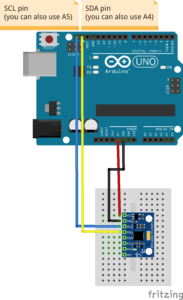
**Wiring layout**

[](data:image/svg+xml,%3Csvg%20xmlns=%22http://www.w3.org/2000/svg%22%20viewBox=%220%200%20183%20300%22%3E%3C/svg%3E)

**Source code**

We make use of the Arduino platform’s in-built library (Wire) to establish an I2C connection between the Arduino Uno and the GY-521 sensor. At the beginning of our source code, the Wire library’s header file is included. Next, we define and declare some variables.

Then, a convert-function is defined. The convert-function makes sure that all sensor values have the same width when they are printed out to the serial monitor later.

In the setup function, a serial connection is established

In the loop function, seven sensor values (3x accelerometer, 1x temperature, and 3x gyro) are requested from the GY-521 module. The MPU-6050 has many registers which can be read. Fourteen of these registers contain the sensor values that we need. As a first step, we tell the GY-521 module where we are going to start reading (“Wire.write(0x3B);”). Then, we request to read 14 registers (“Wire.requestFrom(MPU\_ADDR, 7\*2, true);”). If you are wondering, why 14 registers are read instead of 7 registers, the reason is quite simple: Each sensor value has a size of 2 byte. As each register has a size of one byte, a single sensor value must be retrieved by accessing two registers. The first register contains the so-called “high byte” and the second register contains the “low byte”. Next, all values are retrieved and printed out to the serial connection. At the end of the loop function, a delay of one second is added in order to avoid flooding the serial monitor with messages.

// (c) Michael Schoeffler 2017, http://www.mschoeffler.de

#include "Wire.h" // This library allows you to communicate with I2C devices.

const int MPU\_ADDR = 0x68; // I2C address of the MPU-6050. If AD0 pin is set to HIGH, the I2C address will be 0x69.

int16\_t accelerometer\_x, accelerometer\_y, accelerometer\_z; // variables for accelerometer raw data

int16\_t gyro\_x, gyro\_y, gyro\_z; // variables for gyro raw data

int16\_t temperature; // variables for temperature data

char tmp\_str[7]; // temporary variable used in convert function

char\* convert\_int16\_to\_str(int16\_t i) { // converts int16 to string. Moreover, resulting strings will have the same length in the debug monitor.

sprintf(tmp\_str, "%6d", i);

return tmp\_str;

}

void setup() {

Serial.begin(9600);

Wire.begin();

Wire.beginTransmission(MPU\_ADDR); // Begins a transmission to the I2C slave (GY-521 board)

Wire.write(0x6B); // PWR\_MGMT\_1 register

Wire.write(0); // set to zero (wakes up the MPU-6050)

Wire.endTransmission(true);

}

void loop() {

Wire.beginTransmission(MPU\_ADDR);

Wire.write(0x3B); // starting with register 0x3B (ACCEL\_XOUT\_H) [MPU-6000 and MPU-6050 Register Map and Descriptions Revision 4.2, p.40]

Wire.endTransmission(false); // the parameter indicates that the Arduino will send a restart. As a result, the connection is kept active.

Wire.requestFrom(MPU\_ADDR, 7\*2, true); // request a total of 7\*2=14 registers

// "Wire.read()<<8 | Wire.read();" means two registers are read and stored in the same variable

accelerometer\_x = Wire.read()<<8 | Wire.read(); // reading registers: 0x3B (ACCEL\_XOUT\_H) and 0x3C (ACCEL\_XOUT\_L)

accelerometer\_y = Wire.read()<<8 | Wire.read(); // reading registers: 0x3D (ACCEL\_YOUT\_H) and 0x3E (ACCEL\_YOUT\_L)

accelerometer\_z = Wire.read()<<8 | Wire.read(); // reading registers: 0x3F (ACCEL\_ZOUT\_H) and 0x40 (ACCEL\_ZOUT\_L)

temperature = Wire.read()<<8 | Wire.read(); // reading registers: 0x41 (TEMP\_OUT\_H) and 0x42 (TEMP\_OUT\_L)

gyro\_x = Wire.read()<<8 | Wire.read(); // reading registers: 0x43 (GYRO\_XOUT\_H) and 0x44 (GYRO\_XOUT\_L)

gyro\_y = Wire.read()<<8 | Wire.read(); // reading registers: 0x45 (GYRO\_YOUT\_H) and 0x46 (GYRO\_YOUT\_L)

gyro\_z = Wire.read()<<8 | Wire.read(); // reading registers: 0x47 (GYRO\_ZOUT\_H) and 0x48 (GYRO\_ZOUT\_L)

// print out data

Serial.print("aX = "); Serial.print(convert\_int16\_to\_str(accelerometer\_x));

Serial.print(" | aY = "); Serial.print(convert\_int16\_to\_str(accelerometer\_y));

Serial.print(" | aZ = "); Serial.print(convert\_int16\_to\_str(accelerometer\_z));

// the following equation was taken from the documentation [MPU-6000/MPU-6050 Register Map and Description, p.30]

Serial.print(" | tmp = "); Serial.print(temperature/340.00+36.53);

Serial.print(" | gX = "); Serial.print(convert\_int16\_to\_str(gyro\_x));

Serial.print(" | gY = "); Serial.print(convert\_int16\_to\_str(gyro\_y));

Serial.print(" | gZ = "); Serial.print(convert\_int16\_to\_str(gyro\_z));

Serial.println();

// delay

delay(1000);

}

If the code is compiled and transferred to the Arduino Uno, you should see the sensor values in the serial monitor of the Arduino IDE. Moreover, when the GY-521 board is rotated or moved, the sensor values should change according to the movement.