中科蓝讯
bluetrum

电子烟烟控方案说明

版本号：0.0.1

声 明

本文档是中科蓝讯的原创作品和受版权保护的财产。全部或部分复制使用或传播必须事先获得中科蓝讯的书面批准，并经版权所有人明确确认。中科蓝讯有权随时根据法律、法规的变化以及公司经营策略的调整等修改本文档。修改后的文档将会通过适当的方式将进行公示。如您在本文档修订后仍继续使用本文档内容的，则视为您接受本文档的修订。

请您通过各种方式关注中科蓝讯发布的信息，包括中科蓝讯的官方网站、官方公众号等。中科蓝讯对不当使用本文档的后果不承担任何责任，中科蓝讯提供的信息仅作为参考或典型应用。中科蓝讯保留更改电路设计的权利和/或规格的权利，无需另行事先通知。

您不得因用途原因侵犯第三方的专利或其他权利，否则应自行承担相应责任。实施解决方案/产品可能需要第三方许可证，您应全权负责获取所有适当要求的第三方许可证；中科蓝讯不负责任何所需第三方许可证的任何许可费或版税。

如果您需要了解进一步的业务和技术支持，请发邮箱至：

sales@bluetrum.com/project@bluetrum.co

修 订 历 史

修订日期	版本号	修订记录	修订人
2024-8-1	V0.0.1	初版	张李昱、蒋奇锋

目录

1 名词释义

单 MOS 硬件方案即下图 1-1 所示，双 MOS 硬件方案如下图 1-2 所示；

RL：加热丝阻值；

RM：电池内阻+MOS 内阻；

VBAT：电池电压；

VAT：加热丝电压；

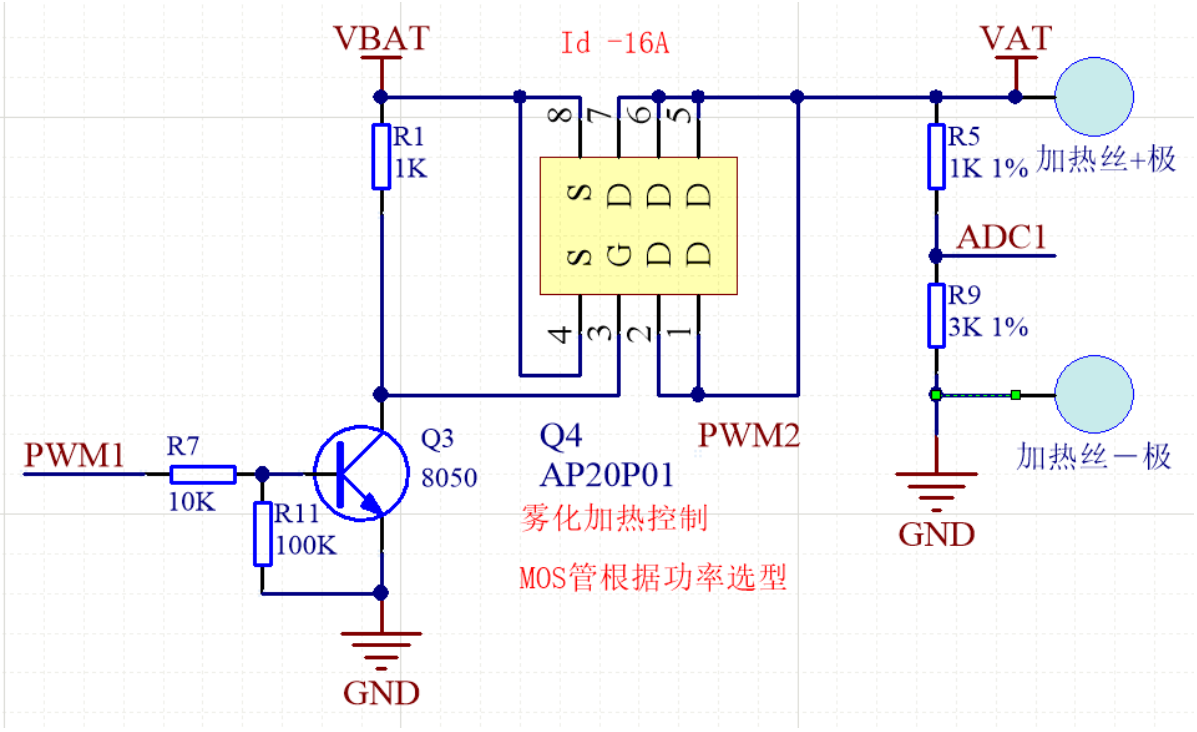


图 1-1 单 MOS 硬件方案

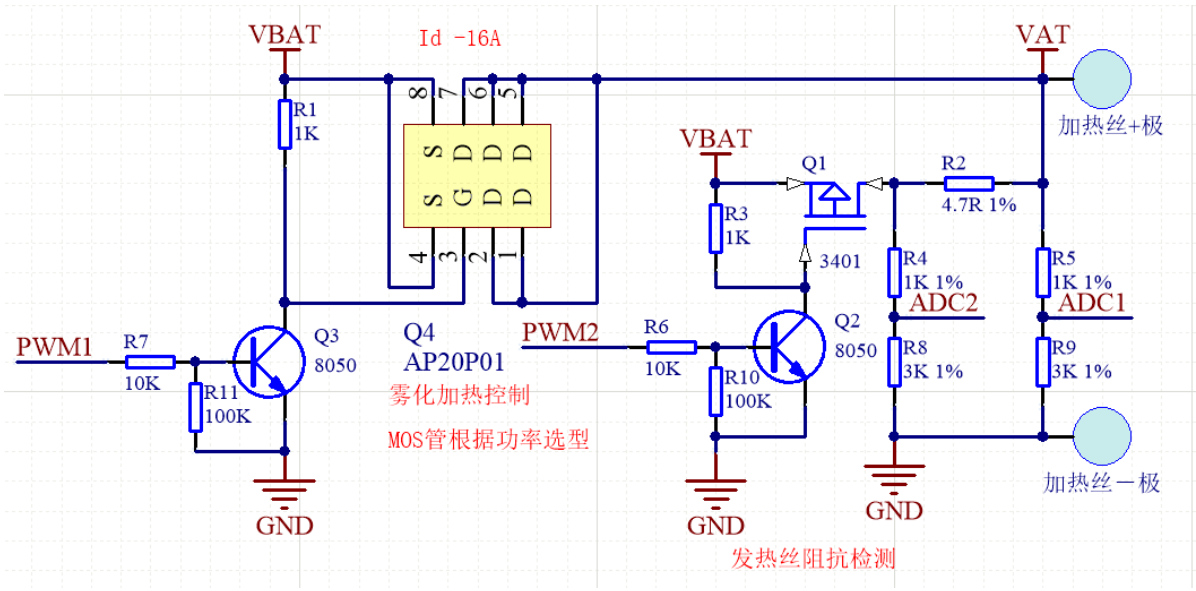


图 1-1 双 MOS 硬件方案

2 软件宏释义

电子烟配置代码示例如图 2-1 所示：

ECIG_POWER_CONTRL：1→使能，0→不使能，默认配置 1，电子烟加热方式为恒功率加热，打开宏默认会在 bsp_sys_init（）里初始化烟控使用的 GPIO、ADC 通道、定时器以及烟控相关参数。

ECIG_ADC2_ENABLE：1→使能，0→不使能，默认配置 0，硬件设计为双 MOS 方案需打开宏。

```
/*******************************************************************************
 * Module    : 电子烟配置
 *******************************************************************************/
#define LCD_DISPLAY_EN                  1           //LCD屏幕总使能
#define LCD_TEST_EN                     1           //LCD测试使能
#define ECIG_POWER_CONTRL               1           //电子烟恒功率控制
#define ECIG_ADC2_ENABLE                0           //电子烟是否有ADC2通路
```

图 2-1 电子烟配置

电子烟参数配置代码示例如图 1-2 所示：

Final_power：加热目标功率，单位（W），例设 8 即 8W。

Max_hot_time：最长加热时间，单位（S），例设 8 即 8S。

Min_proportion：加热丝短路设定值，加热丝与（电池内阻+MOS 内阻）的最小比例。

Max_proportion：加热丝开路设置值，加热丝与（电池内阻+MOS 内阻）的最大比例。

Res19：电阻丝定点值，双 MOS 硬件方案精准检测加热丝阻值需用到。

Res_cal_prev：加热丝阻值定点，单 MOS 方案需手动填加热丝阻值，例加热丝阻值 0.6R 就设 0.6。

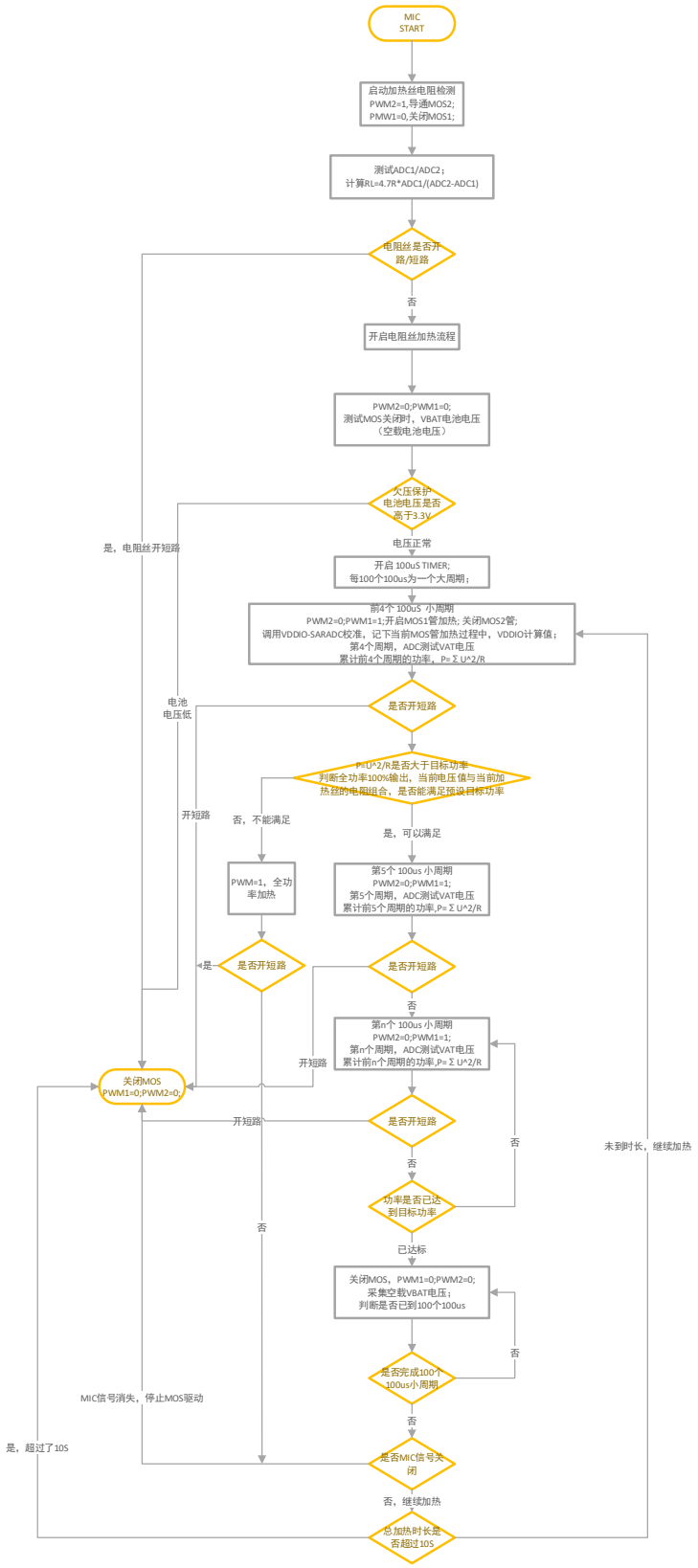
```
void e_cigs_hot_init(void)
{
    printf("%s\n", __func__);
    smoke_ctrl_t *sc = &smk_ctrl;
    memset(&smk_ctrl, 0, sizeof(smoke_ctrl_t));
    e_gpio_init();
    e_saradc_hot_init();
    timer2_hot_init();
    sc->power_on_flag   = 1;
    sc->timer_100us_cnt = 0;

    sc->final_power     = 8;                    //目标功率
    sc->max_hot_time    = 8;                    //最长加热时间
    sc->min_proportion  = 10;                   //电热丝阻值和MOS内阻（130mΩ）的最小比例
    sc->max_proportion  = 153;                  //电热丝阻值和MOS内阻（130mΩ）的最大比例
    sc->res19           = (u16)(4.7 * 8192);    //电阻定点值(双MOS测电阻是需要用到)
    sc->res_cal_prev    = (u16)(0.6 * 8192);    //电热丝阻值定点
}
```

图 2-2 电子烟参数

3 烟控流程

3.1 烟控流程图



3.2 烟控周期

整个烟控流程在 10ms 完成，定时器中断周期默认设置为 100us，100us 为小周期（t），100 个小周期为一个大周期 10ms（T），定时器代码示例如图 3-1 所示。

```
void timer2_hot_init(void)
{
    printf("timer init\n");
    TMR2CON = 0;
    TMR2CNT = 1;
    TMR2PR  = 600 - 1;
    TMR2CON = (2 << 1) | (2 << 4) | BIT(7) | BIT(0);   //timer2 clk = xosc24m ; div 4
#if ECIG_ADC2_ENABLE
    sys_irq_init(IRQ_TMR2_VECTOR, 0, timer2_hot_dual_isr);
#else
    sys_irq_init(IRQ_TMR2_VECTOR, 0, timer2_hot_single_isr);
#endif
}
```

f= 24M / (4 * 600) = 10kHz
T = 1 / 10k = 100us

图 3-1 定时器配置

周期内烟控操作会按表 3-1 顺序执行，在每个 100us 这个小周期内，程序会进行采集 ADC、判断短路开路、计算功率等操作。

周期	PWM 输出状态	操作
1	低	判断电池电压 VBAT 是否欠压
2~4	高	等电压稳定后检测是否短路；采集 VAT 电压，计算功率
5	高	采集 VAT 电压，检测电阻丝是否短路、开路；计算功率；校准 VDDIO
6~n	高	采集 VAT 电压，检测电阻丝是否短路；计算功率
n~100	低	输出达到目标功率，PWM 关闭加热，检测 VBAT
101	低	完成一个 10ms 周期，清除标志位

表 3-1 100us 烟控顺序执行

3.3 烟控欠压保护

防止电池在电压低的情况下还继续工作，一般设定的电压为点火前大于 3.2 或 3.3V 能点火，低则保护，吸烟前欠不允许加热，若吸烟中欠压直接全功率加热，代码示例如图 3-2 所示。

前 4 个 100us 会开启加热，等电压稳定后采集 VAT 电压，计算是否短路，计算功率。

```
//第一个周期不开pwm，检验VBAT是否欠压
if(sc->timer_100us_cnt == 1){
    PWM1_OFF();
    if(sc->AD_BAT_voltage < 3300){                    //欠压保护
        sc->smoke_sta = LOW_POWER;
        PWM1_OFF();
        sc->smoke_sta = LOW_POWER;
        sc->power_on_flag = 0;
    }else{
        sc->smoke_sta = SMOKING;
        sc->power_on_flag = 1;
     }
}
//第2-4三个t打开pwm1，让vat稳定，再进行开路、短路检测
else if(sc->timer_100us_cnt < 5){
    PWM1_ON();
    e_saradc_hot_proc();
    sc->p_current = (u16)((calculate_power_single(sc->AD_hot_voltage) >> 13) + sc->p_prev);
    sc->p_prev = sc->p_current;
}
```

图 3-2 欠压保护

3.4 烟控 VDDIO 校准

电子烟实际应用中 VBAT 会低于 3.3V，需要反推 VDDIO 保证 SAR_ADC 采样精准，电子烟客户使用的芯片都会做 FT 校准，保证 ADC 采样在误差范围内，具体 ADC 性能及校准计算过程请看附录，代码示例如图 3-3 所示。

① SAR_ADC 10bit，参考是 VDDIO，即 Digtial 1024→VDDIO，0→0V，芯片内部有一个 VBG 电路 0.6V；

② FT 校准：机台给一个 3.3V 到 VDDIO（由于耗电，这个电压到芯片会偏低一些）。 机台输出一个 2.7V(10mV 以内偏差）给到 ADC_IO,SAR_ADC 记下当前值，存在 efuse 内。同时记下 VBG 的 SAR_ADC 值，存在 efuse 内。相当于用一个准确的 2.7V TRIM BG 的电压。

③ 常规使用的时候，调用 TRIM 的参数，因为 VDDIO 的值会不准，会使用 FT TRIM 记下来的 2.7V->Digtial 的值与 VBG->Digtial 值做参考，依赖于 BG 在不同环境下，能保持一个比较固定的模拟量。

```
AT(.com_text.saradc_proc)
void e_saradc_vddio(void)
{
    u16 vddio_cur;
    vddio_cur = 1024 * get_vbg_voltage() / adc_vbg; //adc通道读取的模拟值 乘于 标准电压 除于 电压辅助检测模拟值
    vddio_voltage = vddio_cur;
    e_saradc_kick_start(BIT(ADCCH_BGOP));
}
```

图 3-3 VDDIO 校准

3.5 烟控短路判断

单 MOS 硬件方案中，检测加热丝的短路与开路，利用到电池内阻和 MOS 内阻之和（RM）与加热丝（RL）的分压关系，将具体阻值替换为电压比例来计算，代码示例如图 3-4 所示；

$$\frac{RL}{RM}=\frac{VAT*10}{VBAT-VAT} \quad (1)$$

VAT 放大 10 倍是为了更好判断，实际应用中不同电池内阻不同，同一电池不同电压下内阻也不一样，短路判定范围要兼容不同电池不同电压下的应用，RM 是代固定值，例如 100mR,RL 短路值设为 0.3R,则 Min_proportion 短路值应设 30。

双 MOS 硬件方案中，加热前会算一下加热丝阻值，开启 PWM2，计算公式如下，分压比例一致可以直接用 ADC 采的 Digtial 算 RL，消除了分压电阻的误差；

$$RL=\frac{ADC_2*R_{2(4.7R)}}{ADC_2-ADC_1} \quad (2)$$

若短路则不会开启加热，加热中不会用双路 MOS 判断短路，会影响功率计算，还是用单路 MOS 判断短路。

```
//第5个t用于计算，判断是否开路、短路、欠压（不同于vbat欠压，这里是pwm一直拉高也无法达到目标功率即判断欠压）
else if(sc->timer_100us_cnt == 5){
    if(sc->AD_hot_voltage_mv*10/(sc->AD_BAT_voltage - sc->AD_hot_voltage_mv) <= sc->min_proportion){          //短路保护
        sc->short_circuit_cnt++;
    }else if(sc->AD_hot_voltage_mv*10/(sc->AD_BAT_voltage - sc->AD_hot_voltage_mv) >= sc->max_proportion){    //开路保护
        PWM1_OFF();
        sc->smoke_sta = OPEN_CIRCUIT;
        sc->power_on_flag = 0;
    }else{
        sc->smoke_sta = SMOKING;
        sc->power_on_flag = 1;
    }
```

图 3-4 短路判断

3.6 烟控开路判断

单 MOS 硬件方案中，由公式（1），若加热丝开路或者加热丝阻值变大 VAT 会增大，RL/RM 比例会增大，例 RM 为 100mR，假设 2R 短路，则 Max_proportion 开路设定值应为 200，代码示例如图 3-5 所示。

双 MOS 硬件方案中，加热前会检测一下加热丝阻值是否在范围内，不在设定范围内不会开启加热。

```
//第5个t用于计算，判断是否开路、短路、欠压（不同于vbat欠压，这里是pwm一直拉高也无法达到目标功率即判断欠压）
else if(sc->timer_100us_cnt == 5){
    if(sc->AD_hot_voltage_mv*10/(sc->AD_BAT_voltage - sc->AD_hot_voltage_mv) <= sc->min_proportion){          //短路保护
        sc->short_circuit_cnt++;
    }else if(sc->AD_hot_voltage_mv*10/(sc->AD_BAT_voltage - sc->AD_hot_voltage_mv) >= sc->max_proportion){    //开路保护
        PWM1_OFF();
        sc->smoke_sta = OPEN_CIRCUIT;
        sc->power_on_flag = 0;
    }else{
        sc->smoke_sta = SMOKING;
        sc->power_on_flag = 1;
    }
```

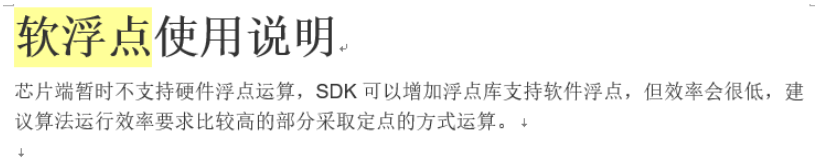
图 3-5 开路判断

3.7 烟控功率计算

3.7.1 定点计算

功率控制过程中少不了功率计算，而在大多数情况下电阻、电压以及最后计算出来的功率都是小数。使用软件浮点来进行运算的话，大概率达不到电子烟烟控的速度要求。

因此，我们的处理方式是将小数转为整数来进行计算，芯片支持硬件的 32bit 除法，可以比较快地完成功率计算。那么为什么要使用定点操作而不直接放大 100 或 1000 倍来消掉小数点呢？对一个小数进行定点时，采用的是直接移位，时效性上会比乘 100 或 1000 更好。



在功率控制代码中需要定点的是电压值和电阻值，使用的是 16 位定点，并且定义为高三位表示整数部分（即范围为 0~7），低 13 位为小数部分。

① 电阻的定点

在单路方案中，由于电阻已知，可以在宏定义中直接使用电阻值乘 2^13，这样在程序运行中不会占用中断的时间再进行计算。

而在双路方案中，吸烟信号拉高后，会用 5 个小周期（500us）计算电阻大小，后续加热过程会一直用到这个阻值，直到结束本次加热下一次吸烟信号拉高时，才会重新计算阻值。

由双 MOS 硬件原理（图 1-2）得到：

$$RL = \frac{ADC_2 * R_2}{ADC_2 - ADC_1} \tag{3}$$

在双路方案的电阻计算过程中，R_2为已知，同样地，可以在宏定义中使用电阻值乘 2^13 获得定点后的阻值，又由于ADC_2和ADC_1为整数，最终得到的电热丝电阻也为定点后的值。

② 电压的定点

电压的定点原理和电阻一致，但在代码中的操作方式不同，这是因为程序中电压值由 ADC 采集到的数字量通过计算获得，默认计算结果为 mV 为单位的整数（VDDIO 的单位是 mV）。结合分压关系不难推出：

$$V_{AT} = \frac{4}{3} * \frac{VDDIO * ADC1}{1024} \tag{4}$$

此时想要得到小数进行定点计算，需要先将V_{AT}除以 1000 得到单位为 V 的电压值，再乘 2^13（左移 13 位）。式（8）中的 1024 正好是 2^10，因此我们把 ADC 转换和定点结合，可以得到：

$$V_{AT} = \frac{4}{3} * \frac{VDDIO * ADC1}{1000} * 2^3 \tag{5}$$

程序中功率计算完成时进行了右移 13 位操作，这一步骤的作用是将功率恢复到定点前，结合式（6）和 16 位定点我们需要乘 2^13 的原理，可知U^2总共左移 26 位，RL 左移的 13 位在做除法的时候会抵消掉，因此结果右移 13 位即可得到原本的数值。

3.7.2 恒功率控制

功率控制方式为恒功率控制，恒功率的方式则烟雾量比较均匀。

首先需要明确一点，在一个周期内我们计算的功率是指平均功率，功率计算公式为：

$$P = \frac{U^2}{RL} \quad (6)$$

式中的 U 为一个周期内的平均电压，而加热过程中 ADC 采集到的电压值为瞬时电压，且为 PWM 拉高时采集到的电压，因此一个周期内的平均电压计算公式为（其中 t 为 pwm 拉高的时间，T 为整个周期）：

$$U = \frac{U_1*t_1+U_2*t_2+\cdots+U_n*t_n}{T} \quad (7)$$

如果按照先计算平均电压，再计算平均功率的话，计算效率会比较低，需要先累加再平均。因此我们采用电热丝一周期内所做有用功与目标功率在一周期所做功相等的方法计算，即：

$$P_1 * t_1 + P_2 * t_2 + \cdots P_n * t_n = P_{目标} * T \quad (8)$$

其中 t 为 PWM 拉高的时间，T 为整个周期，带入代码设置的 t=100us，T=10ms：

$$P_1 + P_2 + \cdots P_n = P_{目标} * 100 \quad (9)$$

将式（6）带入式（9）：

$$\frac{1}{RL}\left({U_1}^2 + {U_2}^2 + \cdots + {U_n}^2\right) = P_{目标} * 100 \quad (10)$$

综上，代码中只需要计算（10）中等号左边的内容，当其大于等于右边时，即可关断 PWM。

功率计算代码示例如图 3-6 所示。

```
sc->p_current = (u16)((calculate_power_single(sc->AD_hot_voltage) >> 13) + sc->p_prev);
if(sc->p_current >= sc->final_power * 100){
    PWM1_OFF();
    sc->power_on_flag = 0;
}else{
    sc->p_prev = sc->p_current;
}
```

图 3-6 功率计算

3.8 烟控时长控制

为了避免因持续加热时间过长对 MOS 和电池容易损坏，时间一般设定在 8-10s，SDK 默认配置是 8S，可更改 Max_hot_time 宏设定加热时长，代码示例如图 3-7 所示。

```
AT(.com_text.e_cigs_tmr)
void e_cigs_warn_isr(void)
{
    smoke_ctrl_t *sc = &smk_ctrl;
    if(sc->hot_time_flag){
        sc->hot_time_cnt++;
    }
    if(sc->hot_time_cnt == sc->max_hot_time){
        printf(info);
    }
}
```

bsp_sys.c

e_cig_new > app > platform > bsp > C bsp_sys.c > ...

```
    //1s timer process
    if ((sys_cb.tmr5ms_cnt % 200) == 0) {
        msg_enqueue(MSG_SYS_1S);
        sys_cb.tmr5ms_cnt = 0;
        sys_cb.lpwr_warning_cnt++;
        #if (DAC_CLASSD_VDET && VBAT_DETECT_EN)
            dac_classd_dec_proc(sys_cb.vbat);
        #endif

        #if ECIG_POWER_CONTRL
            e_cigs_warn_isr();
        #endif
    }
}
```

图 3-7 烟控时长

3.9 烟控清零

设定最长加热时长到，烟控所有标志位需清零，代码示例如图 3-8 所示，等待下一次吸烟信号来再开启烟控流程。

```
if(!mic_start_or_not()){
    sc->smoke_sta = IDLE;
    sc->timer_100us_cnt = 0;
    sc->mic_start = 0;
    sc->hot_time_cnt = 0;
    sc->hot_time_flag = 0;
}
```

图 3-8 烟控标志位清零

AB15X 芯片 ADC 性能及校准应用说明

ADC 性能

表 1 AB15X 芯片 ADC 性能

符号	参数	条件	最小值	典型值	最大值	单位
Idd	功耗	@2MHz fclk	-	0.7	-	mA
Cin	采样电容		-	5	-	pF
Fclk	转换时钟频率	VCC=1.8~3.3V	5	-	-	MHz
Tsamp	转换时间		13			us
Tconv	转换时间		-	14*Tclk	-	
Teoc	转换完读取数据时间		-	0.5*Tclk	-	
DNL			-	±1	-	LSB
INL			-	±1	±2.4	LSB

ADC FT 校准及应用

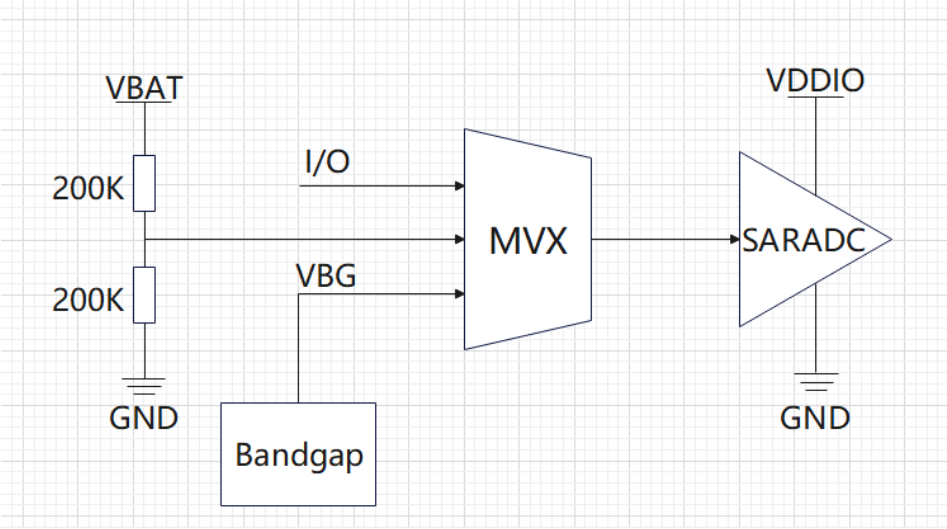


图 1 芯片内部 ADC 电路简易图示

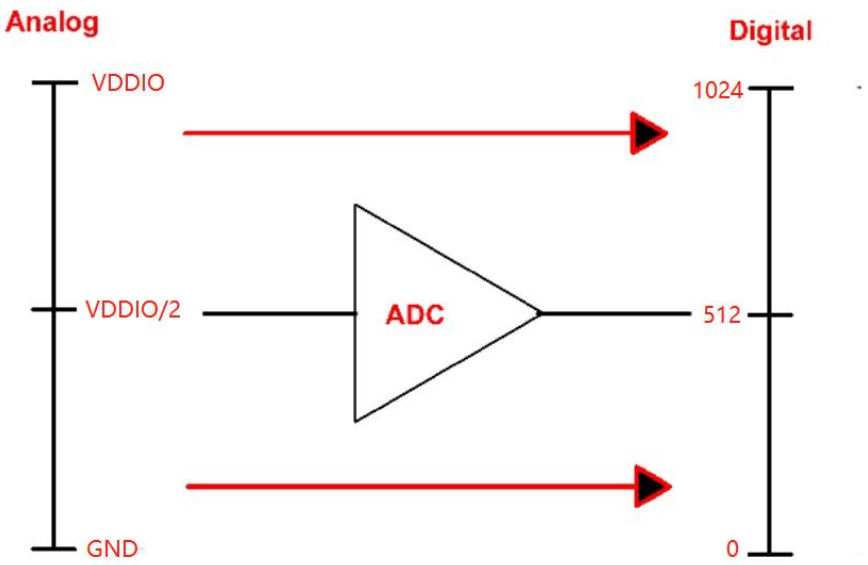


图 2 芯片模拟数字量对应关系图

AB15X

FT 环境下，机台输出一个 2.7V(10mV 以内偏差）给到 ADC_IO。SAR-ADC 记下当前值，同时记下 BG 的 SAR-ADC 值，以此推算 VBG 的采样偏差值，存在 Efuse 内，相当于用一个准确的 2.7V，TRIM BG 的电压。

常规使用的时候，调用 TRIM 的参数，因为实际使用中 VDDIO 的值会不准，会使用 FT TRIM 记下来的 VBG 的采样偏移值来推算 VBG 电压，最终推算出采样电压值，依赖于 BG 在不同环境下，能保持一个比较固定的模拟量

SDK 代码 ADC 计算

AB15X

Vbat 计算公式：

vbat = vbat2 * VBG_VOLTAGE * 2 / (1000 * bg);

说明：

vbat：经过计算后实际的电池电压值

vbat2: SARADC 模块 vbat 通路采样值

VBG_VOLTAGE：经过 FT 修正后的 VBG 电压，该值会在 600 上下浮动

bg: SARADC 模块 bg 通路采样值

IO 获取电压值计算公式：

IO_VOL = VBG_VOLTAGE / bg * IO_ADC

说明：

IO_VOL：经过计算后实际的 IO 电压值

IO_ADC：SARADC 模块此 IO 通路采样值，SDK 获取 10 位精度的采样值（调用 e_get_saradc_data(u16n)获取）

VBG_VOLTAGE：经过 FT 修正后的 VBG 电压，该值会在 600 上下浮动

bg: SARADC 模块 bg 通路采样值

AB202X 芯片 ADC 性能及校准应用说明

ADC 性能

表 1 AB202X 芯片 ADC 性能

符号	参数	条件	最小值	典型值	最大值	单位
Idd	功耗	@2MHz fclk	-	0.7	-	mA
Cin	采样电容		-	4	-	pF
Fclk	转换时钟频率	VCC=1.8~3.3V	5	-	-	MHz
Tsamp	转换时间		13			us
Tconv	转换时间		-	14*Tclk	-	
Teoc	转换完读取数据时间		-	0.5*Tclk	-	
DNL			-	±1	-	LSB
INL			-	±1	±2.2	LSB

ADC FT 校准及应用

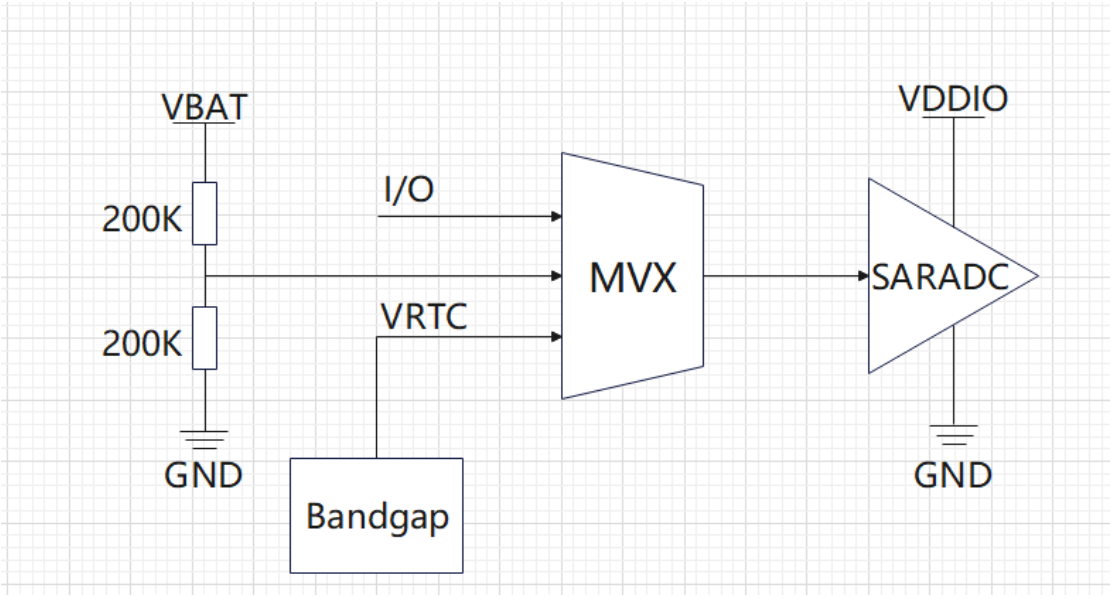


图 1 芯片内部 ADC 电路简易图示

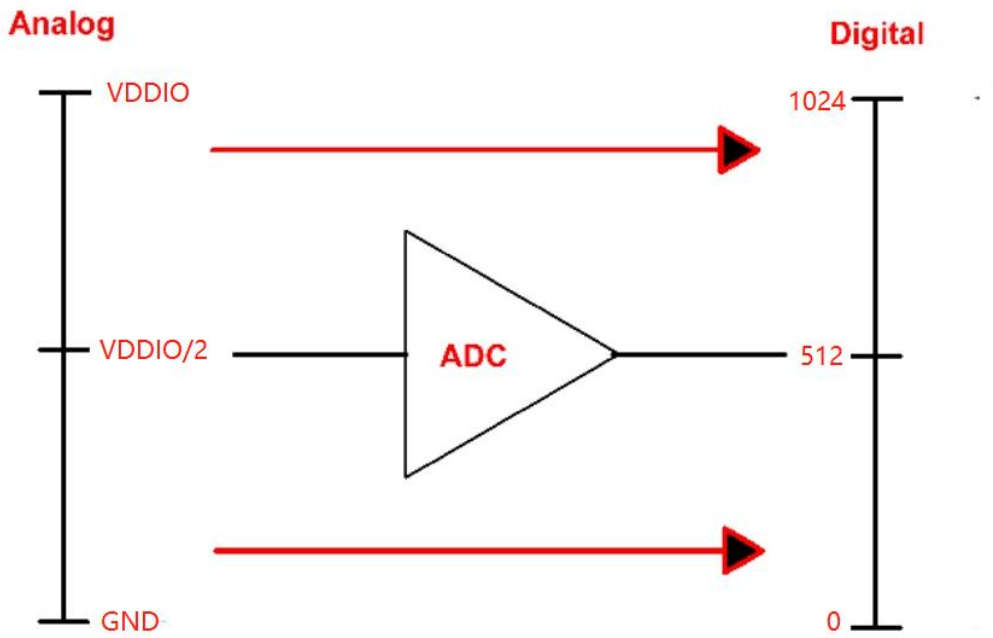


图 2 芯片模拟数字量对应关系图

AB202X

FT 环境下，机台输出一个 2.7V(10mV 以内偏差）给到 ADC_IO。SAR-ADC 记下当前值，同时记下 VRTC 的 SAR-ADC 值(VBG 无法 saradc 采样测试)，以此推算 VRTC 的采样偏差值，存在 Efuse 内，相当于用一个准确的 2.7V，TRIM BG 的电压。同时会对 VBAT 的分压系数做校准，减小代入误差。

常规使用的时候，调用 TRIM 的参数，因为实际使用中 VDDIO 的值会不准，会使用 FT TRIM 记下来的 VRTC 的采样偏移值来推算 VRTC 电压，最终推算出采样电压值

SDK 代码 ADC 计算

AB202X

Vbat 计算公式：

vbat = vbat2 * VRTC_VOLTAGE * VBAT2_COEF / (1000 * RTC);

说明：

vbat：经过计算后实际的电池电压值

vbat2: SARADC 模块 vbat 通路采样值

VRTC_VOLTAGE：经过 FT 修正后的 VRTC 电压，该值会在 2400(V1)/2000(V2)上下浮动

VBAT2_COEF: 分压系数（芯片 vbat 通路采样为二分压，分压电阻有精度误差，FT 会对 VBAT 的分压系数做校准，减小代入误差，vbat div2 的 电压进 saradc 采样， 后面计算 x2，再放大 1000 倍提高精度，该值在 2000 上下浮动）

RTC: SARADC 模块 RTC 通路采样值

IO 获取电压值计算公式：
IO_VOL = VRTC_VOLTAGE / RTC * IO_ADC
说明：
IO_VOL：经过计算后实际的 IO 电压值
IO_ADC：SARADC 模块此 IO 通路采样值，获取 10 位精度的采样值
（调用 saradc_get_data(u32 adc_chx)获取）
VRTC_VOLTAGE：经过 FT 修正后的 VRTC 电压，该值会在 2400(V1)/2000(V2)上下浮动
RTC: SARADC 模块 RTC 通路采样值

AB568X 芯片 ADC 性能及校准应用说明

ADC 性能

表 1 AB568X 芯片 ADC 性能

符号	参数	条件	最小值	典型值	最大值	单位
Idd	功耗	@2MHz fclk	-	0.7	-	mA
Cin	采样电容		-	4	-	pF
Fclk	转换时钟频率	VCC=1.8~3.3V	5	-	-	MHz
Tsamp	转换时间		13			us
Tconv	转换时间		-	14*Tclk	-	
Teoc	转换完读取数据时间		-	0.5*Tclk	-	
DNL			-	±1	-	LSB
INL			-	±1	±2.2	LSB

ADC FT 校准及应用

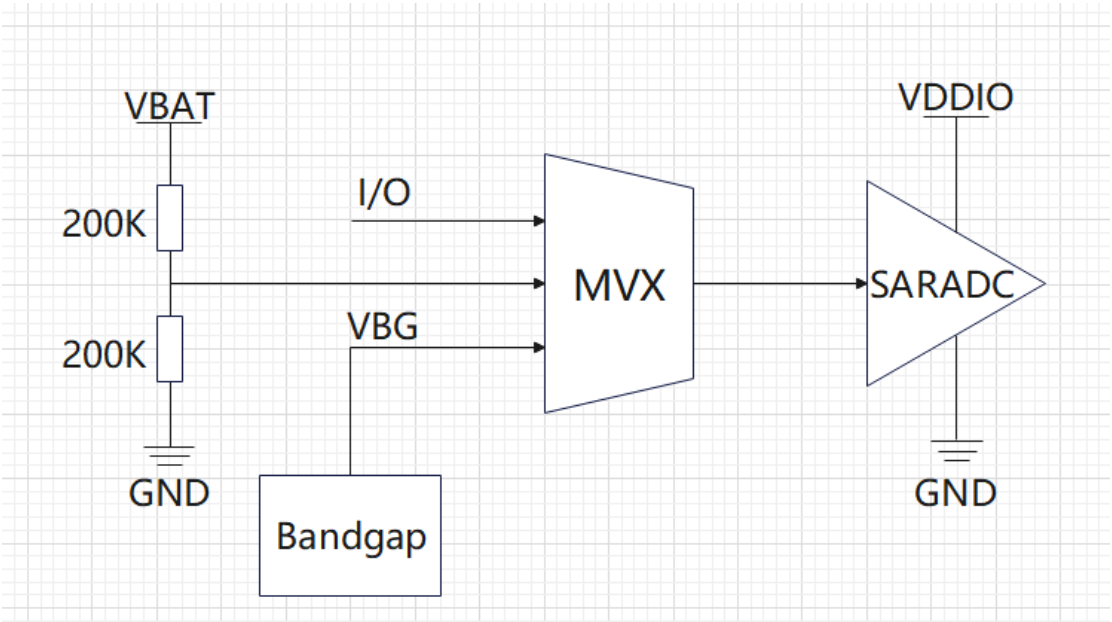


图 1 芯片内部 ADC 电路简易图示

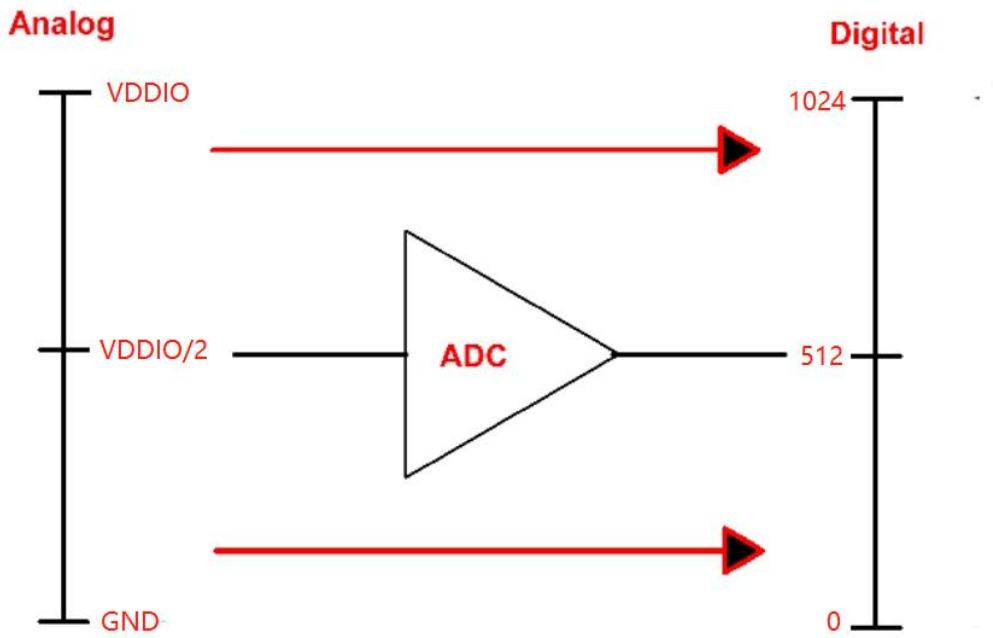


图 2 芯片模拟数字量对应关系图

AB568X

FT 环境下，机台给一个 3.3V 到 VDDIO（由于耗电，这个电压到芯片会偏低一些，保证 saradc 采样 vbg 和 vio 时候 vddio 是大致一样的），机台输出一个 2.7V(10mV 以内偏差）给到 ADC_IO。SAR-ADC 记下当前值，同时记下 BG 的 SAR-ADC 值，以此推算 VBG 的采样偏差值，存在 Efuse 内，相当于用一个准确的 2.7V，TRIM BG 的电压。同时会对 VBAT 的分压系数做校准，减小代入误差。

常规使用的时候，调用 TRIM 的参数，因为实际使用中 VDDIO 的值会不准，会使用 FT TRIM 记下来的 VBG 的采样偏移值来推算 VBG 电压，最终推算出采样电压值，依赖于 BG 在不同环境下，能保持一个比较固定的模拟量

SDK 代码 ADC 计算

AB568X

Vbat 计算公式：

vbat = vbat2 * VBG_VOLTAGE * VBAT2_COEF / (1000 * bg);

说明：

vbat：经过计算后实际的电池电压值

vbat2: SARADC 模块 vbat 通路采样值

VBG_VOLTAGE：经过 FT 修正后的 VBG 电压，该值会在 600 上下浮动

VBAT2_COEF: 分压系数（芯片 vbat 通路采样为二分压，分压电阻有精度误差，FT 会对 VBAT 的分压系数做校准，减小代入误差，vbat div2 的 电压进 saradc 采样， 后面计算 x2，再放大 1000 倍提高精度，该值在 2000 上下浮动）

bg: SARADC 模块 bg 通路采样值

IO 获取电压值计算公式：
IO_VOL = VBG_VOLTAGE / bg * IO_ADC
说明：
IO_VOL：经过计算后实际的 IO 电压值
IO_ADC：SARADC 模块此 IO 通路采样值，获取 10 位精度的采样值（调用 saradc_get_value10(n) 获取）
VBG_VOLTAGE：经过 FT 修正后的 VBG 电压，该值会在 600 上下浮动
bg: SARADC 模块 bg 通路采样值