DRIVER DROWSINESS DETECTION AND ALERTING WITH VIBRATION AND WATERING SYSTEM

A Project Work
Submitted in partial fulfilment of Requirements for the Award of theDegree of

BACHELOR OF TECHNOLOGY

in

ELECTRONICS AND COMMUNICATION ENGINEERING

By

K. Udaya Sree
(208T1A0442)

I. Naga Anudeep Chakravarthy
(208T1A0434)

K. Manisha
(208T1A0439)

K. Giri Sai Krishna
(208T1A0444)

Under the esteemed guidance of

Dr. M Vinod Kumar (M. Tech, Ph. D)

Associate Professor

DHANEKULA INSTITUTE OF ENGINEERING & TECHNOLOGY

(AUTONOMOUS)

Ganguru, Vijayawada - 521139

Affiliated to JNTUK, Kakinada & Approved By AICTE, New Delhi
Certified by ISO 9001-2015, Accredited by NAAC & NBA

APRIL-2024

DHANEKULA INSTITUTE OF ENGINEERING & TECHNOLOGY

(AUTONOMOUS)

Ganguru, Vijayawada - 521139

Affiliated to JNTUK, Kakinada & Approved By AICTE, New Delhi

Certified by ISO 9001-2015, Accredited by NAAC & NBA

APRIL-2024

CERTIFICATE

This is to certify that the project work entitled "DRIVER DROWSINESS DETECTION AND ALERTING WITH VIBRATION AND WATERING SYSTEM" is a bona fide record of project work done jointly by K. Udaya Sree (208T1A0442), I. Naga Anudeep Chakravarthy (208T1A0434), K. Manisha (208T1A0439) and K. Giri Sai Krishna (208T1A0444) under my guidance and supervision and is submitted in partial fulfillment of the requirements for the award of the Degree of Bachelor of Technology in Electronics & Communication Engineering by Jawaharlal Nehru Technological University, Kakinada during the academic year 2023-2024.

PROJECT GUIDE
Dr. M Vinod Kumar
Associate Professor
Department of ECE

HEAD OF THE DEPARTMENT
Dr. M Vamshi Krishna
Professor & H.O.D
Department of ECE

External Examiner

ACKNOWLEDGEMENT

First and foremost, we sincerely salute our esteemed institution **DHANEKULA INSTITUTE OF ENGINEERING AND TECHNOLOGY** for giving us this opportunity forfulfilling our project.

We express our sincere thanks to our beloved Principal **Dr. Ravi Kadiyala** for providing allthe lab facilities and library required for completing this project successfully.

This project work would not have been possible without the support of many people. We wish to express our gratitude to **Dr. M. Vamshi Krishna**, Ph.D., HOD ECE Department for constant support and valuable knowledge in successful completion of this project.

We are glad to express our deep sense of gratitude to **Dr. M Vinod Kumar (M. Tech, Ph.D),** Associate Professor, for abundant help and offered invaluable assistance, support and guidance throughout this work.

We thank one and all who have rendered help directly or indirectly in the completion of this project successfully.

PROJECT ASSOCIATES

K. Udaya Sree (208T1A0442)

I. Naga Anudeep Chakravarthy (208T1A0434)

K. Manisha (208T1A0439)

K. Giri Sai Krishna (208T1A0444)

DECLARATION

I declare that this project report titled "**DRIVER DROWSINESS DETECTION AND ALERTING WITH VIBRATION AND WATERING SYSTEM"** is submitted in partial fulfillment of the degree of **B. Tech in Electronics and Communication Engineering** is a record of original work carried out by us under the supervision of **Dr. M Vinod Kumar** and has not formed the basis for the award of any other degree or diploma, in this or any other Institution or University. In keeping with the ethical practice in reporting scientific information, due acknowledgments have been made wherever the findings of others have been cited.

K. Udaya Sree (208T1A0442)

I. Naga Anudeep Chakravarthy (208T1A0434)

K. Manisha (208T1A0439)

K. Giri Sai Krishna (208T1A0444)

Ganguru,

April 2024

TABLE OF CONTENTS

DHANEKULA INSTITUTE OF ENGINEERING & TECHNOLOGY

Department of Electronics & Communication Engineering

VISION – MISSION - PEOs

Institute Vision	Pioneering Professional Education through Quality
Institute Mission	Providing Quality Education through state-of-art infrastructure, laboratories and committed staff. Molding Students as proficient, competent, and sociallyresponsible engineering personnel with ingenious intellect. Involving faculty members and students in research and development works for betterment of society.
Department Vision	Pioneering Electronics & Communication Engineeringeducation and research to elevate rural community
Department Mission	Imparting professional education endowed with ethics and human values to transform students to be competent and committed electronics engineers. Adopting best pedagogical methods to maximize knowledgetransfer. Having adequate mechanisms to enhance understanding of theoretical concepts through practice. Establishing an environment conducive for lifelong learning and entrepreneurship development. To train as effective innovators and deploy new technologies for service of society.
Program Educational Objectives (PEOs)	PEO1: Shall have professional competency in electronics and communications with strong foundation in science, mathematics and basic engineering. PEO2: Shall design, analyze and synthesize electronic circuits and simulate using modern tools. PEO3: Shall Discover practical applications and design innovative circuits for Lifelong learning. PEO4: Shall have effective communication skills and practice the ethics consistent with a sense of social responsibility.

STATEMENT OF PO'S AND PSO'S

Program Outcomes

PO1 **Engineering knowledge**: Apply the knowledge of mathematics, science, engineeringfundamentals and engineering programs.

PO2 **Problem analysis**: Identify, formulate, review research literature, and analyse complexEngineering problems reaching substantiated conclusions using first principles of Mathematics, natural sciences, and engineering sciences.

PO3 **Design/development of solutions**: design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental Considerations.

PO4 **Conduct investigations of complex problems**: Use research-based knowledge and research Methods including design of experiments, analysis and interpretation of data,and synthesis of the information to provide valid conclusions.

PO5 **Modern tool usage**: Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modelling to complex engineering activities with an understanding of the limitations.

PO6**The engineer and society**: Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.

PO7 **Environment and sustainability:** Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.

PO8 **Ethics:** Apply ethical principles and commit to professional ethics and responsibilityand norms of the engineering practice.

PO9**Individual and team work:** Function effectively as an individual and as a member or leader in diverse teams and in multidisciplinary settings.

PO10 **Communication:** Communicate effectively on complex engineering activities with theEngineering community and with society at large, such as being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.

PO11 **Project management and finance:** Demonstrate knowledge and understand of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.

PO12 **Life-long learning**: Recognize life-long the need for and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

Program Specific Outcomes

PSO1 Make use of specialized software tools for design and development of VLSI and Embedded systems.

PSO2 Innovative and design application specific electronic circuit for modern wireless communications.

PROJECT MAPPING – PO'S & PSO'S

Project Title	PO 1	PO 2	PO 3	PO 4	PO 5	PO 6	PO 7	PO 8	PO 9	PO10	PO11	PO12
DRIVER DROWSINESS DETECTION AND ALERTING WITH VIBRATION AND WATERING SYSTEM	2	3	3	3	3	2	3	-	3	3	3	3

3-High 2-Medium 1- Low

Justification of Mapping of Project with Program Outcomes:

1. The knowledge of mathematics, science, engineering fundamentals and engineeringprograms are strongly correlated to all course outcomes.
2. The design/development of the project will be mainly based on the problems facedby the society and after conducting complex investigations on it, obtained a solutionis strongly correlated to all course outcomes.
3. Application of Ethical principles is not correlated to all course outcomes**.**

Project vs PSOs Mapping

Project Title	PSO1	PSO2
DRIVER DROWSINESS DETECTION AND ALERTING WITH VIBRATION AND WATERING SYSTEM	3	3

3-High 2-Medium 1- Low

Justification of Mapping of Project with Program Specific Outcomes:

1. This project is related to, which helps to expertise in the corresponding area by applying engineering fundamentals and are strongly correlated to all course outcomes.
2. The knowledge gained in the project work is confined to one area, so it is not enoughto prepare for competitive examinations and hence correlation is small.

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVATIONS

ABBREVATION	DESCRIPTION
EAR	Eye Aspect Ratio
MAR	Mouth Aspect Ratio
HDMI	High-Definition Multimedia Interface
VNC	Virtual Network Computing

ABSTRACT

"Driver Drowsiness Detection and Alerting with Vibration and Watering System " involves leveraging the computational power of Raspberry Pi to develop a system that can detect and mitigate driver drowsiness in real-time. The setup includes a USB camera connected to the Raspberry Pi, which captures live video footage of the driver's face while on the road. Using image processing techniques, various parameters such as lip distance and eye aspect ratio are continuously monitored to assess the driver's level of drowsiness. When signs of drowsiness are detected, indicating potential fatigue or sleepiness, the system triggers preventive measures to alert the driver and prevent accidents. This includes activating a pump mechanism to release a wake-up agent or scent in the car cabin to stimulate the driver's senses. Additionally, a vibration motor connected via relay is activated to provide physical feedback, such as vibrating the driver's seat, further alerting them to the need for immediate attention. Overall, the proposed system offers a proactive approach to addressing the serious issue of driver drowsiness, particularly during long journeys or monotonous driving conditions. By leveraging Raspberry Pi's capabilities for real-time image processing and control of peripheral devices, the system aims to enhance road safety and prevent accidents caused by driver fatigue.

Keywords: Raspberry Pi, web camera, Relay, DC water pump, Vibration motor

CHAPTER - 1

INTRODUCTION

1.1 Introduction:

Driver drowsiness is a critical issue contributing to road accidents worldwide, posing a significant risk to both drivers and passengers. To address this concern, the project "Driver Drowsiness Detection Using Raspberry Pi" aims to develop an intelligent system capable of detecting signs of driver fatigue in real-time and providing timely alerts to prevent accidents. Leveraging the computational power and versatility of Raspberry Pi, this project seeks to create a reliable solution that enhances road safety and mitigates the dangers associated with drowsy driving.

The system employs a USB camera connected to the Raspberry Pi, which continuously captures live video footage of the driver's face during travel. Through sophisticated image processing algorithms, various facial parameters such as lip distance and eye aspect ratio are analyzed to assess the driver's level of alertness. By monitoring subtle changes in facial expressions and eye movements, the system can accurately detect signs of drowsiness, indicating potential fatigue or sleepiness.

Upon detecting signs of drowsiness, the system initiates preventive measures to alert the driver and mitigate the risk of accidents. This includes activating a pump mechanism to release a wake-up agent or scent in the car cabin, stimulating the driver's senses and promoting alertness. Additionally, a vibration motor connected to the Raspberry Pi via relay is triggered to provide physical feedback, such as vibrating the driver's seat, thereby prompting immediate attention. Through these proactive measures, the project aims to enhance road safety by addressing the root cause of drowsy driving and preventing potential accidents before they occur.

1.2 Implementation of Embedded system

1.2.1 Introduction for Embedded system

An embedded system is one kind of a computer system mainly designed to perform several tasks like to access, process, and store and also control the data in various electronics-based systems. Embedded systems are a combination of hardware and software where software is usually known as firmware that is embedded into the hardware. One of its most important characteristics of these systems is, it gives the o/p within the time limits. Embedded systems support to make the work more perfect and convenient. So, we frequently use embedded systems in simple and complex devices too.

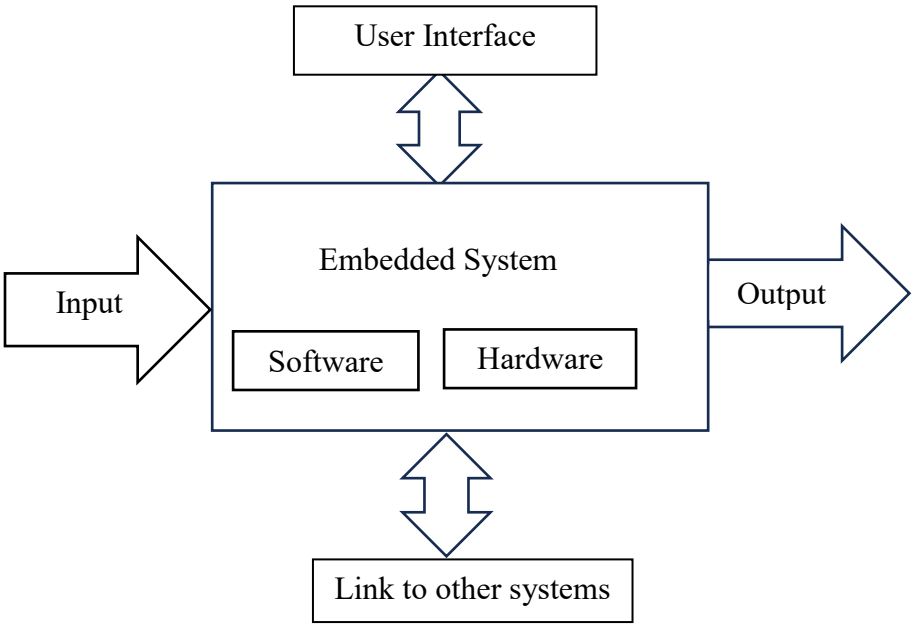

Figure 1.1: Overview of embedded system

1.2.2 Embedded system:

Embedded system includes mainly two sections, they are

1. Hardware
2. Software

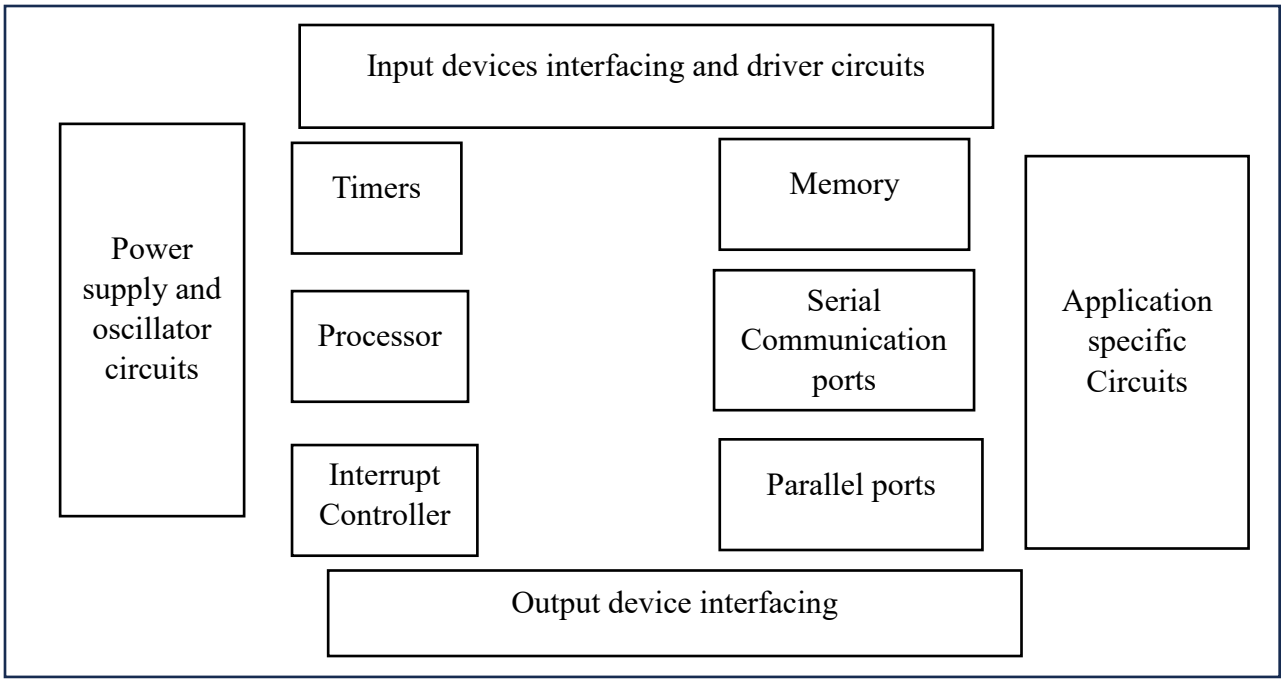

Figure 1.2: Block Diagram of Embedded System

1.3 Embedded System Hardware

1.3.1 Introduction

As with any electronic system, an embedded system requires a hardware platform on which it performs the operation. Embedded system hardware is built with a microprocessor or microcontroller. The embedded system hardware has elements like input output (I/O) interfaces, user interface, memory and the display. Usually, an embedded system consists of:

- Power Supply
- Processor
- Memory
- Timers
- Serial communication ports
- Output/Output circuits
- System application specific circuits

Embedded systems use different processors for its desired operation. Some of the processors used are

1. Microprocessor
2. Microcontroller
3. Digital signal processor

1.3.2 Microprocessor vs. Microcontroller:

Microprocessor:	Microcontroller:
• **CPU** on a chip. • We can attach required amount of ROM, RAM and I/O ports. • Expensive due to external peripherals. • Large in size • general-purpose	• **Computer** on a chip • fixed amount of on-chip ROM, RAM, I/O ports • Low cost. • Compact in size. • Specific –purpose

Table 1.1: Microprocessor vs. Microcontroller

1.4 Embedded System Software:

1.4.1 Introduction

The embedded system software is written to perform a specific function. It is typically written in a high-level format and then compiled down to provide code that can be lodged within a non-volatile memory within the hardware. An embedded system software is designed to keep in view of the three limits:

- Availability of system memory
- Availability of processor's speed
- When the system runs continuously, there is a need to limit power dissipation for events like stop, run and wake up.

1.4.2 Bringing software and hardware together for embedded system:

To make software to work with embedded systems we need to bring software and hardware together .for this purpose we need to burn our source code into microprocessor or microcontroller which is a hardware component and which takes care of all operations to be done by embedded system according to our code.

Generally, we write source codes for embedded systems in assembly language, but the processors run only executable files. The process of converting the source code representation of your embedded software into an executable binary image involves three distinct steps:

1. Each of the source files must be compiled or assembled into an object file.
2. All of the object files that result from the first step must be linked together to produce a single object file, called the re-locatable program.
3. Physical memory addresses must be assigned to the relative offsets within the re-locatable program in a process called relocation.
4. The result of the final step is a file containing an executable binary image that is ready to run on the embedded system.

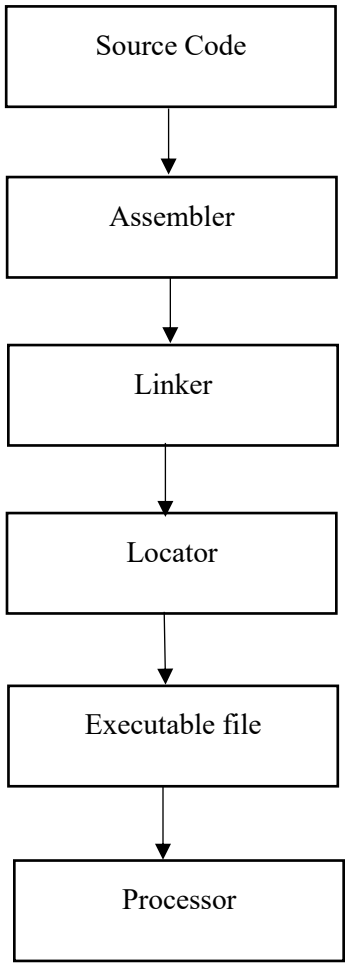

Figure 1.3: Flow of burning source code to processor

1.5 Applications:

Embedded systems have different applications. A few select applications of embedded systems are smart cards, telecommunications, satellites, missiles, digital consumer electronics, computer networking, etc.

Embedded Systems in Automobiles

- Motor Control System
- Engine or Body Safety
- Robotics in Assembly Line
- Mobile and E-Com Access

Embedded systems in Telecommunications

- Mobile computing
- Networking
- Wireless Communications

Embedded Systems in Smart Cards

- Banking
- Telephone
- Security Systems

1.6 Implementation flow:

Stage 1:

Considering the problems of existing methods and giving solution to that problem by considering the basic requirements for our proposed system.

Stage 2:

Considering the hardware requirement for the proposed system

For this we need to select the below components:

1. Microcontroller

2. Inputs for the proposed system (ex: sensors, drivers etc...,)

3. Outputs (ex: relays, loads)

Stage 3:

After considering hardware requirements, now we need to check out the software requirements. Based on the microcontroller we select there exists different software for coding, compiling, debugging. we need to write source code for that proposed system based on our requirements and compile, debug the code in that software. After completing all the requirements of software and hardware we need to bring both together to work our system. For this we need to burn our source code into microcontroller, after burning our source code to microcontroller then connect all input and output modules as per our requirement.

CHAPTER-2

LITERATURE SURVEY

[1] Driver cognitive is one of the faults for an accident lead significantly to dangerous accident. In this Paper a semi-supervised Gaussian Mixture Model (GMM) Clustering data representing Sternum a motion obtained Mixture is applied to acceleration of drivers a head, neck and simulated driving through. Nowadays most of the people were changing their petrol/diesel car to electric car because the electric car has more innovation feature like smart suspension, speed control etc. Hybrid monitoring system is detection techies that are used to find both conditions should satisfy the needs. It is verify used to measure the driver state ECG, GRS in vehicle parameters like yawn rate, speed steering wheel entropy, and various action.[2] Anil Kumar Biswal et.al constructed an alert system to continuously monitor the collision and drowsiness of drivers using the technology of IOT. The Raspberry Pi was exploited as a single-board computer to find the fatigue of drivers and focus on behavioural measures. The crash sensors and FSR sensors were utilized to detect extremity collisions. If drowsiness is perceived, a voice message alert, as well as mail, is sent to the driver instantly. Although the alert systems work properly if any collision happened, the warning or alert messages have been sent to the nearby hospitals located in the prone zone using a Google map link. The OpenCV library package was imported and to analysis eye facial measures computer vision algorithms were used. To avoid mishaps due to drowsiness the proposed system alerts the driver by sending mail using Simple Mail Transfer Protocol (SMPT) using smptlib in python. The usage of SSH prevents unauthorized access. Totally 97% accuracy has been achieved with this research work. [3] Driving when fatigued is a serious issue on the highways. Using OpenCV and the Raspberry Pi 3, a model for sleepiness detection has been constructed. If drowsiness is detected, a warning will be generated. Identifying the degree of tiredness is the primary objective of endeavour. It is a single board computer running the Linux operating system that is used to build and develop the system to detect driver fatigue. The system will warn drivers while they are driving vertical if they are feeling drowsy. As an open-source imitation, Python with OpenCV allows anybody to contribute to the libraries, manuals, and tutorials. There are just a few places where automatic object detection may be included. Controlling a vehicle's acceleration can change that how nearby vehicles react to drowsy of the drivers. [4] Deployment of vision-based drowsiness detection techniques in practice is still far from adequate, despite having achieved great success on empirically organized datasets. The scarcity and absence of datasets that correctly depict the challenges faced in real-world applications, such as the great variety and accumulation of visual cues, as well as the challenges given by diverse camera types and positions, is a key issue. To promote research in this field, we propose a new large-scale dataset, Fatigue View, collected by RGB and

infrared (IR) cameras from five different locations. Real-world film of sleepy drivers on the road is included, along with a variety of subtle to overt tiredness cues, including 17,403 different yawning sets totalling more than 124 million. frames, which is a huge increase over previously active datasets. Offer hierarchical annotations for each video to meet different study demands. These annotations might be anything from temporal drowsiness levels and places to spatial face landmarks and visual signals. Assess representative procedures architecturally in order to establish useful baselines. With Fatigue View, it aims to motivate the community to adapt computer vision models to handle believable real-world problems, particularly those datasets that are presently available. One of the main causes of Malaysia's rising accident numbers is sleepiness among drivers. Thus, this work proposes the design and development of a driver drowsiness detection system based on image processing employing a Raspberry Pi camera module sensor interacting with a Raspberry Pi 3 board. In order to accomplish the research goal, the Eye Aspect Ratio (EAR) method is used for eye blink (open and close) detection, while the Haar Cascade Classifier technique is used for face and eye detection. Six recruited people participated in multiple trials, and the results showed that correct sitting position (head facing the camera) and the absence of eyewear or sunglasses affected the Haar Cascade classifier's ability to accurately distinguish faces and eyes. In the meantime, the system identified an average EAR value ranging from 0.141 (with eyes closed) to 0.339 (with eyes open). In conclusion, the Raspberry Pi platform's image processing-based Haar Cascade and EAR algorithms have been implemented effectively. Future improvements could include adding a physiologically based analysis using alcohol and heart rate sensors and replacing the existing board with a Raspberry Pi Touch Screen to reduce the hardware setup[5]. Examining the data supporting the connection between weariness and safety—particularly in the context of transportation and the workplace—was the aim of this review. Fatigue was defined as "a biological drive for recuperative rest" for the purposes of this review. The review looked first at accidents and injuries before examining negative impacts on performance. It then looked at the relationship between three key causes of fatigue and safety outcomes: type of task effects, circadian influences, and sleep homeostasis parameters. The review provided convincing proof that sleep homeostatic effects can lead to accidents and poor performance. Due to a paucity of research, the nature of task effects—particularly those involving monotonous, prolonged attention—also resulted in notable performance declines, but the implications on accidents and/or injuries remained unclear. More research is required to elucidate the relationship between circadian-related fatigue affects and performance or safety outcomes, as the results did not show a direct link between them.

There's no denying that circadian fluctuation has an impact on safety outcomes, but the data points to a confluence of sleep-related and daytime elements as well. Likewise, whereas certain performance metrics exhibit a direct circadian component, others seem to only demonstrate one in conjunction with sleep-related variables. The review emphasised areas of weakness in the body of literature as well as directions for future study[6]. The current review's findings demonstrated that several strategies have been applied globally to reduce sleepy driving. The exact impact of each intervention is still unknown, despite the fact that these can be utilised in nations with high rates of traffic accidents. To compare the effectiveness of each intervention and localise each intervention in accordance with the traffic patterns of each nation, more research is necessary[7]. Currently, the mining, transportation, and professional sports sectors are using wearable sensors to control weariness. Physical weariness is a difficult ergonomic/safety "issue" in production since it reduces productivity and raises the risk of accidents. Physical exhaustion must therefore be controlled. This study has two main objectives. Initially, we look at how wearable sensors can be used to identify instances of physical exhaustion in industrial operations that are mimicked. Estimating the degree of physical exhaustion over time is the second objective. To accomplish these objectives, sensory data for eight persons in good health were collected. Physical exhaustion identification and level estimate were performed using penalised logistic regression and multiple linear regression models, respectively. A well-liked variable selection mechanism called Least Absolute Shrinkage and Selection Operator (LASSO) was used to choose significant features from each of the five sensor locations. The outcomes demonstrate how effectively the LASSO model worked for both modelling and physical fatigue detection. The modelling technique can be used to various applications since it is not participant- or workload-specific[8]. One of the main factors in road accidents is driver weariness. One of the most promising commercial applications based on facial expression analysis technologies is automatic vision-based driver fatigue recognition. Generally speaking, duplicate data, illumination effects, noise, and image scale have an impact on how well facial emotion detection algorithm’s function. In this paper, we have proposed an effective algorithm that can both overcome the aforementioned challenges and operate with multi-scale images. There are three primary phases to the proposed framework. Using a discrete wavelet transform, the input image is first divided into four sub-band images while maintaining the crucial facial image information. In order to create images of various sizes, the source image is also down-sampled. Each image is then further separated into a number of pieces that are categorised as either informative or non-informative blocks based on entropy analysis. The second step uses the discrete cosine transform to choose the high variance features in a zigzag pattern. To accurately categorise the expressions

into seven generic expression classes, classifiers are trained and tested in the final stage. The empirical findings imply that the suggested framework performs better in terms of classification accuracy rate than other comparable approaches in addition to making efficient use of the multi-scale images[9]. Using a field location suggested by a local operator of the French road network, we studied how drivers' lateral position in relation to risk on rural crest steep curves (Conseil General de Maine-et-Loire, 49). Testing a single road treatment at this field site was the ultimate objective. There were three phases of the investigation. First, two perceptual treatments (i.e., sealed shoulders and rumble strips on both sides of the centreline) were chosen from five that were examined using driving simulations to aid drivers retain lateral control when driving on crest steep curves. On the field, the rumble strips were put in first. The second step was creating a diagnostic tool expressly designed to assess, on the job site, how a perceptual treatment affected the driver's performance (i.e., lateral position). The target crest vertical curve's upstream and downstream portions of the field were equipped with this diagnostic gadget. The final step involved gathering data both before and after the centreline rumble strips were put in place. Next, we contrasted the field study results with the driving simulator study results. The comparison revealed that the crest vertical curve's centreline rumble strips had an impact on lateral positions, leading participants to drive closer to the lane's centre, just like in the simulator experiments. Ultimately, the outcomes demonstrated the value of driving simulators in the process of designing roads[10]. Cognitive weariness in drivers can seriously impair their ability to drive and may result in tragic collisions. This work examines the automatic identification of underload driver cognitive fatigue by an analysis of upper body posture dynamics. A semi-supervised method is devised to discern the patterns of cognitive tiredness associated with driver posture. Using a motion capture suit, acceleration data representing the driver's head, neck, and sternum is first subjected to an unsupervised Gaussian Mixture Model (GMM) clustering. This yields the best possible clusters from the time-series data of driver upper posture that are the most correlated and similar. Subsequently, an algorithm for automatic labelling is created, which analyses the maximum value and standard deviation of every GMM cluster and designates a symbol based on the differences in postural behaviour. In order to detect driver weariness, new machine learning supervised classifiers are trained on the GMM-labelled upper body posture dataset as real-time algorithms. These classifiers include Gaussian Support Vector Machines and Bootstrap-Aggregating based Ensemble Classifiers.

The suggested technique was verified using an electroencephalogram-based neurophysiological method that measures cognitive tiredness. The findings demonstrate that the suggested semi-supervised method performs better at accurately identifying the patterns of cognitive tiredness than the current state-of-the-art methods. For two test subjects, it accurately detects various driving positions with 93% and 90% accuracy rates. There includes a discussion of the flaws of the proposed work as well as some options for expanding the current effort[11-14].

CHAPTER-3

PLATFORM USED – RASPBERRY PI OS (LEGACY 32-Bit)

3.1 Platform used

Raspberry Pi OS (formerly Raspbian) is a Unix-like operating system based on the Debian GNU/Linux distribution for the Raspberry Pi family of compact single-board computers. First developed independently in 2012, it has been produced as the primary operating system for these boards since 2013, distributed by the Raspberry Pi Foundation.

Raspberry Pi OS is highly optimized for the Raspberry Pi with ARM CPUs. It runs on every Raspberry Pi except the Pico microcontroller. Raspberry Pi OS uses a modified LXDE desktop environment with the Open box stacking window manager, along with a unique theme. The default distribution is shipped with a copy of the computer algebra system Wolfram Mathematica, VLC, and a lightweight version of the Chromium web browser.

3.2 Download and Install OS:

Step 1: Download Raspberry pi Imager in your computer

Step 2: Run Raspberry pi Imager and allow the User control access.

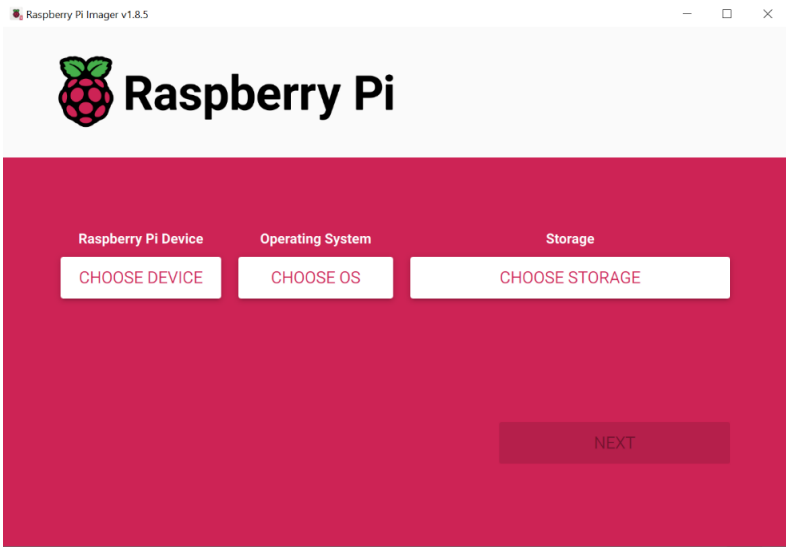

Figure 3.1: Raspberry pi Imager

Step 3: Select the version of Raspberry pi at “CHOOSE DEVICE”.

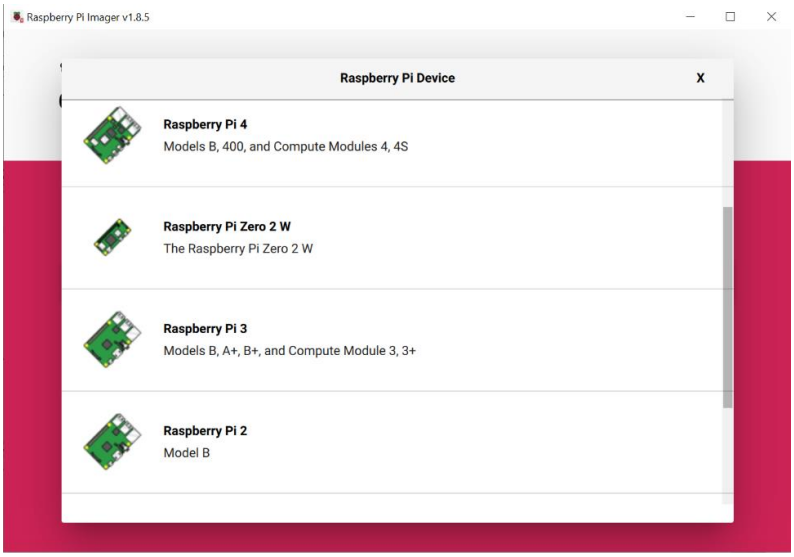

Figure 3.2: Device Selection

Step 4: Select the version of OS that you want to install in your raspberry pi

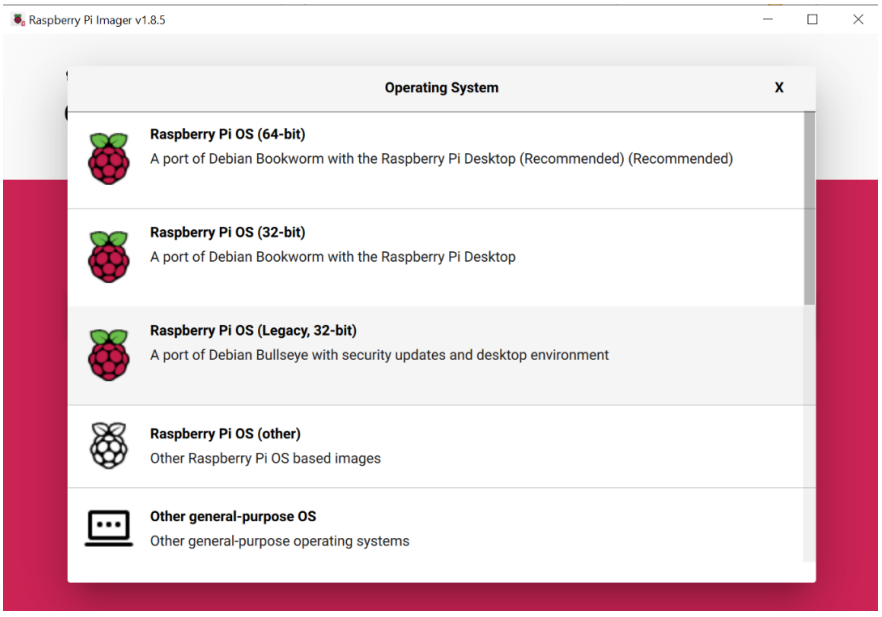

Figure 3.3: Operating System

Step 5: Select the storage in which you want to install the Operating System.

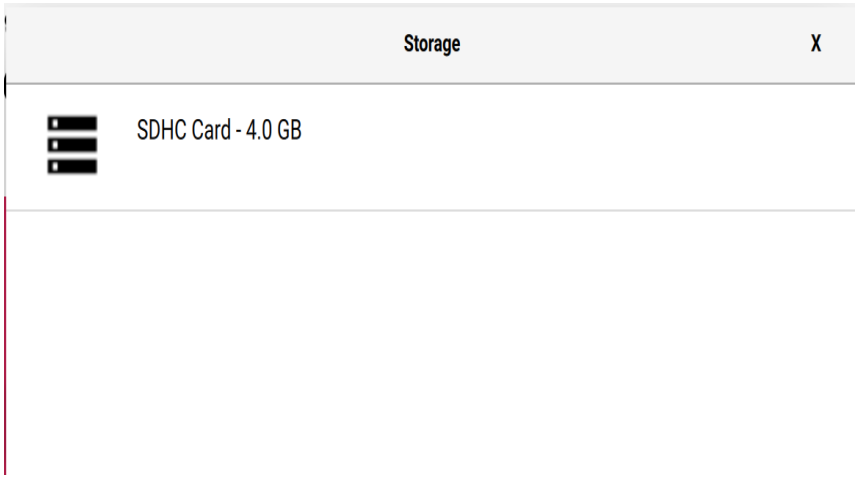

Figure 3.4: Storage Selection

Step 6: After selecting all the required fields click on next to Install.

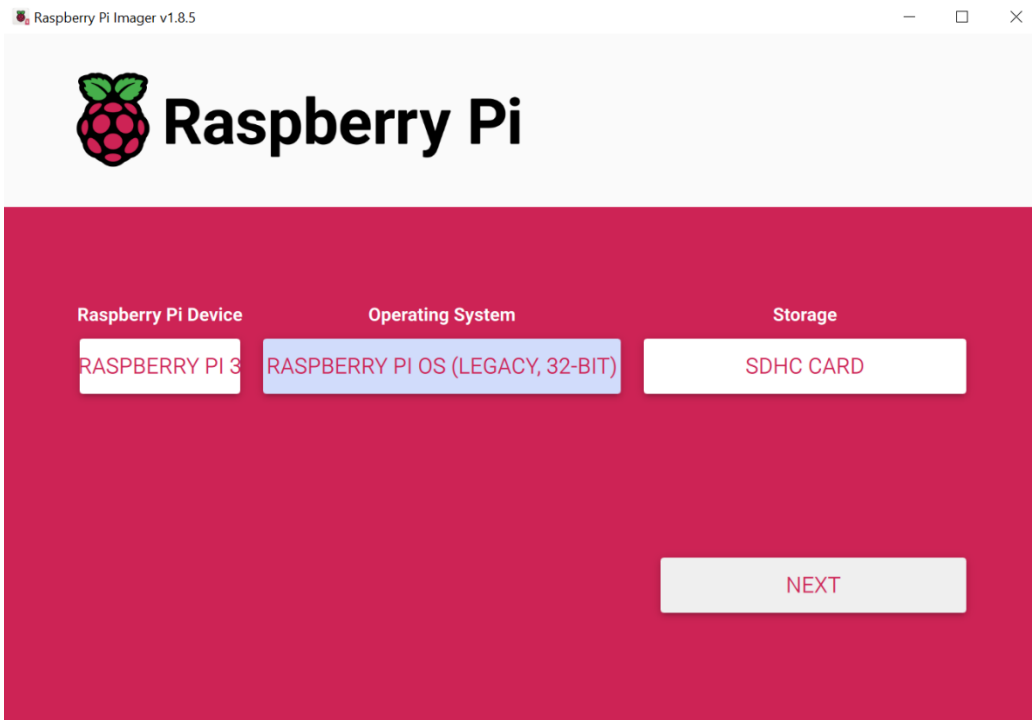

Figure 3.5: OS Installation

Note: Make sure that the SD card using is formatted before installing the OS.

3.3 Setting up the Device

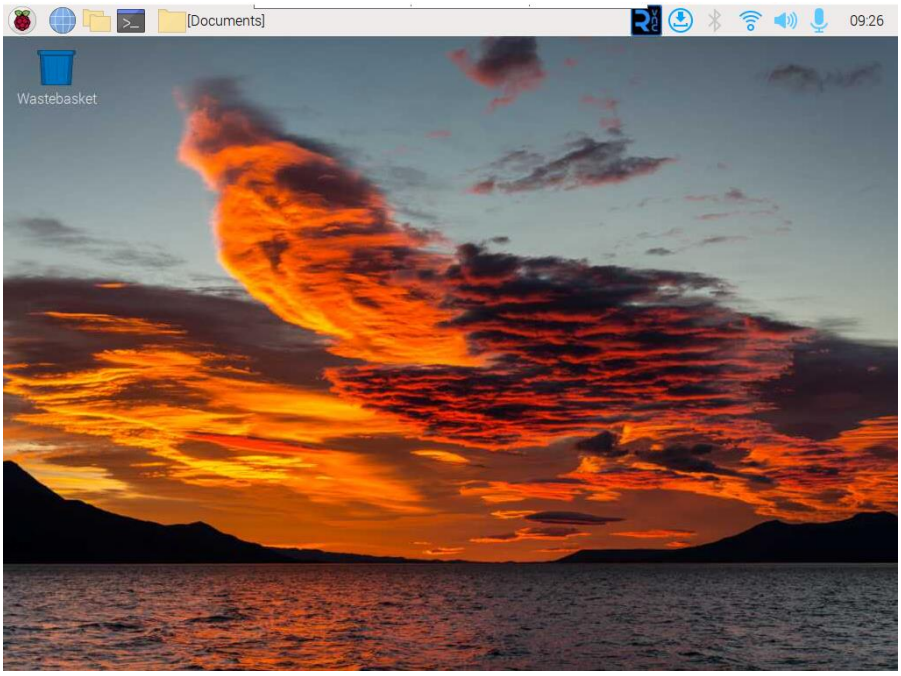

Figure 3.6: Setting up the Device

3.4 Use Command Prompt to install IDLE or any services

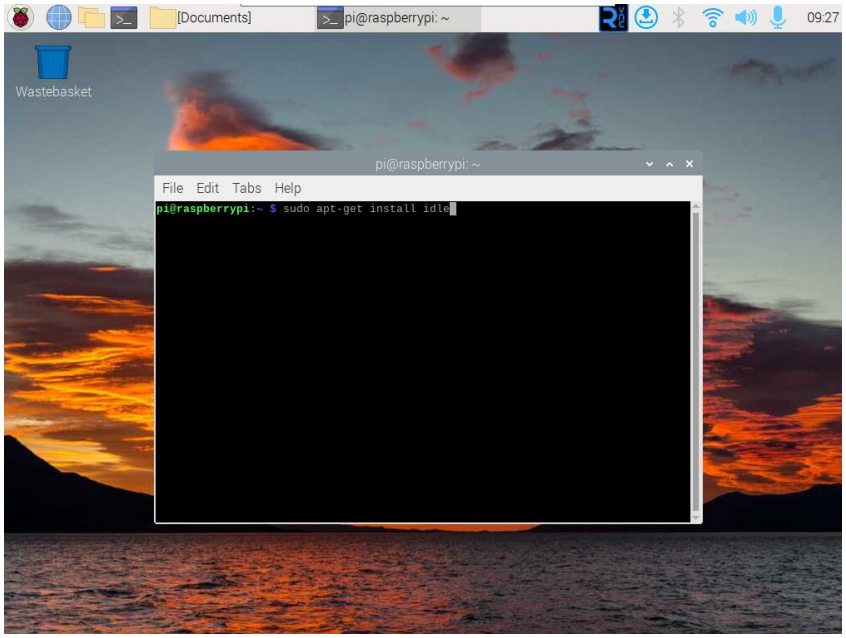

Figure 3.7: Installing IDLE

3.5 Interfacing Raspberry pi Display

We can use different methods to interface the raspberry pi. These are a couple of methods which are used frequently.

3.5.1 External Monitor (HDMI)

Basically, Raspberry pi contains 2 HDMI ports

- Micro HDMI port
- Standard HDMI

By connecting the external display using these ports

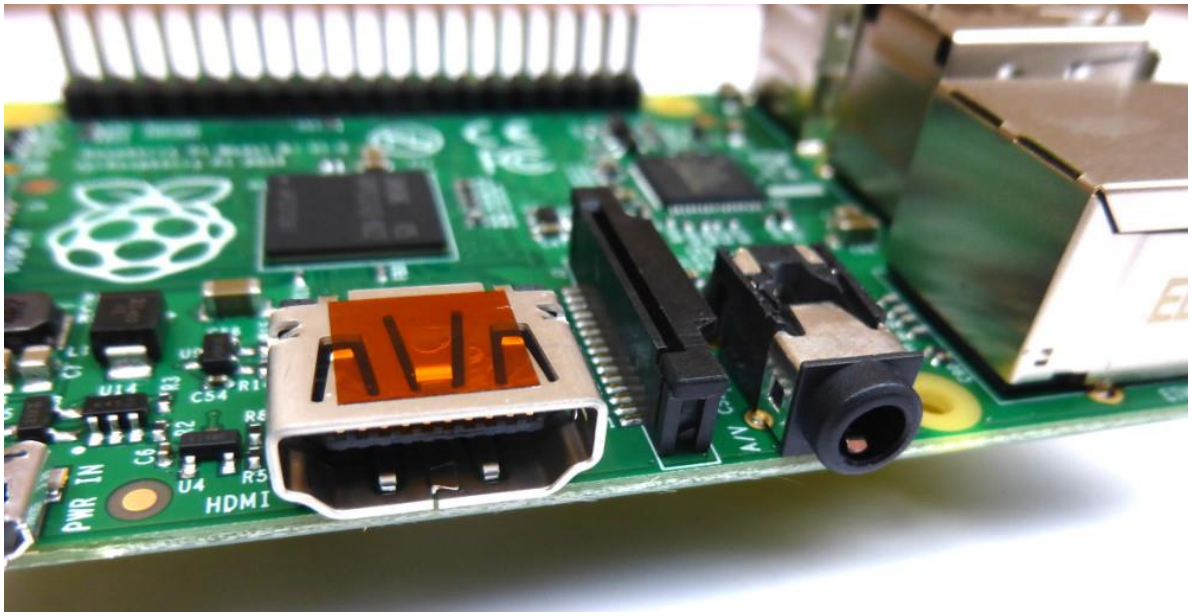

Figure 3.8: HDMI Ports

3.5.2 VNC Viewer

Step 1: Open raspberry pi configurations from references

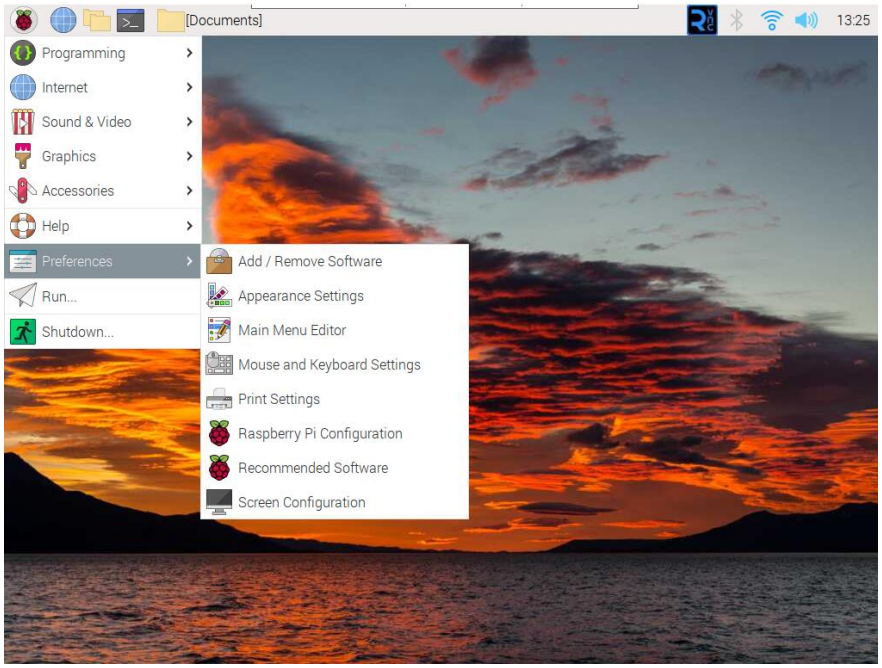

Figure 3.9: Raspberry pi Configuration

Step 2: Select interfaces, Switch on the VNC

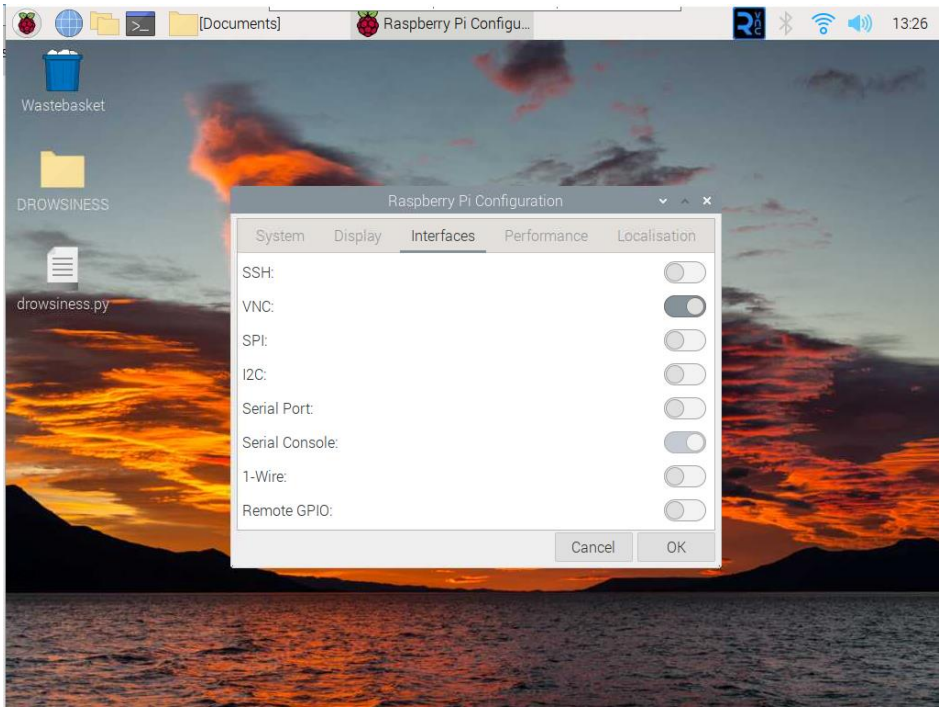

Figure 3.10: Switching on VNC

Step 3: Now click on the VNC ion present in the task bar to get the Ip address

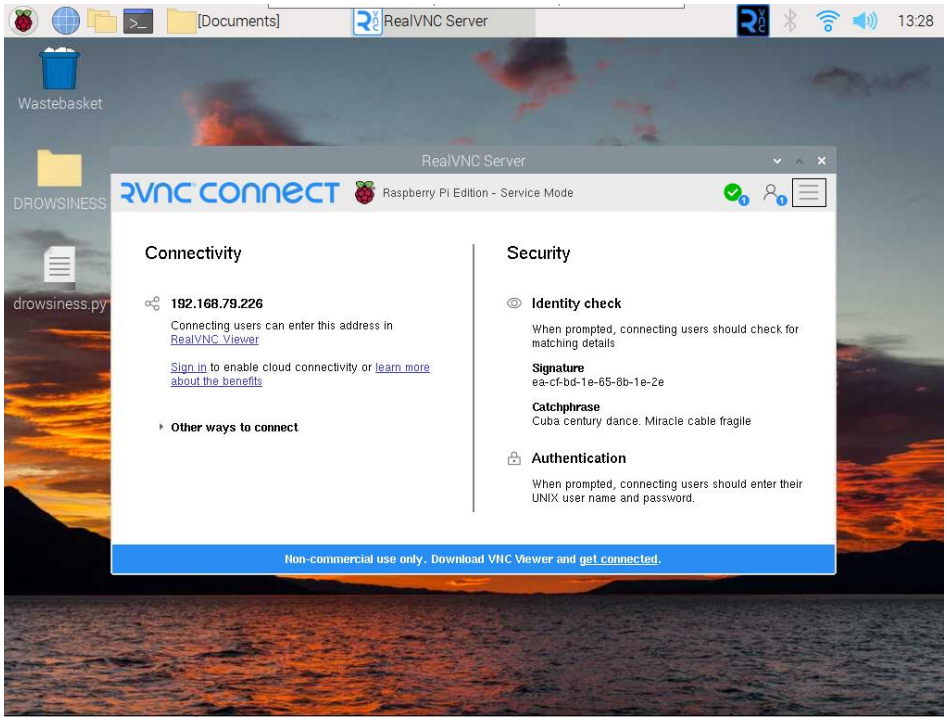

Figure 3.11: VNC IP Address

Step 4: Open the real VNC viewer app in your system/laptop and search the Ip address visible in your raspberry pi desktop.

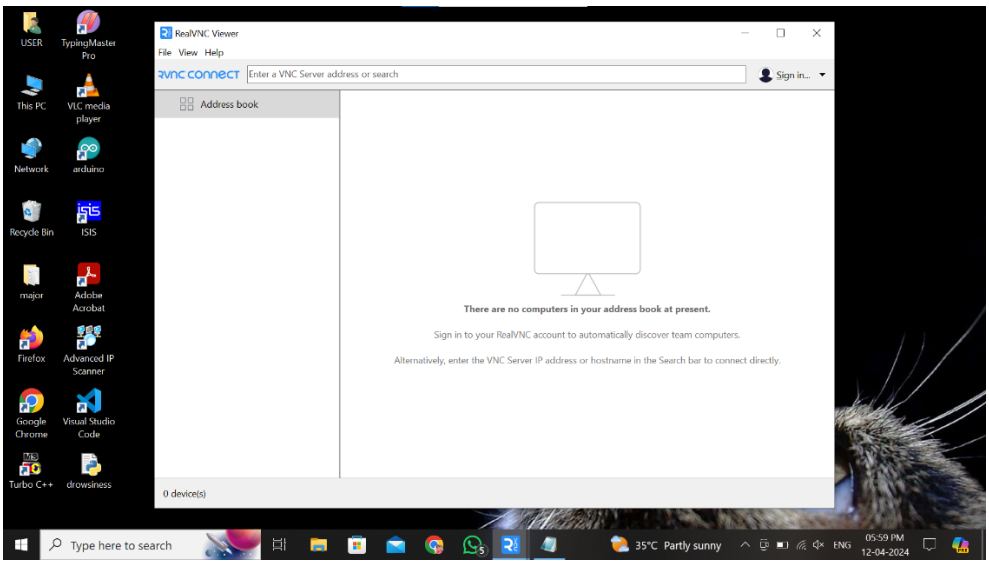

Figure 3.12: VNC Application

Step5: After entering the Ip address a window will pop up by asking the raspberry pi password

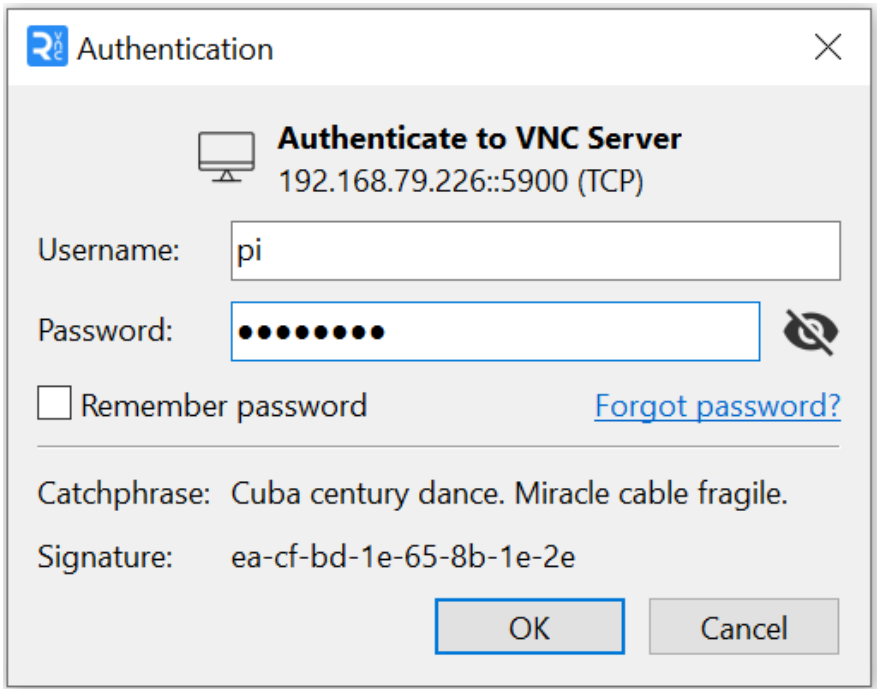

Figure 3.13: Authenticate to VNC Server

Step6: After entering the raspberry pi the password, the raspberry pi will be connected to your system/laptop

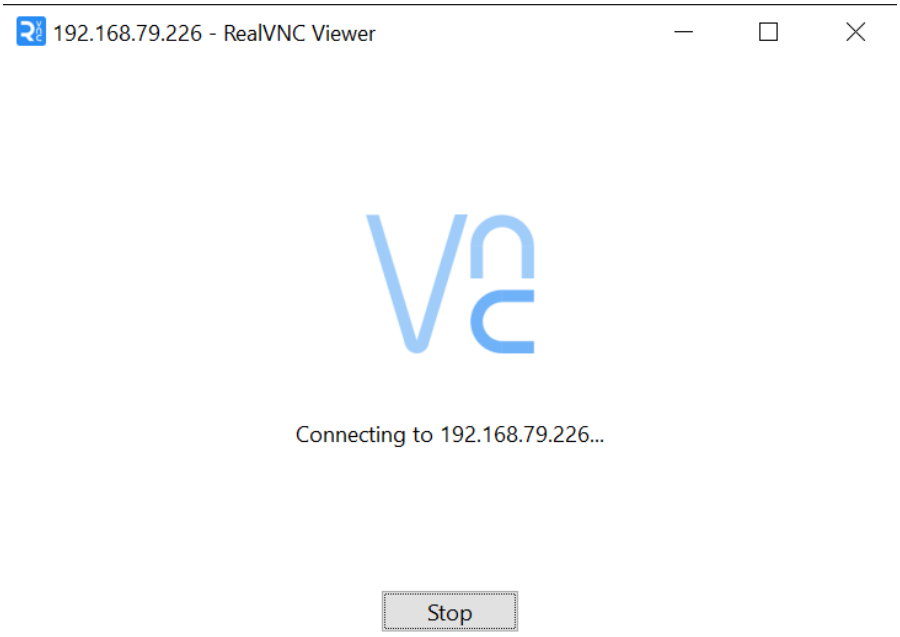

Figure 3.14: Connecting VNC

Step7: Now you can access the raspberry pi with your system/laptop

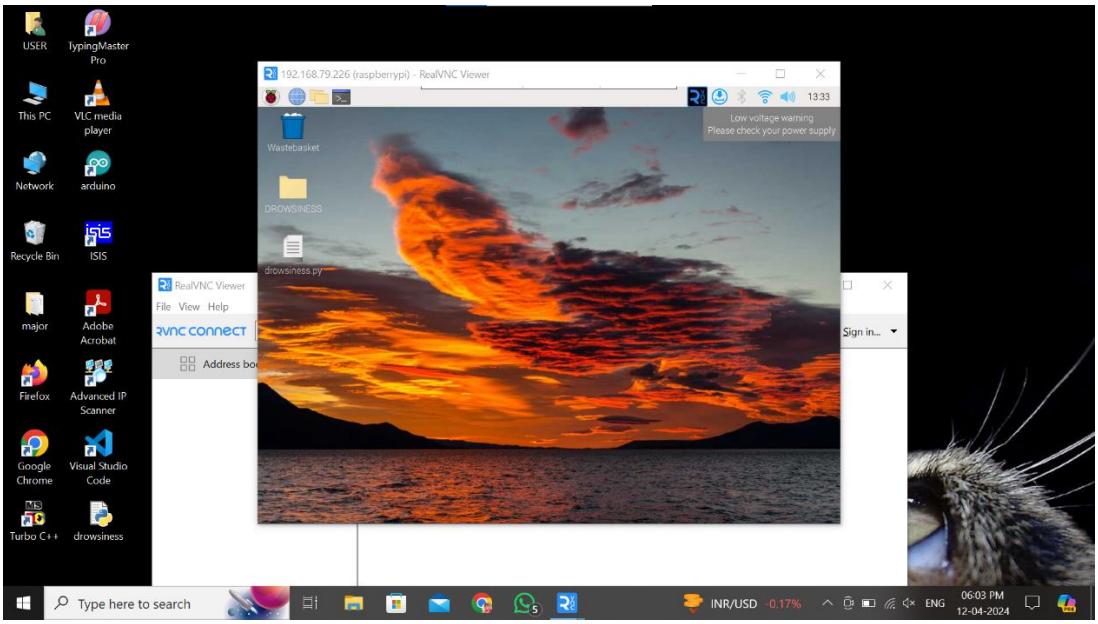

Figure 3.15: VNC in your Device

CHAPTER-4

PROPOSED SYSTEM MODEL

4.1 Existing System

The existing method for driver drowsiness detection often relies on standalone hardware devices or mobile applications that utilize basic sensors like accelerometers or gyroscopes to detect vehicle movements or driver head nods. These systems typically lack the sophistication needed to accurately assess the driver's alertness level and may produce false alarms or fail to detect subtle signs of drowsiness. Moreover, some methods involve wearable devices like smartwatches or headbands, which may be uncomfortable for drivers to wear for extended periods or may interfere with their driving experience. Overall, the current approaches to drowsiness detection often suffer from limited accuracy, reliability, and user convenience, highlighting the need for more advanced and integrated solutions.

4.1.1 Drawbacks:

1. Limited Accuracy
2. Reliability Issues
3. Inconvenience for Drivers
4. False Alarms
5. Lack of Sophistication

Figure 4.1: Existing System

4.2 Proposed System

In the proposed method, we aim to enhance the accuracy and reliability of driver drowsiness detection using Raspberry Pi. The system will utilize a USB camera connected to the Raspberry Pi to continuously monitor the driver's facial features and eye movements. By analysing parameters such as lip distance and eye aspect ratio, the system will be able to accurately detect signs of drowsiness in real-time. Additionally, advanced machine learning algorithms will be employed to improve the detection accuracy and minimize false alarms.

Upon detecting signs of drowsiness, the proposed system will activate safety measures to alert the driver and prevent potential accidents. This will include activating a pump connected to a vibration motor using a relay mechanism. The vibration motor will generate tactile feedback, alerting the driver to their drowsy state and prompting them to take corrective action. By integrating these features into the vehicle's system, the proposed method aims to provide an effective solution for preventing accidents caused by driver drowsiness.

Figure 4.2: Proposed Method

4.2.1 Block Diagram

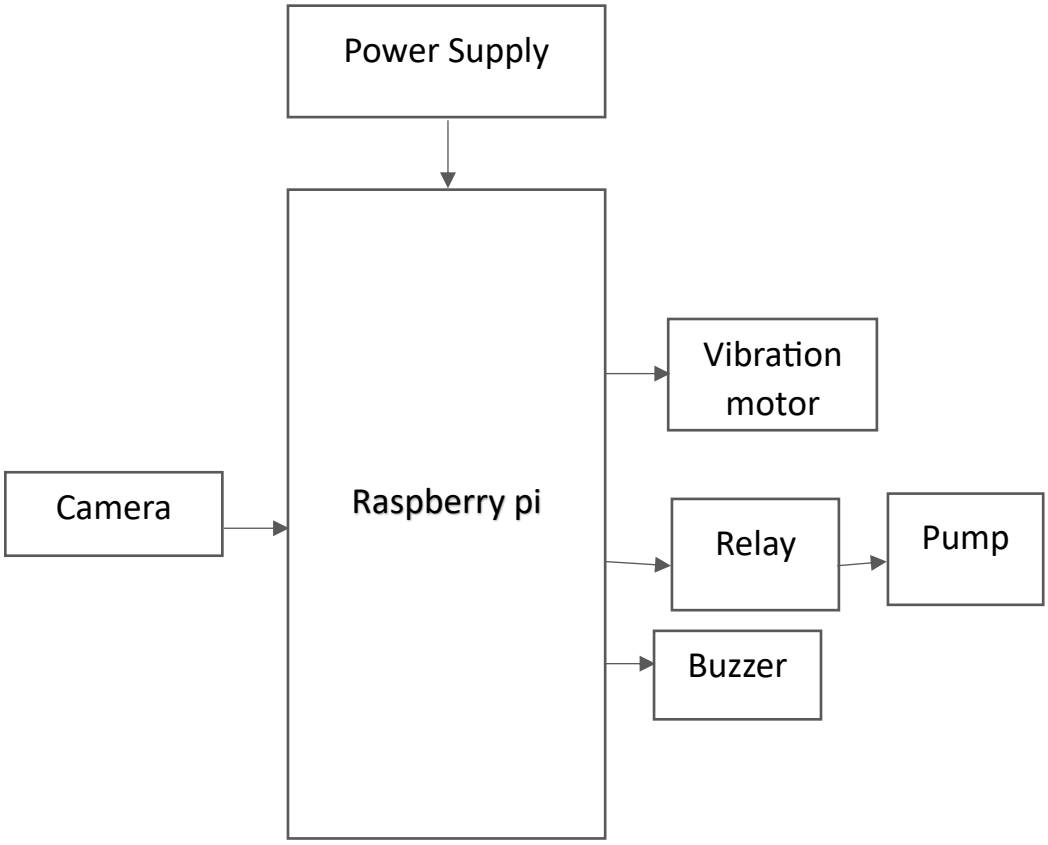

Figure 4.3: Block diagram of proposed system

4.2.2 Advantages

1. Enhanced Accuracy
2. Real-time Monitoring
3. Minimized False Alarms
4. Immediate Alert System
5. Preventative Safety Measure

4.2.3 Applications

- **Transportation Safety Enhancement:** The system can be implemented in various modes of transportation, such as cars, buses, and trucks, to detect driver drowsiness and prevent potential

accidents due to fatigue. It ensures the safety of passengers and other road users by alerting the driver when signs of drowsiness are detected.

- **Logistics and Delivery Services:** Delivery drivers often operate for long hours, increasing the risk of fatigue-related accidents. Integrating the drowsiness detection system in delivery vehicles helps fleet managers monitor driver alertness and take proactive measures to prevent accidents, ensuring timely and safe deliveries.
- **Fleet Management Solutions:** Companies with large vehicle fleets can deploy this technology as part of their fleet management systems. It enables real-time monitoring of driver fatigue across multiple vehicles, allowing managers to intervene when necessary and maintain operational efficiency.
- **Public Transportation Systems:** Public transportation vehicles, such as buses and trains, can benefit from the drowsiness detection system to ensure passenger safety. By alerting the driver to take necessary breaks or actions when drowsiness is detected, it enhances the overall safety and reliability of public transportation services.
- **Commercial Aviation:** Pilots face long flight hours and irregular schedules, making them susceptible to fatigue. Implementing the drowsiness detection system in aircraft cockpits can help prevent fatigue-related errors and improve aviation safety by ensuring pilots remain alert during flights.
- **Maritime Industry:** Similar to aviation, crew members on ships and vessels are often required to work long hours with minimal rest. Integrating the drowsiness detection system in maritime operations can mitigate the risk of accidents at sea by alerting crew members to take necessary breaks and maintain vigilance.
- **Mining and Construction:** In high-risk industries like mining and construction, where heavy machinery is operated, ensuring operator alertness is crucial for preventing accidents and injuries. This system can be employed in heavy equipment vehicles to detect drowsiness and prompt operators to rest or switch tasks when necessary.
- **Emergency Response Vehicles:** Emergency responders, such as paramedics and firefighters, often work extended shifts and respond to emergencies at all hours. Equipping emergency vehicles with the drowsiness detection system helps ensure the well-being of responders and maintains their readiness to handle critical situations.

- **Military Applications:** Military personnel, especially those deployed in combat zones or on long missions, face significant fatigue risks. Integrating the drowsiness detection system in military vehicles and aircraft enhances operational safety and mission effectiveness by preventing fatigue-related errors.
- **Personal Vehicles:** Beyond commercial and industrial applications, the system can also be installed in personal vehicles to promote safe driving practices among individual drivers. It serves as a valuable tool for preventing accidents caused by driver fatigue, particularly during long-distance travel or late-night journeys.

4.3 Hardware Requirements

4.3.1 Raspberry Pi:

Raspberry Pi is a credit-card sized computer manufactured and designed in the United Kingdom by the Raspberry Pi foundation with the intention of teaching basic computer science to school students and every other person interested in computer hardware, programming and DIY-Do-it Yourself projects.

The Raspberry Pi is manufactured in three board configurations through licensed manufacturing deals with Newark element 14(Premier Farnell), RS Components and Ego man. These companies sell the Raspberry Pi online. Ego man produces a version for distribution solely in China and Taiwan, which can be distinguished from other Pi's by their red colouring and lack of FCC/CE marks. The hardware is the same across all manufacturers.

The Raspberry Pi has a Broadcom BCM2835 system on a chip (SoC), which includes an ARM1176JZF-S 700 MHz processor, Video Core IV GPU and was originally shipped with 256 megabytes of RAM, later upgraded (Model B & Model B+) to 512 MB. It does not include a built-in hard disk or solid-state drive, but it uses an SD card for booting and persistent storage, with the Model B+ using a MicroSD.

The Foundation provides Debian and Arch Linux ARM distributions for download. Tools are available for Python as the main programming language, with support for BBC BASIC (via the RISC OS image or the Brandy Basic clone for Linux), C, Java and Perl.

Figure 4.4: Raspberry pi

4.3.2 Hardware Layout:

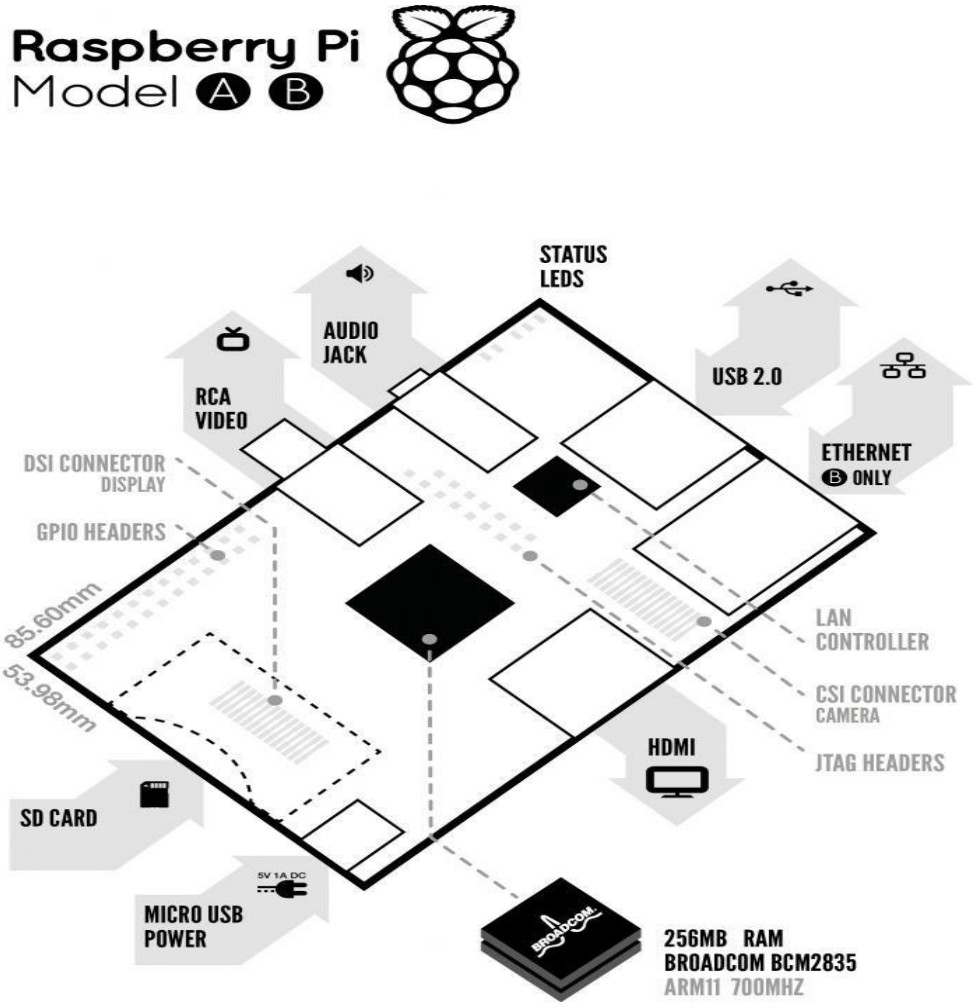

Figure 4.5: Block Diagram of Raspberry Pi

4.3.3 Description of the Components on the Raspberry pi:

1) Processor/SoC (System on Chip):

The Raspberry Pi has a Broadcom BCM2835 System on Chip module. It has a ARM1176JZF-S processor. The Broadcom SoC used in the Raspberry Pi is equivalent to a chip used in an old smartphone. While operating at 700 MHz by default, the Raspberry Pi provides a real-world performance roughly equivalent to the 0.041GFLOPS. The Raspberry Pi chip operating at 700 MHz by default, will not become hot enough to need a heatsink or special cooling.

2) Power source:

The Pi is a device which consumes 700mA or 3W or power. It is powered by a Micro USB charger or the GPIO header. Any good smartphone charger will do the work of powering the Pi.

3) SD Card:

The Raspberry Pi does not have any onboard storage available. The operating system is loaded on a SD card which is inserted on the SD card slot on the Raspberry Pi. The operating system can be loaded on the card using a card reader on any computer.

4) GPIO:

General-purpose input/output (GPIO) is a generic pin on an integrated circuit whose behaviour, including whether it is an input or output pin, can be controlled by the user at run time. GPIO pins have no special purpose defined, and go unused by default. The idea is that sometimes the system designer building a full system that uses the chip might find it useful to have a handful of additional digital control lines, and having these available from the chip can save the hassle of having to arrange additional circuitry.

The production Raspberry Pi board has a 26-pin 2.54 mm expansion header, marked as P1, arranged in a 2x13 strip. They provide 8 GPIO pins plus access to I^2C, SPI, UART), as well as +3.3 V, +5 V and GND supply lines. Pin one is the pin in the first column and on the bottom row.

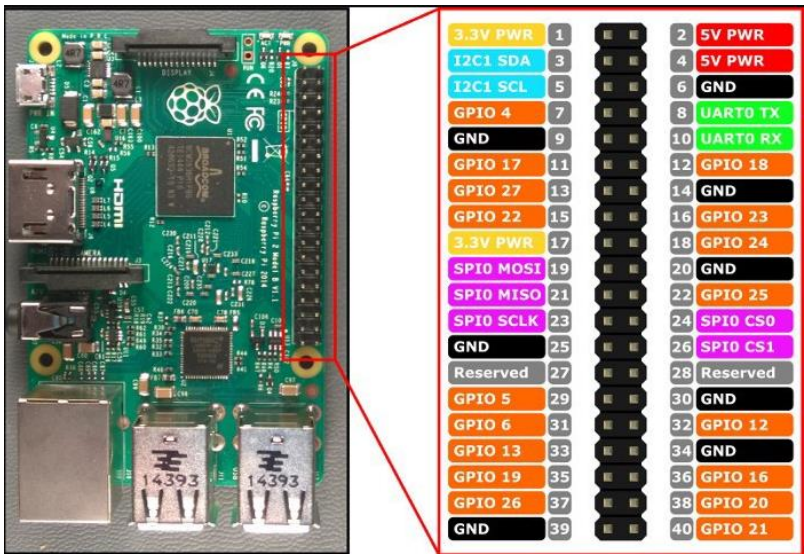

Figure 4.6: Raspberry pi pin description

5) DSI connector:

The Display Serial Interface (DSI) is a specification by the Mobile Industry Processor Interface (MIPI) Alliance aimed at reducing the cost of display controllers in a mobile device. It is commonly targeted at LCD and similar display technologies. It defines a serial bus and a communication protocol between the host and the device. A DSI compatible LCD screen can be connected through the DSI connector, although it may require additional drivers to drive the display.

6) RCA Video:

RCA Video outputs (PAL and NTSC) are available on all models of Raspberry Pi. Any television or screen with an RCA jack can be connected with the RPi.

Figure 4.7: RCA Video Connector

7) Audio Jack:

A standard 3.5 mm TRS connector is available on the RPi for stereo audio output. Any headphone or 3.5mm audio cable can be connected directly. Although this jack cannot be used for taking audio input, USB mics or USB sound cards can be used.

8) Status LEDs:

There are 5 status LEDs on the RPi that show the status of various activities. They are “OK” ,"ACT" „“POWER” (PWR),Full Duplex ("FDX"),“LNK” (Link/Activity), “10M/100” which are shown in figure below.

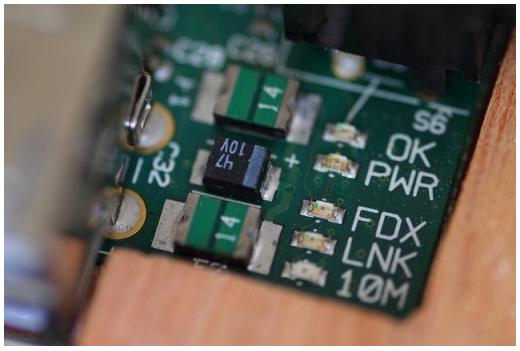

Figure 4.8: Status LEDs

9) USB 2.0 Port:

USB 2.0 ports are the means to connect accessories such as mouse or keyboard to the Raspberry Pi. There is 1 port on Model A, 2 on Model B and 4 on Mo del B+. The number of ports can be increased by using an external powered USB hub which is available as a standard Pi accessory.

10) Ethernet:

Ethernet port is available on Model B and B+. It can be connected to a network or internet using a standard LAN cable on the Ethernet port. The Ethernet ports are controlled by Microchip LAN9512 LAN controller chip.

11) CSI connector:

CSI – Camera Serial Interface is a serial interface designed by MIPI (Mobile Industry Processor Interface) alliance aimed at interfacing digital cameras with a mobile processor. The RPi

foundation provides a camera specially made for the Pi which can be connected with the Pi using the CSI connector.

12) JTAG headers:

JTAG is an acronym for ‘Joint Test Action Group', an organization that started back in the mid 1980's to address test point access issues on PCB with surface mount devices. The organization devised a method of access to device pins via a serial port that became known as the TAP (Test Access Port). In 1990 the method became a recognized international standard (IEEE Std 1149.1). Many thousands of devices now include this standardized port as a feature to allow test and design engineers to access pins.

13) HDMI:

HDMI –High-Definition Multimedia Interface

HDMI 1.3 a type A port is provided on the RPi to connect with HDMI screens.

4.3.4 Specifications:

	Model A	**Model B**	**Model B+**
Target price:	US$25	US$35	
SoC:	Broadcom BCM2835 (CPU, GPU, DSP, SDRAM, and single USB port)		
CPU:	700 MHz ARM1176JZF-S core (ARM11 family, ARMv6 instruction set)		
GPU:	Broadcom Video Core IV @ 250 MHz		
Memory (SDRAM):	256 MB GPU)		512 MB (shared with GPU)

USB 2.0 ports:	1 (direct from BCM2835 chip)	2 (via the on-board 3- port USB hub)	4 (via the on-board 5- port USB hub)
Video input:	15-pin MIPI camera interface (CSI) connector, used with the Raspberry Pi Camera Addon.		
Video outputs:	Composite RCA (PAL and NTSC) –in model B+ via 4-pole 3.5 mm jack, HDMI (rev 1.3 & 1.4), raw LCD Panels via DS		
Audio outputs:	3.5 mm jack, HDMI, and, as of revision 2 boards, I²S audio (also potentially for audio input)		
Onboard storage:	SD / MMC / SDIO card slot (3.3 V card power support only)	MicroSD	
Onboard network:	None	10/100 Mbit/s Ethernet (8P8C) USB adapter on the third/fifth port of the USB hub	
Low-level peripherals:	8× GPIO, UART, I²C bus, SPI bus with two chip selects, I²S audio +3.3 V, +5 V, ground	17× GPIO	

Power ratings:	300 mA (1.5 W)	700 mA (3.5 W)	600 mA (3.0 W)
Power source:	5 V via Micro USB or GPIO header		
Size:	85.60 mm × 56 mm (3.370 in × 2.205 in) – not including protruding Connectors		
Weight:	45 g (1.6 oz)		

Table 4.1: Specifications

4.3.5 Brief Description of System on Chip (SOC):

Since smartphones and tablets are basically smaller computers, they require pretty much the same components we see in desktops and laptops in order to offer us all the amazing things they can do (apps, music and video playing, 3D gaming support, advanced wireless features, etc). But smartphones and tablets do not offer the same amount of internal space as desktops and laptops for the various components needed such as the logic board, the processor, the RAM, the graphics card, and others. That means these internal parts need to be as small as possible, so that device manufacturers can use the remaining space to fit the device with a long-lasting battery life. A system on a chip or system on chip is an integrated circuit (IC) that integrates all components of a computer or other electronic system into a single chip. It may contain digital, analog, mixed-signal, and often radio-frequency functions—all on a single chip substrate. SoCs are very common in the mobile electronics market because of their low power consumption. A typical application is in the area of embedded systems.

The contrast with a microcontroller in one degree. Microcontrollers typically have under 100 KB of RAM (often just a few kilobytes) and often really are single- chip- systems, whereas the term SOC typically used more powerful processors, capable of running software such as the desktop versions of Windows and Linux, which need external memory chips (flash, RAM) to be useful, and which are used with various external peripherals. In short, for larger systems, the term system on a chip is a hyperbole, indicating technical direction more than reality: increasing chip integration to reduce manufacturing costs and to enable smaller systems. Many interesting systems are too complex to fit on just one chip built with a process optimized for just one of the system's tasks.

4.3.6 Accessories:

Raspberry Pi being a very cheap computer has attracted millions of users around the world. Thus, it has a large user base. Many enthusiasts have created accessories and peripherals for the Raspberry Pi. This ranges from USB hubs, motor controllers to temperature sensors. There are some official accessories for the RPi as follows:

Camera – The Raspberry Pi camera board contains a 5 Mpixel sensor, and connects via a ribbon cable to the CSI connector on the Raspberry Pi. In Raspbian support can be enabled by the installing or upgrading to the latest version of the OS and then running Raspi-config and selecting the camera option.

Relay:

What is a relay?

A relay is an electromagnetic switch that is used to turn on and turn off a circuit by a low power signal, or where several circuits must be controlled by one signal.

Most of the high-end industrial application devices have relays for their effective working. Relays are simple switches which are operated both electrically and mechanically. Relays consist of an electromagnet and also a set of contacts. The switching mechanism is carried out with the help of the electromagnet. There are also other operating principles for its working. But they differ according to their applications. Most of the devices have the application of relays.

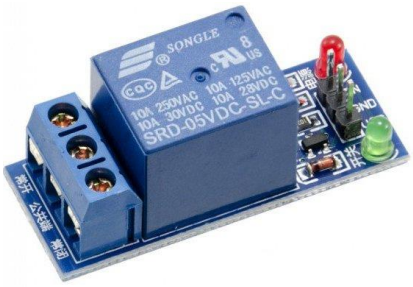

Figure 4.9: Relay

Relay Pin Diagram:

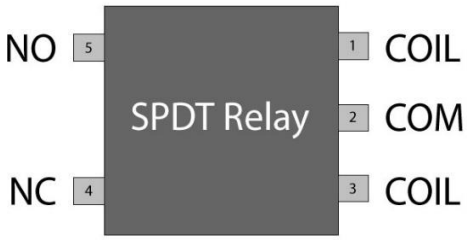

Figure 4.10: Relay pin diagram

Why is a relay used?

The main operation of a relay comes in places where only a low-power signal can be used to control a circuit. It is also used in places where only one signal can be used to control a lot of circuits. The

application of relays started during the invention of telephones. They played an important role in switching calls in telephone exchanges. They were also used in long distance telegraphy. They were used to switch the signal coming from one source to another destination. After the invention of computers, they were also used to perform Boolean and other logical operations. The high-end applications of relays require high power to be driven by electric motors and so on. Such relays are called contactors.

Relay Design:

- There are only four main parts in a relay. They are
- Electromagnet
- Movable Armature
- Switch point contacts
- Spring

The figures given below show the actual design of a simple relay.

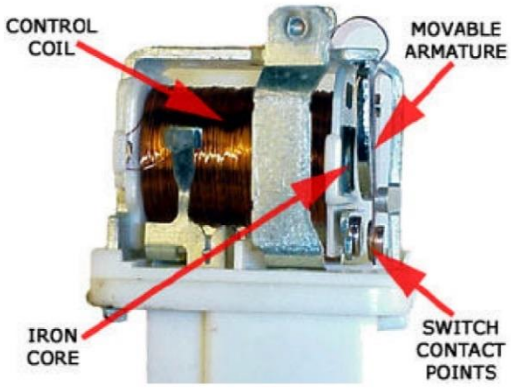

Figure 4.11: Relay Construction

It is an electro-magnetic relay with a wire coil, surrounded by an iron core. A path of very low reluctance for the magnetic flux is provided for the movable armature and also the switch point contacts.

The movable armature is connected to the yoke which is mechanically connected to the switch point contacts. These parts are safely held with the help of a spring. The spring is used so as to produce an air gap in the circuit when the relay becomes de-energized.

How relay works?

The relay function can be better understood by explaining the following diagram given below.

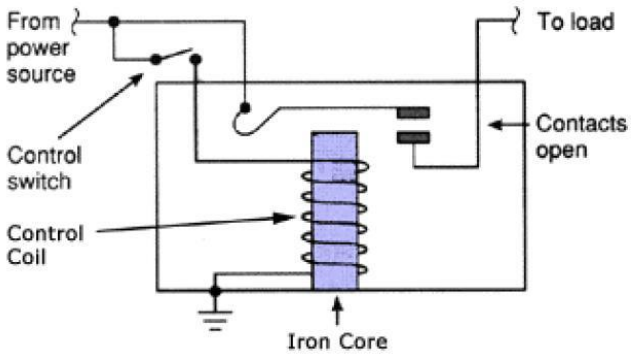

Figure 4.12: Relay Design

The diagram shows an inner section diagram of a relay. An iron core is surrounded by a control coil. As shown, the power source is given to the electromagnet through a control switch and through contacts to the load. When current starts flowing through the control coil, the electromagnet starts energizing and thus intensifies the magnetic field. Thus, the upper contact arm starts to be attracted to the lower fixed arm and thus closes the contacts causing a short circuit for the power to the load. On the other hand, if the relay was already de-energized when the contacts were closed, then the contact move oppositely and make an open circuit.

As soon as the coil current is off, the movable armature will be returned by a force back to its initial position. This force will be almost equal to half the strength of the magnetic force. This force is mainly provided by two factors. They are the spring and also gravity.

Relays are mainly made for two basic operations. One is low voltage application and the other is high voltage. For low voltage applications, more preference will be given to reduce the noise of the whole circuit. For high voltage applications, they are mainly designed to reduce a phenomenon called arcing.

Relay Basics:

The basics for all the relays are the same. Take a look at a 4-pin relay shown below. There are two colors shown. The green color represents the control circuit and the red color represents the load circuit. A small

control coil is connected onto the control circuit. A switch is connected to the load. This switch is controlled by the coil in the control circuit. Now let us take the different steps that occur in a relay.

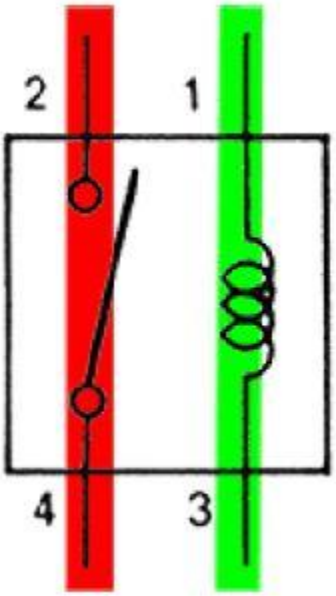

Figure 4.13: Relay operation

- **Energized Relay (ON)**

As shown in the circuit, the current flowing through the coils represented by pins 1 and 3 causes a magnetic field to be aroused. This magnetic field causes the closing of the pins 2 and 4. Thus the switch plays an important role in the relay working. As it is a part of the load circuit, it is used to control an electrical circuit that is connected to it. Thus, when the electrical relay in energized the current flow will be through the pins 2 and 4.

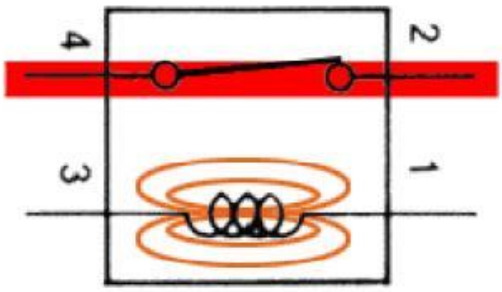

Figure 4.14: Energized Relay (ON)

- **De – Energized Relay (OFF)**

  As soon as the current flow stops through pins 1 and 3, the relay switch opens and thus the open circuit prevents the current flow through pins 2 and 4. Thus the relay becomes de-energized and thus in off position.

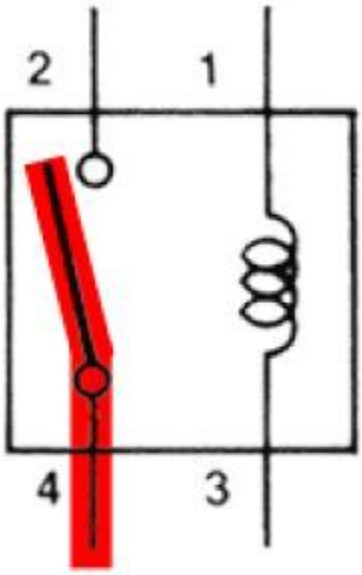

Figure 4.15: De-Energized Relay (OFF)

In simple, when a voltage is applied to pin 1, the electromagnet activates, causing a magnetic field to be developed, which goes on to close the pins 2 and 4 causing a closed circuit. When there is no voltage on pin 1, there will be no electromagnetic force and thus no magnetic field. Thus, the switches remain open.

Pole and Throw:

Relays have the exact working of a switch. So, the same concept is also applied. A relay is said to switch one or more poles. Each pole has contacts that can be thrown in mainly three ways. They are

- **Normally Open Contact (NO):** NO contact is also called a make contact. It closes the circuit when the relay is activated. It disconnects the circuit when the relay is inactive.
- **Normally Closed Contact (NC):** NC contact is also known as break contact. This is opposite to the NO contact. When the relay is activated, the circuit disconnects. When the relay is deactivated, the circuit connects.
- **Change-over (CO) / Double-throw (DT) Contacts:** This type of contacts is used to control two types of circuits. They are used to control a NO contact and also a NC contact with a common terminal. According to their type they are called by the names **break before make** and **make before break** contacts.

Relays can be used to control several circuits by just one signal. A relay switches one or more poles, each of whose contacts can be thrown by energizing the coil.

Relays are also named with designations like

- **Single Pole Single Throw (SPST)**: The SPST relay has a total of four terminals. Out of these two terminals can be connected or disconnected. The other two terminals are needed for the coil to be connected.
- **Single Pole Double Throw (SPDT)**: The SPDT relay has a total of five terminals. Out of these two are the coil terminals. A common terminal is also included which connects to either of two others.
- **Double Pole Single Throw (DPST):** The DPST relay has a total of six terminals. These terminals are further divided into two pairs. Thus, they can act as two SPST which are actuated by a single coil. Out of the six terminals two of them are coil terminals.
- **Double Pole Double Throw (DPDT)**: The DPDT relay is the biggest of all. It has mainly eight relay terminals. Out of these two rows are designed to be change over terminals. They are designed to act as two SPDT relays which are actuated by a single coil.

Relay Applications

- A relay circuit is used to realize logic functions. They play a very important role in providing safety critical logic.

- Relays are used to provide time delay functions. They are used to time the delay open and delay close of contacts.
- Relays are used to control high voltage circuits with the help of low voltage signals. Similarly, they are used to control high current circuits with the help of low current signals.
- They are also used as protective relays. By this function all the faults during transmission and reception can be detected and isolated.

Application of Overload Relay

Overload relay is an electro-mechanical device that is used to safeguard motors from overloads and power failures. Overload relays are installed in motors to safeguard against sudden current spikes that may damage the motor. An overload relay switch works in characteristics with current over time and is different from circuit breakers and fuses, where a sudden trip is made to turn off the motor. The most widely used overload relay is the thermal overload relay where a bimetallic strip is used to turn off the motor. This strip is set to make contact with a contactor by bending itself with rising temperatures due to excess current flow. The contact between the strip and the contactor causes the contactor to de-energize and restricts the power to the motor, and thus turns it off.

Another type of overload motor is the electronic type which continuously watches the motor current, whereas the thermal overload relay shuts off the motor depending on the rise of temperature/heat of the strip.

All overload relays available to buy comes in different specifications, the most important of them being the current ranges and response time. Most of them are designed to automatically reset to work after the motor is turned back on.

Relay Selection

You must note some factors while selecting a particular relay. They are

- Protection Different protections like contact protection and coil protection must be noted. Contact protection helps in reducing arcing in circuits using inductors. Â Coil protection helps in reducing surge voltage produced during switching.
- Look for a standard relay with all regulatory approvals.
- Switching time Ask for high-speed switching relays if you want one.
- Ratings There are current as well as voltage ratings. The current ratings vary from a few amperes to about 3000 amperes. Â In case of voltage ratings, they vary from 300 Volt AC to 600 Volt AC. There are also high voltage relays of about 15,000 Volts.
- Type of contact used whether it is a NC or NO or closed contact.
- Select Make before Break or Break before Make contacts wisely.
- Isolation between coil circuit and contacts

DC Water Pump:

DC powered pumps use direct current from motor, battery, or solar power to move fluid in a variety of ways. Motorized pumps typically operate on 6, 12, 24, or 32 volts of DC power. Solar-powered DC pumps use photovoltaic (PV) panels with solar cells that produce direct current when exposed to sunlight.

Figure 4.16: DC Water Pump

DC Pump Classification:

1. Brush DC water pump

2. Brushless DC magnetic drive isolated water pump

3. Brushless motor DC water pump

1. **Working principle of brushed DC water pump**：

Disadvantages: As long as the motor rotates the carbon brushes, they will wear out. When the water pump runs to a certain level, the carbon brush wear gap will increase and the sound will increase. After hundreds of hours of continuous operation, the carbon brushes will not be able to change direction. Up.

Advantages: low price.

2. **The working principle of brushless motor DC water pump**：

The motor-type brushless DC pump is composed of a brushless DC motor and an impeller. The shaft of the motor is connected to the impeller.

Disadvantages: There is a gap between the stator and the rotor of the water pump. After a long time of use, the water will penetrate into the motor and the motor will easily burn out.

Advantages: The brushless DC motor has been standardized and mass-produced by specialized manufacturers, with relatively low cost and high efficiency.

Web Camera:

A **webcam** is a video camera that feeds or streams an image or video in real time to or through a computer network, such as the Internet. Webcams are typically small cameras that sit on a desk, attach to a user's monitor, or are built into the hardware. Webcams can be used during a video chat session involving two or more people, with conversations that include live audio and video.

Webcam software enables users to record a video or stream the video on the Internet. As video streaming over the Internet requires much bandwidth, such streams usually use compressed formats. The maximum resolution of a webcam is also lower than most handheld video cameras, as higher resolutions would be reduced during transmission. The lower resolution enables webcams to be relatively inexpensive compared to most video cameras, but the effect is adequate for video chat sessions

Figure 4.17: Web cam

Buzzer:

A buzzer or beeper is an audio signaling device, which may be mechanical, electromechanical, or piezoelectric. Typical uses of buzzers and beepers include alarm devices, timers and confirmation of user input such as a mouse click or keystroke. Buzzer is an integrated structure of electronic transducers, DC power supply, widely used in computers, printers, copiers, alarms, electronic toys, automotive electronic equipment, telephones, timers and other electronic products for sound devices. Active buzzer 5V Rated power can be directly connected to a continuous sound, this section dedicated sensor expansion module and the board in combination, can complete a simple circuit design, to “plug and play.

Figure 4.18: Buzzer

Buzzer Pin Configuration

Pin Number	Pin Name	Description
1	Positive	Identified by (+) symbol or longer terminal lead. Can be powered by 5V DC
2	Negative	Identified by short terminal lead. Typically connected to the ground of the circuit

Table 4.2: Buzzer pin Configuration

Buzzer Features and Specifications

- Rated Voltage: 6V DC
- Operating Voltage: 4-8V DC
- Rated current: <30mA
- Sound Type: Continuous Beep
- Resonant Frequency: ~2300 Hz
- Small and neat sealed package
- Breadboard and Perf board friendly

How to use a Buzzer

A **buzzer** is a small yet efficient component to add sound features to our project/system. It is very small and compact 2-pin structure hence can be easily used on breadboard, Perf Board and even on PCBs which makes this a widely used component in most electronic applications.

There are two types are buzzers that are commonly available. The one shown here is a simple buzzer which when powered will make a Continuous Beeeeeeppp.... sound, the other type is called a readymade buzzer which will look bulkier than this and will produce a Beep. Beep. Beep. Sound due to the internal oscillating circuit present inside it. But the one shown here is most widely used because it can be customized with help of other circuits to fit easily in our application.

This buzzer can be used by simply powering it using a DC power supply ranging from 4V to 9V. A simple 9V battery can also be used, but it is recommended to use a regulated +5V or +6V DC supply. The buzzer is normally associated with a switching circuit to turn ON or turn OFF the buzzer at required time and require interval.

Applications of Buzzer

- Alarming Circuits, where the user has to be alarmed about something
- Communication equipment's
- Automobile electronics
- Portable equipment's, due to its compact size

4.4 Software Requirements

Python:

Python is a general purpose, dynamic, high level and interpreted programming language. It supports Object Oriented programming approach to develop applications. It is simple and easy to learn and provides lots of high-level data structures. It is easy to learn yet powerful and versatile scripting language which makes it attractive for Application Development. It's syntax and dynamic typing with its interpreted nature, makes it an ideal language for scripting and rapid application development. It supports multiple programming patterns, including object oriented, imperative and functional or procedural programming styles. It is not intended to work on special area such as web programming. That is why it is known as multipurpose because it can be used with web, enterprise, 3D CAD etc. We don't need to use data types to declare variable because it is dynamically typed so we can write a=10 to assign an integer value in an integer variable. It makes the development and debugging fast because there is no compilation step included in python development and edit-test-debug cycle is very fast.

4.4.1 Python Features:

Python provides lots of features that are listed below.

1. Easy to use:

Python is easy to learn and use. It is developer-friendly and high-level programming language.

2. Expressive Language:

Python language is more expressive means that it is more understandable and readable.

3. Interpreted Language:

Python is an interpreted language i.e. interpreter executes the code line by line at a time. This makes debugging easy and thus suitable for beginners.

4. Cross-platform Language:

Python can run equally on different platforms such as Windows, Linux, Unix and Macintosh etc. So, we can say that Python is a portable language.

5. Free and Open Source:

Python language is freely available at address. The source-code is also available. Therefore, it is open source

6. Object-Oriented Language:

Python supports object-oriented language and concepts of classes and objects come into existence.

7. Extensible:

It implies that other languages such as C/C++ can be used to compile the code and thus it can be used further in our python code.

8. Large Standard Library

Python has a large and broad library and provides rich set of modules and functions for rapid application development.

9. GUI Programming Support

Graphical user interfaces can be developed using Python.

10. Integrated

It can be easily integrated with languages like C, C++, JAVA etc.

4.4.2 Python History:

- Python laid its foundation in the late 1980s.
- The implementation of Python was started in the December 1989 by **Guido Van Rossum** at CWI in Netherland.
- In February 1991, van Rossum published the code (labelled version 0.9.0) to alt.sources.
- In 1994, Python 1.0 was released with new features like: lambda, map, filter, and reduce.
- Python 2.0 added new features like: list comprehensions, garbage collection system.
- On December 3, 2008, Python 3.0 (also called "Py3K") was released. It was designed to rectify fundamental flaw of the language.
- *ABC programming language* is said to be the predecessor of Python language which was capable of Exception Handling and interfacing with Amoeba Operating System.
- Python is influenced by following programming languages:
    - ABC language.
    - Modula-3

4.4.3 Python Version

Python programming language is being updated regularly with new features and supports. There are lots of updates in python versions, started from 1994 to current release.

Python Version used	Released date
Python 3.8	October 14, 2019

Python is a widely-used general-purpose, high-level programming language. It was initially designed by Guido van Rossum in 1991 and developed by Python Software Foundation. It was mainly developed for emphasis on code readability, and its syntax allows programmers to express concepts in fewer lines of code. Python is a programming language that lets you work quickly and integrate systems more efficiently.

Python 3.8 contains many small improvements over the earlier versions. This article outlines the most significant additions and changes in python 3.8

4.4.4 Python Applications Area

Python is known for its general-purpose nature that makes it applicable in almost each domain of software development. Python as a whole can be used in any sphere of development.

Here, we are specifying applications areas where python can be applied.

1. Web Applications

We can use Python to develop web applications. It provides libraries to handle internet protocols such as HTML and XML, JSON, Email processing, request, beautifulSoup, Feedparser etc. It also provides Frameworks such as Django, Pyramid, Flask etc to design and develop web-based applications. Some important developments are: PythonWikiEngines, Pocoo, PythonBlogSoftware etc.

2. Desktop GUI Applications

Python provides Tk GUI library to develop user interface in python-based application. Some other useful toolkits wxWidgets, Kivy, pyqt that are useable on several platforms. The Kivy is popular for writing multitouch applications.

3. Software Development

Python is helpful for software development process. It works as a support language and can be used for build control and management, testing etc.

4. Scientific and Numeric

Python is popular and widely used in scientific and numeric computing. Some useful library and package are SciPy, Pandas, IPython etc. SciPy is group of packages of engineering, science and mathematics.

5. Business Applications

Python is used to build Business applications like ERP and e-commerce systems. Tryton is a high-level application platform

6. Console Based Application

We can use Python to develop console-based applications. For example: **IPython**.

7. Audio or Video based Applications

Python is awesome to perform multiple tasks and can be used to develop multimedia applications. Some of real applications are: Tim Player, cplay etc

8. 3D CAD Applications

To create CAD application Fandango is a real application which provides full features of CAD

9. Enterprise Applications

Python can be used to create applications which can be used within an Enterprise or an Organization. Some real time applications are: OpenErp, Tryton, Picalo etc.

10. Applications for Images

Using Python several application can be developed for image. Applications developed are: VPython, Gogh, imgSeek etc. There are several such applications which can be developed using Python.

CHAPTER - 5

CIRCUIT DESCRIPTION AND RESULTS

5.1 Circuit Description

Designing a drowsiness detection and alerting system with integrated vibrations and a watering system involves a combination of sensor technology, signal processing, and actuation mechanisms to monitor driver fatigue and intervene when necessary. The circuitry for such a system typically includes various sensors, a microcontroller unit (MCU), actuators, and power management components.

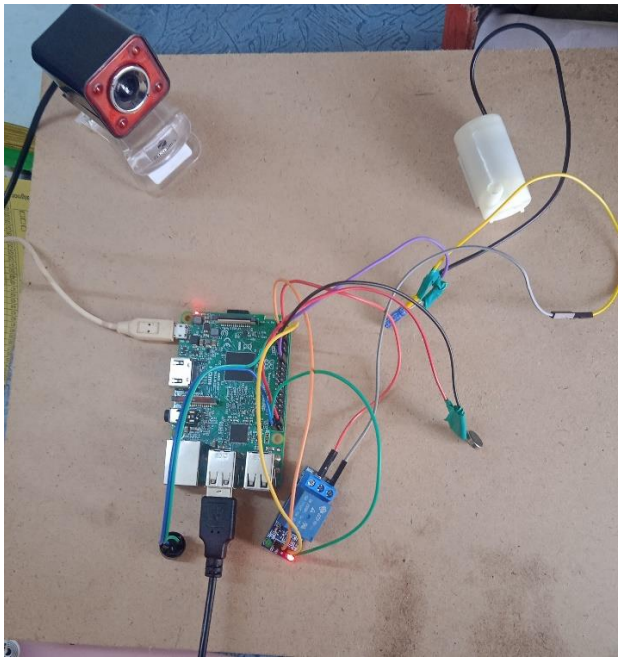

Figure 5.1: Hardware Connections

Connections:

Raspberry pi	Components
Pin 1	Relay VCC
Pin 4	Relay NC
Pin 37	Relay IN
Pin 6	Relay Ground
Pin 9	Vibration Ground
Pin 14	Motor Ground

Pin 5	Buzzer Positive
Pin 38	Vibration Positive
Pin 39	Buzzer Ground
Motor Positive	Relay NC

Table 5.1: Connections

5.2 Results/Outputs

The driver drowsiness detection project successfully developed a real-time monitoring system using a Raspberry Pi and a camera to detect signs of drowsiness in drivers.

Figure 5.2: Drowsiness Detection using EAR

According to the threshold value given in the code, if the value of EAR exceeds, we get the alert as “DROWSINESS ALERT” as shown in the figure 5.2 and actuators like Buzzer, DC water pump, Vibrator motor will be activated with help of relays.

Figure 5.3: Yawn Detection using MAR

According to the threshold value given in the code, if the value of MAR exceeds, we get the alert as "YAWN ALERT" as shown in the figure 5.3 and actuators function is same as in previous case.

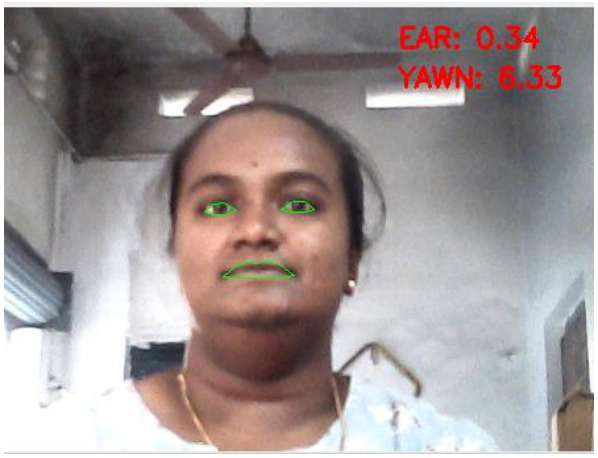

Figure 5.4: Driver without Drowsiness

As the values of EAR and MAR does not exceed their respective thresholds, all the actuators will be in OFF state.

CHAPTER-6

CONCLUSION & FUTURE SCOPE

6.1 Conclusion

In conclusion, the development of the Driver Drowsiness Detection System using Raspberry Pi marks a significant step forward in ensuring road safety. Through the integration of advanced image processing techniques and sensor technologies, the system provides an effective solution for detecting driver fatigue and preventing potential accidents.

The successful implementation of this project underscores its potential applications in various domains beyond just road safety. With further refinement and optimization, the system can be deployed in commercial vehicles, public transportation, and even personal automobiles. This widespread adoption can significantly reduce the number of accidents caused by drowsy driving, ultimately saving lives and preventing injuries.

Overall, the Driver Drowsiness Detection System represents a valuable contribution to the field of automotive safety technology. By leveraging the capabilities of Raspberry Pi and sophisticated algorithms, it offers a practical and reliable solution for addressing the critical issue of driver fatigue on the roads. As advancements continue to be made in this area, we can look forward to even more sophisticated systems that enhance road safety and improve the overall driving experience .

6.2 Future Scope

1.Integration with Autonomous Vehicles: As self-driving vehicles become more prevalent, integrating drowsiness detection systems could be crucial for ensuring passenger safety. Vibrations and watering systems could alert the driver to regain focus.

2. Wearable Technology: The technology could be integrated into wearable devices, such as smartwatches or fitness trackers, to monitor the wearer's alertness and provide timely alerts.

3. Workplace Safety: Industries like transportation, construction, and manufacturing could use these systems to prevent accidents caused by drowsy workers.

4. Health Monitoring: Beyond just detecting drowsiness, these systems could be part of broader health monitoring platforms, providing insights into sleep quality and fatigue levels.

5. Personalized Feedback: Advanced Al algorithms could provide personalized feedback and recommendations to improve sleep quality and reduce fatigue.

7. Data Analytics and Predictive Analysis: Collected data could be used for analytics and predictive analysis to identify patterns, trends, and potential risks related to drowsiness and fatigue.

8. Global Health and Safety Standards: As the technology matures and proves its effectiveness, it could be adopted as a standard safety feature in vehicles, workplaces, and other environments to prevent accidents and improve overall safety.

REFERENCES

[1]. A. Sahayadhas, K. Sundaraj, and M. Murugappan, (2012). “Detecting driver drowsiness based on sensors: A review,”

Sensors (Switzerland), vol. 12, no. 12, pp. 16937–16953, doi:10.3390/s121216937

[2]. Fischer, Frida Marina, Claudia Roberta, De Castro Moreno, Flavio Notarnicola, and Fernando M Louzada, (2000).

"Implementation of 12-hour shifts in a Brazilian Petrochemical Plant: Impact on sleep and alertness." 17, 521-537,

doi:10.1081/CBI-100101062

[3]. Dawson, Drew, and Kirsty McCulloch, (2005). "Managing fatigue: It's about sleep." 365, 80,

doi:10.1016/j.smrv.2005.03.002

[4]. Smolensky, Michael H, Lee Di, Maurice M Ohayon, and Pierre Philip, (2011) "Sleep disorders, medical conditions,

and road accident risk." Accident Analysis & Prevention, 533-548, doi:10.1016/j.aap.2009.12.004

[5]. Pack, Andrew M, Eric Rodgman, Allan I Pacic, David F Dinges, and C William Schwab, (1995) "Characteristicsthe driver of crashes attributed having fallen asleep to” 27, 769-75, doi:10.1016/0001-4575(95)00034-8

[6]. Williamson, Ann, David A Lombardi, Simon Folkard, Jane Stutts, Theodore K Courtney, and Jennie L Connor, (2011)
"The link between fatigue and safety. " Accident Analysis & Prevention 43, 498-515, doi:10.1016/j.aap.2009.11.011

[7]. Saeed, Seyed, Hashemi Nazari, Ali Moradi, and Khaled Rahmani, (2012) “Original Article A Systematic Review of the Effect of Various Interventions on Reducing Fatigue and Sleepiness While Driving,” Chinese Journal of Traumatology. - English Ed., vol. 20, no. 5, pp. 249–258, doi:10.1016/j.cjtee.2017.03.005

[8]. Sedighi, Zahra, Mohammad Ali, Alamdar Yazdi, Lora A Cavuoto, and Fadel M Megahed, (2017) “A data-driven approach to modelling physical fatigue in the workplace using wearable sensors,” Applied Ergonomics., vol. 65, pp.
515–529, doi:10.1016/j.apergo.2017.02.001

[9]. Sałapatek, Damian, Jacek Dybala, Paweł Czapski, and Paweł Skalski, (2017) ‘Proceedings of the Institute of Vehicles 3(112)’, 41–48

[10]. Auberlet, Jean-michel, Florence Rosey, Sébastien Aubin, Patrice Briand, Marie-pierre Pacaux, and Patrick Plainchault, (2012)“The impact of perceptual treatments on driver’s behaviour: From driving simulator studies to field tests – First results. vol. 45, no pp. 91–98, doi:10.1016/j.aap.2011.11.020

[11] Ansari, S., Du, H., Naghdy, F. & Stirling, D., 2022. Automatically detects driver-perceived fatigue based on changes in upper body posture. Expert system with applications, p.117568

[12] Biswal, A.K., Singh, D., Pattanayak, B.K., Samanta, D. and Yang, M.H., 2021. IoT-based intelligent warning system to detect drowsy drivers. Wireless communications and mobile computing, 2021

[13] Chikezie, U.M. and Nwazor, N.O., 2021.A drowsiness detection system using computer vision and IoT, 8(12).

[14] C. Yang, Z. Yang, W. Li and J. See, "FatigueView: A Multi-Camera Video Dataset for Vision-Based Drowsiness Detection," in IEEE Transactions on Intelligent Transportation Systems, vol. 24, no. 1, pp. 233-246, Jan. 2023, doi: 10.1109/TITS.2022.3216017.

[15] Fouad, I.A., 2022. A robust and efficient EEG-based drowsiness detection system using different machine learning algorithms. Ain Shams Engineering Journal, p.101895.

CHAPTER-7

APPENDIX

7.1 Implemented Code:

```
import RPi.GPIO as GPIO
from imutils.video import VideoStream
from imutils import face_utils
from threading import Thread
import numpy as np
import argparse
import imutils
import time
import dlib
import cv2
import os
    GPIO.setwarnings(False)
    GPIO.setmode(GPIO.BCM)
    buzz=3
    pump=26
    vib=20
    GPIO.setup(buzz,GPIO.OUT)
    GPIO.setup(pump,GPIO.OUT)
    GPIO.setup(vib,GPIO.OUT)
    GPIO.output(buzz,GPIO.LOW)
    GPIO.output(vib,GPIO.HIGH)
    GPIO.output(pump,GPIO.HIGH)
    def euclidean_dist(ptA, ptB):
        return np.linalg.norm(ptA - ptB)
    def eye_aspect_ratio(eye):
        A = euclidean_dist(eye[1], eye[5])
        B = euclidean_dist(eye[2], eye[4])

        C = euclidean_dist(eye[0], eye[3])
```

```
        ear = (A + B) / (2.0 * C)
        return ear
    def final_ear(shape):
        (lStart, lEnd) = face_utils.FACIAL_LANDMARKS_IDXS["left_eye"]
        (rStart, rEnd) = face_utils.FACIAL_LANDMARKS_IDXS["right_eye"]
        leftEye = shape[lStart:lEnd]
        rightEye = shape[rStart:rEnd]
        leftEAR = eye_aspect_ratio(leftEye)
        rightEAR = eye_aspect_ratio(rightEye)
        ear = (leftEAR + rightEAR) / 2.0
        return (ear, leftEye, rightEye)
    def lip_distance(shape):
        top_lip = shape[50:53]
        top_lip = np.concatenate((top_lip, shape[61:64]))

        low_lip = shape[56:59]
        low_lip = np.concatenate((low_lip, shape[65:68]))

        top_mean = np.mean(top_lip, axis=0)
        low_mean = np.mean(low_lip, axis=0)
        distance = abs(top_mean[1] - low_mean[1])
        return distance
    EYE_AR_THRESH = 0.3
    EYE_AR_CONSEC_FRAMES = 15
    YAWN_THRESH = 30
    COUNTER = 0
    print("-> Loading the predictor and detector...")
       #detector = dlib.get_frontal_face_detector()
    detector =
cv2.CascadeClassifier("/home/pi/Desktop/DROWSINESS/haarcascade_frontalface_default.xml"
)    #Faster but less accurate
```

```
        predictor =
dlib.shape_predictor('/home/pi/Desktop/DROWSINESS/shape_predictor_68_face_landmarks.da
t')
        print("-> Starting Video Stream")
        vs = VideoStream(src=0).start()
           #vs= VideoStream(usePiCamera=True).start()       //For Raspberry Pi
        time.sleep(1.0)
        while True:
              frame = vs.read()
              frame = imutils.resize(frame, width=450)
              gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
              rects = detector.detectMultiScale(gray, scaleFactor=1.1,
                        minNeighbors=5, minSize=(30, 30),
                        flags=cv2.CASCADE_SCALE_IMAGE)
              for (x, y, w, h) in rects:
                 rect = dlib.rectangle(int(x), int(y), int(x + w),int(y + h))
                 shape = predictor(gray, rect)
                 shape = face_utils.shape_to_np(shape)
                 eye = final_ear(shape)
                 ear = eye[0]
                 leftEye = eye [1]
                 rightEye = eye[2]
                 distance = lip_distance(shape)
                 leftEyeHull = cv2.convexHull(leftEye)
                 rightEyeHull = cv2.convexHull(rightEye)
                 cv2.drawContours(frame, [leftEyeHull], -1, (0, 255, 0), 1)
                 cv2.drawContours(frame, [rightEyeHull], -1, (0, 255, 0), 1)
                 lip = shape[48:60]
                 cv2.drawContours(frame, [lip], -1, (0, 255, 0), 1)
                 cv2.putText(frame, "EAR: {:.2f}".format(ear), (300, 30),
                           cv2.FONT_HERSHEY_SIMPLEX, 0.7, (0, 0, 255), 2)
```

```
                cv2.putText(frame, "YAWN: {:.2f}".format(distance), (300, 60),
                        cv2.FONT_HERSHEY_SIMPLEX, 0.7, (0, 0, 255), 2)
                if ear < EYE_AR_THRESH:
                   COUNTER += 1
                   if COUNTER >= EYE_AR_CONSEC_FRAMES or distance >
YAWN_THRESH :
                      cv2.putText(frame, "DROWSINESS ALERT!", (10, 30),
                            cv2.FONT_HERSHEY_SIMPLEX, 0.7, (0, 0, 255), 2)
                      cv2.putText(frame, "Yawn Alert", (10, 60),
                            cv2.FONT_HERSHEY_SIMPLEX, 0.7, (0, 0, 255), 2)
                      GPIO.output(buzz,GPIO.HIGH)
                      GPIO.output(vib,GPIO.LOW)
                      GPIO.output(pump,GPIO.LOW)
                      time.sleep(3.0)
                      GPIO.output(buzz,GPIO.LOW)
                      GPIO.output(vib,GPIO.HIGH)
                      GPIO.output(pump,GPIO.HIGH)
                else:
                   COUNTER = 0
                   GPIO.output(buzz,GPIO.LOW)
             cv2.imshow("Frame", frame)
             key = cv2.waitKey(1) & 0xFF
             if key == ord("q"):
                break
          cv2.destroyAllWindows()
          vs.stop()
```

7.2 Source Code explanation:

Firstly, the code initializes the GPIO (General Purpose Input/Output) pins for the Raspberry Pi to control the buzzer, water pump, and vibration motor. This is done to set up the hardware alerts. The buzzer is connected to GPIO pin 3, the water pump to pin 26, and the vibration motor to pin 20. The default states for these pins are set: the buzzer is off, the vibration motor is off, and the water pump is off.

Next, the script defines several helper functions:

- **euclidean_dist** calculates the Euclidean distance between two points.
- **eye_aspect_ratio** computes the eye aspect ratio using the detected eye landmarks.
- **final_ear** calculates the average eye aspect ratio of both eyes.
- **lip_distance** measures the lip distance to detect if the driver is yawning.

Following the helper functions, the script sets threshold values for drowsiness and yawning detection. An eye aspect ratio threshold (EYE_AR_THRESH) of 0.3 and a yawning threshold (YAWN_THRESH) of 30 are defined. Additionally, the script sets the number of consecutive frames required to trigger a drowsiness alert (EYE_AR_CONSEC_FRAMES) to 15.

The next part of the code loads the face detector and facial landmark predictor. The face detector is initialized using a Haar cascade file, and the facial landmark predictor uses a dlib shape predictor file. These files are stored on the Raspberry Pi.

After initializing the video stream from the Raspberry Pi camera, the main loop of the program begins. Inside this loop:

- The script reads a frame from the video stream.
- The frame is converted to grayscale, and faces are detected using the previously loaded face detector.
- For each detected face, the script calculates the eye aspect ratio and lip distance.
- The eye aspect ratio and lip distance are then displayed on the frame.
- If the calculated eye aspect ratio is below the defined threshold or the lip distance exceeds the yawning threshold, a drowsiness alert is triggered. This alert activates the buzzer, turns off the vibration motor, and turns off the water pump for 3 seconds to alert the driver.
- If the driver is alert, the buzzer is turned off, and the alert counter is reset.

Finally, the video frame with the annotations is displayed, and the program can be terminated by pressing the 'q' key. The video stream is then stopped, and all OpenCV windows are closed.

Here's how it works:

- **cv2.waitKey(1)** waits for a key event for 1 millisecond.
- **& 0xFF** is a mask to extract the ASCII value from the key event.
- **if key == ord("q"):** checks if the pressed key's ASCII value corresponds to the letter 'q'.