ATHARVA EDUCATIONAL TRUST

ATHARVA COLLEGE OF ENGINEERING MALAD, MUMBAI

IOT MINI PROJECT REPORT

TITLE

"AcciLERT-Accident Alert System"

SUBMITTED BY:

Ravikumar Chaurasia(9)
Bhavya Gada(18)
Shivangkumar Gandhi(19)

UNDER THE GUIDANCE OF:
Prof. Nileema Pathak

DEPARTMENT OF INFORMATION TECHNOLOGY
(2020-2021)

CERTIFICATE

ATHARVA COLLEGE OF ENGINEERING

MALAD (W), MUMBAI 400 095

YEAR – 2020-2021

This is to certify that the following students of Information Technology Department

Ravikumar Chaurasia(9)
Bhavya Gada(18)
Shivangkumar Gandhi(19)

have submitted the mini project report titled

AcciLERT - ACCIDENT ALERT SYSTEM

PROJECT GUIDE **H.O.D.** **PRINCIPAL**

INTERNAL EXAMINER **COLLEGE SEAL** **EXTERNAL EXAMINER**

Approval for Mini Project Report

This project report entitled AcciLERT-ACCIDENT ALERT SYSTEM by Ravikumar Chaurasia (9), Bhavya Gada (18) and Shivangkumar Gandhi (19) is approved for the IOT Mini project in 5th SEM Information Technology.

Internal Examiners_____

Project Guide. _____

Declaration

I declare that this written submission represents my ideas in my own words and where others ideas or words have been included, I have adequately cited and referenced the original sources. I also declare that I have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in my submission. I understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

_____

Ravikumar Chaurasia (9)

_____

Bhavya Gada (18)

_____

Shivangkumar Gandhi (19)

Date:

ABSTRACT

AcciLERT(The Accident Detection and Alert System using Arduino) is very sufficient and worthy to be implemented in the vehicle especially in developing countries like Nepal, India, Bangladesh etc. Accidents are increasing due to an increase in the number of vehicles as a result every year the number of deaths is increasing. The Accident Detection and Alert System using Arduino prevents the uncertain death after an accident because this system sends the message alert to the hospital or police station. The message alert includes longitude, latitude (location of accident), in the form of google map link.

The Technologies used in this system can enhance the traffic and are cost efficient to be fit for any car due to its size. This system can efficiently detect an accident on the roads, with the help of input sensors i.e. the vibration module which could be fit inside air bags and the gyroscope accelerometer sensor which helps the system detect sensors.

Though there are cases when there is a false alarm or the accident is minor. So to overcome this the system is built so efficiently that the driver gets 10 seconds to press the false alarm button and deactivate the SMS system that is used to notify nearby Hospitals and Police Stations.

When the accident is not minor and the driver is unconscious, this system sends an sms to the registered relatives on the app as well to the nearby hospitals and police station immediately after waiting for 10 seconds, Hence saving the life of the commuters in the vehicle.

INDEX

Chapter I

LET'S GET STARTED

1.1 Introduction

The Accident Alert System is an IoT based project, which by detecting the accident the module sends the Notification to your loved ones, nearby hospitals, police station and provides location to the ambulance. Providing First-Aid by smartly saving time by analyzing your Data and even helps you to fight your accident case by much accurate result.

With the integration of Accident Alert System with AirBag Trigger system, the identification of occurrences of Car Accident will be much more accurate than the other Mobile based projects which detect the accident with the sensors of mobile which are not as accurate as AcciLERT system.

This accident alert system - AcciLERT is by-far the most accurate and the fastest way to provide Second-Aid to any person in an accident. We have reached to this conclusion because -

- By notifying the ambulance services, we are making an autonomous, delay free transport facility for the patient to a nearby hospital. We provide the ambulance service with most accurate coordinates of the incident and also provide a google maps link to navigate to that location in the shortest time possible.
- By notifying the nearby hospital's emergency department, we are making them ready better, we aim to give them more time to prepare for the new patient by notifying them autonomously which reduces the delay to zero percent.
- By notifying the nearby police station's we aim on providing a secure and fast path to the ambulance on its journey to the hospital in the massive traffic of Mumbai. We also provide the police with accurate information about the incident for insurance and investigation purposes.
- By notifying the loved ones we take care of them being informed in a calm way so that there is no panic situation when some police or hospital receptionist calls to inform them, and also provide them more time to prepare for many other things like finance, leaving to look for the patient etc.

1.2 Problem Statement

To Design a module which will help notify the responsible authorities about any serious accidents and then take required automated steps to ease the process of second-aid.

1.3 Aim and Objectives

1.3.1 Aim: To implement AcciLERT-ACCIDENT ALERT SYSTEM by taking into account time saving, convenience factors.

1.3.2 Objectives:

- To minimize human efforts.
- To save valuable time.
- To provide information on the accident to responsible authorities.
- Low cost and highly sensitive reliable circuit.
- System can be switched to manual mode whenever required.

1.4 Scope and further applications

Using the AcciLERT module, which is cost efficient, we can detect the accident along with the car's airbag system. Our AcciLERT module is integrated with accelerometer, gyroscope and temperature sensor with settings to change the sensitivity as per required. With all this it becomes easy to calibrate the module.

AcciLERT is connected to the driver's mobile phone which currently sends the SMS to driver's relatives and in future scope it can send to a server, which on further computes and sends the notification to close by hospital, police station, insurance company and many more. With this the critical time is saved in first responding to the victim.

In future we can store the medical past details on our server to save even the operation time, when the victim arrives at the hospital, which can further save a lot of time beforehand.

Overall, such a system can also be made for bicycles and motorcycles on further research on this idea.

Chapter II

LITERATURE

2.1 Literature Survey.

Sr No.	Title	Author	Publication	Approach	Link
1.	IoT Based Intelligent System For Vehicle Accident and Detection At Real Time	Vivek Kinage, Piyush Patil	IEEE	The proposed system provides indication of alcohol level in the driver and the obstacles ahead of the car using the sensors which could prevent accidents.	https://ieeexplore.ieee.org/document/9032662
2.	Automobile Black Box System for Accident Analysis	Monisha J Prasad, Arundathi S, Nayana Anil, Harshikha	IEEE	The proposed system makes use of 12 sensors to record the various driving data. The data received from the sensors are stored on the SD card and uses GPS for positioning.	https://ieeexplore.ieee.org/document/7002430
3.	Airbag ECU coupled Vehicle Accident SMS Alert System	Dheeraj Khandelwal, Manoov R	IEEE	The system uses the in-built vibration sensor in the Airbag Electronic Control Unit to detect the abrupt vibrations from the occurrence of an accident and GSM module for SMS	https://ieeexplore.ieee.org/document/8365258

TABLE 2.1 SURVEY

Chapter III
REQUIREMENT ANALYSIS

Hardware Requirements

- MPU6050 Accelerometer and Gyroscope
- Arduino Nano
- HC-05 Bluetooth module
- Jumping wires.
- Push Button
- 10K Ohm resistor
- SW-420 Shock Sensor

Software Requirements

- Arduino IDE.

Hardware - Description

MPU6050 Accelerometer

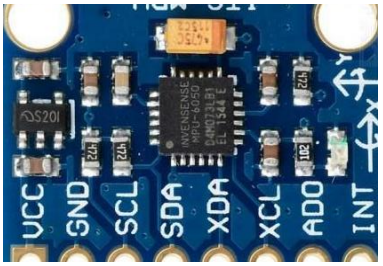


Figure 3.3.1

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-axis 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

The MPU6050 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 parameters of a system or object. This module also has a (DMP) Digital Motion Processor inside it which is powerful enough to perform complex calculation and thus free up the work for Microcontroller.

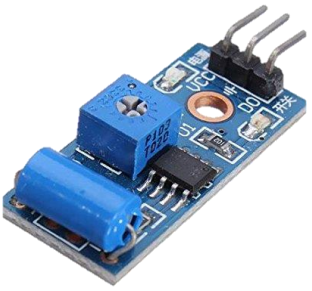
The module also have two auxiliary pins which can be used to interface external IIC modules like a magnetometer, however it is optional. Since the IIC address of the module is configurable more than one MPU6050 sensor can be interfaced to a Microcontroller using the AD0 pin. This module also has well documented and revised libraries available hence it's very easy to use with famous platforms like Arduino. So if you are looking for a sensor to control motion for your RC Car, Drone, Self-balancing Robot, Humanoid, Biped or something like that then this sensor might be the right choice for you.

The MPU6050 module allows us to read data from it through the IIC bus. Any change in motion will be reflected on the mechanical system which will in turn vary the voltage. Then the IC has a 16-bit ADC which it uses to accurately read these changes in voltage and stores it in the FIFO buffer and makes the INT (interrupt) pin to go high. This means that the data is ready to be read, so we use a MCU to read the data from this FIFO buffer through IIC communication. As easy as it might sound, you may face some problem while actually trying to make sense of the data. However there are lots of platforms like Arduino using which you can start using this module in no time by utilizing the readily available libraries explained below.

Applications

- Used for IMU measurement
- Drones / Quad copters
- Self-balancing robots
- Robotic arm controls
- Humanoid robots
- Tilt sensor
- Orientation / Rotation Detector

SW-420 Vibration Sensor



Pin Name	Description
VCC	The Vcc pin powers the module, typically with +5V
GND	Power Supply Ground
DO	Digital Out Pin for Digital Output.

Vibration Sensor Module Features & Specifications

- Operating Voltage: 3.3V to 5V DC
- Operating Current: 15mA
- Using SW-420 normally closed type vibration sensor
- LEDs indicating output and power
- LM393 based design
- Easy to use with Microcontrollers or even with normal Digital/Analog IC
- With bolt holes for easy installation
- Small, cheap and easily available

This Vibration Sensor Module consists of an SW-420 Vibration Sensor, resistors, capacitor, potentiometer, comparator LM393 IC, Power, and status LED in an integrated circuit. It is useful for a variety of shocks triggering, theft alarm, smart car, an earthquake alarm, motorcycle alarm, etc.

LM393 IC

LM393 Comparator IC is used as a voltage comparator in this vibration sensor module. Pin 2 of LM393 is connected to Preset (10KΩ Pot) while pin 3 is connected to vibration sensor. The comparator IC will compare the threshold voltage set using the preset (pin2) and the Vibration Sensor pin (pin3).

Preset (Trimmer pot)

Using the onboard preset, you can adjust the threshold (sensitivity) of the digital output.

SW-420 Vibration Switch

Vibration switch recognizes the amplitude of the vibration to which it is exposed. The switch response can be electrical contact closure or contact opening. The electrical contact may be either an electromechanical relay or a solid-state device.

How to Use SW-420 Vibration Sensor Module

Vibration sensor module consists of three pins i.e. VCC, GND, and DO. The Digital out pin is connected to the output pin of the LM393 comparator IC. The Internal Circuit diagram of the Temperature sensor module is given below.

Using the Vibration sensor module with the microcontroller is very easy. Connect the Digital Output pin of the module to the Digital pin of Microcontroller. Connect VCC and GND pins to 5V and GND pins of the Microcontroller.

Applications of Vibration Sensor Module

- Shocks triggering
- Theft alarm
- Smart car
- Earthquake alarm
- Motorcycle alarm

HC05 Bluetooth Module

Pin Number	Pin Name	Description
1	Enable / Key	This pin is used to toggle between Data Mode (set low) and AT command mode (set high). By default it is in Data mode
2	Vcc	Powers the module. Connect to +5V Supply voltage
3	Ground	Ground pin of module, connect to system ground.
4	TX – Transmitter	Transmits Serial Data. Everything received via Bluetooth will be given out by this pin as serial data.
5	RX – Receiver	Receive Serial Data. Every serial data given to this pin will be broadcasted via Bluetooth
6	State	The state pin is connected to on board LED, it can be used as a feedback to check if Bluetooth is working properly.
7	LED	Indicates the status of Module ● Blink once in 2 sec: Module has entered Command Mode ● Repeated Blinking: Waiting for connection in Data Mode ● Blink twice in 1 sec: Connection successful in Data Mode

8	Button	Used to control the Key/Enable pin to toggle between Data and command Mode

HC-05 Default Settings

Default Bluetooth Name: “HC-05”

Default Password: 1234 or 0000

Default Communication: Slave

Default Mode: Data Mode

Data Mode Baud Rate: 9600, 8, N, 1

Command Mode Baud Rate: 38400, 8, N, 1

Default firmware: LINVOR

HC-05 Technical Specifications

- Serial Bluetooth module for Arduino and other microcontrollers
- Operating Voltage: 4V to 6V (Typically +5V)
- Operating Current: 30mA
- Range: <100m
- Works with Serial communication (USART) and TTL compatible
- Follows IEEE 802.15.1 standardized protocol
- Uses Frequency-Hopping Spread spectrum (FHSS)
- Can operate in Master, Slave or Master/Slave mode
- Can be easily interfaced with Laptop or Mobile phones with Bluetooth
- Supported baud rate: 9600,19200,38400,57600,115200,230400,460800.

The HC-05 is a very cool module which can add two-way (full-duplex) wireless functionality to your projects. You can use this module to communicate between two microcontrollers like Arduino or communicate with any device with Bluetooth functionality like a Phone or Laptop. There are many android applications that are already available which makes this process a lot easier. The module communicates with the help of USART at 9600 baud rate hence it is easy to interface with any microcontroller that supports USART. We can also configure the default values of the module by using the command mode. So if you are looking for a Wireless module that could transfer data from your computer or mobile phone to a microcontroller or vice versa then this module might be the right choice for you. However do not expect this module to transfer multimedia like photos or songs; you might have to look into the CSR8645 module for that.

The HC-05 has two operating modes, one is the Data mode in which it can send and receive data from other Bluetooth devices and the other is the AT Command mode where the default device settings can be changed. We can operate the device in either of these two modes by using the key pin as explained in the pin description.

It is very easy to pair the HC-05 module with microcontrollers because it operates using the Serial Port Protocol (SPP). Simply power the module with +5V and connect the Rx pin of the module to the Tx of MCU and Tx pin of module to Rx of MCU as shown in the figure below

During power up the key pin can be grounded to enter into Command mode, if left free it will by default enter into the data mode. As soon as the module is powered you should be able to discover the Bluetooth device as "HC-05" then connect with it using the default password 1234 and start communicating with it. The name password and other default parameters can be changed by entering into the

Applications

1. Wireless communication between two microcontrollers
2. Communicate with Laptop, Desktops and mobile phones
3. Data Logging application
4. Consumer applications
5. Wireless Robots
6. Home Automation

Arduino Nano

Arduino Nano Pin Configuration

Pin Category	Pin Name	Details
Power	**Vin, 3.3V, 5V, GND**	**Vin:** Input voltage to Arduino when using an external power source (6-12V). **5V:** Regulated power supply used to power microcontroller and other components on the board. **3.3V:** 3.3V supply generated by on-board voltage regulator. Maximum current draw is 50mA. **GND:** Ground pins.
Reset	**Reset**	Resets the microcontroller.
Analog Pins	**A0 – A7**	Used to measure analog voltage in the range of 0-5V
Input/Output Pins	**Digital Pins D0 - D13**	Can be used as input or output pins. 0V (low) and 5V (high)
Serial	Rx, **Tx**	Used to receive and transmit TTL serial data.
External Interrupts	2, 3	To trigger an interrupt.
PWM	3, 5, 6, 9, 11	Provides 8-bit PWM output.
SPI	10 (SS), 11 (MOSI), 12 (MISO) and 13 (SCK)	Used for SPI communication.
Inbuilt LED	**13**	To turn on the inbuilt LED.
IIC	A4 (SDA), A5 (SCA)	Used for TWI communication.

AREF	**AREF**	To provide reference voltage for input voltage.

The Arduino board is designed in such a way that it is very easy for beginners to get started with microcontrollers. This board, especially breadboard friendly, is very easy to handle the connections. Let's start with powering the Board.

Powering you Arduino Nano:

There are totally three ways by which you can power your Nano.

USB Jack: Connect the mini USB jack to a phone charger or computer through a cable and it will draw power required for the board to function

Vin Pin: The Vin pin can be supplied with a unregulated 6-12V to power the board. The on-board voltage regulator regulates it to +5V

+5V Pin: If you have a regulated +5V supply then you can directly provide this o the +5V pin of the Arduino.

Input/output: There are totally 14 digital Pins and 8 Analog pins on your Nano board. The digital pins can be used to interface sensors by using them as input pins or drive loads by using them as output pins. A simple function like pinMode() and digitalWrite() can be used to control their operation. The operating voltage is 0V and 5V for digital pins. The analog pins can measure analog voltage from 0V to 5V using any of the 8 Analog pins using a simple function liken analogRead()

These pins apart from serving their purpose can also be used for special purposes which are discussed below:

- Serial Pins 0 (Rx) and 1 (Tx): Rx and Tx pins are used to receive and transmit TTL serial data. They are connected with the corresponding ATmega328P USB to TTL serial chip.
- External Interrupt Pins 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.
- PWM Pins 3, 5, 6, 9 and 11: These pins provide an 8-bit PWM output by using analogWrite() function.
- SPI Pins 10 (SS), 11 (MOSI), 12 (MISO) and 13 (SCK): These pins are used for SPI communication.
- In-built LED Pin 13: This pin is connected with an built-in LED, when pin 13 is HIGH – LED is on and when pin 13 is LOW, its off.
- I2C A4 (SDA) and A5 (SCA): Used for IIC communication using Wire library.
- AREF: Used to provide reference voltage for analog inputs with analogReference() function.

- Reset Pin: Making this pin LOW, resets the microcontroller.

Software - Description

Arduino IDE

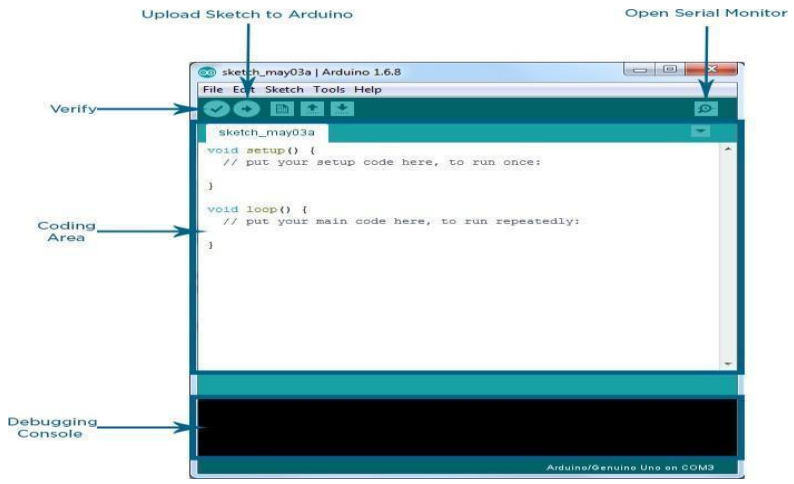


Figure 3.3.6

The Arduino integrated development environment (IDE**)** is a cross-platform application (for Windows, macOS, Linux) that is written in the programming language Java. It is used to write and upload programs to the Arduino board. The Arduino IDE supports the languages C and C++ using special rules of code structuring. The Arduino IDE supplies a software library from the Wiring project, which provides many common input and output procedures.

The Arduino IDE supplies a software library from the Wiring project, which provides many common input and output procedures. User-written code only requires two basic functions, for starting the sketch and the main program loop, that are compiled and linked with a program stub *main()* into an executable cyclic executive program with the GNU toolchain, also included with the IDE distribution. The Arduino IDE employs the program *avrdude* to convert the executable code into a text file in hexadecimal encoding that is loaded into the Arduino board by a loader program in the board's firmware. By default, avrdude is used as the uploading tool to flash the user code onto official Arduino boards

Chapter IV

REPRESENTATION

4.1 Block diagram

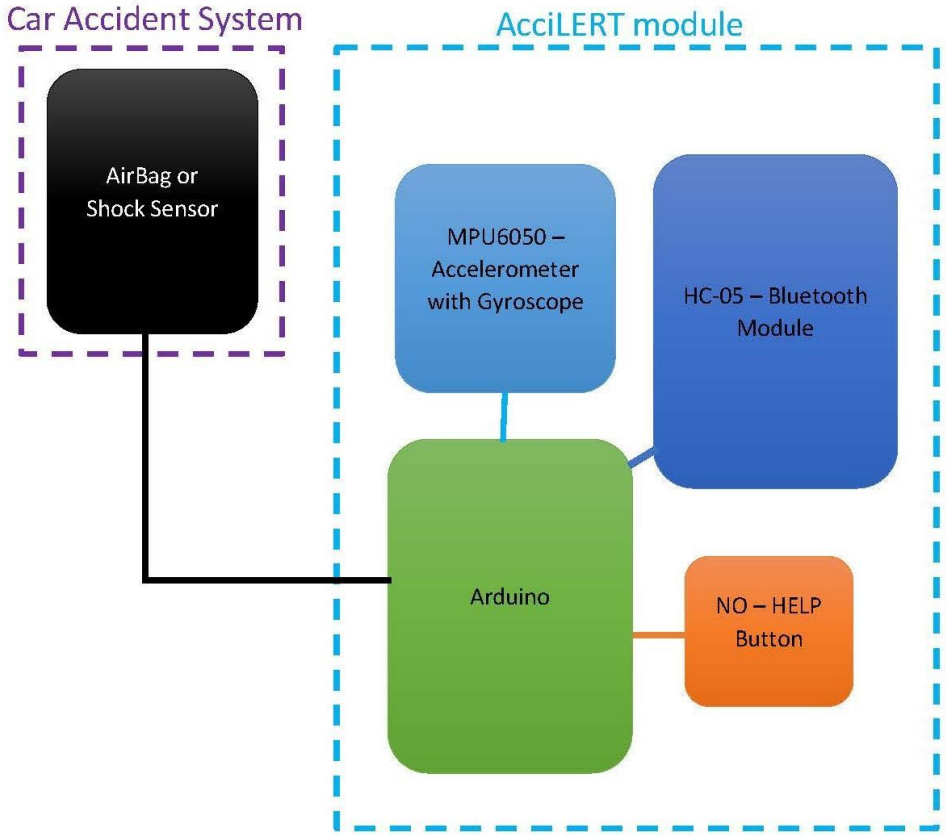


FIG 4.1 BLOCK DIAGRAM

4.1.1 Explanation – The block diagram of the AcciLERT – ACCIDENT ALERT SYSTEM is shown in Fig. The circuit comprises two Arduino nano, an accelerometer and gyroscope sensor, a shock sensor, a Bluetooth, a push button. You can power the Arduino nano using a 7V to 12V wall wart or plug-in adaptor or a separate 9-12V battery. This Arduino nano is connected to the MPU6050 which does the important job, to identify the accident's level. After the sensors collect the data, it sends it to the Arduino nano for further use, it also checks for shock sensor data to make the data more precious.

Once it is verified that the accident is of assist-able's level then it sends the warning to Arduino to send the warning to the driver's device after waiting for 10-15 secs. And if in that interval it receives any input from the NO-HELP button (i.e. push button) it will null that warning and restart the AcciLERT module for regular use.

After it sends the command to send the sms to the driver's phone, the remaining process is on the hand on the driver's device which has our AcciLERT app installed to further send the sms with the accident location.

For Future, we are planning to setup a receiving server which will receive that sms and compute the nearest possible hospital and other essential facility, for better service. On all of this the medical past data, so that the victim can be assisted as soon as possible.

4.2. Circuit Diagram

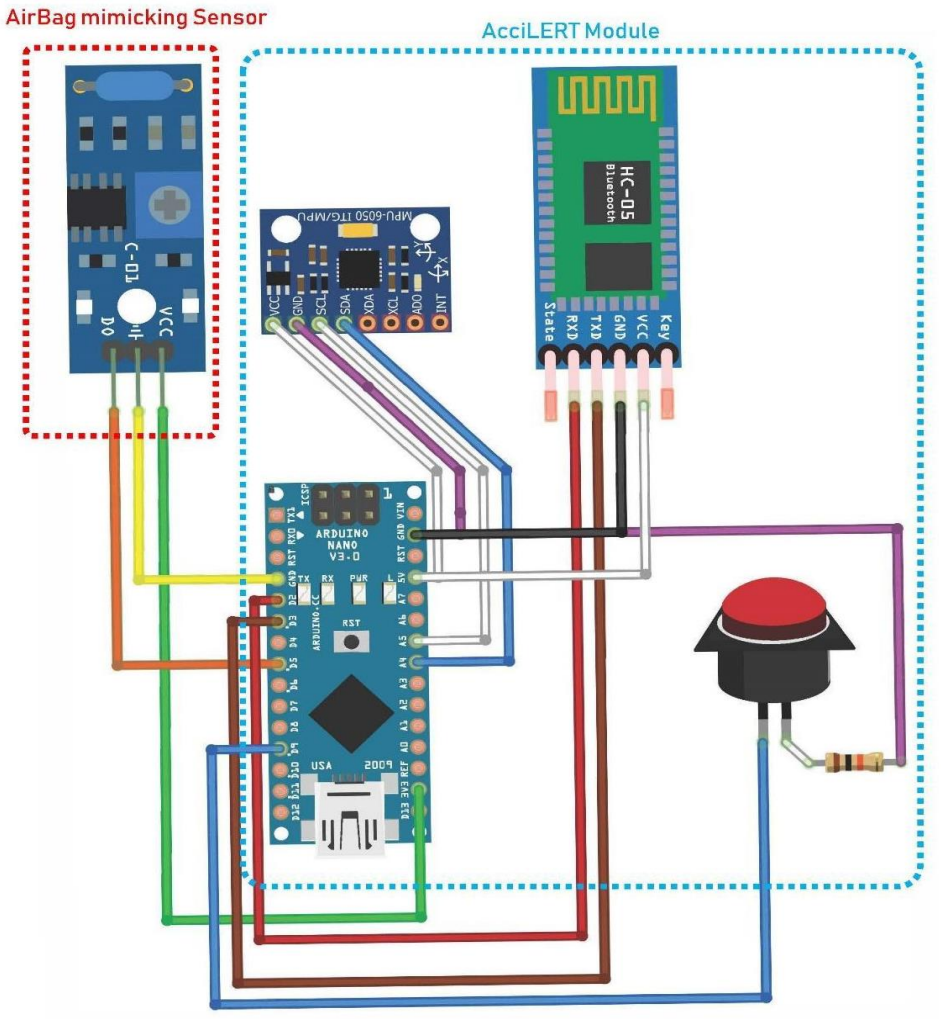


FIG 4.2 CIRCUIT DIAGRAM

In this circuit diagram we see that an Arduino nano is connected to a shock sensor, accelerometer and gyroscope, a Bluetooth module, shock sensor and a push button. This Arduino nano will be powered by a battery. The MPU6050 module is connected to A4 and A5 analog pins, while the other 2 pins are connected to 5V and GND. The Shock sensor is connected to D5 and other 2 pins to 5V and GND.
The HC-05's TX and RX pin is connected to D2 and D3, and the other 2 pins to 5V and GND. Lastly the PUSH button is connected to D4 to receive it's input with a 10K Ohm resistor.

Chapter V

IMPLEMENTATION

5.1. Working

To represent the working of this project, let us look stepwise.

5.1.1 Steps:

1. Detection of Accident is done using both the Shock Sensor and our AcciLERT module(which include Accelerometer and Gyroscope data). The data of all the sensors are analysed and a precise detection is done.

2. Once it's detected the confirmation is done within the interval of 10-15 buffer sec, in this interval if the driver presses the NO-HELP button then the alert command is suspended, and the AcciLERT module gets restarted. In this interval if the driver doesn't give a feedback, then the driver may be unconscious or need the help.

3. The arduino nano sends the command through HC-05 bluetooth module to the driver's device connected.

4. Every driver has a unique number provided, which will be used to store the medical data of the driver in case to save more time, while operating by doctor. The AcciLERT app sends the SMS to the Server and to the relative's mobile number with a Google Map link for navigation.

5. For Future, we are planning to setup a receiving server which will receive that sms and compute the nearest possible hospital and other essential facility, for better service. On all of this the medical past data, so that the victim can be assisted as soon as possible.

5.1.2 Code:

```
#include <Wire.h>
#include <SoftwareSerial.h>
#include "MPU6050_6Axis_MotionApps20.h"
SoftwareSerial BTserial(3, 2); //   TX|RX
#define button 9
#define shock_pin 5
static boolean flag = true;
MPU6050 mpu;
// MPU6050 Slave Device Address
const uint8_t MPU6050SlaveAddress = 0x68;

// Select SDA and SCL pins for I2C communication
const uint8_t scl = A5;
const uint8_t sda = A4;

// sensitivity scale factor respective to full scale setting provided in datasheet
const uint16_t AccelScaleFactor = 16384;
const uint16_t GyroScaleFactor = 131;

// MPU6050 few configuration register addresses
const uint8_t MPU6050_REGISTER_SMPLRT_DIV   =  0x19;
const uint8_t MPU6050_REGISTER_USER_CTRL    =  0x6A;
const uint8_t MPU6050_REGISTER_PWR_MGMT_1   =  0x6B;
const uint8_t MPU6050_REGISTER_PWR_MGMT_2   =  0x6C;
const uint8_t MPU6050_REGISTER_CONFIG       =  0x1A;
const uint8_t MPU6050_REGISTER_GYRO_CONFIG  =  0x1B;
const uint8_t MPU6050_REGISTER_ACCEL_CONFIG =  0x1C;
const uint8_t MPU6050_REGISTER_FIFO_EN      =  0x23;
const uint8_t MPU6050_REGISTER_INT_ENABLE   =  0x38;
const uint8_t MPU6050_REGISTER_ACCEL_XOUT_H =  0x3B;
const uint8_t MPU6050_REGISTER_SIGNAL_PATH_RESET  = 0x68;

int16_t AccelX, AccelY, AccelZ, Temperature, GyroX, GyroY, GyroZ;

int buttonState = 0;
static boolean flag_button=false;
float time=0.0;
static boolean acci_detec=false;
float x=0.0;

void setup() {
  Serial.begin(9600);
  BTserial.begin(9600);
  Wire.begin();
  // sgps.begin(9600);
  Serial.println(F("Initializing I2C devices..."));
  mpu.initialize();

  // verify connection
  Serial.println(F("Testing device connections..."));
  Serial.println(mpu.testConnection() ? F("MPU6050 connection successful") : F("MPU6050
connection failed"));

  // load and configure the DMP
  Serial.println(F("Initializing DMP..."));
  mpu.dmpInitialize();
```

```
  // supply your own gyro offsets here, scaled for min sensitivity
  mpu.setXGyroOffset(220);
  mpu.setYGyroOffset(76);
  mpu.setZGyroOffset(-85);
  mpu.setZAccelOffset(1788);
  pinMode(command,OUTPUT);
  pinMode(buzzer, OUTPUT);
  pinMode(button, INPUT);
  pinMode(shock_pin, INPUT);
}

long TP_init() {
  delay(10);
  long measurement = pulseIn (shock_pin, HIGH); //wait for the pin to get HIGH and returns
measurement
  return measurement;
}

void loop()
{
  if(flag_button)
  {
     buttonState = digitalRead(button);
  }

  if (flag)
  {
    double Ax, Ay, Az, T, Gx, Gy, Gz, shock;

    Read_RawValue(MPU6050SlaveAddress, MPU6050_REGISTER_ACCEL_XOUT_H);

    //divide each with their sensitivity scale factor
    Ax = (double)AccelX / AccelScaleFactor;
    Ay = (double)AccelY / AccelScaleFactor;
    Az = (double)AccelZ / AccelScaleFactor;
    T = (double)Temperature / 340 + 36.53; //temperature formula
    Gx = (double)GyroX / GyroScaleFactor;
    Gy = (double)GyroY / GyroScaleFactor;
    Gz = (double)GyroZ / GyroScaleFactor;
    shock = TP_init();

    Serial.print("Ax: "); Serial.print(Ax);
    Serial.print(" Ay: "); Serial.print(Ay);
    Serial.print(" Az: "); Serial.print(Az);
    Serial.print(" T: "); Serial.print(T);
    Serial.print(" Gx: "); Serial.print(Gx);
    Serial.print(" Gy: "); Serial.print(Gy);
    Serial.print(" Gz: "); Serial.print(Gz);
    Serial.print(" Shock: "); Serial.println(shock);

    if ((Gy >= 10.0||Gy<=-10.0||Gx>=10.0||Gx<=-10 || shock>8000 || Ax>0.25 || Ax<-0.25 ||
Ay>0.25 || Ay<-0.20) && shock>100)
    {
      Serial.println("ANGLE CHANGE IN Z DIRECTION IS SUDDEN!!!!");
      BTserial.println("1");
      x=millis();
      flag_button=true;
      acci_detec=true;
      flag=false;
```

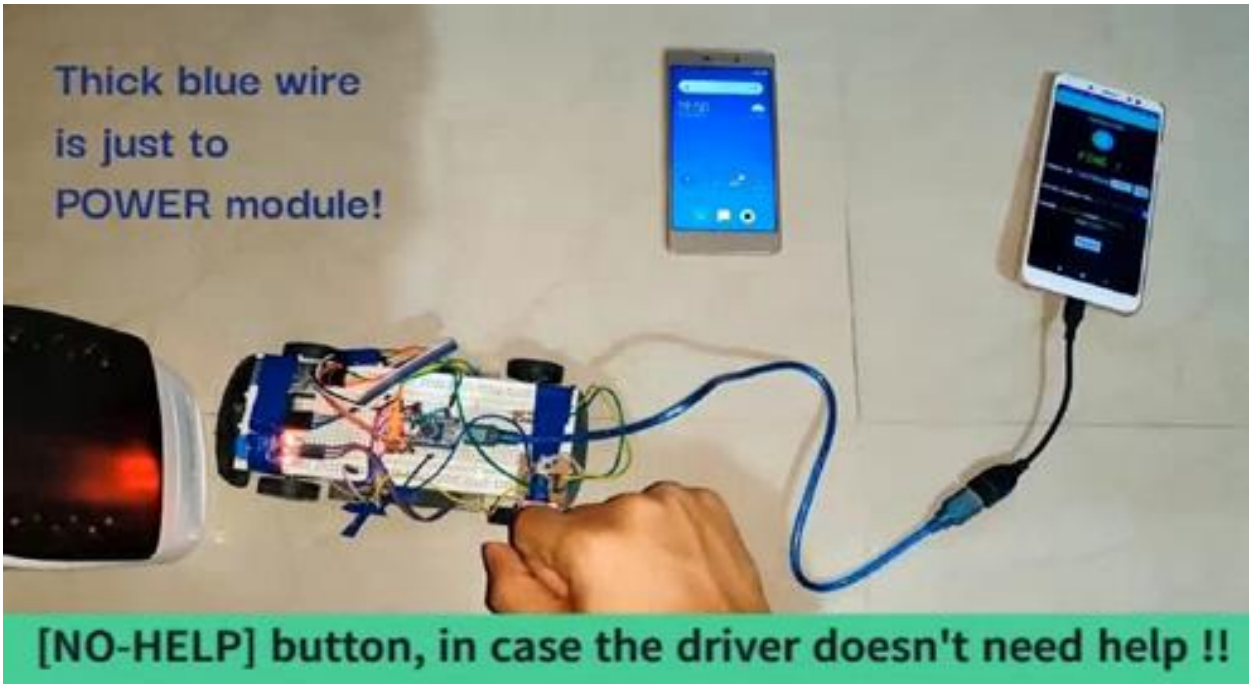
```
    }
  }

  if(acci_detec)
  {
    time=millis()-x;
    if (buttonState == 1 && time <= 15000)
    {
      flag = true;
      acci_detec = false;
      flag_button=false;
      digitalWrite(command, LOW);
      BTserial.println("0");
      buttonState = 0;
    }

    if(time>15000)
    {
      flag=false;
       digitalWrite(command, HIGH);
       Serial.println(F("Accident Ho gaya !!!"));
       BTserial.println("11");
       exit(0);
    }
  }
  delay(100);
}

void I2C_Write(uint8_t deviceAddress, uint8_t regAddress, uint8_t data) {
  Wire.beginTransmission(deviceAddress);
  Wire.write(regAddress);
  Wire.write(data);
  Wire.endTransmission();
}

// read all 14 register
void Read_RawValue(uint8_t deviceAddress, uint8_t regAddress) {
  Wire.beginTransmission(deviceAddress);
  Wire.write(regAddress);
  Wire.endTransmission();
  Wire.requestFrom(deviceAddress, (uint8_t)14);
  AccelX = (((int16_t)Wire.read() << 8) | Wire.read());
  AccelY = (((int16_t)Wire.read() << 8) | Wire.read());
  AccelZ = (((int16_t)Wire.read() << 8) | Wire.read());
  Temperature = (((int16_t)Wire.read() << 8) | Wire.read());
  GyroX = (((int16_t)Wire.read() << 8) | Wire.read());
  GyroY = (((int16_t)Wire.read() << 8) | Wire.read());
  GyroZ = (((int16_t)Wire.read() << 8) | Wire.read());
}

//configure MPU6050
void MPU6050_Init() {
  delay(150);
  I2C_Write(MPU6050SlaveAddress, MPU6050_REGISTER_SMPLRT_DIV, 0x07);
  I2C_Write(MPU6050SlaveAddress, MPU6050_REGISTER_PWR_MGMT_1, 0x01);
  I2C_Write(MPU6050SlaveAddress, MPU6050_REGISTER_PWR_MGMT_2, 0x00);
  I2C_Write(MPU6050SlaveAddress, MPU6050_REGISTER_CONFIG, 0x00);
  I2C_Write(MPU6050SlaveAddress, MPU6050_REGISTER_GYRO_CONFIG, 0x00);//set +/-250 degree/second
full scale
  I2C_Write(MPU6050SlaveAddress, MPU6050_REGISTER_ACCEL_CONFIG, 0x00);// set +/- 2g full scale
```

```
  I2C_Write(MPU6050SlaveAddress, MPU6050_REGISTER_FIFO_EN, 0x00);
  I2C_Write(MPU6050SlaveAddress, MPU6050_REGISTER_INT_ENABLE, 0x01);
  I2C_Write(MPU6050SlaveAddress, MPU6050_REGISTER_SIGNAL_PATH_RESET, 0x00);
  I2C_Write(MPU6050SlaveAddress, MPU6050_REGISTER_USER_CTRL, 0x00);
}
```

5.2 Results

When an accident takes place, the MPU6050 Accelerometer sensor detects the sudden drop in velocity of the vehicle and this when is confirmed by the sudden spike of vibration by the SW-420 Vibration Sensor, the module detects an accident.

On detection of the accident MPU6050 sensor, registers accurate coordinates of the location of the accident and generates a message with latitude and longitudes of the location of the incident. This is then integrated with the google maps api to generate a navigation link from any receivers current location to help anyone navigate to the location of the incident.



After the detection of the accident, the driver is given a 10 second buffer to trigger the No-Help-Required switch, which will disengage the SOS alarm on the module making the trigger a status 0, which means that there is no need for Second-Aid.

If the driver fails to trigger the No-Help-Required switch, then the module identifies the accident as status 1, meaning the intensity of the accident is high and there is a need for Second-Aid.

This is when the bluetooth module will connect with the senders phone to send the sms to ambulance services, hospitals, police station, insurance agency, and loved ones of the person in accident informing them about the incident and notifying them to be ready to provide the fastest and best possible Second-Aid.

This is how we aim to reduce the time delay in provision of Aid to a person in an accident to nearly zero percent.

This is the video of the result of our project. To view the video please double click on the video and then hit play. You can also view the video from this link. Please view using G-suit id.

Link: IOT MINI PROJECT OUTPUT VIDEO

Chapter VI

SUMMARY

6.1. Conclusion

On an severe accident like an accident, this IOT - Project - AcciLert aims on solving the problem of time delay that occurs when the ambulance is informed about the accident and when it reaches the location of the incident to provide first aid and then when it reaches its way to the hospital to provide second aid to nearly zero percent by making the process of notification to in responsible authorities like Ambulance service, Police station and Hospital's Emergency Department completely automatic, hence saving lives and offering better treatment to the ones in need.

On an occurance of an accident, our module is designed to be triggered by the response of the sensors used, that is by the exponential speed drop in the accelerometer/MPU6050 Accelerometer sensor and the spike of vibration in SW-420 Vibration Sensor which confirms that there is a serious accident at the location tracked by the MPU6050 sensor, hence the accident is detected. After detection of the accident the driver is given a 10 second buffer to hit the No-Help-Required switch to turn off the SOS alarm and the system will register the incident as status 0, in which case there is no need for Second-Aid. If the driver fails to hit the No-Help-Required switch in the buffer time then the server will register the accident as status 1, in which case it will make notifications to ambulance service, nearby hospital, police station, insurance agency and the loved ones of the driver so that necessary actions can take place without any time delay in providing best possible aid to the patient in need.

This is how we aim to solve the problem of delay in time, by making the notification process to responsible authorities completely automatic, hence providing a solution to the above mentioned problem statement.

6.2. References

1. IoT Based Intelligent System For Vehicle Accident and Detection At Real Time:

https://ieeexplore.ieee.org/document/9032662

2. Automobile Black Box System for Accident Analysis

https://ieeexplore.ieee.org/document/7002430

3. Airbag ECU coupled Vehicle Accident SMS Alert System

https://ieeexplore.ieee.org/document/8365258

4. Arduino based vehicle accident alert system using GPS, GSM and accelerometer:

https://circuitdigest.com/microcontroller-projects/arduino-based-accident-alert-system-using-gps-gsm-accelerometer

5. Working of the Accident Detection and Alert System using Arduino:

https://bestengineeringprojects.com/accident-detection-and-alert-system-using-arduino/