

	InvenSense Inc. 1745 Technology Drive, 圣何塞, CA, 95110 美国 电话: +1 (408) 501-2200 传真: +1 (408) 988-7339 网站: www.invensense.com	文件编号: AN-XX-XXXX-XX 修订: 1.0 发布日期: 2014 年 2 月 21 日
---	--	---

MPU 硬件偏移寄存器 应用说明

本文件的打印副本不受修订控制

除非它注明日期并用红墨水盖章，
“修订控制副本。”

InvenSense 提供的这些信息被认为是准确和可靠的。但是，InvenSense 对其使用或因使用可能导致的任何侵犯第三方专利或其他权利的行为不承担任何责任。规格如有更改，恕不另行通知。InvenSense 保留更改本产品（包括其电路和软件）以改进其设计和/或性能的权利，恕不另行通知。InvenSense 对本文档中包含的信息和规格不作任何明示或暗示的保证。对于因本文档中包含的信息或使用其中所述的产品和服务而引起的任何索赔或损害，InvenSense 不承担任何责任。这包括但不限于基于侵犯专利、版权、掩模作品和/或其他知识产权的索赔或损害赔偿。

InvenSense 拥有并在本文件中描述的某些知识产权受专利保护。InvenSense 的任何专利或专利权均未通过暗示或其他方式授予许可。本出版物取代并取代之前提供的所有信息。作为注册商标的商标是其各自公司的财产。InvenSense 传感器不得用于或销售任何常规或大规模破坏性武器的开发、存储、生产或利用，或用于任何其他武器或威胁生命的应用，以及任何其他生命攸关的应用，例如医疗设备、运输、航天与核仪器、海底设备、电厂设备、防灾防非设备。

版权所有 ©2011 InvenSense Corporation。

	MPU 硬件偏移寄存器应用笔记	版本号 :1.0 发布日期 :2014 年 2 月 21 日
---	-----------------	-----------------------------------

目录_

1. 修订历史	3
2. 参考	3
3. 目的.....	3
4. 数据信号图	3
5. 找出加速度和陀螺仪的偏移偏差.....	4
5.1 来自运动驱动程序5.1.2的示例代码.....	4
6. 陀螺偏移寄存器.....	5
6.1 注册位置.....	5
6.2 寄存器格式和说明.....	5
6.3 来自运动驱动程序5.1.2的示例代码.....	6
7. 加速度偏移寄存器.....	6
7.1 注册地点.....	6
7.2 寄存器格式和说明.....	7
7.3 来自运动驱动程序5.1.2的示例代码.....	7
8. 杂项	8

	MPU 硬件偏移寄存器应用笔记	版本号 :1.0 发布日期 :2014 年 2 月 21 日
---	-----------------	-----------------------------------

1. 修订历史

修订日期 修订		描述
2014 年 2 月 21 日	1.0	已创建文档
2014 年 6 月 27 日	1.1	修改 Set 6500 Accel Bias 功能
2014 年 7 月 28 日	1.2	加速度偏置的修改比例范围从 +-8G 到 +-16G

2. 参考

请查看相关部件的寄存器图、EVB 和生产规范。
<http://www.invensense.com/mems/gyro/catalog.html>

3. 目的

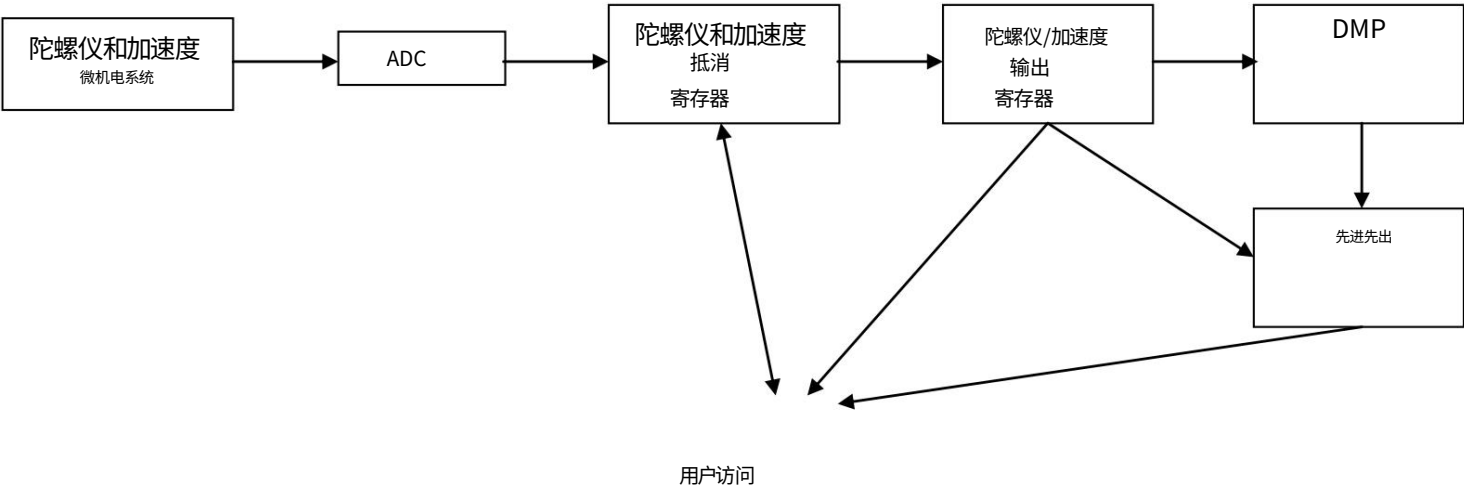
强烈建议所有 MEMS 传感器在工厂校准或在使用时动态校准以获得更准确的数据。Invensense 软件和 DMP 有几种处理校准的算法。但是,如果您不使用这些算法,仍然可以使用 MPU 设备中的内置硬件偏移寄存器来实现偏移消除。

一些 Invensense MPU 和 ICM 器件包含用于加速度计和陀螺仪传感器的偏移消除寄存器。本文档详细介绍了这些寄存器,一个关于如何获得陀螺仪和加速度偏移量的示例,以及如何将这些偏移量应用到这些寄存器中。

已针对本文档进行测试和确认的 MPU 设备是 MPU6050、MPU9150、MPU6500、MPU6515 和 MPU9250。

4. 数据信号图

这些偏移寄存器的工作原理是,来自 MEMS 传感器的所有数据在输出到数据寄存器供用户读取之前都会应用这些偏移。偏移量应用程序也在所有与 FIFO 和 DMP 相关的使用之前。因此,FIFO 中的数据、输出寄存器和 DMP 中使用的数据已经包含了这些偏移量。





5. 找到 Accel 和 Gyro 的偏移偏差

有多种方法可以获得 Accel 和 Gyro 的偏差,但总体思路是设备在已知方向上是静止的,通常 MPU 部分朝上或朝下。

因为设备是静止的,所以每个轴的预期陀螺输出为 0, 0, 0,加速度输出为 0, 0, +1G。有了这些信息,如果我们对每个轴进行采样,我们将能够确定与理想值的平均偏移,这些值将是偏移偏差。

5.1 Motion Driver 5.1.2 的示例代码

```
静态 int get_biases(long *gyro, long *accel) {

    无符号字符数据[MAX_PACKET_LENGTH]; unsigned
    char packet_count, ii;无符号短fifo_count;

    数据[0] = 0x01;数据[1]
    = 0; if (i2c_write(st.hw-
    >addr, st.reg->pwr_mgmt_1, 2, data))返回-1;延迟毫秒 (200);数据[0] = 0; if
    (i2c_write(st.hw->addr, st.reg->int_enable, 1, data))返回-1; if
    (i2c_write(st.hw->addr, st.reg->fifo_en, 1, data))返回-1; if (i2c_write(st.hw-
    >addr, st.reg->pwr_mgmt_1, 1, data))返回-1; if (i2c_write(st.hw->addr, st.reg-
    >i2c_mst, 1, data))返回-1; if (i2c_write(st.hw->addr, st.reg->user_ctrl, 1, data))返
    回-1;数据[0] = BIT_FIFO_RST | BIT_DMP_RST; if (i2c_write(st.hw->addr,
    st.reg->user_ctrl, 1, data))返回-1;延迟毫秒 (15);数据[0] = st.test->reg_lpf; if
    (i2c_write(st.hw->addr, st.reg->lpf, 1, data))

    返回-1;
    数据[0] = st.test->reg_rate_div; if
    (i2c_write(st.hw->addr, st.reg->rate_div, 1, data))返回-1;数据[0] = st.test-
    >reg_gyro_fsr; if (i2c_write(st.hw->addr, st.reg->gyro_cfg, 1, data))返
    回-1;

    if (hw_test) 数据
    [0] = st.test->reg_accel_fsr | 0xE0;否则数据[0] =
    test.reg_accel_fsr;

    if (i2c_write(st.hw->addr, st.reg->accel_cfg, 1, data))返回-1; if (hw_test)
    delay_ms(200);

    /* 为 test.wait_ms 毫秒填充 FIFO。 */数据[0] = BIT_FIFO_EN; if
    (i2c_write(st.hw->addr, st.reg->user_ctrl, 1, data))返回-1;

    数据[0] = INV_XYZ_GYRO | INV_XYZ_ACCEL; if
    (i2c_write(st.hw->addr, st.reg->fifo_en, 1, data))返回-1;
    delay_ms(test.wait_ms);数据[0] = 0; if (i2c_write(st.hw->addr, st.reg-
    >fifo_en, 1, data))返回-1;

    if (i2c_read(st.hw->addr, st.reg->fifo_count_h, 2, data))返回-1;
```

```
fifo_count = (数据[0] << 8) | 数据[1];
packet_count = fifo_count / MAX_PACKET_LENGTH;
陀螺仪 [0] = 陀螺仪 [1] = 陀螺仪 [2] = 0;
加速度[0] = 加速度[1] = 加速度[2] = 0;

对于(ii = 0; ii < packet_count; ii++) {
    短加速曲线[3],陀螺曲线[3];
    如果 (i2c_read (st.hw->addr, st.reg->fifo_r_w, MAX_PACKET_LENGTH,数据) )
        返回-1;
    accel_cur[0] = ((short)data[0] << 8) | 数据[1];
    accel_cur[1] = ((short)data[2] << 8) | 数据[3];
    accel_cur[2] = ((short)data[4] << 8) | 数据[5];
    accel[0] += (long)accel_cur[0];
    accel[1] += (long)accel_cur[1];
    accel[2] += (long)accel_cur[2];
    gyro_cur[0] = (((short)data[6] << 8) | data[7]);
    gyro_cur[1] = (((short)data[8] << 8) | data[9]);
    gyro_cur[2] = (((short)data[10] << 8) | data[11]);
    陀螺仪 [0] += (长) gyro_cur [0];
    陀螺仪 [1] += (长) gyro_cur [1];
    陀螺仪 [2] += (长) gyro_cur [2];
}
gyro[0] = (long)(((long long)gyro[0]) / test.gyro_sens / packet_count);
gyro[1] = (long)(((long long)gyro[1]) / test.gyro_sens / packet_count);
gyro[2] = (long)(((long long)gyro[2]) / test.gyro_sens / packet_count);
accel[0] = (long)(((long long)accel[0]) / test.accel_sens /
    数据包计数) ;
accel[1] = (long)(((long long)accel[1]) / test.accel_sens /
    数据包计数) ;
accel[2] = (long)(((long long)accel[2]) / test.accel_sens /
    数据包计数) ;
/* 移除加速度 Z 的重力 */
如果 (加速度 [2] > 0L)
    加速度[2] -= test.accel_sens;
别的
    加速[2] += test.accel_sens;

    返回0;
}
```

6. 陀螺偏移寄存器

6.1 寄存器位置

对于 MPU6050/MPU6500/MPU6515 和 MPU9150/MPU9250,陀螺仪偏移寄存器位于

地址	名称	描述
0x13	XG_OFFS_USRH	陀螺仪X轴偏移抵消寄存器高字节
0x14	XG_OFFS_USRL	陀螺 X 轴偏移消除寄存器低字节
0x15	YG_OFFS_USRH	陀螺仪Y轴偏移抵消寄存器高字节
0x16	YG_OFFS_USRL	陀螺 Y 轴偏移消除寄存器低字节
0x17	ZG_OFFS_USRH	陀螺 Z 轴偏移消除寄存器高字节
0x18	ZG_OFFS_USRL	陀螺 Z 轴偏移消除寄存器低字节

6.2 寄存器格式及说明

Gyro 寄存器在启动时的默认值为 0.输入的偏置值需要在 +-1000dps 灵敏度范围内。这意味着每 1 dps = 32.8 LSB

6.3 Motion Driver 5.1.2 的示例代码

```
/**
 * @brief 此 将偏置推送到陀螺偏置 6500/6050 寄存器。
 * 函数需要相对于当前传感器输出的偏差,并且
 * 这些偏差将被添加到工厂提供的值中。偏置输入为 LSB
 * +-1000dps 格式。
 * @param[in] gyro_bias 新的偏差。
 * @return 0 如果成功。
 */
int mpu_set_gyro_bias_reg(long *gyro_bias)
{
    无符号字符数据[6] = {0, 0, 0, 0, 0, 0};
    注释i = 0;
    for(i=0;i<3;i++) {
        gyro_bias[i] = (-gyro_bias[i]);
    }
    数据[0] = (gyro_bias[0] >> 8) & 0xff;
    数据[1] = (gyro_bias[0]) & 0xff;
    数据[2] = (gyro_bias[1] >> 8) & 0xff;
    数据[3] = (gyro_bias[1]) & 0xff;
    数据[4] = (gyro_bias[2] >> 8) & 0xff;
    数据[5] = (gyro_bias[2]) & 0xff;
    if (i2c_write(st.hw->addr, 0x13, 2, &data[0]))
        返回-1;
    if (i2c_write(st.hw->addr, 0x15, 2, &data[2]))
        返回-1;
    if (i2c_write(st.hw->addr, 0x17, 2, &data[4]))
        返回-1;
    返回0;
}
```

7. 加速偏移寄存器

7.1 注册位置

对于 MPU6050/MPU9150,寄存器位于以下地址

地址	名称	描述
0x06	XG_OFFS_USRH	Accel X 轴偏移消除寄存器高字节
0x07	XG_OFFS_USRL	Accel X 轴偏移消除寄存器低字节
0x08	YG_OFFS_USRH	Accel Y 轴偏移消除寄存器高字节
0x09	YG_OFFS_USRL	Accel Y 轴偏移消除寄存器低字节
0x0A	ZG_OFFS_USRH	Accel Z 轴偏移消除寄存器高字节
0x0B	ZG_OFFS_USRL	Accel Z 轴偏移消除寄存器低字节

MPU6500/MPU6515/和MPU9250位于以下位置

地址	名称	描述
0x77	XG_OFFS_USRH	Accel X 轴偏移消除寄存器高字节
0x78	XG_OFFS_USRL	Accel X 轴偏移消除寄存器低字节
0x7A	YG_OFFS_USRH	Accel Y 轴偏移消除寄存器高字节
0x7B	YG_OFFS_USRL	Accel Y 轴偏移消除寄存器低字节
0x7D	ZG_OFFS_USRH	Accel Z 轴偏移消除寄存器高字节
0x0E	ZG_OFFS_USRL	Accel Z 轴偏移消除寄存器低字节



7.2 寄存器格式及说明

与陀螺仪不同,加速度偏移寄存器的使用并不直接。

1. 初始值包含 Accel 出厂微调的 OTP 值。因此,在启动时,这些寄存器中会有一个非零值。用户需要首先读取寄存器并将偏差应用于该值。
2. 格式为+-16G 其中1mg = 2048 LSB
3. 各轴低字节位0 为保留位,需保留。

7.3 Motion Driver 5.1.2 的示例代码

```
/**
 * @brief 读取加速度偏置 6500 寄存器的偏置。
 * 此函数从 MPU6500 加速度偏移寄存器中读取。
 * 格式为 +-8G 格式的 G。该寄存器使用 OTP * 出厂调整值进行初始化。 * @param[in] accel_bias 返回具有加速度偏置的结构
 * @return */ int mpu_read_6500_accel_bias(long *accel_bias) {
    // 0 如果成功。

    // 无符号字符数据[6]; if
    (i2c_read(st.hw->addr, 0x77, 2, &data[0]))返回-1; if (i2c_read(st.hw->addr, 0x7A, 2, &data[2]))

    返回-1; if
    (i2c_read(st.hw->addr, 0x7D, 2, &data[4]))返回-1; accel_bias[0] =
    ((long)data[0]<<8) 数据[1]; accel_bias[1] =
    ((long)data[2]<<8) 数据[3]; accel_bias[2] = ((long)data[4]<<8) 数据[5];
    返回0;

}

/**
 * @brief 将偏置推送到加速偏置 6500 寄存器。
 * 此函数需要相对于当前传感器输出的偏置,并且 * 这些偏置将添加到工厂提供的值中,偏置输入为 +-8G 格式的 LSB *。
 * @param[in] accel_bias 新的偏置。 * @return 0 如果成功。 */ int mpu_set_accel_bias_6500_reg(const long *accel_bias) { unsigned
char data[6] = {0, 0, 0, 0, 0, 0}; long accel_reg_bias[3] = {0, 0, 0};

    如果 (mpu_read_6500_accel_bias (accel_reg_bias))返回-1;


    // 保留出厂值的第 0 位 (用于温度补偿) accel_reg_bias[0] := (accel_bias[0] & ~1); accel_reg_bias[1] :=
(accel_bias[1] & ~1); accel_reg_bias[2] := (accel_bias[2] & ~1);

    数据[0] = (accel_reg_bias[0] >> 8) & 0xff;数据[1] = (accel_reg_bias[0])
& 0xff;数据[2] = (accel_reg_bias[1] >> 8) & 0xff;数据[3] =
(accel_reg_bias[1]) & 0xff;数据[4] = (accel_reg_bias[2] >> 8) & 0xff;
    数据[5] = (accel_reg_bias[2]) & 0xff;

    if (i2c_write(st.hw->addr, 0x77, 2, &data[0]))返回-1; if (i2c_write(st.hw->addr, 0x7A, 2, &data[2]))返回-1; if (i2c_write(st.hw->addr, 0x7D,
2, &data[4]))返回-1;

    返回0;

}
```

	MPU 硬件偏移寄存器应用笔记	版本号:1.0 发布日期:2014 年 2 月 21 日
---	-----------------	---------------------------------

8. 杂项

偏置偏移一旦输入到偏移消除寄存器中,将立即生效。但是,在关闭和打开电源后,需要将这些值重新加载回这些寄存器中。建议保存偏差值,以便在启动后轻松重新应用。