










MG32F10x 开发入门

1 固件库介绍

下图所示即为 MG32F10x 标准固件库的目录结构。

- ▼  MG32F10x_StdPeriph_Lib_V0.1.5
 - ▼  Libraries
 - >  CMSIS
 - >  MG32F10x_StdPeriph_Driver
 - >  MG32F10x_USBDevice_Driver
 - ▼  Project
 - >  MG32F10x_StdPeriph_Examples
 - >  MG32F10x_StdPeriph_Template
 - >  Utilities

Documentation 目录中存放了 MG32F10x 固件库的说明文档。

Libraries 目录中包含 **CMSIS**, **MG32F10x_StdPeriph_Driver**, **MG32F10x_USBDevice_Driver** 三个子目录。其中 **CMSIS** 目录中存放了启动文件, 芯片头文件等; **MG32F10x_StdPeriph_Driver** 目录中存放了 MG32F10x 固件库源码文件; **MG32F10x_USBDevice_Driver** 目录中存放了 MG32F10x USB 设备协议栈代码。

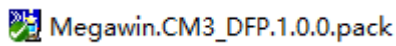
Project 目录中包含 **MG32F10x_StdPeriph_Examples** 和 **MG32F10x_StdPeriph_Template** 两个子目录。其中 **MG32F10x_StdPeriph_Examples** 目录中存放了 megawin 官方提供的固件示例源码, 以便客户参考; **MG32F10x_StdPeriph_Template** 目录中存放了创建工程所需要的文件模板。

Utilities 目录中存放了一些公用的源码。

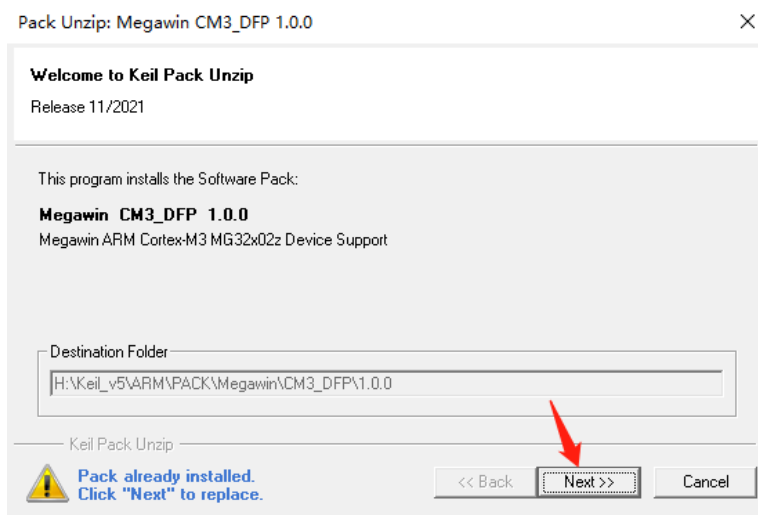
2 使用 Keil MDK 建立工程

1) 安装M3数据包。

打开Megawin_CM3_DFP.1.0.0.pack。

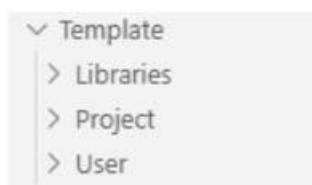


点击Next，并等待Finish即可。



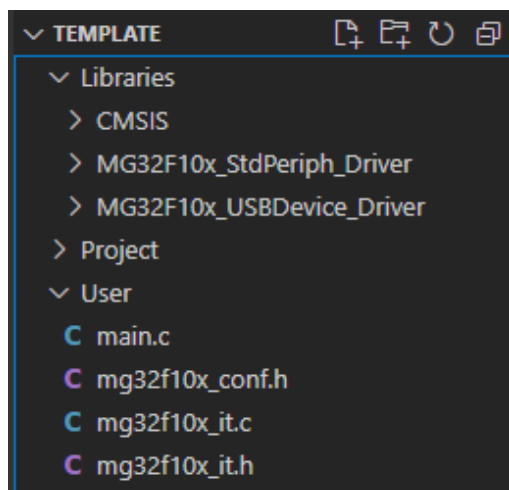
2) 新建一个文件夹命名为 Template 用以存放整个工程。

3) 在 Template 文件夹中新建 Libraries, Project 和 User 三个子文件夹（当然用户可定义自己工程目录结构）

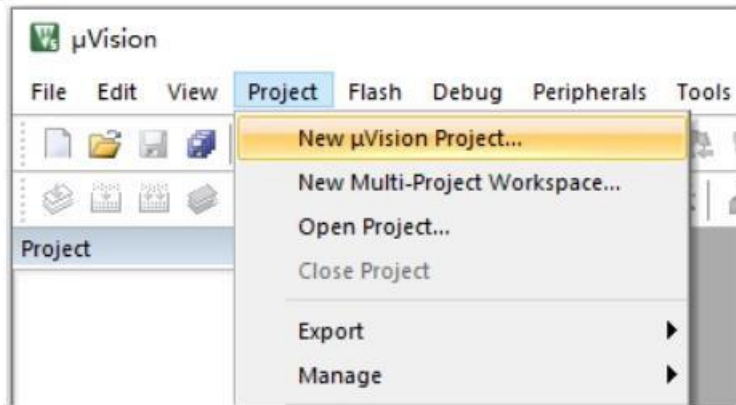


4) 将 MG32F10x 标准固件库中 Libraries 目录中的内容复制到 Template\Libraries 目录中。

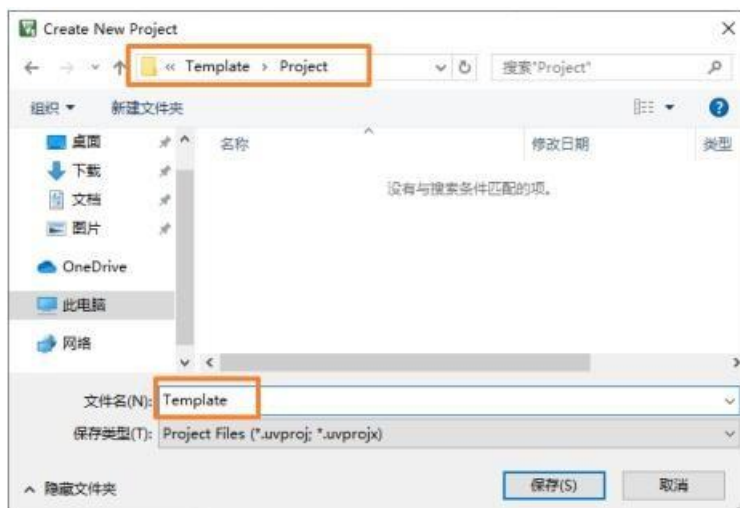
5) 将 MG32F10x 标准固件库中 Project\MG32F10x_StdPeriph_Template 目录中的内容复制到 Template\User 目录中。



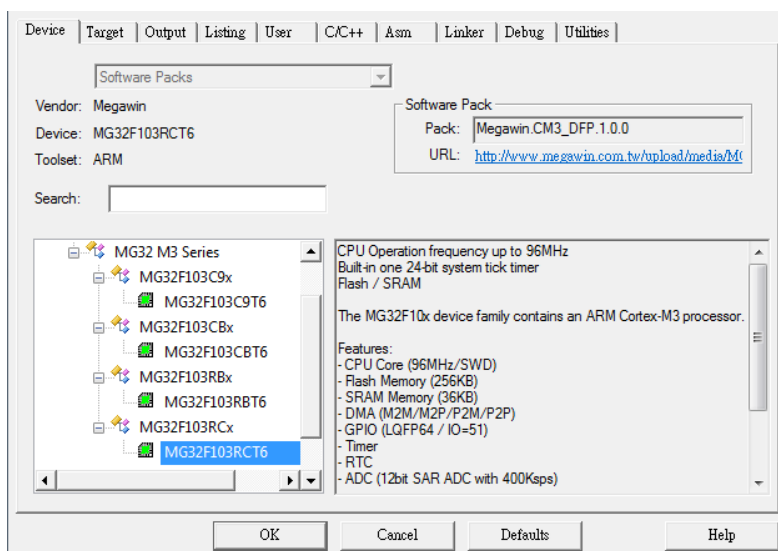
6) 打开 Keil MDK，新建项目。



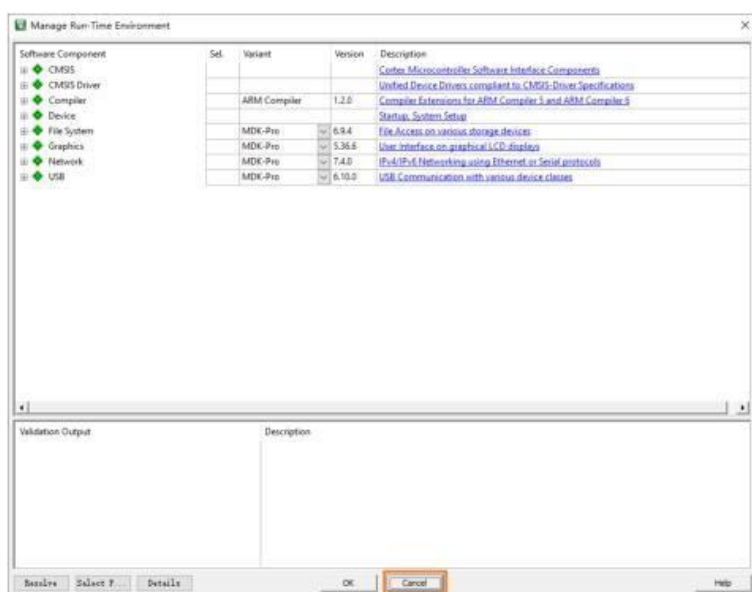
7) 在 Template\Project 目录中新建名为 Template 的工程。



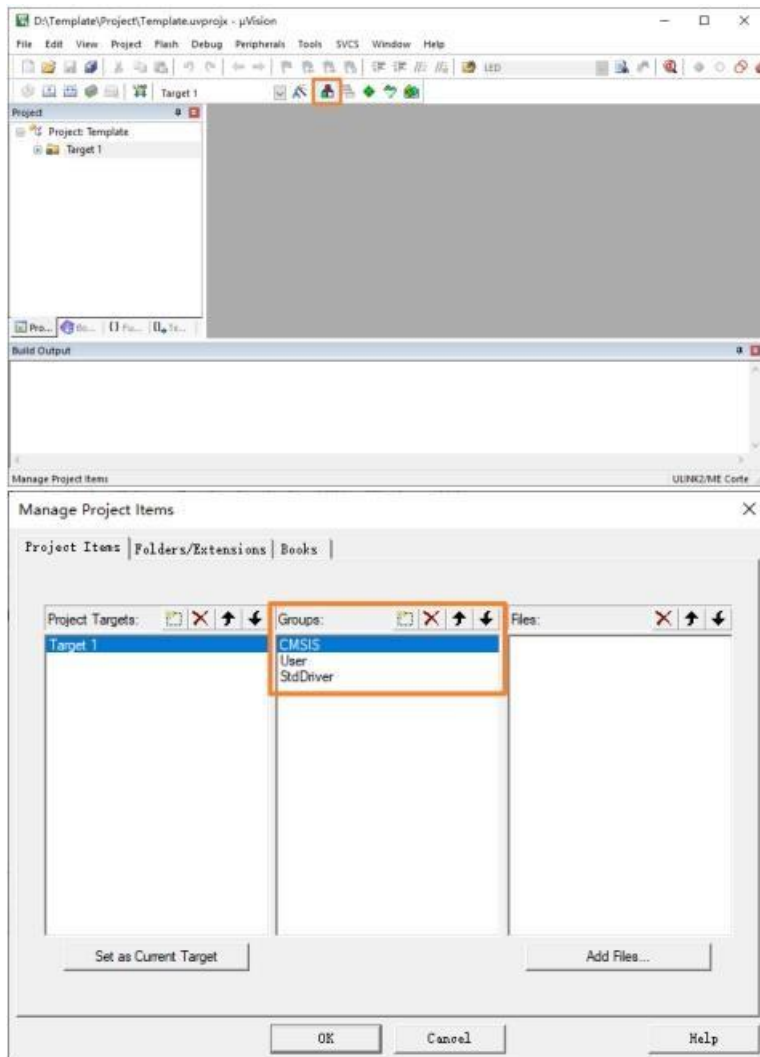
8) 选择项目使用的设备为需要的M3型号，并点击 OK。



9) 此时弹出 Manage Run-Time Environment 对话框，在该对话框上点击 Cancel。



- 10) 在加入固件库文件之前，我们先建立三个 Groups: CMSIS, User, StdDriver。



- 11) 向 Group 里面添加固件库文件。

向 CMSIS Group 中添加：

Template\Libraries\CMSIS\Device\MG\MG32F10x\startup\arm\startup_mg32f10x.s

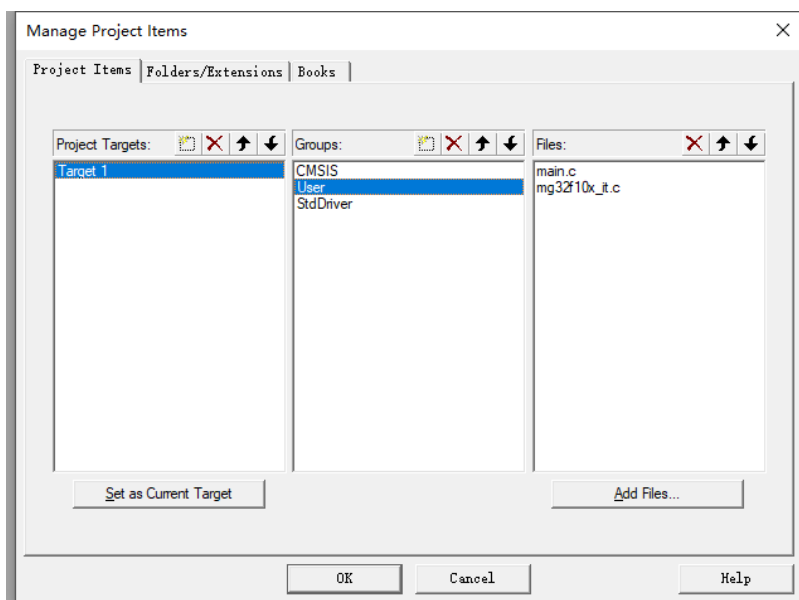
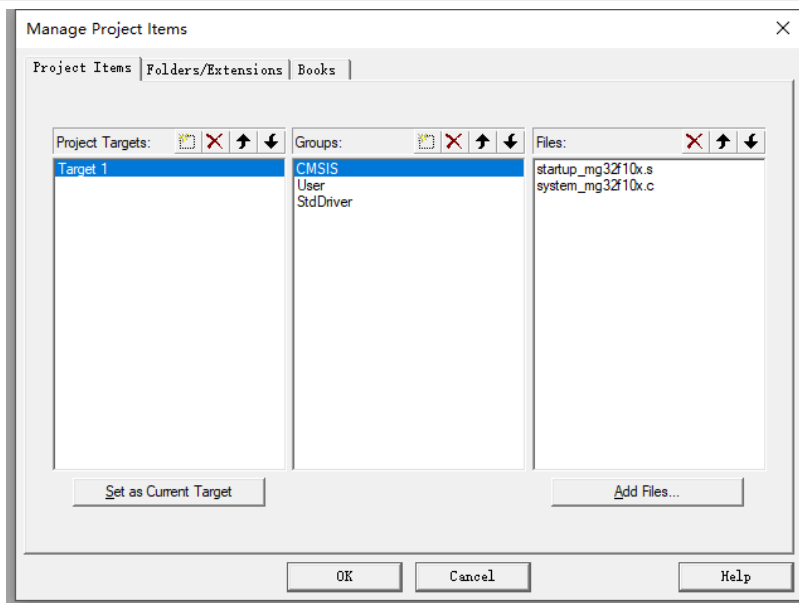
Template\Libraries\CMSIS\Device\MG\MG32F10x\system_mg32f10x.c

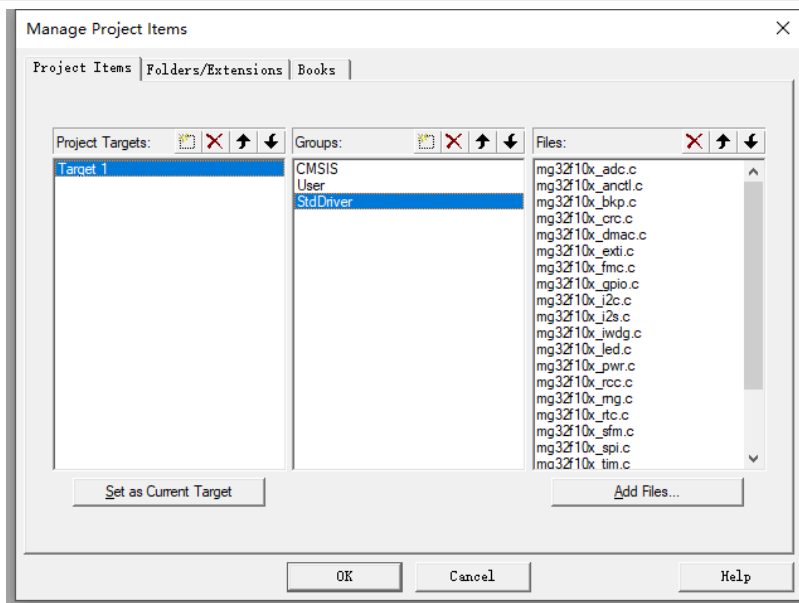
向 User Group 中添加：

Template\User\main.c

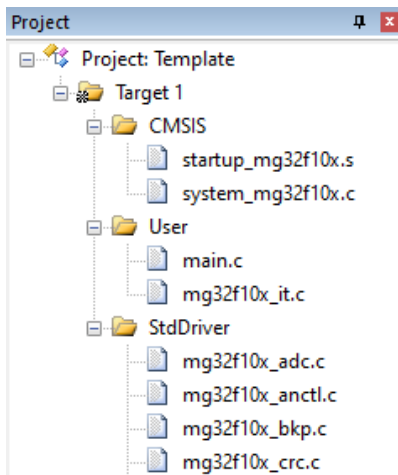
Template\User\mg32f10x_it.c

向 StdDriver Group 中添加 Template\Libraries\MG32F10x_StdPeriph_Driver\src 目录中所有的.c 文件。

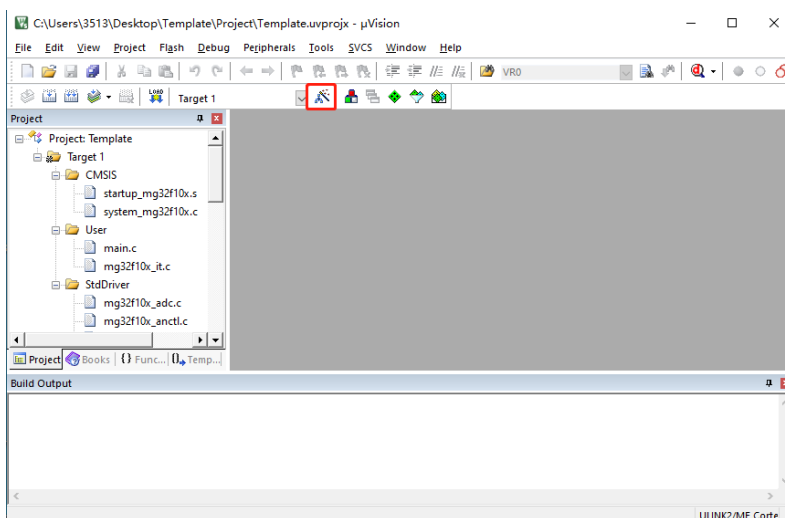




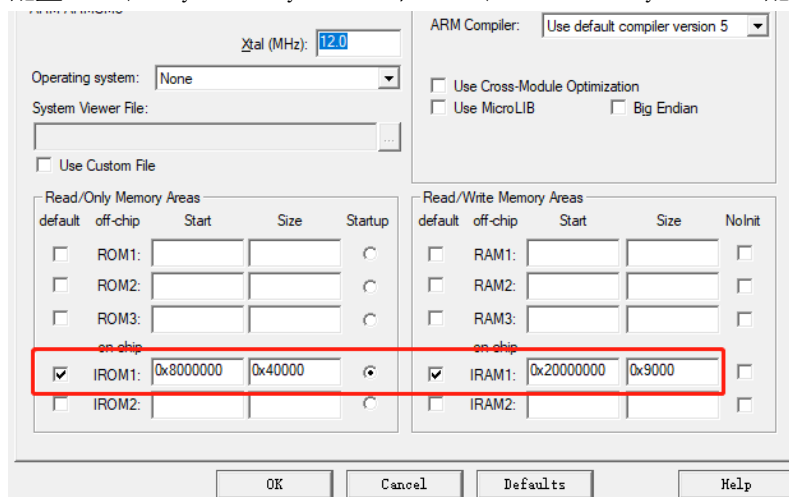
12) 最终项目的结构如下图:



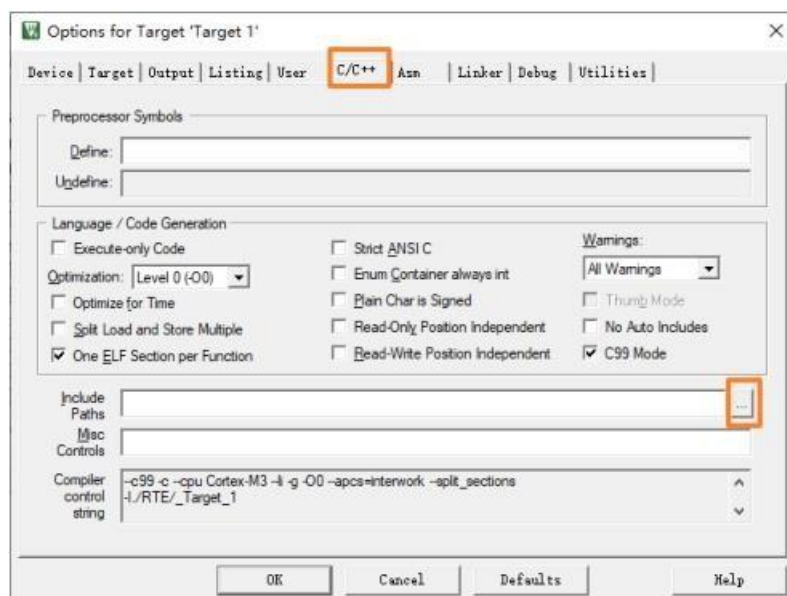
13) 打开 Options for Target 对话框。



14) 配置 Read/Only Memory Areas 和 Read/Write Memory Areas (配置 Flash 和 SRAM 的起始地址和大小)。

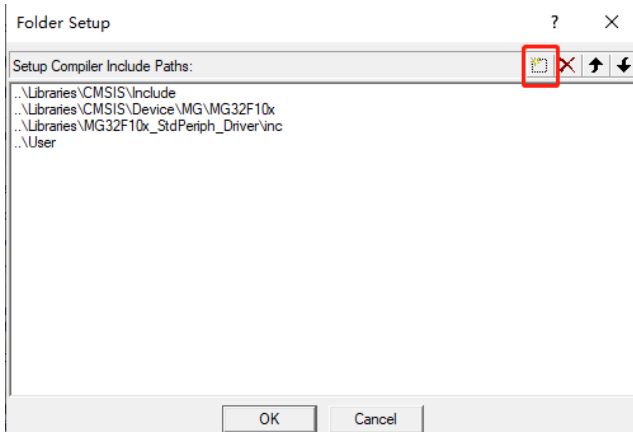


15) 在 C/C++选项卡中配置项目的头文件包含路径。

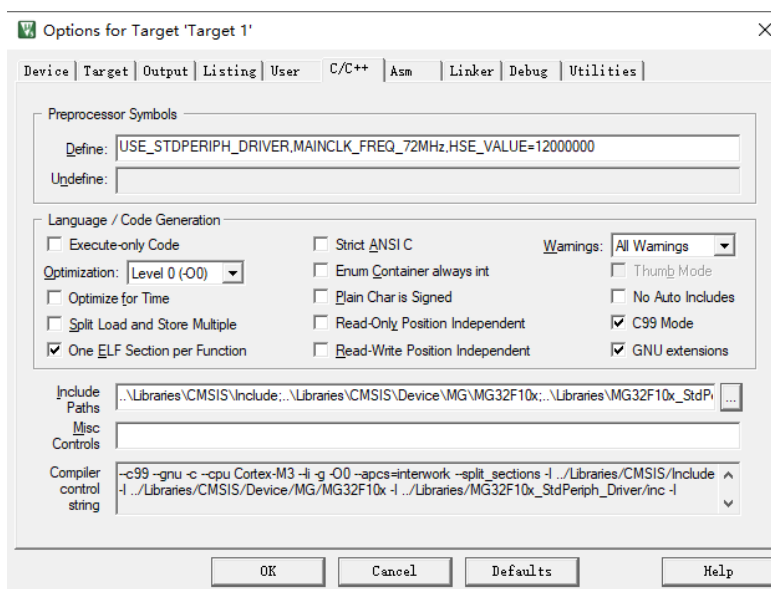


加入以下四个路径：

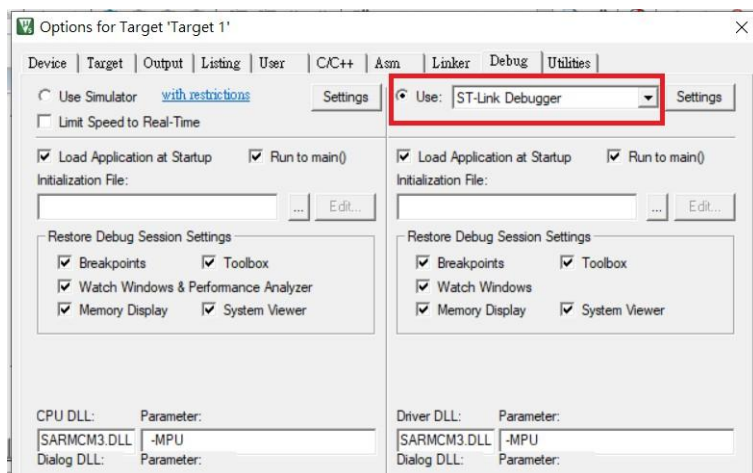
..\Libraries\CMSIS\Include
 ..\Libraries\CMSIS\Device\MG\MG32F10x
 ..\Libraries\MG32F10x StdPeriph Driver\inc
 ..\User



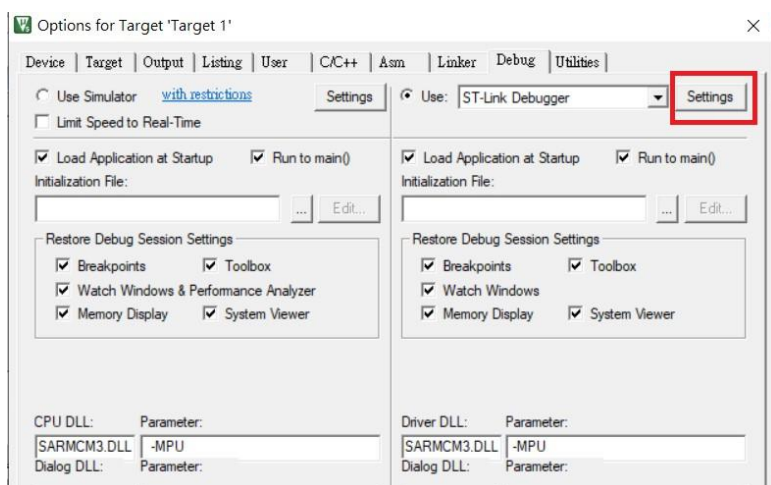
- 16) 在 Preprocessor Symbols 中加入 `USE_STDPERIPH_DRIVER,MAINCLK_FREQ_72MHz` 及 `HSE_VALUE=12000000` 定义。（这两个定义的详情见后文）



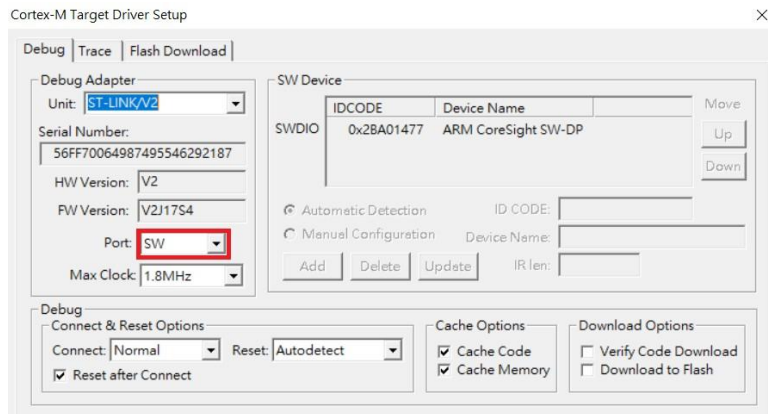
- 17) 点击 OK。至此，项目建立配置完成。接下来进行调试配置。
- 18) MG32F10x是ARM Cortex-M3的芯片，所以可以使用各种支持Cortex-M3的调试器调试程序（比如：JLink，ULink，STLink 和 CMSIS-DAP 等）。下面以 ST-link 为例演示 MG32F10x 的调试配置。若您使用J-Link进行仿真，请参照开发包内Documents目录下的How_to_Use_J_Link_Debug_Download_MG32F10x.pdf
- 19) 将 ST-Link 连接到电脑，使用 ST-link 的 SWD 接口与 MG32F10x 芯片连接，并给芯片上电。
- 20) 打开 Options for Target 对话框，切换到 Debug 选项卡，选择使用 ST-Link 调试器。



20) 配置调试器选项。

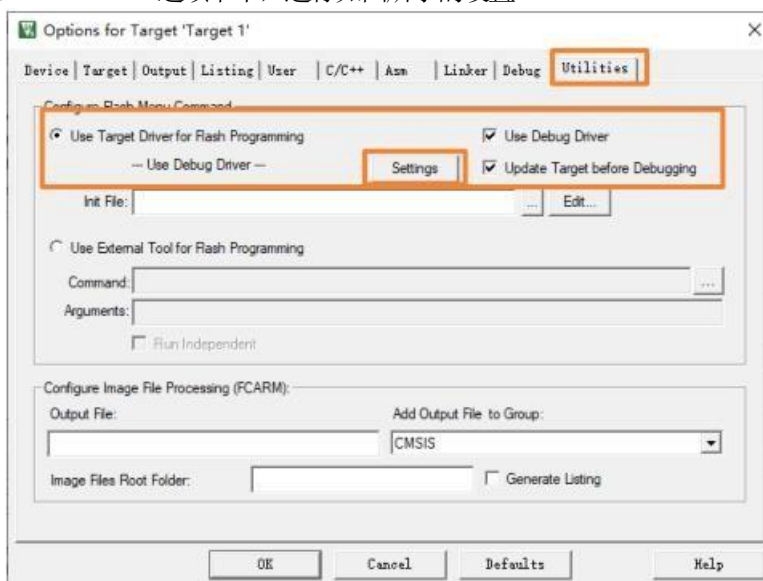


选择 SW 接口，可以在右侧看到 ST-Link 检测到了 MG32F10x 芯片。

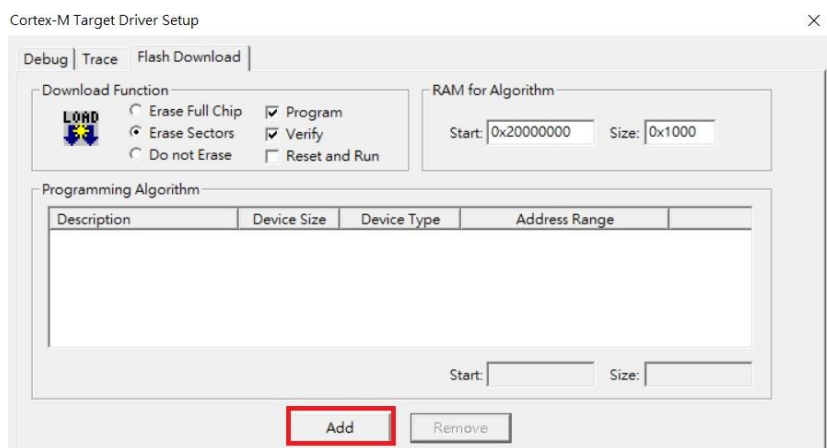
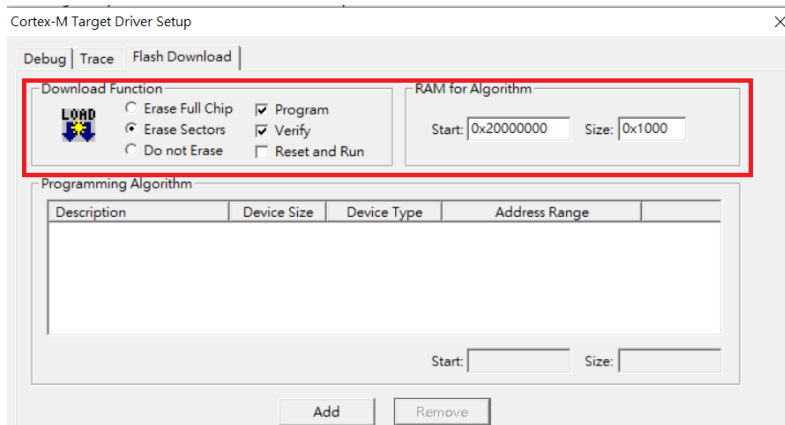


然后点击确定。

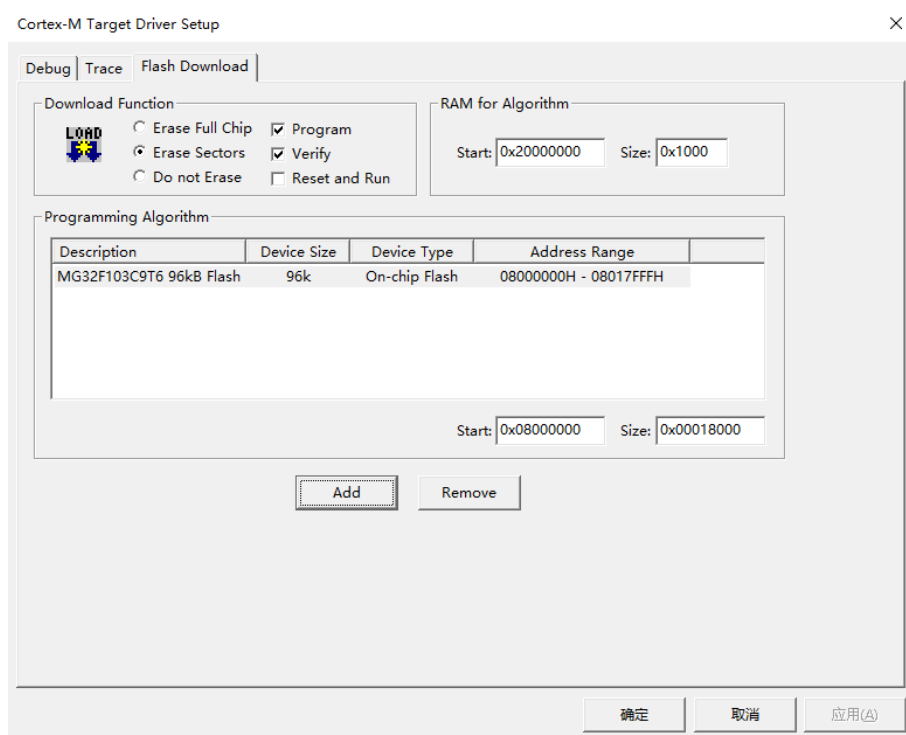
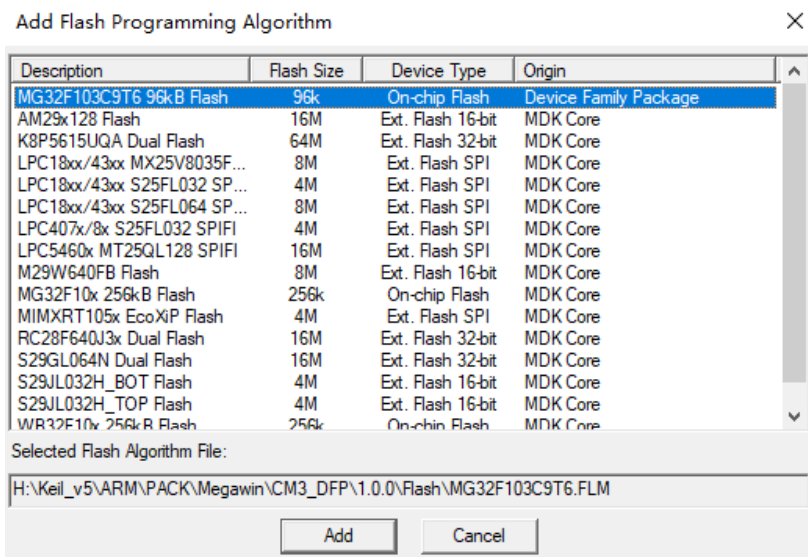
21) 在 Utilities 选项卡中, 进行如图所示的设置。



然后点击 Settings 按钮, 打开烧录算法配置对话框, 进行如图所示的配置。

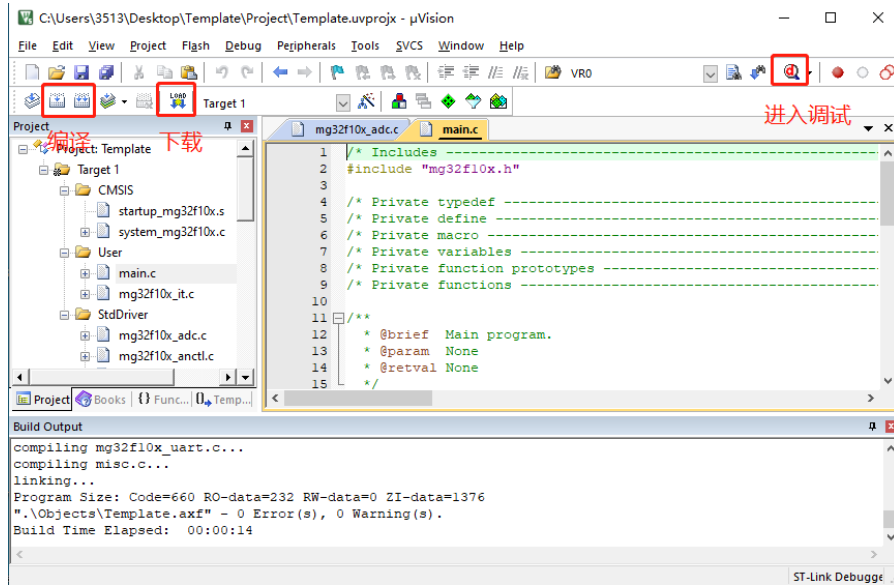


找到对应型号和封装的烧录算法，并点击 Add。



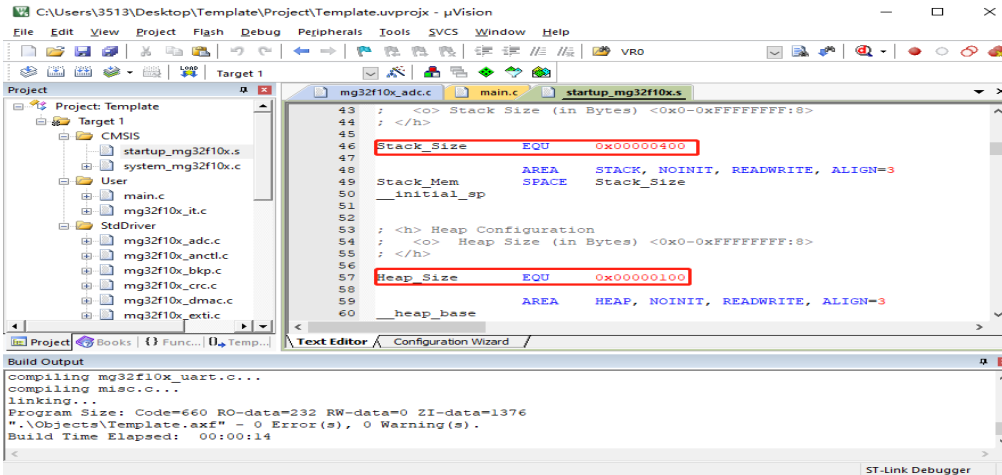
最后点击确定。

22) 至此，用户可以编译，下载和调试该程序了。固件库有关配置详见下节。

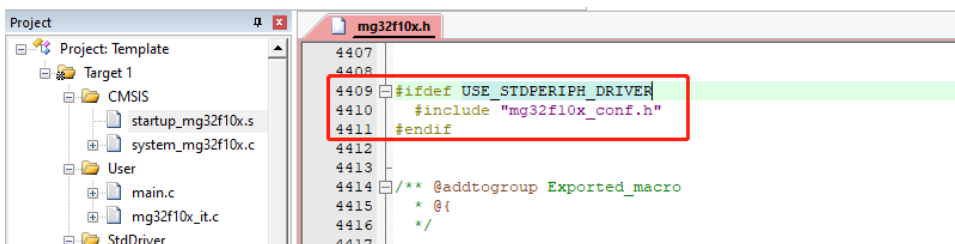
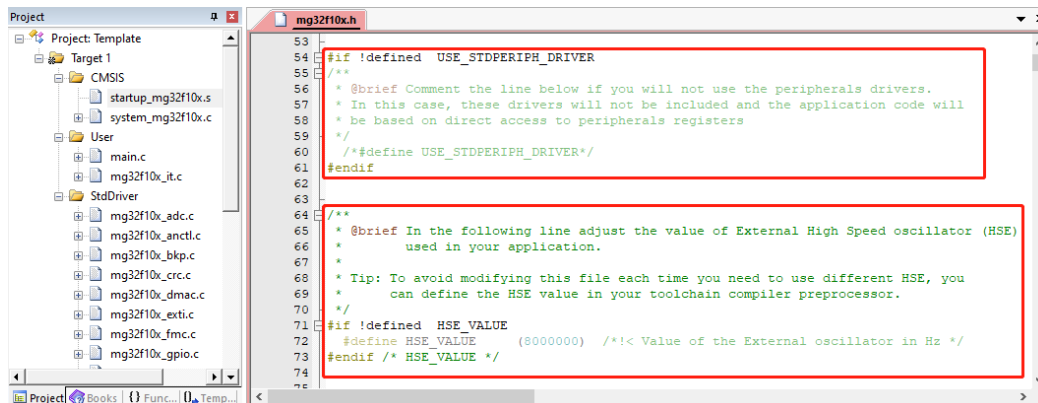


3 固件库配置详解

1) 在 startup_mg32f10x.s 可配置应用程序栈和堆的大小，如下图：



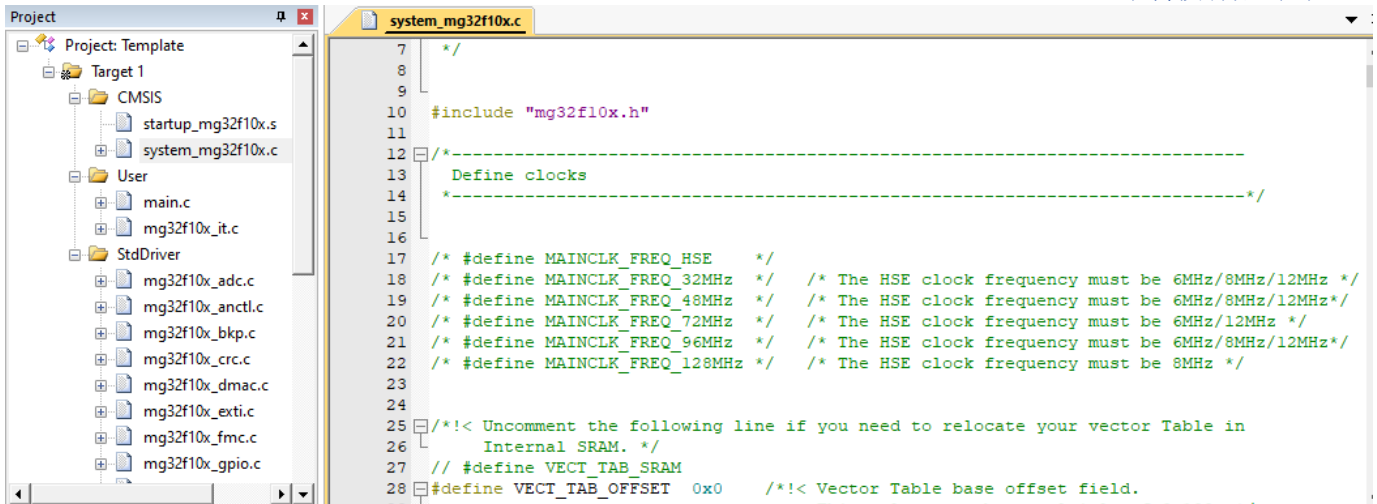
2) 在 mg32f10x.h 中存在两个宏定义，需要用户关注。



USE_STDPERIPH_DRIVER 定义这个宏表示应用程序需要使用固件库中的外设驱动，且会在项目中包含 mg32f10x_conf.h 头文件。

HSE_VALUE 该宏用于指定 MG32F10x 芯片外接晶振的频率。默认情况下，固件库假定外部 HSE 晶振的频率是 8MHz。如果用户外接晶振不是 **8MHz**，务必修改或在编译器全局预定义处覆盖该定义！！

3) 在 system_mg32f10x.c 中有几处定义需要用户关注。



MAINCLK_FREQ_* 这些宏定义用以配置程序在启动后芯片主时钟的频率。只能选择定义其中的一个（如果不定义任何一个，那么芯片主时钟是 **MHSI**）。可以在编译器全局预定义处给出定义。这些宏定义对芯片外部晶振是有要求，比如要定义 **MAINCLK_FREQ_72MHz**，那么芯片外部晶振频率必须是 **12MHz**（切记：也要覆盖 **HSE_VALUE** 的定义）。

VECT_TAB_SRAM 定义这个宏表示将中断向量表映射到 **SRAM** 中（在 **SRAM** 中运行的工程才需要定义这个宏）。

VECT_TAB_OFFSET 该宏用以设置中断向量表起始地址偏移（相对于 **Flash** 或 **SRAM** 的起始地址）。