Der MPU6050 6-Achsen Sensor

Inhalt

1. Aufgabenstellung

Da wir das PCB fertiggestellt und in Betrieb genommen haben, wollen wir mit dem Design der nächsten Platine beginnen. Diese soll unter Anderem mit einigen Sensoren ausgestattet werden. Diese Woche entschlossen wir uns, den MPU6050 Sensor auszutesten.

2. Der Sensor

Der MPU6050 – Sensor verfügt über ein 3- Achsen Accelerometer, ein 3- Achsen Gyroskop und einen Temperatursensor. Die Daten werden über den I2C – Bus eingelesen. Im Sensor befinden sich MEMS (= „Micro-Electrical-Mechanical Systems“). Das sind winzige elektro-mechanische Systeme die billig und in Masse produziert werden können.

Im rechten Bild von Abb. 2 drückt die Beschleunigung nach links die mittlere Kondensatorplatte näher zur rechten. Das verändert die Kapazität zwischen den Platten. Misst man die Kapazitäten kann man so auf die Beschleunigung rückschließen.

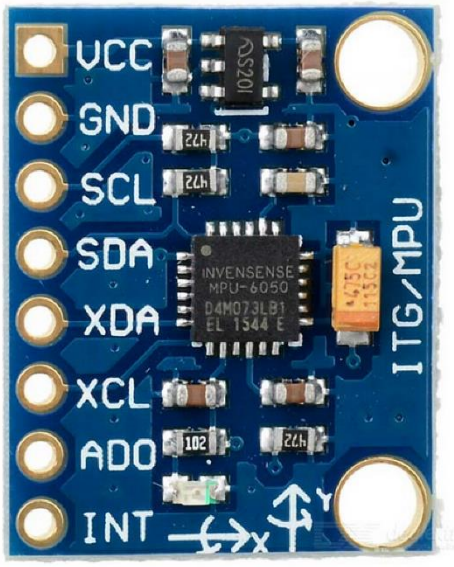

Abbildung 1: Das Modul

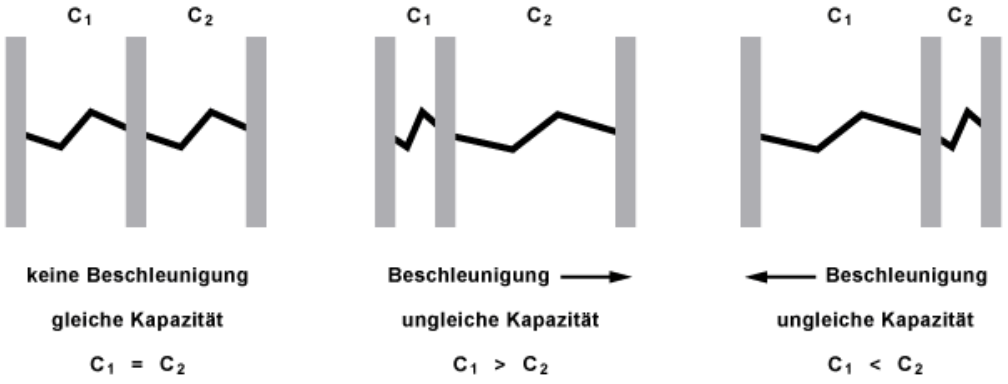

Abbildung 2: Funktionsweise des Accelerometers (https://www.elektronik-kompendium.de/sites/bau/1503041.htm)

Der Acceleromter misst Beschleunigungen entlang der x, y, z-Achsen, während das Gyroskop Geschwindigkeiten um die x,y,z – Achen misst.

3. Der Code

```
// Basic demo for accelerometer readings from Adafruit MPU6050

#include <Adafruit_MPU6050.h>
#include <Adafruit_Sensor.h>
#include <Wire.h>

Adafruit_MPU6050 mpu;

int a_x_offset, a_y_offset, a_z_offset;
int g_x_offset, g_y_offset, g_z_offset;

int a_x, a_y, a_z;
int g_x, g_y, g_z;
int sum;
void setup(void) {
  Serial.begin(115200);
  mpu.begin();

  mpu.setAccelerometerRange(MPU6050_RANGE_8_G);
  mpu.setGyroRange(MPU6050_RANGE_500_DEG);
  mpu.setFilterBandwidth(MPU6050_BAND_260_HZ);
  Serial.print("Filter bandwidth set to: ");
  delay(100);

  //calibration
  sensors_event_t a, g, temp;
  mpu.getEvent(&a, &g, &temp);
  a_x_offset = a.acceleration.x;
  a_y_offset = a.acceleration.y;
  a_z_offset = a.acceleration.z;

  g_x_offset = g.gyro.x;
  g_y_offset = g.gyro.y;
  g_z_offset = g.gyro.z;
}
```

Abbildung 3: Setup- Code für das Gyroskop

Am Anfang werden Wertebereiche des MPUs definiert. Der Accelerometer misst jetzt bis zu ± 8G also bis zu 78.4 m/s². Das Gyro ±500 °/s. setFilterBandwidth() setzt die Grenzfrequenz eines digitalen Filters, der Teil des Moduls ist. Ist die Grenzfrequenz niedriger wird über mehr Messdaten integriert. Das macht den Sensor langsam aber weniger „shaky“.

Im Zuge des Labors wurde das Modul zunächst kalibriert. Es wurde eine Messung im Ruhezustand getätigt und diese wurden von folgenden Messungen abgezogen. Das lässt Gravitation und Potentielle Sensor-Drifts wegfallen. Dann wurden die absoluten Werte aller Sensoren addiert und ausgegeben.

```
void loop() {

  /* Get new sensor events with the readings */
  sensors_event_t a, g, temp;
  mpu.getEvent(&a, &g, &temp);

  a_x = a.acceleration.x - a_x_offset;
  a_y = a.acceleration.y - a_y_offset;
  a_z = a.acceleration.z - a_z_offset;

  g_x = g.gyro.x - g_x_offset;
  g_y = g.gyro.y - g_y_offset;
  g_z = g.gyro.z - g_z_offset;

  sum = abs(a_x) + abs(a_y) + abs(a_z) + abs(g_x) + abs(g_y) + abs(g_z);
  /* Print out the values */
  Serial.print(a_x);
  Serial.print(",");
  Serial.print(a_y);
  Serial.print(",");
  Serial.print(a_z);
  Serial.print(",");
  Serial.print(g_x);
  Serial.print(",");
  Serial.print(g_y);
  Serial.print(",");
  Serial.print(g_z);
  Serial.print(",");
  Serial.print(sum);
  Serial.println("");

  //Serial.print("Temperature: ");
  //Serial.print(temp.temperature);
  //Serial.println(" degC");

  delay(100);
```

Abbildung 4: Main-Loop für Gyroskop

Im Zuge der Recherche für dieses Protokoll fiel jedoch auf, dass hier lineare Beschleunigungen einfach mit Winkelgeschwindigkeiten addiert werden, was nur bedingt Sinn macht. Für einen sehr groben Richtwert genügt es jedoch.

4. Next Steps

Sollten wir uns dafür entschließen, nächste Woche weiter an diesem Sensor zu arbeiten gäbe es in der Dokumentation für dieses Modul Programmcode die eine bessere Kalibrierung des Sensors

ermöglichen. Außerdem gibt es einen Code der alle Messdaten kombiniert und auf die Lage des Sensors im Raum rückschließen kann.

Die Lage im Raum würde sich jedoch nicht bedeutend ändern, wenn der Sensor im Roboter eingebaut ist. Auch die jetzige Kalibrierung reicht wohl für unseren Anwendungszweck aus. Deswegen werde ich mich nächste Woche eher mit der Einbindung weiterer Sensoren beschäftigen.

5. Abbildungsverzeichniss

6. Quellen

https://wolles-elektronikkiste.de/mpu6050-beschleunigungssensor-und-gyroskop

https://www.elektronik-kompendium.de/sites/bau/1503041.htm

https://github.com/ElectronicCats/mpu6050/wiki