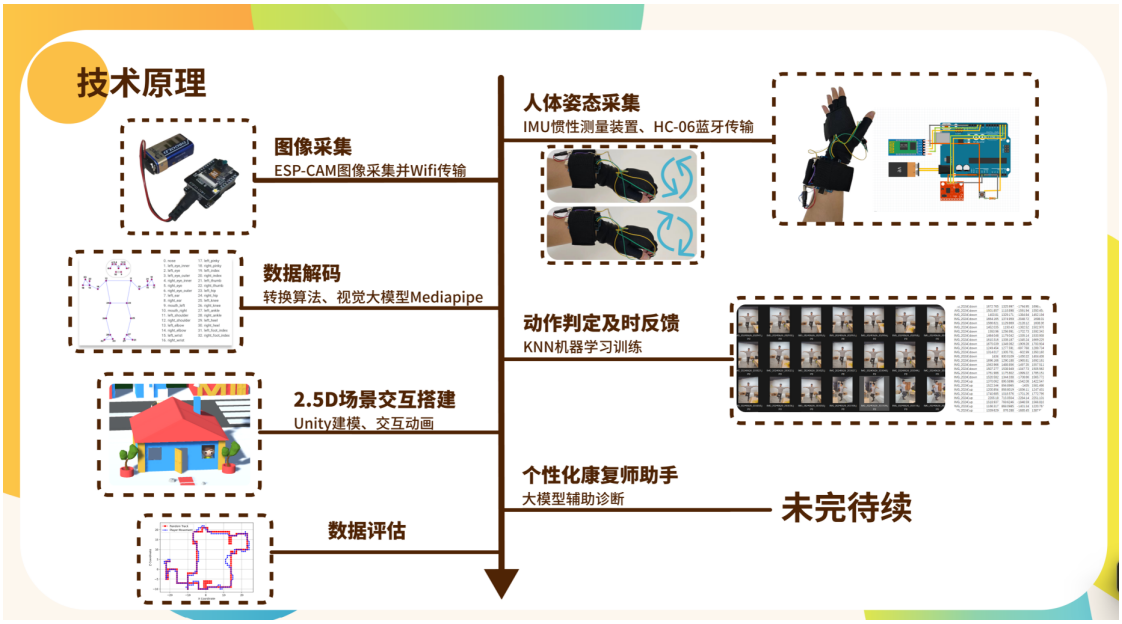
# Inertial Measurement Unit(IMU)

By: Yujie Zang 9/11/2024

## 1 Origins of Personal Journey

When I first heard IMU for the first time in Professor Zhao Liu's ***Open Source and Modeling*** lecture in sophomore year at the Design School of SJTU, I thought it was a cool technology. Therefore, I applied this knowledge to my major assignment, which aimed to support **children with movement challenges**. In our project, we used an **IMU and a camera** to capture the children's motion data, which was then fed into our analysis model. The model's output provided feedback to the children through an interactive game built in Unity, making healthcare more engaging and enjoyable for them.




At the time, we purchased an integrated IMU that included a **built-in microchip, communication protocols, and advanced algorithms.** However, our lack of understanding of the underlying principles and technical details **limited our ability to fully explore creative possibilities for the project**. This experience motivated me to dive deeper into the technical aspects and develop a more comprehensive understanding in my junior year.


# 2 IMU

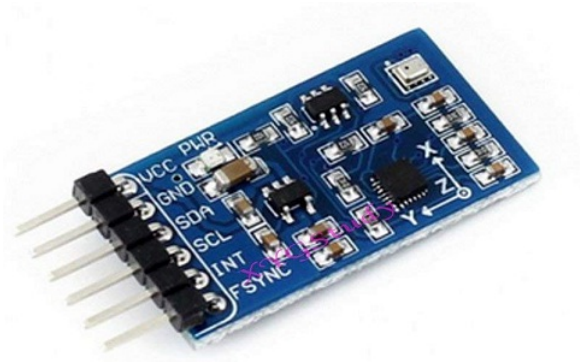
## 2.1What Is an IMU?

The IMU sensor is an electronic device used to calculate and reports an exact force of body, angular rate as well as the direction of the body, which can be achieved by using a blend of 3 sensors like Gyroscope, Magnetometer, and Accelerometer.

**Accelerometer** - measures acceleration in the x, y, and z axes, helping to determine linear movement.

**Gyroscope** - measures angular velocity around each of these axes, detecting rotation.

**Magnetometer** (optional in some IMUs) - measures the magnetic field, which aids in determining orientation relative to Earth's magnetic field (useful for heading and compass functions).




## 2.2 How an IMU works

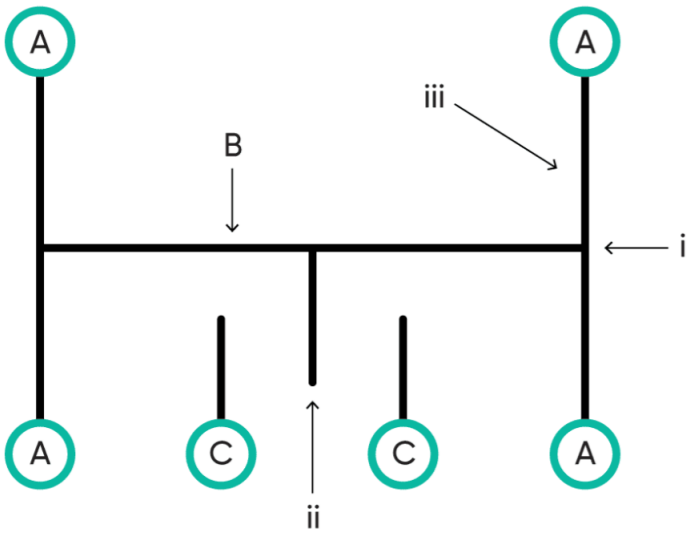
### 2.2.1 Accelerometer

**Accelerometers are motion sensors that measure linear acceleration/rate of change in velocity of an object, relative to a local inertial reference frame(a)**.

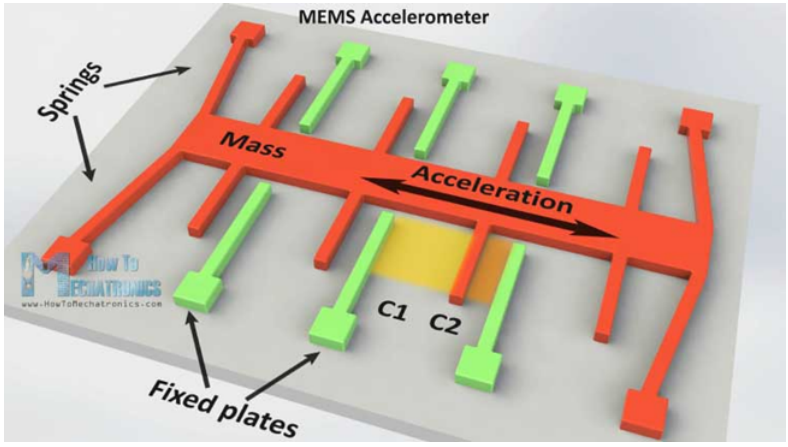
- As a concept, the accelerometer consists of a **proof mass** attached to its reference frame by **mechanical springs.** The springs allow movement of the proof mass along what is known as the sensitivity axis.
- Measurement of the displacement is performed through **electrical capacitance**. The sensor will have several **differential capacitors.** These are electrodes that are fixed in place

either side of the proof mass. The proof mass will have extension arms or “fingers” that protrude into the space between the differential capacitor electrodes. At a standstill, the fingers are centered between the electrodes and a known capacitance is produced to represent zero acceleration. During an acceleration event, the proof mass moves – this is because it is sprung and so will accelerate at a different rate to the rest of the sensor (the reference frame).

The fingers move closer to one electrode,**creating a change in capacitance from which acceleration can be derived.**




*(ii) :fingners C:electrical capacity B:mass  (iii):mechanical springs*

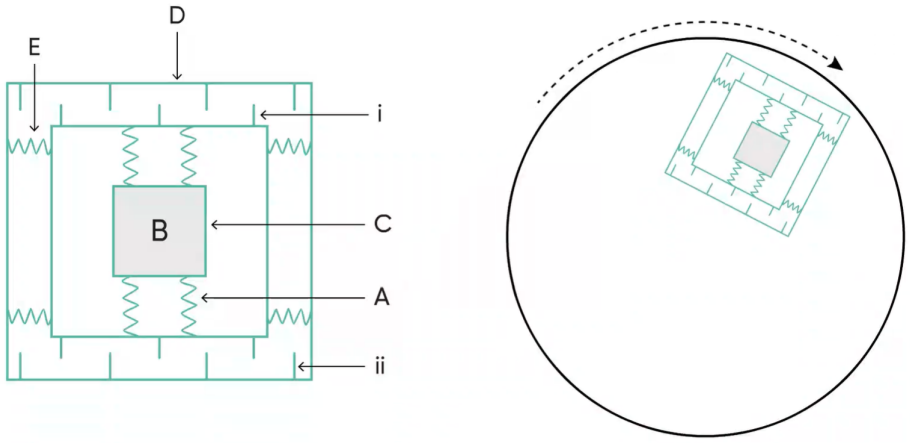



## 2.2.2 Gyroscope

**Angular rate sensors, commonly known as gyroscopes, indicate acceleration (rate of rotation) about an axis**. While multiple techniques for measuring angular acceleration exist, the most common method uses the ***Coriolis effect***.  Gyroscopes detect changes in angular momentum.

**In a typical Coriolis MEMS gyroscope**, a resonating proof mass is attached to its reference frame by mechanical springs. This reference frame is attached to an outer reference frame and isolated by** mechanical springs**. The proof mass is made to vibrate along a particular axis** – this is known as the drive axis. When the gyroscope is rotated, the Coriolis effect induces a secondary vibration along the axis perpendicular to the drive axis – this is known as the sense axis. As with many MEMS accelerometer sensors, the measurement is derived through electrical capacitance. **Rotation causes a change in the output of the differential capacitance between the inner and outer frames of reference. As the rate of rotation increases, so**

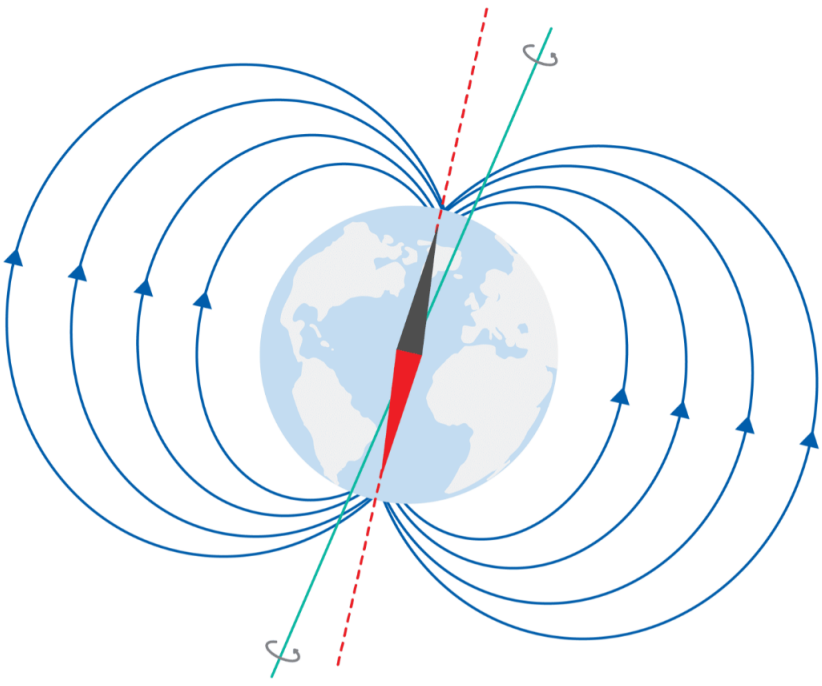
**does the displacement of the proof mass, producing a signal proportional to the Coriolis force / sensed rotation.**




*The springs (**A**) hold the proof mass (**B**) in position within the inner frame of reference (**C**). The inner reference frame is isolated from the outer frame of reference (**D**) also using springs (**E**). The inner frame of reference has several protruding fingers (**i**). The fixed electrodes (**ii**) make up the differential capacitors, with a protruding finger from the inner frame of reference between the capacitor electrodes. The gyroscope may be placed anywhere and at any angle on the rotating object , however, its sense axis must be parallel to the axis of rotation.*

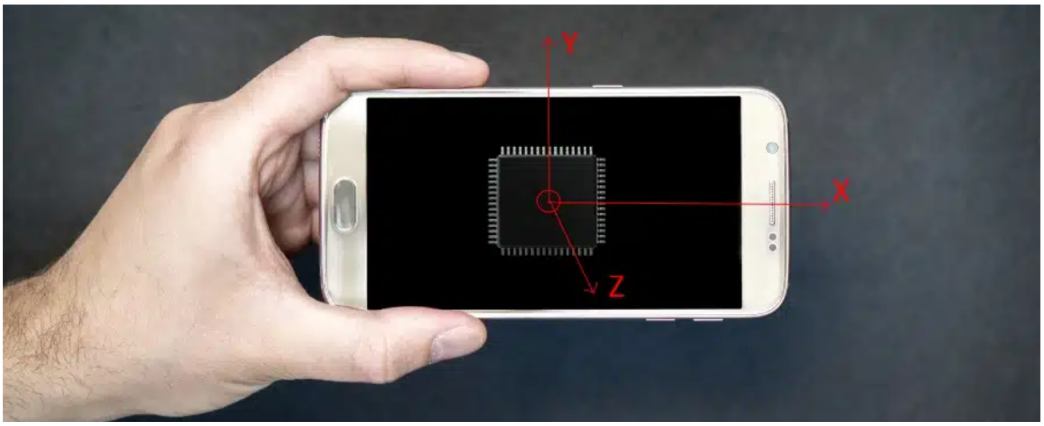
## 2.2.3 Magnetometers

**A magnetometer is used to detect and measure the strength and direction of the Earth's magnetic fields in order to determine heading**. Magnetic heading derives north-related direction using a combination of the Earth's magnetic field strength. Three magnetometers measure the magnetic field strength to provide a three-dimensional orientation, with respect to magnetic north. Note that magnetic north is not the same as "true" (geographical) north, which is the axis that the Earth revolves around.



## 2.3 Applications

- 1. **Aerospace and Aviation:** Helps airplanes and rockets know their position and keep stable in the air.
- 2. **Cars:** Used in self-driving cars to track the car's movement and improve safety features.
- 3. **Phones and Gadgets:** Found in smartphones and tablets to detect if the device is tilted or rotated. Used in fitness trackers to measure your steps and movement.
- 4. **Robots:** Helps robots stay balanced and move smoothly in different directions.
- 5. **Healthcare:** Helps doctors track how people move for physical therapy or rehab. Used in wearables to monitor body movement and posture.
- 6. **Sports:** Analyzes athletes' movements to improve their performance or prevent injuries.
- 7. **Virtual Reality (VR):** Tracks head and body movements in VR games and experiences to make them feel real.




*Application in smart phone*

# 2 Serial communication protocols

As **CAN** is designed for automotive and industrial environments, where noise immunity, reliability, and broadcast communication are essential and **UART** differs by being **asynchronous**, making it suitable for long-distance communication where clock synchronization isn't required.

**I2C** and **SPI** are both **serial communication protocols**, but I2C is often preferred for low-speed, low-power applications and multiple devices on the same bus, while SPI is used for high-speed communication with fewer devices.

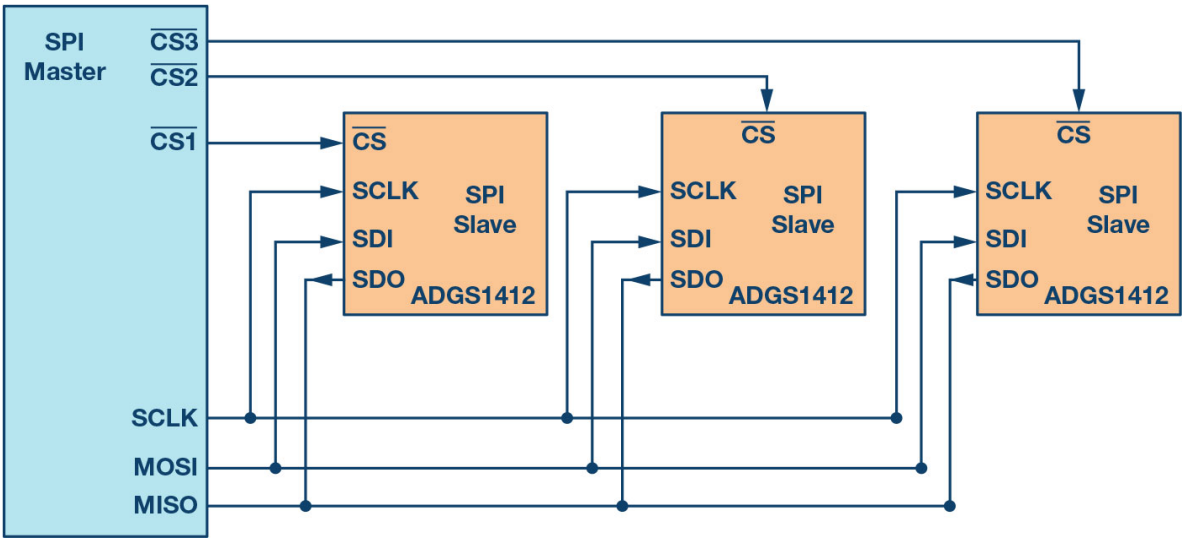
## 2.1 SPI (Serial Peripheral Interface)SPI

**SPI (Serial Peripheral Interface)SPI is a four-wire serial communication protocol used for high-speed data transfer. It is often preferred for applications that require faster communication or need to transfer more data.**

**Key Characteristics of SPI:**

- **Four-Wire Communication**:
  - **MOSI (Master Out Slave In)**: Sends data from the master to the slave.
  - **MISO (Master In Slave Out)**: Sends data from the slave to the master.
  - **SCK (Serial Clock)**: Carries the clock signal.
  - **SS (Slave Select)**: A pin used to select the slave device for communication.
- **Master-Slave**: One device is the master, and it controls the clock and slave device selection.

- **No Addressing**: Unlike I2C, SPI does not use addresses. Instead, each device is connected to a dedicated **Slave Select (SS)** pin on the master.
- **Full-Duplex**: SPI allows simultaneous data transfer in both directions (master to slave and slave to master).
- **Speed**: SPI is much faster than I2C, typically running at speeds of up to 10 Mbps or more, depending on the devices and system.

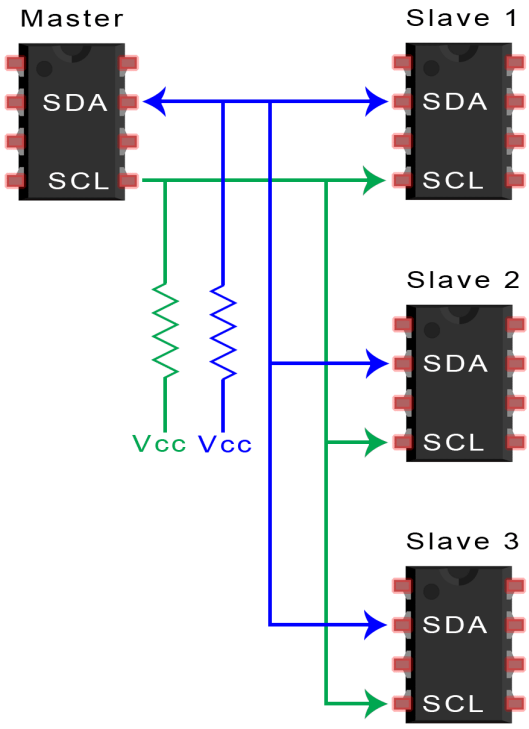



## 2.2 I2C (Inter-Integrated Circuit)

I2C is a two-wire serial communication protocol used to connect low-speed devices like sensors, EEPROMs, or real-time clocks to a microcontroller. It's more commonly used when multiple devices need to be connected to a microcontroller.

**Key Characteristics of I2C:**

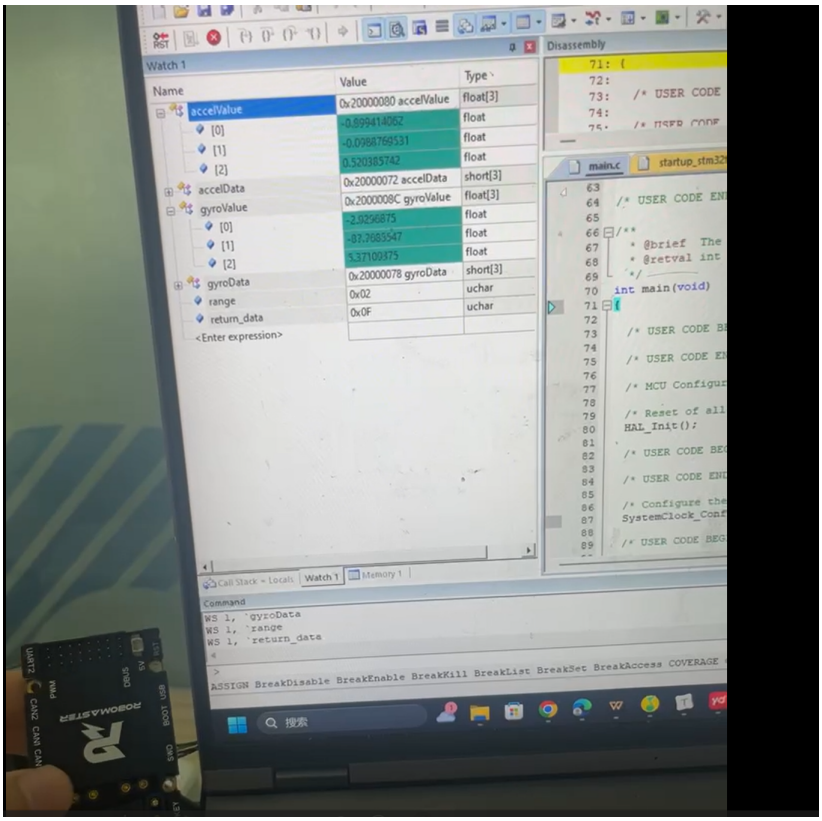
- **Two-Wire Communication**:
  - **SCL (Serial Clock)**: Carries the clock signal.
  - **SDA (Serial Data)**: Carries the data signal.
- **Master-Slave**: One device (master) controls the clock and initiates communication, while the other devices (slaves) respond to requests.
- **Addressing**: Each device has a unique address (7 or 10 bits), and the master uses this address to communicate with the specific device.
- **Multiple Devices**: Multiple devices can share the same bus, and they are addressed by their unique address. However, the total number of devices is limited by the available address space (usually 127 for 7-bit addresses).
- **Speed**: Standard I2C runs at speeds of up to 100 kbps (Standard Mode), but higher speeds of up to 3.4 Mbps are possible with High-Speed Mode.

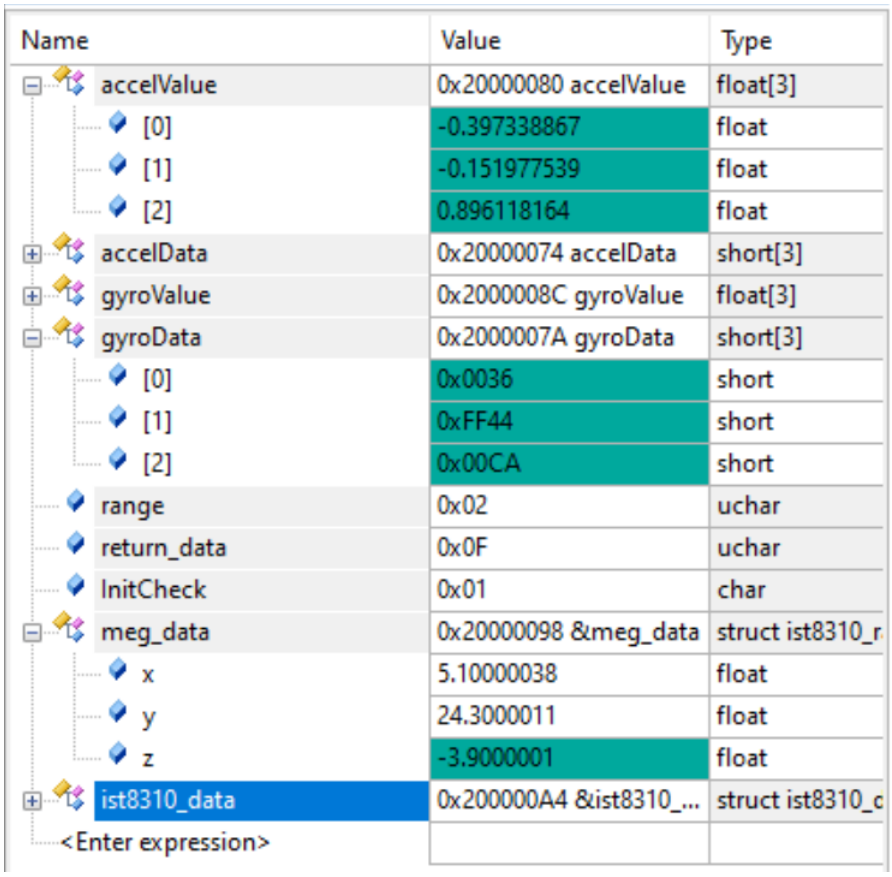



# 3 Practice

## 3.1 Introduction

For my practice, I am using a **DJI-provided C board** equipped with the **IST8310 magnetometer** and the **MPU6500 IMU**. You've successfully implemented code to measure data from both sensors, effectively initializing and reading magnetometer data from the IST8310. The setup allows for sensor data acquisition that could be used in applications requiring **orientation or position tracking.**





| Name | Value | Type |
|---|---|---|
| accelValue | 0x20000080 accelValue | float[3] |
| [0] | -0.397338867 | float |
| [1] | -0.151977539 | float |
| [2] | 0.896118164 | float |
| accelData | 0x20000074 accelData | short[3] |
| gyroValue | 0x2000008C gyroValue | float[3] |
| gyroData | 0x2000007A gyroData | short[3] |
| [0] | 0x0036 | short |
| [1] | 0xFF44 | short |
| [2] | 0x00CA | short |
| range | 0x02 | uchar |
| return_data | 0x0F | uchar |
| InitCheck | 0x01 | char |
| meg_data | 0x20000098 &meg_data | struct ist8310_r |
| x | 5.10000038 | float |
| y | 24.3000011 | float |
| z | -3.9000001 | float |
| ist8310_data | 0x200000A4 &ist8310_... | struct ist8310_c |
| <Enter expression> | | |

## 3.2 Code

MPU6500:

```
// configure
void BMI088_ACCEL_NS_L() {
    HAL_GPIO_WritePin(Acc_GPIO_Port, Acc_Pin, GPIO_PIN_RESET);
}
void BMI088_ACCEL_NS_H() {
    HAL_GPIO_WritePin(Acc_GPIO_Port, Acc_Pin, GPIO_PIN_SET);
}
void BMI088_GYRO_NS_L() {
    HAL_GPIO_WritePin(Gyro_GPIO_Port, Gyro_Pin, GPIO_PIN_RESET);
}
void BMI088_GYRO_NS_H() {
    HAL_GPIO_WritePin(Gyro_GPIO_Port, Gyro_Pin, GPIO_PIN_SET);
}

// read and write reg
void BMI088_ReadReg_ACCEL(uint8_t reg, uint8_t *return_data, uint8_t length) {
    BMI088_ACCEL_NS_L();
    reg = reg | 0x80;
    HAL_SPI_Transmit(&hspi1, &reg, 1, 1000);
    while (HAL_SPI_GetState(&hspi1) == HAL_SPI_STATE_BUSY_TX);
    HAL_SPI_Receive(&hspi1, return_data, 1, 1000);
    while (HAL_SPI_GetState(&hspi1) == HAL_SPI_STATE_BUSY_RX);
    HAL_SPI_Receive(&hspi1, return_data, length, 1000);
    while (HAL_SPI_GetState(&hspi1) == HAL_SPI_STATE_BUSY_RX);
    BMI088_ACCEL_NS_H();
}
void BMI088_ReadReg_GYRO(uint8_t reg, uint8_t *return_data, uint8_t length) {
    BMI088_GYRO_NS_L();
```

```
    reg = reg | 0x80;
    HAL_SPI_Transmit(&hspi1, &reg, 1, 1000);
    while (HAL_SPI_GetState(&hspi1) == HAL_SPI_STATE_BUSY_TX);
    HAL_SPI_Receive(&hspi1, return_data, length, 1000);
    while (HAL_SPI_GetState(&hspi1) == HAL_SPI_STATE_BUSY_RX);
    BMI088_GYRO_NS_H();
}
void BMI088_WriteReg(uint8_t reg, uint8_t write_data) {
    BMI088_ACCEL_NS_L();
    reg = reg | 0x00;
    HAL_SPI_Transmit(&hspi1, &reg, 1, 1000);
    while (HAL_SPI_GetState(&hspi1) == HAL_SPI_STATE_BUSY_TX);
    HAL_SPI_Transmit(&hspi1, &write_data, 1, 1000);
    while (HAL_SPI_GetState(&hspi1) == HAL_SPI_STATE_BUSY_TX);
    BMI088_ACCEL_NS_H();
}

// initiate BMI088
void BMI088_Init() {
    // Soft Reset ACCEL
    BMI088_ACCEL_NS_L();
    BMI088_WriteReg(0x7E, 0xB6); // Write 0xB6 to ACC_SOFTRESET(0x7E)
    HAL_Delay(1);
    BMI088_ACCEL_NS_H();

    // Soft Reset GYRO
    BMI088_GYRO_NS_L();
    BMI088_WriteReg(0x14, 0xB6); // Write 0xB6 to GYRO_SOFTRESET(0x14)
    HAL_Delay(30);
    BMI088_GYRO_NS_H();

    // Switch ACCEL to Normal Mode
    BMI088_ACCEL_NS_L();
    HAL_Delay(1);
    BMI088_WriteReg(0x7D, 0x04); // Write 0x04 to ACC_PWR_CTRL(0x7D)
    HAL_Delay(1);
    BMI088_ACCEL_NS_H();
}

//Task: read reg and return
void BMI088_UserInit(uint8_t *return_data, uint8_t *range)
{
    BMI088_ReadReg_GYRO(0x00, return_data, 1);
    BMI088_WriteReg(0x41, 0x02);
    HAL_Delay(10);
    BMI088_ReadReg_ACCEL(0x41, range, 1);
}

//read and combine data
void BMI088_ReadAccelData(int16_t *accelData, float *accelValue) {
    uint8_t rawData[6];
    BMI088_ReadReg_ACCEL(0x12, rawData, 6);

    accelData[0] = (rawData[1] << 8) | rawData[0];
    accelData[1] = (rawData[3] << 8) | rawData[2];
    accelData[2] = (rawData[5] << 8) | rawData[4];

    uint8_t rangeReg;
```

```
    BMI088_ReadReg_ACCEL(0x41, &rangeReg, 1);
    float scaleFactor = 0.0f;
    switch (rangeReg) {
    case 0x00: scaleFactor = 3.0f / 32768.0f; break;
    case 0x01: scaleFactor = 6.0f / 32768.0f; break;
    case 0x02: scaleFactor = 12.0f / 32768.0f; break;
    case 0x03: scaleFactor = 24.0f / 32768.0f; break;
    }

    accelValue[0] = accelData[0] * scaleFactor;
    accelValue[1] = accelData[1] * scaleFactor;
    accelValue[2] = accelData[2] * scaleFactor;
}
void BMI088_ReadGyroData(int16_t *gyroData, float *gyroValue) {
    uint8_t rawData[6];
    BMI088_ReadReg_GYRO(0x02, rawData, 6);

    gyroData[0] = (rawData[1] << 8) | rawData[0];
    gyroData[1] = (rawData[3] << 8) | rawData[2];
    gyroData[2] = (rawData[5] << 8) | rawData[4];

    float gyroScaleFactor = 2000.0f / 32768.0f * pi / 180; //Convert to radian
system
    gyroValue[0] = gyroData[0] * gyroScaleFactor;
    gyroValue[1] = gyroData[1] * gyroScaleFactor;
    gyroValue[2] = gyroData[2] * gyroScaleFactor;
}
```

**ISP8310**(Provided by peers from:[磁力计数据读取--以IST8310为例 - 知乎 (zhihu.com)](zhihu.com))

```
 void IST8310_INIT(ist8310_data_t* ist8310_data) {
     memset(ist8310_data, 0, sizeof(ist8310_data_t));

     ist8310_data->meg_error = IST8310_NO_ERROR;

     // 把磁力计重启
     HAL_GPIO_WritePin(IST8310_GPIOx, IST8310_GPIOp, GPIO_PIN_RESET);
     HAL_Delay(50);
     HAL_GPIO_WritePin(IST8310_GPIOx, IST8310_GPIOp, GPIO_PIN_SET);
     HAL_Delay(50);

     // 基础配置
     // 不使能中断，直接读取
     WriteSingleDataFromIST8310(IST8310_CNTL2_ADDR, IST8310_STAT2_NONE_ALL);
     // 平均采样四次
     WriteSingleDataFromIST8310(IST8310_AVGCNTL_ADDR, IST8310_AVGCNTL_FOURTH);
     // 200Hz的输出频率
     WriteSingleDataFromIST8310(IST8310_CNTL1_ADDR, IST8310_CNTL1_CONTINUE);

     ist8310_data->meg_error |= VerifyMegId(&ist8310_data->chip_id);
 }

 uint8_t ReadSingleDataFromIST8310(uint8_t addr) {
     uint8_t data;
     HAL_I2C_Mem_Read(&IST8310_I2C, (IST8310_I2C_ADDR << 1), addr,
I2C_MEMADD_SIZE_8BIT, &data, 1, 10);
     return data;
```

```
}

void WriteSingleDataFromIST8310(uint8_t addr, uint8_t data) {
    HAL_I2C_Mem_Write(&IST8310_I2C, (IST8310_I2C_ADDR << 1), addr,
I2C_MEMADD_SIZE_8BIT, &data, 1, 10);
}

void ReadMultiDataFromIST8310(uint8_t addr, uint8_t* data, uint8_t len) {
    HAL_I2C_Mem_Read(&IST8310_I2C, (IST8310_I2C_ADDR << 1), addr,
I2C_MEMADD_SIZE_8BIT, data, len, 10);
}

void WriteMultiDataFromIST8310(uint8_t addr, uint8_t* data, uint8_t len) {
    HAL_I2C_Mem_Write(&IST8310_I2C, (IST8310_I2C_ADDR << 1), addr,
I2C_MEMADD_SIZE_8BIT, data, len, 10);
}

void ReadIST8310Data(ist8310_raw_data_t* meg_data) {
    uint8_t buf[6];
    int16_t temp_ist8310_data = 0;
    ReadMultiDataFromIST8310(IST8310_DATA_XL_ADDR, buf, 6);
    temp_ist8310_data = (int16_t)((buf[1] << 8) | buf[0]);
    meg_data->x = MAG_SEN * temp_ist8310_data;
    temp_ist8310_data = (int16_t)((buf[3] << 8) | buf[2]);
    meg_data->y = MAG_SEN * temp_ist8310_data;
    temp_ist8310_data = (int16_t)((buf[5] << 8) | buf[4]);
    meg_data->z = MAG_SEN * temp_ist8310_data;
}

ist8310_error_e VerifyMegId(uint8_t* id) {
    *id = ReadSingleDataFromIST8310(IST8310_CHIP_ID_ADDR);
    if (*id != IST8310_CHIP_ID_VAL) {
        return MEG_ID_ERROR;
    } else {
        return IST8310_NO_ERROR;
    }
}
```

# 4 Reference


0. SJTU, DJI and RoboMaster
1. 什么是惯性传感器（IMU） - 知乎 (zhihu.com)
2. Inertial Measurement Unit (IMU) – An Introduction | Advanced Navigation
3. Inertial Measurement Unit (IMU) Explained | Built In
4. IMU Sensors: Everything You Need To Know! (embeddedinventor.com)
5. IMU Sensor : Working Principle, Usage and Its Applications (elprocus.com)
6. 磁力计数据读取--以IST8310为例 - 知乎 (zhihu.com)