Application Note

ATK-MB017 模块使用说明

气压计模块

使用说明

正点原子

广州市星翼电子科技有限公司

修订历史

版本	日期	原因
V1.0	2024/11/01	第一次发布

目 录

1，硬件连接

这里以正点原子 M48Z-M3 最小系统板 STM32F103 版为例，给大家介绍一下模块和板卡的连接方法。其它板卡与模块的硬件连接方法，请大家在“**ATK-MB017 气压计模块\3，程序源码\相应板卡例程文件夹\readme.txt**”路径下查看。

气压计模块可通过杜邦线与正点原子 M48Z-M3 最小系统板 STM32F103 版进行连接，具体的连接关系，如下表所示：

模块对应开发板	连接关系				
气压计模块	VCC	GND	SDA	SCL	INT
M48Z-M3 最小系统板 STM32F103 版	3.3V/5V	GND	PA3	PA2	NC

表 1.1.1 气压计模块与 M48Z-M3 最小系统板 STM32F103 版连接关系

2，实验功能

2.1 气压计模块测试实验

2.1.1 功能说明

在本实验中，串口会打印气压计模块的提示信息。需要查看这部分实验信息的用户，可用杜邦线将最小系统板 STM32F103 的 PA9 引脚和 GND 连接至外部的 USB 转串口设备，这样就可以通过 XCOM 上位机查看串口打印的信息了。

当准确识别到气压计模块后，串口会实时打印气压计模块的气压值、温度值以及海拔高度；

开发板的 LED0 闪烁，提示程序运行。

2.1.2 源码解读

打开本实验的工程文件夹，能够在./Drivers/BSP 目录下看到 ATK_SPL06 文件夹，和 IIC 文件夹，其中 ATK_SPL06 文件夹中就包含了气压计模块的驱动文件，IIC 文件夹中就包含了软件模拟 IIC 的驱动文件如下图所示：

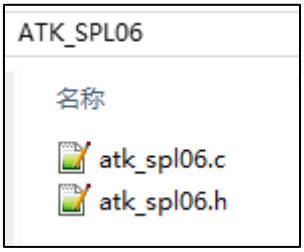

图 2.1.2.1 气压计模块驱动代码

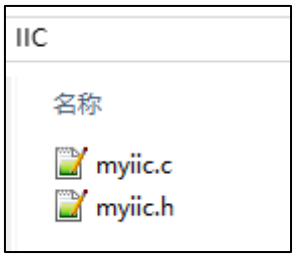

图 2.1.2.2 IIC 驱动代码

2.1.2.1 气压计模块驱动

下面将简要介绍 atk_spl06.c 中几个重要的 API 函数。

1. 函数 atk_spl06_get_calib_param ()

该函数实现读取 SPL06-001 气压传感器的校准系数数据，用于后续的温度和气压补偿计算，具体的代码，如下所示：

```
/**
 * @brief   读取校准数据（用于计算气压值和温度值）
 * @param   无
 * @retval  无
 */
void atk_spl06_get_calib_param(void)
```

```
{
    uint8_t buffer[SPL06_CALIB_COEFFICIENT_LENGTH] = {0};

    atk_spl06_read_nbytes(SPL06_COEFFICIENT_CALIB_REG, buffer,
    SPL06_CALIB_COEFFICIENT_LENGTH);

    spl06calib.c0 = (int16_t)buffer[0]<<4 | buffer[1]>>4;
    spl06calib.c0 = (spl06calib.c0 & 0x0800) ? (spl06calib.c0 | 0xF000) :
    spl06calib.c0;

    spl06calib.c1 = (int16_t)(buffer[1] & 0x0F)<<8 | buffer[2];
    spl06calib.c1 = (spl06calib.c1 & 0x0800) ? (spl06calib.c1 | 0xF000) :
    spl06calib.c1;

    spl06calib.c00 = (int32_t)buffer[3]<<12 | (int32_t)buffer[4]<<4 |
    (int32_t)buffer[5]>>4;
    spl06calib.c00 = (spl06calib.c00 & 0x080000) ? (spl06calib.c00 | 0xFFF00000) :
    spl06calib.c00;

    spl06calib.c10 = (int32_t)(buffer[5] & 0x0F)<<16 | (int32_t)buffer[6]<<8 |
    (int32_t)buffer[7];
    spl06calib.c10 = (spl06calib.c10 & 0x080000) ? (spl06calib.c10 | 0xFFF00000) :
    spl06calib.c10;

    spl06calib.c01 = (int16_t)buffer[8]<<8 | buffer[9];
    spl06calib.c11 = (int16_t)buffer[10]<<8 | buffer[11];
    spl06calib.c20 = (int16_t)buffer[12]<<8 | buffer[13];
    spl06calib.c21 = (int16_t)buffer[14]<<8 | buffer[15];
    spl06calib.c30 = (int16_t)buffer[16]<<8 | buffer[17];
}
```

从上述代码可以看出，首先通过 atk_spl06_read_nbytes 函数从 SPL06 的校准系数寄存器（0x10）中读取校准数据，然后，将读取到的校准系数转换为正确的格式，存入 spl06calib 结构体中，校准数据包括 c0、c1、c00、c10、c01、c11、c20、c21 和 c30，用于温度和气压值的补偿计算。

2. 函数 atk_spl06_rateset ()

该函数主要实现设置 SPL06-001 的测量速率和过采样率，具体代码，如下所示：

```
/**
 * @brief   设置对应传感器的测量速率以及过采样率
 * @param   sensor:         传感器类型
 * @param   measureRate:    测量速率
 * @param   oversamplRate:  过采样率
 * @retval  无
 */
void atk_spl06_rateset(spl06Sensor_e sensor, uint8_t measureRate, uint8_t
```

```
        oversamplRate)
{
    uint8_t reg;

    if (sensor == PRESURE_SENSOR)                /* 压力传感器 */
    {
        kp = scale_factor[oversamplRate];
        atk_spl06_write_byte(SPL06_PRESSURE_CFG_REG, measureRate<<4 |
        oversamplRate);

        if (oversamplRate > SPL06_OVERSAMP_8)
        {
            atk_spl06_read_nbytes(SPL06_INT_FIFO_CFG_REG, &reg, 1);
            atk_spl06_write_byte(SPL06_INT_FIFO_CFG_REG, reg | 0x04);
        }
    }
    else if (sensor == TEMPERATURE_SENSOR)       /* 温度传感器 */
    {
        kt = scale_factor[oversamplRate];
        atk_spl06_write_byte(SPL06_TEMPERATURE_CFG_REG, measureRate<<4 |
          oversamplRate | 0x80); /* Using mems temperature */
        if (oversamplRate > SPL06_OVERSAMP_8)
        {
            atk_spl06_read_nbytes(SPL06_INT_FIFO_CFG_REG, &reg, 1);
            atk_spl06_write_byte(SPL06_INT_FIFO_CFG_REG, reg | 0x08);
        }
    }
}
```

该函数首先通过入口参数，判断需要设置的传感器类型（压力传感器或温度传感器），对于压力传感器，将配置写入压力配置寄存器（0x06），并设置缩放因子 kp；对于温度传感器，将配置写入温度配置寄存器（0x07），并设置缩放因子 kt。

3. 函数 atk_spl06_get_data()

该函数主要实现获取 SPL06-001 的气压和温度数据，并进行补偿计算，同时计算并返回海拔高度。代码如下：

```
/**
 * @brief   获取 spl06 数据
 * @param   spl06_result_t *p_res: 该结构体用于存放获取到的温度、压力、海拔高度数据
 * @retval  无
 */
void atk_spl06_get_data(spl06_result_t *p_res)
{
    uint8_t data[SPL06_DATA_FRAME_SIZE];
```

```
    /* 读取寄存器中的气压值和温度值 */
    atk_spl06_read_nbytes(SPL06_PRESSURE_MSB_REG, data,
    SPL06_DATA_FRAME_SIZE);
    /* 得到气压数据 */
    p_res->praw = (int32_t)data[0] << 16 | (int32_t)data[1] << 8 |
    (int32_t)data[2];
    p_res->praw = (p_res->praw & 0x800000) ? (0xFF000000 | p_res->praw) :
    p_res->praw;
    /* 得到温度数据 */
    p_res->traw = (int32_t)data[3] << 16 | (int32_t)data[4] << 8 |
    (int32_t)data[5];
    p_res->traw = (p_res->traw & 0x800000) ? (0xFF000000 | p_res->traw) :
    p_res->traw;

    /* 获取计算后的补偿温度值和气压值 */
    p_res->tcomp = atk_spl06_get_temperature(p_res->traw);  /* 单位℃ */
    /* 单位 hPa */
    p_res->pcomp = atk_spl06_get_pressure(p_res->praw, p_res->traw) / 100;
    p_res->asl = atk_spl06_pressure_to_asl(p_res->pcomp);  /* 转换成海拔高度 */
}
```

该函数首先从 SPL06 的数据寄存器中读取原始的气压和温度数据，然后使用 atk_spl06_get_temperature() 和 atk_spl06_get_pressure() 进行温度和气压补偿计算，，接着，调用 atk_spl06_pressure_to_asl() 将气压转换为海拔高度。最后将计算结果存储到 spl06_result_t 结构体中，包含气压值、温度值和海拔高度。特别提醒：关于气压、温度以及海拔高度的转换公式可查看 SPL06-001 的数据手册的 5.6 章节

4. 函数 atk_spl06_init ()

该函数实现初始化 SPL06-001 气压传感器。代码如下：

```
/**
 * @brief       初始化 SPL06
 * @param       无
 * @retval      检测结果
 *              0: 检测成功
 *              1: 检测失败
 */
uint8_t atk_spl06_init(void)
{
    uint8_t spl06_id = 0, res = 0;

    /* 初始化 IIC 接口 */
    iic_init();

    spl06_id = atk_spl06_read_byte(SPL06_CHIP_ID);
    if(spl06_id == SPL06_DEFAULT_CHIP_ID)
```

```
    {
        printf("SPL06 ID: 0x%X\r\n", spl06_id);
        res = 0;
    }
    else
    {
        return 1;
    }
    /* 读取校准数据 */
    atk_spl06_get_calib_param();

    /* 设置压力和温度传感器的测量速率以及过采样率 */
    atk_spl06_rateset(PRESURE_SENSOR, SPL06_MWASURE_16, SPL06_OVERSAMP_64);
    atk_spl06_rateset(TEMPERATURE_SENSOR, SPL06_MWASURE_16,
    SPL06_OVERSAMP_64);

    /* 设置连续采集模式。连续采集温度和压力 */
    res = atk_spl06_write_byte(SPL06_MODE_CFG_REG, SPL06_CONTINUOUS_MODE);
    return res;
}
```

从上述代码可以看出，首先初始化 IIC 接口，读取并验证 SPL06 的芯片 ID，如果芯片 ID 正确，读取传感器的校准系数数据，设置气压和温度传感器的测量速率和过采样率，最后，将传感器设置为连续采集模式。

2.1.2.2 IIC 驱动

在图 2.1.2.2 中，myiic.c 和 myiic.h 是开发板与气压计模块通讯而使用的模拟 IIC 驱动文件，关于模拟 IIC 的驱动介绍，请查看正点原子各个开发板对应的开发指南中模拟 IIC 对应的章节。

2.1.2.3 实验测试代码

实验的测试代码为文件 demo.c，在工程目录下的 User 子目录中。测试代码的入口函数为 demo_run()，具体的代码，如下所示：

```
/* 存放气压计数据结构体 */
spl06_result_t spl06_result;

/**
 * @brief       例程演示入口函数
 * @param       无
 * @retval      无
 */
void demo_run(void)
{
    while(atk_spl06_init())     /* 检测不到 spl06 */
    {
        printf("SPL06 Check Failed!\r\n");
        delay_ms(500);
```

```
        LED0_TOGGLE();          /* 红灯闪烁 */
    }
    printf("SPL06 Ready!\r\n");

    while (1)
    {
         /* 获取 spl06 数据 */
        atk_spl06_get_data(&spl06_result);

        printf("气压值: %.2fhPa\r\n", spl06_result.pcomp);
        printf("温度值: %.2f℃\r\n", spl06_result.tcomp);
        printf("海拔高度: %.2fm\r\n", spl06_result.asl);
        printf("\r\n");

        delay_ms(500);
        LED0_TOGGLE();
    }
}
```

demo_run 函数的流程大致是：在 demo_run 函数外部声明了一个 spl06_result_t 类型的变量 spl06_result，用于存储从 SPL06 传感器获取的数据，包括气压值、温度值和海拔高度。在完成气压传感器的初始化后，在 while 循环中，通过 atk_spl06_get_data(&spl06_result)函数获取传感器的数据，数据将存储在 spl06_result 结构体中。然后使用 printf 打印获取到的气压、温度和海拔高度。

2.1.3 实验现象

将气压计模块按照第一节“硬件连接”中介绍的连接方式与开发板连接，并将实验代码编译烧录至开发板中，本实验使用串口输出调试信息，因此需将开发板的 PA9 连接至 DAP 虚拟串口（或 USB 转 TTL 模块）的 RX 引脚。并通过串口调试助手查看实验信息输出，如下图所示：

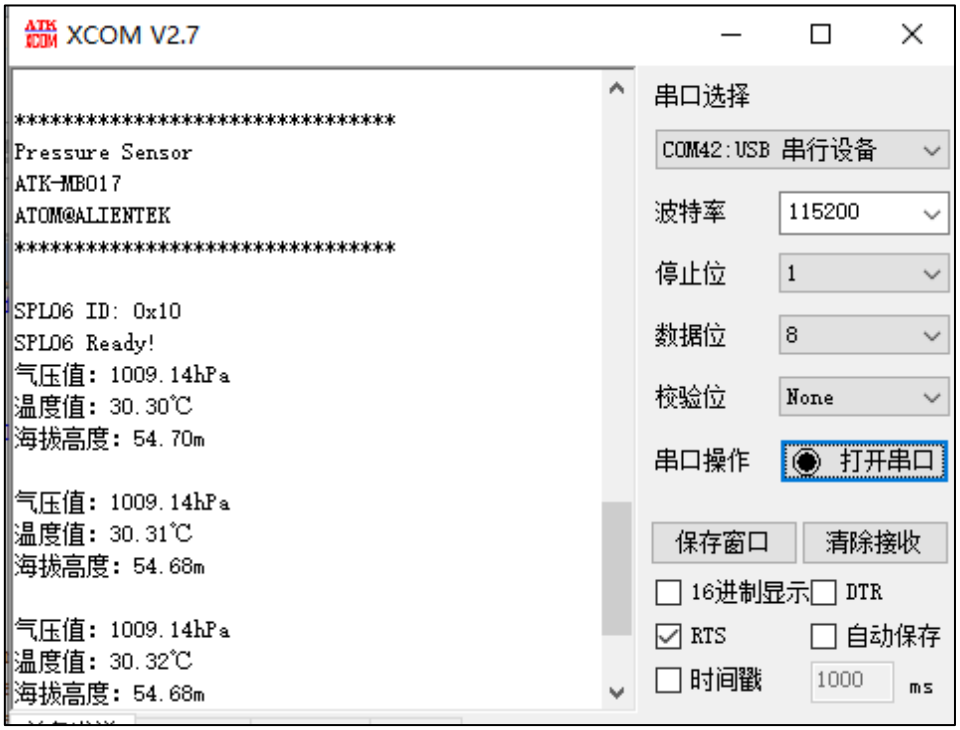

图 2.1.3.2 串口调试助手显示内容

3，其他

1、 购买地址：

天猫：https://zhengdianyuanzi.tmall.com

淘宝：https://openedv.taobao.com

2、 资料下载

模块资料下载地址：http://www.openedv.com/docs/index.html

3、 技术支持

公司网址：www.alientek.com

技术论坛：http://www.openedv.com/forum.php

在线教学：www.yuanzige.com

B 站视频：https://space.bilibili.com/394620890

传真：020-36773971

电话：020-38271790