*Technical Article*

# *上手 AWR2944 在线 CCS 调试*

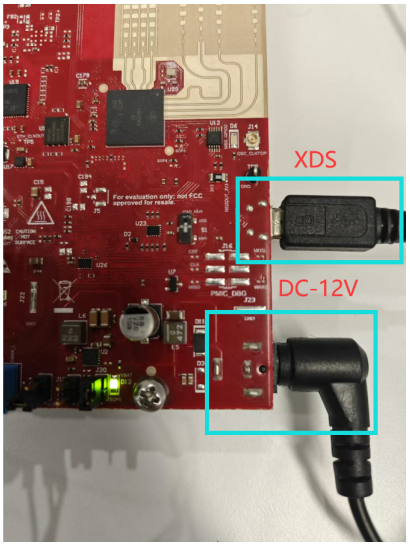
本文内容基于 AWR2944 EVM，以 mmw Demo 为例，介绍如何上手 AWR2944 的在线调试。

## 一、软硬件准备

### 1. 硬件准备

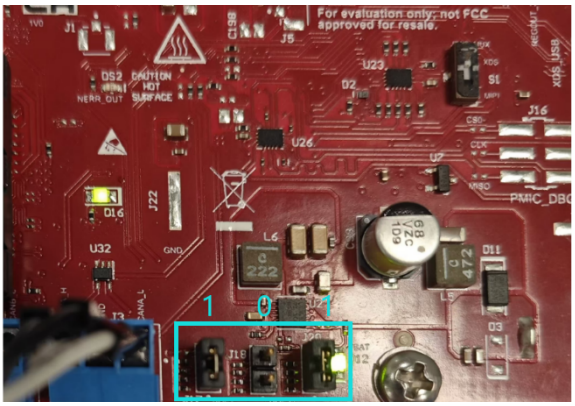
**EVM 板: XWR2944EVM REV.D** (https://www.ti.com/tool/AWR2944EVM)

如下图所示，使用 DC-12V 直流电源(电流容量>2.5A)，连接 EVM 板为整板供电，使用 Micro USB 线连接 EVM 与主机的 USB 口。




将 EVM 的 SOP 模式设置成 101(Flashing mode)，如下图所示。按下 Reset 按键或者 EVM 重新上下电进行重置。

关于 SOP 模式更多的介绍请参考附录 1.SOP 模式。




### 2. 软件准备

**SDK 版本 MMWAVE-MCUPLUS-SDK Version : 4.6.1.2**

**雷达工具箱版本 Radar_Toolbox Version : 2_20_00_05** https://www.ti.com.cn/tool/cn/download/RADAR-TOOLBOX/2.20.00.05

查看文件 C:\ti\mmwave_mcuplus_sdk_04_06_01_02\mcu_plus_sdk_awr294x_09_02_00_15\README_FIRST_AWR294X.html 的 Introduction 下的 Directory Structure 确定软件版本。

**注意，SDK 版本可能不同请根据自身实际安装的 SDK 版本进入对应路径。**

根据 Directory Structure 小节提供的信息，检查软件环境是否安装成功，软件版本需要匹配。

(注：README_FIRST_AWR294X 提供的 CCS 版本为 1270，本文使用 CCS1271，经验证 CCS1271 在本文的操作中能够产生与 CCS1270 一样的结果。)

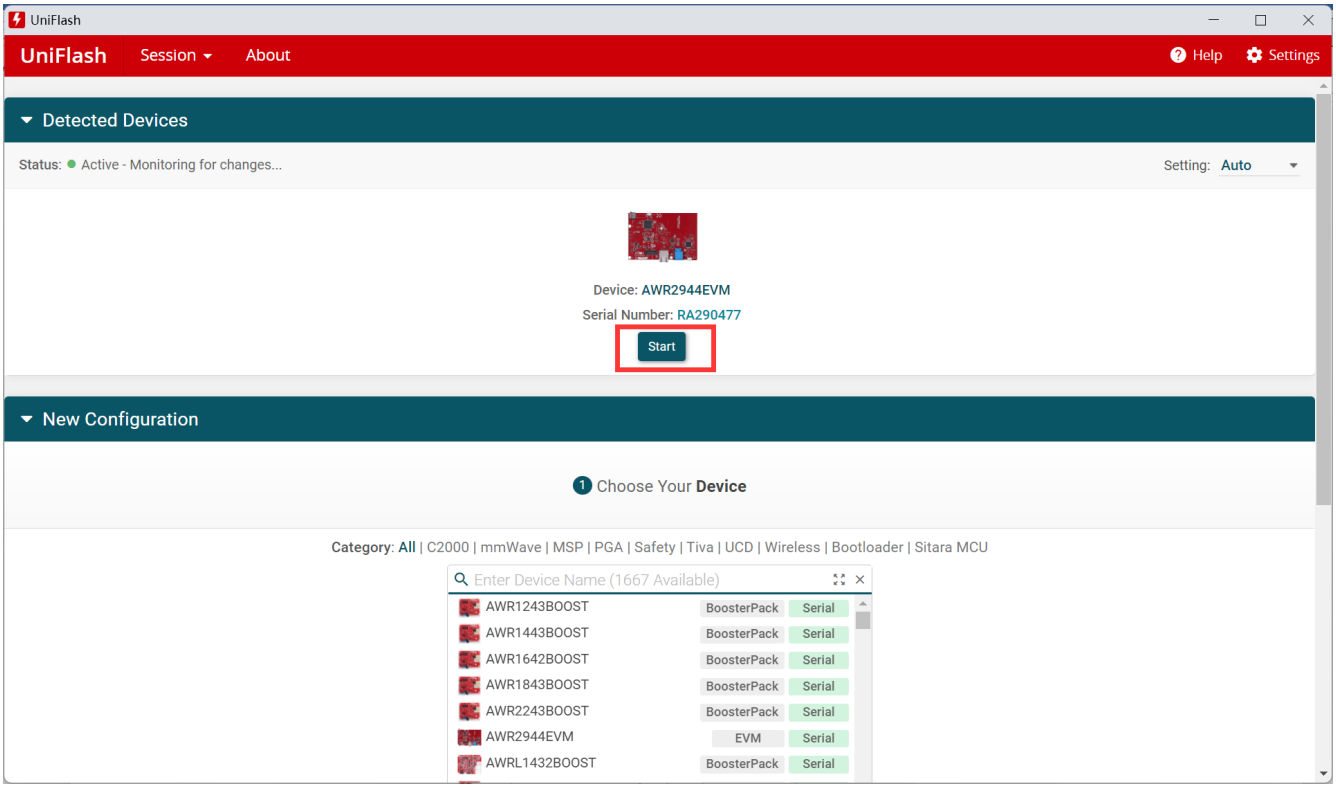
| 文件目录 | 描述 |
|---|---|
| C:/ccs1271 | Code composer studio |
| C:/ti/sysconfig_1.20.0 | SysConfig. NOTE: SysConfig is also installed as part of CCS at $(CCS_INSTALL_PATH)/ccs/utils/sysconfig_x_x_x |
| C:/ti/ti-armllvm_3.2.2.LTS | TI ARM CLANG compiler tool chain |
| C:/ccs1271/ccs/tools/compiler/ti-cgt-6000_8.3.12 | Code composer studio |

## 二、使用 CCS 工程进行在线调试

### 1. 烧录 SBL 和程序镜像

SBL(Secondary Boot Loader, 次级引导程序加载器)通常负责初始化硬件组件、内存以及其他系统资源。只有在这些资源和硬件初始化完成后，应用代码才能正常运行和调试。SBL 程序正确执行并加载应用程序（*awr2944_ccsdebug.appimage*），SBL 程序运行结束后将进入应用程序，应用程序会进行循环等待，确保各个核处于一个确认的状态，使调试器能顺利连接上各个核。更多关于 *awr2944_ccsdebug.appimage* 的介绍请参考附录 4.awr2944_ccsdebug.appimage。用户可以使用 UniFlash 将镜像文件烧录至 AWR2944 EVM 的 Flash 中。具体步骤如下：

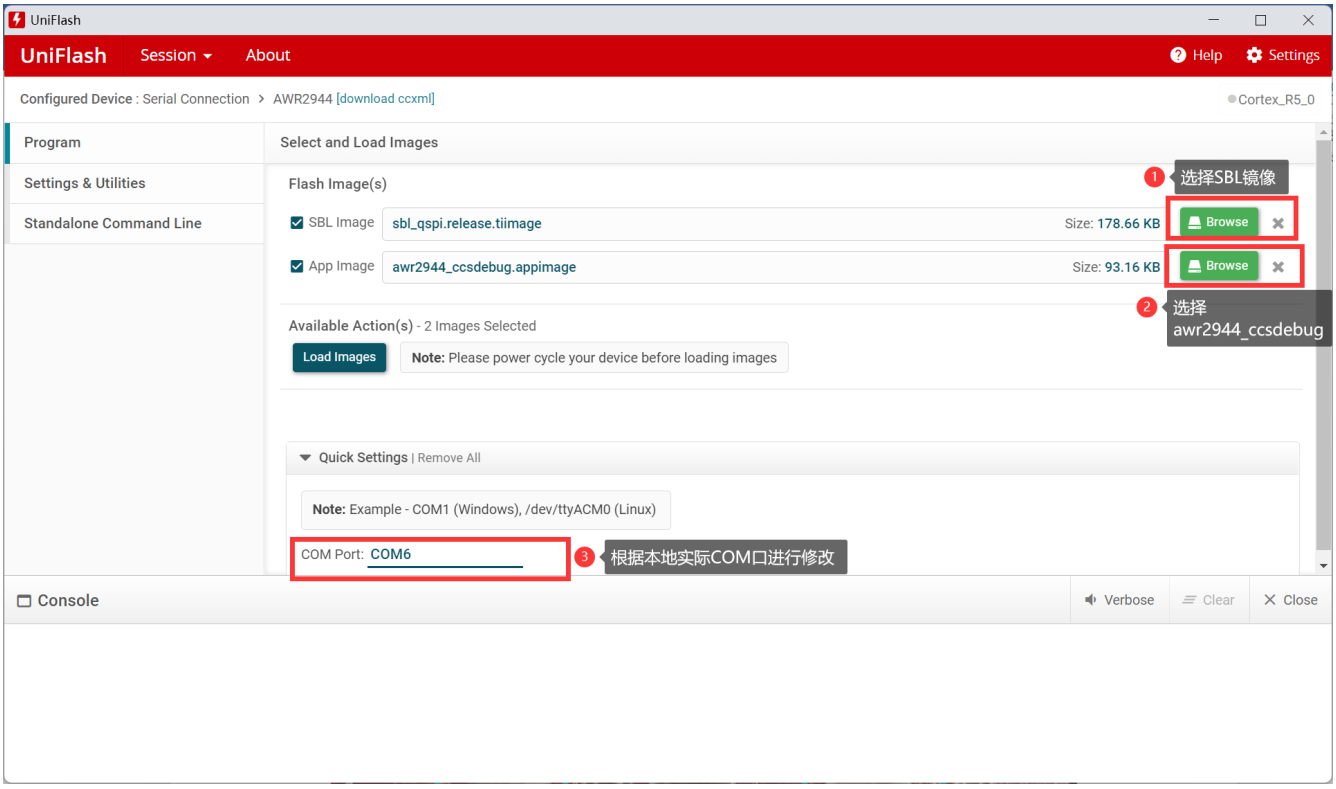
1. 打开 UniFlash 界面入下图所示。点击 Start 按钮后，进入烧写界面，也可以通过选择 device（AWR2944）进入烧写界面。



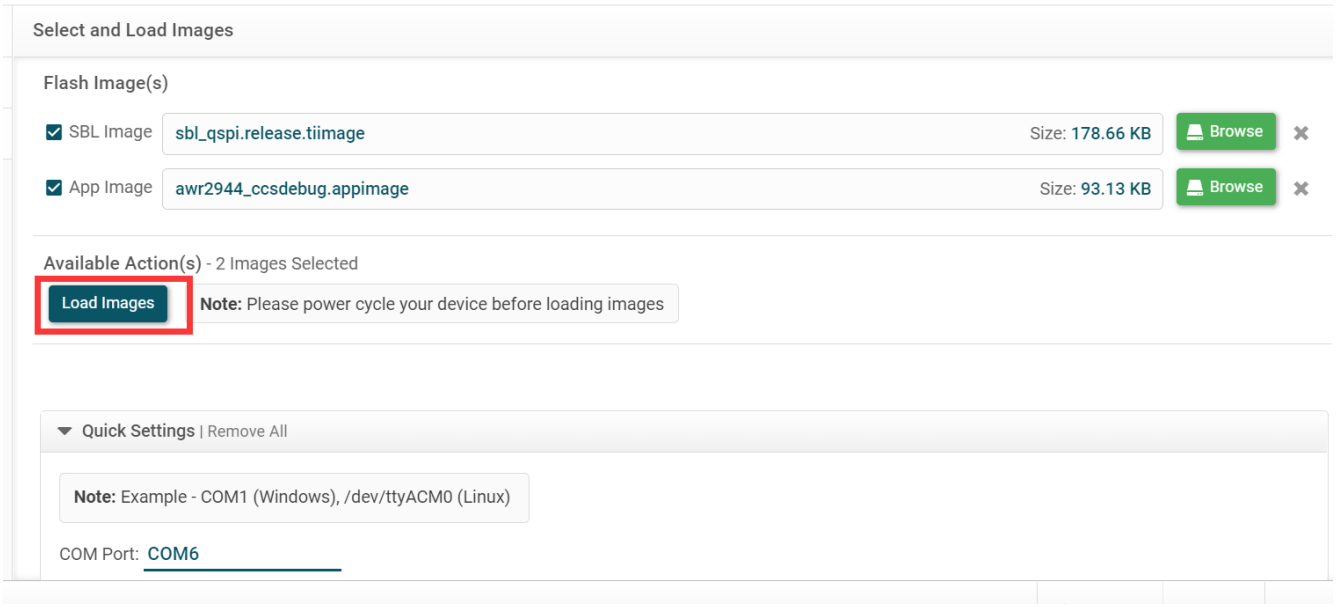

2. 输入烧写文件和配置串口点击 SBL Image 侧栏的 Browse，选择 C:\ti\mmwave_mcuplus_sdk_04_06_01_02\mmwave_mcuplus_sdk_04_06_01_02\tools\awr294x\sbl_qspi.release.tiimage，如下图①所示。

   点击 APP Image 侧栏的 Browse，选择 C:\ti\mmwave_mcuplus_sdk_04_06_01_02\mmwave_mcuplus_sdk_04_06_01_02\ti\utils\ccsdebug\awr2944_ccsdebug.appimage，如下图②所示

   在电脑的资源管理器里查看 AWR2944EVM 的 COM 口（User UART），输入正确的 COM Port，如下图③所示。



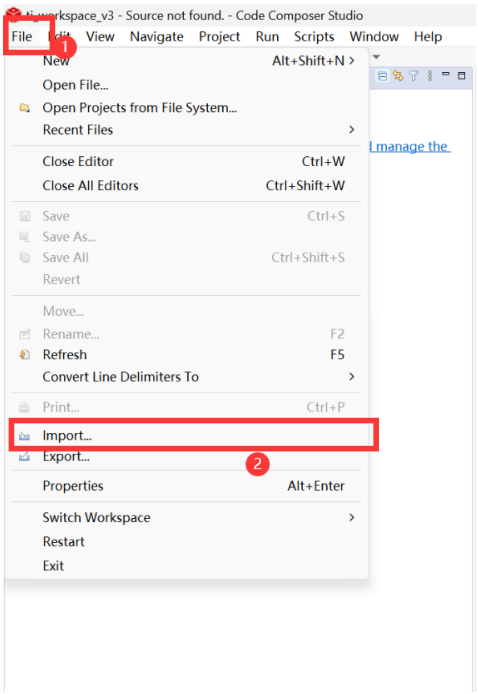

3. 烧写 Flash，点击 Load Image 将镜像文件烧写至 Flash。



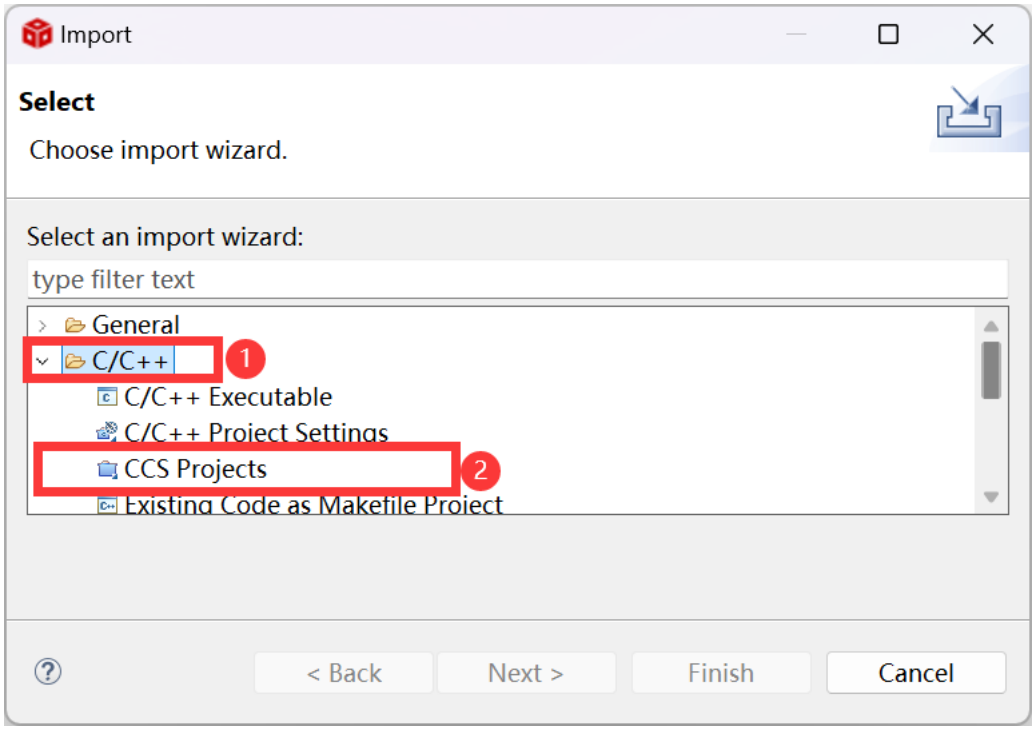

UniFlash 的 Console 出现[SUCCESS] Program Load completed successfully 表示烧录成功。

**2. 将工程文件导入 CCS 并编译**

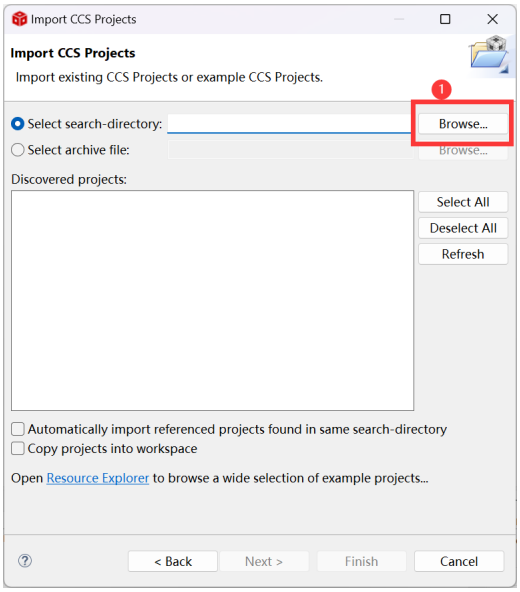
1. 打开 CCS，点击左上角 File->Import



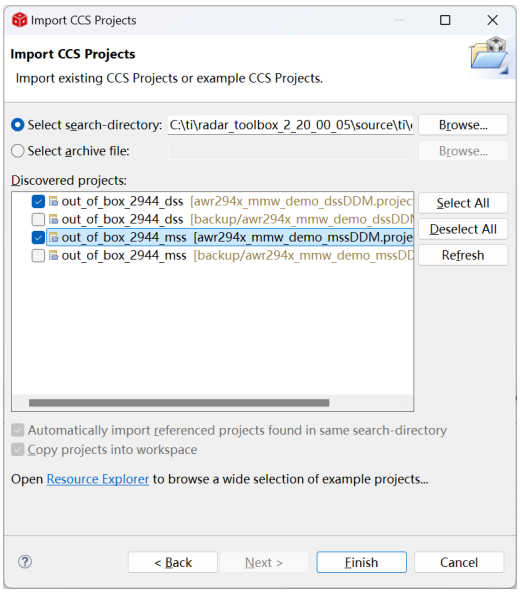

2. 展开 C/C++ Folder，双击 CCS Project



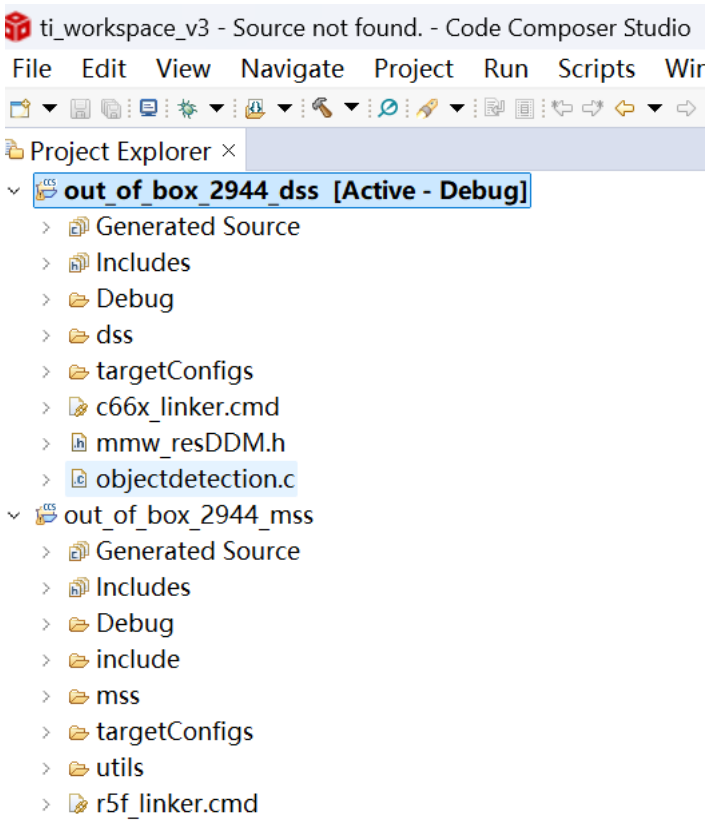

3. 点击 Browser



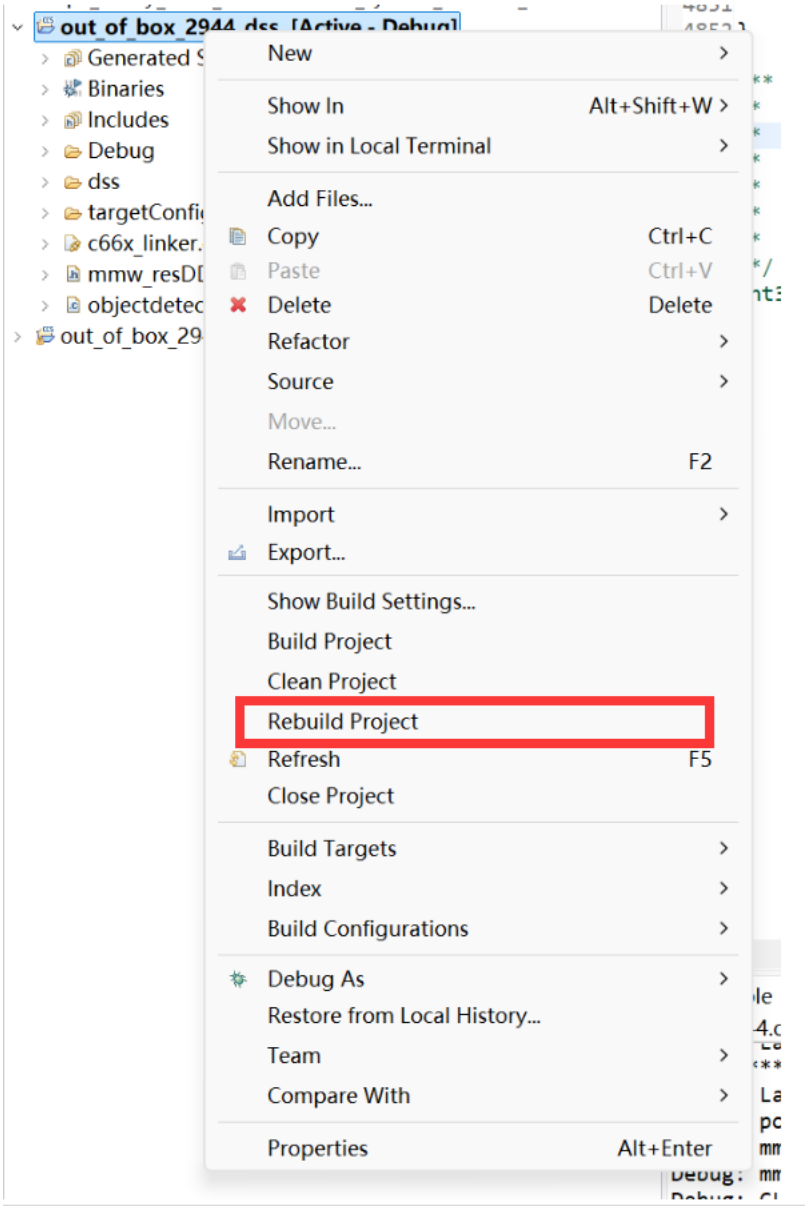

从路径 C:\ti\radar_toolbox_2_20_00_05\source\ti\examples\Out_Of_Box_Demo\src\awr294x 打开相应的文件夹，选择相应的工程文件。




4. 导入工程后，可以在 CCS 看到 DSS 和 MSS 两个工程，如下图所示。

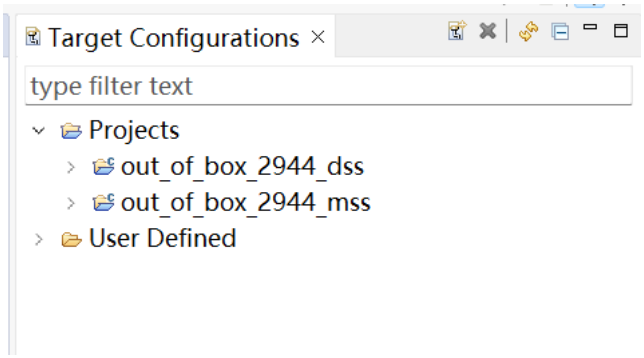



5. 先对 dss 工程进行 ReBuild Project，再对 mss 工程进行 ReBuild Project。

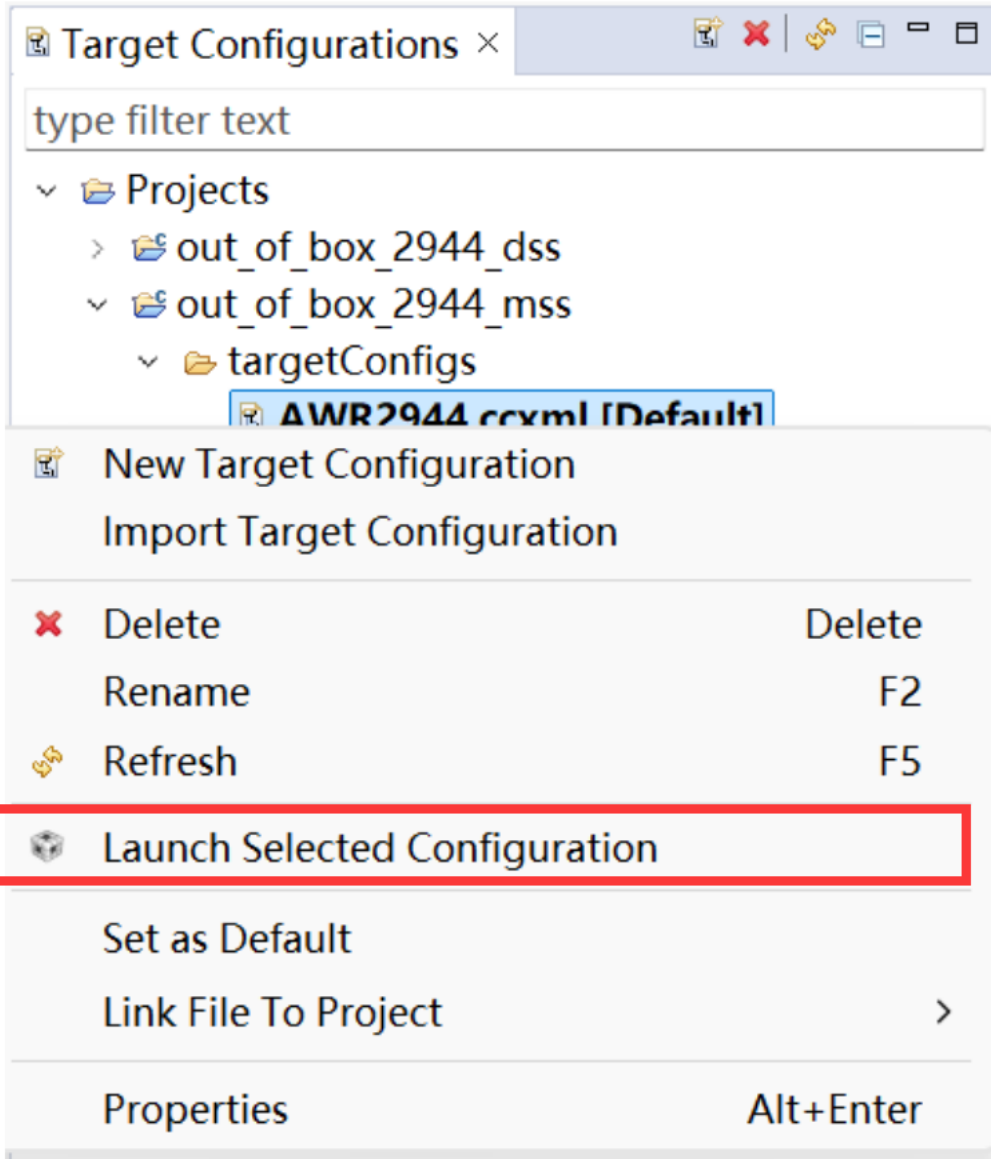



### 3. 连接 XDS110 调试器

1. 找到右侧栏的 Target Configurations。如果不存在 Target Configurations，请到上侧菜单栏点击 View -> Target Configurations。

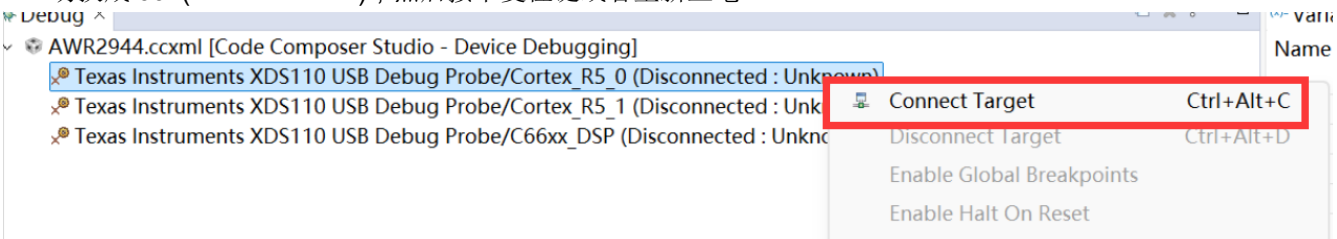


点击 out_of_box_2944_mss/targetConfigs，选择 AWR2944.ccxml，右键，选择 Launch Selected Configuration。

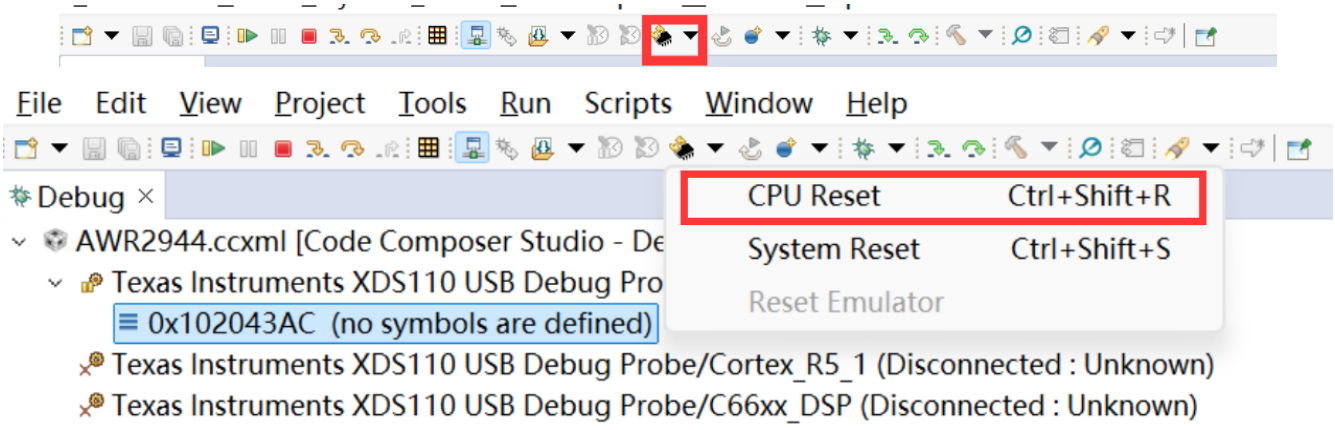



其中 ccxml 文件是一个目标配置文件(Target Configuration File)，它定义了调试会话所需的各种设置用以简化调试配置过程和提高灵活性。目标配置文件已经支持了大多数 TI 的器件，在使用目标配置文件时只需要选择正确的设备以及采用的调试工具即可，其他需要配置的参数将自动生成。更多关于 ccxml 的解释请参考附录 3.目标配置文件。

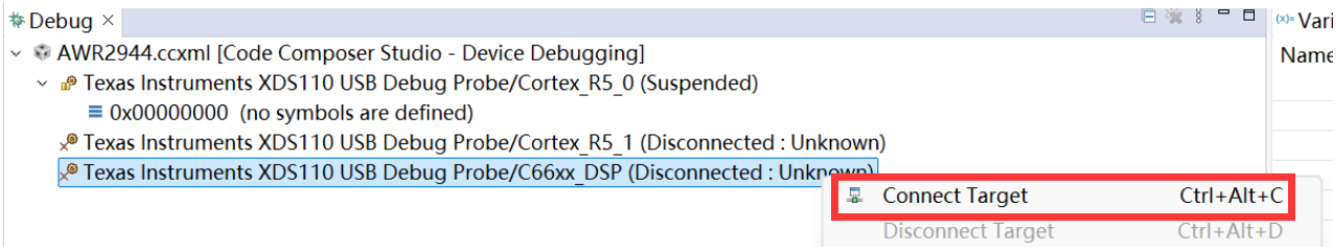
2. 选择 Cortex_R5_0，右键，点击 Connect Target 连接 R5_0 核。在连接核之前，请确认将 EVM 的 SOP 模式切换成 001(function mode)，然后按下复位键或者重新上电。




点击 CPU Reset 的下拉栏，选择 CPU Reset



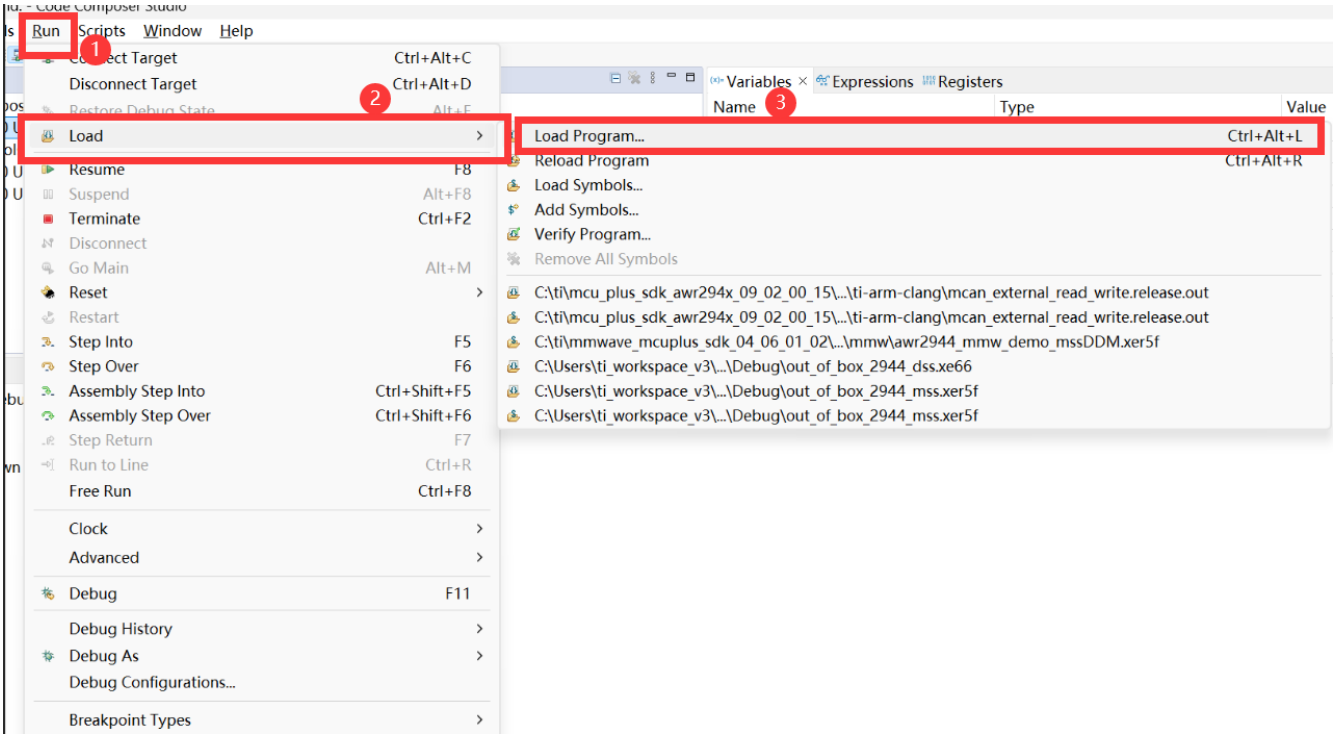

3. 选择 C66xx_DSP，右键，选择 Connect Target



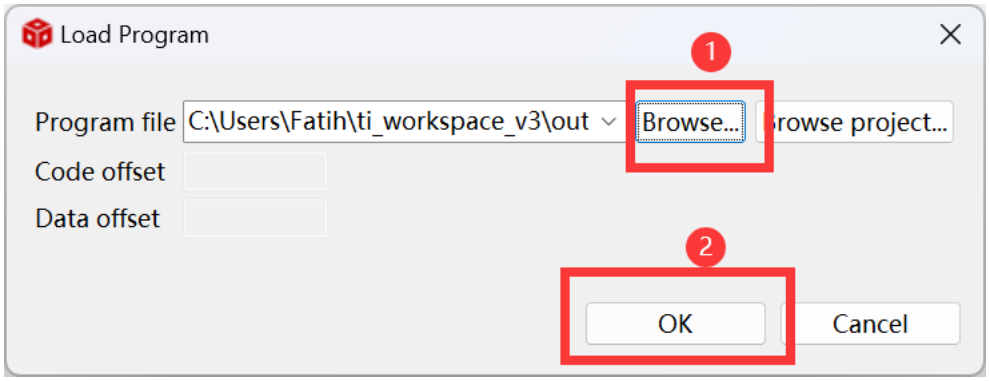

选择 CPU Reset 对 C66xx 核进行 reset。

**4. 加载应用程序到 AWR2944 RAM**

1. 先点击选中 R5_0 核，然后选择菜单栏的 Run->Load->Load Program。

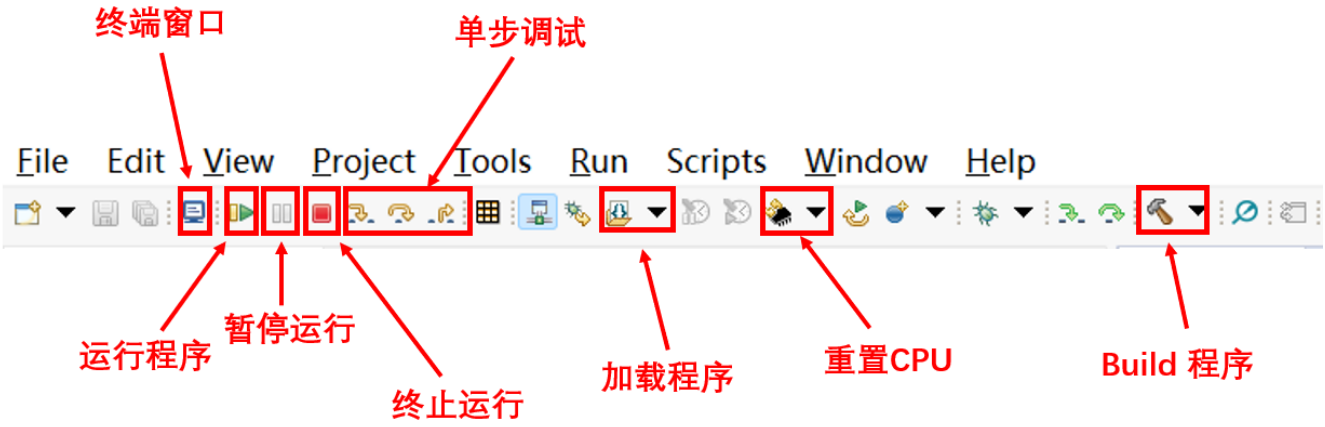



2. 点击 Browse，选择文件 R5F 的可执行文件，例如 C:\Users\Username\workspace_v12\out_of_box_2944_mss\Debug\ out_of_box_2944_mss.xer5f，点击 OK。代码会自动运行到 main 函数。




3. 同理，点击选中 C66xx 核，选择菜单栏的 Run->Load->Load Program

   点击 Browse，选择文件 C:\Users\Username\workspace_v12\out_of_box_2944_dss\Debug\out_of_box_2944_dss.xe66，点击 OK。
4. 然后就可以通过菜单栏对代码进行调试操作，菜单栏基本的调试功能组件如下图所示。




## 三、使用 makefile 编译的代码进行在线调试

除了将程序工程导入 CCS 集成开发环境下进行编译，SDK 的程序还可以使用 Makefile 进行编译，本节以 mmwave_mcuplus_sdk_04_06_01_02 的 mmw demo 为例，介绍使用 Makefile 进行代码编译后如何使用 CCS 集成开发环境进行在线调试。

**1. 使用 Makefile 对 mmw demo 进行编译。**

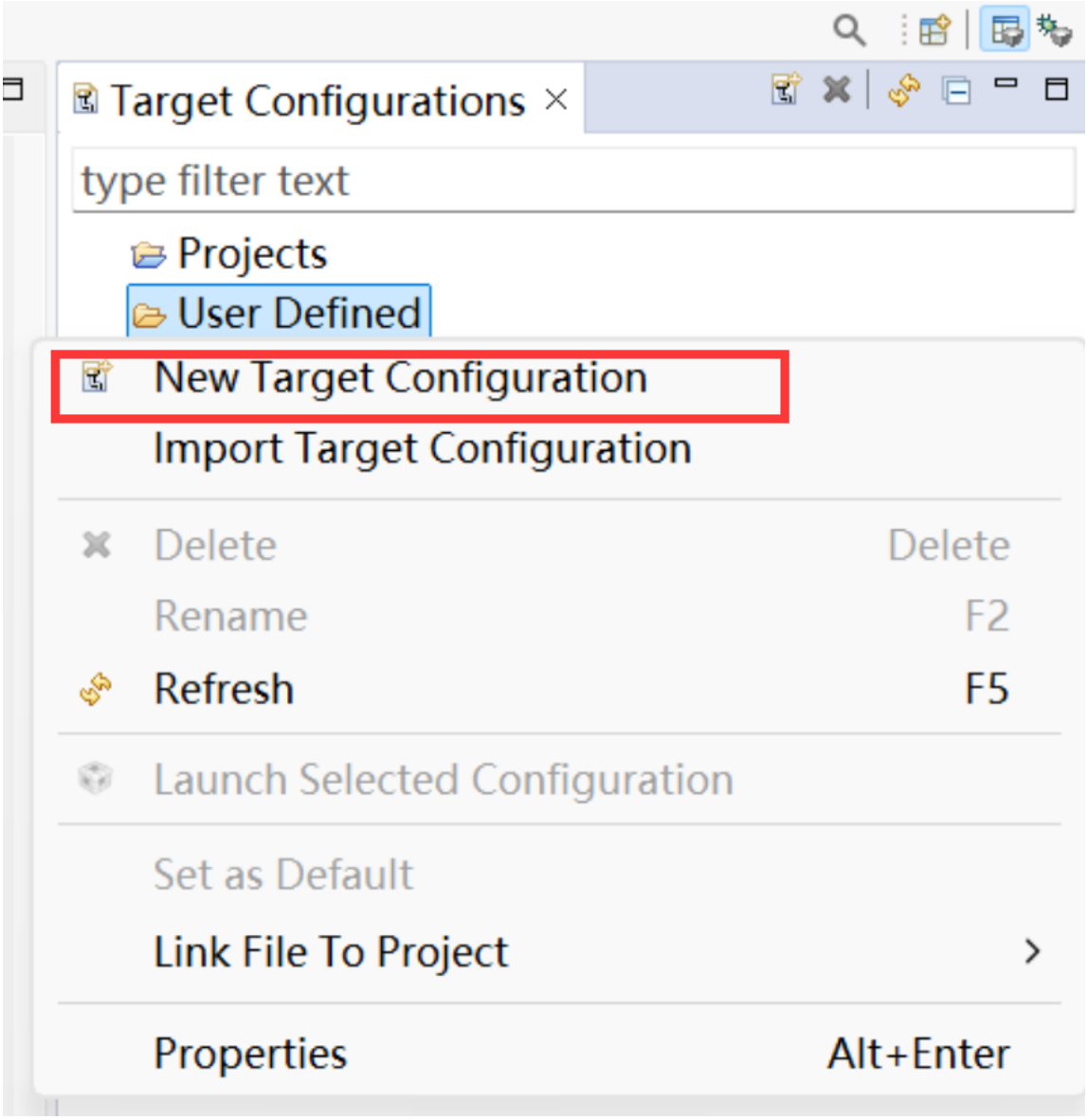
编译的详细过程，请参考附录 5.使用 Makefile 进行编译。

**2. 烧写镜像文件**

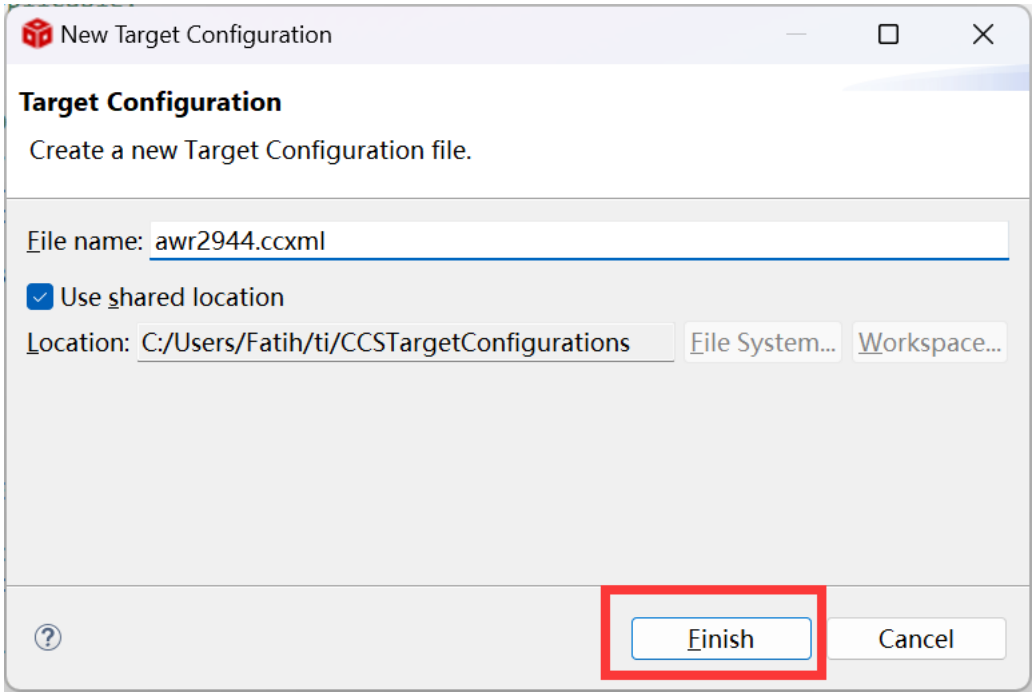
使用 UniFlash 将 *awr2944_ccsdebug.appimage* 和 *sbl_qspi.release.tiimage* 烧写到 AWR2944EVM 的 Flash。

**3. 创建新的目标配置文件(.ccxml)并启动。**

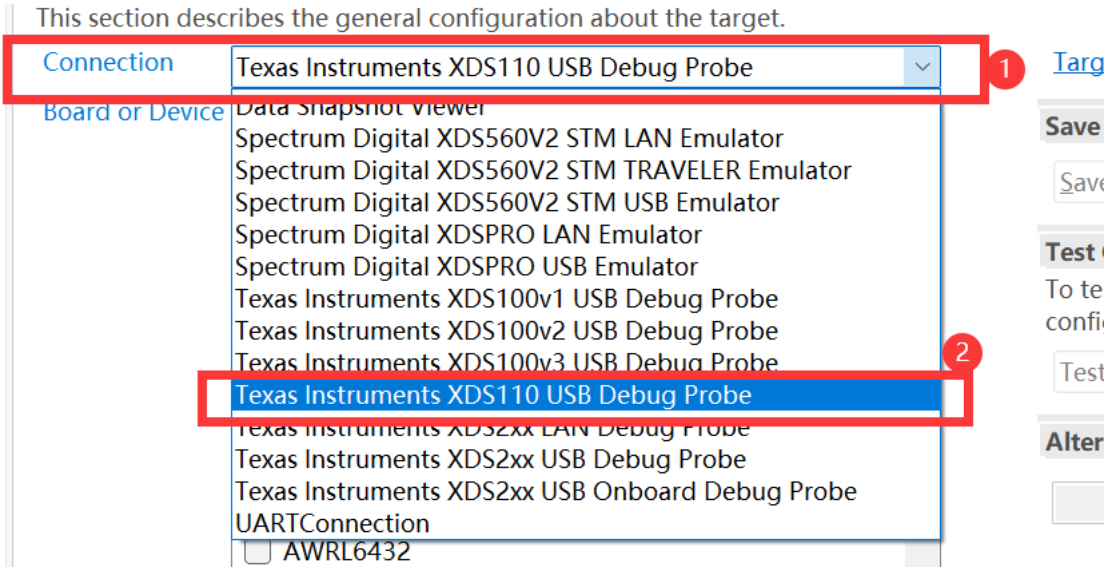
1. 打开 CCS，选择 Target Configurations->User Defined，右键，选择 New Target Configuration。



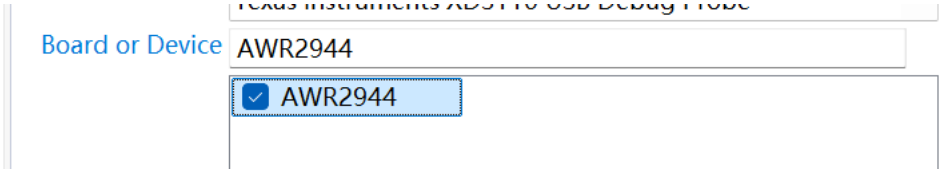

可以为新建的目标配置文件重命名，也可以直接点击 Finish



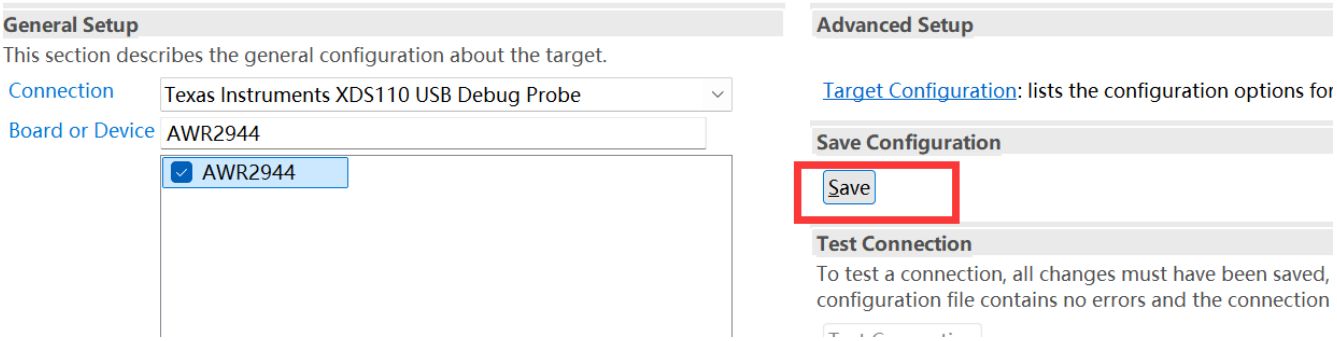

2. 在 Connection 栏选择 XDS110 USB Debug Probe



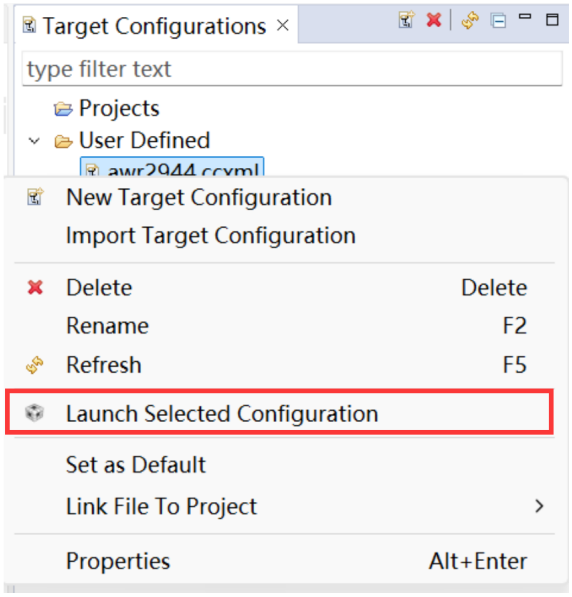

3. Board or Device 选择 AWR2944




4. 选择 Save 保存设置




5. 在 Target Configurations 栏对新建的目标配置文件文件右键，选择 Launch Selected Configuration.




**4. 连接 ARM/C66xx 核**

连接 R5_0 核，点击 CPU Reset，连接 C66xx 核，点击 CPU reset。

**5. 加载应用程序二进制文件**

1. 选择 R5_0，使用 Load Program，选择 C:\ti\mmwave_mcuplus_sdk_04_06_01_02\mmwave_mcuplus_sdk_04_06_01_02\ti\demo\awr294x\mmw 下的 awr2944_mmw_demo_mssDDM.xer5f，选择 OK。
2. 选择 C66xx，使用 Load Program，选择 C:\ti\mmwave_mcuplus_sdk_04_06_01_02\mmwave_mcuplus_sdk_04_06_01_02\ti\demo\awr294x\mmw 下的 awr2944_mmw_demo_dssDDM.xe66，选择 OK。

**6. 选择 R5_0/C66xx 核进行调试。**

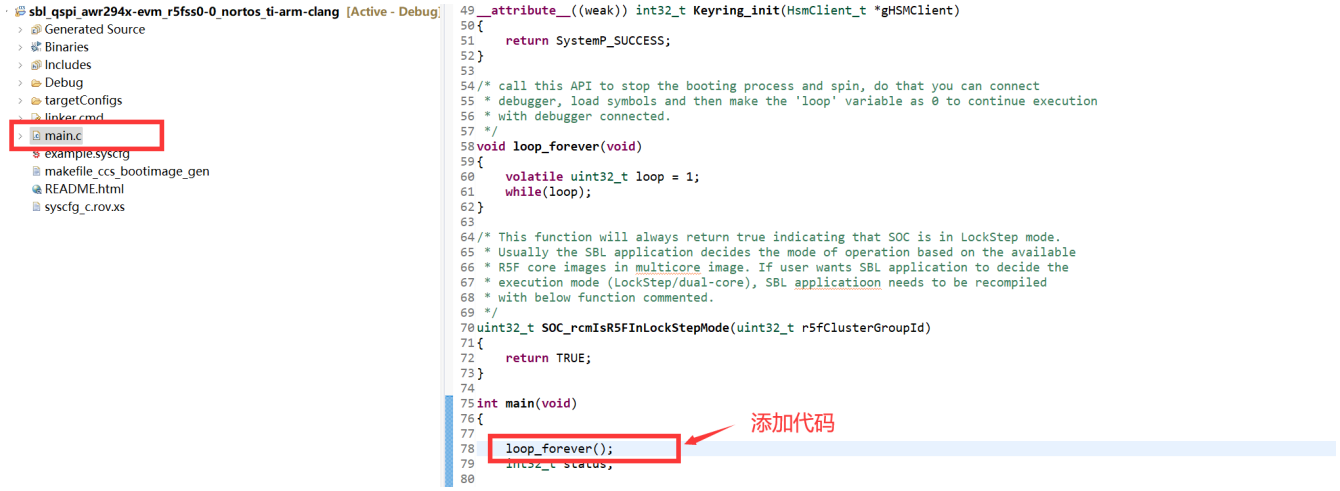
在加载相应可执行文件后，跳转至 main()函数，对程序进行在线调试。

## 四、调试 SBL

SBL 的调试和上面提到的方法不同，调试 SBL 需要将 SBL 烧写到 Flash 后调试。SBL 由 RBL 加载并运行，上电后将立即执行。为了可以停在 SBL 开始的位置，需要在 SBL 程序起始位置加入循环空转函数，在连接上调试器后使用调试器访问并修改循环条件，使其退出循环空转并执行 SBL 代码。

**1. 导入 sbl_qspi 工程并修改**

1. 在 CCS 里点击菜单 File->Import，在路径 C:\ti\mmwave_mcuplus_sdk_04_06_01_02\mcu_plus_sdk_awr294x_09_02_00_15\examples\drivers\boot\sbl_qspi\awr294x-evm 下选择 sbl_qspi_awr294x-evm_r5fss0-0_nortos_ti-arm-clang 工程，导入 AWR2944 的 sbl qspi 工程。
2. 打开 main.c，在 main()函数的起始位置添加 loop_forever，注意 loop_forever 已经被定义。



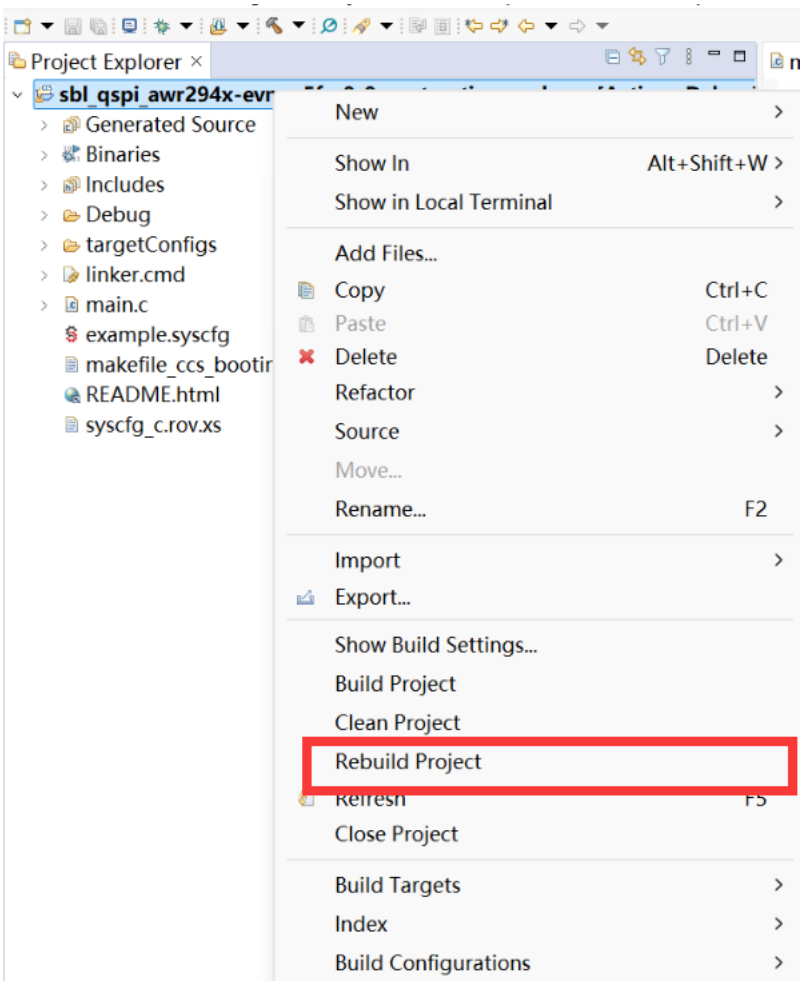

```
__attribute__((weak)) int32_t Keyring_init(HsmClient_t *gHSMClient)
{
    return SystemP_SUCCESS;
}

/* call this API to stop the booting process and spin, do that you can connect
 * debugger, load symbols and then make the 'loop' variable as 0 to continue execution
 * with debugger connected.
 */
void loop_forever(void)
{
    volatile uint32_t loop = 1;
    while(loop);
}

/* This function will always return true indicating that SOC is in LockStep mode.
 * Usually the SBL application decides the mode of operation based on the available
 * R5F core images in multicore image. If user wants SBL application to decide the
 * execution mode (LockStep/dual-core), SBL applicatioon needs to be recompiled
 * with below function commented.
 */
uint32_t SOC_rcmIsR5FInLockStepMode(uint32_t r5fClusterGroupId)
{
    return TRUE;
}

int main(void)
{
    loop_forever();
    int32_t status;
```

3. 保存代码，并进行 Rebuild.




**2. 烧录镜像**

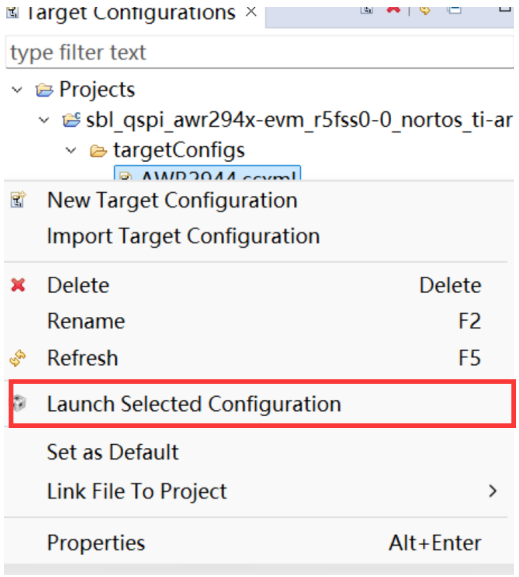
选择 C:\Users\Username\workspace_v12\sbl_qspi_awr294x-evm_r5fss0-0_nortos_ti-arm-clang\Debug\sbl_qspi_awr294x-evm_r5fss0-0_nortos_ti-arm-clang.tiimage 作为 SBL Image。

选择 C:\ti\mmwave_mcuplus_sdk_04_06_01_02\mmwave_mcuplus_sdk_04_06_01_02\ti\utils\ccsdebug\awr2944_ccsdebug.appimage 作为 APP Image。

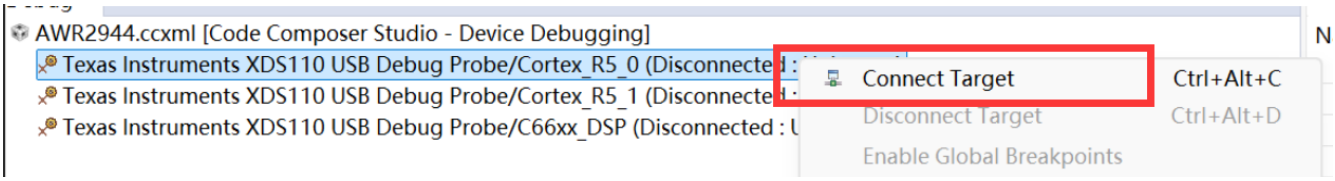
将两个镜像程序通过 UniFlash 烧录至 AWR2944EVM 的 Flash 上。

**3. 加载符号表**

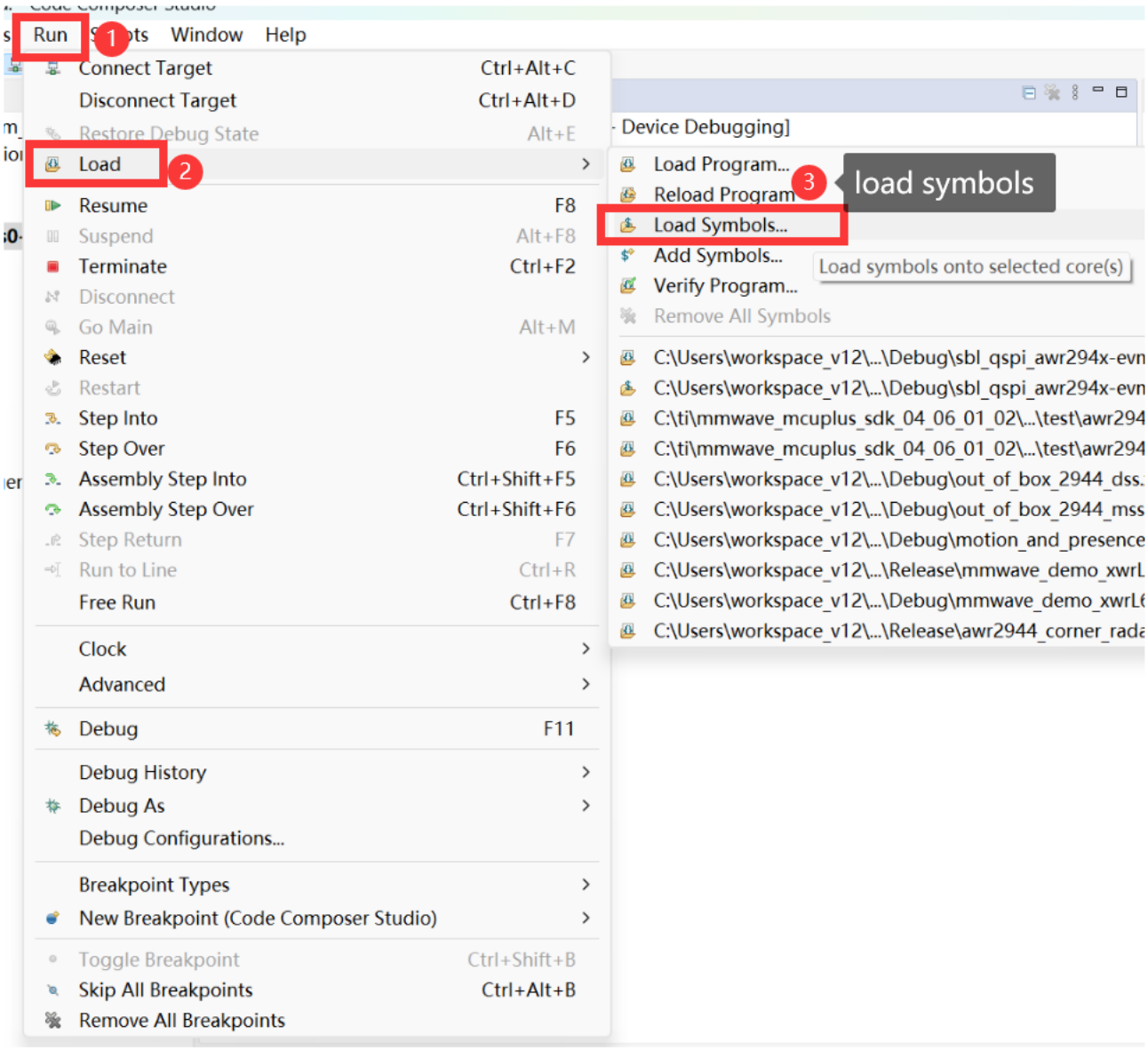
AWR2944 EVM 的 SOP 设置为 001 模式后上电。在 CCS 里点击 SBL 工程里的 AWR2944.ccxml 文件，右键选择 Launch Selected Configuration。




选择 R5_0 核，右键选择 Connect Target。




在 CCS 菜单里点击 Run->Load->Load Symbols，选择 C:\Users\Username\workspace_v12\sbl_qspi_awr294x-evm_r5fss0-0_nortos_ti-arm-clang\Debug 下的 sbl_qspi_awr294x-evm_r5fss0-0_nortos_ti-arm-clang.out，下载 SBL 的符号表。

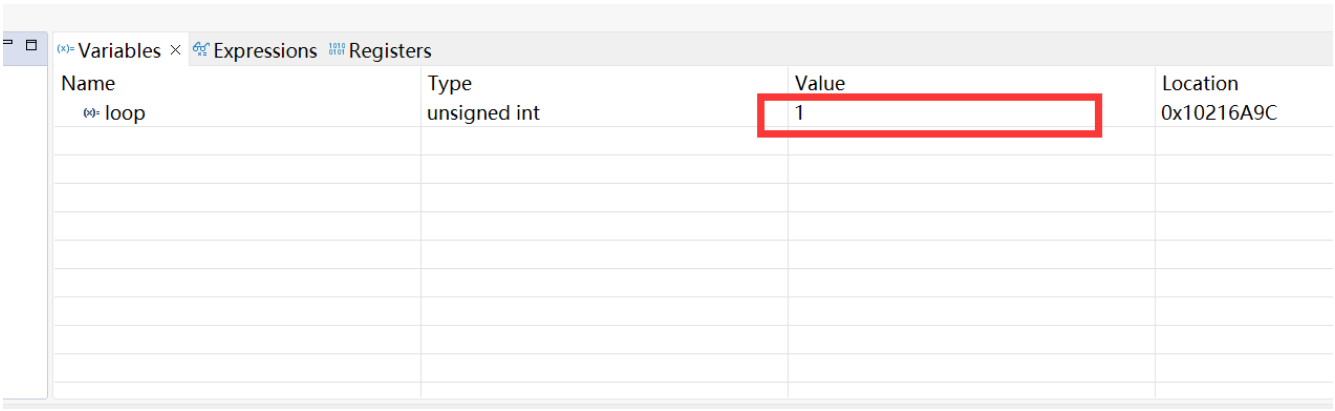



符号表下载后，用户可以看到 PC 指针停在了之前添加的 while 循环内。

```
58 void loop_forever(void)
59 {
60     volatile uint32_t loop = 1;
61     while(loop);
62 }
63
64 /* This function will always return true indicating that SOC is in LockStep mode.
65  * Usually the SBL application decides the mode of operation based on the available
66  * R5F core images in multicore image. If user wants SBL application to decide the
67  * execution mode (LockStep/dual-core), SBL applicatioon needs to be recompiled
68  * with below function commented.
69  */
70 uint32_t SOC_rcmIsR5FInLockStepMode(uint32_t r5fClusterGroupId)
71 {
72     return TRUE;
73 }
74
75 int main(void)
76 {
77     int32_t status;
78
79     loop_forever();
80     Bootloader_profileReset();
```

**4. 跳出循环，调试 SBL**

修改 Variables 里的变量 loop 的值为 0，使得程序跳出 while（looop），loop_forever 函数退出。




接下来就可以单步运行以调试 SBL 程序了。

## 五、附录

### 1. SOP 模式

AWR2944 上电后首先运行的是芯片 ROM 里固化的 RBL。RBL 负责器件的初始化(Power, clock 和 连接外设)与 SOP 模式的检查，SOP 模式决定后续 RBL 的操作模式，具体如下图所示：

SOP[2:0]模式

| 参考 | 用途 | 备注 |
|---|---|---|
| J17 (SOP 2), J18 (SOP 1), J20 (SOP 0) | SOP[2:0] | 101 (SOP 模式 5) = 烧写模式<br>001 (SOP 模式 4) = 功能模式<br>000 (SOP 模式 3) = 保留<br>011 (SOP 模式 2) = 开发模式<br>010 (SOP 模式 1) = 保留 |

模式 2 开发模式(Development mode)：在该模式下不执行后续的启动程序，可以用于 mmWave Studio 的连接使用。

模式 4 功能模式(function mode)：RBL 使用 QSPI 接口从串行 Flash 加载 SBL。它会尝试读取串行 Flash 的第一个镜像（位于偏移量 0x0 处），也就是 SBL 存放的地址。一旦 SBL 被成功加载到 RAM 中，RBL 将执行跳转，将控制权转交给 SBL。如果失败，RBL 还会最多尝试加载 3 个不同 Flash 偏移的镜像。

模式 5 烧写模式(Flashing Mode)：RBL 等待通过 UART 接口接收数据。RBL 开始在 UART 端口上发送 PING 信号（字符"C"），表明它正在等待外部主机（例如电脑）发送镜像文件。外部主机通过 UART 使用 XMODEM 协议将 Flash Writer 镜像写入 AWR2944 RAM 中，随后 Flash Writer 执行后等待主机端将镜像文件通过串口下载。Flash Writer 在设备的 RAM（通常是 MSS L2 内存）中接收并暂存镜像文件。接收完成后，Flash Writer 将接收到的镜像（例如 SBL 和应用程序镜像）写入 Flash 中。

对于 AWR2944 可以使用 UniFlash 这个软件将需要写入的镜像文件烧录到 Flash 中，UniFlash 提供一个 GUI 界面，可以简单方便的操作。除此之外，还可以使用 SDK 中提供的 Python 脚本程序进行 Flash 程序烧录。脚本程序在 mmwave_mcuplus_sdk_04_06_01_02\mcu_plus_sdk_awr294x_09_02_00_15\tools\boot 文件夹里。

### 2. SBL 的功能

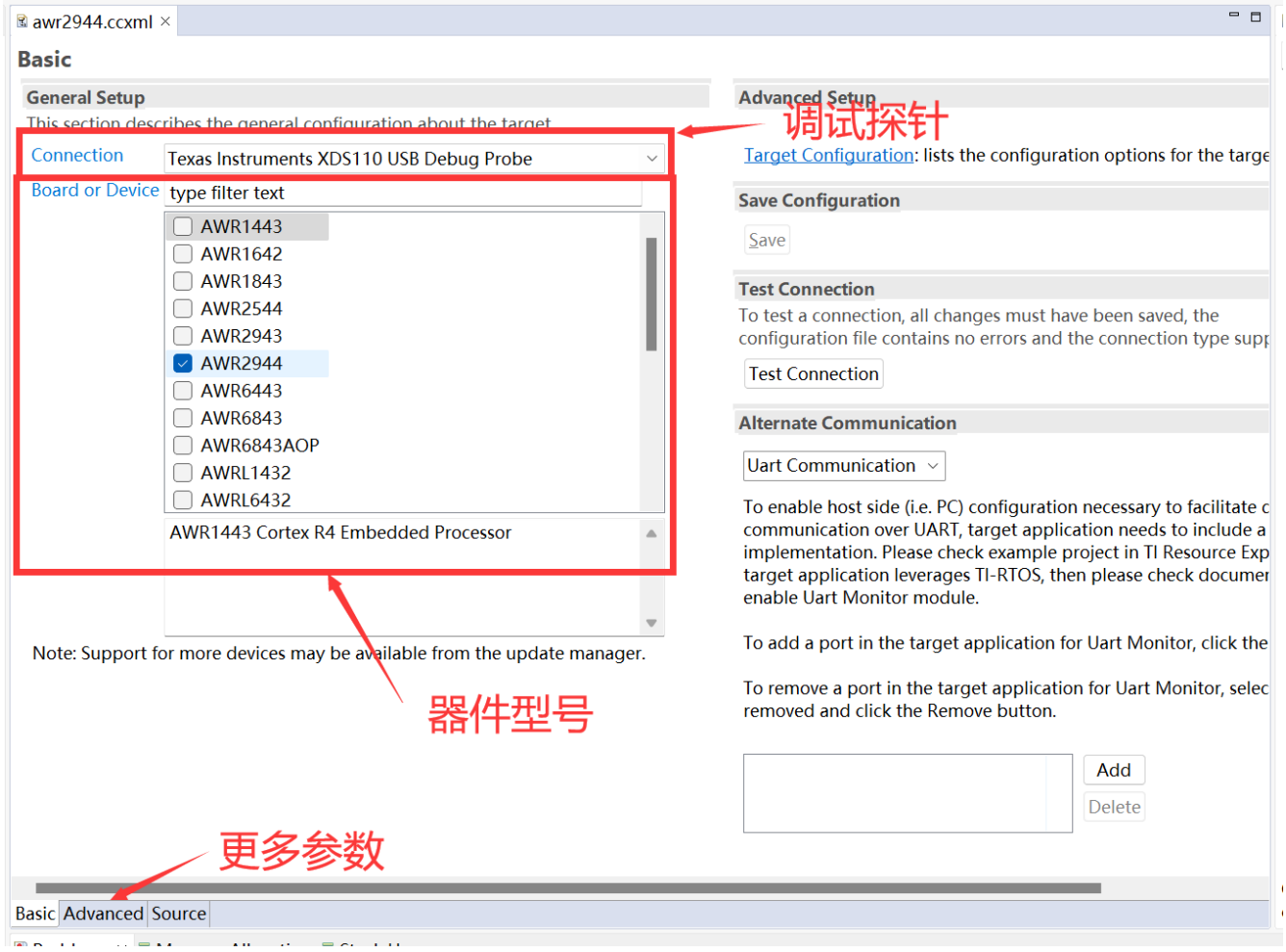
SBL 是一段存在于串行 Flash 中的程序，不同于 RBL，SBL 可以由用户修改，这提供了极大的灵活性。当芯片 SOP=001 并且 Flash 上预先烧写好 SBL，上电后 RBL 会加载 SBL 到芯片内存里并运行。SBL 的功能主要有一下几点：

1. 系统初始化：SBL 在启动后，首先执行一系列系统初始化操作。这些操作通常包括配置系统时钟、初始化内存控制器和其他硬件子系统，以确保 AWR2944 处于一个稳定的状态，准备好执行下一步的操作。
2. 加载应用程序镜像：SBL 的一个核心功能是从 Flash 加载用户的应用程序镜像。这个镜像通常是一个多核应用程序镜像（Appimage），它包含了多个用于不同处理器核（R5F、C66，射频子系统的 R4）的二进制文件。SBL 解析这个多核应用程序镜像，识别其中每个核的专用镜像文件，并将这些文件加载到对应的处理器核的内存中。SBL 需要一个正确的应用程序镜像，因为在 SBL 的程序中会对各个核应用程序镜像的状态进行检查，如果这个处理器核的应用程序镜像不存在，或者格式错误，将导致这个核程序运行的状态（status）被设定为 Failed，这会跳过后续的启动该处理器核的操作，使得这个处理器核无法被正确加载并运行。
3. 启动处理器核：SBL 在将每个处理器核的镜像文件加载到相应内存位置后，设置每个核的入口点（Entry Point）并启动这些核，使它们开始执行各自的应用程序。设备的各个处理器核启动顺序为：RSS_R4，C66SS0，R5FSS_1。最后 RF5SS_0 会加载自身对应的镜像到内存，一旦镜像加载完成，程序执行必要的复位操作，使得 R5FSS_0 开始执行自应用程序。
4. 在线升级/烧写应用程序镜像：SBL 支持通过 UART、CAN-FD、以太网等多种接口进行在线升级。SBL 在运行时可以通过这些接口接收新的应用程序镜像，并将其烧写到 AWR2944 外部的闪存中。

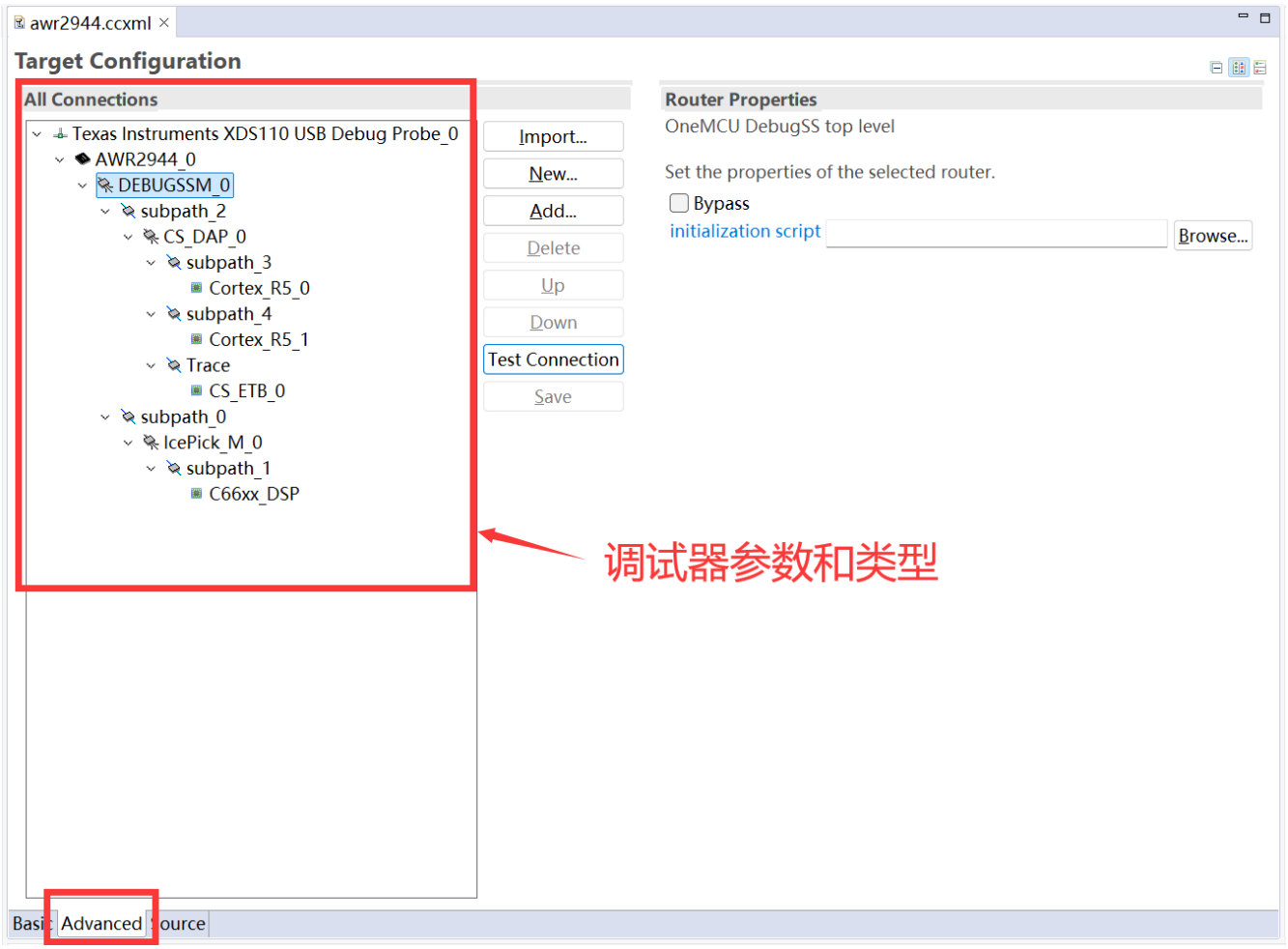
**3. 目标配置文件:**

目标配置文件（Target Configuration File）主要有如下功能：

1. 指定你正在使用的目标处理器或 MCU 的型号和系列
2. 文件配置了所使用的调试器类型及厂商（例如 XDS110， XDS560 等）




3. 定义了调试连接的参数，如时钟速率、接口类型等。




目标配置文件提供了一个直观的图形界面用户选择与配置，也可以通过直接查看源码的方式进行配置


**4. awr2944_ccsdebug.appimage：**

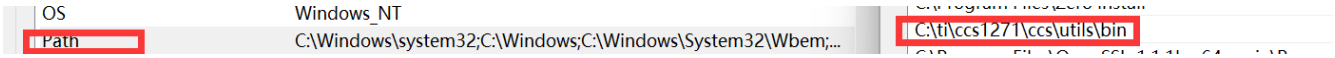
*awr2944_ccsdebug.appimage* 的功能是为 CCS 在线调试提供合适的连接环境。在 CCS 在线调试过程中，需要使用 Load Program 操作将可执行二进制文件加载至芯片的 RAM 中。因此，用户在 CCS 在线调试过程中，实际执行的代码是通过 Load 操作加载的可执行二进制文件，而不是烧录在串行 Flash 中的用户镜像文件。但这并不意味着，不需要烧录用户镜像文件，这是因为在设备的 SOP 模式为 function mode 时，在芯片上电后将立即执行 RBL->SBL->用户程序，其中 SBL 将对用户程序(appimage)进行解析，如果串行 Flash 中没有一个正确格式的用户镜像文件存在，将导致 SBL 将程序运行状态设置为 Failed，进而退出程序，使得 SBL 中的初始化代码无法完整执行。使用 *awr2944_ccsdebug.appimage* 的一个重要原因是，*awr2944_ccsdebug.appimage* 不仅包含了 R5F 的程序，还包含 C66xx 核的镜像，SBL 对 C66xx 核进行正确初始化使得调试器能够连接 C66xx 核。若烧录的用户镜像中不包含 C66xx 核心的镜像，用户则无法连接 C66xx 核。除此之外，SBL 的另一个重要功能是加载 BSS Firmware/Patch 到 R4F 核心，使得用户在加载 ARM 与 C66xx 核的代码后后能够正确的使用射频功能。

综上，为了可以在 CCS 里正确连上 ARM 和 C66xx 核，并且配置射频模块，用户需要烧录一个对 ARM 核和 C66xx 核正确初始化后不断循环等待，并且包含射频固件的具有正确用户镜像格式的镜像文件，即 *awr2944_ccsdebug.appimage，以满足启动的需求，*使得 SBL 程序顺利执行并等待用户通过 CCS Load Program 操作将真正需要调试的程序加载进来。

**5. 使用 Makefile 进行编译**

CCS 中自带 gmake，进入路径 C:\ti\ccs1271\ccs\utils\bin，可以看到文件夹下有一个 gmake.exe 文件。在 windows 搜索框中输入“env”，点击编辑系统环境变量，点击环境变量。

在系统环境变量 Path 中添加路径。




打开 cmd 命令行窗口，键入 gmake --version 确认环境变量已经正确添加。

```
C:\Users\[redacted]>gmake --version
GNU Make 4.1
Built for Windows32
Copyright (C) 1988-2014 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.
```

**进入路径**

C:\ti\mmwave_mcuplus_sdk_04_06_01_02\mmwave_mcuplus_sdk_04_06_01_02\scripts\windows。编辑文件 setenv.mak，将 MMWAVE_SDK_DEVICE 修改成 awr2944。

```
export MMWAVE_SDK_DEVICE ?= awr2944
export DOWNLOAD_FROM_CCS = yes
```

编辑文件 setenv.bat，将 MMWAVE SDK DEVICE 修改成 awr2944。

```
@REM Select your device. Options (case sensitive) are:
@REM    am273x, awr2943, awr2944, awr2544, awr2944LC
set MMWAVE_SDK_DEVICE=awr2944
```

打开命令行窗口，使用 cd 命令，将路径切换至该路径下，也就是 C:\ti\mmwave_mcuplus_sdk_04_06_01_02\mmwave_mcuplus_sdk_04_06_01_02\scripts\windows。键入，setenv.bat，回车，再键入 checkenv.bat，确保命令行没有报错说明操作成功。

```
C:\ti\mmwave_mcuplus_sdk_04_06_01_02\mmwave_mcuplus_sdk_04_06_01_02\scripts\windows>set AWR2544_RADARS
0/firmware/radarss/xwr25xx_radarss_metarprc.bin

C:\ti\mmwave_mcuplus_sdk_04_06_01_02\mmwave_mcuplus_sdk_04_06_01_02\scripts\windows>set C66X_CODEGEN_I

C:\ti\mmwave_mcuplus_sdk_04_06_01_02\mmwave_mcuplus_sdk_04_06_01_02\scripts\windows>set C66x_DSPLIB_IN

C:\ti\mmwave_mcuplus_sdk_04_06_01_02\mmwave_mcuplus_sdk_04_06_01_02\scripts\windows>set C66x_MATHLIB_I

C:\ti\mmwave_mcuplus_sdk_04_06_01_02\mmwave_mcuplus_sdk_04_06_01_02\scripts\windows>call checkenv.bat
-------------------------------------------
mmWave Build Environment Configured
-------------------------------------------
```

不要关掉命令行窗口，如果关闭了命令行窗口则需要再次到路径下键入 setenv.bat 并运行。使用命令行切换路径，cd C:\ti\mmwave_mcuplus_sdk_04_06_01_02\mmwave_mcuplus_sdk_04_06_01_02\ti\demo\awr294x\mmw

在该路径下，使用命令输入 gmake all，按下回车并等待编译完成。如果是第一次编译，先运行 gmake clean，再运行 gmake all。

**参考资料：**

1. mmwave_mcuplus_sdk_user_guide(AWR2944 SDK)

   https://dr-download.ti.com/software-development/software-development-kit-sdk/MD-U4MY7aGNn5/04.06.00.01/mmwave_mcuplus_sdk_04_06_00_01-Windows-x86-Install.exe
2. AWR294X Technical Reference Manual

   https://www.ti.com/lit/pdf/spruiv5
3. AWR2944 Evaluation Module User’s Guide

   https://www.ti.com/lit/pdf/spruj22

# 重要通知和免责声明

TI“按原样”提供技术和可靠性数据（包括数据表）、设计资源（包括参考设计）、应用或其他设计建议、网络工具、安全信息和其他资源，不保证没有瑕疵且不做出任何明示或暗示的担保，包括但不限于对适销性、某特定用途方面的适用性或不侵犯任何第三方知识产权的暗示担保。

这些资源可供使用 TI 产品进行设计的熟练开发人员使用。您将自行承担以下全部责任：(1) 针对您的应用选择合适的 TI 产品，(2) 设计、验证并测试您的应用，(3) 确保您的应用满足相应标准以及任何其他功能安全、信息安全、监管或其他要求。

这些资源如有变更，恕不另行通知。TI 授权您仅可将这些资源用于研发本资源所述的 TI 产品的相关应用。 严禁以其他方式对这些资源进行复制或展示。您无权使用任何其他 TI 知识产权或任何第三方知识产权。您应全额赔偿因在这些资源的使用中对 TI 及其代表造成的任何索赔、损害、成本、损失和债务，TI 对此概不负责。

TI 提供的产品受 TI 的销售条款或 ti.com 上其他适用条款/TI 产品随附的其他适用条款的约束。TI 提供这些资源并不会扩展或以其他方式更改 TI 针对 TI 产品发布的适用的担保或担保免责声明。

TI 反对并拒绝您可能提出的任何其他或不同的条款。

邮寄地址：Texas Instruments, Post Office Box 655303, Dallas, Texas 75265