A Project Report

On

ACCIDENT DETECTION AND NOTIFICATION SYSTEM

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

BACHELOR OF TECHNOLOGY
IN
DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

By

AMIREDDY SIVANAGIREDDY
(218H1A0403)

Under the Esteemed Guidance of

Mr. P.KOTESWARA RAO, M.Tech
DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

MVR College of Engineering & Technology

(AUTONOMOUS)

(Approved by AICTE, Permanently Affiliated to JNTUK, Certified By ISO: 9001-2008,
Accredited by NBA (CIVIL, CSE) & Accredited by NAAC with ‘A’ Grade)
Paritala, NTR Dist., PIN 521180 A.P. India.

2024-2025

A Project Report

On

ACCIDENT DETECTION AND NOTIFICATION SYSTEM

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

BACHELOR OF TECHNOLOGY
IN
DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

By

AMIREDDY SIVANAGIREDDY
(218H1A0403)

Under the Esteemed Guidance of

Mr. P.KOTESWARA RAO, M.Tech
DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

MVR College of Engineering & Technology

(AUTONOMOUS)

(Approved by AICTE, Permanently Affiliated to JNTUK, Certified By ISO: 9001-2008,
Accredited by NBA (CIVIL, CSE) & Accredited by NAAC with 'A' Grade)
Paritala, NTR Dist., PIN 521180 A.P. India.
2024-2025

MVR College of Engineering & Technology

(AUTONOMOUS)

(Approved by AICTE, Permanently Affiliated to JNTUK, Certified By ISO: 9001-2008, Accredited by NBA (CIVIL, CSE) & Accredited by NAAC with 'A' Grade)

Paritala, NTR Dist., PIN 521180 A.P. India.

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

CERTIFICATE

This is to certify that the Project Report entitled "**ACCDIENT DETECTION AND NOTIFICATION SYSTEM**" is being submitted by "AMIREDDY.SIVANAGIREDDY(218H1A0403)" in partial fulfillment for the award of the Degree of Bachelor of Technology in Department of Electronics and Communication Engineering of MVR College of Engineering & Technology, for the record of bonafide work carried out by them.

Mr.P.KOTESWARA RAO	**Dr.S.VENKATASWAMI**
Assistant Professor	**Associate Professor**
Project Supervisor	**Head of the Department**

Examiner

ACKNOWLEDGEMENT

I express my deep sense of gratitude and indebtedness to the **Management & Dr.U.Yedukondalu, Principal** of our college, who gave very good support to me during this work and others who assisted me in this project work.

I also thank **Dr. S.VENKATA SWAMI, Head of the Department of** Electronics and Communication Engineering, who gave very good support to me during this work.

I also thank my **Project Supervisor**, **Ms. P.KOTESWARA RA0, Asst. Professor** of Electronics and Communication Engineering **Department, MVR College of Engineering & Technology**, for her valuable guidance during the course of this project work. I am much indebted to her for suggesting a challenging and interactive project and her valuable advice at every stage of this work. I am very much thankful to her for her coordination in this regard.

Similarly, I am grateful to my friends I have worked with and consider myself extremely privileged and fortunate to work with such honest, hard working and loyal friends.

I wish my warm and grateful thanks to the staff of our department of **MVR College of Engineering & Technology** for the assistance for their continuous encouragement in completing my work successfully.

Finally I thank one and all who directly or indirectly helped me in completing project successfully.

AMIREDDY SIVANAGIREDDY
(218H1A0403)

DECLARATION

I hereby declare that this project entitled "ACCDIENT DETECTION AND NOTIFICATION SYSTEM" has been undertaken by me and this work has been submitted to **MVR college of Engineering & Technology affiliated to JNTUK, Kakinada**, in partial fulfillment of the requirements for award of the degree of **Bachelor of Technology** in **Department of Electronics and Communication Engineering.**

I further declare that this project work has not been submitted in full or part for the award of any degree of this in any other educational institutions.

AMIREDDY SIVANAGIREDDY
(218H1A0403)

LIST OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ABSTRACT

This project aims to develop an IoT-based accident detection and notification system using Arduino Uno, an accelerometer sensor, and a crash sensor. When an accident is detected, the system sends real-time notifications through a NodeMCU module to the ThingSpeak cloud. The buzzer and LCD provide alerts, while a DC motor represents the vehicle, which stops upon detecting an accident. The system also transmits GPS location data for emergency response.

An accident detection system is a technological setup that utilizes sensors like accelerometer, GPS, and sometimes cameras to continuously monitor a vehicle's dynamics, detecting sudden changes in acceleration, impact forces, or unusual movements that could indicate an accident, and then automatically triggers an alert to emergency services or designated contacts with the location of the incident upon detection. Using the sensors such as accelerometer, GPS, and cameras, the system continuously monitors the vehicle's dynamics and surroundings. Upon detecting an abnormal event indicative of a potential accident, such as sudden deceleration, collision impact, or rollover, this system triggers an alert mechanism.

CHAPTER – 1

INTRODUCTION

Nowadays, the rate of accidents has increased rapidly. Due to employment, the usage of vehicles like cars, bikes have increased, because of this reason the accidents can happen due to over speed. People are going under risk because of their over speed, due to unavailability of advanced techniques, the rate of accidents can't be decreased. To reduce the accident rate in the country this paper introduces a solution. Automatic accident detection and alert systems are introduced. The main objective is to control the accidents by sending a message to the registered mobile, hospital and police station using wireless communications techniques. When an accident occurs in a city or any place, the message is sent to the registered mobile through GSM module in less time.

The rapid growth in the number of vehicles and expansion of road networks has led to a significant increase in road accidents across the globe. According to the World Health Organization (WHO), road traffic injuries are one of the leading causes of death, especially among young people aged between 15 and 29. While some accidents result in minor injuries, others can be fatal — particularly when there is a delay in providing medical assistance. These delays often occur when accidents go unreported or unnoticed, especially in remote areas or during nighttime hours.

To tackle this challenge, technology-based solutions are being developed that focus on reducing emergency response time. One such solution is the Accident Detection and Notification System. This system is designed to automatically detect a road accident and immediately alert emergency responders, family members, and nearby hospitals with the exact location of the incident.

This system can be implemented using various technologies including IoT (Internet of Things) hardware or smartphone-based applications. In a hardware-based system, sensors such as accelerometers, gyroscopes, and vibration sensors are used to detect sudden motion or impact. These sensors are connected to a microcontroller (like Arduino or Raspberry Pi) which processes the data and determines whether an accident has occurred. If confirmed, the system sends a notification containing GPS coordinates to emergency contacts via GSM or internet-based services.

Alternatively, the system can be implemented as a mobile application that uses in-built sensors and location services of a smartphone. In both cases, the goal is to ensure that help reaches the accident location as quickly as possible, thus minimizing fatalities and improving the chances of survival.

CHAPTER – 2

LITERATURE SURVEY

2.1 Introduction

A literature survey is conducted to understand the existing technologies, systems, and research efforts made in the field of accident detection and emergency notification. It helps identify the strengths and limitations of current systems and lays the groundwork for proposing a more efficient and improved solution.

Several researchers and developers have explored various approaches involving **IoT devices**, **mobile applications**, and **sensor-based systems** to detect accidents and automate emergency alerts.

2.2 Existing Research and Systems

1. Smart Vehicle Accident Detection System Using GSM and GPS Modules

This system uses an accelerometer to detect sudden changes in motion or impact. If the measured G-force exceeds a certain threshold, the microcontroller triggers an SMS alert with GPS coordinates to emergency contacts using the GSM module. This approach has been implemented using Arduino or Raspberry Pi platforms.

> **Limitation:** Lacks a mechanism to prevent false alarms; no feedback loop from the user.
>
> **Reference:** International Journal of Advanced Research in Electronics and Communication Engineering (IJARECE), 2019.

2. Android-Based Accident Detection Mobile Application

This approach utilizes the smartphone's built-in sensors (accelerometer, gyroscope, GPS) to monitor driving patterns and detect accidents. Upon detection, the app sends real-time notifications to pre-configured contacts.

> **Advantage:** No need for additional hardware; low-cost solution.
>
> **Limitation:** Depends on mobile network availability and phone battery status.
>
> **Reference:** IEEE International Conference on Smart Technologies and Management for Computing, Communication, Controls, Energy and Materials, 2020.

3. Automatic Accident Detection and Rescue System Using Arduino

In this system, an Arduino microcontroller is connected to a vibration sensor and GPS module. When an accident is detected based on high vibration intensity, an SMS with location data is sent to the nearest hospital and police station.

Advantage: Easy to build and deploy.

Limitation: Fixed threshold for impact detection may lead to false positives or negatives.

Reference: International Journal of Engineering Research & Technology (IJERT), 2018.

4. Intelligent Transportation Systems (ITS)

ITS uses advanced communication technologies and cloud computing to monitor traffic and accidents. Vehicles are embedded with onboard units that communicate with roadside units and cloud services to relay accident data.

Advantage: Scalable and integrates with smart city infrastructure.

Limitation: High cost and requires government-level implementation.

Reference: Transportation Research Board Annual Meeting, 2021.

2.3 Comparative Analysis

System	**Platform**	**Detection Method**	**Notification**	**Advantages**	**Limitations**
Arduino + GSM	Hardware	Accelerometer	SMS	Simple, standalone	No user confirmation
Android App	Mobile	Sensor fusion	SMS / App Notification	Low cost	Dependent on phone state
IoT Cloud-based	IoT + Cloud	Sensor + AI	Cloud alert system	Scalable	Expensive setup
ITS	Infrastructure	Sensors + Cameras	Centralized alert	Smart city integration	Needs public infrastructure

2.4 Research Gap Identified

Many systems lack **false alert prevention mechanisms**.

Most rely on a **single detection method**, which may reduce accuracy.

Few systems incorporate **user confirmation** before alerting emergency services.

Limited use of **AI or machine learning** for smarter accident pattern recognition.

2.5 Conclusion

The literature review highlights various approaches for accident detection and notification, ranging from sensor-based hardware solutions to mobile and cloud-based systems. While many existing solutions provide a solid foundation, there is still a need for an **intelligent, hybrid system** that minimizes false alerts, improves detection accuracy, and ensures **faster emergency response**. The proposed system in this project aims to overcome these limitations by integrating multiple technologies and improving user interaction and notification reliability

CHAPTER – 3

SYSTEM ANALYSIS

3.1 Existing System

In the current scenario, most accident reporting systems rely heavily on **manual reporting** either by the victim or nearby individuals. While some high-end vehicles may be equipped with proprietary systems such as OnStar or eCall, these systems are not universally available due to **cost constraints** and **limited accessibility** in low-budget or older vehicles.

Some existing mobile applications use **smartphone sensors** to detect abrupt motion or impact and send alerts. However, they suffer from limitations such as:

- High **false alarm rates** due to sudden braking or potholes
- **Dependency on internet and battery**
- Inability to function if the phone is damaged during the accident
- Lack of driver state monitoring (e.g., alcohol level, drowsiness)

Hence, the need for a **dedicated, standalone, sensor-based, low-cost, and intelligent accident detection system** is evident.

3.2 Proposed System

The proposed **Accident Detection and Notification System** is designed to be **affordable**, **reliable**, and **autonomous**, capable of detecting various types of vehicular incidents and driver impairments. It uses an **Arduino Uno or Raspberry Pi Pico** as the control unit, integrated with a variety of sensors such as:

- **Accelerometer:** Detects collision impact or sudden motion
- **Ultrasonic Sensor:** Monitors proximity and potential collision
- **Crash Sensor (Limit Switch):** Detects physical impact
- **Alcohol Sensor:** Monitors driver intoxication levels
- **IR Eye Blink Sensor:** Detects drowsiness or unconsciousness

When any emergency condition is detected, the system:

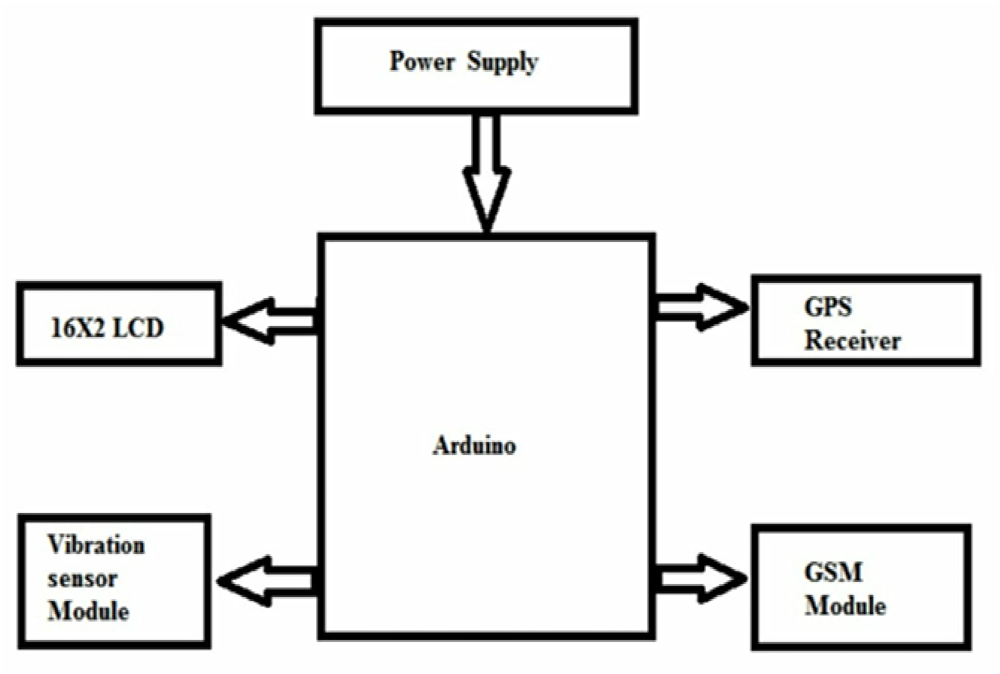
- Activates a **buzzer and displays messages** on the LCD
- Captures **GPS coordinates**
- Sends real-time **notifications via Wi-Fi to the cloud (ThingSpeak)**
- Notifies emergency contacts through a **mobile app**

This enhances **driver safety, accident response time**, and helps reduce fatalities by **automating emergency alerts**.

3.3 System Architecture

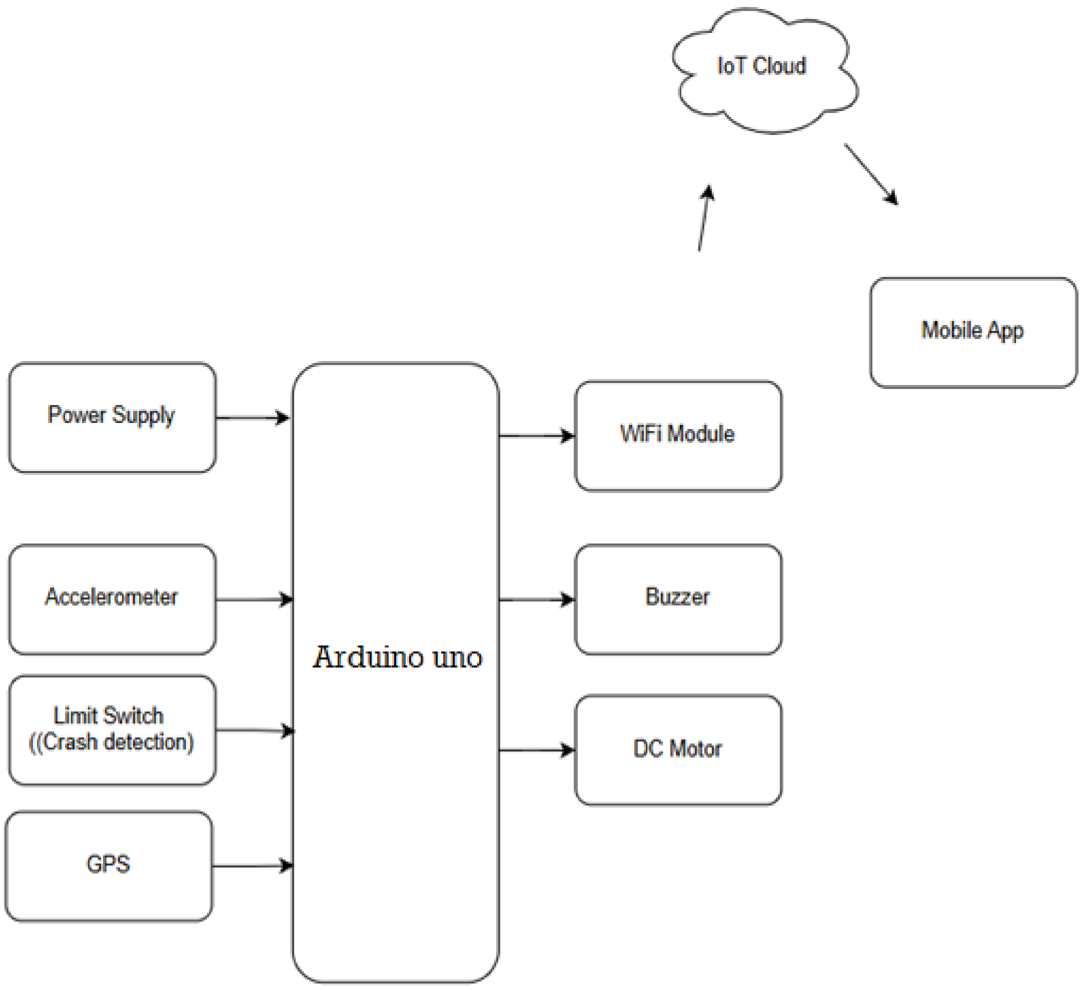
The system architecture is modular and includes:

- **Input Layer:** Collects real-time data from sensors
- **Processing Layer:** Uses Arduino or Raspberry Pi Pico to process sensor data and make decisions
- **Communication Layer:** Transmits information to the cloud using Wi-Fi and notifies users through a mobile app
- **Output Layer:** Includes LCD display, buzzer alert, and motor control



Block Diagram Flow:

1. **Sensors** → Detect accident or driver impairment
2. **Microcontroller** → Processes data and evaluates conditions
3. **GPS Module** → Fetches real-time location
4. **Wi-Fi Module** → Sends data to the cloud
5. **ThingSpeak Cloud** → Stores and forwards data
6. **Mobile Application** → Displays notifications to emergency contacts



This structure allows **scalability**, **real-time communication**, and **remote monitoring**, making it a reliable solution for both urban and rural use cases.

3.4 System Requirements

3.4.1 Hardware Requirements

Arduino Uno / Raspberry Pi Pico – Microcontroller platform**.**

5V Regulated Power Supply – Stable power source for components

Ultrasonic Sensor – Detects obstacles or vehicles in close proximity

Limit Switch – Serves as a crash or impact sensor

Alcohol Sensor (MQ-3) – Measures ethanol concentration in breath

IR Eye Blink Sensor – Detects signs of drowsiness or unconsciousness

GPS Module – Provides location data

Accelerometer (e.g., ADXL345) – Detects sudden impact or abnormal motion

DC Motor – Represents vehicle operation in test models

Buzzer – Alerts locally on detection

LCD Display (16x2) – Shows system status and messages

Wi-Fi Module (e.g., ESP8266/NodeMCU) – Sends data to the cloud

PCB and Connecting Wires – For circuit assembly

3.4.2 Software Requirements

Arduino IDE / Thonny IDE – For writing and uploading embedded code

Embedded C++ / MicroPython – Programming languages used for the controller

ThingSpeak (Open-source IoT Cloud Platform) – For data visualization and notification services

Mobile App (Android/iOS) – For receiving alerts and monitoring status

CHAPTER – 4

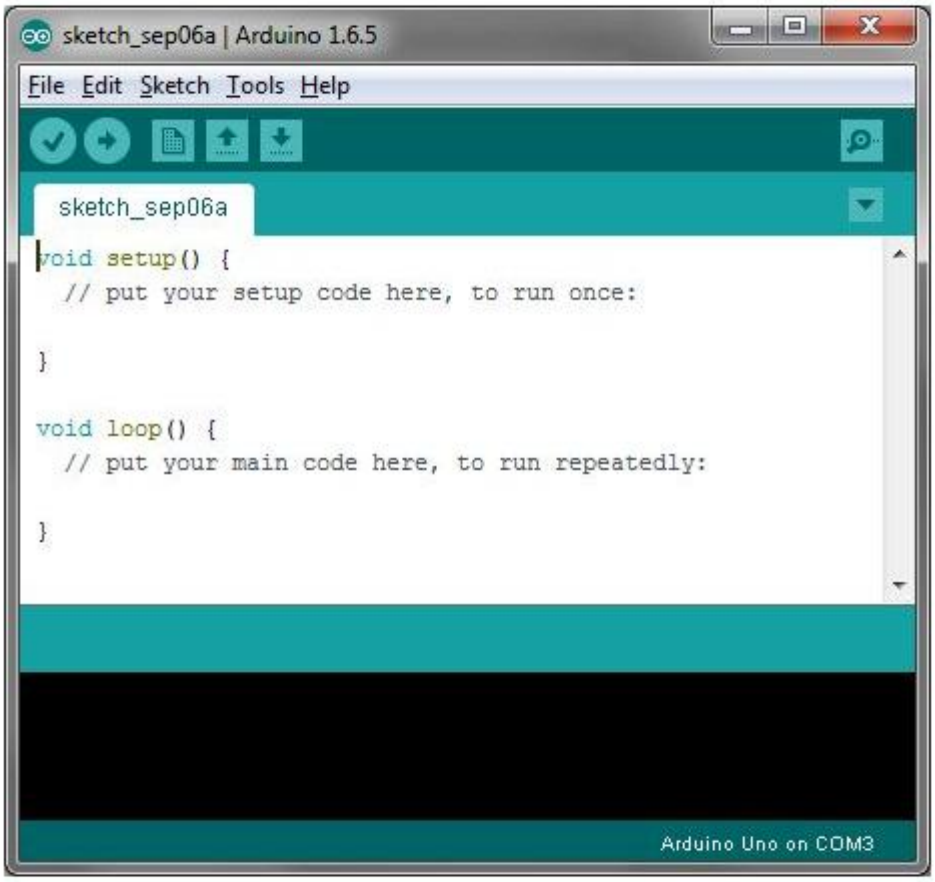
SOFTWARE ENVIRONMENT

ARDUINO IDE:

The Arduino Integrated Development Environment - or Arduino Software (IDE) - contains a text editor for writing code, a message area, a text console, a toolbar with buttons for common functions and a series of menus. It connects to the Arduino and Genuino hardware to upload programs and communicate with them.

Arduino IDE: Initial Setup

This is the Arduino IDE once it's been opened. It opens into a blank sketch where you can start programming immediately. First, we should configure the board and port settings to allow us to upload code. Connect your Arduino board to the PC via the USB cable.

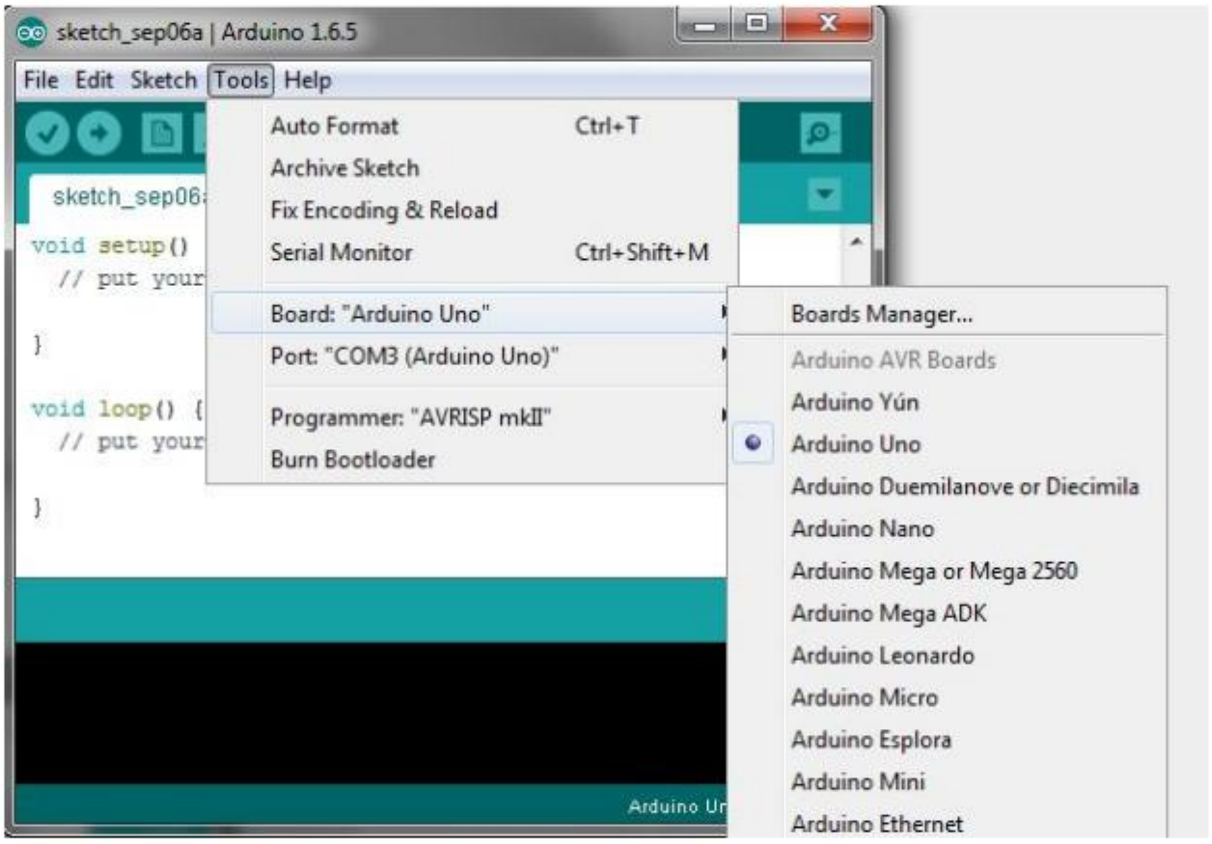


```
void setup() {
  // put your setup code here, to run once:

}

void loop() {
  // put your main code here, to run repeatedly:

}
```

Arduino IDE Default Window

IDE: Board Setup

You have to tell the Arduino IDE what board you are uploading to. Select the Tools pulldown menu and go to Board.This list is populated by default with the currently available Arduino Boards that are developed by Arduino. If you are using an Uno or an Uno-Compatible Clone (ex. Funduino, SainSmart, IEIK, etc.), select Arduino Uno. If you are using another board/clone, select that board.

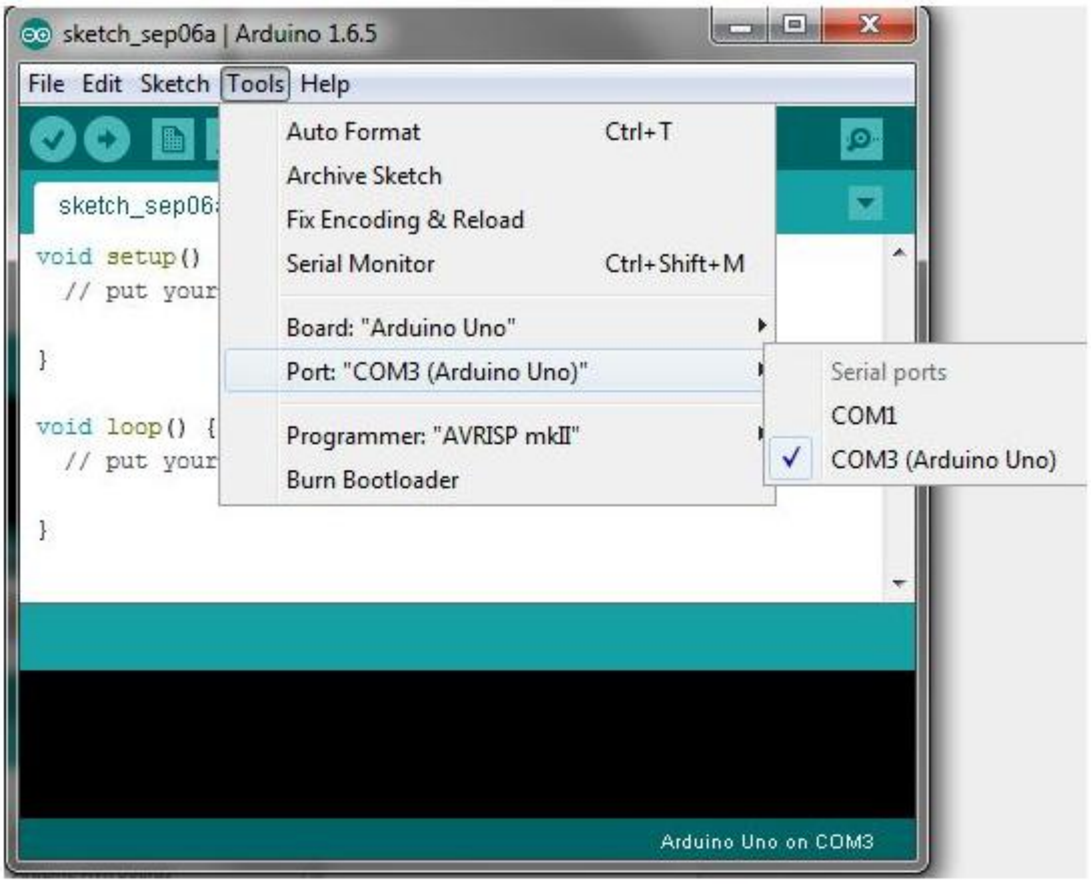


```
void setup()
  // put your
}

void loop() {
  // put your
}
```

Arduino IDE: Board Setup Procedure

IDE: COM Port Setup

If you downloaded the Arduino IDE before plugging in your Arduino board, when you plugged in the board, the USB drivers should have installed automatically. The most recent Arduino IDE should recognize connected boards and label them with which COM port they are using. Select the Tools pulldown menu and then Port.Here it should list all open COM ports, and if there is a recognized Arduino Board, it will also give it’s name. Select the Arduino board that you have connected to the PC. If the setup was successful, in the bottom right of the Arduino IDE, you should see the board type and COM number of the board you plan to program. Note: the Arduino Uno occupies the next available COM port; it will not always be COM3.


sketch_sep06a | Arduino 1.6.5
File Edit Sketch Tools Help
sketch_sep06a
Auto Format Ctrl+T
Archive Sketch
Fix Encoding & Reload
Serial Monitor Ctrl+Shift+M
Programmer: "AVRISP mkII"
Burn Bootloader
Serial ports
COM1

```
void setup()
  // put your

}

void loop() {
  // put your

}
```

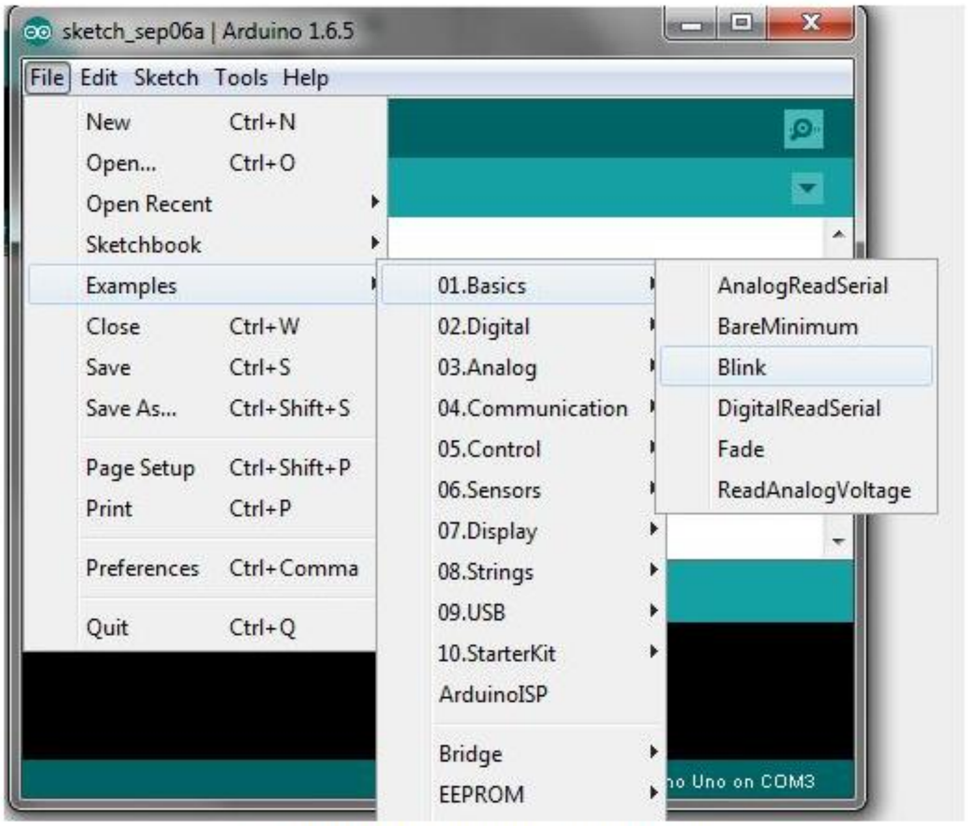
Board: "Arduino Uno"
Port: "COM3 (Arduino Uno)"
✓ COM3 (Arduino Uno)
Arduino Uno on COM3

Arduino IDE: COM Port Setup

Testing Your Settings:

Uploading Blink One common procedure to test whether the board you are using is properly set up is to upload the “Blink” sketch. This sketch is included with all Arduino IDE releases and can be accessed by the Filepull-down menu and going to Examples, 01.Basics, and then select Blink. Standard Arduino Boards include a surface-mounted LED labeled “L” or “LED” next to the “RX” and “TX” LEDs, that is connected to digital pin 13. This sketch will blink the LED at a regular interval, and is an easy way to confirm if your board is set up properly and you were successful in uploading code. Open the “Blink” sketch and press the “Upload” button in the upper-left corner to upload “Blink” to the board.

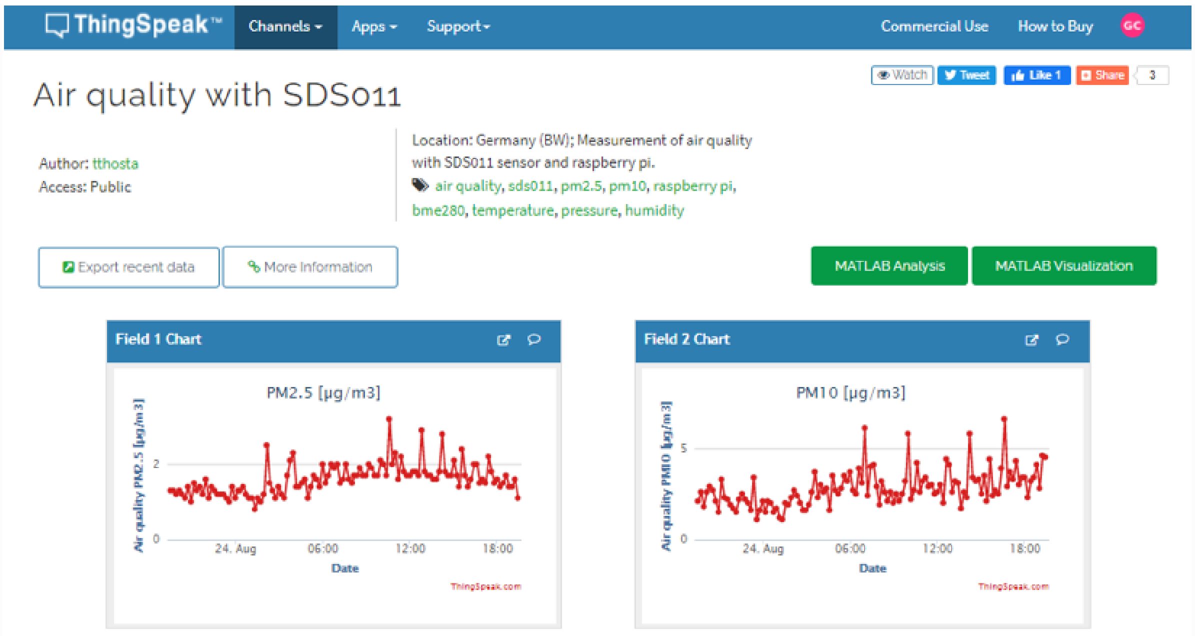
Upload Button:



Arduino IDE: Loading Blink Sketch

ThingSpeak IoT Platform

ThingSpeak is IoT platform for user to gather real-time data; for instance, climate information, location data and other device data. In different channels in ThingSpeak, you can summarize information and visualize data online in charts and analyze useful information.
ThingSpeak can integrate IoT:bit (micro:bit) and other software/ hardware platforms. Through IoT:bit, you can upload sensors data to ThingSpeak (e.g. temperature, humidity, light intensity, noise, motion, raindrop, distance and other device information).



Air quality with SDS011

Author: tthosta
Access: Public

Location: Germany (BW); Measurement of air quality with SDS011 sensor and raspberry pi.
air quality, sds011, pm2.5, pm10, raspberry pi, bme280, temperature, pressure, humidity

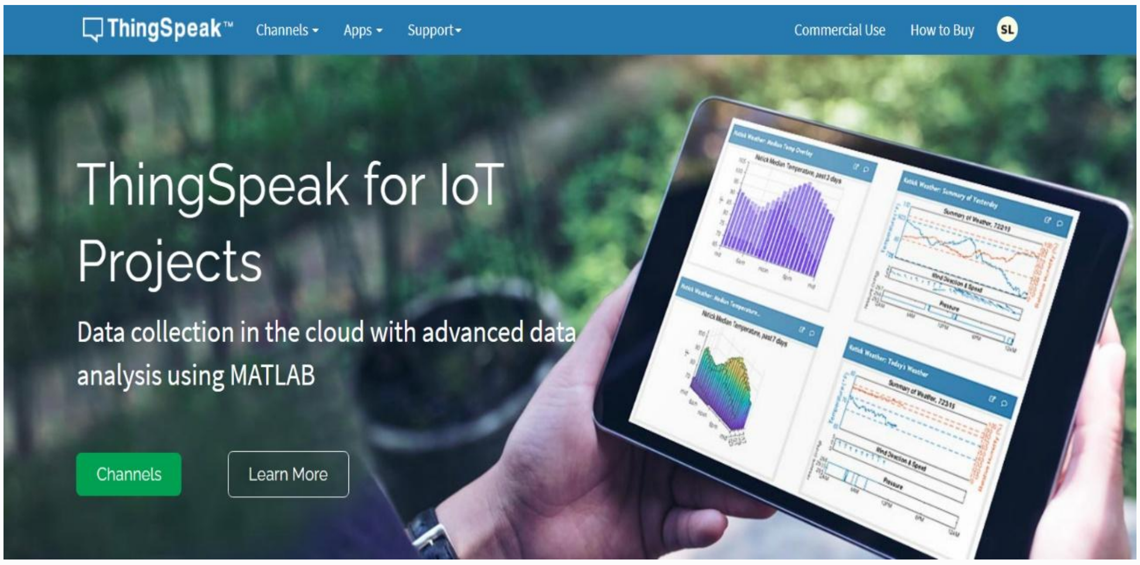
5.1. Thingspeak Configuration

Goal:

we need to create the thingspeak channel and get the key

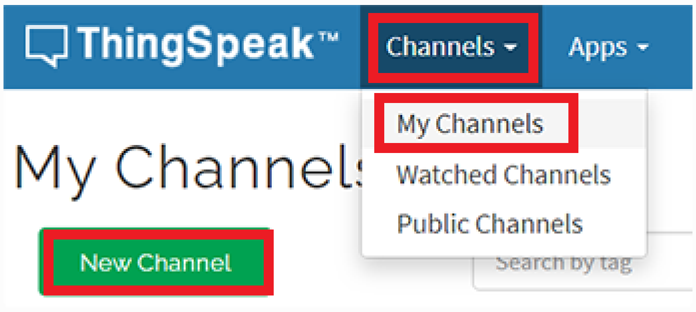
Step 1

Go to https://thingspeak.com/, register an account and login to the platform


ThingSpeak™
Channels
Apps
Support
Commercial Use
How to Buy
ThingSpeak for IoT Projects
Data collection in the cloud with advanced data analysis using MATLAB
Channels
Learn More

Step 2

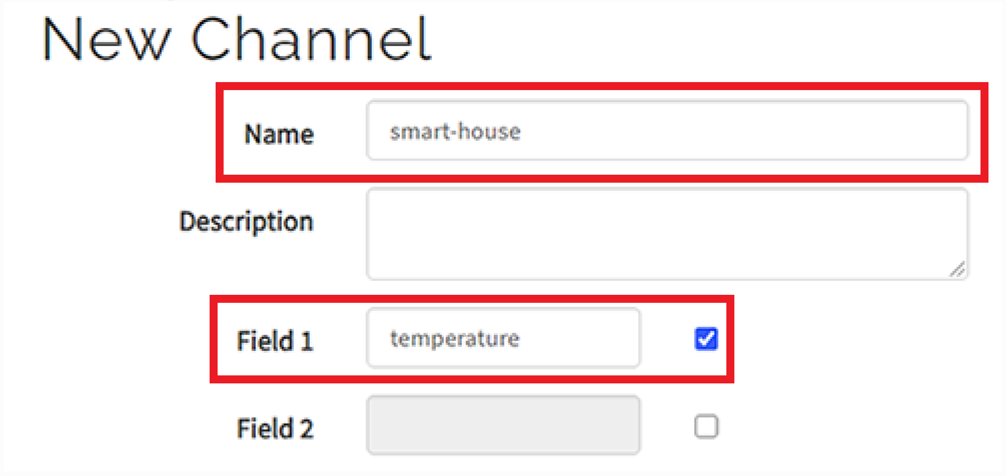
Choose Channels -> My Channels -> New Channel



Step 3

Input Channel name, Field1 , then click "Save Channel"

- Channel name: smart-house
- Field 1: Temperature


New Channel

Name: smart-house

Description:

Field 1: temperature ☑

Field 2: ☐

CHAPTER – 5

SYSTEM STUDY

This chapter focuses on the design methodology, integration of hardware and software modules, feasibility evaluations, and overall system behavior under different scenarios. The objective is to assess the system's practicality, reliability, and efficiency through structured analysis and simulated implementation.

5.1 Feasibility Study

Feasibility analysis is essential before actual implementation to ensure the system is viable from various perspectives—economic, technical, and social.

5.1.1 Economical Feasibility

The proposed accident detection system is designed with a strong emphasis on cost-efficiency. All major components such as the Arduino/Raspberry Pi, sensors, GPS, and Wi-Fi modules are low-cost and easily available in the market. Moreover:

- The software stack relies on open-source platforms like Arduino IDE, Thonny, and ThingSpeak, which eliminates licensing costs.
- Maintenance costs are minimal, as the system components are modular and can be individually replaced or updated.
- The development time is reduced due to the availability of well-documented libraries and community support.
- In comparison to commercial vehicle safety systems, this design offers a significantly more economical alternative with similar functionality.

Overall, the system presents a cost-effective solution for individuals, companies, and institutions aiming to improve vehicular safety.

5.1.2 Technical Feasibility

From a technical standpoint, the system is highly feasible:

- **Hardware Integration:** All selected sensors (e.g., accelerometer, alcohol sensor, IR eye blink sensor) are compatible with the Arduino/Raspberry Pi platforms. The modules communicate through standard protocols like I2C, UART, and GPIO.
- **Software Implementation:** Programming environments such as Arduino IDE and MicroPython simplify sensor data acquisition, processing, and control logic development.
- **Data Communication:** The use of ThingSpeak for data visualization and mobile app integration ensures seamless cloud connectivity using the Wi-Fi module.
- **Expandability:** The system design allows for easy integration of additional features like temperature monitoring, real-time video capture, or GSM-based SMS alerts.
- **Testing Environment:** The prototype has been tested under controlled conditions simulating different accident scenarios, confirming that the system responds accurately and in real-time.

Thus, the system is technically sound, scalable, and suitable for deployment in real-world conditions.

5.1.3 Social Feasibility

The societal impact and acceptance of this system are highly positive:

- **Life-saving Potential:** Rapid response enabled by real-time alerts can significantly reduce fatalities and injuries caused by road accidents.
- **Awareness & Adoption:** With growing awareness of road safety and the emergence of smart cities, such systems are welcomed by the public and authorities.
- **Driver Behavior Improvement:** Features like alcohol detection and drowsiness monitoring encourage responsible driving and can prevent accidents before they happen.
- **Accessibility:** The affordability and simplicity of this system make it accessible even in developing regions or rural areas with limited infrastructure.

In conclusion, the project is not only socially acceptable but also aligns with global goals of smart transportation and public safety.

CHAPTER – 6

SYSTEM DESIGN

System design is the blueprint for constructing a fully functional system. It outlines how each component interacts with others, the logical flow of information, and the architectural foundation for software and hardware integration.

6.1 UML Diagrams

Unified Modeling Language (UML) diagrams help in visualizing the system structure and behavior during the design phase.

6.1.1 Use Case Diagram

A Use Case Diagram shows the interactions between users and the system. In this project, the primary actor is the **Driver**, while other supporting actors include **Emergency Services** and **Mobile Application**.

Use Cases:

- Start vehicle system
- Detect accident
- Monitor driver alertness
- Detect alcohol level
- Send location data
- Notify emergency contacts
- Update cloud dashboard

This helps identify all functionalities the system must provide from a user interaction perspective.

6.1.2 Sequence Diagram

The Sequence Diagram details the flow of operations between components over time.

Example: Accident Scenario Sequence:

1. Sensors detect impact or irregular movement.
2. Microcontroller processes the sensor input.
3. GPS fetches current location.
4. Wi-Fi module sends the data to ThingSpeak.
5. Cloud platform triggers alert to mobile app.
6. App displays alert to emergency contact.

This ensures all steps happen in a synchronized and timely manner.

6.1.3 Activity Diagram

An Activity Diagram illustrates the system's workflow, decision-making processes, and sequential execution.

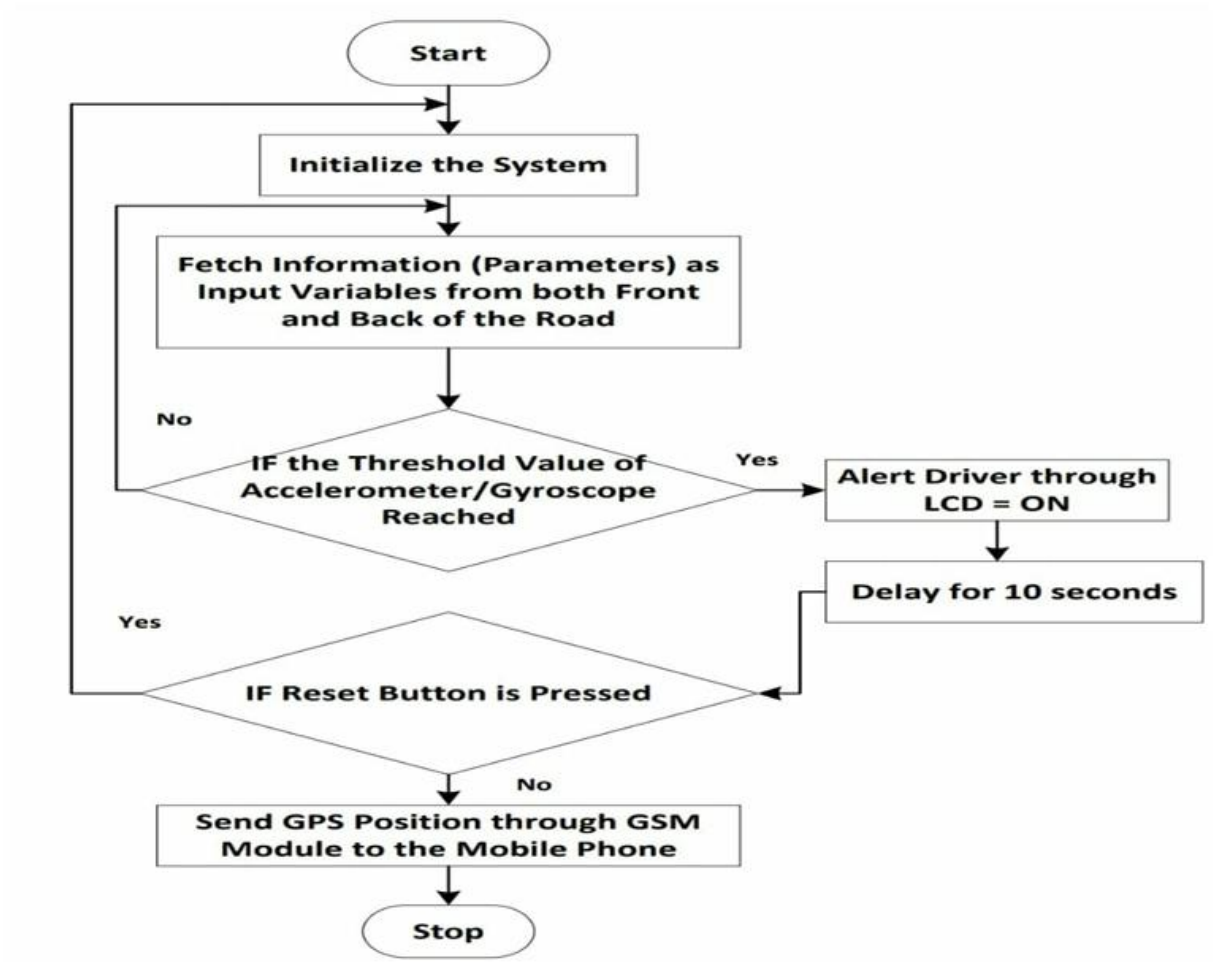
Typical Workflow:

- Start system → Monitor sensors → Accident detected?
    - If yes → Get GPS location → Send data to cloud → Alert app
    - If no → Continue monitoring

This aids in visualizing the system’s dynamic behavior.

6.2 Data Flow Diagram (DFD)

A DFD shows how data moves through the system at different abstraction levels.


Start
Initialize the System
Fetch Information (Parameters) as Input Variables from both Front and Back of the Road
No
IF the Threshold Value of Accelerometer/Gyroscope Reached
Yes
Alert Driver through LCD = ON
Delay for 10 seconds
Yes
IF Reset Button is Pressed
No
Send GPS Position through GSM Module to the Mobile Phone
Stop

Level 0 DFD:

- **Input:** Sensor readings
- **Process:** Data processing by microcontroller
- **Output:** Notifications via Wi-Fi to cloud and mobile

Level 1 DFD:

- Eye blink sensor → Drowsiness detection module → Alert system
- Alcohol sensor → Driver analysis → Motor lock or alert
- Accelerometer → Crash detection → GPS & Notification system

This helps in identifying bottlenecks and refining the flow of information.

6.3 System Architecture

The system architecture integrates all components—both hardware and software:

- **Hardware Layer:** Sensors, GPS, Wi-Fi, Arduino/Raspberry Pi
- **Processing Layer:** Embedded logic for interpreting sensor data
- **Communication Layer:** ThingSpeak cloud platform and mobile app interface
- **Presentation Layer:** Visual indicators (LCD), Buzzer alerts, and App UI

It defines the hierarchy and interaction between layers and ensures reliable real-time operation.

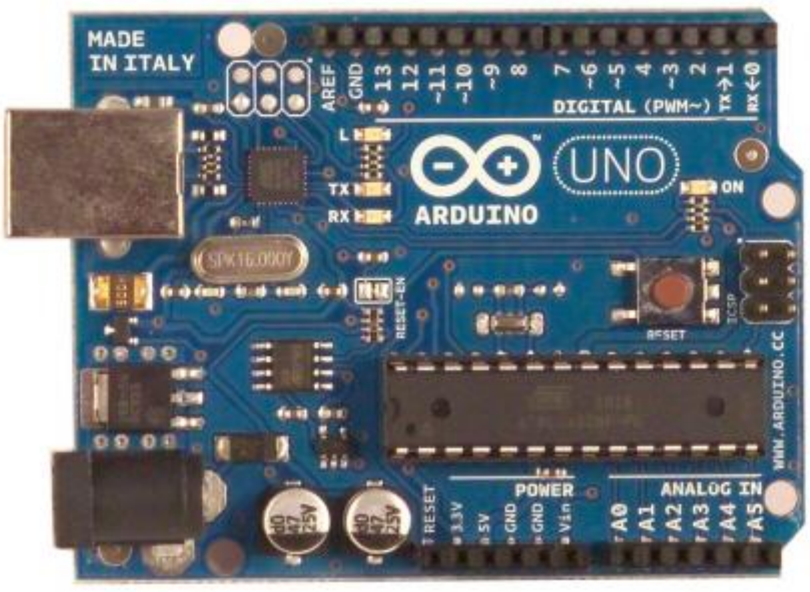
Arduino

Arduino is open source physical processing which is base on a microcontroller board and an incorporated development environment for the board to be programmed. Arduino gains a few inputs, for example, switches or sensors and control a few multiple outputs, for example, lights, engine and others. Arduino program can run on Windows, Macintosh and Linux operating systems (OS) opposite to most microcontrollers' frameworks which run only on Windows. Arduino programming is easy to learn and apply to beginners and amateurs. Arduino is an instrument used to build a better version of a computer which can control, interact and sense more than a normal desktop computer. It's an open-source physical processing stage focused around a straightforward microcontroller board, and an environment for composing programs for the board. Arduino can be utilized to create interactive items, taking inputs from a diverse collection of switches or sensors, and controlling an assortment of lights, engines, and other physical outputs. Arduino activities can be remaining solitary, or they can be associated with programs running on your machine (e.g. Flash, Processing and Maxmsp.) The board can be amassed by hand or bought preassembled; the open-source IDE can be downloaded free of charge. Focused around the Processing media programming environment, the Arduino programming language is an execution of Wiring, a comparative physical computing platform. Figure 7- Arduino's

ARDUINO UNO:

The Arduino Uno is a microcontroller board based on the ATmega328 (datasheet). It 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. It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with a AC-to-DC adapter or battery to get started. The Uno differs from all preceding boards in that it does not use the FTDI USB-to-serial driver chip. Instead, it features the Atmega8U2 programmed as a USB-to-serial converter. "Uno" means one in Italian and is named to mark the upcoming release of Arduino 1.0. The Uno and version 1.0 will be the reference versions of Arduno, moving forward. The Uno is the latest in a series of USB Arduino boards, and the reference model for the Arduino platform;



Technical specifications of arduino:

Microcontroller: ATmega328

Operating Voltage: 5V

Input Voltage (recommended): 7-12V

Input Voltage (limits): 6-20V

Digital I/O Pins 14 (of which 6 provide PWM output)

Analog Input Pins 6

DC Current per I/O Pin 40 mA

DC Current for 3.3V Pin 50 mA

Flash Memory

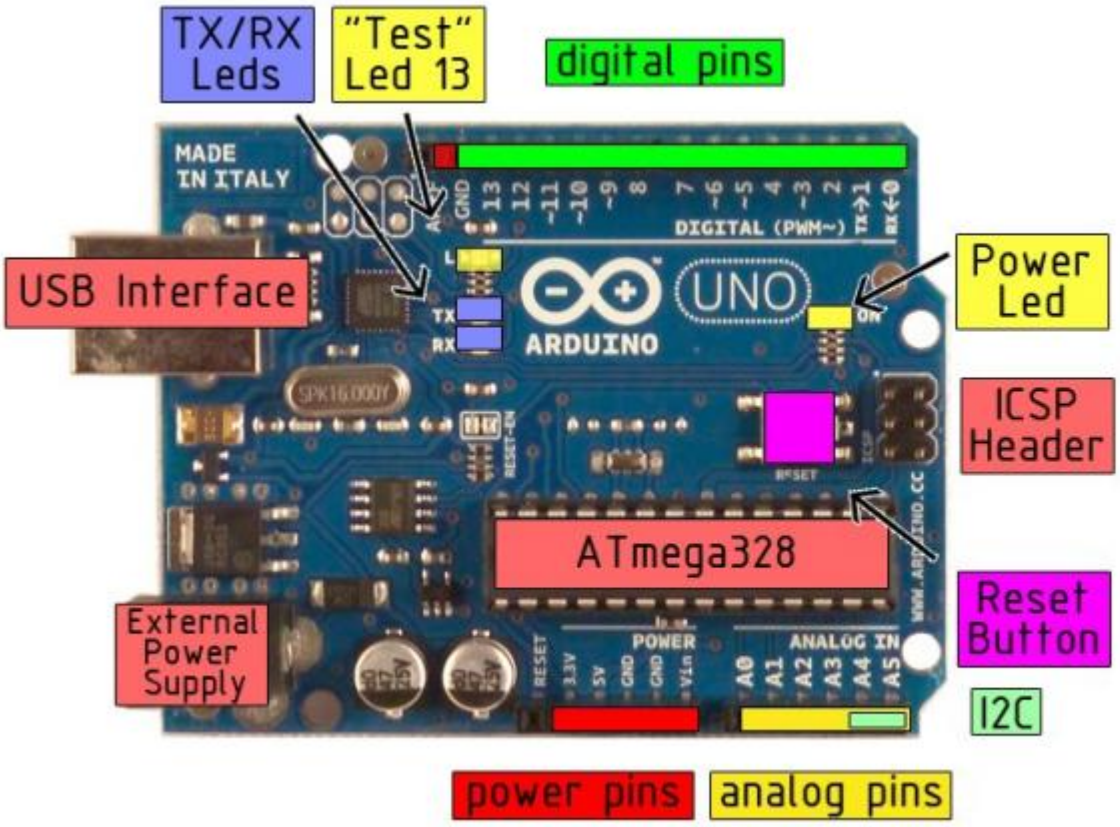
32 KB of which 0.5 KB used by

bootloader

SRAM 2 KB

EEPROM 1 KB

Clock Speed 16 MHz



POWER

The Arduino Uno can be powered via the USB connection or with an external power supply. The power source is selected automatically.

External (non-USB) power can come either from an AC-to-DC adapter (wall-wart) or battery. The adapter can be connected by plugging a 2.1mm center-positive plug into the board's power jack. Leads from a battery can be inserted in the Gnd and Vin pin headers of the POWER connector.

The board can operate on an external supply of 6 to 20 volts. If supplied with less than 7V, however, the 5V pin may supply less than five volts and the board may be unstable. If using more than 12V, the voltage regulator may overheat and damage the board. The recommended range is 7 to 12 volts.

The power pins are as follows:

• **VIN**. The input voltage to the Arduino board when it's using an external power source (as opposed to 5 volts from the USB connection or other regulated power source). You can supply voltage through this pin, or, if supplying voltage via the power jack, access it through this pin.

• **5V**. The regulated power supply used to power the microcontroller and other components on the board. This can come either from VIN via an on-board regulator, or be supplied by USB or another regulated 5V supply.

• **3V3**. A 3.3 volt supply generated by the on-board regulator. Maximum current draw is 50 mA.

• **GND**. Ground pins.

MEMORY:

The Atmega328 has 32 KB of flash memory for storing code (of which 0,5 KB is used for the bootloader); It has also 2 KB of SRAM and 1 KB of EEPROM (which can be read and written with the EEPROM library).

INPUT/OUTPUT

Each of the 14 digital pins on the Uno can be used as an input or output, using pinMode(), digitalWrite(), and digitalRead() functions. They operate at 5 volts. Each pin can provide or receive a maximum of 40 mA and has an internal pull-up resistor (disconnected by default) of 20-50 kOhms. In addition, some pins have specialized functions:

• Serial: 0 (RX) and 1 (TX). Used to receive (RX) and transmit (TX) TTL serial data. TThese pins are connected to the corresponding pins of the ATmega8U2 USB-to-TTL Serial chip .

• External Interrupts: 2 and 3. These pins can be configured to trigger an interrupt on a low value, a rising or falling edge, or a change in value. See the attachInterrupt() function for details.

• PWM: 3, 5, 6, 9, 10, and 11. Provide 8-bit PWM output with the analogWrite() function.

• SPI: 10 (SS), 11 (MOSI), 12 (MISO), 13 (SCK). These pins support SPI communication, which, although provided by the underlying hardware, is not currently included in the Arduino language.

• LED: 13. There is a built-in LED connected to digital pin 13. When the pin is HIGH value, the LED is on, when the pin is LOW, it's off. The Uno has 6 analog inputs, each of which provide 10 bits of resolution (i.e. 1024 different values). By default they measure from ground to 5 volts, though is it possible to change the upper end of their range using the AREF pin and the analogReference() function. Additionally, some pins have specialized functionality:

• I 2C: 4 (SDA) and 5 (SCL). Support I2C (TWI) communication using the Wire library. There are a couple of other pins on the board:

• AREF. Reference voltage for the analog inputs. Used with analogReference().

• Reset. Bring this line LOW to reset the microcontroller. Typically used to add a reset button to shields which block the one on the board.

COMMUNICATION:

The Arduino Uno has a number of facilities for communicating with a computer, another Arduino, or other microcontrollers. The ATmega328 provides UART TTL (5V) serial communication, which is available on digital pins 0 (RX) and 1 (TX). An ATmega8U2 on the board channels this serial communication over USB and appears as a virtual com port to software on the computer. The '8U2 firmware uses the standard USB COM drivers, and no external driver is needed. However, on Windows, an *.inf file is required..

The Arduino software includes a serial monitor which allows simple textual data to be sent to and from the Arduino board. The RX and TX LEDs on the board will flash when data is being transmitted via the USB-toserial chip and USB connection to the computer (but not for serial communication on pins 0 and 1).

A SoftwareSerial library allows for serial communication on any of the Uno's digital pins.

The ATmega328 also support I2C (TWI) and SPI communication. The Arduino software includes a Wire library to simplify use of the I2C bus; see the documentation for details. To use the SPI communication, please see the ATmega328 datasheet.

Programming

The Arduino Uno can be programmed with the Arduino software. The ATmega328 on the Arduino Uno comes preburned with a bootloader that allows you to upload new code to it without the use of an external hardware programmer. It communicates using the original STK500 protocol (reference, C headerfiles). You can also bypass the bootloader and program the microcontroller through the ICSP (InCircuit Serial Programming) header; see these instructions for details. The ATmega16U2 (or 8U2 in the rev1 and rev2 boards) firmware source code is available . The ATmega16U2/8U2 is loaded with a DFU bootloader, which can be activated by: On Rev1 boards: connecting the solder jumper on the back of the board (near the map of Italy) and then resetting the 8U2. 10 On Rev2 or later boards: there is a resistor that pulling the 8U2/16U2 HWB line to ground, making it easier to put into DFU mode. You can then use Atmel's FLIP software (Windows) or the DFU programmer (Mac OS X and Linux) to load a new firmware. Or you can use the ISP header with an external programmer (overwriting the DFU bootloader). See this user-contributed tutorial for more information.

Automatic (Software) Reset

Rather than requiring a physical press of the reset button before an upload, the Arduino Uno is designed in a way that allows it to be reset by software running on a connected computer. One of the hardware flow control lines (DTR) of theATmega8U2/16U2 is connected to the reset line of the ATmega328 via a 100 nanofarad capacitor. When this line is asserted (taken low), the reset line drops long enough to reset the chip. The Arduino software uses this capability to allow you to upload code by simply pressing the upload button in the Arduino environment. This means that the bootloader can have a shorter timeout, as the lowering of DTR can be well-coordinated with the start of the upload. This setup has other implications. When the Uno is connected to either a computer running Mac OS X or Linux, it resets each time a connection is made to it from software (via USB). For the following half-second or so, the bootloader is running on the Uno. While it is programmed to ignore malformed data (i.e. anything besides an upload of new code), it will intercept the first few bytes of data sent to the board after a connection is opened. If a sketch running on the board receives one-time configuration or other data when it first starts, make sure that the software with which it communicates waits a second after opening the connection and before sending this data. The Uno contains a trace that can be cut to disable the auto-reset. The pads on either side of the trace can be soldered together to re-enable it. It's labeled "RESET-EN". You may also be able to disable the auto-reset by connecting a 110 ohm resistor from 5V to the reset line. 11

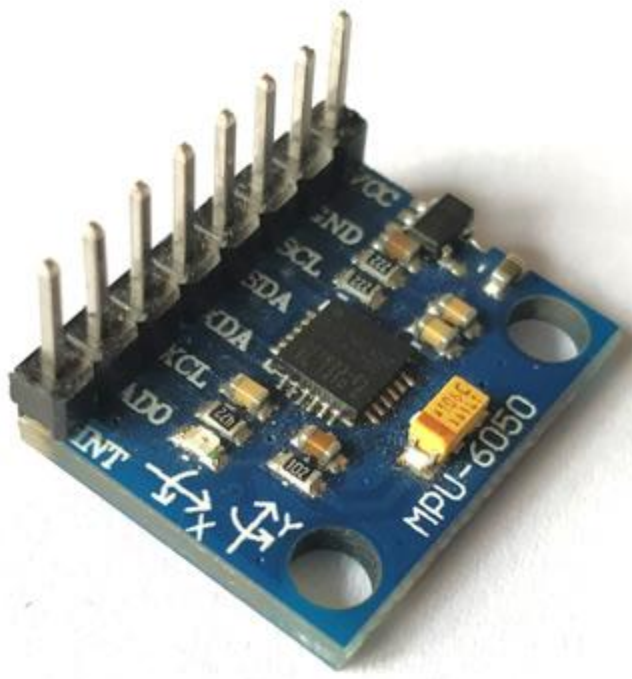
USB Overcurrent Protection

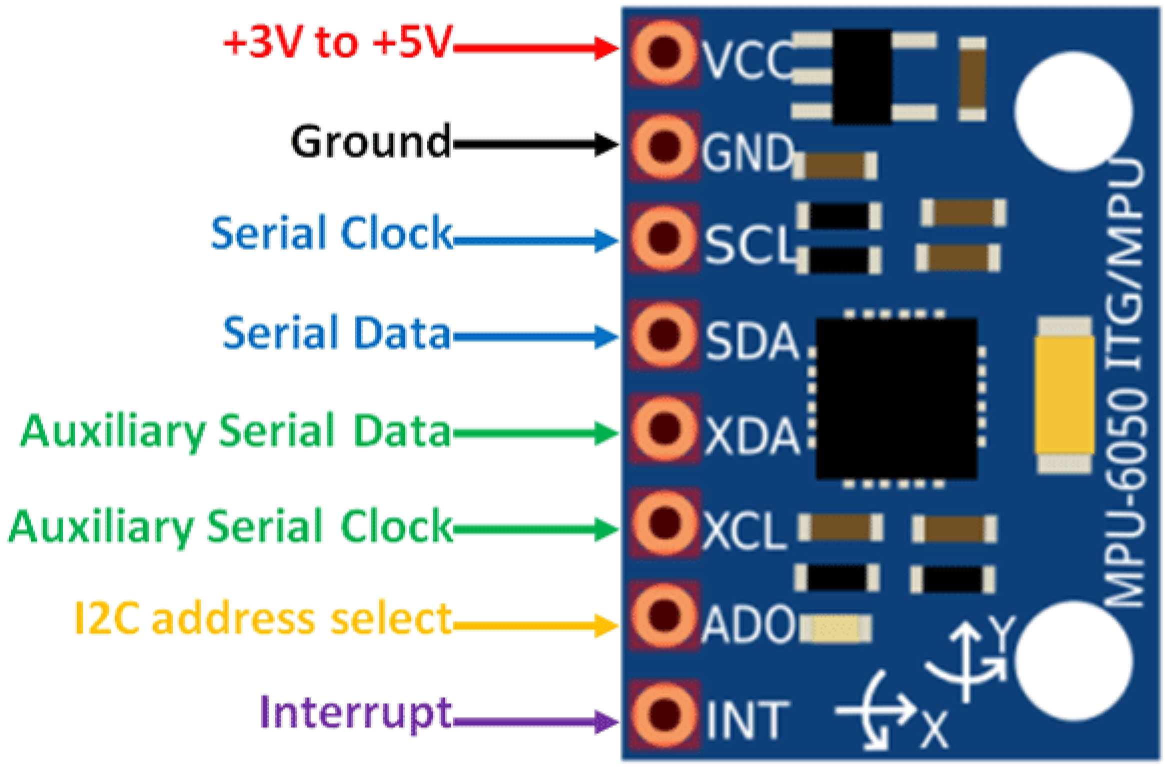
The Arduino Uno has a resettable polyfuse that protects your computer's USB ports from shorts and overcurrent. Although most computers provide their own internal protection, the fuse provides an extra layer of protection. If more than 500 mA is applied to the USB port, the fuse will automatically break the connection until the short or overload is removed.

Physical Characteristics

The maximum length and width of the Uno PCB are 2.7 and 2.1 inches respectively, with the USB connector and power jack extending beyond the former dimension. Four screw holes allow the board to be attached to a surface or case. Note that the distance between digital pins 7 and 8 is 160 mil (0.16"), not an even multiple of the 100 mil spacing of the other pins.

MPU6050 Accelerometer and Gyroscope Module





MPU6050 Pinout

The **MPU6050 module** is a Micro Electro-Mechanical Systems (**MEMS**) which consists of a 3-axis Accelerometer and 3-axis Gyroscope inside it. This helps us to measure acceleration, velocity, orientation, displacement and many other motion related parameter of a system or object.

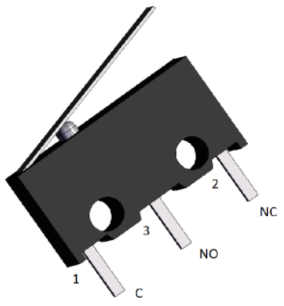
MPU6050 Pinout Configuration

Pin Number	**Pin Name**	**Description**
1	Vcc	Provides power for the module, can be +3V to +5V. Typically +5V is used
2	Ground	Connected to Ground of system
3	Serial Clock (SCL)	Used for providing clock pulse for I2C Communication
4	Serial Data (SDA)	Used for transferring Data through I2C communication
5	Auxiliary Serial Data (XDA)	Can be used to interface other I2C modules with MPU6050. It is optional
6	Auxiliary Serial Clock (XCL)	Can be used to interface other I2C modules with MPU6050. It is optional
7	AD0	If more than one MPU6050 is used a single MCU, then this pin can be used to vary the address
8	Interrupt (INT)	Interrupt pin to indicate that data is available for MCU to read.

MPU6050 Features

- MEMS 3-aixs accelerometer and 3-axis gyroscope values combined
- Power Supply: 3-5V
- Communication : I2C protocol
- Built-in 16-bit ADC provides high accuracy
- Built-in DMP provides high computational power
- Can be used to interface with other IIC devices like magnetometer
- Configurable IIC Address
- In-built Temperature sensor

Micro Switch or Snap-action Switch



Pin Configuration

Sr. No.	Pin Name	Description
1.	C (Common terminal)	Input
2.	NC (Normally Close)	Output 1
3.	NO (Normally Open)	Output 2

Description

- Micro switch ZM and ZM1 Series are **subminiature snap action switches** from the Honeywell micro switch family of Z Series subminiature basic switches.
- Although small in size, the ZM and ZM1 Series are rated for controlling electrical loads ranging from logic level (computer based circuits) to power duty switching (up to 16.1 A and 250 Vac).
- The package size of the subminiature switch is ideal for applications where space on the equipment is at a premium.
- The overall length of the ZM and ZM1 Series are less than 20 mm [0.78 in
- A wide variety of integral stainless steel levers are available and when combined with the subminiature package size, may adapt the switch to a wide variety of applications.
- The ZM Series is agency certified to UL, cUL, CE, and CQC for worldwide use, while the ZM1 Series is agency certified to UL, cUL, ENEC, and CQC for worldwide use

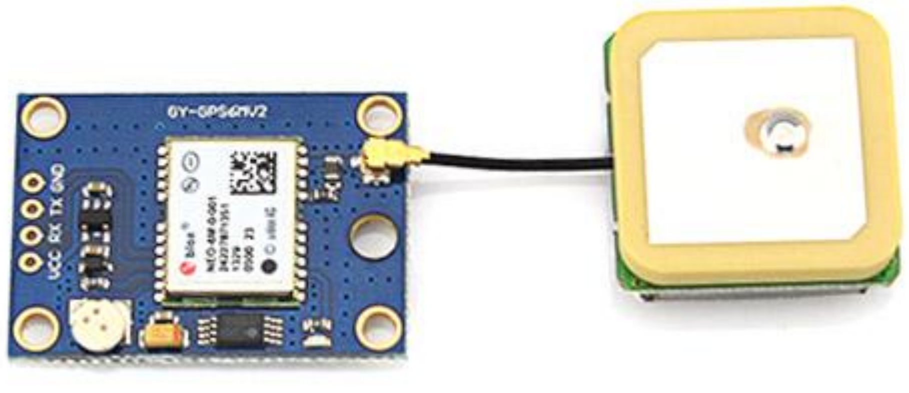
Features

The ZM and ZM1 series comes with the following features in general-

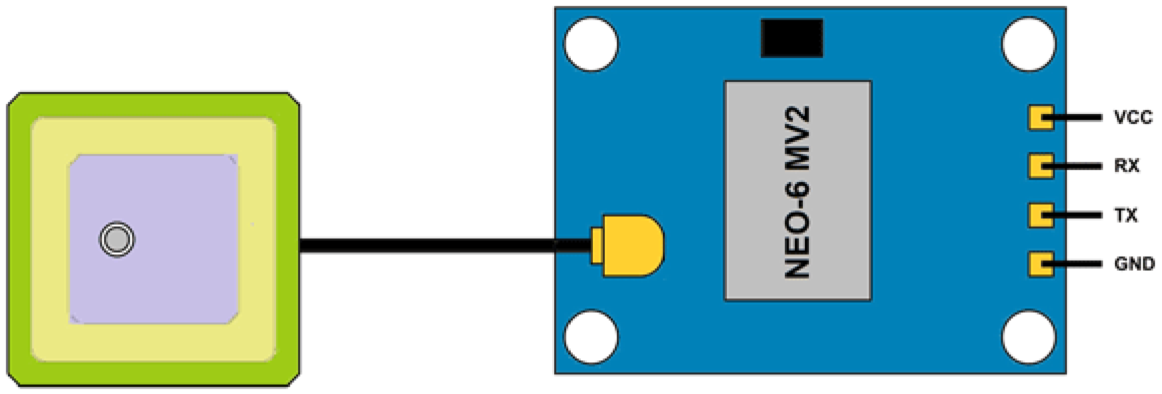
- Well suited for power-duty and logic-level loads
- SPDT, SPNC, or SPNO switch options
- Power duty switching with silver alloy contacts
- Gold-plated, silver alloy contacts for logic-level control

- Pin plunger and various stainless steel lever options, including internal and external mount levers
- Variety of electrical terminations
- Certified per UL, cUL to UL 61058-1, ENEC to IEC 61058-1, and CQC to GB 15092.1

NEO-6MV2 GPS Module



NEO-6MV2 GPS Module



NEO-6MV2 GPS Module Pinout

The **NEO-6MV2** is a **GPS** (Global Positioning System) module and is used for navigation. The module simply checks its location on earth and provides output data which is longitude and latitude of its position.It is from a family of stand-alone GPS receivers featuring the high performance u-blox 6 positioning engine. These flexible and cost effective receivers offer numerous connectivity options in a miniature (16 x 12.2 x 2.4 mm) package. The compact architecture, power and memory options make **NEO-6 modules** ideal for **battery operated mobile devices** with very strict cost and space constraints. Its Innovative design gives **NEO-6MV2** excellent navigation performance even in the most challenging environments.

NEO-6MV2 GPS Module Pin Configuration

The module has four output pins and we will describe the function each pin of them below. The powering of module and communication interface is done through these four pins.

Pin Name	**Description**
VCC	Positive power pin
RX	UART receive pin
TX	UART transmit pin
GND	Ground

Features and Electrical Characteristics

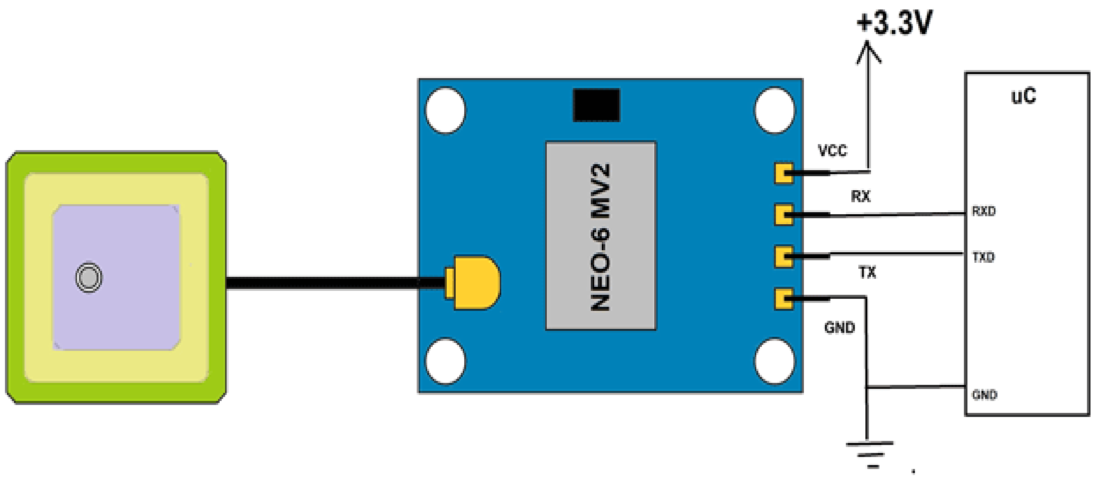
- Standalone GPS receiver
- Anti-jamming technology
- UART Interface at the output pins (Can use SPI ,I2C and USB by soldering pins to the chip core)
- Under 1 second time-to-first-fix for hot and aided starts
- Receiver type: 50 Channels - GPS L1 frequency - SBAS (WAAS, EGNOS, MSAS, GAGAN)
- Time-To-First-fix: For Cold Start 32s, For Warm Start 23s, For Hot Start <1s
- Maximum navigation update rate: 5Hz
- Default baud rate: 9600bps
- EEPROM with battery backup
- Sensitivity: -160dBm
- Supply voltage: 3.6V
- Maximum DC current at any output: 10mA
- Operation limits: Gravity-4g, Altitude-50000m, Velocity-500m/s
- Operating temperature range: -40ºC TO 85°C

Overview of the NEO-6MV2 GPS Module

This module is one of popular GPS modules on the market and is also cheap to buy. The location data provided by it is accurate enough to satisfy most applications. And for it to be included in smart phones and tablets design points out its efficiency. This module is famous among hobbyist and engineers altogether who want to work on applications involving navigation.

How to use the NEO-6MV2 GPS Module

Getting this module to work is very easy. For the application circuit below we have connected the power to board and interfaced the output to the microcontroller UART to get it done.



After circuitry you need to set the baud rate of the controller matching the module, if it’s not matched you will get error. With baud rate setting done you can read the serial data directly from the module. This data will be longitude and latitude values and the user can play with them as desired.

The raw values provided by the module are cumbersome to read directly and so a simple decimal calculation can be done in programming for getting easy to read values.

Applications

- GPS application
- Smart phone and tablets
- Navigation systems
- Drones
- Hobby projects

LCD:

LCD 16×2 Pin Configuration and Its Working

Nowadays, we always use the devices which are made up of LCDs such as CD players, DVD players, digital watches, computers, etc. These are commonly used in the screen industries to replace the utilization of CRTs. Cathode Ray Tubes use huge power when compared with LCDs, and CRTs heavier as well as bigger. These devices are thinner as well power consumption is extremely less. The LCD 16×2 working principle is, it blocks the light rather than dissipate. This article discusses an overview of LCD 16X2, pin configuration and its working.

What is the LCD 16×2?

The term LCD stands for liquid crystal display. It is one kind of electronic display module used in an extensive range of applications like various circuits & devices like mobile phones, calculators, computers, TV sets, etc. These displays are mainly preferred for multi-segment light-emitting diodes and seven segments. The main benefits of using this module are inexpensive; simply programmable, animations, and there are no limitations for displaying custom characters, special and even animations, etc.

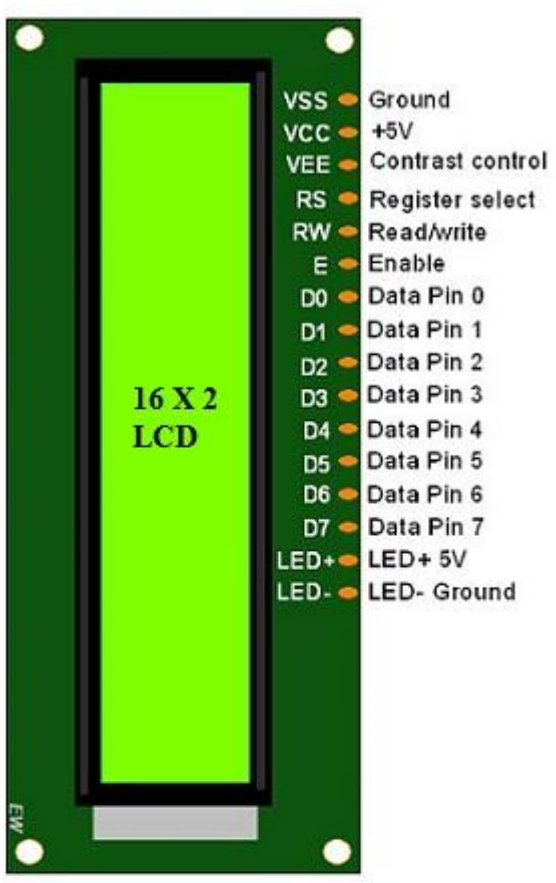
16X2 LCD

LCD 16×2 Pin Diagram

The 16×2 LCD pinout is shown below.

- Pin1 (Ground/Source Pin): This is a GND pin of display, used to connect the GND terminal of the microcontroller unit or power source.
- Pin2 (VCC/Source Pin): This is the voltage supply pin of the display, used to connect the supply pin of the power source.
- Pin3 (V0/VEE/Control Pin): This pin regulates the difference of the display, used to connect a changeable POT that can supply 0 to 5V.

- Pin4 (Register Select/Control Pin): This pin toggles among command or data register, used to connect a microcontroller unit pin and obtains either 0 or 1(0 = data mode, and 1 = command mode).
- Pin5 (Read/Write/Control Pin): This pin toggles the display among the read or writes operation, and it is connected to a microcontroller unit pin to get either 0 or 1 (0 = Write Operation, and 1 = Read Operation).
- Pin 6 (Enable/Control Pin): This pin should be held high to execute Read/Write process, and it is connected to the microcontroller unit & constantly held high.
- Pins 7-14 (Data Pins): These pins are used to send data to the display. These pins are connected in two-wire modes like 4-wire mode and 8-wire mode. In 4-wire mode, only four pins are connected to the microcontroller unit like 0 to 3, whereas in 8-wire mode, 8-pins are connected to microcontroller unit like 0 to 7.
- Pin15 (+ve pin of the LED): This pin is connected to +5V
- Pin 16 (-ve pin of the LED): This pin is connected to GND.



LCD-16×2-pin-diagram

Features of LCD16x2

The features of this LCD mainly include the following.

- The operating voltage of this LCD is 4.7V-5.3V
- It includes two rows where each row can produce 16-characters.

- The utilization of current is 1mA with no backlight
- Every character can be built with a 5×8 pixel box
- The alphanumeric LCDs alphabets & numbers
- Is display can work on two modes like 4-bit & 8-bit
- These are obtainable in Blue & Green Backlight
- It displays a few custom generated characters

Registers of LCD

A 16×2 LCD has two registers like data register and command register. The RS (register select) is mainly used to change from one register to another. When the register set is '0', then it is known as command register. Similarly, when the register set is '1', then it is known as data register.

Command Register

The main function of the command register is to store the instructions of command which are given to the display. So that predefined tasks can be performed such as clearing the display, initializing, set the cursor place, and display control. Here commands processing can occur within the register.

Data Register

The main function of the data register is to store the information which is to be exhibited on the LCD screen. Here, the ASCII value of the character is the information which is to be exhibited on the screen of LCD. Whenever we send the information to LCD, it transmits to the data register, and then the process will be starting there. When register set =1, then the data register will be selected.

16×2 LCD Commands

The commands of LCD 16X2 include the following.

- For Hex Code-01, the LCD command will be the clear LCD screen
- For Hex Code-02, the LCD command will be returning home
- For Hex Code-04, the LCD command will be decrement cursor
- For Hex Code-06, the LCD command will be Increment cursor
- For Hex Code-05, the LCD command will be Shift display right
- For Hex Code-07, the LCD command will be Shift display left
- For Hex Code-08, the LCD command will be Display off, cursor off
- For Hex Code-0A, the LCD command will be cursor on and display off
- For Hex Code-0C, the LCD command will be cursor off, display on
- For Hex Code-0E, the LCD command will be cursor blinking, Display on
- For Hex Code-0F, the LCD command will be cursor blinking, Display on
- For Hex Code-10, the LCD command will be Shift cursor position to left
- For Hex Code-14, the LCD command will be Shift cursor position to the right
- For Hex Code-18, the LCD command will be Shift the entire display to the left
- For Hex Code-1C, the LCD command will be Shift the entire display to the right
- For Hex Code-80, the LCD command will be Force cursor to the beginning (1st line)

- For Hex Code-C0, the LCD command will be Force cursor to the beginning (2nd line)
- For Hex Code-38, the LCD command will be 2 lines and 5×7 matrix

ESP8266 D1 Mini V2 NodeMCU

The **ESP8266 D1 Mini V2 NodeMCU Lua IoT Dev Board** is a compact and versatile development board designed for IoT applications. Featuring the powerful ESP8266 Wi-Fi module, it supports Lua scripting and Arduino IDE for easy programming. Its small form factor and extensive GPIO options make it perfect for smart devices, automation systems, and other Wi-Fi-enabled projects.

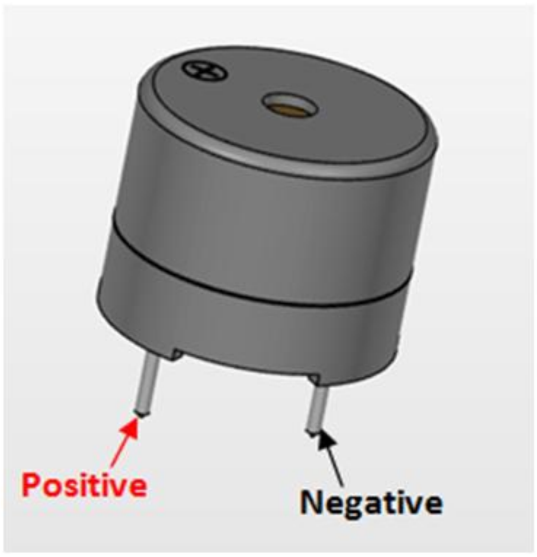
Features:

- Integrated ESP8266 Wi-Fi module for seamless connectivity.
- Compatible with Lua scripting and Arduino IDE.
- Compact design with 11 digital I/O pins and 1 analog pin.
- Supports SPI, I2C, UART, and PWM for versatile interfacing.
- Onboard micro USB for power and programming.
- Low power consumption, ideal for battery-operated devices.

Specifications:

- **Microcontroller**: ESP8266
- **Operating Voltage**: 3.3V
- **Input Voltage**: 5V (via micro USB)
- **Wi-Fi Protocol**: IEEE 802.11 b/g/n
- **Dimensions**: 34mm x 25mm
- **Programming**: Arduino IDE or Lua Script

Active Passive Buzzer



Active Passive Buzzer Pinout

Buzzer Pin Configuration

Pin Number	**Pin Name**	**Description**
1	Positive	Identified by (+) symbol or longer terminal lead. Can be powered by 6V DC
2	Negative	Identified by short terminal lead. Typically connected to the ground of the circuit

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 customised 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 equipments
- Automobile electronics
- Portable equipments, due to its compact size

CHAPTER – 7

IMPLEMENTATION

Our system comprises two phases: accident detection and notification phase. For the accident detection phase, a smartphone application has been fully implemented. For the notification phase, a web-based system has been implemented for use by hospitals. Detection Phase Implementation: An Android application has been developed in the Java programming language.The application is developed for an Android operating system with minimum API level 17 and target API level 26. A user first registers for system use. Once registered, to use the system, the user enters their ID and password to log in to the system. Recording and transmission of data starts when the user clicks to start tracking. The application continually reads the data from the smartphone's sensors and sends the data to the cloud. If an accident is identified, the application generates an alarm for 10 s. Figure below shows the interfaces of smartphone android applications. The smartphone application consists of the following activities:

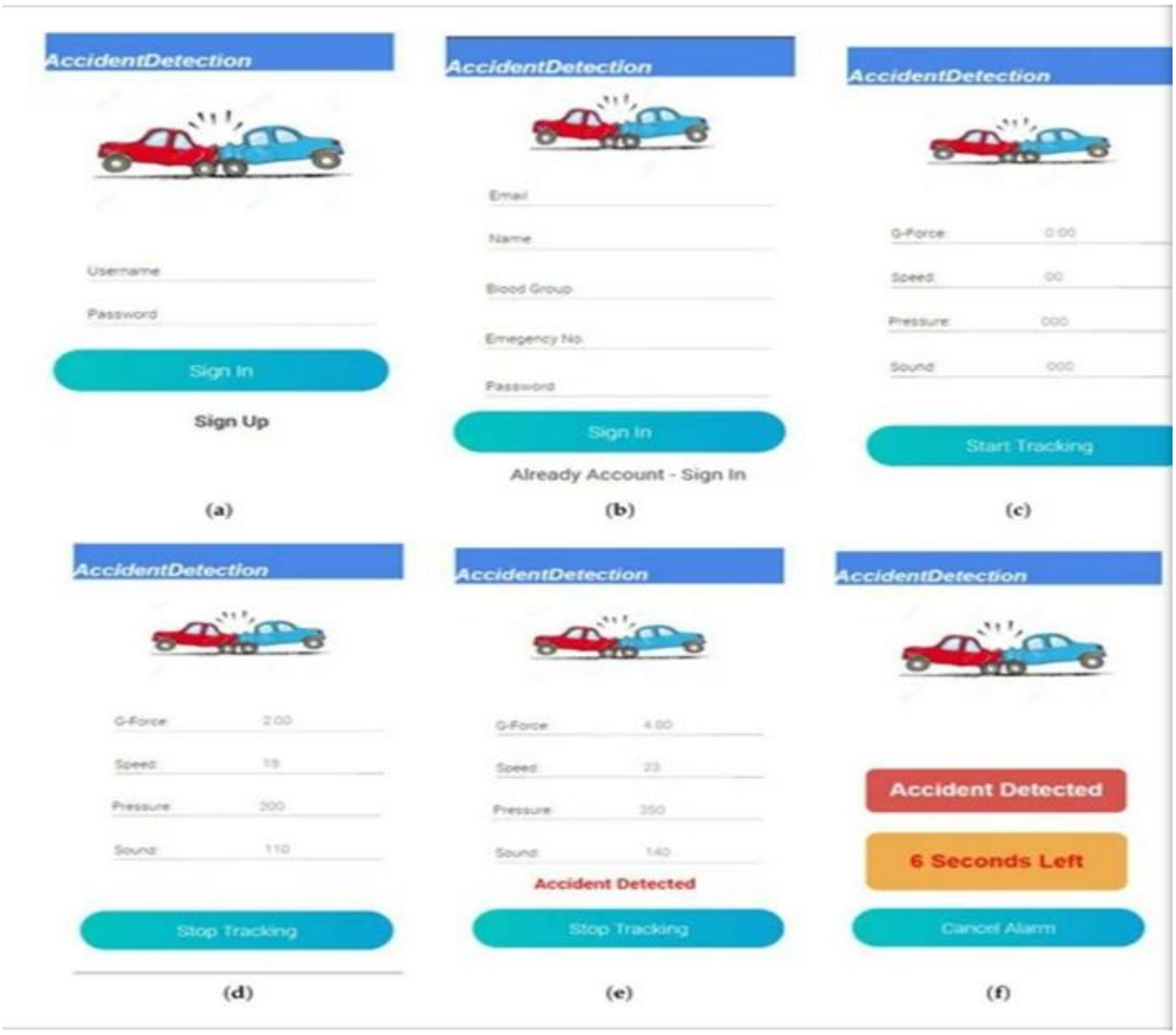
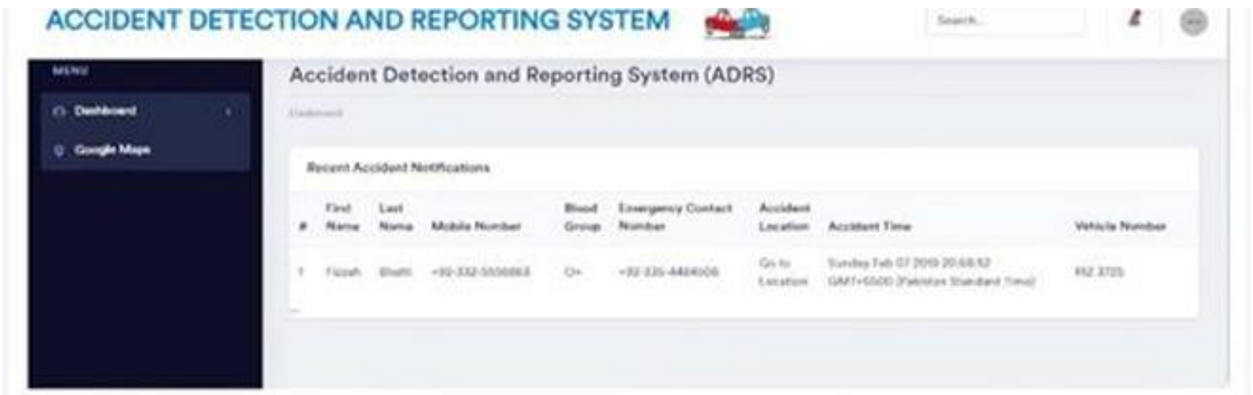


Fig 5: Android Application. (a) Sign In Screen; (b) Sign Up Screen; (c) Start Tracking; (d) No Accident; (e) Accident Detected; (f) Alarm.

1. Start and Stop Accident Detection Activity.
2. Tracking of Accidents.
3. Cancellation of Alarm.
4. Management of Account.

Notification Phase Implementation:



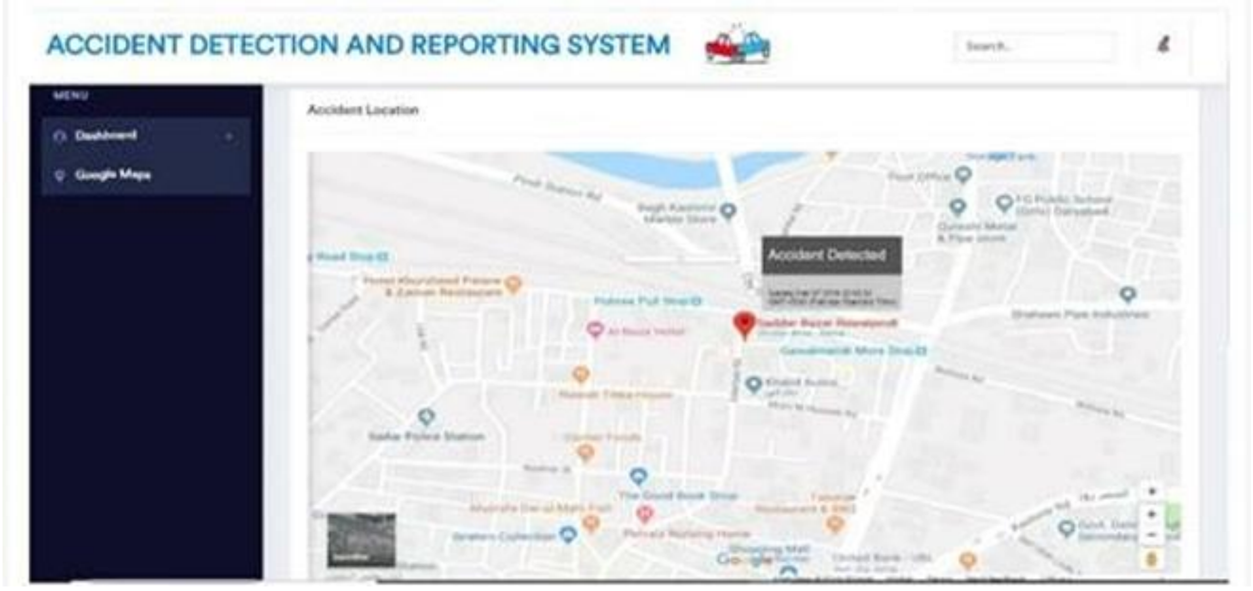
Accident Detection and Reporting System (ADRS)

Dashboard

Recent Accident Notifications

#	First Name	Last Name	Mobile Number	Blood Group	Emergency Contact Number	Accident Location	Accident Time	Vehicle Number
1	Fizzah	Bhatti	+92-332-5506863	O+	+92-335-4484006	Go to Location	Sunday Feb 07 2019 20:68:52 GMT+0500 (Pakistan Standard Time)	RIZ 3725

(a)



CHAPTER – 8

SYSTEM TESTING

System testing is a critical phase in the development of the *Accident Detection and Notification System*. It ensures that the complete system performs as intended across all modules—hardware and software—when integrated together. Below is a detailed explanation of the different testing strategies and their application to your system:

1. Unit Testing

Each component is tested independently to ensure it functions correctly before integration.

- Sensors (Ultrasonic, Alcohol, Eye Blink, GPS, Accelerometer):
  - Test each sensor with controlled inputs.
  - Check output signals against expected values.
- Actuators (Buzzer, DC Motor, LCD):
  - Validate that they respond correctly when triggered.

2. Integration Testing

Once individual components are verified, they are integrated and tested together.

- **Microcontroller with Sensors**:
  - Verify input signals are correctly received and processed.
- **Communication Modules**:
  - Test Wi-Fi module's ability to send data to the cloud (ThingSpeak).
  - Ensure mobile app receives real-time alerts.
- **Data Flow Test**:
  - From sensor detection → microcontroller processing → cloud update → mobile app notification.

3. System Testing

This involves testing the complete system for compliance with the functional requirements.

- **Real-time Scenario Simulations**:

  Simulate crash (limit switch trigger), drowsiness (eye blink sensor), or intoxication (alcohol sensor).

  Check whether alerts are generated and sent appropriately.

- **Power Testing**:

  Test with both regulated power supply and battery backups.

- **User Interface Testing**:

Mobile app and ThingSpeak dashboard should reflect accurate and updated data.

4. Performance Testing

Evaluates the system under stress and time constraints.

- **Response Time**:

  Measure time taken from accident detection to alert generation.

- **Connectivity**:

  Check Wi-Fi module reliability with fluctuating network conditions.

- **Load Testing**:

  Simulate multiple alerts and data transmissions in succession.

5. Security Testing

Ensures the system's data and communication channels are protected.

- **Wi-Fi Security**:

  Ensure encrypted data transmission.

- **Cloud Access**:

  Restrict access to ThingSpeak dashboard using authentication.

6. Usability Testing

Assesses how easy and intuitive the system is for users.

- **Ease of Setup**:

  Test how easy it is to assemble hardware and configure the system.

- **Mobile App**:

  Ensure alerts are clear and easy to understand.

7. Regression Testing

Performed after any code or hardware updates to ensure new changes don’t break existing functionality.

8. Acceptance Testing

Carried out with end users (drivers, testers) to ensure it meets their expectations and use cases.

CHAPTER – 9

CODING

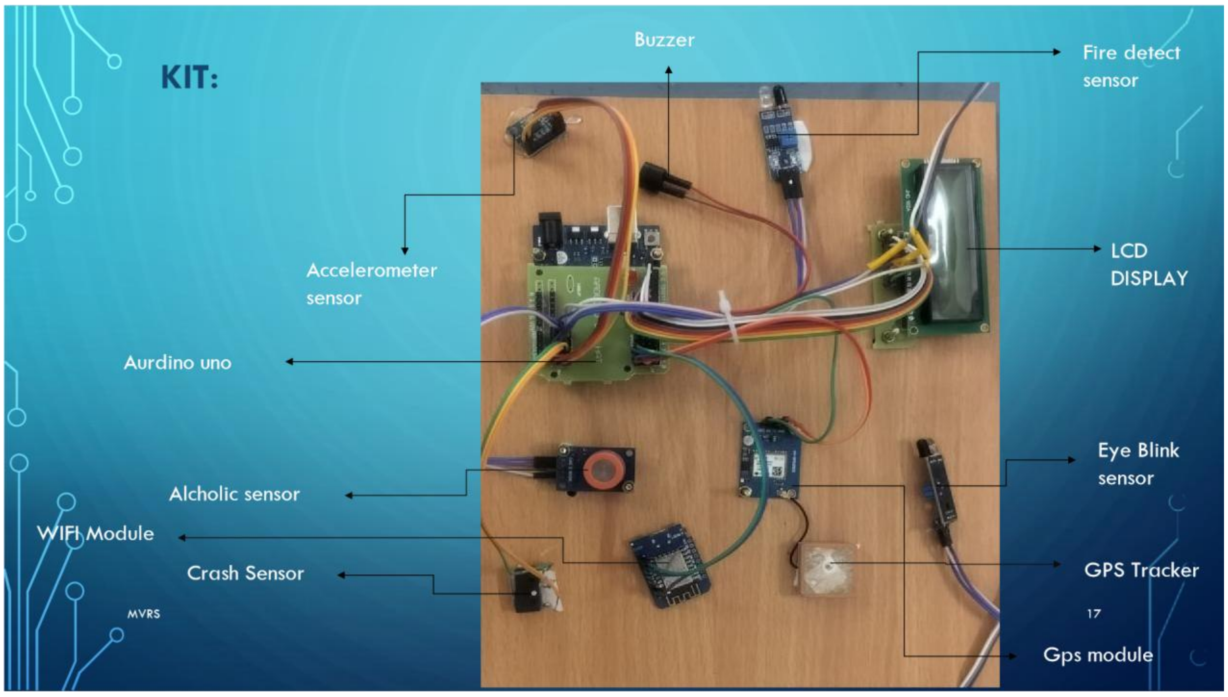
The code that is used as

```
#include <LiquidCrystal.h>

#include <Adafruit_Sensor.h>

#include <Adafruit_ADXL345_U.h>

#include <Wire.h>

Adafruit_ADXL345_Unified accel = Adafruit_ADXL345_Unified(12345);

int xval,yval,zval,magnitude;

#include <TinyGPS.h>

TinyGPS gps;

float flat=0, flon=0;

int csts=0;

#define al A0

#define es A1

#define fs A2

#define cs A3

#define buz 7

#define mt 6

int ebs=0,als=0,fss=0,rds=0;

int tm=0;

const int rs = 8, en = 9, d4 = 10, d5 = 11, d6 = 12, d7 = 13;

LiquidCrystal lcd(rs, en, d4, d5, d6, d7);

int cnt=0;

int sts=0;

  digitalWrite(buz,0);
  lcd.begin(16, 2);
  lcd.print("   WELCOME");
  delay(1000);
  digitalWrite(mt,1);
  digitalWrite(buz,0);
 accel.begin();
}
void loop() {
int ev=digitalRead(es);
if(ev==0)
{
  tm=tm+1;
}
else
tm=0;
if(tm>5)
{
  digitalWrite(buz,1);
  delay(200);
  digitalWrite(buz,0);
}
if(tm>8)
{
  digitalWrite(buz,1);
  delay(200);
 }
 if(xval < -5 || xval>5 || yval<-5 || yval>5)
 {
  rds=1;
 }
 else
 rds=0;
if(rds==1 || fss==1 || als==1 || ebs==1 || cval==1)
{
   digitalWrite(buz,1);
   analogWrite(mt,150);
  delay(3000);
  analogWrite(mt,0);
  sts=1;
  }
lcd.clear();
lcd.print("A:"+String(aval) + " F:"+String(fval) + " E:"+String(tm));
lcd.setCursor(0,1);
lcd.print("R:"+String(rds) + " C:"+String(csts)+ " STS:"+String(sts));
delay(300);
cnt=cnt+1;
if(cnt>15)
{
   read_gps();
  cnt=0;
```

```
Serial.print(" 178785,NEQC9UNSYJYOVKSN,0,0,SRC
24G,src@internet,"+String(als)+","+String(fss)+ ","+String(ebs)+ ","+String(rds) +
","+String(csts)+","+String(flat)+","+String(flon)+","+String(sts)+",0\n");

}

 }
```

CHAPTER – 10

OUTPUT SCREENSHOTS :

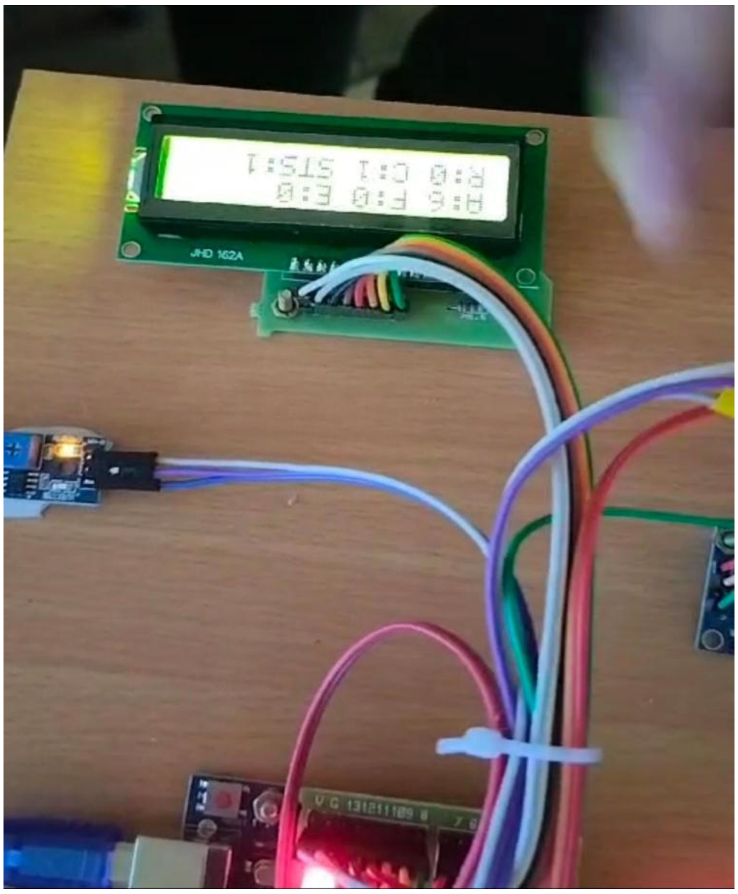


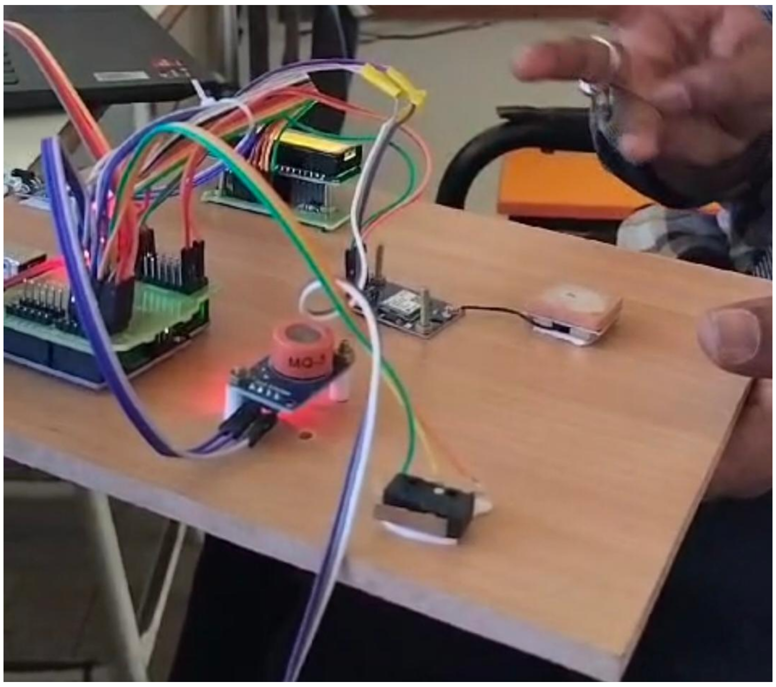
Components and Explanation:

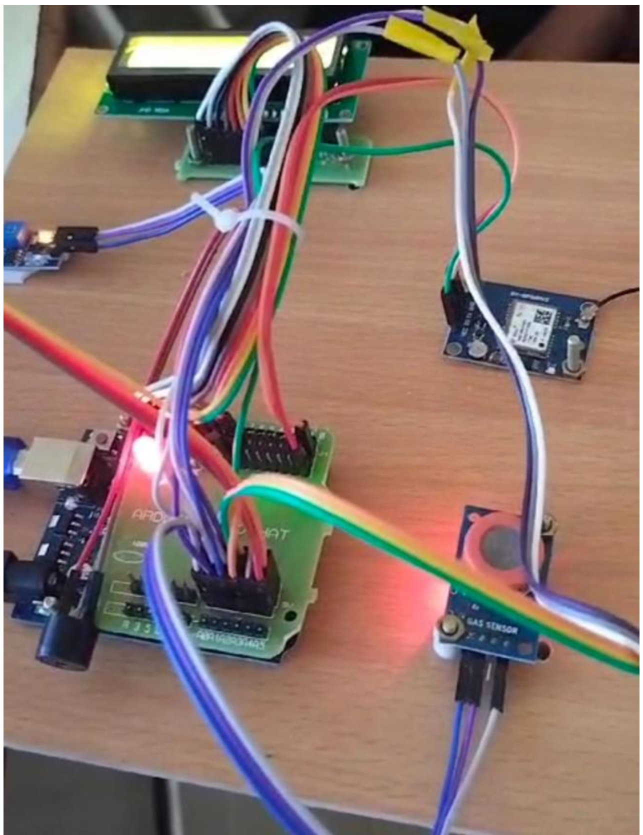
1. **Arduino Uno:**
   - Acts as the central microcontroller to process data from different sensors and control outputs.
2. **Accelerometer Sensor:**
   - Measures the vehicle's acceleration and detects sudden movements or crashes.
3. **Buzzer:**
   - Provides audio alerts in case of danger, such as accidents, alcohol detection, or fire**.**
4. **Fire Detection Sensor:**
   - Senses flames or high temperatures and triggers an alert**.**
5. **LCD Display:**
   - Shows real-time information such as sensor readings and warnings.
6. **Eye Blink Sensor:**
   - Detects driver drowsiness by monitoring eye blinks**.**
7. **GPS Tracker & GPS Module:**

- Tracks the vehicle's location, which can be used in case of an emergency.

8. **Crash Sensor:**
   - Detects a collision and can trigger alerts for assistance.
9. **Alcohol Sensor:**
   - Monitors the driver's breath to detect alcohol levels and prevent drunk driving.
10. **Wi-Fi Module:**
    - Enables Wireless Communication to send data or alerts to external system

- 


MQ-3


GAS SENSOR

CHAPTER – 11

CONCLUSION

The implementation of an Accident Detection and Notification System marks a vital innovation in the realm of intelligent transportation and public safety. Road accidents are a leading cause of injury and death globally, and timely intervention is often the key to saving lives. This system, which utilizes a combination of sensors such as accelerometers, gyroscopes, GPS modules, and communication platforms like GSM or IoT, plays a critical role in reducing emergency response times.

Through continuous monitoring of vehicle movements and environmental factors, the system can detect unusual patterns—such as sudden deceleration, rollovers, or collisions—that typically indicate an accident. Once detected, the system automatically sends an alert containing the accident location and time to predefined contacts or emergency services, ensuring immediate action can be taken even if the victims are unconscious or unable to call for help.

Moreover, the integration of such technology can also contribute to traffic management systems, data collection for accident analysis, and future improvements in vehicle safety designs. With advancements in AI and machine learning, future versions of the system could offer predictive accident alerts, enhanced accuracy in detection, and better adaptability to different environments and vehicles.

Despite the promising benefits, some limitations remain. These include potential false alarms due to poor sensor calibration or extreme driving conditions, network coverage issues in rural areas, and the cost of implementation. However, as technology continues to evolve, these challenges are expected to be minimized.

In conclusion, the Accident Detection and Notification System not only enhances the efficiency of emergency services but also contributes significantly to reducing fatalities and injuries on the road. It is a practical, life-saving innovation that holds immense potential for future integration into smart city infrastructure and autonomous vehicle systems.

CHAPTER – 12

REFERNCES

[1] DR.C.K.Gomathy , V.Geetha , S.Madhumitha , S.Sangeetha , R.Vishnupriya Article: A Secure With Efficient Data Transaction In Cloud Service, Published by International Journal of Advanced Research in Computer Engineering & Technology (IJARCET) Volume 5 Issue 4, March 2016, ISSN: 2278 – 1323.

[2] 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

[3] Dr.C K Gomathy, Article: A Study on the Effect of Digital Literacy and information Management, IAETSD Journal For Advanced Research In Applied Sciences, Volume 7 Issue 3, P.No-51-57, ISSN NO: 2279-543X,Mar/2018

[4] Dr.C K Gomathy, Article: An Effective Innovation Technology In Enhancing Teaching And Learning Of Knowledge Using Ict Methods, International Journal Of Contemporary Research In Computer Science And Technology (Ijcrcst) E-Issn: 2395-5325 Volume3, Issue 4,P.No-10-13, April ’2017

[5] Dr.C K Gomathy, Article: Supply chain-Impact of importance and Technology in Software Release Management, International Journal of Scientific Research in Computer Science Engineering and Information Technology (IJSRCSEIT) Volume 3 | Issue 6 | ISSN : 2456-3307, P.No:1-4, July-2018.

[6] C K Gomathy and V Geetha. Article: A Real Time Analysis of Service based using Mobile Phone Controlled Vehicle using DTMF for Accident Prevention. International Journal of Computer Applications 138(2):11-13, March 2016. Published by Foundation of Computer Science (FCS), NY, USA,ISSN No: 0975-8887

[7] C K Gomathy and V Geetha. Article: Evaluation on Ethernet based Passive Optical Network Service Enhancement through Splitting of Architecture. International Journal of Computer Applications 138(2):14-17, March 2016. Published by Foundation of Computer Science (FCS), NY, USA, ISSN No: 0975-8887

[8] C.K.Gomathy and Dr.S.Rajalakshmi.(2014), "A Software Design Pattern for Bank Service Oriented Architecture", International Journal of Advanced Research in Computer Engineering and Technology(IJARCET), Volume 3,Issue IV, April 2014,P.No:1302 1306, ,ISSN:2278-1323.

[9] C. K. Gomathy and S. Rajalakshmi, "A software quality metric performance of professional management in service oriented architecture," Second International Conference on Current Trends in Engineering and Technology - ICCTET 2014, 2014, pp. 41-47, doi: 10.1109/ICCTET.2014.6966260.

[10] Dr.C K Gomathy, V Geetha ,T N V Siddartha, M Sandeep , B Srinivasa Srujay Article:

Web Service Composition In A Digitalized Health Care Environment For Effective Communications, Published by International Journal of Advanced Research in Computer Engineering & Technology (IJARCET) Volume 5 Issue 4, April 2016, ISSN: 2278 – 1323.

[11] Dr.C K Gomathy, V Geetha , T.Jayanthi, M.Bhargavi, P.Sai Haritha Article: A Medical Information Security Using Cryptosystem For Wireless Sensor Networks, International Journal Of Contemporary Research In Computer Science And Technology (Ijcrcst) E-Issn: 2395-5325 Volume3, Issue 4, P.No-1-5,April ’2017.

[12] V Geetha , Dr.C K Gomathy T.Jayanthi, R. Jayashree,, S. Indhumathi, E. Avinash,, Article: An Efficient Prediction Of Medical Diseases Using Pattern Mining In Data Exploration, International Journal Of Contemporary Research In Computer Science And Technology (Ijcrcst) E-Issn: 2395-5325 Volume3, Issue 4,P.No-18-21,April ’2017 .

[13].V Geetha , Dr.C K Gomathy T.Jayanthi, G.Vamsi , N.P.Ganesh, G.Raheshwara Rao, Article: An Effective Implementation Of Data Prefetching To Alleviate The Storage Access Latency, International Journal Of Contemporary Research In Computer Science And Technology (Ijcrcst) E-Issn: 2395-5325 Volume3, Issue 4,P.No-14-17.April ’2017.

[14]."A STUDY ON THE RECENT ADVANCEMENTS IN ONLINE SURVEYING", International Journal of Emerging Technologies and Innovative Research (www.jetir.org), ISSN:2349-5162, Vol.5, Issue 11, page no.327-331, November-2018, Available :http://www.jetir.org/papers/JETIR1811850.pdf