挑战点四：裸机下驱动gy86

内容概述：

gy86内部集成了三个芯片，分别是mpu6050姿态传感器、ms5611气压传感器、hmc5883l磁力传感器，因此操作gy86即是操作其内部的这三种芯片。

由于gy86与stm32通信使用的是iic协议，因此同样需要掌握iic通信。

所以想要实现裸机下驱动gy86，需要掌握四点：iic通信、驱动mpu6050、驱动ms5611、驱动hmc5883l

iic通信

I2C通信简介

- I2C总线（Inter IC BUS）是由Philips公司开发的一种通用数据总线
- 两根通信线：SCL（Serial Clock）、SDA（Serial Data）
- 同步，半双工
- 带数据应答
- 支持总线挂载多设备（一主多从、多主多从）

I2C初始化

```
#include "stm32f4xx.h"                  // Device header
#include "stm32f4xx_gpio.h"
#include "stm32f4xx_rcc.h"
#include "DELAY1.h"
void myiic_w_scl(uint8_t value)//PC2 SCL
{
    GPIO_WriteBit(GPIOC,GPIO_Pin_2,(BitAction)value);
    Delay_us(10);
}
void myiic_w_sda(uint8_t value)//PC3 SDA
{
    GPIO_WriteBit(GPIOC,GPIO_Pin_3,(BitAction)value);
    Delay_us(10);
}
uint8_t myiic_r_sda()
{
```

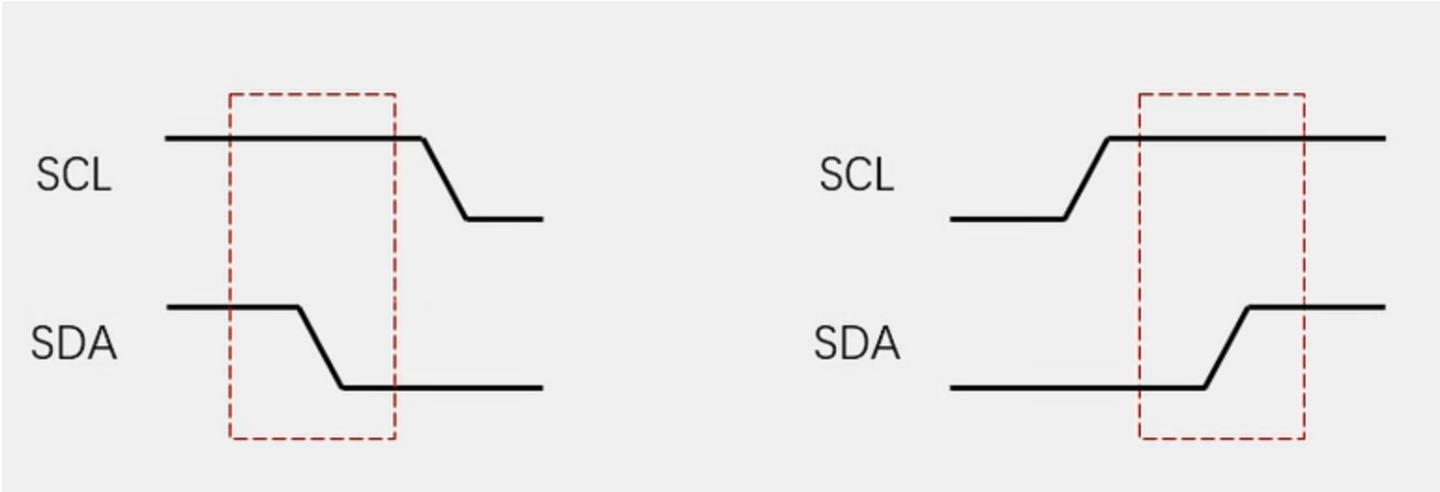
```
    uint8_t value;
    value=GPIO_ReadInputDataBit(GPIOC,GPIO_Pin_3);
    Delay_us(10);
    return value;
}
void myiic_init()
{
    RCC_AHB1PeriphClockCmd(RCC_AHB1Periph_GPIOC,ENABLE);//使能GPIOC

    GPIO_InitTypeDef gpio_initstructure;//GPIO初始化结构体
    gpio_initstructure.GPIO_Mode=GPIO_Mode_OUT;//输出模式
    gpio_initstructure.GPIO_OType=GPIO_O Type_OD;//开漏模式
    gpio_initstructure.GPIO_Pin=GPIO_Pin_2|GPIO_Pin_3;//选择PC2与PC3
    gpio_initstructure.GPIO_PuPd=GPIO_PuPd_NOPULL;//浮空模式
    gpio_initstructure.GPIO_Speed=GPIO_High_Speed;//高速，100MHZ
    GPIO_Init(GPIOC,&gpio_initstructure);

    GPIO_SetBits(GPIOC,GPIO_Pin_2|GPIO_Pin_3);//SCL,SDA置高电平
}
```

I2C基本时序单元

- 起止条件：SCL高电平期间，SDA从高电平切换到低电平
- 终止条件：SCL高电平期间，SDA从低电平切换到高电平




```
void myiic_start()
{
    myiic_w_sda(1);//释放SDA
    myiic_w_scl(1);//释放SCL，确保SCL和SDA都释放
    myiic_w_sda(0);//先拉低SDA
    myiic_w_scl(0);//再拉低SCL，产生起始条件
}
void myiic_stop()
```

```
{
    myiic_w_sda(0);//拉低SDA，确保SDA之后能产生上升沿
    myiic_w_scl(1);//先释放SCL
    myiic_w_sda(1);//再释放SDA，产生终止条件
}
```

- 发送一个字节：SCL低电平期间，主机将数据位依次放到SDA线上（高位先行），然后释放SCL。从机将在SCL高电平期间读取数据位，所以SCL高电平期间SDA不允许有数据变化。循环8次即可发送一个字节。

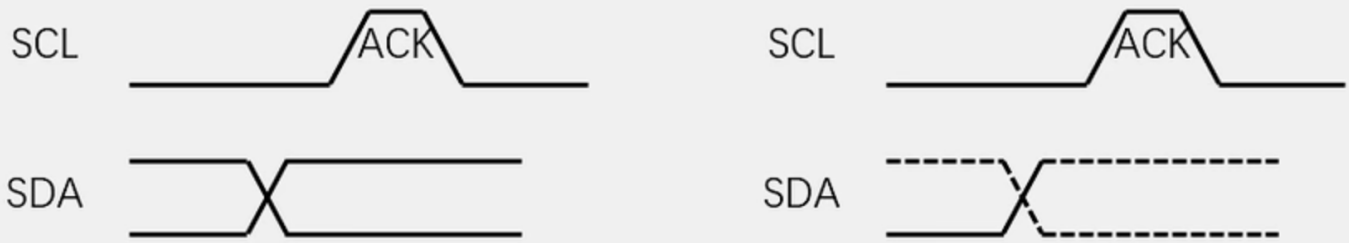

```
void myiic_sendbyte(uint8_t byte)
{
    int i;
    for(i=0;i<8;i++)                //循环8次，即可发送一个字节
    {
        myiic_w_sda(byte&(0x80>>i));//向SDA写入数据位，高位先行
        myiic_w_scl(1);//释放SCL，高电平期间读取数据位
        myiic_w_scl(0);//拉低SCL
    }
}
```

- 接受一个字节：SCL低电平期间，从机将数据位依次放到SDA线上（高位先行），然后释放SCL，主机将在SCL高电平期间读取数据位，所以SCL高电平期间SDA不允许有数据变化（主机在接受之前，需要释放SDA）。

```c
uint8_t myiic_receivebyte()
{
    int i;
    uint8_t byte=0x00;
    myiic_w_sda(1);//主机释放SDA，从机把数据放到SDA

    for(i=0;i<8;i++)
    {
        myiic_w_scl(1);//SCL高电平，主机读取数据
        if(myiic_r_sda()==1)
    {
        byte|=(0x80>>i);//将SDA上读取的数据按位存放在变量中
    }
        myiic_w_scl(0);//拉低SCL，从机把下一位数据放到SDA上
    }
    return byte;
}
```

- 发送应答：主机在接受完一个字节之后，在下一个时钟发送一位数据，数据0表示应答，数据1表示非应答
- 接受应答：主机在发送完一个字节之后，在下一个时钟接受一位数据，判断从机是否应答，数据0表示应答，数据1表示非应答（主机在接受之前，需要释放SDA）。




```c
void myiic_sendack(uint8_t byte)
{
    myiic_w_sda(byte);//进入函数时SCL为低电平，主机把应答位放在SDA上
    myiic_w_scl(1);//SCL高电平，从机读取应答
    myiic_w_scl(0);//SCL低电平，进入下一个时序单元
}
uint8_t myiic_receiveack(void)
{
    uint8_t byte;
    myiic_w_sda(1);//主机释放SDA，防止干扰从机，从机把应答位放在SDA上

    myiic_w_scl(1);//SCL高电平，主机读取应答位
    byte=myiic_r_sda();//将读到的数据赋值给变量
    myiic_w_scl(0);//SCL应答位，进入到下一个时序单元
```

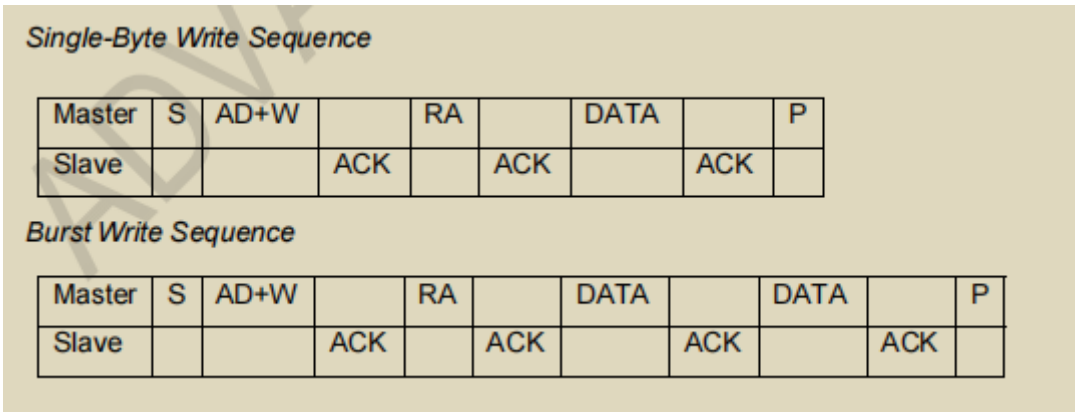
```
15
16      return byte;
17  }
```

组合操作

IIC地址有7位地址和8位地址，8位地址实际是7位地址左移一位，最后一位为读写位，0代表写入数据，1代表读入数据。MPU6050的7位地址为0x68,实际读数据地址为 0x68<<1，写数据地址为 0x68<<1 | 0x01。又如MS5611的7位地址为0x77，8位写地址位 0x77<<1=0xEE, 8位读地址位 0x77<<1 | 0x01 = 0xEF。

- 指定地址写

对于指定设备（Slave Address），在指定地址（Reg Address）下，写入指定数据（Data）。




```
1  void myiic_writereg(uint8_t addr,uint8_t reg,uint8_t data)
2  {
3      myiic_start();
4      myiic_sendbyte(addr);
5      myiic_receiveack();
6      myiic_sendbyte(reg);
7      myiic_receiveack();
8
9      myiic_sendbyte(data);
10     myiic_receiveack();
11     myiic_stop();
12
13 }
```

- 指定地址读

对于指定设备（Slave Address），在指定地址（Reg Address）下，读取指定数据（Data）。

Single-Byte Read Sequence

Master	S	AD+W		RA		S	AD+R			NACK	P
Slave			ACK		ACK			ACK	DATA		

Burst Read Sequence

Master	S	AD+W		RA		S	AD+R			ACK		NACK	P
Slave			ACK		ACK			ACK	DATA		DATA		

```
uint8_t mpu6050_readreg(uint8_t reg)
{
    uint8_t data;
    myiic_start();
    myiic_sendbyte(mpu6050_address);
    myiic_receiveack();
    myiic_sendbyte(reg);
    myiic_receiveack();

    myiic_start();
    myiic_sendbyte(mpu6050_address|0x01);
    myiic_receiveack();

    data=myiic_receivebyte();
    myiic_sendack(1);
    Delay_us(10);
    myiic_stop();

    return data;
}
```

驱动mpu6050

mpu6050简介

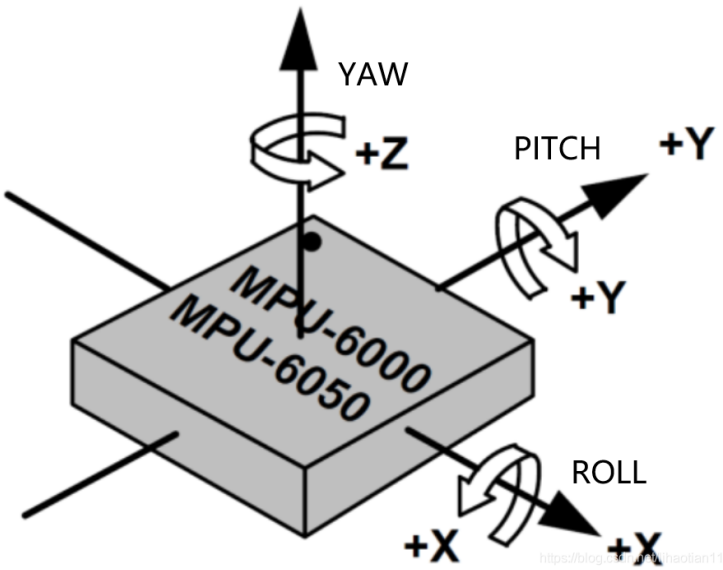
MPU6050内部整合了**三轴陀螺仪、三轴加速度计**，而且还可以连接一个第三方数字传感器(如磁力计)，这样的话，就可以通过IIC接口输出一个9轴信号。陀螺仪的数据和加速度计的数据经过数据融合（互补滤波、卡尔曼滤波等等）才可以得到正确的角速度、加速度。

MPU6050三轴角(姿态角)

PITCH:俯仰角，物体绕x轴旋转

YAW:航向角，物体绕z轴旋转

ROLL:横滚角，物体绕y轴旋转



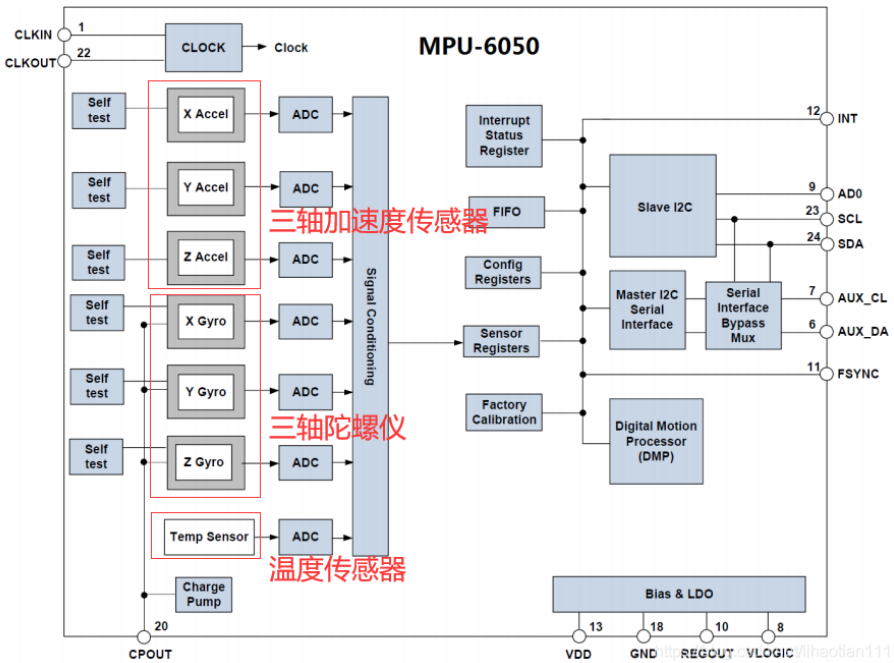
mpu6050相关参数

1.mpu6050内部装有16位ADC采集传感器，范围-32768~32767

2.可配置数字低通滤波器

3.可配置时钟源

4.可配置采样分频



mpu6050读写时序

Single-Byte Write Sequence

Master	S	AD+W		RA		DATA		P
Slave			ACK		ACK		ACK	

Burst Write Sequence

Master	S	AD+W		RA		DATA		DATA		P
Slave			ACK		ACK		ACK		ACK	

Single-Byte Read Sequence

Master	S	AD+W		RA		S	AD+R			NACK	P
Slave			ACK		ACK			ACK	DATA		

Burst Read Sequence

Master	S	AD+W		RA		S	AD+R			ACK		NACK	P
Slave			ACK		ACK			ACK	DATA		DATA		

9.4 I^2C Terms

Signal	Description
S	Start Condition: SDA goes from high to low while SCL is high
AD	Slave I^2C address
W	Write bit (0)
R	Read bit (1)
ACK	Acknowledge: SDA line is low while the SCL line is high at the 9^{th} clock cycle
NACK	Not-Acknowledge: SDA line stays high at the 9^{th} clock cycle
RA	MPU-60X0 internal register address
DATA	Transmit or received data
P	Stop condition: SDA going from low to high while SCL is high

几个比较重要的寄存器

◆ 电源管理寄存器1（0X6B）

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

DEVICE_RESE=1，复位MPU6050，复位完成后，自动清零。SLEEP=1，进入睡眠模式；SLEEP=0，正常工作模式。TEMP_DIS，用于设置是否使能温度传感器，设置为0，则使能CLKSEL[2:0]，用于选择系统时钟源，如下所示：

CLKSEL[2:0]	时钟源
000	内部8M RC晶振
001	PLL，使用X轴陀螺作为参考
010	PLL，使用Y轴陀螺作为参考
011	PLL，使用Z轴陀螺作为参考
100	PLL，使用外部32.768Khz作为参考
101	PLL，使用外部19.2Mhz作为参考
110	保留
111	关闭时钟，保持时序产生电路复位状态

电源管理寄存器最主要的作用就是复位MPU6050

◆ 陀螺仪配置寄存器（0X1B）

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

位7、6、5用于陀螺仪自检，位4、3用于设置陀螺仪的满量程范围：0，±250°/s；1，±500°/s；2，±1000°/s；3，±2000°/s；我们一般设置为3，即±2000°/S，因为陀螺仪的ADC为16位分辨率，所以得到灵敏度为：65536/4000=16.4LSB/(°/S)。

陀螺仪配置寄存器配置陀螺仪满量程范围

◆ 加速度传感器配置寄存器（0X1C）

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

位7、6、5用于加速度计自检，AFS_SEL[1:0]这两个位用于设置加速度传感器的满量程范围：0，±2g；1，±4g；2，±8g；3，±16g；我们一般设置为0，即±2g，因为加速度传感器的ADC也是16位，所以得到灵敏度为：65536/4=16384LSB/g。

加速度传感器配置寄存器配置加速度传感器满量程范围，不宜过大

◆ 配置寄存器（0X1A）

设置：带宽=1/2采样率

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

重点看数字低通滤波器（DLPF）的设置位，即：DLPF_CFG[2:0]，加速度计和陀螺仪，都是根据这三个位的配置进行过滤的，如下表：

DLPF_CFG[2:0]	加速度传感器 Fs=1Khz		角速度传感器（陀螺仪）		
	带宽(Hz)	延迟（ms）	带宽(Hz)	延迟（ms）	Fs(Khz)
000	260	0	256	0.98	8
001	184	2.0	188	1.9	1
010	94	3.0	98	2.8	1
011	44	4.9	42	4.8	1
100	21	8.5	20	8.3	1
101	10	13.8	10	13.4	1
110	5	19.0	5	18.6	1
111	保留		保留		8

配置寄存器就是设置数字低通滤波器的DLPF_CFG位来结合陀螺仪采样分频寄存器来共同设置采样周期

◆ 电源管理寄存器2（0X6C）

Register (Hex)	Register (Decimal)	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
6C	108	LP_WAKE_CTRL[1:0]		STBY_XA	STBY_YA	STBY_ZA	STBY_XG	STBY_YG	STBY_ZG

后面六位，分别控制加速度和陀螺仪的x/y/z轴是否进入待机模式，这里我们全部都不进入待机模式，所以全部设置为：0，即可。

总之，电源管理寄存器2就是用于设置加速度传感器和陀螺仪的X/Y/Z轴是进入休眠还是正常工作

◆ 加速度传感器数据输出寄存器（0X3B~0X40）

Register (Hex)	Register (Decimal)	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
3B	59	ACCEL_XOUT[15:8]							
3C	60	ACCEL_XOUT[7:0]							
3D	61	ACCEL_YOUT[15:8]							
3E	62	ACCEL_YOUT[7:0]							
3F	63	ACCEL_ZOUT[15:8]							
40	64	ACCEL_ZOUT[7:0]							

加速度传感器数据输出寄存器总共由6个寄存器组成，输出X/Y/Z三个轴的加速度传感器值，高字节在前，低字节在后。

总之，加速度传感器数据输出寄存器就是把加速度传感器测量到的数据输出出来。

◆ 陀螺仪数据输出寄存器（0X43~0X48）

Register (Hex)	Register (Decimal)	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
43	67	GYRO_XOUT[15:8]							
44	68	GYRO_XOUT[7:0]							
45	69	GYRO_YOUT[15:8]							
46	70	GYRO_YOUT[7:0]							
47	71	GYRO_ZOUT[15:8]							
48	72	GYRO_ZOUT[7:0]							

陀螺仪数据输出寄存器总共由6个寄存器组成，输出X/Y/Z三个轴的陀螺仪传感器数据，高字节在前，低字节在后。

总之，陀螺仪数据输出寄存器就是把陀螺仪测量到的数据输出出来。

代码部分

```
    AREA GY86, CODE, READONLY
    
    EXPORT MPU6050_WRITE
MPU6050_WRITE;
    MOV R2, R0
    MOV R3, R1
    BL IIC_START
    LDR R0, =mpu6050_address
    BL IIC_SENDBYTE
    BL IIC_RECIEVEACK
    MOV R0, R2
    BL IIC_SENDBYTE
    BL IIC_RECIEVEACK
    MOV R0, R3
    BL IIC_SENDBYTE
    BL IIC_RECIEVEACK
    BL IIC_STOP
    MOV PC, LR
    EXPORT MPU6050_READ
```

```
    EXPORT MPU6050_READ
MPU6050_READ;参数为reg
    MOV R1,R0;R0=R1
    BL IIC_START
    LDR R0,=mpu6050_address
    BL SENDBYTE
    BL RECIEVEACK
    MOV R0,R1
    BL SENDBYTE
    BL RECIEVEACK
    BL IIC_START
    LDR R0,=0XD1
    BL SENDBYTE
    BL RECIEVEACK
    BL IIC_RECIEVEBYTE;这里接受了一个字节
    MOV R1,R0
    LDR R0,=1
    BL IIC_SENDACK
    LDR R0,=1
    BL Delay_us
    BL IIC_STOP
    MOV R0,R1;将返回值保存在r0中
    MOV PC,LR
    EXPORT MPU6050_INIT
```

```
MPU6050_INIT
    BL IIC_INIT

    LDR R0,=MPU6050_PWR_MGMT_1
    LDR R1,=0X01
    BL MPU6050_WRITE

    LDR R0,=MPU6050_PWR_MGMT_2
    LDR R1,=0x00
    BL MPU6050_WRITE

    LDR R0,=MPU6050_SMPLRT_DIV
    LDR R1,=0x09
    BL MPU6050_WRITE

    LDR R0,=MPU6050_CONFIG
    LDR R1,=0x06
    BL MPU6050_WRITE

    LDR R0,=MPU6050_GYRO_CONFIG
    LDR R1,=0x18
    BL MPU6050_WRITE

    LDR R0,=MPU6050_ACCEL_CONFIG
    LDR R1,=0x18
    BL MPU6050_WRITE
    MOV PC,LR
```

驱动ms5611

ms5611简介

MS5611气压传感器是一款支持SPI和I^2C总线接口的新一代高分辨率气压传感器，分辨率可达到10cm。该传感器模块包括一个高线性度的压力传感器和一个超低功耗的24位Σ模数转换器（工厂校准系数）。**MS5611**提供了一个精确的24位数字压力值和温度值以及不同的操作模式，可以提高转换速度并优化电流消耗。高分辨率的温度输出无须额外传感器可实现高度计/温度计功能。可以与几乎任何微控制器连接。

ms5611相关指令

1.复位

2.读取 PROM（128 位校准字）

3. D1 转换

4.D2 转换

5. 读取 ADC 结果（24 位压力/温度）

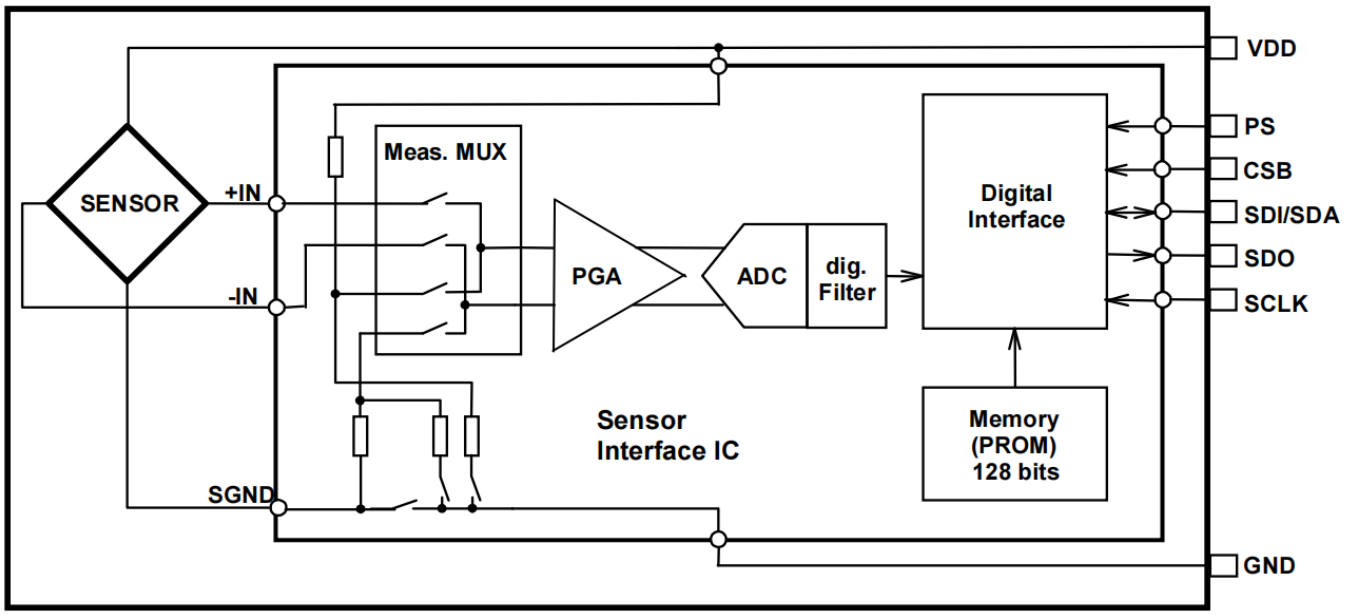


Figure 1: Block diagram of MS5611-01BA

ms5611读写时序

1、复位序列

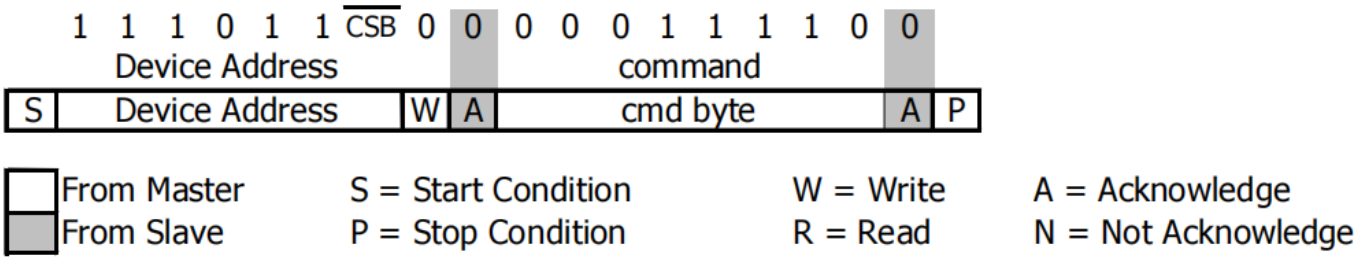


Figure 10: I^2C Reset Command

2、可编程只读存储器（PROM）读取序列

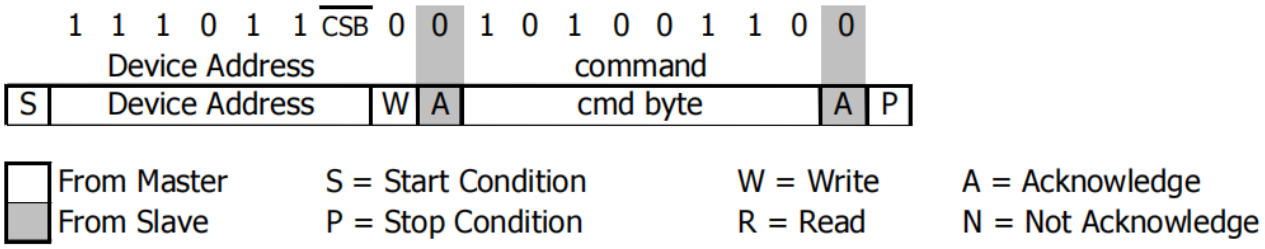


Figure 11: I²C Command to read memory address= 011 (Coefficient 3)

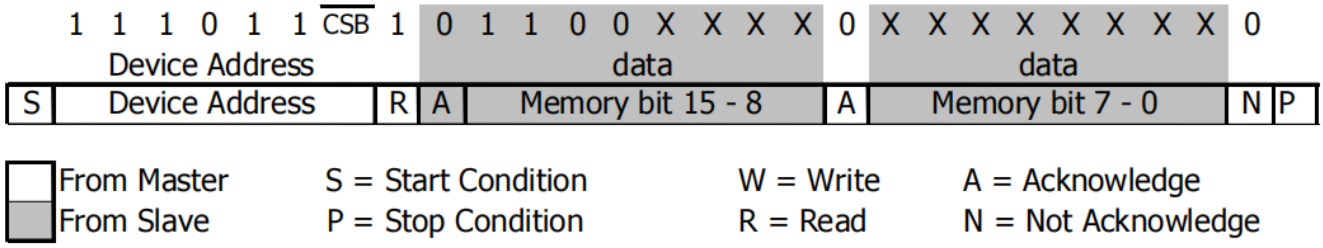


Figure 12: I²C answer from MS5611-01BA

驱动hmc5883l

hmc5883l简介

HMC5883L 是一种表面贴装的高集成模块，主要应用于低成本罗盘和磁场检查领域。HMC5883L磁阻传感器电路采用三轴传感器并应用特殊辅助电路来测量磁场，并通过施加供电电源，传感器可以将量测轴方向上的任何入射磁场转变成一种差分电压输出。

hmc6883l相关参数

- 可配置输出速率（15HZ默认）
- 可配置增益（默认1.3Ga），输出的磁场强度值与实际磁场强度值之间的比例关系。增益值越大，输出的磁场强度值就越大，可以提高磁场测量的灵敏度。
- 可配置转换间隔（7.5ms）
- 可配置输出模式...

重要寄存器

配置寄存器A,配置寄存器B

HMC 5583L

配置寄存器 A

配置寄存器是用来配置该装置设置的数据输出速率和测量配置。 CRA0 通过 CRA7 表明位的位置，用 *CAR* 指示在配置寄存器中的位。 CRA7 指示数据流的第一位。括号中的数目显示是该位的默认值。

CRA7	CRA6	CRA5	CRA4	CRA3	CRA2	CRA1	CRA0
(1)	MA1(1)	MA0(1)	DO2 (1)	DO1 (0)	DO0 (0)	MS1 (0)	MS0 (0)

CRA7 :这个位必须清除以正确运行。

CRA6至 CRA5 :在每次测量输出中选择采样平均数（1-8） 00=1; 01=2;10=4; 11=8(缺省)

CRA4 至 CRA2:数据输出速率位。这些位设置数据写入所有三个数据输出寄存器的速度。

CRA1 至 CRA0 :测量配置位。这些位定义装置的测量流程，特别是是否纳入适用的偏置到测量中去。

HMC 5583L

配置寄存器 B

配置寄存器 B 设置装置的增益。 CRB0 通过 CRB7 识别位的位置，用 CRB 指示在配置寄存器里的位。CRB7 表示数据流中的第一位。括号中的数目显示的是位的默认值。

CRB7	CRB6	CRB5	CRB4	CRB3	CRB2	CRB1	CRB0
GN2 (0)	GN1 (0)	GN0 (1)	(0)	(0)	(0)	(0)	(0)

其中配置寄存器A用来配置输出速率以及增益值,配置寄存器二用来配置转换间隔

模式寄存器

HMC 5583L

模式寄存器

该寄存器是一个8位可读可写的寄存器。该寄存器是用来设定装置的操作模式。 MR0通过MR7识别位的位置，*MR* 表明模式寄存器里的位。MR7指示数据流中的第一位。括号中的数字显示的是位的默认值。

MR7	MR6	MR5	MR4	MR3	MR2	MR1	MR0
(1)	(0)	(0)	(0)	(0)	(0)	MD1 (0)	MD0 (1)

MD1	MD0	模式
0	0	连续测量模式。在连续测量模式下，装置不断进行测量，并将数据更新至数据寄存器。RDY升高，此时新数据放置在所有三个寄存器。在上电或写入模式或配置寄存器后，第一次测量可以在三个数据输出寄存器经过一个2/f_{DO}后设置，随后的测量可用一个频率f_{DO}进行 ， f_{DO}为数据输出的频率。
0	1	单一测量模式（默认）。当选择单测量模式时，装置进行单一测量，RDY设为高位并回到闲置模式。模式寄存器返回闲置模式位值。测量的数据留在输出寄存器中并且RDY仍然在高位，直到数据输出寄存器读取或完成另一次测量。
1	0	闲置模式。装置被放置在闲置模式。
1	1	闲置模式。装置被放置在闲置模式。

用于模式选择:连续测量模式,单一测量模式,闲置模式.

输出数据寄存器

分两类MSB以及LSB分别存储高位以及低位数据,该值以16位二进制补码的形式存在

数据输出 X 寄存器是两个 8 位寄存器，数据输出寄存器 A 和 B。这些寄存器储存从通道 X 所测量结果。数据输出 X 寄存器 A 储存一个来自测量结果中的 MSB(高位数据)，数据输出 X 寄存器 B 储存一个来自测量结果中的 LSB（低位数据）。存储在这两个寄存器的值是一个 16 位值以二进制的补码形式存在，其范围是 0xF800 到 0x07FF。DXRA0 至 DXRA7、DXRB0 至 DXRB7 标识出位置， *DXRA* 和 *DXRB* 标识出在数据输出寄存器 X 中的位。DXRA7 和 DXRB7 标识出数据流的第一位，括号中的数目显示该位的默认值。

在事件的ADC上溢或下溢阅读给定的通道，或者如果有一个数学溢出的过程，这种数据寄存器将包含-4096的值。在下一次有效测量完成进行之后，该寄存器上的值将被清除。

DXRA7	DXRA6	DXRA5	DXRA4	DXRA3	DXRA2	DXRA1	DXRA0
(0)	(0)	(0)	(0)	(0)	(0)	(0)	(0)
DXRB7	DXRB6	DXRB5	DXRB4	DXRB3	DXRB2	DXRB1	DXRB0
(0)	(0)	(0)	(0)	(0)	(0)	(0)	(0)

相关代码

```
 void hmc_init(void)
{
      myiic_init();
// 配置寄存器A,B
//  分别配置输出速率和采样平均数，增益
      hmc_writereg(hmc5338l_CONFIG_A,0x70);
      hmc_writereg(hmc5338l_CONFIG_B,0x20);
 //配置模式寄存器,连续测量模式
      hmc_writereg(hmc5338l_MOD,0x00);
}
```