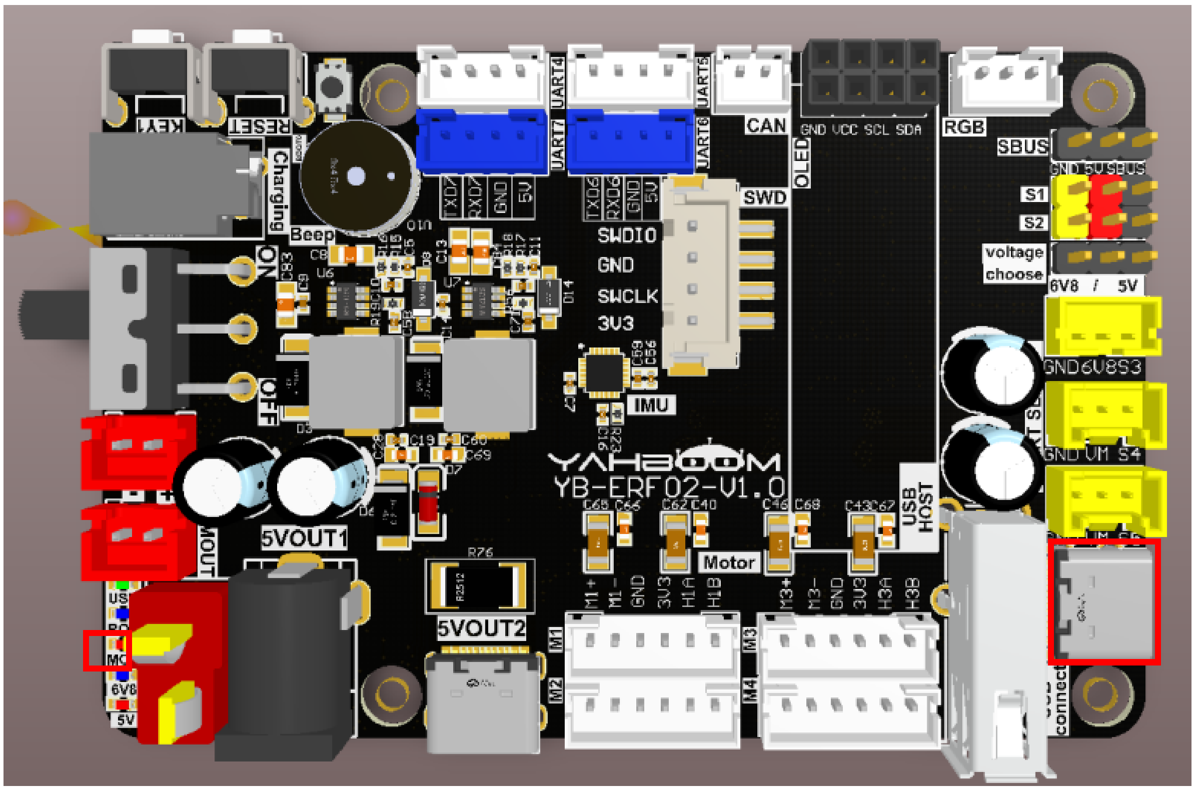
Light up the LED light

1. Experimental Purpose

Control the LED indicator on the STM32 control board to flash.

2. Hardware Connection

As shown in the figure below, the LED indicator is an onboard component, so no external devices are required. Please connect the Type-C data cable between the computer and the USB Connect port on the STM32 control board.

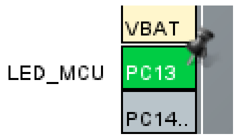


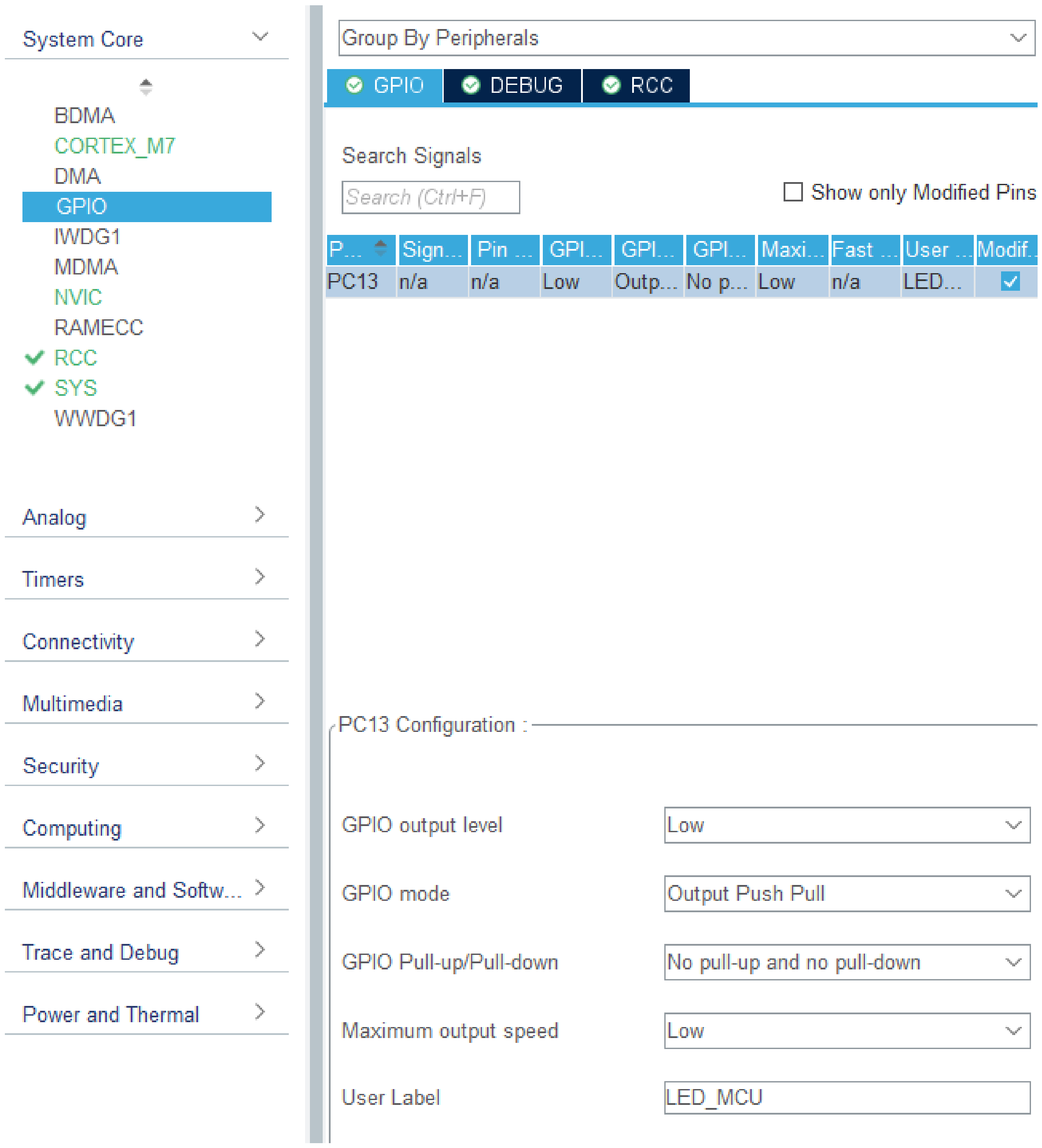
3. Core code analysis

Open STM32CUBEIDE and import the project. The path corresponding to the program source code is:

```
Board_Samples/STM32_Samples/Led
```

Initialize the LED peripheral, where LED_GPIO corresponds to PC13 of the hardware circuit and the GPIO mode is output mode.





P...	Sign...	Pin ...	GPI...	GPI...	GPI...	Maxi...	Fast ...	User ...	Modif...
PC13	n/a	n/a	Low	Outp...	No p...	Low	n/a	LED...	✓

PC13 Configuration :

GPIO output level	Low
GPIO mode	Output Push Pull
GPIO Pull-up/Pull-down	No pull-up and no pull-down
Maximum output speed	Low
User Label	LED_MCU

```
#define LED_MCU_Pin GPIO_PIN_13
#define LED_MCU_GPIO_Port GPIOC

void MX_GPIO_Init(void)
{
  GPIO_InitTypeDef GPIO_InitStruct = {0};

  /* GPIO Ports Clock Enable */
  __HAL_RCC_GPIOC_CLK_ENABLE();
  __HAL_RCC_GPIOH_CLK_ENABLE();
  __HAL_RCC_GPIOA_CLK_ENABLE();

  /*Configure GPIO pin Output Level */
  HAL_GPIO_WritePin(LED_MCU_GPIO_Port, LED_MCU_Pin, GPIO_PIN_RESET);

  /*Configure GPIO pin : PtPin */
  GPIO_InitStruct.Pin = LED_MCU_Pin;
  GPIO_InitStruct.Mode = GPIO_MODE_OUTPUT_PP;
  GPIO_InitStruct.Pull = GPIO_NOPULL;
  GPIO_InitStruct.Speed = GPIO_SPEED_FREQ_LOW;
  HAL_GPIO_Init(LED_MCU_GPIO_Port, &GPIO_InitStruct);
}
```

Turn on the LED light

```
#define LED_MCU_ON() HAL_GPIO_WritePin(LED_MCU_GPIO_Port, LED_MCU_Pin, SET)
```

Turn off LED lights

```
#define LED_MCU_OFF() HAL_GPIO_WritePin(LED_MCU_GPIO_Port, LED_MCU_Pin, RESET)
```

Control the LED light status flip

```
#define LED_MCU_TOGGLE() HAL_GPIO_TogglePin(LED_MCU_GPIO_Port, LED_MCU_Pin)
```

The LED blinking function flips the LED state every time it is called 20 times.

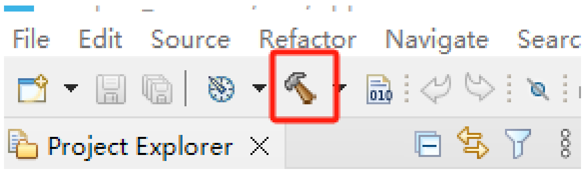
```
void App_Led_Mcu_Handle(void)
{
    static uint8_t led_count = 0;
    led_count++;
    if (led_count >= 20)
    {
        led_count = 0;
        LED_MCU_TOGGLE();
    }
}
```

Call the App_Led_Mcu_Handle function every 10 milliseconds to make the LED blink.

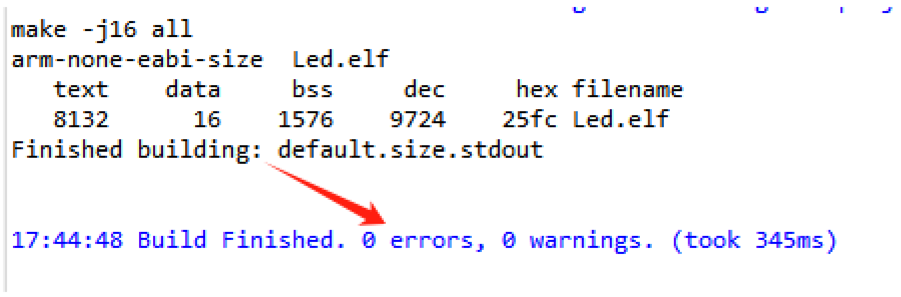
```
while (1)
{
    App_Led_Mcu_Handle();
    HAL_Delay(10);
}
```

4. Compile, download and burn firmware

Select the project to be compiled in the file management interface of STM32CUBEIDE and click the compile button on the toolbar to start compiling.



If there are no errors or warnings, the compilation is complete.



```
make -j16 all
arm-none-eabi-size  Led.elf
   text    data     bss     dec     hex filename
   8132      16    1576    9724    25fc Led.elf
Finished building: default.size.stdout


17:44:48 Build Finished. 0 errors, 0 warnings. (took 345ms)
```

Press and hold the BOOT0 button, then press the RESET button to reset, release the BOOT0 button to enter the serial port burning mode. Then use the serial port burning tool to burn the firmware to the board.

If you have STlink or JLink, you can also use STM32CUBEIDE to burn the firmware with one click, which is more convenient and quick.

5. Experimental Results

The MCU_LED light flashes every 200 milliseconds.