University of Florida   Fall 2020   Computer Engineer

EEL 4924 – Electrical Engineering Design 2 (Senior Design)   Johnny Li

Final Project Design Report:

Car Insight

Sunday, December 13, 2020

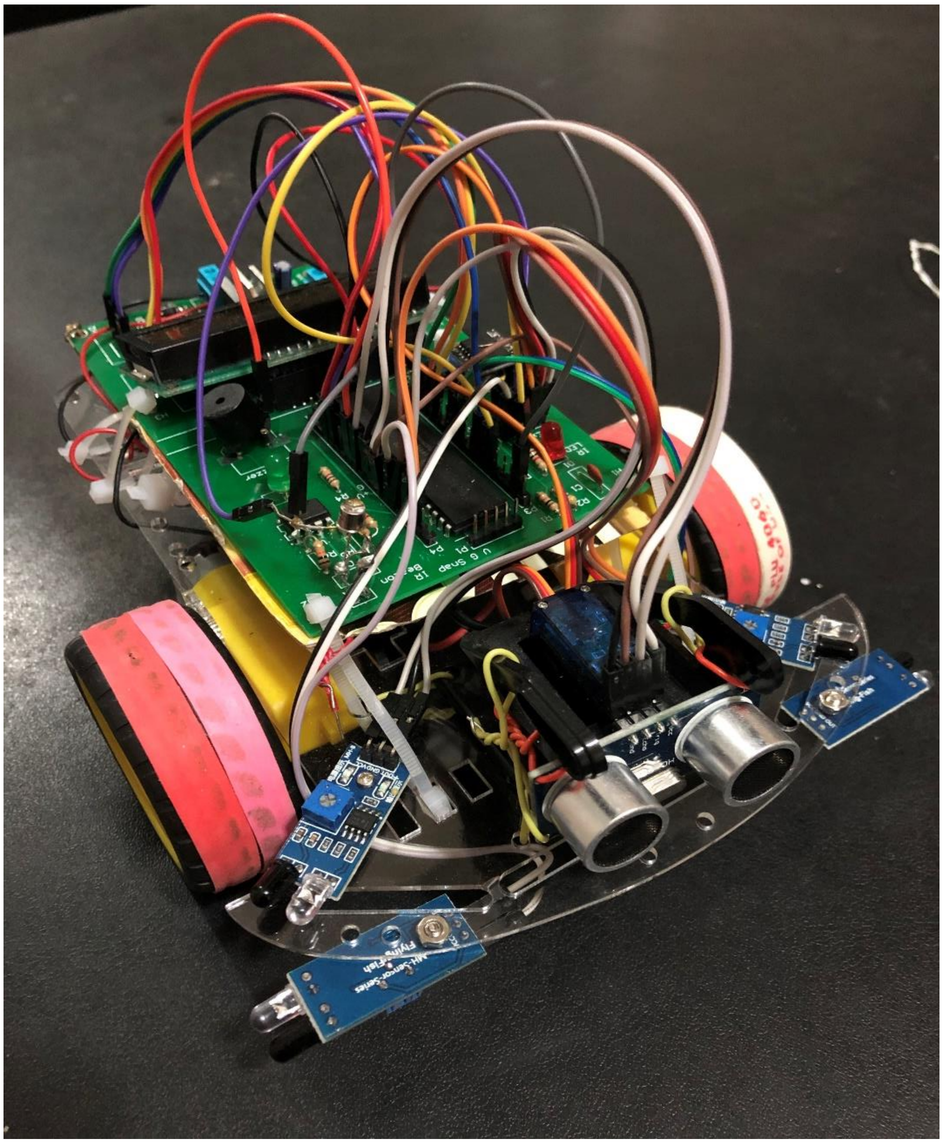


Figure 1: Battery Powered Autonomous Robotic Car

Submitted By: Johnny Li [kadash12@ufl.edu]

Advisory Engineer: Zane P Zudans [zpzane@ufl.edu]

Table of Contents

List of Tables and Figures

Project Summary (Abstract)

The idea for this project is to build a battery-powered autonomous robotic car and use it as a platform for testing proximity detection and collision avoidance to independently solve a given maze. The system allows the car to detect obstructions in its paths and proceed to automatically adjust its movement to avoid a collision while recording that path taken by the car in the maze. The car will use an ultrasonic object detection sensor mounted on a servo to sense 180° degree of its surrounding for obstacles in its path and will produce various warnings, both visually and audial, if an object is within its proximity for the user's benefit. The pathing of the car is determined by the Left-Hand Rule (LHR), "If the maze is simply connected, […] then by keeping one hand in contact with one wall of the maze the solver is guaranteed not to get lost and will reach a different exit" (Maze solving algorithm). Upon reaching the end of the maze indicated by an infrared (IR) beacon, when the car detects the IR beacon it will signal the car that it has reached the end of the maze and to reverse direction to take the shortest path to the entrance calculated by the Dijkstra's algorithm. An LCD mounted on the car will display detected distance from the last scan and current running time in the maze.

Technical challenges include:

- Interfacing with the ultrasonic detector to produce accurate distance measurements.
- Controlling the car's movement and preventing overturning or misguided directions.
- Interfacing with the IR detectors for error correction movements or direction adjustments.
- Having everything fit on the car properly, no conflicts with balance or motion.
- Programming the LHR pathing and Dijkstra's algorithm to successfully navigate a maze.
- PCB designs of the dual H-bridge motor driver, IR beacon, and main controller for the car.

The expected outcome is that everything will work properly.

Introduction

The Car Insight project finds application in the domain of proximity detection and collision avoidance of autonomous self-driving cars. The robotic car has to recognize an object and calculate its distance to avoid hitting the object. With the ever-increasing demand for autonomous vehicles, it is necessary that enhanced detection measures, smarted avoidance methods, improved navigational abilities are applied to ensure the safety of the people within the vehicles and those on the road with them. Such vehicles are also becoming popular in operations that are dangerous or inaccessible by a human. The vehicles can be programmed or designed for several different applications or missions. Such vehicles will be able to detect objects that might have been previously unknown by the driver thus improving safety and decreasing risks. The purpose of the project is to design is to model an autonomous car with navigational abilities.

Project Features/Objectives

The main objective of our project is to design a car that can navigate a maze while detecting and avoiding obstacles in its path.

Size - The system must be designed to the constraints of the car without conflicts to balance or motion.

Battery Powered - The power rails, associated components, and devices should be rated at 5V. The system should be able to take power from a battery pack, 12V, as needed to supply power to the electronics of the system.

Ultrasonic Detector (HC-SR04) - Interfacing the ultrasonic detector, with a range of 2cm to 400cm, to the microcontroller. The ultrasonic detector outputs a high signal whose duration is in relation to the distance of the obstacles. Since the sensor is mounted on a servo, measurements in its left and right directions are taken. The duration of the signals is to be converted to a distance.

Figure 2: Ultrasonic Detector (HC-SR04)

Figure 3: Microcontroller (PIC18F45Q10)

Microcontroller (PIC18F45Q10) – requires ADC, SPI, I^2C, PWM.

- The microcontroller gives control over the function of the motors. The speed and direction of the car can be controlled by specific commands sent to the motors by the controller.
- The microcontroller will calculate the obstacle's distance relative to the car with the ultrasonic detector. The motor will be turned on for some time to move some amount distance till its timeout to rescan forward or encounter an obstacle/wall. The side directions will not be rescanned if there is sufficient distance forward.
- The microcontroller will calculate the obstacle's distance relative to the car with the IR sensor to correct any deviation from the center of the path.
- The microprocessor will control the motion of the gear motors, LEDs and buzzer warning signals, servo motion, and LCD output.
- The microprocessor will find the best path for the car with the aid of the LHR algorithm and Dijkstra's algorithm.

LCD (CFAH1602Z-YYH-ET) – The display will show the estimated distance of the object in front and the running time of the car in the maze.

Figure 4: LCD (CFAH1602Z-YYH-ET)

Buzzer – An audio sound is to be played with a possible frontal collision is detected and a necessary rescan in all directions is needed to determine its next route direction.

LEDs – The car would have a green LED light up to a possible frontal collision is detected and a necessary rescan in all directions is needed to determine its next route direction. A red LED will light upon detecting the IR beacon and indicating the car is now running the Dijkstra's algorithm.

Dual H-Bridge (L298N) – A PCB designed DC motor driver to drive 2 gear motors to move the wheels forward and backward. It must also be able to rotate the car in place at 90° degrees in the left and right directions. The dual H-bridge motor driver also contains the voltage regulator to step down the battery-powered 12V to the usable 5V.

Servo (SG90) – Based on PWM, it will be able to rotate -90° and 90°. The ultrasonic sensor will mount on top of it allows it to scan the right and left front direction of the car to find the area with the most space.

Figure 5: Servo (SG90)

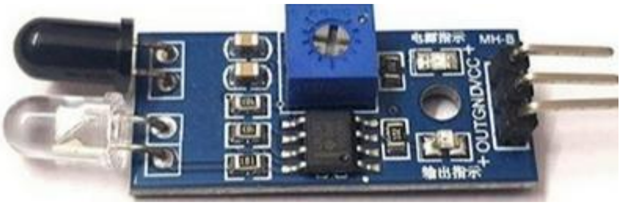


Figure 6: IR Detector (MH-B)

IR Detector (MH-B) – Interfacing the IR detector, with a range of 2cm to 30cm (depending on the obstacle's color) to the microcontroller. The IR detector outputs a low signal when an obstacle is detected. Four of these IR detectors, two on the right and two on the left of the car, are used to ensure the car maintain a center drive on its path, does no reeve to far to the left of right.

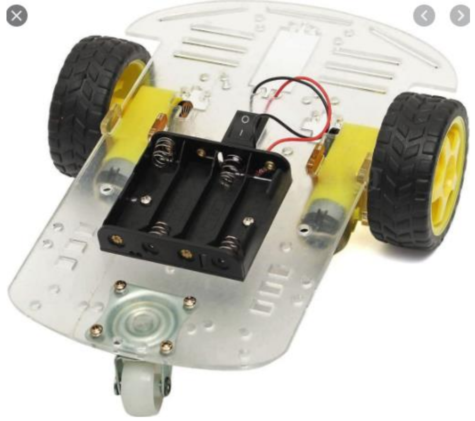
Timer (DS3231) – An accurate I^2C real-time clock with an integrated temperature-compensated crystal oscillator and crystal. The device incorporates a battery input and maintains accurate timekeeping when the main power to the device is interrupted.

LHR – A common maze solving algorithm that involves the car to follows the left wall throughout the maze as it attempts to find the exit.

Dijkstra's Algorithm – Calculates the smallest cost from one node to the next, then compares the source node/beginning location to all other nodes and finds the shortest length to follow.

Robot Car – The car is a simple kit composed of two DC motors with wheels, a ball caster, a rocket switch, and a chassis.

Figure 7: Robot Car



Competitive Products

The concept of a maze solving autonomous car is not new but is one that continuously seeks better designs and innovation. There have been many designs created, from the amateur robotic tinker to professional engineers developing self-driving cars to navigate the city streets, with various pros and cons. There are even competitions held for such designs based on navigational accuracy, completion time, etc. A quick google search on “maze solving car” shows some creative designs as seen below.

Figure 8: Competitive Products

Typical Features of Competitive Products:

- Fully Automatic Driving.
- Detectors on the front, left, and right faces of the car.
- Battery-powered.
- Forward, backward, right, and left driving motion.

Concept/Technology Selection

The original idea for the car was to mainly use infrared proximity. Infrared sensors suffer from the inability to use them in certain lighting conditions due to interference and their accuracy differs depending on the obstacle’s color. Ultrasonic sensors are, for the most part, completely insensitive to hindering factors like light, smoke, or dust. However, ultrasonic isn’t as good as infrared at defining the edges of an area. To compensate for the issues from either sensor, a combination of both sensors was used. The main sensor would be ultrasonic as it was more reliable for judging longer distances while IR sensors were mounted on the left and right front sides of the car to more accurately detect wall edges.

Various components from junior design will be reused since it is already on hand. A PIC microcontroller was chosen since I am already familiar with the family and have the means to program it. The following is a component list for the final design.

Hardware Selection:

Microcontroller (PIC18F45Q10): Qty.-1

The PIC18F45Q10 is the main controller used to governor the motion and action of the car. This microcontroller was chosen because of its availability and previous experience with the PIC device family. The PIC18F45Q10 provides a serial output port that will be useful for communicating with the LCD, the necessary PWMs to vary the speed of the motors as well as to modify the servo's position, and an ADC for the IR detector on the car. The PIC18F45Q10 also could handle I2C to communicate with the timer.

Twin-Motor Gearbox Kit/Chassis: Qty.-1

The robot car kit includes two motors, wheels and axles, a ball caster, a rocket switch, and the motor driver. Note that the give motor driver (L298N) was not used in the final design of the car, rather a similar PCB design was developed and used as its replacement. This kit was chosen because it contained the necessary bare-bone components which will allow me to better customize the car to fit the scope of the project as it develops. The motors will be used to drive the wheels to give motion to the car. The chassis will hold the motors, servo, battery, main PCB controller, PCB motor driver, the essential sensors, and later used to mount the electrical components. The car design is meant to be small and compact in order to have more options for maneuverability.

LCD (CFAH1602Z-YYH-ET): Qty.-1

The rectangular LCD will provide a visual interface to show the estimated distance of the object in front and the running time of the car in the maze. The LCD must be large enough to effectively display the text to the user but must also be simple enough to interface with the microcontroller. The CFAH1602Z-YYH-ET was chosen since it was already at hand from the junior design kit.

Timer (DS3231): Qty.-1

An external timer is used to keep track of the active run time. The device needs to incorporate a battery input and maintains accurate timekeeping when the main power to the device is interrupted or offline. The timer must be simple enough to interface with the microcontroller. The DS3231 was chosen since it was already at hand from the junior design kit.

Ultrasonic Detector (HC-SR04): Qty.-1

The primary sensor used to measure a distance where a void in a direction, a large distance gap, indicates an opening to the microcontroller. The sensor locates by sending out a ping that travels at the speed of sound. The time between the ping being sent and received can be used to calculate the distance between the car and the wall. Ultrasonic is more reliable than infrared when used for finding longer and more effective paths.

IR Detector (MH-B): Qty.-4

The secondary sensors to be used on the car to detect when the car comes in close contact with a wall or object. The IR detector transmits pulses of infrared light to be reflected back towards the sensor. When an object is present, there is an presence of IR light detected by the detector. The sensors we chose are the MH-B IR sensors and are positioned with one on the left, one on the front-left, one on the front-right, and one on the right area. The sensors are used to keep the car centered on its path when driving. These sensors were chosen as they were very cheap and simple to implement.

PCB Motor Driver (L298N): Qty.-1

Designed a PCB that mimics the function and usability of the L298N motor drive module to drive 2 DC motors to move the wheels of the car. The design takes in the positive and negative terminals of the battery and contains the voltage regulator to step down the battery-powered 12V to the usable 5V output to power the main PCB controller. Four terminals, two on the left and two on the right, are connected to the two motors on the left and right sides of the car. There are six control pins, one PWM pin for either motor to set the speed, connected to the microcontroller to control the rotation of the motors.

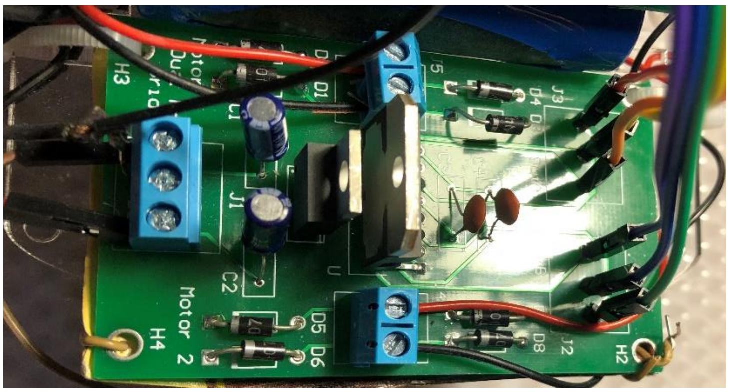


Figure 9: PCB Motor Driver (L298N)

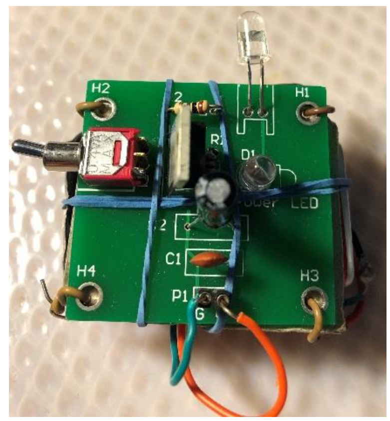


Figure 10: PCB Beacon

PCB IR Beacon: Qty. -2

Designed a PCB to act as a beacon for the end of the maze. The beacon consists of a voltage regulator to step down the attached battery to a usable 5V, a LED to indicate power, a toggle switch, and an IR emitter. Due to the difficultly of aligning the IR Beacon and the IR detector on the car, the beacon was handheld and manually positioned to trigger the detector.

<u>Software Selection:</u>

MPLAB X IDE:

Standard manufacturer software is used to program functions to the microcontroller. The software is based on the C-language and contains various features to assist in programming PIC devices including a debugging mode, status interface, and pre-packaged libraries. This IDE was chosen since it was already at hand from the junior design.

Altium Designer:

PCB design software is used to design, stylize, and configure the schematics or overlays for the PCBs. This software contains various features to assist in designing a PCB including a connection checker, status interface, and pre-packaged libraries. Several components were imported from various sources to fill in any missing from the software. The PCBs Gerber files were generated and sent offsite for manufacturing. This software was chosen since it was already at hand from the junior design.

Bill of Materials (BOM)

Senior Design - Bill of Materials				
Descripton	Quantity	Price/Part	Part Number	Price
Robot Car Chassis Kit	1	$23.33	2WD	$23.33
DC Gear Motors (1:48)	2			$0.00
Micro Servo Motor	1		SG90	$0.00
DC Motor Driver	1		L298N	$0.00
Ultrasonic Sensor	1		HC-SR04	$0.00
Speed Encoders	2			$0.00
Rocket Switch	1			$0.00
Wheels	3			$0.00
Battery Case	1		4x AA	$0.00
Rechargeable Battery 12V	1	$20.37	TS2738	$20.37
Battery Snap 9V	1	$0.49	232	$0.49
IR Proximity Sensor	4	$1.20	MH-B	$4.79
Photodiode	1	$3.42	BPW77NB	$3.42
IR Emitter	2	$0.75	OP293B	$1.50
8-bit Microcontrollers	1	$1.60	PIC18F45Q10-I/P	$1.60
Ceramic Capacitor - 0.1uF	5	$0.22	399-4264-ND	$1.10
Electrolytic Capacitors - 47uF	4	$0.24	493-17141-ND	$0.96
Resistor-10k Ohms- Through-Hole	2	$0.01	CF14JT10K0TR-ND	$0.02
Resistor -410 Ohms-Through-Hole	1	$0.09	CMF55410R00FKEB-ND	$0.09
Resistor -1k Ohms -Through-Hole	4	$0.10	1.0KEBK-ND	$0.40
Resistor -10 Ohms - Through-Hole	2	$0.10	MFR-25FBF52-10R	$0.20
Resistor-3.3k Ohms-Through-Hole	2	$0.31	3.3KZTR-ND	$0.62
Toggle Switch	2	$0.08	PTS647SM38SMTR2LFSTR-ND	$0.15
Potentiometer	1	$2.90	3310C-001-503L-ND	$2.90
Piezoelectric Buzzers	1	$0.20	490-4698-3-ND	$0.20
LED	4	$0.60	511-1264-ND	$2.40
LCD	1	$13.13	CFAH1602Z	$13.13
OpAmp	1	$7.97	LT1632CS8#PBF	$7.97
Timer	1	$2.99	DS3231S#-ND	$2.99
Voltage Regulator - 5V	3	$2.14	LM2940CT-5.0	$6.42
Terminal Block 3	1	$0.42	Screw Type 3	$0.42
Terminal Block 2	2	$0.32	Screw Type 2	$0.64
General Purpose Diode	8	$0.09	1N4007	$0.72
Dual H-Bridge Driver	1	$9.72	L298N	$9.72
Assorted wire connectors	1	$6.29	M to M, F to F, M to F	$6.29
			Total	$112.85

Figure 11: Bill of Materials (BOM)

Breaking of the Work

This project is a single-person project and all the work, including software development and hardware engineering, was entirely done by me with the support of my advisory engineer, teaching assistants, and professors. I hold all the responsibility and am accountable for project completion.

Gantt Chart/Project Schedule

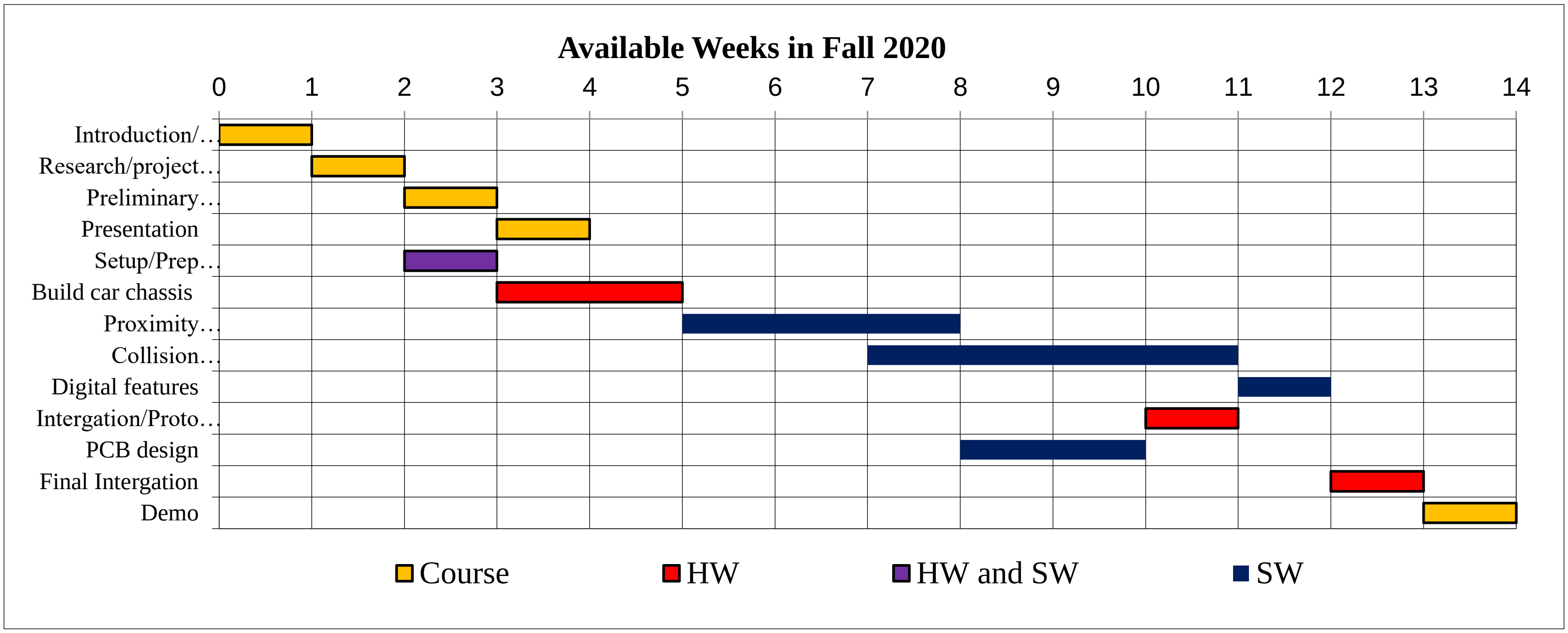


Figure 12: Gantt Chart/Project Schedule

Project Architecture

Upon powering the car, the microcontroller will begin initializing the necessary setup codes. The servo will be set to the forward-facing position and the timer will start counting from zero. Initial forward measurement is taken, assuming there is an opening in front, the car will begin to drive forward. If the car goes too close to the left or right walls at any point, the IR detectors will trigger an interrupt on change upon detecting the walls and correct the car's motion to position it back to the center of the path. This is done by having the microcontroller send control signals to the PCB motor driver, to stop one wheel while having the other wheel continue rotating, to turn the car in its respective direction to avoid a collision with the wall. The microcontroller will continuously send updates to the LCD to have the recent time and distance from the last scan be available to the user.

The car will occasionally rescan the front to ensure that the path is free from obstacles. If an obstacle is encountered in front of the car, then the buzzer will go off and the green LED will turn on. The servo will move to face to the left and then right to let the ultrasonic detector to scan the left and right directions respectively. If there is a single opening, the car will turn in the direction of the opening and continue its normal operation of driving forward. If there is no opening, the car will do a U-turn to reverse its direction and continue to its normal operation of driving forward. If there are two openings, then based on the LHR, the car will take the left path if possible. Every intersection that the car encountered, and the path is taken is recorded in a memory array in the microcontroller. A U-turn erases the "dead-end" path from the memory array.

When the car reaches the end of the maze, a person shall attempt to shine the IR beacon toward the car. This is quite difficult as the light emitted is invisible to humans and aligning the two IR beacon onto the IR detector when the car is actively moving does take some skill. When the IR beacon has been detected, a red LED will turn on follow by the car taking the shortest path back to the entrance of the maze. The process is similar to when the car was finding the exit except the paths is pre-calculated.

Design Procedure Used

Each component of the robot car is supported by the chassis, where the structure on which the PCB main controller, PCB motor driver, batteries, motors, servo, and sensors lie. Specifically, the base of each components (beside the PCB main controller) of the car is mounted on the acrylic chassis in some way. The mounting method includes screws and nuts, zip ties, and electrical wires used as tiedown. The motors are mounted to the frontal bottom layer of the chassis with the wheels attached motor on the sides of the chassis. This allows the wheels to stay stable while moving. In order to keep the robot balanced, there is a ball caster on the bottom rear of the chassis. The rocket switch is placed faced up on the center of the chassis.

The ultrasonic sensor is at the head center of the car to detect the path and walls ahead of it effectively. The IR sensors are placed front of the wheel to detect the walls to the left and right of the robot effectively, and their placing maximizes the amount of space for the servo and wires. The sensors are positioned with one on the left, one on the front-left, one on the front-right, and one on the right area. The PCB motor driver is placed on the top rear of the chassis and the battery on the top middle. The PCB main controller is mounted on top of the battery. The wires are placed through the holes on the chassis, keeping all the electronics neat.

In regard to the processing and power components, the battery is connected to the PCB motor driver and the output from the motor driver is connected to the PCB main controller. The battery provides 12V and the motor driver regulates an output of 5V which runs the components in the main controller. The 12V from the battery also power the motors, where the motor driver allows the polarity of DC motors to be switched, adding forward and backward driving capabilities. The main controller is exposed so that, in case of a problem in the wiring, a change can be made without disassembling the entire car.

In terms of software, assuming the codes work as intended, begin initializing the necessary setup codes involving ADC for the IR detector on the car, I2C for the timer, SPI for the LCD, and PWMs for the servo and speed control. Afterward, run the codes for the noted vehicle actions as stated above recursively. The code is set up in a recursive loop, so it lessens the burden on the memory and its complexity. For instance, the car scans then decide on a path to move and repeat. When the IR detector is triggered by the beacon, the car should reverse and directly take the shortest path to the entrance. For example, go forwards, turn left, turn right, etc.

The algorithm used was programmed into the microcontroller and used a memory array to keep track of the routes taken in the maze. The LHR is a common maze solving algorithm where the robot follows the left wall throughout the maze, possibly encircling the entire maze as it attempts to find the exit. Dijkstra's algorithm compares the "source" node to all other nodes in the graph and finds the shortest length to follow.

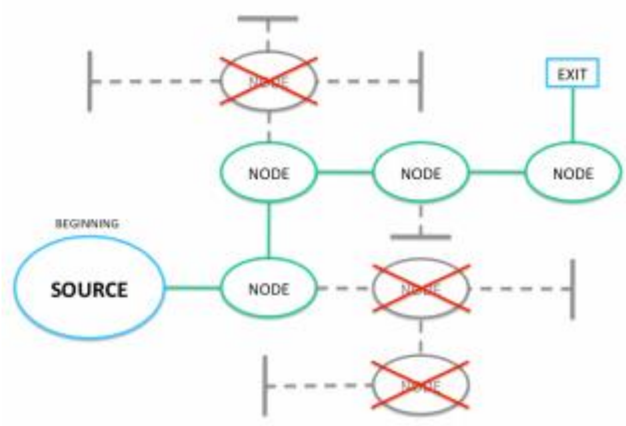


Figure 13: Dijkstra's Algorithm Pathing

<u>Flowcharts & Diagrams</u>

The figure below shows a diagram of the components integrated into the system.

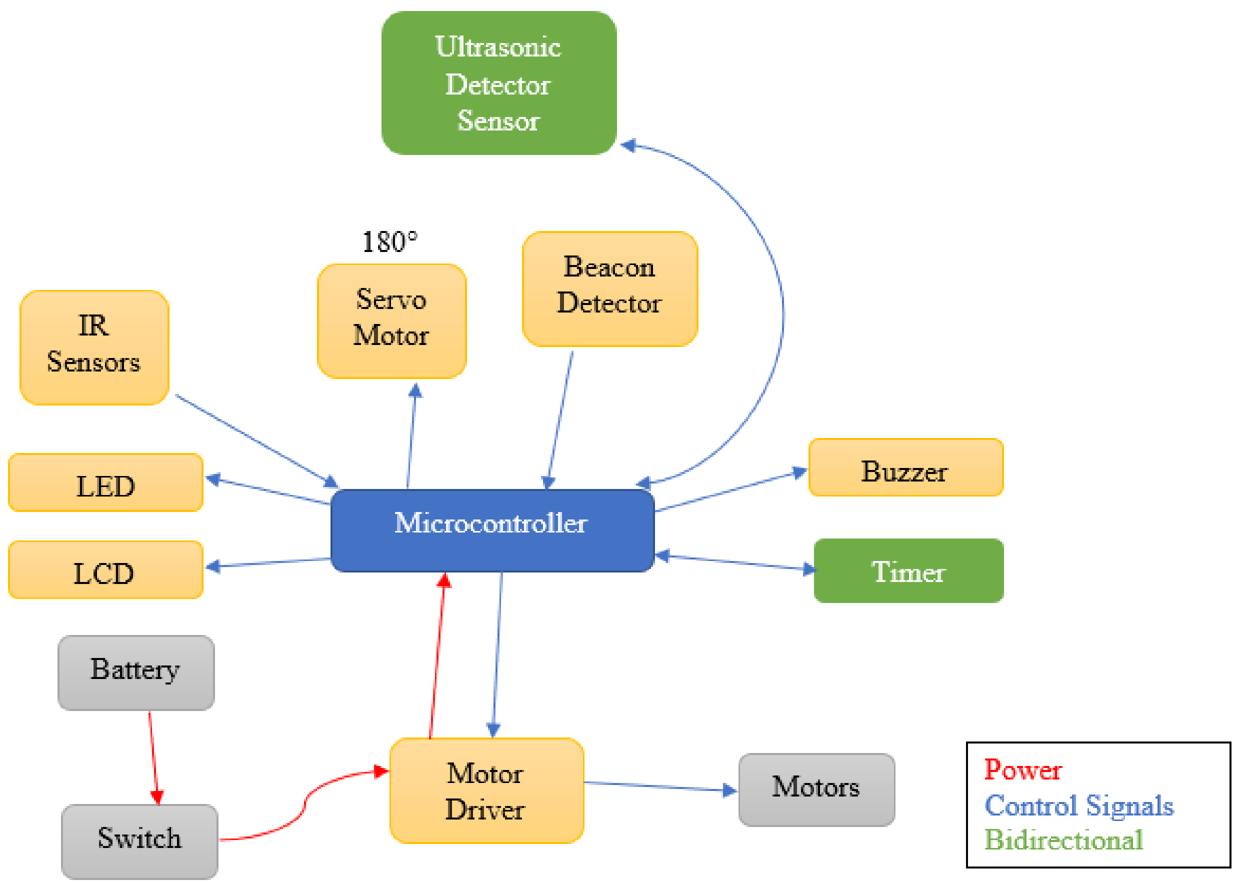
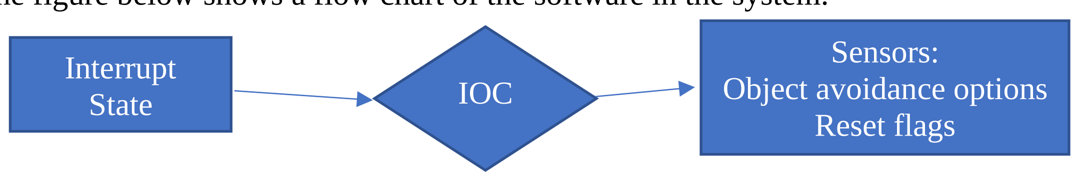


Figure 14: Car Controller Diagram

The figure below shows a flow chart of the software in the system.



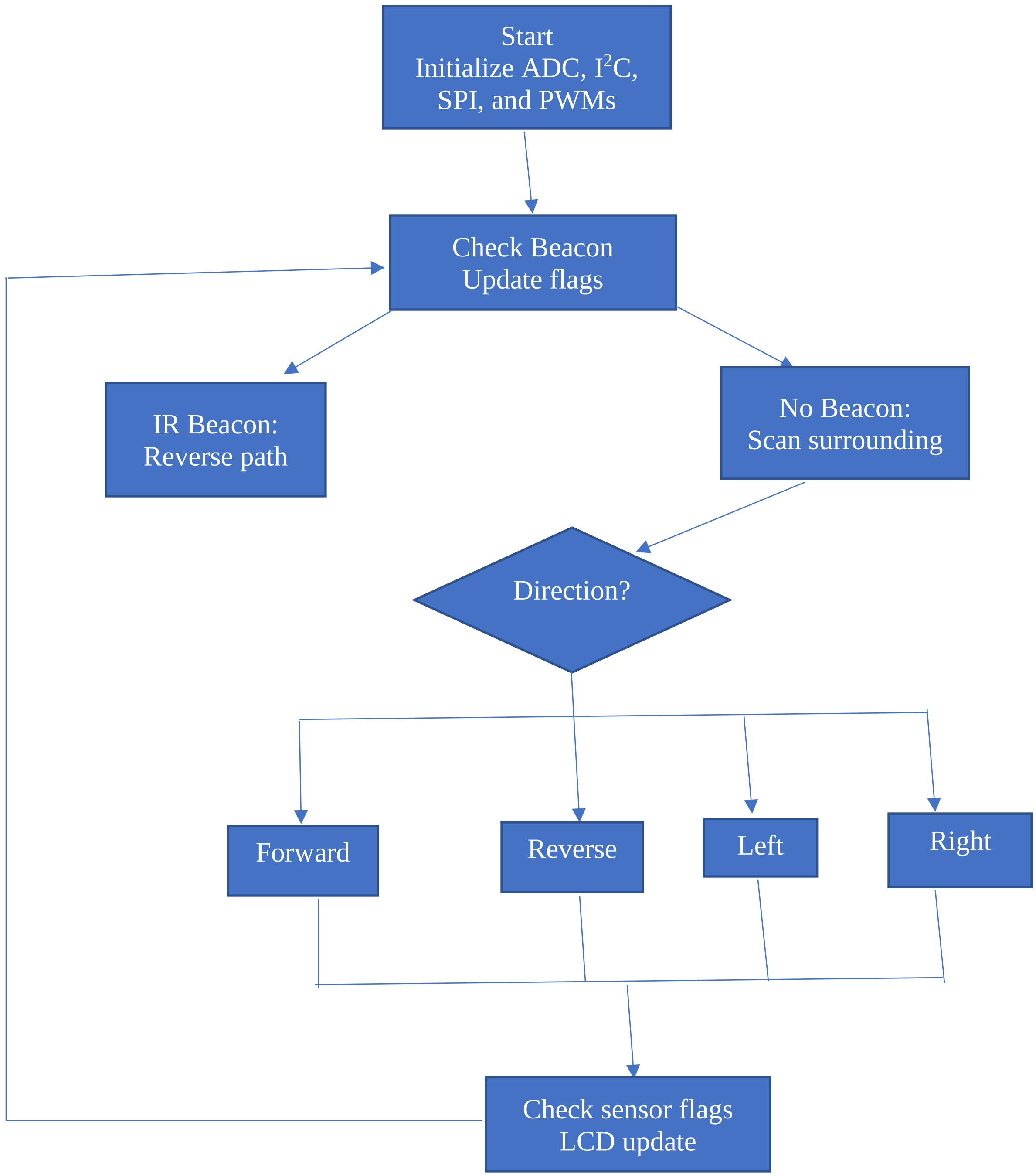


Figure 15: Car Controller Flowchart

Obstacles Overcome

There were many obstacles to overcome in order to get the project running successfully. A list of some is given below:

- There was a lack of complexity in the project which was resolved by designing my own motor driver rather than using an L298N module.
- The servo drew much more current than expected, causing a sudden drop in voltage and resulting in a brownout. This was solved by upgrading to a higher voltage and current battery.
- Due to the difficulty of aligning a beacon and the IR detector on the car, a second beacon was constructed to improve exposure.
- To improve traction, rubber bands were strapped onto the wheels. This helped with the car driving ability.
- The car was originally planned to two only have 2 IR sensors, one on each side, but the was increased to 4 IR sensors, two on each side, to improve the reliability of the proximity detection.
- Because of the increased pins needed for the new IR sensors, the original PCB main controller no longer meets the necessary pin placement, so some rewiring had to be done.

User Manual

Start the car and place it at the entrance of the maze. The car should automatically begin to find its way to the exit. Upon reaching the exit, shine the beacon on the car and ensure that the beacon triggered the return to the entrance via the shortest path.

Appendices

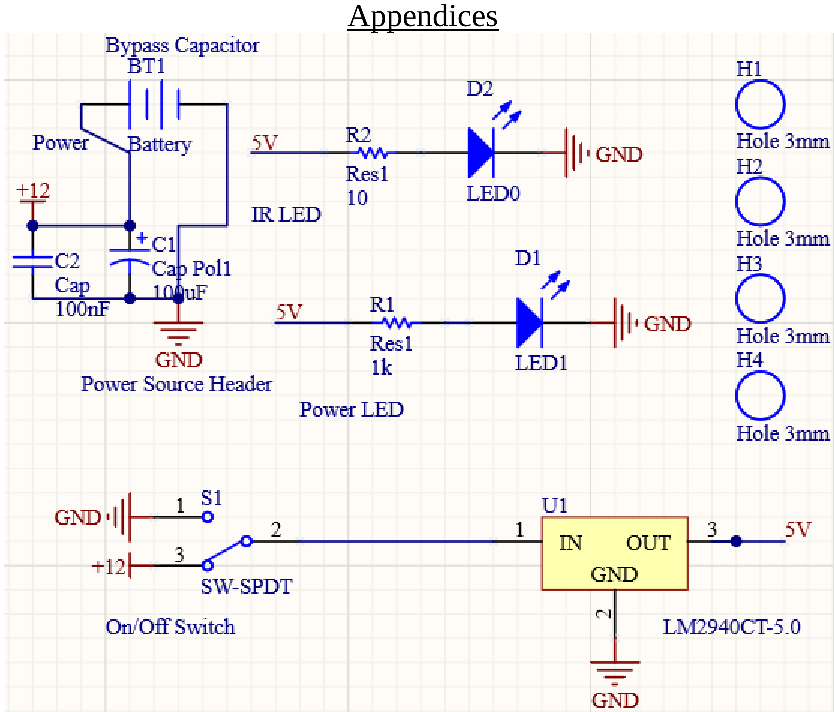


Figure 16: Beacon schematic

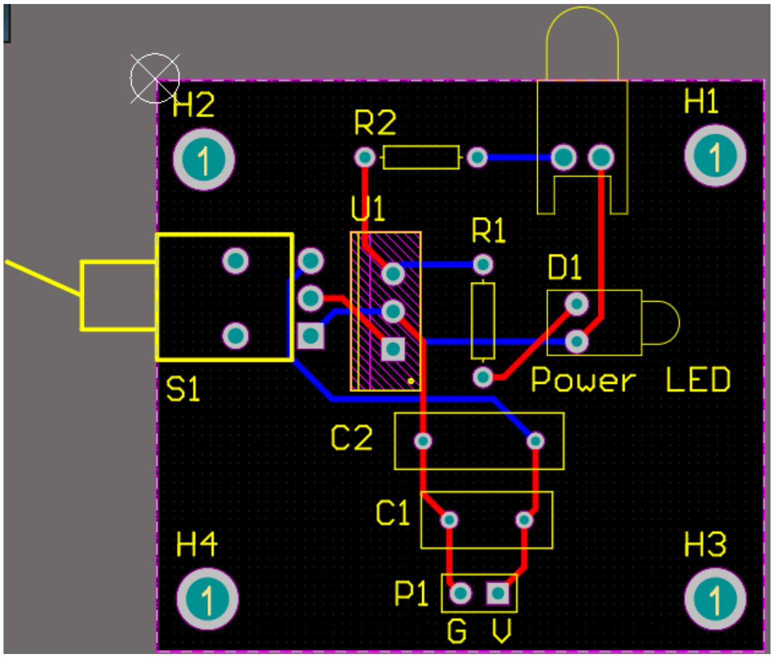


Figure 17: PCB Beacon

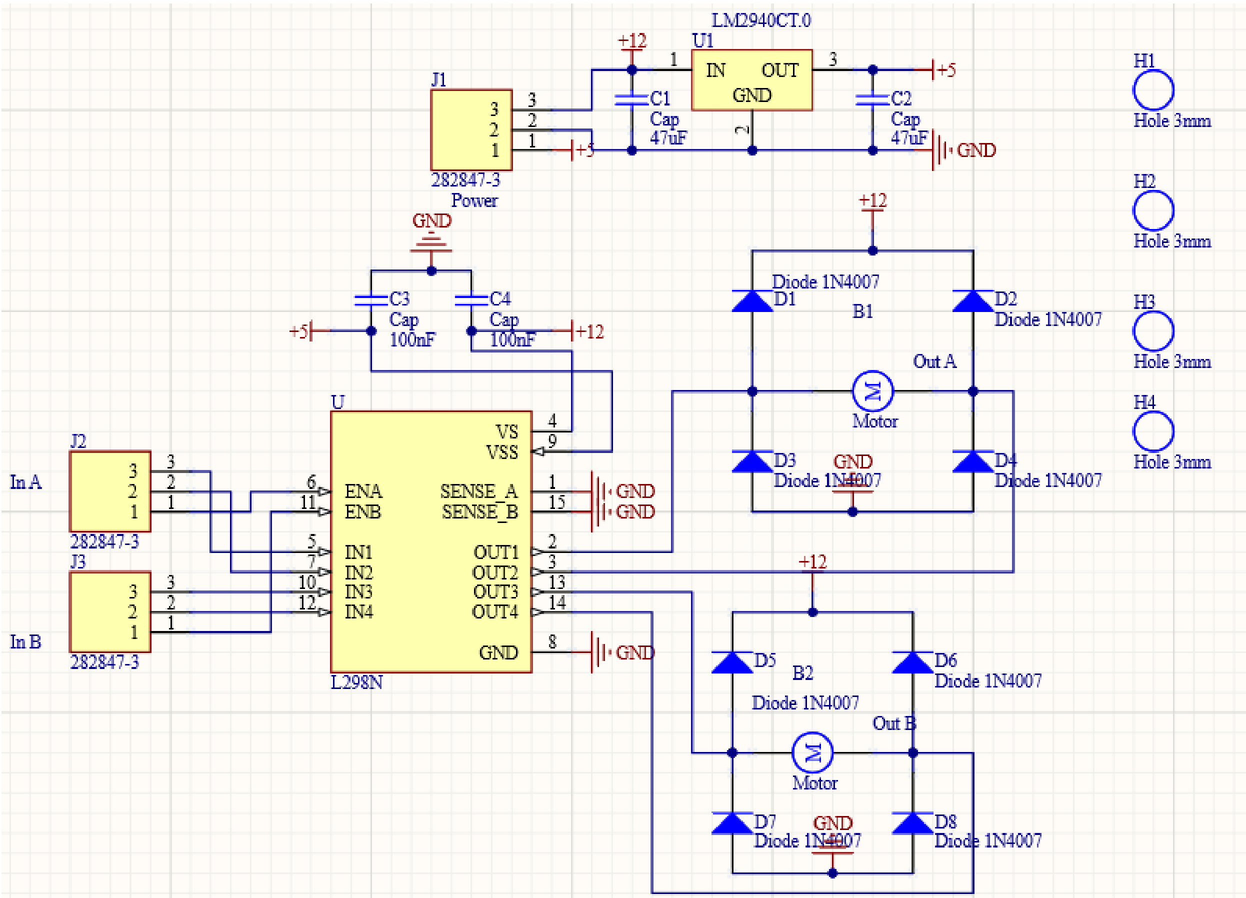


Figure 18: Motor Driver schematic

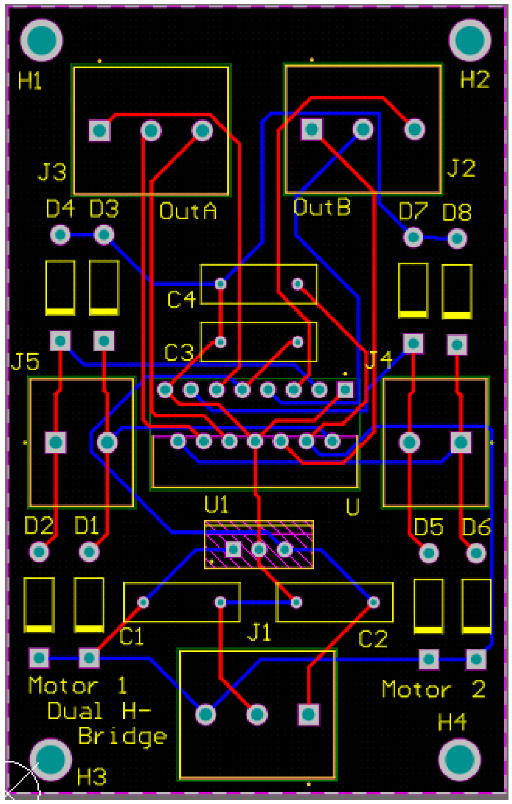


Figure 19: PCB Motor Driver

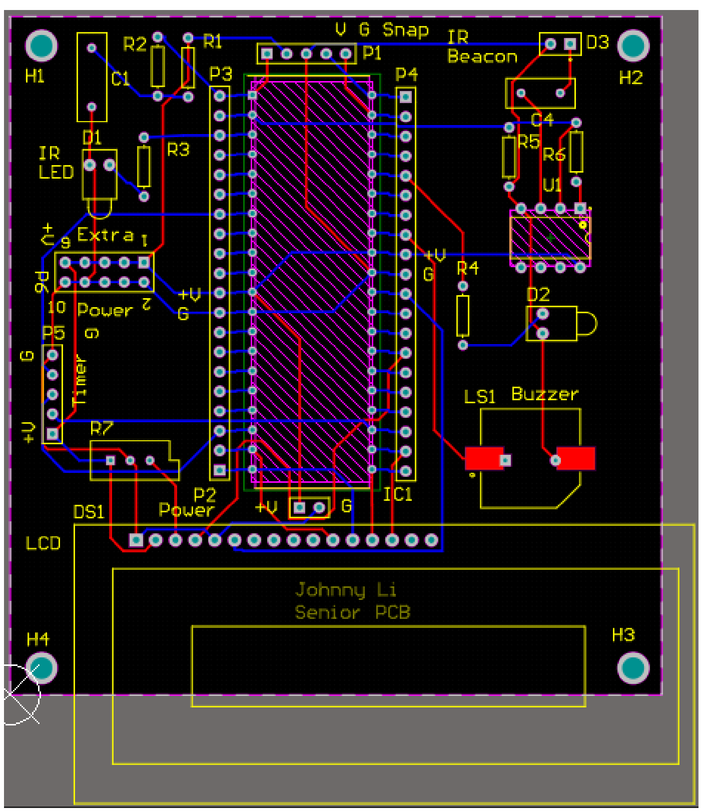


Figure 20: PCB Car Controller

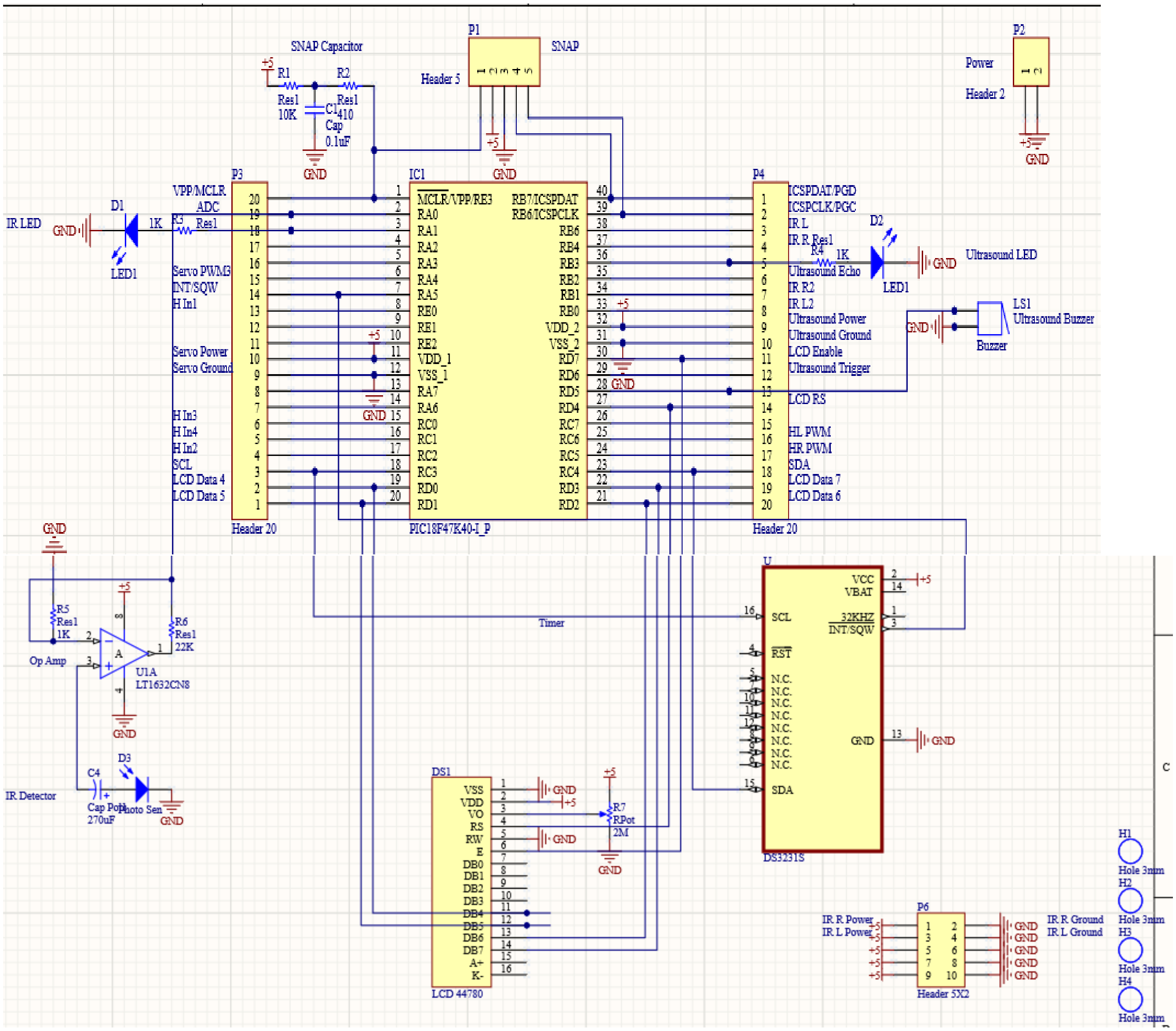


Figure 21: Car Controller schematic

References

"Maze Solving Algorithm." *Wikipedia*, Wikimedia Foundation, 1 Oct. 2020, en.wikipedia.org/wiki/Maze_solving_algorithm.