Gowin PicoRV32

IDE 软件参考手册

IPUG910-1.2,2020-06-01

免责声明

本文档并未授予任何知识产权的许可，并未以明示或暗示，或以禁止发言或其它方式授予任何知识产权许可。除高云半导体在其产品的销售条款和条件中声明的责任之外，高云半导体概不承担任何法律或非法律责任。高云半导体对高云半导体产品的销售和／或使用不作任何明示或暗示的担保，包括对产品的特定用途适用性、适销性或对任何专利权、版权或其它知识产权的侵权责任等，均不作担保。高云半导体对文档中包含的文字、图片及其它内容的准确性和完整性不承担任何法律或非法律责任，高云半导体保留修改文档中任何内容的权利，恕不另行通知。高云半导体不承诺对这些文档进行适时的更新。

版本信息

日期	版本	说明
2020/01/16	1.0	初始版本。
2020/03/06	1.1	● MCU 支持 Wishbone 总线接口的外部设备 GPIO； ● MCU 支持扩展 AHB 总线接口； ● MCU 支持片外 SPI-Flash 下载及运行； ● MCU 支持外部设备 SPI-Flash 读、写和擦除功能； ● MCU 支持 Hardware Stack Protection 和 Trap Stack Overflow 功能。
2020/06/01	1.2	● 支持 MCU 软件在线调试功能； ● 增强 MCU 内核中断处理功能； ● 优化 MCU 内核指令。

目录

图目录

表目录

1 GMD 安装与配置

GOWIN MCU Designer 编译软件，支持 Gowin_PicoRV32 的软件编程设计、编译、下载和调试。

高云半导体官网提供 GOWIN MCU Designer 软件安装包下载：http://www.gowinsemi.com.cn/prodshow.aspx。

GOWIN MCU Designer 软件安装与配置，及 Gowin_PicoRV32 的调试仿真器驱动软件（Olimex Debugger Driver）安装与配置，请参考 SUG549, GOWIN MCU Designer 用户指南。

注！

建议使用 GOWIN MCU Designer V1.1 及以上版本。

2 软件编程模板

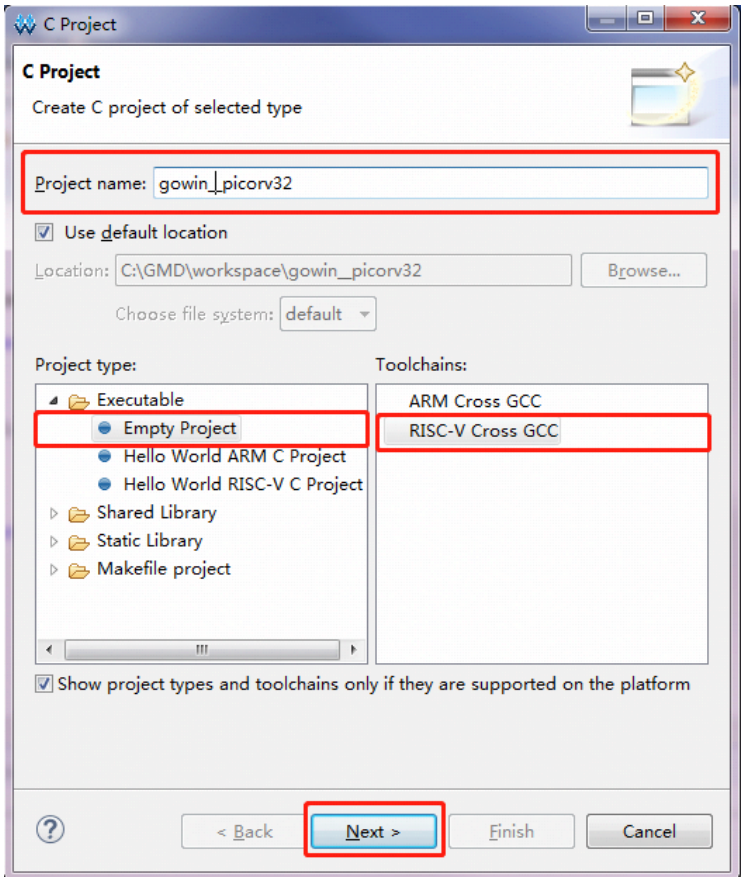
2.1 模板工程创建

2.1.1 工程创建

选择菜单栏“File > New > C Project”，如图 2-1 所示。

1. 建立项目名称和项目位置；
2. 选择项目类型“Empty Project”；
3. 选择工具链“RISC-V Cross GCC”；
4. 单击“Next”。

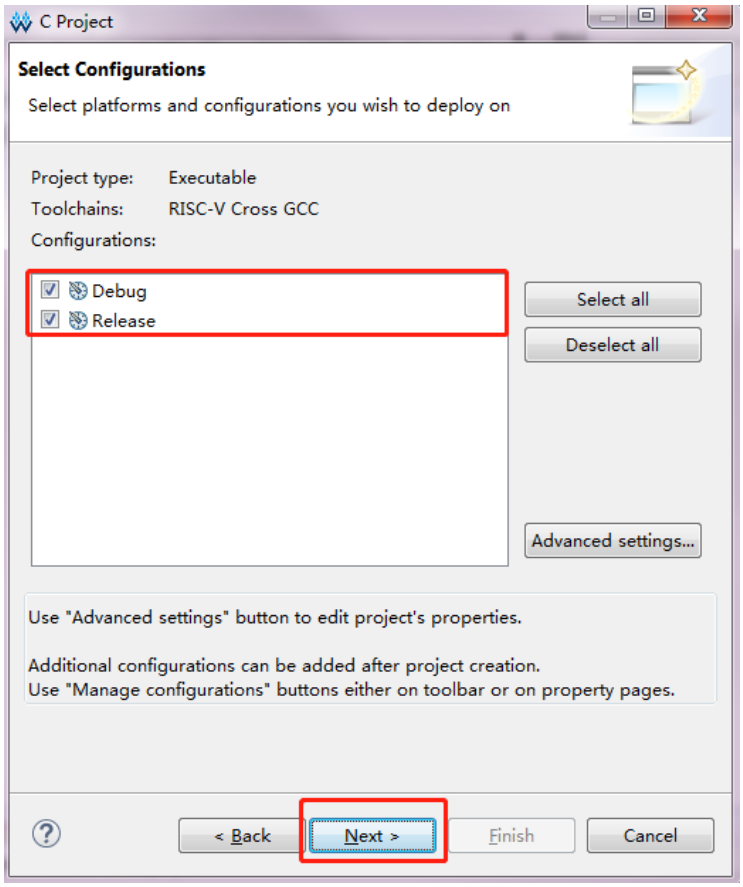
图 2-1 工程创建



2.1.2 平台类型配置

平台类型配置，选择“Debug”和“Release”，单击“Next”，如图 2-2 所示。

图 2-2 平台类型配置

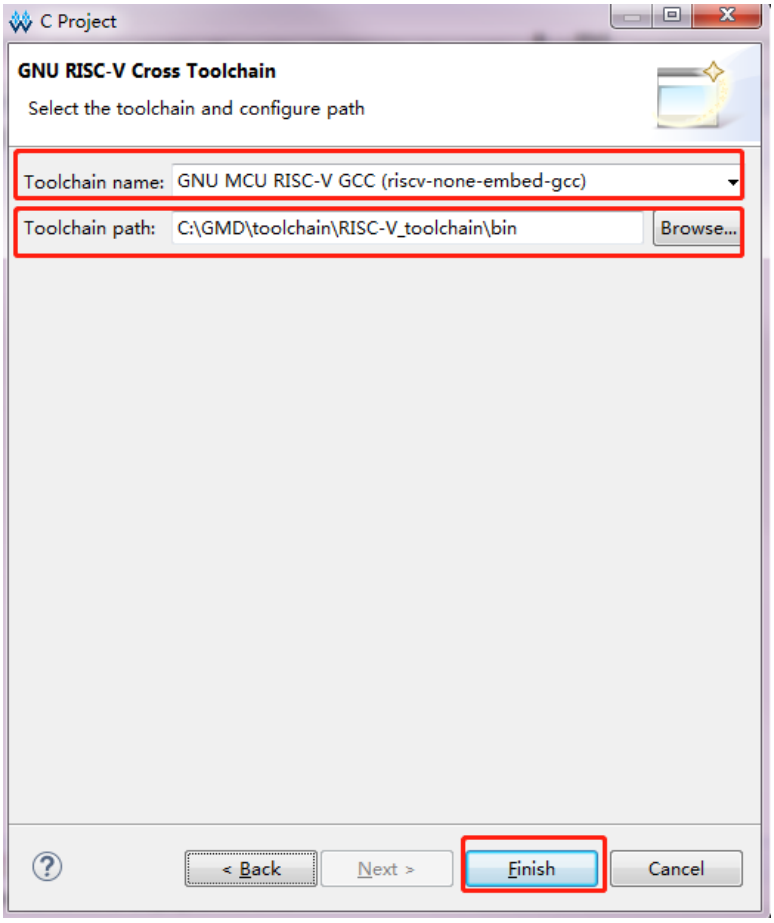


2.1.3 编译工具链和路径配置

选择交叉编译工具链 riscv-none-embed-gcc 和其所在路径，单击“Finish”，完成 Gowin_PicoRV32 软件编程模板工程的创建，如图 2-3 所示。

默认配置为 GOWIN MCU Designer 软件中 RISC-V 交叉编译链安装路径，如需自定义 RISC-V 交叉工具链，则请手动修改指定工具链位置。

图 2-3 编译工具链和路径配置



2.1.4 导入软件编程设计

完成 Gowin_PicoRV32 软件编程新工程创建后，选择工作空间 workspace 下新建的项目工程，导入软件编程设计。

以软件开发工具包参考设计为例，软件编程设计目录结构及代码定义，如下所示。

- startup：Gowin_PicoRV32 内核引导启动文件
- system：Gowin_PicoRV32 内核及系统定义
- peripherals：外部设备驱动函数库
- config：下载方式配置文件
- linker：Flash 链接器
- user：用户应用代码

软件编程设计目录下如有代码更新，请在视图“Project Explorer”当前工程中右键选择“Refresh”，更新 GMD 工程模板中的文件及代码。

2.2 模板工程配置

GOWIN MCU Designer 中，在视图“Project Explorer”选择当前工程，右键选择“Properties”，选择选项卡“C/C++ Build > Settings > Toolchains”选项，配置 Gowin_PicoRV32 模板工程的参数选项。

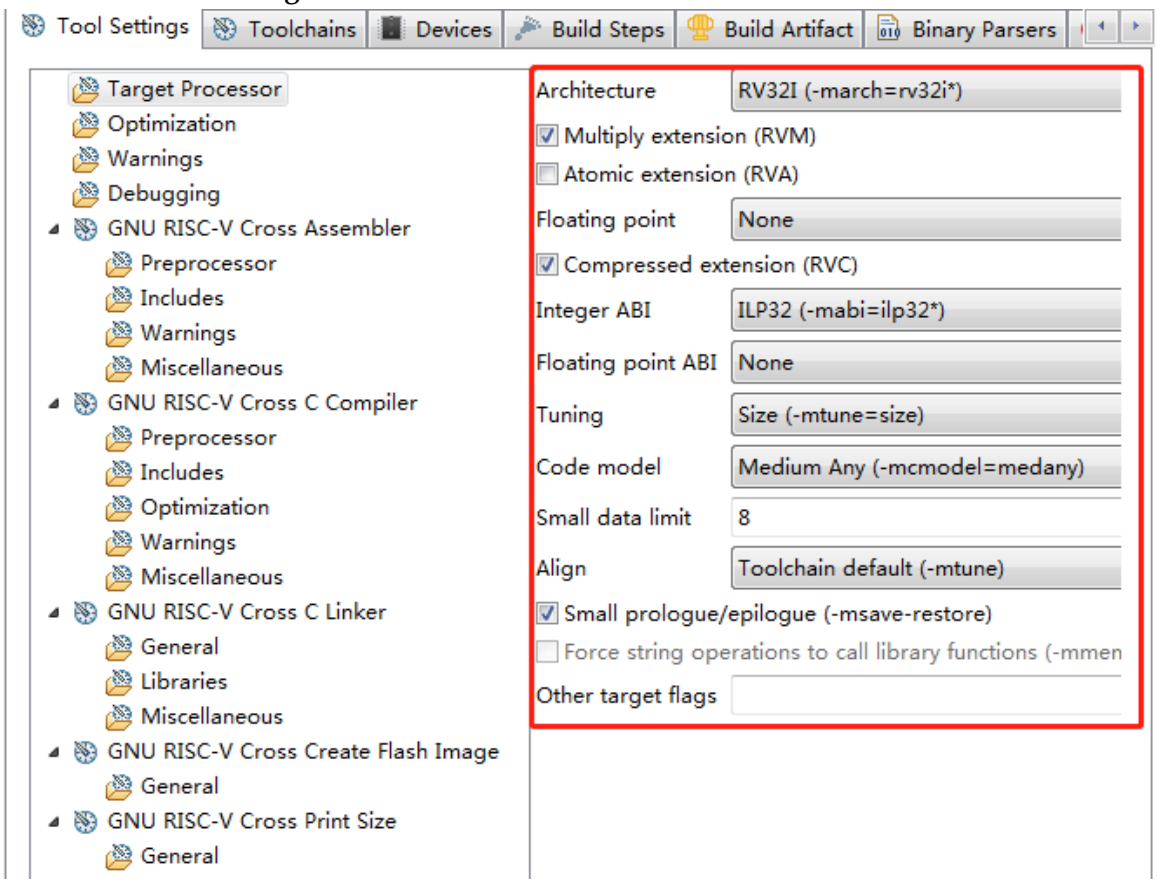
Gowin_PicoRV32 必须配置以下参数选项：

- Target Processor
- Optimization
- GNU RISC-V Cross Assembler
  - Includes
- GNU RISC-V Cross C Compiler
  - Includes
- GNU RISC-V Cross C Linker
  - General
- GNU RISC-V Cross Create Flash Image
  - General

2.2.1 配置 Target Processor

配置“Target Processor”选项，该选项配置，如图 2-4 所示。

图 2-4 配置 Target Processor



- Architecture
  RV32I(-march=rv32i*)， Gowin_PicoRV32 仅支持 RISC-V32 位整型指令，所以选择 RV32I 选项。
- Multiply extension (RVM)
  如果需要使用 RV32M 扩展，使能 Multiply extension(RVM)选项。同时

在 Gowin_PicoRV32 硬件设计 IP Core Generator 的 Gowin PicoRV32 Core 参数配置中，使能 Support RV32M Extends 选项，否则编译包含 RISC-V 快速乘法指令时,会导致 Gowin_PicoRV32 运行错误。

- Atomic extension (RVA)
  Gowin_PicoRV32 不支持原子指令扩展，因此禁用该选项。
- Floating point
  Gowin_PicoRV32 不支持浮点指令，因此设置该选项为 None。
- Compressed extension (RVC)
  如果需要使用 RV32C 扩展，则请使能 Compressed extension (RVC)选项。同时在 Gowin_PicoRV32 硬件设计 IPCore Generator 的 Gowin PicoRV32 Core 参数配置中，使能 Support RV32C Extends 选项，否则编译包含 RISC-V 16 位压缩指令时,会导致 Gowin_PicoRV32 运行错误。
- Integer ABI
  设置 RISC-V 目标平台所支持的 ABI 函数调用规则。Gowin_PicoRV32 为 32 位 RISC-V 架构处理器平台，且不支持硬件浮点指令，因此配置该选项为 ILP32 (-mabi=ilp32*)。32 位 RISC-V 架构处理器数据类型宽度，如表 2-1 所示。

表 2-1 32 位 RISC-V 架构处理器数据类型宽度

C 语言数据类型	32 位 RISC-V 架构数据类型宽度（单位：字节）
char	1
short	2
int	4
long	4
long long	8
void *	4
float	4
double	8
long double	16

- Tuning
  指定编译工具链 GCC 为其调整代码性能的目标处理器名称。该选项不支持 Gowin_PicoRV32，因此配置为 Size(-mtune=size)。
- Code model
  设置参数-mcmodel，该参数指定程序的寻址范围，选择 Medium Any (-mcmodel=medany)。(-mcmodel=medany)选项用于指定该程序的寻址范围可以在任意的 4GB 空间内，寻址空间不预定，应用相对灵活。
- Small data limit
  设置参数-msmall-data-limit，该参数指定可以放置到小数据区域的全局变量和静态变量的最大 Size (以字节为单位)。该参数设置为 8。
- Align
  设置是否避免产生非对齐内存访问的操作。Gowin_PicoRV32 不支持快

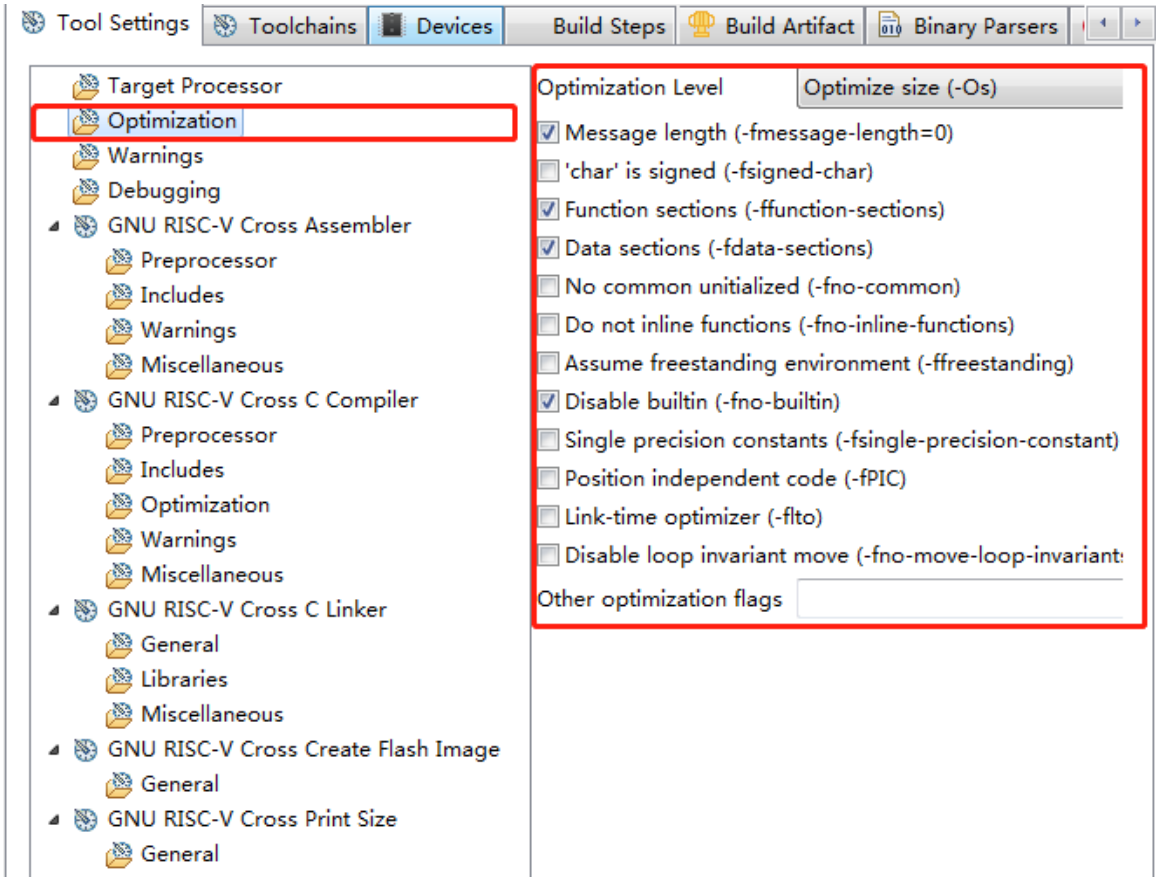
速非对齐访问，因此建议设置为 Strict(-mstrict-align)。

- Small prologue/epilogue(-msave-restore)
  如果使能该选项，则设置为调用 Size 最小、但速度更慢的起始和返回代码的库函数。默认使用快速的内联代码。
- Other target flags 选项中可以手动添加如下配置：
  - Allow use of PLTs(-mplt)
    如果使能该选项，则使用 PLT 生成中断控制代码，否则不使用 PLT。
  - Floating-point divide/sqrt instructions(-mfdiv)
    如果使能该选项，则使用硬件浮点除法和开方指令，需要处理器支持 RV32F 和 RV32D 指令集扩展和浮点寄存器。Gowin_PicoRV32 不支持 RV32F 和 RV32D 指令集扩展，因此禁用该选项。
  - Integer divide instructions(-mdiv)
    如果使能选项，则使用整型除法硬件指令，需要处理器支持 RV32M 指令集扩展。Gowin_PicoRV32 支持 RV32M 指令集扩展，所以可以使能 Multiply extension(RVM)选项，同时在 Gowin_PicoRV32 硬件设计 IP Core Generator 的 Gowin PicoRV32 Core 参数设置中，使能 Support RV32M Extends 选项。
  - -mpreferred-stack-boundary=num
    栈边界对齐为 2num 字节边界对齐。如果未指定，则默认值为 24（即 16 字节或 128 位）对齐。如果配置该选项，则在构建所有模块时都需要使用该选项（包括库，系统库和起始模块）。
  - -mexplicit-relocs / -mno-exlicit-relocs
    处理符号地址时，使用或不使用汇编程序重定向操作符。另一种配置是使用汇编宏，这可能会限制优化。
  - -mrelax
  - -mno-relax
    使用链接器松弛来减少实现符号地址所需的指令数量。默认是使用连接器的松弛。
  - -memit-attribute / -mno-emit-attribute
    输出或不输出 RISC-V 属性信息到 ELF 对象中。该特性需要 binutils 2.32 版本。
  - -malign-data=type
    控制 GCC 如何对齐数组、结构体、联合体等类型的变量和常量。支持的类型值为‘xlen’和‘natural’。其中，‘xlen’使用寄存器宽度作为对齐值，‘natural’使用自然对齐。默认值为‘xlen’。

2.2.2 配置 Optimization

配置“Optimization”选项，该选项配置，如图 2-5 所示。

图 2-5 配置 Optimization



- Optimization Level
  通过-O 等级来设置优化等级，优化编译 Size、运行速度和编译时间等。可选择的包括-O0、-O1、-O2、-O3、-Os、-Og，-O0/-O1/-O2/-O3 优化等级逐级提高。
  - -O0：不做优化；
  - -Os：在-O2 的基础上，关闭导致编译 Size 增大的选项，实现对编译 Size 的最优化；
  - -Og：在-O0 的基础上，增加一些适用于调试的编译选项，Gowin _PicoRV32 暂不支持片上调试，因此建议采用其他优化策略。
- -fmessage-length=n
  将错误信息以每行 n 个字符的形式显示在控制台窗口中。如果设置为 0，则关闭换行功能，一条错误信息显示为一行。默认为-fmessage-length=0。建议禁用该选项。
- ‘char’ is signed (-fsigned-char)
  设置 char 类型数据为有符号数。
- Function sections (-ffunction-sections) / Data sections (-fdata-sections)
  - 若目标支持任意分段，则令每个函数或数据项在输出文件中开辟单独的段，使用函数名或数据项的名称作为输出段的名称。
  - 若连接器可以执行改进指令空间中引用的局部性的优化，则可以使能

该选项。

- 若与连接器垃圾收集（连接器 –gc-sections 选项）一起使用，则最后生成执行文件时自动删除没有使用的函数段和数据项段，产生更小编译 Size。

注！

只有在能产生显著效果时才使能该选项。当指定该选项时，编译器和连接器将创建更大 Size 的对象和可执行文件，降低编译速度，影响代码生成。该选项放置编译器和汇编器使用翻译单元内的相对位置进行优化，因为这些位置在连接之前是未知的。这种优化的一个案例是将调用放宽为短调用指令。建议禁用该选项，并在连接器设置中使能 -gc-sections 选项，以减少编译 Size。

- No common unitialized (-fno-common)
  fno-common 选项指定编译器将未初始化的全局变量放到目标文件的 BSS 段中。这将阻止连接器合并暂定定义，因此如果在多个编译单元中定义了相同的变量，则会出现多定义错误。建议禁用该选项。
- Do not inline functions (-fno-inline-functions)
  若使能该选项，则除了使用 always_inline 属性标记的函数之外，不展开任何内联函数。关闭优化时，此为默认设置。也可使用 noinline 属性标记单个函数，以避免该函数的内联。
- Assume freestanding environment (-ffreestanding)
  断言编译目标时的一个独立环境，独立环境中可能不存在标准库，程序启动可能不是主函数。显著案例是操作系统内核。相当于-fno-hosted。
- Disable builtin (-fno-builtin)
  - 不识别不以“_builtin_”作为前缀的内置函数。受影响的函数，包括使用-ansi 或-std 选项(用于严格的 ISO C 一致性)时不是内置函数的函数，因为没有 ISO 标准含义。
  - GCC 通常生成特殊代码来更有效地处理某些内置函数，例如，对 alloca 的调用可能变成直接调整堆栈的单个指令，对 memcpy 的调用可能变成内联复制循环。生成的代码通常更小、更快，但是由于函数调用不再以这样的方式出现，不能在这些调用上设置断点，也不能通过连接不同的库来更改函数的行为。
  - 此外，当一个函数被识别为内置函数时，GCC 可能会使用关于该函数的信息来警告调用该函数的问题，或者生成更有效的代码，即使生成的代码仍然包含对该函数的调用。例如，当内建 printf 并且已知 strlen 不修改全局内存时，会用-Wformat 给出对 printf 的错误调用的警告。
- Single precision constants (-fsingle-precision-constant)
  将浮点常量作为单精度数据处理。
- Position independent code (-fPIC)
  如果目标机器支持，则生成适合在共享库中使用的与位置无关的代码 (PIC)。Gowin_PicoRV32 不支持 PIC，因此禁用该选项。
  - Link-time optimizer (-flto)
    该选项运行标准的连接时间优化器。
  - 使用源代码调用时，生成 GIMPLE (GCC 的内部表示之一)并将其写

入对象文件中的特殊 ELF 节。当对象文件连接在一起时，所有的函数体都从 ELF 部分读取并实例化，如同是同一翻译单元的一部分。

- 使用连接时优化器，应在编译时和最终连接期间指定-flto 和优化选项，建议使用相同的选项编译参与相同连接的所有文件，并在连接时指定这些选项。

- Disable loop invariant move (-fno-move-loop-invariants)
选择是否取消 RTL 循环优化器中的循环不变动作传递，若优化等级设置为-O1 或更高等级（-Og 除外），自动开启 RTL 循环优化器中的循环不变动作传递。

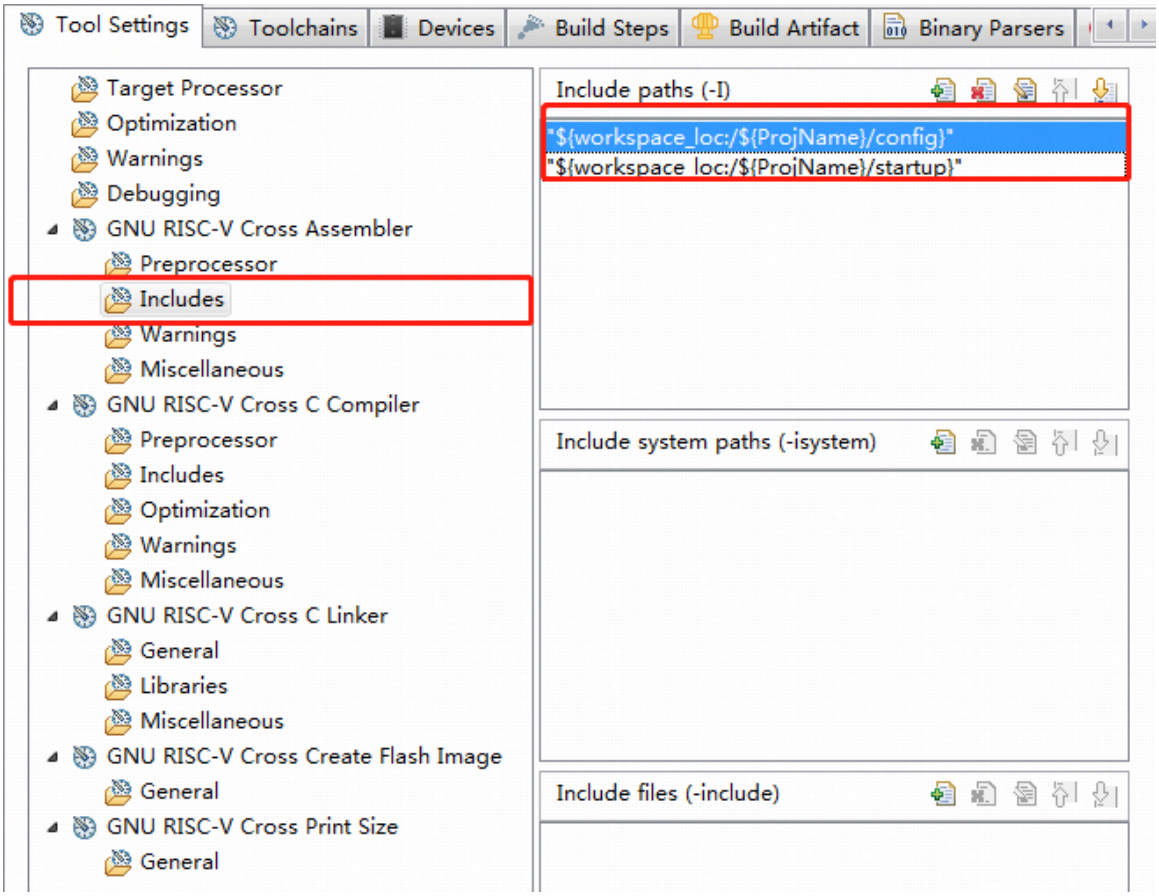
2.2.3 配置 GNU RISC-V Cross Assembler > Includes

配置“GNU RISC-V Cross Assembler > Includes > Include paths (-I)”选项，该选项配置汇编引用文件路径，如图 2-6 所示。

以软件开发工具包参考设计为例，汇编引用文件路径配置，如下所示。

- "${workspace_loc:/${ProjName}/startup}"
- "${workspace_loc:/${ProjName}/config}"

图 2-6 配置 GNU RISC-V Cross Assembler > Includes



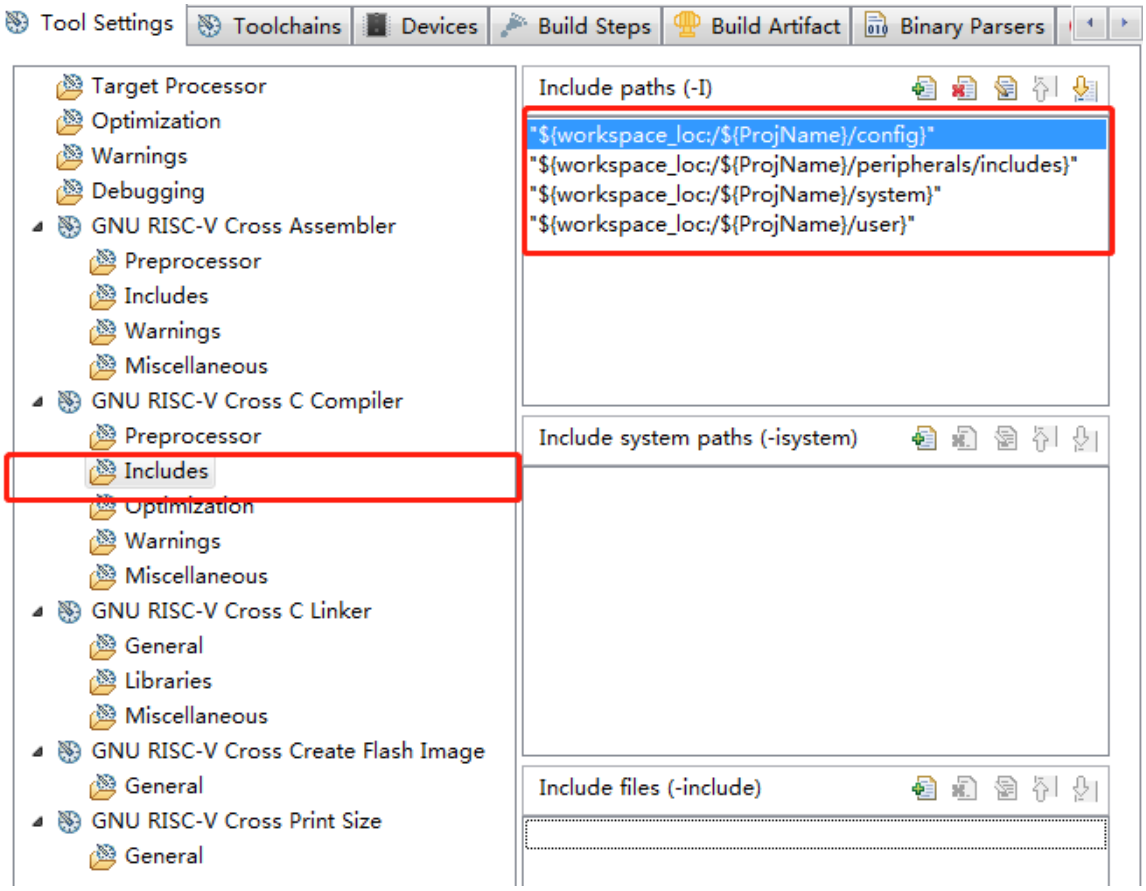
2.2.4 配置 GNU RISC-V Cross C Compiler > Includes

配置“GNU RISC-V Cross C Compiler > Includes > Include paths (-I)”选项，该选项配置 C 头文件引用路径，如图 2-7 所示。

以软件开发工具包参考设计为例，C 头文件引用路径配置，如下所示。

- "${workspace_loc:/${ProjName}/config}"
- "${workspace_loc:/${ProjName}/peripherals/includes}"
- "${workspace_loc:/${ProjName}/system}"
- "${workspace_loc:/${ProjName}/user}"

图 2-7 配置 GNU RISC-V Cross C Compiler > Includes



2.2.5 配置 GNU RISC-V Cross C Linker

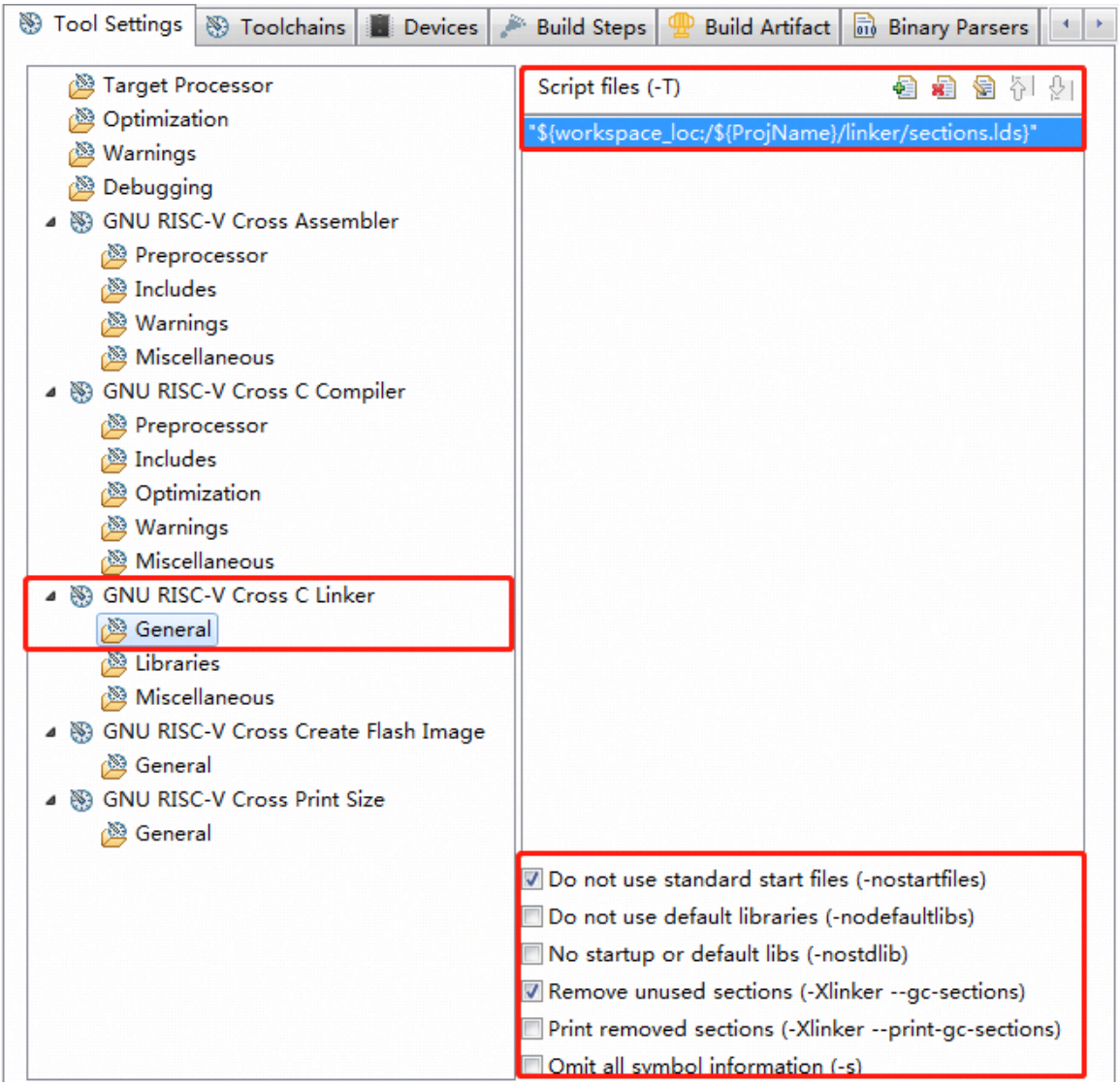
配置“GNU Cross RISC-V C Linker > General > Script files (-T)”选项，该选项配置 Gowin_PicoRV32 Flash 链接器 sections.lds 或 sections_xip.lds 或 sections_debug.lds，如图 2-8 所示。

根据 Gowin_PicoRV32 硬件设计 IPCore Generator 中 ITCM > Boot Mode 配置方式：

- 如果 Boot Mode 配置为 MCU boot from external Flash and run in ITCM 或 MCU boot and run in ITCM，则选择 sections.lds 作为 Flash 链接器，如"${workspace_loc:/${ProjName}/linker/sections.lds}"
- 如果 Boot Mode 配置为 MCU boot and run in external Flash，则选择 sections_xip.lds 作为 Flash 链接器，如 "${workspace_loc:/${ProjName}/linker/sections_xip.lds}"
- 如果 Boot Mode 配置为 MCU boot from external Flash and run in ITCM 或 MCU boot and run in ITCM，并且执行软件在线调试，则选择

sections_debug.lds 作为 Flash 链接器，如 "${workspace_loc:/${ProjName}/linker/sections_debug.lds}"

图 2-8 配置 GNU RISC-V Cross C Linker



- Do not use standard start files (-nostartfiles)
  设置连接时不使用标准启动文件。Gowin_PicoRV32 必须使用自定义的起始文件，因此必须使能该选项。
- Do not use default libraries (-nodefaultlibs)
  设置连接时不使用标准系统库，只有被选择的库传递给连接器。指定系统库连接的选项(如-static-libgcc 或-shared-libgcc)将被忽略。正常使用标准启动文件，除非使用-nostartfiles。编译器可能会生成对 memcmp、memset、memcpy 和 memmove 的调用。
- No startup or default libs (-nostdlib)
  - 连接时不使用标准的系统启动文件或库。
  - 没有任何启动文件，只有用户指定的库被传递给连接器，并且指定系统库连接的选项(如-static-libgcc 或-shared-libgcc)将被忽略。

- 该选项设置为禁用状态。

- Remove unused sections (-Xlinker –gc-sections)
  - 该选项删除没有调用的段。
  - 该选项配合编译器优化设置的-ffunction-sections 和-fdata-sections 选项，可在连接时删除未调用的函数和变量，进一步减小编译 Size。
- Print removed sections (-Xlinker –print-gc-sections)
  当使能 Remove unused sections (-Xlinker –gc-sections)时，可以使能该选项，编译时输出被优化删除的段的名称，标记被优化删除的段。
- Omit all symbol informations (-s)
  - 从可执行文件中删除所有符号表和重定位信息。
  - 建议选择使能 Do not use standard start files (-nostartfiles)和 Remove unused sections (-Xlinker –gc-sections)选项。

2.2.6 配置 GNU RISC-V Cross Create Flash Image

配置 GNU RISC-V Cross Create Flash Image > General 选项，该选项配置，如图 2-9 所示。

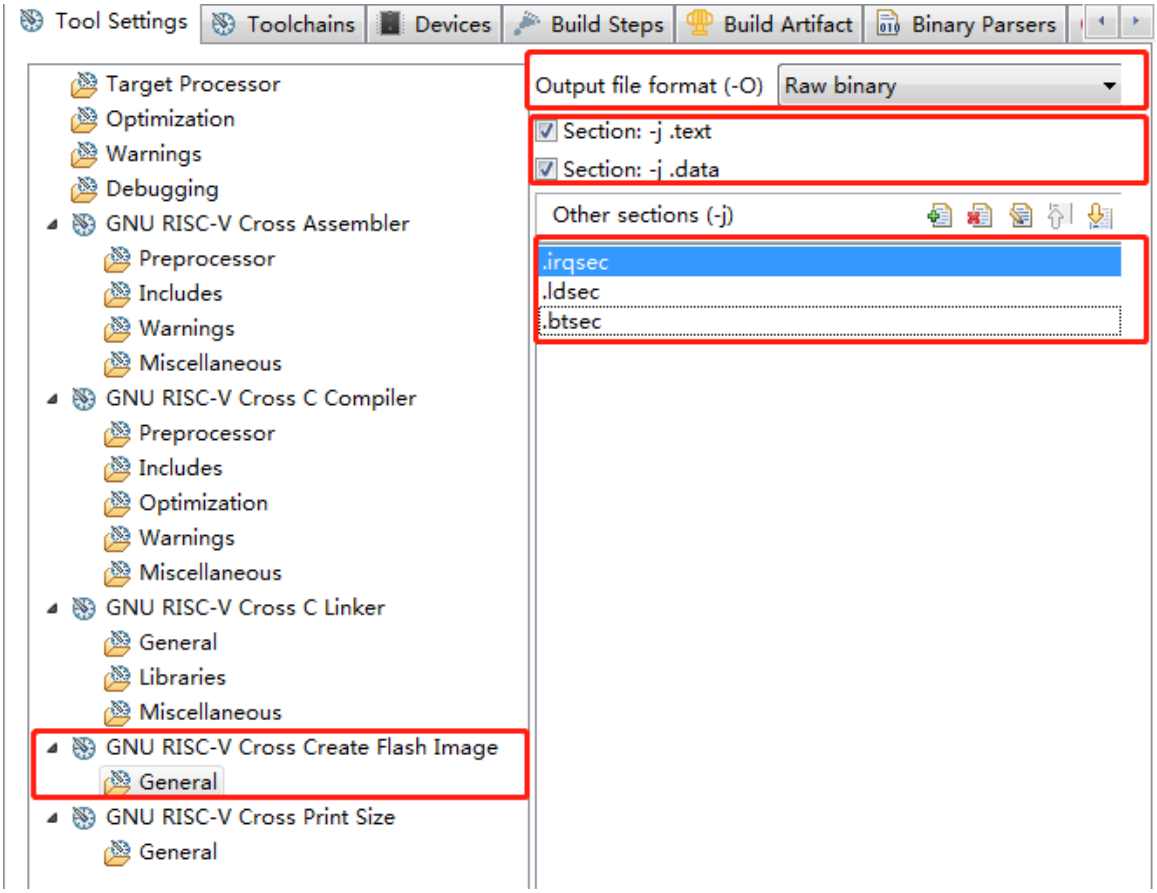
以软件开发工具包参考设计为例，该选项配置，如下所示。

- Output file format (-O)选项，配置输出文件格式为 Raw binary
- 使能 Section: -j .text 选项
- 使能 Section: -j .data 选项

如果在工程中有自定义的段，并将函数或变量映射到自定义的段中，则需在 Other sections (-j)中添加这些自定义的段，如 Gowin_PicoRV32 自定义以下段：

- .irqsec
- .ldsec
- .btsec

图 2-9 配置 GNU RISC-V Cross Create Flash Image



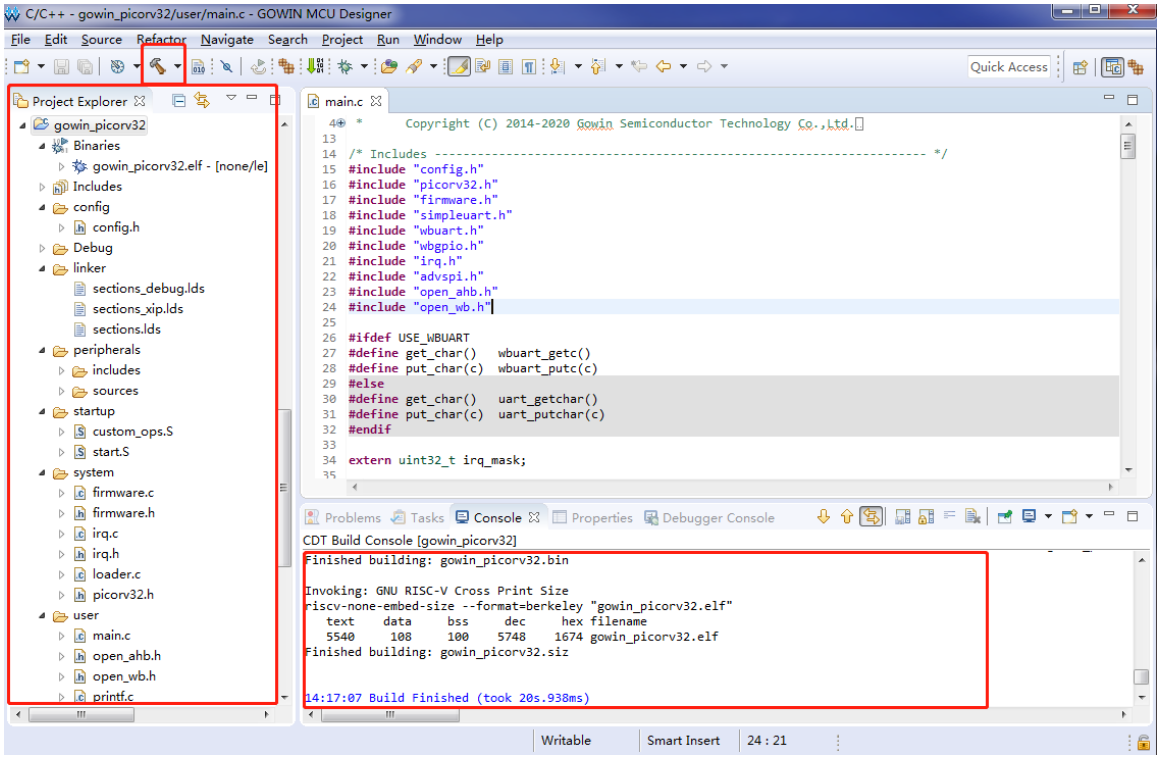
2.3 编译

根据 Gowin_PicoRV32 硬件设计 IP Core Generator 中 ITCM > Boot Mode 配置方式，修改 config.h 启动参数宏定义 BUILD_MODE：

- MCU boot and run in ITCM ：#define BUILD_MODE BUILD_LOAD
- MCU boot from external Flash and run in ITCM：#define BUILD_MODE BUILD_BURN
- MCU boot and run in external Flash：#define BUILD_MODE BUILD_XIP

完成模板工程配置、用户配置和代码编写后，编译工程，单击工具栏编译按钮 “ ”，编译产生软件设计二进制 BIN 文件，如图 2-10 所示。

图 2-10 编译



2.4 下载

完成 Gowin_PicoRV32 软件编程设计编译后，软件编程设计下载方法，请参考 IPUG913, Gowin_PicoRV32 软件下载参考手册。

2.5 在线调试

完成 Gowin_PicoRV32 软件设计二进制 BIN 文件下载后，如果用户软件应用设计出现问题，可以连接开发板与 Olimex 调试仿真器（或 RiscV 指令集架构处理器所支持的其他调试仿真器），在线调试当前 MCU 软件设计。

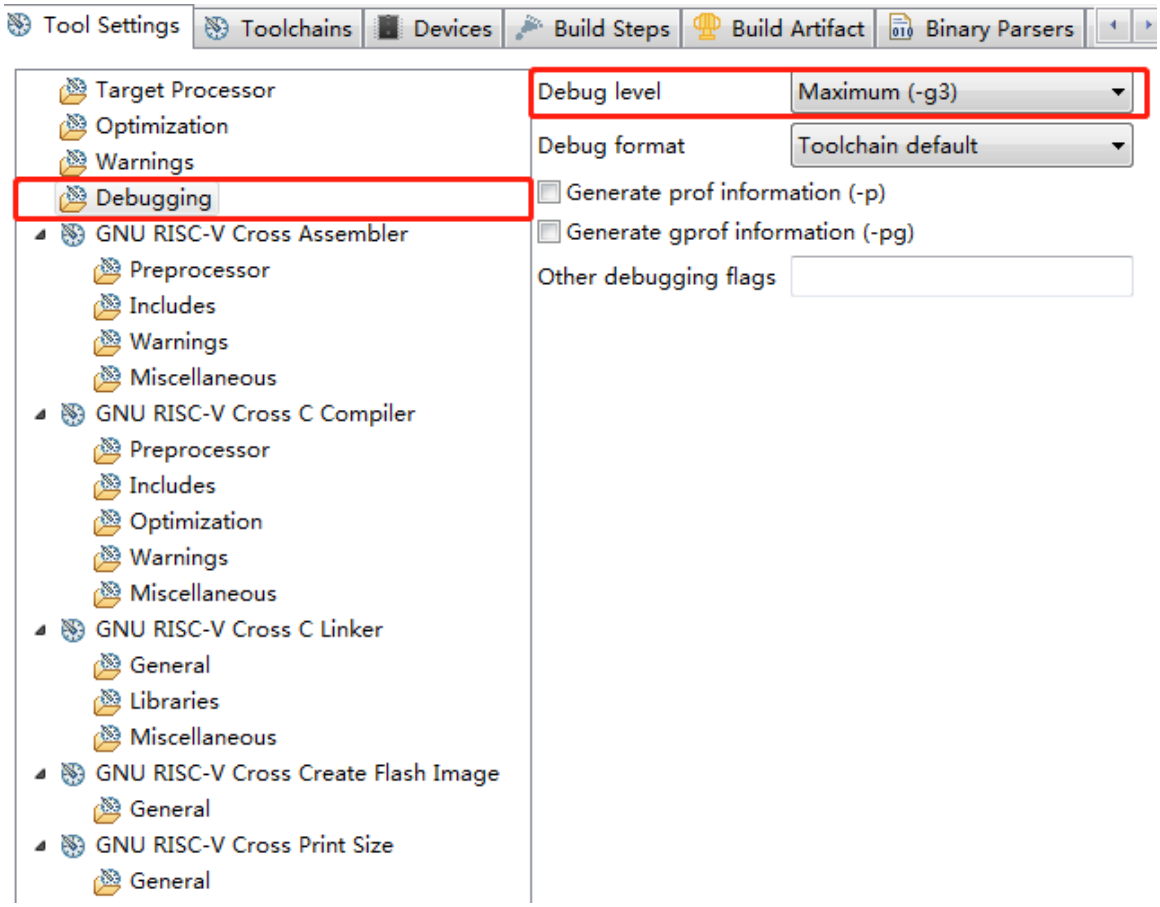
Gowin_PicoRV32 软件调试流程，包括：

- 配置软件调试等级
- 配置软件调试选项
- 连接调试仿真器
- 启动软件调试

2.5.1 配置软件调试等级

在 Project Explorer 视图中，选择当前工程，右键选择“Properties > Debugging > Debug level”选项，该选项配置，如图 2-11 所示。

图 2-11 配置 Debugging



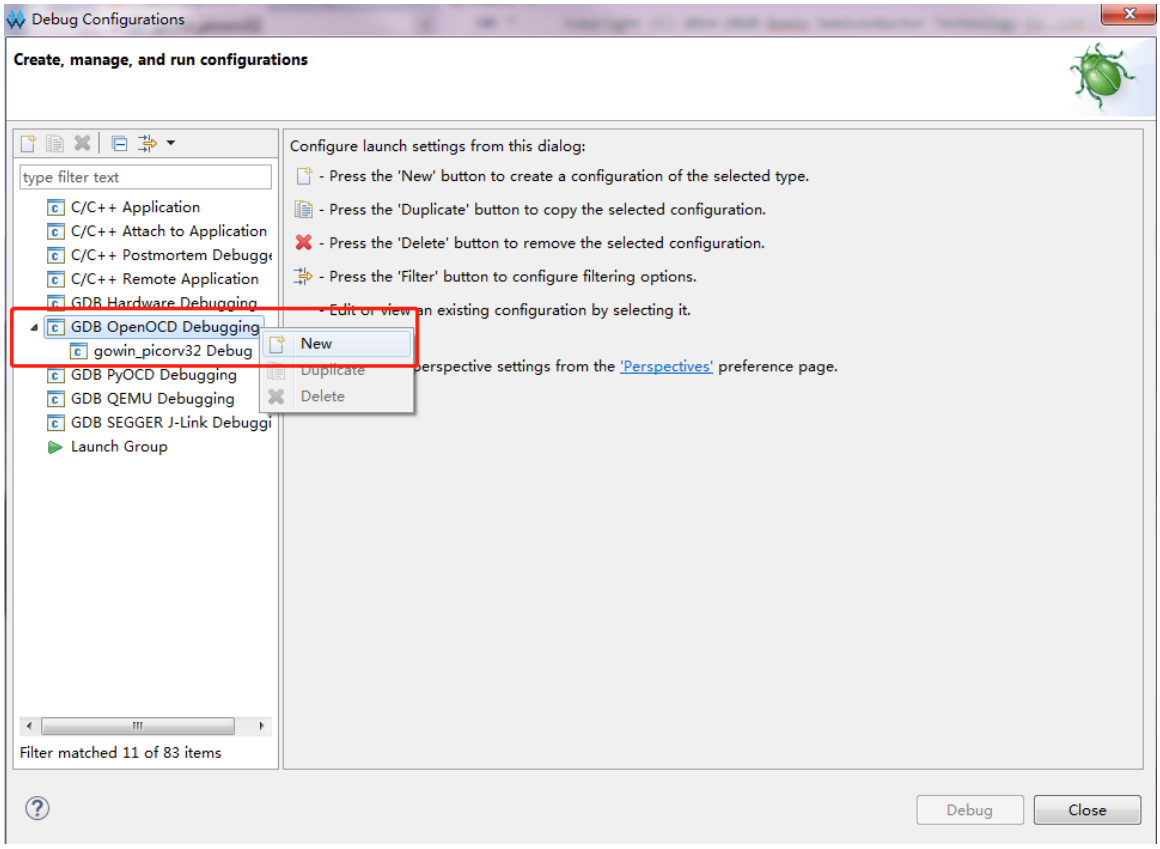
Debug level 配置，包括以下选项：

- None：无调试等级配置
- Minimal (-g1)：最小调试等级配置
- Default (-g)：默认调试等级配置
- Maximum (-g3)：最大调试等级配置

2.5.2 配置软件调试选项

选择菜单栏“Run > Debug Configurations > GDB OpenOCD Debugging”选项，右键选择“New”选项，建立当前工程的软件调试配置选项，如图 2-12 所示。

图 2-12 建立软件调试配置选项

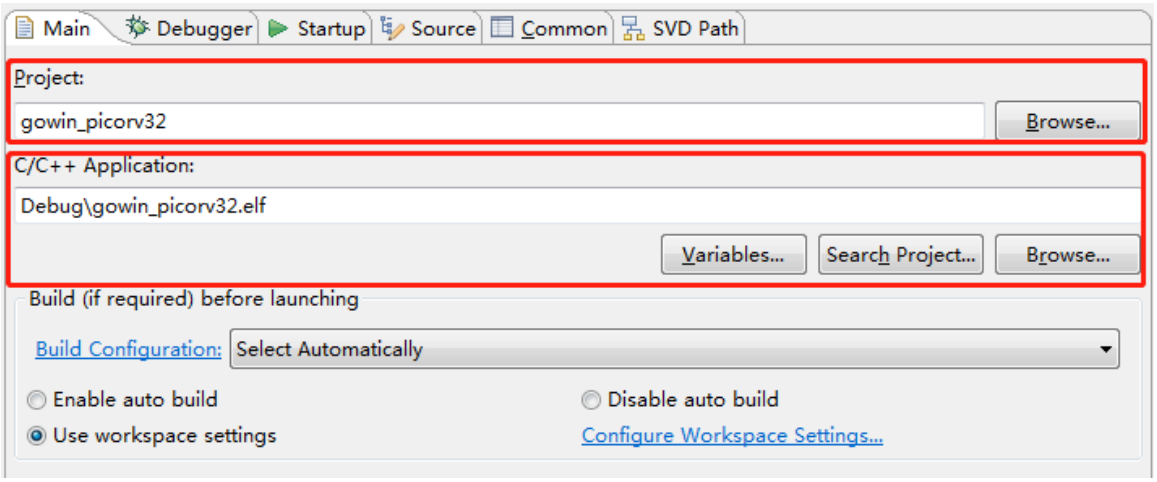


选择已建立的软件调试选项，配置 Main、Debugger、Startup 等调试选项。

配置 Main 选项

选择“Main”选项，配置当前工程的 Project 和 C/C++ Application 等参数，如图 2-13 所示。

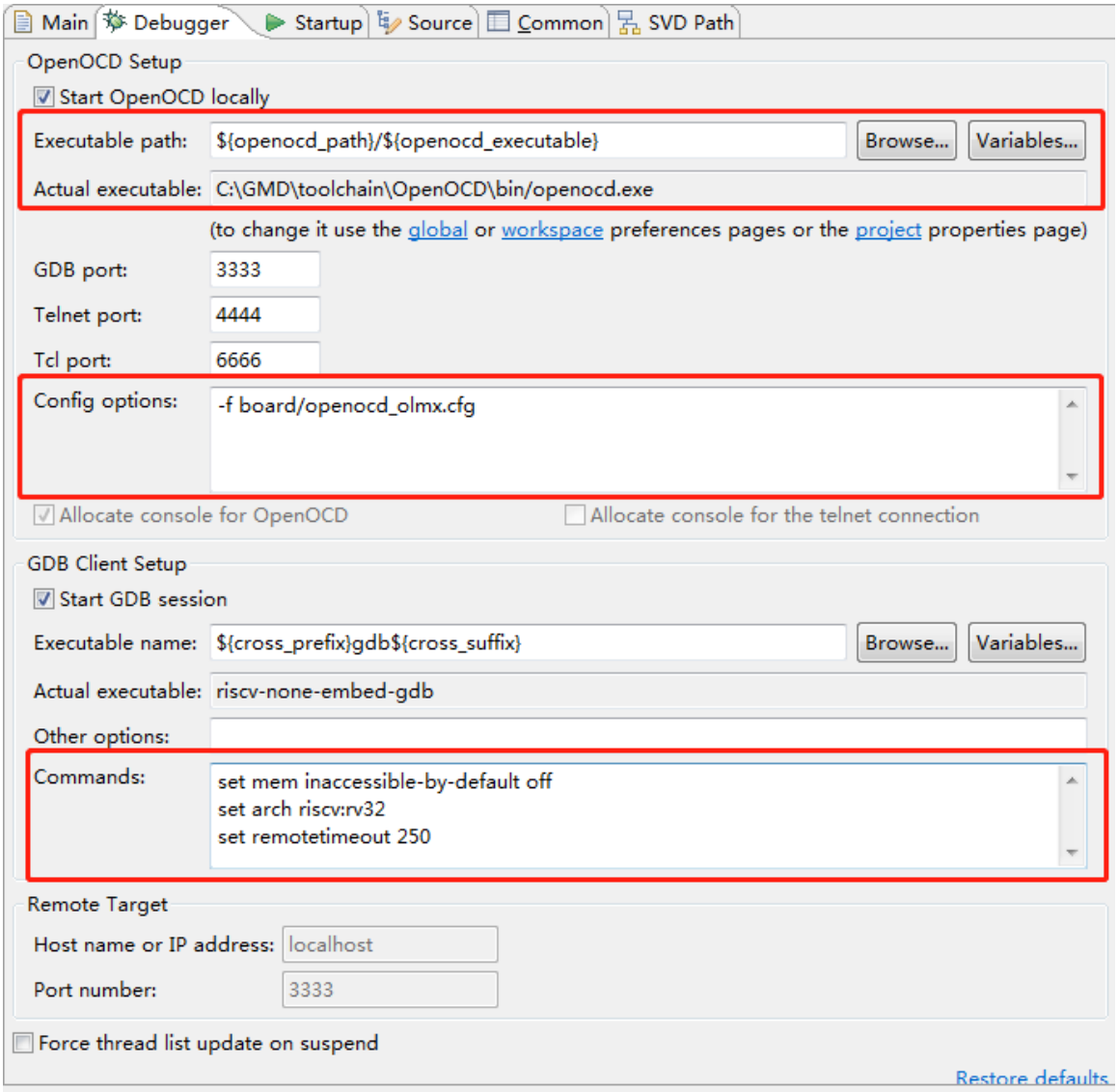
图 2-13 配置 Main 选项



配置 Debugger 选项

选择“Debugger”选项，配置 OpenOCD 和 GDB 等参数，如图 2-14 所示。

图 2-14 配置 Debugger 选项

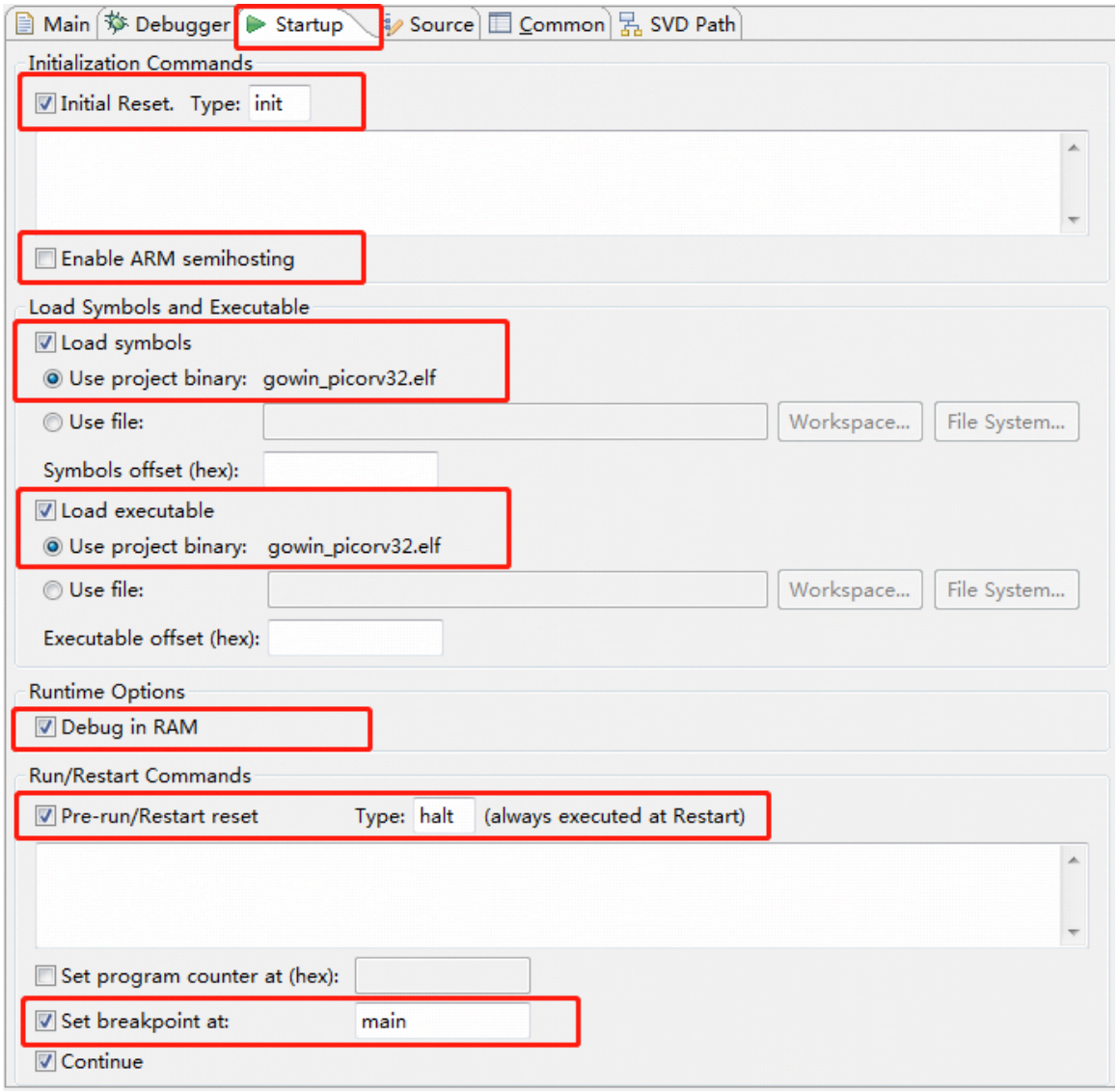


- OpenOCD Config options 选项配置
  该选项配置为“-f board/openocd_olmx.cfg”，指定所用调试仿真器配置文件，GMD 默认使用 Olimex 仿真器。
- GDB Commands 选项配置
  - set mem inaccessible-by-default off
  - set arch riscv:rv32（指定 RiscV 指令集架构）
  - set remotetimeout 250（防止协议超时）

配置 Startup 选项

选择“Startup”选项，配置执行在线调试等选项，如图 2-15 所示。

图 2-15 配置 Startup 选项



- Initialization Commands 选项配置
  - 使能 Initial Reset 选项，Type 类型配置为“init”，执行在线调试时，可以自动执行初始化
  - 禁用 Enable ARM semihosting 选项，Gowin_PicoRV32 不支持该功能
- Load Symbols and Executable 选项配置
  - 使能 Load symbols 选项，选择“Use project binary”
  - 使能 Load executable 选项，选择“Use project binary”选项
- Runtime Options 选项配置
  使能 Debug in RAM 选项，执行在线调试时，读取软件设计二进制文件，写入指令存储器 ITCM
- Run/Restart Commands 选项配置
  - 使能 Pre-run/Restart reset 选项，Type 类型配置为“halt”，执行在

线调试时，可以自动暂停

- 使能 set breakpoint at 选项，配置为“main”，执行在线调试时，可以自动在 main 主函数的第一条语句位置设置断点
- 使能 Continue 选项，执行在线调试时，可以自动运行到断点位置，并暂停

2.5.3 连接调试仿真器

以 Olimex 调试仿真器为例。

连接开发板与 Olimex 调试仿真器（或 RiscV 指令集架构处理器所支持的其他调试仿真器）。

Olimex 调试仿真器，选用 Olimex arm-usb-tiny-h，链接为：https://www.olimex.com/Products/ARM/JTAG/ARM-USB-TINY-H/

Olimex 调试仿真器驱动软件安装与配置，请参考 SUG549, GOWIN MCU Designer 用户指南。

将 Olimex 调试仿真器的 TDI、TDO、TMS、TCK 引脚按照标准 JTAG 接口顺序与开发板连接，VREF 引脚连接 3.3V，4/6/8/…/20 的 GND 引脚只需连接一个即可。

2.5.4 启动在线调试

单击工具栏调试“  ”，执行软件在线调试，调试页面包括：

- 代码区：可以查看 C 代码和汇编代码，设置断点，运行程序等；
- 变量区：可以查看通用寄存器的值或程序变量的值；
- 汇编指令区：可以查看当前指令存储器 ITCM 中的指令和当前正在执行的指令；
- 控制区：可以控制在线调试的进程，设置断点等。

调试中断处理函数

Gowin_PicoRV32 是面积优化的 RiscV 指令集架构处理器，中断控制系统设计简单易用，不支持硬件中断嵌套和中断优先级设置。

因此，当正在执行在线调试，并且程序处于暂停状态时（执行单步运行、断点运行等操作，使程序运行到某行或某条指令，且暂停在该位置的状态），MCU 无法响应外部中断请求。

如果需要对外部中断处理函数执行在线调试，则需要在中断处理函数中的相应位置设置断点，执行程序进入连续运行状态。

单击工具栏“Resume”，可以使程序进入连续运行状态，如果没有遇到断点，则会保持连续运行状态。在此状态下，当外部设备发出中断请求，MCU 执行中断处理函数，运行到中断处理函数中的断点位置进入暂停状态，此时可以在中断处理函数中执行单步调试、断点调试、变量查看等操作。

3 参考设计

Gowin_PicoRV32 支持 GOWIN MCU Designer（V1.1 及以上版本）软件环境的参考设计，通过链接获取如下参考设计：

Gowin_PicoRV32\ref_design\MCU_RefDesign\GMD_RefDesign\gowin_picorv32。


GOWIN高云
智慧逻辑 定制未来