MPU6050 模块是 InvenSense 公司推出的一款低成本的 6 轴传感器模块，包括三轴加速度，三轴角速度。其体积小巧，用途非常广。做平衡小车，四轴飞行器，飞行鼠标等等，都是必不可少而且是最优的传感器解决方案。

我们可以通过 IIC 通讯从 MPU6050 的 XYZ 三个轴的角速度分量和加速度分量还有温度。
MPU6050 在上电以后需要等待一段时间，因为 6050 其实也是一块 MCU(单片机)，里面有自己的处理程序，延时一段时间等待其内部初始化成功，再进行其他的操作。接下来就是初始化一些 6050 的设定，当然这些设定是通过写寄存器来设置的 。

```
#define SMPLRT_DIV    0x19  //陀螺仪采样率，典型值：0x07(125Hz)
#define CONFIG        0x1A  //低通滤波频率，典型值：0x06(5Hz)
#define GYRO_CONFIG   0x1B  //陀螺仪自检及测量范围，典型值：0x18(不自检，2000deg/s)
#define ACCEL_CONFIG  0x1C  //加速计自检、测量范围及高通滤波频率，典型值：0x01(不自检，2G，5Hz)
```

可以在 MPU6050.h 头文件中找到这些设定。
MPU60x0 对陀螺仪和加速度计分别采用了三个 16 位的 ADC 将其测量的模拟量转化为可输出的数字量。为了精确跟踪快速和慢速的运动，传感器的测量范围都是用户可控的，陀螺仪可测范围为±250，±500，±1000，±2000°/s（dps）.加速度计可测范围±2，±4，±8，±16g。

SMPLRT_DIV 8 位无符号值，通过该值将陀螺仪输出分频，得到采样频率（采样频率，也称为采样速度或者采样率，定义了每秒从连续信号中提取并组成离散信号的采样个数，它用赫兹（Hz）来表示）。采样率的计算公式：

采样率=陀螺仪的输出率/（1+ SMPLRT_DIV）

当低通数字滤波不开启的时候，陀螺仪的输出率为 8Khz（DLPF_CFG=0 or 7），当低通数字滤波开启的时候，陀螺仪的输出频率为 1Khz。

CONFIG 8 位无符号值，同时设置 EXT_SYNC_SET 3 位无符号值，配置帧同步引脚的采样，DLPF_CFG 3 位无符号值，配置数字低通滤波器。位分配如下表：

2）Register 26 – Configuration （CONFIG）

Type: Read/Write

Register (Hex)	Register (Decimal)	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
1A	26	-	-	EXT_SYNC_SET[2:0]			DLPF_CFG[2:0]		

通过配置 EXT_SYNC_SET，可以对连接到 FSYNC 引脚的一个外部信号进行采样。FSYNC 引脚上的信号变化会被锁存，这样就能捕获到很短的频闪信号。
采样结束后，锁存器将复位到当前的 FSYNC 信号状态。
根据下面的表格定义的值，采集到的数据会替换掉数据寄存器中上次接收到的有效数据。配置表格如下：

EXT_SYNC_SET	FSYNC Bit Location
0	Input disabled
1	TEMP_OUT_L[0]
2	GYRO_XOUT_L[0]
3	GYRO_YOUT_L[0]
4	GYRO_ZOUT_L[0]
5	ACCEL_XOUT_L[0]
6	ACCEL_YOUT_L[0]
7	ACCEL_ZOUT_L[0]

我们在例程的初始化函数是这样写的

Single_Write_IIC(SLAVEADRESS , CONFIG , 0x06);

0x06=00 000 110

可以得知，我们 EXT_SYNC_SET 位置给的是 0，即不采用外部帧同步。低通滤波器设置的二进制值为 110 即 6

DLPF_CFG	Accelerometer (F_s = 1kHz)		Gyroscope		
	Bandwidth (Hz)	Delay (ms)	Bandwidth (Hz)	Delay (ms)	Fs (kHz)
0	260	0	256	0.98	8
1	184	2.0	188	1.9	1
2	94	3.0	98	2.8	1
3	44	4.9	42	4.8	1
4	21	8.5	20	8.3	1
5	10	13.8	10	13.4	1
6	5	19.0	5	18.6	1
7	RESERVED		RESERVED		8

对于加速度我们采用的是 5HZ，延时 19ms，陀螺仪 5HZ，延时 18.6HZ，陀螺仪输出率为 1Khz。由于上面的 SMPLRT_DIV 我们使用的是默认值 0x07，所以根据前面的公式：

1000/（7+1）=125hz，和注释中相同。

GYRO_CONFIG 8 位无符号值，同时设置 X,Y,Z 三个轴的陀螺仪自检和他们的测量范围。位分布图如下表：

Type: Read/Write

Register (Hex)	Register (Decimal)	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
1B	27	XG_ST	YG_ST	ZG_ST	FS_SEL[1:0]		-	-	-

这里涉及到三个轴的自检（self-test）。这个自检有什么用呢？官方对于自检过程的解释：当自检功能被激活以后，片上的电气设备（即 6050）将会开启适当的传感器，这个动作将传感器的获取的数据与厂家预设的数据进行做差，当差值大到超出规定则无法通过自检，差值在允许范围内则通过自检。
官方手册给的公式：

差值=传感器输出在自检模式下测量的值-传感器输出在非自检模式下的值

通过自检代表什么？不通过代表什么？通过代表数据正常且可用，通不过就说明 MPU6050 可能是损坏了。在位分布表的最后部分 FS_SEL 设定了陀螺仪的测量范围：

FS_SEL	Full Scale Range
0	± 250 °/s
1	± 500 °/s
2	± 1000 °/s
3	± 2000 °/s

这样就可以根据上面推算 CONFIG 赋值的方式来给该寄存器赋值了。注释中的默认值 0x18 可以得出是三轴均不自检，精度范围为±2000。

ACCEL_CONFIG　8 位无符号寄存器，配置和解读方式等同于上述陀螺仪寄存器配置。给出位分配表

Type: Read/Write

Register (Hex)	Register (Decimal)	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
1C	28	XA_ST	YA_ST	ZA_ST	AFS_SEL[1:0]		-		

AFS_SEL	Full Scale Range
0	± 2*g*
1	± 4*g*
2	± 8*g*
3	± 16*g*

最后一个介绍的配置寄存器是
PWR_MGMT_1　8 为无符号寄存器，电源管理寄存器。该寄存器允许用户配置电源模式和时钟源。它也提供了一个位来重置整个设备（Reset），还留有一个位来禁用温度传感器。位分配表如下：

Type: Read/Write

Register (Hex)	Register (Decimal)	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
6B	107	DEVICE _RESET	SLEEP	CYCLE	-	TEMP_DIS	CLKSEL[2:0]		

通过将 SLEEP 位置 1 可以将 MPU6050 设定为低功耗睡眠模式. 当睡眠模式为 0 即睡眠模式不启用时将 CYCLE 位置 1 可以开启循环模式，该模式处于睡眠模式和唤醒模式之间，它按照 LP_WAKE_CTRL（该寄存器我们不涉及，有兴趣的的同学可以自行翻阅数据手册）寄存器中规定的速率从加速度计获取单一的简单的数据 。 我 们 Single_Write_IIC(SLAVEADRESS,PWR_MGMT_1,0x00); 通 过 向 PWR_MGMT_1 该寄存器写入 0x00,不启用循环模式和睡眠模式，就进入了唤醒模式，在设置完前面的参数以后，6050 会自动进入睡眠模式，一定要记得唤醒！否则会出现获取到的数据全为 0 的情况。CLKSEL 配置时钟源：

CLKSEL	Clock Source
0	Internal 8MHz oscillator
1	PLL with X axis gyroscope reference
2	PLL with Y axis gyroscope reference
3	PLL with Z axis gyroscope reference
4	PLL with external 32.768kHz reference
5	PLL with external 19.2MHz reference
6	Reserved
7	Stops the clock and keeps the timing generator in reset

上述为配置寄存器，属性均为可读可写。接下来介绍我们获取数据的寄存器

```
#define ACCEL_XOUT_H  0x3B
#define ACCEL_XOUT_L  0x3C
#define ACCEL_YOUT_H  0x3D
#define ACCEL_YOUT_L  0x3E
#define ACCEL_ZOUT_H  0x3F
#define ACCEL_ZOUT_L  0x40
#define TEMP_OUT_H    0x41
#define TEMP_OUT_L    0x42
#define GYRO_XOUT_H   0x43
#define GYRO_XOUT_L   0x44
#define GYRO_YOUT_H   0x45
#define GYRO_YOUT_L   0x46
#define GYRO_ZOUT_H   0x47
#define GYRO_ZOUT_L   0x48
```

ACCEL_XOUT_H 代表加速度计的高八位，ACCEL_XOUT_L 代表加速度计的低八位。因为只有 8 位寄存器，所以需要两个寄存器合并为 16 位的数据。
以此类推，后面的分别为 Y 轴，Z 轴的加速度，温度，X 轴，Y 轴，Z 轴的角速度值。
怎样合并高低位的数据？

```
GyroYH = Single_Read_IIC( SLAVEADRESS , GYRO_YOUT_H );
GyroYL = Single_Read_IIC( SLAVEADRESS , GYRO_YOUT_L );
```

通过移位操作，左移八位使其数据转移到高八位，低八位全为 0，再做‘或’运算。

```
GyroY = (GyroYH<<8)|GyroYL;
```

陀螺仪的寄存器操作和认识到此结束。但我们获取到的值称为原始值，是没有办法直接用的，需要经过一定的计算才能得到真实的角度值。

AFS_SEL	Full Scale Range	LSB Sensitivity
0	±2*g*	16384 LSB/*g*
1	±4*g*	8192 LSB/*g*
2	±8*g*	4096 LSB/*g*
3	±16*g*	2048 LSB/*g*

可以看到当测量范围为±2g 时，精度是 16384 LSB/g，±2g 其实一共是 4g 的测量范围，输出数据是 16 位有符号数据，即-32768～32767，32768/2=16384，每 16384 个数代表 1g。所以计算方式为：

加速度=读取到的值/16384

其他测量范围以此类推。

FS_SEL	Full Scale Range	LSB Sensitivity
0	± 250 °/s	131 LSB/°/s
1	± 500 °/s	65.5 LSB/°/s
2	± 1000 °/s	32.8 LSB/°/s
3	± 2000 °/s	16.4 LSB/°/s

测量范围为±250° 时，32768/250=131，每 131 的数代表 1° ，所以计算方式为：

角速度=读取到的值/131

其他的测量范围以此类推。
摄氏温度的计算方式为

摄氏度=测量值/340+36.5

由此得到的角速度值和加速度值并不准确，因为陀螺仪内部的机械结构使得三个轴的数据相互影响。精确姿态解算需要复杂的运算和卡尔曼滤波，感兴趣的同学可以自行学习。