

# OpenHarmonyE53模块开 发-智慧烟感



# 目 录

## CONTENTS

- [ 01 ] 硬件设计
- [ 02 ] 硬件连接
- [ 03 ] 烟感检测原理分析
- [ 04 ] API分析
- [ 05 ] 实例分析

# 01

## 硬件设计

- 小凌派+E53智慧烟感模块。

## 02

## 硬件连接

E53智慧烟感模块与小凌派E53接口连接

## 03

## 烟感检测原理分析

MQ-2 型烟雾传感器属于二氧化锡半导体气敏材料,属于表面离子式 N 型半导体。处于  $200\sim 300$  摄氏度时,二氧化锡吸附空气中的氧,形成氧的负离子吸附,使半导体中的电子密度减少,从而使其电阻值增加。当与烟雾接触时,如果晶粒间界处的势垒收到烟雾的调至而变化,就会引起表面导电率的变化。利用这一点就可以获得这种烟雾存在的信息,烟雾的浓度越大,导电率越大,输出电阻越低,则输出的模拟信号就越大。通过ADC采集MQ-2传感器输出电压,根据公式转换即可得到空气烟雾浓度值。

## 04

# API分析

OpenHarmony E53模块开发-智慧烟感头文件在:

`/vendor/lockzhiner/rk2206/samples/c7_e53_intelligent_smoke_sensor/include/e53_intelligent_smoke_sensation.h`

OpenHarmony E53模块开发-智慧烟感，主要API接口:

- (1) 智慧烟感模块初始化;
- (2) 烟雾浓度校准;
- (3) 获取烟雾浓度值;
- (4) 获取烟雾浓度报警值
- (5) 蜂鸣器和LED控制

## 04

# API分析

```
uint32_t e53_iss_init(void);
```

该函数主要功能是E53智慧烟感模块初始化，包括初始化eeprom的I2C0、LED1灯的GPIO、BEEP蜂鸣器的GPIO PWM、MQ2烟雾传感器的ADC。

```
void e53_iss_mq2_ppm_calibration(void);
```

该函数主要功能是E53智慧烟感模块ppm校准。

```
float e53_iss_get_mq2_ppm(void);
```

该函数主要功能是E53智慧烟感控制模块获取ppm值。

## 04

## API分析

```
uint16_t e53_iss_get_mq2_alarm_value();
```

该函数主要功能是获取eeprom烟雾浓度报警阈值，大于此阈值的需要报警。

```
void e53_iss_beep_status_set(e53_iss_status_e status);
```

该函数主要功能是E53智慧烟感控制模块蜂鸣器状态设置。

```
void e53_iss_led_status_set(e53_iss_status_e status);
```

该函数主要功能是E53智慧烟感控制模块LED状态设置。



## 05

## 实例分析

打开sdk下面路径的文件

```
vendor/lockzhiner/rk2206/samples/c7_e53_intelligent_smoke_sensor/e53_intelligent_smoke_sensor_example.c
```

在e53\_iss\_example函数中，创建的一个线程e53\_iss\_thread。

```
task.pfnTaskEntry = (TSK_ENTRY_FUNC)e53_iss_thread;  
task.uwStackSize = 1024 * 2;  
task.pcName = "e53_iss_thread";  
task.usTaskPrio = 24;  
ret = LOS_TaskCreate(&thread_id, &task);
```

## 05

## 实例分析

这部分代码是智慧烟感示例代码：

- (1) `e53_iss_init()` 函数初始化E53智慧烟感模块；
- (2) `e53_iss_mq2_ppm_calibration()` 校准MQ2的ppm；
- (3) 循环读取MQ2值判断ppm是否超过预设值，超过预设值蜂鸣器报警LED亮起提升，低于预设值则关闭蜂鸣器报警并关闭LED。

## 05

## 实例分析

```
void e53_iss_thread(void *arg)
{
    float ppm = 0;
    e53_iss_init();
    /*传感器校准*/
    usleep(2000000); // 开机2s后进行校准
    e53_iss_mq2_ppm_calibration(); // 校准传感器
    while (1)
    {
        printf("=====\r\n");
        printf("*****e53_iss_example*****\r\n");
        printf("=====\r\n");
        ppm = e53_iss_get_mq2_ppm();
        printf("ppm: %.3f \n", ppm);
```

```
        /*判断是否达到报警阈值*/
        if (ppm > e53_iss_get_mq2_alarm_value())
        {
            e53_iss_led_status_set(ON);
            e53_iss_beep_status_set(ON);
        }
        else
        {
            e53_iss_led_status_set(OFF);
            e53_iss_beep_status_set(OFF);
        }
        usleep(1000000); // 延时1s
    }
}
```

# 如何创建、申请、释放互斥锁

## 4.修改编译脚本

修改 `vendor/lockzhiner/rk2206/sample` 路径下 `BUILD.gn` 文件，指定 `c7_e53_intelligent_smoke_sensor` 参与编译。

```
"/c7_e53_intelligent_smoke_sensor:e53_iss_example",
```

修改 `device/lockzhiner/rk2206/sdk_liteos` 路径下 `Makefile` 文件，添加 `-le53_iss_example` 参与编译。

```
hardware_LIBS = -lhal_iohardware -lhardware -le53_iss_example
```

## 5.编译固件

```
hb set -root .
```

```
hb set
```

```
hb build -f
```

# 如何创建、申请、释放互斥锁

6. 烧写固件

7. 通过串口查看结果

运行结果

```
=====
*****e53_iss_example*****
=====
ppm:20.3
=====
*****e53_iss_example*****
=====
ppm:20.9
.....
```



# 谢谢聆听

单击此处添加副标题内容