Fault Detection System using MPU6050

TEAM MEMBERS

JOANNA SARAH S

KARISHMA N

KEERTHANA S

NITHYA PRIYA C

AIM:

To design and implement a fault detection system using the MPU6050 sensor, capable of accurately detecting and analysing real-time orientation, acceleration, and angular velocity for applications in robotics, gesture control, and navigation systems.

COMPONENTS REQUIRED:

- ESP32 Dev board
- MPU6050 Module
- Jumper Wires
- Bread Board
- Power Source
- OLED Display (SSD1306)
- Buzzer

COMPONENTS SPECIFICATION:

ESP32:

- ESP32 has Wi-Fi + Bluetooth, runs on 3.3V, and has a fast processor for real-time tasks.
- It offers 48 GPIO pins, but most dev boards give access to 25–34 usable pins.
- These pins connect sensors, displays, and other hardware for IoT or embedded projects.

MPU6050 Module:

The MPU-6050 is an integrated 6-axis motion sensor combining a 3-axis gyroscope and 3-axis accelerometer, featuring an on-chip Digital Motion Processor (DMP) and 16-bit ADCs for accurate data

OLED Display (SSD1306):

An SSD1306 OLED is a compact, monochrome display with a 128x64 resolution, using an SSD1306 driver IC for self-lit pixels and featuring an I2C or SPI interface for easy connection to microcontrollers.

PIN CONFIGURATION:

ESP32	MPU6050
3.3V	VCC
GND	GND
GPIO21	SDA
GPIO22	SCL

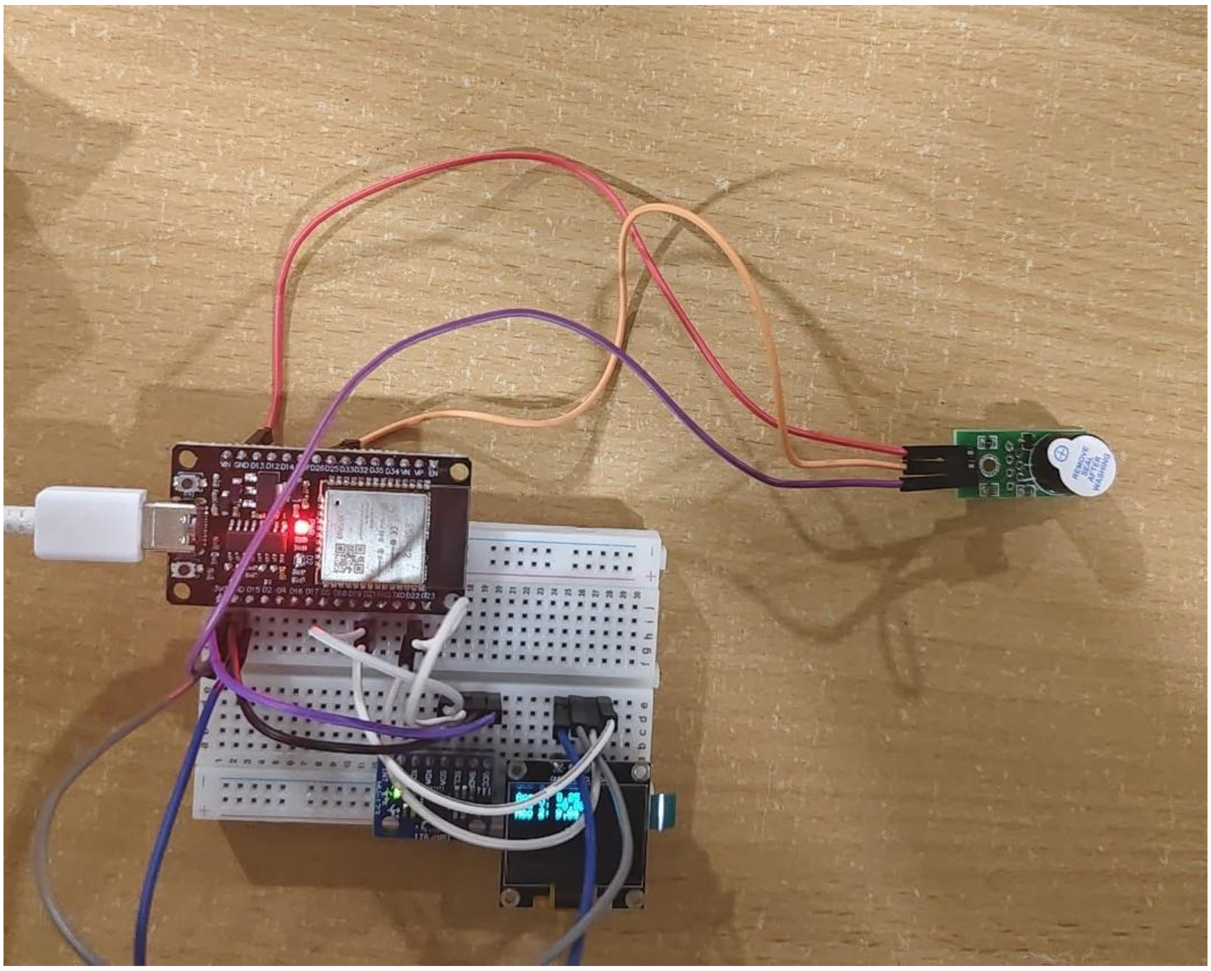
ESP32	**OLED**
3.3V	VCC
GND	GND
GPIO21	SDA
GPIO22	SCL

ESP32	**TMB12A05BUZZER**
GPIO25	+
GND	-

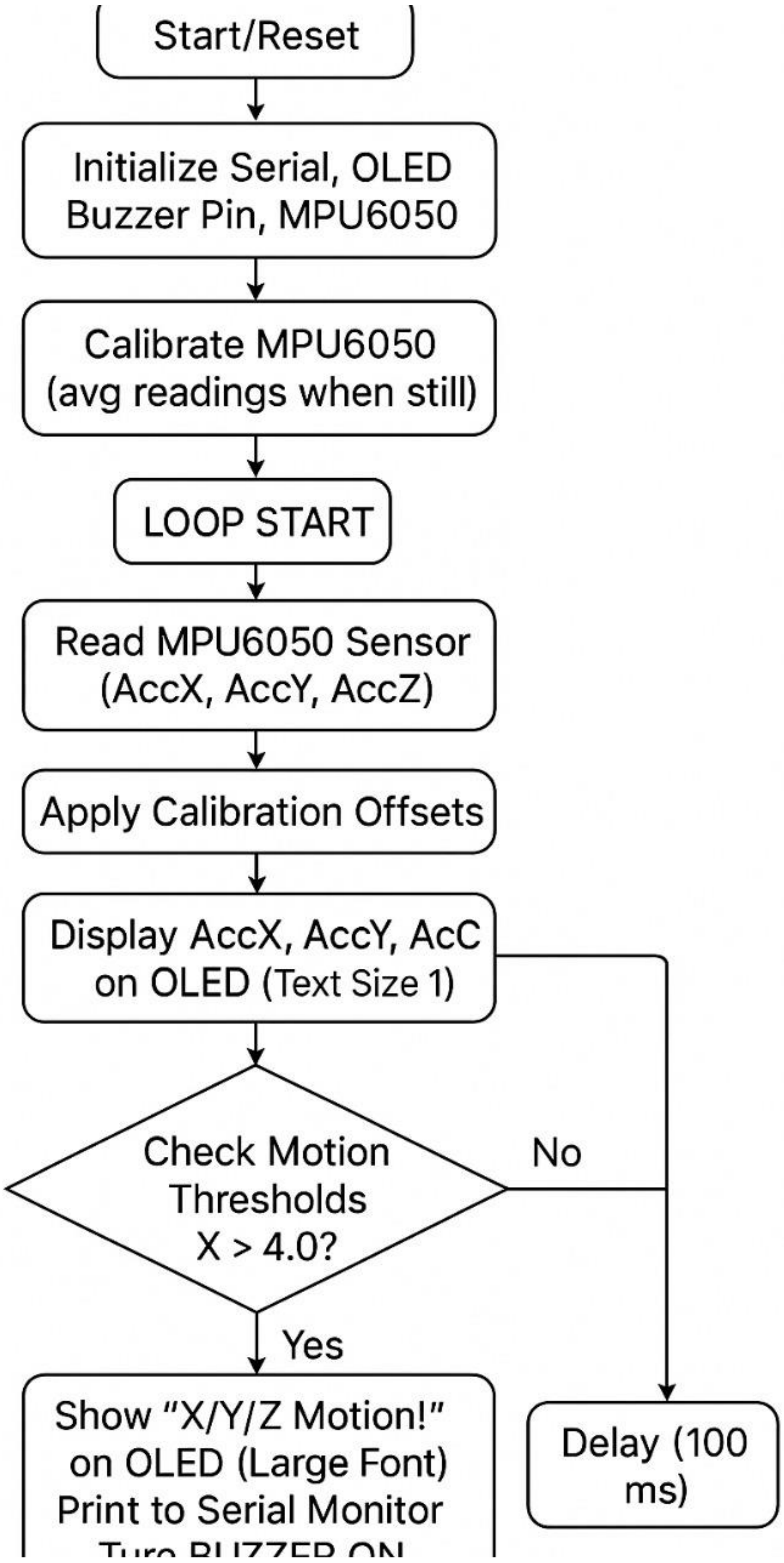
Procedure:

- Connect the ESP32 with the MPU6050 sensor using I^2C pins (SDA, SCL, VCC, GND).
- Connect the OLED display to the same I^2C pins of the ESP32.
- Connect the TMB12A05 buzzer to a GPIO pin of the ESP32 and GND.
- Install Arduino IDE and add ESP32 board support.
- Install the required libraries for MPU6050 and OLED display.
- Write or upload the program to the ESP32 using Arduino IDE.
- Power the ESP32 and observe the sensor values on the OLED display.
- Move the MPU6050 sensor to test motion detection.
- Check that the buzzer turns ON when sudden motion is detected.

CIRCUIT CONNECTION:



FLOWCHAT



Start/Reset
Initialize Serial, OLED Buzzer Pin, MPU6050
Calibrate MPU6050 (avg readings when still)
LOOP START
Read MPU6050 Sensor (AccX, AccY, AccZ)
Apply Calibration Offsets
Display AccX, AccY, AcC on OLED (Text Size 1)
Check Motion Thresholds X > 4.0?
No
Yes
Show "X/Y/Z Motion!" on OLED (Large Font) Print to Serial Monitor
Delay (100 ms)

PROGRAM:

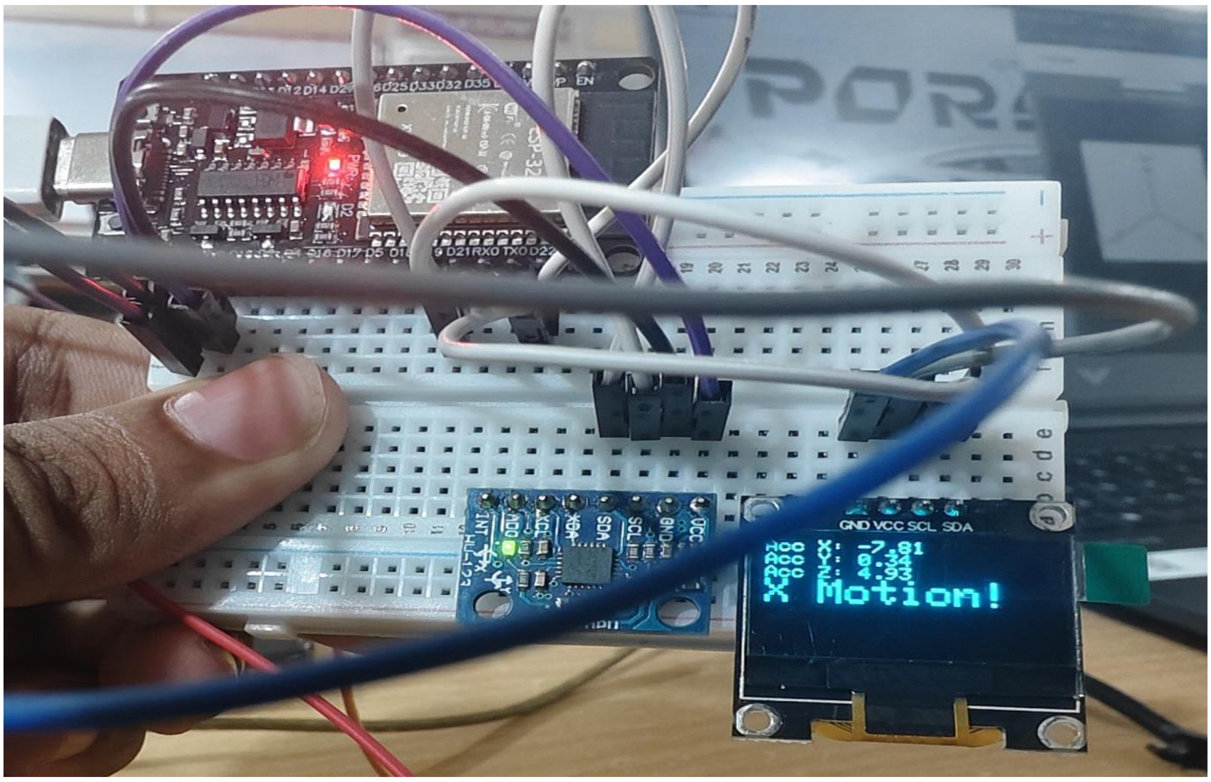
```
#include <Wire.h>

#include <Adafruit_GFX.h>

#include <Adafruit_SSD1306.h>

#include <Adafruit_MPU6050.h>

#define SCREEN_WIDTH 128

#define SCREEN_HEIGHT 64

Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire, -1);

Adafruit_MPU6050 mpu;

#define BUZZER_PIN 25  // Buzzer + to GPIO25, - to GND

float accOffsetX = 0, accOffsetY = 0, accOffsetZ = 0;

void calibrateMPU(int samples = 500) {

 Serial.println("Calibrating... Keep MPU6050 still!");

 accOffsetX = accOffsetY = accOffsetZ = 0;

 for (int i = 0; i < samples; i++) {

  sensors_event_t a, g, temp;

  mpu.getEvent(&a, &g, &temp);

  accOffsetX += a.acceleration.x;

  accOffsetY += a.acceleration.y;

  accOffsetZ += (a.acceleration.z - 9.81); // subtract gravity

  delay(5);

 }

 accOffsetX /= samples;

 accOffsetY /= samples;

 accOffsetZ /= samples;

 Serial.println("Calibration complete!");

 Serial.print("Offsets: ");

 Serial.print(accOffsetX); Serial.print(", ");

 Serial.print(accOffsetY); Serial.print(", ");

 Serial.println(accOffsetZ);
}

void setup() {
 Serial.begin(115200);
 pinMode(BUZZER_PIN, OUTPUT);
 digitalWrite(BUZZER_PIN, LOW);
 if (!display.begin(SSD1306_SWITCHCAPVCC, 0x3C)) {
  Serial.println(F("SSD1306 allocation failed"));
  for (;;);
 }
 display.clearDisplay();
 display.setTextSize(1);
 display.setTextColor(SSD1306_WHITE);
 display.setCursor(0, 0);
 display.println("MPU6050 Init...");
 display.display();
 if (!mpu.begin()) {
  Serial.println("Failed to find MPU6050 chip");
  while (1) delay(10);
 }
 mpu.setAccelerometerRange(MPU6050_RANGE_8_G);
 mpu.setGyroRange(MPU6050_RANGE_500_DEG);
 mpu.setFilterBandwidth(MPU6050_BAND_21_HZ);
 delay(1000);
 calibrateMPU();
 display.clearDisplay();
 display.setCursor(0, 0);
 display.println("Calibration Done!");
 display.display();
 delay(1000);
}
```

```
void loop() {
 sensors_event_t a, g, temp;
 mpu.getEvent(&a, &g, &temp);
 float accX = a.acceleration.x - accOffsetX;
 float accY = a.acceleration.y - accOffsetY;
 float accZ = a.acceleration.z - accOffsetZ;
 display.clearDisplay();
 display.setCursor(0, 0);
 display.setTextSize(1);
 display.print("Acc X: "); display.println(accX, 2);
 display.print("Acc Y: "); display.println(accY, 2);
 display.print("Acc Z: "); display.println(accZ, 2);
 bool motionDetected = false;
 if (fabs(accX) > 4.0) {
  display.setTextSize(2);
  display.println("X Motion!");
  Serial.println("⚠ Motion on X axis!");
  motionDetected = true;
 }
 if (fabs(accY) > 4.0) {
  display.setTextSize(2);
  display.println("Y Motion!");
  Serial.println("⚠ Motion on Y axis!");
  motionDetected = true;
 }
 if (fabs(accZ) > 10.0) {
  display.setTextSize(2);
  display.println("Z Motion!");
  Serial.println("⚠ Motion on Z axis!");
  motionDetected = true;
 }
```

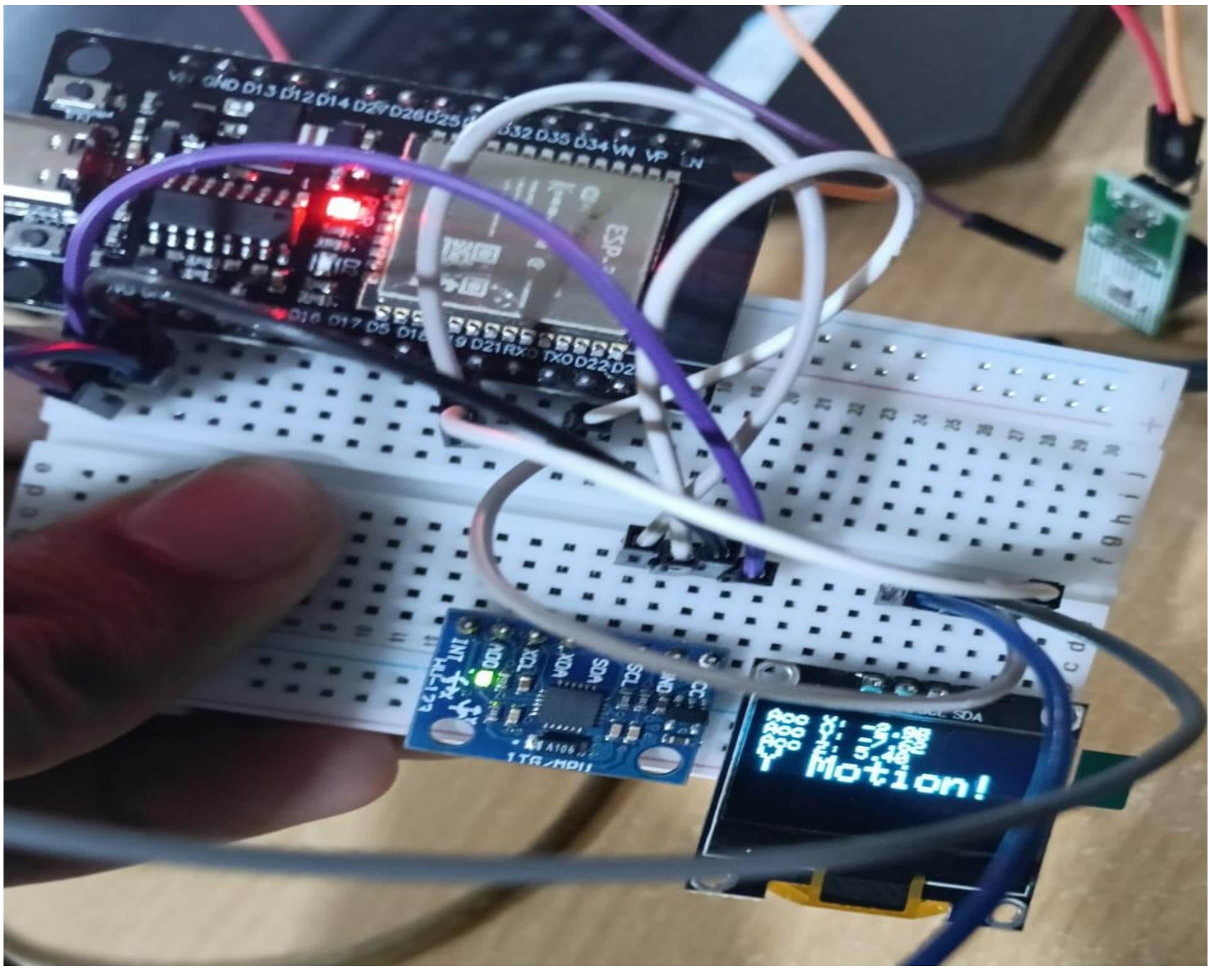
```
  display.display();

  if (motionDetected) {

    digitalWrite(BUZZER_PIN, LOW);

  } else {

    digitalWrite(BUZZER_PIN, HIGH);

  }

  delay(100);

}
```

EXECUTION:

When the ESP32 is powered, the OLED display initializes and the MPU6050 sensor undergoes calibration while kept still. Once calibration is complete, the system continuously monitors the X, Y, and Z acceleration values in real time, which are displayed on the OLED screen. If sudden motion is detected beyond the set thresholds on any axis, the OLED highlights the axis with a clear "Motion Detected" message in larger text, while the buzzer is activated to give an audible alert. When no motion is detected, the buzzer remains silent, ensuring accurate and clear motion detection feedback.



```
Acc X: -7.81
Acc Y: 0.34
Acc Z: 4.93
X Motion!
```



RESULT:

The fault detection system was successfully implemented using the ESP32, MPU6050 sensor, OLED display, and TMB12A05 buzzer. The MPU6050 accurately measured acceleration and angular velocity values, which were displayed in real time on the OLED screen. When sudden motion was detected, the buzzer was activated, providing an audible alert. Thus, the system worked effectively by offering both visual and audio feedback for motion detection.