Telink IoT Studio 用户指南

AN-22063003-C1

Ver1.0.10

2024.03.05

Keyword

Telink, IoT Studio

Brief

本文档旨在指导如何使用 Telink IoT Studio 工具。

Published by
Telink Semiconductor

Bldg 3, 1500 Zuchongzhi Rd,
Zhangjiang Hi-Tech Park, Shanghai, China

Legal Disclaimer

This document is provided as-is. Telink Semiconductor reserves the right to make improvements without further notice to this document or any products herein. This document may contain technical inaccuracies or typographical errors. Telink Semiconductor disclaims any and all liability for any errors, inaccuracies or incompleteness contained herein.

Information

For further information on the technology, product and business term, please contact Telink Semiconductor Company www.telink-semi.com

For sales or technical support, please send email to the address of:

telinksales@telink-semi.com

telinksupport@telink-semi.com

历史版本

版本	修订日期	作者	更改内容
V1.0.0	2022-06	何雄军	首次发布
V1.0.1	2022-07	何雄军	添加更多已知问题 添加IDE设置章节
V1.0.2	2022-09	何雄军	选择带有芯片类型的版本
V1.0.3	2022-11	何雄军，王飞	添加 Jtag 调试章节
V1.0.4	2023-03	何雄军，王飞	Telink IDE 更名为 Telink IoT Studio 添加一些工具的使用说明
V1.0.5	2023-08	何雄军，王飞	V5.1.1 工具链升级成 V5.1.2 增加关于如何同时构建 `V5.1.2` 和 `V3.2.3` 工程的描述 增加 `linux` 中的安装和卸载指引
V1.0.6	2024-03	何雄军，王飞	增加 Smart Build 按钮的使用方法 增加对 `Andes` 文档和不同版本 `ICEman` 的解释
V1.0.7	2024-04	何雄军	增加 Telink Formatter 的使用方法
V1.0.8	2024-06	何雄军，王飞	增加 Telink CDT to CMake 转换工具的使用方法 更新 `Telink Menu` 相关图片 删除所有 `build tool version` 相关信息
V1.0.9	2024-08	何雄军，王飞	修改关于工具链和 ICEman 的描述增加如何在命令行中编译工程的指引增加在 Windows 中开启大小写敏感的方法
V1.0.10	2025-02	何雄军，王飞	更新关于 TelinkFormatter 插件使用的描述增加通过 SVD 查看外设功能的使用方法

Contents

List of Figures

1 桌面快捷方式

Telink IoT studio 的安装程序会在桌面上创建以下两个快捷方式：

- `Telink IoT studio`：用于启动 IoT Studio。
- `Telink TC32 console`：用于启动 TLSR8 的开发控制台，用户可以在里面输入 `tc32-elf-gcc` 和 `make` 等命令。

2 不同的 IDE

本用户指南将涉及以下 3 种不同的 IDE：

- `Telink IoT studio`：本文档描述的 IDE。
- `Telink old IDE`：wiki 页面展示的针对 TLSR8 系列芯片的 IDE。(链接：IDE for TLSR8 Chips)
- `Telink RDS IDE`：wiki 页面展示的针对 TLSR9 系列芯片的 IDE。(链接：IDE for TLSR9 Chips)

上述 3 个 IDE 具有以下不同的桌面快捷方式图标：

Telink IoT studio:

Figure 2.1: Telink IoT studio 图标

Telink old IDE:

Figure 2.2: 用于 TLSR8 系列芯片的 Telink Old IDE 图标

Telink RDS IDE:

Figure 2.3: 用于 TLSR9 系列芯片的 Telink RDS IDE 图标

3 IoT Studio 或集成工具

- 构建 TLSR8 工程和 SDK；
- 导入和构建 TLSR9 Telink RDS 工程和 SDK；
- Telink Windows 版 BDT 工具用于给 TLSR8 和 TLSR9 系列芯片烧录镜像文件；
- Telink libUsb 版 BDT 工具用于给 TLSR8 和 TLSR9 系列芯片烧录镜像文件，仍处于测试阶段；
- Jtag_Burn 通过 JTAG 接口给 TLSR9 系列芯片烧录镜像文件。

4 Telink RDS 和 Telink IoT studio 的工程属性差异

4.1 Ext HW DSP option

这个选项用于配置是否使用 DSP 及其库。

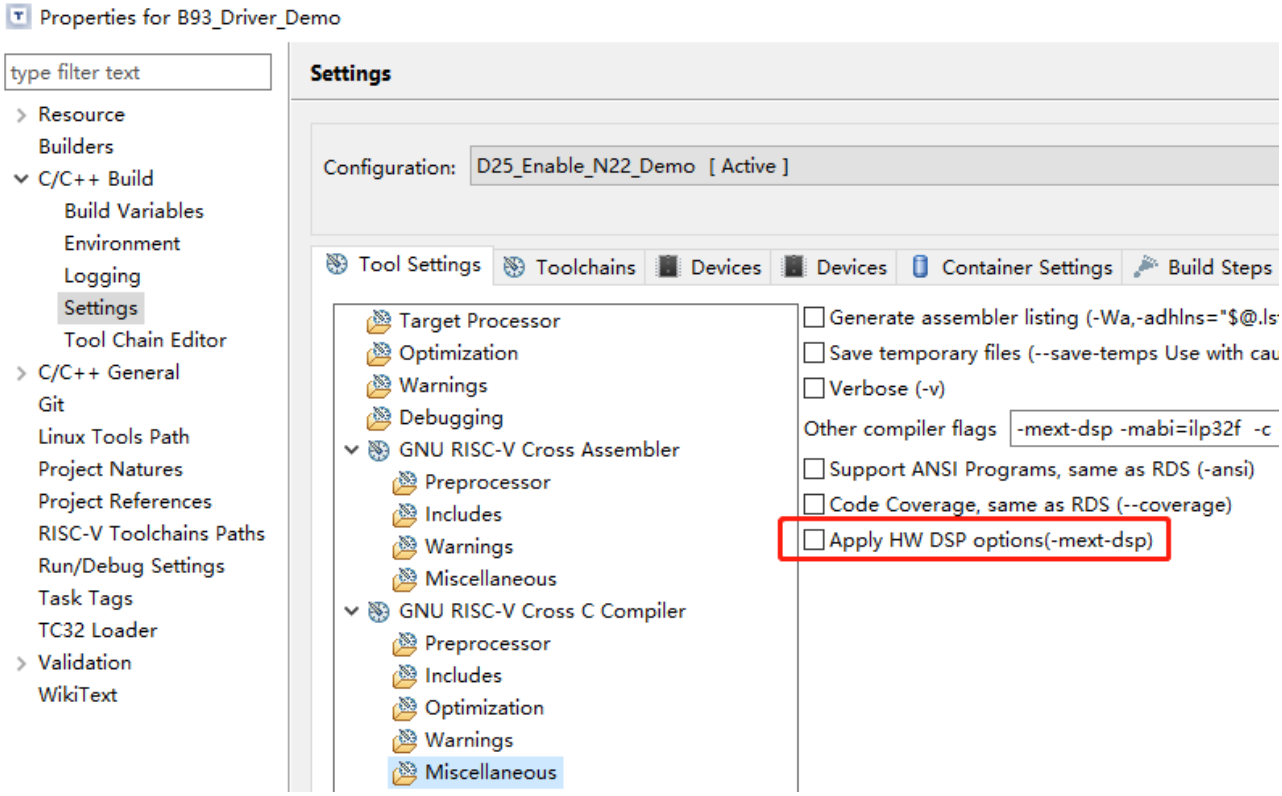

Figure 4.1: Telink IoT studio 中的 DSP 选项

此选项将在 `cc` 和 `ld` 中添加 `-mext-dsp` 标志，同时也将在 Telink RDS IDE 中向链接器添加 `-lm` 和 `-ldsp` 标志。而 Telink IoT studio 只会在 `cc` 和 `ld` 中添加 `-mext-dsp` 标志，而不添加 `-lm` 和 `-ldsp` 这两个额外的库标志。这使得配置更加灵活。

用户可以在链接器库设置页面上专门添加 `math` 和 `dsp` 库到链接器。

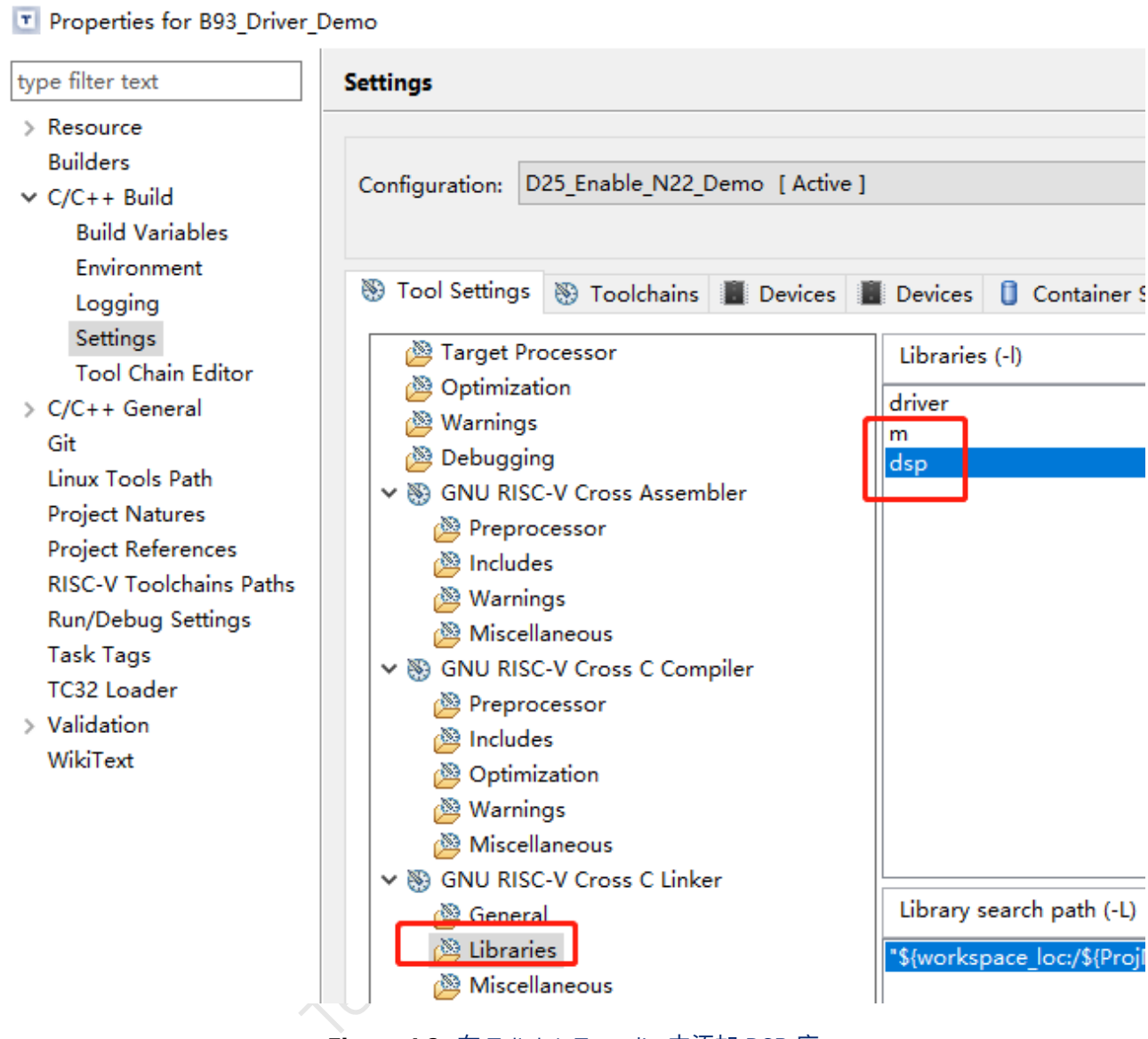

Figure 4.2: 在 Telink IoT studio 中添加 DSP 库

4.2 `-msave-restore` 和 `-msmall-data-limit`

`Telink RDS IDE` 对 `-msave-restore` 和 `-msmall-data-limit` 都使用 IoT Studio 的内置值（默认），因此配置页上没有这些选项，而 `Telink IoT studio` 对这些选项的配置则会更加灵活。

用户应该对 TLSR9 系列芯片使用 `Use toolchain default value` 选项。

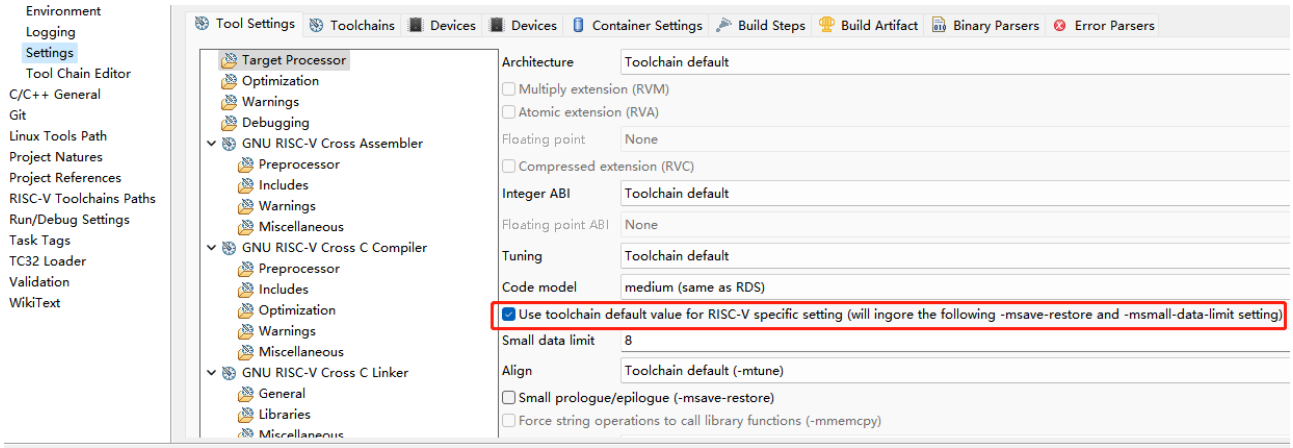

Figure 4.3: Telink IoT studio 上的 RISC-V 特定选项

5 工具链和构建工具

5.1 工具链信息

Telink IoT studio 包含以下几个用于 TLSR8 和 TLSR9 的工具链：

- 用于 TLSR8 芯片，带有 GCC 4.5.1 (tc32-elf) 版本的 Telink TC32 交叉编译器，与Telink old IDE 相同。
- 用于 TLSR9（D25F 和 N22 架构）芯片，Telink RDS IDE V3.2.3 的 GCC 7.4 版交叉编译器。
- 用于 TLSR9（D25F 和 N22 架构）芯片，Telink RDS IDE V5.1.2 的 GCC 10.3 版交叉编译器。
- 用于 TLSR9（D25F 和 N22 架构）芯片，Telink RDS IDE V5.3.x 的 GCC 12.2 版交叉编译器。

5.1.1 工具链位置

我们假设 Telink IoT studio 安装在 `$IoTStudio` 位置：

- 用于 TLSR8 芯片,带有 GCC 4.5.1 (tc32-elf-1.5) 版本的 Telink TC32 交叉编译器：`IoTStudio/opt/tc32/bin`
- 用于 TLSR9（D25F 和 N22 架构）芯片，Telink RDS IDE V3.2.3 的 GCC 7.4 版交叉编译器：
  - N22: `$IoTStudio/RDS/V3.2.3/toolchains/nds32le-elf-mculib-v5`
  - D25F: `$IoTStudio/RDS/V3.2.3/toolchains/nds32le-elf-mculib-v5f`
- 用于 TLSR9（D25F 和 N22 架构）芯片，Telink RDS IDE V5.1.2 的 GCC 10.3 版交叉编译器：
  - N22: `$IoTStudio/RDS/V5.1.2/toolchains/nds32le-elf-mculib-v5`
  - D25F: `$IoTStudio/RDS/V5.1.2/toolchains/nds32le-elf-mculib-v5f`
- 用于 TLSR9（D25F 和 N22 架构）芯片，Telink RDS IDE V5.3.x 的 GCC 12.2 版交叉编译器：
  - N22: `$IoTStudio/RDS/V5.3.x/toolchains/nds32le-elf-mculib-v5`
  - D25F: `$IoTStudio/RDS/V5.3.x/toolchains/nds32le-elf-mculib-v5f`

5.2 为 TLSR9 芯片工程选择一个不同的工具链

由于 TLSR8 芯片仅包含一个工具链，因此本节仅适用于 TLSR9 芯片。

要改变一个工程的工具链，首先打开工程属性，导航到 `C/C++ Build` -> `Settings` -> `Toolchains` 位置，并从下拉列表中选择不同的工具链。

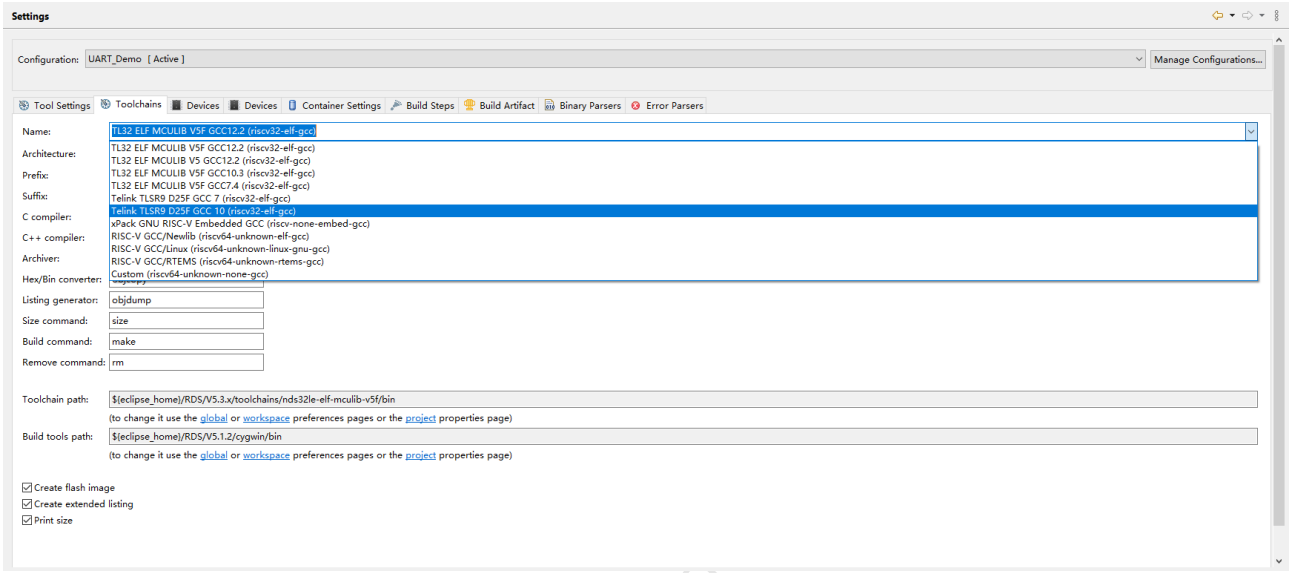

Figure 5.1: 选择工具链

列出的工具链有：

- `TL32 ELF MCULIB V5F GCC12.2` ：用于 D25F 架构芯片，来自 RDS V5.3.x
- `TL32 ELF MCULIB V5 GCC12.2` ：用于 N22 架构芯片，来自 RDS V5.3.x
- `TL32 ELF MCULIB V5F GCC10.3` ：用于 D25F 架构芯片，来自 RDS V5.1.2
- `Telink TLSR9 D25F GCC 10` ：与 `TL32 ELF MCULIB V5F GCC10.3` 相同
- `TL32 ELF MCULIB V5F GCC7.4` ：用于 D25F 架构芯片，来自 RDS V3.2.3
- `Telink TLSR9 D25F GCC 7` ：与 `TL32 ELF MCULIB V5F GCC7.4` 相同

5.2.1 高级用法：根据路径设置工具链

一般来说，用户不应该通过更改工具链路径来设置工具链。相反，应使用上一节中提到的方法设置工具链：`为 TLSR9 芯片工程选择一个不同的工具链` 。

本节适用于有特殊要求的高级用户。

通过点击右侧的属性菜单选项，来打开工程属性对话框，然后点击工程弹出的菜单：

依次选择 `C/C++Build` -> `Settings` -> `Toolchains` 选项：

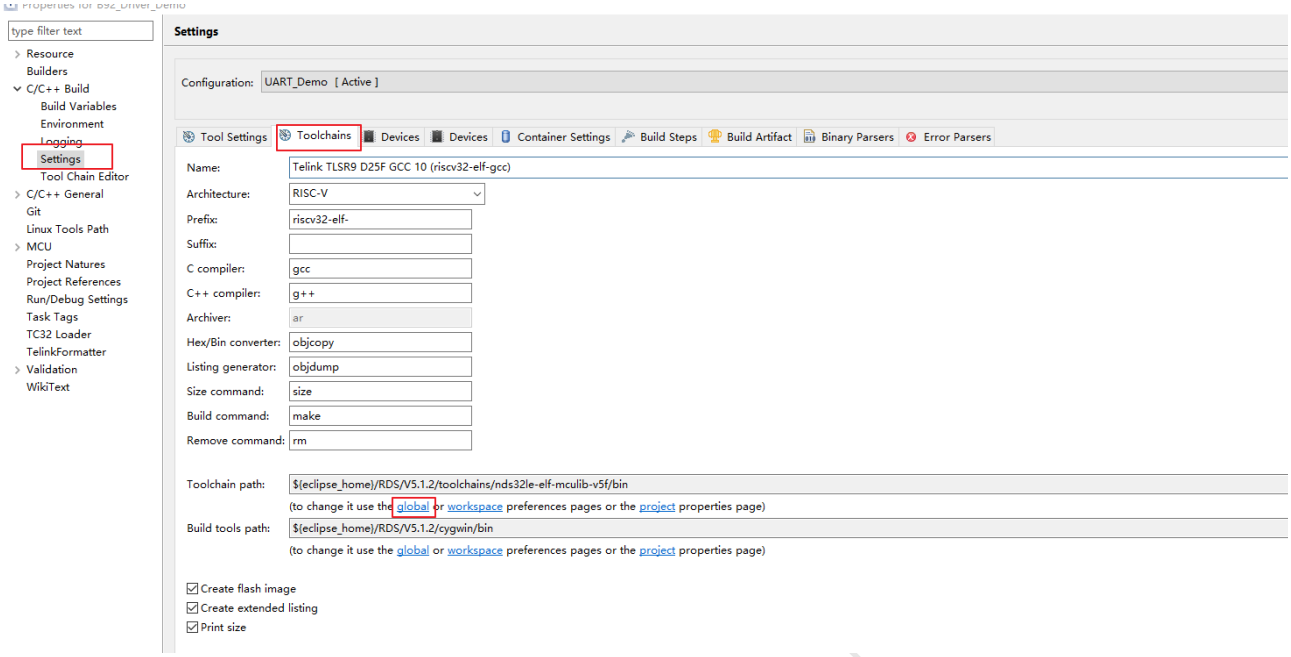

Figure 5.2: 工具链设置

点击下面红框所圈出来的 `global` 链接。

在 `Toolchain folder` 后的编辑框中设置路径。

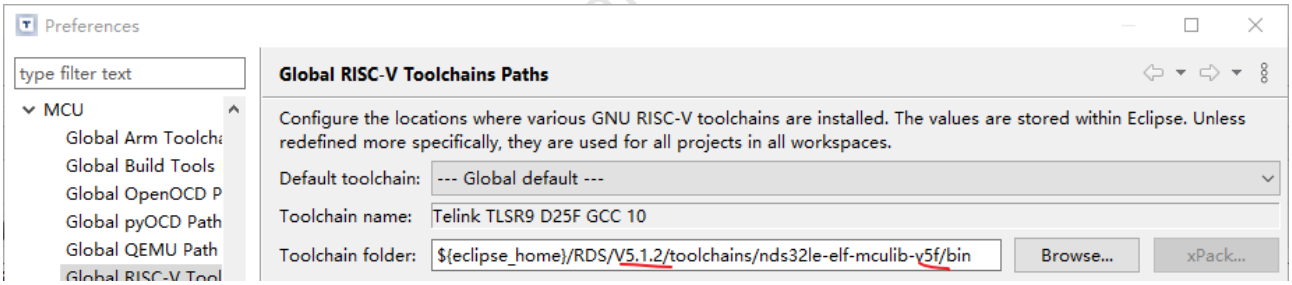

Figure 5.3: 设置工具链路径

一般来说，用户只需要更改红线所划出来的版本和后缀信息。

第一条红线表示工具链的基本版本信息（在图中为 `V5.1.2` ），用户可以将其更改为以下项目之一：

- `V3.2.3` ：Telink RDS IDE V3.2.3 版本。
- `V5.1.2` ：Telink RDS IDE V5.1.2 版本。
- `V5.3.x` ：Telink RDS IDE V5.3.x 版本。

第二条红线表示工具链变体的版本（在图中为 `nds32le-elf-mculib-v5f` ），用户可以将其改为以下项目之一：

- `nds32le-elf-mculib-v5f` ：用于 Telink TLSR9 D25F 芯片。
- `nds32le-elf-mculib-v5` ：用于 Telink TLSR9 N22 芯片。

一旦工具链路径信息发生变化，请点击 `apply and close` 按钮进行保存。

6 导入和构建项目

6.1 TLSR8

Telink IoT studio 支持 SDK 或像Telink old IDE 这样的工程导入功能。

用户可以使用 `File -> Import...` 来导入 SDK 或工程。

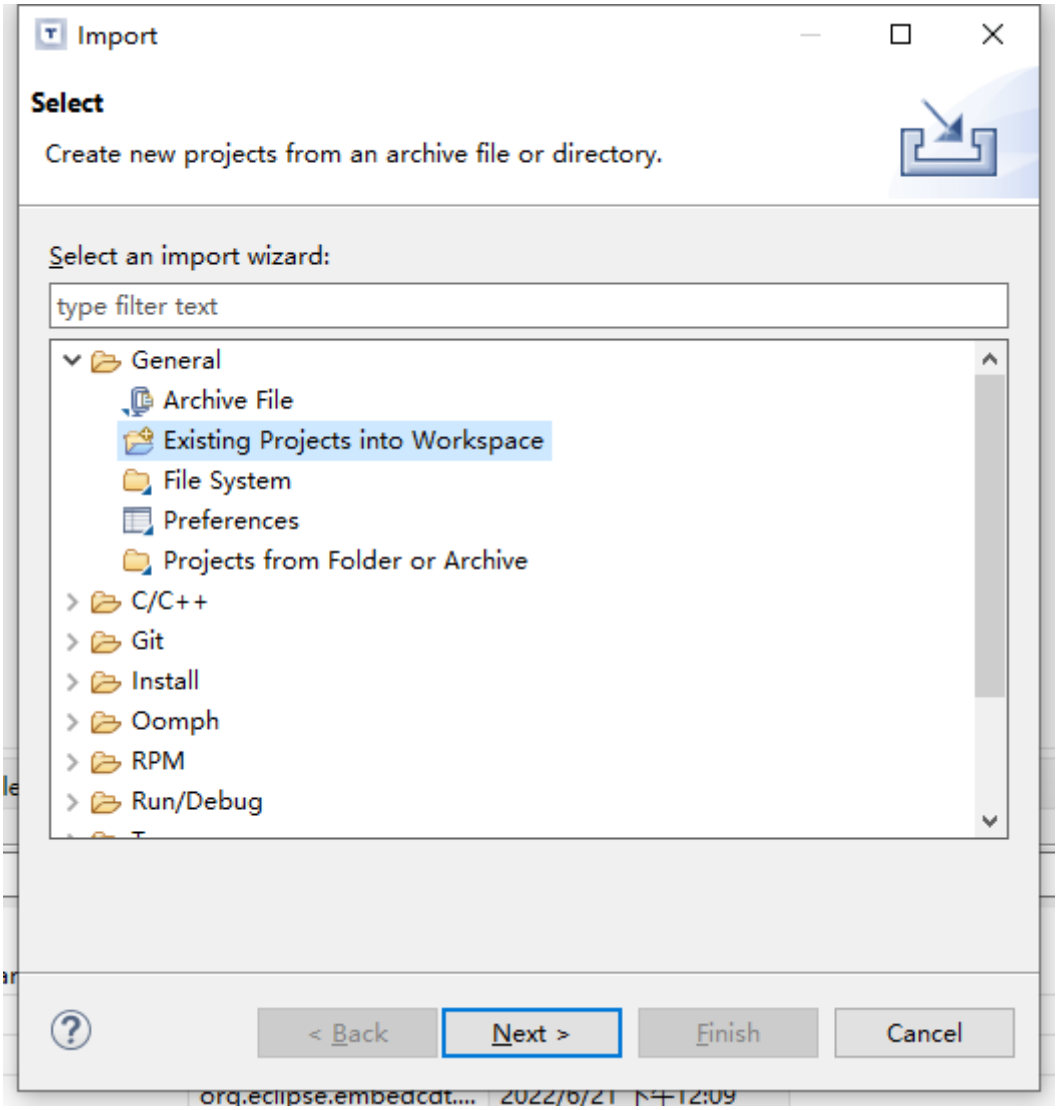

Figure 6.1: 导入界面

用户可以参考这个 Telink wiki 页面。

6.2 TLSR9

对于 `Telink RDS IDE` 格式的工程（例如，泰凌官方 wiki 网站 上的 TLSR9 芯片 SDK），在将其导入工作区之前，必须将其转换为 `Telink IoT studio` 格式。

用户可以参考下面 `Telink 菜单` 小节中的 `RDS 转换为 IDE 格式` 部分，查看如何操作。

7 Telink 菜单和工具栏条目

7.1 RDS 转换为 IoT Studio 格式

如果你从泰凌官方 wiki 网站 中获得 TLSR9 芯片 `B91` 的 SDK 或工程，它们是 `Telink RDS IDE` 格式，你可以使用此功能将其转换为 `Telink IoT studio` 格式，然后将其导入到 Telink IoT studio，`B92` 的 SDk 或工程则不需要转换

单击 `Telink` 菜单中的 `Telink RDS to IDE Converter` ：

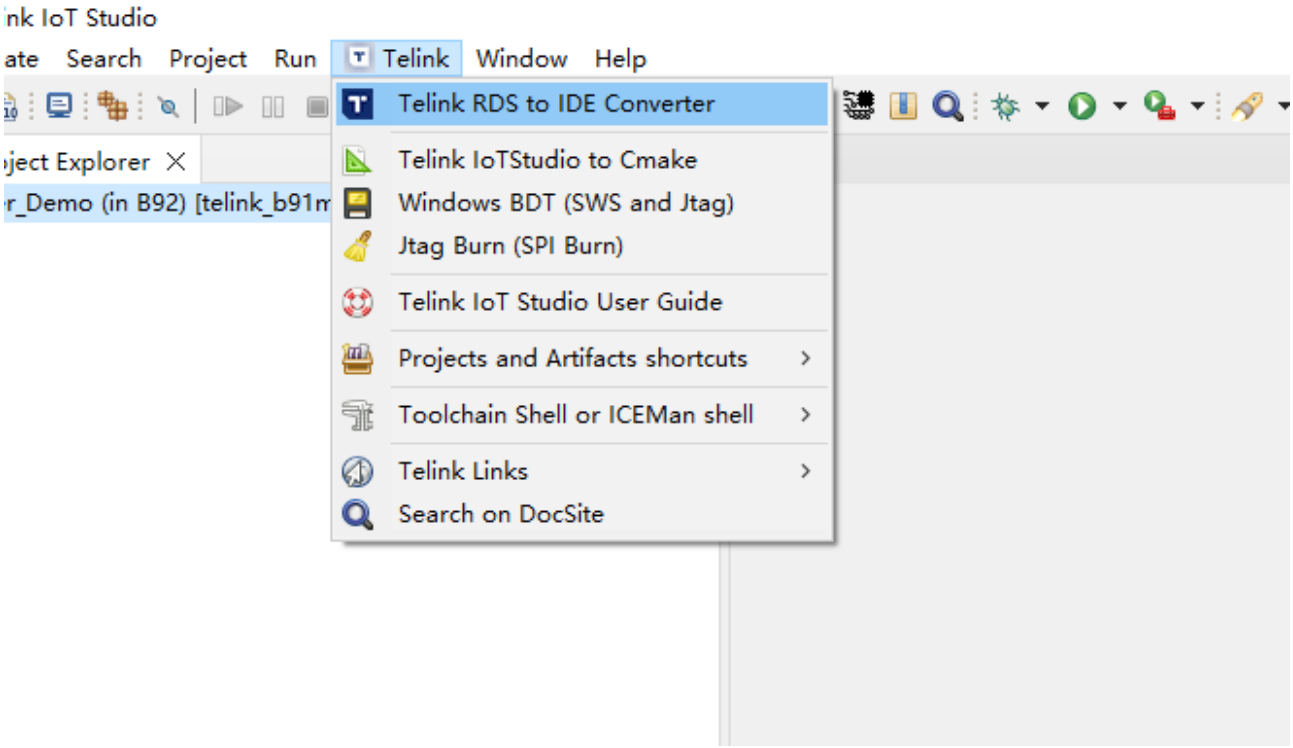

Figure 7.1: Telink 菜单

单击对话框中的 `Select .cproject file` 按钮，选择 Telink RDS IDE 格式的 `.cproject` 文件。

在 Linux 操作系统中，用户可以使用 `Ctrl + h` 或其他快捷键来显示隐藏的（`.cproject` ）文件，筛选框中应更改为 `*` ：

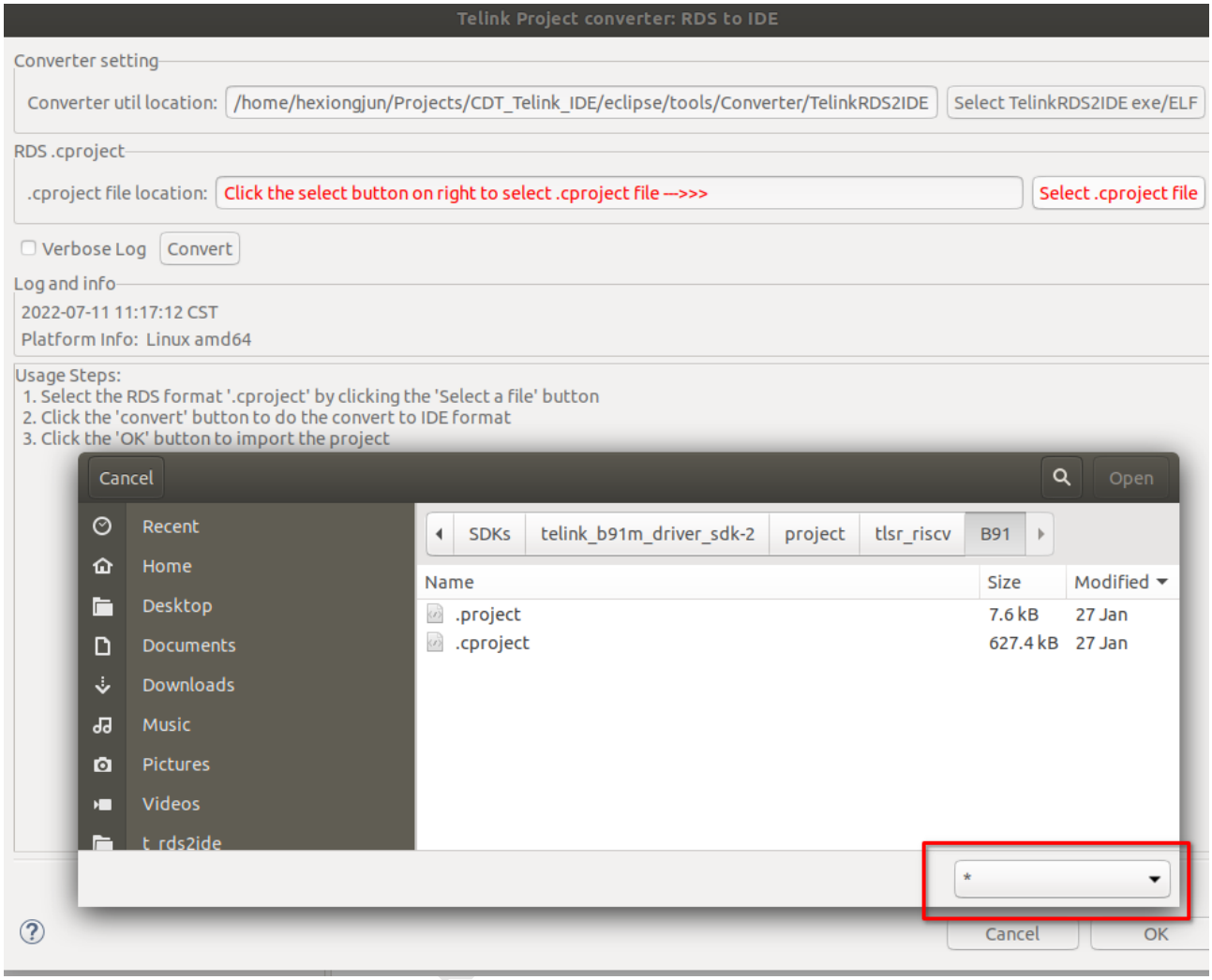

Figure 7.2: Linux 系统中选择 cproject 文件

`.cproject` 文件选择完之后，会显示文件路径：

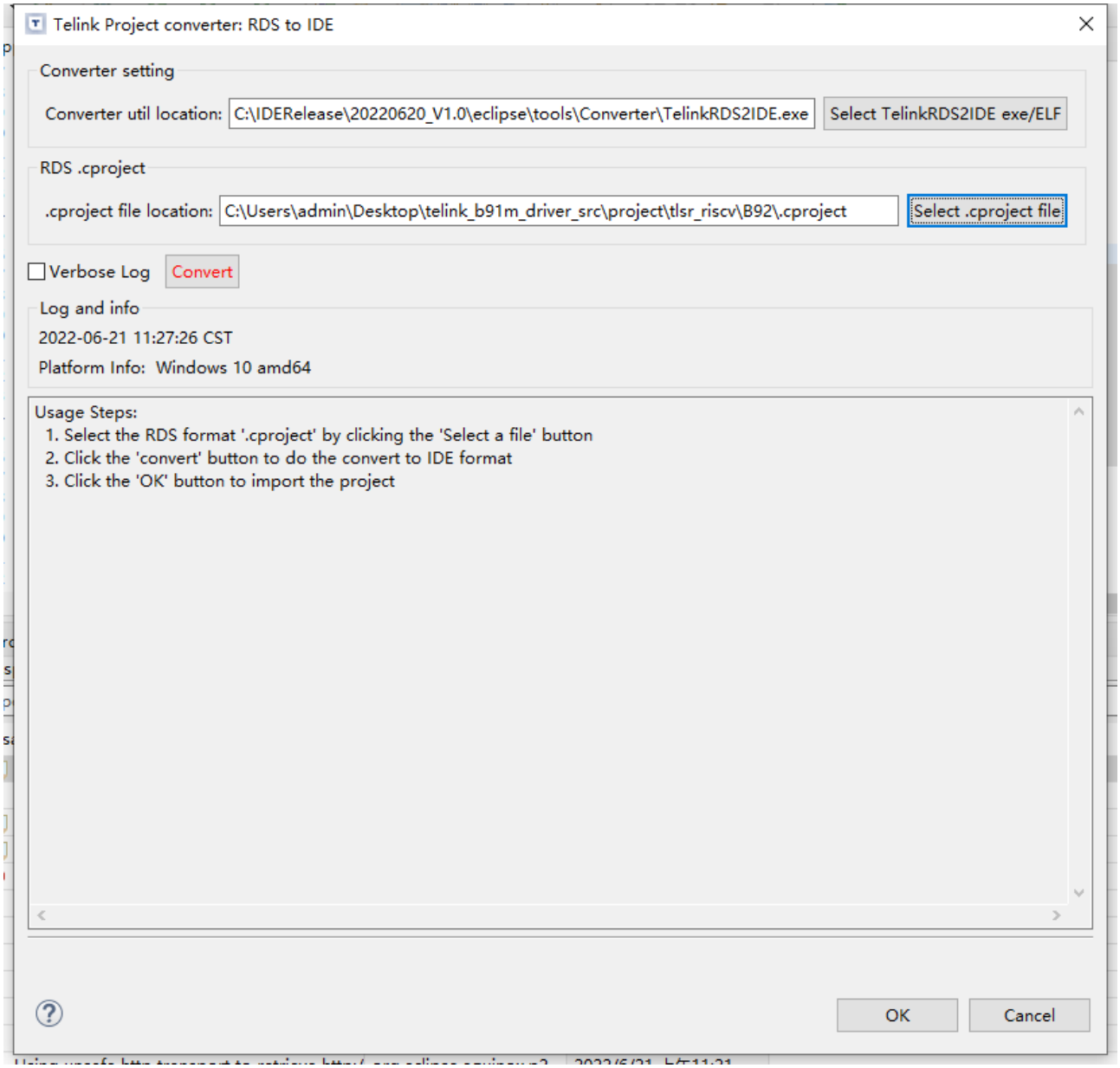

Figure 7.3: Telink 工程转换页面

然后单击 `convert` 按钮，结果将在下面的日志文本框中显示：

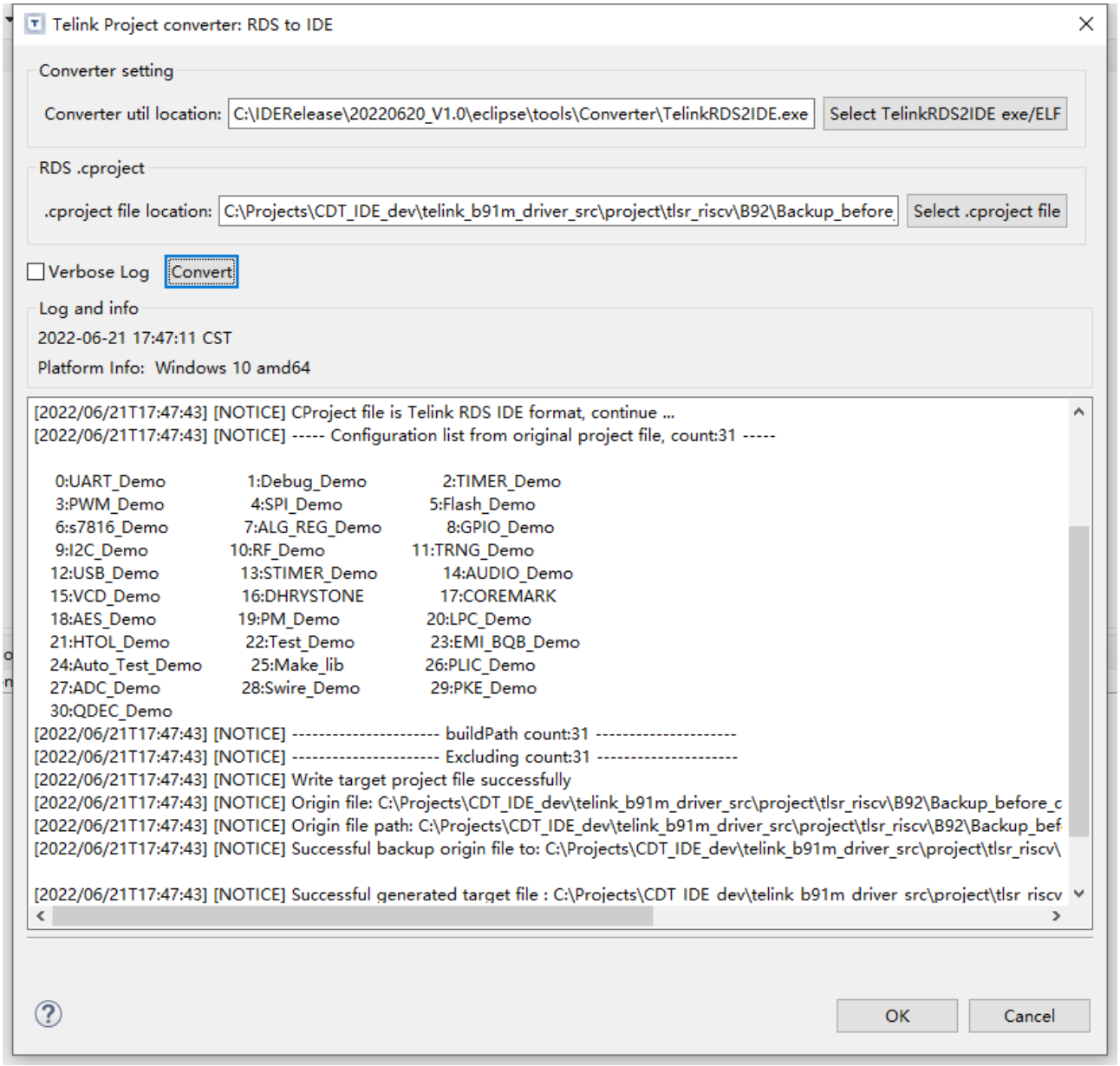

Figure 7.4: 显示转换结果

如果失败，日志文本框的背景将显示为红色。用户可以复制日志并将其发送到 `Telink IoT studio` 开发者或 FAE 以获得帮助。

7.1.1 将转换器作为一个独立程序

转换器可以作为一个独立的实用程序使用，用户可以在命令行中使用它。

转换器安装在：

- Windows：$IoTStudio_PATH/tools/Converter/TelinkRDS2IDE.exe
- Linux：$IoTStudio_PATH/tools/Converter/TelinkRDS2IDE

在 Windows 操作系统中，为了从命令行转换 `.cproject` 文件，用户可以使用以下命令：

```
$IoTStudio_PATH/tools/Converter/TelinkRDS2IDE.exe Path/To/.cproject
```

在 Linux 操作系统中：

```
$IoTStudio_PATH/tools/Converter/TelinkRDS2IDE Path/To/.cproject
```

7.1.2 注意事项

默认情况下，转换后的项目（`.cproject`）会将工具链设置为 `Telink TLSR9 D25F GCC`，为了 D25F 会使用 `Telink RDS IDE` GCC 7.4 版本交叉编译器。

7.2 Telink IoTStudio 工程转换成 Cmake 工程

如果用户需要将 Telink IoTStudio 的工程转换为 CMake 工程，你可以使用下图中的工具。

首先，需要双击选中想要转换的工程，然后在 Telink Menu 中点击 `Telink IoTStudio to Cmake`.

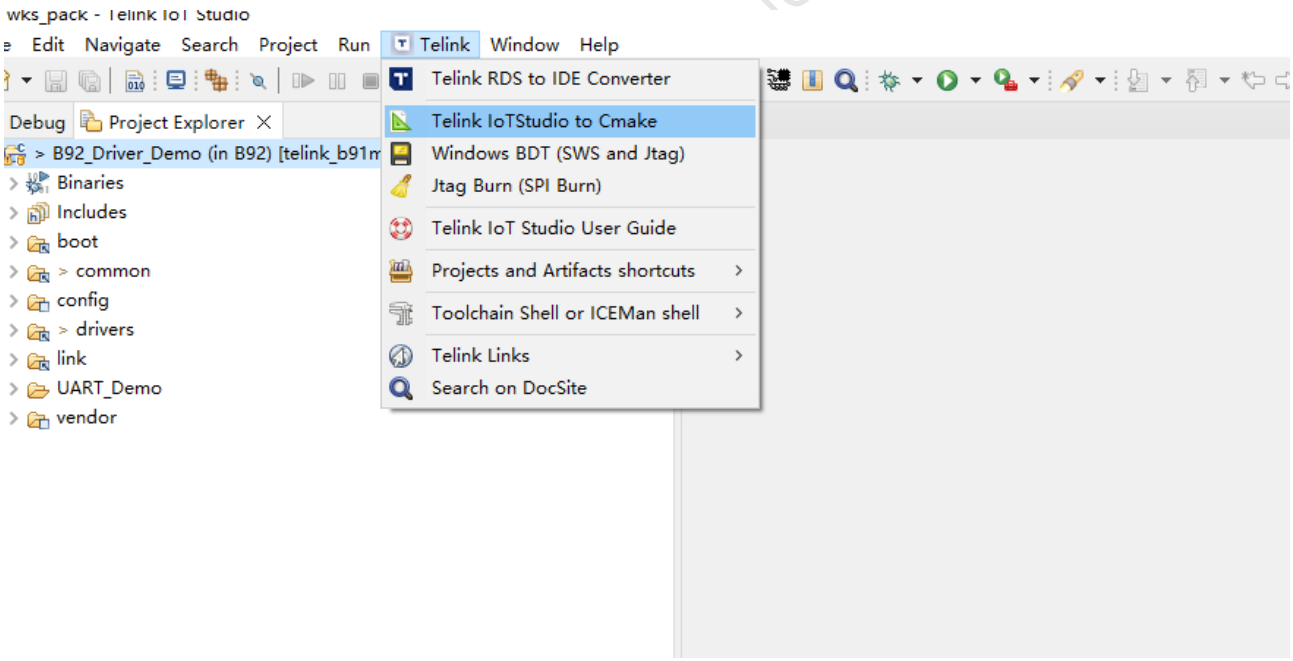

Figure 7.5: Telink IoTStuio to Cmake

你可以选择是否生成 VS Code 工程需要的 `settings.json` 文件. 点击 `Convert` 生成 `CMakelists.txt` 和 `settings.json` 文件, log 指明它们所在的位置。

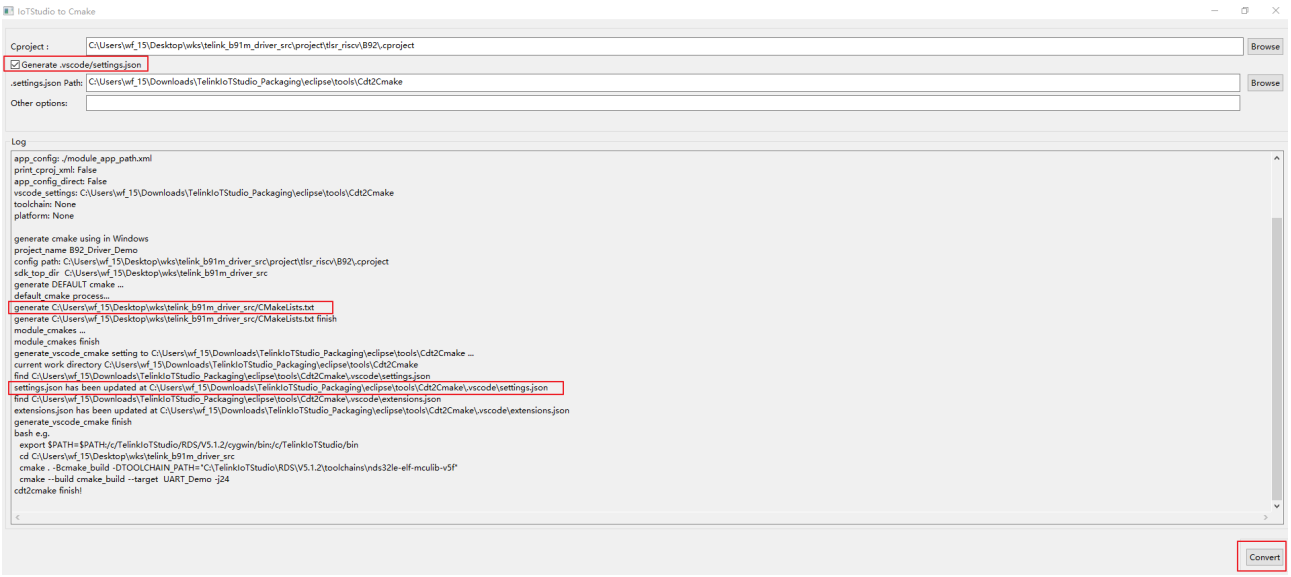

Figure 7.6: Telink IoStudio to Cmake UI

7.3 Jtag burn 菜单

`Jtag_Burn` 功能与 `Telink RDS IDE` Jtag_Burn 功能相似，都是利用 JTAG 接口将图像烧录到 TLSR9X 芯片。

除了 `Jtag_Burn` ，TLSR9 还可以用 `Burning EVK` 通过 `SWS` 协议，使用 `Windows BDT` 软件进行烧录，这是比较推荐的方法。

`SWS` 协议只需要一个 GPIO 就可以将二进制图像烧录到 TLSR8 和 TLSR9 芯片，而 JTAG 对于 TLSR9 芯片来说，会使用 2 个或 4 个 GPIO 引脚。

7.3.1 参数

Jtag_Burn 窗口打开后，用户可以填写或设置 Jtag_Burn 和 ICEMan 参数：

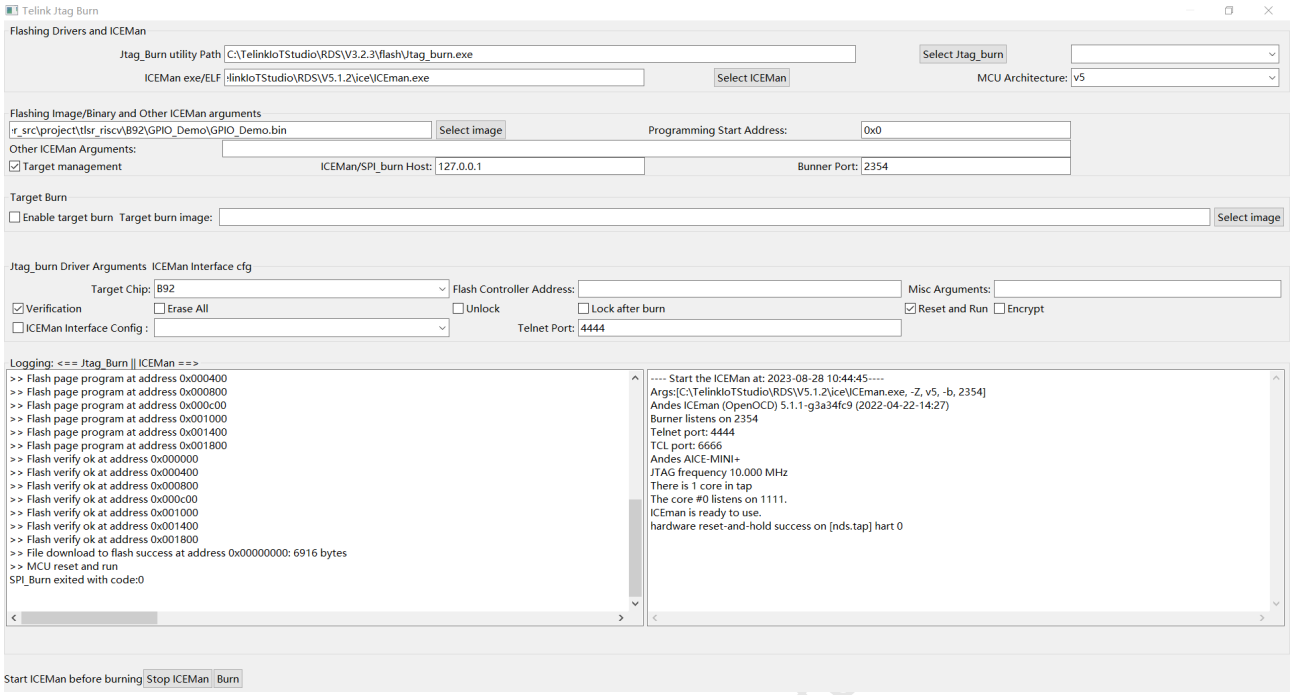

Figure 7.7: Jtag_Burn 窗口

可以在 `Jtag_Burn driver Arguments` 中设置 `Jtag_Burn` 参数.

7.3.2 在 Jtag_Burn 窗口中使用 ICEMan

ICEMan 是管理 TLSR9 JTAG ICE 的工具（支持 4 线和 2 线模式）。

用户可以通过查阅 `doc` 目录中的文档文件来获取更多的信息。

7.3.3 Jtag_Burn ELF 设置中的 ICEMan

用户可以在 `Flashing Drivers and ICEMan` 部分设置其他 ICEMan 和 Jtag_Burn ELF，设置/选择的值将被保存并在下次重新打开对话框时继续使用。

下面列出了标准的 ELF 文件名称以供参考：

- ICEMan：Windows 操作系统中的 `ICEMan.exe` ，以及 Linux 操作系统中的 `ICEMan` 。
- Jtag_Burn：Windows 操作系统中的 `Jtag_Burn.exe` ，以及 Linux 操作系统中的 `Jtag_Burn` 。

7.4 Jtag_Burn 烧录程序

烧录程序时，要注意配置 Jtag_Burn 的路径和芯片类型，然后点击 Start ICEman，ICEman 准备好后，确保 telnet port 和 burner port 与 ICEman 保持一致，再点击烧录，如下图：

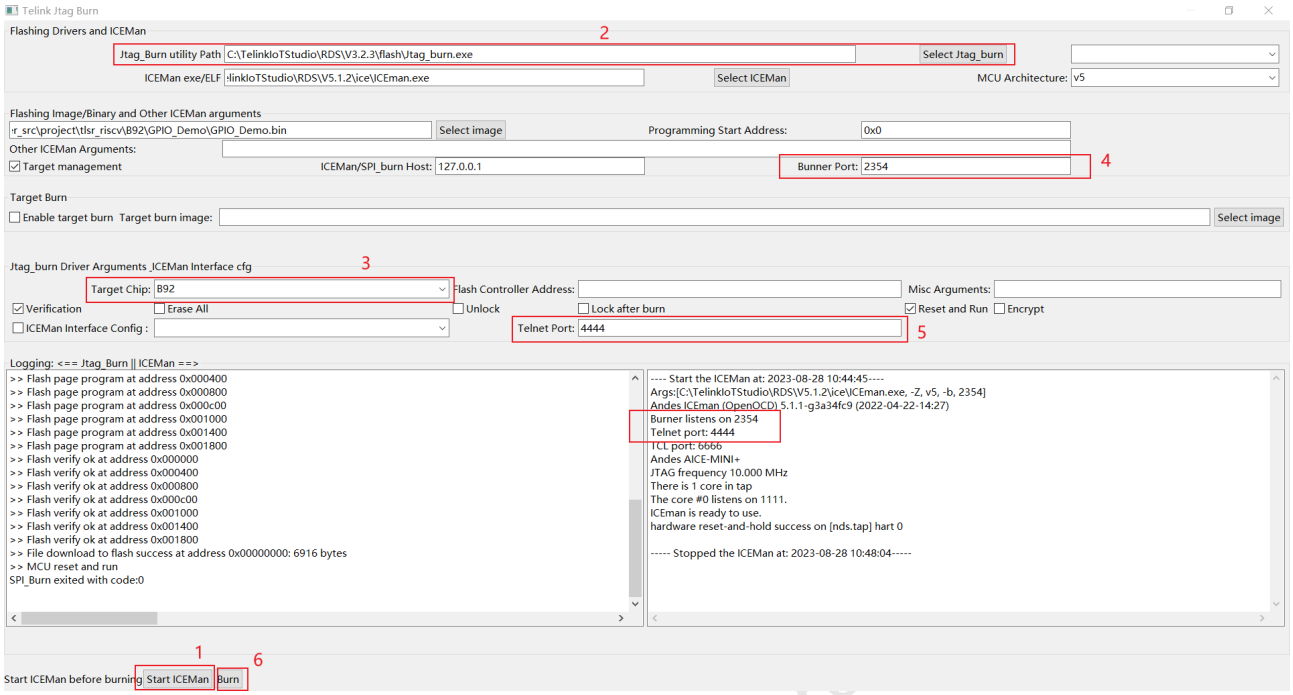

Figure 7.8: Jtag burn

其中，Jtag_Burn 位于 $IoTStudio/RDS/V3.2.3/flash/ 目录下，其帮助文档也在同一目录.

7.5 安全下载功能

若需要使用 Jtag_Burn 的安全下载功能，在 Misc Arguments 输入框中输入选项 –encrypt 即可

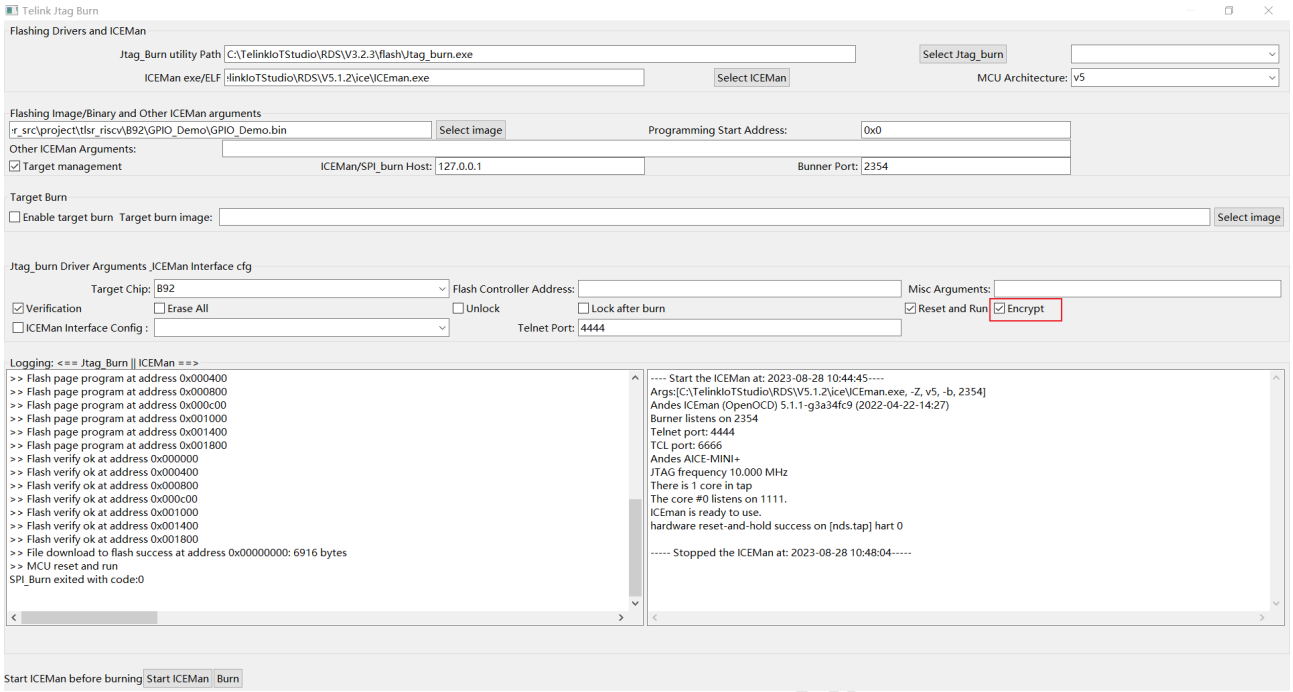

Figure 7.9: Jtag Burn Encrypt

7.6 Telink links

在 Telink links 的子菜单中，用户可以快速访问 Telink 相关网站以获取开发资源或信息：

- Telink forum 泰凌论坛
- Telink wiki 泰凌 wiki
- Telink official webpage 泰凌官网

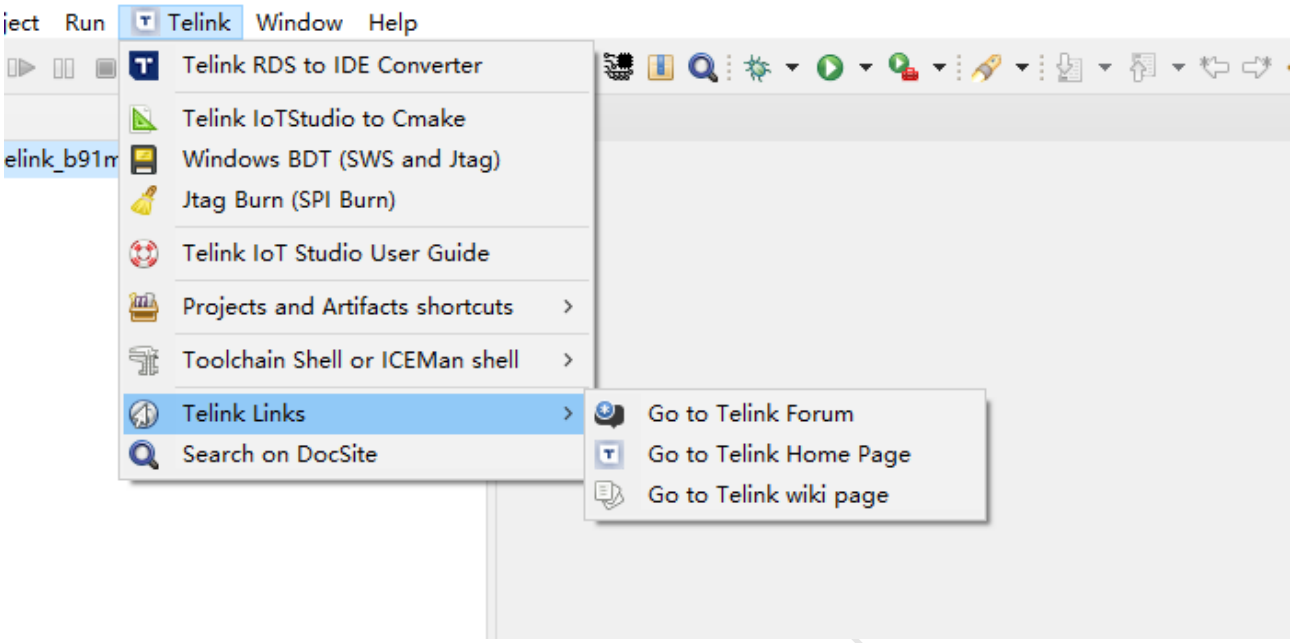

Figure 7.10: Telink Links

7.7 Telink tools launcher

此选项将会启动 `Telink tools launcher` ，如果显示以下对话框，则意味着此版本的 IoT Studio 尚未提供 `Telink tools` ，因此用户无法使用它。

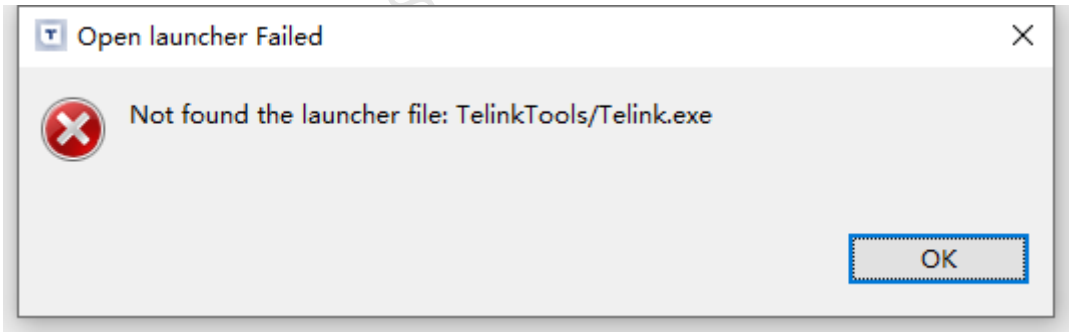

Figure 7.11: Telink tools launcher 启动失败

7.8 libusb 版本 BDT

`libusb version BDT` 是一种新的烧录工具，类似于 Windows 操作系统和 Linux 操作系统中的 `Windows BDT` ，但是这个工具需要使用新的固件来烧录 EVK。此工具可用于通过 SWS 将图像烧录到 TLSR8 和 TLSR9 芯片。该工具仍处于测试阶段。如果这个工具在某些 Windows 操作系统中不起作用，请改用 `Windows BDT` （在下面的 `Windows BDT` 部分中提到）。

用户可以启动此工具，然后点击 libUSB BDT 菜单上的"Help"来了解如何使用。

点击菜单项打开 BDT 的 libusb 版本，如果已经选择工程并且可以找到 artifact，那么 artifact bin 文件路径可以传递过来，所以用户无需手动复制 artifact 路径：

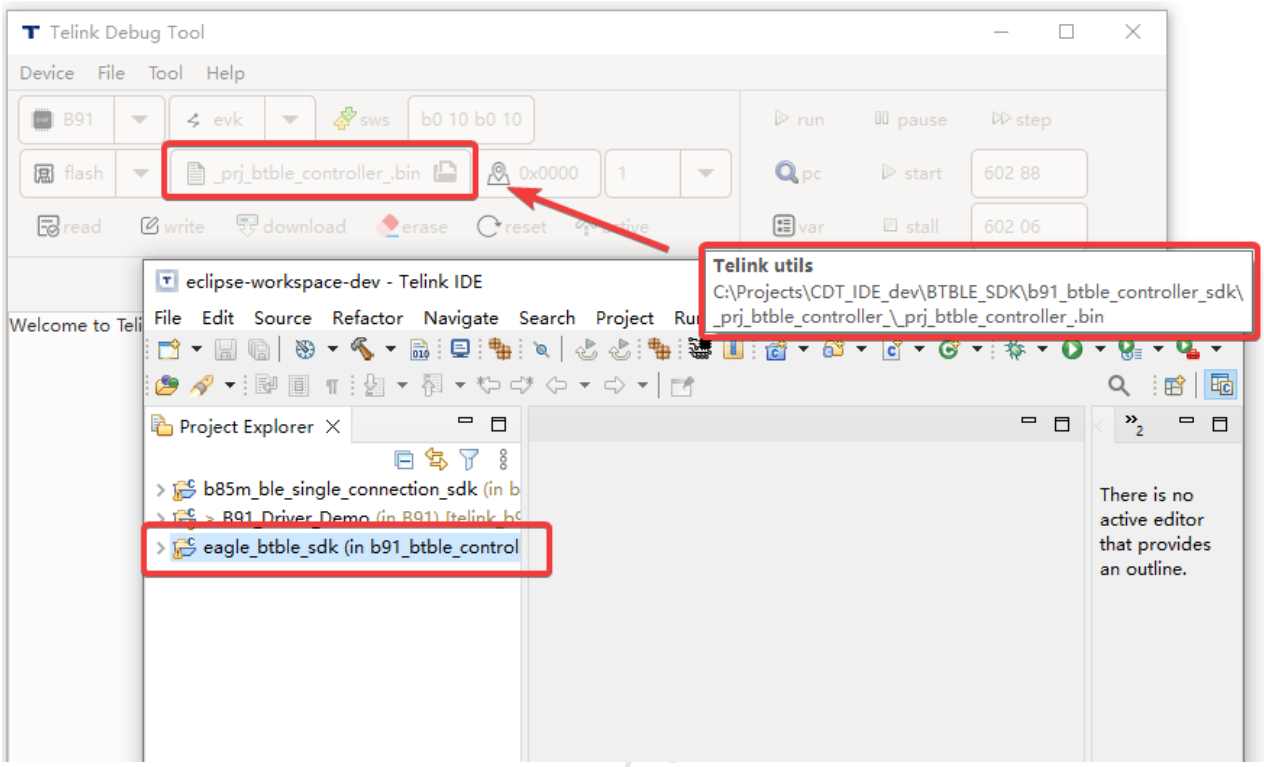

Figure 7.12: 打开已有路径信息的 libUSB BDT

如果未选择任何工程，则会跳出一个错误对话框，用户可以将其关闭，libusb BDT 程序同样可以正常启动：

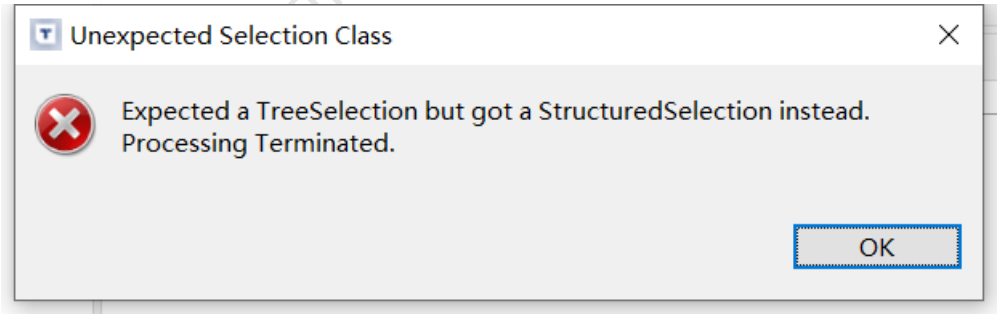

Figure 7.13: 打开没有路径信息的 libUSB BDT

7.9 Toolchain shell or ICEMan shell 菜单项

这是一个子菜单，它有好几个菜单项可以启动 Toolchain 控制台或 ICEMan 控制台，用户可以在这些 shell 中输入命令（例如：`riscv-elf-gcc` 或 `ICEMan`）。

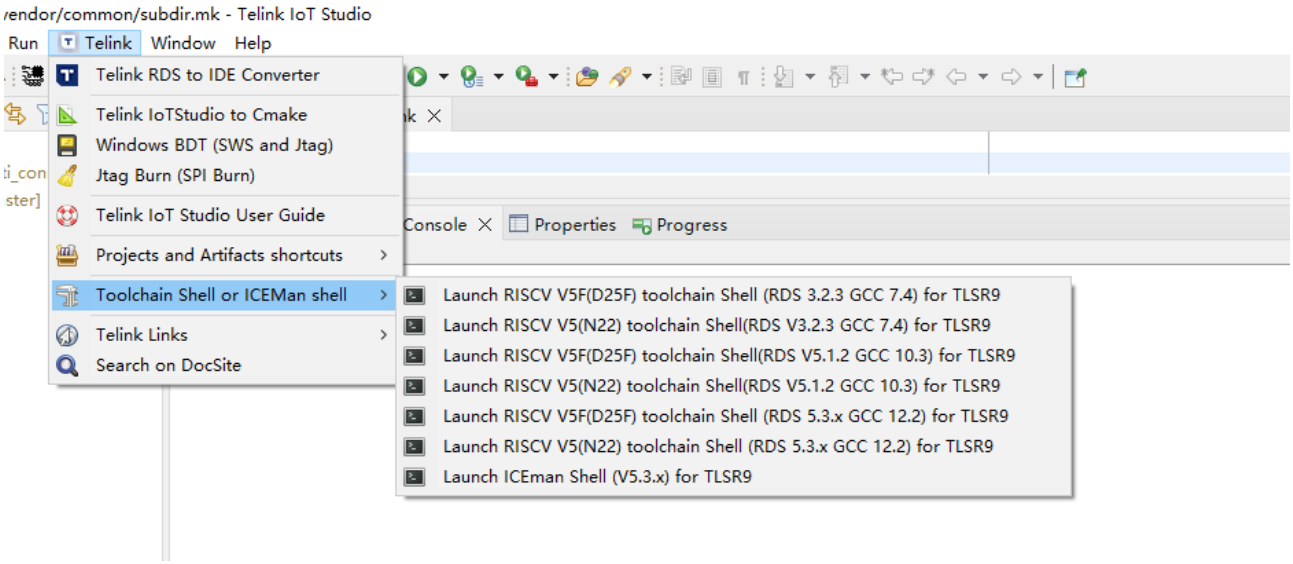

Figure 7.14: Shells

7.10 Windows BDT

`Windows BDT (SWS and JTAG)` 菜单项会打开一个单独的程序 `Windows BDT` 。用户可以使用此工具烧录或调试 TLSR8 和 TLSR9 芯片。

可以参考这个 wiki 页面 了解如何使用。

7.11 Linux tcdb for TLSR8（仅适用于 Linux 操作系统）

`Linux tcdb for TLSR8` 菜单项有 2 个子菜单项， `tcdb` 将打开 `gnome-terminal` 以使用 tcdb。用户可以使用 `tcdb` 工具来烧录 TLSR8 芯片。

```
Terminal
File  Edit  View  Search  Terminal  Help
[1.hexiongjun-5590.09:53:06 ~/Projects/CDT_Telink_IDE/TelinkIDE/eclipse/tools/tcdb]$ tcdb help

usage
    tcdb type [wf|rf|wc|rc|rst|stall|sws|help|activate [adr [dat]]] -i(o) file_input(file_output) -sbeu
      type: chip type, eg: 8266/8267/8258/8278/9518;
      wf:   write flash;
      rf:   read flash;
      wc:   write device;
      rc:   read device
      rst:  reset MCU;
      stall: stall MCU;
      sws: sync according to chip type;
      activate: activate the MCU;
      help:  display this help and exit;
      -v, version:  display version info;
      -s:   specified size
      -i:   file input
      -o:   file output
      -b:   binary file format
      -e:   erase flash(by sector)
      -u:   usb mode

tcdb
    when cmd argument not specified, help message will be displayed
tcdb 8267  rc 0  -s 16
    read 16 bytes from sram address 0
tcdb 8258  rc 40000  -s 16
    read 16 bytes from sram address 40000
tcdb 8258  wc 40000 01020304 -s 4
    write 4 bytes to sram 40000
```

Figure 7.15: Linux tcdb 控制台

`tcdb user guide` 会打开 PDF 格式的用户指南以供用户参考。

7.12 Open artifact path

工具栏上有一个图标，用来打开 artifact 目录。

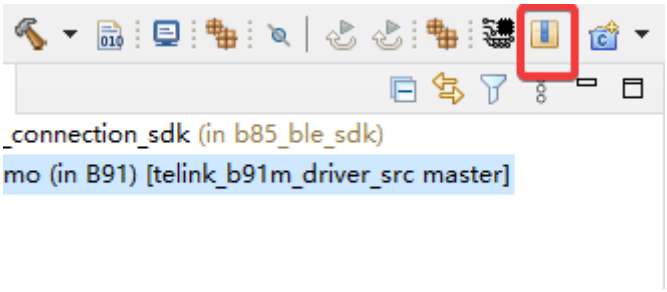

Figure 7.16: 打开 artifact 路径

如果工程没有配置正确（`.cproject` 文件），这个图标无法工作，这个情况在许多之前的 TLSR8 SDK 中都会发生。

而大多数 TLSR9 SDK 工程都能够如期工作。

7.12.1 注意事项

在点击图标之前必须选择工程或工程文件，否则它将无法工作。

7.13 Copy artifact path

Telink IoT studio 增加了一个方便的菜单选项，可以帮助用户复制 artifact 路径。

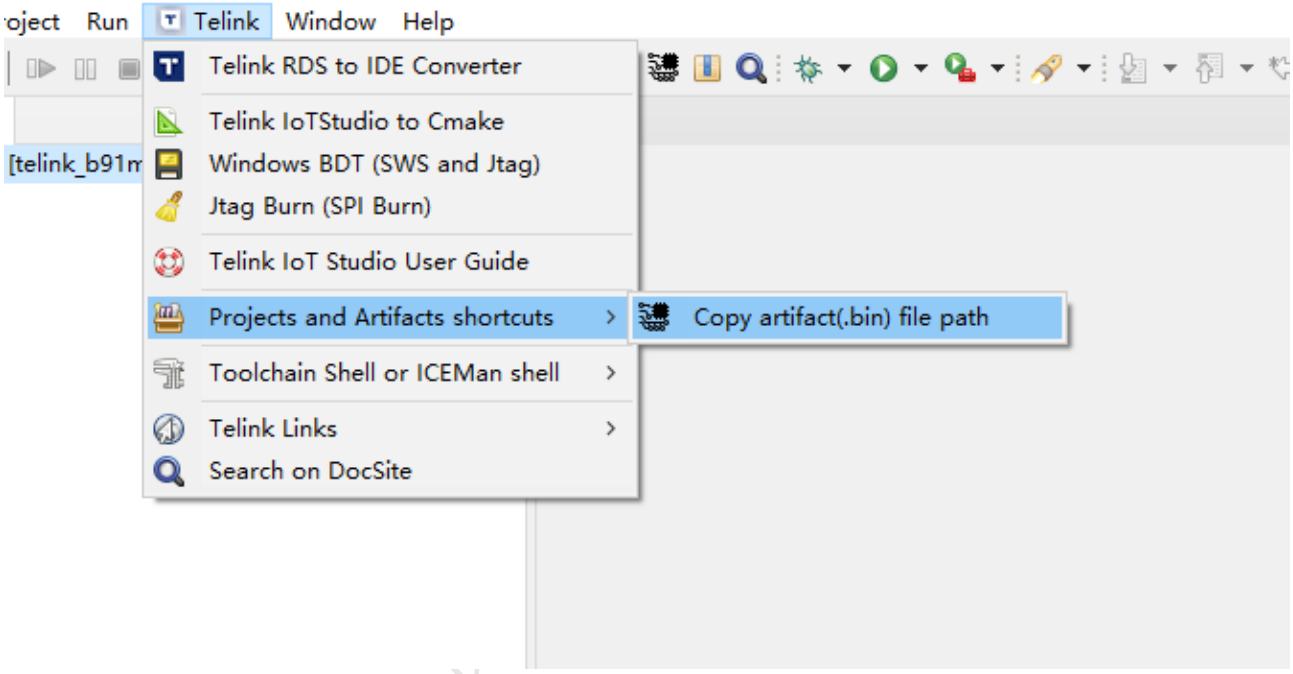

Figure 7.17: 复制 artifact 路径

用户可以点击上图中箭头指向的这个图标，将 artifact 路径复制到系统剪贴板上。

7.13.1 注意事项

在点击图标之前必须选择工程或工程文件，否则它将无法工作。

7.13.2 其他位置

`Copy artifact path` 这个选项会显示在工具栏和右键弹出的菜单中，如下所示：

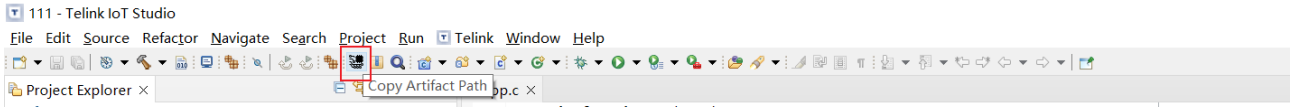

Figure 7.18: 在工具栏中的 Copy artifact path

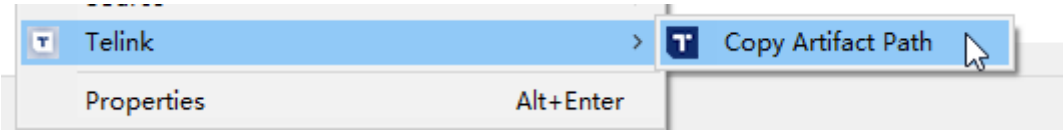

Figure 7.19: 在菜单中的 Copy artifact path

7.14 Search on DocSite

此工具可以用来在 Telink 文档中心网站中搜索特定的关键词，在工程文件中双击选中某关键词后，点击 `Search on DocSite` ，即可跳转至文档中心搜索页面。

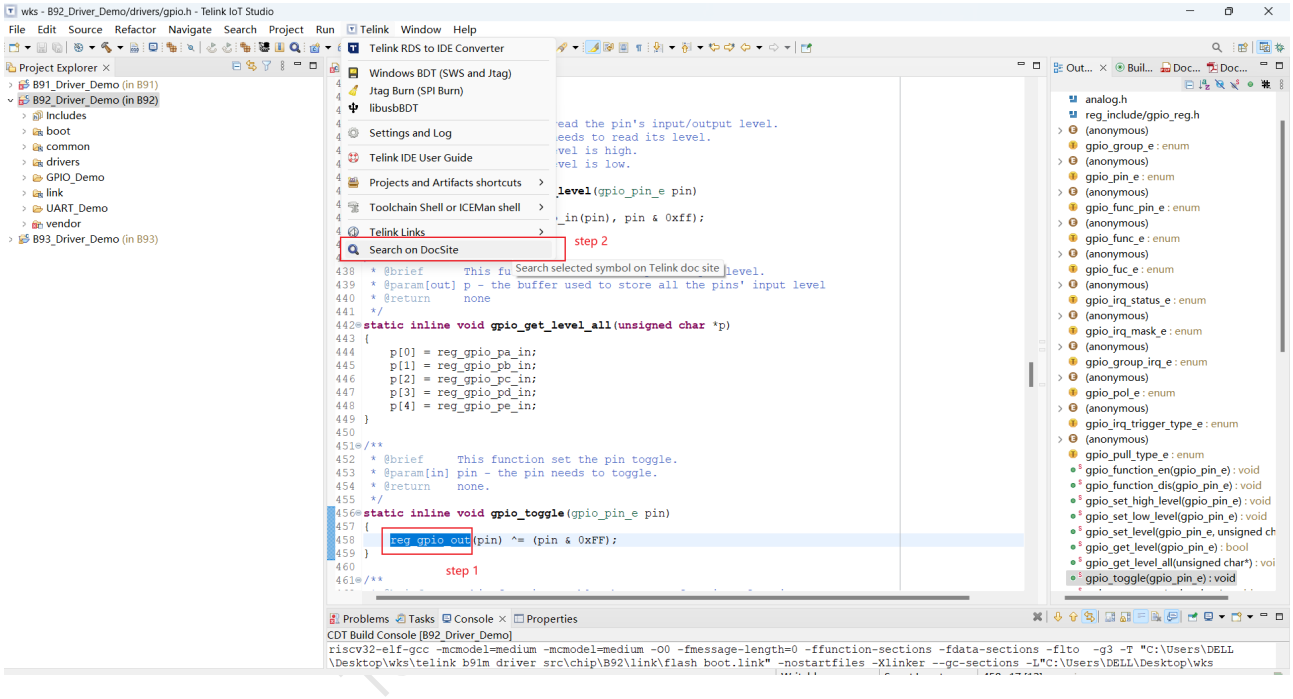

Figure 7.20: 关键词搜索

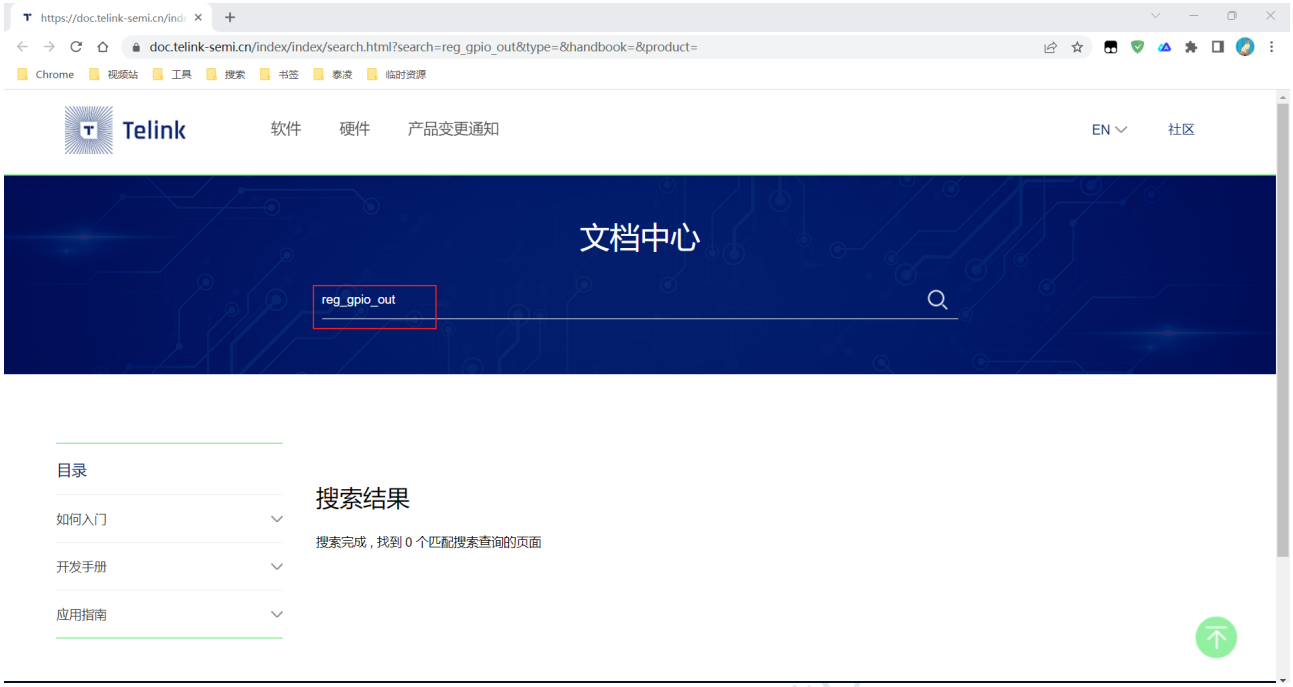

Figure 7.21: 文档中心

8 常见的编译错误

8.1 找不到 link 脚本文件

当 link 脚本文件的路径为相关格式时，这种类型的错误就会出现。它在 `Telink RDS IDE` 中不会报错，但在 `Telink IoT studio` 中会显示错误。

错误日志的提示如下：

```
Building target: AES_Demo.elf
Invoking: GNU RISC-V Cross C Linker
riscv32-elf-gcc -mcmodel=medium -mcmodel=medium -O2 -fmessage-length=0 -ffunction-sections -fdata-sections -flto  -g3 -T ../link/flash_boot.link -nostartfiles -Xlinker --gc-sections -L"C:\Projects
\CDT_IDE_dev\telink_b91m_driver_src\chip\B91\drivers\lib" -Wl,-Map,"AES_Demo.map" -O2  -fpack-struct -fshort-enums -g3 -mcpu=d25f -ffunction-sections -fdata-sections -mext-dsp -mabi=ilp32f -
fmessage-length=0 -fomit-frame-pointer -fno-strict-aliasing -fshort-wchar -fuse-ld=bfd -Wno-gnu-zero-variadic-macro-arguments -o "AES_Demo.elf"  ./vendor/common/flash/puya_common.o
./vendor/common/flash/puya_flash_scratch.o ./vendor/common/flash/puya_flash_trim.o  ./vendor/common/calibration/calibration.o  ./vendor/common/plic_isr.o ./vendor/common/printf.o
./vendor/AES_Demo/app.o ./vendor/AES_Demo/main.o  ./drivers/lib/src/pke/eccp_curve.o ./drivers/lib/src/pke/ecdh.o ./drivers/lib/src/pke/ecdsa.o ./drivers/lib/src/pke/pke.o
./drivers/lib/src/pke/pke_common.o ./drivers/lib/src/pke/x25519.o  ./drivers/lib/src/emi.o ./drivers/lib/src/plic.o ./drivers/lib/src/pm.o ./drivers/lib/src/rf.o ./drivers/lib/src/swire.o
./drivers/lib/src/sys.o ./drivers/lib/src/trng.o  ./drivers/flash/flash_mid146085.o ./drivers/flash/flash_mid156085.o ./drivers/flash/flash_mid166085.o  ./drivers/adc.o ./drivers/aes.o
./drivers/analog.o ./drivers/audio.o ./drivers/clock.o ./drivers/flash.o ./drivers/gpio.o ./drivers/i2c.o ./drivers/lpc.o ./drivers/mdec.o ./drivers/npe.o ./drivers/pwm.o ./drivers/s7816.o
./drivers/spi.o ./drivers/stimer.o ./drivers/timer.o ./drivers/uart.o ./drivers/usbhw.o  ./common/usb_dbg/myudb_usb.o ./common/usb_dbg/myudb_usbdesc.o  ./common/bt_debug/dbgport.o
./boot/cstartup_flash.o ./boot/cstartup_ram.o   -ldriver -laxon_driver_lib
/cygdrive/c/IDERelease/20220620_V1.0/eclipse/RDS/V3.2.3/toolchains/nds32le-elf-mculib-v5f/bin/../lib/gcc/riscv32-elf/7.4.0/../../../../riscv32-elf/bin/ld.bfd: cannot open linker script file
../link/flash_boot.link: No such file or directory
collect2: error: ld returned 1 exit status
make: *** [makefile:71: AES_Demo.elf] Error 1
"make -j4 all" terminated with exit code 2. Build might be incomplete.

10:54:59 Build Failed. 1 errors, 0 warnings. (took 31s.427ms)
```

Figure 8.1: Link 错误

要解决上述问题，用户可以打开工程属性，然后找到 `Tool Settings` 选项中的 `General` 页面，如下图所示，然后双击 link 脚本文件：

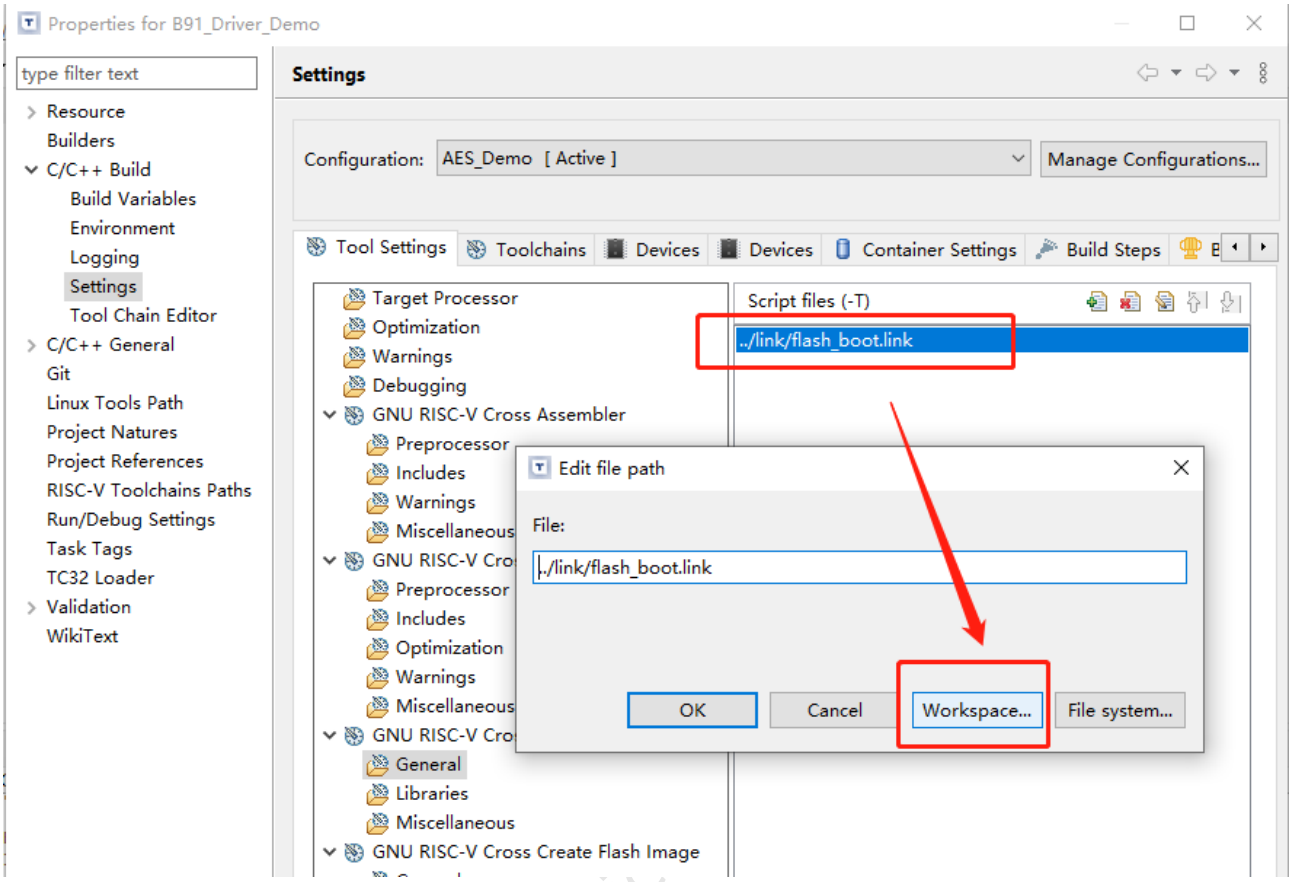

Figure 8.2: Link 文件路径

在打开的对话框中，从 `workspace` 中选择 link 文件：

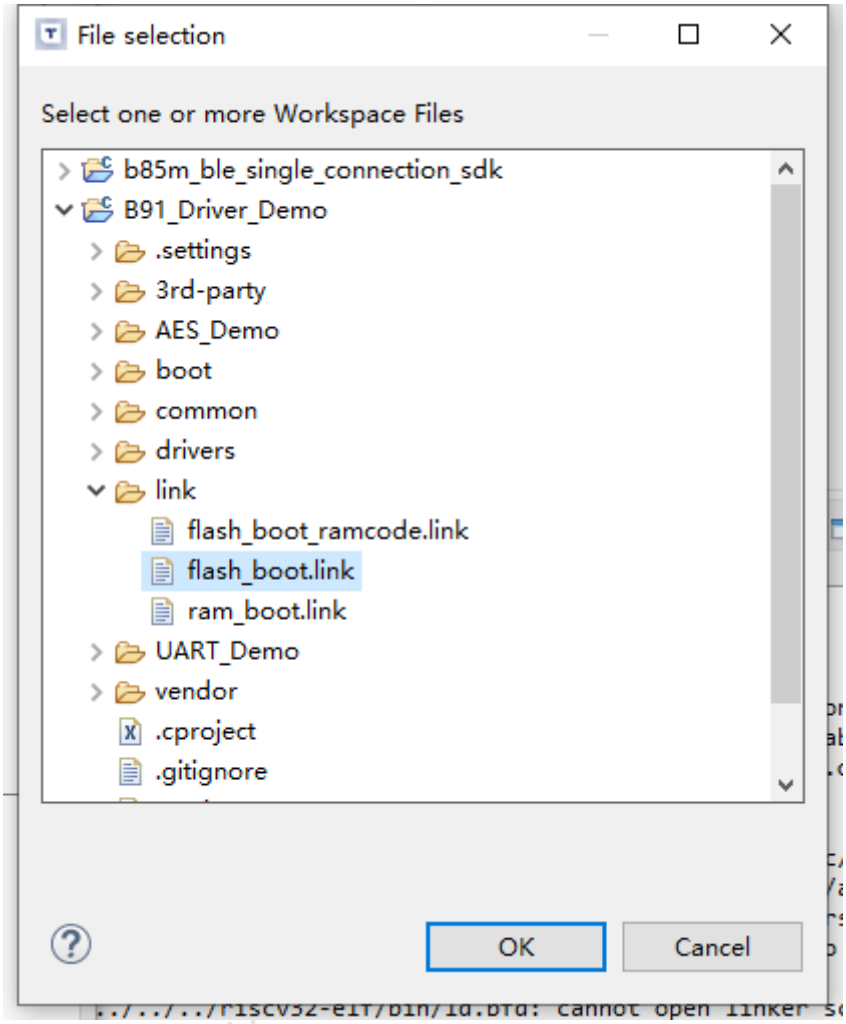

Figure 8.3: 选择 link 文件

点击 `OK` 和 `apply and close` 按钮，然后重新构建工程以检查结果。

9 其他附加的插件或功能

9.1 Telink Formatter

该插件用于使用快捷方式格式化代码/文件或使用 clang-format 保存文件。

9.1.1 配置 Telink formatter

要在 formatter 中使用它，用户应该将 `Telink Formatter` 设置为默认格式化程序。从菜单 `Windows -> Preference -> C/C++ -> Code style -> Formatter` 中打开设置，在 `Code Formatter:` 列表中选择 `TelinkFormatter`，然后单击 `Apply and Close` 按钮：

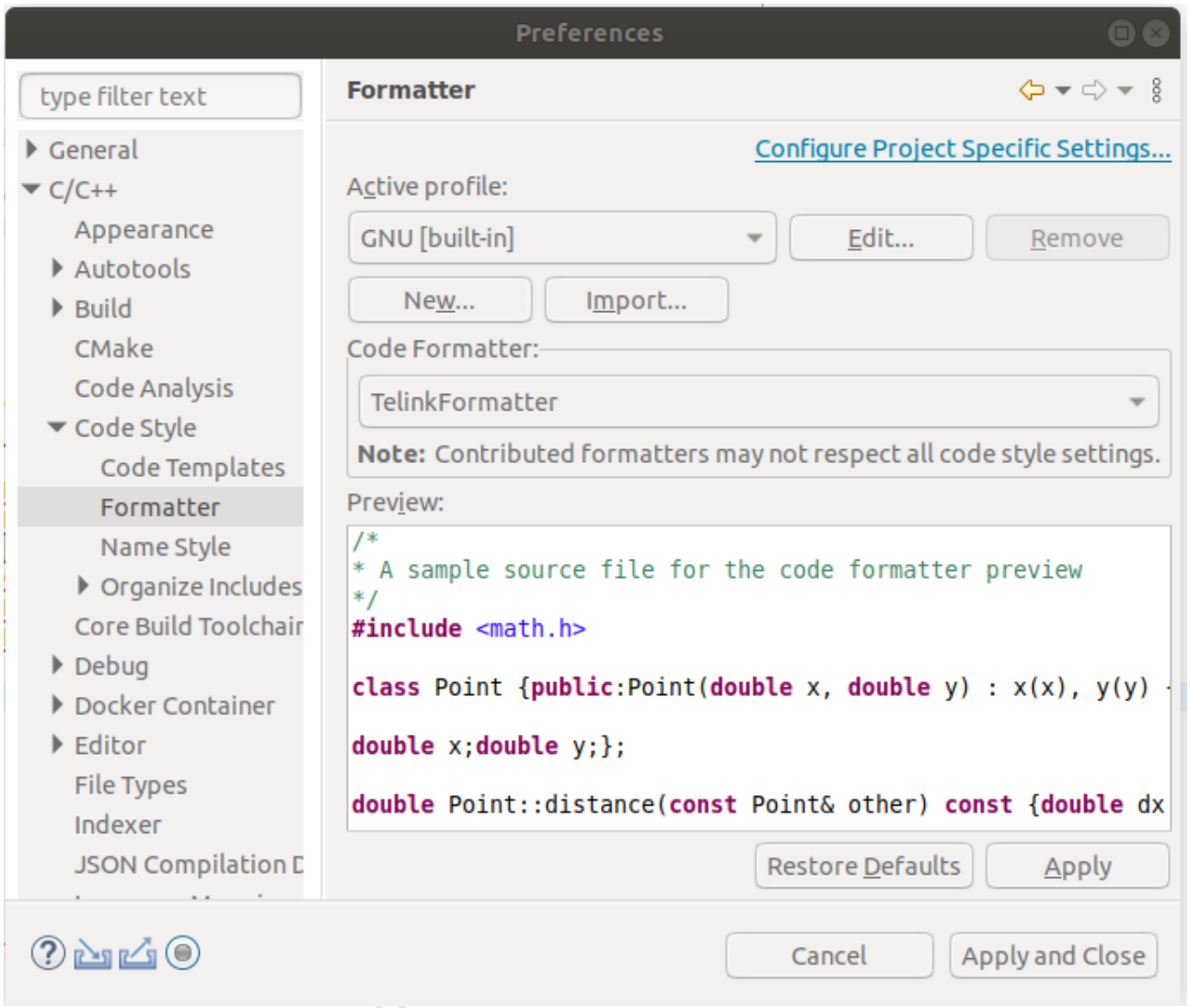

Figure 9.1: Formatter setting

然后，用户需要在 `Windows` -> `Preference` -> `C/C++` -> `TelinkFormatter` 中，将格式化风格设置为 `Telink`

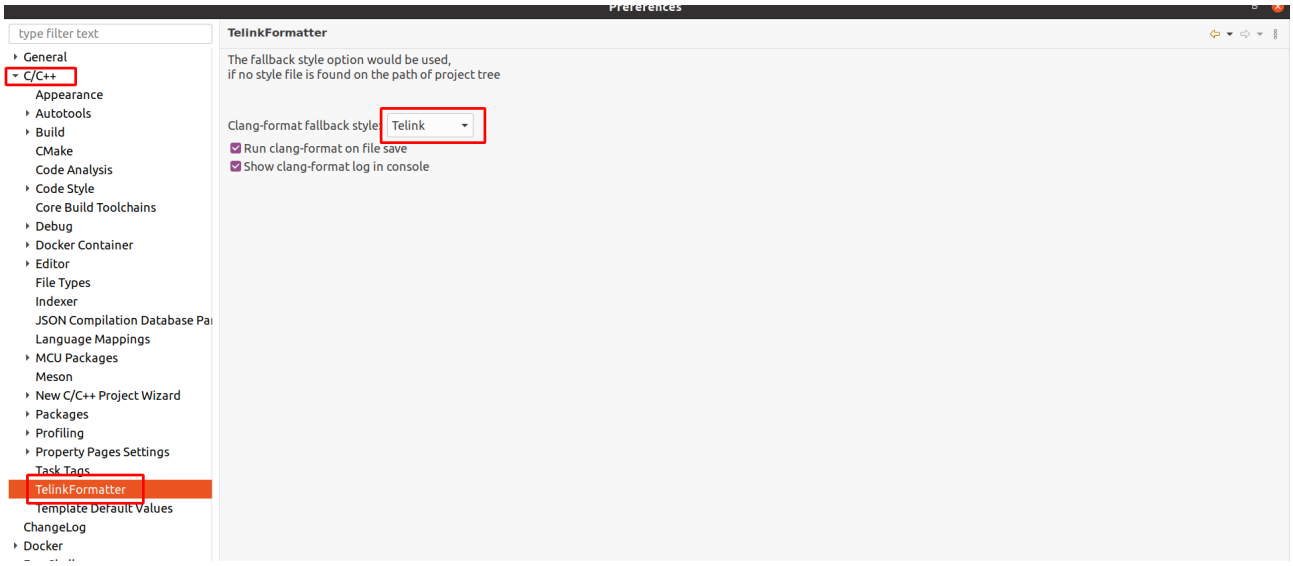

Figure 9.2: Telinkformatter Settings

随后用户可以在任何打开的源代码文件编辑器上按快捷键（默认为 `Ctrl+Shift+F` ），会出现一个对话框，点击相应的按钮和选项进行确认：

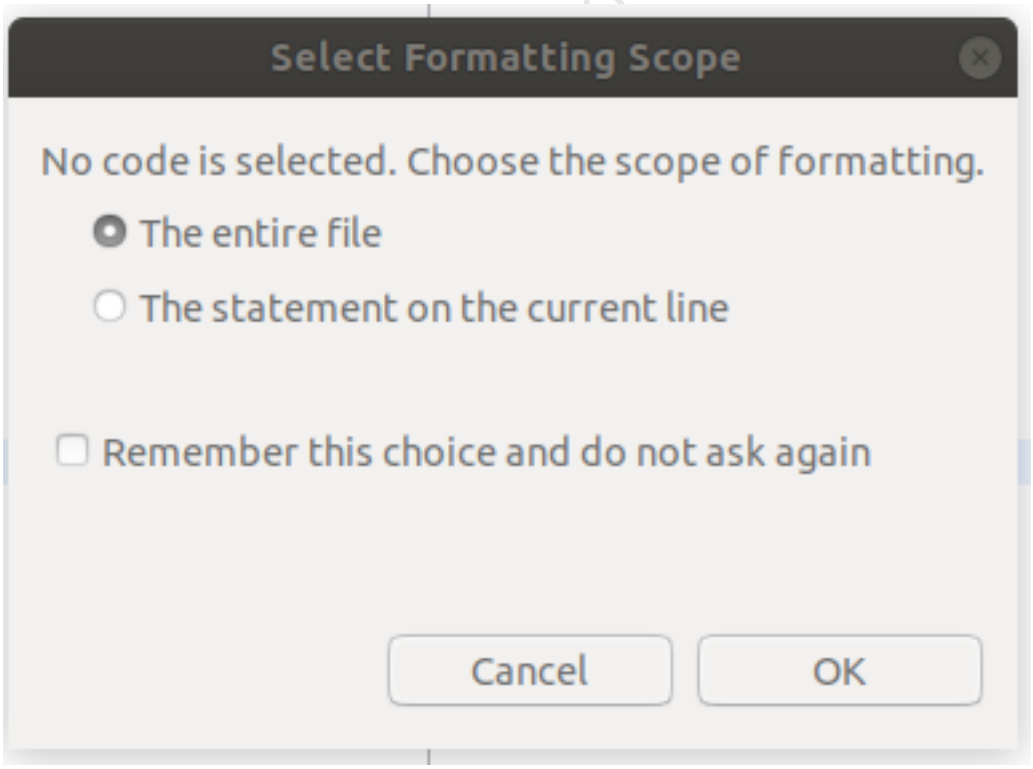

Figure 9.3: Formatter hint

当 `Telink Formatter` 格式化代码/文件后，用户可以在 `Telink Formatter` 输出控制台上查看日志以了解详细信息（请注意，应在 `TelinkFormatter` 首选项对话框中启用日志功能，请查看以下指南）：

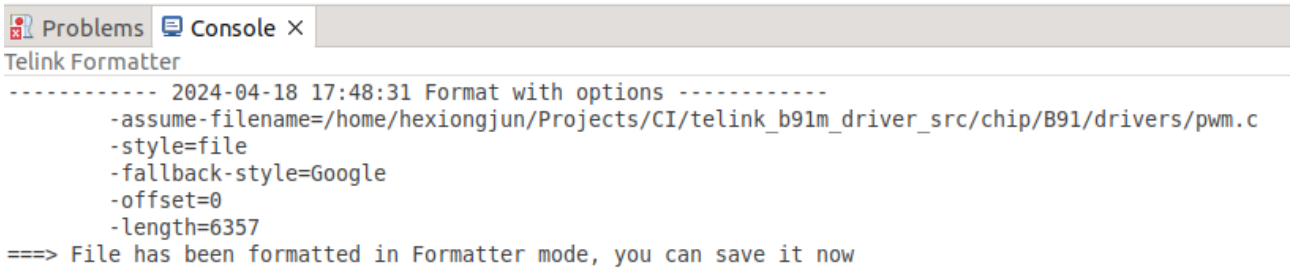

Figure 9.4: Formatter console output after formmtting

9.1.2 保存文本时触发（推荐使用）

格式化程序可以在保存一个或多个文件时触发，这需要在 `TelinkFormat` 设置中选择功能（通过菜单 `Windows -> Preference -> C/C++ -> TelinkFormatter` ）：

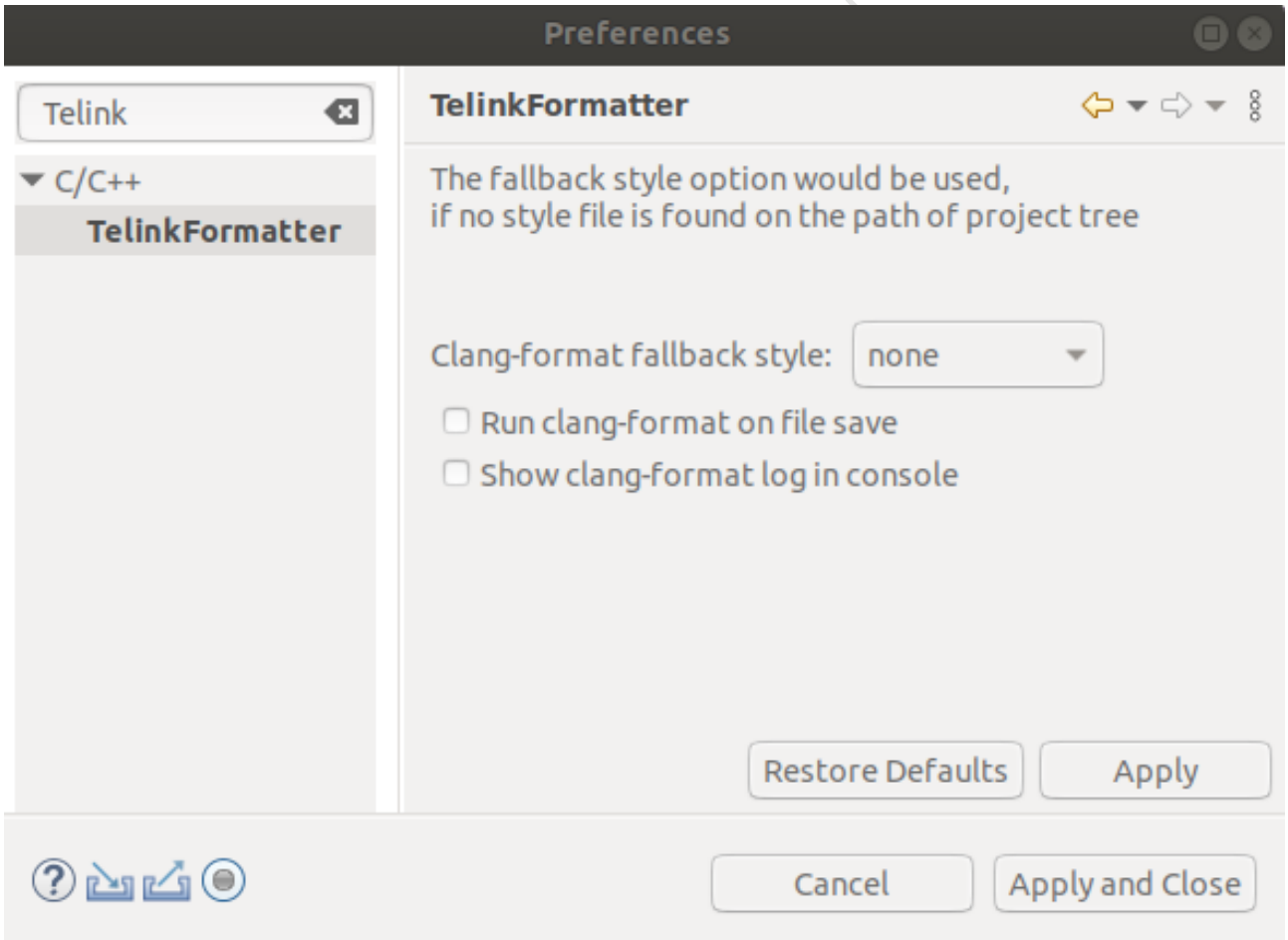

Figure 9.5: Telink Formatter Preference

当 `Telink Formatter` 格式化代码/文件后，如果在设置中启用了日志，用户可以在 `Telink Formatter` 输出控制台上查看日志以了解详细信息：

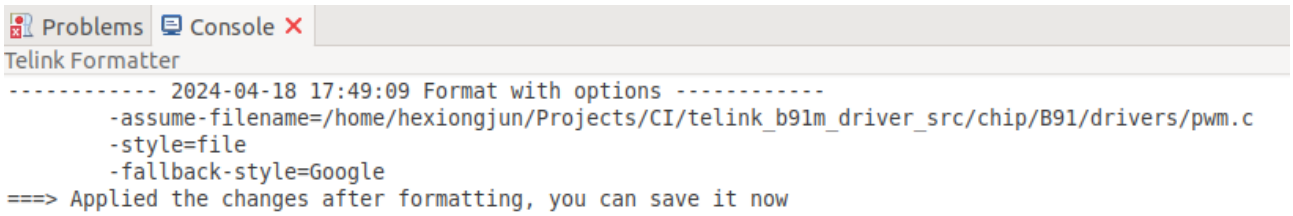

Figure 9.6: Formatter console output after saving

9.1.3 已知的问题

9.1.3.1 首次使用时 `Save` 按钮被禁用

首次修改文件时，"保存"按钮处于禁用状态。使用"CTRL+S"（或"文件"->"保存"，或使用"全部保存"菜单项/按钮）保存后，再次修改此文件，它将变为启用状态。

9.2 Easy Shell

这个插件允许从导航树或编辑器视图的弹出菜单中打开一个 shell 窗口或文件管理器。此外，它可以在 shell 中运行选定文件、复制文件或目录路径、运行用户定义的外部工具。

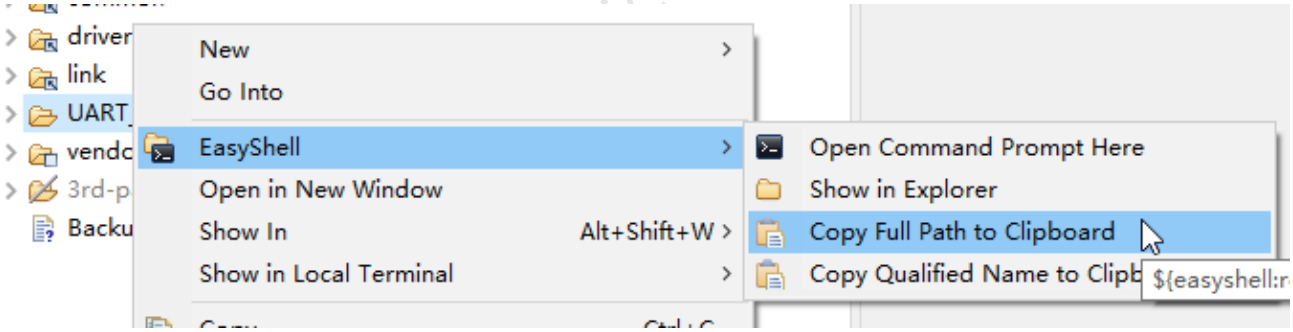

Figure 9.7: Easy Shell

9.3 ECalculator

一个多功能计算器，包括：

- 显示 n 进制数字计算
- 标准、科学的和三角函数计算
- 输入合理的算术表达式

用户可以从菜单中打开：`Window` -> `Show view` -> `other...`，然后输入 `Ecalculator` 搜索并打开：

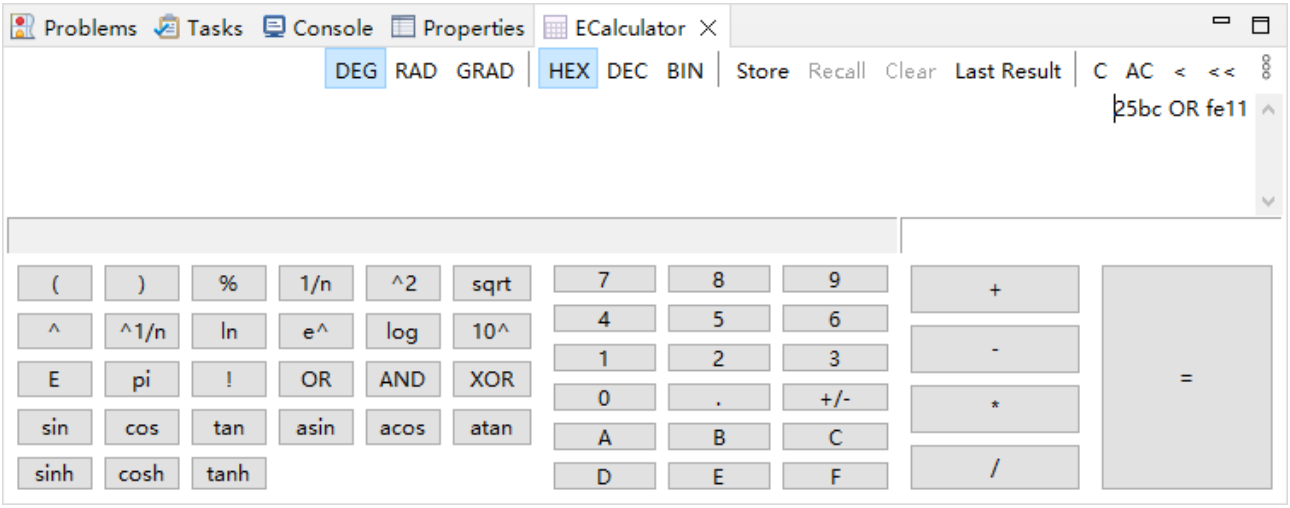

Figure 9.8: ECalculator

9.4 Eclipse 中的终端

该终端可用于输入 git 命令和串口调试。

用户可以从菜单中打开：`Window` -> `Show View` -> `Terminal`：

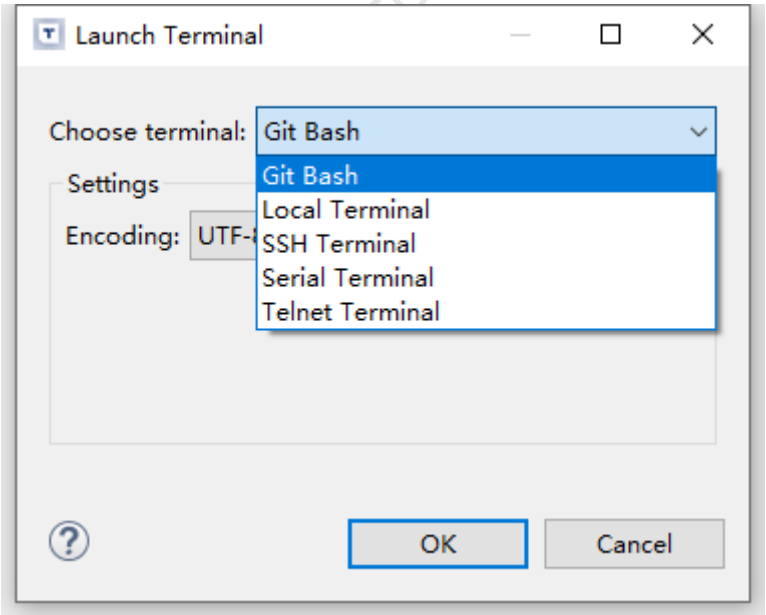

Figure 9.9: 选择终端

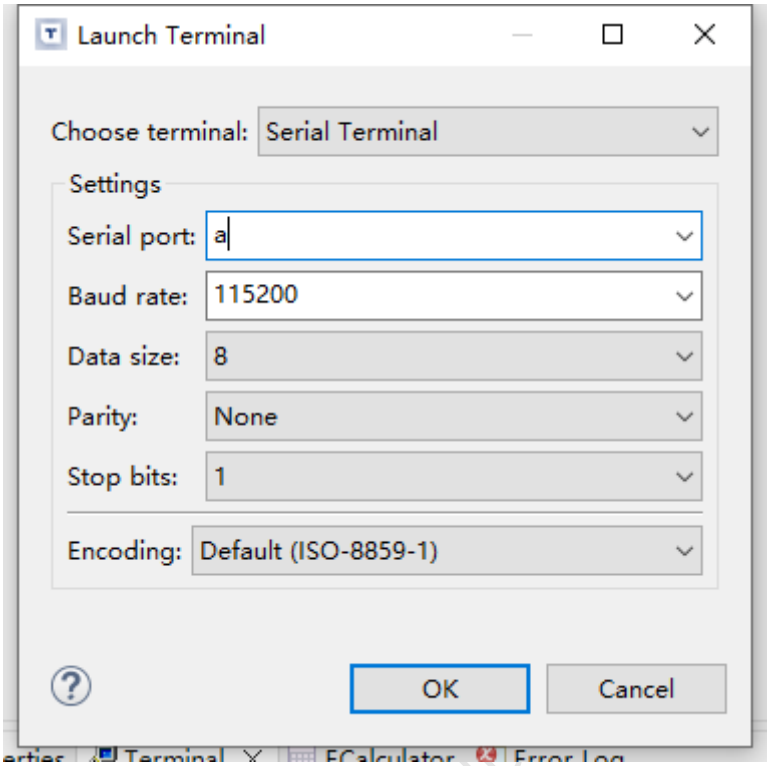

Figure 9.10: 选择串行终端

如果你的串口调试日志输出频率非常快，请使用其他专用工具代替。

9.5 二进制浏览器

这个插件用来打开二进制文件。

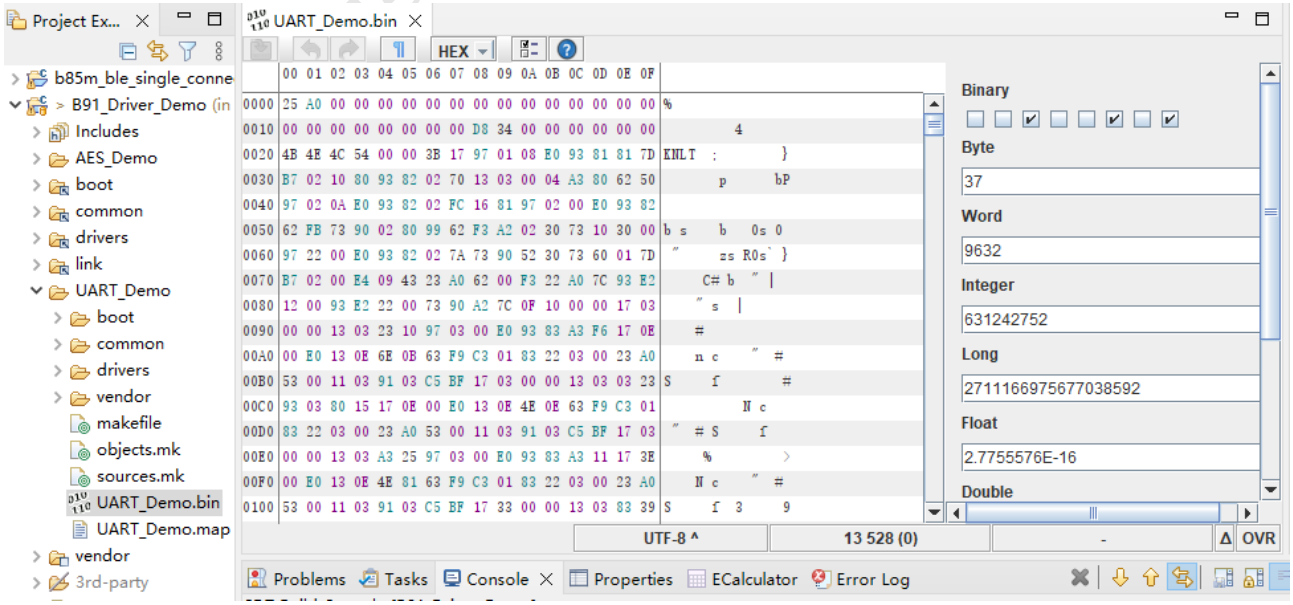

Figure 9.11: 二进制浏览窗口

用户可以双击一个二进制文件，在 IoT Studio 中打开并查看。

10 FAQ

10.1 编译代码时出现 `Error 127`

这个错误会在控制台窗口显示 `Error 127` ：

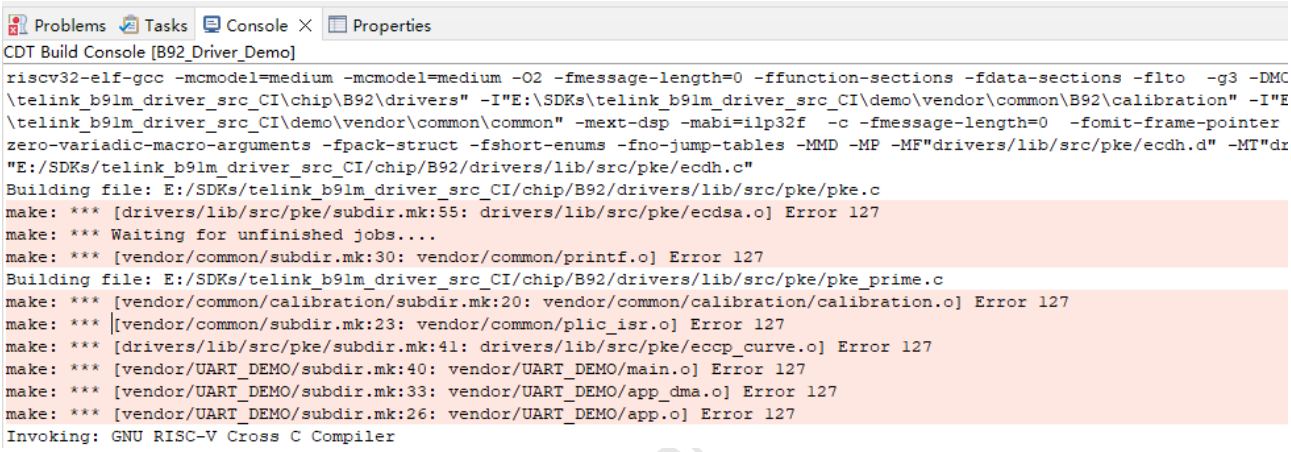

Figure 10.1: Error 127

这个问题是由于构建工具版本的不正确设置引起的，设置正确即可解决这个问题。

10.2 在路径中找不到设备或在构建时没有能够构建的内容

这些问题通常是由于将 Telink RDS IDE 格式的工程直接导入到 Telink IoT studio 而没有进行转换造成的。

要解决这个问题，请先删除该工程并在导入前将其转换为 Telink IoT studio。

10.3 Orphaned configuration 和设置选项卡上没有选项

发生这种情况时，工程属性会出现如下情况：

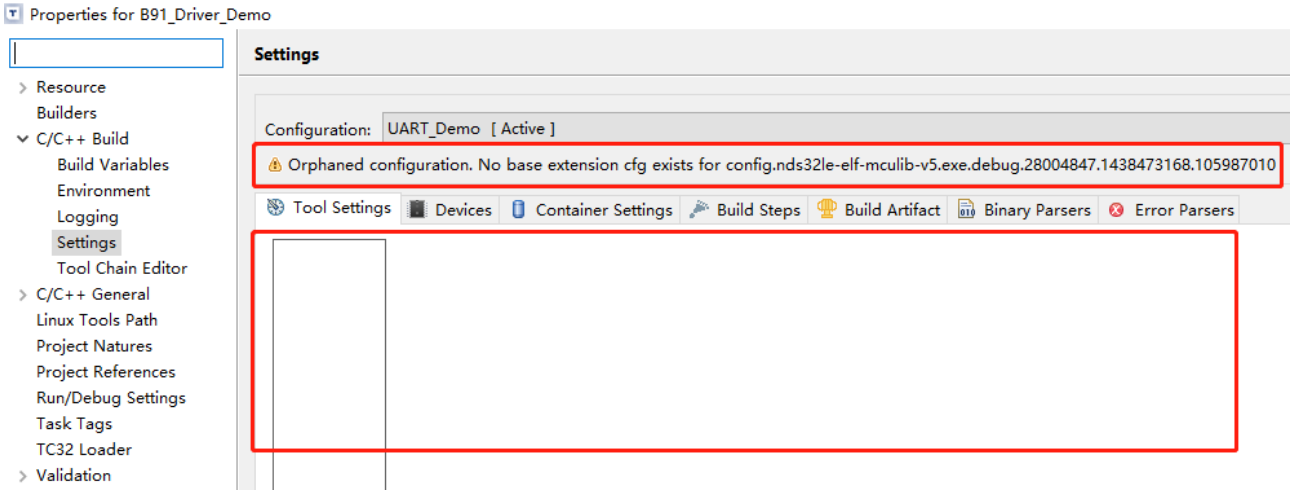

Figure 10.2: Orphaned configuration

此问题的原因与前面的 FAQ 相同，在导入之前请将其转换为 Telink RDS 格式工程。

10.4 如何验证 TLSR9 当前使用的工具链 gcc 版本

用户可以在工程属性中设置以下命令来作为 Post-build 命令：

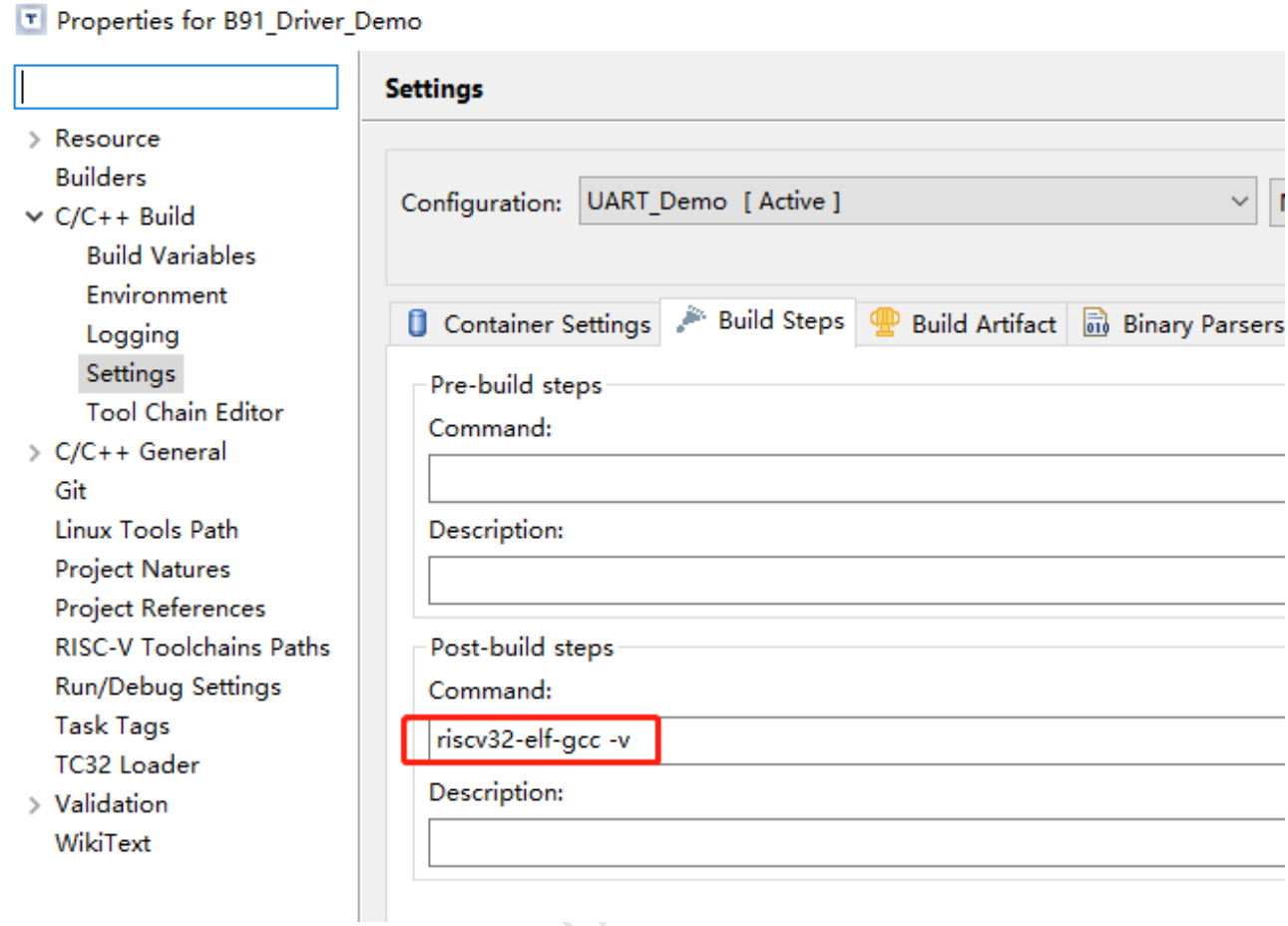

Figure 10.3: Post-build 命令中的 GCC 版本设置

然后构建项目，控制台将输出 GCC 版本：

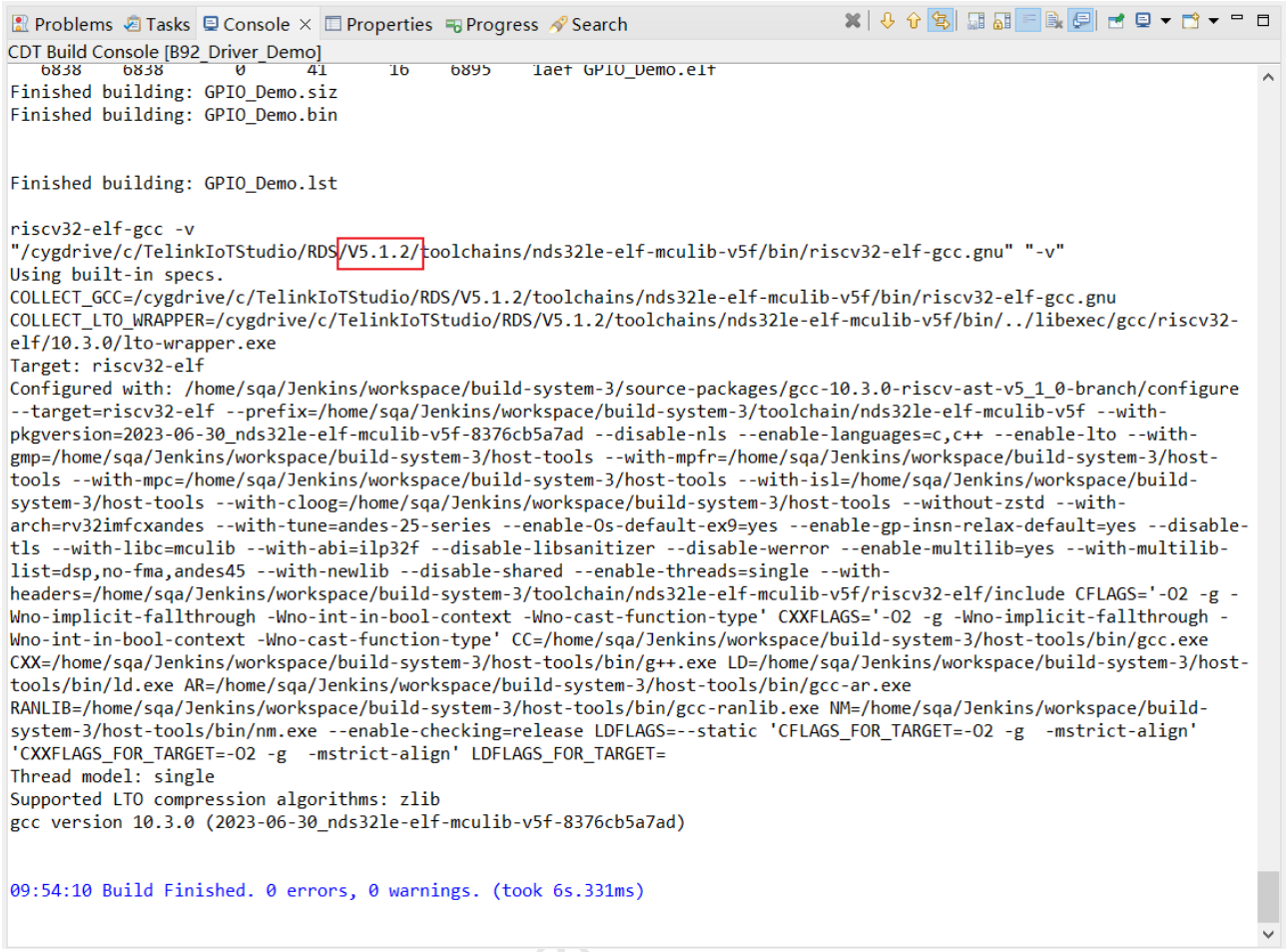

Figure 10.4: 验证 GCC 版本

10.5 如何验证当前使用的 `make tools`

用户可以在工程属性中设置以下命令（`echo "${PATH}"`）来作为 Post-build 命令。

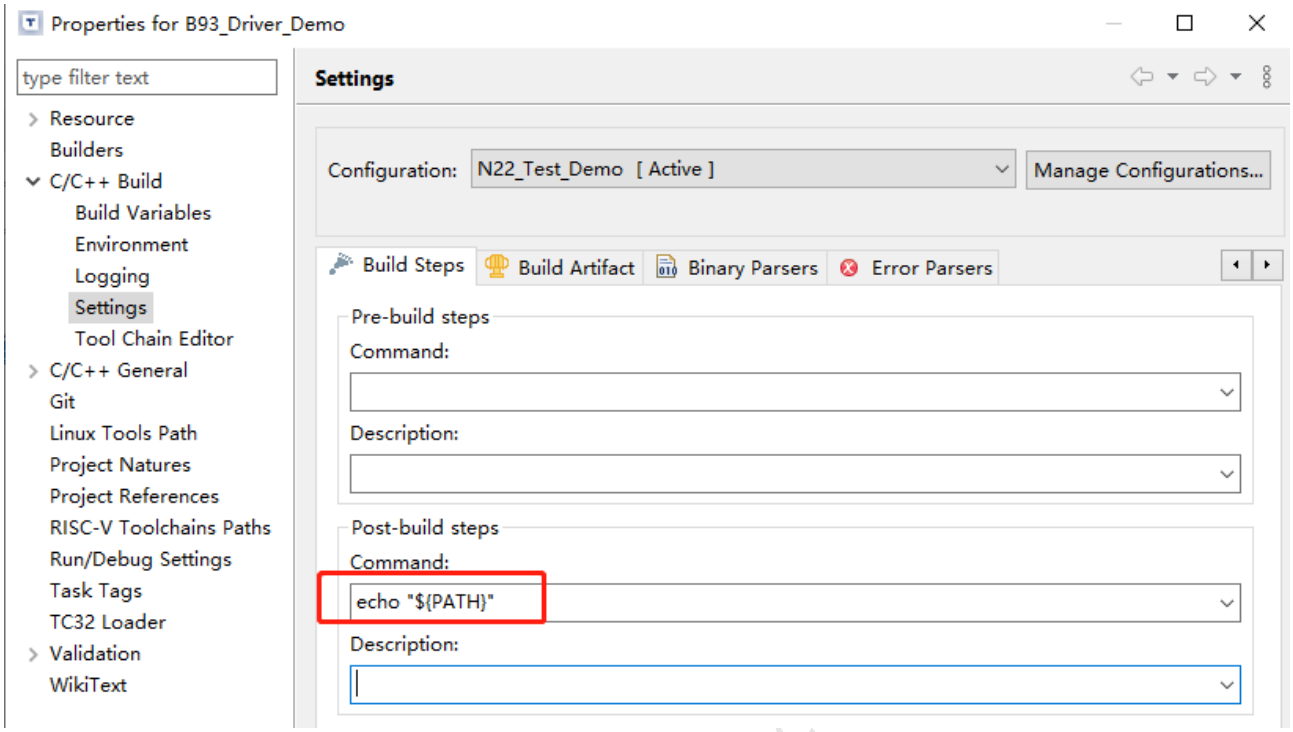

Figure 10.5: 在 Post-build 命令中设置 Make tool

然后构建项目，控制台会输出路径，其中第一个就是 make tool 路径。

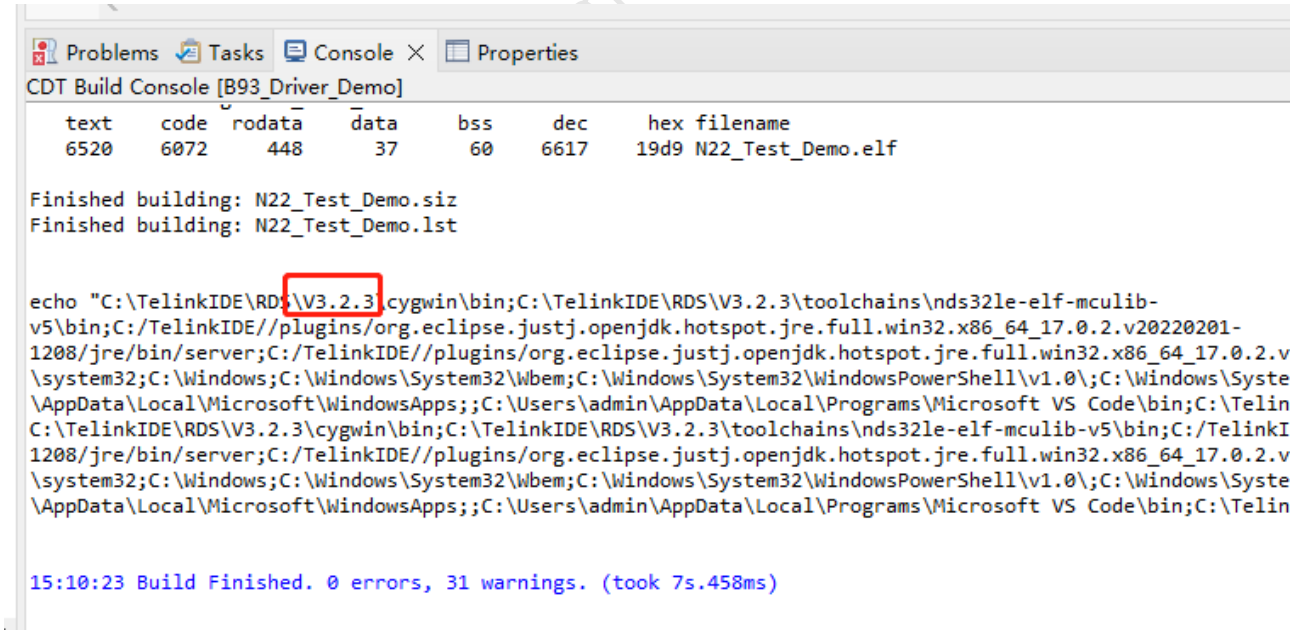

Figure 10.6: 验证 make tool 路径

在上面的示例结果中，`make tool` 路径为 `C:\TelinkIDE\RDS\V3.2.3\cygwin\bin` ，基础版本为 `V3.2.3` 。

10.6 打开 Telink TC32 console 时出现以下权限错误

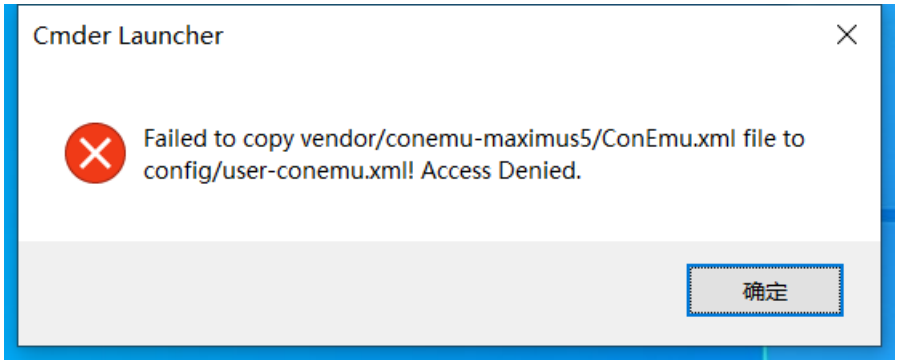

Figure 10.7: Permission problem

用户应以 `管理员` 的身份权限打开 `Telink TC32 console` 。

10.7 如何在 Windows 中开启大小写敏感

在 Windows 的文件系统中，默认情况下是不区分文件名的大小写的。例如，在你的工程中，有一个名为 App.h 的头文件和 app.h 的头文件，当你在源文件中包含 app.h 时，实际预处理时，gcc 很有可能会引用 App.h 文件。若你需要解决这个问题，可以按照下面的步骤开启 Windows 中的大小写敏感功能。

首先，需要打开 Windows 中的 WSL 支持，如下图：

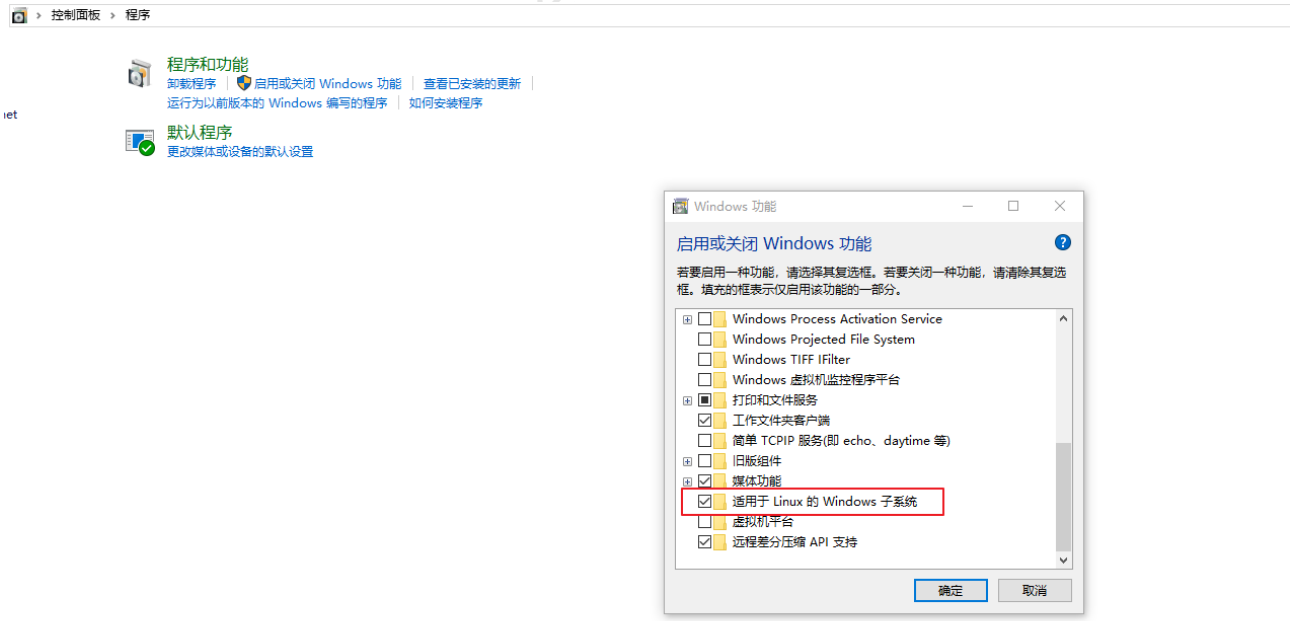

Figure 10.8: WSL 支持

然后，以管理员身份打开 Windows PowerShell, 跳转至你需要打开大小写敏感的文件夹，执行以下命令：

```
(Get-ChildItem -Recurse -Directory).FullName |
ForEach-Object {fsutil.exe file setCaseSensitiveInfo $_ enable}
```

11 已知问题

11.1 如果用户多次点击 `Open artifact path` 会导致 IoT Studio 退出

出现这种情况是因为 IoT Studio 无法找到 artifact 路径。用户应避免在短时间内多次点击工具栏上的 `Open artifact path` 图标。

11.2 在导入工程和改变工具链后，IoT Studio 会阻塞一段时间

这是一个正常的情况，因为 IoT Studio 需要花费一段时间来执行 `code indexing` 和扫描编译器文件（例如工具链系统头文件和目录）。

这个过程将持续几秒钟。用户应该在其他操作（例如：构建）之前等待这项工作完成。这是一个一次性的动作，发生在改变工具链设置或导入工程的时候。

11.3 生成的 artifact 大小为零

如果用户在工程刚刚导入，makefile 还没有完全生成的情况下就开始构建配置，就会出现这个问题。

当这种情况发生时，类似于以下情况的日志将显示在控制台的日志窗口。

```
riscv32-elf-gcc -mcmodel=medium -mcmodel=medium -O2 -fmessage-length=0 -ffunction-sections
-fdata-sections -flto  -g3 -T ../boot.link -nostartfiles -Xlinker --gc-sections
-L"E:\TelinkIDE\[Src]telink_eagle_ble_multi_connection_src\B91_ble_multi_conn_src\algorithm\lc3"
-Wl,-Map,"B91_demo.map" -g3 -mcpu=d25f -ffunction-sections -fdata-sections -fmessage-length=0
-fno-builtin -fomit-frame-pointer -fno-strict-aliasing -fshort-wchar -fuse-ld=bfd -fpack-struct
-O2 -o "B91_demo.elf"  ./vendor/common/hci_transport/hci_dfu.o
./vendor/common/hci_transport/hci_dfu_port.o ./vendor/common/hci_transport/hci_h5.o
./vendor/common/hci_transport/hci_slip.o ./vendor/common/hci_transport/hci_tr.o
./vendor/common/hci_transport/hci_tr_h4.o ./vendor/common/hci_transport/hci_tr_h5.o
./vendor/common/blt_common.o ./vendor/common/blt_fw_sign.o ./vendor/common/blt_led.o
./vendor/common/blt_soft_timer.o ./vendor/common/common_dbg.o ./vendor/common/custom_pair.o
./vendor/common/device_manage.o ./vendor/common/flash_fw_check.o ./vendor/common/simple_sdp.o
./common/tl_queue.o ./common/utility.o   -lm -ldsp
/cygdrive/d/TelinkIDE/RDS/V3.2.3/toolchains/nds32le-elf-mculib-v5f/bin/../lib/gcc/riscv32-elf/7.4.0/
../../../../riscv32-elf/bin/ld.bfd: warning: cannot find entry symbol _RESET_ENTRY; defaulting to
0000000020000000
Finished building target: B91_demo.elf

Invoking: GNU RISC-V Cross Create Flash Image
Invoking: GNU RISC-V Cross Create Listing
Invoking: GNU RISC-V Cross Print Size
riscv32-elf-objcopy -O binary "B91_demo.elf"  "B91_demo.bin"
riscv32-elf-size "B91_demo.elf"
riscv32-elf-objdump --source --all-headers --demangle --line-numbers --wide "B91_demo.elf" >
"B91_demo.lst"
   text    code  rodata    data     bss     dec     hex filename
      0       0       0       0       0       0       0 B91_demo.elf
```

Figure 11.1: Artifact 大小为零

想要解决这个问题，用户应该等待几秒钟，然后重新构建配置。

为了避免此类问题，用户应该等待 makefiles 的生成和索引完成。可以在 IoT Studio 的右下方查看具体完成进度。

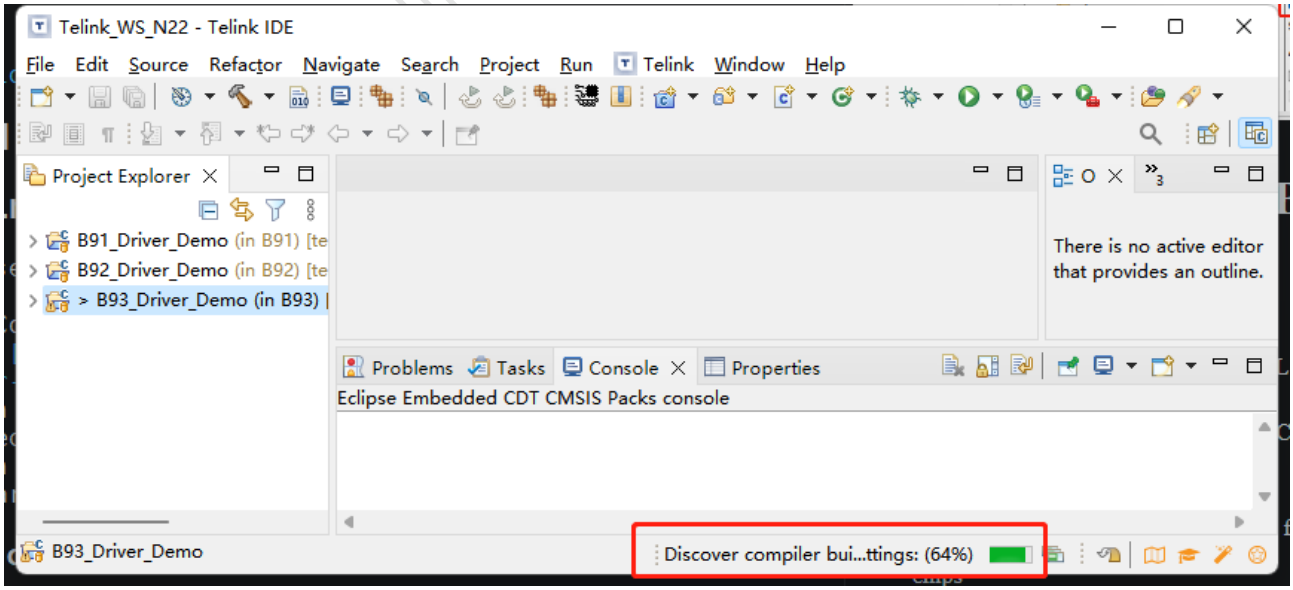

Figure 11.2: 查看进度

11.4 点击 Telink 工具栏的工具无效

当点击 Telink 工具栏中的工具之前，先双击选中一个工程，即可避免此问题

12 IoT Studio 设置

12.1 隐藏打印边距

默认显示打印边距，在代码编辑器中显示为一条垂直线。

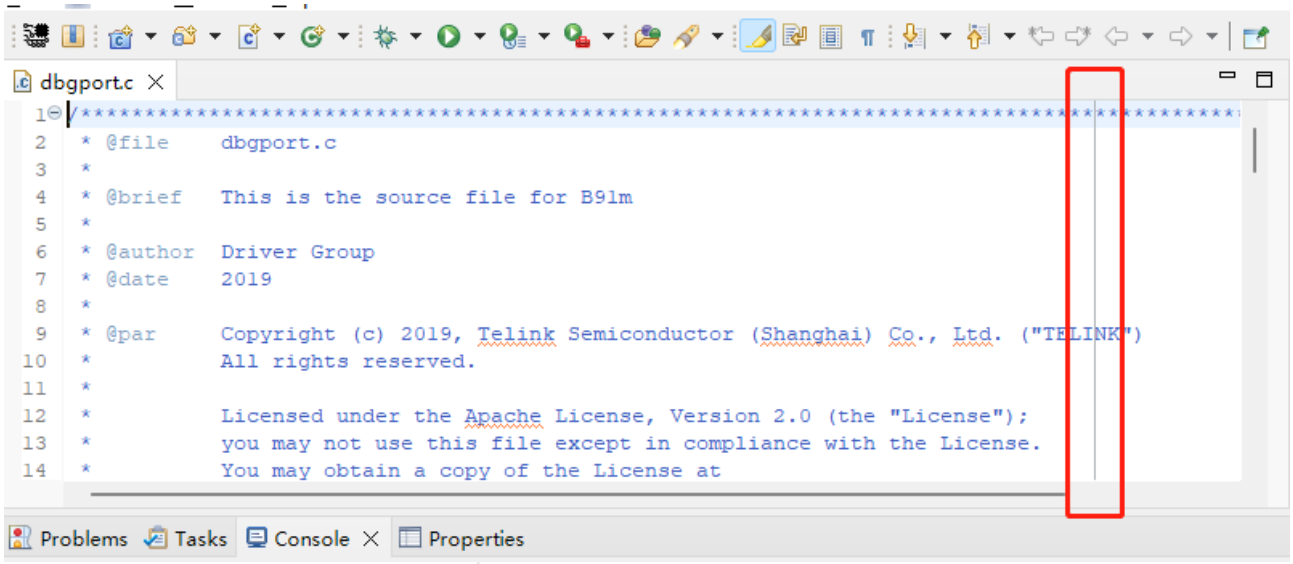

Figure 12.1: 编辑器中的打印边距

如果想要隐藏它，你可以在首选项的 `Text Editors` 页面上取消勾选 `Show print margin` 选项。

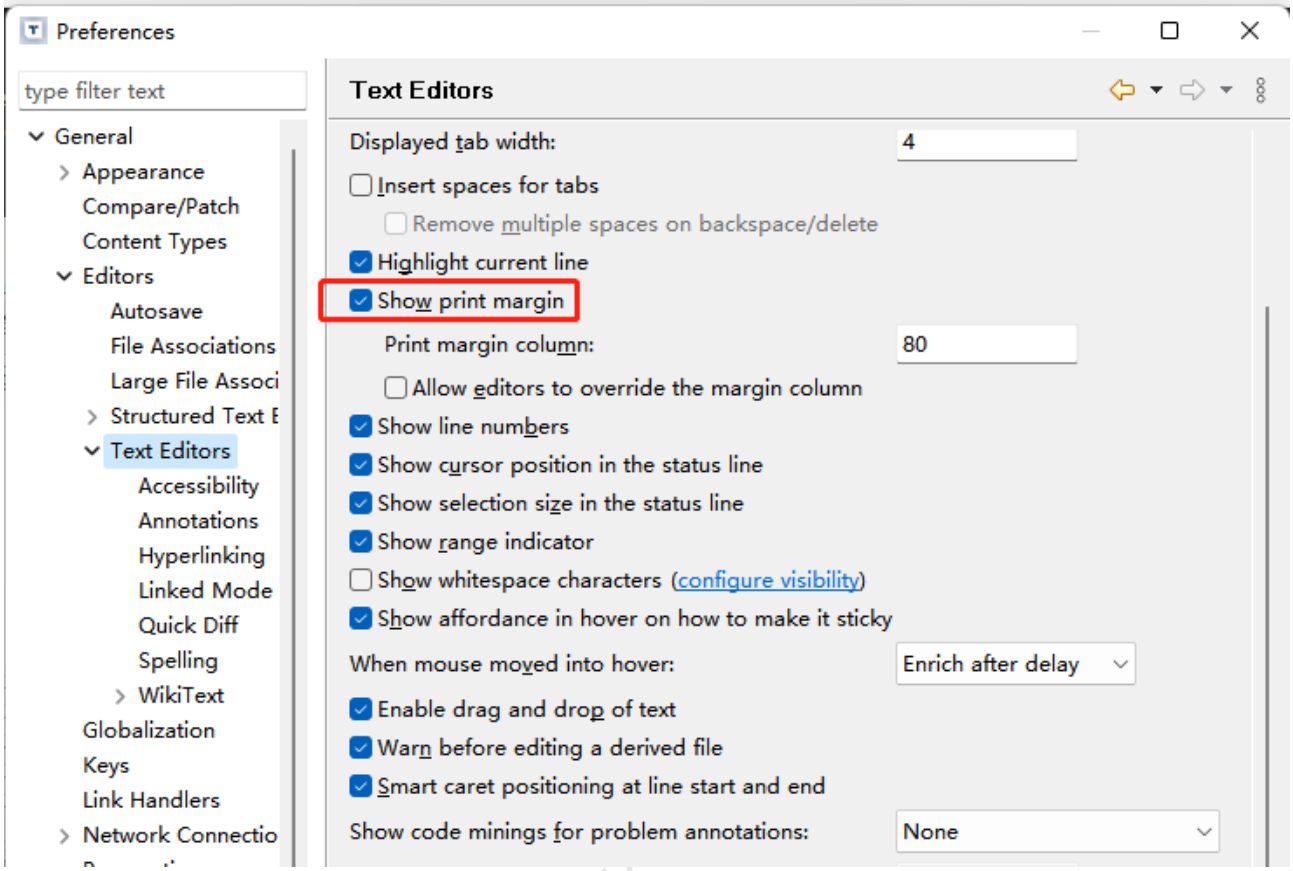

Figure 12.2: 配置打印边距

13 IoT Studio 调试工具（针对 TLSR9）

注意：这个 JTAG 调试工具仅仅针对 TLSR9 系列芯片使用

13.1 构建可调试的程序

为了使用调试功能，构建可执行程序时，要加上 -g 选项，并且删除编译或者链接时的影响调试的优化选项，如 -O2.

这里要注意一点，选择优化级别时，要选择 -Og，不要选择 -O0，因为在测试中发现，烧写过某些使用了 -O0 选项编译出的程序后，想要再次利用 Jtag 擦除烧写时，会发生下面的错误：

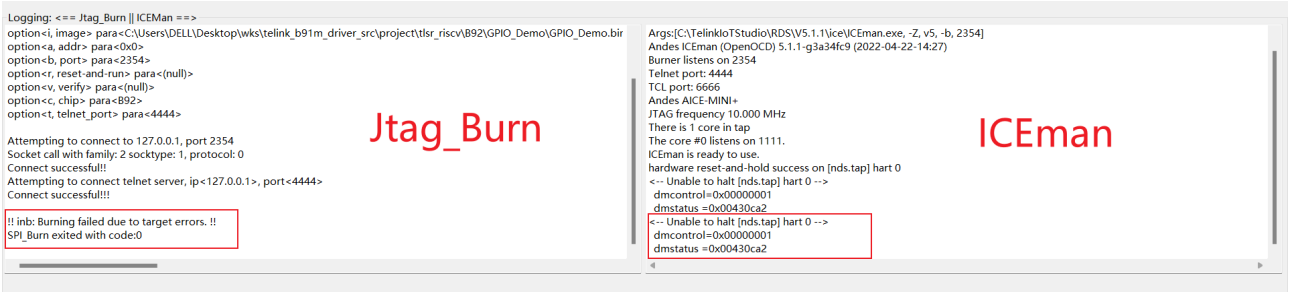

Figure 13.1: Jtag Burn err

要注意的一点是，在构建 V5.1.2 N22 的程序时，因为工具链的缘故，使用 GCC 10 版本的 N22 工具链时，要删掉 -mabi=ilp32f 选项 (同时注意检查 assembler 和 linker 的参数):

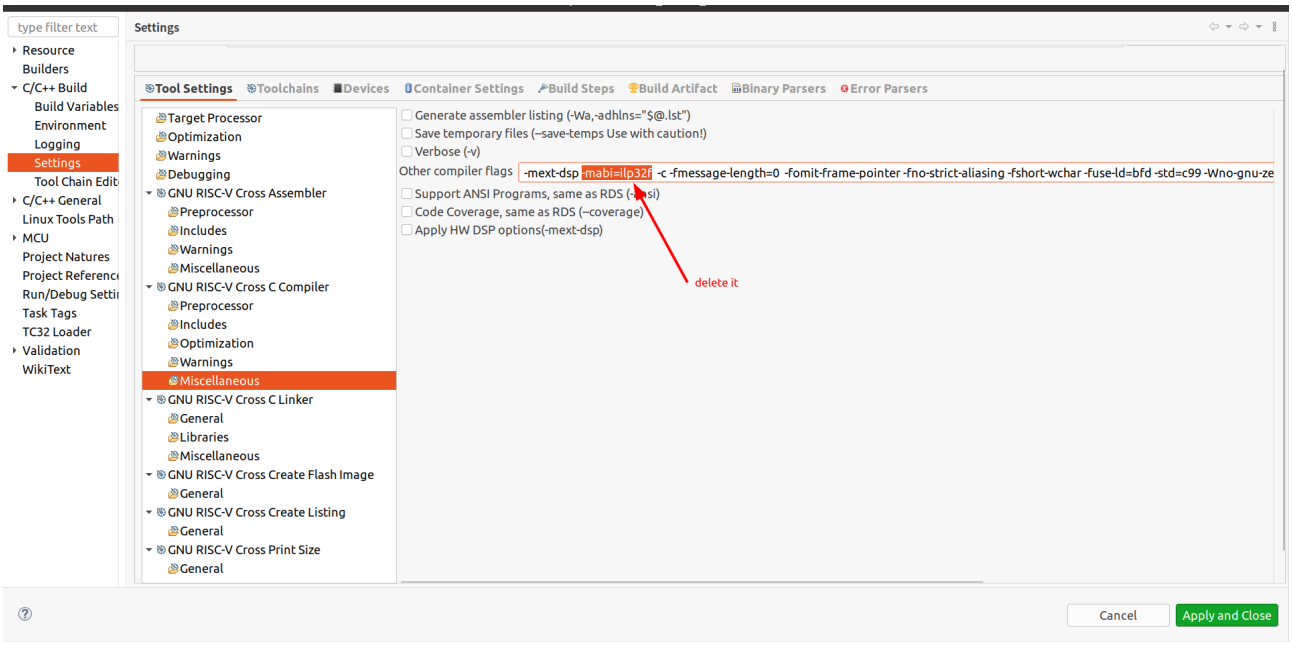

Figure 13.2: 删掉 mabi 选项

13.2 Jtag_Burn 烧录程序

烧录程序时，要注意配置 Jtag_Burn 的路径和芯片类型，然后点击 Start ICEman，ICEman 准备好后，确保 telnet port 和 burner port 与 ICEman 保持一致，再点击烧录，如下图:

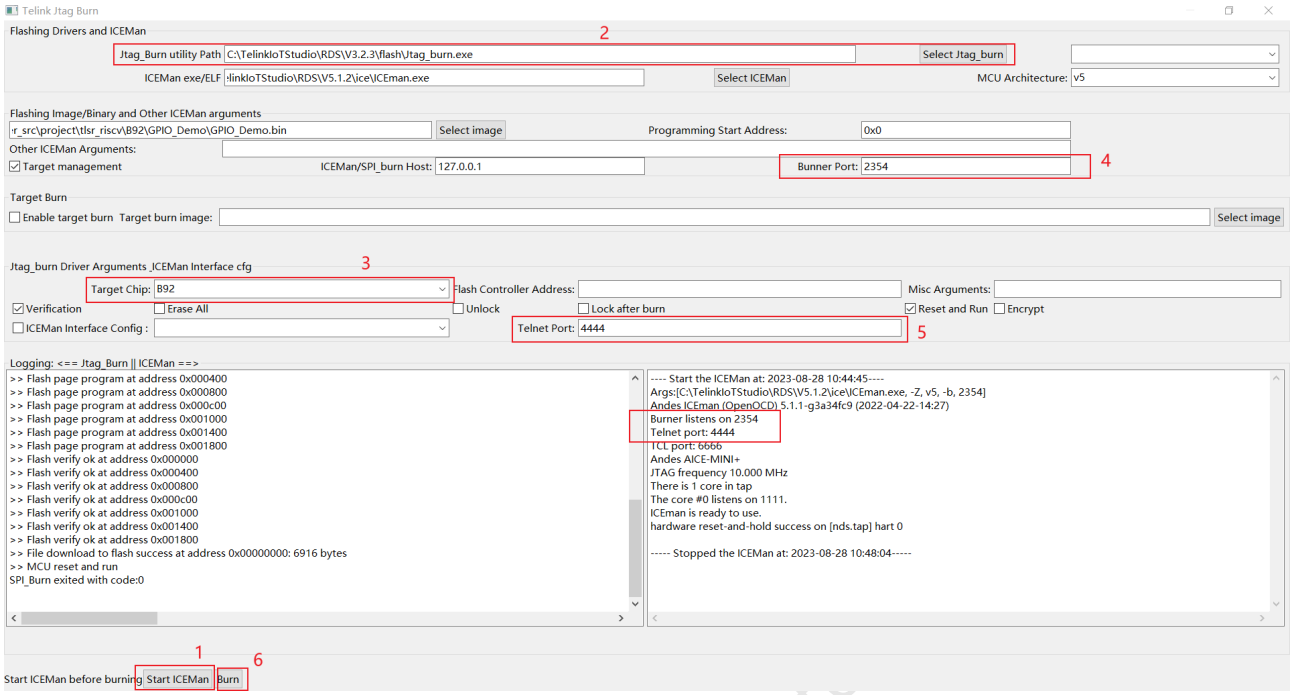

Figure 13.3: Jtag burn

其中，Jtag_Burn 位于 $IoTStudio/RDS/V3.2.3/flash/ 目录下，其帮助文档也在同一目录

13.3 Telink ICEMan GDB Debugging

开始调试之前，必须先双击选中你要调试的 elf 文件，如图中所示

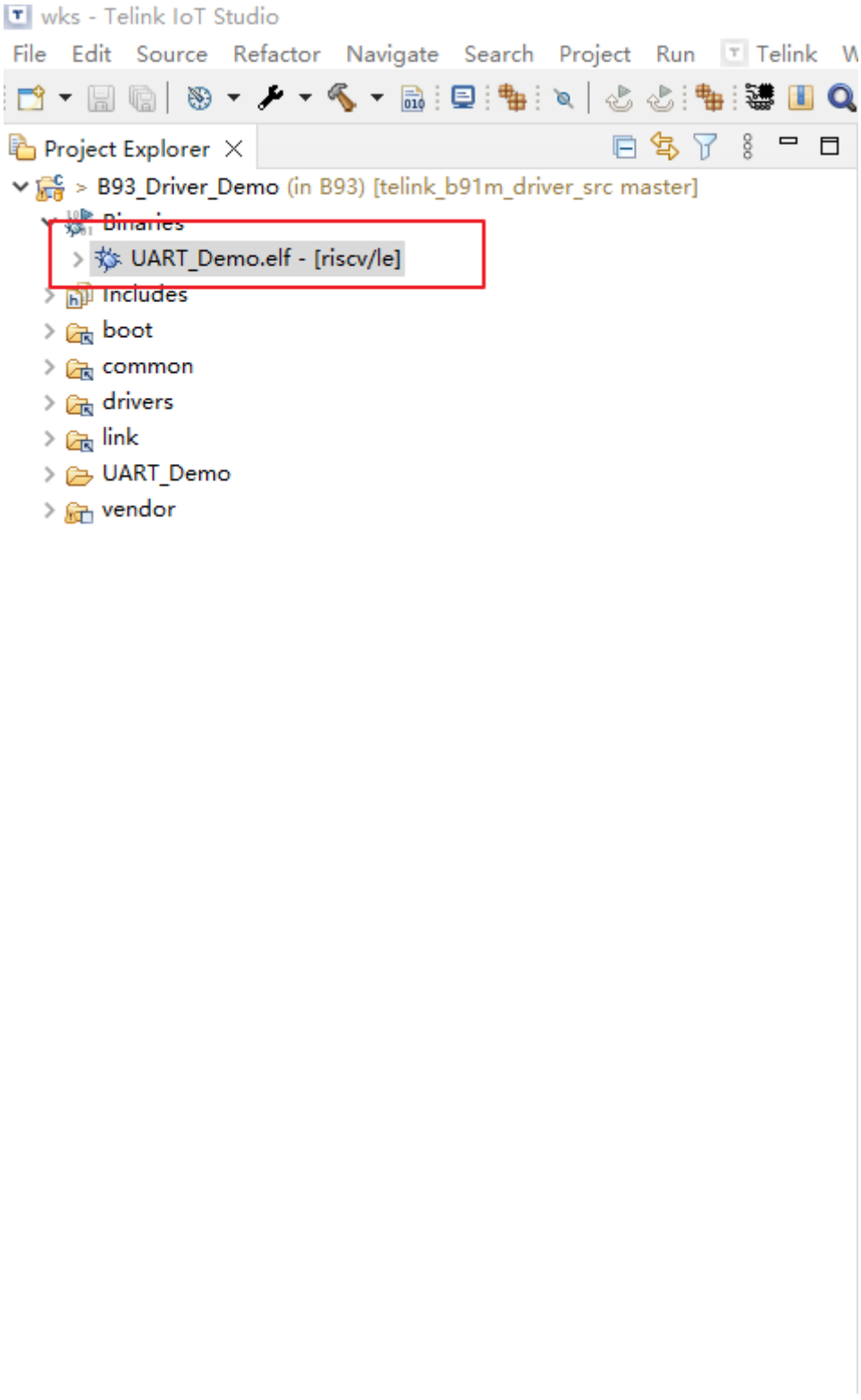

Figure 13.4: 选中 ELF

然后点击 debug 图标的下拉箭头，选择 `Debug Configurations...`

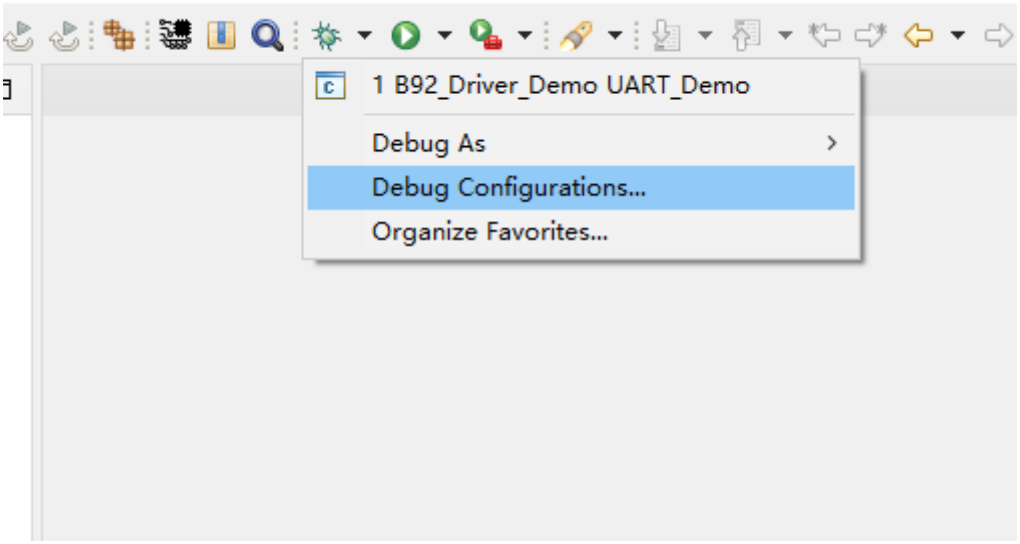

Figure 13.5: debug

双击 Telink ICEMan GDB Debugging，就会生成一个默认的配置，用户可以如图中所示选择是否需要 ICEman Interface 配置，默认情况下，用户不需要做其他更改就可以直接调试。

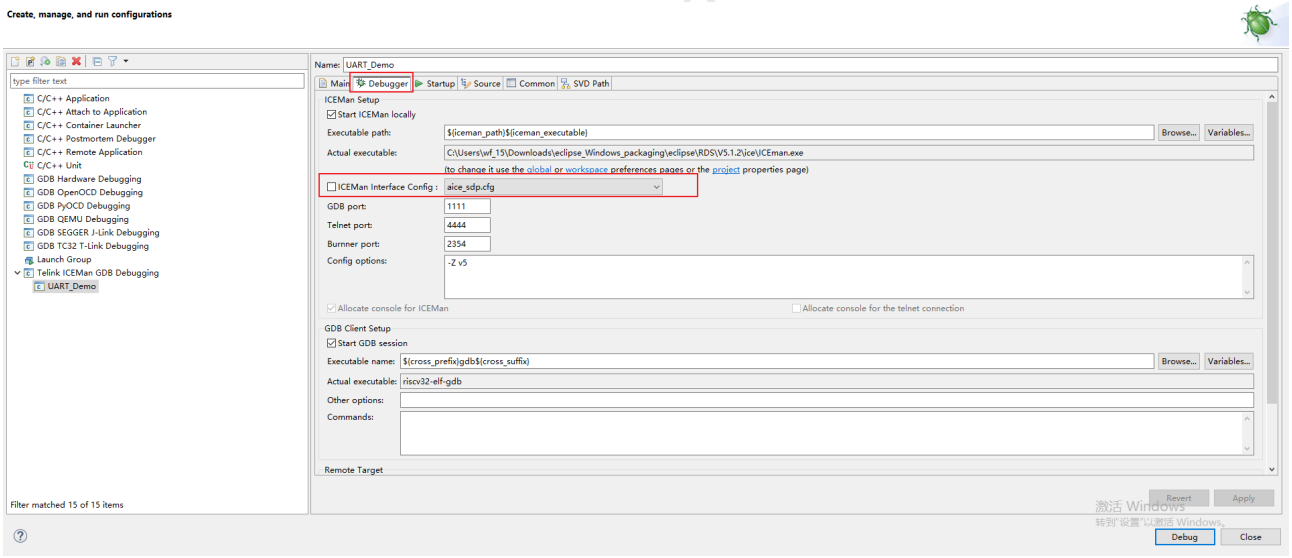

Figure 13.6: ICEman interface

13.4 通过 SVD 在调试中查看外设寄存器

TelinkIoTStudio 支持通过配置 SVD 文件查看目标设备的外设寄存器，用户需要选择所需的芯片类型来自动配置 SVD 文件，若有特别需求，也可以自定路 SVD 文件的路径，但是不推荐这种方式。

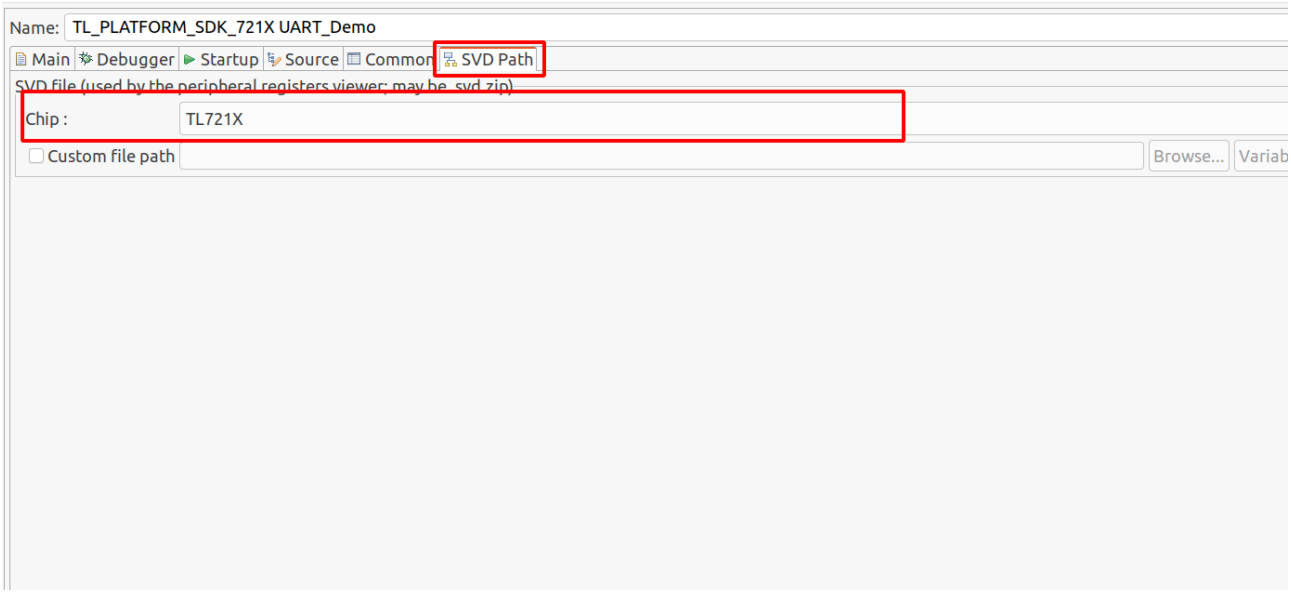

Figure 13.7: SVD Tab

然后在调试中，用户可以 Peripherals Tab 中来选择要查看的外设，然后在 Memory Tab 中查看

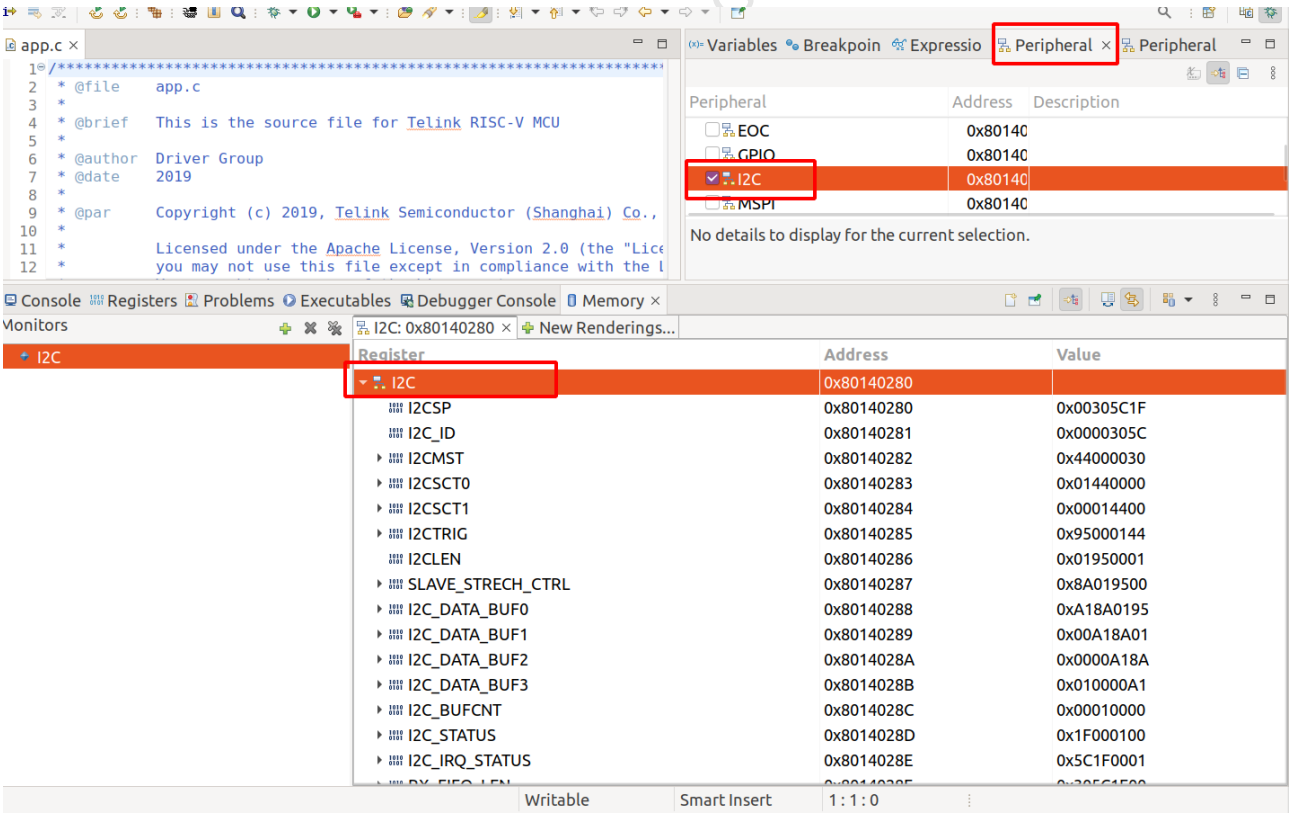

Figure 13.8: View Peripherals

13.5 断点

目前 TLSR9 系列 Soc 支持最多两个硬件断点，当你发现你的程序运行的起始地址是 0x20000000 时，说明它运行在 flash 里面，需要使用硬件断点，而 `step in` 或 `step over` 等命令本身就会用到一个断点，所以用户调试时只能自定义一个断点，否则就会发生异常。这也是为什么不建议勾选 Stop on startup at，因为这个选项实际上的工作是设置了一个断点

当你调试时发生了 cannot access memory at address xx 异常时，可以使用 `info br` 命令检查一下断点数量

上述配置完成后，即可正常调试

14 IoT Studio 中一些工具的单独使用方法

14.1 Converter

该工具的作用是将原 Telink RDS IDE 的工程文件格式转换为现在的 Telink IoT Studio 的工程格式。

其命令行的使用方法是，切换至需要转换的 `.cproject` 工程文件的目录，在此目录下执行 $(converter_path)/TelinkRDS2IDE.

converter 的所在目录是 $IoTStudio/tools/Converter/.

例如：

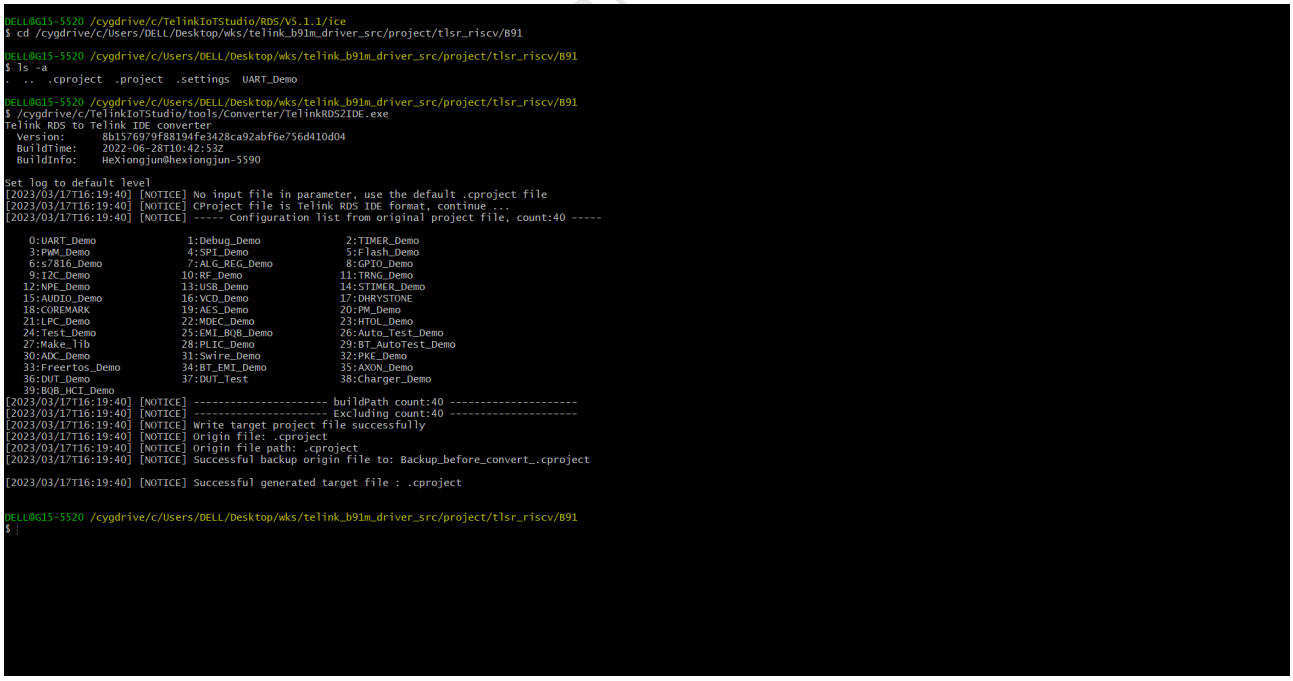

Figure 14.1: converter cmd

14.2 Jtag Burn

该工具的作用是将 binary 文件烧写至 TLSR9 芯片中，使用之前需要先使用 ICEman. Jtag Burn 工具位于 $IoTStudio/RDS/V3.2.3/flash/ 中，用户手册也在该目录下，下面是使用示例：

```
./Jtag_Burn --chip B92 --reset-and-run --verify --addr 0 --image /home/wang/UART_Demo.bin --port 2354 --telnet_port 4444 --unlock
```

其中，--port 和 --telnet_port 选项的参数需要和 ICEman 的 Burner_port 和 telnet_port 参数保持一致，若你看到 ICEman 的运行结果表明它的 burner_port 和 telnet_port 分别是 2354 和 4444（如下图）：

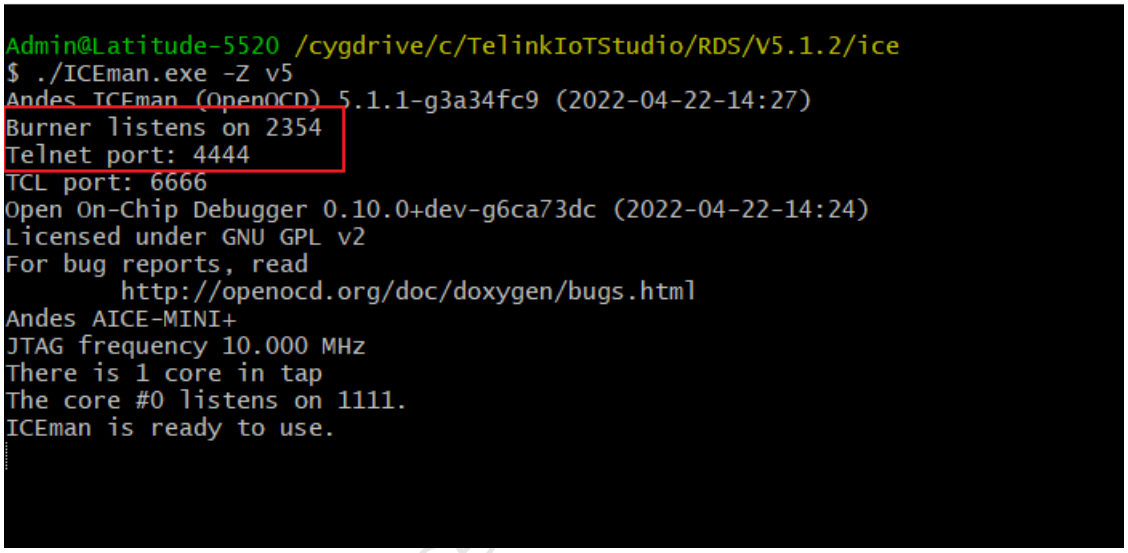

Figure 14.2: ICEman 默认端口

这种情况下，运行 Jtag_Burn 时，可以省略 --port 和 --telnet_port 这两个选项

15 Linux 中的安装和卸载

15.1 安装

在 linux 中，可以通过运行安装程序“ `Telink_IoT_Studio_xxxx_Installer.run` ”来安装 IoTStudio。程序执行中, 用户需要指定安装路径。需要确保输入的是绝对路径，如图中所示。

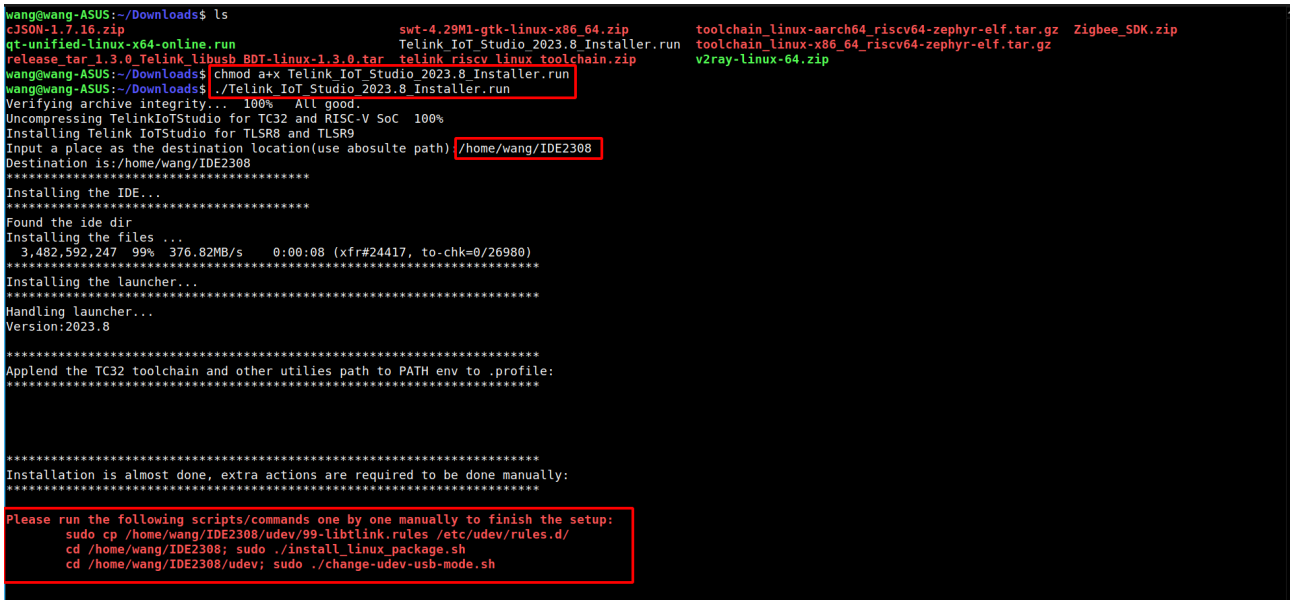

Figure 15.1: Installation

程序结束后，会在终端中列出一些命令，用户需要再手动执行这些命令，就可以完成安装。

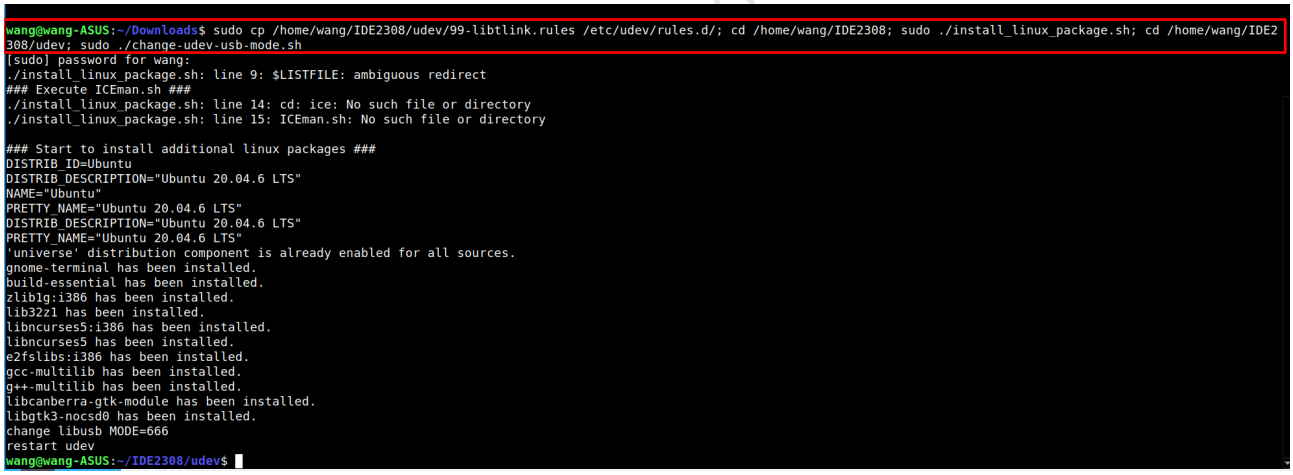

Figure 15.2: Post of installation

15.2 卸载

在 IoTStudio 的安装目录中，有一个 `uninstall.sh` 文件可以用来卸载 IoTStudio. 用户可以给该文件赋予可执行权限然后运行它. 用户需要将 IoTStudio 的.Desktop 文件作为参数传递给 `uninstall.sh` . 需要注意的是，安装目录下可能存在一个 TelinkIoTStudio.desktop 文件，注意不要将这个文件作为参数传递，需要使用带有版本号的.Desktop 文件，可以参考下面的示例。

```
./uninstall.sh /home/wang/IDE2308/TelinkIoTStudio_2023.8.desktop
```

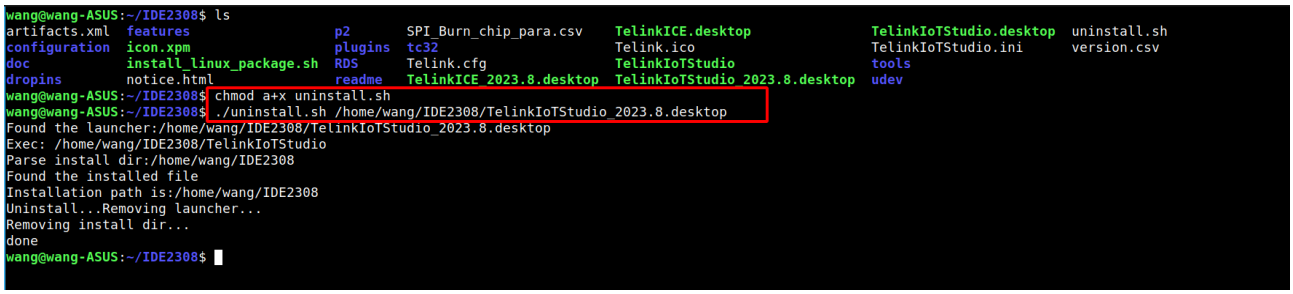

Figure 15.3: Uninstall

16 AndeSight 相关文档

我们提供了 AndeSight 的相关文档，包括 Andes Riscv 工具链以及其他工具的使用文档，用户可以根据自己的需求在下面的路径中找到。

- `${IoTStudio_Path}/doc/Andes_V323_doc/` : Docs for AndeSight V3.2.3
- `${IoTStudio_Path}/doc/Andes_V512_doc/` : Docs for AndeSight V5.1.2
- `${IoTStudio_Path}/doc/Andes_V532_doc/` : Docs for AndeSight V5.3.2

17 如何在命令行中编译 IoTStudio 工程

首先，需要在 IoTStudio 中先构建此项目，然后项目目录中会生成一个包含 makefile 和编译结果的文件夹，该目录通常与.cproject 位于同一目录中。

然后，在 Telink–> Toolchain Shell or ICEman Shell 中打开与项目匹配的 toolchain shell，图中的示例工具链是 `V5.1.2 GCC10.3 V5F toolchain shell`

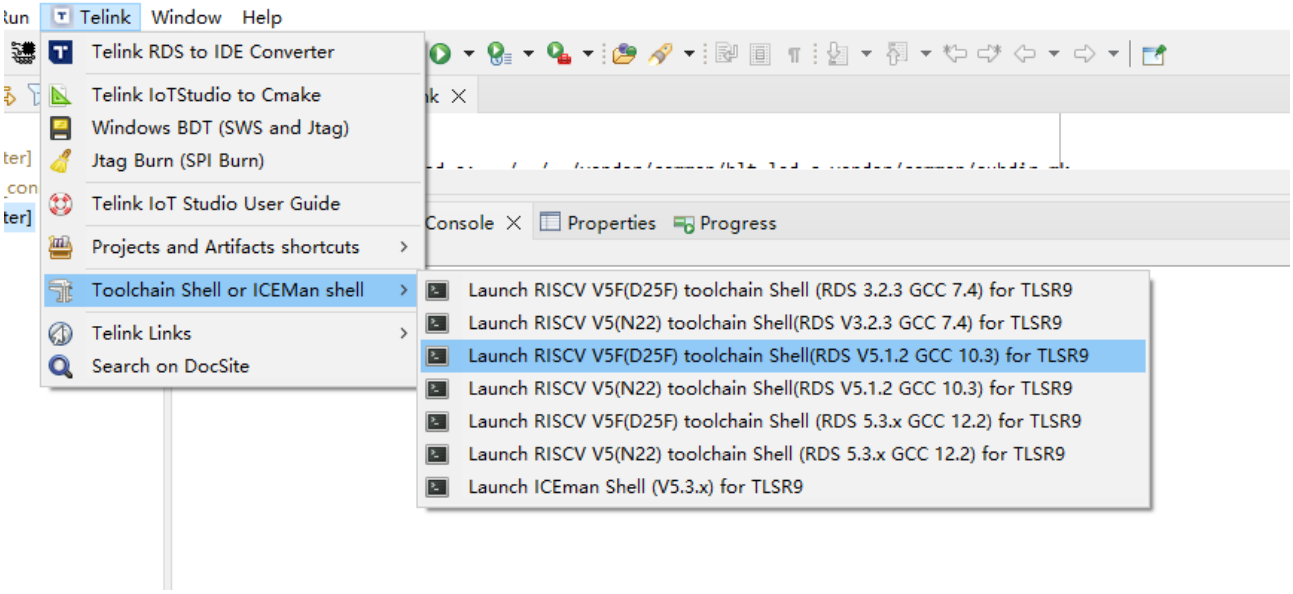

Figure 17.1: select toolchain shell

然后跳转到 makefile 所在的文件夹中，执行 `make clean` `make all`

```
telink@DESKTOP-GOULGO6 ~
$ cd /cygdrive/c/Users/wf_15/Desktop/wks/telink_b91m_driver_src/project/tlsr_riscv/B92/UART_Demo/

telink@DESKTOP-GOULGO6 /cygdrive/c/Users/wf_15/Desktop/wks/telink_b91m_driver_src/project/tlsr_riscv/B92/UART_Demo
$ make clean
rm -rf ./vendor/common/calibration/calibration.o  ./vendor/common/auto_test/pc_interface.o  ./vendor/common/common.o ./vendor/common/excep
_DEMO/main.o  ./drivers/lib/src/pke/eccp_curve.o ./drivers/lib/src/pke/ecdh.o ./drivers/lib/src/pke/ecdsa.o ./drivers/lib/src/pke/pke.o ./
vers/lib/src/aoa.o ./drivers/lib/src/flash_base.o ./drivers/lib/src/hadm_drv.o ./drivers/lib/src/plic.o ./drivers/lib/src/pm.o ./drivers/l
h/flash_mid146085.o ./drivers/flash/flash_mid156085.o ./drivers/flash/flash_mid1560c8.o ./drivers/flash/flash_mid166085.o ./drivers/flash/
ivers/charger.o ./drivers/charger_bin.o ./drivers/clock.o ./drivers/core.o ./drivers/ctb.o ./drivers/emi.o ./drivers/flash.o ./drivers/gpi
16.o ./drivers/spi.o ./drivers/stimer.o ./drivers/timer.o ./drivers/uart.o ./drivers/usbhw.o ./drivers/watchdog.o  ./common/usb_dbg/myudb_
ram.o UART_Demo.bin UART_Demo.lst UART_Demo.siz ./boot/cstartup_flash.d ./boot/cstartup_ram.d ./vendor/common/calibration/calibration.d  .
_DEMO/app.d ./vendor/UART_DEMO/app_dma.d ./vendor/UART_DEMO/app_dma_llp.d ./vendor/UART_DEMO/main.d  ./drivers/lib/src/pke/eccp_curve.d ./
src/pke/pke_utility.d ./drivers/lib/src/pke/rsa.d ./drivers/lib/src/pke/x25519.d  ./drivers/lib/src/aoa.d ./drivers/lib/src/flash_base.d .
s/lib/src/sys.d ./drivers/lib/src/trng.d  ./drivers/flash/flash_common.d ./drivers/flash/flash_mid146085.d ./drivers/flash/flash_mid156085
or_handler.d  ./drivers/adc.d ./drivers/aes.d ./drivers/analog.d ./drivers/audio.d ./drivers/charger.d ./drivers/charger_bin.d ./drivers/c
mdec.d ./drivers/mspi.d ./drivers/plmt.d ./drivers/pwm.d ./drivers/qdec.d ./drivers/s7816.d ./drivers/spi.d ./drivers/stimer.d ./drivers/t
on/bt_debug/dbgport.d  ./common/sdk_version.d  UART_Demo.elf


telink@DESKTOP-GOULGO6 /cygdrive/c/Users/wf_15/Desktop/wks/telink_b91m_driver_src/project/tlsr_riscv/B92/UART_Demo
$ make all
Building file: ../../../../demo/vendor/common/B92/calibration/calibration.c
Invoking: GNU RISC-V Cross C Compiler
riscv32-elf-gcc -mcmodel=medium -O2 -fmessage-length=0 -ffunction-sections -fdata-sections -flto -Werror -Wall -Wextra -Wshadow -Wimplicit
rict-prototypes -Wmissing-field-initializers -Wpointer-arith -Wdeprecated-declarations -Wredundant-decls -Werror=cpp -Wenum-conversion -Wp
DMCU_CORE_B93O=0 -DMCU_CORE_B95=0 -DMCU_CORE_B96=0 -I"C:\Users\wf_15\Desktop\wks\telink_b91m_driver_src\chip\B92\drivers" -I"C:\Users\wf_1
 -I"C:\Users\wf_15\Desktop\wks\telink_b91m_driver_src\demo\vendor\common\common" -Wall -Wextra -Wshadow -Wimplicit-fallthrough -Wpointer-a
g-field-initializers -Wpointer-arith -Wdeprecated-declarations -Wredundant-decls -Werror=cpp -Wenum-conversion -Wpacked-not-aligned -Waddr
td=c99 -fpack-struct -fshort-enums -fno-jump-tables -fno-fat-lto-objects -mcmodel=medium -Wno-nonnull-compare -MMD -MP -MF"vendor/common/c
emo/vendor/common/B92/calibration/calibration.c"
Finished building: ../../../../demo/vendor/common/B92/calibration/calibration.c
```

Figure 17.2: make in shell