



芯海科技

CHIPSEA

股票代码:688595

CSU-IDE32 用户手册

涉密等级：公开



芯海科技(深圳)股份有限公司

www.chipsea.com

+86-0755-8616 9257

sales@chipsea.com

518000

目 录

版本历史.....	4
1 欢迎使用.....	5
1.1 功能总述.....	5
1.2 表达约定.....	5
2 运行环境.....	6
2.1 系统环境.....	6
2.3 仿真器.....	6
1、支持 JLINK 仿真器.....	6
2、支持 CSLINK 仿真器.....	6
3 主界面.....	6
3.1 主界面.....	6
3.2 设置窗口状态.....	8
4 快速入门.....	9
5 工程管理.....	13
5.1 关于解决方案.....	13
5.2 创建解决方案.....	14
5.3 添加工程.....	16
5.4 添加文件.....	17
5.5 添加文件夹.....	18
5.6 解决方案管理器（SOLUTION EXPLORER）.....	19
5.7 工程设置（PROJECT SETTINGS）.....	22
5.8 编译解决方案或工程（BUILD）.....	29
5.9 输出窗口（OUTPUT WINDOW）.....	30
5.10 .LIK 文件.....	31
5.11 CLASS VIEW 窗口.....	31
5.12 INCLUDE/EXCLUDE IN TARGET BUILD.....	33
6 文本编辑.....	34
6.1 编辑区窗口.....	34
7 调试器.....	42
7.1 下载程序.....	42
7.2 调试操作.....	42
7.3 调试窗口（DEBUG WINDOW）.....	43
8 用户自定义.....	52
9 菜单项介绍.....	58
10 工具栏介绍.....	65
11 LIBRARY PROJECT.....	67

12 CMAKE PROJECT	70
12.1 创建工程.....	70
12.2 执行编译命令.....	72
13 INSTALL CMSIS PACKAGE.....	73
14 CMSIS WIZARD.....	74
15 常用技巧.....	75

版本历史

历史版本	修改内容	时间	修改人
V1.0	第一版正式版本	2022-04-14	

1 欢迎使用

CSU-IDE32 是由芯海科技（深圳）股份有限公司开发的一套 32 位 ARM 集成开发工具。

CSU-IDE32 使用 GCC 编译器、汇编器、链接器、调试器、以及新增 lib 制作工具。它支持 CMSIS 标准型号包和向导，并支持用户自定义 GCC 工具链、Makefile、链接脚本。CSU-IDE32 提供了强大的工程管理、编辑、设置、查找以及调试等功能，以及更加友好的界面。

1.1 功能总述

序号	功能描述
1	支持解决方案的概念，提供了强大的项目管理功能和编辑、搜索功能
2	加强编辑器功能，提供智能感应和自动完成功能
3	加强调试器功能，提供全面的调试窗口，如 RAM/Watch/Register/CallStack
4	增加断点窗口、书签窗口，并能持久化断点、书签的设置
5	增加工具链的设置选项，使编译过程更符合用户使用需求
6	支持 lib 工程，能够快速高效的生成库文件
7	简易的编译、汇编、链接选项配置
8	

1.2 表达约定

在使用手册中，有如下约定的表达方式：

➤ 菜单操作

Project | Open Solution：表示从工程菜单栏 Project 中选择 Open Solution 命令。

➤ 鼠标操作

单击一般指单击鼠标左键，双击一般指双击鼠标左键。

➤ 窗口和对话框

“打开”对话框代表该对话框的名字为“打开”。

2 运行环境

2.1 系统环境

支持的操作系统：XP、WIN7、WIN8、WIN10。

2.3 仿真器

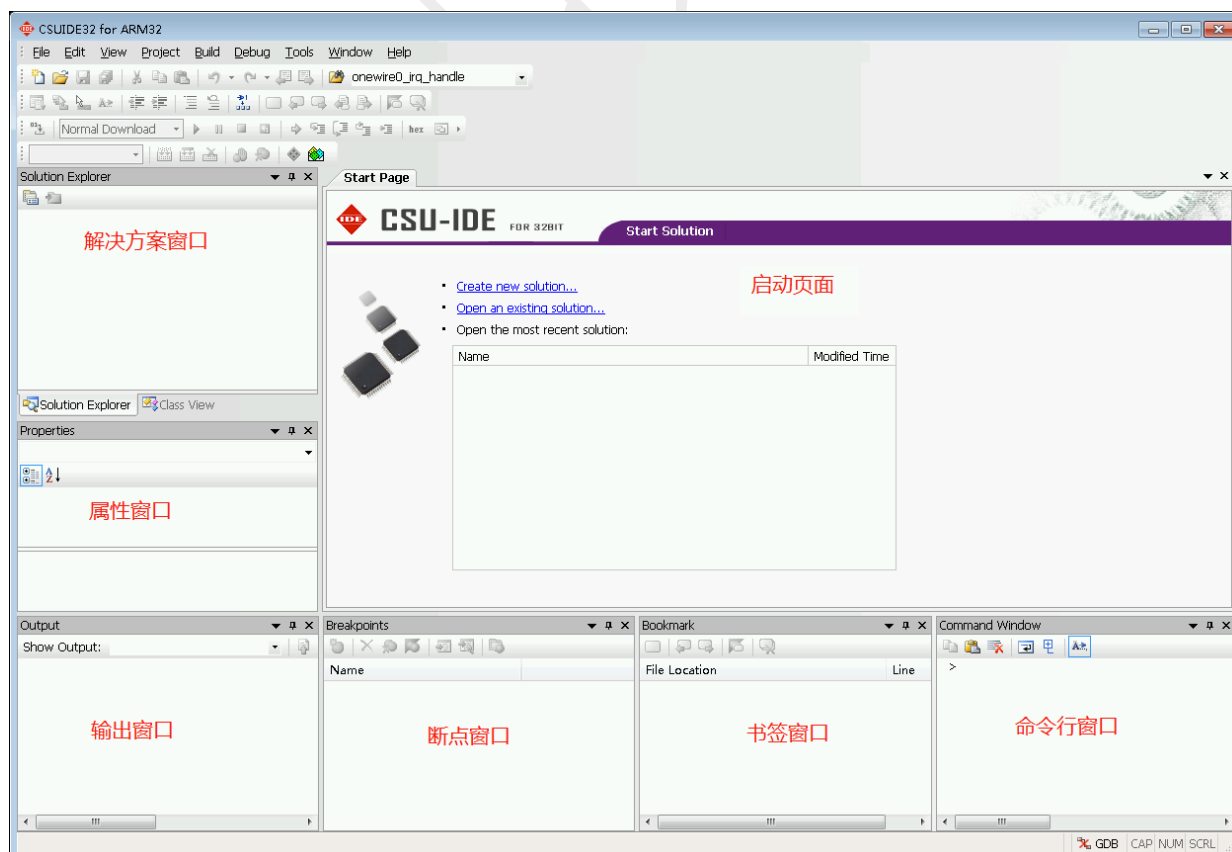
- 1、支持 JLink 仿真器
- 2、支持 CSLink 仿真器

3 主界面

本章将帮助您初步了解 CSU-IDE32 的主界面。界面使用了集成环境开发的经典布局，可以根据个人喜好，随意拖动并组合各个窗口、菜单栏、工具栏等。

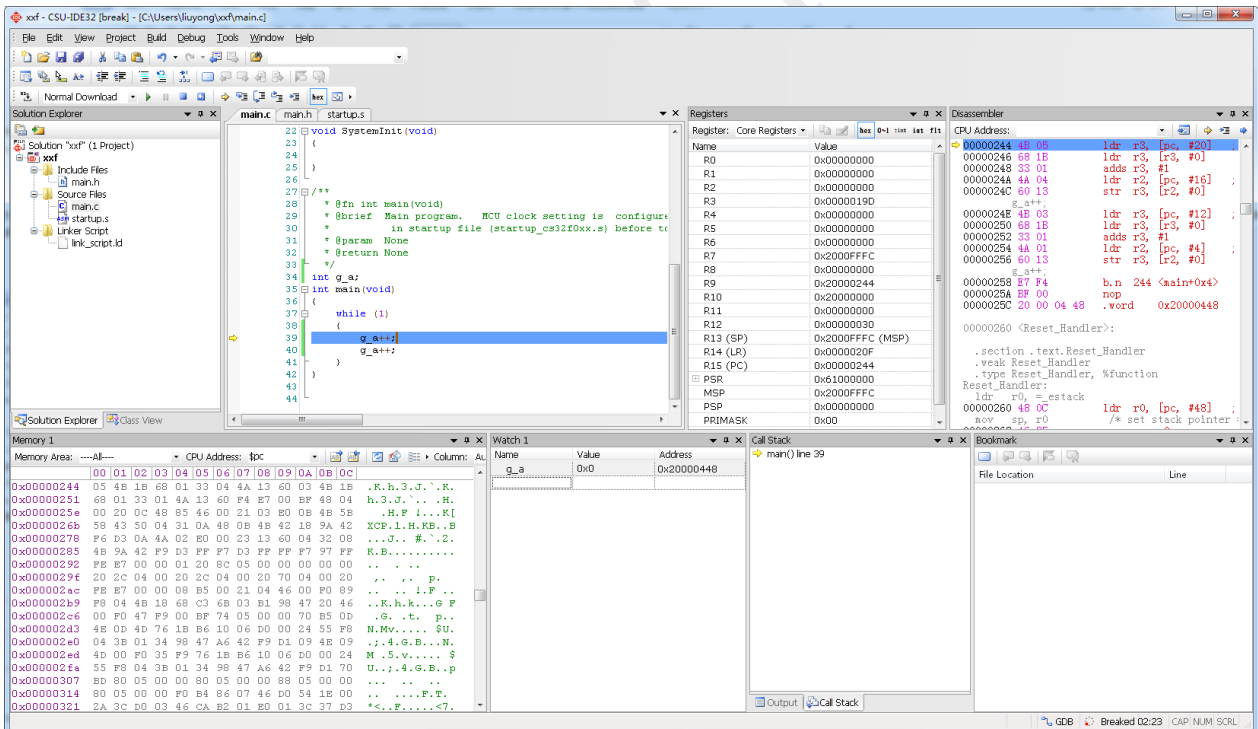
3.1 主界面

第一次开启 IDE，可以看到如下界面，包括启动页面（start page）、解决方案窗口、属性窗口、编辑区、输出窗口等。



窗口	作用
Solution Explorer	显示解决方案、工程以及文件，对所包含的内容进行管理
Properties Window	显示在 Solution Explorer 窗口中选中的解决方案、工程以及文件的属性
Start Page Window	提供快捷方式，可快速新建工程和打开工程
Editor Window	查看、编写代码区域
Output Window	显示编译或调试过程中生成的信息
Bookmark Window	显示书签相关信息
Find Results Window	显示查找结果
BreakPoints Window	显示硬件断点信息，及进行断点设置等操作
Command Window	Windows 的命令行工具，支持执行批处理命令和用户命令
Class View Window	显示定义的函数和全局变量（不包括 SysRegDefine.c 中定义的），默认不显示 Class view 窗口，可通过菜单 View->Class View 打开此窗口。

调试模式下，可查看调试窗口。



窗口	作用
Disassembler	对下载的目标文件进行反汇编操作并显示，显示源程序中每行有效源码对应的汇编指令
Register	显示通用寄存器，CPU 状态寄存器和一些功能寄存器的值

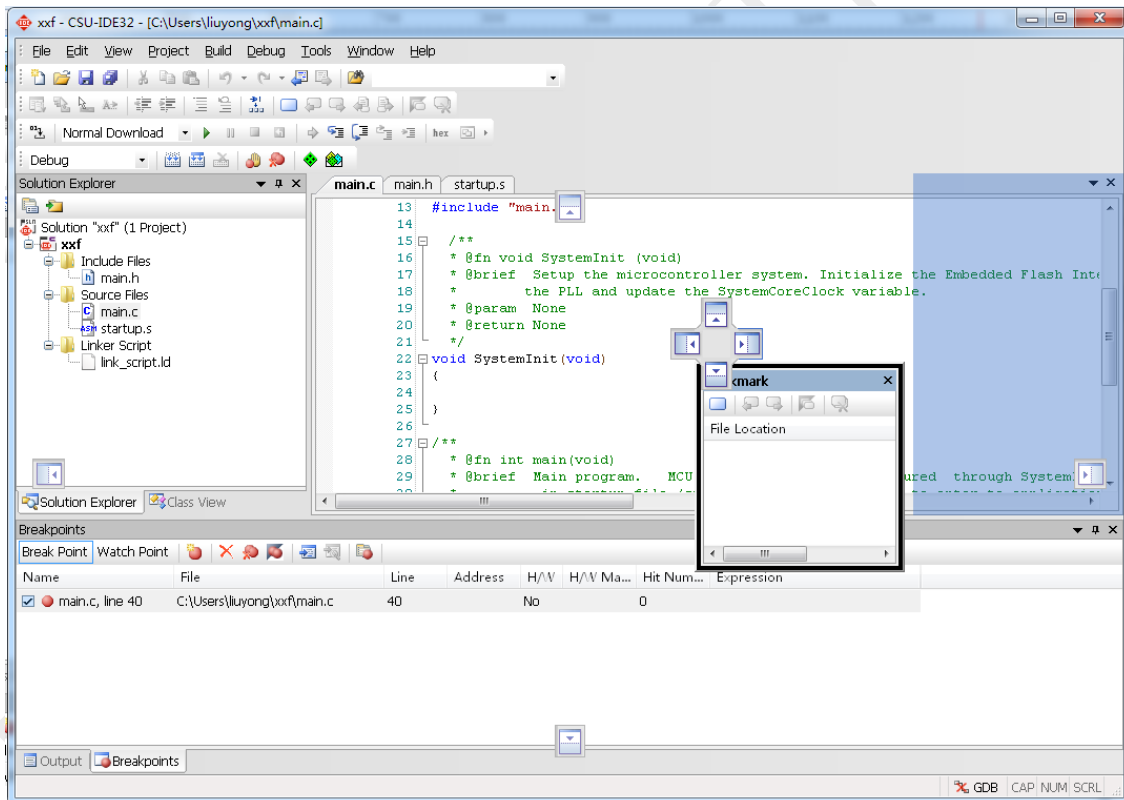
Memory	显示 RAM/ROM/OPTION 区数据
E2PROM	显示 E2PROM 中的内容
Watch	显示变量名、变量值、类型或地址
Call Stack	显示 Call Stack 窗口,默认不显示 Class view 窗口,可通过菜单 View->Debug->Call Stack 打开此窗口。

3.2 设置窗口状态

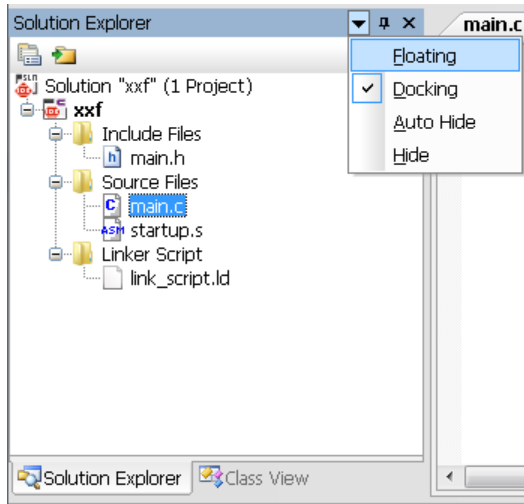
IDE 提供了灵活的窗口布局功能,每个窗口状态可以随意更改:悬浮、停靠、自动隐藏或隐藏等


➤ 拖动窗口

可选中一个窗口进行拖动,拖动过程中会有指示箭头,根据指示箭头停靠在任意位置。参考下图:



在每个窗口的右上角,会有一个向下箭头,点击该箭头,会弹出窗口状态供选择,参考下图:



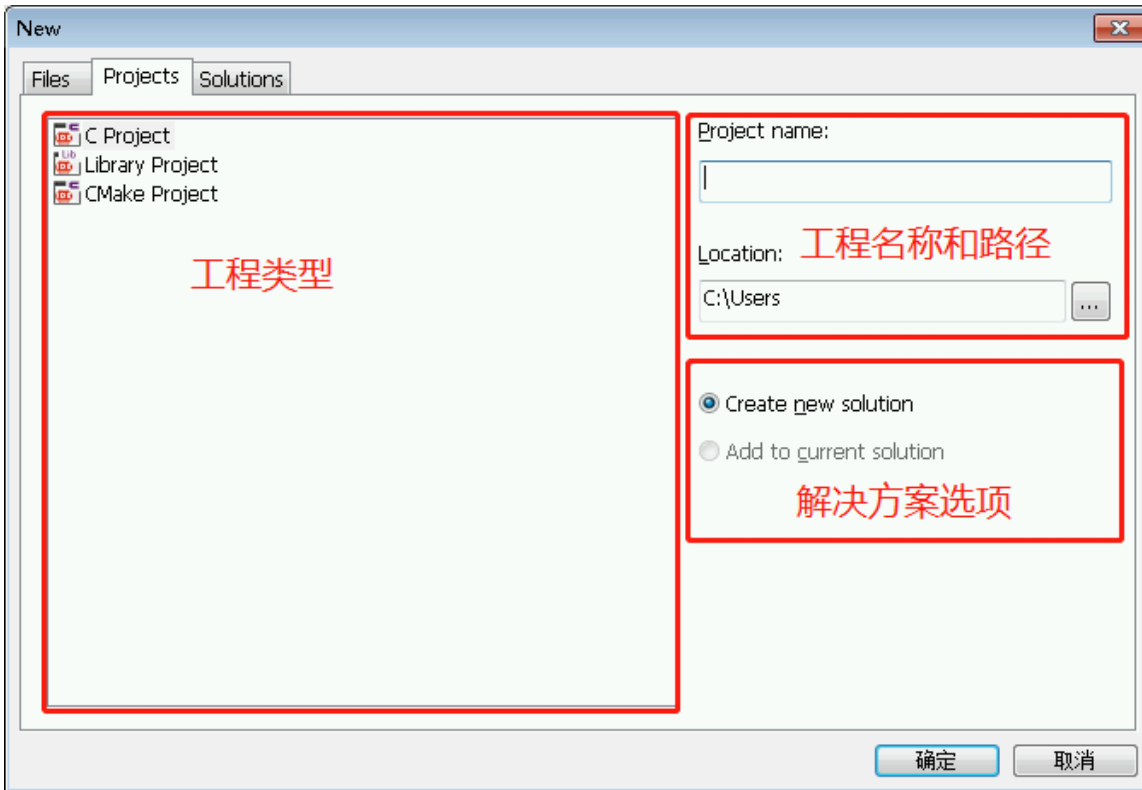
- **Floating:** 将窗口置为悬浮状态，该状态下可随意拖动；选中窗口的标题栏直接拖动，也可以将窗口置为悬浮状态。
- **Docking:** 将窗口停靠在某个位置。
- **Auto Hide:** 自动隐藏。该状态下，不使用时，该窗口会自动隐藏，使用时点击对应标题即可。也可点击右上角 **Auto Hide** 按钮  来实现自动隐藏。
- **Hide:** 将该窗口隐藏，之后只有在菜单栏 **View** 中才能打开。

4 快速入门

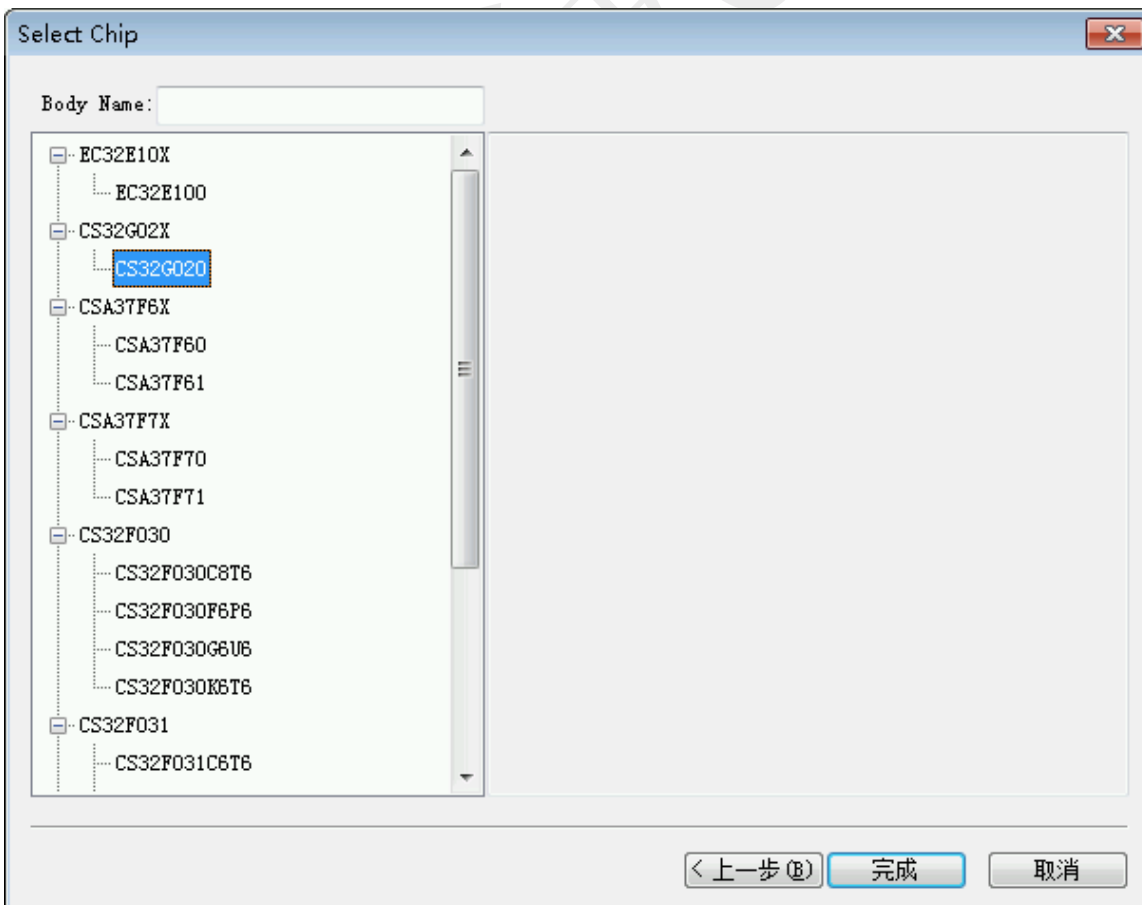
本章我们将用一个简单的工程，介绍主要的操作步骤，比如创建工程、设置工程、编译工程、运行工程以及调试工程。您可以通过查询相关文档了解更多详细功能。

Step1、运行 CSU-IDE32。

Step2、新建工程：选择 **File | Open Project**，可以开启已经存在的工程；或选择 **File | New| Project**，根据向导新建一个工程。如下图所示：



执行上述三个步骤，点击确定，进入芯片类型选择界面，如下图所示：

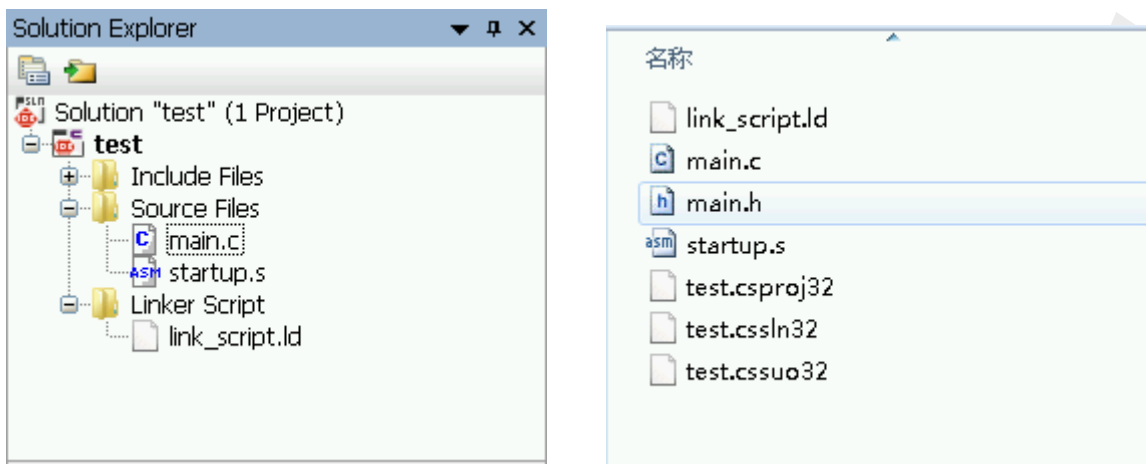


左边区域显示 IDE 支持的芯片类型，右边区域会同步显示选中芯片类型的参数信息。

先选择所需 IC 的类型，再点击完成，一个新的工程创建完毕。

注：可在 Body name 输入框中输入芯片型号，实现快速查找。

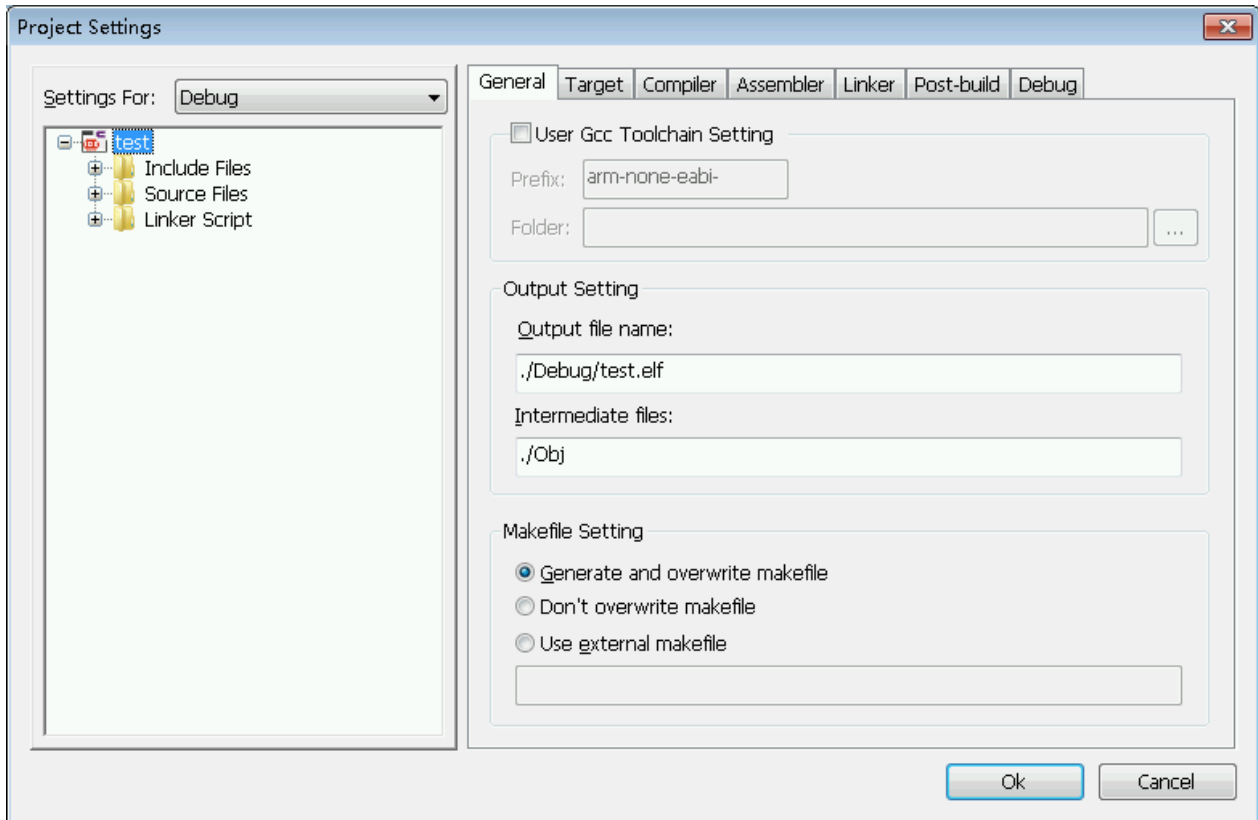
Step3、编辑文件：IDE 会自动开启刚新建完成的工程，在 Solution Explorer 窗口默认位于主界面的左边，当前活动工程的名称会呈黑体加粗显示（如下面左图所示），双击 Source Files 目录，可看到 main.c 的文件，双击将会在编辑器中打开该文件，您可在编辑器中对文件进行编辑。



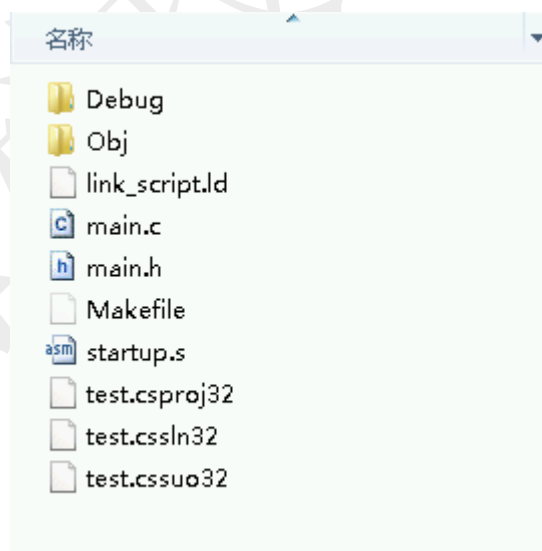
在工程路径下可以看到新建工程产生的文件，如上面右图所示：

- *.csproj32：工程文件
- *.cssln32：解决方案文件

Step4、修改 Setting：选择 Project | Settings，将会弹出工程设置对话框，您可在不同的 Tab 间进行切换并进行相对应的设置。

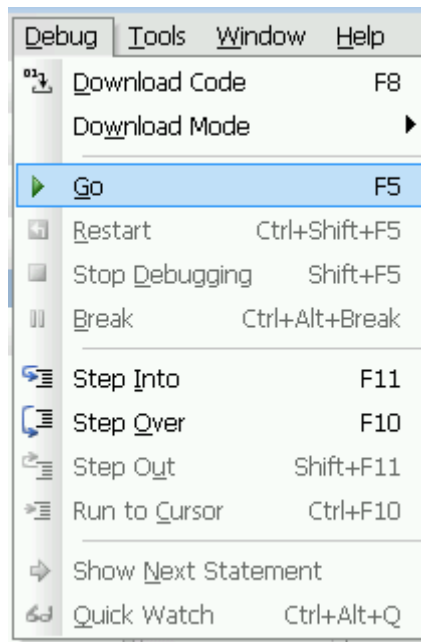


Step5、生成 elf 档：点击 Build | Rebuild All 编译工程，如果编译成功，将会在工程输出路径下（默认是 Debug 文件夹）的输出 elf/hex/map 档。



Step6、下载：点击 Debug | Debug run 下载 code，然后使用 Debug 菜单项中的调试命令开始执行或调试程序。

Step7、调试：使用 Debug 菜单项中的调试命令开始执行或调试工程



5 工程管理

5.1 关于解决方案

1、什么是解决方案？


Solution 被我们称为解决方案。解决方案可理解为用来管理所有工程项目的“管家”。一个解决方案中可以包含多个工程，可以是不同工程类型。使用解决方案，将大大方便开发过程中的相关工程的管理。

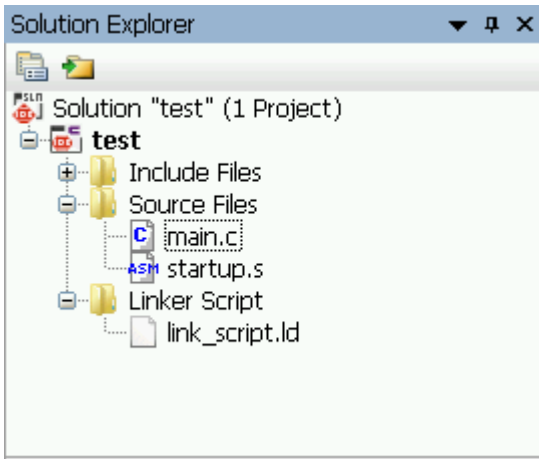
2、解决方案相关文件

新建一个解决方案后，会生成两个文件：`*.csproj32` 和 `*.cssln32`。

3、解决方案管理器

解决方案管理器（**Solution Explorer**）主要用来管理工程以及工程下的文件。您可以创建不同的文件夹，然后将相应的文件添加到对应的文件夹中，这样将大大方便您对文件的查找。但是请注意，解决方案管理器中的文件结构只是一种逻辑关系，和文件的实际存储结构可能没有任何关联。

在解决方案管理器中，提供了一个十分方便而且好用的功能，目录导航，您只需点击 ，可打开所选中的工程或者文件所在文件夹，如下图所示：



4、活动工程（Active Project）

一个解决方案，有且只有一个活动工程。

可通过以下方式设置其它工程为活动工程：

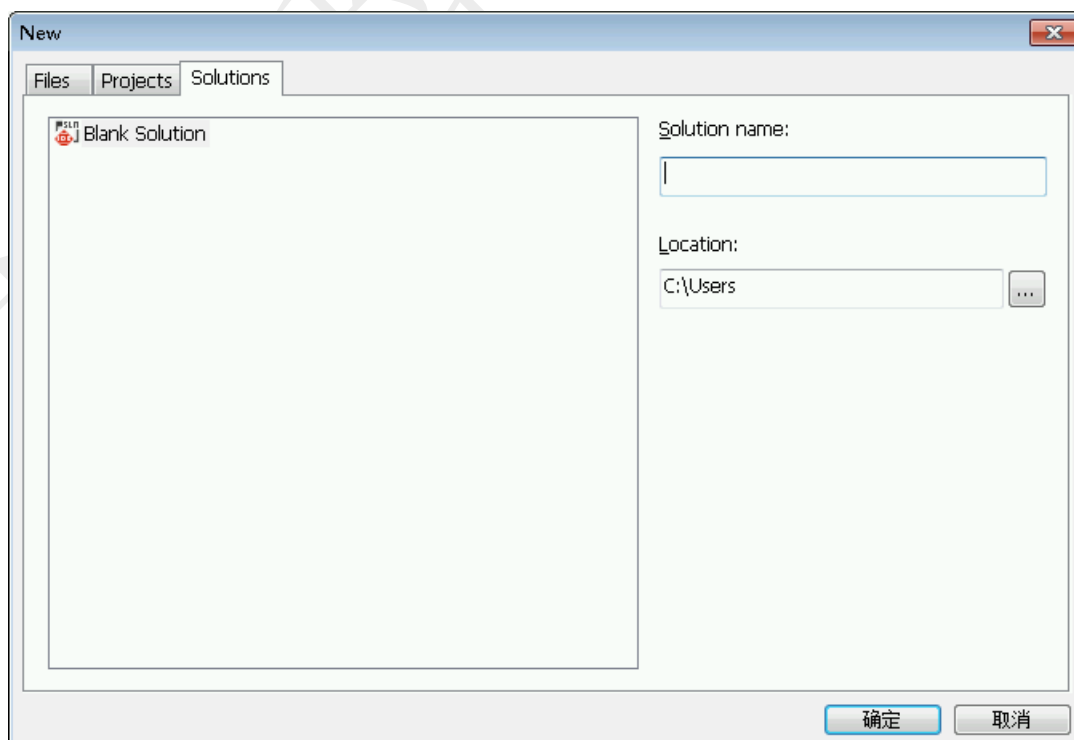
- 1、 选择 Project | Set Active Project，将选中工程设置为活动工程。
- 2、 在 Solution Explorer 窗口中，选中工程名并选择右键快捷方式 Set as active project。

特别说明：执行 Build 和 Debug 菜单功能，均是对当前活动工程进行操作的。

5.2 创建解决方案

- 1、 创建一个空的解决方案

选择 File | New | Blank Solution...，弹出“New”对话框。如下图

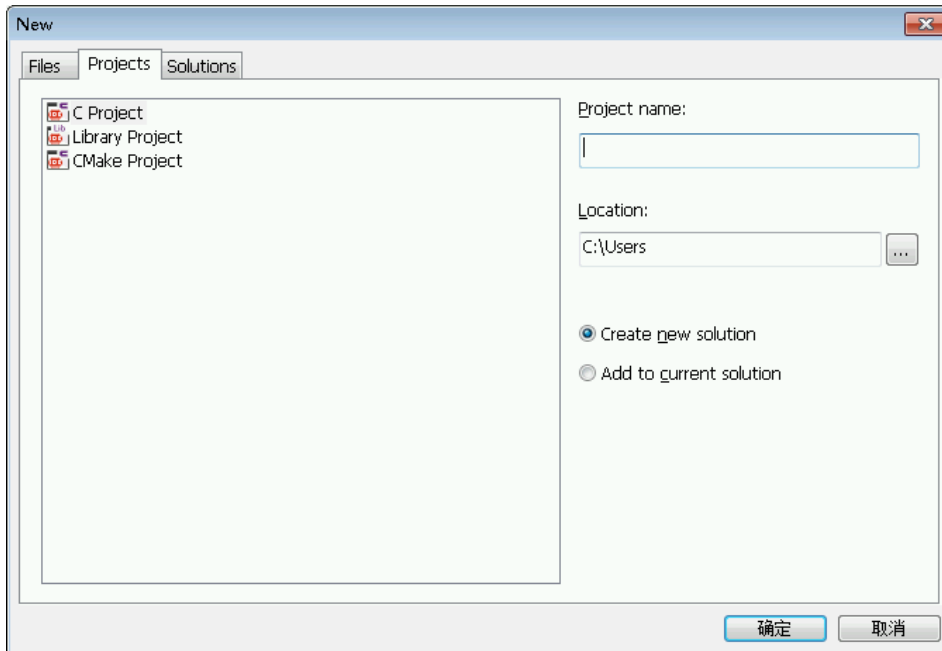


点击确定，将会创建一个空的解决方案。

可再通过菜单 Project | Insert Project into Solution 添加已经存在的工程到解决方案中。

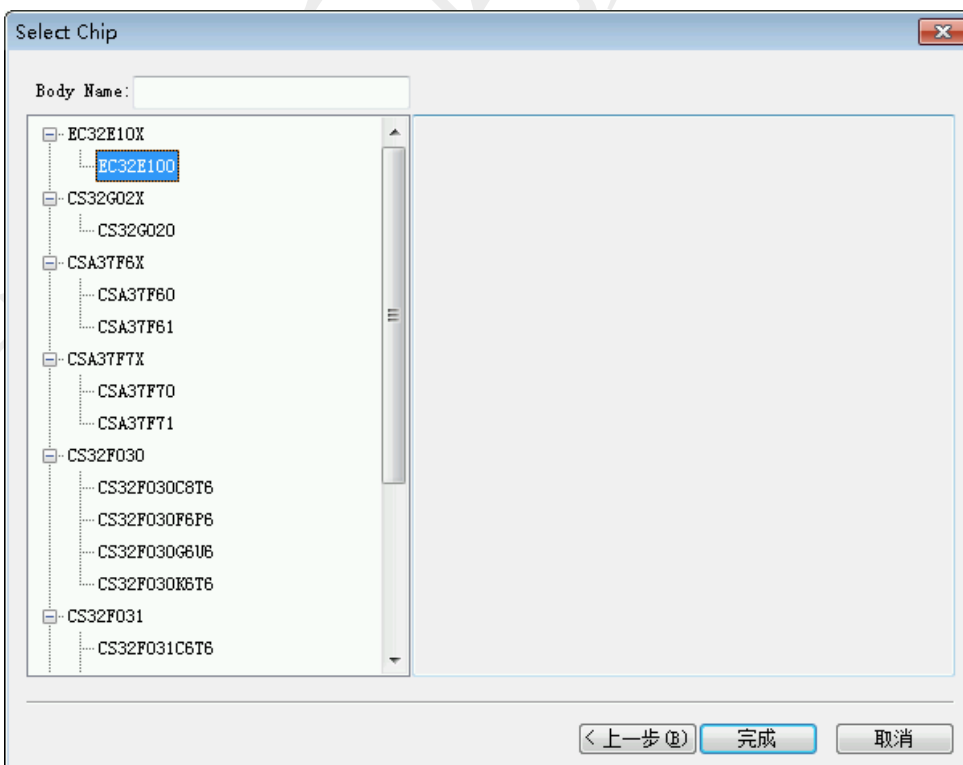
2、创建一个解决方案，包含一个同名工程

选择 File | New | Project，弹出“New”对话框。如下图：



选择 Create new solution，点击确定，将进入 IC 类型选择界面。

注：当前 IDE 已打开一个解决方案时，Add to current solution 控件会显示为有效，选择该控件，将会创建一个新工程，并将新建工程添加至当前的解决方案中。



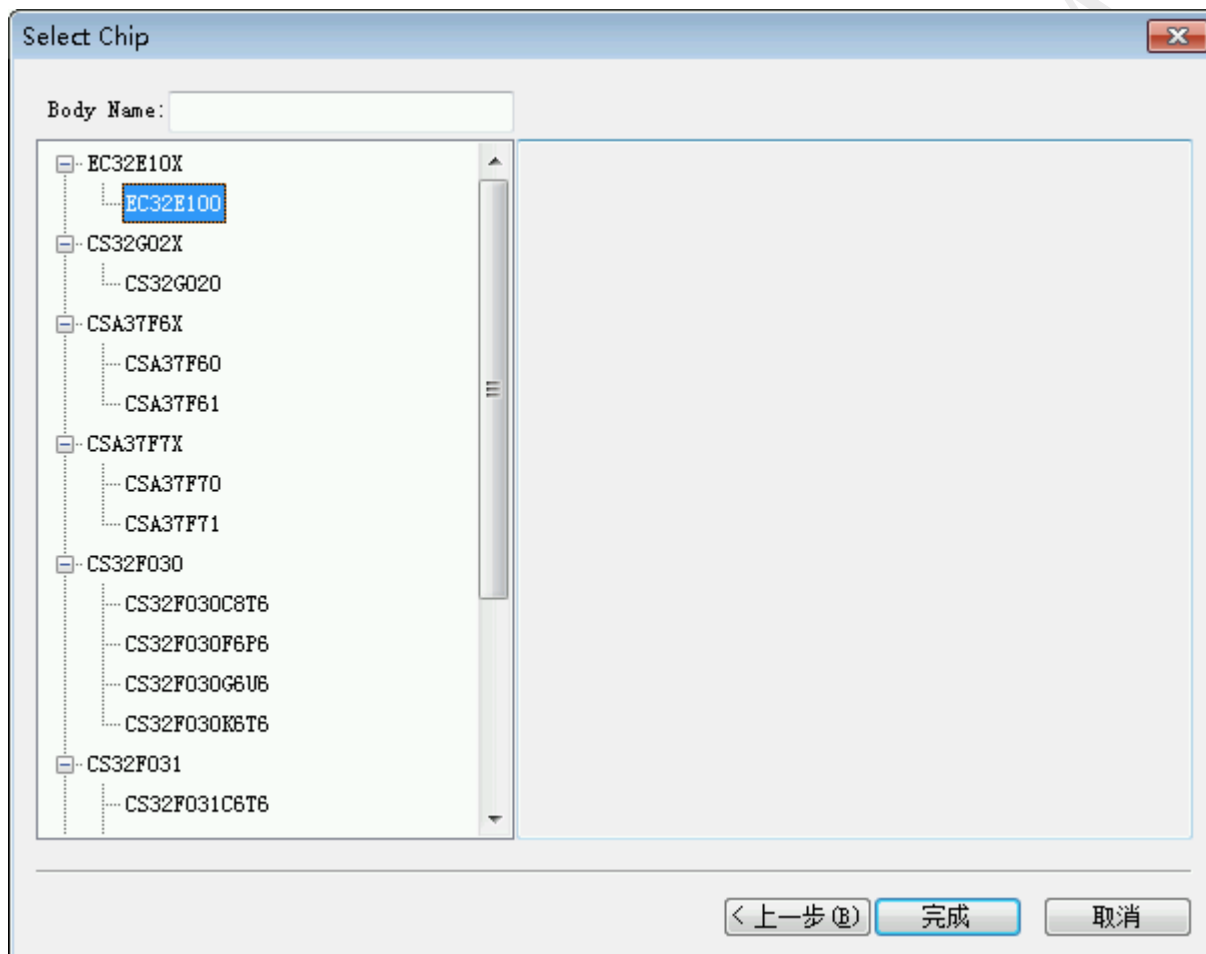
选择您所需的类型，再点击完成，会创建一个新建解决方案，以及一个同名的新工程。

5.3 添加工程

在解决方案中，可添加新的工程，也可以添加已经存在的工程。

1、添加新工程

IDE 开启一个解决方案，再点击选择 **File | New | Project**，会弹出“New”对话框，如下图：

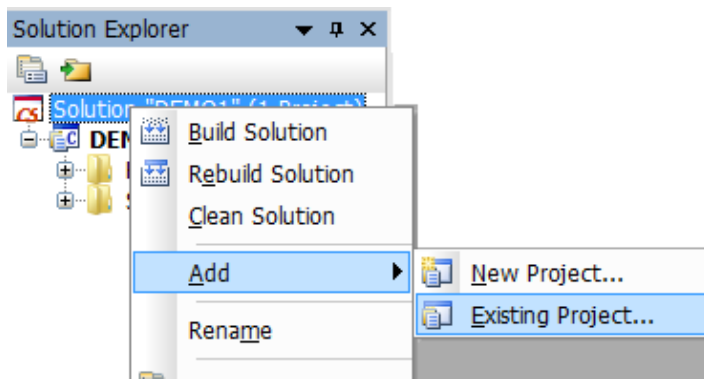


选择 **Add to current solution** 控件，点击确定，将会创建一个新工程，并将新建工程添加至当前的解决方案中。

2、添加已存在的工程

点击 **Project | Insert Project into Solution**，弹出“Open”对话框，选择后缀为“**csuprjx**”的工程文件，点击“OK”，工程即被加入到当前解决方案中。

也可以直接在解决方案管理器窗口中，选中解决方案，单击右键，**Add | Existing Project**。

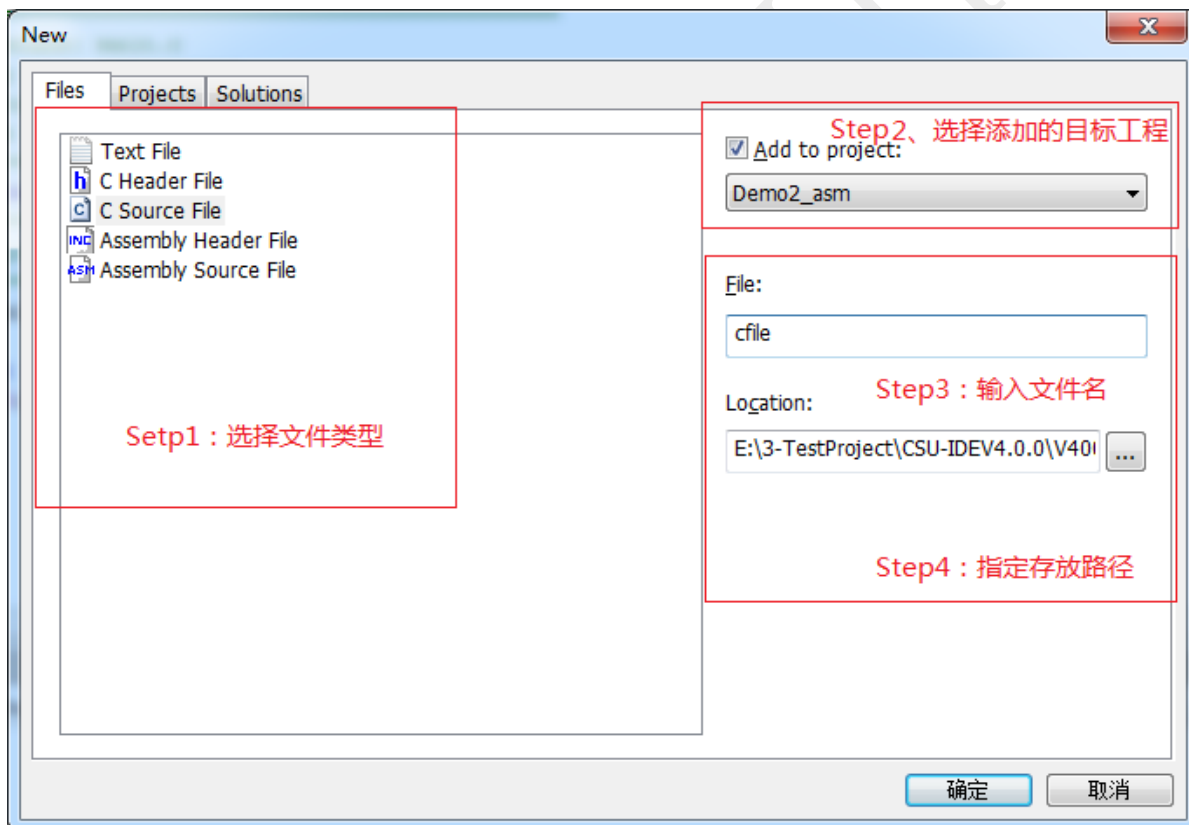


5.4 添加文件

在工程中，可添加新文件和已经存在的文件。

1、创建新文件

选择 File | New | File，弹出“New”对话框，按照下面步骤操作，将会创建一个新文件至指定的目标工程。

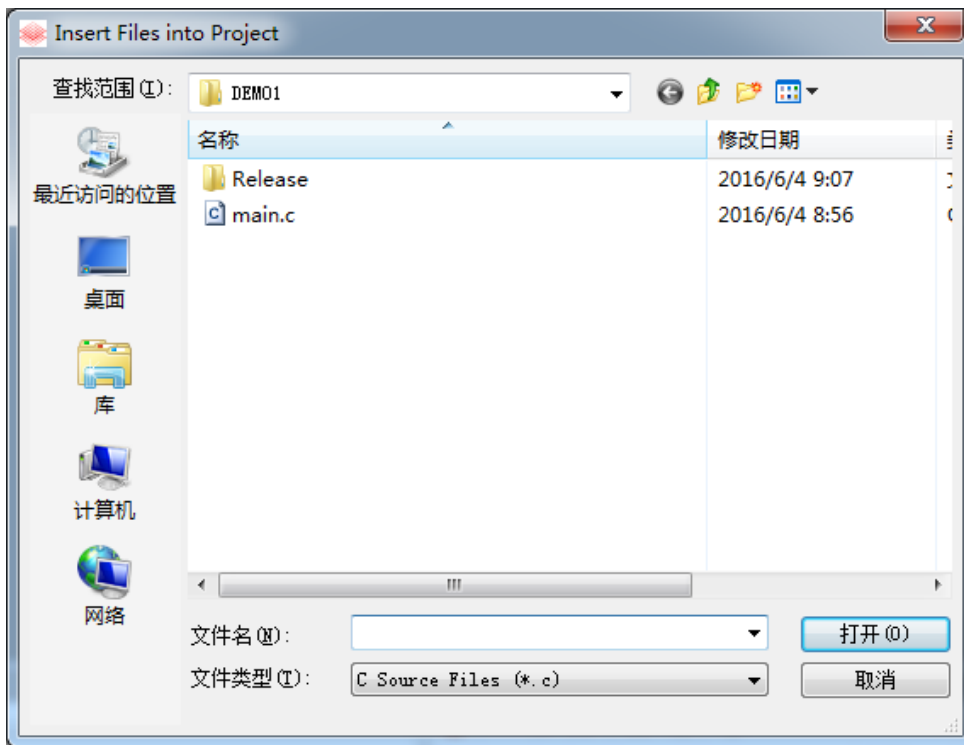


其它方法：

➤ 选择 Project | Add To Project | New File，弹出“New”对话框，其余操作步骤与上面一致。

2、添加已存在的文件

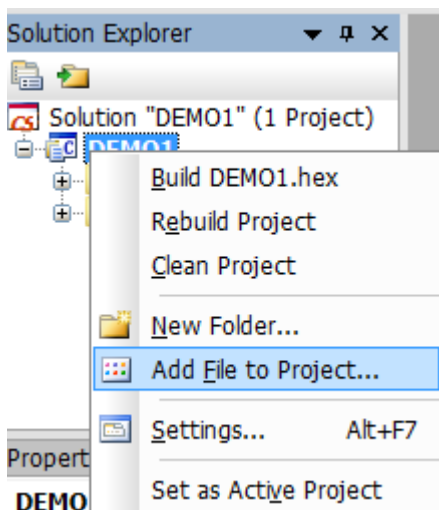
选择 Project | Add To Project | Exist Files，弹出“Insert Files into Project”对话框。



选择要添加的文件，点击【Open】，所选择文件将根据其后缀自动添加到当前工程的对应文件夹中。

其它方法：

- Solution Explorer 窗口的右键快捷方式 Add File to Project。



5.5 添加文件夹

在解决方案管理器中可以看到多个文件夹，文件夹主要用于文件管理，把某一个类型或其相关类型的多个文件放到同一个文件夹中，方便文件查找。IDE 对不同的工程类型默认提供了不同的文件夹，在新建工程后，可在解决方案管理器窗口中看到工程下的默认文件夹。您也可以选中工程，点击右键，通过“New Folder”命令创建文件夹，或选中某个文件夹，点击右键，通过“New Folder”在已有文件夹中创建子文件夹。

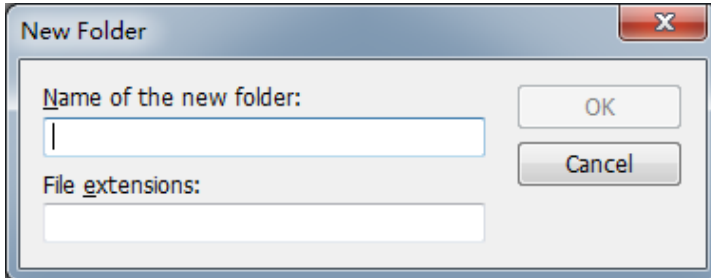
- “Source Files”：源文件夹。添加到该目录通常都是一些源文件，如*.asm，*.c 文件等。

- “Header Files”：头文件夹。用于添加源文件所包含的头文件。

添加文件夹至工程：

Setp1: 选择 Project | Add To Project | New Folder，弹出“New Folder”对话框。

Solution Explorer 窗口也提供了 New Folder 右键快捷方式。



Setp2: 编辑文件名和文件后缀类型，点击【OK】即可。

5.6 解决方案管理器（Solution Explorer）

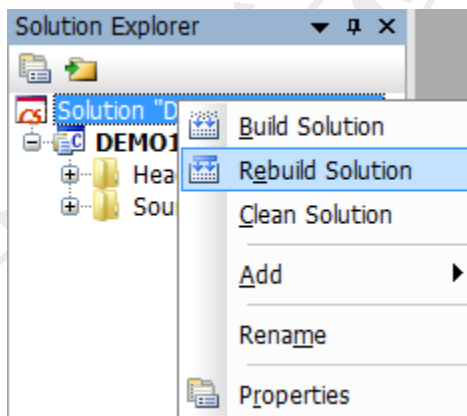
“Solution Explorer”即解决方案管理器，主要用来管理工程以及工程下的文件。您可以创建不同的文件夹，然后将相应的文件添加到对应的文件夹中，这样将大大方便您对文件的查找。

请注意：解决方案管理器中的文件结构只是一种逻辑关系；在解决方案管理器中添加、修改、移除、重命名等操作，不会对文件的实际存储结构产生影响。

右键单击解决方案名称或工程名称、文件夹名、文件，均会弹出相应的快捷菜单，方便用户使用。

1、解决方案

选中解决方案的右键快捷菜单，如下图所示：

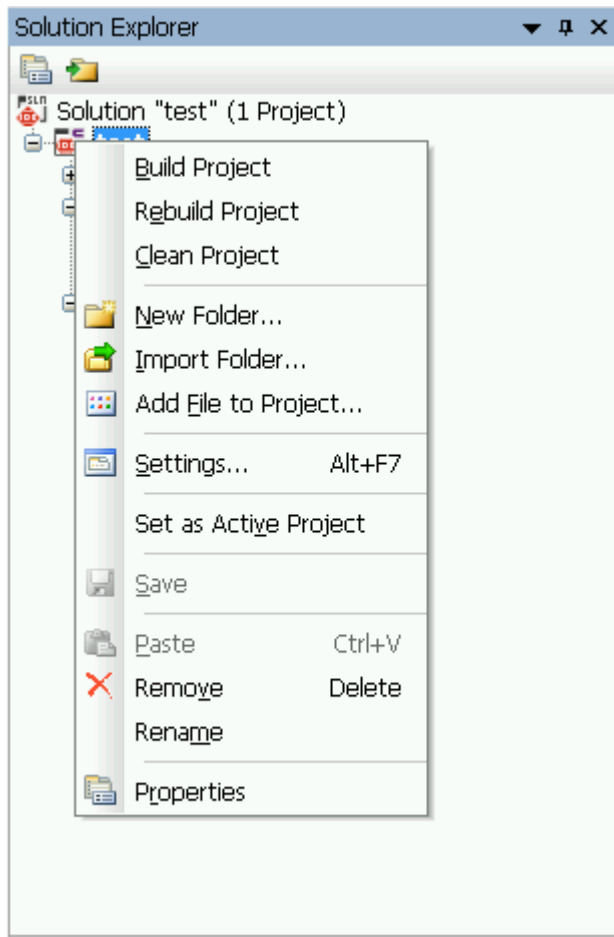


Build Solution	构建解决方案，会对解决方案中的所有工程进行构建。
Rebuild Solution	重新构建解决方案中所有的工程。
Clean Solution	清除解决方案中所有工程的输出文件
Add	在解决方案中，添加新工程或已经存在的工程。

Rename	为解决方案重新命名。
Properties	切换至属性窗口，并显示解决方案的各种属性。

2、工程

选中工程的右键快捷菜单，如下图所示：

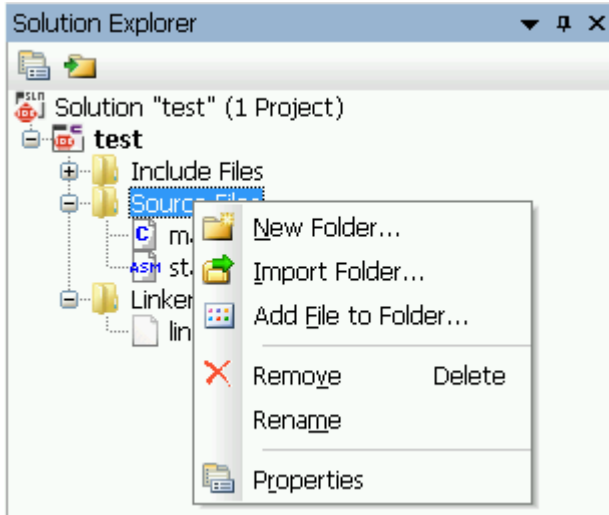


Build Project	增量编译，只对修改的源文件重新编译，生成 ELF/HEX/MAP 档。
Rebuild Project	重新构建该工程，生成 HEX 档。
Clean Project	清除该的输出文件。
New Folder...	在该工程中，新建一个文件夹。
Import Folder...	导入一个目录的所有文件到工程 中
Add File to Project...	在该工程中，添加 1 个或多个文件。
Settings...	打开该工程的工程设置窗口。
Set as Active Project	将该工程设置为活动工程。
Save	保存对该工程的修改。
Rename	为该工程重新命名。

Properties 切换至属性窗口，并显示该工程的各种属性。

3、文件夹

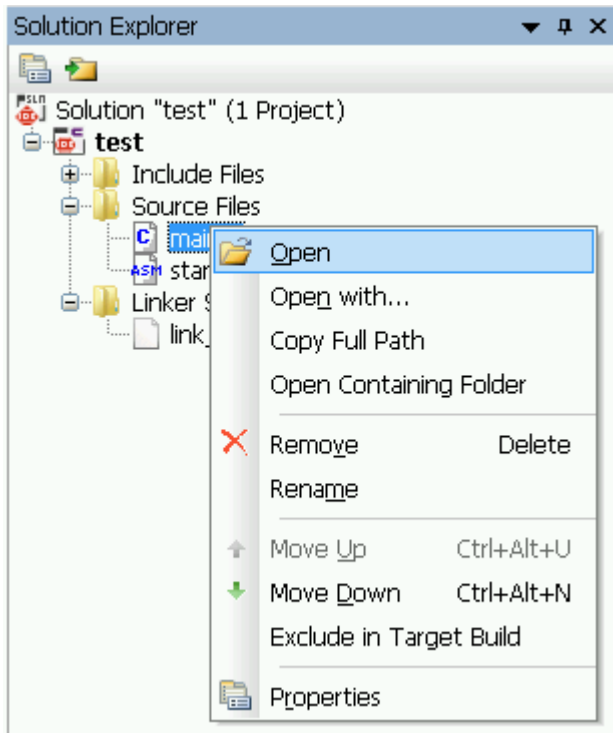
选中文件夹的右键快捷菜单，如下图所示：



New Folder...	在该工程中，新建一个文件夹。
Import Folder...	导入文件夹中所有的文件到工程中
Add File to Project...	在该工程中，添加 1 个或多个文件。
Remove	移除该文件夹和文件。
Rename	为该文件夹重新命名。
Properties	切换至属性窗口，并显示该文件夹的各种属性。

4、文件

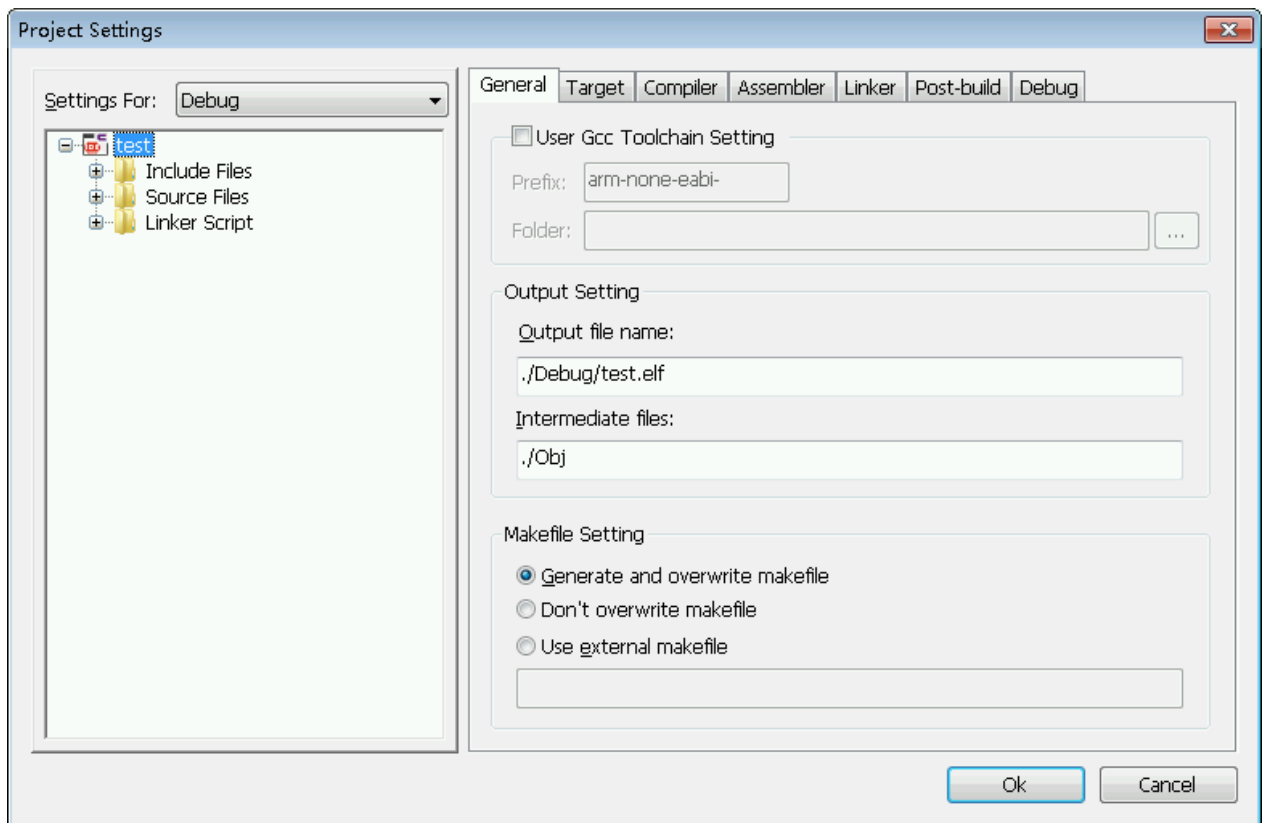
选中文件的右键快捷菜单，如下图所示：



Open	会在编辑区打开该文件
Open with...	会弹出打开方式对话框，再选择想用来打开此文件的程序即可
Copy Full Path	复制文件全路径到剪贴板中
Open Containing Folder	打开文件所在的文件夹
Settings...	打开该文件设置对话框
Remove	移除该文件
Rename	为该文件重新命名
Move Up	向上移动
Move Down	向下移动
Exclude/Include in Target Build	不包含/包含在构建中
Properties	切换至属性窗口，并显示该文件的各种属性

5.7 工程设置（Project Settings）

此功能是针对单个 Project 选项的设置，包括 General、Target、Compiler、Assembler、Linker、Post-build、Debug 等页面。



➤ General

1、User GCC Toolchain Setting

用于修改 IDE 默认 GCC 编译工具链，用户可以下载使用最新版本的 GCC 工具链。

2、Output Setting

该用来设置和查看输出目录、中间文档目录、目标文档名、Makefile 名。

Output Files: 指定工程的 Hex 输出目录，工程编译时会将 Hex 输出在此目录。执行 Clean 操作，会清除该输出目录的文件。

Intermediate Files: 指定工程的中间文档的输出目录，工程编译时会将中间输出在此目录。执行 Clean 操作，会清除该输出目录的所有输出文件。

3、Makefile Setting

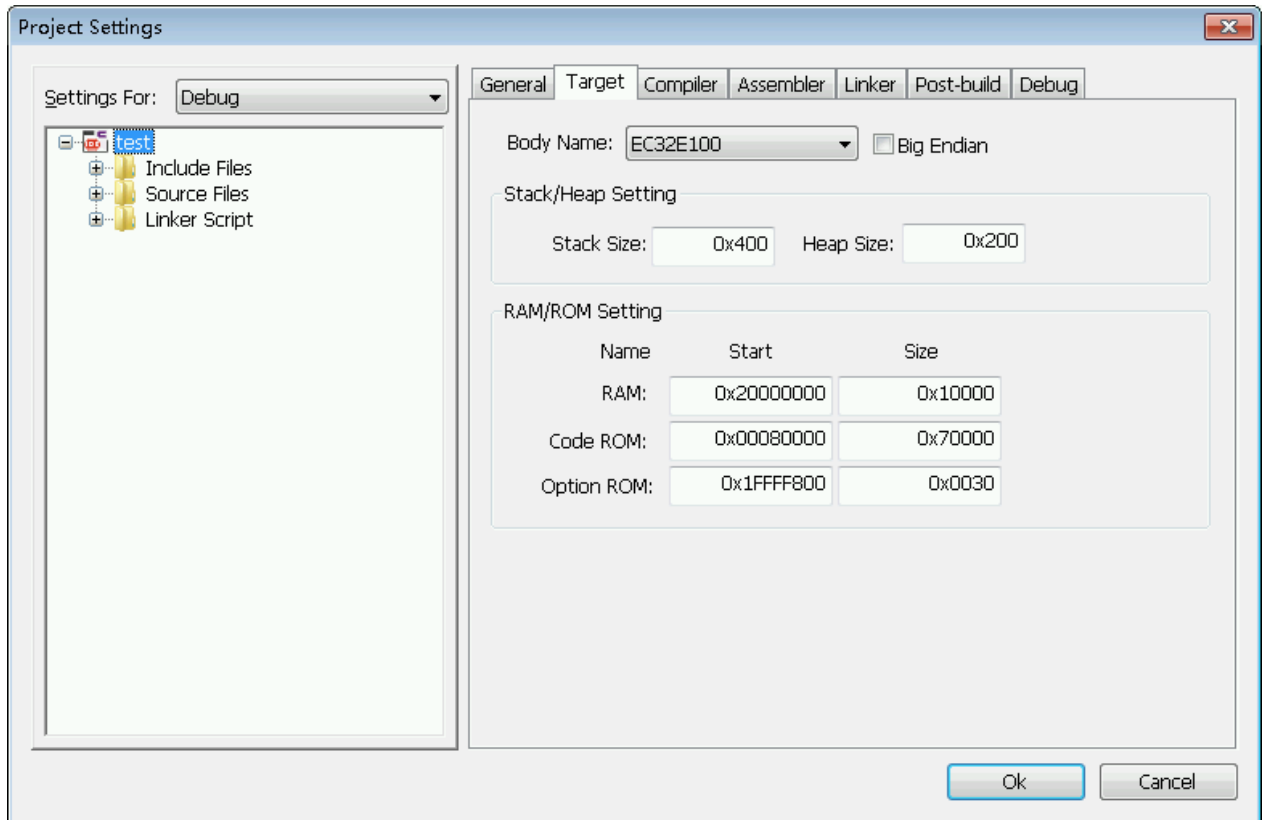
Makefile FileName: 指定 MakeFile 文件名，暂不提供编辑功能。

Generate and overwrite Make file: 是否在编译时重新生成和替换已有的 Make file。

Use external makefile: 指定用户自定义的 makefile。

➤ Target

在该页面显示型号，可以配置堆栈空间大小和 RAM/ROM/OPTION 区域的范围。



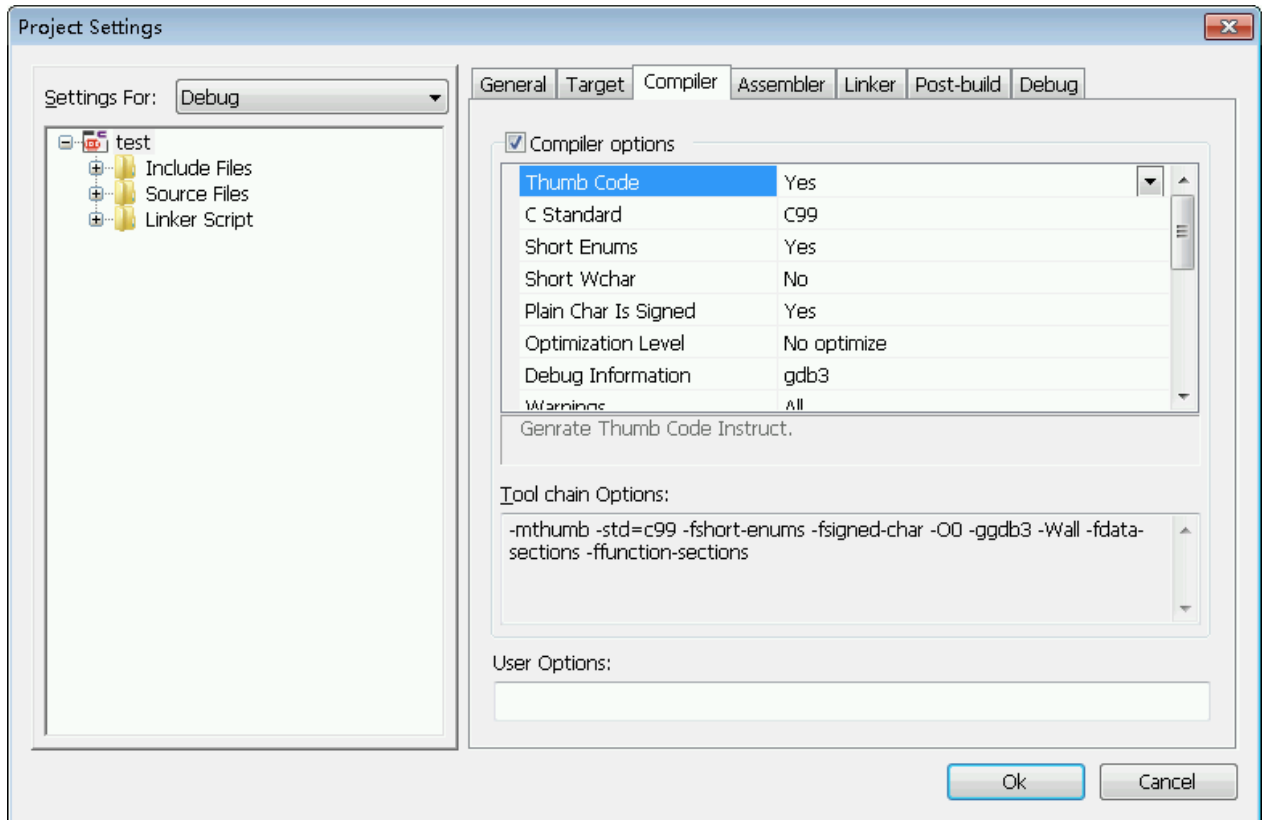
Body Name: 显示 IC 名称;

Stack/Heap Setting: 设置堆栈保留空间大小。

RAM/ROM Setting: 设置 RAM/ROM/Option 空间范围。

➤ Compiler

在该页面可设置编译器的相关参数及行为。



Debug Information: 是否在指定的输出目录中会生成调试信息（后缀为 `c` 的源文件对应的调试信息）。

Additional include paths: 添加 include 文件所搜寻的目录。

操作步骤如下：

Step1: 点击 ，在 Additional include paths 列表中弹出一个编辑区

Step2: 点击 ，弹出路径选择框

Step3: 选择需要添加的路径，就完成了，如下图所示



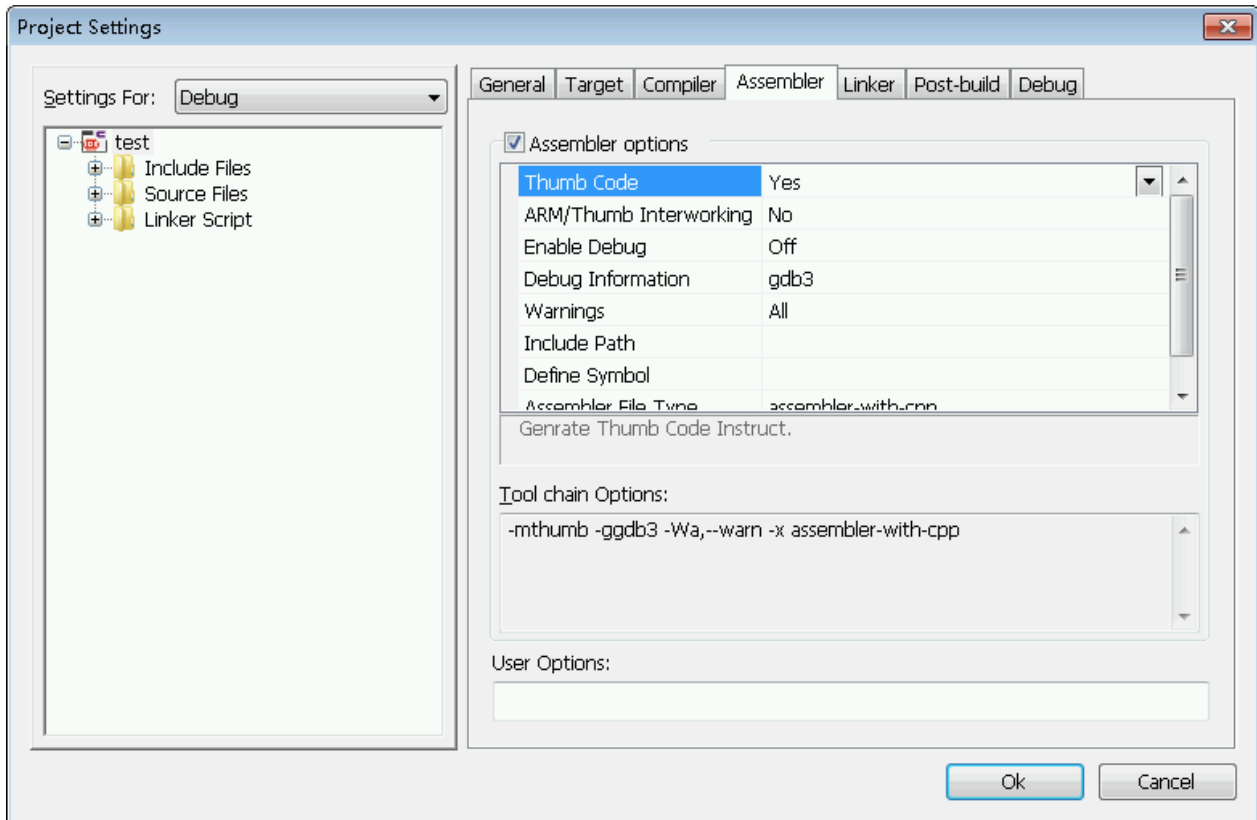
Step4: 在源代码中，就可以使用 `include` 命令来关联该目录中的文件了。

Optimization Level: 可选择 `-O0`、`-O1`、`-O2`、`-Oz`、`-Os` 五种优化级别，暂时支持 `-O0` 优化级别。

注：C 工程才有该 Compiler 页面。

➤ Assembler

在该页面可设置汇编器的相关参数及行为。



Debug Information: 是否在指定的输出目录中会生成调试信息（后缀为 `asm` 的源文件对应的调试信息）。

Warnings: 在编译时是否显示警告信息。勾选该选项，在编译时在 `Output` 窗口不会显示警告信息。

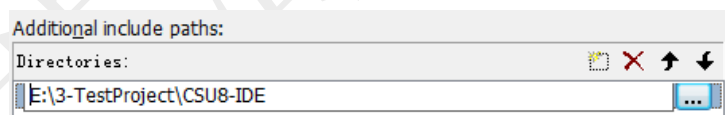
Additional include paths: 添加 `include` 文件所搜寻的目录。

操作步骤如下：

Step1: 点击 ，在 `Additional include paths` 列表中弹出一个编辑区

Step2: 点击 ，弹出路径选择框

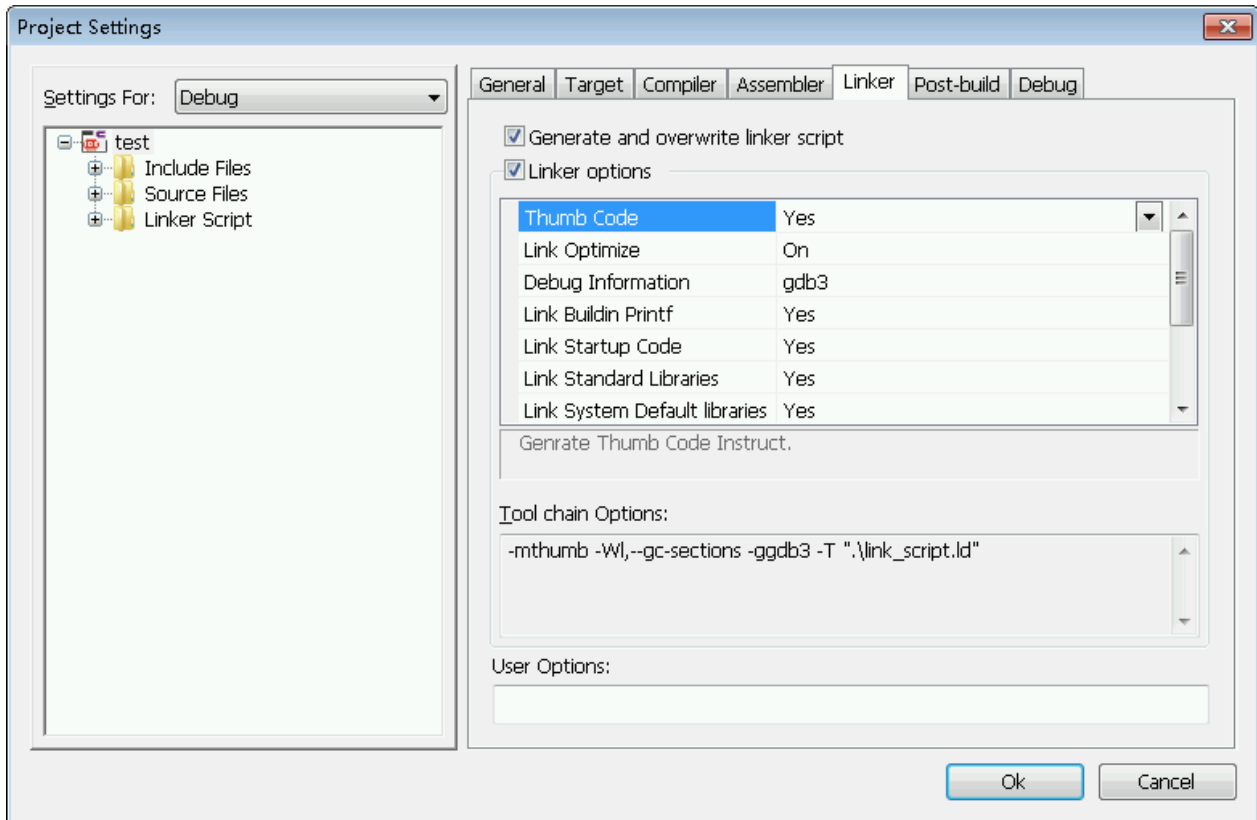
Step3: 选择需要添加的路径，就完成了，如下图所示



Step4: 在源代码中，就可以使用 `include` 命令来关联该目录中的文件了。

➤ Linker

在该页面可设置链接器的相关参数及行为。



Link Optimize: 开启后不使用的函数和变量会被优化

Link Library: 决定是否要链接默认的系统库。

Libraries to link: 添加 lib 文件路径。

操作步骤如下：

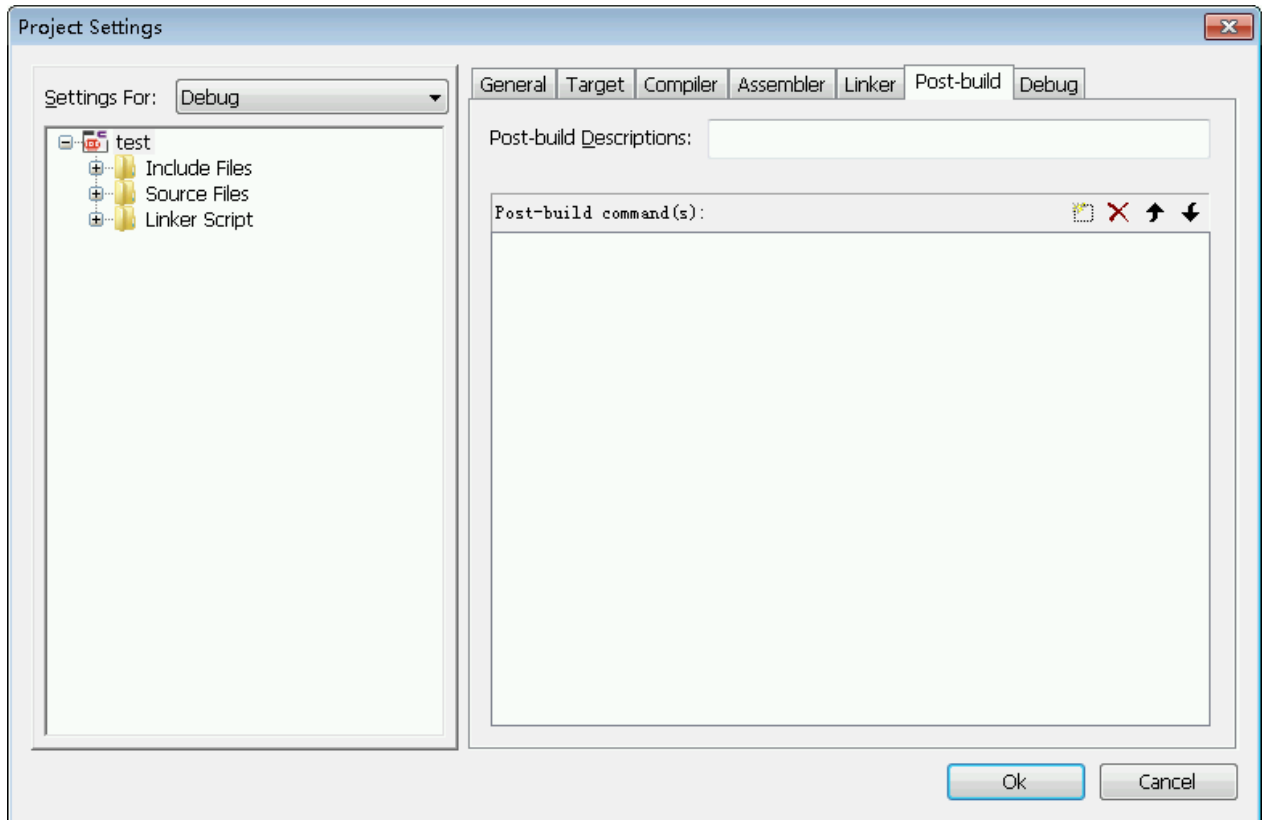
Step1: 点击 ，在 Additional include paths 列表中弹出一个编辑区

Step2: 点击 ，弹出路径选择框

Step3: 选择需要添加的 a 文件就完成了，如下图所示

➤ Post-build

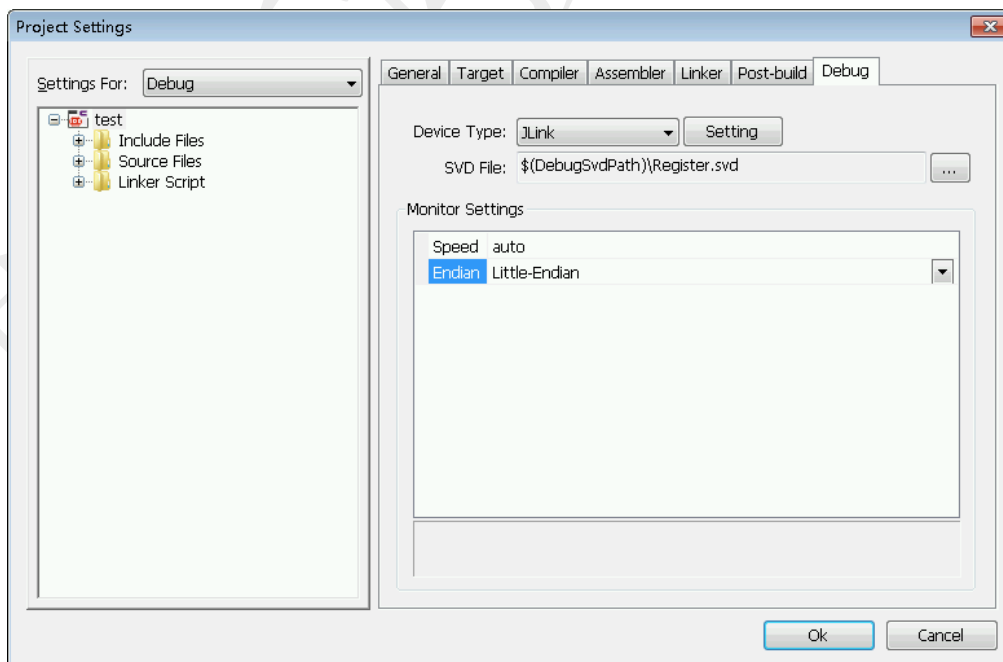
在该页面可添加一个或多个脚本命令。



可添加一个或多个命令，这些命令会在编译结束后执行。

➤ Debugger

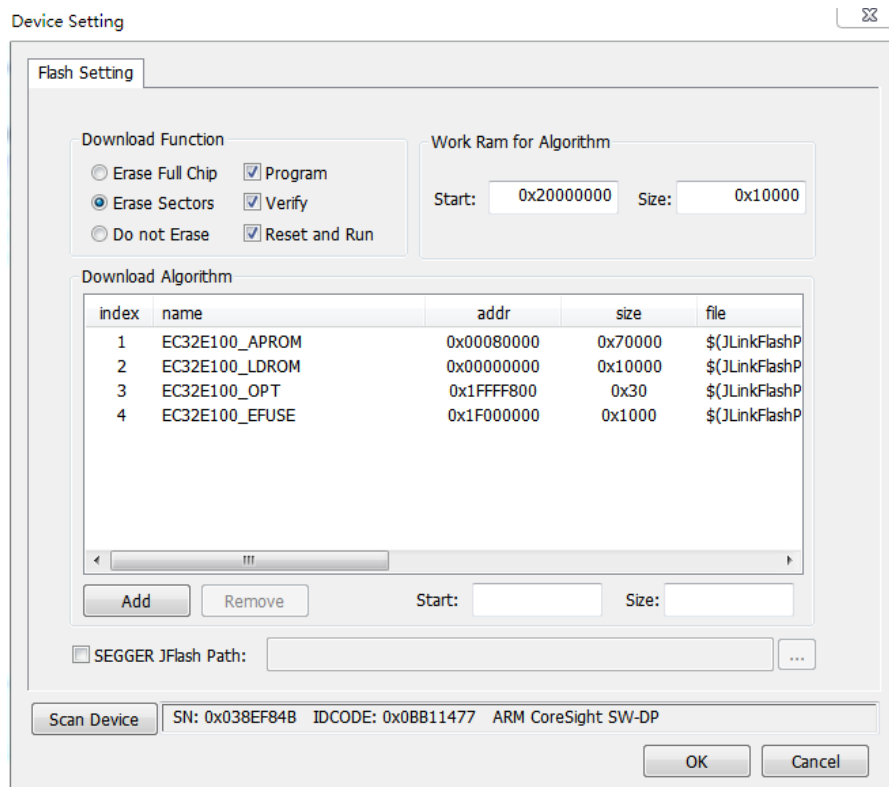
在该页面可设置调试器的相关参数及行为。



Device Type:可以选择设备类型 JLINK/CSLink， 所有型号都支持 JLINK， 但是 CSLink 只有部分支持。

SVD File:可以指定用户自定义的 SVD 文件，其决定 Register 窗口中寄存器列表的显示。

➤ JLink 设备配置



可以配置烧录参数和算法文件，并支持检查 JLink 设备状态。

5.8 编译解决方案或工程 (Build)

IDE 可编译单个工程，也可编译一个解决方案下的所有工程。编译成功，将会在工程输出路径下的输出*.hex 档

1、构建活动工程

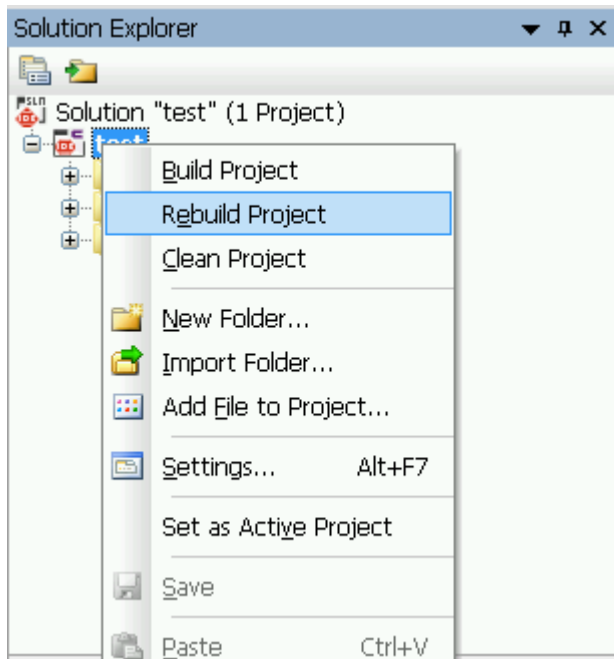
选择 **Build | Rebuild All**，将会对当前解决方案的活动工程进行编译。会先执行 **clean** 操作，再对工程所有文件重新编译，生成 HEX 档。

选择 **Build | Build**，会对修改过的源文件进行重新编译，重新生成 HEX 档。

建议：工程文件较小时，修改源文件后，尽量使用 **Rebuild All**，重新构建工程。

2、构建选中的工程

需要构建非活动的工程，需在 **Solution Explorer** 窗口中，选中一个工程名，选择右键快捷菜单 **Rebuild Project**。



3、编译所有工程

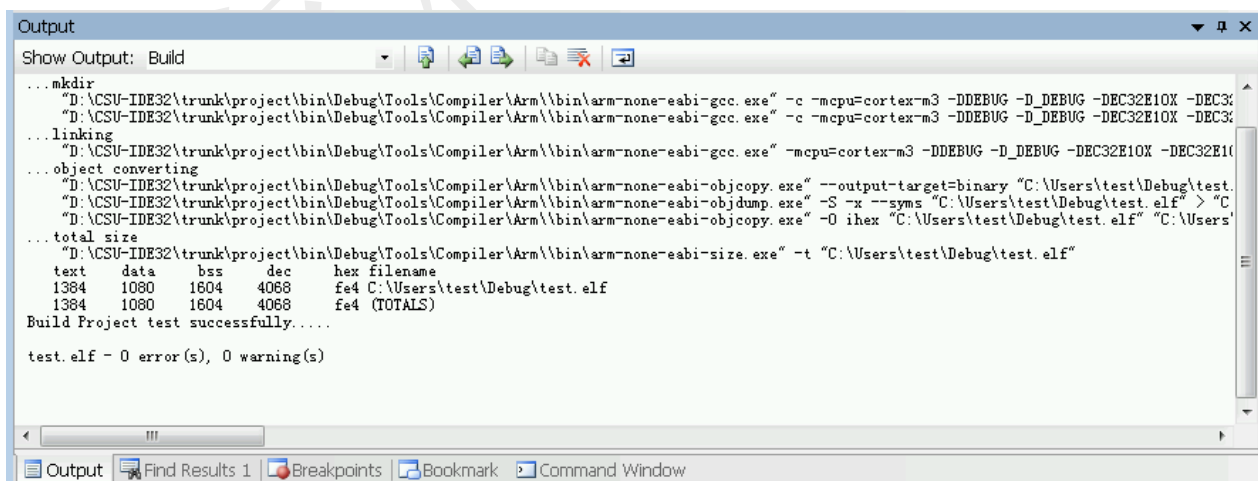
在 Solution Explorer 窗口中，选中解决方案，选择右键快捷菜单 **Rebuild Solution**。

注：编译完毕，可在 Output 窗口中查看结果。

5.9 输出窗口（Output Window）

编译过程中工具链的输出信息都会在输出窗口中显示，可以通过输出窗口查看编译的状态及产生的信息，编译过程中产生的错误和警告也将在该窗口显示。编译结束后，查看统计结果，如果错误为 **error 0**，则表示编译成功。

在输出窗口中，按住 **Ctrl** 键，同时滚动滑鼠，可以放大或缩小其显示字体。



注：存在 **Warning** 信息不影响调试，但说明程序存在问题，建议用户完善代码。

5.10 .lik 文件

新建工程，会自动生成 Script.lik 文件，是标准的 GCC 链接脚本。

```

link_script.ld  main.c  Start Page
1
2  /* Entry Point */
3  ENTRY(Reset_Handler)
4
5  /* Specify the memory areas */
6  MEMORY
7  {
8  RAM (xrw)      : ORIGIN = 0x20000000, LENGTH = 0x10000
9  FLASH (rx)     : ORIGIN = 0x00080000, LENGTH = 0x70000
10 OPTIONFLASH (rx) : ORIGIN = 0x1FFFF800, LENGTH = 0x0030
11 }
12
13 /* Highest address of the user mode stack */
14 __stack = ORIGIN(RAM) + LENGTH(RAM);
15 _estack = __stack; /* specific definition */
16
17 /* Generate a link error if heap and stack don't fit into RAM */
18 _Min_Heap_Size = 0x200; /* required amount of heap */
19 _Min_Stack_Size = 0x400; /* required amount of stack */
20
21 /* Define output sections */
22 SECTIONS
23 {
24 /* The startup code goes first into FLASH */
25 .isr_vector :
26 {
27     . = ALIGN(4);
    
```

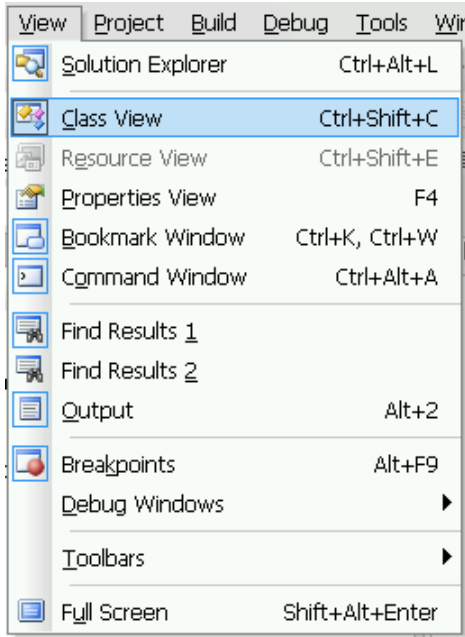
ENTRY: 指定程序入口；

MEMORY: 指定 RAM/ROM/OPTION。

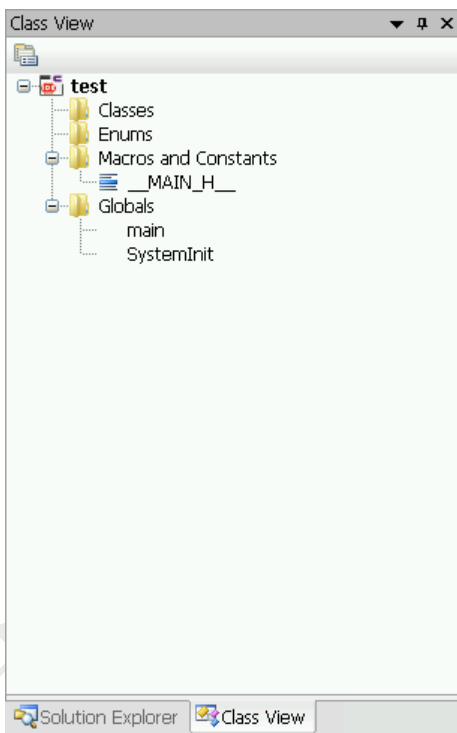
5.11 Class View 窗口

源代码导航窗口是针对用户定义的全局变量、函数等在 Class View 视图中显示出来，方便用户查看并进行定位，在变量、函数使用的地方右键选择跳转到声明、定义处即可跳转到声明、定义的地方。暂不包含 SysRegDefine.c 中定义的变量。

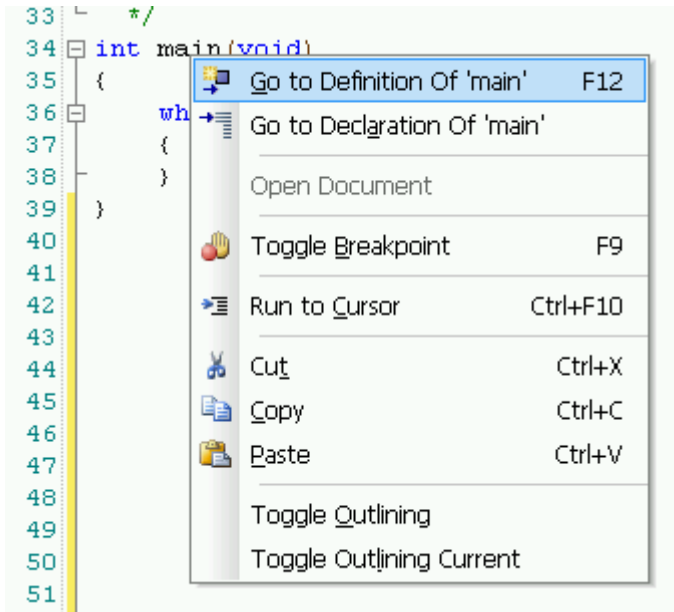
在 IDE 中的菜单



点击【View】->【Class View】，弹出 Class View 窗口

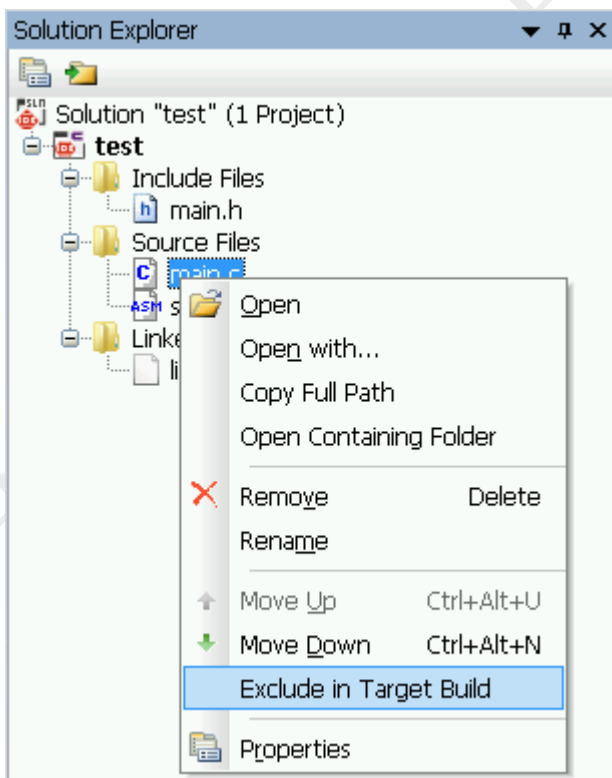



在编辑区中，右键菜单“Go to Definition of '...'”，将跳转至定义行。

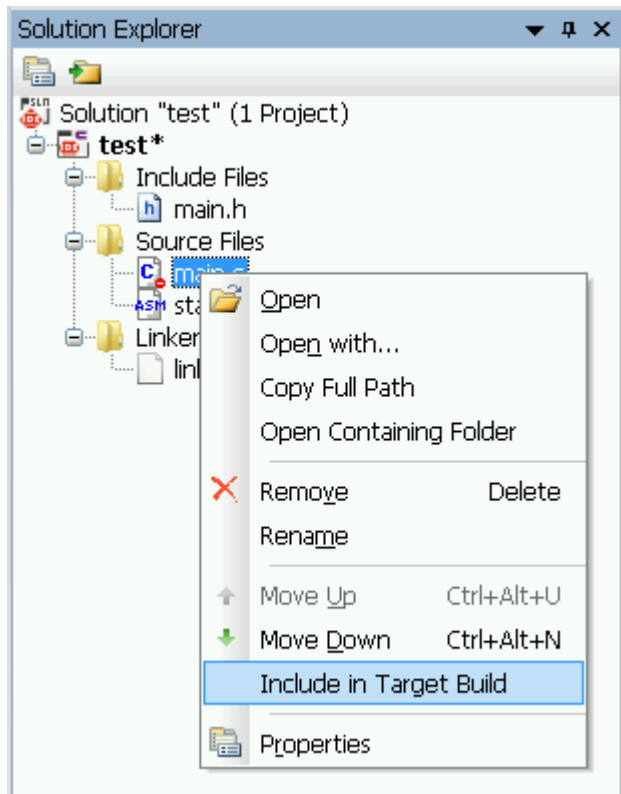


5.12 Include/Exclude in Target Build

在 Solution Explorer 窗口，选中文件时的右键快捷菜单中包含 Exclude in Target Build:



已设置为 Exclude 的文件，文件图标会多一个红色的标识 , 右键菜单为 Include in Target Build:



执行 Build/Rebuild 时，不会编译已设置为 Exclude 的文件。

6 文本编辑

6.1 编辑区窗口

IDE 采用多文档视图的架构，您可同时在编辑器窗口中打开任意多的文档。按住 **Ctrl**，滚动滑鼠，可放大或缩小其字体，按“**Ctrl+F6**”或 **Tab** 键可以在各文档之间进行切换。

编辑区窗口主要功能是用来编辑代码，IDE 还提供了一些辅助编辑的功能。

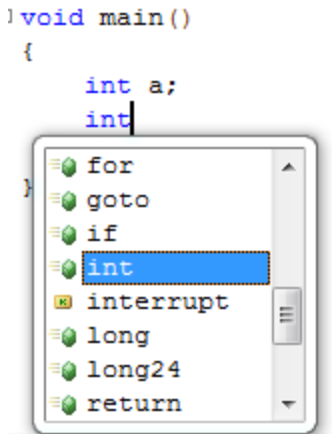
- 关键字着色
- 关键字的智能感应和自动完成功能

- 代码段的缩放
- 更改编辑区的文字大小

注：其它功能请参考菜单和工具栏的详细介绍。

6.2 关键字

在编辑区，输入关键字，会智能感应和自动完成功能，如下图所示：



着色表如下：

项目	颜色
关键字	蓝色
字符串	褐色
数字	紫色
注释	绿色
其它	黑色

6.3 更改文字大小

可依照个人习惯，更改编辑区的文字大小。

光标定位在编辑区，按住 **CTRL** 键，并同时滚动鼠标滑轮，即可放大或缩小编辑区的文字了。

6.4 文件布局

点击 **View | Full Screen**，可以进行全屏显示模式。在该模式下，所有工具栏和状态栏会自动隐藏。按 **Esc** 键或者点击“Close Full Screen”可退出全屏模式。

点击 **Window | New Window**，将在编辑器中创建一个与当前活动文件完全相同的视图，点击多次，则创建多个，并且所有这些文件，将同步改变。

点击 **Window | Rearrange Window**，将重置整个 IDE 界面为默认布局。

6.5 文件保存


在编辑器中，当文件的标题位置有“*”显示，表示文件被改动过且未保存。点击“Save”可保存文件，保存后“*”将自动消失。如果文件未保存状态下直接关闭，将会提示您是否保存。

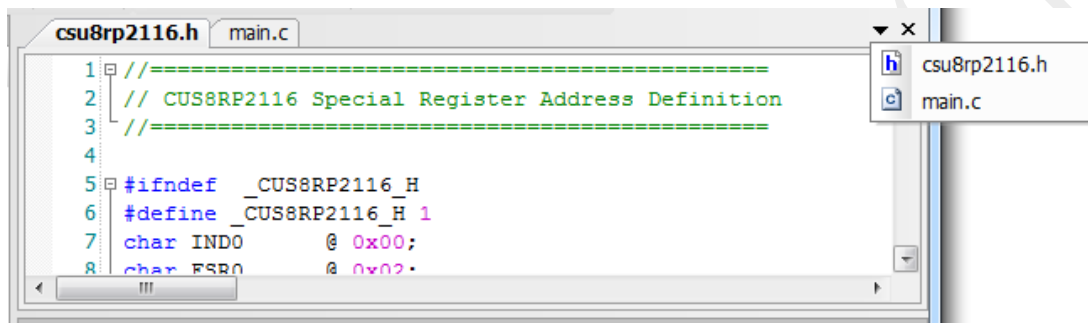
点击 File | Save As，可保存文件为其它文件名；

点击 File | Save All，保存所有打开文件。

6.6 编辑区窗口功能

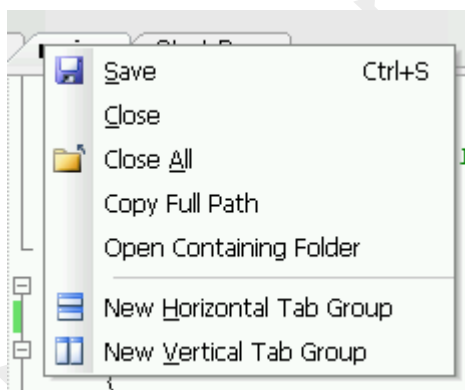
1、切换当前编辑文件

在编辑区窗口右上角，提供一个快速切换当前编辑文件的功能，点击  即可显示当前文件名，如下所示：



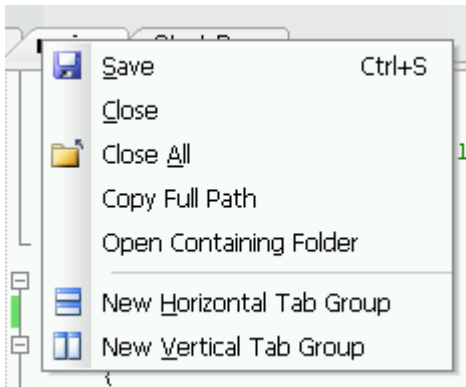
4、标题右键快捷菜单

在编辑区选中一个文件标题，右键快捷菜单如下：



Save	保存该文件。
Close	关闭该文件。
Close All	关闭编辑区所有打开的文件。
Copy Full Path	复制文件路径到剪贴板
Open Containing Folder	打开文件所在的文件夹
New Horizontal Tab Group	新建水平选项卡组。
New Vertical Tab Group	新建垂直选项卡组。

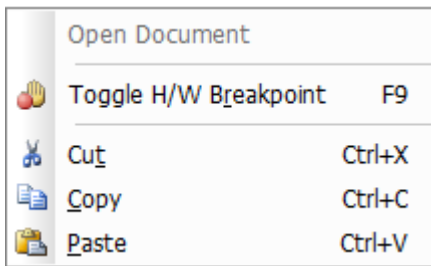
有多个选项卡组时，选中一个文件标题，右键快捷菜单如下：



Move to Next Tab Group: 移动至下一个选项卡组。

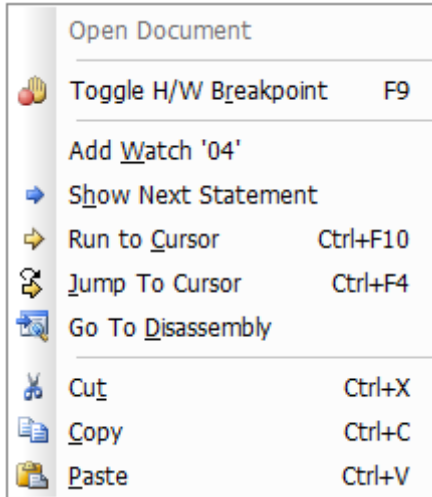
5、编辑区右键快捷菜单




未进入 Debug 状态，在编辑区域内的右键快捷菜单功能如下：



Open Document	选中包含的头文件时，该菜单会亮显，点击该菜单，将会在编辑器中打开该文件。
Toggle H/W Breakpoint 🖐️	插入或删除一个硬件断点。
Cut ✂️	剪切选中内容。
Copy 📄	复制
Paste 📄	粘贴

进入 Debug 状态，在在编辑区域内的右键快捷菜单功能如下：




Open Document	选中包含的头文件时，该菜单会亮起，点击该菜单，将会在编辑器中打开该文件。
Toggle H/W Breakpoint 	插入或删除一个硬件断点。
Add Watch '***'	添加'***'至 Watch 窗口
Show Next Statement 	显示程序的下一条语句。可快速定位到 PC 指针所在行。
Run to Cursor 	运行程序到光标位置。相当于在光标位置设置了一个临时断点。
Jump to Cursor 	直接将 PC 指针跳到新的位置，跳过一段程序后继续进行调试。因为跳过一段程序，执行结果可能和正常调试结果不一致，建议在十分清楚程序逻辑的时候使用。 打开反汇编窗口，并定位到断点对应的反汇编程序
Go to Disassembly 	
Cut 	剪切选中内容。
Copy 	复制
Paste 	粘贴

6.7 选中行功能

在编辑窗口，鼠标左键快速单击 3 次将会选中整行。

6.8 同名的头文件和源文件相互切换功能

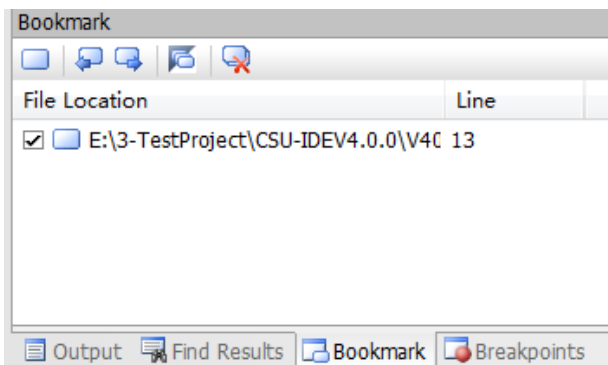
为了方便客户快速查看同名的头文件或源文件，提供了“Open corresponding file”功能。

在编辑区中打开某文件 h/c（例如：A.h），点击工具栏中的  图标，可快速打开同名的 c/h 文件（例如：A.c）。

注：只支持 c 工程中的 h 和 c 文件之间相互切换。

6.9 书签功能（Bookmark）

可以在文本中添加书签，所有书签将会在“Bookmark”窗口中显示，双击书签即可以帮助您快速定位。




设置书签：

将光标移动到目标行，在工具栏“Text Editor”，或者“Bookmark”窗口中点击“Toggle Bookmark”  即可。

在编辑器窗口中，设置为书签的行在左侧会有一个蓝色的标记。

“Previous Bookmark” ：指向“Bookmark”窗口中上一条书签，选中的书签在“Bookmark”窗口中会黑体显示，且会自动关联到源程序。

“Next Bookmark” ：指向“Bookmark”窗口中下一条书签。

“Disable all Bookmark” ：禁用所有书签。

“Clear all Bookmark” ：删除所有书签。

“Previous Bookmark in Current document” ：指向当前文档的上一条书签。

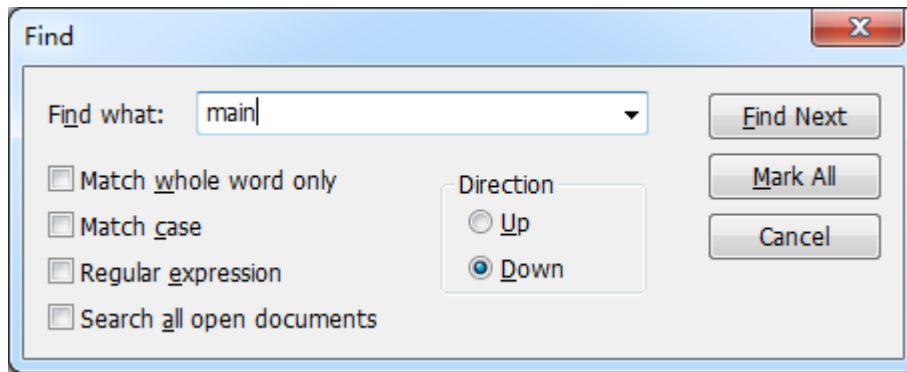
“Next Bookmark in Current document” ：指向当前文档的下一条书签。

6.10 文本查找和替换

IDE 提供了查找和替换功能，支持在指定的文件中查找和替换。

6.11 查找

点击 Edit | Find...，弹出“Find”对话框：



“Find what”：输入所需要查找的文本，支持正则表达式。如果之前有过查找记录，也可直接在下拉列表中选择。

“Match Whole Word Only”：是否仅匹配整个单词。

“Match Case”：是否匹配大小写。

“Regular expression”：是否进行正则表达式的匹配。

“Search all open document”：是否在所有打开文档中进行查找。

上述四项为复选框，可进行多项的组合。但“Regular expression”和“Match whole word only”不能同时选择。

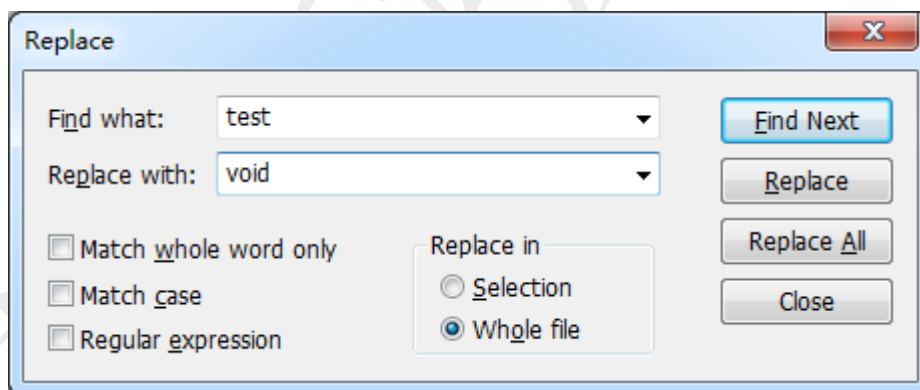
“Up”和“Down”为单选框，用来定义搜索时查找的方向，从当前位置开始向前查找或向后查找。

设置好查找参数和条件后，点击“Find Next”，开始查找匹配的字符串。或使用快捷键 F3，将会使您的查找变得更加方便。

“Mark All”：将所有符合查找条件的行添加书签，并在“Bookmark”窗口中显示。

6.12 替换

点击 Edit | Replace...，弹出“Replace”对话框：



“Find what”、“Match whole word only”、“Match case”、“Regular expression”、“Find Next”，使用方法请参考“查找”章节。

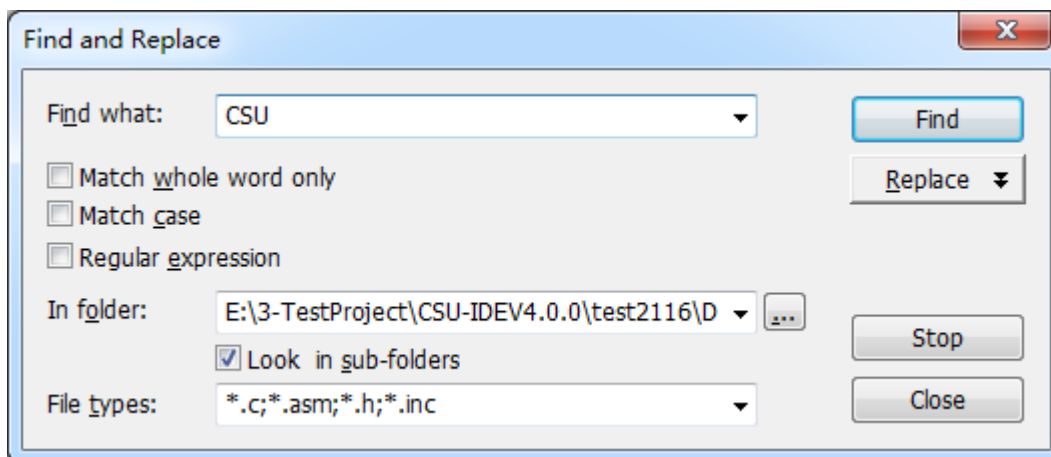
“Replace with”：替换文本编辑框。

“Replace”：进行替换操作。

“Selection”和“Whole file”为单选框，用来设置替换范围，在使用“Replace All”命令时，是对选中区域还是整个文件进行替换。

6.13 在文件中查找/替换

点击 Edit | Find/Replace In Files..., 弹出“Find and Replace”对话框。



“Find what”、“Match whole word only”、“Match case”、“Regular expression”使用方法请参考“查找”章节。

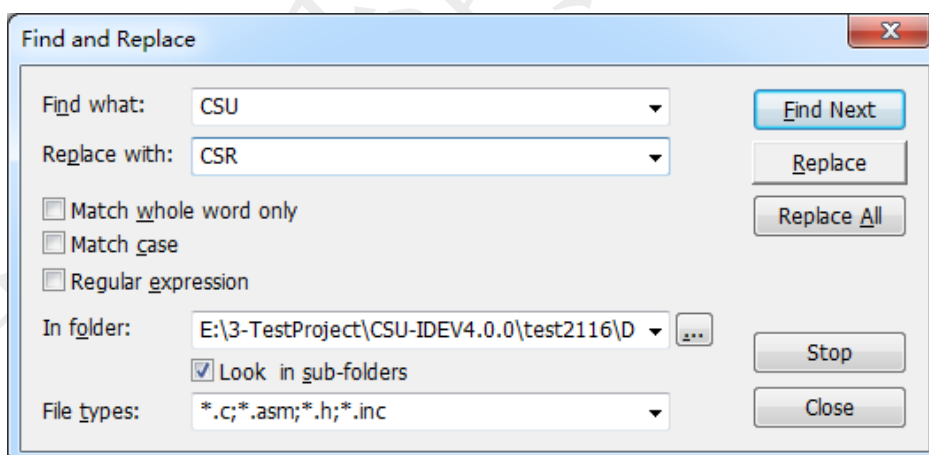
“In folder”：用来设置文件夹的位置，默认为当前工程所在目录。可以更改需要查找的路径，还可以选择在整个解决方案（Entire Solution）、当前活动工程(Active Project)、选择工程(Selected Project)中查找。

“Look in sub-folder”：是否在子文件夹中搜索。

“File types”：设置文件的查找类型，设置后，仅在符合类型的文件中进行搜索。您可以在下拉框中选择您需要的文件类型，或者直接编辑，编辑格式请参考下拉列表中的内容。

“Find”：开始查找，并将所有结果输出到“Find Result”窗口中。

点击“Replace”，“Find and Replace”对话框将会多出替换相关项目，如下图：



“Replace”：仅替换当前文本。

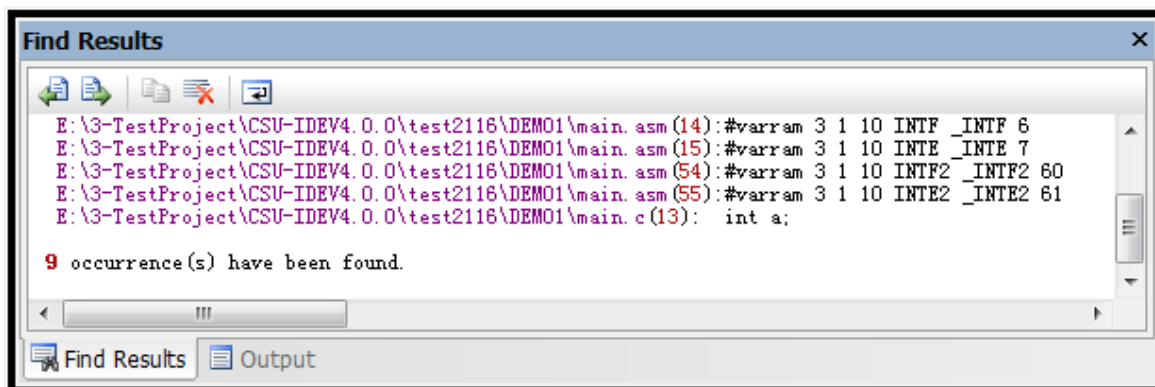
“Replace All”：替换文件夹中所有符合条件的文本。

“Stop”：查找过程中点击“Stop”，提示搜索。

“Close”：关闭对话框。

6.14 Find result 窗口

查找的结果会输出到“Find Result”窗口中，如下图所示：



“Go to Previous Message” ：指向上一个查找结果

“Go to Next Message” ：指向下一个查找结果

“Copy Line” ：复制“Find Result”窗口中选中的内容

“Clear All” ：清空“Find Result”窗口的内容

“Toggle Word Wrap” ：切换自动换行。

注：在“Find Result”窗口中，双击某条查找结果，在编辑区会快速定位该查找结果。

7 调试器

本章将介绍一下与调试相关的操作与窗口功能。

7.1 下载程序

在进行调试之前，需先将编译通过的程序代码下载到仿真版。

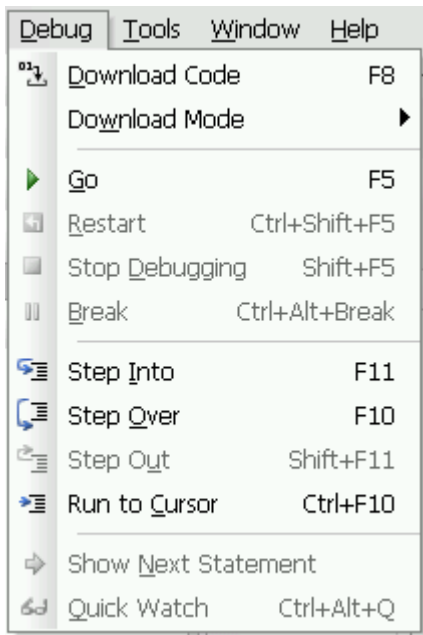
点击 Debug| Debug Run，会执行下载的步骤。

下载完毕，黄色光标会停在程序起始地址 0x00 或 main 函数的起始位置。

7.2 调试操作

下载完成后，即可开始进行调试。

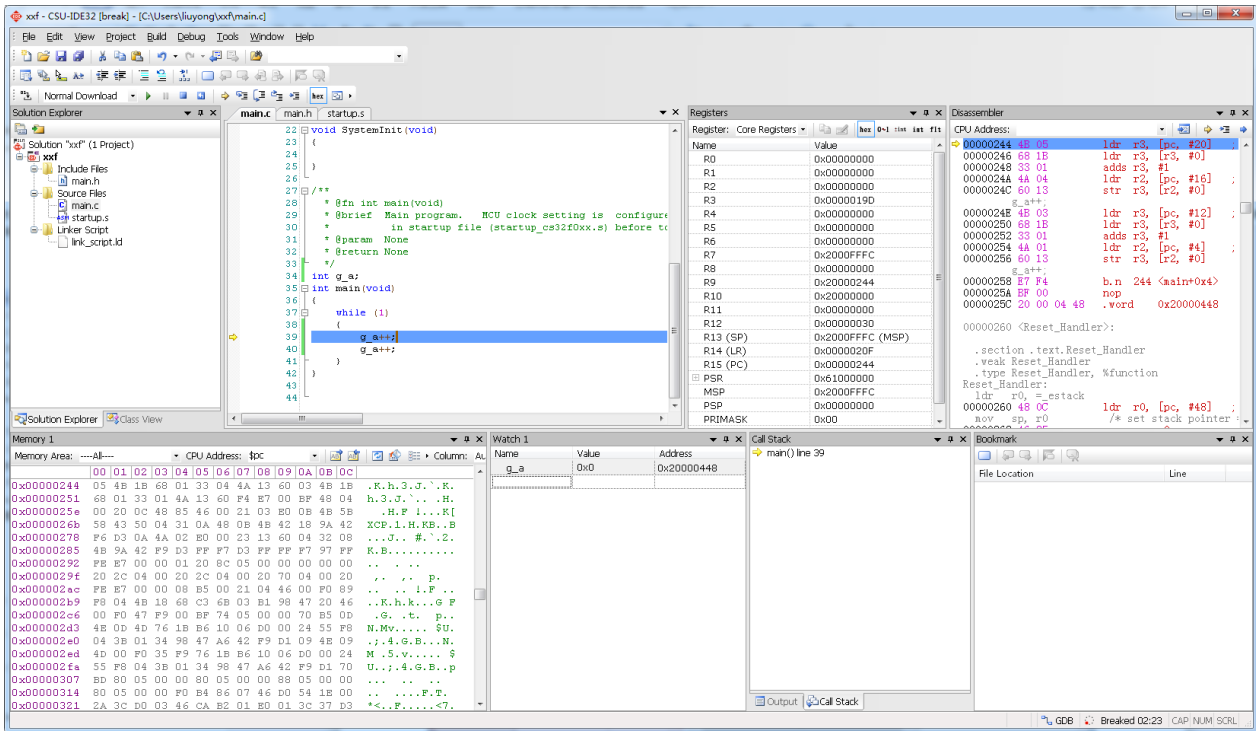
先介绍一下 Debug 菜单功能，如下图所示：



Go	下载程序；如果程序已经下载，则会运行程序，并遇到断点停下来。
Restart	重新运行程序，之前的运行结果将不被保留。
Stop Debugging	退出调试模式，转为编辑模式。
Break	暂停程序运行
Step Into	单步步入，可进入子程序
Step Over	单步步过，不进入子程序
Step Out	单步步出，可跳出子程序。
Run to Cursor	程序全速运行到光标处停止。
Show Next Statement	显示程序的下一条语句。可快速定位到 PC 指针所在行。


7.3 调试窗口（Debug Window）

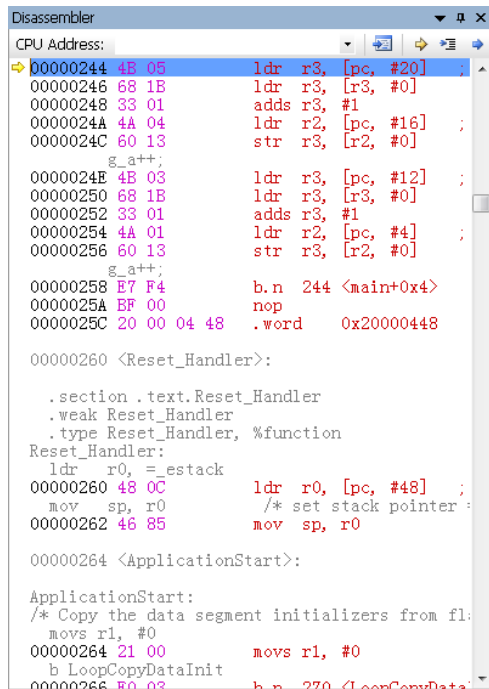
调试窗口主要用来显示或修改当前监视到的变量、内存、寄存器的值以及反汇编结果等。调试状态下，可以在“Debug”工具栏或 View | Debug Windows 菜单中打开或关闭各个调试窗口。



窗口	作用
Disassembler	对下载的目标文件进行反汇编操作并显示，显示的为汇编级别语言
Register	显示通用寄存器，CPU 状态寄存器和一些功能寄存器的值
Memory	显示当前 RAM/ROM 内容
Watch	显示变量名、变量值、类型或地址
Breakpoint	显示软、硬件断点信息，及进行断点设置等操作
Call Stack	显示调用堆栈

7.4 反汇编窗口（Disassembler）

点击 View | Debug Windows | Disassembler，或点击“Debug”工具栏的  即可打开反汇编窗口。



```

Disassembler
CPU Address:
00000244 4B 05 ldr r3, [pc, #20]
00000246 68 1B ldr r3, [r3, #0]
00000248 33 01 adds r3, #1
0000024A 4A 04 ldr r2, [pc, #16]
0000024C 60 13 str r3, [r2, #0]
    g_a++;
0000024E 4B 03 ldr r3, [pc, #12]
00000250 68 1B ldr r3, [r3, #0]
00000252 33 01 adds r3, #1
00000254 4A 01 ldr r2, [pc, #4]
00000256 60 13 str r3, [r2, #0]
    g_a++;
00000258 E7 F4 b.n 244 <main+0x4>
0000025A BF 00 nop
0000025C 20 00 04 48 .word 0x20000448


00000260 <Reset_Handler>:
    .section .text.Reset_Handler
    .weak Reset_Handler
    .type Reset_Handler, %function
Reset_Handler:
    ldr r0, =_estack
00000260 48 0C ldr r0, [pc, #48]
    mov sp, r0 /* set stack pointer
00000262 46 85 mov sp, r0

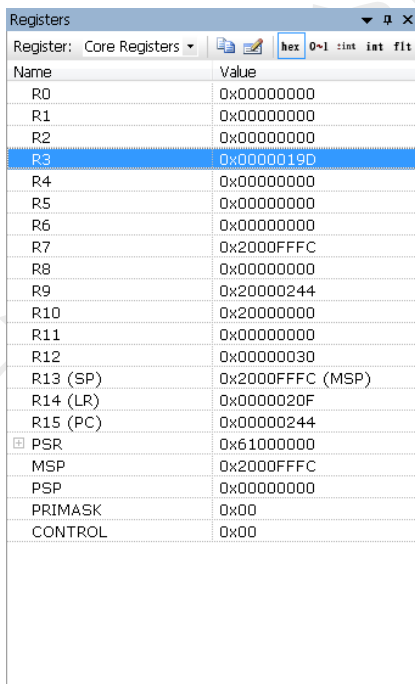
00000264 <ApplicationStart>:
ApplicationStart:
/* Copy the data segment initializers from fl
    movs r1, #0
00000264 21 00 movs r1, #0
    b LoopCopyDataInit
00000266 F0 03 b.n 270 <LoopCopyData
    
```

在反汇编窗口中会显示源代码、反汇编代码。

在工具栏的“Address” 输入框中输入地址值，按回车将会直接定位到该地址所在行。

7.5 寄存器窗口 (Register)

点击 **View | Debug Windows | Register**，或点击“Debug”工具栏的  即可开启寄存器窗口。



Name	Value
R0	0x00000000
R1	0x00000000
R2	0x00000000
R3	0x0000019D
R4	0x00000000
R5	0x00000000
R6	0x00000000
R7	0x2000FFFC
R8	0x00000000
R9	0x20000244
R10	0x20000000
R11	0x00000000
R12	0x00000030
R13 (SP)	0x2000FFFC (MSP)
R14 (LR)	0x0000020F
R15 (PC)	0x00000244
PSR	0x61000000
MSP	0x2000FFFC
PSP	0x00000000
PRIMASK	0x00
CONTROL	0x00

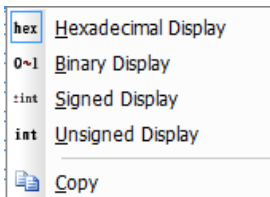
寄存器窗口会来显示 PC、Cycle、状态寄存器和其它功能寄存器的值。

双击 **Value** 可编辑寄存器的值(寄存器的值是否可以修改，由寄存器的属性决定)。修改寄存器的值后，仿真版也会相应

变化。

R0-R15	通用寄存器
PSP	PSP 寄存器
MSP	MSP 寄存器
PRIMASK	PRIMASK 寄存器
CONTROL	控制寄存器

右键快捷菜单如下图所示：

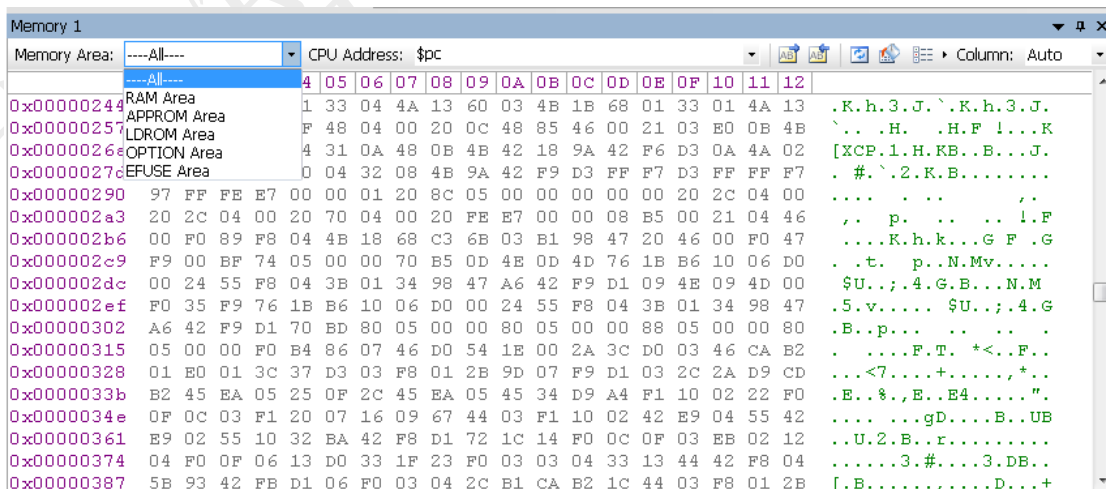


hex	使用十六进制数显示，其它窗口如 watch、Memory 窗口也会相应变化
0~1	二进制数显示
sint	有符号的十进制数显示
int	无符号的十进制数显示
Copy	复制

点击“Register”的下拉框 ，可以查看或修改其它寄存器地址。

7.6 Memory 窗口

在 View | Debug Windows

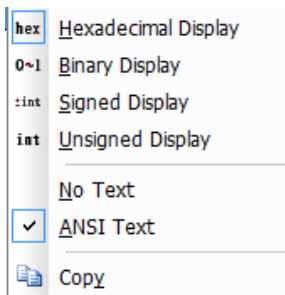


在工具栏的“Address” 输入框中输入地址值，按回车将会直接定位到该地址所在行。

在工具栏“Column” 下拉列表框中选择每行显示的 byte 个数。

“Open the file after the export”：导出完毕之后文件将会在编辑器区域打开该文档。

右键快捷菜单，如下：



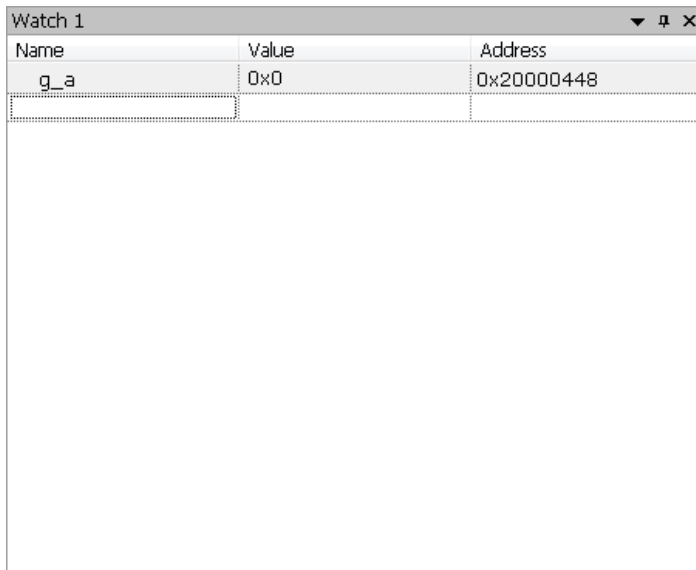
提供了不同的进制显示方式，包括十六进制（Hexadecimal Display）、二进制（Binary Display）、有符号（Signed Display）、无符号 Unsigned Display）；文本显示方式、排列方式，以及复制、编辑功能等。

hex		使用十六进制数显示，其它窗口如 watch、Memory 窗口也会相应变化
0~1		二进制数显示
sint		有符合的十进制数显示
int		无符号的十进制数显示
No Text		不显示文本
ANSI Text		文本用 ANSI 方式显示
Copy		复制

注：E2PROM、ROM 窗口中的值禁止编辑。

7.7 变量窗口（Watch）

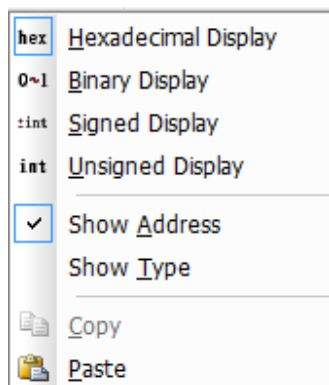
点击 View | Debug Windows | Watch，或点击“Debug”工具栏的 即可开启变量窗口。



“Watch”窗口可用来监视变量、寄存器的值，类型或地址。可以在“Name”列输入需要查看的变量或符号，也可直接从其它文本拖至“Name”中，“Watch”窗口会自动显示对应的值。可以在“Value”列中修改变量的值，修改后“RAM”窗口和仿真版上的值会同步变化。

程序运行过程中，如果监视到的值有被改变，将会红色显示。

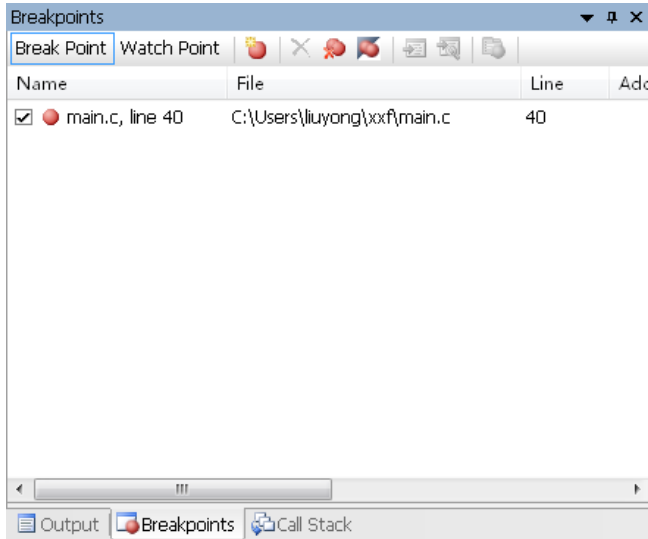
右键快捷菜单如下图所示：



hex	hex	使用十六进制数显示，其它窗口如 watch、Memory 窗口也会相应变化
0~1	0~1	二进制数显示
sint	:int	有符合的十进制数显示
int	int	无符号的十进制数显示
Show Address		显示数据地址
Show Type		显示数据类型
Copy		复制
Paste		粘贴

7.8 断点窗口 (Breakpoint)

“Breakpoint”窗口，可设置硬件断点，包括指令地址中断和数据中断，最多支持 5 个硬件指令地址中断和 3 个数据中断。








Name: 显示断点类型、文件信息、行号信息、地址信息等

Address: 断点地址信息



Trigger On: 所有数据断点的逻辑关系，包含与和或，点击 **Trigger On(&&)** 可相互切换。

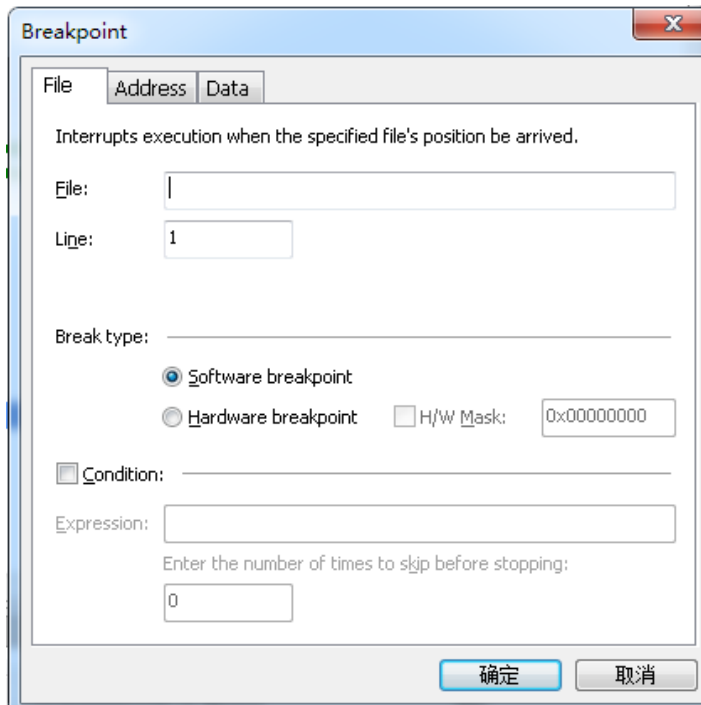
||: 所有数据断点只要有一个数据断点满足触发条件，则触发数据断点。


&&: 所有数据断点全部满足触发条件，才会触发数据断点。

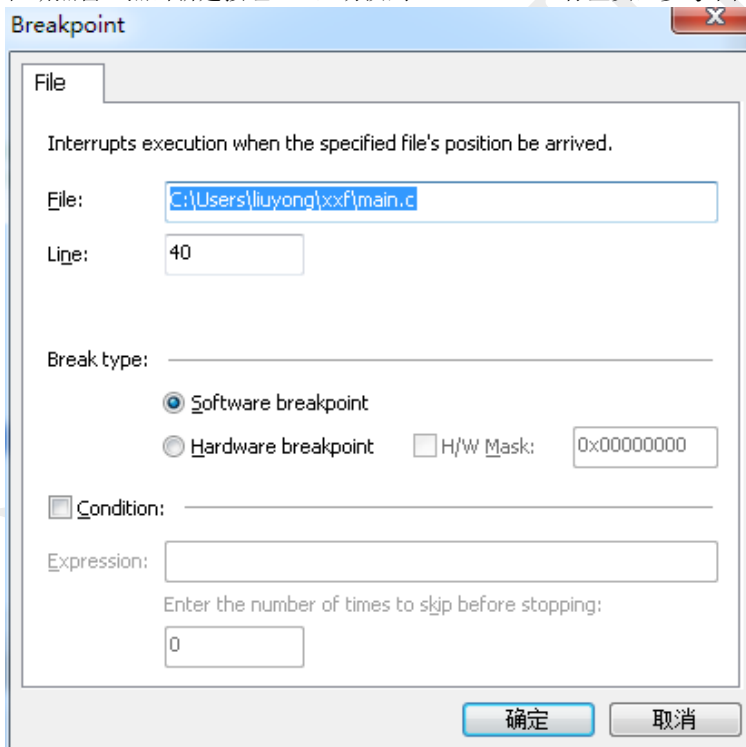
New 	新建一个断点
Delete 	删除选中的断点
Delete All 	删除所有断点
Disable All 	禁用所有断点，禁用后，程序将不会在断点位置产生中断。
Edit 	编辑断点信息

断点设置方式

- 1、在编辑器窗口或反汇编窗口中的指令行按“F9”
- 2、在编辑器窗口或反汇编窗口中的指令行，点击“Build”工具栏上图标 ；
- 3、在断点窗口点击新建按钮 ，在“HW File”编辑框中输入文件的绝对路径，在“Line”编辑框中输入行号，点击“确定”；



4、在断点窗口点击新建按钮 ，切换到“Hardware”标签页，参考下图



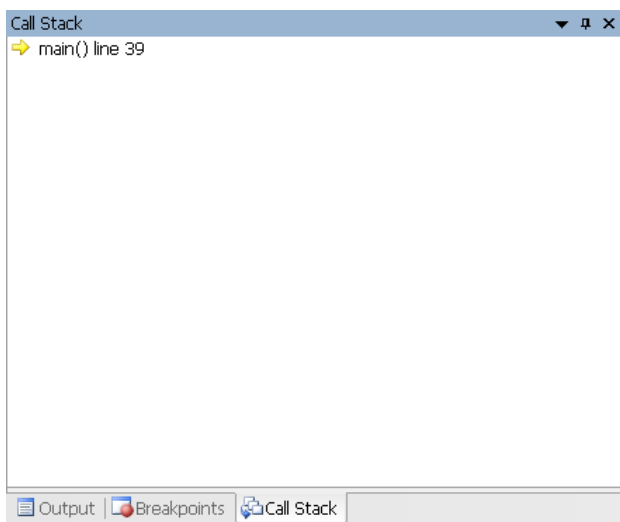
下面分别介绍每一项的作用：

Instruction Break	设置指令中断
Address	设置要中断的地址
Data Break	设置数据断点
Address	设置被监视的数据的地址
Data	从指定地址读取的值 data 符合 Trigger Condition 时，将产生数据中断

Trigger Condition 选择触发条件：==、!=、>、<、>=、<=。

7.9 调用栈窗口（Call Stack）

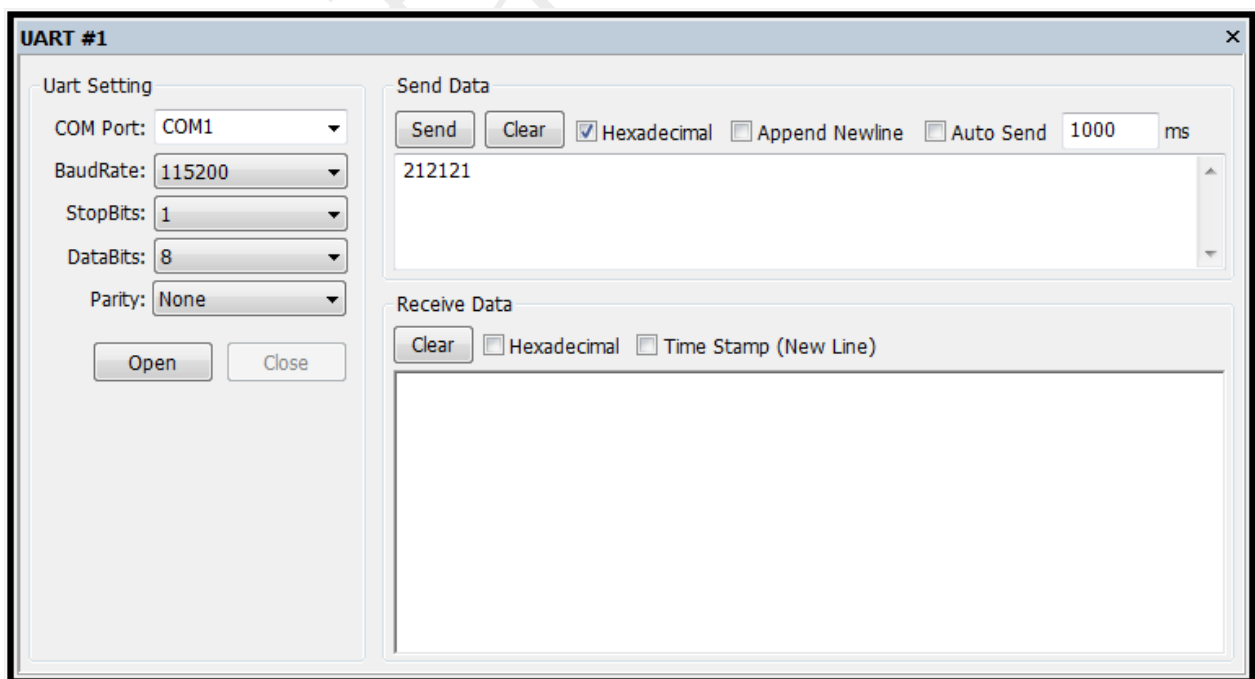
当仿真程序有压栈操作时（如 call 调用子函数、进入中断等），call stack 提供进入子函数的入口地址。当使用 return 等出栈后，call stack 会自动删除该记录。用户可使用 call stack 功能清楚了解函数的调用关系。



7.10 串口辅助调试窗口

当调试串口输出程序时，可以直接使用 IDE 提供的工具进行收发数据。

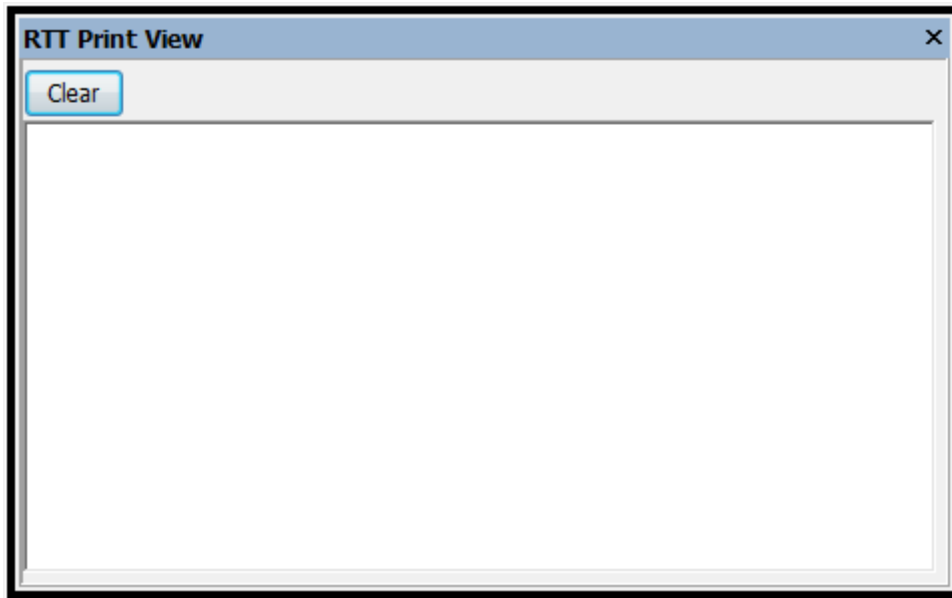
打开窗口：【View】=》【Debug Window】=》【UART】



7.11 RTT 辅助调试窗口

RTT 打印数据显示可以在此窗口进行显示。

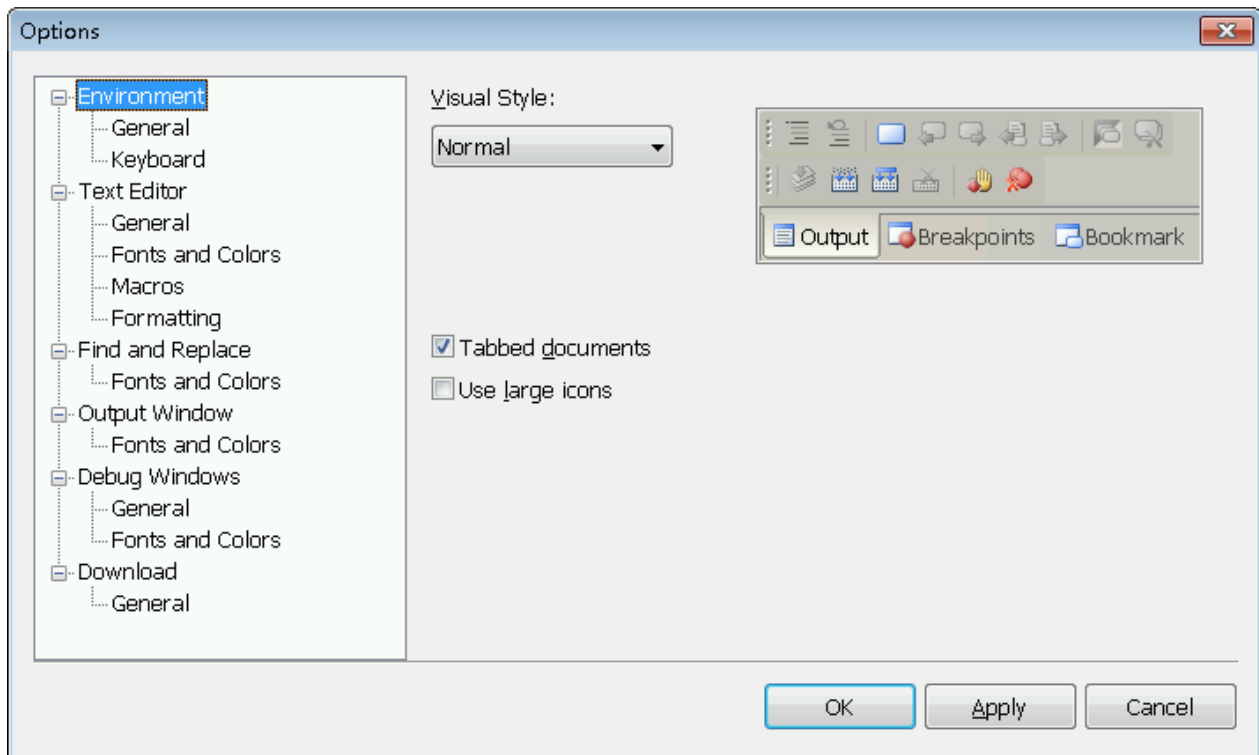
打开窗口：【View】=》【Debug Window】=》【RTT Print View】



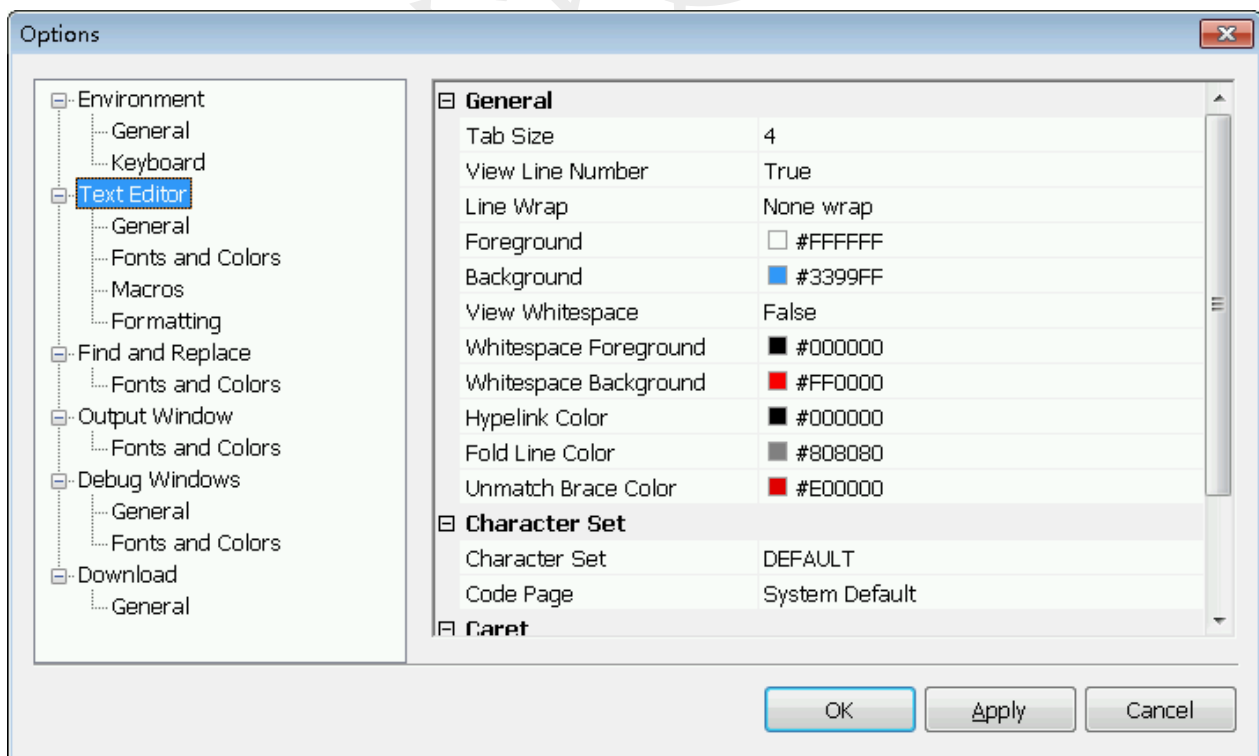
8 用户自定义

提供了用户自定义功能，用户可根据个人习惯来定义 IDE 的用户环境。

点击 **Tools | Options** 将会弹出“Options”对话框。



8.1 常用设置 (General)



Tab Size: 按“Tab”，将会跳几格，设置范围为 1~8。

View Line Number: 设为“True”，将会有行号显示。

True	False
<pre> 1 .RAM 2 uint8 i; 3 uint8 R_LedFlag; 4 </pre>	<pre> .RAM uint8 i; uint8 R_LedFlag; </pre>

Line Wrap

None wrap: 当一行的内容超过编辑器框后，不会换行。

According to word wrap: 当一行的内容超过编辑器框后，会从当前单词开始换行。

According to character wrap: 当一行的内容超过编辑器框后，从当前字符开始换行。

Foreground: 编辑器中选中文本的前景色。

Background: 编辑器中选中文本的背景色。

For example

Selection Foreground	nop
Selection Background	goto \$-1

View Whitespace: 显示空格和“Tab”键的位置。

False	True
<pre> 11 org 000h 12 nop 13 nop 14 goto \$-1 15 16 end -- </pre>	<pre> 11 → org 000h 12 → nop 13 → nop 14 → goto \$-1 15 16 → end </pre>

Whitespace Foreground: 空格和“Tab”键头的前景色。

Whitespace Background: 在空格和“Tab”键位置的背景色，参考上图。

Hypelink Color: 设置超链接的下划线颜色，如下图：

```
http = "http://www.chipsea.com"
```

Fold Line Color: 设置代码折叠线的颜色

```

1 |
2 | filename: PrjType1_16_4.asm
3 | chip : CSU18M83
4 | author :
5 | date : 2016-07-07
6 | http = "http://www.chipsea.com"
7 |
                
```

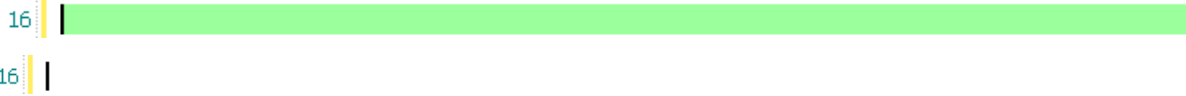
Unmatch Brace Color: 设置括号不匹配时，所显示的颜色

Character Set: 用来设置字符集

Caret

Foreground: 设置光标的颜色

Line Hilight: “True”时，光标所在行将高亮显示，对比下图。

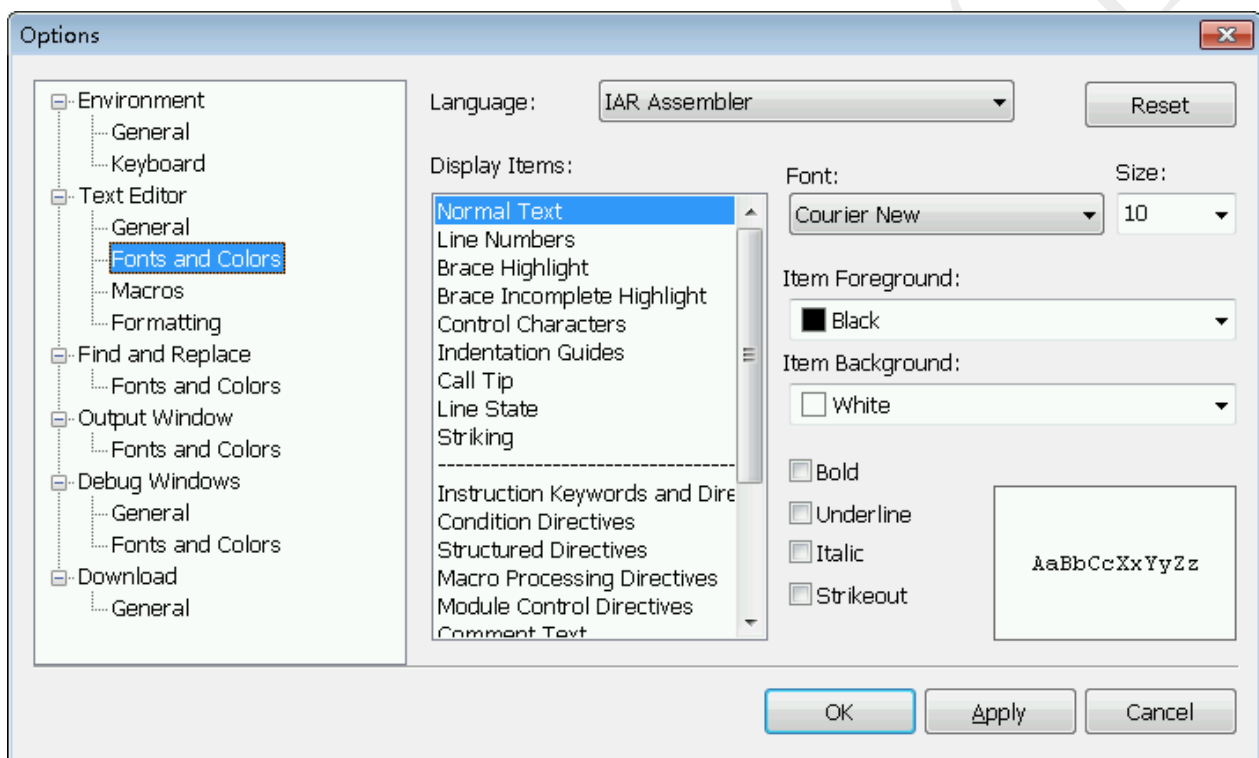


Line Background: 设置光标高亮时的背景色。

Line Background Alpha: 设置光标高亮时的透明度。

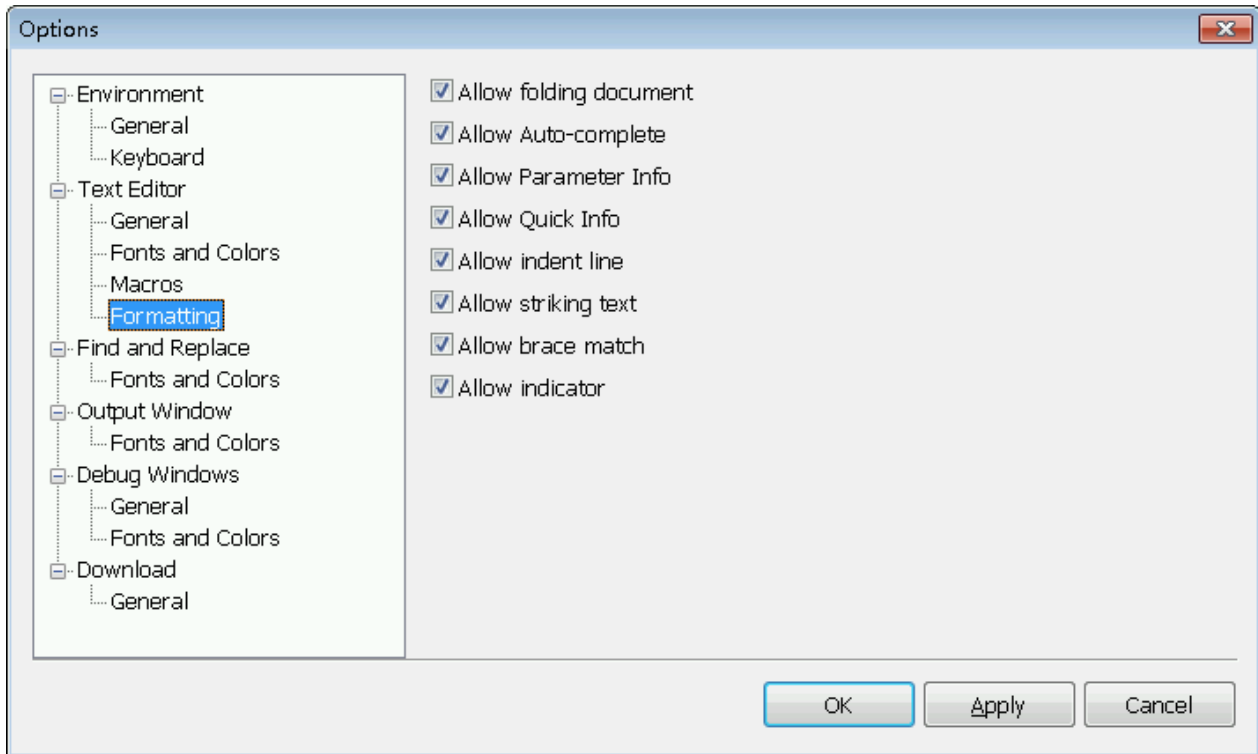
8.2 字体和颜色设置 (Fonts and colors)

提供了不同的字体和颜色设置方式。



8.3 格式定义 (Formatting)

提供了不同的字体和颜色设置方式。



Allow Auto-complete: 是否允许自动完成

Allow folding document: 是否允许折叠文档

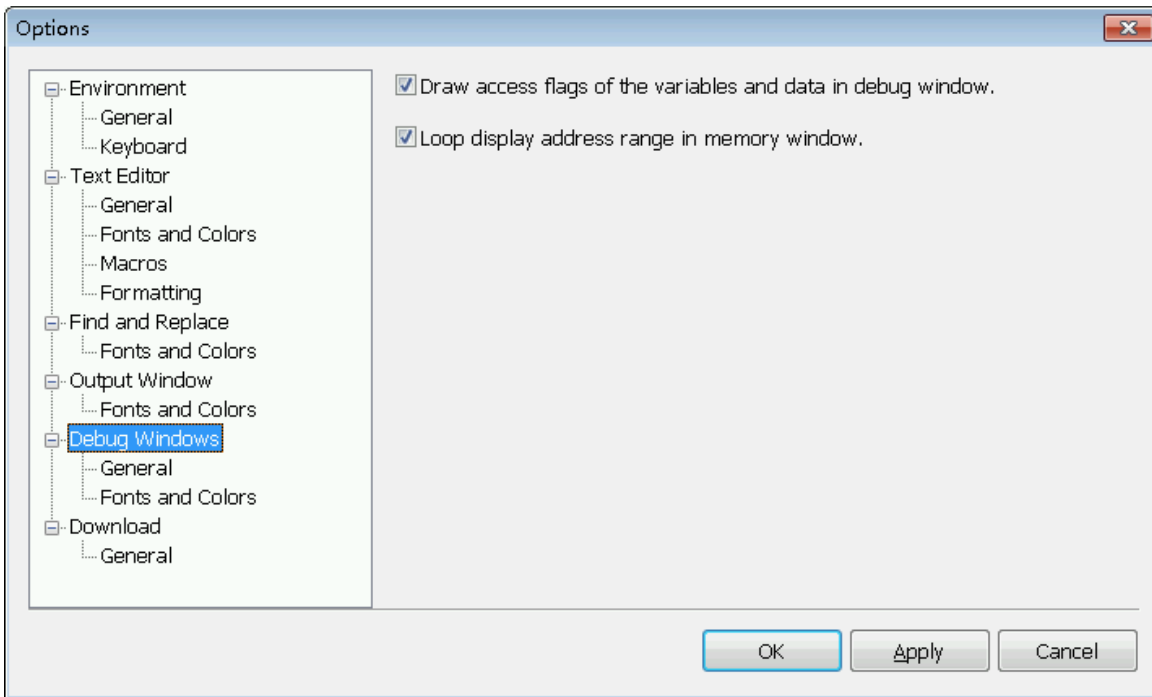
Allow Parameter Info: 是否允许显示参数信息

Allow Quick info: 是否允许快速显示关键字消息

Allow indent line: 是否允许行缩进

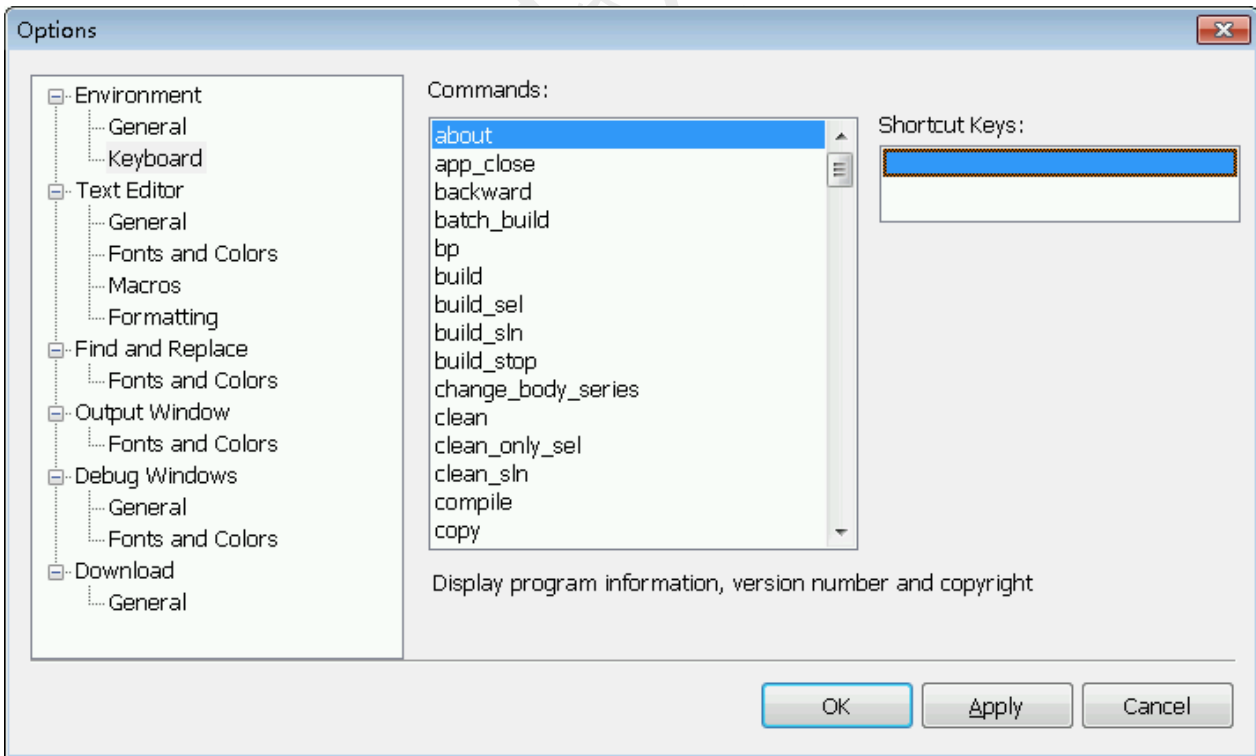
Allow brace match: 是否允许括号匹配

8.4 调试设置 (Debugging)



用户可以在此页面设定调试窗口的显示，通常使用默认值。

8.5 工具快捷菜单设置



用户可以修改系统默认的快捷键，选中需要改变的命令项，双击右列表，输入快捷键即可完成配置。

9 菜单项介绍

菜单项包括“File, Edit, View, Project, Build, Debug, Window, Help”。下面简要描述了每个项目的功能。

File	新建、打开、保存、关闭解决方案或工程或文件等相关操作。
Edit	编辑和查找相关操作。
View	控制各个窗口的显示。
Project	工程管理等相关设置。
Build	工程编译相关设置。
Debug	下载和调试相关操作。
Window	对编辑窗口的操作，排列等。
Help	使用手册、版本信息等。

9.1 文件 (File)

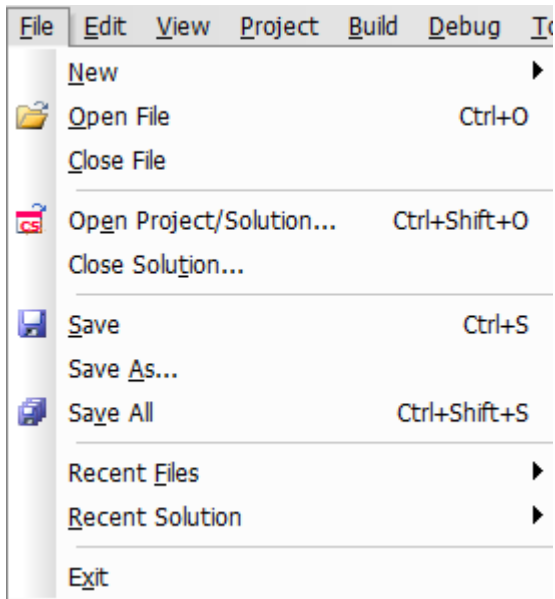


表 1 File 菜单

内容	作用	默认热键
New	(Project)新建工程	Ctrl+Shift+N
	(File)新建文件	Ctrl+N
Open File	(Blank Solution)新建空的解决方案	
	打开已经存在的文档	Ctrl+O
Close File	关闭编辑窗口中当前活动文档	
Open Project Solution...	打开已有解决方案	Ctrl+Shift+O
Close Solution...	关闭当前解决方案	
Save	保存编辑窗口中当前活动文档	Ctrl+S
Save As...	将编辑窗口中当前文档另存为	
Save All	保存所有文档及工程配置	Ctrl+Shift+S
Recent Files	列出最近打开文件，最多记忆 8 个	
Recent Solution	列出最近打开解决方案，最多记忆 8 个	
Exit	关闭 IDE	

9.2 编辑 (Edit)

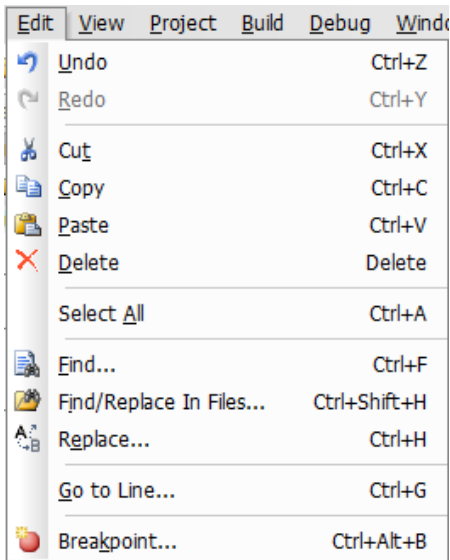


表 2 Edit 菜单

内容	作用	默认热键
Undo	撤销上次操作	Ctrl+Z
Redo	恢复被撤销的操作	Ctrl+Y
Cut	剪切	Ctrl+X
Copy	复制	Ctrl+C
Paste	粘贴	Ctrl+V
Delete	删除	Delete
Select All	选择整个文件内容	Ctrl+A
Find...	文本查找	Ctrl+F
Find/Replace In Files...	在指定的目录下进行文本查找和替换	Ctrl+Shift+H
Replace...	文本替换	Ctrl+H
Go to Line...	定位到指定行	Ctrl+G
Breakpoint...	设置断点	Ctrl+Alt+B

9.3 视图 (View)

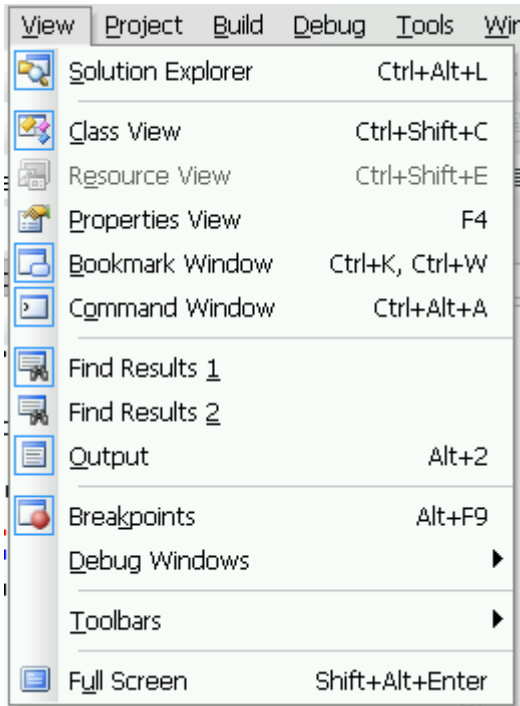


表 3 View 菜单

内容	作用	默认热键
Solution Explorer	显示解决方案管理器窗口	Ctrl+Alt+L
Class View	符号窗口	Ctrl+Shift+C
Properties View	显示属性窗口	F4
Bookmark Window	显示标签窗口	Ctrl+K, Ctrl+W
Find Results	显示查找结果窗口	
Output	显示输出窗口	Alt+2
Breakpoints	显示断点窗口	Alt+F9
Debug Windows	显示调试窗口，包括：	
	(1)变量窗口	Alt+3
	(2) RAM(PAGE0)窗口	Alt+4
	(3) RAM(PAGE1)窗口	Alt+5
	(4)E2PROM 窗口	Alt+6
	(5)反汇编窗口	Alt+7
	(6)寄存器窗口	Alt+8
	(7)Rom 窗口	

Toolbars	显示或隐藏工具条。包括： 标准工具条 文本编辑工具条 调试工具条 编译工具条	
Full Screen	全屏显示	Shift+Alt+Enter

9.4 工程 (Project)

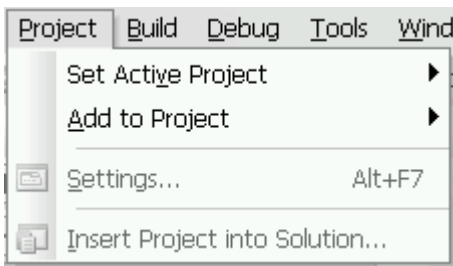


表 4 Project 菜单

内容	作用	默认热键
Set Active Project	设置选中工程为活动工作	
Add to Project	添加新文件、新文件夹或已存在的文件到工程中	
Settings...	工程设置	Alt+F7
Insert Project into Solution...	在当前解决方案中加入一个已存在的工程	

9.5 构建 (Build)

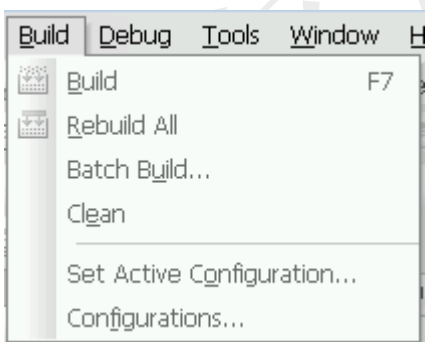


表 5 Build 菜单

内容	作用	默认热键
Build *.hex	构建工程	Ctrl+F7
Rebuild All	清除编译或重构过程中产生的文件，然后重新构建工程	F7

Batch All	编译全部的解决方案
Clean	清除编译或重构过程中产生的文件

9.6 调试 (Debug)

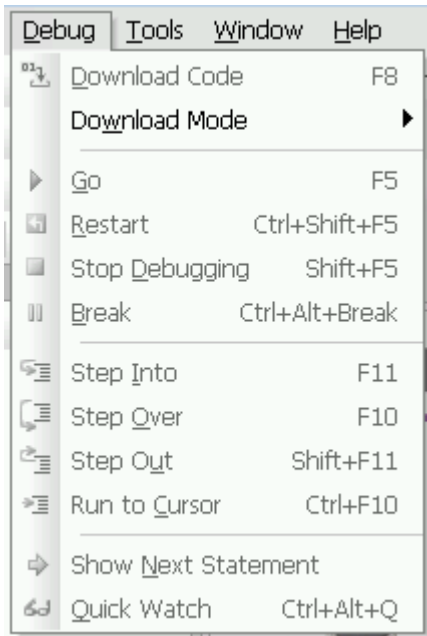
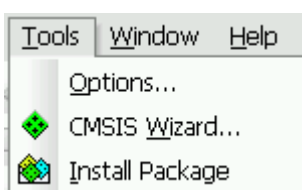


表 6 Debug 菜单

内容	功能	默认热键
Go	下载程序；如果程序已经下载，则会运行程序，并遇到断点停下来。	F5
Restart	重新运行程序，之前的运行结果将不被保留。	Ctrl+ Shift+F5
Stop Debugging	退出调试模式，转为编辑模式。	Ctrl +F5
Step Into	单步步入，可进入子程序	F11
Step Over	单步步过，不进入子程序	F10
Step Out	单步步出，可跳出子程序。	Shift+F11
Run to Cursor	程序全速运行到光标处停止。	F4
Show Next Statement	定位到 PC 指针所在行	

9.7 工具 (Tools)



内容	作用	默认热键
Options	定制开发环境，如热键的设置等	
CMSIS Wizard	CMSIS 工程向导	
Install Package	安装 CMSIS 型号包	

9.8 窗口 (Window)

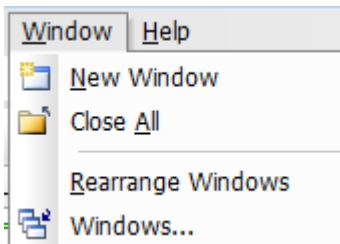


表 7 Window 菜单

内容	作用	默认热键
New Window	在编辑器中创建一个与当前活动文件完全相同的新文件	
Close All	关闭编辑器中所有打开的文件	
Rearrange Windows	重新排列窗口布局	
Windows	进行 windows 相关的设置	

9.9 帮助 (Help)

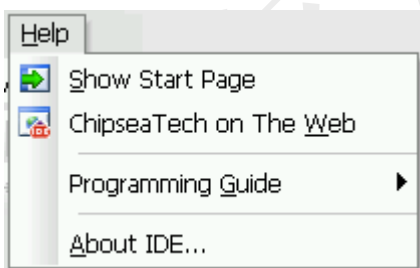


表 8 Help 菜单

内容	作用	默认热键
Show Start Page	在编辑器中显示起始页。	
Chipsea on the Web	在编辑器中打开公司网页。	
Programming Guide	IDE 使用手册。	
About IDE	显示当前版本信息。	

10 工具栏介绍

IDE 在工具栏提供了 4 个工具条，分别是标准、文本编辑、调试和构建。每个工具栏存在三种状态：隐藏、停靠或者悬浮。

10.1 标准工具条 (Standard)



表 1 标准工具条









图标	内容	功能
	New File	新建文本
	Open	打开已有文本
	Save	保存当前文本
	Save All	保存所有文本及工程设置
	Cut	剪切
	Copy	复制
	Paste	粘贴
	Undo	撤销上次操作
	Redo	恢复被撤销的操作
	Find in Files	在指定的目录下进行文本查找和替换

10.2 文本编辑条 (TextEditor)



表 2 文本编辑工具条

图标	内容	功能
	Decrease Indent	减小缩进
	Increase Indent	增加缩进
	Comment Line	添加行注释

	Uncomment Line	删除行注释
	Toggle Bookmark	创建/删除一条书签
	Previous Bookmark	指向书签窗口中的上一条书签
	Next Bookmark	指向书签窗口中的下一条书签
	Previous Bookmark in Current Document	指向当前文本中的上一条书签
	Next Bookmark in Current Document	指向当前文本中的下一条书签
	Disable all Bookmark	禁用所有书签
	Clear all Bookmark	删除所有书签

10.3 构建工具条 (Build)










表 3 构建工具条

图标	内容	作用
	Select Active Configuration	点击下拉框选择所需的工程配置
	Build	构建工程
	Rebuild All	清除编译或重构过程中产生的文件，然后重新构建工程
	Stop Build	停止构建
	Remove All Breakpoint	删除所有断点
	CMSIS Wizard	CMSIS 工程向导
	Install Package	CMSIS 型号包安装器

10.4 调试工具条 (Debug)



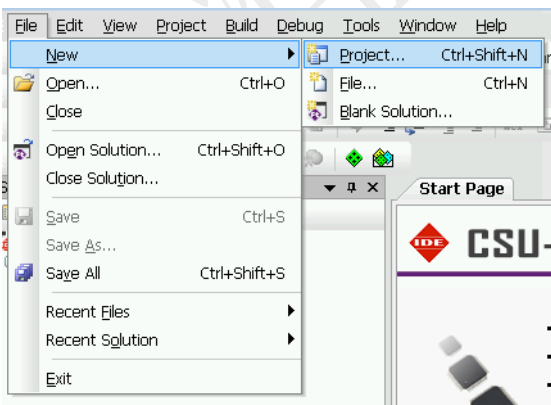
表 4 调试工具条

图标	内容	作用
	Download	下载程序
Normal Download ▾	Debug Download Mode	下载模式：正常下载/不下载
	Go	下载程序：如果程序已经下载，则会运行程序，并遇到断点停下来。
	Break	暂停程序运行
	Stop Debugging	退出调试模式，转为编辑模式。
	Restart	重新运行程序，之前的运行结果将不被保留。
	Step Into	单步步入，可进入子程序
	Step Over	单步步过，不进入子程序
	Step Out	单步步出，可跳出子程序。
	Run to Cursor	程序全速运行到光标处停止。
	Debug Windows	点击弹出调试窗口

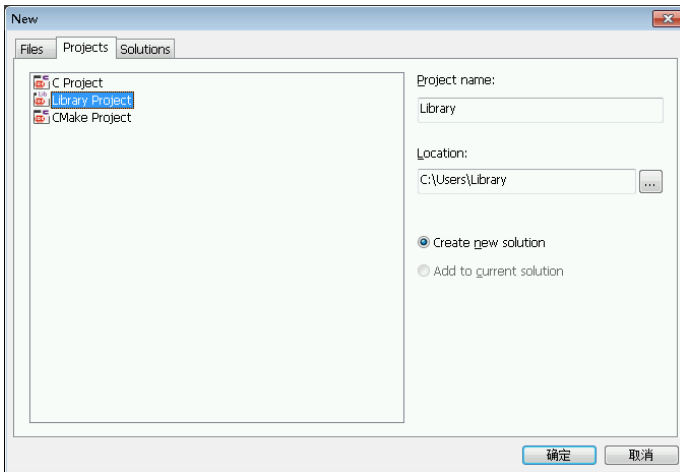
11 Library Project

11.1 创建工程

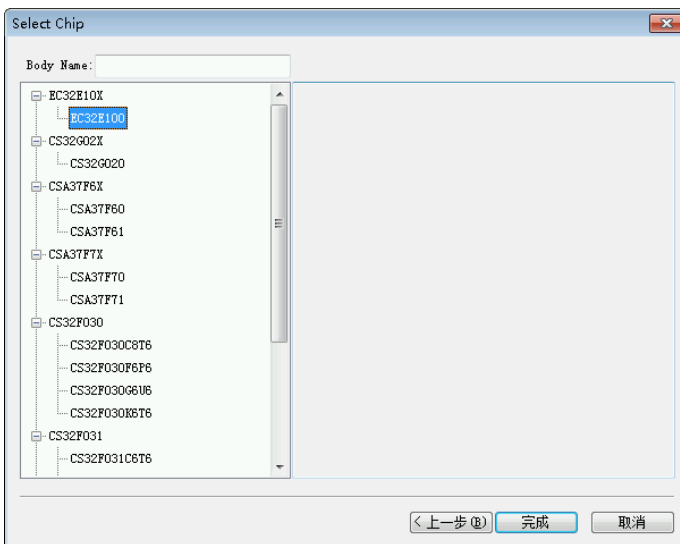
点击【File】=>【New】=>【Create Project】



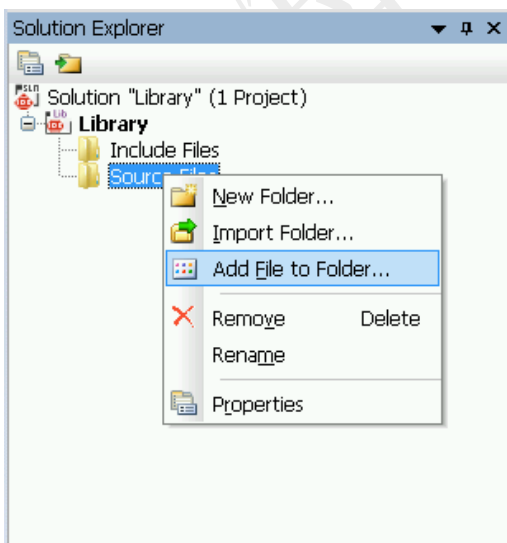
2、选择 Libray 类型



3、选择芯片型号

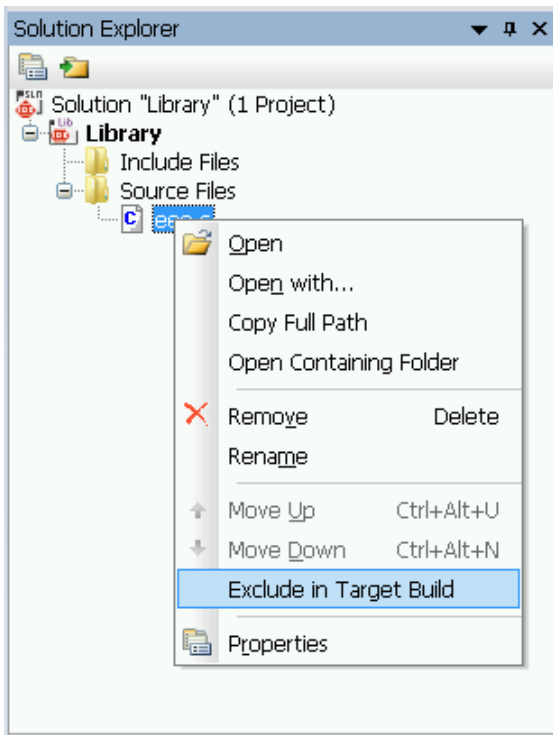


4、添加文件到工程



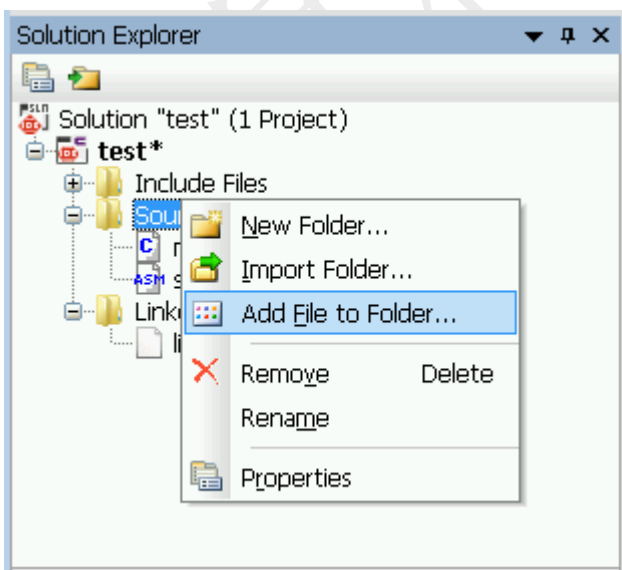
5、点击【编译】按钮，编译生成库文件

注：如果有文件已经加到工程上，但又不想打包到库中，可以使用排除编译功能，就可以不编译此文件

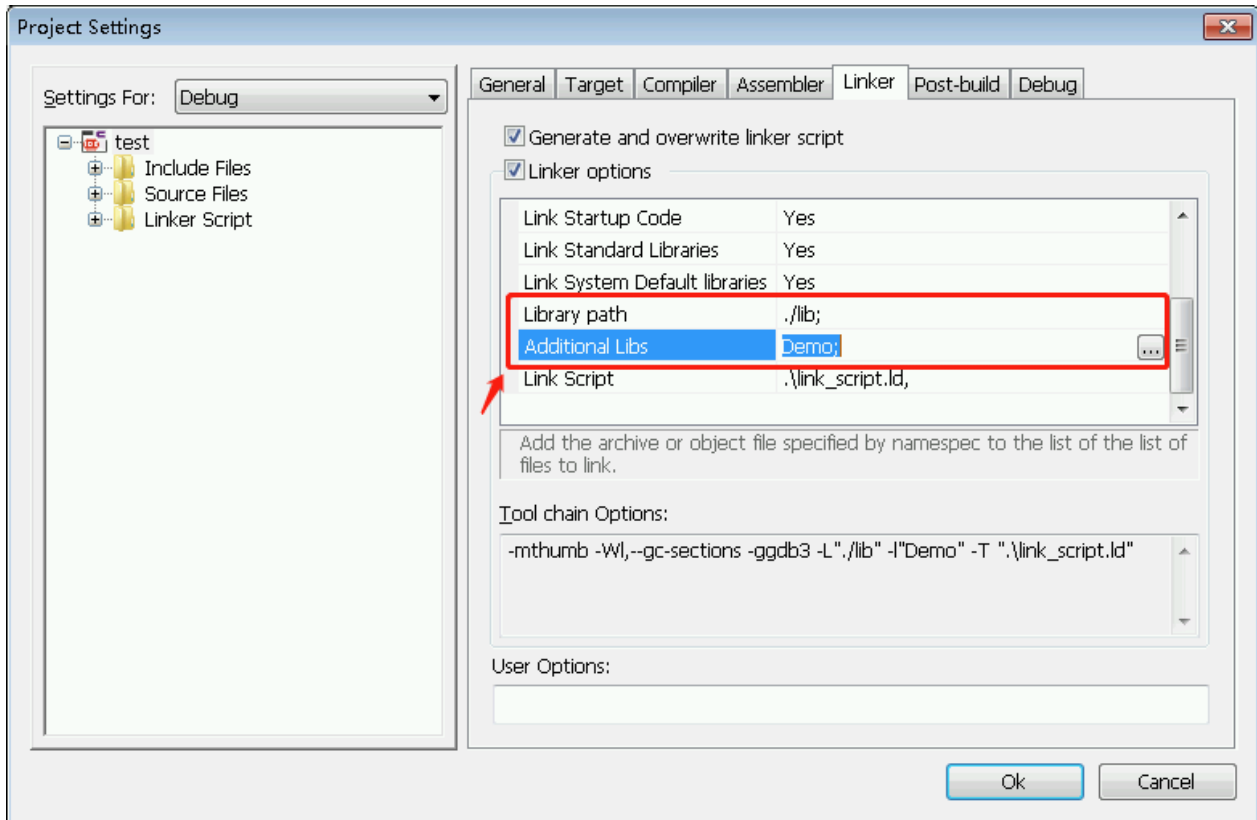


11.2 使用库文件

1、将.a 文件直接添加到工程文件列表中，编译时会自动链接这个库文件



2、链接配置页面增加包含库路径和库文件

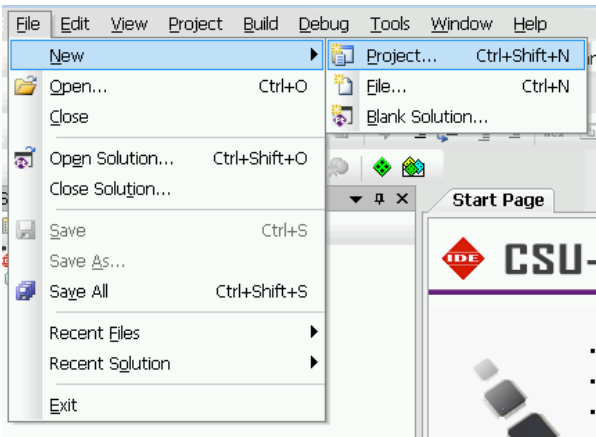


12 CMake Project

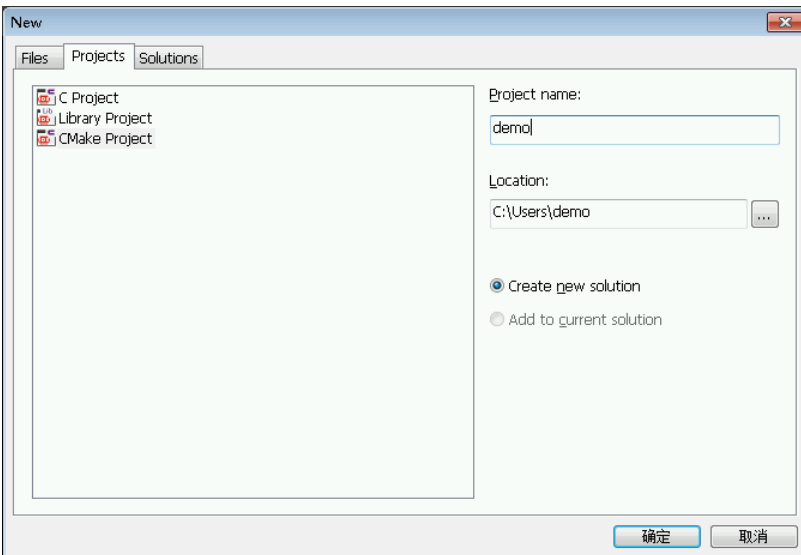
CMake 工程的用途，有些项目已经使用 CMake 建立好了编译环境，只需要工具来处理编辑和调试的功能，类似于 VSCode 基于目录管理项目的方式。

12.1 创建工程

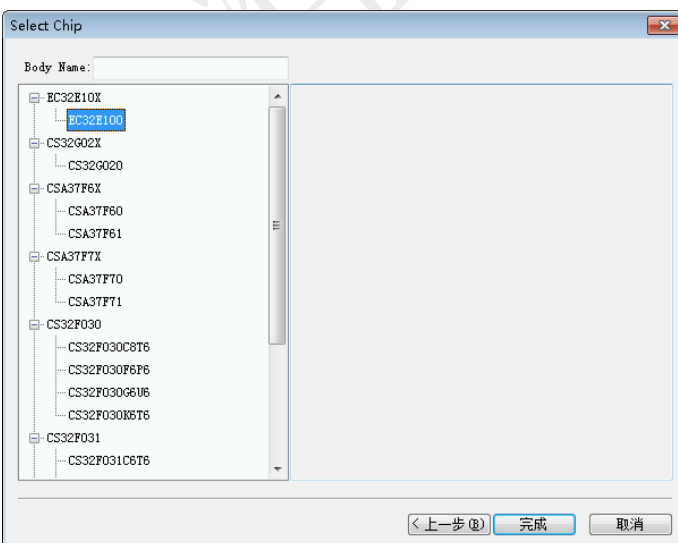
点击 **【File】 => 【New】 => 【Create Project】**



2、选择 CMake 类型



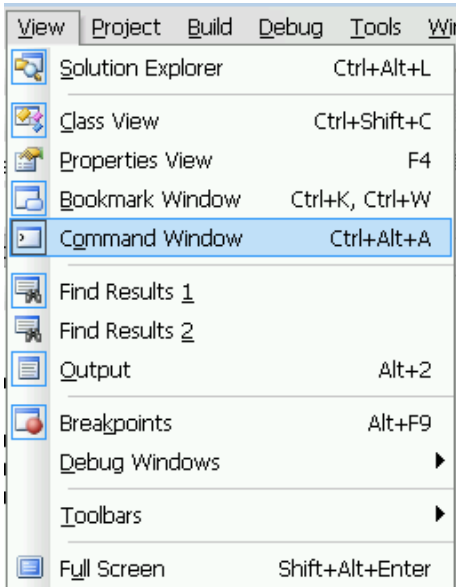
3、选择芯片型号



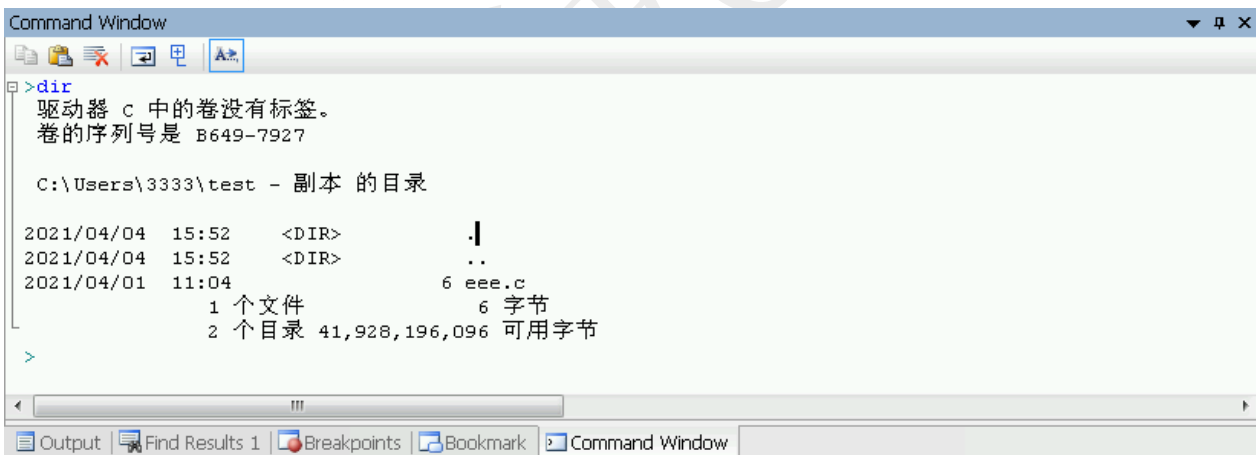
12.2 执行编译命令

12.2.1 命令行窗口

1、打开命令行窗口

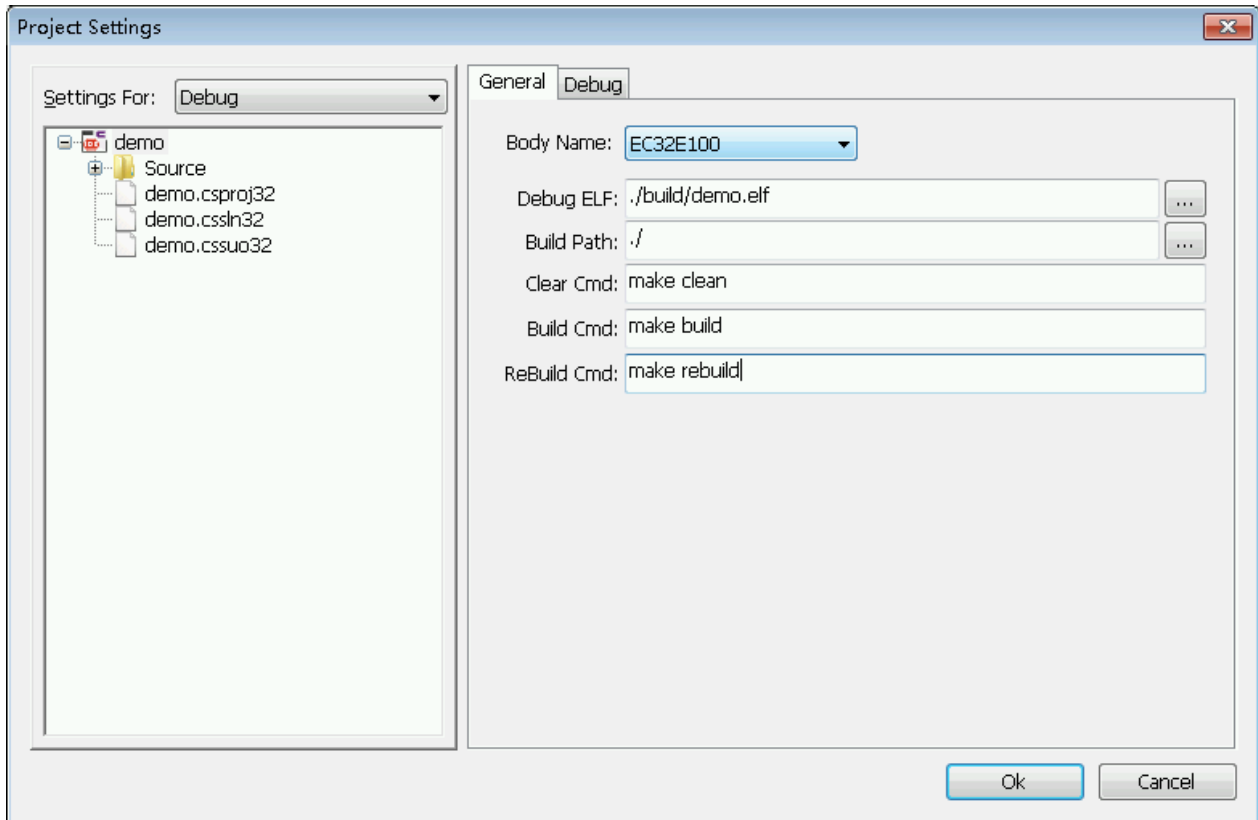


2、在命令行窗口中输入命令



12.2.2 配置 Build/Rebuild/Clean 按钮命令

1、打开工程配置窗口

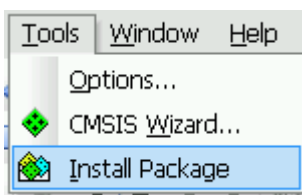


注：可以执行的命令和 Windows 的命令行一样，用户可以设置环境变量后执行扩展命令，如果要调试程序，则需要指定 elf 文件路径。

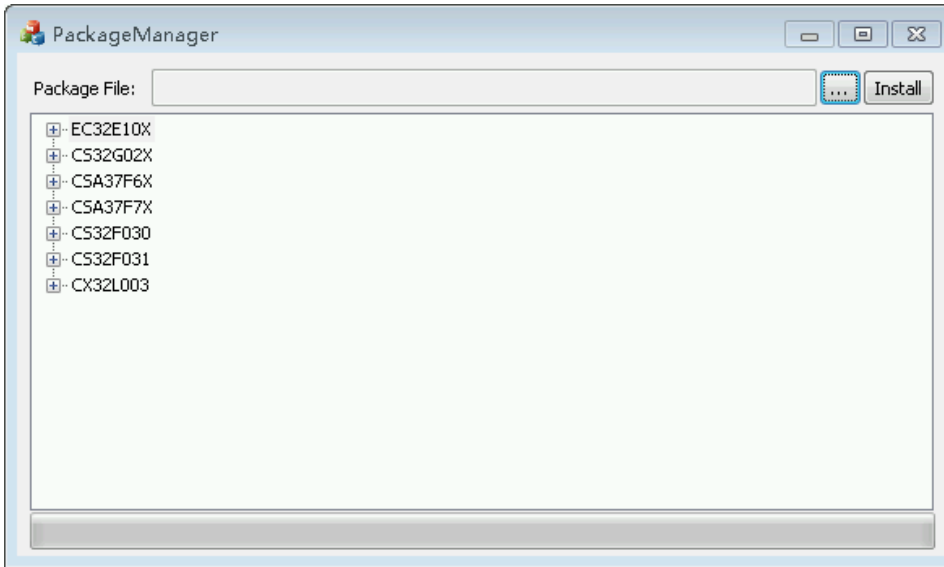
13 Install CMSIS Package


13.1 界面

点击【Tools】-【Pack Installer】，如下图：



会弹出对话框，如下图所示：



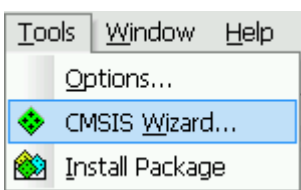
：选择安装包路径，文件格式后缀为*.pack

：点击 Install，执行安装。

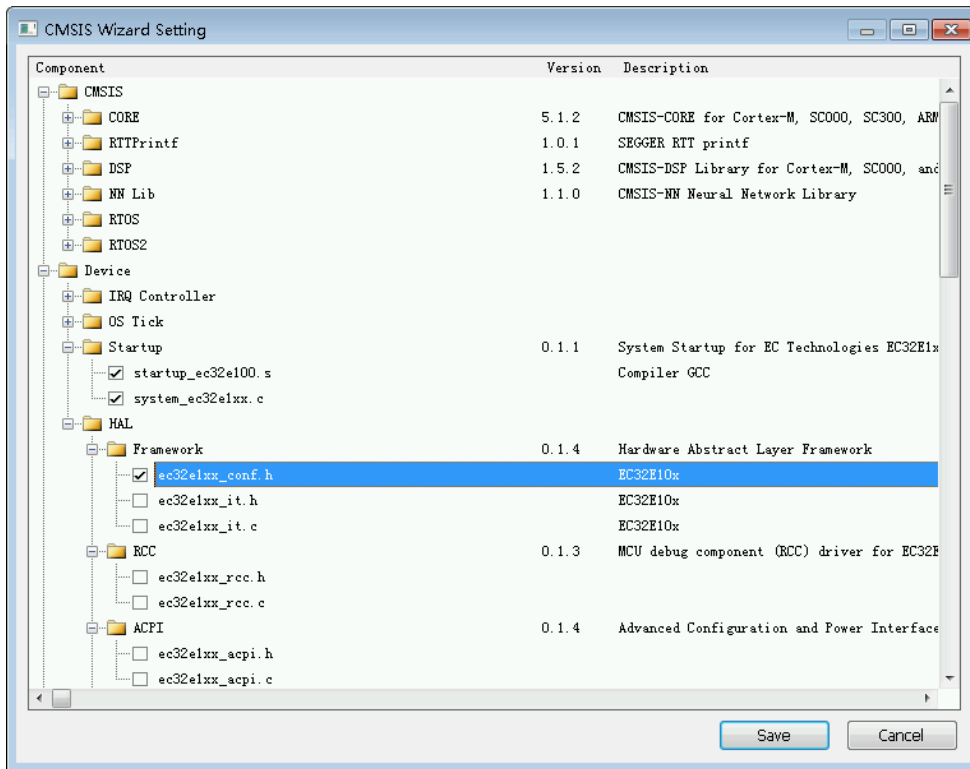
14 CMSIS Wizard

14.1 打开工具

点击【Tools】=》【Install Package】，如下图：



会弹出对话框：

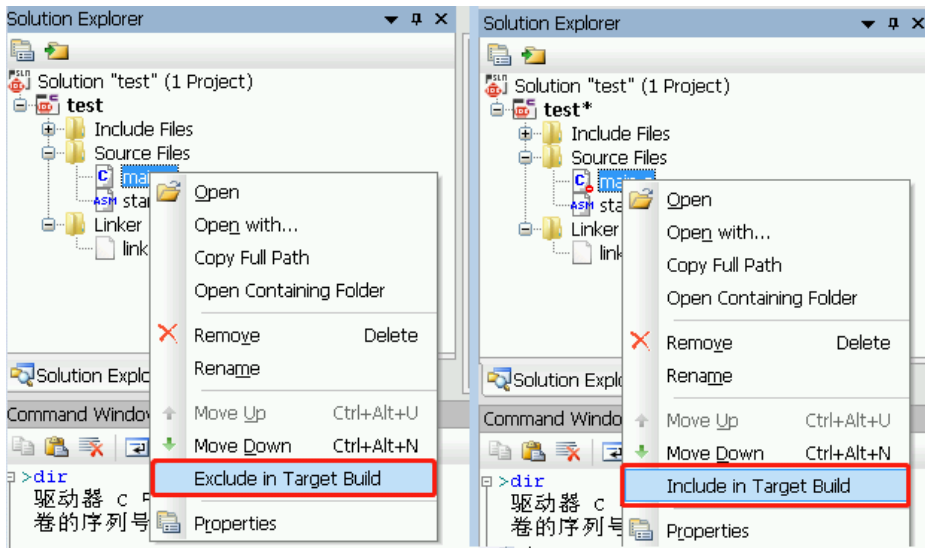


注：用户可以勾选组件，点击【Save】按钮后，自动加入组件到工程中。

15 常用技巧

15.1 排除指定的文件不进行编译

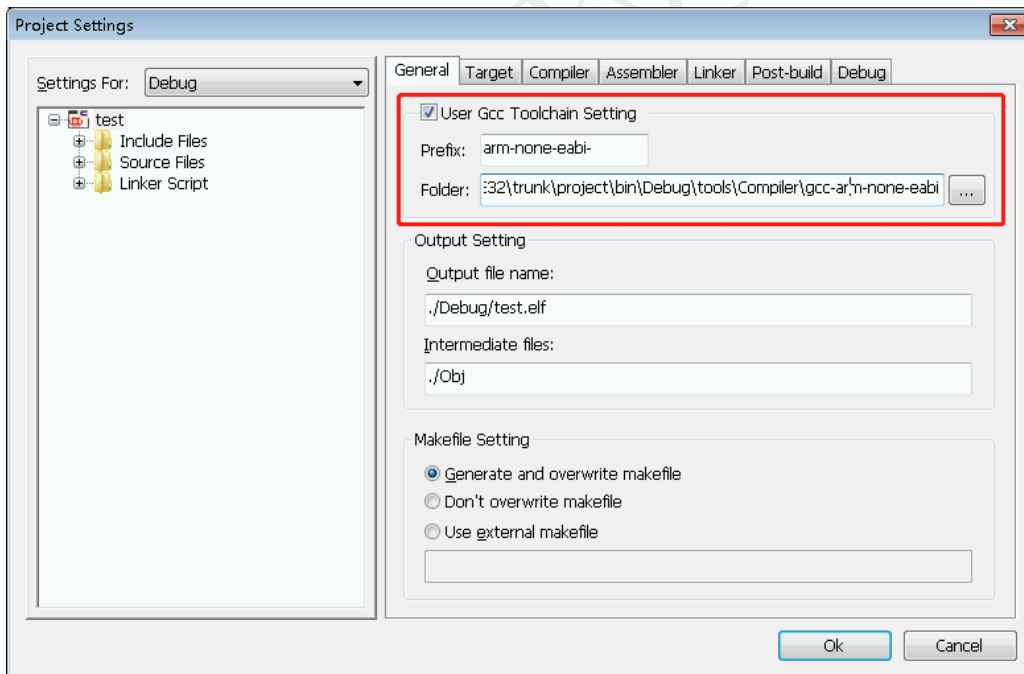
有些时候用户想要排除掉一些文件如：.C/.S 文件，但又想把他放在工程文件列表上管理。右击文件项，点击菜单【Exclude in Target Build】项。



注：被排除编译的文件在其右下角会有一个红色标记，如果恢复编译，再次右键文件项，中的【Include in Target Build】菜单。

15.2 修改默认 GCC 编译工具链

1、打开工程设置界面，指定 GCC 路径

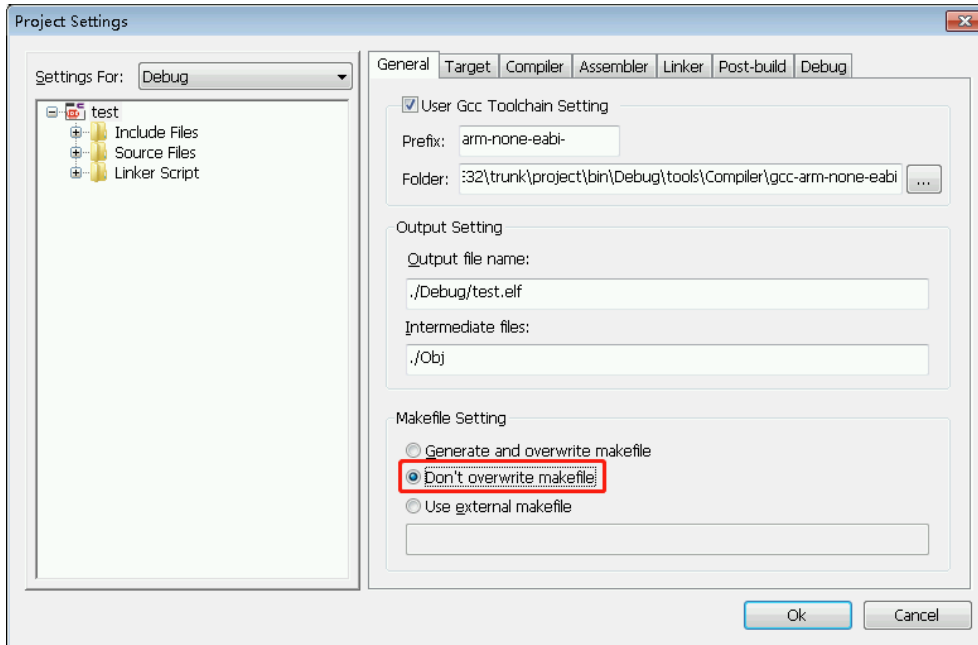


注：默认使用 arm-none-eabi-gcc v9.3.1，用户可以下载最新的 arm gcc，Folder 指定为根目录，Prefix 是 gcc 的前缀，默认名称为 arm-none-eabi-gcc.exe。

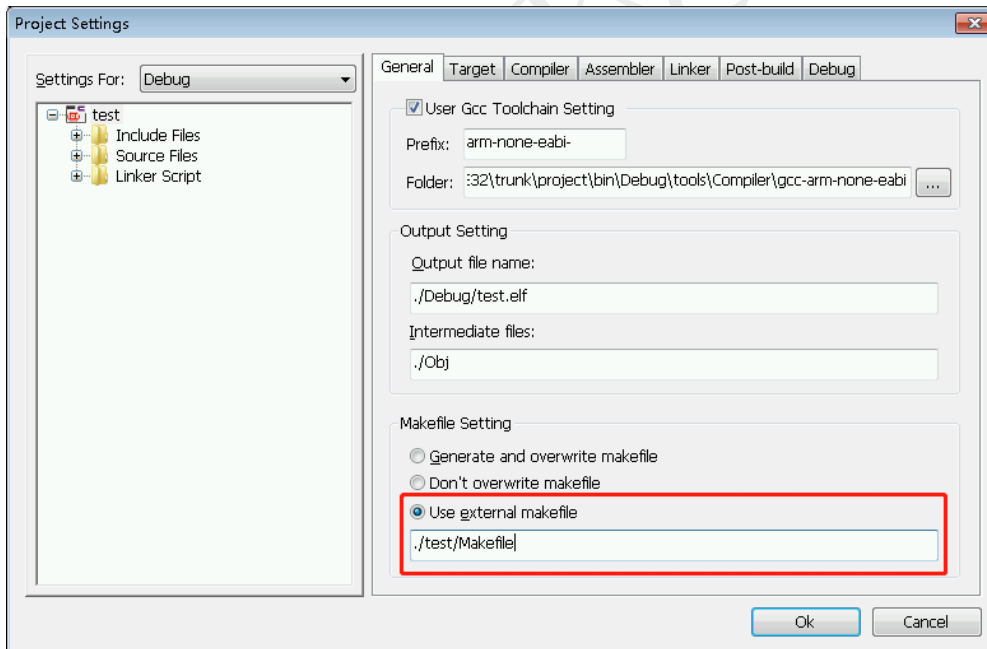
15.3 使用自定义的 Makefile

每次编译前，IDE 自动根据文件列表中的文件更新 Makefile，有时候用户想改变 makefile 的行为，有以下二种方法：

1、勾选【Don't overwrite makefile】，使用默认目录的 Makefile 文件，不自动更新，用户可以自由修改 Makefile，再执行编译命令。

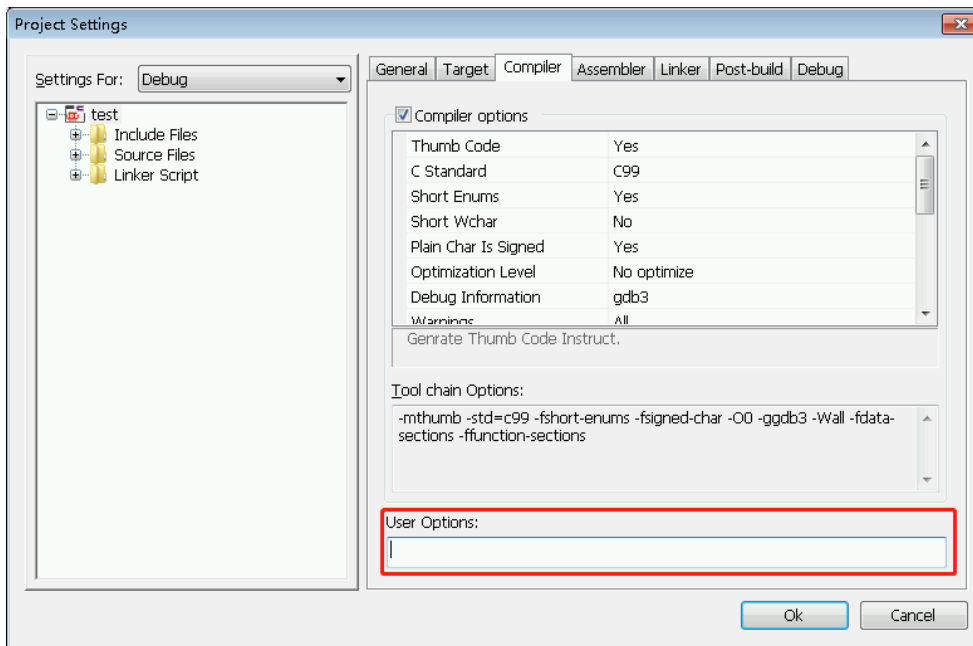


2、使用用户自定义的 Makefile 文件进行编译



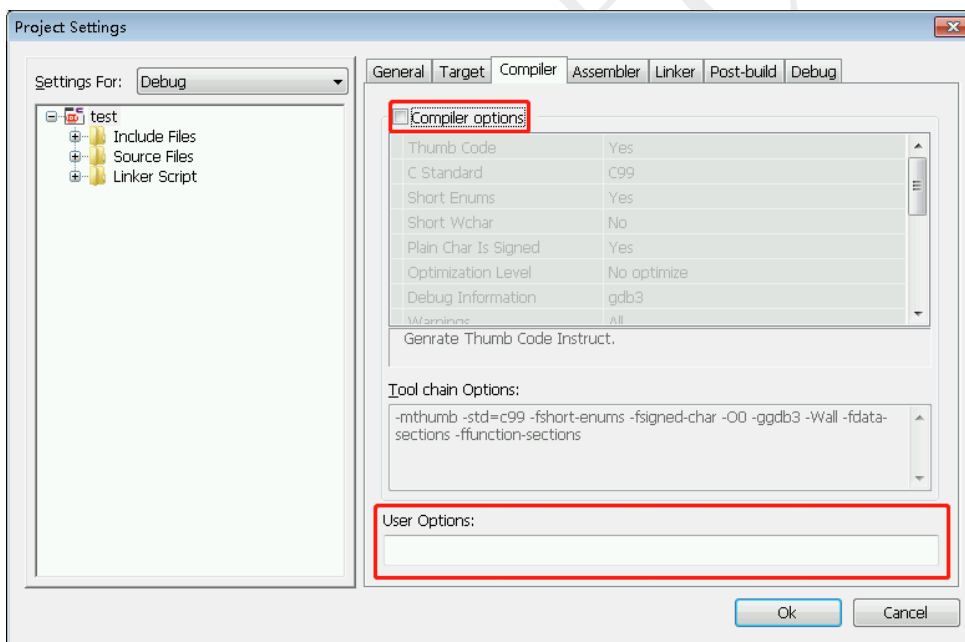
注：Makefile 必须包含几个默认命令：clean/postbuild

15.4 增加自定义的编译/汇编/链接命令



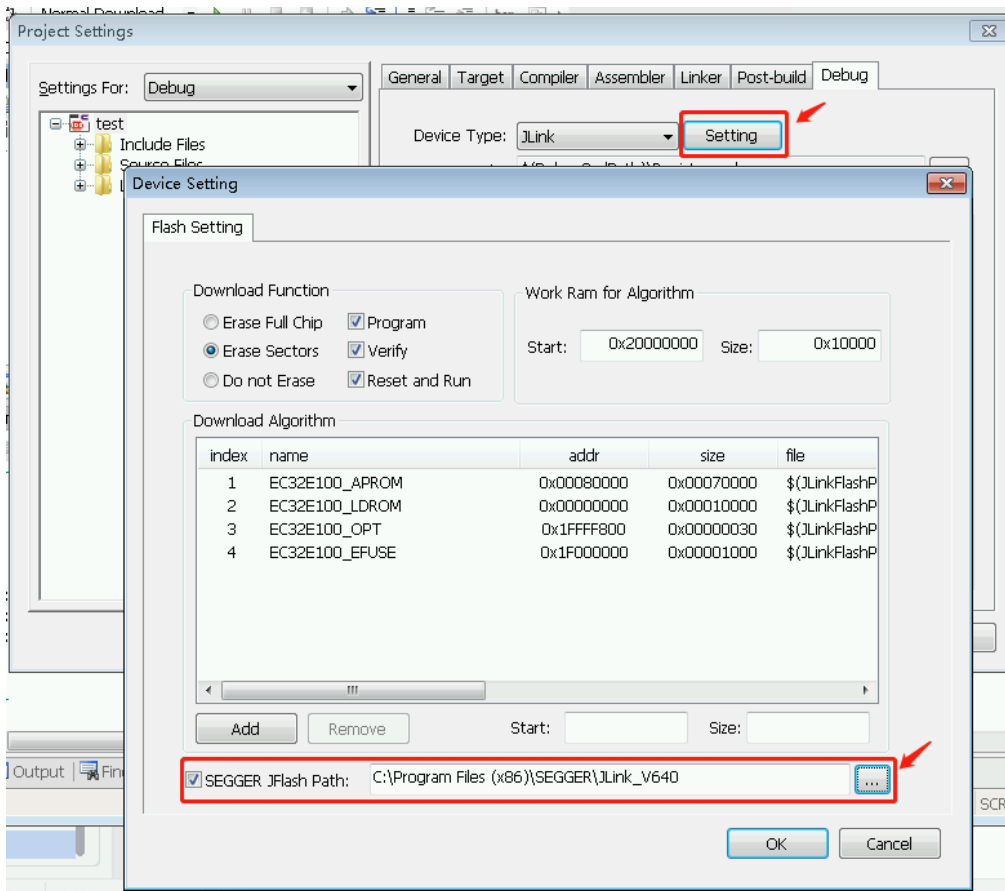
注：编译/汇编/链接配置页面都有 User Option 选项，用户可以自由输入合法的 GCC 参数。

15.5 不使用默认的编译/汇编/链接参数



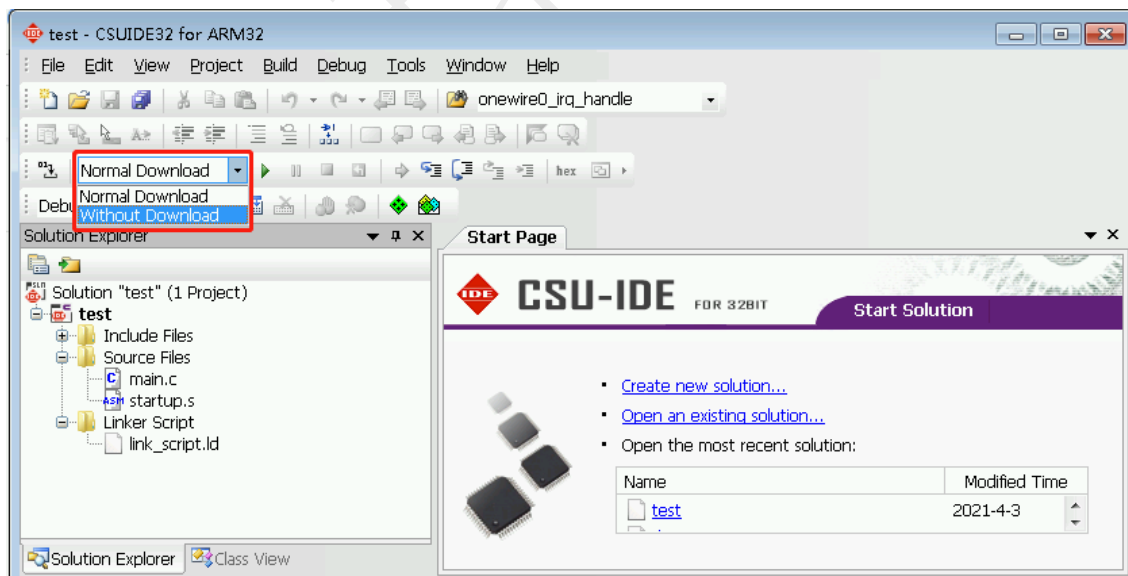
注：用户去掉页面上方的勾，编译时就不会加入上述的参数，用户可以把所有参数放在 User Option 项中，编译时只使用这些参数进行编译。

15.6 使用本地的 JFLASH 程序进行烧录调试



注：IDE 自带了烧录调试的工具，如果用户要使用本地的 JFLASH 进行烧录调试可以指定其目录。

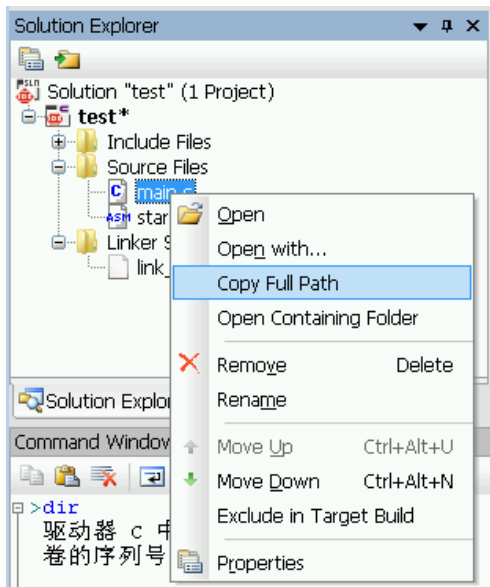
15.7 不下载程序，直接进入调试



注：用户可以选择【Without Download】项，再运行调试，则不会下载程序。

15.8 复制文件列表上的文件路径

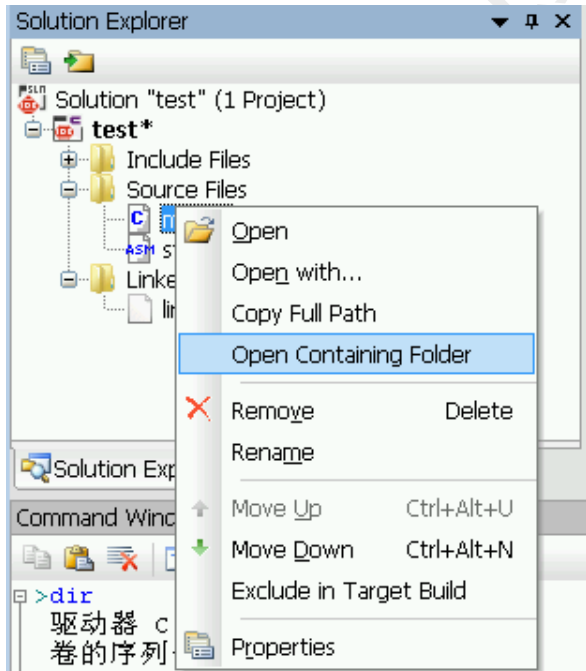
右击文件项，点击【Copy Full Path】菜单



注：【Copy Full Path】自动把文件路径复制到剪贴板

15.9 打开文件列表上的文件路径

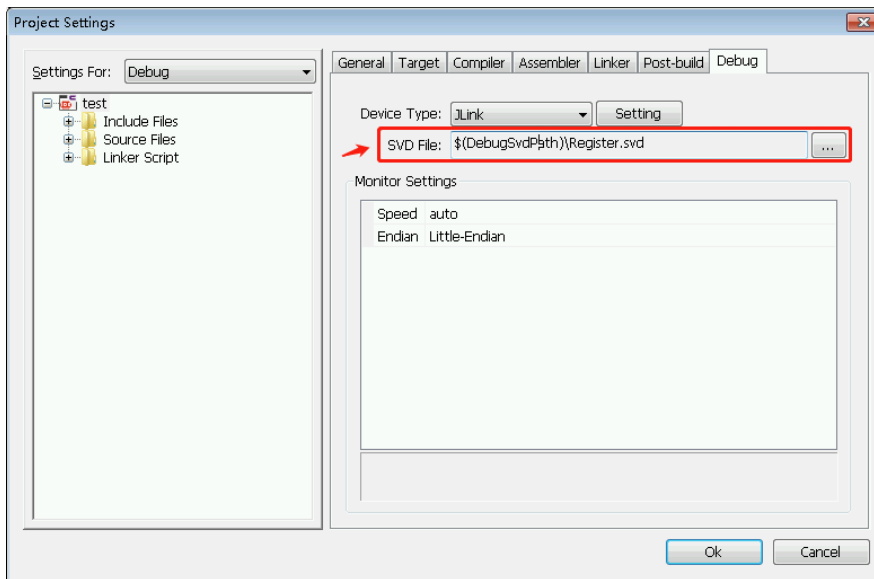
右击文件项，点击【Open Containing Folder】菜单



注：【Open Containing Folder】自动使用资源管理器打开文件目录

15.10 使用自定义的 SVD 文件进行调试

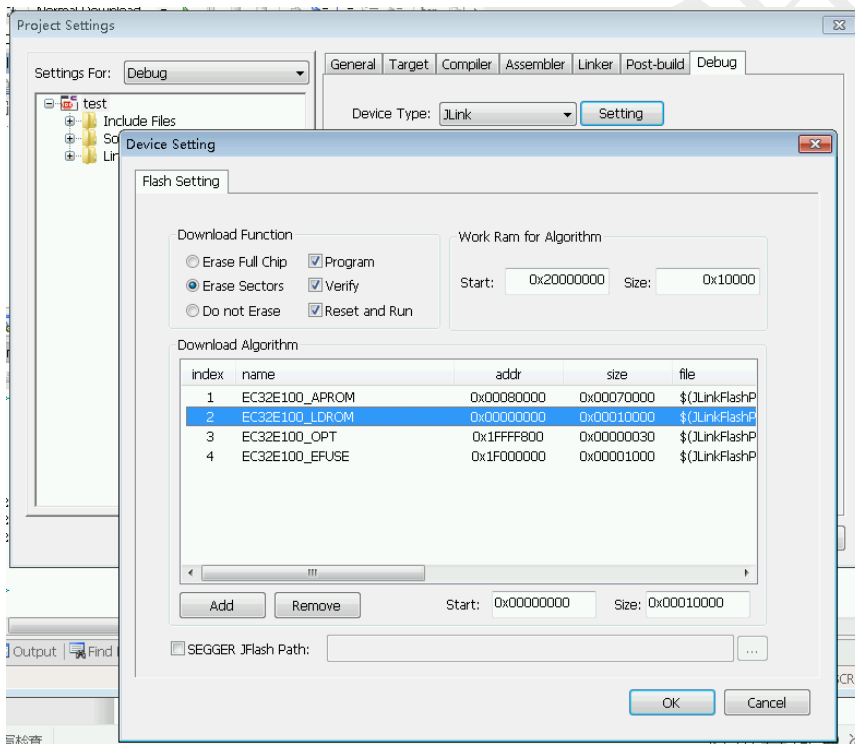
在工程设置窗口的【Debug】页面中，直接选择 svd 后缀的文件。



注：进入调试后，打开 Register 窗口，显示的窗口寄存器信息就会更新。

15.11 修改 JLINK 烧录算法配置

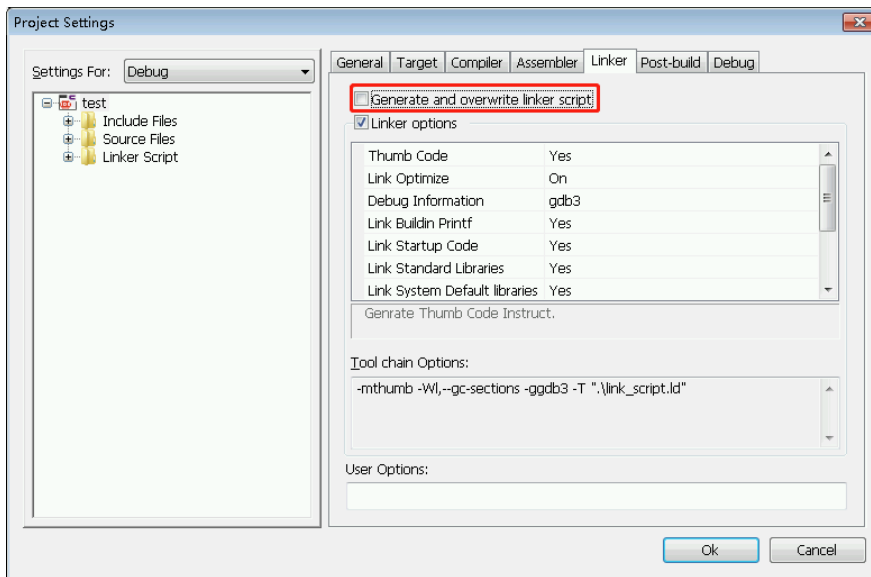
在工程配置窗口的【Debug】页面中，选中 JLink 后，点击【Setting】按钮。



注：可以增加/删除/修改烧录算法配置

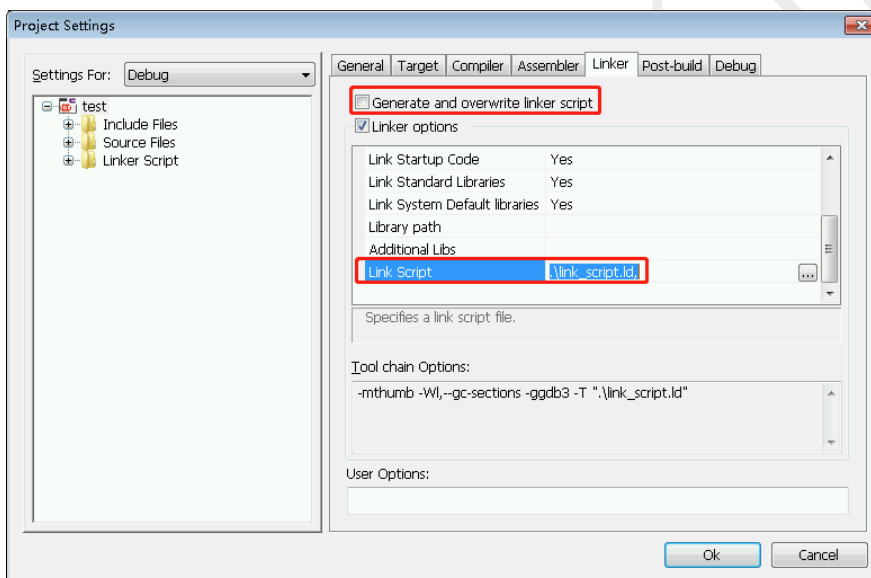
15.12 使用自定义的.Id 链接脚本

1、去掉勾选项，禁止 IDE 刷新 Link 脚本，用户自由修改目录中的 link_script.Id 文件



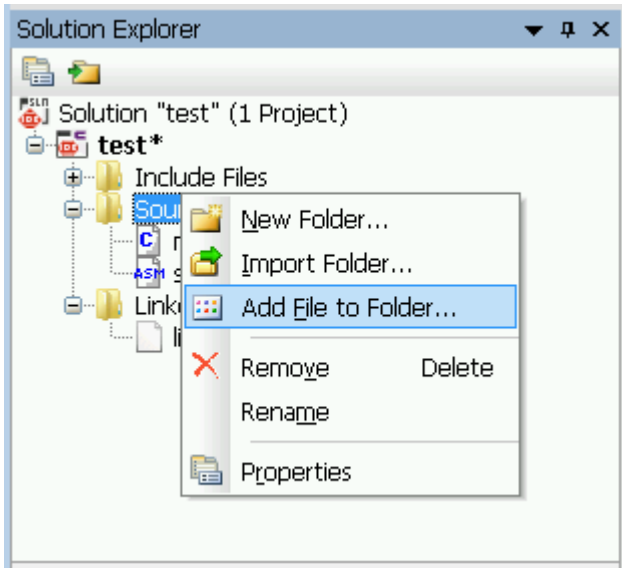
2、设置自定义的脚本文件

去掉勾选项，并修改 LinkScript 文件

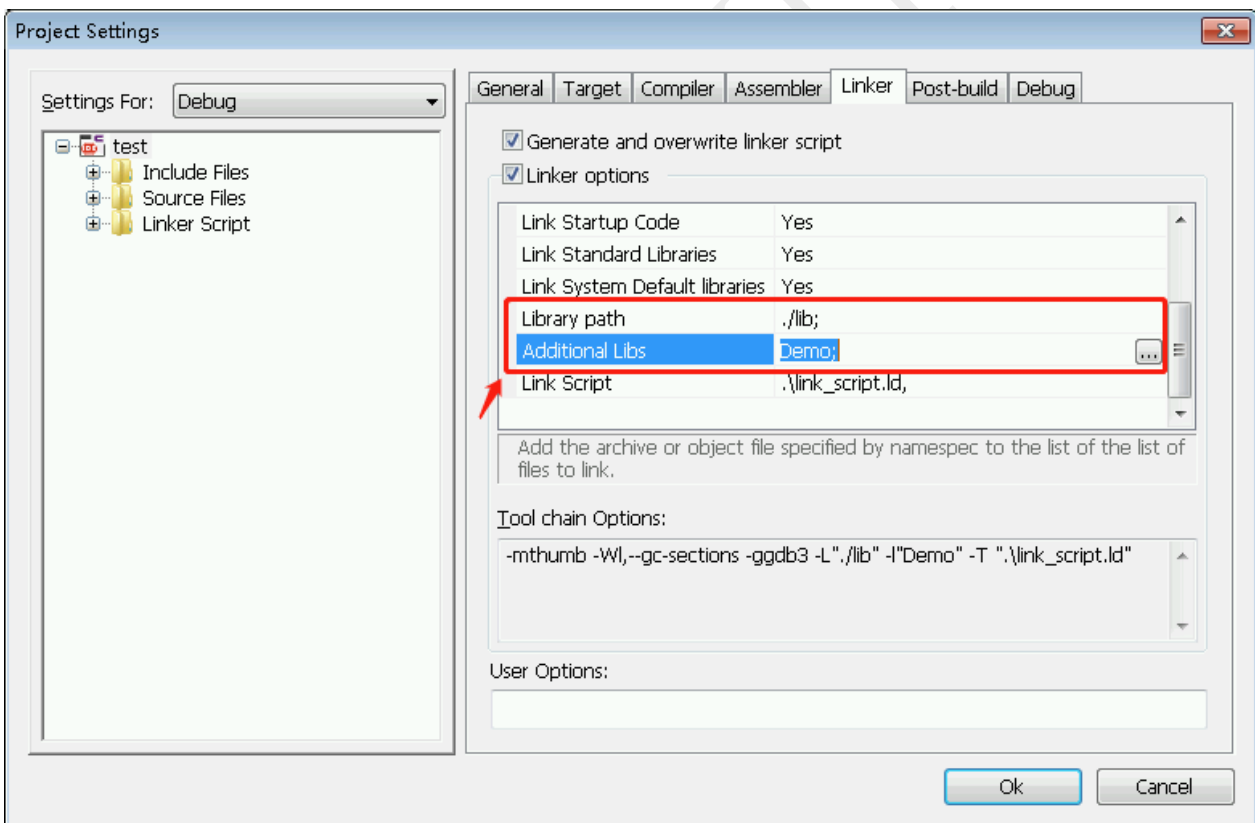


5.13 使用自定义的.a 库文件

1、将.a 文件直接添加到工程文件列表中，编译时会自动链接这个库文件

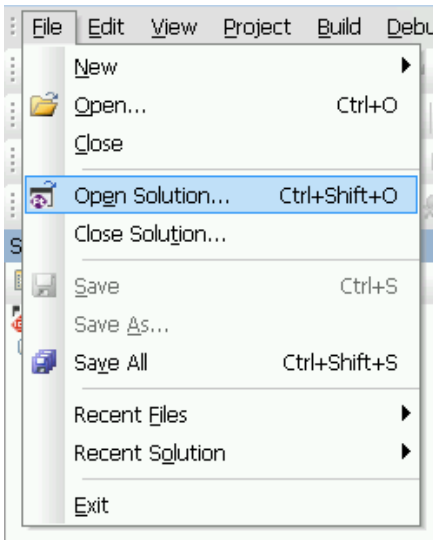


2、链接配置页面增加包含库路径和库文件

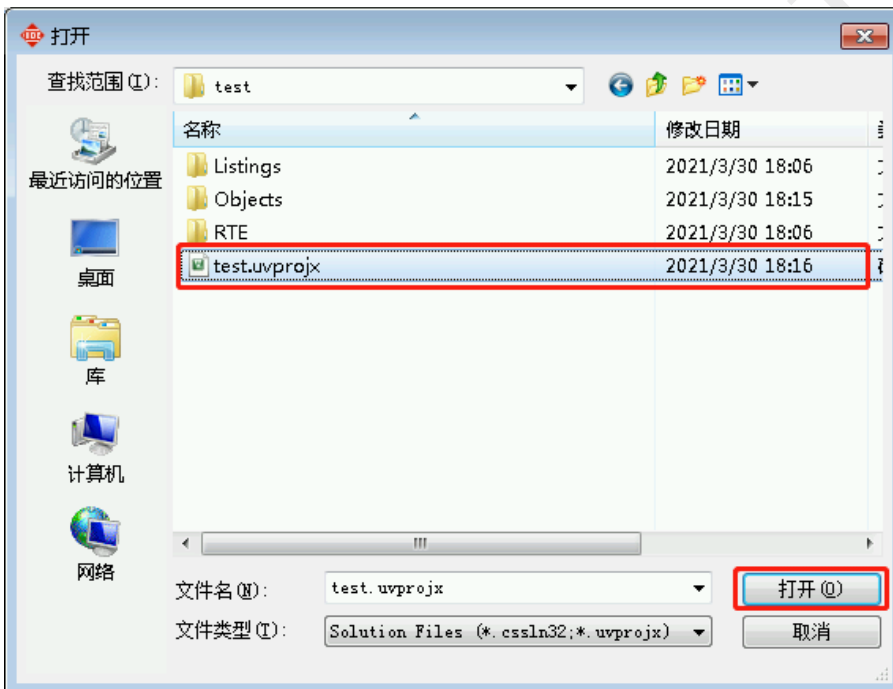


15.14 Keil 工程移植到 CSU-IDE32

1、点击【File】⇒【Open Solution】菜单



2、选择.uvprojx 文件，点击【打开】按钮后自动转换 Keil 工程



3、转换完成后，自动打开工程，由于 Keil 和 GCC 的汇编语法不同，需要更新启动代码。

免责声明和版权公告

本档中的信息，包括供参考的 URL 地址，如有变更，恕不另行通知。

本档可能引用了第三方的信息，所有引用的信息均为“按现状”提供，芯海科技不对信息的准确性、真实性做任何保证。

芯海科技不对本档的内容做任何保证，包括内容的适销性、是否适用于特定用途，也不提供任何其他芯海科技提案、规格书或样品在他处提到的任何保证。

芯海科技不对本档是否侵犯第三方权利做任何保证，也不对使用本档内信息导致的任何侵犯知识产权的行为负责。本档在此未以禁止反言或其他方式授予任何知识产权许可，不管是明示许可还是暗示许可。

Wi-Fi 联盟成员标志归 Wi-Fi 联盟所有。蓝牙标志是 Bluetooth SIG 的注册商标。

文档中提到的所有商标名称、商标和注册商标均属其各自所有者的财产，特此声明。

版权归 © 2022 芯海科技（深圳）股份有限公司。保留所有权利。



芯海科技
CHIPSEA

股票代码:688595