MOTION AND POSITION TRACKING SYSTEM USING MPU6050 SENSOR

TEAM MEMBERS:

BALAJI N M - 312323106028
HAFEEZUR RAHIMAN - 312323106052
DILLAN ROMAULD VITUS - 312323106044
ASHWIN SUDHAKARAN - 312323106026

AIM

To interface an ESP32 microcontroller with MPU5060 and an OLED display using I2C communication and visualize real-time text message on the screen

Tools/Hardware Required

- ESP32
- MPU-6050
- OLED Display
- Buzzer
- Breadboard
- Jumper wires

THEORY

ESP32 - The ESP32 represents an affordable System on a Chip (SoC) microcontroller which features built-in Wi-Fi and Bluetooth capabilities. Espressif Systems created this powerful microcontroller to serve numerous Internet of Things (IoT) applications. The ESP32 core consists of dual-core 32-bit Xtensa LX6 microprocessors which deliver sufficient processing capability to execute advanced calculations including sensor data fusion.

MPU-6050

The MPU-6050 is a sensor that measures motion. It contains both an accelerometer (to measure acceleration and gravity) and a gyroscope (to measure rotation). The ESP32 reads these measurements to determine the object's movement and position.

OLED Display

The OLED display is a small screen used to show information. It receives data from the ESP32 and displays it in real-time, such as the sensor's readings or the calculated position.

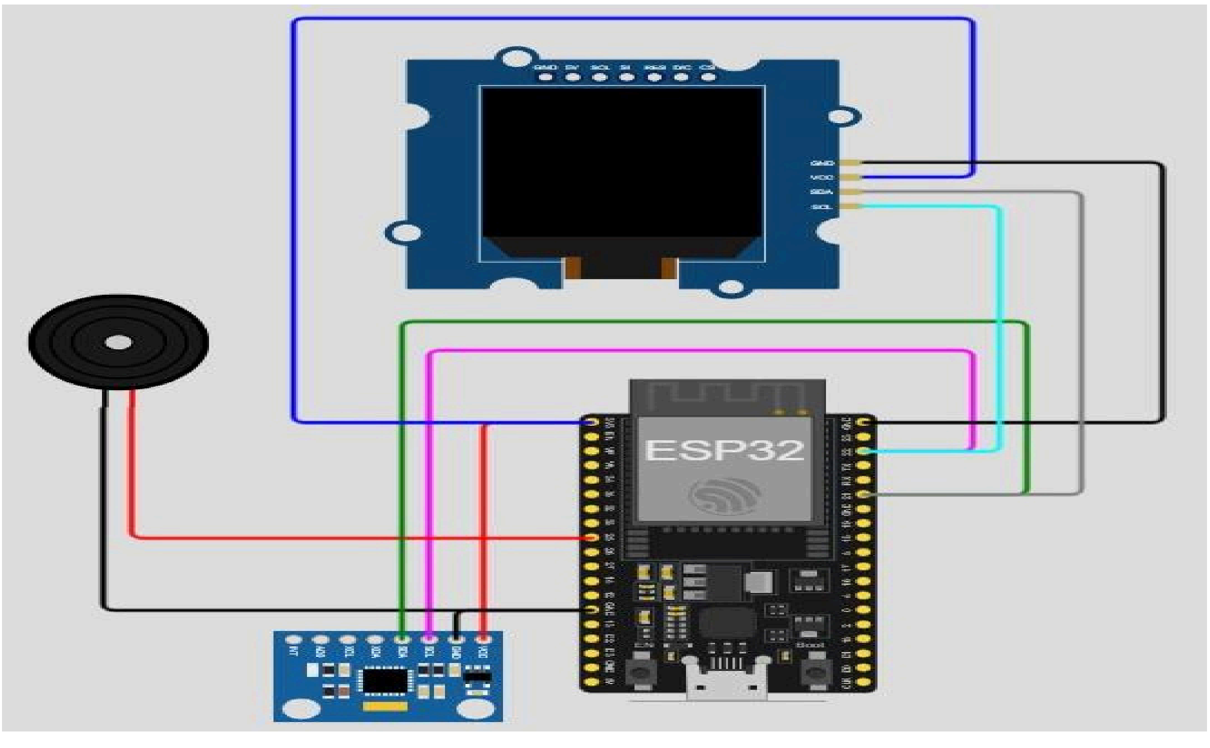
Buzzer

The buzzer is an output device that makes a sound. The ESP32 can be programmed to trigger the buzzer when a specific event occurs, such as a sudden tilt or movement detected by the MPU-6050.

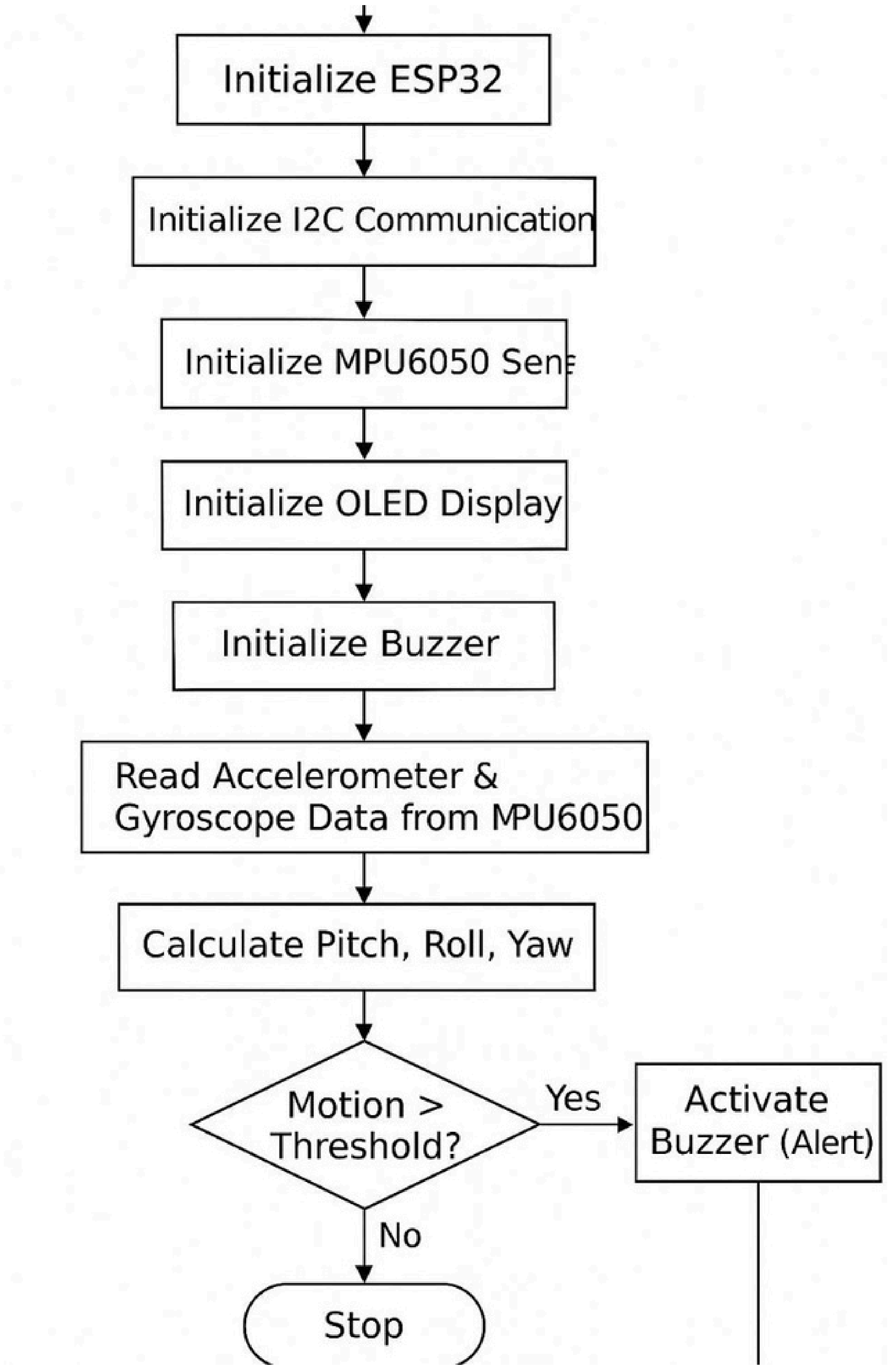
PIN CONNECTIONS

Components	ESP32
MPU-6050 VCC	3.3V
MPU-6050 GND	GND
MPU-6050 SDA	GPI021
MPU-6050 SCL	GPI022
OLED VCC	3.3V
OLED GND	GND
OLED SDA	GPI021
OLED SCL	GPI022
Buzzer +	GPI025
Buzzer -	GND

CIRCUIT DIAGRAM



FLOWCHART

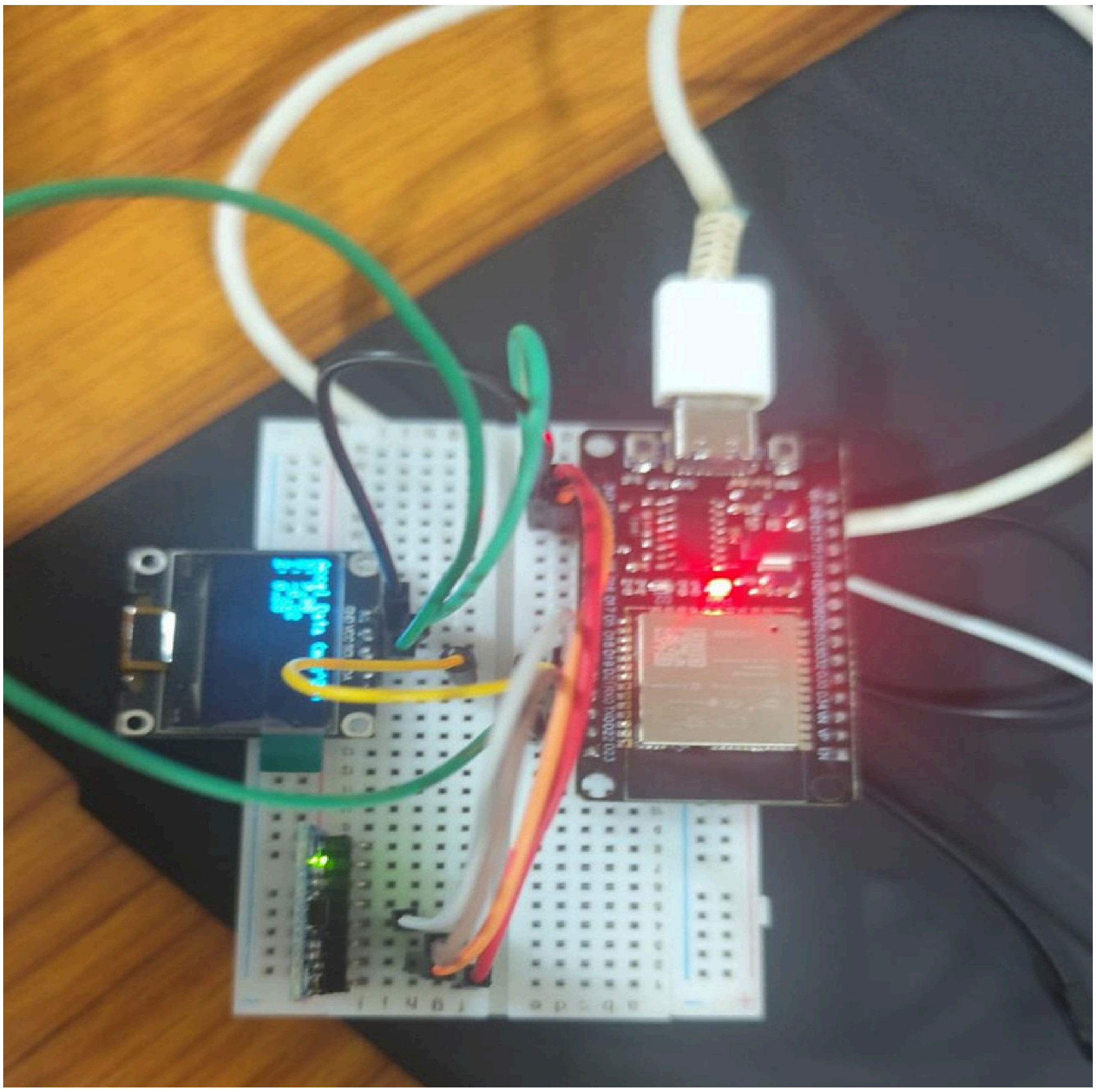

Initialize ESP32
Initialize I2C Communication
Initialize MPU6050 Sen
Initialize OLED Display
Initialize Buzzer
Read Accelerometer & Gyroscope Data from MPU6050
Calculate Pitch, Roll, Yaw
Motion > Threshold?
Yes
Activate Buzzer (Alert)
No
Stop

CODE

```
#define SCREEN_WIDTH 128 #define SCREEN_HEIGHT 64 #define

OLED_RESET -1 Adafruit_SSD1306 display(SCREEN_WIDTH,

SCREEN_HEIGHT, &Wire,
OLED_RESET);



Adafruit_MPU6050 mpu;


#define BUZZER_PIN 5 // ESP32 pin for buzzer

#define MOTION_THRESHOLD 1.5 // adjust sensitivity


// Calibration offsets

float ax_offset = 0, ay_offset = 0, az_offset = 0;


void calibrateMPU() {

 const int samples = 200;

 float ax = 0, ay = 0, az = 0;


 Serial.println("Calibrating MPU6050... keep the sensor still.");


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

 sensors_event_t a, g, temp;

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


  ax  +=  a.acceleration.x;

  ay += a.acceleration.y;
```

```
    az += a.acceleration.z;

    delay(10);
  }

  ax_offset = ax / samples; ay_offset = ay / samples; az_offset =
  (az / samples) - 9.81; // subtract gravity for Z-axis


  Serial.println("Calibration complete!");
  Serial.print("Offsets -> X: "); Serial.print(ax_offset);
  Serial.print(" Y: "); Serial.print(ay_offset);
  Serial.print(" Z: "); Serial.println(az_offset);
}

voidsetup() {
  Serial.begin(115200);
  pinMode(BUZZER_PIN, OUTPUT);

  //OLED init
  if(!display.begin(SSD1306_SWITCHCAPVCC, 0x3C)) {
    Serial.println("SSD1306 allocation failed");
    for(;;);
  }
  display.clearDisplay();
  display.setTextSize(1);
  display.setTextColor(SSD1306_WHITE);

  //MPU init
  if(!mpu.begin()) {
    Serial.println("MPU6050 not found!");
    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();
}

voidloop() {
  sensors_event_t a, g, temp;
  mpu.getEvent(&a, &g, &temp);

  // Apply calibration offsets
  float ax = a.acceleration.x - ax_offset;
  float ay = a.acceleration.y - ay_offset;
  float az = a.acceleration.z - az_offset;

  // Show data on OLED
  display.clearDisplay();
  display.setCursor(0, 0);
  display.printf("X: %.2f", ax);
  display.setCursor(0, 10);
  display.printf("Y: %.2f", ay);
  display.setCursor(0, 20);
  display.printf("Z: %.2f", az);

  //Motion detection
  if(abs(ax) > MOTION_THRESHOLD || abs(ay) > MOTION_THRESHOLD ||
abs(az) < 8) {
    display.setCursor(0, 40);
    display.print("Motion Detected!");
    digitalWrite(BUZZER_PIN, HIGH);
  }else {
    display.setCursor(0, 40);
    display.print("Stable");
    digitalWrite(BUZZER_PIN, LOW);
  }

  display.display();
  delay(200);
}
```

DEMONSTRATION



EXECUTION

1. Hardware Setup:

-Connect MPU6050 sensor to ESP32 via I2C (SDA → GPIO21, SCL → GPIO22).

-Connect SSD1306 OLED display via I2C (SDA → GPIO21, SCL → GPIO22).

-Connect buzzer to GPIO5 (or your chosen pin).

-Provide 3.3V and GND to all components.

2. Install Required Libraries in Arduino IDE:

   - Go to Sketch → Include Library → Manage Libraries.

   - Install the following:

     - Adafruit MPU6050

     - Adafruit Unified Sensor

     - Adafruit GFX Library

     - Adafruit SSD1306

3. Board Selection:

   - In Arduino IDE, go to Tools → Board → ESP32 Arduino → ESP32 Dev Module.

   - Select correct COM port (visible when ESP32 is connected).

4. Upload the Code:

   - Copy and paste the provided calibrated MPU6050 code into Arduino IDE.

   - Verify the code by clicking the checkmark (✓).

   - Upload to ESP32 using the right-arrow button (→).

5. Calibration Process:

   - When ESP32 starts, keep MPU6050 sensor completely still during calibration.

   - Serial Monitor will show offset values.

   - These offsets are automatically applied for stable readings.

6. Execution:

   - OLED will display calibrated accelerometer values (X, Y, Z).

   - If movement is detected above threshold, the buzzer will sound, and OLED will show 'Motion Detected!'.

   - If stable, buzzer will be OFF, and OLED will show 'Stable'.

7. Troubleshooting:

   - If you get COM port not found error → Ensure drivers are installed (CP2102/CH340 depending on ESP32 board).

   - Check Tools → Port and reselect the correct COM.

   - Ensure USB cable supports data transfer (not just charging).