BK3432-BLE SDK

库

函

数

介

绍

V1.0

2018.12.21

深圳市集贤科技有限公司

T:0755-82571152 F:0755-88373753

<http://www.uascent.com>

深圳市南山区朗山路同方信息港A栋4楼

**目录**

[1. 版本记录及免责申明和版权公告 4](#_Toc533605242)

[1.1 版本记录 4](#_Toc533605243)

[1.2 免责申明和版权公告 4](#_Toc533605244)

[（一）GPIO API接口 5](#_Toc533605245)

[gpio\_config 5](#_Toc533605246)

[gpio\_get\_input 5](#_Toc533605247)

[gpio\_set 5](#_Toc533605248)

[（一）PWM API接口 5](#_Toc533605249)

[pwm\_init 5](#_Toc533605250)

[pwm\_isr 6](#_Toc533605251)

[IRQ\_Exception 6](#_Toc533605252)

[（一）软件定时器 API接口 6](#_Toc533605253)

[ke\_timer\_set 6](#_Toc533605254)

[ke\_timer\_clear 6](#_Toc533605255)

[（一）ADC API接口 7](#_Toc533605256)

[adc\_init 7](#_Toc533605257)

[adc\_isr 7](#_Toc533605258)

[（一）Flash API接口 7](#_Toc533605259)

[flash\_init 7](#_Toc533605260)

[flash\_erase\_sector 7](#_Toc533605261)

[flash\_erase 7](#_Toc533605262)

[flash\_writeKey 7](#_Toc533605263)

[flash\_unlock 8](#_Toc533605264)

[flash\_start 8](#_Toc533605265)

[flash\_status 8](#_Toc533605266)

[flash\_clearKey 8](#_Toc533605267)

[flash\_write 8](#_Toc533605268)

[flash\_read 8](#_Toc533605269)

[write\_lmecc\_pointq 8](#_Toc533605270)

[read\_lmecc\_pointq\_status 9](#_Toc533605271)

[read\_lmecc\_pointq 9](#_Toc533605272)

[crc16 9](#_Toc533605273)

[（一）软件定时器 API接口 9](#_Toc533605274)

[ke\_timer\_set 9](#_Toc533605275)

[ke\_timer\_clear 9](#_Toc533605276)

[（一）模拟IIC API接口 9](#_Toc533605277)

[iic\_init 9](#_Toc533605278)

[iic\_start 10](#_Toc533605279)

[iic\_stop 10](#_Toc533605280)

[iic\_ack 10](#_Toc533605281)

[iic\_no\_ack 10](#_Toc533605282)

[iic\_waitask 10](#_Toc533605283)

[iic\_tx\_byte 10](#_Toc533605284)

[iic\_tx\_byte 10](#_Toc533605285)

[iic\_tx\_data 10](#_Toc533605286)

[iic\_tx\_data 11](#_Toc533605287)

[（一）硬件IIC API接口 11](#_Toc533605288)

[i2c\_init 11](#_Toc533605289)

[i2c\_send\_start 11](#_Toc533605290)

[i2c\_send\_addr 11](#_Toc533605291)

[i2c\_msg\_reset 11](#_Toc533605292)

[is\_i2c\_busy 11](#_Toc533605293)

[i2c\_get\_last\_msg 11](#_Toc533605294)

[i2c\_msg\_init 11](#_Toc533605295)

[i2c\_write 12](#_Toc533605296)

[i2c\_read 12](#_Toc533605297)

[adc\_isr 12](#_Toc533605298)

# 版本记录及免责申明和版权公告

## 1.1 版本记录

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Date | Author | Description |
| V1.0 | 2018/12/26 | Eatun | 初始版本 |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

## 1.2 免责申明和版权公告

本文档中所有信息均按产品现状提供，如有变更，恕不另行通知。

本文档内容不负任何担保责任，包括对适销性、适用于特定用途或非侵权性的任何担保，和任何提案、规格或样品在他处提到的任何担保。本文档不对使用本文档内信息产生的任何侵犯专利权的行为负责。本文档在此未以禁止反言或其他方式授予任何知识产权使⽤许可，不管是明示许可还是暗示许可。

文中提到的所有商标均属其各自所有者的财产，特此声明。

**注 意**

由于产品版本升级或其他原因，本手册内容有可能变更，深圳市集贤科技有限公司保留在没有任何通知或者提示的情况下对本手册的内容进行修改的权利，使用者如需获取最新产品信息，请与本公司申请最终文档。本手册仅作为使用指导，深圳市集贤科技有限公司尽力在本手册中提供最新的信息，但不确保手册内容完全准确。本手册中的所有陈述、信息和建议也不构成任何明示或暗示的担保。

**一 概述**

本文档旨在为用户

**二 应用程序接口（API）**

## （一）GPIO API接口

### gpio\_config

|  |
| --- |
| 功能: GPIO口输入输出属性设置  函数定义 : void gpio\_config(uint8\_t gpio, Dir\_Type dir, Pull\_Type pull)  参数:  uint8\_t gpio ：端口号 ，GPIOA对应GPIO0，GPIOB对应GPIO1，GPIOC对应GPIO2，GPIOD对应GPIO3  Dir\_Type dir ：io口状态，OUTPUT：输出 ，INPUT：输入  Pull\_Type pull：是否上下拉。PULL\_HIGH：上拉使能，PULL\_LOW：下拉使能 ，PULL\_NONE：禁止上下拉  返回: ⽆ |

举例：

PB0设置为输出模式：gpio\_config(GPIOB\_0, OUTPUT, PULL\_NONE);

PB2设置为输入上拉: gpio\_config(GPIOB\_2, INPUT, PULL\_HIGH);

### gpio\_get\_input

|  |
| --- |
| 功能: GPIO 口状态获取  函数定义 : uint8\_t gpio\_get\_input(uint8\_t gpio)  参数:  uint8\_t gpio ：GPIO端口号 ，GPIOA对应GPIO0，GPIOB对应GPIO1，GPIOC对应GPIO2，GPIOD对应GPIO3  返回: 对应GPIO的具体值（0或者1） |

举例：读取PB2的数值，gpio\_get\_input(GPIOB\_2);

### gpio\_set

|  |
| --- |
| 功能: 设置GPIO 口输出电平的状态  函数定义 : void gpio\_set(uint8\_t gpio, uint8\_t val)  参数:  uint8\_t gpio ：GPIO端口号 ，GPIOA对应GPIO0，GPIOB对应GPIO1，GPIOC对应GPIO2，GPIOD对应GPIO3  uint8\_t val ：为1，输出高电平， 为0，输出低电平。  返回: 无 |

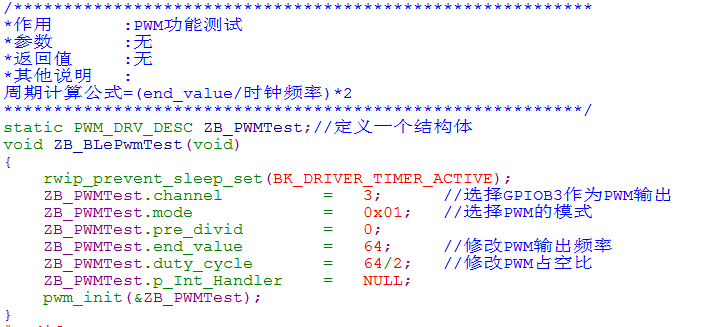
举例：读取PB2输出高电平，gpio\_set(GPIOB\_2,1);

## （一）PWM API接口

### pwm\_init

|  |
| --- |
| 功能: 初始化PWM  函数定义 : void pwm\_init(PWM\_DRV\_DESC \*pwm\_drv\_desc)  参数:  PWM\_DRV\_DESC \*pwm\_drv\_desc：初始化PWM参数的结构体  返回: ⽆ |

举例：



### pwm\_isr

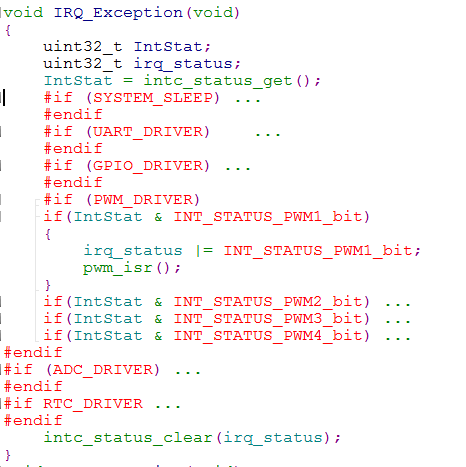
|  |
| --- |
| 功能: 处理PWM中断事件  函数定义 : void pwm\_isr(void)  参数: 无  返回: 无 |

举例：被中断入口函数调用。

### IRQ\_Exception

|  |
| --- |
| 功能: 调用中断处理函数  函数定义 : void IRQ\_Exception(void)  参数: 无  返回: 无 |

举例：



## （一）软件定时器 API接口

### ke\_timer\_set

|  |
| --- |
| 功能: 10ms定时基准函数  函数定义 : void ke\_timer\_set(ke\_msg\_id\_t const timer\_id, ke\_task\_id\_t const task, uint32\_t delay)  参数:  timer\_id 定时器消息.  task 任务标识  delay 定时时间，10ms为单位  返回: ⽆ |

举例：

ke\_timer\_set(APP\_TIMER\_TEST,TASK\_APP ,50)： 50\*10ms回调一次

### ke\_timer\_clear

|  |
| --- |
| 功能: 清除软件定时器  函数定义 : ke\_timer\_clear(ke\_msg\_id\_t const timerid, ke\_task\_id\_t const task);  参数:  timer\_id 定时器消息.  task 任务标识  返回: 无 |

举例：ke\_timer\_clear (APP\_TIMER\_TEST,TASK\_APP ,50)

## （一）ADC API接口

### adc\_init

|  |
| --- |
| 作用 : 初始化ADC  函数定义 : void adc\_init(uint8\_t chanle,uint8\_t mode)  参数 :  Chanle : 选择ADC通道  Mode : ADC模式选择 00:休眠模式01:单步模式10:软件控制11:连续模式  返回值 : 无 |

举例：adc\_init(3，1)： 单步模式读取GPIOC3口的ADC值

### adc\_isr

|  |
| --- |
| 功能: 处理ADC中断事件  函数定义 : void adc\_isr(void)  参数: 无  返回: 无 |

举例：被中断入口函数调用。

## （一）Flash API接口

### flash\_init

|  |
| --- |
| 作用 : 初始化flash  函数定义 : void flash\_init(void)  参数 : 无  返回值 : 无 |

举例：

### flash\_erase\_sector

|  |
| --- |
| 功能: 擦除扇区  函数定义 : void flash\_erase\_sector(uint8\_t type,uint32\_t address)  参数 :  Type : 选择要擦除的flash类型 包括MAIN NVR。  address : 擦除扇区的首地址  返回值 : 无 |

举例：flash\_erase\_sector(FLASH\_SPACE\_TYPE\_NVR,0x8000);擦除NVR区的0x8000地址所在的扇区。

### flash\_erase

|  |
| --- |
| 功能: 擦除扇区  函数定义 : void flash\_erase(uint8\_t flash\_type, uint32\_t address, uint32\_t size)  参数 :  flash\_type : 选择要擦除的flash类型 包括MAIN NVR。  address : 擦除的首地址  size ：擦除flash的长度  返回值 : 无 |

举例：flash\_erase (FLASH\_SPACE\_TYPE\_NVR,0x8000，60);擦除NVR区的0x8000地址开始的60个字节长度的区域。

### flash\_writeKey

|  |
| --- |
| 作用 : 使能flash操作  函数定义 : static void flash\_writeKey(void)  参数 : 无  返回值 : 无 |

### flash\_unlock

|  |
| --- |
| 作用 : flash写保护使能  函数定义 : static void flash\_unlock(void)  参数 : 无  返回值 : 无 |

### flash\_start

|  |
| --- |
| 作用 : 开始操作flash  函数定义 : static void flash\_start(void)  参数 : 无  返回值 : 无 |

### flash\_status

|  |
| --- |
| 作用 : 判断操作flash是否完成  函数定义 : static uint8\_t flash\_status(void)  参数 : 无  返回值 : 返回寄存器0x00803004 bit0的状态，该位flash操作完成后硬件自动清零 |

### flash\_clearKey

|  |
| --- |
| 作用 : 关闭flash操作使能  函数定义 : static void flash\_clearKey(void)  参数 : 无  返回值 : 无 |

### flash\_write

|  |
| --- |
| 作用 : 关闭flash操作使能  函数定义 : void flash\_write(uint8\_t flash\_space, uint32\_t address, uint32\_t length, uint8\_t \*buffer)  参数 :  flash\_space : flash类型。 flash类型:NVR 和MAIN  address : 写入数据的首地址  length : 写入数据的长度  buffer : 写入的数据  返回值 : 无 |

举例：flash\_write(FLASH\_SPACE\_TYPE\_NVR,0x8000,60,temp\_buff);往NVR区域写入60个字节数据，数据存放在temp\_buff，首地址：0x8000

### flash\_read

|  |
| --- |
| 作用 : 关闭flash操作使能  函数定义 : void flash\_read(uint8\_t flash\_space, uint32\_t address, uint32\_t length, uint8\_t \*buffer)  参数 :  flash\_space : flash类型。 flash类型:NVR 和MAIN  address : 读出数据的首地址  length : 读出数据的长度  buffer : 读出的数据  返回值 : 无 |

举例：flash\_read(FLASH\_SPACE\_TYPE\_NVR,0x8000,60,temp\_buff);在NVR区域读出60个字节数据，数据存放在temp\_buff，首地址：0x8000

### write\_lmecc\_pointq

|  |
| --- |
| 作用 : 往flash连续写入数据  函数定义 : void write\_lmecc\_pointq(uint8\_t \*buff1, uint8\_t \*buff2)  参数 :  buff1 : 写入的数据  Buff2 : 写入的数据  返回值 : 无 |

举例：write\_lmecc\_pointq(&writedata\_one[0],&writedata\_one[28]);将writedata\_one数组里面的数据写入flash

### read\_lmecc\_pointq\_status

|  |
| --- |
| 作用 : 对写进去的数据和读出来的数据进行校验  函数定义 : uint8\_t read\_lmecc\_pointq\_status(void)  参数 : 无  返回值 : 校验结果 校验正确返回 1 校验错误返回 0 |

### read\_lmecc\_pointq

|  |
| --- |
| 作用 : 读出写入flash的数据  函数定义 : void read\_lmecc\_pointq( uint8\_t \*buff1,uint8\_t \*buff2 )  参数 :  buff1 : 读出的数据  Buff2 : 读出的数据  返回值 : 无 |

举例：read\_lmecc\_pointq(&readdata\_one[0],&readdata\_one[30]);读出flash里面的数据，放入数组readdata\_one数组里面。

### crc16

|  |
| --- |
| 作用 : 16为crc校验  函数定义 : uint16\_t crc16 (uint8\_t \*pbuff, uint32\_t len)  参数 :  pbuff : 校验数据  len : 数据长度  返回值 : 无 |

举例：crc16(temp\_buff,58);对temp\_buff里面的数据进行校验

## （一）软件定时器 API接口

### ke\_timer\_set

|  |
| --- |
| 功能: 10ms定时基准函数  函数定义 : void ke\_timer\_set(ke\_msg\_id\_t const timer\_id, ke\_task\_id\_t const task, uint32\_t delay)  参数:  timer\_id 定时器消息.  task 任务标识  delay 定时时间，10ms为单位  返回: ⽆ |

举例：ke\_timer\_set(APP\_TIMER\_TEST,TASK\_APP ,50)： 50\*10ms回调一次

### ke\_timer\_clear

|  |
| --- |
| 功能: 清除软件定时器  函数定义 : ke\_timer\_clear(ke\_msg\_id\_t const timerid, ke\_task\_id\_t const task);  参数:  timer\_id 定时器消息.  task 任务标识  返回: 无 |

举例：ke\_timer\_clear (APP\_TIMER\_TEST,TASK\_APP ,50)

## （一）模拟IIC API接口

### iic\_init

|  |
| --- |
| 作用 : 初始化iic  函数定义 : void iic\_init()  参数 : 无  返回值 : 无 |

### iic\_start

|  |
| --- |
| 功能 : 模拟iic起始信号函数  函数定义 : void iic\_start()  参数 : 无  返回 : 无 |

### iic\_stop

|  |
| --- |
| 功能 : 模拟iic停止信号函数  函数定义 : void iic\_stop()  参数 : 无  返回 : 无 |

### iic\_ack

|  |
| --- |
| 功能 : 模拟iic应答信号函数  函数定义 : void iic\_ack(void)  参数 : 无  返回 : 无 |

### iic\_no\_ack

|  |
| --- |
| 功能 : 模拟iic不应答信号函数  函数定义 : void iic\_no\_ack(void)  参数 : 无  返回 : 无 |

### iic\_waitask

|  |
| --- |
| 功能 : CPU产生一个时钟，并读取器件的ACK应答信号  函数定义 : uint8\_t iic\_waitask(void)  参数 : 无  返回 : 返回0表示正确应答，1表示无器件响应 |

### iic\_tx\_byte

|  |
| --- |
| 功能 : 写一个字节数据  函数定义 : void iic\_tx\_byte(uint8\_t dat)  参数 : dat：写入的数据  返回 : 无 |

### iic\_tx\_byte

|  |
| --- |
| 功能 : 读一个字节数据  函数定义 : uint8\_t iic\_rx\_byte()  参数 :  返回 : 读出的数据 |

### iic\_tx\_data

|  |
| --- |
| 功能 : 往AT24C02写数据  函数定义 : void iic\_tx\_data(uint8\_t devAddr,uint8\_t addr,uint8\_t\*buf,uint8\_t size)  参数 :  devAddr：从机地址  addr ：数据写入的首地址  buf ：写入的数据  size ：写入数据的数量  返回 : |

举例：iic\_tx\_data(0Xa0, 0X01,&writebuf[1],1);往AT24C02的0x01地址写入一个数据

### iic\_tx\_data

|  |
| --- |
| 功能 : 往AT24C02读数据  函数定义 : void iic\_rx\_data(uint8\_t devAddr,uint8\_t addr,uint8\_t\*buf,uint8\_t size)  参数 :  devAddr：从机地址  addr ：读出数据的首地址  buf ：读出的数据  size ：读出数据的数量  返回 : |

举例：iic\_tx\_data(0Xa0, 0X01,&writebuf[1],1);往AT24C02的0x01地址读出一个数据

## （一）硬件IIC API接口

### i2c\_init

|  |
| --- |
| 作用 : 初始化硬件iic  函数定义 : void i2c\_init(uint32\_t slaveAddr, uint32\_t baudRate)  参数 :  slaveAddr:从机地址  baudRate :分频 选择0  返回值 : 无 |

### i2c\_send\_start

|  |
| --- |
| 功能 :发送 iic起始信号，只支持主模式  函数定义 : static void i2c\_send\_start(void)  参数 : 无  返回 : 无 |

### i2c\_send\_addr

|  |
| --- |
| 功能 : 发送从机地址，只支持主模式  函数定义 : static void i2c\_send\_addr(void)  参数 : 无  返回 : 无 |

### i2c\_msg\_reset

|  |
| --- |
| 功能 : 重置i2c消息，必须确保最后一条i2c消息Tx/Rx已经完成  函数定义 : static void i2c\_msg\_reset(void)  参数 : 无  返回 : 无 |

### is\_i2c\_busy

|  |
| --- |
| 功能 : CPU产生一个时钟，并读取器件的ACK应答信号  函数定义 : static ASK is\_i2c\_busy(void)  参数 : 无  返回 : 返回0 iic是空闲状态 返回1 iic是忙碌状态 |

### i2c\_get\_last\_msg

|  |
| --- |
| 功能 : 获取i2c最后一个消息，以便您可以检查错误或处理接收数据  函数定义 : static I2C\_MSG \* i2c\_get\_last\_msg(void)  参数 : 无  返回 : 读写正确返回i2c最后一个消息 读写错误返回NULL |

### i2c\_msg\_init

|  |
| --- |
| 功能 : 初始化i2c消息，准备读/写数据。调用函数i2c\_send\_start()启动读/写数据  函数定义 : static STATUS i2c\_msg\_init(I2C\_MSG \*p\_i2c\_msg)  参数 : 信息描述  返回 : 返回 1 表示初始化信息正确 可以开始读/写数据 返回0 表示初始化信息错误 不继续开始读/写数据 |

### i2c\_write

|  |
| --- |
| 功能 : 往AT24C02写数据  函数定义 : STATUS i2c\_write(uint8\_t devAddr, uint8\_t addr, uint8\_t\*buf, uint8\_t size)  参数 :  devAddr：从机地址  addr ：数据写入的首地址  buf ：写入的数据  size ：写入数据的数量  返回 : 写入成功返回 1 写入失败返回 0 |

举例：i2c\_write(0x50,0x00,&iic\_writebuf[1],1);往AT24C02的0x00地址写入一个数据

### i2c\_read

|  |
| --- |
| 功能 : 往AT24C02读数据  函数定义 : STATUS i2c\_read(uint8\_t devAddr, uint8\_t addr, uint8\_t\*buf, uint8\_t size)  参数 :  devAddr：从机地址  addr ：读出数据的首地址  buf ：读出的数据  size ：读出数据的数量  返回 : 读出成功返回 1 读出失败返回 0 |

举例：i2c\_read(0x50,0x00,&iic\_readbuf[0],1);往AT24C02的0x00地址读出一个数据

### i2c\_isr

|  |
| --- |
| 功能: 处理硬件iic中断事件  函数定义 : void i2c\_isr(void)  参数: 无  返回: 无 |

举例：被中断入口函数调用。

## （一）按键睡眠唤醒

### rwip\_sleep

|  |
| --- |
| 功能: 选择睡眠模式  函数定义 : uint8\_t rwip\_sleep(void)  参数: 无  返回: 返回 1 进入低速广播模式 返回3 进入低功耗模式 |

### cpu\_reduce\_voltage\_sleep

|  |
| --- |
| 功能: ARM9 Core时钟关闭和芯片进入低电压模式  函数定义 : void cpu\_reduce\_voltage\_sleep(void)  参数: 无  返回: 无 |

### cpu\_idle\_sleep

|  |
| --- |
| 功能: ARM9 Core时钟关闭  函数定义 : void cpu\_idle\_sleep(void)  参数: 无  返回: 无 |

### gpio\_isr

|  |
| --- |
| 功能: 处理硬件gpio中断事件  函数定义 : void gpio\_isr(void)  参数: 无  返回: 无 |

举例：被中断入口函数调用。