacitor	RS	£0.13	1	£0.13
USB-C PD Power Supply	Amazon	£6.82	1	£6.82
Base Station Speaker	RS	£4.05	1	£4.05
12V Low Drop Out Regulator	RS	0.62	1	£0.62
			Delivery and VAT	**£17.88**
			Total Cost	**£200.25**

Table 2: A Table showing the full breakdown of our budget and spending

F.2 Risk Assessment

Personnel Hazard	Severity (1-5)	Probability (1-5)	Risk (S*P)	Mitigating Factors
Contracting COVID-19	5	2	10	All members enrol in the University testing program Group interaction to held online
Injury from Li-Po Battery	5	2	10	Keep contacts separated and avoid shorting Store battery safely at room temperature in a hard case
High voltage shock	4	2	8	Adhere to lab safety conditions Apply common sense

Project Hazard	Severity (1-5)	Probability (1-5)	Risk (S*P)	Mitigating Factors
Delivery delays	3	3	9	Order components early Pay for express shipping if required
Illness/Absence	4	2	8	Spread workload among the team Use meeting minutes to catch-up
Team Breakdown	5	1	5	Maintain clear communication Voice opinions and concerns to the team
Over-budget spending	2	2	4	Adhere to budget allocation All orders to be approved by the team
Data not saved	3	1	3	Save file in multiple locations (Local and on MS Teams)

Table 3: Health and safety risk assessment

G Specification

Items in **bold** have not been fully achieved and/or tested.

G.1 Primary Goals

- Robot can differentiate between multiple frequencies
- Robot autonomous range of 2m
- Weight of device under 300g
- Size within 15cm x 20cm x 10cm
- Battery life at least 1 hour
- Height of base station under 20cm

G.2 Stretch Goals

- Robot is constantly in range of base station signals
- Incorporate a second robot
- **Can navigate uneven terrain**
- **Navigate using detected sound**