END SEMESTER ASSESSMENT (ESA)
B.TECH. (CSE) IV SEMESTER

UE20CS252 - MICROPROCESSOR AND COMPUTER ARCHITECTURE LABORATORY

PROJECT REPORT
ON

CAR SAFETY SYSTEM

SUBMITTED BY:

NAME	SRN
Ashutosh	PES2UG20CS071
Ashutosh Routray	PES2UG20CS072
Claudius D'Souza	PES2UG20CS097

JANUARY-MAY 2022

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING
PES UNIVERSITY, ELECTRONIC CITY CAMPUS,
BENGALURU -560100
KARNATAKA, INDIA

TABLE OF CONTENTS

ABSTRACT OF THE PROJECT

DESCRIPTION

Our project aims to scale down and model a simple car safety 'detection' system using Arduino. The model comprises of **four primary sensors**:

1. Flame Sensor Infrared Receiver Ignition Source Detection Module
2. Water Detection Sensor
3. HC-SR04-Ultrasonic Range Finder
4. MPU-6050 3-Axis Accelerometer and Gyro Sensor

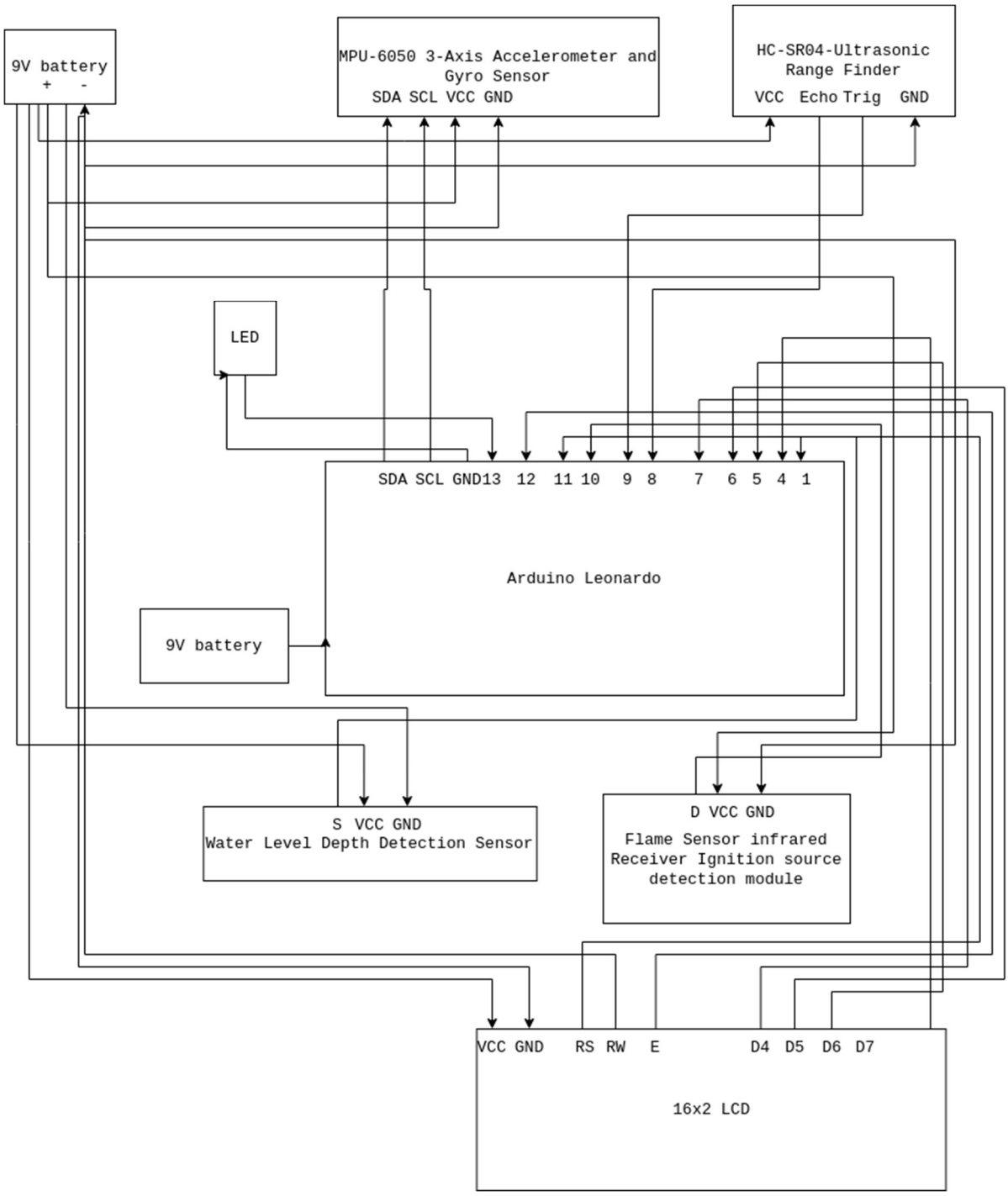
FUNCTIONS OF SENSORS:

- The Flame Sensor Module is used for Fire Detection in a car's internal wiring.
- The Water Detection Sensor is used for detecting the seepage of flood water into the car exhaust.
- The Ultrasonic Range Finder is used as a proximity/collision detector.
- The Gyro Sensor is used to detect the physical stability and balance of a vehicle; used in falls and flip overs, works along with the collision sensor

GENERAL WORKING:

- The sensors and the Arduino Leonardo board are powered by two segregate 9 Volt Batteries – this is to ensure sufficient power supply to all the involved components.
- Each device when triggered according to their sensor, produces a signal that is processed and displayed onto a 16x2 LCD Monitor. This process is mediated by the programmed Arduino Leonardo board.
- The LCD Monitor can serve as a means to know the cause of the accident and the component can be rectified accordingly.
- Additionally, an LED light has been programmed to produce an SOS light; a standard emergency projection in Morse Code to display a call/request for rescue. This is especially used for spotting cars that are underwater or in the dark.
- All the above modules are placed at appropriate positions in the model that applies to a real-life vehicle.

CIRCUIT DIAGRAM



ARDUINO CODE

```
#include <Adafruit_MPU6050.h>
#include <Adafruit_Sensor.h>
#include <Wire.h>
#include <LiquidCrystal.h>
Adafruit_MPU6050 mpu;

#define echoPin 8
#define trigPin 9
#define FlamePin 10
#define Grove_Water_Sensor 11
#define ledPin 13

long duration;
int distance;
int isFlame = HIGH;
int accident=0;

const int rs = 1, en = 12, d4 = 7, d5 = 6, d6 = 5, d7 = 4;
LiquidCrystal lcd(rs, en, d4, d5, d6, d7);

void setup()
{
  Serial.begin(9600);
  lcd.begin(16, 2);
  pinMode(trigPin,OUTPUT);
  pinMode(echoPin,INPUT);
  pinMode(FlamePin, INPUT);
  pinMode(Grove_Water_Sensor, INPUT);
  Serial.begin(115200);
  Serial.println("Starting up...");
  Serial.println("Adafruit MPU6050 test!");

  // Try to initialize!
  if (!mpu.begin()) {
    Serial.println("Failed to find MPU6050 chip");
    while (1) {
      delay(10);
    } }
```

```
Serial.println("MPU6050 Found!");

mpu.setAccelerometerRange(MPU6050_RANGE_8_G);
Serial.print("Accelerometer range set to: ");
switch (mpu.getAccelerometerRange()) {
case MPU6050_RANGE_2_G:
  Serial.println("+-2G");
  break;
case MPU6050_RANGE_4_G:
  Serial.println("+-4G");
  break;
case MPU6050_RANGE_8_G:
  Serial.println("+-8G");
  break;
case MPU6050_RANGE_16_G:
  Serial.println("+-16G");
  break;
}
mpu.setGyroRange(MPU6050_RANGE_500_DEG);
Serial.print("Gyro range set to: ");
switch (mpu.getGyroRange()) {
case MPU6050_RANGE_250_DEG:
  Serial.println("+- 250 deg/s");
  break;
case MPU6050_RANGE_500_DEG:
  Serial.println("+- 500 deg/s");
  break;
case MPU6050_RANGE_1000_DEG:
  Serial.println("+- 1000 deg/s");
  break;
case MPU6050_RANGE_2000_DEG:
  Serial.println("+- 2000 deg/s");
  break;
}

mpu.setFilterBandwidth(MPU6050_BAND_21_HZ);
Serial.print("Filter bandwidth set to: ");
```

```
switch (mpu.getFilterBandwidth()) {
  case MPU6050_BAND_260_HZ:
    Serial.println("260 Hz");
    break;
  case MPU6050_BAND_184_HZ:
    Serial.println("184 Hz");
    break;
  case MPU6050_BAND_94_HZ:
    Serial.println("94 Hz");
    break;
  case MPU6050_BAND_44_HZ:
    Serial.println("44 Hz");
    break;
  case MPU6050_BAND_21_HZ:
    Serial.println("21 Hz");
    break;
  case MPU6050_BAND_10_HZ:
    Serial.println("10 Hz");
    break;
  case MPU6050_BAND_5_HZ:
    Serial.println("5 Hz");
    break;
  }
  Serial.println("");
  delay(100);
}

void flash(int duration){
  digitalWrite(ledPin,HIGH);
  delay(duration);
  digitalWrite(ledPin,LOW);
  delay(duration);
}

void raiseSOS(){
  int sos = 1;
  while (sos --){
    flash(200); flash(200); flash(200); // S
    delay(300); // otherwise the flashes run together
```

```
    flash(500); flash(500); flash(500); // O
    flash(200); flash(200); flash(200); // S
    delay(1000);
  }
}

void SOS(){
  int sos = 5;
  while (sos--){
    flash(200); flash(200); flash(200); // S
    delay(300); // otherwise the flashes run together
    flash(500); flash(500); flash(500); // O
    flash(200); flash(200); flash(200); // S
    delay(1000);
  }
}

void loop() {
//  int accident = 0;
  int fireFlag = 0, gyroFlag = 0, waterFlag = 0, distanceFlag
= 0;
  lcd.setCursor(0, 0);
  sensors_event_t a, g, temp;
  mpu.getEvent(&a, &g, &temp);
  lcd.print("Ready");
  digitalWrite(trigPin,LOW);
  delayMicroseconds(2);
  digitalWrite(trigPin,HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin,LOW);
  duration=pulseIn(echoPin,HIGH);

  distance=(duration*0.034/2);

  if (digitalRead(FlamePin) == LOW){
    accident = 1;
    fireFlag = 1;
  }
```

```
else if (digitalRead(Grove_Water_Sensor) == HIGH) {
    accident = 1;
    waterFlag = 1;
}
else if (g.gyro.x > 1.4 || g.gyro.x <-1.4){
  accident = 1;
  gyroFlag = 1;
}
else if (g.gyro.y > 1.4 || g.gyro.y <-1.4){
  accident = 1;
  gyroFlag = 1;
}


else if (distance < 4){
  accident = 1;
  distanceFlag = 1;
}
else{
  if(accident){
    lcd.clear();
    lcd.setCursor(0, 1);
  Serial.println("All good!");
  lcd.print("All good");
  accident=0;
  }
}
lcd.setCursor(0, 1);
if (accident == 1 && gyroFlag == 1){
  lcd.setCursor(0, 0);
  lcd.clear();
  Serial.println("SOS!");
  Serial.println("Fall detected");
  lcd.print("SOS");
  lcd.setCursor(0, 1);
  lcd.print("Fall detected");
  SOS();

} else if (accident == 1 && distanceFlag == 1) {
```

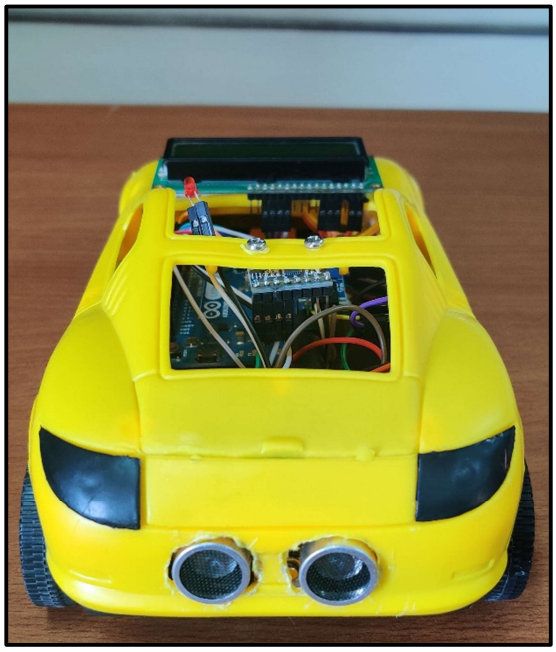
```
    lcd.clear();
    lcd.setCursor(0, 0);
    Serial.println("SOS!");
    Serial.println("Vehicle too close");
    lcd.print("SOS");
    lcd.setCursor(0, 1);
    lcd.print("Too Close");
    SOS();
  } else if (accident == 1 && waterFlag == 1) {
    lcd.setCursor(0, 0);
    lcd.clear();
    Serial.println("SOS!");
    Serial.println("Water detected in the vehicle");
    lcd.print("SOS");
    lcd.setCursor(0, 1);
    lcd.print("Water detected");
    raiseSOS();
  } else if (accident == 1 && fireFlag == 1) {
    lcd.setCursor(0, 0);
    lcd.clear();
    Serial.println("SOS!");
    Serial.println("Fire detected in the vehicle");
    lcd.print("SOS");
    lcd.setCursor(0, 1);
    lcd.print("Fire detected");
    raiseSOS();
  }

  delay(500);
}
```

PHYSICAL MODEL STRUCTURE

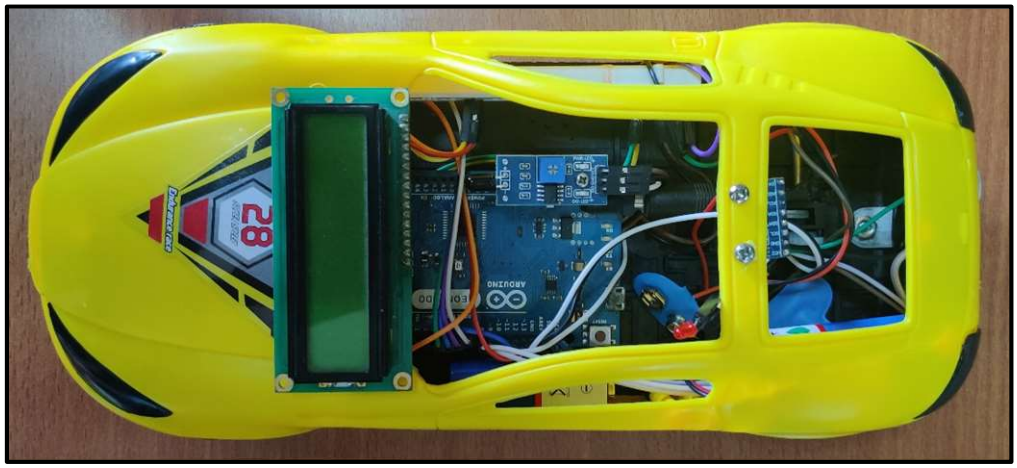
1. RIGHT SIDE

2. LEFT SIDE

3. SEMI-OVERVIEW (FRONT AND BACK)



4. OVERSIDE



5. UNDERSIDE

SCREENSHOTS OF THE OUTPUT

1. **Default State**

2. **Fire Warning**

3. **Flood Warning**

4. **Collision Warning**

5. **Fall/Flip Warning**

REFERENCES

1. http://docs.arduino.cc/
2. https://github.com/adafruit/Adafruit_MPU6050