

**GigaDevice Semiconductor Inc.**

**GD32VW553 快速开发指南**

**应用笔记**

**AN154**

1.0 版本

(2023 年 10 月)

# 目录

目录.....	2
图索引.....	4
表索引.....	5
<b>1. 认识开发板.....</b>	<b>6</b>
<b>1.1. 开发板实物图.....</b>	<b>6</b>
1.1.1. START 开发板.....	6
1.1.2. EVAL 开发板.....	6
<b>1.2. 启动模式.....</b>	<b>7</b>
<b>1.3. 调试器接口.....</b>	<b>8</b>
<b>1.4. 下载接口.....</b>	<b>8</b>
<b>1.5. 查看日志.....</b>	<b>8</b>
<b>2. 搭建开发环境.....</b>	<b>10</b>
<b>2.1. GD32Eclipse IDE 安装.....</b>	<b>10</b>
<b>3. 开发需知.....</b>	<b>11</b>
<b>3.1. SDK 执行程序组.....</b>	<b>11</b>
<b>3.2. SDK 配置.....</b>	<b>11</b>
3.2.1. 无线模块配置.....	11
3.2.2. SRAM 布局.....	12
3.2.3. FLASH 布局.....	12
3.2.4. 固件版本号.....	12
3.2.5. APP 配置.....	13
3.2.6. Configurations 选择.....	13
<b>3.3. 正确日志示例.....</b>	<b>13</b>
<b>4. GD32Eclipse IDE 工程.....</b>	<b>14</b>
<b>4.1. 打开工程组.....</b>	<b>14</b>
<b>4.2. 编译.....</b>	<b>17</b>
<b>4.3. 下载.....</b>	<b>23</b>
<b>4.4. 调试.....</b>	<b>23</b>
<b>5. 常见问题.....</b>	<b>26</b>
<b>5.1. DAPLINK 盘识别.....</b>	<b>26</b>
<b>5.2. No image 错误.....</b>	<b>27</b>

---

5.3. 代码跑在 SRAM .....	27
6. 版本历史 .....	28

## 图索引

图 1-1. START 开发板实物图.....	6
图 1-2. EVAL 开发板实物图.....	7
图 1-3. 开发板类型配置.....	7
图 1-4. 设备和驱动器列表.....	8
图 1-5. 串口配置.....	9
图 2-1. GD32Eclipse IDE 工具包.....	10
图 3-1. 启动过程.....	11
图 3-2. 无线模块配置.....	11
图 3-3. SRAM 布局.....	12
图 3-4. FLASH 布局.....	12
图 3-5. 固件版本号.....	13
图 3-6. 工程启动信息.....	13
图 4-1. SDK 目录.....	14
图 4-2. 启动 GD32EclipseIDE.....	14
图 4-3. Open Projects from file System.....	15
图 4-4. 选择 MBL 工程路径.....	15
图 4-5. MBL 工程界面.....	16
图 4-6. 选择 MSDK 工程路径.....	16
图 4-7. MSDK 和 MBL 工程界面.....	17
图 4-8. 打开工程 Properties.....	18
图 4-9. 工具链配置.....	19
图 4-10. 编译 MBL 工程.....	20
图 4-11. 编译 MBL 结果.....	21
图 4-12. target project 选择.....	21
图 4-13. 编译 MSDK 工程.....	22
图 4-14. MSDK 编译结果.....	22
图 4-15. images 输出.....	23
图 4-16. 打开调试配置选项.....	23
图 4-17. MSDK 调试配置.....	24
图 4-18. 进入 MSDK 调试.....	24
图 4-19. MSDK 调试界面.....	25
图 5-1. Mbed 串口驱动安装.....	26
图 5-2. 设备管理器串口列表.....	26

## 表索引

表 1-1. 启动模式.....	8
表 6-1. 版本历史.....	28

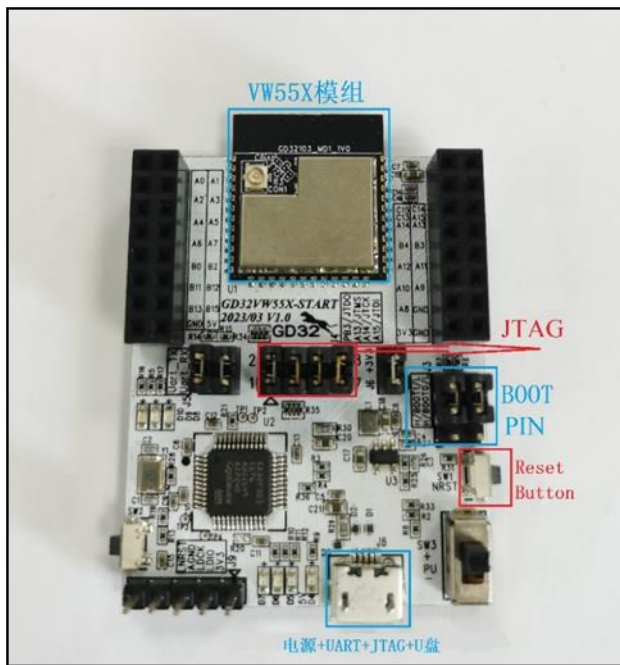
## 1. 认识开发板

### 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



对于开发者来说，可能主要关注开发板的以下几个部分，都已经在图中标注出来。

- 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

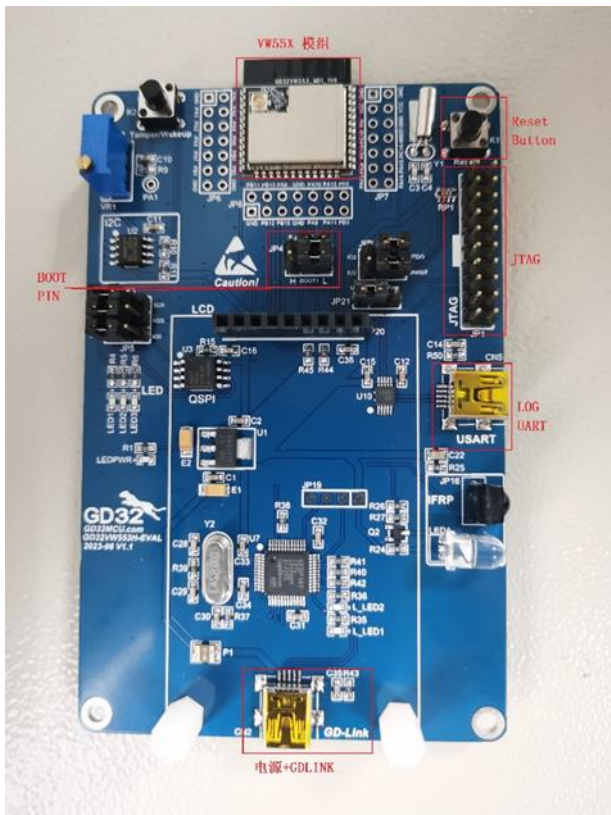
EVAL 开发板由底板和模组组成，模组搭载了 GD32VW55x WiFi+BLE 芯片，底板提供了众多外设测试口，例如：I2C，IFRP，ADC 等等。

对于开发者来说，可能主要关注开发板的以下几个部分，都已经在图中标注出来。

- 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
#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 BOOT SWD 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 (GD32F303)，可以搭配 OpenOCD 使用，但受限于芯片能力，调试和下载速度较慢，建议在开发板 JTAG 接口处，外接 GDLINK 调试器或者 JLINK 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 GDLINK 调试器或者 JLINK 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 GDLINK 调试器或者 JLINK 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 DAPLINK 盘。将 image-all.bin 文件（见后续章节）直接拷贝进 DAPLINK 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。

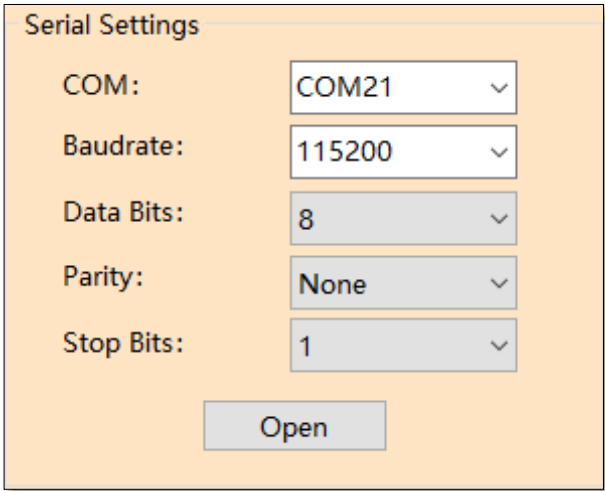
### 1.5. 查看日志

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



The image shows a 'Serial Settings' dialog box with a light orange background. It contains five configuration options, each with a dropdown menu:

- COM: COM21
- Baudrate: 115200
- Data Bits: 8
- Parity: None
- Stop Bits: 1

At the bottom center of the dialog is an 'Open' button.

## 2. 搭建开发环境

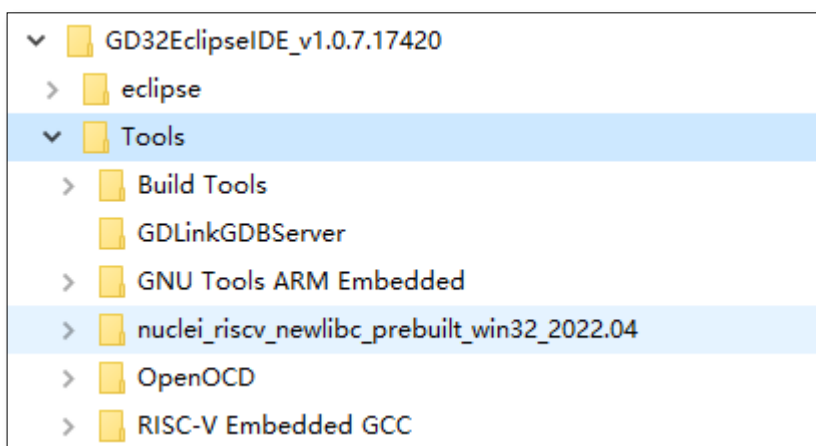
在编译和烧录固件之前，需要搭建开发环境。

目前使用的开发工具是 GD32Eclipse IDE。

### 2.1. GD32Eclipse IDE 安装

- 解压 GD32EclipseIDE\_v1.0.7.17420.7z。
- 检查工具包，工具包内容如[图 2-1. GD32Eclipse IDE 工具包](#)所示。

图 2-1. GD32Eclipse IDE 工具包



- 安装 JDK，双击 GD32EclipseIDE 目录下的 eclipse/jdk/jdk-8u152-windows-x64.exe。
- 启动 IDE，双击 GD32EclipseIDE 目录下的 eclipse/GD32EclipseIDE.exe。

## 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，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

配置文件为 GD32VW55x\_RELEASE\config\config\_gdm32.h。修改以下宏定义值，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\plfriscv\env\_ Eclipse\gd32vw55x.ld 文件。

### 3.2.3. FLASH布局

配置文件为 GD32VW55x\_RELEASE\config\config\_gdm32.h。修改以下宏定义值，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 ..... 0 ..... /* !Keep unchanged! */
#define RE_SYS_STATUS_OFFSET ..... 0x8000 ..... /* !Keep unchanged! */
#define RE_IMG_0_OFFSET ..... 0xA000
#define RE_IMG_1_OFFSET ..... 0x200000
#define RE_END_OFFSET ..... 0x400000
```

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\plfriscv\env\_ Eclipse\gd32vw55x.ld 文件。

### 3.2.4. 固件版本号

配置文件为 GD32VW55x\_RELEASE\config\config\_gdm32.h。修改以下宏定义值，可以指定版本号。但是影响将来用户升级的版本号只有 RE\_IMG\_VERSION。

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 ..... 0x00000600  
#define RE_IMG_VERSION ..... 0x00000600
```

### 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 等等。

### 3.2.6. Configurations选择

SDK 支持两种 Configurations，一种是 msdk，另一种是 msdk\_ffd，默认使用 msdk。与 msdk 相比，msdk\_ffd 支持一些扩展的 WiFi 功能，并实现了更完整的 BLE 功能，相应地占用了较多的 memory 资源。两种 Configurations 链接不同的 lib，msdk 链接了 libwpas 和 libble，msdk\_ffd 链接了 libwpa\_supplicant 和 libble\_max。此外，msdk\_ffd 默认使能了宏 CONFIG\_WPA\_SUPPLICANT，此宏内置在 msdk\_ffd configuration 中。

通常情况下，msdk 即可满足功能需求，如有额外需要如 WFA 认证，可使用 msdk\_ffd。

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

BLE 支持两种配置，ble(对应 libble)和 ble\_max(对应 libble\_max，支持更多功能，参考 blesw/src/export config 和 config\_max 头文件中的特性宏定义)。Configuration 和 BLE lib 是配套使用的，默认 msdk 使用 ble，如果想要 msdk 使用 ble\_max，可修改链接 lib 和包含头文件进行灵活选择。

## 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[常见问题](#)，看能否找到帮助。

图 3-6. 工程启动信息

```
# Build date: 2023/10/12 11:22:01  
=== RF initialization finished ===  
=== WiFi calibration done ===  
=== PHY initialization finished ===  
BLE local addr: AB:89:67:45:23:01, type 0x0  
=== BLE Adapter enable complete ===
```

## 4. GD32Eclipse IDE 工程

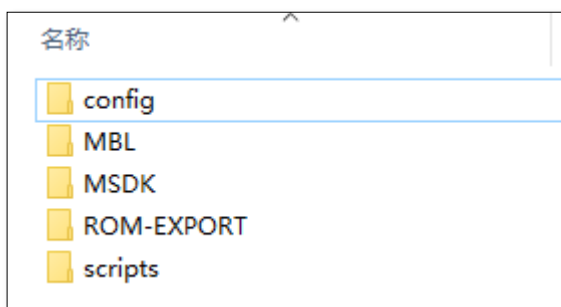
本章将介绍如何在 GD32Eclipse IDE 下编译和调试 SDK。

工程组由 MBL/MSDK 这两个工程组成。MSDK 包含 WiFi 协议栈、BLE 协议栈、外设驱动及应用程序等等，MBL 主要负责从两个 MSDK 固件（一个为当前固件，一个为升级后固件）中选择一个正确的运行。

### 4.1. 打开工程组

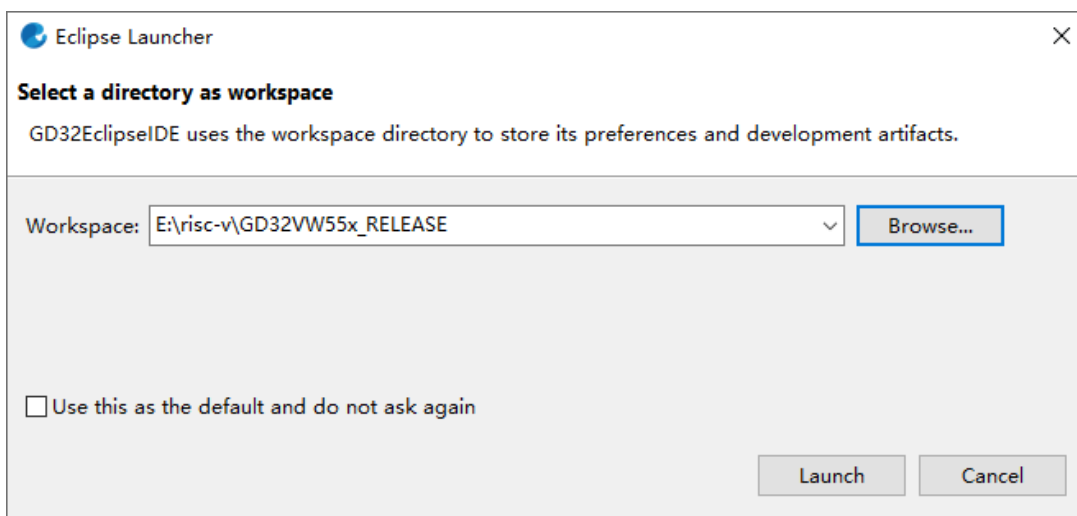
- 检查 SDK 目录 GD32VW55x\_RELEASE，如[图 4-1. SDK 目录](#)所示。

图 4-1. SDK 目录



- 启动 IDE，双击 GD32EclipseIDE 目录下的 eclipse/GD32EclipseIDE.exe，并选择 SDK 目录 GD32VW55x\_RELEASE 为 workspace，点击 launch 按钮，如[图 4-2. 启动 GD32EclipseIDE](#)所示。

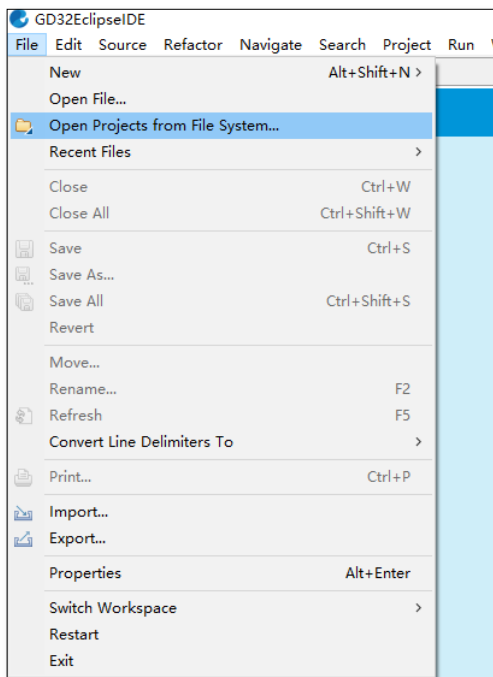
图 4-2. 启动 GD32EclipseIDE



- 导入 MBL 工程

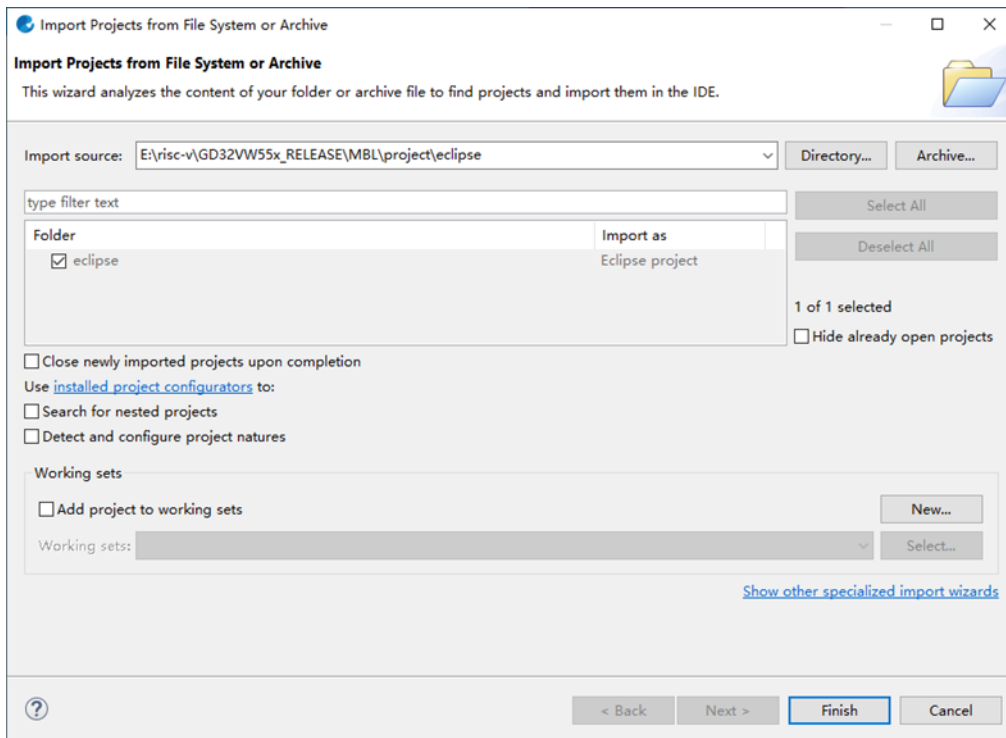
File 菜单点击 Open Projects from file System，如[图 4-3. Open Projects from file System](#)所示。

图 4-3. Open Projects from file System



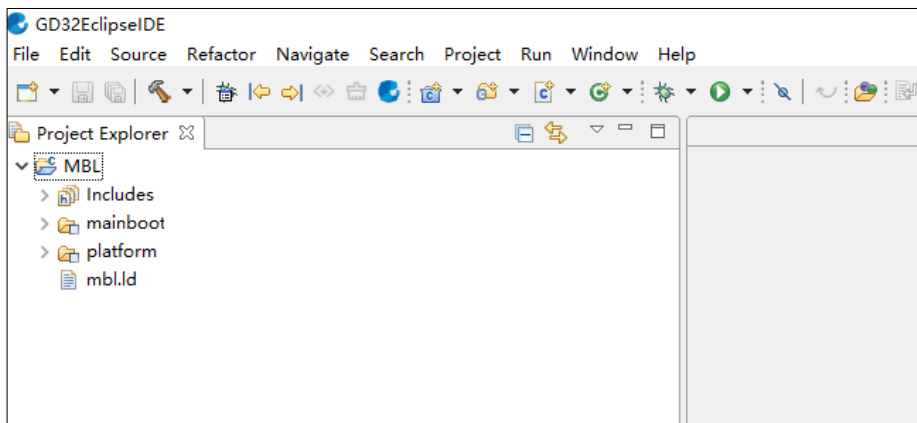
选择工程路径 GD32VW55x\_RELEASE\MBL\project\eclipse，如 [图 4-4. 选择 MBL 工程路径](#) 所示，并点击 finish。

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

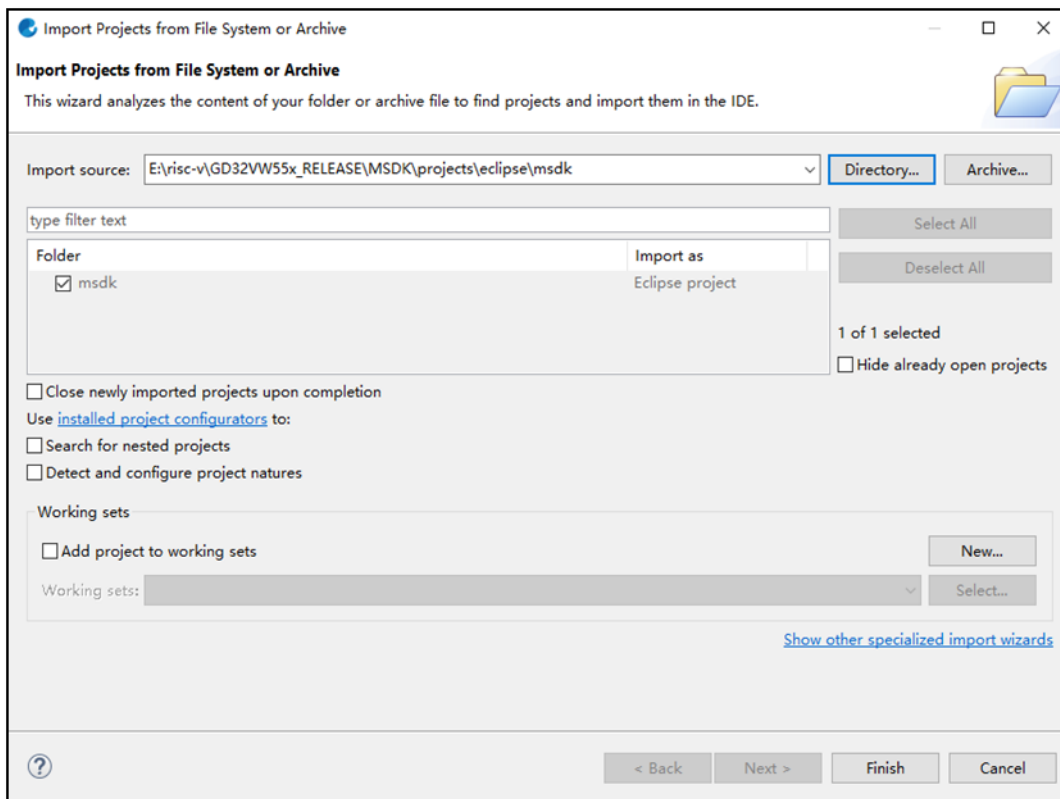


#### ■ 导入 MSDK 工程

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

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。

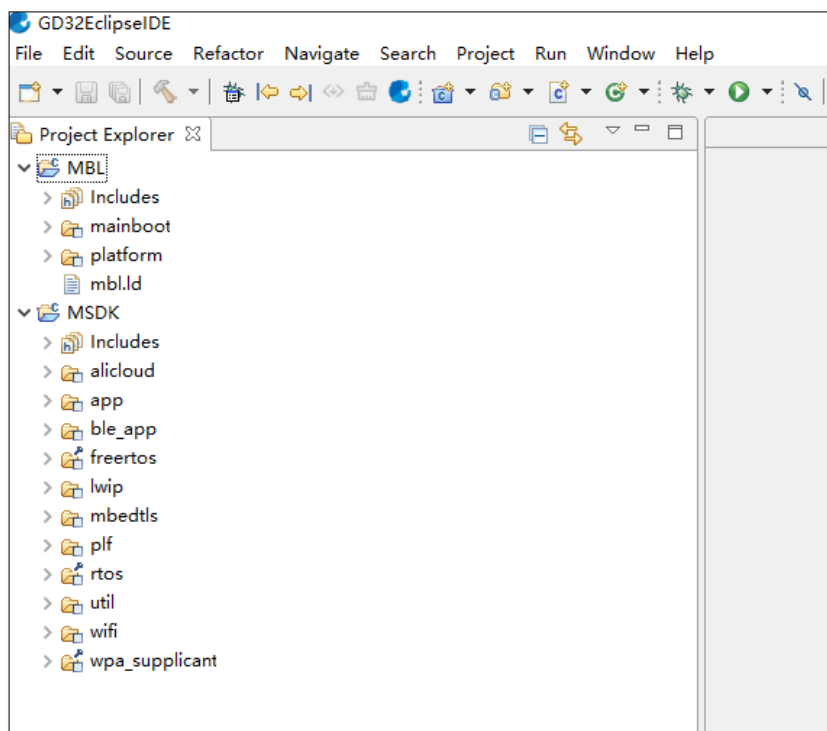
图 4-6. 选择 MSDK 工程路径



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

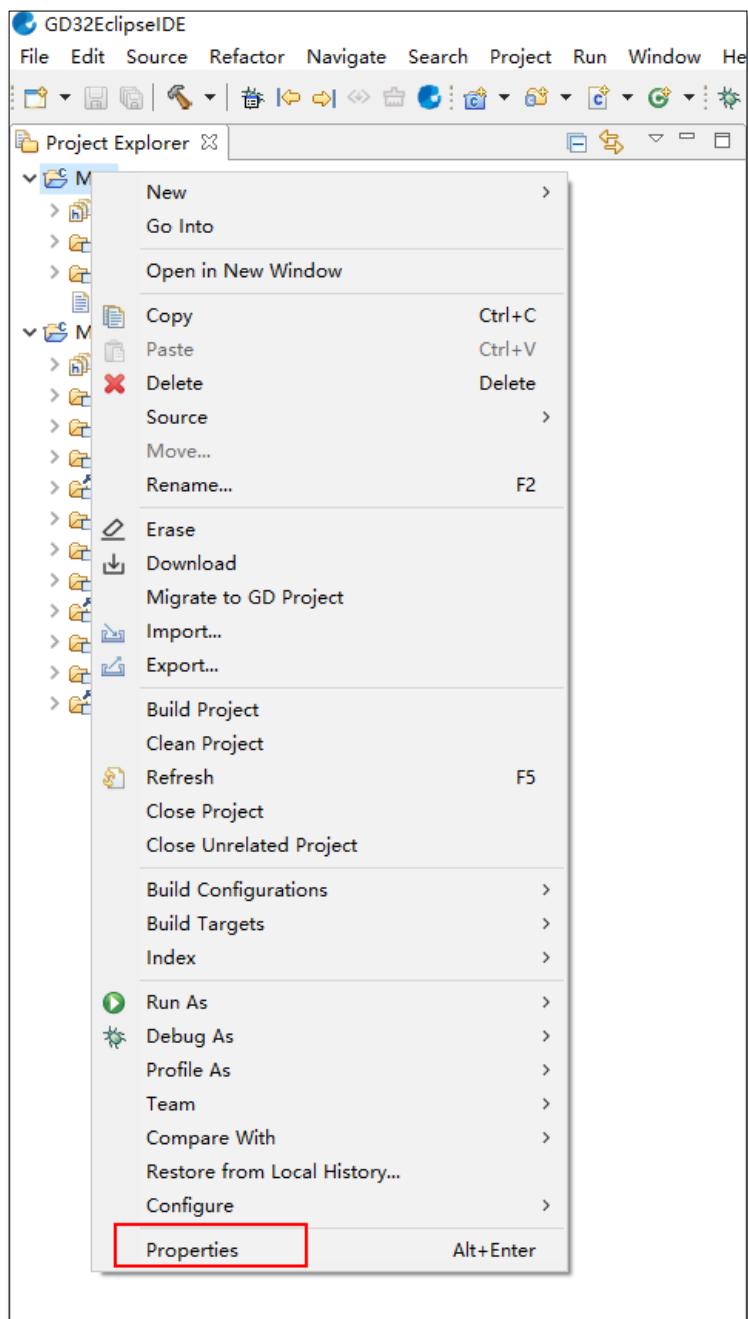


## 4.2. 编译

### ■ 检查工程编译工具配置

右击工程, 点击 properties, 如[图 4-8. 打开工程 Properties](#)所示。

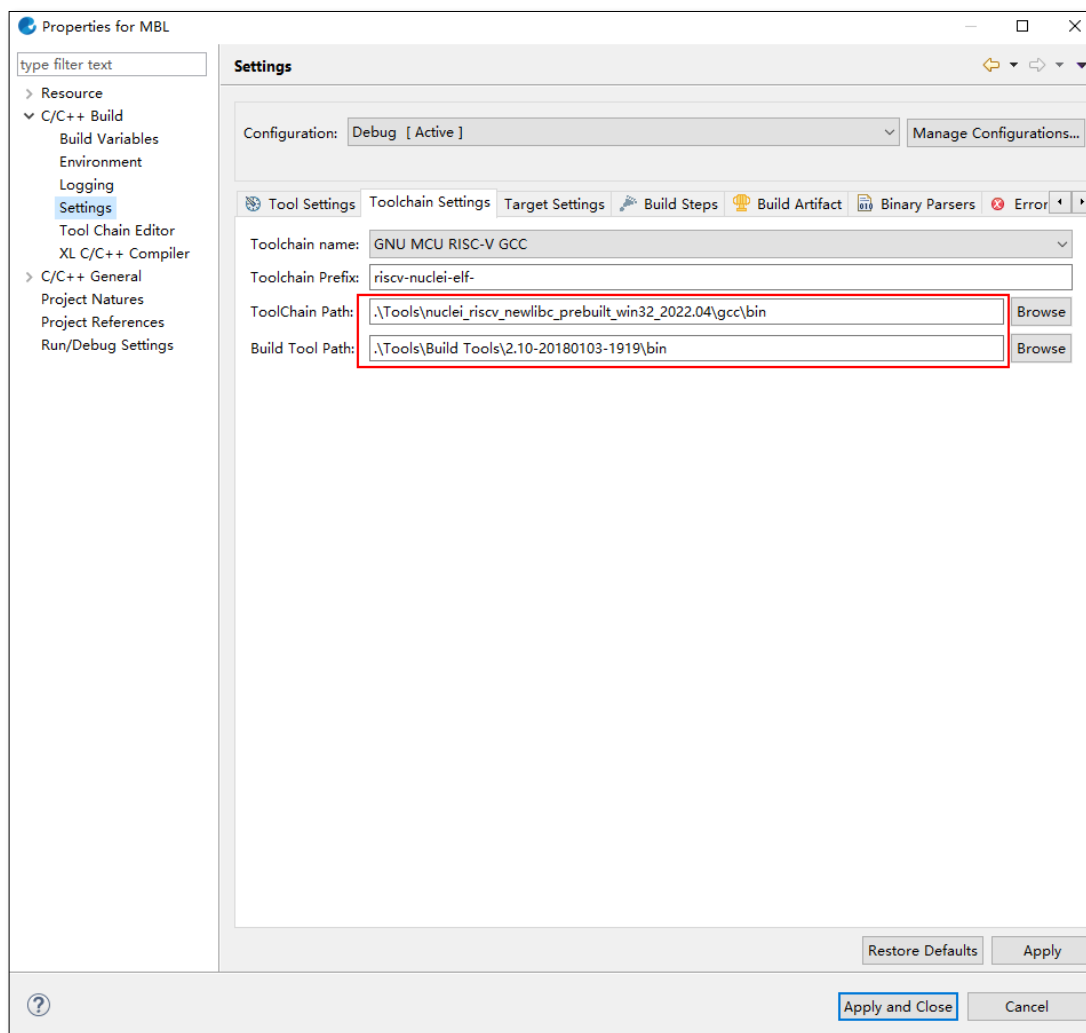
图 4-8. 打开工程 Properties



依次选择 C/C++ Build → Settings，选项卡点击 toolchain settings，如[图 4-9. 工具链配置](#)所示。

默认情况下，路径就是使用 IDE 目录下的 tools，可不用更改。点击 apply and close。

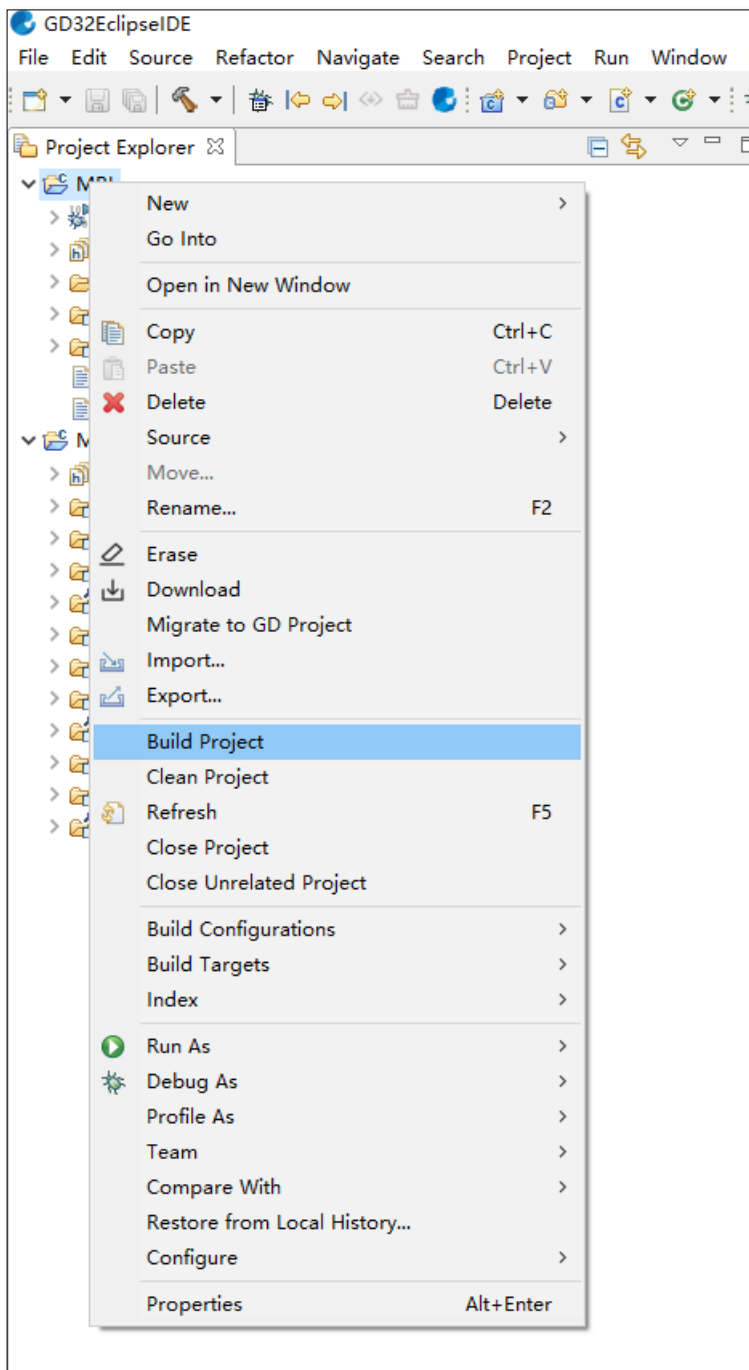
图 4-9. 工具链配置



#### ■ 编译 MBL 工程

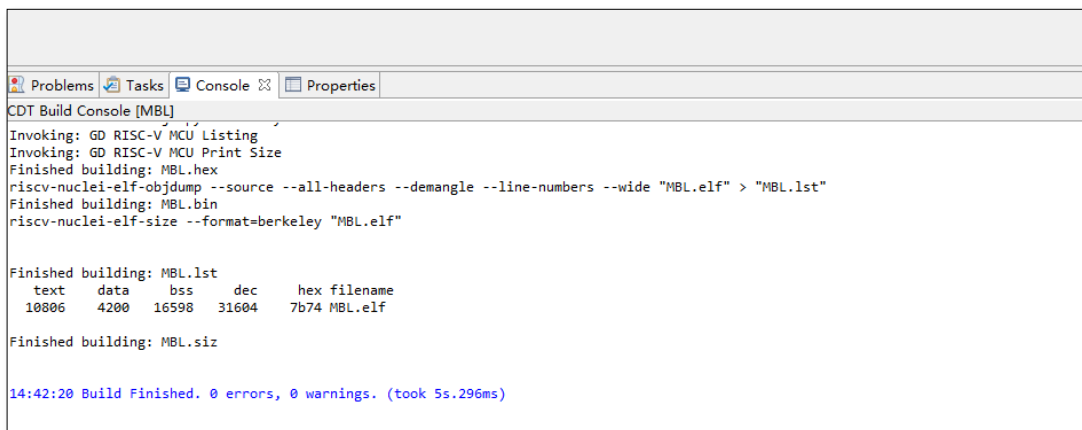
右击工程，点击 build project，如[图 4-10. 编译 MBL 工程](#)所示。

图 4-10. 编译 MBL 工程



编译结果，如[图 4-11. 编译 MBL 结果](#)所示。

图 4-11. 编译 MBL 结果



```

CDT Build Console [MBL]
Invoking: GD RISC-V MCU Listing
Invoking: GD RISC-V MCU Print Size
Finished building: MBL.hex
riscv-nuclei-elf-objdump --source --all-headers --demangle --line-numbers --wide "MBL.elf" > "MBL.lst"
Finished building: MBL.bin
riscv-nuclei-elf-size --format=berkeley "MBL.elf"

Finished building: MBL.lst
text    data    bss     dec     hex filename
10806   4200   16598   31604   7b74 MBL.elf

Finished building: MBL.siz

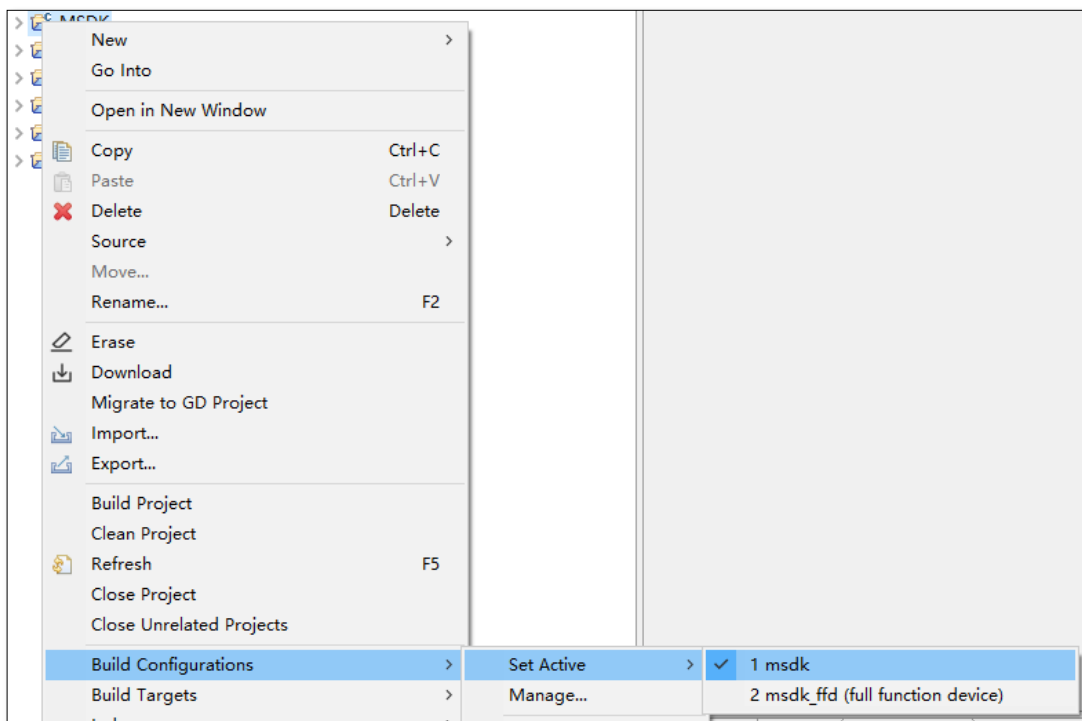
14:42:20 Build Finished. 0 errors, 0 warnings. (took 5s.296ms)

```

■ 编译 MSDK 工程

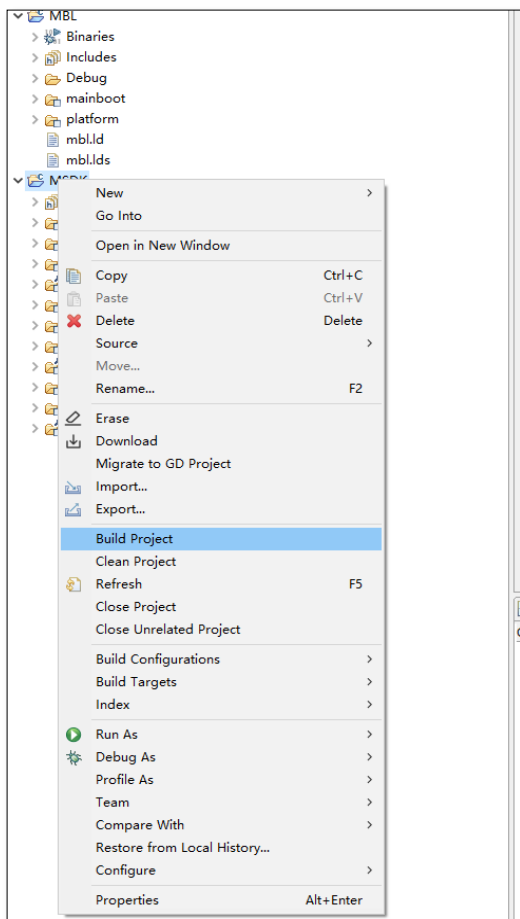
右击工程，依次点击 Build Configurations—>Set Active—><msdk 或 msdk\_ffd>，如 [图 4-12. Configurations 选择](#) 所示，当前默认 Configuration 为 msdk。

图 4-12. Configurations 选择



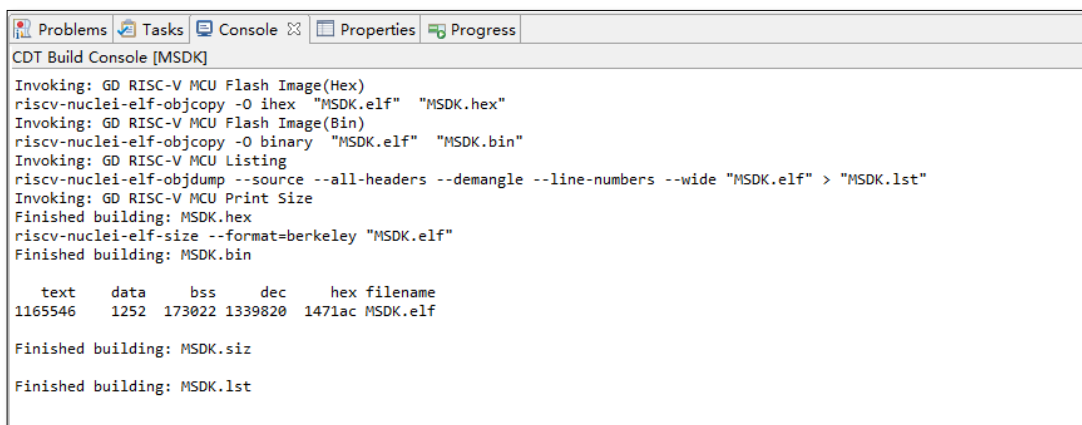
再次右击工程，点击 Build project，如 [图 4-13. 编译 MSDK 工程](#) 所示。

图 4-13. 编译 MSDK 工程



编译结果，如 [图 4-14. MSDK 编译结果](#) 所示。

图 4-14. MSDK 编译结果



#### ■ SDK 生成的 image

在 MSDK 编译完成之后，images 输出在 GD32VW55x\_RELEASE\scripts\images 路径，如 [图 4-15. images 输出](#) 所示。

Image-all.bin 中包含可执行程序段 MBL 和 MSDK，该固件可用于生产，烧录到空白 FLASH 中

图 4-15. images 输出

名称	修改日期
image-all.bin	2023/7/13 14:59
mbl.bin	2023/7/13 14:42
msdk.bin	2023/7/13 14:59
readme.txt	2023/7/13 10:11

### 4.3. 下载

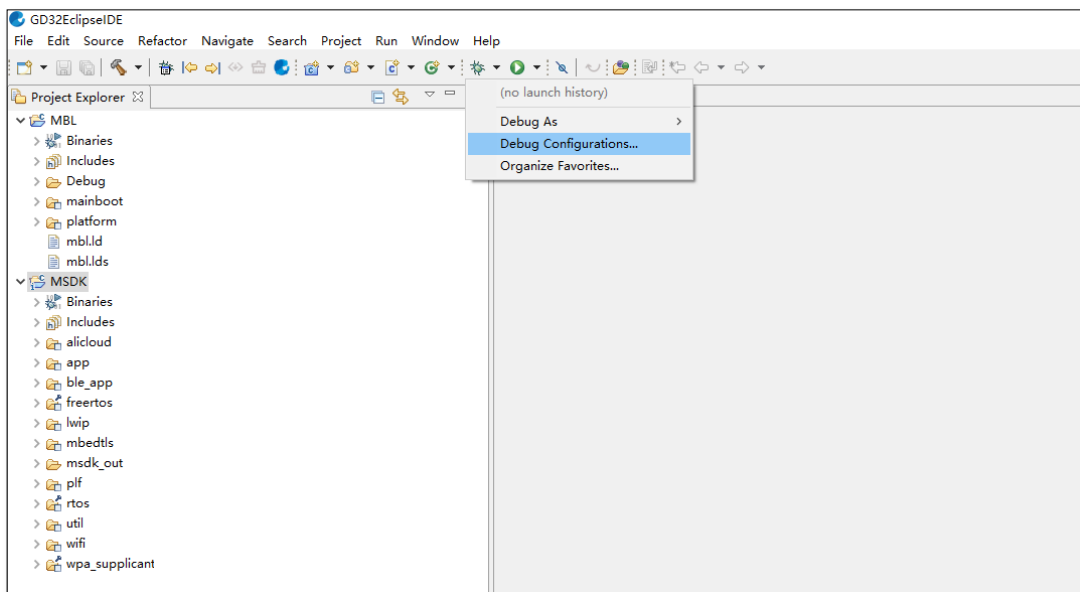
如 [1.4 下载接口](#) 章节拖放 image-all.bin 即可烧写。

### 4.4. 调试

#### ■ 配置调试配置

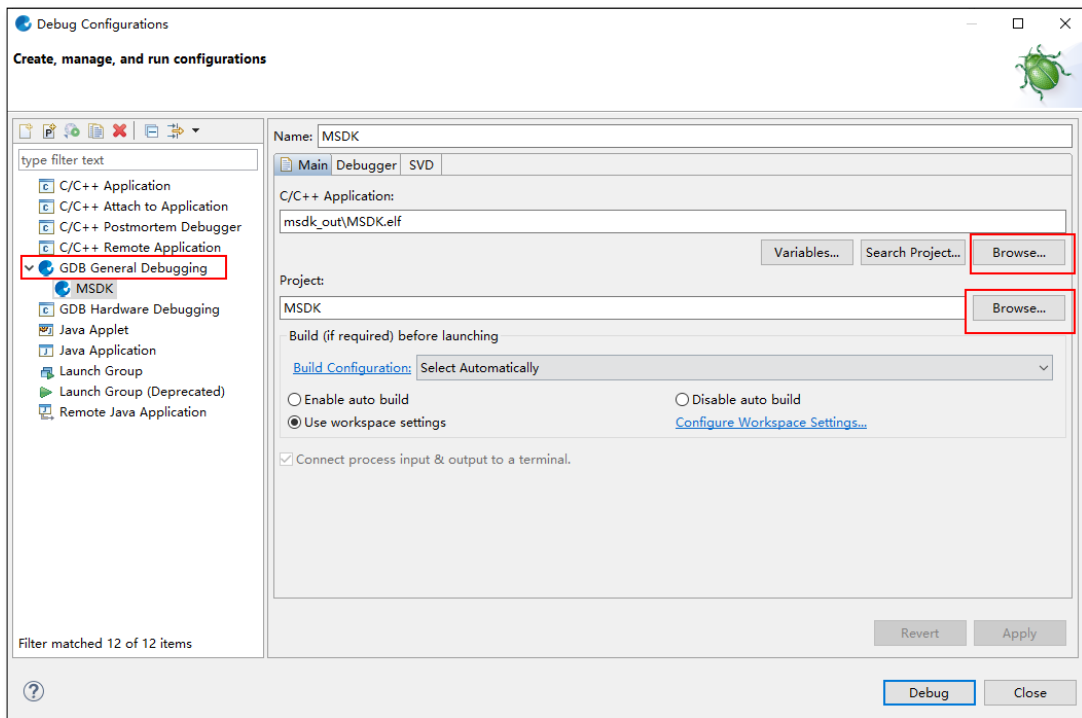
先选择 MSDK 工程，点击 debug 下拉的 debug configuration，如 [图 4-16. 打开调试配置选项](#) 所示。

图 4-16. 打开调试配置选项



双击 GDB General Debugging，可命名为 MSDK，c/c++ application 可通过 browse 选择 msdk\_out\MSDK.elf，project 可通过 browse 选择 MSDK，如 [图 4-17. MSDK 调试配置](#) 所示，最后点 close。

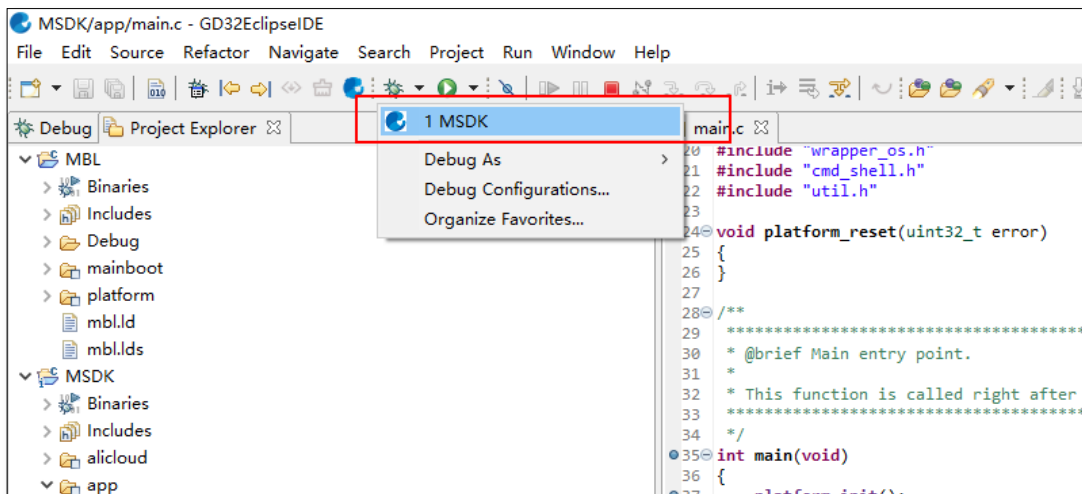
图 4-17. MSDK 调试配置



■ 开始调试

点击 debug 下拉的目标 MSDK 即可开始调试。

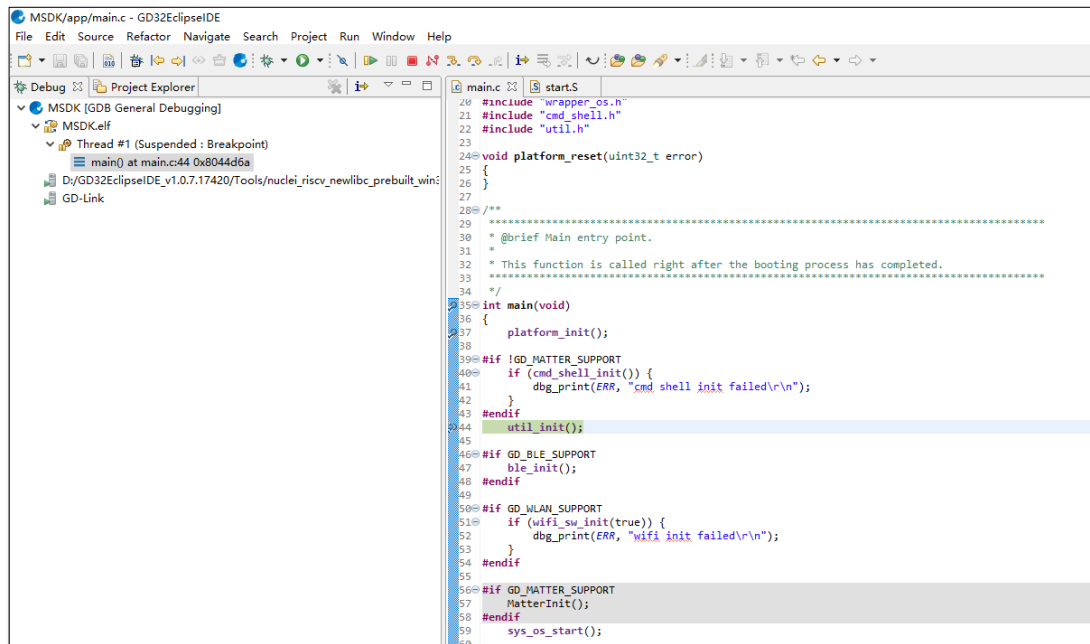
图 4-18. 进入 MSDK 调试



等待 debug session 建立完成之后，点击浏览选项卡下的 debug—>thread 就可开始调试，如 [图 4-19. MSDK 调试界面](#) 所示。



图 4-19. MSDK 调试界面



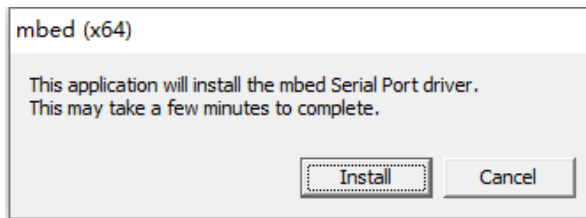
## 5. 常见问题

### 5.1. DAPLINK 盘识别

使用 START 开发板，如果设备管理器中串口不能被识别为“mbedSerialPort (COMx)”，如 Windows 7 及以下版本，则需要安装 Mbed 串口驱动。

- 下载驱动
  - [https://os.mbed.com/media/downloads/drivers/mbedWinSerial\\_16466.exe](https://os.mbed.com/media/downloads/drivers/mbedWinSerial_16466.exe)
- 安装
  - 双击运行 mbedWinSerial\_16466.exe，点击“Install”，如 [图 5-1. Mbed 串口驱动安装](#)，安装成功后，点击“Finish”。

图 5-1. Mbed 串口驱动安装



- 检查
  - 安装成功后，设备管理器能看到串口“mbed Serial Port (COMx)”，如 [图 5-2 设备管理器串口列表](#)，设备和驱动器列表能看到“DAPLINK”盘，如 [图 1-4. 设备和驱动器列表](#)。

图 5-2. 设备管理器串口列表



## 5.2. 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。

## 5.3. 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\plf\riscv\env\_Eclipse\gd32vw55x.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。

## 6. 版本历史

表 6-1. 版本历史

版本号.	说明	日期
1.0	首次发布	2023 年 10 月 17 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 under the intellectual property laws and treaties of the People's Republic of China and other jurisdictions worldwide. The Company reserves all rights under such laws and treaties and does not grant any license under its patents, copyrights, trademarks, or other intellectual property rights. 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.

The Company makes no warranty of any kind, express or implied, with regard to this document or any Product, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose. The Company does not assume any liability arising out of the application or use of any Product described in this document. Any information provided in this document is provided only for reference purposes. It is the responsibility of the user of this document to properly design, program, and test the functionality and safety of any application made of this information and any resulting product. Except for customized products which has been expressly identified in the applicable agreement, the Products are designed, developed, and/or manufactured for ordinary business, industrial, personal, and/or household applications only. The Products are not designed, intended, or authorized for use as components in systems designed or intended for the operation of weapons, weapons systems, nuclear installations, atomic energy control instruments, combustion control instruments, airplane or spaceship instruments, transportation instruments, traffic signal instruments, life-support devices or systems, other medical devices or systems (including resuscitation equipment and surgical implants), pollution control or hazardous substances management, or other uses where the failure of the device or Product could cause personal injury, death, property or environmental damage ("Unintended Uses"). Customers shall take any and all actions to ensure using and selling the Products in accordance with the applicable laws and regulations. The Company is not liable, in whole or in part, and customers shall and hereby do 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 Products. Customers shall indemnify and hold the Company 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 Products.

Information in this document is provided solely in connection with the Products. The Company reserves the right to make changes, corrections, modifications or improvements to this document and Products and services described herein at any time, without notice.