Advanced Smart and Safe Car Using Object Detection, Alcohol Detection and Safe Parking

A project report submitted in partial fulfilment of the requirement for *the award of Degree of B. Tech*

COMPUTER SCIENCE ENGINEERING

Submitted by,

M. NAVEEN - 20NT1A0565

D V V AKHIL - 21NT5A0541

P. SUSHMA – 20NT1A0577

V. DINISHA REDDY- 20NT1A0525

M. SAI KUMAR – 20NT1A0559

Under the guidance of,

M. USHA

VISAKHA INSTITUTE OF ENGINEERING AND TECHNOLOGY

(Approved by AICTE NEW DELHI, Affiliated to JNTU-VIZIANAGARAM)

57TH Division, Narava, GVMC, Visakhapatnam-530027 A P

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CERTIFICATE

This is to certify that this project report is a bonafide work of, M. Naveen(20NT1A0565), D V V Akhil(21NT5A0541), P. Sushma(20NT1A0577), V. Dinisha Reddy(20NT1A0525), M. Sai Kumar(20NT1A0559) who have done the project which entitled "**Advanced Smart and Safe Car Using Object Detection, Alcohol Detection and Safe Parking"** from --------- to --------

Project Guide,
M. USHA

Head of the Department,
ASN TEJASWINI KONE

Submitted for Viva voice Examination held on ---------

Internal Examiner

External Examiner

DECLARATION

We,

M. Naveen(20NT1A0565), D V V Akhil(21NT5A0541),
P. Sushma(20NT1A0577), V. Dinisha Reddy(20NT1A0525),
M. SaiKumar(20NT1A0559) the students of B. Tech Program of the Department of Computer Science and Engineering, Visakha Institute of Engineering and Technology do hereby declare that we have completed the mandatory project work underthe Faculty Guideship of M. Usha, Department of Computer Science andEngineering in Visakha Institute of Engineering and Technology.

(Signature and Date)

Endorsements

Faculty Guide

Head of the Department

Principal

ACKNOWLEDGEMENT

It gives me immense pleasure to acknowledge all those who helped me throughout in making project of great success.

With profound gratitude I thank **Prof. V. SRIDHAR PATNAIK, MTech, Ph.D.,** Principal Sir, Visakha Institute of Engineering and Technology, for his timely suggestions which helped me to complete this project work successfully.

Our sincere thanks and deep sense of gratitude to **ASN TEJASWINI KONE,Mam**, Head of the Department CSE, for her valuable guidance, in completion of this project successfully.

I also thank our guide **M. USHA Mam**, for her guidance, in completion of this project successfully.

I am thankful to both Teaching and Non-Teaching staff of CSE department for their kind cooperation and all sorts of help bringing out this project work successfully.

With Gratitude,

M. NAVEEN - 20NT1A0565
D V V AKHIL - 21NT5A0541
P. SUSHMA – 20NT1A0577
V. DINISHA REDDY- 20NT1A0525
M. SAI KUMAR – 20NT1A0559

INDEX

Chapter-1 The Introduction

Chapter-2 Contextual Exploration

Chapter-3 Connecting the dots: From customer needs to project outcomes in real time

Chapter-4 The Algorithmic Constructs Used

Chapter-5 Insights Of Raspberry Pi

Chapter-6 Sensors and Data Collection Devices

Chapter-1 The Introduction

1.1 ABSTRACT

Nowadays, we are facing many car accidents as the objects are not visible to us while driving and parking also when the drivers are driving the car when they are drunk. So, we came up with a solution for that. We created a safer, more convenient driving experience using a Raspberry Pi and also advanced tech like IoT and AI. A backup camera will be attached to the car. Using advanced computer vision techniques, the system identifies the objects within the environment and provides an immediate visual and textual feedback to the users. Whereas for parking the car, 4 sensors will be placed at 4 ends of the car. The parking is done according to the distance calculation between the other cars or any other objects in the parking areas. If any car or object is detected, there will be a textual representation of that particular object on the screen so that the driver ensures safety while parking. A chip will be placed on the car's steering. Once the driver enters the car, the chip generates based in the car if the driver consumed alcohol. The abstract highlights of the system's ability is to bridge the gap between physical objects and digital interfaces through real-time object detection, leveraging sensor-based technology for intelligent car parking, alcohol detection and also enhancing user convenience.

1.2 INTRODUCTION

This project is to ensure safe driving of a car. The features in this include:

- Alcohol Detection
- Object Detection
- Safe Parking

1.2.1 Alcohol Detection

The main purpose of the Alcohol Detection with Vehicle Controlling project is "Drunk driving detection". Nowadays, many accidents are happening because of the alcohol consumption of the driver or the person who is driving the vehicle. Thus, Drunk driving is a major reason for accidents in almost all countries all over the world. The alcohol Detector in Car project is designed for the safety of the people seating inside the car. The alcohol Detection with Vehicle Controlling project helps to control the vehicle in case the driver has consumed alcohol. An alcohol breath analyzer project should be fitted/ installed inside the vehicle.

1.2.2 Object Detection

The main task of autonomous driving is to accurately and quickly detect the vehicles, pedestrians, traffic lights, traffic signs, and other objects around the vehicles, in order to ensure the safety in driving.

1.2.3 Safe Parking

Due to the surge in urbanization, people don’t depend on public vehicles. They use their vehicles to travel. So, traffic increases. When people travel through a city the most difficult problem is to park the vehicle. It causes not only a waste of time and fuel for drivers looking for parking, but it also leads to additional waste of time and fuel for other drivers as a result of traffic congestion. The usage of automobiles has increased which in turn has led to traffic and parking difficulties. The most important thing is safe parking where we can't see the sides through camera perfectly so it is important to use safe parking mechanism.

1.3 PROJECT DEFINITION

The "Smart and Safe Car with Object Detection and Alcohol Detection" project aims to enhance the safety and intelligence of a vehicle by incorporating advanced technologies for object detection and alcohol impairment detection. This project will utilize a combination of hardware and software components to achieve these objectives, ensuring the well-being of both the driver and others on the road.

1.4 OBJECTIVE

1.4.1 For Object Detection

- Implement a real-time object detection system using computer vision and deep learning techniques.
- Utilize cameras and sensors to detect and identify obstacles, pedestrians, and other vehicles.
- Provide visual and auditory alerts to the driver when potential collisions or hazards are detected.

1.4.2 For Alcohol Detection

- Integrate an alcohol detection system, such as a breathalyzer or other nonintrusive technologies, into the car's interior.
- Continuously monitor the driver's alcohol levels to ensure they are below the legal limit.
- Disable the vehicle's ignition and alert authorities if alcohol impairment is detected, preventing drunk driving.

1.4.3 For Safe Parking

- Implementing safe parking system using ir and ultrasonic sensors
- It continuously monitors the obstacles that are being passed and distance between them. • Therefore, it provides a safe parking of the car.

1.5 FEATURES OF THE PROJECT

1.5.1. Object Detection Features

- ***Sensors:*** Equipped with advanced LiDAR, radar, and camera systems for real-time object detection.
- ***Algorithms:*** Utilizes machine learning and computer vision algorithms to identify pedestrians, vehicles, obstacles, and road conditions.

- ***Driver Alerts:*** Provides visual and audible alerts to the driver when potential hazards are detected.

- ***Emergency Braking:*** Can autonomously engage emergency braking to prevent collisions if the driver doesn't react in time.

1.5.2. Alcohol Detection Features

- ***Breathalyzer or Ignition Interlock:*** Incorporates a breathalyzer or ignition interlock system to detect alcohol in the driver's breath.

- ***Engine Lock:*** Prevents the vehicle from starting if the driver's blood alcohol concentration (BAC) exceeds legal limits.

- ***Data Logging:*** Records alcohol detection results for later review or law enforcement purposes.

1.5.3. Safe Parking Features

- ***Parking Assistance:*** Offers automated parking assistance for parallel parking, perpendicular parking, and other maneuvers.

- ***Parking Space Detection:*** Utilizes sensors to identify available parking spaces and provides real-time data to the driver.

- ***Safety Features:*** Activates hazard lights or alerts when it detects potential parking-related obstacles or pedestrians.

- ***Mobile App Integration:*** Allows drivers to reserve parking spots in advance and receive guidance to the chosen spot.

1.6 BACKGROUND PROBLEM

In recent times, the roads have witnessed a disturbing increase in accidents primarily attributed to two critical factors: alcohol-impaired driving and the presence of unseen objects behind vehicles. Accidents caused by intoxicated drivers, who exhibit impaired judgment and slowed reaction times, pose a significant threat to road safety. These incidents result in injuries, fatalities, and immeasurable emotional distress for the victims and their families. Simultaneously, the issue of unseen objects behind vehicles, particularly in blind spots, is contributing to accidents during parking and reversing

maneuvers. The combination of these problems highlights the pressing need for advanced technological solutions in smart and safe cars that can detect alcohol impairment and provide object detection systems and also allows smart parking that avoids accidents.

1.7 CONNECTIONS

1.7.1 Alcohol Sensor Connection

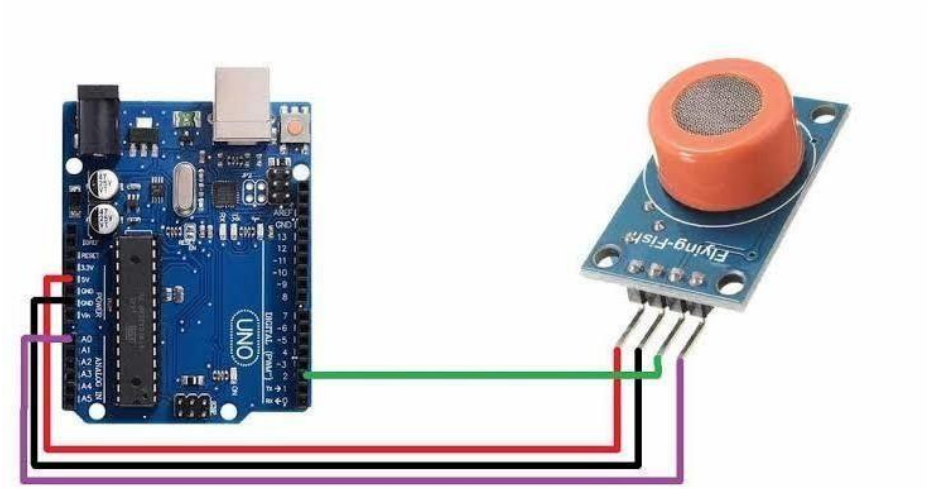


Figure 1.1: Alcohol Sensor Connection

To connect an alcohol sensor to a Raspberry Pi,

- Use GPIO pins on the Raspberry Pi, with digital sensors, or ADC pins for analog sensors.
- Connect the sensor's output to a suitable GPIO or ADC pin on the Raspberry Pi.
- Provide the sensor with required power (usually +5V) and ground (GND) connections.

1.7.2 Ultrasonic Sensor Connection

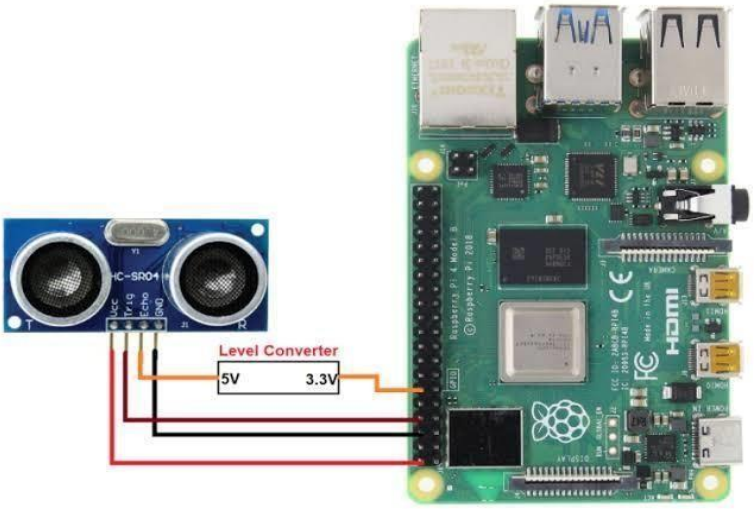


Figure 1.2: Ultrasonic Sensor Connection

To connect an ultrasonic sensor to a Raspberry Pi,

- Connect the VCC (power) pin of the ultrasonic sensor to a 5V pin on the Raspberry Pi.
- Connect the GND (ground) pin of the sensor to a ground pin on the Pi.
- Connect the Trig (trigger) pin of the sensor to a GPIO pin on the Pi (e.g., GPIO23).
- Connect the Echo pin of the sensor to another GPIO pin on the Pi (e.g., GPIO24).
- Use GPIO control and programming (e.g., Python) to read data from the sensor and measure distances

1.7.3 IR Sensor Connection

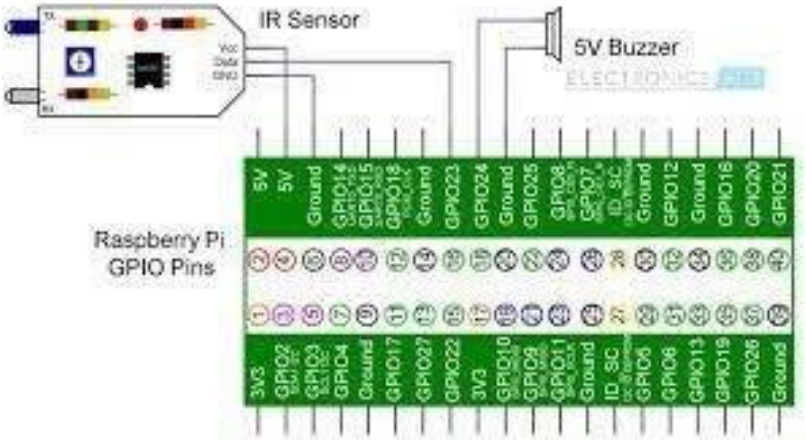


Figure 1.3: IR Sensor Connection

Connecting an IR sensor to a Raspberry Pi is straightforward,

- Identify the IR sensor's pins: Typically, it has three pins (VCC, GND, and OUT).
- Connect the VCC pin to a 3.3V GPIO pin on the Raspberry Pi, GND to a ground pin, and OUT to a GPIO pin.

1.7.4 Camera Connection

Figure 1.4: Camera Connection

Connecting a Raspberry Pi camera is straightforward, • Locate the camera connector on the board (near the HDMI port).

- Insert the ribbon cable from the camera module into the connector, making sure it's properly aligned.

1.7.5 Buzzer Connection

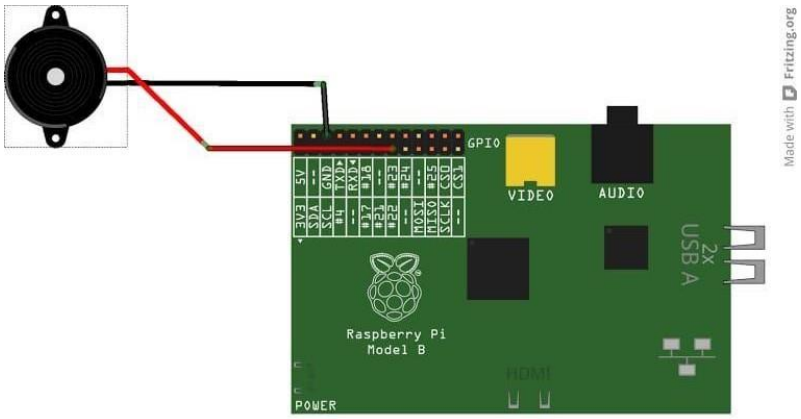


Figure 1.5: Buzzer Connection

To connect a buzzer to a Raspberry Pi

- Connect the buzzer's positive (red) wire to a GPIO pin (e.g., GPIO17) on the Raspberry Pi.
- Connect the buzzer's negative (black) wire to a ground (GND) pin on the Raspberry Pi.

1.7.6 Driver Connection

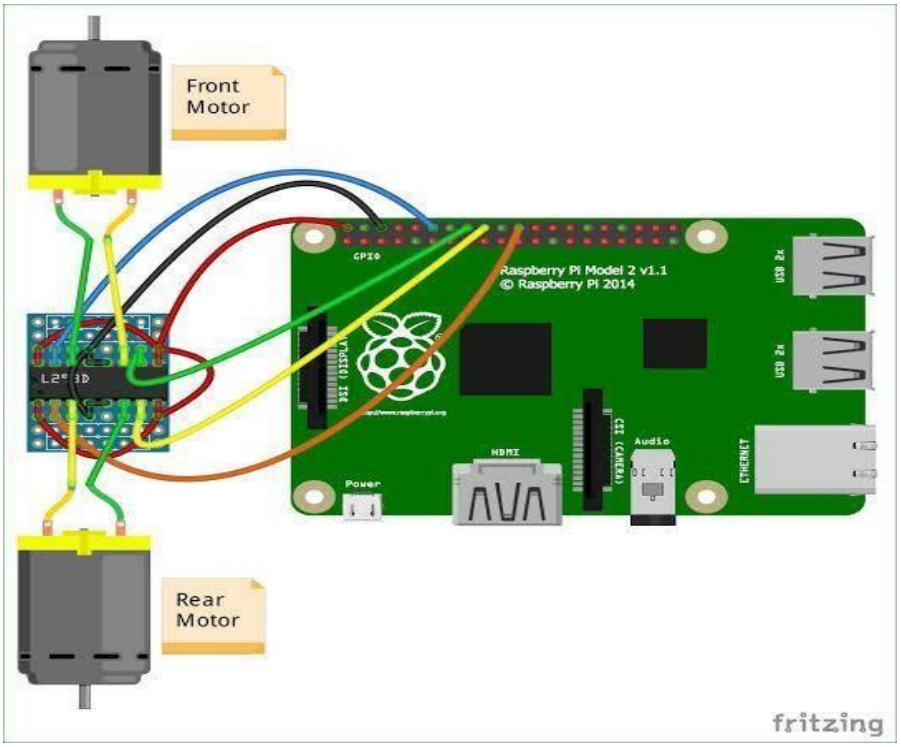


Figure 1.6: L293D Driver Connection

In a Raspberry Pi car project, connecting the driver (motor controller) and wheels (motors) is a critical part of building a functional robotic or remote-controlled vehicle.

- *Motor Controller (Driver)*

The motor controller, often an H-bridge or motor driver board, is used to control the direction and speed of the motors. It receives commands from the Raspberry Pi to drive the vehicle.

- *Motors*

You'll have two motors, one for each wheel. These are responsible for driving the car's movement.

- *Power Supply*

The motors typically require a separate power supply, as they draw more current than the Raspberry Pi can provide. This power supply can be a battery pack or an external power source.

- *Raspberry Pi GPIO Pins*

The Raspberry Pi communicates with the motor controller through its GPIO pins. You'll use specific GPIO pins to send signals to the motor controller for controlling the motors' direction and speed.

1.8 BLOCK DIAGRAM

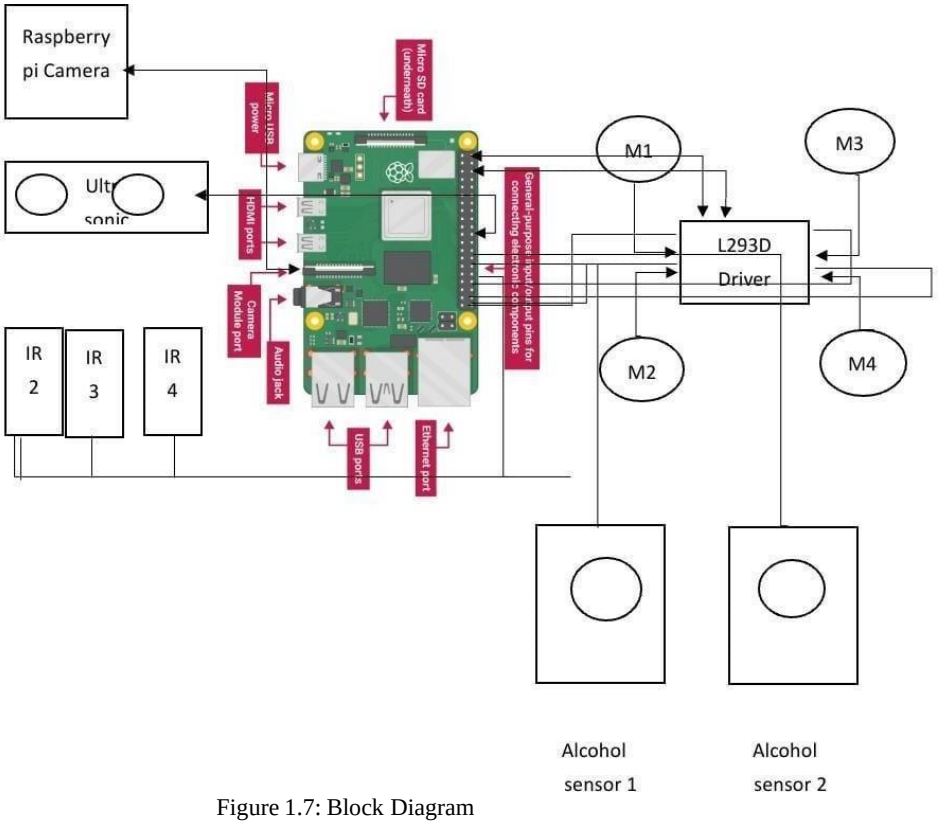
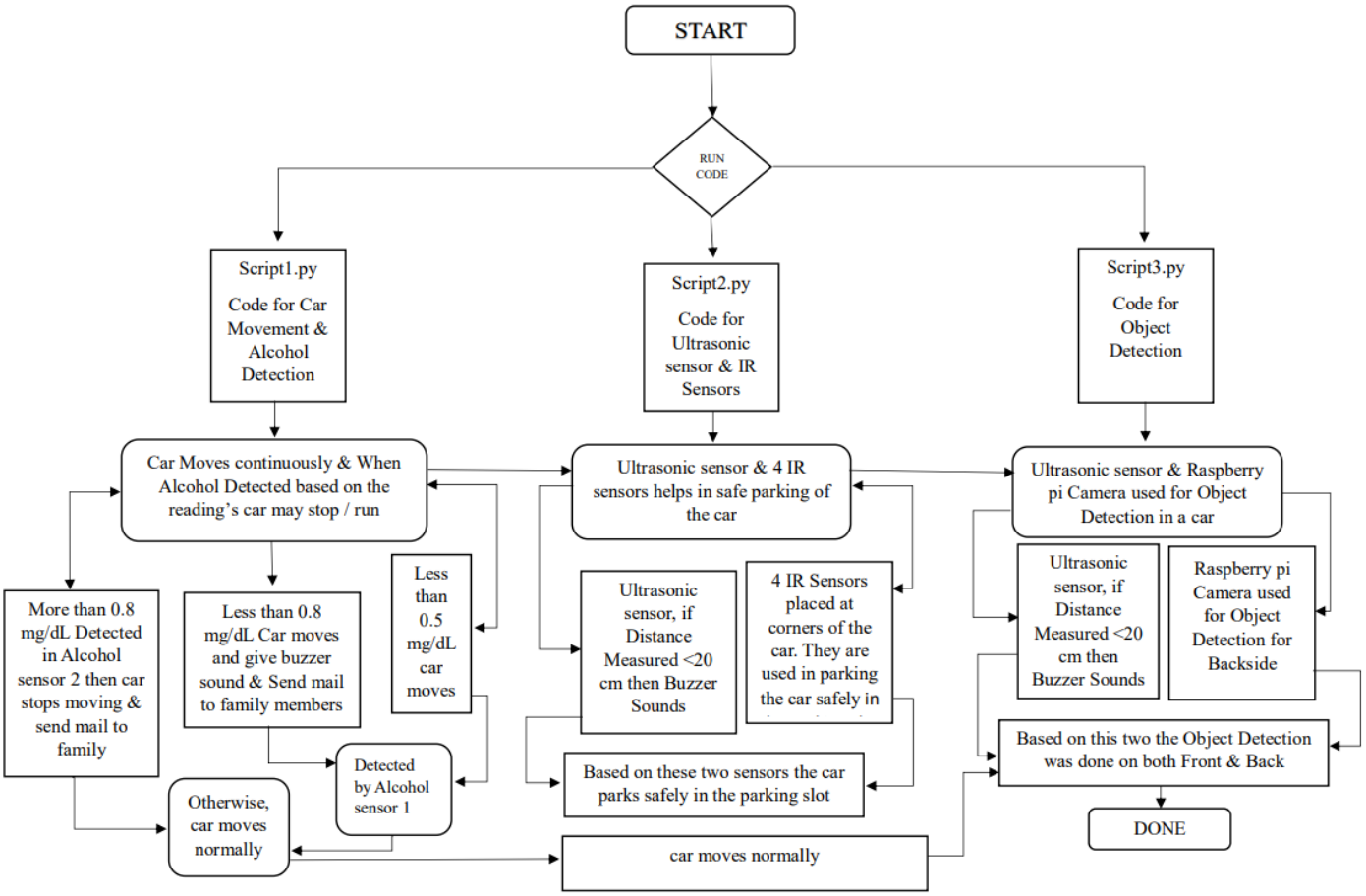


Figure 1.7: Block Diagram

1.9 FLOW CHART



Chapter-2 Contextual Exploration

2.1 LITERATURE REVIEW

1. The authors of the reference paper are ***Zhong*-Qiu Zhao, Peng Zheng.** The paper is titled "**Object Detection with Deep Learning".**

In reviewing this paper, it is evident from the literature provided that deep learning has emerged as a transformative and groundbreaking initiative in the field of artificial intelligence and machine learning. This initiative signifies a profound shift in our approach to complex problems, particularly those involving pattern recognition. Deep learning methods, notably neural networks, have showcased remarkable capabilities in tasks such as image and speech recognition, as well as autonomous decision-making. The continuous development and refinement of deep learning algorithms, coupled with advances in hardware and data availability, are propelling this initiative forward. It holds great promise for addressing a wide range of challenges and unlocking new possibilities in fields like healthcare, finance, autonomous vehicles, and beyond.

2. The authors of the reference paper are **Pranita Jadhav, Vrushali Koli, Priyanka Shinde, and Dr. M.M. Pawar**. The paper is titled "**Object Detection Using Deep Learning.**"

A review of this paper, titled 'Object Detection Using Deep Learning,' has been instrumental in shaping my understanding of object detection techniques. The book provides comprehensive insights into the fundamental concepts and methodologies of object detection, laying a strong foundation for my research in this field. I extend my gratitude to the authors for their significant and well-structured work, which has greatly informed and guided my study.

3. The authors of the reference paper are **Hironori Wakana, Masuyoshi Yamada, and Minoru Sakairi**. The paper is titled "**Portable Alcohol Detection System with Breath-Recognition Function**."

I would like to express my appreciation for the book 'Alcohol Detection Technologies and Applications.' This comprehensive work has played a pivotal role in shaping my understanding of alcohol detection systems and technologies. The insights and research findings presented in this book have been instrumental in guiding my own research into the development of a Portable Alcohol Detection System with Breath-Recognition Function. I am indebted to the authors for their substantial contributions to this field, which have been a significant influence on my work.

4. **The authors of the reference paper are** Chungsan Lee, Youngtak Han, and Soobin Jeon. **The paper is titled** "**Smart Parking System for Internet of Things**."

Our review of the literature presented in this paper has provided valuable insights into the methods and significance of implementing smart parking systems. We now comprehend the techniques involved in smart parking and the reasons behind its adoption. The knowledge gained from this literature has proven instrumental in shaping our project, as we have successfully incorporated some of the key qualities identified in the paper. We extend our appreciation to the authors for their contribution, as their work has significantly influenced and enriched our project in the realm of smart parking systems for the Internet of Things.

5. The authors of the reference paper are **Cheng-Hsiung Hsieh, Dung-Ching Lin, and Cheng-Jia Wang.** The paper is titled "**Real-Time Car Detection and Driving Safety Alarm System with Google TensorFlow Object Detection.**"

In reviewing the content of this literature, we have gained valuable insights into the algorithms employed for object detection and their crucial role in smart car technology. The real-time car and object detection methodologies discussed in the paper have been particularly enlightening. We express our appreciation to the authors for their significant contribution to this project, which has served to elevate our exploration of object detection within our own work. Their insights have played a pivotal role in enhancing the depth and relevance of our project in this field

6. The authors of the reference paper are **Shahad Al-Yousif, Musab A. M. Ali, and M. N. Mohammed**. The paper is titled **"Alcohol Detection for Car Locking System."**

Our review of "Alcohol Detection for Car Locking" underscores the critical and potentially lifesaving nature of this feature, which has become a widespread integration in modern vehicles to address the issue of drunk driving. This technology serves as a crucial safety measure, preventing individuals from operating a vehicle under the influence of alcohol. In such systems, various features and components collaborate to detect alcohol presence and ensure the car remains locked. We have carefully considered the insights presented in the paper and incorporated relevant changes in our approach to align with these key points.

7. The authors of the reference paper are **G. Aravindh and M. Arun Kumar.** The paper is titled **" An Efficient Car Parking Management System using Raspberry Pi."**

Reflecting on our project, it has proven to be a valuable learning experience, offering numerous insights and takeaways during its development. We recognize that Raspberry Pi stands out as an excellent and cost-effective platform for IoT projects. Its versatility, low power consumption, and compatibility with a wide range of sensors and peripherals make it an ideal choice for car parking management systems. Incorporating the insights gleaned from the referenced literature, we have enhanced the features of our project, making it more accurate and aligned with the advancements highlighted in the paper.

2.2 REFERENCE BOOKS

2.2.1 Computer Vision and Object Detection

Computer vision is a field of study that focuses on enabling machines to interpret visual information from the world. Object detection is a specific subfield of computer vision that involves identifying and locating objects within images or video streams. It plays a crucial role in various applications, including autonomous vehicles and safety systems.

Computer vision techniques can be used to analyse the video feed from the backup camera, identify objects, and provide visual and textual feedback to the driver in real time.

Some key concepts and technologies in this area include:

- ***Image Processing:*** Image processing techniques like edge detection, filtering, and image segmentation are used to preprocess the video feed and extract relevant information.
- ***Deep Learning:*** Deep learning models, such as Convolutional Neural Networks (CNNs), are commonly employed for object detection tasks. They can be trained to recognize a wide range of objects in real-time.
- ***Real-time Object Tracking:*** To ensure safety, real-time tracking of objects, such as pedestrians, other vehicles, or obstacles, can be implemented to assist the driver in making informed decisions.

1. Computer Vision: Algorithms and Applications - by Richard Zaleski
2. Practical Computer Vision - by Martin Görner, Ryan Gillard, et al.
3. Object Detection for Autonomous Vehicles: A Survey.

2.2.2 Raspberry Pi and IoT

Raspberry Pi is a small, affordable, and versatile single-board computer that is widely used for various IoT projects. IoT (Internet of Things) refers to the network of physical objects connected to the internet, and it can provide data and control capabilities to enhance a wide range of applications.

In your project, Raspberry Pi serves as the central computing unit, connecting various components of your system. It can manage the data from the backup camera, sensors, and alcohol detection system while also providing the user interface.

Key concepts include:

- ***GPIO (General Purpose Input/Output):*** Raspberry Pi's GPIO pins allow you

to connect sensors, cameras, and other peripherals to the board, making it suitable for IoT applications.

- ***Communication Protocols:*** You might use protocols like MQTT or HTTP to facilitate communication between the Raspberry Pi and IoT devices, enabling data exchange and control.

1. Raspberry Pi Cookbook - by Simon Monk.
2. IoT Projects with Raspberry Pi - by John C. Shovic.
3. Getting Started with Raspberry Pi - by Matt Richardson and Shawn Wallace.

2.2.3 Advanced Driver Assistance Systems (ADAS)

- ***Advanced Driver Assistance Systems:*** Principles, Challenges, and Implementation - by Dominik Sankowski

- ***Vision-Based Driver Assistance Systems:*** Build Safe, Smart, Self-Driving Cars - by Gaurav Dubey

2.2.4 Alcohol Detection Systems

Alcohol detection systems are crucial for ensuring that drivers are sober and safe while operating a vehicle. The integration of a chip into the car's steering wheel that can detect alcohol consumption is a valuable safety feature. Key points include:

- ***Sensors and Breath Analysis***: Alcohol detection systems often use sensors that can analyse a person's breath to determine their blood alcohol content (BAC). This can be done using technologies like infrared spectroscopy or electrochemical sensors.

- ***Safety Interlock:*** When alcohol is detected above the legal limit, the system can prevent the car from starting, ensuring the safety of both the driver and others on the road.

1. Alcohol Detection by Means of Remote Car Ignition Block.
2. Smart Vehicle Alcohol Detection System.

2.2.5 Sensor Technology and Parking Assistance

The parking assistance system in your project relies on sensors to measure the distance between the car and nearby objects, making parking safer and more convenient.

Key concepts include:

- ***Ultrasonic Sensors:*** Ultrasonic sensors are commonly used for proximity

detection in parking assistance systems. They emit ultrasonic waves and measure the time it takes for them to bounce back from nearby objects to calculate distances.

- ***Parking*** Algorithms: Algorithms are used to process sensor data and determine the optimal steering and acceleration inputs to safely park the vehicle.

- ***Display and Feedback:*** The textual representation of detected objects on the screen provides the driver with real-time feedback about the proximity of obstacles, enhancing safety during parking manoeuvres.

1. Sensors for Automotive Applications - by Sergey Y. Yurish and Vladimir A. Petrovsky.
2. Intelligent Vehicle Technologies: Theory and Applications - by Ljubo Vlacic, Yimin Jia, et al.

Chapter-3 Connecting the dots: From customer needs to project outcomes in real time

3.1 FEATURES OF PROJECT

3.1.1 Alcohol Detection

This feature is designed to prevent alcohol-impaired driving by monitoring the driver's alcohol level in real-time using two alcohol sensors. Two alcohol sensors are integrated into the vehicle, typically near the driver's seat. These sensors continuously monitor the driver's breath or other bodily fluids to measure alcohol concentration. When the alcohol level is below 0.9 (a predefined safe threshold), the system allows normal vehicle operation. Raspberry Pi processes data from the alcohol sensors and makes real-time decisions. It runs the program responsible for monitoring and controlling the vehicle based on alcohol levels. However, if the alcohol level exceeds 0.9, the system takes immediate action to stop the car. Additionally, the system sends an email notification to family members or designated contacts to inform them of the situation. This enhances road safety by preventing drunk driving and ensuring the well-being of both the driver and other road users. It acts as a deterrent against impaired driving and provides a safety net in case the driver attempts to operate the vehicle while intoxicated.

3.1.2 Object Detection

This feature combines the power of object detection and ultrasonic sensing to enhance rear-view safety for vehicles. With a camera mounted behind the car, the system continuously monitors and analyses the area behind the vehicle for any objects approaching within a distance of 20cm. When an object is detected within this range, the integrated ultrasonic sensor triggers a series of audible alerts, typically buzzers, to warn the driver of the imminent collision risk, thus helping prevent accidents and improving driver awareness during reverse maneuvers. This innovative solution not only enhances vehicle safety but also contributes to mitigating accidents caused by blind spots and obstacles while parking or backing up. This greatly enhances safety during reversing and parking, reducing the likelihood of collisions with objects or pedestrians behind the vehicle.

3.1.3 Safe Parking

This feature is designed to prevent collisions between the car and nearby objects while parking. It utilizes four IR sensors, positioned at different points around the car, typically mounted on the front, rear, and sides. These sensors emit infrared beams and measure the time it takes for the beams to bounce back from nearby objects. to measure the distance between the car and nearby objects. When the car moves too close to an object, one or more of the sensors detect the proximity and activate a buzzer to alert the driver, thereby reducing the risk of collisions. This significantly reduces the risk of collisions between the car and objects during parking maneuvers, enhancing parking safety and preventing damage to both the car and nearby objects.

3.2 FEATURES APPLICATIONS

3.2.1 Alcohol Detection

a) ***Preventing Alcohol-Impaired Driving:*** The primary goal of this project is to deter individuals from driving under the influence of alcohol, which is a major contributor to accidents and fatalities on the road.

b) ***Family Safety and Awareness:*** Family members are notified via email when the driver's alcohol level is above the defined threshold (0.9 or any predetermined limit). This enables them to take appropriate action, such as preventing the driver from operating the vehicle or offering assistance.

c) ***Real-Time Monitoring:*** The project continuously monitors the driver's alcohol level in real-time, ensuring that any impairment is detected and addressed promptly.

d) ***Vehicle Safety:*** The ability to disable the car when the alcohol level is above the threshold adds an extra layer of safety. It prevents the vehicle from being driven by an impaired individual, reducing the risk of accidents.

e) ***Application in Public Transport:*** Similar technology can be applied in public transport systems, such as buses and taxis, to ensure that drivers are not impaired while on duty.

f) ***Preventing Underage Drinking:*** This system can also be adapted to prevent underage individuals from accessing and driving vehicles if their alcohol level is detected.

3.2.2 Object Detection

a) ***Rear-View Parking Assistance:*** Object detection at the back of the car aids in parking by identifying obstacles, curbs, or other vehicles behind the car, providing the driver with visual and audible cues to maneuver safely into parking spaces.

b) ***Preventing Back over Accidents:*** These systems are crucial for preventing accidents involving pedestrians, particularly children or pets, who may be in the vehicle's blind spot when the driver is backing up.

c) ***Trailer Hitching:*** Object detection helps align the car with a trailer or hitch, making it easier to connect the vehicle to a trailer without assistance.

d) ***Avoiding Rear-End Collisions:*** Object detection behind the car can warn the driver when a vehicle is approaching too closely from behind, reducing the risk of rear-end collisions.

e) ***Cargo and Loading Assistance:*** In commercial and cargo vehicles, object detection assists in loading and unloading operations, ensuring that the rear of the vehicle is clear of obstructions and that cargo is secured safely.

3.2.3 Safe Parking

a) ***Personal Vehicles:*** The system can be installed in personal cars to assist with parking in residential areas, parking garages, or tight spaces.

b) ***Commercial Vehicles:*** Commercial vehicles, like delivery trucks and taxis, can benefit from this system to prevent accidents while parking in urban environments.

c) ***Car Rental Companies:*** Car rental agencies can enhance their vehicle safety by installing safe parking systems, reducing the risk of rental car damage.

d) ***Fleet Vehicles:*** Fleet managers can improve safety for company vehicles, ensuring that drivers park safely during their routes.

e) ***Parking Facilities:*** Parking lots and garages can use this system to improve safety for customers and reduce the risk of vehicle collisions with structures.

3.3 CUSTOMER NEEDS

3.3.1 Alcohol Detection Customer Needs

A customer needs a reliable and accurate alcohol detection system that ensures safe and responsible driving behaviour while respecting their privacy and adhering to legal standards. The system should provide real-time feedback and, if necessary, intervention to prevent alcohol-impaired driving.

3.3.2 Object Detection Customer Needs

Customers typically seek reliable object detection systems for their vehicles to enhance safety by preventing collisions, aiding in parking, and improving overall driving convenience. They prioritize accuracy, real-time alerts, and user-friendly interfaces to make driving safer and more efficient.

3.3.3 Safe Parking Customer Need

Customers need advanced detection systems and real-time guidance, ensuring they can park with confidence and avoid collisions with obstacles or other vehicles.

3.4 REAL TIME PROBLEMS

3.4.1 Object Detection

Some of our team mates have also seen some situations where due to usage of sudden break without seeing the behind vehicle there occurred an accident to avoid that object detection is the solution by using that we can see the person is at safe distance or not.

3.4.2 Alcohol Detection

Alcohol-impaired accident, a person, under the influence, lost control of his car, colliding head-on with an oncoming vehicle. The collision resulted in severe injuries and fatalities, highlighting the devastating consequences of drunk driving on our roads. Made a huge accident where 2 innocent persons lost their lives this inspired us to come with this solution

3.4.3 Safe Parking

We have observed that many people are facing problems in parking the car on a perfect and adjusted slot due to the objects like walls and pillars present at the back as they can't see them and many of them dashes the car so we have come with the solution of safe parking.

3.5 CONCEPT SELECTION

3.5.1 Alcohol Detection Concept Selection

a) ***Breath Alcohol Testing:*** These devices measure the alcohol content in a person's breath to estimate their blood alcohol concentration (BAC). They are commonly used by law enforcement for roadside testing.

b) ***Blood Alcohol Testing:*** The most accurate method for measuring a person's BAC by analysing a blood sample. Often used in legal and medical contexts.

c) ***Saliva Alcohol Testing:*** These tests detect alcohol in a person's saliva, providing a quicker and less invasive alternative to blood testing.

d) ***Urine Alcohol Testing:*** Used to detect alcohol metabolites in the urine, providing a longer window of detection but with lower accuracy compared to blood or breath tests.

e) ***Sweat Alcohol Testing:*** Sweat Patches These wearable patches collect sweat over time and can detect alcohol use. They are often used in monitoring programs.

f) ***Hair Alcohol Testing:*** Hair Tests Alcohol metabolites can be detected in hair strands, offering a longer-term history of alcohol use.

g) ***Infrared Spectroscope:*** Infrared Breath Analyzer These devices measure alcohol concentration in a person's breath using infrared light absorption.

h) ***Gas Chromatography:*** Gas Chromatography-Mass Spectrometry (GC-MS) A highly precise laboratory method for analysing blood, breath, or urine samples to determine BAC.

i) ***Semiconductor-Based Sensors:*** Semiconductor Alcohol Sensors Used in ignition interlock devices and car alcohol detection systems, these sensors detect alcohol vapor in the breath.

j) ***Field-Effect Transistor (FET) Sensors:*** FET Alcohol Sensor Utilized in wearable alcohol detection devices, FET sensors measure changes in electrical conductivity caused by the presence of alcohol.

k) ***Biometric Detection:*** Behavioural biometrics analysing behavioural changes, such as speech patterns, eye movements, or gait, to identify signs of alcohol impairment.

l) ***Machine Learning and AI:*** Leveraging artificial intelligence and machine learning algorithms to analyse data from various sensors, including smartphone sensors, for behavioural and biometric indicators of alcohol impairment.

From all the above method we have choose the breath testing as it suits our project.

3.5.2 Object Detection Concept Selection

a) ***Haar Cascade Classifiers:*** Haar cascades are based on machine learning and are effective for detecting objects with distinct features, such as faces. They are computationally efficient and are used in real-time applications.

b) ***YOLO (You Only Look Once):*** YOLO is a deep learning-based object detection system that divides an image into a grid and predicts bounding boxes and class probabilities simultaneously. It's known for its speed and accuracy.

c) ***Faster R-CNN:*** Faster Region-based Convolutional Neural Networks (RCNN) use a combination of deep learning and region proposal networks to detect objects. They are highly accurate but may be slower compared to YOLO.

d) ***SSD (Single Shot MultiBox Detector):*** SSD is another deep learning model that is optimized for speed and accuracy. It predicts object classes and bounding boxes for multiple objects in a single pass.

e) ***HOG (Histogram of Oriented Gradients) with SVM (Support Vector Machine):*** HOG with SVM is a traditional computer vision approach for object detection. It analyses gradient information in images to detect objects, making it particularly useful for pedestrian detection and similar tasks. From all the above object detection algorithms we have selected the most accurate one.

3.5.3 Safe Parking Concept Selection

a) ***Designated Parking Areas:*** Clearly marked parking spaces with adequate signage and pavement markings help drivers navigate parking lots and minimize confusion.

b) ***Adequate Lighting:*** Well-lit parking areas getting deter criminal activity and improve visibility, reducing the risk of accidents and ensuring the safety of those using the parking facility.

c) ***Pedestrian Crosswalks:*** Designating safe pedestrian crosswalks with clear markings and signage helps ensure that pedestrians can safely cross parking areas.

d) ***Obstacle Detection Systems:*** Installing object detection systems, such as sensors and cameras, can help drivers identify obstacles and other vehicles while parking or manoeuvring in tight spaces.

e) ***Traffic Calming Measures:*** Implementing speed bumps, stop signs, and other traffic-calming measures within parking lots can help reduce the risk of collisions and protect pedestrians.

f) ***Driver Education and Awareness:*** Promoting responsible parking and awareness of parking lot rules and etiquette is crucial. Encouraging drivers to check their surroundings, avoid distracted driving, and respect pedestrian right of way enhances overall safety.

We have come up with a new way of parking using ir sensors as an experiment and we have succeeded in our way of selecting a concept.

3.6 PROJECT OUTCOME

The project aims to create a prototype of a smart and safe car with object detection and alcohol detection systems that significantly reduce the risk of accidents caused by impaired or distracted drivers. By merging these technologies, the vehicle will become more aware of its surroundings and the driver's condition, ultimately contributing to road safety and potentially saving lives. The combination of these features in a smart and safety car can lead to a significant reduction in accidents, injuries, and fatalities on the road, as well as improved parking experiences for drivers. It promotes responsible and safe driving practices, which benefits both individuals and society.

Chapter-4 The Algorithmic Constructs Used

4.1 RASPBERRY PI OS

Formerly known as Raspbian is a specialized operating system designed for the Raspberry Pi single-board computers. It is the official and recommended operating system for Raspberry Pi devices.

Here's a detailed explanation of Raspberry Pi OS:

4.1.1 Purpose

Raspberry Pi OS is designed to provide a lightweight, optimized, and user-friendly operating system for the Raspberry Pi hardware. It allows users to run a variety of applications and projects on their Raspberry Pi devices, making them suitable for a wide range of purposes, from basic computing to robotics and embedded systems.

4.1.2 Key Features

a) ***Optimized for Raspberry Pi:*** Raspberry Pi OS is specifically tailored to work efficiently on Raspberry Pi hardware, taking full advantage of its capabilities.

b) ***Desktop Environment:*** The default Raspberry Pi OS installation includes the PIXEL (Pi Improved X-Window Environment, Lightweight) desktop environment, which provides a familiar and user-friendly graphical interface**.**

c) ***Software Repository:*** Raspberry Pi OS includes a vast repository of prepackaged software that can be easily installed using the package manager. This ensures access to a wide range of software and tools.

d) ***Customization:*** Users can customize their Raspberry Pi OS installations, installing and configuring software and settings to suit their specific needs.

e) ***Raspberry Pi Configuration Tool:*** The Raspberry Pi Configuration tool provides an easy way to adjust system settings and configurations, such as enabling/disabling hardware interfaces, configuring the boot options, and more.

f) ***Access to GPIO:*** Raspberry Pi OS allows users to interact with the GPIO (General-Purpose Input/Output) pins of the Raspberry Pi, which is essential for hardware interfacing and DIY electronics projects.

4.1.3 Remote Access

The OS supports remote access protocols like SSH (Secure Shell) and VNC (Virtual Network Computing), which are helpful for headless (without a monitor) operations and remote administration.

4.1.4 Educational Tools

Raspberry Pi OS includes educational software and resources for learning programming and electronics, making it an excellent platform for educational purposes**.**

Installation: • Raspberry Pi OS can be installed on a Raspberry Pi using an SD card.

- The official Raspberry Pi website provides downloadable images that can be flashed onto an SD card using tools like Etcher.
- Once the SD card is prepared, it can be inserted into the Raspberry Pi, and the OS boots from it.

4.1.5 Variants

There are two main variants of Raspberry Pi OS:

a) Raspberry Pi OS (32-bit): This is the standard version of the OS, suitable for most Raspberry Pi models. It is based on the Debian Linux distribution.

b) Raspberry Pi OS (64-bit): This is a 64-bit version designed for Raspberry Pi models with 64-bit CPUs, such as the Raspberry Pi 3 and 4.

Community Support: Raspberry Pi OS benefits from a large and active community. This means there are extensive resources, tutorials, and forums available for users to get help, troubleshoot issues, and share projects.

4.1.6 Regular Updates

The Raspberry Pi Foundation regularly updates and improves Raspberry Pi OS to provide security updates, bug fixes, and new features. It's important to keep the OS up to date to ensure the best performance and security. In summary, Raspberry Pi OS is a specialized operating system built for the Raspberry Pi single-board computers. It provides an easy-to-use environment for various projects, making it an ideal choice for hobbyists, educators, and developers working with Raspberry Pi hardware.

4.2 VNC (VIRTUAL NETWORK COMPUTING)

Viewer is a software application that allows you to remotely access and control another computer or device over a network, typically the internet. It is a part of the broader VNC system, which includes both the VNC Viewer (client) and the VNC Server (host) components.

Here's a detailed explanation of VNC Viewer:

4.2.1 Purpose

VNC Viewer is used to connect to a remote computer that is running a VNC Server. It enables you to view and interact with the desktop or graphical user interface (GUI) of the remote computer as if you were physically present in front of it. This can be useful for a variety of purposes, such as technical support, remote troubleshooting, or accessing your home or office computer from a different location.

4.2.2 How It Works

The remote computer runs a VNC Server, which captures the screen and transmits it over the network. The VNC Viewer, installed on your local computer or device, connects to the remote computer using a VNC protocol. The VNC Viewer receives the transmitted screen data and displays it on your local screen. You can send keyboard and mouse input from your local computer to control the remote computer.

4.2.3 Key Features

a) ***Remote Control****:* You can control the remote computer as if you were physically present, including opening and running applications, moving the mouse cursor, and typing on the keyboard.

b) ***File Transfer****:* Many VNC Viewer applications allow you to transfer files between the local and remote computers.

c) ***Cross-Platform****:* VNC is available for various operating systems, making it possible to access and control a remote computer from different platforms.

d) ***Encryption****:* Some VNC implementations support secure encryption to protect data transmission between the client and server.

4.2.4 Setting Up VNC Viewer

- Install a VNC Viewer application on your local computer or device.

- There are many VNC Viewer applications available for different platforms, including RealVNC, TightVNC, and UltraVNC.
- Know the IP address or hostname of the remote computer where the VNC Server is running.
- Enter the IP address or hostname and the appropriate port (usually 5900) in the VNC Viewer.
- Authenticate yourself with a password or other authentication method set up on the remote computer.

4.2.5 Security Considerations

It's essential to set strong passwords for the VNC Server to prevent unauthorized access. Use encryption to secure data transmission between the client and server, especially if you're accessing remote computers over the internet. Only use VNC on trusted networks, as it can pose security risks if not configured properly.

4.2.6 VNC Variants

There are several VNC variants and implementations, such as RealVNC, TightVNC, UltraVNC, and more. Each may offer additional features or have variations in how they work, but the core concept of remote desktop access is the same. In summary, VNC Viewer is a client application that allows you to remotely control and view the desktop of a computer running a VNC Server. It provides a way to manage and troubleshoot remote computers, making it a valuable tool for IT support, system administration, and remote work.

4.3 RASPBERRY PI IMAGE

The Raspberry Pi Imager is a software utility developed by the Raspberry Pi Foundation that facilitates the installation of operating systems onto SD cards or microSD cards for Raspberry Pi single-board computers. It simplifies the process of preparing an SD card with the required software, making it a user-friendly tool for both beginners and experienced Raspberry Pi users.

Here's a detailed explanation of the Raspberry Pi Imager:

4.3.1 Purpose

The Raspberry Pi Imager is primarily designed for the Raspberry Pi community to easily and quickly set up the Raspberry Pi's operating system on a compatible SD card or microSD card. It can be used to install the official Raspberry Pi OS, as well as other supported operating systems like Ubuntu, Debian, and more.

4.3.2 Key Features

a) ***User-Friendly Interface:*** The Imager has a simple and intuitive graphical user interface (GUI) that makes it accessible to users of all skill levels.

b) ***Cross-Platform****:* The Imager is available for various operating systems, including Windows, macOS, and Linux, allowing users to create bootable cards on their preferred platform.

c) ***Automated OS Download****:* The Imager can automatically download the latest version of the selected operating system from the internet, ensuring that users get the most up-to-date software.

d) ***Verify Write:*** After writing the OS to the SD card, the Imager can verify that the write process was successful to avoid issues related to corrupted or incomplete installations.

e) ***Advanced Options****:* The Imager offers advanced options that allow users to configure the storage size for the OS partition, enabling the use of larger SD cards for storage.

f) ***Support for Custom OS Images****:* While the Imager prominently features official Raspberry Pi OS images, it also allows users to load custom OS images that are not listed by default.

4.3.3 Installation Process

To use the Raspberry Pi Imager,

- Download and install the Imager for your specific operating system.
- Launch the Imager and insert an SD card or microSD card into your computer.
- Select the desired operating system from the list, or use the "Use Custom" option to load a custom image.
- Choose the target storage device (the SD card you inserted).
- Click the "Write" button to begin the installation process.
- The Imager will download the selected OS image (if needed) and write it to the SD card.
- After completion, you can eject the card and insert it into your Raspberry Pi for booting.

4.3.4 Compatibility

The Raspberry Pi Imager supports various Raspberry Pi models, including the Raspberry Pi 1, 2, 3, 4, and Raspberry Pi Zero series. It is also compatible with other single-board computers or devices that use SD cards or microSD cards as their primary storage medium.

4.3.5 Community Support

The Raspberry Pi community provides extensive support and resources related to the Imager. This includes tutorials, forums, and documentation to assist users with the installation process and troubleshooting. In summary, the Raspberry Pi Imager is a user-friendly software utility developed by the Raspberry Pi Foundation that simplifies the process of preparing SD cards with operating systems for Raspberry Pi computers. It automates the process of downloading and writing OS images, making it a valuable tool for anyone looking to set up a Raspberry Pi quickly and efficiently.

4.4 PYTHON PROGRAMMING

Python is a high-level, general-purpose programming language. Its design philosophy emphasizes code readability with the use of significant indentation. Python is an interpreted, interactive, object-oriented programming language. It incorporates modules, exceptions, dynamic typing, very high-level dynamic data types, and classes. It supports multiple programming paradigms beyond object-oriented programming, such as procedural and functional programming. Python is commonly used for developing websites and software, task automation, data analysis, and data visualization. Since it's relatively easy to learn, Python has been adopted by many nonprogrammers such as accountants and scientists, for a variety of everyday tasks, like organizing finances.

4.4.1 Advantages of Python

- Easy to Learn and Use.
- Free and Open-Source.
- Rapid Development.
- Interpreted Language.
- Wide Range of Libraries and Frameworks.
- Dynamically Typed.
- Portability.
- Strong Community Support.

4.4.2 Steps to run Python

- Download Thonny IDE.
- Run the installer to install Thonny on your computer.
- Go to: File > New. Then save the file with .py extension.
- Write Python code in the file and save it. ...
- Then Go to Run > Run current script or simply click F5 to run it.

4.5 MQTT (MESSAGE QUEUING TELEMETRY TRANSPORT)

It is a lightweight, publish-subscribe messaging protocol designed for low-bandwidth, high-latency, or unreliable networks. It was created by IBM in the late 1990s and has since become an open standard widely used in the Internet of Things (IoT) and various other applications.

Here's a detailed explanation of MQTT:

4.5.1 Purpose

MQTT operates on a publish-subscribe messaging model, which allows for efficient communication between multiple clients or devices. In this model, clients communicate by sending and receiving messages through a central message broker.

4.5.2 Key Concepts

- ***Publisher:*** **A** client that generates and sends messages to the broker**.**
- ***Subscriber:*** A client that receives messages from the broker based on subscription**.**
- ***Broker:*** The central server or middleware that manages message distribution. It is responsible for routing messages from publishers to subscribers**.**
- ***Topic****:* Messages are categorized into topics, which are used by subscribers to filter and specify the types of messages they want to receive. Topics are hierarchical, using a structure similar to a file path, e.g., "home/livingroom/temperature."

4.5.3 Quality of Service (QoS)

MQTT provides different levels of quality of service to ensure message delivery reliability:

- ***QoS 0 (At most once):*** Messages are delivered at most once, with no confirmation or acknowledgment.
- ***QoS 1 (At least once):*** Messages are guaranteed to be delivered at least once, but duplicates may occur.
- ***QoS 2 (Exactly once):*** Messages are guaranteed to be delivered exactly once, with the highest level of assurance.

4.5.4 Retained Messages

MQTT allows for the retention of the last message sent on a topic. This means that when a client subscribes to a topic, it immediately receives the last retained message for that topic.

4.5.5 Last Will and Testament

Clients can specify a "last will" message that the broker will publish on a specified topic if the client unexpectedly disconnects. This feature is useful for handling client failures and monitoring.

4.5.6 Lightweight Protocol

MQTT is designed to be lightweight and efficient. It uses a small header and binary message payloads, making it suitable for resource-constrained devices and lowbandwidth networks**.**

4.5.7 Connection Patterns

MQTT supports three common connection patterns**:**

- ***Device-to-Broker****:* Devices (publishers and subscribers) connect to a central MQTT broker.
- ***Broker-to-Broker:*** Multiple brokers can be interconnected to facilitate communication between different MQTT networks**.**
- ***Device-to-Device****:* Devices can communicate directly with each other, known as MQTT-SN (MQTT for Sensor Networks), which extends MQTT for more complex topologies.

4.5.8 Security

MQTT itself does not provide built-in security features. However, security can be implemented by using encryption (TLS/SSL), authentication, and access control mechanisms at the broker level to protect data and ensure secure communication**.**

4.5.9 Use Cases

MQTT is commonly used in various applications, including:

- IoT and smart home devices for real-time sensor data and device control.
- Telemetry and remote monitoring systems.
- Messaging in industrial automation and control systems.
- Mobile applications for real-time data updates.

- Chat applications and instant messaging services.

In summary, MQTT is a lightweight and efficient messaging protocol that uses a publish-subscribe model for communication. It is particularly well-suited for scenarios where network bandwidth and reliability are limited, such as in IoT devices, remote monitoring, and real-time communication applications. Its flexibility and scalability have made it a popular choice for a wide range of use cases.

4.6 LIBRARIES

In the context of software development and programming, a "library" refers to a collection of pre-written code and functions that can be reused by developers to perform common tasks or solve specific problems. Libraries are an essential part of software development as they help save time and effort by providing a set of functions, classes, or modules that can be incorporated into your own programs.

Here's a detailed explanation of libraries:

4.6.1 Purpose

Libraries are created to serve a specific purpose or solve a particular problem. They are designed to be reusable, which means developers can use them in multiple projects without having to re-implement the same functionality. Libraries encapsulate commonly used code, making it more efficient and easier to maintain.

4.6.2 Types of Libraries

There are various types of libraries, depending on their purpose and scope. Some common types include:

a) Standard Libraries*:* These come with the programming language itself and provide essential functionality like input/output, data structures, and mathematical operations.

b) Third-party Libraries*:* Developed by third-party individuals or organizations, these libraries extend the capabilities of a programming language. They can cover a wide range of domains, including web development, graphics, data analysis, and more.

c) Custom Libraries: Developers can create their own libraries to encapsulate code that they want to reuse across multiple projects.

4.6.3 Key Characteristics

- ***Modularity:*** Libraries are designed to be modular, with functions or classes organized in a structured manner. This makes it easy to use only the parts of the library that are needed for a specific task.

- ***Abstraction:*** Libraries often provide high-level abstractions, simplifying complex tasks. This abstraction can help developers work with the library without needing to understand the underlying implementation details.

- ***Documentation:*** Good libraries come with comprehensive documentation, including usage examples, API references, and guides, to help developers understand how to use the library effectively.

- ***Community and Support:*** Popular libraries often have a community of users who provide support, contribute to the library's development, and share knowledge and best practices.

4.6.4 Benefits of Using Libraries

- ***Time Efficiency:*** Using libraries can significantly reduce development time by eliminating the need to reinvent the wheel for common tasks.

- ***Code Reusability:*** Libraries encourage code reusability, making it easier to maintain and update code across multiple projects.

- ***Reliability:*** Well-established libraries are thoroughly tested, and issues are addressed by the community, enhancing the reliability of the code.

- ***Scalability:*** Libraries can grow in functionality and scope as new features and improvements are added over time.

- ***Specialized Expertise:*** Libraries created by domain experts often provide solutions to specific problems or challenges in a particular field.

4.6.5 Examples of Popular Libraries

Some of the examples where libraries are stored are as follows,

- ***Python**:* The Python Standard Library includes a wide range of modules for tasks like file handling, regular expressions, data serialization, and more. Thirdparty libraries like NumPy (for numerical computing), TensorFlow (for machine learning), and Flask (for web development) are also widely used.
- ***JavaScript**:* JavaScript has libraries like jQuery (for DOM manipulation), React (for building user interfaces), and Axios (for making HTTP requests).
- ***C/C++:*** The C Standard Library provides functions for input/output, string manipulation, and memory management. C++ has the Standard Template Library (STL) for data structures and algorithms.
- ***Java:*** Java provides libraries like Java Standard Library, JavaFX (for graphical user interfaces), and Hibernate (for database interaction).

In summary, libraries are a fundamental component of software development, offering pre-built code and functionality to solve common programming tasks efficiently. They promote code reusability, reduce development time, and often serve as a valuable resource for developers to build applications and projects across various domains and programming languages.

4.7 DATA BASE

A database is a structured collection of data that is organized and stored for efficient retrieval, management, and manipulation. Databases are used in various applications, including business, science, education, and software development.

Here's a detailed explanation of databases:

4.7.1 Purpose

Databases are designed to store and manage large amounts of data in a structured and organized manner. They serve as a central repository for information and provide tools for storing, retrieving, updating, and analyzing data.

4.7.2 Key Concepts

- ***Data**:* Databases store data, which can be anything from text and numbers to multimedia files and complex data structures.

- ***Tables**:* Data is organized into tables, which consist of rows and columns. Each row represents a record, while each column represents a field or attribute.

- ***Schemas:*** A database schema defines the structure of the database, specifying the tables, relationships, and constraints that govern how data is stored.

- ***Queries:*** Users can retrieve and manipulate data using database queries, which are instructions that specify what data to fetch, update, or perform calculations.

- ***Indexes****:* Indexes are data structures that improve data retrieval speed by creating a map of where specific data is stored in the database**.**

- ***Normalization****:* The process of organizing data to eliminate redundancy and improve data integrity is known as normalization**.**

4.7.3 Types of Databases

a) Relational Databases: These databases use tables with structured rows and columns to store data. Examples include MySQL, PostgreSQL, Oracle, and Microsoft SQL Server.

b) NoSQL Databases*:* NoSQL databases are designed for unstructured or semistructured data and include document stores (MongoDB), key-value stores (Redis), column-family stores (Apache Cassandra), and graph databases (Neo4j).

c) In-Memory Databases: These databases store data in memory rather than on disk, offering extremely fast data retrieval. Examples include Redis and Memcached.

d) NewSQL Databases*:* A relatively new category that aims to combine the scalability of NoSQL databases with the transactional integrity of traditional relational databases.

4.7.4 Database Management Systems (DBMS)

A DBMS is software that manages the interaction between users, applications, and the database. It provides tools for creating, maintaining, and querying the database. Common DBMSs include MySQL, PostgreSQL, SQLite, Microsoft SQL Server, and Oracle Database.

a) **Normalization**

Database normalization is a process used in relational databases to organize data efficiently. It minimizes data redundancy and reduces the chances of data anomalies, ensuring that data is stored in a consistent and logical manner.

b) **Database Operations**

CRUD Operations are as follows,

- Create: Inserting new records into a database.
- Read: Retrieving data from the database.
- Update: Modifying existing data in the database.
- Delete: Removing records from the database.

c) **Data Querying:** Retrieving specific data using query languages such as SQL (Structured Query Language).

d) **Transactions:** Ensuring data integrity through transaction management, allowing for multiple operations to be treated as a single, atomic unit of work.

4.7.5 Security and Access Control

Databases implement access controls and security measures to protect data from unauthorized access, including authentication, authorization, and encryption.

a) ***Scalability:***

Databases can be designed for scalability to handle growing amounts of data and users. Scalability can be achieved through various techniques, including sharding, replication, and distributed databases.

b) ***Data Integrity and Constraints:***

Databases often enforce data integrity constraints, such as primary keys, foreign keys, unique constraints, and check constraints, to ensure the consistency and accuracy of data.

4.7.6 Use Cases

Databases are used in a wide range of applications, including:

- Business applications for managing customer data, inventory, and financial records.
- Web applications to store user profiles, content, and transaction records.
- Scientific research for data storage and analysis.
- Healthcare systems for managing patient records.
- E-commerce platforms for product catalogs and order processing**.**

In summary, databases are critical components of modern computing, enabling efficient data storage, retrieval, and management. They come in various types and are managed by database management systems. Understanding databases and their principles is essential for developers, data analysts, and anyone working with data-driven applications.

4.8 THE UNIVERSAL DATA TOOL (UDT)

It is an open-source software tool designed to facilitate data labeling, data annotation, and data management tasks. It is a versatile platform that allows users to create, edit, and manage labeled data for machine learning and data analysis projects. The UDT supports a wide range of use cases, including image annotation, text annotation, audio annotation, and more.

4.8.1 Key features

Key features of the Universal Data Tool include:

- ***Data Labeling:*** Users can label and annotate data with various data types, such as images, text, audio, video, and more. This is particularly useful for training and testing machine learning models.

- ***Collaboration****:* UDT supports collaboration features, enabling multiple users to work on the same project simultaneously. This can be beneficial for teams working on data annotation tasks.

- ***Data Export****:* The tool allows users to export labeled data in various formats that are compatible with popular machine learning libraries and frameworks.

- ***Extensibility****:* UDT is highly customizable and extensible. Users can create custom interfaces and labeling tools tailored to their specific data labeling needs**.**

- ***Versatility:*** The Universal Data Tool is platform-independent and can be used on various operating systems, including Windows, macOS, and Linux. It also offers a web-based version for online use**.**

- ***Active Community****:* UDT has an active community of users and developers who contribute to its development and provide support**.**

The Universal Data Tool is particularly useful for tasks like object detection, image segmentation, sentiment analysis, and natural language processing, among others. It simplifies the process of creating high-quality labeled datasets, which are essential for training and evaluating machine learning models. If you're working on a project that involves data annotation or labeling for machine learning, the Universal Data Tool can be a valuable resource to streamline your workflow and improve the efficiency of your data labeling process.

4.9 COCO.NAMES

This refers to a file used in computer vision and deep learning, specifically in the context of object detection and image classification tasks. This file contains a list of class names that are commonly used in the COCO (Common Objects in Context) dataset. The COCO dataset is a widely used dataset in the field of computer vision and is often used to train and evaluate object detection and image classification models.

Here's a detailed explanation of coco.names:

4.9.1 Purpose

The coco.names file serves as a reference for the names of object classes that can be detected or recognized by models trained on the COCO dataset. These class names represent a wide range of common objects found in everyday scenes, making it a valuable resource for object detection and image analysis tasks.

4.9.2 Content

The coco. names file typically contains a list of class names, with one class name per line. Each class name corresponds to a specific object or category that can be recognized by the model. Examples of class names that may be found in the coco.names file include "person," "car," "dog," "chair," "bicycle," "cup," "tree," and many more.

4.9.3 Usage

- ***Model Training****:* When training deep learning models for object detection or image classification using the COCO dataset, the coco.names file is often used to label the objects in the training data. The model learns to recognize and classify these objects based on the class names provided in the file.

- ***Inference:*** During the inference or prediction phase, the class names in coco.names are used to label the detected objects in an image or video frame. This helps users understand what objects the model has recognized in a given scene.

4.9.4 Common Frameworks

Popular deep learning frameworks like TensorFlow, PyTorch, and Darknet/YOLO often use the COCO dataset and the associated coco.names file for training and inference in object detection models**.**

4.9.5 Community and Standardization

The COCO dataset and the coco.names file are widely adopted in the computer vision community, leading to a level of standardization in object detection and image classification tasks. This makes it easier to share and replicate research results.

4.9.6 Extensibility

While the COCO dataset provides a comprehensive list of common objects, developers and researchers can extend the class names in the coco.names file to include additional categories relevant to their specific applications. Summary, coco.names is a text file that contains a list of class names representing objects or categories commonly found in everyday scenes. It is used in the context of object detection and image classification tasks, serving as a standard reference for labeling and identifying objects in images and videos when working with models trained on the COCO dataset.

4.9.7 Examples

Datasets of animals, datasets of birds, datasets of sports equipment, etc.

Chapter-5 Insights of Raspberry Pi

5.1 RASPBERRY PI

The Raspberry Pi is a series of small, single-board computers developed by the Raspberry Pi Foundation. These versatile and affordable devices are designed for educational purposes, hobbyist projects, and various applications, including IoT (Internet of Things), robotics, media centers, and more. Below is a detailed explanation of the Raspberry Pi, including its common features and GPIO (General Purpose Input/Output) pins.

Figure 5.1: Raspberry Pi 3B+

5.1.1 Features of Raspberry Pi

a) ***Processor:*** Raspberry Pi models are equipped with different processors, with the CPU architecture evolving over time. Common CPU architectures include ARM Cortex-A53, Cortex-A72, and Cortex-A53, among others.

b) ***RAM:*** Raspberry Pi models come with varying amounts of RAM, typically ranging from 1GB to 8GB, depending on the model.

c) ***I/O Ports:*** Raspberry Pi boards have various I/O ports, including USB ports, HDMI, audio, Ethernet, and more, for connecting peripherals and external devices.

d) ***Storage:*** Raspberry Pi models often support microSD cards for storage, and some models include USB ports for connecting external hard drives or SSDs.

e) ***Power Supply:*** Raspberry Pi boards require a 5V power supply, typically provided through a micro-USB or USB-C connector.

f) ***Operating System:*** Raspberry Pi supports various operating systems, with Raspberry Pi OS (formerly Raspbian) being the official and most commonly used option.

g) ***GPIO Pins:*** GPIO pins are used for general-purpose input and output, allowing you to interact with external devices and sensors.

5.1.2 Add on Features for Raspberry pi 3B+

- A 1.4GHz 64-bit quad-core processor
- Dual-band wireless LAN
- Bluetooth 4.2/BLE
- Faster Ethernet
- Power-over-Ethernet support (with separate PoE HAT)
- Two extra USB ports
- Faster than Raspberry pi 2 & 3 models

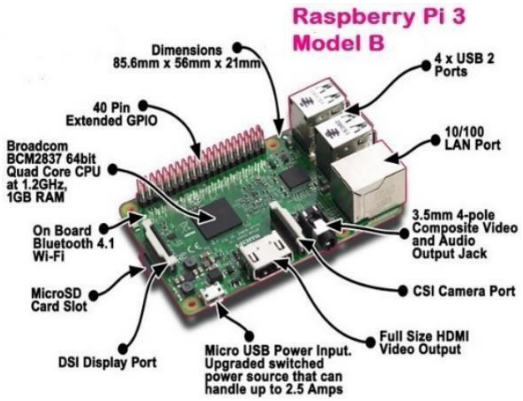


Figure 5.2: Features of Raspberry pi 3B+

5.1.3 Properties of Raspberry pi 3B+

a) ***Processor:*** A 1.4GHz 64-bit quad-core Broadcom BCM2837B0 CortexA53 SoC.

b) ***RAM:*** 1GB LPDDR2 SDRAM

c) ***Connectivity:*** Dual-band 2.4GHz and 5GHz IEEE 802.11.b/g/n/ac wireless LAN, Bluetooth 4.2/BLE, and Gigabit Ethernet over USB 2.0

d) ***Access:*** 4 × USB 2.0 ports, extended 40-pin GPIO header, full-size HDMI, MIPI DSI display port, and MIPI CSI camera port

e) ***Video & sound:*** H.264, MPEG-4 decode (1080p30); H.264 encode (1080p30); OpenGL ES 1.1, 2.0 graphics; 4 pole stereo output and composite video port.

f) ***Multimedia:*** Micro SD format for loading operating system and data storage

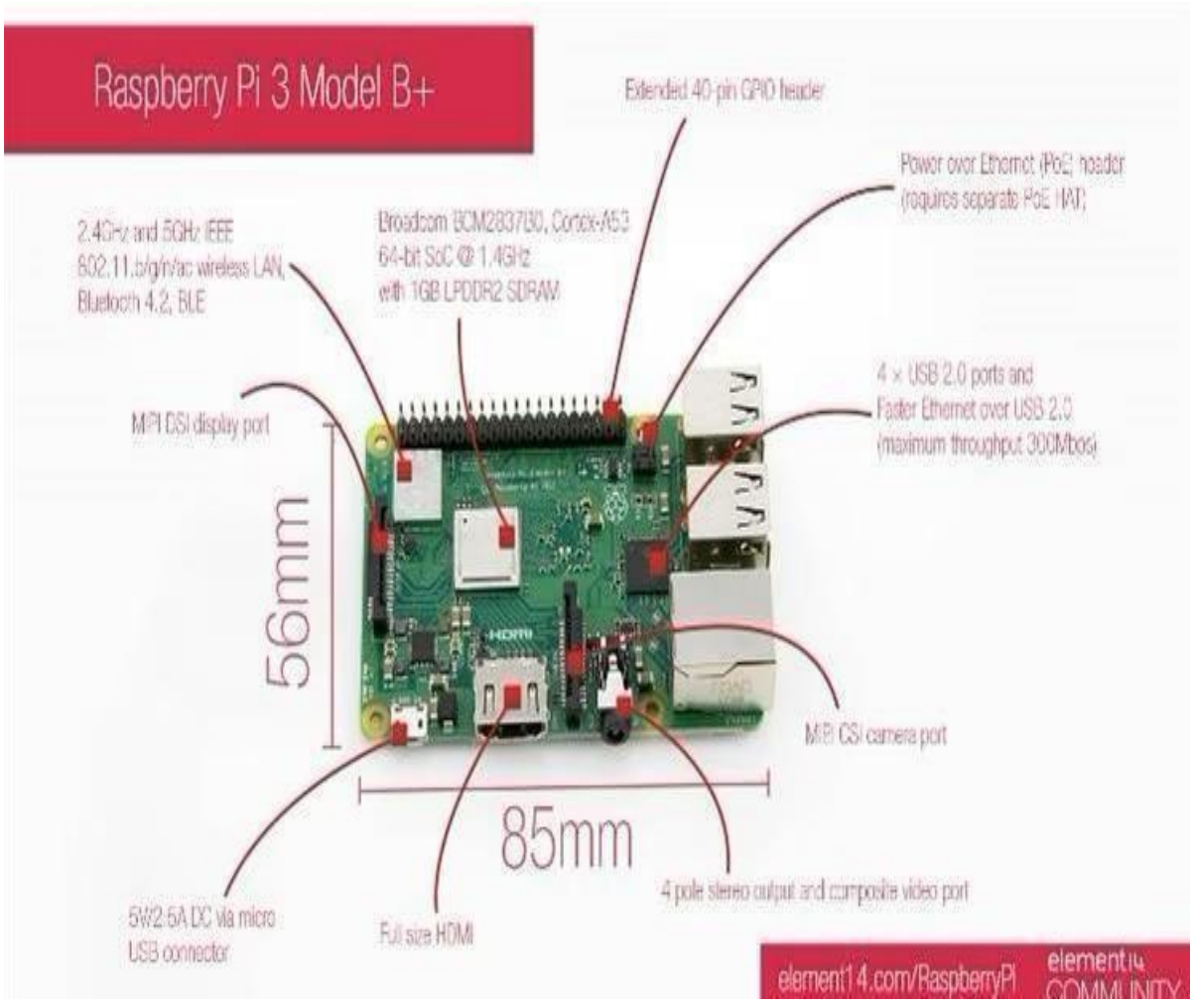


Figure 5.3: Properties of Raspberry pi 3B+

- Input power: 5V/2.5A DC via micro-USB connector, 5V DC via GPIO header, and Power over Ethernet (PoE)-enabled (requires separate PoE HAT)
- Environment: Operating temperature, 0–50°C
- Compliance: For a full list of local and regional product approvals, please visit www.raspberrypi.org/products/raspberry-pi-3-model-b+

5.1.4 Schematic Diagram of Raspberry pi

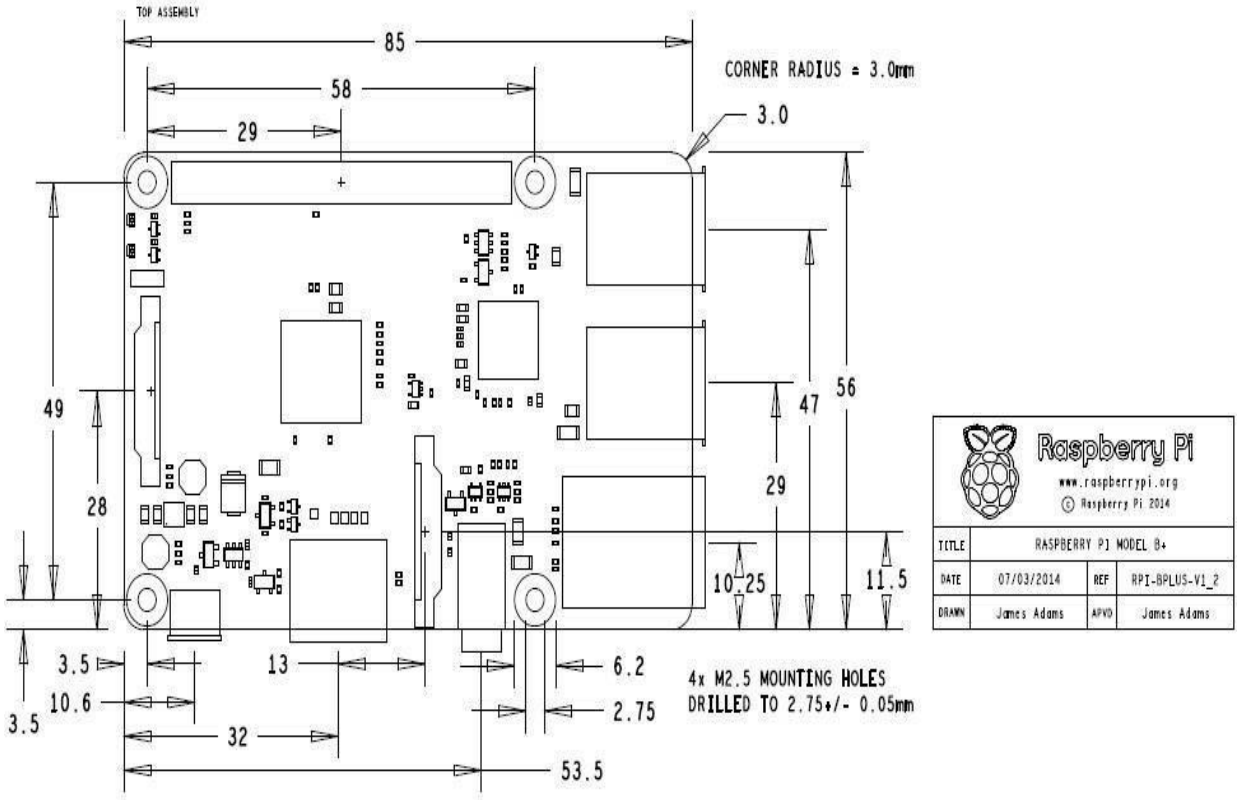


Figure 5.4: Schematic Diagram of Raspberry pi

5.1.5 Operations of Raspberry Pi B+

The Raspberry Pi 3 Model B+ is a single-board computer that can be used for a variety of purposes. Here are some of the common operations that can be performed using the Raspberry Pi 3 Model B+

a) ***Media center:*** The Raspberry Pi 3 Model B+ can be used as a media center by installing Kodi or OSMC on it. This allows you to stream movies, TV shows, and music from the internet or your local network.

b) ***Retro gaming:*** The Raspberry Pi 3 Model B+ can be used to play retro games by installing RetroPie on it. This allows you to play classic games from consoles like the NES, SNES, and Sega Genesis.

c) ***Home automation:*** The Raspberry Pi 3 Model B+ can be used for home automation by installing Home Assistant on it. This allows you to control your smart home devices from a single interface.

d) ***Programming:*** The Raspberry Pi 3 Model B+ can be used for programming in languages like Python, C++, and Java. It comes with a variety of programming tools pre-installed, including the Thonny Python IDE 2.

e) ***Robotics:*** The Raspberry Pi 3 Model B+ can be used for robotics projects by connecting it to a variety of sensors and motors. It can be programmed using languages like Python and Scratch

5.1.6 Raspberry Pi 3B+ Pin configuration

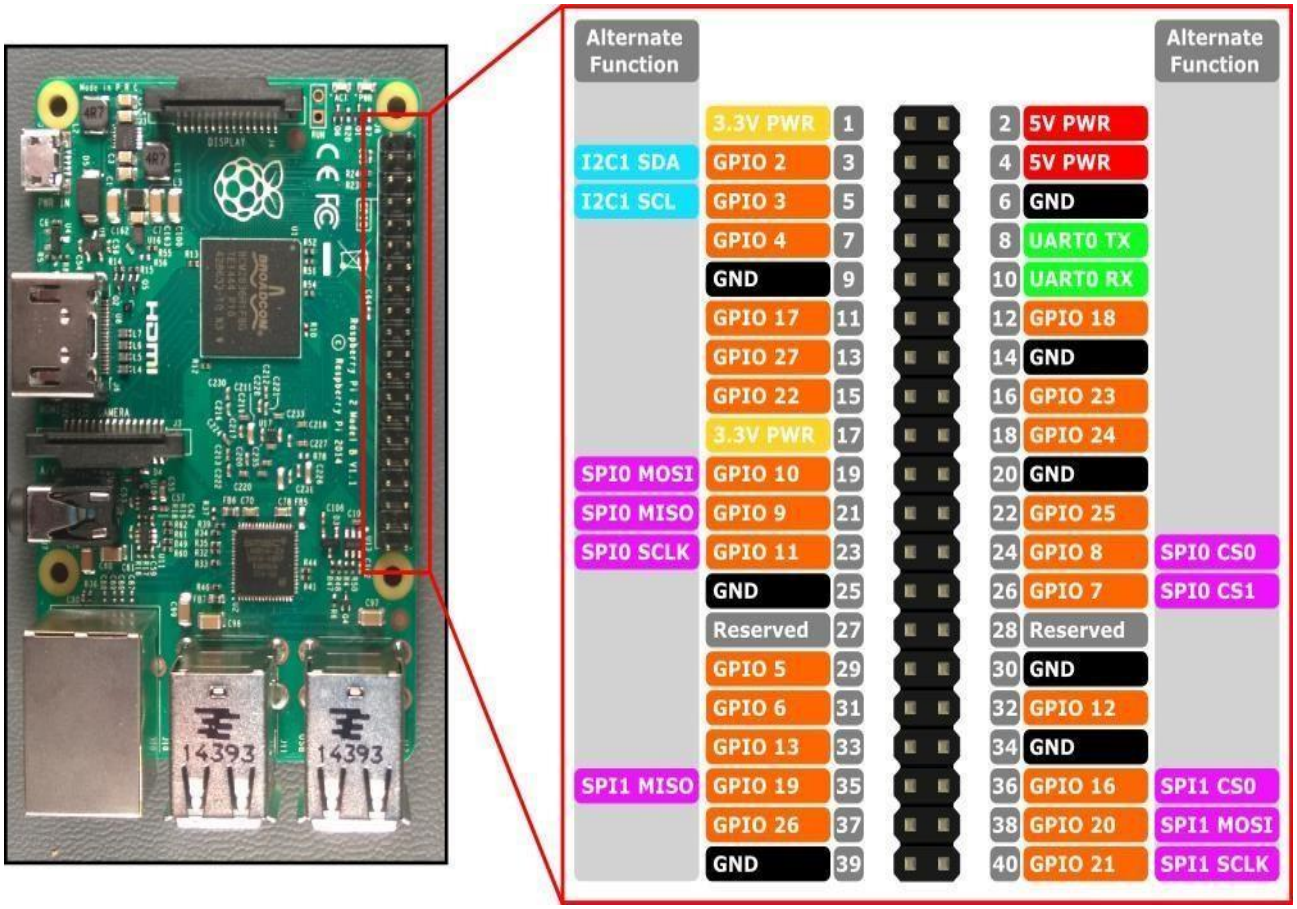


Figure 5.5: Raspberry pi 3B+ Pin configuration

The above diagram includes:

- GPIO functions
- Schematic

- Specifications

The Raspberry Pi 3 Model B+ is a single-board computer that can be used for a variety of purposes. The board has **40** pins, out of which **26** are used as digital I/O pins, and **9** of the remaining **14** pins are termed as dedicated I/O pins, which indicate they don't come with an alternative function. The power pins on the Raspberry Pi 3 Model B+ are as follows:

Figure 5.6: Processor of Raspberry pi 3B+

- Pin 2 (5V): This pin supplies 5V power to the Raspberry Pi.
- Pin 4 (5V): This pin also provides 5V of power to the Raspberry Pi.
- Pin 6 (GND): This pin is the ground pin, and it is used to complete the circuit and provide a reference voltage for the Raspberry Pi.
- Pin 14 (GND): This is another ground pin on the Raspberry Pi.

The Raspberry Pi 3 Model B+ also has PWM (pulse-width modulation) pins, SPI (Serial Peripheral Interface) pins, I2C (Inter-Integrated Circuit) pins, and UART (Universal Asynchronous Receiver-Transmitter) pins.

There are two kinds of Input and Output pin numbering for the Raspberry pi. One is the BCM and the other is BOARD. Basically, these pin numberings are useful for writing python script for the Raspberry Pi.

a) ***GPIO BOARD***
This type of pin numbering refers to the number of the pin in the plug, i.e, the numbers printed on the board, for example, P1. The advantage of this type of numbering is, it will not change even though the version of board changes.

b) ***GPIO BCM*** The BCM option refers to the pin by "Broadcom SOC Channel. They signify the Broadcom SOC channel designation. The BCM channel changes as the version number changes.

Note: It is very important to wire the GPIO pins with limited resistors to avoid serious damage to the Raspberry Pi. LEDs must have resistors to limit the current passing through them. The motors should not be connected to the GPIO pins directly.

5.1.7 Raspberry Pi Processor

The Broadcom BCM2835 SoC used in the first-generation Raspberry Pi includes a 700 MHz 32-bit ARM1176JZF-S processor, Video Core IV graphics processing unit (GPU),[50] and RAM. It has a level 1 (L1) cache of 16 KB and a level 2 (L2) cache of 128 KB. The level 2 cache is used primarily by the GPU. The SoC is stacked underneath the RAM chip, so only its edge is visible. The
ARM1176JZ(F)-S is the same CPU used in the original iPhone, although at a higher clock rate, and mated with a much faster GPU. The earlier V1.1 model of the Raspberry Pi 2 used a Broadcom BCM2836 SoC with a 900 MHz 32-bit, quad-core ARM Cortex-

A7 processor, with 256 KB shared L2 cache. The Raspberry Pi 2 V1.2 was upgraded to a Broadcom BCM2837 SoC with a 1.2
GHz 64-bit quad-core ARM Cortex-A53
processor, the same one which is used on the Raspberry Pi 3, but underclocked (by default) to the same 900 MHz CPU clock speed as the V1.1. The BCM2836 SoC is no longer in production as of late 2016.

The Raspberry Pi 3 Model B uses a Broadcom BCM2837 SoC with a 1.2 GHz 64-bit quad-core ARM Cortex-A53 processor, with 512 KB shared L2 cache. The Model A+ and B+ are 1.4 GHz

The Raspberry Pi 4 uses a Broadcom BCM2711 SoC with a 1.5 GHz (later models: 1.8 GHz) 64-bit quad-core ARM Cortex-A72 processor, with 1 MB shared L2 cache. Unlike previous models, which all used a custom interrupt controller poorly suited for virtualisation, the interrupt controller on this SoC is compatible with the ARM Generic Interrupt Controller (GIC) architecture 2.0, providing hardware support for interrupt distribution when using ARM virtualisation capabilities. The Video Core IV of the previous models has also been replaced with a Video Core VI running at 500 MHz

The Raspberry Pi Zero and Zero W use the same Broadcom BCM2835 SoC as the first-generation Raspberry Pi, although now running at 1 GHz CPU clock speed.

The Raspberry Pi Zero 2 W uses the RP3A0-AU, which is a System-inPackage (Sip) design. The package contains a Broadcom BCM2710A1 processor, which is a 64-bit quad-core ARM Cortex-A53 clocked at 1 GHz, along with 512 MB of LPDDR2 SDRAM layered above. The Raspberry Pi 3 also uses the BCM2710A1 in its Broadcom BCM2837 SoC, but clocked at a higher 1.2 GHz.

The Raspberry Pi Pico uses the RP2040, a microcontroller containing dual ARM Cortex-M0+ cores running at 133 MHz, 6 banks of SRAM totalling 264 KB, and programmable IO for peripherals.

5.1.8 Different Models in Raspberry Pi

Family	Model	SoC	Memory	Form Factor	Ethernet	Wireless	GPIO	Released	Discontinued
Raspberry Pi	B	BCM2835	256 MB	Standard[a]	Yes	No	26-pin	2012	Yes (????)
Raspberry Pi	B	BCM2835	512 MB	Standard[a]	Yes	No	26-pin	2012[38]	Yes (????)
Raspberry Pi	A	BCM2835	256 MB	Standard[a]	No	No	26-pin	2013	No
Raspberry Pi	B+	BCM2835	512 MB	Standard[a]	Yes	No	40-pin	2014	No
Raspberry Pi	A+	BCM2835	512 MB	Compact[b]	No	No	40-pin	2014	No
Raspberry Pi 2	B	BCM2836 / 7	1 GB	Standard[a]	Yes	No	40-pin	2015	No
Raspberry Pi Zero	Zero	BCM2835	512 MB	Ultra-compact[c]	No	No	40-pin	2015	No
Raspberry Pi Zero	W / WH	BCM2835	512 MB	Ultra-compact[c]	No	Yes	40-pin	2017	No
Raspberry Pi Zero	2 W	BCM2710A1[d][39]	512 MB	Ultra-compact[c]	No	Yes	40-pin	2021	No
Raspberry Pi 3	B	BCM2837A0 / B0	1 GB	Standard[a]	Yes	Yes	40-pin	2016	No
Raspberry Pi 3	A+	BCM2837B0	512 MB	Compact[b]	No	Yes[e]	40-pin	2018	No
Raspberry Pi 3	B+	BCM2837B0	1 GB	Standard[a]	Yes[f]	Yes[e]	40-pin	2018	No
Raspberry Pi 4	B	BCM2711B0 / C0[40]	1 GB	Standard[a]	Yes[g]	Yes[e]	40-pin	2019[41]	Yes (2020)[42]
Raspberry Pi 4	B	BCM2711B0 / C0[40]	1 GB	Standard[a]	Yes[g]	Yes[e]	40-pin	2021[43]	No
Raspberry Pi 4	B	BCM2711B0 / C0[40]	2 GB	Standard[a]	Yes[g]	Yes[e]	40-pin	2019[41]	No
Raspberry Pi 4	B	BCM2711B0 / C0[40]	4 GB	Standard[a]	Yes[g]	Yes[e]	40-pin	2019[41]	No
Raspberry Pi 4	B	BCM2711B0 / C0[40]	8 GB	Standard[a]	Yes[g]	Yes[e]	40-pin	2020	No
Raspberry Pi 4	400	BCM2711B0 / C0[40]	4 GB	Keyboard	Yes[g]	Yes[e]	40-pin	2020	No
Raspberry Pi Pico	Pico	RP2040	264 KB	Pico[h]	No	No	26-pin	2021	No
Raspberry Pi Pico	W	RP2040	264 KB	Pico[h]	No	Yes[i]	26-pin	2022	No
Raspberry Pi 5[44]		BCM2712	4 GB	Standard[a]	Yes[g]	Yes[e]	40-pin	2023	No
Raspberry Pi 5[44]		BCM2712	8 GB	Standard[a]	Yes[g]	Yes[e]	40-pin	2023	No

Figure 5.7: Different Models of Raspberry pi

5.1.9 Raspberry Pi Operating System

Raspberry Pi board, you need to provide with an OS (operating system). **Linux** is the most frequently used OS on the Raspberry Pi. For using an OS, we need to create a Secure Digital (SD) or MicroSD card with an OS on it. The prerequisite for setting up the SD or MicroSD is a computer having an internet connection and the ability to write to SD or MicroSD cards.

NOOBS Software

NOOBS means **new-out-of-box software** and it is the easiest way to get started with the Raspberry Pi. It is easy to copy NOOBS to your SD or MicroSD card. Once copied, it provides us with a simple menu for installing various operating systems. There is an option to buy a card with NOOBS already installed on it, but it is always useful to know how to create your own NOOBS cards.

Download NOOBS:

Follow the below given steps to download NOOBS −

- **Step 1** − Go to the website www.raspberrypi.org/downloads/noobs
- **Step 2** − Select from the two versions of NOOBS available. Version 1 is the main version and includes Raspbian. This is the officially supported OS, which you can use even without any network connection.

Another option is to choose the OS from the menu. You can download and install the OS from the menu, if you have a network connection. It is always recommended to download NOOBS for your first OS.

MicroSD card Formatting

Before downloading and installing OS, we first need to format our SD or MicroSD card. We can use an application program, called SD card Formatter, from SD Association. The latest version is SD Memory Card Formatter 5.0.1.

For Windows and Mac, it can be downloaded from the link https://www.sdcard.org/downloads/formatter/.

Let us see how we can format the SD card by using windows, Mac OS, and Linux.

Using Windows

Step 1 − Download and install the SD formatter application. It will be as follows

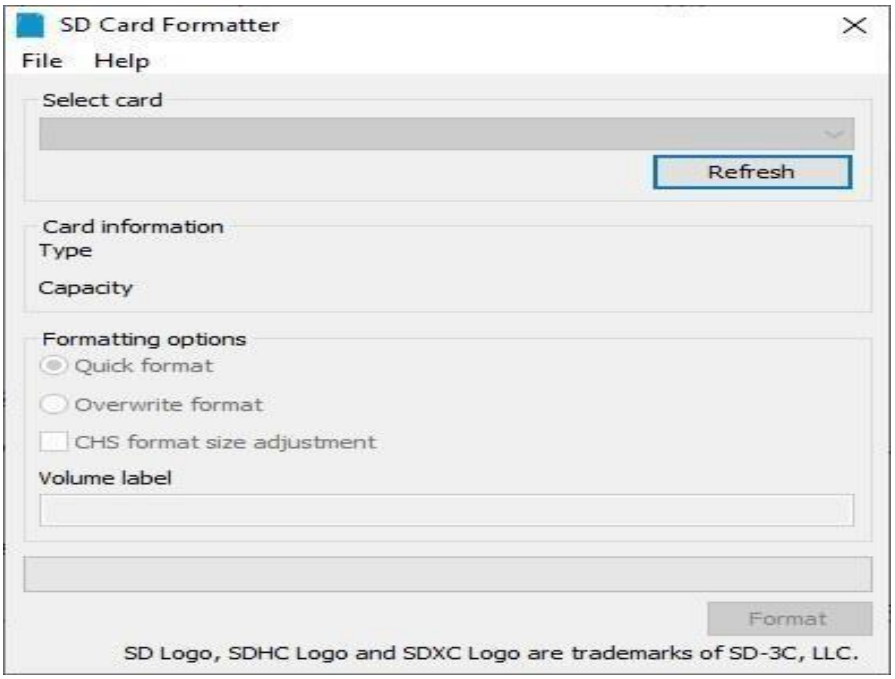


Figure 5.8: Step1 of Raspberry pi 3B+ OS installation

Step 2 − Next, we need to select the drive in which we have our SD High Capacity SDHC/SDXC card. Once selected, click on the format button to format it.

The following screen will appear −

Figure 5.9: Step2 of Raspberry pi 3B+ OS installation

Step 3 − The program will ask for the confirmation. You need to click **yes** to confirm the format process.

Step 4 − Once the format process is completed, your SD card will be formatted completely.

Using Mac OS

The process of formatting is similar as we did in windows. You just need to download and install the Mac version of SD card formatter.

Using Linux

We will be using the **GParted**application program, which is an open-source partition manager for Linux.

Use the steps given below to format a SD card in Ubuntu software −

Step 1 − Download and install the **GParted** application by using the terminal as follows −

sudo apt-get install gparted

Step 2 − Once installation is completed, you need to insert the SD card. Next, by using Unity dash, launch the **GParted** application.

Step 3 − You will get the screen as below, which shows the partitions of the removable disk. But before starting the formatting, we need to unmount the disk by right-clicking on the partition as shown below −

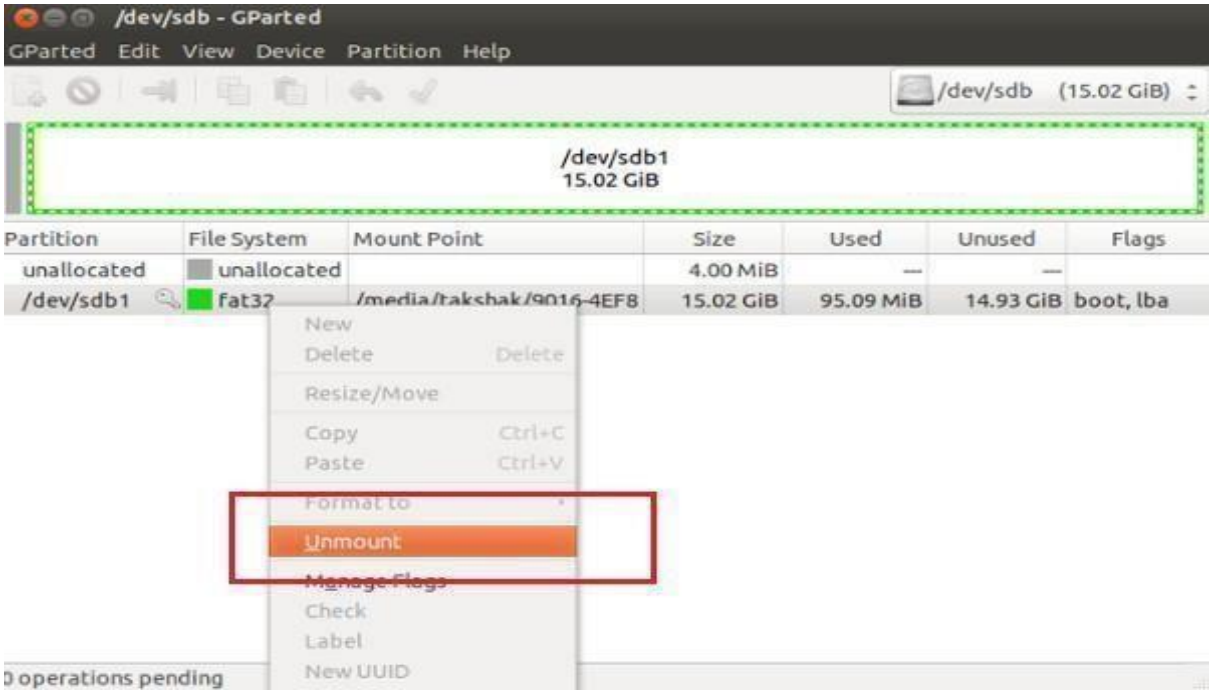


Figure 5.10: Step3 of Raspberry pi 3B+ OS installation

Step 4 − After unmounting, we need to right click on it, which will show us the **Format to** option. Now from the list, you can choose whatever type of file system you want on the disk.

After selecting the drive to format, you need to click on the **Tick sign** as shown below

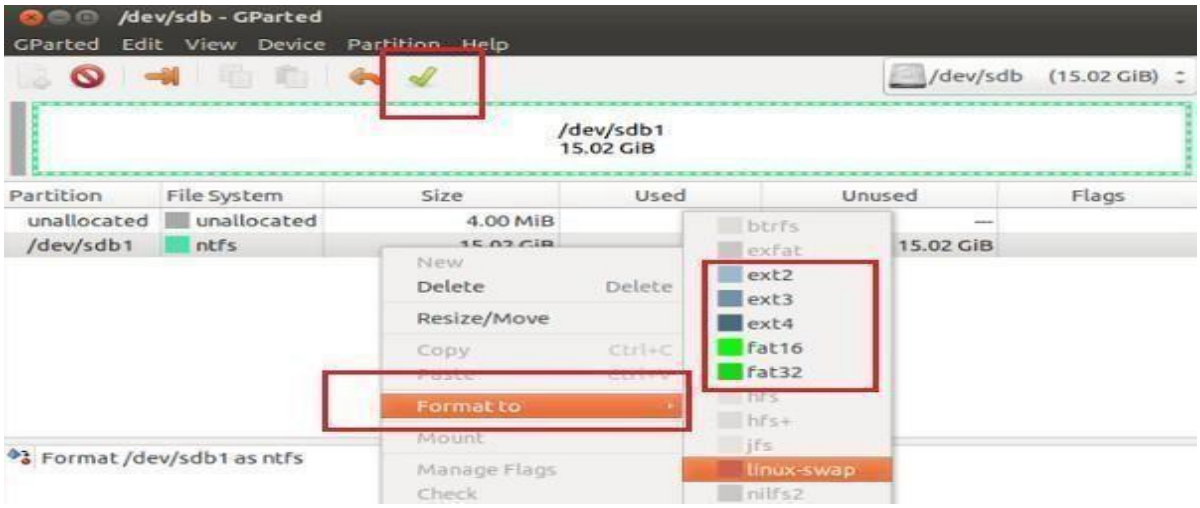


Figure 5.11: Step4 of Raspberry pi 3B+ OS installation

Step 5 − It will show you a couple of warnings and the format procedure will be started.

Install NOOBS to Memory card

Now, you have a formatted card and the .zip file that was downloaded from the Raspberry website. Hence, you can install NOOBS on your card.

On windows PC, you can simply double click the .zip file. It will open the file. Once opened, you can select all the files and copy them to your formatted card.

Similarly, on a Mac OS, you can see the folder that contains all the files by double clicking on the NOOBS .zip file. Now, click on the **Edit menu** and select all. Drag all the files onto your SD card.

In the same way, on Linux we can use the desktop environment to copy the NOOBS .zip files to our SD card.

Flashing a MicroSD card

Some operating systems (OS) may not be available through NOOBS. One of them is the Reduced Instruction Set Computer (RISC) OS.

For creating a card for such an OS, we need to first download the OS as an image file. Once an image file is downloaded, we need to use the process called flashing your card. Later on, the single file can be converted into all the files which we need on our card (SD or MicroSD).

To download the OS images, we can find the links at the website https://www.raspberrypi.org/software/.
Now to flash the card or you can say burning an image to the card, we can use an OS image flasher **Etcher**. It is available for windows, Mac OS and Linux at https://www.balena.io/etcher/.

Raspbian configuration

For configuring Raspberry Pi in Raspbian, we are using Raspbian with PIXEL desktop. It is one of the best ways to get Raspbian started with the Raspberry Pi. Once we finish booting, we will be in the PIXEL desktop environment.

Now to open the menu, you need to click the button that has the Raspberry Pi logo on it. This button will be in the top left. After clicking the button, choose **Raspberry Pi configuration** from the preferences.

Configuration tool

Following is the configuration tool in PIXEL desktop −

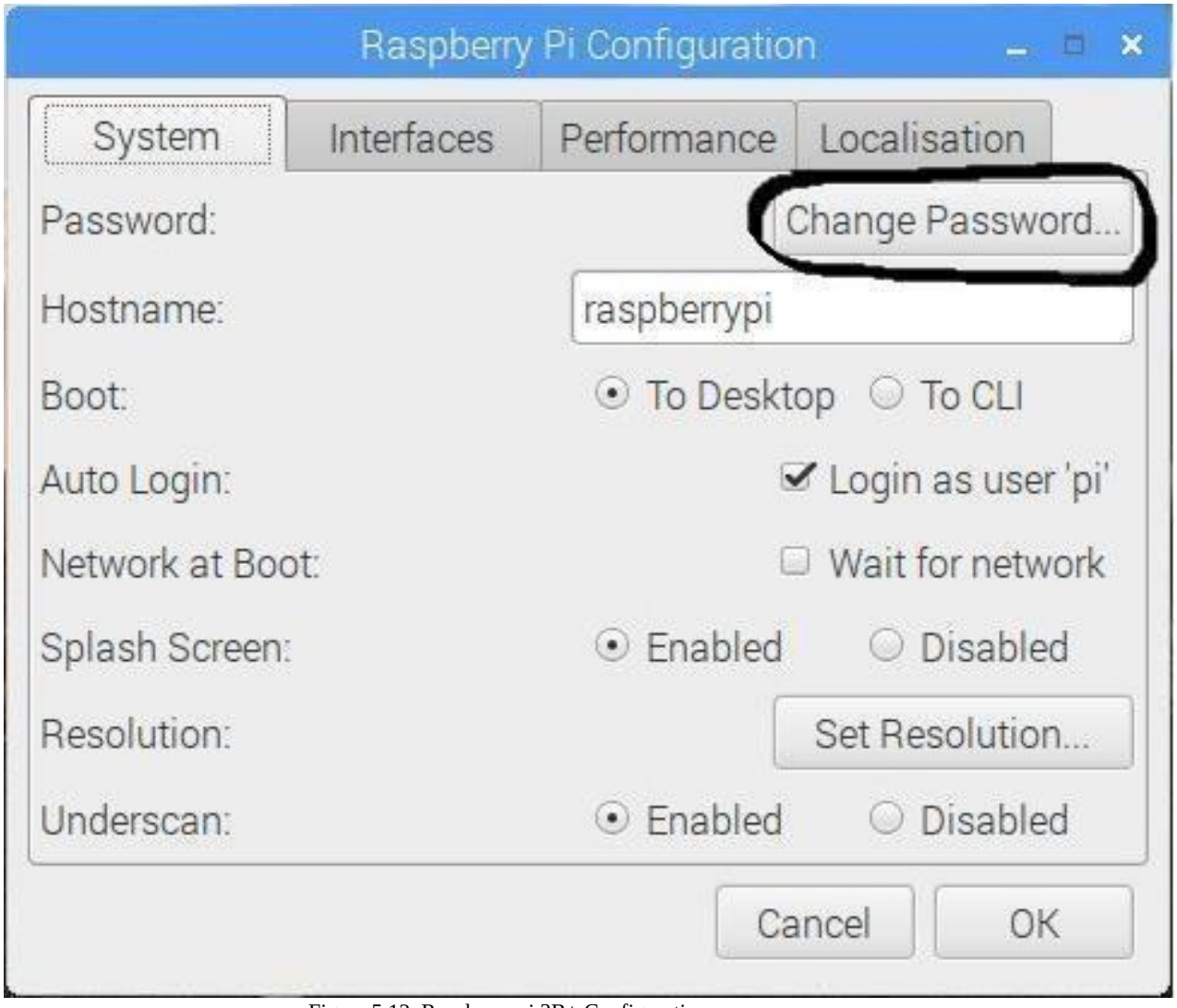


Figure 5.12: Raspberry pi 3B+ Configuration

By default, the configuration tool opens to its system tab which has the following options −

- **Change Password** − The default password is **raspberry**. You can change it by clicking the change password button.
- **Change the hostname** − The default name is **raspberry pi**. You can also change it to the name, which you want to use on the network. • **Boot** − You can choose from the two options and control whether Raspberry Pi boots into the desktop or CLI i.e., command line interface.
- **Auto Login** − With the help of this option, you can set whether the user should automatically log in or not.
- **Network at Boot** − By choosing this option, you can set whether the pi user is automatically logged in or not.
- **Splash screen** − You can enable or disable it. On enabling, it will display the graphical splash screen that shows when Raspberry Pi is booting.
- **Resolution** − With the help of this option, you can configure the resolution of your screen.
- **Underscan** − There are two options, enable or disable. It is used to change the size of the displayed screen image to optimally fill the screen. If you see a black border around the screen, you should disable the under scan. Whereas, you should enable the underscan, if your desktop does not fit your screen.

There are three other tabs namely Interfaces, Performance, and Localization. The job of interface tab is to enable or disable various connection options on your Raspberry Pi.

You can enable the Pi camera from the interface tab. You can also set up a secure connection between computers by using SSH (short for Secure Shell) option.

If you want to remote access your Pi with a graphical interface then, you can enable RealVNC software from this tab. SPI, I2C, Serial, 1-wire, and Remote GPIO are some other interfaces you can use.

There is another tab called Performance, which will give you access to the options for overclocking and changing the GPU memory.

The localization tab, as the name implies, enable us to set −

- The character set used in our language.
- Our time zone.
- The keyboard setup as per our choice.
- Our Wi-Fi country.

Configure Wi-Fi

You can check at the top right, there would be icons for Bluetooth and Wi-Fi. The fan-shaped icon is on the Wi-Fi. To configure your Wi-Fi, you need to click on that icon. Once clicked, it will open a menu showing the available networks. It also shows the option to turn off your Wi-Fi.
Among those available networks, you need to select a network. After selecting, it will prompt for entering the Wi-Fi password i.e., the Pre-Shared Key.

If you see a red cross on the icon, it means your connection has been failed or dropped. To test whether your Wi-Fi is working correctly, open a web browser and visit a web page.

Configure Bluetooth Devices

We can use wireless Bluetooth devices such as keyboard and/or mouse with Pi 3 and Pi zero W because these models are Bluetooth-enabled. In PIXEL desktop, you can set up your Bluetooth devices easily.

Following are the steps to configure the Bluetooth devices − •

First, make your device discoverable for pairing.

- Now, you need to click on the Bluetooth menu at the top right of the screen. It is aligned to the Wi-Fi button.
- Now, choose the Add Device option.
- The Raspberry will start searching for the devices and when it finds your device, click it and click the pair button.

Data Partition Setup

As we know that data partition is that area on your memory card (SD or MicroSD) which can be shared by various distributions. One of the best examples of use of a data partition is transferring the files between distributions.

The data partition has the **label** data.

You can use this labelled data to make a directory point to it as follows −

Step 1 − First, you need to boot the Raspberry Pi into Raspbian.

Step 2 − Now, click the Terminal icon to get to the command line.

Step 3 − Next, type the command **mkdir shared**. It will create a directory named **shared**.

Step 4 − Write the command **sudo mount -L data shared**. This command will point the directory to the shared partition.

Step 5 − Write the command **sudo chown $USER: shared**. It will set the permission for writing in this shared folder.

Step 6 − Now, to go to this shared folder, you need to type the command **cd shared**. Once all the files are created in this shared folder, they will be available to all the distributions that have the permission to access the data partition.

5.1.10 Raspberry pi Desktop Overview

PIXEL Desktop Environment

PIXEL (Pi Improved Xwindows Environment, Lightweight) is a visual desktop environment which is a part of the recommended Raspbian Linux distribution. It is the quickest way to get started with Raspberry Pi and by default, it appears when our Raspberry Pi computer finishes starting up.

Some of the characteristics of PIXEL are as follows −

- It is based on LXDE (Lightweight X11 Desktop Environment) open-source desktop.
- Raspberry Pi foundation has redesigned LXDE and converted it into a PIXEL desktop environment.
- The PIXEL desktop environment works in a similar way to Mac OS and Windows OS.
- We can manage and find files by using mouse and icons. • Using this desktop environment, it's really intuitive to navigate.

Navigate Desktop Environment

The image below is of the PIXEL desktop environment. You can see a taskbar (a strip along the top of the screen), which is usually visible in every program we will be using.

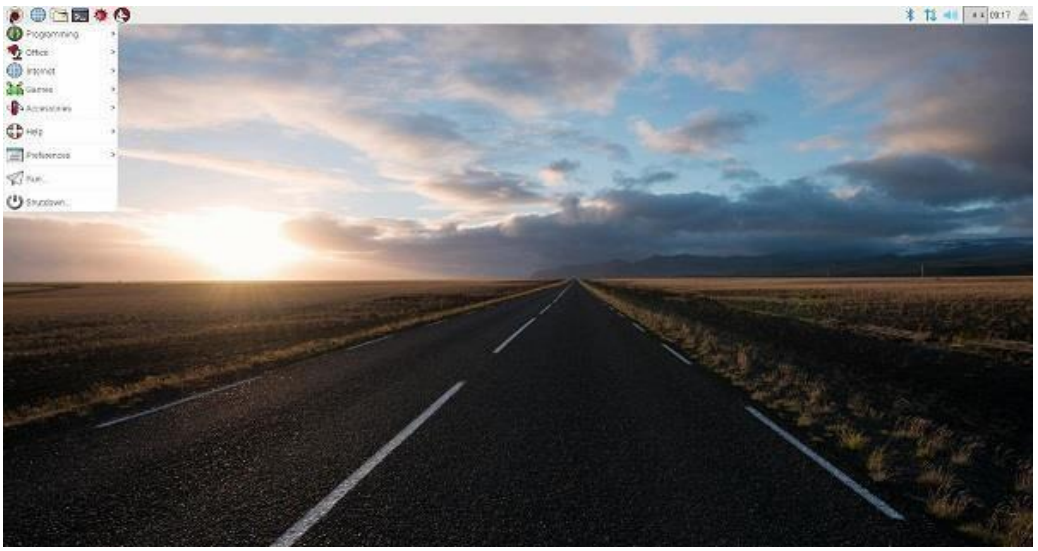


Figure 5.13: Raspberry pi 3B+ Desktop environment

The Application Menu

For most of the programs, which we would like to run under PIXEL desktop or any other desktop environment, we need to use the application menu.

You can get it by clicking the Raspberry Pi icon at the top left side of the desktop screen.

You will see the image as follows −

Figure 5.14: Raspberry pi 3B+ Application menu

Submenu Programs

You will get the submenu program on the right, after moving the cursor over the categories of the programs. It will show the programs in that particular category.

You need to click on that category to start with that. If you want to add that category icon to the desktop, just right-click that program on the menu.

Following are the wealth of programs under submenu program −

Claws Mail

It is in the internet part of the Application with the help of which you can send or receive messages on your Raspberry Pi computer.

Debian Reference

As we have discussed earlier, the Raspbian version of Linux is the Pi-specific version of the Debian distribution. This icon will guide us how to use Linux on your Raspberry Pi computer.

This is a reference document, which is stored on your SD card and to find this, you need to go through the help section of the Application menu.
To get started with this, first, you need to click the icon and then, click the multi-files link (it is an HTML link) which is at the top of the screen.

LibreOffice

This is the most popular suite of productivity applications. It mainly includes word processing, spreadsheets, and presentations. You can get it from the office section of the Application menu.

Mathematica

Mathematica, under the programming section of the Application menu, is based on the Wolfram programming language. It is used for scientific and technical computing.

Minecraft Pi

We know about the world-building game called Minecraft. Similarly, Minecraft Pi is the Raspberry version of that. You can find it under the Game section of the Application program, and you can also program it by using the Python programming language.

Python 2 and Python 3

Raspberry Pi provides us the Python programming language, which can be found under Programming in the Application menu. We can also use the Thonny IDE (integrated development environment) which provides the Pi users, an alternative way of creating Python programs.

Python games

Raspberry Pi has games such as Reversi, Four in a Row, a sliding puzzle game as well as a snake game. These all are built in Python programming language and can be found in the Game section of the Application menu.

Scratch

Raspberry Pi foundation provides us a simple programming language called Scratch, which is approachable for the peoples of all ages. You can use it to create games and animations. It can also be used to manage electronic projects. You can find it under the Programming section of the Application menu.

Sense HAT emulator

As the name implies, it has some built-in sensors that can be used for creating experiments and other projects. It is an add-on for the Raspberry Pi users, which can be found under the Programming section of the Application menu.

Shutdown

Shutdown, a top-level option in the Application menu, can be used for switching off your Raspberry Pi, before you remove the power. With this, we will also get the options to log out as well as restart your Pi computer.

Sonic Pi

It is another programming language provided by Raspberry Pi foundation which is mainly used for creating music. You can also find it under the Programming section of the Application menu.

Terminal

Terminal is a window that let us issue the instructions from a command line without leaving your PIXEL desktop environment. There are two ways through which you can reach the terminal window. One is to get it in the Accessories part of the Programs menu and other is to use the button on the taskbar.

Wolfram

Wolfram is a programming language provided by Raspberry Pi foundation. It aims to incorporate knowledge, so that the programmers can get results quickly. You can get more information about this at www.wolfram.com/language. It is under the Programming section of the Application menu.

Running programs

Even after installing, some of the programs won't appear on the Application menu. You can use the **Run option** to run those programs.

Follow the below given steps −

- **Step 1** − First, we need to open the Application menu. For this, click the icon at top left of the desktop.
- **Step 2** − Now, we need to select the Run option from this menu.
- **Step 3** − Run option will give you a dialog box. You can type the name of the program, which you want

5.2 SD CARD

An SD card, which stands for Secure Digital card, is a small, removable storage device that is commonly used to store digital data. SD cards are widely used in various electronic devices, including cameras, smartphones, tablets, laptops, digital audio players, gaming consoles, and more.

Here are some key points about SD cards:

Figure 5.15: SD Card

a) Types and Sizes:

SD cards come in various physical sizes and types, including:

- SDSC (Standard Capacity): These are the original SD cards with a storage capacity of up to 2GB.
- SDHC (High Capacity): These cards have a storage capacity of 2GB to 32GB.
- SDXC (Extended Capacity): These cards can store up to 2TB of data.
- MicroSD: These are smaller cards primarily
- used in smartphones, action cameras, and other compact devices.

b) Speed Classes:

SD cards have speed classes that indicate their minimum data transfer rates. Classes include: Class 2, 4, 6, and 10--These numbers represent the minimum write speeds in megabytes per second (MB/s).
UHS (Ultra High Speed): UHS-I and UHS-II cards provide higher data transfer rates, suitable for high-definition video recording and fast data transfer.

c) Applications:

SD cards are used for various applications, including storing photos and videos, expanding storage on smartphones and tablets, running operating systems on single-board computers like the Raspberry Pi, and recording video in digital cameras and camcorders.

d) Compatibility:

Ensure that your device is compatible with the type and capacity of the SD card you plan to use. Some devices may not support higher-capacity SDXC cards, and older devices may not be compatible with newer SD card standards.

e) Data Transfer and Backup:

SD cards can be used to transfer data between devices or to create backups of important files. They are also handy for transferring photos from a camera to a computer.

f) File Systems:

SD cards are typically formatted with file systems like FAT32 or exFAT, depending on their capacity. Ensure compatibility with your devices and operating systems.

g) Write Protection:

Some SD cards have a physical write protection switch that prevents data from being written or modified. This can be useful to protect valuable data from accidental erasure.

h) Care and Handling:

Handle SD cards with care, avoiding physical damage, and protecting them from extreme temperatures, moisture, and static electricity.

i) Storage and Lifespan:

It's essential to store SD cards in a cool, dry place and to use them regularly to maintain their lifespan. Like all flash storage, SD cards have a limited number of write cycles.

j) Data Recovery:

In case of accidental data loss, there are data recovery tools and services that may help recover lost data from an SD card.

SD cards are a convenient and portable way to expand storage, transfer data, and back up files. Selecting the right type and capacity for your specific needs and ensuring proper care and maintenance will help you get the most out of your SD cards.

5.2.1 Set up your SD card

If you have an SD card that doesn't have the Raspberry Pi OS operating system on it yet, or if you want to reset your Raspberry Pi, you can easily install Raspberry Pi OS yourself. To do so, you need a computer that has an SD card port — most laptop and desktop computers have one.

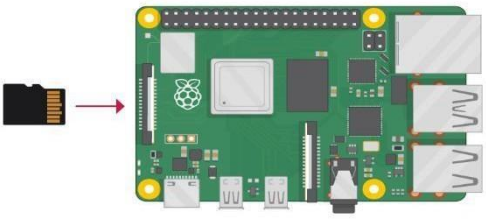


Figure 5.16: SD Card Setup

The Raspberry Pi OS operating system via the Raspberry Pi Imager

Using the Raspberry Pi Imager is the easiest way to install Raspberry Pi OS on your SD card.

Note: More advanced users looking to install a particular operating system should use this guide to installing operating system images.

Installation of Raspberry Pi Imager

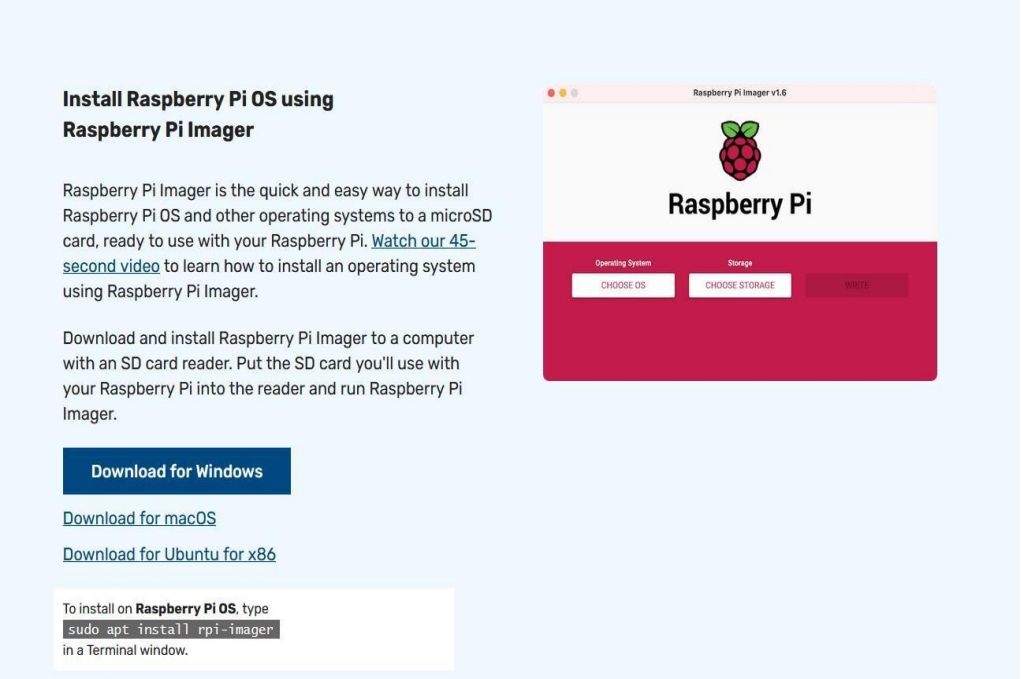


Figure 5.17: Step1 Raspberry pi Imager OS installation in Sd card

- Visit the Raspberry Pi downloads page
- Click on the link for the Raspberry Pi Imager that matches your operating system

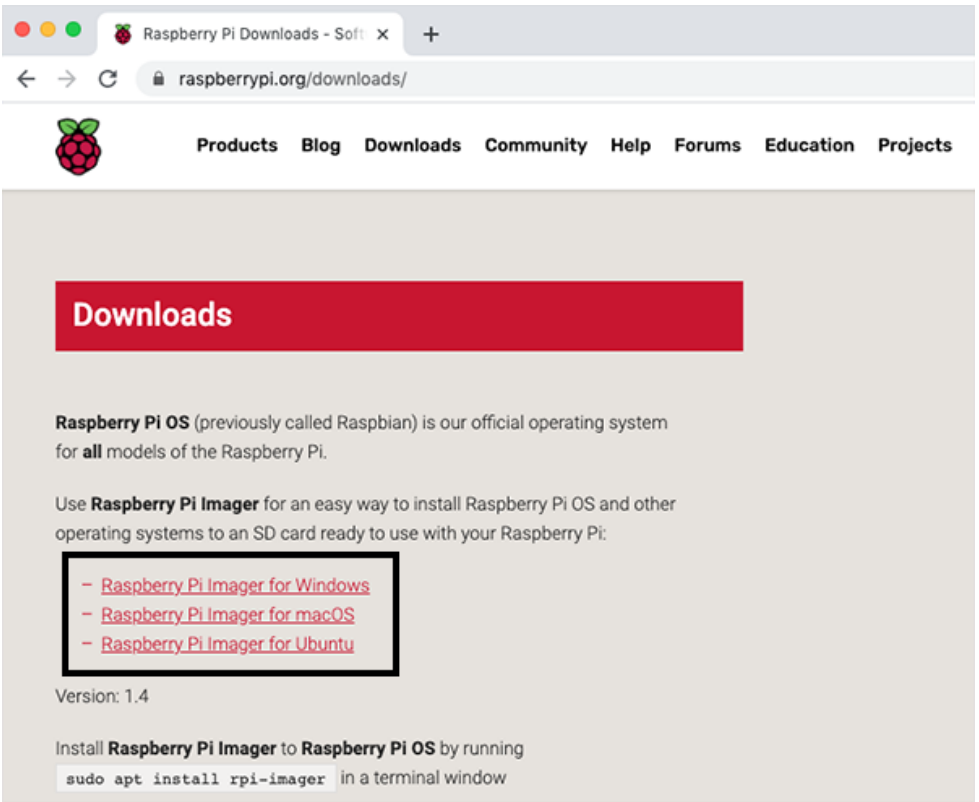


Figure 5.18: Step2 Raspberry pi Imager OS installation in Sd card

- When the download finishes, click it to launch the installer

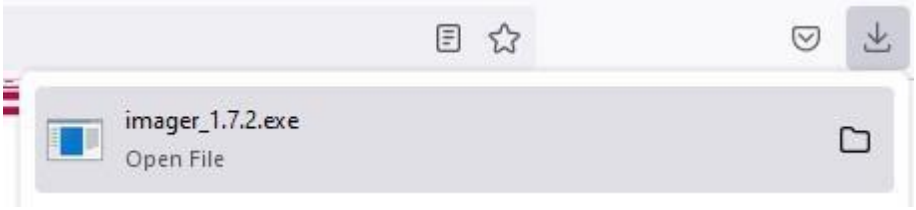


Figure 5.19: Step3 Raspberry pi Imager OS installation in Sd card

Using the Raspberry Pi Imager

Anything that's stored on the SD card will be overwritten during formatting. If your SD card currently has any files on it, e.g., from an older version of Raspberry Pi OS, you may wish to back up these files first to prevent you from permanently losing them.

When you launch the installer, your operating system may try to block you from running it. For example, on Windows I receive the following message:

Figure 5.20: Step4 Raspberry pi Imager OS installation in Sd card

- If this pops up, click on and then
- Follow the instructions to install and run the Raspberry Pi Imager
- Insert your SD card into the computer or laptop SD card slot
- In the Raspberry Pi Imager, select the OS that you want to install and the SD card you would like to install it on

Note: You will need to be connected to the internet the first time for the the Raspberry Pi Imager to download the OS that you choose. That OS will then be stored for future offline use. Being online for later uses means that the Raspberry Pi imager will always give you the latest version.

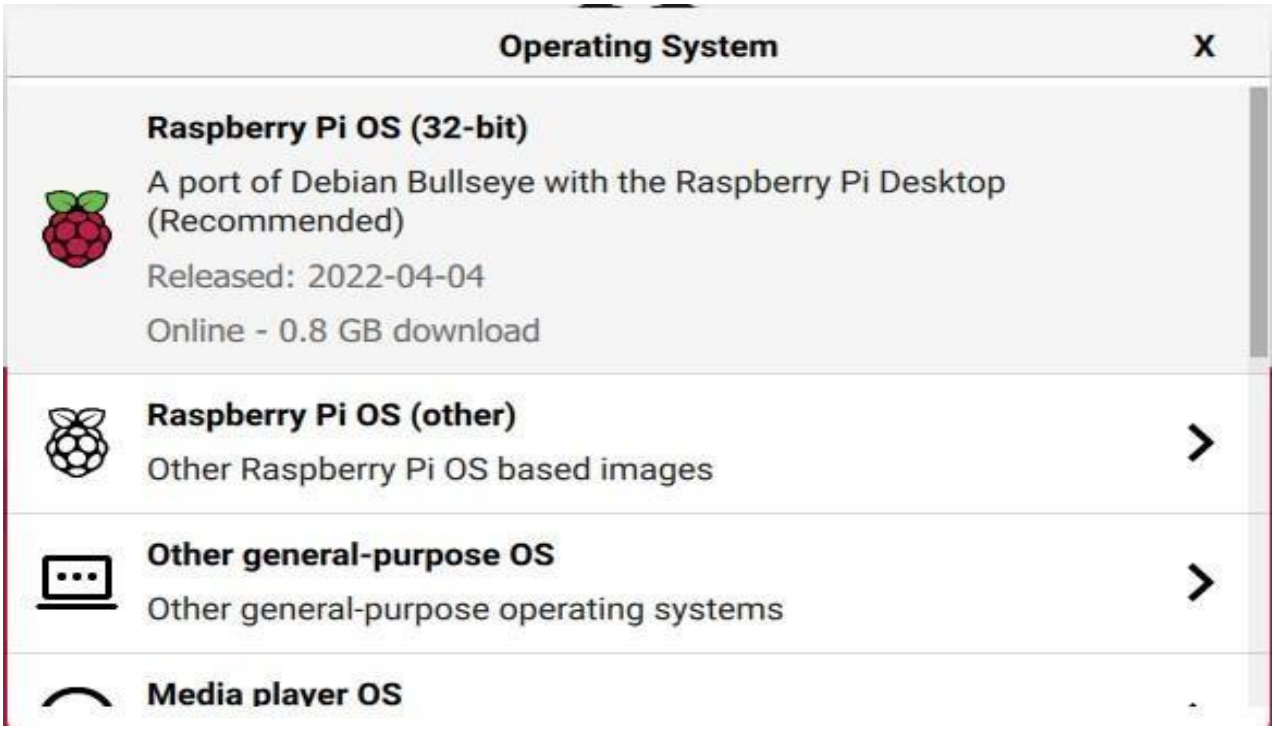


Figure 5.21: Step5 Raspberry pi Imager OS installation in Sd card

Figure 5.22: Step6 Raspberry pi Imager OS installation in Sd card

Figure 5.23: Step1 Raspberry pi Imager OS installation in Sd card

- Then simply click the `WRITE` button
- Wait for the Raspberry Pi Imager to finish writing
    - Once you get the following message, you can eject your SD card

5.3 USB CABLE

A USB (Universal Serial Bus) cable is a common type of cable used for connecting various electronic devices to each other or to a computer. USB cables come in different shapes and sizes, with the most recent standard being USB Type-C, which is a reversible, versatile, and widely adopted connection.

5.3.1 General steps for using a USB cable in different scenarios

a) Identify the USB Cable Type:

Determine the type of USB cable you're using, such as USB Type-A, USB Type-B, USB Micro-B, USB Mini-B, or the newer USB Type-C. The type of cable you need depends on the devices you want to connect.

Figure 5.24: USB Cable

b) Plug It In:

Connect one end of the USB cable to the USB port on your computer or a USB charger, depending on the purpose of the connection.

c) Connect the Other End:

Connect the other end of the USB cable to the device you want to connect. The type of port on your device should match the cable's connector. For example, if you're connecting a smartphone, you'll typically use a USB TypeA to USB Micro-B or USB Type-C cable.

d) Power and Data Transfer:

USB cables can be used for both power and data transfer, depending on the specific devices and the type of USB cable. If you're connecting a smartphone to a computer, it can charge your phone and transfer data between your device and the computer. If you're connecting a USB storage device, it can transfer data.

e) Confirm Connection:

Check whether the connected device is recognized by your computer or charger. The device may prompt you with options for data transfer, charging, or other actions depending on the connection type.

f) Safely Eject or Disconnect:

If you're connecting a storage device (e.g., a USB flash drive or an external hard drive), be sure to safely eject or disconnect the device before physically removing the cable. This prevents data corruption and ensures the device is safely disconnected.

g) Unplug the Cable:

When you're finished using the connected device, safely unplug the USB cable. If you're using it for charging, disconnect it when the device is fully charged.

h) Store the Cable:

Properly store the USB cable to prevent damage or tangling. You can use cable organizers or Velcro straps to keep cables organized.

USB cables are versatile and used for a wide range of devices, such as smartphones, tablets, cameras, external hard drives, printers, and more. The specific steps may vary depending on the devices involved, so always follow the manufacturer's instructions for the best results. Additionally, USB Type-C cables have become increasingly popular for their reversible design and support for faster data transfer and power delivery.

5.4 SD CARD READER

An SD card reader is a hardware device or component that allows you to read and write data to Secure Digital (SD) cards. SD cards are commonly used for storing and transferring data in various electronic devices, including cameras, smartphones, laptops, tablets, and more. An SD card reader provides a means to access the data stored on an SD card by connecting it to a computer or other compatible device.

5.4.1 Here are some key aspects of SD card readers

a) ***Types of SD Cards:*** SD cards come in various sizes and types, including standard SD, miniSD, microSD, and different capacities (e.g., SDHC, SDXC). SD card readers may have multiple slots to accommodate different card sizes and types.

b) ***Connection Types:*** SD card readers can connect to devices via various interfaces, including USB, USB-C, Thunderbolt, and even built-in card readers in laptops and some desktop computers.

c) ***Data Transfer Speed:*** The speed at which data is transferred between the SD card and the connected device may vary depending on the reader's specifications. Faster SD card readers can significantly reduce the time it takes to transfer large files.

d) ***Compatibility:*** Ensure that the SD card reader is compatible with the type and capacity of SD cards you plan to use. Some readers are designed specifically for microSD cards and may require an adapter for larger SD card formats.

e) ***Plug and Play:*** Most SD card readers are plug-and-play devices, which means they don't require additional drivers or software installations. Once connected to a compatible device, it should be recognized and accessible.

f) ***Use Cases:*** SD card readers are widely used for various purposes, including transferring photos and videos from digital cameras, backing up data from mobile devices, expanding storage on laptops and tablets, and sharing data between devices.

g) ***Operating Systems:*** SD card readers are typically compatible with a range of operating systems, including Windows, macOS, Linux, and others. However, it's a good practice to check compatibility with your specific operating system.

h) ***SD Card Management Software:*** Some SD card readers come with proprietary software for managing and organizing the data on your SD cards. This software may also include features for data recovery and backup.

i) ***SD Card Duplication:*** Some specialized SD card readers are designed for duplicating or cloning multiple SD cards simultaneously, making them useful in settings like data backup and content distribution.

j) ***Card Reader Accessories:*** Depending on the reader's design, it may come with additional features such as LED indicators, multiple card slots, protective covers, and built-in cable storage.

5.4.2 Applications and Uses of Sd Card Reader in Installing Raspberry Pi OS

An SD card reader is an essential tool when it comes to installing an operating system (OS) on a Raspberry Pi. The Raspberry Pi is a popular single-board computer that uses an SD card as its primary storage medium for the operating system and data.

Here are the applications and uses of an SD card reader in the context of installing an OS on a Raspberry Pi:

a) ***Writing the Raspberry Pi OS Image:*** To set up a Raspberry Pi, you need to write the Raspberry Pi OS (formerly known as Raspbian) image file onto an SD card. An SD card reader is used to connect the SD card to a computer or laptop, allowing you to write the OS image to the card. This process involves formatting the SD card and copying the necessary files.

b) ***Downloading and Preparing OS Images***: You can download the Raspberry Pi OS image from the official Raspberry Pi website or other trusted sources. Once downloaded, you'll use an SD card reader to write this image to the SD card. You may also need to unzip or extract the downloaded file before writing it to the SD card.

c) ***Verifying and Backing Up SD Card Data:*** Before writing the OS image to the SD card, it's a good practice to verify and back up any data on the card. An SD card reader is useful for connecting the card to a computer to perform these tasks.

d) ***Choosing the Right OS for Your Project:*** Depending on your specific project or application, you may need to install a different OS image on the Raspberry Pi. An SD card reader makes it easy to switch between different OS images by simply swapping out the SD card.

Figure 5.25: SD Reader connection

e) ***Installing Custom OS Builds:*** If you're working with custom or specialized OS builds for the Raspberry Pi, you'll use an SD card reader to write these custom images to the SD card.

f) ***Flashing Tools:*** Some Raspberry Pi enthusiasts prefer using specialized software tools like Etcher or Win32 Disk Imager to write OS images to SD cards. An SD card reader is needed to connect the card to the computer during the flashing process.

g) ***Updating the OS:*** After the initial installation, you can use an SD card reader to update the Raspberry Pi OS by writing new versions or applying updates to the SD card.

h) ***Troubleshooting and Data Recovery:*** In case of any issues with the Raspberry Pi or the SD card, an SD card reader can be used to access and recover data, reformat the card, or flash a fresh OS image.

i) ***Reusing SD Cards:*** SD cards may be reused for different Raspberry Pi projects. An SD card reader allows you to easily swap out cards, each with a different OS or project configuration.

j) ***Testing and Development:*** When developing software or testing various configurations on a Raspberry Pi, you can have multiple SD cards with different OS setups. An SD card reader makes it convenient to switch between these setups.

Chapter-6 Sensors and Data Collection Devices

6.1 IR SENSOR

An IR sensor, also known as an infrared sensor, is a device that can detect and measure infrared radiation in its surroundings. Infrared radiation is part of the electromagnetic spectrum, which includes wavelengths longer than those of visible light. Infrared sensors are commonly used in various applications for detecting heat, motion, and proximity.

Figure 6.1: IR Sensor

6.1.1 Types of IR Sensors

There are several types of IR sensors each designed for specific purposes:

a) ***Infrared Thermometers:*** Infrared thermometers, also known as non-contact thermometers or IR thermometers, are used to measure the temperature of objects without making physical contact. They work by detecting the thermal radiation emitted by an object and converting it into a temperature reading.

b) ***Infrared Motion Sensors:*** Infrared motion sensors, often used in security systems and automatic lighting, can detect changes in the heat signature within their field of view. When a warm object, such as a person or an animal, moves within the sensor's range, it triggers an output signal, typically used to activate an alarm or turn on lights.

c) ***Infrared Proximity Sensors:*** These sensors are used to detect the presence or proximity of an object without physical contact. They emit infrared light and measure the reflection or absence of that light to determine the object's distance. Infrared proximity sensors are used in various applications, including obstacle detection in robotics and touchless interfaces in consumer electronics.

d) ***Infrared Receivers (IR Receivers***): IR receivers are commonly used in remote control devices, like TVs and home entertainment systems. They can detect infrared signals transmitted by remote controls and decode the signals to perform specific actions, such as changing the channel or adjusting the volume.

e) ***Infrared Imaging Sensors:*** Infrared imaging sensors, often referred to as thermal imaging cameras, capture infrared radiation emitted by objects to create thermal images. These cameras are used in various applications, such as detecting heat leaks in buildings, locating people in the dark, and monitoring equipment for overheating.

f) ***Infrared Gas Sensors:*** Infrared gas sensors are designed to detect the presence and concentration of certain gases based on their unique absorption of infrared wavelengths. These sensors are commonly used for industrial safety and environmental monitoring.

g) ***Infrared Array Sensors:*** These sensors consist of an array of individual IR sensors, which can be used to create thermal maps and detect temperature variations across a surface or within a field of view. They are used in applications like thermal imaging and industrial process control.

IR sensors are valuable in a wide range of applications due to their ability to detect temperature, motion, and proximity. They are often used in combination with other technologies, such as microcontrollers or signal processing algorithms, to make them even more versatile and applicable in various fields, including electronics, automation, and remote sensing.

6.1.2 Circuit Diagram for IR Sensor

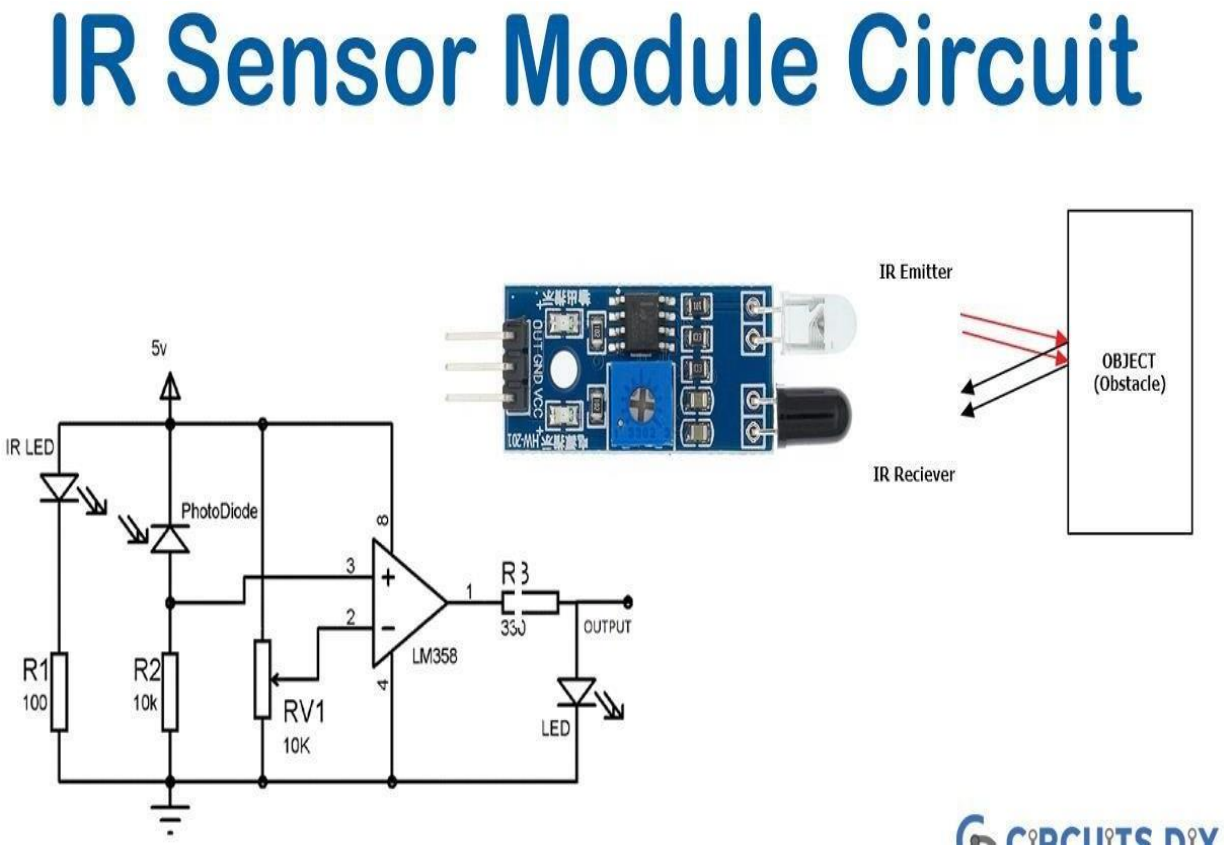


Figure 6.2: IR Sensor circuit

The IR sensor circuit diagram explains the links. The photodiode is connected to the variable resistor by all LM358 (PIN 2) inverting ends to change the sensor's sensitivity. And the photodiode and resistor linkage are joined to the non-inverting end (PIN 3).

There is no IR radiation to the photodiode when we switch ON the circuit, and the comparator output is LOW. If we take an item (not a black object) in front of an IR set, the IR emitted by an IR LED reflects the thing and ejects it. The voltage drops across the photodiode, and the voltage across the resistance R2, now increases as reflected IR Falls on Photodiode. When the voltage at R2 (connected to the comparator's non-inverting end) is greater than the voltage at the inverted end, the result is Important.

6.1.3 Applications and Uses

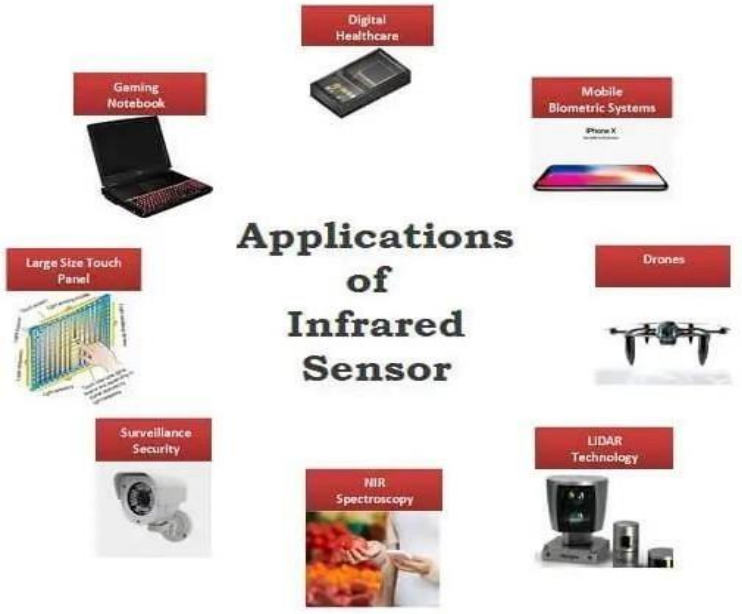


Figure 6.3: Applications and uses of IR Sensor

Detector infrasound emits or senses infrared radiation to get its setting. There was a mistake. The basic idea for an infrared sensor used to send an infrared signal is to transfer this simple pleasure signal from an image, and the signal is received in an indirect receiver.

6.2 BUZZER

A piezoelectric buzzer alarm that operates at both 5V and 12V typically means that it can handle a range of voltage inputs within that range and still produce sound. These buzzers are versatile and can be used in various projects where different power supply voltages are available.

6.2.1 Here's how to use a piezoelectric buzzer alarm with both 5V and 12V power sources

a) Identify the Buzzer Pins:

Most piezoelectric buzzers have two pins: one for the positive connection (+) and one for the ground (−). Identify these pins on your buzzer.

Figure 6.4: Buzzer

b) Check the Voltage Requirements:

Look at the datasheet or the label on the buzzer to determine its voltage requirements. For a buzzer that operates at both 5V and 12V, it should be clearly stated.

c) Choose the Voltage Source:

Select either a 5V or 12V power source, depending on your project's requirements. Ensure that the chosen voltage source matches the requirements of the buzzer.

d) Connect the Buzzer:

- If you are using a 5V power source:
  - Connect the positive lead of the buzzer to the + (positive) terminal of the 5V power source.
  - Connect the negative lead of the buzzer to the − (ground) terminal of the 5V power source.
- If you are using a 12V power source:
  - Connect the positive lead of the buzzer to the + (positive) terminal of the 12V power source.
  - Connect the negative lead of the buzzer to the − (ground) terminal of the 12V power source.

e) Apply Power:

Apply power to the buzzer by turning on the power source or connecting it to a power supply. The buzzer should produce sound when powered with the selected voltage.

f) Test and Integrate:

Test the buzzer to ensure it is functioning as expected. Depending on your project, you can integrate the buzzer as an alarm, indicator, or notification device.

g) Safety Considerations:

h) Ensure that you are using the correct voltage for the buzzer to prevent damage. Be cautious about the electrical connections, and follow safety guidelines for handling electrical components.

Remember that not all piezoelectric buzzers are the same, so it's essential to consult the datasheet or product documentation for your specific buzzer to ensure you are providing

the correct voltage and connections. These versatile buzzers can be useful in a wide range of applications, including alarms, timers, and notification systems.

6.2.2 Circuit of Buzzer

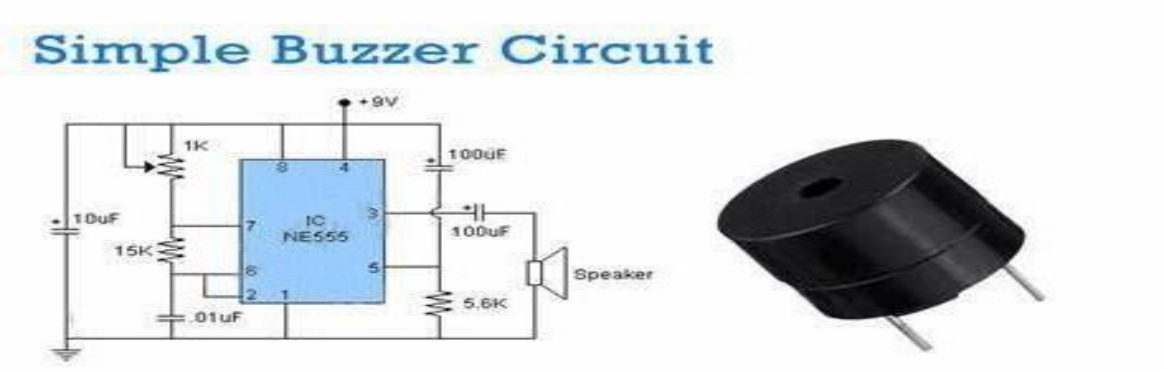


Figure 6.5: Buzzer circuit

6.3 CAR BODY

The term "car body" refers to the main structural and exterior component of an automobile that encases the vehicle's interior and houses its various components, including the engine, transmission, chassis, and passenger compartments. The car body plays a crucial role in defining the vehicle's appearance, aerodynamics, safety, and overall functionality.

Figure 6.6: Car Body

6.3.1 Here are some key aspects of a car body

a) ***Materials:*** Car bodies are typically made from various materials, with steel being the most common. Aluminium, carbon Fiber, and composite materials are also used, depending on the desired properties (e.g., weight, strength, corrosion resistance).

b) ***Components:*** The car body includes multiple components, such as the chassis, frame, doors, hood, trunk, roof, fenders, bumpers, and various panels. These components provide structural integrity and aesthetic appeal to the vehicle.

c) ***Aerodynamics:*** The design of the car body influences its aerodynamics. Engineers strive to reduce air resistance, which improves fuel efficiency and handling while minimizing wind noise.

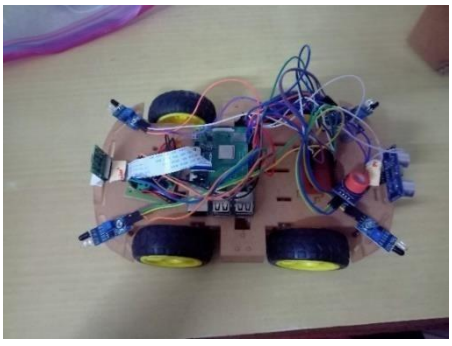


Figure 6.7: Car body connection

d) ***Crashworthiness:*** Modern car bodies are designed to provide a high level of crash safety for occupants. This involves incorporating features like crumple zones, reinforced passenger compartments, airbags, and seatbelt systems.

e) ***Design and Styling:*** The car body's design, styling, and colour play a significant role in the vehicle's appeal and marketability. Car manufacturers often invest in extensive design and styling to create unique and recognizable vehicle models.

f) ***Manufacturing:*** Car bodies are manufactured through various processes, including stamping, welding, and assembly. Mass production techniques are used to create large quantities of vehicles efficiently.

g) ***Customization:*** Car owners sometimes customize their car bodies with features like paint jobs, spoilers, body kits, and decals to achieve a unique appearance.

h) ***Maintenance:*** Maintaining the car body involves regular cleaning, waxing, and rust protection to preserve the vehicle's appearance and protect it from corrosion.

i) ***Repair and Restoration:*** In cases of accidents or wear and tear, car bodies may require repair or restoration. Body shops and auto mechanics specialize in repairing and restoring car bodies.

j) ***Environmental Impact:*** The production and disposal of car bodies have environmental implications. The auto industry is increasingly focused on sustainability and recycling to reduce the environmental impact of car production and disposal.

The car body is a fundamental element of an automobile, and its design, materials, and construction are continually evolving to meet safety, performance, and aesthetic goals while considering environmental concerns. Car bodies also reflect the manufacturer's branding and design philosophy, contributing to the distinctive appearance and identity of each car model.

6.4 JUMPER WIRES

Jumper wires are essential components in electronics and prototyping, commonly used to create electrical connections between various electronic components on a breadboard, circuit board, or between different parts of a circuit. These wires are flexible and typically come with connectors on either end that make them easy to use in building, testing, and debugging electronic circuits.

6.4.1 Here are some key aspects of jumper wires

a) ***Male-to-Male (M-M) Jumper Wires:*** These wires have male connectors on both ends and are used to connect two female connectors, such as those on a breadboard or a header pin.

b) ***Male-to-Female (M-F) Jumper Wires:*** These wires have a male connector on one end and a female connector on the other. They are often used to connect a male pin to a female pin, making them versatile for various connections.

c) ***Female-to-Female (F-F) Jumper Wires:*** These wires have female connectors on both ends and are used to extend or connect female pins together.

Figure 6.8: Jumper Wires

6.4.2 Key Uses and Applications

a) ***Breadboarding:*** Jumper wires are indispensable when prototyping on a breadboard, allowing you to create temporary connections between components to test and experiment with various circuit designs.

b) ***Circuit Building:*** When building or repairing electronic circuits, jumper wires help establish connections between components on a PCB (printed circuit board).

c) ***Debugging:*** Jumper wires are handy for troubleshooting and diagnosing problems in circuits by allowing you to make temporary connections or bypass components.

d) ***Education and Learning:*** Jumper wires are often used in educational settings to teach electronics and electrical circuits, as they provide a practical way to demonstrate and experiment with circuit concepts.

e) ***Microcontroller Projects:*** In projects involving microcontrollers like Arduino or Raspberry Pi, jumper wires are used to connect sensors, displays, and other peripheral devices to the microcontroller's GPIO pins.

f) ***Prototyping with Solderless Breadboards:*** For quick and non-permanent connections, solderless breadboards are commonly used, and jumper wires are essential for making connections between components on the breadboard.

6.4.3 Types of Connector Ends

Jumper wires may come with various connector types, such as:

- DuPont Connectors: These are commonly used for connecting to header pins on microcontrollers, sensors, and other components.
- Alligator Clips: These are used for making temporary connections to various components, wires, and test points.
- Ring Terminals: Often used for attaching jumper wires to screw terminals and other connectors.

Colour Coding: Jumper wires often come in various colours, making it easier to visually organize and identify connections in a complex circuit. Jumper wires are versatile tools for electronics enthusiasts, students, engineers, and hobbyists, as they simplify the process of building and testing electronic circuits, making them an essential part of any electronics workbench.

6.5 ALCHOL DETECTION SENSOR

The MQ-3 alcohol gas sensor module is a widely used sensor for detecting the presence of alcohol vapor in the air, particularly ethanol. This sensor can be integrated into various projects and applications, such as breathalysers, alcohol monitoring devices, and safety systems.

6.5.1 Components and Features

Figure 6.9: Alcohol detection sensor

a) ***MQ-3 Sensor:*** The core component is the MQ-3 sensor itself, which is sensitive to alcohol vapor. It contains a sensing element that changes its resistance when exposed to alcohol.

b) ***Analog-to-Digital Converter (ADC):*** The sensor module may have an onboard ADC to convert the analog resistance changes into digital values for easy interfacing with microcontrollers.

c) ***Potentiometer:*** Some modules have a potentiometer for sensitivity adjustment. You can use this to fine-tune the sensor's response.

d) ***Indicator LEDs:*** There may be indicator LEDs on the module to show the sensor's status or trigger an alert.

6.5.2 Basic Connection and Usage

Here's a simple guide on how to use an MQ-3 alcohol sensor module for ethanol concentration detection:

Components Needed:

- MQ-3 alcohol sensor module
- Microcontroller (e.g., Arduino)
- Power supply (usually 5V DC) **Steps:**

a) ***Connect Power:*** Provide power to the sensor module. Most MQ-3 modules operate at 5V, so connect the VCC and GND pins accordingly.

b) ***Connect to a Microcontroller:*** Connect the analog output (usually labelled "AO") of the sensor module to one of the analog input pins of your microcontroller (e.g., Arduino).

c) ***Adjust Sensitivity:*** If your module has a potentiometer, adjust it to set the sensor's sensitivity. You can use a screwdriver to turn the potentiometer and observe the sensor's response on the microcontroller.

d) ***Read Sensor Data:*** Write a simple code on your microcontroller to read the analog voltage output from the sensor (analogRead in Arduino). The analog value corresponds to the sensor's resistance, which changes with the concentration of alcohol vapor in the air.

e) ***Calibration:*** The sensor's response can vary, and you may need to calibrate it to obtain accurate ethanol concentration readings. To do this, expose the sensor to known alcohol concentrations and note the analog readings. Use this data to create a calibration curve for your specific sensor.

f) ***Ethanol Concentration Calculation:*** Using the calibration curve, convert the analog reading into an estimated ethanol concentration.

g) ***Threshold Detection***: You can set a threshold value to trigger an alert or action when the ethanol concentration surpasses a certain level. For example, if the concentration exceeds a predefined limit, you can activate an alarm or notification.

h) ***Data Display:*** You can display the alcohol concentration on an LCD or send it to a computer or other monitoring devices for further analysis.

Keep in mind that the MQ-3 sensor module may have limitations, including sensitivity to various alcohol compounds and environmental factors. Calibration and proper environmental conditions are crucial for accurate and reliable ethanol concentration measurements. Additionally, the sensor may require warm-up time before it provides stable readings.

6.5.3 Circuit of Alcohol sensor

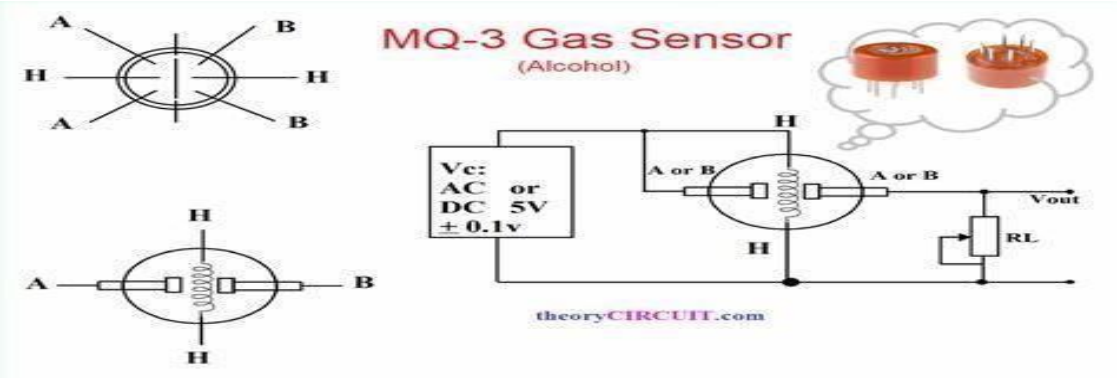


Figure 6.10: Alcohol detection sensor circuit

6.6 ULTRASONIC SENSOR

Figure 6.11: Ultrasonic sensor

Ultrasonic sensors are devices that use sound waves of very high frequency, beyond the range of human hearing, to measure distances and detect objects. They are commonly used in various applications, including robotics, industrial automation, automotive safety, and more.

6.6.1 Here are the key aspects of ultrasonic sensors

Working Principle: Ultrasonic sensors operate on the principle of sending out ultrasonic (high-frequency) sound waves and measuring the time it takes for the sound waves to bounce off an object and return to the sensor. The time-of-flight measurement is then used to calculate the distance to the object.

The process involves the following steps:

- The ultrasonic sensor emits a burst of ultrasonic waves, typically in the range of 20 kHz to 200 kHz.
- The sound waves propagate through the air until they encounter an object in their path.
- Upon hitting the object's surface, the sound waves bounce back towards the sensor.
- The sensor measures the time it takes for the sound waves to return, and using the speed of sound in the air, it calculates the distance to the object.

6.6.2 Key Features and Components

a) ***Transducer:*** The transducer is the core component of the ultrasonic sensor. It emits and receives ultrasonic waves. In most cases, it includes a single transducer for both emitting and receiving.

b) ***Housing:*** The sensor is typically enclosed in a housing that protects the transducer and electronic components. The design of the housing can affect the sensor's field of view and performance.

c) ***Control Electronics:*** These electronics control the emission and reception of ultrasonic waves and perform the necessary calculations to determine the distance to the object.

d) ***Output Interface:*** Ultrasonic sensors typically provide output in the form of analog voltage, digital pulse width, or digital serial data. This output can be processed by microcontrollers or other electronic devices.

6.6.3 Applications

a) ***Distance Measurement:*** They are commonly used for non-contact distance measurement in robotics, industrial automation, and level sensing.

b) ***Object Detection:*** Ultrasonic sensors can detect the presence or absence of objects in various applications, such as parking assistance in vehicles.

c) ***Obstacle Avoidance:*** They are used in robots and drones to avoid collisions with obstacles by providing distance information.

d) ***Liquid Level Measurement:*** Ultrasonic sensors can be used to measure the level of liquid in tanks and containers.

e) ***Proximity Sensing:*** Ultrasonic proximity sensors can detect the presence of people or objects in certain areas, making them useful for security and access control.

6.6.4 Circuit of Ultrasonic Sensor

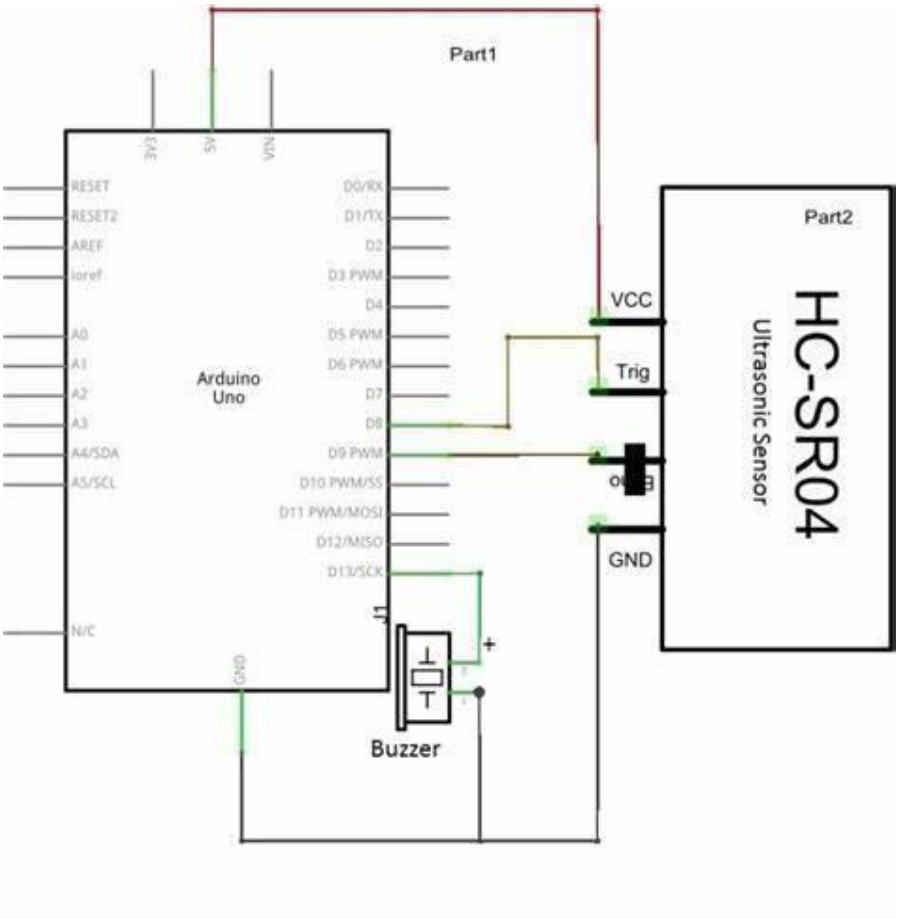


Figure 6.12: Ultrasonic sensor circuit

6.7 SOLDERING BOARD

A "soldering board" typically refers to a piece of equipment or material used in soldering and electronics work to provide a stable and heat-resistant surface. These boards come in various forms and serve different purposes in soldering and electronics projects.

6.7.1 Here are some common types of soldering boards and their uses

a) ***Soldering Mat or Pad:*** A soldering mat or pad is typically made of silicone or other heat-resistant materials. It provides a non-slip surface for soldering and protects your work surface from heat and solder splatter. It's especially useful when soldering small components and prevents damage to your table or workbench.

b) ***Soldering Station Base:*** In some soldering stations, the base unit contains a built-in soldering board or pad. This area is designed for soldering work and may have a heat-resistant, flame-retardant surface. It often includes storage compartments for soldering iron tips, solder, and other accessories.

c) ***Soldering Tile or Block:*** Soldering tiles or blocks are made of materials like ceramic or firebrick, which can withstand high temperatures. They are used to support the workpiece during soldering and help distribute heat evenly. Ceramic soldering tiles are often used for jewellery and plumbing work.

d) ***Breadboard:*** A soldering breadboard is a circuit board with pre-drilled holes and copper pads or traces that allow for the creation of soldered electronic circuits. It's often used for prototyping and testing electronic circuits. Components are inserted into the holes, and solder is used to make connections.

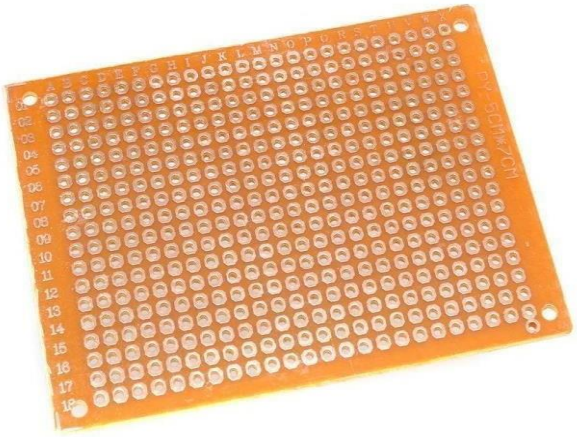


Figure 6.13: Soldering board

e) ***Printed Circuit Board (PCB):*** Soldering is a common process used to assemble electronic components onto printed circuit boards (PCBs). While PCBs are not traditionally referred to as soldering boards, they serve the function of providing a surface for soldering electronic components in a permanent and structured manner.

f) ***Desoldering Board:*** A desoldering board is used for removing soldered components. It typically includes a pattern of holes and channels to help wick away molten solder. This is especially useful for desoldering and salvaging electronic components.

g) ***Soldering Jig or Fixture:*** In some cases, a soldering board can be a specialized jig or fixture that holds components in place during soldering. These fixtures are commonly used in production and manufacturing environments to ensure precise soldering of components.

h) ***Soldering Iron Stand:*** While not a board in the traditional sense, a soldering iron stand often includes a base with a holder for the soldering iron and a sponge or brass wool pad for cleaning the iron's tip. It is an essential accessory for soldering work, ensuring safety and convenience.

6.8 CAR MOTORS

A "Motor" typically refers to an electric motor that rotates at a speed of 100 revolutions per minute. These motors are commonly used in various applications where a specific rotational speed is required. The actual use and specifications of such a motor can vary widely based on the type of motor, its size, power source, and intended application.

6.8.1 Some common types of motors that can operate at 100 rpm

a) ***DC Motors:*** Direct current (DC) motors are available in various sizes and configurations. They can be controlled to achieve specific speeds and are often used in robotics, small appliances, and industrial automation.

b) ***Gearmotors:*** Gearmotors are a combination of a motor and a gearbox. The gearbox reduces the motor's high-speed output to a lower, more usable speed, such as 100 rpm. These are used in a wide range of applications, including conveyors, conveyor belts, and other machinery.

Figure 6.14: Car motor

c) ***Stepper Motors:*** Stepper motors move in discrete steps, making them suitable for applications where precise control of position and speed is required. They are often used in CNC machines, 3D printers, and various motion control systems.

d) ***AC Synchronous Motors:*** These motors are designed to rotate at a constant, synchronous speed when connected to an AC power source. They are commonly used in appliances like microwave turntables and clock mechanisms.

When selecting a 100-rpm motor for your specific application, it's essential to consider factors such as motor type, voltage requirements, torque, and physical size to ensure it meets your needs.

6.8.2 Uses of Motors

A 100 RPM (revolutions per minute) DC motor has various applications across different industries due to its moderate speed and versatility.

Here are some common uses of a 100 RPM DC motor:

a) ***Conveyors:*** 100 RPM DC motors can be used in conveyor systems to transport materials in manufacturing, warehousing, and packaging industries. They provide controlled and continuous motion for moving items along a production line.

b) ***Automated Blinds and Curtains:*** These motors are used in home automation systems to open and close blinds, curtains, or drapes. The slow speed allows for smooth and precise control.

c) ***Rotating Displays:*** In trade shows or retail environments, 100 RPM motors can be used to rotate product displays or promotional signs, creating eyecatching and interactive displays.

d) ***Turntables:*** Turntables, such as those used in display cases, photography studios, or cake decorating, can benefit from a 100 RPM motor to achieve a consistent and slow rotation.

e) ***Low-Speed Robotics:*** In some robotic applications, slower motor speeds are preferred for precision, accuracy, and control. 100 RPM motors can be used in robot arms, grippers, or mobility systems.

f) ***Stirrers and Mixers:*** In laboratories and industrial settings, these motors can be used to drive stirrers and mixers for liquids or other substances, ensuring a uniform mixture.

g) ***Pottery Wheels:*** For artists and potters, 100 RPM motors can be used to power pottery wheels, allowing for precise control while shaping clay.

h) ***Exhibit Models:*** In museums and educational displays, 100 RPM motors can be used to create moving models and dioramas to demonstrate various processes or historical events.

i) ***Model Trains and Railways***: Hobbyists often use 100 RPM DC motors in model trains and railway layouts to control the speed of locomotives and rolling stock.

j) ***Automated Chicken Coops:*** For small-scale poultry farming, these motors can be used to open and close chicken coop doors at specific times, providing security for the birds.

k) ***Agricultural Equipment:*** In farming applications, 100 RPM motors can be employed in equipment like seeders, grain augers, and small-scale irrigation systems.

l) ***Homebrewing:*** Some homebrewers use 100 RPM motors in their brewing setups to automate stirring or milling processes during beer or wine production.

6.9 L293D DRIVER

The L293D is a popular integrated circuit (IC) used as a motor driver or motor controller. It is specifically designed to drive small to medium-sized direct current (DC) motors and is widely used in various robotics and electronics projects. The L293D IC provides the necessary control and power amplification to drive these motors bidirectionally, allowing them to rotate in both directions (forward and reverse).

6.9.1 Here are some key features and uses of the L293D motor driver:

Key Features:

a) ***Bidirectional Control:*** The L293D can control the direction of rotation of a DC motor by allowing current to flow in either direction through the motor windings.

b) ***Dual H-Bridge Configuration:*** The IC is often used to control two motors independently. It contains two H-bridge circuits, each capable of controlling one motor.

c) ***Built-In Clamping Diodes:*** The L293D includes built-in clamping diodes to protect against back electromotive force (EMF) generated by the motor. This helps to prevent damage to the IC.

d) ***High Current Handling:*** The L293D can handle relatively high currents, making it suitable for driving a range of small to medium-sized motors.

Figure 6.15: L293D Driver

e) ***TTL and CMOS Compatibility:*** It is compatible with both TTL (Transistor-Transistor Logic) and CMOS (Complementary Metal-Oxide-Semiconductor) control signals.

Common Uses:

a) ***Robotics:*** The L293D is commonly used in robotics projects to control the movement of wheels or other actuators. It allows for precise control of motor speed and direction, making it a fundamental component in robot design.

b) ***RC (Remote Control) Vehicles:*** L293D can be used in remote-controlled cars, boats, and airplanes to control the motors responsible for propulsion and steering.

c) ***Automated Systems:*** It can be used in various automated systems, such as conveyor belts, automatic door openers, and machinery with motor-driven components.

d) ***Electronics Projects:*** Hobbyists and electronics enthusiasts use the L293D in a wide range of DIY projects, including home automation, smart devices, and model trains.

e) ***Low-Power Motor Control:*** The L293D is suitable for low-power applications where precision motor control is needed, such as small appliances, fans, and automatic curtains.

f) ***Education:*** It is often used in educational settings to teach students about motor control and electronics.

When using the L293D, it's important to consult the IC's datasheet and wiring diagrams to ensure correct connections and to avoid damaging the IC or connected motors. Proper voltage and current ratings for the motors should also be considered to avoid overheating and failure. Additionally, note that the L293D is a bipolar IC, and more modern alternatives, such as the L298N or motor driver modules based on MOSFETs, are available for high-power applications and have some advantages in terms of efficiency and heat dissipation.

6.9.2 Circuit of L293D Driver

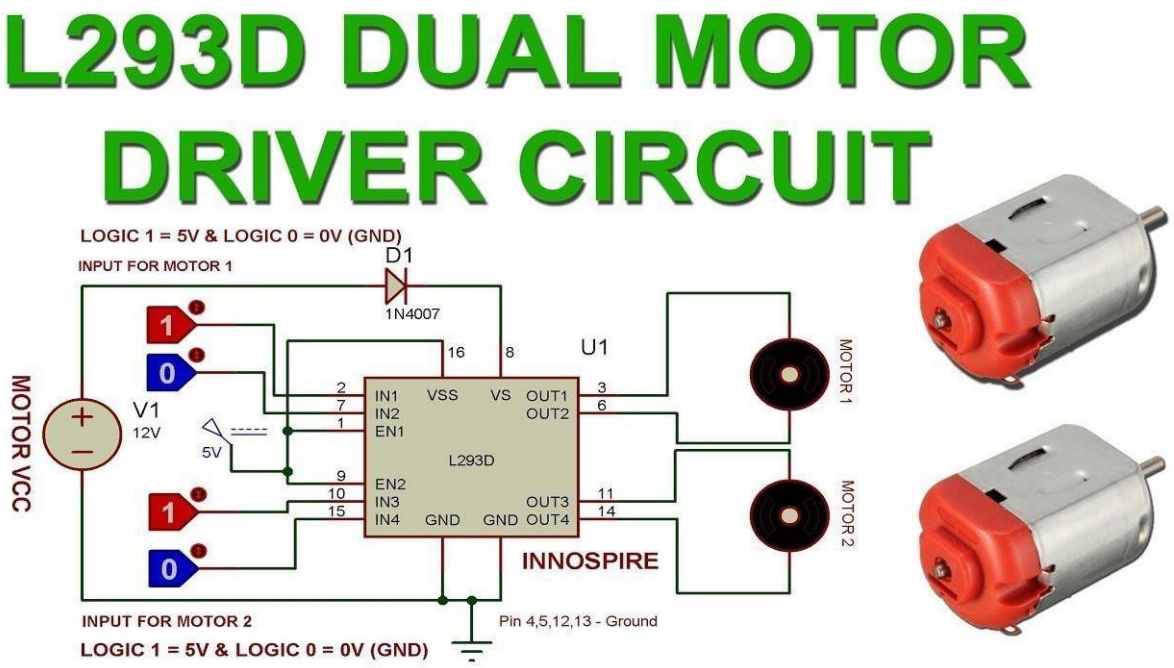


Figure 6.16: L293D Driver circuit

6.10 RASPBERRY PI 5MP CAMERA

Using a Raspberry Pi 5MP Camera Module with the accompanying cable is a common and straightforward process. The camera module is designed to be easily interfaced with a Raspberry Pi, making it suitable for various projects, such as video streaming, surveillance, and image capture.

6.10.1 Here are the steps to use the Raspberry Pi 5MP Camera Module with its cable Requirements

- Raspberry Pi (any model with the camera connector)
- Raspberry Pi Camera Module (5MP or higher resolution

- Ribbon cable (included with the camera module)
- Access to the Raspberry Pi's GPIO pins
- Access to a screen or remote access to the Raspberry Pi (e.g., SSH or VNC).

Figure 6.17: Raspberry pi camera rev1.3

6.10.2 Raspberry pi camera configuration Steps

a) ***Prepare Your Raspberry Pi:***
Ensure that your Raspberry Pi is powered off and disconnected from the power source.

b) Enable the Camera Interface:
On your Raspberry Pi, you need to enable the camera interface in the Raspberry Pi Configuration settings:

- Open a terminal on your Raspberry Pi or access it remotely.
- Run `sudo raspi-config`.
- Navigate to "Interfacing Options."
- Select "Camera" and enable it.
- Reboot your Raspberry Pi.

c) Connect the Camera Module:

- Locate the camera connector on the Raspberry Pi board. It's a small, flat, rectangular connector typically found near the HDMI port.
- Carefully unlock the connector by pulling back the plastic tabs on either side.
- Insert the ribbon cable's flat end into the camera connector, ensuring the contacts are facing down
- Secure the cable by gently pressing the plastic tabs back down.

d) Attach the Camera Module:
Carefully attach the camera module to the other end of the ribbon cable. Align the camera's connector with the cable and gently press them together. Make sure it's secure.

e) Power On the Raspberry Pi:
Power on your Raspberry Pi by connecting it to a power source.

f) Capture Images or Video:
You can use the terminal or Python code to capture images or record video using the camera module.

g) View or Transfer the Captured Media:

You can view the captured media on your Raspberry Pi's screen or transfer it to another device for further use.

This basic setup allows you to use the Raspberry Pi Camera Module with its cable. Depending on your project, you can explore various camera settings and functionalities, including adjusting camera parameters, streaming video, or integrating it into specific applications. The Raspberry Pi's official website and documentation offer.

6.10.3 Circuit of Raspberry pi camera module rev1.3

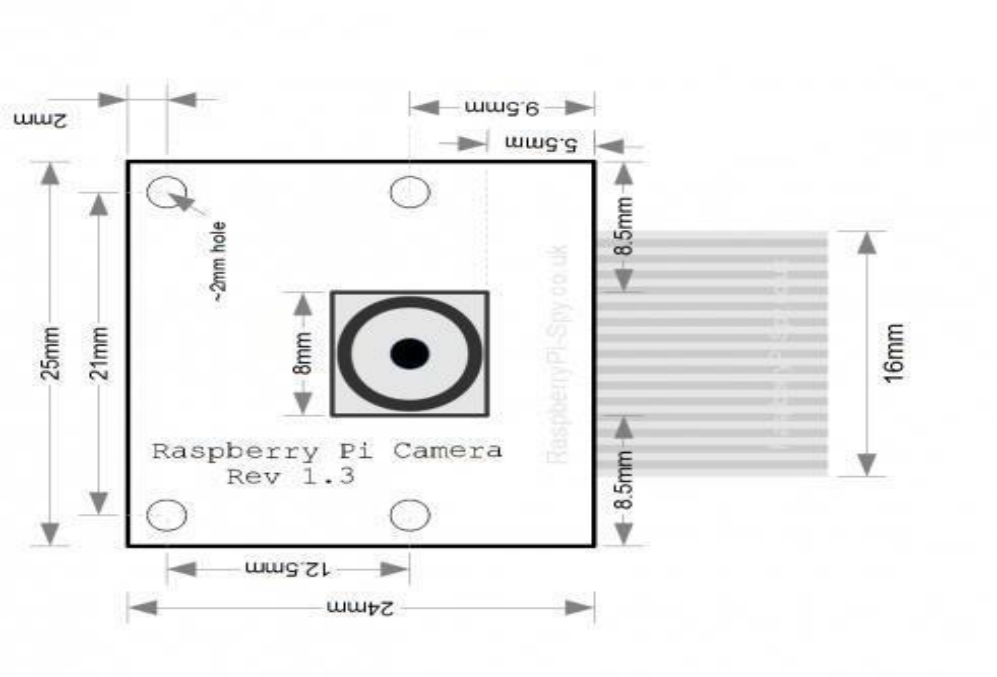


Figure 6.18: raspberry pi camera rev1.3 Circuit of

- **Camera Internal Connections Pins**

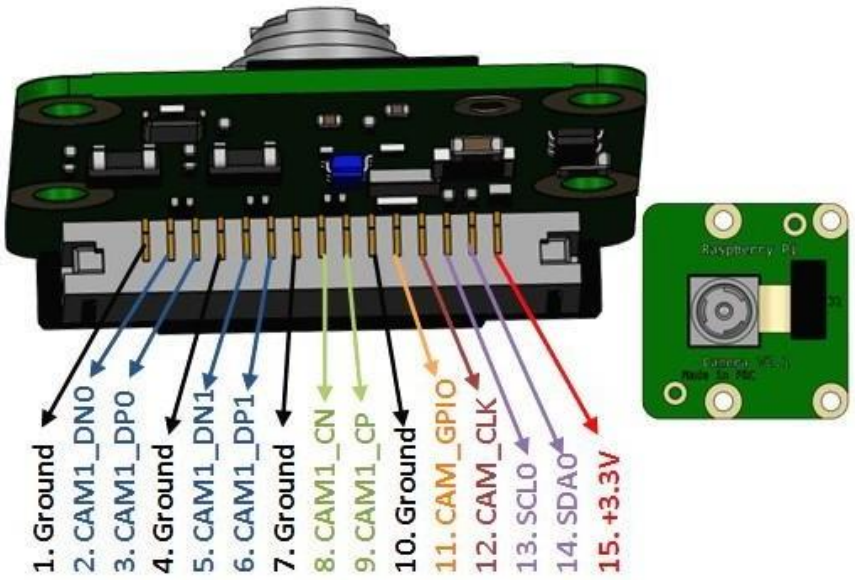


Figure 6.19: Internal connection of raspberry pi camera rev1.3

6.10.4 Applications and uses of Raspberry pi camera

a) ***Video and Image Capture***: The most basic use is capturing photos and recording video. This can be used for personal photography, time-lapse photography, or video recording.

b) ***Security and Surveillance:*** Raspberry Pi cameras are often used in DIY home security and surveillance systems. They can be set up to monitor an area, record footage, and even send alerts in case of motion detection.

c) ***Remote Monitoring:*** You can use the camera to remotely monitor places or situations, such as a remote cabin, wildlife, or construction sites. You can access the camera feed from anywhere with an internet connection.

d) ***Video Conferencing:*** The camera can be used for video conferencing, turning your Raspberry Pi into a low-cost video chat or teleconferencing solution.

e) ***Home Automation:*** Integrate the camera into home automation projects. For example, you can set it up to take pictures when a doorbell rings or when a motion sensor is triggered.

f) ***Time-Lapse Photography:*** The camera is excellent for capturing time-lapse sequences, which can be used for documenting changes in nature, construction projects, and other long-term events.

g) ***Wildlife Monitoring:*** Raspberry Pi cameras are used in wildlife monitoring projects to observe animals in their natural habitat without human presence, helping with research and conservation efforts.

h) ***3D Printing:*** Some 3D printers use Raspberry Pi cameras to monitor the printing process and capture time-lapse videos of the printing.

i) ***DIY Drones and Robotics:*** Raspberry Pi cameras are used in drones and robotics projects to provide live video feeds for remote control and navigation.

j) ***Educational Projects:*** The camera module is an excellent tool for educational purposes, helping students learn about photography, computer vision, and image processing.

k) ***Healthcare:*** Raspberry Pi cameras have been used in healthcare applications such as telemedicine and monitoring patient conditions remotely.

l) ***Barcode Scanning:*** The camera can be used for barcode or QR code scanning, enabling inventory management and product identification.

m) ***OCR (Optical Character Recognition):*** It can be used to capture text from printed or handwritten documents, which is useful in document digitization and automation.

n) ***Object Detection and Recognition:*** Raspberry Pi cameras can be used for object detection, face recognition, and other computer vision tasks in various applications.

o) ***Scientific Research:*** Researchers use Raspberry Pi cameras for various scientific experiments and observations, such as studying plant growth, tracking animal behaviour, and more.

p) ***Art and Creative Projects:*** Artists and makers use the camera for interactive art installations, visual arts, and creative photography projects.

q) ***Weather Station:*** Raspberry Pi cameras can be integrated into DIY weather stations to capture images of the sky and weather conditions.

r) ***Astronomy:*** Enthusiasts use the camera for astrophotography, capturing images of the night sky, stars, and celestial objects.

Chapter-7 Physical Components and Program Logic Integration

7.1 PROJECT DEVELOPMENT PROCESS

Total Hardware before Connection

Figure 7.1: Total Hardware without connection

Step 1 – Body Making

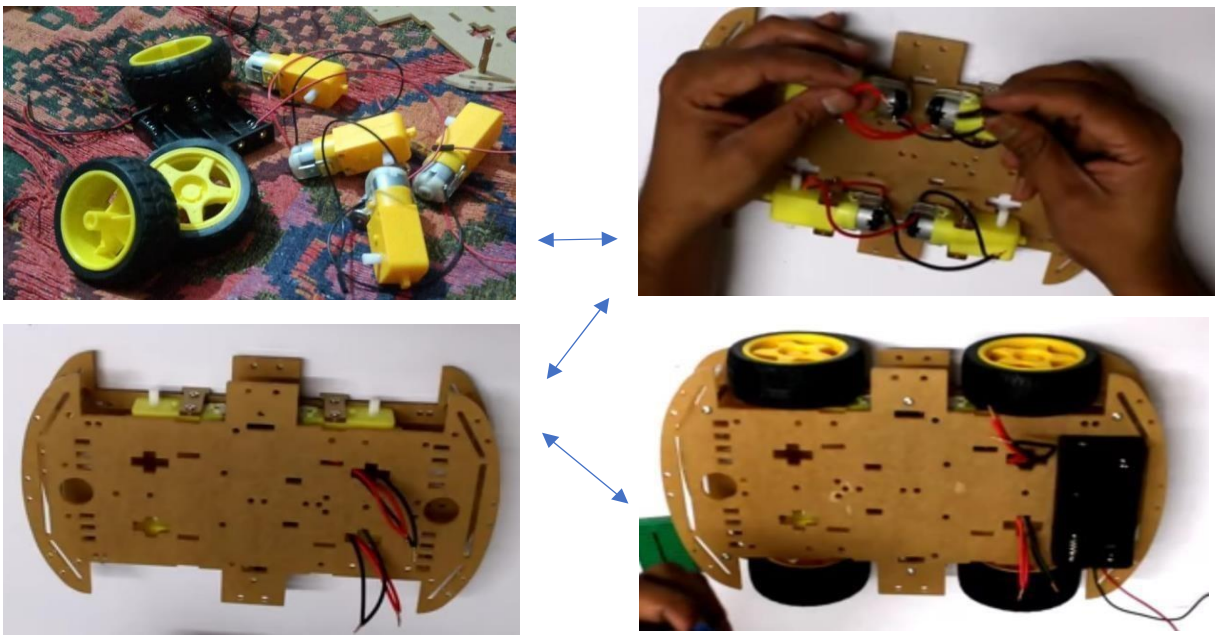
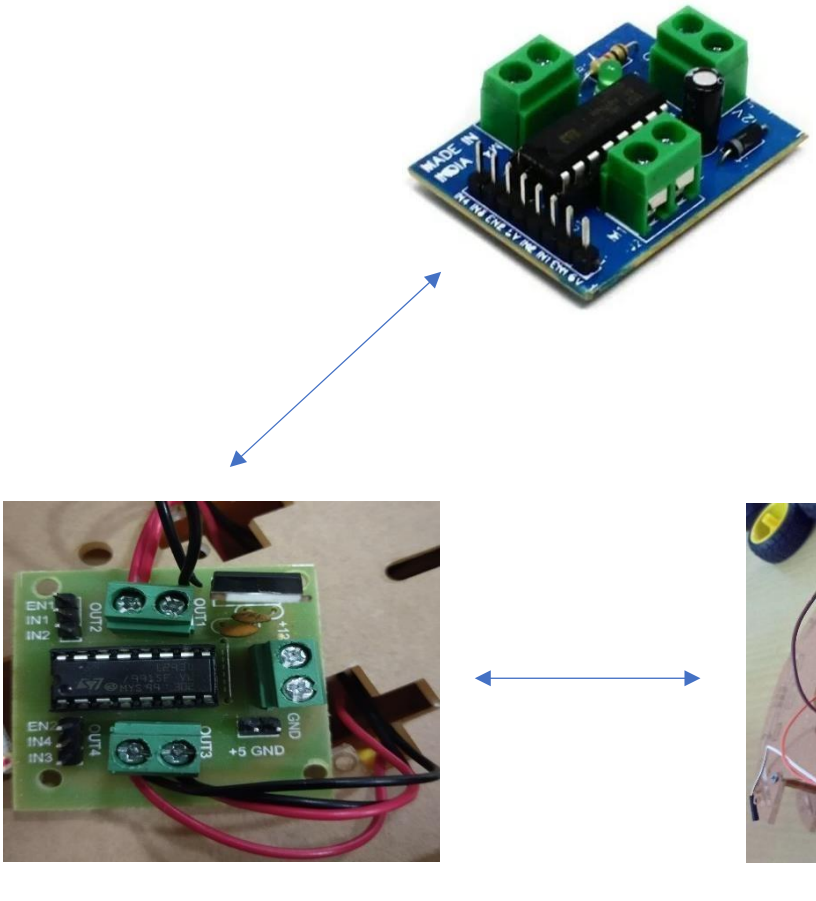


Figure 7.2: Body making with motors connection

Step 2 – Adding L293D Driver



Pin Configuration

VCC
GND
Motor 1 – 21
Motor 2 – 20
Motor 3 – 16
Motor 4 - 12

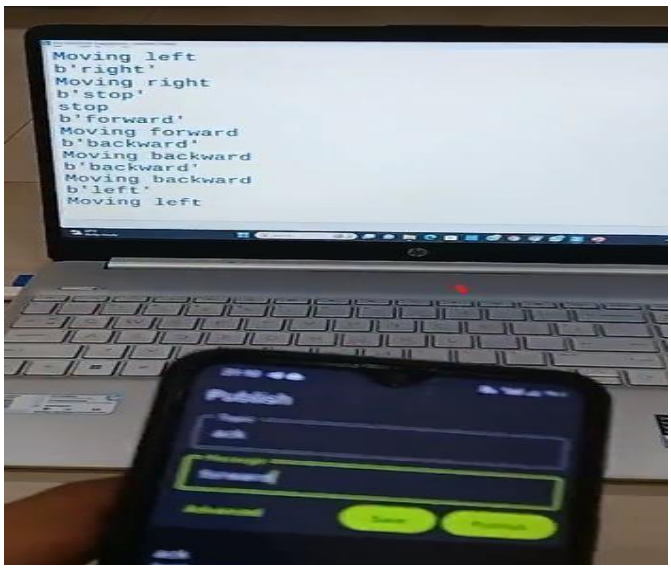
Figure 7.3: L293D Driver connection with motors

Output until this step Code

```
import RPi.GPIO as GPIO
import time

GPIO.setwarnings(False)

GPIO.setmode(GPIO.BCM)

GPIO.setup(21,GPIO.OUT)
GPIO.setup(20,GPIO.OUT)
GPIO.setup(16,GPIO.OUT)
GPIO.setup(12,GPIO.OUT)


def forward():
    GPIO.output(21,True)
    GPIO.output(20,False)
    GPIO.output(16,True)
    GPIO.output(12,False)
def back():
    GPIO.output(20,True)
    GPIO.output(21,False)
    GPIO.output(12,True)
    GPIO.output(16,False)
def right():
    GPIO.output(21,True)
    GPIO.output(20,False)
    GPIO.output(16,False)
    GPIO.output(12,True)
def left():
    GPIO.output(21,False)
    GPIO.output(20,True)
    GPIO.output(16,True)
    GPIO.output(12,False)
def stop():
    GPIO.output(21,False)
    GPIO.output(20,False)
    GPIO.output(16,False)
    GPIO.output(12,False)
```

```
import robot
import time
import paho.mqtt.client as mqtt


def on_connect(client, userdata, flags, rc):
   print("Connected with result code " + str(rc))

   client.subscribe("ack")

def on_message(client, userdata, msg):
   h = str( msg.payload)
   print(h)
   if(msg.payload.decode('utf-8') == "forward"):
      print("Moving forward")
      robot.forward()
   elif(msg.payload.decode('utf-8') == "backward"):
       print("Moving backward")
       robot.back()
   elif(msg.payload.decode('utf-8') == "right"):
       print("Moving right")
       robot.right()
   elif(msg.payload.decode('utf-8') == "left"):
       print("Moving left")
       robot.left()
   elif(msg.payload.decode('utf-8') == "stop"):
       print("stop")
       robot.stop()

client = mqtt.Client()
client.on_connect = on_connect
client.on_message = on_message
client.connect('broker.mqtt-dashboard.com', 1883, 60)
client.loop_start()
```

Output

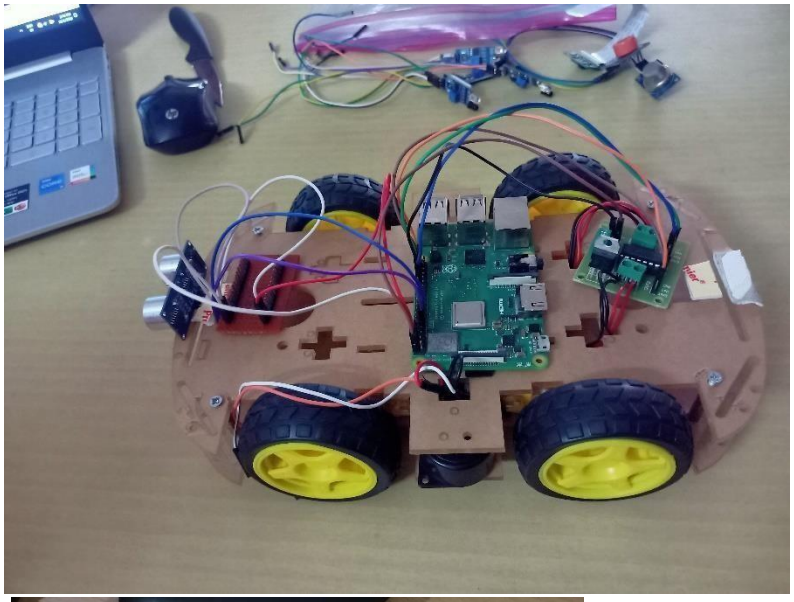


Adding a Library to Run the Above code

pip install paho-mqtt

Figure 7.4: Output for Car Movement

Step 3 – Adding Ultrasonic Sensor



Pin Configuration
VCC – Red
GND – Black
Trigger – Purple
Echo – Blue

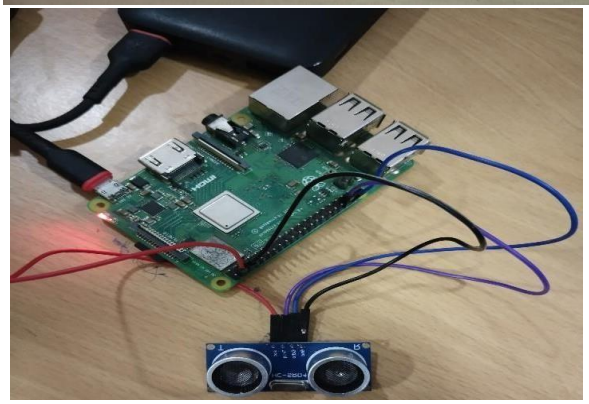


Figure 7. 5: Adding ultrasonic sensor to the car

Code

```
import RPi.GPIO as GPIO
import time
# Ultrasonic Sensor Configuration
sensor_ultrasonic = 17
echo_ultrasonic = 27
GPIO.setmode(GPIO.BCM)
GPIO.setwarnings(False)
# Ultrasonic Sensor Setup
GPIO.setup(sensor_ultrasonic, GPIO.OUT)
GPIO.setup(echo_ultrasonic, GPIO.IN)
GPIO.output(sensor_ultrasonic, False)
print("Ultrasonic Sensor Ready...")
print()
def distance_ultrasonic():
    GPIO.output(sensor_ultrasonic, True)
    time.sleep(0.00001)
    GPIO.output(sensor_ultrasonic, False)
    StartTime = time.time()
    StopTime = time.time()
    while GPIO.input(echo_ultrasonic) == 0:
        StartTime = time.time()
    while GPIO.input(echo_ultrasonic) == 1:
        StopTime = time.time()
    TimeElapsed = StopTime - StartTime
    distance = (TimeElapsed * 34300) / 2
    return distance
if __name__ == '__main__':
    try:
        while True:
            # Ultrasonic Sensor
            dist_ultrasonic = distance_ultrasonic()
            print("Ultrasonic Sensor: Object is at a distance of %.1f cm" % dist_ultrasonic)
            if dist_ultrasonic < 20:
                print("Ultrasonic Sensor: Object detected!")

            time.sleep(1)

    except KeyboardInterrupt:
        print("Measurement stopped by User")
        GPIO.cleanup()
```

Output

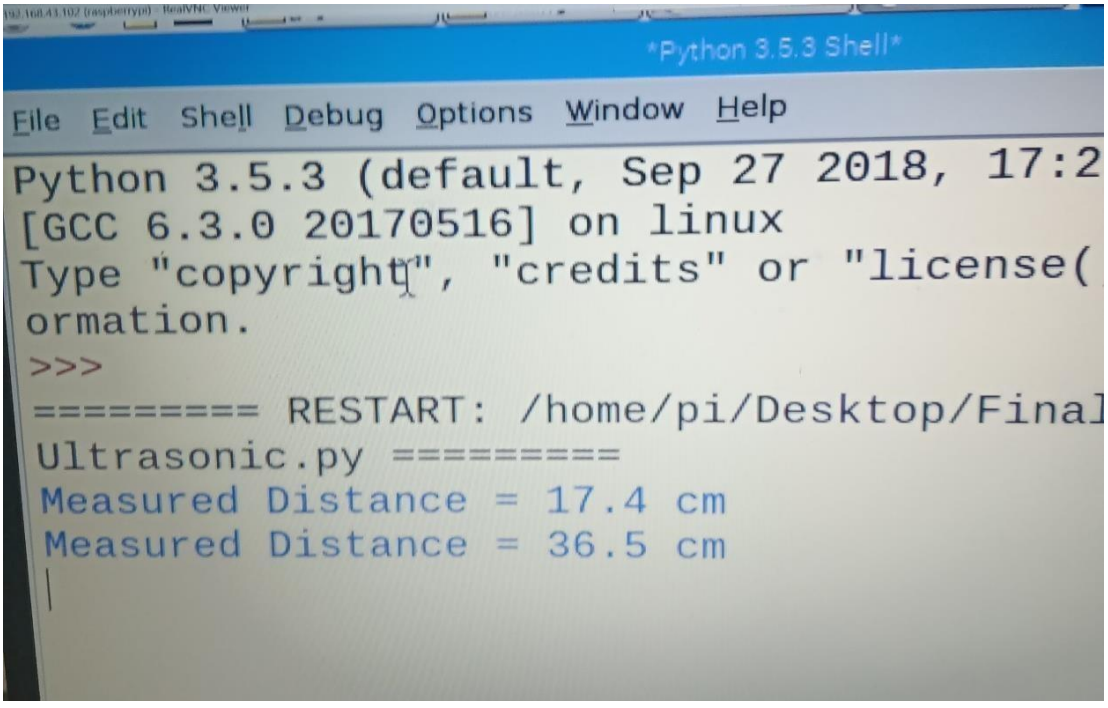
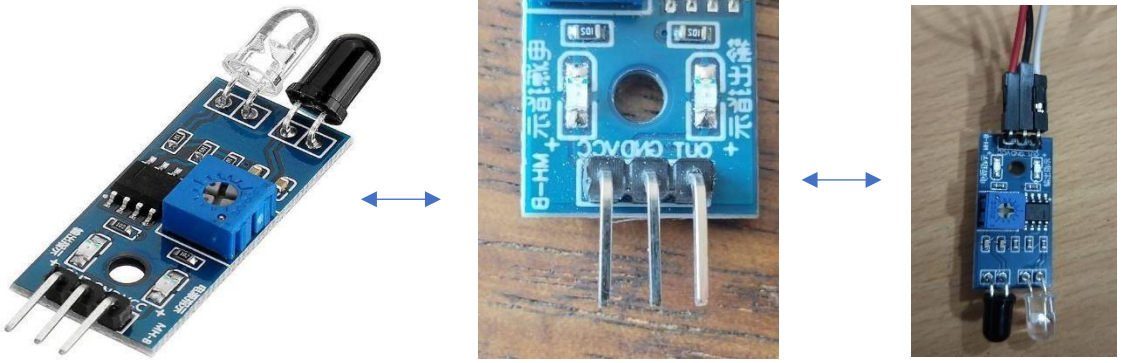
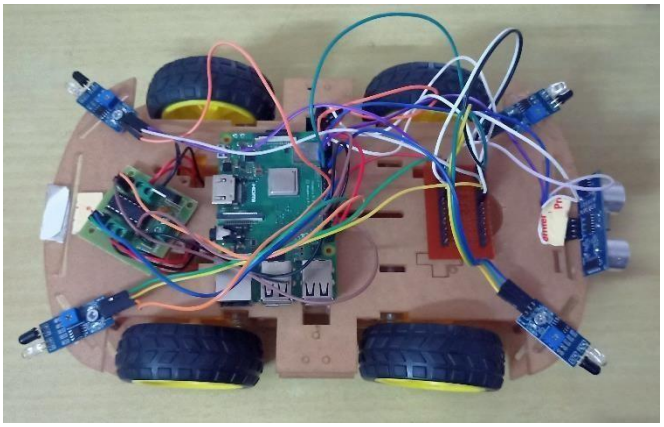


Figure 7.6: Output for ultrasonic sensor

Step 4 - Adding Infrared Sensor





Pin Configuration
VCC – Red
GND – Black
OUT – GPIO Pin - white

Figure 7.7: Adding IR sensor to the car

Code

```
GPIO.setmode(GPIO.BCM)
GPIO.setwarnings(False)

# IR Sensor Setup
GPIO.setup(sensor_ir, GPIO.IN)

print("IR Sensor Ready...")
print()

def distance_ir():
    if GPIO.input(sensor_ir):
        return 1
    return 0

def activate_buzzer_ir():
    # Placeholder function, you can add the buzzer code here if needed
    pass

def deactivate_buzzer_ir():
    # Placeholder function, you can add the buzzer code here if needed
    pass

if __name__ == '__main__':
    try:
        while True:
            # IR Sensor
            ir_status = distance_ir()
            if ir_status:
                activate_buzzer_ir()
            else:
                deactivate_buzzer_ir()
                print("IR Sensor: Object Detected")

            time.sleep(1)

    except KeyboardInterrupt:
        print("Measurement stopped by User")
        GPIO.cleanup()
```

Output

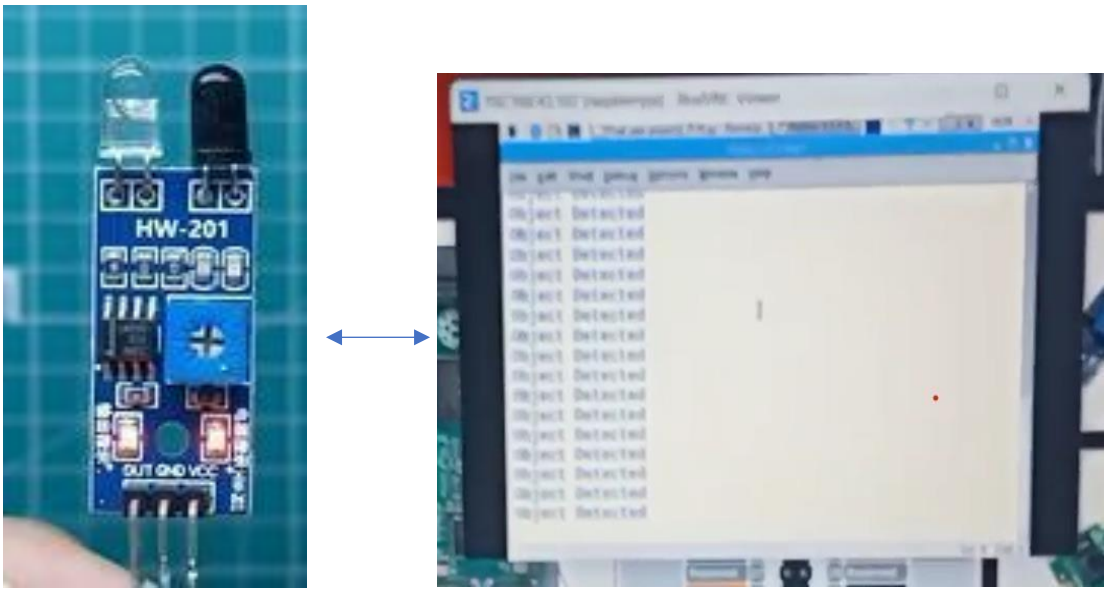
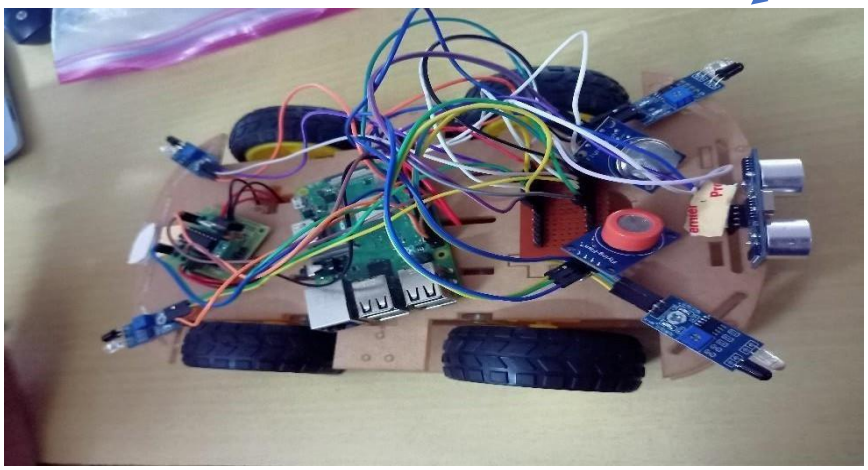


Figure 7.8: Output for IR sensor

Step 5 – Alcohol Detection Sensor



Pin Configuration
VCC – Red
GND – Black
DOUT – GPIO Pin - white

Figure 7.9: Adding Alcohol detection sensor to the car

Code

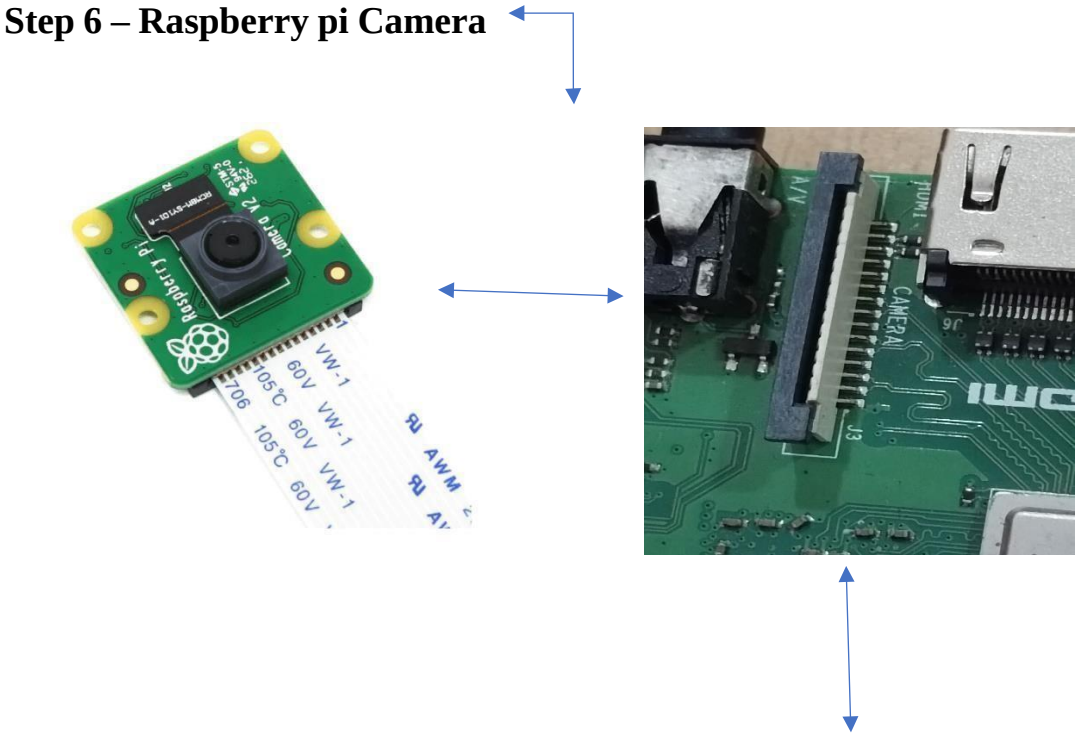
```
import RPi.GPIO as GPIO
import time
import smtplib

GPIO.setwarnings(False)
GPIO.setmode(GPIO.BCM)

# Define GPIO pins
sensor_pin1 = 2
sensor_pin2 = 3
buzzer = 4

# Email configuration
smtp_server = 'smtp.gmail.com'
smtp_port = 587
email_address = 'akhildasari5159@gmail.com'
email_password = 'kvuv qgjk tcpf hbch'
recipient_email = 'akhildasari369@gmail.com'

def send_email(message):
    subject = 'Alcohol Sensor Alert'
    msg = 'Subject: Alcohol Sensor Alert\n\n' + message

    try:
        server = smtplib.SMTP(smtp_server, smtp_port)
        server.starttls()
        server.login(email_address, email_password)
        server.sendmail(email_address, recipient_email, msg)
        server.quit()
        print("Email sent successfully.")
    except Exception as e:
        print(f"Failed to send email: {str(e)}")

def stop_car():
    # Function to stop the car - you can leave it empty if not controlling a car.
    pass

try:
    while True:
        adcValue1 = 0
        adcValue2 = 0
        for i in range(10):
            adcValue1 += GPIO.input(sensor_pin1)
            adcValue2 += GPIO.input(sensor_pin2)
            time.sleep(0.01)

        val1 = adcValue1 / 10
        val2 = adcValue2 / 10

        mgL1 = 0.67 * val1
        mgL2 = 0.67 * val2

        print(f"Sensor 1 Value: {val1}, Alcohol Level (mg/L): {mgL1}")
        print(f"Sensor 2 Value: {val2}, Alcohol Level (mg/L): {mgL2}")

        if mgL2 > 0.9:
            send_email("High Alcohol Level Detected. Car Stopped.")
            GPIO.output(buzzer, GPIO.HIGH)
            stop_car()
            time.sleep(0.3)
        elif mgL1 > 0.8:
            send_email("Alcohol Detected. Be Safe while Driving.")
            GPIO.output(buzzer, GPIO.HIGH)
            time.sleep(0.3)
        elif mgL1 < 0.5:
            send_email("No Alcohol Detected. Normal.")
            print("No Alcohol Detected. Normal.")
        else:
            print("Have a safe and happy driving")

        GPIO.output(buzzer, GPIO.LOW)
        time.sleep(1)

except KeyboardInterrupt:
    GPIO.cleanup()
```

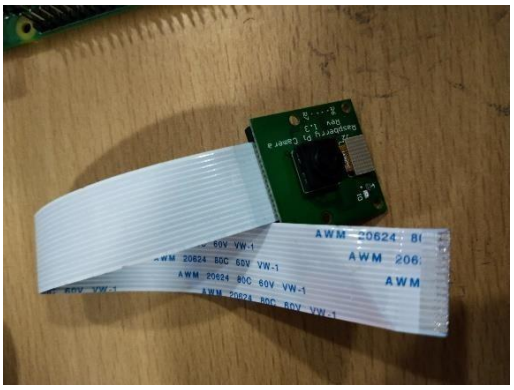
Output

Adding a Library to Run the Above code **pip install**

secure-smtplib

Step 6 – Raspberry pi Camera





Pin Configuration Connected with Ribbon cable from Camera to Raspberry pi to object Detection.

Figure 7.10: Adding Raspberry pi rev1.3 camera to the car

Output for this step

Code

```
import cv2

#thres = 0.45 # Threshold to detect object

classNames = []
classFile = "/home/pi/Desktop/Object_Detection_Files/coco.names"
with open(classFile,"rt") as f:
    classNames = f.read().rstrip("\n").split("\n")

configPath = "/home/pi/Desktop/Object_Detection_Files/ssd_mobilenet_v3_large_coco_2020_01_14.pbtxt"
weightsPath = "/home/pi/Desktop/Object_Detection_Files/frozen_inference_graph.pb"

net = cv2.dnn_DetectionModel(weightsPath,configPath)
net.setInputSize(320,320)
net.setInputScale(1.0/ 127.5)
net.setInputMean((127.5, 127.5, 127.5))
net.setInputSwapRB(True)


def getObjects(img, thres, nms, draw=True, objects=[]):
    classIds, confs, bbox = net.detect(img,confThreshold=thres,nmsThreshold=nms)
    #print(classIds,bbox)
    if len(objects) == 0: objects = classNames
    objectInfo =[]
    if len(classIds) != 0:
        for classId, confidence,box in zip(classIds.flatten(),confs.flatten(),bbox):
            className = classNames[classId - 1]
            if className in objects:
                objectInfo.append([box,className])
                if (draw):
                    cv2.rectangle(img,box,color=(0,255,0),thickness=2)
                    cv2.putText(img,classNames[classId-1].upper(),(box[0]+10,box[1]+30),
                    cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0),2)
                    cv2.putText(img,str(round(confidence*100,2)),(box[0]+200,box[1]+30),
                    cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0),2)

    return img,objectInfo


if __name__ == "__main__":
```

```
if __name__ == "__main__":

    cap = cv2.VideoCapture(0)
    cap.set(3,640)
    cap.set(4,480)
    #cap.set(10,70)


    while True:
        success, img = cap.read()
        result, objectInfo = getObjects(img,0.45,0.2)
        #print(objectInfo)
        cv2.imshow("Output",img)
        cv2.waitKey(1)
```

Adding a Library to Run the Above code

```
Setting Up Open-CV for Object Detection
This is an in-depth procedure to follow to get your Raspberry Pi to install Open-CV that will work with Computer Vision for Object Identification. Soon I will create either a script/a
separate tutorial to streamline this process. Turn on a Raspberry Pi 4 Model B running a fresh version of Raspberry Pi 'Buster' OS and connect it to the Internet.

Open up the Terminal by pressing the Terminal Button found on the top left of the button. Copy and paste each command into your Pi's terminal, press Enter, and allow it to finish before
moving onto the next command. If ever prompted, "Do you want to continue? (y/n)" press Y and then the Enter key to continue the process.

sudo apt-get update && sudo apt-get upgrade


We must now expand the swapfile before running the next set of commands. To do this type into terminal this line.

sudo nano /etc/dphys-swapfile


The change the number on CONF_SWAPSIZE = 100 to CONF_SWAPSIZE=2048. Having done this press Ctrl-X, Y, and then Enter Key to save these changes. This change is only temporary and you should
change it back after completing this. To have these changes affect anything we must restart the swapfile by entering the following command to the terminal. Then we will resume Terminal
Commands as normal.

sudo apt-get install build-essential cmake pkg-config

sudo apt-get install libjpeg-dev libtiff5-dev libjasper-dev libpng12-dev

sudo apt-get install libavcodec-dev libavformat-dev libswscale-dev libv4l-dev

sudo apt-get install libxvidcore-dev libx264-dev

sudo apt-get install libgtk2.0-dev libgtk-3-dev

sudo apt-get install libatlas-base-dev gfortran

sudo pip3 install numpy

wget -O opencv.zip https://github.com/opencv/opencv/archive/4.4.0.zip

wget -O opencv_contrib.zip https://github.com/opencv/opencv_contrib/archive/4.4.0.zip

unzip opencv.zip

unzip opencv_contrib.zip
```

```
unzip opencv.zip

unzip opencv_contrib.zip

cd ~/opencv-4.4.0/

mkdir build

cd build

cmake -D CMAKE_BUILD_TYPE=RELEASE \

                              -D CMAKE_INSTALL_PREFIX=/usr/local \

                              -D INSTALL_PYTHON_EXAMPLES=ON \

                              -D OPENCV_EXTRA_MODULES_PATH=~/opencv_contrib-4.4.0/modules \

                              -D BUILD_EXAMPLES=ON ..

make -j $(nproc)


This | make | Command will take over an hour to install and there will be no indication of how much longer it will take. It may also freeze the monitor display. Be ultra patient and it
will work. Once complete you are most of the way done. If it fails at any point and you receive a message like | make: *** [Makefile:163: all] Error 2 | just re-type and enter the above
line | make -j $(nproc) |. Do not fear it will remember all the work it has already done and continue from where it left off. Once complete we will resume terminal commands.

sudo make install && sudo ldconfig

sudo reboot
```

Output

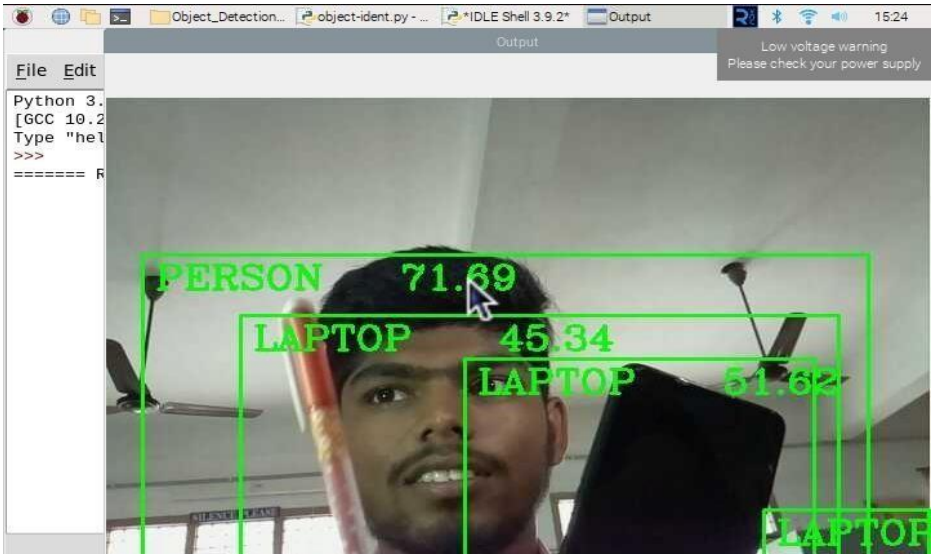


Figure 7.11: Output for Raspberry pi rev1.3 camera

Step 7 – Adding Buzzer

Pin Configuration
GPIO pin - 4
GND – Black

Output for this Step

Code

```
import RPi.GPIO as GPIO
import time

# Buzzer Configuration
buzzer_pin = 4  # Replace with the actual GPIO pin number

GPIO.setmode(GPIO.BCM)
GPIO.setwarnings(False)

# Buzzer Setup
GPIO.setup(buzzer_pin, GPIO.OUT)
GPIO.output(buzzer_pin, False)

def activate_buzzer():
    GPIO.output(buzzer_pin, False)

def deactivate_buzzer():
    GPIO.output(buzzer_pin, True)

if __name__ == '__main__':
    try:
        while True:
            # Your logic here, if needed

            # Example: Activate the buzzer for 1 second
            activate_buzzer()
            time.sleep(1)
            deactivate_buzzer()
            time.sleep(1)

    except KeyboardInterrupt:
        print("Measurement stopped by User")
        GPIO.cleanup()
```

Figure 7.11: Output for Raspberry pi rev1.3 camera

Step 8 – Final Output of Project after adding the Features

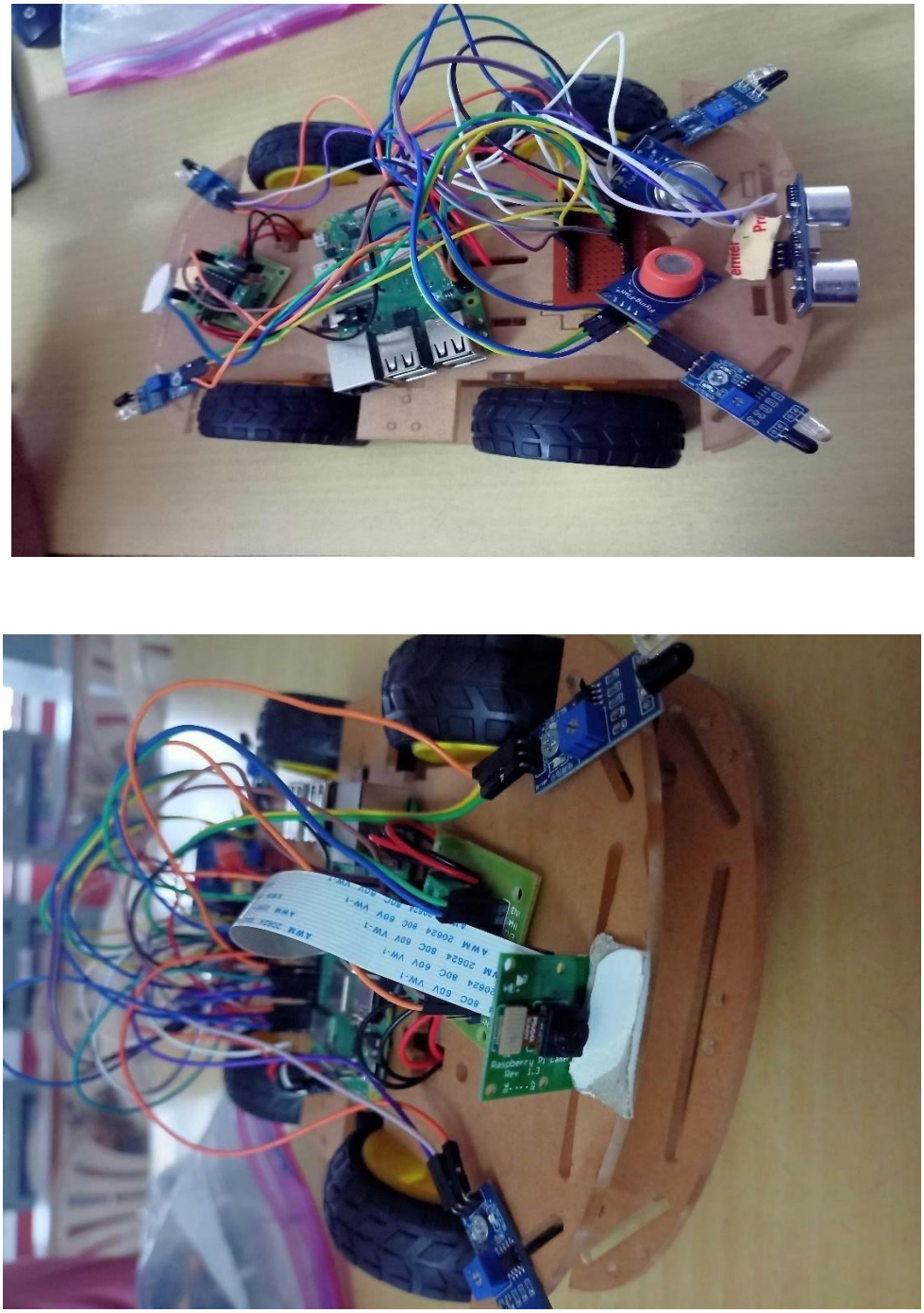


Figure 7.13: Final Project Output

BEFORE MAKING

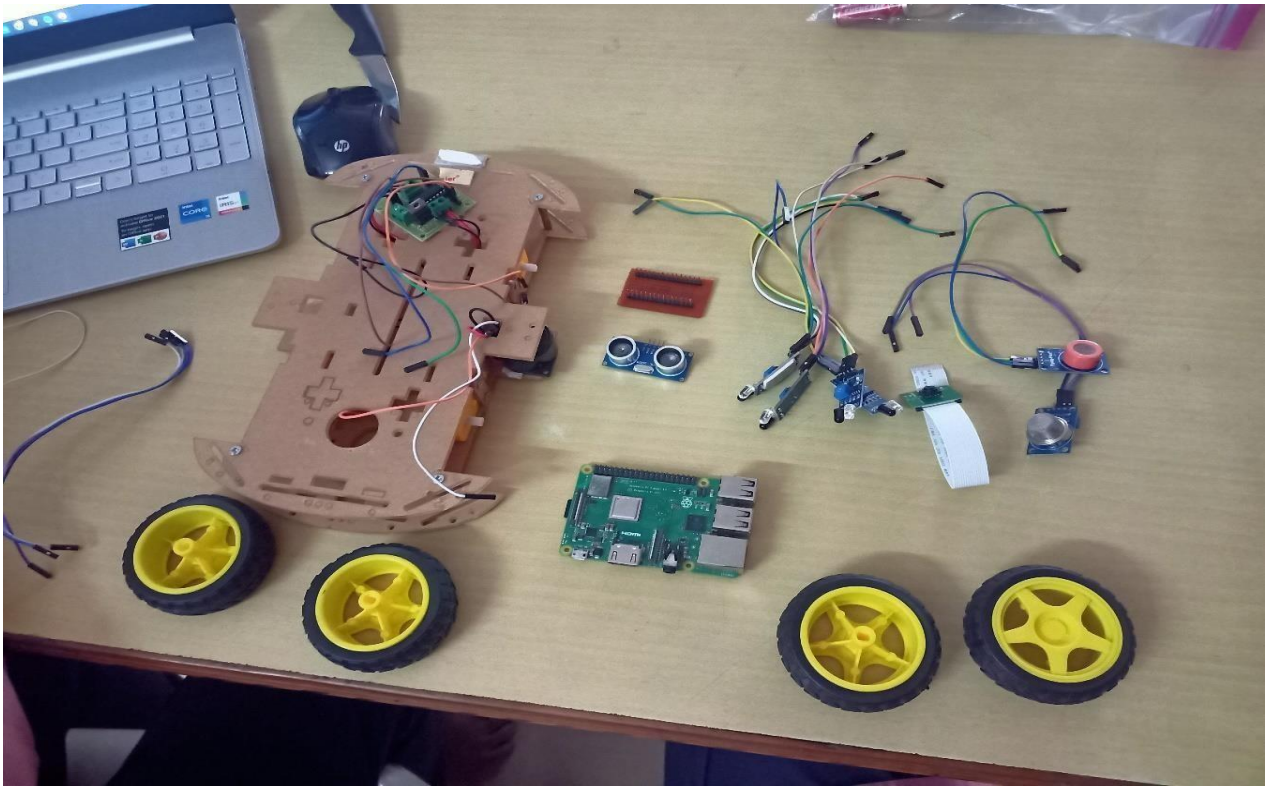


Figure 7.14: Before Connection

AFTER MAKING

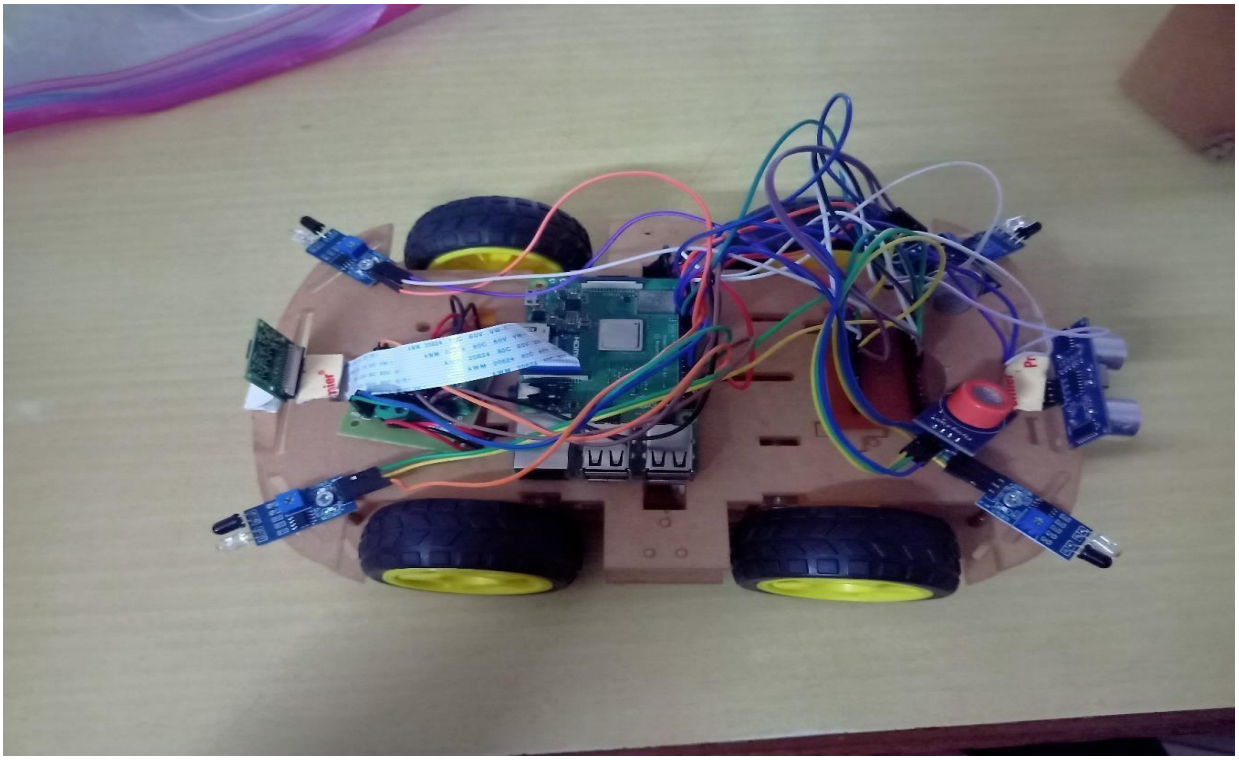


Figure 7.15: After Connection

Chapter-8 Issues Tackled and Project Pathways

8.1 CHALLENGES FACED

Many challenges were faced in implementing this project such as:

8.1.1 Learning

- Learning Python language was difficult in the starting because it is included more in this project.
- Knowing about different libraries like – poho, cv2(OpenCV), TensorFlow, GPIO, and many more was tough, also installing these libraries became a big task for us.
- In installation of these libraries, we faced issues related to network and installation process was time taking.
- Gaining knowledge related to hardware components like Raspberry pi, jumper wires connections, sensors connections, etc became tough in the starting.
- While collecting datasets from search engines it was tough to make the code read the datasets and detect objects in object detection feature.

8.1.2 Connections

a) While connecting Jumper Wires to Raspberry Pi,

- We made sure to have more Jumper wires than required as we have 3 different features in our project and everything must be connected to Raspberry Pi individually.
- After all the connections of sensors, we didn't have enough GPIO pins on Raspberry pi to have separate input and output.
- So, we placed an extension board for VCC and ground in the project.
- Also, in some cases the wires were short for connections so we soldered the wires and connected them.

b) While connecting the alcohol sensor to the Raspberry Pi,

- Firstly, we used sanitizer as it contains some percentage of alcohol but the sensor didn't detect the alcohol as the sanitizer contains less percentage of alcohol.

- Secondly, we used only small amount of alcohol, here detection is done but the accurate alcohol values were not predicted.
- Finally, we used more amount of alcohol and also overcame the challenge of predicting accurate alcohol percentage value.

c) While connecting Ultrasonic sensor to Raspberry Pi,

- As this sensor is used for 2 features Object detection and Smart parking, it was difficult in connecting at first but later we were able to connect it properly.

d) While connecting IR Sensors to Raspberry Pi,

- In our project we used four IR sensors for the feature-Safe parking and connecting all the 4 to the Raspberry Pi was tough.
- At first, the IR sensors used to detect more distance and after adjusting the distance it worked well and we overcame the issue with IR sensors.

e) While connecting Camera to Raspberry Pi,

- In this case, we didn't face difficulty while connecting the Camera to the Raspberry Pi because here no jumper wires or other connections are used as there is a direct connection of camera connector on Raspberry Pi.
- Only this connection was easy compared to others.

f) While connecting Buzzer to Raspberry Pi,

- In buzzer connection we gave special importance because it is connected and used by all 3 features.
- During starting stage, the buzzer used to not respond for all the 3 features but after some successful connections it started responding accurately.
- As the jumper wires are a bit small, we elongated the wired in order for proper connections.

g) While connecting Driver to Raspberry Pi,

- Here, the driver (H-bridge or motor driver board) is connected to Raspberry Pi as it is used to control the direction and speed of motors.
- In our project, 4 motors are used and it was difficult giving the power supply for all 4 constantly as they need more power than the Raspberry Pi.
- So temporarily we inserted a power bank as a power supply in order for the motors to move.

h) While connecting I/O wires to raspberry pi,

- As there are more wires used in our project it was difficult in receiving signals for input and output because of network issues.
- But later we overcame this issue.

i) While connecting Driver to the wheels, • We have 4 wheels and each driver is connected to one of them.

• In the starting, as we used to test our car model the wheels consume more amount of energy, so we replaced the battery usage with power bank.

8.1.3 Coding

- We faced issues while working with MQTT as it works only when the network signals are proper and the code is correct.
- While working on coding part, we faced challenges with some syntax errors and then execution of code was not done, but we cleared all the errors and overcame this issue.
- As we import different libraries for different parts of code it was tough downloading those libraries and executing them accordingly.
- Our project require network with more speed and accuracy, so it took time for all the working implementation and execution of codes.

8.2 PROJECT RELATED LINKS

- We used IEEE Xplore for collecting all the reference papers which are used for reference in this project.

  Link: - https://ieeexplore.ieee.org/Xplore/home.jsp

- While coming to other data like definitions, explanation, etc we took help from Google Search Engine.

  Link: - https://www.google.com/

- We uploaded all our project related code in GitHub link below,

  Link: - https://core-electronics.com.au/guides/object-identifyraspberry-pi/#Down

Chapter-9 The Conclusion

9.1 CONCLUSION

In conclusion, the combination of all 3 features- Object detection, Alcohol detection and Safe parking in this project was successful and ensures safety to the driver and also the passengers using buzzer alert system. By the implementation of this project, we can say that the road accidents can be prevented.

By object detection, the driver will be able to see things which are on backside of the car in the screen and can be alerted through the buzzer if any object is much nearer.

By alcohol detection, the passenger is alerted through the buzzer sound if the driver is a bit drunk and at the higher risk stage the car's engine will be stopped to avoid accidents.

By Safe parking, when the IR sensors detect any object a buzzer sound is observed and also the driver will be able to see the distance between the car and the object on the screen.

In this way our model car will be working as it was modified in such a way to prevent accidents in the blind spot cases also. As technology continue to evolve our project, the scalability of our project allows for future enhancements and adaptations ensuring its relevance and effectiveness in ever-changing landscape. Ultimately, our project represents a significant step towards a safer, smarter and more connected future.

REFERENCE LINKS

1. https://ieeexplore.ieee.org/document/8627998
2. https://ieeexplore.ieee.org/document/8110383
3. https://ieeexplore.ieee.org/document/8589877
4. https://ieeexplore.ieee.org/abstract/document/7430607

5.https://www.researchgate.net/publication/335259703_Real-Time_Car_Detection_and_Driving_Safety_Alarm_System_With_Google_Tensorflow_Object_Detection_API

6. https://ieeexplore.ieee.org/document/8405475
7. https://ieeexplore.ieee.org/document/9316049