HiSpark-WiFi-IoT 烟雾气体传感器编程指南

文档版本　00B01

发布日期　2020/8/3

商标声明

 、HISILICON、海思和其他海思商标均为上海海思技术有限公司的商标。

本文档提及的其他所有商标或注册商标，由各自的所有人拥有。

注意

您购买的产品、服务或特性等应受海思公司商业合同和条款的约束，本文档中描述的全部或部分产品、服务或特性可能不在您的购买或使用范围之内。除非合同另有约定，海思公司对本文档内容不做任何明示或默示的声明或保证。

由于产品版本升级或其他原因，本文档内容会不定期进行更新。除非另有约定，本文档仅作为使用指导，本文档中的所有陈述、信息和建议不构成任何明示或暗示的担保。

上海海思技术有限公司

地址：　　上海市青浦区金泽镇（西岑）水秀路 318 号 101 室　邮编：201718

网址：　　http://www.hisilicon.com

前　言

概述

本文档主要介绍基于海思 WiFi 芯片 Hi3861 开发的 HiSpark-WiFi-IoT 套件演示指导书。

产品版本

与本文档相对应的主芯片版本如下。

产品名称	产品版本
Hi3861	V100R001C00SPC021

读者对象

本文档（本指南）主要适用于以下工程师：

- 软件开发工程师
- 硬件开发工程师

修订记录

修订记录累积了每次文档更新的说明。最新版本的文档包含以前所有文档版本的更新内容。

修订日期	版本	修订说明
2020-08-3	00B01	第一次临时版本发布。

目 录

1 WIFI-IoT 开发板烟雾气体传感器

1.1 烟雾气体传感器介绍

WIFI-IOT 开发板套件，名称 HiSpark，以下简称 HiSpark 开发板套件。

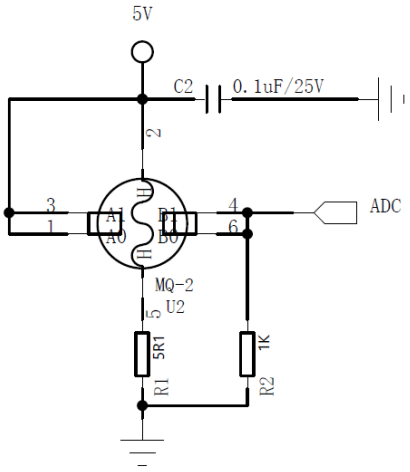
HiSpark 开发板的硬件上有 1 个烟雾气体传感器，位于智能环境监测模块。

智能环境监测模块作用：通过 ADC 功能实时采集当前环境可燃气体浓度，在超过阈值时蜂鸣器会打出警报。

图 1.1-1

1.2 烟雾气体传感器实现逻辑

图 1.1-2



原理图中引脚连接说明: ADC ---- GPIO_11

Rs 为传感器当前阻值，R0 为洁净空气中的值，ppm 为可燃气体浓度。三者之间的关系为

①Rs/R0 = 11.5428*ppm^(-0.6549) 。

由上图的电路图可得出 ②(Vc – V2)/Rs = V2/R2。Vc = 5V，V2 位 R2 电阻电压，即 ADC 采集到的电压,R2 = 1k。在空气洁净时测出电压为 0.25V 左右。故 R0 = 19K。

通过公式①②可以推导出电压 V2 与 ppm 值的关系为：

ppm = pow(11.5428*19* V2/(5- V2),1.0/0.6549);

函数 pow 为计算 x 的 y 次方，x,y 均为 double 类型。

1.3 烟雾气体传感器软件实现

1. HiSpark 开发板上 ADC 初始化配置如下：

图 1.3-1

```
ret = hi_cipher_init();
if (ret != HI_ERR_SUCCESS) {
    err_info |= PERIPHERAL_INIT_ERR_CIPHER;
}
hi_adc_init();
```

读取 ADC 数值并进行电压转化：

之后直接调用 hi_void mq2_get_data(hi_void)函数获取电压值并转换为传感器数值，函数实现如下：

图 1.3-2

```
hi_void mq2_get_data(hi_void)
{
    hi_u32 ret =0;
    hi_u8 i =0;
    hi_u16 data =0;  /* 10 */
    hi_float voltage;

    ret = hi_adc_read(HI_ADC_CHANNEL_5, &data, HI_ADC_EQU_MODEL_4, HI_ADC_CUR_BAIS_DEFAULT, 0xFF)
    if (ret != HI_ERR_SUCCESS) {
        printf("ADC Read Fail\n");
        return HI_NULL;
    }
    voltage = (hi_float)(data * 1.8 * 4 / 4096.0);  /* vlt * 1.8* 4 / 4096.0为将码字转换为电压 */
    g_combustible_gas_value = mq2_get_ppm(voltage);
}
```

ADC采集数值并转换为电压

电压转换为气体浓度实现函数如下：

图 1.3-3

```
hi_float mq2_get_ppm(hi_float voltage)
{

    hi_u16 adc_value =0;
    hi_double ppm =0;
    hi_float vol_d =0;
    hi_float aht_ratio =0;
    static hi_u8 flag = 1;
    static hi_u8 count = 0;
    hi_float RS = (5 - voltage) / voltage * RL;
    // printf("RS:%03f\r\n", RS);
     printf("R0/RS:%03f\r\t", RS/R0);
    memset(&ppm, 0x0, sizeof(ppm));
    aht_ratio = (hi_float) (RS/R0);
    hi_pwm_init(HI_PWM_PORT_PWM0);
    hi_pwm_set_clock(PWM_CLK_160M);
    hi_io_set_func(HI_IO_NAME_GPIO_9, HI_IO_FUNC_GPIO_9_PWM0_OUT);
    hi_gpio_set_dir(HI_IO_NAME_GPIO_9, HI_GPIO_DIR_OUT);


    ppm = pow(11.5428*19*voltage/(5-voltage),1.0/0.6549);
```