

STM32 V3.4 固件库使用实例

1 概述:

刚入手 STM32，查了好多资料也看了好多开发板的程序和教程，以自己的思路先上一个工程框架吧，剩下的就是大家想加什么就加什么了，希望对和我一样的 STM32 初学者有点帮助，如果您是初次使用 ARM MDK 希望对你也有帮助，本实例是一个“神马”功能都没有的 RS422 模块。

2 建立目录

2.1 新建工程目录：“RS422”，你也可以根据自己的需要命名此顶层目录；

2.2 在目录“RS422”下新建“RVMDK”目录，表示采用 ARM MDK 开发环境；

2.3 在“RVMDK”目录下新建目录“V1”，表示软件版本 V1.0，这个好处在于下次将整个目录复制一下改为“V2”，软件版本就是 V2.0 了。

2.4 在“V1”目录下新建“Libraries”目录；

2.5 在“V1”目录下新建“Project”目录。

2.6 在“Project”目录下新建“OBJ”、“LIST”、“Pro”3个目录。

2.7 在“V1”目录下新建“USER”目录。

2.8 在“USER”目录下新建“INC”、“SRC”2个目录。

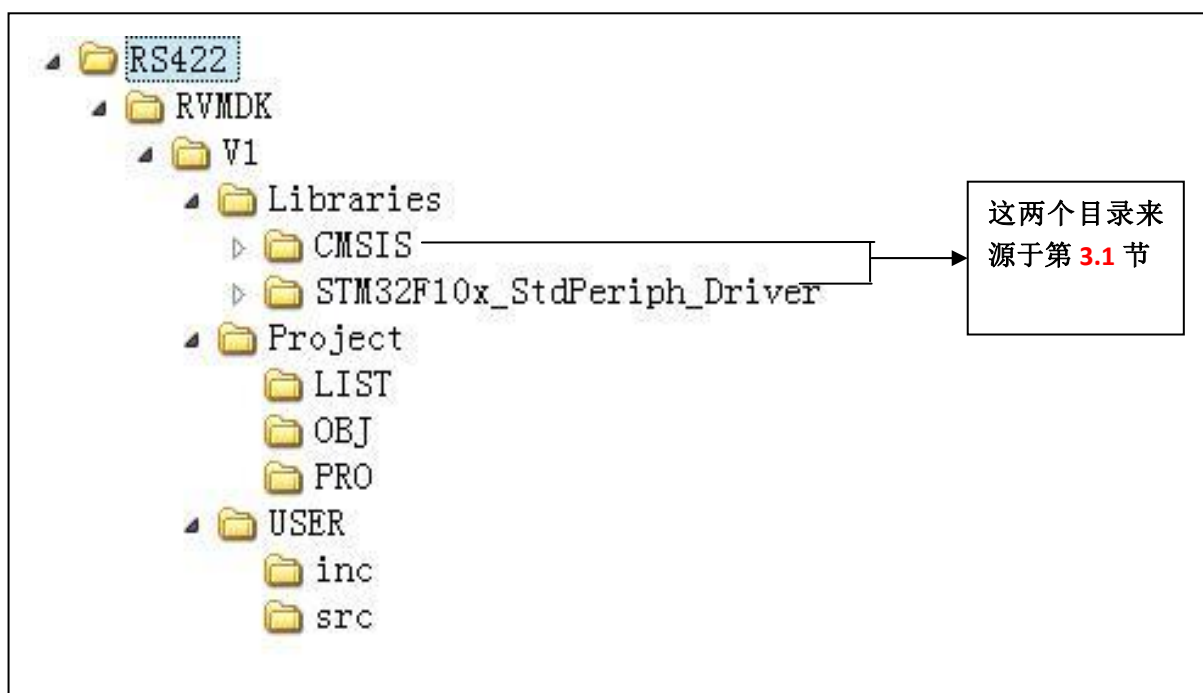


图 1: 目录结构图

3 拷贝文件 *（已经写的很详细了，就不上图了）*

3.1 将固件库目录 “STM32F10X_StdPeriph_lib_V3.4.0” → “Libraries” 下的所有目录拷贝到工程目录 “RS422” → “RVMDK” → “V1” → “Libraries” 目录下。

3.2 将固件库目录 “STM32F10X_StdPeriph_lib_V3.4.0” → “Project” → “STM32F10x_StdPeriph_Examples” → “GPIO” → “IOToggle” 目录下的 “stm32f10x_it.c”、“system_stm32f10x.c” 拷贝到工程目录 “RS422” → “RVMDK” → “V1” → “USER” → “SRC” 目录下，在此 “SRC” 目录下新建 “main.c” 文件，“main.c” 先 “神马” 也不写。

3.3 将固件库目录 “STM32F10X_StdPeriph_lib_V3.4.0” → “Project” → “STM32F10x_StdPeriph_Examples” → “GPIO” → “IOToggle” 目录下的 “stm32f10x_conf.h”、“stm32f10x_it.h” 这 2 个文件拷贝到工程目录 “RS422” → “RVMDK” → “V1” → “USER” → “INC” 目录下。

4 建立工程：*（已经写的很详细了，就不上图了）*

4.1 我用的是 ARM 的 **MDK4.14** 开发环境，运行 “Keil uVision4” ；

4.2 点击主菜单栏 “Project” → “New uVision Project”，选择在工程目录的 “V1” → “Project” → “Pro” 目录下命名新建工程为 “RS422_MODULE.uvproj”（当然也可以命名为你自己需要的工程名）；

4.3 接下来出现 CPU 选择窗口，选择 CPU 为 “STMicroelectronics” → “STM32F103ZE”（这个大家根据自己的需要选择），点击 “OK” 按钮；

4.4 接下来出现 “Copy STM32 Startup Code to Project Folder and ADD File to Project ?” 提示时选择 “否”（在后面的步骤中会根据 CPU 选择启动文件的，这里不用选择），完成工程建立。

5 工程管理

上一节新建的工程还是空空的，这一步要将它实例化。

5.1 在 “Project” 窗口中用鼠标左键点击 “Target 1”，再点击右键弹出菜单选择菜单中的 “Manage Components” 子菜单（见图 2），出现 “Components, Environment and Books” 窗口（见图 3）。

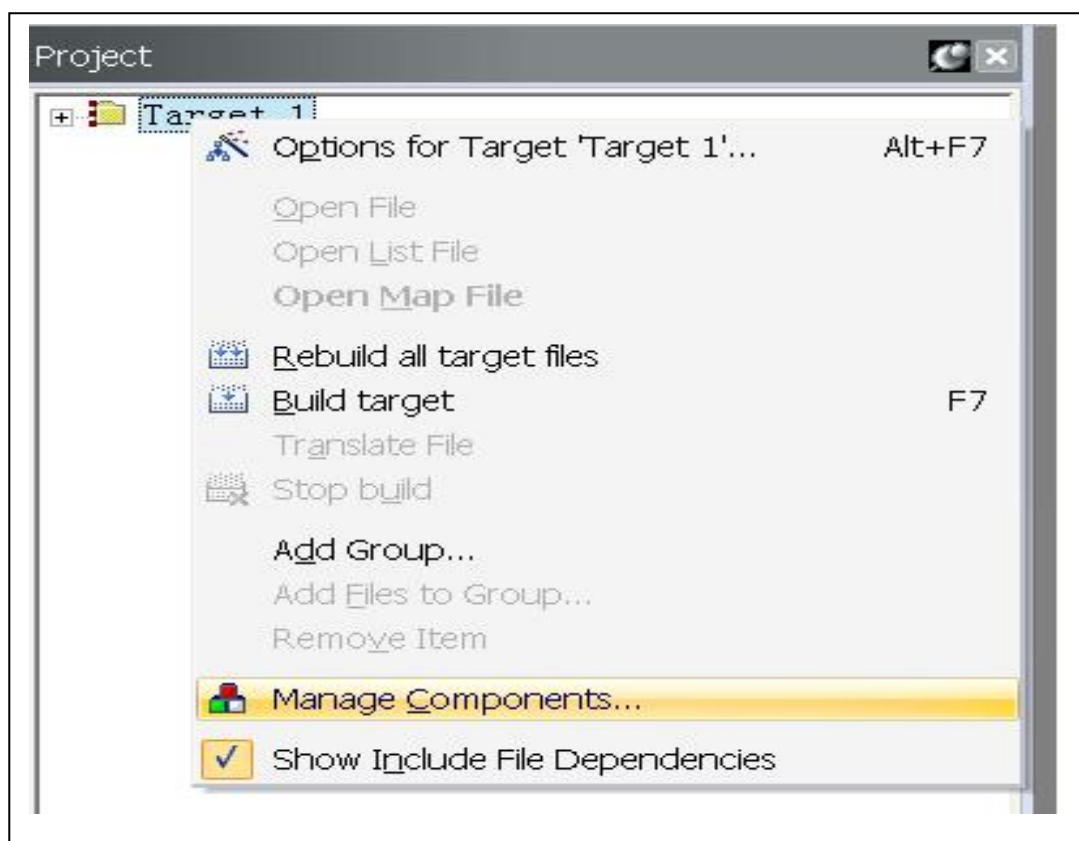


图 2

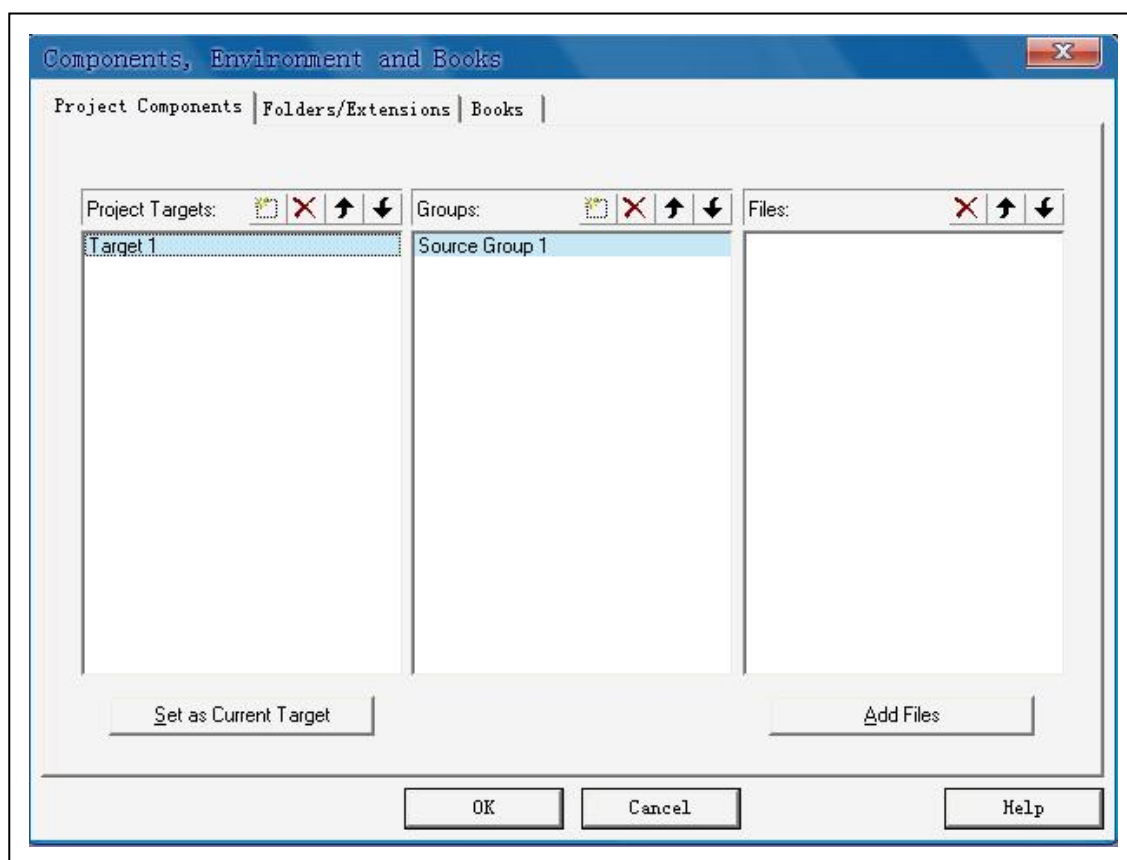


图 3

5.2 用鼠标双击“Project Targets”栏中的“Target 1”将“Target 1”改为“RS422_MODULE”（见图 4），在中间的“Groups”栏中添加“USER”、“STM32_LIB”、“MDK_STARTUP”、“CMSIS”4 个条目（见图 5）。

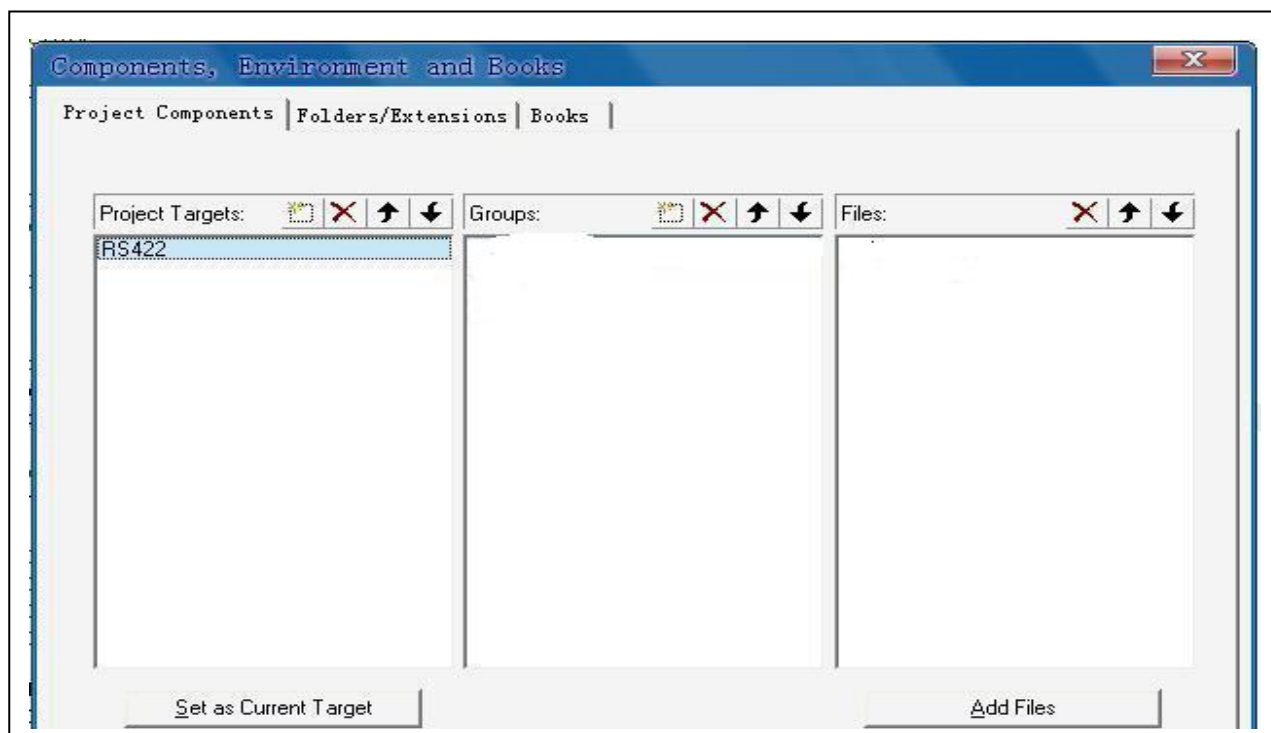


图 4

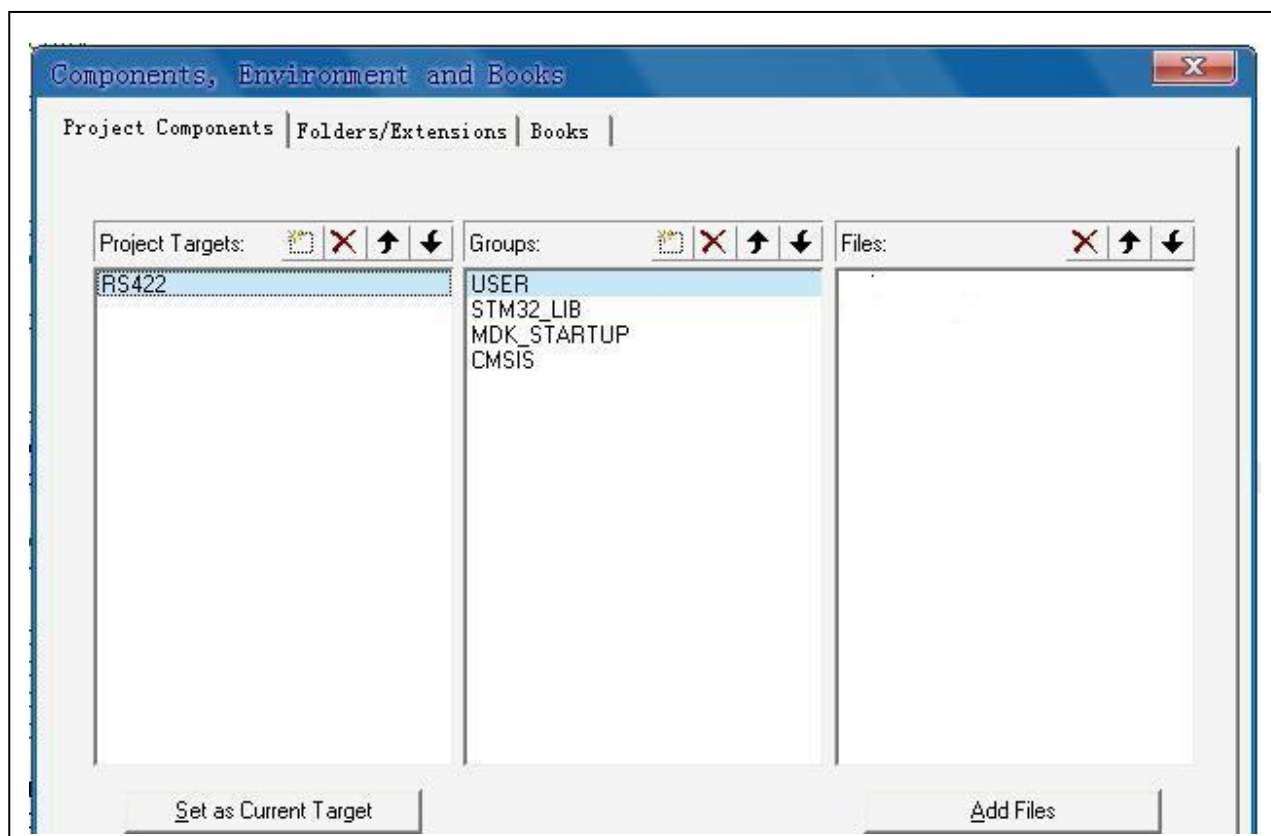


图 5

5.3 在“USER”条目的“Files”栏中添加目录“V1” → “USER” → “SRC”下的“main.c”、“stm3210x_it.c”2个文件，通过窗口的“Add Files”按钮可以添加文件（见图6）。

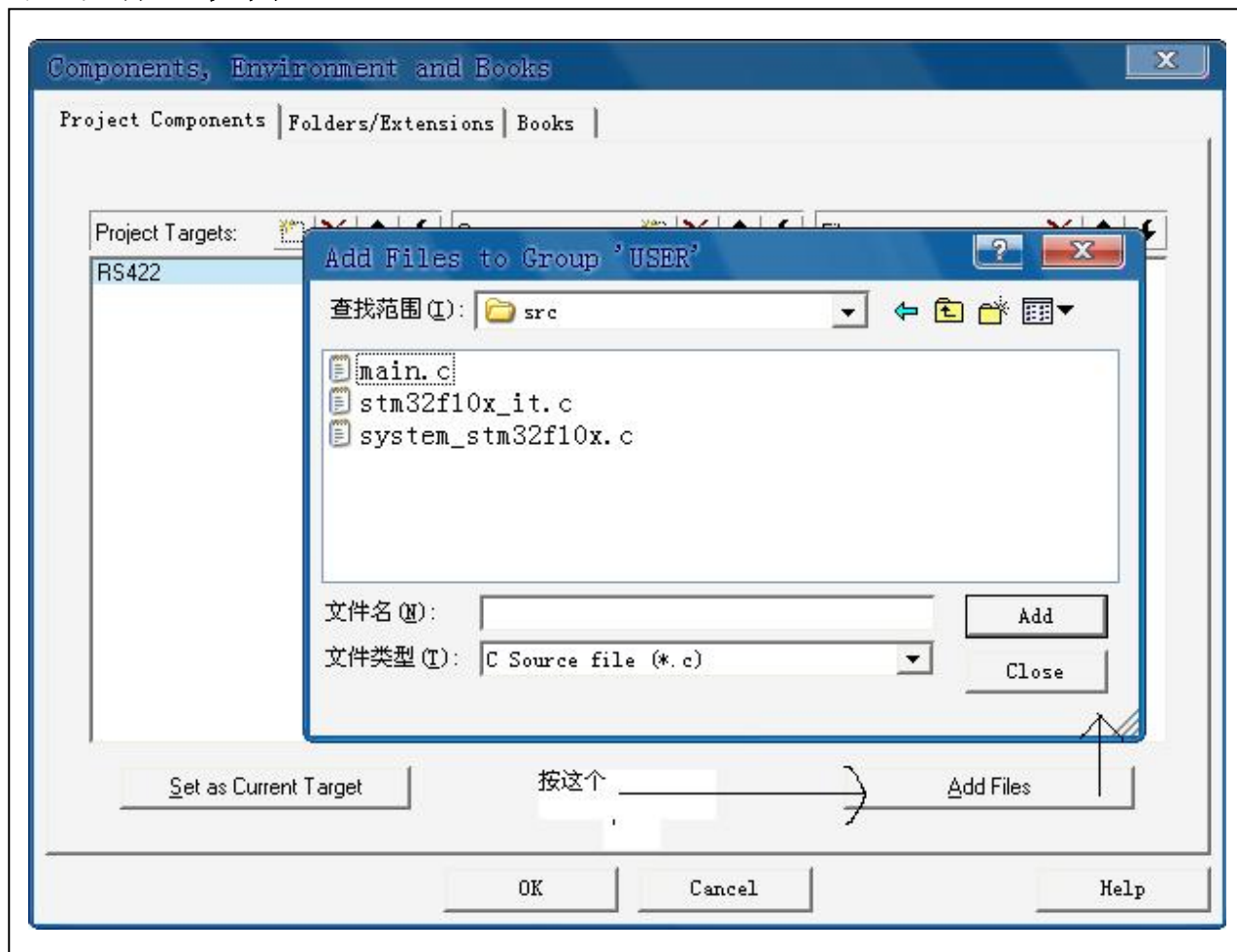


图 6

5.4 在“STM32_LIB”条目的“Files”栏中添加目录“V1” → “Libraries” → “STM32F10x_StdPeriph_Driver” → “SRC”目录下的“stm3210x_misc.c”、“stm3210x_rcc.c”、“stm3210x_gpio.c”、“stm3210x_usart.c”4个文件（见图7），其中“stm3210x_misc.c”、“stm3210x_rcc.c”这2个文件是必须的（我自己的理解）。因为我的程序要用到“GPIO”和“串口”，所以又添加了“stm3210x_gpio.c”、“stm3210x_usart.c”这2个文件，大家在开发中如果用到STM32的其他功能，再添加相应的接口库文件就可以了。

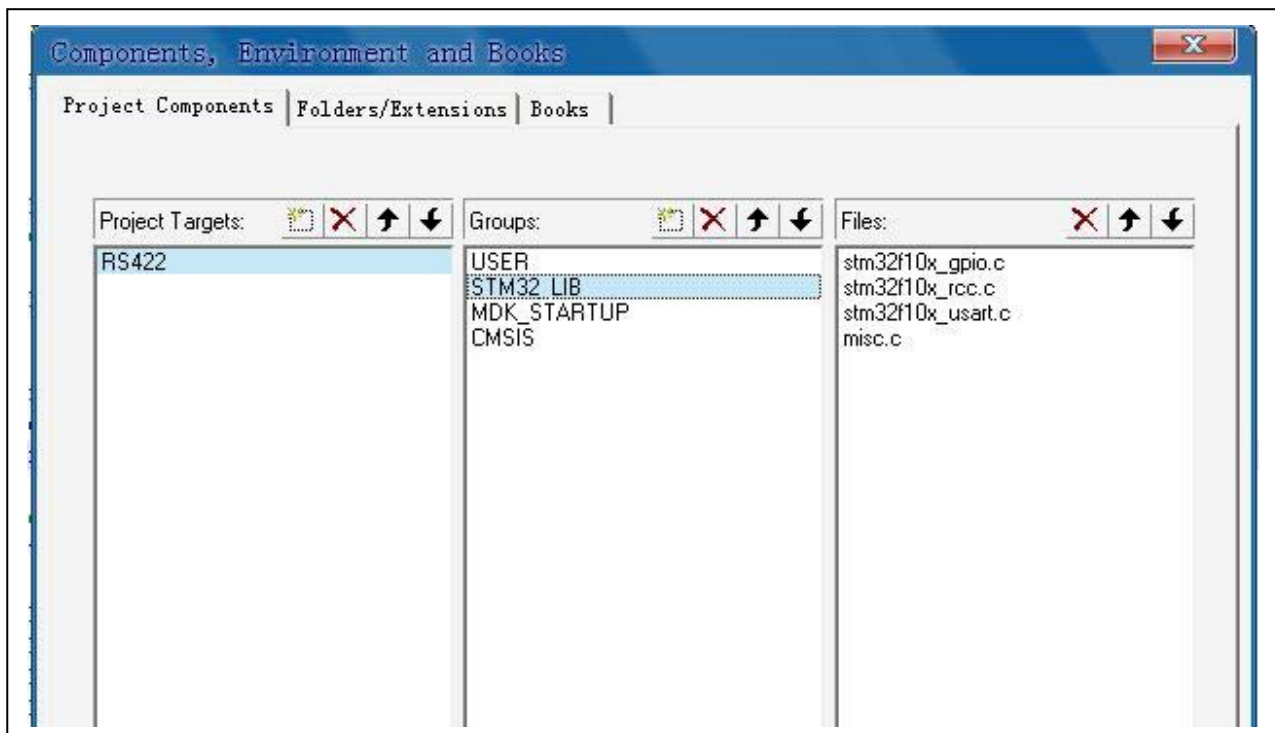


图 7

5.5 在“MDK_STARTUP”条目的“Files”栏中添加目录“V1” → “Libraries” → “CMSIS” → “CM3” → “DeviceSupport” → “ST” → “STM32F10x” → “startup” → “arm”（我的神啊：ST 的目录够深的）下的“startup_stm3210x_hd.s”这个文件（见图 8），因为我选用的 CPU 是“STM32103ZE” 是高容量 FLASH 的 CPU 所以选择这个文件。

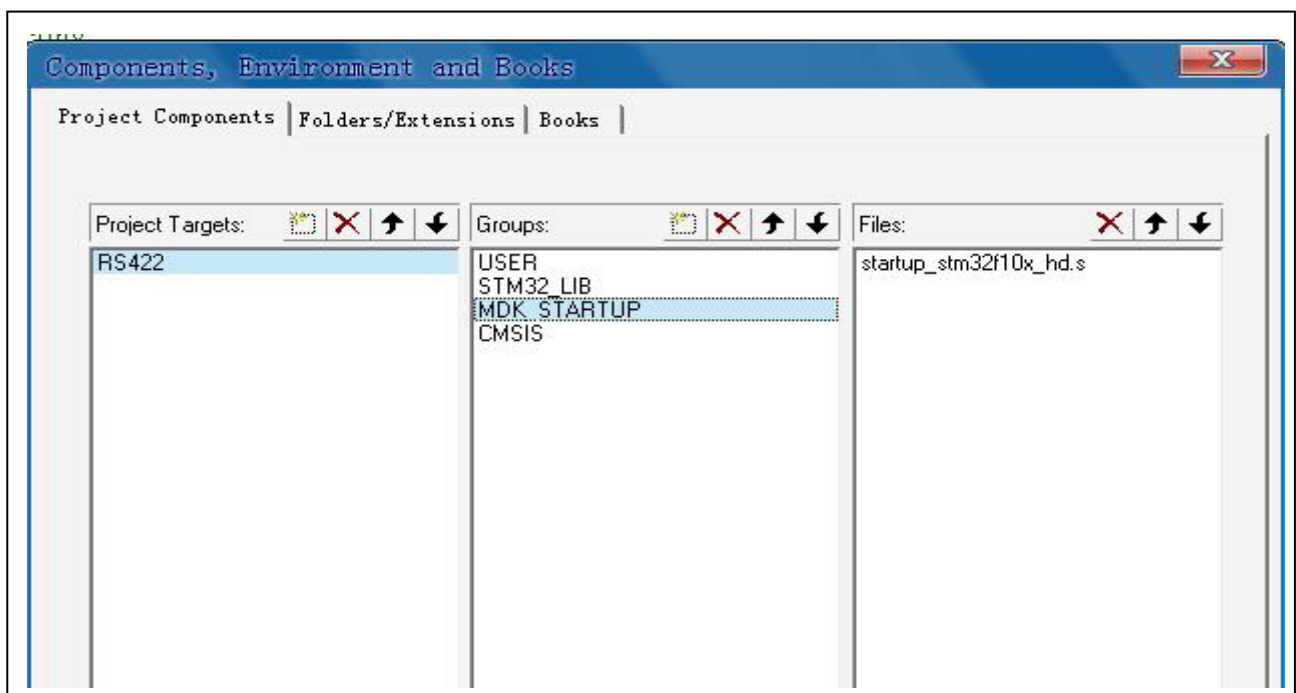


图 8

5.6 在“CMSIS”条目的“Files”栏中添加目录“V1”→“Libraries”→“CMSIS”→“CM3”→“CoreSupport”下的“core_cm3.c”和目录“V1”→“USER”→“SRC”下的“system_stm32f10x.c”这 2 个文件（见图 9）。

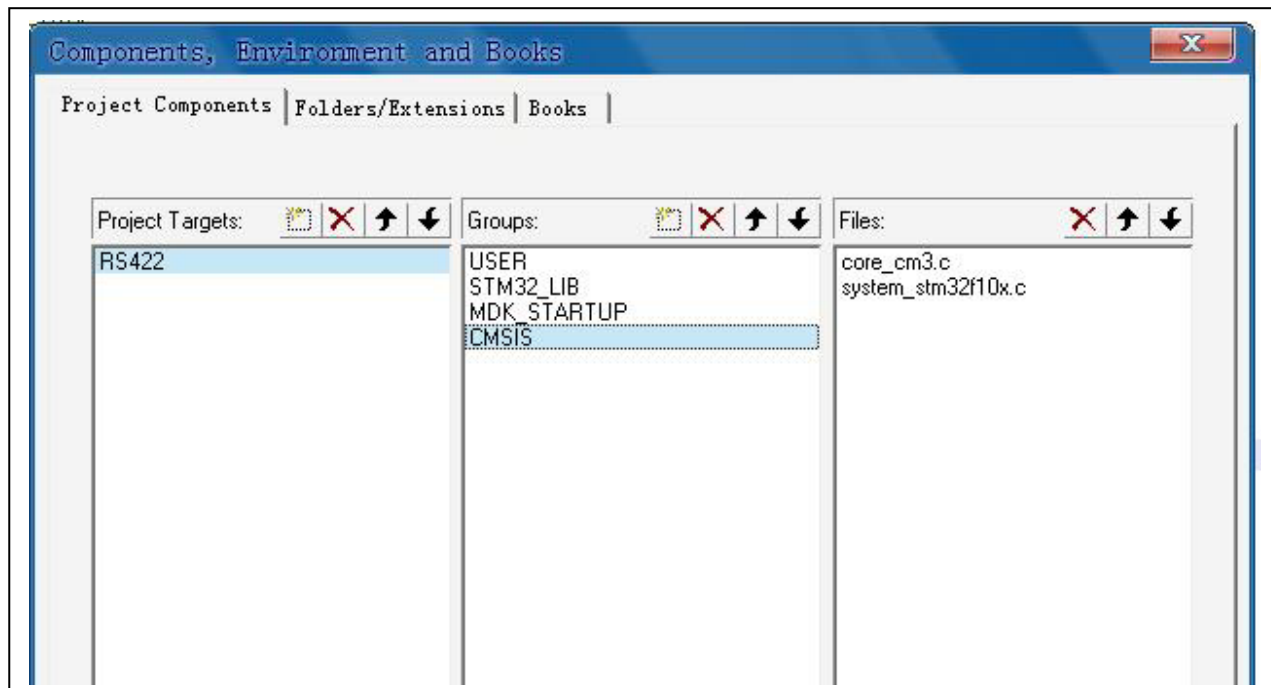


图 9

5.7 退出“Keil uVision4”开发环境，将工程目录“RS422”去掉文件夹的只读属性，并“应用到子目录和所有文件”，这是因为 STM32 固件库下载下来是只读的，无法修改库中的文件。去掉只读属性后再进入“Keil u Vision4”，然后打开工程“RS422_MODULE.uvproj”。

6 工程设置

6.1 在“Project”窗口中用鼠标左键点击最顶层的“RS422”，再点击右键弹出菜单选择菜单中的“Options for Target ‘RS422’...”子菜单（见图 10），出现“Options for Target ‘RS422’”窗口（见图 11），可以按图 11 进行设置。

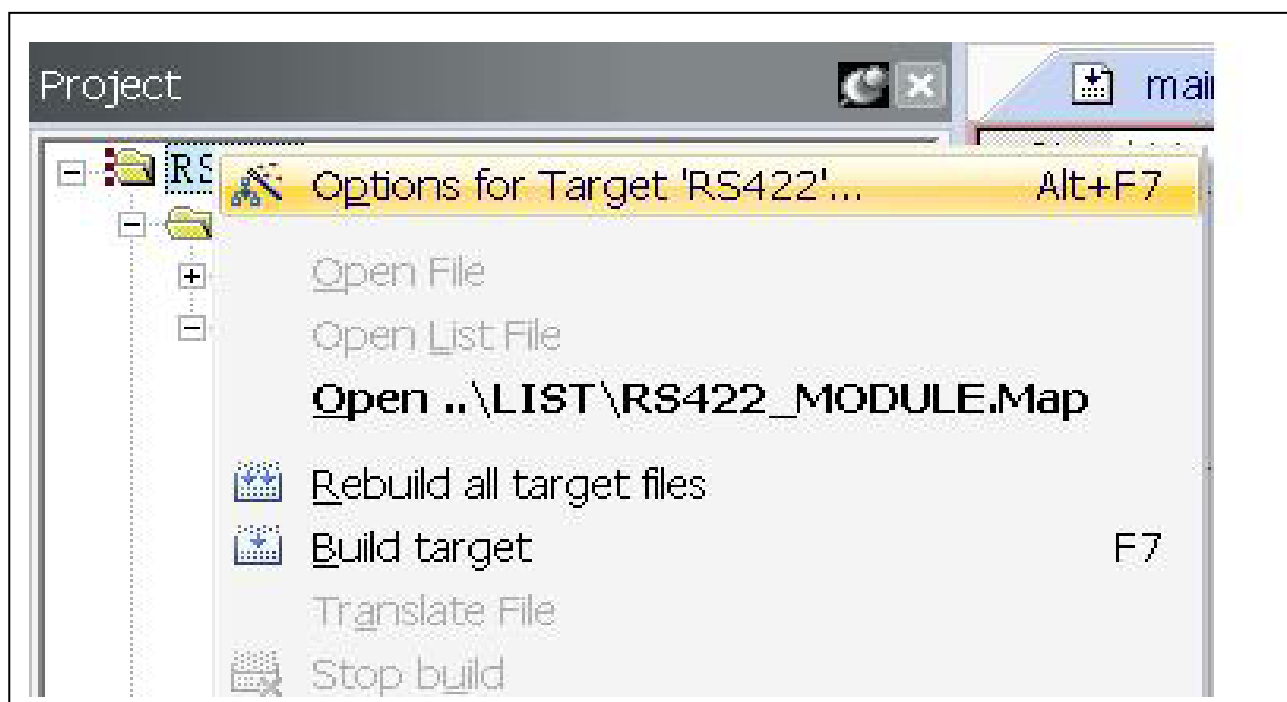


图 10

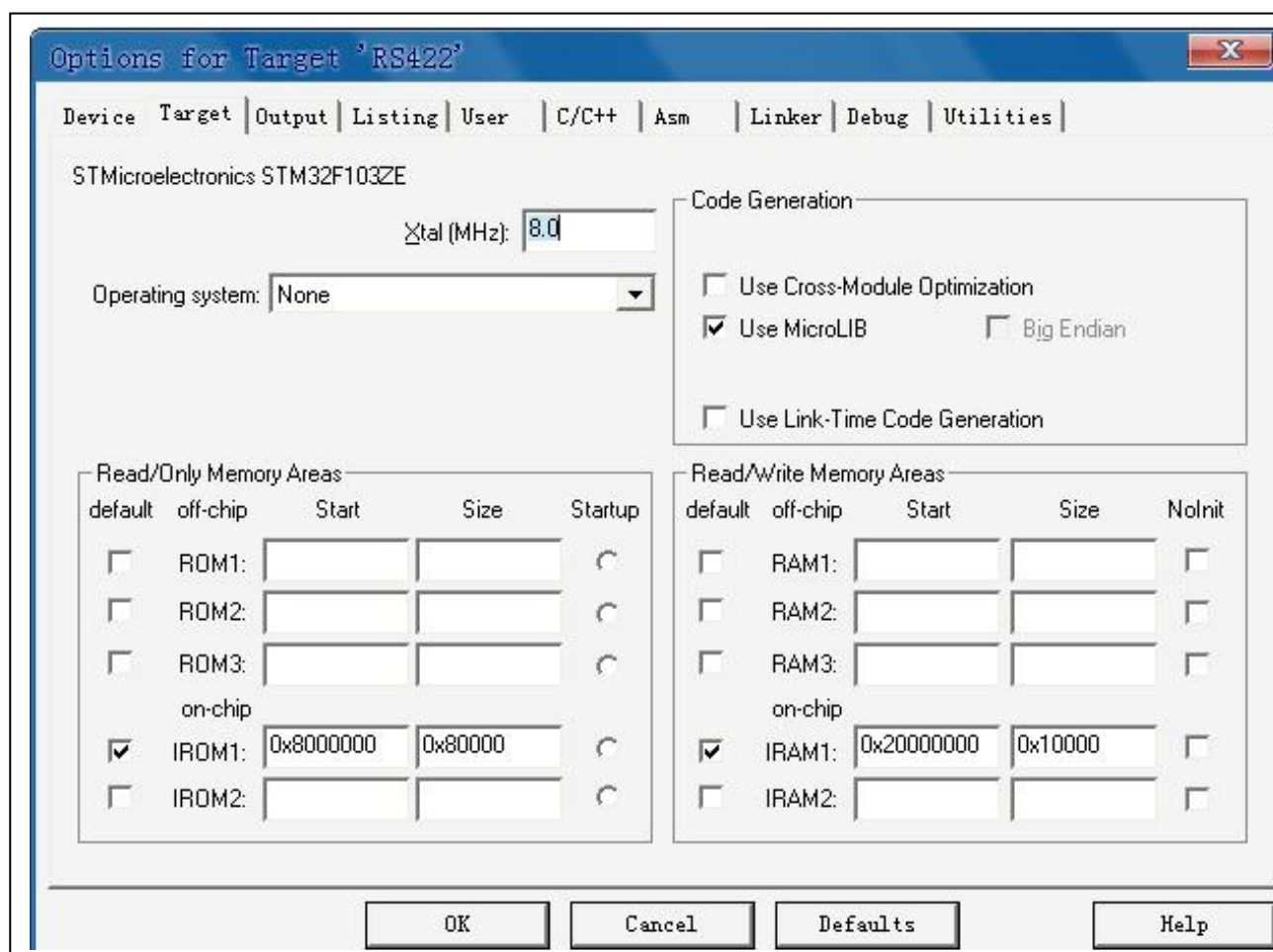


图 11

IROM1、IRAM1 根据芯片不同而不同，一般 **KEIL** 会自动根据芯片填好的

6.2 选择顶层的“Output”，进入“Output”设置页，点击下面的“Select Folder for Objects...”按钮，选择目录“V1” → “Project” → “OBJ”为目标文件目录，选中“Create HEX File”（见图 12）。

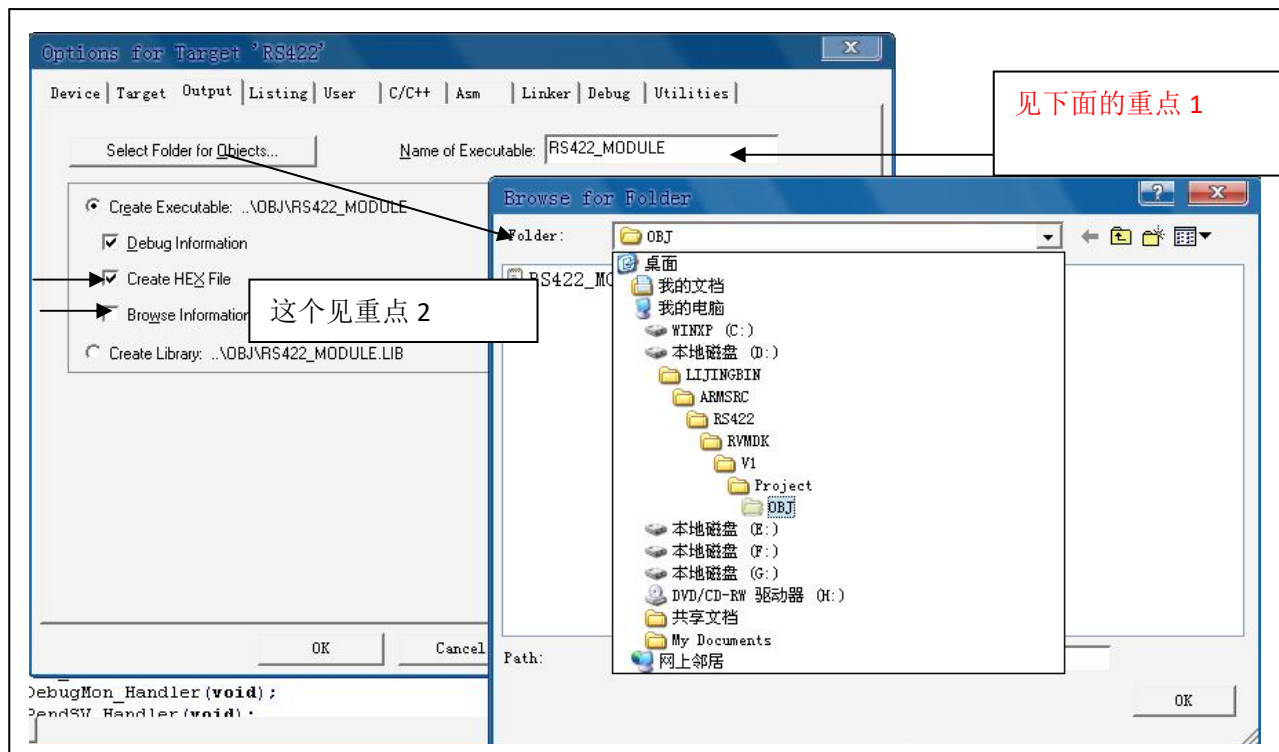


图 12

以下两点很重要:

1. 将上图“Name of Executable”后的文本框中的“RS422_MODULE” 更改为“RS422”,否则用 JLINK 或 ULINK 仿真时会报错,这是经过 1 个上午的奋斗才找见的 KEIL BUG, 我的截图忘了更改了。
2. 选中“Browse information”前的复选框, 我的截图忘了选了, 要不会找不见函数或变量的定义的。

6.3 选择顶层的“Listing”页，进入“Listing”设置页，点击下面的“Select Folder for Listings...”按钮，选择目录“V1”→“Project”→“LIST”目录为list文件生成目录（见图13）。

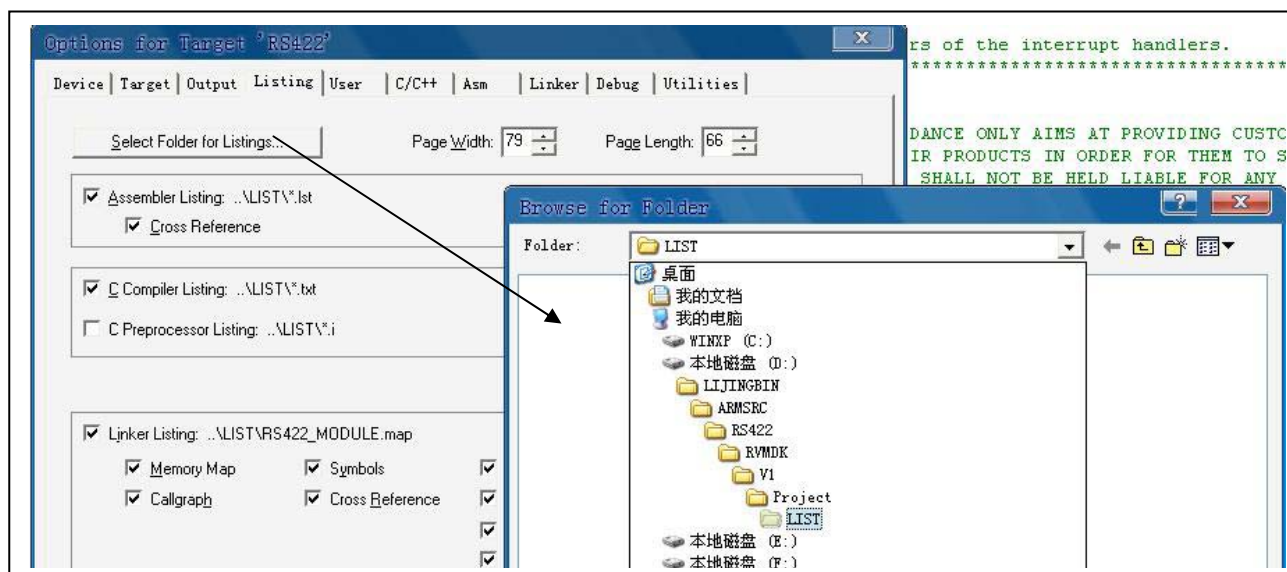
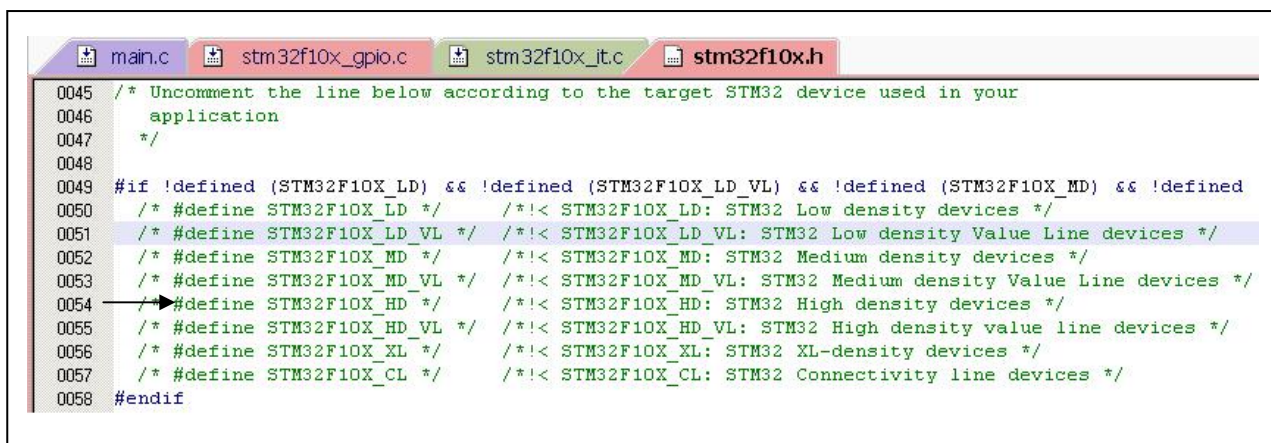


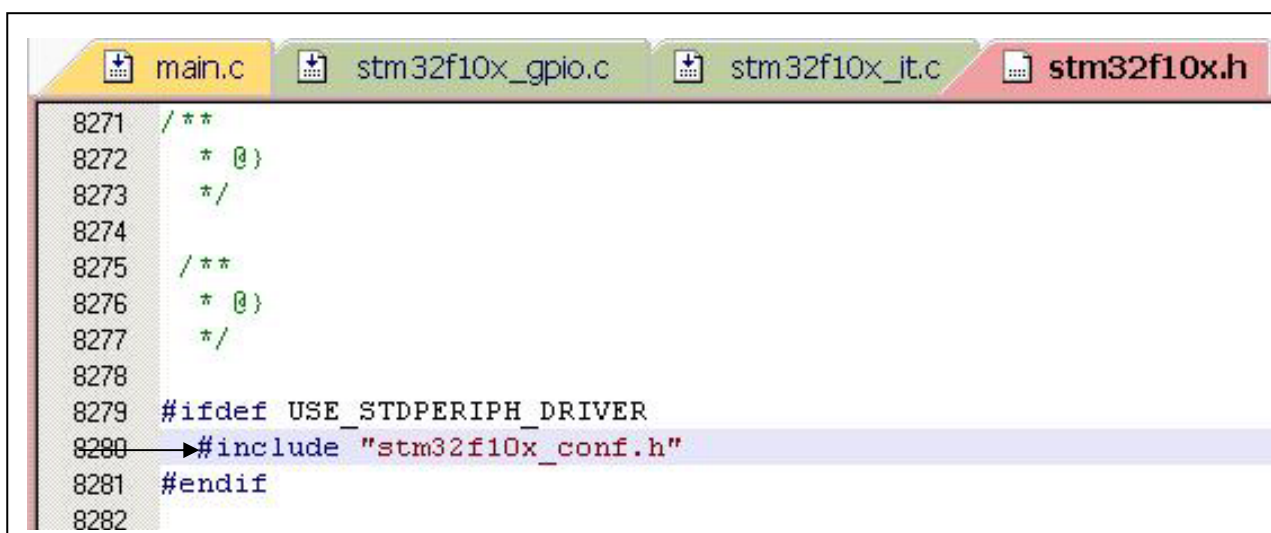
图 13

6.4 这是关键的一步，选择顶层的“C/C++”页，进入“C/C++”设置页，在“Preprocessor Symbols”的“Define:”文本框中填入“STM32F10X_HD,USE_STDPERIPH_DRIVER”。如果不填编译会报错，大家可以试一下，这是因为在固件库“stm32f10x.h”中有如下的片断（见图14），需要根据你的CPU来去掉相应行的注释，我选的是STM32系列高容量的CPU，所以需要去掉“stm32f10x.h”中第“0054”行“/*define STM32F10X_HD */”这一行的注释符，但是不能换一个CPU就改一次注释吧，老去改文件自己哪天不定就改的迷糊了，幸好编译器提供了这个功能，只要按本步骤的方法加入“STM32F10X_HD”就可以了，省得自己换了CPU还得把文件找出来更改。加入“USE_STDPERIPH_DRIVER”是同样的道理，这个出现在“stm32f10x.h”的第“8280”行（见图15），但是固件库中并未定义“USE_STDPERIPH_DRIVER”，所以也需要用这种办法灵活的加入，否则编译时就不会链接“stm32f10x_conf.h”，这个文件可是大有用处，打开看看吧，这里就不描述了。



```
0045 /* Uncomment the line below according to the target STM32 device used in your
0046 application
0047 */
0048
0049 #if !defined (STM32F10X_LD) && !defined (STM32F10X_LD_VL) && !defined (STM32F10X_MD) && !defined
0050 /* #define STM32F10X_LD */ /*!< STM32F10X_LD: STM32 Low density devices */
0051 /* #define STM32F10X_LD_VL */ /*!< STM32F10X_LD_VL: STM32 Low density Value Line devices */
0052 /* #define STM32F10X_MD */ /*!< STM32F10X_MD: STM32 Medium density devices */
0053 /* #define STM32F10X_MD_VL */ /*!< STM32F10X_MD_VL: STM32 Medium density Value Line devices */
0054 /* #define STM32F10X_HD */ /*!< STM32F10X_HD: STM32 High density devices */
0055 /* #define STM32F10X_HD_VL */ /*!< STM32F10X_HD_VL: STM32 High density value line devices */
0056 /* #define STM32F10X_XL */ /*!< STM32F10X_XL: STM32 XL-density devices */
0057 /* #define STM32F10X_CL */ /*!< STM32F10X_CL: STM32 Connectivity line devices */
0058 #endif
```

图 14



```
8271 /**
8272 * @}
8273 */
8274
8275 /**
8276 * @}
8277 */
8278
8279 #ifndef USE_STDPERIPH_DRIVER
8280 #include "stm32f10x_conf.h"
8281 #endif
8282
```

图 15

6.5 引用固件库文件所在的目录也在顶层的“C/C++”页中进行设置，上面写的啰里啰唆的，已经太多了，所以写在这里吧。如果现在编译程序，会报错的，看看出错提示，有一些库文件跑到开发工具 Keil 的安装目录下链接去了，这是因为没有设置 STM32 固件库的目录，编译器就默认到“Keil”根目录下的某某目录找去了。我们工程用到的 STM32 3.4.0 固件库的相关文件在上面第 3 节中已经直接拷贝到工程里了，这个是因为 STM32 库升级到 3.4 已经经历了好多个版本了，而安装完 KEIL 后并不是最新的 3.4 库，不能说哪天系统 OVER 了，装一次 KEIL 就升级一次库吧，干脆就把库文件直接放在工程中，即使将工程拷贝到其他机器上也可以编译，不会因为缺少库文件而报错，唉！又啰嗦了这么多。在窗口的“Include Paths”旁边的文本框后有一个按钮，点击调出“Folder Setup”窗口。这里要添加 4 个目录（见图 16），因为这 4 个目录里都有“.h”文件：

- “ RS422\RVMDK\V1\Libraries\STM32F10x_StdPeriph_Driver\inc ” ；
- “ RS422\RVMDK\V1\Libraries\CMSIS\CM3\CoreSupport ” ；

“ RS422\RVMDK\V1\Libraries\CMSIS\CM3\DeviceSupport\ST\STM32F10x ” ；
“ RS422\RVMDK\V1\USER\inc ” ；

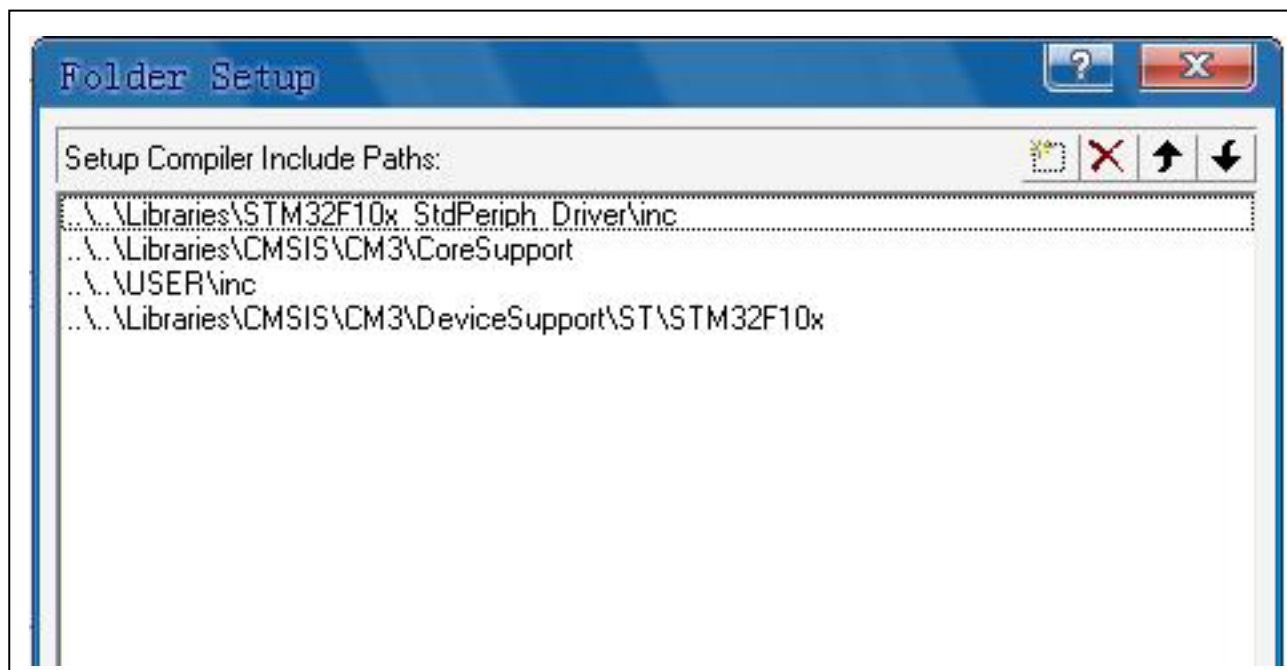


图 16

6.6 其他“C/C++”按图 17 设置默认就可以了。

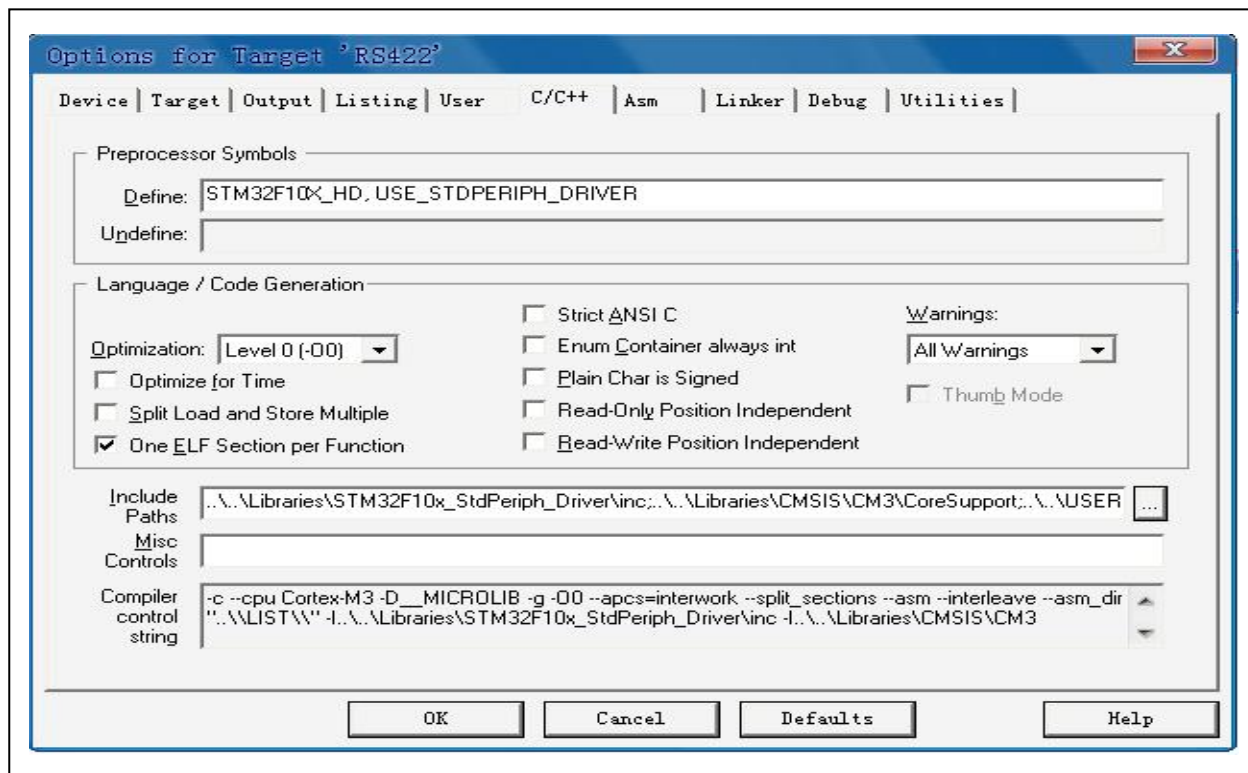


图 17

7. main.c 文件

因为这是一个没有任何功能的工程框架，所以 **main.c** 可以这么写：

```
#include "stm32f10x.h"

main ()
{
    while (1)
    {
        ;
    }
}
```

编译程序，没有警告和错误，框架就算生成了，想添加自己的代码就可以以后添加了，自己的代码放在工程目录“**RS422**” → “**RVMDK**” → “**V1**” → “**USER**” 下的 “**INC**” 或 “**SRC**” 目录下，“**INC**” 下放你的 “.h” 文件，“**SRC**” 目录下放你的 “.c” 文件。

以上是本人对 **STM32** 工程框架的实例实现，如果路过的有更好的，希望可以交流。