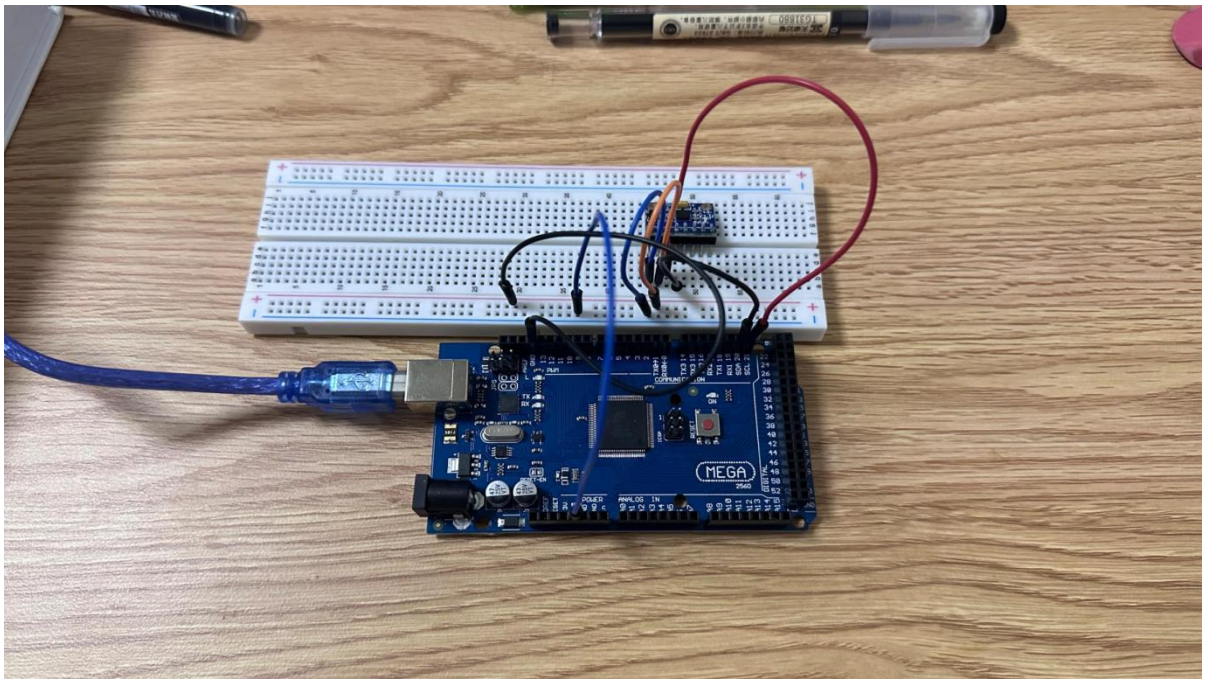
I MPU6050 MODULE

Summary and find:

The MPU6050 is a widely used six-axis motion tracking device with an integrated 3-axis gyroscope and 3-axis accelerometer. It can provide a variety of motion information including acceleration, angular velocity, temperature, etc., and is widely used in flight control, motion detection and other fields. The MPU6050 communicates with the microcontroller through an I2C interface and supports a variety of motion processing algorithms.

After research and discovery, the MPU6050 can be used as a motion tracking device because it can provide acceleration and angular velocity data, which can realize a variety of motion detection and analysis applications

Connect the sensor to Arduino:

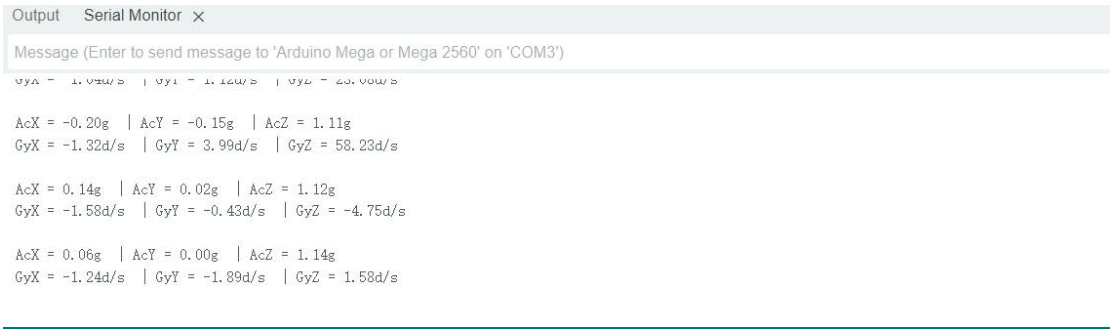


Arduino code

```
#define ACCELE_RANGE 4
#define GYROSC_RANGE 500

#include<Wire.h>
const int MPU_addr = 0x68; // I2C address of the MPU-6050
float AcX, AcY, AcZ, Tmp, GyX, GyY, GyZ;
void setup() {
  Wire.begin();
  Wire.beginTransmission(MPU_addr);
  Wire.write(0x6B);   // PWR_MGMT_1 register
  Wire.write(0);      // set to zero (wakes up the MPU-6050)
  Wire.endTransmission(true);
  Serial.begin(9600);
}
void loop() {
  Wire.beginTransmission(MPU_addr);
  Wire.write(0x3B);   // starting with register 0x3B (ACCEL_XOUT_H)
  Wire.endTransmission(false);
  Wire.requestFrom(MPU_addr, 14, true); // request a total of 14 registers
  AcX = Wire.read() << 8 | Wire.read(); // 0x3B (ACCEL_XOUT_H) & 0x3C (ACCEL_XOUT_L)
  AcY = Wire.read() << 8 | Wire.read(); // 0x3D (ACCEL_YOUT_H) & 0x3E (ACCEL_YOUT_L)
  AcZ = Wire.read() << 8 | Wire.read(); // 0x3F (ACCEL_ZOUT_H) & 0x40 (ACCEL_ZOUT_L)
  Tmp = Wire.read() << 8 | Wire.read(); // 0x41 (TEMP_OUT_H) & 0x42 (TEMP_OUT_L)
  GyX = Wire.read() << 8 | Wire.read(); // 0x43 (GYRO_XOUT_H) & 0x44 (GYRO_XOUT_L)
  GyY = Wire.read() << 8 | Wire.read(); // 0x45 (GYRO_YOUT_H) & 0x46 (GYRO_YOUT_L)
  GyZ = Wire.read() << 8 | Wire.read(); // 0x47 (GYRO_ZOUT_H) & 0x48 (GYRO_ZOUT_L)
  Serial.print(" AcX = "); Serial.print(AcX / 65536 * ACCELE_RANGE-0.01); Serial.print("g ");
  Serial.print(" | AcY = "); Serial.print(AcY / 65536 * ACCELE_RANGE); Serial.print("g ");
  Serial.print(" | AcZ = "); Serial.print(AcZ / 65536 * ACCELE_RANGE+0.02); Serial.println("g ");
  // Serial.print(" | Tmp = "); Serial.println(Tmp/340.00+36.53);   //equation for temperature in degrees
C from datasheet
  Serial.print(" GyX = "); Serial.print(GyX / 65536 * GYROSC_RANGE+1.7); Serial.print("d/s ");
  Serial.print(" | GyY = "); Serial.print(GyY / 65536 * GYROSC_RANGE-1.7); Serial.print("d/s ");
  Serial.print(" | GyZ = "); Serial.print(GyZ / 65536 * GYROSC_RANGE+0.25); Serial.println("d/s
\n");
  delay(500);
}
```

Screenshot of the sensor readings



Accelerometer: (1)Range: ±2g, ±4g, ±8g, ±16g; (2) Sensitivity: varies depending on the range, up to 16384 LSB/g

Gyroscope: (1) Range: ±250°/s, ±500°/s, ±1000°/s, ±2000°/s; (2) Sensitivity: varies depending on the range, up to 131 LSB/°/s

Communication interface: I2C (up to 400kHz)

Power supply voltage: 2.3V-3.4V

Operating temperature range: -40°C to +85°C

Output: Digital output (16-bit ADC conversion value)

The investigation found that it is difficult to accurately measure the acceleration sensitivity of the MPU6050 and have a reference data, because various physical factors need to be calculated. But there is a simple way to use the mobile phone as a reference, as we all know, the acceleration sensor and gyroscope in the mobile phone are more accurate than the MPU6050, so the sensitivity of the MPU6050 can be calculated by comparing with the mobile phone. For example, I use an iPhone and download the app named "phyphox", through which the actual acceleration can be accurately measured. (The green line is the X-axis, the blue line is the Y-axis, and the

yellow line is the Z-axis.)

Sensitivity；

Fix the MPU6050 and the phone together, noting that the x, y, and z axes of the two need to be in the same direction, flat and horizontally fixed to increase accuracy. Accelerate the movement of the system (horizontal or rotational motion), record the data displayed by the Arduino for the corresponding time, and compare it with the actual data displayed on the phone to calculate the sensitivity.

Resolution:

Accuracy:

As mentioned above, the mobile phone and MPU6050 are fixed together, and the measurement value of the sensor is compared with the measurement value of the mobile phone, although there is no guarantee that the value of the mobile phone is the actual value, but the measurement value of the mobile phone is definitely more accurate than the MPU6050. Evaluate the error range of the sensor in the form of percentages.

Drift:

The mobile phone and the MPU6050 are fixed, and the output value of the sensor at different times is recorded while other conditions remain unchanged, so as to obtain drift.