VEMANA INSTITUTE OF TECHNOLOGY

(Affiliated to VTU & approved by AICTE)
#1, Mahayogi Vemana Road, Koramangala, Bengaluru-34

Department of Electronics and Communication Engineering

CERTIFICATE

Certified that the Project work phase-1 entitled **"IOT BASED SMART VEHICLE SAFETY AND ACCIDENT MONITORING SYSTEM"** is carried out by Mithun R (1VI20EC045), N Nikhilesh (1VI20EC048), Nivya D (1VI20EC050), Vaishnavi D S (1VI20EC076) in partial fulfilment for the award of Bachelor of Engineering degree in Electronics and Communication Engineering, Visvesvaraya Technological University (VTU), Belagavi during the academic year 2023-2024.

The Project work Phase-1 report has been approved as it satisfies the academic requirements in respect of the project work prescribed for the said degree.

Guide	HoD	Principal
(Prof. Sandhya K V)	**(Dr. Parameshwara M C)**	**(Dr. Vijayasimha Reddy B G)**

	Name of the Examiners	Signature with date
1.	______________________________	______________________
2.	______________________________	______________________

VISVESVARAYA TECHNOLOGICAL UNIVERSITY

JNANA SANGAMA, Belagavi-590018, Karnataka

Project work Phase-1 Report

On

"IOT BASED SMART VEHICLE SAFETY AND ACCIDENT MONITORING SYSTEM"

Submitted in partial fulfillment of the requirement for the award of the degree of

Bachelor of Engineering

In

Electronics and Communication Engineering

Submitted by

MITHUN R	**1VI20EC045**
N NIKHILESH	**1VI20EC048**
NIVYA D	**1VI20EC050**
VAISHNAVI D S	**1VI20EC076**

Under the guidance of
Prof. SANDHYA K V
Assistant Professor

Department of Electronics & Communication Engineering
VEMANA INSTITUTE OF TECHNOLOGY
(Affiliated to VTU & Approved by AICTE)
#1, Mahayogi Vemana Road, Koramangala, Bengaluru– 560034

2022-23

ACKNOWLEDGEMENT

We sincerely thank Visvesvaraya Technology University (VTU) for providing a platform to do the project work.

We express our sincere thanks to **Dr. Vijayasimha Reddy B G, Principal**, Vemana Institute of Technology, Bengaluru, for providing necessary facilities and motivation to carry out project work successfully.

We express heartfelt gratitude and humble thanks to **Dr. Parameshwara M C, HoD,** ECE, Vemana Institute of Technology, for his constant encouragement, inspiration and help to carry out project work successfully.

We would like to express our sincere gratitude towards our internal guide **Prof. Sandhya K V, Asst. Professor** for providing encouragement and inspiration for this work.

We thank project coordinators **Dr Girish N. and Prof. Amulya** for their cooperation and support during the project work.

We are thankful to all the teaching and non-teaching staff members of Electronics & Communication Engineering department for their help and needed support rendered throughout the work.

Mithun R (1VI20EC045)
N Nikhilesh (1VI20EC048)
Nivya D (1VI20EC050)
Vaishnavi D S (1VI20EC076)

ABSTRACT

The Advanced Vehicle Safety Monitoring System presented in this project addresses the pressing issue of road safety by leveraging advanced technologies. Integrating Vibration Sensors and Accelerometers ensures precise accident detection, while an integrated camera system records and stores accident scenes both offline and online through IoT infrastructure. The addition of a Gas Sensor enhances safety by detecting and responding to gas leakages. To streamline emergency responses, the system incorporates a GPS sensor for real-time location tracking and a GSM module for automated accident reports. Intelligent vehicle interaction features, such as automatic light dimming and dipping triggered by oncoming vehicles, further enhance road safety. This comprehensive approach aims to revolutionize accident prevention and response, providing accurate and timely information for prompt emergency interventions. The proposed system not only contributes to safer roads but also sets a benchmark for innovative solutions in vehicular safety. Integrating Vibration Sensors and Accelerometers ensures precise accident detection, while an integrated camera system records and stores accident scenes both offline and online through IoT infrastructure. To streamline emergency responses, the system incorporates a GPS sensor for real-time location tracking and a GSM module for automated accident reports. Intelligent vehicle interaction features, such as automatic light dimming and dipping triggered by oncoming vehicles, further enhance road safety. This comprehensive approach aims to revolutionize accident prevention and response, providing accurate and timely information for prompt emergency interventions. The proposed system not only contributes to safer roads but also sets a benchmark for innovative solutions in vehicular safety.

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER 1

INDRODUCTION

In India, there are more than 150,000 deaths due to road accidents annually and this number continues to rise each year. These Death rates can be considerably reduced by providing medical assistance immediately. The major elements that delay the provision of medical help are lack of ambulance services, no network connectivity and negligence.

To resolve these factors, an automated response system is required. In our project we introduce the Automatic Accident Detection and Monitoring System (ADMS) that will minimize the time gap and ensure medical assistance immediately. This system will integrate a Vibration Sensor, accelerometer sensor, gas sensor, GPS sensor, and GSM module into vehicles to detect sudden changes in acceleration, vibrations, and gas leakage. It will automatically generate and transmit precise accident reports, including the exact location of the incident. The system will utilize cameras to gather visual evidence, which can be vital for insurance claims, legal proceedings, and accident reconstruction.

By using these sensors, the system will swiftly identify and respond to accidents, thereby reducing response time and potentially saving lives. By IoT technology, these systems can operate autonomously, without the need for human intervention in the initial stages of an accident. This system server will statically analyze the data received from the software and will notify or contact the client family, friends or predefined number about the accident.

The system will also incorporate Intelligent Vehicle Interaction features to enhance overall road safety. According to the law of “Indian Roads and Transport Council”, low intensity light has to be used always, however in the time of overtaking or other mandatory situations, high beam can be turned on, but most of the drivers use elevated, bright beam during night driving.

This automatic vehicle headlight dim and dip system adjust the intensity beam when finds the vehicles in opposite direction. It utilizes a Light Dependent Resistor (LDR) sensor was intended to dim the headlight of vehicles automatically to prevent the impacts of human eyes. These features will include automatic light dimming and dipping, triggered by the detection of oncoming vehicles. By intelligently adjusting vehicle lighting, the system aims to reduce glare and distractions for other drivers, thereby contributing to a safer driving environment.

1.1 AIM

To implement a comprehensive system by integrating vibration sensors, accelerometers, and a camera module into vehicles, establishing an IoT network, and enabling dynamic adjustment of head lighting based on real-time conditions to ensure optimal visibility for the driver and enhance safety during night driving.

1.2 OBJECTIVES

1. Develop and test an Accident Monitoring System using Vibration sensor, Accelerometer.
2. Integrate a Gas Sensor to detect and respond to gas leakage and integrate camera module to enhance overall safety measures.
3. To enable real-time Location Tracking using GPS sensor with an integrated GSM module for automated accident reports.
4. To incorporate Intelligent Vehicle Interaction features, such as automatic light dimming and dipping, triggered by the detection of oncoming vehicles, to enhance overall road safety.

CHAPTER 2

LITERATURE SURVEY

In this chapter, will discuss about the information found by study and research that is critical and have an important value in the contribution of the whole project. It also gives some basic knowledge or theoretical base and is used as a foundation to successfully achieve the main objectives. Most of the literature are from the related articles, journals, books and previous works of the same fields. These literatures are then compiled and used as guidance to the work.

[1] “**A smart IoT based black-box system for automobiles**”, Vanitha M, Arun kumar K, Hemamalini A, Alam Yaswanth. IEEE published in the year of 2023.

The implementation of a black box system in automobiles represents a significant step forward in addressing the alarming number of deaths resulting from transportation-related collisions, as reported by the World Health Organization. Similar to the role of flight data recorders in planes, "Black Box technology" is poised to play a vital role in motor vehicle collision cases. This innovative system, developed with a focus on safety and data collection, incorporates a postal contact and utilizes the Internet of Things for data transfer. In the event of an accidental crash, the system's sensors provide input details to an Arduino, while GSM technology enables the transmission of location coordinates to a registered mobile number, potentially saving lives. The data is stored in an SD card for future retrieval, ensuring that crucial information is preserved for analysis. Looking ahead, the integration of image processing techniques for driver video recording is being considered, demonstrating a commitment to continuous improvement in vehicle safety measures.

[2] **“IoT based automatic smart black-box system to detect accidents”,** Srinivasan, Kommalapati, Kommineni Manjunadh, K Jyothish. IEEE published in the year of 2023.

This paper explores an advanced controller system for automobile recording equipment and its functionalities. The system accurately captures driving data and reconstructs accidents using information method code, enabling swift and legal assessment of collisions. A dedicated controller continuously monitors sensor parameters within the SD card module, updating information during abnormal events. It prioritizes data recording to optimize storage space and addresses challenges in collecting long-term data for on-road vehicles, emphasizing the need for global optimization of data deletion and compression options.

[3] **“Smart Security System for Vehicles using Internet of Things (IoT)”** Mithileysh Sathiyanarayanan, Santosh Mahendra, Rajesh Babu Vasu. IEEE published in the year of 2021.

The paper's primary objective is to develop a smart vehicle security system using IoT, with the goal of transforming conventional vehicle security systems into Intelligent Anti-Theft Tracking Systems (iATTS). This system will be designed to be lightweight, cost-effective, and will utilize a range of technologies including RFID, GPS, GSM, wireless communication, cloud networking, and fuzzy algorithms. By integrating these technologies, the system will provide real-time vehicle information such as position, time, and alarms to the vehicle owner through SMS or a mobile application. This comprehensive approach aims to offer enhanced security for all types of vehicles, with a focus on ease of installation and the capability to prevent, detect, and counteract vehicle theft.

[4] **“Arduino Based Vehicle Accident Alert System Using GPS, GSM and MEMS Accelerometer”,** Pachipala Yellamma; NSNSP Chandra; Puli Sukhesh; Puligadda Shrunith; Sunkesula Siva Teja. IEEE published in the year of 2021.

The proposed system aims to reduce the number of accidents by utilizing a combination of technologies. It involves sending an alert message to the family members of the person involved in an accident using a GSM module. Additionally, a GPS modem continuously senses the location of the accident and transmits the latitude and longitude coordinates. The system incorporates a MEMS Accelerometer ADXL335 to detect changes in axes when the vehicle is impacted by another vehicle. By capturing the values of the X and Y axes, the MEMS Accelerometer can determine the type of accident that has occurred. This system integrates various technologies to provide comprehensive information to the family members of the accident victim and emergency services.

[5] “**Accident detection using Automotive Smart Black-Box based Monitoring system**”, P. Josephinshermila, S. Sharon priya, K. Malarvizhi, Ramakrishna hegde, S. Gokul Pran, B. Veerasamy. Science direct published in the year of 2023.

In this paper the demonstration of The Automotive Monitoring using Black-Box system which authenticates the driver and then periodically monitors the vehicle parameters such as temperature, speed, gas detection, the exact location of the vehicle, humidity, accident detection and store the data in the black-box and also in the cloud database and which can be monitored through the website. The stored data will be used for live monitoring of vehicles by the organizations, and accident analysis, and investigative purposes of insurance claims by the organizations. Immediate medical services can also be availed from the nearest hospital automatically in case of an accident by sensing an alert message which also includes an accurate vehicle's location. According to experimental findings, the proposed technique achieves 29.3103%, 22.70 %, 18.103% and 11.206 % higher accuracy compared to RFID, SVM, CNN and RNN Methods.

[6] **"Accident Detection & Alert System"**, Prashant Kapri, Shubham Patane, Arul Shalom A. ResearchGate published in the year 2020.

In this paper the Automatic Accident Detection and Alert System (ADAS), designed to minimize response time and ensure immediate medical assistance. The system consists of two main components: the server and the ADAS software. Utilizing various sensors in smartphones, the ADAS system can accurately identify the location of an accident and promptly send a message to the nearest ambulance. By leveraging the smartphone's built-in sensors, the ADAS client system can swiftly alert the closest medical assistance provider, enabling them to reach the accident site and provide timely support to the victim. This project has the potential to significantly reduce response times and, consequently, lower the number of fatalities resulting from accidents.

CHAPTER 3

DESIGN METHODOLOGY

3.1 BLOCK DIAGRAM

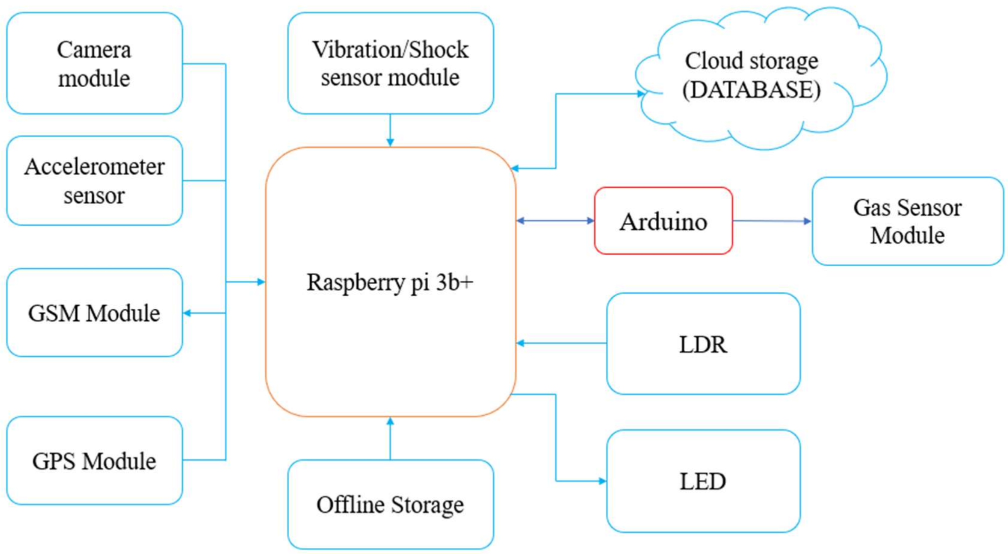

Fig. 3.1 Block Diagram of Accident Monitoring System

The above block diagram illustrates the functioning of smart vehicle safety and accident monitoring system through the use of IoT technology. The central processing unit of the system is the Raspberry Pi 3B+, acting as the control centre that manages various components. These include sensors like the SW-420, which detects vibrations, and the ADXL345, which detects the acceleration when the vehicle is tilted, providing crucial information in the event of an accident. We incorporate a camera module for visual data, a GSM module for alerting predefined contact, and a GPS module for pinpointing the exact location of an accident. To address the challenge of carbon monoxide gas detection, we employ an MQ-07 gas sensor connected to an Arduino, which serves as an intermediary to convert analog data into a digital format for the Raspberry Pi. We have included LED and LDR which will automatic light dimming and dipping, triggered by the detection of oncoming vehicles, to enhance overall road safety.

CHAPTER 4

DESIGN IMPLEMENTATION

The design implementation provides structured information that can be used to analyze and monitor the impact of product trends on their components and concentration of the elements in electronics.

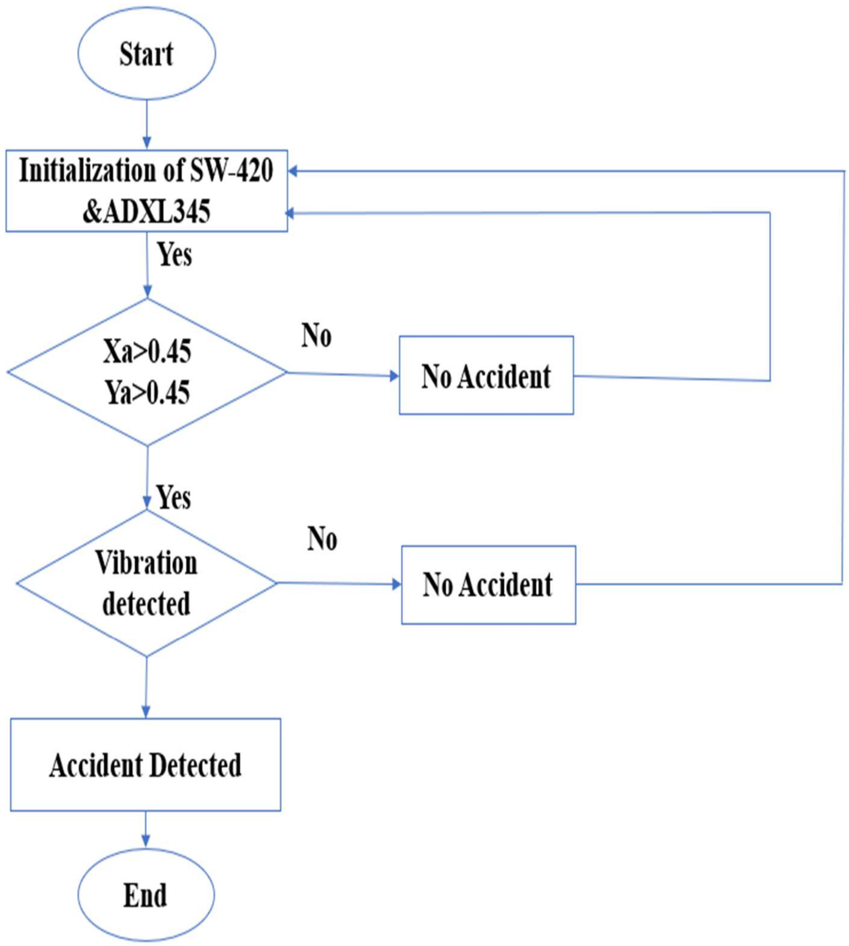

Fig. 4.1 Flow chart of Accident Detection

4.1 HARDWARE REQUIREMENTS

4.1.1 Raspberry pi 3b+
4.1.2 NEO-6M GPS Module
4.1.3 SIM800A Quad Band GSM GPRS Module
4.1.4 ADXL345 Module Accelerometer
4.1.5 SW-420 Vibration/Shock sensor module
4.1.6 MQ-7 Gas Sensor Module
4.1.7 Raspberry Pi 5MP Camera Module
4.1.8 Arduino Uno
4.1.9 LED (LIGHT EMITTING DIODE)
4.1.10 LDR (LIGHT DEPENDENT RESISTORS)

4.2 SOFTWARE REQUIREMENTS

4.2.1 Raspberry pi imager
4.2.2 Proteus 8.15
4.2.3 Arduino IDE 2.1.1

4.1.1 Raspberry pi 3b+

To connect the mobile robot to the Internet and the installation of the various programs, we need a certain computer specification. In this project we used microcomputer called 'Raspberry Pi because of its good specifications and possibilities and high flexibility in dealing with different programs. Figure 4.1.1 shows the Raspberry Pi model B useful in the field of electronics, computer science, computer and related projects of camera, gaming machines, robot, web server, media centre. Different versions available in market are pi-1, 2, 3 with model A. A+. B and B+.

a) Model A & B launched in 2012, Model B+ launched in 2014, Raspberry pi-3 introduced in 2016.

b) By installing Linux kernel operating system and interfacing corresponding port connections with devices it can be used for numerous operations of industries automation, robot, military spy and rescue. office, hospital, hazardous environment and agriculture, fine tasks, space missions, welding, painting.

c) Intended for education.

d) Used for many hobby projects, strong community.

e) Used even in commercial solutions due to low cost even that it is not intended for such use.

Fig. 4.1.1 Raspberry pi 3b+

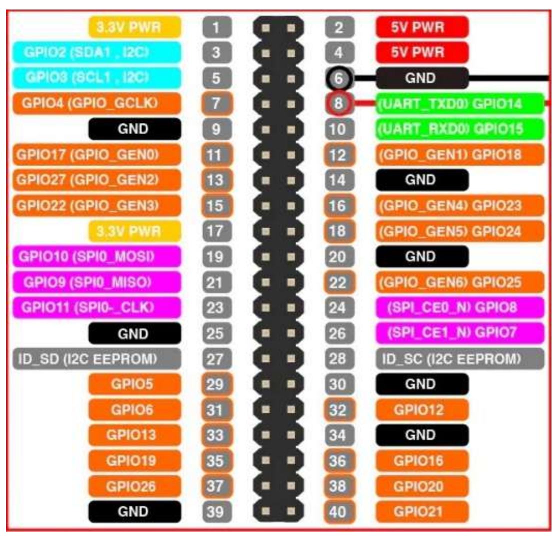

Fig. 4.1.2 Pin Configuration of Raspberry pi 3b+

4.1.2 NEO-6M GPS Module

This module has a default baud rate of 9600 and has multiple LEDs to indicate communication. We can save the configuration parameters in a pre-fitted EPROM memory. The GPS module has four pins: Ground, RXD, TXD, and VCC. The GPS module is connected to a Raspberry pi using these four pins. The new NEO 7 series introduces the

Ublox Neo 6M GPS Module, a highly sensitive and low-power module offering 56 channels for precise position updates at a rate of 10Hz. Equipped with an external GPS antenna and UART TTL connections, it ensures reliable connectivity. Additionally, the onboard rechargeable li-ion battery enables quick hot starts and faster GPS lock acquisition. The onboard battery allows configuration settings to be stored in EEPROM. The UFL connector facilitates easy connection with the GPS antenna using a cable, offering flexibility in module placement for optimal performance.

- The NEO-6M GPS module is a robust GPS receiver featuring a built-in ceramic antenna measuring 25 x 25 x 4mm.
- It has 5Hz position update rate.
- Operating temperature range: -40 TO 85.
- It has rechargeable battery for backup and the cold start time of 38 s and Hot start time of 1 s.
- It has supply voltage of 3.3 V and configurable from 4800 Baud to 115200 Baud rates.
- GPS with 162 dBm tracking sensitivity and separated by 18X18mm GPS antenna.

Fig. 4.1.2 NEO-6M GPS Module

4.1.3 SIM800A Quad Band GSM GPRS Module

The SIM800A Quad-Band GSM/GPRS Module with RS232 Interface is a comprehensive solution for Quad-band GSM/GPRS communication. It comes in an LGA (Land grid array) form factor and offers low power consumption while enabling voice, SMS, and data transmission. This module supports the 850/900/1800/1900 MHz bands.

To establish a connection with a computer, laptop, or microcontroller, the GSM module features an RS232 interface, which can be utilized through either a USB to Serial connector or an RS232 to TTL converter. Communication with the SIM800A module is achieved using AT instructions. The module comes with a SIM cardholder and an SMA antenna connector.

Fig. 4.1.3 SIM800A GSM module

The Fig. 4.1.3 shows SIM800A modem consists of a SIM800A GSM chip and an RS232 interface, facilitating easy connections to PCs or laptops via USB to Serial or to microcontrollers using the RS232 to TTL converter. Once the SIM800A modem is connected to the USB to RS232 connector, the correct COM port should be selected by checking the Device Manager of the USB to Serial Adapter.A serial connection needs to be established between the computer and the microcontroller by using a terminal program such as Putty. The connection should be made to the designated COM port at a baud rate of 9600, which is the default baud rate for the modem.

4.1.4 ADXL345 Module Accelerometer

ADXL345 is a Breakout board for the Analog Device's ADXL345 triple-axis accelerometer. The module is populated with MOSFET-based Voltage level conversion circuitry to enable you to interface with different types of microcontrollers (3V3 & 5V). It is ideal for motion and acceleration sensing applications.It is a low-power, 3-axis MEMS accelerometer module with both I2C and SPI interfaces.The ADXL345 is a low-power, MEMS, three-axis accelerometer module with both SPI and I2C interfaces to communicate with your controller like Arduino, Raspberry Pi, PIC, etc. It also has user-selectable sensitivity and 10-13bit of resolution. This Digital Accelerometer has a voltage regulator on board hence can be connected to both 5V and 3.3V powered controllers. Its high-resolution 4mg/LSB also enables it to measure less than 1° change in the orientation of the object. The ADXL Sensor can be used in Robotics applications, measuring vibration in a machine, in the data acquisition system of a vehicle, measuring the motions of a bridge, etc. It can also be used to detect taps on an object. The ADXL345 features 4 sensitivity ranges from +/- 2G to +/- 16G. And it supports output data rates ranging from 10Hz to 3200Hz such as the Raspberry pi. Output voltage: 3.3/5V.

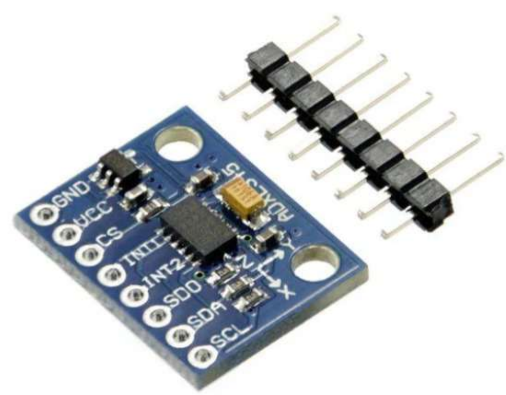

Fig. 4.1.4 ADXL345 Accelerometer

4.1.5 SW-420 Vibration/Shock sensor module

Vibration sensor module alarm Motion sensor module vibration switch SW-420 module based on the vibration sensor SW-420 and Comparator LM393 to detect if there is any vibration that beyond the threshold. The threshold can adjust using an onboard potentiometer. When this no vibration, this module output logic LOW the signal indicates LED light, and vice versa. If the module does not vibrate, the vibration switch was closed on state, the output of low output, and the green indicator light. The product vibrates, vibration switches momentary disconnect, the output is driven high, and the green light does not shine. The output can be directly connected to the microcontroller, which detects high and low levels, so as to detect whether the environment exists vibration, play a role in the alarm.

- Operating voltage 3.3V ~ 5V
- Current :15 mA

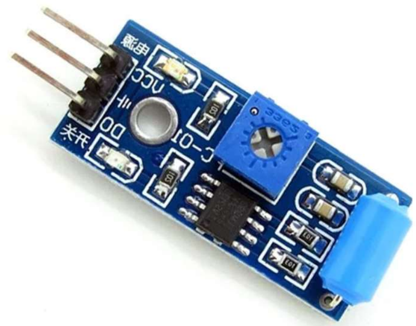

Fig. 4.1.5 SW-420 Vibration sensor

4.1.6 MQ-7 Gas Sensor Module

The MQ-7 gas sensor module is a device that can detect the presence of carbon monoxide (CO) gas in the air. It is commonly used in applications such as gas leak detectors, safety devices, and industrial control systems. The MQ-7 sensor operates by using a chemical reaction between the CO gas and a conductive metal oxide to produce a current. The magnitude of the current is proportional to the concentration of CO gas in the air. The gas sensor can be used to detect CO gas concentrations in the range of 20-2000 ppm. It is important to note that CO is a poisonous gas that can be dangerous at high concentrations.

It is recommended to use the MQ-7 sensor with caution and to take appropriate safety measures when working with it. The Fig 4.1.6 module provides both digital and analog outputs.

- Temperature: 20°C ± 2°C
- Humidity: 65% ± 5% RH
- Voltage: Vc: 5.0V ± 0.1V; Vh (high):5.0V±0.1V, Vh (low): 1.5V±0.1V

Fig. 4.1.6 MQ-07 gas sensor

4.1.7 Raspberry Pi 5MP Camera Module

The Fig 4.1.7 shows 5mp raspberry pi camera module which is a portable lightweight camera that is capable of 1080p video and still images and connects directly to your Raspberry Pi. Connect the included USB/ ribbon cable to the CSI (Camera Serial Interface) port on your Raspberry Pi, boot up the latest version of Raspbian and you are good to go. The module attaches to Raspberry Pi, by way of a 15 Pin Ribbon Cable, to the dedicated 15-pin MIPI Camera Serial Interface (CSI), which was designed especially for interfacing to the camera. It is commonly used in image processing, machine learning, and surveillance projects. Because the camera payload is so small, it is commonly used in capturing small images. Aside from these modules, this pi camera can also use standard USB webcams that are connected to a computer.

- Image Size (Pixels): 2592 x 1944.
- Dimensions: 2.5 x 2.3 x 1 cm.

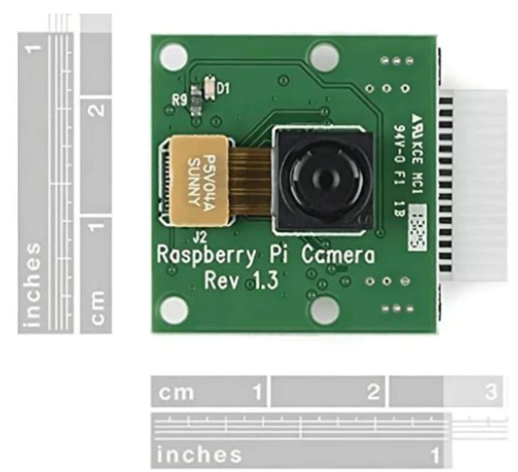

Fig. 4.1.7 5MP Raspberry pi camera module

4.1.8 Arduino Uno

The UNO R3 board compatible with Arduino has micro controllers: ATmega328. This UNO board compatible with arduino has 14 digital input/output pins (of which 6 can be used as PWM outputs) 6 analog inputs, a 16 MHz crystal oscillator, a USB connection, a power jack, an ICSP header, and a reset button.

- Microcontroller: ATmega328P
- Operating Voltage: 5V

- Input Voltage: 7-12V
- Digital I/O Pins: 14 (of which 6 provide PWM output)
- Analog Input Pins: 6
- DC Current per I/O Pin: 20 mA
- DC Current for 3.3V Pin: 50 mA
- Clock Speed: 16 MHz

Fig. 4.1.8 Arduino Uno

4.2.1 Raspberry pi imager

Raspberry Pi OS, previously known as Raspbian, features the PIXEL desktop environment, based on LXDE and Openbox window manager. It resembles common desktops like macOS and Windows, with a menu bar at the top containing application menu, web browser (Chromium), file manager, and terminal shortcuts. Additionally, it includes Bluetooth, Wi-Fi, volume control, and clock menus. The default distribution comes with Wolfram Mathematica, VLC, and a lightweight version of the Chromium web browser. Raspberry Pi OS is optimized for Raspberry Pi with ARM CPUs and is available in three installation versions: Raspberry Pi OS Lite (32-bit & 64-bit), Raspberry Pi OS with desktop (32-bit & 64-bit), and Raspberry Pi OS with desktop and recommended software (32-bit). There are also two legacy versions: Raspberry Pi OS Lite (Legacy) (32-bit) and Raspberry Pi OS (Legacy) with desktop (32-bit). Packages can be installed via APT, the Recommended Software app, and the Add/Remove Software tool, which is a GUI wrapper for APT.

4.2.2 Proteus 8.15 software

Proteus Design Suite is a Windows application that encompasses schematic capture, simulation, and PCB layout design. It is available in various configurations based on the size of designs and microcontroller simulation requirements. Schematic capture is a core component of the suite, used for both design simulation and the initial phase of PCB layout projects. The microcontroller simulation involves applying a hex file or debug file to the microcontroller part on the schematic, allowing co-simulation with connected analog and digital electronics. This feature is widely used in project prototyping, including areas such as motor control, temperature control, and user interface design, as well as in the hobbyist and educational communities. The PCB Layout module automatically receives connectivity information from the schematic capture module in the form of a netlist, and utilizes this data, along with user-specified design rules and design automation tools, to facilitate error-free board design. The suite supports the production of PCBs with up to 16 copper layers, with design size limitations based on the product configuration. Additionally, the 3D Viewer module enables the visualization of the board in 3D, along with a semi-transparent height plane representing the board's enclosure. STEP output can be used to transfer the design to mechanical CAD software for accurate mounting and positioning of the board.

4.2.3 Arduino IDE 2.1.1

The Arduino IDE 2 is an improvement of the classic IDE, with increased performance, improved user interface and many new features, such as autocompletion, a built-in debugger and syncing sketches with Arduino Cloud.

Verify / Upload - compile and upload your code to your Arduino Board.
Select Board & Port - detected Arduino boards automatically show up here, along with the port number.
Sketchbook - here you will find all of your sketches locally stored on your computer. Additionally, you can sync with the Arduino Cloud, and also obtain your sketches from the online environment.
Boards Manager - browse through Arduino & third party packages that can be installed. For example, using a MKR WiFi 1010 board requires the Arduino SAMD Boards package installed.
Library Manager - browse through thousands of Arduino libraries, made by Arduino & its

community.

Debugger - test and debug programs in real time.

Search - search for keywords in your code.

Open Serial Monitor - opens the Serial Monitor tool, as a new tab in the console.

CHAPTER 5

RESULTS AND DISCUSSIONS

5.1 Software Result:

The simulation of the smart vehicle safety and accident monitoring system using proteus 8 professional software

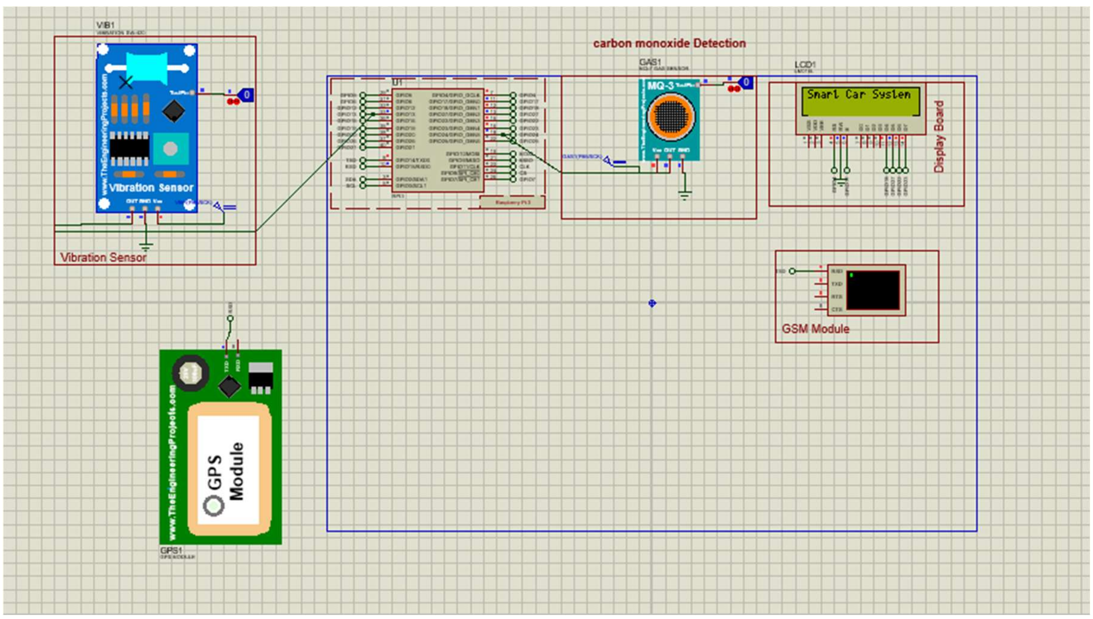

Fig. 5.1.1 Proteus Simulation

The output from the software simulation obtained in LCD module and Virtual terminal

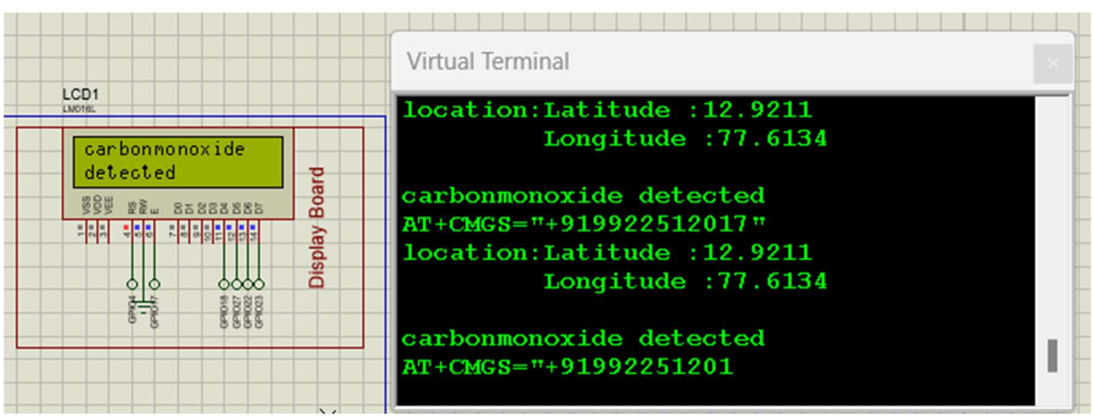

Fig. 5.1.2 carbonmonoxide detection output from proteus simulation

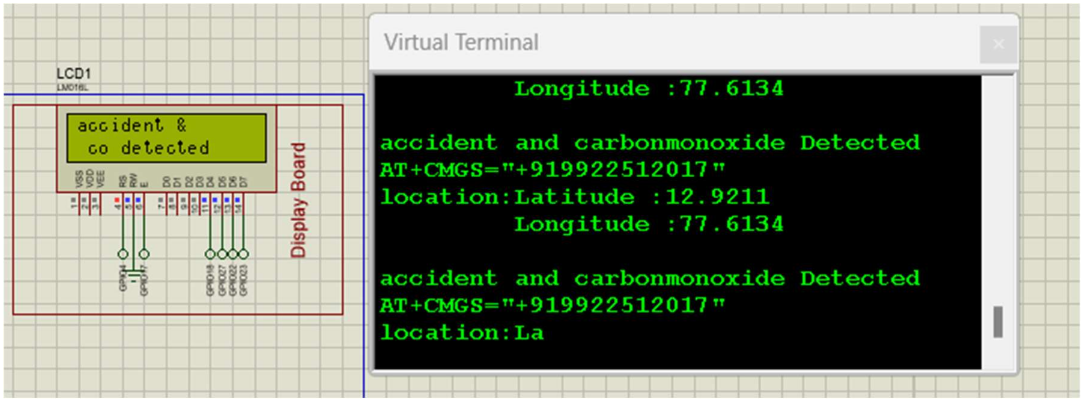

Fig. 5.1.3 accident and carbon monoxide detection output from proteus simulation

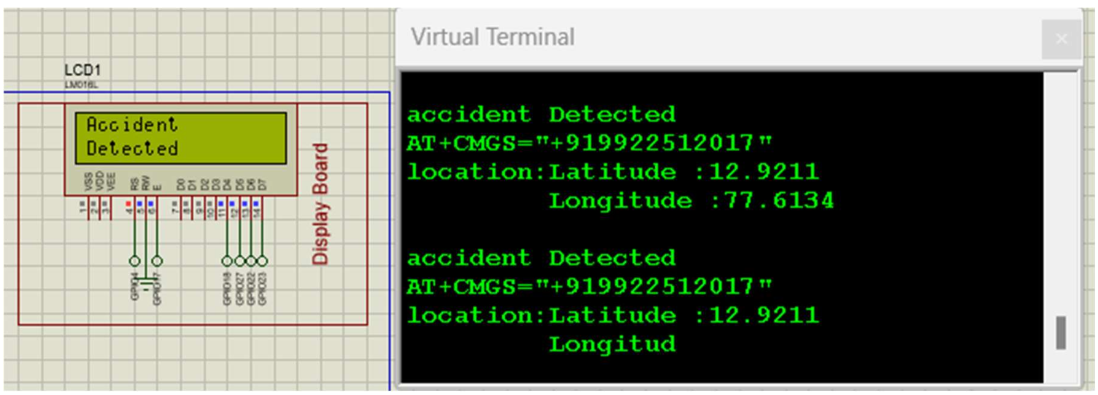

Fig. 5.1.4 accident detection output from proteus simulation

5.2 Hardware Result:

The model has been integrated with ADXL345 and SW-420 is shown in fig 5.2

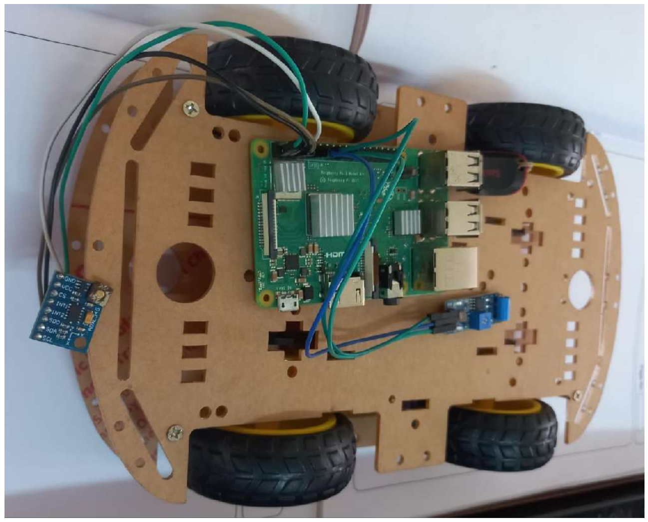

Fig. 5.2 Model of Accident Monitoring System

CHAPTER 6

CONCLUSION

6.1 CONCLUSION

The integration of accelerometer sensors and vibration sensors in IoT-based smart vehicle safety and accident monitoring systems represents a significant advancement in safety technology. Accelerometer sensors are employed to monitor and analyze vibrations and provide real-time data for accident detection and safety enhancement.

The proposed system is designed to improve road safety by reducing the likelihood of accidents, thereby enhancing the overall safety and monitoring capabilities in diverse sectors. This advancement ultimately leads to safer driving experiences and benefits both individuals and communities.

6.2 FUTURE SCOPE

The future scope of the IoT-based smart vehicle safety and accident monitoring system includes integrating a gas sensor to detect and respond to gas leakage and a camera module to enhance overall safety measures. It will enable real-time Location Tracking using GPS sensor with an integrated GSM module for automated accident reports. Additionally, it will incorporate Intelligent Vehicle Interaction features, such as automatic light dimming and dipping triggered by the detection of oncoming vehicles, to enhance overall road safety

REFERENCE

[1] **"Arduino Based Vehicle Accident Alert System Using GPS, GSM and MEMS Accelerometer",** by Pachipala Yellamma; N S N S P Chandra; Puli Sukhesh; Puligadda Shrunith; Sunkesula Siva Teja.IEEE published in the year 2021.

[2] **"Accident detection using Automotive Smart Black-Box based Monitoring system"** P. Josephinshermila, S. Sharon priya, K. Malarvizhi, Ramakrishna hegde, S. Gokul Pran, B. Veerasamy. Science direct published in the year of 2023.

[3] **"Smart Security System for Vehicles using Internet of Things (IoT)",** Mithileysh Sathiyanarayanan, Santosh Mahendra, Rajesh Babu Vasu. IEEE published in the year of 2021.

[4] **"A smart IOT based black-box system for automobiles"**, Vanitha M, Arun kumar K, Hemamalini A, Alam Yaswanth,Department of Electronics and Communication Engineering, Saveetha Engineering College. IEEE published in the year 2023.

[5] **"IoT based Automatic Smart Black Box System to Detect Accidents",** Srinivasan, Kommalapati Avinash, Kommineni Manjunadh, K Jyothish. IEEE published in the year of 2023.

[6] **"Research on Traffic Accident Detection Based on Vehicle Perspective"**, by Zhuofei Xia; Jiayuan Gong; Yue Long; Wenbo Ren; Jingnan Wang; Haitao Lan Publisher: IEEE in the year 2022.

[7] **"Certifying Black-Box Policies With Stability for Nonlinear Control",** Tongxin L, Ruixiao Yang, Guannan Qu, YihengI Lin, Adam Wierman, and Steven H, 2023.

[8] "**AUTOMATIC DIM AND DIP SYSTEM FOR VEHICLE",** by. Manikandan, P. Sivakumar, G. Kumar Sai Reddy, M. Charan Teja Vyas. Published in the year of 2021.

[9] **"Black-box Adversarial Attacks in Autonomous Vehicle Technology",** K. Naveen Kumar, C. Vishnu, Reshmi Mitra, C. Krishna Mohan. IEEE published in the 2020.

[10] **"Design of GPS Service System for Vehicle Monitoring of Intelligent Expressway",** Changhua Wang, Hancheng Yu. IEEE published in the year of 2022.

APPENDIX

Software:

```
#!/usr/bin/python
import time
import RPi.GPIO as GPIO
import pio
import Ports
#import serial
GPIO.setmode(GPIO.BOARD)
GPIO.setwarnings(False)
pio.uart=Ports.UART () # Define serial port
'''
define pin for lcd
'''
# Timing constants
E_PULSE = 0.0005
```

```
E_DELAY = 0.0005

# Define GPIO to LCD mapping

LCD_RS = 7

LCD_E  = 11

LCD_D4 = 12

LCD_D5 = 13

LCD_D6 = 15

LCD_D7 = 16

alcohol_Sensor = 18

vibration_sensor = 33

GPIO.setup(LCD_E, GPIO.OUT)  # E

GPIO.setup(LCD_RS, GPIO.OUT) # RS

GPIO.setup(LCD_D4, GPIO.OUT) # DB4

GPIO.setup(LCD_D5, GPIO.OUT) # DB5

GPIO.setup(LCD_D6, GPIO.OUT) # DB6

GPIO.setup(LCD_D7, GPIO.OUT)

GPIO.setup(alcohol_Sensor, GPIO.IN)

GPIO.setup(vibration_sensor, GPIO.IN)

# Define some device constants

LCD_WIDTH = 16    # Maximum characters per line

LCD_CHR = True

LCD_CMD = False

LCD_LINE_1 = 0x80 # LCD RAM address for the 1st line

LCD_LINE_2 = 0xC0 # LCD RAM address for the 2nd line
```

```
'''
Function Name :lcd_init()
Function Description : this function is used to initialized lcd by sending the different
commands
'''
def lcd_init():
  # Initialise display
  lcd_byte(0x33,LCD_CMD) # 110011 Initialise
  lcd_byte(0x32,LCD_CMD) # 110010 Initialise
  lcd_byte(0x06,LCD_CMD) # 000110 Cursor move direction
  lcd_byte(0x0C,LCD_CMD) # 001100 Display On,Cursor Off, Blink Off
  lcd_byte(0x28,LCD_CMD) # 101000 Data length, number of lines, font size
  lcd_byte(0x01,LCD_CMD) # 000001 Clear display
  time.sleep(E_DELAY)
'''
Function Name :lcd_byte(bits ,mode)
Fuction Name :the main purpose of this function to convert the byte data into bit
and send to lcd port
'''
def lcd_byte(bits, mode):
  # Send byte to data pins
  # bits = data
  # mode = True  for character
  #        False for command
```

```
GPIO.output(LCD_RS, mode) # RS

# High bits

GPIO.output(LCD_D4, False)

GPIO.output(LCD_D5, False)

GPIO.output(LCD_D6, False)

GPIO.output(LCD_D7, False)

if bits&0x10==0x10:

  GPIO.output(LCD_D4, True)

if bits&0x20==0x20:

  GPIO.output(LCD_D5, True)

if bits&0x40==0x40:

  GPIO.output(LCD_D6, True)

if bits&0x80==0x80:

  GPIO.output(LCD_D7, True)


# Toggle 'Enable' pin

lcd_toggle_enable()


# Low bits

GPIO.output(LCD_D4, False)

GPIO.output(LCD_D5, False)

GPIO.output(LCD_D6, False)

GPIO.output(LCD_D7, False)

if bits&0x01==0x01:
```

```
    GPIO.output(LCD_D4, True)
  if bits&0x02==0x02:
    GPIO.output(LCD_D5, True)
  if bits&0x04==0x04:
    GPIO.output(LCD_D6, True)
  if bits&0x08==0x08:
    GPIO.output(LCD_D7, True)


  # Toggle 'Enable' pin
  lcd_toggle_enable()
'''
Function Name : lcd_toggle_enable()
Function Description:basically this is used to toggle Enable pin
'''
def lcd_toggle_enable():
  # Toggle enable
  time.sleep(E_DELAY)
  GPIO.output(LCD_E, True)
  time.sleep(E_PULSE)
  GPIO.output(LCD_E, False)
  time.sleep(E_DELAY)
'''
Function Name :lcd_string(message,line)
Function  Description :print the data on lcd
```

```
'''
def lcd_string(message,line):
  # Send string to display


  message = message.ljust(LCD_WIDTH," ")


  lcd_byte(line, LCD_CMD)


  for i in range(LCD_WIDTH):
    lcd_byte(ord(message[i]),LCD_CHR)


lcd_init()
lcd_string("welcome ",LCD_LINE_1)
time.sleep(0.1)
lcd_byte(0x01,LCD_CMD) # 000001 Clear display
lcd_string("Smart Car System",LCD_LINE_1)
time.sleep(0.1)
while 1:
    # Print out results
    alcohol_data = GPIO.input(alcohol_Sensor)
    vibration_data = GPIO.input(vibration_sensor)


    if vibration_data == True:
        if alcohol_data == True:
```

```
        lcd_byte(0x01, LCD_CMD)  # 000001 Clear display
        lcd_string("accident &", LCD_LINE_1)
        lcd_string(" co detected", LCD_LINE_2)
        pio.uart.println("accident and carbonmonoxide Detected")
        pio.uart.println("AT+CMGS=\"+919922512017\"")
        pio.uart.println("location:Latitude :12.9211")
        pio.uart.println(" Longitude :77.6134\r")
      else:
        lcd_byte(0x01, LCD_CMD)  # 000001 Clear display
        lcd_string("Accident", LCD_LINE_1)
        lcd_string("Detected", LCD_LINE_2)
        pio.uart.println("accident Detected")
        pio.uart.println("AT+CMGS=\"+919922512017\"")
        pio.uart.println("location:Latitude :12.9211")
        pio.uart.println("        Longitude :77.6134\r")
    elif alcohol_data == True:
      lcd_byte(0x01, LCD_CMD)  # 000001 Clear display
      lcd_string("carbonmonoxide", LCD_LINE_1)
      lcd_string("detected", LCD_LINE_2)
      pio.uart.println("carbonmonoxide detected")
      pio.uart.println("AT+CMGS=\"+919922512017\"")
      pio.uart.println("location:Latitude :12.9211")
      pio.uart.println("        Longitude :77.6134\r")
    else:
```

```
        lcd_byte(0x01, LCD_CMD)  # 000001 Clear display
        lcd_string("No Accident", LCD_LINE_1)
        lcd_string("Detected", LCD_LINE_2)
```

Hardware:

```
import smbus
import time
import RPi.GPIO as GPIO

ADXL345_ADDRESS = 0x53
VIBRATION_SENSOR_PIN = 17  # GPIO pin for the vibration sensor

bus = smbus.SMBus(1)

# Set ADXL345 in measuring mode
bus.write_byte_data(ADXL345_ADDRESS, 0x2D, 8)  # Enable measurement
time.sleep(0.01)

# Set up GPIO for the vibration sensor
GPIO.setmode(GPIO.BCM)
GPIO.setup(VIBRATION_SENSOR_PIN, GPIO.IN)

try:
    while True:
        # Read accelerometer data
```

```
    bus.write_byte_data(ADXL345_ADDRESS, 0x32, 0x80 | 0x40 | 0x20 | 0x08)  # Set to read mode
    data = bus.read_i2c_block_data(ADXL345_ADDRESS, 0x32, 6)

    xa = (data[1] << 8) | data[0]
    ya = (data[3] << 8) | data[2]
    za = (data[5] << 8) | data[4]

    xa = xa if xa < 0x8000 else xa - 0x10000
    ya = ya if ya < 0x8000 else ya - 0x10000
    za = za if za < 0x8000 else za - 0x10000

    xa /= 256.0
    ya /= 256.0
    za /= 256.0

    print(f"Xa= {xa:.2f}   Ya= {ya:.2f}   Za= {za:.2f}")

    # Check for significant change in x-axis values
    if (abs(xa) > 0.45 or abs(ya) > 0.45):
        # Check vibration from the sensor
        if GPIO.input(VIBRATION_SENSOR_PIN):
            print("Accident Detected!")
            break
```

```
            else:
                print("No accident detected.")
                continue

        else:
            print("No accident detected.")
            continue  # Continue to the next iteration

        time.sleep(1)

except KeyboardInterrupt:
    pass
finally:
    bus.close()
    GPIO.cleanup()  # Cleanup GPIO settings before exiting
```