8. RGB color light strip

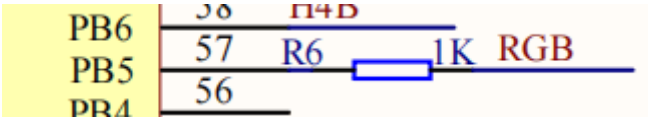
8.1. Experimental purpose

The SPI communication of STM32 is used to simulate the communication protocol of WS2812B module and drive the RGB light strip to display the effect.

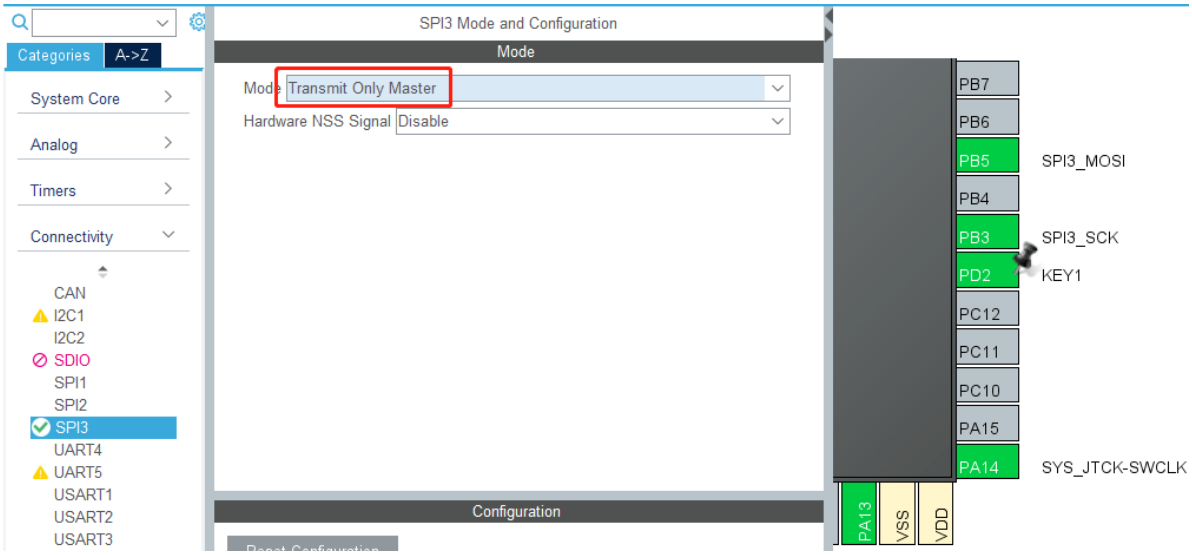
8.2. Configure pin information

1. Import ioc file from Beep project and name it RGB_Strip.

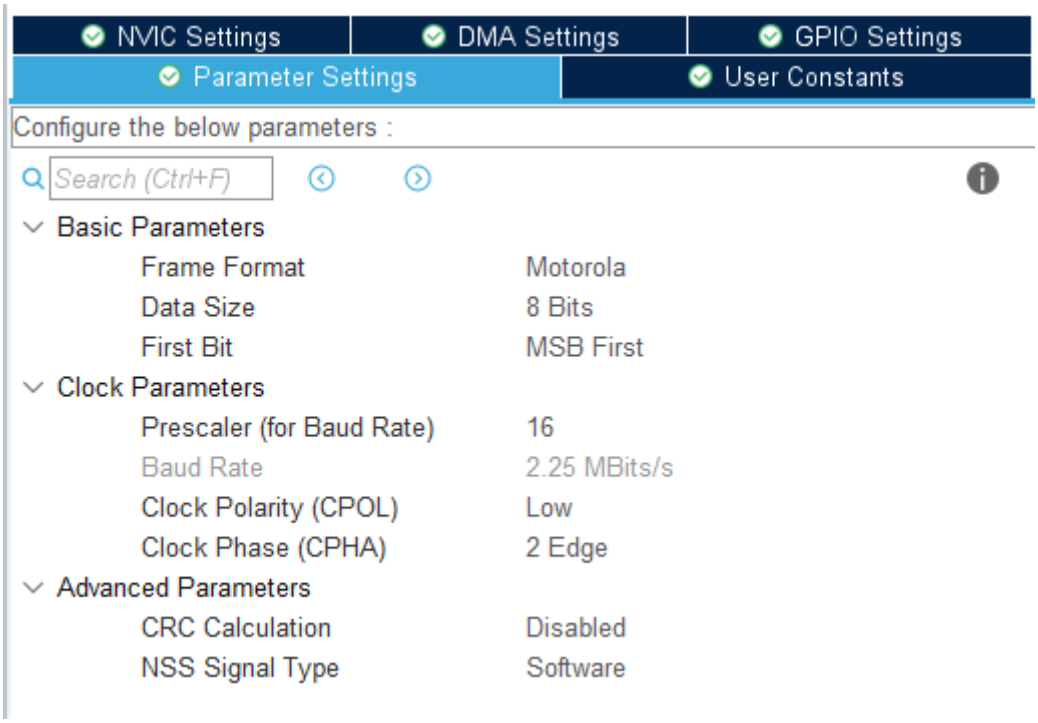
According to the schematic diagram, the pin of the RGB light bar connection is PB5. The driving mode of the RGB light bar can use timer PWM output or SPI output, and the PB5 pin can be redefined as timer PWM or SPI output. Considering the subsequent timer conflict problem, SPI communication mode is used here to drive the RGB light bar.



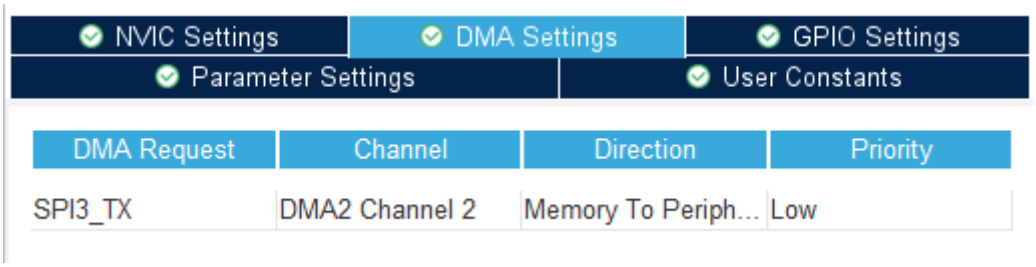
2. Set the mode of SPI3 to Transmit Only Master, so that pin PB5 is automatically set to SPI3_MOSI.



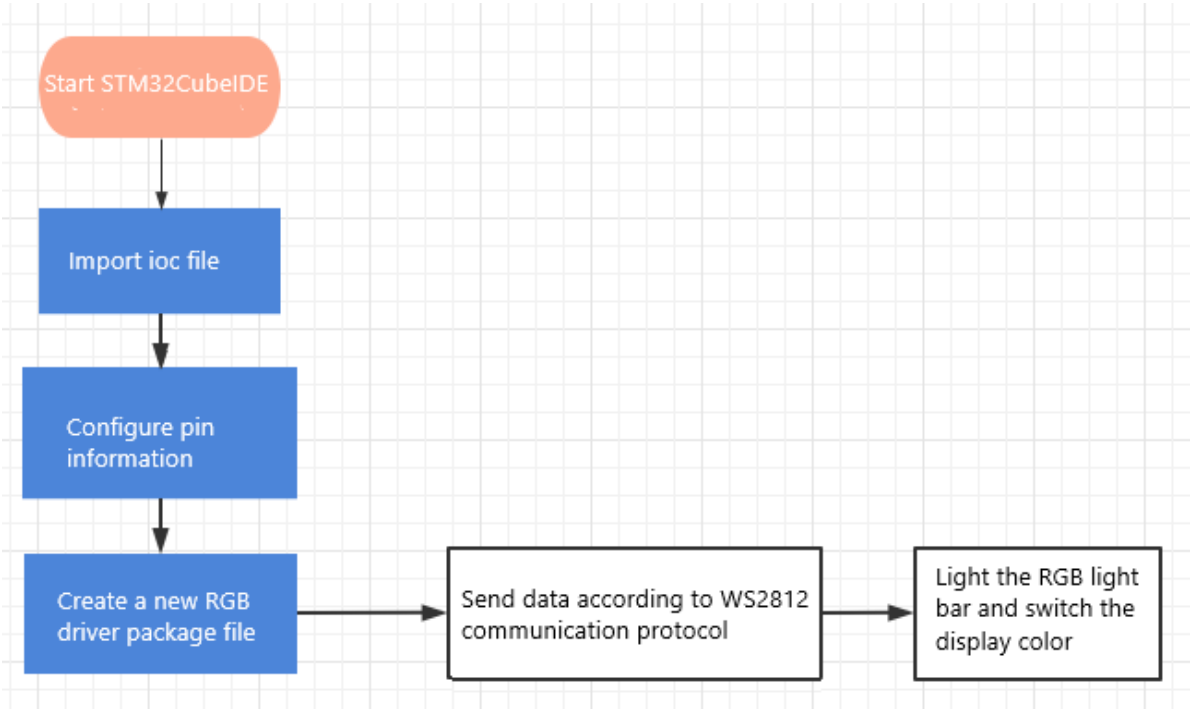
3. Modify the parameters of data sent by SPI3. Refer to the following picture for specific parameters.



4. Add DMA Settings for SPI3_TX.



8.3. Analysis of experimental flow chart



8.4. Core code interpretation

1. Create bsp_rgb.h and bsp_rgb.c, and add the following content to bsp_rgb.h:

```
#define RGB_CTRL_ALL    0xFF
#define MAX_RGB         14

typedef struct
{
    union
    {
        uint8_t Buff[9];
        struct
        {
            uint8_t G[3]; // G First
            uint8_t R[3]; // R Second
            uint8_t B[3]; // B Third
        } RGB;
    } Strip[MAX_RGB];
} ws2812_t;


void RGB_Init(void);
void RGB_Update(void);

void RGB_Set_Color(uint8_t index, uint8_t r, uint8_t g, uint8_t b);
void RGB_Set_Color_U32(uint8_t index, uint32_t color);
void RGB_Clear(void);
```

Among them, the ws2812_t structure is used to store the cache data of the light bar.

2. Create the following content in the bsp_rgb.c file: According to the communication protocol of WS2812, the three bits of SPI are simulated to the one bit of WS2812. When the value of SPI output three digits is 0b110, it corresponds to the high level of WS2812; when the value of SPI output three digits is 0b100, it corresponds to the low level of WS2812.

```
    __
// |  |_|   0bl10  high level
//  _
// | |__|   0bl00  low level
#define TIMING_ONE          0x06
#define TIMING_ZERO         0x04
```

3. WS2812_Set_Color_One() function sets a single RGB light color value, index=[0, MAX_RGB-1], RGB=[0x00000000, 0x00FFFFFF], RGB value permutation: red in the front, middle is green, the last is blue.

```
// 设置单个RGB灯颜色值，index=[0, MAX_RGB-1]，RGB=[0x00000000, 0x00FFFFFF]
// Set single RGB light color value, index=[0, MAX_RGB-1], RGB=[0x00000000, 0x00FFFFFF]
static void WS2812_Set_Color_One(uint8_t index, uint32_t RGB)
{
    if (index >= MAX_RGB) return;
    uint8_t i;
    uint32_t TempR = 0, TempG = 0, TempB = 0;

    for(i = 0; i < 8; i++)
    {
        (RGB & 0x00010000) == 0 ? (TempR |= (TIMING_ZERO<<(i*3))) : (TempR |= (TIMING_ONE<<(i*3)));
        (RGB & 0x00000100) == 0 ? (TempG |= (TIMING_ZERO<<(i*3))) : (TempG |= (TIMING_ONE<<(i*3)));
        (RGB & 0x00000001) == 0 ? (TempB |= (TIMING_ZERO<<(i*3))) : (TempB |= (TIMING_ONE<<(i*3)));
        RGB >>= 1;
    }
    for (i = 0; i < 3; i++)
    {
        g_ws2812.Strip[index].RGB.R[i] = TempR >> (16-8*i);
        g_ws2812.Strip[index].RGB.G[i] = TempG >> (16-8*i);
        g_ws2812.Strip[index].RGB.B[i] = TempB >> (16-8*i);
    }
}
```

4. RGB_Set_Color(index, r, g, b) and RGB_Set_Color_U32(index, color) Set the RGB strip color value. index=[0, MAX_RGB-1] controls the color of the corresponding lamp bead, and index=255 controls the color of all lamp bead.

```
// 设置颜色，index=[0, MAX_RGB-1]控制对应灯珠颜色，index=0xFF控制所有灯珠颜色。
// Set the color, index=[0, max_RGB-1] controls the corresponding bead color,
void RGB_Set_Color(uint8_t index, uint8_t r, uint8_t g, uint8_t b)
{
    uint32_t color = r << 16 | g << 8 | b;
    RGB_Set_Color_U32(index, color);
}

// 设置RGB灯条颜色值，index=[0, MAX_RGB-1]控制对应灯珠颜色，index=255控制所有灯珠颜色。
// Set the RGB bar color value, index=[0, max_RGB-1] controls the corresponding bead color
void RGB_Set_Color_U32(uint8_t index, uint32_t color)
{
    if (index < MAX_RGB)
    {
        WS2812_Set_Color_One(index, color);
        return;
    }
    if (index == RGB_CTRL_ALL)
    {
        for (uint16_t i = 0; i < MAX_RGB; i++)
        {
            WS2812_Set_Color_One(i, color);
        }
    }
}
```

5.RGB_Clear() Clears the RGB indicator cache.

```
// Clear color (off)  清除颜色（熄灭）
void RGB_Clear(void)
{
    for (uint8_t i = 0; i < MAX_RGB; i++)
    {
        WS2812_Set_Color_One(i, 0);
    }
}
```

6. The RGB_Update() function refreshed the RGB light bar color. The RGB light must call the RGB_Update() function to refresh the display after changing the color, otherwise it will not produce the effect.

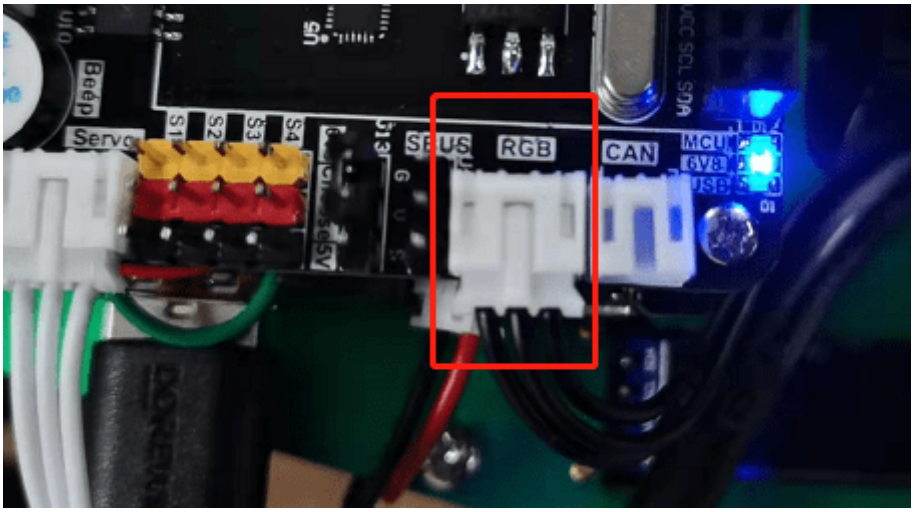
```
// 刷新RGB灯条颜色。下方函数调用修改RGB颜色后，必须调用此函数更新显示。
// Refresh RGB light bar color. This function must be called to update
void RGB_Update(void)
{
    WS2812_Send_Data((uint8_t*)&g_ws2812.Strip[0].Buff, 9*MAX_RGB);
}
```

7. Send data using DMA of SPI3.

```
// transmitter data  发送数据
static void WS2812_Send_Data(uint8_t *buf, uint16_t buf_size)
{
    HAL_SPI_Transmit_DMA(&hspi3, buf, buf_size);
}
```

8.5. Hardware connection

Since the RGB indicator bar needs to be connected to the position of the RGB indicator bar on the expansion board, the interface has been set for anti-reverse connection. You can insert the interface in the correct direction.



8.6. Experimental effect

After burning the program, the LED light flashes every 200 milliseconds, and the RGB light strip displays red, green, and blue in turn, switching a color every 500 milliseconds.