# **GigaDevice Semiconductor Inc.**

# **GD32VW553 Quick Development Guide**

# Application Note AN154

Revision 1.2

(July 2024)



# **Table of Contents**

Table	of Contents	2
List o	f Figures	1
List o	f Tables	ò
1.	Introduction to development board	7
1.1.	Picture of real development board	7
	1.1. The START development board	
1.1 1.2.	2. The EVAL development board	
1.2.	Debugger interface	
-		
1.4.	Download interface	
1.5.	Viewing log10	
	Building development environment1	
2.1.	Installation of GD32 Embedded Builder1	
2.2.	Installation of SEGGER Embedded Studio IDE1	I
2.3.	Toolchain download1′	I
3.	What developers must know13	3
3.1.	SDK execution program group13	3
3.2.	SDK configuration13	3
-	2.1.       Configuration of wireless module	
	2.2. SRAM layout	
• • •	2.4. Firmware version No	
	2.5.       APP configuration	
3.3.	Correct log example	
	GD32 Embedded Builder IDE project1	
4.1.	Opening the project group	
4.1.	Compilation	
4.3.	Download firmware	
4.4.	Debugging	
5.	SEGGER Embedded Studio IDE project20	3



# AN154 GD32VW553 Quick Development Guide

	5.1.	Open projects	26
	5.2.	Compilation	. 27
	5.3.	Download firmware	. 29
	5.4.	Debugging	. 30
6.		FAQ	32
	6.1.	No image error	. 32
	6.2.	Code running in SRAM	. 32
7.		Revision history	33



# **List of Figures**

Figure 1-1. The picture of the START development board	7
Figure 1-2. The picture of the EVAL development board	8
Figure 1-3. Development Board Type Configuration	9
Figure 1-4. List of devices and drivers	. 10
Figure 1-5. Configuration of serial port	. 10
Figure 2-1 The Directory Structure of GD32 Embedded Builder	11
Figure 2-2 Download Toolchain	. 12
Figure 2-3 Nuclei Toolchain Catalog Structure	. 12
Figure 3-1. Boot process	. 13
Figure 3-2. Configuration of wireless module	. 13
Figure 3-3. SRAM layout	. 14
Figure 3-4. FLASH layout	. 14
Figure 3-5. Firmware version No	. 15
Figure 3-6. Project boot information	. 16
Figure 4-1. SDK directory	. 17
Figure 4-2. Starting GD32 Embedded Builder IDE	. 17
Figure 4-3. Open Projects from file System	. 18
Figure 4-4. Selecting MBL project path	. 18
Figure 4-5. MBL project interface	. 19
Figure 4-6. Selecting MSDK project path	. 19
Figure 4-7. MSDK and MBL project interfaces	. 20
Figure 4-8. Properties of the project	
Figure 4-9. Compiling the MBL project	. 21
Figure 4-10. MBL compilation result	. 21
Figure 4-11. target configuration selection	. 22
Figure 4-12. MSDK compilation result	. 22
Figure 4-13. Images output	. 23
Figure 4-14. Opening the Debug Configuration option	. 23
Figure 4-15. MSDK debug configuration	. 24
Figure 4-16. MSDK Debugging Configuration Interface with openocd	. 25
Figure 4-17. MSDK debug interface	. 25
Figure 5-1. MBL SES Project Project Interface	. 26
Figure 5-2. MSDK SES Project Interface	. 27
Figure 5-3. nuclei toolchain content	. 27
Figure 5-4. Compiling the MBL project	. 28
Figure 5-5. MBL compilation result	. 28
Figure 5-6. Compile MSDK project	. 28
Figure 5-7 MSDK Project Configuration Options	. 29
Figure 5-8 MSDK compilation result	. 29



# AN154 GD32VW553 Quick Development Guide

Figure 5-9. Images output	. 29
Fogure 5-10 SES IDE image download	. 30
Figure 5-11. MSDK SES Project Configuration Interface	. 31
Figure 5-12. SES IDE Debug Interface	. 31



# **List of Tables**

Table 1-1. Boot mode	
Table 7-1. Revision history	33



# 1. Introduction to development board

# 1.1. Picture of real development board

### 1.1.1. The START development board

The START development board consists of a baseboard and a module equipped with the GD32VW55x Wi-Fi+BLE chip.



#### Figure 1-1. The picture of the START development board

Mainly focus on the following parts of the development board, which have been marked in the *Figure 1-1. The picture of the START development board*.

- Boot mode (Boot PIN);
- Power supply port (power supply);
- View log (UART);
- Debugger interface (DAPLINK, JLINK, or GDLINK);
- Reboot (Reset Button).



#### 1.1.2. The EVAL development board

The EVAL development board consists of a baseboard and a module equipped with the GD32VW55x Wi-Fi+BLE chip. The baseboard lead out many peripheral test ports, such as I2C, IFRP, ADC and so on.



Figure 1-2. The picture of the EVAL development board

Developers mainly focus on the following parts of the development board, which have been marked in the *Figure 1-2. The picture of the EVAL development board*.

- Boot mode (Boot PIN);
- Power supply port (power supply);
- View log (UART);
- Debugger interface (JLINK, or GDLINK);
- Reboot (Reset Button).

For the START development board and the EVAL development board, the SDK configuration is different and different macros need to be selected to enable them. As shown in *Figure 1-3. Development Board Type Configuration*, the SDK selects the START development board configuration as the default. The configuration file is GD32VW55x\_RELEASE/config/ platform\_def.h.



Figure 1-3. Development Board Type Configuration

// 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. Boot mode

GD32VW55x can boot from ROM, FLASH, or SRAM.

The level selection of the two pins BOOT0 and BOOT1 in the BOOT SWD box of the development board determines the boot mode. See <u>Table 1-1. Boot mode</u>. For more instructions on the boot mode, please refer to the document "GD32VW55x\_User\_Manual".

EFBOOTLK	BOOT0	BOOT1	EFSB	Boot address	Boot area
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

#### Table 1-1. Boot mode

## 1.3. Debugger interface

For START development board, it comes with a DAPLINK(GD32F505) debugger that can be used with OpenOCD. Can also use an external debugger (GD-Link or J-Link) at the JTAG interface of the board for debugging and download. The DAP chip also integrates the UART function, so only one USB cable is required to supply power, debug, and view the log. Connect the pins JCLK, JTWS, JTDO and JTDI to the lower four pins through jumper caps, and then download and debug the code through DAPLINK. *Figure 1-1. The picture of the START development board* shows how to debug through DAPLINK.

For EVAL development board, GD-Link or J-Link debugger can be used for debugging and download. DAPLINK is not supported.



## 1.4. Download interface

For START development board, in addition to the firmware download through the debugger mentioned in the previous section, if debugging is not required and only the firmware needs to be downloaded, it can also be downloaded by dragging it into the USB disk. Connect the development board to the computer through a USB cable, and the GigaDevice disk as shown in *Figure 1-4. List of devices and drivers* is displayed. Copy the image-all.bin file(see subsequent chapters) directly into the GigaDevice disk to complete FLASH downloading of the GD32VW55x chip.

#### Figure 1-4. List of devices and drivers



For EVAL development board, GDLINK or JLINK debugger can be used for download. Dragging into the USB disk is not supported.

# 1.5. Viewing log

Connect a MicroUSB cable to the START development board, use a serial port tool on the PC, and configure it according to the parameters in *Figure 1-5. Configuration of serial port* and connect to the board. After that, use the serial port to output logs.

#### Figure 1-5. Configuration of serial port

Serial Settings		
COM:	COM21	~
Baudrate:	115200	~
Data Bits:	8	$\sim$
Parity:	None	$\sim$
Stop Bits:	1	~
0	pen	



# 2. Building development environment

Build a development environment before compiling and downloading the firmware.

The development tool currently used is GD32 Embedded Builder and SEGGER Embedded Studio IDE.

## 2.1. Installation of GD32 Embedded Builder

The GD32 Embedded Builder can select GD32VW5 at website: <u>https://gd32mcu.com/cn/download</u> to download. The uncompress downloaded files is as *Figure 2-1 The Directory Structure of GD32 Embedded Builder* shows. It is also recommended to unzip the "nuclei\_riscv\_newlibc\_prebuilt\_win32\_2022.04" download in subsection <u>2.3Toolchain download</u> and place it in the Tools directory as well.

#### Figure 2-1 The Directory Structure of GD32 Embedded Builder



# 2.2. Installation of SEGGER Embedded Studio IDE

Please visit the website: <u>https://wiki.segger.com/GD32V</u> for how to get the SEGGER Embedded Studio IDE and License Activation Key

### 2.3. Toolchain download

download

Official website: https://nucleisys.com/download.php

As shown in *Figure 2-2 Download Toolchain*, ToolChain selects version 2022.04.



The directory structure of the downloaded Nuclei RISC-V Toolchain after unpacking is shown in *Figure 2-3 Nuclei Toolchain Catalog Structure* 

#### Figure 2-2 Download Toolchain

	芯来工具链	
Nuclei RISC-V Toolchain ( Baremetal/RTOS )		♦ Windows ▼
Nuclei RISC-V Toolchain ( Linux/Glibc )		2024.02
Nuclei OpenOCD		2022.12 2022.08 2022.04 dows ▼ ▲ Linux x86-64 ▼  Conline Doc
Nuclei QEMU		2022.01 2020.08 ▲ Linux x86-64 ▼ Conline Doc
Nuclei Studio IDE	▲ Windows x86-64 ▼	x x86-64 🔻 🚺 🙆 Nuclei Studio User Guide 💌 📝 🕅 Nuclei Studio FAQ
Windows Build Tools		▲ Windows ▼

#### Figure 2-3 Nuclei Toolchain Catalog Structure

nuclei_riscv_newlibc_prebuilt_win32_2022.04.zip > gcc	
	类型
📜 bin	文件夹
📕 include	文件夹
📕 lib	文件夹
📙 libexec	文件夹
📙 riscv-nuclei-elf	文件夹
📕 share	文件夹
📔 build.txt	TXT 文件
📔 gitrepo.txt	TXT 文件



# 3. What developers must know

Before getting started with development, first understand the members of the SDK execution program group, how to correctly configure the SDK.

## 3.1. SDK execution program group

SDK will finally generate two main execution programs: MBL (Main Bootloader) and MSDK (Main SDK), which will eventually be downloaded to FLASH to run. After power-on, the programs will boot from Reset\_Handler of MBL, and then jump to the MSDK main program to run, as shown in *Figure 3-1. Boot process*.

#### Figure 3-1. Boot process



### 3.2. SDK configuration

#### 3.2.1. Configuration of wireless module

The configuration file is GD32VW55x\_RELEASE/config/platform\_def.h, whose main content is as shown in *Figure 3-2. Configuration of wireless module*.

Figure 3-2. Configuration of wireless module

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

- In the case of BLE/ WiFi combo mode, please enable:
  - #define CFG\_WLAN\_SUPPORT
  - #define CFG\_BLE\_SUPPORT
- In the case of BLE only, please only enable:
  - #define CFG\_BLE\_SUPPORT
- In the case of WiFi only, please only enable:
  - #define CFG\_WLAN\_SUPPORT
- To disable the wireless module, please disable all



#### 3.2.2. SRAM layout

The configuration file is GD32VW55x\_RELEASE\config\config\_gdm32.h. Modify the following macro definition (as *Figure 3-3. SRAM layout* shows) values to plan the SRAM space occupied by the executable program segments MBL and IMG. These values are offset addresses, and the base address is defined at the beginning of the file.

The line marked "!Keep unchanged!" cannot be modified; otherwise, the operation of the MbedTLS code in the ROM will be affected.

#### Figure 3-3. SRAM layout

/\* SRAM LAYOUT \*/
#define RE\_MBL\_DATA\_START 0x300 /\* !Keep unchanged! \*/
#define RE\_IMG\_DATA\_START 0x200 /\* !Keep unchanged! \*/

For the planning of SRAM space in each executable program segment, refer to the .ld file under the corresponding project, such as MBL\project\eclipse\mbl.ld and MSDK\plf\riscv\env\gd32vw55x.ld.

#### 3.2.3. FLASH layout

The configuration file is GD32VW55x\_RELEASE\config\config\_gdm32.h. Modify the following macro definition(as *Figure 3-4. FLASH layout* shows) values to plan the FLASH space occupied by the executable program segments MBL and MSDK. These values are offset addresses, and the base address is defined at the beginning of the file.

The line marked "!Keep unchanged!" cannot be modified; otherwise, the operation of the project will be affected.

#### Figure 3-4. FLASH layout

/* FLASH LAYEROUT */
#define RE_VTOR_ALIGNMENT ······ 0x200 ······ /* ·!Keep ·unchanged! ·*/
#define RE_SYS_SET_OFFSET ······0x0 ······/* !Keep unchanged! */
#define RE_MBL_OFFSET
#define RE_SYS_STATUS_OFFSET · · · · 0x8000 · · · · · · /* · !Keep · unchanged! · */
#define RE_IMG_0_OFFSET ········0xA000······/*·!Keep ·unchanged! ·*/
#define RE_IMG_1_OFFSET ·······0x1E0000
#define·RE_IMG_1_END······@x3CB000·····/*·reserved·192KB·for·user·data·*/
#define·RE_NVDS_DATA_OFFSET·····0x3FB000·····/*·reserved·20KB·for·nvds·data·*/
#define-RE_END_OFFSET ····································

For the planning of FLASH space in each executable program segment, refer to the .ld file under the corresponding project, such as MBL\project\eclipse\mbl.ld and MSDK\plf\riscv\env\gd32vw55x.ld.

#### 3.2.4. Firmware version No.

The configuration file is GD32VW55x\_RELEASE\config\config\_gdm32.h. Modify the following



macro definition values showed in *Figure 3-5. Firmware version No.* to specify the version No. In addition, the macro RE\_IMG\_VERSION is used in Securt Boot to determine the firmware version.

MBL only supports local upgrade, while IMG supports online upgrade. The version No. released by the SDK is consistent with RE\_IMG\_VERSION.

#### Figure 3-5. Firmware version No.



#### 3.2.5. APP configuration

The configuration file is GD32VW55x\_RELEASE\MSDK\app\app\_cfg.h. Choose whether to enable some applications, such as ATCMD, Alibaba Cloud, MQTT, COAP and so on.

#### 3.2.6. Configuration Selection

MSDK supports various Configurations, namely: msdk (default), msdk\_ffd (ffd: full function device), msdk\_threadx, msdk\_ffd\_threadx, msdk\_azure. Compared to msdk, msdk\_ffd supports non-usable but more complete WiFi functionality and implements more complete BLE functionality, but accordingly takes up more memory resources. The ffd and non-ffd in configurations link to different libs, msdk, msdk\_threadx and msdk\_azure link to libwpas and libble, instead msdk\_ffd and msdk\_ffd\_threadx link to libwpa\_supplicant and libble\_max. In addition, ffd enables macro CONFIG\_WPA\_SUPPLICANT by default, and this macro is built into the ffd configuration. The msdk\_azure is a configuration exclusive to the azure cloud, which can be selected for using the azure cloud.

In the WiFi section, msdk can meet most of the requirements, ffd supports more complete functions, such as: can connect to WPS-enabled APs, can connect to enterprise Aps with EAP-TLS; in addition, if you need to pass the WFA authentication, you also need to use ffd.

In the BLE section, msdk supports only peripheral and 1M PHY; ffd additionally supports central, periodic advertising, 2M and coded PHY, GATT client, BIS and CIS.

For details on how to make configuration selection for actual use, see the Compiling MSDK Projects section in subsection <u>4.2Compilation</u>.

### 3.3. Correct log example

After the firmware group (MBL+MSDK) is successfully downloaded, open the serial port tool, and press the Reset button on the development board. The startup information is shown in *Figure 3-6. Project boot information*. If an exception occurs, please check <u>6FAQ</u> for help.



#### Figure 3-6. Project boot information

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: v1.0.2-239348362415d646 Build date: 2024/07/17 10:53:06 === WiFi calibration done === === PHY initialization finished === BLE local addr: 76:BA:ED:23:00:05, type 0x0 === BLE Adapter enable complete ===



# 4. GD32 Embedded Builder IDE project

This chapter introduces how to compile and debug the SDK under GD32 Embedded Builder IDE.

The project group consists of two projects: MBL/MSDK. MSDK includes Wi-Fi protocol stack, BLE protocol stack, peripheral drivers, applications, etc. The MBL is mainly responsible for selecting the correct MSDK firmware from the two (current firmware and OTA firmware) to run.

# 4.1. Opening the project group

Check the SDK directory GD32VW55x\_RELEASE, as shown in <u>Figure 4-1. SDK</u> <u>directory</u>.

#### Figure 4-1. SDK directory

1	~ 名称
	config
	MBL
	MSDK
	ROM-EXPORT
	scripts

To start the IDE, double-click Embedded Builder.exe in the Embedded Builder directory, and select the SDK directory GD32VW55x\_RELEASE as the workspace, and then click the launch button, as shown in *Figure 4-2. Starting GD32 Embedded Builder IDE*.

#### Figure 4-2. Starting GD32 Embedded Builder IDE

GD Eclipse Launcher	×
Select a directory as workspace	
Embedded Builder uses the workspace directory to store its preferences and development artifacts.	
Workspace: D:\risc-v\GD32VW55x_RELEASE	
Use this as the default and do not ask again	
Recent Workspaces	
Launch Cano	el

#### Import the MBL project

In the File menu, click Open Projects from file System, as shown in *Figure 4-3. Open Projects from file System*.



#### Figure 4-3. Open Projects from file System

GD E	mbed-Builder - Embedde	d Builder		
File	Edit Source Refactor	Navigate	Search	Project
	New		Alt+Sh	ift+N >
	Open File			
	Open Projects from File S	System		
	Recent Files			>
	Close		C	trl+W
	Close All		Ctrl+Sh	ift+W
	Save		C	Ctrl+S
	Save As			
R	Save All		Ctrl+S	nift+S
	Revert			
	Move			
	Rename			F2
\$	Refresh			F5
	Convert Line Delimiters T	ō		>
۵	Print		C	Ctrl+P
2	Import			
4	Export			
	Properties		Alt+	Enter
	Switch Workspace			>
	Restart			
	Exit			

Select the project path GD32VW55x\_RELEASE\MBL\project\eclipse, as shown in *Figure* <u>4-4. Selecting MBL project path</u>, and click Finish.

Figure 4-4. Selecting MBL project path

Import Project	s from File System or Archive		— L X
	om File System or Archive zes the content of your folder or archive file to find projects and	import them in the IDE.	
Import source:	D:\risc-v\GD32VW55x_RELEASE_V1.0.2\MBL\project\eclipse	~	Directory Archive
type filter text			Select All
Folder eclipse		Import as Eclipse project	Deselect All
			1 of 1 selected ☐ Hide already open projects
Use installed pro	nported projects upon completion ject configurators to:		
✓ Search for ne ✓ Detect and co	ted projects nfigure project natures		
Working sets	to working sets		New V Select
		Show	other specialized import wizards
?		< Back Next >	Finish Cancel



Close the welcome interface, and the MBL project is shown as *Figure 4-5. MBL project interface* shows.

#### Figure 4-5. MBL project interface



Import the MSDK project

In the File menu, click Open Projects from file System, Select the project path GD32VW55x\_RELEASE\MSDK\projects\eclipse\msdk, as shown in *Figure 4-6. Selecting* <u>MSDK project path</u>, and click Finish.

#### Figure 4-6. Selecting MSDK project path

GD Import Projec	ts from File System or Archive	– 🗆 X					
Import Projects from File System or Archive This wizard analyzes the content of your folder or archive file to find projects and import them in the IDE.							
Import source:	D:\risc-v\GD32VW55x_RELEASE_V1.0.2\MSDK\projects\eclipse\msdk	Directory Archive					
type filter text		Select All					
Folder	Import as	Deselect All					
🗹 msdk	Eclipse project	Deselect All					
Use <u>installed pro</u> ✓ Search for ne	nported projects upon completion <u>oject configurators</u> to: sted projects nfigure project natures	1 of 1 selected □ Hide already open projects					
Add projec	t to working sets	New					
Working sets:		V Select					
?	< Back Next >	other specialized import wizards					

View the MSDK and MBL project interfaces, as shown in *Figure 4-7. MSDK and MBL project interfaces*.



Figure 4-7. MSDK and MBL project interfaces

File Edit Navigate Search Project W	indow I	Help					
in <u>⊕</u> P ← → 10 = = = = = = = = = = = = = = = = = =			- 81	<b>-</b> *5	6 <b>-</b>	0	-
			. 1ta		V	~~	
Project Explorer 🛛 📃 🗖							
🖻 🔄 🏹							
∽ 📂 MBL							
> 🗊 Includes							
> 🔁 mainboot							
> 🔁 platform							
🗃 mbl.ld							
V 😤 MSDK							
> 🔊 Includes							
> 🚘 alicloud							
> 🔁 app							
> 🔁 ble_app							
> 🚘 ble_profile							
> 🚰 coap							
> 🥦 FatFS							
> 🔁 lwip							
> 🔁 mbedtls							
> 🚰 os							
> 🔁 plf							
> 🔁 tuya							
> 🔁 util							
> 🔁 wifi_manager							
> 🔀 azure							

# 4.2. Compilation

Check the configuration of the project compilation tool

Right-click on the project, click on properties, select C/C++ Build -> Settings in order, and on the tab click on toolchain settings., as shown in *Figure 4-8. Properties of the project*.

Figure 4-8. Properties of the project

GD Properties for MBL		-		×
type filter text	Settings		⇔ - ⇒ -	•
> Resource				~
✓ C/C++ Build	Configuration: Del	ug [Active] V Manage Conf	n	
Build Variables	Configuration: Der	Manage Cont	igurations	1
Environment				
Logging Settings	🛞 Tool Settings 1	oolchain Settings 🛛 Target Settings 🎤 Build Steps 🜻 Build Artifact 🗟 Binary Parsers 😣 Error P	arsers	
Tool Chain Editor XL C/C++ Compiler	Toolchain name:	GNU MCU RISC-V GCC	~	
> C/C++ General	Toolchain Prefix:	riscv-nuclei-elf-		
Project Natures Project References	ToolChain Path:	\Tools\nuclei_riscv_newlibc_prebuilt_win32_2022.04\gcc\bin	Browse	
Refactoring History	Build Tool Path:	\Tools\Build Tools\2.10-20180103-1919\bin	Browse	
Run/Debug Settings				
				~
?	м	Apply and Close	Cancel	

Fill in the paths of ToolChain and Build Tool downloaded by <u>2.3Toolchain download</u>. Click Apply and Close.

Compile the MBL project

Right-click the project, and click Build Project, as shown in *Figure 4-9. Compiling the MBL project*.



Figure 4-9. Compiling the MBL project



The compilation result is as shown in Figure 4-10. MBL compilation result.

Figure 4-10. MBL compilation result

```
🔊 Tasks 📃 Console 🛛
CDT Build Console [MBL]
Invoking: GD RISC-V MCU Flash Image(Hex)
riscv-nuclei-elf-objcopy -O ihex "MBL.elf" "MBL.hex"
Invoking: GD RISC-V MCU Flash Image(Bin)
riscv-nuclei-elf-objcopy -O binary "MBL.elf" "MBL.bin"
riscv-nuclei-elf-objcopy -O binary
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"
    text
               data
                           bss
                                       dec
                                                  hex filename
                                                8594 MBL.elf
               4200 17070 34196
  12926
Finished building: MBL.siz
Finished building: MBL.lst
14:19:01 Build Finished. 0 errors, 0 warnings. (took 7s.401ms)
```

After the compilation is complete, the script MBL\project\mbl\_afterbuild.bat will be automatically called to generate mbl.bin and copied to the directory \scripts\images.



#### Compile the MSDK project

Right-click the project, and click Build Configurations—>Set Active—><target configuration> in order. as shown in *Figure 4-11. target configuration selection*, the default target project is msdk.





Right-click the project again, and click Build Project, The compilation result is as shown in *Figure 4-12. MSDK compilation result*.

#### Figure 4-12. MSDK compilation result

😰 Problems 🧟 Tasks 📮 Console 🛛 🔲 Properties 🖶 Progress
CDT Build Console [MSDK]
Script processing completed.
Invoking: GD RISC-V MCU Flash Image(Hex)
riscv-nuclei-elf-objcopy -0 ihex "MSDK.elf" "MSDK.hex"
Invoking: GD RISC-V MCU Flash Image(Bin) riscv-nuclei-elf-objcopy -O binary "MSDK.elf" "MSDK.bin"
Invoking: GD RISC-V MCU Listing
Invoking: GD RISC-V MCU Print Size
Finished building: MSDK.hex
riscv-nuclei-elf-objdumpsourceall-headersdemangleline-numberswide "MSDK.elf" > "MSDK.lst" Finished building: MSDK.bin
risce uclei-elf-sizeformat=berkeley "MSDK.elf"
text data bss dec hex-filename
1231668 1144 103276 1336088 146318 MSDK.elf
Finished building: MSDK.siz

Images generated by SDK

After MSDK is compiled, it will call MSDK\projects\ image\_afterbuild.bat to generate imageota.bin and image-all.bin, and copy the generated bin files to \scripts\images, as shown in *Figure 4-13. Images output*.



image-ota.bin is the bin file generated by MSDK project, which can be used for OTA upgrade. image-all.bin is the combination of MBL(mbl.bin) and MSDK(image-ota.bin), the firmware can be used for production, download into FLASH and run.

Figure	4-13.	Images	output

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. Download firmware

Refer to <u>**1.4 Download interface</u>**, copy GD32VW55x\_RELEASE\scripts\images\imageall.bin to the Gigadevice disc to download bin to flash. And the firmware can also be downloaded via DAPLINK and J-LINK.</u>

## 4.4. Debugging

#### Debugging configuration

Right-click on the MSDK project and click Debug As->Debug Configurations, as shown in *Figure 4-14. Opening the Debug Configuration option*.

#### Figure 4-14. Opening the Debug Configuration option

V 😂 MSD		New	>	h.		
> 🐝 Bi > 🗊 In		Go Into		L .		
> 🔁 al		Open in New Window				
> 🔁 al > 🔁 bl		Сору	Ctrl+C			
> 🔁 bl	Ē	Paste	Ctrl+V	L .		
> 🚰 co	×	Delete	Delete	L .		
> 👝 Fa		Source	>	L .		
> 🔁 lw		Move		L .		
> 🔁 m		Rename	F2	L .		
> 👉 m > 🖧 o: > 🖧 pl > 🏠 tu > 🤓 ut	<u>&amp;</u> 2 (±)	Erase Download Migrate to GD Project Import				
> 🔁 w	4	Export		L .		
> 🇭 az 📴 Outline An outline i	8	Build Project Clean Project Refresh Close Project	F5			Zasks ⊒ Console ⊠ CDT Build Console [MSDk
		Close Unrelated Project				
		Build Configurations Build Targets Index	>			
	0	Run As	>			
	*	Debug As	>	GD	1 GD	
		Profile As	>	C	2 Local C	/C++ Application
		Team	>		Debug C	onfigurations
		Compare With	>		2 todg of	



# AN154 GD32VW553 Quick Development Guide

Double-click GDB General Debugging to open the interface shown in *Figure 4-15. MSDK* <u>debug configuration</u>, the c/c++ application has automatically selected msdk\MSDK.elf, you can browse to select different configurations corresponding to elf file, for example, msdk\_ffd generated elf The file is located at GD32VW55x\_RELEASE\MSDK\projects\eclipse\msdk\ msdk\_ffd\MSDK.elf.

#### Figure 4-15. MSDK debug configuration

GD Debug Configurations				-	
Create, manage, and run configur	ations				Ú.
<ul> <li>Image: Second Se</li></ul>	Name: MSDK msdk	WD	Variables	Search Project	Browse
	MSDK Build (if required) bet	Select Automatically	○ Disable auto bui <u>Configure Workspa</u>		Browse
< >> Filter matched 12 of 12 items				Revert	Apply
?				Debug	Close

#### Start debugging

Start board integrated GD-Link, it is recommended to use openocd for debugging, openocd is in the path: EmbeddedBuilder\_v1.3.13.23294\Tools\OpenOCD, and the GD-Link script for GD32VW55x integrated internally. Switch the GDB Server to openOCD as shown in *Figure* <u>4-16. MSDK Debugging Configuration Interface with openocd</u>, and specify the config options as shown in the figure, click Debug, and wait for the completion of image downloading to debug. The debugging interface is shown in *Figure* <u>4-17. MSDK debug interface</u>.



Figure 4-16. MSDK	Debugging	Configuration	Interface	with openocd

GD Debug Configurations		
Create, manage, and run configurations		<b>N</b>
Image: Second	GDB Server other options  -f D\tools\EmbeddedBuilder_v1.3.13.23294\Tools\OpenOCD\xpack-openocd-0.11.0-3\scripts\target Config Options -f D\tools\EmbeddedBuilder_v1.3.13.23294\Tools\OpenOCD\xpack-openocd-0.11.0-3\scripts\target -f D\tools\EmbeddedBuilder_v1.3.13.23294\target -f D\tools\EmbeddedBuilder_v	irowse
Filter matched 12 of 12 items	Revert	Apply
?	Debug	Close

Figure 4-17. MSDK debug interface

	Studio V8.10c (64-bit) - Licensed to Gigadevice (Stopped) X
File Edit View Search	Navigate Project Build Debug Target Tools Window Help Q X ■ 10 × 0 × 0 × 1 × 1 ₩ + 1 + 1 + 1 + 1 + 1 + 1 + 1 + 1 + 1
Project Explorer 🔲 🧟 🗙	alicloud_entry.c main.c 😫 🗸 🗶 Locals 🗔 🖾 🗙
tot msdk	[ ] MSDK_msdk ▼
Solution 'MSDK' Solution 'MSDK' Project ALICLOUD' Project MSDK' ■ alicloud 15 files ■ coap 2 files ■ coap 2 files ■ discloud 15 fi	<pre>59</pre>
<ul> <li>be_app 23 files</li> <li>ble_profile 16 files</li> <li>Fatfs 5 files</li> <li>files</li> <li>libcoap-4.3.4 27 fil</li> </ul>	<pre>74 #if-defined-CONFIG_BASECMD+  .defined-CONFIG_RF_TEST_SUPPORT</pre>
<ul> <li>Iwip 55 files</li> <li>mbedtls 12 files</li> <li>os 206 files</li> <li>plf 49 files</li> </ul>	 Coutput Coutput Show: Target ▼ ↓ Qutput 
interpretation	J-Link: Flash download: Bank 0 @     Function     Call Address     File     ABI, RV321     ABI, RV321     File       J-Link: Flash download: Total: 26     b int main()     0x0802A6F8     main.c: 67     ACPU, RV321     File       J-Link: Flash download: Program s     0x0800A288     start.S: 334     0x0802a6F8     0x0802a6F8       Download successful     Memory map 'after startup complet     ACPU, RV321     Call Address     Call Address
	<ul> <li></li> <li></li></ul>



# 5. SEGGER Embedded Studio IDE project

This chapter introduces how to compile and debug the SDK under SEGGER Embedded Studio IDE.

# 5.1. Open projects

Open MBL project

Open the directory: GD32VW55x\_RELEASE\MBL\project\segger, double click MBL.emProject to open the MBL SES project. The opened project is shown in *Figure 5-1. MBL SES Project Project Interface*.

-	mbedded Studio V8.	-		- Gigad					-		×
File Edit View	Search Navigate	Project Build	l Debug	Target To	ols Windov	v Help					
MBL	- 🕆 🖌	i 🖅 🖓	🏼 ! 🗸	⊿ 🖑	→ ■ ←	প্য 🕻 🗉	ç⊒ →⊒   ξ⊒   1	~ 🖓 🔊	•		
Project Explorer			× Empty	Dock							×
🕄 Common	- 🛛 💼	😭 😯 🚸 🖞	è								
Project Items		Code Data-	RO								
Solution 'MBL'											
⊳ 📄 mainboot ⊳ 📄 platform (											
	12 11103						Dock Here				
			Outpu	•							×
						11	-		l	M	
			Show:	Transcript	•	. * *	Output 🔻				¢
			-								
					0.5		11 1) <b>O</b> B 15 OF	1910	(h) (h) (h)		20
					U Disc	onnected (J·	Link) 🥥 Built OK	INS	(No editor)	16	.28

Figure 5-1. MBL SES Project Project Interface

Open MSDK project

Open the directory: GD32VW55x\_RELEASE\MSDK\projects\segger, double-click on the MSDK.emProject to open the MSDK project, open the project as <u>Figure 5-2. MSDK SES</u> <u>Project Interface</u> shown.



# AN154 GD32VW553 Quick Development Guide

#### Figure 5-2. MSDK SES Project Interface

0													
MSDK - SEGGER Embedded Studio V	′8.10c (64-l	oit) - Licens	ed to	-	Gigade	vice							×
File Edit View Search Navigate	Project	Build	Debug	Target	Tools	Window	v He	lp					
MSDK 🔻 🐮 🍟	<b>*</b>	.: ¢: ()		⊿ 🖑		■ ←	প্ত (	I ÇI	÷≣	ξΞ 00	,   ⊅≣	•	
Project Explorer			Empty	Dock									×
🕄 msdk 🔻 🔟 🧰	<b>P</b> 0	</td <td></td>											
Project Items	Code	Data+RO											
Solution 'MSDK'													
<ul> <li>Project 'ALICLOUD'</li> <li>Project 'MSDK'</li> </ul>							D	ock Here					
Project 'WPA_SUPPLICANT'								UCKTIER	-				
			Outpu	ıt								10 M	×
			Show:	Transcri	pt	-	7	*	Output	•			ø

# 5.2. Compilation

Nuclei Toolchain configuration

Please put the directory of Toolchain downloaded in subsection <u>2.3Toolchain download</u> nuclei\_riscv\_newlibc\_prebuilt\_win32\_2022.04\gcc\bin into the path of environment variable of windows.

SES build tool configuration

SES compiles the GD32VW55x project using the riscv32-none-elf toolchain by default. In order to better support the extended instruction set of riscv, it needs to be compiled using the nuclei toolchain: riscv-nuclei-elf. The compilation tool can be obtained by contacting sales or FAE. The details of the toolchain are shown in *Figure 5-3. nuclei toolchain content*. Where the Segger\_IDE is the SES IDE installation directory.

Figure 5-3. nuclei toolchain content

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

#### Compile the MBL project

Right-click the project and click build to guild MBL, as shown in *Figure 5-4. Compiling the MBL project*; or click Build->Build MBL in the menu bar.



#### Figure 5-4. Compiling the MBL project

MBL		<ul> <li>✓</li> <li>✓</li></ul>	<u>_</u> %_ ↓	
Project Explorer				
🕄 Common		- 🖸 🗎	🗗 🖸	
Project Items			Code	Data+RO
Solution 'MBL'	P	Options		Alt+Return
🛛 🖉 Project 'MBL				
🔋 🕞 📄 mainboo	<b>8</b>	Build		
b in platform		Rebuild		
		Clean		

The compilation result is as shown in Figure 5-5. MBL compilation result.

Figure 5-5. MBL compilation result

Output						
Show: Transcript						
1> Compiling 'init_rom_symbol.c'						
4> Compiling 'lib_hook_mbl.c'						
<pre>2&gt; Assembling 'start.S'</pre>						
<pre>3&gt; Compiling 'system_gd32vw55x.c'</pre>						
2> Linking MBL.elf						
<pre>2&gt; Post-Building 'MBL'</pre>						
2> Active code page: 65001						
2> "Not add image header and tailer, goto download!"						
<pre>2&gt; 1 file(s) copied.</pre>						
Build complete						
Build complete						

After the compilation is complete, the script MBL\project\mbl\_afterbuild.bat will be automatically called to generate mbl.bin and copied to the directory \scripts\images.

■ Compile the MSDK project

Right-click Project 'MSDK' and click Build, as shown in Figure 5-6. Compile MSDK project.

#### Figure 5-6. Compile MSDK project

4	1		1		
	•	°a <b>°a '</b> i	<b>1</b>	ta ta (la	1
Project Explorer					Empt
🕄 msdk		• 🛛 🗀	🗗 🗘		
Project Items			Code	Data+RO	
Solution 'MSDK'					
Project 'ALICLOU	P	Options		Alt+Re	turn
Project 'MSDK'	==	options		Alterio	cum
Project 'WPA_SU	镭	Build			
		Rebuild			
		Clean			
		Export Build			

Configuration selection of MSDK

MSDK configuration switch as shown in *Figure 5-7 MSDK Project Configuration Options*. MSDK SES project only supports msdk and msdk\_ffd; if you need to use the configuration of msdk\_threadx, msdk\_ffd\_threadx and msdk\_azure please use the GD32 EmbeddedBuilder IDE project or wait for subsequent updates.



#### Figure 5-7 MSDK Project Configuration Options

File	Edit	View	Search	Navig	ate	Proje	ct	Build	
	<b>/</b> ISDK		<b>•</b>	) H	*	M	ţ⊒	ţ⊒	(je
Proje	ct Explor	rer					I	3	×
🗘 m	sdk		-		<u> </u>	P (	€ </td <td>b 1</td> <td></td>	b 1	
🛟 m	sdk					Code	. Ir	Data+F	0
🖓 m	sdk_ffd					Coue		Jala+I	
ि <	Edit Con	figuratio	ns>						
P 🛓	Projec	t ALICLU	00						
	Projec	figuratio t ALICLO t <b>MSDK</b>							
			UPPLICAN	IT'					

After selecting the corresponding configuration, right-click the project and click Build, the compilation result is shown in *Figure 5-8 MSDK compilation result*.

Figure 5-8 MSDK compilation result



#### ■ Image generated by SDK

After MSDK is compiled, it will call MSDK\projects\image\_afterbuild.bat to generate imageota.bin and image-all.bin, and copy the generated bin files to \scripts\images, as shown in *Figure 5-9. Images output*.

image-ota.bin is the bin file generated by MSDK project, which can be used for OTA upgrade. image-all.bin is the combination of MBL(mbl.bin) and MSDK(image-ota.bin), the firmware can be used for production, download into flash and run.

#### Figure 5-9. Images output

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

### 5.3. Download firmware

Refer to 1.4 Download interface, copy GD32VW55x\_RELEASE\scripts\images\image-



all.bin to the Gigadevice disc to download it. Or download it by clicking Target->Download MSDK in the menu bar, as shown in *Fogure 5-10 SES IDE image download*.

#### File Edit Navigate Build Debug Window View Search Project Target Tools Help 🍖 Connect J-Link Ctrl+T, C MSDK -🖥 🖥 🐪 涾 ta Da 👔 🚦 🏠 Disconnect Ctrl+T. D Empty Dock 🗞 Reconnect Project Explorer X Ctrl+T, E 🕄 msdk 📜 Attach Debugger Ctrl+T, H - 🖸 💼 😭 € <> Project Items Code Data+RO \$ Reset Ctrl+T. S Solution 'MSDK' ↓ ■ Download MSDK Ctrl+T, L Project 'ALICLOUD' JE Verify MSDK Ctrl+T, V 740.1K 101.0K Project 'WPA\_SUPPLICANT' ⊳ Ctrl+T K Erase All Upload Range... Download File Verify File Start Cycle Counter Pause Cycle Counter 12 Zero Cycle Counter Ctrl+T, Z Switch Project Output Target Connection Properties

#### Fogure 5-10 SES IDE image download

### 5.4. Debugging

Debugging configuration

SES IDE recommends using J-link to debug, and J-link driver version at least V7.92o, this version of J-link driver support GD32VW55x chip.

The project has been configured with Debug information by default, if you need to change it, right-click on the MSDK project, click Options to open the configuration interface, you can modify the Debugger and J-Link under the Debug option, as shown in <u>Figure 5-11. MSDK</u> <u>SES Project Configuration Interface</u>.



#### Figure 5-11. MSDK SES Project Configuration Interface

► 🔱 🗘 msdk	▼ Search Options	
⊿ Code	Option	Value
Assembler		
Build	🔺 🔳 Debugger	
Code Analyzer	<ul> <li>Target Connection</li> </ul>	J-Link inherits
Code Generation	<ul> <li>Target Device [RISC-V]</li> </ul>	GD32VW553HMQ7 inherits
Compiler	Run To Control	Always
Compiler Warning	Run To	main
External Build	<ul> <li>Startup Completion Point</li> </ul>	inherits
File	<ul> <li>Start From Entry Point Symbol</li> </ul>	Yes inherits
Library	<ul> <li>Leave Target Running</li> </ul>	No
· · · · · · · · · · · · · · · · · · ·	<ul> <li>CPU Register File</li> </ul>	\$(StudioDir)/targets/cpu_registers_riscv.xml
Linker	<ul> <li>Register Definition File</li> </ul>	<pre>\$(ProjectDir)/GD32VW553x_Registers.xml modifie</pre>
Preprocessor	<ul> <li>Debug Terminal Log File</li> </ul>	None
Printf/Scanf	<ul> <li>HTML Watch File</li> </ul>	<pre>\$(StudioDir)/html/heap.htm</pre>
Section	<ul> <li>Threads Script File</li> </ul>	None
Source Code	<ul> <li>Thread Maximum</li> </ul>	25
User Build Step	<ul> <li>Working Directory</li> </ul>	\$(ProjectDir)
Debug	<ul> <li>Command Arguments</li> </ul>	<pre>\$(ProjectName)\$(EXE)</pre>
Debugger	<ul> <li>Debug Additional Configurations</li> </ul>	
GDB Server	<ul> <li>Debug Additional Projects</li> </ul>	
	<ul> <li>Debug Project Name</li> </ul>	<pre>\$(ProjectName)_\$(Configuration)</pre>
J-Link	Entry Point Symbol	_start inherits
Loader	<ul> <li>Ignore .debug_aranges Section</li> </ul>	No
Simulator [RISC-V]	Ignore .debug_frame Section	No
Target Script	<ul> <li>ISA Extensions Debug</li> </ul>	None (inherits)
	<ul> <li>Load Additional Projects</li> </ul>	
	<ul> <li>Memory Upload Page Size</li> </ul>	1,024
	<ul> <li>Reserved Member Name</li> </ul>	reserved

#### Start Debugging

Click Debug->GO in the menu bar to debug, click and wait for the image downloading to complete and enter the interface shown in *Figure 5-12. SES IDE Debug Interface*.

Figure 5-12. SES IDE Debug Interface





# 6. FAQ

### 6.1. No image error

Print ERR: No image to boot (ret = -5).

**Reason:** An error occurs during the previous boot of WIFI\_IOT, and the MBL records operation exception of the IMAGE. If another IMAGE is not downloaded or also has a boot exception, this message will be printed. In other words, the MBL believes that there is no valid IMAGE to jump to, and the boot fails.

Solution: Download the MBL again. After that, the IMAGE status will be cleared.

### 6.2. Code running in SRAM

To run programs faster to achieve higher performance, move them to the SRAM.

Open GD32VW55x\_RELEASE\MSDK\plf\riscv\env\gd32vw55x.ld, and find the line ".code\_to\_sram:". The code in the braces runs in the SRAM. To add new content, add it at the end of the code. Refer to existing files for the format, for exemple:

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

It is to put the entire port.c file in the SRAM and run it. For example:

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

It is to put the xTaskIncrementTick () function in tasks.c in the SRAM and run it.



# 7. Revision history

### Table 7-1. Revision history

Revision No.	Description	Date
1.0	Initial release	Nov.24.2023
1.1	Chapter 2 revision	Jan.26.2024
	SES IDE project added, GD32	
1.2	Eclipse IDE updated to GD32	July.17. 2024
	Embedded Builder	



#### 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 have 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.

© 2023 GigaDevice – All rights reserved