HiSpark-WiFi-IoT 蜂鸣器功能编程指南

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 开发板蜂鸣器功能实现

1.1 蜂鸣器硬件准备

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 2 个蜂鸣器，分别位于智慧红绿灯模块和智能环境监测模块。

1. 智慧红绿灯模块上的蜂鸣器作用：在选择启用蜂鸣器功能后，在绿灯亮时会长周期的响，红灯或黄灯亮时会短周期的响。

图 1.1-1

2. 智能环境监测模块上的蜂鸣器作用：当温度湿度或者可燃气体弄到达到阈值时，蜂鸣器响起。

图 1.1-2

1.2 蜂鸣器硬件介绍

智慧红绿灯模块和智能环境监测模块上的蜂鸣器共用一个 GPIO9 PIN 口 ，采用 PWM 复用输出。

当 IO 口为高电平时，蜂鸣器响；当 IO 口为低电平时，蜂鸣器不响。

1.3 蜂鸣器功能软件实现

1. LiteOS 打开 WIFI-IOT 的 SDK 后，首先初始化 GPIO 配置

hi_u32 hi_io_set_func(hi_io_name id, hi_u8 val);

hi_u32 hi_gpio_set_dir(hi_gpio_idx id, hi_gpio_dir dir);

hi_u32 hi_pwm_init(hi_pwm_port port);

hi_u32 hi_pwm_set_clock(hi_pwm_clk_source clk_type);

调用以上进行 GPIO 口配置；

所使用接口参数含义如下：

hi_io_name id：gpio 硬件管脚编号，对应的是硬件管脚上的 GPIO pin 脚，枚举类型如下：

```
typedef enum {
    HI_IO_NAME_GPIO_0,      /**< GPIO0 */
    HI_IO_NAME_GPIO_1,      /**< GPIO1 */
    HI_IO_NAME_GPIO_2,      /**< GPIO2 */
    HI_IO_NAME_GPIO_3,      /**< GPIO3 */
    HI_IO_NAME_GPIO_4,      /**< GPIO4 */
    HI_IO_NAME_GPIO_5,      /**< GPIO5 */
    HI_IO_NAME_GPIO_6,      /**< GPIO6 */
    HI_IO_NAME_GPIO_7,      /**< GPIO7 */
    HI_IO_NAME_GPIO_8,      /**< GPIO8 */
    HI_IO_NAME_GPIO_9,      /**< GPIO9 */
    HI_IO_NAME_GPIO_10,     /**< GPIO10 */
    HI_IO_NAME_GPIO_11,     /**< GPIO11 */
    HI_IO_NAME_GPIO_12,     /**< GPIO12 */
    HI_IO_NAME_GPIO_13,     /**< GPIO13 */
    HI_IO_NAME_GPIO_14,     /**< GPIO14 */
    HI_IO_NAME_SFC_CSN,      /**< SFC_CSN */
    HI_IO_NAME_SFC_IO1,     /**< SFC_IO1 */
    HI_IO_NAME_SFC_IO2,     /**< SFC_IO2 */
```

```
    HI_IO_NAME_SFC_IO0,      /**< SFC_IO0 */
    HI_IO_NAME_SFC_CLK,      /**< SFC_CLK */
    HI_IO_NAME_SFC_IO3,      /**< SFC_IO3 */
    HI_IO_NAME_MAX,
} hi_io_name;
```

hi_u8 val :对应 GPIO 引脚功能，如果该引脚复用，可以配置为其他的复用功能，如：UART/PWM/I2C/SPI/SDIO 等功能；

hi_gpio_idx idx：选择 GPIO 引脚，枚举类型如下：

```
typedef enum {
    HI_GPIO_IDX_0,       /**< GPIO0*/
    HI_GPIO_IDX_1,       /**< GPIO1*/
    HI_GPIO_IDX_2,       /**< GPIO2*/
    HI_GPIO_IDX_3,       /**< GPIO3*/
    HI_GPIO_IDX_4,       /**< GPIO4*/
    HI_GPIO_IDX_5,       /**< GPIO5*/
    HI_GPIO_IDX_6,       /**< GPIO6*/
    HI_GPIO_IDX_7,       /**< GPIO7*/
    HI_GPIO_IDX_8,       /**< GPIO8*/
    HI_GPIO_IDX_9,       /**< GPIO9*/
    HI_GPIO_IDX_10,      /**< GPIO10*/
    HI_GPIO_IDX_11,      /**< GPIO11*/
    HI_GPIO_IDX_12,      /**< GPIO12*/
    HI_GPIO_IDX_13,      /**< GPIO13*/
    HI_GPIO_IDX_14,      /**< GPIO14*/
    HI_GPIO_IDX_MAX,     /**< Maximum value, which cannot be used.CNcomment:最大值，不可输入使用 CNend*/
} hi_gpio_idx;
```

hi_gpio_dir dir:将选择的 GPIO 引脚设置方向，设置 GPIO 输入/输出；枚举类型如下：

```
typedef enum {
    HI_GPIO_DIR_IN = 0,          /**< Input.CNcomment:输入方向 CNend*/
    HI_GPIO_DIR_OUT              /**< Output.CNcomment:输出方向 CNend*/
} hi_gpio_dir;
```

hi_pwm_port: 选择对应的 PWM 端口初始化，枚举类型如下：

typedef enum {

HI_PWM_PORT_PWM0 = 0,

HI_PWM_PORT_PWM1 = 1,

HI_PWM_PORT_PWM2 = 2,

HI_PWM_PORT_PWM3 = 3,

HI_PWM_PORT_PWM4 = 4,

HI_PWM_PORT_PWM5 = 5,

HI_PWM_PORT_MAX

} hi_pwm_port;

不同 IO 口对应的 PWM 端口不同，需选择对应端口。

hi_pwm_clk_source clk_type：时钟源类型，取值范围为 hi_pwm_clk_source 枚举值，枚举类型如下：

typedef enum {

/*160M 工作时钟*/

PWM_CLK_160M,

/*24M 或 40M 晶体时钟*/

PWM_CLK_XTAL,

/*最大值，不可使用*/

PWM_CLK_MAX

} hi_pwm_clk_source;

例如：蜂鸣器初始化配置如下：

函数调用：

图 1.3-1 蜂鸣器 GPIO 初始化

```
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);
```

2.蜂鸣器通过 PWM 输出函数发出声音

PWM 输出函数：hi_u32 hi_pwm_start(hi_pwm_port port, hi_u16 duty, hi_u16 freq);

此函数为启动 PWM 信号输出。其中，各参数的含义为：

hi_pwm_port port：PWM 端口号

hi_u16 duty：占空比计数值（此值可自己设置）

hi_u16 freq：分频倍数（此值可自己设置）

信号占空比为：duty/freq，占空比越大，蜂鸣器叫声越大。

PWM 频率为：时钟源频率/freq。

例如：蜂鸣器使用案例

图 1.3-2

```
if (beep == BEEP_ON) {
    hi_pwm_start(HI_PWM_PORT_PWM0, PWM_DUTY, PWM_FULL_DUTY); //开启蜂鸣器
    beep = BEEP_OFF;
} else {
    hi_pwm_start(HI_PWM_PORT_PWM0, PWM_LOW_DUTY, PWM_FULL_DUTY); // 关闭蜂鸣器
    beep = BEEP_ON;
}
```