



## 建立基于 STM32 固件库的工程模版(ALIENTEK)

这是一个独立的建立工程的资料，如果您手头没有我们 ALIENTEK STM32 开发板的源码和资料，请您到我们 ALIENTEK 官方技术论坛:[www.openedv.com](http://www.openedv.com) 下载，下载地址为：<http://openedv.com/posts/list/565.htm> 所有资料均免费提供。

初学者搭建开发环境建立工程模版大约需要 2-3 个小时，请耐心等待按照手册的讲解来。

一. 需要准备的资料:

1. STM32F10x\_StdPeriph\_Lib\_V3.5.0 (这是 ST 官网下载的固件库完整版)

我们官方论坛下载地址：<http://openedv.com/posts/list/6054.htm>

2. MDK3.8a (我们的板子的开发环境目前是使用这个版本)

二. 安装 MDK3.8a (Keil): 这个在我们不完全手册中已经讲解，这里重复一次。

1.找到 MDK 的安装文件并点击安装  MDK380a :

文件在我们光盘的目录： ALIENTEK 开发板资料\软件\MDK3.80A

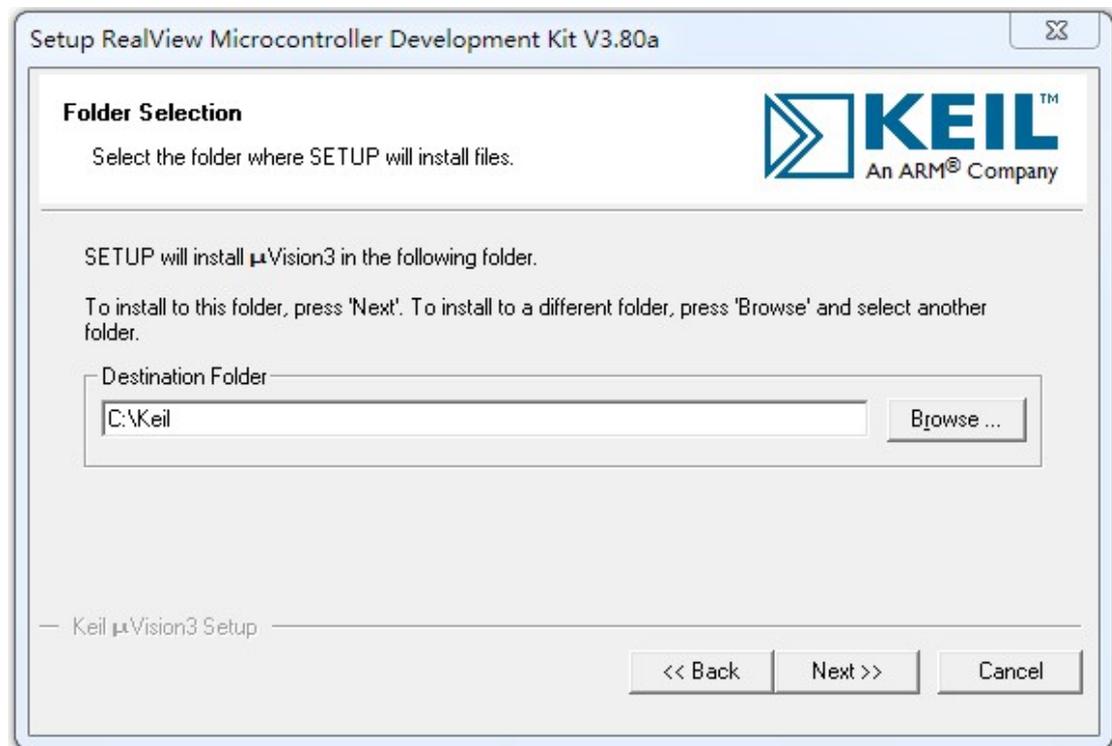
2.点击之后出现界面，选择 Next



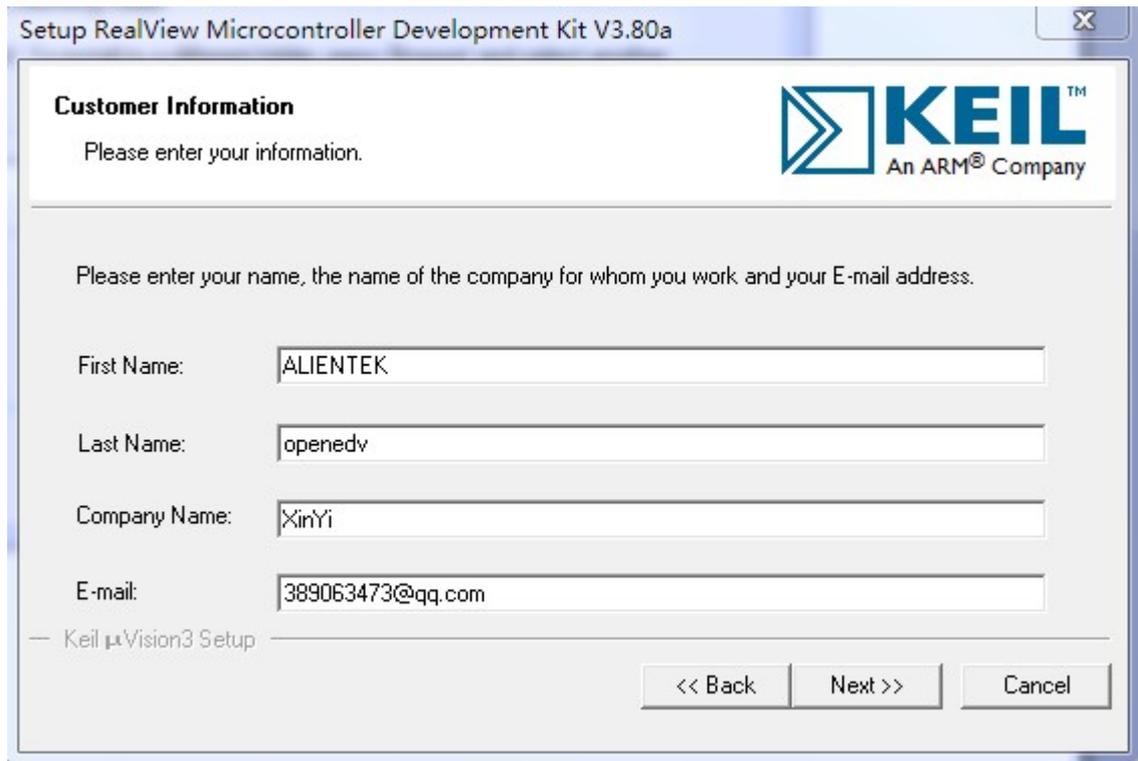
3.选择“I Agree...Licence agreement”同意协议:



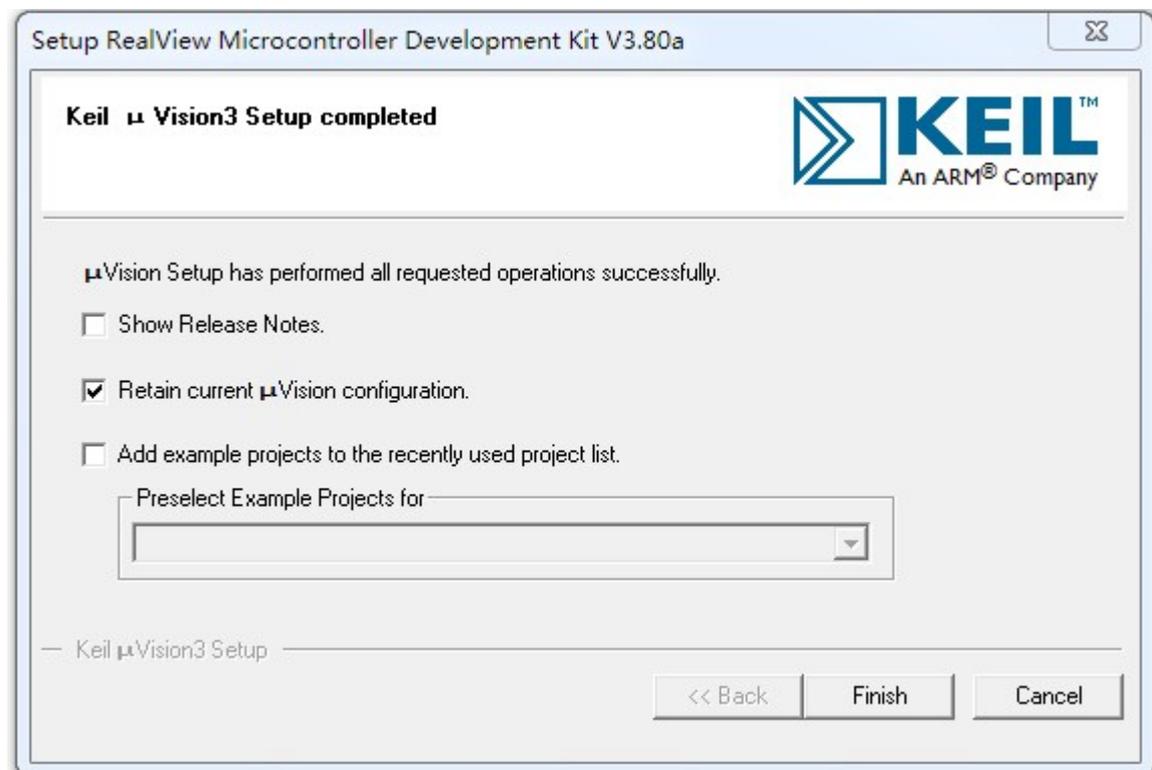
4. 选择安装目录，这里用户自行选择安装的目录后，点击 Next 即可：



5. 随便输入邮箱之类的信息即可，点击 Next 开始安装：



6. 按图配置，然后点击 Finish, 左面会出现 keil 快捷图标。



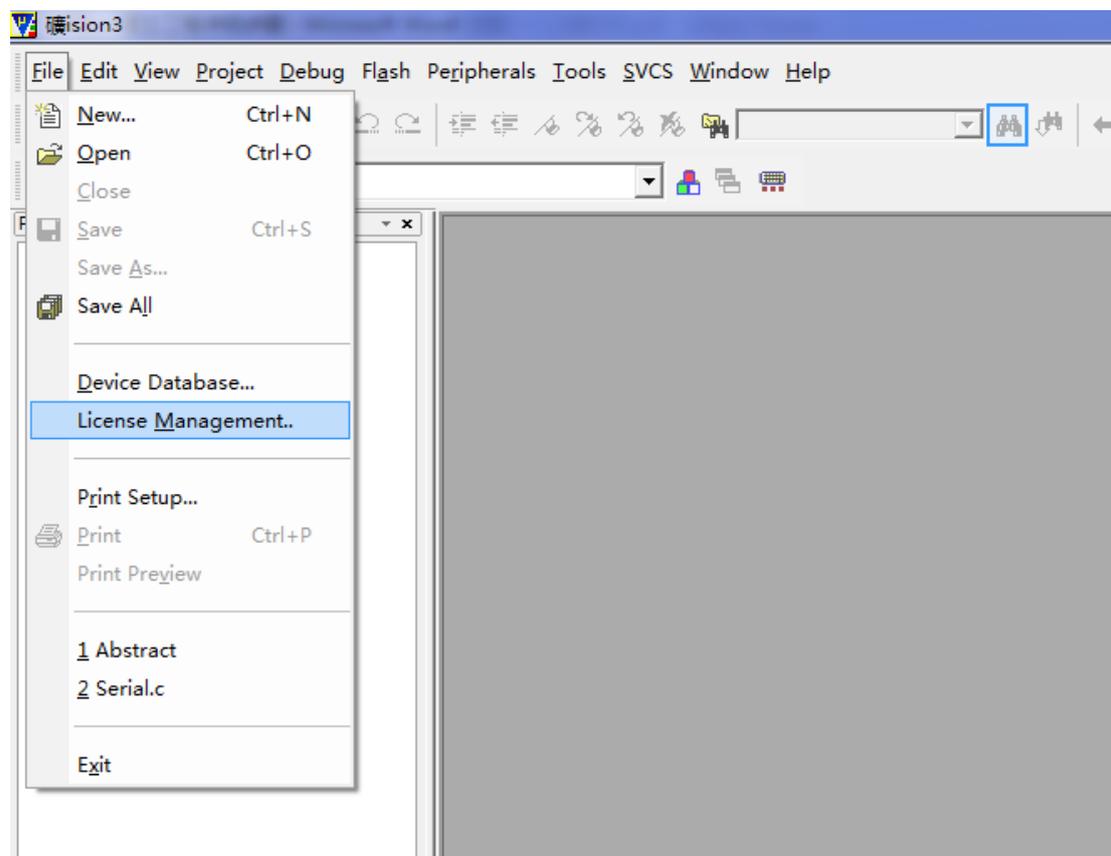
### 三. 注册 License(破解, 如果不破解只能支持 32K 的代码)

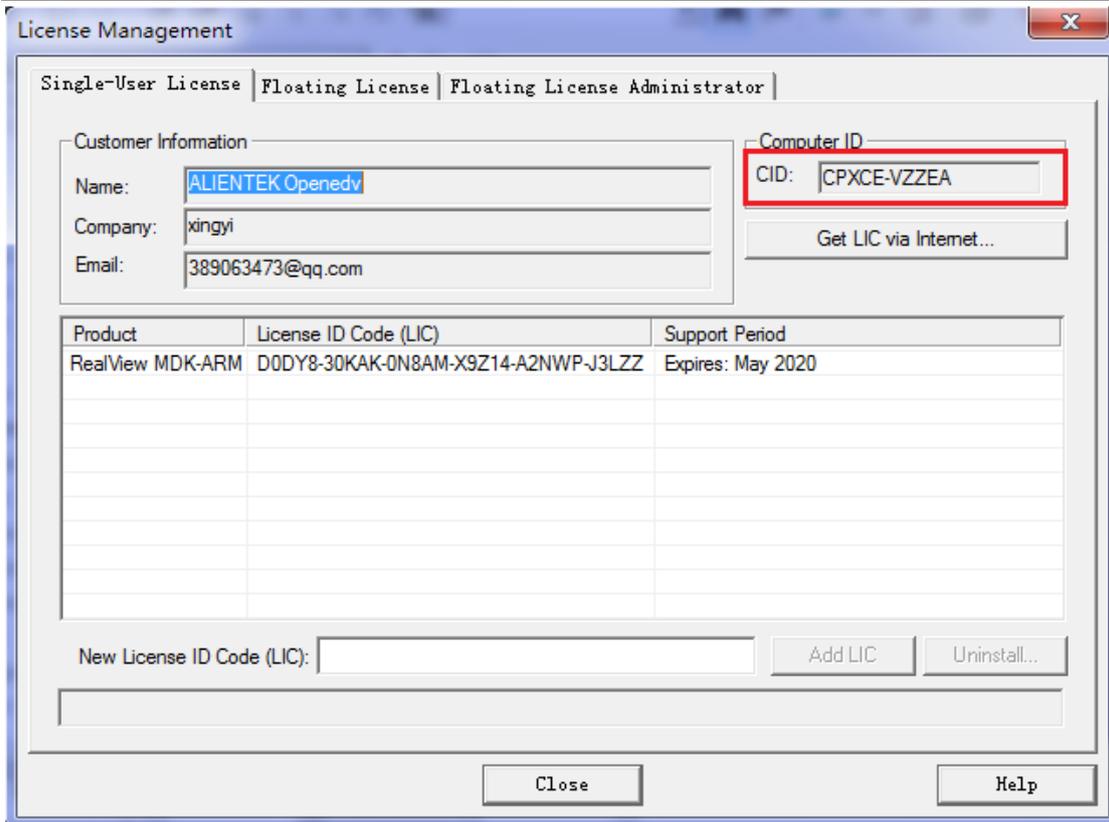
步骤简介:

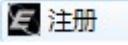
在 MDK 针对每台机会会有一个 CID, copy 这个 CID 到注册机处生成 License Key, 然后再将这个 License Key 添加到 MDK 里面去注册。

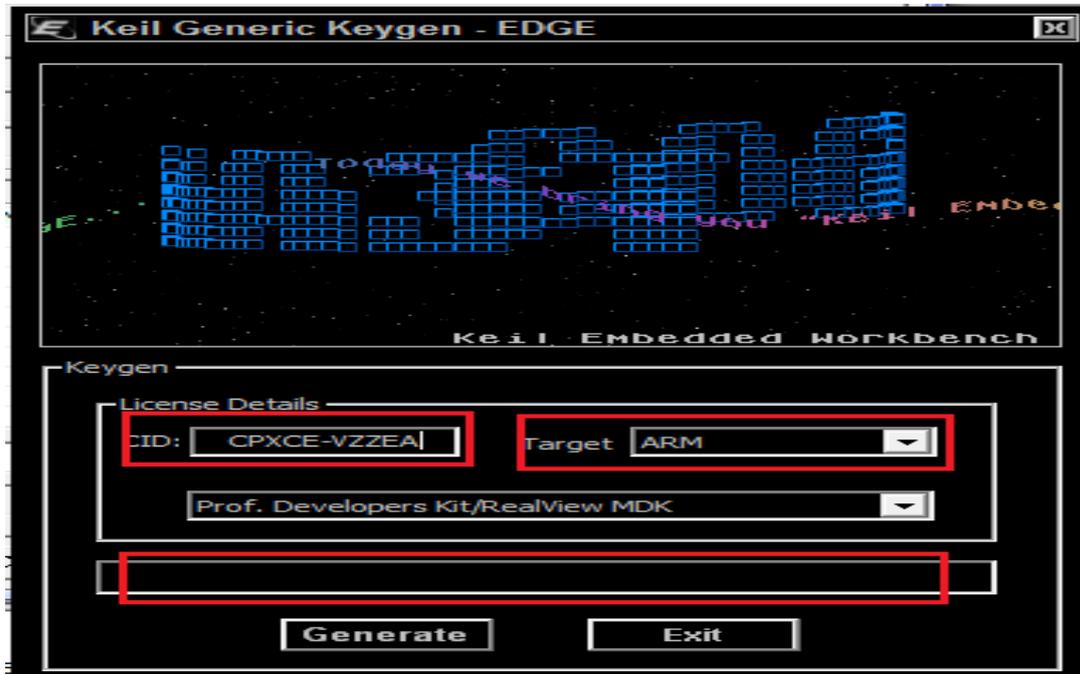
1. 右键点击左面的 MDK 快捷方式, 选择“以管理员身份运行”, 因为注册 license 需要管理员权限。然后会打开 MDK, 然后默认会有一个名字叫“LPC2129 simulator”的 Project, 暂时我们可以不用理会。

2. 点击: File->License Management, 弹出一个 License Management 界面, copy 界面中的 (CID):





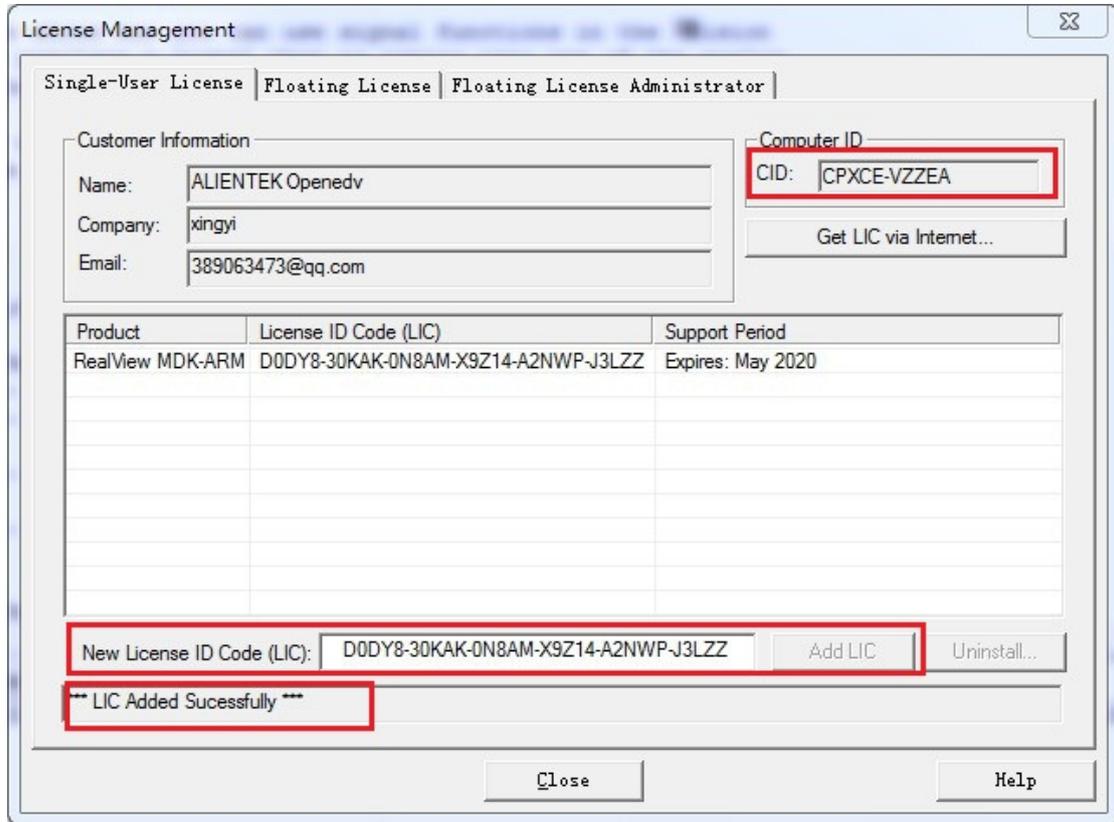
- 3. 打开光盘下面的注册机  ，目录为：ALIENTEK 开发板资料\软件\MDK3.80A\注册
- 4. 出现注册界面，黏贴刚才 copy 的 cid 到 CID 一栏，然后 Target 选择 ARM



- 5. 选择好之后点击“Generate”，下面的空白栏会生成一个 License Key，类似：D0DY8-30KAK-0N8AM-X9Z14-A2NWP-J3LZZ，copy 这个 license.

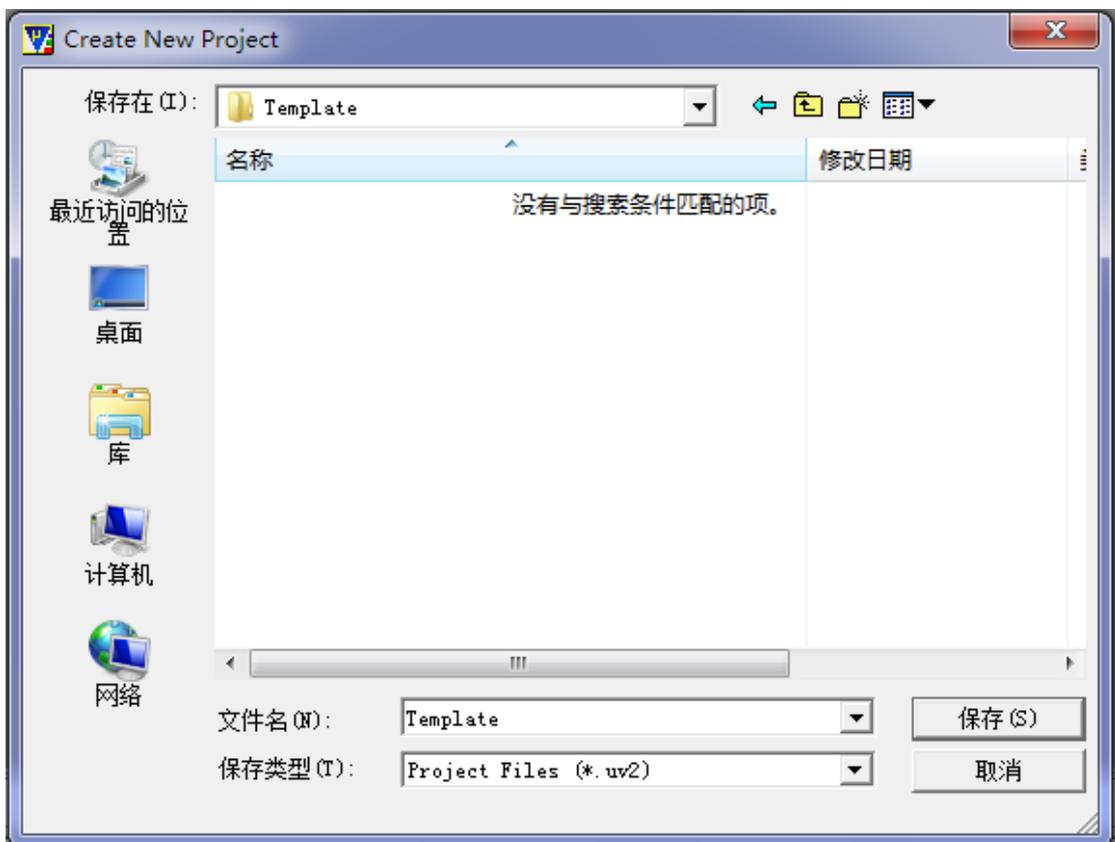
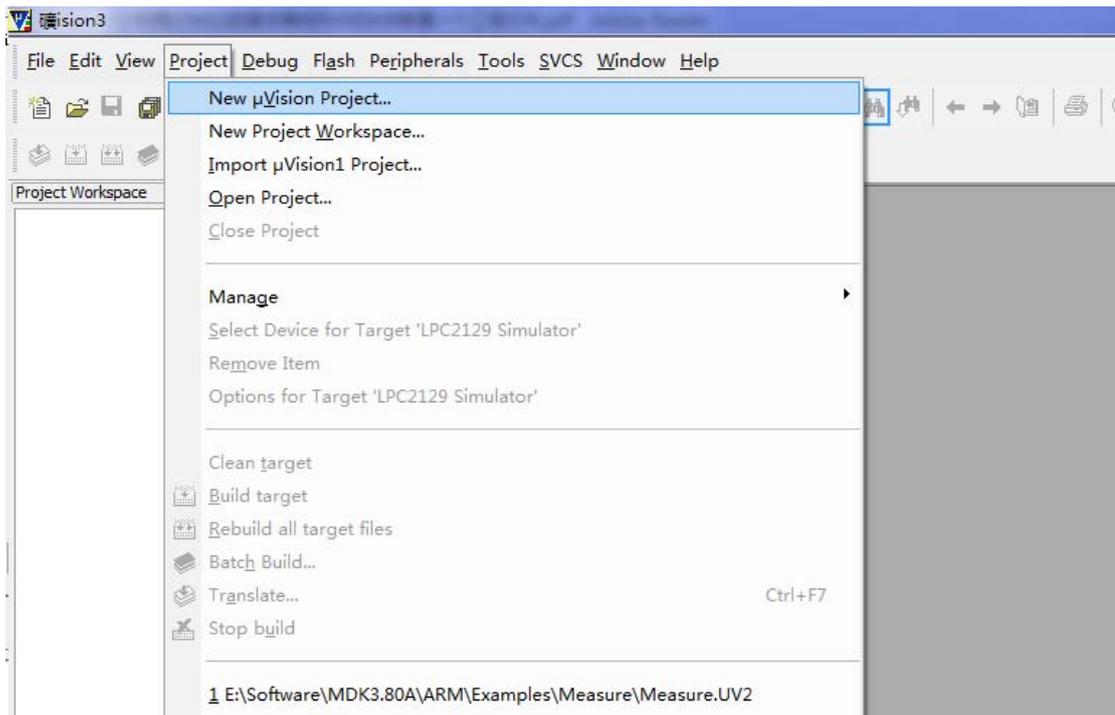


6. 将这个 License Key 黏贴到 Keil 的 License Management 界面的 New License Id Code 一栏，然后点击“Add LIC”，添加成功后会出现成功提示。然后点击 Close 关闭这个界面即可。

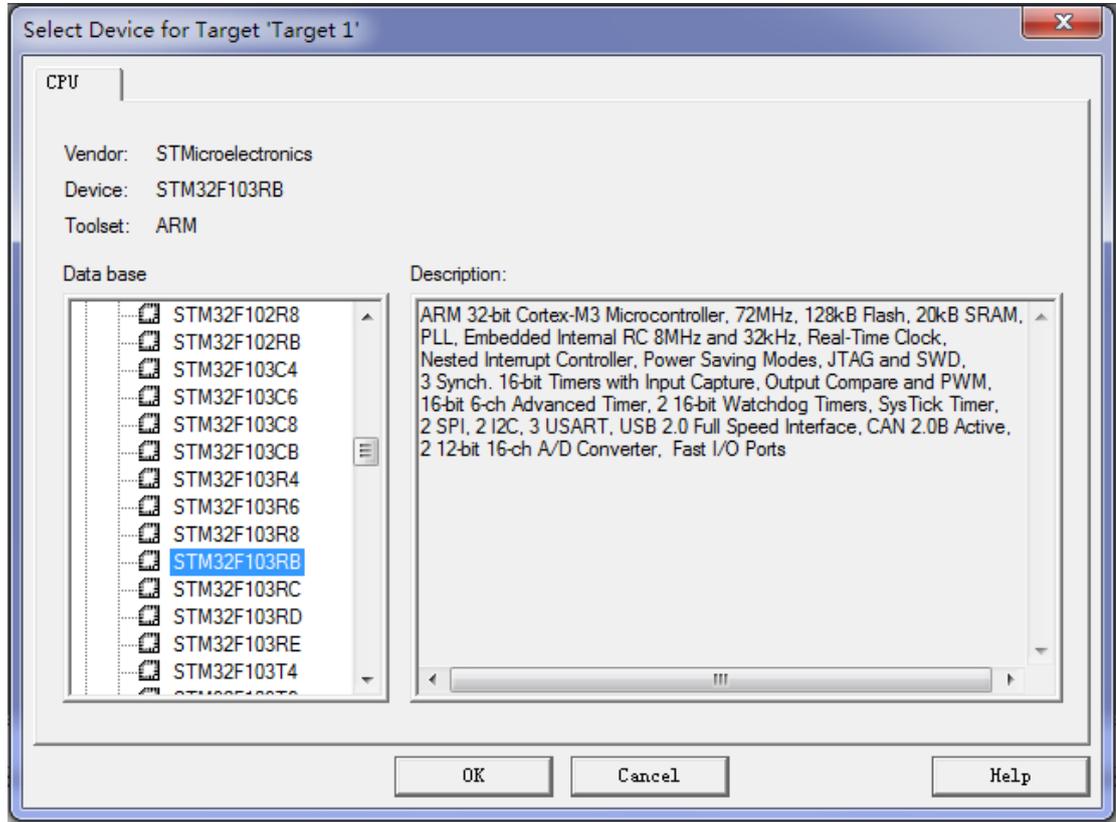


### 三. 新建工程

1. 回到 MDK 主界面，可以看到工程中有一个默认的工程，点击这个工程名字，然后选择菜单 Project->Close Project，就关闭掉这个工程了！这样整个 MDK 就是一个空的了，接下来我们将建立我们的工程模版。
2. 在建立工程之前，我们建议用户在电脑的某个目录下面建立一个文件夹，后面所建立的工程都可以放在这个文件夹下面，这里我们建立一个文件夹为: STM32-Projects.
3. 点击 Keil 的菜单: Project ->New Uvision Project ，然后将目录定位到刚才建立的文件夹 STM32-Projectst 之下，在这个目录下面建立子文件夹 Tempalte,然后定位到 Template 目录下面，我们的工程文件就都保存到 Template 文件夹下面。工程命名为 Template,点击保存

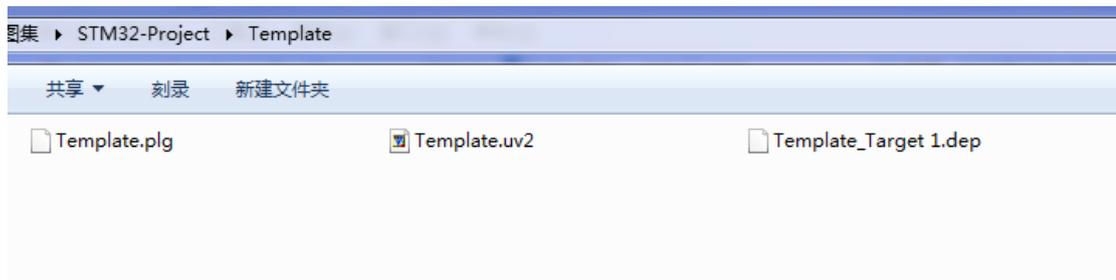


4.接下来会出现一个选择 Device 的界面，就是选择我们的芯片型号，这里我们定位到 STMicroelectronics 下面的 STM32F103RB(针对我们的 mini 板子是这个型号，如果是其他芯片，请选择对应的型号即可)。

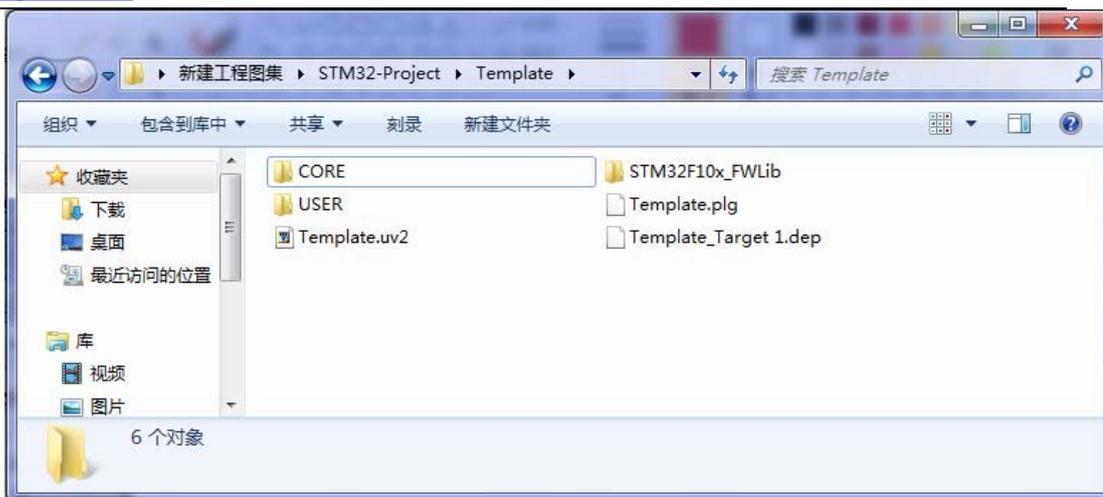


5.弹出对话框“Copy STM32 Startup Code to project ....”，询问是否添加启动代码到我们的工程中，这里我们选择“否”，因为我们使用的 ST 固件库文件已经包含了启动文件。

6.可以看到工程建立了，我们回到 Template 目录下面，可以看到只有三个文件：



7.接下来，我们在 Template 工程目录下面，新建 3 个文件夹 CORE, USER, STM32F10x\_FWLib。USER 用来放我们主函数文件 main.c,以及其他包括 system\_stm32f10x.c 等等，CORE 用来存放启动文件等，STM32F10x\_FWLib 文件夹顾名思义用来存放 ST 官方提供的库函数源码文件。



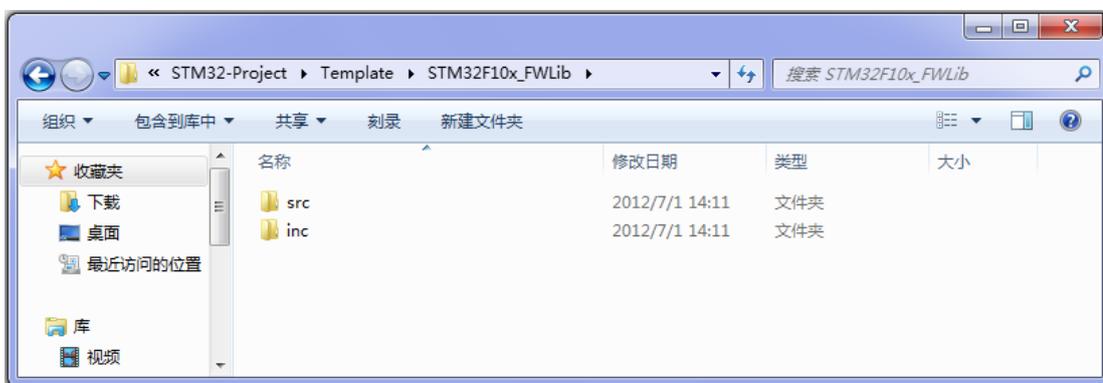
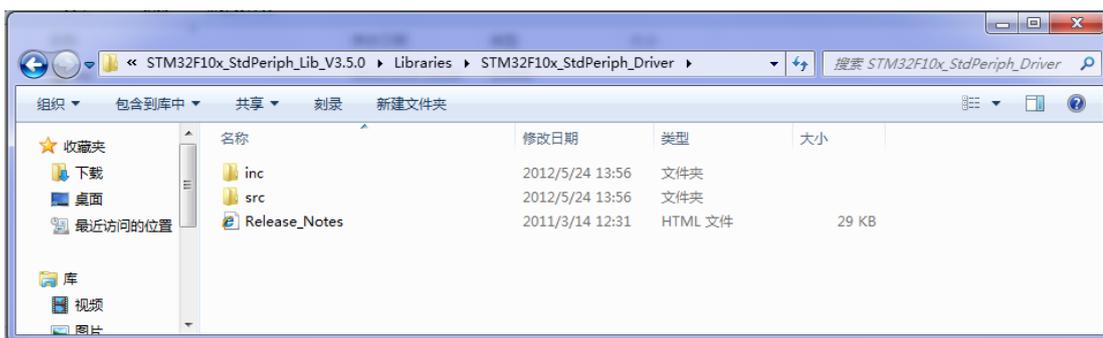
8.下面我们要将官方的固件库包里的源码文件复制到我们的工程目录文件夹下面。

打开官方固件库包，定位到我们之前准备好的固件库包的目录

STM32F10x\_StdPeriph\_Lib\_V3.5.0\Libraries\STM32F10x\_StdPeriph\_Driver 下面，

将目录下面的 src,inc 文件夹 copy 到我们刚才建立的 STM32F10x\_FWLib 文件夹下面。

src 存放的是固件库的.c 文件，inc 存放的是对应的.h 文件，您不妨打开这两个文件目录过目一下里面的文件，每个外设对应一个.c 文件和一个.h 头文件。



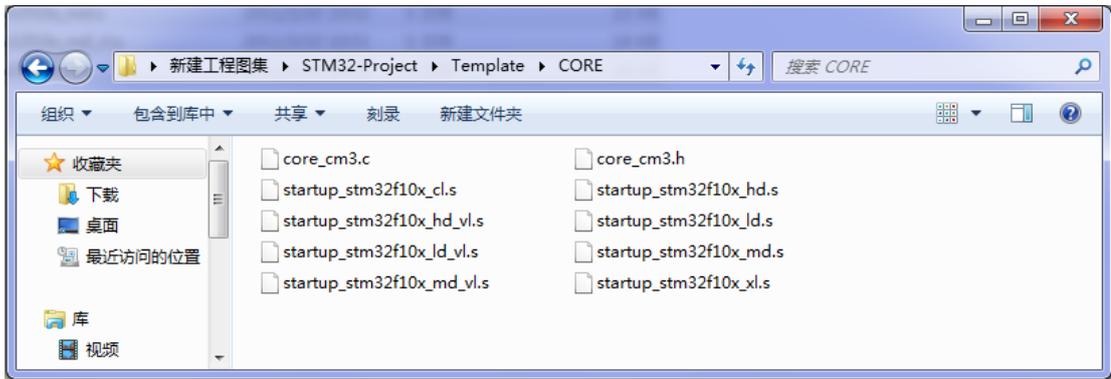


9.下面我们要将固件库包里面相关的启动文件复制到我们的工程目录 CORE 之下。

打开官方固件库包，定位到目录

STM32F10x\_StdPeriph\_Lib\_V3.5.0\Libraries\CMSIS\CM3\CoreSupport 下面，将文件 core\_cm3.c 和文件 core\_cm3.h 复制到 CORE 下面去。然后定位到目录 STM32F10x\_StdPeriph\_Lib\_V3.5.0\Libraries\CMSIS\CM3\DeviceSupport\ST\STM32F10x\startup\arm 下面，将里面所有的文件同样复制到 CORE 下面。这里我们解释一下，其实我们只用到 arm 目录下面的 startup\_stm32f10x\_md.s 文件，这个文件是针对中等容量芯片的启动文件。其他两个主要的为 startup\_stm32f10x\_ld.s 为小容量，startup\_stm32f10x\_hs.c 为大容量芯片的启动文件。这里 copy 进来是方便其他开发者使用小容量或者大容量芯片的用户。

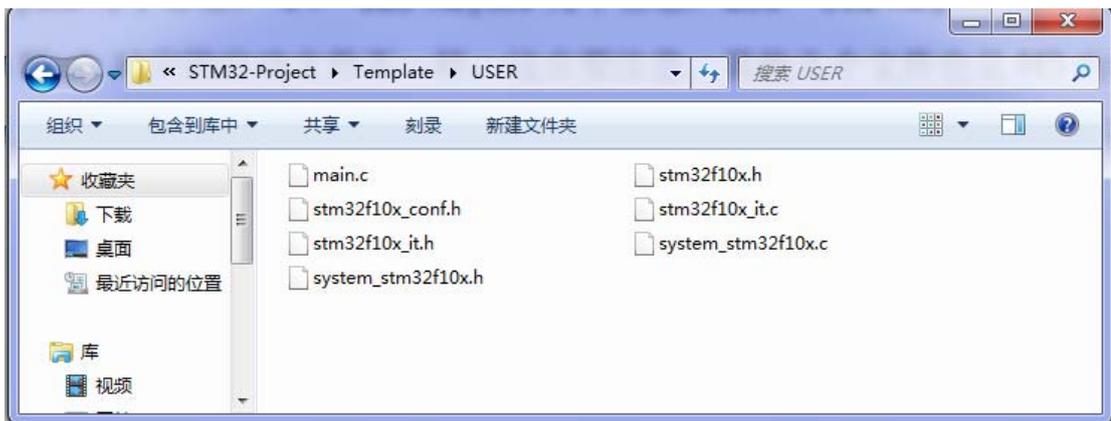
现在看看我们的 CORE 文件夹下面的文件：



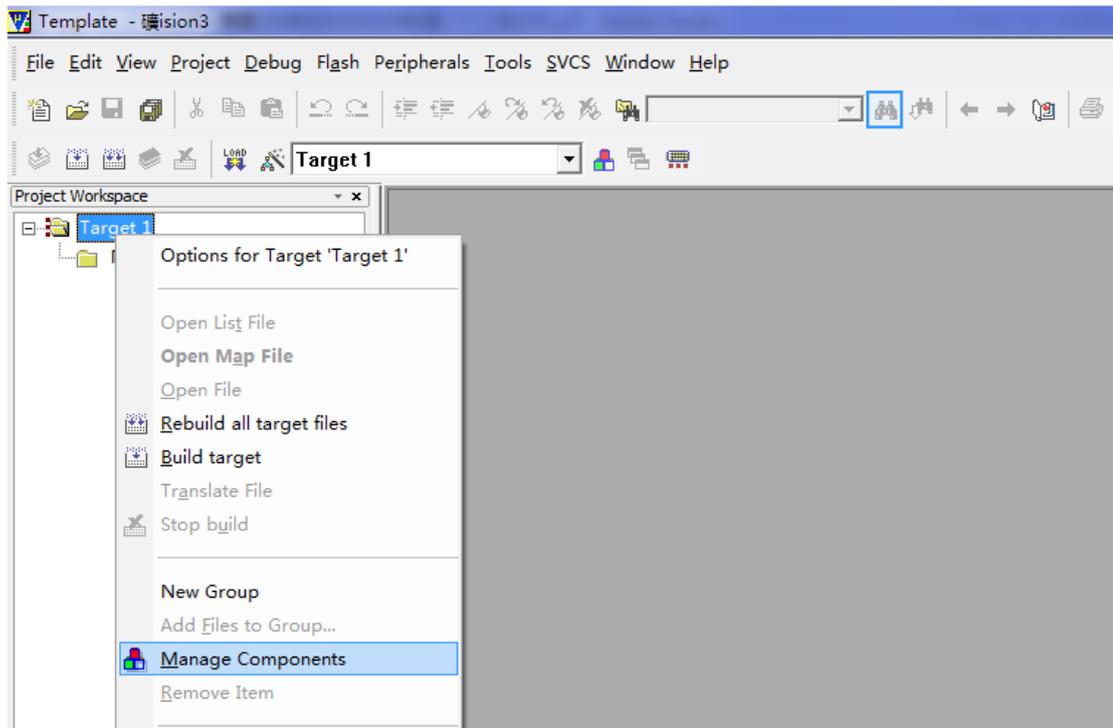
10.定位到目录：

STM32F10x\_StdPeriph\_Lib\_V3.5.0\Libraries\CMSIS\CM3\DeviceSupport\ST\STM32F10x 将里面的三个文件 stm32f10x.h, system\_stm32f10x.c, system\_stm32f10x.h, 复制到我们的 USER 目录之下。然后将

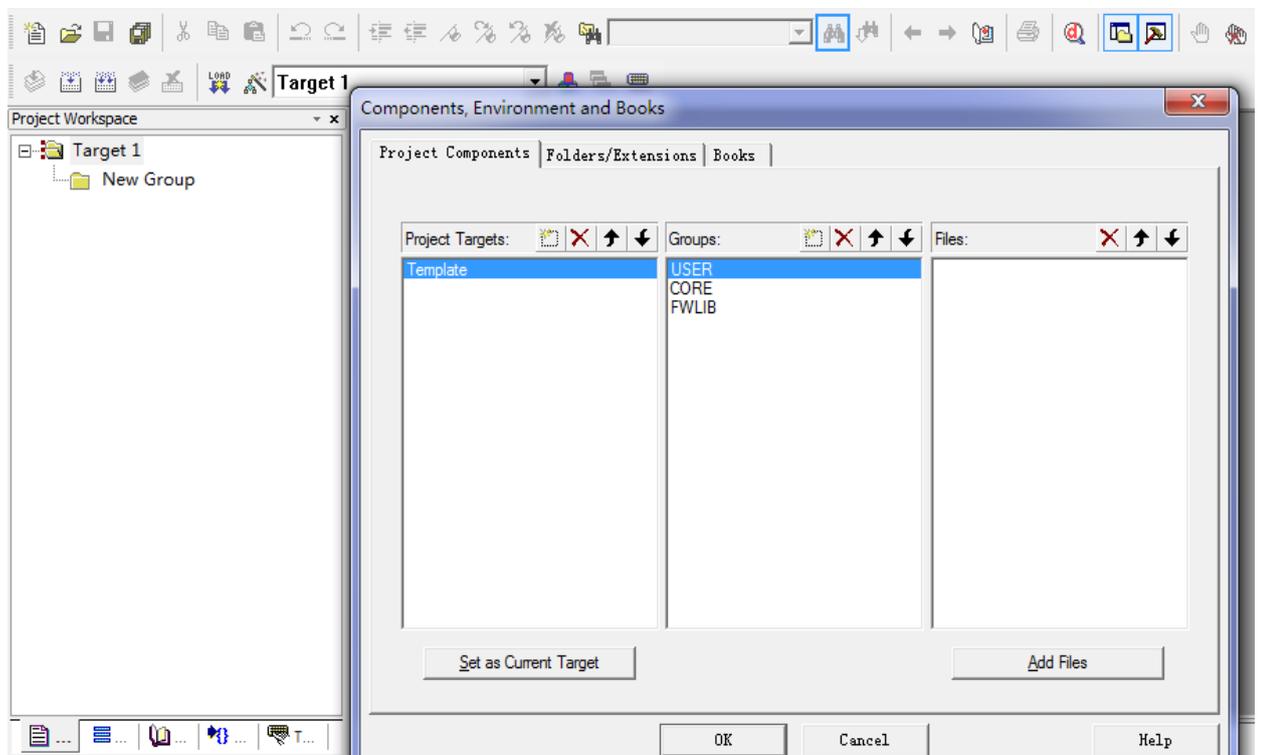
STM32F10x\_StdPeriph\_Lib\_V3.5.0\Project\STM32F10x\_StdPeriph\_Template 下面的 4 个文件 main.c, stm32f10x\_conf.h, stm32f10x\_it.c, stm32f10x\_it.h 复制到 USER 目录下面。

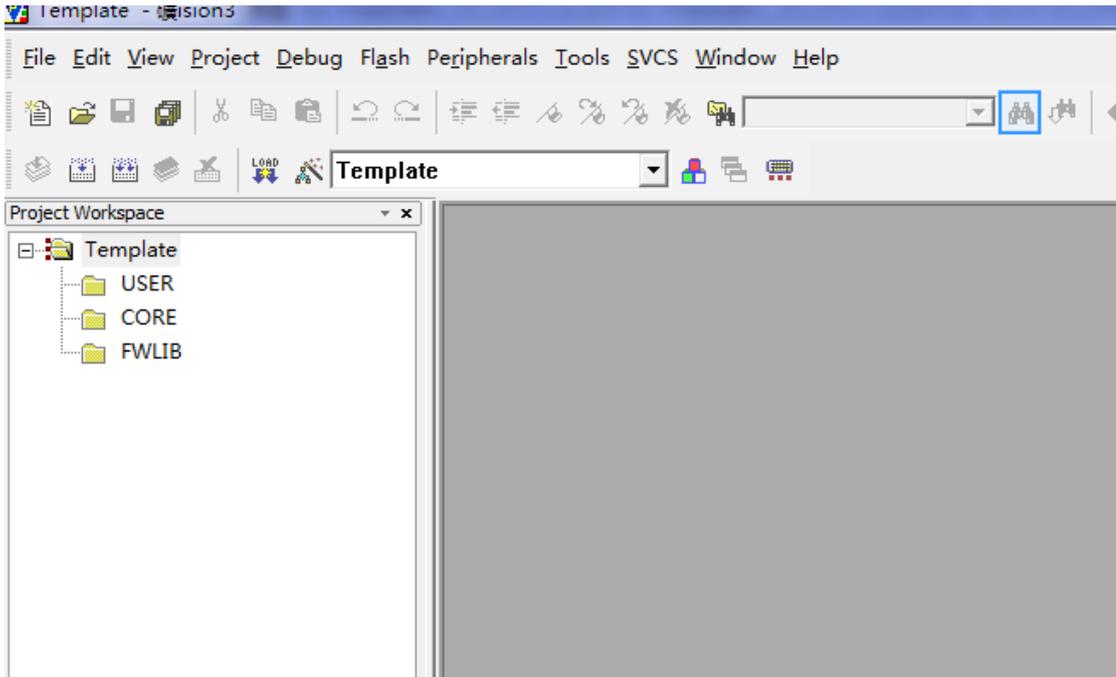


11.前面 10 个步骤，我们将需要的固件库相关文件复制到了我们的工程目录下面，下面我们将这些文件加入我们的工程中去。右键点击 Target1，选择 Manage Components



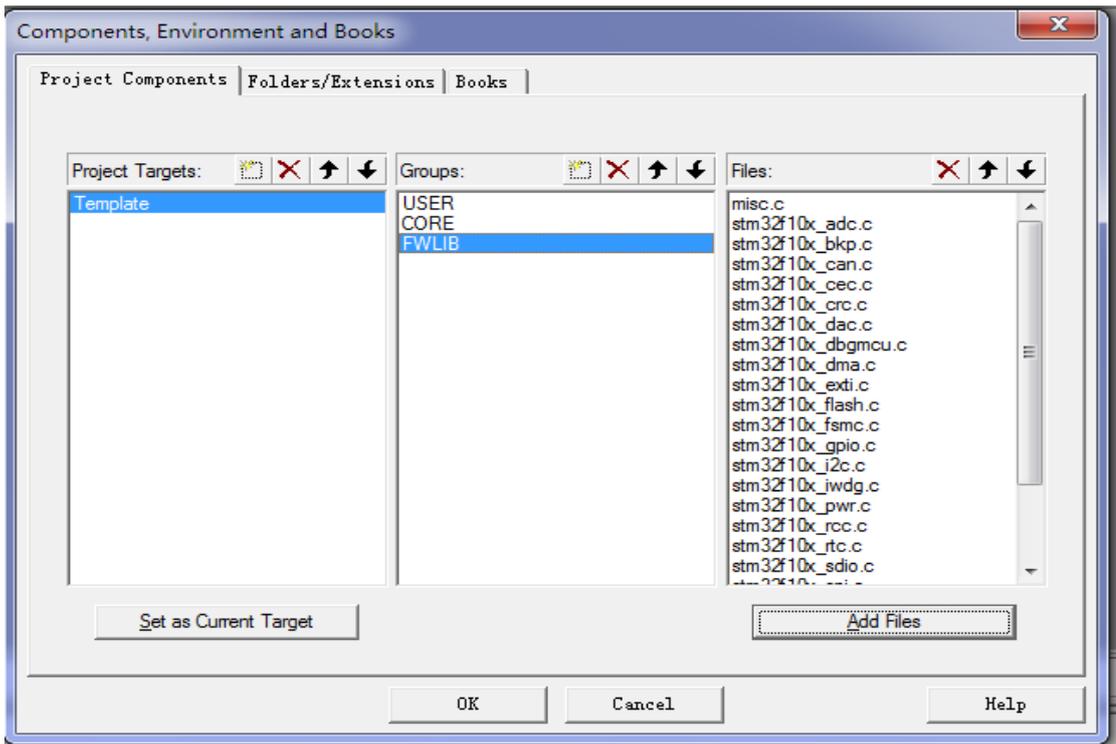
12. Project Targets 一栏，我们将 Target 名字修改为 Template,然后在 Groups 一栏删掉一个，建立三个 Groups: USER,CORE,FWLIB.点击 OK.可以看到我们的 Target 名字以及 Groups 情况。



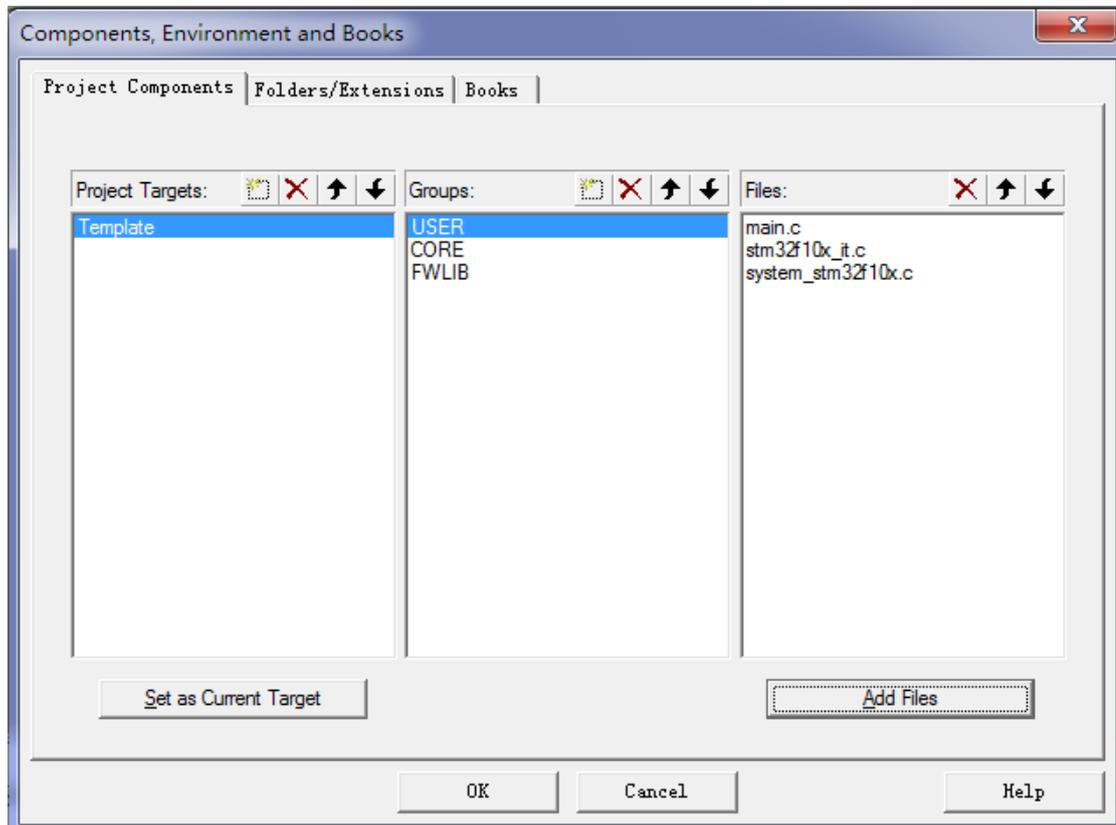


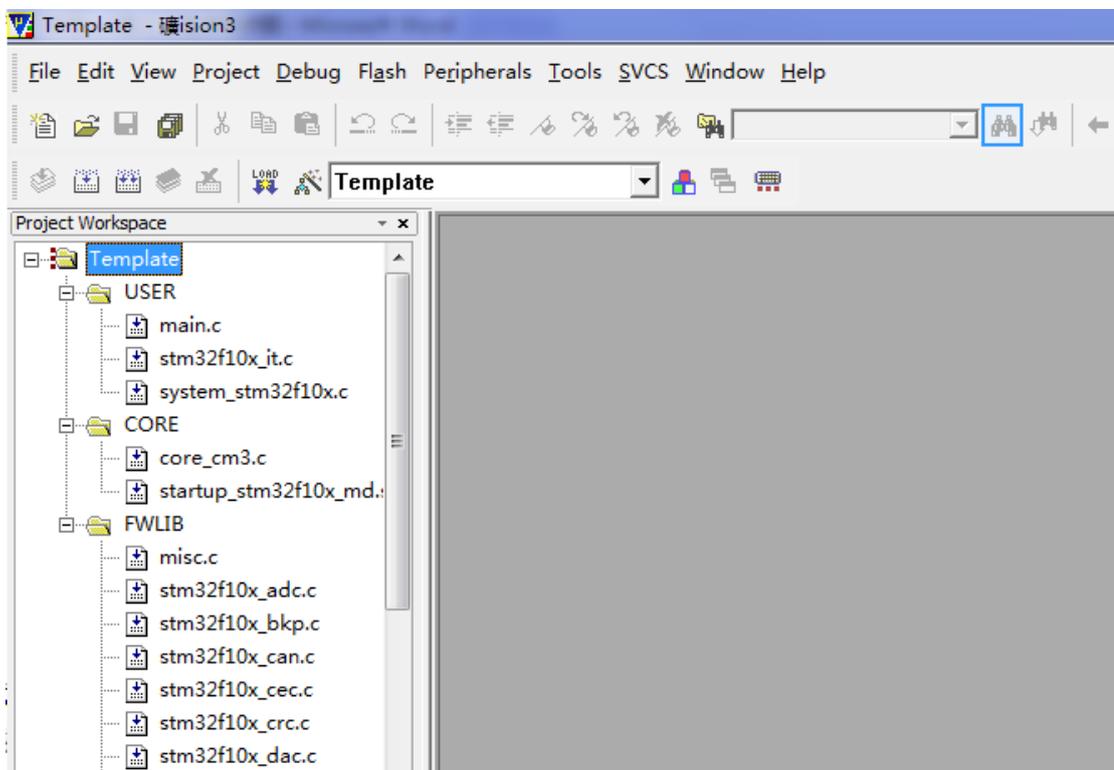
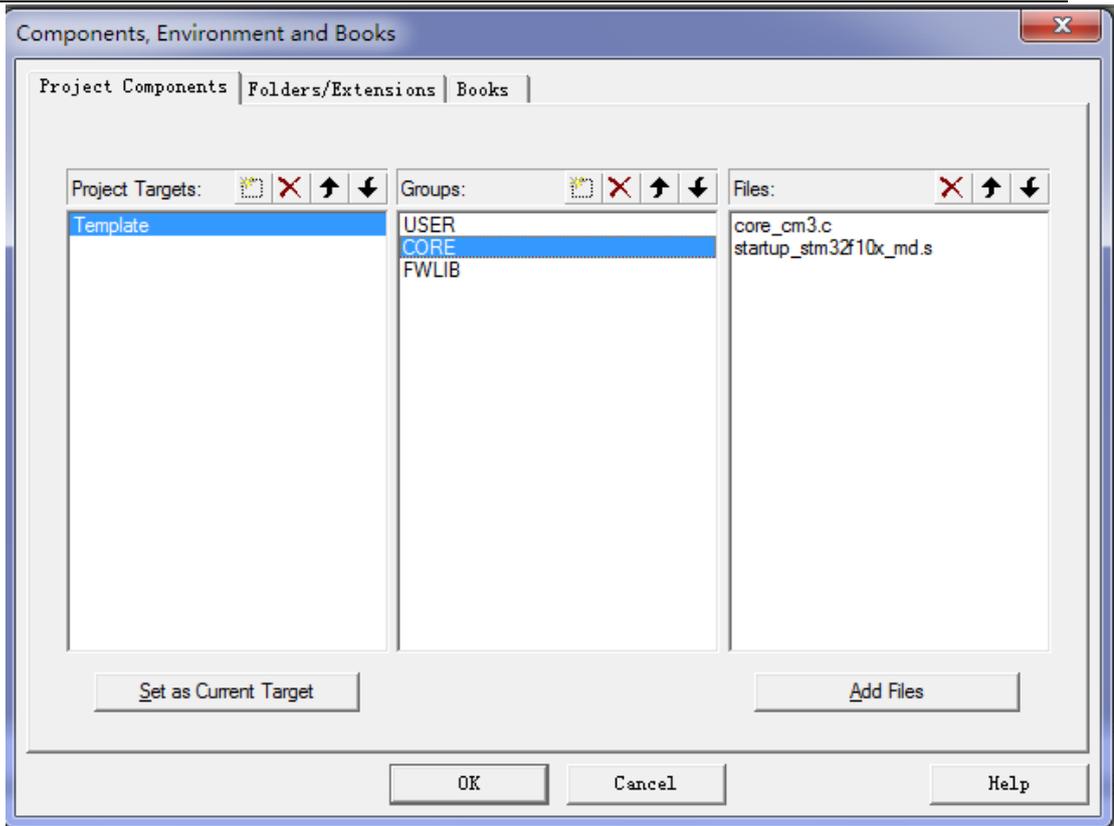
13.下面我们往 Group 里面添加我们需要的文件。我们按照步骤 12 的方法， 右键点击点击 Tempate， 选择选择 Manage Components.然后选择需要添加文件的 Group， 这里第一步我们选择 FWLIB， 然后点击右边的 Add Files,定位到我们刚才建立的目录 STM32F10x\_FWLib/src 下面， 将里面所有的文件选中(Ctrl+A)， 然后点击 Add， 然后 Close.可以看到 Files 列表下面包含我们添加的文件。

这里需要说明一下，对于我们写代码，如果我们只用到了其中的某个外设，我们就可以不用添加没有用到的外设的库文件。例如我只用 GPIO， 我可以只用添加 stm32f10x\_gpio.c 而其他的可以不用添加。这里我们全部添加进来是为了后面方便，不用每次添加，当然这样的坏处是工程太大，编译起来速度慢，用户可以自行选择。

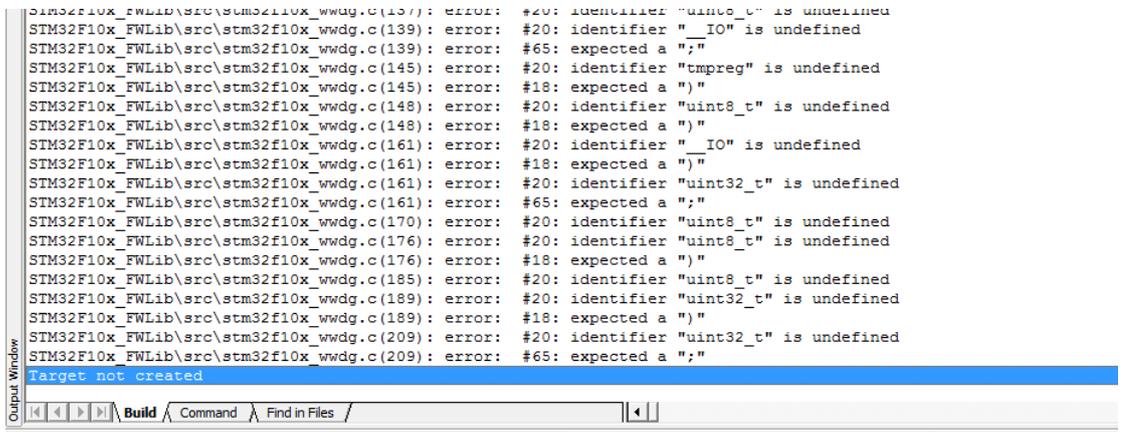
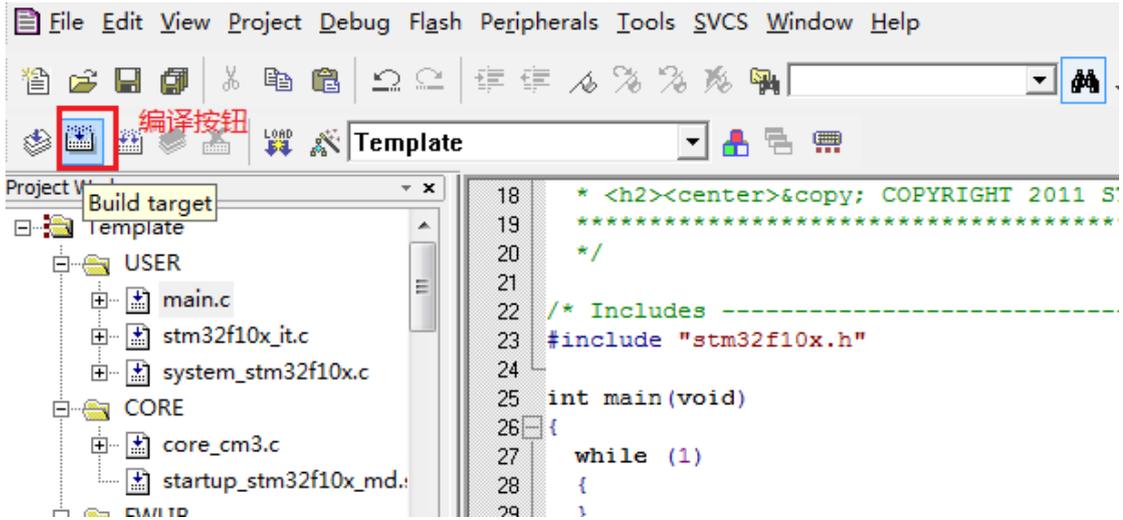


14.用同样的方法，将 Groups 定位到 CORE 和 USER 下面，添加需要的文件。这里我们的 CORE 下面需要添加的文件为 core\_cm3.c, startup\_stm32f10x\_md.s, USER 目录下面需要添加的文件为 main.c, stm32f10x\_it.c, system\_stm32f10x.c。这样我们需要添加的文件已经添加到我们的工程中去了，最后点击 OK，回到工程主界面。

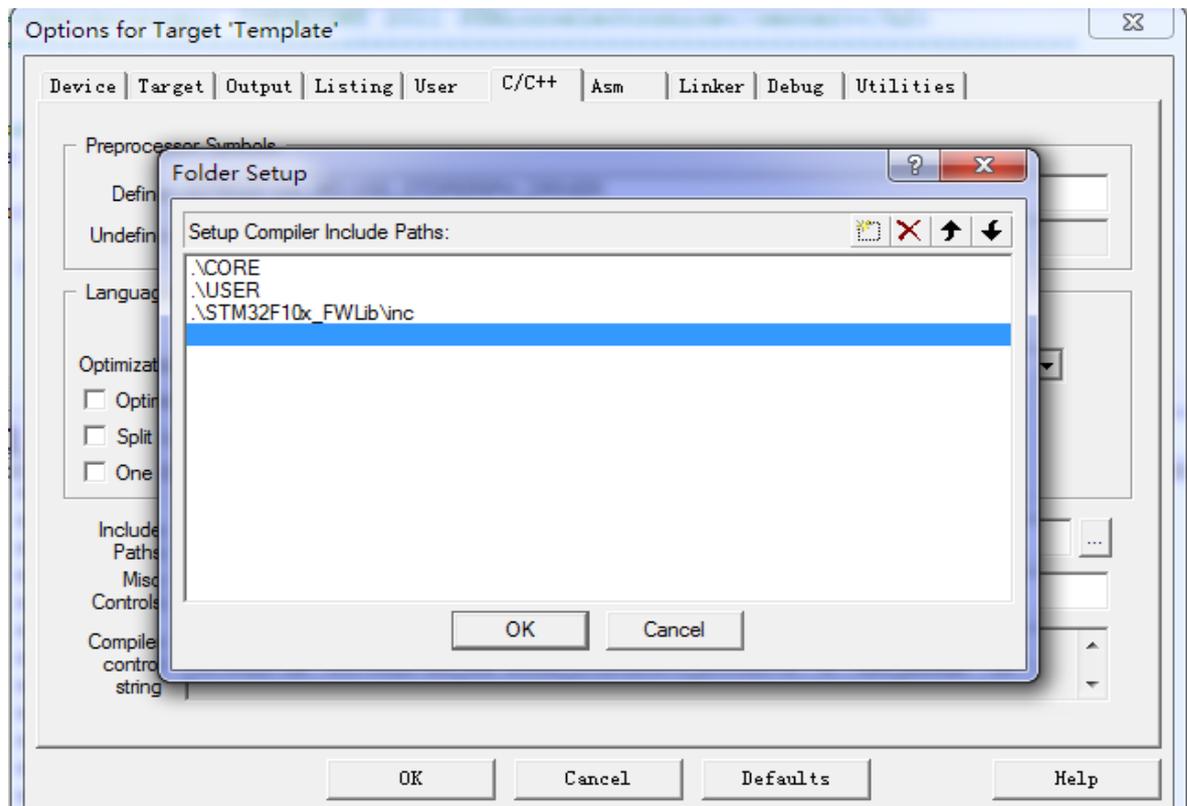
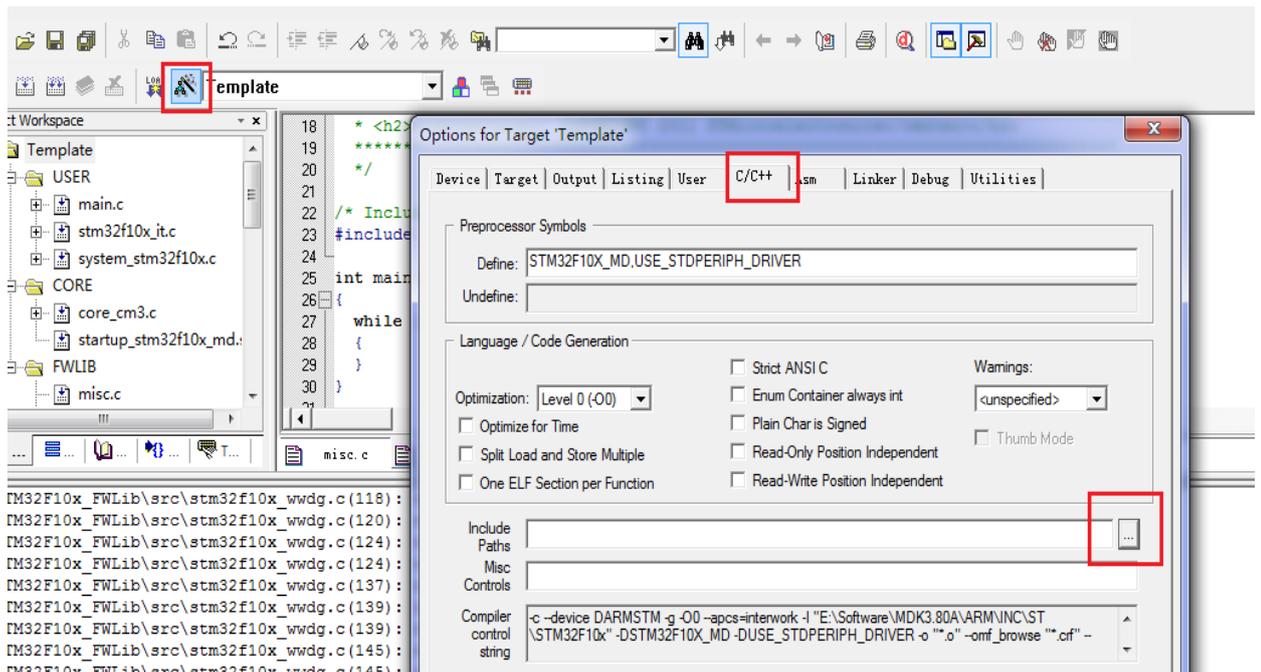




15.下面我们点击编译按钮  编译工程，可以看到很多报错，因为找不到库文件。



16.下面我们要告诉 MDK，在哪些路径之下搜索相应的文件。回到工程主菜单，点击魔术棒，出来一个菜单，然后点击 c/c++选项.然后点击 Include Paths 右边的按钮。弹出一个添加 path 的对话框，然后将图上面的 3 个目录添加进去。记住，keil 只会在一级目录查找，所以如果你的目录下面还有子目录，记得 path 一定要定位到最后一级子目录。然后点击 OK.

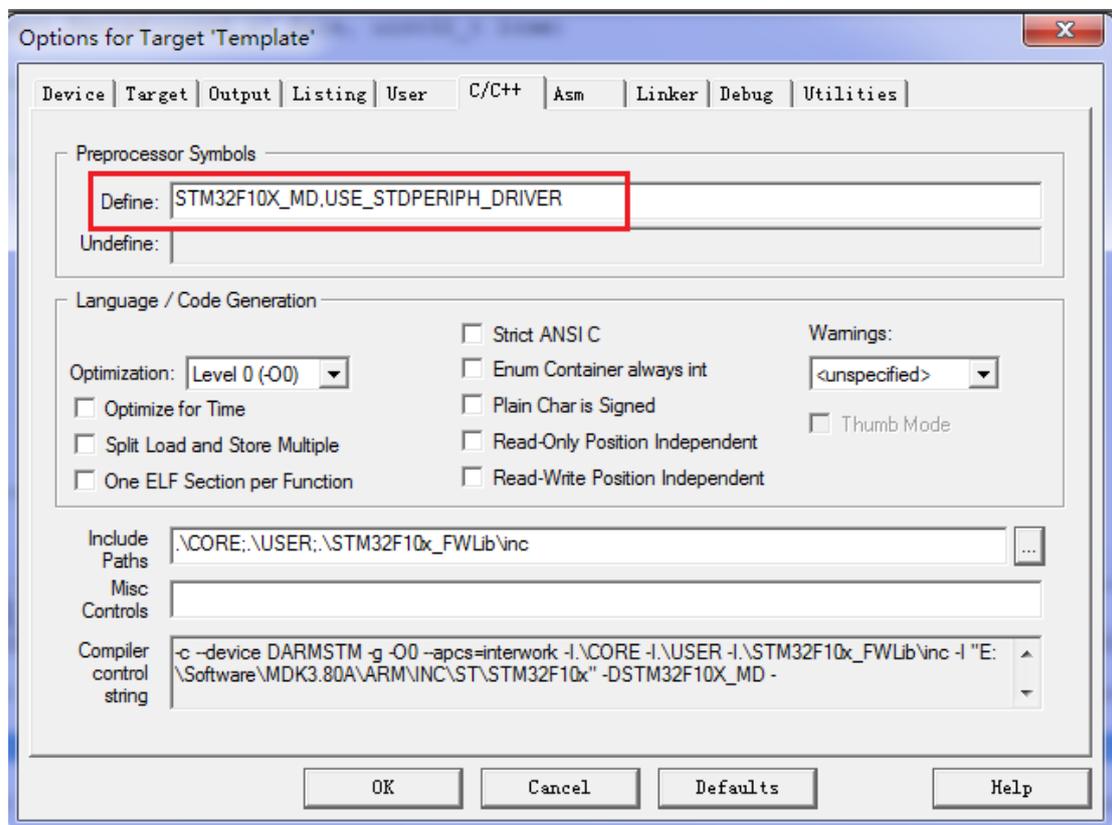


17.接下来，我们再来编译工程，可以看到又报了很多同样的错误。为什么呢？  
我们可以双击错误，然后会自动定位到文件 stm32f10x.h 中出错的地方，可以看到代码：



```
#if !defined (STM32F10X_LD) && !defined (STM32F10X_LD_VL) && !defined (STM32F10X_MD) && !defined (STM32F10X_MD_VL) && !defined (STM32F10X_HD) && !defined (STM32F10X_HD_VL) && !defined (STM32F10X_XL) && !defined (STM32F10X_CL)
#error "Please select first the target STM32F10x device used in your application (in stm32f10x.h file)"
#endif
```

这是因为 3.5 版本的库函数在配置和选择外设的时候通过宏定义来选择的，所以我们需要配置一个全局的宏定义变量。按照步骤 16，定位到 c/c++ 界面，然后 copy “STM32F10X\_MD,USE\_STDPERIPH\_DRIVER”到 Define 里面。这里解释一下，如果你用的是大容量那么 STM32F10X\_MD 修改为 STM32F10X\_HD，小容量修改为 STM32F10X\_LD. 然后点击 OK。



18.这次在编译之前，我们记得打开工程 USUR 下面的 main.c，复制下面代码到 main.c 覆盖已有代码，然后进行编译。（记得在代码的最后面加上一个回车，否则会有警告）

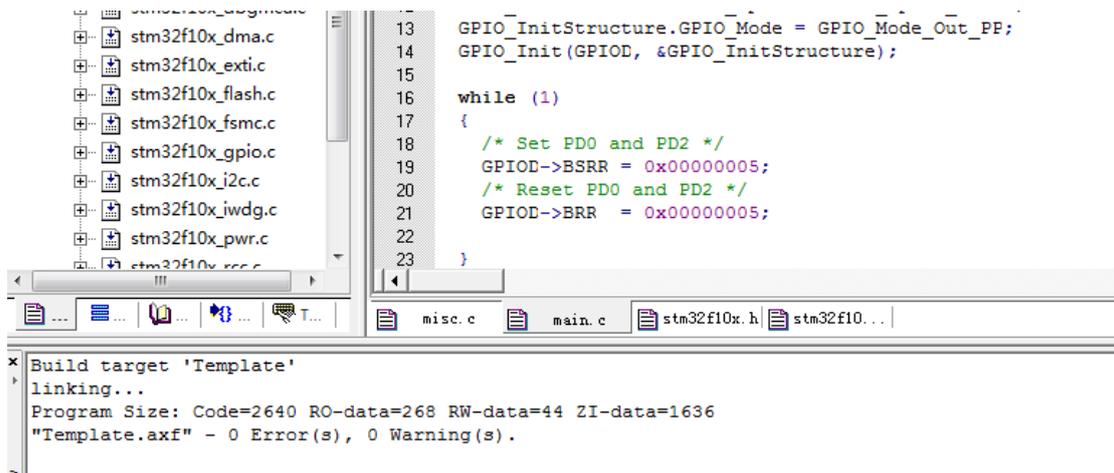
```
#include "stm32f10x.h"
GPIO_InitTypeDef GPIO_InitStructure;
int main(void)
{
    SystemInit();
    RCC_APB2PeriphClockCmd(RCC_APB2Periph_GPIOA, ENABLE);
    GPIO_InitStructure.GPIO_Pin = GPIO_Pin_0 | GPIO_Pin_2;
    GPIO_InitStructure.GPIO_Speed = GPIO_Speed_50MHz;
```

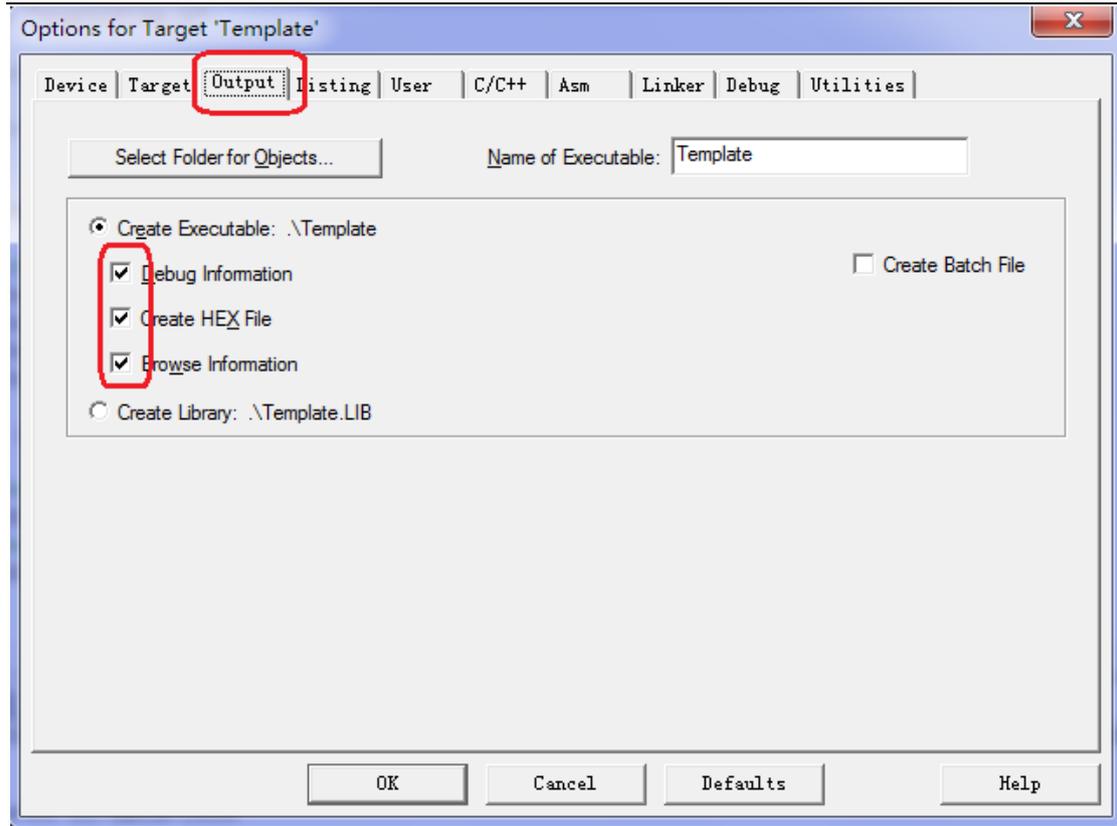


```
GPIO_InitStructure.GPIO_Mode = GPIO_Mode_Out_PP;
GPIO_Init(GPIOD, &GPIO_InitStructure);
while (1)
{
    /* Set PD0 and PD2 */
    GPIOD->BSRR = 0x00000005;
    /* Reset PD0 and PD2 */
    GPIOD->BRR = 0x00000005;
}
}
#endif USE_FULL_ASSERT

void assert_failed(uint8_t* file, uint32_t line)
{
    while (1)
    {
    }
}
#endif
```

19.这次编译可以看出，已经成功了。这样一个工程模版建立完毕。下面还需要配置，让编译之后能够生成 hex 文件。同样点击魔术棒，进入配置菜单，选择 Output。然后勾上下三个选项。其中 Create HEX file 是编译生成 hex 文件，Browser Information 是可以查看变量和函数定义，这里我们不做过多解释，在我们的不完全手册里面有讲解。





20.重新编译代码，可以看到生成了 hex 文件，这个文件我们用 mcuisp 下载到 mcu 即可。

ALIEN TEK//广州星翼电子科技有限公司  
开发板购买店铺: <http://eboard.taobao.com>  
技术支持论坛: [www.openedv.com](http://www.openedv.com)  
2012年7月22日