#include <Adafruit\_MPU6050.h>

#include <Adafruit\_Sensor.h>

#include <Wire.h>

// Pin definitions for Ultrasonic Sensor 1

const int trigPin1 = 3;

const int echoPin1 = 2;

// Pin definitions for Ultrasonic Sensor 2

const int trigPin2 = 5;

const int echoPin2 = 4;

// Buzzer Pin

const int buzzerPin = 6;

// Variables for ultrasonic distance sensing

long duration1, duration2;

int distance1, distance2;

const float speedOfSound = 0.0343; // Speed of sound in cm/us

// Distance thresholds

const int closeDistance = 30; // Less than or equal to 30 cm

const int farDistance = 200; // Between 31 and 200 cm

// MPU6050 setup for fall detection

Adafruit\_MPU6050 mpu;

const float fallThreshold = 15.0; // Fall detection threshold in m/s²

void setup(void) {

// Start serial communication

Serial.begin(115200);

// Ultrasonic sensor setup

pinMode(trigPin1, OUTPUT);

pinMode(echoPin1, INPUT);

pinMode(trigPin2, OUTPUT);

pinMode(echoPin2, INPUT);

pinMode(buzzerPin, OUTPUT);

// Initialize the MPU6050 sensor

if (!mpu.begin()) {

Serial.println("Failed to find MPU6050 chip");

while (1) {

delay(10);

}

}

// Set accelerometer and gyroscope settings for the MPU6050

mpu.setAccelerometerRange(MPU6050\_RANGE\_8\_G);

mpu.setGyroRange(MPU6050\_RANGE\_500\_DEG);

mpu.setFilterBandwidth(MPU6050\_BAND\_21\_HZ);

delay(100);

}

void loop() {

// Ultrasonic Sensor 1 distance measurement

digitalWrite(trigPin1, LOW);

delayMicroseconds(2);

digitalWrite(trigPin1, HIGH);

delayMicroseconds(10);

digitalWrite(trigPin1, LOW);

duration1 = pulseIn(echoPin1, HIGH);

distance1 = (duration1 \* speedOfSound) / 2; // Calculate distance in cm

// Ultrasonic Sensor 2 distance measurement

digitalWrite(trigPin2, LOW);

delayMicroseconds(2);

digitalWrite(trigPin2, HIGH);

delayMicroseconds(10);

digitalWrite(trigPin2, LOW);

duration2 = pulseIn(echoPin2, HIGH);

distance2 = (duration2 \* speedOfSound) / 2; // Calculate distance in cm

// MPU6050 for fall detection

sensors\_event\_t a, g, temp;

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

// Calculate the total acceleration magnitude

float totalAcceleration = sqrt(sq(a.acceleration.x) + sq(a.acceleration.y) + sq(a.acceleration.z));

// Fall detection based on acceleration magnitude

if (totalAcceleration > fallThreshold) {

Serial.println("Fall detected!");

tone(buzzerPin, 1000); // Continuous loud beep when fall is detected

delay(5000); // Buzzer sounds for 5 seconds

noTone(buzzerPin); // Stop buzzer

} else {

// Control buzzer based on the distance from both ultrasonic sensors

if (distance1 <= closeDistance || distance2 <= closeDistance) {

tone(buzzerPin, 300); // Continuous beep with higher frequency

delay(50);

noTone(buzzerPin);

delay(150);

}

else if ((distance1 > closeDistance && distance1 <= farDistance) || (distance2 > closeDistance && distance2 <= farDistance)) {

tone(buzzerPin, 200); // Intermittent beep with lower frequency

delay(100);

noTone(buzzerPin);

delay(400);

}

else {

noTone(buzzerPin); // No beep if outside far distance

}

}

// Print distance readings for debugging

Serial.print("Distance1: ");

Serial.print(distance1);

Serial.println(" cm");

Serial.print("Distance2: ");

Serial.print(distance2);

Serial.println(" cm");

// Print total acceleration for debugging

Serial.print("Total Acceleration: ");

Serial.print(totalAcceleration);

Serial.println(" m/s^2");

delay(100); // Small delay to stabilize readings

}

#include <Wire.h>

#include <MPU6050.h>

MPU6050 mpu;

void setup() {

Wire.begin(); // Initialize I2C communication

mpu.initialize(); // Initialize the MPU6050 sensor

if (mpu.testConnection()) {

Serial.println("MPU6050 connection successful");

} else {

Serial.println("MPU6050 connection failed");

while (1); // Stop if MPU6050 fails

}

}

void loop() {

int16\_t ax, ay, az, gx, gy, gz;

mpu.getMotion6(&ax, &ay, &az, &gx, &gy, &gz);

Serial.print("ax: "); Serial.print(ax); Serial.print(" ");

Serial.print("ay: "); Serial.print(ay); Serial.print(" ");

Serial.print("az: "); Serial.print(az); Serial.print(" ");

Serial.print("gx: "); Serial.print(gx); Serial.print(" ");

Serial.print("gy: "); Serial.print(gy); Serial.print(" ");

Serial.print("gz: "); Serial.println(gz);

delay(100);

}