应用笔记

Application Note

文档编号：**AN_1105**

APM32F407 TMR1 调制呼吸灯应用笔记

版本：**V1.0**

1 引言

本应用笔记提供用户在 APM32F407 系列上配置和应用定时器的指南，包括应用方法和实现代码。

本文以 APM32F407 微控制器 TMR1 来实现呼吸灯，APM32F407 微控制器内置 14 个定时器，其中包括 2 个高级定时器（TMR1、TMR8）、2 个基本定时器（TMR6、TMR7）和 10 个通用定时器。定时器和定时器之间是独立的，根据单片机内部的设计，定时器和定时器之间可以实现同步和级联。定时器最基本的功能是为单片机提供时间基准，通过配置可以产生 DMA 请求。

目录

2 定时器简介

在微控制系统（Microcontroller）中，定时器（Timer）是微控制器不可或缺的外设，用于在特定的时间间隔内执行任务、测量时间、生成时序信号和控制各种时间相关的操作。定时器对于控制系统、嵌入式系统和实时系统非常重要，主要用途包括延时、时间测量、定期中断、PWM（脉冲宽度调制）生成、计数和时间基准等。

定时器通常由一个计数器寄存器和一些控制寄存器组成。计数器寄存器用于保存计数器的当前值，而控制寄存器用于配置定时器的工作模式、时钟源、中断使能等。计数器根据时钟源的脉冲递增，当计数器达到预定的值时，可以触发中断或执行其他操作。

定时器是嵌入式系统中的关键组件，它们提供了精确的时间测量和控制能力，可以实现多种时间相关的功能，对于控制和自动化应用非常重要。

2.1 时基单元

定时器的时基单元通常包括 3 个寄存器：

- 16 位计数器寄存器
- 16 位自动重装载寄存器
- 16 位预分频寄存器

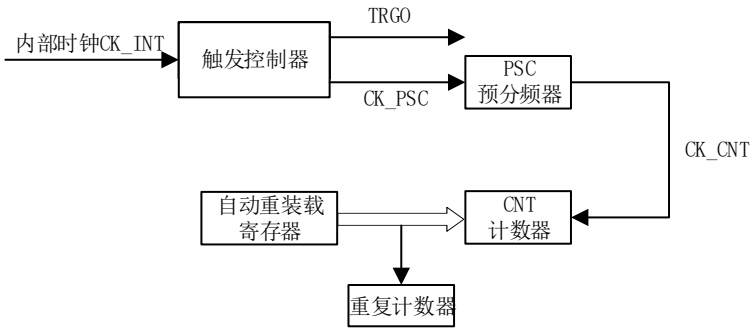

图 1 时基单元框图

所有定时器时钟频率分配由硬件按两种不同情况自动设置：

（1） 若相应的 APB 预分频系数是 1，定时器的时钟频率与所在 APB 总线频率一致。

（2） 相应的 APB 预分频系数不为 1，定时器的时钟频率被设为与其相连的 APB 总线频率的 2 倍。

在 APM32F407 微控制器中，系统时钟频率最大为 168MHz，APB1 的预分频系数为 4，APB2 的预分频系数为 2，根据 APM32F407 系统框架图得知，TMR1/8/9/10/11 与 APB2 总线相连，TMR2/3/4/5/6/712/13/14 与 APB2 总线相连，所以定时器的时钟频率分别为：

TMR1/8/9/10/11:

$$CK\_INT = \frac{168MHz}{2} * 2 = 168MHz \quad （式 2-1）$$

TMR2/3/4/5/6/712/13/14:

$$CK\_INT = \frac{168MHz}{4} * 2 = 84MHz \quad （式 2-2）$$

16 位计数寄存器存放的是计数器的计数值，16 位自动重装载寄存器存放的是自动重装载值，当自动重装载值为空时，计数器不进行计数，16 位预分频寄存器存放的是预分频器数值，根据预分频器数值可计算出计数器的时钟频率（CK_CNT）。

$$CK\_CNT = \frac{CK\_PSC}{PSC+1} \quad （式 2-3）$$

当禁止从模式控制器时，预分频的时钟源 CK_PSC 由内部时钟 CK_INT 驱动, 则有:

$$CK\_CNT = \frac{CK\_INT}{PSC+1} \quad （式 2-4）$$

此外，高级定时器还有一个特有的寄存器：8 位重复计数寄存器。重复计数寄存器是 APM32F407 的高级定时器独有的寄存器，该寄存器存放的是重复计数数值（REPCNT），其主要作用是当定时器发生上溢或下溢时，重复计数值（REPCNT）减 1，只有当重复计数寄存器的值为 0 时，定时器发生上溢或下溢，才会产生更新事件。

2.2 计数模式

计数器是定时器的重要组成部分，计数器的计数模式对定时器的功能有重要影响。在 APM32F407 中，根据定时器的不同，计数器的计数模式也不同。对于定时器 TMR6/7/9/10/11/12/13/14 计数模式只有一种：向上计数模式。对于定时器 TMR1/2/3/4/5/8 计数器有 3 种计数模式：向上计数模式、向下计数模式、中央对齐模式。

向上计数模式：当计数器使能时，计数器从 0 开始计数，每接收到一个 CK_CNT 的脉冲，计数器的值就增加 1，直到计数器的值从 0 增加到与自动重装载值相等，此时定时器产生一个更新事件。然后计数器重新从 0 开始向上计数，如此循环往复。

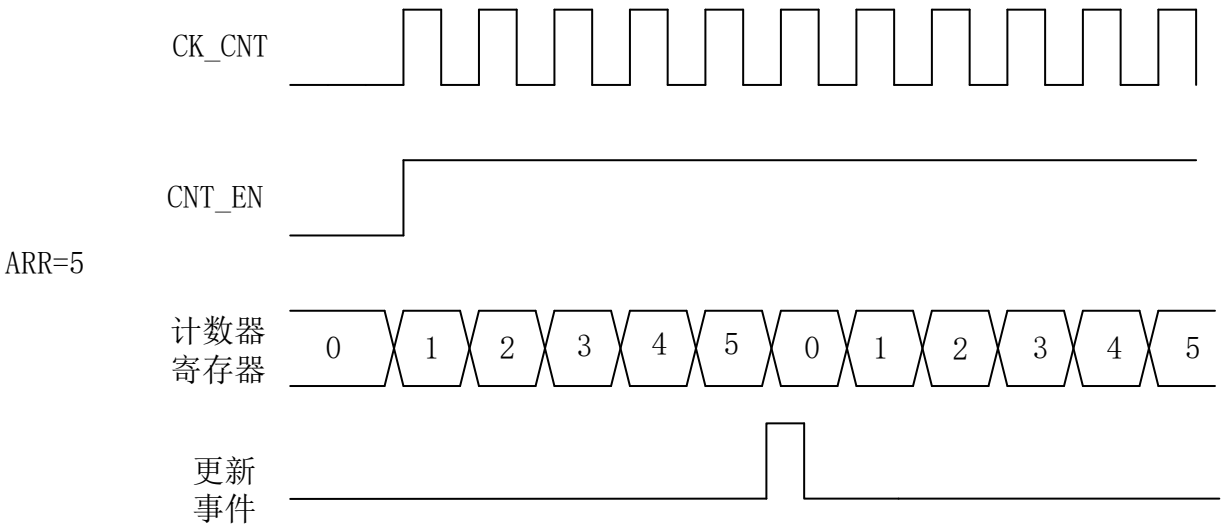

图 2 向上计数模式时序图

向下计数模式：当计数器使能时，计数器从自动重装载值开始计数，每接收到一个 CK_CNT 的脉冲，计数器的值就减 1，直到计数器的值从自动重装载值递减到 0，此时定时器产生一个更新事件。然后计数器重新从自动重装载值开始向下计数，如此循环往复。

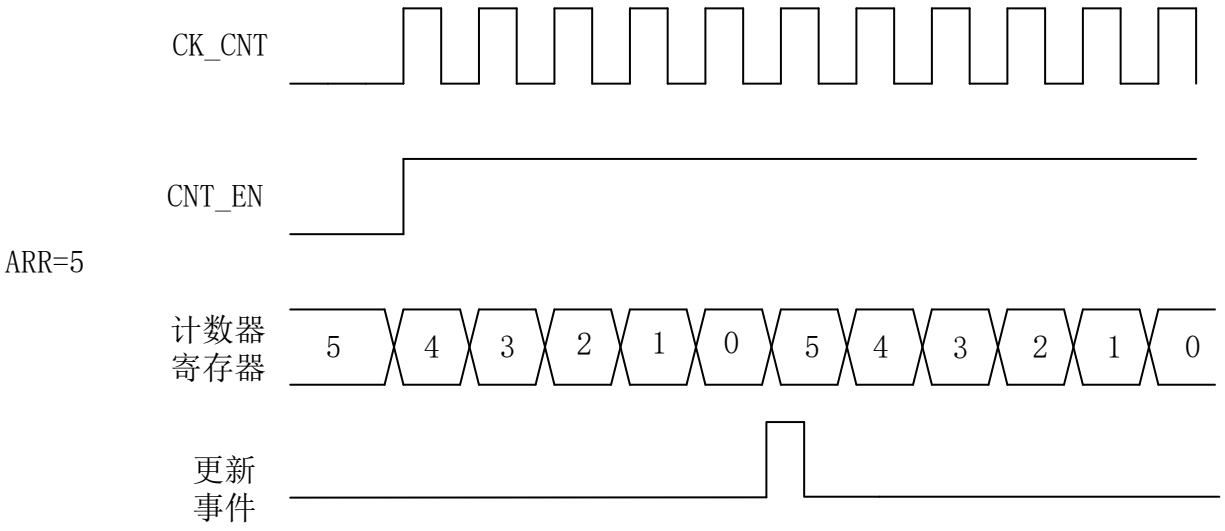

图 3 向下计数模式时序图

中心对齐模式：当计数器使能时，计数器从 0 开始计数，每接收到一个 CK_CNT 的脉冲，计数器的值就增加 1，当计数器的值等于（自动重装载值-1）时，此时定时器产生一个更新事件。当计数器的值递减到 0，计数器的值从 0 递增到与自动重装载值相等。然后计数器从自动重装载值向下计数，每接收到一个 CK_CNT 的脉冲，计数器的值就减 1，直到计数器的值递减到 1，此时定时器产生一个更新事件，计数器又从 0 开始向上计数，如此循环往复。

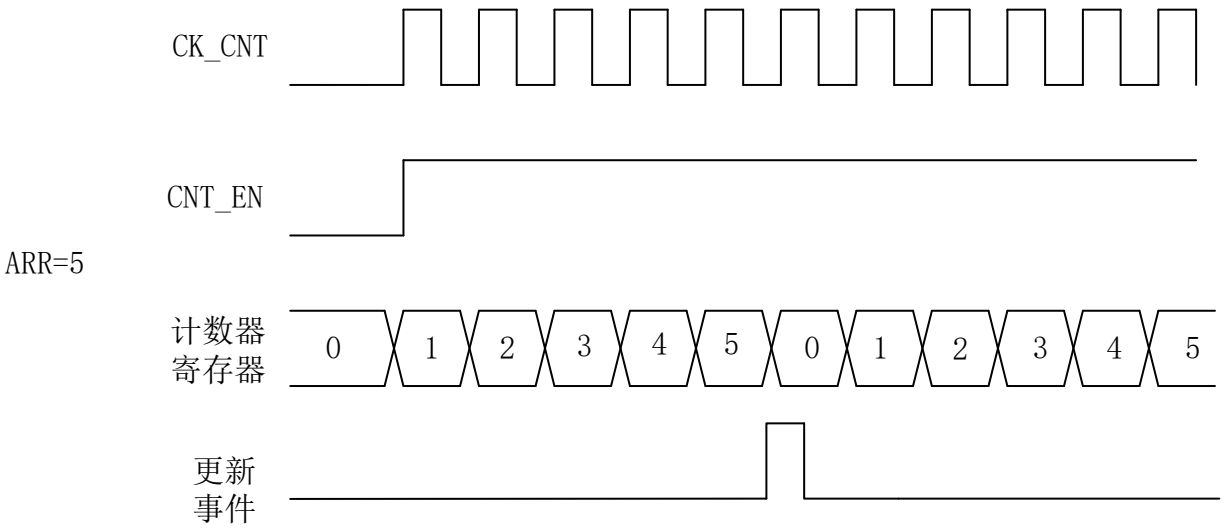

图 4 中心对齐模式时序图

在 APM32F407 微控制器中，定时器的中心对齐模式允许 PWM 信号的占空比在计数器的上升沿和下降沿之间变化，这使得 PWM 信号可以在整个计数器周期内进行平滑的占空比变化，这种模式提供了精确的占空比控制，同时也有助于减少电磁干扰和噪音。不同的中心对齐模式，输出通道的输出比较中断标志位置 1 的时机也不同。通过操作控制寄存器 1（TMRx_CTRL1）的 CNTDIR 位和 CAMSEL 位可实现对定时器计数模式的控制。

中心对齐模式 1：计数器交替进行递增和递减计数，当且仅当计数器递减计数时，输出通道的输出比较中断标志位置 1。

中心对齐模式 2：计数器交替进行递增和递减计数，当且仅当计数器递增计数时，输出通道的输出比较中断标志位置 1。

中心对齐模式 3：计数器交替进行递增和递减计数，当计数器递增或递减计数时，输出通道的输出比较中断标志位置 1。

3 定时器输出比较

3.1 简介

定时器的输出比较功能是指定时器模块与外部输入或其他内部事件进行比较，并在比较条件满足时执行相应的操作。这个功能通常用于产生精确的时间延迟、生成 PWM 信号、控制外部设备等应用。

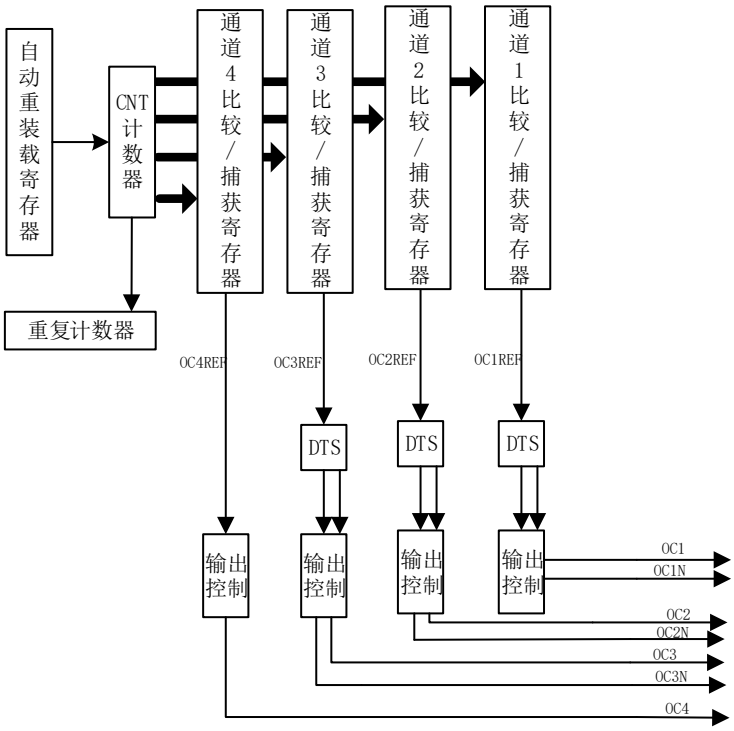

图 5 定时器输出比较结构框图

APM32F407 微控制器定时器通常提供 4 个独立比较通道，每个比较通道都有一个相关的比较寄存器和比较值。定时器的输出比较功能支持多种模式，通过配置 TMRx_CCMx 的 OCxMOD 控制输出比较通道的模式，包括以下几种常见的：

⑴输出比较模式：定时器的输出与比较结果相关联。当计数器的值与比较寄存器中的值相等时，可以触发一个输出事件，例如使输出比较信号发生翻转或强制输出比较信号为高（低）电平。当计数器的值与比较寄存器中的值相等时，可以触发中断请求，允许在中断服务程序中执行相应的操作。

⑵强制输出模式：无论输出比较值和计数值是何种比较结果，将输出比较信号强制设置为高（低）电平。当计数器的值与比较寄存器中的值比较仍然会执行，可以触发中断请求。

⑶PWM 模式：输出比较功能通常用于生成 PWM 信号。通过将比较通道的比较值设置为合适的值，可以控制 PWM 信号的占空比和频率。

通过配置 TMRx_BDT 的 DTS 位设置互补输出通道的死区持续时间，死区互补功能用于确保两个相对的开关器件不会同时导通，以避免短路和损坏。死区是两个开关器件之间的一段时间，其中两者都处于关闭状态。互补则是指当一个开关器件导通时，另一个处于断开状态，反之亦然。在 APM32F407 中，死区互补功能通常与 PWM 生成和输出比较通道结合使用，使用输出比较通道来配置死区时间，确保两个开关器件之间有足够的时间间隔，从而避免导通冲突。

3.2 PWM 输出模式

PWM 输出模式是输出比较的一种，由定时器对外输出可调节的脉冲信号，脉冲信号的频率由定时器的自动重装载值决定，脉冲信号的占空比由比较值决定。在 APM32F407 中，有 2 种 PWM 输出模式：PWM 模式 1 和 PWM 模式 2。

PWM 模式 1 是指当计数值（CNT）小于比较值（CCx）时，输出有效电平，否则输出无效电平。

PWM 模式 2 是指当计数值（CNT）小于比较值（CCx）时，输出无效电平，否则输出有效电平。有效电平的极性通过配置 TMRx_CCEN 的 CCxPOL 位进行控制。根据 PWM 模式和计数模式的不同，产生的 PWM 信号也不同。

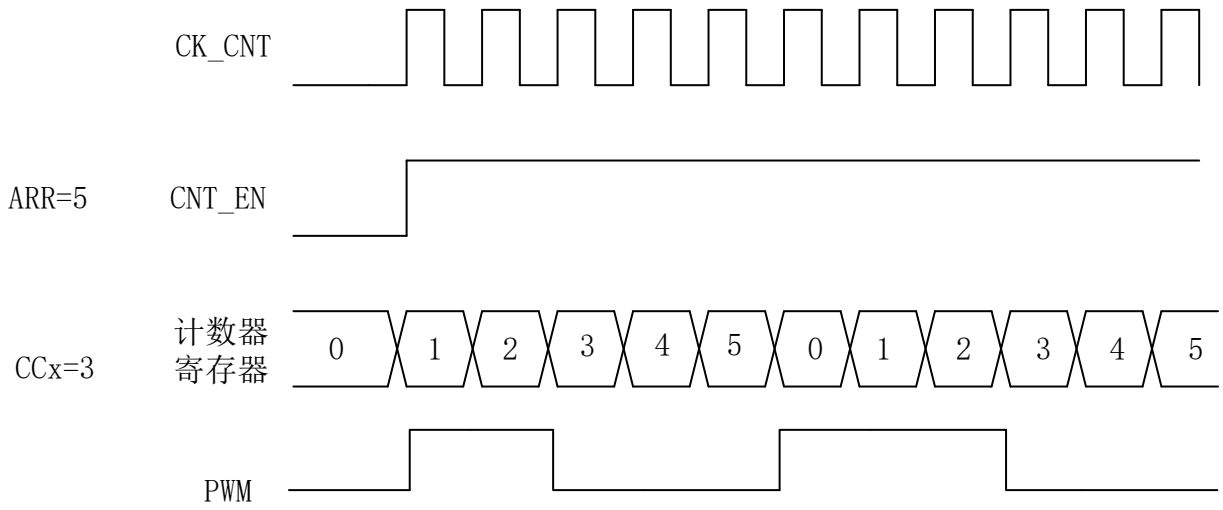

图 6 PWM 模式 1 下的向上计数时序图

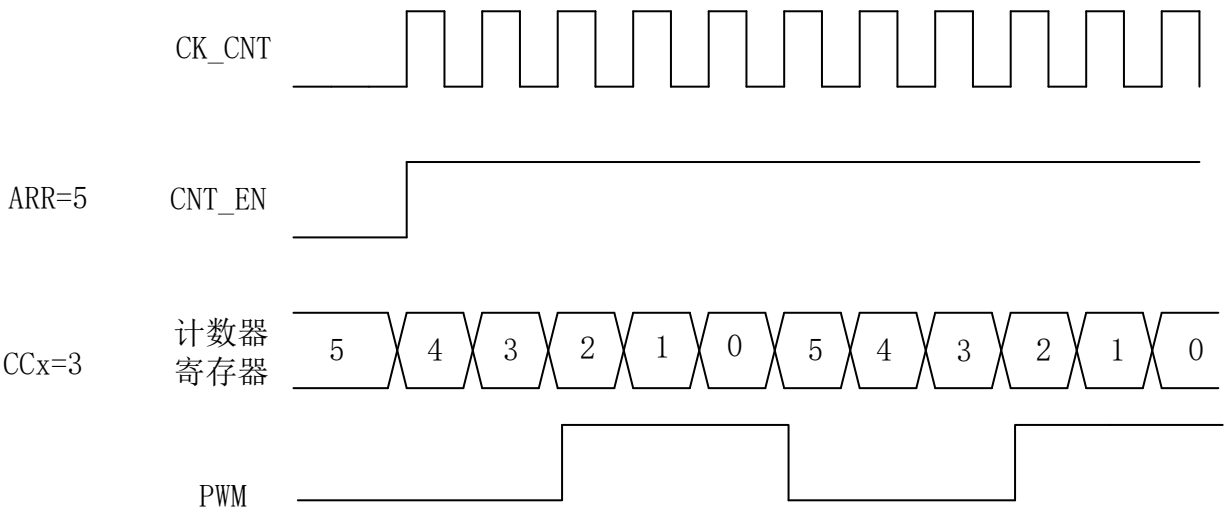

图 7PWM 模式 1 下的向下计数时序图

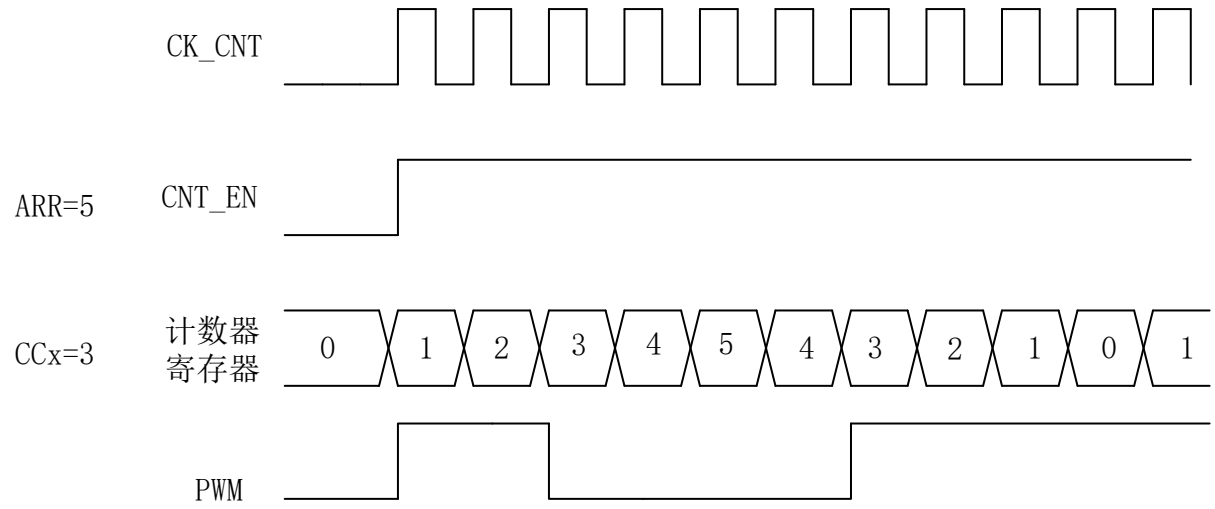

图 8PWM 模式 1 下的中心对齐模式时序图

PWM 输出模式通过改变脉冲的占空比（高电平与总周期的比例）来控制输出。占空比增加时，平均输出电压或电流也增加，反之亦然。优点：简单易实现，需要的硬件资源较少。适用于大多数基本应用，如电机控制和 LED 亮度调节，但分辨率较低，无法实现非常精细的控制。

4 APM32F407 TMR1 调制呼吸灯例程

呼吸灯（Breathing LED）是一种常见的 LED 灯效模式，其效果类似于生物体的呼吸节律，即逐渐变亮然后逐渐变暗，形成一种周期性的呼吸效果。呼吸灯效果能够产生一种柔和的渐变光效果，是一种简单而有效的方式来改善产品的外观和用户体验，尤其在需要指示状态或吸引用户注意力的场合非常实用。

LED 灯的亮度是由电压决定的，LED 两端电压越大，亮度越大。使用 PWM 输出来控制 LED 的亮度实现呼吸灯，实际上就是控制 LED 灯两端的模拟电压。在开始时，设置 PWM 的频率和占空比，通过对占空比的增加或减小，来改变一个周期内高电平的时间，也就是改变这个周期内 LED 两端的模拟电压大小。呼吸灯效果的核心是在一定的时间内逐渐改变 PWM 的占空比，从而控制 LED 的亮度。

开始时，逐渐增加 PWM 占空比，使 LED 逐渐变亮。当 LED 达到最大亮度时，开始逐渐减小 PWM 占空比，使 LED 逐渐变暗。一旦亮度减小到设定的最小值，就反向操作，重新逐渐增加亮度。这样就形成了一个循环。通过调整 PWM 的频率、最大亮度、最小亮度、呼吸周期等参数来改变呼吸灯的效果。

4.1 硬件连接

本例程需要用到 TMR1 通道 1 和 LED 灯，MINI Board 已经将 TMR1 的通道接出，可以根据设计需求使用相关 IO。用到的 LED 灯原理图如下：

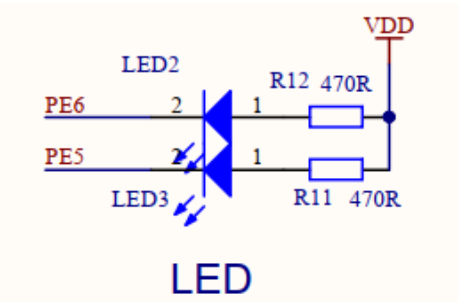

图 9LED 原理图

由原理图可知，当 PE5、PE6 段端为低电平时，LED 灯被点亮，为高电平则 LED 则为熄灭状态。将 TMR1 通道 1 对应的 IO（PA8）连接到 LED 灯相应的 IO，就完成了硬件连接。

4.2 软件设计

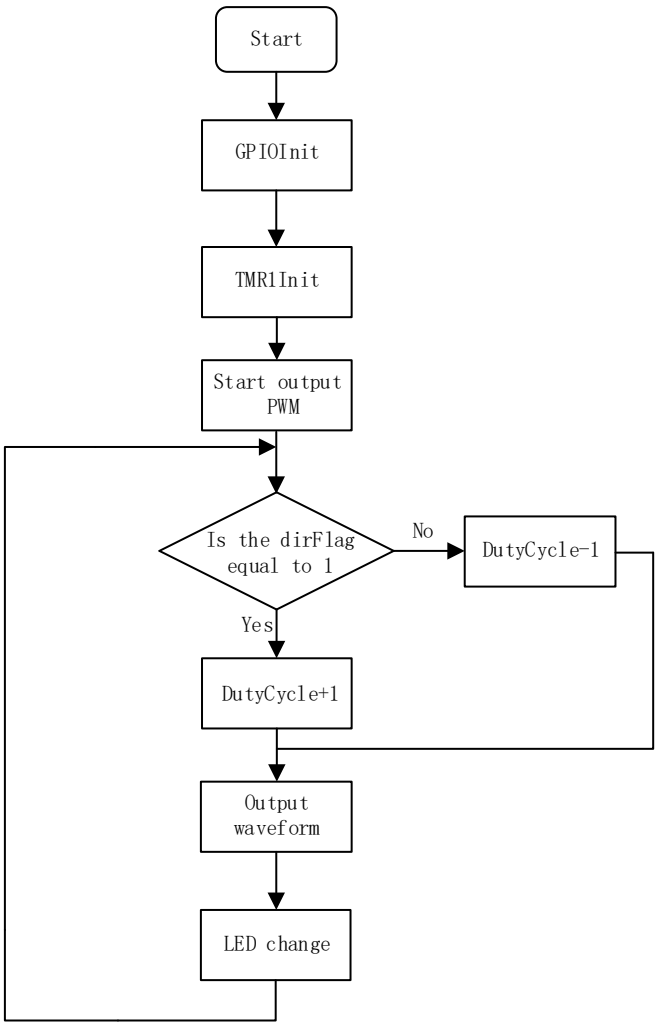

图 10 程序设计流程图

初始化 TMR1 通道 1 的 IO 和 LED 灯的 IO，初始化定时器，配置定时器的基础参数，通道 1 参数，在中断中对输出的 PWM 波进行脉冲调制。

4.2.1 引脚配置

将 TMR1 通道 1 所使用到的 GPIO 进行配置，GPIOA_PIN_8 配置为端口复用模式。将 LED 灯的 GPIO 配置为推挽输入模式，参考代码如下：

```
void GPIO_Init()
{
    GPIO_Config_T GPIO_ConfigStruct;

    GPIO_ConfigPinAF(GPIOA, GPIO_PIN_SOURCE_8, GPIO_AF_TMR1);
```

```
    /* Config PA8 for output PWM */
    GPIO_ConfigStruct.pin = GPIO_PIN_8;
    GPIO_ConfigStruct.mode = GPIO_MODE_AF;
    GPIO_ConfigStruct.otype = GPIO_OTYPE_PP;
    GPIO_ConfigStruct.speed = GPIO_SPEED_100MHz;
    GPIO_Config(GPIOA, &GPIO_ConfigStruct);

    GPIO_ConfigStruct.pin = GPIO_PIN_5;
    GPIO_ConfigStruct.mode = GPIO_MODE_IN;
    GPIO_ConfigStruct.otype = GPIO_OTYPE_PP;
    GPIO_ConfigStruct.speed = GPIO_SPEED_50MHz;
    GPIO_Config(GPIOE, &GPIO_ConfigStruct);
}
```

4.2.2 定时器初始化

4.2.2.1 TMR1 结构体

TMR_BaseConfig_T 结构体定义在 APM32F4xx_dmc.h 文件中，具体定义如下：

```
typedef struct
{
    TMR_COUNTER_MODE_T  countMode;
    TMR_CLOCK_DIV_T     clockDivision;
    uint32_t            period;
    uint16_t            division;
    uint8_t             repetitionCounter;
} TMR_BaseConfig_T;
```

结构体中参数的含义：

countMode：定时器的计数模式有向上计数模式、向下计数模式、中心对齐模式 1、中心对齐模式 2、中心对齐模式 3 可供选择。

clockDivision：时钟分频系数选择，通过设置改位可调整死区时间的采用时钟。

period：自动重装载值，可以选择 0x0000 到 0xFFFFFFFF 的任意值。

division:预分频器数值，可以选择 0x0000 到 0xFFFF 的任意值。

repetitionCounter：重复计数值，只有 TMR1 和 TMR8 可设置该位，可以选择 0x00 到 0xFF 的任意值。

4.2.2.2 定时器配置

开启 GPIO 和 TMR1 时钟后，初始化 GPIO，然后配置定时器时基和通道 1，具体代码如下：

```
void TMR_Init()
{
    TMR_BaseConfig_T TMR_TimeBaseStruct;
    TMR_OCConfig_T OCcongigStruct;

    /* config TMR1 */
    TMR_TimeBaseStruct.clockDivision = TMR_CLOCK_DIV_1;
    TMR_TimeBaseStruct.countMode = TMR_COUNTER_MODE_UP;
    TMR_TimeBaseStruct.division = DIV;
    TMR_TimeBaseStruct.period = 999;

    TMR_ConfigTimeBase(TMR1, &TMR_TimeBaseStruct);
```

配置 TMR1 的计数模式为向上计数模式，死区时间 $t_{DTS}=t_{CK\_INT}$，自动重装载值为 999，预分频值为 167，重复计数数值为 0，则 TMR1 的频率为：

$$f = \frac{CK\_PSC}{PSC+1} * \frac{1}{999+1} = \frac{168MHz}{(167+1)*1000} = 1kHz \qquad (式\ 4\text{-}1)$$

为了使 TMR1 输出 PWM 波，则需配置通道 1，具体代码如下：

```
    OCcongigStruct.idleState = TMR_OC_IDLE_STATE_RESET;
    OCcongigStruct.mode = TMR_OC_MODE_PWM2;
    OCcongigStruct.nIdleState = TMR_OC_NIDLE_STATE_RESET;
    OCcongigStruct.nPolarity = TMR_OC_NPOLARITY_HIGH;
    OCcongigStruct.outputNState = TMR_OC_NSTATE_DISABLE;
    OCcongigStruct.outputState = TMR_OC_STATE_ENABLE;
    OCcongigStruct.polarity = TMR_OC_POLARITY_HIGH;
    OCcongigStruct.pulse = 1;
    TMR_ConfigOC1(TMR1, &OCcongigStruct);

    TMR_Enable(TMR1);

    TMR_EnablePWMOutputs(TMR1);
```

配置通道 1 的 PWM 模式为 PWM 模式 2，配置通道 1 和通道 1 的互补通道输出空闲点

平为低电平，配置通道 1 和通道 1 的互补通道输出极性为高电平，使能通道 1 输出，设置通道 1 比较值为 1。

使能 TMR1 更新中断，设置 TMR1 更新中断的响应优先级：

```
    TMR_EnableInterrupt(TMR1, TMR1_UP_TMR10_IRQn);
    NVIC_EnableIRQRequest(TMR1_UP_TMR10_IRQn, 0, 0);
}
```

4.2.2.3 呼吸功能函数

定时器每产生一次更新事件，就执行一次中断服务函数，在中断服务函数中，调用呼吸功能函数，在呼吸功能函数中先清除中断标志位，根据方向标志位确定占空比递增还是递减，当占空比递增到最大或递减到最小时，改变方向标志位的值，具体代码如下：

```
void Breath()
{
    TMR_ClearIntFlag(TMR1, TMR_INT_UPDATE);

    if(dirFlag)
    {
        dutyCycle++;
        TMR_ConfigCompare1(TMR1, dutyCycle);
        if(dutyCycle == 999)
        {
            dirFlag = 0;
        }
    }
    else
    {
        dutyCycle--;
        TMR_ConfigCompare1(TMR1, dutyCycle);
        if(dutyCycle == 0)
        {
            dirFlag = 1;
        }
    }
}
```

5 总结

在 APM32F407 微控制系统中，使用定时器 1 的输出比较功能，通过对占空比的调制，使得输出的 PWM 波的占空比递增或递减，高电平的时间在逐渐增大或逐渐减少。将输出的 PWM 信号连接到 LED 灯，LED 灯两端电压逐渐增大或逐渐减小，从而调节 LED 灯的亮度，实现呼吸灯。

6 版本历史

表格 1 文件版本历史

日期	版本	变更历史
2023.09.27	1.0	新建

声明

本手册由珠海极海半导体有限公司（以下简称“极海”）制订并发布，所列内容均受商标、著作权、软件著作权相关法律法规保护，极海保留随时更正、修改本手册的权利。使用极海产品前请仔细阅读本手册，一旦使用产品则表明您（以下称“用户”）已知悉并接受本手册的所有内容。用户必须按照相关法律法规和本手册的要求使用极海产品。

1、权利所有

本手册仅应当被用于与极海所提供的对应型号的芯片产品、软件产品搭配使用，未经极海许可，任何单位或个人均不得以任何理由或方式对本手册的全部或部分内容进行复制、抄录、修改、编辑或传播。

本手册中所列带有“®”或“TM”的“极海”或“Geehy”字样或图形均为极海的商标，其他在极海产品上显示的产品或服务名称均为其各自所有者的财产。

2、无知识产权许可

极海拥有本手册所涉及的全部权利、所有权及知识产权。

极海不应因销售、分发极海产品及本手册而被视为将任何知识产权的许可或权利明示或默示地授予用户。

如果本手册中涉及任何第三方的产品、服务或知识产权，不应被视为极海授权用户使用前述第三方产品、服务或知识产权，除非在极海销售订单或销售合同中另有约定。

3、版本更新

用户在下单购买极海产品时可获取相应产品的最新版的手册。

如果本手册中所述的内容与极海产品不一致的，应以极海销售订单或销售合同中的约定为准。

4、信息可靠性

本手册相关数据经极海实验室或合作的第三方测试机构批量测试获得，但本手册相关数据难免会出现校正笔误或因测试环境差异所导致的误差，因此用户应当理解，极海对本手册中可能出现的该等错误无需承担任何责任。本手册相关数据仅用于指导用户作为性能参数参照，不构成极海对任何产品性能方面的保证。

用户应根据自身需求选择合适的极海产品，并对极海产品的应用适用性进行有效验证和测试，以确认极海产品满足用户自身的需求、相应标准、安全或其它可靠性要求；若因用户未充分对极海产品进行有效验证和测试而致使用户损失的，极海不承担任何责任。

5、合规要求

用户在使用本手册及所搭配的极海产品时，应遵守当地所适用的所有法律法规。用户应了解产品可能受到产品供应商、极海、极海经销商及用户所在地等各国有关出口、再出口或其它法律的限制，用户（代表其本身、子公司及关联企业）应同意并保证遵守所有关于取得极海产品及／或技术与直接产品的出口和再出口适用法律与法规。

6、免责声明

本手册由极海“按原样”（as is）提供，在适用法律所允许的范围内，极海不提供任何形式的明示或暗示担保，包括但不限于对产品适销性和特定用途适用性的担保。

对于用户后续在针对极海产品进行设计、使用的过程中所引起的任何纠纷，极海概不承担责任。

7、责任限制

在任何情况下，除非适用法律要求或书面同意，否则极海和/或以“按原样”形式提供本手册的任何第三方均不承担损害赔偿责任，包括任何一般、特殊因使用或无法使用本手册相关信息而产生的直接、间接或附带损害（包括但不限于数据丢失或数据不准确，或用户或第三方遭受的损失）。

8、适用范围

本手册的信息用以取代本手册所有早期版本所提供的信息。