A project report on

ACCIDENT DETECTION ALARM

Submitted in partial fulfilment for the award of the degree of

INTEGRATED M-TECH

By

R NAGA ROHITH (19MIS7109)

K SAI KIRAN (19MIS7091)

A SHASHIDHAR (19MIS7066)

AMARAVATI

SCOPE

July,2022

CERTIFICATE

This is to certify that the Summer Project/ summer Internship work titled "**Accident Detection Alarm**" that is being submitted by **Naga Rohith Ramanadham (19MIS7109)** is in partial fulfillment of the requirements for the award of **Master of Technology (Integrated 5 Year) software engineering**, is a record of bonafide work done under my guidance. The contents of this Project work, in full or in parts, have neither been taken from any other source nor have been submitted to any other Institute or University for award of any degree or diploma and the same is certified.

Dr. D Sumathi

Guide

The thesis is satisfactory

Approved by

PROGRAM CHAIR

M. Tech. SE

DEAN

School Of Computer Science and Engineering

ABSTRACT

Every hour or so, several accidents occur. The two-wheeler vehicles are the source of the majority of dangerous accidents out of all accidents. Many individuals use two-wheelers to go around in their daily lives and get where they need to go. Out of all accidents, some take place away from populated places where people don't often live or commute, and some happen close to busy intersections. Who are the victim's well-wishers or family members is the first query people ask? Even if we try to contact his relatives, his/her phone can be locked.

In order to solve this issue, we developed a smart helmet that could recognise accidents and notify the victim's family of the accident's location. This might be useful in a variety of situations, such as when a person has an accident in a remote area outside of a city or town where no one else is present. Since the helmet alerts the family, they can react quickly to the situation and take action to save the life. From the scene of the accident, the family member can reach neighbouring medical facilities.

ACKNOWLEDGEMENT

It is my pleasure to express with deep sense of gratitude to Dr. D Sumathi, Associate Professor Grade - 1, VIT-AP, for her constant guidance, continual encouragement, understanding; more than all, he taught me patience in my endeavor. My association with her is not confined to academics only, but it is a great opportunity on my part of work with an intellectual and expert in the field of Internet of Things (IoT).

I would like to express my gratitude to Dr. G. Viswanathan, Dr. Sekar Viswanathan, Sankar Viswanathan, G. V. Selvam, Kadhambari S. Viswanathan, Dr. S. V. Kota Reddy, and Dr. Sudha S V, School of Computer Science and Engineering, for providing with an environment to work in and for his inspiration during the tenure of the course.

In jubilant mood I express ingeniously my whole-hearted thanks to Dr. Reeja S R, Program Chair Master of Technology in Integrated Software Engineering (MTSE), Associate Professor Grade-1, all teaching staff and members working as limbs of our university for their not-self-centered enthusiasm coupled with timely encouragements showered on me with zeal, which prompted the acquirement of the requisite knowledge to finalize my course study successfully. I would like to thank my parents for their support.

It is indeed a pleasure to thank my friends who persuaded and encouraged me to take up and complete this task. At last but not least, I express my gratitude and appreciation to all those who have helped me directly or indirectly toward the successful completion of this project.

Place: Amaravati

Date: 30-07-2022

Naga Rohith Ramanadham

K Sai Kiran

Akula Shashidhar

LIST OF FIGURES

CONTENTS

Chapter 1

SYSTEM REQUIREMENT SPECIFICATION

1.1 Introduction

1.1.1 PURPOSE

The purpose of this project is to help people who might meet with an accident in the future. The main purpose of the project is to avoid unidentified accidents that happen now and then, and to make sure that no one would be left untreated in a road accident.

1.1.2 PROBLEM STATEMENT

The problem of:

a) Unidentified accidents
b) Untreated deaths
c) Late responses

Affects:

a) Getting to know that his well-wisher had met with an accident.
b) Ease of finding location of accident occurred area.
c) Contacting nearby hospitals

Results in:

a) The victim's well-wisher gets notified when he/she had met with an accident.

Benefits:

a) Getting to know the location of accident occurred area.
b) Reduction of unidentified accidents.
c) Victim can get the treatment as soon as possible as victim's well-wisher can do the needful at the earliest.

1.2 OVERVIEW

This project goal is to make sure that no one would be left untreated in a road accident. Therefore, this smart helmet can be used to send message to the victim's family or well-wisher when an accident had occurred which includes the location of the spot. To achieve this project, we have used various components that will be discussed in detail.

a) RaspberryPi
   It is a low-cost microprocessor that is of a pocket-sized machine, it contains GPIO pins where various sensors can be put to work.

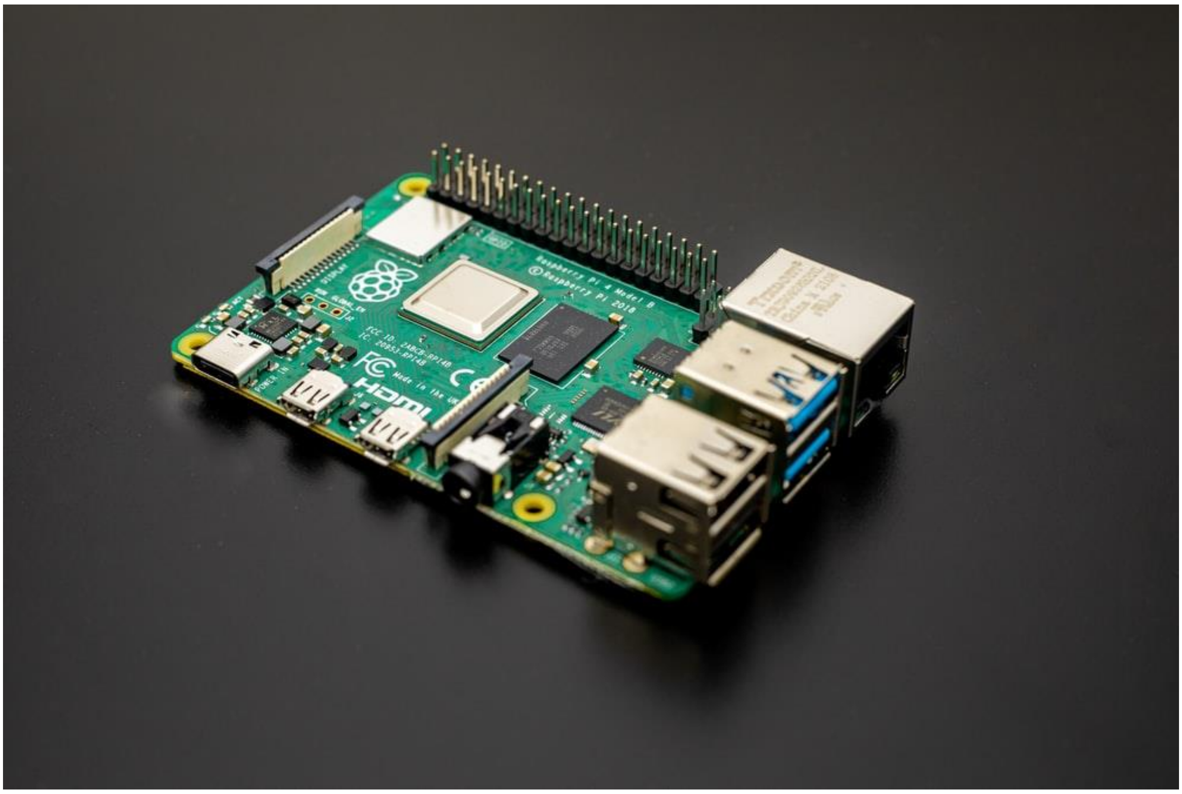

Fig 1.1

b) Accelerometer MPU6050
   It is made up of a three-axis gyroscope and an accelerometer. It aids in the measurement of motion properties such as acceleration, displacement, velocity, and orientation. These features allow us to determine the rider's acceleration and speed. The acceleration numbers indicate whether the helmet has experienced a significant impact.

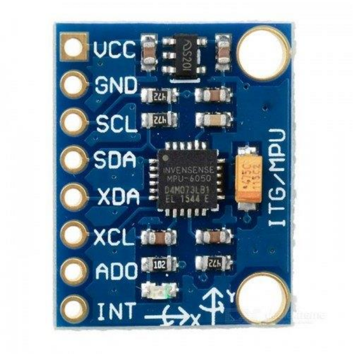

Fig 1.2

c) NEO 6M GPS Module

This component can be used to get the exact location of the accident, it gives the exact latitude and longitude of the accident spot. It has a built-in antenna that connects with the satellites which helps in getting the accurate GPS location.

Fig 1.3

The expected outcome from this project is to successfully send a message to the victim's well-wishers or family member with accurate location of the accident spot.

Fig 1.4

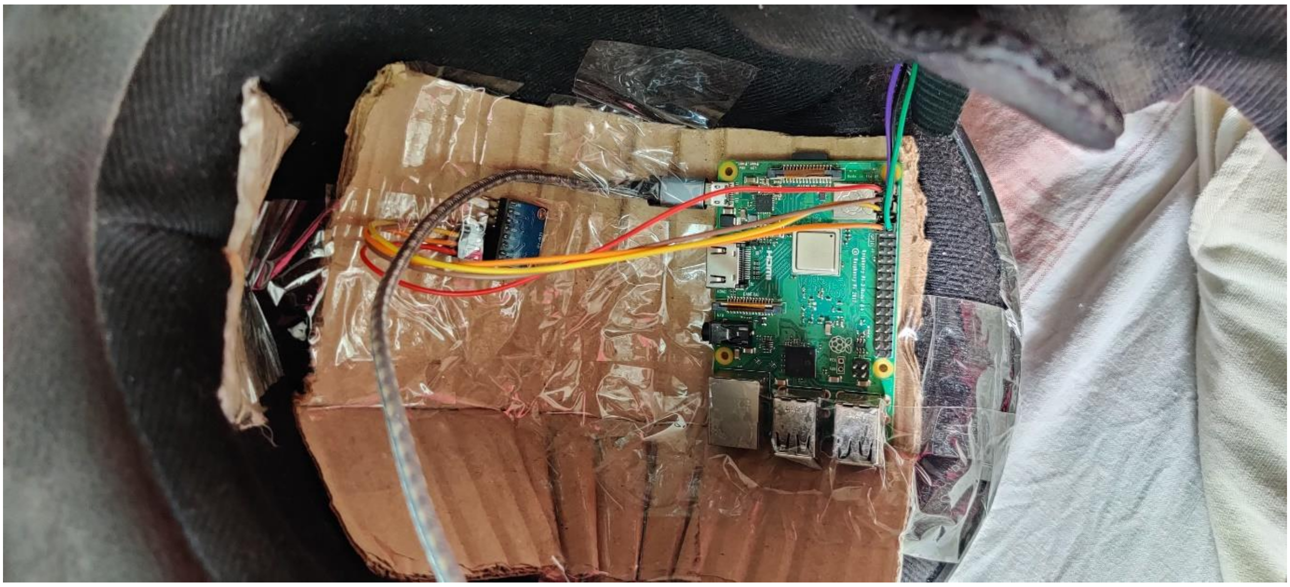

Fig 1.5

Flow chart of the project:

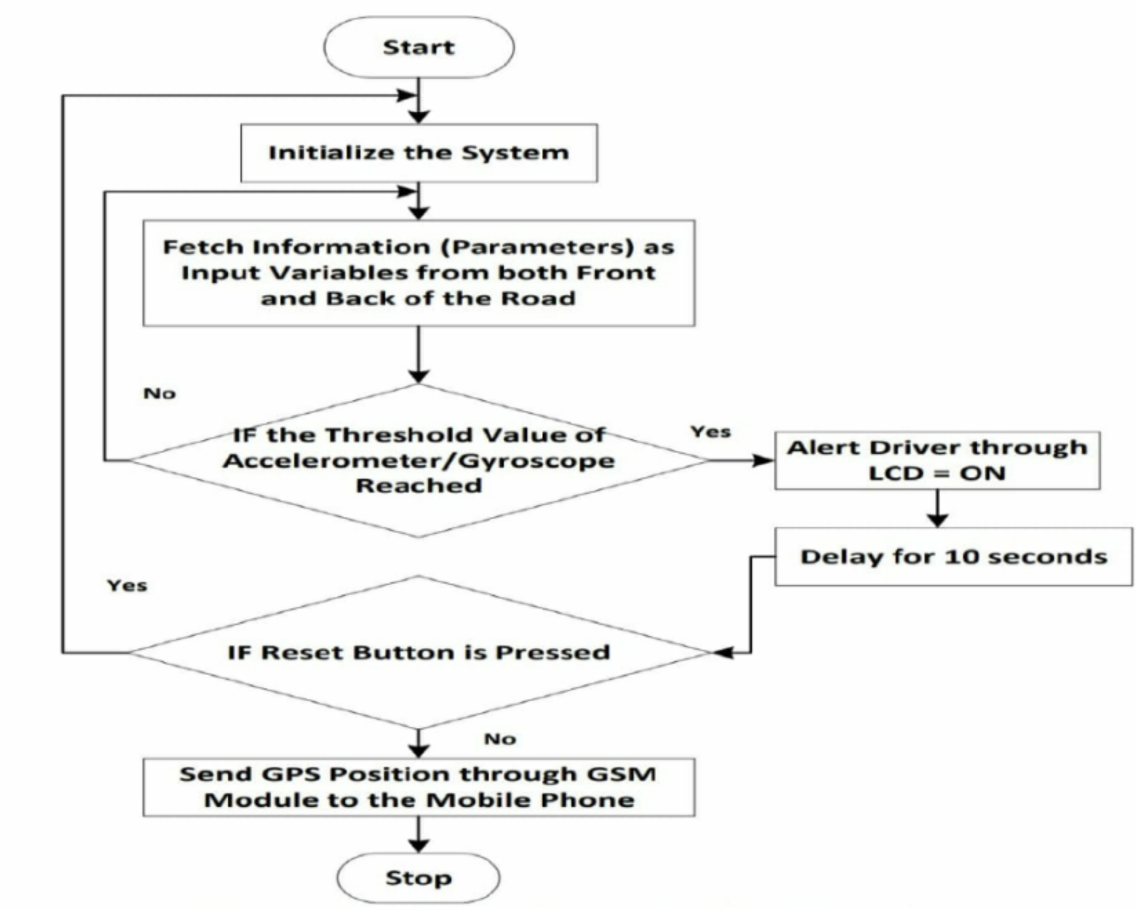

Fig 1.6

1.3 CHALLENGES IN THE PROJECT

a) Biker not wearing the helmet
b) False alarms
c) Component failure
d) Inaccurate location

1.4 PROJECT STATEMENT

To caution the well-wishers or family member when an accident had occurred.

1.5 OBJECTIVES

This project goal is to make sure that no one would be left untreated in a road accident. Therefore, this smart helmet can be used to send message to the victim's family or well-wisher when an accident had occurred which includes the location of the spot, so they can do the needful at the earliest.

1.6 SCOPE OF THE PROJECT

The scope of the project is to send message to the family or well-wishers when the person had met with an accident which includes the location of the accident spot. The future scope of this project is to send an automated call or message to the nearby hospital which includes the exact location of the accident spot.

Chapter 2

LITERATURE SURVEY

Everyone faces a serious problem with traffic accidents. Numerous priceless lives are lost as a result of vehicle accidents every day. Given the high rates of mortality associated with traffic accidents, it is the most important field that demands major investigation. The two most frequent causes are driver mistake and sluggish emergency department response times. To rescue injured people, a reliable identification and information-sharing system for traffic accidents is needed. It is crucial to have a gadget that can tell local emergency personnel about the crash site so they can act swiftly. Numerous researchers have suggested various automatic accident warning systems in the scientific literature. The Global System for Mobile Communications (GSM) and Global Positioning System (GPS) technologies, vehicular ad hoc networks, different machine learning techniques, and mobile apps are a few examples. An automatic road accident detection and information communication system should be placed in every vehicle. By highlighting their advantages, drawbacks, and issues that need to be fixed in order to assure traffic safety and save lives, this study paper offers a comprehensive analysis of various emerging approaches for anticipating and averting road crashes.

Over 1,300,000 deaths occur each year in road accidents, which are one of the leading causes of death in the globe. Those closest to the scene of the collision or the traffic police who are on the roadways are typically the ones to report it. As a result, it takes longer for accident reports to be filed and longer for police and rescue personnel to arrive at the accident scene.

The major goal of this study is to build a system that will shorten the time needed to report an accident and will pinpoint its position more precisely and quickly. As a result, it will take less time for the police and other emergency workers to arrive at the scene of the accident. The suggested method will make location identification automatic, making it more accurate and speeding up the process. Thus, it will lessen the number of people killed in traffic accidents.

A GPS receiver linked with a GSM/GPRS module put on the car or helmet, a communication link between the car and the dispatcher, and PC-based tracking software for dispatching make up an accident detection alarm system in general.

Today, a wide range of cutting-edge technology are available to secure the vehicle and tracking. In the past, it was possible to transfer accident information but not the location of the accident. Airbags are utilised for security and safety while travelling in any vehicle. In 1968, the air bag system was first made available.

There have been other additional ways suggested to determine the accident. The current system uses two sensors, including an accelerometer sensor for detecting changes in a vehicle's acceleration.

The other current approach makes advantage of cloud computing and IOT. Where the Ant Colony Algorithm (ACA) is used to develop SVM (support vehicle machine) is used to recognise vehicles. Magneto resistive sensors will be used in this instance to monitor the automobiles. This project's primary goal is to distinguish between accidents that occurred in traffic and those that occurred in non-traffic areas.

Using the NEO 6M GPS Module, the current system additionally provides the accident's position. The saved cell phone numbers are used to get the information.

Chapter 3

COMPONENTS

3.1 Raspberry PI

Raspberry PI is a line of single-board computers that are increasingly used to link Internet of Things (IoT) devices. A computer monitor can be connected to an RPi. It is a competent tiny gadget that permits individuals to investigate computing and learn how to write in languages like Scratch and Python. It can also do everything that users expect a computer to do. The Raspberry Pi has a means of contacting the outside world. It is utilised in many different types of digital maker projects, including as music players, parent detectors, weather stations, and many more.

In addition to being a relatively affordable Linux-running computer, the Raspberry Pi also offers a set of GPIO (general purpose input/output) pins that let you explore the Internet of Things and control electronic components for physical computing (IoT).

The Raspberry Pi works within the open-source community; it uses Linux (a number of versions), and its primary supported operating system, Pi OS, is open source and makes use of a number of open-source applications. The Raspberry Pi Foundation releases a large portion of its own software as open source and also contributes to the Linux kernel and numerous other open-source projects.

The row of GPIO (general-purpose input output) pins on the Raspberry Pi is one of its great features, and the GPIO Pinout offers an interactive reference to these GPIO pins.

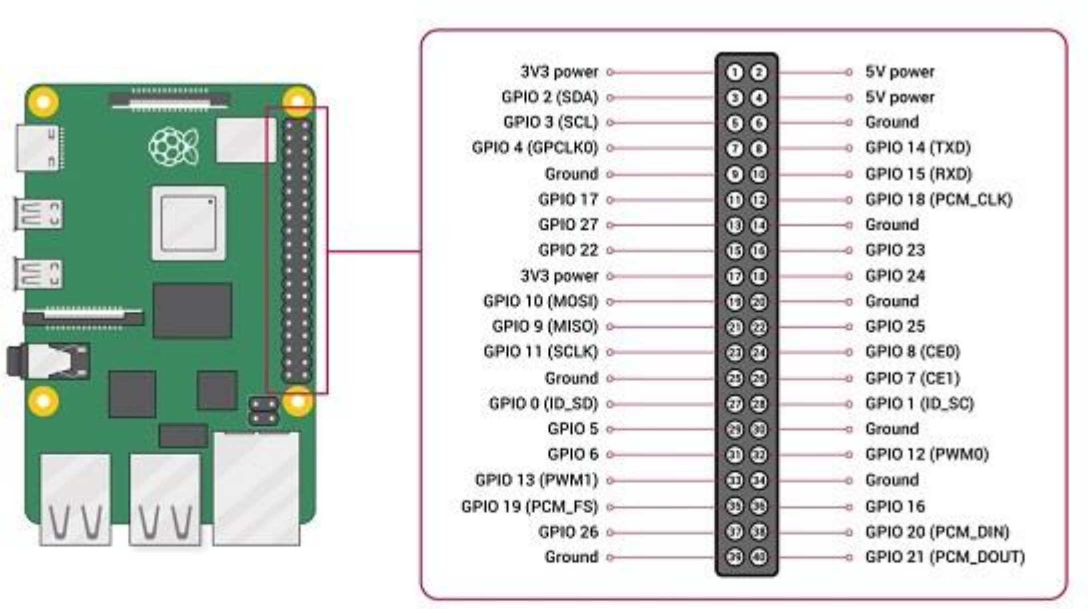

Fig 3.1

Two 5V pins and two 3V3 pins are present on the board, as seen in the above diagram.

An input pin can be assigned to a GPIO pin. The input pin can be interpreted as 0V (low) or 3V3 (high). You can use internal pull-up or pull-down resistors.

Out of all the pins we will be using pin number 1,3,5 and 9 to connect accelerometer where pin number 1 refers to 3v3 power supply, pin number 3 refers to I2C SDA, pin number 5 refers to I2C SCL and pin number 9 refers to ground connection.

And we will be using 2,6 and 10 to connect GPS module where pin number 2 refers to 5V power supply, pin number 6 refers to ground and pin number 10 refers to Rx that is used to receive data from the GPS module.

This device is used as a microcontroller to control the overall sensors that are put to work.

PI front side:

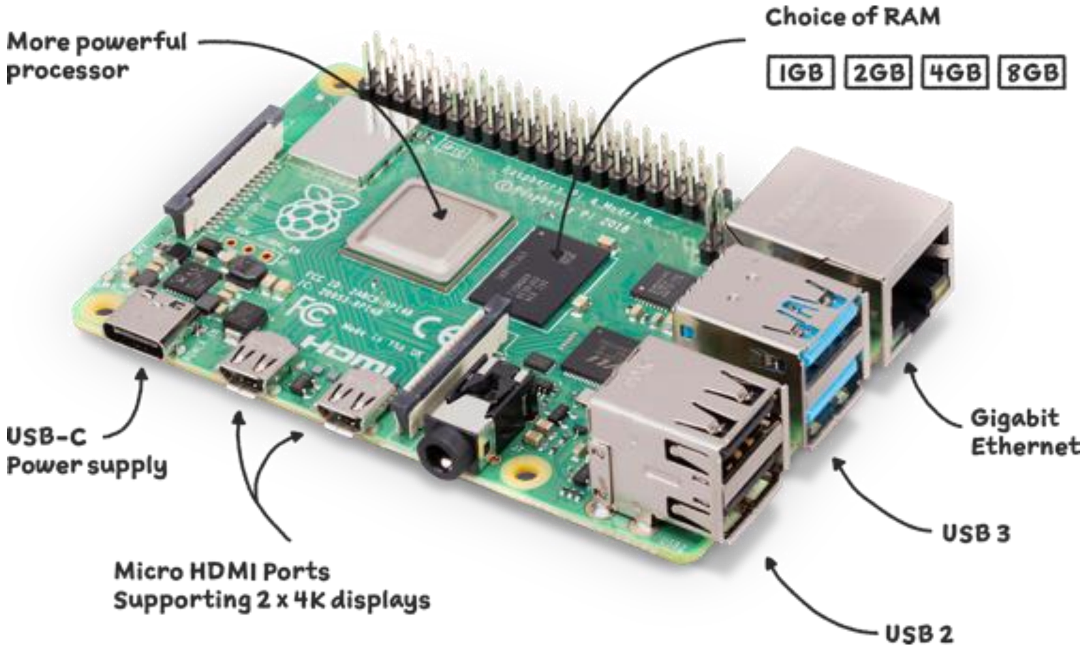

Fig 3.2

PI back side:

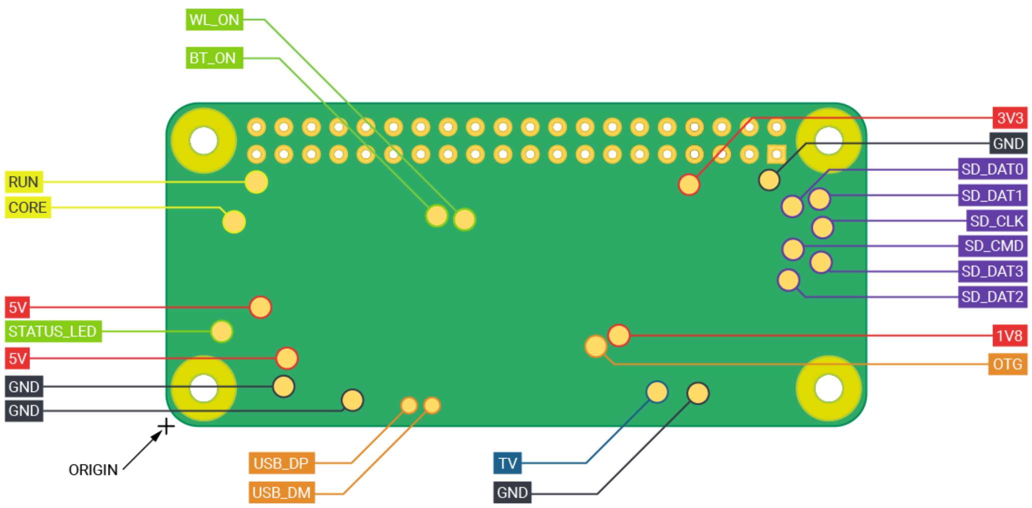

Fig 3.3

PI schematic diagram:

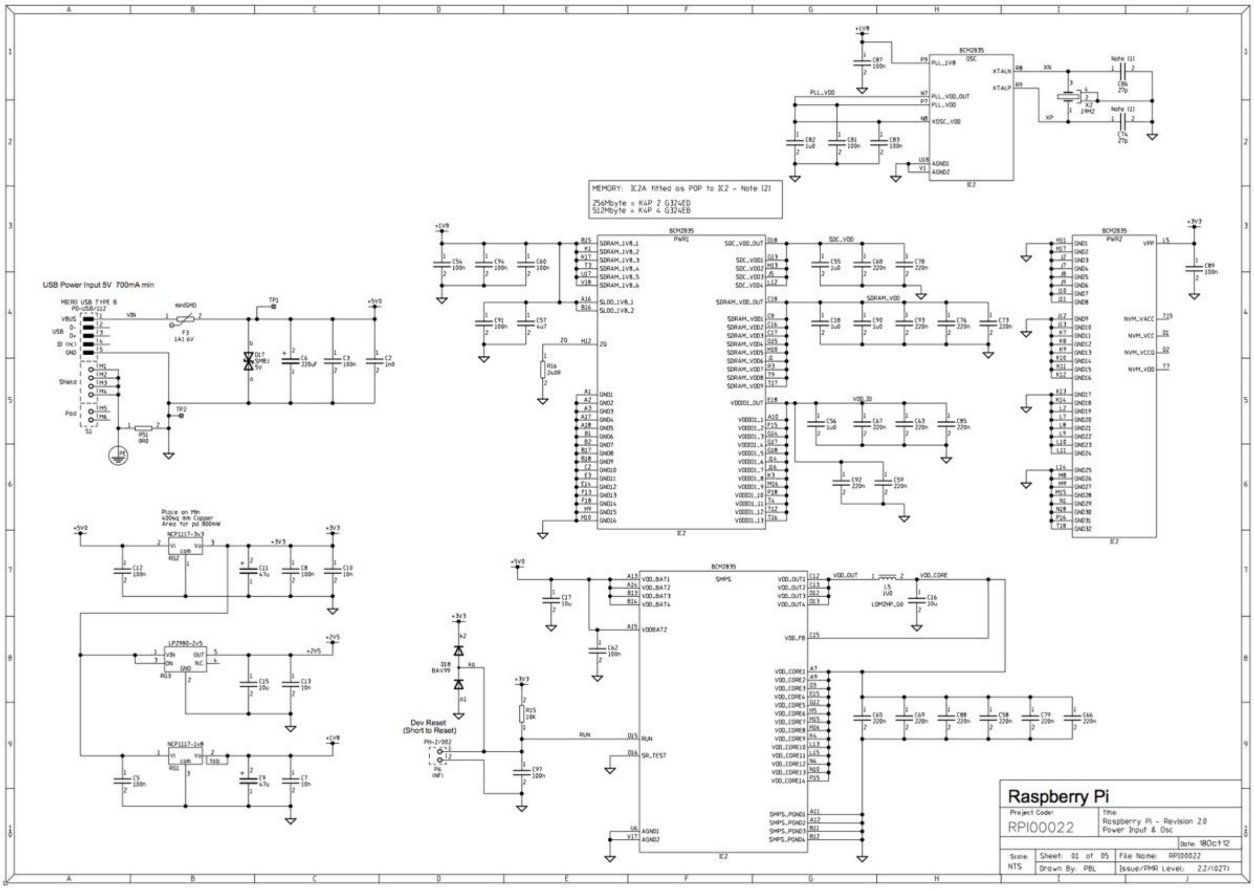

Fig 3.4

3.2 MPU6050 Accelerometer

The MPU-6050 is the first and only 6-axis motion tracking system in the world created to meet the needs of smartphones, tablets, and wearable sensors for low power, low cost, and excellent performance.

Micro Electro-Mechanical Systems (MEMS) called the MPU6050 have a 3-axis accelerometer and a 3-axis gyroscope. This makes it easier for us to measure a system's or object's acceleration, velocity, orientation, displacement, and many other motion-related parameters. Additionally, this module contains a (DMP) Digital Motion Processor, which is strong enough to do complex calculations and free up the Microcontroller's time.

Micro Electro-Mechanical System (MEMS) MPU6050 has a three-axis gyroscope and an accelerometer. It aids in the measurement of motion-related characteristics such as velocity, direction, acceleration, and displacement.

Digital Motion Processor (DMP), a component of MPU6050, has the ability to resolve difficult calculations.

A hardware 16-bit analogue to digital converter is part of the MPU6050. This feature allows it to simultaneously capture three-dimensional motion.

This module can be used with a well-known microcontroller like the Raspberry PI because of its widely available and well-known characteristics. MPU6050 will be a fantastic option for us if we need a sensor to regulate the movements of our drone, self-balancing robot, RC cars, or something similar.

So, in order to detect the motions, I utilised this programme. For communication with Raspberry PI, this module interfaces using the I2C module. The key advantage of the less priced MPU6050 is that it combines with an accelerometer and gyro without much difficulty.

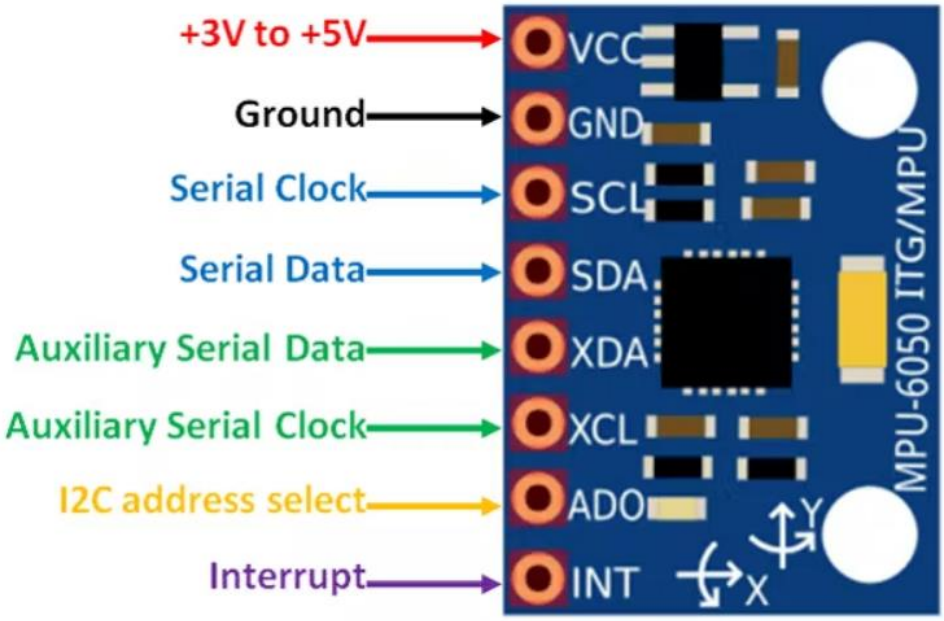

Fig 3.5

For this project, we will be using 4 pins they are VCC, SDA, SCL and Ground. Where VCC provides power for the module, can be +3V to +5V. Typically +5V is used, SDA is used for transferring Data through I2C communication, SCL is used for providing clock pulse for I2C Communication and Ground is used to connect to the ground of system.

This device is used to detect the motions of the vehicle or the person. Uneven acceleration means that the person has met with accident.

MPU6050 front side:

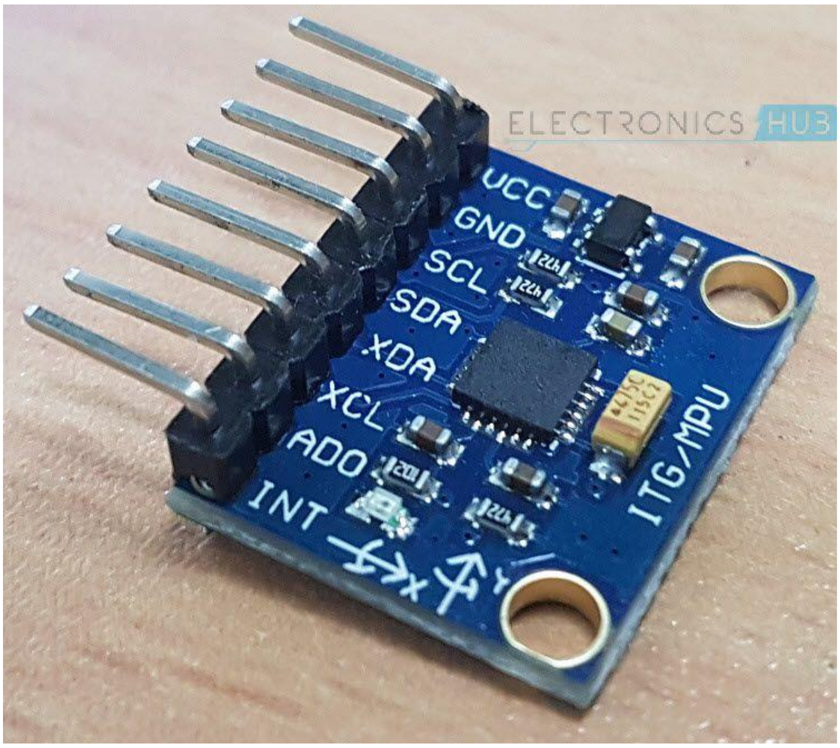

Fig 3.6

MPU6050 backside:

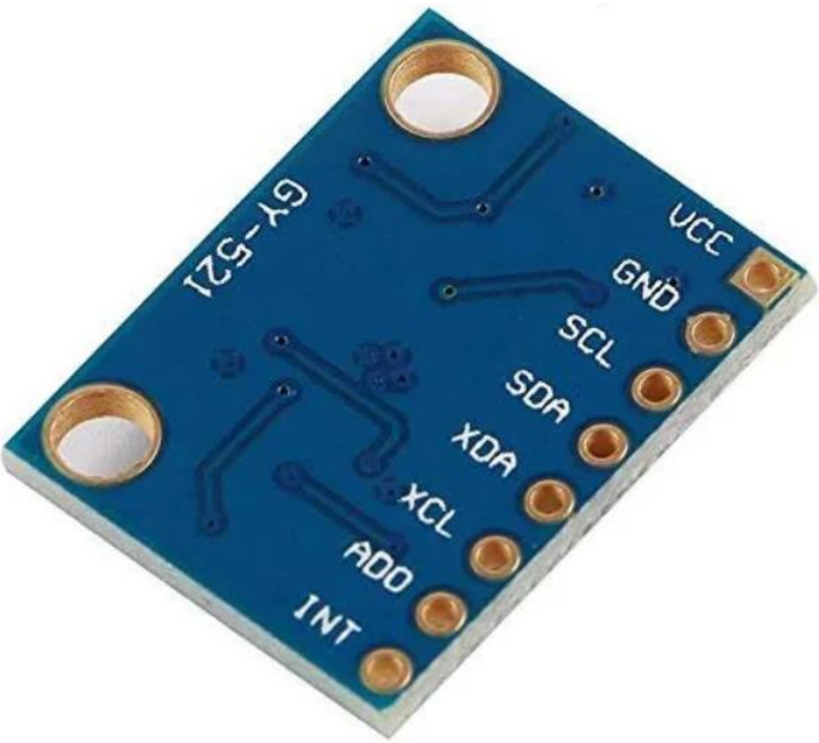

Fig 3.7

MPU6050 schematic diagram:

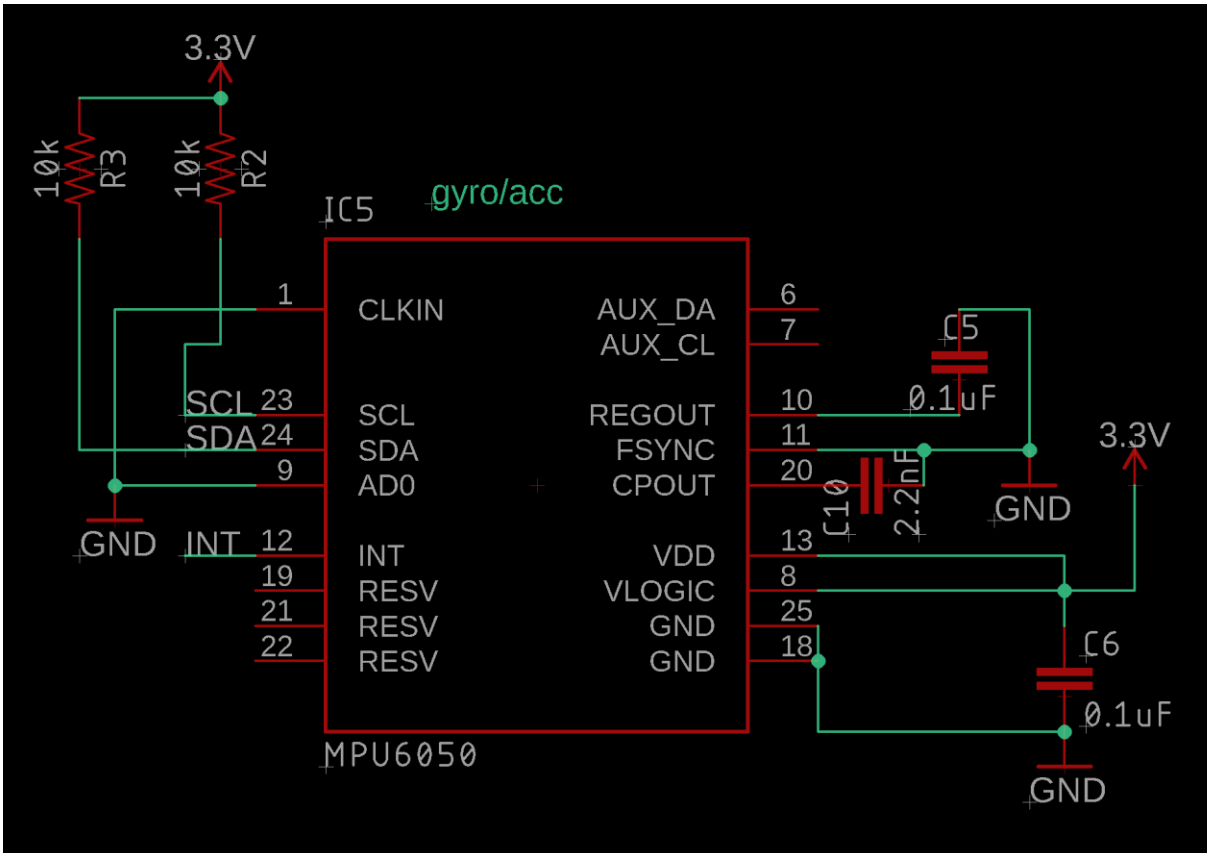

Fig 3.8

3.3 NEO 6M GPS Module

The NEO-6M GPS module, which can locate places everywhere in the world and track 22 satellites. It can be a fantastic starting point for anyone wishing to enter the GPS industry.

A GPS chip from U-blox NEO-6M is the module's brain. The chip is less than the size of a postage stamp, but it contains a surprising amount of functionality. It has a maximum tracking sensitivity of -161 dB and can monitor up to 22 satellites across 50 channels while using only 45 mA of current.

It can update its location five times in a second with a 2.5m horizontal position precision, which is faster than conventional GPS units. Additionally, the Time-To-First-Fix (TTFF) of the U-blox 6 positioning engine is less than one second.

Power Save Mode (PSM) is one of the chip's best features. This enables a decrease in system power usage by selectively turning on and off specific receiver components. As a result, the module now uses only 11mA, which makes it appropriate for power-sensitive applications like GPS wristwatches.

The NEO-6M GPS chip's necessary data pins are divided up onto 0.1″ pitch headers. It has the pins required for UART communication with the microcontroller. With a default baud rate of 9600, the module supports baud rates ranging from 4800bps to 230400bps.

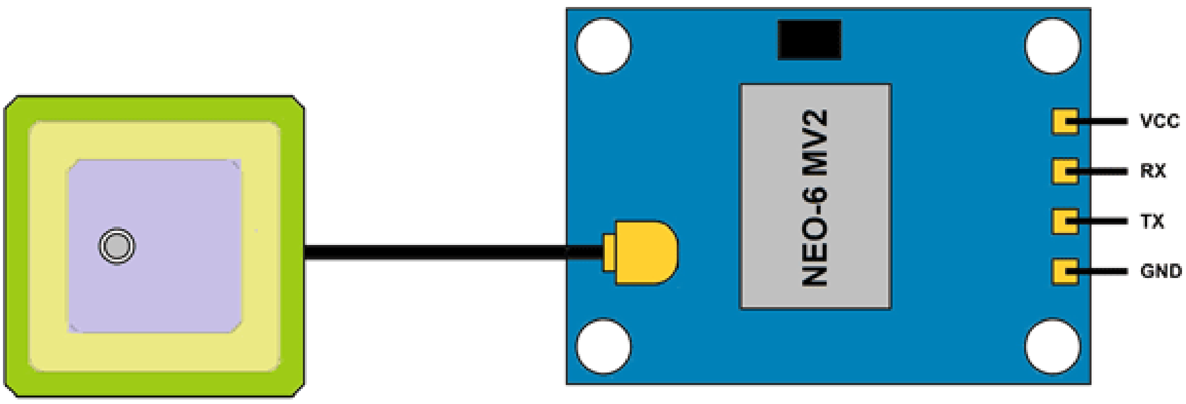

Fig 3.9

One of the more well-liked GPS modules on the market, this one is also reasonably priced. Its location data is precise enough to meet the needs of the majority of applications. Its inclusion in tablet and smart phone designs also testifies to its effectiveness. This module is well-known among engineers and developers alike who want to work on navigation-related applications.

For this project, we will be using 3 of the pins, they are VCC, TX and GND, where VCC positive power pin, TX is UART transmit pin and GND is ground.

Navigation is the principal purpose for which it is utilised. The module only verifies its position on the planet and outputs data, including its longitude and latitude. It belongs to a group of standalone GPS receivers powered by the powerful u-blox 6 positioning engines. These adaptable and reasonably priced receivers come in a small (16 x 12.2 x 2.4 mm) design and provide a wide range of connectivity choices. NEO-6 modules are perfect for battery-operated mobile devices with very limited budget and space restrictions due to their tiny architecture, power, and memory options. Even in the most difficult environments, the NEO-6MV2 performs excellently because to its innovative design.

This device is used to detect the exact location of the accident spot.

NEO 6M front side:

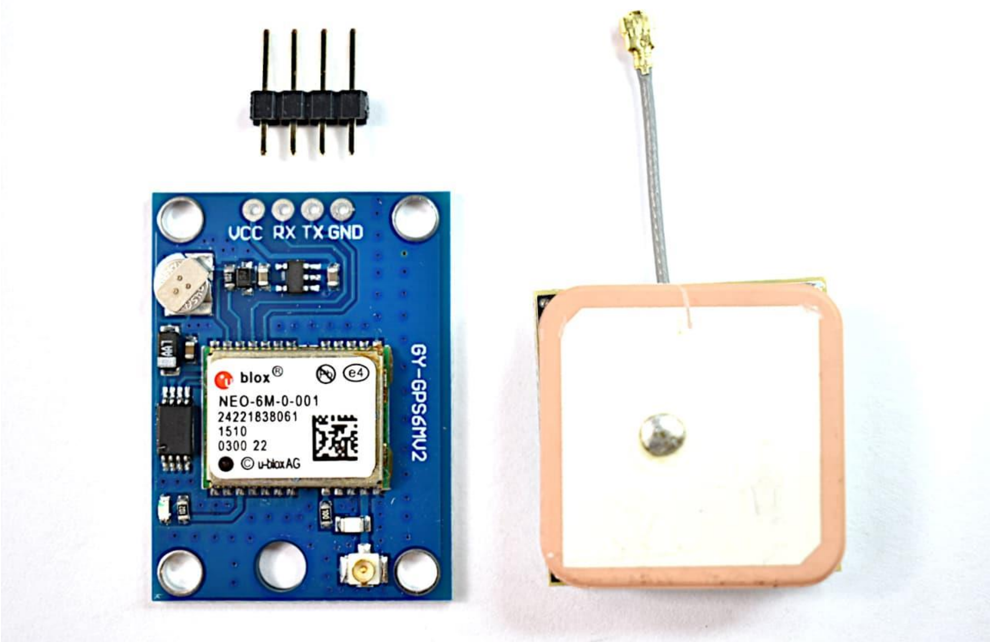

Fig 3.10

NEO 6M back side:

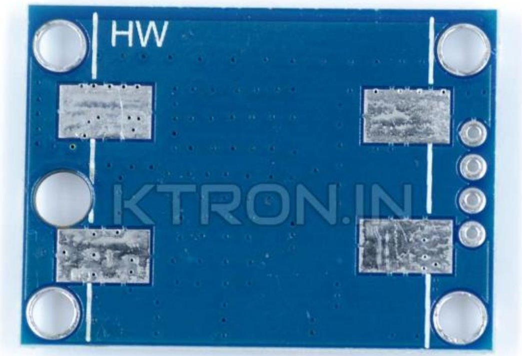

Fig 3.11

NEO 6M schematic diagram:

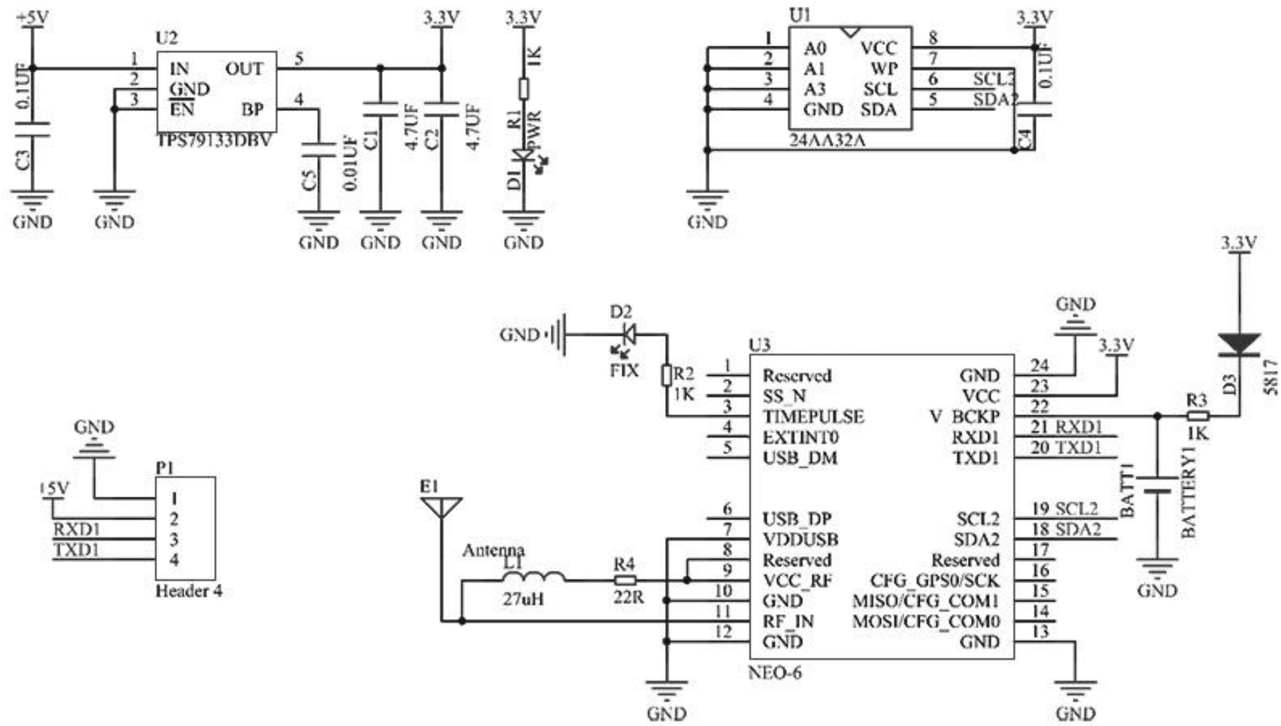

Fig 3.12

Chapter 4

CONNECTIONS AND SETUP

4.1 Raspberry Pi

A power source and an SD card with the necessary Operating System are required for the Raspberry Pi.

You need a screen and a cable to connect the screen and your Raspberry Pi in order to display the Raspberry Pi OS desktop environment. The display could be a computer or television screen. Raspberry Pi may use the screen's built-in speakers to play music if they are present.

We will be using raspberry pi imager to install the required OS.

Fig 4.1

For the Raspberry Pi Imager to download the OS of our choice the first time, we must be online. The OS will then be saved for usage offline in the future. The Raspberry Pi imager will always provide us with the most recent version because it is online for future purposes.

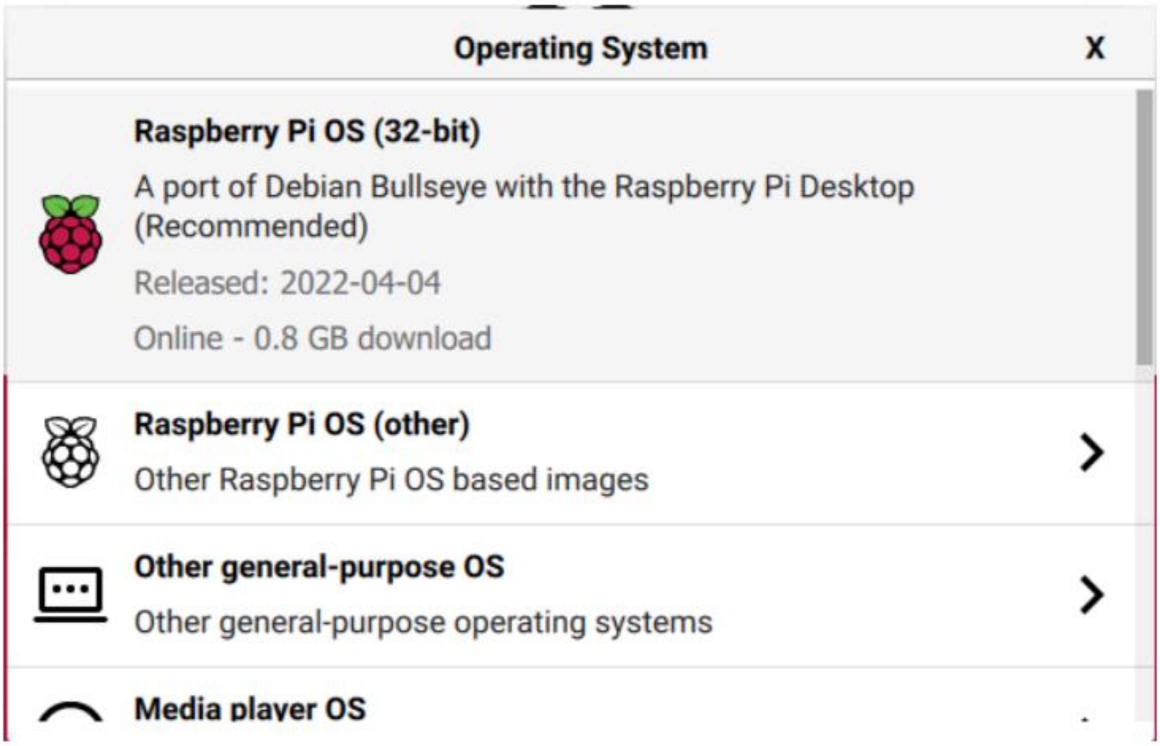

Fig 4.2

After installing the required files in the SD card, we can mount the SD card to raspberry pi, then power on the pi. The pi will be ready to use.

4.2 MPU6050 Accelerometer

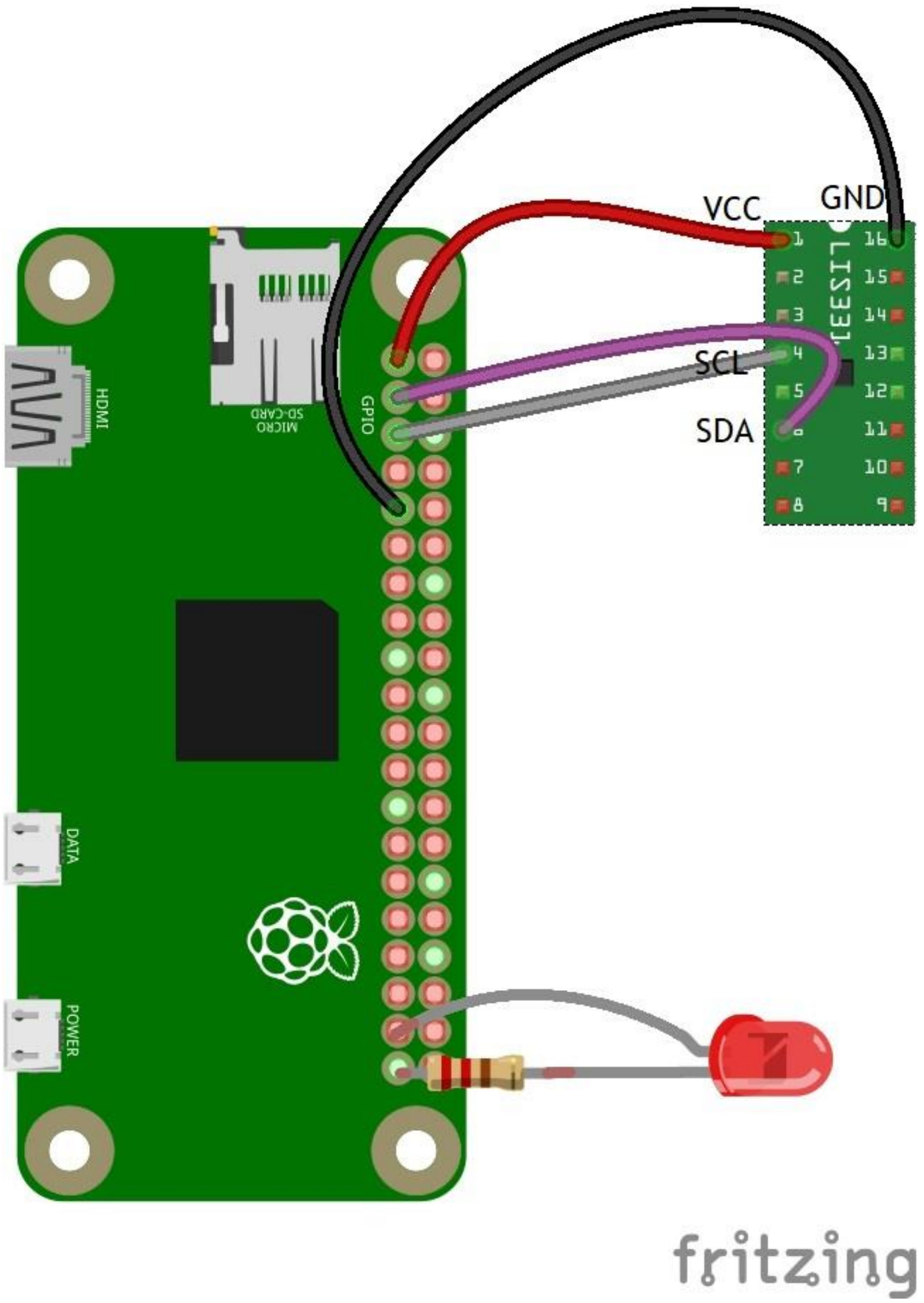

Fig 4.3

Connections are fairly simple. I started by connecting VCC pin to the 5V output on the raspberry pi and connected GND to ground.

We are still working with the I2C communication pins at this point. So, for the I2C communication, I have connected MPU6050 SDA and SCL to SDA, SCL of raspberry pi.

The 3-axis acceleration values will be sent to the raspberry pi.

To start the setup, we just need to enable the I2C in the interfacing options of **sudo raspi-config** command, and then **sudo reboot**.

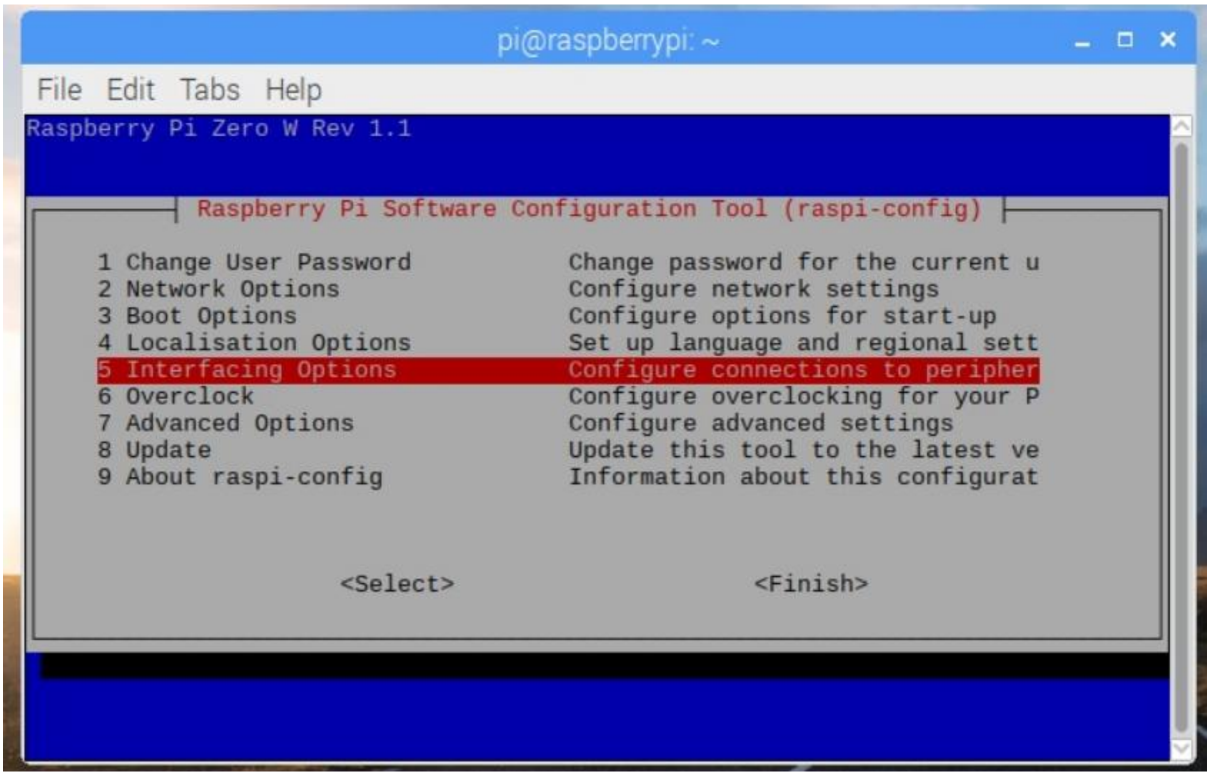

Fig 4.4

After this, we need to install raspberry pi GPIO library using the command **sudo pip install RPi.GPIO**

Finally, we need to use the command **sudo apt-get install python-smbus** to install the smbus library on the Raspberry Pi.

After following all the above process, the MPU6050 module is ready to be put to work.

4.3 NEO 6M GPS Module

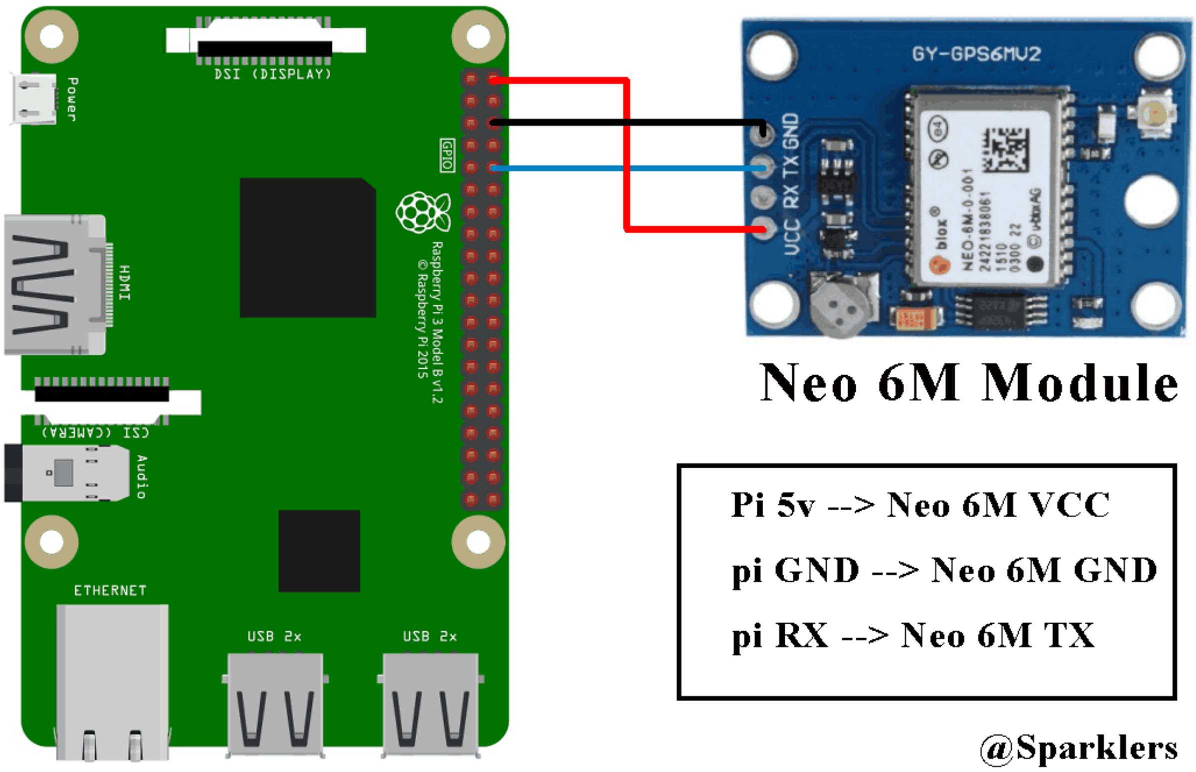

Fig 4.5

Connect VCC of GPS module to Power Supply Pin no2 (5V) of Raspberry Pi.

Connect Tx (Transmitter Pin) of GPS module to Pin no10 of Raspberry Pi.

The Ground pin of GPS module should be connected to the ground pin of raspberrypi that id pin no 6.

The transmitter pin of the GPS module is connected to the Receiver pin of raspberry pi we need the data from the GPS module to be received and saved.

Set Up the UART in Raspberry Pi

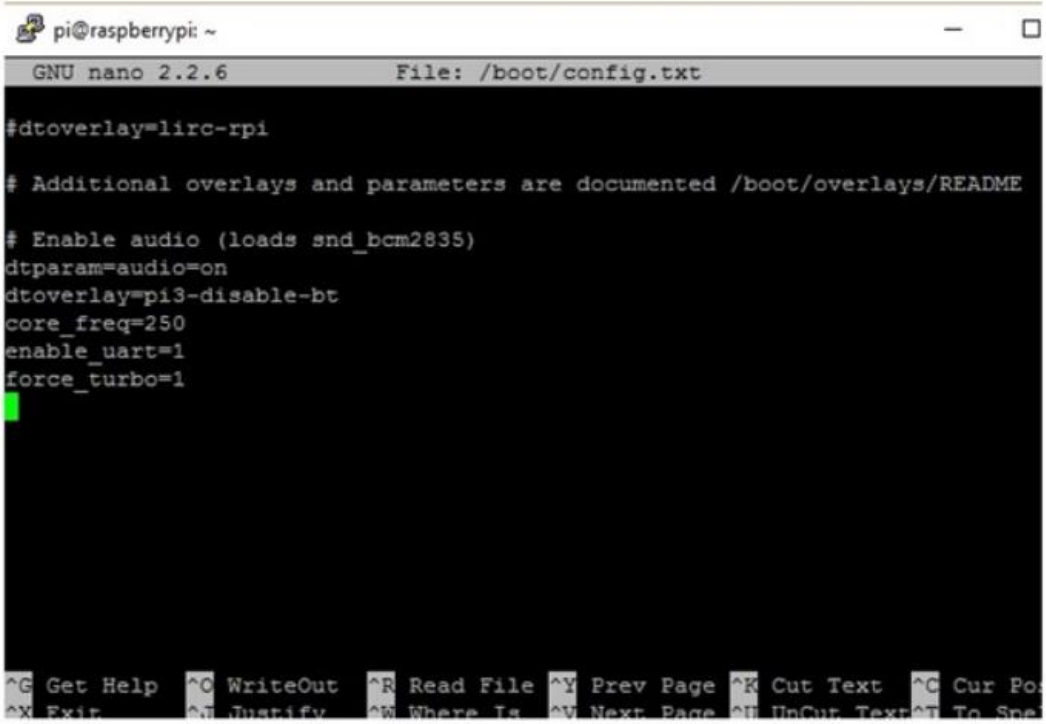

Fig 4.6

The second step under this UART setup section is to edit the boot/cmdline.txt, Replace the content with: **dwc_otg.lpm_enable=0 console=tty1 root=/dev/mmcblk0p2 rootfstype=ext4 elevator=deadline fsck.repair=yes root wait quiet splash plymouth.ignore-serial-consoles**

Up next, we need to disable the Raspberry Pi Serial Getty Service using the following commands:

sudo systemctl stop serial-getty@ttyAMA0.service

sudo systemctl disable serial-getty@ttyAMA0.service

We need to use the pynmea2 python library to parse the received NMEA data.

So, to install the pynmea2 use the command **pip install pynmea2**.

After following all the above process, the GPS module is ready to be put to work.

Chapter 5

IMPLEMENTATION AND OUTPUT

5.1 MPU6050 Accelerometer

In Python Program, we have imported some required library like time, smbus, and GPIO.

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

After this, we need to take some register address to configure MPU6050 and for getting values from the same. We have also taken some variables for calibrating and initializing bus for I2C.

```
PWR_M   = 0x6B
DIV   = 0x19
CONFIG       = 0x1A
GYRO_CONFIG  = 0x1B
INT_EN   = 0x38
ACCEL_X = 0x3B
ACCEL_Y = 0x3D
ACCEL_Z = 0x3F
GYRO_X  = 0x43
GYRO_Y  = 0x45
GYRO_Z  = 0x47
TEMP = 0x41
bus = smbus.SMBus(1)

Device_Address = 0x68   # device address
AxCal=0
AyCal=0
AzCal=0
GxCal=0
```

GyCal=0
GzCal=0

After this, we need to initialize the MPU6050 Module

```
def InitMPU():
    bus.write_byte_data(Device_Address, DIV, 7)
    bus.write_byte_data(Device_Address, PWR_M, 1)
    bus.write_byte_data(Device_Address, CONFIG, 0)
    bus.write_byte_data(Device_Address, GYRO_CONFIG, 24)
    bus.write_byte_data(Device_Address, INT_EN, 1)
    time.sleep(1)
```

After this, we need to write some functions to read values from MPU6050. Given function is used to read data from MPU6050

```
def readAxes(addr):
        data0 = bus.read_byte_data(addr, ACC_XOUT_H+1)
        data1 = bus.read_byte_data(addr, ACC_XOUT_H)
        data2 = bus.read_byte_data(addr, ACC_YOUT_H+1)
        data3 = bus.read_byte_data(addr, ACC_YOUT_H)
        data4 = bus.read_byte_data(addr, ACC_ZOUT_H+1)
        data5 = bus.read_byte_data(addr, ACC_ZOUT_H)
        #Combine the two bytes and leftshit by 8
        x = data0 | data1 << 8
        y = data2 | data3 << 8
        z = data4 | data5 << 8
        #in case overflow
        if x > 32767 :
                x -= 65536
        if y > 32767:
                y -= 65536
        if z > 32767 :
                z -= 65536
```

```
    #Calculate the two's complement as indicated in the datasheet
        x = ~x
        y = ~y
        z = ~z
        return x, y, z
```

We get raw values from the above function and then we conert them to the G force using the below function.

```
#Function to calculate g-force from acceleration data
def convertToG(maxScale, xAccl, yAccl, zAccl):
    #Caclulate "g" force based on the scale set by user
    #Eqn: (2*range*reading)/totalBits (e.g. 48*reading/2^16)
        X = (2*float(maxScale) * float(xAccl))/(2**16);
        Y = (2*float(maxScale) * float(yAccl))/(2**16);
        Z = (2*float(maxScale) * float(zAccl))/(2**16);
        return X, Y, Z
```

5.2 NEO 6M GPS Module

```
import serial
Import time
import string import pynmea2
while True: port=“/dev/ttyAMAO”
ser=serial.Serial(port,baudrate=9600,timeout=0.5)
dataout =pynmea2.NMEAStreamReader()
newdata=ser.readline()
if newdata[0:6]==“$GPRMC”:
newmsg=pynmea2.parse(newdata)
lat=newmsg.latitude
lng=newmsg.longitude
gps=“Latitude=” +str(lat) + “and Longitude=” +str(lng)
print(gps)
```

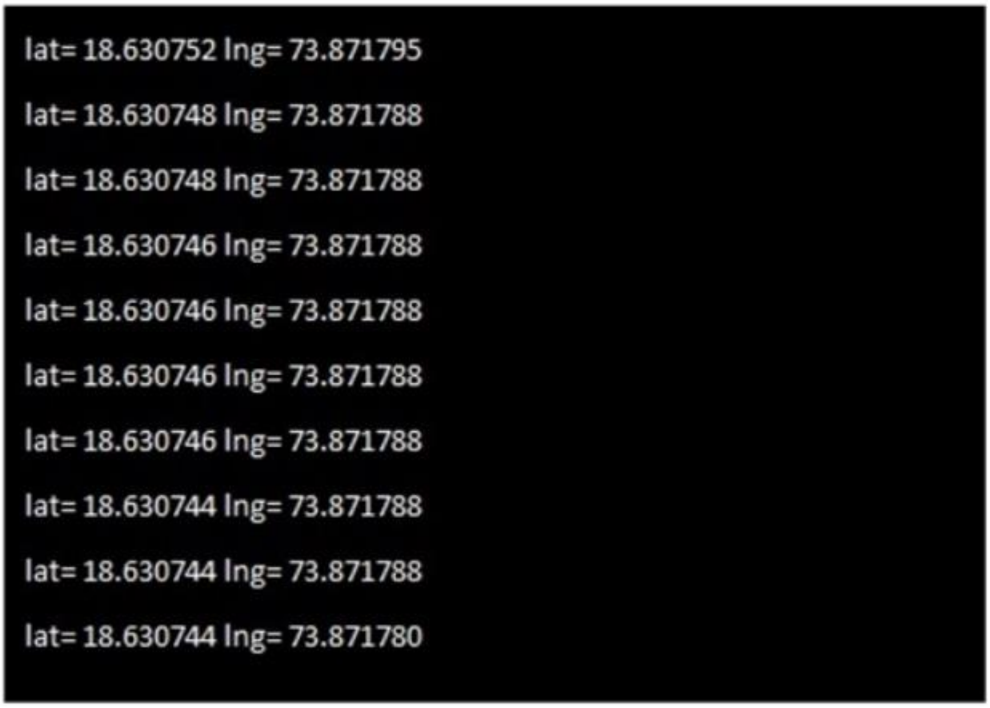

Fig 5.1

5.3 Sending message to mobile using Twillio

To implement this part, we need to create an account in twillio and we need to add our emergency contacts in twillio to whom this message needs to be sent.

Next, we need to install the Twillio library in the raspberrypi using the following command,
pip install twillio

```
from twilio.rest import Client

account_sid ="XXXXXXX" # Put your Twilio account SID here

auth_token ="XXXXXXX" # Put your auth token here

client = Client(account_sid, auth_token)

message = client.api.account.messages.create(

            to="+#####", # Put your cellphone number here

            from_="+######", # Put your Twilio number here

            body="This is my message that I am sending to my phone!")
```

Chapter 6

COMPLETE IMPLEMENTATION AND OUTPUT

6.1 Code

```
import smbus
import serial
from time import sleep
import sys



from time import sleep

#If gps not working, alternative
“”””
import json
from urllib.request import urlopen
“”””

from twilio.rest import Client

ser = serial.Serial (“/dev/ttyAMA0”)
gpgga_info = “$GPGGA,”
GPGGA_buffer = 0
NMEA_buff = 0
flag = 0
loc=””

addr = 0x68

maxScale = 24
```

```
PWR_MGMT_1 = 0x6B
SMPLRT_DIV = 0x19
CONFIG = 0x1A
INT_ENABLE = 0x38
ACC_XOUT_H = 0x3B
ACC_YOUT_H = 0x3D
ACC_ZOUT_H = 0x3F

bus = smbus.SMBus(1)

def MPU_Init():
	bus.write_byte_data(addr,SMPLRT_DIV, 7)
	bus.write_byte_data(addr,PWR_MGMT_1, 1)
	bus.write_byte_data(addr,CONFIG, 0)
	bus.write_byte_data(addr,INT_ENABLE, 1)



def convert_to_degrees(raw_value):
	decimal_value = raw_value/100.00
	degrees = int(decimal_value)
	mm_mmmm = (decimal_value – int(decimal_value))/0.6
	position = degrees + mm_mmmm
	position = “%.4f” %(position)
	return position

def readAxes(addr):
	data0 = bus.read_byte_data(addr, ACC_XOUT_H+1)
	data1 = bus.read_byte_data(addr, ACC_XOUT_H)
	data2 = bus.read_byte_data(addr, ACC_YOUT_H+1)
	data3 = bus.read_byte_data(addr, ACC_YOUT_H)
	data4 = bus.read_byte_data(addr, ACC_ZOUT_H+1)
```

```
        data5 = bus.read_byte_data(addr, ACC_ZOUT_H)
        #Combine the two bytes and leftshit by 8
        x = data0 | data1 << 8
        y = data2 | data3 << 8
        z = data4 | data5 << 8
        #in case overflow
        if x > 32767 :
            x -= 65536
        if y > 32767:
            y -= 65536
        if z > 32767 :
            z -= 65536
    #Calculate the two’s complement as indicated in the datasheet
        x = ~x
        y = ~y
        z = ~z
        return x, y, z

#Function to calculate g-force from acceleration data
def convertToG(maxScale, xAccl, yAccl, zAccl):
    #Caclulate “g” force based on the scale set by user
    #Eqn: (2*range*reading)/totalBits (e.g. 48*reading/2^16)
        X = (2*float(maxScale) * float(xAccl))/(2**16);
        Y = (2*float(maxScale) * float(yAccl))/(2**16);
        Z = (2*float(maxScale) * float(zAccl))/(2**16);
        return X, Y, Z

def isDanger(x, y, z):

        if abs(x) > 20 or abs(y) > 20 or abs(z) > 20:
            “””
            url = ‘http://ipinfo.io/json’
            response = urlopen(url)
```

```
		data = json.load(response)
		“””


		try:
			while True:
				received_data = (str)(ser.readline()) #read NMEA string
received
				GPGGA_data_available = received_data.find(gpgga_info)
#check for NMEA GPGGA string
				if (GPGGA_data_available>0):
					GPGGA_buffer =
received_data.split(“$GPGGA,”,1)[1]  #store data coming after “$GPGGA,” string
					NMEA_buff = (GPGGA_buffer.split(‘,’))
					#nmea_time = []
					nmea_latitude = []
					nmea_longitude = []
					#nmea_time = NMEA_buff[0]			#extract time
from GPGGA string
					nmea_latitude = NMEA_buff[1]		#extract
latitude from GPGGA string
					nmea_longitude = NMEA_buff[3]		#extract
longitude from GPGGA string
					#print(“NMEA Time: “, nmea_time,’\n’)
					lat = (float)(nmea_latitude)
					lat = convert_to_degrees(lat)
					longi = (float)(nmea_longitude)
					longi = convert_to_degrees(longi)
					loc = “NMEA Latitude:”+ lat+ “ NMEA Longitude:”+
longi
					flag=1
					if (flag==1):
						break
```

```
                except KeyboardInterrupt:
                        sys.exit(0)

                msg = "Person had met with an Accident at location "+loc
                print(msg)

                accSID = "AC821bb7e081d4aa86fb95afe79189b26f"
                authToken = "3dbe85b32f9cfbd3e57be2e1e6a8fa01"

                client = Client(accSID, authToken)
                message =
client.api.account.messages.create(to="+919949286689",from_="+17315357765",body=msg
)
                return True
        else:
                return False

def main():
   #Run this program unless there is a keyboard interrupt
        print ("Starting stream")
        while True:
     #initialize LIS331 accelerometer
     #initialize(addr, 24)

                MPU_Init()




     #Get acceleration data for x, y, and z axes
                xAccl, yAccl, zAccl = readAxes(addr)
     #Calculate G force based on x, y, z acceleration data
                x, y, z = convertToG(maxScale, xAccl, yAccl, zAccl)
     #Determine if G force is dangerous to human body & take proper action
```

```
            if isDanger(x, y, z):
                break



    #print G values (don’t need for full installation)
            print(“Acceleration in X-Axis : %d” %x)
            print(“Acceleration in Y-Axis : %d” %y)
            print(“Acceleration in Z-Axis : %d” %z)
            print(“\n”)

    #Short delay to prevent overclocking computer
            sleep(0.2)

if __name__ ==”__main__”:
    main()
```

6.2 Output

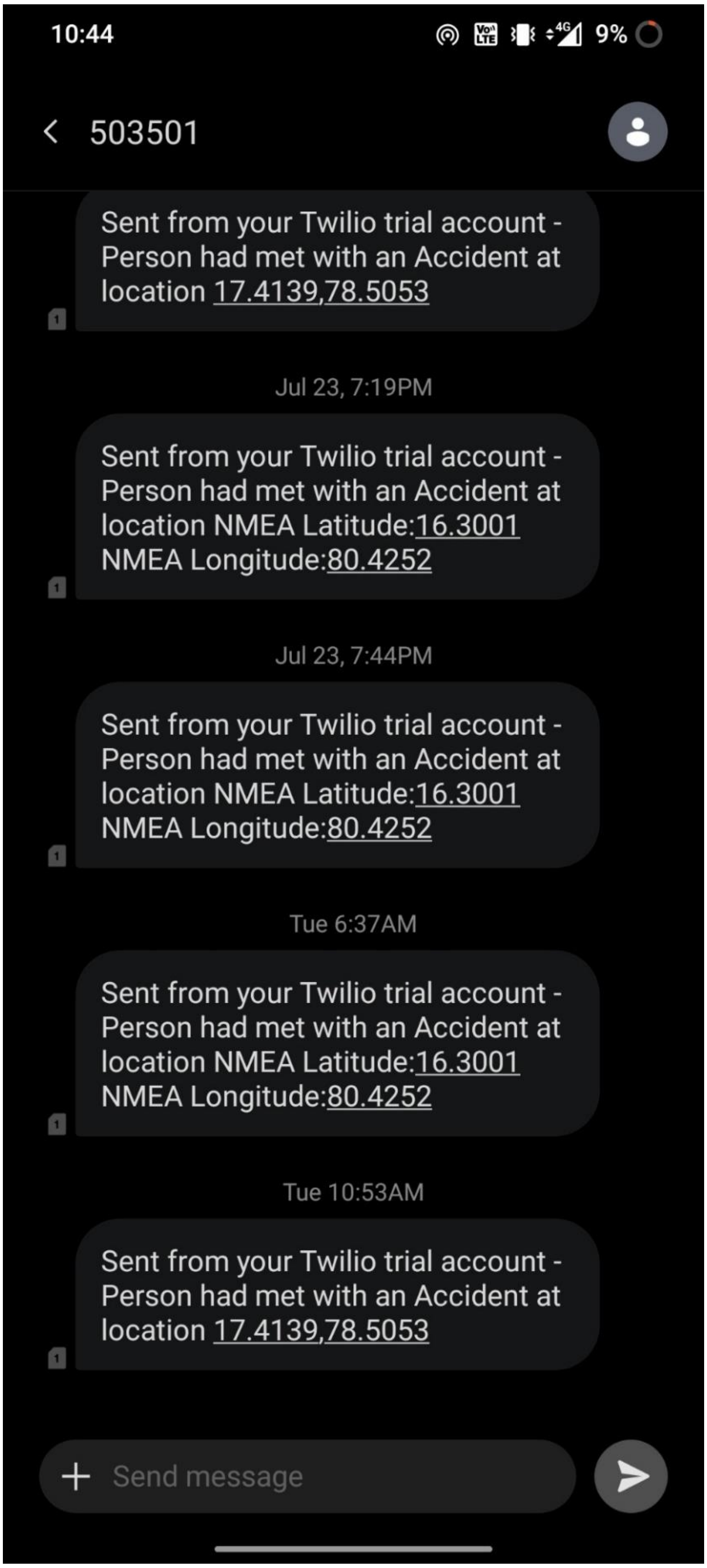

Fig 6.1

Chapter 7

CONCLUSION AND FUTURE WORK

7.1 Conclusion

The suggested programmed accident detection alert may save the lives of those who were involved in accidents. The suggested approach is quite simple to comprehend, and even a non-specialist can utilise it without any issues. The system is made up of hardware and code chunks. The equipment unit is installed in the vehicle or on the helmet of biker and incorporates sensors for accident detection, including an accelerometer and a GPS module. However, the programming component is an Android app that was made available for drivers' smartphones and is used to obtain a point-by-point map. In general, this system offers minimal costs, security, and ease of use. The technique used in this work lowers the number of accident-related fatalities.

Different accident detection and notification systems have been researched and developed over time. This effort aims to outline their benefits and drawbacks using a comparative analysis. The project has a number of benefits, but there are some drawbacks as well. For instance, the issue of false alarms is a significant disadvantage.

7.2 Future Scope

The suggested system focuses on accident detection. However, this might be prolonged by giving the accident victims medicine on the scene. By developing alarm systems that can stop a vehicle in order to prevent accidents, we can also prevent accidents as technology advances. We may also modify the system so that it automatically alerts both the hospital and the local police station. Additionally, this project can be used to automobiles other than bikes.

For isolated places, a more dependable system might be created. In order to provide faster medical care, GPS and Wi-Fi should be combined in a way that the prototype itself automatically alerts the local police station and hospitals. To further enhance the driver's security, the device can be interfaced with the vehicle's air bag system. The system might also be interfaced with a camera module.

Chapter 8

REFERENCES

Dr.C.K.Gomathy,C K Hemalatha, Article: A Study On Employee Safety And Health Management International Research Journal Of Engineering And Technology (Irjet)- Volume: 08 Issue: 04 | Apr 2021

https://www.youtube.com/watch?v=hzNJtNnvi5A&ab_channel=jenfoxbot

https://www.electronicwings.com/raspberry-pi/mpu6050-accelerometergyroscope-interfacing-with-raspberry-pi#:~:text=We%20can%20interface%20MPU6050%20module,I2C%20functions%20for%20Raspberry%20Pi.

https://www.youtube.com/watch?v=N8fH0nc9v9Q&t=853s&ab_channel=SPARKLERS%3AWeAreTheMakers

https://www.youtube.com/watch?v=Oi37lg_ciJ8&ab_channel=AlexanderBaran-Harper

https://nevonprojects.com/accident-identification-alert-system/

https://www.researchgate.net/publication/277848807_Vehicle_Accident_Automatic_Detection_and_Remote_Alarm_Device