KE0075 keyes module Accéléromètre **MPU6050** **capteur moderne d'accélération couplé à un gyroscope**

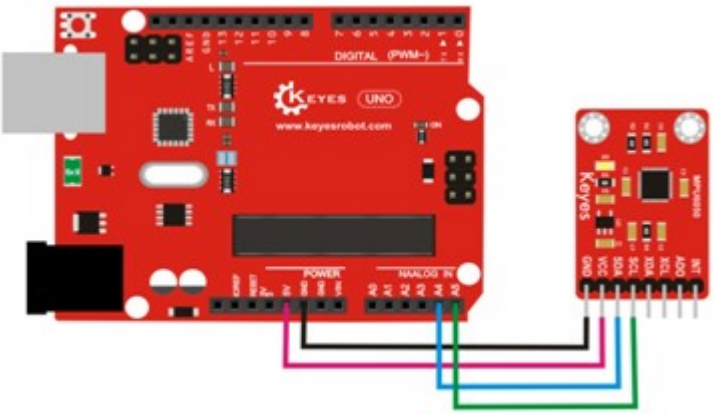
Vue d'ensemble

Ce module est un module de capteur à six axes à accélération et à gyroscope à trois axes hautes performances, utilisant la puce MPU6050 comme cœur, utilisant le processeur de mouvement numérique DMP, via l'interface IIC, les données calculées en fonction de l'attitude de sortie. Il présente les caractéristiques d'une petite taille et d'une utilisation pratique. Le module est livré avec 2 trous de positionnement pour vous permettre de fixer le module à d'autres appareils.

Spécifications

- Tension de travail: 3.3-5V (DC)
- Méthode de communication: protocole de communication standard IIC
- Convertisseur AD 16 bits intégré dans la puce, sortie de données 16 bits
- Gamme gyroscopique: ± 250 500 1000 2000 ° / s
- Gamme d'accélération: ± 2 ± 4 ± 8 ± 16g
- Courant de fonctionnement du gyroscope: 5mA
- Courant de veille gyroscopique: 5μA
- Courant de fonctionnement de l'accélérateur: 350μA
- Pas de pin: 2.54mm
- Interface: interface 8PIN
- Poids: 3.0g

Schéma de connexion



Code de test

```
 // I2Cdev et MPU6050 doivent être installés en tant que bibliothèques,
sinon les fichiers .cpp / .h
 // pour les deux classes doit être dans le chemin d'inclusion de votre
projet
 #include "I2Cdev.h"
 #include "MPU6050.h"

 // La bibliothèque Arduino Wire est requise si la mise en œuvre I2Cdev
I2CDEV_ARDUINO_WIRE
 // est utilisé dans I2Cdev.h
 #if I2CDEV_IMPLEMENTATION == I2CDEV_ARDUINO_WIRE
     #include "Wire.h"
 #endif

 // l'adresse I2C par défaut de la classe est 0x68
 // des adresses I2C spécifiques peuvent être passées ici en tant que
paramètre
 // AD0 low = 0x68 (valeur par défaut pour la carte d'évaluation
InvenSense)
 // AD0 haut = 0x69
 MPU6050 accelgyro;
 // MPU6050 accelgyro (0x69); // <- à utiliser pour AD0 élevé

 Int16_t ax, ay, az;
 Int16_t gx, gy, gz;
 // décommenter "OUTPUT_READABLE_ACCELGYRO" si vous voulez voir une
séparation par tabulation
 // liste des valeurs d'accélération X / Y / Z, puis gyroscopiques X / Y /
Z en décimale.
 // pas si facile à analyser, et lent (plus) sur UART.
 #define OUTPUT_READABLE_ACCELGYRO

 // décommenter "OUTPUT_BINARY_ACCELGYRO" pour envoyer les 6 axes de
données en 16 bits
 // binaire, l'un après l'autre, c'est très rapide (aussi vite que
possible)
 // sans compression ni perte de données), facile à analyser, mais
impossible à lire
 // pour un humain.
 // # définit OUTPUT_BINARY_ACCELGYRO
```

```
 #define LED_PIN 13
 Bool blinkState = false;

 Annuler la configuration () {
     // rejoindre le bus I2C (la bibliothèque I2Cdev ne le fait pas
automatiquement)
     #if I2CDEV_IMPLEMENTATION == I2CDEV_ARDUINO_WIRE
         Wire.begin ();
     #elif I2CDEV_IMPLEMENTATION == I2CDEV_BUILTIN_FASTWIRE
         Fastwire :: setup (400, true);
     #endif

     // initialise la communication série
     Serial.begin (9600);

     // initialise le périphérique
     Serial.println ("Initialisation de périphériques I2C ...");
     Accelgyro.initialize ();

     // vérifier la connexion
     Serial.println ("Test des connexions de périphérique ...");
     Serial.println (accelgyro.testConnection ()? "Connexion MPU6050
réussie": "Connexion MPU6050 échouée");

     // utilise le code ci-dessous pour changer les valeurs d'accélération
/ gyroscopie
     / *
     Serial.println ("Mise à jour des décalages de capteurs internes ...");
     // -76 -2359 1688 0 0 0
     Serial.print (accelgyro.getXAccelOffset ()); Serial.print ("\ t"); //
-76
     Serial.print (accelgyro.getYAccelOffset ()); Serial.print ("\ t"); //
-2359
     Serial.print (accelgyro.getZAccelOffset ()); Serial.print ("\ t"); //
1688
     Serial.print (accelgyro.getXGyroOffset ()); Serial.print ("\ t"); // 0
     Serial.print (accelgyro.getYGyroOffset ()); Serial.print ("\ t"); // 0
     Serial.print (accelgyro.getZGyroOffset ()); Serial.print ("\ t"); // 0
     Serial.print ("\ n");
     accelgyro.setXGyroOffset (220);
     accelgyro.setYGyroOffset (76);
     accelgyro.setZGyroOffset (-85);
     Serial.print (accelgyro.getXAccelOffset ()); Serial.print ("\ t"); //
-76
```

```
    Serial.print (accelgyro.getYAccelOffset ()); Serial.print ("\ t"); //
-2359
    Serial.print (accelgyro.getZAccelOffset ()); Serial.print ("\ t"); //
1688
    Serial.print (accelgyro.getXGyroOffset ()); Serial.print ("\ t"); // 0
    Serial.print (accelgyro.getYGyroOffset ()); Serial.print ("\ t"); // 0
    Serial.print (accelgyro.getZGyroOffset ()); Serial.print ("\ t"); // 0
    Serial.print ("\ n");
    * /

    // configure Arduino LED pour
    pinMode (LED_PIN, OUTPUT);
 }

 Boucle vide () {
    // lit les mesures brutes d'accélération / gyroscope de l'appareil
    accelgyro.getMotion6 (& ax, & ay, & az, & gx, & gy, & gz);

    // ces méthodes (et quelques autres) sont également disponibles
    //accelgyro.getAcceleration(&ax, & ay, & az);
    //accelgyro.getRotation(&gx, & gy, & gz);

    #ifdef OUTPUT_READABLE_ACCELGYRO
        // affiche les valeurs d'accél / gyromètre x / y / z séparées par
des tabulations
        Serial.print ("a / g: \ t");
        Serial.print (ax); Serial.print ("\ t");
        Serial.print (ay); Serial.print ("\ t");
        Serial.print (az); Serial.print ("\ t");
        Serial.print (gx); Serial.print ("\ t");
        Serial.print (gy); Serial.print ("\ t");
        Serial.println (gz);
    #endif

    #ifdef OUTPUT_BINARY_ACCELGYRO
        Serial.write ((uint8_t) (ax >> 8)); Serial.write ((uint8_t) (ax &
0xFF));
        Serial.write ((uint8_t) (ay >> 8)); Serial.write ((uint8_t) (ay &
0xFF));
        Serial.write ((uint8_t) (az >> 8)); Serial.write ((uint8_t) (az &
0xFF));
        Serial.write ((uint8_t) (gx >> 8)); Serial.write ((uint8_t) (gx &
0xFF));
```

```
        Serial.write ((uint8_t) (gy >> 8)); Serial.write ((uint8_t) (gy &
0xFF));
        Serial.write ((uint8_t) (gz >> 8)); Serial.write ((uint8_t) (gz &
0xFF));
    #endif

    // clignotent LED pour indiquer l'activité
    blinkState =! blinkState;
    digitalWrite (LED_PIN, blinkState);
 }
```

Résultat du test

Connectez la ligne conformément à la figure ci-dessus, gravez le code, après la mise sous tension, après la mise sous tension, vous pouvez voir la valeur correspondante sur le moniteur série du logiciel, comme indiqué ci-dessous.