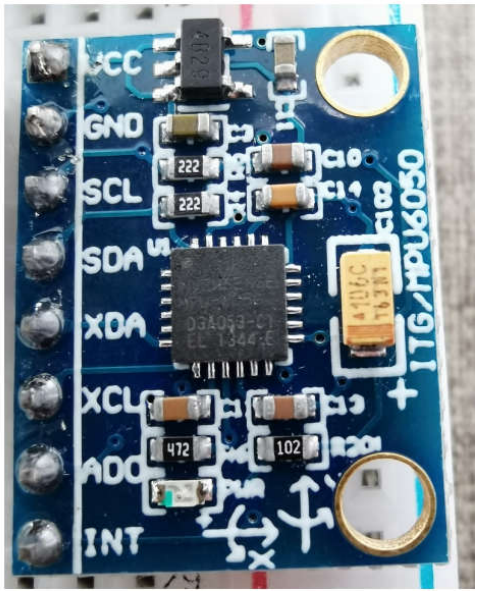
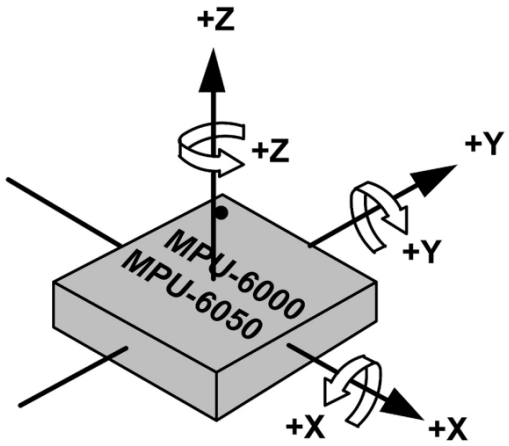
智能系统与控制



树莓派：GPIO-陀螺仪-MPU6050 姿态角解算（DMP）

```
DMP initialization was successful
0x0
42
0
 roll: 0.36  pitch: 3.73  yaw: -4.54
```

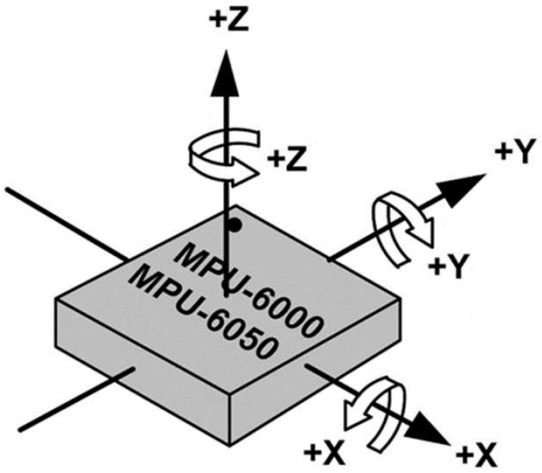


于泓
鲁东大学
信息与电气工程学院

2021.11.8

MPU6050

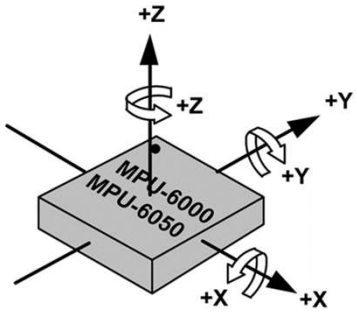
MPU-6050 是全球首款也是唯一一款为智能手表、平板电脑和可穿戴设备提供运动追踪功能的低成本 6 轴运动传感器。它具有低功耗、低成本、高性能等特点被广泛的应用于平衡车、自平稳云台、动作捕捉器等产品的设计中。

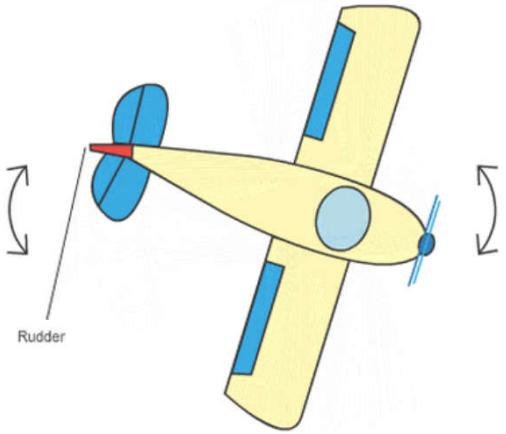


获取三个轴的加速度和角速度

三种姿态角

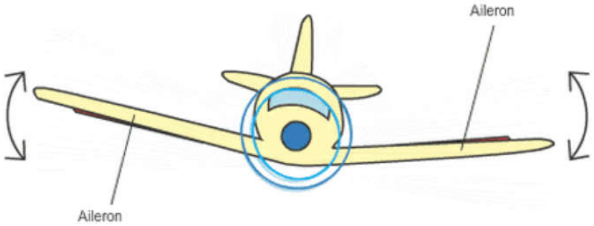
横滚roll，俯仰pitch，偏航yaw





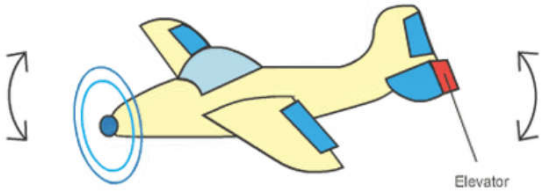
Use the rudder to control
Yaw

绕Z轴



Use the ailerons to control
Roll

绕X轴



Use the elevator to control
Pitch

绕Y轴

$$\begin{cases} pitch=\arcsin\left(2(q_0\cdot q_2-q_1\cdot q_3)\right) \\ roll=\arctan\left(\dfrac{2(q_0\cdot q_1+q_2\cdot q_3)}{q_0^2-q_1^2-q_2^2+q_3^2}\right) \\ yaw=\arctan\left(\dfrac{2(q_0\cdot q_3+q_1\cdot q_2)}{q_0^2+q_1^2-q_2^2-q_3^2}\right) \end{cases} \text{ 或 } \begin{cases} pitch=\arcsin\left(2(q_0\cdot q_2-q_1\cdot q_3)\right) \\ roll=\arctan\left(\dfrac{2(q_2\cdot q_3+q_0\cdot q_1)}{1-2(q_1^2+q_2^2)}\right) \\ yaw=\arctan\left(\dfrac{2(q_0\cdot q_3+q_1\cdot q_2)}{1-2(q_2^2+q_3^2)}\right) \end{cases}$$

注：单位四元数 $q_0^2+q_1^2+q_2^2+q_3^2=1$

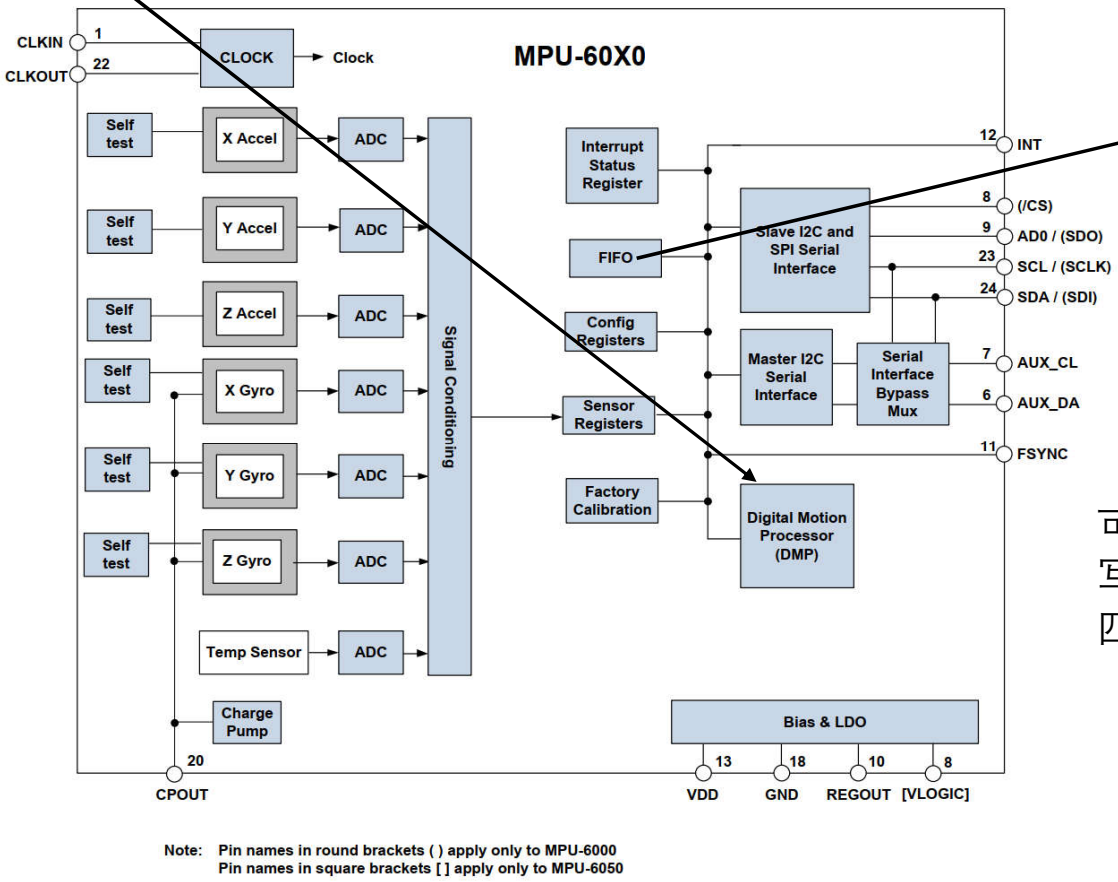
利用 三轴的加速度，角速度
计算4元组q，利用q计算俯仰角

整个过程比较复杂

Motion Driver是Invensense针对其运动传感器的软件包，并非全部开源，

```
pitch = asin(-2 * q1 * q3 + 2 * q0* q2)
roll  = atan2(2 * q2 * q3 + 2 * q0 * q1, -2 * q1 * q1 - 2 * q2* q2 + 1)
yaw   = atan2(2*(q1*q2 + q0*q3),q0*q0+q1*q1-q2*q2-q3*q3)
```

Motion Driver是Invensense针对其运动传感器的软件包，并非全部开源，

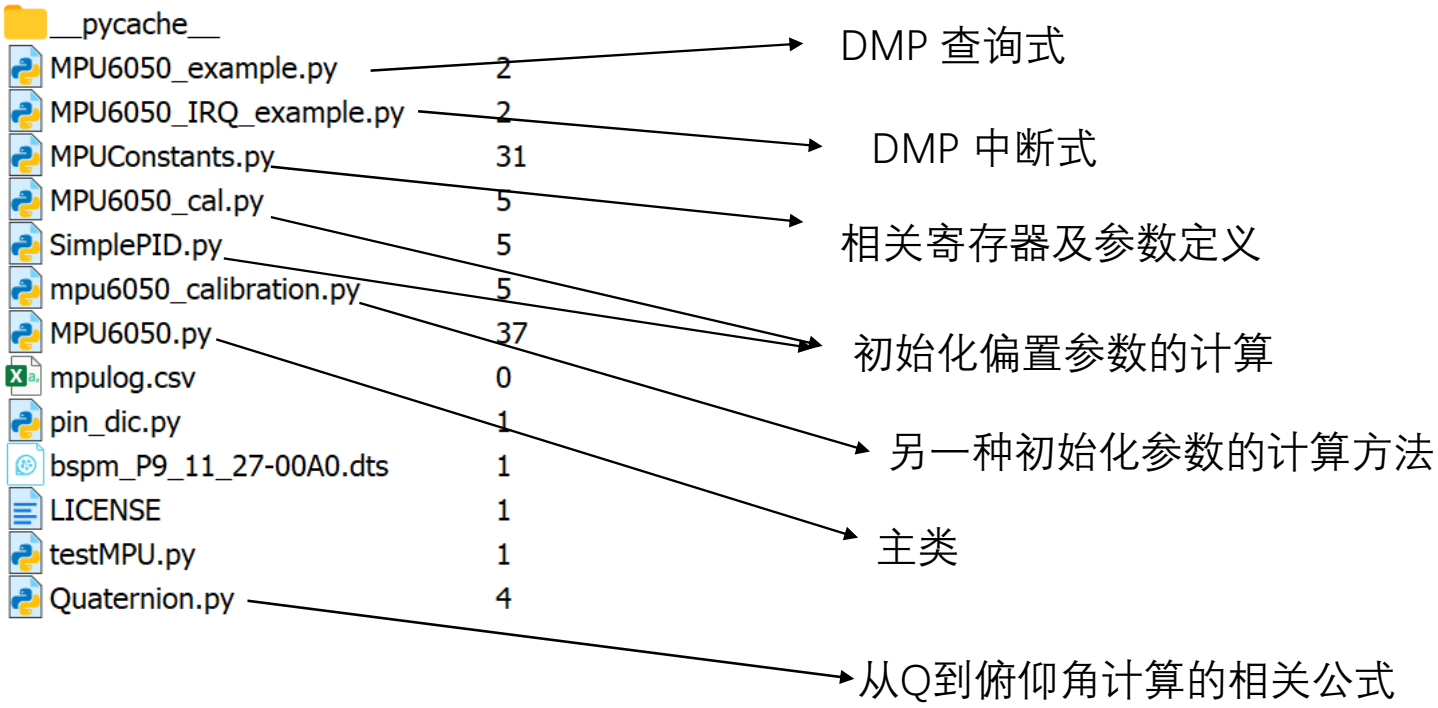


Note: Pin names in round brackets ( ) apply only to MPU-6000
Pin names in square brackets [ ] apply only to MPU-6050

数据同步

可以把编译好的程序
写入DMP，并利用DMP进行
四元组q的解算

参考代码： https://github.com/thisisG/MPU6050-I2C-Python-Class



查询方式

```
from MPU6050 import MPU6050
import time

if __name__ == "__main__":
    i2c_bus = 1
    device_address = 0x68

    x_accel_offset = int(926)
    y_accel_offset = int(2136)
    z_accel_offset = int(-856)
    x_gyro_offset = int(36)
    y_gyro_offset = int(-8)
    z_gyro_offset = int(-28)

    # x_avg_read: 0.08 x_avg_offset: 926.423499999996
    # y_avg_read: -0.72 y_avg_offset: 2136.152999999997
    # z_avg_read: 0.34 z_avg_offset: -856.0713749999996
    # x_avg_read: 0.16 x_avg_offset: 35.570499999999996
    # y_avg_read: -0.07 y_avg_offset: -8.081437499999883
    # z_avg_read: -0.01 z_avg_offset: -28.00306249999999

    enable_debug_output = True
    mpu = MPU6050(i2c_bus, device_address, x_accel_offset, y_accel_offset,z_accel_offset, x_gyro_offset, y_gyro_offset, z_

    mpu.dmp_initialize()
    mpu.set_DMP_enabled(True)
    mpu_int_status = mpu.get_int_status()
    print(hex(mpu_int_status))

    packet_size = mpu.DMP_get_FIFO_packet_size()
    print(packet_size)
    FIFO_count = mpu.get_FIFO_count()
    print(FIFO_count)
```

初始化的偏置，通过运行mpu6050_cal.py来获取

Mpu初始化

DMP 初始化（程序写入）

FIFO的大小

```
FIFO_buffer = [0]*64
try:
    while True:
        FIFO_count = mpu.get_FIFO_count()
        mpu_int_status = mpu.get_int_status()

        # If overflow is detected by status or fifo count we want to reset
        if (FIFO_count == 1024) or (mpu_int_status & 0x10):
            mpu.reset_FIFO()
            print('overflow!')
        # Check if fifo data is ready
        elif (mpu_int_status & 0x02):
            # Wait until packet_size number of bytes are ready for reading, default
            # is 42 bytes
            while FIFO_count < packet_size:
                FIFO_count = mpu.get_FIFO_count()
            FIFO_buffer = mpu.get_FIFO_bytes(packet_size)
            accel = mpu.DMP_get_acceleration_int16(FIFO_buffer)
            quat = mpu.DMP_get_quaternion_int16(FIFO_buffer)
            grav = mpu.DMP_get_gravity(quat)
            roll_pitch_yaw = mpu.DMP_get_euler_roll_pitch_yaw(quat, grav)

            str_show = "roll: %.2f  pitch: %.2f  yaw: %.2f        "%(roll_pitch_yaw.x,roll_pitch_yaw.y,roll_pitch_yaw.z)

            print("\r %s"%(str_show),end='')
except KeyboardInterrupt:
    print('\n Ctrl + C QUIT')
```

数据准备好

计算俯仰角

```python
import time
from MPU6050 import MPU6050
from pin_dic import pin_dic
import RPi.GPIO as GPIO

def action(channel):
    global mpu
    global FIFO_buffer
    global packet_size

    try:
        FIFO_count = mpu.get_FIFO_count()
        mpu_int_status = mpu.get_int_status()
    except:
        return
    # If overflow is detected by status or fifo count we want to reset
    if (FIFO_count == 1024) or (mpu_int_status & 0x10):
        mpu.reset_FIFO()
        # print('overflow!')
        return
    # Check if fifo data is ready
    elif (mpu_int_status & 0x02):
        # Wait until packet_size number of bytes are ready for reading, default
        # is 42 bytes
        while FIFO_count < packet_size:
            FIFO_count = mpu.get_FIFO_count()
        try:
            FIFO_buffer = mpu.get_FIFO_bytes(packet_size)
        except:
            return
        accel = mpu.DMP_get_acceleration_int16(FIFO_buffer)
        quat = mpu.DMP_get_quaternion_int16(FIFO_buffer)
        grav = mpu.DMP_get_gravity(quat)
        roll_pitch_yaw = mpu.DMP_get_euler_roll_pitch_yaw(quat, grav)

        str_show = "roll: %.2f  pitch: %.2f  yaw: %.2f        "%(roll_pitch_yaw.x,roll_pitch_yaw.y,roll_pitch_yaw.z)
```

角度解算

定义中断引脚

```
if __name__ == "__main__":
    pin_IRQ = pin_dic['G27']

    i2c_bus = 1
    device_address = 0x68

    x_accel_offset = int(926)
    y_accel_offset = int(2136)
    z_accel_offset = int(-856)
    x_gyro_offset = int(36)
    y_gyro_offset = int(-8)
    z_gyro_offset = int(-28)

    enable_debug_output = True

    mpu = MPU6050(i2c_bus, device_address, x_accel_offset,
                  y_accel_offset, z_accel_offset, x_gyro_offset,
                  y_gyro_offset, z_gyro_offset, enable_debug_output)

    mpu.dmp_initialize()
    mpu.set_DMP_enabled(True)
    mpu_int_status = mpu.get_int_status()
    print(hex(mpu_int_status))

    packet_size = mpu.DMP_get_FIFO_packet_size()
    print(packet_size)
    FIFO_count = mpu.get_FIFO_count()
    print(FIFO_count)

    FIFO_buffer = [0]*64
```

```
GPIO.setmode(GPIO.BOARD)
GPIO.setup(pin_IRQ, GPIO.IN)
GPIO.add_event_detect(pin_IRQ, GPIO.RISING, callback=action)

try:
    while True:
        pass
except KeyboardInterrupt:
    GPIO.cleanup()
```

设置中断函数

注意：

```
692     # dmpConfig has size MPU6050_DMP_CONFIG_SIZE = 192
693     dmpConfig = [
694         #  BANK    OFFSET  LENGTH  [DATA]
695         0x03, 0x7B, 0x03, 0x4C, 0xCD, 0x6C,  # FCFG_1 in
696         0x03, 0xAB, 0x03, 0x36, 0x56, 0x76,  # FCFG_3 in
697         0x00, 0x68, 0x04, 0x02, 0xCB, 0x47, 0xA2,
698         # D_0_104 inv_set_gyro_calibration
699         0x02, 0x18, 0x04, 0x00, 0x05, 0x8B, 0xC1,
700         # D_0_24 inv_set_gyro_calibration
701         0x01, 0x0C, 0x04, 0x00, 0x00, 0x00, 0x00,
702         # D_1_152 inv_set_accel_calibration
703         0x03, 0x7F, 0x06, 0x0C, 0xC9, 0x2C, 0x97, 0x97,
704         # FCFG_2 inv_set_accel_calibration
705         0x03, 0x89, 0x03, 0x26, 0x46, 0x66,  # FCFG_7 inv_set_accel_calibration
706         0x00, 0x6C, 0x02, 0x20, 0x00,  # D_0_108 inv_set_accel_calibration
707         0x02, 0x40, 0x04, 0x00, 0x00, 0x00, 0x00,
708         # CPASS_MTX_00 inv_set_compass_calibration
709         0x02, 0x44, 0x04, 0x00, 0x00, 0x00, 0x00,  # CPASS_MTX_01
710         0x02, 0x48, 0x04, 0x00, 0x00, 0x00, 0x00,  # CPASS_MTX_02
711         0x02, 0x4C, 0x04, 0x00, 0x00, 0x00, 0x00,  # CPASS_MTX_10
712         0x02, 0x50, 0x04, 0x00, 0x00, 0x00, 0x00,  # CPASS_MTX_11
713         0x02, 0x54, 0x04, 0x00, 0x00, 0x00, 0x00,  # CPASS_MTX_12
714         0x02, 0x58, 0x04, 0x00, 0x00, 0x00, 0x00,  # CPASS_MTX_20
715         0x02, 0x5C, 0x04, 0x00, 0x00, 0x00, 0x00,  # CPASS_MTX_21
716         0x02, 0xBC, 0x04, 0x00, 0x00, 0x00, 0x00,  # CPASS_MTX_22
717         0x01, 0xEC, 0x04, 0x00, 0x00, 0x40, 0x00,
```

```
        0x02, 0x16, 0x02, 0x00, 0x05]  # D_0_22 inv_set_fifo_rate

# This very last 0x01 WAS a 0x09, which drops the FIFO rate dow
# 0x07 is 25 Hz, 0x01 is 100Hz. Going faster than 100Hz (0x00=2
# to result in very noisy data. DMP output frequency is calcula
# using this equation: (200Hz / (1 + value))

# It is important to make sure the host processor can keep up w
# and processing the FIFO output at the desired rate. Handling
# cleanly is also a good idea.

# dmpUpdates has size MPU6050_DMP_UPDATES_SIZE = 47
```

决定了FIFO的采样速度
设置太小容易出问题