Invensense MPU-9250
I²C

I²C stands for Inter-Integrated Circuit

is a serial protocol for two-wire interface to connect devices like microcontrollers and other similar peripherals in embedded systems

It is used by almost all major IC manufacturers

I²C bus is popular because it is simple to use (only need 2 signal lines)
I²C

4 wires in total

VCC and GND

SDA (serial data) and SCL (serial clock)

Primary-standby (Master-Slave) architecture (bi-directional)
How does the ESP32 know which peripheral devices to talk/listen to?

Each of the peripheral device has a 7-bit address in the datasheet – it’s the “name” of the device.

The address is written in the datasheet:
https://invensense.tdk.com/wp-content/uploads/2
I²C

How does the ESP32 know which slave devices to talk/listen to?

Can we use any pins on Arduino to connect to SDA and SCL?

GPIO 21 -> SDA; GPIO 22 -> SCL
I²C

4 wires in total signal lines

VCC and GND

SDA (serial data) and SCL (serial clock)

Address: 0b1101000 (0x68)

A4 -SDA
A5 -SCL
How to read data from the IC?
I²C library in Arduino  - Wire library

Setup

Reading a register

Updating a register
**I²C library in Arduino** - **Wire library**

```cpp
#include <Wire.h>
```

### Setup

#### Reading a register

What are register(s)?
- Consider it as a **specific multi-functional storage space** of an IC

#### Updating a register

- We can **read** sensor data from certain register(s)
- We can also **write** a **specific data to one register** to change the sensor’s behavior
I²C library in Arduino - Wire library

Setup

```cpp
#include <Wire.h>
const int sda = 21;
const int scl = 22;

void setup() {
    Wire.begin(sda, scl); // SDA, SCL
}
```

Updating a register

Reading a register
I²C library in Arduino - Wire library

Setup

Reading a register

Updating a register
I²C library in Arduino - Wire library

Setup

Reading a register

Let’s try to read the accelerometer data along X axis from our sensor

Updating a register

First, send a read request
- `Wire.beginTransmission(addr)` opens communication with addr -> 0x68
- `Wire.write(register_name)` register that you are looking for
- `Wire.endTransmission()` sends the request and returns

Then, read the answer to the request
- `Wire.requestFrom(addr, length)` prepares to read length bytes from addr
- `Wire.read()` reads the next available byte

endTransmission()
I²C library in Arduino - Wire library

Setup

https://cdn.sparkfun.com/assets/learn_tutorials/5/5/0/MPU-9250-Register-Map.pdf

Reading a register

First, send a read request
- `Wire.beginTransmission(addr)` opens communication with addr -> 0x68
- `Wire.write(register_name)` register that you are looking for
- `Wire.endTransmission()` sends the request and returns

Then, read the answer to the request
- `Wire.requestFrom(addr, length)` prepares to read length bytes from addr
- `Wire.read()` reads the next available byte

Updating a register

<table>
<thead>
<tr>
<th>Addr (Hex)</th>
<th>Addr (Dec.)</th>
<th>Register Name</th>
<th>Serial</th>
<th>Bit7</th>
<th>Bit6</th>
<th>Bit5</th>
<th>Bit4</th>
<th>Bit3</th>
<th>Bit2</th>
<th>Bit1</th>
<th>Bit0</th>
</tr>
</thead>
<tbody>
<tr>
<td>3B</td>
<td>59</td>
<td>ACCEL_XOUT_H</td>
<td>R</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3C</td>
<td>60</td>
<td>ACCEL_XOUT_L</td>
<td>R</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3D</td>
<td>61</td>
<td>ACCEL_YOUT_H</td>
<td>R</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3E</td>
<td>62</td>
<td>ACCEL_YOUT_L</td>
<td>R</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3F</td>
<td>63</td>
<td>ACCEL_ZOUT_H</td>
<td>R</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>40</td>
<td>64</td>
<td>ACCEL_ZOUT_L</td>
<td>R</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
I²C library in Arduino - Wire library

**Setup**

We first need to read the left 8 bit of accel_x from register 3B

**Reading a register**

First, send a read request
- Wire.beginTransmission(0x68); opens communication with the sensor using its address
- Wire.write(0x3B); tell the sensor which register we are requesting
- Wire.endTransmission(); sends the request and returns

Then, read the answer to the request
- Wire.requestFrom(0x68, 1); prepares to read length byte from the sensor address
- acc_x_h = Wire.read() reads the available byte

**Updating a register**

Address: 0b11010000 (0x68)

<table>
<thead>
<tr>
<th>Addr (Hex)</th>
<th>Addr (Dec.)</th>
<th>Register Name</th>
<th>Serial</th>
<th>Bit7</th>
<th>Bit6</th>
<th>Bit5</th>
<th>Bit4</th>
<th>Bit3</th>
<th>Bit2</th>
<th>Bit1</th>
<th>Bit0</th>
</tr>
</thead>
<tbody>
<tr>
<td>3B</td>
<td>59</td>
<td>ACCEL_XOUT_H</td>
<td>R</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3C</td>
<td>60</td>
<td>ACCEL_XOUT_L</td>
<td>R</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3D</td>
<td>61</td>
<td>ACCEL_YOUT_H</td>
<td>R</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3E</td>
<td>62</td>
<td>ACCEL_YOUT_L</td>
<td>R</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3F</td>
<td>63</td>
<td>ACCEL_ZOUT_H</td>
<td>R</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>40</td>
<td>64</td>
<td>ACCEL_ZOUT_L</td>
<td>R</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
I²C library in Arduino - Wire library

**Setup**

**Reading a register**

First, **send a read request**
- Wire.beginTransmission(0x68); opens communication with the sensor using its address
- Wire.write(0x3C); tell the sensor which register we are requesting
- Wire.endTransmission(); sends the request and returns

Then, **read the answer to the request**
- Wire.requestFrom(0x68, 1); prepares to read length byte from the sensor address
- acc_x_l = Wire.read() reads the available byte

**Updating a register**

To get the entire 16bit of the ACC_X data, we need the ACC_L as well

**Address:** 0b11010000 (0x68)

![Sensor Circuit Diagram]

| Addr (Hex) | Addr (Dec.) | Register Name       | Serial | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|------------|-------------|---------------------|--------|------|------|------|------|------|------|------|------|------|
I²C library in Arduino - Wire library

Setup

Reading a register

Next, we need to combine the data from 3B and 3C

Updating a register

acc_x_combined = acc_x_h << 8 | acc_x_l;

Address: 0b1101000 (0x68)

bitwise SHIFT and OR operation

A = 0b11110101
B = 0b01010101

A <<8

A <<8 | B

0b11110101 00000000
0b11110101 01010101
I^{2}C library in Arduino - Wire library

Setup

Reading a register

Updating a register

Address: 0b1101000 (0x68)

Last step, make the data mean something to us

The result is raw data

Divided raw data by 16384.0 to get meaningful data

\[ gX = \text{acc}_x\_\text{combined} / 16384.0; \]
**I²C library in Arduino - Wire library**

```cpp
byte ACCEL_XOUT_H = 0;
byte ACCEL_XOUT_L = 0;
int16_t ACCEL_X_RAW = 0;
float gX;

void loop() {
    // put your main code here, to run repeatedly:
    Wire.beginTransmission(address);
    Wire.write(0x3B);
    Wire.endTransmission();

    Wire.requestFrom(address, 1);
    ACCEL_XOUT_H = Wire.read();

    Wire.beginTransmission(address);
    Wire.write(0x3C);
    Wire.endTransmission();

    Wire.requestFrom(address, 1);
    ACCEL_XOUT_L = Wire.read();

    ACCEL_X_RAW = ACCEL_XOUT_H << 8 | ACCEL_XOUT_L;
    gX = ACCEL_X_RAW / 16384.0;
    Serial.println(gX);
    delay(10);
}
```

**Setup**

**Reading a register**

**Updating a register**

**Address:** 0b1101000 (0x68)
$I^2C$ library in Arduino  - Wire library

Setup

**Reading** a register

**Updating** a register

Address: 0b1101000  
(0x68)
Assignment:

1. Read all XYZ Accelerometer Data and print them out in a meaningful way
2. Read all XYZ Gyro Data, print the raw value.
3. Move the IMU along each axis with acceleration, observe how the data looks like. You can use Serial plotter for observation.
4. Send us a video link of the experiment.