旋智有限公司
Spintrol Limited Co.

SPINTROL 软件调试

使用指南

V01-20200311

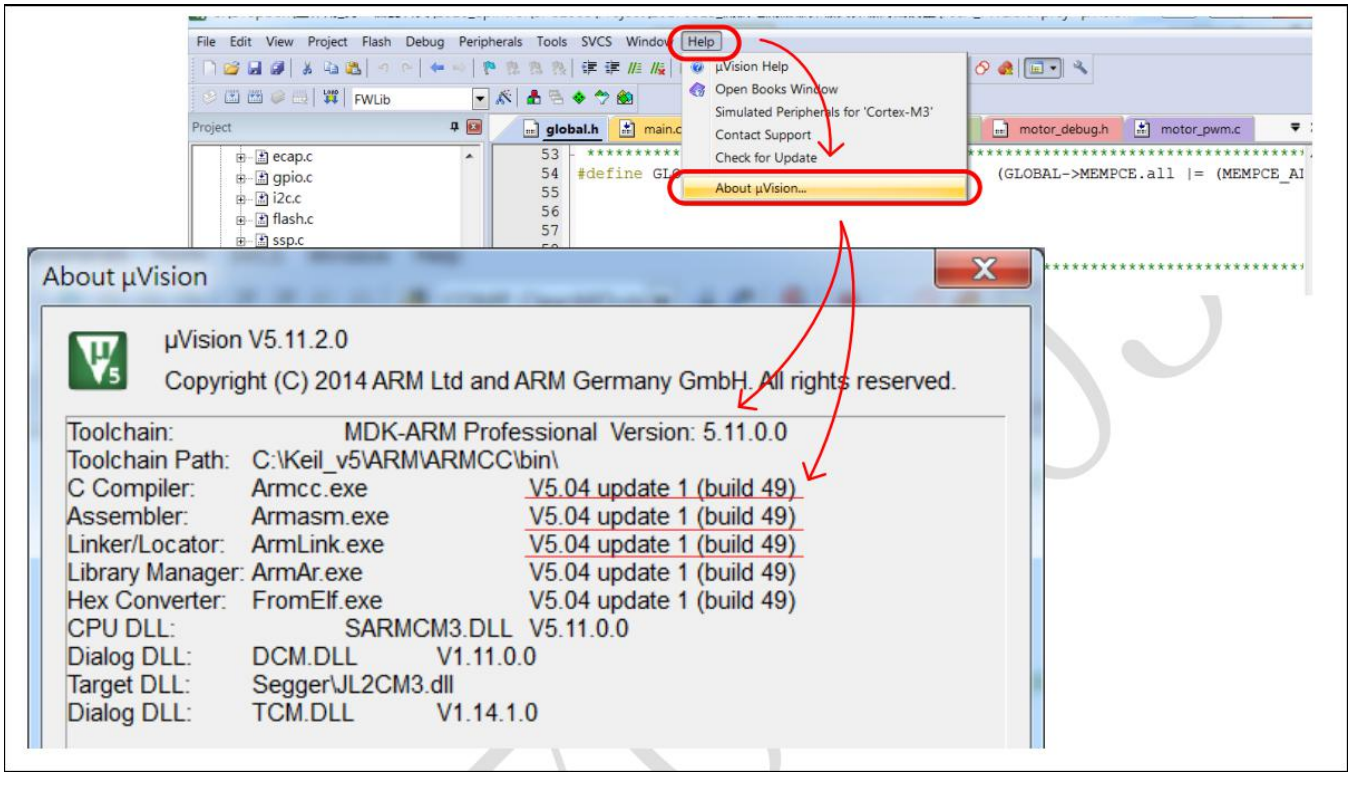
上海市浦东新区龙东大道 3000 号张江集电港 4 号楼 601 室
Room 601, Building 4, 3000 Long dong Avenue,
Pudong District, Shanghai
Tel: 86-21-58212991
E-mail: info@spintrol.com

目录

1. Keil 工具配置导引

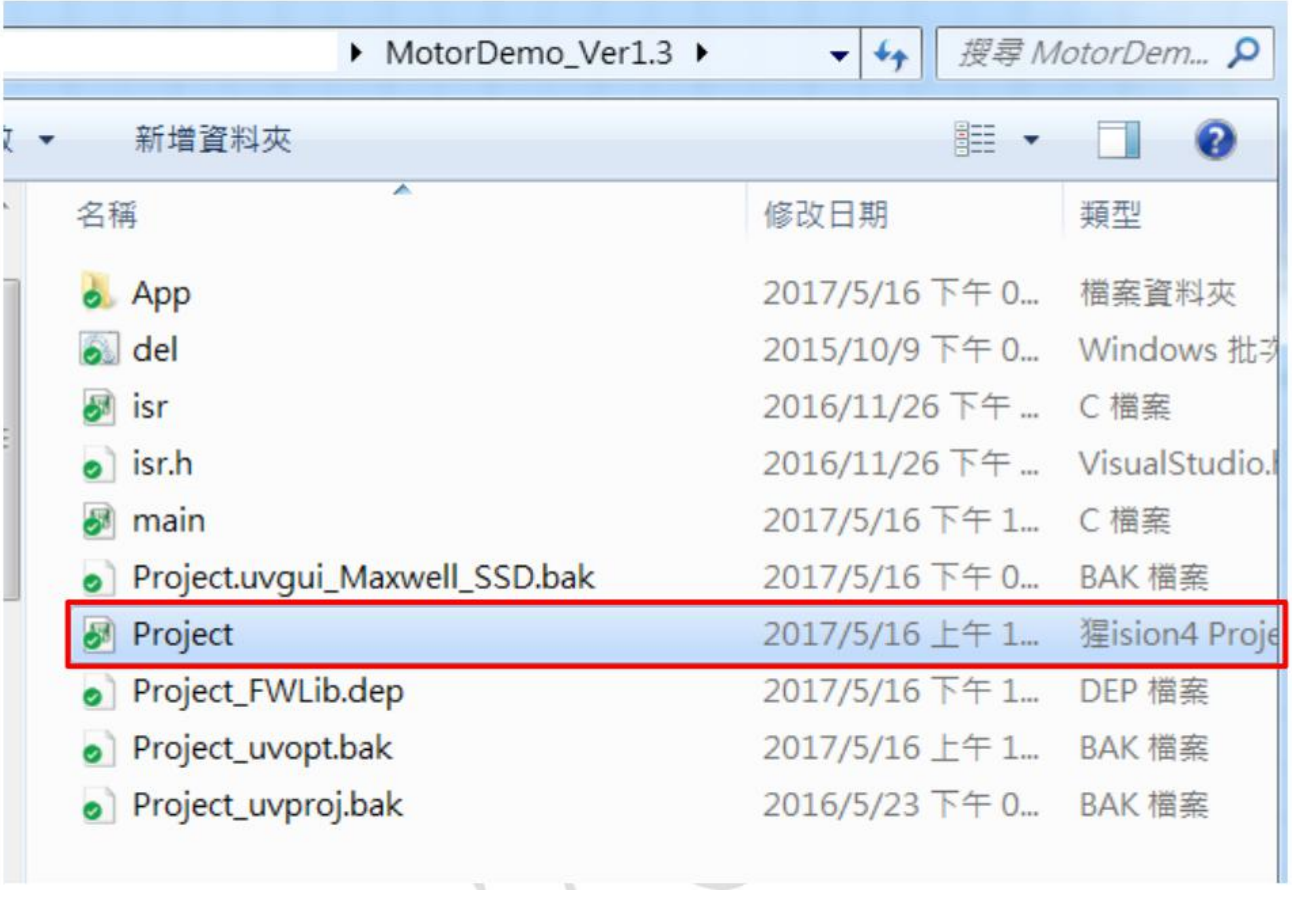
1.1. 版本检测

建议安装 Keil 5 之后的版本



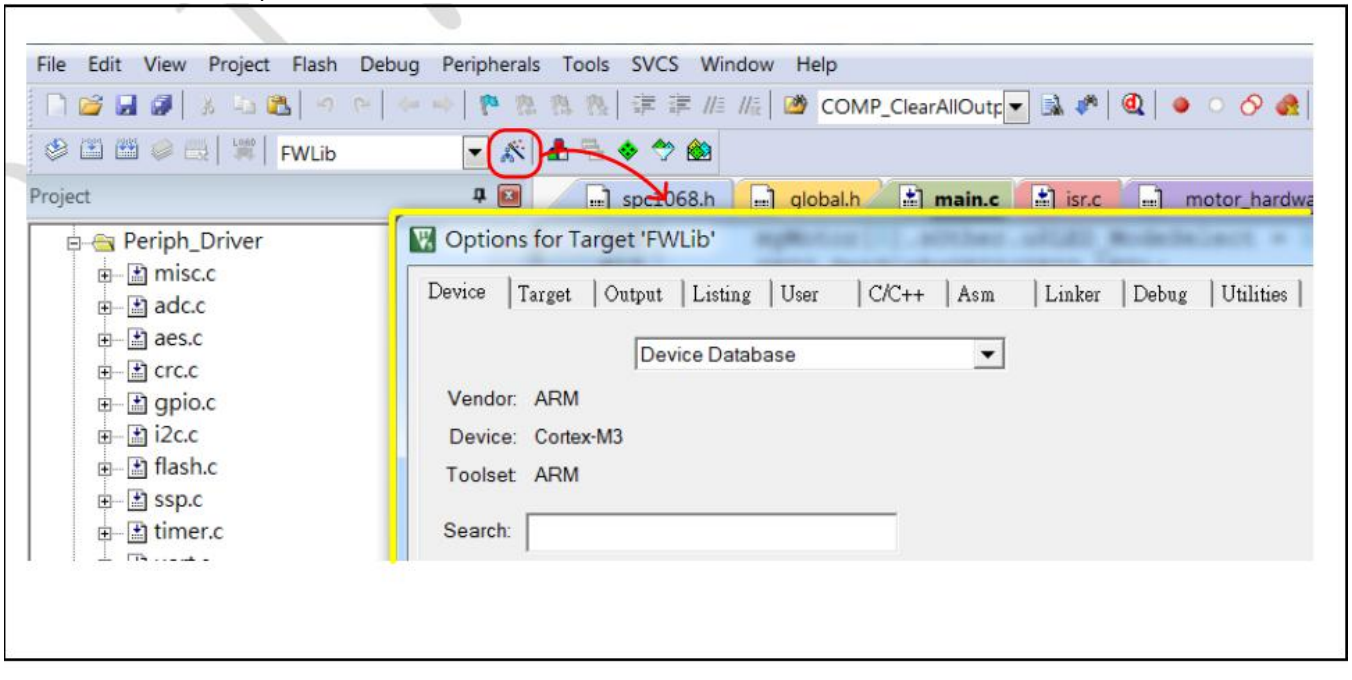
1.2. 打开项目文件

点击*.proj 格式的文件，就可以打开项目工程



1.3. 打开 Optional 配置

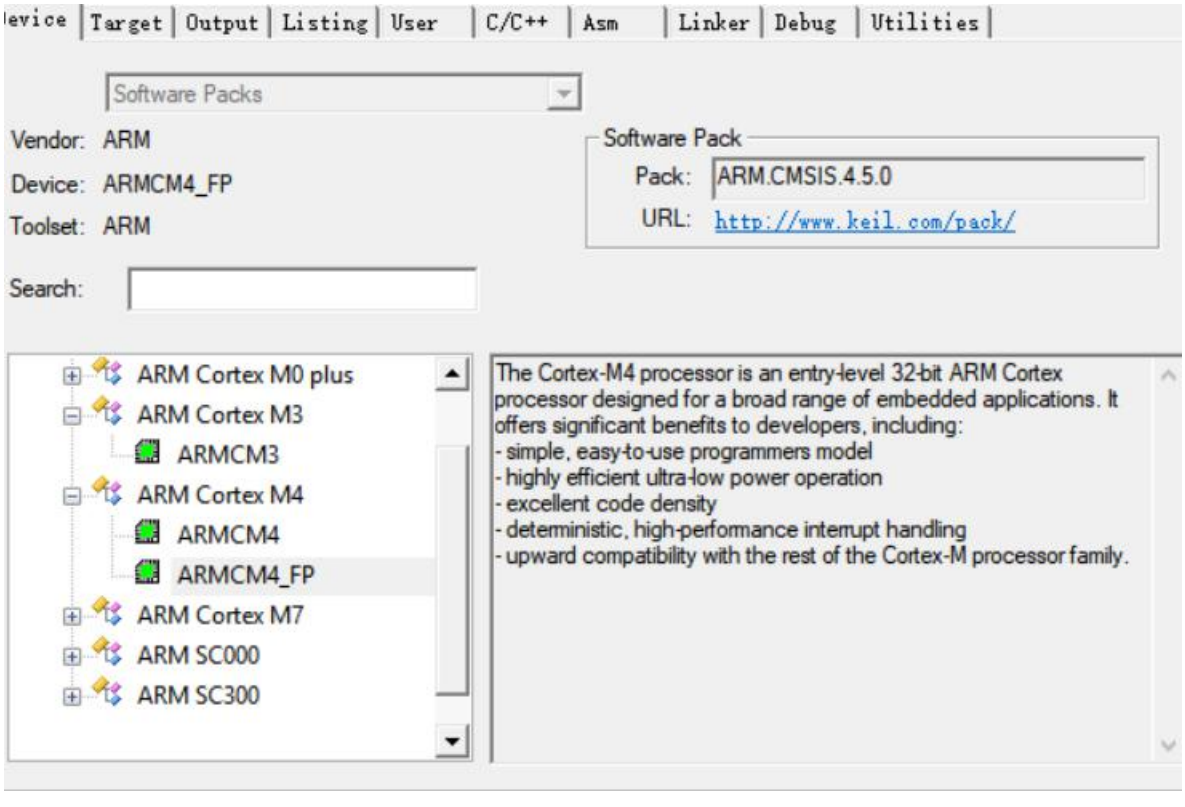
请点开魔法棒，开启 Optional 设定窗口



1.4. Optional 配置 Device 选项

请选择相应的芯片对应的 Device.

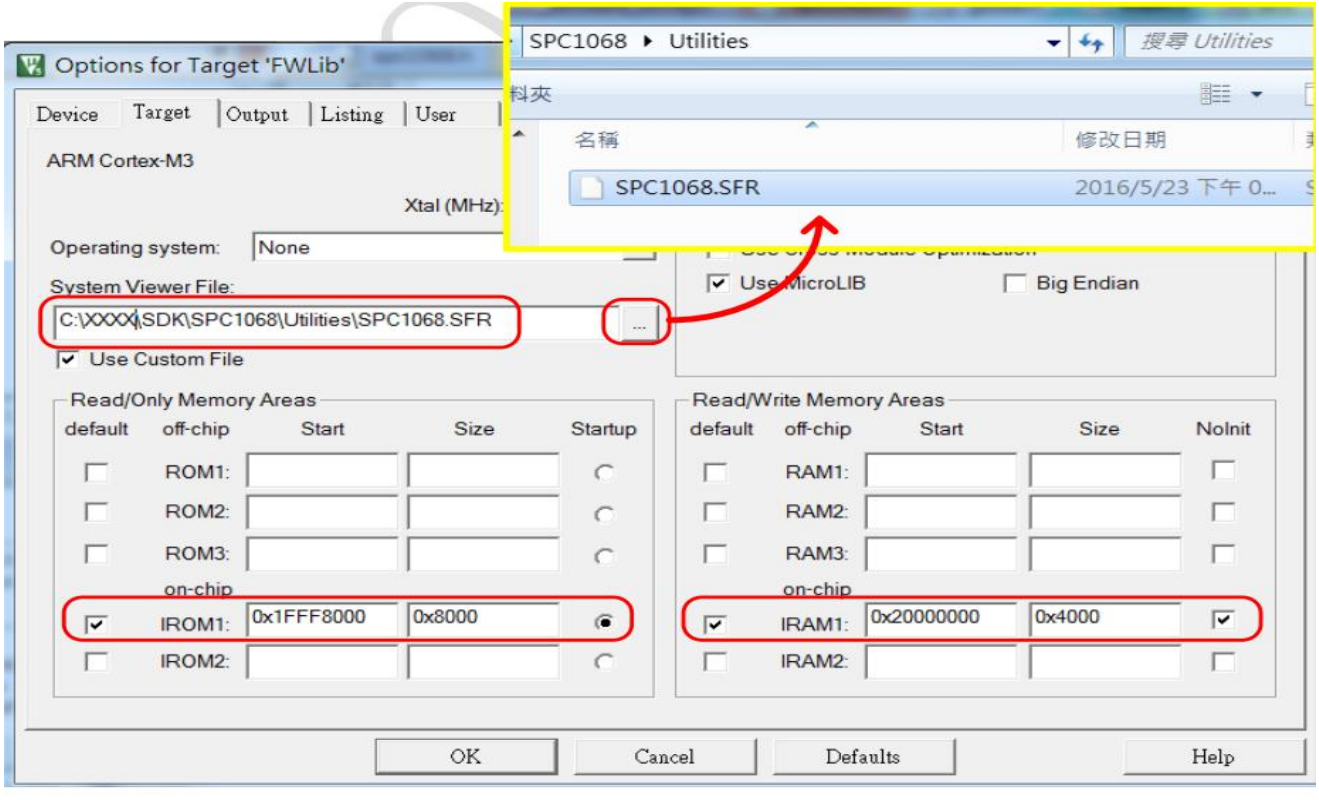
chip	Device
SPC1068	Cortex-M3 下的 ARMCM3
SPD1078	Cortex-M3 下的 ARMCM3
SPC1158	Cortex-M4 下的 ARMCM4_FP
SPC1168	Cortex-M4 下的 ARMCM4_FP
SPC1178	Cortex-M4 下的 ARMCM4_FP



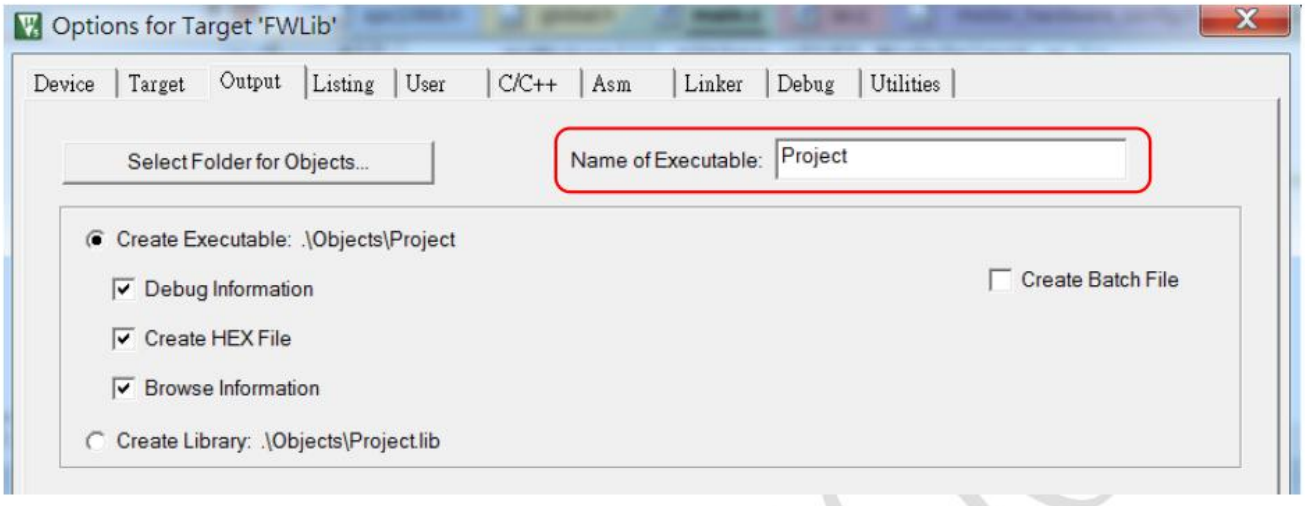
1.5. Optional 配置 Target 选项

chip	System Viewer File
SPC1068	SPC1068.SFR
SPD1078	SPC1068.SFR
SPC1158	SPC1168.SFR
SPC1168	SPC1168.SFR
SPC1178	SPC1168.SFR
注：文件一般放在此文件放置在 SDK 内的 Utilities 内，如 \SDK\SPC1068\Utilities\SPC1068.SFR	

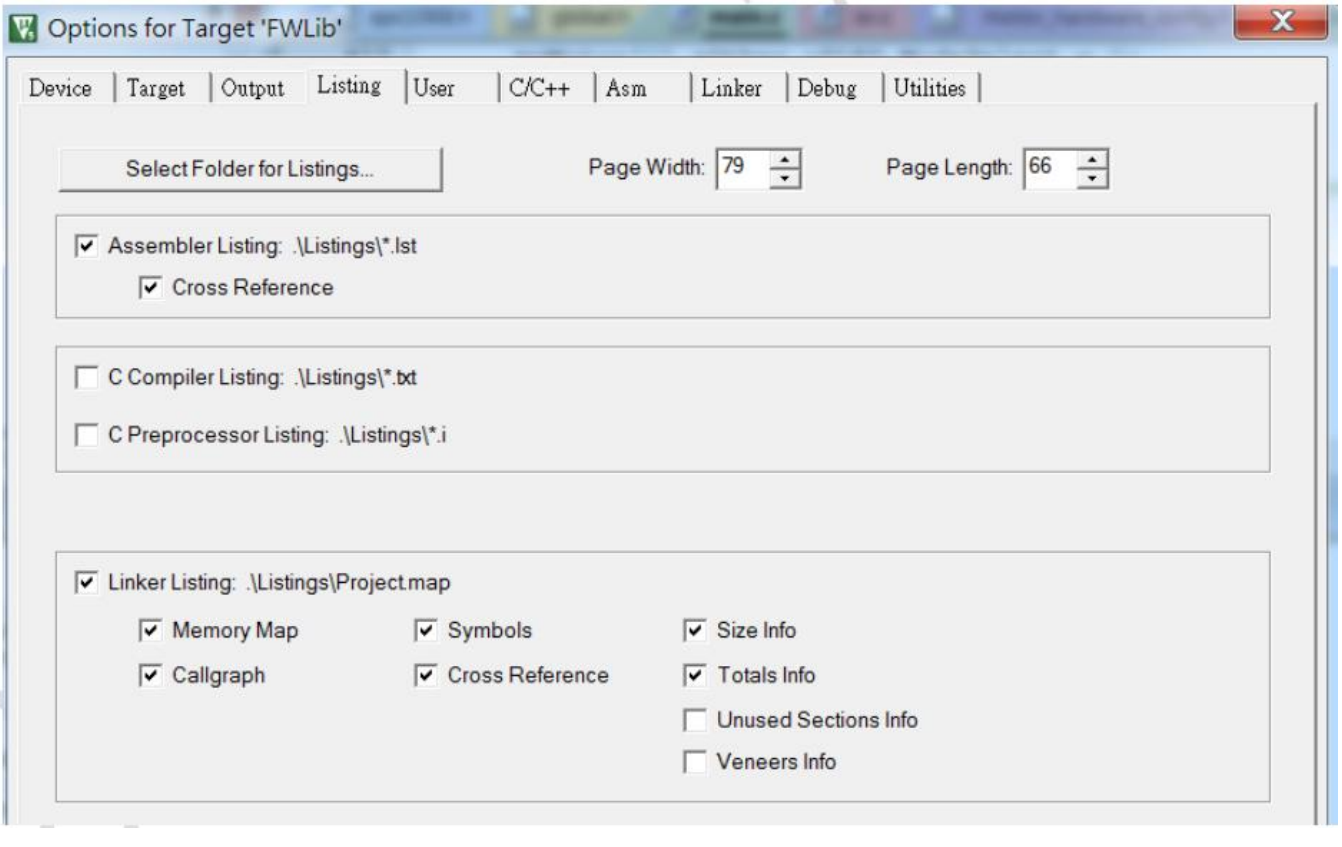
chip	IROM1	SIZE	IRAM1	SIZE	IRAM2	SIZE
SPC1068	0x1FFF8000	0x8000	0x20000000	0x4000		
SPD1078	0x1FFF8000	0x8000	0x20000000	0x4000		
SPC1158	0x10000000	0x10000	0x1FFFC000	0x8000		
SPC1168	0x10000000	0x20000	0x20000000	0x4000	0x1FFF4000	0xC000
SPC1178	0x10000000	0x20000	0x20000000	0x4000	0x1FFF4000	0xC000
详细请参考 Technical Reference Manual 里面的 Memory map						



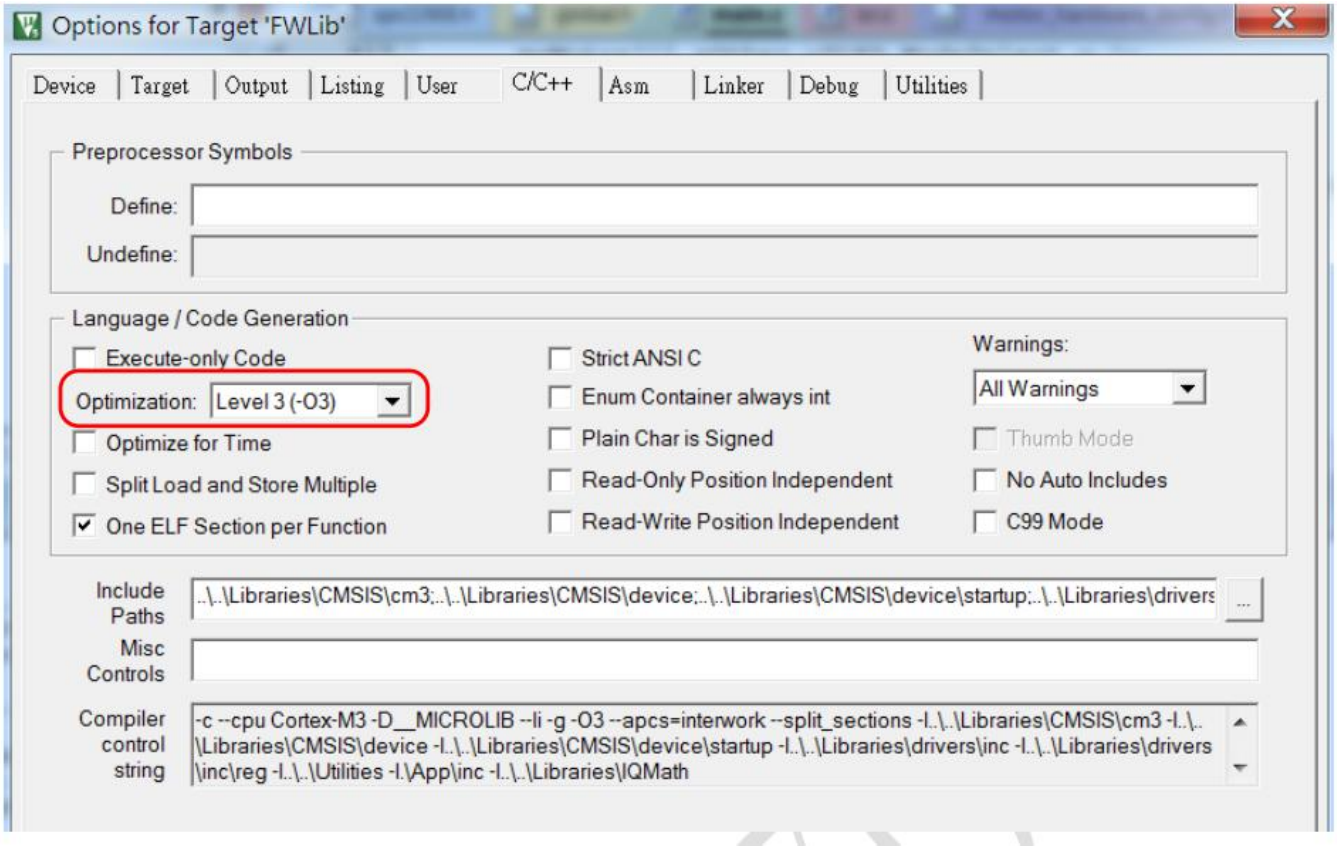
1.6. Optional 配置 Output 选项



1.7. Optional 配置 Listing 选项



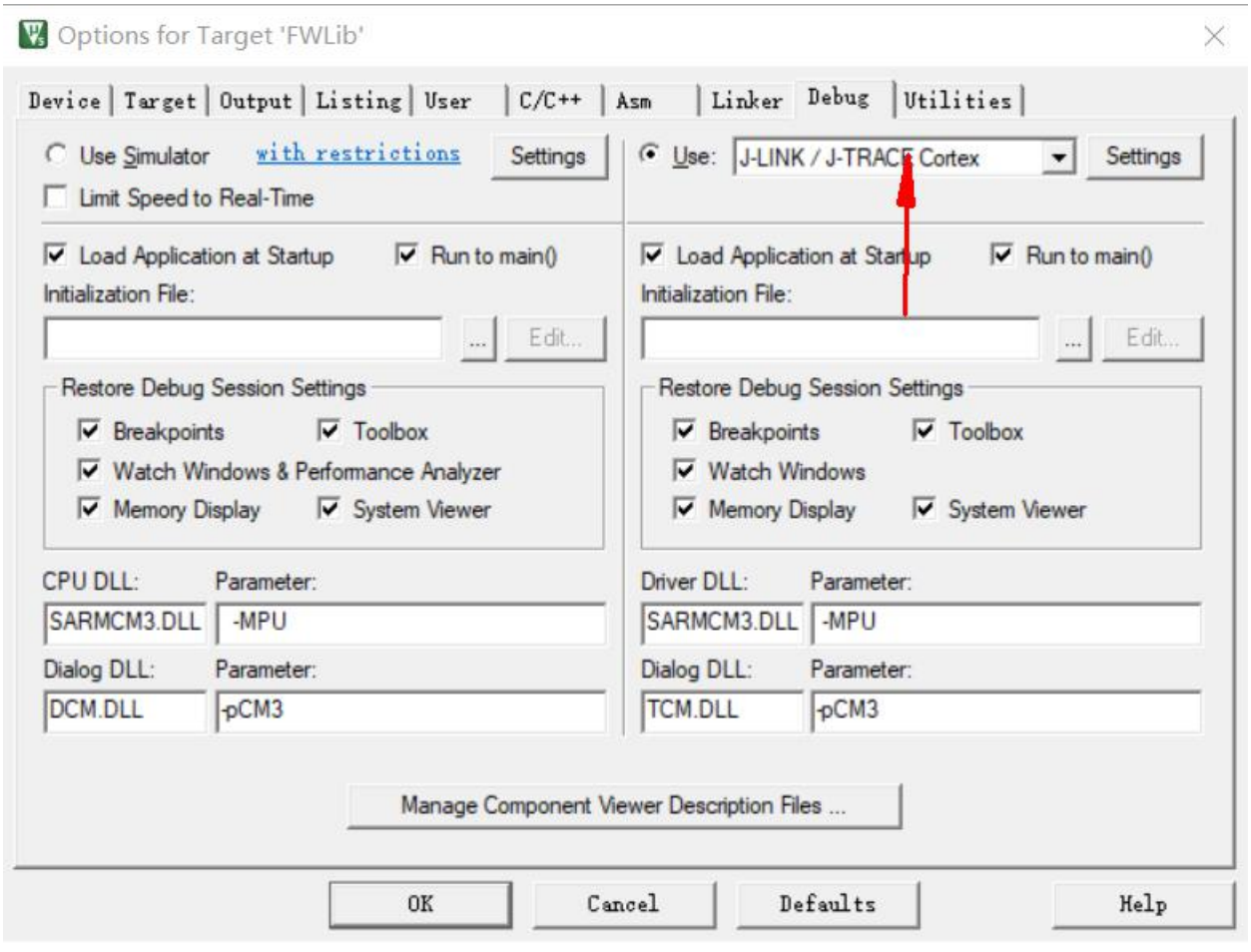
1.8. Optional 配置 Listing 选项

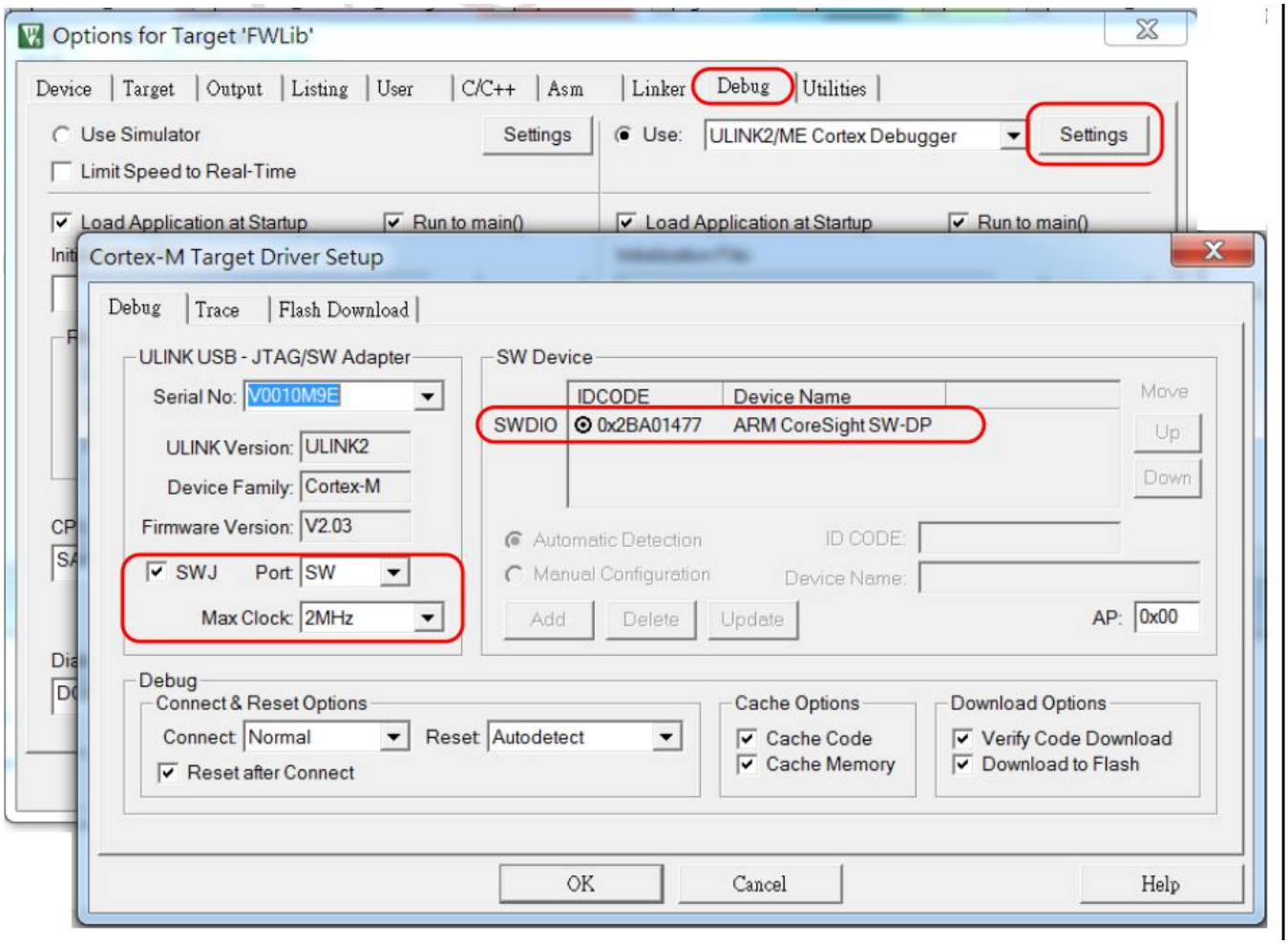


Tips:
开发初期，C 的 Optimization 建议选择 O0。使用 O2 与 O3 虽然有较佳的代码大小或是运算效率，但是在调试时变量数值不一定与预期相符合，即使运算结果正确，却容易造成开发困难。

1.9. Optional 配置 Debug 选项

选择你所需要用的 JLINK 或者 ulink 调试工具

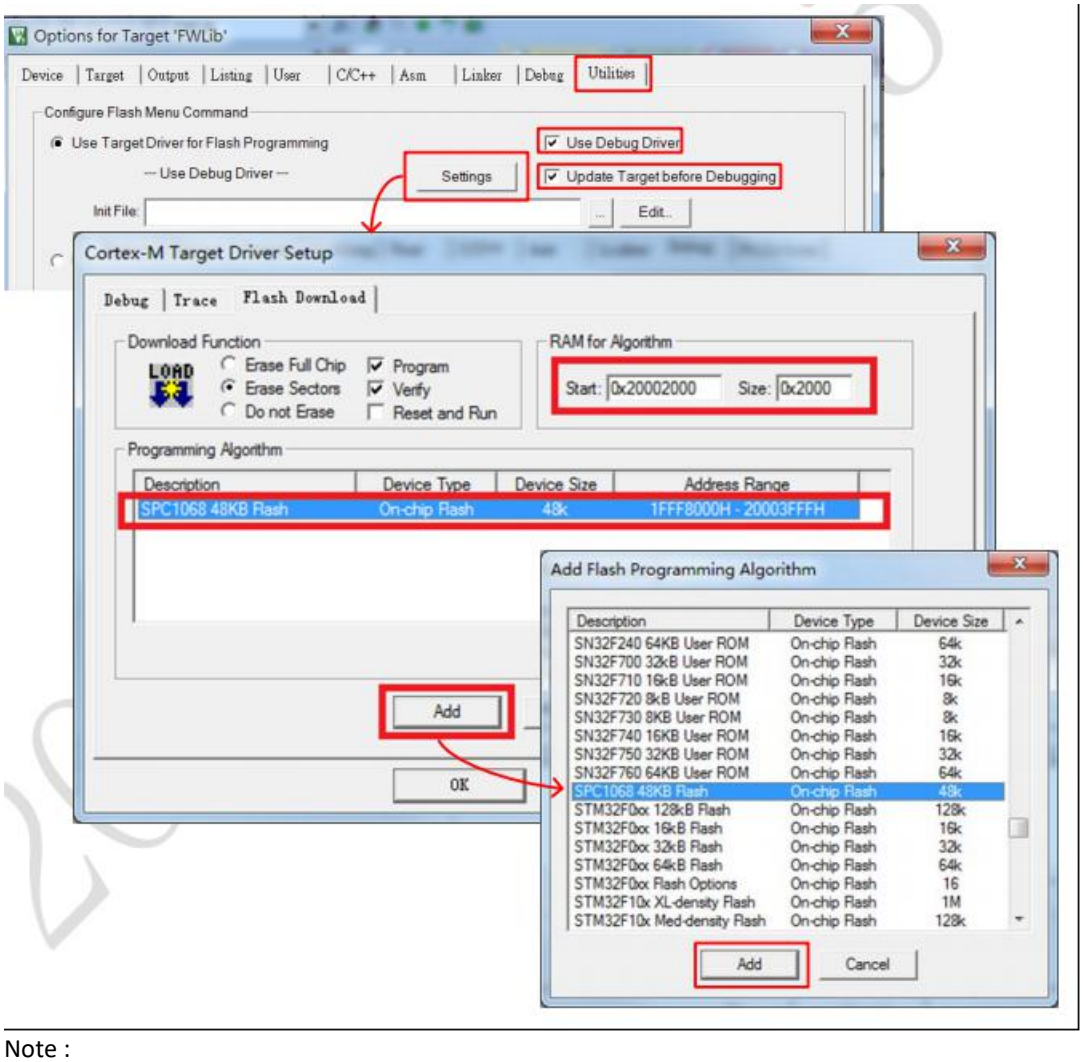

Options for Target 'FWLib'
Device
Target
Output
Listing
User
C/C++
Asm
Linker
Debug
Utilities
Use Simulator
with restrictions
Settings
Limit Speed to Real-Time
Use:
J-LINK / J-TRACE Cortex
Settings
Load Application at Startup
Run to main()
Initialization File:
Edit...
Restore Debug Session Settings
Breakpoints
Toolbox
Watch Windows & Performance Analyzer
Watch Windows
Memory Display
System Viewer
CPU DLL:
Parameter:
SARMCM3.DLL
-MPU
Driver DLL:
Dialog DLL:
DCM.DLL
-pCM3
TCM.DLL
Manage Component Viewer Description Files ...
OK
Cancel
Defaults
Help



建议选择 SW 模式进行 Debug，请注意 SW Device 必须要有显示芯片 IDCODE 才能算是连结成功。

此步骤最常出现无法与芯片沟通之状况，若发生问题，可以检查：
(1) 芯片的 TRSTn 是否没有拉到高电位，拉到高电位后，单击板子上的 Reset 键 。
(2) JTAG 或是 SWD 走在线的滤波电容是否过大，建议可先去除滤波电容

1.10. Optional 配置 Debug 的 Flash 算法选项



Note :

若在图中没有拷贝算法程序的档案，此处不会出现 相关芯片的选项。

chip	算法文件
SPC1068	SPC1068.FLM
SPD1078	SPC1068.FLM
SPC1158	SPC1168.FLM
SPC1168	SPC1168.FLM
SPC1178	SPC1168.FLM

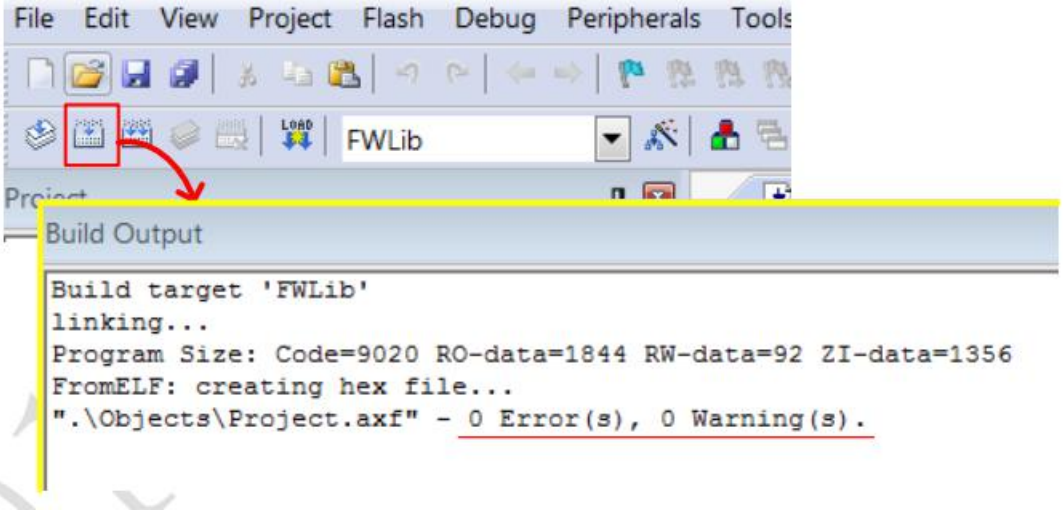
注：文件一般放在此文件放置在 SDK 内的 Utilities 内，如 \SDK\SPC1068\Utilities\SPC1068.FLM

请把相关算法文件拷贝到相应的目录下 C:\Keil_v5\ARM\Flash

1.11. 保存配置

按下 Save All，请务必记得此步骤，当 Keil 重启时才会纪录之前所有的步骤

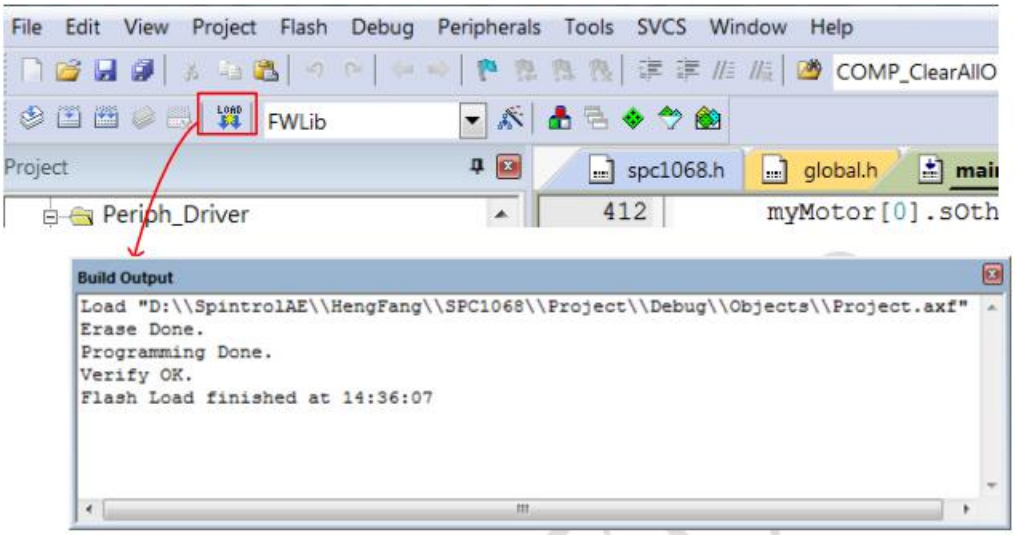
1.12. 编译结果



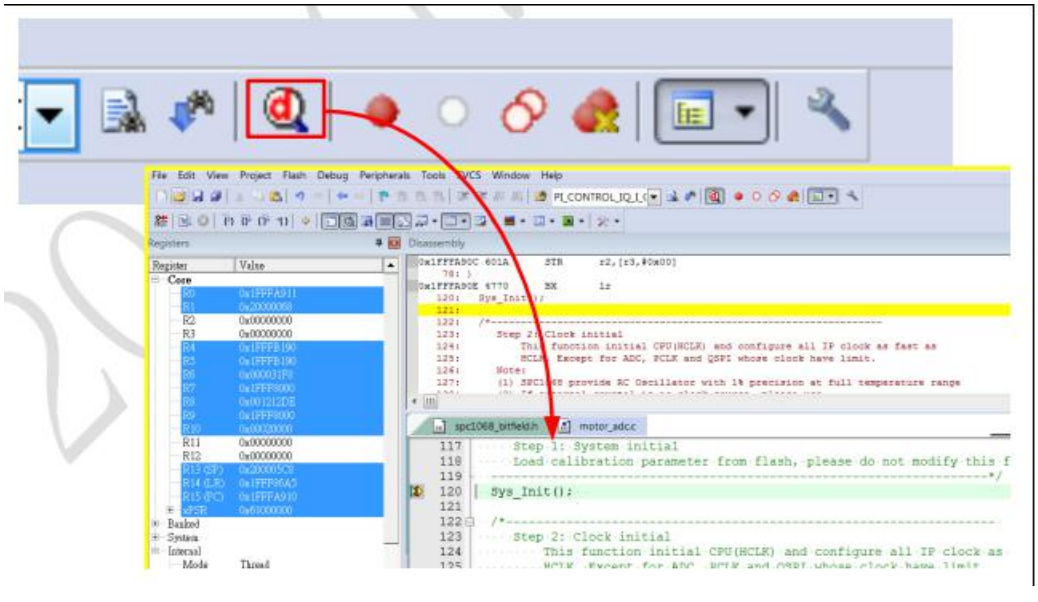
Tips:
除非项目出现严重错误需要 Rebuild all，建议按下中间的按钮即可，compiler 只会编译改动过的代码，减少等待时间

Note :
编译之后，专案的资料夹会产生编译后的 object 档案，会导致资料夹过大，不方便夹带在电邮中，请参考 Appendix **錯誤! 找不到參照來源。**的方法进行解决。

1.13. 下载程序



1.14. 调试程序



1.15. 仿真注意事项

Jlink 在线仿真时，遇到无法正确复位的问题。此处是因为在点复位后需要在 4s 内点运行程序才能正常的在线仿真，否则会出错！

2. JLINK/ULINK2 工具使用指南

2.1. 调试工具与开发板 JTAG 连接

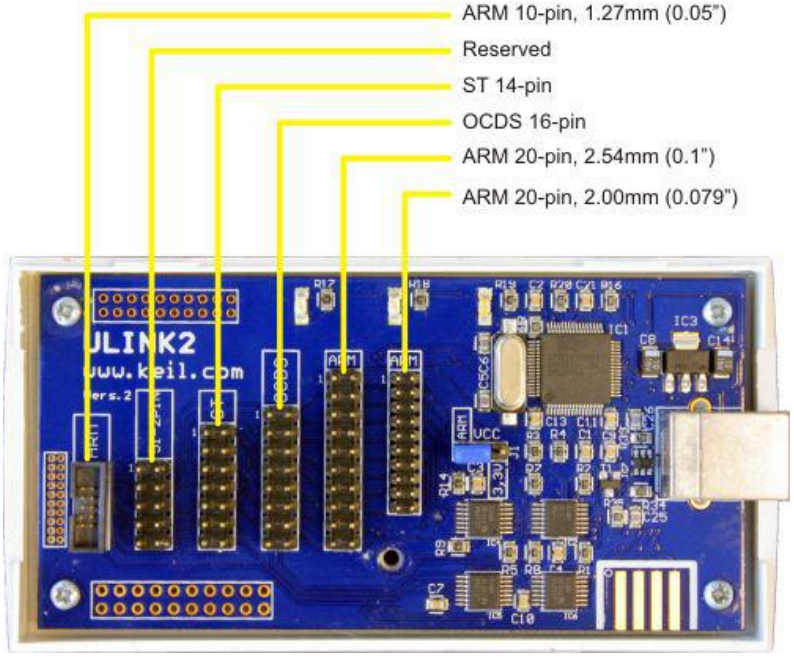


图 2- 1 : ULINK2 适配器接口

图 2- 1 : JLINK OB 调试器

图 2- 1 ： JLINK OB 调试器为推荐的调试工具，淘宝搜索一下 jlink ob 就可以看到。 ulink 适配器支持 5 种 JTAG 接口，如图 2-1 所示。其中，ARM 20-pin，2.54mm 接口是用于 ARM 芯片调试的标准 JTAG 接口。该接口信号定义如图 1-2 所示。ulink 或者 jlink 只需要选择一种就可以了。

Signal	Connects to...
TMS	Test Mode State pin — Use 100K Ohm pull-up resistor to VCC
TDO	Test Data Out pin
RTCK	JTAG Return Test Clock
TDI	Test Data In pin — Use 100K Ohm pull-up resistor to VCC
TRST	Test Reset/ pin — Use 100K Ohm pull-up resistor to VCC
TCLK	Test Clock pin — Use 100K Ohm pull-down resistor to GND
VCC	Positive Supply Voltage — Power supply for JTAG interface drivers
GND	Digital ground
RESET	RSTIN/ pin — Connect this pin to the (active low) reset input of the target CPU

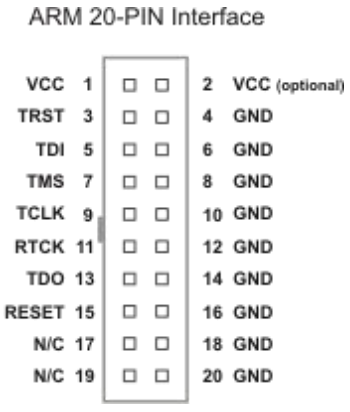
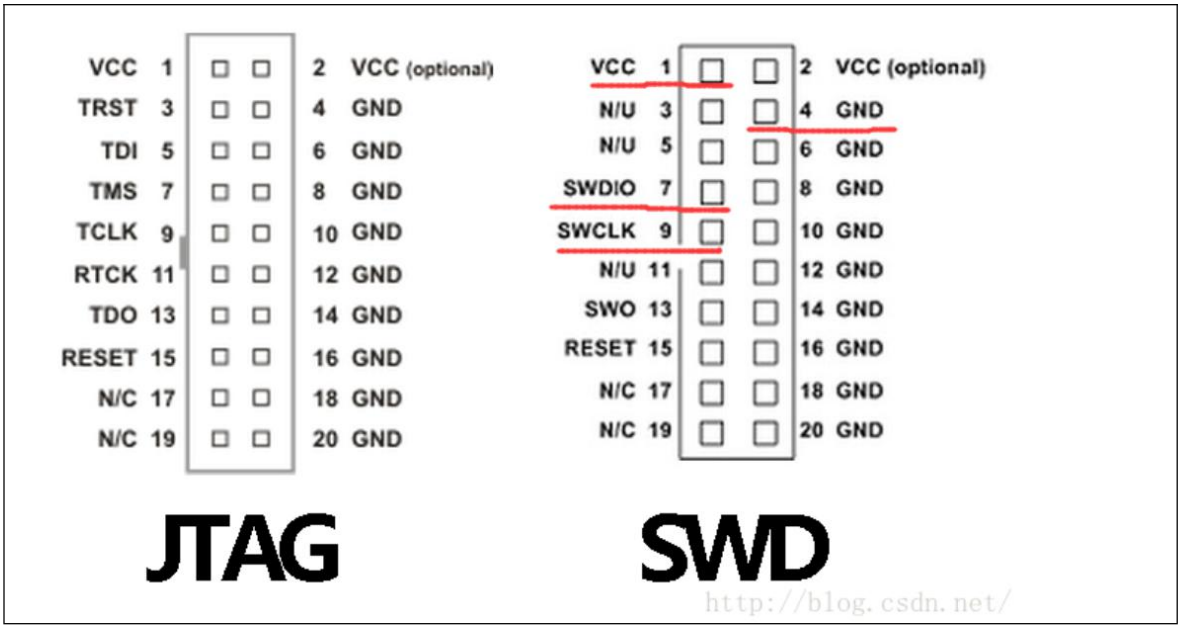


图 3.1- 2: ARM 20-PIN 接口

在采用 SPINTROL 芯片进行应用开发的过程的中，需要经常使用 JTAG 进行程序的调试。调试工具与 SPINTROL 的硬件连接如图 1-3 所示。表 3.1-1 中为具体的 PIN 脚连接关系。

表 3.1- 1: 与硬件连接图

JTAG	SWD	SPC1068	SPD1078	SPC1168 SPC1158 SPC1178 SPC1188	SPC1148	持续更新
TMS	SWDIO	GPIO16	GPIO30	GPIO38	GPIO38	
TDO	/	GPIO17	GPIO29	GPIO36	GPIO36	
RTCK	/	/	/	/	/	
TDI	/	GPIO15	GPIO28	GPIO37	GPIO37	
TRST	/	TRSTn	TRSTn	TRST	TRST	
TCLK	SWDCLK	GPIO18	GPIO31	GPIO39	GPIO39	
VCC	VCC	+3.3V	+3.3V	+3.3V	+5V	
GND	GND	GND	GND	GND	GND	
RESET	/	/	/	/	/	
注：相关管脚位置可以参考 datasheet 的 Pin-outs and pin description 章节						



2.2. 硬件管脚 JTAG/ULINK/JLNK 配置

使用 ULINK/JLINK 工具下载程序时，**首先需要设置 Boot Pin 管脚，以及 TRSTN，然后按下 RESET 按键**，芯片处于正常模式，不然将处于 ISP 模式。此时不能进行 JTAG/ULINK/JLINK 调试。

芯片	Boot PIN	TRSTn	RESET 按键
SPC1068/SPD1078	GPIO0 需要拉高	TRSTn 需要拉高	最后按下 reset 按键，进入下载模式
SPC1158/SPD1148 SPC1168/SPC1178/SPC1188	GPIO40 需要拉高	TRSTn 需要拉高	最后按下 reset 按键，进入下载模式
注： 详细可以查看芯片的 datasheet 的 Boot mode 章节，具体管脚参考 Pinout and pin description 章节			

2.3. KEIL 环境下 JLINK 配置

在安装 KEIL MDK 时，软件会默认安装 JLINK 设备的驱动。按照上面步骤将 Jlink 与硬件连接，然后给芯片上电。这时打开 KEIL 软件，鼠标左键单击图标，弹出界面如下：

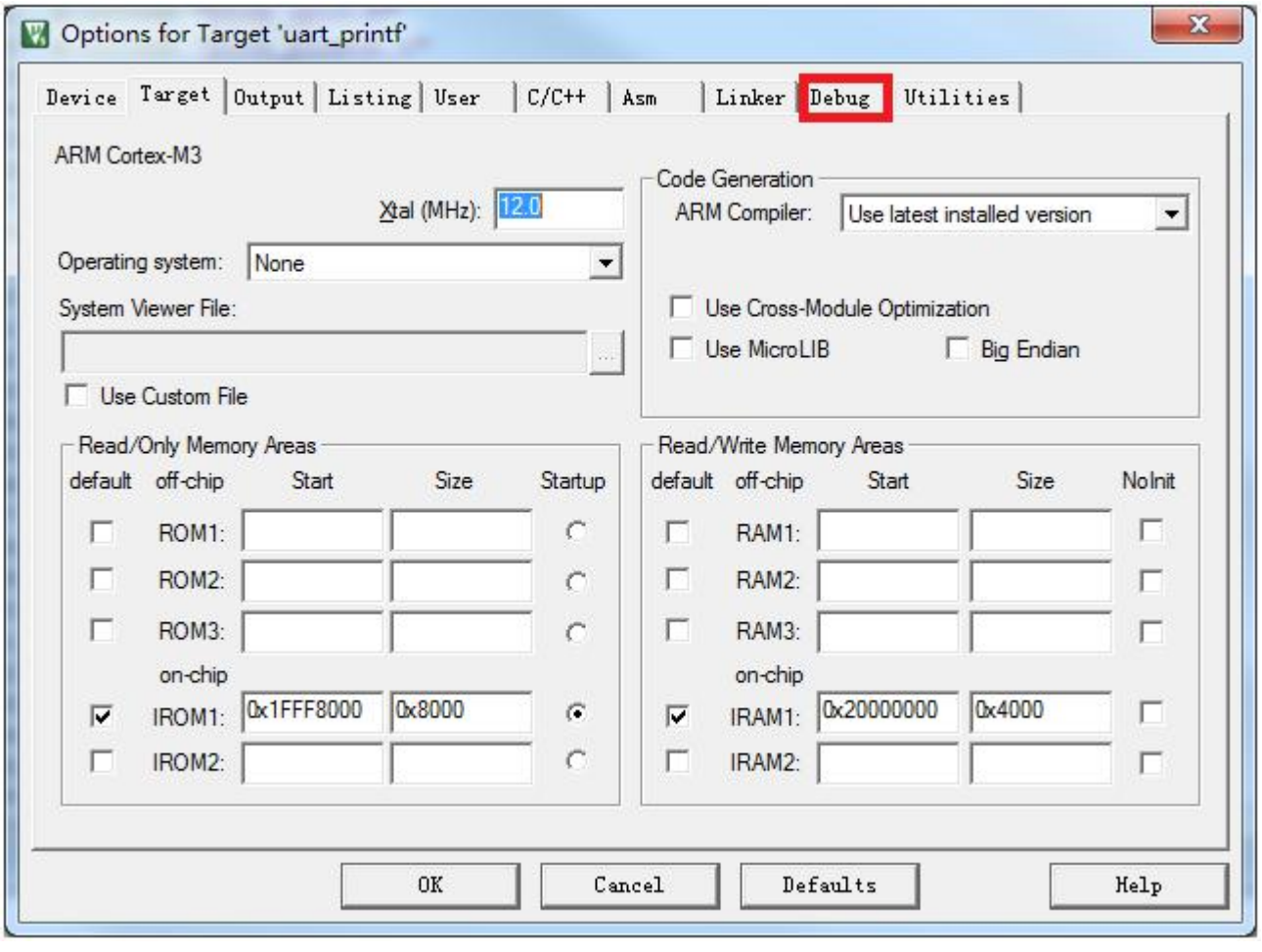


图 2- 1: Options for Target

选择 Debug 选项卡，会看到如图 2-2 所示的界面。红色矩形框标记的内容是 Debug 时需要设置的选项。

图 2-2 所示界面中，左侧是仿真调试相关的配置选项，右侧则是与硬件调试相关的选项。根据实际情形，如果是使用 ULINK2，就选择使用 ULINK2/ME Cortex Debugger 选项，如果是 JLNK 就选择 J-LINK/J-TRACE Cortes 选项。单击 Settings 按钮，会弹出相关的设置，如图 2-3 所示。可以看到，红色矩形框中出现 Debug targets 的信息，表明 ULINK2/J-LNK 设备此时是正常工作的；否则，则表明设备不可用。因此，在用 ULINK2/J-LNK 调试程序时，常常用此方法检查 ULINK2/J-LNK

设备是否正常。目前，SPINTROL 芯片的 Debug 模块只能支持 VECTRESET 功能，即在 Debug 的时候只能 Reset 芯片的内核。因此，必须按照图 2-2 配置 Connect & Reset Options。此外，SPINTROL 芯片支持 JTAG 和 SWD 两种 Debug 协议，用户可以根据需要进行配置。

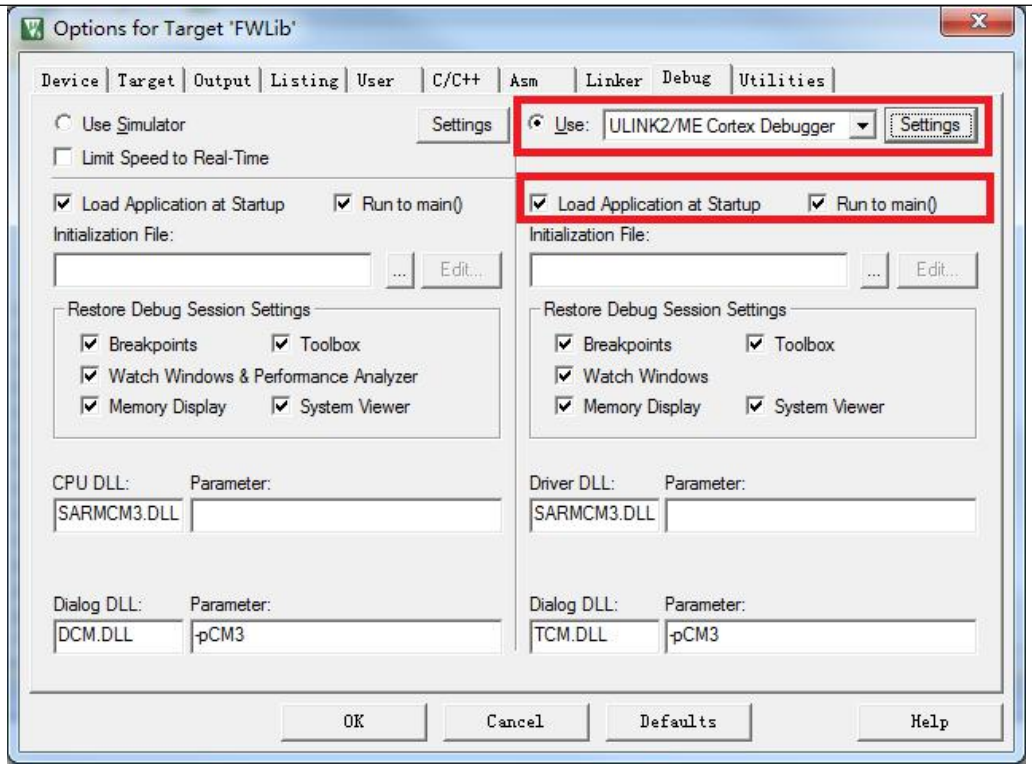


图 2.3.1: ULINK Debug 配置界面

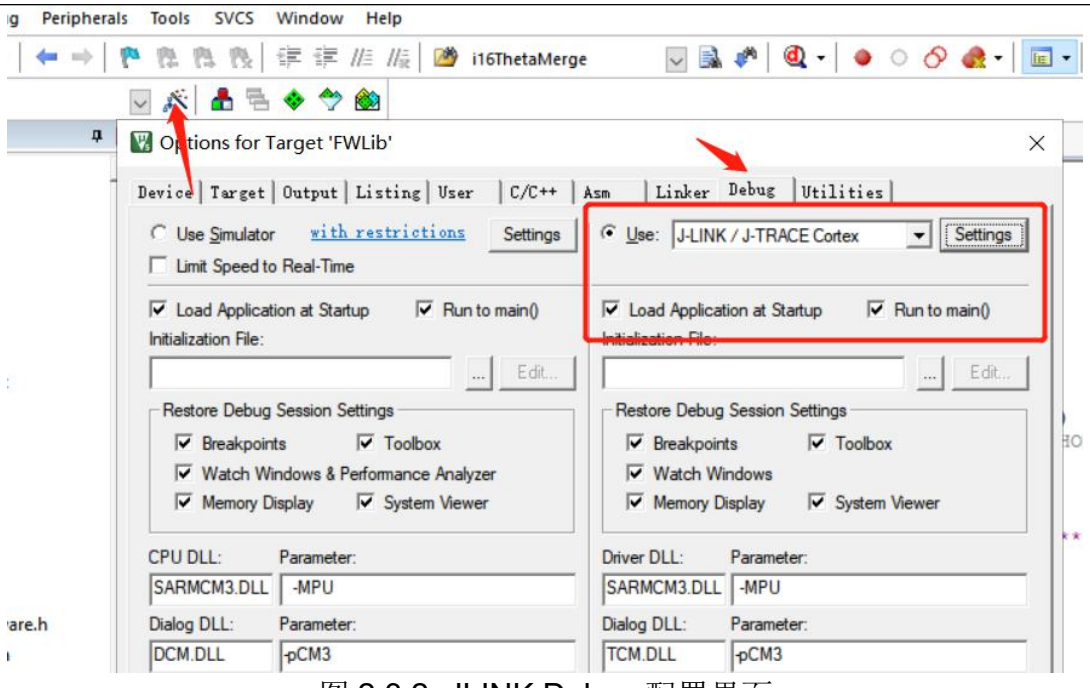


图 2.3.2: JLINK Debug 配置界面

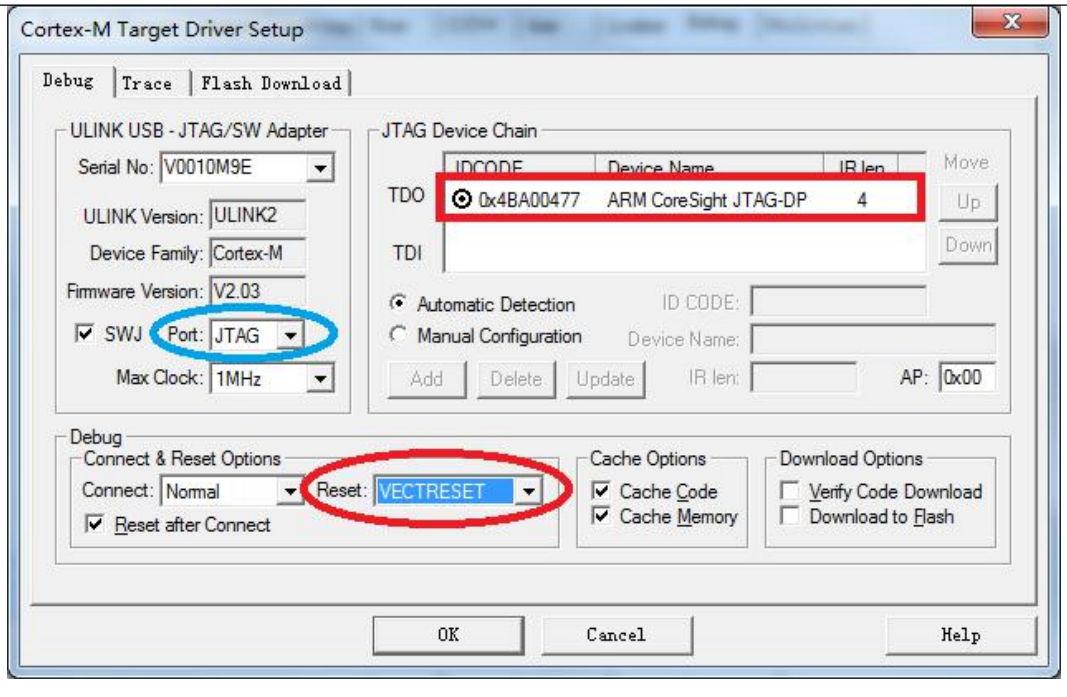


图 2- 3.3: ULINK2 设置对话框

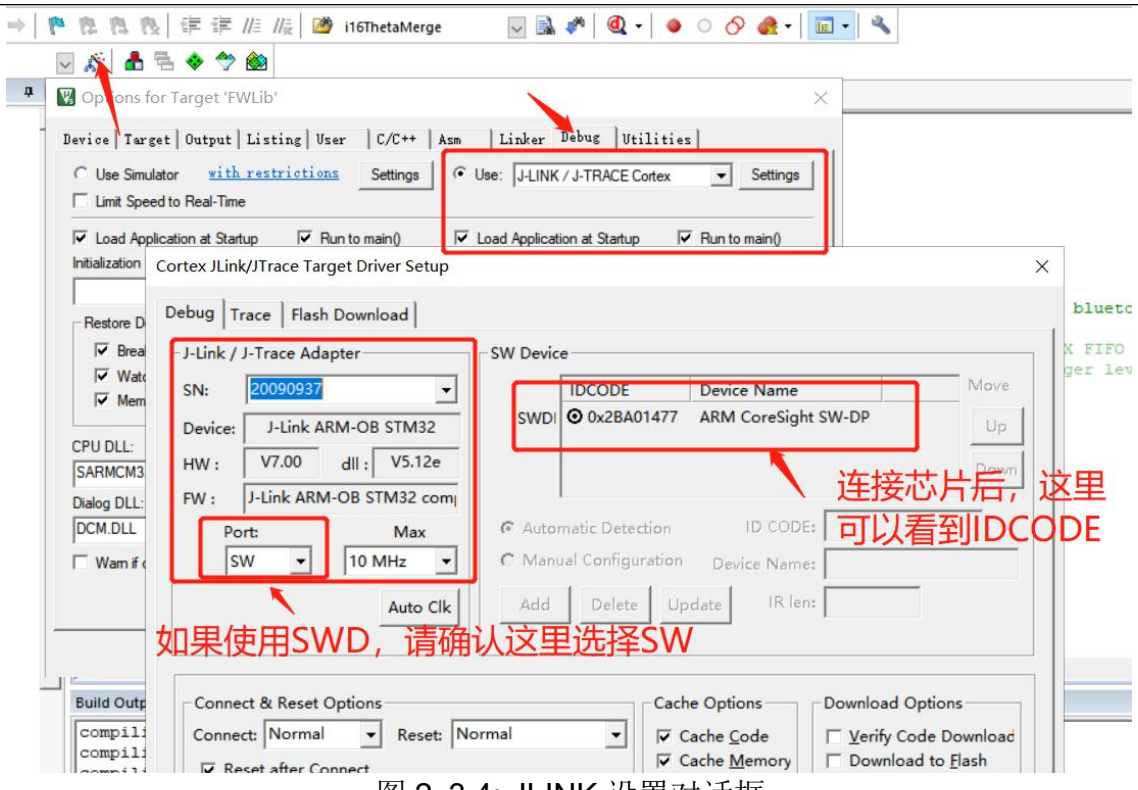


图 2-3.4: JLINK 设置对话框

接下来，在图 2-3.1 或者图 2-3.2，我们看到有两个选项：Load Application at Startup 和 Run to main()。其中 Load Application at Startup 选项是必须要勾选的，Run to main()选项根据需要决定要不要勾选。如果勾选 Run to main()选项，当启动 Debug 调试后，程序会直接 Run 到 main 函数的入口，如图 2-4 所示；相反，如果没有勾选 Run to main()选项，程序会停在 Boot Loader 程序的入口，如图 2-5 所示，此时在应用程序 main 中设置一个断点，单击按钮或者按下 F5 键，程序就会快速执行到断点处，如图 2-3.7 所示。

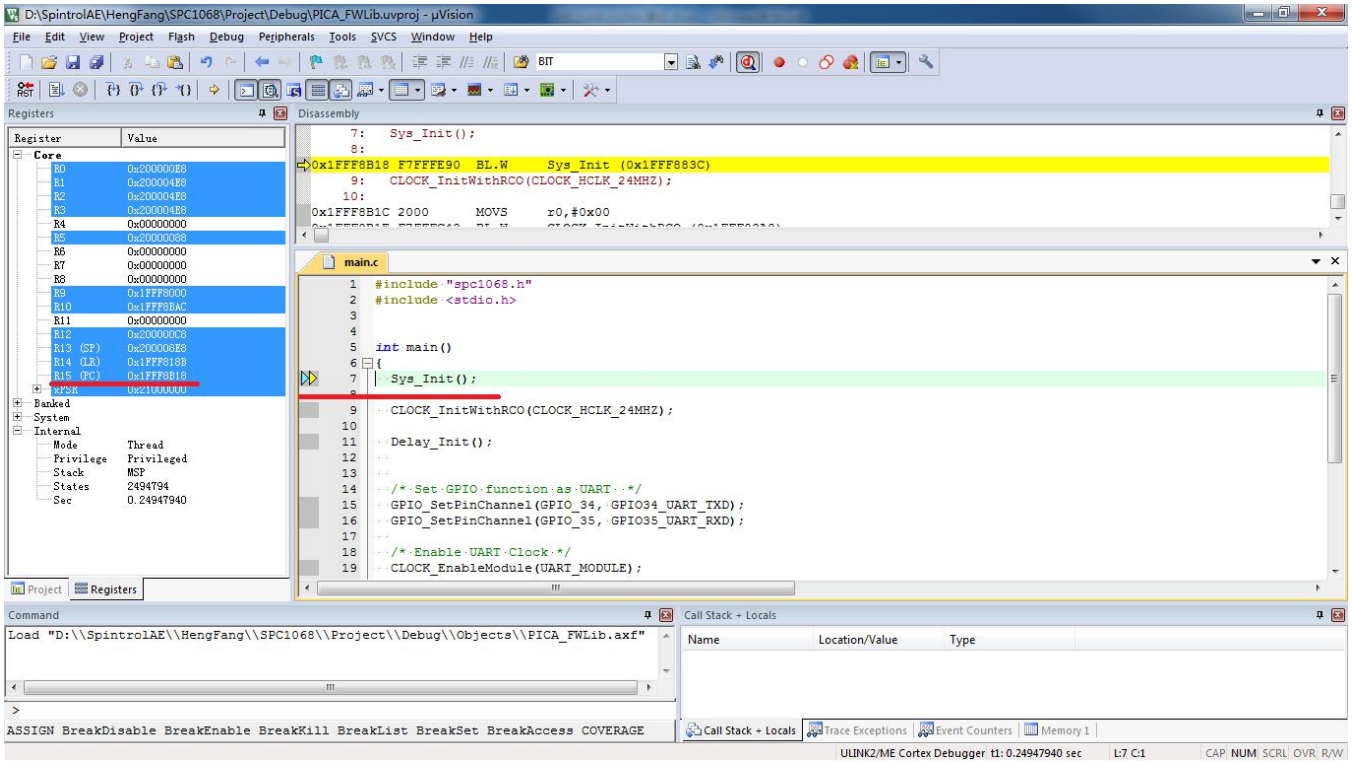


图 2- 3.5: 勾选 Run to main()选项运行结果

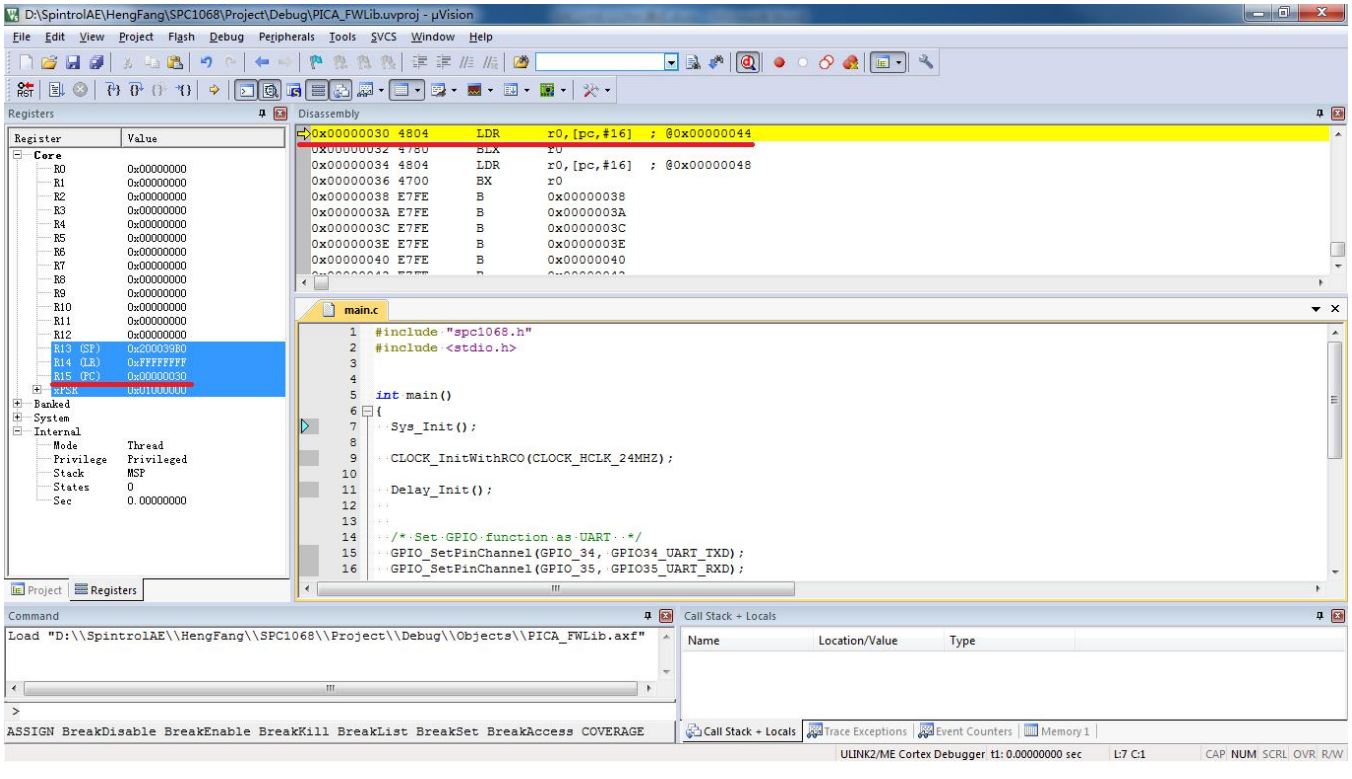


图 2- 3.6: 未勾选 Run to main()选项运行结果

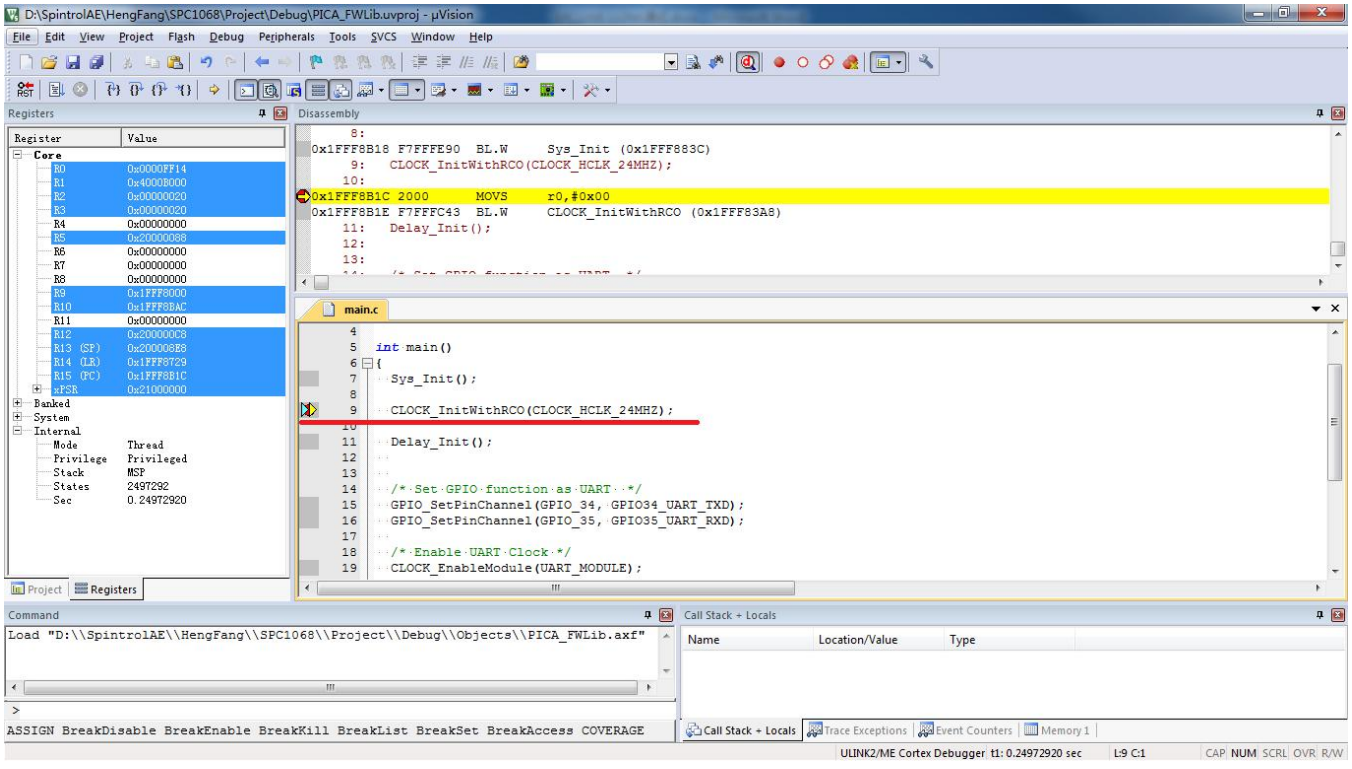


图 2- 3.7: 未勾选 Run to main()选项执行至断点情形

2.4. 配置 ALGorithm

在使用 ULINK2/J-LNK 调试程序之前，还需要设置 Flash Download 选项，如图 2-7 所示。其中，**SPC1068 Programming Algorithm 可以通过点击 Add 按钮来添加，如图 2-8 所示。（注意：需要将本目录下的 SPC1068.FLM 文件复制到 KEIL 软件安装路径下的目录 Keil_v5\ARM\Flash\）**

各种芯片对应的 FLM 文件，都能在相应的 SDK 里面找到。

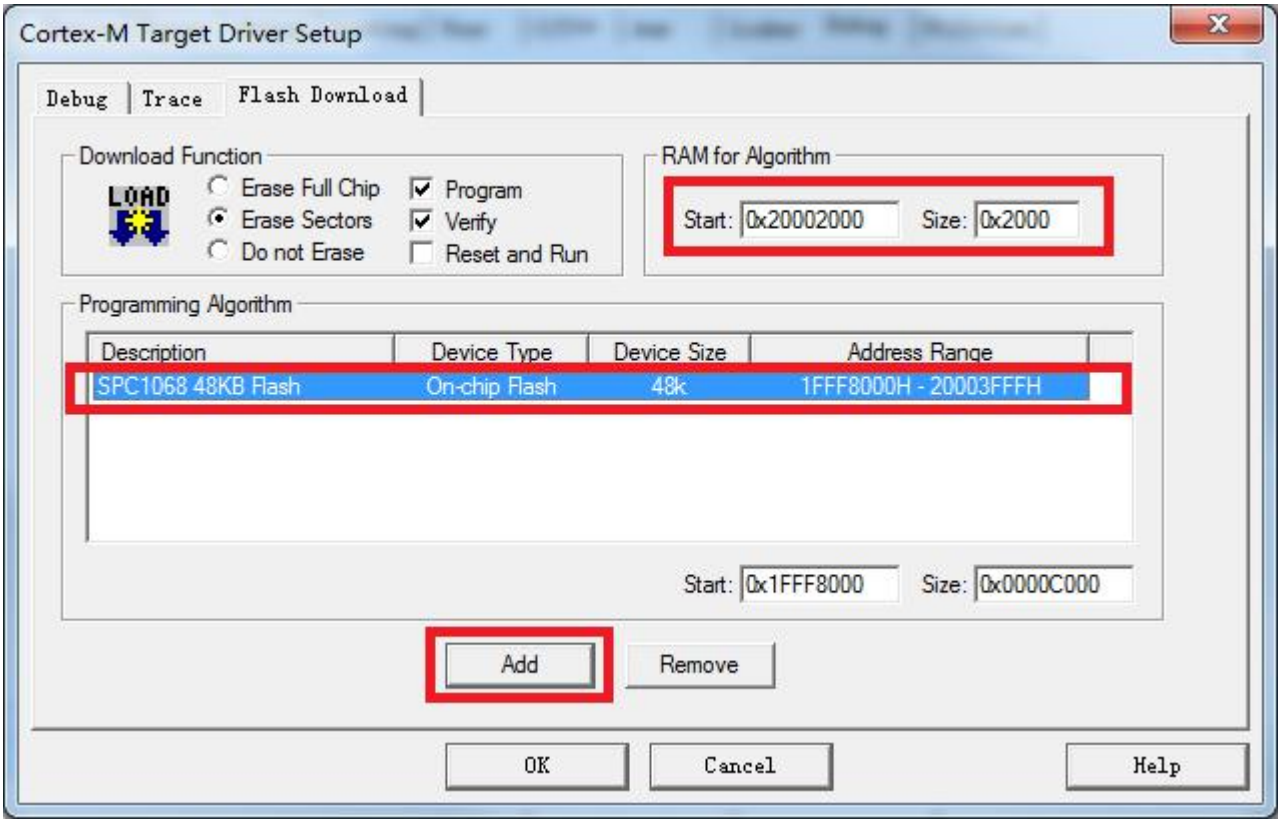


图 2- 7: Flash Download 设置

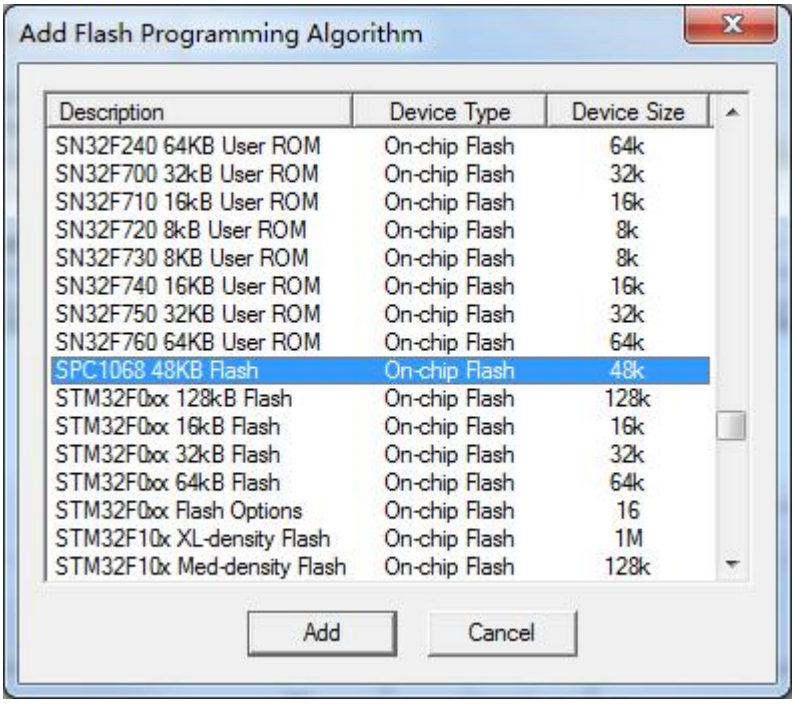


图 2- 8: Add Flash Programming Algorithm

Flash Download 设置完成之后，将应用程序编译，然后点击 KEIL 软件工具栏上的按钮，就可以将应用程序下载到芯片中。用户可以在 Build Output 窗口中查看具体的 Download 过程信息，如图 2-9 所示。

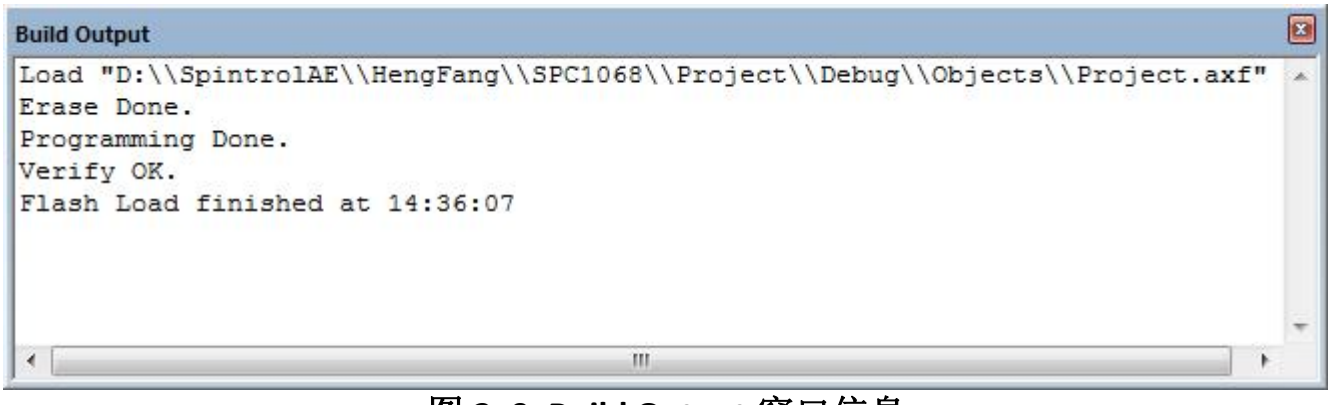


图 2- 9: Build Output 窗口信息

2.5. KEIL 环境下使用 ULINK2/JLNK 调试

根据前面的介绍，将 ULINK2/Jlink 设备与板子正确连接后，按照图 2-2、图 2-3 以及图 2-7 设置 Debug 的相关选项，就可以使用 ULINK2/Jlink 设备调试程序了。使用 ULINK2/Jlink 调试程序时，必须保证 Flash 存储器中的程序与当前程序一致。这就需要用户每次修改代码后，都要点击LOAD按钮将程序下载到 Flash 存储器中。值得一提的是，KEIL 软件提供了一个功能，可以自动上述动作，如图 3-1 所示。用户只需勾选 Update Target before Debugging 选项，那么在每次启动 Debug 会话时，KEIL 软件会自动通过 ULINK2 设备将程序下载到 Flash 中，从而保证了 Flash 中的程序与当前调试的程序一致。

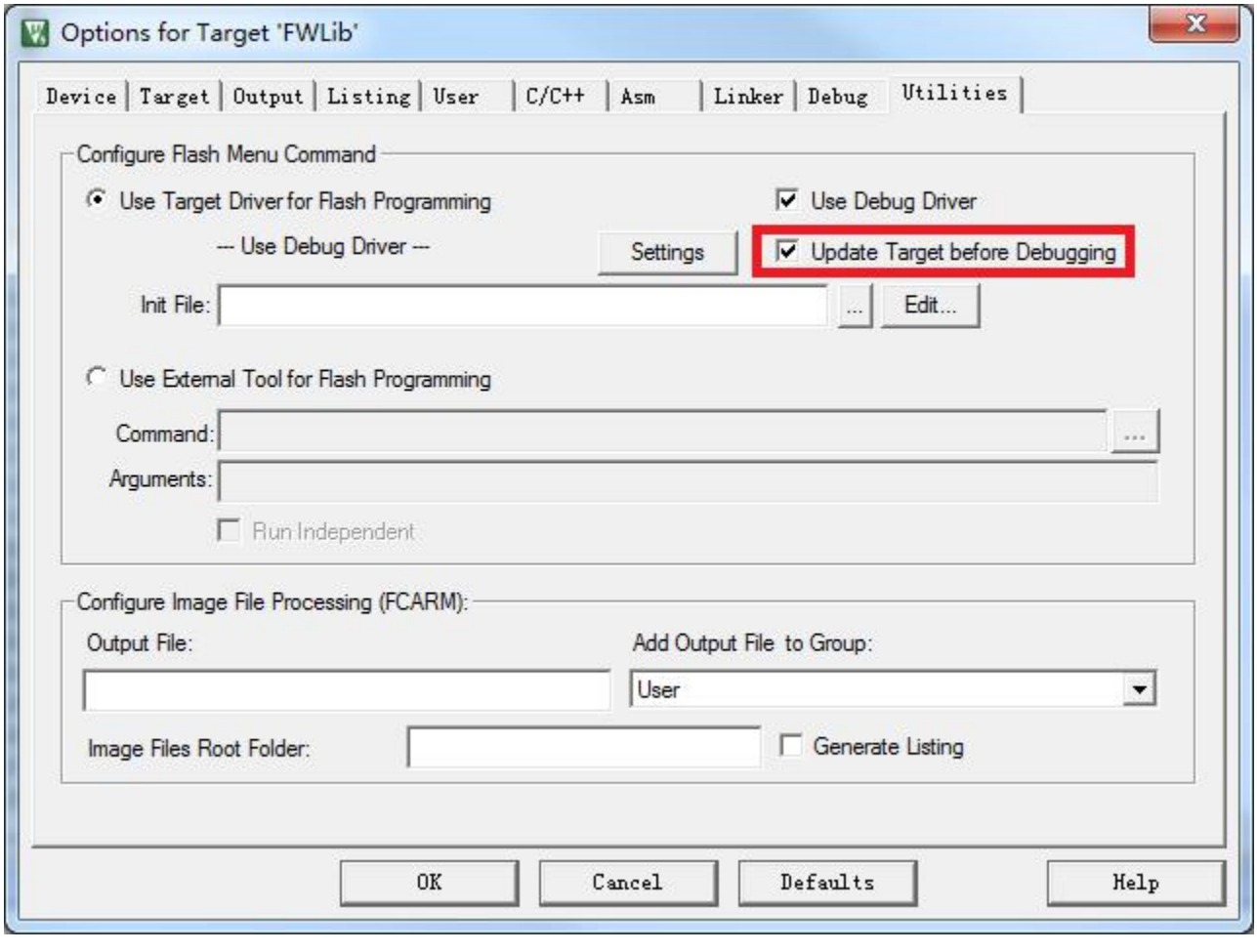


图 3- 1: Update Target before Debugging 设置

单击工具栏上的  按钮进入 Debug 状态，程序界面如图 3-2 所示。程序执行到 main 函数入口处后停止，等待用户的进一步操作。此时，KEIL 软件的界面也发生了变化：除了用户源代码窗口，还出现了汇编代码窗口和 CPU 寄存器窗口。在汇编代码窗口中，黄色底纹的汇编代码对应于用户代码窗口中光标所在位置的 C 代码；此外，菜单栏上也出现了一些与 Debug 相关的菜单选项，如表 3-1 所示。

注意：（1）目前版本的 SPC1068 芯片 Reset CPU 命令只支持 VECTRESET 功能。

（2）在程序进入 Debug 状态后，代码是不可以修改的。如果想修改代码，需要单击按钮  退出 Debug 模式，然后才能修改代码。修改后的代码编译通过后，将代码重新下载到 Flash 中，用户可以继续单击按钮  进行 Debug。

表 3- 1: Debug Menu and Commands

Debug Menu	Toolbar	Shortcut	Description
Start/Stop Debug Session		Ctrl+F5	Starts or stops a debugging session.
Reset CPU			Sets the CPU to RESET state.
Run		F5	Continues executing the program until the next active breakpoint is reached.
Stop			Stops the program execution immediately.
Step		F11	Executes a single-step into a function; Executes the current instruction line.
Step Over		F10	Executes a single-step over a function.
Step Out		Ctrl+F11	Finishes executing the current function and stops afterwards.
Run to Cursor Line		Ctrl+F10	Executes the program until the current cursor line is reached.
Show Next Statement			Shows the next executable statement/instruction.
Breakpoints		Ctrl+B	Opens the dialog Breakpoints.
Insert/Remove Breakpoint		F9	Toggles the breakpoint on the current line.
Enable/Disable Breakpoint		Ctrl+F9	Enables/disables the breakpoint on the current line.
Disable All			Disables all breakpoints in the

Breakpoints			program.
Kill All Breakpoints		Ctrl+Shift+F9	Removes all breakpoints in the program.

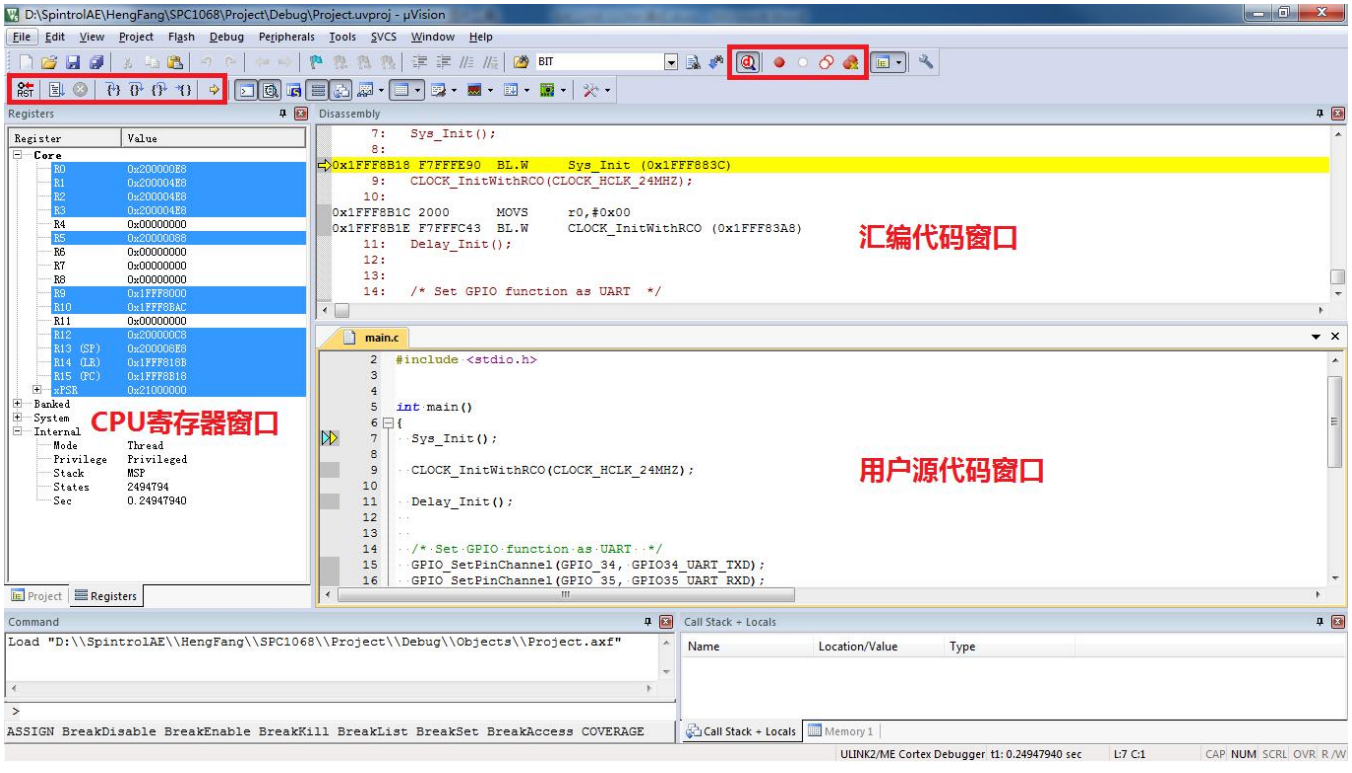


图 3-2：启动 Debug 后的界面

1) 单步调试

单击工具栏上的按钮后，程序进入 Debug 会话状态。此时单击工具栏上的按钮或者按下快捷键 F10 就可以单步执行程序。在单步调试的时候，用户代码窗口左侧边框处有两个三角箭头：表示当前光标所在的位置；表示当前位置的代码为下一次要执行的语句。因此，可以通过这两个三角箭头快速判断程序执行到哪条语句以及光标的位置。

有时候，我们希望程序能够快速地执行到某个位置，再进行单步调试。这时我们可以将光标定位到该位置，然后单击工具栏上的按钮，程序就会立即执行到当前光

标处。该功能也可以通过单击鼠标右键，在弹出的快捷菜单中选择 Run to Cursor Line 实现，如图 3-3 所示。

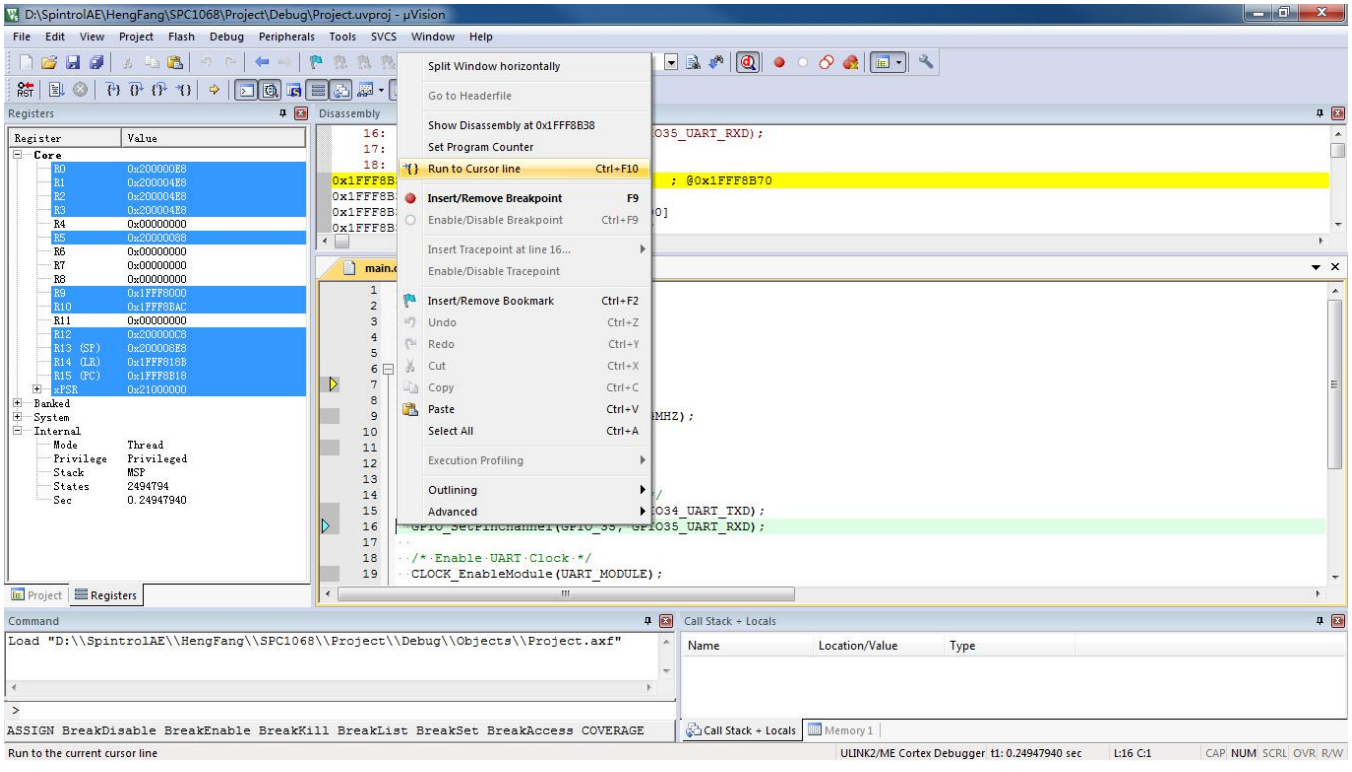


图 3- 3: Run to Cursor Line 实现

此外，我们也可以通过设置断点的方式来实现上述功能。

2) 断点设置

当启动 Debug 会话后，在用户代码所在行左侧边框处单击鼠标左键，即可快速地设置断点，此时左侧边框会出现一个红色的圆形标记，如图 3-4 所示。当然，也可以通过工具栏上的●按钮设置断点，具体做法是：将光标定位到欲设置断点的代码行，然后单击●按钮，即可设置断点。此时，单击工具栏上的按钮或者按下快捷键 F5，程序就会执行起来，一直执行到断点处停下来，如图 3-5 所示。

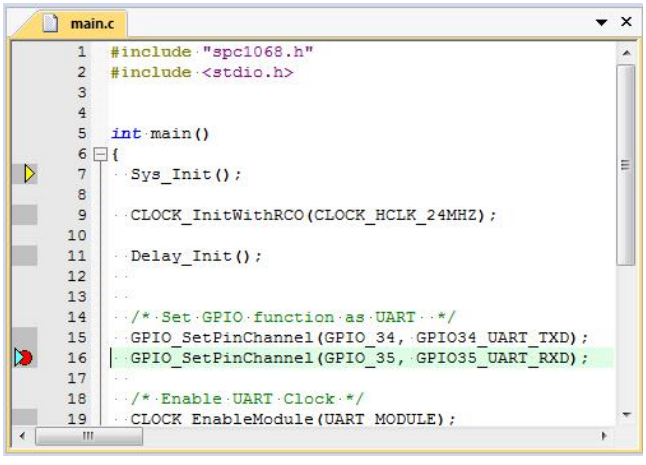


图 3-4: 设置断点

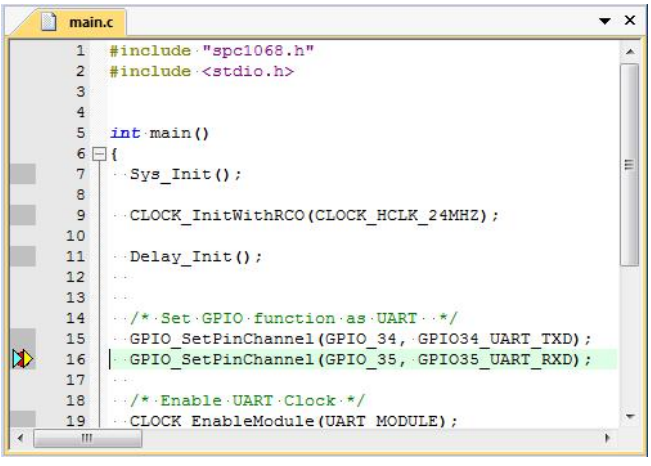


图 3-5: 程序执行到断点

设置断点除了可以让程序快速的执行到某个位置外，在 Debug 程序时也非常有用。例如，可以在中断服务函数中设置断点，用来判断相应的中断是否发生。

3) 观察变量值

在调试程序的时候，常常需要观察变量的值。这个可以通过 KEIL 软件提供的 Watch Window 来实现。具体实现过程如下：将光标定位到要观察的变量（光标移到变量名左侧），单击鼠标右键，在弹出的快捷菜单中即可将变量添加到观察窗口中，如图 3-6 所示。

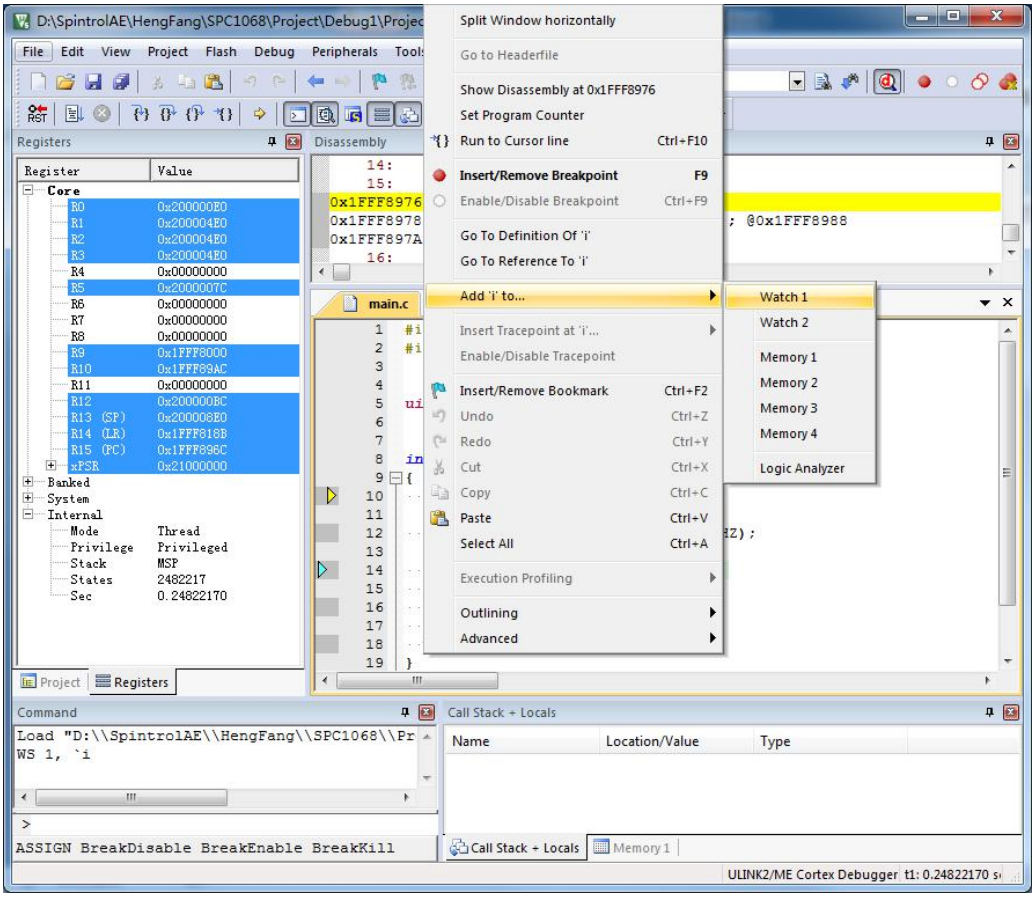


图 3-6: 添加变量到观察窗口

从图 3-6 可以看出，我们将变量 i 添加到观察窗口 Watch1 里，结果如图 3-7 所示。从图中可以看出在代码区下方出现了 Watch1 窗口，在 Watch1 中可以看到刚刚添加的变量 i。

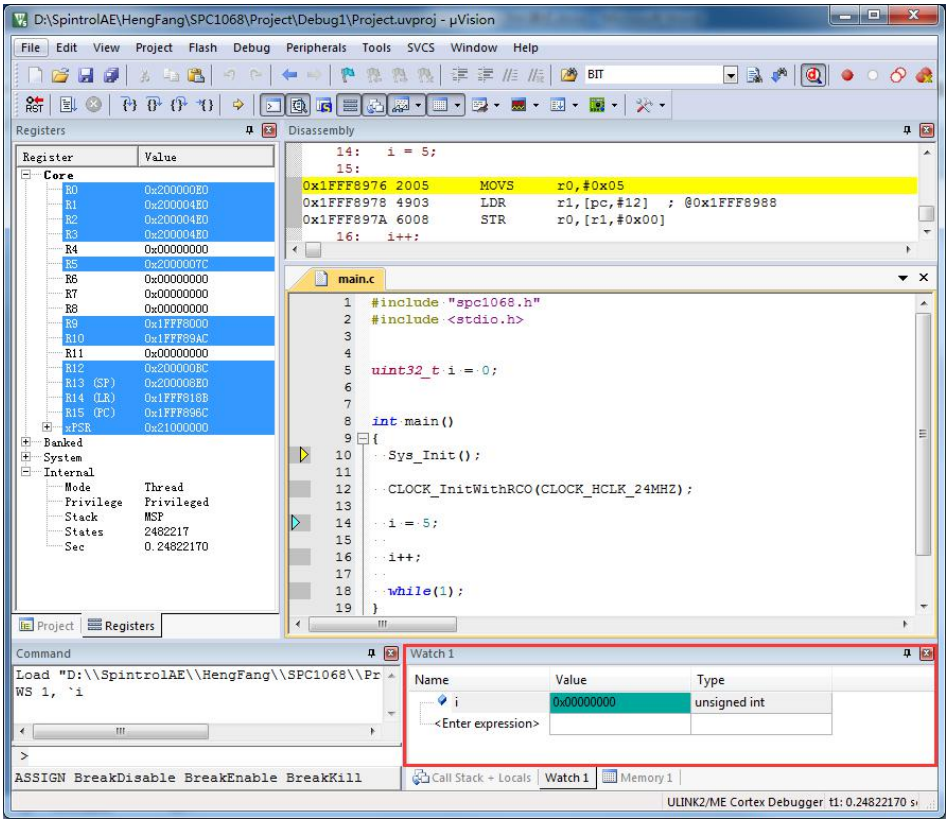


图 3-7: 添加变量到观察窗口的结果

接下来，我们就通过单步执行观察变量 i 的值。

第一步：单步执行语句 i = 5，执行结果如图 3-8 所示，可以看出变量 i 的值变为 5；

第二步：单步执行语句 i++，执行结果如图 3-9 所示，可以看出变量 i 的值变为 6。

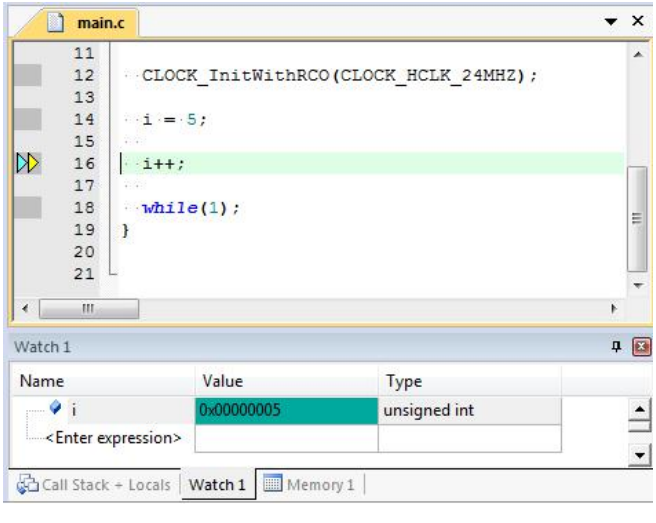


图 3-8: i=5 执行结果

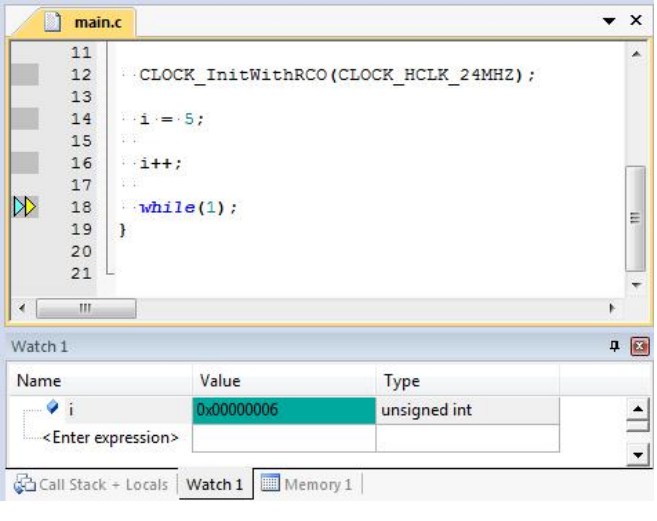


图 3-9: i++执行结果

4) 观察外设寄存器值

在调试程序的时候，我们不仅需要观察变量的值，也需要查看芯片外设 Register 的值。本节以芯片 SPC1068 外设模块 PWM0 为例介绍实现过程。

（1）通过 KEIL 软件添加芯片 System Viewer File。单击图标  ，在弹出的界面中勾选 Use Custom File 选项，然后单击图标  ，在弹出的对话框中选中芯片的 System Viewer File。设置结果如图 3-10 所示。

（2）单击  按钮，进入 Debug 模式，将芯片外设 PWM0 添加到 System Viewer 窗口，如图 3-11 所示。添加后的结果如图 3-12 所示。从图 3-12 可以看出，在 System Viewer 窗口中，不仅可以看到 PWM0 模块各个 Register 的值，而且还可以看到 Register 各个位段的值。

按照上面的步骤将 PWM0 添加到 System Viewer 窗口后，就可以在 Debug 程序的过程中观察到 PWM0 各个寄存器的值。我们单步执行图 3-12 中 main 函数的程序，分别将 TBCTL 寄存器赋值为 0 和 0x1234，结果分别如图 3-13 和图 3-14 所示。

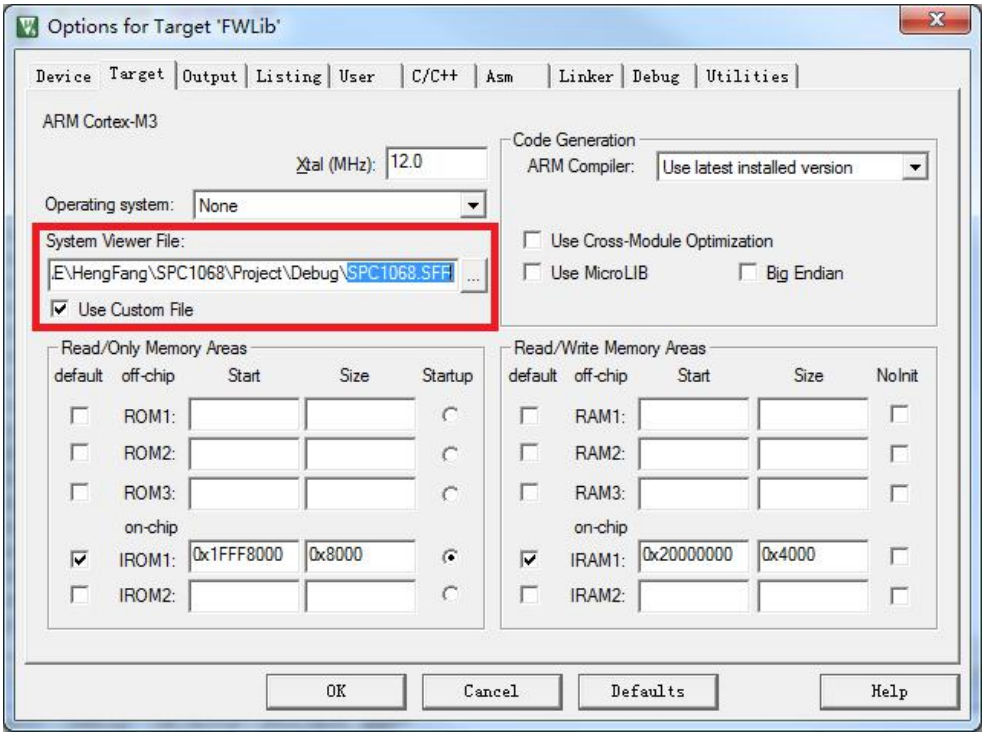


图 3-10: System Viewer File 设置界面

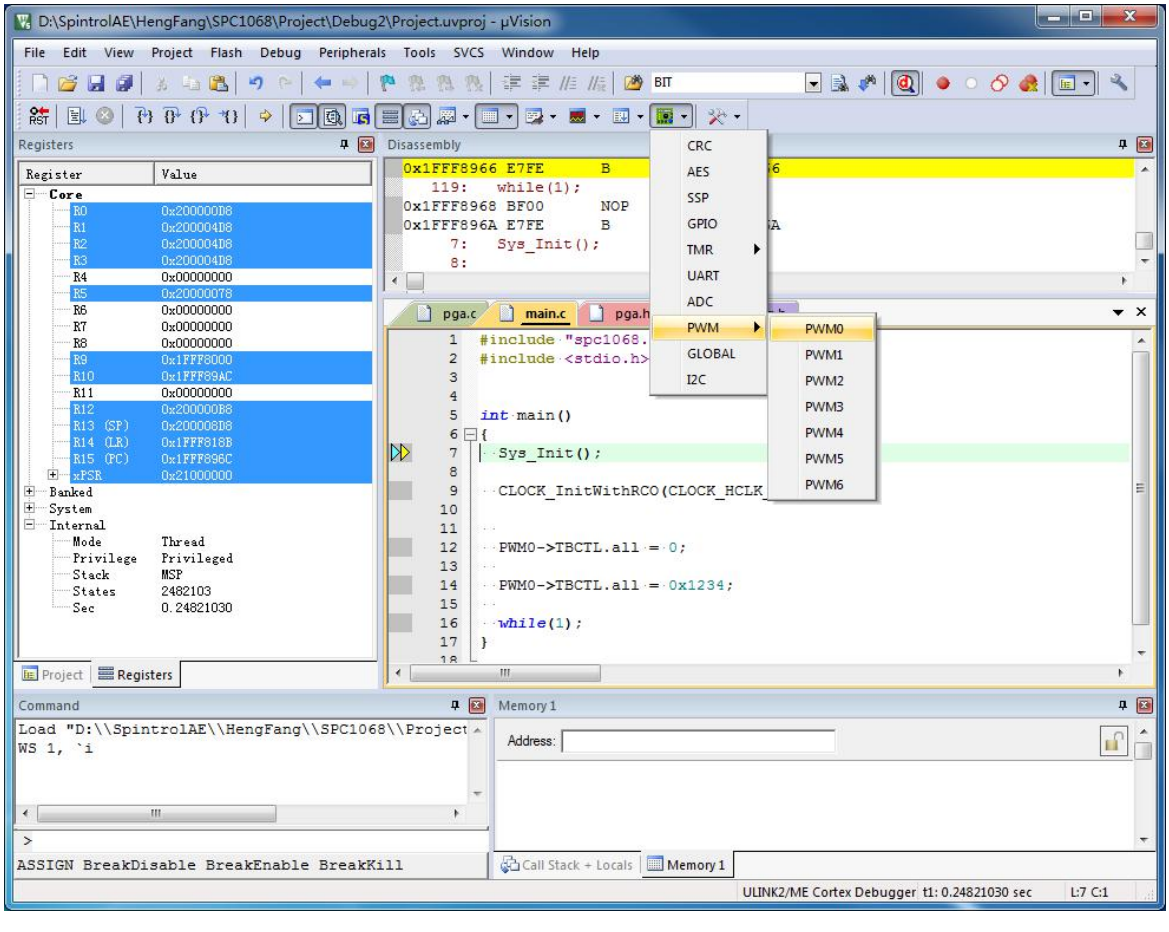


图 3-11：添加 PWM0 到 System Viewer 窗口

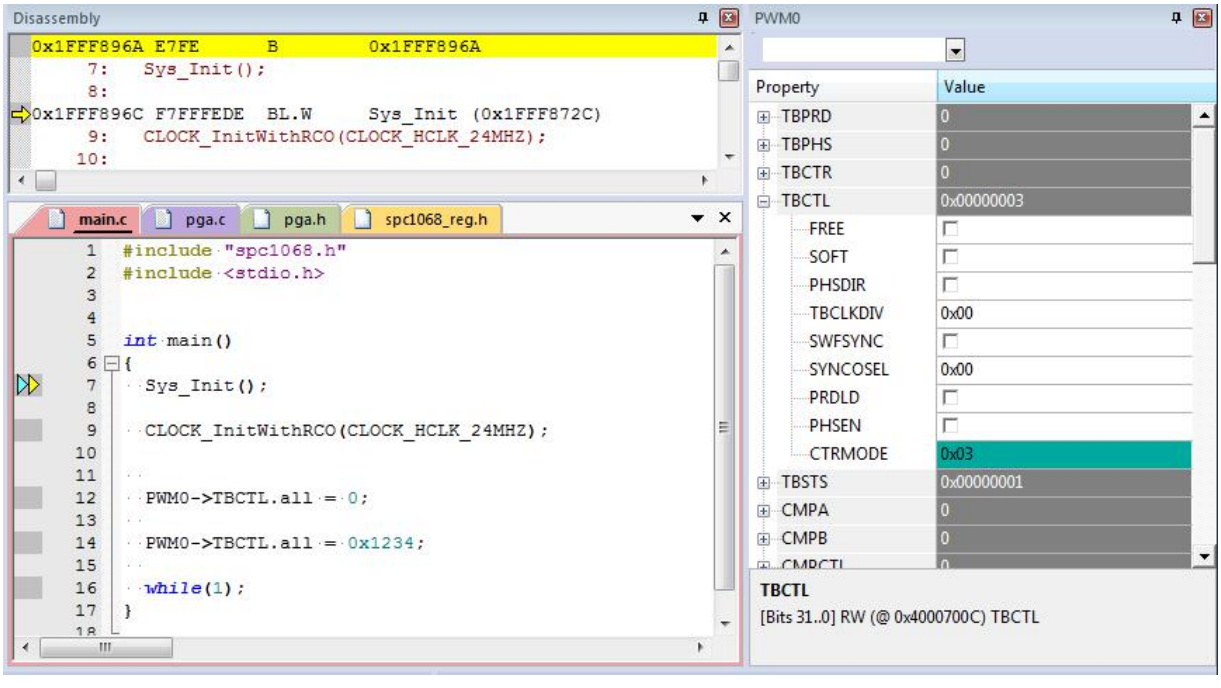


图 3-12：添加 PWM0 到 System Viewer 窗口的结果

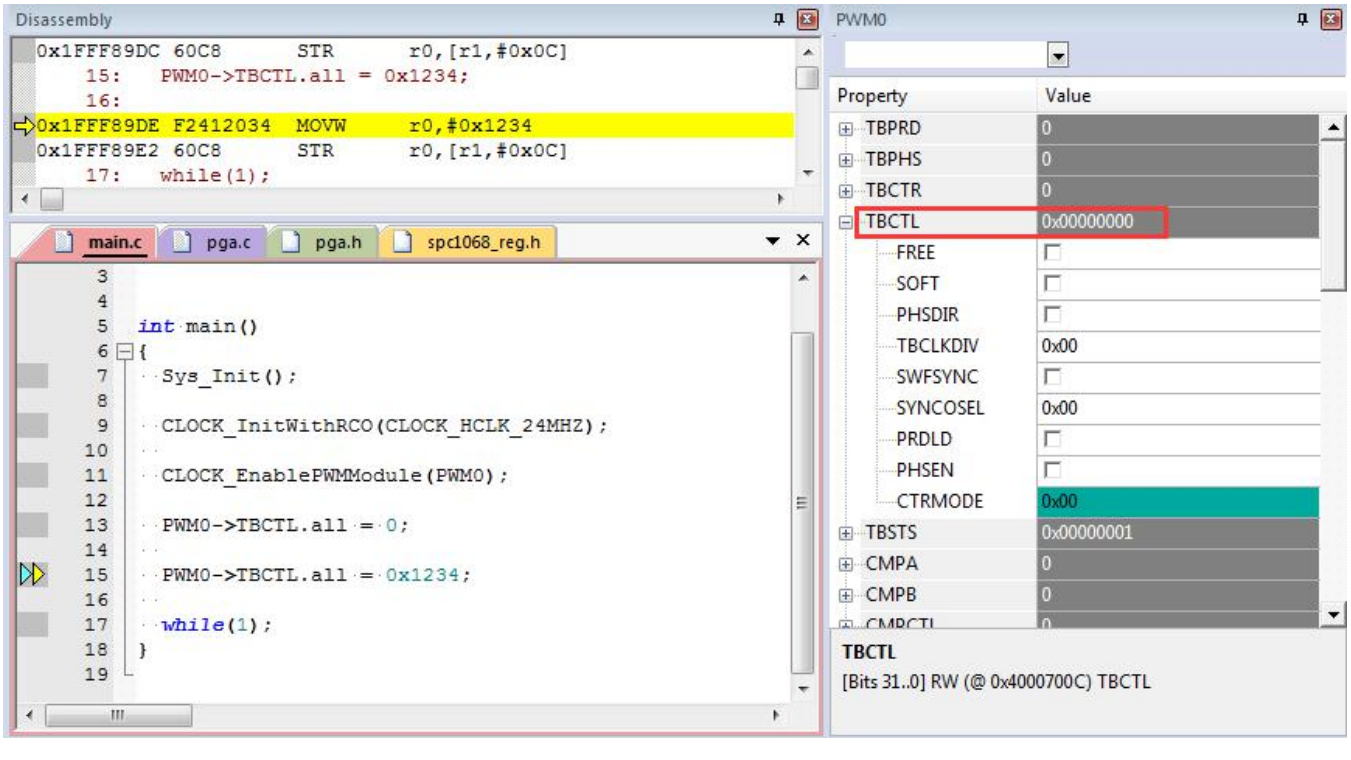


图 3-13: TBCTL=0 执行结果

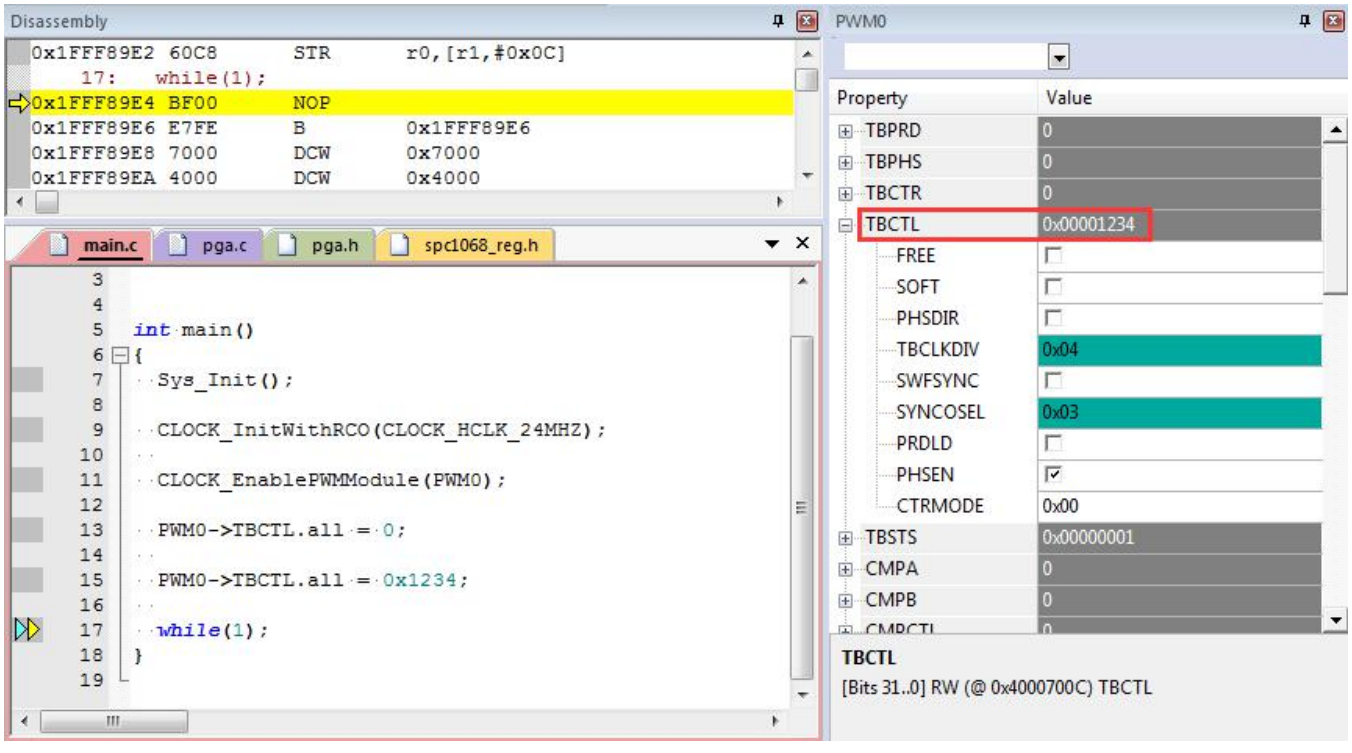


图 3-14: TBCTL=0x1234 执行结果

5) Memory 窗口

在 Debug 程序的过程中，我们还可以通过 Memory 窗口观察芯片内任一存储单元的地址。我们以章节 3.4 中的程序为例，其中 SPC1068 芯片 PWM0 模块 TBCTL 寄存器的地址为 0x4000700C。

首先，打开一个 Memory 观察窗口（Memory1），如图 3-15 所示。

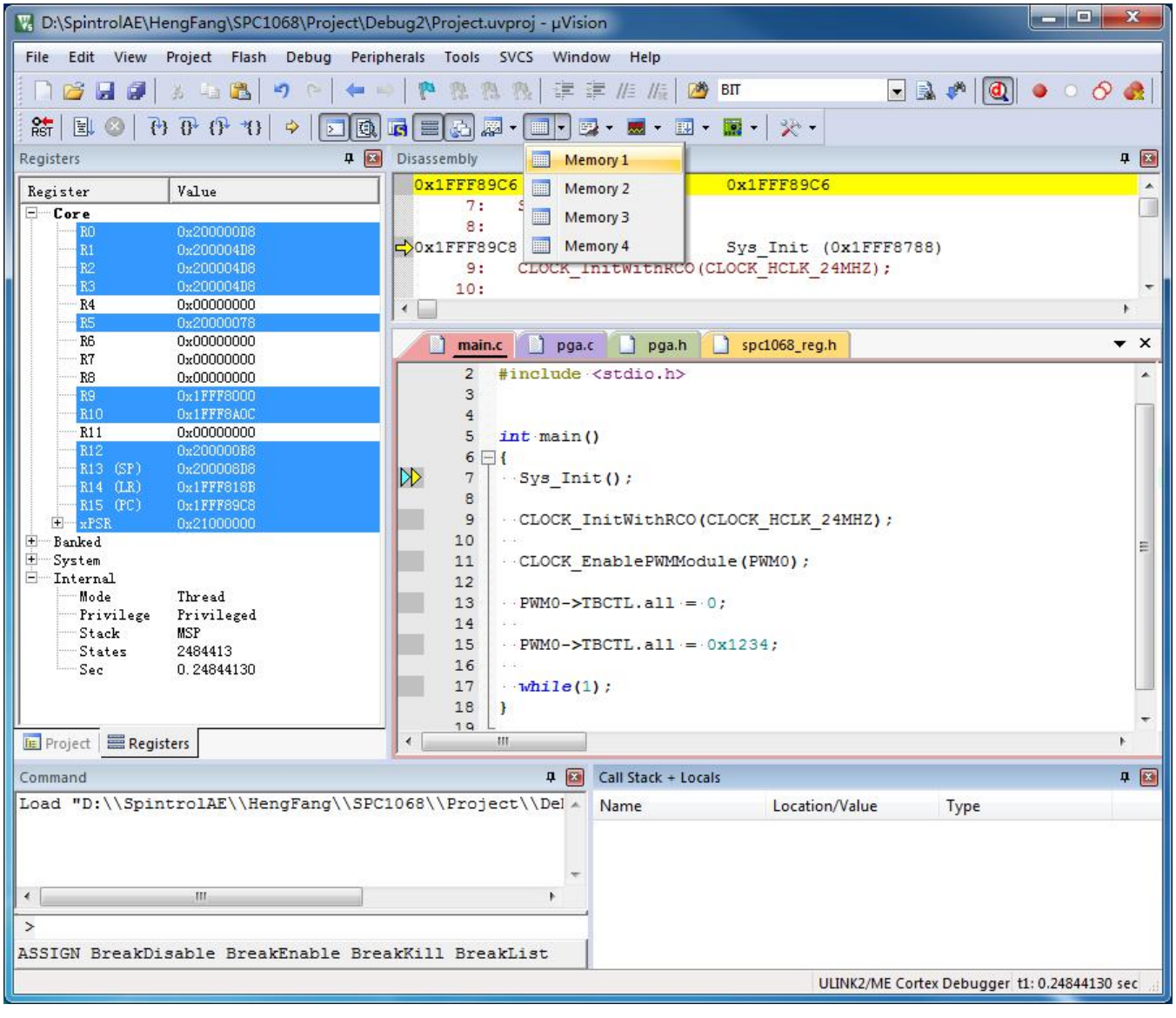


图 3-15: 打开 Memory 观察窗口

在 Memory1 窗口中输入地址 0x4000700C 后回车，结果如下：

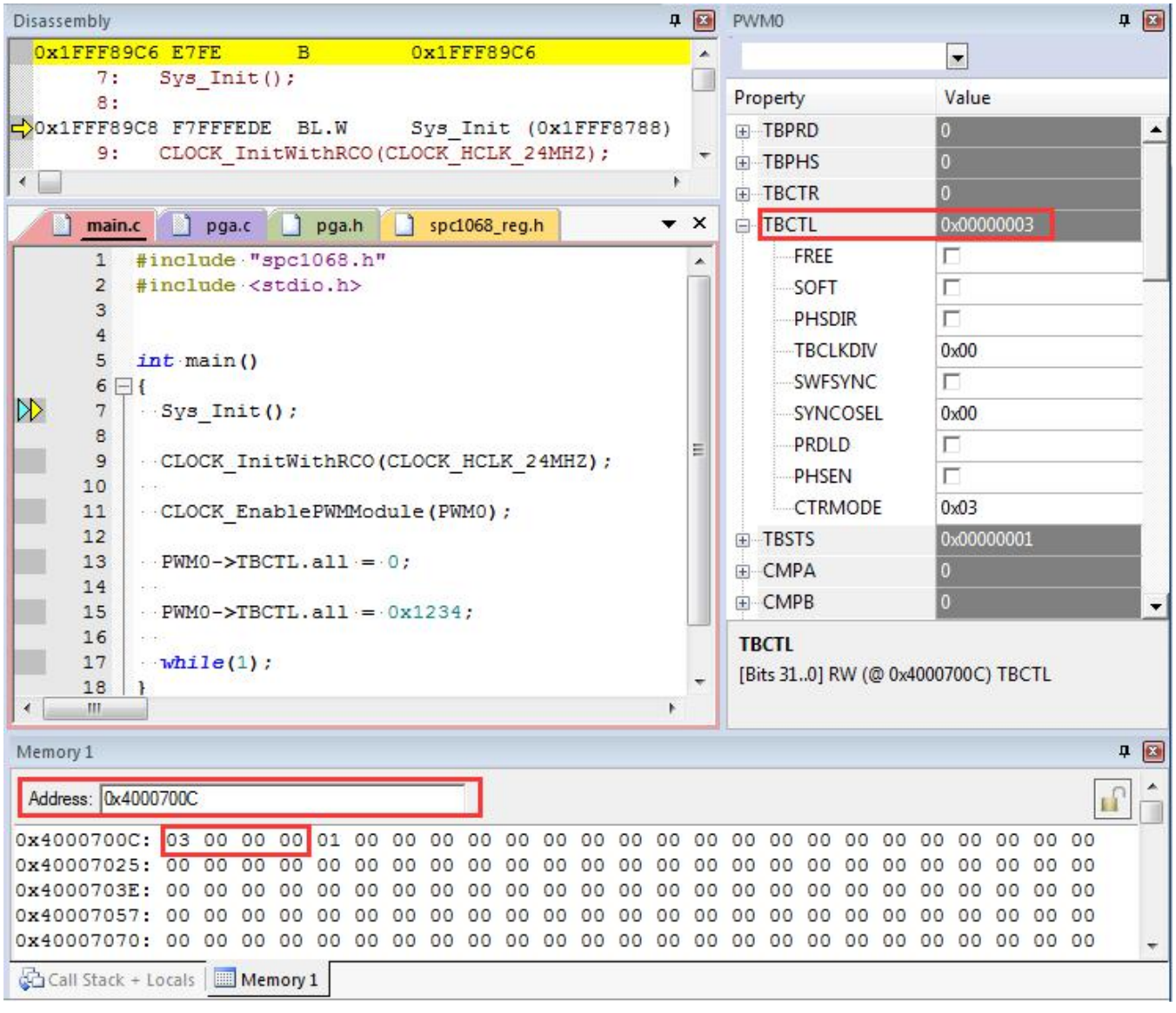


图 3-16: Memory 窗口中观察到的 TBCTL 初始值

分别单步执行图 3-16 中 main 函数的程序，分别将 TBCTL 寄存器赋值为 0 和 0x1234，结果分别如图 3-17 和图 3-18 所示。

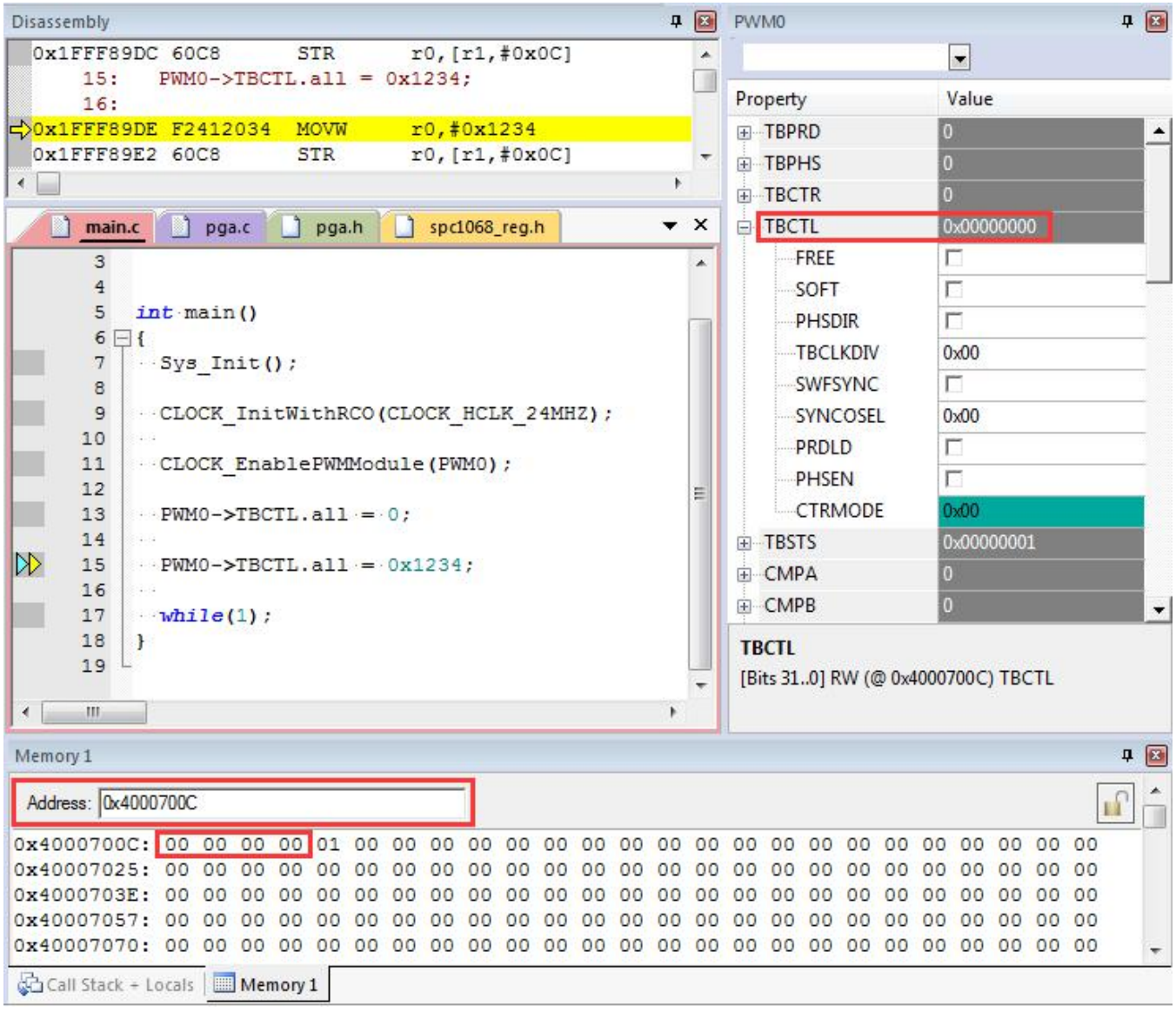


图 3-17: Memory 窗口结果（TBCTL=0）

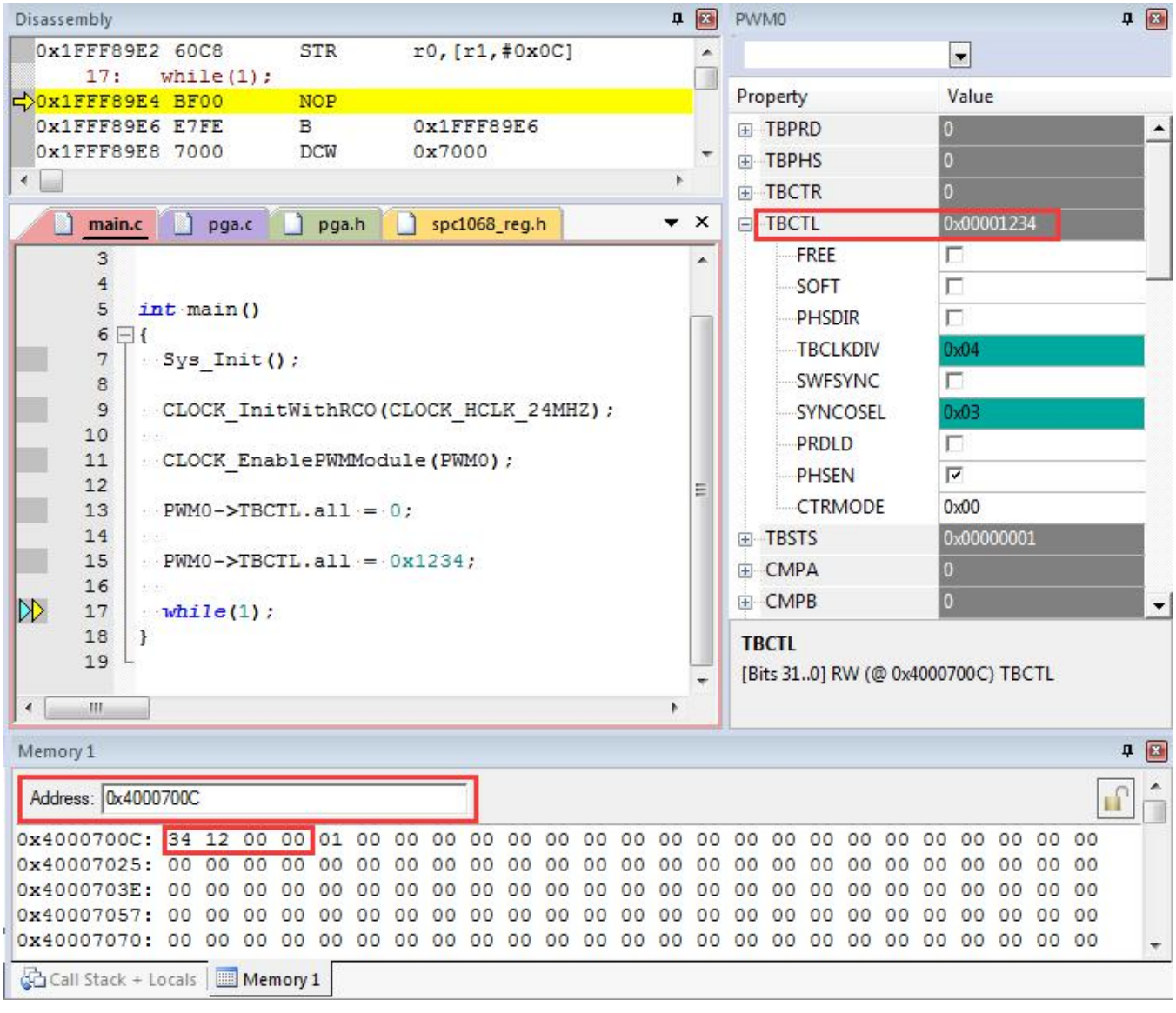


图 3-18: Memory 窗口结果（TBCTL=0x1234）

3. J-FLASH 烧录下载指南

3.1. 基本要求

J-Link 的驱动版本要是 6.16 以后的；

3.2. 更新 FLM 文件

在 J-Link 驱动的安装目录下，在 Devices 文件夹下新建文件夹 SPINTROL，并将附件中的 SPC1168.FLM 文件放到 SPINTROL 文件夹中；

注：SPC1168.FLM 是烧录中用到的 Flash 编程算法文件，不过为了适应 J-Flash 软件，做了一些修改，已经更新。不同的芯片对应的.FLM 文件有所不同，如下：

芯片	所需文件
SPC2168	SPC2168.FLM
SPD1148	SPC1168.FLM
SPC1158	SPC1168.FLM
SPC1168	SPC1168.FLM
SPD1178	SPC1168.FLM
SPD1188	SPC1168.FLM

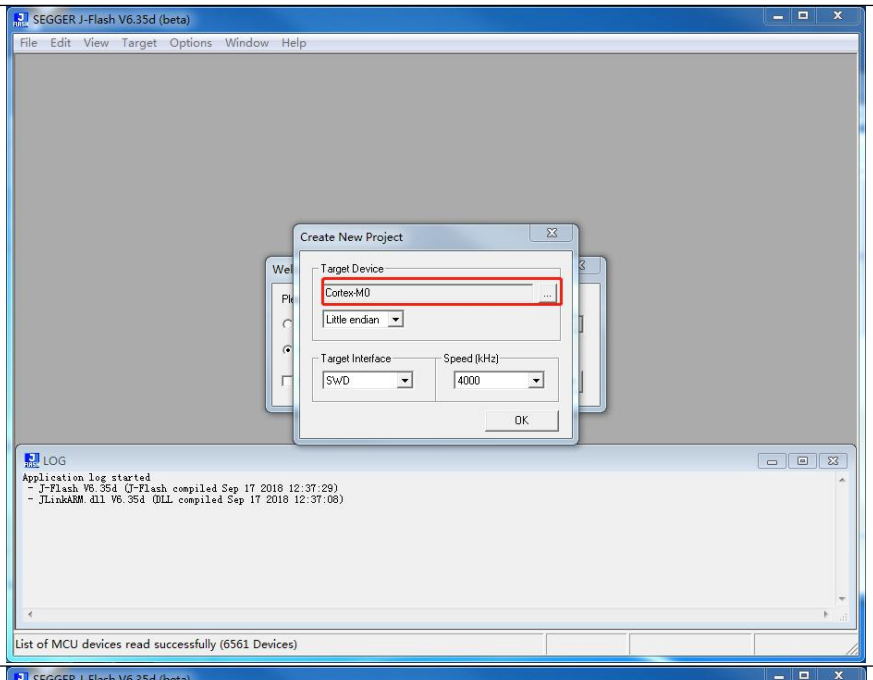
3.3. 更新 Spintrol 产品信息

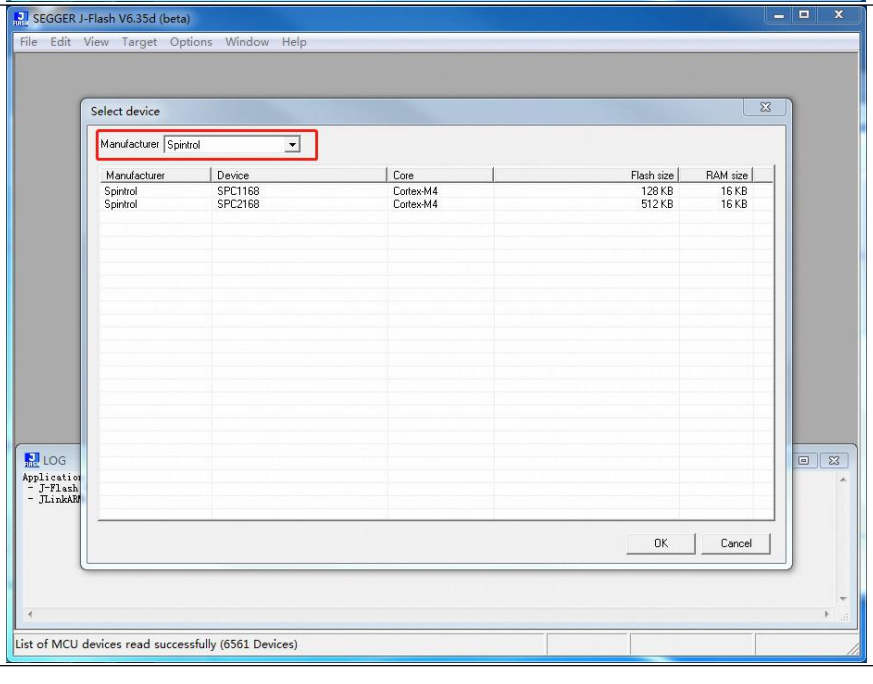
J-Link 驱动的安装目录下找到 JLinkDevices.xml，将 Spintrol 产品信息添加到里面，具体可参见附件中 JLinkDevices.xml 文件的最后的配置语句。如下：

```
<Device>
    <ChipInfo Vendor="Spintrol" Name="SPC1168" WorkRAMAddr="0x20000000" WorkRAMSize="0x4000"
Core="JLINK_CORE_CORTEX_M4"/>
    <FlashBankInfo Name="Internal Flash" BaseAddr="0x10000000" MaxSize="0x20000"
Loader="Devices/SPINTROL/SPC1168.FLM" LoaderType="FLASH_ALGO_TYPE_OPEN" AlwaysPresent="1"/>
  </Device>
  <Device>
    <ChipInfo Vendor="Spintrol" Name="SPC2168" WorkRAMAddr="0x20000000" WorkRAMSize="0x4000"
Core="JLINK_CORE_CORTEX_M4"/>
    <FlashBankInfo Name="Internal Flash" BaseAddr="0x10000000" MaxSize="0x80000"
Loader="Devices/SPINTROL/SPC2168.FLM" LoaderType="FLASH_ALGO_TYPE_OPEN" AlwaysPresent="1"/>
  </Device>
```

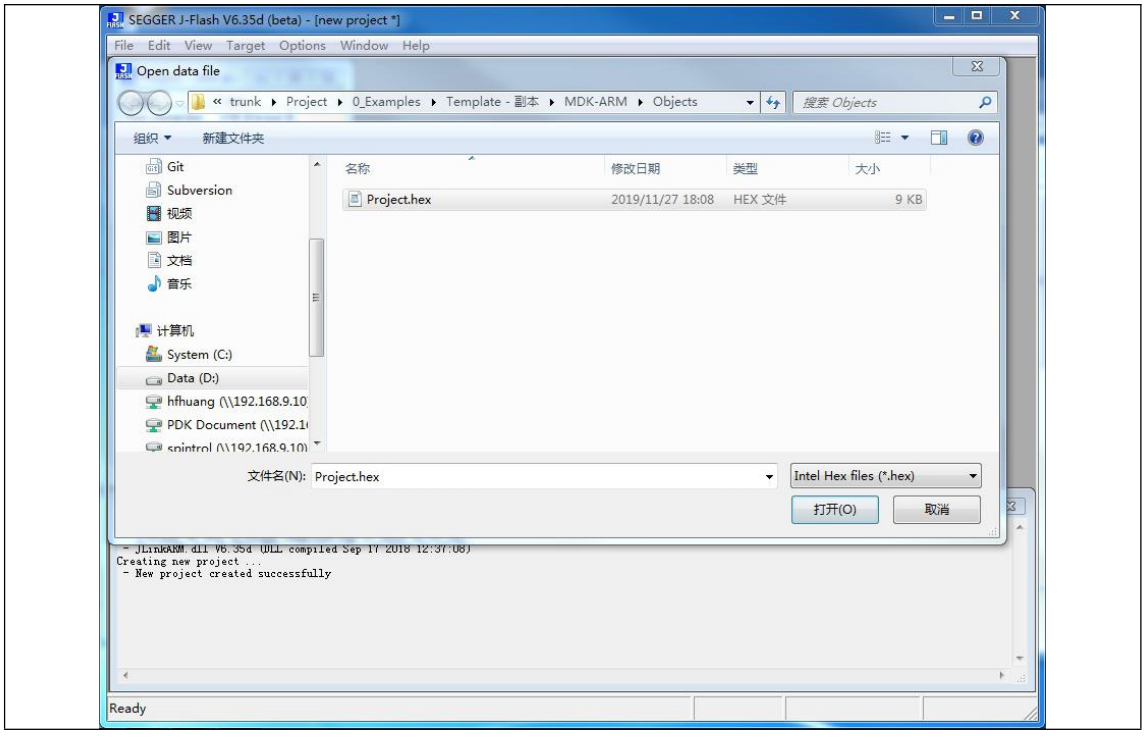
3.4. 用 J-Flash 软件烧录 Hex 文件了

1) 运行 JFlash.exe，新建一个工程，选择要烧录的芯片

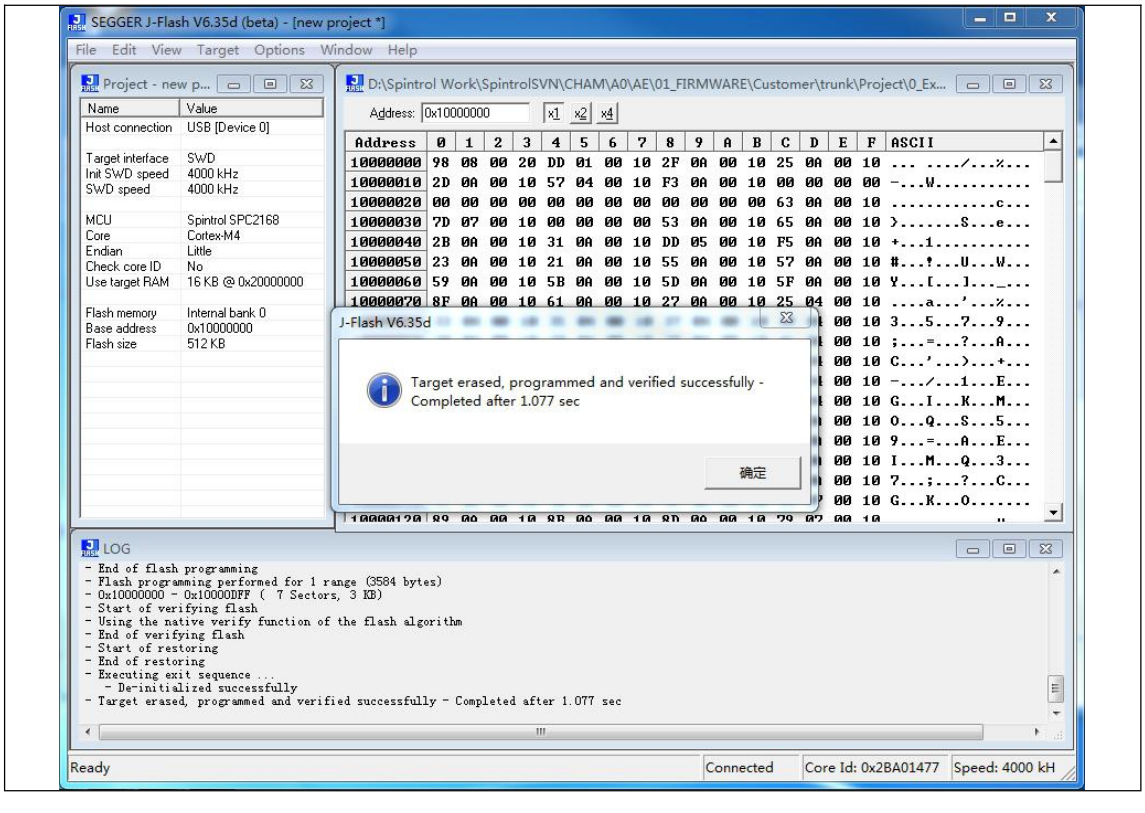




2) 选择要下载的 Hex 文件，File ->Open data file



3) 烧录芯片，Target -> Production Programming



4. ISP 串口工具概述

用户可以使用 ULINK 工具将应用程序下载到芯片中。但是这种方式不支持代码的保护等功能。Spintrol 公司提供了专门的 SPINTROL ISP 工具，帮助用户通过 UART 实现代码的下载、调试以及加密保护等功能。

SPINTROL ISP 工具运行推荐的环境配置为：

- Windows 7 操作系统
- Microsoft .NET Framework 4.0 及以上版本

如果用户电脑安装的操作系统是 Windows XP，用户需要到 Microsoft 的官网下载并安装 Microsoft .NET Framework 4.0 组件。否则，无法运行 SPINTROL ISP 工具。

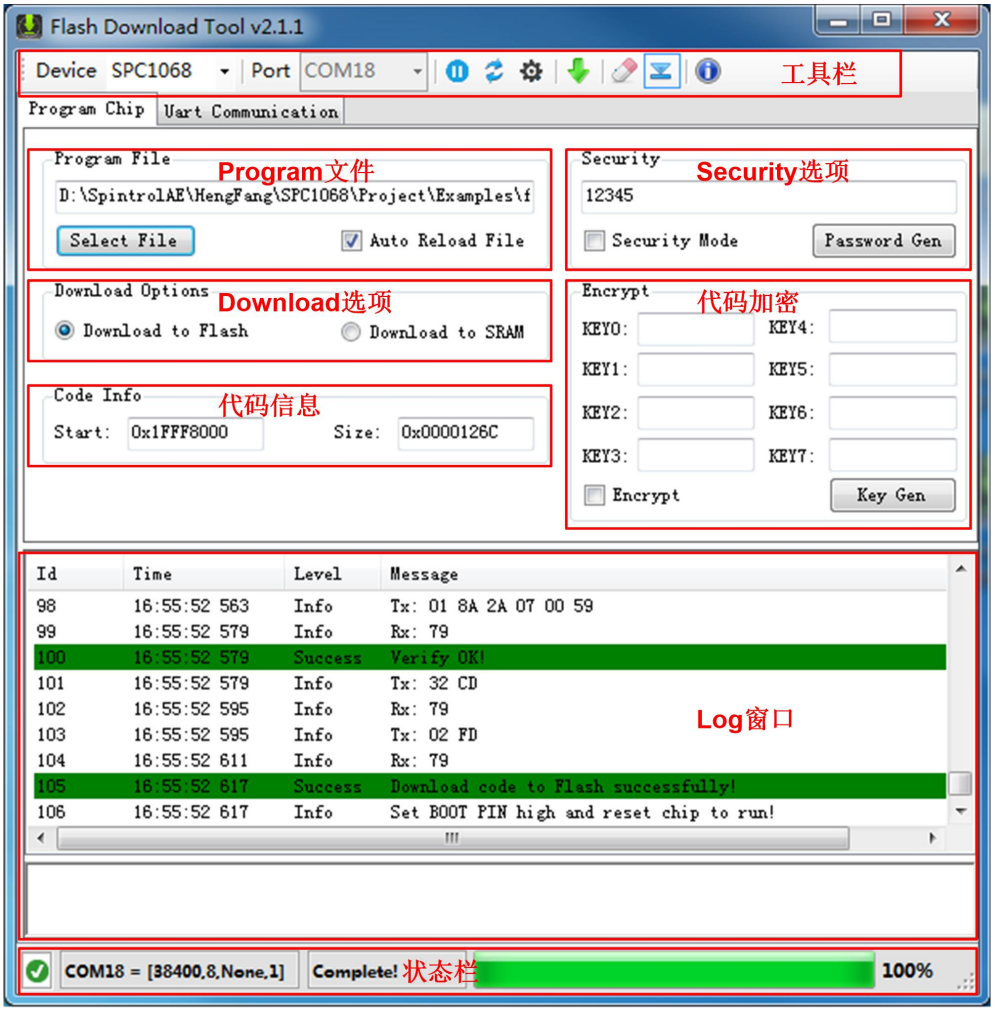


图 1- 1 : SPINTROL ISP 工具界面

用户运行 SPINTROL ISP 工具后，可以看到如图 1-1 所示的界面。在图 1-1 中，每个功能组件都用红色矩形框进行了标记。下面就逐一介绍各个功能组件的功能及使用方法。

4.1. 工具栏

工具栏 Device 列表中包含 ISP 工具目前能够支持的芯片名称，Port 列表中包含用户电脑上连接的所有可用的串口信息，用户需要从中选择所需的串口。工具栏中其他图标功能说明见表 1-1。

表 1- 1: 工具栏图标功能说明

Toolbar Icon	Description
	表明串口处于关闭状态，单击该按钮则会打开相应的串口，同时该图标变为
	表明串口处于打开状态，单击该按钮则会关闭相应的串口，同时该图标变为
	该按钮在串口关闭状态下有效，单击该按钮，ISP 工具会重新搜索所有可用的串口
	设置当前串口的通信参数，单击该按钮会弹出图 1-2 所示的对话框
	下载程序
	清空 Log 窗口
	表明 Log Auto-Scroll 功能开启，单击该按钮则会关闭 Auto-Scroll 功能，同时图标变为
	表明 Log Auto-Scroll 功能关闭，单击该按钮则会开启 Auto-Scroll 功能，同时图标变为
	单击该按钮会弹出 ISP 工具的相关说明信息

4.2. 状态栏

状态栏中各个图标的含义如图 1-2 所示。

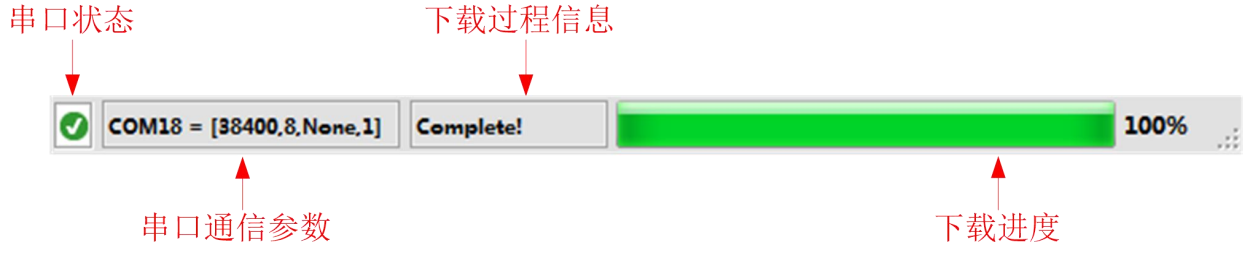


图 1-2：状态栏图标含义说明

图 1-2 中所示串口状态图标表示串口已被成功打开，串口其他状态的图标如表 1-2 所示。

表 1-2: 串口状态图标说明

Status Icon	Description
	表明 ISP 工具未发现可用的串口
	表明串口状态未知，一般 ISP 工具打开后为该图标
	表明串口打开或者关闭时出错
	表明串口处于关闭状态
	表明串口处于打开状态

此外，串口通信参数[38400, 8, None, 1]含义为：波特率 38400bps、Data Bits 为 8、无校验、Stop Bit 为 1。在下载程序时，UART 参数配置为 8 Data Bits、None Parity、1 Stop Bit。因此，用户需要确保串口相关参数的配置正确。通信波特率参数可由用户根据需要自行设定（默认设置为 38400，如果通信不正常，请联系技术支持确认波特率）。用户可以单击工具栏上的图标⚙进行串口参数的设置，对话框如图 1-3 所示。

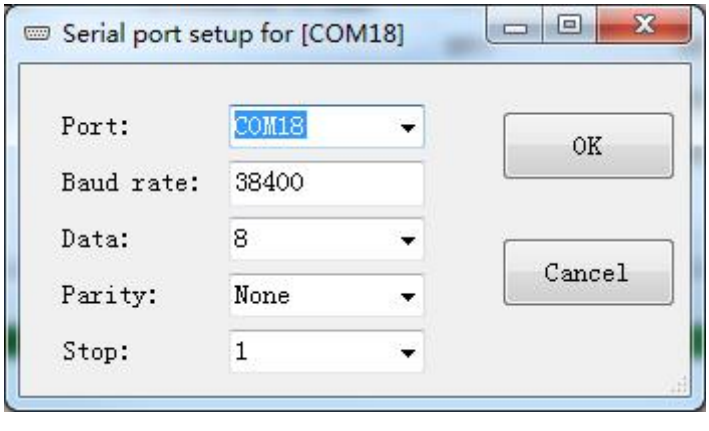


图 1-3：串口通信参数设置

4.3. Program 文件

Program 文件选项用来选择要下载到芯片中的 HEX 格式文件。单击图 1-1 中的 Select File 按钮，会弹出如图 1-4 所示的文件对话框。用户选中相应的 HEX 文件，确认即可。另外，如果勾选 Auto Reload File，则 ISP 工具在每次下载程序时，都会重新装载选中的 HEX 文件并提取文件中的数据；如果未勾选，那么每次下载到芯片中的数据都是第一次选择 HEX 文件时的数据，即使后面 HEX 文件被更新过，也不会被下载到芯片中。

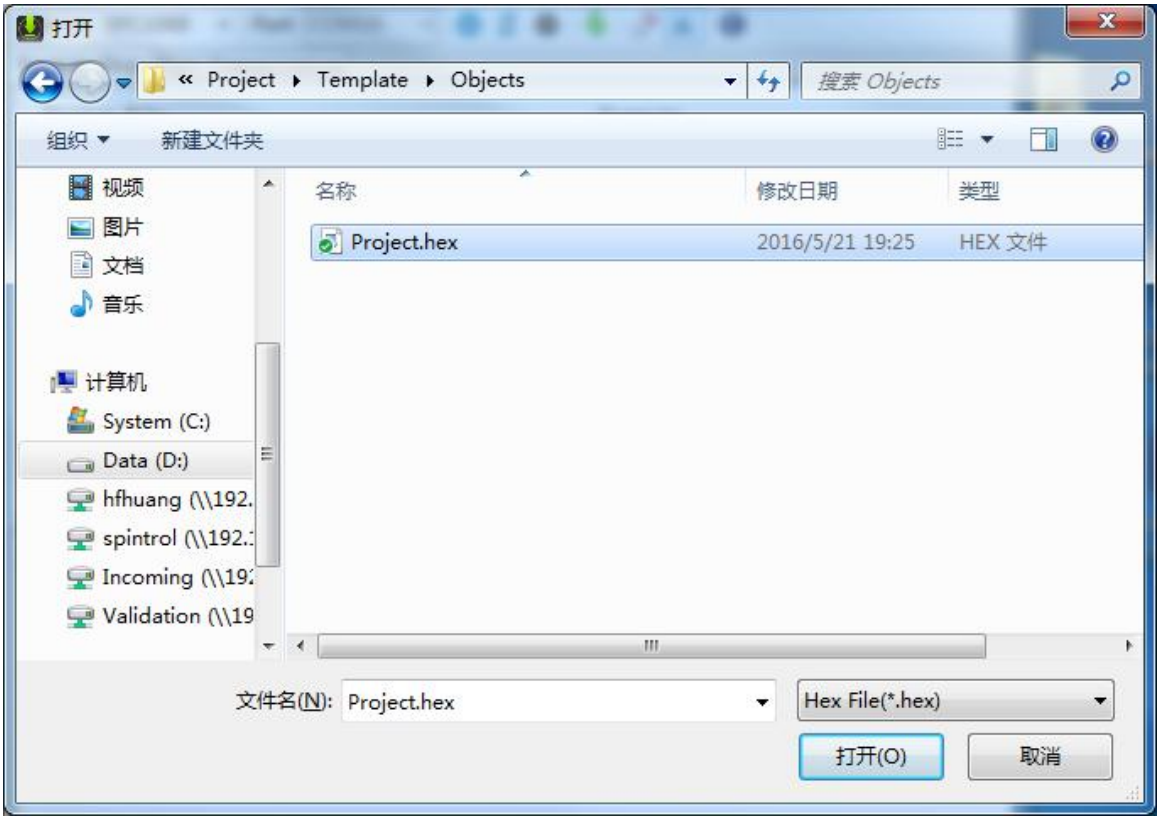


图 1-4：添加 Program 文件对话框

4.4. 代码信息

代码信息是用来给用户提供程序的起始地址以及程序大小（字节）信息。这些信息都是从选中的 HEX 文件中提取的。

4.5. Log 窗口

Log 窗口用来显示 ISP 操作信息、错误提示以及程序下载等信息。用户需要特别留意黄色和红色背景的信息：黄色背景 Log 代表警告信息；红色背景 Log 代表错误信息。

Download 选项、Security 选项以及代码加密将在下面的章节进行介绍。

4.6. 下载程序

SPINTROL ISP 工具提供两种 Download 选项：Download to Flash 和 Download to SRAM。Download to Flash 意味着用户的程序会被下载到芯片的内部 Flash；而 Download to SRAM 则意味着用户的程序被直接下载到芯片内部 SRAM 中。但是有些芯片不支持 Download to SRAM 这种模式。

（1）如果用户选择 Download to Flash 选项，那么当用户按下 ISP 工具上的下载⬇按键后，Boot Loader 会将通过 UART 接收到的数据写到 Flash 中。当程序下载成功后，用户需要将 Boot Pin 接高电平，然后按下 RESET 按键，Boot Loader 就会将 Flash 中的程序装载到 SRAM 中执行。

（2）如果用户选择 Download to SRAM 选项，那么当用户按下 ISP 工具上的下载⬇按键后，Boot Loader 会将通过 UART 接收到的数据写到 SRAM 中。当程序下载成功后， Boot Loader 就会直接执行 SRAM 中的程序。

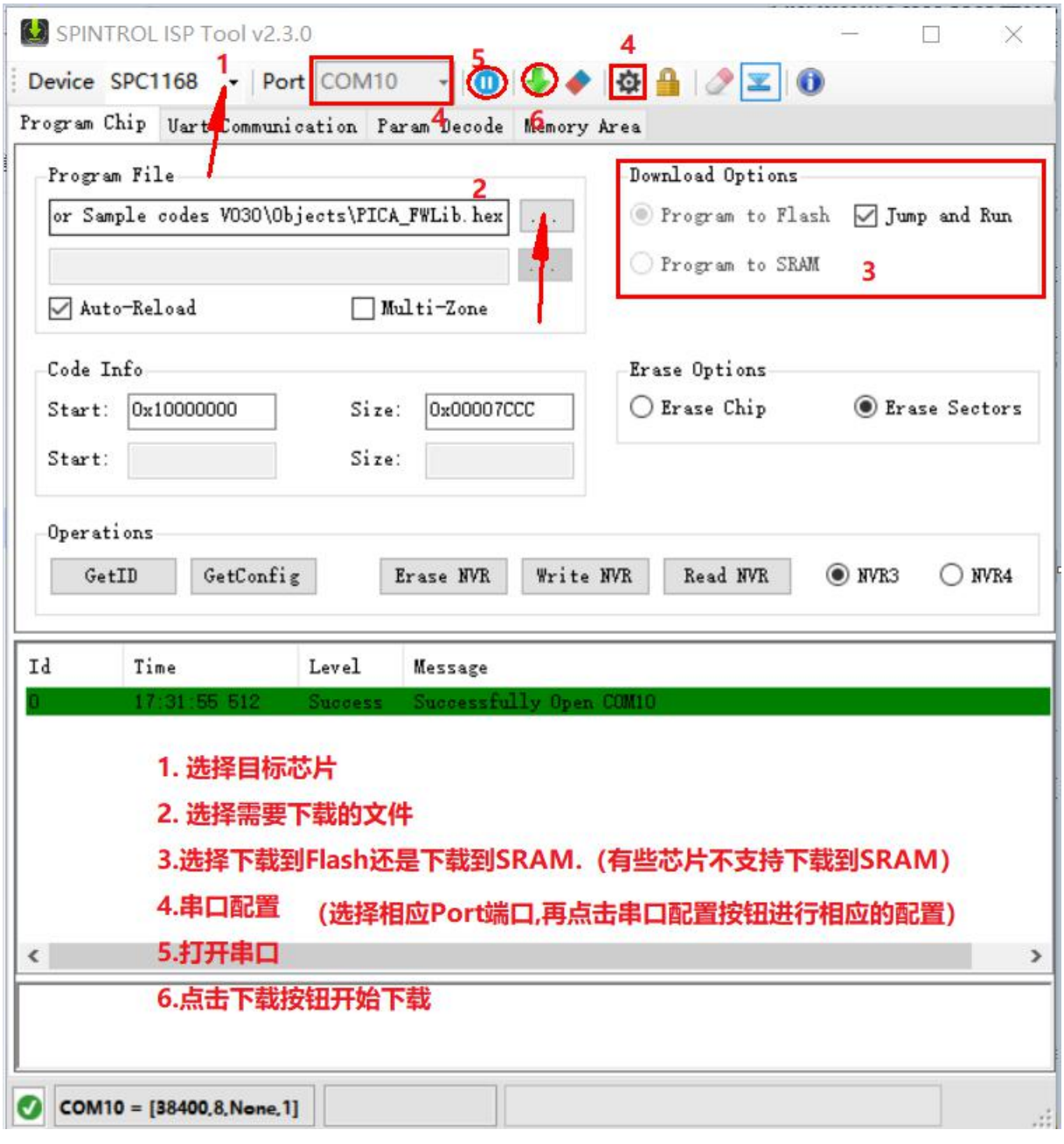
详细下载过程如下：

1) 硬件 ISP 模式选择

使用 SPINTROL ISP 工具下载程序时，需要设置 Boot Pin 管脚，然后按下 RESET 按键，此时 Boot Loader 就进入程序下载模式：

芯片	Boot PIN	TRSTn	RESET 按键
SPC1068\SPD1078	GPIO0 需要拉低	TRSTn 需要拉低	最后按下 reset 按键，进入下载模式
SPC1158\SPD1148 SPC1168\SPD1178\SPD1188	GPIO40 需要拉低	TRSTn 需要拉低	最后按下 reset 按键，进入下载模式
注： 详细可以查看芯片的 datasheet 的 Boot mode 章节，具体管脚参考 Pinout and pin description 章节			

2) SPINTROL ISP TOOL 配置



4.7. Security 功能

Security 功能主要是用来控制芯片的 Debug 模块。如果用户勾选了图 1-1 中的 Security 选项，同时用户还需要设定一个最大长度为 32 字节的密码，那么 ISP 工具会在下载程序时将这些信息传递给芯片，芯片会将这些信息加密后存储于芯片内

部。当芯片再次启动后，Debug 模块就会处于关闭状态，防止其他人员通过 Debug 接口获取芯片内部的数据。

当芯片的 Security 功能开启后，如果需要更新芯片中的程序，那么用户需要事先在 ISP 工具中输入密码。在启动程序下载功能时，ISP 工具会将该密码传递给 Boot Loader 进行校验。如果密码校验正确，那么 Boot Loader 会继续进行程序的下载；如果密码校验失败，那么 Boot Loader 就会拒绝程序下载请求并擦除芯片 Flash 中的原有数据。

注意：（1）Security 功能仅在用户选择 Download to Flash 选项时有效；

（2）当 SPC1068 Security 功能开启后，Boot Loader 会拒绝 Download to SRAM 请求；

（3）密码格式限定为英文大小写字母和数字 0~9。

4.8. 代码加密功能

代码加密功能是用来对用户的原始程序进行加密，防止其他人员通过反汇编技术获取用户源程序。如果用户勾选了图 1-1 中的 Encrypt 选项，那么 ISP 工具会在下载程序时将加密用的 KEY（32 字节）传递给芯片，Boot Loader 会利用这些 KEY 将用户程序进行加密后写入 Flash 中。程序下载完成后，芯片会将这些 KEY 加密后存储于芯片内部。当芯片再次上电后，Boot Loader 会将 Flash 中的程序进行解密，然后装载到 SRAM 中，最后执行 SRAM 中的程序。

注意：（1）Encrypt 功能仅在用户选择 Download to Flash 选项时有效；

（2）密钥 KEY 为 32 字节数据，如果 Boot Loader 解密失败，Flash 中原有的程序数据会被擦除。

4.9. UART 通信交互

SPINTROL ISP 工具为了方便用户使用 UART 调试程序，还特别集成了 UART 通信交互功能，如图 5-1 所示。

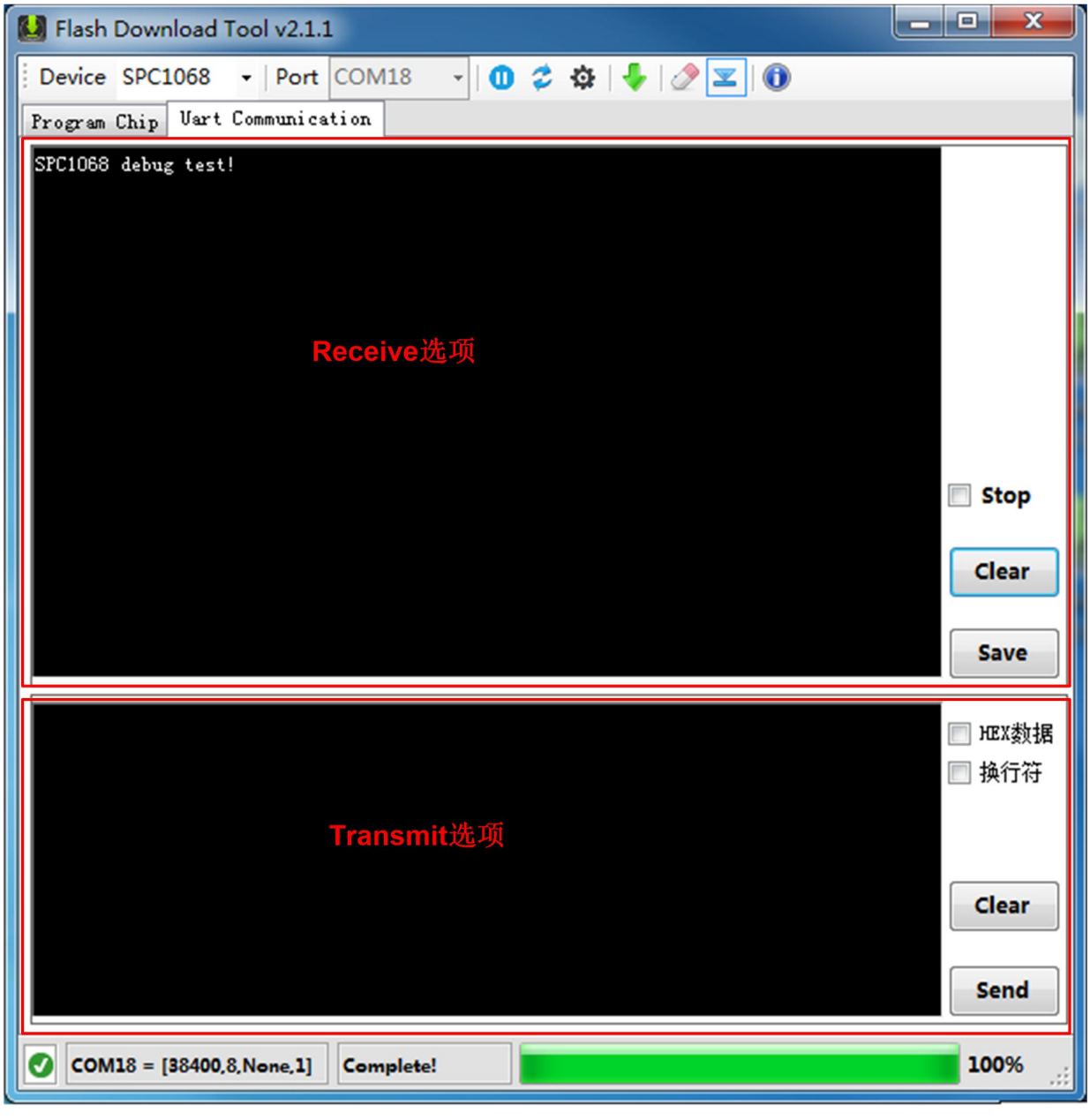


图 5- 1 : ISP 工具 UART 交互界面

从图 5-1 可以看出，UART 交互界面主要分为两大部分：上半部分为 Receive 选项，下半部分为 Transmit 选项。

Receive 选项包含以下功能：

（1）Receive 窗口：接收芯片 UART 发送出来的数据，以 ASCII 格式进行显示；

（2）Clear 按钮：单击该按钮则会清空 Receive 窗口中的信息；

（3）Save 按钮：单击该按钮则会将 Receive 窗口中的信息以文本形式保存在本地；

（4）Stop 勾选项：如果勾选该选项，ISP 工具会停止显示从芯片 UART 接收到的数据。

Transmit 选项包含以下功能：

（1）Transmit 窗口：接收用户要发送的数据；

（2）Clear 按钮：单击该按钮则会清空 Transmit 窗口中的数据；

（3）Send 按钮：单击该按钮则会将 Transmit 窗口中的数据发送给芯片的 UART；

（4）HEX 数据：如果勾选该选项，则表明 Transmit 窗口中的数据是 HEX 数据，用户输入数据时，需要用空格将各个 HEX 字节数据分开；如果未勾选该选项，则表明 Transmit 窗口中的数据为字符数据；

（5）换行符：

5. 蓝牙调试

图 5- 1： 蓝牙模块

5.1. 蓝牙模块使用

蓝牙模块是为了方便隔离调试，推荐购买渠道是淘宝店搜索 HC05 模块，就可以看到相应的模块，图 5-1 就是 HC05。购买时候请跟淘宝店主确认模块波特率以及模块密码。多数默认密码为 1234，波特率为 38400。蓝牙连接名称可能为 HC05，或者 HC01，HC02 字样。

5.2. 转接板

如果使用 SPINTROL 的开发板，接口可以接入转接板，可以直接使用转接板跟蓝牙相接入。请确保转接口可以接入转接板。图 5-2.1 为转接模块，

图 5- 2.1： 转接模块

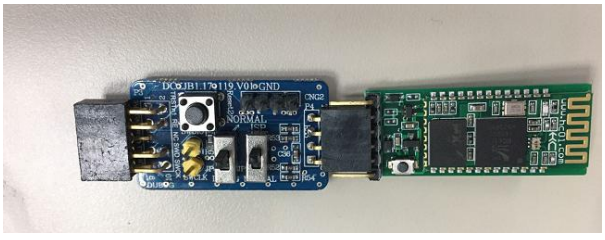


图 5- 2.2： 蓝牙与转接口

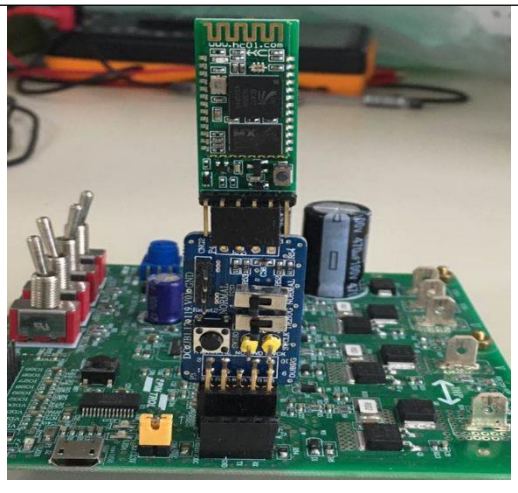


图 5- 2.3： 蓝牙与转接口

5.3. 波特率设置

蓝牙模块如果与程序的串口波特率不一致，会导致通信不上。需要进行修改相应的波特率。使用串口模块与蓝牙模块相连接，进行设置波特率。

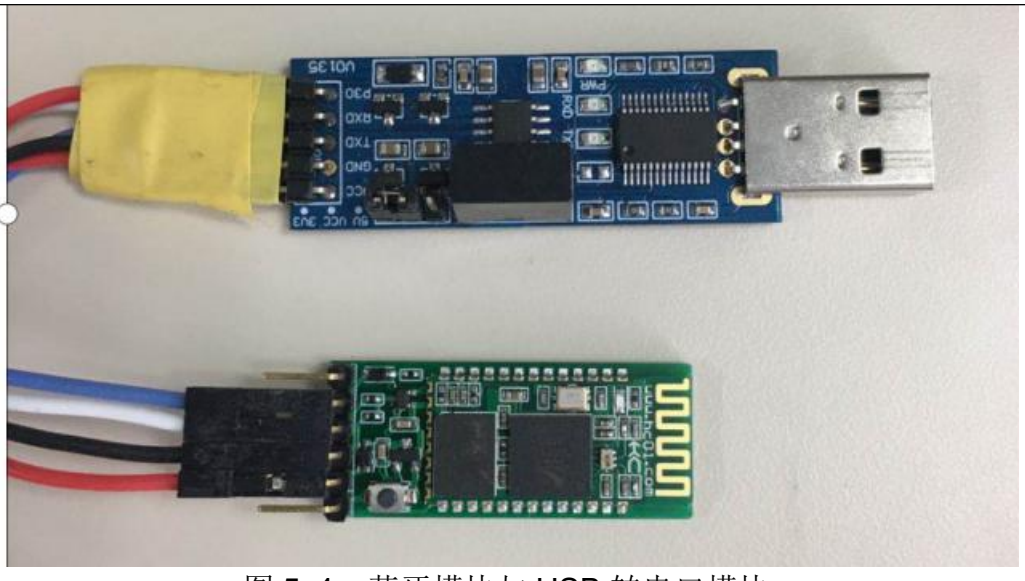


图 5-4： 蓝牙模块与 USB 转串口模块

蓝牙修改波特率

蓝牙的硬件连接如下：

蓝牙模块	USB 转串口模块
STATE	不需要接
RXD	TXD
TXD	RXD
GND	GND
VCC	VCC
KEY	不需要接

按下蓝牙的按键，然后进行上电，上电的时候后可以松开按键。此时蓝牙灯慢闪，通过串口工具发送相应的指令可以进行波特率修改。指令如下：

串口指令	是否需要发送换行符	功能
AT+UART=57600,0,0	是	修改波特率为 57600
AT	是	返回 OK 表示通信正常
AT+PSWD=1234	是	设置蓝牙连接密码为 1234
AT+NAME=SPINTROL	是	设置蓝牙名称为 SPINTROL

注： 串口的波特率设置 38400，可以使用发送 AT+换行符进行测试是否通信 OK。

5.4. 配置软件波特率与蓝牙波特率一致

```
#endif

  GPIO_SetPinChannel(GPIO_34, GPIO34_UART_TXD);
  GPIO_SetPinChannel(GPIO_35, GPIO35_UART_RXD);
  UART_Init(UART, 57600);
  #if ((UART_MODE == 1) || (UART_MODE == 2) || (UART_MOI
    UART_SetFifoTriggerLevel(UART,UARTFCR_BIT_ITL_RX_FIF
                             UART_TXFIFO_EMPTY);
    UART_EnableRxTimeoutInt(UART);
  #endif
  printf("*************************Reboot**************
#endif
```

图 5-4：配置软件串口波特率

5.5. 添加蓝牙串口进行串口下载

图 5-5.1：win10 添加蓝牙设备

图 5-4.2 win10 添加蓝牙名称为 sg57600 的蓝牙

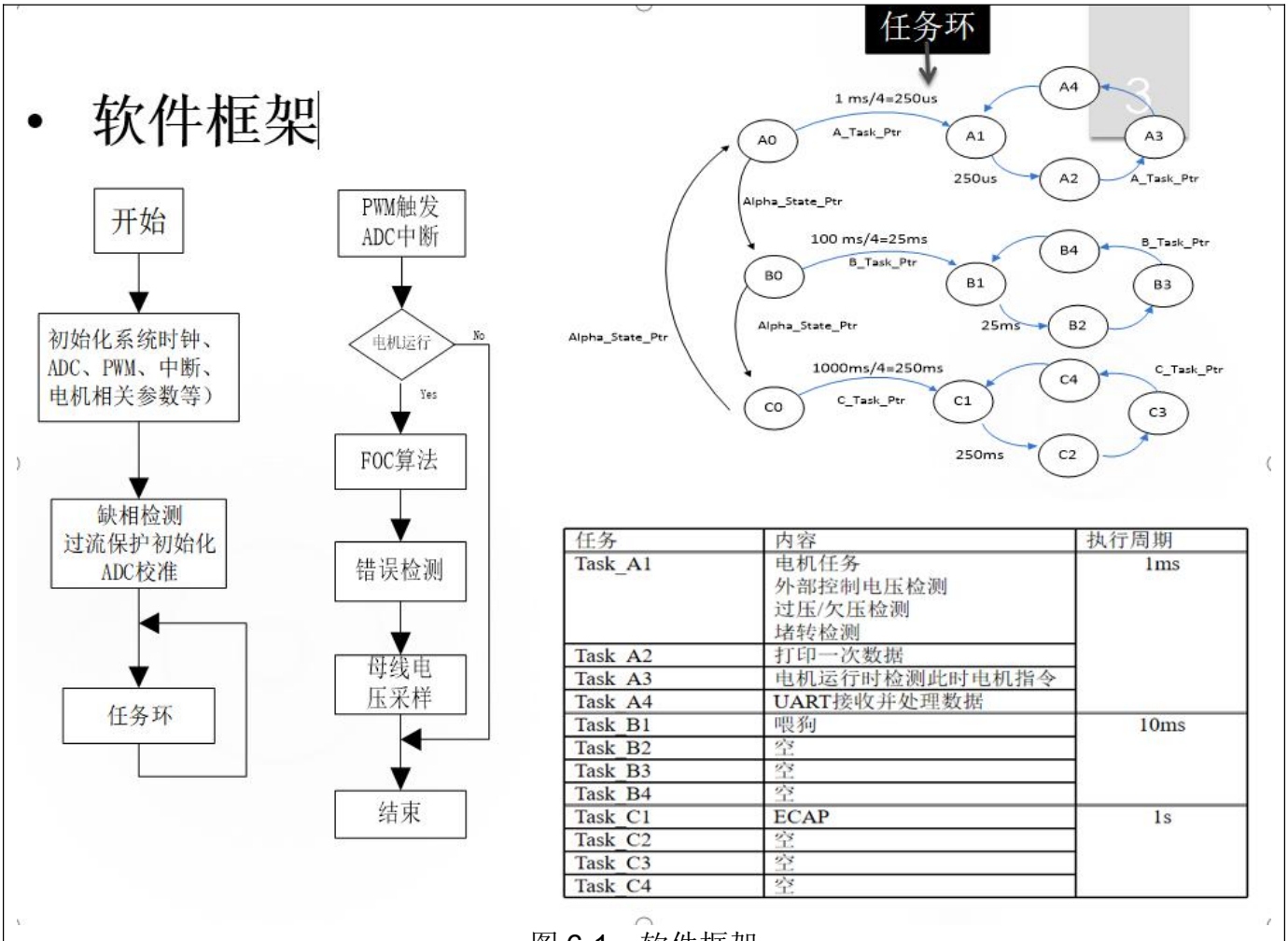
图 5-4.3 win10 查看蓝牙串口

蓝牙按照图 5-4.1，图 5-4.2，图 5-4.3 操作后，可以看到再串口上多出了两个串口设备，此时就可以按照 **ISP 串口工具的 4.5 章节**进行下载程序。

注意：选择哪个 com 口，此时可以先进行连接，如果蓝牙能够慢闪，证明蓝牙连接成功。

6. 软件框架

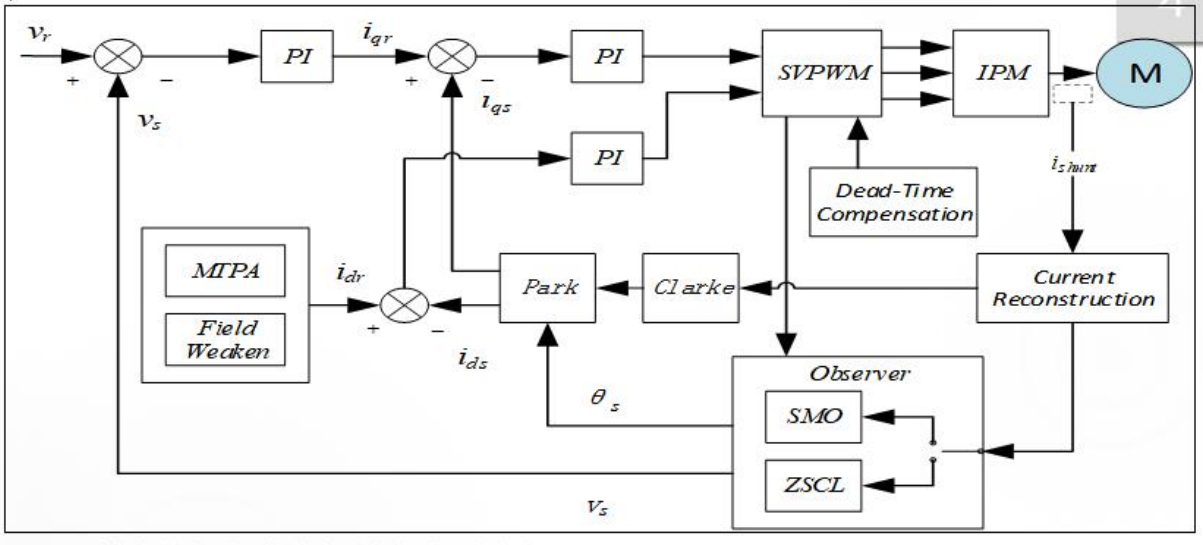
软件的算法框架大概如下框图。



任务	内容	执行周期
Task_A1	电机任务 外部控制电压检测 过压/欠压检测 堵转检测	1ms
Task_A2	打印一次数据	
Task_A3	电机运行时检测此时电机指令	
Task_A4	UART接收并处理数据	
Task_B1	喂狗	10ms
Task_B2	空	
Task_B3	空	
Task_B4	空	
Task_C1	ECAP	1s
Task_C2	空	
Task_C3	空	
Task_C4	空	

图 6-1　软件框架

算法框架



MTPA:最大转矩电流比控制(凸极电机)
Field Weaken:弱磁控制
Park: 将电机定子的a, b, c三相电流投影到随着转子旋转的直轴（d轴），
交轴（q轴）与垂直于dq平面的零轴（0轴）上去，
即abc坐标系变换到dq坐标系.
Clarke:将abc三相电流变换到两相静止 的α β坐标系下.
SVPWM: 空间矢量脉宽调制
LowSpeed:低速观测器,主要为启动时候估算角度，实现平稳平滑启动过程
SMO: 滑模观测控制，主要为高速运行时候估算角度，实现高性能的稳定驱动

图 6-2　FOC 算法框架

7. 数据采集功能

DATA_COLLECT_ENABLE	电机数据采集使能
DEBUG_BUFFER_RECORD_PERIOD	多少个PWM周期采集一次数据
TOTAL_RECOED_LENGTH	采集变量的数据总量
NUM_RECORD_VAR	采集变量的个数
void MotorDebugBuffer_Config (void)	采集变量初始化
MotorDebugBuffer_StartRecData(&gsDebugBuffer);	开始采集数据
MotorDebugBuffer_StartDumpData(&gsDebugBuffer);	开始打印数据

1. 首先使能DATA_COLLECT_ENABLE 为 1，
   填写需要采集的变量多少个NUM_RECORD_VAR，
   需要采集的数据TOTAL_RECOED_LENGTH多少，
   每次采集的间隔DEBUG_BUFFER_RECORD_PERIOD是多少
2. 然后再MotorDebugBuffer_Config函数里面初始化需要的所需要变量
   个数要与NUM_RECORD_VAR一致，请参考原有的写法。
3. 最后就可以使用MotorDebugBuffer_StartRecData(&gsDebugBuffer);进行采集数据。
   程序已经写有串口指令'r' 'R' 来进行采集数据，打印数据。
4. 把数据放到excel表格进行处理

图 7-2 数据采集功能

8. 电机参数

所在文件：motor_sys_config_basic.h

U8_MOTOR_POLES	电机极数，一般为2极（1对极）、4极（2对极）、6极（3对极）。。。
F32_MOTOR_REAL_PHASE_R_OHM	电机相电阻阻值，单位为欧姆
F32_MOTOR_REAL_PHASE_LD_H	电机D轴相电感，单位为亨利
F32_MOTOR_REAL_PHASE_LQ_H	电机Q轴相电感，单位为亨利
MOTOR_LOWSPEED_BEMF_FREQ_HZ	电机反电势的频率
MOTOR_LOWSPEED_BEMF_VL2L_VOLT	电机两个波峰之间的电压差
BEMF_COEF_AUTO	是否打开反电势系数自动计算（根据经验数据），1：使用；0：禁用。默认为0，当条件不允许的时候，可以打开这个选项。

图 8-1： 电机参数对应的宏

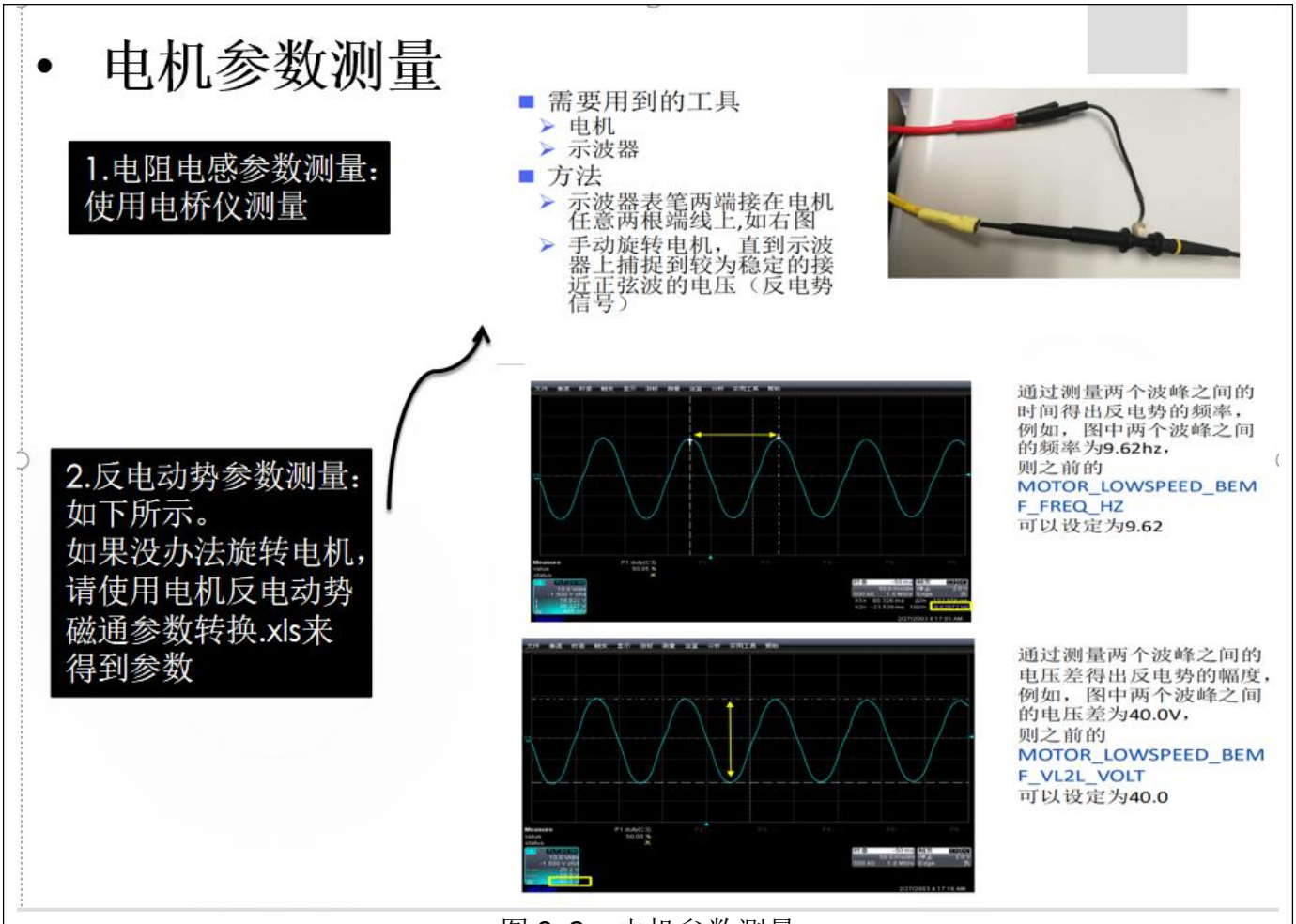


图 8-2： 电机参数测量

9. 电机保护功能

Firmware中包含多种保护的功能，需要用户根据需要打开或关闭

用户参数	描述
参数调试档案位置：motor_sys_config_basic.h	
STALL_DETECTION_EANBLE	是否启用堵转保护
OVER_CURRENT_ENABLE	是否启用过流保护
OVER_VOLTAGE_ENABLE	是否启用过压保护
UNDER_VOLTAGE_ENABLE	是否启用欠压保护
PHASE_LACK_DETECTION_ENABLE	是否启用缺相保护，其中默认开启的是运行过程中的缺相保护，启动前的缺相保护取决于是否有相电压检查电路

用户参数	描述
参数调试档案位置：motor_sys_config_basic.h	
F32_MOTOR_SYS_PHASE_OVERCURRENT_A	过流保护相电流大小，当任意相电流大于这个值，触发过流保护
F32_MOTOR_SYS_OVER_VOLTAGE_V	过压保护电压大小，当电压大于这个值，触发过压保护
F32_MOTOR_SYS_UNDER_VOLTAGE_V	欠压保护电压大小，当电压小于这个值，触欠过压保护

10. 开环电流采集测试

1) 首先需要了解第 7 章节的数据采集功能，然后开启开环 VDVQ 控制的相关宏，具体相关的宏参考图 10-1：电机电流验证的相关宏。
2) 采集到相应的结果后参考图 10-2：电机电流采集验证。

   正常的情况，需要看看从 5% - 30% - 70% - 95%的电压占空比下，电流采样波形是否连续，在这过程中，电流可能会非常大，可以通过降低母线电压，或者使用电阻较大的电机来做对应测试。
3) 如果电流采集正常，则可以通过
   MOTOR_RAMP_ENABLE 为 1，PURE_VOLTAGE_RAMP_ENABLE 设置为 0，
   进行开环电流环控制，通过电流增加或者减少指令操作（查询第 15 章节串口指令集），通过示波器观测电流是否增大减少。如果正常，则开环电流测试正常。

4) 注意调试完成后，启用的宏需要关闭。
   MOTOR_RAMP_ENABLE 与 PURE_VOLTAGE_RAMP_ENABLE 务必需要恢复为 0。

采集三相电流以及电压变量进行excel处理。
motor_sys_config.h

MOTOR_RAMP_ENABLE	开环使能 1：使能，0：关闭
F32_MOTOR_RAMP_UP_CURRENT_START_A	额定电流20%
F32_MOTOR_RAMP_UP_CURRENT_FINAL_A	额定电流20%
MOTOR_RAMP_UP_FORCE_TIME_MS	推荐5000ms
MOTOR_RAMP_UP_SPEED_START_RPM	0
MOTOR_RAMP_UP_SPEED_FINAL_RPM	额定转速10%
MOTOR_RAMP_UP_SPEED_TIME_MS	推荐5000ms
PURE_VOLTAGE_RAMP_ENABLE	开环电压控制使能，需要使能1
F32_MOTOR_RAMP_UP_VOLTAGE_DUTY_START	初始Duty1%，建议7%
F32_MOTOR_RAMP_UP_VOLTAGE_DUTY_FINAL	最终pwm Duty ，建议7%

图 10-1： 电机电流验证的相关宏

采集三相电流以及电压变量
进行excel处理。

1. 如果电机不能正常运行，
则需要适当调整如右图参数：

2. 采集的 电流电压变量：
myMotor[0].sCoreSignal.i16_I_U
myMotor[0].sCoreSignal.i16_I_V
myMotor[0].sCoreSignal.i16_I_W

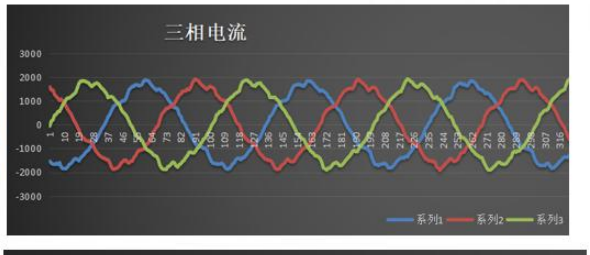
myMotor[0].sCoreSignal.i16_DutyU
myMotor[0].sCoreSignal.i16_DutyV
myMotor[0].sCoreSignal.i16_DutyW

3. 三相电流相位相差120度，
以及上下关于0轴对齐，
并且三相的电流与duty的方向是一致的
则电流采样正常

4. 如果不正常，则需要检查ADC配置

10

MOTOR_RAMP_UP_SPEED_FINAL_RPM
MOTOR_RAMP_UP_FORCE_TIME_MS
MOTOR_RAMP_UP_SPEED_TIME_MS
F32_MOTOR_RAMP_UP_VOLTAGE_DUTY_FINAL



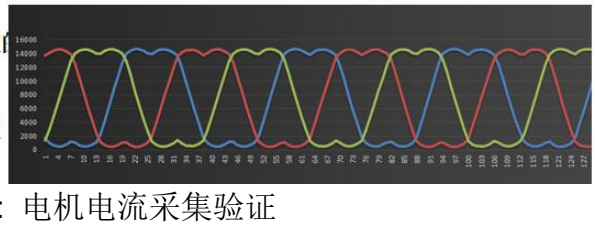


图 10-2： 电机电流采集验证

11. 开环算法适用验证

再第 10 章节验证电流正常后，可以采集如下两个变量，进行算法适用验证。算法不适用，可以联系 SPINTROL 的 fae,让其帮忙修改算法的**频率滤波参数**进行适配。

myMotor[0].sCoreSignal.i16_Esti_Theta	smo算法估算的角度
myMotor[0].sCoreSawGen.sOutput.i16Theta	开环实际的角度

观测得到，运行一段时间后，角度与smo估算的角度很逼近。如果不接近，请参考下文的smo调试。

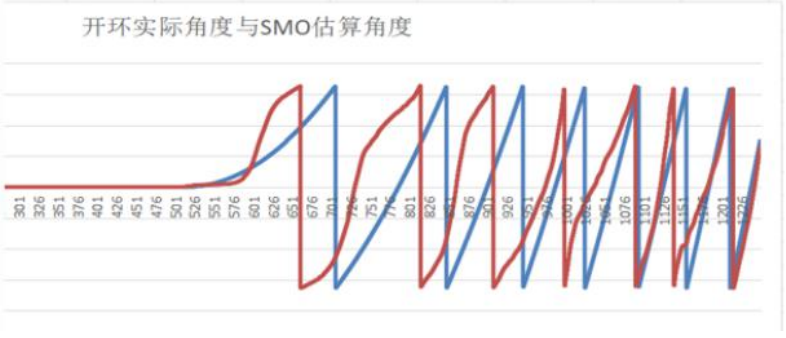


图 11-1： 电机算法对比验证

12. 闭环调试

闭环电流环测试正常后，现在开始外环调试。
外环包含speed，power，duty, voltage,这里使用速度环来调试参考。
相关宏配置如下：

MOTOR_RAMP_ENABLE	开环关闭使能0
PURE_VOLTAGE_RAMP_ENABLE	0
TORQUE_LOOP_ENABLE	0
SPEED_LOOP_ENABLE	1
POWER_LOOP_ENABLE	0
DUTY_LOOP_ENABLE	0
VOLTAGE_LOOP_ENABLE	0
CURRENT_LOOP_BANDWIDTH_HZ	电流环带宽，大小为载波的1/20，1/60之间

速度环PI参数: PI_CONTROL_SPEED_I_GAIN,
PI_CONTROL_SPEED_P_GAIN
DUTY环的PI参数: PI_CONTROL_DUTY_I_GAIN
PI_CONTROL_DUTY_P_GAIN
功率环的PI参数: PI_CONTROL_POWER_I_GAIN
PI_CONTROL_POWER_P_GAIN

图 12-1： 电机算法对比验证

myMotor[0].sThetaMerge.i32SpeedMergeRPM	融合后的速度
myMotor[0].sPI_Loop.i16_Cmd	给定速度
myMotor[0].sThetaMerge.u8MergeStatus	2：处于SMO与LOWSPEED融合阶段 1：进入smo算法高速运行 0：在lowspeed低速运行
myMotor[0].sFault.u32FaultState	运行出错的错误代码

通过调节PI参数，
确保采集到的数据如蓝色曲线
（融合速度）一样.

图 12-2： 电机闭环 PI 调试

13. 电机启动调试

如图 13-1： 电机启动参数配置，其中 fRs fLs fOneOverFlux 等为电机参数。Alpha0 和 Lambda 是观测其中的增益。所以电机参数确认正常的情况下，Alpha0 和 Lambda 保持默认的情况下启动参数就配置完了。

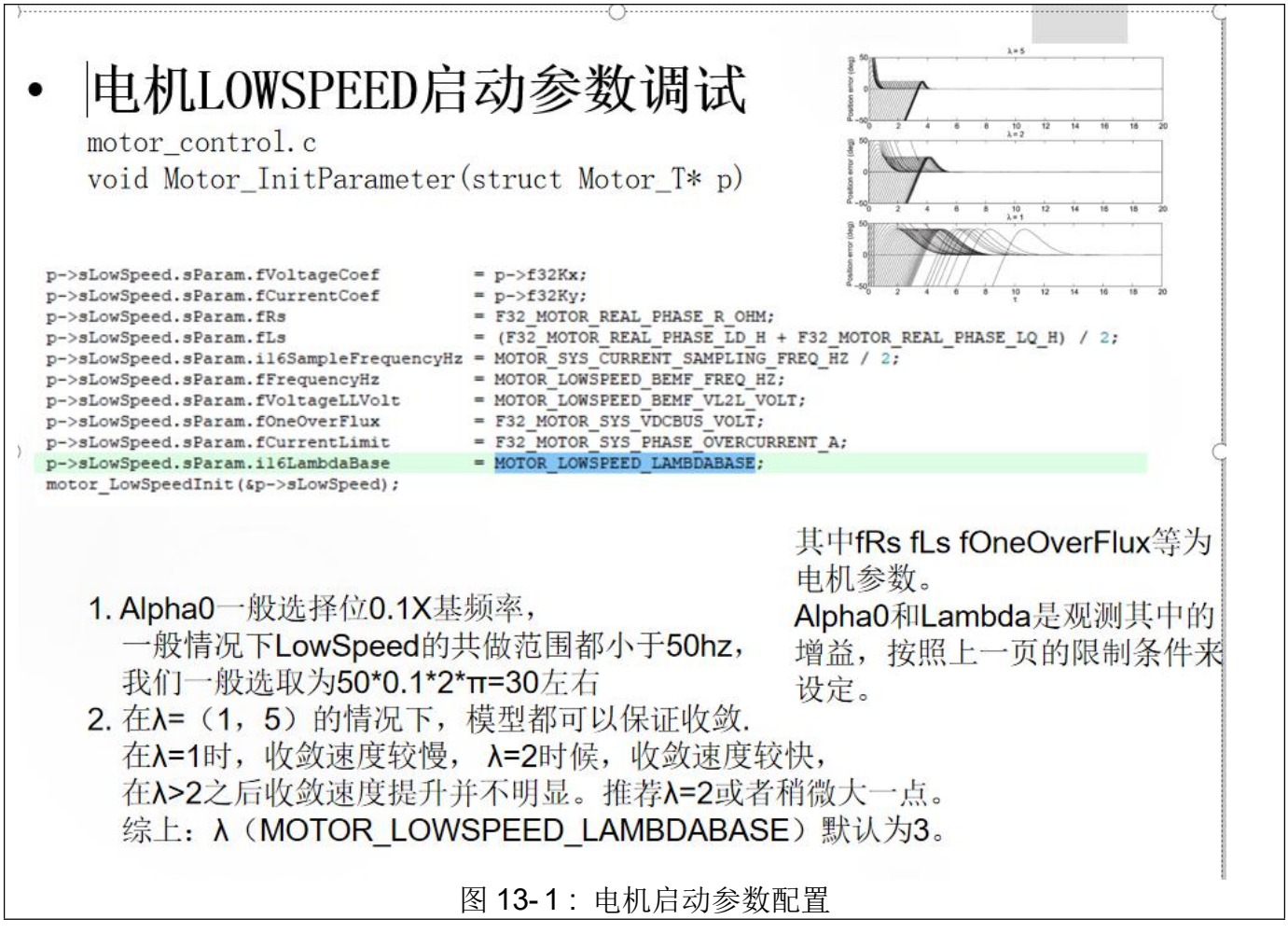


图 13-1： 电机启动参数配置

参数配置完后，进行启动调试。

配置完lowspeed的参数，下面进行闭环电流环调试。

MOTOR_RAMP_ENABLE	0
PURE_VOLTAGE_RAMP_ENABLE	0
TORQUE_LOOP_ENABLE	1
SPEED_LOOP_ENABLE	0
POWER_LOOP_ENABLE	0
DUTY_LOOP_ENABLE	0
VOLTAGE_LOOP_ENABLE	0
CURRENT_LOOP_BANDWIDTH_HZ	电流环带宽，大小为载波的1/20，1/60之间

串口指令给定相应的电流（‘[’,’]’,’o’,’p’），然后给定电机启动指令‘t’，进行启动测试。如果启动不成功，则尝试打开平滑启动。

如果启动还是各种不顺，可以按修改下面两个阀值。
myMotor[0].sThetaMerge.u16ToSMOThreshold （HZ）
myMotor[0].sThetaMerge.u16ToLowThreshold （HZ）
u16ToSMOThreshold的值可以修改为额定转速频率的20%附近。

图 13-2： 电机启动调试

如果启动还不是很顺畅，可以进行再调节平滑电流:

MOTOR_STANDSTILL_SMOOTH_ENABLE	是否打开平滑启动，打开：1，关闭：0
STANDSTILL_SMOOTH_MAX_CURRENT_A	平滑启动电流
STANDSTILL_SMOOTH_DONE_SPEED_THRESHOLD_RPM	平滑启动结束的速度阈值

用电流钳观察相电流，把这个值从小到大调节，直到保证每次都可能快速启动。这个值不要超过额定时候的最大电流。

图 13-3： 电机平滑启动调试

14. 串口指令集

串口命令	描述
t	电机启动
+ p	增加给定(电流/速度/功率)： +0.1A/+100RPM/+1.0w
- o	减小给定(电流/速度/功率)： -0.1A/-100RPM/-1.0w
[	增加给定(电流/速度/功率)： +0.01A/+10RPM/+0.1w
]	减小给定(电流/速度/功率)： -0.01A/-10RPM/-0.1w
8	增加给定(电流/速度/功率)： +0.001A/+1RPM/+0.01w
9	减小给定(电流/速度/功率)： -0.001A/-1RPM/-0.01w
s	电机停止
r	打印记录数据
q	打印运行相关状态数据

15. MemDump 工具使用

MemDump 工具是用于把烧入芯片的固件读取出来，然后与现有的固件进行 CRC 判断，从而可以判断固件是否烧写正确或者完整。

15.1. 相关工具

硬件	usb 转串口工具 / MiniSPLink 工具 / SPC1068 开发板（目标板子）
软件	MemDump.exe
MiniSPLink 的固件	MiniSPLink_MemDump.hex
测试固件	测试的 hex 文件

表 15-1： 相关工具

15.2. 更新 MiniSPLink 的固件

需要把 MiniSPLink_MemDump.hex 下载进到 MiniSPLink 工具。
接线如下:

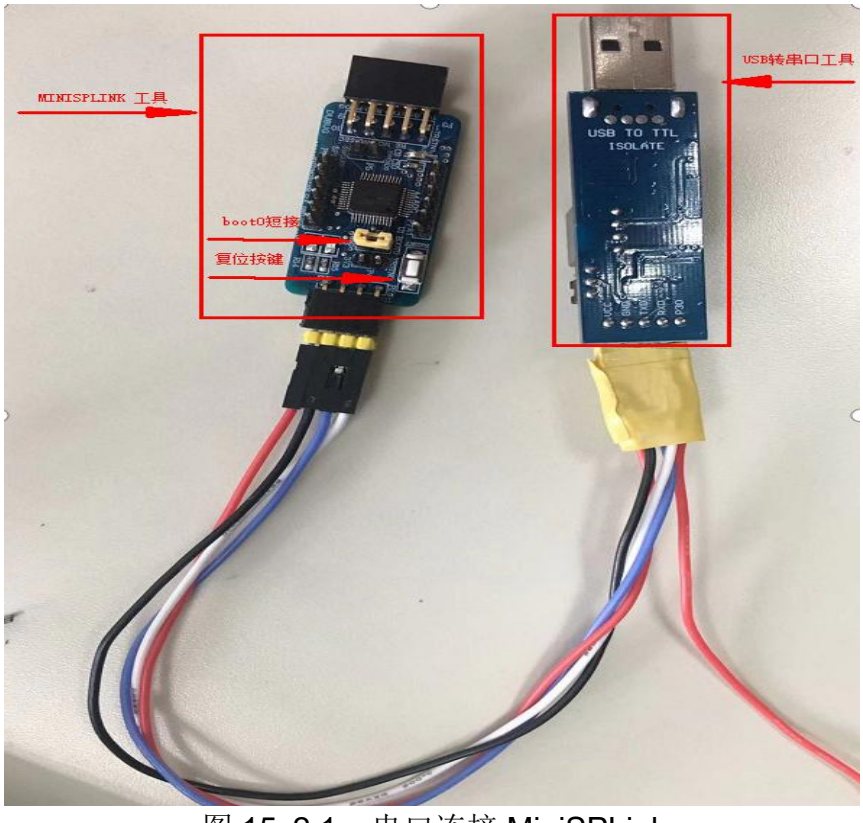


图 15-2.1： 串口连接 MiniSPLink

接线完成，usb 转串口接入电脑， 使用 ISP 工具进行下载，选择相应的串口，波特率使用 38400。
注意：MiniSPLink 的芯片为 1068，需要把 boot0 拉低，才能进行下载，所以需要跳冒进行短接才能进行下载.

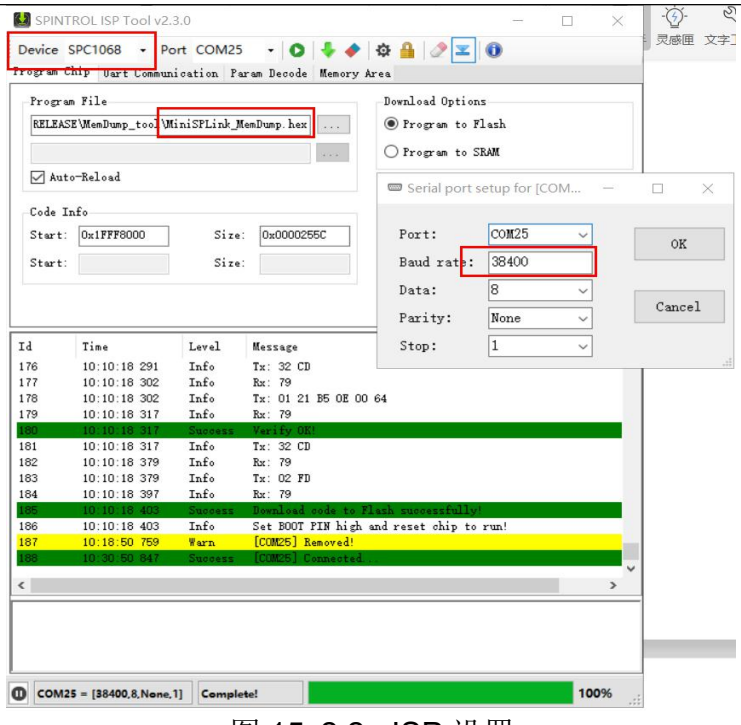


图 15- 2.2 : ISP 设置

15.3. 接入开发板

1) 接线如下

一端接 USB 转串口，一段接口接板子 SWD。
接完后按一下 MinSPLink 的复位按键，绿灯常亮表示连接正常。
注意：**MiniSPLink 的 boot0 不能短接。**
如图：

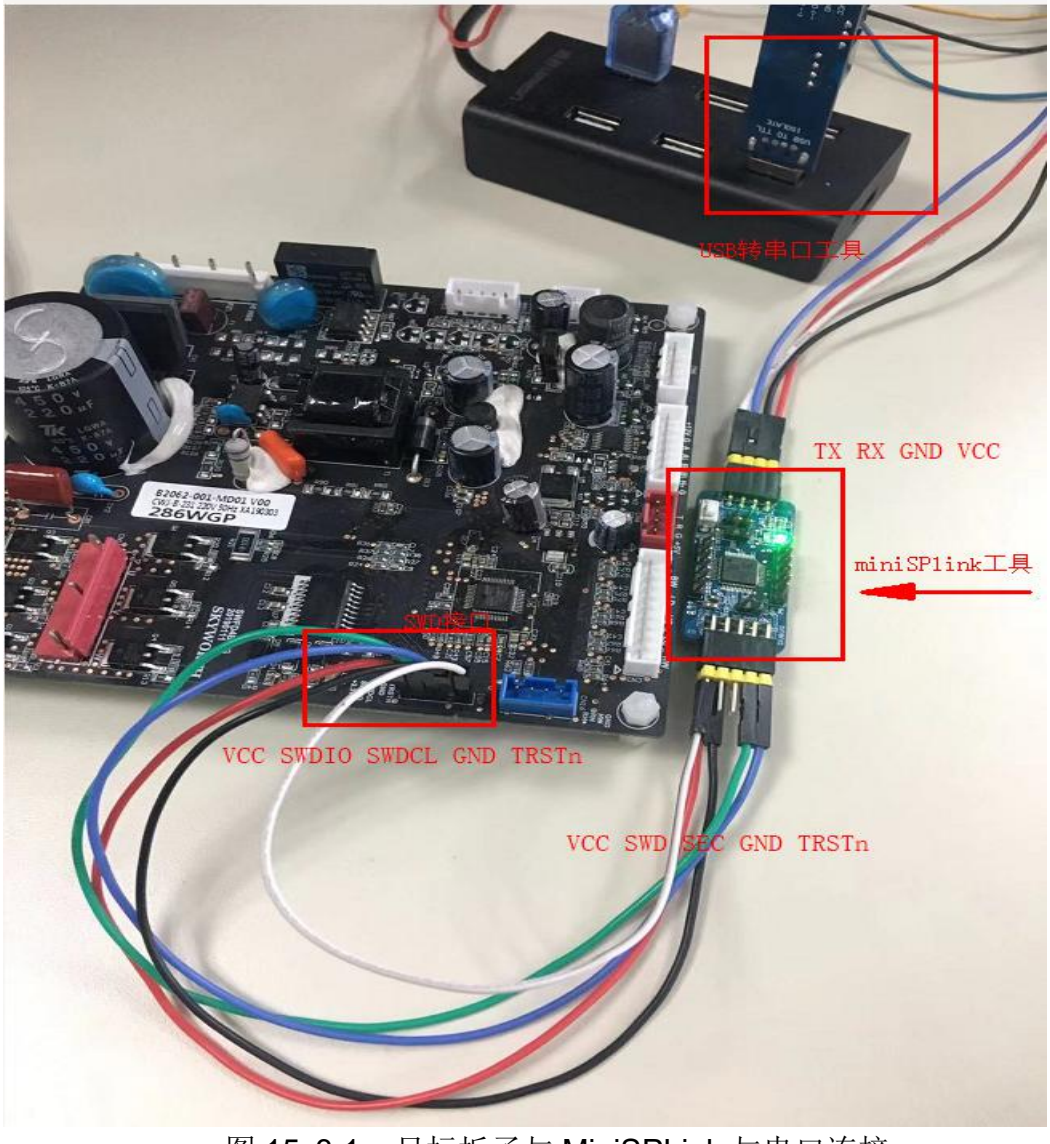


图 15-3.1： 目标板子与 MiniSPLink 与串口连接

2) memdump 配置

打开 memdump 软件，选择相应的串口，波特率选择 115200，选择相应的芯片，以及设置相应的地址。

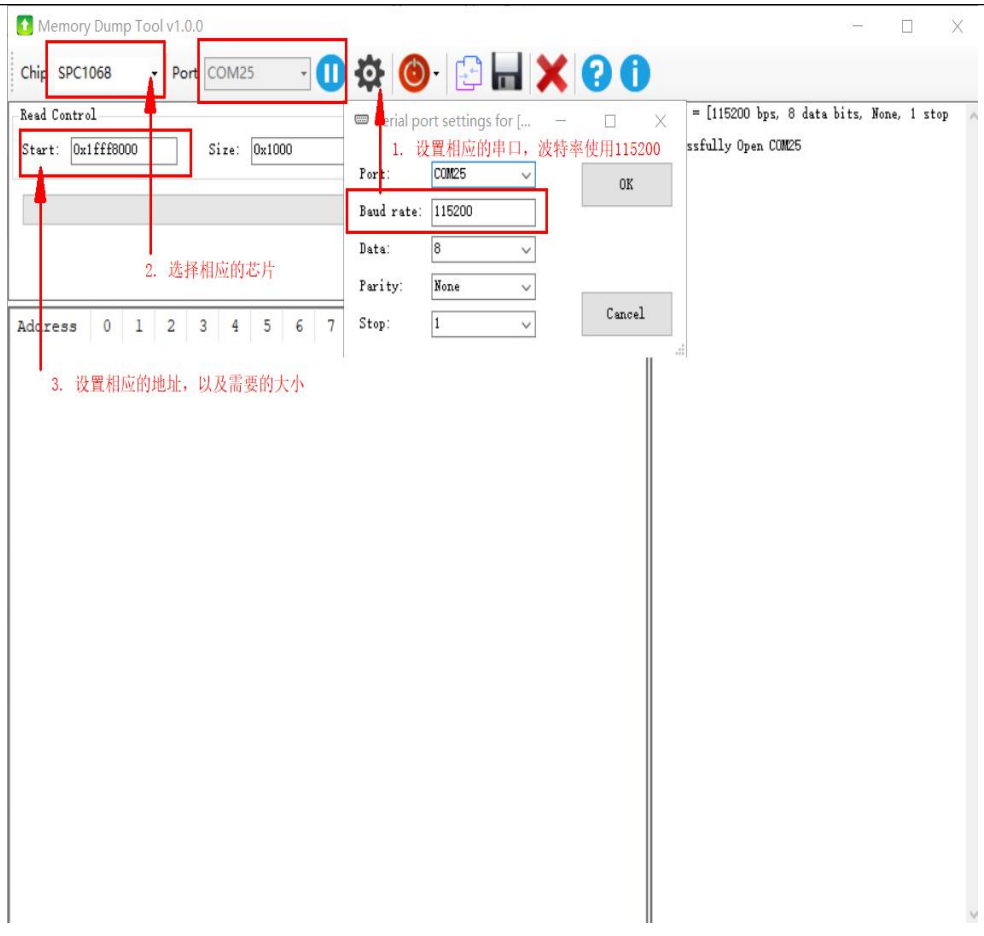


图 15-3.2 : Memdump 工具设置

3) 连接串口

配置完串口后，选择连接串口.

4) 复位 MCU

如图点击红色的按钮，在然后选择 Reset Target MCU.,此步很重要。

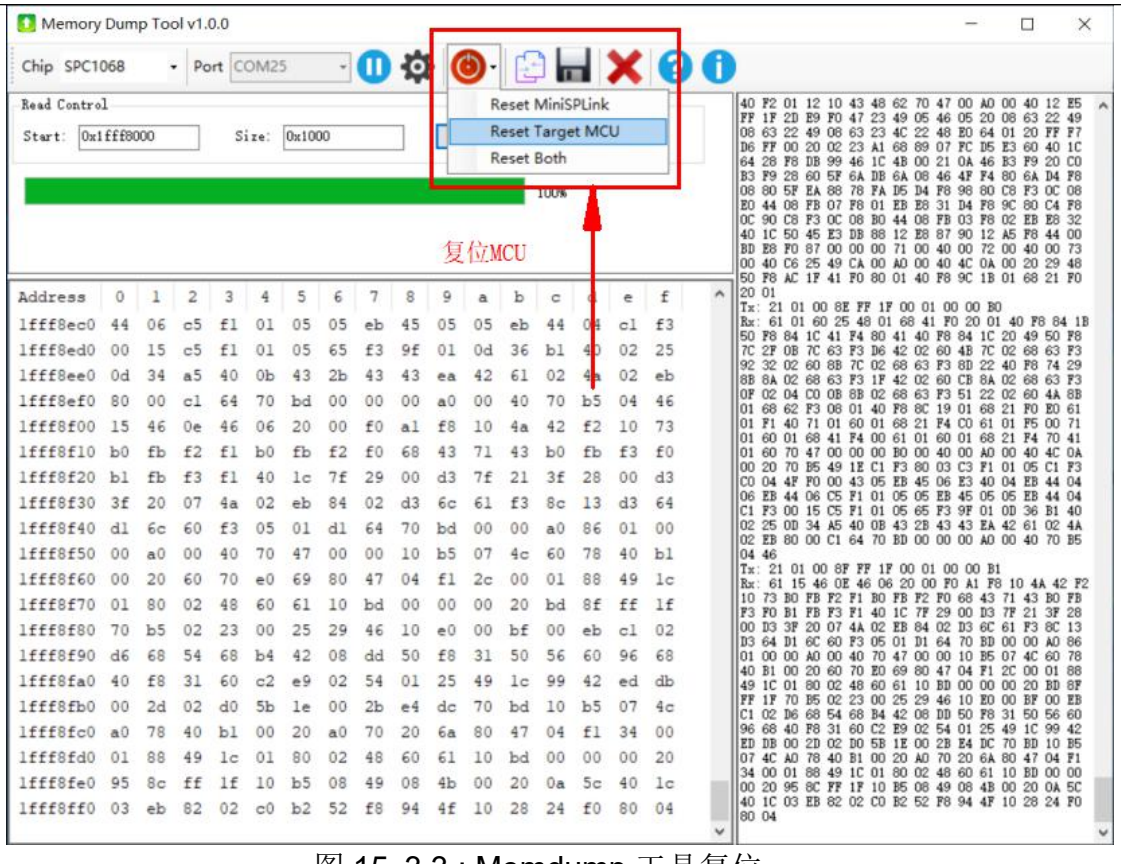


图 15-3.3 : Memdump 工具复位

5) 点击 read，数据读取完毕。

注意：如果 read 不成功，接线也正常的话，可能的原因为电压不够，需要外接 3.3V 电压给 MiniSPLink 小板子。

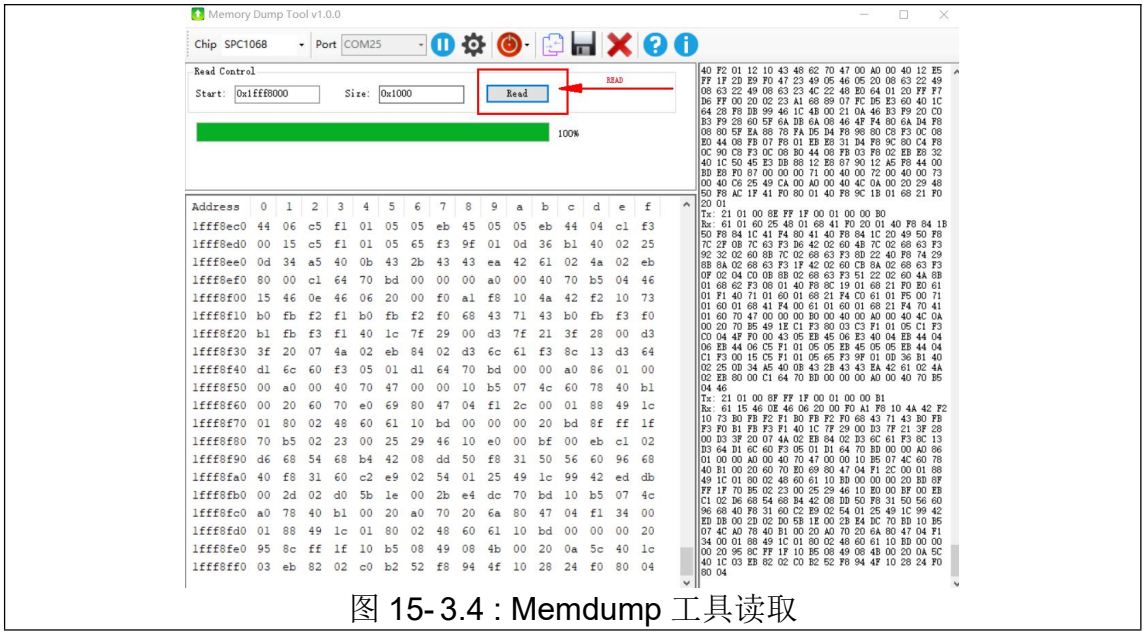


图 15- 3.4 : Memdump 工具读取

6) 保存 hex 文件

选择保存，保存后使用 **beyong compare** 等对比工具进行对比。

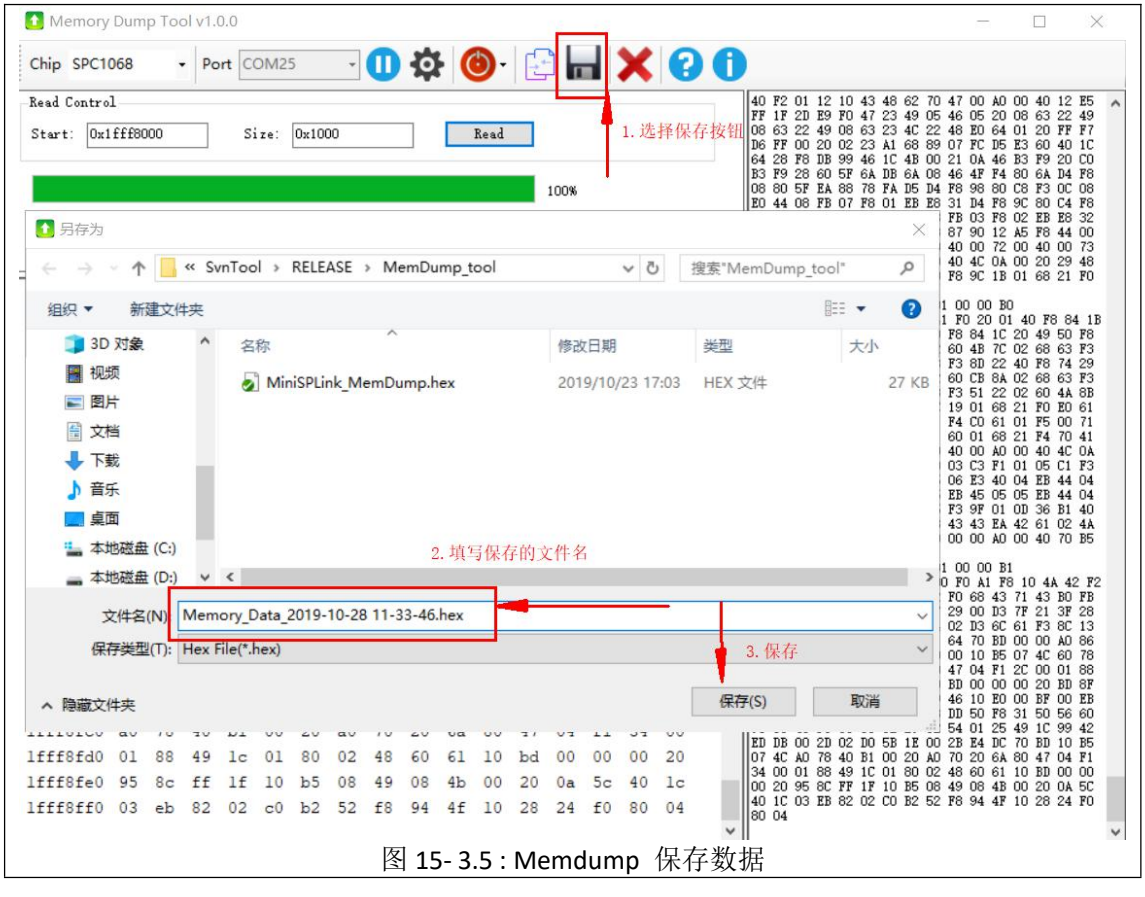


图 15- 3.5 : Memdump 保存数据

16. 持续更新中