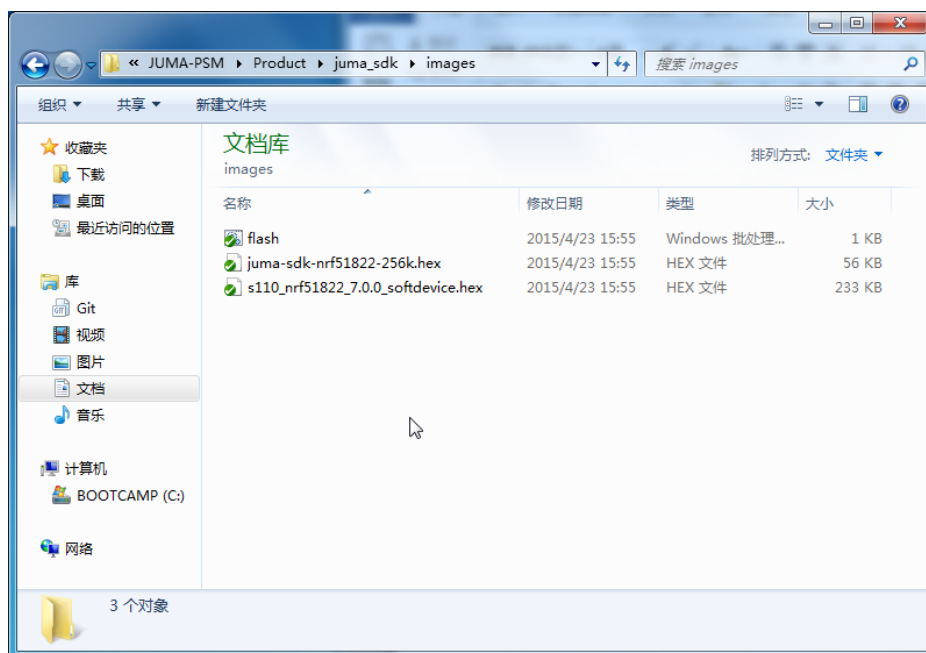


# 使用 JUMA SDK 开始一个 Blink

## 1. 烧写底层固件

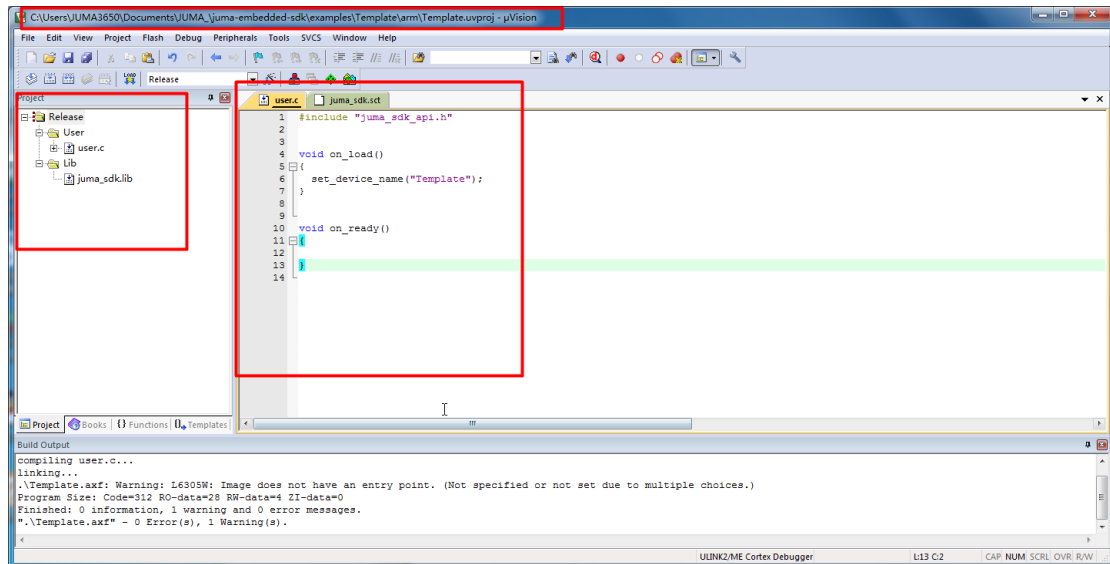
在使用之前，需要为芯片烧写底层的固件。

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。



## 2. 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”。

## 3. 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
    gpio_write(LED_1, 1);
    run_after_delay(led_off_task, NULL, 250);
}
```

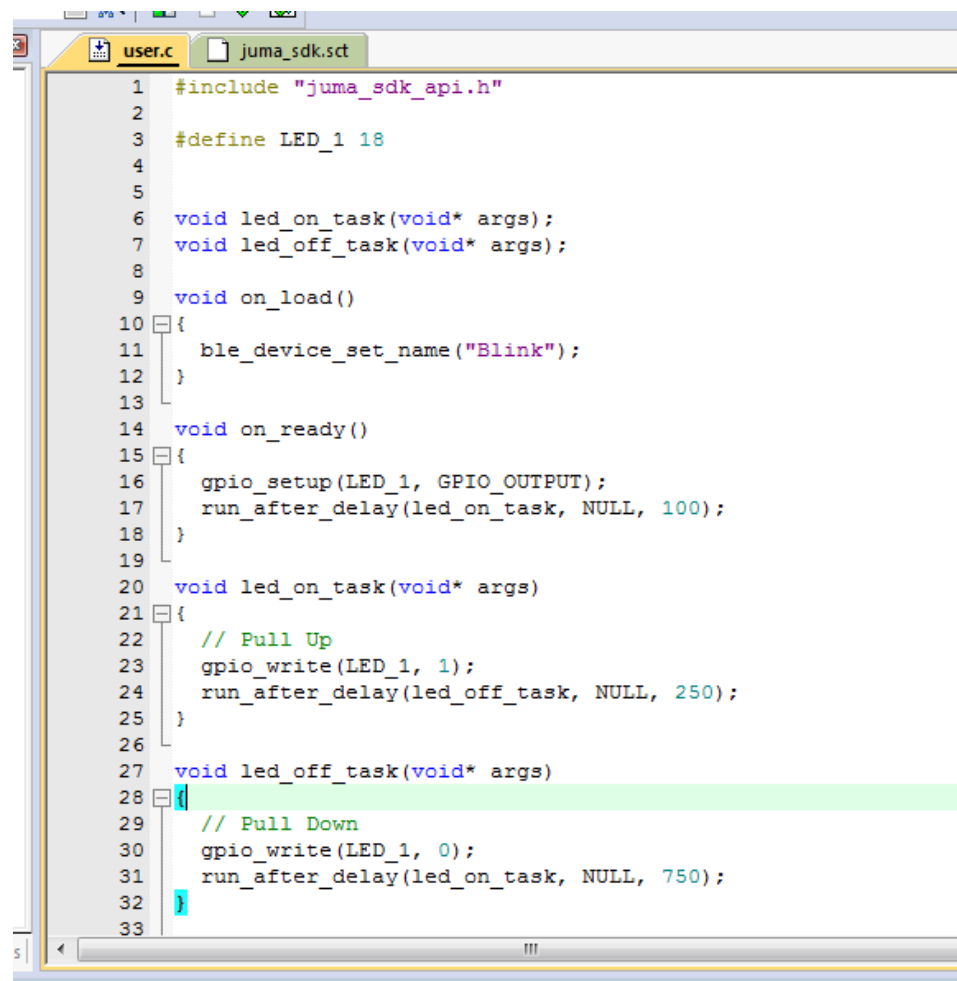
编写关闭 LED 的函数

```
void led_off_task(void* args)
{
    // Pull Down
    gpio_write(LED_1, 0);
    run_after_delay(led_on_task, NULL, 750);
}
```

然后在 on\_ready()函数中添加

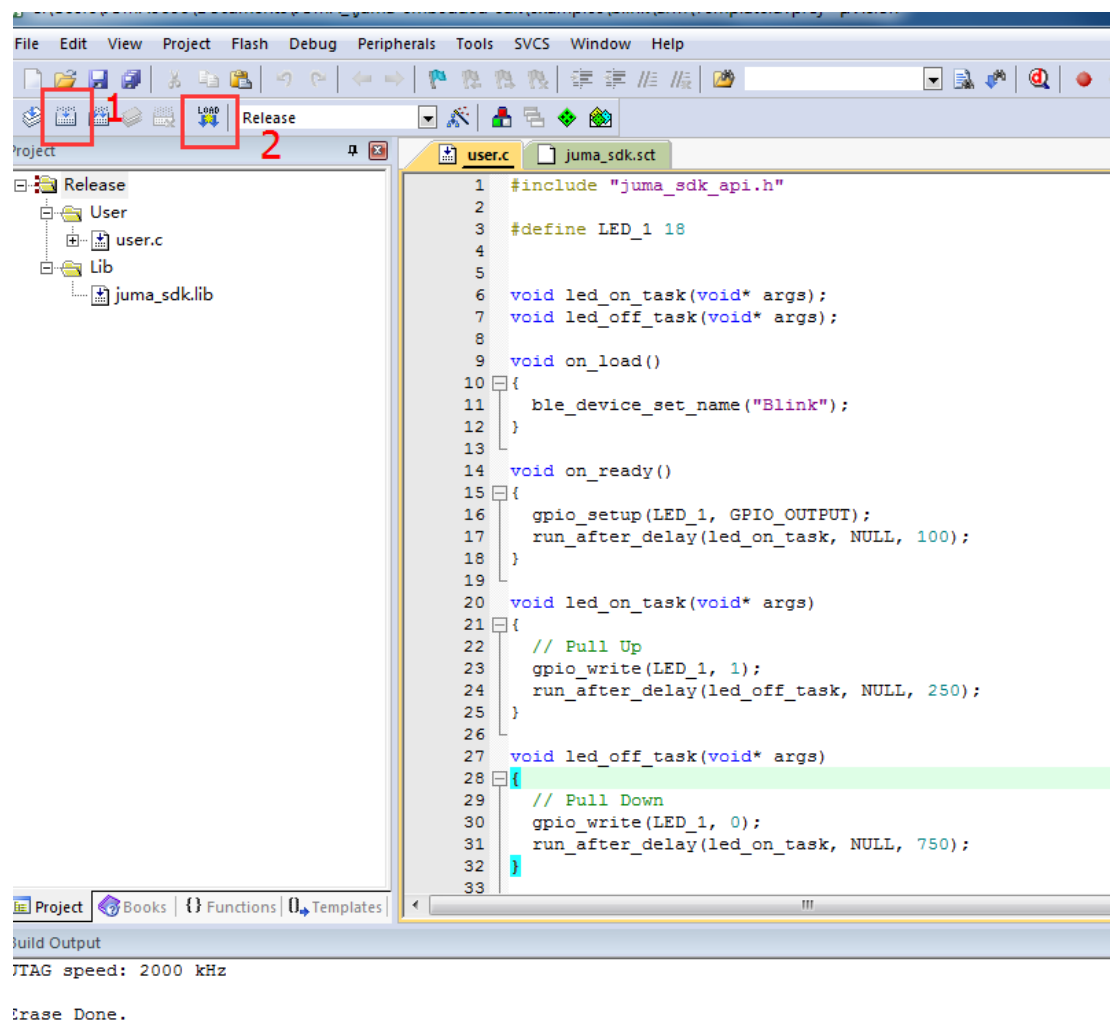
```
void on_ready()
{
    gpio_setup(LED_1, GPIO_OUTPUT);
    run_after_delay(led_on_task, NULL, 100);
}
```

编写完毕的效果如图



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

## 4. 下载程序



则会看到开发板上的 LED 在闪烁。