University of New South Wales, School of EE&T

Designing a Dalek: A Study in Integrating Signal Detection and Control Systems

Casey Thomas, Freesia Gaul, Sean Coulon-Clark, Aaron See, Matthew Shih,
Jiusi Gao, Hanlin Li.

1. Abstract

Signal detection and control systems are essential to electronic systems design. This report explores how these principles can be leveraged to design a Dalek. The Dalek is comprised of three key systems: a signal detector, a motor that rotates toward a target signal, and an "Extermination sequence" which audibly and visually indicates the Dalek has successfully located the target. A review of other similar projects and experimentation was conducted before describing and analysing the design concepts, which are then used to provide recommendations on replicating and furthering this design. Many methods were found to solve this problem across the analogue and digital space. Ultimately, the final design featured a signal detector that provided peak values to an Arduino, which acted as an analogue to digital converter and then carried out the signal processing. Then the Arduino output values to the control system. This value from the detector then drove the motor driver and speaker. Once the motor faces the target, the speaker breaks from signal jamming and declares "Exterminate," with LEDs synchronised to this audio indicating the target was found. This solution worked as subsystems but could not be entirely integrated due to external factors such as time and soldering mistakes. From this, it can be determined that the provided design works in theory as subsystems and two systems. Since the problems arose from a lack of time and the testing of subsystems was positive, the design is likely functional with further work. This also means that this report may be more helpful for those beginning or actively creating a design but not for those with finished designs that require optimisation.

2. Table of Contents

3. Introduction

3.1 Purpose

This report explores the development of a signal detection and control system in the form of a Dalek. The Dalek functions as a control system that alters its behaviour depending on the signals it receives. The signal detection system is responsible for determining the angle of arrival of infrared or audible signals from a target that emits signals at a predefined frequency. The control system interprets the processed signals and causes the Dalek to react, rotating toward the target until it is aligned with it. Once stopped, the Dalek must set off an auditory or visual signal to communicate that it has completed its search. Classically, the signal is an audible message declaring “Exterminate.”

3.2 Background

This project presents three key objectives: the system must detect, track, and signal when it has located the target. However, there are several constraints and requirements which define the scope of the project. These include limitations in resource availability and access, time, and labour. The project has a ten-week deadline and can accommodate a maximum of eight students. Additionally, it largely relies on axial components and prototyping tools such as breadboards and loose wires. This reliance introduces concerns about noise, and the Dalek designs serve as proof of concepts rather than fully self-contained, reliable systems.

3.3 Scope

The report provides a technical roadmap of the requirements and constraints to assist the reader: describing the Dalek in terms of subsystems to elaborate on the working design principles; and then examining how it was designed to work in synergy as a fully implemented Dalek. Following that there is reflection on what was executed well, and what to improve on, providing recommendations on how to improve the final design.

It is important to note that this report cannot answer some questions. This is because the team could not adequately integrate the final design due to scope creep. Resultantly, the team

could not provide as much insight into the optimisation of the final system, and troubleshooting of the final design, since the team was unable to reach that stage. Despite this, the conclusions reached through the design process and research are valuable in understanding the process of constructing signal detection and control systems.

4. Initial Information and Planning

The planning and prerequisite work analysis precedes the discussion of the design. Effective designs rely on a strong foundation of planning, research, and brainstorming. Despite a standard team consisting of eight people, this team consisted of only seven, due to one student dropping the subject.

4.1. The Design Process

The design process consisted of several key milestones, beginning with a rigorous understanding of the requirements of the design, which aided in the early identification of dominant challenges the design needed to address. Then, research into similar designs was conducted for concept designs. Analysing these ideas and completing preliminary tests, a design concept was chosen and a detailed plan was constructed before the prototyping phase. This prototype was tested, analysed, and adapted based on new information, ideas, and unexpected results from testing. Despite the process seeming linear, there were still undeniably management errors that were difficult to predict. Furthermore, some processes require many repeated steps; such procedures are reflected in this report, especially in *Section 5*.

Throughout the design process, distinct phases were drawn. During the first phase, the team was split into two subgroups, subgroup A and subgroup B. Initially, subgroup A attempted to solve the problem of rotating the Dalek and signalling extermination, while subgroup B focused on analysing the enemy target's signal and outputting voltages according to the angle of arrival. This was done as this project had extensive but also significantly varying objectives, and as such, it was important to distribute the expertise of this seven-person team effectively.

Breaking this team into two subgroups allowed each subgroup to focus on their specific design challenges and the intricacies and nuances associated with them.

4.2. Understanding Objectives and Constraints

The objectives for this design are briefly outlined in the introduction, but the constraints and specifications are expanded upon in more detail below, so the design challenges are more clearly understood.

4.2.1. The Enemy Target

The Dalek must detect and rotate toward an enemy target. The target emits a characteristic audio and IR signature.

The emitter schematic is provided below in *Figure 1*. The target features two emitting signals with frequencies at 20kHz for the infrared (IR) signature and 1.25kHz for the audio signal. In reality, both signals were empirically found to deviate nearly 20% from the ideal signal frequencies mentioned for each unique emitter PCB in the laboratories, due to cheap PCB printing. These frequency variations were not anticipated from the beginning of the design process, so it is recommended that the identification of the real signal levels of the allocated board is prioritised at the beginning of the design process, or to spend extra time considering tolerance adjustments for the detector through self-conducted tests to adjust for these concerns. Further discussion of signal detection, processing, and self-conducted testing is in *Section 5*.

A final challenge associated with the target is that it will be tested in both indoor and outdoor conditions. This raises a further challenge to isolate the Dalek's IR signal when audible and electromagnetic noise is introduced.

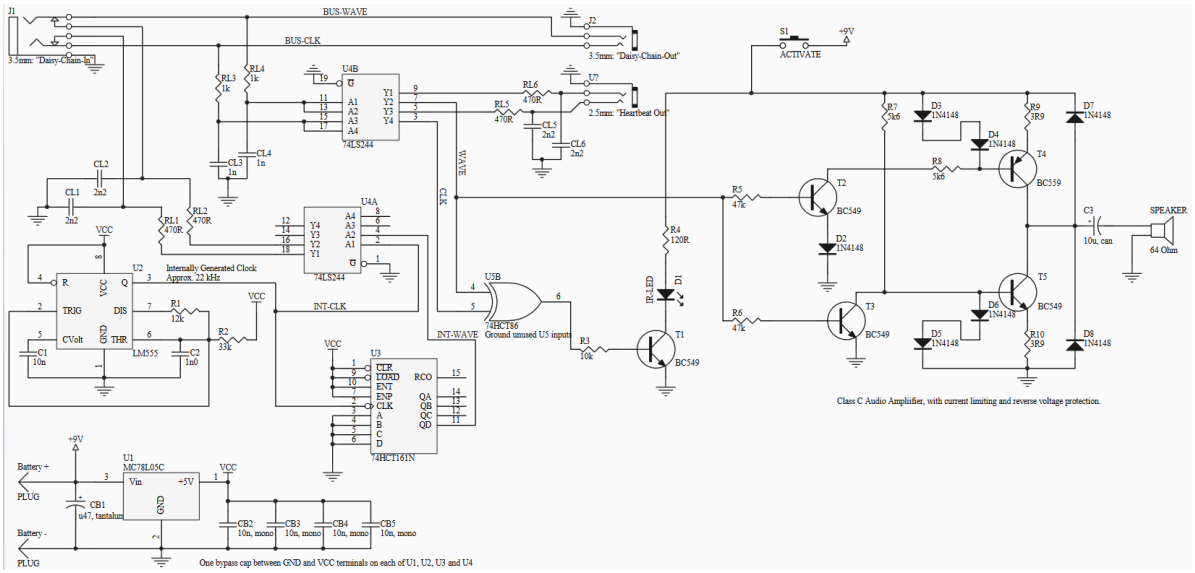

Figure 1: Emitter; Enemy Target Schematic [1].

4.2.3 Overview of Allowed Resources and Associated Constraints

Design constraints arise from the accessibility and availability of resources. The provided components' quality, size, and stock limits introduce boundaries to how well any system can perform. Due to limitations in the quality and quantity of components, circuits had to be optimised and adjusted to ensure performance instead of purchasing the most optimal component for the job. For example, the design's weight and dimensions had to be optimised for the allowed motor rather than choosing the motor that best accommodated the Dalek's weight. Similarly, cascading operational amplifiers to increase gain in the detection circuits were used as a workaround to them not being ideal.

5. The Subsystems

This Section discusses the subsystems which make the Dalek a working system. It outlines the challenges that each subsystem attempts to solve and the design solutions that were

developed in response to each subsystem. The final integrated solution will be summarised in *Section 6*.

As illustrated by *Figure 2*, the subsystems include detecting and filtering the enemy target's signal, analysing this signal, using the analysis to control an extermination response and rotating the Dalek. Rotating the Dalek involves controlling a motor, creating a Dalek model, and interfacing these two systems. There is also a discussion on competitive design strategies. Power distribution to these systems will be discussed in *Section 6*.

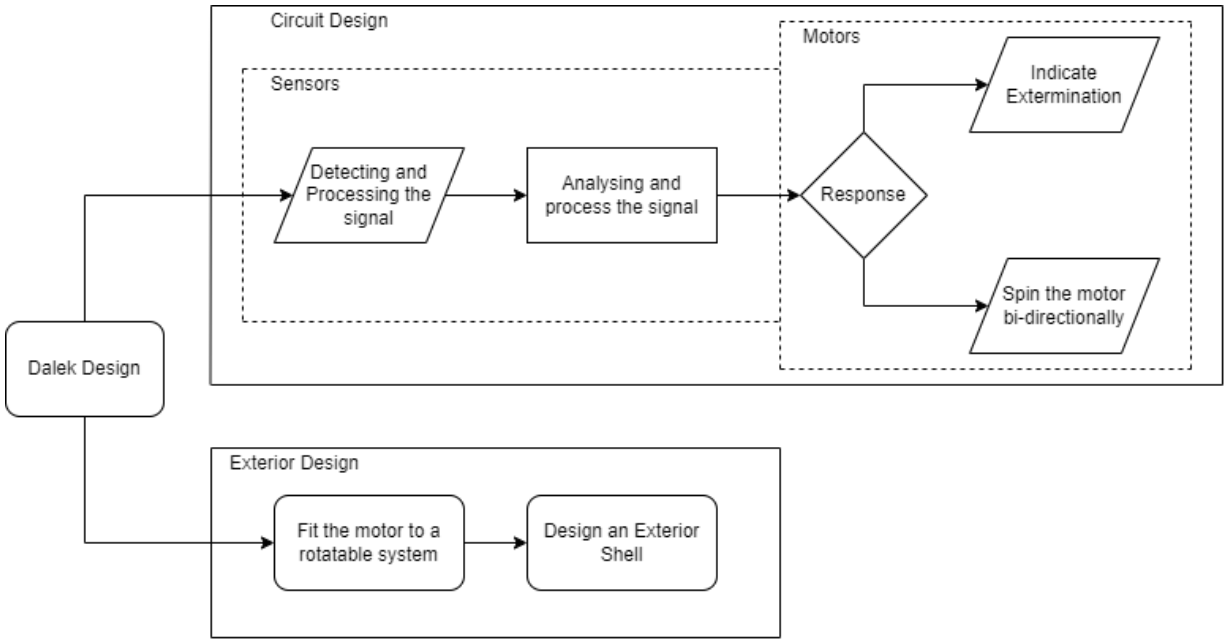

Figure 2: Flow Diagram showing the breakdown of the Dalek design

5.1. Emitting Source Detection

As discussed in *Section 4.2.1*, the enemy target emits an IR frequency of 20kHz and an audible frequency of 1.25kHz. Using the components given, the design requires a detection system to be made, taking advantage of these signals.

5.1.1. IR Detector

The design required to detect the enemy target's IR signal with two L-53P3C IR photo-transistors and output a signal that can be analysed by the system in *Section 5.2.*

The team initially researched, looking for similar work to generalise and learn whilst also taking onboard ideas from the lecturer and other teams throughout the design process. This was especially useful in the brainstorming process for the development of angle of arrival (AoA) methods. This research phase was in areas such as direction finding (DF) and more general telecommunications theory. Rohde and Schwarz material proved to be useful in the initial introduction to AoA and DF methods [2], [3].

This acquisition of knowledge allowed the team to explore ideas for both physical placements of the sensor on the model with reference to real-world sensor designs, as well as prioritise the most relevant areas to learn first. This allowed for the signals team to quickly design their first model for the operational-amplifier in Fig. 3 which was essential knowledge for the later construction of the detector.

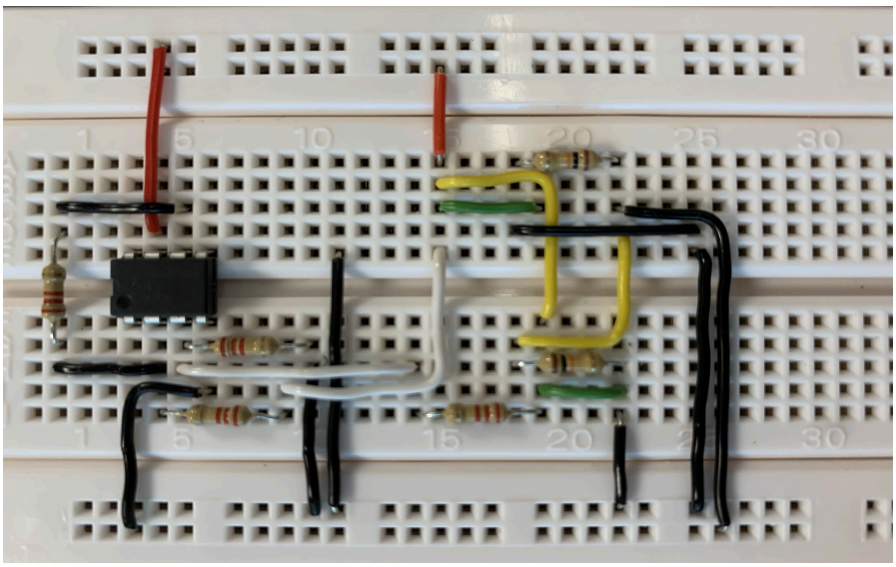

Figure 3: Initial Op-amp Configuration testing that went on to be incorporated into the final design

In parallel, the signals team focused on finding filter research to generalise from, one useful source came from [4], notably a figure from their paper involving a band pass filter (BPF) and a peak detector in Fig. 4. This helped focus the team's attention when the same concepts arose in lectures during rapid prototyping.

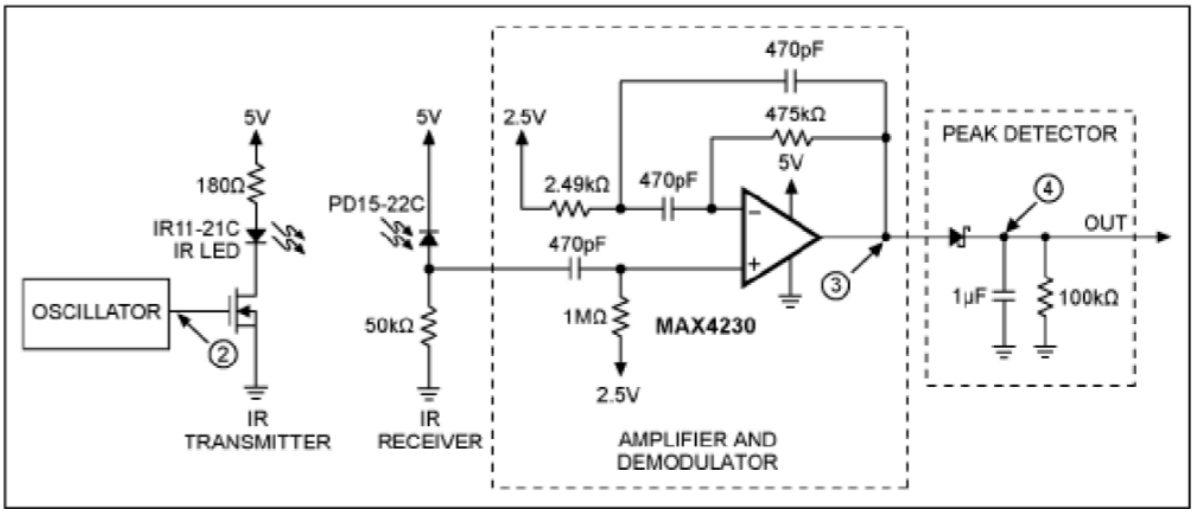

Figure 4: A simple IR transceiver that inspired our band-pass filter and amplification approach, given by [4, p.2]

This figure helped transform components into a bigger picture, inspiring the later design. One other area of inspiration was from a household appliance, wondering how TV remotes and industry sensor worked, looking at schematics for existing detectors, specifically the GP1UX51QS Series which provided an insightful internal block diagram given in *Figure 5* taken from [5, p.2].

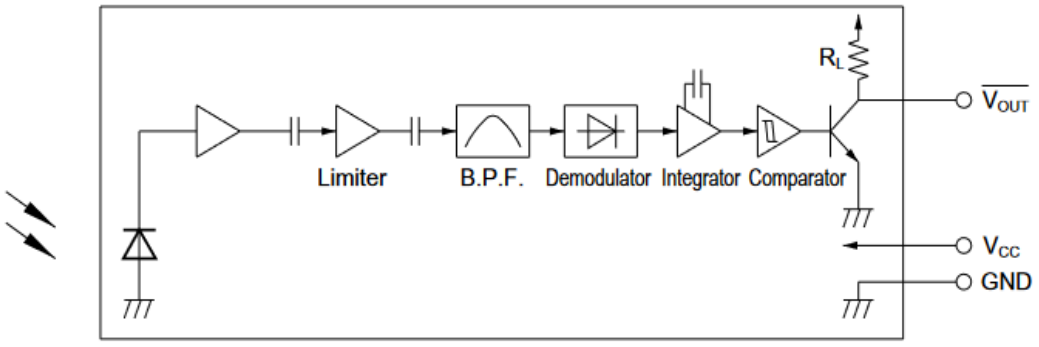

Figure 5: Internal Block Diagram of Holder-less Type IR Detecting Unit for Remote Control, given by [5, p.2].

From here, the final detector circuit was created, but underwent continuous tests and redesigns. The logical flow of the detector implemented and detector schematic is outlined in *Figure 7*, and *Figure 6* respectively. Firstly, a signal is received by a L-53P3C phototransistor. Following the signal's detection, it is buffered by a voltage follower using an LM741 to prevent attenuation in subsequent stages. The signal is then amplified by two cascaded active band-pass filters (BPF's) in an LM348, allowing only the 20 kHz frequency to pass while amplifying it. A second voltage follower buffers the filtered signal, isolating it from interference or resonance before it goes through the final BPF - the follower and last BPF; these are also within an LM348. Then, the refined signal is fed into a peak detector, where an op-amp buffers it. The 1N4148 diode then rectifies it by passing only the positive peaks, and a capacitor-resistor pair holds the peak voltage momentarily while slowly discharging, which is useful in performing difference calculations on the Arduino once read in. Finally, this DC signal is reduced through a Zener diode, 1N4148, to be at a cooperative voltage level with the Arduino's 0-5V input range.

Pivotal to the design is the cascaded active resonant band pass filters. They utilise a resonant frequency to create a higher Q value for the desired frequency. The Q value, seen as a signal-to-noise ratio, was an important consideration, as the circuit and its target had significant noise. A system of equations can be formed from the band pass filter form:

$$f_c = \frac{1}{\sqrt{R_1 R_2 C_1 C_2}},$$

$$Q = \frac{1}{2}\sqrt{\frac{R_2}{R_1}},$$

$$A_{max} = -\frac{R_2}{2R_1},$$

Where f_c is the centre frequency of the BPF, Q is the quality of the band pass filter, and A_{max} is the maximum gain of the band pass filter.

Using the values:

$$f_c = 20000,$$

$$Q = 1,$$

$$A_{max} = 10,$$

$$\text{and } C_1 = C_2 = 10^{-9}.$$

In the end, the team arrived at a solution, balancing range, precision, and op-amp restrictions. The effect of the band pass filter is modelled in *Figure 8* under ideal conditions. The shown response does not reflect the actual response, as the op amp will fail to respond sufficiently to anything above 25kHz due to GBW limitations.

Overall, the design is effective due to its robust filtering system, strategic placement of voltage followers to isolate and enhance signals, and the use of multiple BPF stages for sharp frequency discrimination. It balances precision and simplicity, ensuring accurate detection while minimising noise, complexity, and interference. This band-pass filter was created to more effectively isolate the desired frequency than a typical inverter form.

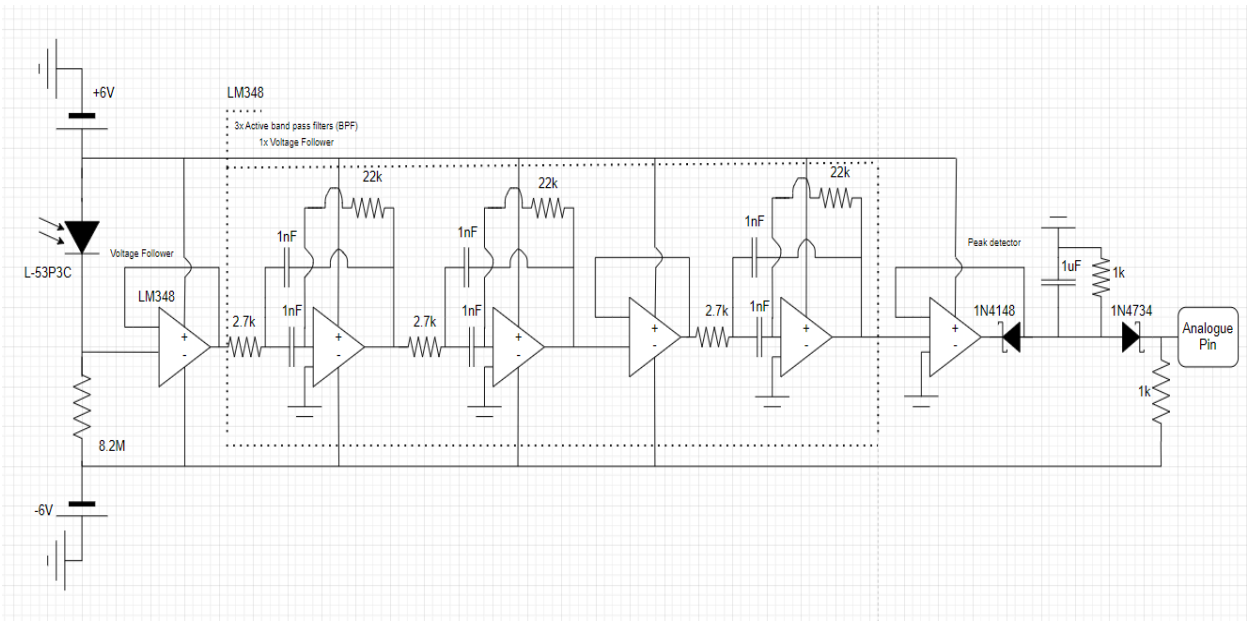

Figure 6: Final Detector Design

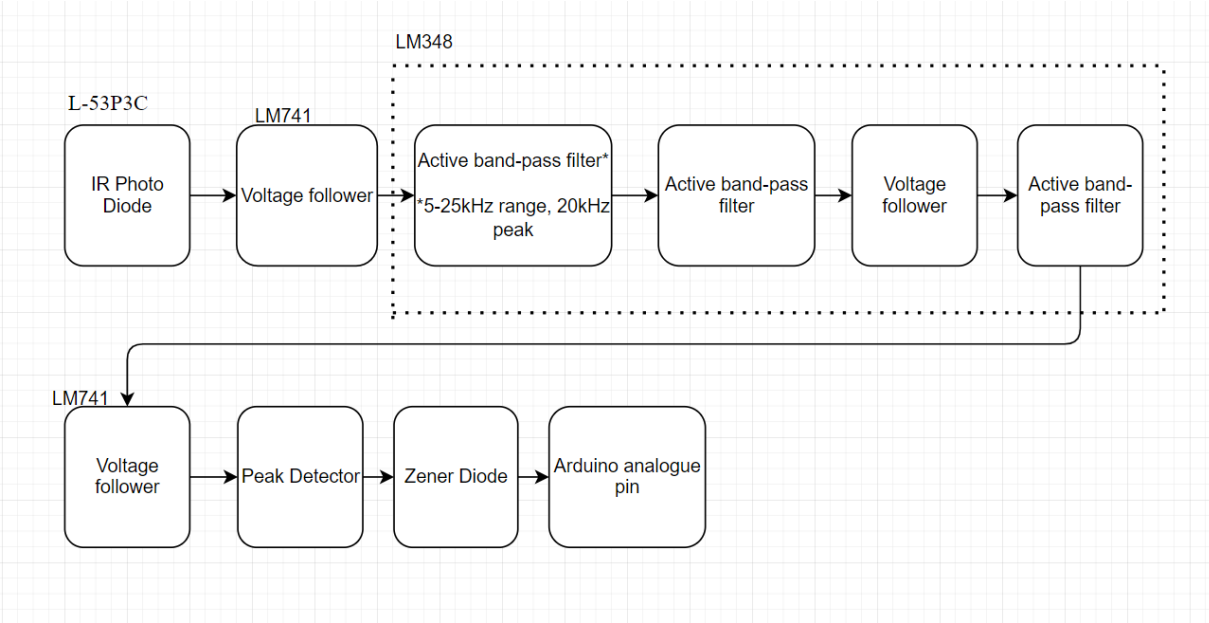

Figure 7: Final Detector Logic Flow Chart

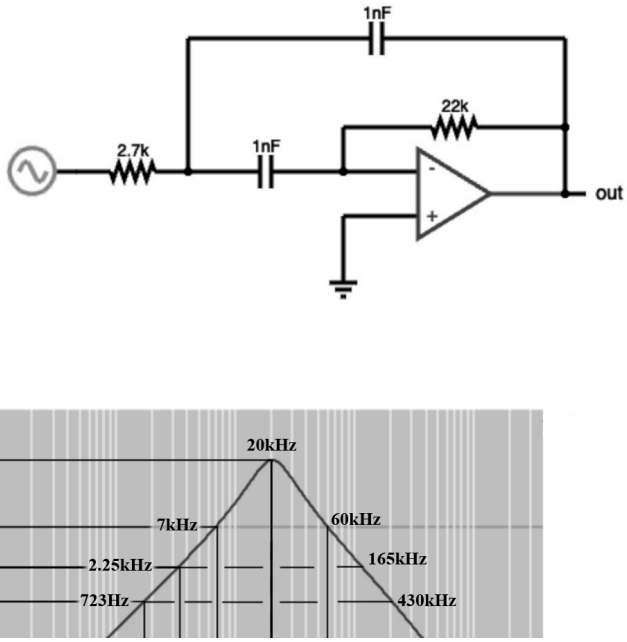

Figure 8: Band pass filter response mapped in Falstad showing the frequency attenuation. Peak value at 20kHz

5.1.2. Audio Detector

The enemy target emits a characteristic audio at a ~1.25 kHz frequency. While IR can find the target, auditory signals are a useful way to error correct in challenging environments for IR sensors, such as in direct sunlight. Ultimately, the design did not address the use of auditory signals for error correction, as signal jamming in the auditory range was pursued instead. Signal detection using the allowed AM4010 microphones was often temperamental and distorted. Additionally, sabotage from other teams meant that the group's allocated audio jack was repeatedly stolen, and thus, audio detection was further deemed unreliable. External disruptions to the audio detector were more difficult to block out using physical means, whereas the IR detector could have physical modifications done to the shell based on empirical tests outdoors to

improve range. Hence, IR was the avenue pursued, aiming to make it as polished and resistant as possible.

5.2. Signal Processing

This subsystem was required to analyse the outputs from the IR detectors in *Section 5.1* such that the detector circuit could communicate AoA information with the control system.

This team chose to use the Arduino as an analogue to digital converter (ADC) and to perform operations using code instead of using an analogue circuit due to the Arduino being more consistent, efficient – given the time constraints, and customisable. Methods of signal processing were explored, the most intuitive and scalable processing method was implemented using ratio calculations on voltages read in from the peak of the signal.

The Arduino was fed two analogue inputs from the detector circuit, acting as an ADC, reading in the peak values of the signal and converting them to a ratio using code; see *Figure 9* for a code snippet outlining the calculateRatio() function.

```
// Function to calculate the normalized ratio difference
float calculateRatio(int left, int right) {
  int difference = left - right;
  int sum = left + right;

  if (sum == 0) return 0;  // Avoid division by zero

  return (float)difference / sum * 100;  // Ratio in range [-100, 100]
}
```

Figure 9: Signal-ratio code showing the process of calculating the ratio between the two sensor inputs.

The code works off the principle of reading in the two analogue inputs from the detectors and computing the ratio of the peak values of the signals. This ratio is then compared against a

user-defined ratio which can be empirically adjusted based on the angle one requires for a target to be found. To find the ideal ratio value, the team empirically tested with the target and detectors, moving it 180° around the target and finding the most precise and farthest reaching IR signal given the adjusted ratio value.

5.3. Extermination Signal

This subsystem consists of recognising when an extermination should be completed and designing an extermination sequence.

5.3.1. Determine When to Exterminate

The system must first determine when an extermination must happen, as determined by communication with the IR sensors and filters from *Section 5.1.*

Several communication methods were considered for a minimum voltage input. As many of the subsystems move towards Arduinos, choosing a method that effectively interfaces with Arduinos became necessary. Instead of interfacing between multiple Arduino's, just one was used. This problem was solved by the subsystem presented in *Section 5.2*, which already provides a condition for determining when an extermination should occur. All tests and analyses of this design are completed in *Section 5.2.*

5.3.2. Controlling the Speaker

The first part of the extermination sequence controls a speaker; the LED sequence will be discussed in *Section 5.3.3*. The challenge here is to create a loud and obvious noise when an extermination occurs.

The final design uses David Mellis' Arduino PCM library. [6] To use this library, an audio file was first acquired to play. This was done using 101 soundboards, a Dalek text-to-speech (TTS) application [7]. The audio stated, "Exterminate for our supreme leader David." a sequence of numbers was generated using a PCM audio encoder, which was output through pin 11.

The output from pin 11 controlled an AS3000 speaker. Since an Arduino can only supply small currents, a BD139 NPN transistor was used to amplify this signal. This design placed the

NPN transistor between the Arduino and speaker so that the Arduino's output was at the transistor's base, an external 5V voltage source was connected to the collector, and the emitter was connected to the ground through the speaker.

Ultimately, this design allowed for easy control through transistor and input voltage choices.

5.3.3. LED Synchronisation

The second part of the extermination sequence was required to control LEDs. The design challenge here was to use LEDs in a creative and obvious way to indicate extermination.

In this design process, preliminary designs were not abandoned but improved. This began by using one LED that the Arduino turned on when required. This was basic, hence three additional LEDs were added and manually synchronised through code to match the syllables spoken in the extermination audio. This was done by first determining the temporal location of each syllable and calculating the time interval between each syllable. Next, each interval was split into two segments, one where the LED was on, and another where it was off. Further ratios were tested such as ⅓ off and ⅔ on until an ideal design was determined. In the final design, it was found that ratio thresholds from *Figure 10* looked best and aligned the nicest.

However, this method increased code complexity, so it was optimised with a for-loop shown in *Figure 10*, which looped through a list of on and off times as time delays.

```
// array of on and off times for the exterminate LEDs whilst
// the speaker is playing
int timeOn[17] = {150, 150, 50, 150, 100, 100, 50, 150, 100, 100, 100, 150, 50, 50, 50, 50, 50};
int timeOff[17] = {150, 150, 50, 150, 100, 100, 50, 150, 100, 100, 100, 1000, 50, 50, 50, 50, 50};

// if both the left and right inputs indicate a low signal, then the
// Dalek completes a large turn/pulse
if (leftInput < lowVoltageThreshold && rightInput < lowVoltageThreshold) {
  digitalWrite(turnRight, HIGH);
  digitalWrite(turnLeft, LOW);
  delay(1000);
  digitalWrite(turnRight, LOW);
  delay(500); // delay gives the Dalek a chance to stop and readjust

}

// if the difference between outputs is small enough and each input is
// large enough, then the Dalek stops moving and begins its extermination sequence
else if (abs(readingDifference) < lowReadingDifference && (leftInput >= highVoltageThreshold && rightInput >= highVol
{
  // Dalek stops moving
  digitalWrite(turnRight, LOW);
  digitalWrite(turnLeft, LOW);

  //Begin speaker audio
  startPlayback(sample, sizeof(sample));

  // Begin LED sequence
  for (int t=0; t < 9; t+=1) {
    digitalWrite(whiteLED, HIGH);
    digitalWrite(redLED, HIGH);
    delay(timeOn[t]);
    digitalWrite(whiteLED, LOW);
    digitalWrite(redLED, LOW);
    delay(timeOff[t]);
  }
}
```

Figure 10: LED sequence code showing how the LED sequence was compressed and executed.

5.4. Rotating a Motor Bi-Directionally

This design is required to control the bi-directional rotation of an MM28 motor. [8]

This team decided to rotate the motor in small pulses towards the apparent direction of the target. As shown in *Figure 11*, this was done by turning the motor on and off with a delay after

```
// if the difference between outputs is small enough and each input is
// large enough, then the Dalek stops moving and begins its extermination sequence
  else if (abs(readingDifference) < lowReadingDifference && (leftInput >= highVoltageThreshold && rightInput >= highVoltageThreshold))
  {
    // Dalek stops moving
    digitalWrite(turnRight, LOW);
    digitalWrite(turnLeft, LOW);

    //Begin speaker audio
    startPlayback(sample, sizeof(sample));

    // Begin LED sequence
    for (int t=0; t < 9; t+=1) {
      digitalWrite(whiteLED, HIGH);
      digitalWrite(redLED, HIGH);
      delay(timeOn[t]);
      digitalWrite(whiteLED, LOW);
      digitalWrite(redLED, LOW);
      delay(timeOff[t]);
    }
  }
// if the right input is larger than the left input, then
// rotate the Dalek right
  else if (rightInput > leftInput) {
    digitalWrite(turnRight, HIGH);
    digitalWrite(turnLeft, LOW);
    delay(500);
    digitalWrite(turnRight, LOW);
    delay(400);
  }
// if the left input is larger than the right input, then
// rotate the Dalek left
  else {
    digitalWrite(turnRight, LOW);
    digitalWrite(turnLeft, HIGH);
    delay(500);
    digitalWrite(turnLeft, LOW);
    delay(400);

  }
```

Figure 11: Motor pulsing code, showing how pulses can be created using Arduino code

each command. Determining the direction of this pulse was discussed in *Section 5.2.*

The L298N motor driver module powered these pulses. Specifically, pin 7 and pin 8 of the Arduino were connected to the motor driver's in 1 pin and in 2 pin, respectively. Since the speed of the motor did not need to vary, it was consistently maximised by connecting a constant 5 V signal to this pin.

This prepackaged system was chosen as it increased consistency, prevented the IC from overheating and extra safety measures with capacitor systems and other onboard ICs. The pulsing motion also allowed increased accuracy and effectively eliminated the motor's overshooting problem when breaking.

5.5. Designing the rotating platform

The design must be able to rotate a plunger-wielding platform with the motor system from *Section 5.4*. This required an interface system between the platform and motor.

First, it was determined that the motor would rotate a circular stand on which the Dalek exterior could be mounted, shown in *Figure 12*. A planetary gear system was used to transfer the motor's rotation to this platform. This system is shown in *Figure 12*, where the motor rotates the central gear. This central gear rotates the three outer gears. The circular stand is then mounted on this system with a planetary ring at its base. The main motivation behind this design was to allow wires to pass from below the platform to above the platform internally, without twisting or moving the wires. The gear system was also chosen to maximise the torque provided by the motor at the expense of speed, with an overall gear ratio between the planetary ring (Outer gear) and sun gear (Central gear) of 5:1. This enables the motor to accurately (slowly) rotate, allowing the system to accurately rotate a higher mass.

This ratio was selected based on preliminary tests of other ratios. Furthering testing showed that the 60:12 teeth ratio was just enough to rotate the weight of the exterior (discussed in Section *5.6.*) with enough speed.

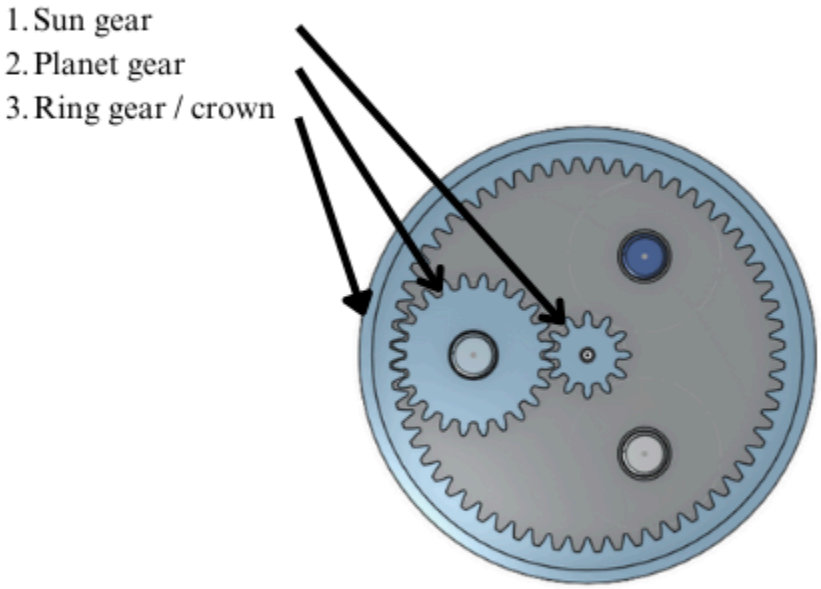

Figure 12: Planetary Gear System showing the sun gear, planet gear, and ring gear/crown, and their locations on the platform.

5.6. Designing and Producing the Dalek's Shell

The team also needed to create an exterior shell that was required to hold the rotating platform and improve the aesthetics of the design, so three extra tasks had to be completed: choosing dimensions, shell production, and decoration.

3D printing was the most versatile and lightweight method for creating the exterior shell. Research was conducted to see if there were available Dalek STL files to save man hours, however, they lacked quality or were too detailed, so a new model had to be created for the exterior. To minimise the model's weight, the Dalek had to be simplified, removing elements such as its whisk. The model was also designed with integration of circuitry in mind, along with internal shelves to organise circuits, and a battery storage in the base.

The 3D printed exterior had been made as minimal as possible to reduce the overall weight, and further physical alterations were made after the Dalek was printed. Four holes were drilled around the Dalek's lower perimeter to hold the extermination LEDs, and two more holes drilled in the Dalek's upper vents, each 10° degrees from the centre. A cone was placed over these holes to mitigate incoming noise.

This exterior was interfaced with the platform in *Section 5.5.* by placing the platform on the Daleks base and resting the Dalek's neck and head on the spinning stand, as shown in *Figure 13*. An illusion of a metal dalek was created, through paint. Rather than giving the polylactic acid (PLA) a fake metallic shine, layering different paint colours resulted in a rusted appearance, meant to mimic oxidising metal. See *Figure 14*.

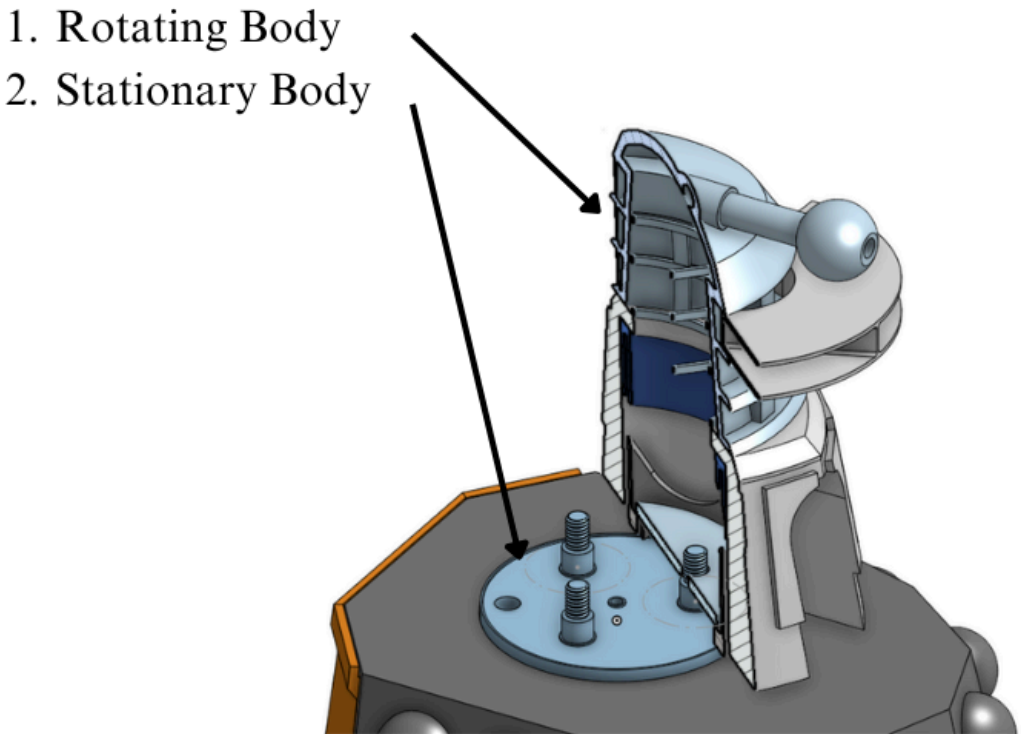

Figure 13: Dalek rotating shell showing how each Section was designed and assembled.

Figure 14: Dalek exterior showing the oxidised copper paint design.

5.7. Competitive Design

Since there is a competitive element required for this design, it became evident that sabotage would be advantageous. The design had not utilised audio detection for error correction and detection, so an was implemented to signal jam opponents. Using Arduino code from [9], the Dalek played “Never gonna give you up” by Rick Astley, implementing the *tone* function, which sent out an audible signal near the provided audio range using a square wave to sabotage opponents in the competitive phase [10]. Different tempos were tested until the desired frequency range was acquired [11]. This was done on a second Arduino for easier modification on different design assessment phases.

As shown by *Figure 15*, the song was set to play indefinitely but could be turned off through an inter-Arduino signal which disables the looping function of the jamming noise. When the motor Arduino outputted a low voltage, the song was on, when the Arduino communicated a high voltage, the song turned off and the extermination sequence began.

The configuration of this speaker was identical to that of *Section 5.3.2.*

```
void loop() {
  // iterate over the notes of the melody.
  // Remember, the array is twice the number of notes (notes + durations)
  float anread = analogRead(inputPin);
  if (anread < 613) {
    for (int thisNote = 0; thisNote < notes * 2; thisNote = thisNote + 2) {

    // calculates the duration of each note
    divider = melody[thisNote + 1];
    if (divider > 0) {
      // regular note, just proceed
      noteDuration = (wholenote) / divider;
    } else if (divider < 0) {
      // dotted notes are represented with negative durations!!
      noteDuration = (wholenote) / abs(divider);
      noteDuration *= 1.5; // increases the duration in half for dotted notes
    }

    // we only play the note for 90% of the duration, leaving 10% as a pause
    tone(buzzer, melody[thisNote], noteDuration * 0.9);

    // Wait for the specief duration before playing the next note.
    delay(noteDuration);

    // stop the waveform generation before the next note.
    noTone(buzzer);
    float anread = analogRead(inputPin);
    if (anread >= 613) {
      break;
    }
    }
  } else {
    noTone(buzzer);
  }

```

Figure 15: Signal jam code showing how the melody is played, loops, and can be stopped.

6. Final Design

Now that each subsystem has been described, their spatial and electrical connections will be further analysed. This Section will also discuss how the final design was tested and the implications of such tests.

6.1. Inter-circuit connections

6.1.1. Circuit Logic

Connecting each circuit system to one another required careful considerations of current amperage drawn by each component, and the voltage which needed to be supplied for the circuit to work.

The flow chart in *Figure 16* describes the high-level design logic. Each IR photo-transistor receives a signal from the enemy target, amplified and filtered by the IR detector circuit. These IR detectors output to the Arduino's A1 and A2 analog pins. The ratio of the sum of these inputs to their difference is found and compared to the defined ratio threshold. Varying the ratio threshold alters the system's precision.

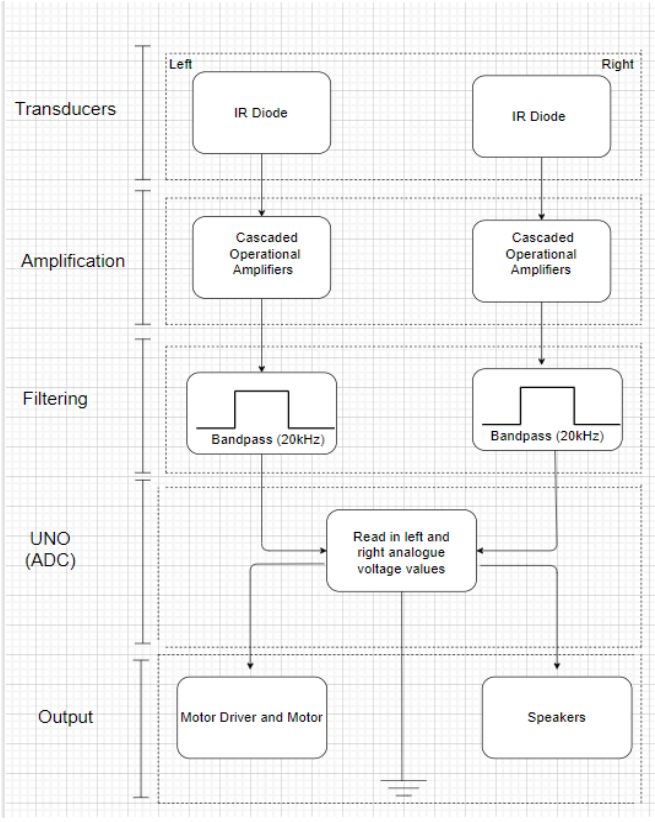

Figure 16: IR sensor flow chart breaking down the specific sections of the circuit

The Arduino provides two outputs to the motor driver from pins 7 and 8 according to the system's current directional state. Whether these inputs are high or low affects directly determines their corresponding outputs.

When the exterminate state is reached, both pins 7 and 8 provide low outputs. The LED sequence is output from pins 2 and 4, each connecting to ground through a 300 ohm resistor in series with two LEDs in parallel. The speaker sequence is output from pin 11 to the base of an NPN transistor which amplifies the signal and sends it to the speaker.

Pin #	**Application**
Pin 7 & 8	Provide the logical states for the H-bridge. The H-bridge is then connected to a 5V source and a 12V power source. This motor driver then outputs to the motor.
Pin 11	Controls the exterminate speaker by sending a controlled PWM signal to the transistor which amplifies the current through the speaker.
Pin 2 & 4	Control the exterminate LEDs, by outputting high or low voltages to ground through the LED and a resistor.
Pin 13	Connected to the second Arduino to control when it turns off and on. This is done by keeping the second Arduino on when the input is LOW and turning it off when the input is high. The output from this second Arduino goes to a speaker through a transistor amplifying circuit just like the other speaker.

Figure 17: Pin mapping table explaining what pinsouts and their purposes on the primary Arduino

6.1.2. Battery configuration

The power distribution of the circuit is critical to the overall design, as it transforms circuits into real systems. The discussion of power was delayed so that each individual circuit working rough understanding of the design before understanding how it is powered. The problem here is that all of the circuitry must be powered by a maximum of 8 D-size 1.5V alkaline batteries. Below is a table outlining how the voltages are distributed across the design.

System	Voltage Requirements
Sensors	+6 V to -6V, with GND in between. (With reference to the Arduino.)
Speakers	5V
Arduino	8V
Motor Driver	Two 5 V low current inputs, one 12V higher current input

Figure 18: Voltage requirements table

Each speaker required a 5V source. The two Arduinos each required an 8V source. The Motor driver required a 5V signal to its enable pin and 5V input pin. This driver further required a 12V high current source, to power the motor. The IR detectors also each need a 12V range, with their grounds 6V above all other circuit systems.

One of the main challenges was addressing the inconsistencies of batteries compared to power supply units, having less regularity in supply current and voltage. To combat this, the team

implemented voltage regulators into the design. The only regulators that could be obtained were LM7805 (5V, 1A), 7808/LM340T8 (8V, 1A), and LM7812 (12V, 3A). According to *Figure 19*, an 8V regulator was placed between 6 batteries (9V) and the Arduino. The 5V regulator was placed between 4 batteries (6V) and the two power transistor speaker amplifiers. The 5V regulator was further connected to its EnA pin and 5V input pin four. Finally, the 12V regulator was placed across the entire circuit and the motor driver's main power.

It is important to notice that ground for the IR detectors was 6V above all other grounds. This must be taken into account during any new design concepts.

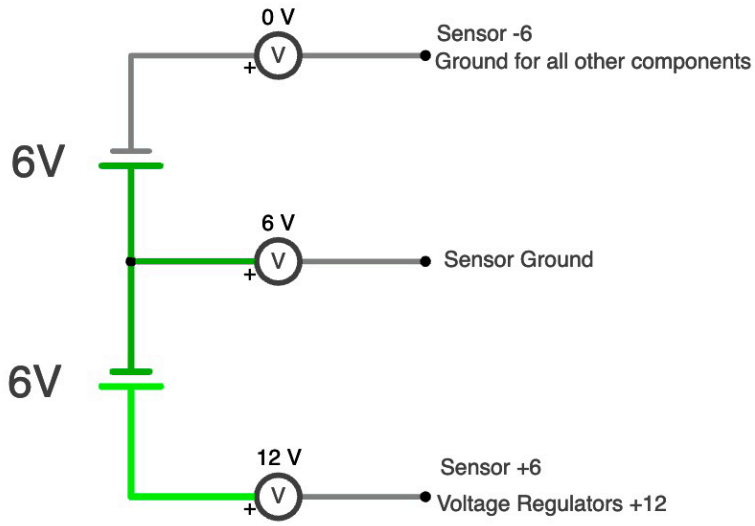

Figure 19: Voltage requirements showing the voltages taken by each subsystem

6.1.4. Spatial Organisation

The arrangement of the circuitry in the Dalek was designed to optimise space and facilitate intercircuit connections. *Figure 20* shows where circuitry is placed within the Dalek. The batteries are placed in the base so that they do not hinder the visibility of the rest of the

circuit. Next, the LEDs are mounted in several equally spaced holes around the Dalek's lower perimeter. Shelves section the rest of the Dalek's interior. The circuitry is placed on the shelves in ascending order: Speakers, Arduinos, motor driver, motor attached to rotating platform.

Within the Dalek's head, the IR detectors are placed in two openings around the front of the Dalek and the IR detectors are attached to the inner walls of the head. These communicate with the main Arduino through a small hole in the non-rotating part of the platform. The Dalek's design ensured that the wires can travel down through this hole without getting twisted with the plunger - the worst case scenario being the wires getting bumped.

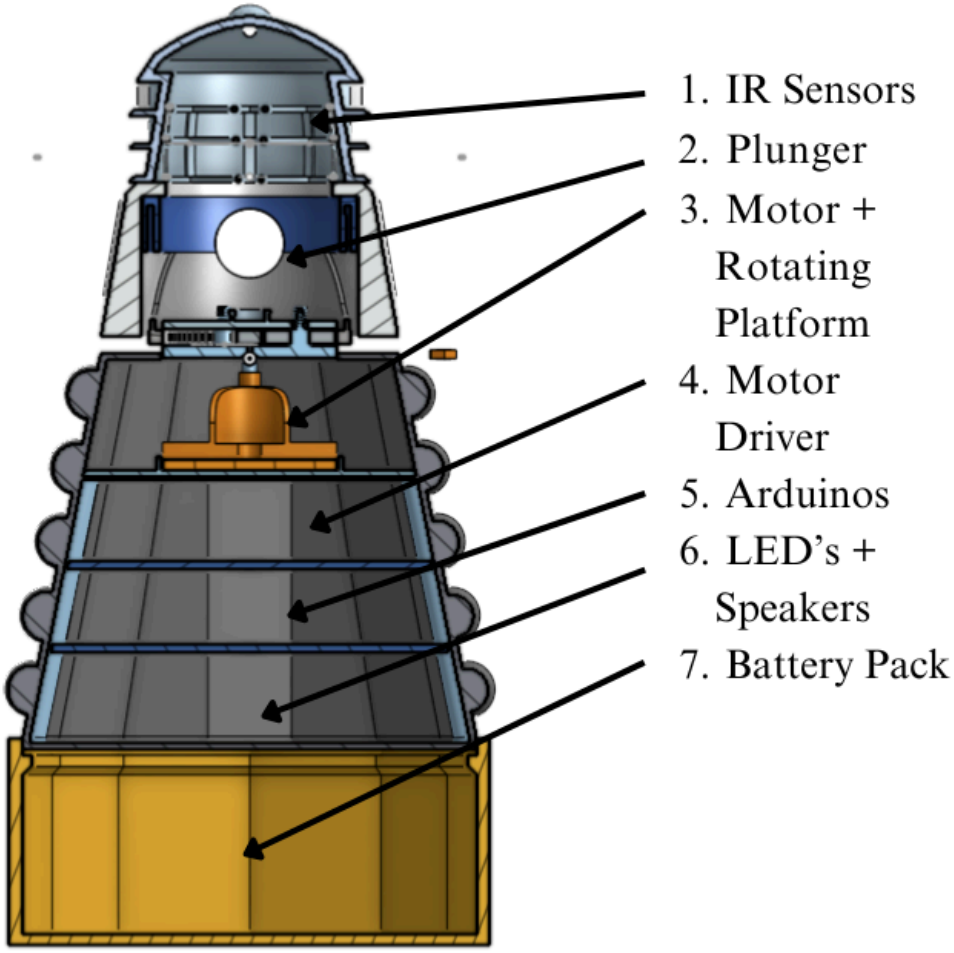

Figure 20: Dalek circuitry placement showing a cross Section view of the Dalek. Call outs show placement of circuitry

6.2. Testing the Final Design

6.2.1. Self-conducted Tests

In various light conditions, the distance between the Dalek and the target was increased until it appeared unable to detect the signal accurately. *Figure 21* shows how different lighting conditions affect the Dalek's range.

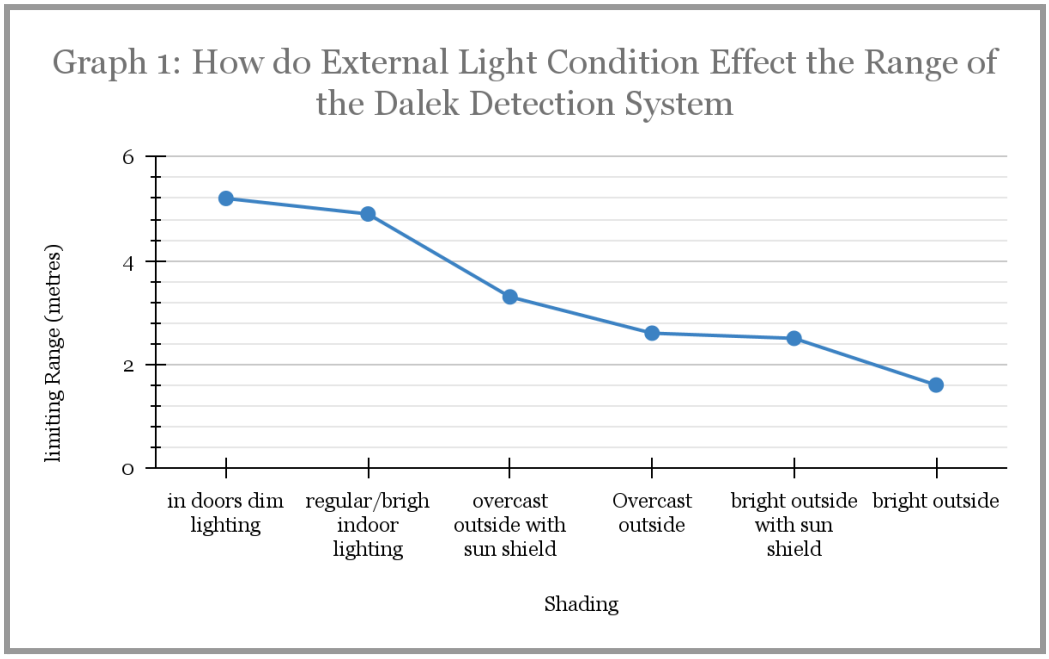

Figure 21: Distance/Light graph showing the response of the photo-transistor in varying light levels and situations

In this system, the motor is turned on for 500 ms and turned off for 400 ms, each pulse represents approximately 10° degrees

$$\text{number of pulses} = \frac{180}{10},$$

$$\text{number of pulses} = 18,$$

As each pulse, including breaking time, take 900 ms, the time it takes to turn 180°, or the maximum possible detection time would be:

$$t_{max} = 18 \cdot 900$$

$$t_{max} = 16200\text{ms}$$

$$t_{max} = 16.20\text{s}$$

Therefore it would take a maximum of 16.2 seconds to find the target, this is an extreme case, most exterminations will be much quicker.

6.2.2 Specifications

The total mass of the Dalek weighs in at about 835g and the rotating shaft weighs in at around 200g.

To find the approximate speed the team uses the pulse value given from *Section 6.2.1*, given the formula $v = \frac{\Theta}{t}$, given that θ equals 10° over a period on 0.9 seconds gives the team an angular velocity of 11.11 rad/s

The power consumption of the entire dalek can be calculated using an estimation of the average energy drawn from the battery pack, if we consider power is drawn from each of the main components when running, Motor (10W), Arduino (0.12W) 2x, Motor Driver (0.1W), Sensors (0.1W) x2 equalling to 10.25W.

To determine the total runtime of the Dalek the team used the capacity of 8 D-size batteries which is 15,000mAh in series. The batteries are run in series hence that is the total capacity. Given an estimate of 2.1A of current draw when the Dalek is in operation. 7.14 hours of operation time.

Specifications
835 Grams
11.11 Rad/s

10.25 Watts
7.14 Hours

Figure 22: Specifications

6.2.3. Final Performance Testing

The final performance testing was completed with the IR detector separate from the rest of the design. This is because the team decided to solder the circuitry onto the perfboard at the last minute, which did not account for inevitable difficult-to-identify mistakes. This reveals the importance of allocating and effectively planning the time leading up to this final performance test. It proved highly unnecessary to solder the design to perfboards. If instead that time had been spent consolidating and optimising the final design on breadboards, a much more successful performance, like the ones tested in *Section 6.2.1*, would have been achieved.

However, this experience revealed the importance of having several backup constructions of the design for circumstances such as this one. However, despite these flaws, the extermination audio, "*Exterminate for our supreme leader David*," garnered an extra point during the performance.

7. Recommendations

Here, the final design concepts are reflected upon. There will be discussion of unrealised design concepts and further extensions past the scope of this design's objectives. Finally, notable challenges that are characteristic of this design and their related solutions will be described.

7.1. Analysis of the Final Design

7.1.1. Good Design Concepts

The planetary gear system in *Section 5.5.* alleviated many weight, wire tangling, and aesthetic issues. The gear ratio between the planetary ring and sun gear of 5 : 1 also maximised the torque provided by the motor. Adding helper functions to the Arduino code, like in *Section 5.2* was crucial to reducing the memory load on the Arduino, enhancing code readability and integrating multiple coding files. The open-back design from *Section 5.6*. allowed quick access to the circuitry components for debugging, and allowed for easy and safe transport of the circuit.

The ratio threshold discussed in *Section 5.2* could be adjusted to vary the system's angular precision. This allowed the team to balance accuracy and target locating speed. Cascading multiple BPFs in *Section 5.1.* Allowed multiple lower quality op amps to be used in place of a more ideal op-amp.

7.1.2. Bad Design Concepts

D-size batteries were very large and often provided more power than was necessary. Smaller batteries would allow more space for other circuitry and minimise power wastage.

The use of perfboards proved to be a bad design concept. The design worked unsoldered, and the groups that appeared to excel in the final performance testing had not soldered their designs. Whilst soldering is usually good, it becomes a bad concept when not implemented with adequate time and resources. The use of plastic gears resulted in the gradual loss in friction between the motor shaft and the gear bore hole. More resilient materials such as wood or metal would provide a longer use time of each gear before they had to be changed.

7.2. How this Design Could be Furthered and Improved

7.2.1. Unrealised Design Ideas and Extensions

Many of the design ideas that were planned but not implemented revolve around design optimisation.

It would be beneficial to have different pulse intensities (longer delays before turning the motor on and off) based on the apparent distance of the target. This would optimise the speed at which the Dalek can rotate towards the target when it is far away and increase the precision when the target is nearby. Due to time constraints the team was not confident to move forward with this approach.

Further, it would be beneficial to use one speaker for the audio signal jamming and the extermination method. However such a challenge requires overcoming the single threaded nature of the Arduino, such as by implementing the *millis* function, and integrating two independent but complicated code files. The *millis* function can make the Arduino act as a soft state-machine, making it appear as if it is running tasks in parallel by allocating time slots for each program to run [13].

Another extension surrounds the sensor system; specifically this involves using better sensors, operational amplifiers with a higher gain potential, or using one more sensor so that the Dalek works over a 360° range as opposed to 180°. Due to constraints the team was unable to acquire quality components.

An alternative solution not attempted was to create a coordinate system, where the sensors detect the location of the signal, the Arduino processes this into an angular position and then rotates the motor until it reaches this angle, before returning to its starting orientation, a complex system which would have required lots of testing to get working properly.

A further extension would be to implement a system to prevent wire twisting such as a commutator ring which would allow for brushes to interface moving and stationary circuits. The team came up with this idea too late into production to change the wiring configuration.

It would also be preferable to use multiple motors in unison. For example the three planetary gears could each be fit with motors instead of the sun gear. Due to the high current

draw the team was not sufficiently equipped to handle that much current draw safely. A further addition of linear movement as well as the rotation could allow this design to be implemented in further searches, such as finding lost items or determining the source of a hazard.

The improvement of sensor and op amp quality would allow us to sense, in a wider array of harsher lighting conditions and increase the reliability of the sensors, specifically during a sunny day when the maximum amount of interference is received. A commutator system eliminates the wire bending issue which allows for 360° of freedom of the sensor platform. Having three independent motors driving the rotating platform would have tripled the movable load, this would also increase the overall top speed of the platform. Thus reducing the time to eliminate the enemies. A coordinate based system to sense this would be more efficient than the current solution and would decrease wire entanglement as it always automatically untwists.

7.2.2. Applications of this Design

This design solution has many applications for other solutions. For example, some robots use IR sensing to locate a charging base, for that, a design similar to the one shown here could be used to pinpoint the location of the charging base.

7.3. Common Problems and Related Solutions

Common problems encountered during the design are presented here, along with solutions found or the lack thereof.

7.3.1. H-bridge

The H-bridge repeatedly overheated due to the low current it is meant to supply. Possible solutions are to construct the H-bridge with powerful transistors, use a higher-rated H-bridge, add heat sinks, or take advantage of external circuitry that can control the current, such as voltage regulators and diodes.

7.3.2. Pulse Code Modulation and Speaker

Getting the speaker loud enough was often difficult with Arduino requiring the use of an amplifying system for the design; a power transistor was used here.

Speakers with integrated amplification systems often break, so a custom one was created for more control by adjusting the component values. This acts as a good safety system, as it prevents the destruction of the speaker if too much current were to be supplied, instead of destroying the transistor. The code's logic had to be inverted in order for the speaker to signal the exterminate sequence at 3V, as shown in figure 11. This occurred because the code is called in the *loop* function, so therefore the speaker audio is repeatedly started when the voltage is below 3V resulting in a humming noise being emitted. However, when the voltage goes above the 3V threshold, the audio will stop being called and the speaker plays out the audio from the last time it was called. A simple solution is to add a delay after the audio is called. For the design, the team implemented delays through calling the LEDs to flash throughout the audio.

7.3.3. Filters

Filter resonance would occur frequently, due to multiple factors: op amps having a low, but still noticeable amount of self-feedback; noise created by the circuit; and the circuit itself providing self-feedback. If these issues are ignored, the filter would output a continuous sinusoidal wave. This issue can be dealt with by placing buffer voltage followers between each filter, preventing some of the self-feedback from the op amp, and stopping the resonance issue.

Gain Bandwidth Product (GBW) became an issue due to the requirements on the op amp from the filter. Some original filter designs required high GBW, which could not be provided by either the LM741 or LM328. Instead, multiple low-GBW filters were used, which required a GBW that the op amps could provide. This however meant that the amplification of each filter was low, so multiple were required to achieve a high enough peak-to-peak voltage that could be detected by the Ardiuno.

The conversion from a sinusoidal wave, or similar, to a constant voltage required a peak detector to be designed with a specific frequency in mind; the capacitor values in a peak detector depend on the frequency. Additionally, peak detectors are not ideal: outputting a voltage equivalent to the average voltage across the sinusoidal wave, resulting in a lower output voltage than expected; and outputting only a semi-constant voltage, with minor peaks. These flaws must be taken into consideration when creating the Arduino code.

7.3.4. Arduino

The Arduino often overheated when battery powered through the V_{in} pin. One could power the Arduino with an alkaline 9V battery through the power jack. To maintain the use of the V_{in} pin, it would be necessary to not use the 5V or 3.3V pin from the Arduino as research and the team's own experimentation shows that this overheating originates from their use.

Furthermore, the code was not running at the right time. The Arduino is single threaded so it processes one task at a time, requiring the strategic ordering of commands. For example, if audio is played by looping through the notes, then the Arduino will loop through all of these notes before moving onto the next command. Another alternative is to use the *millis* function.

7.3.5. IR Sensors

The phototransistor had a maximum current of mA, however the change in current when detecting the enemy was in μA. This led to a design choice to use a high-ohm resistor, which would allow us longer range. However, this meant that outdoors with direct sunlight, the IR sensor would almost immediately saturate. This was dealt with by creating cones which protected the IR sensors from the majority of direct sunlight.

The photo-transistor would only work in a certain range and radius, approximately 60° both sides from the centre. This can be dealt with by precisely placing the photo-transistors in locations where they are still active, however have a large enough view angle such that the final design would still function.

7.3.6. Motors

The motors are difficult to start but easy to easily spin once started. The motors require a large amount of initial current to start, this needs to be factor into the design if you desire speed control, that is, for slow rotation, the power input would need to start high and then decrease to lower level for the slow rotation. This can be dealt with using pulses, which rotates the Dalek slowly but maintains a high degree of control.

8. Conclusion

This report outlined the Dalek task proposed, the challenges involved and the solutions found. The design was shown to be a working concept however, it does require some more work to integrate all of the subsystems adequately in an unstressed environment. However, there is complete confidence in the design's ability to work as a combined system when the time is available to properly construct the design since many failures came not from the concept or logic but from external factors such as perfboard short circuits.

Reporting this design has made clear the importance of using both analogue and digital systems to optimise and simplify systems. It has also emphasised the importance of incremental design. If the full design had been immediately integrated, it will be assumed at first that it did not work, however it is known that the subsystems work due to the incremental design process. Further, the moment of deviation from incremental design, where everything was soldered onto perfboard, was when the design began to falter. A final lesson, connected to incremental design, is to allow a basic system to work before moving onto more difficult but effective designs. In the beginning stage this rule was followed, as demonstrated in the construction of the H-bridge from transistors, before moving through packed H-bridges, resulting in an effective motor controlling system. However, flaws started to show in the design when optimising with soldering was skipped before ensuring that everything in the design was to the highest possible standard. This was especially evident when compared to the successful groups in the competitive phase, having their circuits still built upon breadboards.

Overall, the Dalek project served as a valuable learning lesson in communication early, teamwork and navigating time pressures efficiently.

9. References

[1]

- **Note: Access to this resource is restricted to members of the University of New South Wales**

D. Taubman, "DESN1000 EE&T Stream 2024 Term 3: Dalek Target," PDF, David Taubman, 2024. Available: https://moodle.telt.unsw.edu.au/mod/folder/view.php?id=7137030

[2]

P. Denisowski, *Webinar: An Introduction to Direction Finding*, (Feb. 21, 2020). Accessed: Nov. 23, 2024. [Online Video]. Available: https://www.rohde-schwarz.com/au/knowledge-center/videos/webinar-an-introduction-to-direction-finding-video-detailpage_251220-761216.html

[3]

Rohde & Schwarz (Australia) Pty Ltd, "Technology Fundamentals: Fundamentals of direction finding and radiolocation." [Online]. Available: https://www.rohde-schwarz.com/au/knowledge-center/technology-fundamentals/radio-direction-finding-techniques/radio-direction-finding-techniques_255557.html

[4]

M. Arpit, "High-Speed Op Amp Enables Infrared (IR) Proximity Sensing," *Analog Devices*, p. 3, Oct. 2009, [Online]. Available: https://www.analog.com/en/resources/design-notes/highspeed-op-amp-enables-infrared-ir-proximity-sensing.html

[5]

SHARP, “Holder-less Type IR Detecting Unit for Remote Control.” [Online]. Available: https://media.digikey.com/pdf/Data%20Sheets/Sharp%20PDFs/GP1UX51QS.pdf

[6]
A. Liu and D. Su, “PAGE: 1 OF 5 APPROVED: WYNEC CHECKED: PHOTOTRANSISTOR Part Number: L-53P3C,” 2012. Accessed: Nov. 23, 2024. [Online]. Available: https://www.farnell.com/datasheets/1683594.pdf

[7]
D. Mellis, “PCM Code,” *Arduinolibraries.info*, Apr. 16, 2017. Available: https://www.arduinolibraries.info/libraries/pcm

[8]
“Over 3,318,424 audio sound MP3 clips to play and download,” *101soundboards.com — Have Fun Playing Sound Clips*, 2018 Available: https://www.101soundboards.com/

[9]
Element 14 Incorporated, “Miniature Motor” Jun. 23, 2022. Available: https://www.farnell.com/datasheets/3747216.pdf

[10]
SHARP, “Holder-less Type IR Detecting Unit for Remote Control.” [Online]. Available: https://media.digikey.com/pdf/Data%20Sheets/Sharp%20PDFs/GP1UX51QS.pdf

[11]
C. Reinke, “Spectroid,” Google Play. Accessed: Nov. 23, 2024. [Online]. Available: https://play.google.com/store/apps/details?id=org.intoorbit.spectrum&hl=en_AU

[12]
M. Arpit, “High-Speed Op Amp Enables Infrared (IR) Proximity Sensing,” *Analog Devices*, p. 3, Oct. 2009, [Online]. Available: https://www.analog.com/en/resources/design-notes/highspeed-op-amp-enables-infrared-ir-proximity-sensing.html

[13]

"millis()," Arduino Documentation. [Online]. Available: https://docs.Arduino.cc/language-reference/en/functions/time/millis/

[14]
R. Astley, "Rick Astley - Never Gonna Give You up (Video)," *YouTube*. Oct. 25, 2009. [YouTube Video]. Available: https://www.youtube.com/watch?v=dQw4w9WgXcQ

[15]
Texas Instruments Incorporated, "LM741 Operational Amplifier." Jan. 10, 2015. Available: https://www.ti.com/lit/ds/symlink/lm741.pdf

[16]
Texas Instruments Incorporated, "LM148/LM248/LM348 Quad 741 Op Amps." Jan. 03, 2013. Available: https://www.ti.com/lit/ds/symlink/lm348.pdf?ts=1727675571603&ref_url=https%253A%252F%252Fau.mouser.com%252F

Appendix A

Acronyms

DF = Direction finding

AoA = Angle of Arrival

ADC = Analogue to Digital converter

Arduino = Arduino Uno

PCM = Pulse Code Modulation

PWM = Pulse Width Modulation

IC = Integrated chip

IR = Infrared

BPF = Band-pass Filter

Appendix B

Bill of Materials (BoM) for the circuit used:

Qty	Part	Qty	Part
2	Arduino Uno	1	LM7812 (Voltage Regulator, 12V, 3A)
2	BD139 (Transistor NPN/PNP)	2	Red LED
2	AS3000 (57mm speaker)	2	White LED
4	LM741 (Op Amp)	8	D-size Battery
2	LM348 (Quad Op Amp)	1	ST0506 (Switch)
2	1N4148 (Diode)	2	PH9222 (Battery Holder)
1	MM28 (Motor)	2	L-53P3C (IR Photo-transistor)
1	L298N (Motor Driver)	1	LM7805 (Voltage Regulator 5V, 1A)