[MS5611气压计数据采集(模拟IIC)/温度采集/相对高度求解\_岁月 靜好的博客-CSDN博客](https://blog.csdn.net/qq_43445076/article/details/110439895)

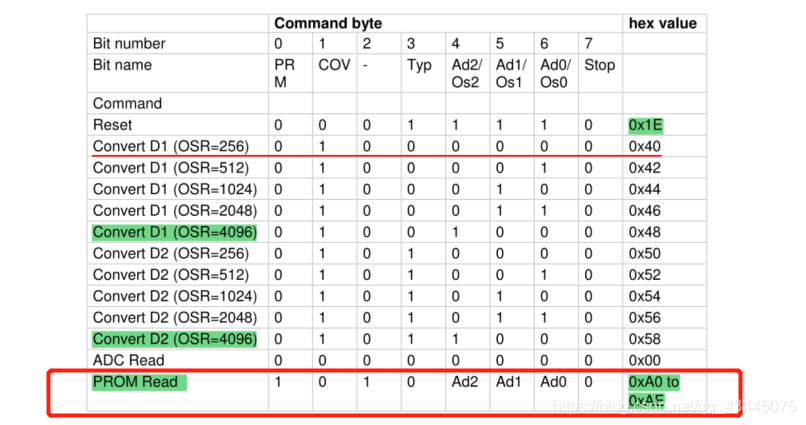
1、 MS5611气压计属性

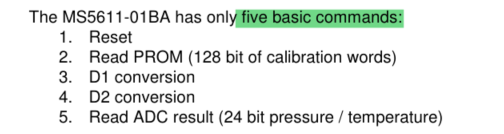
1.1 基础属性

MS5611使用24位ADC,可以采集温度和气压，并且温度可以用来补偿气压，MS5611在出厂时进行了校准，校准的6个系数存储在PROM寄存器中， PROM寄存器起始地址0XA0, 从0xA0到0XAE, 一共16字节， 一共16\*8 = 128位，其中每两个字节为一个系数：

第一个系数： 制造商定的，我们不用在意

第二到第七个系数：我们需要读取，后面用于气压计算

第八个系数：CRC

1.2 5个命令（COMMANDS）

分别是：

1、复位

2、读取PROM

3、D1转换, 其实就是设置读取气压的一些参数，然后MS5611会返回24字节（下文会说明）

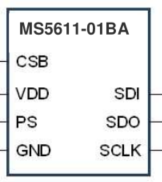
4、D2转换，其实就是设置读取温度的一些参数

5、读取ADC中的数据

1.3 SPI和IIC接口

IIC模式：PS拉高

SPI模式：PS拉低



本实验用的是模拟IIC读取数据

IIC模式中CSB拉低时, IIC从设备地址为0111 0111, CSB拉高时，IIC设备地址为0111 0110

2、 温度气压读取

IIC开始、结束、等待ack、读取发送函数等（参考匿名）

#define MS5611\_ADDR 0x77 //0xee //

#define CMD\_RESET 0x1E // ADC reset command

#define CMD\_ADC\_READ 0x00 // ADC read command

#define CMD\_ADC\_CONV 0x40 // ADC conversion command

#define CMD\_ADC\_D1 0x00 // ADC D1 conversion

#define CMD\_ADC\_D2 0x10 // ADC D2 conversion

#define CMD\_ADC\_256 0x00 // ADC OSR=256

#define CMD\_ADC\_512 0x02 // ADC OSR=512

#define CMD\_ADC\_1024 0x04 // ADC OSR=1024

#define CMD\_ADC\_2048 0x06 // ADC OSR=2048

#define CMD\_ADC\_4096 0x08 // ADC OSR=4096

#define CMD\_PROM\_RD 0xA0 // Prom read command

#define PROM\_NB 8

#define MS5611\_OSR 0x08 //CMD\_ADC\_4096

//开始信号

int I2c\_Soft\_Start()

{

MS5611\_IIC\_SDA = 1;

MS5611\_IIC\_SCL = 1;

delay\_us(4);

if(!MS5611\_READ\_SDA)return 0; //SDA线为低电平则总线忙,退出

MS5611\_IIC\_SDA = 0;

delay\_us(4);

if(MS5611\_READ\_SDA) return 0; //SDA线为高电平则总线出错,退出

MS5611\_IIC\_SDA = 0;

delay\_us(4);

return 1;

}

//结束信号

void I2c\_Soft\_Stop()

{

MS5611\_IIC\_SCL = 0;

delay\_us(4);

MS5611\_IIC\_SDA = 0;

delay\_us(4);

MS5611\_IIC\_SCL = 1;

delay\_us(4);

MS5611\_IIC\_SDA = 1;

delay\_us(4);

}

//应答

void I2c\_Soft\_Ask()

{

MS5611\_IIC\_SCL = 0;

delay\_us(4);

MS5611\_IIC\_SDA = 0;

delay\_us(4);

MS5611\_IIC\_SCL = 1;

delay\_us(4);

MS5611\_IIC\_SCL = 0;

delay\_us(4);

}

//非应答

void I2c\_Soft\_NoAsk()

{

MS5611\_IIC\_SCL = 0;

delay\_us(4);

MS5611\_IIC\_SDA = 1;

delay\_us(4);

MS5611\_IIC\_SCL = 1;

delay\_us(4);

MS5611\_IIC\_SCL = 0;

delay\_us(4);

}

// 等待回复

int I2c\_Soft\_WaitAsk(void) //返回为:=1无ASK,=0有ASK

{

u8 ErrTime = 0;

MS5611\_IIC\_SCL = 0;

delay\_us(4);

MS5611\_IIC\_SDA = 1;

delay\_us(4);

MS5611\_IIC\_SCL = 1;

delay\_us(4);

while(MS5611\_READ\_SDA)

{

ErrTime++;

if(ErrTime>50)

{

I2c\_Soft\_Stop();

return 1;

}

}

MS5611\_IIC\_SCL = 0;

delay\_us(4);

return 0;

}

// IIC发送一个字节

void I2c\_Soft\_SendByte(u8 SendByte) //数据从高位到低位//

{

u8 i=8;

while(i--)

{

MS5611\_IIC\_SCL = 0;

delay\_us(4);

if(SendByte&0x80)

MS5611\_IIC\_SDA = 1;

else

MS5611\_IIC\_SDA = 0;

SendByte<<=1;

delay\_us(4);

MS5611\_IIC\_SCL = 1;

delay\_us(4);

}

MS5611\_IIC\_SCL = 0;

}

//读1个字节，ack=1时，发送ACK，ack=0，发送NACK

u8 I2c\_Soft\_ReadByte(u8 ask) //数据从高位到低位//

{

u8 i=8;

u8 ReceiveByte=0;

MS5611\_IIC\_SDA = 1;

while(i--)

{

ReceiveByte<<=1;

MS5611\_IIC\_SCL = 0;

delay\_us(4);

MS5611\_IIC\_SCL = 1;

delay\_us(4);

if(MS5611\_READ\_SDA)

{

ReceiveByte|=0x01;

}

}

MS5611\_IIC\_SCL = 0;

if (ask)

I2c\_Soft\_Ask();

else

I2c\_Soft\_NoAsk();

return ReceiveByte;

}

// IIC写一个字节数据

u8 IIC\_Write\_1Byte(u8 SlaveAddress,u8 REG\_Address,u8 REG\_data)

{

I2c\_Soft\_Start();

I2c\_Soft\_SendByte(SlaveAddress<<1);

if(I2c\_Soft\_WaitAsk())

{

I2c\_Soft\_Stop();

return 1;

}

I2c\_Soft\_SendByte(REG\_Address);

I2c\_Soft\_WaitAsk();

I2c\_Soft\_SendByte(REG\_data);

I2c\_Soft\_WaitAsk();

I2c\_Soft\_Stop();

return 0;

}

// IIC读1字节数据

u8 IIC\_Read\_1Byte(u8 SlaveAddress,u8 REG\_Address,u8 \*REG\_data)

{

I2c\_Soft\_Start();

I2c\_Soft\_SendByte(SlaveAddress<<1);

if(I2c\_Soft\_WaitAsk())

{

I2c\_Soft\_Stop();

return 1;

}

I2c\_Soft\_SendByte(REG\_Address);

I2c\_Soft\_WaitAsk();

I2c\_Soft\_Start();

I2c\_Soft\_SendByte(SlaveAddress<<1 | 0x01);

I2c\_Soft\_WaitAsk();

\*REG\_data= I2c\_Soft\_ReadByte(0);

I2c\_Soft\_Stop();

return 0;

}

**2.1 第一步：初始化MS5611**

初始化包括：  
1、复位  
2、读取PROM中的6个关键系数

// 复位

void MS5611\_Reset(void)

{

// MS5611\_ADDR：0x77 CMD\_RESET:0x1E

IIC\_Write\_1Byte(MS5611\_ADDR, CMD\_RESET, 1);

}

u8 MS5611\_Read\_Prom(void)

{

uint8\_t rxbuf[2] = { 0, 0 };

u8 check = 0;

u8 i;

for (i = 0; i < PROM\_NB; i++)

{

check += IIC\_Read\_nByte(MS5611\_ADDR, CMD\_PROM\_RD + i \* 2, 2, rxbuf); // send PROM READ command

ms5611\_prom[i] = rxbuf[0] << 8 | rxbuf[1];

}

if(check == PROM\_NB)

return 1;

else

return 0;

}

#### 在这里插入图片描述2.2 第二步：获取温度数据

首先发送Convert D1命令，进行相应的配置，再发送ADC Read命令，然后MS5611会返回24位的温度数据， 这时候再去读取这些温度数据。

// 写入数据

void MS5611\_Start\_T(void)

{

//CMD\_ADC\_CONV + CMD\_ADC\_D2 + MS5611\_OSR = 0x48

IIC\_Write\_1Byte(MS5611\_ADDR, CMD\_ADC\_CONV + CMD\_ADC\_D2 + MS5611\_OSR, 1); // D2 (temperature) conversion start!

}

// 读取24位温度数据

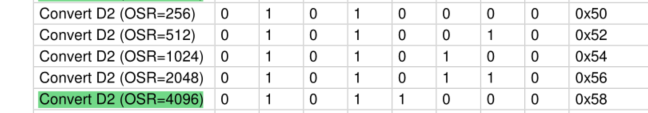
void MS5611\_Read\_Adc\_T(void)

{

IIC\_Read\_nByte( MS5611\_ADDR, CMD\_ADC\_READ, 3, t\_rxbuf ); // read ADC

}

#### 2.3 第三步：获取气压数据

  
首先发送Convert D2命令，进行相应的配置，然后再发送ADC Read命令， 然后MS5611会返回24位的气压数据， 这时候再去读取这些气压数据。

// 写入数据

void MS5611\_Start\_P(void)

{

CMD\_ADC\_CONV + CMD\_ADC\_D1 + MS5611\_OSR = 0x58

IIC\_Write\_1Byte(MS5611\_ADDR, CMD\_ADC\_CONV + CMD\_ADC\_D1 + MS5611\_OSR, 1); // D1 (pressure) conversion start!

}

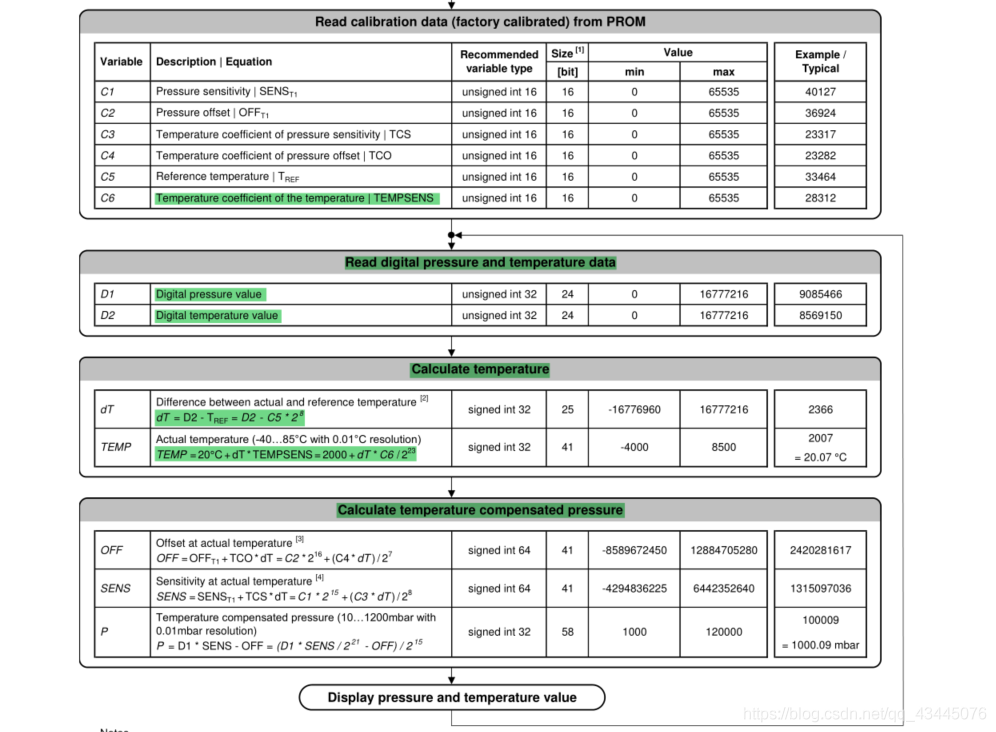
// 读取24位气压数据

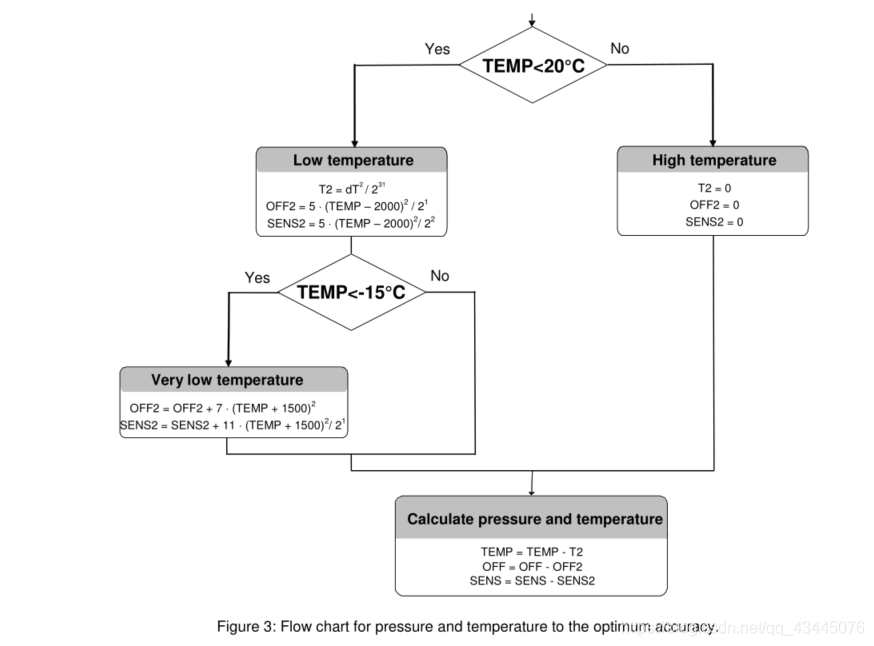
void MS5611\_Read\_Adc\_P(void)

{

IIC\_Read\_nByte(MS5611\_ADDR, CMD\_ADC\_READ, 3, p\_rxbuf); // read ADC

}

3、温度补偿气压



绝对高度(海拔)的求解公式：

H= 44300\*(1- (P/P0)^(1/5.256) )

H为高度

P0为标准大气压(≈101Kpa)

P为读取的气压 （注意单位得和P0相同）

相对高度(例如飞机起飞前和飞行中的高度差)的求解

这里提供一个思路：

MS5611开机上电一段时间后，读取静止100组数据并转换成海拔，取个平均，这个海拔高度就相当于基准海拔，如果你把MS5611拿高或者拿低， 此时的海拔与基准海拔相见就是相对高度。

ps:求相对高度的时候，可以对此时的海拔高度数据进行一个滑动窗滤波，这样求出的相对高度比较平滑。

void MS5611\_BaroAltCalculate(void)

{

float height = 0;

int64\_t off2 = 0, sens2 = 0, delt = 0, f = 0, f2 = 0;

int32\_t temperature = 0, pressure = 0, T2 = 0;

int32\_t dT = 0;

int64\_t off =0, sens =0;

static int num = 0;

static int cnt = 0;

double ms5611SumDat = 0.0;

ms5611\_ut = (t\_rxbuf[0] << 16) | (t\_rxbuf[1] << 8) | t\_rxbuf[2]; // 读取的原始温度值

ms5611\_up = (p\_rxbuf[0] << 16) | (p\_rxbuf[1] << 8) | p\_rxbuf[2]; // 读取的原始气压值

dT = ms5611\_ut - ((uint32\_t)ms5611\_prom[5] << 8);

off = ((uint64\_t)ms5611\_prom[2] << 16) + (((int64\_t)dT \* ms5611\_prom[4]) >> 7);

sens = ((uint64\_t)ms5611\_prom[1] << 15) + (((int64\_t)dT \* ms5611\_prom[3]) >> 8);

// 实际温度： 2007：20.07 ℃

temperature = 2000 + (((int64\_t)dT \* ms5611\_prom[6]) >> 23);

if (temperature < 2000)

{

T2 = (dT\*dT)>>31;

delt = temperature - 2000;

delt = delt \* delt;

off2 = (5 \* delt) >> 1;

sens2 = (5 \* delt) >> 2;

if (temperature < -1500)

{

delt = temperature + 1500;

delt = delt \* delt;

off2 += 7 \* delt;

sens2 += (11 \* delt) >> 1;

}

}

temperature -= T2;

off -= off2;

sens -= sens2;

ms5611Data.temperature = (double)temperature / 100.0; // ℃

// 温度补偿后的气压 100009 = 1000.09 mbar = 100009 Pa

pressure = (((ms5611\_up \* sens ) >> 21) - off) >> 15; // mbar

ms5611Data.pressure = (double)pressure / 100.0;

height = (double)((1.0f - pow((double)pressure / 101325.0f, 0.190295f)) \* 44330.0f); // meter

ms5611Data.height = height;

// 求相对高度

// 上电100个数据后

if(num > 100)

{

if(cnt < MS5611\_WIN\_FILTER)

{

alt[cnt] = ms5611Data.height;

cnt++;

}

else

{

ms5611SumDat = 0.0;

for(int i=0; i<MS5611\_WIN\_FILTER-1; i++)

{

alt[i] = alt[i+1];

ms5611SumDat += alt[i];

}

alt[cnt-1] = ms5611Data.height;

ms5611SumDat += alt[cnt-1];

ms5611CurAlt = ms5611SumDat/MS5611\_WIN\_FILTER; // 当前100组数据的高度平均

if(baroCalOk == true)

{

ms5611CurAlt -= ms5611StartAlt; // 当前相对高度

}

}

num = 100;

}

num++;

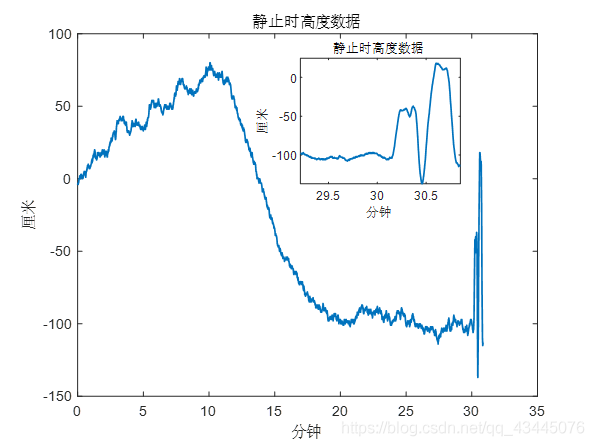
}

4、总结

MS5611主要是用于计算高度，比如可以用于四旋翼的定高等。

MS5611求海拔高度的时候，由于受温度，气流等各种因素的影响，最终的结果并不理想，

例如下面是我静止采集的30分钟左右的数据，可以明显看到高度是飘的。



最后在30分钟左右是我把ms5611拿上拿下来测试的， 所以出现了上图中的波动。