

10 DOF Sensor (SKU:SEN0140)



From Robot Wiki

Contents

- [1 Introduction](#)
 - [1.1 Applications](#)
- [2 Specification](#)
- [3 Connection Diagram](#)
- [4 Example Code](#)

Introduction

At the beginning, the inertial measurement unit is an electronic device that measures and reports on a craft's velocity, orientation, and gravitational forces, using a combination of accelerometers, gyroscopes, and magnetometers. Now IMUs are commonly used in the Human-computer interaction (HCI), navigational purposes and balancing technology used in the Segway Personal Transporter as we all know.

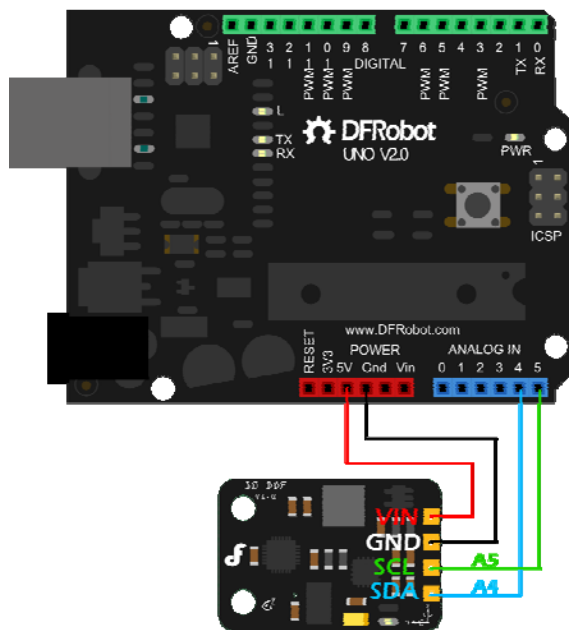
Applications

- Aircraft
- Balancing robots
- Indoor inertial navigation
- Altimeter
- Human-computer Interaction (HCI)

Specification

- Wide power input range from 3 to 8 volts
- Low noise LDO regulator
- Low cost IMU
- Interface: I2C
- M3x2 mounting holes for easily fixing in your mobile platforms, robots, HCI or UAVs
- LED power indication
- Integrate 10 dof sensors
 - Adxl345 accelerometer
 - ITG3200 gyro
 - HMC5883L Compass
 - BMP085 pressure sensor
- Compact size design and easy-to-use
- Compatible with Arduino controllers
- Electricity gold PCB
- Size: 26x18mm

Connection Diagram



10 Dof Connection Diagram

Example Code

Please download the [libraries](#) for the all the IMU sensors first!

```
#include <Wire.h>
#include <FreeSixIMU.h>
#include <FIMU_ADXL345.h>
#include <FIMU_ITG3200.h>
#include <HMC5883L.h>

float angles[3]; // yaw pitch roll
float heading;

short temperature;
long pressure;

// Set the FreeSixIMU object
FreeSixIMU sixDOF = FreeSixIMU();

HMC5883L compass;
// Record any errors that may occur in the compass.
int error = 0;

void setup(){

  Serial.begin(9600);
  Wire.begin();

  delay(5);
  sixDOF.init(); //init the Acc and Gyro
  delay(5);
  compass = HMC5883L(); // init HMC5883
```

```

    error = compass.SetScale(1.3); // Set the scale of the compass.

    error = compass.SetMeasurementMode(Measurement_Continuous); // Set the measurement mode to Continuous

    if(error != 0) // If there is an error, print it out.
        Serial.println(compass.GetErrorText(error));

    bmp085Calibration(); // init barometric pressure sensor

}

void loop(){

    sixDOF.getEuler(angles);

    temperature = bmp085GetTemperature(bmp085ReadUT());
    pressure = bmp085GetPressure(bmp085ReadUP());

    getHeading();
    PrintData();

    delay(300);
}

void getHeading(){
    // Retrieve the raw values from the compass (not scaled).
    MagnetometerRaw raw = compass.ReadRawAxis();

    // Retrieved the scaled values from the compass (scaled to the configured scale).
    MagnetometerScaled scaled = compass.ReadScaledAxis();

    // Values are accessed like so:
    int MilliGauss_OnThe_XAxis = scaled.XAxis; // (or YAxis, or ZAxis)

    // Calculate heading when the magnetometer is level, then correct for signs of axis.

```

```

heading = atan2(scaled.YAxis, scaled.XAxis);

float declinationAngle = 0.0457;
heading += declinationAngle;

// Correct for when signs are reversed.
if(heading < 0)
    heading += 2*PI;

// Check for wrap due to addition of declination.
if(heading > 2*PI)
    heading -= 2*PI;

// Convert radians to degrees for readability.
heading = heading * 180/M_PI;
}

void PrintData(){

    Serial.print("Eular Angle: ");
    Serial.print(angles[0]);
    Serial.print(" ");
    Serial.print(angles[1]);
    Serial.print(" ");
    Serial.print(angles[2]);
    Serial.print(" ");
    Serial.print("Heading: ");
    Serial.print(heading);
    Serial.print(" ");
    Serial.print("Pressure: ");
    Serial.print(pressure, DEC);
    Serial.println(" Pa");
}

```

Open the arduino IDE serial monitor, you will get:

