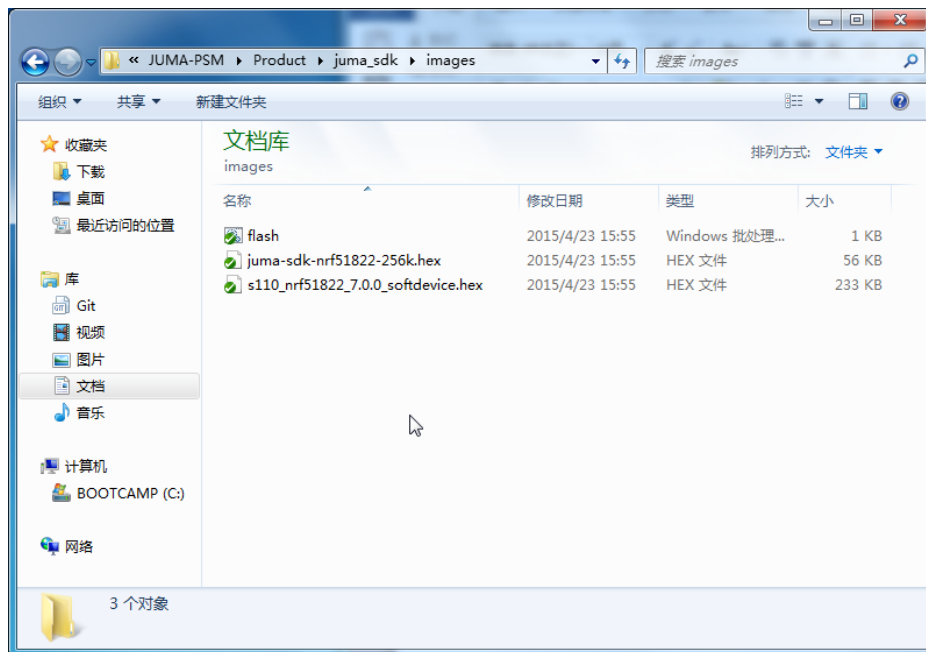


使用 JUMA SDK 开始一个 Blink

烧写底层固件

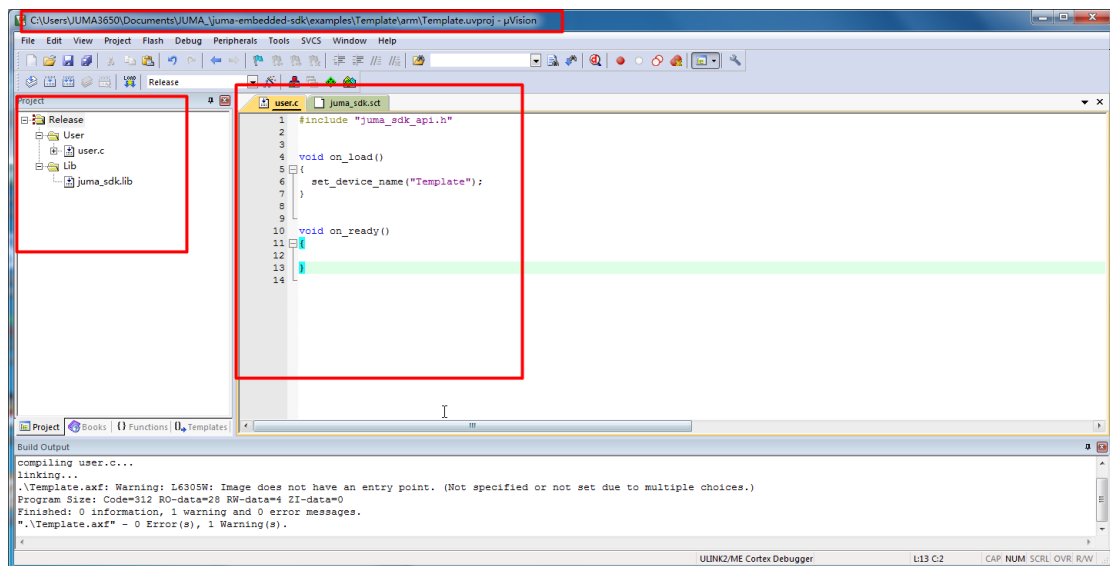
在使用之前，需要为芯片烧写底层的固件。

首先确保芯片和电脑已经通过正确的线路连接，
然后进入到 JUMA SDK 文件夹 `juma_sdk/images` 下，点击 `flash` 即可。



找到 JUMA SDK 给出的工程模板（examples Template）

打开的工程如图所示



最上边的内容为 工程模板的位置。

左侧为要包含的 JUMA 的库文件和用户文件

右侧为用户的代码区

在用户代码中

On_load()函数必须存在并且不能为空。

并且在此处需要设置设备的名字，否则的话扫描到的设备就叫做“Template”。

编写 Blink 的代码

定义引脚

```
#define LED_1 18
```

编写开启 LED 的函数

```
void led_on_task(void* args)
{
    // Pull Up
    set_gpio_pin_state(LED_1, 1);
    run_after_delay(led_off_task, NULL, 250);
}
```

编写关闭 LED 的函数

```
void led_off_task(void* args)
{
    // Pull Down
```

```

        set_gpio_pin_state(LED_1, 0);
        run_after_delay(led_on_task, NULL, 750);
    }

```

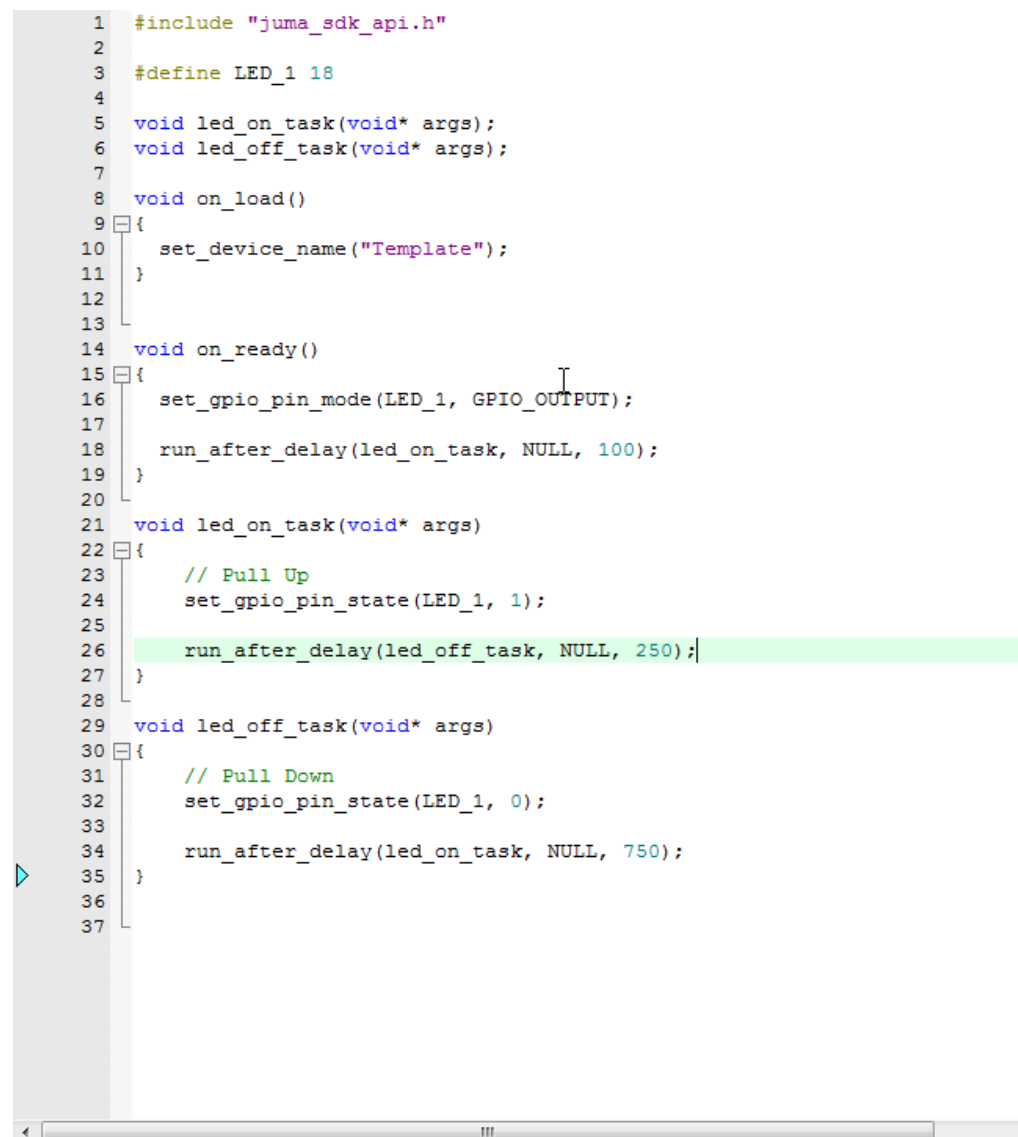
然后在 on_ready()函数中添加

```

void on_ready()
{
    set_gpio_pin_mode(LED_1, GPIO_OUTPUT);
    run_after_delay(led_on_task, NULL, 100);
}

```

编写完毕的效果如图

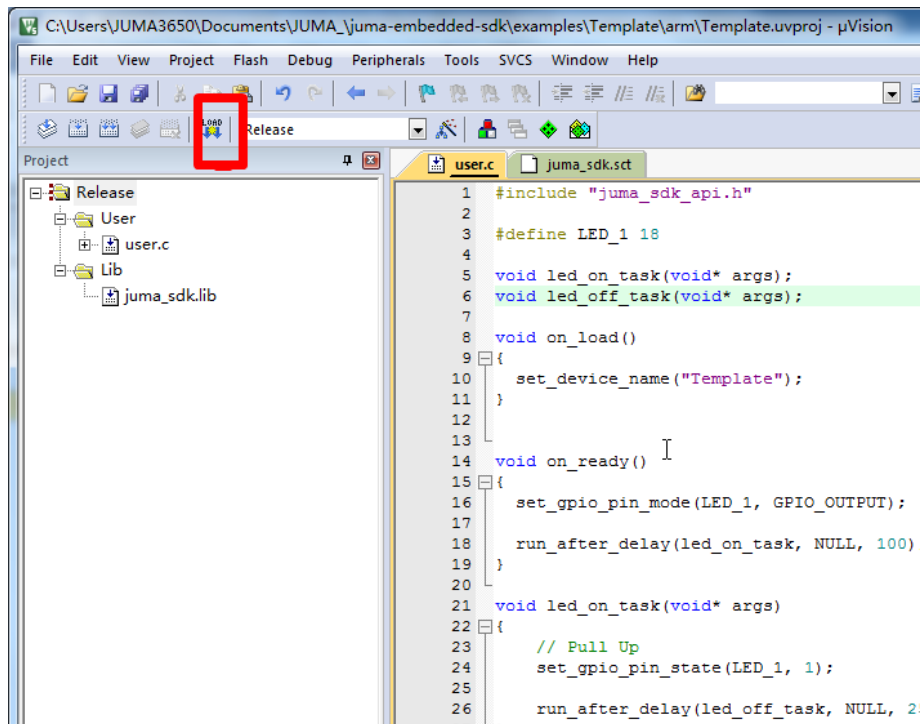


```

1  #include "juma_sdk_api.h"
2
3  #define LED_1 18
4
5  void led_on_task(void* args);
6  void led_off_task(void* args);
7
8  void on_load()
9  {
10     set_device_name("Template");
11 }
12
13
14 void on_ready()
15 {
16     set_gpio_pin_mode(LED_1, GPIO_OUTPUT);
17
18     run_after_delay(led_on_task, NULL, 100);
19 }
20
21 void led_on_task(void* args)
22 {
23     // Pull Up
24     set_gpio_pin_state(LED_1, 1);
25
26     run_after_delay(led_off_task, NULL, 250);
27 }
28
29 void led_off_task(void* args)
30 {
31     // Pull Down
32     set_gpio_pin_state(LED_1, 0);
33
34     run_after_delay(led_on_task, NULL, 750);
35 }
36
37

```

之后点击下载



则会看到开发板上的 LED 在闪烁。