TINYCHIP

APPLICATION NOTE

TSAAS0023C：血氧仪方案参考设计

Tinychip AFE SoC

保密要求

本文档仅由上海泰矽微电子有限公司提供给与其签署过保密协议的合作伙伴。

请各合作伙伴遵循保密协议中的相关要求，对涉及的相关信息保密，并承诺采取合理的措施以保证保密信息不被泄露。

未经披露方的事先书面批准，接受方不得直接或间接以任何形式或任何方式把保密信息和(或)其中的任何部分，披露、透露给第三方或者公开。接受方仅能向有知悉必要的接受方人员披露保密信息。

合作伙伴违反上述任何要求，均视为违约。违约方应当对其违约行为，以及给披露方造成的损失承担赔偿责任。

目录

1. 概述

本篇应用笔记主要介绍泰矽微电子 TCAS 系列 MCU 的血氧仪设计方案，介绍了血氧仪的方案框架、血氧仪产品使用的外设模块、产品原理图、软件架构、主要模块的测试方法和数据，以及使用 MCU 的注意事项。目的是为了让用户更好地理解该产品的设计要点、所具有的功能和性能指标，帮助用户快速地基于该 MCU 进行相关产品的开发。

本篇应用笔记主要包括：

- 血氧仪产品介绍
- 产品设计方案
- 产品原理图
- 各模块设计详解
- 软件架构和软件设计
- 示例代码介绍
- 模块测试数据
- 使用 MCU 的注意事项

注意：

— 本应用笔记为泰矽微电子 TCAS 系列 MCU 的应用补充材料，不能代替用户手册，具体功能及寄存器的操作等相关事项请以用户手册为准。

2. 血氧仪产品介绍

本节介绍血氧仪产品功能，包括工作原理、设计难点和主要功能介绍。

2.1 脉搏血氧仪工作原理

脉搏血氧仪以非介入方式测量血液中的含氧量，它以完全饱和水平的百分比来衡量，即所谓血氧饱和百分比，常常称之为 SpO2。

该测量基于血液中血红蛋白的光吸收特性。在可见光谱和近红外光谱内，含氧血红蛋白(HbO2)与脱氧血红蛋白(Hb)具有不同的吸收曲线。Hb 吸收的红光频率的光线较多，红外光(IR)频率的光线较少；HbO2 则相反，吸收的红光频率的光线较少，红外光(IR)频率的光线较多。红光和红外光 LED 尽可能相互靠近，通过人体中的单一组织位置透射光线。红光和红外光 LED 采用分时发送来透射光线，因此不会相互干扰。环境光线经过估算后，从红光和红外光信号中减去。一个能够响应红光和红外光的光电二极管接收光线，然后由一个跨阻放大器（TIA）转换成与所接收光线强度成比例的电压。

光电二极管接收的红光与红外光的比值用于计算血液中的氧气百分比，根据血液流动的脉冲特性，还会在测量周期中确定并显示脉搏率和强度。

血氧仪通过人体的静脉血、动脉血和其它生理组织的光吸收特性进行测量，如下图所示：

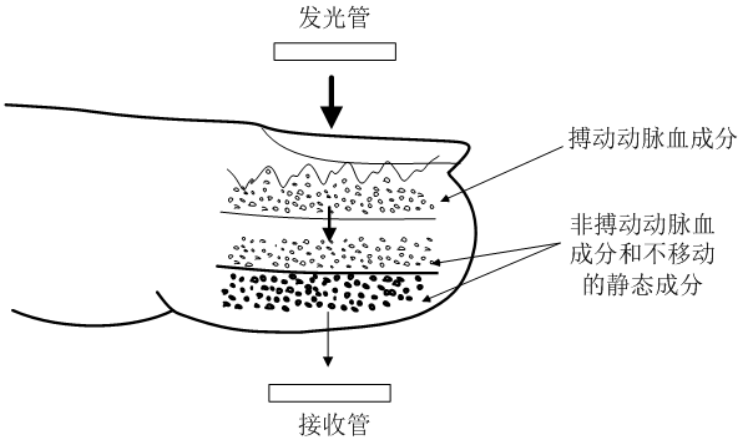

图 1 血氧仪测量示意图

2.2 产品设计难点

脉搏血氧仪包括发射路径、接收路径、显示和背光、数据接口以及蜂鸣器报警。

发射路径包括红光 LED、红外光 LED 和用于驱动 LED 的 DAC。

接收路径包括光电二极管传感器、信号调理、模数转换器和处理器。

系统设计考虑和难点：

1) 低血流灌注(小信号水平)：光电二极管测量需要宽动态范围和低噪声增益的信号调理，以便捕捉脉搏事件。发射路径需要具有高分辨率 DAC，低噪声 LED 驱动电路；接收路径需要具有高分辨率 ADC 的高精度模拟前端电路。

2) 杂散光干扰：使用光电二极管来接收红光和红外光，它很容易受环境光干扰。因此，用于过滤出红光和红外光目标信号的算法非常重要，这意味着信号处理算法比较复杂。这种情况下，需要使用具有更高信号处理能力的 MCU。

3) 生理组织的信号受干扰：一氧化碳(CO)很容易与血红蛋白结合，使血液变得更像红色 HbO2，导致测得的 SpO2 值虚高。可以通过使用更多波长来提高精度，但这需要更高性能的数字处理，算法处理时间至关重要。

2.3 主要功能介绍

采用 TCAS 系列 MCU 的脉搏血氧仪产品，主要包括以下功能模块：

1) MCU 处理单元
2) 信号发送单元
3) 信号接收和处理单元
4) 电源管理
5) 显示单元
6) 按键模块（开关机）
7) 电池检测

8) 蜂鸣器报警

9) 蓝牙通信模块

TCAS 系列血氧仪产品功能框图如下所示：

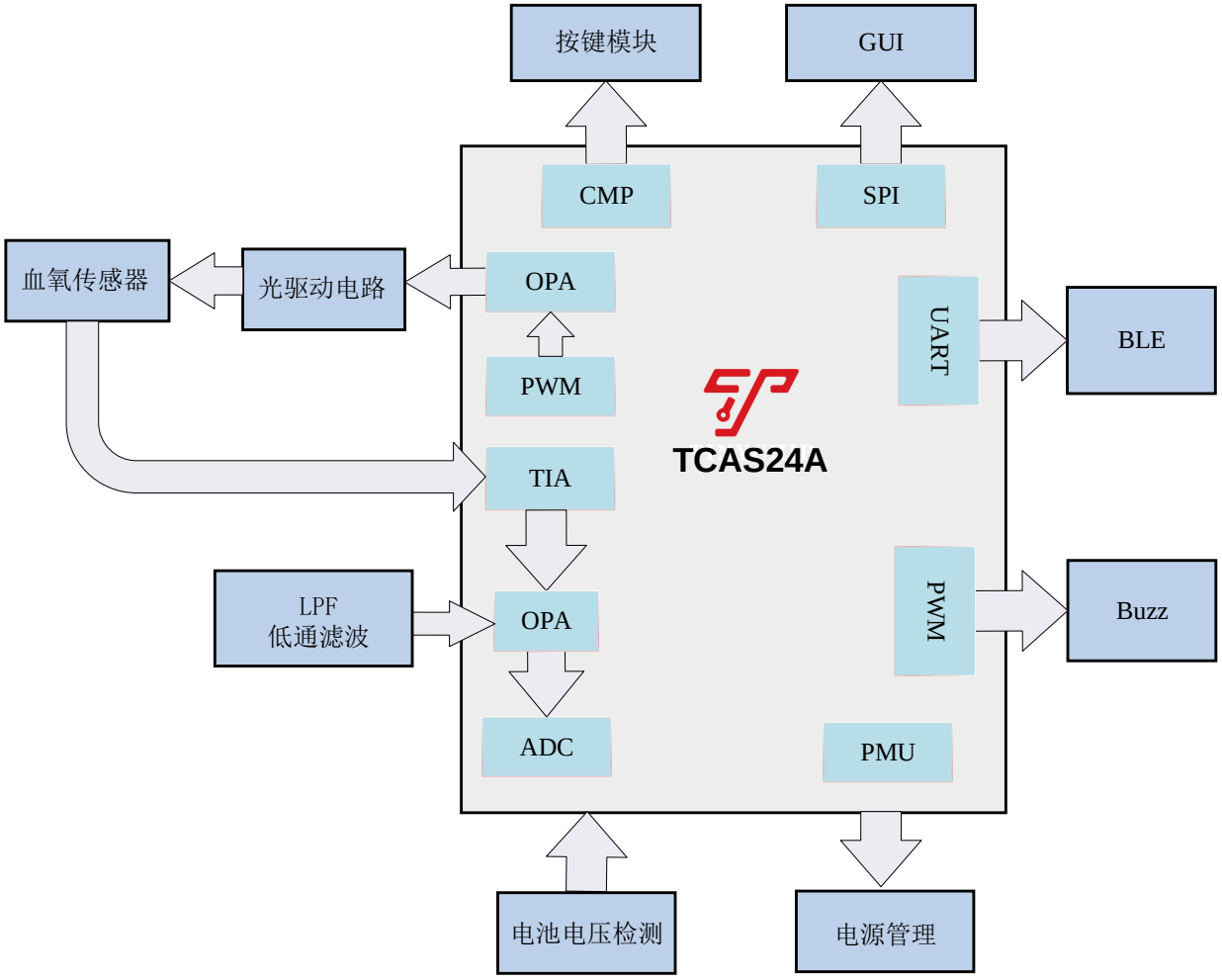

图 2 TCAS 系列血氧仪产品功能框图

3. TCAS 系列血氧仪方案

本节介绍采用 TCAS 系列 MCU 的血氧仪设计方案，用户可以优先采用本文介绍的参考方案。

3.1 硬件架构

采用 TCAS 系列 MCU 开发的血氧仪产品硬件架构如下图所示，主要包括以下几个模块：

1) **电源模块：**电池采用 2 节 AAA 电池供电，经 DCDC 升压至 3.6V，再通过 2 路 LDO 降压至 3.3V，确保随电池电量降低后主板供电电压的稳定。

2) **发光模块：**发光部分通过 MCU 控制发光序列和驱动电压信号，通过一对 H 桥切换开关实现红光 LED 与红外光 LED 的正反切换点亮。

3) **接收模块：**PD 信号经 IV 变换和一级运放，送至 MCU 采样。MCU 内含 14 位 ADC 实现模数转换。

4) **显示模块：**支持 OLED 显示屏和 LCD 显示屏。

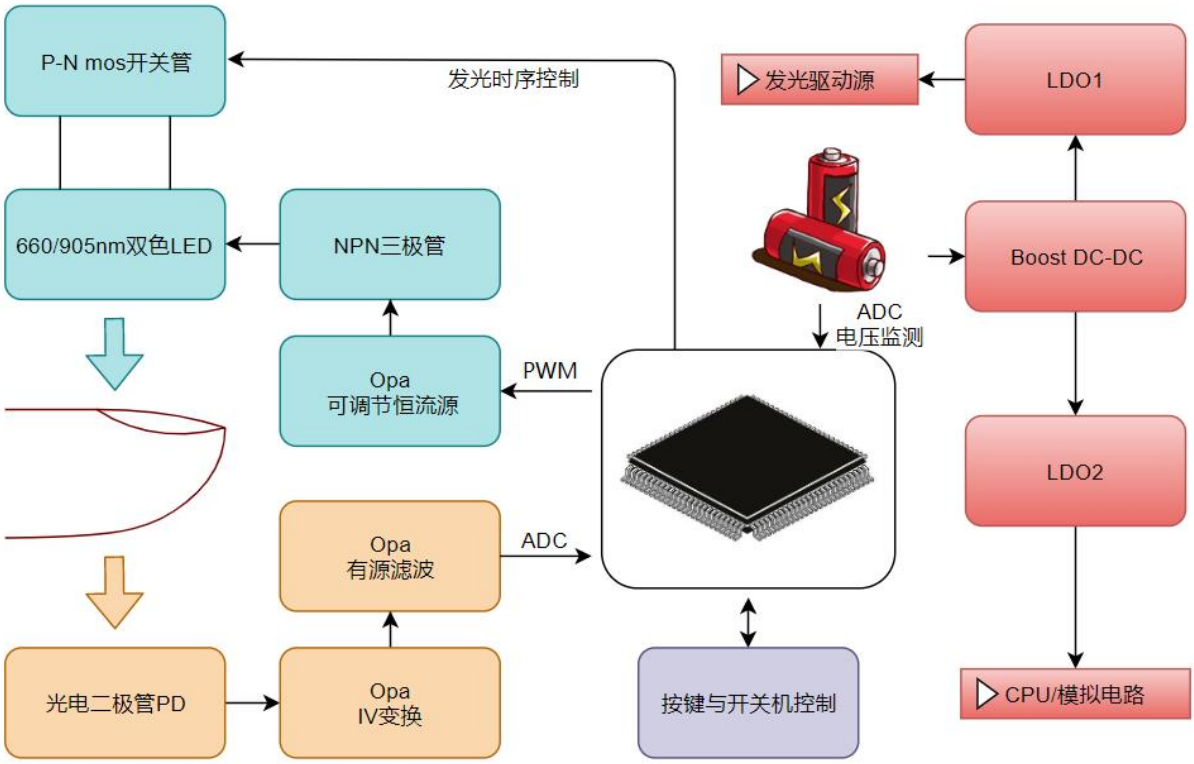

图 3 硬件架构示意图

说明：

1、发光模块的 OPA 输出可以驱动 NPN 三极管或者 NMOS 管。

3.2 产品原理图

血氧仪产品原理图参考设计如下：

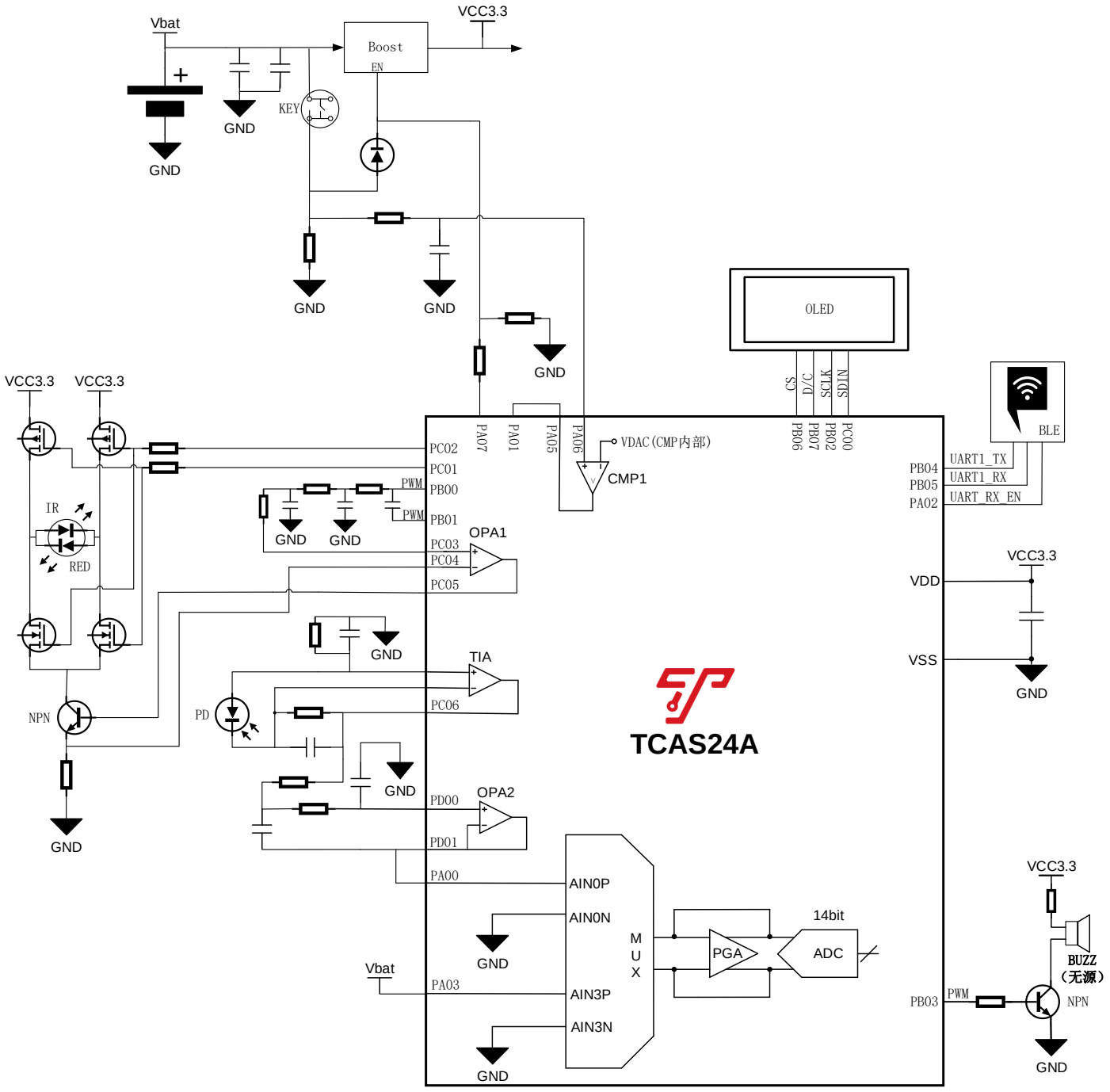

图 4 血氧仪产品原理图

3.3 软件架构

TCAS 系列血氧仪产品软件架构如图所示，包括以下各个单元功能：

1) MCU 主控：所有单元的运行载体，通过中断任务调度，实现各单元有序执行；
2) 采样处理单元：完成生理信号采样和噪声预处理，并输送给算法单元；
3) 血氧算法单元：完成生理信号的实时处理，并输出测量结果；
4) 按键处理单元：响应用户触发的按键消息，实现人机交互功能；
5) GUI 显示单元：提供参数显示、功能设置、趋势浏览等交互功能；

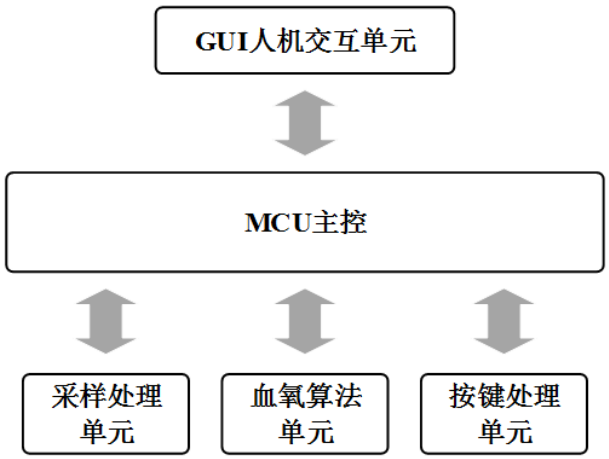

图 5 血氧仪产品软件架构

3.4 软件设计方案

采用 TCAS 系列 MCU 开发的血氧仪产品软件设计包括以下主要模块：

1. 信号发送控制，PWM 输出
2. 定时器（精准的定时器时间）
3. 光电二极管信号接收（SARADC、TIA）
4. 接收信号预处理（信号滤波处理）
5. 算法模块，后台计算和处理
6. 中断控制器、中断优先级管理
7. FLASH 存储管理
8. 应用协议通信
9. OLED/LCD 显示管理
10. 按键管理
11. 电池电压检测
12. 串口（连接蓝牙，选配）

由于算法计算和处理功能占用 CPU 时间会比较长，为了提升按键和显示等功能的响应速度，提供更好的人机交互友好性，软件设计采用中断驱动的模式。后台循环中只处理算法计算功能，提供计算结果。按键和显示等人机交互功能放在 AIC 低优先级中断执行，可以随时打断后台的算法计算和处理执行。而 FIC 等高优先级中断可以通过中断嵌套来优先运行，执行 LED 发送控制等最高实时性要求的任务。

血氧仪产品软件设计总体框图如下所示：

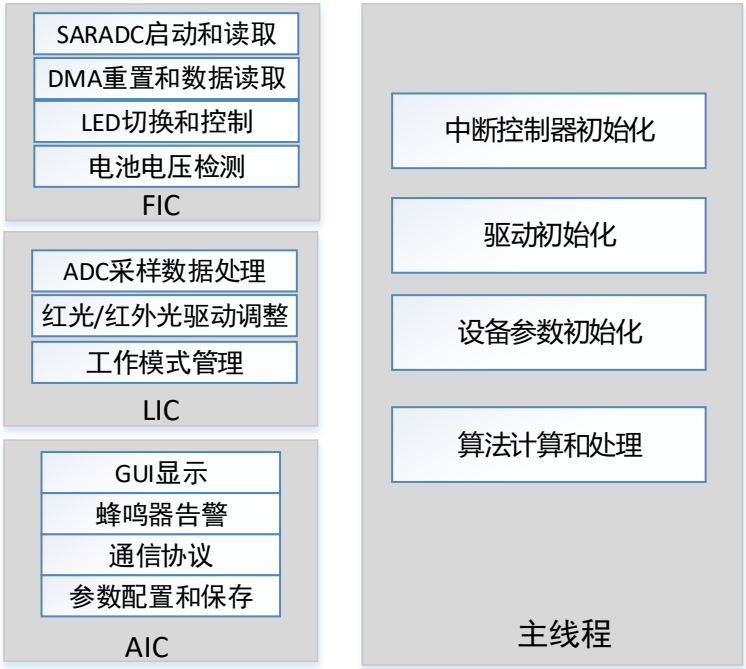

图 6 软件设计总体框图

其中：

1. FIC 中断优先级最高，由定时器(TIMER2)中断触发，通过 GPIO（PC01/PC02）实现红光和红外光的发送开关。
2. LIC 中断优先级次高，由 FIC 中断 ISR 触发，主要实现 ADC 采集的光电传感器数据预处理，红光/红外光驱动发射的电流调整等功能。
3. AIC 优先级最低，由 LIC 中断 ISR 触发，主要实现 GUI 显示、告警、通信协议和参数配置等人机交互功能。这些功能的处理时间较长，优先级较低。所以要求高优先级的中断可以嵌套实现。
4. 后台 while 循环主要实现算法计算和处理功能，占用 CPU 时间会比较长，所以需要考虑人机交互的响应实时性。

4. 各模块设计

4.1 LED 驱动与控制

根据 SpO2 检测原理，通过控制 RED 和 IR LED 的发光强度来改变接收信号的 AC 部分的比率，可以有效的将微弱的交流信号进行放大处理；否则微弱的交流信号叠加在较大的 DC 信号上，信号放大后很容易造成信号饱和。

有两种方式可供选择用来驱动 LED 发光：

方式一、VDAC 驱动

方式二、PWM 驱动

本方案采用两个 GPIO 控制 LED 切换：PC01 用来控制红光 LED，PC02 用来控制红外光 LED。通过定时器 TIMER4 的 PWM 通道 CHA 和 CHB 控制 LED 驱动电流的大小，从而将信号的 AC/DC 比例控制在比较合理的范围内。CHA 和 CHB 配置为互补型 PWM 输出波形。

红光二极管(RED)与红外光二极管(IR)驱动电路如图所示：

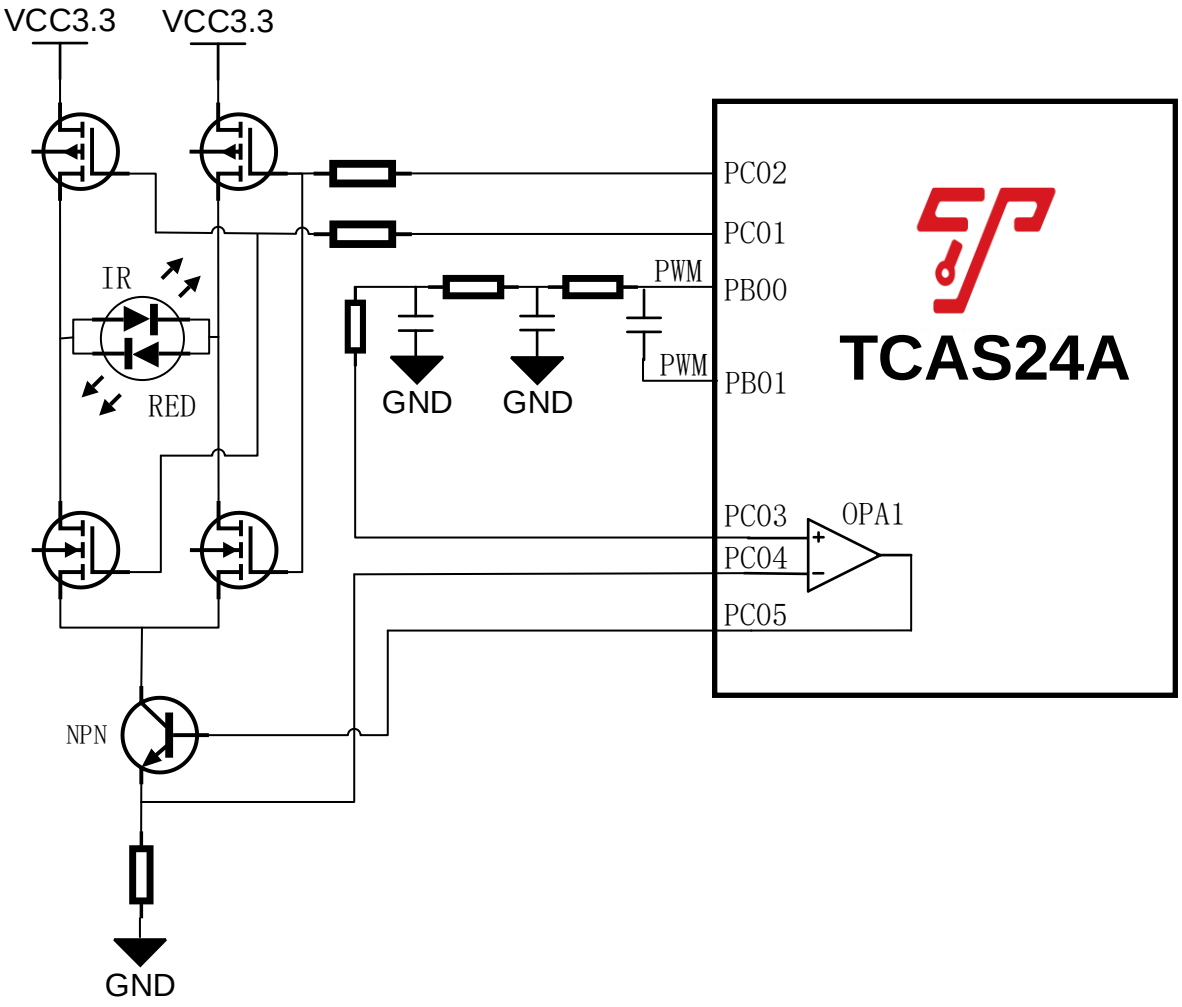

图 7 LED 驱动电路

具体原理说明：

改变 TIM4_CHA 和 TIM4_CHB 的 PWM 占空比，OP1 同相输入端电压会相应的改变，控制 NPN

三极管工作在线性放大区，由于 NPN 拓扑结构为射极跟随器，因此转换电阻上的电压约等于 OP1 的输出电压。通过电压变化改变 LED 的驱动电流，可以控制 LED 亮度。

本方案的 H 桥电路采用分立的 MOS 管搭建实现。用户也可以通过集成电路 H 桥 IC 芯片来实现，比如 RUNIC 的 RS2105 或其他模拟开关 IC 等。

4.2 SpO2 信号接收

跨阻放大器 TIA 将 PD 产生的电流信号转换为电压信号。转换的电压信号中含有大量的干扰信号，比如环境光等，需要对原始信号进行滤波处理。通常有用信号的频率在 1~10Hz 的范围，本方案用巴特沃斯低通滤波器来实现滤波，截止频率设置为 10Hz。经过滤波后的信号送至 SARADC 进行采样转换，然后经软件算法处理得到血氧含量值。

本方案 SpO2 信号接收使用的模拟外设均集成在 TCAS24A 芯片内部，对片外器件几乎无要求，仅需要一些辅助的阻容器件。

SpO2 检测信号链原理图如图所示：

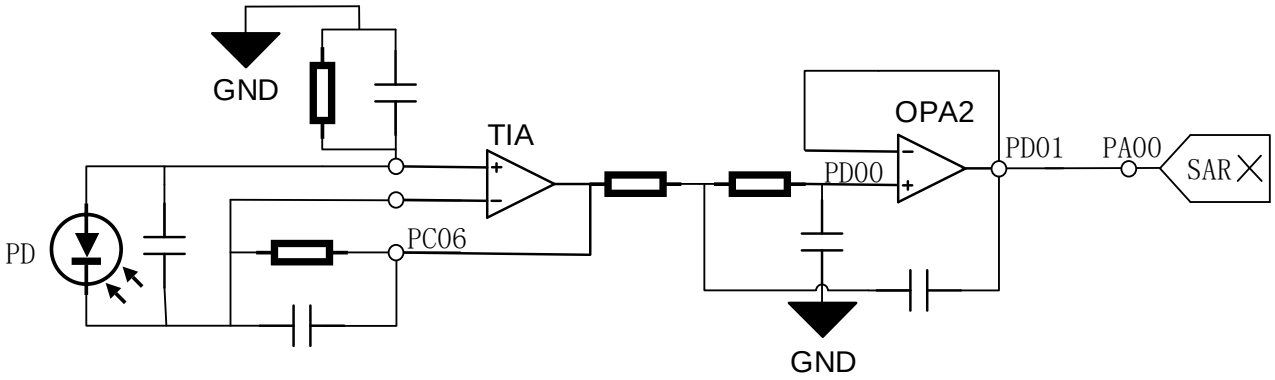

图 8 SpO2 信号采集信号链

4.3 UI 显示

UI 显示模块将信号链采集到的血氧含量信号经过算法处理后进行实时显示，并配合按键实现参数配置、开关机等人机交互功能。

本方案采用 OLED 进行显示，OLED 屏与 TCAS24A 通过 SPI 接口进行连接。由于 OLED 仅用于单方向进行显示用，本方案的 SPI 接口不使用 MISO 线，改为 D/C 控制信号线。

OLED 屏电路设计如图所示：

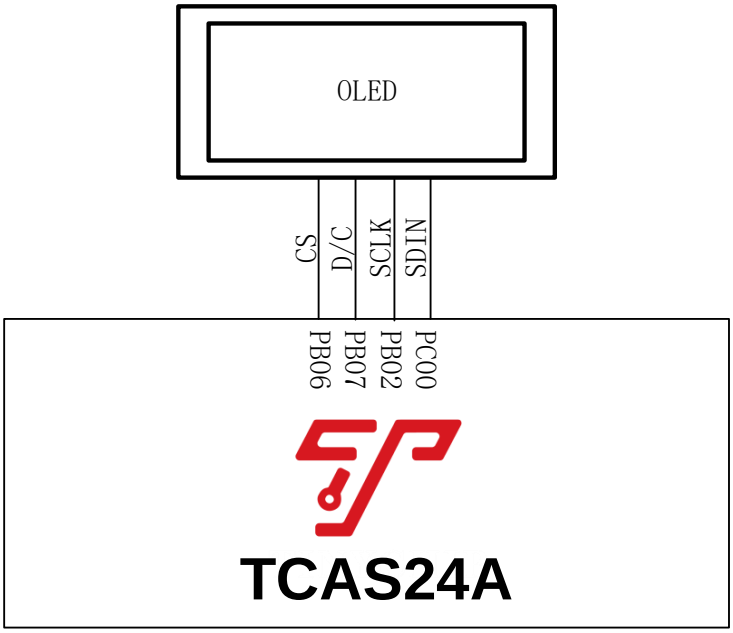

图 9 OLED 显示

4.4 蜂鸣器报警

报警模块用于血氧含量高于或低于阈值时提示用户，本方案采用无源蜂鸣器来实现。

通过控制蜂鸣器的 PWM 占空比来调节蜂鸣器的音量，通过控制 PWM 频率来调节蜂鸣器的声音频率。具体电路如下图所示：

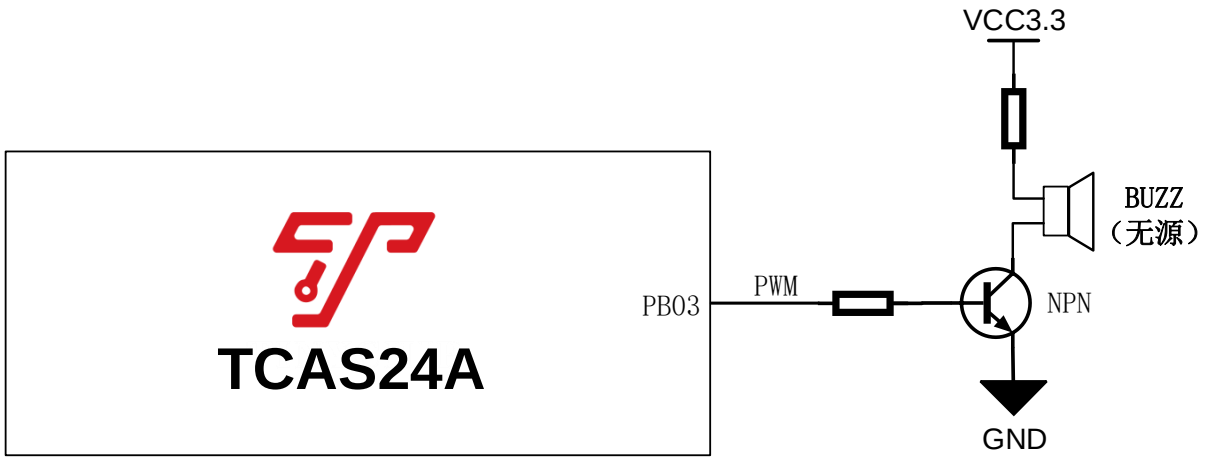

图 10 蜂鸣器控制电路

4.5 电池电量采集

本方案提供一路 ADC 通道用于检测电池电压，通过电池电压来判断当前电池的电量状况。具体实现原理如图所示：

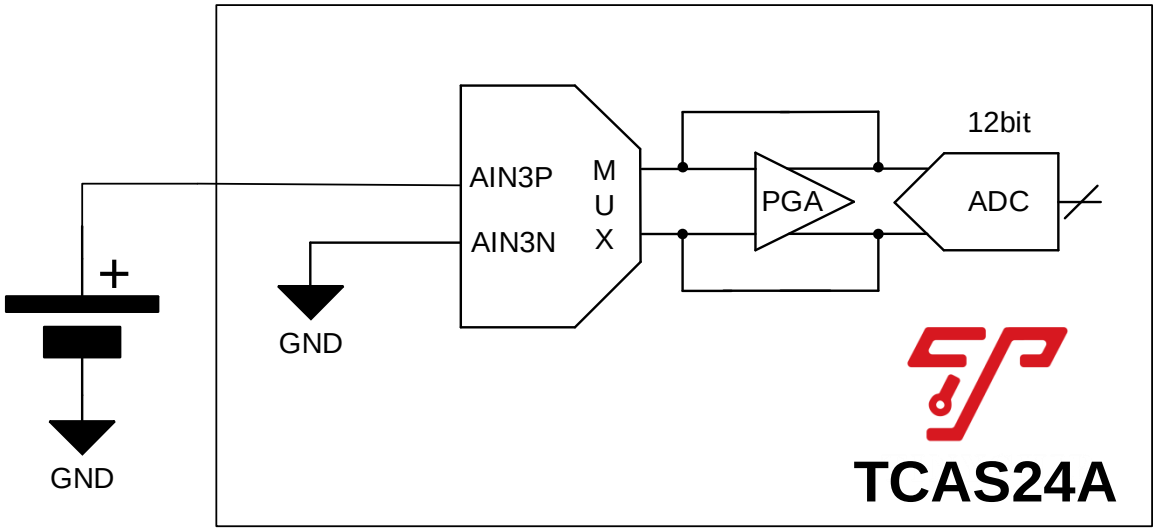

图 11 电池电压检测

本方案通过切换 ADC 通道的方式来实现电池电压采集，在需要采集电池电压的情况下将 SARADC 的输入通道切换至 AIN3(PA03)进行电池电压采集，采集完成后再将 ADC 的通道切至 AIN0(PA00)上进行血氧含量信号采集。

4.6 蓝牙模块

本方案可以选配一个 BLE 蓝牙通信模块，通过串口与 MCU 连接。可以通过蓝牙将血氧含量等数据发送给远程终端，比如手机。这样终端 APP 可以显示当前血氧的含量情况。

本方案支持两种应用场景：单工发送模式、全双工模式。串口采用 UART1 进行数据发送和接收，PA02 作为接收控制信号。具体原理如图所示：

图 12 蓝牙模块连接示意图

4.7 按键和电源管理

本方案设计一个开关机电路，该电路中包括一个按键，并用到了比较器 CMP1。

如果将电池电压通过按键连接到 GPIO，在按键按下时 GPIO 获得的电压为电池电压。在电池电压低于 TCAS24A 的逻辑高电平范围时 MCU 无法获取当前按键的状态，因此本方案采用比较器的方式实现按键的开关机电路。

开机时，按下按键使能 BOOST 电路，从而使 MCU 上电。软件在 MCU 初始化时将 PA07 输出高电平，保证 BOOST 芯片的 EN 脚始终为使能状态。

软件检测到手指脱离红外传感器一定时间、用户长按键等情况，可以通过控制 PA07 输出低电平关机。关机的应用场景包括以下几种：

1) 电池电压低于阈值；

2) 检测到手指脱离红外传感器超时；

3) 用户长按键；

以上应用场景的判断需要由软件实现。例如用户长按键关机的判断逻辑如下：

用户长按键时，电池电压与 VDAC 比较（比如 VDAC 设为 1.2V），比较器输出高电平到 PA01；软件检测 PA01，检测到高电平并持续超过一定时间（比如 3 秒）后，判断为关机操作，软件控制 PA07 输出低电平 DISABLE 升压 DCDC 芯片，最后 MCU 掉电。

具体设计如图所示：

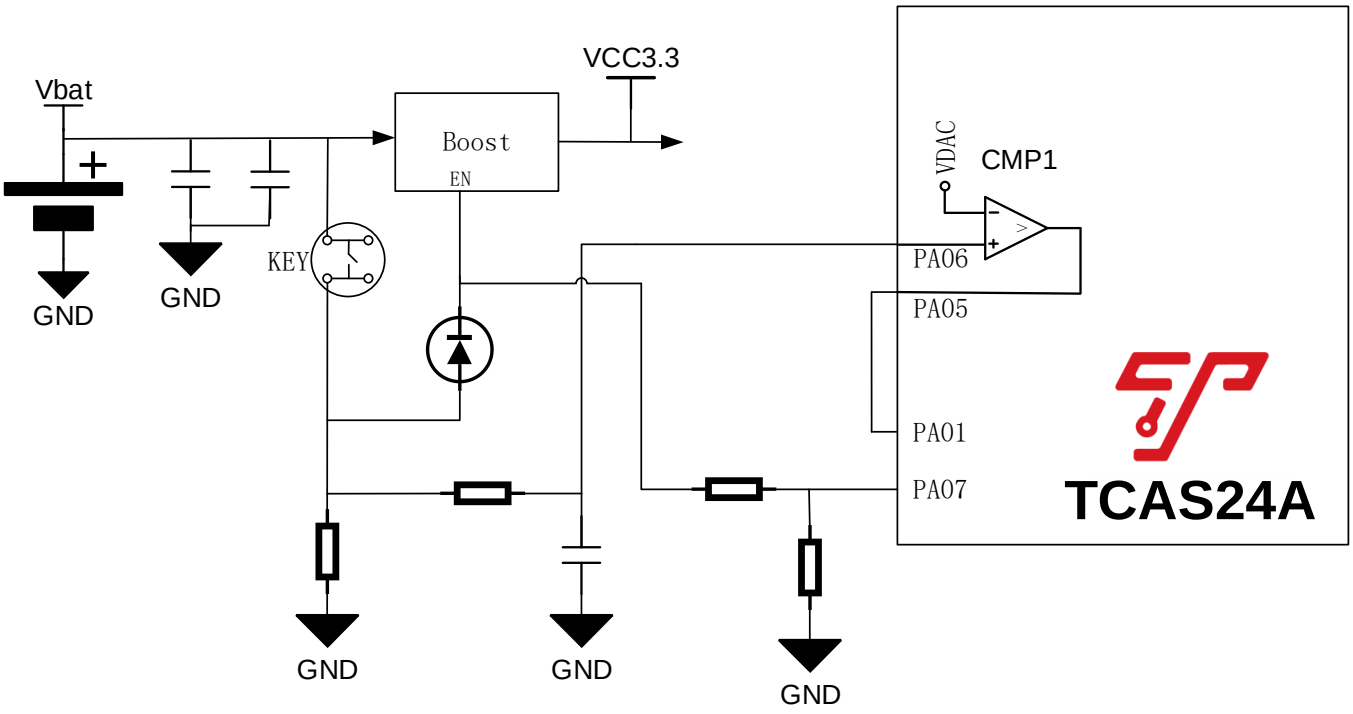

图 13 开关机电路

说明：

比较器 CMP1 使用的 VDAC 参考电压由比较器内部的 VDAC 输出。

5. 软件设计介绍

5.1 系统中断

➢ 中断初始化

中断初始化通过 IC_PowupInit 函数实现。中断源一共 8 个，其中 5 个为内核中断，另外 3 个分别是 FIC、LIC、AIC 中断。

中断初始化的示例代码：

```
void IC_PowupInit(void)
{
    // init FIC/LIC/AIC interrupt vector table
    memset((void*)jump_table_base, 0, sizeof((void*)jump_table_base));

    jump_table_base[0] = NMI_Handler;
    jump_table_base[1] = HardFault_Handler;
    jump_table_base[2] = SVC_Handler;
    jump_table_base[3] = PendSV_Handler;
    jump_table_base[4] = SysTick_Handler;
    jump_table_base[5] = fic_IRQHandler;
    jump_table_base[6] = lic_IRQHandler;
    jump_table_base[7] = aic_IRQHandler;
}
```

➢ 中断优先级

通过 NVIC_SetPriority 设置中断优先级。Fic_IRQn 优先级最高（0），Lic_IRQn 优先级为 1，Aic_IRQn 优先级为 2。

➢ 中断使能

中断使能由 EnableInterrupt 实现，该函数配置 FIC/LIC/AIC 中断优先级，并使能这三个中断，同时打开 TIMER2 定时器中断屏蔽。

Fic_IRQn 由定时器(TIMER2)中断触发，周期为 60us。fic_IRQHandler 函数执行一定周期数（74）后通过调用 NVIC_SetPendingIRQ(Lic_IRQn)函数，触发 Lic_IRQn 中断。lic_IRQHandler 函数执行完成后，会通过 NVIC_SetPendingIRQ(Aic_IRQn)调用，触发 Aic_IRQn 中断。aic_IRQHandler 中断处理函数负责显示和通信等人机交互功能。

➢ 中断处理流程

中断处理流程如下图所示：

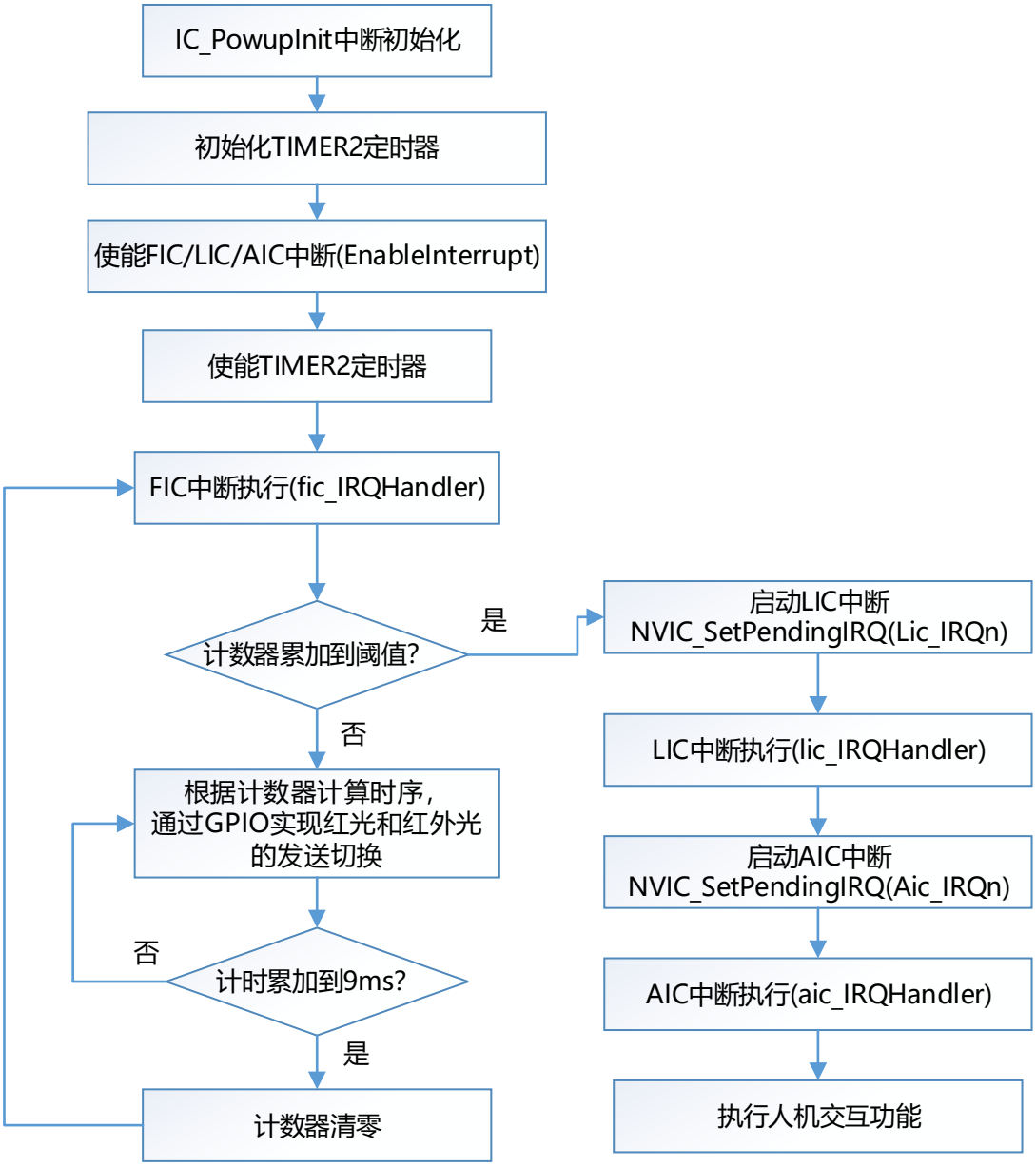

图 14 中断执行流程图

5.2 定时器

方案使用了 3 个定时器：TIMER2、TIMER3、TIMER4

TIMER2 定时器用于产生 FIC 中断，控制 LED 发光和 ADC 采样过程。

TIMER3 定时器用于产生 PWM，驱动蜂鸣器告警。

TIMER4 定时器用于产生 PWM，驱动 LED 红光和红外光发送。PWM 配置为互补 PWM 模式，2 通道 CHA 和 CHB 输出互补型 PWM 波形。

TIMER4 定时器配置示例代码如下：

1、配置 IO 的模式

```
// Complementary PWM (PB.0 & PB.1)
```

```
GPIO_SetAFMode(GPIOPortB, GPIOPin0, AF3);
GPIO_SetAFMode(GPIOPortB, GPIOPin1, AF3);
```

2、配置定时器

```
void InitTimerPWM(void)
{
    timerconfig_t timerConfig;

    timerConfig.timer    = TIMER4;
    timerConfig.prescale = 0;

    timerConfig.period    = PWM_GAP_FRQ - 1;
    timerConfig.matchAOp  = 4;
    timerConfig.matchA0   = 0;
    timerConfig.repeatVal = 0;

    timerConfig.mode = 0x6;
    timerConfig.wave = 0;
    timerConfig.matchnum  = 0;/* 一个匹配点 */
    timerConfig.ece  = 0;

    SetTimer(&timerConfig);
    ADT_StartTimer(&timerConfig);
}
```

5.3 TIA 配置

TIA 用于传感器接收信号采集，配置为独立模式。正端和负端都连到 PIN 脚，输出也是到 PIN 脚。另外，为了提升性能，按照默认配置 TIA_STG1 和 TIA_STG2 的电流（最大），并使能 REG_OPA_IV 寄存器的高功耗 opa_iv_high_en 开关。

寄存器 REG_OPA_IV 用于配置 TIA 的输入、输出选择等功能。当输入配置为 IO 口时，TIA 的正端和负端输入固定为 OP0_INP、OP0_INN 两个 IO 引脚。

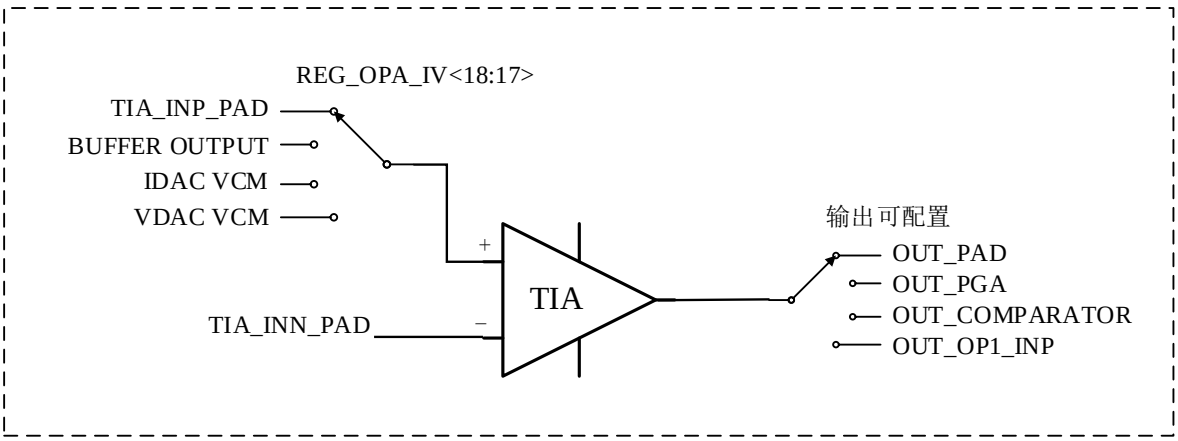

图 15 TIA 功能框架图

说明：

1) 寄存器 REG_OPA_IV 位<18:17>控制 TIA 正端输入选择，分别是片外参考电压、TIA 内部 BUFFER 输出、IDAC 输出和 VDAC 输出；
2) 当 TIA 正端输入选择 TIA 内部 BUFFER 输出时，电压范围从 0 到 1.5V，100mV 一个档位。由寄存器寄存器 REG_OPA_IV 位<11:8>控制；
3) 寄存器 REG_OPA_IV 位<16>是 TIA 输出到下一级 OPA 选择开关，0 表示 TIA 输出到 IO；
4) TIA 的转换电阻由外部连接；

TIA 配置示例代码如下：

```
// TIA initialize
Tia_Init_t tia_init_t = {0};

tia_init_t.tiaHighPowerEn = ENABLE;
tia_init_t.tiaStg1Level = TIA_STG1_LEVEL3;
tia_init_t.tiaStg2Level = TIA_STG2_LEVEL3;
tia_init_t.tiaOutConfig = TIA_OUT_CONECT_NONE;
tia_init_t.tiaPosConfig = TIA_VIP_CONECT_NONE;
tia_init_t.tiaVipSelect = TIA_VIP_SELECT_EXTERNAL;
tia_init_t.tiaOutSelect = TIA_OUT_SELECT_PAD;

Tia_Init(&tia_init_t);
Tia_Enable();
```

5.4 OPA 配置

本方案使用了 OP1/OP2 两个运算放大器。

OP1 配置为独立运放，输入和输出都连接到 PIN 脚，外部硬件配置成电压跟随模式。OP1 输出稳定的电压到 LED 发送电路，从而驱动 LED 发送并可以调节光的强度。

OP2 通过内部配置为电压跟随器，P 端和输出端配置到 PIN 脚，N 端配置为 BUFFER 模式。OP2 实现了低通滤波器功能，用于对接收的 PD 传感器信号进行滤波。

OP1 的配置示意如下图所示，OP2 的配置与 OP1 类似。

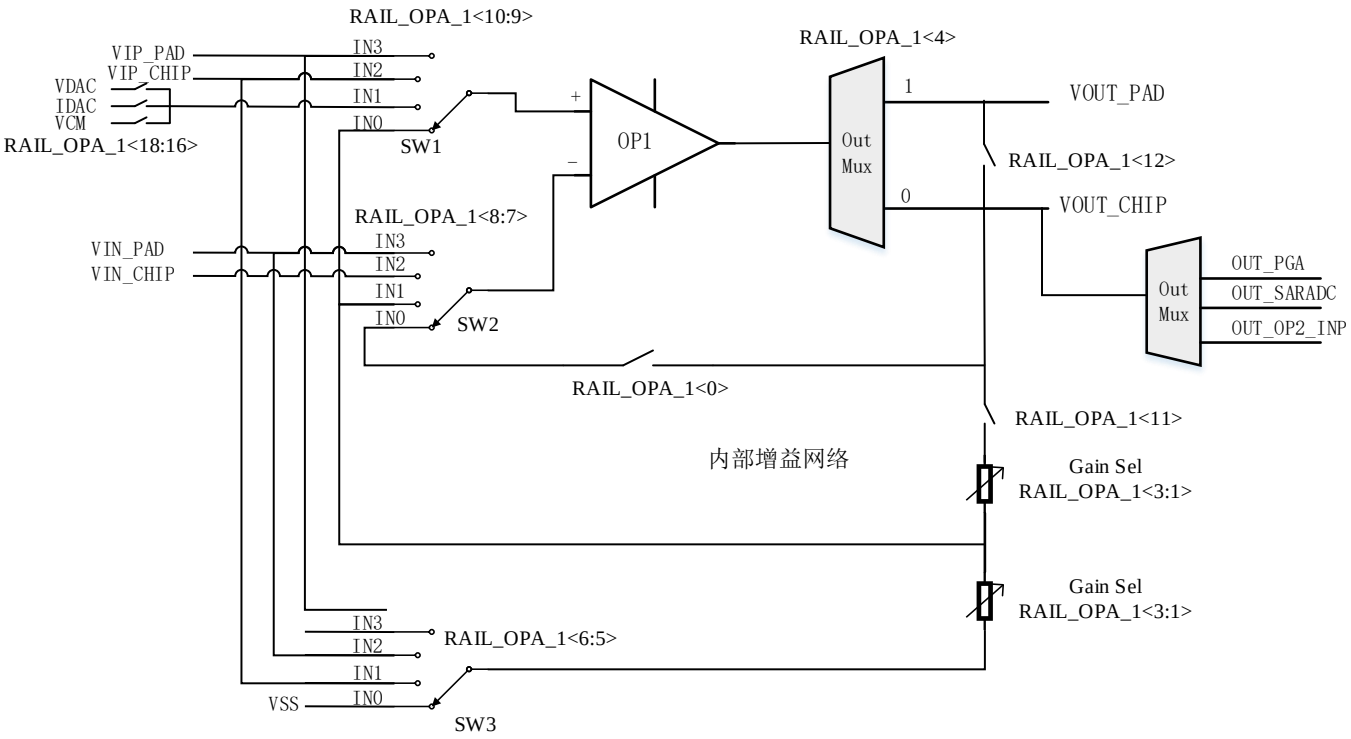

图 16 OP1 功能和配置示意图

OP1 的软件配置主要通过寄存器 REG_RAIL_OPA_1 实现。其中：

1) SW1 由寄存器 REG_RAIL_OPA_1 的位<10:9>控制，选择 OPA 的 P 端输入信号；
2) SW2 由寄存器 REG_RAIL_OPA_1 的位<8:7>控制，选择 OPA 的 N 端输入信号；
3) SW3 由寄存器 REG_RAIL_OPA_1 的位<6:5>控制，选择反馈电阻（FRE 使能打开）；
4) GainSel 由寄存器 REG_RAIL_OPA_1 的位<3:1>控制，选择放大增益倍数。
5) 寄存器 REG_RAIL_OPA_1 位<0>是内部电压跟随 buffer 使能开关；
6) 寄存器 REG_RAIL_OPA_1 位<4>是 OP 输出到 PAD 选择开关；
7) 寄存器 REG_RAIL_OPA_1 位<11>是内部反馈电路使能开关；
8) 寄存器 REG_RAIL_OPA_1 位<12>是 OP 输出 PAD 脚连接片内反馈电路的开关；

OP2 配置示例代码如下：

```
// OP2 initialize
Opa_Init_t op2_init_t = {0};

op2_init_t.opaBufferEn   = ENABLE;
op2_init_t.opaOutPadEn   = ENABLE;
op2_init_t.opaFeedbackEn = DISABLE;

op2_init_t.opaPosConfig = OPA_VIP_PAD;
op2_init_t.opaNegConfig = OPA_VIN_BUFFER;
op2_init_t.opaOutSelect = OPA_OUT_PAD;
```

```
op2_init_t.opaResConfig = OPA_RES_GND;

op2_init_t.opaGainLevel = OPA_GAIN_LEVEL0;
op2_init_t.opaVipSelect = OPA_VIP_VREF_NONE;

OP2_VinPadConfig(OP2_VIN_PAD_PC07);
OP2_VipPadConfig(OP2_VIP_PAD_PD00);
OP2_OutPadConfig(OP2_OUT_PAD_PD01);
OPA_Init(OP_2, &op2_init_t);
OPA_Enable(OP_2);
```

5.5 ADC 数据采集

本方案使用 SARADC 进行数据采集。采集的数据是血氧仪接收到，并经过 TIA 转换和低通滤波处理的电压信号。另外，软件每 30 秒切换采样通道，采集电池电压信号。

SARADC 软件模块主要通过寄存器 REG_ADC_MUX_IN 配置输入信号源和通路配置。下图是 ADC 输入信号通路选择和配置示意图。

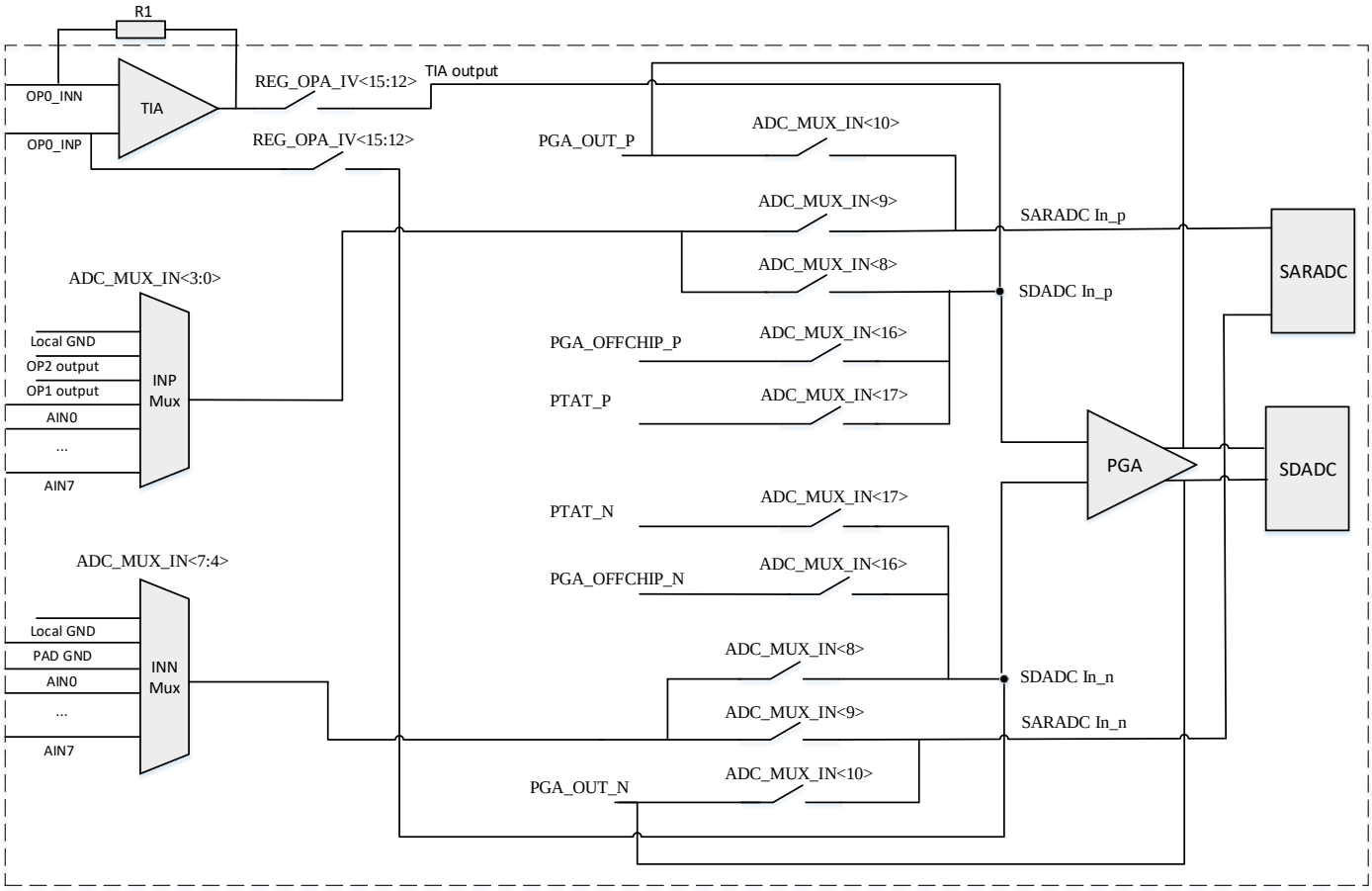

图 17 ADC 输入信号通路配置示意图

说明：

1、 TIA 转换电阻 R1 需要用户外接；

2、 PGA_OUT_P 和 PGA_OUT_N 是 PGA 的差分信号输出端；

3、 PTAT_P 和 PTAT_N 是片内温度传感器输出信号；

4、 PGA_OFFCHIP_P 和 PGA_OFFCHIP_N 是将信号通过 IO 脚（PA02/PA03）输入到 PGA 的开关；
5、 SARADC In_p 和 SARADC In_n 是 SARADC 的信号输入端；
6、 SDADC In_p 和 SDADC In_n 是 PGA 的差分信号输入端；
7、 TIA 的输出可以由寄存器 REG_OPA_IV 配置到 SDADC In_p 和 SDADC In_n；

SARADC 主要配置如下：

1、 参考电压配置为 VDD，SARADC_REF_SELECT_VDD；
2、 关闭 PGA 功能；
3、 转换通道为单通道，信号为单端信号。P 端连接 PA00，N 端接地；
4、 采样模式为连续采样，一次触发采样 16 个数据；
5、 FCLK 为 16M 内部时钟，不分频；
6、 采样时间配置为 14 个 CLK，等待周期 21 个 CLK，数据采样率 457K；
7、 使用 DMA 功能；
8、 adc_fifo_thrhold 配置为 2 个 WORD，与 DMA burst size 对应；

SARADC 示例代码如下：

```
ANAREG_1->reg_adsar_top = SARADC_REF_SELECT_VDD; // reference voltage

// PA00, SpO2 Sample pin
ANAREG_1->reg_adsar_mux_in_f.adc_ext_input_p = SARADC_AIN0_PA00;
ANAREG_1->reg_adsar_mux_in_f.adc_ext_input_n = SARADC_INN_GND_PAD;
ANAREG_1->reg_adsar_mux_in_f.adc_sar_sel = 1;

ANAREG_1->reg_adc_conf1_f.adc_fifo_format = 0x01;  /* adc_fifo_format */
ANAREG_1->reg_adc_conf1_f.adc_smp_cycle = 14;      /* adc_smp_cycle */
ANAREG_1->reg_adc_conf1_f.adc_wait_cycle = 21;     /* adc_wait_cycle */
ANAREG_1->reg_adc_conf1_f.adc_dma_en = 0x01;       /* adc_dma_en */
ANAREG_1->reg_adc_conf1_f.adc_fifo_thrhold = 0x02; /* adc_fifo_thrhold */
ANAREG_1->reg_adc_conf1_f.adc_trig_num = 0x02;     /* adc_trig_num */
ANAREG_1->reg_adc_conf1_f.adc_scan_chnum = 0x00;   /* adc_scan_chnum */
ANAREG_1->reg_adc_conf1_f.adc_irq_mode = 0;
```

5.6 SPI 配置

本方案 OLED 屏与 TCAS24A 通过 SPI0 接口进行连接。为了更快的刷新屏幕内容，提升良好的交互体验，SPI 通信需要在允许的范围内配置尽可能快的通信速度。TCAS24A 支持 8MHz，及最高 16MHz 的 SPI_FCLK_OUT 速率。

SPI 的速率计算公式如下：

$$SPI\_FCLK\_OUT = SPI\_FCLK/(CPSDVR*(1+SCR))$$

其中：

SPI_FCLK_OUT：SPI CLK 线的频率。

SPI_FCLK： SPI 的输入时钟频率。

CPSDVR： 时钟预分频因子，2-254 之间的偶数。

SCR： SPI clock 频率设置参数，最小为 0。

8MHz SPI_FCLK_OUT 的配置示例代码：

```
static void SYS_SPI0_Init(void)
{
    /* 标准 4 线，主机,CPOL= 0,CPHA = 0,数据位 8 */
    SPI_Config config =
    {
        .spi             = SPI0,
        .slave           = 0,
        .loopback        = 0,
        .CPOL            = 0,
        .CPHA            = 0,
        .frame_size      = 8,
        .imsc            = 0,          /* 中断关掉 */
        .dmaConfig.txDmaEn = 0,
        .dmaConfig.rxDmaEn = 0,
    };

    SPI_Init(config);
}
void SPI_Init(SPI_Config config)
{
    SYSCTL->SPI0_CLKRST_CTRL_f.sw_fclk_en_spi = ENABLE;
    SYSCTL->SPI0_CLKRST_CTRL_f.sw_fclk_sel_spi = 0;     // RC16M
    SYSCTL->SPI0_CLKRST_CTRL_f.sw_fclk_div_spi = 0;
    /* ssp 速率 FSPICLKOUT = FSPICLK/(CPSDVR*(1+SCR)) = 16000000/(2*(1+0))=8M */
    // Set DSS data to 8-bit, Frame format SPI, CPOL = 0, CPHA = 0
    config.spi->CR0 = 0x00000007;
    SPI_SetRfr(config.spi,config.frf);
    SPI_SetFrameSize(config.spi, config.frame_size);
    SPI_SetCpolCpha(config.spi,config.CPOL,config.CPHA);

    // SPICPSR clock prescale register, master mode, minimum divisor is 0x02.
    config.spi->CPSR = 2;
    ...
}
```

说明：

1. 配置 SPI 时钟时，需满足 SPI_FCLK 小于等于系统 PCLK 的条件。
2. PCLK 配置说明：

PCLK_FREQ=HCLK_FREQ/(SW_PCLK_DIV+1)

SW_PCLK_DIV 缺省是 3，可以通过配置修改，例如将 PCLK 配置与 HCLK 相等，分频系数为 0：

SYSCTL->AHB2APB_CLKRST_CTRL_F.SW_PCLK_DIV_BRIDGE = 0;

5.7 FLASH 存储

本方案采用片内 FLASH 来保存数据。系统上电时会从片内 FLASH 读取数据，如果数据校验不对，会写入默认参数。软件运行过程中，会根据用户操作，实时保存数据和参数。

保存的数据内容包括：

1. 告警、显示、SpO2 阈值等参数；
2. 用户通过 UI 配置的数据；

为了提升代码执行速度，可以在系统初始化的时候配置 FLASH 读时间寄存器 RCTC，使得读 FLASH 读速度最快。

配置读 FLASH 时间的示例代码：

```
#if defined   ( __ICCARM__   ) /* iar */
#pragma location = "RAMUSERCODE"
#elif defined ( __CC_ARM )     /* keil */
    __attribute__((section("RAMCODE")))
#endif
void Flash_ConfigRCTC(uint8_t rcTime)
{
    uint32_t rctcData;

    rctcData = M0P_FLASH->RCTC;
    rctcData = M0P_FLASH->RCTC;

    rctcData &= 0xfff0;
    rctcData |= rcTime;
    M0P_FLASH->RCTC = rctcData;      /** Read time (HCLK) ,  reset:1*/
}
```

注意：

MCU 主频小于 30MHz 时，rcTime 最小可以配置为 0；否则，rcTime 最小可以配置为 1。

说明：

1. 写数据到 FLASH 之前，需要先调用 Flash_Init()函数，使能 FLASH 写功能；保存完成后需要调用 Flash_DeInit()函数关闭 FLASH 写功能；
2. 写 FLASH 的代码需要在 RAM 执行；
3. 写 FLASH 过程中需要关中断；

6. 模块测试

根据 TCAS 系列 MCU 的血氧仪设计方案，我们对部分模块进行了性能测试。测试结果显示性能可以满足血氧仪产品需求。

6.1 LED 发送测试

PC02 与 PC01 引脚分别控制红光与红外光 LED 通道的开关，时序如下：

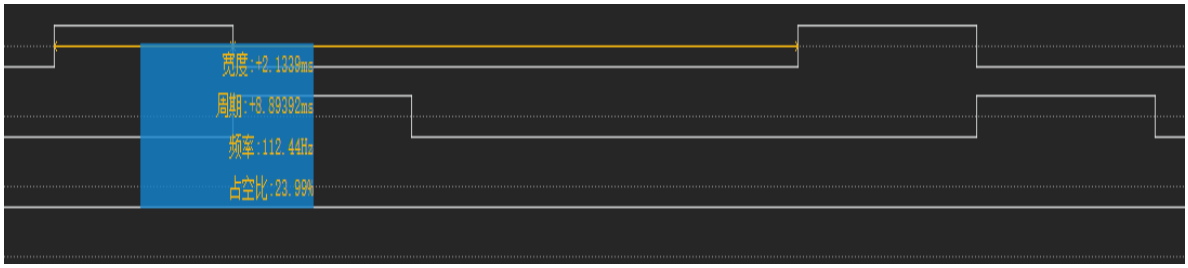

图 18 红光通道时序

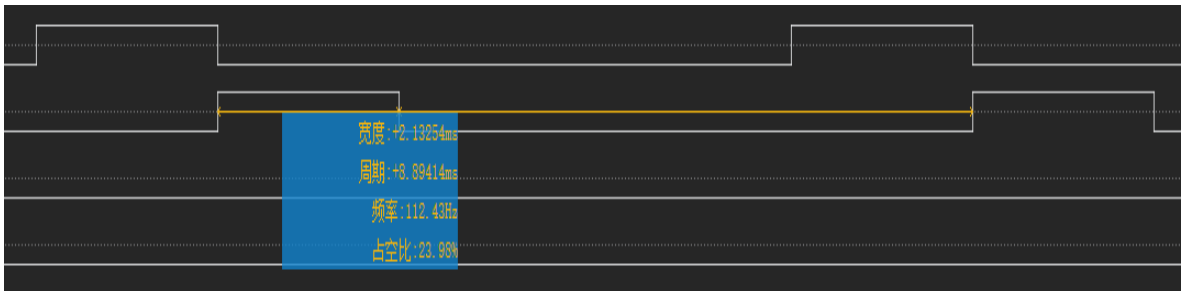

图 19 红外光通道时序

PB00、PB01 是 TIMER4 通道 CHA 和 CHB，输出互补型 PWM 波形，用于控制 LED 驱动电流的大小。PWM 波形如下：

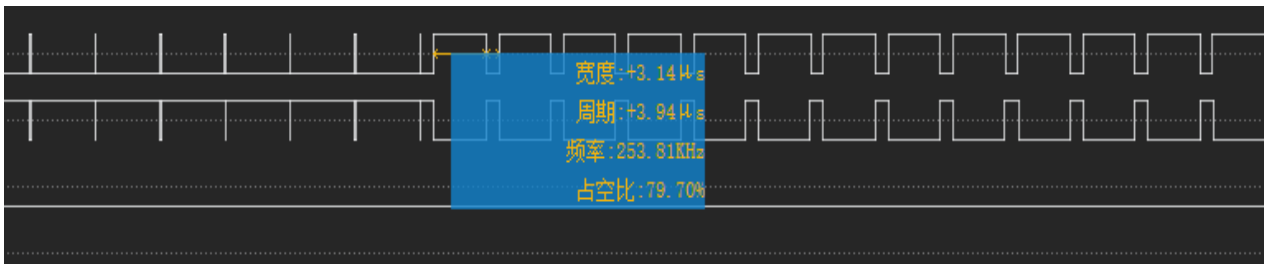

图 20 互补性 PWM 输出波形

6.2 SPI 输出（显示）测试

基于 8M 速率的 SPI 通信，用于控制 OLED 显示，包括命令通信和数据通信。SPI 通信时序如下：

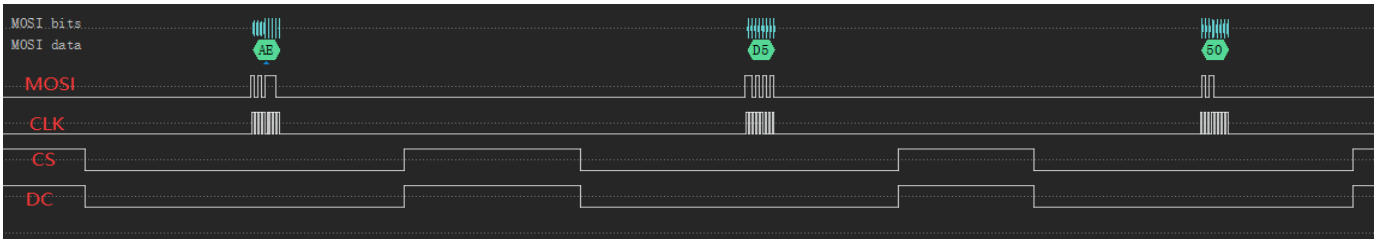

图 21 OLED 命令时序

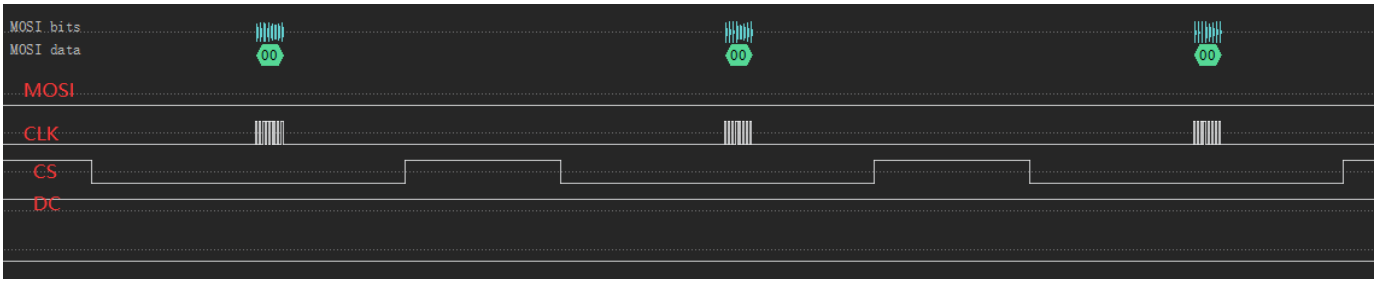

图 22 OLED 数据时序

7. 使用 MCU 的注意事项

1. LPUART 位于 AON 电源域，使用的 APB2 时钟为 32KHz。由于 PCLK 时钟较慢，CPU 操作寄存器时会占用 APB 总线，影响系统的实时性。建议在血氧仪方案设计时，不要使用 LPUART。可以将 LPUART 只作为唤醒源或者上电检测使用。
2. AON_GPIO 位于 AON 电源域，使用了 APB2 时钟，为 32KHz。由于 PCLK 时钟较慢，CPU 操作寄存器时会影响系统的实时性。建议在血氧仪方案设计时，将 AON_GPIO 使用在实时性要求不高的场景。例如，可以将 AON_GPIO 作为按键或者电源控制使用。
3. 配置 SPI 时钟时，需满足 SPI FCLK 小于等于 PCLK 的条件。最大 SPI FCLK 可以达到 16MHz。
4. VDAC 无法快速切换不同的电压挡位，来满足不同的 LED 驱动电流大小。建议用 PWM 来替代 VDAC 的功能。
5. 因为 IDAC 默认不校准，精度不够，建议 LED 驱动电流不要使用 IDAC 来控制。
6. TCAS24A 的跨阻放大器（TIA）不是轨到轨类型，建议用户采用外部阻容电路提供正端 0.2V 以上共模电压。
7. 用户关机时，由于按键按下 IO 口，PA06 获得的电压为电池电压，在电池电压低于 TCAS24A 的逻辑高电平范围时 MCU 无法获取当前按键的状态，因此本方案采用比较器的方式实现按键的开关机电路。
8. 建议用户使用芯片片内 FLASH 的 NVR 区域保存数据。
9. GPIO 翻转速度：在 HCLK 配置为 16MHz 时，TOP_GPIO 的最大翻转速度可以达到 320KHz；AON_GPIO（PA02~PA07）的最大翻转速度可以达到 4.2KHz。
10. SPI 的时钟配置，不要使用以下这种配置：HCLK32M\PCLK 8M\SPI FCLK32M\SPI CLK 8M。

8. 参考样例及驱动

Tinychip 官方 SDK 提供了基于 TCAS 系列 MCU 的血氧仪设计方案的应用样例及驱动库，同时支持 KEIL 和 IAR 两种开发环境。详见 TCAS 系列血氧仪方案 SDK 软件包。

9. 版本

版本	日期	备注
V1.0	2021-10-22	初始版本

10. 关于我们

上海泰矽微电子有限公司 2019 年成立于上海张江，是一家中国领先的高性能专用 SoC 芯片供应商。公司专注于物联网应用相关的各类芯片的研发，已获得多个知名投资机构的大力扶持与投资。公司聚集了一批顶尖的半导体专家，致力于发展成为平台型芯片企业。团队具有各类系统级复杂芯片的研发能力，所开发的芯片累计出货达数十亿颗。公司已在信号链、电源及射频等方向积累了大量的 SoC 芯片方案，可覆盖消费类，工控及汽车等应用领域。差异化的芯片产品在树立行业标杆的同时，也将为更多物联网企业赋能，更好服务于客户需求。

上海泰矽微电子有限公司

地址：上海浦东新区纳贤路 800 号 1 幢 A 座 602 室

南京市雨花台区软件大道 170-1 号天溯科技园 1 栋 508 室

网址：http://www.tinychip.com.cn

技术支持窗口

电邮：support@tinychip.com.cn