

Nuclei Studio Supply

Documents

Table of contents

Home

● Nuclei Studio Supply Documents	14
● Documents	14

因内存不足，导致在Nuclei Studio中启动qemu失败

● 因内存不足，导致在Nuclei Studio中启动qemu失败	16
● 问题说明	16
● 解决方案	16

windows 11下使用Nuclei Studio进行qemu调试程序时报错

● windows 11下使用Nuclei Studio进行qemu调试程序时报错	17
● 问题说明	17
● 解决方案	17

How to print memory usage in Nuclei Studio

● How to print memory usage in Nuclei Studio	18
● 问题说明	18
● 解决方案	19

在编译工程时，使用了Pre-build Command/Post-build Command时报错

● 在编译工程时，使用了Pre-build Command/Post-build Command时报错	20
● 问题说明	20
● 解决方案	22

升级npk.yml以支持Nuclei Studio 2023.10

● 升级npk.yml以支持Nuclei Studio 2023.10	23
● npk.yml中的工具链升级	23
● 除标准的IMAFDC之外的扩展(ARCHEXT)的升级	24
● libncrt的升级	27
● Link Warning的消除	30
● 关于Nuclei SDK 0.5.0 npk.yml 详细变更	30

GCC13 auto generated RVV instructions when RVV enabled

● GCC13 auto generated RVV instructions when RVV enabled	32
● 问题说明	32
● 解决方案	32

更新 Nuclei Studio 2023.10 到最新修正版本

● 更新 Nuclei Studio 2023.10 到最新修正版本	34
● 问题描述	34
● 升级Nuclei Studio 2023.10 到最新版本的方法	34
● 对2023年11月18日之前下载了Nuclei Studio 2023.10进行升级	34
● 从官网下载最新的版本	37
● 参考资料	38

OpenOCD在操作容量大于16M-Byte的nor-flash时的问题

● OpenOCD在操作容量大于16M-Byte的nor-flash时的问题	39
● 问题说明	39
● 解决方案	39

通过修改.cproject文件，升级工程工具链到GCC 13

● 通过修改.cproject文件，升级工程工具链到GCC 13	40
● 问题描述	40
● 修改toolchain相关配置	40
● 修改RISC-V扩展相关配置	43
● 修改libncrt C库相关配置	45
● 增加link warning消除的配置	47

在Nuclei Studio下用命令行编译工程

● 在Nuclei Studio下用命令行编译工程	49
● 问题说明	49
● 解决方案	49

OpenOCD烧写程序时报错Error:Device ID 8xle2g8a6d is not known as FESPI capable

● OpenOCD烧写程序时报错Error:Device ID 8xle2g8a6d is not known as FESPI capable	72
● 问题说明	72
● 解决方案	72

关于dhystone在IDE上跑分和NSDK 0.5.0命令行跑分不一致的问题

● 关于dhystone在IDE上跑分和NSDK 0.5.0命令行跑分不一致的问题	74
● 问题说明	74
● 解决方案	74

Error: Couldn't find an available hardware trigger / Error: can't add breakpoint: resource not available

● Error: Couldn't find an available hardware trigger / Error: can't add breakpoint: resource not available	78
● 问题说明	78

cannot find -lncrt_balanced: No such file or directory

● cannot find -lncrt_balanced: No such file or directory	80
● 问题说明	80
● 解决方案	82

UnsatisfiedLinkError of swt-win32-4965r8.dll on Windows 7

● UnsatisfiedLinkError of swt-win32-4965r8.dll on Windows 7	83
● 问题说明	83
● 解决方案	85

使用 Profiling 功能时可能遇到的一些问题

● 使用 Profiling 功能时可能遇到的一些问题	87
● 问题1：日志打印中报片上内存不足，没有充足内存来存放 gprof/gcov 数据	87
● 解决方案	87
● 问题2：Console 或 Terminal 收集的数据不全导致数据解析时失败	88
● 解决方案	90
● 问题3：删掉 gmon.out 文件，再次解析，弹出 No files have been generated 错误弹框	92
● 解决方案	93
● 问题4：Linux 环境中使用 Nuclei studio 导入 amrwb_profiling_demo.zip 编译报错	94

Nuclei Studio使用Profiling功能进行性能调优举例

● Nuclei Studio使用Profiling功能进行性能调优举例	96
● 问题说明	96
● 解决方案	96
● 1 环境准备	96
● 2 Profiling 功能	96
● 2 Call Graph 功能	108
● 3 Code coverage 功能	110
● 4 补充	113

通过Profiling展示Nuclei Model NICE/VNICE指令加速

● 通过Profiling展示Nuclei Model NICE/VNICE指令加速	114
● 背景描述	114
● Nuclei Model Profiling	114
● NICE/VNICE 自定义指令加速	114
● 解决方案	115
● 环境准备	115
● Model Profiling	115
● step1：新建 demo_vnice 工程	115
● step2：基于 demo_vnice 工程移植 aes_demo 裸机用例	116
● step3：model 仿真程序	117
● step4：解析 gprof 数据	120
● step5：NICE/VNICE 指令替换	120
● step6：在 Nuclei Model 中实现 NICE/VNICE 指令	125

Nuclei Model结合Nice Wizard快速验证NICE/VNICE指令加速

● Nuclei Model结合Nice Wizard快速验证NICE/VNICE指令加速	131
● 背景描述	131
● xlmodel_nice	131
● Nuclei NICE Wizard	131
● test code	131
● 解决方案	131
● 环境准备	131
● Nuclei Model运行原始程序	132
● NICE指令替换	134
● VNICE指令替换	140
● 总结	144

Flash Programming使用案例

● Flash Programming使用案例	145
● 解决方案	145
● 环境准备	145
● Flash Programming 使用演示	145
● 总结	154

Live Watch 功能的使用

● Live Watch 功能的使用	155
● 背景描述	155
● 解决方案	155
● 环境准备	155
● Live Watch 使用演示	155
● 总结	162

在llvm中新增自定义汇编指令教程

● 在llvm中新增自定义汇编指令教程	163
● 自定义扩展名的识别	163
● 自定义汇编指令识别	163
● 使用说明	165
● 参考链接	165

如何使用芯来提供的DebugMap寄存器分析错误现场

● 如何使用芯来提供的DebugMap寄存器分析错误现场	166
● 首先需要确定硬件支持DebugMap功能	166
● 什么是DebugMap寄存器	166
● OpenOCD里DebugMap的输出信息	166
● 可能出现的错误现场	166
● 如何正确利用DebugMap分析错误现场	168

● 通过OpenOCD读取其他DebugMap寄存器	168
----------------------------	-----

在binutils中新增自定义汇编指令教程

● 在binutils中新增自定义汇编指令教程	169
● 自定义扩展名的识别	169
● 自定义汇编指令识别	170
● 使用说明	171
● 参考链接:	171

OpenOCD 中 Nuclei 交叉触发功能使用指南

● OpenOCD 中 Nuclei 交叉触发功能使用指南	173
● 功能概述	173
● 配置文件示例	173
● 1. 同步暂停组配置	173
● 2. 同步恢复组配置	174
● 命令行验证步骤	175
● 1. 同步暂停组验证	175
● 2. 同步恢复组验证	176
● IDE 验证步骤	177
● 1. 同步暂停组验证	177
● 2. 同步恢复组验证	178

OpenOCD对FreeRTOS的调试支持使用指南

● OpenOCD对FreeRTOS的调试支持使用指南	180
● 环境准备	180
● 使用步骤	180
● 使用说明	183

如何同时使用多个蜂鸟调试器进行调试

● 如何同时使用多个蜂鸟调试器进行调试	184
● 问题说明	184
● 解决方案	184
● 下载FT_PROG	184
● 查看串号	184
● 修改串号	185
● 更新OpenOCD配置	186
● Linux	187
● Windows	187
● 参考资料	187

Nuclei SDK基于evalsoc快速适配customsoc

● Nuclei SDK基于evalsoc快速适配customsoc	188
● 方案说明	188

● 解决方案	188
● 环境准备	188
● 适配修改	188
● 1 修改cpu特性描述宏文件	188
● 2 修改cpu特性isa配置	188
● 3 修改链接地址的memory map	189
● 4 修改openocd配置文件	189
● 5 修改Systimer频率	189
● 6 修改CPU主频	189
● 7 修改串口驱动	189
● 8 修改串口波特率	190
● 9 修改_premain_init	190
● 10 删除Nuclei内部使用的代码	190
● 11 检查外设地址	190
● 测试运行	191
● 调整名称	191
● 精简代码	191
● IAR工程	192
● IDE工程支持	192
● 参考资料	193

在链接脚本中使用OVERLAY命令

● 在链接脚本中使用OVERLAY命令	194
● 问题说明	194
● 解决方案	194
● 示例程序	194

● Overlay布局	195
● 编写链接脚本	195
● 测试结果	196
● 注意事项	198
● 参考资料	198

Nuclei Studio Supply Documents¶

 Deploy MkDocs passing

 pages-build-deployment passing

This repository is utilized for providing supply documents, user guides, wikis, and facilitating discussions related to Nuclei Studio.

Note

- The latest version of Nuclei Studio IDE is 2025.10, which can be found in <https://github.com/Nuclei-Software/nuclei-studio/releases/tag/2025.10>
- In Ubuntu 20.04, you must install libncursesw5 libtinfo5 libfdt1 libpixman-1-0 libpng16-16 libasound2 libglib2.0-0 to make riscv64-unknown-elf-gdb and qemu able to run.

PDF Version can be found here: https://doc.nucleisys.com/nuclei_studio_supply/pdf/nuclei_studio_supply.pdf

- Nuclei Studio IDE Documentation: https://doc.nucleisys.com/nuclei_tools/ide/index.html
- Nuclei Tools(Toolchain/OpenOCD/Qemu/Model) Documentation: https://doc.nucleisys.com/nuclei_tools/
- Nuclei Studio NPK Introduction:
- <https://github.com/Nuclei-Software/nuclei-sdk/wiki/Nuclei-Studio-NPK-Introduction>
- https://doc.nucleisys.com/nuclei_tools/ide/npkoverview.html

Please create new doc based on [Doc Template](#)

Click [this link](#) to see online version.

如果您在文档中发现任何拼写错误或不完善之处，我们欢迎您提交Pull Request或Issue，以协助我们进行改进！

If you come across any spelling errors or areas that need improvement in the document, feel free to submit a Pull Request or Issue to help us enhance it!

Documents¶

Generated by python3 update.py @ 2026-02-13 02:15:40

- 1. 因内存不足，导致在Nuclei Studio中启动qemu失败
- 2. windows 11下使用Nuclei Studio进行qemu调试程序时报错
- 3. How to print memory usage in Nuclei Studio
- 4. 在编译工程时，使用了Pre-build Command/Post-build Command时报错

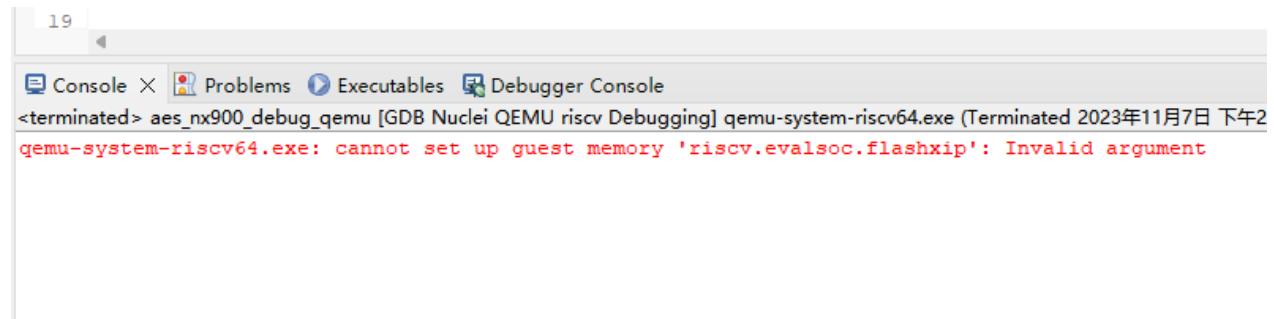
- 5. 升级npk.yml以支持Nuclei Studio 2023.10
- 6. GCC13 auto generated RVV instructions when RVV enabled
- 7. 更新 Nuclei Studio 2023.10 到最新修正版本
- 8. OpenOCD在操作容量大于16M-Byte的nor-flash时的问题
- 9. 通过修改.cproject文件，升级工程工具链到GCC 13
- 10. 在Nuclei Studio下用命令行编译工程
- 11. OpenOCD烧写程序时报错Error:Device ID 8xle2g8a6d is not known as FESPI capable
- 12. 关于dhrystone在IDE上跑分和NSDK 0.5.0命令行跑分不一致的问题
- 13. Error: Couldn't find an available hardware trigger / Error: can't add breakpoint:
resource not available
- 14. cannot find -lncrt_balanced: No such file or directory
- 15. UnsatisfiedLinkError of swt-win32-4965r8.dll on Windows 7
- 16. 使用 Profiling 功能时可能遇到的一些问题
- 17. Nuclei Studio使用Profiling功能进行性能调优举例
- 18. 通过Profiling展示Nuclei Model NICE/VNICE指令加速
- 19. Nuclei Model结合Nice Wizard快速验证NICE/VNICE指令加速
- 20. Flash Programming使用案例
- 21. Live Watch 功能的使用
- 22. 在llvm中新增自定义汇编指令教程
- 23. 如何使用芯来提供的DebugMap寄存器分析错误现场
- 24. 在binutils中新增自定义汇编指令教程
- 25. OpenOCD 中 Nuclei 交叉触发功能使用指南
- 26. OpenOCD对FreeRTOS的调试支持使用指南
- 27. 如何同时使用多个蜂鸟调试器进行调试
- 28. Nuclei SDK基于evalsoc快速适配customsoc
- 29. 在链接脚本中使用OVERLAY命令

因内存不足，导致在Nuclei Studio中启动qemu失败¶

问题说明¶

在实际开发中发现，因电脑同时运行了很多的进程或者电脑本身的系统内存不足，致使在Nuclei Studio中，使用qemu进行程序调试时，可能出现如下报错：

```
qemu-system-riscv64.exe: cannot set up guest memory  
'riscv.evalsoc.flashxip' Invalid argument
```



解决方案¶

一般可以通过关闭某些应用，释放一部分内存以供qemu使用，即可解决些问题。

windows 11下使用Nuclei Studio进行qemu调试程序时报错¶

问题说明¶

windows 11下使用Nuclei Studio开发时，当使用qemu调试程序时，会有报错如下，是因为在 windows 11下缺少相关依赖，但一般不影响qemu的正确使用，可以忽略此错误。

```
qemu-system-riscv32.exe: warning: GLib-GIO: Unexpectedly, UWP app
`Microsoft.ScreenSketch_11.2309.16.0_x64_8wekyb3d8bbwe' (AUMId
`Microsoft.ScreenSketch_8wekyb3d8bbwe!App') supports 29 extensions
but has no verbs
qemu-system-riscv32.exe: warning: GLib-GIO: Unexpectedly, UWP app
`Clipchamp.Clipchamp_2.8.1.0_neutral_yxz26nhyzhsrt' (AUMId
`Clipchamp.Clipchamp_yxz26nhyzhsrt!App') supports 41 extensions but
has no verbs
```

```
<terminated> 050hello_debug_qemu [GDB Nuclei QEMU riscv Debugging] qemu-system-riscv32.exe (Terminated 2023年11月7日 下午2:52:59)
GDB Server listening on: 'tcp:1234'...
Nuclei SDK Build Time: Nov 7 2023, 14:46:26
Download Mode: ILM
CPU Frequency 2295653007 Hz
CPU HartID: 0
qemu-system-riscv32.exe: warning: GLib-GIO: Unexpectedly, UWP app `Microsoft.ScreenSketch_11.2309.16.0_x64_8wekyb3d8bbwe' (AUMId `Microsoft.ScreenSketch_8wekyb3d8bbwe!App') supports 29 extensions
qemu-system-riscv32.exe: warning: GLib-GIO: Unexpectedly, UWP app `Clipchamp.Clipchamp_2.8.1.0_neutral_yxz26nhyzhsrt' (AUMId `Clipchamp.Clipchamp_yxz26nhyzhsrt!App') supports 41 extensions
qemu-system-riscv32.exe: QEMU: Terminated via GDBstub
```

解决方案¶

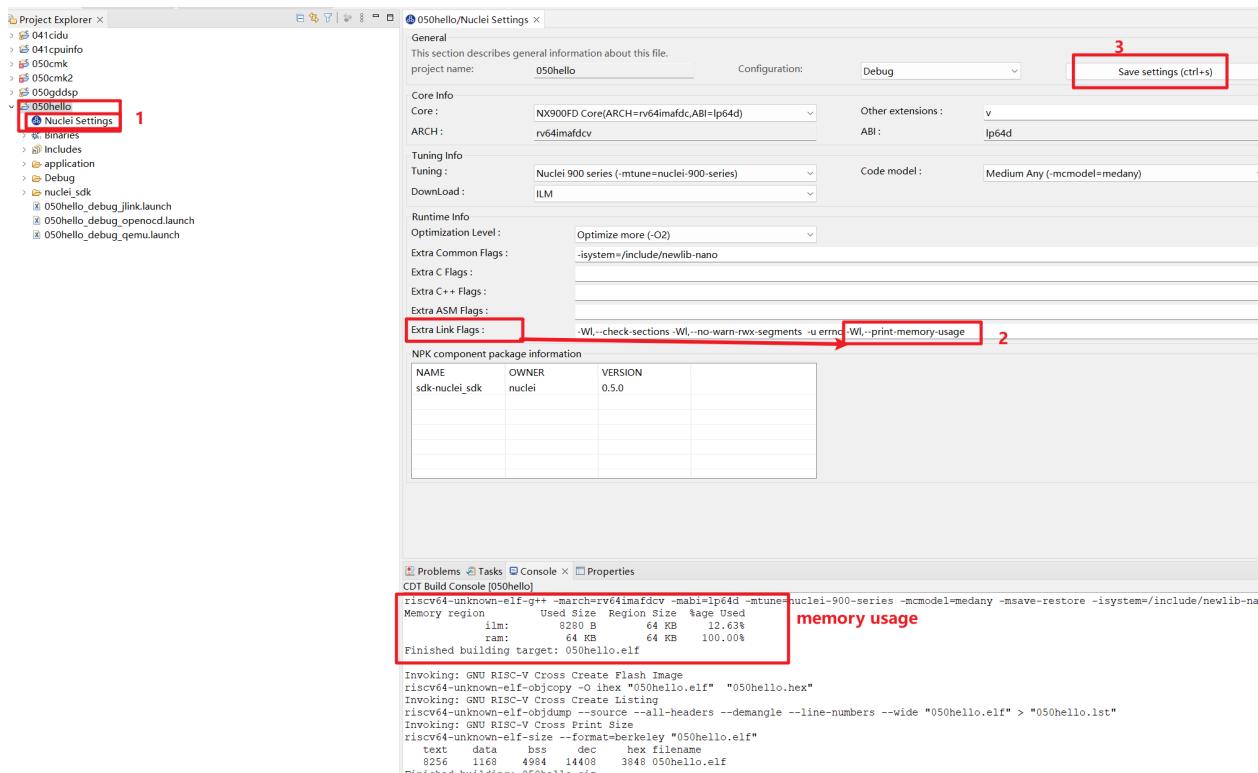
这个是Windows 10/11上存在的系统库匹配问题，不影响Qemu正常使用，可以忽略。

How to print memory usage in Nuclei Studio¶

问题说明¶

In order to print memory usage when compile an application, you can do it like this:

Click Nuclei Settings in selected project, and pass extra `-Wl,--print-memory-usage` in Extra Link Flags, and save settings, and then build this project, you will be able to see memory usage.



```
Building target: 050hello.elf
Invoking: GNU RISC-V Cross C++ Linker
.....
Memory region      Used Size  Region Size %age Used
    ilm:          8280 B       64 KB   12.63%
    ram:           64 KB       64 KB  100.00%
Finished building target: 050hello.elf
```

Why the ram usage here is 100% used?

For Nuclei SDK or NMSIS template linker script, the stack is placed at the bottom of ram memory, so the ram usage is 100%.

解决方案

Add extra link option -Wl, --print-memory-usage will solve this issue.

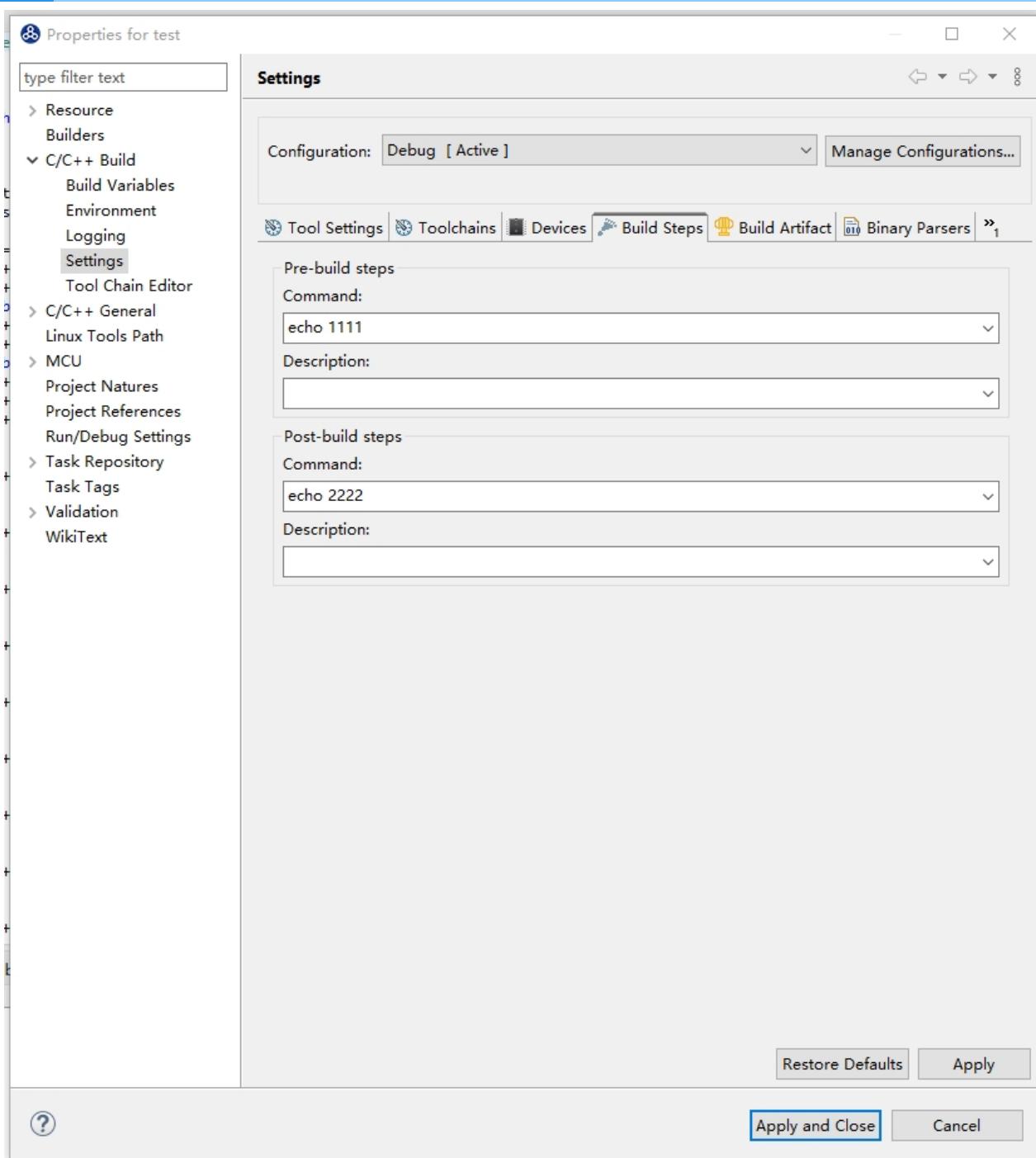
在编译工程时，使用了Pre-build Command/Post-build Command时报错¶

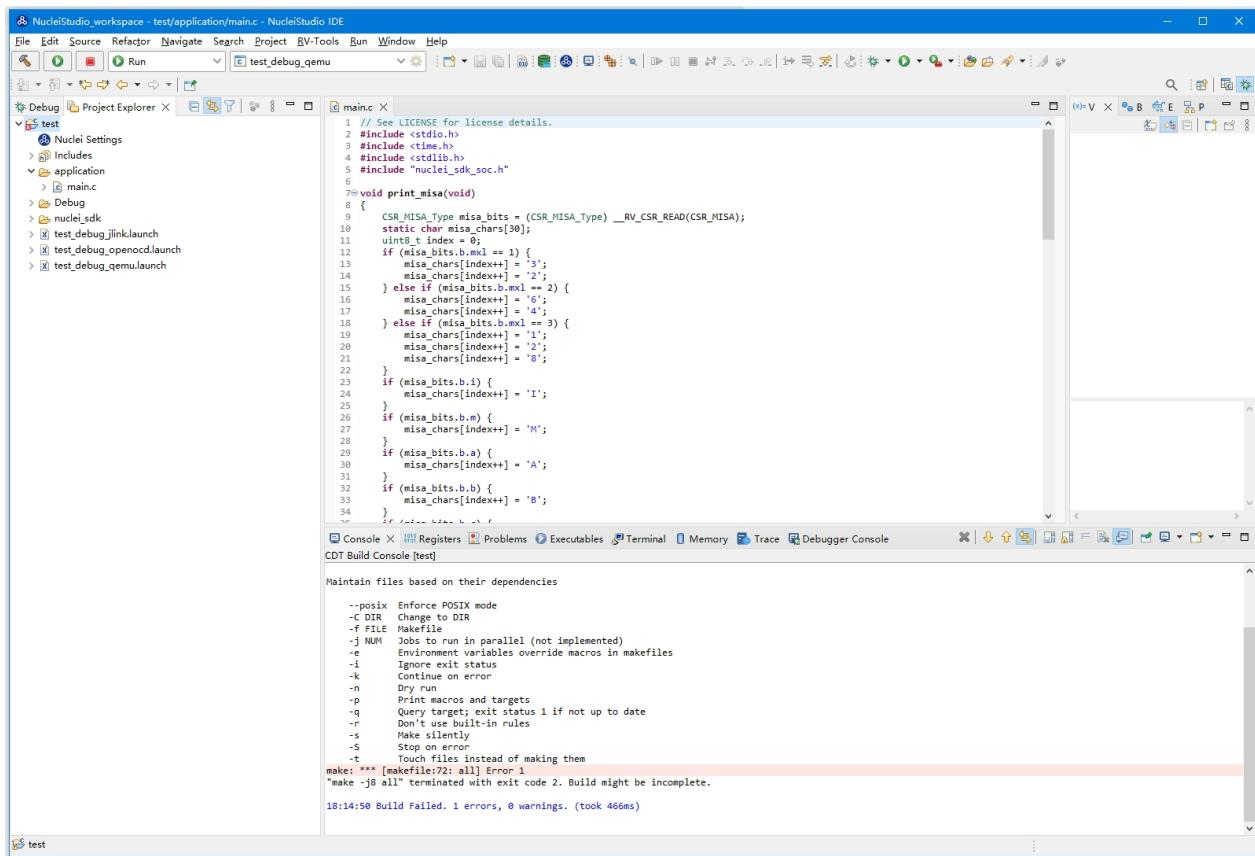
问题说明¶

在 Nuclei Studio 2023.10.17上传的2023.10 更正版本中修正，参见[本文](#)

参见 [eclipse-embed-cdt/eclipse-plugins#597](#)

在Nuclei Studio 2023.10版本中，如果在工程编译中需要使用到Pre-build Command/Post-build Command，因Nuclei Studio中集成的build-tools为v4.4.0版本，而上游CDT中在处理Pre-build Command/Post-build Command的方法，在build-tools v4.4.0无法正常使用，所以会出现报错问题。





解决方案¶

遇到这种情况时，可以下载 https://www.nucleisys.com/upload/files/toochain/build-tools/build-tools_202002.zip，并替换工具链中的build-tools，问题可以得到解决。

升级到最新的 2024.06 和 2025.02 就不存在这个问题。

NucleiStudio\toolchain\build-tools

升级npk.yml以支持Nuclei Studio 2023.10¶

在Nuclei Studio 2023.10中，一个重要变更，是支持GCC 13，所以之前发布的NPK Package也需要做对应的变更，以更好的适用于Nuclei Studio 2023.10，其中有以下几个变更点。

需要注意新版的npk.yml 不再支持以前 2022.12版本的IDE

npk.yml中的工具链升级¶

在npk中，我们定义了buildconfig来自定义工程build时的各种参数，Nuclei Studio通过type标识使用的是那一种toolchain，如gcc、clang等，通过 type->toolchain_name & cross_prefix 来标识使用的toolchain里面具体的那个发行版本。升级SDK以支持GCC 13，对比以下两个例子不难看出，只需要修改 toolchain_name: RISC-V GCC/Newlib 和 cross_prefix: riscv64-unknown-elf-，就可以使SDK支持在创建工程时，可以选择GCC 13工具链。

以下内容是支持gcc 10 的buildconfig配置（为了方便举例，隐藏了部分参数，具体参数根据实际情况定义）。

```
## Build Configuration
buildconfig:
  - type: gcc
    description: Nuclei GNU Toolchain
    cross_prefix: riscv-nuclei-elf- # optional
    common_flags: # flags need to be combined together across all
packages
    ldflags:
    cflags:
    asmflags:
    cxxflags:
    commonDefines:
    prebuild_steps: # could be override by app/bsp type
      command:
      description:
    postbuild_steps: # could be override by app/bsp type
      command:
      description:
```

下以内容，是支持GCC 13和Clang的buildconfig配置（为了方便举例，隐藏了部分参数，具体参数根据实际情况定义）。

```
## Build Configuration
buildconfig:
  - type: gcc
    description: Nuclei GNU Toolchain
    # 升级到GCC13时，这里进行如下两行的改变
    # 且针对所有npk.yml的文件只要包含buildconfig的都需要进行修改，不仅仅限于ssp/bsp类型，还包括bsp/app/mwp/osp/sdk类型
    toolchain_name: RISC-V GCC/Newlib
    cross_prefix: riscv64-unknown-elf- # optional
    common_flags: # flags need to be combined together across all packages
      ldflags:
      cflags:
      asmflags:
      cxxflags:
      commonDefines:
      prebuild_steps: # could be override by app/bsp type
        command:
        description:
      postbuild_steps: # could be override by app/bsp type
        command:
        description:
  - type: clang
    description: Nuclei LLVM Toolchain
    toolchain_name: RISC-V Clang/Newlib
    cross_prefix: riscv64-unknown-elf- # optional
    common_flags: # flags need to be combined together across all packages
      ldflags:
      cflags:
      asmflags:
      cxxflags:
      commonDefines:
      prebuild_steps: # could be override by app/bsp type
        command:
        description:
      postbuild_steps: # could be override by app/bsp type
        command:
        description:
```

除标准的IMAFDC之外的扩展(ARCHEXT)的升级¶

以下示例以Nuclei SDK 0.5.0的evalsoc的npk.yml升级举例，仅考虑GCC的支持，如果需要考虑CLANG的支持，请参见SDK中evalsoc的npk.yml的详细变更

在GCC 13中，对RISC-V 指令扩展使用有了很大的变更，具体内容可以查看Nuclei Studio用户手册2.1.4章内容和Nuclei SDK中ARCH_EXT说明。

- [Nuclei Studio用户手册](#)

- [ARCH_EXT说明](#)

升级npk.yml时，如果SDK中使用到了RISC-V 除了标准的IMAFDC之外指令扩展，例如B/P/K/V，也需要升级对应的配置。

在NPK中，RISC-V 指令扩展以是-march=xxx的方式传递给Nuclei Studio，Nuclei Studio接收到相关配置，就会存储并应用到编译的过程中。以Nuclei SDK中的npk.yml为例，通过下面这段配置我们就可以得到-march=的值，不难看出与RISC-V指令扩展相关的是NPK中的变量nuclei_archext。

```
## (为了方便举例，隐藏了部分参数，具体参数根据实际情况定义)
## Build Configuration
buildconfig:
  - type: gcc
    description: Nuclei RISC-V GNU Toolchain #must
    cross_prefix: riscv-nuclei-elf- # optional
    common_flags: # flags need to be combined together across all
      packages
      # 这里 -march 传递的值就是 nuclei_core.arch 和 nuclei_archext 两
      个变量拼接而来
      # 例如 nuclei_core.arch设置为rv32imafdc, nuclei_archext设置为
      _zba_zbb_zbc_zbs_xxldspn1x,
      # 那么传递的就是 -march=rv32imafdc_zba_zbb_zbc_zbs_xxldspn1x
      # 如果你的 march是已知和确定的，这里直接就可以给定 -march/-mabi的选
      项，无需通过 configuration字段来进行传递
      - flags: -march=${nuclei_core.arch}$(join($
        {nuclei_archext},'')) -mabi=${nuclei_core.abi}
      ldflags:
      cflags:
      asmflags:
      cxxflags:
      commonDefines:
      prebuild_steps: # could be override by app/bsp type
        command:
        description:
      postbuild_steps: # could be override by app/bsp type
        command:
        description:
```

在旧版的SDK中，nuclei_archext定义的是一个multicheckbox，用户可以自己选择，而在新版的SDK中nuclei_archext定义的是一个text输入框，这样用户可以更灵活的使用RISC-V 指令扩展，如果在某些工程或场景下，想要预设一些RISC-V 指令扩展，建议给一个默认值就可以了，可以参考下代的示例代码。

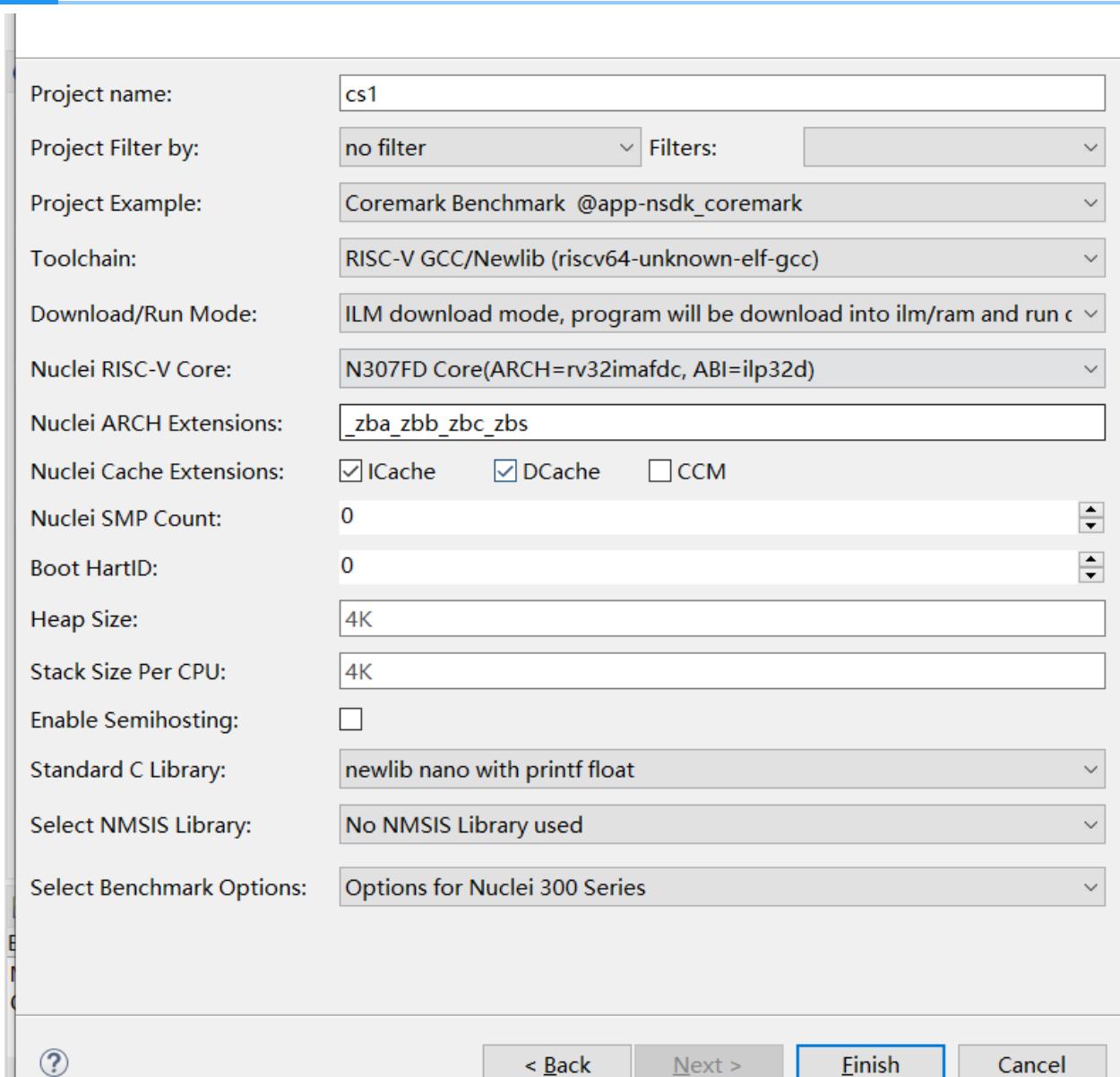
- 用于支持Nuclei RISC-V Toolchain 2022.12的写法

```
## 旧版的SDK中，nuclei_archext定义的是一个multicheckbox
## (为了方便举例，隐藏了部分参数，具体参数根据实际情况定义)
nuclei_archext:
    default_value: []
    type: multicheckbox
    global: true
    description: Nuclei ARCH Extensions
    choices:
        - name: b
          description: Bitmanip Extension
        - name: p
          description: Packed SIMD Extension
        - name: v
          description: Vector Extension
```

- 用于支持Nuclei RISC-V Toolchain 2023.10的写法

```
## 新版的SDK中nuclei_archext定义的是一个text输入框
## Package Configurations
configuration:
nuclei_archext:
    default_value: "_zba_zbb_zbc_zbs"
    type: text
    global: true
    # hints and tips are introduced in Nuclei Studio 2023.10
    # used to show tool tips and input hints
    tips: "Possible other ISA extensions, seperated by
underscores, like '_zba_zbb_zbc_zbs_xxldspn1x'"
    hints: "_zba_zbb_zbc_zbs_xxldspn1x"
    description: Nuclei ARCH Extensions
```

最终显示创建项目的时候显示效果如下



libncrt的升级

libncrt较之前也有了些许变化，在NPK中使用libncrt之前，新旧版SDK中都是一样的在configuration中定义了一个变量stdcplib，它的值是一个下拉框，可以选择不同的值。不同点是在得到stdcplib后，在common_flags或者其它地方使用stdcplib时略有不同。

关于stdcplib的一些说明，可以参见[这里](#)

```
## 定义stdcplib变量
## (为了方便举例，隐藏了部分参数，具体参数根据实际情况定义)
## Package Configurations
configuration:
  stdcplib:
    default_value: newlib_nano
    type: choice
```

```

global: true
description: Standard C Library
choices:
  - name: newlib_full
    description: newlib with full feature
  - name: newlib_fast
    description: newlib nano with printf/scanf float
  - name: newlib_small
    description: newlib nano with printf float
  - name: newlib_nano
    description: newlib nano without printf/scanf float
  - name: libncrt_fast
    description: nuclei c runtime library, optimized for speed
  - name: libncrt_balanced
    description: nuclei c runtime library, balanced, full
feature
  - name: libncrt_small
    description: nuclei c runtime library, optimized for size,
full feature
  - name: libncrt.nano
    description: nuclei c runtime library, optimized for size,
no float support
  - name: libncrt_pico
    description: nuclei c runtime library, optimized for size,
no long/long long support
  - name: nostd
    description: no std c library will be used, and don't
search the standard system directories for header files
  - name: nospec
    description: no std c library will be used, not pass any --
specs options

```

在新版的SDK中，如果使用`--specs=libncrt_xxx.specs` 或者链接库里面包含`-lncrt_xxx`（表示采用libncrt c库），则需变更为`-lncrt_xxx -lfileops_uart -lheapops_basic`，这也是旧SDK变更为支持GCC 13的新SDK的原则。

下面配置为在旧版SDK中的npk变量`stdclib`,当变量`stdclib`以`libncrt`开头时，会直接定义一个`--specs=${stdclib}.specs`，按照上面我们说的原则，这里应该变成设置`-l$(subst(${stdclib},lib,)) -lfileops_uart -lheapops_basic`，所以在新版SDK中的写法就变成了下面的配置方式。

```

## 在旧版SDK中使用stdclib变量
## (为了方便举例，隐藏了部分参数，具体参数根据实际情况定义)
## Build Configuration
buildconfig:

```

```
- type: gcc
  description: Nuclei GNU Toolchain
  cross_prefix: riscv-nuclei-elf- # optional
  common_flags: # flags need to be combined together across all
    packages
    - flags: --specs=${stdcplib}.specs
      condition: $(startswith(${stdcplib}, "libncrt") )
  ldflags:
  cflags:
  asmflags:
  cxxflags:
  commonDefines:
  prebuild_steps: # could be override by app/bsp type
    command:
    description:
  postbuild_steps: # could be override by app/bsp type
    command:
    description:
```

转变为

```
## 在新版SDK中使用stdcplib变量
## (为了方便举例，隐藏了部分参数，具体参数根据实际情况定义)
## Build Configuration
buildconfig:
  - type: gcc
    description: Nuclei GNU Toolchain
    toolchain_name: RISC-V GCC/Newlib
    cross_prefix: riscv64-unknown-elf- # optional
    common_flags: # flags need to be combined together across all
      packages
      - flags: --specs=${stdcplib}.specs
        condition: $(startswith(${stdcplib}, "libncrt") )
    ldflags:
      - flags: -l$(subst(${stdcplib},lib,)) -lheapops_basic -
    lfileops_uart
      condition: $(startswith(${stdcplib}, "libncrt") )
    cflags:
    asmflags:
    cxxflags:
    commonDefines:
    prebuild_steps: # could be override by app/bsp type
      command:
      description:
    postbuild_steps: # could be override by app/bsp type
```

```
command:  
description:
```

Link Warning的消除¶

在Nuclei Studio 2023.10中集成的GCC 13,在使用过程中会有warning, 链接选项增加一个-Wl, --no-warn-rwx-segments可以隐藏warning。

具体可以参考以下配置 (为了方便举例, 隐藏了部分参数, 具体参数根据实际情况定义)

```
## Build Configuration  
buildconfig:  
  - type: gcc  
    description: Nuclei GNU Toolchain  
    toolchain_name: RISC-V GCC/Newlib  
    cross_prefix: riscv64-unknown-elf- # optional  
    common_flags: # flags need to be combined together across all  
    packages  
    ldflags:  
      # 用于消除gcc13链接阶段的warning  
      - flags: -Wl,--no-warn-rwx-segments  
    cflags:  
    asmflags:  
    cxxflags:  
    commonDefines:  
    prebuild_steps: # could be override by app/bsp type  
      command:  
      description:  
    postbuild_steps: # could be override by app/bsp type  
      command:  
      description:
```

关于Nuclei SDK 0.5.0 npk.yml 详细变更¶

关于支持Nuclei Studio + Nuclei RISC-V Toolchain 2023.10的npk.yml变更, 可以参考nuclei-sdk 0.5.0的变更。

- gd32vf103的变化 git diff 0.4.1..0.5.0 SoC/gd32vf103/**/npk.yml
- evalsoc的变化: git diff 0.4.1..0.5.0 SoC/evalsoc/**/npk.yml
- NMSIS的变化: git diff 0.4.1..0.5.0 NMSIS/**/npk.yml

- application的变化: `git diff 0.4.1..0.5.0 application/**/npk.yml`
- RTOS的变化: `git diff 0.4.1..0.5.0 OS/**/npk.yml`

执行查看代码变更命令方法如下

```
git clone https://github.com/Nuclei-Software/nuclei-sdk/
cd nuclei-sdk
git fetch --all
git diff 0.4.1..0.5.0 SoC/gd32vf103/**/npk.yml
git diff 0.4.1..0.5.0 SoC/evalsoc/**/npk.yml
git diff 0.4.1..0.5.0 NMSIS/**/npk.yml
git diff 0.4.1..0.5.0 application/**/npk.yml
git diff 0.4.1..0.5.0 OS/**/npk.yml
```

GCC13 auto generated RVV instructions when RVV enabled¶

问题说明¶

If you are using Nuclei SDK 0.5.0 with Nuclei RISC-V Toolchain 2023.10, and when compile some examples with RVV enabled, it may generate rvv instructions which called auto-vectorization.

Take application/baremetal/benchmark/dhrystone for example:

```
cd application/baremetal/benchmark/dhrystone
# enable extra vector extension, which means the -march=rv64imafdcv
make CORE=nx900fd ARCH_EXT=v clean
make CORE=nx900fd ARCH_EXT=v dasm
```

Then if you check the dhrystone.dasm, you will be able to see rvv instructions:

解决方案¶

This auto generated instructions may affect your hardware performance, so if you want to disable it, you don't need to pass rvv extension when compile application.

```
$ cat dhrystone.dasm |grep vs
 800003e2: cc3ff057           vsetivli      zero,
31,e8,m8,ta,ma
 800003f8: 02038427           vse8.v v8,(t2)
 8000040c: 020b8027           vse8.v v0,(s7)
 800004a2: cc3ff057           vsetivli      zero,
31,e8,m8,ta,ma
 800004b2: 02098827           vse8.v v16,(s3)
 80000524: cc3ff057           vsetivli      zero,
31,e8,m8,ta,ma
 80000530: 02098c27           vse8.v v24,(s3)
 80000df2: cdb3f057           vsetivli      zero,
7,e64,m8,ta,ma
 80000dfa: 0204f427           vse64.v v8,(s1)
 80000e20: cdb3f057           vsetivli      zero,
7,e64,m8,ta,ma
 80000e28: 02047027           vse64.v v0,(s0)
```

You can check https://gcc.gnu.org/bugzilla/show_bug.cgi?id=112537 for more details.

In gcc 14.x, if you want to disable the RISC-V RVV automatic vectorization, you can use the options -fno-tree-loop-vectorize -fno-tree-slp-vectorize.

In gcc 13.x, you need to pass --param=riscv-autovec-preference=none

更新 Nuclei Studio 2023.10 到最新修正版本¶

2023.11.06上传的Nuclei Studio 2023.10版本存在一些问题，我们进行了修正，并于2023.11.17 13:30替换线上2023.10版本。

问题描述¶

2023年11月06日发布的Nuclei Studio 2023.10版本中存在一些问题,影响用户使用:

- build tools的busybox存在问题导致make 带 pre- post- steps时编译出问题
- Nuclei Settings中corner cases在特定场景下会出错
- Nuclei Settings的打开方式影响工程中其他文件的打开方式
- 在QEMU中使用V扩展时, 没有传入RVV length
- 修复打开一个全新的workspace, 创建新的工程的时候, 能够创建同名项目的问题, 重开workspace即可解决这个问题

我们重新做了一些变更, 以修复以上问题 :

- 修改并发布Nuclei Studio Plugins 2.1.0, 上传到插件更新网站
- 修改并发布Windows build-tools 1.2, 替换了线上的Windows Build Tools 2023.10
- 发布了新的Nuclei Studio 2023.10, 替换了线上的Nuclei Studio 2023.10

升级Nuclei Studio 2023.10 到最新版本的方法¶

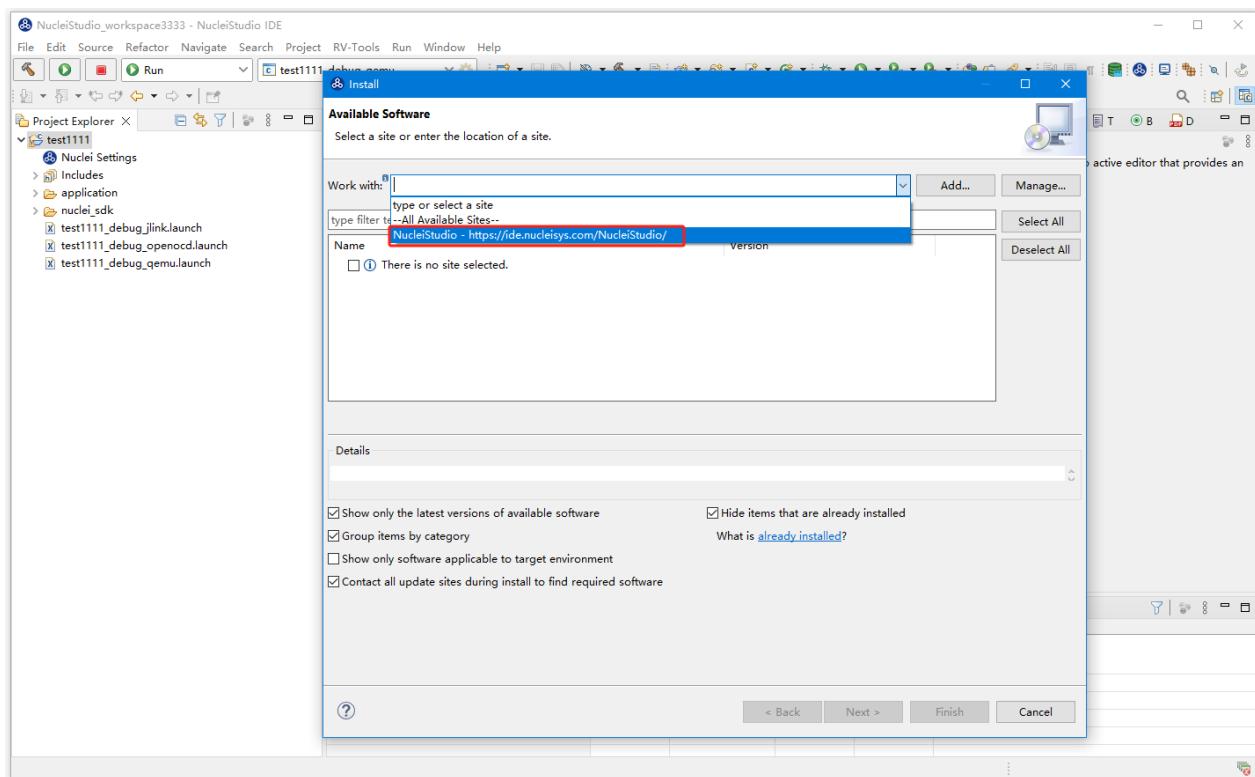
如果您的Nuclei Studio 2023.10, 是在2023年11月18日之前下载, 版本中存在的上述问题可能会引响您的使用体验, 您可以选择手动进行升级, 也可以选择重官网上下载我们最新发布的版本。

对2023年11月18日之前下载了Nuclei Studio 2023.10进行升级¶

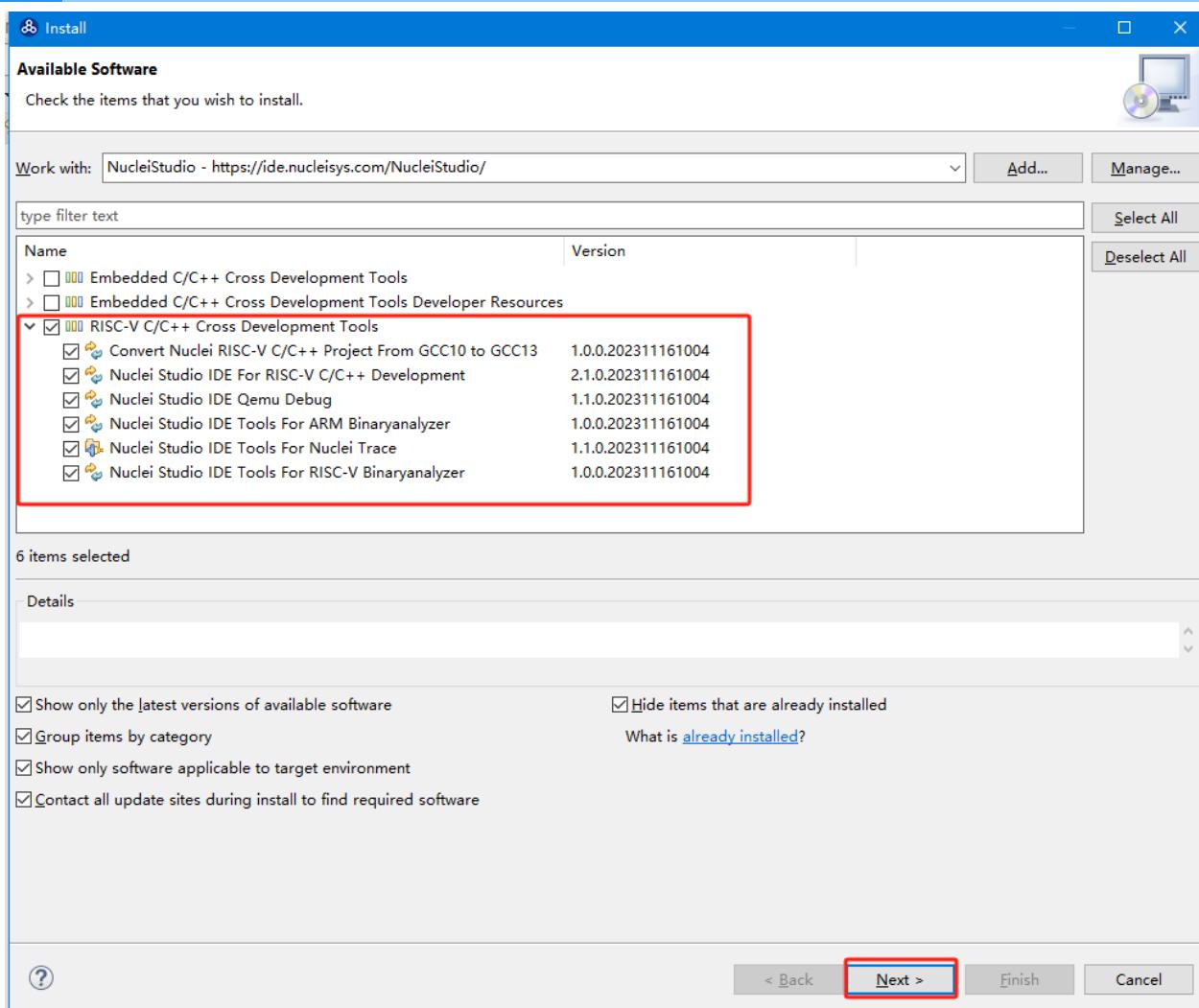
如果您是在2023年11月18日之前下载了Nuclei Studio 2023.10, 可以通过以下方式更新您的Nuclei Studio 2023.10 到最新版本

1. 升级Nuclei Studio Plugins

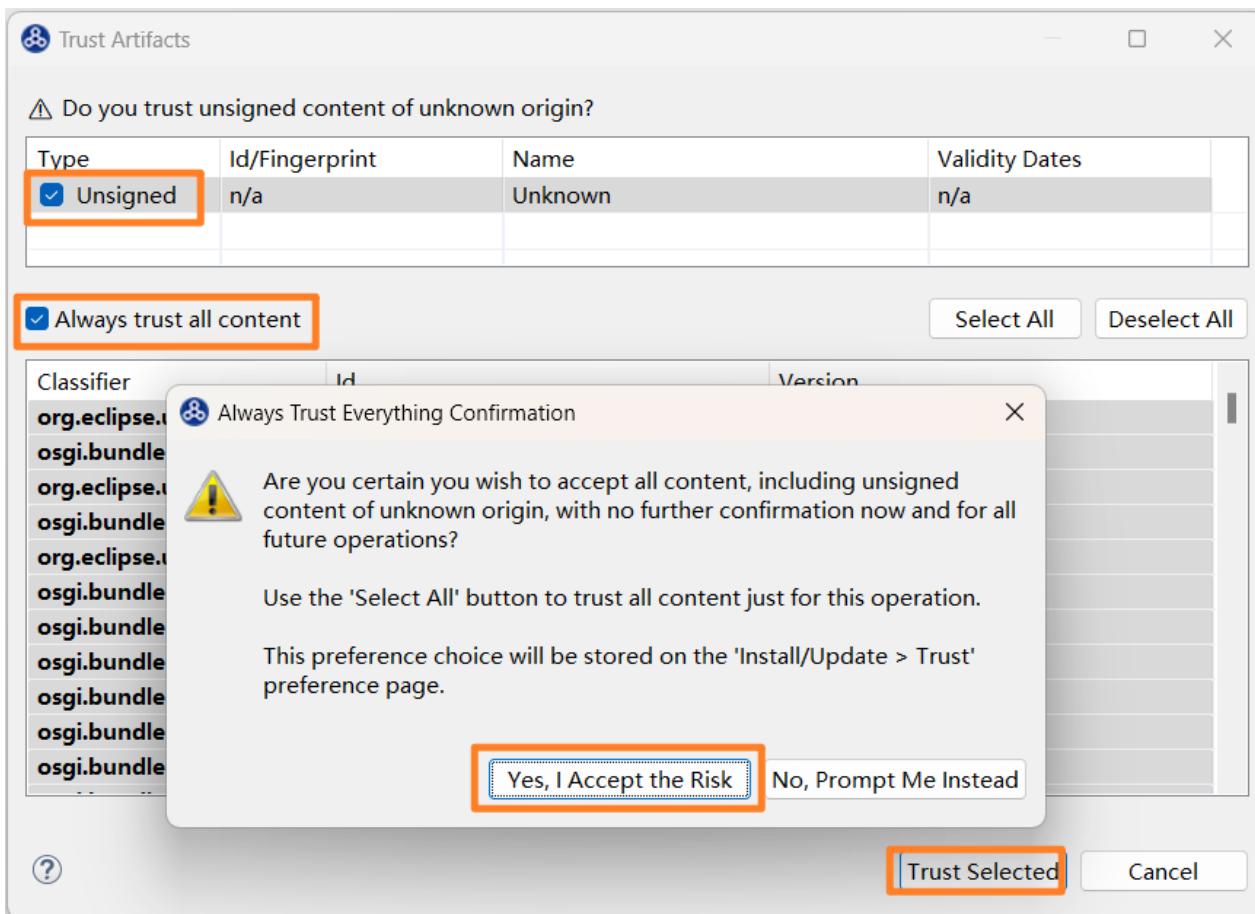
在Nuclei Studio菜单中找到Help->Install New Software, 然后在Install工具的Work with 选中NucleiStudio - <https://ide.nucleisys.com/NucleiStudio/>,下面会列出所有待更新的插件。



在弹出的插件列表中选中需要升级的插件，我们选中RISC-V C/C++ Cross Development Tools，然后Next。



在升级过程中，Nuclei Studio会询问Trust Artifacts时，操作如下图，选择Trust Selected，然后升级完成，Nuclei Studio会重启。至此Nuclei Studio Plugins升级完成。



2. 升级build-tools

Linux版本不需要执行此步骤，只需要确保系统中装了make工具就行。

下载build-tools-1.2，并替换Nuclei Studio 2023.10中的
NucleiStudio\toolchain\build-tools中内容。

关于这部分，可以查阅[编译工程时，使用了Pre-build Command/Post-build Command时报错](#)中的详细说明。

- [build-tools-1.2下载](#)

经此两步，完成了对Nuclei Studio 2023.10的升级。

从官网下载最新的版本¶

如果不想做手动升级工作，可以直接从我们的网站上下载最新的Nuclei Studio 2023.10。

- [Windows版下载](#)
- [Linux版下载](#)

参考资料¶

- [Nuclei Studio FAQs](#)
- [Nuclei Studio/Tools 不断更新的补充文档](#)
- [Nuclei Studio Issues](#)

OpenOCD在操作容量大于16M-Byte的nor-flash时的问题¶

问题说明¶

操作0 ~ 16M地址区间spi控制器需要发送三个字节的地址信息，称为3byte地址模式；操作16M ~ 2G地址区间spi控制器则需要发送四个字节的地址信息，称为4byte地址模式；

nuspi控制器的普通spi和xip默认都是3byte地址模式

解决方案¶

我们在OpenOCD里开发了两组spi驱动分别是nuspi和custom，都可以支持3byte模式和4byte模式，其中nuspi可通过判断操作地址，自动切换模式

在OpenOCD里有很多种方式可以read/verify flash内的数据，可以归结为两大类，一类是直接通过xip的方式读取flash数据，另一类则是通过调用驱动使用普通spi的方式读取flash数据。

因此，直接通过xip的方式读取flash数据时，就会有只能读到前面16M地址范围的限制，这样的命令有

- flash verify_image filename [offset] [type]
- dump_image filename address size
- gdb的x命令
- 等等直接读取memory的命令

当然OpenOCD里面也存在一些读取flash的命令，会直接调用cfg文件注册的spi驱动，这样的命令有

- flash read_bank num filename [offset [length]]
- flash verify_bank num filename [offset]

通过修改.cproject文件，升级工程工具链到GCC 13¶

问题描述

Nuclei Studio 2023.10的IDE进行了一次大版本的升级，其中自带的工具链从gcc10升级到了gcc13，并且工具链的前缀也发生了变化。参见 <https://github.com/Nuclei-Software/nuclei-studio/releases/tag/2023.10>

虽然我们在2023.10的IDE中提供了右键选中工程一键升级的工具（参见IDE的手册第8章节），但是这个只能一个工程一个工程的转换，对于有大量工程需要批量转换的项目而言不太友好，因此我们这里列出来如果写脚本进行工程的转换升级，则可以参考如下的思路进行转换。

以下变更仅针对Nuclei Studio 2023.10之前版本创建的gcc10的工程，进行升级变更，如果需要批量变更，编写脚本的时候应先检查工程是否是riscv gcc10的工程。

修改toolchain相关配置

在Nuclei Studio 2023.10之前的版本中使用的gcc是做了许多个性化的变更，需要Nuclei Studio 2023.10版中使用的gcc,继承了官方版本的特性和一些命名方式，在工程中的.cproject文件中，主要是要修改以下几个值。其中

ilg.gnumcueclipse.managedbuild.cross.riscv.option.toolchain.name的值是
RISC-V Nuclei GCC 、

ilg.gnumcueclipse.managedbuild.cross.riscv.option.toolchain.id的值是
3901352267

`ilg.gnumcueclipse.managedbuild.cross.riscv.option.command.prefix`的值是
`riscv-nuclei-elf-`,则说明工程在创建时所使用的是GCC 10。如果需要使工程支持GCC 13,
需要进行如下变更：

- toolchain.name的值从RISC-V Nuclei GCC变更为RISC-V GCC/Newlib
 - toolchain.id的值从3901352267变更为2262347901
 - command.prefix的值从riscv-nuclei-elf-变更为riscv64-unknown-elf-

变更前.cproject文件的内容

```
<option  
id="ilg.gnumcueclipse.managedbuild.cross.riscv.option.toolchain.name.  
129748485"  
superClass="ilg.gnumcueclipse.managedbuild.cross.riscv.option.toolchain.name"  
value="RISC-V Nuclei GCC" valueType="string"/>
```

```
<option  
id="ilg.gnumcueclipse.managedbuild.cross.riscv.option.toolchain.id.  
1143901706"  
superClass="ilg.gnumcueclipse.managedbuild.cross.riscv.option.toolchain.id"  
value="3901352267" valueType="string"/>  
<option  
id="ilg.gnumcueclipse.managedbuild.cross.riscv.option.command.prefix.  
1270840820"  
superClass="ilg.gnumcueclipse.managedbuild.cross.riscv.option.command.prefix"  
value="riscv-nuclei-elf-" valueType="string"/>  
<option  
id="ilg.gnumcueclipse.managedbuild.cross.riscv.option.command.c.  
718590769"  
superClass="ilg.gnumcueclipse.managedbuild.cross.riscv.option.command.c"  
value="gcc" valueType="string"/>  
<option  
id="ilg.gnumcueclipse.managedbuild.cross.riscv.option.command.cpp.  
243660928"  
superClass="ilg.gnumcueclipse.managedbuild.cross.riscv.option.command.cpp"  
value="g++" valueType="string"/>  
<option  
id="ilg.gnumcueclipse.managedbuild.cross.riscv.option.command.ar.  
416250093"  
superClass="ilg.gnumcueclipse.managedbuild.cross.riscv.option.command.ar"  
value="ar" valueType="string"/>  
<option  
id="ilg.gnumcueclipse.managedbuild.cross.riscv.option.command.objcopy.  
741068581"  
superClass="ilg.gnumcueclipse.managedbuild.cross.riscv.option.command.objcopy"  
value="objcopy" valueType="string"/>  
<option  
id="ilg.gnumcueclipse.managedbuild.cross.riscv.option.command.objdump.  
1474975752"  
superClass="ilg.gnumcueclipse.managedbuild.cross.riscv.option.command.objdump"  
value="objdump" valueType="string"/>  
<option  
id="ilg.gnumcueclipse.managedbuild.cross.riscv.option.command.size.  
2085350427"  
superClass="ilg.gnumcueclipse.managedbuild.cross.riscv.option.command.size"  
value="size" valueType="string"/>  
<option  
id="ilg.gnumcueclipse.managedbuild.cross.riscv.option.command.make.  
1355881376"  
superClass="ilg.gnumcueclipse.managedbuild.cross.riscv.option.command.make"  
value="make" valueType="string"/>  
<option  
id="ilg.gnumcueclipse.managedbuild.cross.riscv.option.command.rm.
```

```
1330665916"
superClass="ilg.gnumcueclipse.managedbuild.cross.riscv.option.command.rm"
value="rm" valueType="string"/>
```

变更后.cproject文件的内容

```
<option
id="ilg.gnumcueclipse.managedbuild.cross.riscv.option.toolchain.name.
129748485"
superClass="ilg.gnumcueclipse.managedbuild.cross.riscv.option.toolchain.name"
value="RISC-V GCC/Newlib" valueType="string"/>
<option
id="ilg.gnumcueclipse.managedbuild.cross.riscv.option.toolchain.id.
1143901706"
superClass="ilg.gnumcueclipse.managedbuild.cross.riscv.option.toolchain.id"
value="2262347901" valueType="string"/>
<option
id="ilg.gnumcueclipse.managedbuild.cross.riscv.option.command.prefix.
1270840820"
superClass="ilg.gnumcueclipse.managedbuild.cross.riscv.option.command.prefix"
value="riscv64-unknown-elf-" valueType="string"/>
<option
id="ilg.gnumcueclipse.managedbuild.cross.riscv.option.command.c.
718590769"
superClass="ilg.gnumcueclipse.managedbuild.cross.riscv.option.command.c"
value="gcc" valueType="string"/>
<option
id="ilg.gnumcueclipse.managedbuild.cross.riscv.option.command.cpp.
243660928"
superClass="ilg.gnumcueclipse.managedbuild.cross.riscv.option.command.cpp"
value="g++" valueType="string"/>
<option
id="ilg.gnumcueclipse.managedbuild.cross.riscv.option.command.ar.
416250093"
superClass="ilg.gnumcueclipse.managedbuild.cross.riscv.option.command.ar"
value="ar" valueType="string"/>
<option
id="ilg.gnumcueclipse.managedbuild.cross.riscv.option.command.objcopy.
741068581"
superClass="ilg.gnumcueclipse.managedbuild.cross.riscv.option.command.objcopy"
value="objcopy" valueType="string"/>
<option
id="ilg.gnumcueclipse.managedbuild.cross.riscv.option.command.objdump.
1474975752"
superClass="ilg.gnumcueclipse.managedbuild.cross.riscv.option.command.objdump"
value="objdump" valueType="string"/>
```

```

<option
  id="ilg.gnumcueclipse.managedbuild.cross.riscv.option.command.size.
  2085350427"
  superClass="ilg.gnumcueclipse.managedbuild.cross.riscv.option.command.size"
  value="size" valueType="string"/>
<option
  id="ilg.gnumcueclipse.managedbuild.cross.riscv.option.command.make.
  1355881376"
  superClass="ilg.gnumcueclipse.managedbuild.cross.riscv.option.command.make"
  value="make" valueType="string"/>
<option
  id="ilg.gnumcueclipse.managedbuild.cross.riscv.option.command.rm.
  1330665916"
  superClass="ilg.gnumcueclipse.managedbuild.cross.riscv.option.command.rm"
  value="rm" valueType="string"/>

```

修改RISC-V扩展相关配置¶

在Nuclei Studio 2023.10之前的版创建的工程中，RISC-V扩展是存放在四个单独的boolean类型的值中，而在Nuclei Studio 2023.10创建的工程中，改为一个string类型的值中,所以在要在工程的.cproject文件中找到四个旧的值，并按规则转换成为新的RISC-V扩展的字符串，存放到`ilg.gnumcueclipse.managedbuild.cross.riscv.option.target.isa.extensions`中，同时将旧的四个单独的boolean类型的值置空或者删除。

```

# 四个单独的boolean类型的值
ilg.gnumcueclipse.managedbuild.cross.riscv.option.target.isa.extensions.rvb
ilg.gnumcueclipse.managedbuild.cross.riscv.option.target.isa.extensions.rvk
ilg.gnumcueclipse.managedbuild.cross.riscv.option.target.isa.extensions.dsp
ilg.gnumcueclipse.managedbuild.cross.riscv.option.target.isa.extensions.vector

# 一个string类型的值
ilg.gnumcueclipse.managedbuild.cross.riscv.option.target.isa.extensions

```

1. 首先，根据

`ilg.gnumcueclipse.managedbuild.cross.riscv.option.target.isa.base`
确认工程对应的arch是rv32/rv64

2. 其次，根据

`ilg.gnumcueclipse.managedbuild.cross.riscv.option.target.isa.fp`确认是否带f/d

3. 最后，根据对应转换规则转换出正确的RISC-V扩展字符串

转换规则(特别说明，p的值需要接在RISC-V扩展字符串的最后)：

- b -> _zba_zbb_zbc_zbs
- k -> _zk_zks
- v -> rv32f/d : _zve32f, rv64f: _zve64f, rv64fd: v
- p -> rv64: _xxldsp, rv32: _xxldspn1x

例如，现在有一个N307FD的工程，它的`arch=rv32imafdcbpv(gcc10)`，可以知道它是一个rv32,带fd并且使用了bpv扩展，那么根据转换规则，转换出来的RISC-V扩展字符串为`_zba_zbb_zbc_zbs_zve32f_xxldspn1x`。

变更前.cproject文件的内容

```
<option
  id="ilg.gnumcueclipse.managedbuild.cross.riscv.option.target.isa.base.
  489743203"
  superClass="ilg.gnumcueclipse.managedbuild.cross.riscv.option.target.isa.base"
  value="ilg.gnumcueclipse.managedbuild.cross.riscv.option.target.arch.rv32i"
  valueType="enumerated"/>
<option
  id="ilg.gnumcueclipse.managedbuild.cross.riscv.option.target.isa.fp.
  1936924005"
  superClass="ilg.gnumcueclipse.managedbuild.cross.riscv.option.target.isa.fp"
  value="ilg.gnumcueclipse.managedbuild.cross.riscv.option.isa.fp.double"
  valueType="enumerated"/>
<option
  id="ilg.gnumcueclipse.managedbuild.cross.riscv.option.target.isa.extension.rvb.
  168405526"
  superClass="ilg.gnumcueclipse.managedbuild.cross.riscv.option.target.isa.extension"
  value="true" valueType="boolean"/>
<option
  id="ilg.gnumcueclipse.managedbuild.cross.riscv.option.target.isa.extension.dsp.
  565204765"
  superClass="ilg.gnumcueclipse.managedbuild.cross.riscv.option.target.isa.extension"
  value="true" valueType="boolean"/>
<option
  id="ilg.gnumcueclipse.managedbuild.cross.riscv.option.target.isa.extension.vector.
  1142078455"
  superClass="ilg.gnumcueclipse.managedbuild.cross.riscv.option.target.isa.extension"
  value="true" valueType="boolean"/>
```

变更后.cproject文件的内容

```
<option
  id="ilg.gnumcueclipse.managedbuild.cross.riscv.option.target.isa.extensions.
  1832321358"
  superClass="ilg.gnumcueclipse.managedbuild.cross.riscv.option.target.isa.extensions.
  value=_zba_zbb_zbc_zbs_zve32f_xxldspn1x" valueType="string"/>
```

修改libncrt C库相关配置

在Nuclei Studio 2023.10之前的版创建的工程中,使用libncrt C库时,会在工程中包含一个--specs=libncrt_xxx.specs或者链接库里面包含-lncrt_xxx,而在Nuclei Studio 2023.10创建的工程中,如果使用了libncrt C库,需要将--specs=libncrt_xxx.specs的方式变更为-lncrt_xxx,然后额外需要链接的时候补上-lncrt_small -lheapops_basic -lfileops_uart,通用的target编译选项需要补上-isystem=/include/libncrt

举例如下：* -lncrt_small->-lncrt_small -lheapops_basic -lfileops_uart * --specs=libncrt_small.specs ->-lncrt_small -lheapops_basic -lfileops_uart

1. 在.cproject文件中确认否存在--specs=libncrt_xxx.specs, 如果存在, 则表示这个是一个使用了libncrt的工程, 则可以进行后续的步骤
2. 如果--specs=libncrt_xxx.specs存在, 先将其删除
3. 如果-lm存在, 则先将其删除
4. 查找ilg.gnumcueclipse.managedbuild.cross.riscv.option.c.linker.libs或者ilg.gnumcueclipse.managedbuild.cross.riscv.option.cpp.linker.libs中是否存m, 如果存在则先将删除
5. 查找ilg.gnumcueclipse.managedbuild.cross.riscv.option.c.linker.libs或者ilg.gnumcueclipse.managedbuild.cross.riscv.option.cpp.linker.libs中是否存ncrt_xxx
6. 根据上面的结果, 在ilg.gnumcueclipse.managedbuild.cross.riscv.option.c.linker.libs或者ilg.gnumcueclipse.managedbuild.cross.riscv.option.cpp.linker.libs中补充对应的值
 - --specs=libncrt_xxx.specs存在, 添加ncrt_xxx; 或者ncrt_xxx存在。需要额外添加heapops_basic和fileops_uart
7. 在ilg.gnumcueclipse.managedbuild.cross.riscv.option.target.other中补上-isystem=/include/libncrt

```
# --specs=libncrt_xxx1.specs可能存在于以下string类型的值
ilg.gnumcueclipse.managedbuild.cross.riscv.option.target.other
ilg.gnumcueclipse.managedbuild.cross.riscv.option.optimization.other
ilg.gnumcueclipse.managedbuild.cross.riscv.option.warnings.other
ilg.gnumcueclipse.managedbuild.cross.riscv.option.debugging.other
ilg.gnumcueclipse.managedbuild.cross.riscv.option.assembler.otherwarnings
ilg.gnumcueclipse.managedbuild.cross.riscv.option.assembler.other
ilg.gnumcueclipse.managedbuild.cross.riscv.option.c.compiler.otheroptimizations
ilg.gnumcueclipse.managedbuild.cross.riscv.option.c.compiler.otherwarnings
ilg.gnumcueclipse.managedbuild.cross.riscv.option.c.compiler.other
ilg.gnumcueclipse.managedbuild.cross.riscv.option.c.linker.other
ilg.gnumcueclipse.managedbuild.cross.riscv.option.cpp.compiler.otheroptimizations
ilg.gnumcueclipse.managedbuild.cross.riscv.option.cpp.compiler.otherwarnings
ilg.gnumcueclipse.managedbuild.cross.riscv.option.cpp.compiler.other
ilg.gnumcueclipse.managedbuild.cross.riscv.option.cpp.linker.other
ilg.gnumcueclipse.managedbuild.cross.riscv.option.createflash.other
ilg.gnumcueclipse.managedbuild.cross.riscv.option.createlisting.other
ilg.gnumcueclipse.managedbuild.cross.riscv.option.printsize.other
```

举例，工程中用到了--specs=libncrt_balanced.specs

变更前.cproject文件的内容

```
<option
  id="ilg.gnumcueclipse.managedbuild.cross.riscv.option.target.other.1735566114"
  superClass="ilg.gnumcueclipse.managedbuild.cross.riscv.option.target.other"
  value=" " valueType="string"/>
<option
  id="ilg.gnumcueclipse.managedbuild.cross.riscv.option.optimization.other.443378574"
  superClass="ilg.gnumcueclipse.managedbuild.cross.riscv.option.optimization.other"
  value="--specs=libncrt_balanced.specs" valueType="string"/>
```

变更后.cproject文件的内容

```

<option
  id="ilg.gnumcueclipse.managedbuild.cross.riscv.option.target.other.
  1735566114"
  superClass="ilg.gnumcueclipse.managedbuild.cross.riscv.option.target.other"
  value="-isystem=/include/libcrt" valueType="string"/>
<option IS_BUILTIN_EMPTY="false" IS_VALUE_EMPTY="false"
  id="ilg.gnumcueclipse.managedbuild.cross.riscv.option.cpp.linker.libs.
  146128417"
  superClass="ilg.gnumcueclipse.managedbuild.cross.riscv.option.cpp.linker.libs"
  valueType="libs">
  <listOptionValue builtIn="false" value="ncrt_balanced"/>
  <listOptionValue builtIn="false" value="fileops_uart"/>
  <listOptionValue builtIn="false" value="heapops_basic"/>
</option>

```

增加link warning消除的配置¶

在GCC 13使用过程中会产生很多的warning信息，可以在链接选项中额外增加-Wl,--no-warn-rwx-segments参数，用以关闭这些warning信息。

具体参见

https://sourceware.org/binutils/docs/ld/Options.html#index-_002d_002dwarn_002drwx_002dsegments

变更前.cproject文件的内容

```

<option
  id="ilg.gnumcueclipse.managedbuild.cross.riscv.option.cpp.linker.other.
  1000044097"
  superClass="ilg.gnumcueclipse.managedbuild.cross.riscv.option.cpp.linker.other"
  value="" valueType="string"/>

```

变更后.cproject文件的内容

```

<option
  id="ilg.gnumcueclipse.managedbuild.cross.riscv.option.cpp.linker.other.
  1000044097"
  superClass="ilg.gnumcueclipse.managedbuild.cross.riscv.option.cpp.linker.other"
  value="-Wl,--no-warn-rwx-segments" valueType="string"/>

```

完成以上变更后，reload一下工程，工程就可以在Nuclei Studio 2023.10下正常编译、调试、运行了。

说明：

本文档中，所有引用的例子中关于.cproject文件，出现的类似
id="ilg.gnumcueclipse.managedbuild.cross.riscv.option.target.o
ther.1735566114"中，1735566114是一个Nuclei Studio生成的hash值，不同时间不同工程各不相同，且其不影响配置，如果能保持与原值相同的情况下，尽量保持相同。

在Nuclei Studio下用命令行编译工程¶

问题说明¶

很多客户咨询怎么在Nuclei Studio上使用IDE的无头Headless模式来构建和编译工程。

解决方案¶

所有以 NucleiStudio.exe 开头的命令行执行时，会有一个弹框显示执行日志，如果需屏蔽弹框，可以将命令改为 eclipse.exe

以下文档是在2024.06版本的IDE中实测，作为补充说明。

因NucleiStudio 2024.06版运行在java 21的环境上，实际应用中很多用户的本地没有java 21环境，故在运行命令时发现在执行该命令时，因找不到对应的jre而报错。为解决上述问题，可以在本地机器上安装java 21的环境（如何安装用户可以自行搜索相关教程），也可以在命令行中通过 -vm 参数指定NucleiStudio 2024.06中自带的jre的路径。

```
NucleiStudio.exe -vm "<user_nucleistudio_path>/plugins/
org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_21.0.3.v202
40426-1530/jre/bin" --launcher.suppressErrors -nosplash -
application org.eclipse.cdt.managedbuilder.core.headlessbuild -
data C:\NucleiStudio_workspace -cleanBuild test/Debug -Debug
```

提供一组批量导入工程并批量编译工程的命令

创建workspace并批量导入工程

```
NucleiStudio.exe -vm "<user_nucleistudio_path>/plugins/
org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_21.0.3.v202
40426-1530/jre/bin" --launcher.suppressErrors -noSplash -
application org.eclipse.cdt.managedbuilder.core.headlessbuild -
data $CI_PROJECT_DIR -importAll $CI_PROJECT_DIR
```

编译这组导入的工程

```
NucleiStudio.exe -vm "<user_nucleistudio_path>/plugins/
org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_21.0.3.v202
40426-1530/jre/bin" --launcher.suppressErrors -noSplash -
```

```
application org.eclipse.cdt.managedbuilder.core.headlessbuild -  
data $CI_PROJECT_DIR -build ${TARGET_PHASE}_Project/Debug
```

以下文档是在2023.10版本的IDE中实测，其他版本可能需要做一些调整适配才可以正常工作。

Nuclei Studio是图形化（GUI）的代码编写工具，但是在某些特定的场景下，用户需要通过命令行来快速编译工程，在Nuclei Studio中，只需要一行命令就可以实现。下载好Nuclei Studio后，在Nuclei Studio的workspace已经创建好了需要编译的工程test，同时Nuclei Studio已退出运行，执行以下命令就可以完成工程的编译。

提醒：请确保 NucleiStudio的PATH已经设置到系统中，这样 NucleiStudio.exe/NucleiStudio 才可以被执行。

下面以Windows系统举例

```
NucleiStudio.exe --launcher.suppressErrors -nosplash -application  
org.eclipse.cdt.managedbuilder.core.headlessbuild -data C:  
\NucleiStudio_workspace -cleanBuild test/Debug -Debug
```

--launcher.suppressErrors 用来屏蔽构建出错时，Eclipse会出错弹窗。

如果需要在2022.12版本的IDE上进行使用，则需要先设置好toolchain目录下gcc/bin和build-tools/bin的路径到系统PATH中，然后将NucleiStudio.exe换成eclipsec.exe

针对2022.12版本，命令举例如下：

```
# 这里请修改成自己的IDE路径  
set NSIDE=D:\NucleiStudio_IDE_202212-win64\NucleiStudio  
# 必须设置好系统PATH  
set PATH=%NSIDE%\toolchain\gcc\bin;%NSIDE%\toolchain\build-  
tools\bin;%PATH%  
# 注意NucleiStudio.exe换成了eclipsec.exe  
%NSIDE%\eclipsec.exe --launcher.suppressErrors -nosplash -  
application org.eclipse.cdt.managedbuilder.core.headlessbuild -  
data C:\NucleiStudio_workspace -cleanBuild test/Debug
```

这个2023.10版本的举例的命令会弹出一个额外的命令行窗口进行输出。

```

E:\NucleiStudio_IDE_202310-win64\NucleiStudio>NucleiStudio.exe --launcher.suppressErrors -n
osplash -application org.eclipse.cdt.managedbuilder.core.headlessbuild -data C:\Users\11653\NucleiStudio_workspace2023
-cleanBuild test2222/Debug -Debug

& NucleiStudio.exe --launcher.suppressErrors -nosplash -application org.eclipse.cdt.managedbuilder.core.headlessbu... - 

sdk/SoC/hbirdv2/Common/Source/Stubs/lseek.o ./hbird_sdk/SoC/hbirdv2/Common/Source/Stubs/open.o ./hbird_sdk/SoC/hbirdv2/C
ommon/Source/Stubs/read.o ./hbird_sdk/SoC/hbirdv2/Common/Source/Stubs/sbrk.o ./hbird_sdk/SoC/hbirdv2/Common/Source/Stubs
/stat.o ./hbird_sdk/SoC/hbirdv2/Common/Source/Stubs/times.o ./hbird_sdk/SoC/hbirdv2/Common/Source/Stubs/unlink.o ./hbird
_sdk/SoC/hbirdv2/Common/Source/Stubs/wait.o ./hbird_sdk/SoC/hbirdv2/Common/Source/Stubs/write.o ./hbird_sdk/SoC/hbirdv2
/Common/Source/GCC/intexc_hbirdv2.o ./hbird_sdk/SoC/hbirdv2/Common/Source/GCC/startup_hbirdv2.o ./hbird_sdk/SoC/hbirdv2
/Common/Source/Drivers/hbirdv2_gpio.o ./hbird_sdk/SoC/hbirdv2/Common/Source/Drivers/hbirdv2_i2c.o ./hbird_sdk/SoC/hbirdv
2/Common/Source/Drivers/hbirdv2_pwm.o ./hbird_sdk/SoC/hbirdv2/Common/Source/Drivers/hbirdv2_spi.o ./hbird_sdk/SoC/hbirdv
2/Common/Source/Drivers/hbirdv2_uart.o ./hbird_sdk/SoC/hbirdv2/Common/Source/Drivers/htif.o ./hbird_sdk/SoC/hbirdv2/Com
mon/Source/hbirdv2_common.o ./hbird_sdk/SoC/hbirdv2/Common/Source/system_hbirdv2.o ./application/main.o test2222_hex te
st2222.lst test2222.siz ./hbird_sdk/SoC/hbirdv2/Common/Source/GCC/intexc_hbirdv2.d ./hbird_sdk/SoC/hbirdv2/Common/Source
/GCC/startup_hbirdv2.d ./hbird_sdk/SoC/hbirdv2/Common/Source/Stubs/clock_getres.d ./hbird_sdk/SoC/hbirdv2/Common/Source/
Stubs/clock_gettime.d ./hbird_sdk/SoC/hbirdv2/Common/Source/Stubs/clock_settime.d ./hbird_sdk/SoC/hbirdv2/Common/Source/
Stubs/close.d ./hbird_sdk/SoC/hbirdv2/Common/Source/Stubs/execve.d ./hbird_sdk/SoC/hbirdv2/Common/Source/Stubs/exit.d ./
hbird_sdk/SoC/hbirdv2/Common/Source/Stubs/fork.d ./hbird_sdk/SoC/hbirdv2/Common/Source/Stubs/fstat.d ./hbird_sdk/SoC/hbi
rdv2/Common/Source/Stubs/getpid.d ./hbird_sdk/SoC/hbirdv2/Common/Source/Stubs/gettimeofday.d ./hbird_sdk/SoC/hbirdv2/Com
mon/Source/Stubs/isatty.d ./hbird_sdk/SoC/hbirdv2/Common/Source/Stubs/kill.d ./hbird_sdk/SoC/hbirdv2/Common/Source/Stubs
/link.d ./hbird_sdk/SoC/hbirdv2/Common/Source/Stubs/lseek.d ./hbird_sdk/SoC/hbirdv2/Common/Source/Stubs/open.d ./hbird_s
dk/SoC/hbirdv2/Common/Source/Stubs/read.d ./hbird_sdk/SoC/hbirdv2/Common/Source/Stubs/sbrk.d ./hbird_sdk/SoC/hbirdv2/Com
mon/Source/Stubs/stat.d ./hbird_sdk/SoC/hbirdv2/Common/Source/Stubs/times.d ./hbird_sdk/SoC/hbirdv2/Common/Source/Stubs/
unlink.d ./hbird_sdk/SoC/hbirdv2/Common/Source/Stubs/wait.d ./hbird_sdk/SoC/hbirdv2/Common/Source/Stubs/write.d ./hbird
_sdk/SoC/hbirdv2/Common/Source/Drivers/hbirdv2_gpio.d ./hbird_sdk/SoC/hbirdv2/Common/Source/Drivers/hbirdv2_i2c.d ./hbir
d_sdk/SoC/hbirdv2/Common/Source/Drivers/hbirdv2_pwm.d ./hbird_sdk/SoC/hbirdv2/Common/Source/Drivers/hbirdv2_spi.d ./hbir
d_sdk/SoC/hbirdv2/Common/Source/Drivers/hbirdv2_uart.d ./hbird_sdk/SoC/hbirdv2/Common/Source/Drivers/htif.d ./hbird_sdk
/SoC/hbirdv2/Common/Source/hbirdv2_common.d ./hbird_sdk/SoC/hbirdv2/Common/Source/system_hbirdv2.d ./application/main.d
test2222.elf

15:34:36 Build Finished. 0 errors, 0 warnings. (took 326ms)

```

- **NucleiStudio.exe**：该参数是Nuclei Studio的启动应用，在Nuclei Studio的安装目录下。
- **--launcher.suppressErrors**：该参数是用于抑制Nuclei Studio启动时的错误信息。
- **-nosplash**：该参数用于关闭启动时的 Splash 屏幕。这意味着在启动 Eclipse 时不会显示一个短暂的加载屏幕。
- **-application**：该参数用于指定要运行的应用程序。在这里，**org.eclipse.cdt.managedbuilder.core.headlessbuild** 是指 Headless 构建应用程序。该应用程序用于执行构建操作，而不需要图形用户界面（GUI）。
- **-data**：该参数用于指定工作区路径。它告诉 Nuclei Studio 将数据存储在哪里，例如工作空间、项目和文件。
- **-build**：该参数用于指定需要编译的工程，**test/Debug**，表示的是编译**test**工程中的 Debug 配置；一般Nuclei Studio创建的工程有Debug、Release两套配置，如果不指定配置，这个默认会编译出Debug、Release，可以看到编译后工程目录下有Debug、Release两个目录。

```

├-.settings
└application
├Debug
│└application
│└nuclei_sdk
└nuclei_sdk
└Release

```

```
└─application  
  └─nuclei_sdk
```

- **-cleanBuild**：该参数与**-build**类似，只是在编译之前，会清空清理工作空间。建议使用**-cleanBuild**。
- **-Debug**：该参数用于指定编译过程是Debug模式，在编译时会输出详细的编译过程日志。如果不带此参数，命令将静默执行，没有任何输出。

以下为上面举例命令的输出内容，以供参考

```
17:00:17 **** Clean-only build of configuration Debug for project  
test ****  
make -j8 clean  
rm -rf ./nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/  
chown.o ./nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/  
clock_getres.o ./nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/  
clock_gettime.o ./nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/  
newlib/clock_settime.o ./nuclei_sdk/SoC/evalsoc/Common/Source/  
Stubs/newlib/close.o ./nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/  
newlib/environ.o ./nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/  
newlib/errno.o ./nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/  
execve.o ./nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/  
exit.o ./nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/  
fork.o ./nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/  
fstat.o ./nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/  
getpid.o ./nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/  
gettimeofday.o ./nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/  
isatty.o ./nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/  
kill.o ./nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/  
link.o ./nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/  
lseek.o ./nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/  
open.o ./nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/  
read.o ./nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/  
readlink.o ./nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/  
sbrk.o ./nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/  
stat.o ./nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/  
symlink.o ./nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/  
times.o ./nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/  
unlink.o ./nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/  
wait.o ./nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/  
write.o ./nuclei_sdk/SoC/evalsoc/Common/Source/GCC/  
intexc_evalsoc.o ./nuclei_sdk/SoC/evalsoc/Common/Source/GCC/  
intexc_evalsoc_s.o ./nuclei_sdk/SoC/evalsoc/Common/Source/GCC/  
startup_evalsoc.o ./nuclei_sdk/SoC/evalsoc/Common/Source/Drivers/  
evalsoc_uart.o ./nuclei_sdk/SoC/evalsoc/Common/Source/  
evalsoc_common.o ./nuclei_sdk/SoC/evalsoc/Common/Source/
```

```
system_evalsoc.o ./application/main.o test.hex test.lst
test.siz ./nuclei_sdk/SoC/evalsoc/Common/Source/GCC/
intexc_evalsoc.d ./nuclei_sdk/SoC/evalsoc/Common/Source/GCC/
intexc_evalsoc_s.d ./nuclei_sdk/SoC/evalsoc/Common/Source/GCC/
startup_evalsoc.d ./nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/
newlib/chown.d ./nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/
clock_getres.d ./nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/
clock_gettime.d ./nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/
newlib/clock_settime.d ./nuclei_sdk/SoC/evalsoc/Common/Source/
Stubs/newlib/close.d ./nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/
newlib/environ.d ./nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/
newlib/errno.d ./nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/
execve.d ./nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/
exit.d ./nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/
fork.d ./nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/
fstat.d ./nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/
getpid.d ./nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/
gettimeofday.d ./nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/
isatty.d ./nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/
kill.d ./nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/
link.d ./nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/
lseek.d ./nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/
open.d ./nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/
read.d ./nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/
readlink.d ./nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/
sbrk.d ./nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/
stat.d ./nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/
symlink.d ./nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/
times.d ./nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/
unlink.d ./nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/
wait.d ./nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/
write.d ./nuclei_sdk/SoC/evalsoc/Common/Source/Drivers/
evalsoc_uart.d ./nuclei_sdk/SoC/evalsoc/Common/Source/
evalsoc_common.d ./nuclei_sdk/SoC/evalsoc/Common/Source/
system_evalsoc.d ./application/main.d test.elf
```

17:00:17 Build Finished. 0 errors, 0 warnings. (took 371ms)

```
17:00:18 **** Build of configuration Debug for project test ****
make -j8 all
Building file: ../nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/
newlib/chown.c
Building file: ../nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/
newlib/clock_getres.c
Building file: ../nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/
newlib/clock_gettime.c
```

```
Invoking: GNU RISC-V Cross C Compiler
Invoking: GNU RISC-V Cross C Compiler
riscv64-unknown-elf-gcc -march=rv32imafdc -mabi=ilp32d -
mtune=nuclei-300-series -mcmodel=medlow -msave-restore -isystem=/
include/newlib-nano -O2 -ffunction-sections -fdata-sections -fno-
common -g -D__IDE_RV_CORE=n307fd -DBOOT_HARTID=0 -DRUNMODE_IC_EN=0
-DRUNMODE_DC_EN=0 -DRUNMODE_CCM_EN=0 -
DDOWNLOAD_MODE=DOWNLOAD_MODE_ILM -DDOWNLOAD_MODE_STRING=\"ILM\" -
I"C:\NucleiStudio_workspace\test\nuclei_sdk\NMSIS\Core\Include" -
I"C:
\NucleiStudio_workspace\test\nuclei_sdk\SoC\evalsoc\Common\Include"
-I"C:
\NucleiStudio_workspace\test\nuclei_sdk\SoC\evalsoc\Board\nuclei_fp
ga_eval\Include" -I"C:\NucleiStudio_workspace\test\application" -
std=gnu11 -MMD -MP -MF"nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/
newlib/chown.d" -MT"nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/
newlib/chown.o" -c -o "nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/
newlib/chown.o" "../nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/
newlib/chown.c"
Building file: ../nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/
newlib/clock_settime.c
riscv64-unknown-elf-gcc -march=rv32imafdc -mabi=ilp32d -
mtune=nuclei-300-series -mcmodel=medlow -msave-restore -isystem=/
include/newlib-nano -O2 -ffunction-sections -fdata-sections -fno-
common -g -D__IDE_RV_CORE=n307fd -DBOOT_HARTID=0 -DRUNMODE_IC_EN=0
-DRUNMODE_DC_EN=0 -DRUNMODE_CCM_EN=0 -
DDOWNLOAD_MODE=DOWNLOAD_MODE_ILM -DDOWNLOAD_MODE_STRING=\"ILM\" -
I"C:\NucleiStudio_workspace\test\nuclei_sdk\NMSIS\Core\Include" -
I"C:
\NucleiStudio_workspace\test\nuclei_sdk\SoC\evalsoc\Common\Include"
-I"C:
\NucleiStudio_workspace\test\nuclei_sdk\SoC\evalsoc\Board\nuclei_fpga_eval\Inc
-I"C:\NucleiStudio_workspace\test\application" -std=gnu11 -MMD -MP
-MF"nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/
clock_getres.d" -MT"nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/
newlib/clock_getres.o" -c -o "nuclei_sdk/SoC/evalsoc/Common/Source/
Stubs/newlib/clock_getres.o" "../nuclei_sdk/SoC/evalsoc/Common/
Source/Stubs/newlib/clock_getres.c"
Invoking: GNU RISC-V Cross C Compiler
riscv64-unknown-elf-gcc -march=rv32imafdc -mabi=ilp32d -
mtune=nuclei-300-series -mcmodel=medlow -msave-restore -isystem=/
include/newlib-nano -O2 -ffunction-sections -fdata-sections -fno-
common -g -D__IDE_RV_CORE=n307fd -DBOOT_HARTID=0 -DRUNMODE_IC_EN=0
-DRUNMODE_DC_EN=0 -DRUNMODE_CCM_EN=0 -
DDOWNLOAD_MODE=DOWNLOAD_MODE_ILM -DDOWNLOAD_MODE_STRING=\"ILM\" -
I"C:\NucleiStudio_workspace\test\nuclei_sdk\NMSIS\Core\Include" -
I"C:
```

```
\NucleiStudio_workspace\test\nuclei_sdk\SoC\evalsoc\Common\Include"
-I"C:
\NucleiStudio_workspace\test\nuclei_sdk\SoC\evalsoc\Board\nuclei_fpga_eval\Inc
-I"C:\NucleiStudio_workspace\test\application" -std=gnu11 -MMD -MP
-MF"nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/
clock_gettime.d" -MT"nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/
newlib/clock_gettime.o" -c -o "nuclei_sdk/SoC/evalsoc/Common/
Source/Stubs/newlib/clock_gettime.o" "../nuclei_sdk/SoC/evalsoc/
Common/Source/Stubs/newlib/clock_gettime.c"
Invoking: GNU RISC-V Cross C Compiler
riscv64-unknown-elf-gcc -march=rv32imafdc -mabi=ilp32d -
mtune=nuclei-300-series -mcmodel=medlow -msave-restore -isystem=
/include/newlib-nano -O2 -ffunction-sections -fdata-sections -fno-
common -g -D__IDE_RV_CORE=n307fd -DBOOT_HARTID=0 -DRUNMODE_IC_EN=0
-DRUNMODE_DC_EN=0 -DRUNMODE_CCM_EN=0 -
DDOWNLOAD_MODE=DOWNLOAD_MODE_ILM -DDOWNLOAD_MODE_STRING=\"ILM\"
-I"C:\NucleiStudio_workspace\test\nuclei_sdk\NMSIS\Core\Include" -
I"C:
\NucleiStudio_workspace\test\nuclei_sdk\SoC\evalsoc\Common\Include"
-I"C:
\NucleiStudio_workspace\test\nuclei_sdk\SoC\evalsoc\Board\nuclei_fpga_eval\Inc
-I"C:\NucleiStudio_workspace\test\application" -std=gnu11 -MMD -MP
-MF"nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/
clock_settime.d" -MT"nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/
newlib/clock_settime.o" -c -o "nuclei_sdk/SoC/evalsoc/Common/
Source/Stubs/newlib/clock_settime.o" "../nuclei_sdk/SoC/evalsoc/
Common/Source/Stubs/newlib/clock_settime.c"
Building file: ../nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/
newlib/close.c
Building file: ../nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/
newlib/environ.c
Invoking: GNU RISC-V Cross C Compiler
Building file: ../nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/
newlib/errno.c
Building file: ../nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/
newlib/execve.c
riscv64-unknown-elf-gcc -march=rv32imafdc -mabi=ilp32d -
mtune=nuclei-300-series -mcmodel=medlow -msave-restore -isystem=
/include/newlib-nano -O2 -ffunction-sections -fdata-sections -fno-
common -g -D__IDE_RV_CORE=n307fd -DBOOT_HARTID=0 -DRUNMODE_IC_EN=0
-DRUNMODE_DC_EN=0 -DRUNMODE_CCM_EN=0 -
DDOWNLOAD_MODE=DOWNLOAD_MODE_ILM -DDOWNLOAD_MODE_STRING=\"ILM\"
-I"C:\NucleiStudio_workspace\test\nuclei_sdk\NMSIS\Core\Include" -
I"C:
\NucleiStudio_workspace\test\nuclei_sdk\SoC\evalsoc\Common\Include"
-I"C:
\NucleiStudio_workspace\test\nuclei_sdk\SoC\evalsoc\Board\nuclei_fpga_eval\Inc
```

```
-I"C:\NucleiStudio_workspace\test\application" -std=gnu11 -MMD -MP  
-MF"nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/close.d" -  
MT"nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/close.o" -c -o  
"nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/close.o" "../  
nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/close.c"  
Invoking: GNU RISC-V Cross C Compiler  
riscv64-unknown-elf-gcc -march=rv32imafdc -mabi=ilp32d -  
mtune=nuclei-300-series -mcmodel=medlow -msave-restore -isystem=/  
include/newlib-nano -O2 -ffunction-sections -fdata-sections -fno-  
common -g -D__IDE_RV_CORE=n307fd -DBOOT_HARTID=0 -DRUNMODE_IC_EN=0  
-DRUNMODE_DC_EN=0 -DRUNMODE_CCM_EN=0 -  
DDOWNLOAD_MODE=DOWNLOAD_MODE_ILM -DDOWNLOAD_MODE_STRING=\"ILM\" -  
I"C:\NucleiStudio_workspace\test\nuclei_sdk\NMSIS\Core\Include" -  
I"C:  
\NucleiStudio_workspace\test\nuclei_sdk\SoC\evalsoc\Common\Include"  
-I"C:  
\NucleiStudio_workspace\test\nuclei_sdk\SoC\evalsoc\Board\nuclei_fpga_eval\Inc  
-I"C:\NucleiStudio_workspace\test\application" -std=gnu11 -MMD -MP  
-MF"nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/environ.d" -  
MT"nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/environ.o" -c -o  
"nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/environ.o"  
"../nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/environ.c"  
Invoking: GNU RISC-V Cross C Compiler  
riscv64-unknown-elf-gcc -march=rv32imafdc -mabi=ilp32d -  
mtune=nuclei-300-series -mcmodel=medlow -msave-restore -isystem=/  
include/newlib-nano -O2 -ffunction-sections -fdata-sections -fno-  
common -g -D__IDE_RV_CORE=n307fd -DBOOT_HARTID=0 -DRUNMODE_IC_EN=0  
-DRUNMODE_DC_EN=0 -DRUNMODE_CCM_EN=0 -  
DDOWNLOAD_MODE=DOWNLOAD_MODE_ILM -DDOWNLOAD_MODE_STRING=\"ILM\" -  
I"C:\NucleiStudio_workspace\test\nuclei_sdk\NMSIS\Core\Include" -  
I"C:  
\NucleiStudio_workspace\test\nuclei_sdk\SoC\evalsoc\Common\Include"  
-I"C:  
\NucleiStudio_workspace\test\nuclei_sdk\SoC\evalsoc\Board\nuclei_fpga_eval\Inc  
-I"C:\NucleiStudio_workspace\test\application" -std=gnu11 -MMD -MP  
-MF"nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/errno.d" -  
MT"nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/errno.o" -c -o  
"nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/errno.o" "../  
nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/errno.c"  
Invoking: GNU RISC-V Cross C Compiler  
riscv64-unknown-elf-gcc -march=rv32imafdc -mabi=ilp32d -  
mtune=nuclei-300-series -mcmodel=medlow -msave-restore -isystem=/  
include/newlib-nano -O2 -ffunction-sections -fdata-sections -fno-  
common -g -D__IDE_RV_CORE=n307fd -DBOOT_HARTID=0 -DRUNMODE_IC_EN=0  
-DRUNMODE_DC_EN=0 -DRUNMODE_CCM_EN=0 -  
DDOWNLOAD_MODE=DOWNLOAD_MODE_ILM -DDOWNLOAD_MODE_STRING=\"ILM\" -  
I"C:\NucleiStudio_workspace\test\nuclei_sdk\NMSIS\Core\Include" -
```

```
I"C:  
\NucleiStudio_workspace\test\nuclei_sdk\SoC\evalsoc\Common\Include"  
-I"C:  
\NucleiStudio_workspace\test\nuclei_sdk\SoC\evalsoc\Board\nuclei_fpga_eval\Inc  
-I"C:\NucleiStudio_workspace\test\application" -std=gnu11 -MMD -MP  
-MF"nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/execve.d" -  
MT"nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/execve.o" -c -  
o "nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/execve.o" "../  
nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/execve.c"  
Finished building: ../nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/  
newlib/environ.c  
Finished building: ../nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/  
newlib/chown.c  
Finished building: ../nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/  
newlib/clock_getres.c  
Finished building: ../nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/  
newlib/clock_settime.c  
Finished building: ../nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/  
newlib/close.c  
Finished building: ../nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/  
newlib/clock_gettime.c  
Building file: ../nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/  
newlib/exit.c  
Building file: ../nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/  
newlib/fork.c  
Building file: ../nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/  
newlib/fstat.c  
Invoking: GNU RISC-V Cross C Compiler  
Invoking: GNU RISC-V Cross C Compiler  
riscv64-unknown-elf-gcc -march=rv32imafdc -mabi=ilp32d -  
mtune=nuclei-300-series -mcmodel=medlow -msave-restore -isystem=/  
include/newlib-nano -O2 -ffunction-sections -fdata-sections -fno-  
common -g -D_IDE_RV_CORE=n307fd -DBOOT_HARTID=0 -DRUNMODE_IC_EN=0  
-DRUNMODE_DC_EN=0 -DRUNMODE_CCM_EN=0 -  
DDOWNLOAD_MODE=DOWNLOAD_MODE_ILM -DDOWNLOAD_MODE_STRING=\"ILM\" -  
I"C:\NucleiStudio_workspace\test\nuclei_sdk\NMSIS\Core\Include" -  
I"C:  
\NucleiStudio_workspace\test\nuclei_sdk\SoC\evalsoc\Common\Include"  
-I"C:  
\NucleiStudio_workspace\test\nuclei_sdk\SoC\evalsoc\Board\nuclei_fpga_eval\Inc  
-I"C:\NucleiStudio_workspace\test\application" -std=gnu11 -MMD -MP  
-MF"nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/exit.d" -  
MT"nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/exit.o" -c -o  
"nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/exit.o" "../  
nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/exit.c"  
Building file: ../nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/  
newlib/getpid.c
```

```
Invoking: GNU RISC-V Cross C Compiler
riscv64-unknown-elf-gcc -march=rv32imafdc -mabi=ilp32d -
mtune=nuclei-300-series -mcmodel=medlow -msave-restore -isystem=/
include/newlib-nano -O2 -ffunction-sections -fdata-sections -fno-
common -g -D__IDE_RV_CORE=n307fd -DBOOT_HARTID=0 -DRUNMODE_IC_EN=0
-DRUNMODE_DC_EN=0 -DRUNMODE_CCM_EN=0 -
DDOWNLOAD_MODE=DOWNLOAD_MODE_ILM -DDOWNLOAD_MODE_STRING=\"ILM\" -
I"C:\NucleiStudio_workspace\test\nuclei_sdk\NMSIS\Core\Include" -
I"C:
\NucleiStudio_workspace\test\nuclei_sdk\SoC\evalsoc\Common\Include"
-I"C:
\NucleiStudio_workspace\test\nuclei_sdk\SoC\evalsoc\Board\nuclei_fpga_eval\Inc
-I"C:\NucleiStudio_workspace\test\application" -std=gnu11 -MMD -MP
-MF"nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/fork.d" -
MT"nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/fork.o" -c -o
"nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/fork.o" "../
nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/fork.c"
Finished building: ../nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/
newlib/execve.c

riscv64-unknown-elf-gcc -march=rv32imafdc -mabi=ilp32d -
mtune=nuclei-300-series -mcmodel=medlow -msave-restore -isystem=/
include/newlib-nano -O2 -ffunction-sections -fdata-sections -fno-
common -g -D__IDE_RV_CORE=n307fd -DBOOT_HARTID=0 -DRUNMODE_IC_EN=0
-DRUNMODE_DC_EN=0 -DRUNMODE_CCM_EN=0 -
DDOWNLOAD_MODE=DOWNLOAD_MODE_ILM -DDOWNLOAD_MODE_STRING=\"ILM\" -
I"C:\NucleiStudio_workspace\test\nuclei_sdk\NMSIS\Core\Include" -
I"C:
\NucleiStudio_workspace\test\nuclei_sdk\SoC\evalsoc\Common\Include"
-I"C:
\NucleiStudio_workspace\test\nuclei_sdk\SoC\evalsoc\Board\nuclei_fpga_eval\Inc
-I"C:\NucleiStudio_workspace\test\application" -std=gnu11 -MMD -MP
-MF"nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/fstat.d" -
MT"nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/fstat.o" -c -o
"nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/fstat.o" "../
nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/fstat.c"
Invoking: GNU RISC-V Cross C Compiler

riscv64-unknown-elf-gcc -march=rv32imafdc -mabi=ilp32d -
mtune=nuclei-300-series -mcmodel=medlow -msave-restore -isystem=/
include/newlib-nano -O2 -ffunction-sections -fdata-sections -fno-
common -g -D__IDE_RV_CORE=n307fd -DBOOT_HARTID=0 -DRUNMODE_IC_EN=0
-DRUNMODE_DC_EN=0 -DRUNMODE_CCM_EN=0 -
DDOWNLOAD_MODE=DOWNLOAD_MODE_ILM -DDOWNLOAD_MODE_STRING=\"ILM\" -
I"C:\NucleiStudio_workspace\test\nuclei_sdk\NMSIS\Core\Include" -
I"C:
\NucleiStudio_workspace\test\nuclei_sdk\SoC\evalsoc\Common\Include"
-I"C:
```

```
\NucleiStudio_workspace\test\nuclei_sdk\SoC\evalsoc\Board\nuclei_fpga_eval\Inc
-I"C:\NucleiStudio_workspace\test\application" -std=gnu11 -MMD -MP
-MF"nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/getpid.d" -
-MT"nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/getpid.o" -c -
-o "nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/getpid.o" "../
nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/getpid.c"
Building file: ../nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/
newlib/gettimeofday.c
Building file: ../nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/
newlib/isatty.c
Invoking: GNU RISC-V Cross C Compiler
Building file: ../nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/
newlib/kill.c
riscv64-unknown-elf-gcc -march=rv32imafdc -mabi=ilp32d -
mtune=nuclei-300-series -mcmodel=medlow -msave-restore -isystem=/
include/newlib-nano -O2 -ffunction-sections -fdata-sections -fno-
common -g -D__IDE_RV_CORE=n307fd -DBOOT_HARTID=0 -DRUNMODE_IC_EN=0
-DRUNMODE_DC_EN=0 -DRUNMODE_CCM_EN=0 -
DDOWNLOAD_MODE=DOWNLOAD_MODE_ILM -DDOWNLOAD_MODE_STRING=\"ILM\" -
I"C:\NucleiStudio_workspace\test\nuclei_sdk\NMSIS\Core\Include" -
I"C:
\NucleiStudio_workspace\test\nuclei_sdk\SoC\evalsoc\Common\Include"
-I"C:
\NucleiStudio_workspace\test\nuclei_sdk\SoC\evalsoc\Board\nuclei_fpga_eval\Inc
-I"C:\NucleiStudio_workspace\test\application" -std=gnu11 -MMD -MP
-MF"nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/
gettimeofday.d" -MT"nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/
newlib/gettimeofday.o" -c -o "nuclei_sdk/SoC/evalsoc/Common/Source/
Stubs/newlib/gettimeofday.o" "../nuclei_sdk/SoC/evalsoc/Common/
Source/Stubs/newlib/gettimeofday.c"
Invoking: GNU RISC-V Cross C Compiler
riscv64-unknown-elf-gcc -march=rv32imafdc -mabi=ilp32d -
mtune=nuclei-300-series -mcmodel=medlow -msave-restore -isystem=/
include/newlib-nano -O2 -ffunction-sections -fdata-sections -fno-
common -g -D__IDE_RV_CORE=n307fd -DBOOT_HARTID=0 -DRUNMODE_IC_EN=0
-DRUNMODE_DC_EN=0 -DRUNMODE_CCM_EN=0 -
DDOWNLOAD_MODE=DOWNLOAD_MODE_ILM -DDOWNLOAD_MODE_STRING=\"ILM\" -
I"C:\NucleiStudio_workspace\test\nuclei_sdk\NMSIS\Core\Include" -
I"C:
\NucleiStudio_workspace\test\nuclei_sdk\SoC\evalsoc\Common\Include"
-I"C:
\NucleiStudio_workspace\test\nuclei_sdk\SoC\evalsoc\Board\nuclei_fpga_eval\Inc
-I"C:\NucleiStudio_workspace\test\application" -std=gnu11 -MMD -MP
-MF"nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/isatty.d" -
-MT"nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/isatty.o" -c -
-o "nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/isatty.o" "../
nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/isatty.c"
```

```
Invoking: GNU RISC-V Cross C Compiler
riscv64-unknown-elf-gcc -march=rv32imafdc -mabi=ilp32d -
mtune=nuclei-300-series -mcmodel=medlow -msave-restore -isystem=/
include/newlib-nano -O2 -ffunction-sections -fdata-sections -fno-
common -g -D__IDE_RV_CORE=n307fd -DBOOT_HARTID=0 -DRUNMODE_IC_EN=0
-DRUNMODE_DC_EN=0 -DRUNMODE_CCM_EN=0 -
DDOWNLOAD_MODE=DOWNLOAD_MODE_ILM -DDOWNLOAD_MODE_STRING=\"ILM\" -
I"C:\NucleiStudio_workspace\test\nuclei_sdk\NMSIS\Core\Include" -
I"C:
\NucleiStudio_workspace\test\nuclei_sdk\SoC\evalsoc\Common\Include"
-I"C:
\NucleiStudio_workspace\test\nuclei_sdk\SoC\evalsoc\Board\nuclei_fpga_eval\Inc
-I"C:\NucleiStudio_workspace\test\application" -std=gnu11 -MMD -MP
-MF"nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/kill.d" -
MT"nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/kill.o" -c -o
"nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/kill.o" "../
nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/kill.c"
Finished building: ../nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/
newlib/exit.c

Finished building: ../nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/
newlib/errno.c
Finished building: ../nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/
newlib/fork.c
Building file: ../nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/
newlib/link.c

Building file: ../nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/
newlib/lseek.c
Invoking: GNU RISC-V Cross C Compiler
Finished building: ../nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/
newlib/gettimeofday.c
riscv64-unknown-elf-gcc -march=rv32imafdc -mabi=ilp32d -
mtune=nuclei-300-series -mcmodel=medlow -msave-restore -isystem=/
include/newlib-nano -O2 -ffunction-sections -fdata-sections -fno-
common -g -D__IDE_RV_CORE=n307fd -DBOOT_HARTID=0 -DRUNMODE_IC_EN=0
-DRUNMODE_DC_EN=0 -DRUNMODE_CCM_EN=0 -
DDOWNLOAD_MODE=DOWNLOAD_MODE_ILM -DDOWNLOAD_MODE_STRING=\"ILM\" -
I"C:\NucleiStudio_workspace\test\nuclei_sdk\NMSIS\Core\Include" -
I"C:
\NucleiStudio_workspace\test\nuclei_sdk\SoC\evalsoc\Common\Include"
-I"C:
\NucleiStudio_workspace\test\nuclei_sdk\SoC\evalsoc\Board\nuclei_fpga_eval\Inc
-I"C:\NucleiStudio_workspace\test\application" -std=gnu11 -MMD -MP
-MF"nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/link.d" -
MT"nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/link.o" -c -o
```

```
"nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/link.o" "../  
nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/link.c"  
Building file: ../nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/  
newlib/open.c  
Invoking: GNU RISC-V Cross C Compiler  
riscv64-unknown-elf-gcc -march=rv32imafdc -mabi=ilp32d -  
mtune=nuclei-300-series -mcmodel=medlow -msave-restore -isystem=/  
include/newlib-nano -O2 -ffunction-sections -fdata-sections -fno-  
common -g -D__IDE_RV_CORE=n307fd -DBOOT_HARTID=0 -DRUNMODE_IC_EN=0  
-DRUNMODE_DC_EN=0 -DRUNMODE_CCM_EN=0 -  
DDOWNLOAD_MODE=DOWNLOAD_MODE_ILM -DDOWNLOAD_MODE_STRING=\"ILM\" -  
I"C:\NucleiStudio_workspace\test\nuclei_sdk\NMSIS\Core\Include" -  
I"C:  
\NucleiStudio_workspace\test\nuclei_sdk\SoC\evalsoc\Common\Include"  
-I"C:  
\NucleiStudio_workspace\test\nuclei_sdk\SoC\evalsoc\Board\nuclei_fpga_eval\Inc  
-I"C:\NucleiStudio_workspace\test\application" -std=gnu11 -MMD -MP  
-MF"nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/lseek.d" -  
MT"nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/lseek.o" -c -o  
"nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/lseek.o" "../  
nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/lseek.c"  
  
Invoking: GNU RISC-V Cross C Compiler  
riscv64-unknown-elf-gcc -march=rv32imafdc -mabi=ilp32d -  
mtune=nuclei-300-series -mcmodel=medlow -msave-restore -isystem=/  
include/newlib-nano -O2 -ffunction-sections -fdata-sections -fno-  
common -g -D__IDE_RV_CORE=n307fd -DBOOT_HARTID=0 -DRUNMODE_IC_EN=0  
-DRUNMODE_DC_EN=0 -DRUNMODE_CCM_EN=0 -  
DDOWNLOAD_MODE=DOWNLOAD_MODE_ILM -DDOWNLOAD_MODE_STRING=\"ILM\" -  
I"C:\NucleiStudio_workspace\test\nuclei_sdk\NMSIS\Core\Include" -  
I"C:  
\NucleiStudio_workspace\test\nuclei_sdk\SoC\evalsoc\Common\Include"  
-I"C:  
\NucleiStudio_workspace\test\nuclei_sdk\SoC\evalsoc\Board\nuclei_fpga_eval\Inc  
-I"C:\NucleiStudio_workspace\test\application" -std=gnu11 -MMD -MP  
-MF"nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/open.d" -  
MT"nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/open.o" -c -o  
"nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/open.o" "../  
nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/open.c"  
Building file: ../nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/  
newlib/read.c  
Finished building: ../nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/  
newlib/kill.c  
Finished building: ../nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/  
newlib/getpid.c  
Invoking: GNU RISC-V Cross C Compiler  
riscv64-unknown-elf-gcc -march=rv32imafdc -mabi=ilp32d -
```

```
mtune=nuclei-300-series -mcmode=medlow -msave-restore -isystem=/  
include/newlib-nano -O2 -ffunction-sections -fdata-sections -fno-  
common -g -D__IDE_RV_CORE=n307fd -DBOOT_HARTID=0 -DRUNMODE_IC_EN=0  
-DRUNMODE_DC_EN=0 -DRUNMODE_CCM_EN=0 -  
DDOWNLOAD_MODE=DOWNLOAD_MODE_ILM -DDOWNLOAD_MODE_STRING=\"ILM\" -  
I"C:\NucleiStudio_workspace\test\nuclei_sdk\NMSIS\Core\Include" -  
I"C:  
\NucleiStudio_workspace\test\nuclei_sdk\SoC\evalsoc\Common\Include"  
-I"C:  
\NucleiStudio_workspace\test\nuclei_sdk\SoC\evalsoc\Board\nuclei_fpga_eval\Inc  
-I"C:\NucleiStudio_workspace\test\application" -std=gnu11 -MMD -MP  
-MF"nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/read.d" -  
MT"nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/read.o" -c -o  
"nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/read.o" "../  
nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/read.c"
```

```
Building file: ../nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/  
newlib/readlink.c  
Building file: ../nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/  
newlib/sbrk.c  
Invoking: GNU RISC-V Cross C Compiler  
riscv64-unknown-elf-gcc -march=rv32imafdc -mabi=ilp32d -  
mtune=nuclei-300-series -mcmode=medlow -msave-restore -isystem=/  
include/newlib-nano -O2 -ffunction-sections -fdata-sections -fno-  
common -g -D__IDE_RV_CORE=n307fd -DBOOT_HARTID=0 -DRUNMODE_IC_EN=0  
-DRUNMODE_DC_EN=0 -DRUNMODE_CCM_EN=0 -  
DDOWNLOAD_MODE=DOWNLOAD_MODE_ILM -DDOWNLOAD_MODE_STRING=\"ILM\" -  
I"C:\NucleiStudio_workspace\test\nuclei_sdk\NMSIS\Core\Include" -  
I"C:  
\NucleiStudio_workspace\test\nuclei_sdk\SoC\evalsoc\Common\Include"  
-I"C:  
\NucleiStudio_workspace\test\nuclei_sdk\SoC\evalsoc\Board\nuclei_fpga_eval\Inc  
-I"C:\NucleiStudio_workspace\test\application" -std=gnu11 -MMD -MP  
-MF"nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/readlink.d" -  
MT"nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/readlink.o" -c  
-o "nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/readlink.o"  
"../nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/readlink.c"  
Invoking: GNU RISC-V Cross C Compiler  
riscv64-unknown-elf-gcc -march=rv32imafdc -mabi=ilp32d -  
mtune=nuclei-300-series -mcmode=medlow -msave-restore -isystem=/  
include/newlib-nano -O2 -ffunction-sections -fdata-sections -fno-  
common -g -D__IDE_RV_CORE=n307fd -DBOOT_HARTID=0 -DRUNMODE_IC_EN=0  
-DRUNMODE_DC_EN=0 -DRUNMODE_CCM_EN=0 -  
DDOWNLOAD_MODE=DOWNLOAD_MODE_ILM -DDOWNLOAD_MODE_STRING=\"ILM\" -  
I"C:\NucleiStudio_workspace\test\nuclei_sdk\NMSIS\Core\Include" -  
I"C:
```

```
\NucleiStudio_workspace\test\nuclei_sdk\SoC\evalsoc\Common\Include"
-I"C:
\NucleiStudio_workspace\test\nuclei_sdk\SoC\evalsoc\Board\nuclei_fpga_eval\Inc
-I"C:\NucleiStudio_workspace\test\application" -std=gnu11 -MMD -MP
-MF"nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/sbrk.d" -
MT"nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/sbrk.o" -c -o
"nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/sbrk.o" "../
nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/sbrk.c"
Finished building: ../nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/
newlib/isatty.c

Finished building: ../nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/
newlib/fstat.c
Finished building: ../nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/
newlib/link.c
Finished building: ../nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/
newlib/lseek.c
Finished building: ../nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/
newlib/read.c

Building file: ../nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/
newlib/stat.c

Building file: ../nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/
newlib/symlink.c
Building file: ../nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/
newlib/times.c
Invoking: GNU RISC-V Cross C Compiler
Building file: ../nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/
newlib/unlink.c
riscv64-unknown-elf-gcc -march=rv32imafdc -mabi=ilp32d -
mtune=nuclei-300-series -mcmodel=medlow -msave-restore -isystem=
/include/newlib-nano -O2 -ffunction-sections -fdata-sections -fno-
common -g -D__IDE_RV_CORE=n307fd -DBOOT_HARTID=0 -DRUNMODE_IC_EN=0
-DRUNMODE_DC_EN=0 -DRUNMODE_CCM_EN=0 -
DDOWNLOAD_MODE=DOWNLOAD_MODE_ILM -DDOWNLOAD_MODE_STRING=\"ILM\" -
-I"C:\NucleiStudio_workspace\test\nuclei_sdk\NMSIS\Core\Include" -
I"C:
\NucleiStudio_workspace\test\nuclei_sdk\SoC\evalsoc\Common\Include"
-I"C:
\NucleiStudio_workspace\test\nuclei_sdk\SoC\evalsoc\Board\nuclei_fpga_eval\Inc
-I"C:\NucleiStudio_workspace\test\application" -std=gnu11 -MMD -MP
-MF"nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/stat.d" -
MT"nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/stat.o" -c -o
"nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/stat.o" "../
```

```
nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/stat.c"
Finished building: ../nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/
newlib/readlink.c
Invoking: GNU RISC-V Cross C Compiler
Invoking: GNU RISC-V Cross C Compiler
riscv64-unknown-elf-gcc -march=rv32imafdc -mabi=ilp32d -
mtune=nuclei-300-series -mcmodel=medlow -msave-restore -isystem=/
include/newlib-nano -O2 -ffunction-sections -fdata-sections -fno-
common -g -D__IDE_RV_CORE=n307fd -DBOOT_HARTID=0 -DRUNMODE_IC_EN=0
-DRUNMODE_DC_EN=0 -DRUNMODE_CCM_EN=0 -
DDOWNLOAD_MODE=DOWNLOAD_MODE_ILM -DDOWNLOAD_MODE_STRING=\"ILM\"
-I"C:\NucleiStudio_workspace\test\nuclei_sdk\NMSIS\Core\Include" -
I"C:
\NucleiStudio_workspace\test\nuclei_sdk\SoC\evalsoc\Common\Include"
-I"C:
\NucleiStudio_workspace\test\nuclei_sdk\SoC\evalsoc\Board\nuclei_fpga_eval\Inc
-I"C:\NucleiStudio_workspace\test\application" -std=gnu11 -MMD -MP
-MF"nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/symlink.d" -
MT"nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/symlink.o" -c
-o "nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/symlink.o"
"../nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/symlink.c"
Invoking: GNU RISC-V Cross C Compiler
riscv64-unknown-elf-gcc -march=rv32imafdc -mabi=ilp32d -
mtune=nuclei-300-series -mcmodel=medlow -msave-restore -isystem=/
include/newlib-nano -O2 -ffunction-sections -fdata-sections -fno-
common -g -D__IDE_RV_CORE=n307fd -DBOOT_HARTID=0 -DRUNMODE_IC_EN=0
-DRUNMODE_DC_EN=0 -DRUNMODE_CCM_EN=0 -
DDOWNLOAD_MODE=DOWNLOAD_MODE_ILM -DDOWNLOAD_MODE_STRING=\"ILM\"
-I"C:\NucleiStudio_workspace\test\nuclei_sdk\NMSIS\Core\Include" -
I"C:
\NucleiStudio_workspace\test\nuclei_sdk\SoC\evalsoc\Common\Include"
-I"C:
\NucleiStudio_workspace\test\nuclei_sdk\SoC\evalsoc\Board\nuclei_fpga_eval\Inc
-I"C:\NucleiStudio_workspace\test\application" -std=gnu11 -MMD -MP
-MF"nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/times.d" -
MT"nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/times.o" -c -o
"nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/times.o" "../
nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/times.c"
riscv64-unknown-elf-gcc -march=rv32imafdc -mabi=ilp32d -
mtune=nuclei-300-series -mcmodel=medlow -msave-restore -isystem=/
include/newlib-nano -O2 -ffunction-sections -fdata-sections -fno-
common -g -D__IDE_RV_CORE=n307fd -DBOOT_HARTID=0 -DRUNMODE_IC_EN=0
-DRUNMODE_DC_EN=0 -DRUNMODE_CCM_EN=0 -
DDOWNLOAD_MODE=DOWNLOAD_MODE_ILM -DDOWNLOAD_MODE_STRING=\"ILM\"
-I"C:\NucleiStudio_workspace\test\nuclei_sdk\NMSIS\Core\Include" -
I"C:
\NucleiStudio_workspace\test\nuclei_sdk\SoC\evalsoc\Common\Include"
```

```
-I"C:  
\NucleiStudio_workspace\test\nuclei_sdk\SoC\evalsoc\Board\nuclei_fpga_eval\Inc  
-I"C:\NucleiStudio_workspace\test\application" -std=gnu11 -MMD -MP  
-MF"nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/unlink.d" -  
MT"nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/unlink.o" -c -  
o "nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/unlink.o" "../  
nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/unlink.c"  
  
Building file: ../nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/  
newlib/wait.c  
Building file: ../nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/  
newlib/write.c  
Finished building: ../nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/  
newlib/sbrk.c  
Invoking: GNU RISC-V Cross C Compiler  
Invoking: GNU RISC-V Cross C Compiler  
  
riscv64-unknown-elf-gcc -march=rv32imafdc -mabi=ilp32d -  
mtune=nuclei-300-series -mcmodel=medlow -msave-restore -isystem=/  
include/newlib-nano -O2 -ffunction-sections -fdata-sections -fno-  
common -g -D__IDE_RV_CORE=n307fd -DBOOT_HARTID=0 -DRUNMODE_IC_EN=0  
-DRUNMODE_DC_EN=0 -DRUNMODE_CCM_EN=0 -  
DDOWNLOAD_MODE=DOWNLOAD_MODE_ILM -DDOWNLOAD_MODE_STRING=\"ILM\" -  
I"C:\NucleiStudio_workspace\test\nuclei_sdk\NMSIS\Core\Include" -  
I"C:  
\NucleiStudio_workspace\test\nuclei_sdk\SoC\evalsoc\Common\Include"  
-I"C:  
\NucleiStudio_workspace\test\nuclei_sdk\SoC\evalsoc\Board\nuclei_fpga_eval\Inc  
-I"C:\NucleiStudio_workspace\test\application" -std=gnu11 -MMD -MP  
-MF"nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/wait.d" -  
MT"nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/wait.o" -c -o  
"nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/wait.o" "../  
nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/wait.c"  
riscv64-unknown-elf-gcc -march=rv32imafdc -mabi=ilp32d -  
mtune=nuclei-300-series -mcmodel=medlow -msave-restore -isystem=/  
include/newlib-nano -O2 -ffunction-sections -fdata-sections -fno-  
common -g -D__IDE_RV_CORE=n307fd -DBOOT_HARTID=0 -DRUNMODE_IC_EN=0  
-DRUNMODE_DC_EN=0 -DRUNMODE_CCM_EN=0 -  
DDOWNLOAD_MODE=DOWNLOAD_MODE_ILM -DDOWNLOAD_MODE_STRING=\"ILM\" -  
I"C:\NucleiStudio_workspace\test\nuclei_sdk\NMSIS\Core\Include" -  
I"C:  
\NucleiStudio_workspace\test\nuclei_sdk\SoC\evalsoc\Common\Include"  
-I"C:  
\NucleiStudio_workspace\test\nuclei_sdk\SoC\evalsoc\Board\nuclei_fpga_eval\Inc  
-I"C:\NucleiStudio_workspace\test\application" -std=gnu11 -MMD -MP  
-MF"nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/write.d" -  
MT"nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/write.o" -c -o
```

```
"nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/write.o" "../
nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/write.c"
Building file: ../nuclei_sdk/SoC/evalsoc/Common/Source/GCC/
intexc_evalsoc.S
Invoking: GNU RISC-V Cross Assembler
riscv64-unknown-elf-gcc -march=rv32imafdc -mabi=ilp32d -
mtune=nuclei-300-series -mcmodel=medlow -msave-restore -isystem=
/include/newlib-nano -O2 -ffunction-sections -fdata-sections -fno-
common -g -x assembler-with-cpp -D__IDE_RV_CORE=n307fd -
DBOOT_HARTID=0 -DRUNMODE_IC_EN=0 -DRUNMODE_DC_EN=0 -
DRUNMODE_CCM_EN=0 -DDOWNLOAD_MODE=DOWNLOAD_MODE_ILM -
DDOWNLOAD_MODE_STRING=\"ILM\" -I"C:
\NucleiStudio_workspace\test\nuclei_sdk\NMSIS\Core\Include" -I"C:
\NucleiStudio_workspace\test\nuclei_sdk\SoC\evalsoc\Common\Include" -
-I"C:
\NucleiStudio_workspace\test\nuclei_sdk\SoC\evalsoc\Board\nuclei_fpga_eval\Inc
-I"C:\NucleiStudio_workspace\test\application" -MMD -MP -
MF"nuclei_sdk/SoC/evalsoc/Common/Source/GCC/intexc_evalsoc.d" -
MT"nuclei_sdk/SoC/evalsoc/Common/Source/GCC/intexc_evalsoc.o" -c -o
"nuclei_sdk/SoC/evalsoc/Common/Source/GCC/intexc_evalsoc.o" "../
nuclei_sdk/SoC/evalsoc/Common/Source/GCC/intexc_evalsoc.S"
Finished building: ../nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/
newlib/unlink.c

Finished building: ../nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/
newlib/symlink.c
Building file: ../nuclei_sdk/SoC/evalsoc/Common/Source/GCC/
intexc_evalsoc_s.S

Invoking: GNU RISC-V Cross Assembler
Building file: ../nuclei_sdk/SoC/evalsoc/Common/Source/GCC/
startup_evalsoc.S
Finished building: ../nuclei_sdk/SoC/evalsoc/Common/Source/GCC/
intexc_evalsoc.S
riscv64-unknown-elf-gcc -march=rv32imafdc -mabi=ilp32d -
mtune=nuclei-300-series -mcmodel=medlow -msave-restore -isystem=
/include/newlib-nano -O2 -ffunction-sections -fdata-sections -fno-
common -g -x assembler-with-cpp -D__IDE_RV_CORE=n307fd -
DBOOT_HARTID=0 -DRUNMODE_IC_EN=0 -DRUNMODE_DC_EN=0 -
DRUNMODE_CCM_EN=0 -DDOWNLOAD_MODE=DOWNLOAD_MODE_ILM -
DDOWNLOAD_MODE_STRING=\"ILM\" -I"C:
\NucleiStudio_workspace\test\nuclei_sdk\NMSIS\Core\Include" -I"C:
\NucleiStudio_workspace\test\nuclei_sdk\SoC\evalsoc\Common\Include" -
-I"C:
\NucleiStudio_workspace\test\nuclei_sdk\SoC\evalsoc\Board\nuclei_fpga_eval\Inc
-I"C:\NucleiStudio_workspace\test\application" -MMD -MP -
MF"nuclei_sdk/SoC/evalsoc/Common/Source/GCC/intexc_evalsoc_s.d" -
```

```
MT"nuclei_sdk/SoC/evalsoc/Common/Source/GCC/intexc_evalsoc_s.o" -c
-o "nuclei_sdk/SoC/evalsoc/Common/Source/GCC/intexc_evalsoc_s.o"
"../nuclei_sdk/SoC/evalsoc/Common/Source/GCC/intexc_evalsoc_s.S"
Finished building: ../nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/
newlib/open.c
Finished building: ../nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/
newlib/wait.c
Invoking: GNU RISC-V Cross Assembler
```

```
riscv64-unknown-elf-gcc -march=rv32imafdc -mabi=ilp32d -
mtune=nuclei-300-series -mcmodel=medlow -msave-restore -isystem=/
include/newlib-nano -O2 -ffunction-sections -fdata-sections -fno-
common -g -x assembler-with-cpp -D__IDE_RV_CORE=n307fd -
DBOOT_HARTID=0 -DRUNMODE_IC_EN=0 -DRUNMODE_DC_EN=0 -
DRUNMODE_CCM_EN=0 -DDOWNLOAD_MODE=DOWNLOAD_MODE_ILM -
DDOWNLOAD_MODE_STRING=\"ILM\" -I"C:
\NucleiStudio_workspace\test\nuclei_sdk\NMSIS\Core\Include" -I"C:
\NucleiStudio_workspace\test\nuclei_sdk\SoC\evalsoc\Common\Include" -
-I"C:
\NucleiStudio_workspace\test\nuclei_sdk\SoC\evalsoc\Board\nuclei_fpga_eval\Inc
-I"C:\NucleiStudio_workspace\test\application" -MMD -MP -
MF"nuclei_sdk/SoC/evalsoc/Common/Source/GCC/startup_evalsoc.d" -
MT"nuclei_sdk/SoC/evalsoc/Common/Source/GCC/startup_evalsoc.o" -c -
-o "nuclei_sdk/SoC/evalsoc/Common/Source/GCC/startup_evalsoc.o" "../
nuclei_sdk/SoC/evalsoc/Common/Source/GCC/startup_evalsoc.S"
Building file: ../nuclei_sdk/SoC/evalsoc/Common/Source/Drivers/
evalsoc_uart.c
Invoking: GNU RISC-V Cross C Compiler
Building file: ../nuclei_sdk/SoC/evalsoc/Common/Source/
evalsoc_common.c
Building file: ../nuclei_sdk/SoC/evalsoc/Common/Source/
system_evalsoc.c
riscv64-unknown-elf-gcc -march=rv32imafdc -mabi=ilp32d -
mtune=nuclei-300-series -mcmodel=medlow -msave-restore -isystem=/
include/newlib-nano -O2 -ffunction-sections -fdata-sections -fno-
common -g -D__IDE_RV_CORE=n307fd -DBOOT_HARTID=0 -DRUNMODE_IC_EN=0 -
-DRUNMODE_DC_EN=0 -DRUNMODE_CCM_EN=0 -
DDOWNLOAD_MODE=DOWNLOAD_MODE_ILM -DDOWNLOAD_MODE_STRING=\"ILM\" -
-I"C:\NucleiStudio_workspace\test\nuclei_sdk\NMSIS\Core\Include" -
-I"C:
\NucleiStudio_workspace\test\nuclei_sdk\SoC\evalsoc\Common\Include" -
-I"C:
\NucleiStudio_workspace\test\nuclei_sdk\SoC\evalsoc\Board\nuclei_fpga_eval\Inc
-I"C:\NucleiStudio_workspace\test\application" -std=gnu11 -MMD -MP -
-MF"nuclei_sdk/SoC/evalsoc/Common/Source/Drivers/evalsoc_uart.d" -
```

```

MT"nuclei_sdk/SoC/evalsoc/Common/Source/Drivers/evalsoc_uart.o" -c
-o "nuclei_sdk/SoC/evalsoc/Common/Source/Drivers/evalsoc_uart.o"
"../nuclei_sdk/SoC/evalsoc/Common/Source/Drivers/evalsoc_uart.c"
Invoking: GNU RISC-V Cross C Compiler
riscv64-unknown-elf-gcc -march=rv32imafdc -mabi=ilp32d -
mtune=nuclei-300-series -mcmodel=medlow -msave-restore -isystem=/
include/newlib-nano -O2 -ffunction-sections -fdata-sections -fno-
common -g -D__IDE_RV_CORE=n307fd -DBOOT_HARTID=0 -DRUNMODE_IC_EN=0
-DRUNMODE_DC_EN=0 -DRUNMODE_CCM_EN=0 -
DDOWNLOAD_MODE=DOWNLOAD_MODE_ILL -DDOWNLOAD_MODE_STRING=\"ILL\"
-I"C:\NucleiStudio_workspace\test\nuclei_sdk\NMSIS\Core\Include" -
I"C:
\NucleiStudio_workspace\test\nuclei_sdk\SoC\evalsoc\Common\Include"
-I"C:
\NucleiStudio_workspace\test\nuclei_sdk\SoC\evalsoc\Board\nuclei_fpga_eval\Inc
-I"C:\NucleiStudio_workspace\test\application" -std=gnu11 -MMD -MP
-MF"nuclei_sdk/SoC/evalsoc/Common/Source/evalsoc_common.d" -
MT"nuclei_sdk/SoC/evalsoc/Common/Source/evalsoc_common.o" -c -o
"nuclei_sdk/SoC/evalsoc/Common/Source/evalsoc_common.o" "../
nuclei_sdk/SoC/evalsoc/Common/Source/evalsoc_common.c"
Invoking: GNU RISC-V Cross C Compiler
riscv64-unknown-elf-gcc -march=rv32imafdc -mabi=ilp32d -
mtune=nuclei-300-series -mcmodel=medlow -msave-restore -isystem=/
include/newlib-nano -O2 -ffunction-sections -fdata-sections -fno-
common -g -D__IDE_RV_CORE=n307fd -DBOOT_HARTID=0 -DRUNMODE_IC_EN=0
-DRUNMODE_DC_EN=0 -DRUNMODE_CCM_EN=0 -
DDOWNLOAD_MODE=DOWNLOAD_MODE_ILL -DDOWNLOAD_MODE_STRING=\"ILL\"
-I"C:\NucleiStudio_workspace\test\nuclei_sdk\NMSIS\Core\Include" -
I"C:
\NucleiStudio_workspace\test\nuclei_sdk\SoC\evalsoc\Common\Include"
-I"C:
\NucleiStudio_workspace\test\nuclei_sdk\SoC\evalsoc\Board\nuclei_fpga_eval\Inc
-I"C:\NucleiStudio_workspace\test\application" -std=gnu11 -MMD -MP
-MF"nuclei_sdk/SoC/evalsoc/Common/Source/system_evalsoc.d" -
MT"nuclei_sdk/SoC/evalsoc/Common/Source/system_evalsoc.o" -c -o
"nuclei_sdk/SoC/evalsoc/Common/Source/system_evalsoc.o" "../
nuclei_sdk/SoC/evalsoc/Common/Source/system_evalsoc.c"
Finished building: ../nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/
newlib/stat.c
Finished building: ../nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/
newlib/write.c

```

```

Building file: ../application/main.c
Invoking: GNU RISC-V Cross C Compiler
riscv64-unknown-elf-gcc -march=rv32imafdc -mabi=ilp32d -
mtune=nuclei-300-series -mcmodel=medlow -msave-restore -isystem=/

```

```
include/newlib-nano -O2 -ffunction-sections -fdata-sections -fno-common -g -D _IDE_RV_CORE=n307fd -DBOOT_HARTID=0 -DRUNMODE_IC_EN=0 -DRUNMODE_DC_EN=0 -DRUNMODE_CCM_EN=0 -DDOWNLOAD_MODE=DOWNLOAD_MODE_ILM -DDOWNLOAD_MODE_STRING=\"ILM\" -I"C:\NucleiStudio_workspace\test\nuclei_sdk\NMSIS\Core\Include" -I"C:\NucleiStudio_workspace\test\nuclei_sdk\SoC\evalsoc\Common\Include" -I"C:\NucleiStudio_workspace\test\nuclei_sdk\SoC\evalsoc\Board\nuclei_fpga_eval\Include" -I"C:\NucleiStudio_workspace\test\application" -std=gnu11 -MMD -MP -MF"application/main.d" -MT"application/main.o" -c -o "application/main.o" "../application/main.c"
Finished building: ../nuclei_sdk/SoC/evalsoc/Common/Source/GCC/startup_evalsoc.S

Finished building: ../nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/times.c

Finished building: ../nuclei_sdk/SoC/evalsoc/Common/Source/GCC/intexc_evalsoc_s.S

Finished building: ../nuclei_sdk/SoC/evalsoc/Common/Source/Drivers/evalsoc_uart.c

Finished building: ../nuclei_sdk/SoC/evalsoc/Common/Source/evalsoc_common.c

Finished building: ../nuclei_sdk/SoC/evalsoc/Common/Source/system_evalsoc.c

Finished building: ../application/main.c

Building target: test.elf
Invoking: GNU RISC-V Cross C++ Linker
riscv64-unknown-elf-g++ -march=rv32imafdc -mabi=ilp32d -mtune=nuclei-300-series -mcmodel=medlow -msave-restore -isystem=/include/newlib-nano -O2 -ffunction-sections -fdata-sections -fno-common -g -T "C:
\NucleiStudio_workspace\test\nuclei_sdk\SoC\evalsoc\Board\nuclei_fpga_eval\Source\ -nostartfiles -nodefaultlibs -Xlinker --gc-sections -Wl,-Map,"test.map" -Wl,--check-sections -Wl,--no-warn-rwx-segments -u __isatty -u __write -u __sbrk -u __read -u __close -u __fstat -u __lseek -u __errno -o "test.elf" ./nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/chown.o ./nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/clock_getres.o ./nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/clock_gettime.o ./nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/clock_settime.o ./nuclei_sdk/SoC/evalsoc/Common/Source/
```

```

Stubs/newlib/close.o ./nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/
newlib/environ.o ./nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/
newlib/errno.o ./nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/
execve.o ./nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/
exit.o ./nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/
fork.o ./nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/
fstat.o ./nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/
getpid.o ./nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/
gettimeofday.o ./nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/
isatty.o ./nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/
kill.o ./nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/
link.o ./nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/
lseek.o ./nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/
open.o ./nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/
read.o ./nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/
readlink.o ./nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/
sbrk.o ./nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/
stat.o ./nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/
symlink.o ./nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/
times.o ./nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/
unlink.o ./nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/
wait.o ./nuclei_sdk/SoC/evalsoc/Common/Source/Stubs/newlib/
write.o ./nuclei_sdk/SoC/evalsoc/Common/Source/GCC/
intexc_evalsoc.o ./nuclei_sdk/SoC/evalsoc/Common/Source/GCC/
intexc_evalsoc_s.o ./nuclei_sdk/SoC/evalsoc/Common/Source/GCC/
startup_evalsoc.o ./nuclei_sdk/SoC/evalsoc/Common/Source/Drivers/
evalsoc_uart.o ./nuclei_sdk/SoC/evalsoc/Common/Source/
evalsoc_common.o ./nuclei_sdk/SoC/evalsoc/Common/Source/
system_evalsoc.o ./application/main.o -lstdc++ -lc_nano -lgcc
Finished building target: test.elf

```

```

Invoking: GNU RISC-V Cross Create Flash Image
riscv64-unknown-elf-objcopy -O ihex "test.elf" "test.hex"
Invoking: GNU RISC-V Cross Create Listing
riscv64-unknown-elf-objdump --source --all-headers --demangle --
line-numbers --wide "test.elf" > "test.lst"
Invoking: GNU RISC-V Cross Print Size
riscv64-unknown-elf-size --format=berkeley "test.elf"
    text      data      bss      dec      hex filename
  8824      1272     4592    14688      3960 test.elf
Finished building: test.siz
Finished building: test.hex

```

Finished building: test.lst

```
17:00:23 Build Finished. 0 errors, 0 warnings. (took 5s.75ms)
```

以下为org.eclipse.cdt.managedbuilder.core.headlessbuild所提供的参数，以供参考。

```
-data      {/path/to/workspace}
-remove    {[uri:/]/path/to/project}
-removeAll {[uri:/]/path/to/projectTreeURI} Remove all projects
under URI
  -import   {[uri:/]/path/to/project}
  -importAll {[uri:/]/path/to/projectTreeURI} Import all projects
under URI
  -build    {project_name_reg_ex{/config_reg_ex} | all}
  -cleanBuild {project_name_reg_ex{/config_reg_ex} | all}
  -markerType Marker types to fail build on {all | cdt |
marker_id}
  -no-indexer Disable indexer
  -verbose   Verbose progress monitor updates
  -printErrorMarkers Print all error markers
  -I        {include_path} additional include_path to add to
tools
  -include   {include_file} additional include_file to pass to
tools
  -D        {preproc_define} addition preprocessor defines to
pass to the tools
  -E        {var=value} replace/add value to environment
variable when running all tools
  -Ea       {var=value} append value to environment variable
when running all tools
  -Ep       {var=value} prepend value to environment variable
when running all tools
  -Er       {var} remove/unset the given environment variable
  -T        {toolid} {optionid=value} replace a tool option
value in each configuration built
  -Ta       {toolid} {optionid=value} append to a tool option
value in each configuration built
  -Tp       {toolid} {optionid=value} prepend to a tool option
value in each configuration built
  -Tr       {toolid} {optionid=value} remove a tool option value
in each configuration built
  Tool option values are parsed as a string, comma
separated list of strings or a boolean based on the options type
```

OpenOCD烧写程序时报错Error:Device ID 8xle2g8a6d is not known as FESPI capable¶

问题说明¶

Nuclei Studio 2023.10版中烧写程序时有报以下错误：

参见这个 <https://github.com/riscv-mcu/hbird-sdk/issues/8>

```
Info : Using libusb driver
Info : clock speed 1000 kHz
Info : JTAG tap: riscv.cpu tap/device found: 0x1e200a6d (mfg: 0x536 (Nuclei System Technology Co Ltd), part: 0xe200, ver: 0x1)
Info : [riscv.cpu] Found 0 triggers
halted at 0x200000b2 due to debug interrupt
Info : Examined RISCV core; XLEN=32, misa=0x40001105
[riscv.cpu] Target successfully examined.
Info : starting gdb server for riscv.cpu on 3333
Info : Listening on port 3333 for gdb connections
Error: Device ID 0x1e200a6d is not known as FESPI capable
Error: auto_probe failed
```

解决方案¶

因为在openocd 2023.10中，将flash bank \$_FLASHNAME从fespi修改为了nuspi，需要工程中的openocd配置文件中的fespi修改为了nuspi，以蜂鸟工程为例，将 hbird_sdk/SoC/hbirdv2/Board/mcu200t/openocd_hbirdv2.cfg修改为如下配置，工程即可正常使用。

```
adapter_khz      1000

interface ftdi
ftdi_vid_pid 0x0403 0x6010
ftdi_oscanc1_mode off

transport select jtag

ftdi_layout_init 0x0008 0x001b
ftdi_layout_signal nSRST -oe 0x0020 -data 0x0020
ftdi_layout_signal TCK -data 0x0001
ftdi_layout_signal TDI -data 0x0002
```

```
ftdi_layout_signal TDO -input 0x0004
ftdi_layout_signal TMS -data 0x0008
ftdi_layout_signal JTAG_SEL -data 0x0100 -oe 0x0100

set _CHIPNAME riscv
jtag newtap $_CHIPNAME cpu -irlen 5

set _TARGETNAME $_CHIPNAME.cpu
target create $_TARGETNAME riscv -chain-position $_TARGETNAME
$_TARGETNAME configure -work-area-phys 0x80000000 -work-area-size
10000 -work-area-backup 1

set _FLASHNAME $_CHIPNAME.flash
flash bank $_FLASHNAME nuspi 0x20000000 0 0 0 $_TARGETNAME
# Set the ILM space also as flash, to make sure it can be add
breakpoint with hardware trigger
#flash bank onboard_ilm nuspi 0x80000000 0 0 0 $_TARGETNAME

# Expose Nuclei self-defined CSRS range
770-800,835-850,1984-2032,2064-2070
# See https://github.com/riscv/riscv-gnu-toolchain/issues/
319#issuecomment-358397306
# Then user can view the csr register value in gdb using: info reg
csr775 for CSR MTVT(0x307)
riscv expose_csr 770-800,835-850,1984-2032,2064-2070

init
#reset
if {[info exists pulse_srst]} {
    ftdi_set_signal nSRST 0
    ftdi_set_signal nSRST z
}
halt
# We must turn on this because otherwise the IDE version debug
cannot download the program into flash
flash protect 0 0 last off
```

关于dhrystone在IDE上跑分和NSDK 0.5.0命令行跑分不一致的问题¶

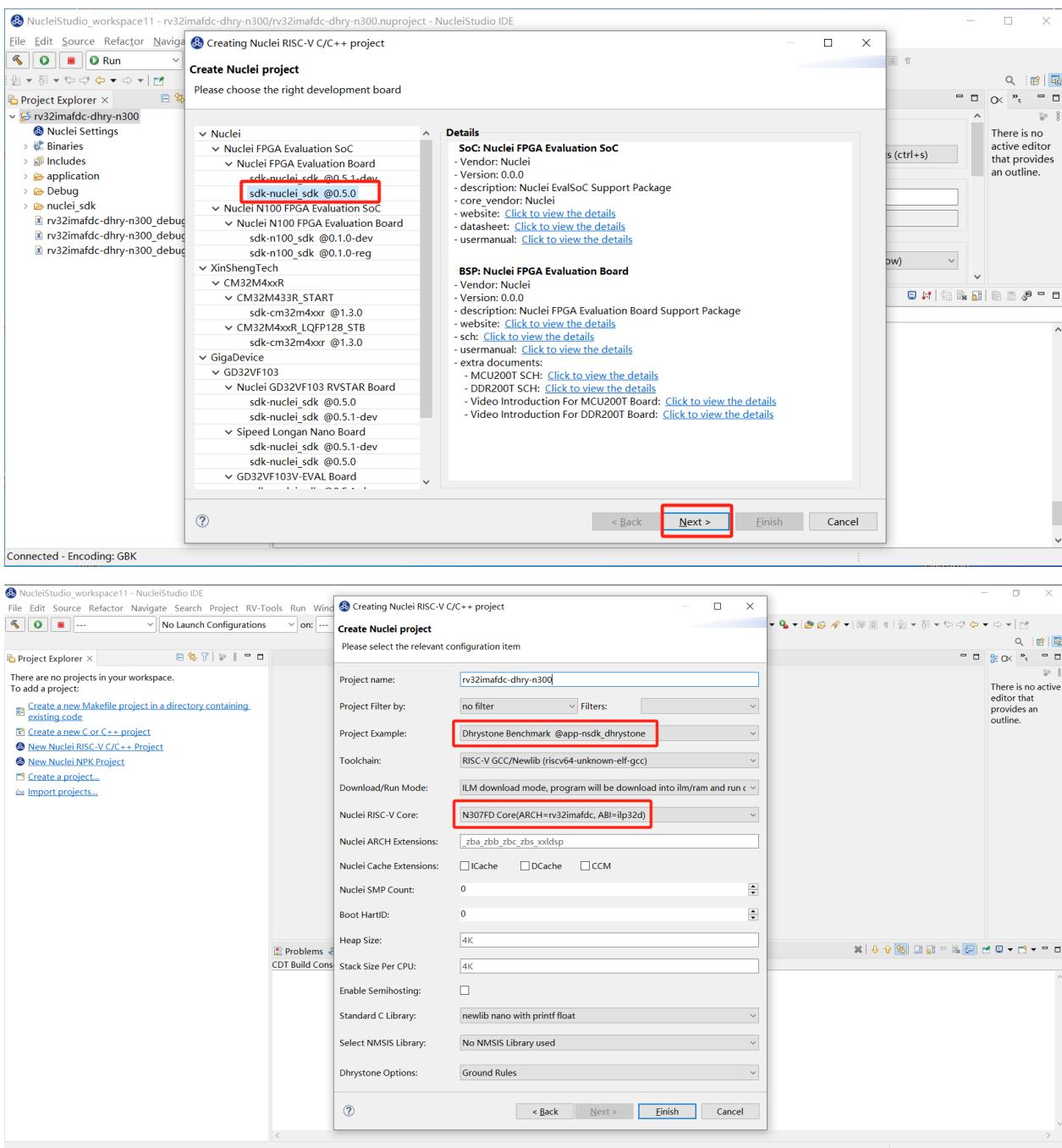
问题说明¶

在0.5.0版本的sdk-nuclei_sdk中，为了IDE上使用libncrt库的时候编译有些程序不报错，设置了会默认带上`-msave-restore`。但在创建dhrystone用例工程时，选择使用newlib库后，该选项会导致跑分降低，不符合CPU的真实跑分。

解决方案¶

在跑分的时候，需要在对应项目的Properties -> C/C++ Build -> Settings中，取消对Small prologue/epilogue(`-msave-restore`)的选中。具体流程和示例图如下：

1. 下载sdk-nuclei_sdk 0.5.0 NPK组件包。
2. 新建一个Nuclei RISCV-V C/C++ project。
3. 在新建项目的过程中，选中Dhrystone Benchmark和N307FD Core,其他选项默认设置即可。此时直接编译运行，跑分为1.405。
4. 但实际需要跑分时，要先取消选中`-msave-restore`选项，该跑分结果为1.664。



```

NucleiStudio_workspace11 - rv32imafdc-dhyr-n300/rv32imafdc-dhyr-n300.nuproject - NucleiStudio IDE
File Edit Source Refactor Navigate Search Project RV-Tools Run Window Help
Project Explorer X rv32imafdc-dhyr-n300/Nuclei Settings X
General
This section describes general information about this file.
project name: rv32imafdc-dhyr-n300 Configuration: Debug Save settings (ctrl+s)
Core Info
Core : N307FD Core(ARCH=rv32imafdc,ABI=ilp32c) Other extensions: zba_zbb_zbc_zbs_xxldsp
ARCH : rv32imafdc ABI : ilp32d
Tuning Info
Tuning : Nuclei 300 series (-mtune=nuclei-300-series) Code model : Medium Low (-mcmodel=medlow)
Download : ILM
Problems Tasks Console Properties Terminal
CDT Build Console [rv32imafdc-dhyr-n300]
BUILDING target: RV32IMAFDC-DHYR-N300.ELF
Invoking: GNU RISC-V Cross C++ Linker
riscv64-unknown-elf-g++ -march=rv32imafdc -mabi=ilp32d -mtune=nuclei-300-series -mcmodel=medlow -msave-restore isystem=/include/newl:
Finished building target: rv32imafdc-dhyr-n300.elf

Invoking: GNU RISC-V Cross Create Flash Image
riscv64-unknown-elf-objcopy -ihex "rv32imafdc-dhyr-n300.elf" "rv32imafdc-dhyr-n300.hex"
Invoking: GNU RISC-V Cross Create Listing
Invoking: GNU RISC-V Cross Print Size
riscv64-unknown-elf-objdump --source --all-headers --demangle --line-numbers --wide "rv32imafdc-dhyr-n300.elf" > "rv32imafdc-dhyr-n300.siz"
riscv64-unknown-elf-size --format=berkeley "rv32imafdc-dhyr-n300.elf"
text data bss dec hex filename
19968 3904 14828 38700 972c rv32imafdc-dhyr-n300.elf
Finished building: rv32imafdc-dhyr-n300.hex
Finished building: rv32imafdc-dhyr-n300.siz

rv32imafdc-dhyr-n300

```



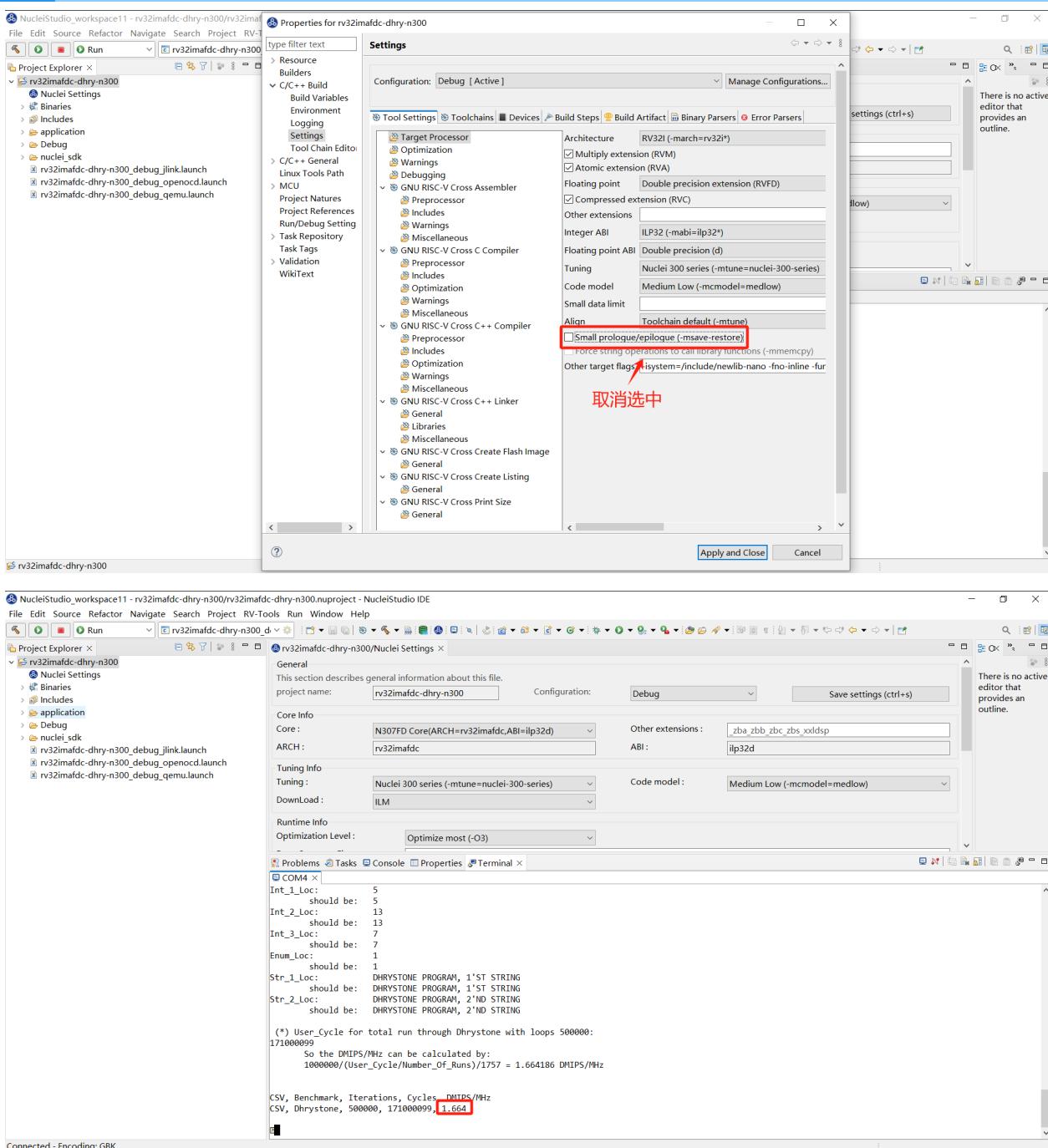
```

NucleiStudio_workspace11 - rv32imafdc-dhyr-n300/rv32imafdc-dhyr-n300.nuproject - NucleiStudio IDE
File Edit Source Refactor Navigate Search Project RV-Tools Run Window Help
Project Explorer X rv32imafdc-dhyr-n300/Nuclei Settings X
General
This section describes general information about this file.
project name: rv32imafdc-dhyr-n300 Configuration: Debug Save settings (ctrl+s)
Core Info
Core : N307FD Core(ARCH=rv32imafdc,ABI=ilp32d) Other extensions: zba_zbb_zbc_zbs_xxldsp
ARCH : rv32imafdc ABI : ilp32d
Tuning Info
Tuning : Nuclei 300 series (-mtune=nuclei-300-series) Code model : Medium Low (-mcmodel=medlow)
Download : ILM
Runtime Info
Optimization Level: Optimize most (-O3)
COM4 X
Int_1_Loc: 5
    should be: 5
Int_2_Loc: 13
    should be: 13
Int_3_Loc: 7
    should be: 7
Enum_Loc: 1
    should be: 1
Str_1_Loc: DHRYSTONE PROGRAM, 1'ST STRING
    should be: DHRYSTONE PROGRAM, 1'ST STRING
Str_2_Loc: DHRYSTONE PROGRAM, 2'ND STRING
    should be: DHRYSTONE PROGRAM, 2'ND STRING

(*) User_Cycle for total run through Dhrystone with loops 500000:
202500067
So the DMIPS/MHz can be calculated by:
1000000/(User_Cycle/Number_Of_Runs)/1757 = 1.405313 DMIPS/MHz

CSV, Benchmark, Iterations, Cycles, DMIPS/MHz
CSV, Dhrystone, 500000, 202500067, 1.405
Connected - Encoding: GBK

```



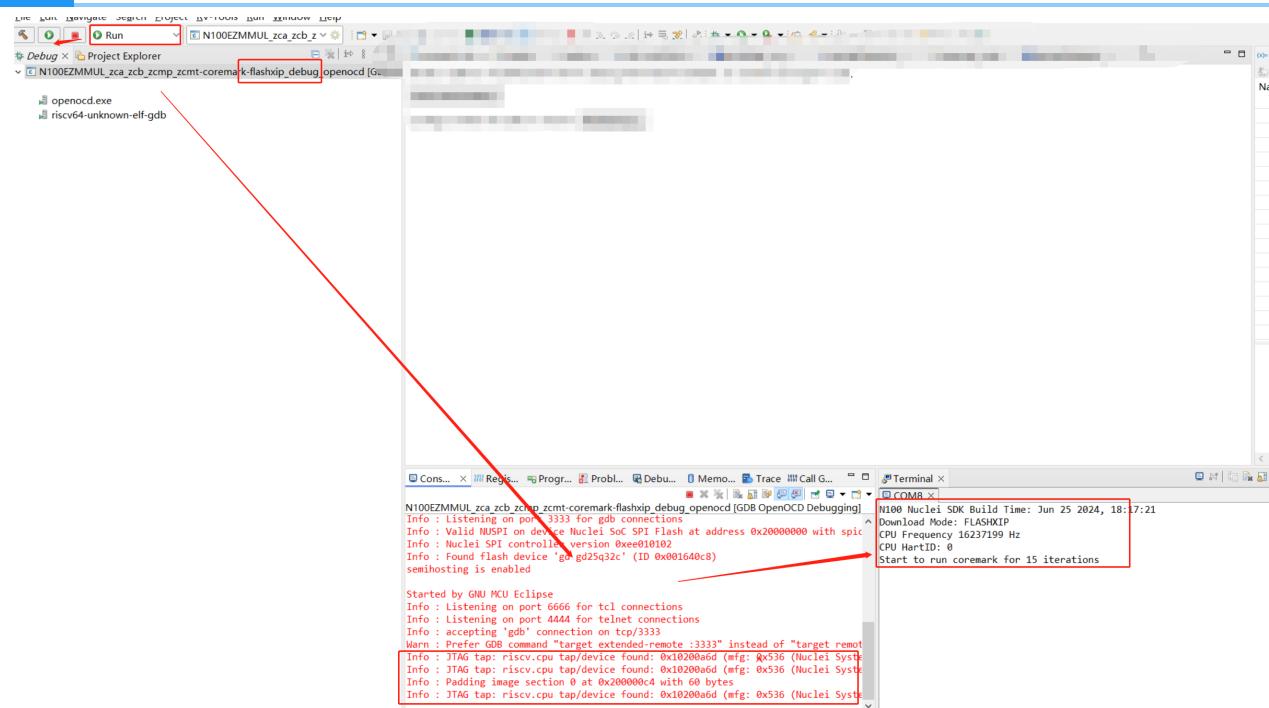
Error: Couldn't find an available hardware trigger / Error: can't add breakpoint: resource not available¶

问题说明¶

在NucleiStudio中使用OpenOCD调试hbird/hbirdv2处理器（不支持硬件断点）或者 Nuclei 100 系列的处理器时，当程序运行在Flash/FlashXip下时，会报Error。

```
Error: Couldn't find an available hardware trigger.  
Error: can't add breakpoint: resource not available
```

```
Info : coreid=0, 10 : 0x0  
Info : coreid=0, 32 : 0x0  
Info : Examined RISC-V core; found 1 harts  
Info : hart 0: XLEN=32, misa=0x0001104  
[riscv.cpu] Target successfully examined.  
Info : starting gdb server for riscv.cpu on 3333  
Info : Listening on port 3333 for gdb connections  
Info : Valid NUSPI on device Nuclei SoC SPI Flash at address 0x20000000 with spictrl regbase at 0x10014000  
Info : Nuclei SPI controller version 0xee010102  
Info : Found flash device 'gd gd25q32c' (ID 0x001640c8)  
semihosting is enabled  
  
Started by GNU MCU Eclipse  
Info : Listening on port 6666 for tcl connections  
Info : Listening on port 4444 for telnet connections  
Info : accepting 'gdb' connection on tcp/3333  
Warn : Prefer GDB command "target extended-remote :3333" instead of "target remote :3333"  
Info : JTAG tap: riscv.cpu tap/device found: 0x10200a6d (mfg: 0x536 (Nuclei System Technology Co Ltd), part: 0x0200, ver: 0x1)  
Info : JTAG tap: riscv.cpu tap/device found: 0x10200a6d (mfg: 0x536 (Nuclei System Technology Co Ltd), part: 0x0200, ver: 0x1)  
Info : Padding image section 0 at 0x200000c4 with 60 bytes  
Info : JTAG tap: riscv.cpu tap/device found: 0x10200a6d (mfg: 0x536 (Nuclei System Technology Co Ltd), part: 0x0200, ver: 0x1)  
Info : [riscv.cpu] Found 0 triggers  
Error: Couldn't find an available hardware trigger.  
Error: can't add breakpoint: resource not available
```



是因为所运行的CPU不支持硬件断点，导致程序运行在Flash上的时候，IDE调试功能无法正常工作，这个是IDE会需要打一个临时断点的缘故导致的。如果需要下载并运行程序，切换到Run运行模式可以正常运行程序。

解决方案¶

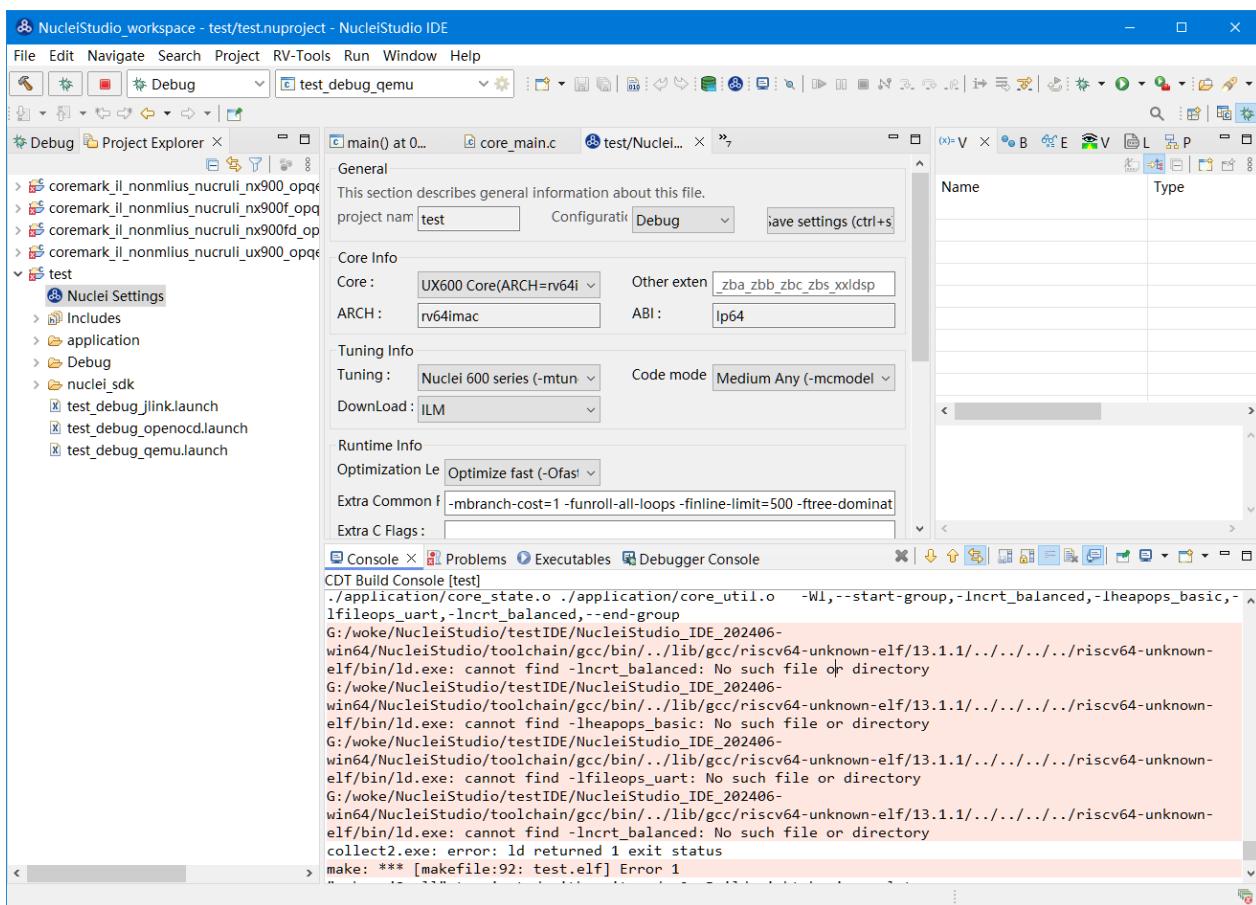
当在调试此类型处理器时，如果需要调试的话，就需要将程序编译运行在RAM上。

cannot find -lncrt_balanced: No such file or directory¶

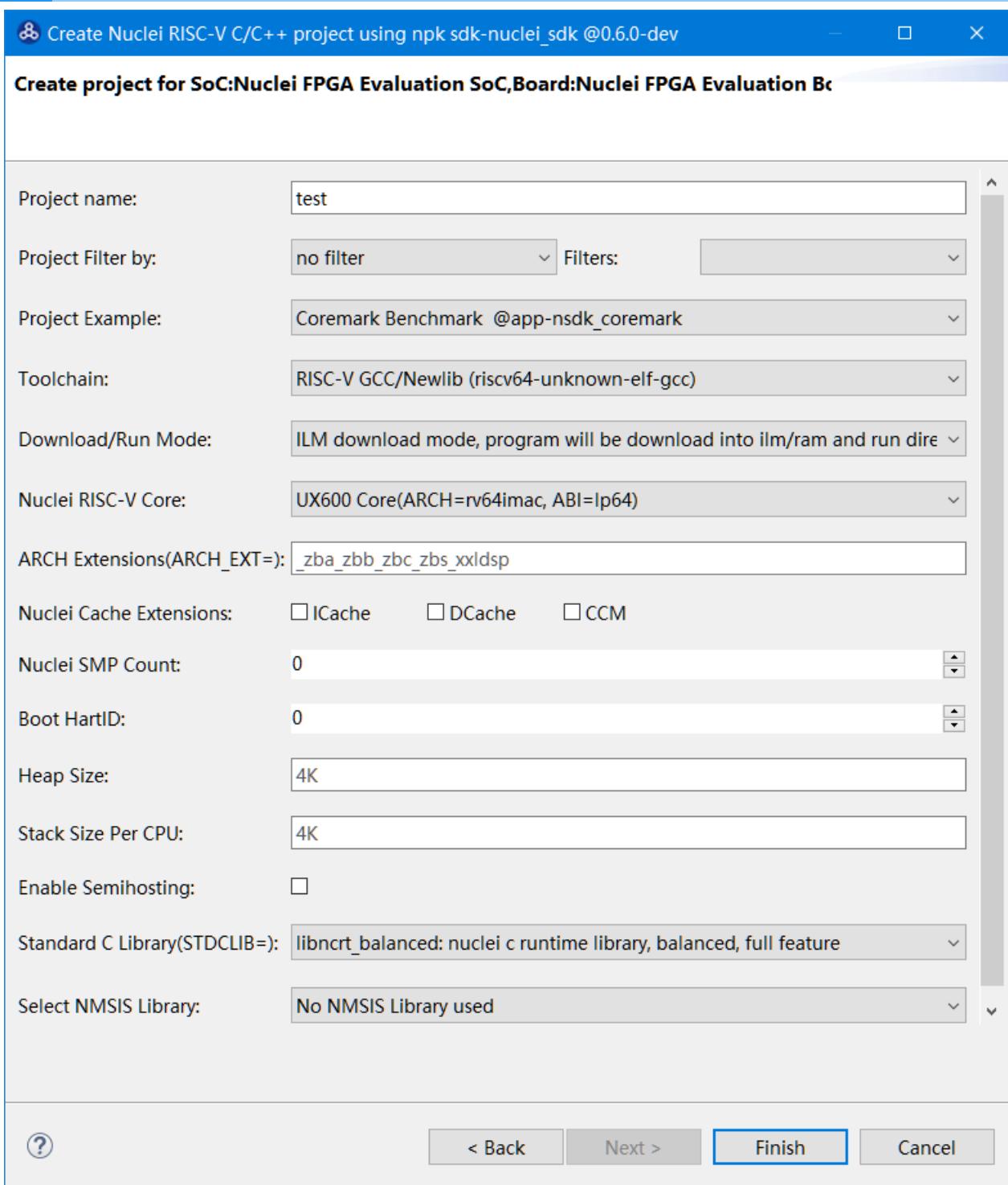
问题说明¶

在NucleiStudio中使用编译工程时有报错信息如下：

```
G:/NucleiStudio/toolchain/gcc/bin/..../lib/gcc/riscv64-unknown-elf/
13.1.1/..../.../.../riscv64-unknown-elf/bin/ld.exe: cannot find -
lncrt_balanced: No such file or directory
G:/NucleiStudio/toolchain/gcc/bin/..../lib/gcc/riscv64-unknown-elf/
13.1.1/..../.../.../riscv64-unknown-elf/bin/ld.exe: cannot find -
lheapops_basic: No such file or directory
G:/NucleiStudio/toolchain/gcc/bin/..../lib/gcc/riscv64-unknown-elf/
13.1.1/..../.../.../riscv64-unknown-elf/bin/ld.exe: cannot find -
lfileops_uart: No such file or directory
G:/NucleiStudio/toolchain/gcc/bin/..../lib/gcc/riscv64-unknown-elf/
13.1.1/..../.../.../riscv64-unknown-elf/bin/ld.exe: cannot find -
lncrt_balanced: No such file or directory
```



是因为在创建工程时，我们创建了一个64位的工程，同时在Standard C Library时，选择了带-lncrt_balanced、-lfileops_uart的扩展，而此类扩展又不支持64位，导致编译不通过。



解决方案¶

-lncrt_balanced、-lfileops_uart不支持64位处理器，在创建此类处理器工程时，避免使用libncrt库。

UnsatisfiedLinkError of swt-win32-4965r8.dll on Windows 7¶

问题说明¶

用户在Windows 7、Windows 8下使用NucleiStudio 2024.06时，发现启动不了，在NucleiStudio\configuration目录的日志中可以看到以下报错内容：

```
!ENTRY org.eclipse.osgi 4 0 2024-07-16 10:41:57.010
!MESSAGE Application error
!STACK 1
java.lang.UnsatisfiedLinkError: Could not load SWT library.
Reasons:
        C:\NucleiStudio\configuration\org.eclipse.osgi\492\0\.cp\swt-
win32-4965r11.dll: 找不到指定的程序。
        no swt-win32 in java.library.path: C:\NucleiStudio;C:
\Windows\Sun\Java\bin;C:\Windows\system32;C:\Windows;C:/
NucleiStudio//plugins/
org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_21.
0.3.v20240426-1530/jre/bin/server;C:/NucleiStudio//plugins/
org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_21.
0.3.v20240426-1530/jre/bin;C:\Java\JCDK3.0.4_ClassicEdition\bin;C:
\Java\jdk1.6.0_26\bin;C:\Java\jdk1.6.0_26\lib;C:
\Windows\system32;C:\Windows;C:\Windows\System32\Wbem;C:
\Windows\System32\WindowsPowerShell\v1.0\;C:\Program
Files\TortoiseSVN\bin;C:\Program Files (x86)\Microsoft SQL
Server\90\Tools\binn\;D:\Python25;C:\NucleiStudio;;
        no swt in java.library.path: C:\NucleiStudio;C:
\Windows\Sun\Java\bin;C:\Windows\system32;C:\Windows;C:/
NucleiStudio//plugins/
org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_21.
0.3.v20240426-1530/jre/bin/server;C:/NucleiStudio//plugins/
org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_21.
0.3.v20240426-1530/jre/bin;C:\Java\JCDK3.0.4_ClassicEdition\bin;C:
\Java\jdk1.6.0_26\bin;C:\Java\jdk1.6.0_26\lib;C:
\Windows\system32;C:\Windows;C:\Windows\System32\Wbem;C:
\Windows\System32\WindowsPowerShell\v1.0\;C:\Program
Files\TortoiseSVN\bin;C:\Program Files (x86)\Microsoft SQL
Server\90\Tools\binn\;D:\Python25;C:\NucleiStudio;;
        C:\Users\username\.swt\lib\win32\x86_64\swt-win32-4965r11.dll:
找不到指定的程序。
Can't load library: C:
```

```
\Users\username\.swt\lib\win32\x86_64\swt-win32.dll
  Can't load library: C:
\Users\username\.swt\lib\win32\x86_64\swt.dll
  C:\Users\username\.swt\lib\win32\x86_64\swt-win32-4965r11.dll:
找不到指定的程序。

  at org.eclipse.swt.internal.Library.loadLibrary(Library.java:
345)
  at org.eclipse.swt.internal.Library.loadLibrary(Library.java:
254)
  at org.eclipse.swt.internal.C.<clinit>(C.java:19)
  at
org.eclipse.swt.internal.win32.STARTUPINFO.<clinit>(STARTUPINFO.java:
42)
  at org.eclipse.swt.widgets.Display.<clinit>(Display.java:149)
  at
org.eclipse.ui.internal.Workbench.createDisplay(Workbench.java:721)
  at org.eclipse.ui.PlatformUI.createDisplay(PlatformUI.java:185)
  at
org.eclipse.ui.internal.ide.application.IDEApplication.createDisplay(IDEApplication.java:
182)
  at
org.eclipse.ui.internal.ide.application.IDEApplication.start(IDEApplication.java:
125)
  at
  at
org.eclipse.equinox.internal.app.EclipseAppHandle.run(EclipseAppHandle.java:
208)
  at
  at
org.eclipse.core.runtime.internal.adaptor.EclipseAppLauncher.runApplication(EclipseAppLauncher.java:
143)
  at
  at
org.eclipse.core.runtime.internal.adaptor.EclipseAppLauncher.start(EclipseAppLauncher.java:
109)
  at
  at
org.eclipse.core.runtime.adaptor.EclipseStarter.run(EclipseStarter.java:
439)
  at
  at
org.eclipse.core.runtime.adaptor.EclipseStarter.run(EclipseStarter.java:
271)
  at java.base/
jdk.internal.reflect.DirectMethodHandleAccessor.invoke(DirectMethodHandleAccessor.java:
103)
  at java.base/java.lang.reflect.Method.invoke(Method.java:580)
  at org.eclipse.equinox.launcher.Main.invokeFramework(Main.java:
668)
  at org.eclipse.equinox.launcher.Main.basicRun(Main.java:605)
  at org.eclipse.equinox.launcher.Main.run(Main.java:1481)
```

是因为在eclipse 2024.06版本中，有使用到一些特性，而该特性对操作系统有要求，可以参考<https://github.com/eclipse-platform/eclipse.platform.swt/issues/1252>

That commit references `SystemParametersInfoForDpi`. See <https://learn.microsoft.com/en-us/windows/win32/api/winuser/nf-winuser-systemparametersinfofordpi>

And `f809372` references `GetSystemMetricsForDpi`. See <https://learn.microsoft.com/en-us/windows/win32/api/winuser/nf-winuser-getsystemmetricsfordpi>

Both of these functions require a minimum version of Windows 10:

Minimum supported client	Windows 10, version 1607 [desktop apps only]
Minimum supported server	Windows Server 2016 [desktop apps only]



But this does still not explain problems with MacOS.

并且在eclipse的官方文档中，针对eclipse测试的操作系统中也做了说明，对某些版本的操作系统不再做兼容。可以参考

https://eclipse.dev/eclipse/development/plans/eclipse_project_plan_4_32.xml#target_environments

Operating System	Version	Hardware	JRE	Windowing System
Windows	10 11	x86 64-bit	OpenJDK 17.0.8 (LTS) OpenJDK 21 (LTS) Oracle Java 17.0.8 (LTS) Oracle Java 21 (LTS)	Win32
Red Hat Enterprise Linux	9.0	x86 64-bit aarch64	OpenJDK 17.0.8 (LTS) OpenJDK 21 (LTS) Oracle Java 17.0.8 (LTS) Oracle Java 21 (LTS)	GTK 3
		Power 64-bit LE	OpenJDK 17.0.8 (LTS)	
SUSE Linux Enterprise Server	15 SP4	x86 64-bit	OpenJDK 17.0.8 (LTS) OpenJDK 21 (LTS)	GTK 3
		Power 64-bit LE	OpenJDK 17.0.8 (LTS)	
Ubuntu Long Term Support	22.04	x86 64-bit aarch64	OpenJDK 17.0.8 (LTS) OpenJDK 21 (LTS)	GTK 3
Apple macOS	12	x86 64-bit	OpenJDK 17.0.8 (LTS) OpenJDK 21 (LTS) Oracle Java 17.0.8 (LTS) Oracle Java 21 (LTS)	Cocoa
	13	M1 (arm64)	OpenJDK 17.0.8 (LTS) OpenJDK 21 (LTS)	

而NucleiStudio 2024.06是基于eclipse 2024.06，所以也会有同类型的问题。

解决方案

请在windows 10或以上的版本操作系统上使用 NucleiStudio 2024.06。

如果想在Windows 7、Windows 8等低版本的操作系统上使用NucleiStudio，可以考虑使用NucleiStudio 2024.02及以下版本。

使用 Profiling 功能时可能遇到的一些问题¶

目前使用 Profiling 功能可能遇到一些问题，记录如下：

- 问题1：日志打印中报片上内存不足，没有充足内存来存放 gprof/gcov 数据
 - 问题2：采用串口输出的方式收集数据，打印被冲掉，Console 或 Terminal 收集的数据不全，导致数据解析失败，弹出 No files have been generated 错误弹框
 - 问题3：删掉 gmon.out 文件，再次解析时，弹出 No files have been generated 错误弹框

问题1：日志打印中报片上内存不足，没有充足内存来存放 gprof/gcov 数据

gprof/gcov data 需要存到片上内存上，占用内存的大小与用例规模有关(几十到几百KB不等)，需要确保片上内存足够大。

解决方案

首先需要确认软件配置的内存大小与硬件实际大小相匹配（ilm/sram/flash/ddr），否则需要适配软件链接脚本内存布局：

比如，如果是 DOWNLOAD=ilm 模式下载，可以按硬件的 ilm 与 dlm 大小适配。对于 nuclei sdk 0.6.0 版本，修改的文件为nuclei-sdk/SoC/evalsoc/Board/nuclei_fpga_eval/Source/GCC/gcc evalsoc ilm.ld

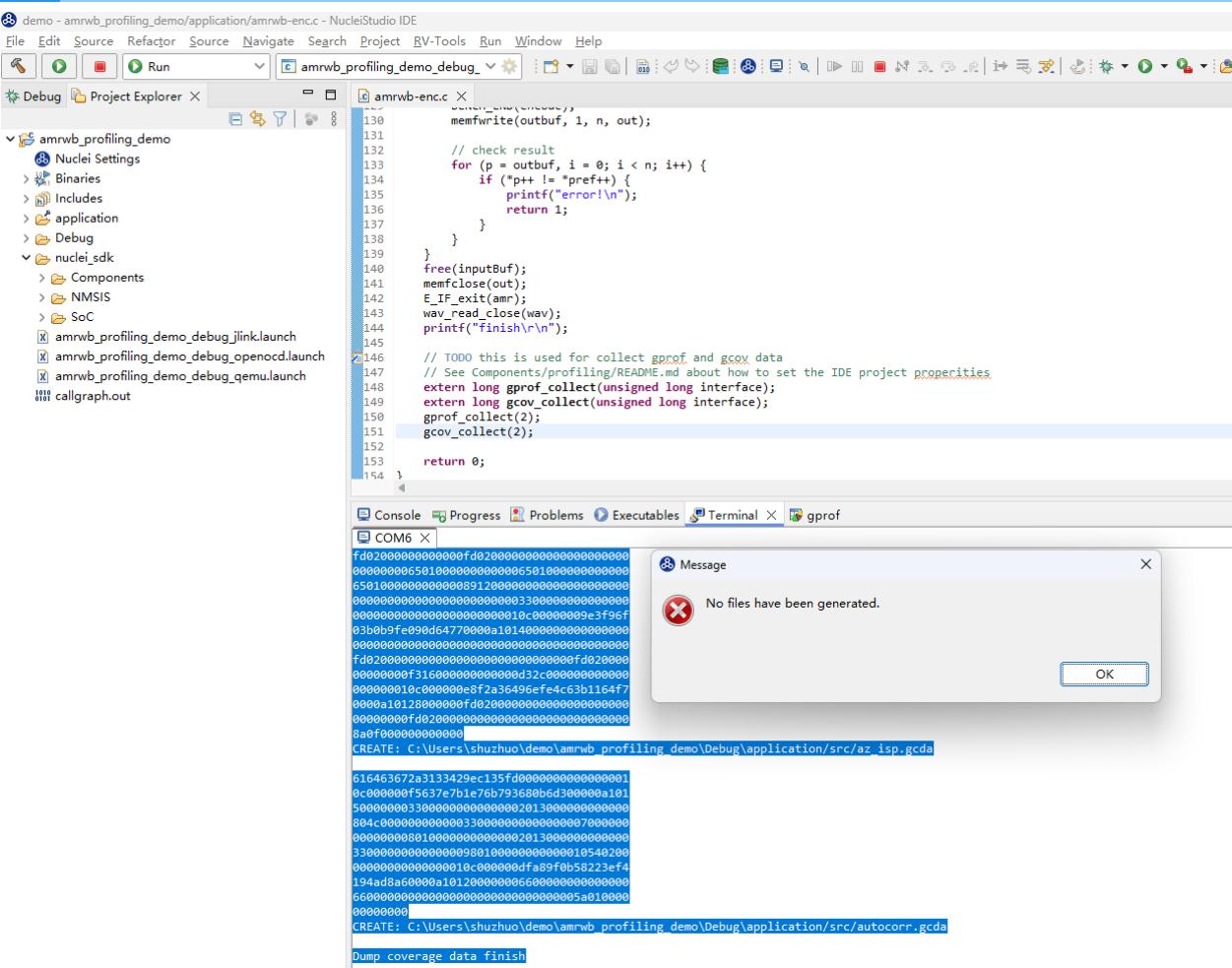
```
INCLUDE evalsoc.memory

MEMORY
{
    ilm (rxa!w) : ORIGIN = ILM_MEMORY_BASE, LENGTH =
ILM_MEMORY_SIZE
    ram (wxa!r) : ORIGIN = DLM_MEMORY_BASE, LENGTH =
DLM_MEMORY_SIZE
}
```

如果 DOWNLOAD=ilm 模式内存不足，可以使用内存大一点的下载方式（如 DOWNLOAD=ddr）。

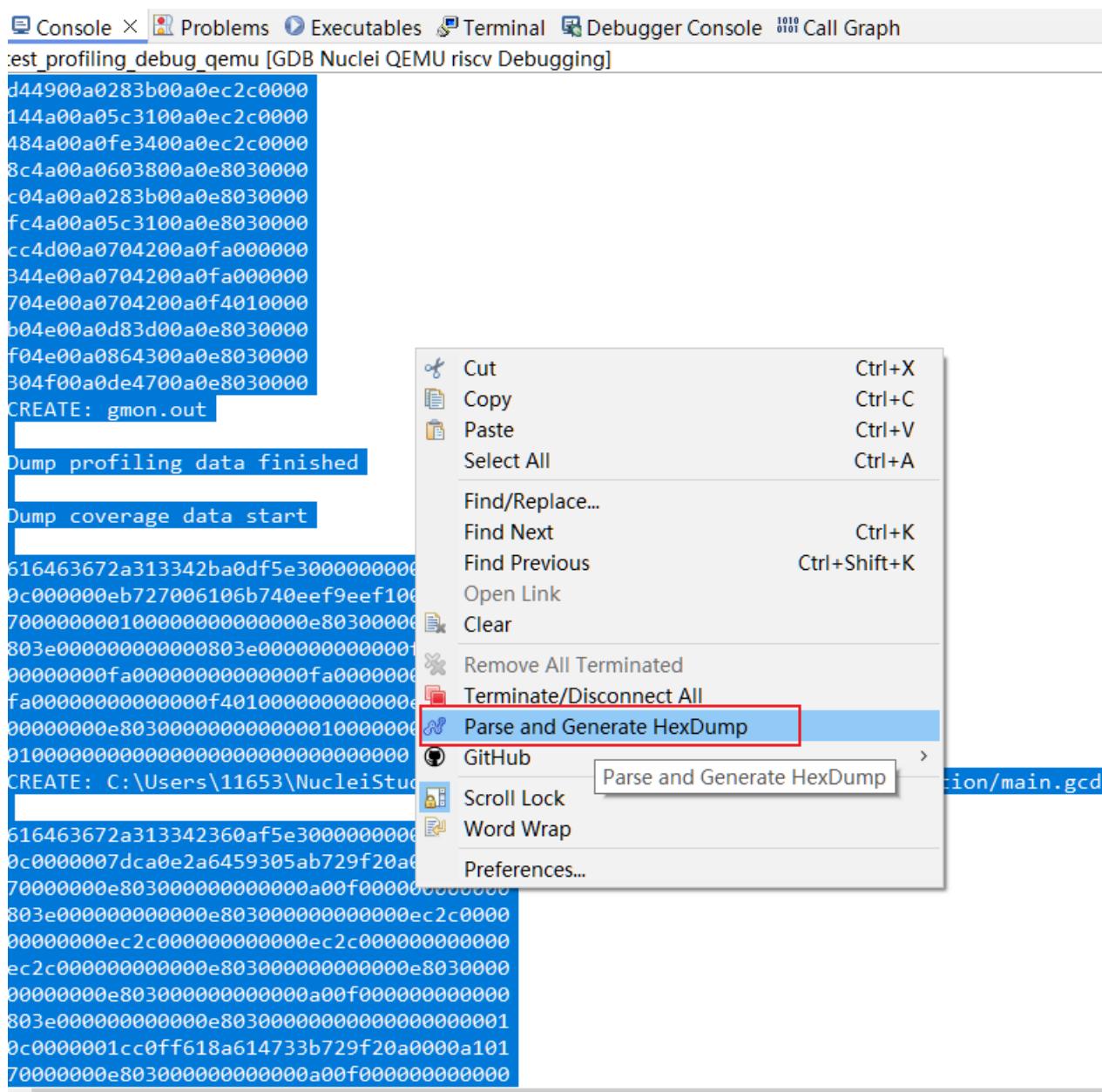
问题2：Console 或 Terminal 收集的数据不全导致数据解析时失败¶

在 NucleiStudio 2024.06 中，当选择使用串口输出的方式使用 Profiling 功能时，可能使用 Parse and Generate Hexdump 解析数据时弹出 No files have been generated 错误弹框，最后没有生成对应的 gmon.out 文件或者 *.gcno 文件。这可能是因为串口数据被冲掉，导致数据不完整从而解析失败



确认方法：

需确保串口开始时的打印没有被冲掉，参考Nuclei Studio使用Profiling功能进行性能调优举例



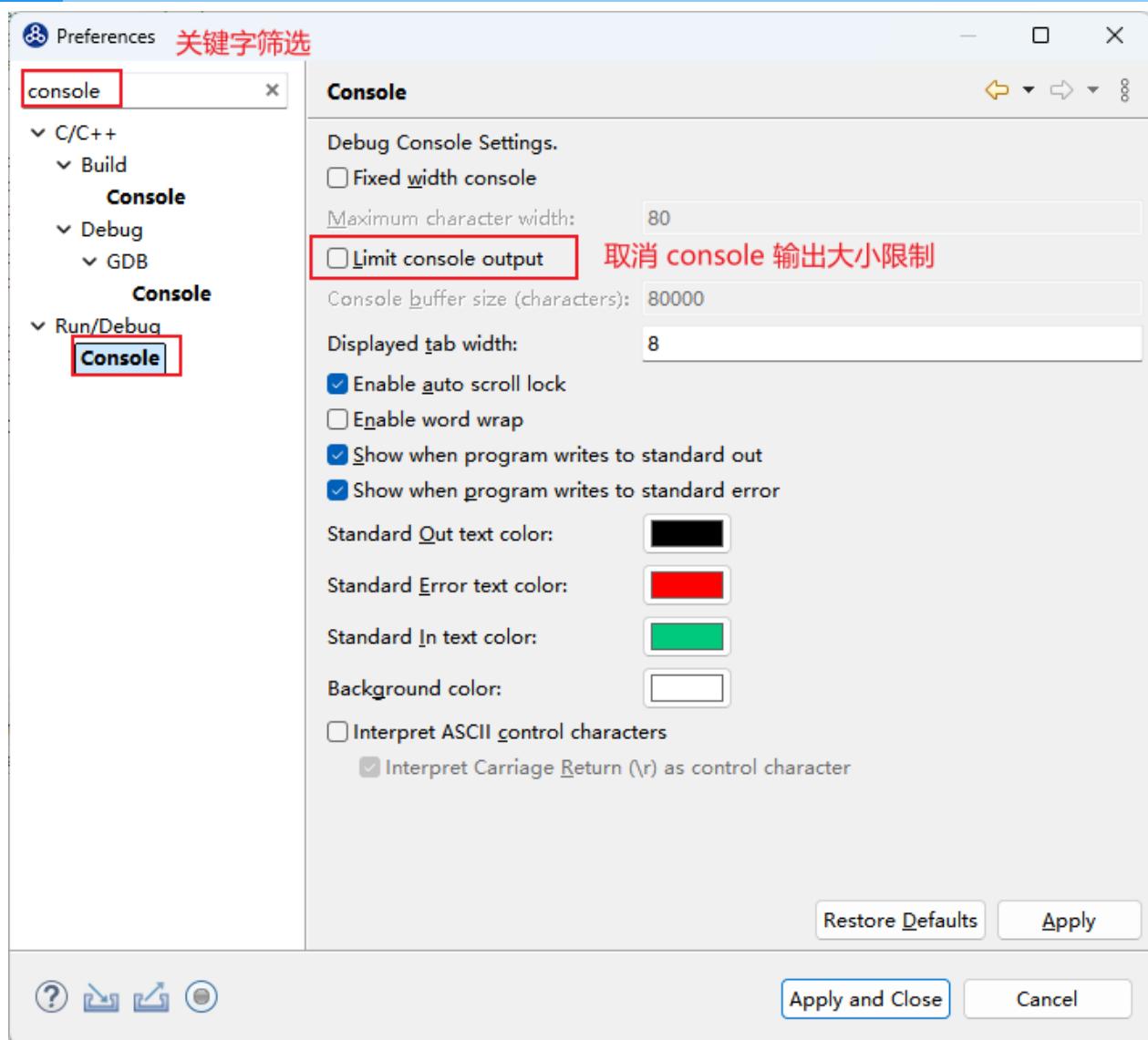
解决方案¶

因为在Console或者Terminal中，对输出的内容条数有限制，当输出的内容长度超过限制时，前面的内容会被冲掉，导致内容不完整，这样会解析失败。

需要调节 Console 或 Terminal 输出大小限制，确保数据没有被冲掉。

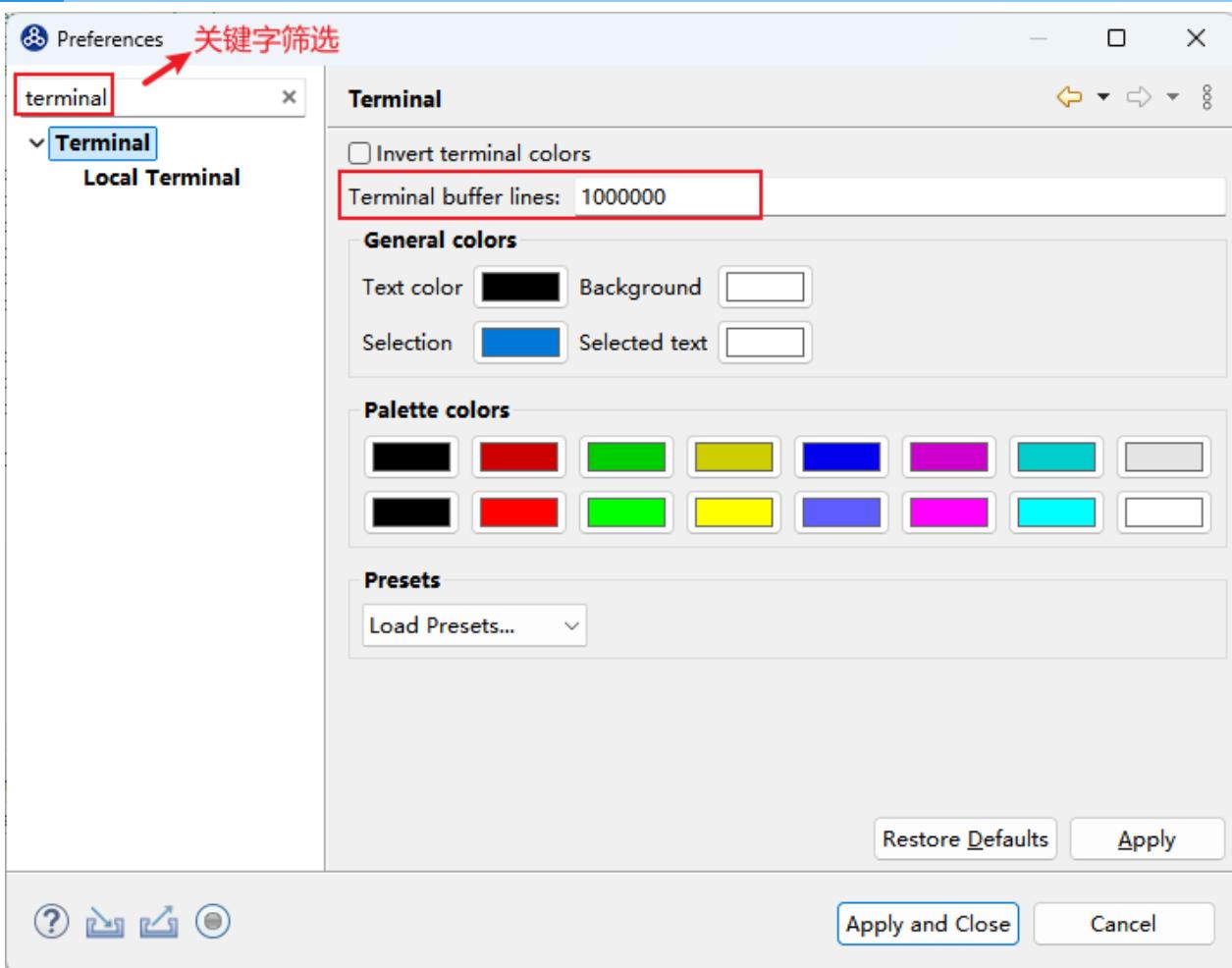
- 建议将Console中输出内容条限修改为不受限制。

Window->Preference 进入如下界面：



- 建议将Terminal中输出内容条限修改为一个较大的值。

Window->Preference 进入如下界面：



问题3：删掉 gmon.out 文件，再次解析，弹出 No files have been generated 错误弹框¶

手动删掉工程文件夹下的 gmon.out 文件，再次解析时出现 No files have been generated 的错误弹框

```

demo - amrwb_profiling_demo\application\amrwb-enc.c - NucleiStudio IDE
File Edit Source Refactor Source Navigate Search Project RV-Tools Run Window Help
Debug Project Explorer amrwb-enc.c
amrwb_profiling_demo
  Nuclei Settings
    Binaries
    Includes
    application
    Debug
  nuclei_sdk
    Components
    NMSIS
    SoC
  amrwb_profiling_demo_debug_jlink.launch
  amrwb_profiling_demo_debug_openocd.launch
  amrwb_profiling_demo_debug_qemu.launch
callgraph.out

130     memfwrite(outbuf, 1, n, out);
131
132     // check result
133     for (p = outbuf, i = 0; i < n; i++) {
134         if (*p++ != *pref++) {
135             printf("error!\n");
136             return 1;
137         }
138     }
139 }
140 free(inputBuf);
141 memfclose(out);
142 E_IF_exit(0);
143 wav_read_close(wav);
144 printf("finish!\n");
145
146 // TODO this is used for collect gprof and gcov data
147 // See Components/profiling/README.md about how to set the IDE project properties.
148 extern long gprof_collect(unsigned long interface);
149 extern long gcov_collect(unsigned long interface);
150 gprof_collect(2);
151 gcov_collect(2);
152
153
154 }

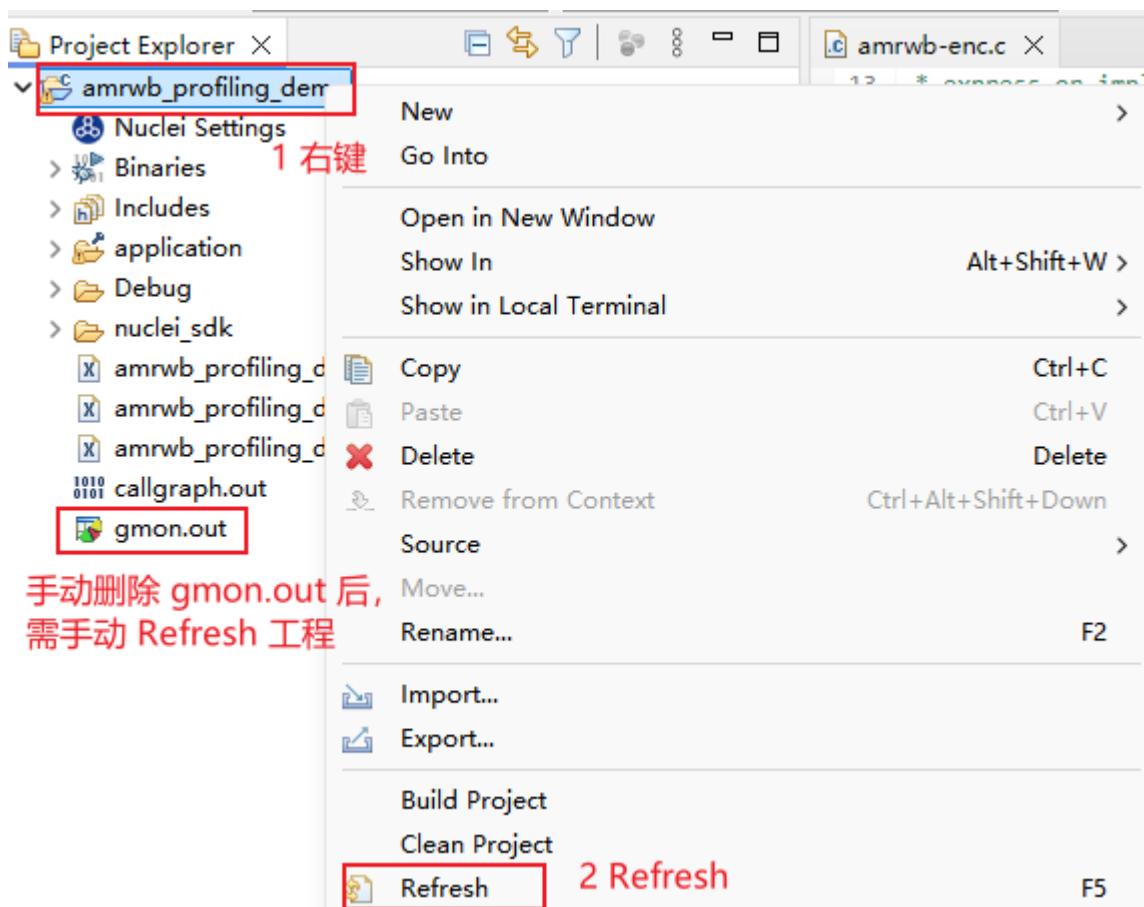
Console Progress Problems Executables Terminal gprof
COM6 ×
Message ×
No files have been generated.
OK

```

CREATE: C:\Users\shuzhuo\demo\amrwb_profiling_demo\Debug\application\src\az_isp.gcda
CREATE: C:\Users\shuzhuo\demo\amrwb_profiling_demo\Debug\application\src\autocorr.gcda
Dump coverage data finish

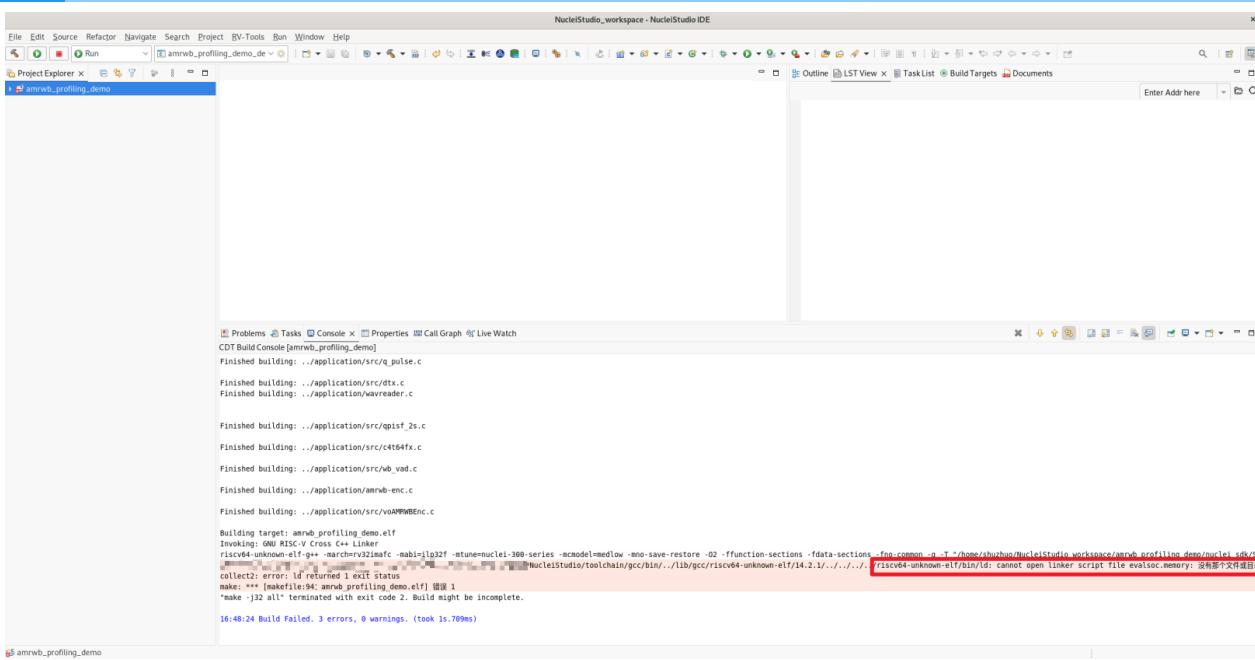
解决方案¶

手动删掉 gmon.out 文件后，需要手动刷新一下工程。



问题4：Linux 环境中使用 Nuclei studio导入 amrwb_profiling_demo.zip 编译报错¶

文档《[使用 Profiling 功能时可能遇到的一些问题](#)》中制作的用例有问题，导致使用Linux Nuclei studio导入amrwb_profiling_demo.zip 编译时报错
具体错误如下：evalsoc.memory: 没有那个文件或目录



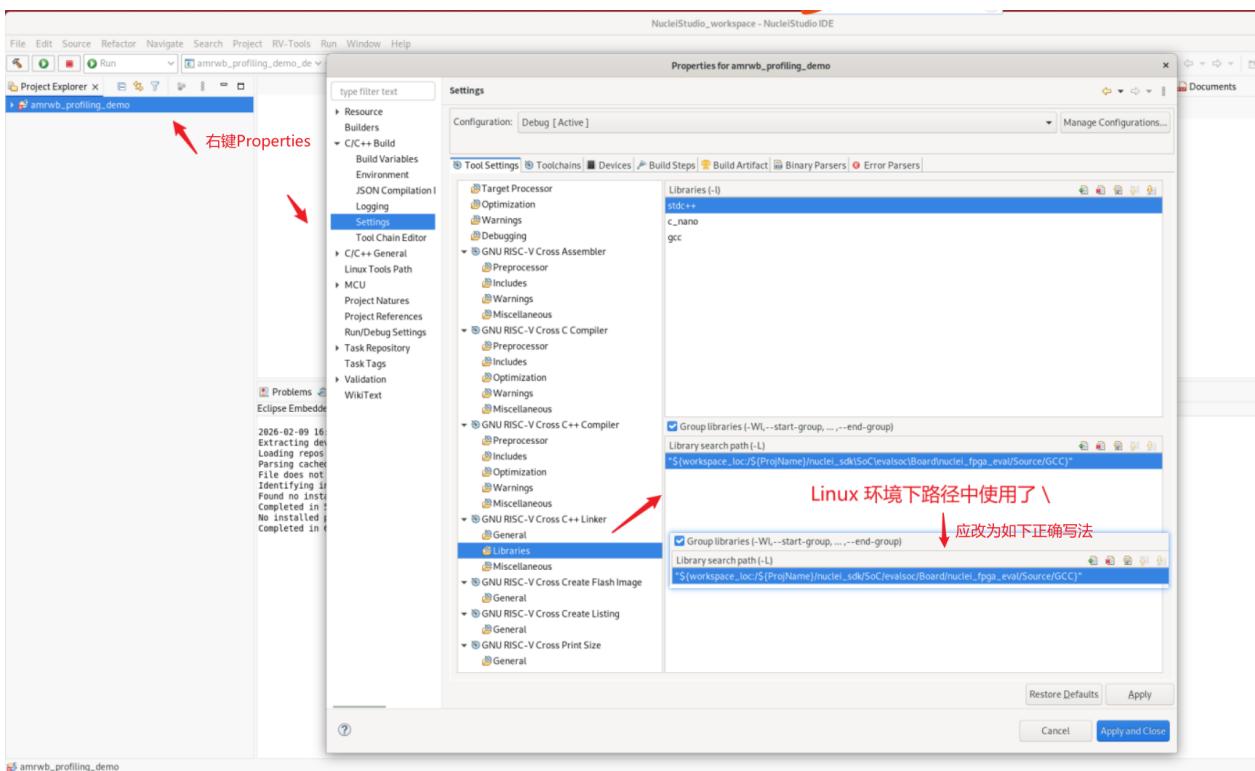
原因：是因为Linux环境中混入了Windows路径分隔符

****解决方法：****

可采取如下两种方法：

方法1. 文档中的zip包已经修复这个问题，下载新的用例包即可

方法2. 按照下图示意，将路径中的\改为/



Nuclei Studio 使用 Profiling 功能进行性能调优 举例

文档是基于 Nuclei Studio 的 2024.06 Windows/Linux 版本实测。

问题说明

Nuclei Studio 2024.06 提供 Profiling 功能、Call Graph 功能以及 Code coverage 功能，方便用户使用。简单描述如下：

- Profiling 功能：基于 binutils gprof 工具，可用于分析函数调用关系、调用次数、以及运行时间；通过 Profiling 抓取热点函数可以用来分析程序的瓶颈，以便进行性能优化。
- Call Graph 功能：基于 Profiling 功能，将函数调用关系、调用次数、以及运行时间用图展示出来，方便开发人员分析。
- Code coverage 功能：基于 gcc 编译器提供 gcov 工具，可用来查看源码文件的代码覆盖率，帮助开发人员确定测试用例是否足够充分，是否覆盖了被测代码的所有分支和路径。

在 [NucleiStudio_User_Guide.pdf](#) 相关章节对这几个功能已经有较详细的描述，这篇文档以一个例子来展示它们的实际应用。

解决方案

1 环境准备

所需材料：

- Nuclei Studio：[NucleiStudio 2024.06](#)，以 Windows 版本为例
- 用例：以 [AMR-WB-enc](#) 即自适应多速率宽带编码音频算法为例，用户可以移植自己的用例

基于 nuclei-sdk v0.6.0 移植 amrwbenc 裸机用例：

打开 Nuclei Studio 建立 amrwbenc 工程，然后移植 amrwbenc 源码，最终用例可正常运行。用户可以移植自己的用例，不同用例移植的细节各不相同，这一步不是这篇文档的重点，略过。

2 Profiling 功能

Nuclei studio 中 Profiling 功能基于 binutils gprof 工具。编译时需带特定的编译选项 -pg 来编译指定源码文件，编译成功后得到 ELF 文件，然后在实际开发板上运行并收集需要的 gmon.out 文

件，最终在 IDE 上以图形化的方式展示。所以还需要在用例末尾添加 gprof 数据收集代码，有两种方式：

- 方式1：移植 gprof 数据收集代码到自己的工程中，代码可以参考 [Profiling README](#)
- 方式2：基于 Nuclei Studio 中的 Profiling demo 进行改造，即用自己的用例替换掉 Profiling demo 工程的用例部分

下面示例采用后一种方法进行演示：

step1：新建 Profiling demo 工程

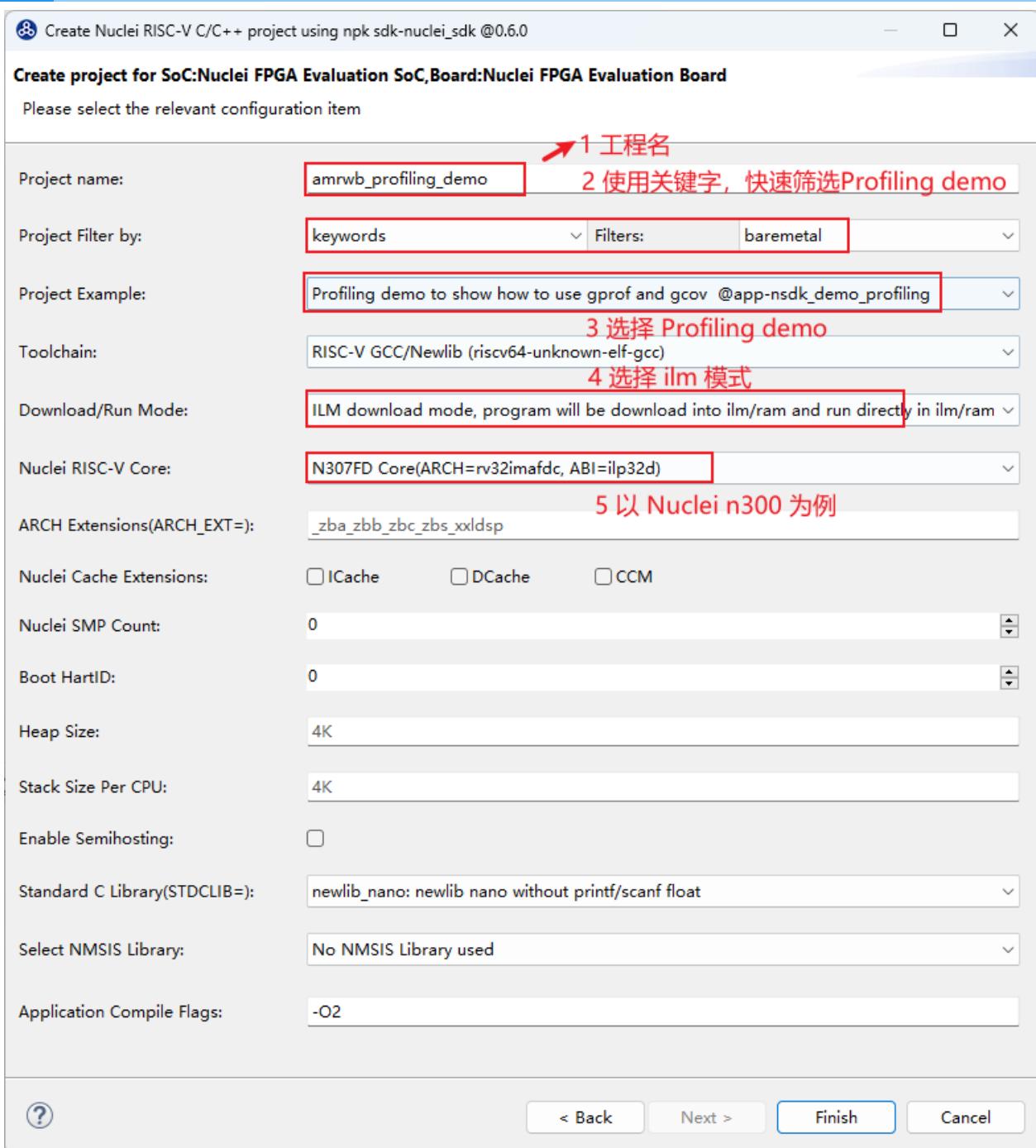
File->New->New Nuclei RISC-V C/C++ Project, 选择 Nuclei FPGA Evalution Board->sdk-nuclei_sdk @0.6.0

注意：Nuclei SDK 需选择 0.6.0 及以后版本才支持 Profiling 与 Code coverage 功能

可以根据需求选择不同的工具链：

1. RISC-V GCC/Newlib(riscv-unknown-elf-gcc)
2. RISC-V Clang/Newlib(riscv-unknown-elf-clang)
3. Terapines ZCC(zcc)

注意：Nuclei SDK 需选择 0.7.1及以后版本才支持 Terapines ZCC(zcc)工具链，若选择 Terapines ZCC(zcc)工具链使用profiling功能请同步修改Standard C Library(STDCLIB=)选项为newlib_small 注意：对于RISC-V Clang和Terapines ZCC工具链当前最新SDK（0.8.1）及之前版本仅支持profiling功能，未支持Code coverage功能



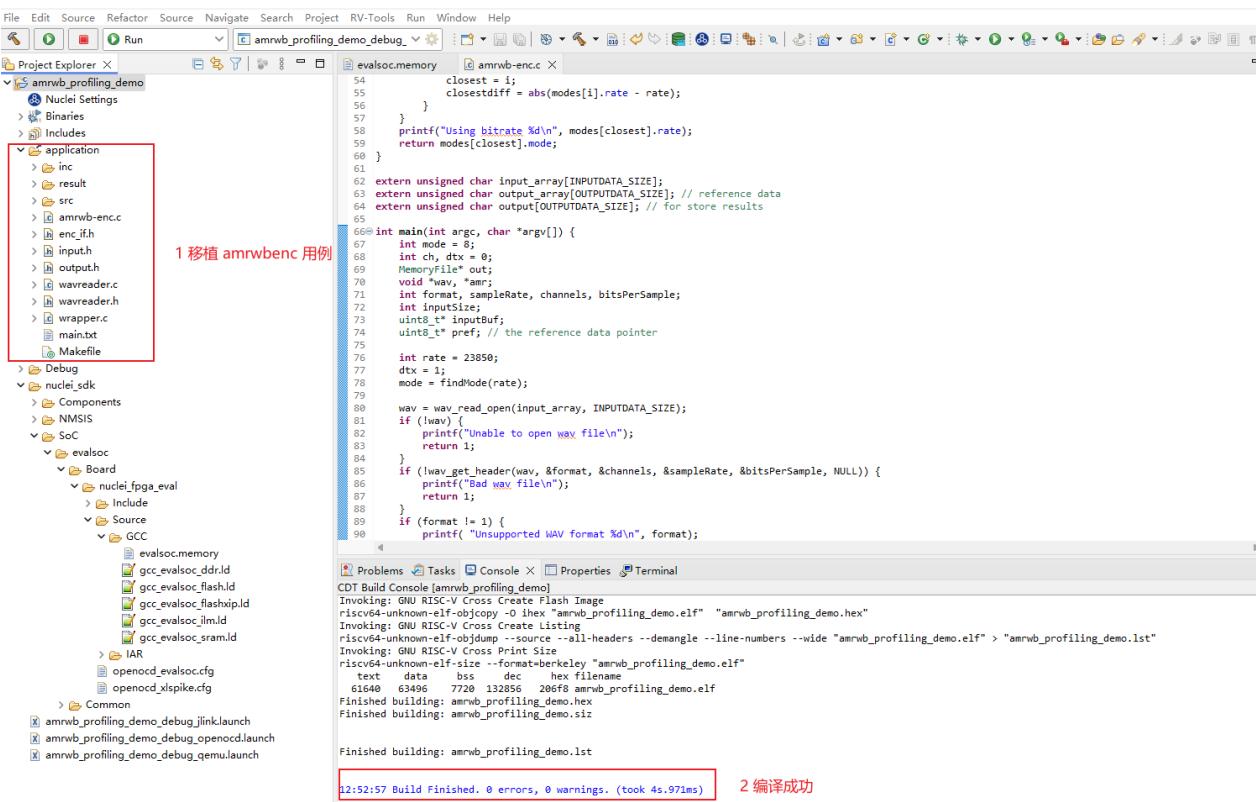
step2：基于 Profiling demo 工程移植 amrwbenc 裸机用例

删掉 Profiling demo 工程中 application 中的原始用例，替换成 amrwbenc 用例，形成如下目录结构，并确保能编译成功。

这里提供本示例使用的工程，有兴趣可以下载使用：

[优化前的工程下载链接](#)

下载 zip 包后，可以直接导入到 Nuclei Studio 中运行(导入步骤：File->Import->Existing Projects into Workspace->Next->Select archive file->选择zip压缩包->next即可)



注意：在Linux环境中使用nuclei studio导入旧的用例包可能会出现报错（找不到evalsoc.memory），这是因为Linux环境混入Windows路径分隔符导致的，2026-02-09日修复了这个问题，此文档用例链接已经更新，可以重新下载新用例包，或者直接修改错误路径，具体可参考[Profiling与Code coverage功能可能遇到的问题](#)问题4

step3：在用例结尾处添加gprof数据收集代码，并添加-pg编译选项，重新编译代码

在main函数的结尾处添加gprof数据收集代码：

```
int main(int argc, char *argv[]) {
    /*
     * 代码省略
     */

    /*
     * 在main函数的结尾处添加gprof数据收集代码
     */
    // TODO this is used for collect gprof and gcov data
    // See Components/profiling/README.md about how to set the IDE
    // project properties
    extern long gprof_collect(unsigned long interface);
    gprof_collect(2);

    return 0;
}
```

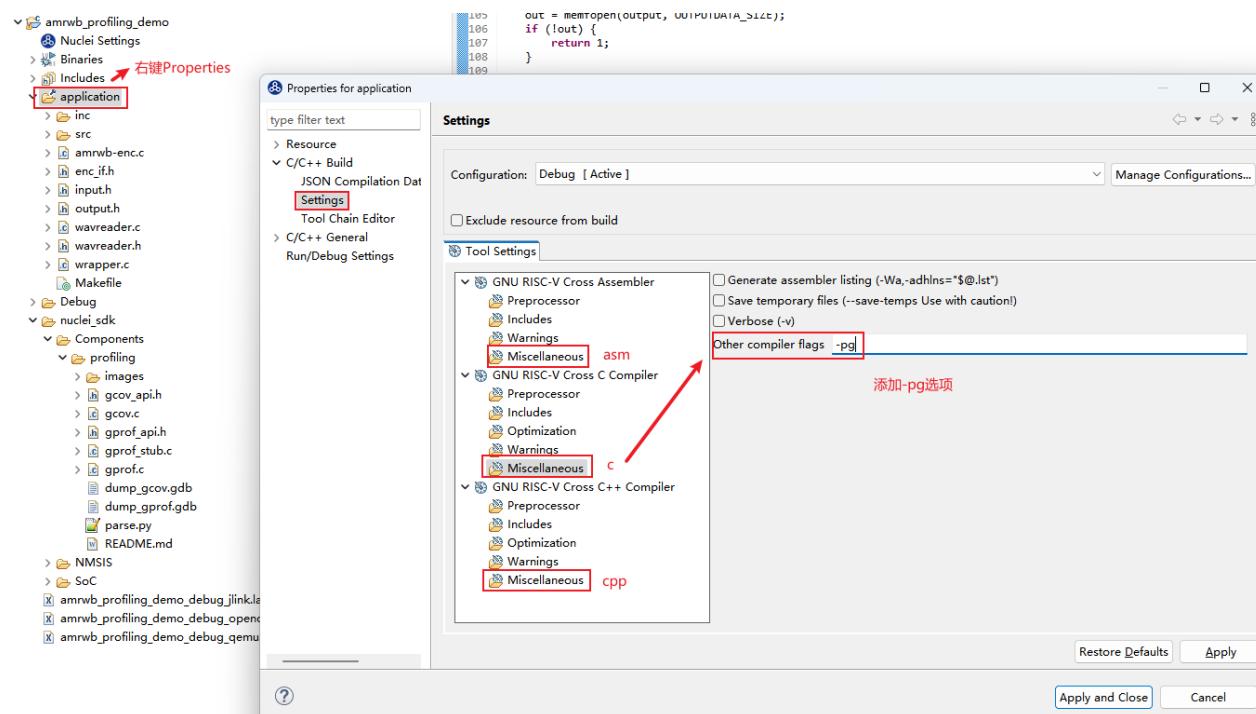
收集 gprof data 有三种方式，通过入参不同进行区分：

- gprof_collect(0)：在缓冲区中收集 gprof 或 gcov 数据，在调试程序时可以使用 GDB 脚本转储 gcov 或 gprof 二进制文件
- gprof_collect(1)：使用 semihost 直接将 gprof 或 gcov 数据写入文件中
- gprof_collect(2)：直接在 Console 或 Serial Terminal 中打印 gcov 或 gprof 数据，然后可以通过IDE中 Parse and Generate HexDump 功能进行解析数据并保存到PC上

详情可参考 [Profiling README](#)，这里以将 gprof data 打印到串口（Console 或 Serial Terminal）为例。

对需要进行profiling的代码添加 -pg 编译选项，重新编译代码：

注意：选择 application, 对关键代码添加 -pg 编译选项，这个用例只有 C 代码，只对 C 代码添加 -pg 编译选项即可



step4：运行程序

有几种方式可以运行程序：

- qemu 模拟器（不需要硬件，简单跑一下流程，测试结果不准确）
- 上板测试（基于定时器采集数据）
- 基于 xl_cpumodel (Nuclei Near Cycle Model)，参考：[通过Profiling展示Nuclei Model NICE/VNICE指令加速](#)

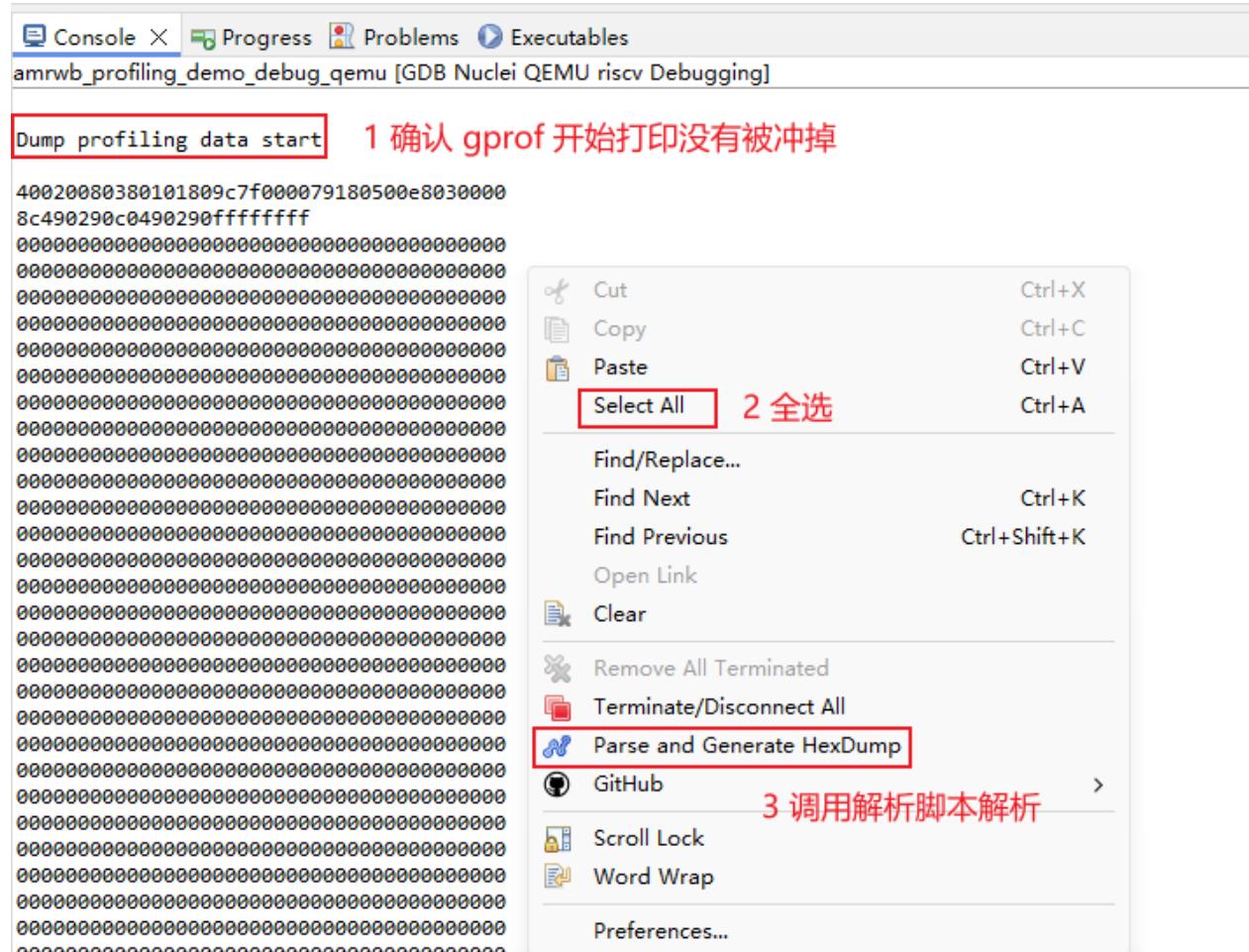
这一篇文章只介绍 qemu 仿真与上板测试两种方式，qemu 收集的数据打印到 Console 口，上板实际运行输出到 Nuclei Studio 的 Serial Terminal 口。

step5：解析 gprof 数据

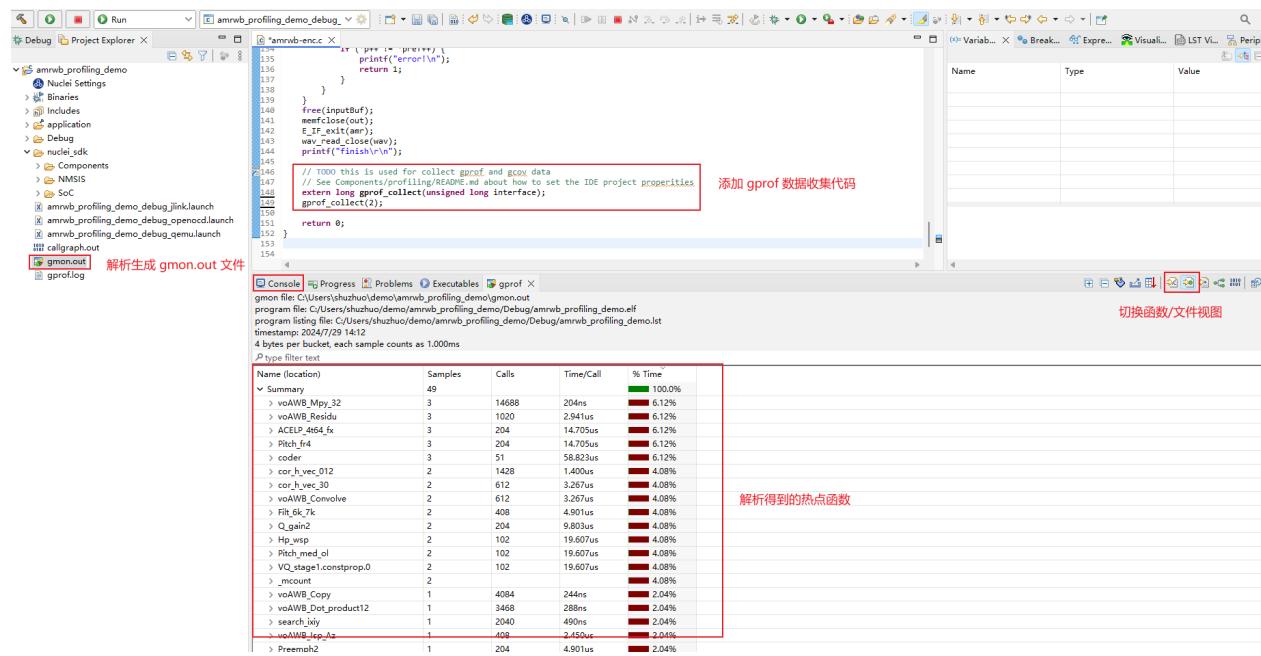
开始解析 gprof 数据。注意：这一步可能遇到一些问题，解决方法可参考 [Profiling与Code coverage 功能可能遇到的问题](#)

- 在 qemu 上测试, log 打印到 Console 口

注意：qemu 仅用来模拟展示，如果希望得到准确的热点函数，需要上板测试。



解析完成后，会在当前工程目录下生成 gmon.out，双击打开展示：

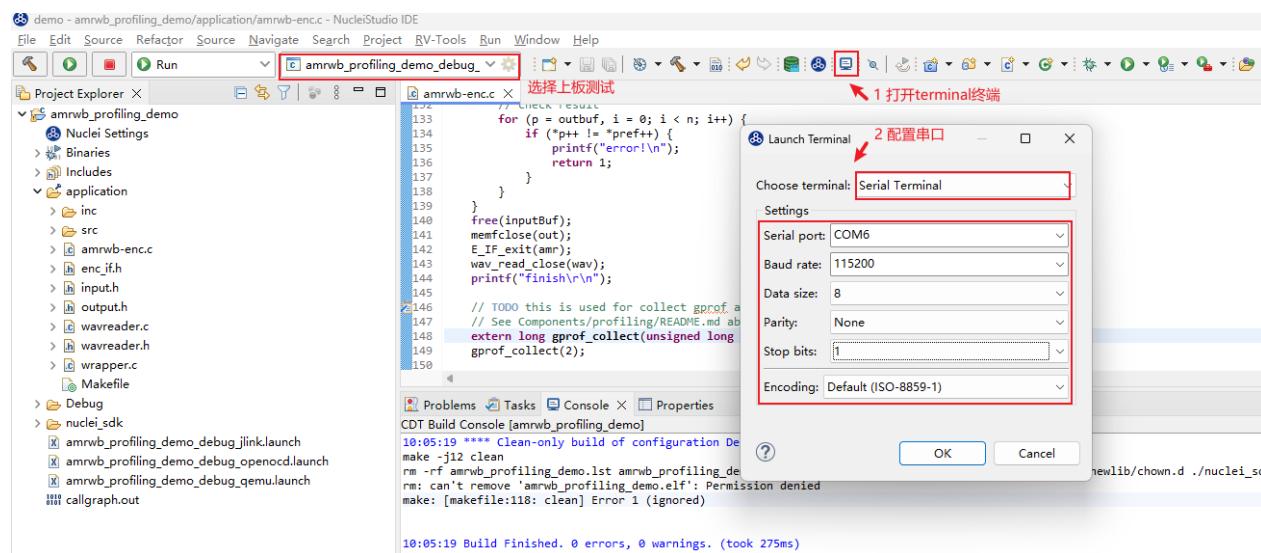


• 上板测试

上板测试的步骤与 qemu 类似，唯一不同的是 gprof 数据输出到 Serial Terminal 上。

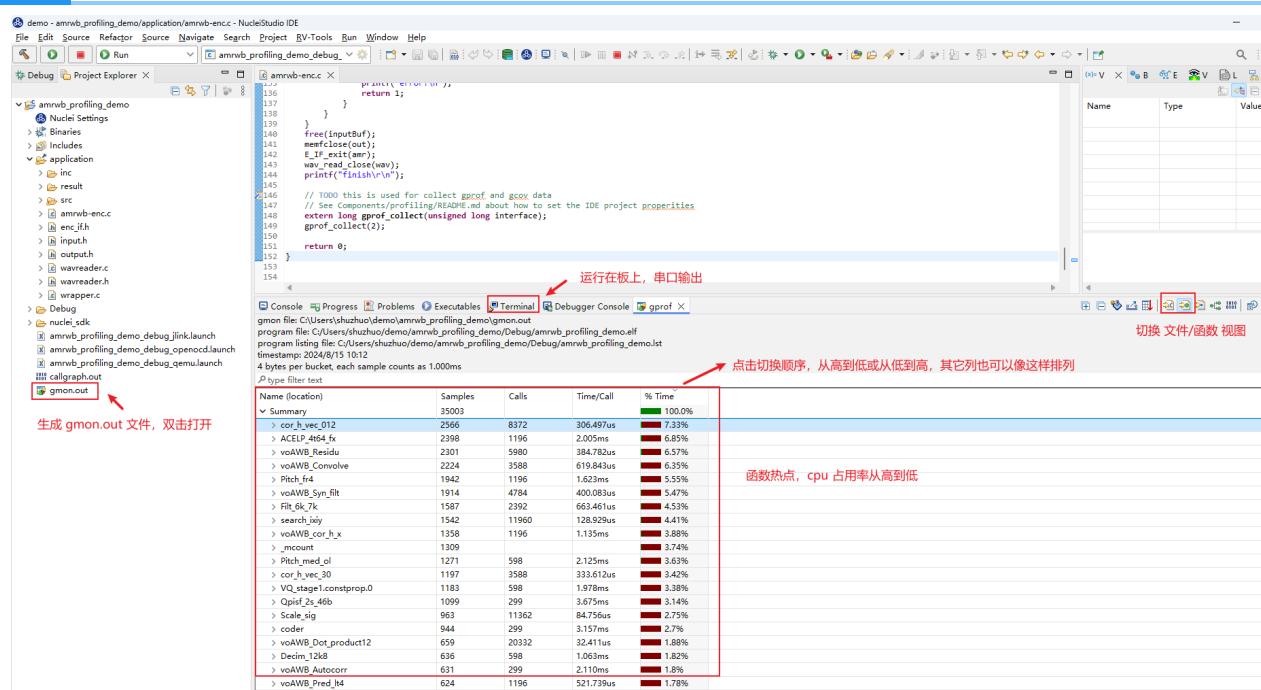
配置 Serial Terminal:

注意:如果串口工具已经打开，确保每次运行 gprof 前，清除掉串口打印（鼠标右键-> Clear Terminal），避免对数据解析产生影响。



同样，全选 log，右键选择 Parse and Generate HexDump 功能，就会在工程文件夹下生成 gmon.out 文件，刷新工程后，就可以双击打开这个gmon.out 文件。

如下图是在板子上实际运行得到的 gprof 数据：



从而得到 TOP5 热点函数为（实际上板测试）：

```
cor_h_vec_012
ACELP_4t64_fx
voAWB_Residu
voAWB_Convolve
voAWB_Syn_filt
```

获得热点函数后，可以从热点函数入手开始优化，优化 TOP 函数往往可以事半功倍。

step6：优化热点函数

有如下几种方法优化热点函数：

- 调节编译器参数，针对整个工程或单独算子使用 O2/O3/Ofast 等优化等级，开启 -finline-functions -funroll-all-loops 等优化选项
- 针对算法进行优化，使用更好的算法实现热点函数
- 使用 RISC-V 扩展指令（RVP/RVV 扩展等）优化

这里以 RVP 扩展为例，按照热点函数从高到低，用 RVP 扩展来优化。需要确定所用硬件支持 RVP 扩展。

举例如下：

TOP1 热点函数为 cor_h_vec_012，分析函数，尝试使用 RVP 扩展优化：

如下以`#if defined __riscv_xxldspn3x`隔开的代码表示使用 Nuclei N3 P 扩展指令优化的代码。其中`_RV_DSMALDA`是一条 Nuclei N3 P 扩展指令，实现了一次完成 4 笔 int16 相乘，最后累加，结果存放到 int64 变量中。

这些指令 Intrinsic API 可参考 [Nuclei P 扩展指令 Intrinsic API](#)

具体的 RVP 指令手册，请联系芯来科技获取。

优化后的工程如下，可以与优化之前的工程做对比，只优化了`cor_h_vec_012` 算子：

[优化后的工程下载链接](#)

使用 Nuclei N3 P 扩展指令优化的代码片段如下：

```
void cor_h_vec_012(
    Word16
    h[],                                /* (i) scaled impulse
    response                           */
    Word16 vec[],                         /* (i) scaled vector
(/8) to correlate with h[] */
    Word16 track,                          /* (i) track to
use                               */
    Word16 sign[],                         /* (i) sign
vector                            */
    Word16 rrixix[]                       /* (i) correlation of h[x] with h[x]      */
[NB_POS],                                /* (o) result of
    Word16 cor_1[],                      /* (o) result of
correlation (NB_POS elements) */
    Word16 cor_2[]                       /* (o) result of
correlation (NB_POS elements) */
)
{
    Word32 i, j, pos, corr;
    Word16 *p0, *p1, *p2,*p3,*cor_x,*cor_y;
    Word32 L_sum1,L_sum2;
    cor_x = cor_1;
    cor_y = cor_2;
    p0 = rrixix[track];
    p3 = rrixix[track+1];
    pos = track;

    for (i = 0; i < NB_POS; i+=2)
    {
        p1 = h;
        p2 = &vec[pos];
#if defined __riscv_xxldspn3x
        Word32 tmp1, tmp2;
```

```
int64_t sum64_1, sum64_2;
int64_t p64_1, p64_2;
sum64_1 = 0;
sum64_2 = 0;
for (j=62-pos ;(j - 4) >= 0; j -= 4)
{
    p64_1 = *_SIMD64(p1)++;
    tmp1 = __RV_PKBB16(*(p2 + 1), *p2);
    tmp2 = __RV_PKBB16(*(p2 + 3), *(p2 + 2));
    p64_2 = __RV_DPACK32(tmp2, tmp1);
    sum64_1 = __RV_DSMALDA(sum64_1, p64_1, p64_2);

    tmp1 = __RV_PKBB16(*(p2 + 2), *(p2 + 1));
    tmp2 = __RV_PKBB16(*(p2 + 4), *(p2 + 3));
    p64_2 = __RV_DPACK32(tmp2, tmp1);
    sum64_2 = __RV_DSMALDA(sum64_2, p64_1, p64_2);
    p2 += 4;
}
L_sum1 = (Word32)sum64_1;
L_sum2 = (Word32)sum64_2;
for ( ;j >= 0; j--)
{
    L_sum1 += *p1 * *p2++;
    L_sum2 += *p1++ * *p2;
}
#endif
L_sum1 += *p1 * *p2;
L_sum1 = (L_sum1 << 2);
L_sum2 = (L_sum2 << 2);

corr = (L_sum1 + 0x8000) >> 16;
cor_x[i] = vo_mult(corr, sign[pos]) + (*p0++);
corr = (L_sum2 + 0x8000) >> 16;
cor_y[i] = vo_mult(corr, sign[pos + 1]) + (*p3++);
pos += STEP;

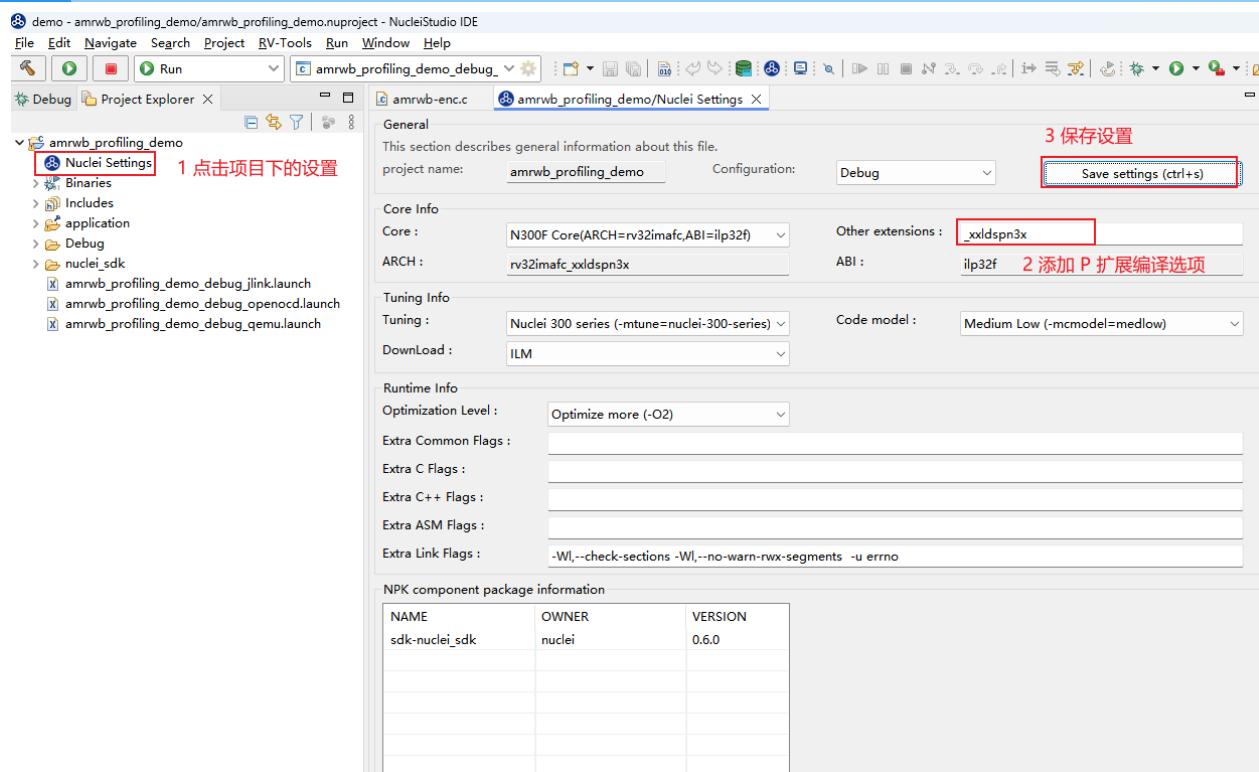
p1 = h;
p2 = &vec[pos];
#if defined __riscv_xxldspn3x
sum64_1 = 0;
sum64_2 = 0;
for (j=62-pos ;(j - 4) >= 0; j -= 4)
{
    p64_1 = *_SIMD64(p1)++;
    tmp1 = __RV_PKBB16(*(p2 + 1), *p2);
    tmp2 = __RV_PKBB16(*(p2 + 3), *(p2 + 2));
    p64_2 = __RV_DPACK32(tmp2, tmp1);
```

```
    sum64_1 = __RV_DSMALDA(sum64_1, p64_1, p64_2);

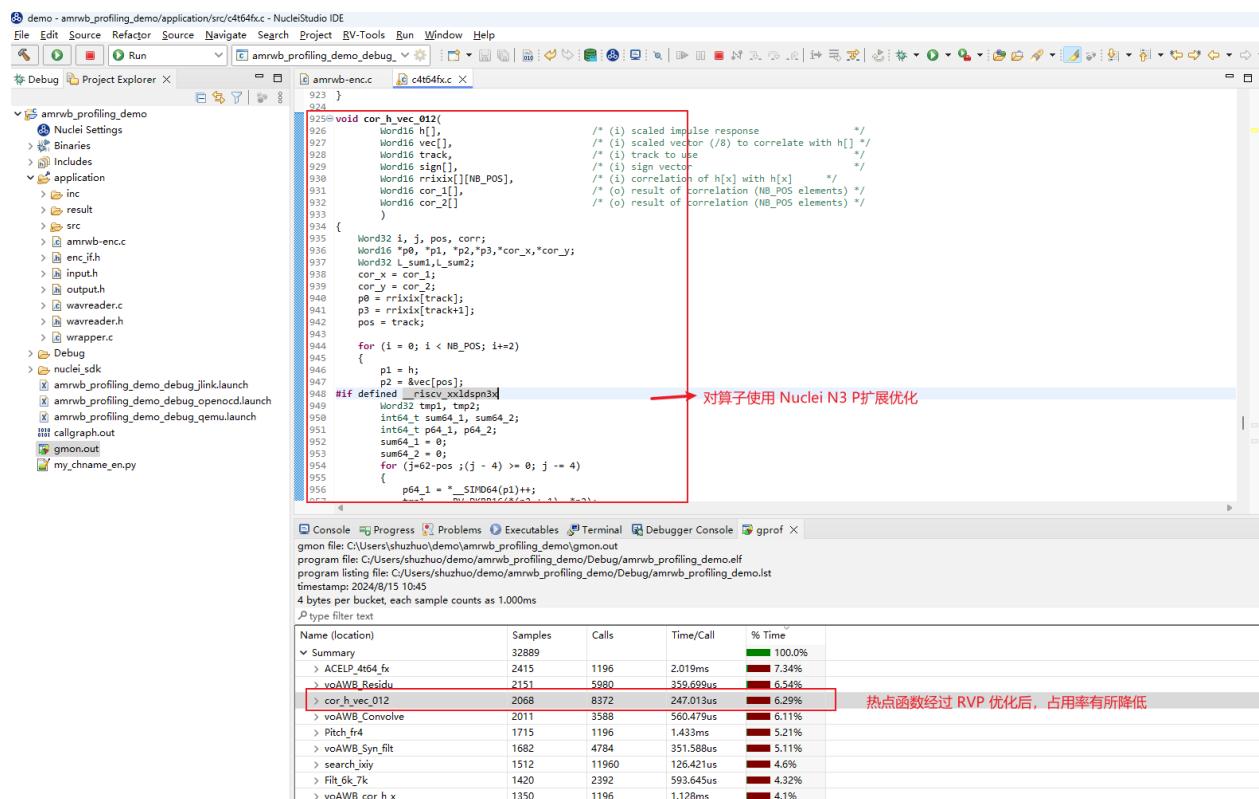
    tmp1 = __RV_PKBB16(*(p2 + 2), *(p2 + 1));
    tmp2 = __RV_PKBB16(*(p2 + 4), *(p2 + 3));
    p64_2 = __RV_DPACK32(tmp2, tmp1);
    sum64_2 = __RV_DSMALDA(sum64_2, p64_1, p64_2);
    p2 += 4;
}
L_sum1 = (Word32)sum64_1;
L_sum2 = (Word32)sum64_2;
for ( ;j >= 0; j--)
{
    L_sum1 += *p1 * *p2++;
    L_sum2 += *p1++ * *p2;
}
#endif
L_sum1 += *p1 * *p2;
L_sum1 = (L_sum1 << 2);
L_sum2 = (L_sum2 << 2);

corr = (L_sum1 + 0x8000) >> 16;
cor_x[i+1] = vo_mult(corr, sign[pos]) + (*p0++);
corr = (L_sum2 + 0x8000) >> 16;
cor_y[i+1] = vo_mult(corr, sign[pos + 1]) + (*p3++);
pos += STEP;
}
return;
}
```

这个算子进行 P 扩展优化后，编译时务必带上 dsp 扩展选项进行编译，如下图所示：



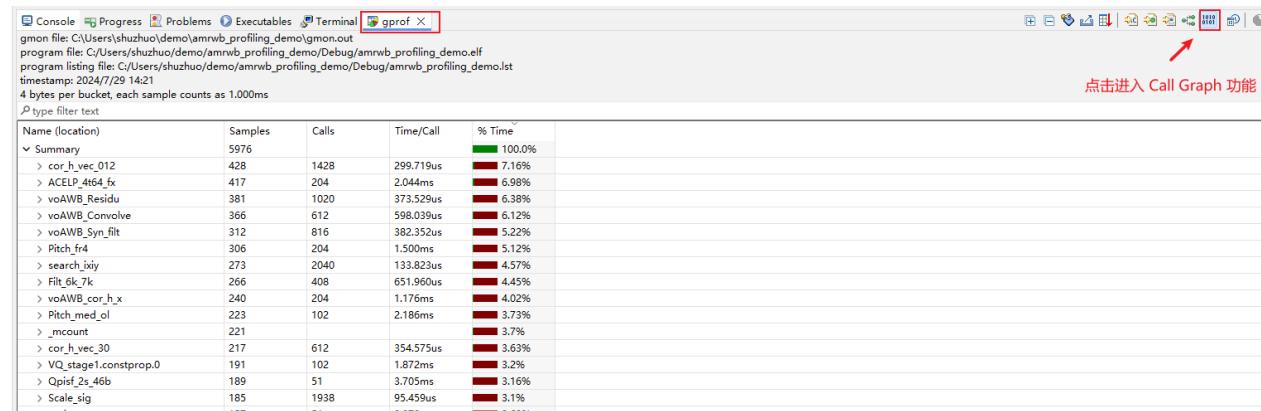
CLean Project 并重新编译，重新跑一次profiling，可以看到优化效果，`cor_h_vec_012` 函数占用率有所下降，函数调用时间也有所减少。



注意： 上述仅提供简单的示例，用户可以依次对热点函数进行分析并优化，运行过程中由于采样等原因，导致 TOP 函数分布有所波动，这是正常的，最终精确的分析需要统计最终的总 cycle 数，然后计算提升比。

2 Call Graph 功能

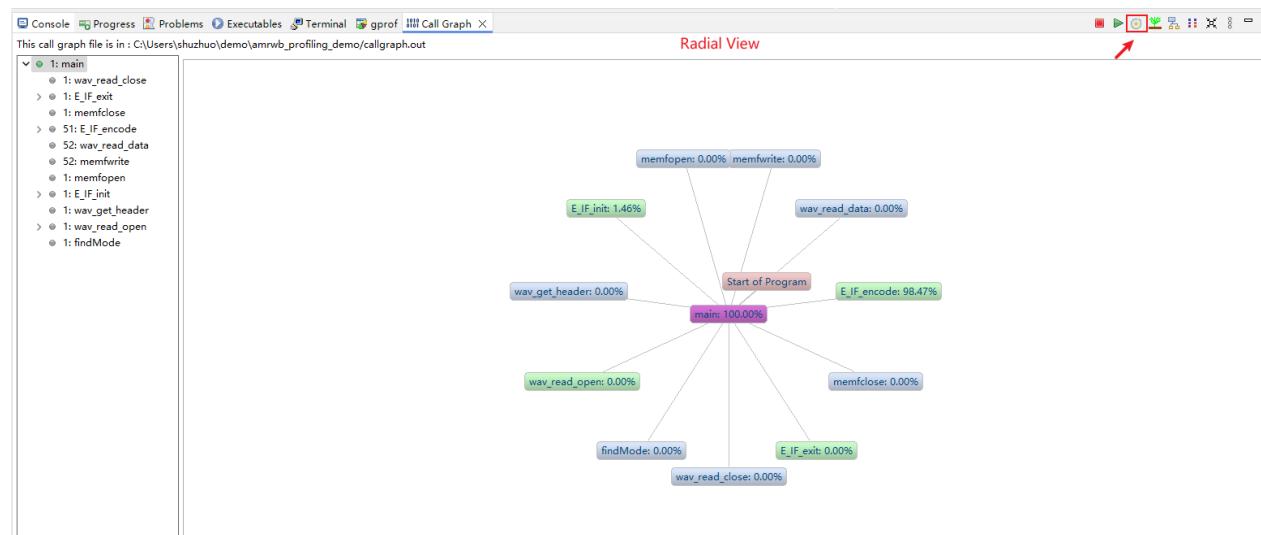
Nuclei Studio 中 Call Graph 主要是通过分析 Profiling 的数据来获取到程序中函数的调用关系。



Call Graph 功能包括如下几种视图：

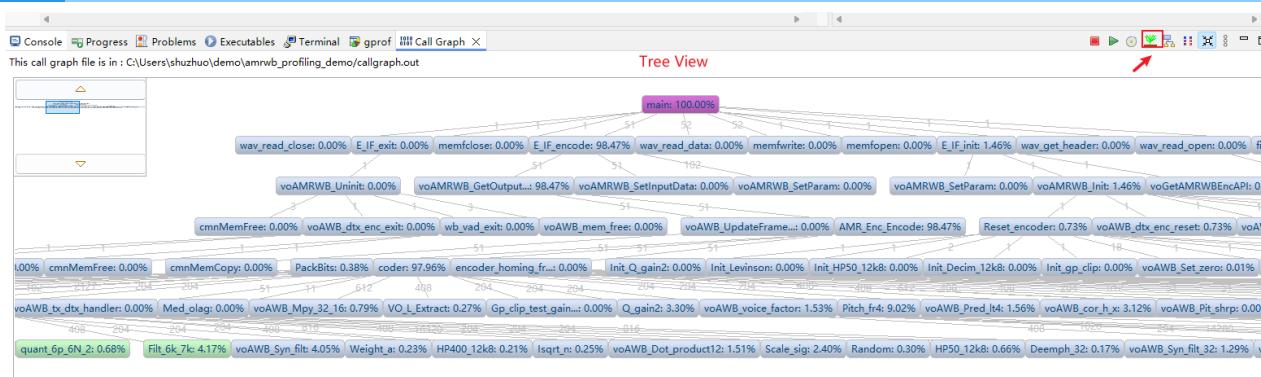
- Radial View

本视图中展示了程序的调用关系。



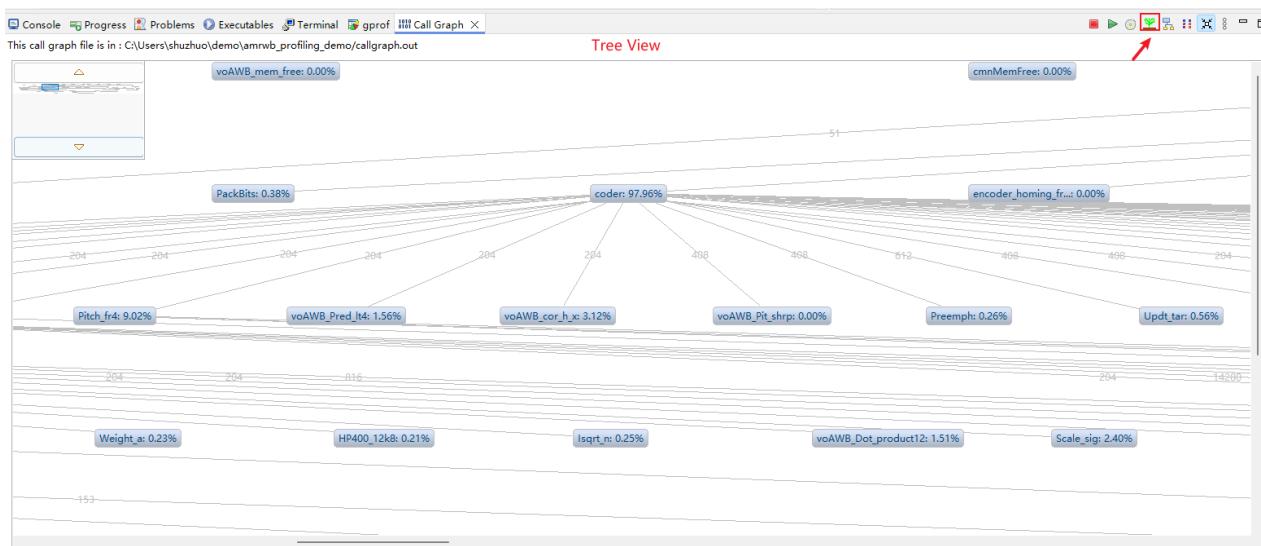
- Tree View

展示了 Radial View 中所选中的程序的调用关系、耗时所占比率、调用次数等信息；选中某一个函数，可以查看到它的父节点以及子节点等信息。



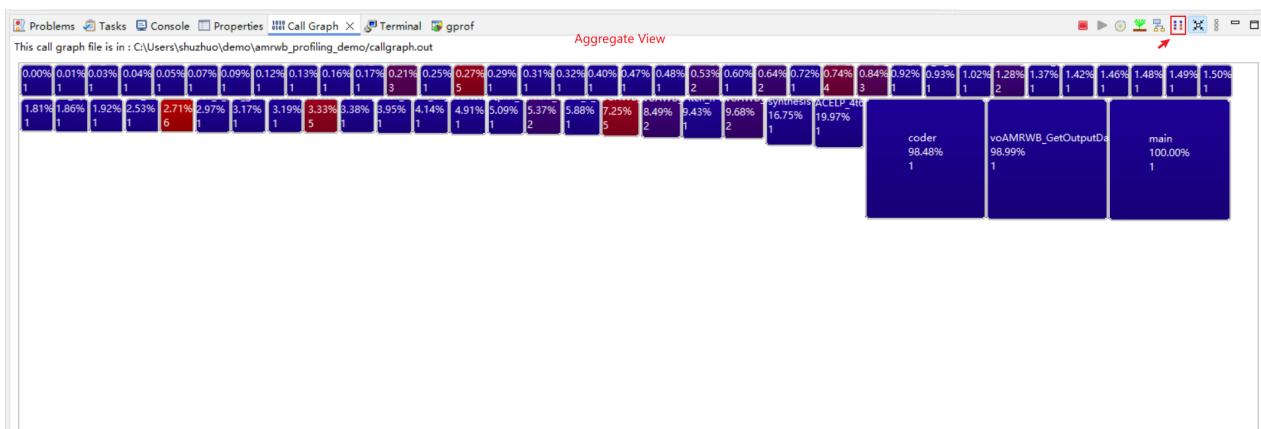
- Level View

与 Tree View 有点类似，展示了程序的调用关系以及调用次数。



- Aggregate View

以方图的方式，非常直观的展示了程序的耗时关系。



3 Code coverage 功能¶

Nuclei studio 中 Code coverage 功能基于 gcc 编译器提供的 gcov 工具，编译时需带特定的编译选项 `-coverage` 来编译指定源码文件，编译成功后得到 ELF 文件，然后在实际开发板上运行并收集需要的 coverage 文件(gcda/gcno 文件)，最终在 IDE 上以图形化的方式展示。

使用方法与 Profiling 功能类似，这里仅对不同的地方进行说明：

step1：新建 Profiling demo 工程

step2：基于 Profiling demo 工程移植 amrwbenc 裸机用例

step3：添加 gcov 数据收集代码，并添加 `-coverage` 编译选项，重新编译代码

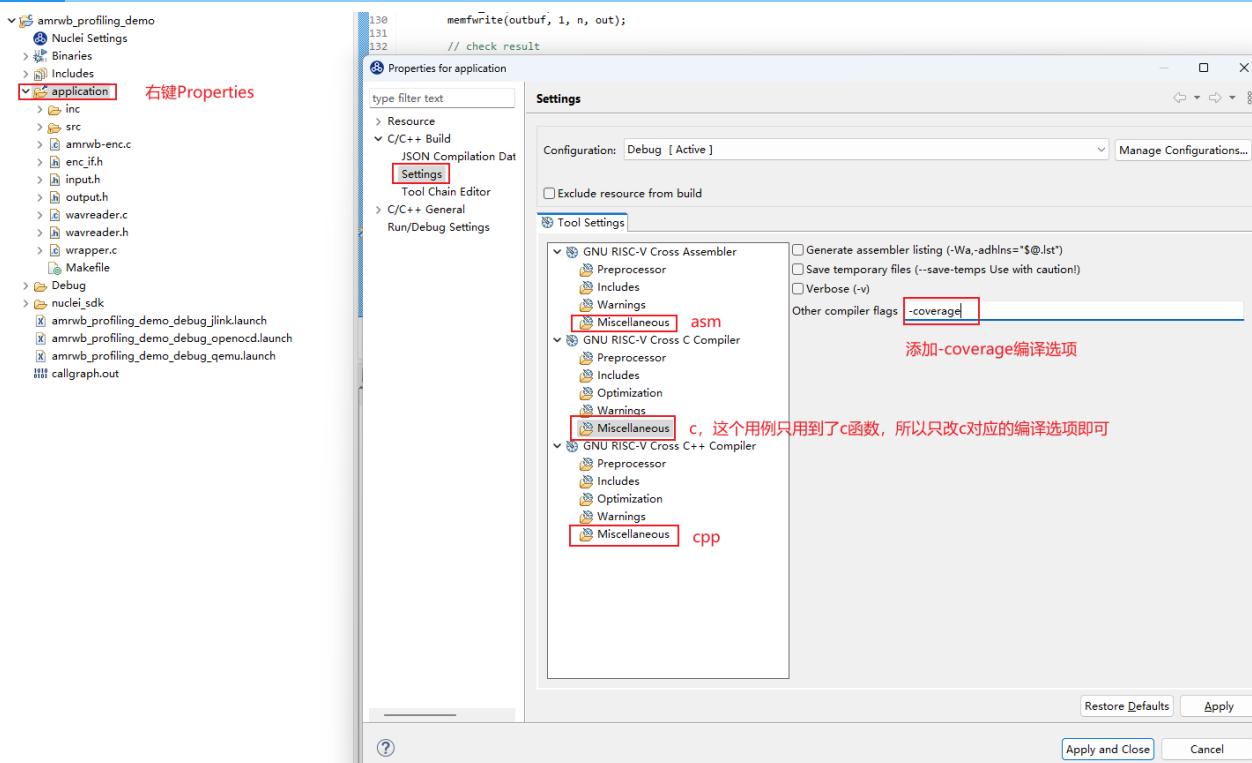
在main函数的结尾处添加gprof数据收集代码：

```
int main(int argc, char *argv[]) {
    /*
     * 代码省略
     */

    /*
     * 在main函数的结尾处添加 gcov 数据收集代码
     */
    // TODO this is used for collect gprof and gcov data
    // See Components/profiling/README.md about how to set the IDE
    project properties
    extern long gcov_collect(unsigned long interface);
    gcov_collect(2);

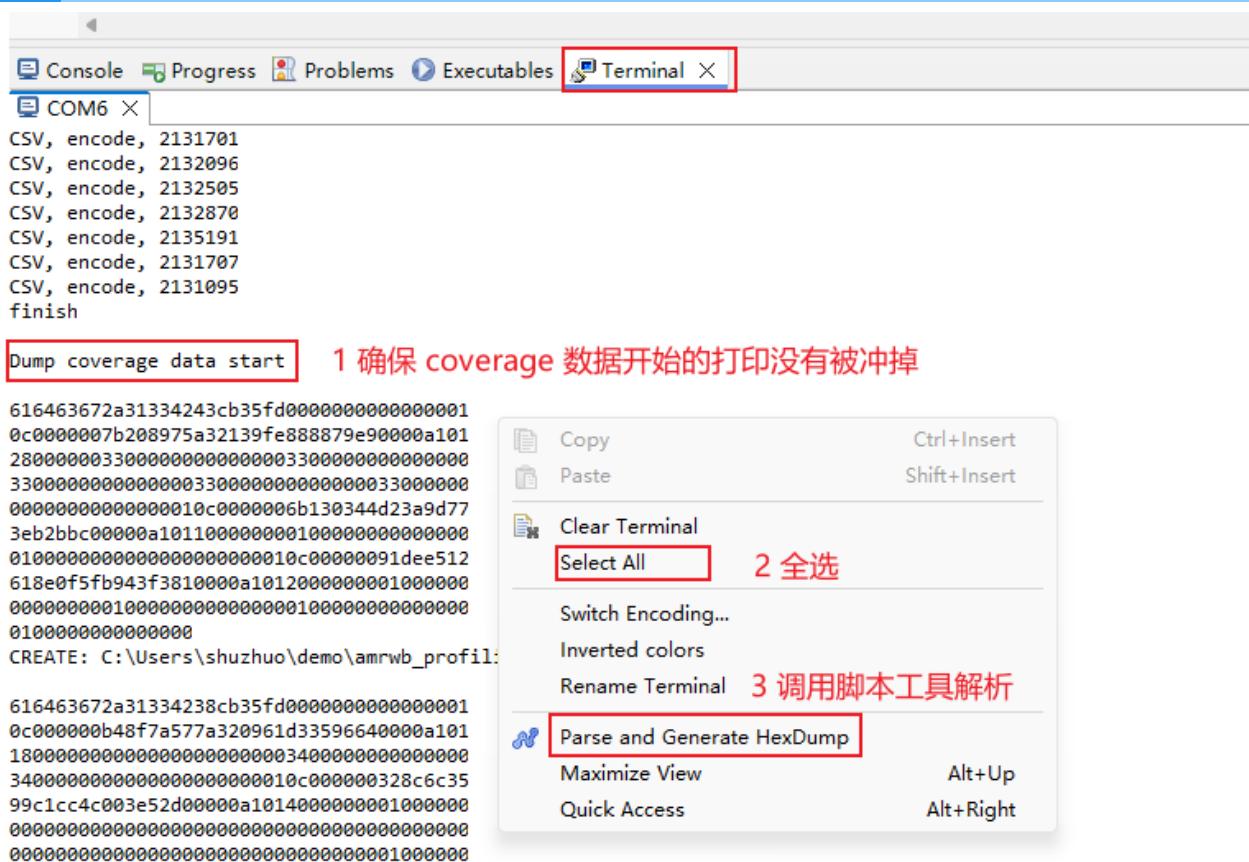
    return 0;
}
```

添加`-coverage`编译选项，重新编译代码：

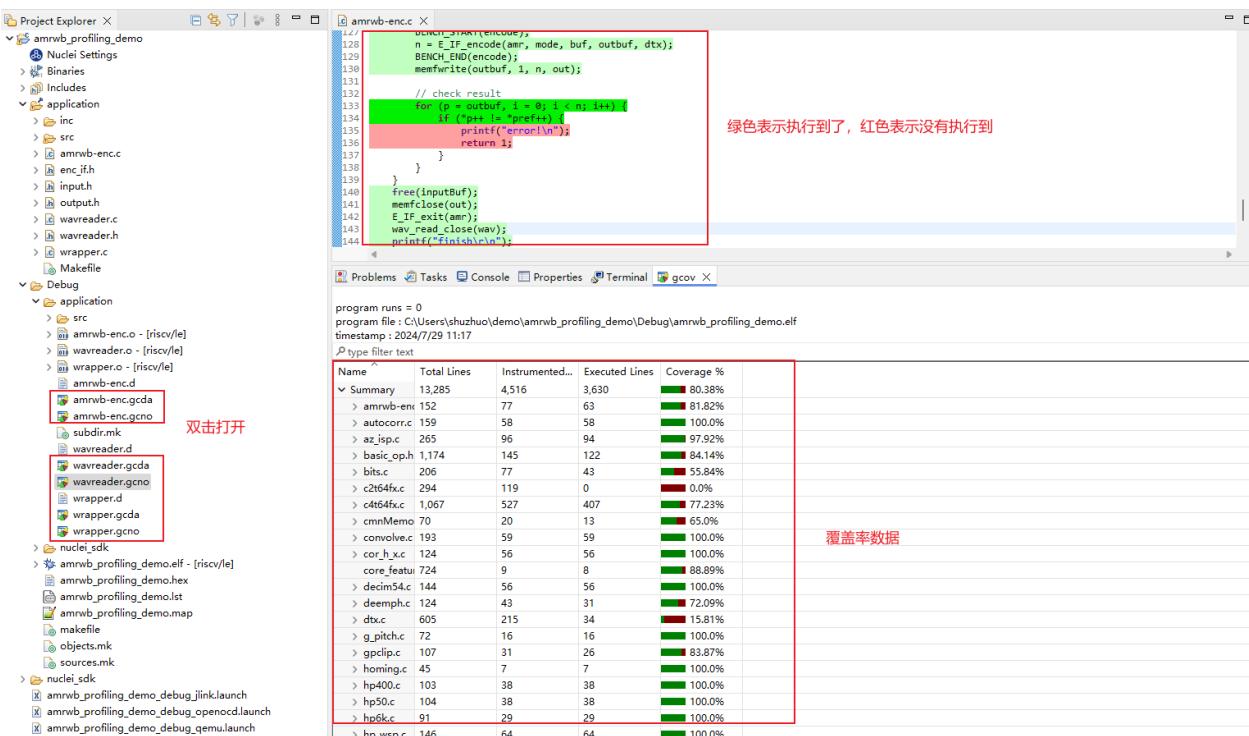


step4：运行程序

可以在qemu中模拟运行，或者上板实际运行都可以（统计覆盖率，不涉及到性能分析，所以使用qemu或者上板测试都可以）。



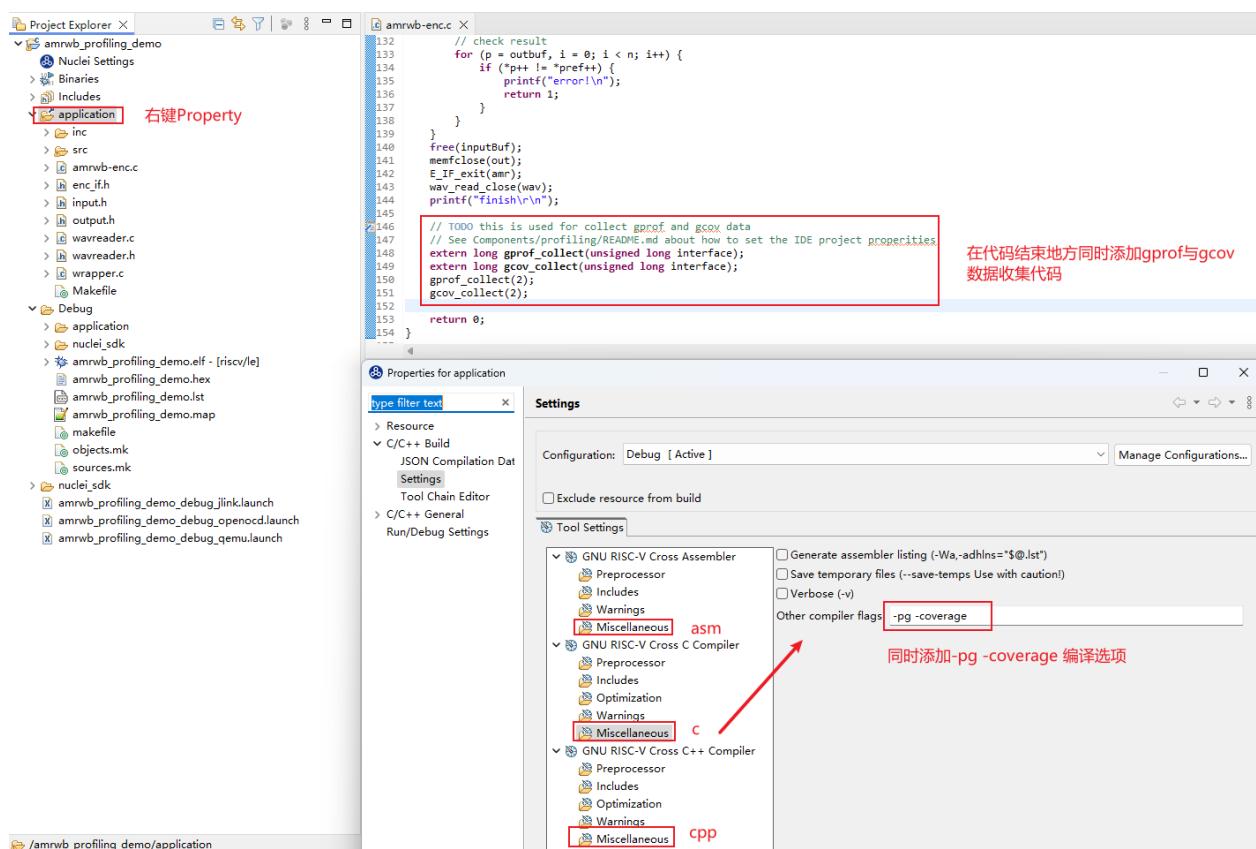
解析之后，在Debug->application文件夹下生成了 gcda 与 gcno 文件，双击打开即可



4 补充

1. Profiling 与 Code coverage 功能可以同时打开，只需添加一起收集 Profiling 数据与 Code coverage 数据的代码，并在编译时添加 -pg -coverage 编译选项。

```
// TODO this is used for collect gprof and gcov data
// See Components/profiling/README.md about how to set the IDE
project properties
extern long gprof_collect(unsigned long interface);
extern long gcov_collect(unsigned long interface);
gprof_collect(2);
gcov_collect(2);
```



1. 使用Profiling可能遇见的问题：
2. 片上内存不足，打印日志中有错误打印，gprof/gcov data 需要占用一定大小空间
3. Console 或 Terminal 收集的数据不全导致解析数据不正确，需确认数据没有被冲掉，需要调节 Console 或 Terminal 输出大小限制
4. 手动删掉 gmon.out 文件，再次解析，弹出 No files have been generated 错误弹框

上述具体解决方法可参考 [Profiling与Code coverage功能可能遇到的问题](#)

通过Profiling展示Nuclei Model NICE/VNICE指令加速¶

Nuclei Model 已支持 Windows/Linux 版本，此文档测试都是基于 Nuclei Studio 的 Windows 版本 (>= 2025.10) 完成的。

背景描述¶

Nuclei Model Profiling¶

在[Nuclei Studio使用Profiling功能进行性能调优举例](#)中已经通过 qemu 以及上板测试两种运行方式展示了如何在IDE中导入特定程序进行 Profiling，此文档中的一部分将介绍如何针对 Nuclei Model 完成 Profiling。

Nuclei Model Profiling 的优势：

- 无需使用开发板等硬件
- model 中内建了 gprof 功能，无需 Profiling 库和 gcc -pg 选项就可以产生 Profiling 文件
- 采取了指令级别的采样，可以进行指令级别的 Profiling 分析

在[NucleiStudio_User_Guide.pdf](#)相关章节对 Nuclei Model 如何仿真性能分析配置已经有较详细的描述，此文档以一个例子来展示其实际应用。

NICE/VNICE 自定义指令加速¶

NICE/VNICE使得用户可以结合自己的应用扩展自定义指令，将芯来的标准处理器核扩展成为面向领域专用的处理器，NICE 具体编码规则可以参考 [Nuclei_RISC-V_ISA_Spec.pdf](#) 中的 NICE Introduction。NICE 适用于无需使用 RISCV Vector 的自定义指令，VNICE 适用于需要使用 RISCV Vector 的自定义指令。

[demo_nice/demo_vnlice](#)介绍了 Nuclei 针对 NICE/VNICE 的 demo 应用是如何编译运行的，此文档将通过改造一个更为常见的 AES 加解密的例子，重点说明该如何使用 NICE/VNICE 指令替换热点函数以及如何在 model 里实现 NICE/VNICE 指令，然后通过 Nuclei Studio 的 Profiling 功能分析替换前后的程序性能。

解决方案¶

环境准备¶

Nuclei Studio：[NucleiStudio 2025.10 Windows](#)

Model Profiling¶

工程创建方式有两种：

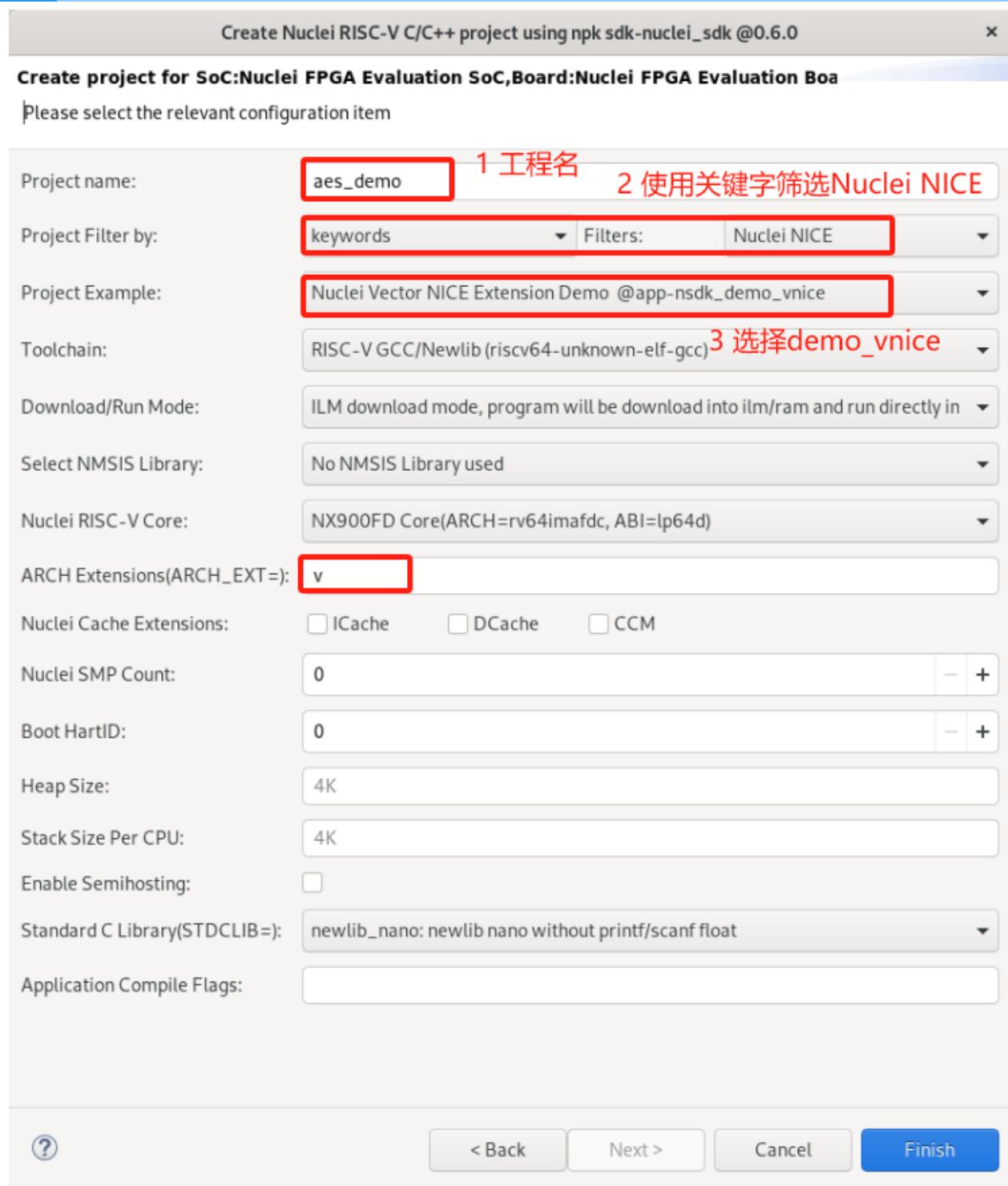
- 方式1：用户可以使用 Nuclei Studio 中的 demo_nice 或 demo_vniced 模板来移植改造自己的 NICE/VNICE 程序
- 方式2：用户导入自己的工程到 Nuclei Studio 中，然后再添加 NICE 内嵌汇编头文件、NICE CSR 使能等代码

此文档将采取前一种方式创建工作，由于此 demo 会用到 VNICE 指令，故创建 demo_vniced 工程，然后将 AES 加解密程序移植替换到其中。

step1：新建 demo_vniced 工程¶

File->New->New Nuclei RISC-V C/C++ Project，选择Nuclei FPGA Evalution Board->sdk-nuclei_sdk @0.6.0

注意：Nuclei SDK 需选择 0.6.0 及以后版本



step2：基于 demo_vnlice 工程移植 aes_demo 裸机用例¶

移植 aes_demo 时，需要保留 demo_vnlice 中的 insn.h 内嵌汇编头文件框架，方便后续添加自定义的 NICE/VNICE 指令，在 main.c 中需要保留 NICE/VNICE 指令执行前的 CSR 使能代码：

```
__RV_CSR_SET(CSR_MSTATUS, MSTATUS_XS);
```

其余 demo_vnive 工程中 application 原始用例可删除，替换成 aes_demo 用例，形成如下目录结构，并确保能够编译通过。

1 移植aes_demo示例

```

46 // Start with known inputs from
47 // https://csrc.nist.gov/CSRC/media/Projects/Cryptographic-Standards-and-Guidelines/documents/examples/AES_Core256.pdf
48 void aes_test(int num_tests)
49 {
50     BENCH_INIT();
51     // Start with known inputs from
52     // https://csrc.nist.gov/CSRC/media/Projects/Cryptographic-Standards-and-Guidelines/documents/examples/AES_Core256.pdf
53     uint8_t key [AES_256_KEY_BYTES] = {
54         0x60, 0x3D, 0xE8, 0x10,
55         0x15, 0xCA, 0x71, 0x80,
56         0x2B, 0x4C, 0x9E, 0xF9,
57         0x85, 0x7D, 0x77, 0x81,
58         0x1F, 0x35, 0x2C, 0x87,
59         0x3B, 0x61, 0x88, 0xD7,
60         0x2D, 0x98, 0x10, 0xA3,
61         0x09, 0x14, 0xDF, 0xF4
62     };
63     uint8_t pt [AES_BLOCK_BYTES] = {
64         0x6B, 0xC1, 0xBE, 0xE2,
65         0x2E, 0x40, 0x9F, 0x96,
66         0xE9, 0x3D, 0x7E, 0x11,
67         0x73, 0x93, 0x17, 0x2A,
68     };
69     uint32_t erk [AES_256_RK_WORDS] : //< Roundkeys (encrypt)
70     uint32_t drk [AES_256_RK_WORDS] : //< Roundkeys (decrypt)
71     uint8_t ct [AES_BLOCK_BYTES];
72     uint8_t pt2 [AES_BLOCK_BYTES];
73
74     for(int i = 0; i < num_tests; i++) {
75         PRINT_DEBUG("AES 256 test %d\n", i + 1, num_tests);
76         for(int l = 0; l < AES_256_RK_WORDS; i++) {
77             erk[i] = 0;
78             drk[i] = 0;
79         }
80     }
81 #if (LOCAL_DEBUG != 0)
82     BENCH_START(aes_256_ecb);
83 #endif
84
85     aes_256_enc_key_schedule(erk, key);
86     aes_256_ecb_encrypt(ct, pt, erk);
87     aes_256_dec_key_schedule(drk, key);
88     aes_256_ecb_decrypt(pt2, ct, drk);
89
90 }
```

22:59:34 Build Finished. 0 errors, 0 warnings. (took 879ms) | 2 保证 demo 编译通过

用户可以下载我们移植好的 AES 加解密 demo：优化前AES工程链接下载

下载 zip 包后，可以直接导入到 Nuclei Studio 中运行(导入步骤：File->Import->Existing Projects into Workspace->Next->Select archive file->选择zip压缩包->next即可)

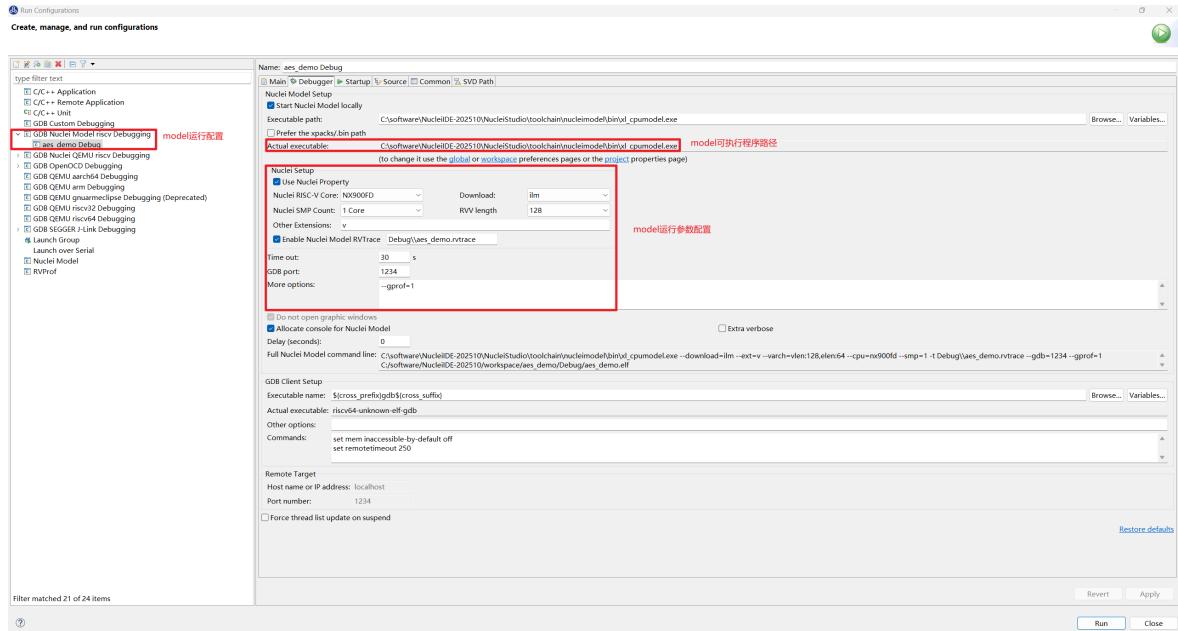
step3：model 仿真程序¶

首先将 aes_debug.h 中的 LOCAL_DEBUG 打开，准备测试 AES 算法的整体 cycle 数。

Model 仿真程序需要配置 Nuclei Studio 中的 GDB Nuclei Model riscv Debugging 配置项，步骤如下：

1. 打开 Nuclei Studio 主菜单栏的 Run 选项的 Run Configurations
2. 选择 GDB Nuclei Model riscv Debugging 配置项，右键选择 New Configuration，会自动生成项目名的 Model 配置页面，launch bar 也会同步更新
3. 在右侧 Main 选项卡中点击 Search Project... 选择编译好的 elf 文件
4. 在右侧 Debugger 选项卡中选择 Browse 找到 Nuclei Model 可执行程序默认路径：NucleiStudio/toolchain/nucleimodel/bin/xl_cpumodel.exe

5. 在右侧 Debugger 选项卡中的 Nuclei Setup 中完成 model 运行配置, 选择 Nuclei RISC-V Core 和 Other Extensions 需要保持和 Nuclei Settings 的 Core 和 Other extensions 配置一致, Other Extensions 为空时不传递此参数, Enable Nuclei Model RVTrace 表示运行时生成 rvtrace, More options 加上 --gprof=1 开启 Profiling 功能, 然后点击 Apply 和 Run, model 就开始运行程序了



Nuclei Studio (< 2025.10) 只能使用 Run Configurations 中的 Nuclei Model 来配置 model, Nuclei Studio (>= 2025.10) 建议切换到使用 GDB Nuclei Model riscv Debugging 来配置

在 Console 中会看到 Total elapsed real time 说明 model 已经完成仿真了, 得到 AES 算法整体消耗 161108 cycle。

```

1 #ifndef __AES_DEBUG_H__
2 #define __AES_DEBUG_H__
3
4 #ifdef __cplusplus
5 extern "C" {
6 #endif
7
8
9 #define LOCAL_DEBUG 1
10 #if LOCAL_DEBUG
11     // 开启打印AES算法整体消耗cycle数
12 #endif
13
14 #endif // __AES_DEBUG_H__

```

ALL RIGHTS RESERVED

[XUMODEL-INFO] Executable segment 0: buf=0x000001ef995dfa10, vaddr=0x80000000, paddr=0x80000000, len=0x4aa8

[XUMODEL-INFO] Data segment 0: buf=0x000001ef995e4dc0, vaddr=0x90000000, paddr=0x90000000, len=0xde0

[XUMODEL-INFO] Executable profiling segments: lowpc=0x80000000, highpc=0x80004aa8

[XUMODEL-INFO] Found the following .elf files:

[XUMODEL-INFO] C:/software/NucleiIDE-202510/workspace/aes_demo/Debug/aes_demo.elf

[XUMODEL-INFO] start pc: 0x80000000

[XUMODEL-INFO] iself=0x80004aa8

[XUMODEL-INFO] load_image start:

[XUMODEL-INFO] addr 0x80000000 len 0x4aa8 ilm <0x80000000 ~ 0x80000000>

[XUMODEL-INFO] addr 0x80000000 len 0x6e0 dlm <0x90000000 ~ 0xa0000000>

[XUMODEL-INFO] load_image over

Nuclei SDK Build Time: Dec 4 2025, 18:23:49

Download Mode: ILM

CPU Frequency 655687 Hz

CPU HartID: 0

benchmark_name : aes

Benchmark initialized

#

AES 256 test 1/1

AES算法整体消耗cycle数

CSV, aes_256_ecb, 101108

test complete!

total run 155931 instruction

[XUMODEL-INFO] Program simulation completed after 0.943418s

[XUMODEL-INFO] Waiting for gprof generating..

Info: /OSCI/System: Simulation stopped by user.

[XUMODEL-INFO] Total elapsed real time: 0.945844s

[XUMODEL-INFO] Press Enter to finish

将 aes_debug.h 中的 LOCAL_DEBUG 关掉去掉程序打印，为了准确测试 Profiling 数据，确保 Nuclei Studio 的 launch bar 为 aes_demo Debug，重新 Run model，运行结束后会生成 Profiling 文件：

```

1 #ifndef __AES_DEBUG_H__
2 #define __AES_DEBUG_H__
3
4 #ifdef __cplusplus
5 extern "C" {
6 #endif
7
8
9 #define LOCAL_DEBUG 0
10 #endif // __AES_DEBUG_H__

```

ALL RIGHTS RESERVED

[XUMODEL-INFO] filename: C:/software/NucleiIDE-202510/workspace/aes_demo/Debug/aes_demo.elf

[XUMODEL-INFO] expected real execution time: 30

[XUMODEL-INFO] Elf parsing start:

[XUMODEL-INFO] Executable segment 0: buf=0x000002b505d5fa60, vaddr=0x80000000, paddr=0x80000000, len=0x658

[XUMODEL-INFO] Data segment 0: buf=0x000002b505d44400, vaddr=0x90000000, paddr=0x90000000, len=0x658

[XUMODEL-INFO] Executable profiling segments: lowpc=0x80000000, highpc=0x800049d0

[XUMODEL-INFO] Found the following .elf files:

[XUMODEL-INFO] C:/software/NucleiIDE-202510/workspace/aes_demo/Debug/aes_demo.elf

[XUMODEL-INFO] start pc: 0x80000000

[XUMODEL-INFO] iself=0x800049d0

[XUMODEL-INFO] load_image start:

[XUMODEL-INFO] addr 0x80000000 len 0x49d0 ilm <0x80000000 ~ 0x80000000>

[XUMODEL-INFO] addr 0x80000000 len 0x658 dlm <0x90000000 ~ 0xa0000000>

[XUMODEL-INFO] load_image over

Nuclei SDK Build Time: Dec 4 2025, 18:23:49

Download Mode: ILM

CPU Frequency 655687 Hz

CPU HartID: 0

Benchmark initialized

total run 146036 instruction

[XUMODEL-INFO] Program simulation completed after 0.917593s

profiling文件生成提示

Info: /OSCI/System: Simulation stopped by user.

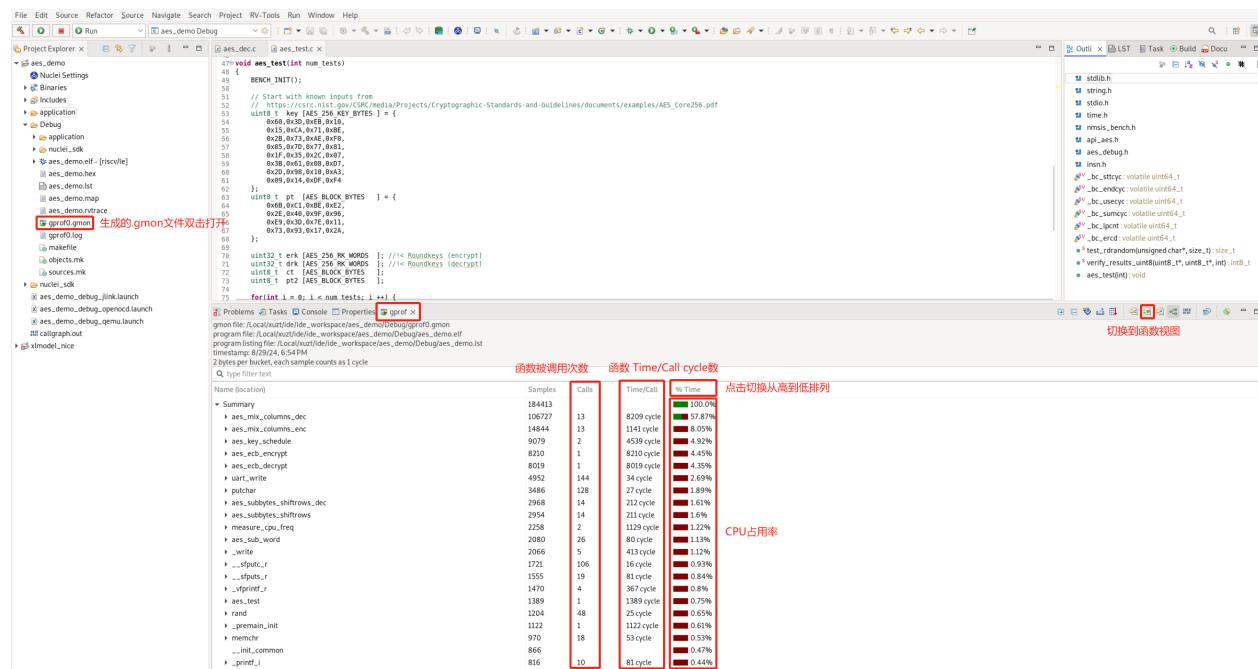
[XUMODEL-INFO] Total elapsed real time: 0.928176s

[XUMODEL-INFO] Press Enter to finish

step4：解析 gprof 数据¶

model 仿真程序完成后，双击打开生成的 `gprof*.gmon` 文件，切换到函数视图，点击 % Time 从高到低排列函数 CPU 占用率。

注意：Time/Call 显示的是每个函数的函数体 text 段的 cycle 数，并不是整个函数的 cycle 数，是不计入其中子函数占用的 cycle 数的。



从而得到 CPU 占用率最高的 TOP5 热点函数为：

```
aes_mix_columns_dec
aes_mix_columns_enc
aes_key_schedule
aes_ecb_decrypt
aes_ecb_encrypt
```

注意：此时需要备份当前的 `aes_demo` 工程，改名为 `aes_demo_nice` 工程，这样可以在 Nuclei Studio 中同时打开两个工程，方便添加 NICE/VNICE 指令优化后的工程和原 `aes_demo` 工程进行 Profiling 比较。

step5：NICE/VNICE 指令替换¶

用户需要在备份的 `aes_demo_nice` 工程下，研究热点函数算法特点，将其替换为 NICE/VNICE 指令，从而提升整体程序性能。

在包含 AES 加解密的 TOP5 热点函数的 `aes_dec.c` 和 `aes_dec.c` 两个C文件中 `#include "insn.h"` 以便添加 NICE/VNICE 指令替换。

TOP1 热点函数为 `aes_mix_columns_dec`, 实现了 AES 算法解密的逆混合列, 输入一个状态矩阵, 经过计算后原地址输出一个计算后的状态矩阵, 实现了 Load 数据、逆混合运算以及 Store 数据, 代码如下:

```
static void aes_mix_columns_dec(
    uint8_t      pt[16]          //!< Current block state
){
    // Col 0
    for(int i = 0; i < 4; i++) {
        uint8_t b0,b1,b2,b3;
        uint8_t s0,s1,s2,s3;

        s0 = pt[4*i+0];
        s1 = pt[4*i+1];
        s2 = pt[4*i+2];
        s3 = pt[4*i+3];

        b0 = XTE(s0) ^ XTB(s1) ^ XTD(s2) ^ XT9(s3);
        b1 = XT9(s0) ^ XTE(s1) ^ XTB(s2) ^ XTD(s3);
        b2 = XTD(s0) ^ XT9(s1) ^ XTE(s2) ^ XTB(s3);
        b3 = XTB(s0) ^ XTD(s1) ^ XT9(s2) ^ XTE(s3);

        pt[4*i+0] = b0;
        pt[4*i+1] = b1;
        pt[4*i+2] = b2;
        pt[4*i+3] = b3;
    }
}
```

由于输入输出地址一样, 可以考虑用一条 NICE 指令替换, 指令的 `opcode`、`funct3` 和 `funct7` 都可以在编码位域中自定义, 该指令设置 `opcode` 为 `Custom-0`, `funct3` 设置为 0, `funct7` 设置为 `0x10`, 寄存器只使用到 `rs1` 描述入参地址, 不需要使用 `rd` 和 `rs2`, 指令写到 `insn.h` 中, 内嵌汇编如下:

```
_STATIC_FORCEINLINE void custom_aes_mix_columns_dec(uint8_t* addr)
{
    int zero = 0;
    asm volatile(".insn r 0xb, 0, 0x10, x0, %1, x0" : "=r"(zero) :
    "r"(addr));
}
```

用户可以在 `insn.h` 中定义一个 `USE_NICE` 的宏选择是否使用 NICE , 在 `aes_dec.c` 改写 `aes_mix_columns_dec` 如下 :

```

static void aes_mix_columns_dec(
    uint8_t      pt[16]           //!< Current block state
){

#ifndef USE_NICE
    custom_aes_mix_columns_dec(pt);
#else
    // Col 0
    for(int i = 0; i < 4; i++) {
        uint8_t b0,b1,b2,b3;
        uint8_t s0,s1,s2,s3;

        s0 = pt[4*i+0];
        s1 = pt[4*i+1];
        s2 = pt[4*i+2];
        s3 = pt[4*i+3];

        b0 = XTE(s0) ^ XTB(s1) ^ XTD(s2) ^ XT9(s3);
        b1 = XT9(s0) ^ XTE(s1) ^ XTB(s2) ^ XTD(s3);
        b2 = XTD(s0) ^ XT9(s1) ^ XTE(s2) ^ XTB(s3);
        b3 = XTB(s0) ^ XTD(s1) ^ XT9(s2) ^ XTE(s3);

        pt[4*i+0] = b0;
        pt[4*i+1] = b1;
        pt[4*i+2] = b2;
        pt[4*i+3] = b3;
    }
#endif
}

```

TOP2 热点函数为 `aes_mix_columns_enc`, 和 TOP1 类似, 实现的是 AES 加密的逆混合列, 同样也是输入一个状态矩阵, 经过计算后原地址输出一个计算后的状态矩阵 :

```

static void aes_mix_columns_enc(
    uint8_t      ct [16]           //!< Current block state
){
    for(int i = 0; i < 4; i++) {
        uint8_t b0,b1,b2,b3;
        uint8_t s0,s1,s2,s3;

        s0 = ct[4*i+0];
        s1 = ct[4*i+1];
        s2 = ct[4*i+2];
        s3 = ct[4*i+3];
    }
}

```

```

    b0 = XT2(s0) ^ XT3(s1) ^ (s2) ^ (s3);
    b1 = (s0) ^ XT2(s1) ^ XT3(s2) ^ (s3);
    b2 = (s0) ^ (s1) ^ XT2(s2) ^ XT3(s3);
    b3 = XT3(s0) ^ (s1) ^ (s2) ^ XT2(s3);

    ct[4*i+0] = b0;
    ct[4*i+1] = b1;
    ct[4*i+2] = b2;
    ct[4*i+3] = b3;
}
}

```

考虑到指令实现可能无法只用1条指令完成，可使用2条 VNICE 指令替换此算法，第一条 load 16 byte 数据到 Vector 寄存器，第二条再完成计算以及 store。

指令的 `opcode`、`funct3` 和 `funct7` 仍然可以在编码位域中自定义，第一条指令使用 `rd` 描述 Vector 寄存器，`rs1` 描述入参地址，第二条指令使用 `rs1` 描述入参地址，`rs1` 描述入参 Vector 寄存器，两条 VNICE 指令的内嵌汇编写到 `insn.h` 中，定义如下：

```

__STATIC_FORCEINLINE vint8m1_t __custom_vnice_load_v_i8m1
(uint8_t* addr)
{
    vint8m1_t rdata ;
    asm volatile(".insn r 0xb,4,0,%0,%1,x0"
               : "=vr"(rdata)
               : "r"(addr)
               );
    return rdata;
}

__STATIC_FORCEINLINE void __custom_vnice_aes_mix_columns_enc_i8m1
(uint8_t *addr, vint8m1_t data)
{
    int zero = 0;
    asm volatile(".insn r 0xb,4,1,x0,%1,%2"
               : "=r"(zero)
               : "r"(addr)
               , "vr"(data)
               );
}

```

用户通过定义 Vector 寄存器以及使用上定义好的 VNICE 指令内嵌汇编改写 `aes_enc.c` 中的 `aes_mix_columns_enc` 如下：

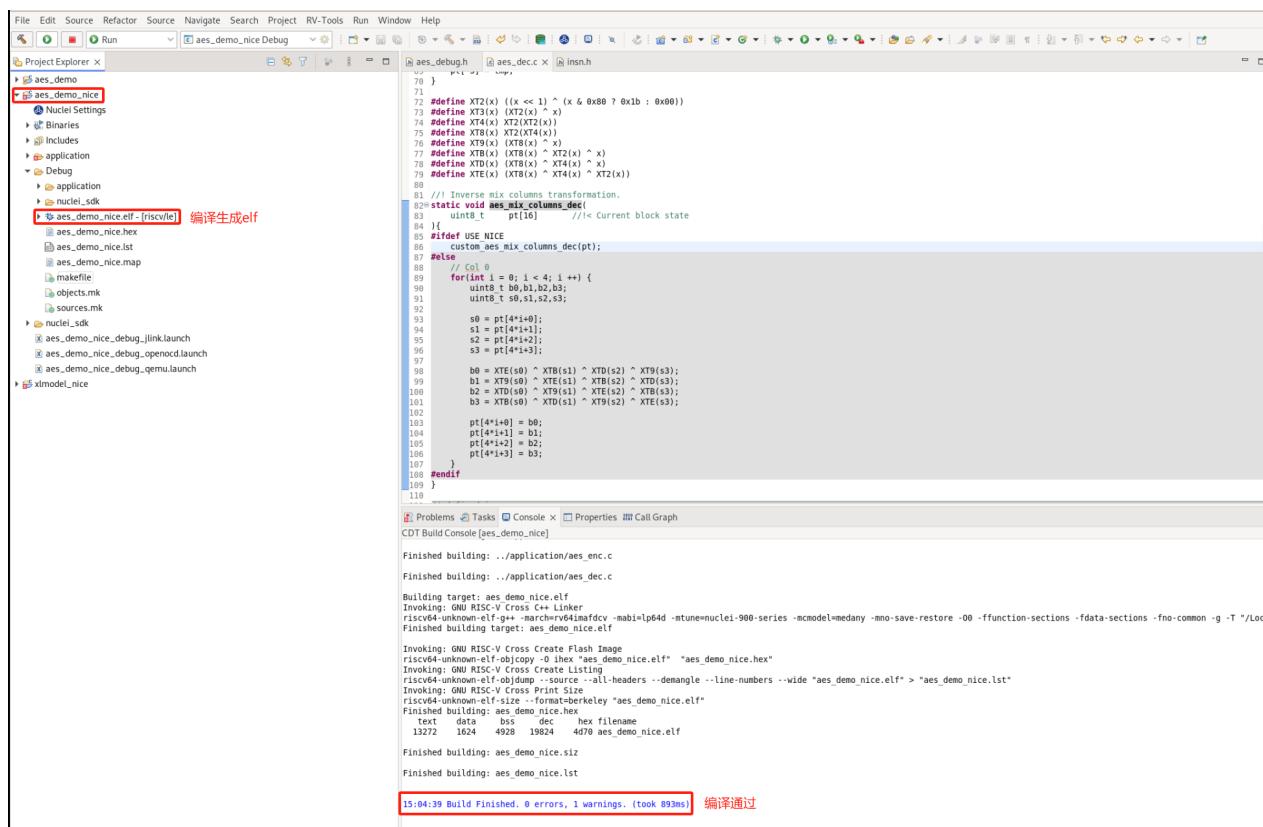
```
static void aes_mix_columns_enc(
    uint8_t      ct [16]           //!< Current block state
){
#ifdef USE_NICE
    uint32_t blkCnt = 16;
    size_t l;
    vint8m1_t vin;
    for (; (l = __riscv_vsetvl_e8m1(blkCnt)) > 0; blkCnt -= l) {
        vin = __custom_vnlice_load_v_i8m1(ct);
        __custom_vnlice_aes_mix_columns_enc_i8m1(ct, vin);
    }
#else
    for(int i = 0; i < 4; i++) {
        uint8_t b0,b1,b2,b3;
        uint8_t s0,s1,s2,s3;

        s0 = ct[4*i+0];
        s1 = ct[4*i+1];
        s2 = ct[4*i+2];
        s3 = ct[4*i+3];

        b0 = XT2(s0) ^ XT3(s1) ^ (s2) ^ (s3);
        b1 = (s0) ^ XT2(s1) ^ XT3(s2) ^ (s3);
        b2 = (s0) ^ (s1) ^ XT2(s2) ^ XT3(s3);
        b3 = XT3(s0) ^ (s1) ^ (s2) ^ XT2(s3);

        ct[4*i+0] = b0;
        ct[4*i+1] = b1;
        ct[4*i+2] = b2;
        ct[4*i+3] = b3;
    }
#endif
}
```

修改后的程序代码编译通过：(aes_demo_nice 工程)



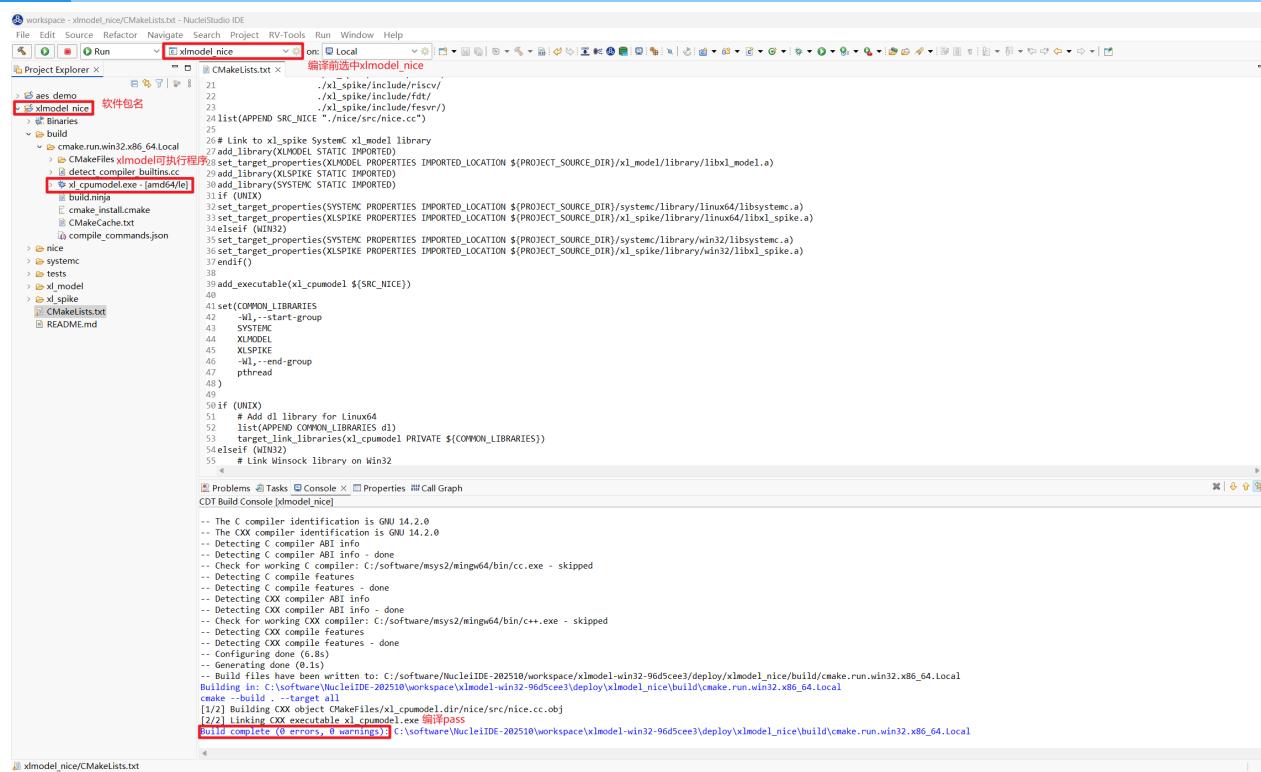
step6：在 Nuclei Model 中实现 NICE/VNICE 指令

首先需要下载支持用户配置自定义 NICE/VNICE 指令的原始 Nuclei Model 软件包[原始model软件包下载](#)，解压软件包为 `xlmodel_nice`，然后将其导入 Nuclei Studio。

导入步骤：File->Import->Projects from Folder or Archive->Next->Directory->选择 xlmodel nice->Finish即可

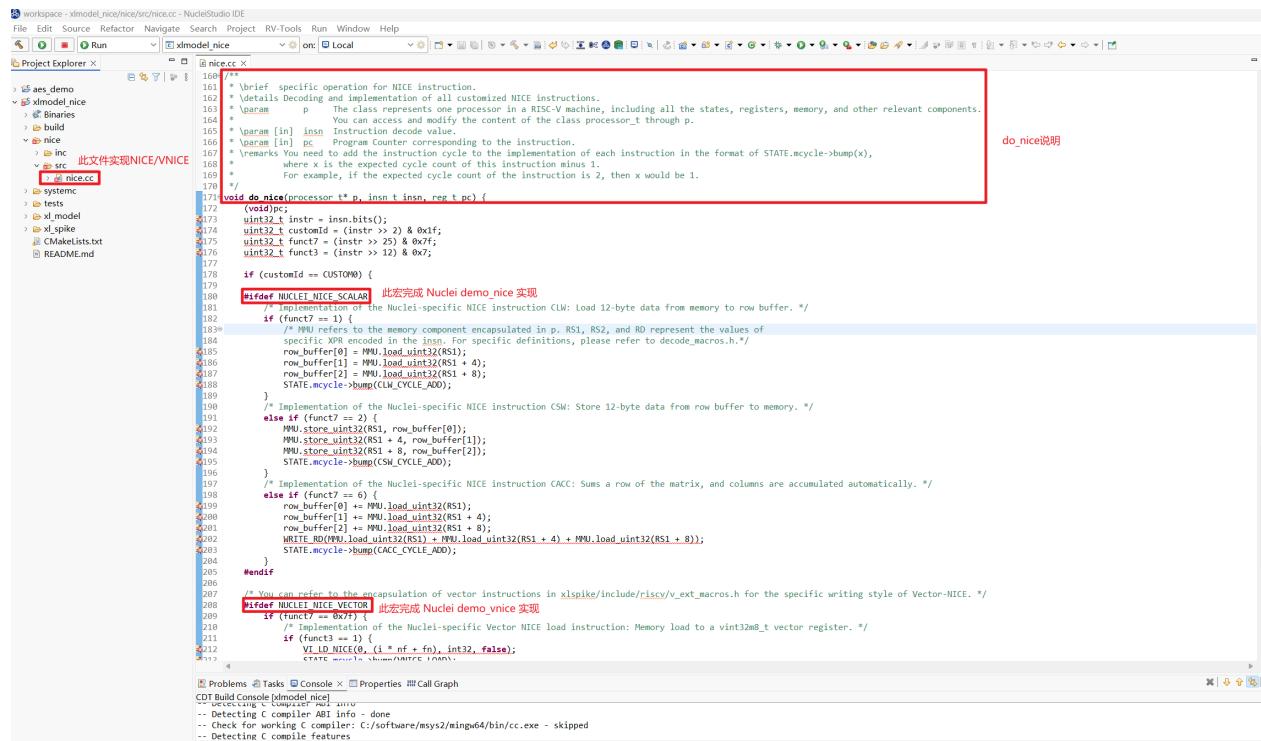
如何使用 Nuclei Model 以及查看 `xlmodel_nice` 软件包的目录结构可以参考[Nuclei Model介绍](#), `xlmodel_nice` 是由CMake构建的, 用户无需修改即可编译, 在 编译前选择 Nuclei Studio 的 launch bar 的 `xlmodel nice`, 然后点击编译, 确保软件包本身编译通过：

Nuclei Studio (< 2025.10) 生成的 elf 文件所在路径为 build/default/xl_cpumodel



打开 `nice.cc` 文件，用户需要用该文件的 `do_nice` 函数实现所有自定义的 NICE/VNICE 指令，当前 `do_nice` 里包含了针对 `demo_nice` 或 `demo_vniced` 的 Nuclei 定义的 NICE/VNICE 指令，用户可以参考其中注释完成自己的自定义指令。

注意：当用户编写自定义 NICE/VNICE 指令时，需要关掉和 Nuclei demo_nice/demo_vnice 对应的 NUCLEI_NICE_SCALAR/NUCLEI_NICE_VECTOR 宏，以免和用户自定义的指令编码相冲突。



AES demo 中定义的 NICE/VNICE 指令实现如下图，通过指令的 `opcode`、`funct3` 和 `funct7` 编写条件判断语句指定该条指令，然后在其中实现指令行为以及指令 `cycle` 数添加。

NICE 指令实现中，MMU 宏表示 memory 访问，load memory 使用 `MMU.load_uint<n>`，store memory 使用 `MMU.store_uint<n>`，RD、RS1、RS2、RS3 宏表示其对应标量寄存器中的值，FRS1、FRS2、FRS3 宏表示其对应浮点寄存器中的值，这些宏的使用可以参考 `nice/inc/decode_macros.h`。

VNICE 指令实现中仍然是用 MMU 宏访问 memory，只不过 Vector 寄存器数据会存储在 `P.VU_elt` 类中，用户可以参考 `xlspike/include/riscv/v_ext_macros.h` 完成相关代码编写。

在指令实现完后，将自定义指令需要的 `cycle` 数 `n` 直接标定：`STATE.mcycle->bump(n)`；即可，这里根据硬件通过 NICE/VNICE 实现此算法的理论值，标定 `custom_aes_mix_columns_dec` 为 7 cycle，`_custom_vnive_load_v_i8m1` 为 1 cycle，`_custom_vnive_aes_mix_columns_enc_i8m1` 为 2 cycle。

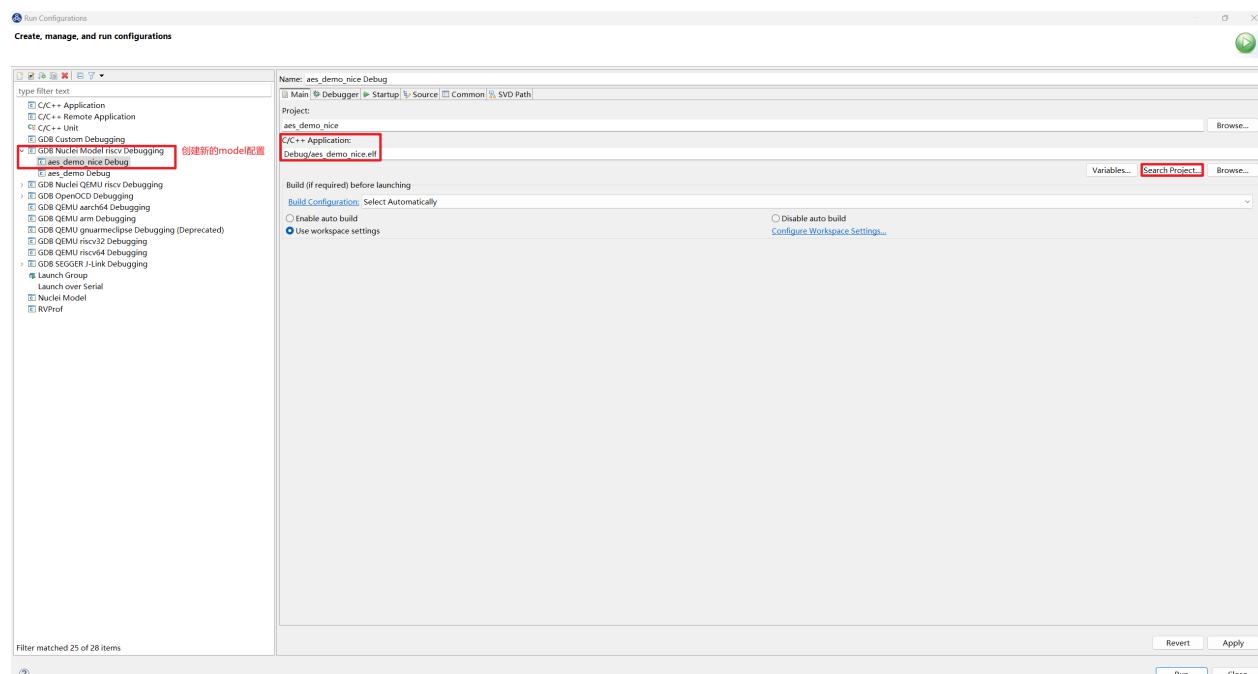
```

201 void do_nice(processor_t* p, insn_t insn, reg_t pc) {
202     (void)p;
203     uint32_t instr = insn.bits();
204     uint32_t custoId = (instr >> 2) & 0x1f;
205     uint32_t funct7 = (instr >> 25) & 0x7f;
206     uint32_t funct3 = (instr >> 12) & 0x7;
207
208     if (custoId == CUSTOM0) {
209         if ((funct3 == 0) && (funct7 & 0x1f) == 0x10) {
210             uint8_t b0, b1, b2, b3;
211             uint8_t s0, s1, s2, s3;
212             for (int i = 0; i < 4; i++) {
213                 b0 = MMU.load_uint8(4 * i + RS1);
214                 s1 = MMU.load_uint8(4 * i + 1 + RS1);
215                 s2 = MMU.load_uint8(4 * i + 2 + RS1);
216                 s3 = MMU.load_uint8(4 * i + 3 + RS1);
217
218                 b0 = XTE(s0) ^ XTD(s1) ^ XTD(s2) ^ XTD(s3);
219                 b1 = XTE(s0) ^ XTE(s1) ^ XTB(s2) ^ XTD(s3);
220                 b2 = XTB(s0) ^ XTE(s1) ^ XTE(s2) ^ XTB(s3);
221                 b3 = XTB(s0) ^ XTD(s1) ^ XTE(s2) ^ XTE(s3);
222
223                 MMU.store_uint8(4 * i + RS1, b0);
224                 MMU.store_uint8(4 * i + 1 + RS1, b1);
225                 MMU.store_uint8(4 * i + 2 + RS1, b2);
226                 MMU.store_uint8(4 * i + 3 + RS1, b3);
227             }
228         }
229         STATE.mcycle->bump(AES_MIX_COLUMNS_DEC);
230     } else if ((funct3 == 4) && (funct7 & 0x1f) == 0x0) {
231         V_AES_MX_COLUMNS_ENC_CAC(0, (1 * nf + fn), uint8_t, false);
232         STATE.mcycle->bump(AES_MIX_COLUMNS_ENC_VLOAD0);
233     } else if ((funct3 == 4) && (funct7 & 0x1f) == 0x1) {
234         V_AES_MX_COLUMNS_ENC_CAC(0, (1 * nf + fn), uint8_t, false);
235         STATE.mcycle->bump(AES_MIX_COLUMNS_ENC_CAC);
236     }
237 }
238
239 #ifndef NUCLEI_NICE_SCALAR
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
729
730
731
732
733
734
735
736
737
738
739
739
740
741
742
743
744
745
746
747
748
749
749
750
751
752
753
754
755
756
757
758
759
759
760
761
762
763
764
765
766
767
768
769
769
770
771
772
773
774
775
776
777
778
779
779
780
781
782
783
784
785
786
787
788
789
789
790
791
792
793
794
795
796
797
798
799
799
800
801
802
803
804
805
806
807
808
809
809
810
811
812
813
814
815
816
817
818
819
819
820
821
822
823
824
825
826
827
828
829
829
830
831
832
833
834
835
836
837
838
839
839
840
841
842
843
844
845
846
847
848
849
849
850
851
852
853
854
855
856
857
858
859
859
860
861
862
863
864
865
866
867
868
869
869
870
871
872
873
874
875
876
877
878
879
879
880
881
882
883
884
885
886
887
887
888
889
889
890
891
892
893
894
895
896
897
898
899
899
900
901
902
903
904
905
906
907
908
909
909
910
911
912
913
914
915
916
917
918
919
919
920
921
922
923
924
925
926
927
928
929
929
930
931
932
933
934
935
936
937
938
939
939
940
941
942
943
944
945
946
947
948
949
949
950
951
952
953
954
955
956
957
958
959
959
960
961
962
963
964
965
966
967
968
969
969
970
971
972
973
974
975
976
977
978
979
979
980
981
982
983
984
985
986
987
987
988
989
989
990
991
992
993
994
995
996
997
998
999
999
1000
1001
1002
1003
1004
1005
1006
1007
1008
1009
1009
1010
1011
1012
1013
1014
1015
1016
1017
1018
1019
1019
1020
1021
1022
1023
1024
1025
1026
1027
1028
1029
1029
1030
1031
1032
1033
1034
1035
1036
1037
1038
1039
1039
1040
1041
1042
1043
1044
1045
1046
1047
1048
1049
1049
1050
1051
1052
1053
1054
1055
1056
1057
1058
1059
1059
1060
1061
1062
1063
1064
1065
1066
1067
1068
1069
1069
1070
1071
1072
1073
1074
1075
1076
1077
1078
1079
1079
1080
1081
1082
1083
1084
1085
1086
1087
1087
1088
1089
1089
1090
1091
1092
1093
1094
1095
1096
1097
1097
1098
1099
1099
1100
1101
1102
1103
1104
1105
1106
1107
1108
1109
1109
1110
1111
1112
1113
1114
1115
1116
1117
1118
1119
1119
1120
1121
1122
1123
1124
1125
1126
1127
1128
1129
1129
1130
1131
1132
1133
1134
1135
1136
1137
1138
1139
1139
1140
1141
1142
1143
1144
1145
1146
1147
1148
1149
1149
1150
1151
1152
1153
1154
1155
1156
1157
1158
1159
1159
1160
1161
1162
1163
1164
1165
1166
1167
1168
1169
1169
1170
1171
1172
1173
1174
1175
1176
1177
1178
1179
1179
1180
1181
1182
1183
1184
1185
1186
1187
1187
1188
1189
1189
1190
1191
1192
1193
1194
1195
1196
1197
1197
1198
1199
1199
1200
1201
1202
1203
1204
1205
1206
1207
1208
1209
1209
1210
1211
1212
1213
1214
1215
1216
1217
1218
1219
1219
1220
1221
1222
1223
1224
1225
1226
1227
1228
1229
1229
1230
1231
1232
1233
1234
1235
1236
1237
1238
1239
1239
1240
1241
1242
1243
1244
1245
1246
1247
1248
1249
1249
1250
1251
1252
1253
1254
1255
1256
1257
1258
1259
1259
1260
1261
1262
1263
1264
1265
1266
1267
1268
1269
1269
1270
1271
1272
1273
1274
1275
1276
1277
1278
1278
1279
1280
1281
1282
1283
1284
1285
1286
1287
1287
1288
1289
1289
1290
1291
1292
1293
1294
1295
1296
1297
1297
1298
1299
1299
1300
1301
1302
1303
1304
1305
1306
1307
1308
1309
1309
1310
1311
1312
1313
1314
1315
1316
1317
1318
1319
1319
1320
1321
1322
1323
1324
1325
1326
1327
1328
1329
1329
1330
1331
1332
1333
1334
1335
1336
1337
1338
1339
1339
1340
1341
1342
1343
1344
1345
1346
1347
1348
1349
1349
1350
1351
1352
1353
1354
1355
1356
1357
1358
1359
1359
1360
1361
1362
1363
1364
1365
1366
1367
1368
1369
1369
1370
1371
1372
1373
1374
1375
1376
1377
1378
1378
1379
1380
1381
1382
1383
1384
1385
1386
1387
1387
1388
1389
1389
1390
1391
1392
1393
1394
1395
1396
1397
1397
1398
1399
1399
1400
1401
1402
1403
1404
1405
1406
1407
1408
1409
1409
1410
1411
1412
1413
1414
1415
1416
1417
1418
1419
1419
1420
1421
1422
1423
1424
1425
1426
1427
1428
1429
1429
1430
1431
1432
1433
1434
1435
1436
1437
1438
1438
1439
1440
1441
1442
1443
1444
1445
1446
1447
1448
1448
1449
1450
1451
1452
1453
1454
1455
1456
1457
1458
1458
1459
1460
1461
1462
1463
1464
1465
1466
1467
1468
1469
1469
1470
1471
1472
1473
1474
1475
1476
1477
1478
1478
1479
1480
1481
1482
1483
1484
1485
1486
1487
1487
1488
1489
1489
1490
1491
1492
1493
1494
1495
1496
1497
1497
1498
1499
1499
1500
1501
1502
1503
1504
1505
1506
1507
1508
1509
1509
1510
1511
1512
1513
1514
1515
1516
1517
1518
1519
1519
1520
1521
1522
1523
1524
1525
1526
1527
1528
1529
1529
1530
1531
1532
1533
1534
1535
1536
1537
1538
1538
1539
1540
1541
1542
1543
1544
1545
1546
1547
1548
1548
1549
1550
1551
1552
1553
1554
1555
1556
1557
1558
1558
1559
1560
1561
1562
1563
1564
1565
1566
1567
1568
1568
1569
1570
1571
1572
1573
1574
1575
1576
1577
1578
1578
1579
1580
1581
1582
1583
1584
1585
1586
1587
1587
1588
1589
1589
1590
1591
1592
1593
1594
1595
1596
1597
1597
1598
1599
1599
1600
1601
1602
1603
1604
1605
1606
1607
1608
1609
1609
1610
1611
1612
1613
1614
1615
1616
1617
1618
1618
1619
1620
1621
1622
1623
1624
1625
1626
1627
1628
1629
1629
1630
1631
1632
1633
1634
1635
1636
1637
1638
1638
1639
1640
1641
1642
1643
1644
1645
1646
1647
1648
1648
1649
1650
1651
1652
1653
1654
1655
1656
1657
1658
1658
1659
1660
1661
1662
1663
1664
1665
1666
1667
1668
1668
1669
1670
1671
1672
1673
1674
1675
1676
1677
1678
1678
1679
1680
1681
1682
1683
1684
1685
1686
1687
1687
1688
1689
1689
1690
1691
1692
1693
1694
1695
1696
1697
1697
1698
1699
1699
1700
1701
1702
1703
1704
1705
1706
1707
1708
1709
1709
1710
1711
1712
1713
1714
1715
1716
1717
1718
1718
1719
1720
1721
1722
1723
1724
1725
1726
1727
1728
1729
1729
1730
1731
1732
1733
1734
1735
1736
1737
1738
1738
1739
1740
1741
1742
1743
1744
1745
1746
1747
1748
1748
1749
1750
1751
1752
1753
1754
1755
1756
1757
1758
1758
1759
1760
1761
1762
1763
1764
1765
1766
1767
1768
1768
1769
1770
1771
1772
1773
1774
1775
1776
1777
1778
1778
1779
1780
1781
1782
1783
1784
1785
1786
1787
1787
1788
1789
1789
1790
1791
1792
1793
1794
1795
1796
1797
1797
1798
1799
1799
1800
1801
1802
1803
1804
1805
1806
1807
1808
1809
1809
1810
1811
1812
1813
1814
1815
1816
1817
1818
1818
1819
1820
1821
1822
1823
1824
1825
1826
1827
1828
1829
1829
1830
1831
1832
1833
1834
1835
1836
1837
1838
1838
1839
1840
1841
1842
1843
1844
1845
1846
1847
1848
1848
1849
1850
1851
1852
1853
1854
1855
1856
1857
1858
1858
1859
1860
1861
1862
1863
1864
1865
1866
1867
1868
1868
1869
1870
1871
1872
1873
1874
1875
1876
1877
1877
1878
1879
1880
1881
1882
1883
1884
1885
1886
1887
1887
1888
1889
1889
1890
1891
1892
1893
1894
1895
1896
1897
1897
1898
1899
1899
1900
1901
1902
1903
1904
1905
1906
1907
1908
1909
1909
1910
1911
1912
1913
1914
1915
1916
1917
1918
1918
1919
1920
1921
1922
1923
1924
1925
1926
1927
1928
1929
1929
1930
1931
1932
1933
1934
1935
1936
1937
1938
1938
1939
1940
1941
1942
1943
1944
1945
1946
1947
1948
1948
1949
1950
1951
1952
1953
1954
1955
1956
1957
1958
1959
1959
1960
1961
1962
1963
1964
1965
1966
1967
1968
1968
1969
1970
1971
1972
1973
1974
1975
1976
1977
1977
1978
1979
1980
1981
1982
1983
1984
1985
1986
1987
1987
1988
1989
1989
1990
1991
1992
1993
1994
1995
1996
1997
1997
1998
1999
1999
2000
2001
2002
2003
2004
2005
2006
2007
2008
2009
2009
2010
2011
2012
2013
2014
2015
2016
2017
2018
2018
2019
2020
2021
2022
2023
2024
2025
2026
2027
2028
2029
2029
2030
2031
2032
2033
2034
2035
2036
2037
2038
2038
2039
2040
2041
2042
2043
2044
2045
2046
2047
2048
2048
2049
2050
2051
2052
2053
2054
2055
2056
2057
2058
2059
2059
2060
2061
2062
2063
2064
2065
2066
2067
2068
2068
2069
2070
2071
2072
2073
2074
2075
2076
2077
2078
2078
2079
2080
2081
2082
2083
2084
2085
2086
2087
2087
2088
2089
2089
2090
2091
2092
2093
2094
2095
2096
2097
2097
2098
2099
2099
2100
2101
2102
2103
2104
2105
2106
2107
2108
2109
2109
2110
2111
2112
2113
2114
2115
2116
2117
2118
2118
2119
2120
2121
2122
2123
2124
2125
2126
2127
2128
2129
2129
2130
2131
2132
2133
2134
2135
2136
2137
2138
2138
2139
2140
2141
2142
2143
2144
2145
2146
2147
2148
2148
2149
2150
2151
2152
2153
2154
2155
2156
2157
2158
2159
2159
216
```

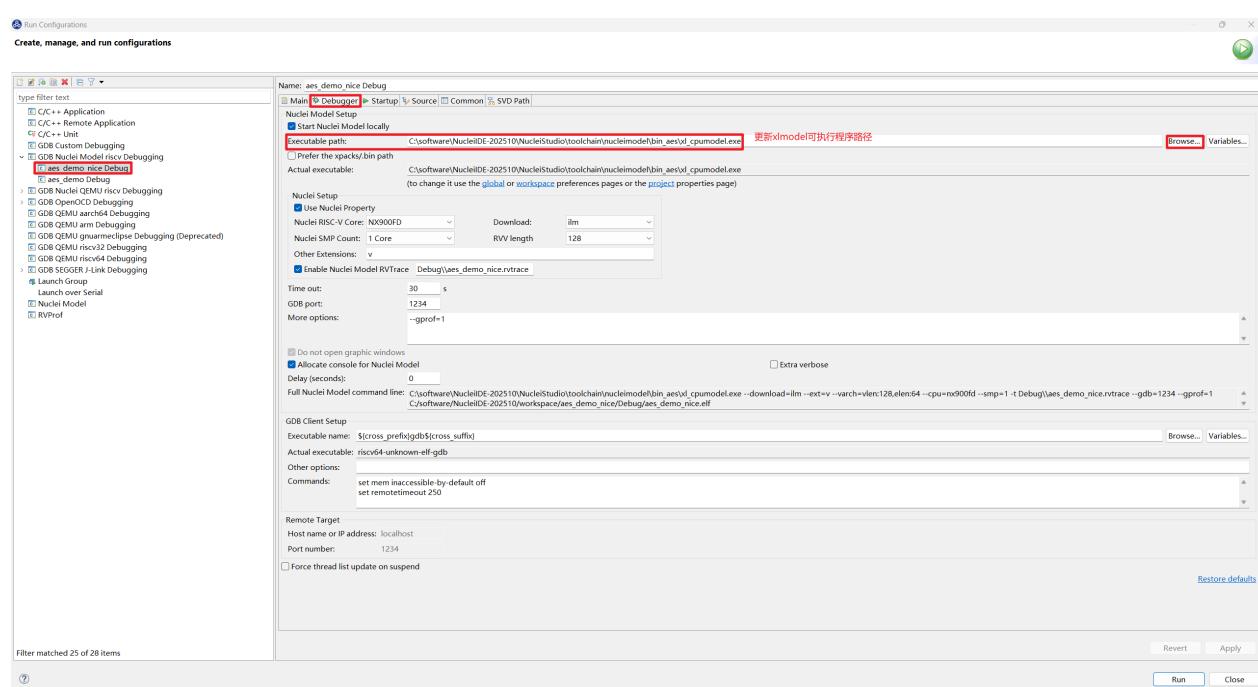
step7：热点函数再分析¶

注意：请务必完成 step6 中介绍的实现了 NICE/VNICE 指令的 model 导入 Nuclei Studio 中才能用 model Run aes_demo_nice 工程。

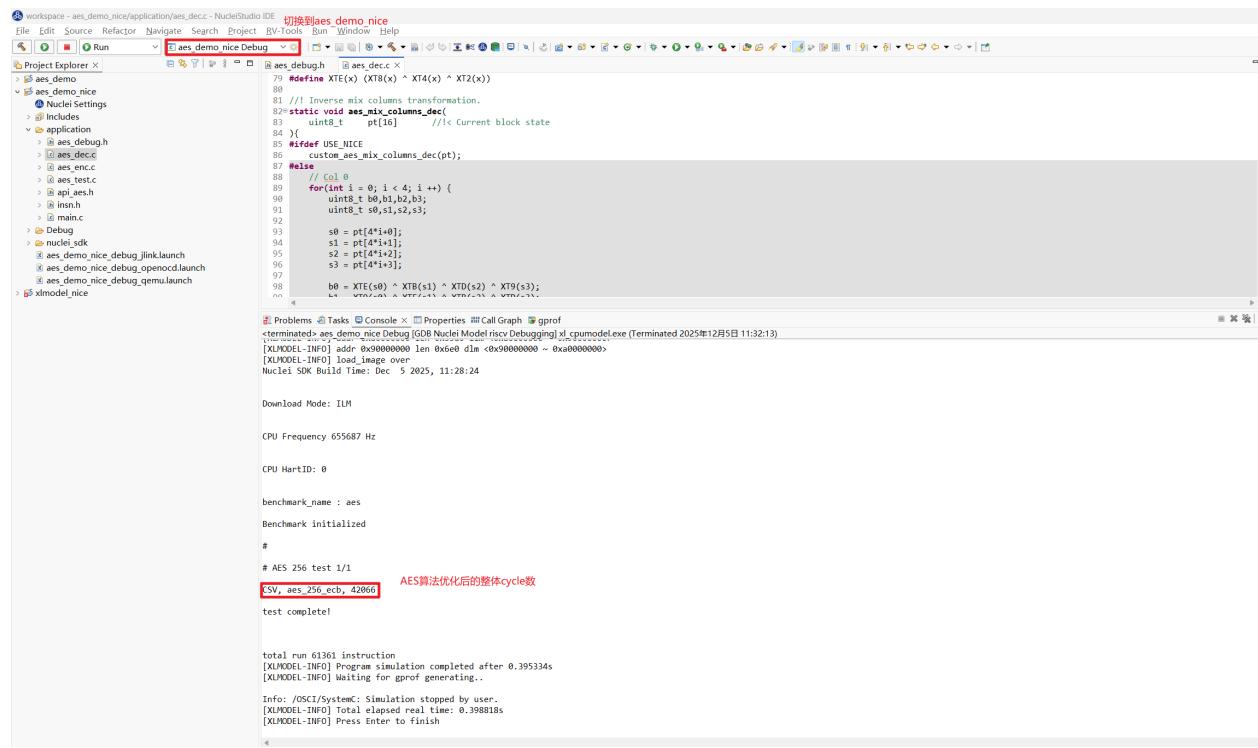
首先打开 Nuclei Studio 主菜单栏的 Run 选项的 Run Configurations, model 配置需要重新添加新的 GDB Nuclei Model riscv Debugging 运行配置 aes_demo_nice Debug, 在 Main 选项卡中选择 aes_demo_nice.elf :



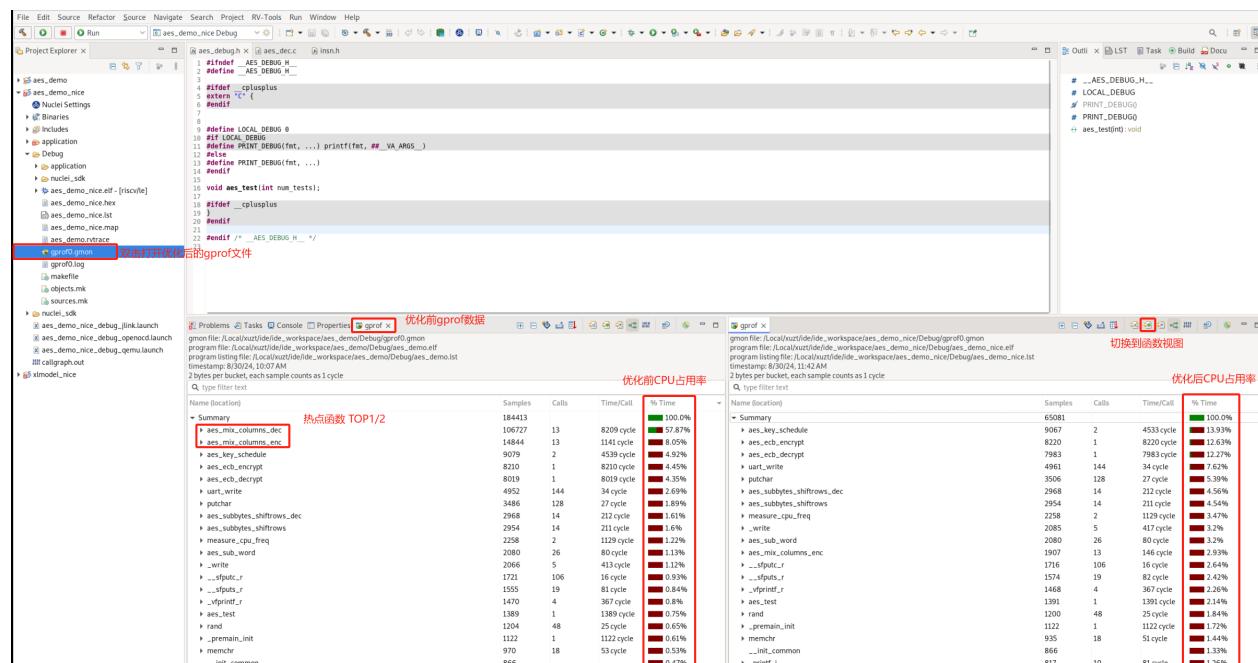
然后在 Debugger 选项卡中的 model 执行路径 Executable path 改为 step6 中新修改 model 的执行路径：.../NucleiStudio/toolchain/nucleimodel/bin_aes/xl_cpumodel:



运行前将 `aes_debug.h` 中的 `LOCAL_DEBUG` 打开，测试优化后 AES 算法的整体 cycle 数，选择 Nuclei Studio 的 launch bar 的 `aes_demo_nice Debug` 后 Run model，得到 AES 算法优化后整体消耗 cycle 数从优化前的 161108 降到了 42066 cycle。

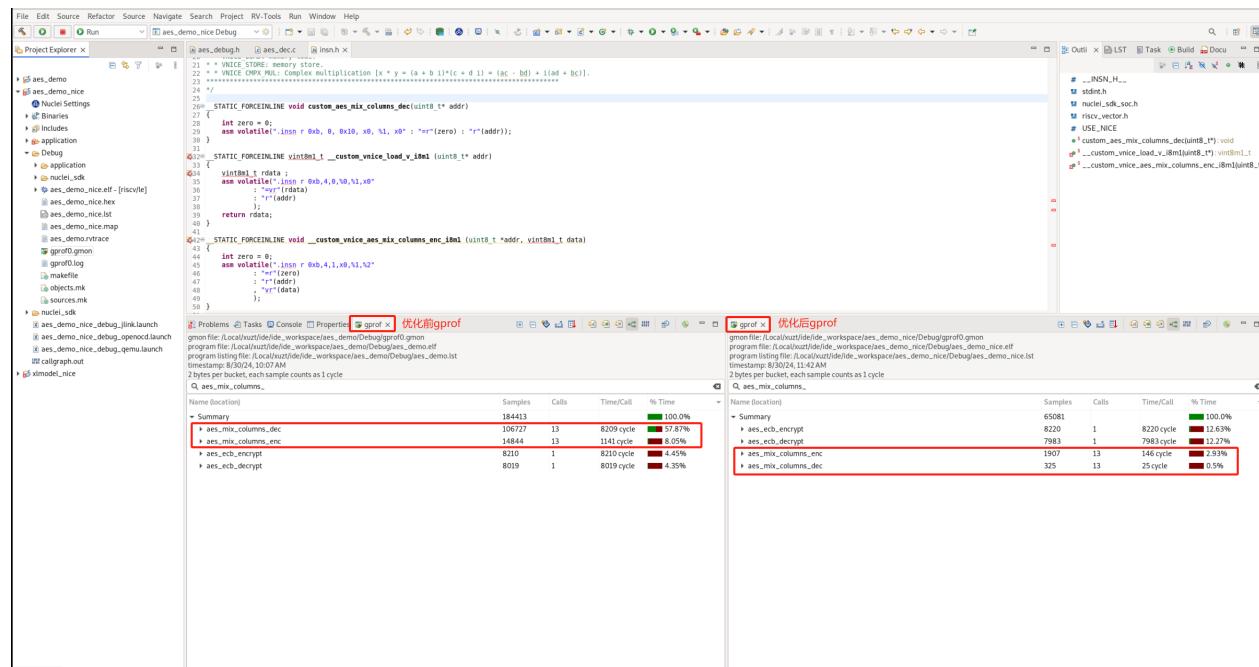


将 `aes_debug.h` 中的 `LOCAL_DEBUG` 关掉测试重新 Run model 测试 Profiling 数据，双击 `gprof0.gmon` 可以看到 CPU 占用率较高的热点函数已经没有 `aes_mix_columns_enc` 和 `aes_mix_columns_dec` 了：



搜索 aes_mix_columns_enc 和 aes_mix_columns_dec，CPU 占用率
aes mix columns enc 从 8.05% 隆到了 2.93%，aes mix columns dec 从 57.87% 隆到了

0.5%，函数 Time per Call 消耗 cycle 数 aes_mix_columns_enc 从 1141 cycle 降到了 146 cycle，aes_mix_columns_dec 从 8209 cycle 降到了 25 cycle，说明了通过 NICE/VNICE 指令替换热点函数可以大幅提高程序算法性能。



数据统计如下：(enc: aes_mix_columns_enc, dec: aes_mix_columns_dec)

Function	Before Optimization NICE/VNICE Optimization		
CPU Usage % (enc)	8.05	2.93	
CPU Usage % (dec)	57.87	0.5	
Time per Call Cycles (enc)	1,141	146	
Time per Call Cycles (dec)	8,209	25	
AES Program Total Before Optimization NICE/VNICE Optimization			
Cycles	161,108	42,066	

AES加解密 NICE/VNICE demo：[优化后AES工程链接下载](#)

Nuclei Model结合Nice Wizard快速验证NICE/VNICE指令加速¶

Nuclei Model 已支持 Windows/Linux 版本，此文档测试都是基于 Nuclei Studio 的 Windows 版本 (≥ 2025.10) 完成的。

背景描述¶

xlmodel_nice¶

Nuclei Model 会不断更新提供用户可自定义实现 NICE/VNICE 的 `xlmodel_nice` 软件包，用户通过在 `xlmodel_nice/nice/src/nice.cc` 实现指令的具体行为，编译出新的 Nuclei Model 供应用程序配置调用。

Nuclei NICE Wizard¶

Nuclei NICE Wizard 是 Nuclei Studio 上提供的 NICE/VNICE 指令生成控件，用户配置好自定义指令后，可以自动生成两个文件：

1. `insn.h`: 指令内嵌汇编头文件，用户需要将此文件的指令内嵌汇编添加到应用程序头文件中
2. `nice.cc`: 指令实现文件，用户需要将此文件的指令 `decode` 框架添加到 `xlmodel_nice/nice/src/nice.cc` 中

test code¶

在 AI 与深度学习中常见的批量矩阵运算中，存在需要多次处理小矩阵块的场景，此测试将使用标量的多个 4×4 矩阵的乘法和累加操作的算法函数作为 `golden_case`，然后通过配置 NICE Wizard 生成 NICE/VNICE 加速指令，分别添加到测试应用程序和 `xlmodel_nice` 软件包工程中重新编译，最后通过运行 Nuclei Model 查看优化后的算法函数的指令数和 `cycle` 数，以查看 NICE/VNICE 加速效果。

解决方案¶

环境准备¶

Nuclei Studio IDE 集成的 NICE Wizard 相关功能，需要配合 Nuclei CPU Model - NICE Support (`xlmodel_nice`) 软件包使用。

Nuclei Studio :

- NucleiStudio 202510 Windows
 - NucleiStudio 202510 Linux

xlmodel_nice :

- 原始xlmodel_nice软件包 Windows
 - 原始xlmodel_nice软件包 Linux

Nuclei Model运行原始程序

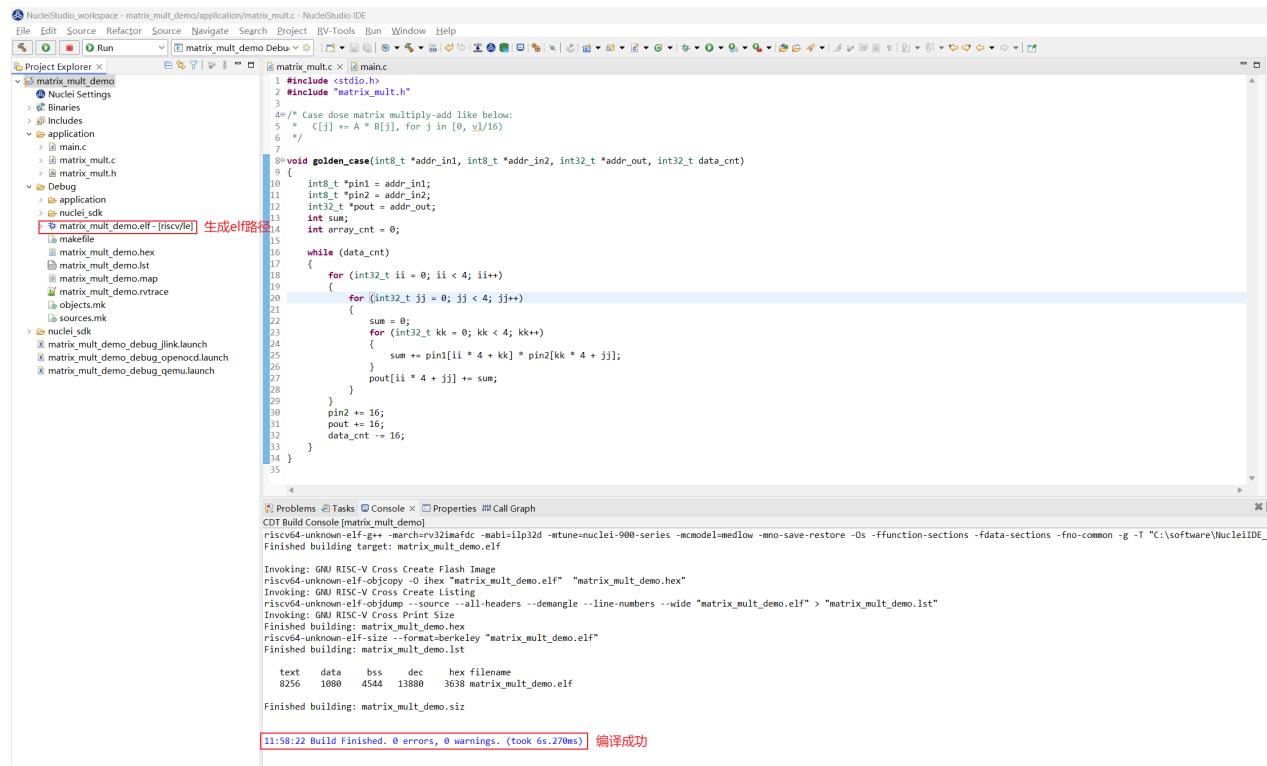
step1：导入 Nuclei SDK 原始工程

优化前的工程下载链接

下载 zip 包后，可以直接导入到 Nuclei Studio 中运行 (导入步骤 :File->Import->Existing Projects into Workspace->Select archive file->选择zip压缩包->Finish即可)

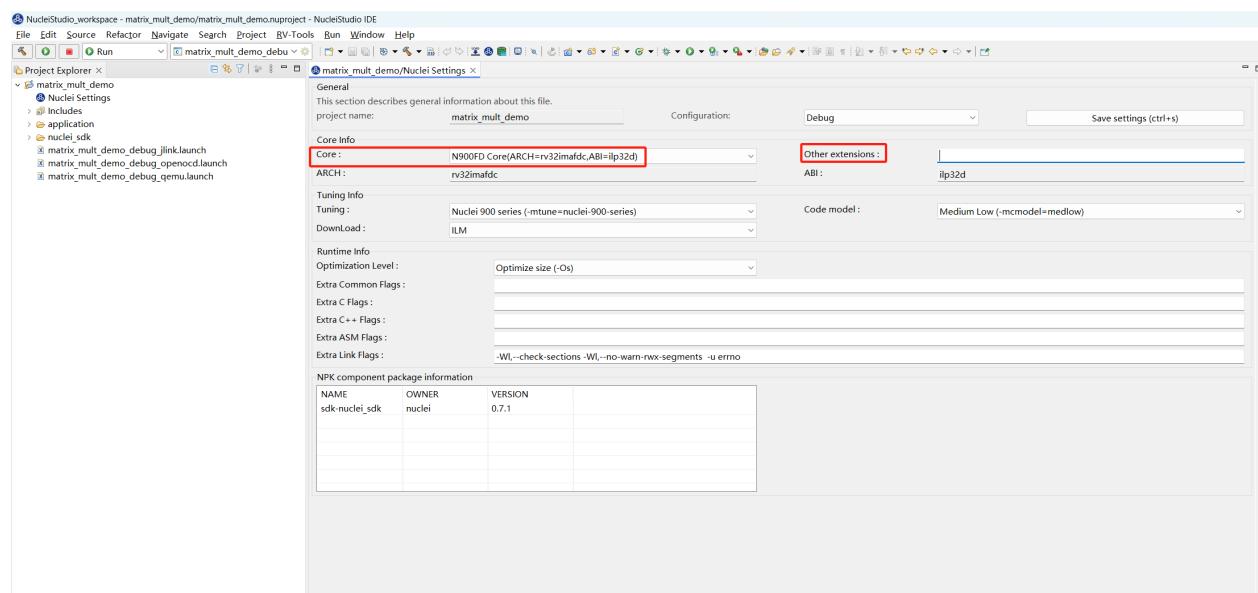
step2：编译 Nuclei SDK 原始工程

编译原始工程，确保编译成功以及在 Debug 下可以找到生成的 elf 文件：



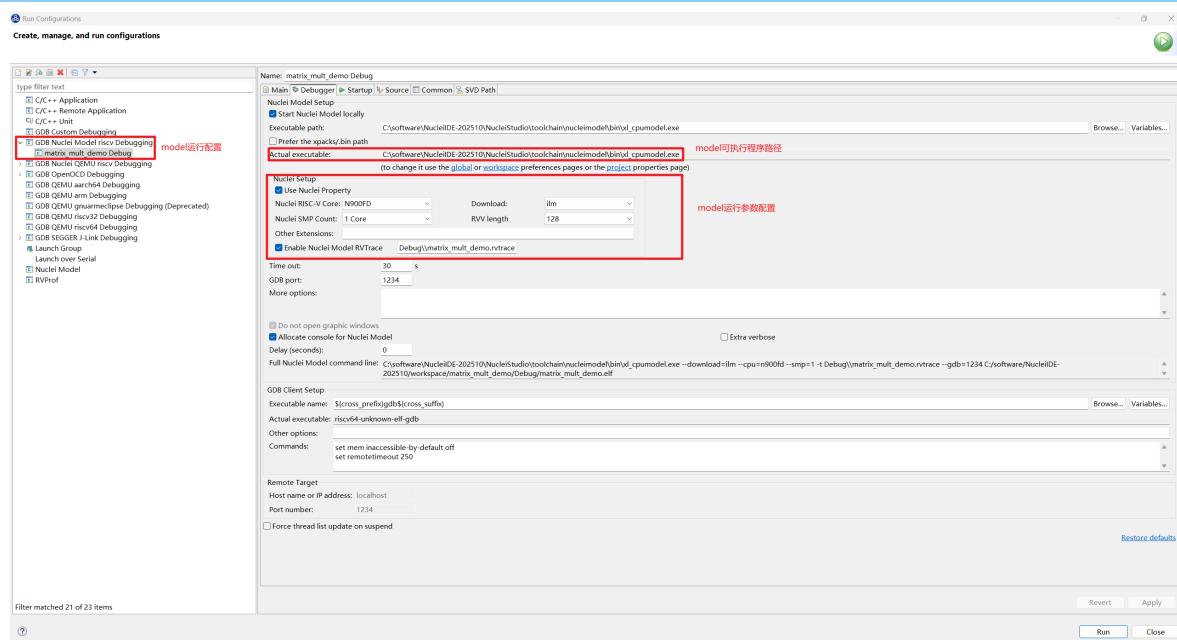
step3：运行 Nuclei SDK 原始工程

在使用 Nuclei Model 运行程序时，需要先确定工程 Nuclei Settings 中的 Core 配置和 Other extensions 配置，这些配置需要传递给 Model 使用。当前使用的 Core 是 n900fd，Other extensions 未配置。



Model 仿真程序需要配置 Nuclei Studio 中的 GDB Nuclei Model riscv Debugging 配置项，步骤如下：

1. 打开 Nuclei Studio 主菜单栏的 Run 选项的 Run Configurations
2. 选择 GDB Nuclei Model riscv Debugging 配置项，右键选择 New Configuration，会自动生成项目名的 Model 配置页面，launch bar 也会同步更新
3. 在右侧 Main 选项卡中点击 Search Project... 选择编译好的 elf 文件
4. 在右侧 Debugger 选项卡中选择 Browse 找到 Nuclei Model 可执行程序默认路径：
NucleiStudio/toolchain/nucleimodel/bin/xl_cpumodel.exe
5. 在右侧 Debugger 选项卡中的 Nuclei Setup 中完成 model 运行配置，选择 Nuclei RISC-V Core 和 Other Extensions 需要保持和 Nuclei Settings 的 Core 和 Other extensions 配置一致，Other Extensions 为空时不传递此参数，Enable Nuclei Model RVTrace 表示运行时生成 rvtrace，然后点击 Apply 和 Run，model 就开始运行程序了



Nuclei Studio (< 2025.10) 只能使用 Run Configurations 中的 Nuclei Model 来配置 model, Nuclei Studio (>= 2025.10) 建议切换到使用 GDB Nuclei Model riscv Debugging 来配置

在 Console 中会看到 Total elapsed real time 说明 model 已经完成仿真了, 程序会提取标量矩阵乘算法函数 golden_case 的执行指令数和 cycle 数如下：

```

1 #include <stdio.h>
2 #include "matrix_mult.h"
3
4 /* Case dose matrix multiply-add like below:
5   C[i] += A * B[j], for j in [0, v1/16]
6 */
7
8 void golden_case(int8_t *addr_in1, int8_t *addr_in2, int32_t *addr_out, int32_t data_cnt)
9 {
10    int8_t *pin1 = addr_in1;
11    int8_t *pin2 = addr_in2;
12    int32_t *pout = addr_out;
13    int sum;
14    int array_cnt = 0;
15
16    while (data_cnt)
17    {
18        for (int32_t ii = 0; ii < 4; ii++)
19        {
20            for (int32_t jj = 0; jj < 4; jj++)
21            {
22                sum = 0;
23                for (int32_t kk = 0; kk < 4; kk++)
24                {
25                    sum += pin1[ii * 4 + kk] * pin2[kk * 4 + jj];
26                }
27                pout[ii * 4 + jj] = sum;
28            }
29            pin2 += 16;
30            pout += 16;
31        }
32    }
}

```

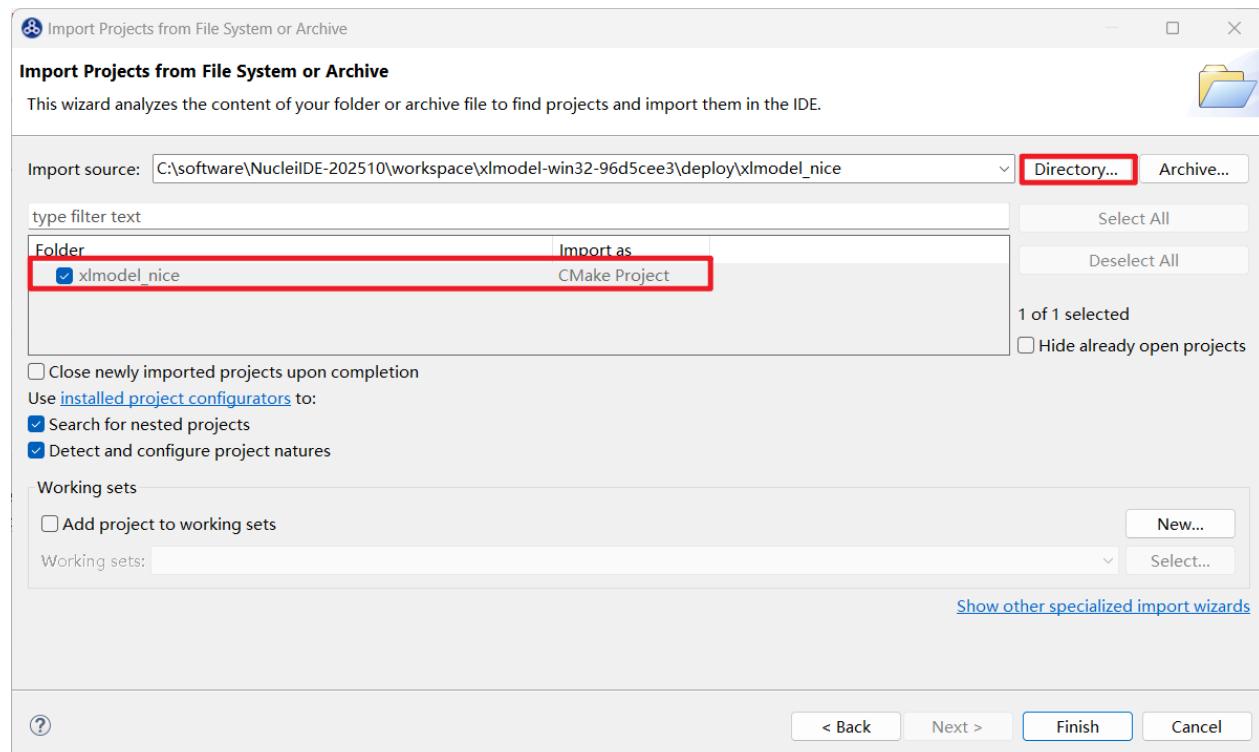
golden:
instret: 2854, cycle: 3820

total run 139874 instruction
[XLMODEL-INFO] Program simulation completed after 0.865867s
Info: /OSCI/SystemC: Simulation stopped by user, model仿真完成标志
[XLMODEL-INFO] Total elapsed real time: 0.865747s
[XLMODEL-INFO] Press Enter to finish

NICE指令替换¶

step1：编译 xlmodel_nice 软件包

下载并解压 xlmodel_nice.zip 包后，可以直接导入到 Nuclei Studio 中运行 (导入步骤：File->Import->Projects from Folder or Archive->Next->Directory->选择 xlmodel_nice 文件夹->Finish 即可)



在编译 xlmodel_nice 前需先配置好 xlmodel 的编译环境 ([xlmodel_nice 编译环境配置](#))，然后编译确保原始软件包可以成功编译生成 model 的可执行程序：

Nuclei Studio (< 2025.10) 生成的 elf 文件所在路径为 build/default/xl_cpumodel

```

1 #include "nice.h"
2
3 /* The macros NUCLEI_NICE_SCALAR and NUCLEI_NICE_VECTOR respectively represent the scalar and Vector-NICE instructions implemented by Nuclei.
4 If you need to implement your own NICE instructions, you can comment out these two macros.*/
5 #define NUCLEI_NICE_SCALAR
6 #define NUCLEI_NICE_VECTOR
7
8 #define CUSTOM0 (0b000010)
9 #define CUSTOM1 (0b01010)
10 #define CUSTOM2 (0b01110)
11 #define CUSTOM3 (0b11110)
12
13 #define VNICE_SIZE 1024
14
15 #define CLK_CYCLE_ADD 2
16 #define CSK_CYCLE_ADD 1
17 #define CACX_CYCLE_ADD 1
18 #define VNICE_LOAD 8
19 #define VNICE_STORE 8
20 #define VNICE_CMPLX_MUL 8
21
22 #define VL_LD_NICE(cstride, offset, elt_width, is_mask_ldst) \
23     const reg_t nf = 1; \
24     const reg_t vl = is_mask_ldst ? ((P.VU.vl->read() + 7) / 8) : P.VU.vl->read(); \
25     const reg_t vd = insn.rd(); \
26     VL_CDR(vd, elt_width, is_mask_ldst); \
27     for (reg_t fn = 0; fn < nf; ++fn) { \
28         VL_ELEMENT_SKIP(1); \
29         VL_STRIP(1); \
30         /*P.VU.vstart->write(1);*/ \
31         for (reg_t fn = 0; fn < nf; ++fn) { \
32             elt_width##_t val; \
33             memory((uint8_t *)Vnice_ld_st_addr((cstride) + (offset) * sizeof(elt_width##_t)), sizeof(elt_width##_t)); \
34             P.VU.vl[elv_width##_t](vd + fn * emul, vreg_idx, true) = val; \
35         } \
36     }
37
38 // The C compiler identification is GNU 14.2.0
39 // The CXX compiler identification is GNU 14.2.0
40 // Detecting C compiler ABI info
41 // Detecting C compiler ABI info - done
42 // Check for working C compiler: C:/software/msys2/mingw64/bin/cc.exe - skipped
43 // Detecting C compile features
44 // Detecting C compile features - done
45 // Detecting CXX compiler ABI info
46 // Detecting CXX compiler ABI info - done
47 // Check for working CXX compiler: C:/software/msys2/mingw64/bin/c++.exe - skipped
48 // Detecting CXX compile features
49 // Detecting CXX compile features - done
50 // Detecting CXX compile features - done
51 // Generating done (0.1s)
52 // Build files have been written to: C:/software/NucleiIDE-202510/workspace/xlmodel-win32-96d5ce3/deploy/xlmodel_nice/build/cmake.run.win32.x86_64.Local
Building in: C:\software\NucleiIDE-202510\workspace\xlmodel\win32-96d5ce3\deploy\xlmodel_nice\build\cmake.run.win32.x86_64.local
cmake -B . -target all
[1/2] Building CXX object CMakeFiles/xl_pumodel.dir/nice/src/nice.cc.obj
[2/2] Linking CXX executable xl_pumodel.exe
Build complete (0 errors, 0 warnings) C:\software\NucleiIDE-202510\workspace\xlmodel-win32-96d5ce3\deploy\xlmodel_nice\build\cmake.run.win32.x86_64.local

```

编译model成功

step2：NICE Wizard生成NICE指令替换

应用程序的热点函数可以先用 Nuclei Model Profiling 来定位，具体使用可以参考 [通过Profiling展示Nuclei Model NICE/VNICE指令加速](#)，这里不再赘述了。

此用例的热点函数已知是矩阵乘累加，A矩阵某行 * B矩阵某列计算如下：

```

for (int32_t kk = 0; kk < 4; kk++)
{
    sum += pin1[ii * 4 + kk] * pin2[kk * 4 + jj];
}

```

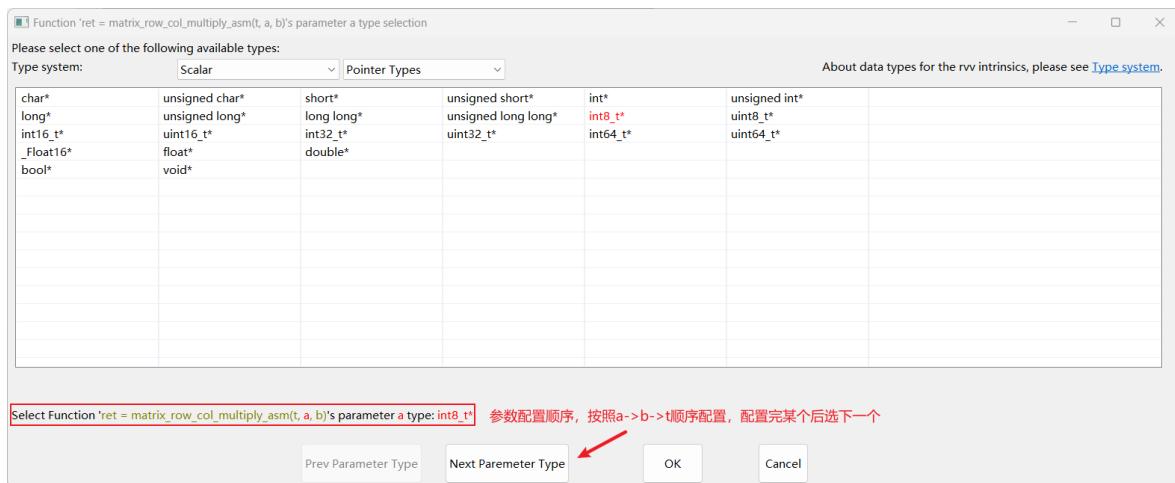
此算法完全可以替换成一条 NICE 指令来完成，输入为 sum 值， pin1 地址， pin2 地址，输出为 sum。

接下来用 NICE Wizard 来生成设想的 NICE 指令，用户可以在 Nuclei Studio 的 xlmodel_nice 工程根目录创建一个 aicc.nice 的文件，此文件创建后就会弹出 NICE Wizard 的指令生成窗口，配置生成 NICE 指令步骤如下：

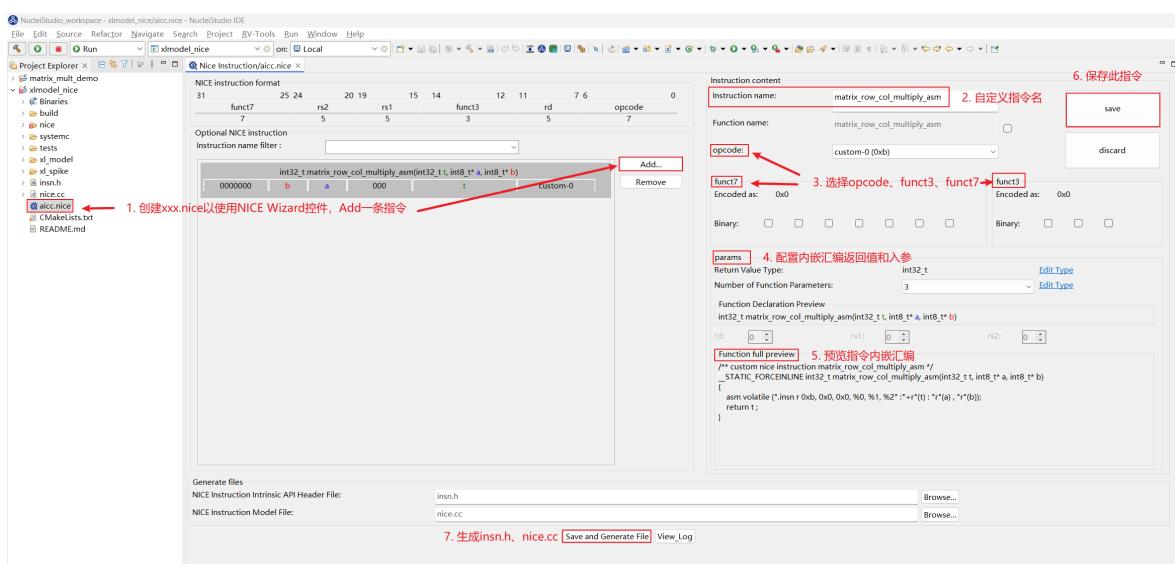
- 选择 Add 添加一条 NICE 指令，指令格式如左上角 NICE instruction format 所示，首先填写 Instruction name 项为 matrix_row_col_multiply_asm 表示矩阵行列乘加操作
- 依次选择填写 opcode、funct3、funct7

3. `params` 是指令内嵌汇编的返回值和入参配置，构想的 NICE 指令返回值为 `int32_t`，入参个数为3个，分别是 `int32_t t`、`int8_t* a`、`int8_t* b`，分别在 `params` 中设置好

注意：在入参的 `Edit Type` 设置界面中，是按照 `a->b->t` 的顺序配置的：



4. 在 `Function full preview` 中预览指令内嵌汇编格式是否正确，确保没有问题后点击 `save`，`save` 完成后可以在左侧指令栏中看到生成好的自定义指令了
5. 点击下方 `Save and Generate File`，在 `aicc.nice` 同路径下会生成 `insn.h` 和 `nice.cc`



6. 将生成好的 `insn.h` 中的 NICE 指令内嵌汇编复制到应用程序的头文件中，将生成好的 `nice.cc` 直接替换 `xlmodel_nice/nice/src/nice.cc`

```

insn.h
1 #ifndef __INSN_H__
2 #define __INSN_H__
3 
4 #ifdef __cplusplus
5 extern "C" {
6 #endif
7 
8 #include <stdint.h>
9 #include <nuclei_soc.h>
10 #include <criscc_vector.h>
11 /* custom nice instruction matrix_row_col_multiply_asm */
12 static inline int32_t matrix_row_col_multiply_asm(int32_t t, int8_t* a, int8_t* b)
13 {
14     asm volatile ("*.insn r %0, %1, %2, %3" : "+r"(t) : "r"(a), "r"(b));
15     return t;
16 }
17 
18 #endif // __INSN_H__

```

```

nice.cc
1 #include "nice.h"
2 /* The macros NUCLEI_NICE_SCALAR and NUCLEI_NICE_VECTOR respectively represent the scalar and Vector-NICE instructions implemented by Nuclei.
3 If you need to implement your own NICE instructions, you can comment out these two macros.*/
4 #define NUCLEI_NICE_SCALAR
5 #define NUCLEI_NICE_VECTOR
6 
7 
8 void do_nice(processor_t* p, insn_t insn, reg_t pc)
9 {
10     (void)p;
11     uint32_t instr = insn.bits();
12     uint32_t opcode = instr & 0x7f;
13     uint32_t funct7 = (instr >> 5) & 0x7f;
14     uint32_t funct3 = (instr >> 12) & 0x7f;
15     uint32_t rd = (instr >> 7) & 0x1f;
16     uint32_t rs1 = (instr >> 15) & 0x1f;
17     uint32_t rs2 = (instr >> 20) & 0x1f;
18 
19     if (opcode == 0xb && funct3 == 0xb && funct7 == 0x0) {
20         /* Implement matrix_row_col_multiply_asm here */
21         int8_t r1, r2;
22         int32_t sum = RD;
23         for (int i = 0; i < 4; i++) {
24             r1 = MMU_load_int8(RS1 + i);
25             r2 = MMU_load_int8(RS2 + i * 4);
26             sum += r1 * r2;
27         }
28         WRITE_RD(sum);
29         /* Modify matrix_row_col_multiply_asm cycle here, default is 1 */
30         STATE.mcycle+=bump(1);
31     }
32 }

```

当然也可以将 `insn.h` 直接生成到应用程序工程路径下引用，这样可以省去每次手动的复制文件内容。

step3 : xlmodel_nice实现NICE指令

打开 `xlmodel_nice/nice/src/nice.cc` 文件，使用 spike 中定义的宏来实现 NICE 指令：
MMU 宏表示 memory 访问，load memory 使用 `MMU.load_xxx<n>`，store memory 使用 `MMU.store_xxx<n>`，RD、RS1、RS2、RS3 宏表示其对应标量寄存器中的值，写目标寄存器使用 `WRITE_RD`，这些宏的使用可以参考 `nice/inc/decode_macros.h`。

在指令实现完后，将自定义指令额外需要的 cycle 数 n 直接标定：`STATE.mcycle->bump(n)`；即可，这里标定此条 NICE 指令额外需要 1 cycle，由于指令默认需要 1 cycle，因此此条 NICE 指令需要消耗 2 cycle。

实现的 NICE 指令实现和 cycle 标定如下：

```

nice.cc
1 #include "nice.h"
2 /* The macros NUCLEI_NICE_SCALAR and NUCLEI_NICE_VECTOR respectively represent the scalar and Vector-NICE instructions implemented by Nuclei.
3 If you need to implement your own NICE instructions, you can comment out these two macros.*/
4 #define NUCLEI_NICE_SCALAR
5 #define NUCLEI_NICE_VECTOR
6 
7 
8 void do_nice(processor_t* p, insn_t insn, reg_t pc)
9 {
10     (void)p;
11     uint32_t instr = insn.bits();
12     uint32_t opcode = instr & 0x7f;
13     uint32_t funct7 = (instr >> 5) & 0x7f;
14     uint32_t funct3 = (instr >> 12) & 0x7f;
15     uint32_t rd = (instr >> 7) & 0x1f;
16     uint32_t rs1 = (instr >> 15) & 0x1f;
17     uint32_t rs2 = (instr >> 20) & 0x1f;
18 
19     if (opcode == 0xb && funct3 == 0xb && funct7 == 0x0) {
20         /* Implement matrix_row_col_multiply_asm here */
21         int8_t r1, r2;
22         int32_t sum = RD;
23         for (int i = 0; i < 4; i++) {
24             r1 = MMU_load_int8(RS1 + i);
25             r2 = MMU_load_int8(RS2 + i * 4);
26             sum += r1 * r2;
27         }
28         WRITE_RD(sum);
29         /* Modify matrix_row_col_multiply_asm cycle here, default is 1 */
30         STATE.mcycle+=bump(1);
31     }
32 }

```

重新编译 `xlmodel_nice` 保证编译通过。

step4：Nuclei Model重新运行程序

首先需要编写一个带 NICE 指令内嵌汇编的算法函数 nice_case 方便和 golden_case 对比，添加函数输出结果比对，然后重新编译应用程序工程：

The screenshot shows the NucleoStudio IDE interface with two code editors and a terminal window.

Code Editors:

- matrix_mult.c**: Contains the C code for matrix multiplication. A red arrow points from the line `asm("matrix_row_col_multiply.asm", sum, &pinl[i * 4], &pin2[j]);` to the assembly label `nicem_case` in the **xmodel.nice** editor.
- xmodel.nice**: Contains the NICE assembly code for the `nicem_case` function.
- main.c**: Contains the main program logic, including printing debug information and comparing results.

Terminal Window:

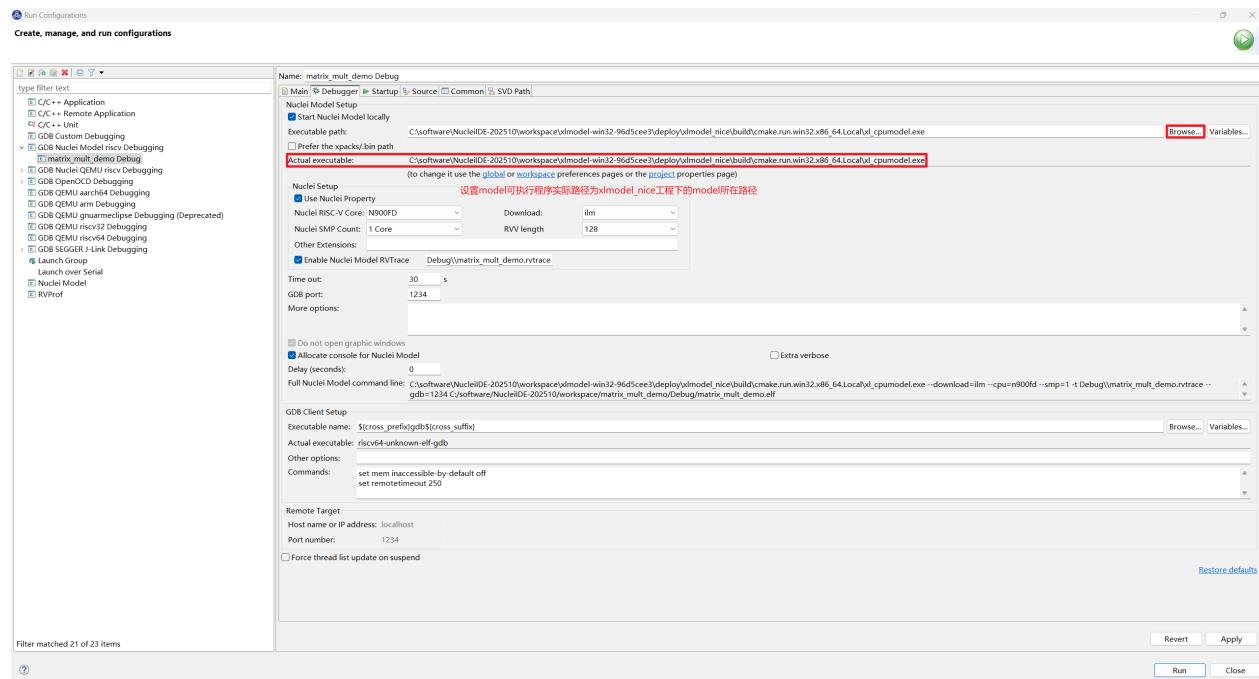
```
Inoking: GM RISC-V Cross Create Flash Image
riscv64-unknown-elf-objcopy -O ihex "matrix_mult_demo.elf" "matrix_mult_demo.hex"
Inoking: GM RISC-V Cross Create Listing
riscv64-unknown-elf-objdump -s --all-headers --demangle --line-numbers --wide "matrix_mult_demo.elf" > "matrix_mult_demo.lst"
Invoking: GM RISC-V Cross Print Size
riscv64-unknown-elf-size --format=berkeley "matrix_mult_demo.elf"
Finished building: matrix_mult_demo.hex

Finished building: matrix_mult_demo.lst

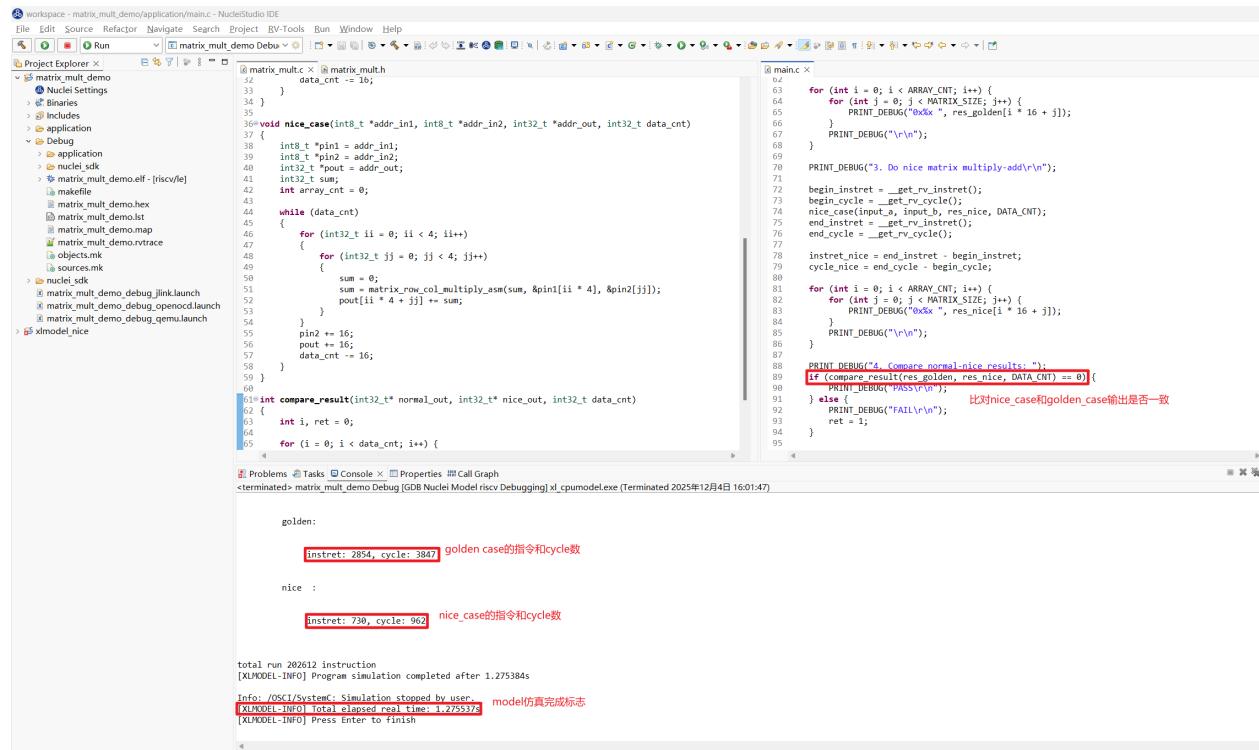
      text    data     bss     dec   hex filename
 8784 1288 4544 14536 38C8 matrix_mult_demo.elf
Finished building: matrix_mult_demo.siz

17:24:17 Build Finished. 0 errors, 0 warnings. (took 75.482ms)
```

因为 model 已经使用 xlmodel_nice 重新编译出新的可执行程序了，需要重新配置 Nuclei Studio Nuclei Model 配置项中的 model 可执行程序路径为 xlmodel_nice/build/default/xl_cpumodel.exe，其余配置不变：

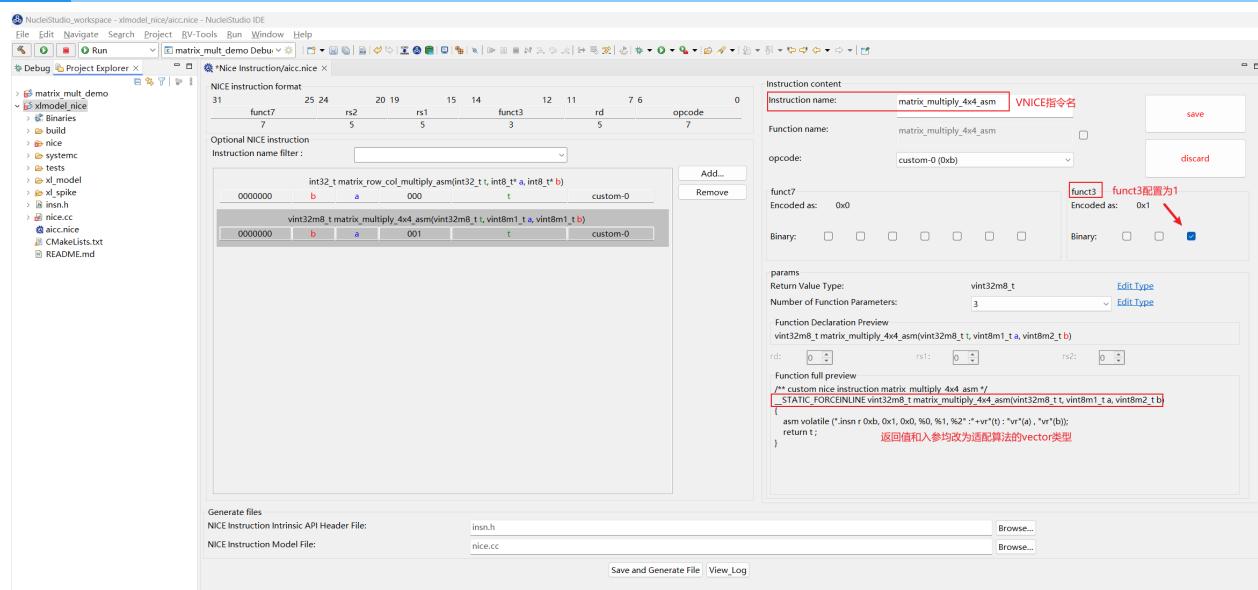


Apply 后重新 Run 应用程序，可以发现 `nice_case` 和 `golden_case` 输出结果一致，`nice_case` 的指令数和 `cycle` 数均大幅下降了，构想的 NICE 指令实现正确，并优化了原标量算法。

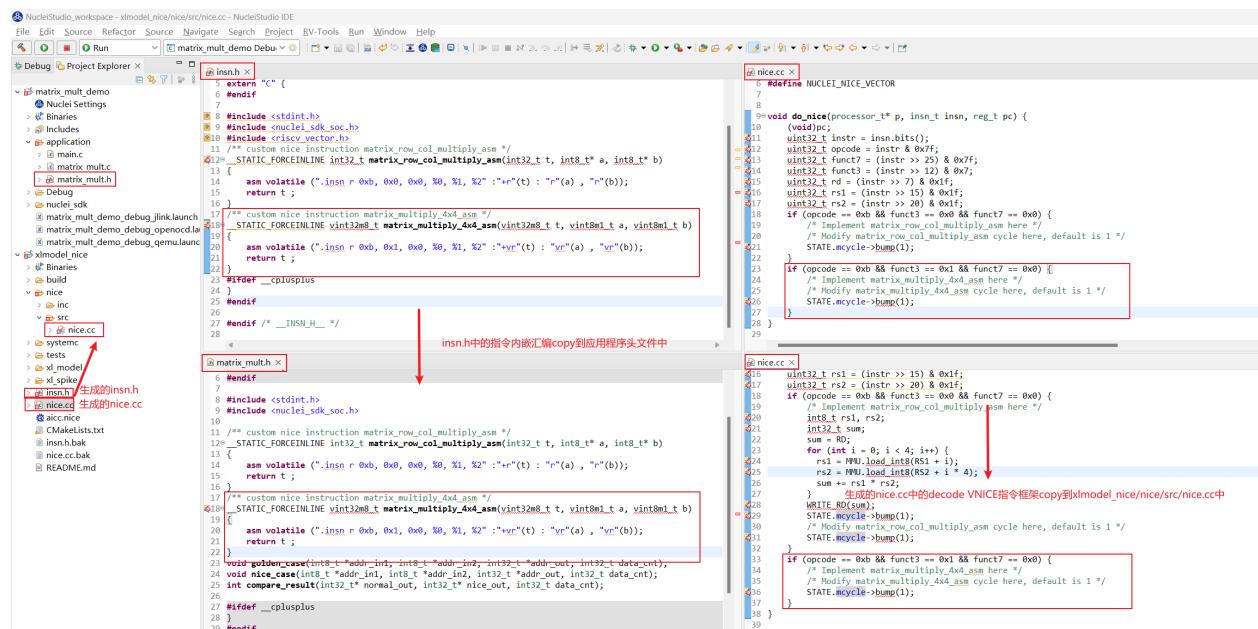


```

workspace - matrix_mult_demo/application/main.c - NucleiStudio IDE
File Edit Source Refactor Search Project RV-Tools Run Window Help
Project Explorer X
matrix_mult_demo
  Nuclei Settings
  Binaries
  Includes
  Application
  Debug
  matrix_mult_demo
    application
    nuclei_sdk
    matrix_mult_demo.elf - [riscv/le]
    makefile
    matrix_mult_demo.hex
    matrix_mult_demo.lst
    matrix_mult_demo.map
    matrix_mult_demo.vtrace
    matrix_mult_demo.vtraces
    source.sink
  nuclei_sdk
  matrix_mult_demo.debug.jlink.launch
  matrix_mult_demo.debug.openocd.launch
  matrix_mult_demo.debug.qemu.launch
  xmodel.nice
@ main.c X
  b2
  63  void nice_case(int8_t *addr_in1, int8_t *addr_in2, int32_t *addr_out, int32_t data_cnt)
  64  {
  65      for (int i = 0; i < ARRAY_CNT; i++) {
  66          for (int j = 0; j < MATRIX_SIZE; j++) {
  67              PRINT_DEBUG("0x%08x ", res_golden[i * 16 + j]);
  68          }
  69          PRINT_DEBUG("\r\n");
  70      }
  71      PRINT_DEBUG("3. Do nice matrix multiply-add\r\n");
  72      begin_instrret = __get_rv_instrret();
  73      begin_cycle = __get_rv_cycle();
  74      nice_case(input_a, input_b, res_nice, DATA_CNT);
  75      end_instrret = __get_rv_instrret();
  76      end_cycle = __get_rv_cycle();
  77
  78      instrret_nice = end_instrret - begin_instrret;
  79      cycle_nice = end_cycle - begin_cycle;
  80
  81      for (int i = 0; i < ARRAY_CNT; i++) {
  82          for (int j = 0; j < MATRIX_SIZE; j++) {
  83              PRINT_DEBUG("0x%08x ", res_nice[i * 16 + j]);
  84          }
  85          PRINT_DEBUG("\r\n");
  86      }
  87
  88      PRINT_DEBUG("4. Compare normal nice results: ");
  89      if (compare_result(res_golden, res_nice, DATA_CNT) == 0) {
  90          PRINT_DEBUG("PASS!\r\n");
  91      } else {
  92          PRINT_DEBUG("FAIL!\r\n");
  93      }
  94  }
  95
  61 int compare_result(int32_t* normal_out, int32_t* nice_out, int32_t data_cnt)
  62 {
  63     int i, ret = 0;
  64
  65     for (i = 0; i < data_cnt; i++) {
  66         if (normal_out[i] != nice_out[i])
  67             ret++;
  68     }
  69
  70     return ret;
  71 }
  72
  73
  74
  75
  76
  77
  78
  79
  80
  81
  82
  83
  84
  85
  86
  87
  88
  89
  90
  91
  92
  93
  94
  95
  96
  97
  98
  99
  100
  101
  102
  103
  104
  105
  106
  107
  108
  109
  110
  111
  112
  113
  114
  115
  116
  117
  118
  119
  120
  121
  122
  123
  124
  125
  126
  127
  128
  129
  130
  131
  132
  133
  134
  135
  136
  137
  138
  139
  140
  141
  142
  143
  144
  145
  146
  147
  148
  149
  150
  151
  152
  153
  154
  155
  156
  157
  158
  159
  160
  161
  162
  163
  164
  165
  166
  167
  168
  169
  170
  171
  172
  173
  174
  175
  176
  177
  178
  179
  180
  181
  182
  183
  184
  185
  186
  187
  188
  189
  190
  191
  192
  193
  194
  195
  196
  197
  198
  199
  200
  201
  202
  203
  204
  205
  206
  207
  208
  209
  210
  211
  212
  213
  214
  215
  216
  217
  218
  219
  220
  221
  222
  223
  224
  225
  226
  227
  228
  229
  230
  231
  232
  233
  234
  235
  236
  237
  238
  239
  240
  241
  242
  243
  244
  245
  246
  247
  248
  249
  250
  251
  252
  253
  254
  255
  256
  257
  258
  259
  260
  261
  262
  263
  264
  265
  266
  267
  268
  269
  270
  271
  272
  273
  274
  275
  276
  277
  278
  279
  280
  281
  282
  283
  284
  285
  286
  287
  288
  289
  290
  291
  292
  293
  294
  295
  296
  297
  298
  299
  300
  301
  302
  303
  304
  305
  306
  307
  308
  309
  310
  311
  312
  313
  314
  315
  316
  317
  318
  319
  320
  321
  322
  323
  324
  325
  326
  327
  328
  329
  330
  331
  332
  333
  334
  335
  336
  337
  338
  339
  340
  341
  342
  343
  344
  345
  346
  347
  348
  349
  350
  351
  352
  353
  354
  355
  356
  357
  358
  359
  360
  361
  362
  363
  364
  365
  366
  367
  368
  369
  370
  371
  372
  373
  374
  375
  376
  377
  378
  379
  380
  381
  382
  383
  384
  385
  386
  387
  388
  389
  390
  391
  392
  393
  394
  395
  396
  397
  398
  399
  400
  401
  402
  403
  404
  405
  406
  407
  408
  409
  410
  411
  412
  413
  414
  415
  416
  417
  418
  419
  420
  421
  422
  423
  424
  425
  426
  427
  428
  429
  430
  431
  432
  433
  434
  435
  436
  437
  438
  439
  440
  441
  442
  443
  444
  445
  446
  447
  448
  449
  450
  451
  452
  453
  454
  455
  456
  457
  458
  459
  460
  461
  462
  463
  464
  465
  466
  467
  468
  469
  470
  471
  472
  473
  474
  475
  476
  477
  478
  479
  480
  481
  482
  483
  484
  485
  486
  487
  488
  489
  490
  491
  492
  493
  494
  495
  496
  497
  498
  499
  500
  501
  502
  503
  504
  505
  506
  507
  508
  509
  510
  511
  512
  513
  514
  515
  516
  517
  518
  519
  520
  521
  522
  523
  524
  525
  526
  527
  528
  529
  530
  531
  532
  533
  534
  535
  536
  537
  538
  539
  540
  541
  542
  543
  544
  545
  546
  547
  548
  549
  550
  551
  552
  553
  554
  555
  556
  557
  558
  559
  560
  561
  562
  563
  564
  565
  566
  567
  568
  569
  570
  571
  572
  573
  574
  575
  576
  577
  578
  579
  580
  581
  582
  583
  584
  585
  586
  587
  588
  589
  590
  591
  592
  593
  594
  595
  596
  597
  598
  599
  600
  601
  602
  603
  604
  605
  606
  607
  608
  609
  610
  611
  612
  613
  614
  615
  616
  617
  618
  619
  620
  621
  622
  623
  624
  625
  626
  627
  628
  629
  630
  631
  632
  633
  634
  635
  636
  637
  638
  639
  640
  641
  642
  643
  644
  645
  646
  647
  648
  649
  650
  651
  652
  653
  654
  655
  656
  657
  658
  659
  660
  661
  662
  663
  664
  665
  666
  667
  668
  669
  670
  671
  672
  673
  674
  675
  676
  677
  678
  679
  680
  681
  682
  683
  684
  685
  686
  687
  688
  689
  690
  691
  692
  693
  694
  695
  696
  697
  698
  699
  700
  701
  702
  703
  704
  705
  706
  707
  708
  709
  710
  711
  712
  713
  714
  715
  716
  717
  718
  719
  720
  721
  722
  723
  724
  725
  726
  727
  728
  729
  730
  731
  732
  733
  734
  735
  736
  737
  738
  739
  740
  741
  742
  743
  744
  745
  746
  747
  748
  749
  750
  751
  752
  753
  754
  755
  756
  757
  758
  759
  760
  761
  762
  763
  764
  765
  766
  767
  768
  769
  770
  771
  772
  773
  774
  775
  776
  777
  778
  779
  780
  781
  782
  783
  784
  785
  786
  787
  788
  789
  790
  791
  792
  793
  794
  795
  796
  797
  798
  799
  800
  801
  802
  803
  804
  805
  806
  807
  808
  809
  810
  811
  812
  813
  814
  815
  816
  817
  818
  819
  820
  821
  822
  823
  824
  825
  826
  827
  828
  829
  830
  831
  832
  833
  834
  835
  836
  837
  838
  839
  840
  841
  842
  843
  844
  845
  846
  847
  848
  849
  850
  851
  852
  853
  854
  855
  856
  857
  858
  859
  860
  861
  862
  863
  864
  865
  866
  867
  868
  869
  870
  871
  872
  873
  874
  875
  876
  877
  878
  879
  880
  881
  882
  883
  884
  885
  886
  887
  888
  889
  890
  891
  892
  893
  894
  895
  896
  897
  898
  899
  900
  901
  902
  903
  904
  905
  906
  907
  908
  909
  910
  911
  912
  913
  914
  915
  916
  917
  918
  919
  920
  921
  922
  923
  924
  925
  926
  927
  928
  929
  930
  931
  932
  933
  934
  935
  936
  937
  938
  939
  940
  941
  942
  943
  944
  945
  946
  947
  948
  949
  950
  951
  952
  953
  954
  955
  956
  957
  958
  959
  960
  961
  962
  963
  964
  965
  966
  967
  968
  969
  970
  971
  972
  973
  974
  975
  976
  977
  978
  979
  980
  981
  982
  983
  984
  985
  986
  987
  988
  989
  990
  991
  992
  993
  994
  995
  996
  997
  998
  999
  1000
  1001
  1002
  1003
  1004
  1005
  1006
  1007
  1008
  1009
  1010
  1011
  1012
  1013
  1014
  1015
  1016
  1017
  1018
  1019
  1020
  1021
  1022
  1023
  1024
  1025
  1026
  1027
  1028
  1029
  1030
  1031
  1032
  1033
  1034
  1035
  1036
  1037
  1038
  1039
  1040
  1041
  1042
  1043
  1044
  1045
  1046
  1047
  1048
  1049
  1050
  1051
  1052
  1053
  1054
  1055
  1056
  1057
  1058
  1059
  1060
  1061
  1062
  1063
  1064
  1065
  1066
  1067
  1068
  1069
  1070
  1071
  1072
  1073
  1074
  1075
  1076
  1077
  1078
  1079
  1080
  1081
  1082
  1083
  1084
  1085
  1086
  1087
  1088
  1089
  1090
  1091
  1092
  1093
  1094
  1095
  1096
  1097
  1098
  1099
  1100
  1101
  1102
  1103
  1104
  1105
  1106
  1107
  1108
  1109
  1110
  1111
  1112
  1113
  1114
  1115
  1116
  1117
  1118
  1119
  1120
  1121
  1122
  1123
  1124
  1125
  1126
  1127
  1128
  1129
  1130
  1131
  1132
  1133
  1134
  1135
  1136
  1137
  1138
  1139
  1140
  1141
  1142
  1143
  1144
  1145
  1146
  1147
  1148
  1149
  1150
  1151
  1152
  1153
  1154
  1155
  1156
  1157
  1158
  1159
  1160
  1161
  1162
  1163
  1164
  1165
  1166
  1167
  1168
  1169
  1170
  1171
  1172
  1173
  1174
  1175
  1176
  1177
  1178
  1179
  1180
  1181
  1182
  1183
  1184
  1185
  1186
  1187
  1188
  1189
  1190
  1191
  1192
  1193
  1194
  1195
  1196
  1197
  1198
  1199
  1200
  1201
  1202
  1203
  1204
  1205
  1206
  1207
  1208
  1209
  1210
  1211
  1212
  1213
  1214
  1215
  1216
  1217
  1218
  1219
  1220
  1221
  1222
  1223
  1224
  1225
  1226
  1227
  1228
  1229
  1230
  1231
  1232
  1233
  1234
  1235
  1236
  1237
  1238
  1239
  1240
  1241
  1242
  1243
  1244
  1245
  1246
  1247
  1248
  1249
  1250
  1251
  1252
  1253
  1254
  1255
  1256
  1257
  1258
  1259
  1260
  1261
  1262
  1263
  1264
  1265
  1266
  1267
  1268
  1269
  1270
  1271
  1272
  1273
  1274
  1275
  1276
  1277
  1278
  1279
  1280
  1281
  1282
  1283
  1284
  1285
  1286
  1287
  1288
  1289
  1290
  1291
  1292
  1293
  1294
  1295
  1296
  1297
  1298
  1299
  1300
  1301
  1302
  1303
  1304
  1305
  1306
  1307
  1308
  1309
  1310
  1311
  1312
  1313
  1314
  1315
  1316
  1317
  1318
  1319
  1320
  1321
  1322
  1323
  1324
  1325
  1326
  1327
  1328
  1329
  1330
  1331
  1332
  1333
  1334
  1335
  1336
  1337
  1338
  1339
  1340
  1341
  1342
  1343
  1344
  1345
  1346
  1347
  1348
  1349
  1350
  1351
  1352
  1353
  1354
  1355
  1356
  1357
  1358
  1359
  1360
  1361
  1362
  1363
  1364
  1365
  1366
  1367
  1368
  1369
  1370
  1371
  1372
  1373
  1374
  1375
  1376
  1377
  1378
  1379
  1380
  1381
  1382
  1383
  1384
  1385
  1386
  1387
  1388
  1389
  1390
  1391
  1392
  1393
  1394
  1395
  1396
  1397
  1398
  1399
  1400
  1401
  1402
  1403
  1404
  1405
  1406
  1407
  1408
  1409
  1410
  1411
  1412
  1413
  1414
  1415
  1416
  1417
  1418
  1419
  1420
  1421
  1422
  1423
  1424
  1425
  1426
  1427
  1428
  1429
  1430
  1431
  1432
  1433
  1434
  1435
  1436
  1437
  1438
  1439
  1440
  1441
  1442
  1443
  1444
  1445
  1446
  1447
  1448
  1449
  1450
  1451
  1452
  1453
  1454
  1455
  1456
  1457
  1458
  1459
  1460
  1461
  1462
  1463
  1464
  1465
  1466
  1467
  1468
  1469
  1470
  1471
  1472
  1473
  1474
  1475
  1476
  1477
  1478
  1479
  1480
  1481
  1482
  1483
  1484
  1485
  1486
  1487
  1488
  1489
  1490
  1491
  1492
  1493
  1494
  1495
  1496
  1497
  1498
  1499
  1500
  1501
  1502
  1503
  1504
  1505
  1506
  1507
  1508
  1509
  1510
  1511
  1512
  1513
  1514
  1515
  1516
  1517
  1518
  1519
  1520
  1521
  1522
  1523
  1524
  1525
  1526
  1527
  1528
  1529
  1530
  1531
  1532
  1533
  1534
  1535
  1536
  1537
  1538
  1539
  1540
  1541
  1542
  1543
  1544
  1545
  1546
  1547
  1548
  1549
  1550
  1551
  1552
  1553
  1554
  1555
  1556
  1557
  1558
  1559
  1560
  1561
  1562
  1563
  1564
  1565
  1566
  1567
  1568
  1569
  1570
  1571
  1572
  1573
  1574
  1575
  1576
  1577
  1578
  1579
  1580
  1581
  1582
  1583
  1584
  1585
  1586
  1587
  1588
  1589
  1590
  1591
  1592
  1593
  1594
  1595
  1596
  1597
  1598
  1599
  1600
  1601
  1602
  1603
  1604
  1605
  1606
  1607
  1608
  1609
  1610
  1611
  1612
  1613
  1614
  1615
  1616
  1617
  1618
  1619
  1620
  1621
  1622
  1623
  1624
  1625
  1626
  1627
  1628
  1629
  1630
  1631
  1632
  1633
  1634
  1635
  1636
  1637
  1638
  1639
  1640
  1641
  1642
  1643
  1644
  1645
  1646
  1647
  1648
  1649
  1650
  1651
  1652
  1653
  1654
  1655
```



点击下方 Save and Generate File，覆盖之前生成的 `insn.h` 和 `nice.cc`，此时在同路径下还会出现 `insn.h.bak` 和 `nice.cc.bak`，这两个文件是上一次保存的 `insn.h` 和 `nice.cc` 备份文件不会被用到，再次将生成好的 `insn.h` 中的 NICE 指令内嵌汇编复制到应用程序的头文件中，将生成好的 `nice.cc` 中的新指令 decode 框架复制到 `xlmodel_nice/nice/src/nice.cc`：



step2 : xlmodel_nice实现VNICE指令

在 `xlmodel_nice/nice/src/nice.cc` 中实现 VNICE 指令，`V_MATRIX_ST` 实现将指令输入的 vector 寄存器 store 到自定义 buffer 中，`V_MATRIX_LD` 实现将指令输出的结果 load 到 RD 寄存器，`V_MATRIX_CALC` 实现两矩阵乘加运算，VNICE 指令实现可以参考 spike 中的 vector 指令实现：`xlmodel_nice/xl_spike/include/riscv/v_ext_macros.h`

标定此条 VNICE 指令需要 2 cycle，即实际消耗 3 cycle，实现的 VNICE 指令实现和 cycle 标定如下：

```

1 // Implement matrix_row_col_multiply_asm here
2 void do_nice(processor_t* p, insn_t insn, reg_t pc) {
3     /* Example: You need to add the instruction cycle to the implementation of each instruction in the format of STATE->bump(x),
4      * where x is the expected cycle count of this instruction minus 1.
5      * For example, if the expected cycle count of the instruction is 2, then x would be 1.
6      */
7     uint32_t instr = insn.bits();
8     uint32_t opcode = instr & 0x7f;
9     uint32_t funct3 = (instr >> 12) & 0xf;
10    uint32_t rd = (instr >> 7) & 0xf;
11    uint32_t rs1 = (instr >> 15) & 0xf;
12    uint32_t rs2 = (instr >> 20) & 0xf;
13
14    if (opcode == 0xb && funct3 == 0x0 && funct7 == 0x0) {
15        /* Implement matrix_row_col_multiply_asm here */
16        int32_t rs1, rs2;
17        int32_t sum;
18
19        sum = RD;
20
21        for (int i = 0; i < 4; i++) {
22            rs1 = MMU_load_int32(rs1 + i * 4);
23            rs2 = MMU_load_int32(rs2 + i * 4);
24            sum += rs1 * rs2;
25        }
26
27        /* RTE_RD(sum); */
28
29        /* Modify matrix_row_col_multiply_asm cycle here, default is 1 */
30        STATE->bump(1);
31
32    }
33
34    if (opcode == 0xb && funct3 == 0x1 && funct7 == 0x0) {
35        /* Implement matrix_multiply_4x4_asm here */
36        memset((uint8_t *)vnice_op1_addr, 0, VNICE_SIZE);
37        memset((uint8_t *)vnice_op2_addr, 0, VNICE_SIZE);
38        memset((uint8_t *)vnice_res_addr, 0, VNICE_SIZE*4);
39
40        V_MATRIX_S1D0((i * nf + fn), uint32, false, vnicc_op1_addr, (reg_t)insn.rs1(), vnicc_op2_addr, (reg_t)insn.rs2(), vnicc_res_addr, (reg_t)insn.rd());
41        V_MATRIX_CALC();
42
43        /* Modify matrix_multiply_4x4_asm cycle here, default is 1 */
44        STATE->bump(2);
45
46    }
47
48 }
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123

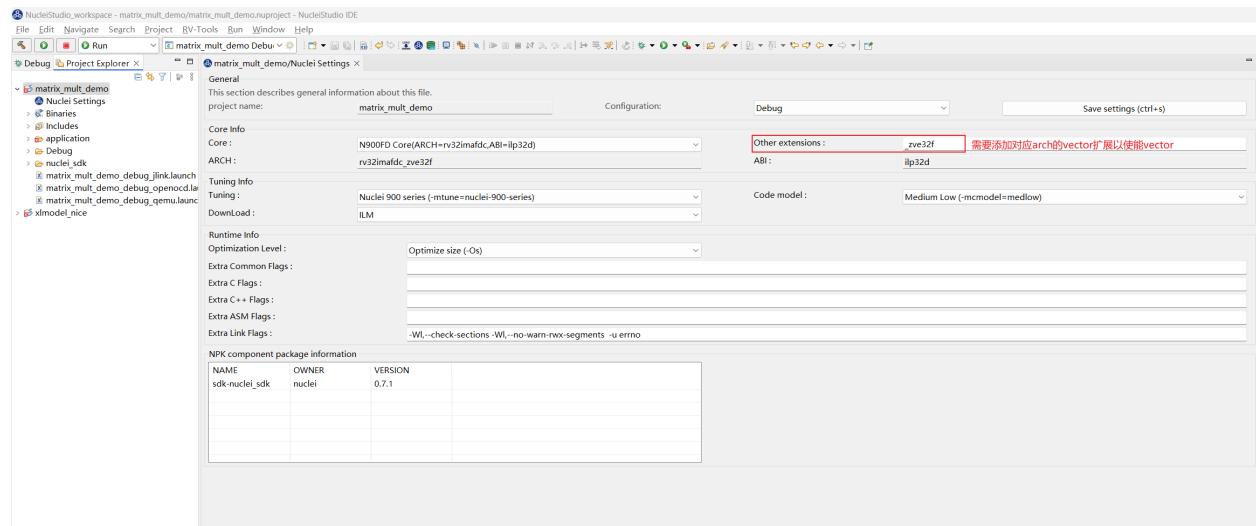
```

VNICE指令实现
VNICE指令cycle标定

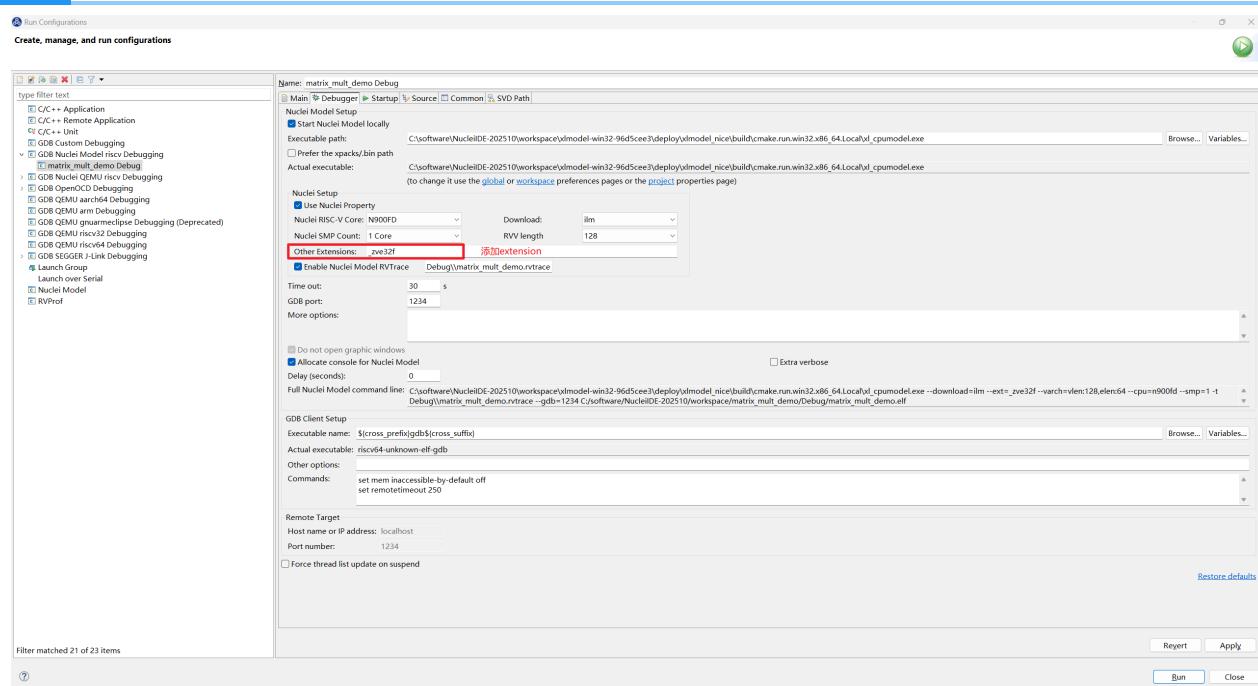
再重新编译 xlmodel_nice 保证编译通过。

step3：Nuclei Model重新运行程序

因为 VNICE 指令的输入输出均为 vector 寄存器，需要配置应用程序的 Nuclei Settings，使能对应 ARCH 的 vector 扩展，这里针对 rv32imafdc 添加 _zve32f 扩展：



对应的 Nuclei Model 配置项也需要添加 --ext=_zve32f 使能 model 的 vector 功能，然后 Apply：



需要编写一个带 VNICE 指令内嵌汇编的算法函数 `vnice_case`, VNICE 内嵌汇编需要的输入输出需要写相应的 vector intrinsic API 来构造, 然后添加和 `golden_case` 的结果比对, 重新编译应用程序工程。

注意：在应用程序头文件中需要添加 `#include <riscv_vector.h>` 以使能 vector intrinsic API

```

#include <stdint.h>
#include <nuclei_soc.h>
#include <riscv_vector.h> 使用vector需要添加此头文件
...
void vnice_case(int8_t *addr_in1, int8_t *addr_in2, int32_t *addr_out, int32_t data_cnt) {
    int32_t *pint = addr_in1;
    int32_t *pin2 = addr_in2;
    int32_t *pout = addr_out;
    size_t v1;
    vint32m8_t v1in;
    size_t v2;
    vint32m8_t v2in;
    vint32m8_t vout;
    vint32m8sm_t v1sm;
    vint32m8sm_t v2sm;
    vint32m8sm_t voutsm;

    for (; v1 < data_cnt && v2 < data_cnt; v1 += v1, v2 += v2) {
        v1in = riscv_vle32_v_i32m8(data_cnt) > 0; data_cnt -= v1;
        v2in = riscv_vle32_v_i32m8(data_cnt) > 0; data_cnt -= v2;
        v1sm = matrix_multiply_4x4_asmvn(v1in, v1);
        v2sm = matrix_multiply_4x4_asmvn(v2in, v2);
        voutsm = matrix_multiply_4x4_asmvn(v1sm, v2sm);
        riscv_vse32_v_i32m8(pout, vout, v1);
        pout += v1;
        v1 += v1;
        v2 += v2;
    }
}

```

重新 Run 应用程序, 可以发现 `vnice_case` 和 `golden_case` 输出结果一致, 其指令数和 cycle 数相对 `nice_case` 进一步大幅下降了, 构想的 VNICE 指令实现正确, 并利用了 vector 的高并行度加速了矩阵乘加算法。

```

96     end_cycle = __get_rv_cycle();
97
98     instret_vnice = end_instret - begin_instret;
99     cycle_vnice = end_cycle - begin_cycle;
100
101    for (int i = 0; i < ARRAY_CNT; i++) {
102        for (int j = 0; j < MATRIX_SIZE; j++) {
103            PRINT_DEBUG("%d", res_vniced[i * 16 + j]);
104        }
105        PRINT_DEBUG("\r\n");
106    }
107
108    PRINT_DEBUG("5. Compare normal-nice results: ");
109    if ((compare_result(res_golden, res_nice, DATA_CNT) == 0) {
110        PRINT_DEBUG("PASS\r\n");
111    } else {
112        PRINT_DEBUG("FAIL\r\n");
113    }
114    ret = 1;
115
116    PRINT_DEBUG("6. Compare normal-vnice results: ");
117    if ((compare_result(res_golden, res_vnice, DATA_CNT) == 0) {
118        PRINT_DEBUG("PASS\r\n");
119    } else {
120        PRINT_DEBUG("FAIL\r\n");
121    }
122
123    total_run 264856 instruction
[XUMODEL-INFO] Program simulation completed after 1.677620s
Info: [OSCI] System: Simulation stopped by user.
[XUMODEL-INFO] Total elapsed real time: 1.678819s model仿真完成标志
[XUMODEL-INFO] Press Enter to finish

```

总结

下表是实现了 NICE/VNICE 指令优化算法后的 instret/cycle 数据统计，相较于 golden_case, nice_case 优化后的性能提高了约 4 倍，vnice_case 优化后的性能提高了超过 30 倍。

instret/cycle golden_case nice_case vnice_case golden / nice golden / vnice nice / vnice

instret	2854	730	88	3.91	32.43	8.30
cycle	3844	964	122	3.99	31.51	7.90

用户通过研究现有算法的优化策略，就可以将构想快速通过 NICE Wizard 生成相关 NICE/VNICE 指令，再通过 Nuclei Studio 导入 xlmodel_nice 软件包实现指令，编写应用程序指令优化 case，就可以很快的利用 Nuclei Model 验证算法优化效果，整个测试过程只需使用 Nuclei Studio 就可以完成。

[优化后的工程下载链接](#)

[优化后的xlmodel_nice软件包](#)

Flash Programming使用案例¶

为了满足用户将编译好的二进制文件直接下载到硬件开发板的需求，Nuclei Studio 提供了 Flash Programming 功能。该功能允许用户快速、便捷地将编译好的二进制文件直接下载到硬件开发板中，极大提升了开发和调试的效率。用户只需点击一次即可完成二进制文件的下载，简化了操作流程。

解决方案¶

环境准备¶

Nuclei Studio：

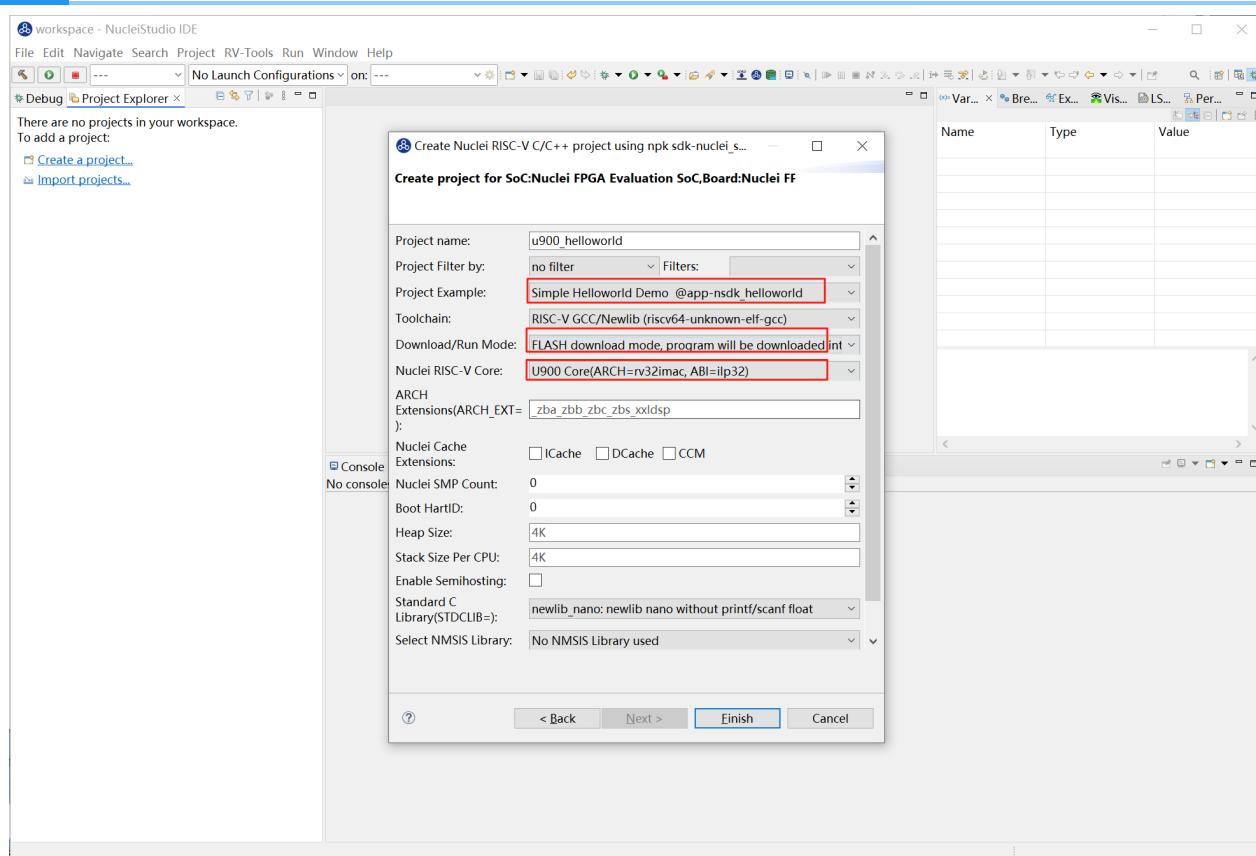
要求版本 >= 202412，下面提供202502版本。

- [NucleiStudio 202502 Windows](#)
- [NucleiStudio 202502 Linux](#)

Flash Programming 使用演示¶

step1：创建项目，烧写bit

使用0.7.1版本的sdk-nuclei_sdk创建一个u900的helloworld项目，依次选择Simple Helloworld Demo,FLASH下载模式和U900 Core，点击Finish。

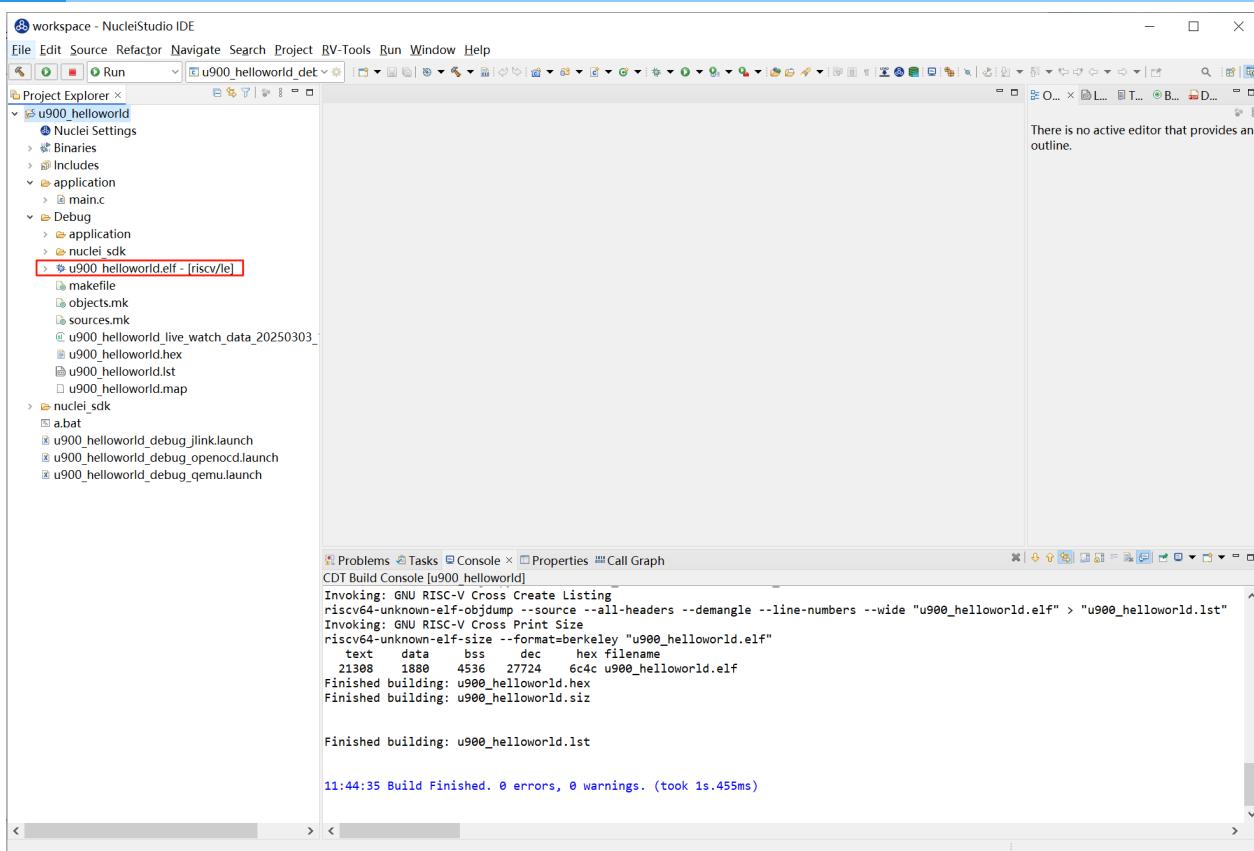


开发板烧写对应的bit即可，这里我们使用trace-

u900_best_config_ku060_16M_e85631d489_e82e2771f_202409232110_v3.12.0.bit

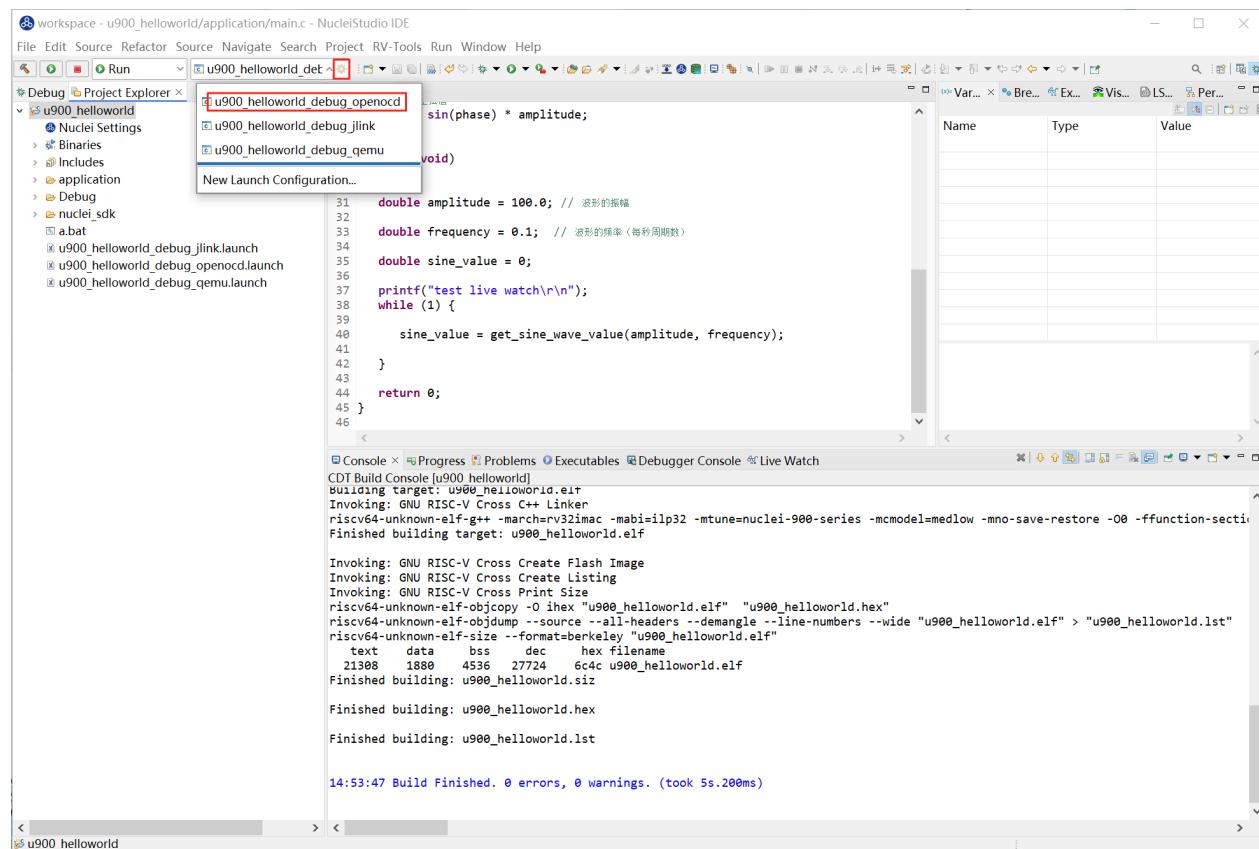
step2：配置编译 Nuclei SDK 原始工程

编译原始工程，确保编译成功以及在 Debug 下可以找到生成的 elf 文件：



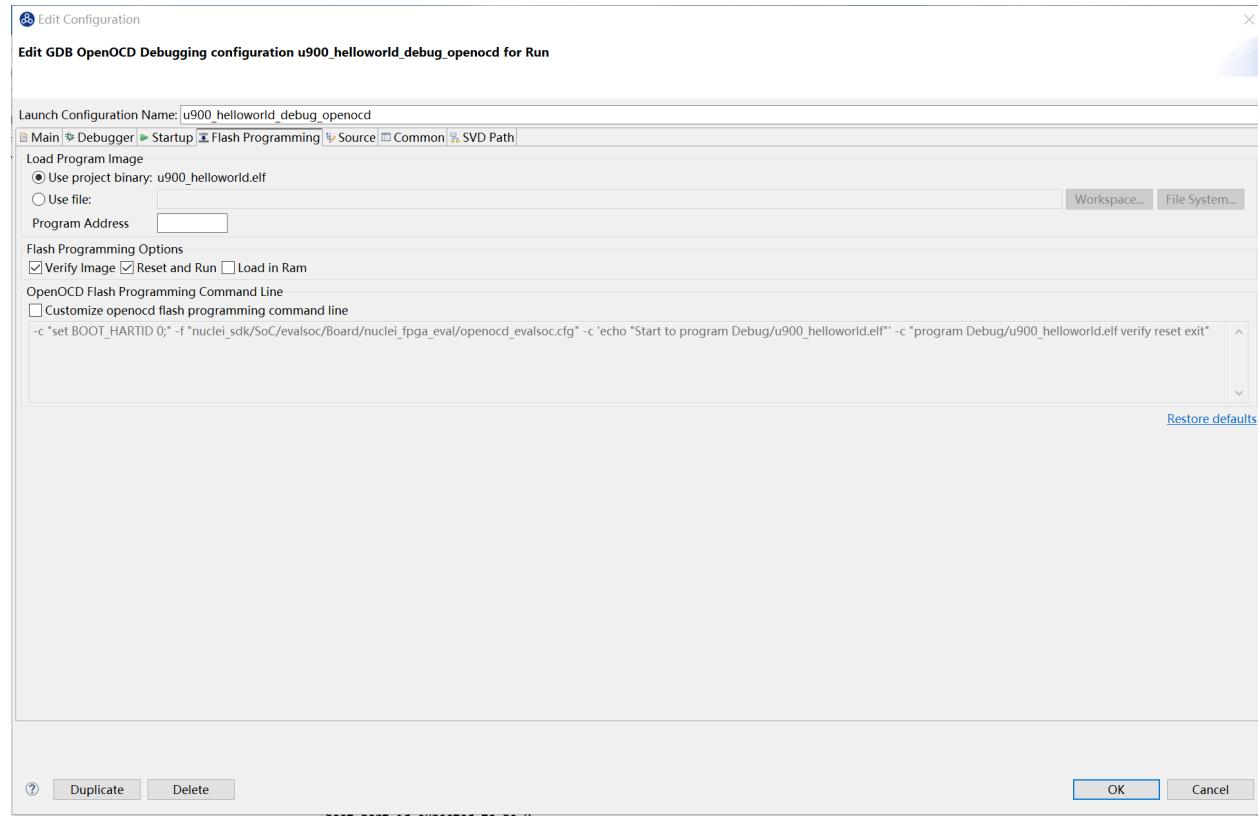
step3：配置Flash Programming选项卡

在Launch Configuration 选中对应调试选项(openocd)，点击edit打开配置页面。



选择 Flash Programming 选项卡，进入配置页面。

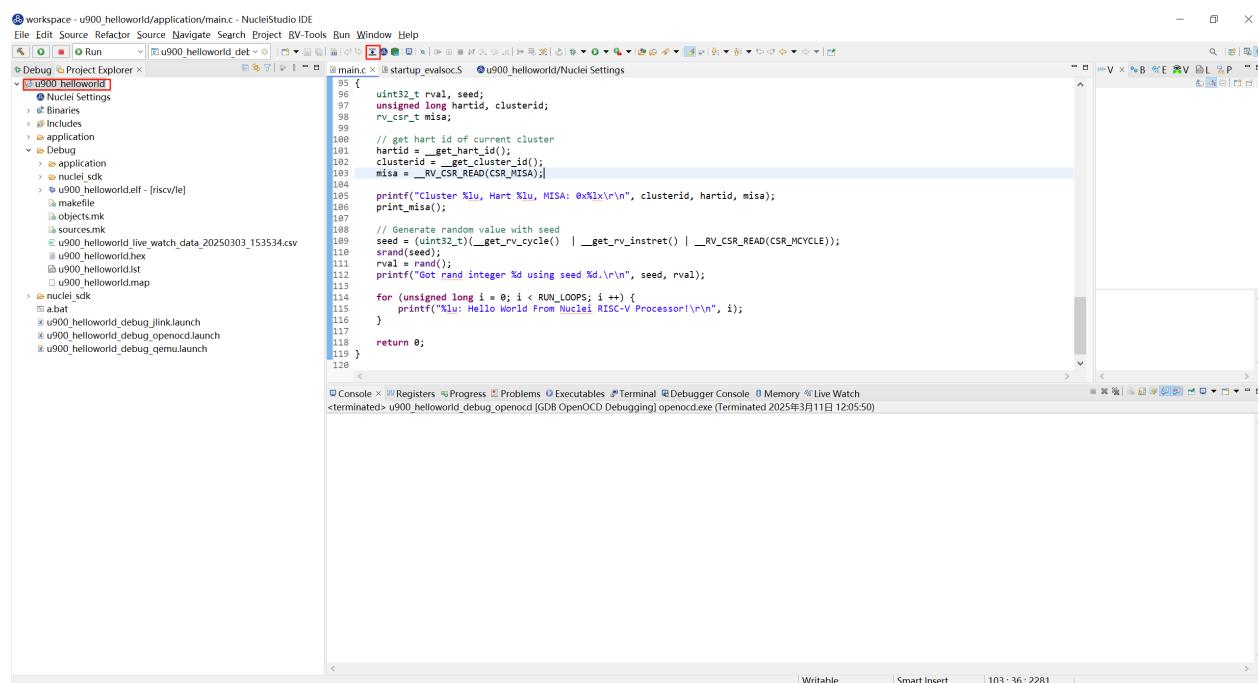
由于是Flash下载模式，这里默认选择的verify image和reset and run即可。



具体配置项内容可参考[Nuclei Development Tool Guide](#)

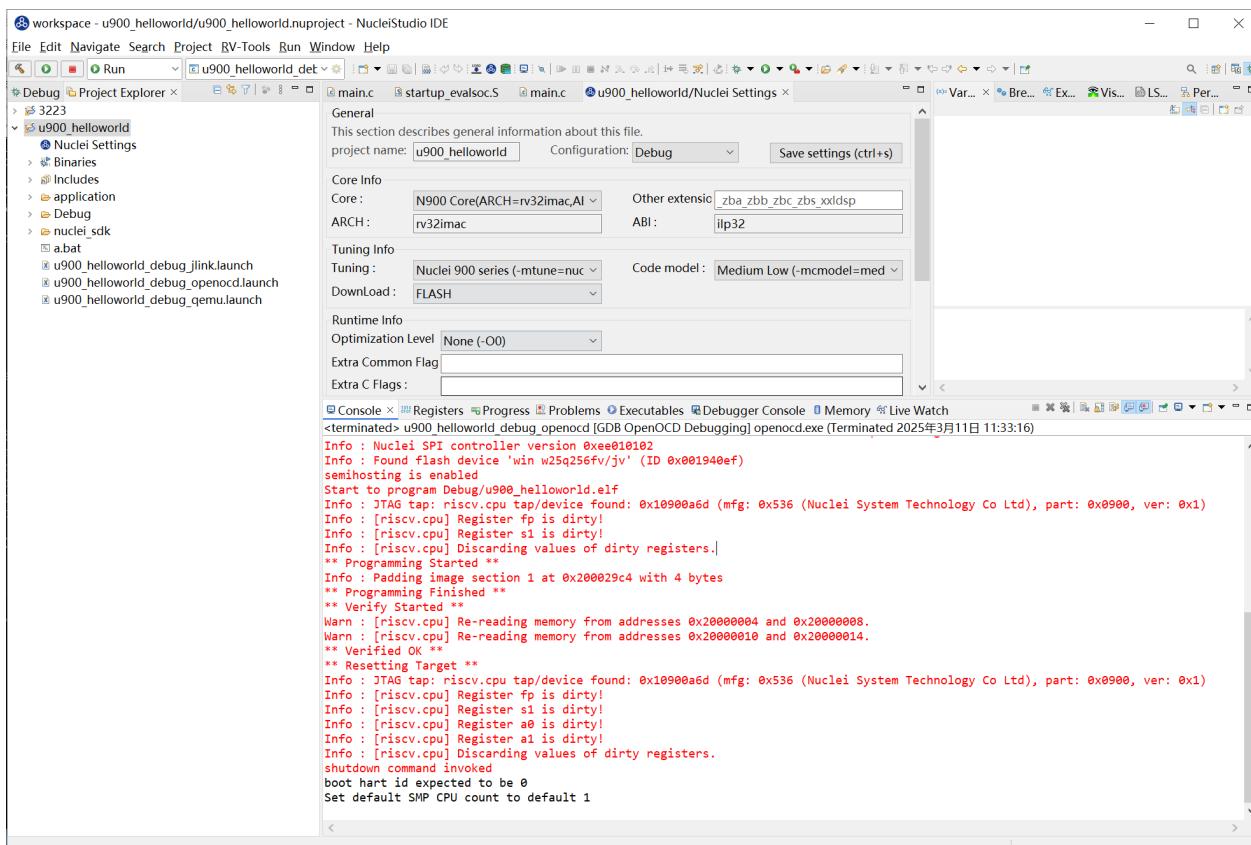
step4：下载

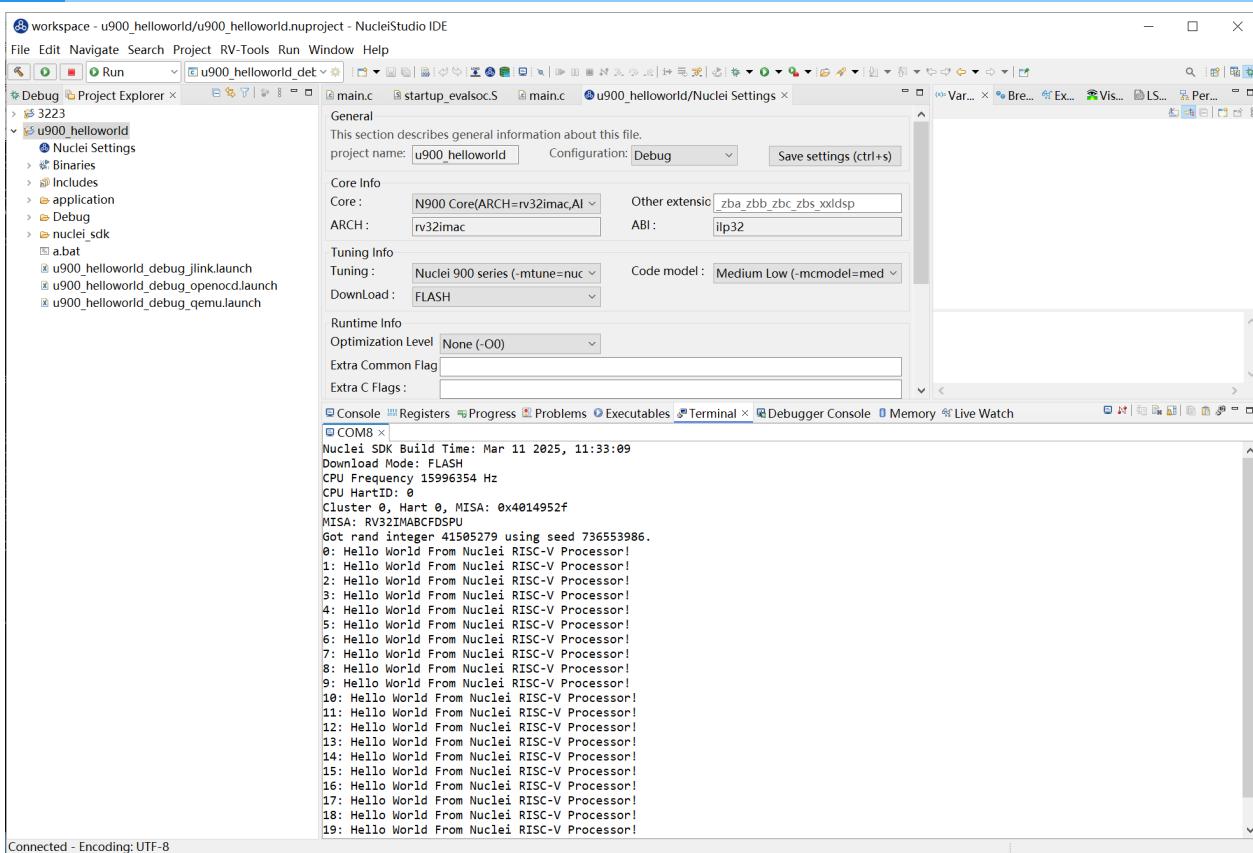
选中项目，点击Flash Programming，下载二进制文件到硬件开发板。



下载成功后，用户可以在 Console 中看到下载结果，确认二进制文件已成功烧录到硬件中。

```
** Programming Started **
Info : Padding image section 1 at 0x200029c4 with 4 bytes
** Programming Finished **
** Verify Started **
Warn : [riscv.cpu] Re-reading memory from addresses 0x20000004 and
0x20000008.
Warn : [riscv.cpu] Re-reading memory from addresses 0x20000010 and
0x20000014.
** Verified OK **
** Resetting Target **
Info : JTAG tap: riscv.cpu tap/device found: 0x10900a6d (mfg:
0x536 (Nuclei System Technology Co Ltd), part: 0x0900, ver: 0x1)
Info : [riscv.cpu] Register fp is dirty!
Info : [riscv.cpu] Register s1 is dirty!
Info : [riscv.cpu] Register a0 is dirty!
Info : [riscv.cpu] Register a1 is dirty!
Info : [riscv.cpu] Discarding values of dirty registers.
shutdown command invoked
```



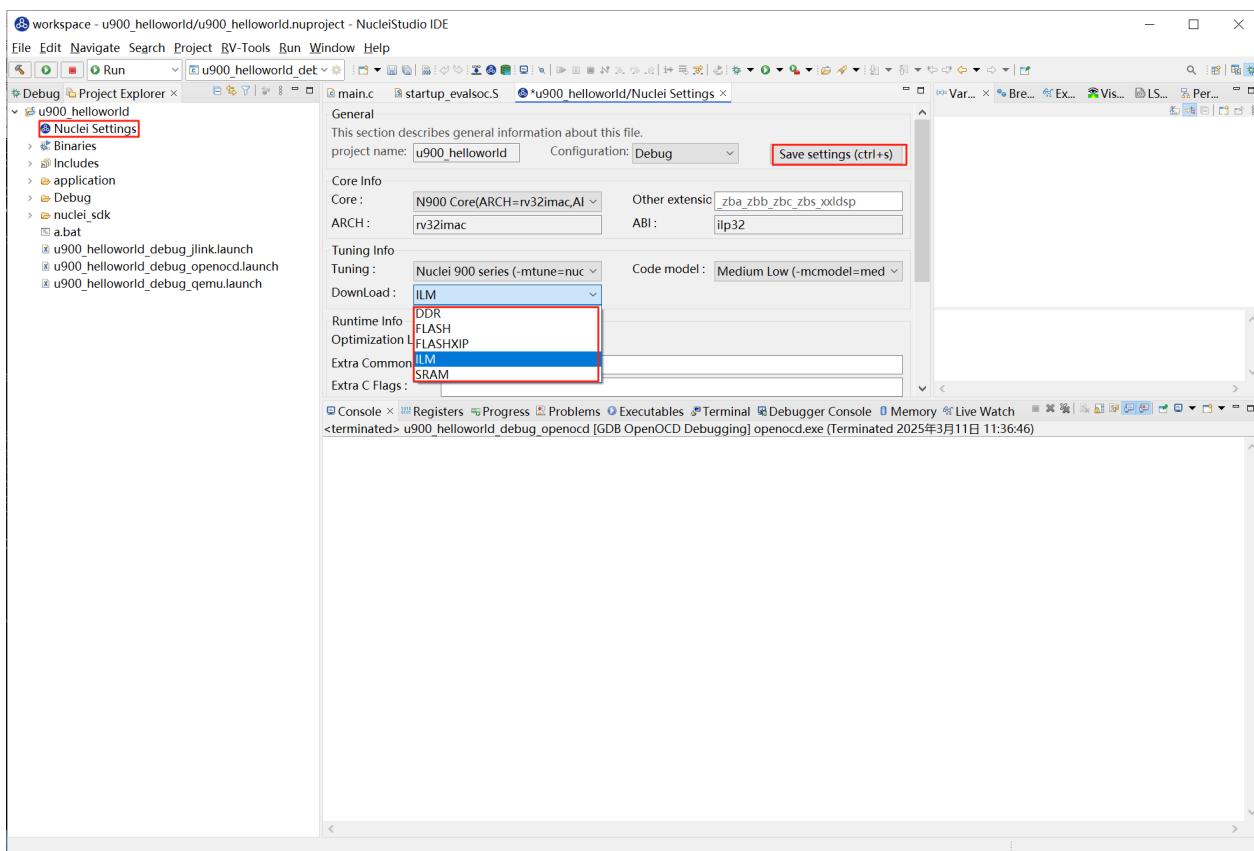


step5：下载到内存的区别

Nuclei Studio有DDR、FLASH、FLASHXIP、ILM、SRAM多种下载模式。

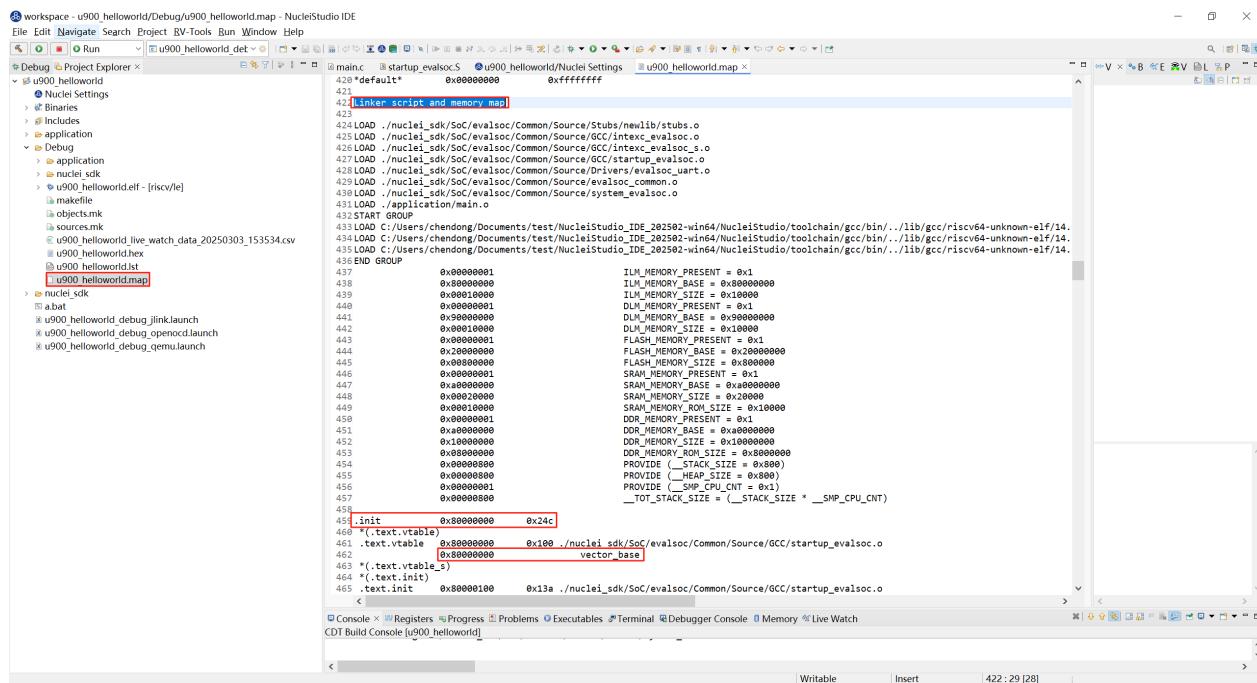
FLASH、FLASHXIP模式按上面的步骤使用即可，而DDR、ILM、SRAM是下载到内存中的与Flash有所区别，下面以ILM为例。

点击Nulcei Settings打开页面，在Download中选择ILM并保存。



重新编译项目， clean project -> build project

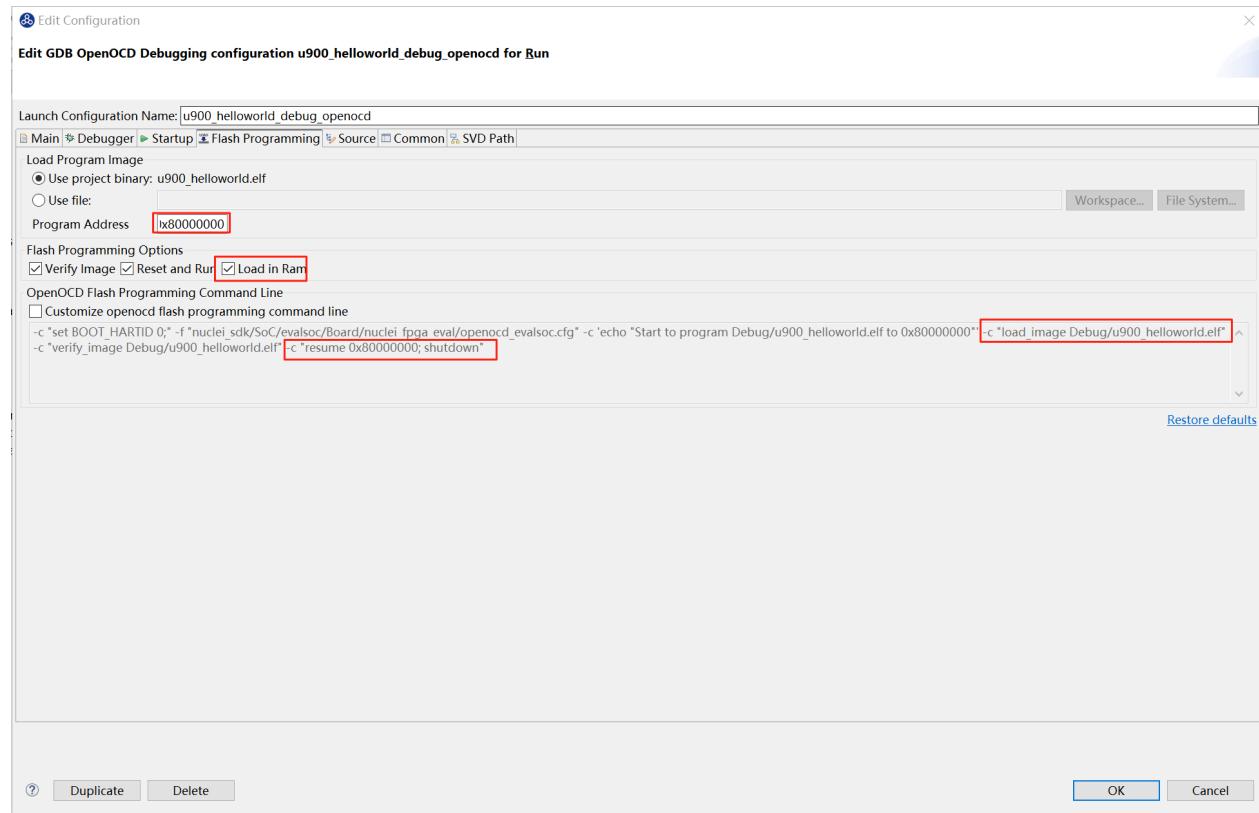
然后打开对应的.map文件，这里是u900_helloworld.map，在里面找到起始加载地址，如下图的0x80000000



打开Flash Programming选项卡，因为是下载到内存，这里要勾选Load in Ram,此时下面的command line会增加load_image命令，

再在Program Address中填入上面获取到的地址0x80000000, command line会带上 resume 0x80000000参数。

点击OK。

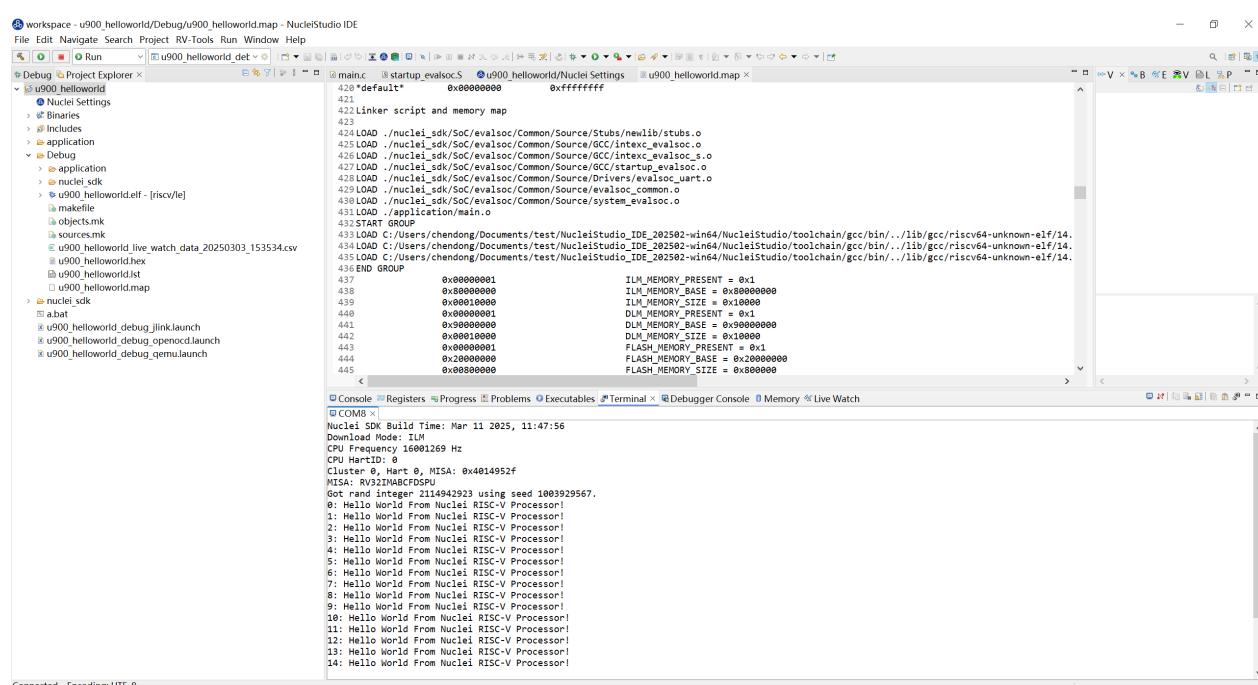
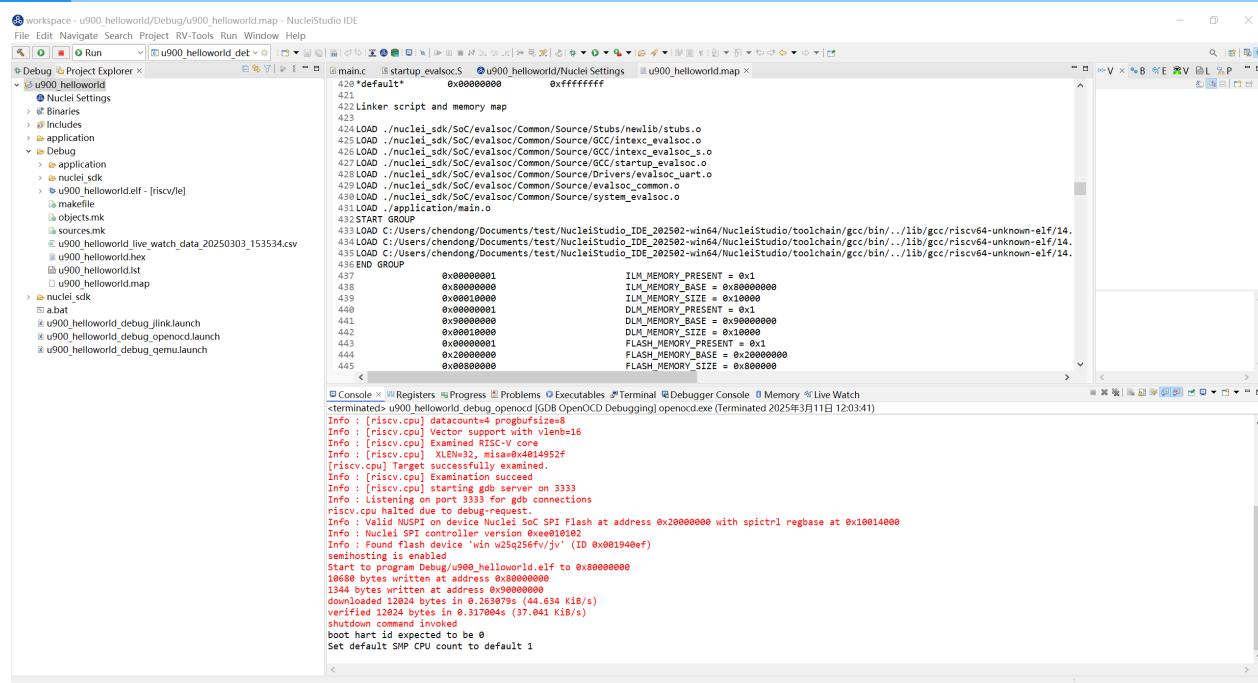


选中项目，点击Flash Programming下载。结果如下。

```

Info : Valid NUSPI on device Nuclei SoC SPI Flash at address
0x20000000 with spictrl regbase at 0x10014000
Info : Nuclei SPI controller version 0xee010102
Info : Found flash device 'win w25q256fv/jv' (ID 0x001940ef)
semihosting is enabled
Start to program Debug/u900_helloworld.elf to 0x80000000
10680 bytes written at address 0x80000000
1344 bytes written at address 0x90000000
downloaded 12024 bytes in 0.263079s (44.634 KiB/s)
verified 12024 bytes in 0.317004s (37.041 KiB/s)
shutdown command invoked

```



step6：可能出现的问题

1. Error: checksum mismatch , attempting binary compare

出现这个错误是因为flash下载和ram下载搞错了，需要在nuclei settings里面进行修改Download模式。

总结

Flash Programming 功能为用户提供了一种快速、便捷的方式将编译好的二进制文件下载到硬件开发板中。通过简单的配置，用户可以轻松适配不同的硬件环境，并确保二进制文件的正确烧录。

Live Watch 功能的使用¶

Live Watch 是一款强大的实时监控工具，专为开发者设计，旨在帮助您更高效地调试和优化代码。通过 Live Watch，您可以即时查看程序运行过程中变量的变化情况，无需打断执行流程或手动添加日志语句。在 Nuclei Studio 2025.02 版中实现了 Live Watch 功能，它支持自动刷新变量值，确保始终看到最新的数据变化。直观的图形化界面，能轻松管理需要监控的变量。

背景描述¶

Live Watch 功能依赖 Nuclei OpenOCD >= 2025.02 版本，并且仅支持 Nuclei CPU 配置了 RISC-V SBA 功能。通过 Live Watch，开发者可以在调试过程中实时监控变量的变化，帮助快速定位问题并优化代码性能。

解决方案¶

环境准备¶

Nuclei Studio：

- [NucleiStudio 202502 Windows](#)
- [NucleiStudio 202502 Linux](#)

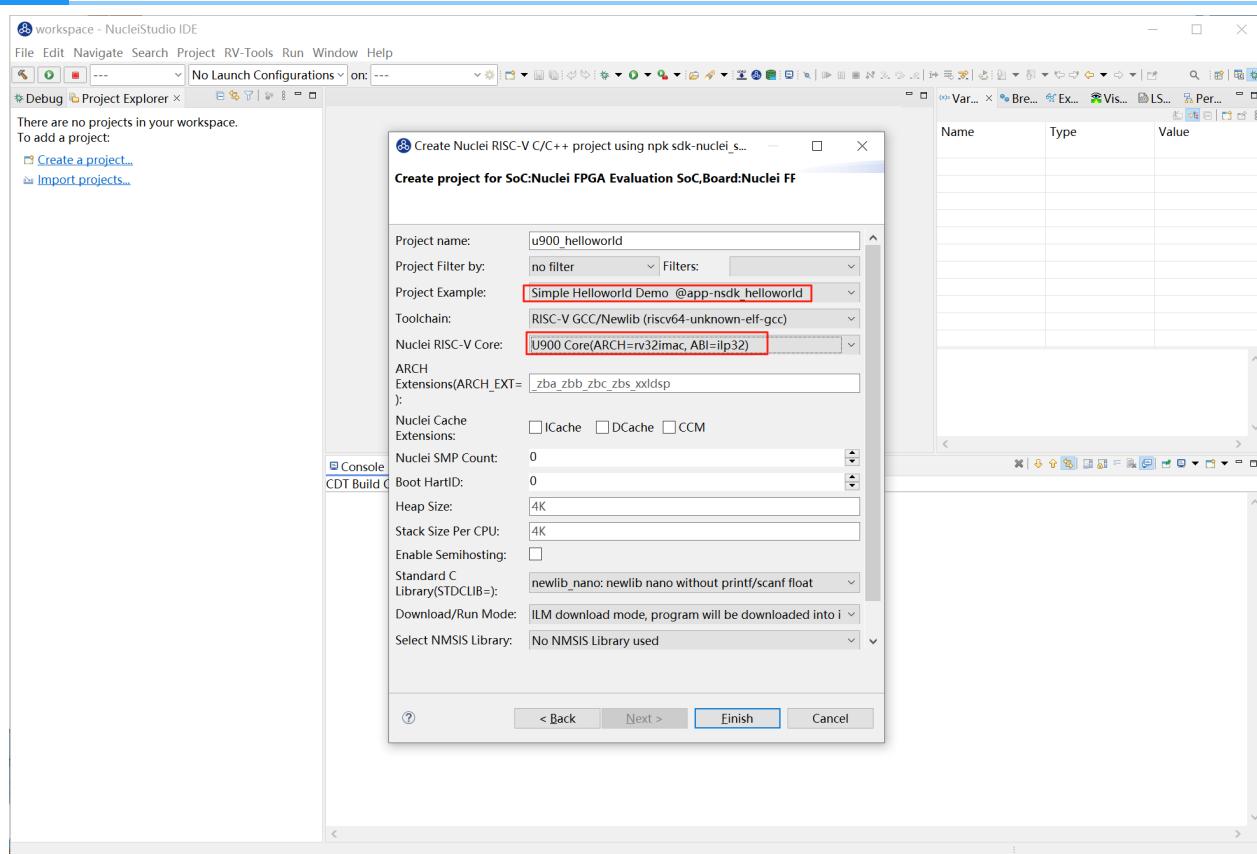
Nuclei OpenOCD：

- 确保安装的 OpenOCD 版本 >= 2025.02，并且支持 RISC-V SBA 功能。

Live Watch 使用演示¶

step1：创建项目，烧写bit

使用0.7.1版本的sdk-nuclei_sdk创建一个u900的helloworld项目，依次选择Simple Helloworld Demo和U900 Core，点击Finsh。

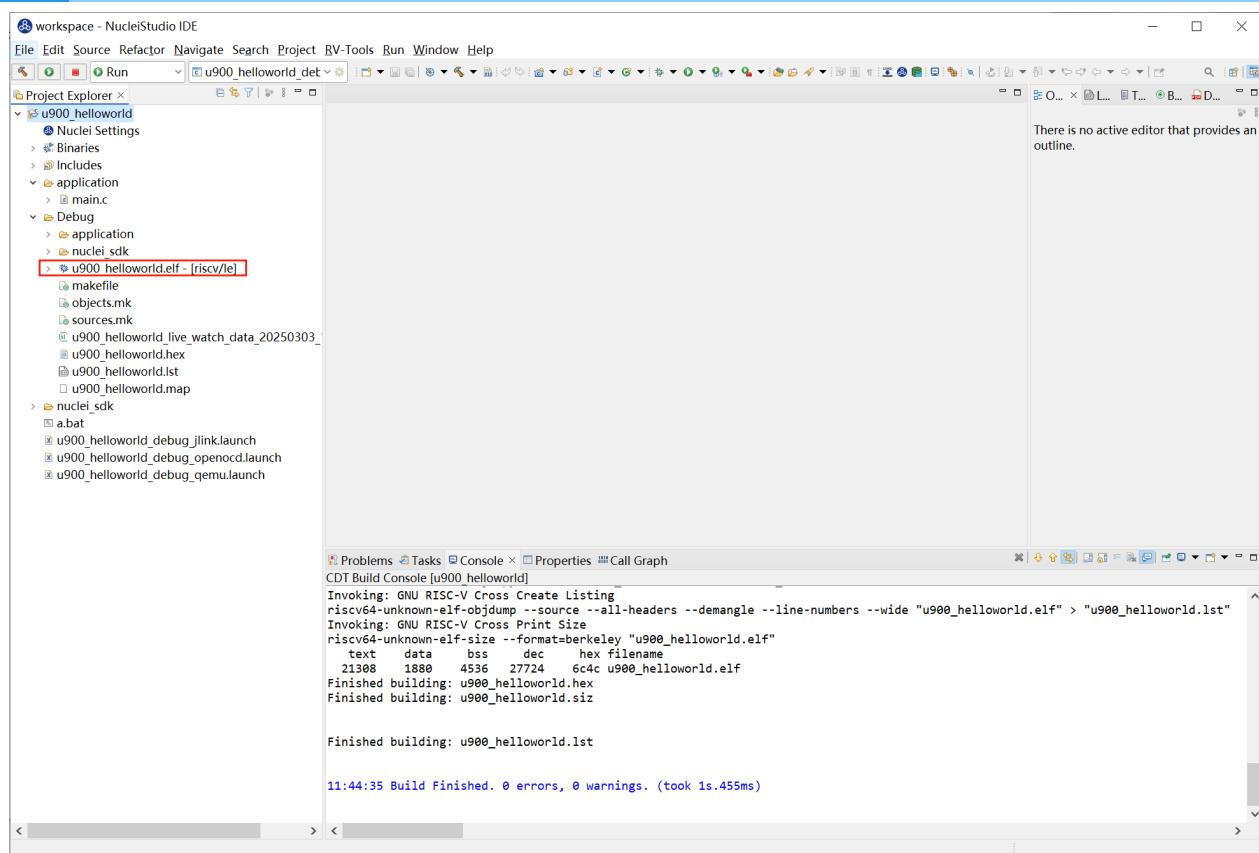


开发板烧写对应的bit即可，这里我们使用trace-

u900_best_config_ku060_16M_e85631d489_e82e2771f_202409232110_v3.12.0.bit

step2：编译 Nuclei SDK 原始工程

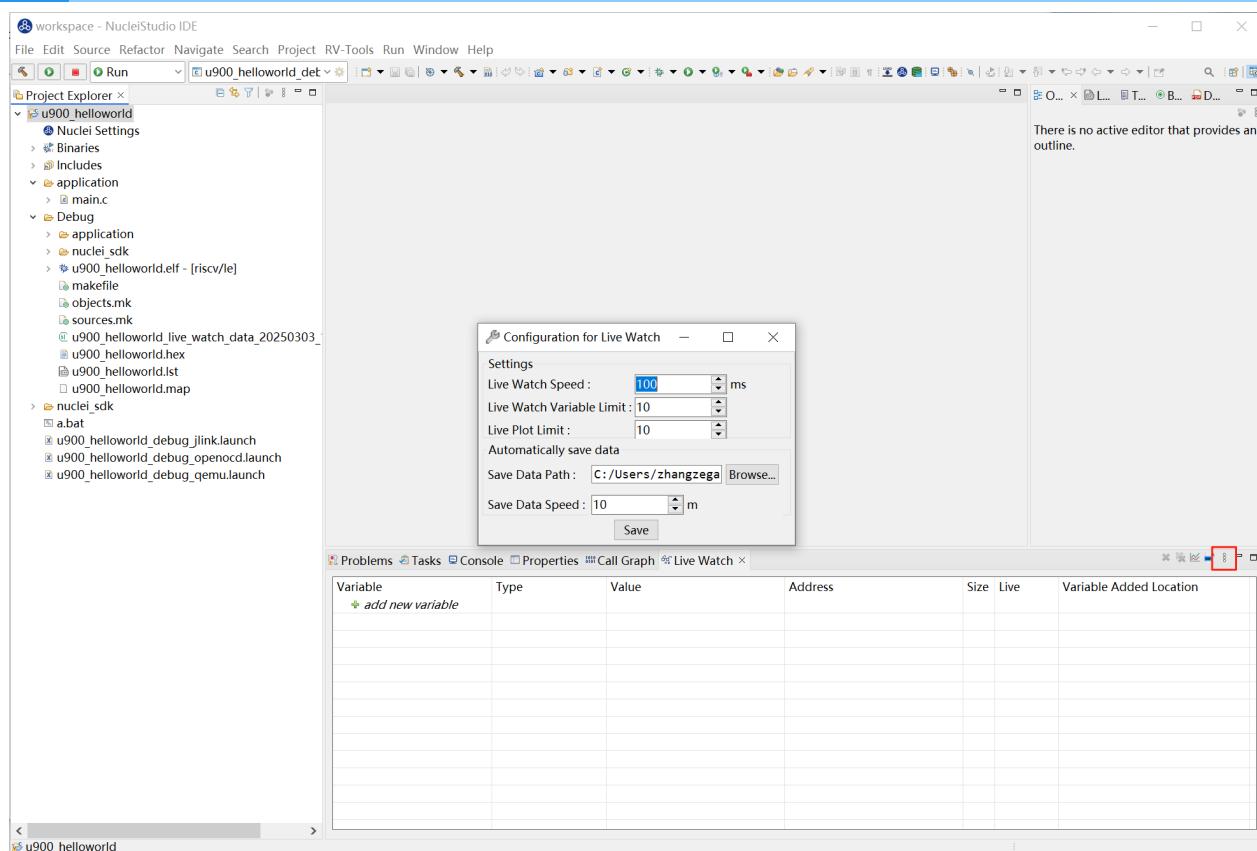
编译原始工程，确保编译成功以及在 Debug 下可以找到生成的 elf 文件：



step3：打开 Live Watch 视图

打开 Live Watch 视图，找到 Live Watch Settings 并根据需要设置相关参数，这里我们直接使用默认值。

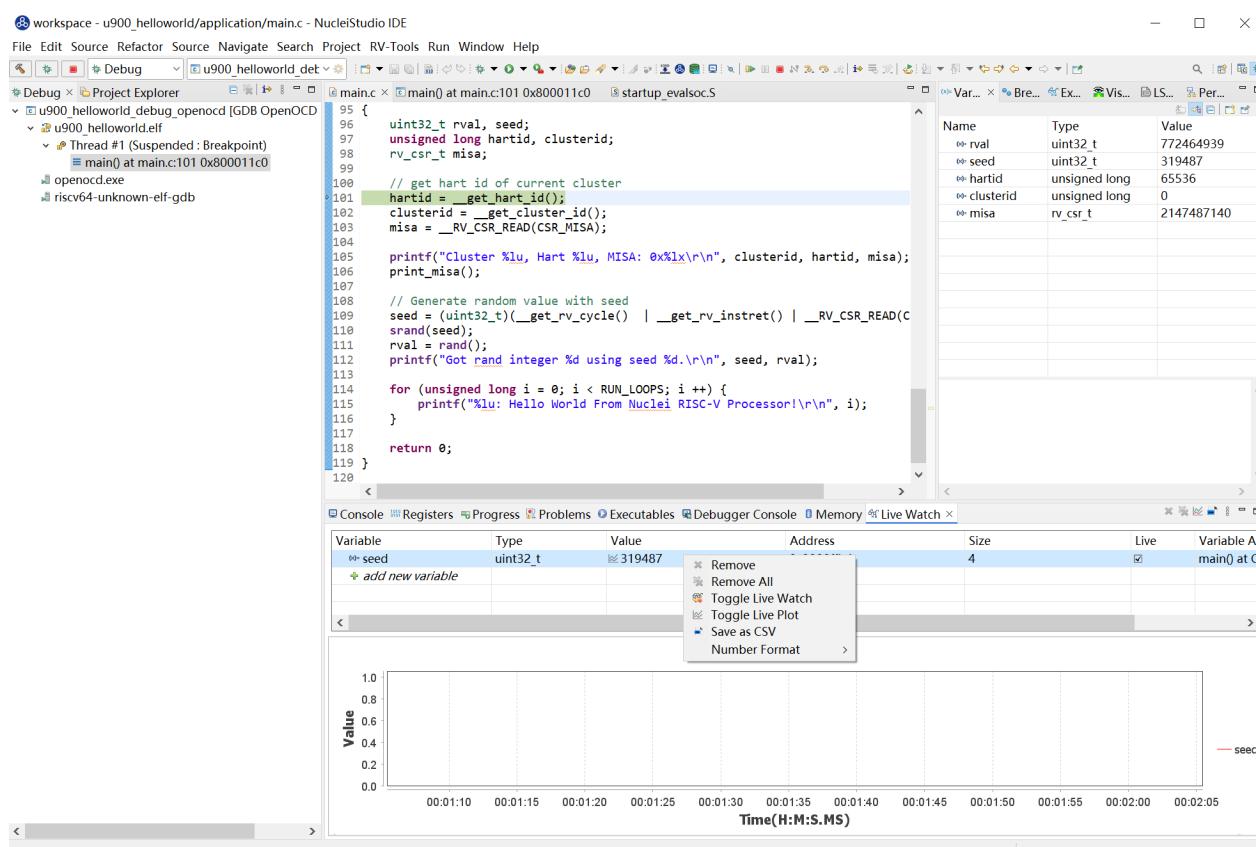
如需配置可参考下图或[Nuclei Development Tool Guide](#)



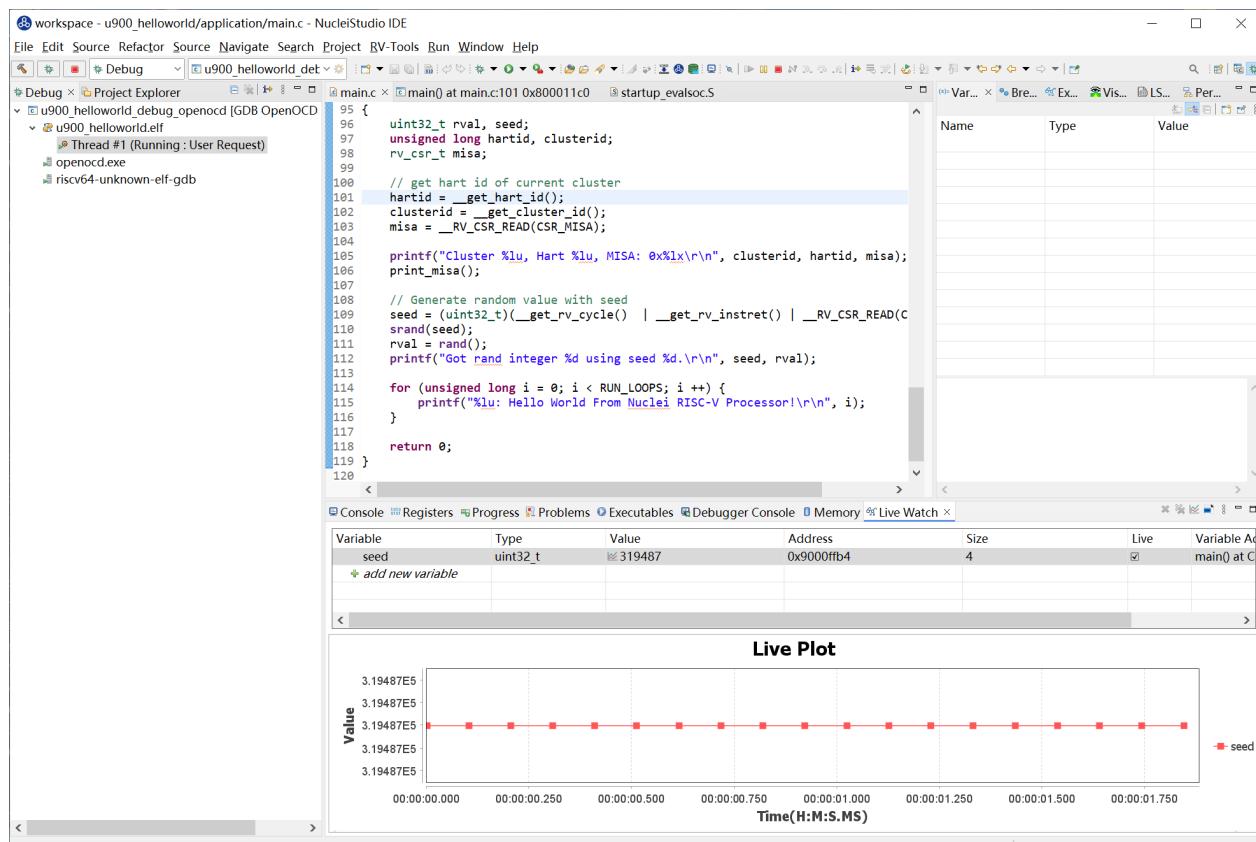
step4：运行Nuclei SDK原始工程

Debug运行程序，在Live Watch视图中添加需要查看的变量seed。

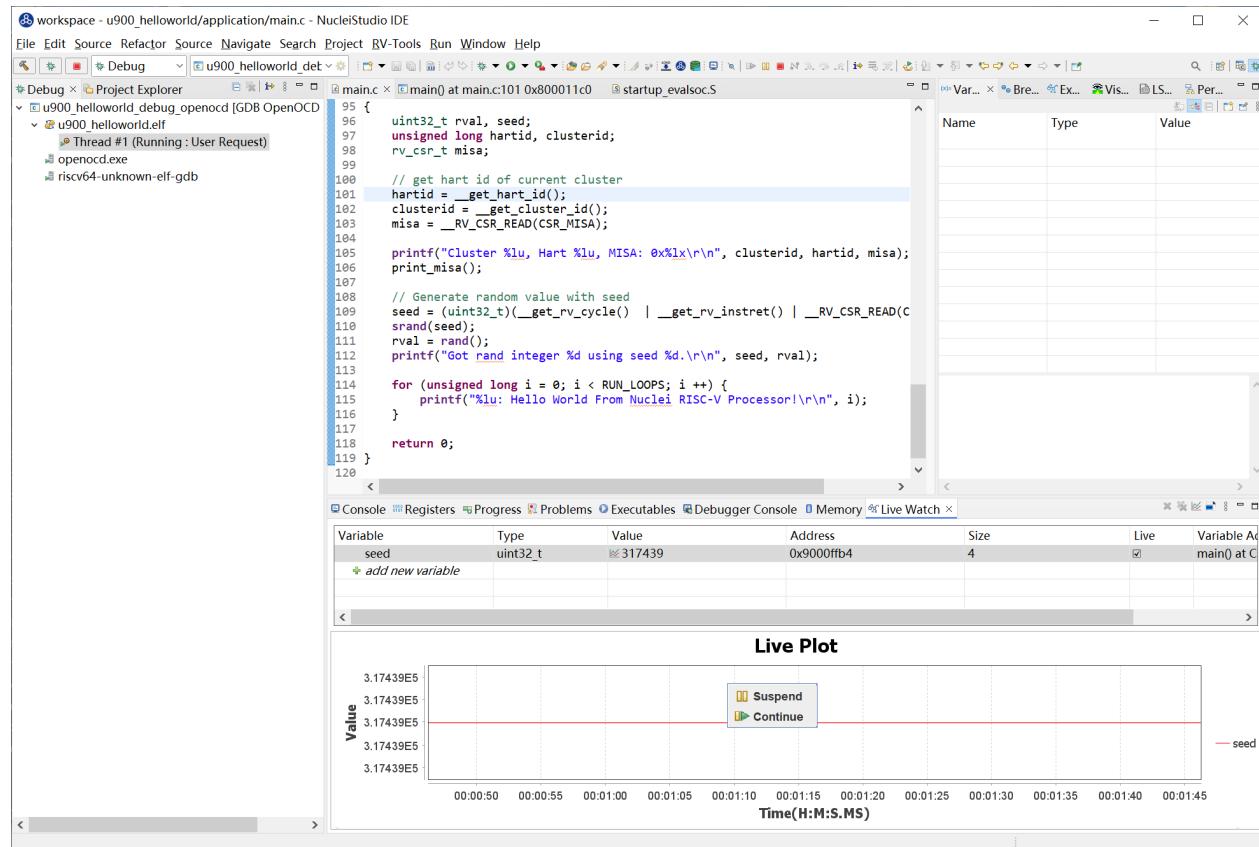
这里想要通过Live Plot查看变量的变化曲线，选中该条记录，并点击鼠标右键，在弹出的菜单中选中 Toggle Live Plot ,Live Plot工具就会弹出。



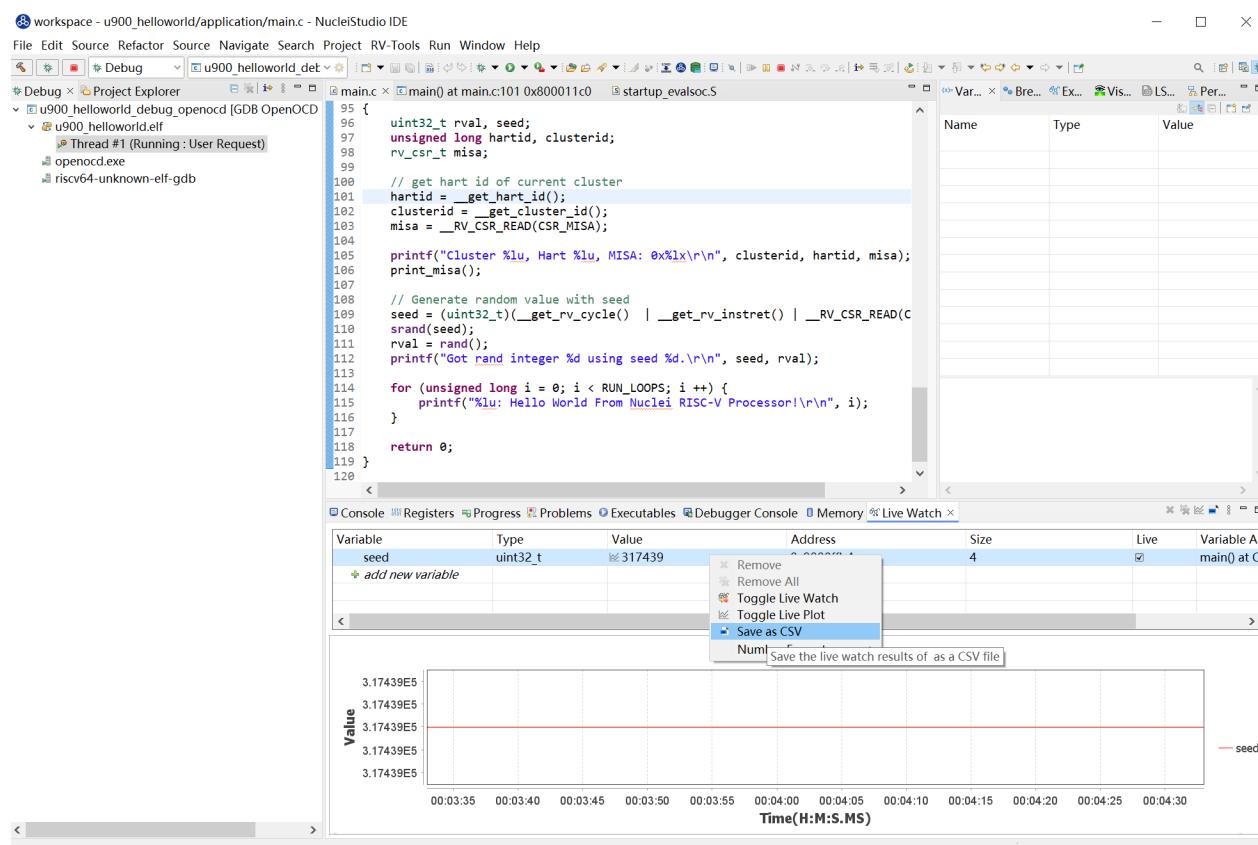
让工程全速运行时，可以看到变量的值，以设定的Live Watch Speed变化，Live Plot绘制的曲线图如下。



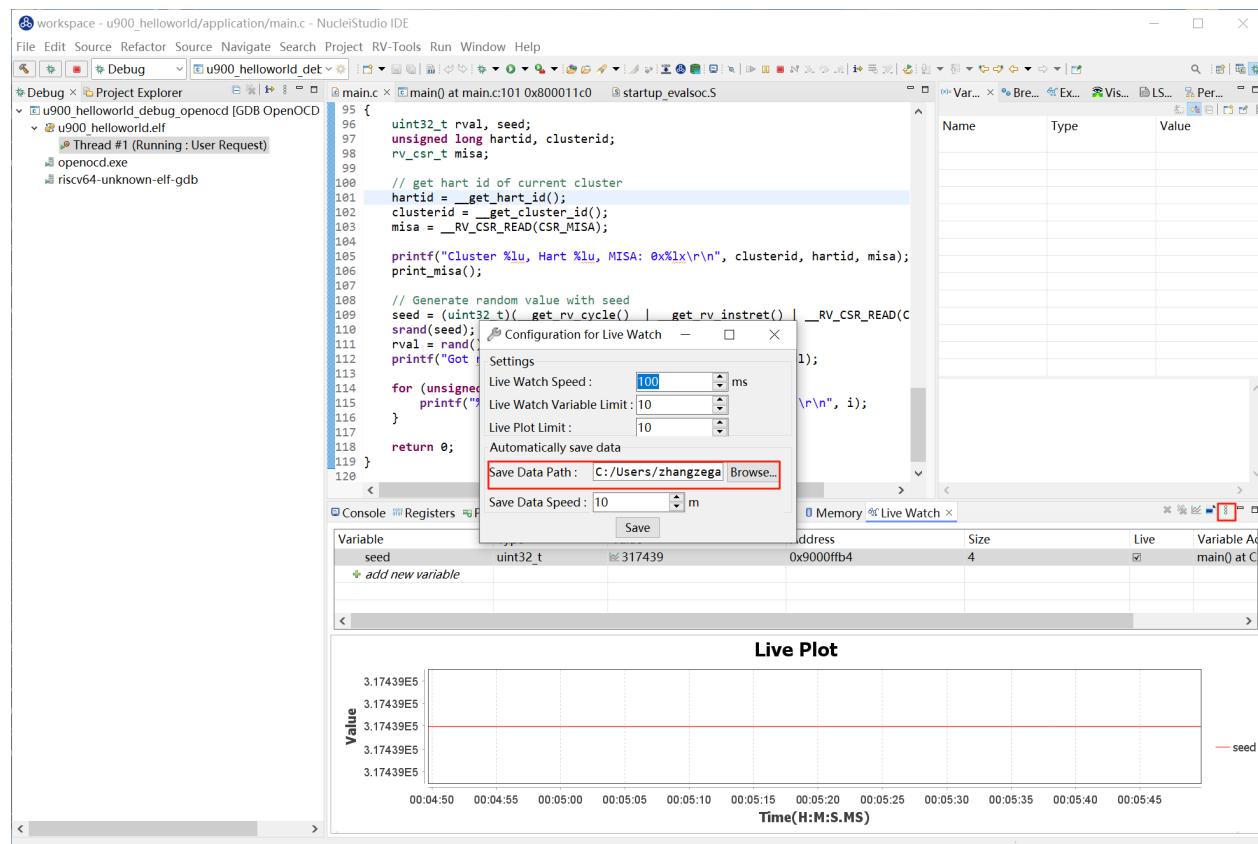
当随着时间数据节点越来越多时，届时会隐藏数据节点。用户可以在Live Plot右键点击Suspend暂停，然后通过滚动鼠标放大曲线，放大到一定倍数会显示节点，鼠标移至节点可查看数据详情；点击 Continue Live Plot则继续绘制曲线。



选中seed行,点击鼠标右键, 将该变量的结果存为CSV格式文件, 用来查阅和使用。



Live Watch也会自动将查询到的数据结果保存到 Save Data Path 中，可以在Save Data Path对应地址找到对应的CSV格式的数据文件。



电脑 > Windows (C) > 用户 > [REDACTED] workspace > u900_helloworld > Debug

名称	修改日期	类型
application	2025/3/10 11:44	文件夹
nuclei_sdk	2025/3/10 11:44	文件夹
makefile	2025/3/10 11:44	文件
objects.mk	2025/3/5 12:06	Makefile 源文件
sources.mk	2025/3/5 12:04	Makefile 源文件
u900_helloworld.elf	2025/3/10 11:44	ELF 文件
u900_helloworld.hex	2025/3/10 11:44	HEX 文件
u900_helloworld.lst	2025/3/10 11:44	LST 文件
u900_helloworld.map	2025/3/10 11:44	MAP 文件
u900_helloworld_live_watch_data_20250303_153534.csv	2025/3/3 15:39	XLS 工作表

总结

Live Watch 功能为开发者提供了一个强大的实时监控工具，极大地提升了调试效率和代码优化的能力。通过合理使用 Live Watch，开发者可以更轻松地应对复杂的调试任务，提升开发效率。

在llvm中新增自定义汇编指令教程¶

以下皆以32位指令为例说明

自定义扩展名的识别¶

以下以xnice扩展为例

文件:llvm/lib/Target/RISCV/RISCVFeatures.td

添加内容：

```
def FeatureVendorXnice
  : RISCVEExtension<"xnice", 1, 0,
    "'Xnice' (Xnice extension)">;
def HasVendorXnice
  : Predicate<"Subtarget->hasVendorXnice()">,
  AssemblerPredicate<(all_of FeatureVendorXnice),
    "'Xnice' (Xnice extension)">;
```

注意：RISCVEExtension处第一个xnice为实际llvm编译器可识别的扩展名，后面的1，0为该扩展的版本号，第二个Xnice只是用于扩展功能描述

自定义汇编指令识别¶

以下以新增一条标准R类型nice指令为例

1、添加对应解码器

文件:llvm/lib/Target/RISCV/Disassembler/RISCVDisassembler.cpp

函数:DecodeStatus RISCVDisassembler::getInstruction32()

内容：

```
TRY_TO_DECODE_FEATURE(RISCV::FeatureVendorXnice,
DecoderTableXnice32,
  "Xnice extension");
```

2、创建编解码文件

在`llvm/lib/Target/RISCV/`下创建一个`RISCVInstrInfoXnice.td`的编解码文件，并在`llvm/lib/Target/RISCV/RISCVInstrInfo.td`中将该文件include进来

```
include "RISCVInstrInfoXnice.td"
```

3、指令编码

假设该nice指令汇编格式为`nice rd, rs1, rs2`, 并且使用的是RISC-V预留的custom3区域的编码空间，则编码步骤如下：

- 新建一个`XniceInstr`的类用于说明XNICE扩展的所有指令的统一格式，由于是R类型指令，所以可以直接从llvm中预先写好的`RVInstR`类继承而来，否则需要继承其他相匹配的类或者继承基类重新写一个指令类出来， llvm中所有指令类的声明位于`llvm/lib/Target/RISCV/RISCVInstrFormats.td`
- 通过`Predicates`限定nice指令所在的扩展（`[HasVendorXnice]`）以及使用的解码器（`DecoderNamespace = "Xnice"`）
- 通过`def`新增一个指令说明，只需要通过`XniceInstr`以及填充该类在声明时缺少的参数即可完成一条指令的编码，例如自定义的R类型指令只需要再次声明`func7`, `func3`以及汇编指令名

完整示例如下：

```
let hasSideEffects = 0, mayLoad = 0, mayStore = 0 in {
  class XniceInstr<bits<7> funct7, bits<3> funct3, string opcodestr>
    : RVInstR<funct7, funct3, OPC_CUSTOM_3, (outs GPR:$rd),
      (ins GPR:$rs1, GPR:$rs2), opcodestr, "$rd, $rs1,
$rs2">;
}

let Predicates = [HasVendorXnice], DecoderNamespace = "Xnice" in {
  def NICE : XniceInstr<0b1111010, 0b001, "nice">;
}
```

`OPC_CUSTOM_3`是`llvm/lib/Target/RISCV/RISCVInstrFormats.td`中已经预留的宏，如果使用的其他编码空间，则可以直接查找更改

对于`def`后大写的`NICE`一般用于`intrinsic`或者自定向量化等调用，只做汇编时可以只给汇编指令名的大写格式用于区分

另外，以上示例中没有限制指令在RV32/64下的使用场景，因此RV32/64下都可识别，如果需要限定只在RV32下使用，则需要额外在`Predicates`中指定扩展时同时进行限定，例如
`[HasVendorXxlczbitop, IsRV32]`

使用说明¶

使用时与GCC一样，只需要将`xniced`通过`-march`选项传递给llvm编译器即可，例如`-march=rv32imafdc_xniced`

参考链接¶

llvm table gen语法手册 <https://llvm.org/docs/TableGen/ProgRef.html>

PLCT关于在LLVM中添加RISC-V的自定义指令的示例

<https://www.bilibili.com/video/BV1JR4y1J7he>

nuclei自定义vpu指令的扩展识别及汇编实现

<https://github.com/riscv-mcu/llvm-project/commit/f5d025b9800f3cd662e93c11eb7c7b0f65ca4472>

如何使用芯来提供的DebugMap寄存器分析错误现场¶

首先需要确定硬件支持DebugMap功能¶

core的顶层有一个信号叫做dm_map_enable，这个信号接1表示使能DebugMap功能

详情参见 *Nuclei_CPU_Debug_Function_Specification.pdf* 文档的 *Debug Control Interface* 部分内容

什么是DebugMap寄存器¶

DebugMap功能就是当Core被hang的时候可以通过OpenOCD查看core内部的状态，将若干内部状态映射到DM寄存器中，目前只实现了下面三个状态的映射：

- 00: Commit PC(i0 for dual issue)
- 16: ICache miss address(ICache is supported)
- 32: DCache address waiting for retire(DCache is supported)

详情参见 *Nuclei_CPU_Debug_Function_Specification.pdf* 文档的 *CFR0 (Custom Feature Register0)* 部分内容

OpenOCD里DebugMap的输出信息¶

在使用OpenOCD连接FPGA/芯片时，经常会看到类似下面这样的输出信息：

```
Info : coreid=0, nuclei debug map reg 00: 0xa00003ac, 16:  
0xa0003240, 32: 0x10003014
```

- coreid 表示当前输出的是哪个core的debug-map信息

可能出现的错误现场¶

错误现场一：

```
Info : Using libusb driver  
Info : clock speed 1000 kHz  
Info : JTAG tap: riscv.cpu tap/device found: 0x10900a6d (mfg:
```

```

0x536 (Nuclei System Technology Co Ltd), part: 0x0900, ver: 0x1)
Info : [riscv.cpu] datacount=4 progbufsize=8
Info : coreid=0, nuclei debug map reg 00: 0xa0000496, 16:
0xa0003140, 32: 0x10002ff8
Error: [riscv.cpu] Unable to halt hart 0. dmcontrol=0x00000001,
dmstatus=0x00400ca2
Error: [riscv.cpu] Fatal: Hart 0 failed to halt during examine()
Warn : target riscv.cpu examination failed
Info : [riscv.cpu] datacount=4 progbufsize=8
Error: Hart 0 doesn't exist.
Error: Fatal: Failed to read s0 from hart 0.
Info : [riscv.cpu] datacount=4 progbufsize=8
Error: Hart 0 doesn't exist.
Error: Fatal: Failed to read s0 from hart 0.
Info : starting gdb server for riscv.cpu on 22800
Info : Listening on port 22800 for gdb connections
Error: Target not examined yet

```

错误现场二：

```

Info : libusb_open() failed with LIBUSB_ERROR_NOT_FOUND
Info : no device found, trying D2xx driver
Info : D2xx device count: 2
Info : Connecting to "(null)" using D2xx mode...
Info : clock speed 1000 kHz
Info : JTAG tap: riscv0.cpu tap/device found: 0x10300a6d (mfg:
0x536 (Nuclei System Technology Co Ltd), part: 0x0300, ver: 0x1)
Info : [riscv0.cpu] datacount=4 progbufsize=2
Info : coreid=0, nuclei debug map reg 00: 0xa0000496, 16:
0xa0003140, 32: 0x10002ff8
Info : Examined RISC-V core; found 1 harts
Info : hart 0: XLEN=32, misa=0x40001127
[riscv0.cpu] Target successfully examined.
Info : starting gdb server for riscv0.cpu on 3333
Info : Listening on port 3333 for gdb connections
Started by GNU MCU Eclipse
Info : Listening on port 6666 for tcl connections
Info : Listening on port 4444 for telnet connections
Info : accepting 'gdb' connection on tcp/3333
Warn : Prefer GDB command "target extended-remote :3333" instead
of "target remote :3333"
Error: Timed out after 2s waiting for busy to go low
(abstractcs=0x2001004). Increase the timeout with riscv
set_command_timeout_sec.

```

```
Error: Abstract command ended in error 'busy'  
(abstractcs=0x2001104)
```

如何正确利用DebugMap分析错误现场¶

- 在出现Core被hang的现象之后，需要在不断电、不复位的情况下再次使用OpenOCD连接FPGA/芯片，此时OpenOCD输出的DebugMap才可被用于分析错误现场
- “00”：当前Commit的PC——用来指示最近正在Commit的PC，通过此信息可以大概推测CPU跑到了什么PC位置
- “16”：配置了ICache的话，记录ICache最近发出去的地址（暂时没有记录ILM的地址），理论上ICache有2个Outstanding，记录的是那个最先发出去还没有返回Response的地址
- “32”：配置了DCache 的话，记录DCache最近发出去的地址（DLM、Mem也可以被记录，暂时没有记录PPI/FIO发出去的地址），理论上DCache有很多个Outstanding，记录的是那个最先发出去还没有返回Response的地址

通过OpenOCD读取其他DebugMap寄存器¶

OpenOCD里有一组 *nuclei expose_cpu_core nuclei examine_cpu_core* 命令，可以使用这两个命令读取其他DebugMap寄存器， 详细参见 https://doc.nucleisys.com/nuclei_tools/openocd/intro.html#debug-map-feature

OpenOCD里的命令实现及使用方法 [source code](#)

- 注意 *nuclei expose_cpu_core* 命令需要在init命令之前使用
- *nuclei examine_cpu_core* 在init命令之后使用，也可以在gdb/telent连接上后使用，注意gdb给openocd发送命令需要使用monitor关键词 *monitor nuclei examine_cpu_core*

在binutils中新增自定义汇编指令教程¶

以下皆以32位指令为例说明

自定义扩展名的识别¶

以下以xnice扩展为例

文件:bfd\elfxx-riscv.c

riscv_supported_vendor_x_ext[] 函数:

```
static struct riscv_supported_ext riscv_supported_vendor_x_ext[] ={
    {"xnice",    ISA_SPEC_CLASS_DRAFT, 1, 0, 0},
}
```

Tips:该函数负责添加扩展名称和版本号，其中前面两位1,0为该扩展版本号

riscv_multi_subset_supports 函数:

```
/* Each instruction is belonged to an instruction class
INSN_CLASS_*.
Call riscv_subset_supports to make sure if the instruction is
valid. */

bool
riscv_multi_subset_supports (riscv_parse_subset_t *rps,
                            enum riscv_insn_class insn_class)
{
    switch (insn_class)
    {
        case INSN_CLASS_XNICE:
            return riscv_subset_supports (rps, "xnice");
    }
}
```

Tips: switch里面是要添加的内容，添加了xnice扩展的指令所对应的INSN_CLASS_XNICE与xnice扩展之间的联系

riscv_implicit_subsets[] 函数:(可选)

```
/* Please added in order since this table is only run once time.
 */
static struct riscv_implicit_subset riscv_implicit_subsets[] ={
    {"xnice", "+zve32x", check_implicit_always},
}
```

Tips:该函数控制自定义的xnice扩展是否依赖其他扩展，如果不依赖，则不需要添加。假设依赖zve32x扩展，则需要在该函数内按上面形式添加依赖关系，若依赖多个扩展，则在zve32x扩展后面继续添加

文件 include\opcode\riscv.h

```
enum riscv_insn_class
{
    INSN_CLASS_XNICE,
}
```

Tips:该文件主要负责在riscv_insn_class枚举类中，对INSN_CLASS_XNICE进行声明

自定义汇编指令识别¶

以下以新增一条标准R类型nice指令为例

1、添加指令编码

假设该nice指令汇编格式为nice rd, rs1, rs2，并且使用的是RISC-V预留的custom3区域的编码空间，其编码为：

Inst. format	Func7	rs2	rs1	Func3	rd	opcode
xnive rd,rs1,rs2	1011101	rs2	rs1	001	rd	1111011

- 生成编译器所需的opcode宏(推荐使用riscv-opcodes <https://github.com/riscv/riscv-opcodes/tree/master> 仓库)

```
git clone https://github.com/riscv/riscv-opcodes.git
cd riscv-opcodes/extensions/unratified/
vim rv_xnive //在该文件夹下创建xnive扩展指令文件(文件名规则是rv_name)，并根据指令模板添加一条指令
nice rd rs1 rs2 31..25=0x5D 14..12=1 6..2=0x1E 1..0=3 //此为需要添加
```

的指令

```
cd ../..
make EXTENSIONS='unratified/rv_xnice'
```

上述步骤后得到了opcode宏，在 riscv-opcodes/encoding.out.h 文件中，如下所示：

```
#define MATCH_NICE 0xba00107b
#define MASK_NICE 0xfe00707f
```

注意：也可以根据编码手动生成宏，规则为：MATCH_NICE 的编码是未定义位置全为0，其余位置不变。MASK_NICE 的编码是未定义位置全为0，其余位置全为1.

- 在 include\opcode\riscv-opc.h 文件中，添加上述生成的宏

2、添加扩展与扩展指令编码之间的联系

文件：opcodes\riscv-opc.c

riscv_opcodes[]函数：

```
const struct riscv_opcode riscv_opcodes[] =
{
/* name, xlen, isa, operands, match, mask, match_func, pinfo. */
{"xnice", 0, INSN_CLASS_XNICE, "d,s,t", MATCH_XNICE, MASK_XNICE,
match_opcode, 0 },
}
```

Tips: 第一个0代表该指令对xlen没有要求。d,s,t 分别代表rd,rs1,rs2, 其中对应的映射关系可在 gas/config/tc-riscv.c 文件 validate_riscv_insn 函数中查找

使用说明¶

使用时需要将xnice通过 -march选项传递给编译器，例如-march=rv32imafdc_xnice

参考链接:¶

修改binutils在RISC-V上添加汇编指令：

<https://blog.cyyself.name/add-compile-instr-for-riscv/>

nuclei自定义vpu指令的扩展识别及汇编实现:

<https://github.com/riscv-mcu/riscv-binutils-gdb/commit/c8806f4bd8c1a1673ec61ad3badfc3d490fa52f7>

OpenOCD 中 Nuclei 交叉触发功能使用指南

功能概述

为满足 AMP 多核调试中同步暂停(halt)与恢复(resume)的需求，Nuclei RISC-V CPU实现了 cross-trigger 功能，OpenOCD 已集成以下两种同步控制功能：

1. 同步暂停组 (halt_group) - 组内任一核暂停时，其他成员自动同步暂停
2. 同步恢复组 (resume_group) - 组内任一核恢复运行时，其他成员自动同步恢复

基本命令格式：

```
# add target to halt_group
nuclei cti halt_group on $_TARGETNAME0 $_TARGETNAME1

# remove target from halt_group
nuclei cti halt_group off $_TARGETNAME0 $_TARGETNAME1

# add target to resume_group
nuclei cti resume_group on $_TARGETNAME0 $_TARGETNAME1

# remove target from resume_group
nuclei cti resume_group off $_TARGETNAME0 $_TARGETNAME1
```

配置文件示例

1. 同步暂停组配置

```
adapter_khz      1000

interface ftdi
ftdi_vid_pid 0x0403 0x6010
ftdi_oscanc1_mode off

transport select jtag

ftdi_layout_init 0x0008 0x001b
ftdi_layout_signal nSRST -oe 0x0020 -data 0x0020
ftdi_layout_signal TCK -data 0x0001
ftdi_layout_signal TDI -data 0x0002
```

```

ftdi_layout_signal TDO -input 0x0004
ftdi_layout_signal TMS -data 0x0008
ftdi_layout_signal JTAG_SEL -data 0x0100 -oe 0x0100

set _CHIPNAME0 riscv0
jtag newtap $_CHIPNAME0 cpu -irlen 5 -expected-id 0x10900a6d
set _TARGETNAME0 $_CHIPNAME0.cpu
target create $_TARGETNAME0 riscv -chain-position $_TARGETNAME0 -
coreid 0

set _CHIPNAME1 riscv1
jtag newtap $_CHIPNAME1 cpu -irlen 5 -expected-id 0x10900a6d
set _TARGETNAME1 $_CHIPNAME1.cpu
target create $_TARGETNAME1 riscv -chain-position $_TARGETNAME1 -
coreid 0

init
#reset

if {[info exists pulse_srst]} {
    ftdi_set_signal nSRST 0
    ftdi_set_signal nSRST z
}

# 添加目标到暂停组
nuclei cti halt_group on $_TARGETNAME0 $_TARGETNAME1

foreach t [target names] {
    targets $t
    halt
}

```

2. 同步恢复组配置¶

```

adapter_khz      1000

interface ftdi
ftdi_vid_pid 0x0403 0x6010
ftdi_oscanc1_mode off

transport select jtag

ftdi_layout_init 0x0008 0x001b
ftdi_layout_signal nSRST -oe 0x0020 -data 0x0020
ftdi_layout_signal TCK -data 0x0001

```

```

ftdi_layout_signal TDI -data 0x0002
ftdi_layout_signal TDO -input 0x0004
ftdi_layout_signal TMS -data 0x0008
ftdi_layout_signal JTAG_SEL -data 0x0100 -oe 0x0100

set _CHIPNAME0 riscv0
jtag newtap $_CHIPNAME0 cpu -irlen 5 -expected-id 0x10900a6d
set _TARGETNAME0 $_CHIPNAME0.cpu
target create $_TARGETNAME0 riscv -chain-position $_TARGETNAME0 -
coreid 0

set _CHIPNAME1 riscv1
jtag newtap $_CHIPNAME1 cpu -irlen 5 -expected-id 0x10900a6d
set _TARGETNAME1 $_CHIPNAME1.cpu
target create $_TARGETNAME1 riscv -chain-position $_TARGETNAME1 -
coreid 0

init
#reset

if {[info exists pulse_srst]} {
    ftdi_set_signal nSRST 0
    ftdi_set_signal nSRST z
}

# add target to resume_group
nuclei cti resume_group on $_TARGETNAME0 $_TARGETNAME1

foreach t [target names] {
    targets $t
    halt
}

```

命令行验证步骤¶

1. 同步暂停组验证¶

1. 配置文件中已添加目标到 `halt_group`
2. 为两个核心分别加载不同固件
3. 仅在 `core0` 的 `_amp_wait()` 函数设置断点
4. 执行流程：先恢复 `core1`, 再恢复 `core0`
5. 验证结果：当 `core0` 触发断点暂停时，`core1` 同步暂停

The screenshot shows two terminal windows. The left window displays the OpenOCD log for the riscv1 core, indicating successful examination and connection. The right window shows the OpenOCD log for the riscv0 core, also indicating successful examination and connection. Both logs include configuration commands for exposing CSR registers and defining halt and resume groups.

```
[riscv1.cpu] Target successfully examined.
Info : [riscv1.cpu] Examination succeed
Info : [riscv1.cpu] starting gdb server on 3333
Info : Listening on port 3334 for gdb connections
Info : [riscv1.cpu] starting gdb server on 3334
Info : Listening on port 3334 for gdb connections
Error: Target cmd name is riscv0.cpu
Error: Target current_target_name is riscv0.cpu
Error: Target cmd name is riscv1.cpu
Error: Target current_target_name is riscv1.cpu
riscv0.cpu halted due to debug-request
riscv1.cpu halted due to debug-request
Info : Listening on port 6666 for tcl connections
Info : Listening on port 4444 for telnet connections
Info : accepting gdb connection on tcp/3334
riscv1.cpu halted due to debug-request
Info : New GDB Connection: 1, Target riscv1.cpu, state: halted
Info : accepting gdb connection on tcp/3333
riscv0.cpu halted due to debug-request
Info : New GDB Connection: 2, Target riscv0.cpu, state: halted
Info : [riscv0.cpu] Found 4 triggers
Info : [riscv0.cpu] Found 4 triggers
[...]
问题 帮助 调试控制台 终端 窗口 评论
```

```
openocd_ns_core0_all.cfg ×
SoC > ns_core0 > Board > fpga_eval > openocd_ns_core0_all.cfg
49 # Expose Nuclei self-defined CSRs range 770-800,835-850,1984-2032,2064-2070
50 # See https:
51 # Then user can view the csr register value in gdb using: info reg csr775 for CSR MTIV(0x307)
52 #riscv expose_csr 770-800,835-850,1984-2032,2064-2070
53
54 init
55 #reset
56 if {[ info exists pulse_srst]} {
57   ftdi_set_signal nSRST 0
58   ftdi_set_signal nSRST z
59 }
60
61 nuclei cti halt_group on $ _TARGETNAME0 $ _TARGETNAME1
# nuclei cti resume_group on $ _TARGETNAME0 $ _TARGETNAME1
62
63
64 foreach t [target names] {
```

For help, type "help".
Type "apropos word" to search for commands related to "word"...
Reading symbols from coremark.elf...
Remote debugging using localhost:3334
Breakpoint pending in ?? ()
(gdb) load
Loading section .init, size 0x344 lma 0x80000000
Loading section .text, size 0x8c8 lma 0x80000300
Loading section .data, size 0x120 lma 0x80000448
Start address 0x80000100, load size 3996
Transfer rate: 76 kB/sec, 3996 bytes/write.
(gdb) s
[riscv1.cpu] Found 4 triggers
164 la gp, _global_pointer\$

(gdb) b __amp_wait
Breakpoint 1 at 0x800002ec: file ../../../../../../SoC/ns_core0/Common/Source/GCC/startup_ns_core0.S, line 399.
(gdb) c
Continuing.

Breakpoint 1, __amp_wait () at ../../../../../../SoC/ns_core0/Common/Source/GCC/startup_ns_core0.S:399
399 wfi
(gdb) [...]

For help, type "help".
Type "apropos word" to search for commands related to "word"...
Reading symbols from helloworld.elf...
Remote debugging using localhost:3333
Breakpoint pending in ?? ()
(gdb) load
Loading section .init, size 0x344 lma 0x80000000
Loading section .text, size 0x208 lma 0x80000380
Loading section .data, size 0x10 lma 0x80000430
Start address 0x80000100, load size 10280
Transfer rate: 73 kB/sec, 3500 bytes/write.
(gdb) s
[riscv0.cpu] Found 4 triggers
164 la gp, _global_pointer\$

(gdb) c
Continuing.

Program received signal SIGINT, Interrupt.
__amp_wait () at ../../../../../../SoC/ns_core0/Common/Source/GCC/startup_ns_core0.S:400
400 j ib
(gdb) [...]

2. 同步恢复组验证

1. 配置文件中已添加目标到 resume_group
2. 为两个核心加载相同 helloworld 固件
3. 仅向 core0 发送继续运行命令：
4. 验证结果：串口输出显示两个核心同时运行

The screenshot shows two terminal windows. The left window displays the OpenOCD log for the riscv1 core, and the right window shows the OpenOCD log for the riscv0 core. Both logs include configuration commands for exposing CSR registers and defining halt and resume groups. The resume_group command is highlighted in both logs.

```
[riscv1.cpu] XLEN=32, misa=0x4010112f
[riscv1.cpu] Target successfully examined.
Info : [riscv1.cpu] examination succeed
Info : [riscv1.cpu] starting gdb server on 3333
Info : Listening on port 3334 for gdb connections
Info : [riscv1.cpu] starting gdb server on 3334
Info : Listening on port 3334 for gdb connections
Error: Target cmd name is riscv0.cpu
Error: Target current_target_name is riscv0.cpu
Error: Target cmd name is riscv1.cpu
Error: Target current_target_name is riscv1.cpu
riscv0.cpu halted due to undefined.
riscv1.cpu halted due to undefined.
Info : Listening on port 6666 for tcl connections
Info : Listening on port 4444 for telnet connections
Info : accepting gdb connection on tcp/3334
riscv1.cpu halted due to undefined.
Info : New GDB Connection: 1, Target riscv1.cpu, state: halted
undefined debug reason 8 (UNDEFINED) - target needs reset
Info : accepting gdb connection on tcp/3333
riscv0.cpu halted due to undefined.
Info : New GDB Connection: 2, Target riscv0.cpu, state: halted
undefined debug reason 8 (UNDEFINED) - target needs reset
[...]
问题 帮助 调试控制台 终端 窗口 评论
```

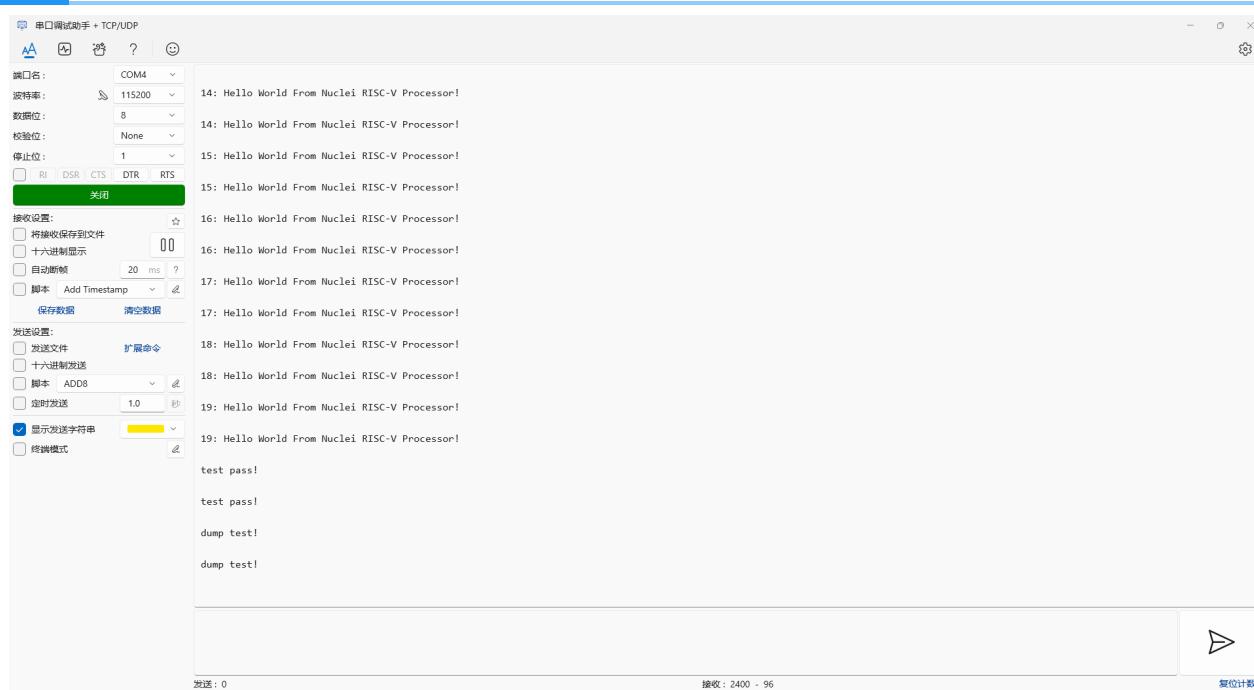
```
openocd_ns_core0_all.cfg ×
SoC > ns_core0 > Board > fpga_eval > openocd_ns_core0_all.cfg
52 #riscv expose_csr 770-800,835-850,1984-2032,2064-2070
53
54 init
55 #reset
56 if {[ info exists pulse_srst]} {
57   ftdi_set_signal nSRST 0
58   ftdi_set_signal nSRST z
59 }
60
61 # nuclei cti halt_group on $ _TARGETNAME0 $ _TARGETNAME1
nuclei cti resume_group on $ _TARGETNAME0 $ _TARGETNAME1
62
63
64 foreach t [target names] {
```

"Run gdb to connect openocd server and debug"
riscv64-unknown-elf-gdb helloworld.elf --ex "set remotetimeout 240" -ex "target extended-remote localhost:3333"
GNU gdb (GDB) 16.2.20250219-git
Copyright (C) 2024 Free Software Foundation, Inc.
License GPLv3+: GNU General Public License 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.
Type "show copying" and "show warranty" for details.
This GDB was configured as "--host=i686-wsl2-mingw32 --target=riscv64-unknown-elf".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<https://www.gnu.org/software/gdb/bugs/>.
Find the GDB manual and other documentation resources online at:
<http://www.gnu.org/software/gdb/documentation/>.

For help, type "help".
Type "apropos word" to search for commands related to "word"...
Reading symbols from helloworld.elf...
Remote debugging using localhost:3334
__amp_wait () at ../../../../../../SoC/ns_core0/Common/Source/GCC/startup_ns_core0.S:400
400 j ib
(gdb) load
Loading section .init, size 0x344 lma 0x80000000
Loading section .text, size 0x208 lma 0x80000380
Loading section .data, size 0x10 lma 0x80000430
Start address 0x80000100, load size 10280
Transfer rate: 73 kB/sec, 3500 bytes/write.
(gdb) c
Continuing.
[...]

"Run gdb to connect openocd server and debug"
riscv64-unknown-elf-gdb helloworld.elf --ex "set remotetimeout 240" -ex "target extended-remote localhost:3333"
GNU gdb (GDB) 16.2.20250219-git
Copyright (C) 2024 Free Software Foundation, Inc.
License GPLv3+: GNU General Public License 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.
Type "show copying" and "show warranty" for details.
This GDB was configured as "--host=i686-wsl2-mingw32 --target=riscv64-unknown-elf".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<https://www.gnu.org/software/gdb/bugs/>.
Find the GDB manual and other documentation resources online at:
<http://www.gnu.org/software/gdb/documentation/>.

For help, type "help".
Type "apropos word" to search for commands related to "word"...
Reading symbols from helloworld.elf...
Remote debugging using localhost:3333
__amp_wait () at ../../../../../../SoC/ns_core0/Common/Source/GCC/startup_ns_core0.S:400
400 j ib
(gdb) load
Loading section .init, size 0x344 lma 0x80000000
Loading section .text, size 0x208 lma 0x80000380
Loading section .data, size 0x10 lma 0x80000430
Start address 0x80000100, load size 10280
Transfer rate: 74 kB/sec, 3500 bytes/write.
(gdb) [...]



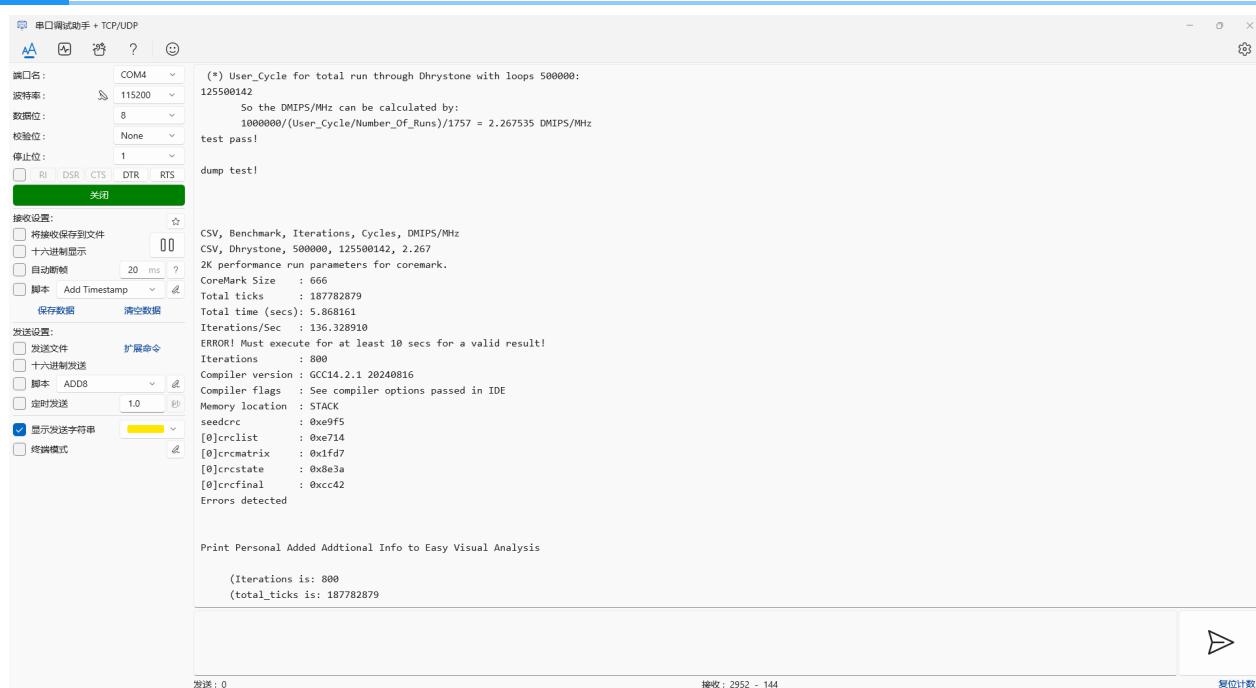
IDE 验证步骤

1. 同步暂停组验证

1. 配置文件中已配置 `halt_group`
2. 为两个核心加载不同固件
3. 在 `core0` 的 `core_main.c` 第 152 行设置断点
4. 操作顺序：
5. 先启动 `core1` 运行
6. 再启动 `core0` 运行
7. 验证结果：`core0` 触发断点时，`core1` 同步暂停

```

// core_main.c
114 //endif
115
116 ee_u16 i, j = 0, num_algorithms = 0;
117 ee_u16 known_id = -1, total_errors = 0;
118 ee_u32 seed0, seed1 = 0;
119 CORE_TICKS total_time;
120 core_results results[MULTITHREAD];
121 #if (MEM_METHOD==MEM_STACK)
122 ee_u8 stack_memblk[TOTAL_DATA_SIZE];
123 #endif
124
125 /* first call any initializations
126 portable_init(&results[0].port);
127 /* First some checks to make sure
128 if (sizeof(struct list_head_d) > 128) {
129     printf("list_head structure too big for comparable data!\n");
130 }
131
132 /* put in some default values based on one seed only for easy testing */
133 if ((results[0].seed1 == 0) && (results[0].seed2 == 0) && (results[0].seed3 == 0)) { /* validation run */
134     results[0].seed1 = get_seed(1);
135     results[0].seed2 = get_seed(2);
136     results[0].seed3 = get_seed(3);
137 }
138
139 #ifdef CFG_SIMULATION
140 // 200/300 4 iterations are enough for training
141 if ((CPU_SERIES && (CPU_SERIES == 200)) || (CPU_SERIES == 300))
142     results[0].iterations = 4;
143 else
144     results[0].iterations = 20;
145 #endif
146
147 results[0].iterations = ITERATIONS;
148 #endif
149
150 ee_printf("Start to run coremark for Xu iterations\n", (unsigned int)results[0].iterations);
151
152 if (results[0].execs == get_seed_32(0))
153     if (results[0].execs == 0) { /* if not supplied, execute all algorithms */
154         results[0].execs = ALL_ALGORITHMS_MASK;
155     }
156
157 /* put in some default values based on one seed only for easy testing */
158 if ((results[0].seed1 == 0) && (results[0].seed2 == 0) && (results[0].seed3 == 0)) { /* validation run */
159     results[0].seed1 = 0x4141;
160     results[0].seed2 = 0x4141;
161     results[0].seed3 = 0x4141;
162 }
163
164 #if (MEM_METHOD==MEM_STACK)
165 ee_u8 memblk[8] = {0}; static memblk;
166 results[0].size = TOTAL_DATA_SIZE;
167 result=&f1; err = 0;
168
169 #endif
170
171 MAIN_RETURN_TYPE main(void)
172 {
173     int argc = 0;
174     char* argv[1];
175     #else
176     MAIN_RETURN_TYPE main(int argc, char* argv[])
177     {
178         #endif
179
180         ee_u16 i, j = 0, num_algorithms = 0;
181         ee_u16 known_id = -1, total_errors = 0;
182         ee_u32 seed0, seed1 = 0;
183         CORE_TICKS total_time;
184         core_results results[MULTITHREAD];
185         #if (MEM_METHOD==MEM_STACK)
186         ee_u8 stack_memblk[TOTAL_DATA_SIZE * MULTITHREAD];
187         #endif
188
189         /* first call any initializations needed */
190         portable_init(&results[0].port, argc, argv);
191         /* First some checks to make sure our work will run ok */
192         if (sizeof(struct list_head_d) > 128) {
193             printf("list_head structure too big for comparable data!\n");
194         }
195
196         /* put in some default values based on one seed only for easy testing */
197         if ((results[0].seed1 == 0) && (results[0].seed2 == 0) && (results[0].seed3 == 0)) { /* validation run */
198             results[0].seed1 = get_seed(1);
199             results[0].seed2 = get_seed(2);
200             results[0].seed3 = get_seed(3);
201             results[0].iterations = get_seed_32(4);
202         }
203
204         #if (CORE_DEBUG)
205             results[0].iterations = 1;
206         #endif
207
208         /* put in some default values based on one seed only for easy testing */
209         if ((results[0].seed1 == 0) && (results[0].seed2 == 0) && (results[0].seed3 == 0)) { /* validation run */
210             results[0].seed1 = 0;
211             results[0].seed2 = 0;
212             results[0].seed3 = 0x4141;
213         }
214
215         #if (RESULTS==RESULTS_ALL)
216             results[0].seed1 = 0;
217             results[0].seed2 = 0;
218             results[0].seed3 = 0x4141;
219         #endif
220
221         /* put in some default values based on one seed only for easy testing */
222         if ((results[0].seed1 == 0) && (results[0].seed2 == 0) && (results[0].seed3 == 0)) { /* validation run */
223             results[0].seed1 = get_seed(1);
224             results[0].seed2 = get_seed(2);
225             results[0].seed3 = get_seed(3);
226             results[0].iterations = get_seed_32(4);
227         }
228
229         #if (CORE_DEBUG)
230             results[0].iterations = ITERATIONS;
231         #endif
232
233         /* put in some default values based on one seed only for easy testing */
234         if ((results[0].seed1 == 0) && (results[0].seed2 == 0) && (results[0].seed3 == 0)) { /* validation run */
235             results[0].seed1 = get_seed_32(0);
236             results[0].seed2 = 0;
237             results[0].seed3 = 0;
238         }
239
240         #if (RESULTS==RESULTS_ALL)
241             results[0].seed1 = 0;
242             results[0].seed2 = 0;
243             results[0].seed3 = 0x4141;
244         #endif
245
246         result=&f1; err = 0;
247
248     }
249
250     #endif
251
252     #endif
253
254     #endif
255
256     #endif
257
258     #endif
259
260     #endif
261
262     #endif
263
264     #endif
265
266     #endif
267
268     #endif
269
270     #endif
271
272     #endif
273
274     #endif
275
276     #endif
277
278     #endif
279
280     #endif
281
282     #endif
283
284     #endif
285
286     #endif
287
288     #endif
289
290     #endif
291
292     #endif
293
294     #endif
295
296     #endif
297
298     #endif
299
299 #endif
300
301 #endif
302
303 #endif
304
305 #endif
306
307 #endif
308
309 #endif
310
311 #endif
312
313 #endif
314
315 #endif
316
317 #endif
318
319 #endif
320
321 #endif
322
323 #endif
324
325 #endif
326
327 #endif
328
329 #endif
330
331 #endif
332
333 #endif
334
335 #endif
336
337 #endif
338
339 #endif
340
341 #endif
342
343 #endif
344
345 #endif
346
347 #endif
348
349 #endif
350
351 #endif
352
353 #endif
354
355 #endif
356
357 #endif
358
359 #endif
360
361 #endif
362
363 #endif
364
365 #endif
366
367 #endif
368
369 #endif
370
371 #endif
372
373 #endif
374
375 #endif
376
377 #endif
378
379 #endif
380
381 #endif
382
383 #endif
384
385 #endif
386
387 #endif
388
389 #endif
390
391 #endif
392
393 #endif
394
395 #endif
396
397 #endif
398
399 #endif
399 #endif
400
401 #endif
402
403 #endif
404
405 #endif
406
407 #endif
408
409 #endif
410
411 #endif
412
413 #endif
414
415 #endif
416
417 #endif
418
419 #endif
420
421 #endif
422
423 #endif
424
425 #endif
426
427 #endif
428
429 #endif
430
431 #endif
432
433 #endif
434
435 #endif
436
437 #endif
438
439 #endif
440
441 #endif
442
443 #endif
444
445 #endif
446
447 #endif
448
449 #endif
450
451 #endif
452
453 #endif
454
455 #endif
456
457 #endif
458
459 #endif
459 #endif
460
461 #endif
462
463 #endif
464
465 #endif
466
467 #endif
468
469 #endif
470
471 #endif
472
473 #endif
474
475 #endif
476
477 #endif
478
479 #endif
479 #endif
480
481 #endif
482
483 #endif
484
485 #endif
486
487 #endif
488
489 #endif
489 #endif
490
491 #endif
492
493 #endif
494
495 #endif
496
497 #endif
498
499 #endif
499 #endif
500
501 #endif
502
503 #endif
504
505 #endif
506
507 #endif
508
509 #endif
509 #endif
510
511 #endif
512
513 #endif
514
515 #endif
516
517 #endif
518
519 #endif
519 #endif
520
521 #endif
522
523 #endif
524
525 #endif
526
527 #endif
528
529 #endif
529 #endif
530
531 #endif
532
533 #endif
534
535 #endif
536
537 #endif
538
539 #endif
539 #endif
540
541 #endif
542
543 #endif
544
545 #endif
546
547 #endif
548
549 #endif
549 #endif
550
551 #endif
552
553 #endif
554
555 #endif
556
557 #endif
558
559 #endif
559 #endif
560
561 #endif
562
563 #endif
564
565 #endif
566
567 #endif
568
569 #endif
569 #endif
570
571 #endif
572
573 #endif
574
575 #endif
576
577 #endif
578
579 #endif
579 #endif
580
581 #endif
582
583 #endif
584
585 #endif
586
587 #endif
588
589 #endif
589 #endif
590
591 #endif
592
593 #endif
594
595 #endif
596
597 #endif
598
599 #endif
599 #endif
600
601 #endif
602
603 #endif
604
605 #endif
606
607 #endif
608
609 #endif
609 #endif
610
611 #endif
612
613 #endif
614
615 #endif
616
617 #endif
618
618 #endif
619
620 #endif
621
622 #endif
623
624 #endif
625
626 #endif
627
627 #endif
628
629 #endif
629 #endif
630
631 #endif
632
633 #endif
634
635 #endif
636
636 #endif
637
638 #endif
638 #endif
639
639 #endif
640
641 #endif
642
643 #endif
644
644 #endif
645
646 #endif
646 #endif
647
647 #endif
648
649 #endif
649 #endif
650
651 #endif
652
653 #endif
654
655 #endif
655 #endif
656
656 #endif
657
658 #endif
658 #endif
659
659 #endif
660
661 #endif
662
663 #endif
664
665 #endif
665 #endif
666
666 #endif
667
668 #endif
668 #endif
669
669 #endif
670
671 #endif
672
673 #endif
674
675 #endif
675 #endif
676
676 #endif
677
678 #endif
678 #endif
679
679 #endif
680
681 #endif
682
683 #endif
683 #endif
684
684 #endif
685
686 #endif
686 #endif
687
687 #endif
688
689 #endif
689 #endif
690
690 #endif
691
692 #endif
693
693 #endif
694
695 #endif
695 #endif
696
696 #endif
697
698 #endif
698 #endif
699
699 #endif
700
701 #endif
702
703 #endif
703 #endif
704
704 #endif
705
706 #endif
706 #endif
707
707 #endif
708
709 #endif
709 #endif
710
710 #endif
711
712 #endif
712 #endif
713
713 #endif
714
715 #endif
715 #endif
716
716 #endif
717
718 #endif
718 #endif
719
719 #endif
720
721 #endif
721 #endif
722
722 #endif
723
723 #endif
724
724 #endif
725
725 #endif
726
726 #endif
727
727 #endif
728
728 #endif
729
729 #endif
730
730 #endif
731
731 #endif
732
732 #endif
733
733 #endif
734
734 #endif
735
735 #endif
736
736 #endif
737
737 #endif
738
738 #endif
739
739 #endif
740
740 #endif
741
741 #endif
742
742 #endif
743
743 #endif
744
744 #endif
745
745 #endif
746
746 #endif
747
747 #endif
748
748 #endif
749
749 #endif
750
750 #endif
751
751 #endif
752
752 #endif
753
753 #endif
754
754 #endif
755
755 #endif
756
756 #endif
757
757 #endif
758
758 #endif
759
759 #endif
760
760 #endif
761
761 #endif
762
762 #endif
763
763 #endif
764
764 #endif
765
765 #endif
766
766 #endif
767
767 #endif
768
768 #endif
769
769 #endif
770
770 #endif
771
771 #endif
772
772 #endif
773
773 #endif
774
774 #endif
775
775 #endif
776
776 #endif
777
777 #endif
778
778 #endif
779
779 #endif
780
780 #endif
781
781 #endif
782
782 #endif
783
783 #endif
784
784 #endif
785
785 #endif
786
786 #endif
787
787 #endif
788
788 #endif
789
789 #endif
790
790 #endif
791
791 #endif
792
792 #endif
793
793 #endif
794
794 #endif
795
795 #endif
796
796 #endif
797
797 #endif
798
798 #endif
799
799 #endif
800
800 #endif
801
801 #endif
802
802 #endif
803
803 #endif
804
804 #endif
805
805 #endif
806
806 #endif
807
807 #endif
808
808 #endif
809
809 #endif
810
810 #endif
811
811 #endif
812
812 #endif
813
813 #endif
814
814 #endif
815
815 #endif
816
816 #endif
817
817 #endif
818
818 #endif
819
819 #endif
820
820 #endif
821
821 #endif
822
822 #endif
823
823 #endif
824
824 #endif
825
825 #endif
826
826 #endif
827
827 #endif
828
828 #endif
829
829 #endif
830
830 #endif
831
831 #endif
832
832 #endif
833
833 #endif
834
834 #endif
835
835 #endif
836
836 #endif
837
837 #endif
838
838 #endif
839
839 #endif
840
840 #endif
841
841 #endif
842
842 #endif
843
843 #endif
844
844 #endif
845
845 #endif
846
846 #endif
847
847 #endif
848
848 #endif
849
849 #endif
850
850 #endif
851
851 #endif
852
852 #endif
853
853 #endif
854
854 #endif
855
855 #endif
856
856 #endif
857
857 #endif
858
858 #endif
859
859 #endif
860
860 #endif
861
861 #endif
862
862 #endif
863
863 #endif
864
864 #endif
865
865 #endif
866
866 #endif
867
867 #endif
868
868 #endif
869
869 #endif
870
870 #endif
871
871 #endif
872
872 #endif
873
873 #endif
874
874 #endif
875
875 #endif
876
876 #endif
877
877 #endif
878
878 #endif
879
879 #endif
880
880 #endif
881
881 #endif
882
882 #endif
883
883 #endif
884
884 #endif
885
885 #endif
886
886 #endif
887
887 #endif
888
888 #endif
889
889 #endif
890
890 #endif
891
891 #endif
892
892 #endif
893
893 #endif
894
894 #endif
895
895 #endif
896
896 #endif
897
897 #endif
898
898 #endif
899
899 #endif
900
900 #endif
901
901 #endif
902
902 #endif
903
903 #endif
904
904 #endif
905
905 #endif
906
906 #endif
907
907 #endif
908
908 #endif
909
909 #endif
910
910 #endif
911
911 #endif
912
912 #endif
913
913 #endif
914
914 #endif
915
915 #endif
916
916 #endif
917
917 #endif
918
918 #endif
919
919 #endif
920
920 #endif
921
921 #endif
922
922 #endif
923
923 #endif
924
924 #endif
925
925 #endif
926
926 #endif
927
927 #endif
928
928 #endif
929
929 #endif
930
930 #endif
931
931 #endif
932
932 #endif
933
933 #endif
934
934 #endif
935
935 #endif
936
936 #endif
937
937 #endif
938
938 #endif
939
939 #endif
940
940 #endif
941
941 #endif
942
942 #endif
943
943 #endif
944
944 #endif
945
945 #endif
946
946 #endif
947
947 #endif
948
948 #endif
949
949 #endif
950
950 #endif
951
951 #endif
952
952 #endif
953
953 #endif
954
954 #endif
955
955 #endif
956
956 #endif
957
957 #endif
958
958 #endif
959
959 #endif
960
960 #endif
961
961 #endif
962
962 #endif
963
963 #endif
964
964 #endif
965
965 #endif
966
966 #endif
967
967 #endif
968
968 #endif
969
969 #endif
970
970 #endif
971
971 #endif
972
972 #endif
973
973 #endif
974
974 #endif
975
975 #endif
976
976 #endif
977
977 #endif
978
978 #endif
979
979 #endif
980
980 #endif
981
981 #endif
982
982 #endif
983
983 #endif
984
984 #endif
985
985 #endif
986
986 #endif
987
987 #endif
988
988 #endif
989
989 #endif
990
990 #endif
991
991 #endif
992
992 #endif
993
993 #endif
994
994 #endif
995
995 #endif
996
996 #endif
997
997 #endif
998
998 #endif
999
999 #endif
1000
1000 #endif
1001
1001 #endif
1002
1002 #endif
1003
1003 #endif
1004
1004 #endif
1005
1005 #endif
1006
1006 #endif
1007
1007 #endif
1008
1008 #endif
1009
1009 #endif
1010
1010 #endif
1011
1011 #endif
1012
1012 #endif
1013
1013 #endif
1014
1014 #endif
1015
1015 #endif
1016
1016 #endif
1017
1017 #endif
1018
1018 #endif
1019
1019 #endif
1020
1020 #endif
1021
1021 #endif
1022
1022 #endif
1023
1023 #endif
1024
1024 #endif
1025
1025 #endif
1026
1026 #endif
1027
1027 #endif
1028
1028 #endif
1029
1029 #endif
1030
1030 #endif
1031
1031 #endif
1032
1032 #endif
1033
1033 #endif
1034
1034 #endif
1035
1035 #endif
1036
1036 #endif
1037
1037 #endif
1038
1038 #endif
1039
1039 #endif
1040
1040 #endif
1041
1041 #endif
1042
1042 #endif
1043
1043 #endif
1044
1044 #endif
1045
1045 #endif
1046
1046 #endif
1047
1047 #endif
1048
1048 #endif
1049
1049 #endif
1050
1050 #endif
1051
1051 #endif
1052
1052 #endif
1053
1053 #endif
1054
1054 #endif
1055
1055 #endif
1056
1056 #endif
1057
1057 #endif
1058
1058 #endif
1059
1059 #endif
1060
1060 #endif
1061
1061 #endif
1062
1062 #endif
1063
1063 #endif
1064
1064 #endif
1065
1065 #endif
1066
1066 #endif
1067
1067 #endif
1068
1068 #endif
1069
1069 #endif
1070
1070 #endif
1071
1071 #endif
1072
1072 #endif
1073
1073 #endif
1074
1074 #endif
1075
1075 #endif
1076
1076 #endif
1077
1077 #endif
1078
1078 #endif
1079
1079 #endif
1080
1080 #endif
1081
1081 #endif
1082
1082 #endif
1083
1083 #endif
1084
1084 #endif
1085
1085 #endif
1086
1086 #endif
1087
1087 #endif
1088
1088 #endif
1089
1089 #endif
1090
1090 #endif
1091
1091 #endif
1092
1092 #endif
1093
1093 #endif
1094
1094 #endif
1095
1095 #endif
1096
1096 #endif
1097
1097 #endif
1098
1098 #endif
1099
1099 #endif
1100
1100 #endif
1101
1101 #endif
1102
1102 #endif
1103
1103 #endif
1104
1104 #endif
1105
1105 #endif
1106
1106 #endif
1107
1107 #endif
1108
1108 #endif
1109
1109 #endif
1110
1110 #endif
1111
1111 #endif
1112
1112 #endif
1113
1113 #endif
1114
1114 #endif
1115
1115 #endif
1116
1116 #endif
1117
1117 #endif
1118
1118 #endif
1119
1119 #endif
1120
1120 #endif
1121
1121 #endif
1122
1122 #endif
1123
1123 #endif
1124
1124 #endif
1125
1125 #endif
1126
1126 #endif
1127
1127 #endif
1128
1128 #endif
1129
1129 #endif
1130
1130 #endif
1131
1131 #endif
1132
1132 #endif
1133
1133 #endif
1134
1134 #endif
1135
1135 #endif
1136
1136 #endif
1137
1137 #endif
1138
1138 #endif
1139
1139 #endif
1140
1140 #endif
1141
1141 #endif
1142
1142 #endif
1143
1143 #endif
1144
1144 #endif
1145
1145 #endif
1146
1146 #endif
1147
1147 #endif
1148
1148 #endif
1149
1149 #endif
1150
1150 #endif
1151
1151 #endif
1152
1152 #endif
1153
1153 #endif
1154
1154 #endif
1155
1155 #endif
1156
1156 #endif
1157
1157 #endif
1158
1158 #endif
1159
1159 #endif
1160
1160 #endif
1161
1161 #endif
1162
1162 #endif
1163
1163 #endif
1164
1164 #endif
1165
1165 #endif
1166
1166 #endif
1167
1167 #endif
1168
1168 #endif
1169
1169 #endif
1170
1170 #endif
1171
1171 #endif
1172
1172 #endif
1173
1173 #endif
1174
1174 #endif
1175
1175 #endif
1176
1176 #endif
1177
1177 #endif
1178
1178 #endif
1179
1179 #endif
1180
1180 #endif
1181
1181 #endif
1182
1182 #endif
1183
1183 #endif
1184
1184 #endif
1185
1185 #endif
1186
1186 #endif
1187
1187 #endif
1188
1188 #endif
1189
1189 #endif
1190
1190 #endif
1191
1191 #endif
1192
1192 #endif
1193
1193 #endif
1194
1194 #endif
1195
1195 #endif
1196
1196 #endif
1197
1197 #endif
1198
1198 #endif
1199
1199 #endif
1200
1200 #endif
1201
1201 #endif
1202
1202 #endif
1203
1203 #endif
1204
1204 #endif
1205
1205 #endif
1206
1206 #endif
1207
1207 #endif
1208
1208 #endif
1209
1209 #endif
1210
1210 #endif
1211
1211 #endif
1212
1212 #endif
1213
1213 #endif
1214
1214 #endif
1215
1215 #endif
1216
1216 #endif
1217
1217 #endif
1218
1218 #endif
1219
1219 #endif
1220
1220 #endif
1221
1221 #endif
1222
1222 #endif
1223
1223 #endif
1224
1224 #endif
1225
1225 #endif
1226
1226 #endif
1227
1227 #endif
1228
1228 #endif
1229
1229 #endif
1230
1230 #endif
1231
1231 #endif
1232
1232 #endif
1233
1233 #endif
1234
1234 #endif
1235
1235 #endif
1236
1236 #endif
1237
1237 #endif
1238
1238 #endif
1239
1239 #endif
1240
1240 #endif
1241
1241 #endif
1242
1242 #endif
1243
1243 #endif
1244
1244 #endif
1245
1245 #endif
1246
1246 #endif
1247
1247 #endif
1248
1248 #endif
1249
1249 #endif
1250
1250 #endif
1251
1251 #endif
1252
1252 #endif
1253
1253 #endif
1254
1254 #endif
1255
1255 #endif
1256
1256 #endif
1257
1257 #endif
1258
1258 #endif
1259
1259 #endif
1260
1260 #endif
1261
1261 #endif
1262
1262 #endif
1263
1263 #endif
1264
1264 #endif
1265
1265 #endif
1266
1266 #endif
1267
1267 #endif
1268
1268 #endif
1269
1269 #endif
1270
1270 #endif
1271
1271 #endif
1272
1272 #endif
1273
1273 #endif
1274
1274 #endif
1275
1275 #endif
1276
1276 #endif
1277
1277 #endif
1278
1278 #endif
1279
1279 #endif
1280
1280 #endif
1281
1281 #endif
1282
1282 #endif
1283
1283 #endif
1284
1284 #endif
1285
1285 #endif
1286
1286 #endif
1287
1287 #endif
1288
1288 #endif
1289
1289 #endif
1290
1290 #endif
1291
1291 #endif
1292
1292 #endif
1293
1293 #endif
1294
1294 #endif
1295
1295 #endif
1296
1296 #endif
1297
1297 #endif
1298
1298 #endif
1299
1299 #endif
1300
1300 #endif
1301
1301 #endif
1302
1302 #endif
1303
1
```



OpenOCD对FreeRTOS的调试支持使用指南

通过更新您的Nuclei Studio IDE到202502版本和下载0.7.1的sdk-nuclei_sdk，并配合一些下面的修改，就可以使用OpenOCD对FreeRTOS进行调试。

环境准备

Nuclei Studio：

- NucleiStudio 202502 Windows
- NucleiStudio 202502 Linux

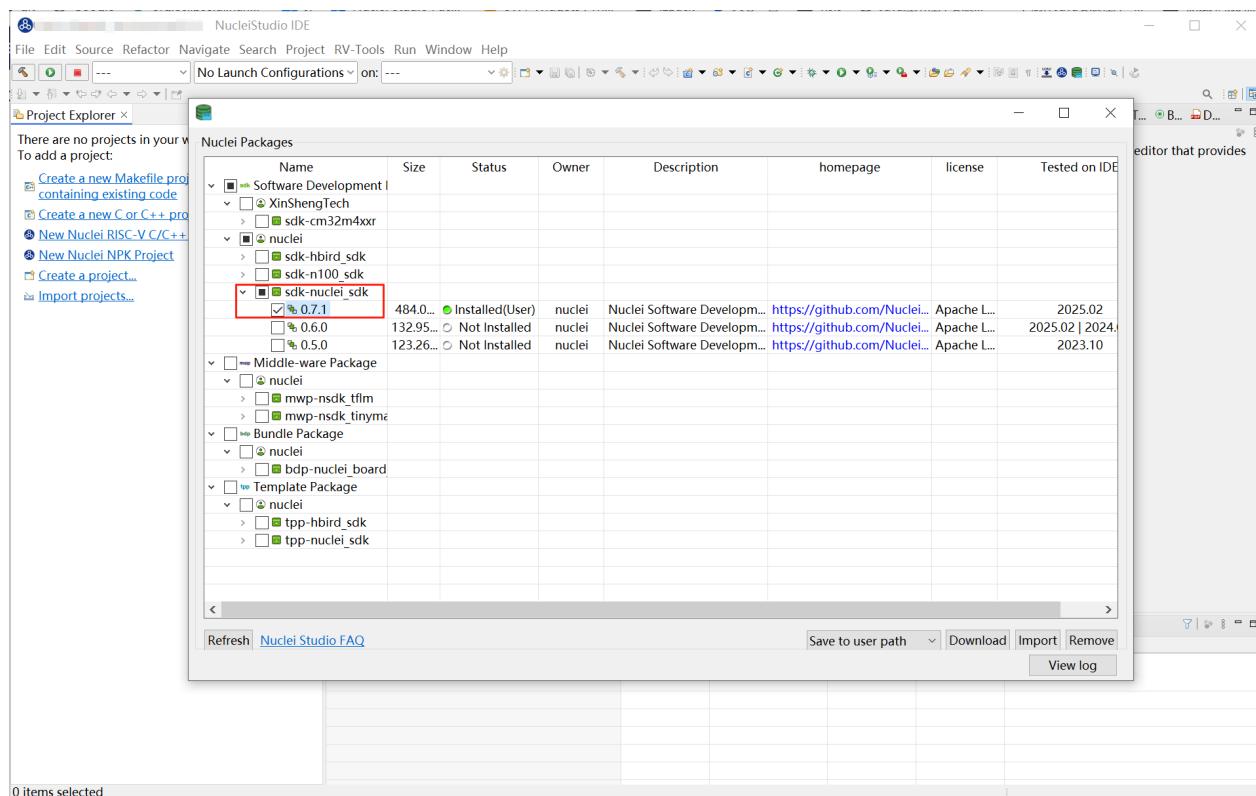
Nuclei OpenOCD：

- 使用NucleiStudio 202502自带的的OpenOCD即可。

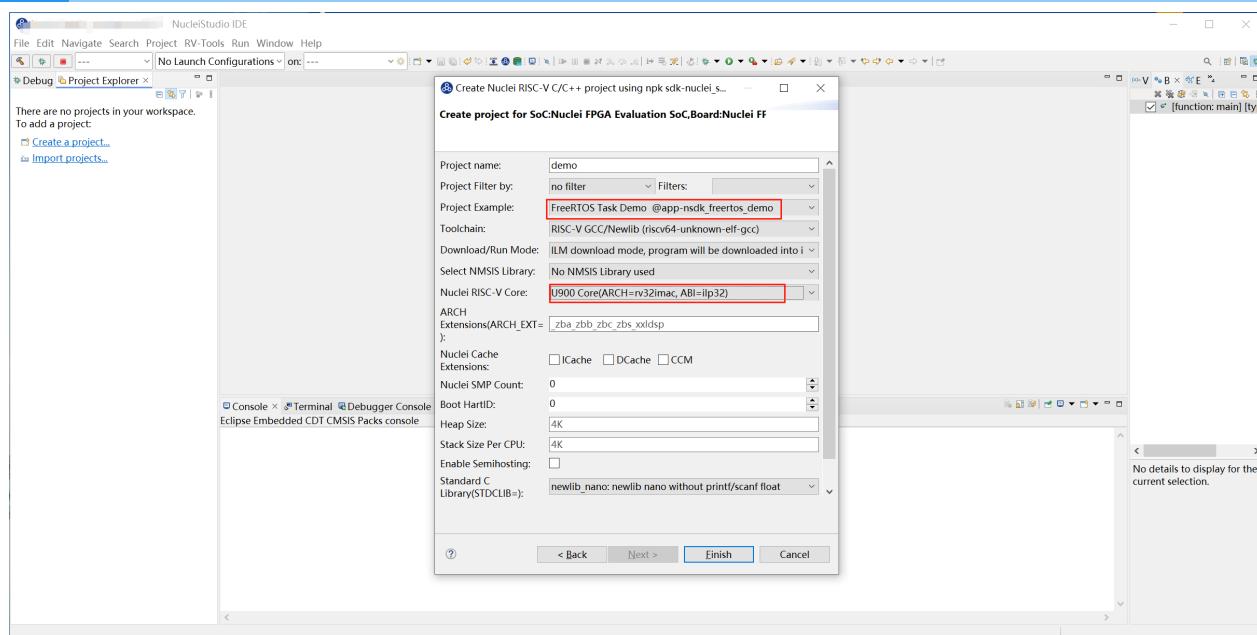
使用步骤

step1：创建原始工程

在NucleiStudio IDE下载好0.7.1版本的sdk-nuclei_sdk。



创建一个900的项目，如下图。



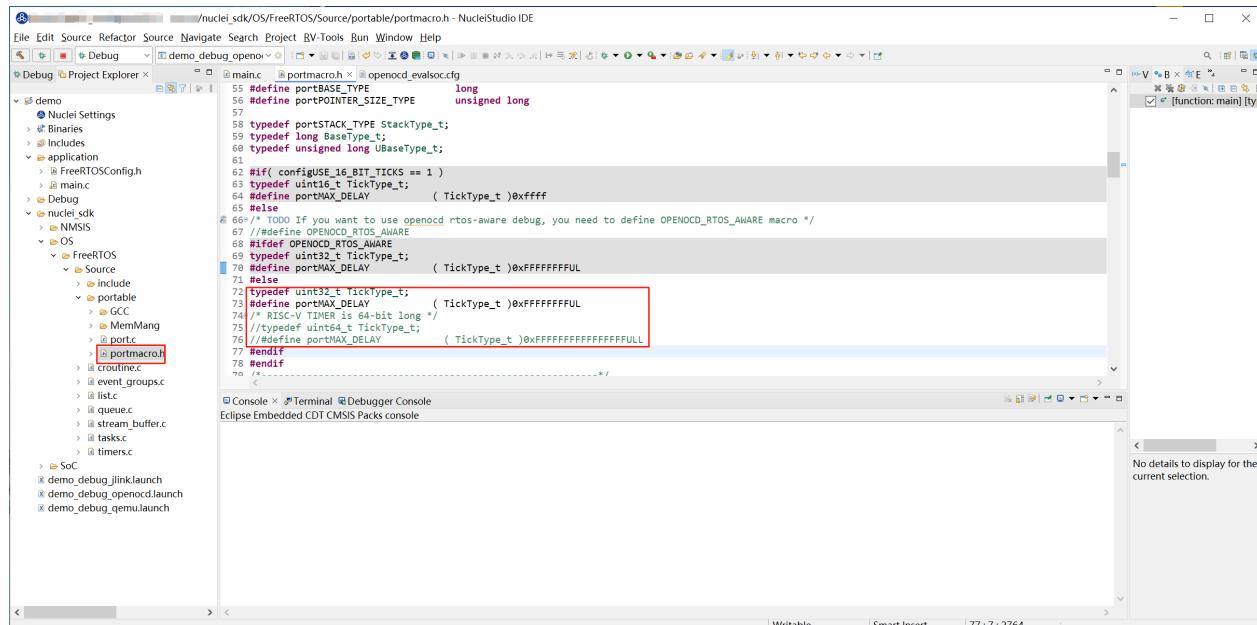
开发板烧写对应的bit即可，这里我们使用

u900_best_config_ku060_50M_c1dd7f44af_915aef97_202504141013_v4.1.0.bit

step2：修改portmacro.h内容

项目创建好，找到nuclei_sdk\OS\FreeRTOS\Source\portable\portmacro.h，修改该文件内容如下

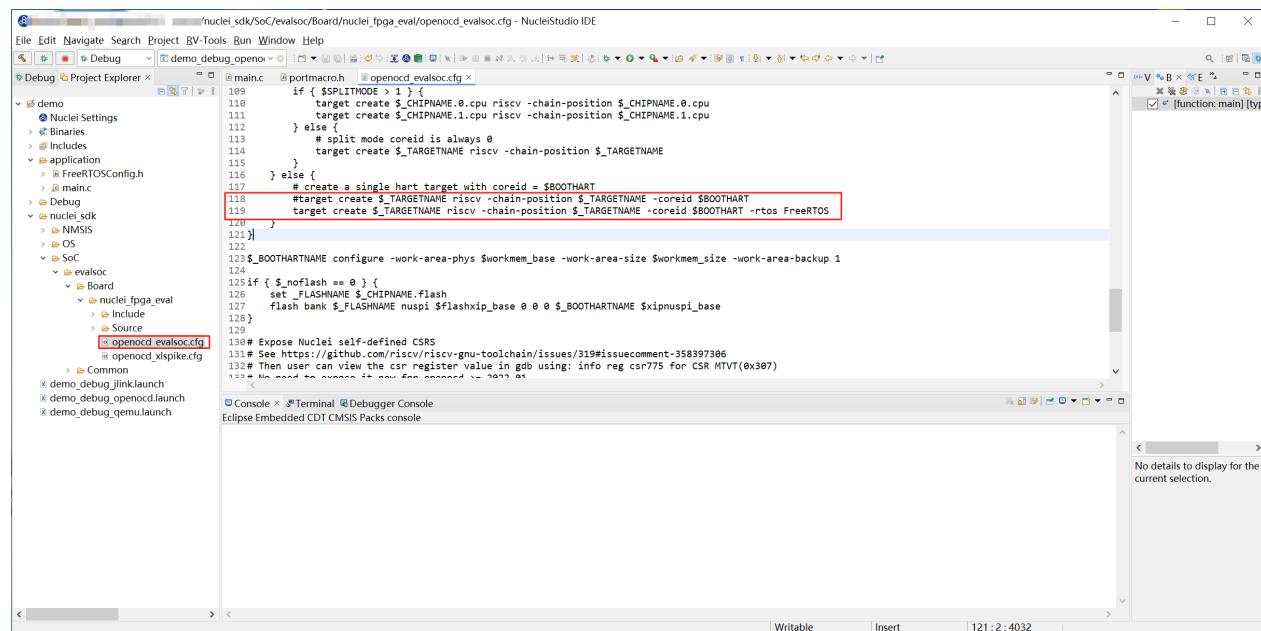
```
typedef uint32_t TickType_t;
#define portMAX_DELAY ( TickType_t )0xFFFFFFFFUL
/* RISC-V TIMER is 64-bit long */
//typedef uint64_t TickType_t;
//#define portMAX_DELAY
( TickType_t )0xFFFFFFFFFFFFFFFULL
```



step3：修改openocd_evalsoc.cfg内容

找到nuclei_sdk/SoC/evalsoc/Board/nuclei_fpga_eval/openocd_evalsoc.cfg,修改第118行内容

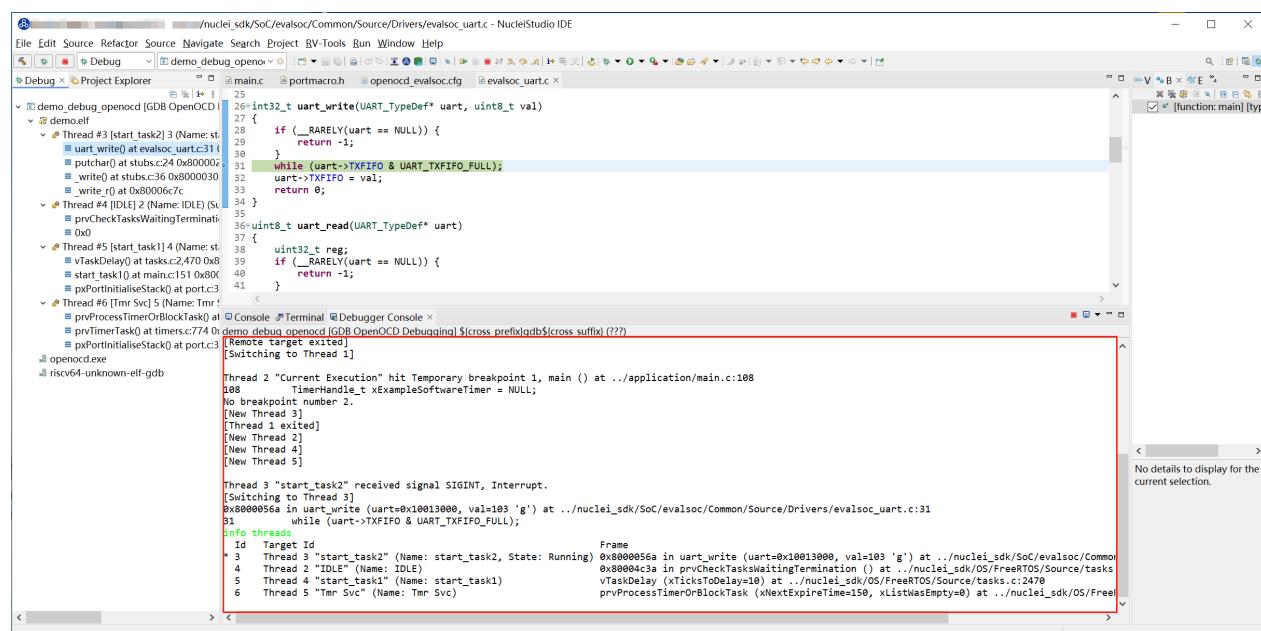
```
target create $_TARGETNAME riscv -chain-position $_TARGETNAME -
coreid $BOOTHART -rtos FreeRTOS
```



step4：openocd调试工程

Debug运行程序，打开Debugger Console视图。

在Debugger Console视图下输入info threads，回车。



使用说明

目前支持FreeRTOS,不支持Zephyr、ThreadX、UCOSII。

如何同时使用多个蜂鸟调试器进行调试¶

问题说明¶

芯来科技的蜂鸟调试器采用FTDI-FT232H作为USB接口转换芯片。 在同时连接多个蜂鸟调试器的情况下，如何区分不同的调试器？如何配置OpenOCD识别指定的蜂鸟调试器？

解决方案¶

FT232H提供了一个可配置的串号（Serial Number），可用于区分不同的调试器。

下载FT_PROG¶

FT_PROG是一个用于烧写FT232H片内的EEPROM的工具。可用于查看和修改FT232H的串号。

FT_PROG下载地址：<https://ftdichip.com/utilities/>

从这个页面中可以找到下载链接，如下图所示：

FT_PROG 3.12.61.670 - EEPROM Programming Utility

FT_PROG is a free EEPROM programming utility for use with FTDI devices. It is used for modifying EEPROM conte

PLEASE NOTE – The use of some of these utilities by an end user may result in a device being rendered useless.

FT_PROG is available for download by clicking [here](#).

The full FT_PROG User Guide can be downloaded [here](#).

Please Note: FT_PROG requires the Microsoft .NET Framework 4.0 installed on your system to run the application.
[id=17851&WT.mc_id=MSCOM_EN_US_DLC_DETAILS_121LSUS007996](#)

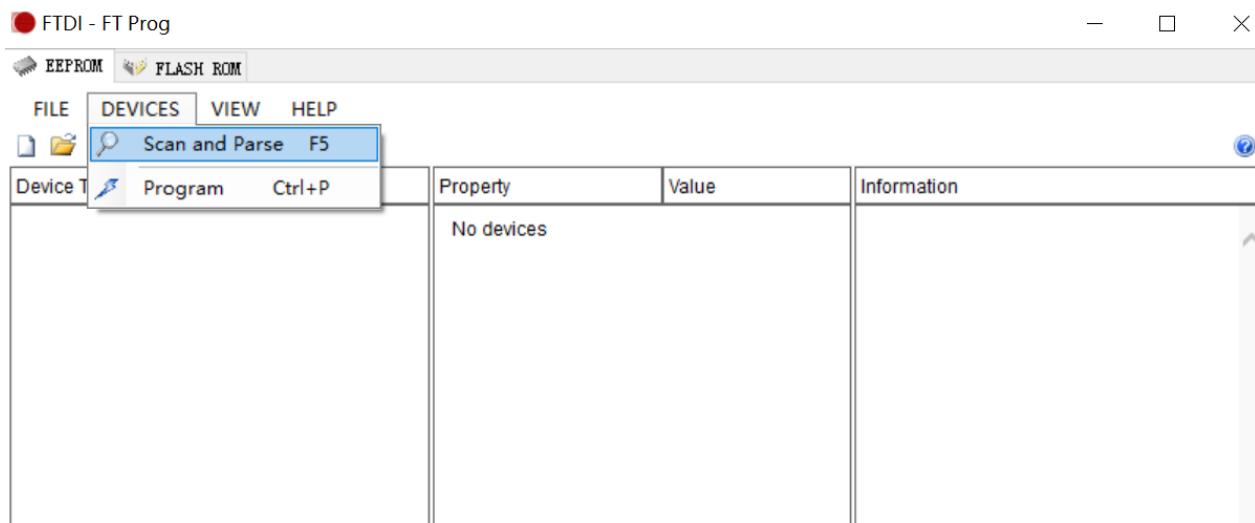
If your system does not have .NET 4.0 installed please download the file from the above link. To install, double click

下载并安装后，会在桌面生成FT_Prog工具的图标。

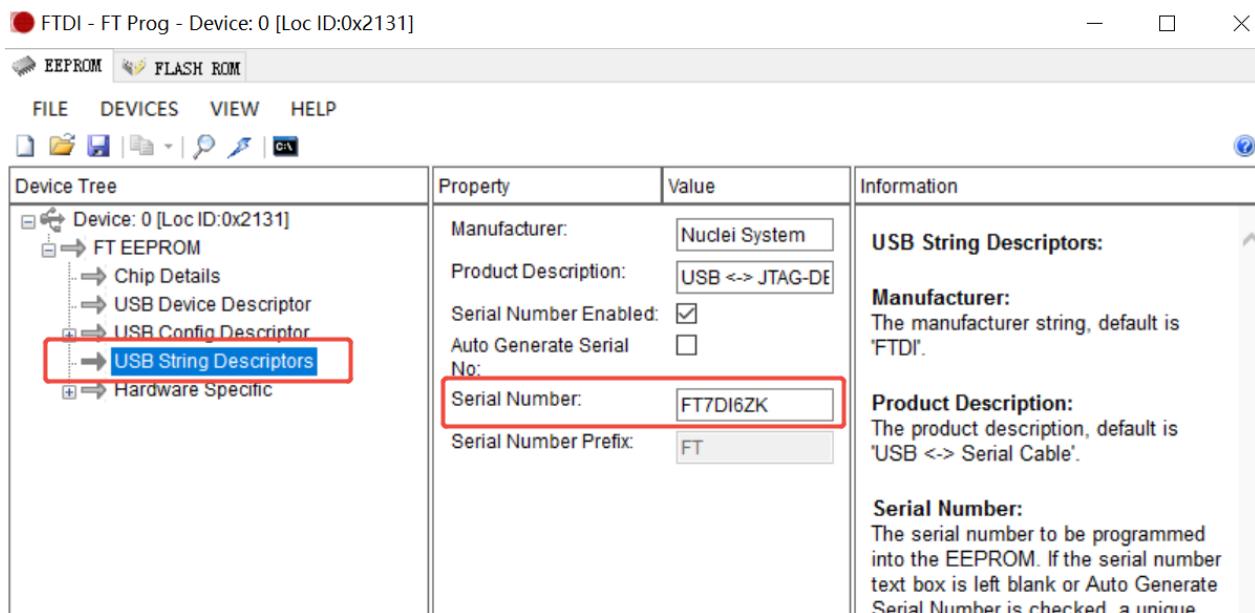
查看串号¶

使用FT_PROG工具，可以查看FT232H的串号。

1. 连接蜂鸟调试器 建议在无法区分多个蜂鸟调试器的情况下，先只连接一个蜂鸟调试器。
2. 打开FT_PROG工具 点击FT_PROG图标打开工具。
3. 扫描设备 点击菜单栏DEVICES中的Scan and Parse，扫描已连接的蜂鸟调试器。



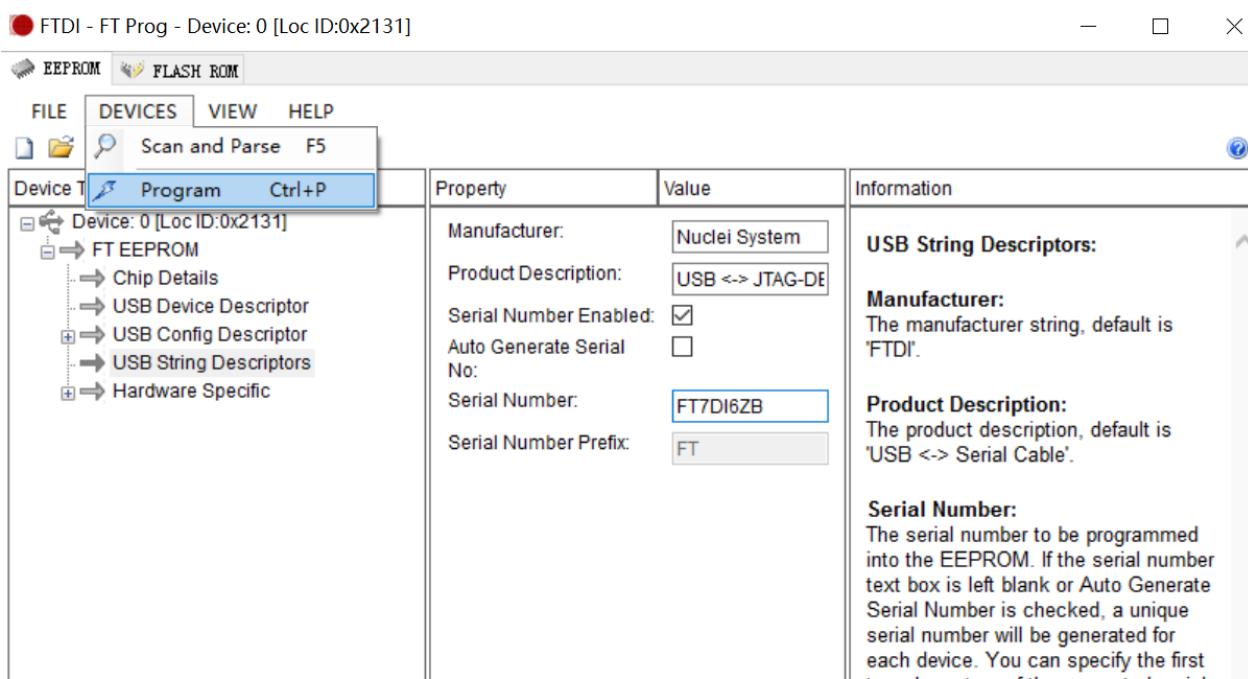
1. 查看串号 通过USB String Descriptors中的Serial Number可以查看蜂鸟调试器的串号。



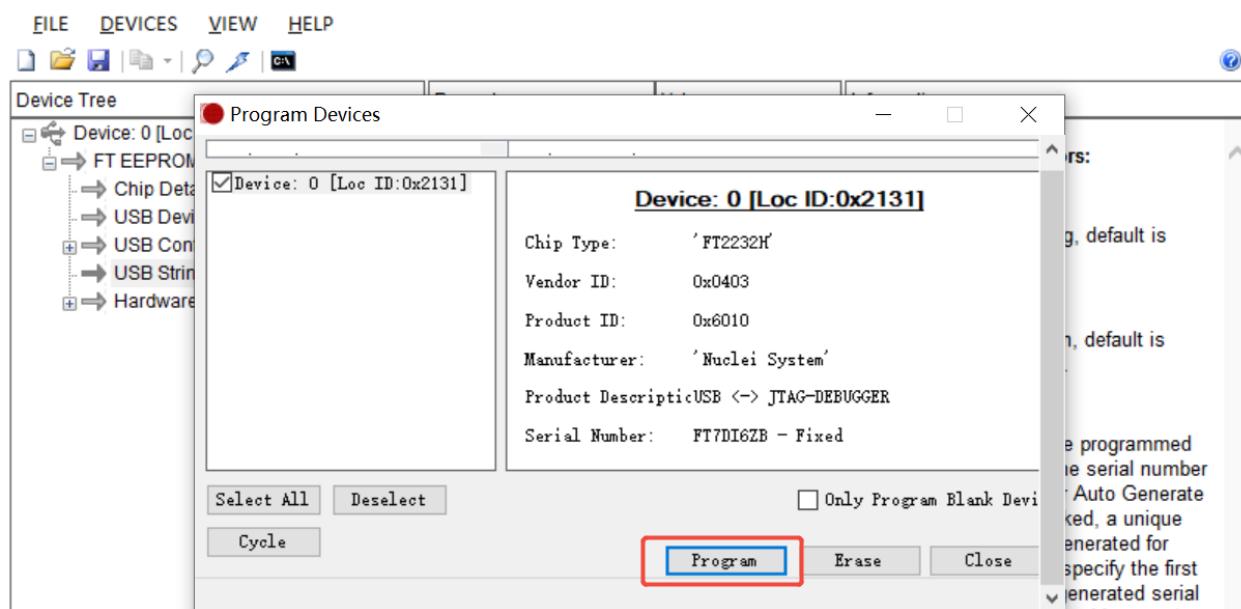
修改串号

在查看串号的页面可以修改蜂鸟调试器的串号。

比如下图中，我将原来的串号FT7DI6ZK改成了FT7DI6ZB



再通过菜单栏DEVICES中的Program选项，可以将修改后的串号写入到FT232H的EEPROM中。



注意：多个蜂鸟调试器需要分别设置不同的串号来进行区分。

更新OpenOCD配置

在使用Nuclei FPGA Evaluation Board时，打开Nuclei Studio中的工程OpenOCD配置文件，可以看到如下内容：

The screenshot shows the Nuclei Studio interface with the 'helloworld' project selected. On the left, the project tree includes 'Binaries', 'Includes', 'application', 'Debug', 'nuclei_sdk' (with 'NMSIS' and 'SoC' subfolders), 'evalsoc' (with 'Board', 'nuclei_fpga_eval', 'Include', and 'Source' subfolders), and 'Common'. Under 'evalsoc/Source', the file 'openocd_evalsoc.cfg' is highlighted with a red box labeled '1'. On the right, the content of 'openocd_evalsoc.cfg' is displayed:

```

1 # please use >= 2022.08 openocd
2 # some commands are changed to match latest openocd changes
3 adapter speed 1000
4
5 adapter driver ftdi
6 ftdi vid_pid 0x0403 0x6010
7 # for 2023.10 openocd, change oscan1_mode to nscan1_mode
8 ftdi oscan1_mode off
9
10## bindto 0.0.0.0 can be used to cover all available interfaces.
11## Uncomment bindto line to enable remote machine debug
12# bindto 0.0.0.0
13
14## you can also specify adapter serial to select a ftdi chip
15# adapter serial "FT6S9RD6"
16
17## Bind JTAG with specified serial number passed by JTAGSN
18# https://doc.nucleisys.com/nuclei_sdk/develop/buildsystem.html#jtagnsn
19if { [ info exists JTAGSN ] } {
20    puts "Bind JTAG with serial number $JTAGSN"
21    adapter serial $JTAGSN
22}
23

```

A red box labeled '2' highlights the line '14## you can also specify adapter serial to select a ftdi chip'.

Linux¶

修改openocd_evalsoc.cfg文件，即根据图中红框中的说明进行修改：

```
# 注意要去掉adapter serial前面的注释符号 #
adapter serial "<Serial Number>"
```

其中的<Serial Number>需要替换成实际的串号。

修改后的工程即可使用指定串号的蜂鸟调试器进行调试。

Windows¶

注意：在Windows系统下，需要在实际的串号后加上A才是有效的设置。

例如实际的串号是FT7DI6ZB，那么OpenOCD的配置文件需要添加如下设置：

```
adapter serial "FT7DI6ZBA"
```

参考资料¶

- [Nuclei Studio FAQs —— How to select correct FTDI debugger?](#)
- [FTDI Utilities](#)
- [User Guide for FTDI FT_PROG Utility](#)

Nuclei SDK基于evalsoc快速适配customsoc¶

方案说明¶

Nuclei Eval SoC(简称evalsoc)是芯来科技提供的一款用于评估芯来CPU的SoC，具有On-Chip SRAMs, UART, SPI等；

Nuclei SDK和Nuclei N100 SDK提供基于evalsoc的软件开发平台。客户通过evalsoc评估完芯来CPU后，希望在对应的SDK中快速适配为自己的SoC(本文称为customsoc)。

- Nuclei SDK 主要支持Nuclei 200/300/600/900/1000 series RISC-V CPU, 用于基于这些系列CPU的EvalSoC快速软件评估和开发
- Nuclei N100 SDK 主要支持Nuclei 100 series RISC-V CPU, 用于基于这些系列CPU的EvalSoC快速软件评估和开发

解决方案¶

根据需要移植适配的CPU系列，拉取最新的对应的SDK仓库或者直接使用cpu交付包中的SDK。

环境准备¶

适配修改¶

如果通过nuclei_gen工具生成了配套的文件，则直接替换同名文件即可，这样比较简单不出错；如果手动修改，则注意下文提到的文件和修改点

先不要改任何目录名，文件名，按步骤修改如下文件。

1 修改cpu特性描述宏文件¶

SoC/evalsoc/Common/Include/cpufeature.h 文件定义了customsoc支持的特性、参数相关的#define宏。CPU交付包中的nuclei_gen工具会自动生成该文件，直接替换即可。

2 修改cpu特性isa配置¶

SoC/evalsoc/cpufeature.mk 文件定义了customsoc的CORE(是否支持单/双精度浮点)ARCH_EXT(是否支持b和v扩展等)。CPU交付包中的nuclei_gen工具会自动生成该文件，直接替换即可。

3 修改链接地址的memory map¶

SoC/evalsoc/Board/nuclei_fpga_eval/Source/GCC/evalsoc.memory 描述了ILM/DLM/FLASH/SRAM/DDR 的BASE address和SIZE以及代码段的大小。CPU交付包中的nuclei_gen工具会自动生成该文件，直接替换即可。

4 修改openocd配置文件¶

openocd会通过jtag与cpu建立gdb server port，供gdb debug和load使用

SoC/evalsoc/Board/nuclei_fpga_eval/openocd_evalsoc.cfg 是openocd的配置描述文件。CPU交付包中的nuclei_gen工具会自动生成该文件，直接替换即可。关键参数如下：

```
# TODO: variables should be replaced by nuclei_gen
set workmem_base      0x80000000
set workmem_size      0x10000
set flashxip_base    0x20000000
set xipnuspi_base    0x10014000
```

5 修改Systimer频率¶

SoC/evalsoc/Common/Include/evalsoc.h 中修改SOC_TIMER_FREQ为customsoc的Systimer的真实频率（需咨询你们SoC硬件设计人员）

```
// 单位是hz 比如32768hz, 这里填32768
#define SOC_TIMER_FREQ           customsoc_systimer_freq
```

6 修改CPU主频¶

SoC/evalsoc/Common/Source/system_evalsoc.c 中，SystemCoreClock = get_cpu_freq()自动计算cpu主频(依赖Systimer)，可以直接修改为customsoc的主频

```
// 单位是hz 比如50Mhz, 这里填50000000
SystemCoreClock = customsoc_cpu_freq;
```

7 修改串口驱动¶

evalsoc的UART IP是评估版本

evalsoc_uart.c和evalsoc_uart.h里面的uart_xxx API名称不要修改，因为SoC/evalsoc/Common/Source/Stubs下的一些桩函数使用了uart的API

串口驱动位于 `SoC/evalsoc/Common/Source/Drivers/evalsoc_uart.c` , `SoC/evalsoc/Common/Include/evalsoc_uart.h` , 如果使用其它串口IP, 根据实际的串口寄存器定义适配。

8 修改串口波特率¶

`SoC/evalsoc/Common/Source/system_evalsoc.c`: `uart_init(SOC_DEBUG_UART, 115200);`
一般波特率为115200

9 修改_premain_init¶

一些在main函数之前执行的初始化可以放在这个函数

如果有 IOMUX 和 PLL 等其他相关的配置, 可以在 `SoC/evalsoc/Common/Source/system_evalsoc.c`: `_premain_init` 函数里面实现; 如果没有, 可以跳过

10 删除Nuclei内部使用的代码¶

`SoC/evalsoc/Common/Source/system_evalsoc.c`: `SIMULATION_EXIT` 宏定义是用于Nuclei内部仿真标记, 可以定义为空

```
#define SIMULATION_EXIT(ret)      {}
```

11 检查外设地址¶

建议CPU配置时不要修改, 保持与evalsoc一致

串口使用的`SOC_DEBUG_UART`定义为`UART0`

- 外设的Base address由`EVALSOC_PERIPS_BASE`决定, `EVALSOC_PERIPS_BASE`在`SoC/evalsoc/Common/Include/cpufeature.h`(由nuclei_gen工具生成, 拷贝覆盖即可)中定义, 一般无需再修改
- 外设的offset address在 `SoC/evalsoc/Common/Include/evalsoc.h` 中定义, 搜索 `Peripheral memory map`, 一般无需修改

```
#define UART0_BASE          (EVALSOC_PERIPH_BASE +  
0x13000)           /*!< (UART0) Base Address */  
#define QSPI0_BASE         (EVALSOC_PERIPH_BASE +  
0x14000)           /*!< (QSPI0) Base Address */  
#define UART0              ((UART_TypeDef *) UART0_BASE)
```

测试运行¶

如果以上修改完毕，就可以测试SoC能否正常工作了

这里因为是在evalsoc的基础上改的，还没有修改相关地方的名称为customsoc

所以仍然SOC=evalsoc BOARD=nuclei_fpga_eval

```
# Test helloworld application
## cd to helloworld application directory
cd application/baremetal/helloworld
## clean and build helloworld application for ncstar_eval board
make SOC=evalsoc BOARD=nuclei_fpga_eval clean all
## connect your board to PC and install jtag driver, open UART
terminal
## set baudrate to 115200bps and then upload the built application
## to the fpga board using openocd, and you can check the
## run message in UART terminal
make SOC=evalsoc BOARD=nuclei_fpga_eval upload
```

如果可以正常运行打印Hello World From Nuclei RISC-V Processor，那基本没有问题了。如果还需要运行更多case，请参考如下应用示例文档确认是否运行成功。

- Nuclei SDK: https://doc.nucleisys.com/nuclei_sdk/design/app.html
- Nuclei N100 SDK: https://doc.nucleisys.com/nuclei_n100_sdk/design/app.html

调整名称¶

重命名的地方有点多，这里就不列举了，最终保证编译通过就可以。

测试通过后，就可以把涉及evalsoc的文件名和目录名修改为customsoc，以及eval/EVAL开头的宏名/文件名替换成custom

```
# 修改完后，再次测试运行
make SOC=customsoc BOARD=nuclei_fpga_custom upload
```

至此，SDK就去掉了eval的logo,成为SDK for custom了。

精简代码¶

因为Nuclei SDK/N100 SDK支持Nuclei多款CPU系列的评估和内部测试，需要考虑非常多的场景，因此存在一些冗余代码，建议在阅读SDK文档并且熟悉代码框架后，再进行精简删除。

IAR工程¶

- IAR的工程有专门的链接脚本，位于SoC/evalsoc/Board/nuclei_fpga_eval/Source/IAR/*.icf IAR的链接脚本当前没有通过nuclei_gen工具生成，所以需要手动检查调整ROM_region32/ILM_region32/RAM_region32的base address和size，这里的from就是代表base address，size 表示该region的大小

```
define region ROM_region32 = mem:[from 0x20000000 size 0x800000];
define region ILM_region32 = mem:[from 0x80000000 size 0x10000];
define region RAM_region32 = mem:[from 0x90000000 size 0x10000];
```

- IAR的工程位于ideprojects/iar，也是prebuilt for evalsoc，在未调整名称之前是可以直接运行的如果经过了调整名称，路径和文件名都变化了，也需要重新新建工程，建议文本打开ewp文件，搜索“eval”关键词替换

```
diff --git a/ideprojects/iar/baremetal/coremark.ewp b/ideprojects/iar/baremetal/coremark.ewp
index 3eed66a8..17443eae 100644
--- a/ideprojects/iar/baremetal/coremark.ewp
+++ b/ideprojects/iar/baremetal/coremark.ewp
@@ -434,8 +434,8 @@
<option>
    <name>CCIIncludePath2</name>
    <state>$PROJ_DIR$...\\..\\..\\NMSIS\\Core\\Include</
state>
-        <state>$PROJ_DIR$...\\..\\..
\SoC\\evalsoc\\Board\\nuclei_fpga_eval\\Include</state>
-        <state>$PROJ_DIR$...\\..\\..
\SoC\\evalsoc\\Common\\Include</state>
+            <state>$PROJ_DIR$...\\..\\..
\SoC\\customsoc\\Board\\nuclei_fpga_custom\\Include</state>
+            <state>$PROJ_DIR$...\\..\\..
\SoC\\customsoc\\Common\\Include</state>
                <state>$PROJ_DIR$...\\..\\..
\\application\\baremetal\\benchmark\\coremark</state>
            </option>
```

IDE工程支持¶

如果希望Nuclei Studio IDE能支持custom soc，需要修改以下文件中涉及eval的名字，npk.yml的语法格式见 [2.4. Nuclei Studio NPK 介绍](#)

```
evalsoc/Common/npk.yml  
evalsoc/Board/nuclei_fpga_eval/npk.yml
```

参考资料¶

- [Nuclei 200/300/600/900/1000 Eval SoC](#)
- [Port your SoC into Nuclei SDK](#)
- [Nuclei 100 Eval SoC](#)
- [Port your SoC into Nuclei N100 SDK](#)

在链接脚本中使用OVERLAY命令¶

问题说明¶

CPU Core内的SRAM具有速度快，容量小，面积大的特点。在嵌入式系统中，这部分Core内的RAM很可能无法放下所有的函数。为了解决这个问题，有一种方案是在链接脚本中使用OVERLAY命令。

GNU ld 提供的OVERLAY命令，可以在同一块内存区域上“叠放”多个段（Section）。若干个段可以共享运行时的VMA（Virtual Memory Address），只是在运行时需要手动管理这些Overlay的段的加载和卸载。

那么如何在Nuclei Studio IDE中使用OVERLAY命令呢？本文将提供一个示例程序演示如何使用OVERLAY命令。

解决方案¶

示例程序¶

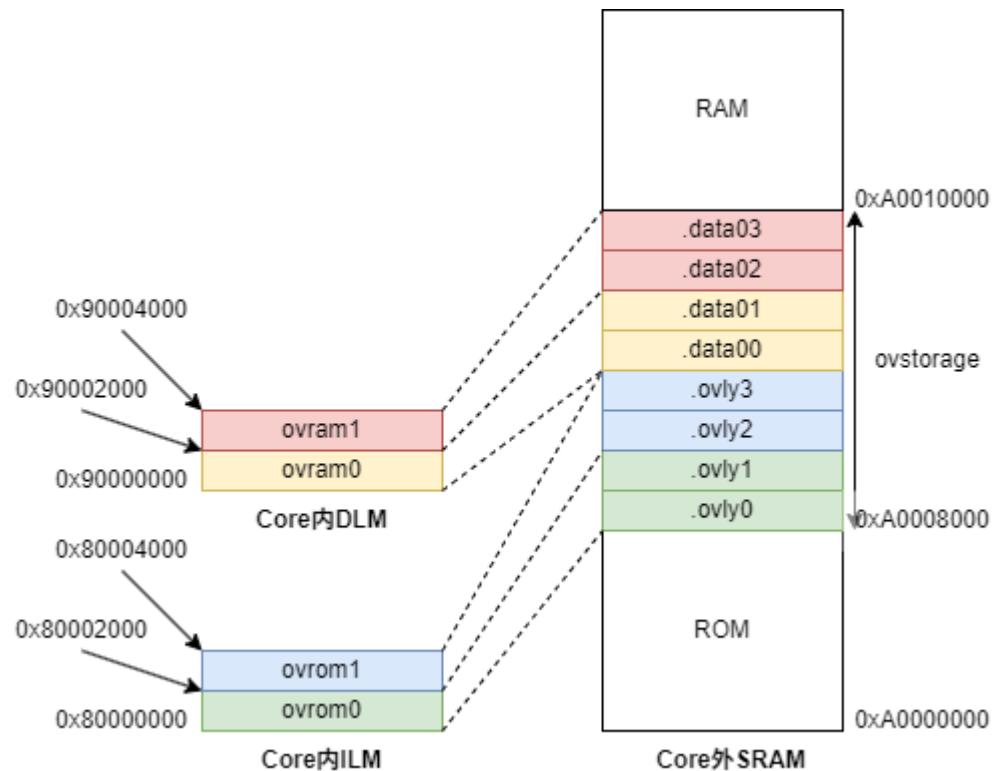
[demo_overlay](#)是基于 Nuclei Studio IDE 2025.02 创建的示例工程，支持Linux和Windows两种平台，演示了如何在链接脚本中使用OVERLAY命令。

示例工程中的代码与[Overlay Sample Program](#) 中提供的代码基本一致，主要区别是在main函数中增加了测试结果的打印，另外根据evalsoc的地址映射关系对链接脚本做了修改。

```
├── bar.c
├── baz.c
├── foo.c
├── grbx.c
├── overlays.c
└── ovlymgr.c
    └── ovlymgr.h
```

原始代码可以从[bminor/binutils-gdb/gdb/testsuite/gdb.base](#)获取。

Overlay布局¶



在我们的evalsoc上，如果程序无法完全放在Core内的ILM/DLM上，就可以采取上图中的方法 将部分Section以Overlay的形式动态加载到ILM/DLM中运行。

.ovlyx和.data0x 共8个Section的LMA（Load Memory Address）都位于Core外的SRAM上，但它们的VMA都在Core内，并且有部分重叠。

编写链接脚本¶

在MEMORY命令中先划分出需要用到的Memory区域，比如这里4个Core内的区域ovrom0, ovrom1, ovram0, ovram1和Core外的ovstorage区域。这些Memory区域的大小都可以根据实际的代码和数据的大小进行调整。

```
MEMORY
{
    rom (rxa!w) : ORIGIN =
    SRAM_MEMORY_BASE, LENGTH =
    SRAM_MEMORY_ROM_SIZE
    ovstorage (rwa) : ORIGIN = SRAM_MEMORY_BASE +
    SRAM_MEMORY_ROM_SIZE, LENGTH = SRAM_OVLY_STORAGE_SIZE
    ram (wxa!r) : ORIGIN = SRAM_MEMORY_BASE + SRAM_MEMORY_ROM_SIZE +
    SRAM_OVLY_STORAGE_SIZE, LENGTH = SRAM_MEMORY_SIZE -
    SRAM_MEMORY_ROM_SIZE - SRAM_OVLY_STORAGE_SIZE
    ovrom0 (rwx) : ORIGIN = ILM_MEMORY_BASE, LENGTH =
    ILM_OVLY_SIZE0
```

```

ovrom1 (rwx) : ORIGIN = ILM_MEMORY_BASE + ILM_OVLY_SIZE0,
LENGTH = ILM_OVLY_SIZE1
ovram0 (rwx) : ORIGIN = DLM_MEMORY_BASE, LENGTH =
DLM_OVLY_SIZE0
ovram1 (rwx) : ORIGIN = DLM_MEMORY_BASE + DLM_OVLY_SIZE0,
LENGTH = DLM_OVLY_SIZE1
}

```

OVERLAY需要放在SECTION命令中实现。例如下方的代码就是将.ovly0和.ovly1两个段放到ovrom0所表示的同一个VMA地址区间中，同时它们的LMA则是连续地位于ovstorage所表示的地址区间中。

```

OVERLAY :
{
    .ovly0 { *foo.o(.text .text.*) }
    .ovly1 { *bar.o(.text .text.*) }
} >ovrom0 AT>ovstorage

```

参考[Automatic Overlay Debugging](#)，链接脚本中的以下代码则是将Overlay的段的VMA, LMA和数量以_ovly_table和_novlys变量的形式供C代码访问；在ovlymgr.c中进一步实现动态加载和卸载Section的功能。

```

/* _ovly_table used for gdb debug overlay sections */
_ovly_table = .;
_ovly0_entry = .;
LONG(ABSOLUTE(ADDR(.ovly0)));
LONG(SIZEOF(.ovly0));
LONG(LOADADDR(.ovly0));
LONG(0);
...
_novlys = .;
LONG((._novlys - _ovly_table) / 16);

```

测试结果¶

观察编译后生成的map文件，可以看到相应的代码段和数据段都按照预期的VMA和LMA进行了分配。

例如.ovly0和.ovly1具有相同的VMA 0x80000000，同时它们的LMA分别为0xa0008000和0xa0008028。

.ovly0	0x80000000	0x28 load address 0xa0008000
foo.o(.text .text.)		

```

.text.foo      0x80000000    0x28 ./application/foo.o
              0x80000000          foo
              [!provide]           PROVIDE
(__load_start_ovly0 = LOADADDR (.ovly0))
              [!provide]           PROVIDE
(__load_stop_ovly0 = (LOADADDR (.ovly0) + SIZEOF (.ovly0)))

.ovly1        0x80000000    0x28 load address 0xa0008028
*bar.o(.text .text.*)
.text.bar      0x80000000    0x28 ./application/bar.o
              0x80000000          bar
              [!provide]           PROVIDE
(__load_start_ovly1 = LOADADDR (.ovly1))
              [!provide]           PROVIDE
(__load_stop_ovly1 = (LOADADDR (.ovly1) + SIZEOF (.ovly1)))

```

main函数中通过OverlayLoad切换调用不同的函数，并在最后将每个函数的返回值累加，校验累加后的结果。

```

/* load .text and .data for `foo` */
OverlayLoad (0);
OverlayLoad (4);
a = foo (1);
/* load .text and .data for `bar` */
OverlayLoad (1);
OverlayLoad (5);
b = bar (1);
/* load .text and .data for `baz` */
OverlayLoad (2);
OverlayLoad (6);
c = baz (1);
/* load .text and .data for `grbx` */
OverlayLoad (3);
OverlayLoad (7);
d = grbx (1);

e = a + b + c + d;
if (e != ('f' + 'o' + 'o'
+ 'b' + 'a' + 'r'
+ 'b' + 'a' + 'z'
+ 'g' + 'r' + 'b' + 'x')) {
printf ("Overlay Test FAIL\r\n");
} else {
printf ("Overlay Test PASS\r\n");
}

```

通过QEMU仿真或者利用FPGA开发板进行测试，可以看到如下的结果。其中Overlay Test PASS表明结果符合预期。

```
Nuclei SDK Build Time: Sep 25 2025, 17:02:19
Download Mode: SRAM
CPU Frequency 16003235 Hz
CPU HartID: 0
Overlay Test PASS
```

注意事项¶

1. 数据段Overlay在要替换Section时，需要保存数据，也就是需要“卸载”的操作，将数据保存到外部SRAM中；而代码段是只读的，所以不需要Unload。
2. 在示例工程中，ILM/DLM都不会经过Cache，所以不需要考虑Cache一致性的问题。但如果Overlay的Section所在的VMA是Cacheable的区域，则一般都需要考虑Cache一致性的问题，除非ICache和DCache之间有硬件支持的Snoop。更多细节可以参考示例工程中ovlymgr.c的实现。

参考资料¶

- [How Overlays Work](#)
- [Overlay Commands](#)
- [Automatic Overlay Debugging](#)
- [Overlay Sample Program](#)