联睿微SDK\_V3.0使用手册

**文档修改记录**

|  |  |  |  |
| --- | --- | --- | --- |
| **版本** | **日期** | **修改内容** | **作者** |
|  |  |  |  |
|  |  |  |  |

# 第一章 SDK简介

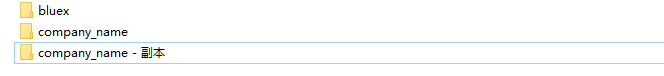
## 1、目录结构



## application

此文件夹下，起码有两个文件夹，bluex和company\_name，其中bluex为本公司定义的一些应用，company\_name为通用模板，用户如需定义自己的项目，可以直接复制文件夹，然后更改相关文件名即可。如一个名为bluex001的企业，新建一个蓝牙锁的项目，可以按以下步骤快速新建项目：

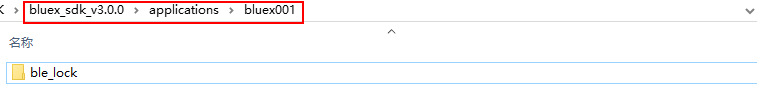
* 复制company\_name文件夹

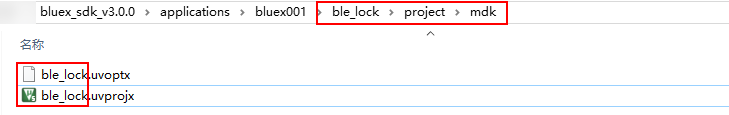


* 更改企业名称为bluex001



* 更改项目名称为ble\_lock





* 最后打开工程项目即可。

# 第二章 搭建开发环境

## 1、硬件工具

|  |  |
| --- | --- |
| 名称 | 描述 |
| BX2400\_EVK开发板 | 开发板 |
| Jlink仿真器 | 仿真和下载程序 |
| 杜邦线、跳线帽 | 连接开发板和仿真器 |

## 2、软件工具

|  |  |
| --- | --- |
| 名称 | 描述 |
| win7/win10 | 计算机系统 |
| mdk集成开发环境 | BlueX芯片开发环境 |
| J-Flash软件 | 读写芯片，查看调试信息，非必须 |
| nRF Connect | 手机端软件，用来扫描、连接、读写设备等 |

## 3、安装工具

### 3.1安装MDK

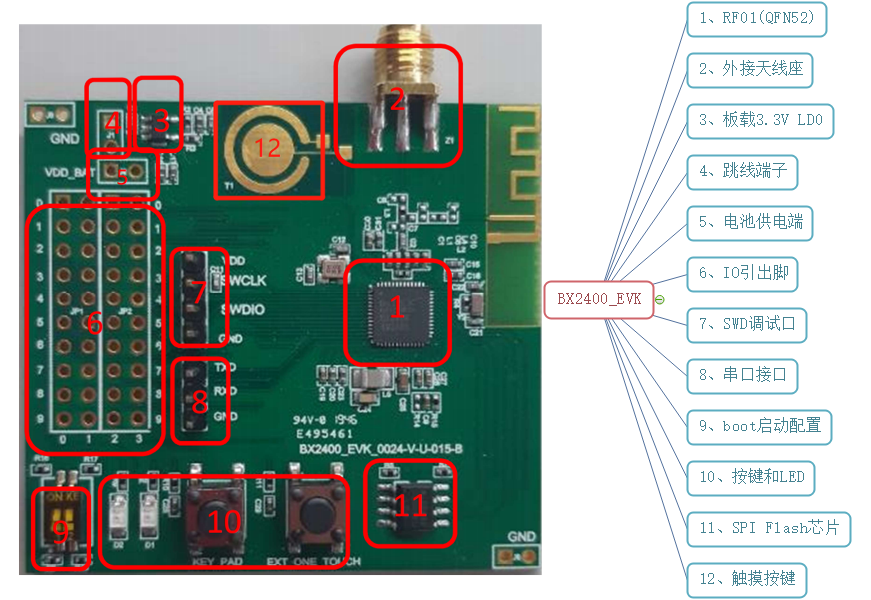
### 3.2安装JFlash

### 3.3安装手机端APP

# 第三章 开发板介绍

## 1、BX2400\_EVK

### 1.1功能特点



注意：在第9个boot启动配置中，左边的开关（P16）用于选择SPI Flash启动还是UART启动，下拨（接 GND）则正常从SPI Flash启动，上拨一般用于强制烧录芯片。右边的开关（P23）用于适配SPI Flash芯片的供电电压，EVK开发板保持下拔接地即可。

### 1.2硬件原理

#### 1.2.1按钮

#### 1.2.2触摸按钮

#### 1.2.3 LED

#### 1.2.4拨码开关

#### 1.2.5天线

#### 1.2.6 Jlink接口

#### 1.2.7串口接口

#### 1.2.8电源接口

#### 1.2.9 IO拓展

# 第四章 新建工程

## 1、新建模板

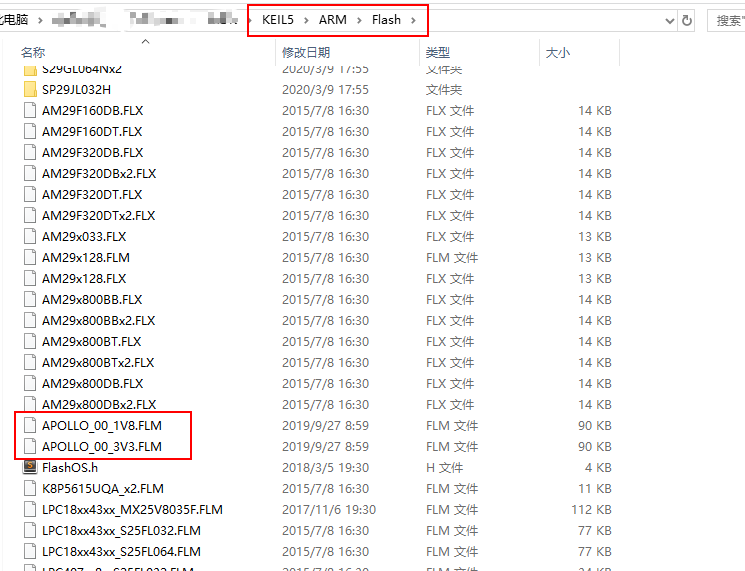
### 1.1 事前准备

#### 1.1.1 把文件移动到相关路径

打开SDK工具箱，找到prog\_tool\_v2：



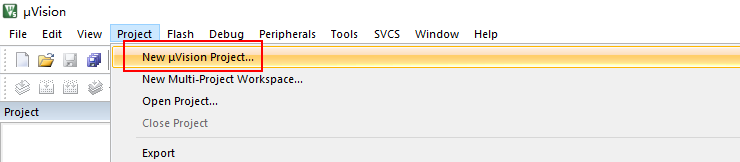
将BlueX文件夹中的 APOLLO\_00\_1V8.FLM 和 APOLLO\_00\_3V3.FLM 拷贝到 Keil\_v5安装目录下的ARM/Flash 目录中，如下图：



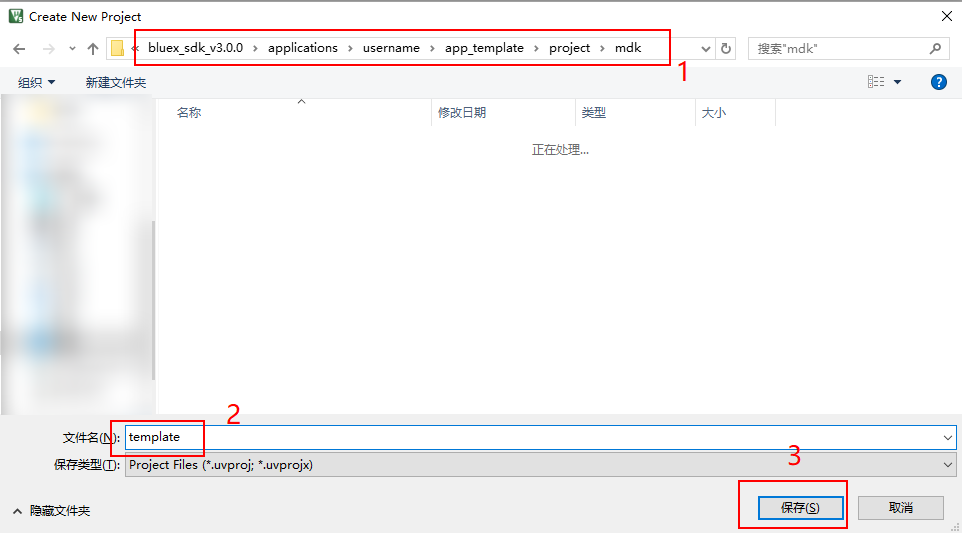
#### 1.1.2 文件目录规划

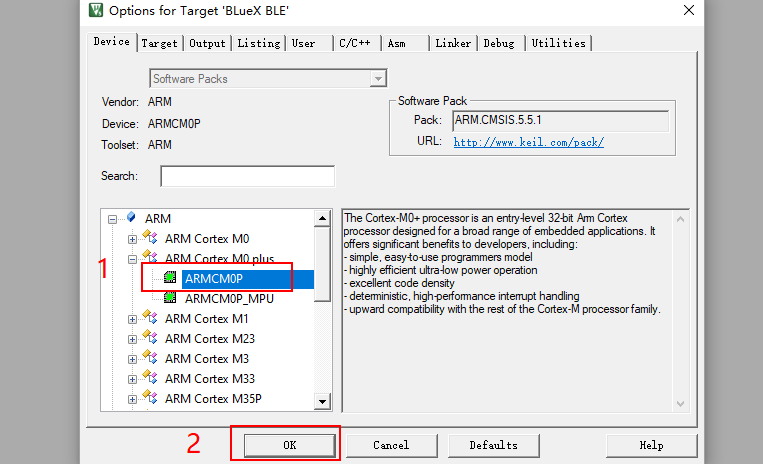


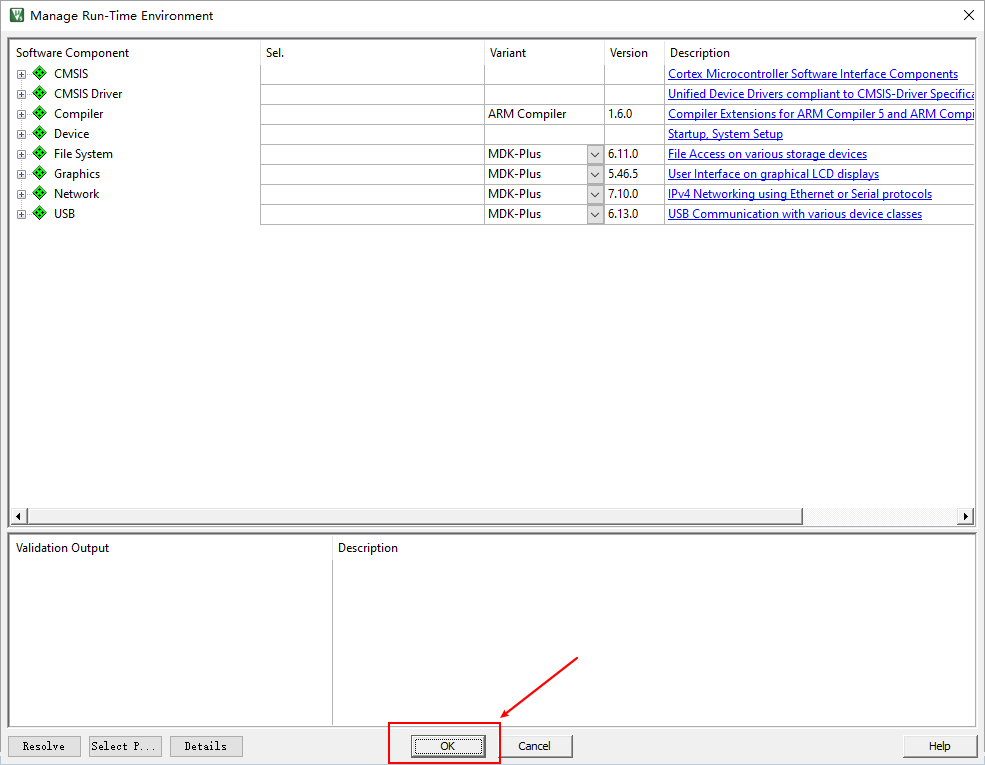
打开MDK：



此处工程保存在sdk的application中，命名为template：

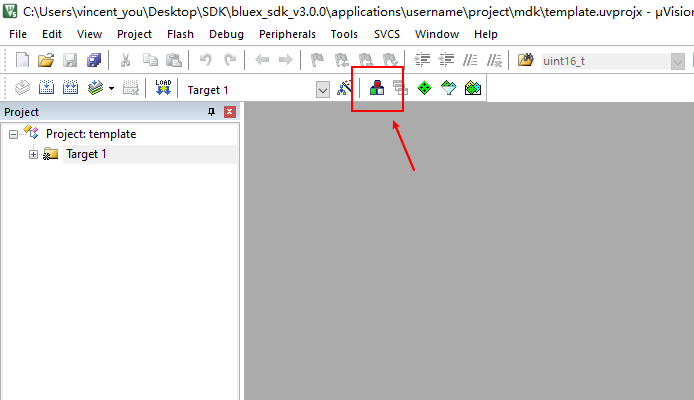


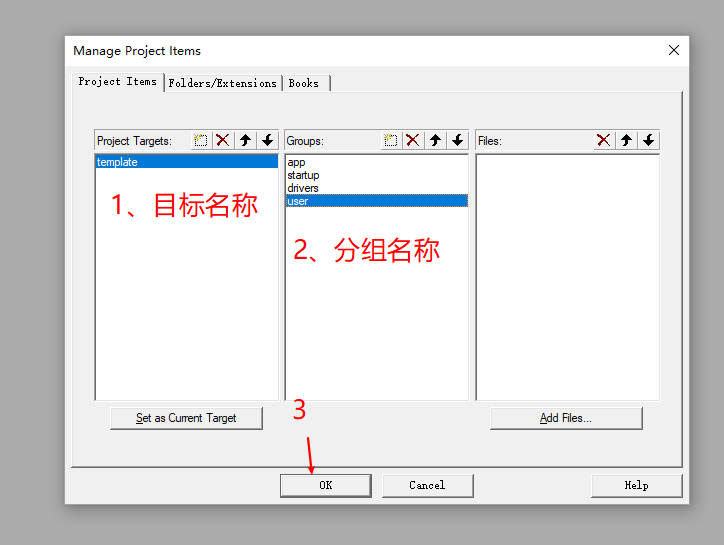


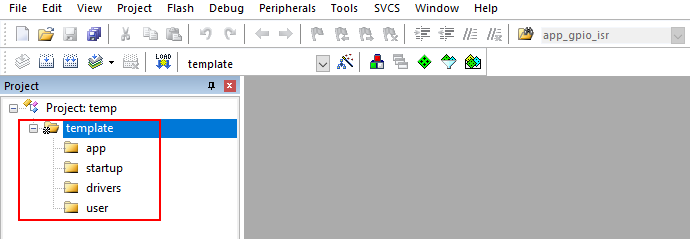


什么都不用选，直接OK

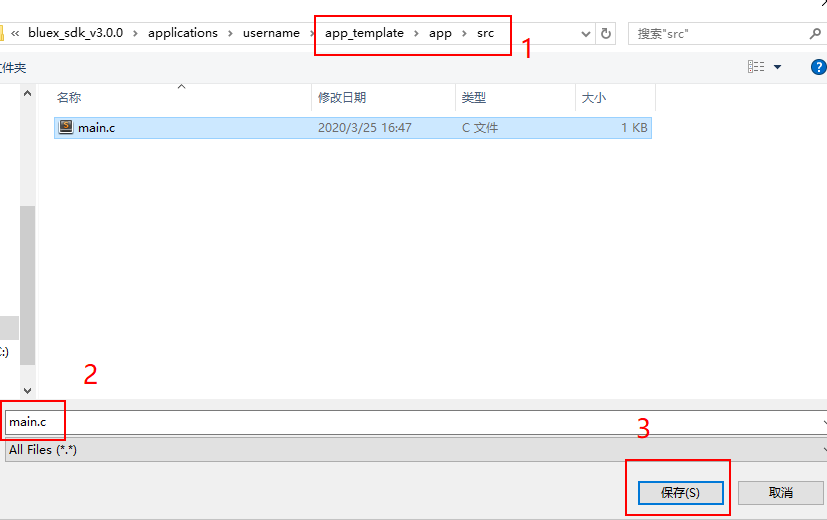
## 2、规划工程目录



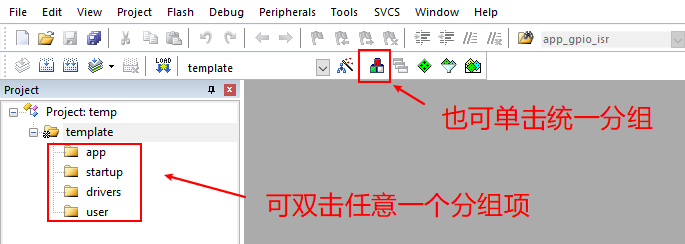


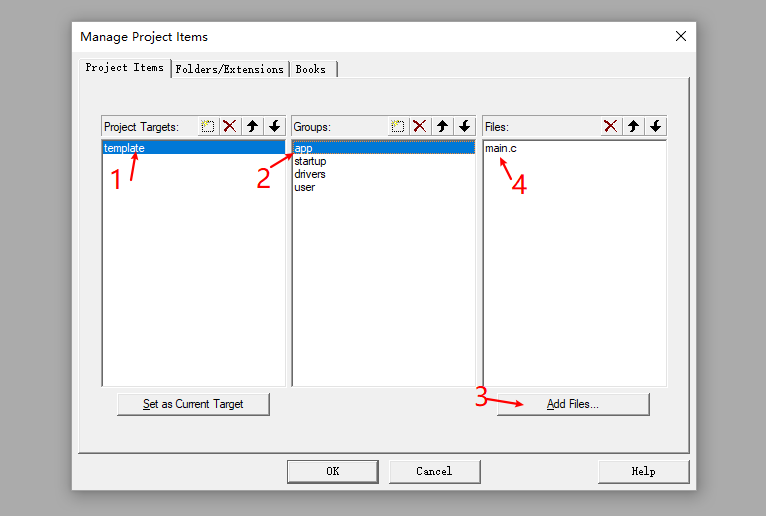


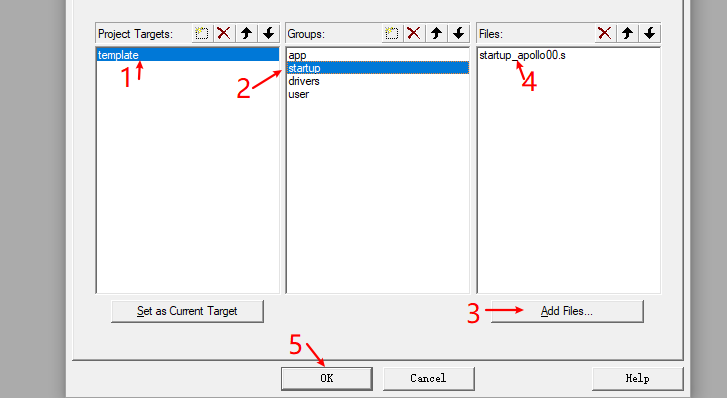
Ctrl+N，然后Ctrl+s，命名为main.c，并将其保存在以下目录：



## 3、添加文件



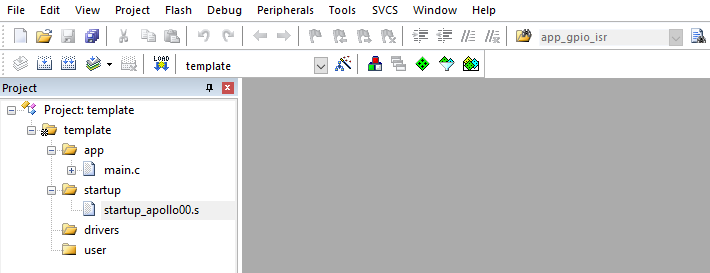




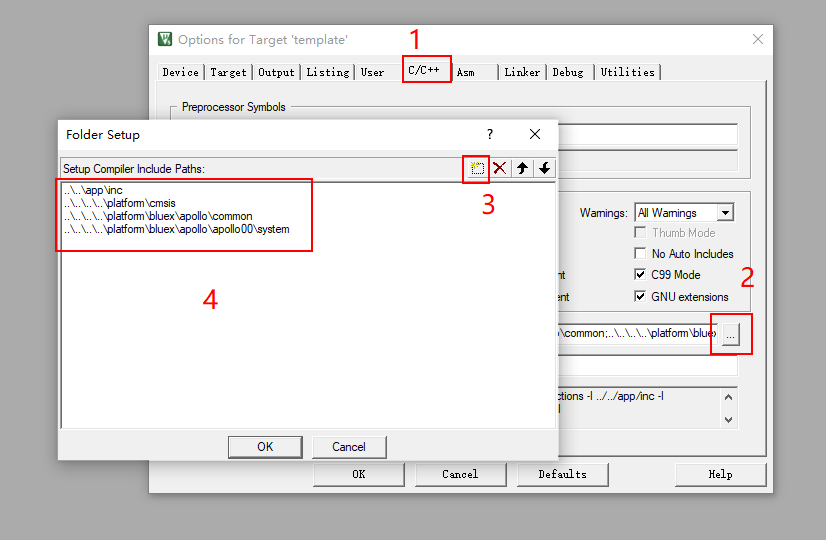
起处使用apollo00的启动文件作为演示，不同平台的.s启动文件在其相对应的路径下:



Drivers目录暂时不需要添加文件。然后点击OK即可，添加后文件目录如下：



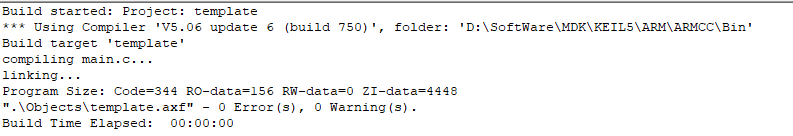
## 4、工程配置



点击OK即可，然后打开main.c输入以下代码:

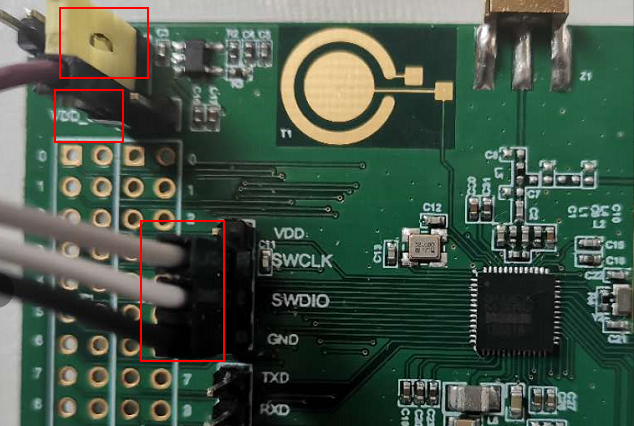
1. **int** main( **void** )
2. {
4. }

可正常通过编译:



# 第五章 固件烧录

## 1、硬件连接



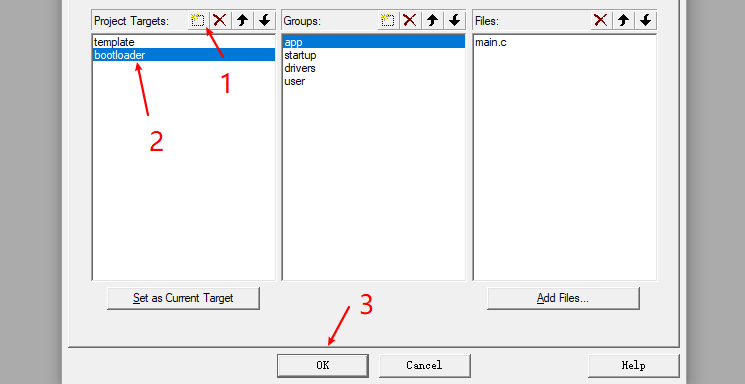


图中紫色-VDD、黑色-GND、白色-SWCLK、灰色-SWDIO。

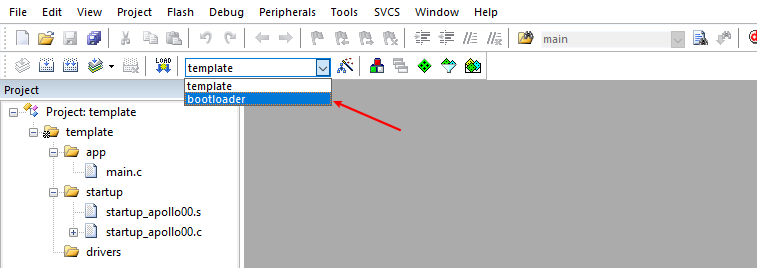
注意:Jlink的版本不同，VDD有时1引脚，有时在2引脚，图中接入的为2引脚，用户请自行判断VDD接入点，建议使用外部供电，然后共地。

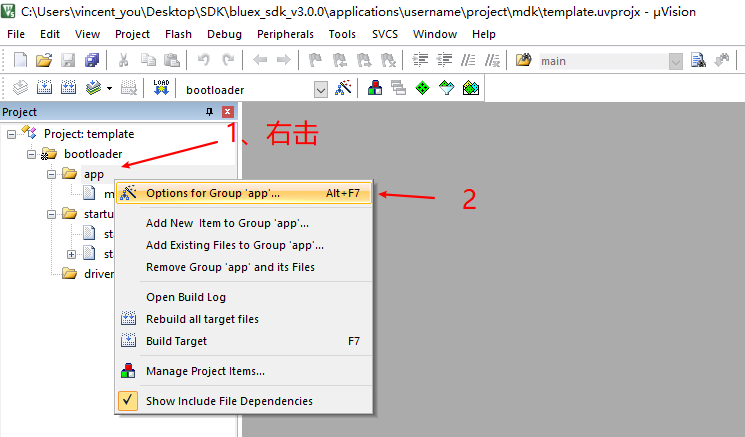
## 2、MKD直接下载

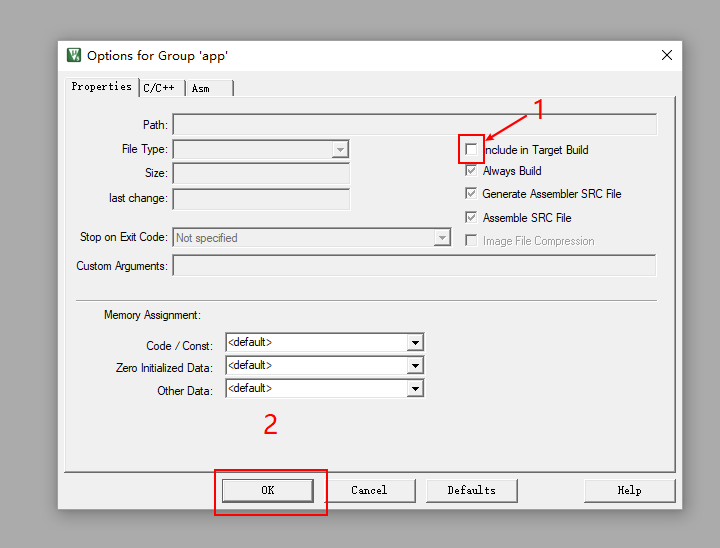
以下载bootloader作为演示，后面的例程都是需要烧录bootloader才能正常运行的（SDK中example的外设例程全都都配置好bootloader的烧录方式）。首先新建一个目标名称：



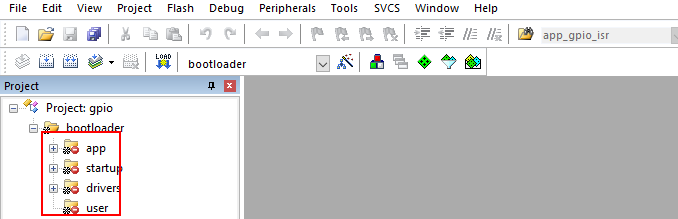
选中bootloader目标：



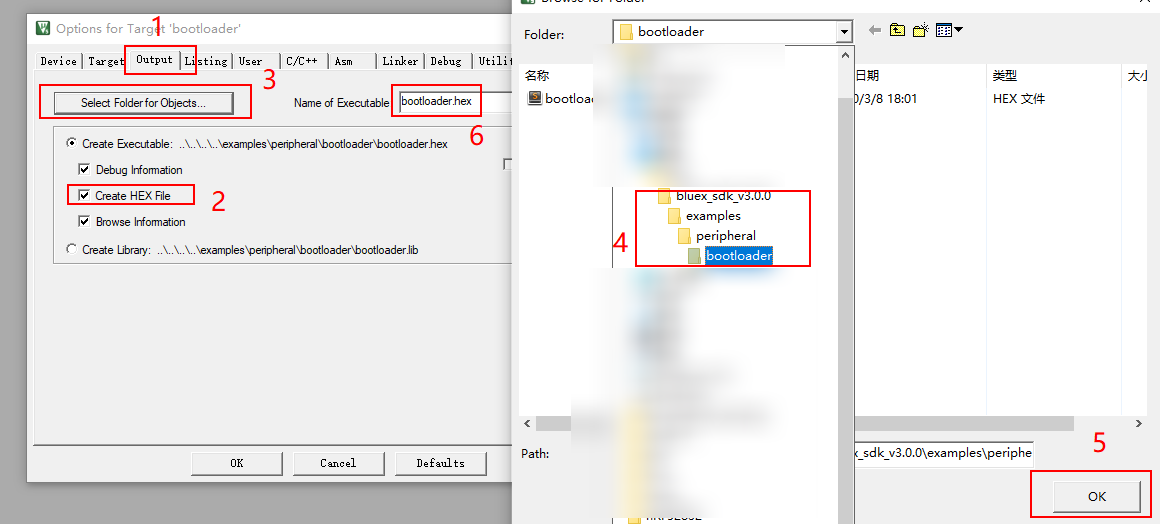




同理去掉所有：

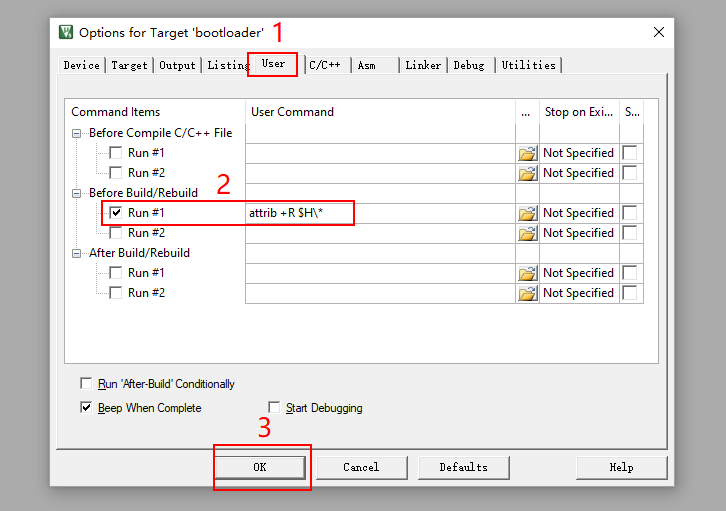




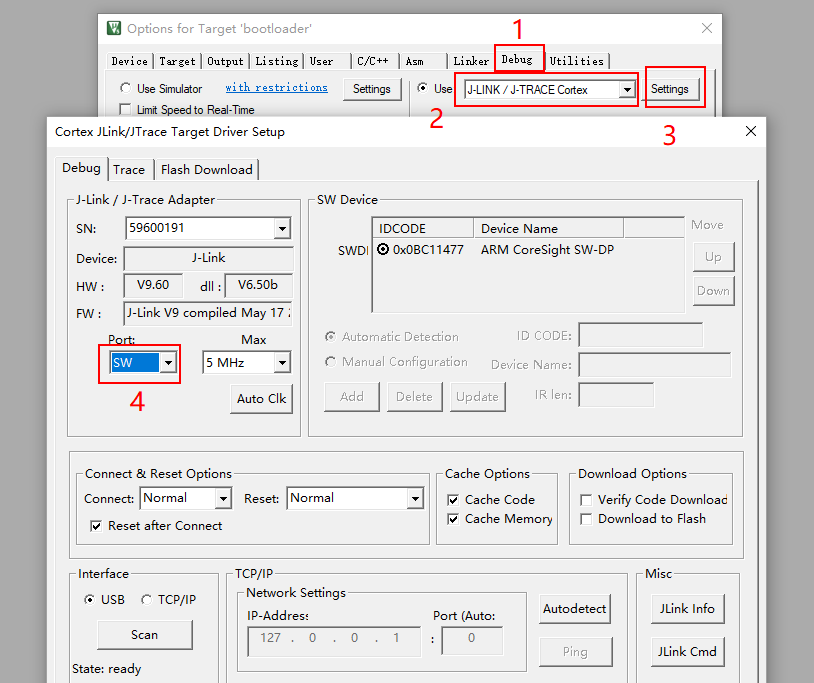


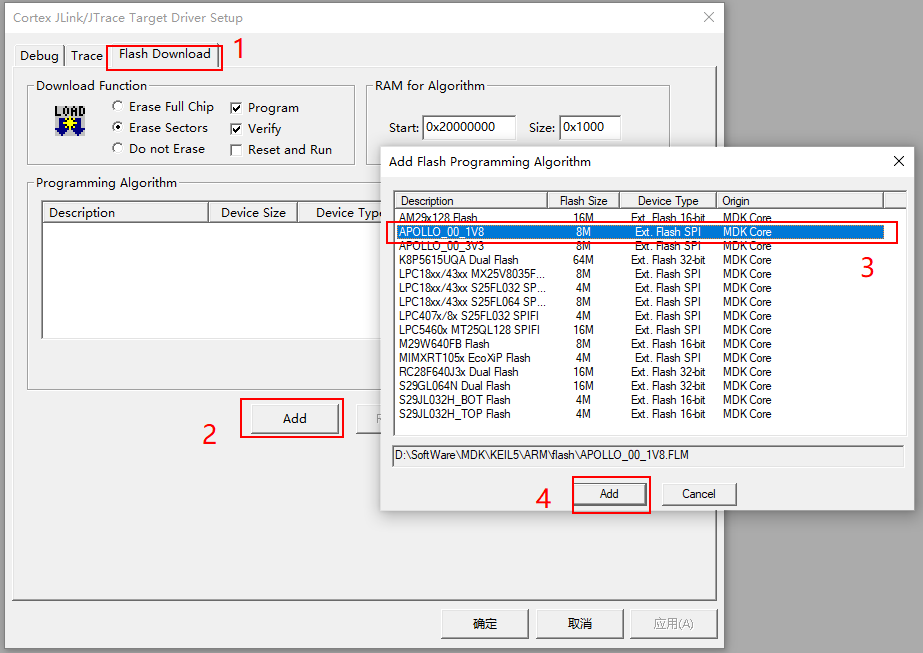
注意图中红框第4步，选择的路径必须跟图中所示一致。

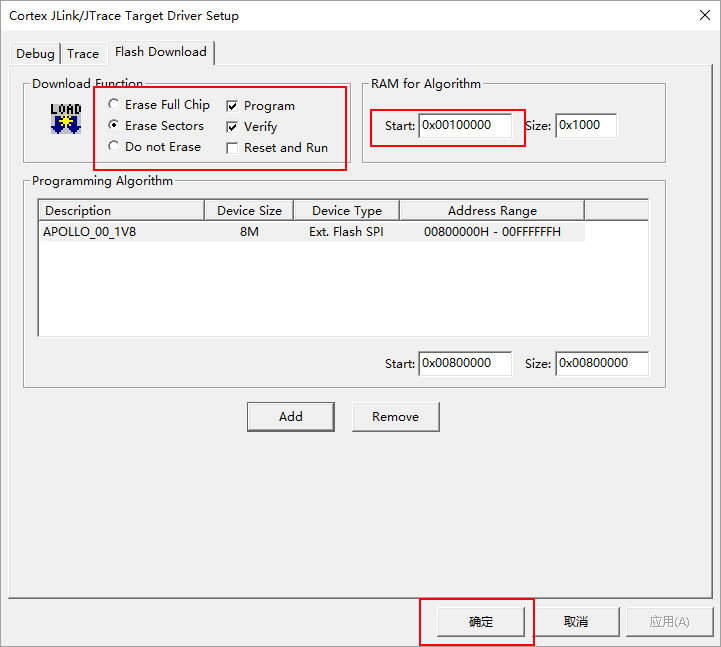
注意图中红框第6步，写入的名字必须跟bootloader文件夹中hex文件的文件名一致，注意把后缀.hex也写上。



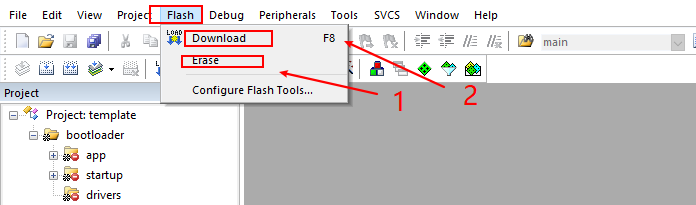
输入此命令“attrib +R $H\\*”







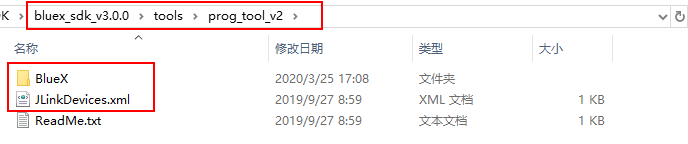
为确保烧录的是此bootloader，请先擦除芯片，然后再烧录。



先擦除，后烧写。

## 3、J-Flash下载

把目录下的 Bluex 文件夹和 JLinkDevices.xml 复制到 JLink 目录下，如图所示：



目录下的

## 4、常见错误

# 第六章 点亮LED

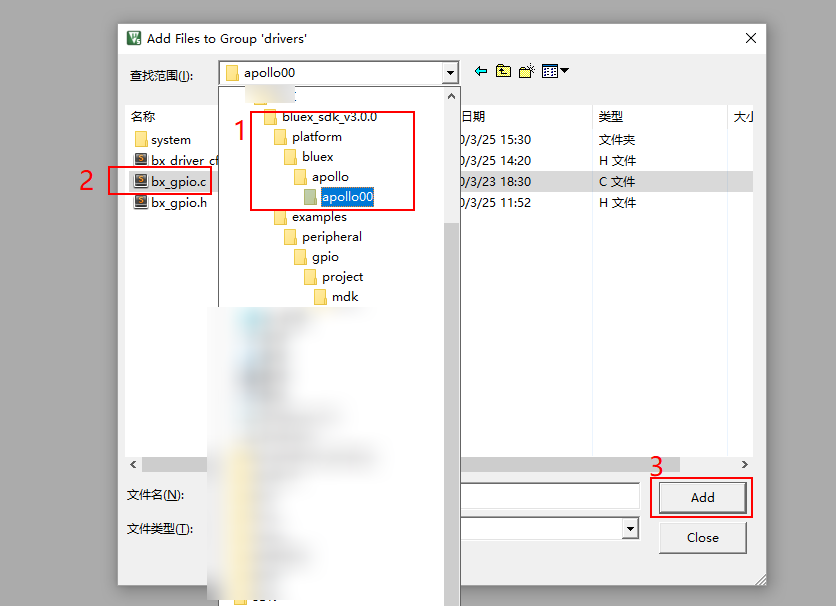
如果还没看第四、第五章节，建议先查看第四、第五章节内容。

## 1、新建工程

新建工程与第4章基本一致，只是路径跟名称稍微修改了一下，直接复制一份，将工程名称修改为gpio即可。



添加GPIO文件：



## 2、代码编写

1. **void** user\_delay( uint32\_t val )
2. {
3. **for**( uint32\_t i = 0; i < val; i++ )
4. **for**( uint32\_t j = 0; j < 5000; j++ );
5. }
7. **void** output\_test( **void** )
8. {
9. uint32\_t pin\_mark = GPIO\_PIN\_2 | GPIO\_PIN\_3;
10. gpio\_clk\_set();
11. gpio\_pin\_cfg\_output( pin\_mark );
12. **while**( 1 ) {
13. gpio\_pin\_toggle( pin\_mark );
14. user\_delay( 500 );
15. }
16. }
17. /\*\* ---------------------------------------------------------------------------
18. \* @brief   :
19. \* @note    :
20. \* @param   :
21. \* @retval  :
22. -----------------------------------------------------------------------------\*/
23. **int** main( **void** )
24. {
25. /\* peripheral init \*/
27. /\* board \*/
29. /\* component \*/
31. /\* device \*/
33. /\* user \*/
35. \_\_DMB();
36. SCB->VTOR = 0x00803000;
37. \_\_DSB();
39. output\_test()
41. **while**( 1 );
42. }

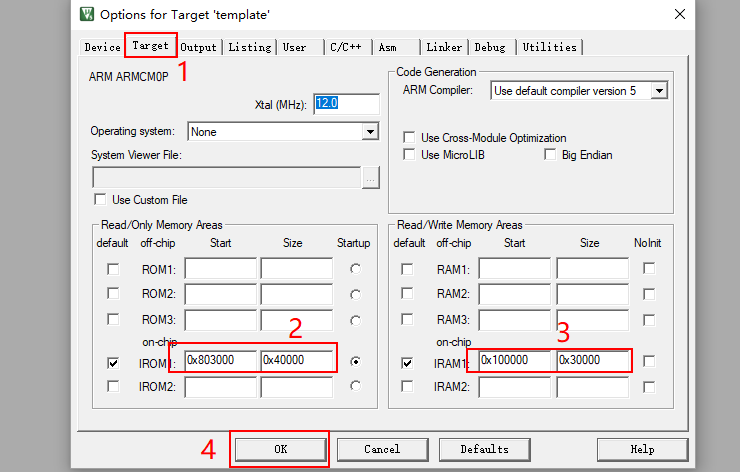
此处只需编写初级演示代码，具体API及寄存器放在后4、5、6小结上，详细可参考example例程。

## 3、实例演示

### 3.1硬件连接

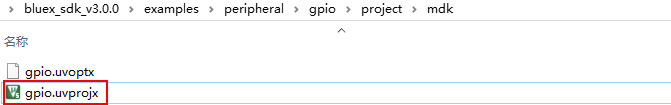
开发板正常连接Jlink。

### 3.2软件配置



### 3.3演示步骤

* 打开SDK中的gpio的例程



* 编译文件
* 按本章硬件连接与软件配置要求正确操作
* 按第五章方式烧录固件
* 重启开发板，观察实验现象与代码是否相符合

## 4、芯片GPIO介绍

### 4.1寄存器

## 5、API简介

|  |  |
| --- | --- |
| 原型 |  |
| 参数 |  |
| 功能 |  |

# 第七章 调试输出

## 1、新建工程

## 2、代码编写

## 3、实例演示

### 3.1硬件连接

### 3.2软件配置

### 3.3演示步骤

# 第八章 BLE连接

## 1、BLE连接流程

## 2、硬件设计

## 3、代码编写

## 4、实例演示

### 4.1硬件连接

### 4.2软件配置

### 4.3演示步骤

## 5、API简介

|  |  |
| --- | --- |
| 原型 |  |
| 参数 |  |
| 功能 |  |
| 响应事件 |  |
| 触发消息 |  |

只需说明复杂函数即可，功能函数需要简单说明调用函数后会触发什么动作/事件/消息，主要针对CEVA类。

## 6、相关文档

只需说明相关文档名称即可，如果可以最好具体到文档章节。

# 第九章 BLE数据交换

## 1、BLE数据交换方式

## 2、硬件设计

## 3、代码编写

## 4、实例演示

### 4.1硬件连接

### 4.2软件配置

### 4.3演示步骤

## 5、API简介

|  |  |
| --- | --- |
| 原型 |  |
| 参数 |  |
| 功能 |  |
| 响应事件 |  |
| 触发消息 |  |

只需说明复杂函数即可，功能函数需要简单说明调用函数后会触发什么动作/事件/消息，主要针对CEVA类。

## 6、相关文档

只需说明相关文档名称即可，如果可以最好具体到文档章节。

# 第十章 BLE从机

## 1、基础知识

## 2、硬件设计

## 3、代码编写

## 4、实例演示

### 4.1硬件连接

### 4.2软件配置

### 4.3演示步骤

## 5、API简介

|  |  |
| --- | --- |
| 原型 |  |
| 参数 |  |
| 功能 |  |
| 响应事件 |  |
| 触发消息 |  |

只需说明复杂函数即可，功能函数需要简单说明调用函数后会触发什么动作/事件/消息，主要针对CEVA类。

## 6、相关文档

只需说明相关文档名称即可，如果可以最好具体到文档章节。

# 第十一章 BLE主机

## 1、基础知识

## 2、硬件设计

## 3、代码编写

## 4、实例演示

### 4.1硬件连接

### 4.2软件配置

### 4.3演示步骤

## 5、API简介

|  |  |
| --- | --- |
| 原型 |  |
| 参数 |  |
| 功能 |  |
| 响应事件 |  |
| 触发消息 |  |

只需说明复杂函数即可，功能函数需要简单说明调用函数后会触发什么动作/事件/消息，主要针对CEVA类。

## 6、相关文档

只需说明相关文档名称即可，如果可以最好具体到文档章节。

# 第十二章 BLE主从一体

## 1、基础知识

## 2、硬件设计

## 3、代码编写

## 4、实例演示

### 4.1硬件连接

### 4.2软件配置

### 4.3演示步骤

## 5、API简介

|  |  |
| --- | --- |
| 原型 |  |
| 参数 |  |
| 功能 |  |
| 响应事件 |  |
| 触发消息 |  |

只需说明复杂函数即可，功能函数需要简单说明调用函数后会触发什么动作/事件/消息，主要针对CEVA类。

## 6、相关文档

只需说明相关文档名称即可，如果可以最好具体到文档章节。

# 第十三章 BLE组网MESH

## 1、基础知识

## 2、硬件设计

## 3、代码编写

## 4、实例演示

### 4.1硬件连接

### 4.2软件配置

### 4.3演示步骤

## 5、API简介

|  |  |
| --- | --- |
| 原型 |  |
| 参数 |  |
| 功能 |  |
| 响应事件 |  |
| 触发消息 |  |

只需说明复杂函数即可，功能函数需要简单说明调用函数后会触发什么动作/事件/消息，主要针对CEVA类。

## 6、相关文档

只需说明相关文档名称即可，如果可以最好具体到文档章节。