



《*iCore Stm32*教程》



iCoreStm32 教程

版本 V 0.1 日 期 6/19/2012



作者 : ---

深入交流 QQ 群 :

A: 204255896 (500 人超级群 , 满员)

B: 165201798 (500 人超级群 , 满员)

C: 215053598 (200 人高级群 , 满员)

D: 215054675 (200 人高级群)

E: 215055211 (200 人高级群)

F: 78538605 (500 人高级群)

G:158560047 (500 人高级群 , 满员)

<http://XiaomaGee.cnblogs.com>

<http://i-board.taobao.com>

<http://www.heijin.org>



目 录

目 录	3
第一章 KEIL 软件开发	4
一、 KEIL MDK_ARM 软件安装	6
二、 KEIL MDK_ARM 软件安装破解	9
三、 新建工程	15
四、 程序下载	42

第一章 KEIL 软件开发

KEIL 软件开发

本章介绍了 Keil MDK_ARM 软件的安装、破解、工程建立及下载。

本章分为以下几个部分：

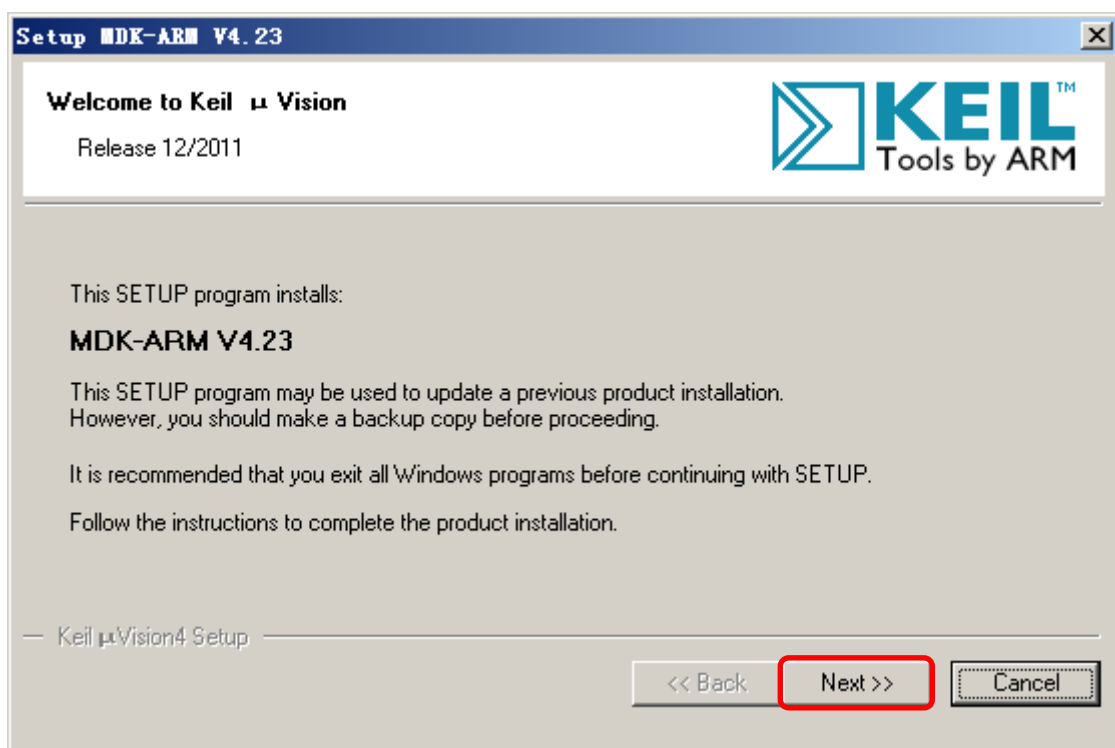
- 一、Keil MDK_ARM 软件安装
- 二、Keil MDK_ARM 软件安装破解
- 三、新建工程
- 四、程序下载

一、 Keil MDK_ARM 简介

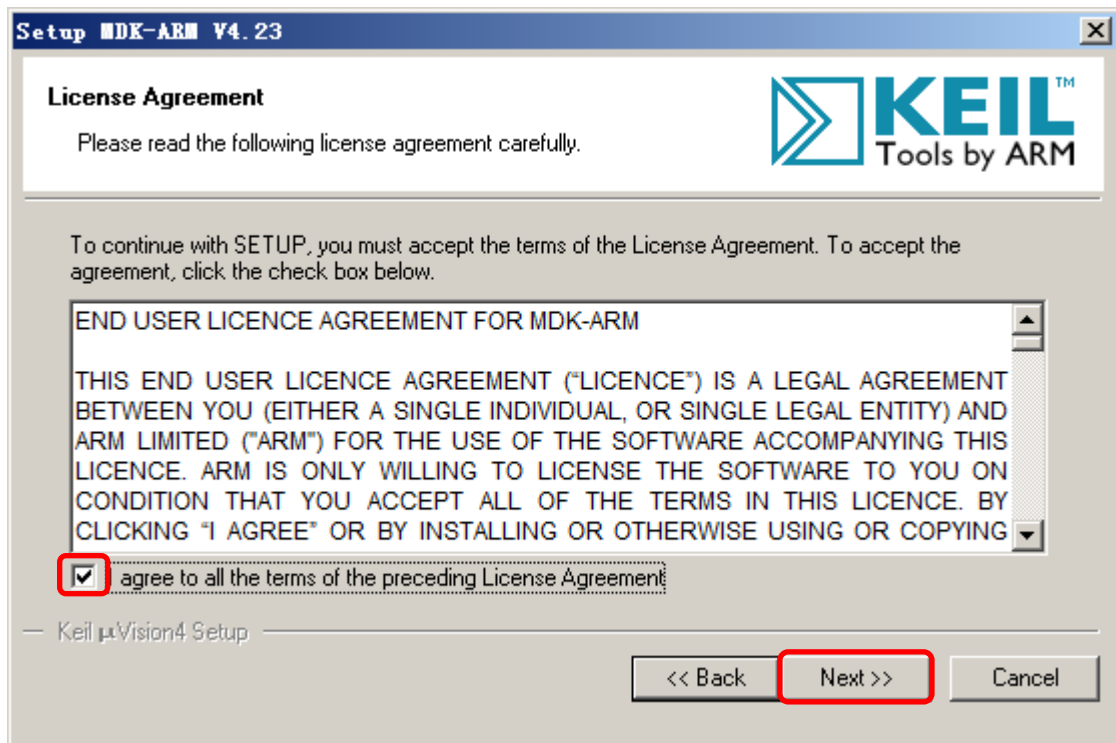
RealView MDK 开发套件源自德国 Keil 公司，是 ARM 公司目前最新推出的针对各种嵌入式处理器的软件开发工具。RealView MDK 集成了业内最领先的技术，包括 μ Vision4 集成开发环境与 RealView 编译器。RealView MDK 的设备模拟器可以仿真整个目标硬件，包括快速指令集仿真、外部信号和 I/O 仿真、中断过程仿真、片内所有外围设备仿真等。在无硬件的情况下即可开始软件开发和调试，使软硬件开发同步进行，大大缩短开发周期，提高开发效率。

二、 Keil MDK_ARM 软件安装

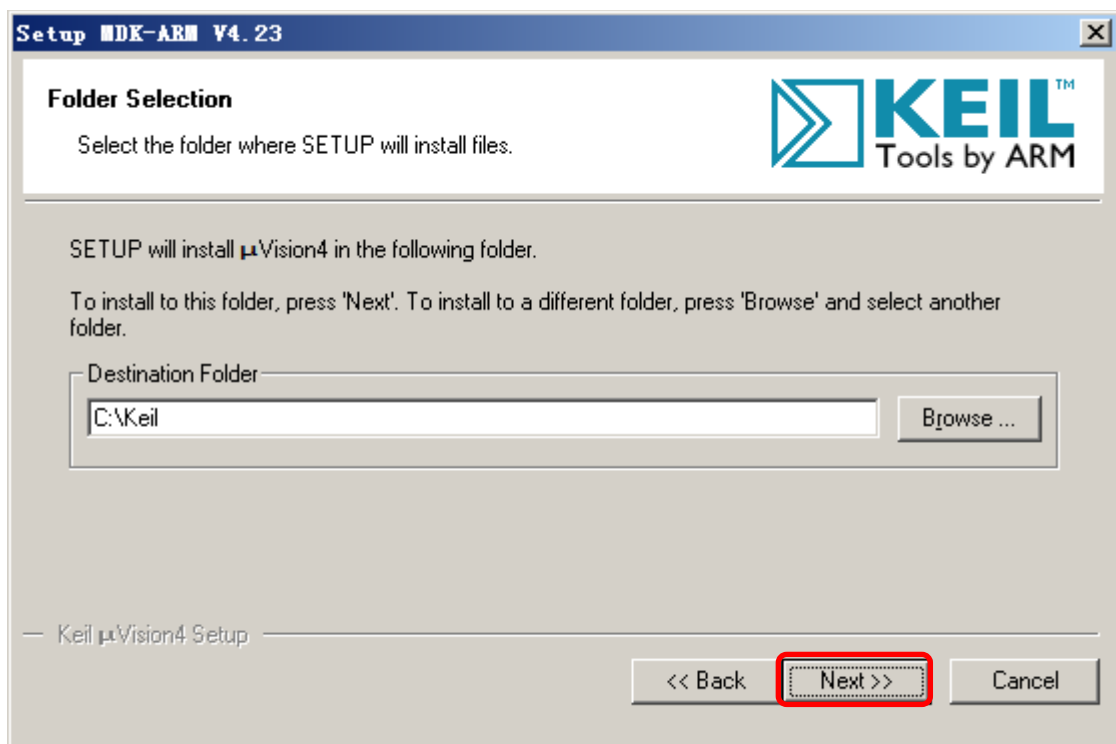
- 1、打开 iCore (A) 光盘，在 KEIL MDK_ARM 文件夹中，双击 “mdk423.exe” 可执行安装文件，出现如下界面：



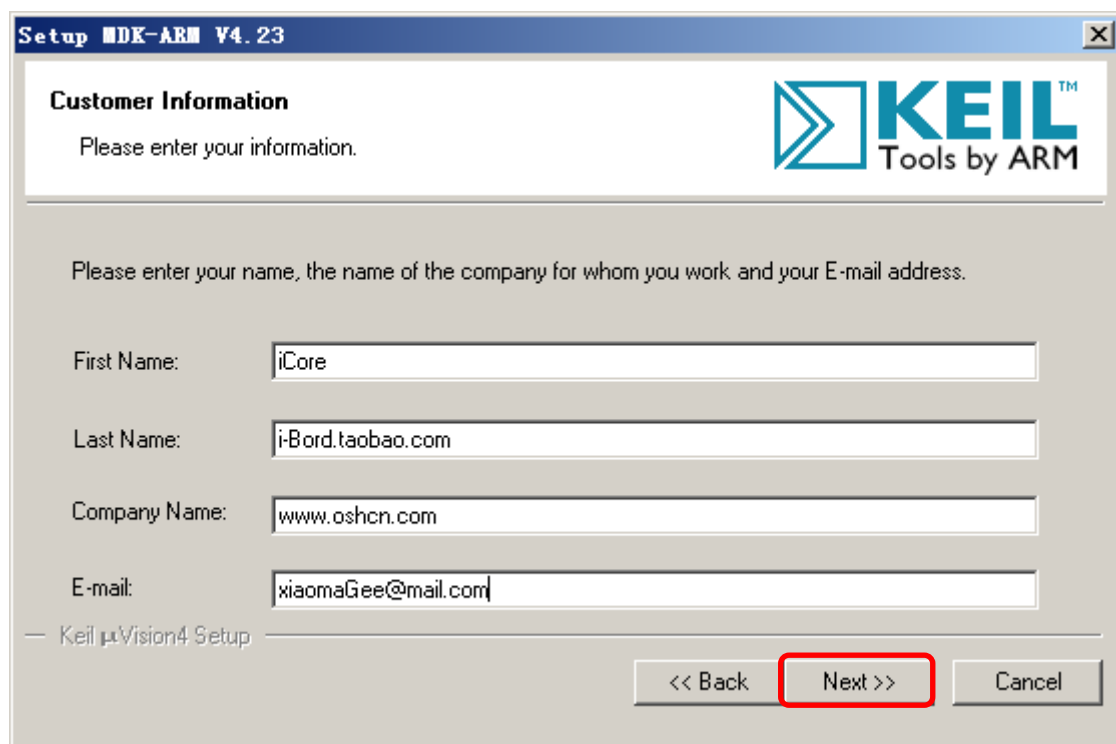
- 2、点击 “Next”，下一步：



3、点击 “I agree.....” ，再点击 “Next” ，下一步：

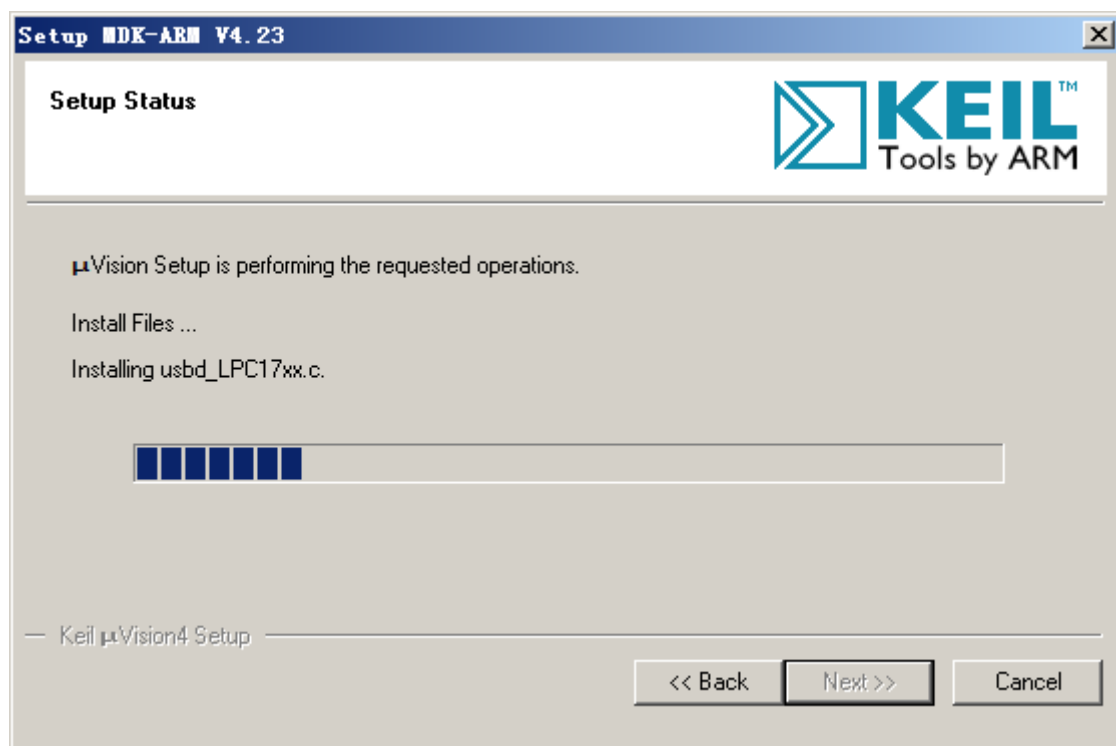


4、在 “Browse.....” 中选择安装路径，然后点击 “Next” ，下一步：

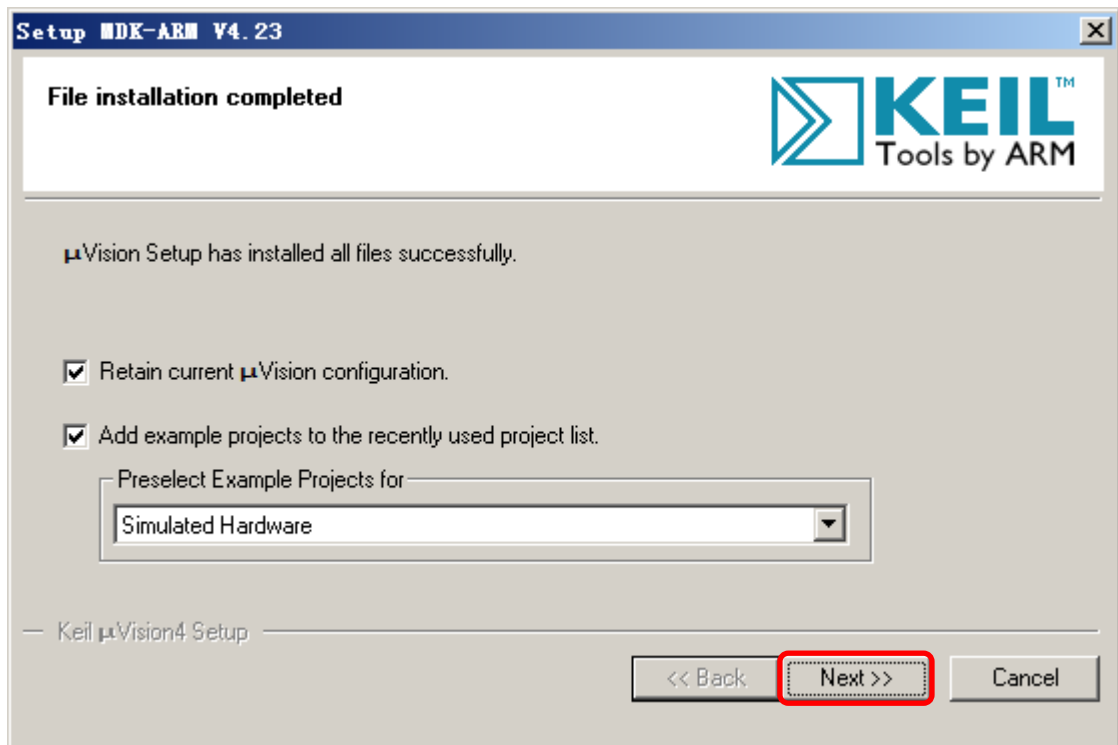


在窗口内的“First Name”，“Last Name”，“Company Name”，“E_mail”中填入相应信息。

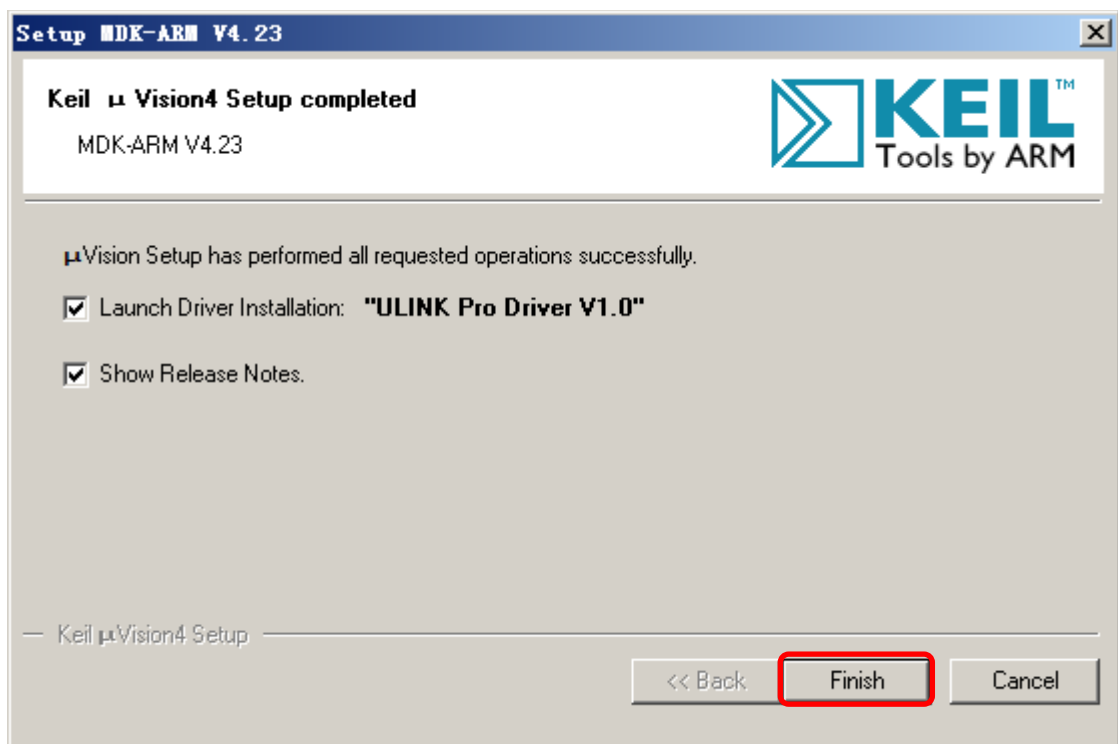
5、点击“Next”，下一步，出现如下界面：



稍等片刻，即出现如下界面：



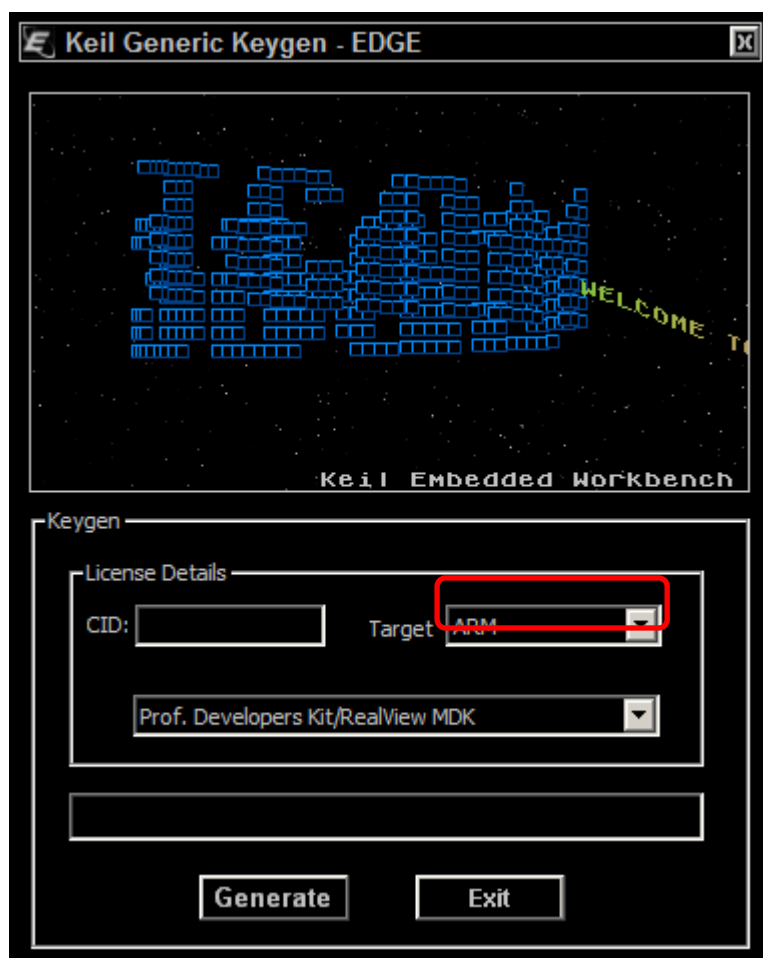
6、点击“Next”，下一步，出现如下界面：




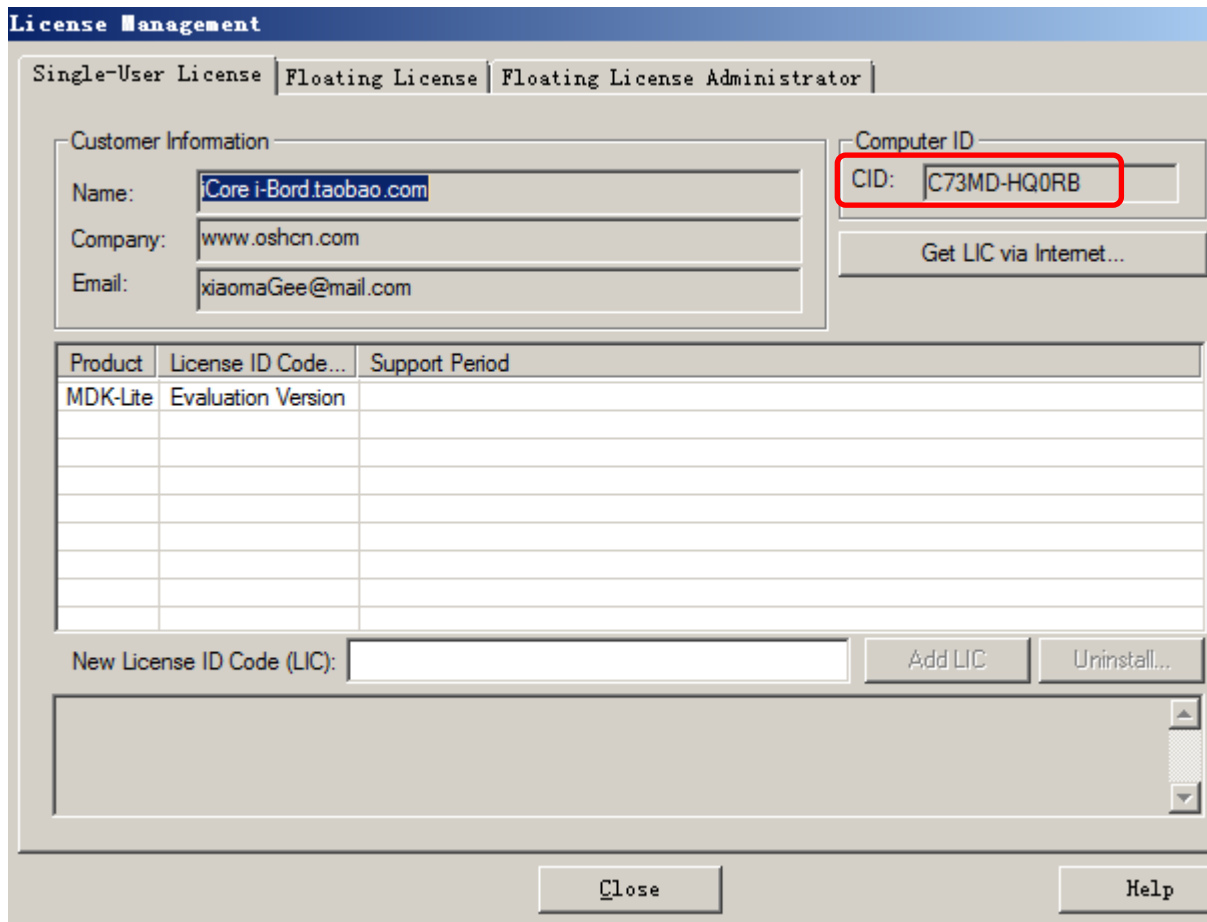
7、点击“Finish”即完成此应用程序的安装。

三、 Keil MDK_ARM 软件安装破解

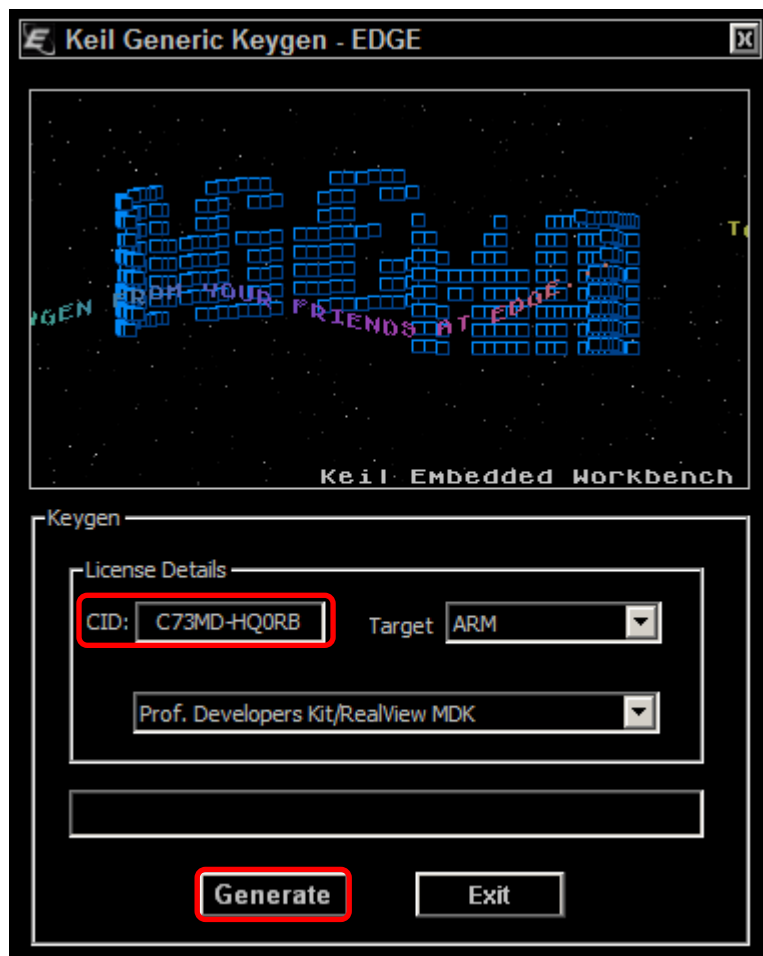
- 1、双击(若为 Win7 系统则右击选择“以管理员身份运行”)“KEIL_Lic.exe”的应用程序，Target 处选择 ARM，出现如下窗口：



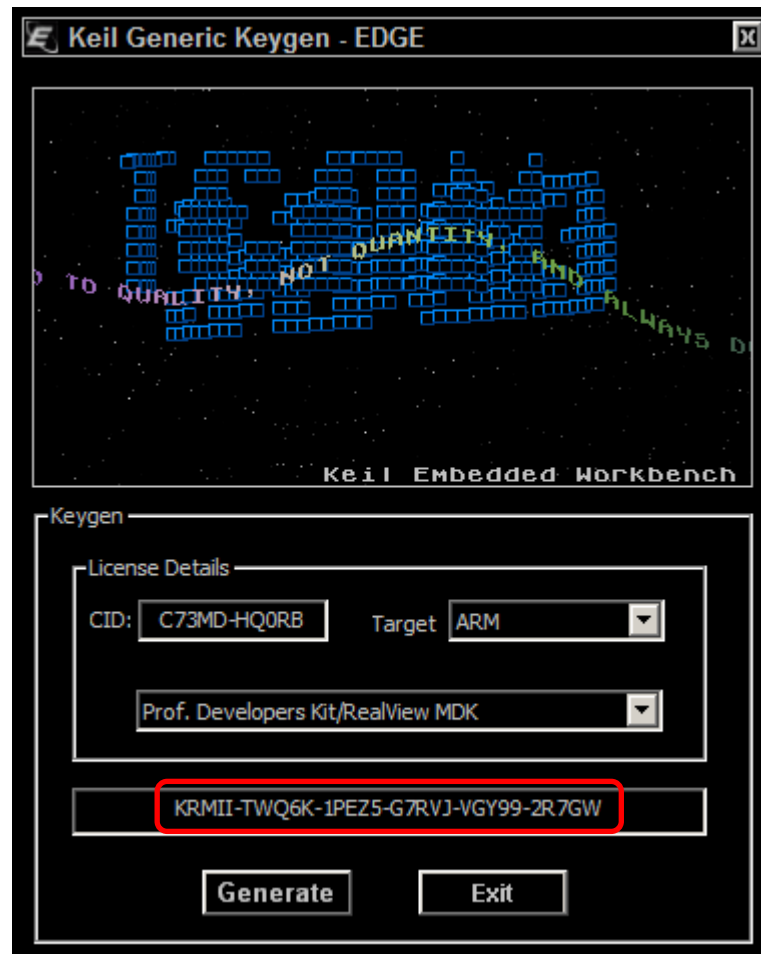
- 2、在桌面双击(若为 Win7 系统则右击选择“以管理员身份运行”)图标 ，打开软件，点击“File>License Management”，出现如下窗口：



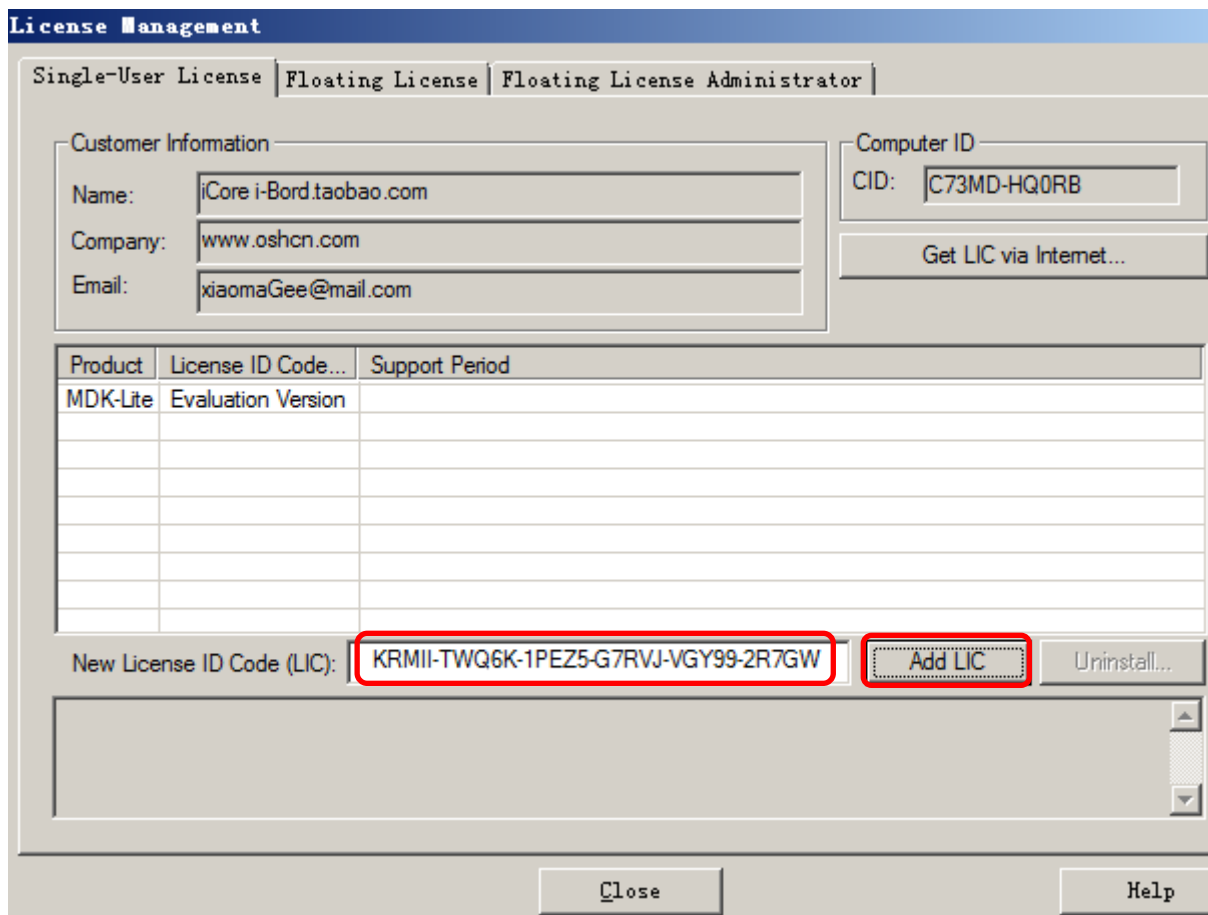
- 3、将右上角 CID 框内的内容 C73MD-HQ0RB 复制到如下窗口的 CID 框内，出现如下窗口：



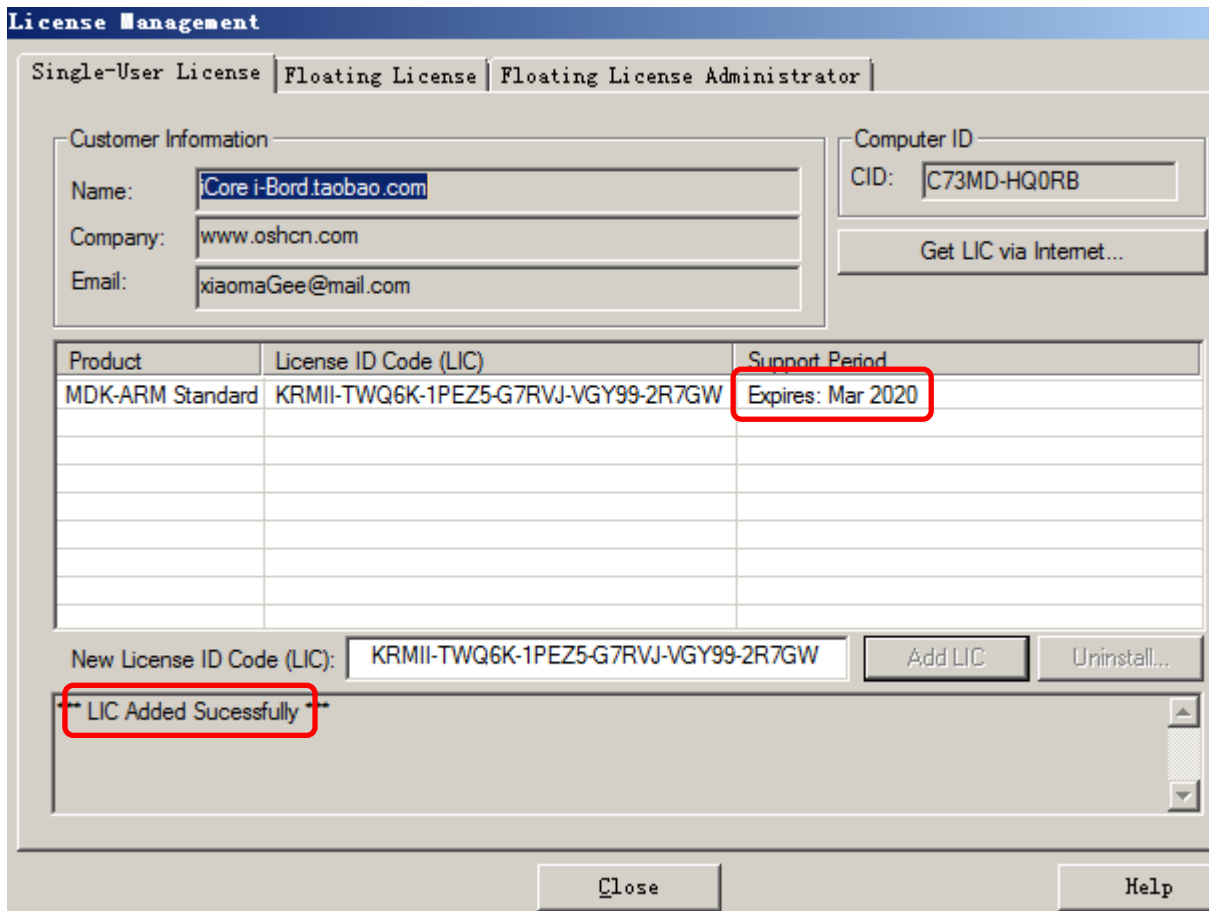
4、点击 “Generate”，窗口的红框中生成如下内容：



- 5、将上图中红色框中内容复制到“License Management”窗口内正下方“New License ID Code(LIC)”的框内，出现如下窗口：

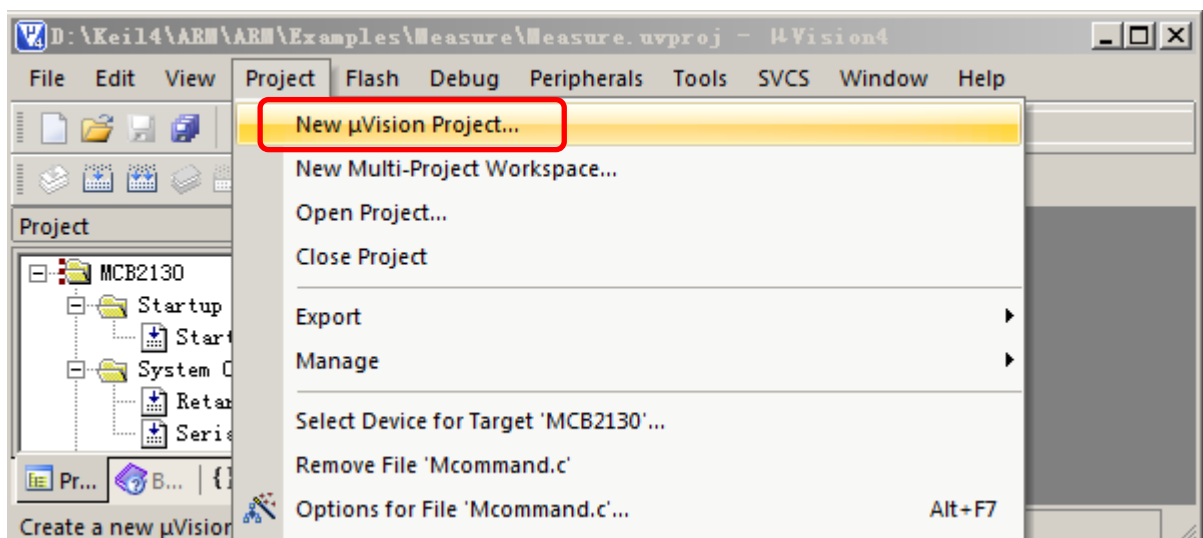


- 6、点击 “Add LIC” ，出现如下图所示的红框内的内容 (Expires:Apr2020) ， LIC Added Sucessfully。 最后点击 “Close” 即完成该软件的破解。



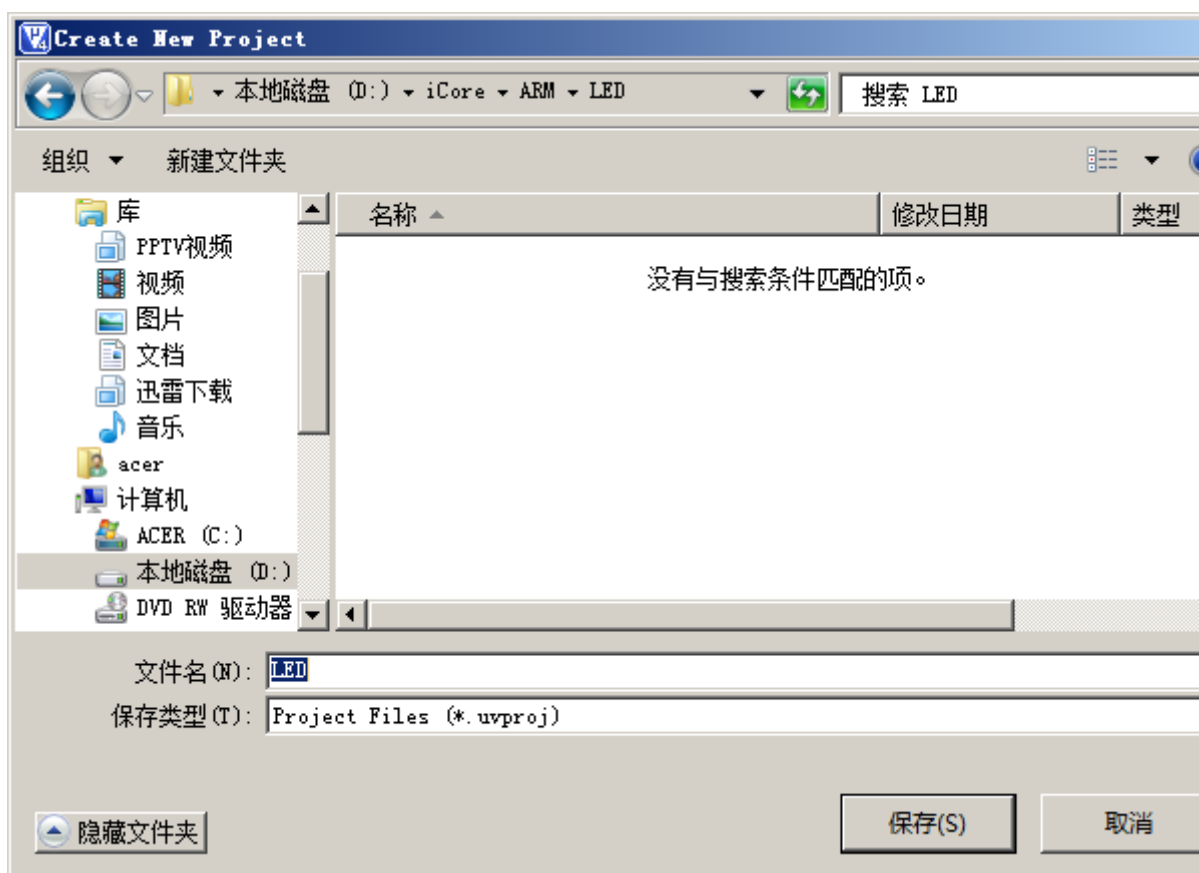
四、新建工程

- 1、首先，打开 Keil MDK_ARM 软件，执行 Keil MDK_ARM 软件的菜单“Project/New uVision Project...”开始建立新工程，如下图所示：



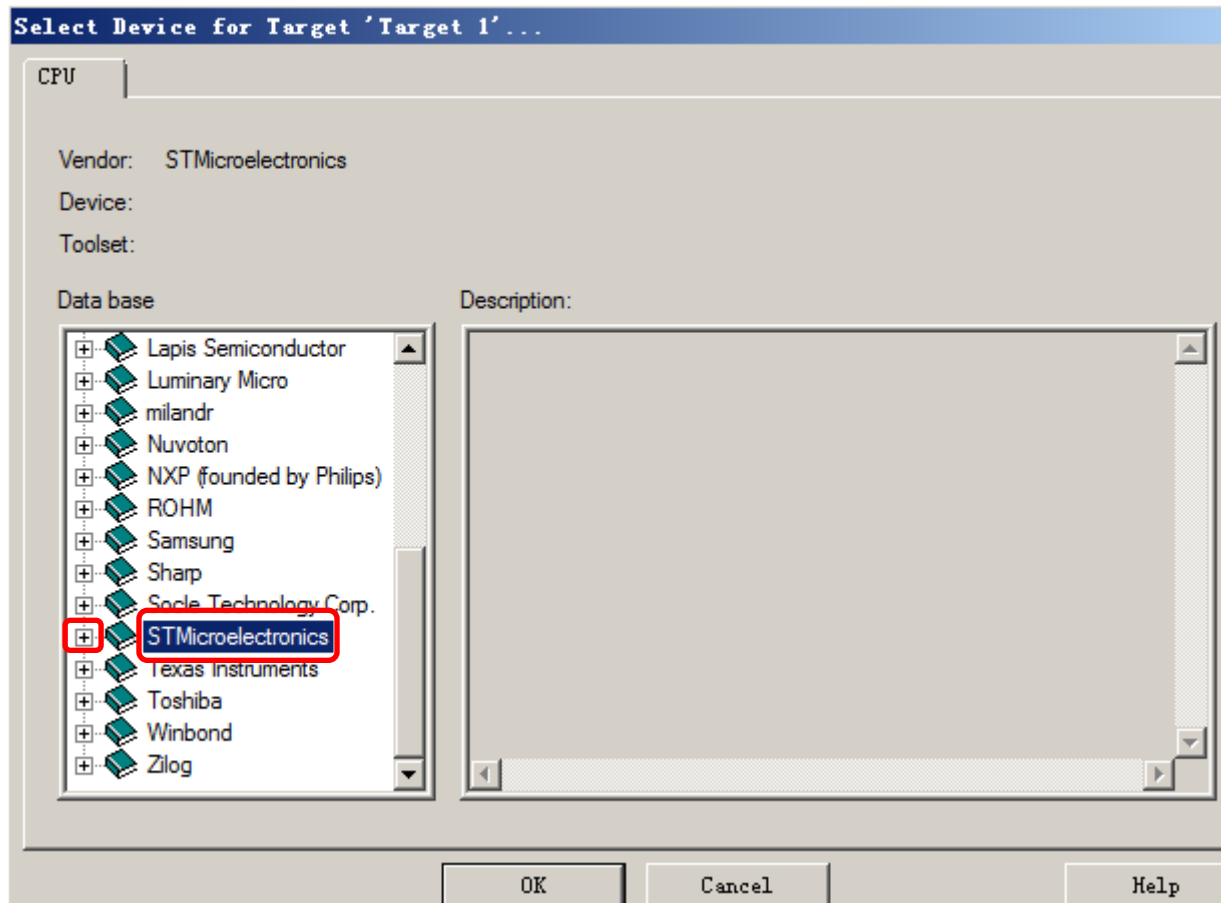
- 2、弹出如下一个名为“Create New Project”的窗口，先选择一个合适的文件夹

来存放工程文件，比如“D:/iCore/ARM/LED”，如下图所示：

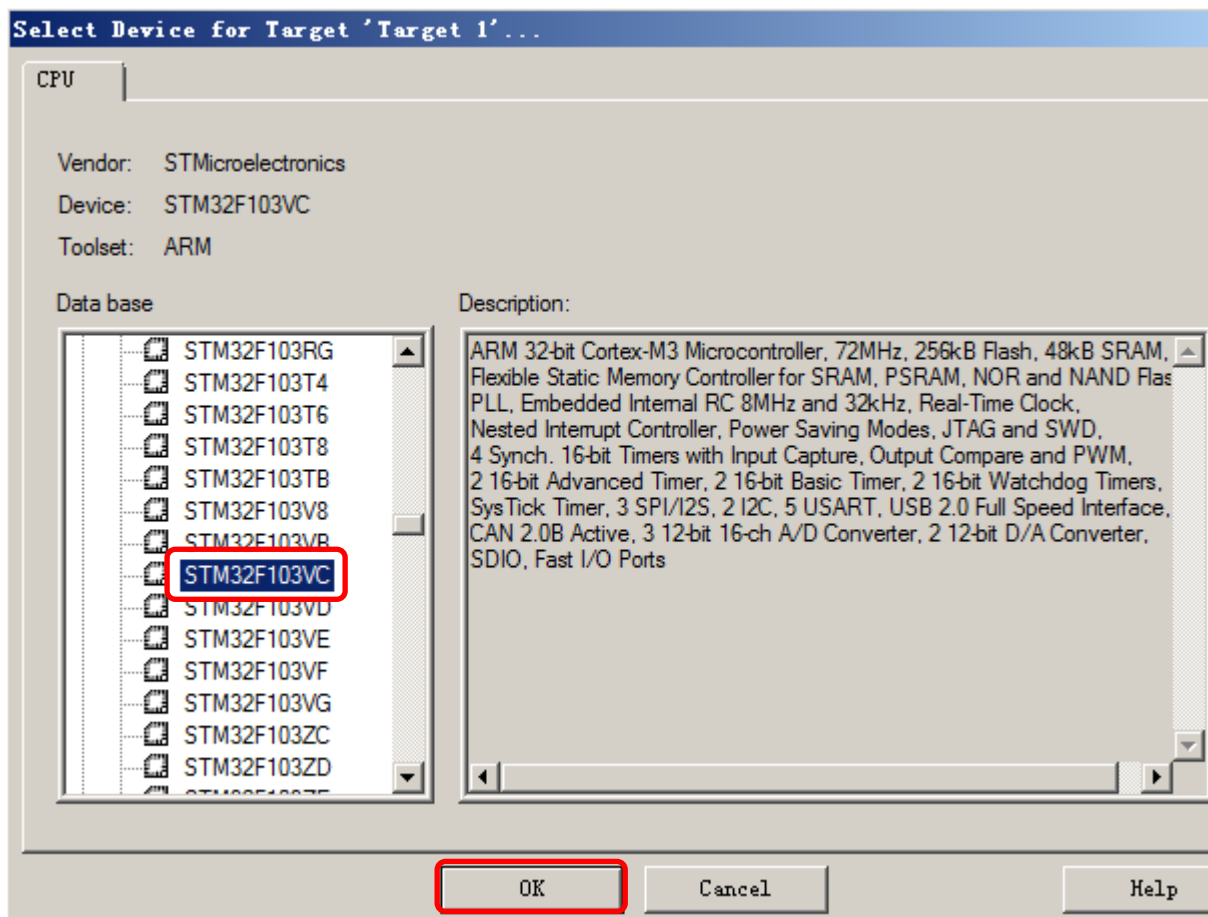


注意：路径中不能有汉字。

- 3、点击保存，Keil C 软件会弹出单片机型号选择窗口，此处选择“STMicroelectronics”



4、点击 STMicroelectronics 左边的 “+”，会弹出如下窗口

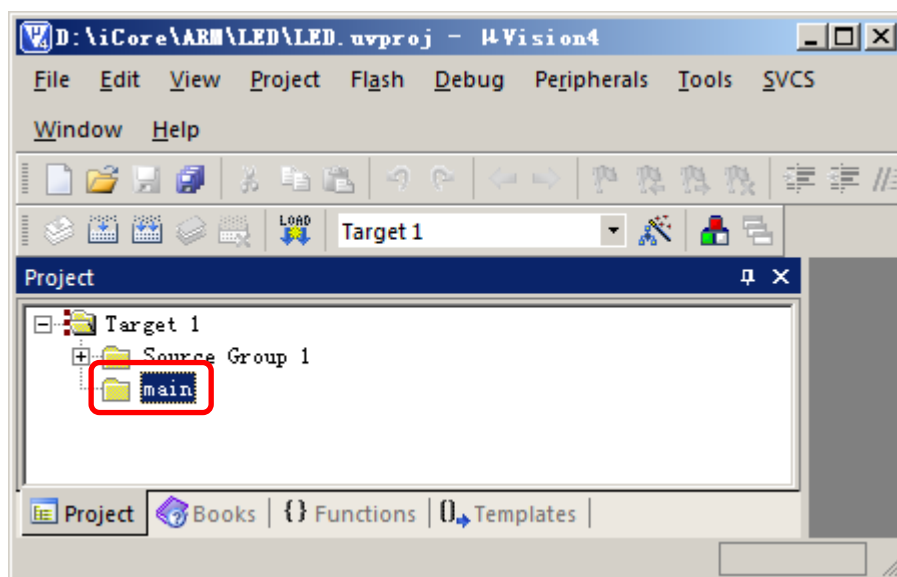
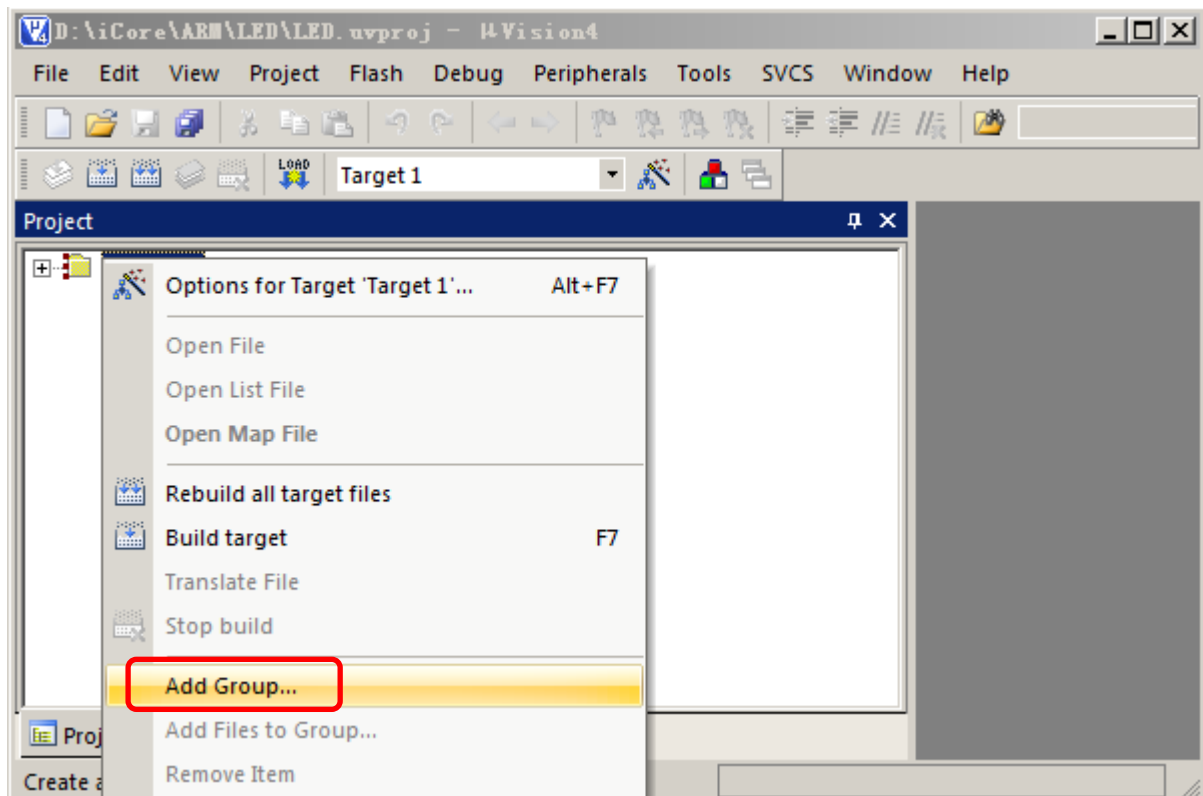


5、选择 “STM32F103VC” ，点击 “OK”

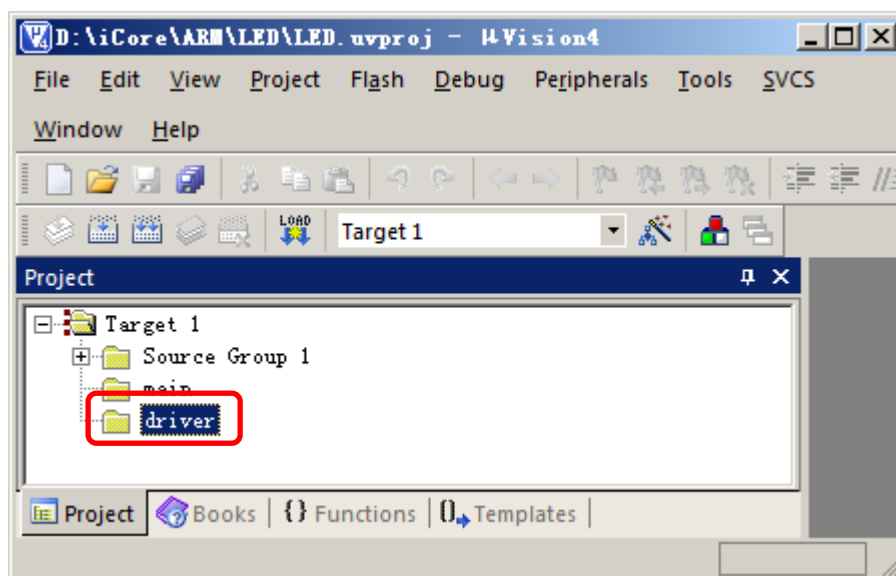


6、点击“是”，至此一个空白的 Keil C 工程就建好了。

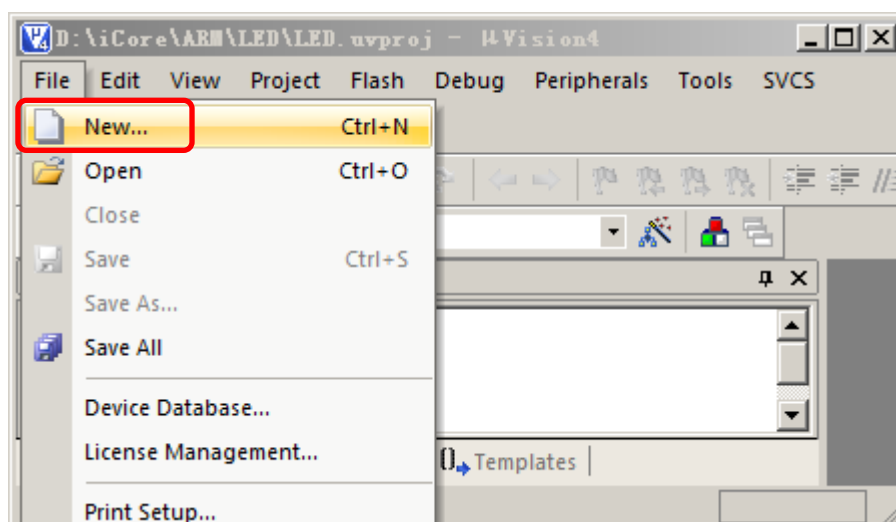
7、之后在新建工程中新建文件夹。在 Keil C 软件左边项目工作窗口 “Target” 上右击鼠标，如下图所示，单击 “Add Group”，新建一个文件夹，命名为 “main”



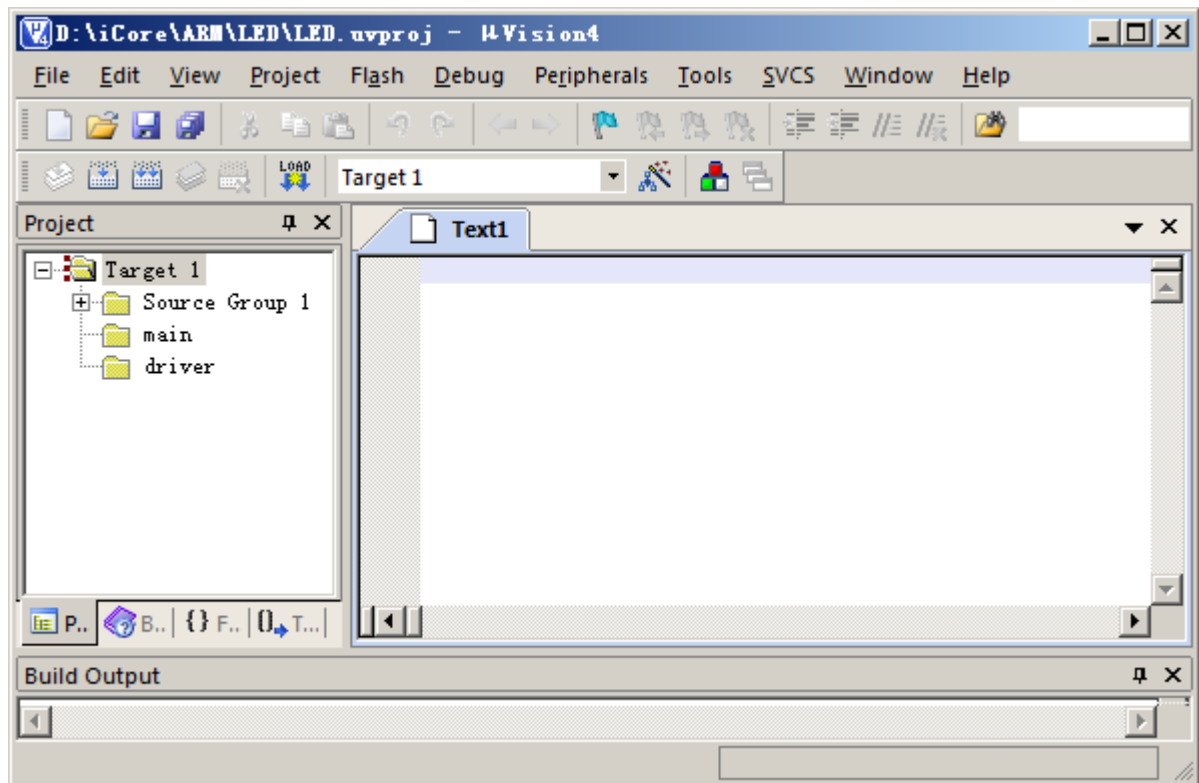
同上，再建一个文件夹，命名为“ driver” ，如下图所示：



8、下面开始建立头文件（.h 文件），点击菜单“File/New”选项，如下图所示：



执行后会弹出一个空的 Text 文档，如下所示：



9、将下列程序代码输入到此空的 Text 文档中：

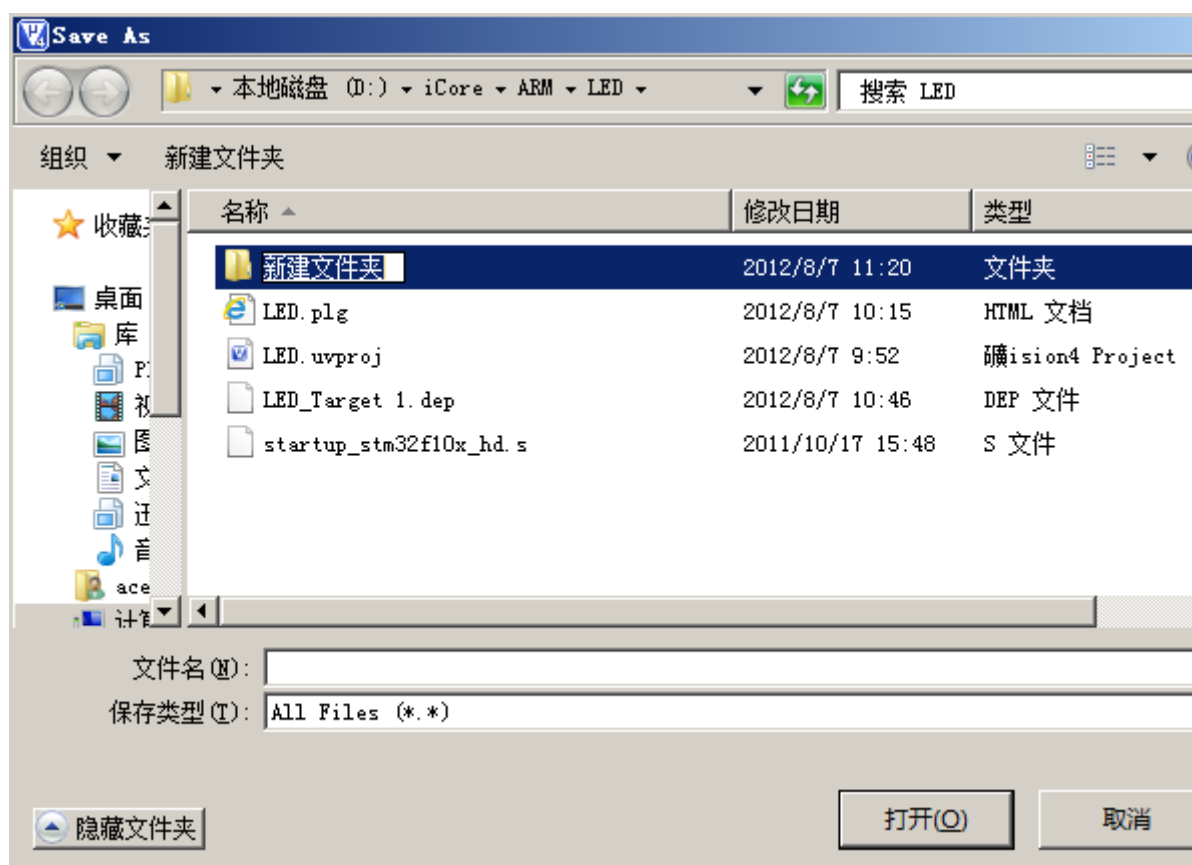
```

1  /*
2  * FILE           : hardware.h
3  * DESCRIPTION    : This file is iCore hardware file header.
4  * Author        : XiaomaGee@Gmail.com
5  * Copyright     :
6  *
7  * History
8  * -----
9  * Rev           : 0.00
10 * Date          : 03/05/2012
11 *
12 * create.
13 * -----
14 */
15
16 //-----define-----//
17
18 #ifndef __hardware_h__
19 #define __hardware_h__
20
21 //-----Include files-----//
22

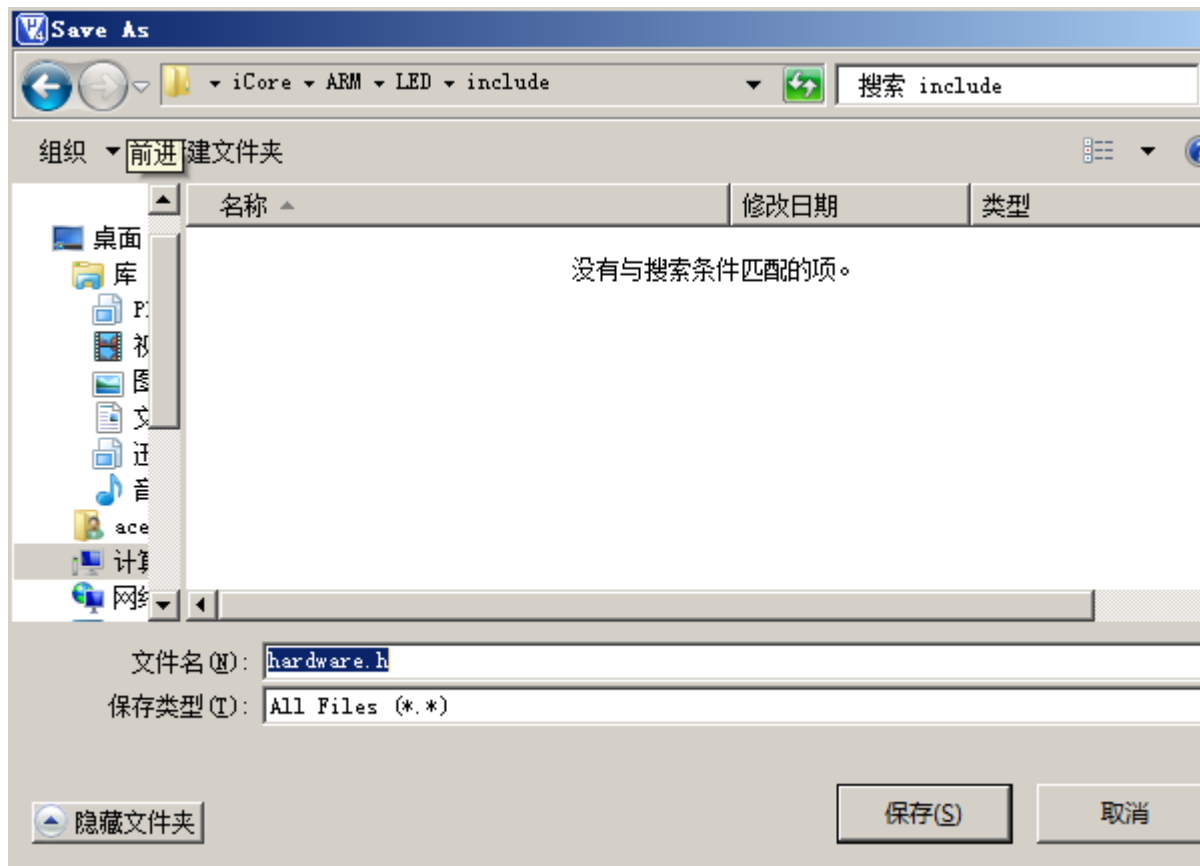
```

```
23 #include "..\include\stm32f10x_reg.h"
24
25 //----- Typedef -----//
26
27 #define LED_OUTPUT GPIOC->CRH.B.MODE10=0X3;\
28             GPIOC->CRH.B.CNF10=0
29
30 #define LED_ON      GPIOC->BSRR.B.SET10=1
31 #define LED_OFF     GPIOC->BSRR.B.CLRI0=1
32
33 #endif //__hardware_h__
34
```

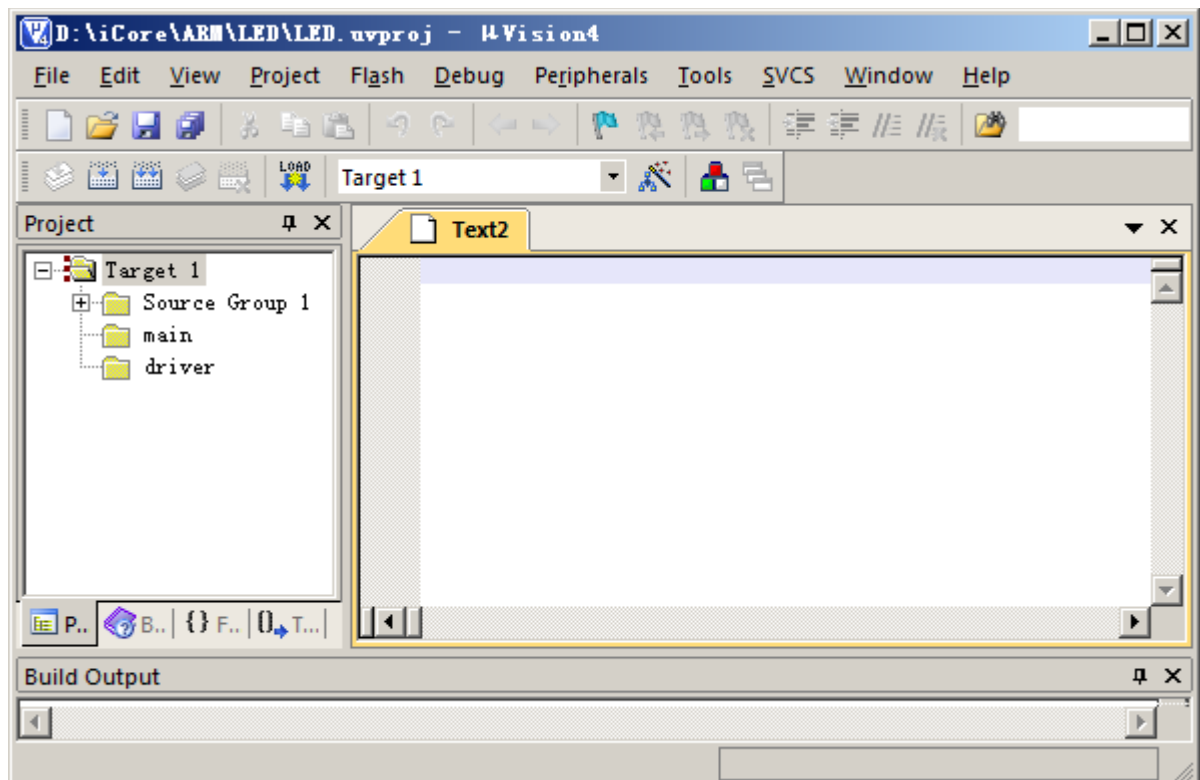
10、 点击保存 ,如下图所示 ,在 LED 文件夹下新建一个文件夹 ,命名为“include” :



11、 打开 include 文件夹 ,在文件名处输入 “hardware.h” ,如下图所示 ,然后点击保存 :



12、 下面建立原程序文件(.c 文件),点击菜单“File/New”选项,如下图所示:



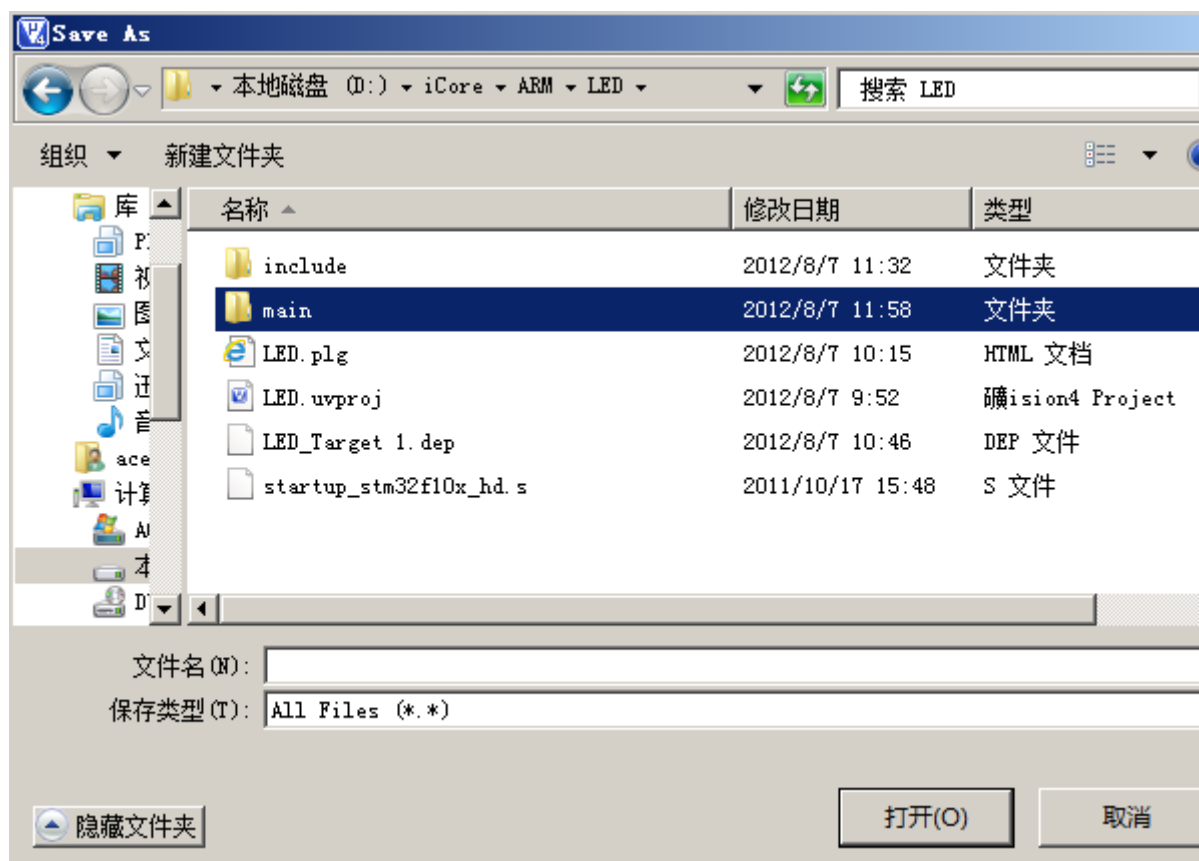
13、 将下列程序代码输入到此空的 Text2 文档中

```
1 /*
2  * FIL                      : main.c
3  * DESCRIPTION              : This file is main files.
4  * Author                   : XiaomaGee@Gmail.com
5  * Copyright                :
6  *
7  * History
8  * -----
9  * Rev                      : 0.00
10 * Date                     : 03/05/2012
11 *
12 * create.
13 * -----
14 */
15
16 //-----Include files-----//
17
18 #include "..\include\nvic.h"
19 #include "..\include\rcc.h"
20
21 #include "..\include\hardware.h"
22
23 #include <string.h>
24 #include <stdlib.h>
25
26 //-----Function-----//
27 /*
28  * Name                     : SystemInit
29  * Description              : ---
30  * Author                   : XiaomaGee.
31  *
32  * History
33  * -----
34  * Rev                      : 0.00
35  * Date                     : 03/05/2012
36  *
37  * create.
38  * -----
39  */
40
41 void SystemInit(void)
42 {
43 }
44 /*
```

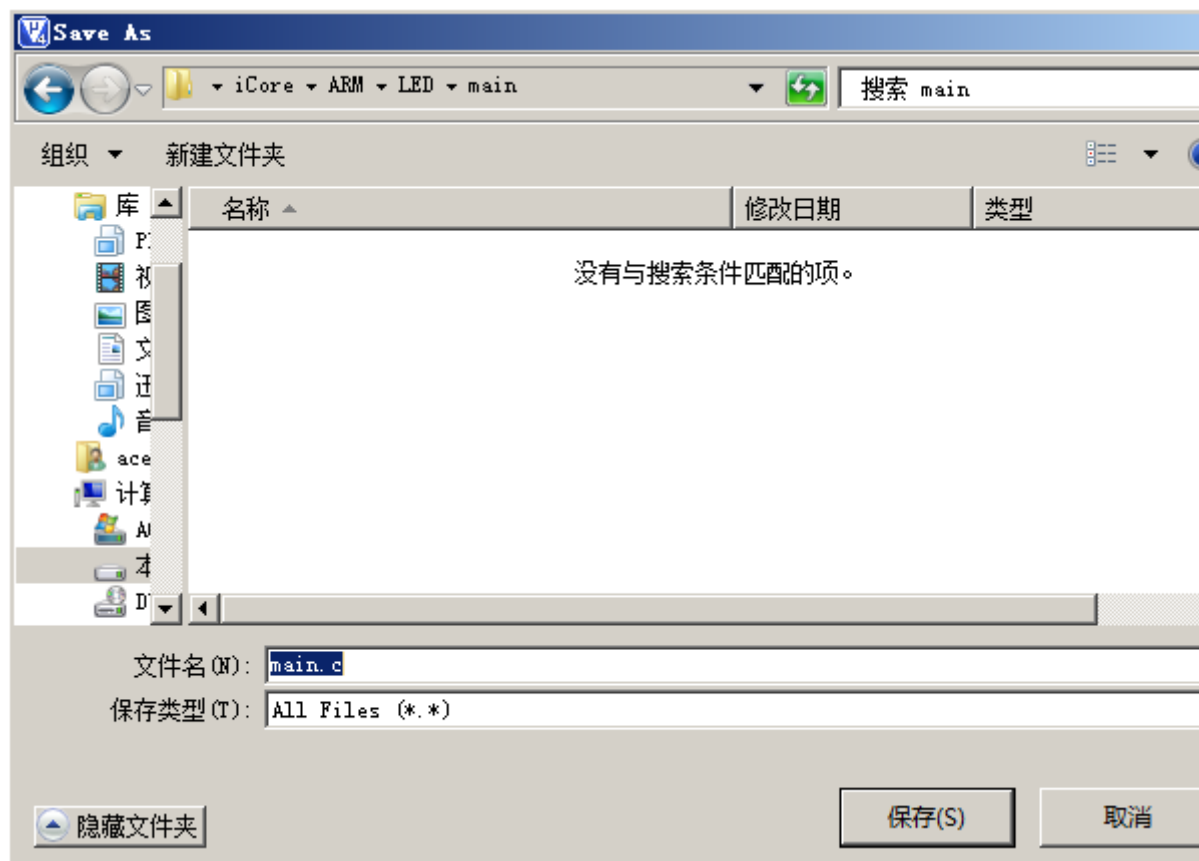


```
45 * Name           : main
46 * Description     : ---
47 * Author          : XiaomaGee.
48 *
49 * History
50 * -----
51 * Rev             : 0.00
52 * Date            : 03/05/2012
53 *
54 * create.
55 * -----
56 */
57 int main(void)
58 {
59     int i;
60
61     rcc.initialize();
62     nvic.initialize();
63
64     LED_OUTPUT;
65
66     while (1) {
67         LED_ON;
68         for (i = 0; i < 1000000; i++) ;
69         LED_OFF;
70         for (i = 0; i < 1000000; i++) ;
71     }
72 }
73
```

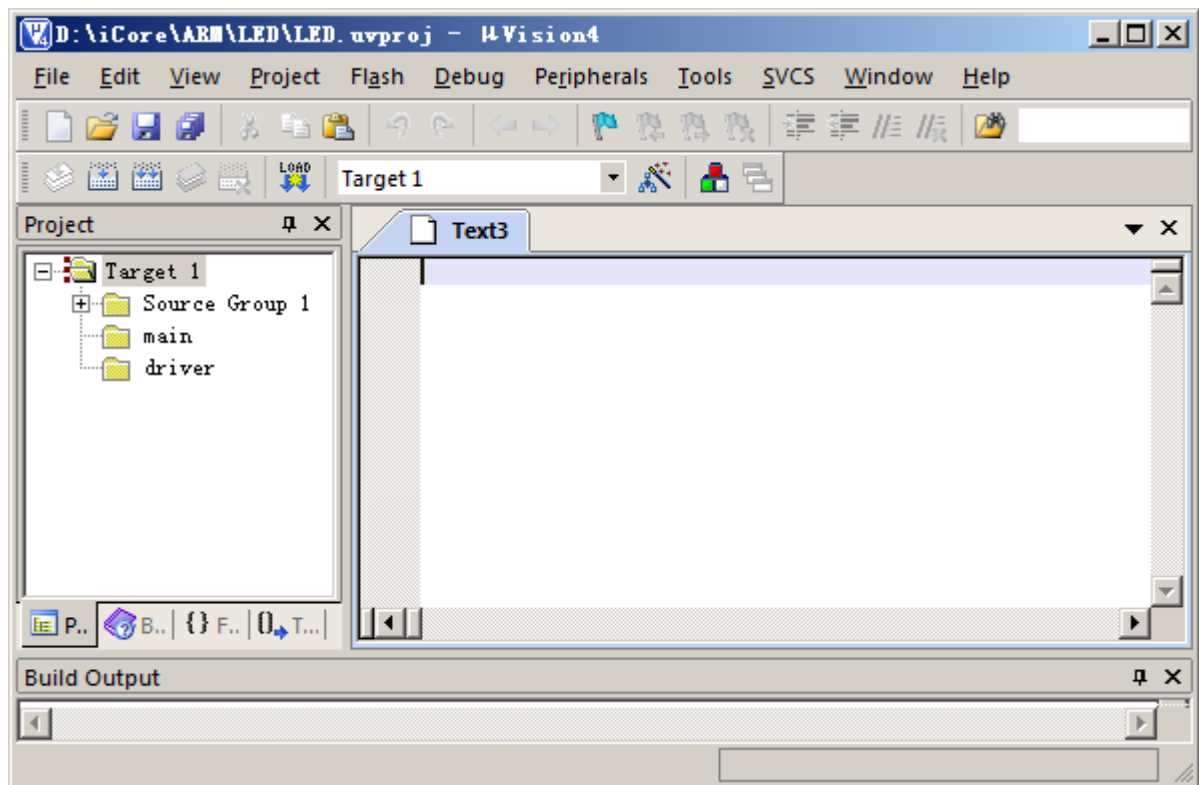
14、 点击保存,如下图所示,在 LED 文件夹下新建一个文件夹,命名为“main”



15、 打开 main 文件夹，文件名输入 “main.c”，如下图所示，然后点击保存



16、 点击菜单 “File/New” 选项，如下图所示：



17、 将下列程序代码输入到此空的 Text3 文档中

```
1 /*
2  * FILE                : rcc.c
3  * DESCRIPTION          : This file is iCore rcc driver demo.
4  * Author               : XiaomaGee@gmail.com
5  * Copyright            :
6  *
7  * History
8  * -----
9  * Rev                  : 0.00
10 * Date                 : 03/05/2012
11 *
12 * create.
13 * -----
14 */
15
16 //----- Include files -----//
17
18 #include "..\include\hardware.h"
19 #include "..\include\rcc.h"
20
21 //----- Function Prototype -----//
22
23 static int initialize(void);
24
25 //----- Variable -----//
26
27 SYS_RCC_T rcc = {
28     .initialize = initialize,
29 };
30
31 //-----Function-----//
32
33 /*
34  *Name                  : initialize
35  * Description: ---
36  *Author                : XiaomaGee.
37  *
38  * History
39  * -----
40  * Rev
41  * Date                 : 03/05/2012
42  *
43  * create.
44  * -----
```

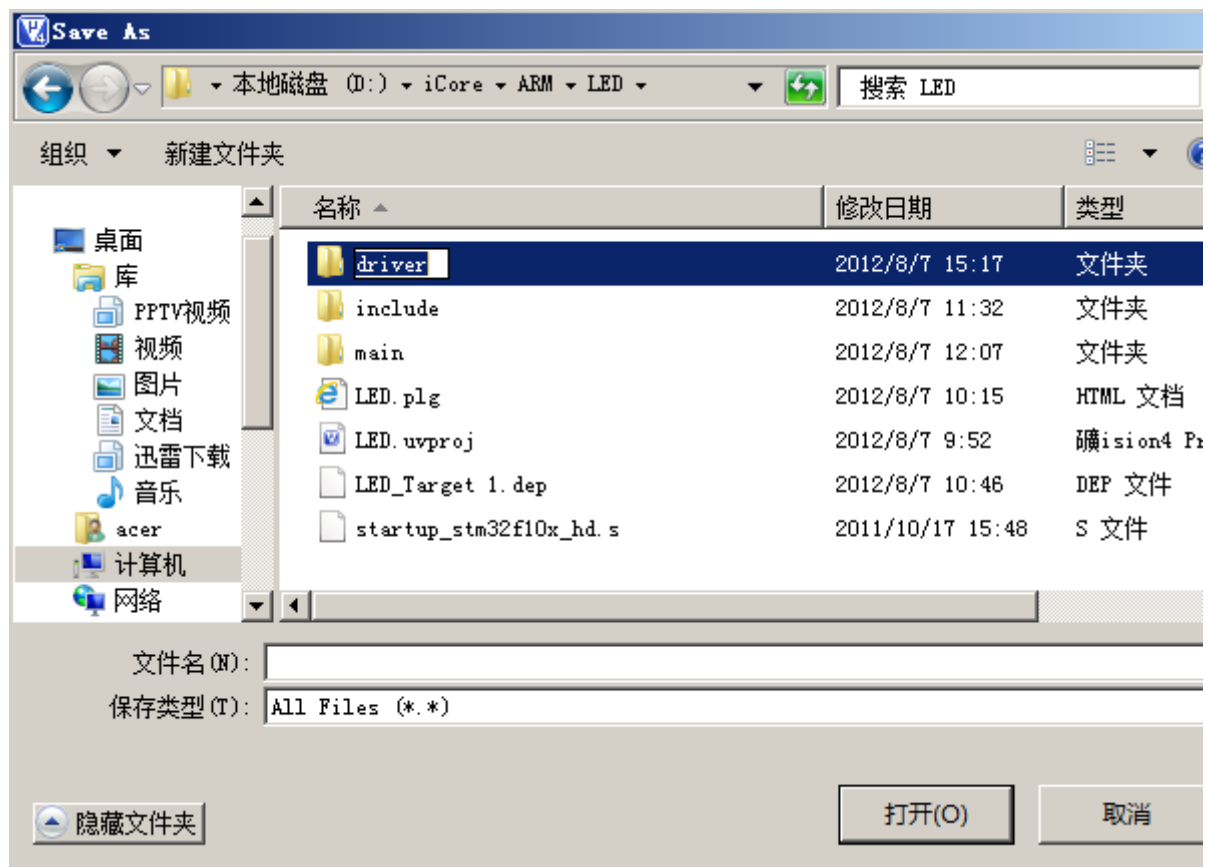
```

45  */
46  static int
47  initialize(void)
48  {
49      RCC->CR.B.HSEON = 1;    //Enable external high-speed clock
50
51      while (!(RCC->CR.B.HSERDY == 1)) ; //检测 HSE 时钟是否稳定
52      RCC->CFGR.B.HPRE = 0;    //AHB 预分频系数// 0 不分频
53      RCC->CFGR.B.PPRE2 = 0;   //APB2 预分频系数//0 不分频
54      RCC->CFGR.B.PPRE1 = 4;   //APB1 预分频系数 100: HCLK 2 分频
55                                //APB1 不超过 36MHz
56      RCC->CFGR.B.ADCPRE = 1;  //ADC 预分频系数
57                                //01: PCLK2 4 分频后作为 ADC 时钟
58      RCC->CFGR.B.PLLSRC = 1;  //选择 HSE 还是 HSI/2 作为 PLL 输入
59                                // 1: HSE 时钟作为 PLL 输入时钟。
60      RCC->CFGR.B.PLLXTPRE = 0; //HSE 分频作为 PLL 输入
61                                //0: HSE 不分频
62                                //1: HSE 2 分频
63      RCC->CFGR.B.PLLMUL = 7;  //PLL 倍频倍数
64                                //0111: PLL 9 倍频输出
65                                //PLLCLK = 8MHz * 9 = 72 MHz
66      RCC->CR.B.PLLON = 1;     //使能 PLL
67      while (!(RCC->CR.B.PLLRDY == 1)) ; //检测 PLL 是否准备完毕
68
69      FLASH->ACR.B.LATENCY = 2; //000: 零等待状态, 当 0 < SYSCLK < 24MHz
70                                //001: 一个等待状态, 当 24MHz < SYSCLK < 48MHz
71                                //010: 两个等待状态, 当 48MHz < SYSCLK < 72MHz
72
73      RCC->CFGR.B.SW = 2;      //选择 PLL 输出作为系统时钟
74                                //10: PLL 输出作为系统时钟;
75      while (!(RCC->CFGR.B.SWS == 2)) ; //检测 PLL 是否已经作为系统时钟
76                                // 10: PLL 输出作为系统时钟;
77
78      RCC->APB2ENR.B.IOPAEN = 1; //Enable GPIOA clocks
79      RCC->APB2ENR.B.IOPBEN = 1; //Enable GPIOB clocks
80      RCC->APB2ENR.B.IOPCEN = 1; //Enable GPIOC clocks
81      RCC->APB2ENR.B.IOPDEN = 1; //Enable GPIOD clocks
82      RCC->APB2ENR.B.IOPEEN = 1; //Enable GPIOE clocks
83
84      RCC->APB2ENR.B.AFIOEN = 1;
85
86      RCC->APB2ENR.B.USART1EN = 1; //Enable USART clocks
87      RCC->AHBENR.B.FSMCEN = 1;   //Enable FSMC clocks

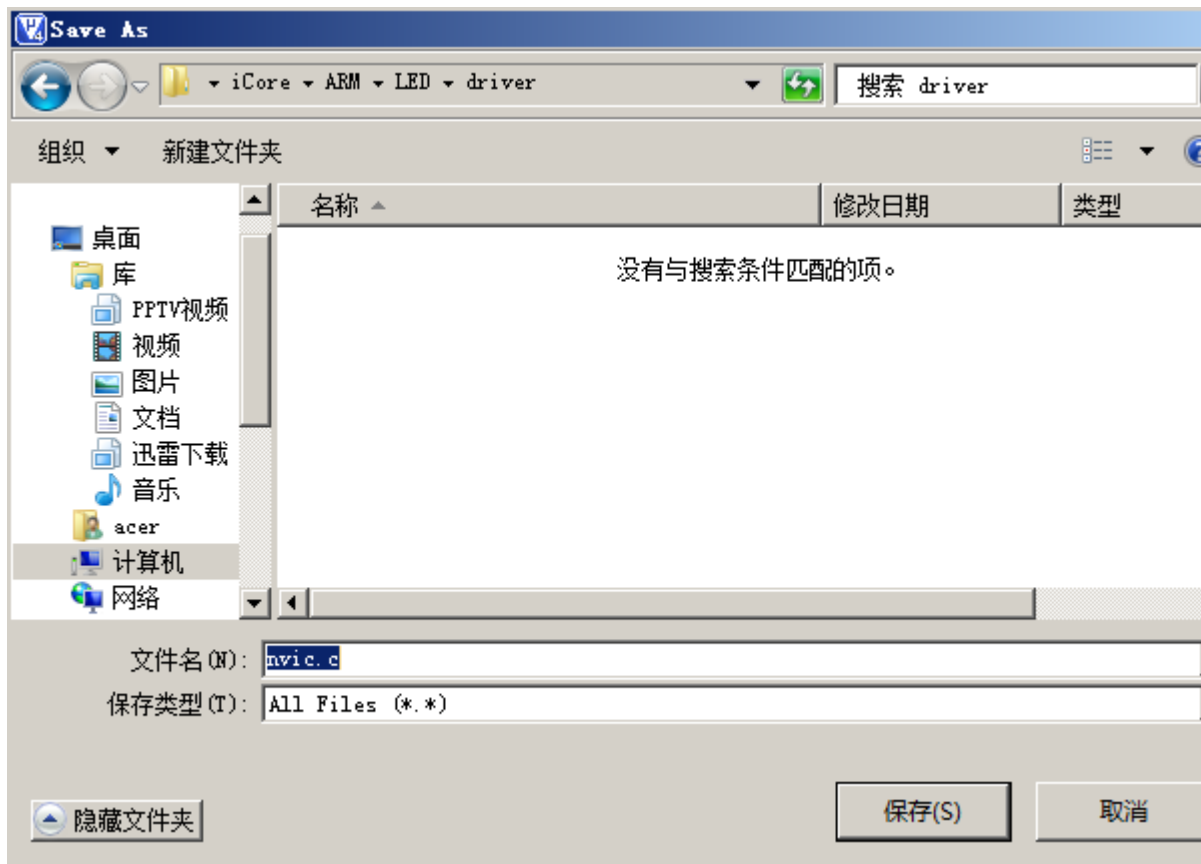
```

```
82
83     RCC->APB1ENR.B.TIM2EN = 1;
84     RCC->APB2ENR.B.TIM1EN = 1;
85
86     return 0;
87 }
88
```

18、 点击保存，如下图所示，新建一个文件夹，命名为“driver”

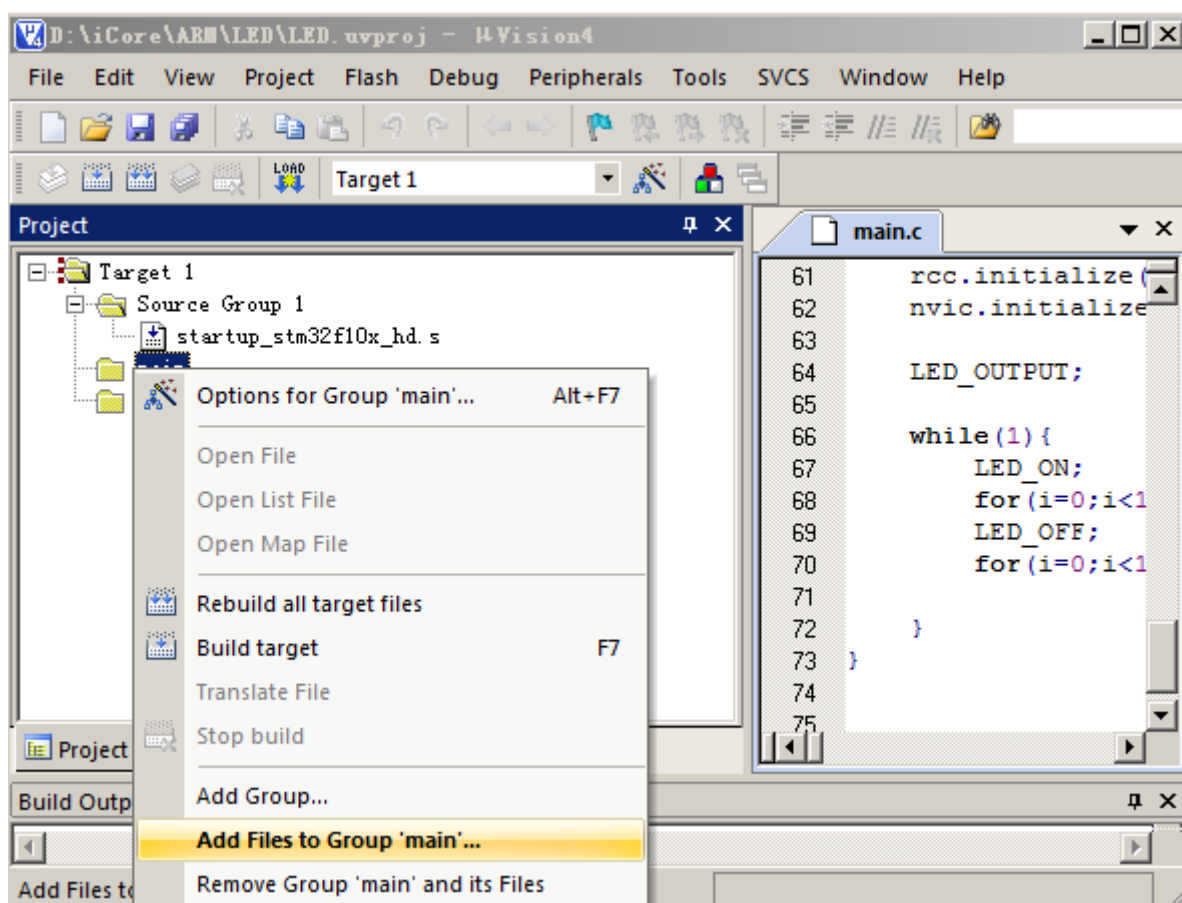


19、 打开 driver 文件夹，文件名输入“nvic.c”，如下图所示，然后点击保存

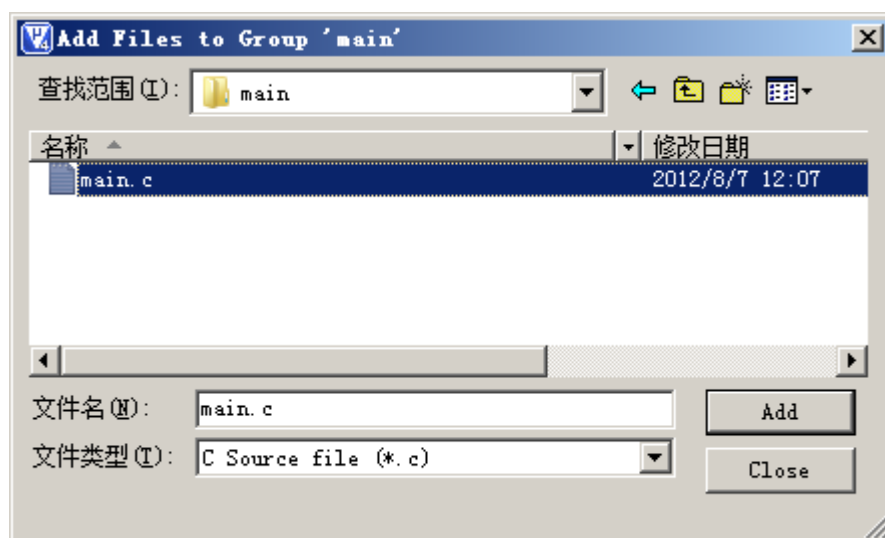


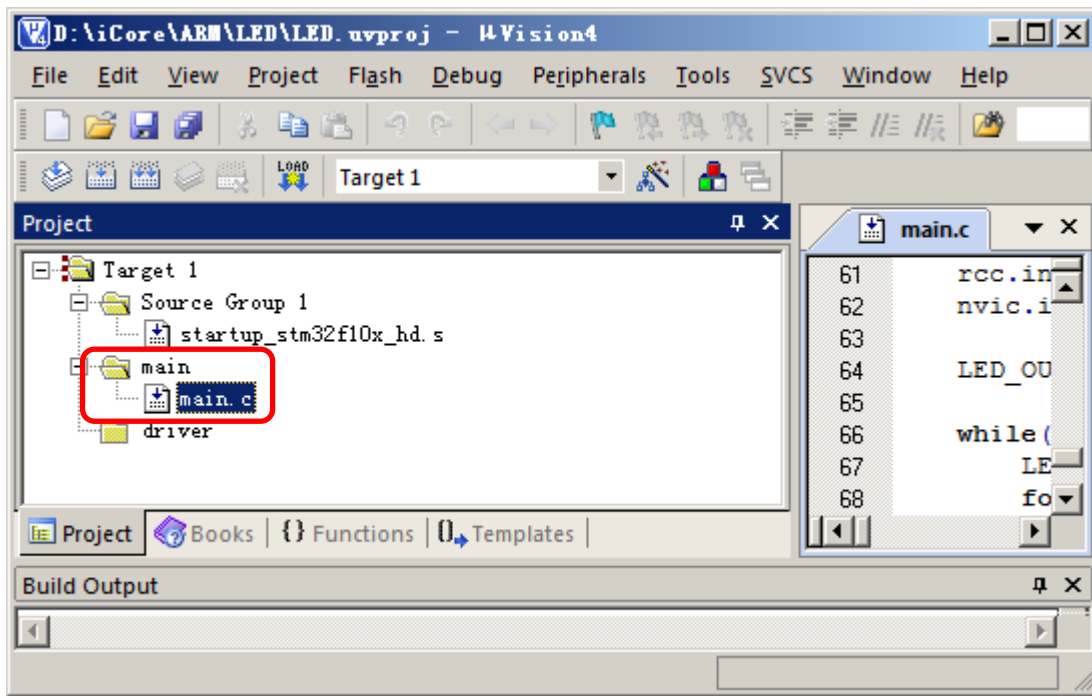
20、 参 照 上 面 的 步 骤 ， 分 别 输 入 `rcc.h` / `rcc.c` / `nvic.h` / `stm32f10x_reg.h/config.h`(这些程序在配套的光盘里都有 ,在此不一 一贴出)的代码，并把 `rcc.h` 和 `nvic.h` 保存到文件夹 `include` 中，把 `rcc.c` 保存到文件夹 `driver` 中,把 `config.h` 保存到一新建文件夹 `config` 中

21、 在 `main` 文件上，点击右键，选中“Add Files to Group ‘main’ ”，如下图所示：

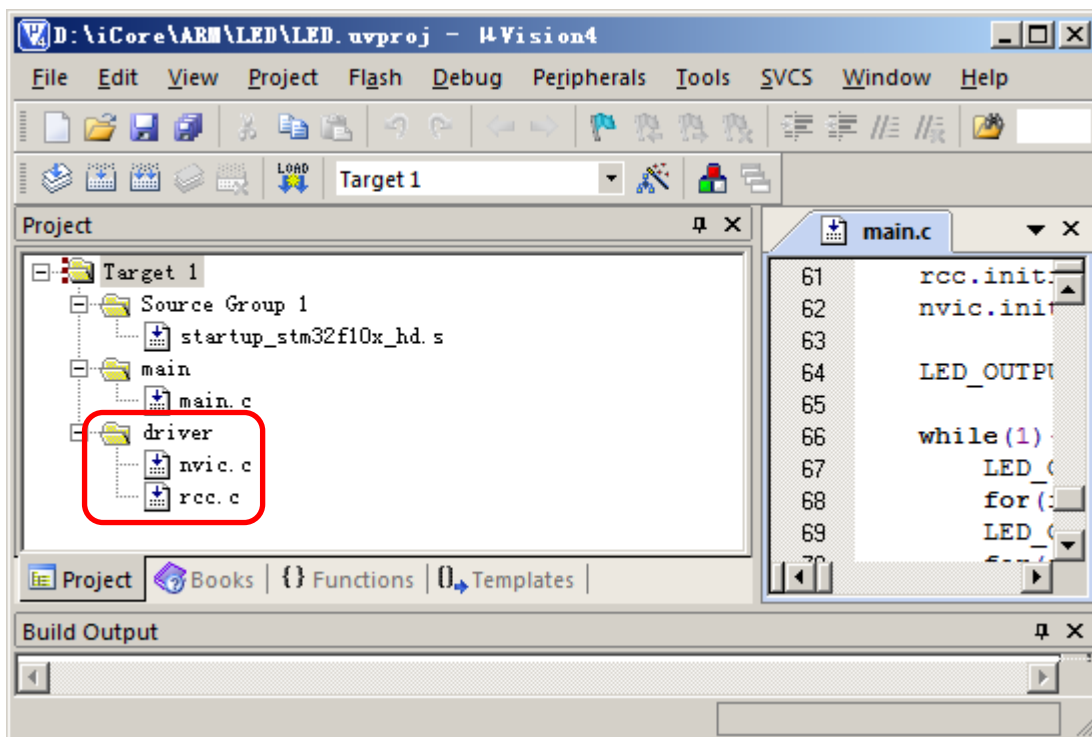


22、把在第 15 步中建立的 main.c 文件，添加进来，如下图所示：

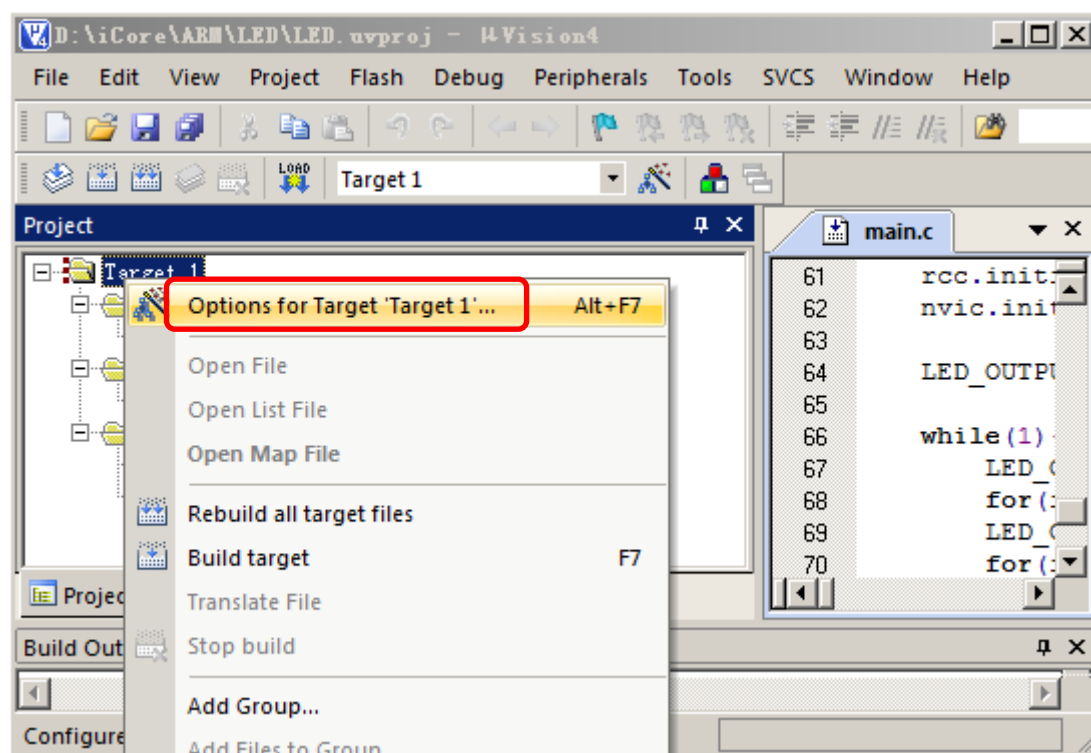




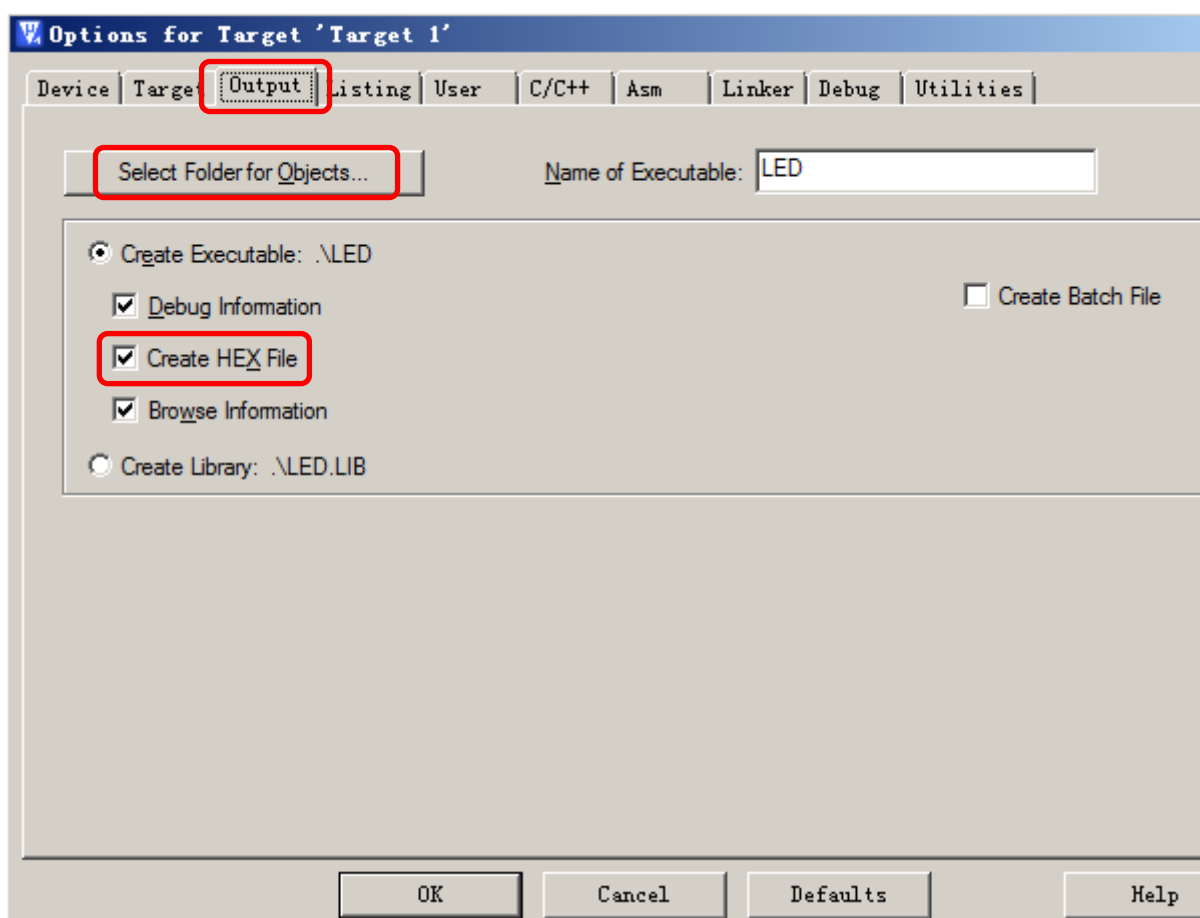
23、 参照 21~22 的步骤，把文件 rcc.c 和 nvic.c 添加到 “driver” 中



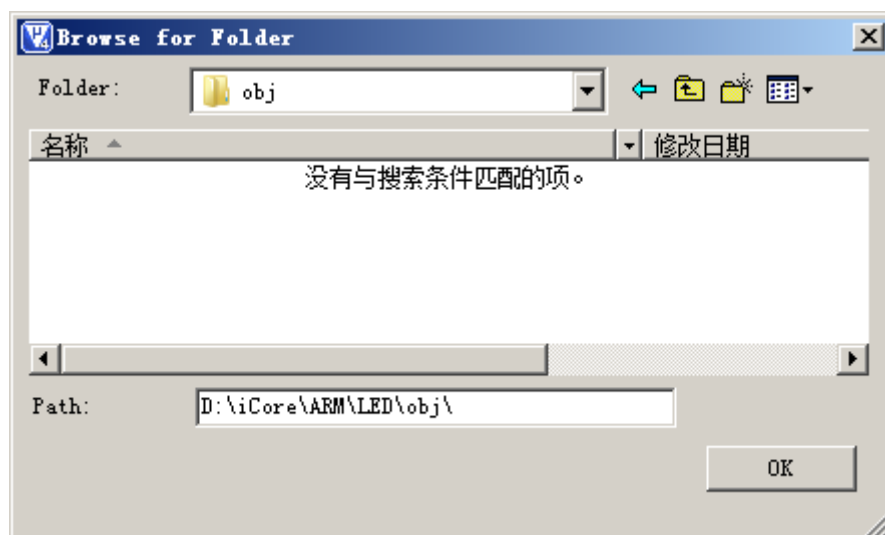
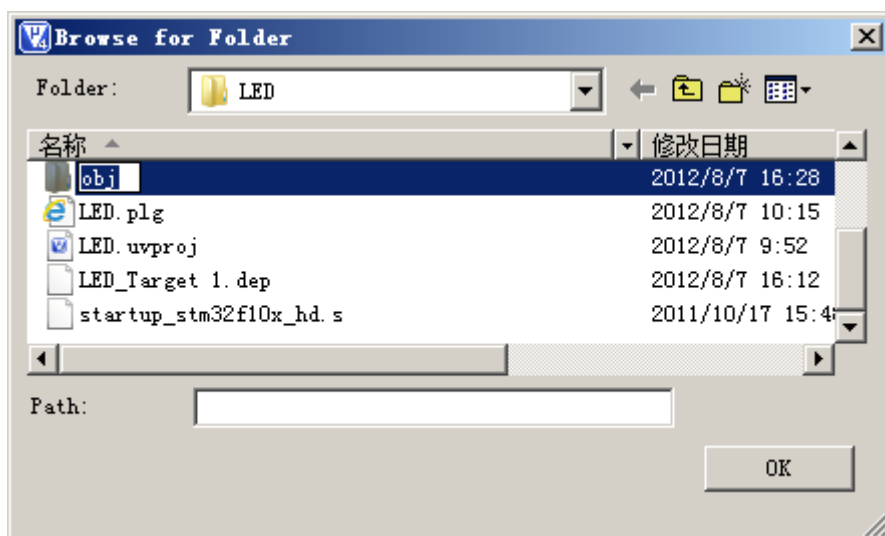
24、 在 Target1 上，点击右键，如下图，选中 “Option for Target.....”：



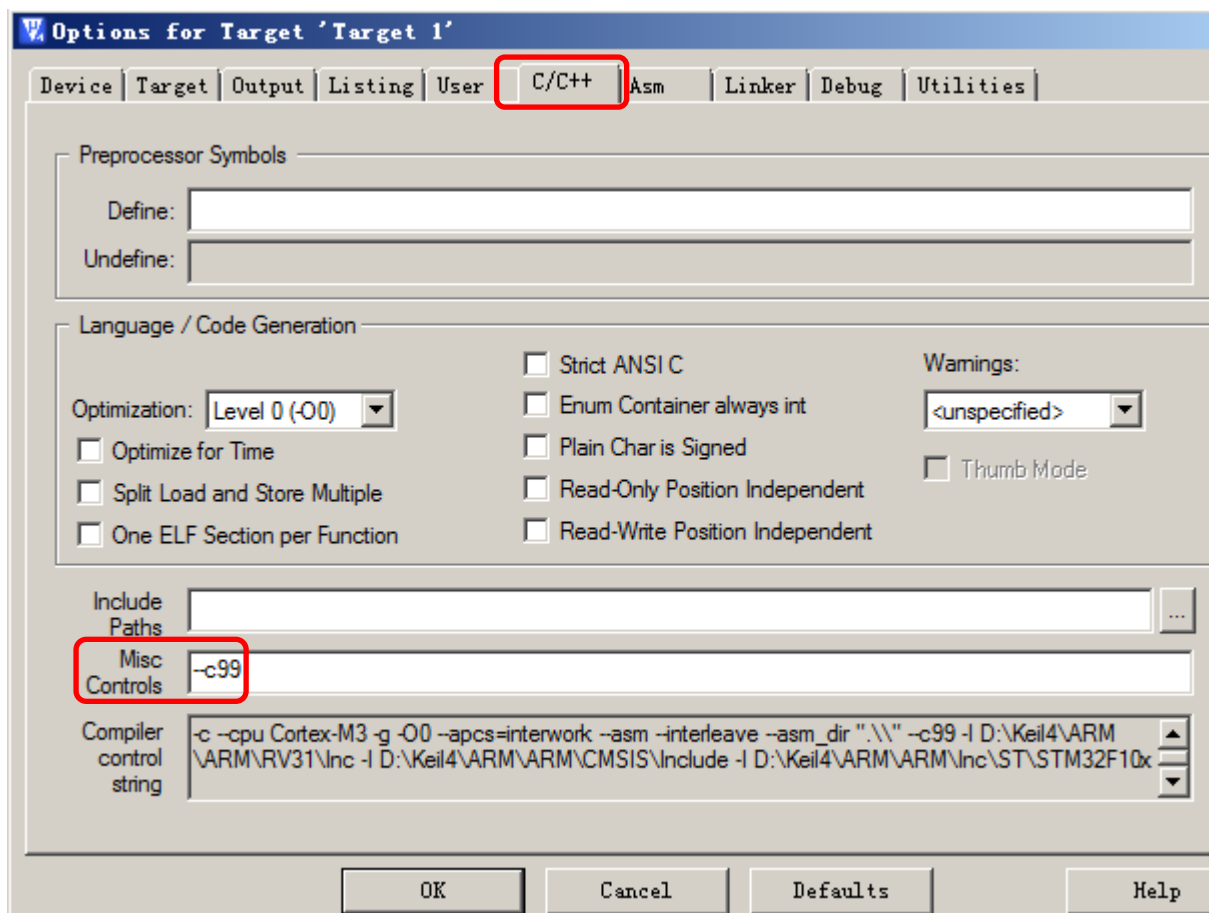
25、 在弹出的界面中，选择“output”，同时在 Creat HEX File 前面打对勾，如下图所示：



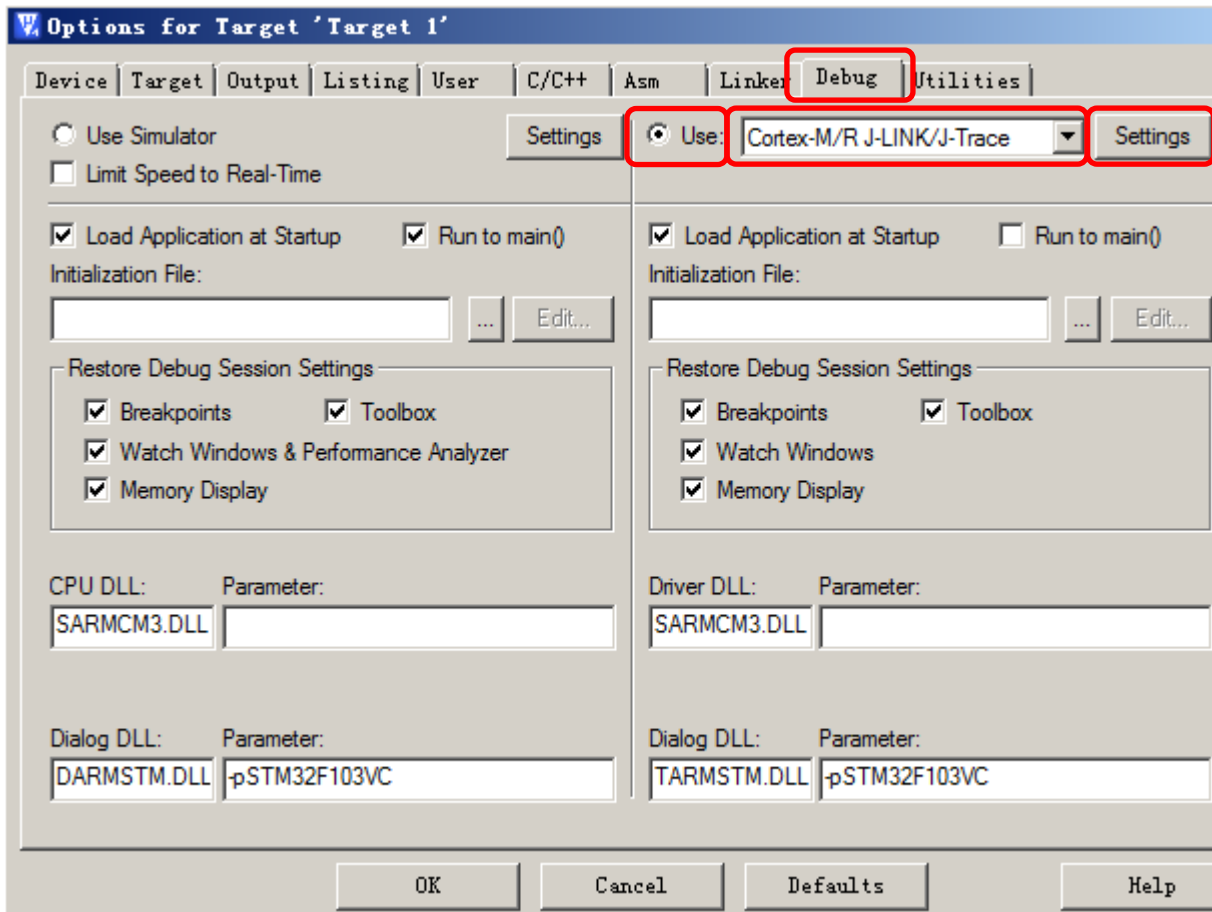
- 26、 点击 “Select Folder for Object”，新建一个 obj 文件夹，将编译的.Hex 文件放在 obj 文件中，点击 “OK”



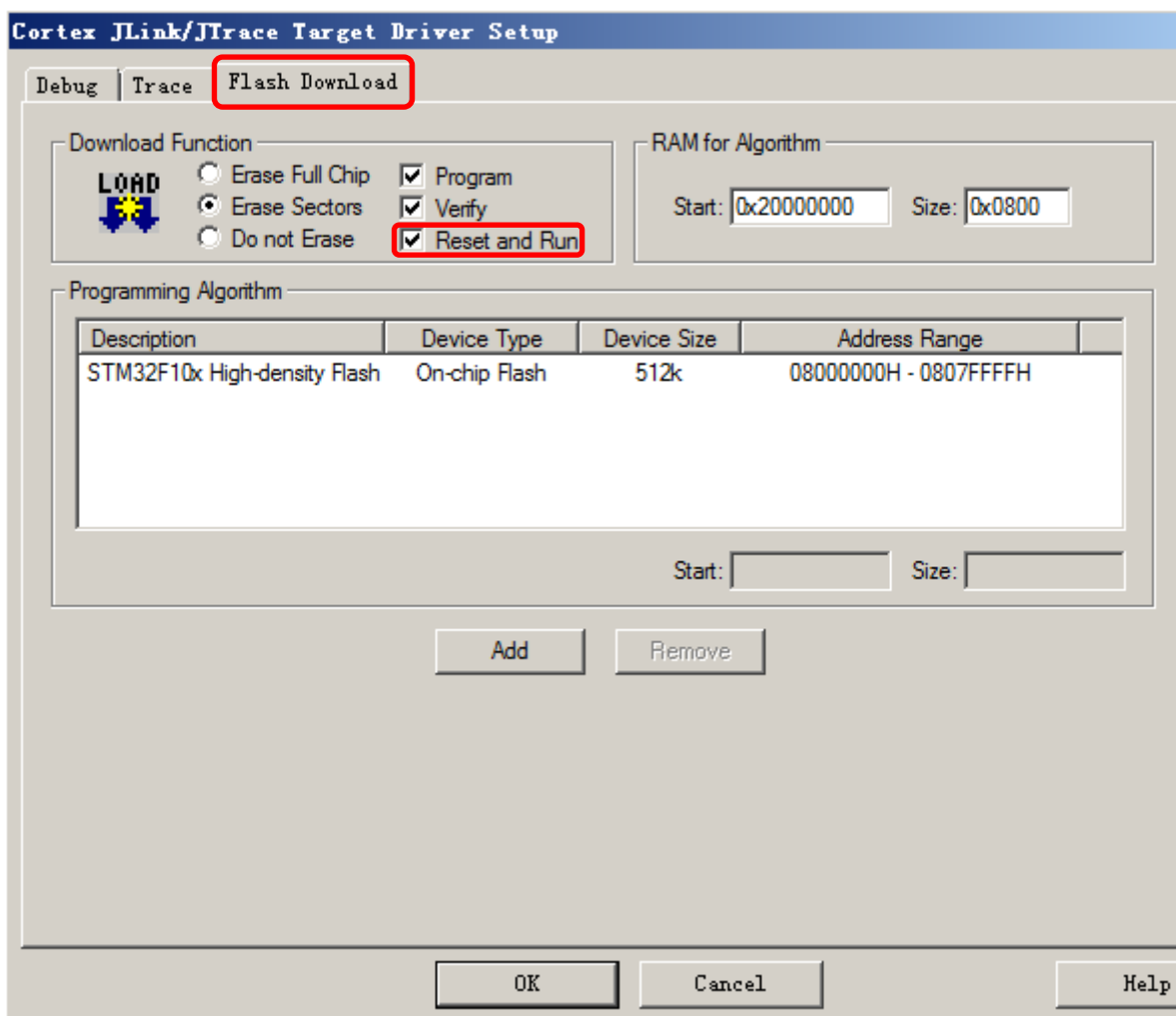
- 27、 点击 “C/C++”，在 Misc controls 后输入 “--c99”，如下图所示：



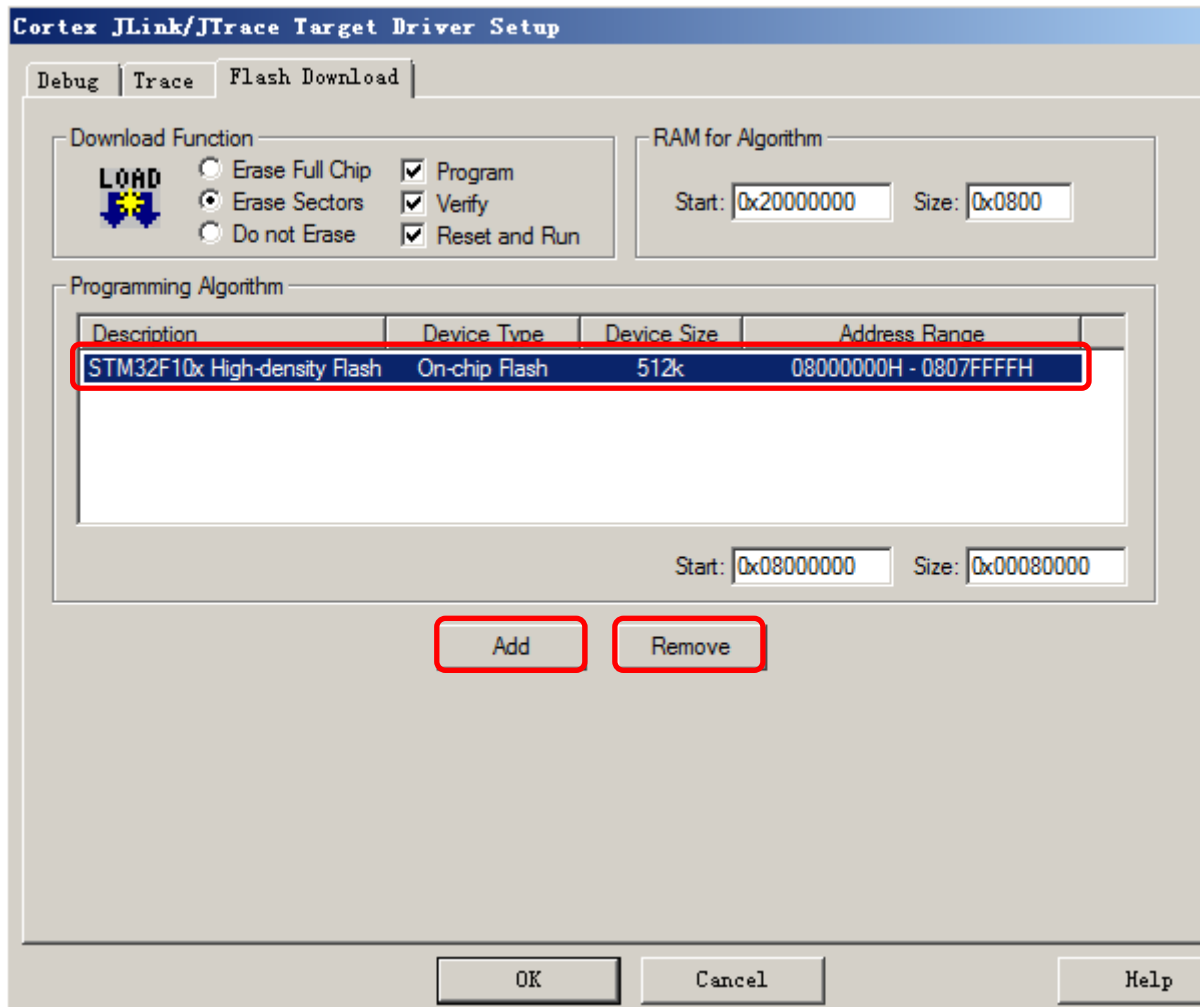
- 28、 点击 “debug”，如下图所示，点击 “use” 和红框内的下拉菜单，选择 “Cortex-M/R J-LINK/J-Trace”



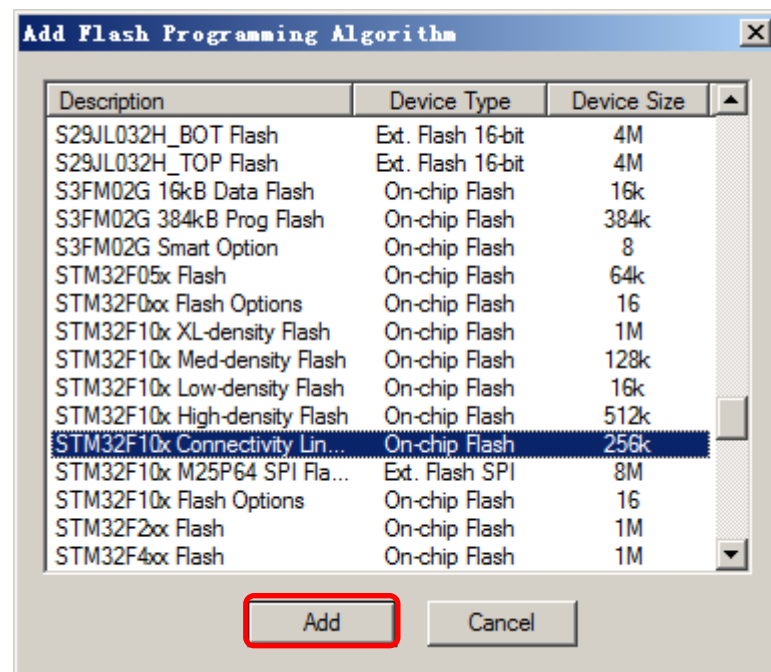
29、 点击上图中的“Settings”，弹出下面的窗口，选择“Flash Download”，勾选“Reset and Run”



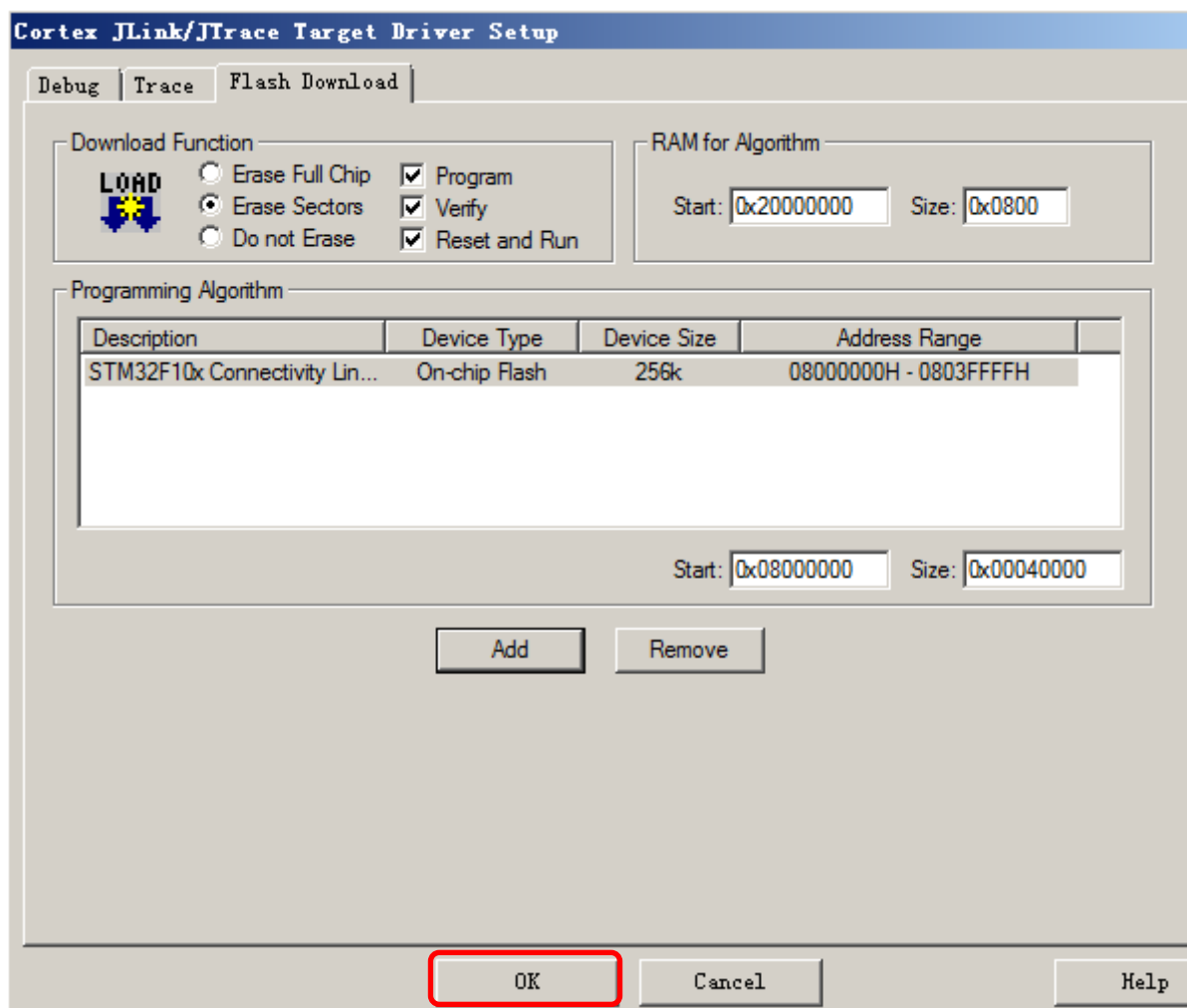
30、 选中下图中红框内容，再点击“Remove”



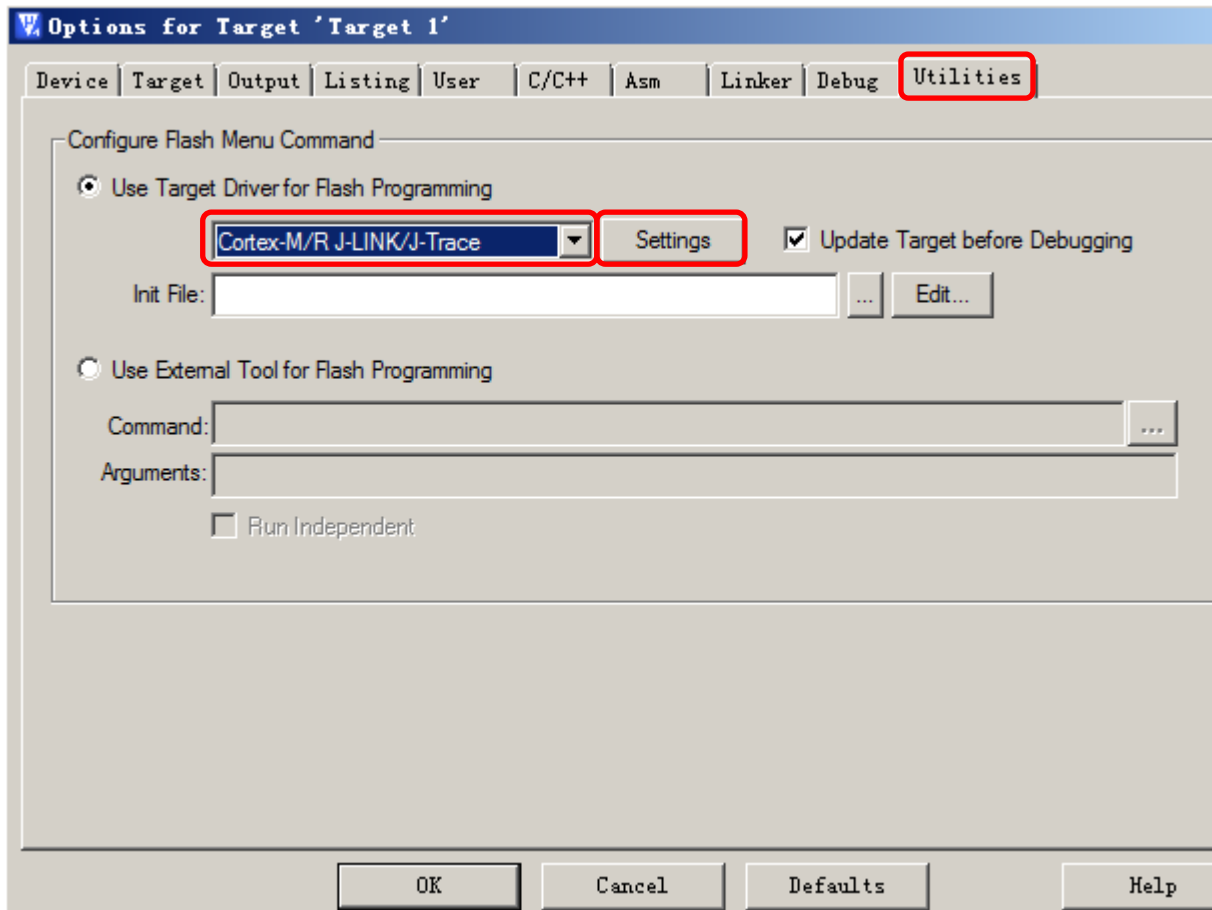
31、 点击上图 “Add”，弹出如下窗口：



- 32、 选中“ STM32F10x Connectivity Lin..” 点击 “Add”，弹出下面窗口后点击 “OK”

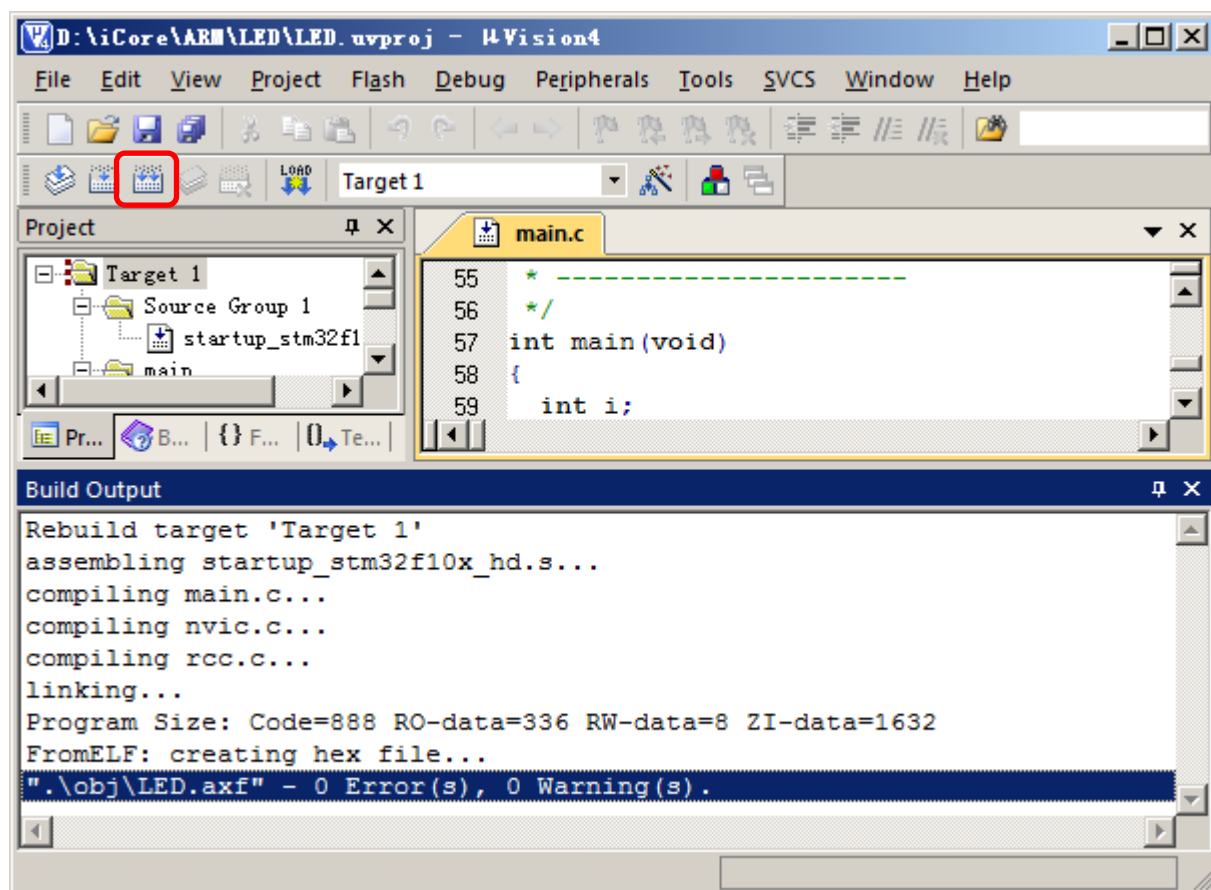


- 33、 点击 “Utilities”，在下面窗口中点击红框内的下拉菜单选择 “Cortex-M/R J-LINK/J-Trace”，再点击 “Settings”




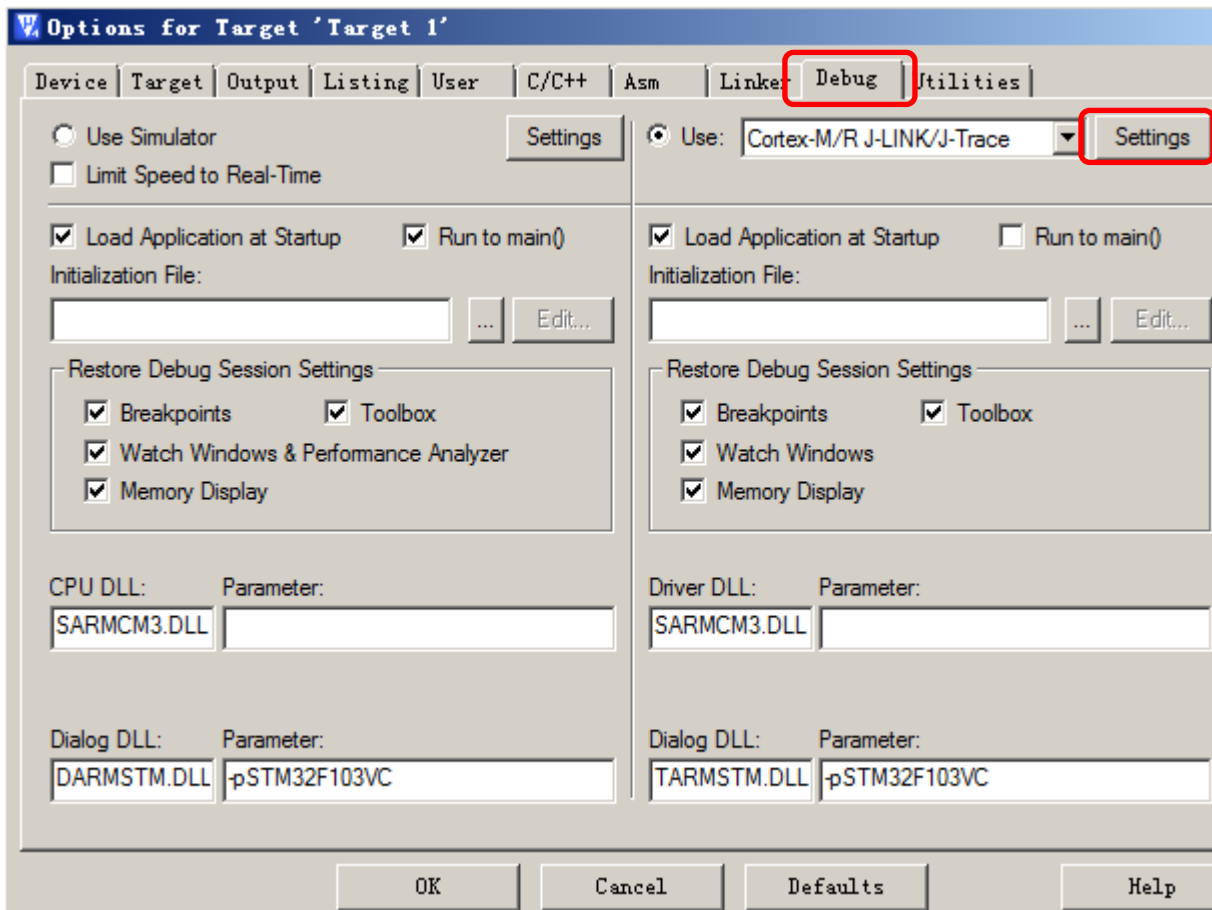
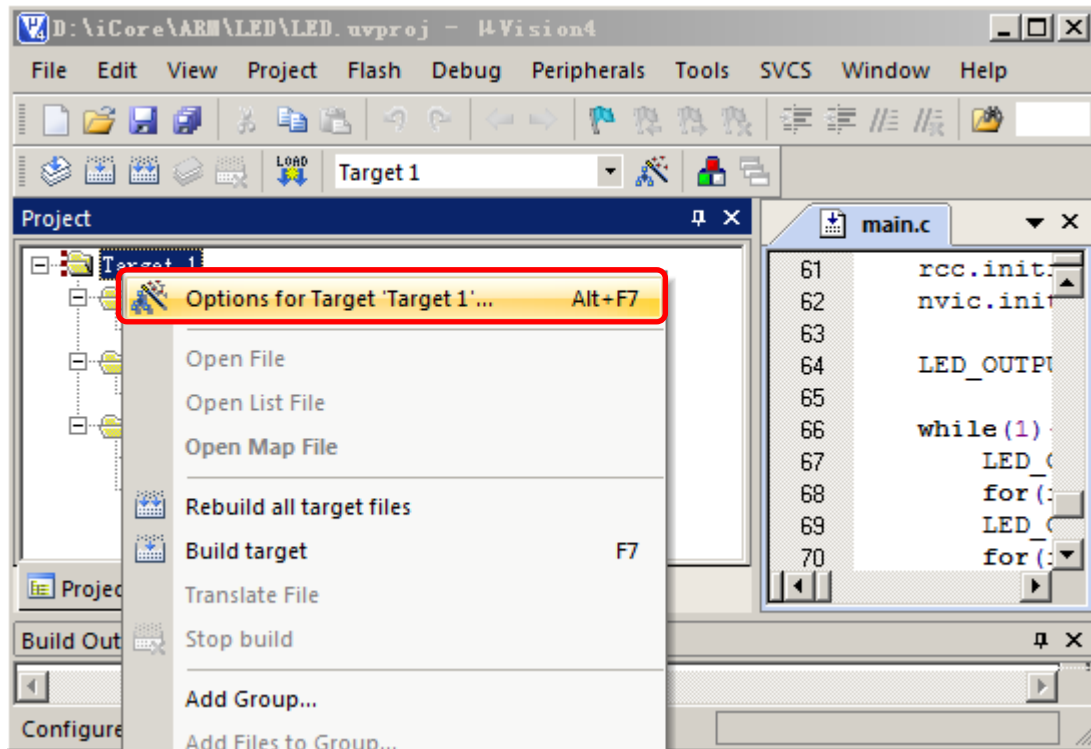
34、 完成后点击 “OK”

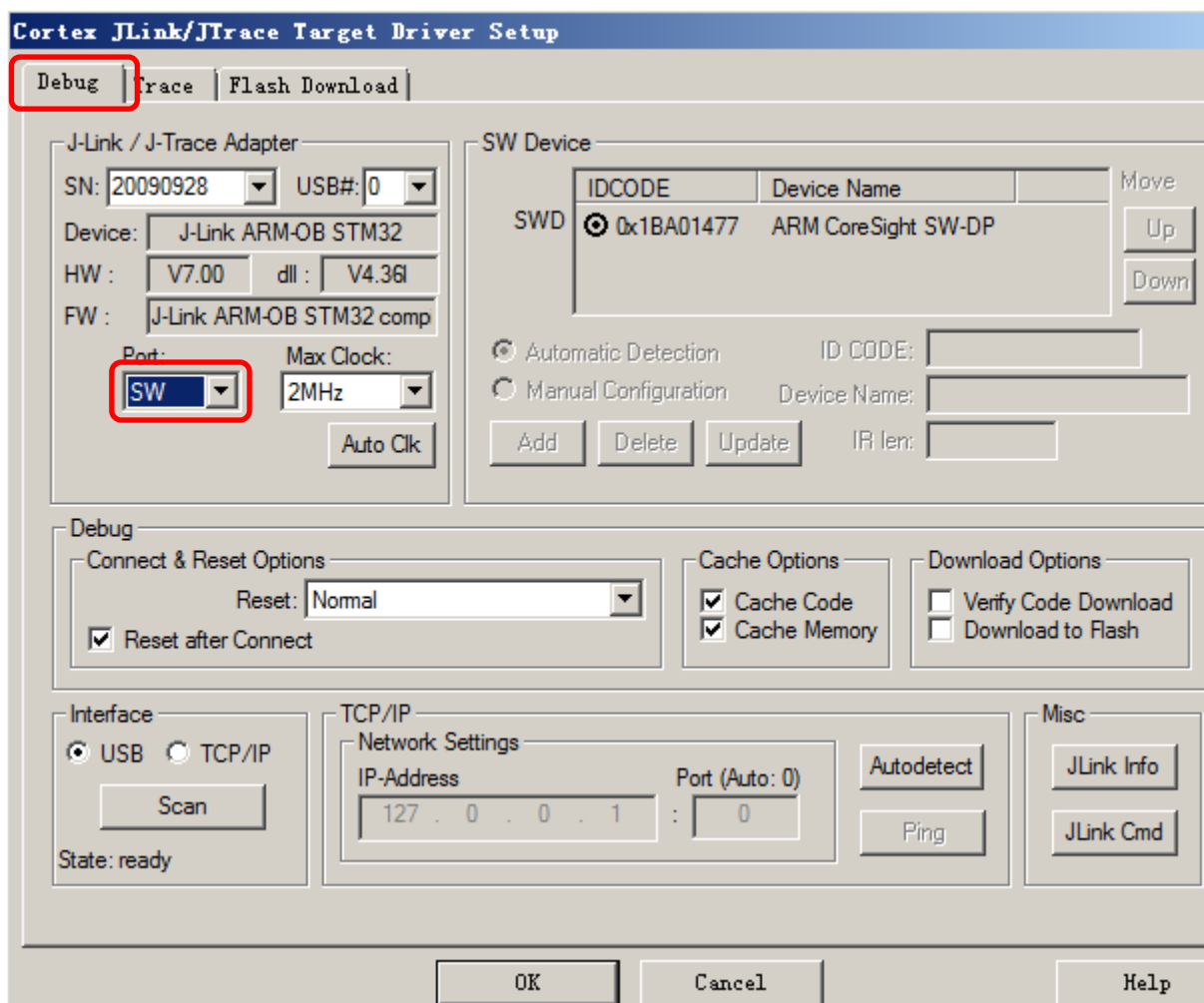
35、 然后点击 “Rebuild”，如下图所示，这样一个完整的 Keil 工程就建立了



五、 程序下载

- 1、 打开 iCore (光盘 A), 在驱动文件夹中, 点击  Setup_JLinkARM_V446f.exe 安装驱动。
- 2、 JLink 驱动安装成功后, 把 JLink 一端用 3P 线连接 iCore 的串口, 另一端直接插入电脑 USB 口; 再用 USB 线分别连接电脑和 iCore。
- 3、 下面对端口进行设置, 如下面系列图所示:





然后点击“OK”即可

4、把上面编译好的程序下载到 iCore 即可通过 iCore 开发板观察现象，如下图

