FINAL PRODUCT DESIGN REPORT

Pinky & The Brains

Abstract

This report details the different stages of designing and implementing a functioning micromouse that can solve a maze.

Table of Contents

Membership Lists

Group Members

- Aaish Bakhtiar
- Dawood Shafique
- Muhammad Malik Tiwana
- Mohammad Abedin
- Asif Chowdhury

Contributions Table

This table shows the sections each group member was assigned and which section each member also completed or helped completing.

Group Member	Sections Assigned	Sections Completed/*Assisted with*
Aaish Bakhtiar	- Background (1.0 – 1.9) - Hardware Design (4.3, 4.4) - Testing (7.0)	- Background (1.0 – 1.9) - Hardware Design (4.3, 4.4) - Testing (7.0) - *Requirements (2.0 – 2.3)* - *Conclusion*
Dawood Shafique	- Introduction - Requirements (2.0 – 2.3) - Hardware Design (4.1) - Conclusion	- Introduction - Requirements (2.0 – 2.3) - Hardware Design (4.1) - Conclusion
Muhammad Malik Tiwana	- Abstract - Requirements (2.0 – 2.3) - Specification (3.0) - Hardware Design (4.5)	- Abstract - Requirements (2.0 – 2.3) - Specification (3.0) - Hardware Design (4.5)
Mohammad Abedin	- Hardware Design (4.2 – 4.4) - Software Design & Logic (5.0 – 5.2) - Implementation of Hardware (6.0)	- Hardware Design (4.2 – 4.4) - Software Design & Logic (5.0 – 5.2) - Implementation of Hardware (6.0)
Asif Chowdhury	- Hardware Design (4.2) - Software Design & Logic (5.0 – 5.2) - Implementation of Hardware (6.0)	- Hardware Design (4.2) - Software Design & Logic (5.0 – 5.2) - Implementation of Hardware (6.0)

Abstract

This report discusses the requirements, design, and testing strategy of a micro-mouse robot. The report is final detailing all critical detail of what our group solution to the micro-mouse maze. It details all final conclusions of the critical stages of the mouse. Due to the current climate controls with the global pandemic having labs been unable to open, the only way in which we could design and test the micro-mouse was through an online virtual environment. The virtual environment we could create a virtual replication of the IEEE maze in which we can place the mouse in for it to operate around. In this scenario we would have been able to test the functionality of the mouse, see how it manoeuvred itself around the maze.

Introduction

In the following report, we will discuss at the very highest level of what this report contains.
This report has 7 different sections of being Background, Requirements, Specification, Hardware Design, Software Design & Logic, Implementation of Hardware and Testing.
There will also be another 3 section of which are the reports Conclusion, Appendix A: Schematics and Appendix B: Coding.
To break down what will be seen in the document, we will summarise a little bit about each sections and subsections of the document to get a knowledge of what should be seen of our micro-mouse.

What the micro-mouse problem is?
The aim is to travel from the start to the target in a short time possible. The mouse has 10 minutes in which any trip from the start to the target will be eligible for measurement. The mouse shall operate within the IEEE maze. This is a grid of 256 cells laid out in a 16x16 arrangement. The centre 4 squares are the target, and the mouse starts in one corner (nominally the south-west corner, facing north).

In **section one** of the background, we will be discussing the background theory of all the devices, algorithms and platform that was used by Pinky & the brains. To help aide the breakdown of the background there are the following 9 subsections which will provide a fully covered background of our intended micro-mouse. The 9 subsections are Motor theory, stepper motor types & control, sensors & electromagnetic waves, noise & interference, navigation, mapping and path analysis, batteries & load, driving large currents and the MBED platform.

In **section two** of the requirements, we will be discussing the operating environment of the micro-mouse will be operating in such as in a well-lit room, a room with limited light or in the maze with dimensions. In this requirements section, we will be also discussing the micro-mouse maze rules in accordance with the IEEE maze as well as the mechanical requirements of the micro-mouse such as maximum velocity, required acceleration etc…
These discussions will be seen in the 3 subsections – Operating Environment, Functional requirements, and Mechanical requirements.

In **section three** of the specification, we will be discussing the selected components used in our mouse and the power/weight/monetary budgets. In this section we will be discussing how the components specification met the system requirements.

In **section four** of the hardware design, we will be discussing the dimensions, materials, layout, mounting positions & shape of the chassis. We will detail where the sensors will be mounted on the mouse as well as where the batteries, circuitry, motors, and wheels will be positioned on the micro-mouse. With the aid of schematics, this section will show where all components will be on the mouse with detailed explanations of how the circuitry was designed and how it meets the functional requirements of the micro-

mouse. Furthermore, an explanation of the configuration of the sensor circuitry as well as any mechanical/hardware solutions used to reduce problems e.g., noise/interference
Five subsections will be used which are chassis dimensions, layout & design, power distribution, motor drivers, sensor configuration and circuits and the power source. Subsections will aid the breaking down of this section to help support this discussion.

In **section five** of the software design & logic, we will be discussing the implementation of the software onto the micro-mouse. Flowcharts, short sections of code and block diagrams to demonstrate these implementations. 5 subsections which will help the design & logic of these are overall system discussion, software structure, sensor logic used for navigation, post processing of sensory information and mapping and path analysis solutions.

In **section six** of the implementation of hardware, we will be discussing how the circuitry is laid out on the veroboard. We will show how the circuit is laid out and why it was laid out in a particular way.

In **section seven** of the testing, we will be discussing how the micro-mouse allows for easy testing of internal states and sensor readings. To show this we need to be able to program the mouse itself, so it be able to track these readings with the use of test vectors. We can set these test vectors to test the mouse in all kinds of conditions. A better tested mouse makes it a better operating mouse.

1 – Background

1.1 - Motor Theory

The motion created by the DC electric motors is the result of a conductor carrying current and it being placed on a magnetic field, from this a force known as the Lorenz Force is created at right angles to both the magnetic field flux and the flow of current. This Lorenz Force pushes the conductor downwards.
The motors also cause rotary motion, and it is done by placing a simple one-turn coil or loop of wire in a magnetic field. As the one side of the coil in the middle is forced down, the opposing side is forced up because the current through the top of the coil is flowing in the direction opposite to that of the bottom side of the coil; **See Figure 1.**

Reversing the polarity of the current through the coil at the moment when both sides of the coil reach their highest and lowest points at the same time, reverses the Lorenz Forces in both segments and this then causes the coil to keep on rotating in a full circle. This is how rotary motion is achieved. [1]

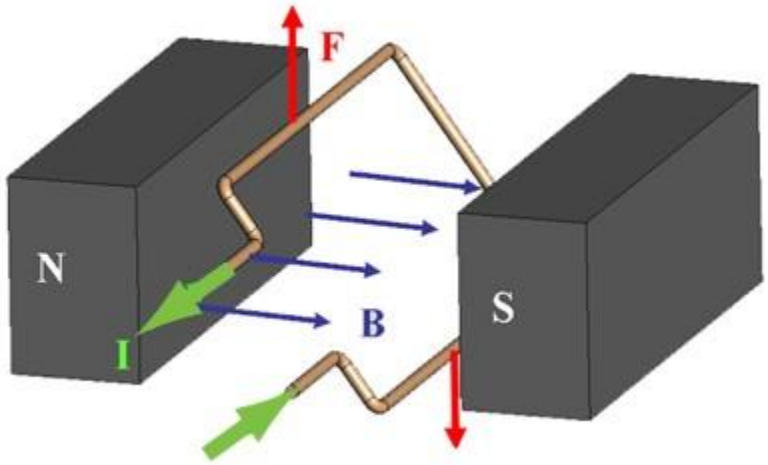


Figure 1

Motors Purpose & Function -
The movement of the micromouse is more known as its drivetrain. The motors and motor controllers are the key part to the drivetrain of our micromouse.
The motors and motor controllers make up the most important part of the micromouse drive train. There are many varieties of motors that are available for different uses, but we can focus on the 3 common motors used in a micromouse. These are servo motors, dc motors and stepper motors. These 3 motors are the most used motors because they are suited for robotics applications. They are easy to use and come in small sizes that can be attached onto robots like a micromouse. They are also fitted with their own small batteries that makes it easier for the engineer to power the motor. [2]

The process and thought behind choosing a motor for the micromouse is quite hard as the motor will have a big impact on the other aspects of the micromouse. Most notably, motors make up the largest part of the micromouse, and their power requirements will then dictate the design specifications for the power system of the micromouse.

1.2 - Stepper Motor Types & Control

There are 3 basic types of DC electric motors commonly used in micromouse projects.

Servo Motor:
Servo motors are closed-loop devices consisting of a continuous DC motor, a gear box, and a motor controller assembly housed inside a plastic outer casing. The built-in circuitry lets the user have better control and positioning of the motor. Servo motors are great as they come in a lot of different varieties and are easy to configure, but they come with the cost of having a lower weight capability, meaning the motor can only handle lighter micromouse and this then restricts the users to select the rest of the components of the micromouse with care knowing they cannot make the micromouse too heavy.

Continuous DC Motor:
A continuous DC motor is known as a permanent magnet DC motor. It has a stator in it which is an arrangement of two permanent magnets that provide a magnetic field in which the coil rotates. The coil or more commonly known as the armature, positioned in the centre of the motor, has an odd number of poles that have windings connected to a contact pad on the centre shaft known as the commutator. There are also brushes that provide power to the windings of the armature so that they are alternately attracted to and repelled from the magnets of the stator, causing the Lorenz Force to be applied on the coils and propelling it, creating torque to be transmitted through the shaft and therefore causing the armature to turn in a circular motion continuously. This type of motor is commonly used and commonly available, but they are quite expensive compared to the other motors, they also are very powerful and require a gearbox to control their fast speed.
The continuous DC motor is a great powerful motor, but it may be too powerful for our small micromouse's needs, if the motor is too powerful, we may have difficulties figuring out the proper power needs and the right speed control. This motor also requires gearing, with a lot of other components needed for the micromouse, having an extra component for the motor itself will be quite a nuisance and it would be better if we did not need to add it. Servo motors are also a great type of motor as they contain built in circuitry with a DC motor and a gearbox all together in one unit. This is quite convenient but servo motors in general are quite weak and can only handle a certain amount of micromouse weight, this is not ideal as we would want a motor that can give us an adamant amount of torque to propel our micromouse forward and to finish the maze under a certain amount of time.

Stepper Motor –
A stepper motor is a brushless non-continuous DC motor. Stepper Motors aim to have a repeatable and robust movement and are capable of good incremental shaft rotation, this incremental stepping motion is what gave the motor its name. The stepper motor can hold their position and resist turning. Regarding the stepper motor it is not able to run at high speeds although it does have a high holding torque. The motor has an open loop communicator and rotates purely based on the stepping patterns.

Method of control of motors –
Stepper motors use electrical signals to create their output of mechanical movement. The current that passes through the stator coil creates a magnetic field and that's what creates motion in a rotor. So, this means that the modification of the current passing through the stator coil can make the rotor spin at different speeds. To do this we can go into the 3 different ways that we could make this happen; series resistance, gearboxes, and voltage regulation.

Series Resistance -
As said above, stepper motors run based on electrical input and as seen in **Figure 2** below, speed and current are directly proportional to each other.

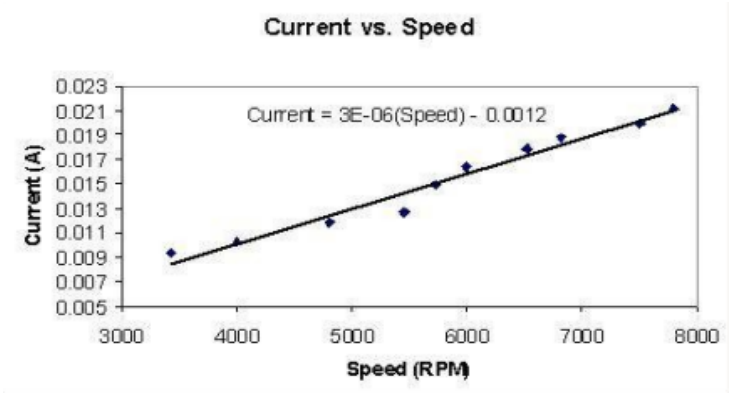


Figure 2

This shows that by modifying the resistance of the circuit therefore modifying the amount of current passing through a motor, we can control the speed that the rotor inside will turn with this simple method of controlling the current.
This method is a simple method but may not be practical as in a real micromouse, we would have multiple rotors and other components, having many resistors in the circuit may then cause current problems in the other components of the micromouse. So, if we were to use this method, we must take into consideration the other components in the micromouse.

Gearboxes –
We can also use external components like a gearbox to directly be able to control the speed of the motor. This method requires the stepper motor to connect through a gearbox. The gearboxes use the gear ratio formula to control the output speed of the stepper motor.
Gearboxes are a complex way to control the speed of the stepper motor, so we will not be taking it into consideration for this project, because gearboxes are quite large components in size and in power consumption and it is not practical enough to fit it onto our micromouse and have enough power to let it function in unison with the other components on the micromouse.

Voltage Regulation –
The most probable and practical way to control the speed of the stepper motor can be to use voltage regulators. There are many voltage regulators chips that exist that are built for the purpose of controlling the voltage. The voltage regulators use square waves to directly control the speed of a motor. This are very common to use to control the speed of a stepper motor, as they are small components that require small amounts of energy are quite cheap to buy.
[3]

1.3 - Sensors & Electromagnetic Waves

Sensors Purpose & Function -

Sensors are one of the most important components of the micromouse, it has a very important role of being the “eyes” of the micromouse robot. The sensors allow the micromouse to be able to read and detect the walls of the maze it is going to traverse in. The micromouse processes the information received from the sensors attached to it and reacts in a predetermined manner depending on the design of the control system. For this micromouse maze project, the micromouse needs to sense the surrounding walls to try map the maze and keep track of the different possible paths it could take. It's also necessary to help keep the micromouse aligned to the centre of a pathway in the maze.
There are different types of sensors that could be used for our micromouse

Infrared Sensors -

An infrared sensor consists of an infrared transmitter that sends out a beam of light into the maze which an infrared receiver then absorbs the same beam of light that is reflected to the sensor.

Ultrasonic/Acoustic Sensors -

Ultrasonic or Acoustic sensors use sound in addition to light to detect objects. It uses the speed of sound through air to process the objects nearby. The sonar sensor has an emitter transducer that projects high-frequency sound waves outside the hearing range of humans into the environment, this then bounces off the objects ahead, in this case the maze walls. And returns to a receiver transducer in the sensor. The time taken for the sound wave to return to the receiver is then used to determine the distance of the robot from the object. This works because the speed of sound traveling through the air is considerably slower than the speed it takes for the microprocessor circuits in the sensor to receive and process the information.

Mechanical/Touch Sensors -

These sensors use a switch located on the front or of the robot, and when the switch is engaged by it being pressed or bent by an object. The microcontroller processes the touch to the sensor and then responds accordingly.
Going through the advantages and disadvantages of the sensors will help us in deciding the best and most efficient sensor for our micromouse. The mechanical or touch sensors will be ruled out as it is not the most practical sensor in this case. The micromouse would need to detect a wall with enough time to then adjust and position correctly to avoid crashing into the wall, and with the touch sensors, this is obviously not possible.

How light sensors work -

Light sensors send out a beam of light that reflects back into the sensors receiver against the first object hit, the sensors then read the information like the distance of the object from the sensor and sends the information to the device it is connected to; in this case the micromouse.
The light that is sent out of the sensor one of the many rays that come under electromagnetic radiation. Radio waves, microwaves, infra-red, visible light, ultraviolet, x-rays are all examples of electromagnetic radiation. Optical sensors like photodiodes use the infra-red and visible wave lengths.

How infrared sensors work -

An infrared sensor measures and detects infrared radiation in its surrounding environment. In the case of an active infrared sensor, it emits out this infrared radiation and then detects is as it comes back to the sensor. The infrared light from the LED reflects off the object and is then detected by the receiver for it to process the information and send to the micromouse.
The sensors calculate the distance depending on what type of sensor it is.

Distance vs Proximity -
The infrared sensor does come in two different variations, distance, and proximity. The difference between them is that the IR proximity sensor detects if an object is within a predetermined distance from the micromouse, whereas IR distance sensors determine the actual distance between the object and the micromouse. Both sensors are great at detecting whether there is a wall near the micromouse or not however the distance sensor may have an edge because it uses an ADC as the receiver to calculate the distance between the walls.
The proximity sensor only detects how much distance the micromouse should keep from the object reflected. The proximity sensor may also have problems regarding the noise and it being a very sensitive sensor. This may then cause it to then also use more power in the entire micromouse. [4]
In the case of the infrared distance sensors, the distance to nearby objects is measured using triangulation. The emitter of the unit illuminates a small spot ahead with modulated infrared light. The light from the illuminated spot is focused by a special lens onto the detector element, and a triangle is then formed between the emitter, the spot on the maze wall that is illuminated and the detector. Depending on the distance between the sensor and the target surface, the angle of incidence of the reflected light will change, causing the light to strike a different point on the position sensitive detector. The distance of the object is then calculated by the location of the spot on the sensor. The distance is outputted as an analog voltage, which is then sent to the microcontroller for further processing.

1.4 - Noise & Interference

In a micromouse, there are a lot of different electrical components all grouped together to make the robot function correctly. All these electrical components distribute electrical noise throughout the power lines they are connected to when the current is switched on and off. This noise is the current flowing through each component and through the power lines of the whole circuit. Things like motors can create a great amount of electrical noise through the power lines of the circuit, especially when using a process like pulse width modulation to control the motor speed. This electrical noise created can cause noise-sensitive components like the microprocessors connected to the same power supply to reboot, have current spikes, or even be damaged because of this voltage and current spike that is created because of the many electrical components. Choosing the right battery component is crucial here to make sure that the circuit board connected to the big components like motors and sensors does not get damaged when they are operating. A good battery pack can make sure that all components are protected from large amounts of electrical noise, letting the micromouse components all be able to work in unison.
With the sensors that are being used in a micromouse, there are bound to be interference that can cause the results of the sensors to be inaccurate or wrong. Making sure that the n interference is kept to a minimum will greatly increase the functionality of the sensor and provide the micromouse the right information to switch to the right state and perform the right movement.
Things that can cause noise and interference are the ambient light that is in the room of the micromouse is running the maze in. Obviously, there are going to be lights working in the room that the micromouse will be operating in. These fluorescent lights will emit light rays to every part of the room and could interfere with the infrared light that is being emitted and received by the sensor. The sensor receiver would be flooded with light and cause it to take in light that is not useful to the sensor and cause it to give inaccurate readings. A way to help reduce this interference is choosing the right frequency on the sensors so that it only reads and takes in light reflected from the object, in this case the maze wall. We can reduce the amount of outside light sources from interfering with the infrared sensors.by controlling the frequency of the light beam that the transducer can process. If we use a low-pass filter that is equating to the same wavelength that the infrared sensor initially beams out, then any ambient light that is usually fluorescent light, on a higher frequency, will not be processed by the transducer and the sensors will then only pick up the light beams reflected off the walls of the maze.

1.5 - Navigation

Navigation of the micromouse is one of the most important part of the micromouse. The navigation is what will make the micromouse be able to traverse the maze, map it and be able to complete it in the fastest path possible. The way that the sensors are placed onto the micromouse chassis itself will determine how well it will move through the maze. Using the right number of sensors will help aid our micromouse in traversing the maze correctly with the most efficient amount of power used.
There are a lot of key issues that may occur if the sensors are not placed correctly. We need to make sure that the micromouse can traverse the maze as fast as possible, making quick efficient turns. This means that we will need to position our sensors well up to the front of our chassis so that it is able to detect an incoming wall and leave the micromouse sufficient time to be able to decide its state and change direction or stop accordingly. The micromouse will use the information received from the infrared sensors to decide whether it should be in a stop state, turn state or a drive state. If the sensor detects a wall in front of the micromouse, the micromouse should be in a stop state, the left and right sensors will then detect where the missing wall is and the micromouse will be in a turn state turning the direction of the empty wall that is chosen according to the maze solve algorithm. When the micromouse needs to make a turn, it will use the binary bits corresponding to its state of the sensors provided and the do the right movement needed to traverse the maze.
The micromouse will also need to stay in the middle of the corridors in the maze, this is where the sensors placed facing left and right of the micromouse will come into use. These distance cells will measure the distance between the wall and the micromouse and send the information to the microcontroller. The microcontroller will adjust the position of the micromouse if the distance between the sensor and the wall on the left and the distance between the sensor and the wall on the right is not somewhat similar. If the distance is not the same for both side of the micromouse then that means the micromouse is off centre. This will make the micromouse decide what state it should be in. If the distance between the right sensor and the right wall is too little; meaning it is too close to the right wall, the micromouse will turn into a state to gradually adjust and turn left to re-centre to the middle, and it is the same with the left sensor too.
To process the information that the sensors receive, the microprocessor receiving the information needs to have a bit and state table that lets it know what state to switch to depending on the sensor patterns. The front sensor on the micromouse will be the most important sensor as it will let the micromouse know about walls in front and when to stop. To make for efficient turns and stops from the micromouse, the front sensor must detect the wall Infront in enough time to allow the mouse to stop safely. This means that we must take all delays into account to guarantee a safe and correct stop from the micromouse. To do this we have to take into account the sensor sampling rate, the software in the microprocessors and the material of the tyres on the micromouse; meaning taking into account the inertia and skidding when the micromouse mouse tries to stop at a relatively high speed. Another way to make sure that the micromouse stops correctly is to make sure that the front sensor is not too close to the wheels and is quite far ahead of the chassis itself so that it can detect the wall a lot sooner than the micromouse reaches the wall.

1.6 - Mapping & Path Analysis

The principal goal and objective of the micromouse is to solve the maze and reach the end as quickly as possible. The micromouse uses maze-searching algorithm to perform this remember each cell of the maze and be able to calculate the fastest route possible.

There are many different maze-searching algorithms out there that are available to use and all of them use a different method and approach to complete the same objective, finish the maze as fast as possible. Researchers have been studying maze solving algorithms for many years, even since before micromouse were a thing. They have been researching the best method to solve any and every maze possible with the fastest and most efficient approach. This research has led to many computer-generated algorithms that work for other uses but are quite impractical for a micromouse to use. A more micromouse suited maze-solving algorithm would come under wall following algorithms.

Wall following algorithm is where the mouse chooses a left or right wall and always keeps which ever chosen wall on its side moving through the maze. A very simple algorithm that allows the mouse to traverse the whole maze and eventually finish it. This is one of the simplest algorithms, but it is ineffective in making the micromouse be able to finish the maze in the quickest time.

A more effective algorithm is the Flood-Fill or Lee's algorithm, his method meant we could program the micromouse to be able to go from start to finish whilst guaranteeing finding the shortest route. Flood-Fill/Lee's algorithm required only a map of the maze in which we don't have however we do know some properties of the maze such as the size of the maze, size of each square and the height and depth of walls just not their positions.

The flood-fill/Lee's algorithm uses a sophisticated system of distance and wall information to create the shortest path to the centre of the maze or a predetermined location. It assigns a value to each cell in the maze, each number represents the distance from that cell to the centre or the predetermined location and it stores these values in an array. While the numbers are being assigned, the micromouse also stores a wall map which is continually updating as it is traversing through the maze assigning numbers and form the sensors detecting new walls in the maze. The flood-fill algorithm is great as it always finds the shortest path to the centre of the maze or to the predetermined location.

Having this in mind we can use the Flood-Fill algorithm to aid the micromouse around the maze. Firstly, we will need something called square numbering. Square numbering is where we determine the start and target position of the maze, we can label the target position(s) as 0 and then the label surrounding squares of the target +1 each time we increase in distance. This will be repeated until the entire maze is numbered. We would also need to number the start square so that that micromouse acknowledges it has taken a path.

An example of a maze; **see Figure 3**, 'S' refers to the start position and '0' refers to the target position. As you can see in the fully labelled maze in the second diagram, each number represents the steps away from the target. As our micromouse has a stepper counter which essentially counts the steps it takes, it will proceed through the maze following the labelled squares using the shortest path it finds.

Flood fill is a lot better suited for the micromouse, and our objective for the maze. This is because the flood-fill algorithm always finds the shortest path to the centre of the maze, and even though the shortest path is not the quickest path due to the shortest path still may have a lot more turns, it is a lot better than the other algorithms that take a lot longer to map the maze and finish it.

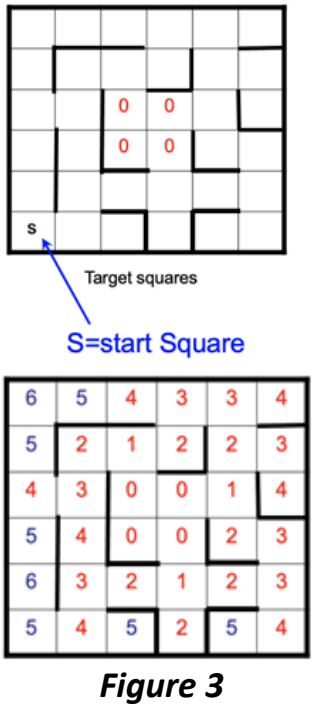


Figure 3

1.7 - Batteries & Load

The power system of a micromouse is the most important part of the whole structure, without power none of the attached components will be able to work to make the micromouse complete its intended functions. This means that the micromouse will have to have a considerable amount of energy stored and distributed to the different components for them to all work in unison for a certain period to traverse and solve the maze. The batteries must provide energy for a long enough time for the micromouse to solve the maze without having to replace the battery, allowing it to solve the maze in one run without any of the components failing. Batteries are the most practical option in this case to provide the micromouse components and circuit with power.

There are two main types of batteries that can be used in this project. Which is the following two: rechargeable Batteries and Alkaline batteries. Both are very different from each other even though they are a type of battery.

The rechargeable batteries do not require any type of replacement, heavier for the given Ah and they are cheaper if the batteries will be reused, but on the flip side the alkaline batteries do require replacement, lighter for the given Ah and these batteries are cheaper per unit if compared to the Rechargeable batteries. There are also other battery types like zinc and lithium batteries, but they contain chemicals that may be too dangerous for the micromouse project if any malfunction were to occur.

We needed to choose a battery that meets the power needs of the components on the micromouse and one that also meets the limitations of the chassis strength and area. We needed a strong battery that was able to fit on our micromouse. This battery would also need to be able to last for a certain amount of time, in this case from 10 minutes to 30 minutes in 1 run. Considering the test runs and such, this battery would also have to be rechargeable.

The alkaline batteries life expectancy is 300 to 800 percent higher than that of zinc batteries and the only disadvantage they have is that they are unable to deliver high currents. A solution to that could be to use bigger d sized cells batter to be able to handle the increased load of the many components on the micromouse.

1.8 - The MBED Platform

The MBED platform is made up of many microcontrollers that all have different capacities and functions. MBED provides software tools for creating microcontroller firmware. MBED is very useful and essential in the case of building a micromouse robot. They have the hardware and software components needed to allow the micromouse to be able to make the right decision at the right time. These come in the form of hardware boards that contain the microcontroller and other components like RAM, ethernet, USB device and many input/outputs. A MBED HDK Architecture diagram shows how all these components are connected on the board; **See Figure 4.**
It is a rapid prototyping platform for Internet of Things, it defines hardware design kit; with minimum specifications and pin connectivity, it also defines software design kits; APIs, hardware controlling drivers, small operating systems, and real-time support.
You can use the online compiler and simulator on the MBED website to create a code for the hardware you want to control and be able to simulate it to see the output. This is the main source of coding software we will use to control all the software aspects of our micromouse, this includes the sensors configuration, mapping algorithm and movement of the micromouse.

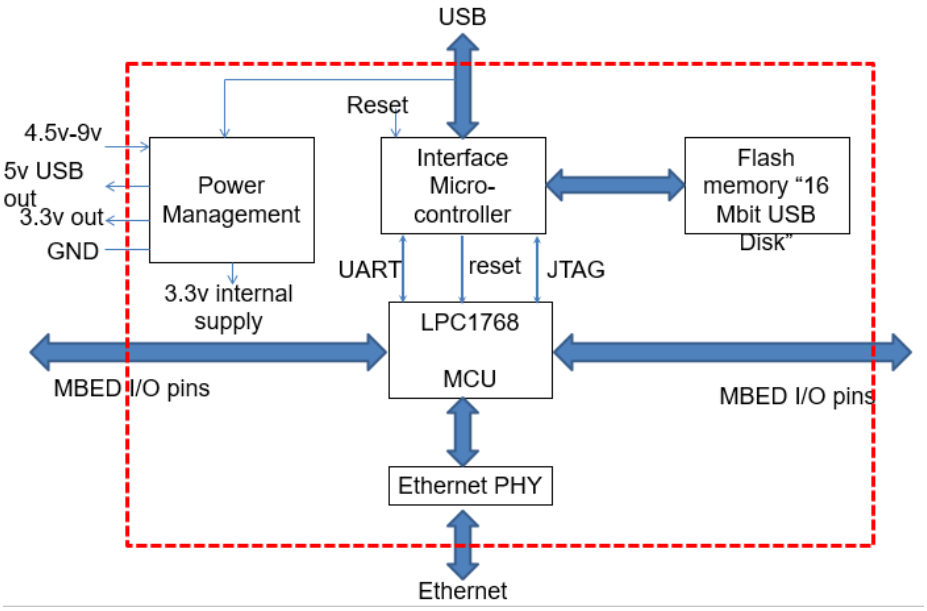


Figure 4

2 - Requirements

2.1 – Operating Environment

Maze Specification –

Once finalising the micro-mouse development, we intend to operate the mouse in the set environment set out by the IEEE. By this the micro-mouse will be operating in the 16 x 16 maze, made up of 256 cells where the mouse will start at one corner and find its path towards the centre made up of 4 squares. What we know about the maze is that each square is 18 x 18 cm, the wall positioning is unknown, the possible junctions the mouse could encounter are T-junction/junction/left turn/right turn/dead end, the walls are approximately 5cm in height and approximately 12mm in width, the top of the walls are coloured red, the side of the walls are coloured white, and the floor is coloured matt black.

The floor of the maze will be on a flat table, to help negate any factors of uneven surfaces slowing down the micromouse while it travels through the maze. It will also make sure that the micromouse does not turn over or change course due to an uneven surface. The walls are painted white so the sensors rays will be reflected from a white surface. This is good to know for testing so, when we do test our sensors, we can test against a white surface to get the most accurate results. The floor will be matt black, and this also helps with reducing the amount of ambient light in the maze, the matt black will reflect less light upwards and in turn reduce the amount of ambient light interfering with the sensors.

To conform with the IEEE maze rules, we had to ensure the micro-mouse has no prior knowledge of the maze except what was mentioned in the paragraph of above. The mouse doesn't necessarily have to complete the maze within "first attempt" but to attempt to complete the maze within 10 minutes. The "mouse" is defined as device which contains a chassis, motor(s) for motion, a steering method, sensors to detect the presence or absence of walls, control logic, batteries and designers and builders. [5]

Room Specification -

The room in which the micromouse shall be tested and ran in should be a room with controlled factors of illumination, temperature, humidly and wind. We will need to operate the micromouse in a room with the windows closed to make sure that no wind or outside debris has any way of disrupting the micromouse. To prevent any sunlight from interfering with the sensor rays, we will close the curtains or blinds to not allow for any sunlight to enter the room. The room will still have lights operating, so we will have to take this into consideration when designing the sensor layout. One way we could help prevent the ambient light in the room from interfering with the sensor rays is to use light shields to cover the front of the sensor so that any ambient light is not disturbing the sensor rays.

2.2 – Functional requirements

The functional requirements of the micromouse are important to know, so that we can create a micromouse with all the different factors considered before going ahead and creating it.

Sensors Requirements –

Range -

We need to consider the range of sensors we need to use for our micromouse. We need to use a range that allows the sensor to be able to detect a wall in ample amount of time to be able to communicate with the micromouse to perform the right action. Since each cell of the maze is 18cm and our micromouse chassis is 12-14cm with the wheels, meaning that our sensor will have to have a minimum required range of 4cm. This is great as most infrared sensors come from ranges between 10cm and 80cm.

Sampling Rate –

Different sampling rates can cause different outputs. *"Sampling as fast as possible will allow us to obtain great accuracy or sampling as slow as possible will conserve processor time."* [6]. We have decided that accuracy will be key to making sure that our micromouse can perform its action in the quickest time and in enough time so that it can move through the maze as efficiently and quickly as possible. A sampling rate of 5ms is fast enough that we will get accurate and precise information from our sensor and fast enough to let our micromouse decide on its next action.

Communication from sensor to robot –

The sensor should detect a wall ahead of the micromouse and let the micromouse know the distance between the wall and the micromouse. The micromouse processer can then choose the right action to let the micromouse know what manoeuvre to do next. If there is a wall in front of the micromouse and no walls to the right or left, based on the algorithm that the micromouse is using, the micromouse stop a few cm from the wall and either turn left or right to continue traversing the maze. The sensors detecting landmarks like the corners of walls is also essential in helping to position the micromouse correctly when rotating or turning to a new direction. The micromouse can use the position of the landmarks given by the sensors to correctly position to rotate or turn to continue completing the maze.

2.3 – Mechanical requirements

Maze Navigation –

The micromouse itself must be able to navigate the maze by itself using the algorithm. It must use the algorithm to map the maze as it goes along each cell and be able to perform an analysis on the shortest possible path of the maze. We must make sure that the hardware and software components like the sensors and mapping algorithm are able to communicate with each other effectively so that the micromouse always knows what its current position is, and what position it must take next to complete the maze in the shortest time possible. It must be able to detect the maze walls and map it and be able to change between the moving forward, stop, rotate 90o left, rotate 90o right states correctly.

Running time –

The micromouse will have to complete the maze within 10 - 30 minutes for it to comply with the IEEE maze regulations.

We also need to consider the battery options for the micromouse to make sure that it will be able to run for 10 – 30 minutes plus have enough capacity to let the micromouse run for test runs. This means that a battery with a large capacity and reachability capability would be optimal for our micromouse.

3 – Specification

In this section, we will list all the components we have chosen and will need for our micromouse. This section will discuss the reasons for the choice of those components and at the end there will also be a final costing table, listing all the components description and price. It will also include a final weighting of the micromouse table listing the full weight of all the components and our final micromouse.

Choice of motor for our micromouse -

We must make sure that the motor we choose for our micromouse can give us the right power and precision to finish the maze in the most efficient way possible. Therefore, the DC stepper motor is the motor we have chosen to use. The stepper motor has the great advantage of precise movement control over the other two motors. The incremental stepping motion allows the motor to control the micromouse in a very detailed manner. The fact that steppers can move in very small incremental steps, as well as hold their position and resist turning, makes them the perfect motor choice for our micromouse project. The only disadvantage of the stepper motor is their complex control requirements. However, this disadvantage can be easily overcome by using the numerous stepper motor controllers that are widely available for us to use.

The motor we are going to use is a unipolar stepper motor, the main reason for this is because with this specific version of the motor it has a simple communication circuit this is great because the motor can communicate with the other components easily so that the commands its receiving will be completed. To differentiate between a bipolar and unipolar motor depends on the number of wires the motor has. Bipolar motors are better than unipolar motors in the sense that they have more torque and are more efficient, but they are more complicated to drive and attach to the micromouse because they need reverse current.

We will be using the Astrosyn Stepper motor for our micromouse. This micromouse is great as it has high torque, with reduced noise and it comes with either unipolar or bipolar modes which is great as it keeps the option flexible for us to change between the two depending on which is better for us.
We will also need to use a smaller motor for the front wheel as if we use the same Astrosyn motor, it will require a greater amount of energy to allow all the rest of the components to work. For this reason, we will be using a small RS PRO dc motor that only requires 1.21 Watts of energy and between 1.5V to 3V of power as the motor for our front wheel.

Choice of Sensor for our Micromouse -
Deciding on which sensor to use on the micromouse will be vital, as the sensor used needs to be power efficient and good at directing the micromouse in the right direction through the maze without it colliding with the maze walls. A good sensor is needed for the micromouse to be able to be guided through the maze from the information received from the sensor.

Going through the advantages and disadvantages of the sensors will help us in deciding the best and most efficient sensor for our micromouse. The mechanical or touch sensors will be ruled out as it is not the most practical sensor in this case. The micromouse would need to detect a wall with enough time to then adjust and position correctly to avoid crashing into the wall, and with the touch sensors, this is obviously not possible. This leaves us with the choice between infrared sensors and ultrasonic sensors. The advantage of ultrasonic sensors is that sound is not sensitive to the different colours of objects and the light reflecting properties of objects. On the other hand, certain materials do reflect sound better than others, while some absorb most of the sound. The disadvantage of ultrasonic sensors is that they are susceptible to any type of electrical noise interference in the power circuit, while the motor and the micromouse is running, the noise made from the micromouse may interfere with the ultrasonic sensor receiver transducer, hence altering the information received. This could cause transducer ringing, where the transmitter transducer may receive vibrations not from the objects intended to get vibrations from and cause a false trigger on the receiver transducer.

The problem that is most common with infrared sensors is the ambient light or outside light that is active outside the maze. The infrared receiver is sensitive to any type of light rays and it may receive stray light rays causing the sensor to send the wrong information to the micromouse. This can be resolved by changing the frequency at which the infrared light is modulated, this helps to then avoid any other distracting effects from outside light sources not from the sensor itself. Another plus side to the infrared sensor is that it cannot be interfered by electromagnetic waves unlike the ultrasonic sensors.
We want to choose a sensor that is affordable, practical, and efficient for our micromouse. A sensor that has the least number of problems that could occur during the maze run. We have concluded that the infrared sensor is the best sensor to use for the micromouse. To further add to this, it would be best that the micromouse receives the distance between the object and the micromouse, so it knows when to stop and adjust position, so in this case, an infrared distance sensor is the best option here for our micromouse.

The GP2Y0A41YSK0F is a distance measuring sensor unit. This is the exact sensor we will be using on our micromouse. Composed of an integrated combination of PSD, IRED and signal processing circuit. The variety of the reflectivity of the object, the environmental temperature and the operating duration are not influenced easily to the distance detection because of adopting the triangulation method.

Choice of MBED board for our micromouse:
The most used MBED board for a micromouse project like this is the LPC1768 board. This board is great as it contains all the pins and components we need inside. This board is commonly used because it is very easy to code MBED code into and apply it to real life hardware like the micromouse.

Costing Table –

Part Number	Description	Company/Catalogue	Unit Cost	Quantity	Total Cost
1	Y129 Astrosyn motor	Farnell	£19.95	2	£39.9
2	RS PRO Motor	RS PRO	£1.65	1	£3.3
3	GP2Y0A41SKOF SHARP distance sensor	SHARP	£4.32	3	£12.96
4	L297 ST Stepper Motor Controllers	ST. life augmented	£6.21	3	£18.63
5	MBED NXP LPC1768 module	Mbed/ Amazon	£55.00	1	£55.00
6	LM78XX / LM78XXA Volage Regulator	Fairchild	£0.46	1	£0.46
7	9V Alkaline Battery	Duracell	£4.99	2	£9.98
8	Wheels	TEM Electronic Components	£4.68, 3+ = £4.12	4	£4.12
9	Aluminium Mounting Bracket	RS	£2.44	3	£7.32
10	Circuit Board	Amazon	£0.809	10	£8.09
11	Standoff pillars	Essentra Components	£0.26	4	£1.04
				Total Cost:	**£147.54**

Weighting of Micromouse –

Part Number	Description	Unit Weight (g)	Quantity in final mouse	Total Weight (g)
1	Y129 Astrosyn motor	220g	2	440g
2	RS Pro motor	20g	1	20g
3	GP2Y0A41SKOF SHARP distance sensor	3.6g	3	10.8g
4	L297 ST Stepper Motor Controllers	1g-20g	3	3g-60g
5	MBED NXP LPC1768 module	1g-20g	1	1g-20g
6	Duracell Alkaline 9V Battery	45g	1	45g
7	Wheels	116 g	3	348g
8	Circuit Board	70g	1	70g
9	Miscellaneous components (give or take)	100g	1	100g
			Total Weight (g):	1037.8g – 1113.8g

4 - Hardware Design

This section will list all the details of the dimensions, layout, and design of the hardware components that we are going to use for our micromouse.

4.1 – Chassis dimensions, layout & design

Wheel mounting –

We will be using rubber wheels for the development of the micro-mouse. The wheels; **See Figure 5** , will be 80mm in diameter and 20mm in width. The wheels are originally 30mm in width according to the wheel's manufacturer specifications however we will order the same wheels however just 10mm smaller to fit with the total width of our planned micro-mouse. The wheels are mechanically pushed into the wheels axle of the shaft which fed onto the motors enabling movement. The wheels weigh 116g and cost £4.12 for 3+ wheels which is suitable for our requirements because we will only need 3 wheels. This means these wheels are lightweight as well as cheap. [7]

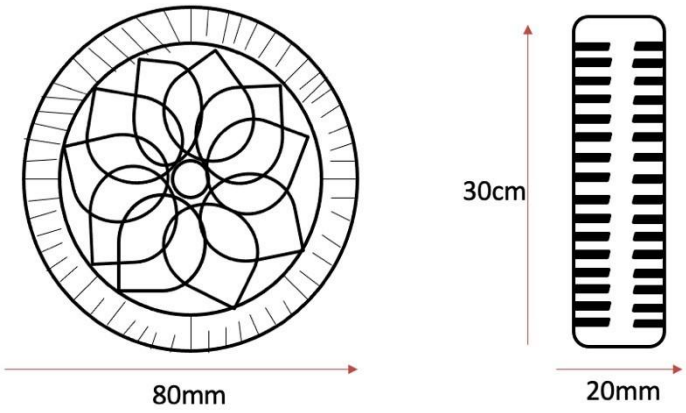


Figure 5

Fixing of Motors –

Firstly, we need to get our wheels, mounting brackets and motors for it to all interconnect to one another. The wheels will be connected to the motor with the use of the mounting brackets, the mounting brackets will be placed over the motor as such that no wires are damaged in the process. As we have three motors for three wheels, we do this process three times. Once attached, the mounting brackets can now be connected to the chassis.
We think the best mounting bracket will be one made from aluminium. As the stepper motor generates a lot of heat, we will need the brackets to be made from a material which we conduct heat efficiently therefore we will prefer the use of aluminium brackets to steel; **See Figure 6.** [8] [9]

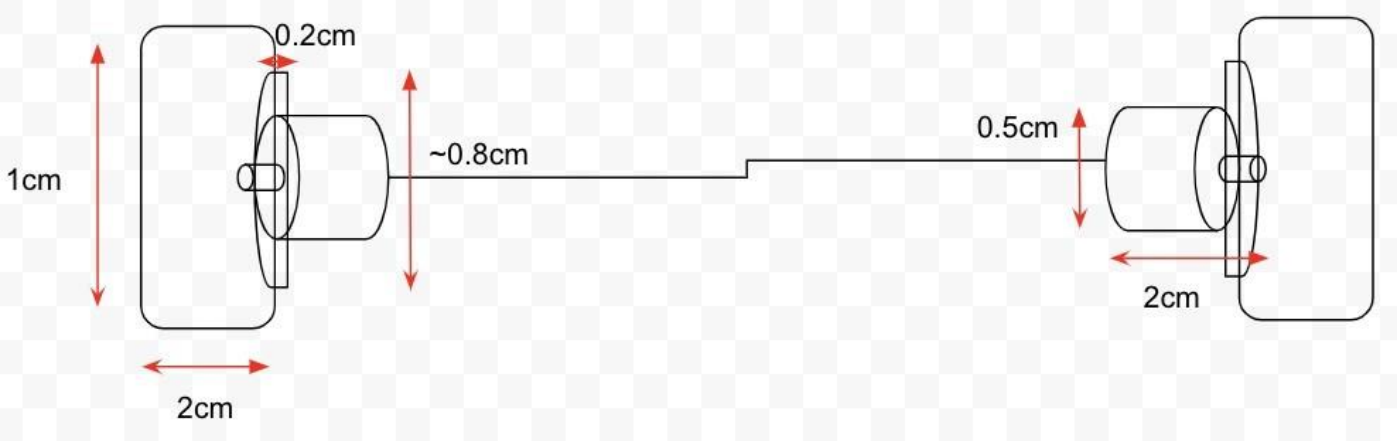


Figure 6

Fixing of Circuitry –
As shown in ***Figure 7***, we decided to make the circuit board 7cm x 7cm, there will be enough space to fit all the circuit components on it.

The circuit will fit on the top of the micro-mouse as this will allow easy access to the circuit if errors arise. However, the circuit will have about a +/- 0.1mm *(in length)* gap between the chassis and circuit held by a standoff pillar. The standoff pillar will require three drilled in holes in the PCB and the chassis to feed the standoff pillar into them; **See Figure 7.**

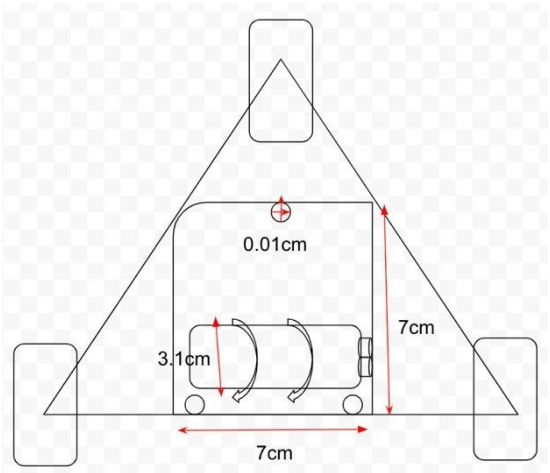


Figure 7

Battery Compartment –

We have decided to use AA batteries for our micro-mouse, these will be attached to the circuitry to provide power to the mouse. The battery will be securely tightened using a battery holder; **See Figure 8,** which will also be connected to the PCB using plastic straps. Battery holder measures at 57.6 x 31.2 x 15mm black polypropylene material.

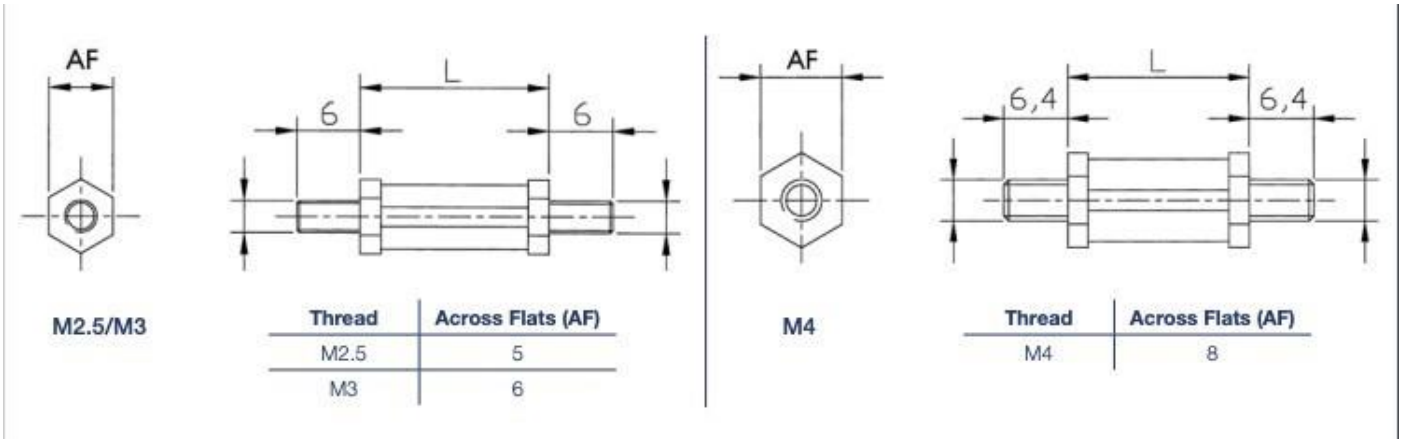


Figure 8

Figure 9 shows how we intend to implement the battery compartment onto the PCB with the use of battery straps. These straps we attend to develop our self-using a bit of plastic which we will strip on top of the battery holder for it to remain stationary on the circuit.

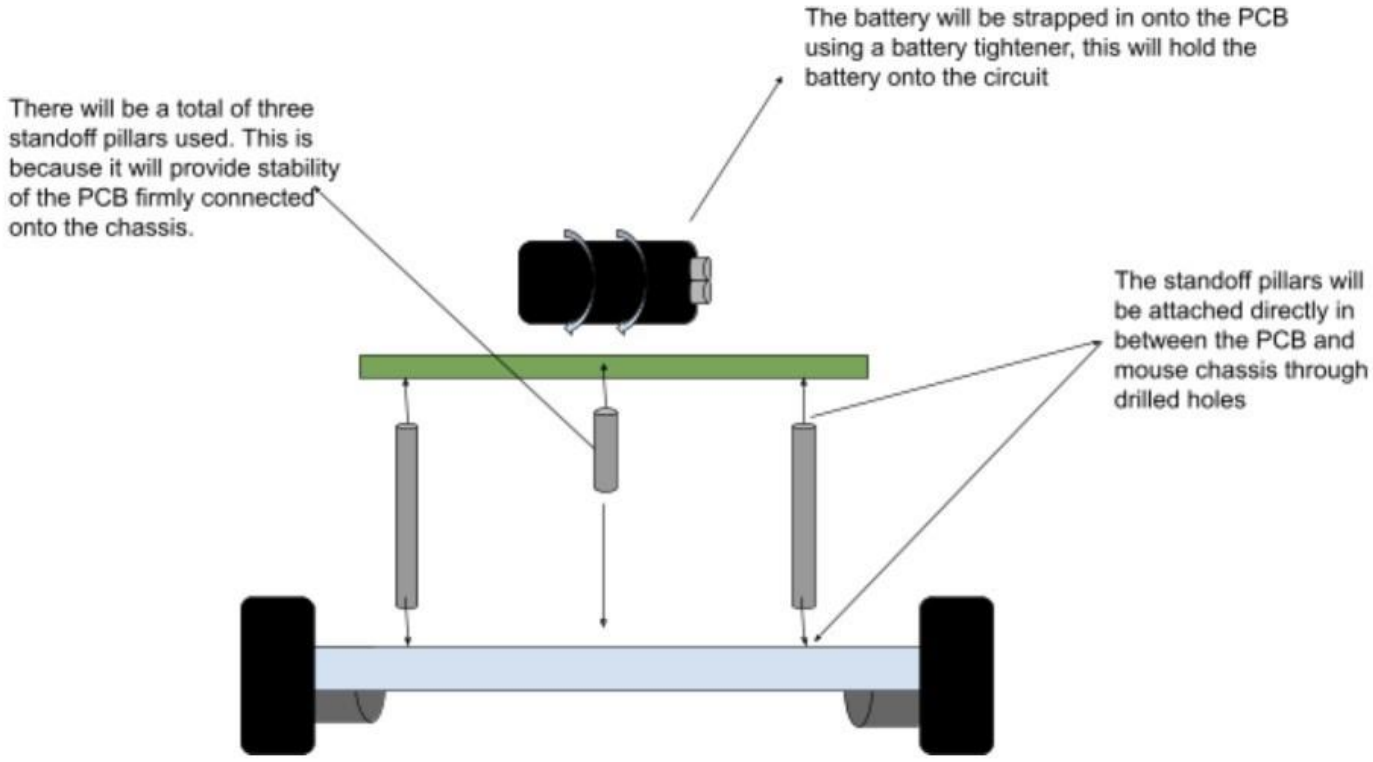


Figure 9

Chassis Dimensions –

As of the chassis dimensions, we intended to make the micro-mouse of tricycle type configuration. Having a tricycle configuration, we had to encounter the fact for the back-axis to be 12cm wide *(including wheels)* and the front axis of a width of 6cm. As this is configured in this sort of configuration so for this to work, we would need the front axis to be shorter than the back to enable manoeuvrability; **See Figure 10.**

Figure 10 refers to all the dimensions of the micro-mouse, the ~ symbolises approximately the dimensions of some of the parts of the micro-mouse but we have tried to be as accurate with dimensions as possible. It is also important to note that there will be loads of other components added on top of these dimensions such as the sensors, circuit board, etc but we have made sure to cover the most necessary parts of the chassis dimensions.

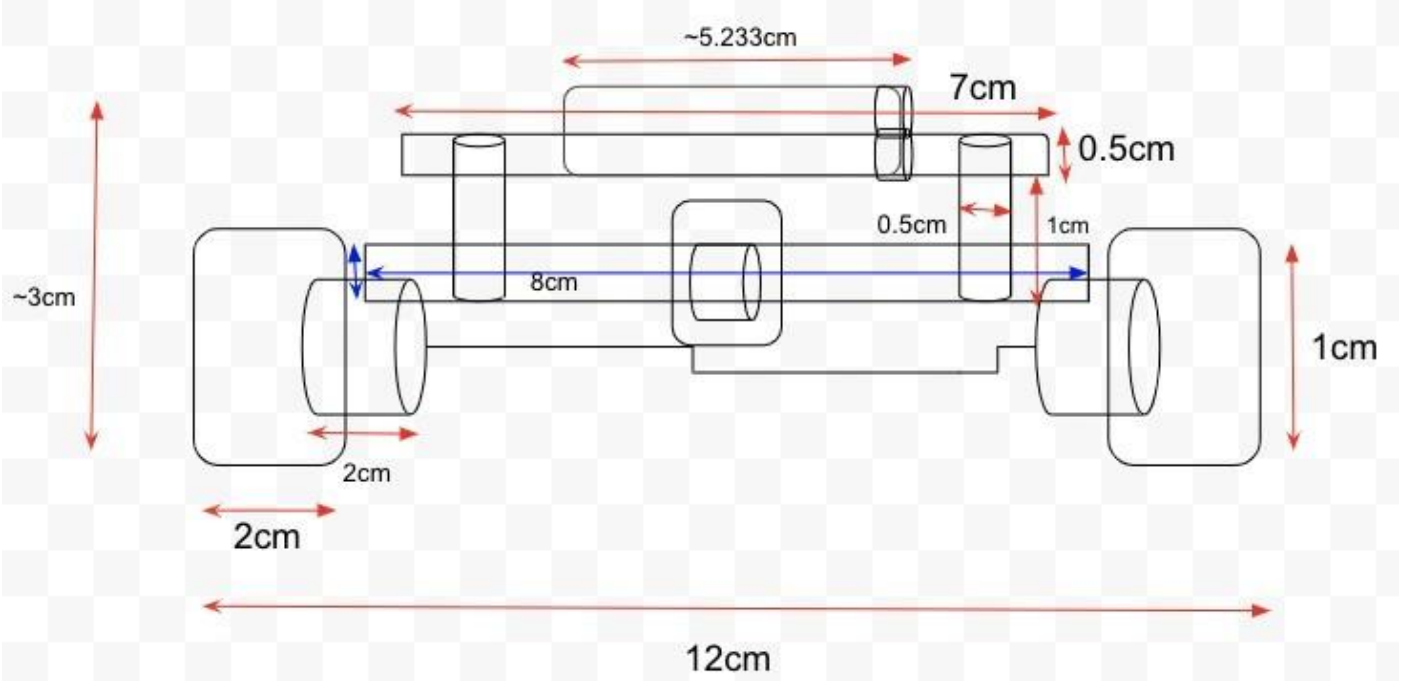


Figure 10

Layout Drawing –

We have created a drawing of how the micromouse is intended to look like after completion. There are four types of view as seen in the drawing in which it shows an abstract idea of where the circuit board will be as shown in the upper deck of the micromouse, the upper desk refers to the top part of the micromouse in which we intend for the circuit to be located shadowing the shape of the chassis being the triangle. The lower deck of drawing refers to the actual chassis of the mouse, on the chassis we intend for the following components to be located on the chassis. Fixing the two points of the upper and lower deck of the mouse will be a standoff pillar as shown in the microcontroller board/side view. As seen in **Figure *11*** there are three-cylinder shaped objects, these will be the pillars. Also, to note in the side view we have taken the picture from a complete side on view, our mouse will have three wheels which isn't shown in **Figure 11.**

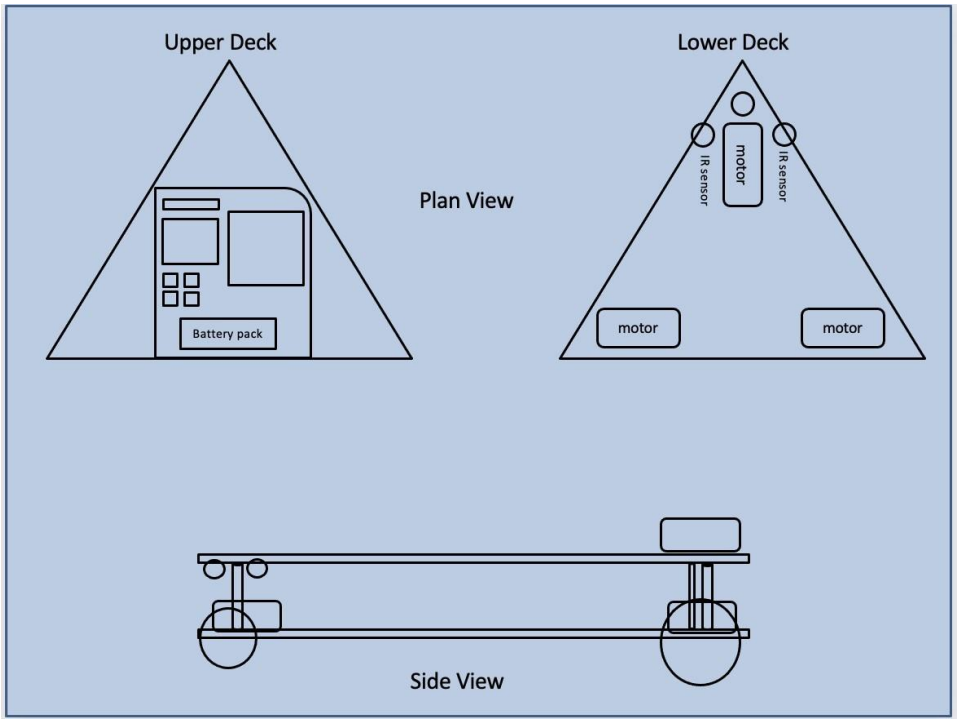


Figure 11

4.2 – Power Distribution

The power supply that we have from the battery stores the chemical energy and converts it to electrical energy which powers up the motors and makes the micromouse move. We needed a powerful energy distributor to make the micromouse move efficiently without faults, so using AA batteries are the source we are going for. The way these batteries provide power supply depends on the flow of the electrons; this provides the electric current needed to perform the workload. Considering, we are using a stepper motor, we need a high-capacity battery. We understand that sometimes the motor will be turned off, to analyse the surroundings, so using a 600 mAh battery will be most reliable as the errors and margin will most likely not occur.

Figure 12 is a schematic of a voltage regulator and the way this works is that it generates a fixed output voltage and this will be constant throughout whereas, the input voltage can be altered and changed and this will not affect the circuit in any way. The reason as to why a voltage regulator is required is because this can maintain the power source within its required limits. In addition, this is needed as the voltage regulator can supply the micromouse with the required voltage needed to function as well as the other components which will be implemented such as the distance infrared sensor, stepper motor and the rest which will be added to the micromouse.

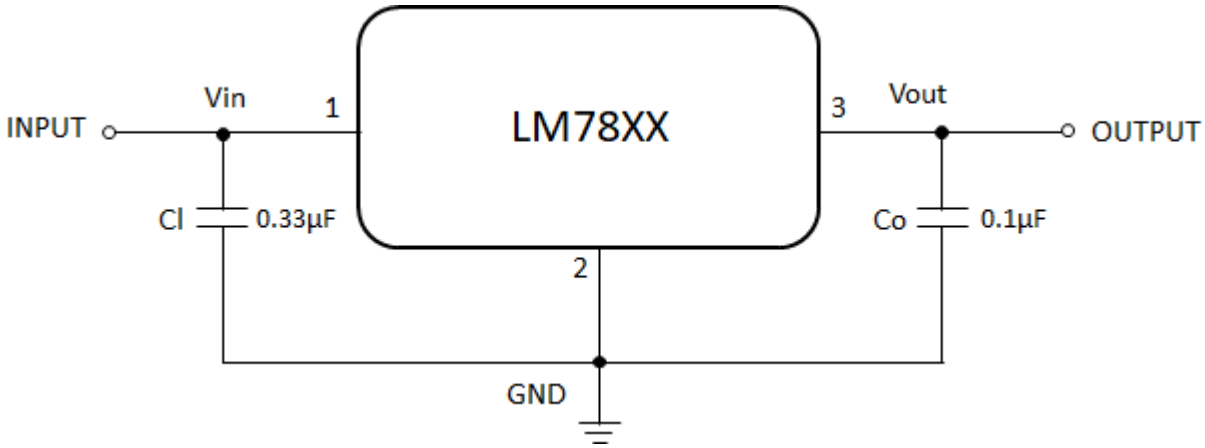


Figure 12

4.3 – Motor Drivers

Stepper motor controllers come in two types, the L/R types, and the chopper type. The purpose of both types of motor drivers is to provide the stepper motor's coil with voltage. They use different techniques to supply the maximum rated current into the coils as quickly as possible, it needs to be as fast as possible so that the maximum torque of the stepper motor can be achieved sooner. The L/R type controller uses the resistance to offset the L/R time constant. They use this method to allow the maximum rated current to reach the motors coil. Where the chopper controllers take a more efficient approach by supplying a large voltage well more than the rated voltage into the coil and then it actively monitors the coils current. [10]

We will be using the "L297" Stepper Motor Controller for our micromouse's motor. This is a great motor driver as it can generate drive signals for both two phases bipolar and four phase unipolar step motors. This motor driver also gives the option to make the motor drive in half step, normal and wave drive modes and it also has a switch mode that allows us to load current regulation. We can program load current; it has little external components and we are able to reset input & home output.

See Figure 13, Within this schematic this includes several components such as, two 3.3k ohm resistors, four 2N6045NPN Darlington Power Transistors, 8 1N4001 general purpose diodes as well as a 12v unipolar stepper motor. In terms of power to make the stepper motor function the power needed is a low power of +5v and a maximum high power of 12v.

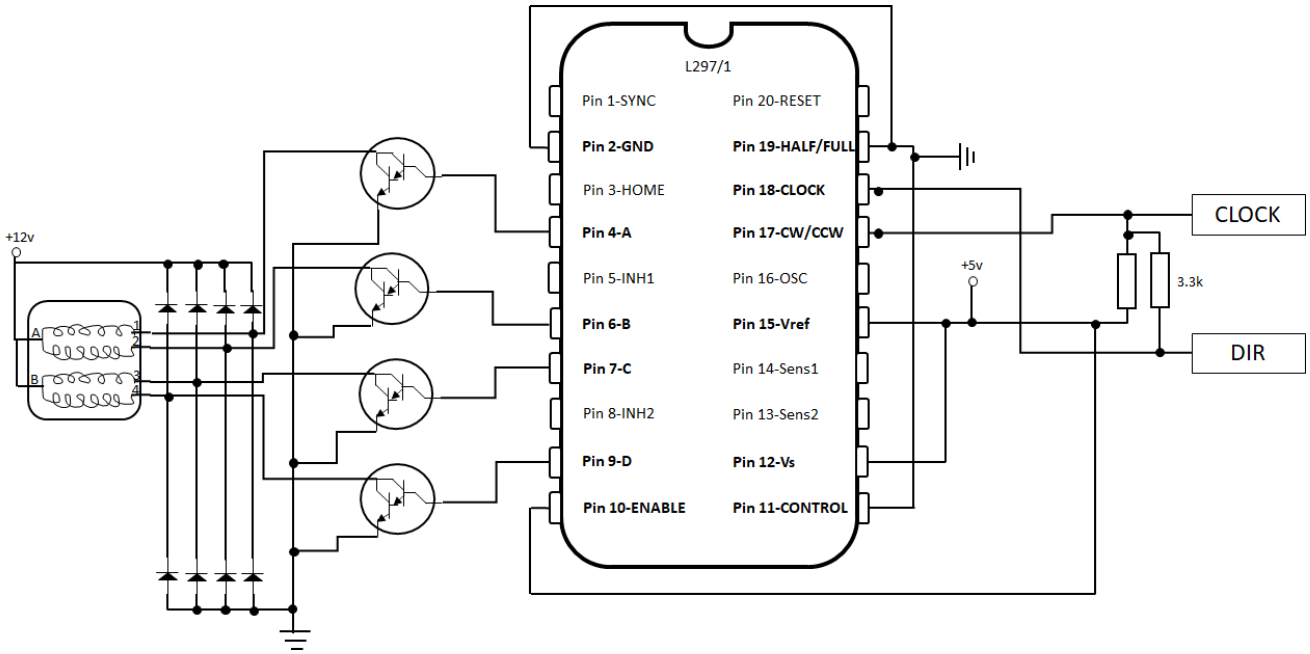


Figure 13

4.4 – Sensor configuration & Logic

Sensor Placement –

The 3 sensors that we are using at the front left and right of our micromouse will need to be configured so that they work correctly and provide the microprocessor with the right information. In this section we will discuss the final design of the sensor system that affects the control and navigation of the micromouse.

The sensor layout of our final micromouse is one sensor at the front of the micromouse, one facing left of the micromouse and one facing right of the micromouse. These are all fitted onto the front of the micromouse so that it can detect the walls as soon as possible to give the micromouse enough time to change to the right corresponding states. The table below shows the 3 sensors and their positioning on our chassis with their bit number that we will use attached.

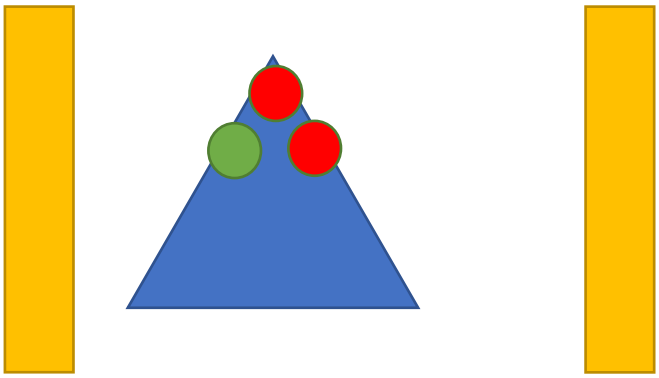
Bit	Sensor Location	Sensor
0	Right of Chassis	Right
1	Front	Front Centre
2	Left of Chassis	Left

We are only using three sensors as we think that 3 sensors will be enough to allows us to navigate through the maze efficiently and it also is cheaper in terms of price and power consumption. The only downside will be that we will not have as great resolution with more sensors but that is a downside, we are fine with as we just want the micromouse to navigate through the maze as fast as possible.

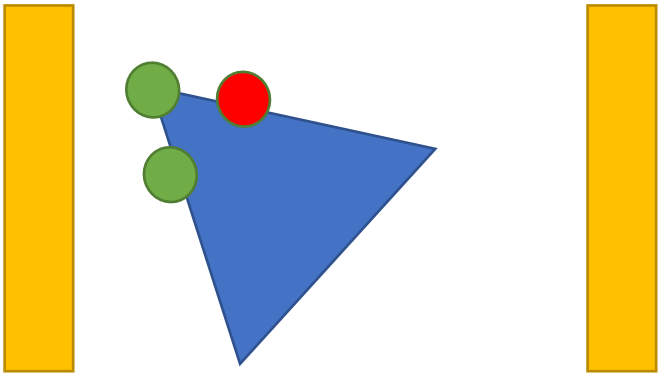
To reduce the amount of correction needed while turning or adjusting the micromouse, the sidewall sensors are placed well forward, this helps the sensors to detect the missing maze walls quicker and lets the micromouse know of the right adjustment quicker too, and in turn this then reduces the angular displacement of the turn before correction.

Sensor Patterns –
“Correction right” possible patterns:

- 100

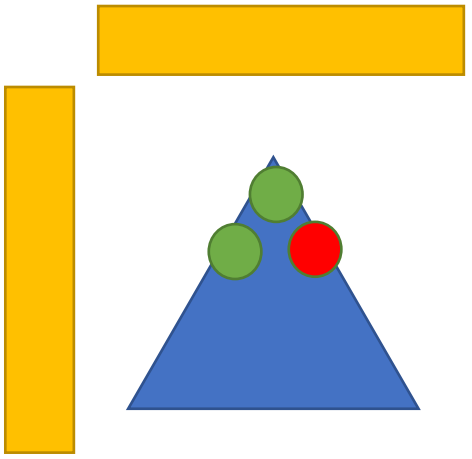


- 110

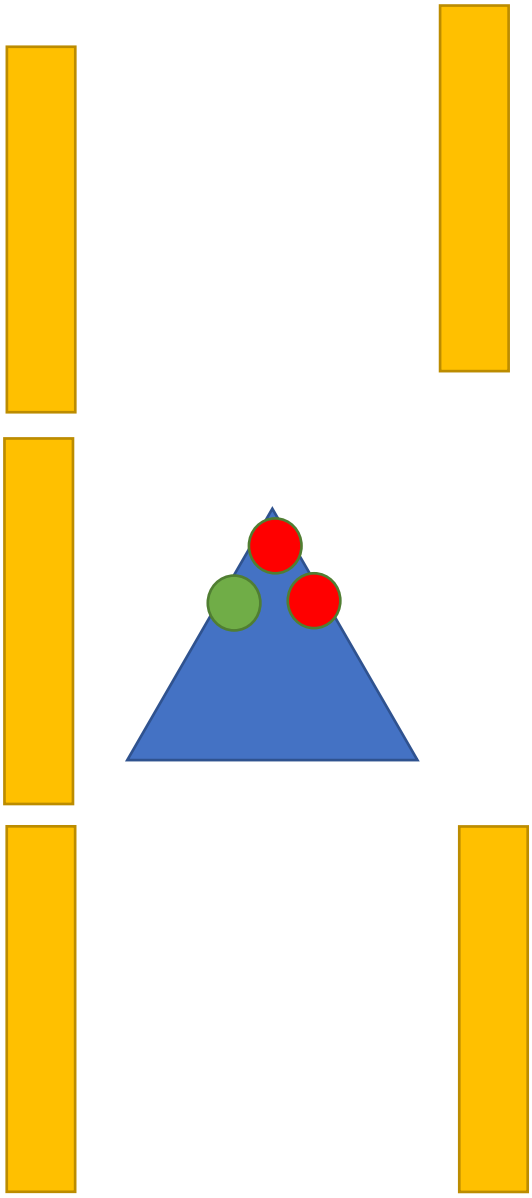


“Right Turn” Possible Patterns:

- 110

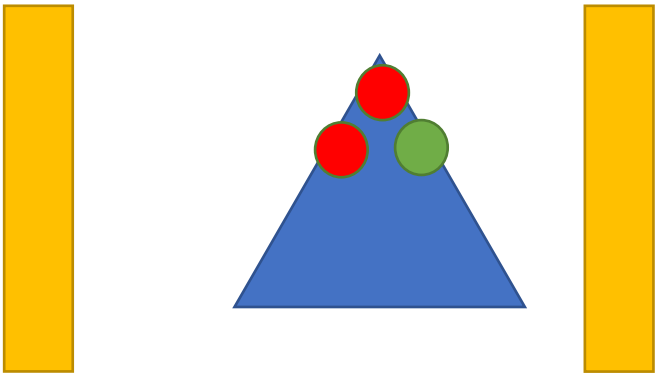


- 100

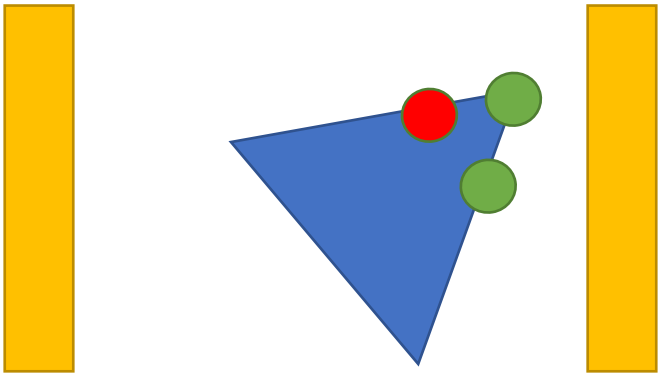


“Correction Left” Possible Patterns:

- 001

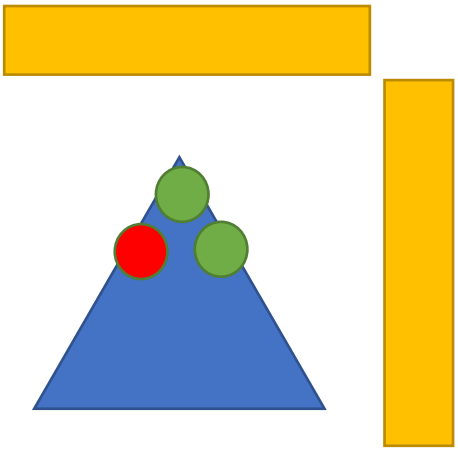


- 011

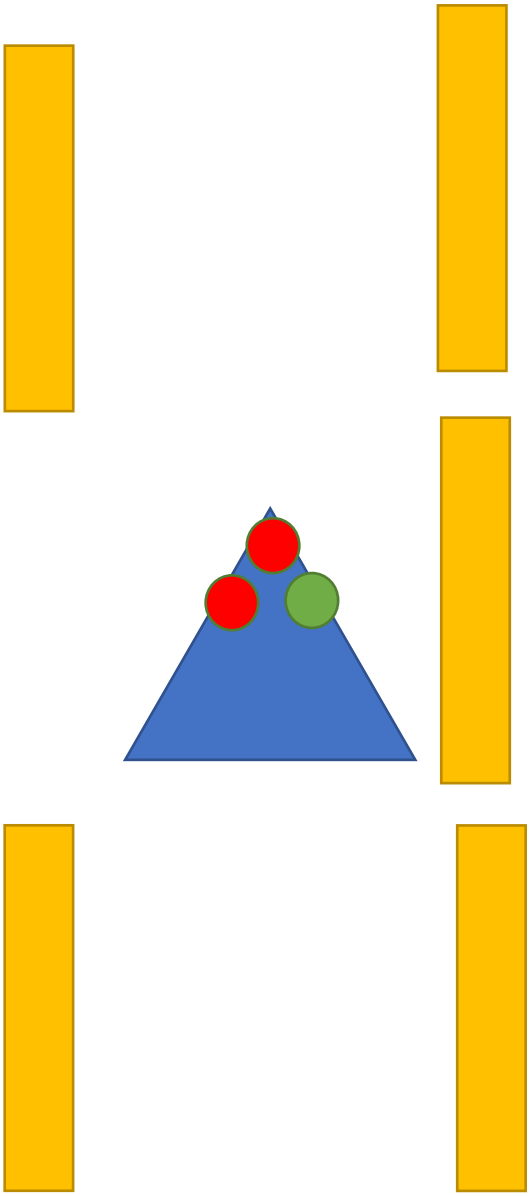


“Left Turn” Possible Patterns:

- 011

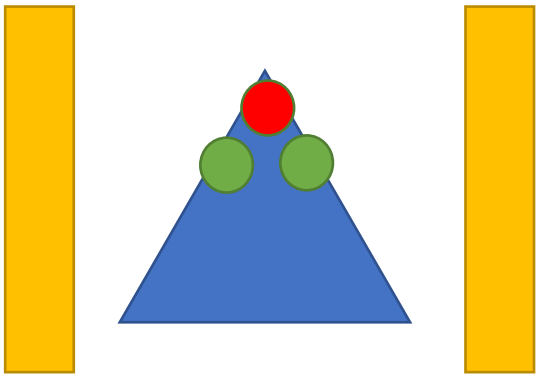


- 001



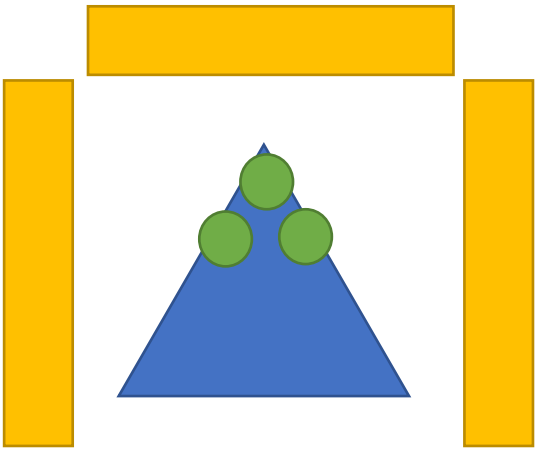
“Drive Forward” Possible Patterns:

- 101



“U TURN” Possible Patterns:

- 1,1,1



State Table –

Now that we have seen the possible patterns our micromouse sensors could have, we can put it in a state table so that the micromouse can use the bits pattern to determine which state it should go into.
The pattern of sensors is “L, F, R”, where L is the left sensor on the left side, F is the front wall sensor and R is the right sensor on the right side.
The way that the sensors can recognise the bits and determine the right state is with the voltage, the sensor will send a voltage to its corresponding pin when a wall is detected, and where there is no wall, 0 volts is sent to the corresponding pin. The microprocessor of the micromouse can use this voltage information from the sensors to decide exactly which state to switch to.

Pattern (L, F, R)	State
0,1,1 0,0,1	LEFT TURN
1,0,1	DRIVE FORWARD
1,1,0 1,0,0	RIGHT TURN
1,1,1	U TURN

The schematic of the distance infrared sensor we will be implementing into our micromouse. The reason as to why we have decided to use this sensor is because it can accurately calculate the distances within the maze where the micromouse will be. The way the sensor works is that it utilises infrared sensors which can measure distance, from this a reflection of light be produced. Once the light has been reflected it can then detect and estimate the distance between the infrared sensor and the object which in this case is the walls for the maze**; See Figure 14.**

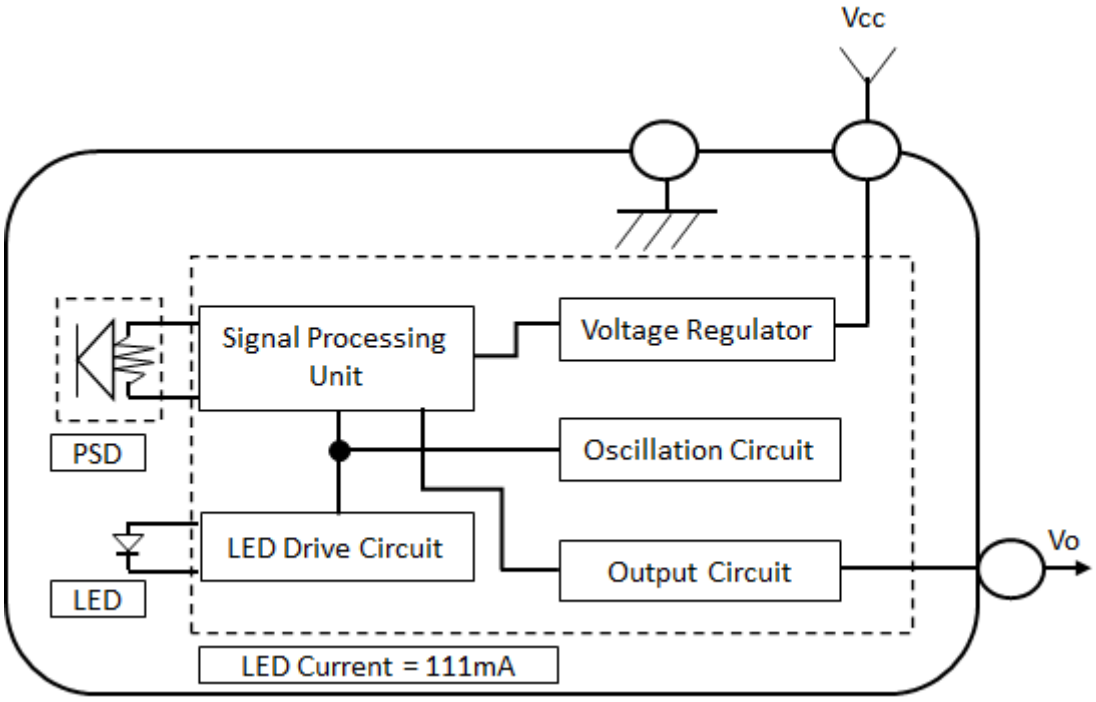


Figure 14

4.5 – Power source

The power system of the micromouse is important as without it the micromouse and all its components would not work at all. This means that the power system must include a source of power that stores just enough energy for the micromouse to run for a certain amount of time without it stopping for any sort of replacement or recharge. Power must also be provided to all the components of the micromouse constantly so that all the circuitry and components work throughout the traversing of the maze correctly.

Small micromouse robots can use batters to consume electric energy as it is very practical and can fit on the micromouse itself. These batteries have enough power in a small size to power all the components and circuitry on the micromouse

We will be using an alkaline battery to power our mouse; the alkaline batteries work in a complete circuit in which it stores chemical energy and converts to electrical energy. The electrodes in the battery contain atoms that conduct certain materials. In an alkaline battery the anode and cathode are typically made of zinc and manganese dioxide. It will cause a chemical reaction because the anode becomes negatively charged whilst the cathode becomes positively charged, whilst that occurs, the difference causes the electrons to move toward the positively charged cathode. When the micromouse is being powered up by the battery the electrons move towards the cathode and it will continue to repeat that cycle if the battery is on.

The alkaline batteries life expectancy is 300 to 800 percent higher than that of zinc batteries and the only disadvantage they have is that they are unable to deliver high currents. A solution to that could be to use bigger d sized cells batter to be able to handle the increased load of the many components on the micromouse.
The amount of current the alkaline battery can deliver is roughly proportional to its physical size. This is a result of decreasing internal resistance as the internal surface area of the cell increases.
A general rule of thumb is that an AA alkaline battery can deliver 1000mA without any significant heating.
Volume for volume, alkaline batteries have inferior current handling capacity when compared to other chemistries like NiCd and NiMH. However, alkaline batteries cost significantly less and are safer
The materials used for large-scale storage will need to be low cost and safe at the desired scale.
The Zn–MnO2 “alkaline” battery chemistry is associated with one-time use, despite being rechargeable. This is due to material irreversibility’s that can be triggered in either the anode or cathode. However, as Zn and MnO2 have high energy density and low cost, they are economically attractive even at limited depth of discharge.
As received, a standard bobbin-type alkaline cell costs roughly $20 per kW h. The U.S. Department of Energy ARPA-E $100 per kW h cost target for grid storage is thus close to the cost of alkaline consumer primary cells if re-engineered and/or cycled at 5–20% nominal capacity.
Here we use a deeply-penetrating in sit technique to observe ZnO precipitation near the separator in an alkaline cell anode cycled at 5% DOD, which is consistent with cell failures observed at high cycle life. Alkaline cells designed to avoid such causes of cell failure could serve as a low-cost baseload for large-scale storage.

There are also other battery types like zinc and lithium batteries, but they contain chemicals that may be too dangerous for the micromouse project if any malfunction were to occur.

We needed to choose a battery that meets the power needs of the components on the micromouse and one that also meets the limitations of the chassis strength and area. We needed a strong battery that was able to fit on our micromouse. This battery would also need to be able to last for a certain amount of time, in this case from 10 minutes to 30 minutes in 1 run. Considering the test runs and such, this battery would also have to be rechargeable.

The alkaline batteries life expectancy is 300 to 800 percent higher than that of zinc batteries and the only disadvantage they have is that they are unable to deliver high currents. A solution to that could be to add multiple batteries or to reduce the current need in total with less components on the micromouse.

The alkaline battery is optimal for our micromouse as it can provide for the power needed to run the components on the micromouse, they have a small internal resistance to allow for potential current surges and they are also quite small and compact. This makes it so that the micromouse can hold the battery and be able to travel and manoeuvre inside the maze without the battery being a hindrance because of its size. The chassis of the micromouse is a maximum of 12cm and the size of the motors are fixed, so the space for batteries is limited and having small batteries with high power is essential. The mass of the battery is also quite small thus decreasing the burden of load on the motor torque. The alkaline battery is quite cheap and meets the requirements of our micromouse accordingly, and the downside of not enough current from one battery can be solved by using two different batteries for different components in the micromouse. Meaning we could use a separate battery to power the motors alone and another battery to power the rest of the components and circuitry.

So, after looking at all the possibly batteries that we can used and the different types of battery materials, we decided to choose the battery of type of 9V which given in the name will provide 9 volts, to our micromouse, to the stepper motor and to all the other components that we had added to it. We ended up choosing this battery type because the stepper motors require the voltage of 2.8 V and the other batteries only provide up to 1.5 volts so instead of using couple of AA or AAA batteries, we decided to use the 9V one in this project. [11]

5 - Software Design & Logic

5.1 – Overall system discussion

We needed to create a flow chart that showed how the process of our micromouse would work and how each of the components would communicate with each other while the micromouse is running. This flowchart can assist us in writing the code and knowing the order of operations; **See Figure 15.**

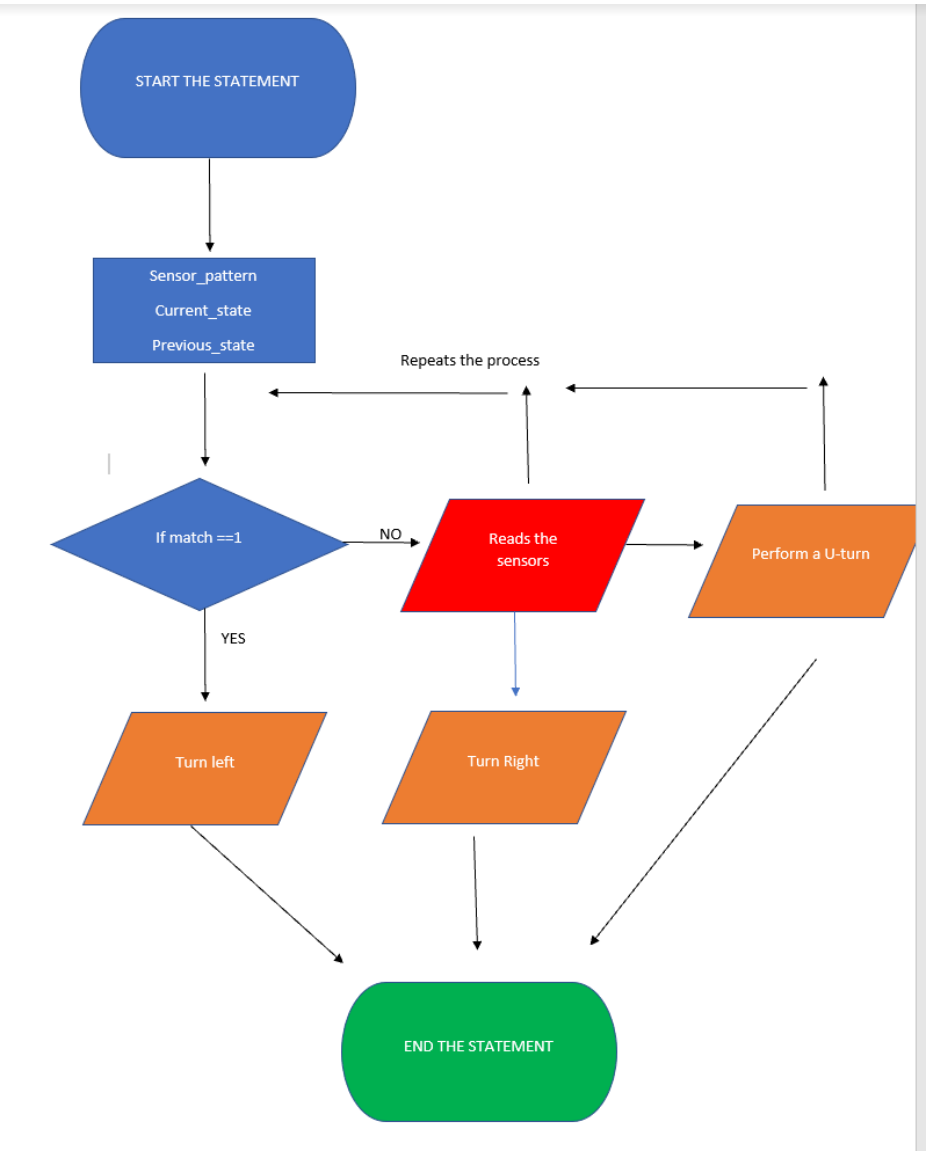


Figure 15

Ultimately, we were able to draw the micromouse and type out the code. We were able to balance the components to make sure the micromouse can perform the necessary requirements. By checking the width and length of the tunnels of the maze, we were able to think of a reasonable configuration to make to avoid any unfortunate errors that could occur. The micromouse is designed as a tricycle where the front axis is 6cm wide and the back axis is 12cm wide. As this will be connected to the “L297” stepper motor, we can repeat multiple steps in the micromouse so the code itself will include multiple void functions to allow the micromouse to move forward, when there are no obstructions and able to turn if there is an obstruction ahead and reverse if there is an obstruction on both the front wall and side wall. The stepper motor may not be able to run in high speed, but we are making our micromouse efficient, so it does not have to go through the tunnels in the maze whilst hitting the walls of the maze. We are using multiple sensors such as infrared, acoustic and touch sensors. The way we connected the sensors in the code is by using one of the pins from the mbed board which was the analogin pin16 where we used to code the infrared sensor and make it do functions and controlling the time of when it's going to happen with the wait function. As we are using AA batteries for our power source to power up the stepper motor, ultimately, it’s a good match knowing that stepper motors do require high amount of power to perform multiple steps and AA batteries can transfer 1000mA without having aftereffects like overheating. We’ve made our micromouse to perform steady and efficiently and this is what we wanted from our micromouse.

5.2 – Software structure

Figure 16 |

This is the code we used to make our micromouse move to different directions. We have used multiple if else statements to state the outcome if the micromouse is turning or not. As we had to implement the motor in the code we typed #include “Motor.h”. We also needed to add the sensors from different pins, so for our infrared sensor we used pin16 from Analogin. We used the wait statement to give the micromouse time to process what to do if a wall is blocking the route or if there is not a wall on the side view. We have made the character variable called “char update” where “sensor_pattern,current_state and previous_state is stored in char update. For each if statement we made a statement what will happen if it turns and what would happen if it does not turn. In our if statement, if the match is == 1 then the next state for the micromouse will be true. However, if it is not true then it would return to the current state (how it was before). Within this if statement it is stored in “sensor_pattern”. We have implanted the same block of code for other movements like right turn, left turn, and u turn. By storing each if statement in “sensor_pattern”, we made an if statement to determine if the pattern is true then it will proceed whatever the next state is, and it will have a break to analyse what is next. For the else statement if it is false then it will not turn therefore it breaks to analyse what happens next.

```
//pinky and the brains
char match = 0
match = (check_TURN_LEFT)(sensor_pattern);//storing into sensor_pattern
{
    if(match==1)
{
    next_state = TURN LEFT; //if its true then it will turn left
    else                    //if not it will read the sensors and check
{
    read sensors;
    return_current state; //it wont turn and continues to move
    return next_state;
}
}
match = (check_TURN_RIGHT)(sensor_pattern);// if its true then it will turn right
if(match==1)                               //if not it will read the sensors and check
{
    next_state = TURN RIGHT;
    else
{
    read sensors;
    return_current state; //it wont turn and continues to move
    return next_state;
}
}
match = (check_U_TURN)(sensor_pattern);//if its true it will perfrom a u turn
if(match==1)                           //if not it will read the sensors
{
    next_state = U_TURN;
    else
{
    read sensors;
    return_current state; //it wont turn and continues to move
    return next_state;
}
}
match = (check_TURN_RIGHT)(sensor_pattern);

return previous_state; //if it doesnt match then it will return what it was before
    }
```

```
#include "mbed.h"

Analogin IR(p16); //For the IR sensor from the analogin pin

void Leftturn() //Controlling the motor to turn left

{
    MR.speed(0.1);
    ML.speed(0.3);
    wait(2.0);
}

void Rightturn() //Controlling the motor to turn right

{

    MR.speed(0.1);
    ML.speed(0.3);
    wait(2.0);
}

  void Moveforward() //Controlling the motor to move forward

{
    MR.speed(0.5);
    ML.speed(0.5);
    wait(0.05);
}

Void Reverse() //Controlling the motor to reverse

{
    MR.speed(0.5);
    ML.speed(0.5);
    wait(0.05);
}
```

Figure 16

6- Implementation of Hardware

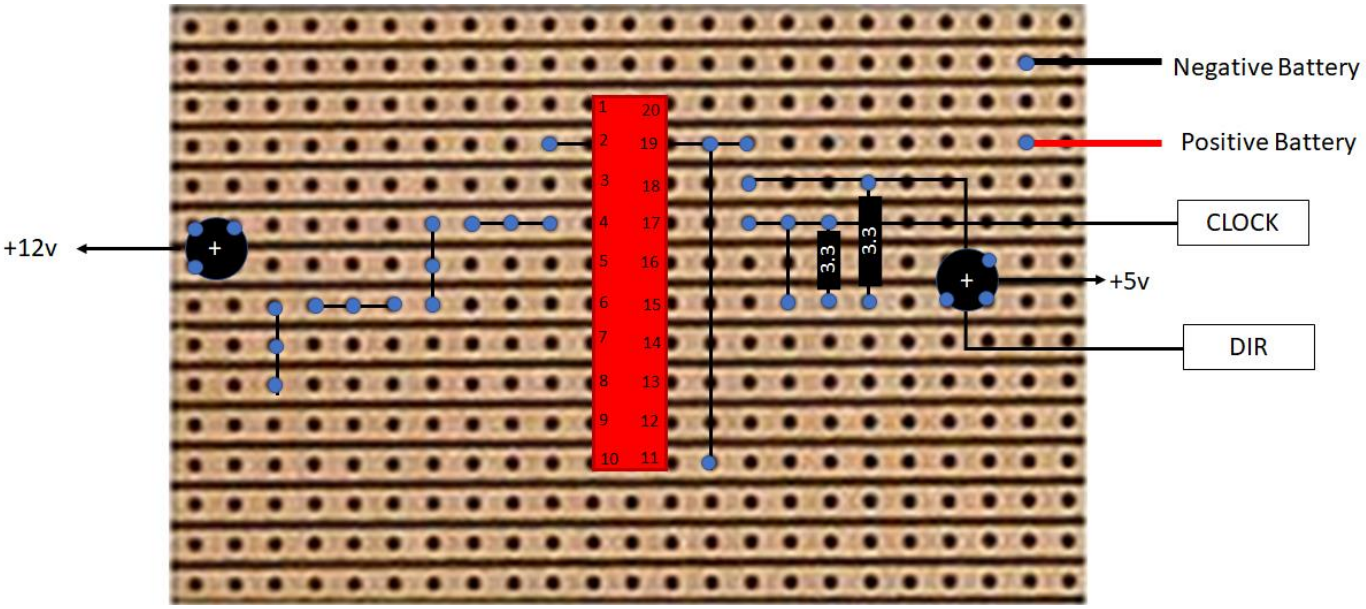


Figure 17

7 – Testing

We will have to design the micromouse in a way so that we are able to test and debug any of the components if need be. Testing of all the micromouse components will be essential as it will let us know that everything is working according to plan and that if they are not what type of changes we need to make.
Designing the micromouse in terms of testing for power in batteries. We would have to choose a battery that is rechargeable and to choose a battery with individual cells that have a relatively high voltage. With the rechargeable alkaline batteries chosen, when we test the micromouse with all the batteries connected to the components. We can test the micromouse by running it through a test maze and seeing if the batteries are able to produce enough voltage and current to power all the components on the micromouse. The rechargeable battery will a allow us to perform multiple tests without having to worry about getting a new battery to replace it, as removing the battery from the chassis may cause some accidental damage unwillingly.

To test if the infrared sensor is detecting objects correctly, we could place the infrared sensor face a piece of paper and measure the output voltage produced by the sensor depending on the distance between the sensor and the piece of paper. The distances can be measured with ruler in centimetres and it can be incremented by half a centimetre for every output voltage test. This test can give us results and confirm the distance in which the sensors are able to detect the piece of paper and which distance it does not detect the piece of paper. To make sure we can minimise any anomalies, three tests can be done at each distance and averaged to one final output voltage. These distance and output voltages can then be plotted on a graph to further analyse any problems with the sensors that may be occurring.

We also need to test the sensors states to see if the sensors are giving the microprocessors the right information and then to test to see if the microprocessor is using the information to correctly switch to the right state. We can do this with small LED's that are on the MBED board. The micromouse will read the value of the state variable and displays the state that it is currently in using the 4 LEDS on board the MBED board of the micromouse. Using the onboard LEDs will help us to save space and cost of buying external LEDs to test the states.
The pattern of the 4 LEDs will show us which state the micromouse is currently in. The state display table for our micromouse is shown below.

State	Display LED1, LED2, LED3, LED4
LEFT TURN	On, off, off, off
DRIVE FORWARD	Off, on, on, off
RIGHT TURN	Off,off,off,on
U TURN	On,off,off,on

Once we have completed the chassis design, we will be able to test the strength of our chassis by attaching all the components on it and activating the micromouse for a test run. This test run will not only allow us to see if the chassis is able to withstand the weight of the components, but also see how well the stepper motors and wheels of the micromouse perform. We can analyse the speed and turning of the micromouse to see if there are any ways to make it more efficient and improve it.

Conclusion

During the time of the development of the micromouse, we were able to come up with an idea to producing the micromouse on paper however due to current global circumstances we were not able to physically build the mouse and be able to test it. What we have completed as a group is that we were able to come together to put forward ideas on the kind of components we thought was best for the mouse in order for it to perform within the IEEE maze. We were able to successfully put together a micromouse in which we thought was best suited to operate in the micromouse. We do believe if we were able to physically build the mouse in a laboratory, we believe we might have come across issues which may have not arisen from our own judgement from what we see in the virtual environment in which we were able to test in. From these real-life errors, we could've detected fault in maybe the choice of technical components used in the mouse in which we may not have seen in the first place.
During the development of the mouse, we had to really consider our options of what components we were going to implement into the mouse. As a group we came together to break down the IEEE description of the maze of what we were given in order to be able to produce a mouse which would be able to tackle the maze with no prior knowledge of it. Firstly, we needed a structure for our mouse, the chassis, this had to be a good enough structure in terms of stability, durability, and operability. We also wanted to think roughly where we would have sensors on the mouse as well as other important components. We decided that the chassis would be in the triangle configuration, the benefits of this configuration would enable us to have three wheels of which the front one being independent for steering. Having this sort of configuration allowed us to have three points where we can place our sensors for a more accurate movement of the mouse, by having multiple sensors the mouse would be able to take into its surroundings in much more detail making out where obstacles will come up in its path more distinctively with the aid of multiple sensors. We decided that we would be using IR sensors, these types of sensors would be located on the underside of the chassis on each wheel. By having it on each wheel it will make the mouse be able to recognise its surroundings much better as it moves long the maze.

Having three wheels we had to have motors for each wheel, as the front wheel would be for steering, we were come across the issue of the front motor being two big as it was the same as the back two. To overcome this, we had implemented a steering motor of much smaller size that will fit on the mouse. By having the motor, we also believed it would help with the weight distribution of the mouse, if we had a big heavy motor at the front, it would affect the mouse performance whereby during a turn there may be the worry of the mouse tilting or tipping over due to weight disproportion. By having big motors on the back axle, it will act as an anchor holding the mouse down on ground. By having these sorts of motors, they would require a lot of power to operate them. At first, we thought AA batteries would be enough to power the mouse because they were relatively cheap but looking at the motors data sheet, we then saw that more power was required. More power would also mean it would be costing more however we needed enough power to power all three motors as well as the rest of the mouse. Thus, we were able to decide that we needed a battery in which surges more power, we concluded that 9v would be superficial for powering the mouse.

Creating the structure and designing all the different parts of the micromouse as a group was a challenging task to do. It involved 5 brains having to work together in unison to create a micromouse out of nothing, just like how our micromouse has many different components that must work together in unison to finish the maze. It was a daunting task to all of us to undertake a project with such great magnitude in the current conditions of the world too. The goal of the project was to create reports of the design and implementations of the micromouse that we could have made if we were indeed able to work in the practical workshops. Alas, we were not able to and we couldn't do anything about it.

This project has been although challenging, quite a great learning experience for our group. We learnt how to better communicate and work with each other by delegating tasks and trusting each other to come up with the work. This project has taught us a lot of things and we think it is of great value to our future academic goals as succeeding engineers. Completing the written part of this project gives us a lot of experience in the report form of creating and engineering robots. In addition, the fact that we could still be able to code and write complex algorithm for the code also gives us aspects of how it will be like creating or engineering a robot in the real world.

Some possible future work we intend to do with the micromouse to make it more efficient and effective. Such things to make it more efficient is to change the material of the chassis, if we were to have a large budget, we would change the Lego chassis into something of which is more lightweight whilst still holding the properties. Even thought we have made a large amount of progress in a short time of working together, and with the current global condition. We hope to be able to one day use this project as a learning experience for our future work with anything related to creating a robot.

To conclude, we have had a lot of fun and a lot of agony during this project, fun being actually having concepts that work with each other and the agony being the parts where things don't really work out together, but all in all it has been an amazing learning experience for every member in our group and we cant wait to see what is entailed for us in the future!

References

[1] M. Kramer, “The DC Motor,” 13 December 2012. [Online]. Available: http://large.stanford.edu/courses/2012/ph240/kramer2/. [Accessed 22 March 2021].

[2] R. Bosch, Automotive Handbook, Germany: Robert Bosch, 2014.

[3] S. Sutara, “Speed Control Methods for Stepper Motors,” 2009. [Online]. Available: https://www.egr.msu.edu/classes/ece480/capstone/spring09/group06/Sutara%20AN-%20Stepper%20Motor%20Speed%20Control%20Methods-1.pdf. [Accessed 23 March 2021].

[4] A. A.-S. G. E. R. Institute, “Electromagnetic Radiation Sensors,” Emerald insight, 27 June 2008. [Online]. Available: https://doi.org/10.1108/sr.2008.08728caa.002. [Accessed 24 March 2021].

[5] I. o. E. Engineers, “Micromouse Competition Techincal Information Pack 1993,” 1993. [Online]. [Accessed 23 March 2021].

[6] E. Staff, “Sampling rates for analog sensors,” embedded, 11 June 2003. [Online]. Available: https://www.embedded.com/sampling-rates-for-analog-sensors/. [Accessed 13 February 2021].

[7] “FUT0336 DFROBOT,” TEM - Electric Components, [Online]. Available: https://www.tme.eu/gb/details/df-fit0336/accessories-for-robotics-and-rc/dfrobot/fit0336/?brutto=1&gclid=Cj0KCQjwrsGCBhD1ARIsALILBYormwsOkrvARBI-7ftKmUhnDhmMZ2EcYUJ61k288nnh93FY3jAmhSEaAgS_EALw_wcB. [Accessed 15 March 2021].

[8] J. Kordik, “How do you mount a step motor?,” Applied Motion Products - A Moon's Company, [Online]. Available: https://www.applied-motion.com/news/2015/10/how-do-you-mount-step-motor. [Accessed 16 March 2021].

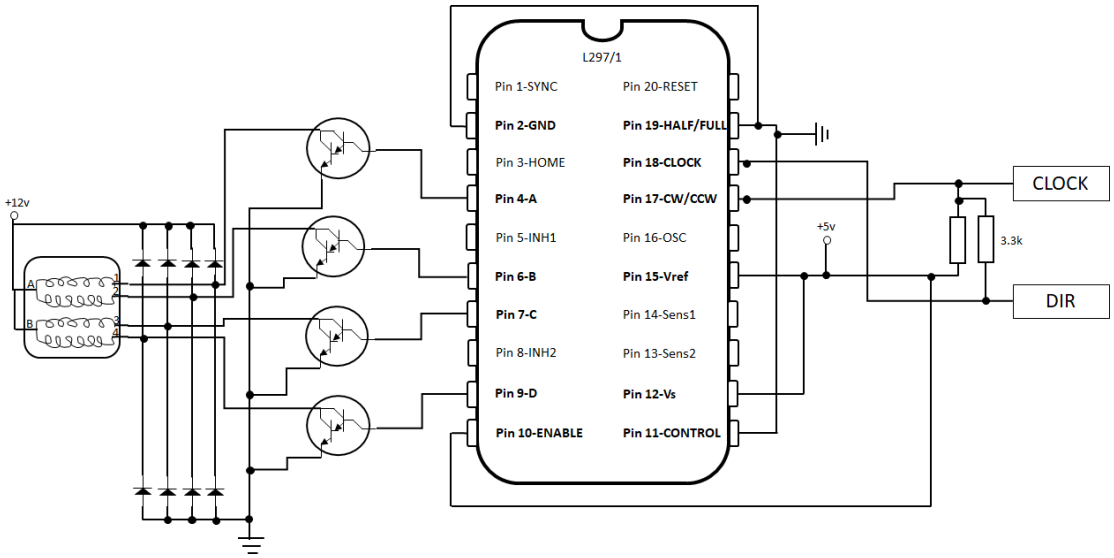
[9] “Mounting Bracket,” RS, [Online]. Available: https://uk.rs-online.com/web/p/enclosure-mounting-brackets/1368956/?cm_mmc=UK-PLA-DS3A-_-google-_-CSS_UK_EN_Enclosures_%26_Server_Racks_Whoop-_-Enclosure+Mounting+Brackets_Whoop-_-1368956&matchtype=&pla-305354315265&gclid=Cj0KCQjw0caCBhCIARIsAGAfuMxLCfQbL. [Accessed 16 March 2021].

[10] H. K. Pittman, “Stepper Motor Drives: Factors to Help Determine Proper Selection,” Tech Briefs, [Online]. Available: https://www.techbriefs.com/component/content/article/tb/supplements/md/features/articles/34885#:~:text=The%20motors%20operated%20with%20an,more%20(see%20Figure%203).. [Accessed 06 April 2021].

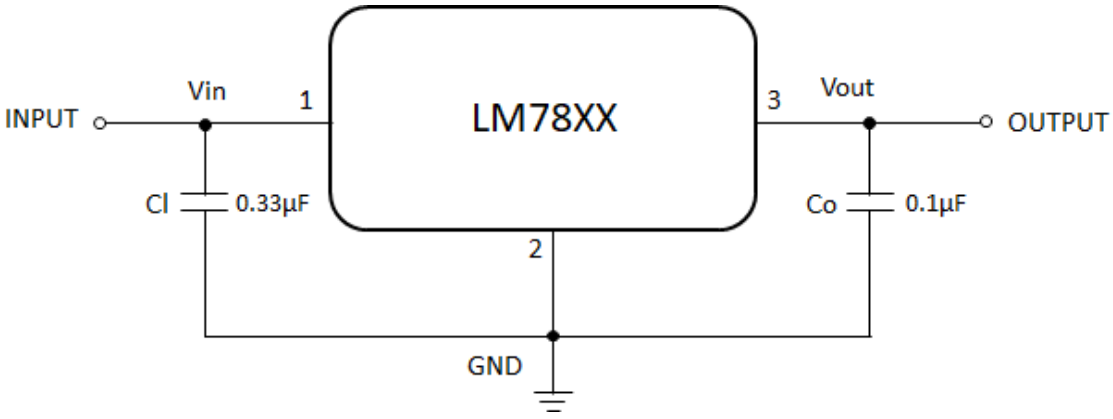
[11] J. E. C. Z. Z. C. M. S. L. S. T. T. D. B. S. a. S. D. Gallaway, Real-Time Materials Evolution Visualized Within Intact Cycling Alkaline Batteries, 2014.

Appendix A: Schematics

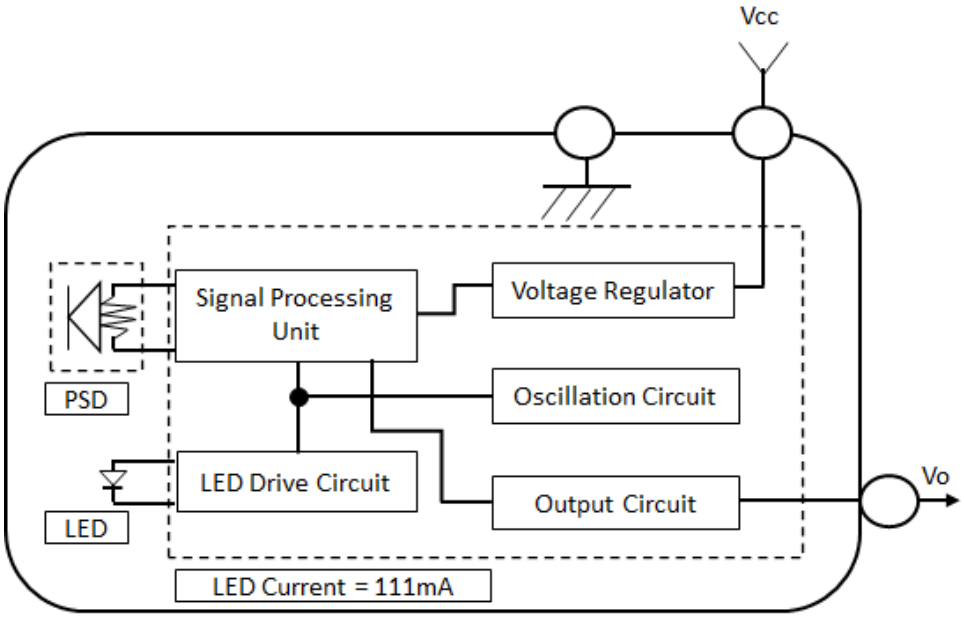
Unipolar Stepper Motor Schematic



Power Management Circuitry/Voltage Regulator



Distance Infrared Sensor Schematic



Appendix B: Full code

```
// Pinky and the brains
#include "mbed.h"
#include "Motor.h"

DigitalOut myled(pin21,pin22,pin23);
tick_count++  //count how many ticks there are
char update(sensor_pattern,current_state,previous_state)
char control(sensor_control,final_state)
char pattern(pattern1,pattern2,pattern3,pattern4)
char turning_end
Analogin IR(p16); //For the IR sensor from the analogin pin
Servo myservo(p24); //To control the servo from the pwm pin

void Leftturn() //Controlling the motor to turn left
{
   MR.speed(0.1);
   ML.speed(0.3);
   wait(2.0);
}

void Rightturn() //Controlling the motor to turn right
{
   MR.speed(0.1);
   ML.speed(0.3);
   wait(2.0);
}

void Moveforward() //Controlling the motor to move forward
{
   MR.speed(0.5);
   ML.speed(0.5);
   wait(0.05);
}

Void Reverse() //Controlling the motor to reverse
{
   MR.speed(0.5);
   ML.speed(0.5);
   wait(0.05);
}

if(tick_count = n){
   switch(state)
   Case 1: Turn_left;break; // Turns left then break
   Case 2: Turn_right;break; // Turns right then break

}
```

```
if (state != turning)
{
   read sensors; // If turning equals to false then it reads the sensors
else
{
   update_position // If it does turn then it will continue the loop again
}
}
if (turning_end)
{
   update_orientation() // If the turning is successful
                  // it repeats the process
}

int main()
while(1)
{
   myled = 1;
   wait(0.4);
   myled = 0;
   wait(0.6);
}
char match = 0
match = (check_TURN_LEFT)(sensor_pattern);//storing into sensor_pattern
{
   if(match==1)
{
   next_state = TURN LEFT; //if its true then it will turn left
   else                //if not it will read the sensors and check
{
   read sensors;
   return_current state; //it wont turn and continues to move
   return next_state;
}
}
match = (check_TURN_RIGHT)(sensor_pattern);// if its true then it will turn right
if(match==1)                        //if not it will read the sensors and check
{
   next_state = TURN RIGHT;
   else
{
   read sensors;
   return_current state; //it wont turn and continues to move
   return next_state;
}
}
match = (check_U_TURN)(sensor_pattern);//if its true it will perform a u turn
if(match==1)                        //if not it will read the sensors
{
   next_state = U_TURN;
```

```
   else
{
   read sensors;
   return_current state; //it wont turn and continues to move
   return next_state;
}
}
match = (check_TURN_RIGHT)(sensor_pattern);

return previous_state; //if it doesnt match then it will return what it was before
   }

char match = 0
if (match ==1)
{
   sensor_pattern = match; //proceeds with the next state
   break;
else
{
   match = 0; //it wont do anything therefore check it again
   break;
}
return match;
}
```