源码阅读报告

#### 主函数

源码注释：

void main(void) {

initUARTtest(); // 初始化串口

Write\_Data(0x26, 0x39); // 激活传感器

while(1){ // 持续读取传感器数据

// 下面5行分别读取传感器的5个寄存器，存储到data\_result数组中

Read\_Data(0x01, &data\_result[1]);

Read\_Data(0x02, &data\_result[2]);

Read\_Data(0x03, &data\_result[3]);

Read\_Data(0x04, &data\_result[4]);

Read\_Data(0x05, &data\_result[5]);

// 下面7行将压力的20位数据合并，并换算单位

pressure\_temp = data\_result[1];

pressure\_temp = pressure\_temp << 8;

pressure\_temp = pressure\_temp + data\_result[2];

pressure\_temp = pressure\_temp << 8;

pressure\_temp = pressure\_temp + data\_result[3];

// 第3个寄存器只有前4位是压力数据

pressure\_temp = pressure\_temp >> 4;

pressure\_value = pressure\_temp \* 0.25;

// 格式化字符串并发送

sprintf(pressure\_result,"%s","pressure: ");

UartTX\_Send\_String(pressure\_result, 10);

sprintf(pressure\_result,"%9.2f",pressure\_value);

pressure\_result[9] = ':';

UartTX\_Send\_String(pressure\_result, 10);

// 下面5行从两个寄存器中读取温度数据并换算单位

temp\_temp = data\_result[4];

temp\_temp = temp\_temp << 8;

temp\_temp = temp\_temp + data\_result[5];

temp\_temp = temp\_temp >> 4;

temp\_value = temp\_temp \* 0.0625;

// 格式化温度数据并发送

sprintf(pressure\_result,"%s","temperature: ");

UartTX\_Send\_String(pressure\_result, 13);

sprintf(temp\_result, "%8.4f", temp\_value);

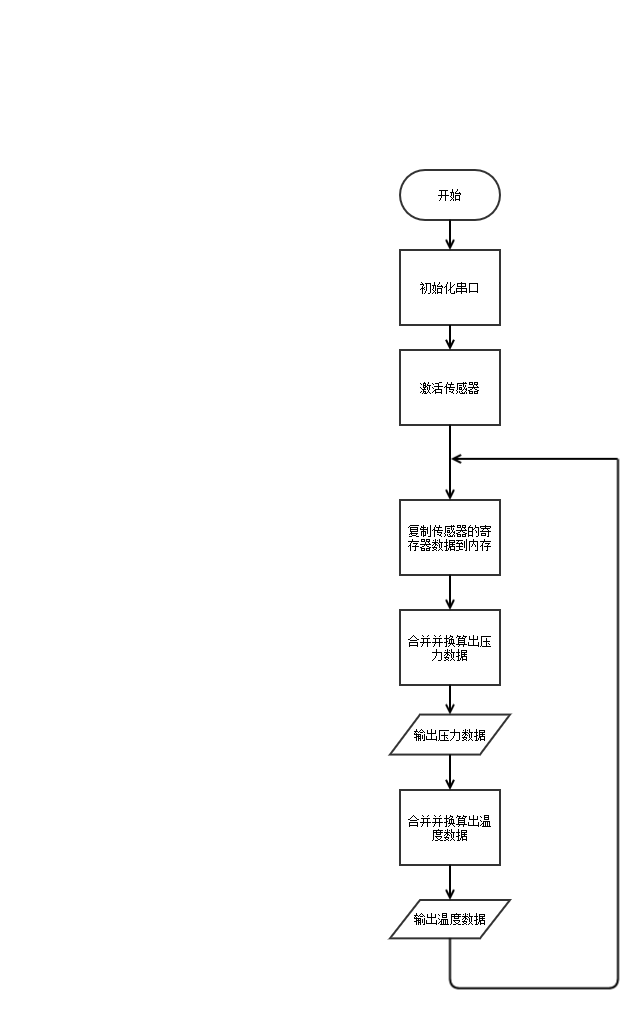
temp\_result[8] = '\n';

UartTX\_Send\_String(temp\_result, 9);

}

}

流程图：



图一、 main函数流程图

#### 读数据 (写数据是读数据的子集，不再单独解析)

源码注释：

void Read\_Data(char reg, char \*data)

{

I2C\_Start(); // 启动I2C总线，当SCL为高电平时使SDA产生一个负跳变

WriteI2CByte(MPL3115A2\_I2C\_ADDRESS + I2C\_WRITE); // 通知传感器要写数据

while(Check\_Acknowledge() == FALSE); // 等待传感器确认接收

WriteI2CByte(reg); // 发送要读取的寄存器

while(Check\_Acknowledge() == FALSE); // 等待传感器响应

I2C\_Start(); // 再次开启I2C会话

WriteI2CByte(MPL3115A2\_I2C\_ADDRESS + I2C\_READ); // 通知传感器要读数据

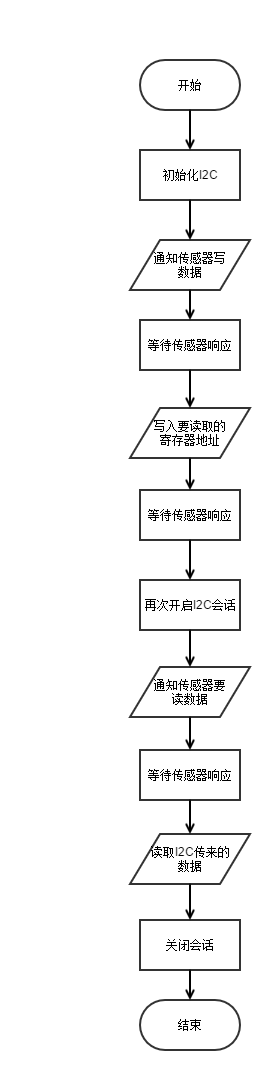
while(Check\_Acknowledge() == FALSE); // 等待传感器响应

\*data = ReadI2CByte(); // 读取数据并写进\*data

I2C\_Stop(); // 关闭I2C会话

}

流程图：



图二、 Read\_Data流程图

#### 串口发送字符串

源码注释：

void UartTX\_Send\_String(char \*Data,int len)

{

int j;

for(j=0;j<len;j++)

{

U0DBUF = \*Data++; // 将下一个字节放入BUF中

while(UTX0IF == 0); // 等待发送

UTX0IF = 0; // 标志为未发送状态

}

}

#### I2C发送字节

源码注释：

void WriteI2CByte(char b)

{

char i;

for(i=0;i<8;i++) // 逐位发送一个字节

{

if((b<<i)&0x80) // 判断第i位是1还是0

{

SEND\_1();

}

else

{

SEND\_0();

}

}

}

#### I2C读取字节

源码注释：

char ReadI2CByte(void)

{

char b=0,i;

WriteSDA1(); // 开始读取

for(i=0;i<8;i++)

{

// 根据协议进行同步

WriteSCL0();

Delay\_1u(50);

WriteSCL1();

Delay\_1u(50);

ReadSDA(); //读取一位

F0=SDA;//寄存器中的一位,用于存储SDA中的一位数据

if(F0==1) //判断读取到的是1还是0

{

b=b<<1; //将之前已读取的部分左移

b=b|0x01; //将末位置1(因为读取到的是1)

}

else

b=b<<1; //将之前已读取的部分左移，并将末尾置0(左移操作自动完成)

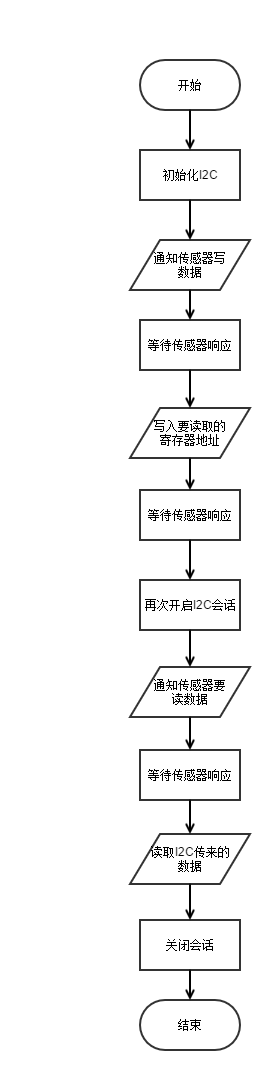
}

WriteSCL0(); // 根据协议结束会话

return b;

}

和I2C通信有关的底层函数如SEND\_0, SEND\_1, Check\_Acknowledge, ReadSDA, WriteSCL0, WriteSCL1, WriteSDA0, WriteSDA1等都是直接实现协议。



图二、 Read\_Data流程图