第一章 PWM 全彩 LED 灯显示

1. 学习目的及目标

- 学习 LED 灯电路及硬件原理
- 学习 ESP32 的 PWM（ledc）功能的配置
- 掌握 PWM（ledc）控制全彩 LED 灯渐变程序

2. 全彩 LED 灯简介

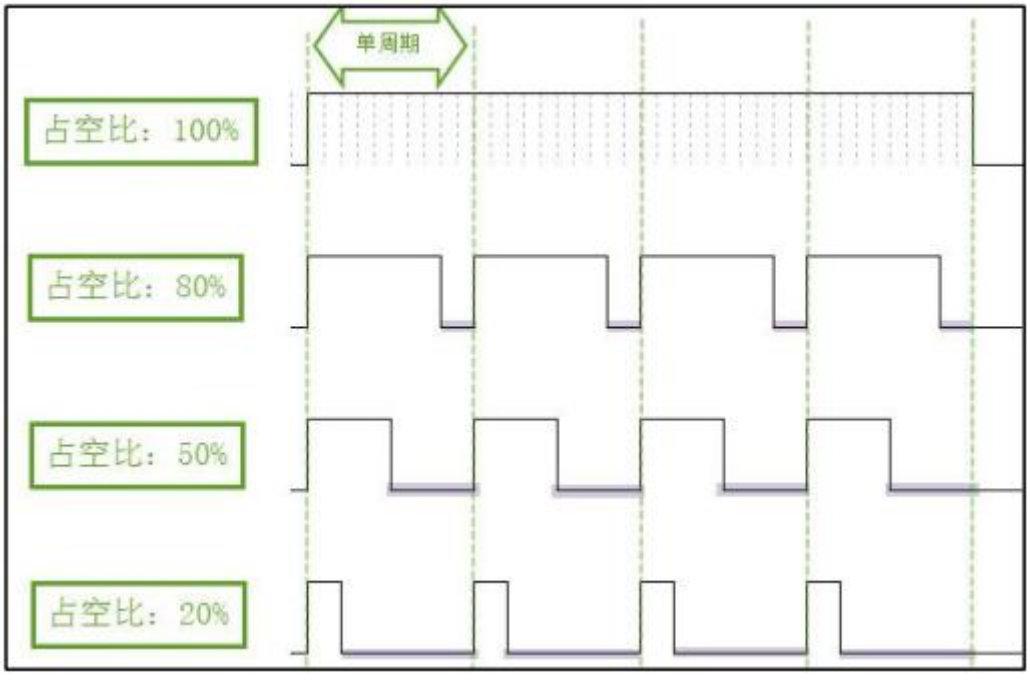
全彩 LED 灯，实质上是一种把红、绿、蓝单色发光体集成到小面积区域中的 LED 灯，控制时对这三种颜色的灯管输出不同的光照强度，即可混合得到不同的颜色，其混色原理与光的三原色混合原理一致。

例如，若红绿蓝灯都能控制输出光照强度为[0:255]种等级，那么该灯可混合得到使用 RGB888 表示的所有颜色(包括以 RGB 三个灯管都全灭所表示的纯黑色)。

3. 全彩 LED 灯控制原理

本教程配套开发板中的 RGB 灯就是一种全彩 LED 灯，前面介绍 LED 基本控制原理的时候，只能控制 RGB 三色灯的亮灭，即 RGB 每盏灯有[0:1]两种等级，因此只能组合出 8 种颜色。

要使用 ESP32 控制 LED 灯输出多种亮度等级，可以通过控制输出脉冲的占空比来实现。就是我们今天要学习的 PWM 功能，在 ESP32 中就是 ledc 功能。



上图列出了周期相同而占空比分别为 100%、80%、50 和 20%的脉冲波形，假如利用这样的脉冲控制 LED 灯，即可控制 LED 灯亮灭时间长度的比例。若提高脉冲的频率，LED 灯将会高频率进行开关切换，由于视觉暂留效应，人眼看不到 LED 灯的开关导致的闪烁现象，而是感觉到使用不同占空比的脉冲控制 LED 灯时的亮度差别。即单个控制周期内，LED 灯亮的平均时间越长，亮度就越高，反之越暗。

把脉冲信号占空比分成 256 个等级，即可用于控制 LED 灯输出 256 种亮度，使用三种

这样的信号控制 RGB 灯即可得到 256*256*256 种颜色混合的效果。而要控制占空比，直接使用 ESP32 的 LEDC 功能即可。

4. 硬件设计及原理

同 LED 章原理图。

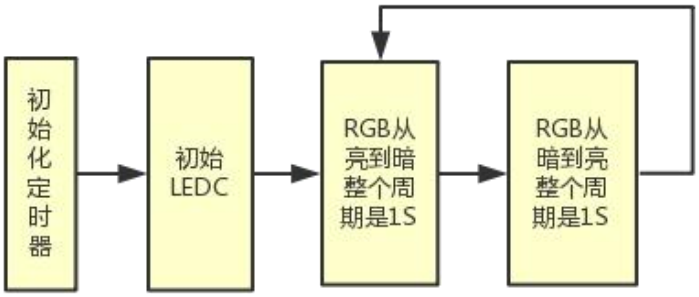
本实验板连接了一个 RGB 彩灯，RGB 彩灯实际上由三盏分别为红色、绿色、蓝色的 LED 灯组成，通过控制 RGB 颜色强度的组合，可以混合出各种色彩。

LED 标号	LED 颜色	接至 ESP32 的引脚
LP2A	红色	IO2
LP2B	绿色	IO18
LP2C	蓝色	IO19

若您使用的实验板 LED 灯的连接方式或引脚不一样，只需根据我们的工程修改引脚即可，程序的控制原理相同。

5. 软件设计

5.1. 代码逻辑



5.2. ESP32 的 PWM（ledc）接口介绍

➢ LEDC 配置函数：ledc_channel_config();

函数原型	esp_err_t ledc_channel_config (const ledc_channel_config_t* ledc_conf)
函数功能	LEDC 配置函数

参数	[in] ledc_conf:ledc 配置结构体 ledc_channel_config_t
	typedef struct {
	int gpio_num;//IO 编号
	ledc_mode_t speed_mode;//速度
	ledc_channel_t channel;//通道
	ledc_intr_type_t intr_type;//中断使能
	ledc_timer_t timer_sel; //定时器通道

	uint32_t duty; //占空比
	} ledc_channel_config_t;

返回值	ESP_OK：成功 ESP_ERR_INVALID_ARG ： 参数错误

➢ LEDC 渐变安装函数：ledc_fade_func_install();

函数原型	esp_err_t ledc_fade_func_install (int intr_alloc_flags)
函数功能	LEDC 渐变安装使能函数
参数	[in] intr_alloc_flags:分配中断标记
返回值	ESP_OK：成功 ESP_ERR_INVALID_ARG ： 参数错误

➢ LEDC 渐变函数：ledc_set_fade_with_time();

函数原型	esp_err_t ledc_set_fade_with_time (ledc_mode_t speed_mode, ledc_channel_t channel, uint32_t target_duty, int max_fade_time_ms)
函数功能	LEDC 渐变函数
参数	[in] speed_mode:速度 [in] channel:通道 [in] target_duty:目标占空比 [in] max_fade_time_ms:到达目标占空比需要的时间
返回值	ESP_OK：成功 ESP_ERR_INVALID_ARG ： 参数错误 ESP_ERR_INVALID_STATE：LEDC 渐变未安装 ESP_FAIL：LEDC 渐变安装失败

➢ LEDC 渐变开始函数：ledc_fade_start();

函数原型	esp_err_t ledc_fade_start (ledc_mode_t speed_mode, ledc_channel_t channel, ledc_fade_mode_t wait_done)
函数功能	LEDC 渐变开始函数
参数	[in] speed_mode:速度 [in] channel:通道 [in] wait_done:是否等待
返回值	ESP_OK：成功 ESP_ERR_INVALID_ARG ： 参数错误 ESP_ERR_INVALID_STATE：LEDC 渐变未安装

更多更详细接口请参考官方指南。

5.3. 渐变 RGB 彩灯源码编写

获取按键状态，等于 0 表示按键按下，然后死等松手，切换灯状态。

- 包含头文件、IO 宏定义、ledc 宏定义

```
#include <stdio.h>
#include "esp_system.h"
#include "esp_spi_flash.h"
#include "esp_wifi.h"
#include "esp_event_loop.h"
#include "esp_log.h"
#include "esp_err.h"
#include "nvs_flash.h"
#include "freertos/FreeRTOS.h"
#include "freertos/task.h"
#include "driver/ledc.h"
#include <stdio.h>
#include "freertos/FreeRTOS.h"
#include "freertos/task.h"
#include "driver/ledc.h"
#define LED_R_IO          2
#define LED_G_IO          18
#define LED_B_IO          19
#define LEDC_MAX_DUTY               (8191)    //2 的 13 次方-1(13 位 PWM)
#define LEDC_FADE_TIME              (1000)    //渐变时间(ms)
```

- RGB 灯 PWM（ledc）配置，并使能渐变函数

```
/*
* void ledc_init(void):定时器 0 用在 PWM 模式，输出 3 通道的 LEDC 信号
*/
void ledc_init(void)
{
    //定时器配置结构体
    ledc_timer_config_t  ledc_timer;
    //定时器配置->timer0
    ledc_timer.duty_resolution = LEDC_TIMER_13_BIT; //PWM 分辨率
    ledc_timer.freq_hz = 5000;                      //频率
    ledc_timer.speed_mode = LEDC_HIGH_SPEED_MODE;       //速度
    ledc_timer.timer_num = LEDC_TIMER_0;                // 选择定时器
    ledc_timer_config(&ledc_timer);                     //设置定时器 PWM 模式
    //PWM 通道 0 配置->IO2->红色灯
    g_ledc_ch_R.channel    = LEDC_CHANNEL_0;       //PWM 通道
    g_ledc_ch_R.duty       = 0;                         //占空比
    g_ledc_ch_R.gpio_num   = LED_R_IO;                  //IO 映射
```

```
    g_ledc_ch_R.speed_mode = LEDC_HIGH_SPEED_MODE;     //速度
    g_ledc_ch_R.timer_sel  = LEDC_TIMER_0;             //选择定时器
    ledc_channel_config(&g_ledc_ch_R);                 //配置 PWM
    //PWM 通道 1 配置->IO18->绿色灯
    g_ledc_ch_G.channel    = LEDC_CHANNEL_1;       //PWM 通道
    g_ledc_ch_G.duty       = 0;                        //占空比
    g_ledc_ch_G.gpio_num   = LED_G_IO;                 //IO 映射
    g_ledc_ch_G.speed_mode = LEDC_HIGH_SPEED_MODE;     //速度
    g_ledc_ch_G.timer_sel  = LEDC_TIMER_0;             //选择定时器
    ledc_channel_config(&g_ledc_ch_G);                 //配置 PWM
    //PWM 通道 2 配置->IO19->蓝色灯
    g_ledc_ch_B.channel    = LEDC_CHANNEL_2;       //PWM 通道
    g_ledc_ch_B.duty       = 0;                        //占空比
    g_ledc_ch_B.gpio_num   =LED_B_IO ;                 //IO 映射
    g_ledc_ch_B.speed_mode = LEDC_HIGH_SPEED_MODE;     //速度
    g_ledc_ch_B.timer_sel  = LEDC_TIMER_0;             //选择定时器
    ledc_channel_config(&g_ledc_ch_B);                 //配置 PWM
    //使能 ledc 渐变功能
    ledc_fade_func_install(0);
}
```

- 主函数：渐变 RGB 灯

  RGB 灯由 0%到 100%渐变（暗到亮），历时 1 秒

  RGB 灯由 100%到 0%渐变（亮到暗），历时 1 秒

```
/*
 * 应用程序的函数入口
*/
void app_main()
{
    ledc_init();//ledc 初始化
    while(1)
     {
          //ledc 红灯渐变至 100%，时间 LEDC_FADE_TIME
          ledc_set_fade_with_time(g_ledc_ch_R.speed_mode,
                                  g_ledc_ch_R.channel,
                                  LEDC_MAX_DUTY,
                                  LEDC_FADE_TIME);
          //渐变开始
          ledc_fade_start(g_ledc_ch_R.speed_mode,
                          g_ledc_ch_R.channel,
                          LEDC_FADE_NO_WAIT);
          //略过
          vTaskDelay(LEDC_FADE_TIME / portTICK_PERIOD_MS);

```

```
            //ledc 红灯 渐变至 0%，时间 LEDC_FADE_TIME
            ledc_set_fade_with_time(g_ledc_ch_R.speed_mode,
                                    g_ledc_ch_R.channel,
                                    0,
                                    LEDC_FADE_TIME);
            //渐变开始
            ledc_fade_start(g_ledc_ch_R.speed_mode,
                            g_ledc_ch_R.channel,
                            LEDC_FADE_NO_WAIT);
            //略过
            vTaskDelay(LEDC_FADE_TIME / portTICK_PERIOD_MS);
        }
}
```

5.4. 硬件连接

红旭开发板默认已经连接好 RGB 灯，下载程序即可，使用其他开发板需要修改程序或者修改硬件连接皆可。

5.5. 效果展示

RGB 灯由暗到亮，再由亮变暗，如此循环。

6. PWM（ledc）总结

- 渐变在指示灯处比较常用。
- 不用渐变的时候可以使用，设置占空比函数直接怼。
  - 设置占空比 ledc_set_duty();
  - 还有几个常用的 ledc 库函数，请参考 ledc.h 文件，太简单不做介绍。
- 源码地址：https://github.com/xiaolongba/wireless-tech