

GigaDevice Semiconductor Inc.

GD32VW553 快速开发指南

应用笔记

AN154

1.3 版本

(2025 年 04 月)

目录

目录.....	2
图索引.....	4
表索引.....	6
1. 认识开发板.....	7
1.1. 开发板实物图.....	7
1.1.1. START 开发板.....	7
1.1.2. EVAL 开发板.....	8
1.2. 启动模式.....	9
1.3. 调试器接口.....	9
1.4. 下载接口.....	9
1.5. 查看日志.....	10
2. 搭建开发环境.....	11
2.1. GD32 Embedded Builder 安装.....	11
2.2. SEGGER Embedded Studio IDE 安装.....	11
2.3. ToolChain 下载.....	11
3. 开发需知.....	13
3.1. SDK 执行程序组.....	13
3.2. SDK 配置.....	13
3.2.1. 无线模块配置.....	13
3.2.2. SRAM 布局.....	14
3.2.3. FLASH 布局.....	14
3.2.4. 固件版本号.....	14
3.2.5. APP 配置.....	15
3.2.6. Configurations 选择.....	15
3.3. 正确日志示例.....	15
4. GD32 Embedded Builder IDE 工程.....	17
4.1. 打开工程组.....	17
4.2. 编译.....	20
4.3. 固件下载.....	23
4.4. 调试.....	23
5. SEGGER Embedded Studio IDE 工程.....	26

5.1.	打开工程组.....	26
5.2.	编译.....	27
5.3.	固件下载.....	29
5.4.	调试.....	30
6.	常见问题	32
6.1.	代码跑在 SRAM	32
7.	版本历史	33

图索引

图 1-1. START 开发板实物图.....	7
图 1-2. EVAL 开发板实物图.....	8
图 1-3. 开发板类型配置.....	9
图 1-4. 设备和驱动器列表.....	10
图 1-5. 串口配置.....	10
图 2-1 GD32 Embedded Builder 目录结构.....	11
图 2-2 Toolchain 下载.....	12
图 2-3 Nuclei Toolchain 目录结构.....	12
图 3-1. 启动过程.....	13
图 3-2. 无线模块配置.....	13
图 3-3. SRAM 布局.....	14
图 3-4. FLASH 布局.....	14
图 3-5. 固件版本号.....	15
图 3-6. 工程启动信息.....	16
图 4-1. SDK 目录.....	17
图 4-2. 启动 Embedded Builder.....	17
图 4-3. Open Projects from file System.....	18
图 4-4. 选择 MBL 工程路径.....	18
图 4-5. MBL 工程界面.....	19
图 4-6. 选择 MSDK 工程路径.....	19
图 4-7. MSDK 和 MBL 工程界面.....	20
图 4-8. 工具链配置.....	20
图 4-9. 编译 MBL 工程.....	21
图 4-10. 编译 MBL 结果.....	21
图 4-11. Configurations 选择.....	22
图 4-12. MSDK 编译结果.....	22
图 4-13. images 输出.....	23
图 4-14. 打开调试配置选项.....	23
图 4-15. MSDK 调试配置.....	24
图 4-16 MSDK 使用 openocd 调试配置界面.....	24
图 4-17. MSDK 调试界面.....	25
图 5-1. MBL SES Project 工程界面.....	26
图 5-2. MSDK SES 工程界面.....	27
图 5-3. nuclei 工具链内容.....	27
图 5-4. 编译 MBL 工程.....	28
图 5-5. MBL 编译结果.....	28
图 5-6. 编译 MSDK 工程.....	28
图 5-7. MSDK 工程配置选择.....	29
图 5-8. MSDK 编译结果.....	29

图 5-9. images 输出	29
图 5-10 SES IDE image 下载.....	30
图 5-11 MSDK SES 工程配置界面.....	31
图 5-12 SES IDE Debug 界面	31

表索引

表 1-1. 启动模式.....	9
表 7-1. 版本历史.....	33

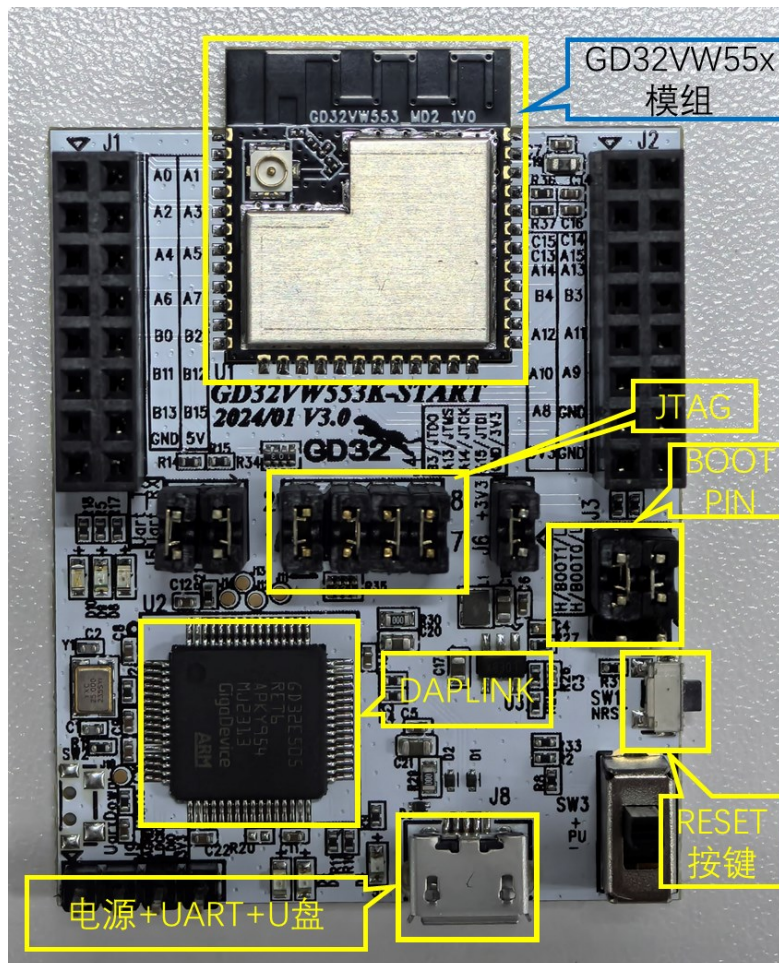
1. 认识开发板

1.1. 开发板实物图

1.1.1. START 开发板

START 开发板由底板和模组组成，模组搭载了 GD32VW55x WiFi+BLE 芯片。

图 1-1. START 开发板实物图



主要关注开发板的以下几个部分，已在[图 1-1. START 开发板实物图](#)中标注出来。

- 启动模式（Boot PIN）；
- 供电口（电源）；
- 查看日志（UART）；
- 调试器接口（DAPLINK、JLINK 或 GDLINK）；
- 重启（Reset Button）。

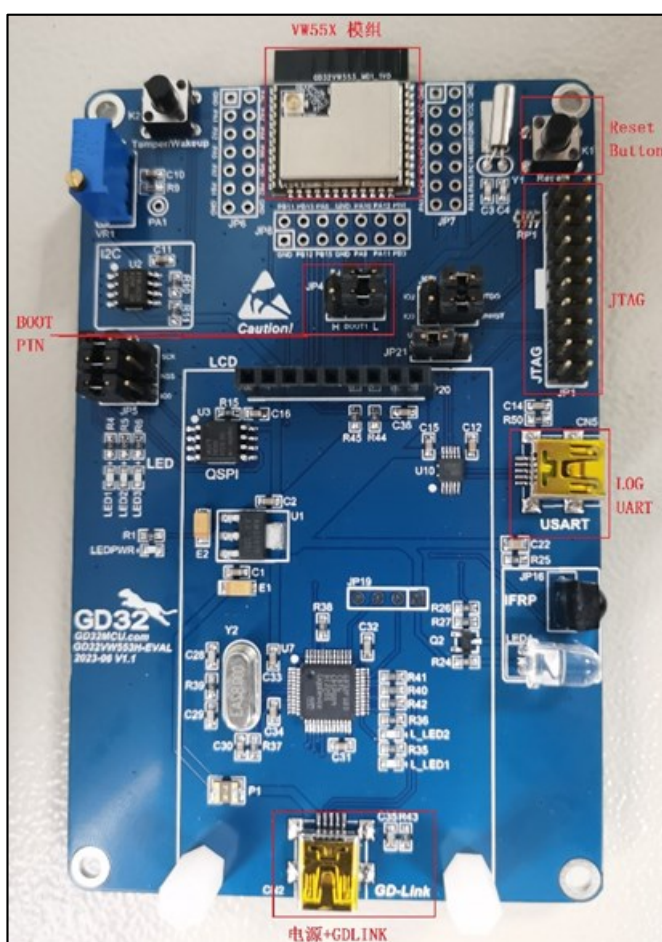
1.1.2. EVAL开发板

EVAL 开发板由底板和模组组成，模组搭载了 GD32VW55x WiFi+BLE 芯片，底板引出了众多外设测试口，例如：I2C，IFRP，ADC 等。

对于开发者来说，主要关注开发板的以下几个部分，已在[图 1-2. EVAL 开发板实物图](#)中标注出来。

- 启动模式（Boot PIN）；
- 供电口（电源）；
- 查看日志（UART）；
- 调试器接口（DAPLINK、JLINK 或 GDLINK）；
- 重启（Reset Button）。

图 1-2. EVAL 开发板实物图



对于 START 和 EVAL 开发板，SDK 配置不同，需要选择不同的宏使能。如下所示，SDK 默认选择 START 开发板配置。配置文件为 GD32VW55x_RELEASE/config/platform_def.h。

图 1-3. 开发板类型配置

```
// board type
#define PLATFORM_BOARD_32VW55X_START 0
#define PLATFORM_BOARD_32VW55X_EVAL 1
#define PLATFORM_BOARD_32VW55X_F527 2
#ifdef CONFIG_PLATFORM_ASIC
#define CONFIG_BOARD PLATFORM_BOARD_32VW55X_START
#endif
```

1.2. 启动模式

GD32VW55x 可以选择从 ROM 启动，FLASH 启动或者 SRAM 启动。

开发板 BOOTSWD 框内的 BOOT0 和 BOOT1 两根引脚的高低选择决定了启动模式，见[表 1-1. 启动模式](#)。更多关于启动模式的说明请参考文档《GD32VW55x_User_Manual》。

表 1-1. 启动模式

EFBOOTLK	BOOT0	BOOT1	EFSB	启动地址	启动区域
0	0	-	0	0x08000000	SIP Flash
0	0	-	1	0x0BF46000	secure boot
0	1	0	-	0x0BF40000	Bootloader / ROM
0	1	1	-	0x20000000	SRAM
1	0	-	0	0x08000000	SIP Flash
1	0	-	1	0x0BF46000	Secure boot
1	1	-	-	0x0BF40000	Bootloader / ROM

1.3. 调试器接口

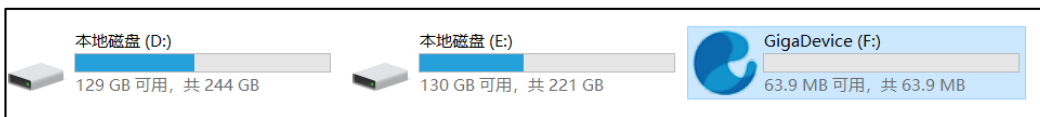
对于 START 开发板，开发板自带 DAPLINK（GD32F505），可以搭配 OpenOCD 使用，也可外接 GD-Link 调试器或者 J-Link 调试器进行调试和下载。DAP 芯片还集成了 UART 功能，所以只需要一根 USB 线，就可以完成供电、调试和查看日志。将引脚 JTCK、JTMS、JTDO 和 JTDI 与下侧四引脚通过跳线帽连接，即可通过 DAPLINK 下载和调试代码。[图 1-1. START 开发板实物图](#)中展示的是透过 DAPLINK 进行调试。

对于 EVAL 开发板，可使用 GD-Link 调试器或者 J-Link 调试器进行调试和下载，不支持 DAPLINK。

1.4. 下载接口

对于 START 开发板，除了上一节提到的通过 GD-Link 调试器或者 J-Link 调试器进行固件下载外，如果不需要调试功能，仅需要下载固件，还可以使用 U 盘拷贝的方式下载。将开发板通过 USB 线插入电脑，可以看到[图 1-4. 设备和驱动器列表](#)所示 GigaDevice 盘。将 image-all.bin 文件（见后续章节）拷贝进 GigaDevice 盘，就能完成对 GD32VW55x 芯片的 FLASH 烧写。

图 1-4. 设备和驱动器列表

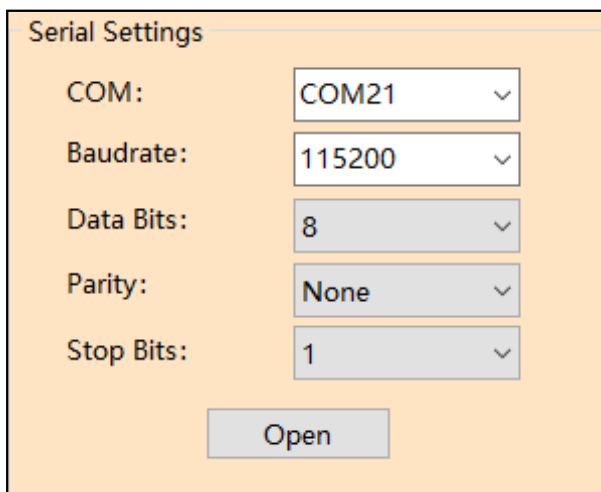


对于 EVAL 开发板，可使用 GDLINK 调试器或者 JLINK 调试器进行下载，不支持 U 盘拷贝。

1.5. 查看日志

使用 MicroUSB 线连接 START 开发板，PC 端使用串口工具并根据[图 1-5. 串口配置](#)的参数配置并连接，就可以使用串口输出日志了。

图 1-5. 串口配置



2. 搭建开发环境

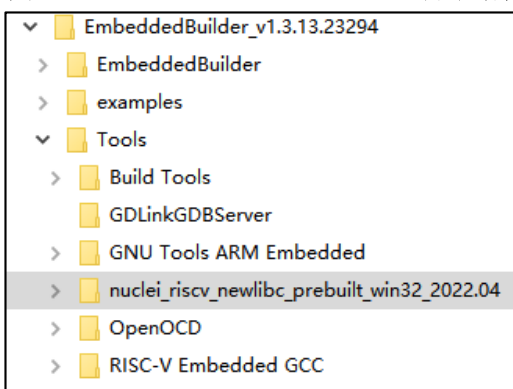
在编译和烧录固件之前，需要搭建开发环境。

目前可用的开发工具是 GD32 Embedded Builder 和 SEGGER Embedded Studio IDE。

2.1. GD32 Embedded Builder 安装

GD32 Embedded Builder 可在地址 <https://gd32mcu.com/cn/download> 选择 GD32VW5 下载，解压后的路径如 [图 2-1 GD32 Embedded Builder 目录结构](#) 所示，同时建议将在小节 [2.3 ToolChain](#) 中下载的 nuclei_riscv_newlibc_prebuilt_win32_2022.04 也解压放在 Tools 目录下。

图 2-1 GD32 Embedded Builder 目录结构



2.2. SEGGER Embedded Studio IDE 安装

请访问 <https://wiki.segger.com/GD32V> 获取 SEGGER Embedded Studio IDE 和 License Activation Key。

2.3. ToolChain 下载

■ 下载

官网地址：<https://nucleisys.com/download.php>

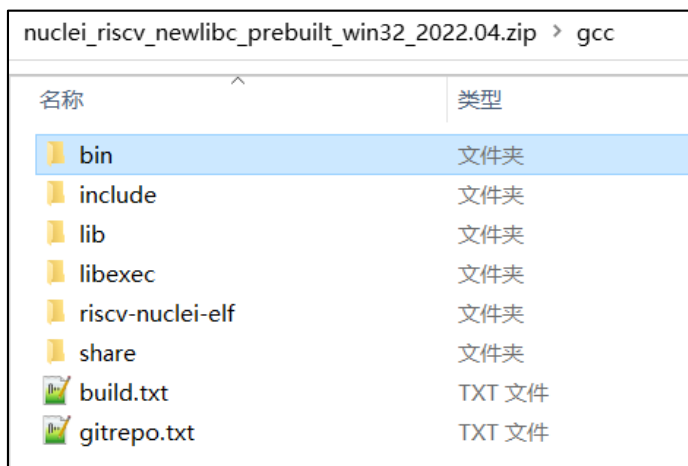
如 [图 2-2 Toolchain 下载](#) 所示，ToolChain 选择 2022.04 版本。

下载的 Nuclei RISC-V Toolchain 解压后目录结构如 [图 2-3 Nuclei Toolchain 目录结构](#) 所示。

图 2-2 Toolchain 下载



图 2-3 Nuclei Toolchain 目录结构



3. 开发需知

在着手开发之前，先了解下 SDK 执行程序组成员有哪些，以及如何正确配置 SDK。

3.1. SDK 执行程序组

SDK 最终生成的执行程序主要有两个：一个是 MBL (Main Bootloader)，一个是 MSDK (Main SDK)。它们最终都将被烧写到 FLASH 运行。上电之后，程序将从 MBL 的 Reset_Handler 启动，然后跳转到 MSDK 主程序运行，如[图 3-1. 启动过程](#)所示。

图 3-1. 启动过程



3.2. SDK 配置

3.2.1. 无线模块配置

配置文件为 GD32VW55x_RELEASE/config/platform_def.h，主要内容见[图 3-2. 无线模块配置](#)。

图 3-2. 无线模块配置

```
#define CFG_WLAN_SUPPORT
#define CFG_BLE_SUPPORT
#if defined(CFG_WLAN_SUPPORT) && defined(CFG_BLE_SUPPORT)
    #define CFG_COEX
#endif
```

- 如果是 BLE/WiFi combo 模式，请打开：
 - #define CFG_WLAN_SUPPORT
 - #define CFG_BLE_SUPPORT
- 如果是 BLE only，请只打开：
 - #define CFG_BLE_SUPPORT
- 如果是 WiFi only，请只打开：
 - #define CFG_WLAN_SUPPORT
- 如果关闭无线模块，请全部关闭。

3.2.2. SRAM布局

SRAM layout 配置文件为 GD32VW55x_RELEASE\config\config_gdm32.h。修改 [图 3-3. SRAM 布局](#) 中宏定义值，可以对可执行程序段 MBL 及 IMG 占用的 SRAM 空间进行规划。这些值是偏移地址，基地址定义在该文件开头处。

标注 “!Keep unchanged!” 的行不能修改，否则会影响 ROM 中代码 MbedTLS 的运行。

图 3-3. SRAM 布局

```
/* SRAM LAYOUT */
#define RE_MBL_DATA_START ..... 0x300 ..... /* !Keep unchanged! */
#define RE_IMG_DATA_START ..... 0x200 ..... /* !Keep unchanged! */
```

每个可执行程序段内部的 SRAM 空间规划可以参考对应工程下的 .ld 文件，如 MBL\project\eclipse\mbl.ld 和 MSDK\plfrisc\env\gd32vw55x.ld。

3.2.3. FLASH布局

Flash layout 配置文件为 GD32VW55x_RELEASE\config\config_gdm32.h。修改 [图 3-4. FLASH 布局](#) 中宏定义值，可以对可执行程序段 MBL 及 MSDK 占用的 FLASH 空间进行规划。这些值是偏移地址，基地址定义在该文件开头处。

标注 “!Keep unchanged!” 的行不能修改，否则会影响工程运行。

图 3-4. FLASH 布局

```
/* FLASH LAYEROUT */
#define RE_VTOR_ALIGNMENT ..... 0x200 ..... /* !Keep unchanged! */
#define RE_SYS_SET_OFFSET ..... 0x0 ..... /* !Keep unchanged! */
#define RE_MBL_OFFSET ..... 0x0 ..... /* 0x0: Boot from MBL, 0x1000: Boot from ROM */
#define RE_SYS_STATUS_OFFSET ..... 0x8000 ..... /* !Keep unchanged! */
#define RE_IMG_0_OFFSET ..... 0xA000 ..... /* !Keep unchanged! */
#define RE_IMG_1_OFFSET ..... 0x1E000
#define RE_IMG_1_END ..... 0x3CB000 ..... /* reserved 192KB for user data */
#define RE_NVDS_DATA_OFFSET ..... 0x3FB000 ..... /* reserved 20KB for nvds data */
#define RE_END_OFFSET ..... 0x400000 ..... /* equal to flash total size */
```

每个可执行程序段内部的 FLASH 空间规划可以参考对应工程下的 .ld 文件，如 MBL\project\eclipse\mbl.ld 和 MSDK\plfrisc\env\gd32vw55x.ld。

3.2.4. 固件版本号

配置文件为 GD32VW55x_RELEASE\config\config_gdm32.h。修改 [图 3-5. 固件版本号](#) 中宏定义值，可以指定版本号。此外 RE_IMG_VERSION 在 Secure Boot 中会用于固件版本的判定。

MBL 只能本地升级，IMG 可以支持在线升级，SDK 发布的版本号与 RE_IMG_VERSION 保持一致。

图 3-5. 固件版本号

```
/* FW_VERSION */  
#define RE_MBL_VERSION ..... 0x01000002  
#define RE_IMG_VERSION ..... 0x01000002
```

3.2.5. APP配置

配置文件为 GD32VW55x_RELEASE\MSDK\app\app_cfg.h。可以选择是否打开一些应用，例如：ATCMD，阿里云，MQTT，COAP 等。

3.2.6. Configurations选择

- msdk
FreeRTOS + Libwpas.a + libble.a + MbedTLS 3.6.2
- msdk_ffd
FreeRTOS + wpa_supplicant + libble_max.a + MbedTLS 3.6.2
- msdk_mbedtls_2.17.0
FreeRTOS + Libwpas.a + libble.a + ROM MbedTLS 2.17.0
- msdk_rtthread
RTThread + Libwpas.a + libble.a + MbedTLS 3.6.2
- msdk_threadx
Threadx + Libwpas.a + libble.a + MbedTLS 3.6.2

如上，MSDK 支持多种 Configurations，分别是：msdk（默认）、msdk_ffd、msdk_threadx、msdk_mbedtls_2.17.0、msdk_rtthread。

Libwpas.a 与 wpa_supplicant 均是 Wi-Fi 相关的 lib，后者可用于过 WPA 认证。

libble 与 libble_max 均是 BLE lib，libble_max 可以支持更多的功能。msdk 默认链接 libble，但也可以修改配置链接 libble_max。

与 MbedTLS 3.6.2 相比，ROMMbedTLS2.17.0 代码固化在 ROM 中，可以节省 Flash 资源。

在实际使用时，如何进行 Configurations 选择详见 [4.2 编译](#) 小节中编译 MSDK 工程部分。

3.3. 正确日志示例

在固件组（MBL+MSDK）下载成功后，打开串口工具，按下开发板上的 Reset 键，可以看到 [图 3-6. 工程启动信息](#)。如果出现异常，请查阅 [6 常见问题](#)，看能否找到帮助。

图 3-6. 工程启动信息

```
#ALW: MBL: First print.
ALW: MBL: Boot from Image 0.
ALW: MBL: Validate Image 0 OK.
ALW: MBL: Jump to Main Image (0x0800a000).
Chip: GD32VW55x
=== RF initialization finished ===
SDK Version: v1.0.3-f0e77956a91696b2
Build date: 2025/04/07 14:43:40
=== WiFi calibration done ===
=== PHY initialization finished ===
BLE local addr: 76:BA:ED:23:00:D5, type 0x0
=== BLE Adapter enable complete ===
```


4. GD32 Embedded Builder IDE 工程

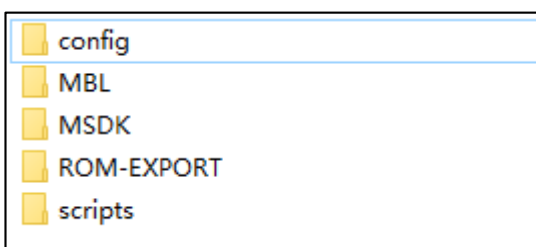
本章将介绍如何在 GD32 Embedded Builder IDE 下编译和调试 SDK。

工程组由 MBL 和 MSDK 两个工程组成。MSDK 包含 WiFi 和 BLE 协议栈、外设驱动及应用程序等，MBL 主要负责从两个 MSDK 固件（当前固件和 OTA 固件）中选择正确的去运行。

4.1. 打开工程组

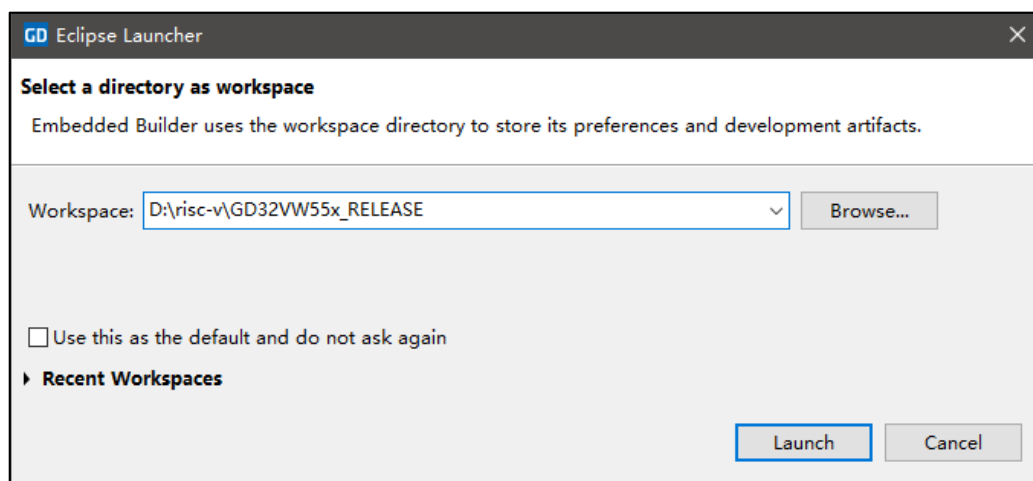
- 检查 SDK 目录 GD32VW55x_RELEASE，如[图 4-1. SDK 目录](#)所示。

图 4-1. SDK 目录



启动 IDE，双击 EmbeddedBuilder 目录下的 Embedded Builder.exe，并选择 SDK 目录 GD32VW55x_RELEASE 为 workspace，点击 launch 按钮，如[图 4-2. 启动 Embedded Builder](#)所示。

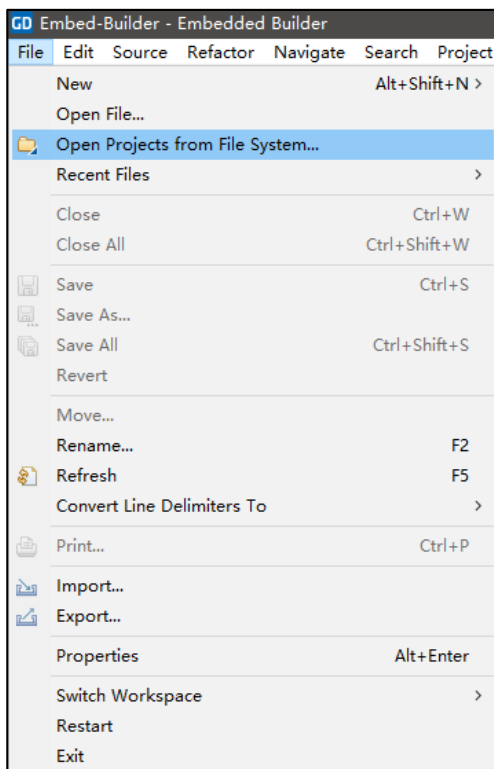
图 4-2. 启动 Embedded Builder



- 导入 MBL 工程

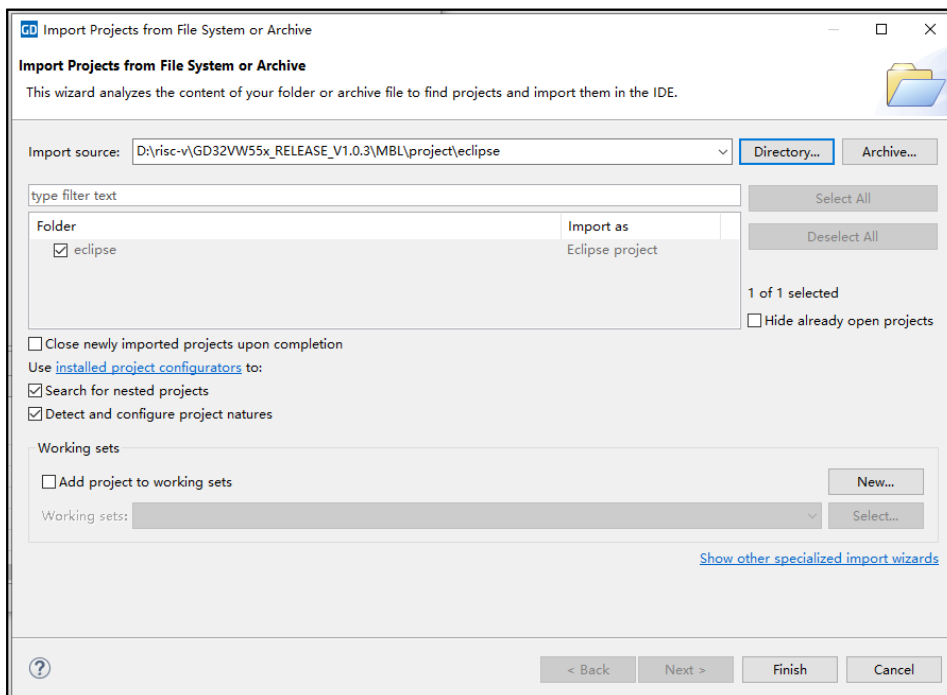
File 菜单点击 Open Projects from file System，如[图 4-3. Open Projects from file System](#)所示。

图 4-3. Open Projects from file System



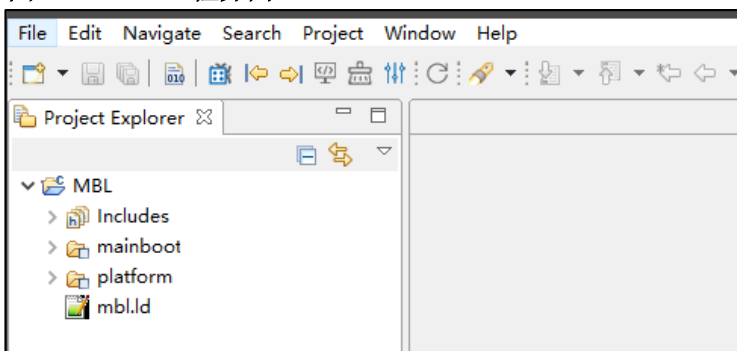
选择工程路径 GD32VW55x_RELEASE\MBL\project\eclipse，如 [图 4-4. 选择 MBL 工程路径](#) 所示，并点击 finish。

图 4-4. 选择 MBL 工程路径



关闭 welcome 界面就可以看到 MBL 工程，如 [图 4-5. MBL 工程界面](#) 所示。

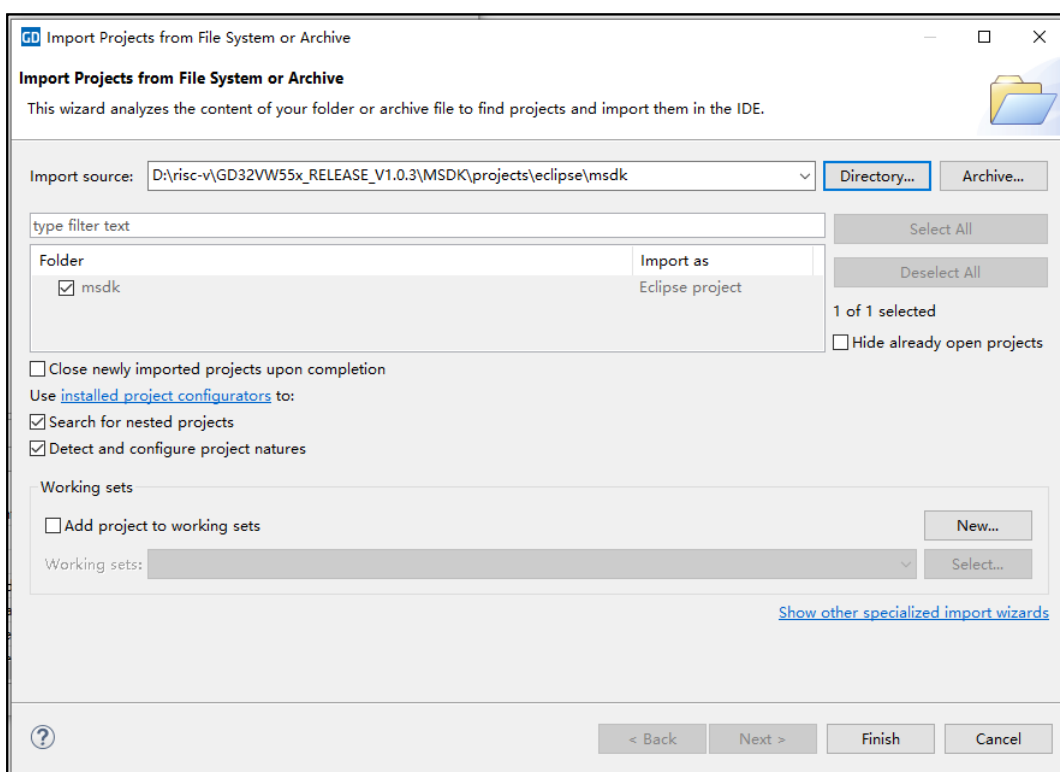
图 4-5. MBL 工程界面



■ 导入 MSDK 工程

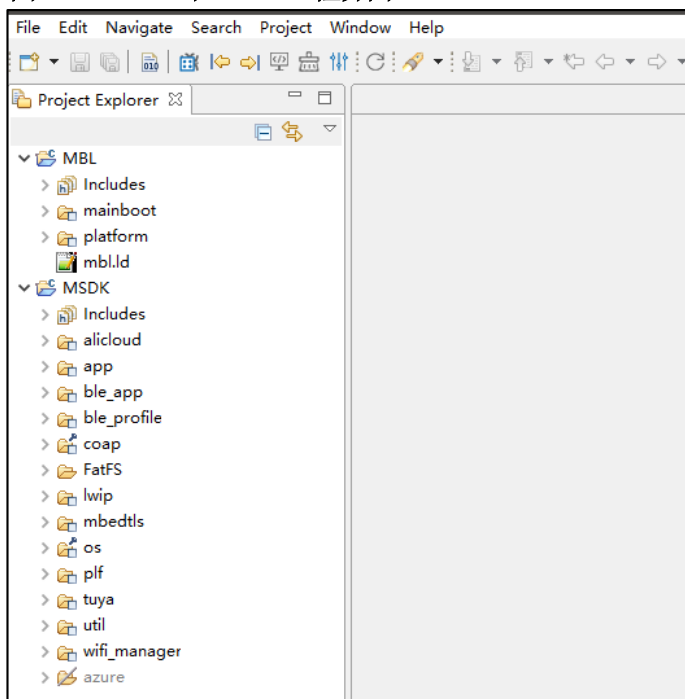
File 菜单点击 Open Projects from file System, 工程路径选择 GD32VW55x_RELEASE\MSDK\projects\eclipse\msdk, 如[图 4-6. 选择 MSDK 工程路径](#)所示, 并点击 finish。

图 4-6. 选择 MSDK 工程路径



查看 MSDK 和 MBL 工程界面, 如[图 4-7. MSDK 和 MBL 工程界面](#)所示。

图 4-7. MSDK 和 MBL 工程界面

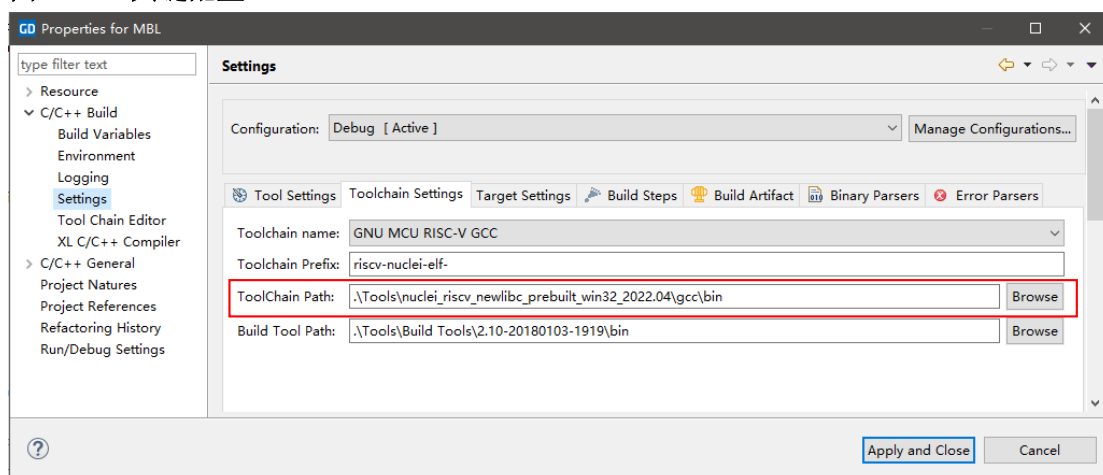


4.2. 编译

■ 检查工程编译工具配置

右击工程，点击 **properties**，依次选择 **C/C++ Build** → **Settings**，选项卡点击 **toolchain settings**，如 [图 4-8. 工具链配置](#) 所示，将 [2.3 ToolChain](#) 下载的 ToolChain 路径填入。点击 **apply and close**。

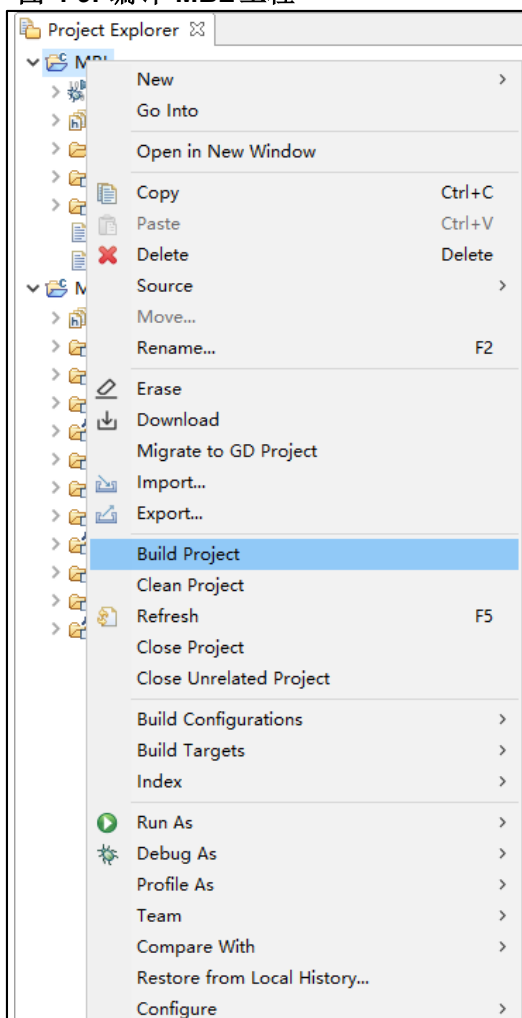
图 4-8. 工具链配置



■ 编译 MBL 工程

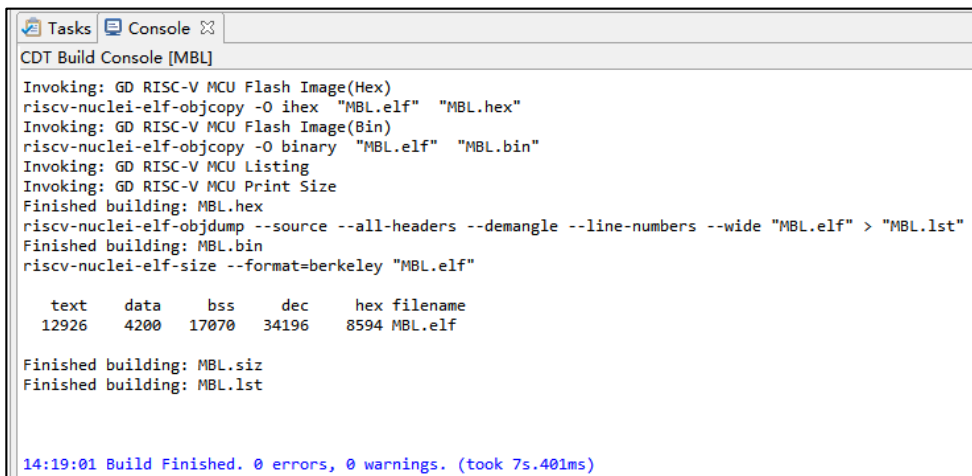
右击工程，点击 **build project**，如 [图 4-9. 编译 MBL 工程](#) 所示。

图 4-9. 编译 MBL 工程



编译结果，如 [图 4-10. 编译 MBL 结果](#) 所示。

图 4-10. 编译 MBL 结果

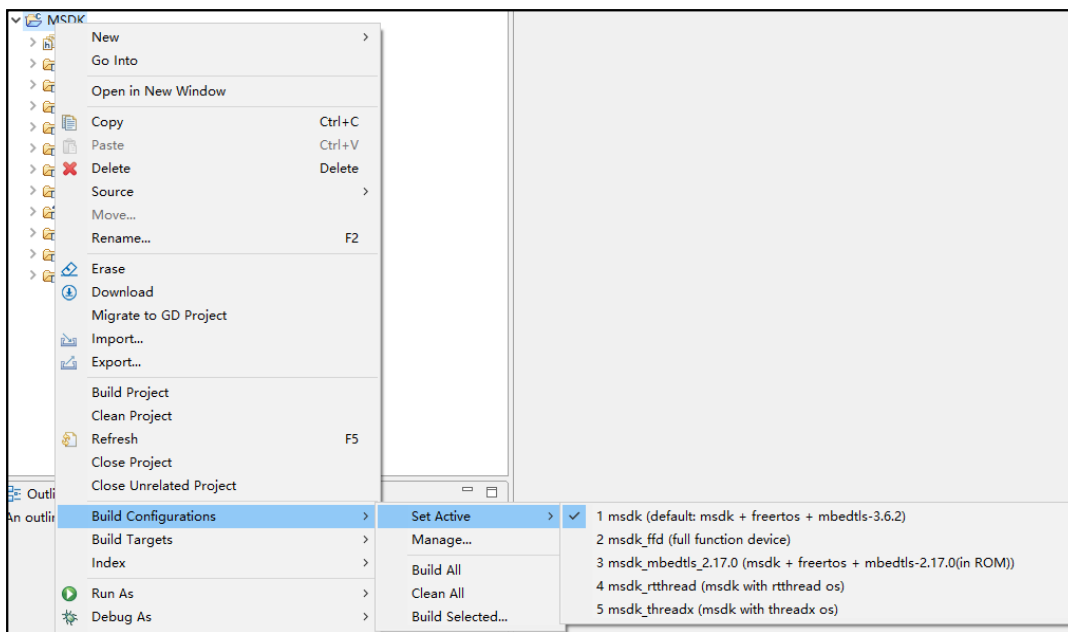


编译完成后会自动调用脚本 `MBL\project\mbl_afterbuild.bat` 生成 `mbl.bin`，并拷贝至目录 `scripts\images`。

■ 编译 MSDK 工程

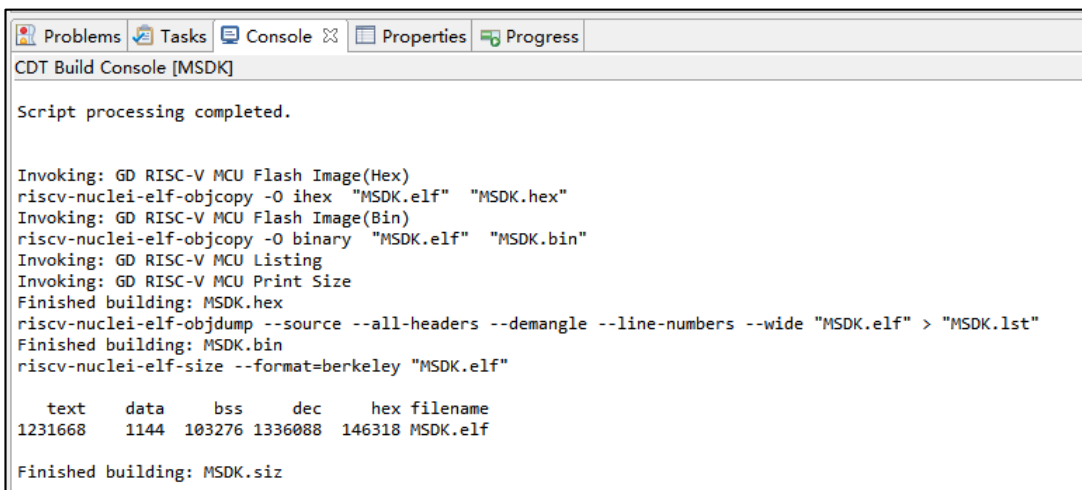
右击工程，依次点击 **Build Configurations**—>**Set Active**—> 选择需要的配置，如 [图 4-11. Configurations 选择](#)所示，MSDK 默认 configuration 为 msdk。

图 4-11. Configurations 选择



再次右击工程，点击 **Build project**，编译结果如 [图 4-12. MSDK 编译结果](#)所示。

图 4-12. MSDK 编译结果



■ SDK 生成的 image

MSDK 编译完成之后，会调用 `MSDK\projects\image_afterbuild.bat` 生成 `image-ota.bin` 和 `image-all.bin`。并将生成的 bin 文件拷贝至 `\scripts\images` 内，如 [图 4-13. images 输出](#)所示。

`image-ota.bin` 是 MSDK 工程生成的 bin 文件，可用于 OTA 升级，`image-all.bin` 是 MBL(`mbi.bin`) 和 MSDK(`image-ota.bin`)的组合，该固件可用于生产，烧录到 FLASH 中运行。

图 4-13. images 输出

名称	修改日期	类型	大小
 image-all.bin	2024/7/11 14:26	BIN 文件	788 KB
 image-ota.bin	2024/7/11 14:26	BIN 文件	748 KB
 mbl.bin	2024/7/11 14:19	BIN 文件	17 KB

4.3. 固件下载

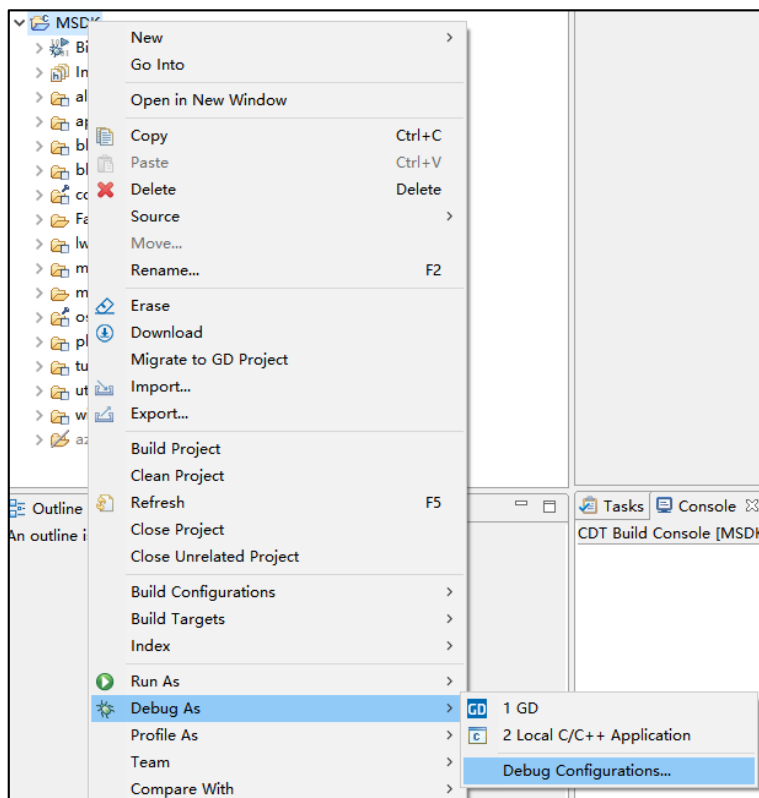
如 [1.4 下载接口](#) 章节，拷贝 GD32VW55x_RELEASE\scripts\images\image-all.bin 至 GigaDevice 盘即可烧写。或者通过 DAPLINK 和 J-LINK 也可下载固件。

4.4. 调试

■ 配置调试配置

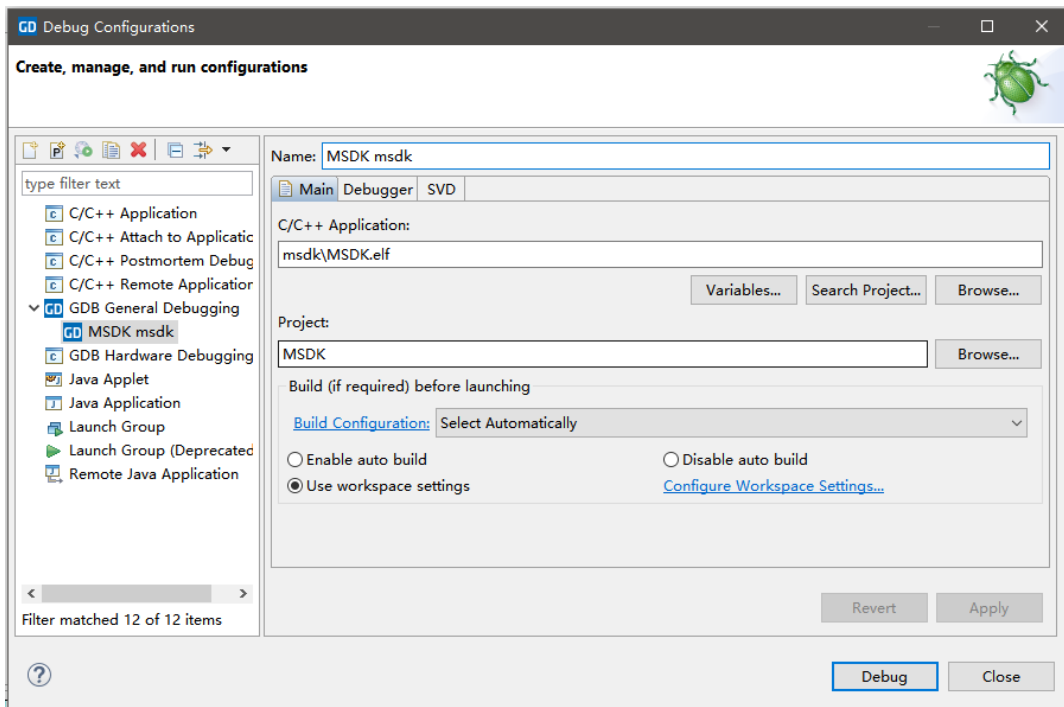
右击 MSDK 工程，依次点击 Debug As->Debug Configurations，如 [图 4-14. 打开调试配置选项](#) 所示。

图 4-14. 打开调试配置选项



双击 GDB General Debugging，打开 [图 4-15. MSDK 调试配置](#) 所示界面，c/c++ application 已经自动选择 msdk\MSDK.elf，可通过 browse 选择不同 configurations 对应的 elf 文件，例如 msdk_ffd 产生的 elf 文件位于 GD32VW55x_RELEASE\MSDK\projects\eclipse\msdk\msdk_ffd\MSDK.elf

图 4-15. MSDK 调试配置



■ 开始调试

start 板集成了 GD-Link 功能，建议使用 openocd 进行调试，openocd 在路径：EmbeddedBuilder_v1.3.13.23294\Tools\OpenOCD 下，内部集成了 GD32VW55x 的 GD-Link 脚本。按图 4-16 MSDK 使用 openocd 调试配置界面所示将 GDB Server 切换到 openocd，同时将 config Options 按图中所示进行指定，点击 Debug，等待 image 烧录完成即可 debug。调试界面如图 4-17. MSDK 调试界面所示。

图 4-16 MSDK 使用 openocd 调试配置界面

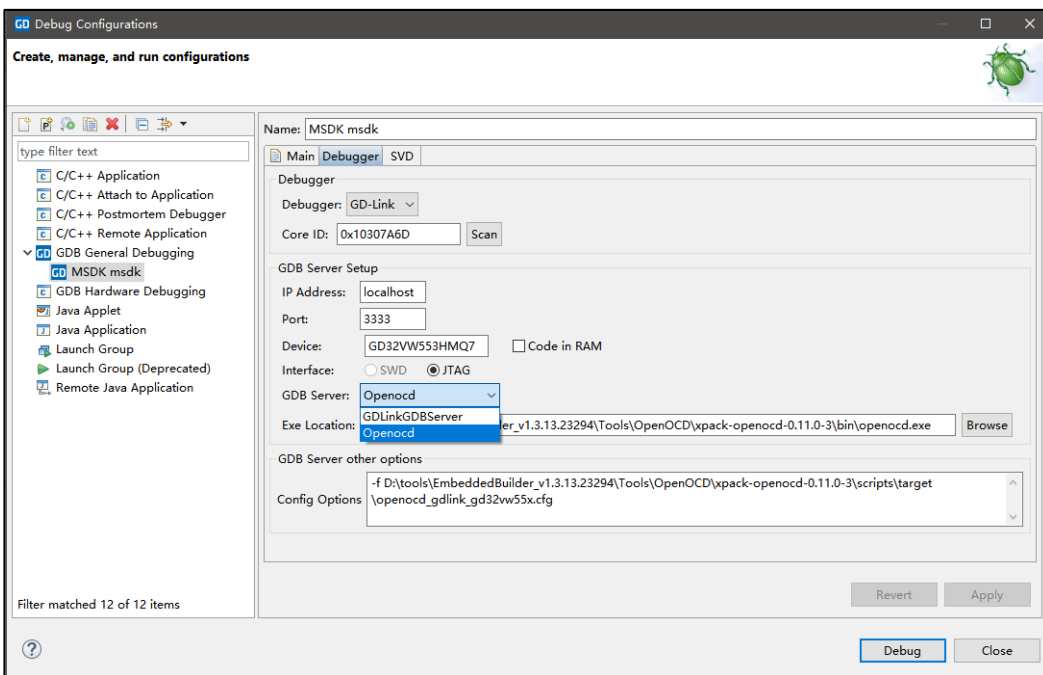
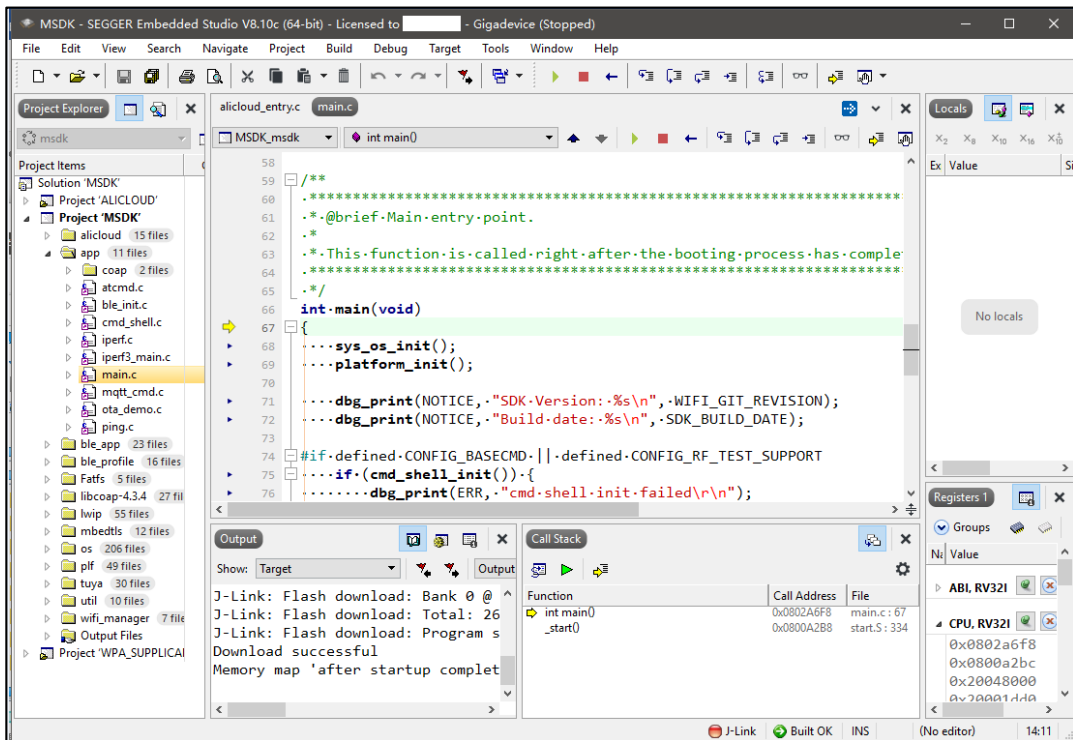


图 4-17. MSDK 调试界面



5. SEGGER Embedded Studio IDE 工程

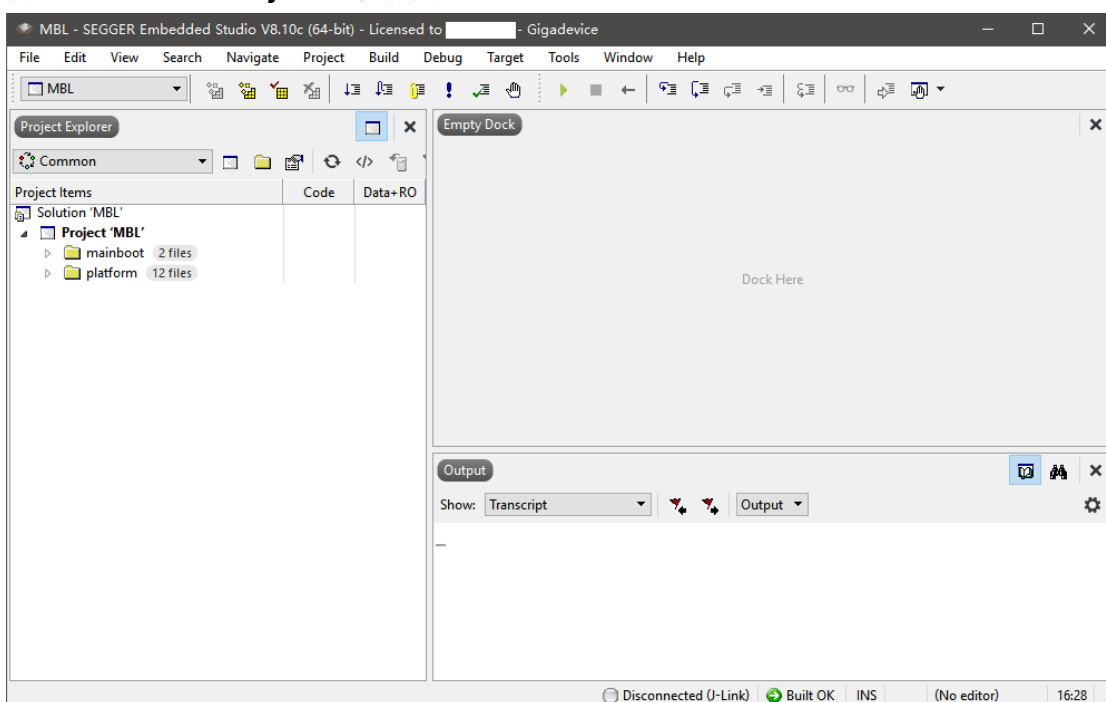
本章将介绍如何在 SEGGER Embedded Studio IDE 下编译和调试 SDK。

5.1. 打开工程组

■ 打开 MBL 工程

打开目录：GD32VW55x_RELEASE\MBL\project\segger，双击 MBL.emProject 打开 MBL SES 工程，打开后的工程如[图 5-1. MBL SES Project 工程界面](#)所示。

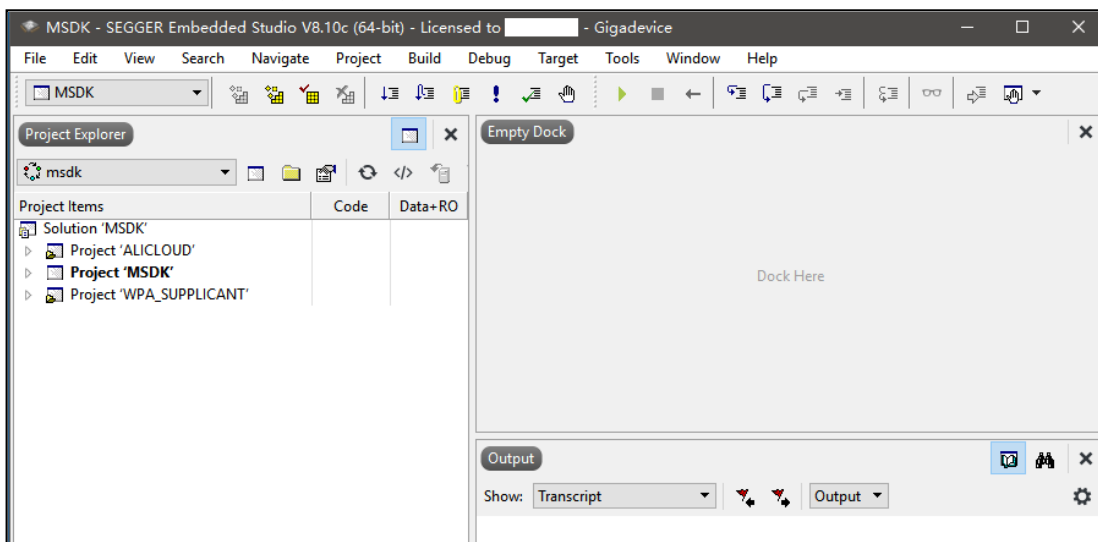
图 5-1. MBL SES Project 工程界面



■ 打开 MSDK 工程

打开目录：GD32VW55x_RELEASE\MSDK\projects\segger，双击 MSDK.emProject 打开 MSDK 工程，打开后的工程如[图 5-2. MSDK SES 工程界面](#)所示。

图 5-2. MSDK SES 工程界面



5.2. 编译

■ Nuclei Toolchain 配置

请在小节 [2.3ToolChain](#) 中下载的 Toolchain 的目录：`nuclei_riscv_newlibc_prebuilt_win32_2022.04\gcc\bin` 放至 windows 的环境变量 `path` 中。

■ SES build tool 配置

SES 默认使用 `riscv32-none-elf` 的工具链编译 GD32VW55x 的工程，为了更好地支持 riscv 的拓展指令集，需要使用 `nuclei` 的工具链即 `riscv-nuclei-elf` 编译。编译工具可联系销售或 FAE 获取，工具链具体内容如 [图 5-3. nuclei 工具链内容](#) 所示。图中 Segger_IDE 为 SES IDE 安装目录。

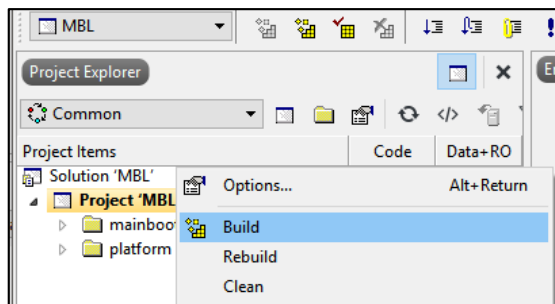
图 5-3. nuclei 工具链内容

Segger_IDE > gcc > riscv-nuclei-elf		
名称	修改日期	类型
bin	2024/3/18 17:59	文件夹
include	2024/3/19 9:40	文件夹
lib	2024/7/12 15:21	文件夹

■ 编译 MBL 工程

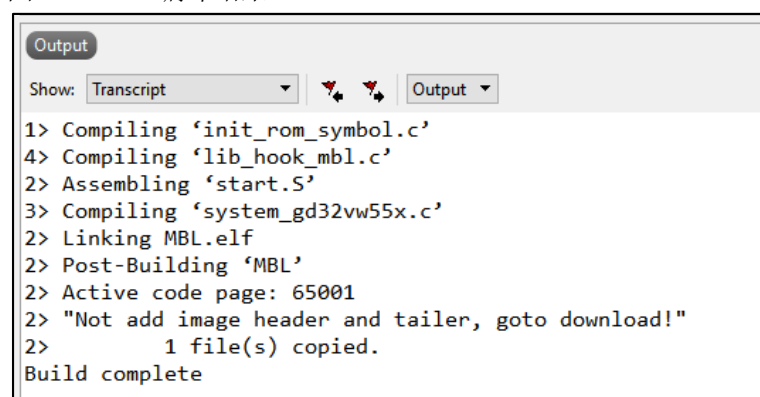
右击工程，点击 `build`，如 [图 5-4. 编译 MBL 工程](#) 所示；或者点击菜单栏 `Build->Build MBL`。

图 5-4. 编译 MBL 工程



编译结果，如 [图 5-5. MBL 编译结果](#) 所示。

图 5-5. MBL 编译结果

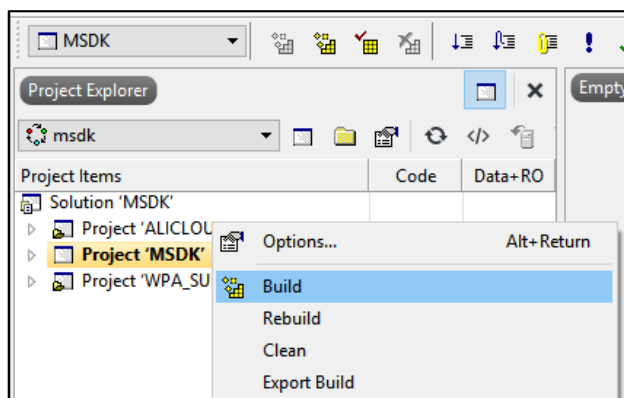


编译完成后会自动调用脚本 MBL\project\mbl_afterbuild.bat 生成 mbl.bin，并拷贝至目录 scripts\images。

■ 编译 MSDK 工程

右击 MSDK 工程内 Project 'MSDK'，点击 Build，如 [图 5-6. 编译 MSDK 工程](#) 所示。

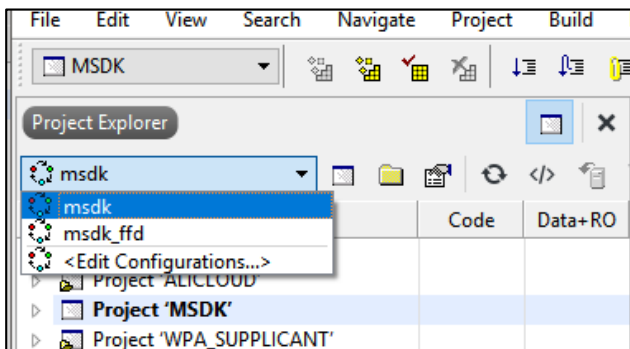
图 5-6. 编译 MSDK 工程



■ MSDK configuration 选择

MSDK 的 configuration 切换如 [图 5-7. MSDK 工程配置选择](#) 所示，当前 SES 的工程仅支持了 msdk 和 msdk_ffd；若需使用配置 msdk_threadx、msdk_ffd_threadx 和 msdk_azure 还请使用 GD32 EmbeddedBuilder IDE 工程或者等待后续更新。

图 5-7. MSDK 工程配置选择



选择对应的配置后，右击工程，点击 Build，编译结果如 [图 5-8. MSDK 编译结果](#) 所示。

图 5-8. MSDK 编译结果

```

Output
Show: Transcript Output
2> Linking MSDK.elf
2> Post-Building 'MSDK'
2> Active code page: 65001
2> ECHO is off.
2> "msdk.elf"
2> Invalid switch - "risc-v".
2> mbl_offset=0x0 image0_offset=0xA000 image1_offset=0x1E0000
2> rftest_on=0
2> mbl_len = 18060
2> File Not Found
2> Missing operand.
2> rftest_end =
2> "Not add image header and tailer!"
2>          1 file(s) copied.
2> Goto download!
Build complete

```

■ SDK 生成的 image

MSDK 编译完成之后，会调用 MSDK\projects\image_afterbuild.bat 生成 image-ota.bin 和 image-all.bin。并将生成的 bin 文件拷贝至\scripts\images 内，如 [图 5-9. images 输出](#) 所示。

image-ota.bin 是 MSDK 工程生成的 bin 文件，可用于 OTA 升级，image-all.bin 是 MBL(mbl.bin) 和 MSDK(image-ota.bin) 的组合，该固件可用于生产，烧录到 flash 中运行。

图 5-9. images 输出

名称	修改日期	类型	大小
image-all.bin	2024/7/11 18:12	BIN 文件	782 KB
image-ota.bin	2024/7/11 18:12	BIN 文件	742 KB
mbl.bin	2024/7/11 17:16	BIN 文件	18 KB

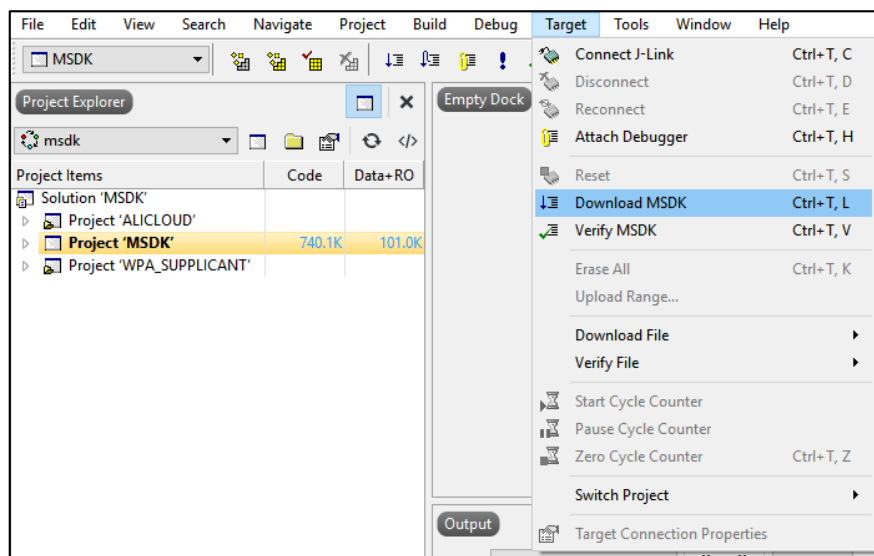
5.3. 固件下载

如 [1.4 下载接口](#) 章节，拷贝 GD32VW55x_RELEASE\scripts\images\image-all.bin 至

Gigadevice 盘即可烧写。

或者通过点击菜单栏 Target->Download MSDK 下载，如[图 5-10 SES IDE image 下载](#)所示。

图 5-10 SES IDE image 下载



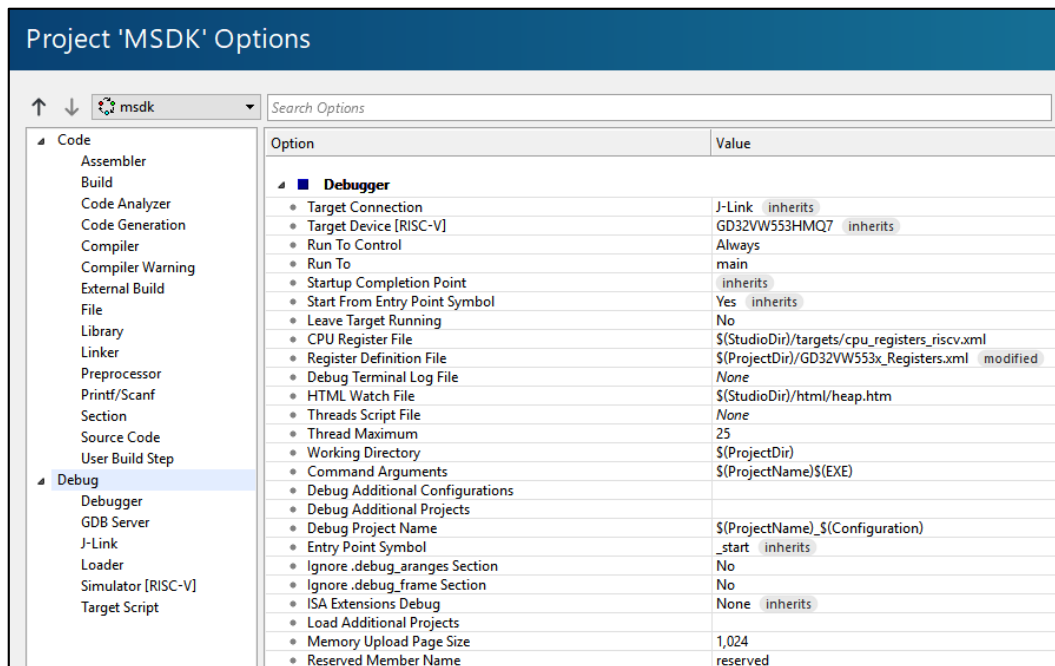
5.4. 调试

■ 调试配置

SES IDE 建议使用 jlink 调试，且 jlink driver 版本至少为 V7.92o，该版本 jlink driver 支持 GD32VW55x 芯片。

工程默认已经配置好 Debug 信息，如需更改，右击 MSDK 工程，点击 Options 打开配置界面，可在 Debug 选项下修改 Debugger 和 J-Link，如[图 5-11 MSDK SES 工程配置界面](#)所示。

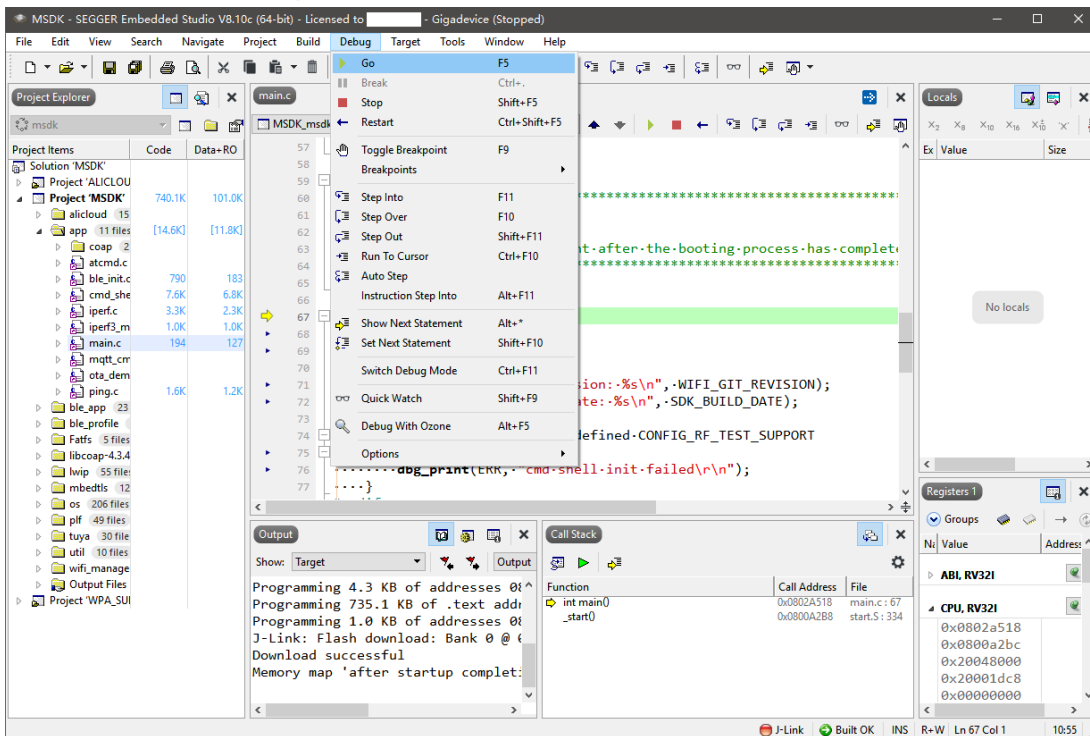
图 5-11 MSDK SES 工程配置界面



■ 开始调试

点击菜单栏的 Debug->GO 即可调试，点击后等待烧录 image 完成后进入如 [图 5-12 SES IDE Debug 界面](#) 所示界面。

图 5-12 SES IDE Debug 界面



6. 常见问题

6.1. No image 错误

打印 ERR: No image to boot (ret = -5) 。

原因：前一次引导 WIFI_IOT 出错，MBL 记录该 IMAGE 运行异常，如果另一个 IMAGE 未烧录或者也发生过引导异常，则会打印此消息。也就是说 MBL 认为没有可跳转的合法 IMAGE，引导失败。

解决方法：再烧录一遍 MBL 即可，烧录后 IMAGE 状态会清空。

6.2. 代码跑在 SRAM

如果有程序需要更快速运行，以便达到更高的性能，可以考虑将它们移动到 SRAM 里面运行。

可以打开 GD32VW55x_RELEASE\MSDK\plfrisc\len\gd32w55x.ld，找到“.code_to_sram:”这一行，这段大括号中包含的代码都是运行在 SRAM 的。如果需要添加新内容，可以加在最后面。格式参照已有的文件，例如：

```
KEEP (*port.o* (.text* .rodata*))
```

是将整个 port.c 文件放进 SRAM 运行。例如：

```
KEEP (*tasks.o* (.text.xTaskIncrementTick))
```

是将 tasks.c 中的 xTaskIncrementTick () 函数放进 SRAM 运行。

7. 版本历史

表 7-1. 版本历史

版本号.	说明	日期
1.0	首次发布	2023 年 10 月 17 日
1.1	第二章开发环境内容修订	2024 年 01 月 26 日
1.2	新增 SES IDE 工程, GD32 Eclipse IDE 更新为 GD32 Embedded Builder	2024 年 7 月 17 日
1.3	更新部分图示与 SDK1.0.3 相匹配	2025 年 4 月 8 日

Important Notice

This document is the property of GigaDevice Semiconductor Inc. and its subsidiaries (the "Company"). This document, including any product of the Company described in this document (the "Product"), is owned by the Company according to the laws of the People's Republic of China and other applicable laws. The Company reserves all rights under such laws and no Intellectual Property Rights are transferred (either wholly or partially) or licensed by the Company (either expressly or impliedly) herein. The names and brands of third party referred thereto (if any) are the property of their respective owner and referred to for identification purposes only.

To the maximum extent permitted by applicable law, the Company makes no representations or warranties of any kind, express or implied, with regard to the merchantability and the fitness for a particular purpose of the Product, nor does the Company assume any liability arising out of the application or use of any Product. Any information provided in this document is provided only for reference purposes. It is the sole responsibility of the user of this document to determine whether the Product is suitable and fit for its applications and products planned, and properly design, program, and test the functionality and safety of its applications and products planned using the Product. The Product is designed, developed, and/or manufactured for ordinary business, industrial, personal, and/or household applications only, and the Product is not designed or intended for use in (i) safety critical applications such as weapons systems, nuclear facilities, atomic energy controller, combustion controller, aeronautic or aerospace applications, traffic signal instruments, pollution control or hazardous substance management; (ii) life-support systems, other medical equipment or systems (including life support equipment and surgical implants); (iii) automotive applications or environments, including but not limited to applications for active and passive safety of automobiles (regardless of front market or aftermarket), for example, EPS, braking, ADAS (camera/fusion), EMS, TCU, BMS, BSG, TPMS, Airbag, Suspension, DMS, ICMS, Domain, ESC, DCDC, e-clutch, advanced-lighting, etc.. Automobile herein means a vehicle propelled by a self-contained motor, engine or the like, such as, without limitation, cars, trucks, motorcycles, electric cars, and other transportation devices; and/or (iv) other uses where the failure of the device or the Product can reasonably be expected to result in personal injury, death, or severe property or environmental damage (collectively "Unintended Uses"). Customers shall take any and all actions to ensure the Product meets the applicable laws and regulations. The Company is not liable for, in whole or in part, and customers shall hereby release the Company as well as its suppliers and/or distributors from, any claim, damage, or other liability arising from or related to all Unintended Uses of the Product. Customers shall indemnify and hold the Company, and its officers, employees, subsidiaries, affiliates as well as its suppliers and/or distributors harmless from and against all claims, costs, damages, and other liabilities, including claims for personal injury or death, arising from or related to any Unintended Uses of the Product.

Information in this document is provided solely in connection with the Product. The Company reserves the right to make changes, corrections, modifications or improvements to this document and the Product described herein at any time without notice. The Company shall have no responsibility whatsoever for conflicts or incompatibilities arising from future changes to them. Information in this document supersedes and replaces information previously supplied in any prior versions of this document.