

MPU-6050

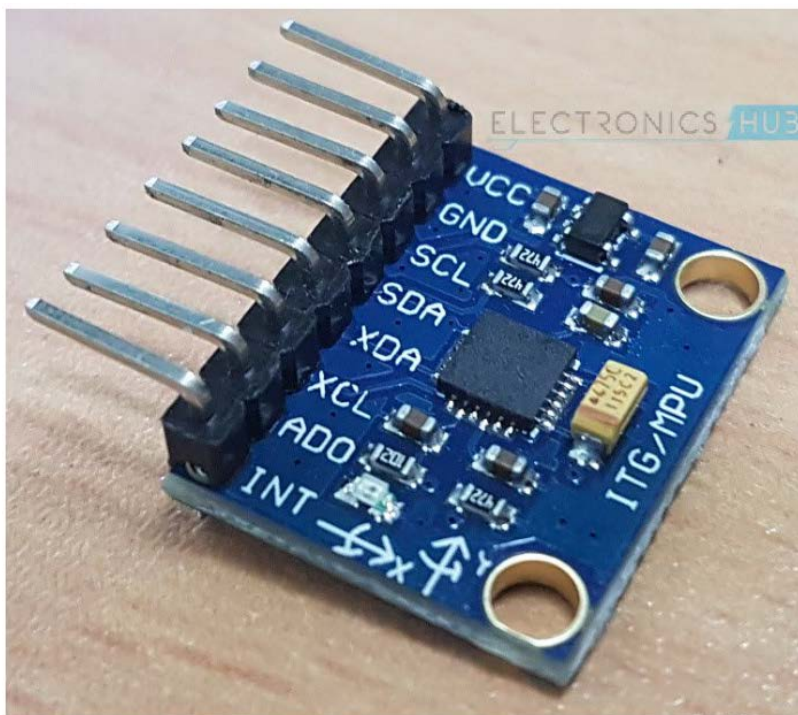
MPU-6050 is an IMU Sensor that contains a MEMS (Microelectromechanical System) Accelerometer and MEMS Gyroscope on a single chip.

Here, IMU Sensor, where IMU stands for Inertial Measurement Unit, is a device that measures the specific force using Accelerometer, angular rate using Gyroscope and magnetic field using Magnetometers.

IMU Sensors are used in self-balancing robots, aircrafts, mobile phones, tablets, spacecraft, satellites, drones, UAVs (unmanned aerial vehicles) etc. for guidance, position detection, orientation detection, motion tracking and flight control.

The two common IMUs are ADXL 335 Accelerometer and MPU-6050. The ADXL 335 contains a 3-axis Accelerometer.

In case of MPU-6050, it is a six-axis motion tracking device that combines a 3-axis Accelerometer and a 3-axis Gyroscope on a single chip. We will see more details about MPU6050 in the next section.



The MPU-6050 is a six-axis motion tracking device developed by InvenSense. The main features of the MPU6050 device are mentioned below.

- Three – axis Accelerometer
- Three – axis Gyroscope

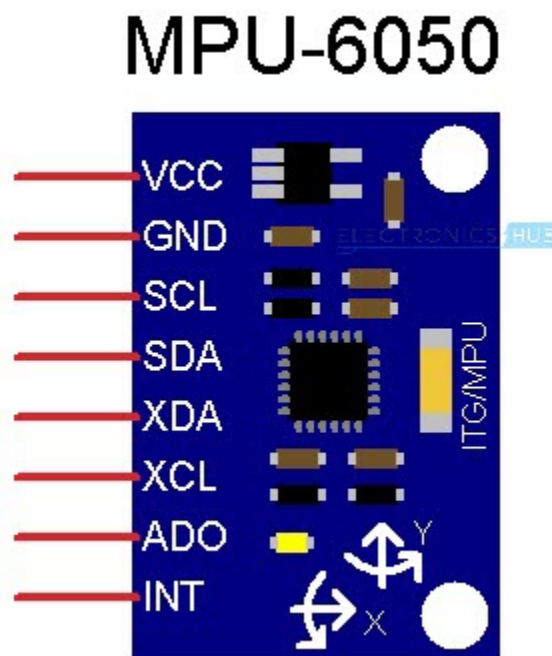
- Digital Output Temperature Sensor
- Six 16-bit ADC (three for Accelerometer and three for Gyro)
- Integrated Digital Motion Processor (DMP)
- 1024B FIFO Buffer

The six-axis MPU-6050 is some time called as a 6 DoF (six Degrees of Freedom) device, as it provides six output values (three from Accelerometer and three from Gyro). The MPU-6050 can communicate using I2C Protocol.

Digital Motion Processor or the DMP is an embedded processor that can reduce the computational load from the host processor, like an Arduino, by acquiring and processing data from Accelerometer, Gyroscope and an external Magnetometer.

Interfacing MPU6050 with Arduino

As mentioned earlier, the MPU6050 supports only I2C Communication and hence, it must be connected only to the I2C Pins of the Arduino. The I2C pins of Arduino are multiplexed with the analog input pins A4 and A5 i.e. A4 is SDA and A5 is SCL.



Coming to the MPU6050, we have used a normal breakout board that provided eight pins. The above image shows the schematic representation of the MPU6050 Breakout board.

In this, we will be using the SCL, SDA and the INT pins to connect with Arduino.

INTERFACING WITH AN ARDUINO

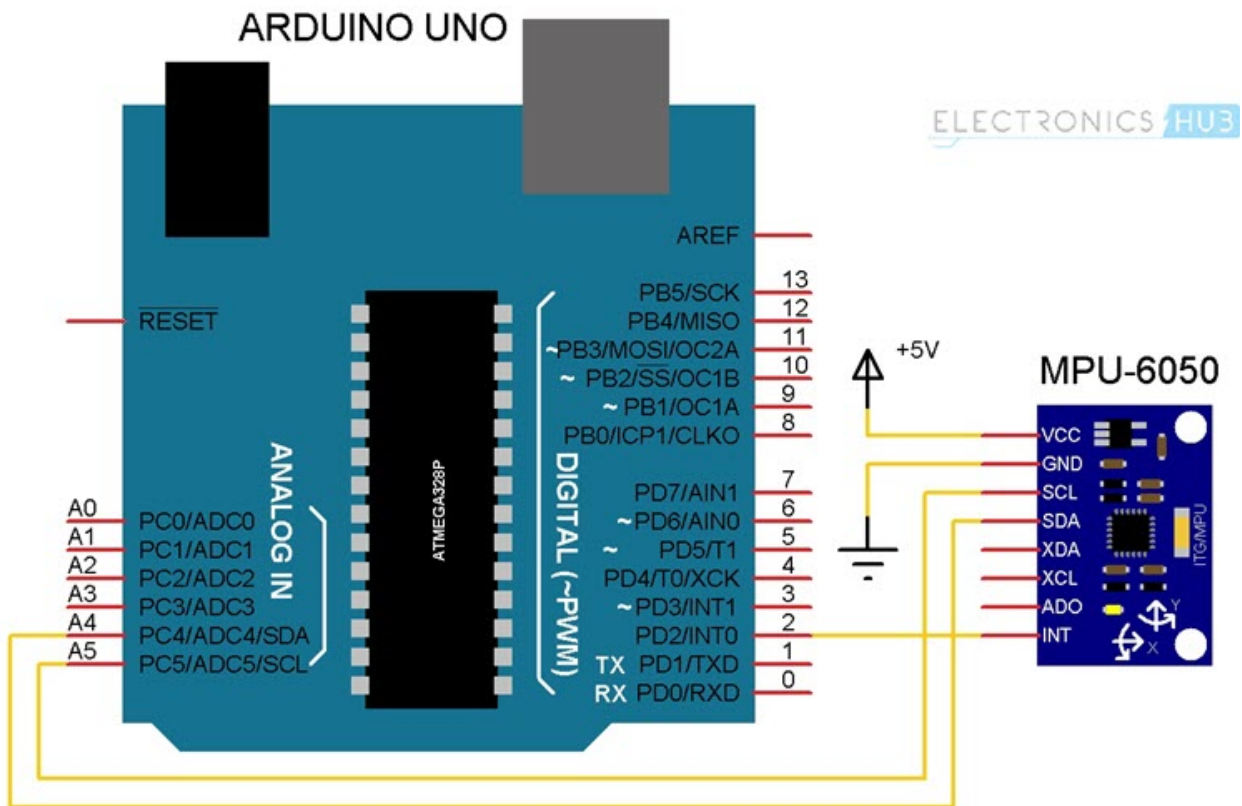
Circuit Diagram

The following image shows the circuit diagram for interfacing MPU6050 with Arduino UNO. As mentioned earlier, the interface between MPU6050 and Arduino must be implemented using I2C Protocol.

Hence, the SCL Pin of the Arduino (A5) is connected to the SCL Pin of the MPU6050. Similarly, the SDA Pin of the Arduino (A4) is connected to the SDA Pin of the MPU6050 board.

Additionally, we will be using the Interrupt feature of the MPU6050 to indicate (or interrupt) Arduino when the 1024 Byte FIFO buffer is full. So, connect the INT pin of the MPU6050 to the external interrupt 0 (INT0) pin of Arduino UNO i.e. Pin 2.

NOTE: In I2C Communication, the MPU-6050 always acts as a slave.



Reading RAW Values from MPU6050

Before uploading the actual program, we will first see a simple program to read the raw values from the Accelerometer, Gyroscope and the Temperature Sensor. Simply connect the SCL and SDA wires of the MPU6050 to the corresponding I2C Pins of Arduino (A4 and A5) and upload the following code.

```
#include<Wire.h>

const int MPU6050_addr=0x68;

int16_t AccX,AccY,AccZ,Temp,GyroX,GyroY,GyroZ;

void setup(){

Wire.begin();

Wire.beginTransmission(MPU6050_addr);

Wire.write(0x6B);

Wire.write(0);

Wire.endTransmission(true);

Serial.begin(9600);

}

void loop(){

Wire.beginTransmission(MPU6050_addr);

Wire.write(0x3B);

Wire.endTransmission(false);

Wire.requestFrom(MPU6050_addr,14,true);

AccX=Wire.read()<<8|Wire.read();

AccY=Wire.read()<<8|Wire.read();

AccZ=Wire.read()<<8|Wire.read();

Temp=Wire.read()<<8|Wire.read();

GyroX=Wire.read()<<8|Wire.read();

GyroY=Wire.read()<<8|Wire.read();
```

```

GyroZ=Wire.read()<<8|Wire.read();

Serial.print("AccX = "); Serial.print(AccX);

Serial.print(" || AccY = "); Serial.print(AccY);

Serial.print(" || AccZ = "); Serial.print(AccZ);

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

Serial.print(" || GyroX = "); Serial.print(GyroX);

Serial.print(" || GyroY = "); Serial.print(GyroY);

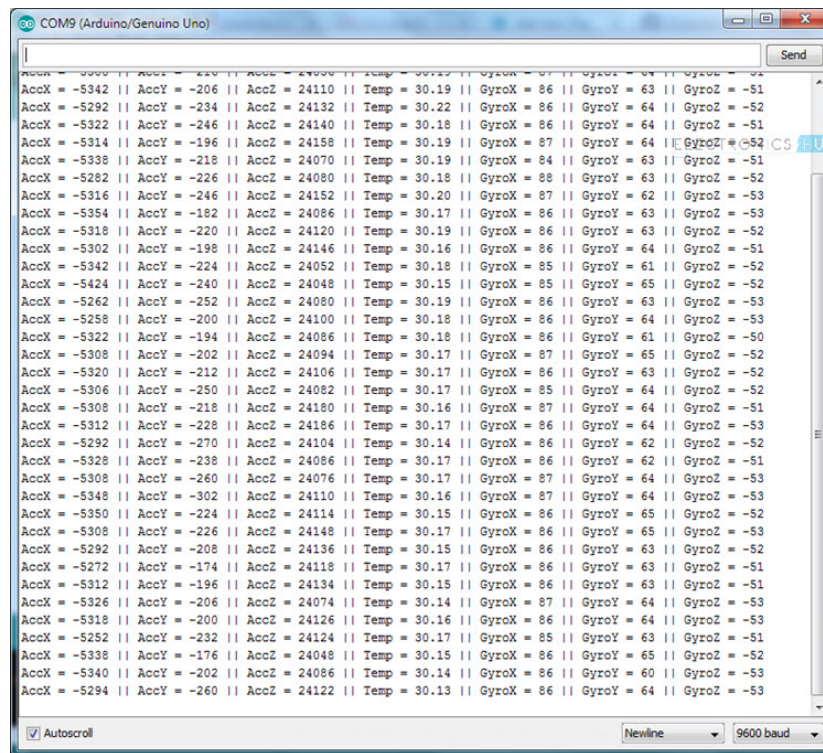
Serial.print(" || GyroZ = "); Serial.println(GyroZ);

delay(100);

}

```

If you open the serial terminal, you will get the raw values from the Accelerometer and Gyroscope and calibrated Temperature from the Temperature Sensor. The data looks something like this.



The screenshot shows a serial terminal window titled "COM9 (Arduino/Genuino Uno)". The window displays a continuous stream of sensor data lines. Each line contains 11 values separated by "||", representing AccX, AccY, AccZ, Temp, GyroX, GyroY, and GyroZ. The data is formatted as follows: `AccX = -5342 || AccY = -206 || AccZ = 24110 || Temp = 30.19 || GyroX = 86 || GyroY = 63 || GyroZ = -51`. The values for AccX, AccY, and AccZ are negative, while Temp, GyroX, GyroY, and GyroZ are positive. The window also shows a "Send" button at the top right and a status bar at the bottom with "Autoscroll" checked, "Newline" selected, and "9600 baud" set.