

Application Note

Document No.: AN1126

Instructions for Use of G32R501 IDE and Tool Chain

Version: V1.1

© Geehy Semiconductor Co., Ltd.



1. Introduction

In modern embedded system development, choosing appropriate development tools and environments is crucial for successfully achieving the project goals. Regardless of development of new projects or porting of existing projects, developers need to flexibly respond to different development environments and chip architectures. This document aims to provide a detailed guide for developers to help them smoothly configure and use these two popular development environments when developing with the G32R501 MCU.

To make full use of the information in this guide, users shall be familiar with the characteristics of G32R501 series MCU. Users can refer to the following relevant technical documents:

- G32R501 Series User Manual, Datasheet, Programming Manual, and Migration Manual
- Related documents for G32R501 series core, including ARMv8.1-M Architecture Reference Manual, etc.

This document provides a detailed introduction to the steps of configuring projects in MDK-ARM and IAR EW for Arm, covering the entire process from project creation, compilation linking to debugging. By comparing the characteristics and settings of different development environments, readers can better understand how to efficiently develop embedded projects in different development environments.

This document will guide readers to master the skills of developing G32R501 in MDK-ARM and IAR EW for Arm environments, to lay a solid foundation for the successful implementation of the projects.

1.1. Full Name and Abbreviation Description of Terms

The keywords and descriptions used in this document are as shown in Table 1.

	•
Full name in English	English abbreviation
Configurable Static Memory Subsystem	CFGSMS
Software	SW
Hardware	HW
Interface	IF
AHB slave interface	ahbs_if

Table 1 Abbreviation Descriptic

1.2. R5xx SoC Introduction

R5xx SoC is a system on a chip (SoC) based on Cortex-M52 dual cores. This SoC adopts the widely used AMBA 2.0 bus to integrate peripheral IP components: the components that require broad bandwidth are connected to the AMBA AHB bus, while the low-speed components are connected to the AMBA APB bus. DMA cannot access the registers related to FLB.



The architecture of R5xx SoC is as shown in Figure 1.





Note:

- 1) The red lines in the figure represent bus bridge.
- 2) The size of ITCM/DTCM/SRAM is configured using CFGSMS.



Contents

1.	Introduction	1
1.1.	Full Name and Abbreviation Description of Terms	1
1.2.	R5xx SoC Introduction	1
2.	MDK-ARM Development Tool Chain	5
2.1.	Simulator Support	5
2.2.	IDE Version	5
2.3.	Project Operation	5
2.4.	C Language Compatibility	
2.5.	Assembly Compatibility	
2.6.	Linker Script Files	
2.7.	RAM Operation	
3.	IAR EW for Arm Development Tool Chain	19
3.1.	Simulator Support	19
3.2.	IDE Version	19
3.3.	Install the Chip Support	
3.4.	Project Operation	
3.5.	C Language Compatibility	
3.6.	Assembly Compatibility	
3.7.	Linker Script Files	29
3.8.	RAM Operation	
4.	Eclipse	31
4.1.	Emulator Support	
4.2.	IDE Version	
4.3.	LLVM_For_ARM_Toolchain	
4.4.	GDB Service	
4.5.	Project Operations	



6.	Revision	.61
5.5.	Integration with Eclipse	54
5.4.	Command Line Usage	. 53
5.3.	pyocd Installation	49
5.2.	PyOCD Adaptation Modifications	46
5.1.	Background	46
5.	pyocd Adaptation for G32R501	.46
4.9.	RAM Operating	. 45
4.8.	Linker Script Files	. 44
4.7.	Assembly Compatibility	. 43
4.6.	C Language Compatibility	43



2. MDK-ARM Development Tool Chain

During the migration from Txx320F28004x series MCU to G32R501 series MCU, the porting of development tools is an important link. Code Composer Studio (CCStudio) integrated development environment (IDE) and MDK-ARM are both commonly used tools in embedded development.

This chapter will introduce how to migrate the existing CCStudio projects to the MDK-ARM environment, and discuss in detail the compatibility of C language and assembly code, and the configuration of linker script files.

2.1. Simulator Support

Support of simulators by G32R501:

- Geehy-Link (WinUSB), DAP Link (the firmware version is CMSIS-DAP V2 and above)
- J-Link V12 (J-Link V7.94g and above)
- Ulink Pro

2.2. IDE Version

Please make sure to use MDK-ARM V5.40 or higher-version IDE.

Note: The known problem with MDK 5.40/5.41 is that the function jump (press F12, and it will jump to the function definition) and other functions during code editing cannot be used properly.

2.3. Project Operation

Note: The following operations are all performed on MDK-ARM v5.40.

2.3.1. Install Pack support

You can choose any of the following methods for installation:

- 1. Direct installation
 - Look for the file "Geehy.G32R5xx_DFP.x.x.x.pack" under the package directory of SDK.
 - Double-click the file and install in the pop-up installation interface (as shown in Figure 2) according to the instructions.



Figure 2 Click Install in Geehy.G32R5xx_DFP.x.x.x.pack

Version 1.4.0.83			
$C:\ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ $			
This program installs the Software Pack:			
package\Geeh	y.G32R5xx_DFP.1.0	0.0.pack	
Destination Folder			
C:\Keil_v540\Packs			
1			
Keil PackInstaller			
	<< Back	Next >>	Cancel

- 2. Use Pack Installer (as shown in Figure 3)
 - Open MDK-ARM v5.40.
 - In the menu bar, click "Project"->"Manage"->"Pack Installer".
 - This will open a new window, displaying the available support packages.
 - In the new window, select "File"->"Import...".
 - In the file browser, try to find the file "Geehy.G32R5xx_DFP.x.x.x.pack" under the package directory of SDK, and then select.
 - Click "Open" to install.

Figure 3 Import Pack Installer to Geehy.G32R5xx_DFP.x.x.x.pack

Back Installer - C:\Keil_v540\Packs	
File Packs Window Help	
Refresh	
Import b 4 Parks Fyamples	
Import from Folder	\times
Manage Local Repositories	
Settings Solt > SDK > V0.6 > package V 0 在 package 中搜索	Q
Exit ces sea a market sea	0
Ben SPEAK 12/DE01	•
ABOV Semiconductor 34 Devi 3D 对象 个 名称 修改日期 类型	大小
Active-Semi 17 Devi 書 视频	662
Alif Semiconductor 13 Devi Ret	002
Ambiq Micro 16 Devi	
Amiccom 5 Device 2 XA	
8 ^v Analog Devices 14 Devit ↓ 下载	
● APEXMIC 23 Devit ♪ 音乐	
ARM 51 Devi	
BrainChip 1 Device	
e Cmsemicon 95 Devi 本地磁菌 (C:)	
8	
B V Dialog Semiconductor 20 Devi 新加卷 (E:)	
ELAN 1 Device	
B→V FMD 50 Devi ● 网络	
B P FMSH 11 Devi	>
B Geehy 125 Dev	
● GigaDevice 388 Dev 文件名(N): Geehy.G32R5xx_DFP.1.0.0.pack ∨ Software Pack - PACK (*.zi	o;* ~
HDSC 120 Dev	
Himax 2 Device	_



2.3.2. Open the example project

- 1. Start MDK-ARM v5.40.
- 2. Click "Project"->"Open Project" in the menu bar.
- 3. Browse the path of the SDK project file you provide, select the corresponding .uvprojx file, and then click "Open".
- Or after installing MDK-ARM v5.40, directly click the project file ".uvprojx file" to open it.

Note: Please complete the above steps after the Pack support installation is completed; otherwise MDK-ARM will indicate that the chip cannot be found.

2.3.3. Project establishment

- 1. Start MDK-ARM:
 - Open MDK-ARM v5.40.
- 2. Create a new project:
 - Click "Project"->"New uVision Project" in the menu bar.
 - Select the path to save the project, enter the project name, and click "Save".
- 3. Select the MCU:
 - In the pop-up dialog box, select the appropriate G32R501 series MCU model, and click "OK".

2.3.4. File import

- 1. Open the project view:
 - Make sure your new project is already open in the Project window.
- 2. Add an existing file:
 - Click "File Extensions, Books and Environment..."

Figure 4 File Extensions



- Click "Source Group 1" or the folder where you want to add files, and select "Add Files...".
- In the pop-up file browser, find and select the source files to be imported (e.g. .c and .h files), and then click "Add".



Figure 5 Add Source Files

📥 🗟 🗇 🍏				
Manage Project Items				×
Project Items Folders/Extension	s Books Projec	ct Info/Layer		
Project Targets: 🖄 🗙 🛧 🖡	Groups:	🖺 🗙 🗲 🗲	Files:	× 🛧 🗲
g32r501	device driverlib user		system_g32r501.c startup_g32r501.c device.c	
Set as Current Target			Add Files	
	ОК	Cancel		Help

- 3. New file (if needed):
 - To create a new source file, right-click "Source Group 1" and select "Add New Item".
 - Enter the file name, select the file type (e.g. C file or assembly file), and then click "Add".
- 4. Configure the header file path (as shown in Figure 6):
 - Select "Project" ->"Options" in the menu, check "Include Paths" under the "C/C++" tab, and ensure that all necessary library file paths have been added.

Figure 6 Add the Header File Path

I Options for Target 'g32r501'	Folder Setup
Device Target Output Listing User C/C++ (AC6) Asm Linker Debug Utilities	Setup Compiler Include Paths:
Define:G32R501XXARM_ARCH_8M_MAINARM_TARGET_COPROCG32R501CORE_C	
Language / Code Generation Execute only Code Warnings: AC54ke Warnings ▼ Language C: c11 ▼ Optimization: O0 ▼ Turn Warnings into Errors Language C++: c++11 ▼	
Ink-Time Optimization Plain Char is Signed Short enums/wchar Split Load and Store Multiple Read-Only Position Independent use RTTI One ELF Section per Function Read-Write Position Independent No Auto Includes	
Include Paths Mice	OK Cancel

2.3.5. Configure macro definition

In "Project"->"Options", select the "C/C++" tab, add the required macro definition, and configure the corresponding compilation control.



Figure 7 Configure Macro Definition

W Options for Target 'g32r501'
Device Target Output Listing User C/C++ (AC6) Asm Linker Debug Utilities
Preprocessor Symbols
Define: M_MAIN,_ARM_TARGET_COPROC,G32R501,_CORE_CPU0,G32R501_LAUNCHBOARD
Undefine:

2.3.6. Compilation command control

In MDK-ARM, compilation commands can be controlled through the "Misc Controls" under the "C/C++(AC6)" tab in the "Options for Target" window of the project attributes. Under the "C/C++(AC6)" tab, users can set the preprocessor directives, compiler flags, and so on, refer to the compiler command support in Table 3; for more related information, please consult the help documentation in MDK.

Characteristic/Option	Scalar FP half- precision	Scalar FP single- precision	Scalar FP double- precision	MVE integer	MVE FP half- precision	Custom Datapath Extension (CDE) cp0
cortex-m52	Included	Included	Not included	Not included	Included	Not included
cortex-m52+nomve	Included	Not included	Included	Not included	Included	Not included
cortex-m52+nomve.fp+nofp.dp	Included	Not included	Included	Not included	Included	Not included
cortex-m52+nomve+nofp.dp	Not included	Not included	Included	Not included	Included	Not included
cortex-m52+nomve.fp+nofp	Included	Not included	Not included	Not included	Included	Not included
cortex-m52+nopacbti	Included	Included	Not included	Not included	Not included	Not included
cortex-m52+nomve+nopacbti	Included	Not included	Not included	Not included	Not included	Not included
cortex-m52+ nomve.fp+nofp.dp+nopacbti	Included	Not included	Not included	Not included	Not included	Not included
cortex-m52+	Not	Not	Not	Not	Not	Not
nomve+nofp.dp+nopacbti	included	included	included	included	included	included
cortex-m52+nomve.fp+nofp+nopacbti	Not included	Not included	Included	Not included	Not included	Not included

Table 2 Compiler Command



Characteristic/Option	Scalar FP half- precision	Scalar FP single- precision	Scalar FP double- precision	MVE integer	MVE FP half- precision	Custom Datapath Extension (CDE) cp0
aartay mE2 Ladaar0	Not	Not	Not	Not	Not	Included
conex-m52+cdecp0	included	included	included	included	included	Included

Figure 8 Help Documentation in MDK

😵 ARM Development Tools									
ភា គា ៤ ំ ភា កើ									
静康 査找 上一歩 前辺 打印 透项(の)									
□ (1) (1) (1) (1) (1) (1) (1) (1) (1) (1)	Supported architecture feature combinations for specific processors								
E Arm Development Tools	For some Arm processors, the armchang option -necro and the armlink and fromelf optioncro support specific combinations of the architecture features.								
E 💭 Arm Compiler 6 User's Guides									
🗄 👁 Compiler Getting Started Guide	For armclang, the options in the tables assume that you also includetargetwarm-arm-none-eabi.								
Migration and Compatibility Guide									
E U Compiler Reference Guide	Note								
Arm Compiler for Embedded 5.23 documentation	The default and in the line is	to make a Danalamont	Engineer (IDE) with	to fill and a labor sector	A de decide and and the d	a to all the in			
armaig Compiler for Embedded Tools Overview	The default options in an 1/	itegratea Development I	Environment (IDE) might	or different and might overne	te me detauat opuolis for u	e tooknam.			
a armciang Reference	If you are building a validat	ion test provided as part	of the IP deliverables for w	our processor see the Releas	e Notes and makefiles incl	uded in those deliverables for details of th	e command-line ontions l	seing used	
(i) (ii) (iii) (ii	I Jou are changed standard	ion test provided as part	or are in dearerables for j.	ou processor, see un reacus	e rotes and matches at		te continuite and options (rung urea.	
Compiler-specific Keywords and Operators	Combinations o	f architecture f	eatures sunnort	ed for the Cortex	R-M52 process	ar.			
Compiler specific Function, Variable, and Type Attributes	Combinations o	i ai chitectui c i	catures support	cu loi the cortex	S-moz process	51			
Compiler specific Intrinsics	The following M-profile V	ector Extension (MVE)	Floating-noint (FP) and	PACBTI combinations for th	e Cortex®-M52 processo	r are supported:			
🗑 🧰 Compiler-specific Pragmas									
🖂 🔄 Other Compiler-specific Features	Note								
ACLE support	Do not use the armlink on	tion							
Teles fueries	De not use un un minime op	don ope men meng.							
D Valatila unclobia	2		Con	binations of architecture feat	ares supported for the Cor	tex®-M52 processor			
Nalf-presizion floating-point data tunar				A COLOR AND A COLO	M CL. BACDTI				
Half-precision floating-point number format	Scalar FF hall-precision	Scalar FP double-	MVE integer	MVE FF half-precision	M-prome FACBII	armclang option -mepu	armlink option cpu	fromelf option cpu ²	
Half-precision floating-point intrinsics	and single-precision	precision	10000	and single-precision	Extension	100		a second in the	
Library support for Float16 data type	Included	Included	Included	Included	Included	cortex-m52	-	Armv8.1-M.Main.mve	
BFloat16 floating-point number format	Included	Included	Not included	Not included	Included	cortex-m52+nomve	-	Armv8.1-M.Main.mve	
TT instruction intrinsics	Included	Not included	Included	Not included	Included	cortex-m52+nomve.fp+nofp.dp	÷.	Armv8.1-M.Main.mve	
Non-secure function pointer intrinsics	Included	Not included	Not included	Not included	Included	cortex-m52+nomve+nofp.dp	-	Armv8.1-M.Main.mve	
🗉 🧰 Supported architecture features	Not included	Not included	Included	Not included	Included	cortex-m52+nomve.fp+nofp		Armv8.1-M.Main.mve	
Supported architecture feature combinations for spec	Included	Included	Included	Included	Not included	cortex-m52+nopacbt1	-	Armv8.1-M.Main.mve	
armclang Integrated Assembler	Included	Included	Not included	Not included	Not included	cortex-m52+nomve+nopacbt1	-	Armv8.1-M.Main.mve	
🗄 🧰 armclang inline assembler						cortex-		Survey and survey and	
armlink Reference	Included	Not included	Included	Not included	Not included	m52+nomve.fp+nofp.dp+nopacbti	-	Armv8.1-M.Main.mve	
 	Included	Not included	Not included	Not included	Not included	cortex- m52+nomve+nofp.dp+nopacbti	-	Armv8.1-M.Main.mve	
🗉 🧰 armasm Legacy Assembler Reference	Not included	Not included	Included	Not included	Not included	cortex-	-	ArmyS.1-M.Main.mve	
🗉 🤤 Supporting Information Appendix		THE BEILDER	included	. tot actuded	1101 menueu	m52+nomve.fp+nofp+nopachti		Theorem and the second	
Intended audience									
E Revisions	I able notes								
Useful resources	1.00 1.0.20	C DIODTIT			DICOTI				
E Strors and Warnings Reference Guide	* Although the M-pr	onle PACBII Extension	is enabled by default, armo	clang does not automatically	insert PACB II instructions	into user code by default. You must also	use the armclang option	-mbranch-protection to	
E 👽 Libraries and Floating-Point Support User Guide	generate the PACE	r usuucuous. Also, tile ?	vi-prome FACB11 variant	or me rum C doranes is not :	selected by default. For the	re anormanon, see die -moranen-protec	tion and mentally security	-protection.	
E Debugger Adapter User's Guides	2 The second second	A Marca and antian	for the ULE succession will	in francis If is seen in day and	his managed discountly	- Call Ann Roll 1 March MATE instantia			
E Getting Assistance	ind cpu=Armva	m.main.mve opuon	tor me nur processing util	ny fromen is required to ena	ore successful disassembly	or an Armitovo, 1-34 and MVE instructio	us.		
a w ucensing user's Guide	Combinations	farchitacture	onturne cunnant	ad for the Conter	® M55 process				
c	Compinations o	architecture	catures support	cu for the Cortex	S-14135 process	91			
No									



Figure 9 Compilation Command Control

🔣 Options for Target 'g32r501'
Device Target Output Listing User C/C++ (AC6) Asm Linker Debug Utilities
Preprocessor Symbols Define:G32R501XXARM_ARCH_8_1M_MAINARM_TARGET_COPROC,G32R501CORE_ Undefine:
Language / Code Generation Warnings: AC5-like Warnings Language C: c11 Execute-only Code Warnings: AC5-like Warnings Language C: c11 Optimization: -00 Ink-Time Optimization Plain Char is Signed Split Load and Store Multiple Read-Only Position Independent Image C: c++11 Image C: C++11
Includeinclude;
Misc Controls
Compiler control string -xc -std=c11target=arm-arm-none-eabi -mcpu=cortex-m52+pacbti -mfloat-abi=hard -c fno-rtti -funsigned-char -fshort-enums -fshort-wchar v
OK Cancel Defaults Help

2.3.7. Compilation optimization level settings

MDK-ARM provides multiple optimization level settings, which can be adjusted at the global, single-file, and single-function levels:

1. Global optimization level settings: In the "Options for Target" window, select the "C/C++" tab, and then select an appropriate optimization level from the "Optimization" drop-down menu.

🔀 Options for Target 'g32r501'	Х
Device Target Output Listing User C/C++ (AC6) Asm Linker Debug Utilities	
Preprocessor Symbols	
Language / Code Generation Execute-only Code Warnings: AC5-like Warnings Language C: C11	
Optimization: -00 Image: Turn Warnings into Errors Language C++: C++11 Image: C++: C++11 <	

Figure 10 Global Optimization Level Settings



2. Single-file optimization level settings: Right-click a specific source file, select "Options for File...", and then set the optimization level in the "C/C++" tab.

	🔀 Options for File 'adc.c'	×
Image: Construction of the state of th	Properties C/C++ (AC6) Preprocessor Symbols Define: Undefine: Language / Code Generation V Execute-only Code Varnings: <unspecified> Language C: <default> V Turn Warnings into Errors Language C++: <default> V Turn Warnings into Errors Language C++: <default> V Turn Warnings into Errors Split Load a -01 V Plain Char is Signed V Short enums/wchar V Read-Only Position Independent V use RTT1 V One ELF Se -03 Include Oz image size</default></default></default></unspecified>	
gpio.c hrcap.c hrpwm.c i2c.c interrupt.c interrupt.c interrupt.c interrupt.c interrupt.c	Misc Controls -xc -std=c11 -target=arm-arm-none-eabi -mcpu=contex-m52+pacbti -mfloat-abi=hard -c -fmo-rtti -funsigned-char -fshort-enums -fshort-wchar ^ .D_MICROLIB -mlittle-endian -gdwarf4 -00 -ffunction-sections -Wno-packed -Wno-missing-variable- ^ 0K Cancel Defaults Help	

Figure 11 Single-file Optimization Level Settings

 Single-function optimization level settings: Specific compiler instructions can be used for optimization setting before the function, e.g. using __attribute__((optnone)) to declare no optimization or use of __attribute__((optimize("O2"))) for optimization.

2.3.8. Program compilation

1. In the menu bar, click "Project" ->"Build Target" (or directly click the "Build" button on the toolbar, or press the "F7" button).



- Figure 12 Compiler Program
- 2. After the compilation process is completed, check for any errors or warning messages in the output window. If there are errors, make corresponding modifications according to the prompt

2.3.9. Program download

- 1. Select the simulator:
- Click "Project" ->"Options" in the menu bar.
- In the pop-up dialog box, select the "Debug" tab.



- In the "Use" drop-down menu, select the appropriate debugging simulator (e.g. Geehy-Link, J-Link, etc.).
- 2. Configure the download script:
- Open the project options: Select "Options for Target".
- Select the downloading tool: In the "Utilities" tab, select the appropriate downloading tool.
- Load the download script: Click the "Settings" button to load the specific download script file.

Figure 13 Select Download Script

📱 Options for Target 'g32r501'		×
Device Target Output Listing User C/C++ (AC	C6) Asm Linker Debug Utilities	
Configure Flash Menu Command		
Use Target Driver for Flash Programming	✓ Use Debug Driver	
Use Debug Driver	Settings Update Target before Debugging	
Init File: .\r501_key.ini	Edit	

- 3. Download program:
- On the toolbar, click the "Download" button (small arrow icon) or select "Flash"
 ->"Download" from the menu (or press "F8" directly).

Figure 14 Download Program



• Ensure that the target device is connected, wait for the download to be completed, and view the output window to confirm the download status.

2.3.10. Program simulation

2.3.10.1. J-Link emulation

- Open the project options: Select "Options for Target".
- Select the debugger: In the "Debug" tab, choose the J-Link debugger.



- Add the r501_dbg.ini and JLinkSettings.JLinkScript files in the target project folder.
 - The JLinkSettings.JLinkScript file is a C-like scripting language used to customize the operation of the J-Link debugger. The JLinkScript file includes basic syntax, custom operations, API functions, and global constants (variables) from DLLs, and its syntax is similar to that of the C language.
 - The r501_dbg.ini and JLinkSettings.JLinkScript files are located in SDK/device_support/g32r501/common/Jlink/.

601	\rightarrow examples \rightarrow eval \rightarrow led \rightarrow led_ex1_blink	y → project → MDK	~	م ق	在 MDk
	^ 名称	修改日期	类型	大小	
	💣 g32r501dxy_cpu0_cbus_flash.sct	2025/3/31 10:10	Windows Script	9 KB	
	JLinkSettings.JLinkScript	2025/4/11 15:50	JLINKSCRIPT 文件	13 KB]
	🔣 project.uvprojx	2025/4/8 14:57	♦Vision5 Project	23 KB	_
	😹 r501 dbg.ini	2025/4/11 15:50	配置设置	1 KB	
	👼 r501_key.ini	2024/12/17 17:43	配置设置	5 KB	

Figure 15 Add J-Link document

Note: The r501_dbg.ini file is different from the r501_dbg.ini file used with GEEHY LINK emulation; please make sure to distinguish between them when in use.

4. Load the emulation script: In the "Settings" menu, click the "Load" button to load the specific emulation script file.

Options for Target 'g32r501'	:
Device Target Output Listing User C/C++ (A	C6) Asm Linker Debug Utilities
C Use <u>Simulator</u> <u>with restrictions</u> <u>Settings</u> ☐ Limit Speed to Real-Time	Use: U-LINK / J-TRACE Cortex Settings
✓ Load Application at Startup ✓ Run to main() Initialization File:	Load Application at Startup I Run to main() Initialization File:
	Vr501 dha ini

图 16 J-Link emulator and emulation script.

5. Remove the download script: When using the J-Link emulator, no additional download scripts are required. Please delete the setting for the "Init File" in the Utilities download script.



Figure 17 Remove the download script

Options for Target 'g32r501'	×
Device Target Output Listing User C/C++ (AC6) Asm Linker Debug Utilities	
Configure Flash Menu Command	
Use Debug Driver Settings 🔽 Update Target before Debugg	ing
Init File: Edit	

- 6. Startup debugging:
 - After the download is complete, click the "Debug" button to start debugging the program.
 - In debug mode, it can set breakpoints, view variables, perform step-by-step execution, and other operations.

2.3.10.2. Geehy Link emulation

- 1. Open the project options: Select "Options for Target".
- 2. Select the debugger: In the "Debug" tab, choose the appropriate debugger.
- 3. Load the simulation script: In the "Settings" menu, click the "Load" button to load a specific simulation script file.

Figure 18 Select the Simulator and Simulation Script

🕎 Options for Target 'g32r501'	×
Device Target Output Listing User C/C++ (AC6)	Asm Linker Debug Utilities
C Use Simulator <u>with restrictions</u> Settings Limit Speed to Real-Time	ତ Use: CMSIS-DAP ARMv8-M Debugger <mark>→</mark> Settings
Coad Application at Startup CRun to main()	✓ Load Application at Startup ✓ Run to main() Initialization File: ✓ 501 dbo ini
	Derfere Detries Cetting

- 4. Start debugging:
- After downloading, click the "Debug" button to start the debugging the program.
- In debugging mode, such operations as setting breakpoints, viewing variables, and single-step execution can be performed.

2.4. C Language Compatibility

When migrating the tool chains, the compatibility of C language code needs special attention. Different compilers may have differences in file format support, built-in functions, memory



operation, and data type. To ensure that the code can be compiled and run correctly in a new compilation environment, necessary adjustments and optimization shall be made on the existing code.

2.4.1. File format support

Support .c/.h files.

2.4.2. Use of sizeof

Different compilers and architectures may have differences in the size of data type, so special attention shall be paid to the results of the sizeof operator on different platforms. The following is a size comparison of common data types on G32R501 and Txx320F28004x:

Data type	G32R501	Txx320F28004x
char	1	1
short	2	1
int	4	1
long	4	2
long long	8	4
float	4	2
double	8	4
long double	8	4
void*	4	2
int8_t	1	1
uint8_t	1	1
int16_t	2	1
uint16_t	2	1
int32_t	4	2
uint32_t	4	2
int64_t	8	4
uint64_t	8	4

Table 3 Size Comparison of Different Data Types on G32R501 and Txx320F28004x

2.5. Assembly Compatibility

There may be significant differences in instruction set and assembly syntax among different target platforms, so the existing assembly code needs to be rewritten and adapted to ensure correct operation on the new platforms

2.5.1. File format support

AC6 is based on LLVM and Clang technology, mainly using GNU-style assembly syntax. Assembly file formats supported by AC6:



• .s file: GNU-style assembly file format.

Meanwhile, AC6 supports the use of inline assembly in C functions.

2.5.2. Assembly code format requirements

1. Single assembly file: Here is a simple assembly code example, which defines an assembly function add, used to add two integers. The content of the file "add.s" is as follows:

.syntax unified .global add .type add, %function

add:

@ Function entry
@ Parameters: r0 and r1
@ Return value: r0
adds r0, r0, r1
@ Add r0 and r1, store result in r0
bx lr
@ Return to calling function
.end

2. Use inline assembly in C function: The following is an example of using inline assembly in C function, and an inline assembly function add inline is defined to add two integers:

```
// Inline assembly function
static inline int add_inline(int a, int b) {
    int result;
    __asm volatile (
        "adds %0, %1, %2\n"
        : "=r" (result) // Output operand
        : "r" (a), "r" (b) // Input operands
        : "cc" // Clobbered registers
    );
    return result;
}
```

2.6. Linker Script Files

The linker script files are used to define the memory layout and section allocation of the program. In the migration process, it is necessary to use linker script files in the corresponding format according to different target platforms and development environments. The G32R501 uses linker script files in the ".sct" format in the MDK development environment, which comply with Arm Company's specifications.



Differences in linker script files

- File format:
 - The G32R501 uses linker script files in the ".sct" format, which comply with Arm Company's specifications.
 - Txx320F28004x uses the linker script files in ".CMD" format, which comply with the company's specifications.
- Memory layout and section allocation:
 - The ".sct" file of G32R501 arranges memory allocation by defining the load region and execution region.
 - The ".CMD" file of Txx320F28004x arranges memory allocation by defining the Memory section (MEMORY) and section allocation (SECTIONS).

2.7. RAM Operation

The Txx320F28004x compiler supports the Progma syntax, which tells the compiler that if a specific function, target file, or the attributes of a section of code are modified, such as CODE SECTION, a Section is assigned to a certain function, and its usage is as follows:

#pragma CODE_SECTION(funcA, "codeA")

G32R501 can implement the same function using the following statement:

__attribute__((section("xxx")))

Where, xxx represents the SECTION name which a certain function or global variable is assigned. During use, users can refer to the following format:

__attribute__((section("itcm.ramfunc"))) void SysCtl_delay(uint32_t count){}

You only need to specify the attributes when defining functions and variables.

Note: When using "__attribute__((section("itcm.ramfunc")))", a declaration of the "*itcm.ramfunc*" field in the .sct (chained footstep file) is required: .*ANY (itcm.ramfunc)*



3. IAR EW for Arm Development Tool Chain

3.1. Simulator Support

Please refer to Chapter 2.1.

3.2. IDE Version

Please make sure to use IAR EW for Arm 9.60.2 or higher-version IDE.

3.3. Install the Chip Support

Before officially using IAR EW for Arm to develop G32R5 series MCU, please install the chip support package first. The path for chip support

- $is: .. \ utilities \ G32R5xx_AddOn \ G32R5xx_AddOn \ vx.x.x.exe$
- Open G32R5xx_AddOn_vx.x.x.exe with the administrator rights and go to the interface for selecting the path to install the chip support. This path is the installation path for IAR EW for Arm, e.g. the example: D:\iar\ewarm-9.60.2\.

Figure 19 Installation of Chip Support on G32R5xx_AddOn_vx.x.x.exe

Geehy G32R5xx Device AddOn Package to IAR 1.0.0 Set					
Destination Folder					
Select the file directory for installation (IAR installed file directory)					
This AddOn will install into the following product folder. To install to this folder,press 'Install'.To install to a different folder,press 'Browse' and select another folder.					
Destination Folder					
D:\jar\ewarm-9.60.2\ Browse					
Space required: 2.7 MB					
Space available: 193.9 GB					
ttps://www.geehy.com					
< <u>B</u> ack <u>I</u> nstall Cancel					

If the software cannot obtain the IAR EW for Arm installation path on the computer, please manually select it.

2. After selecting the correct path and adding the chip support, open IAR EW for Arm, select "New project", and in the chip selection tab, the G32R5 series MCU list can be seen.



Figure 20 G32R5 Series MCU List

3.4. Project Operation

3.4.1. Open the example project

- 1. Start IAR EW for Arm 9.60.2.
- 2. Click "File" ->"Open Workspace..." in the menu bar.
- 3. Browse the path of the SDK project file you provide, select the corresponding .eww file, and then click "Open".
- Or, after installing IAR EW for Arm 9.60.2, directly click the project file ".eww file" to open it.

Note: Please complete the above steps in Section 3.3 Install Chip Support; otherwise, the IAR EW for Arm will prompt that the chip cannot be found.

3.4.2. Project establishment

- 1. Start IAR EW for Arm:
 - Open IAR EW for Arm 9.60.2.
- 2. Create a new project:
 - Click "Project"->"Create New Project" in the menu bar.
 - Select "Tool chain" as "Arm".
 - Select the "main" under "C" and then click "OK".
 - Select the path to save the project, enter the project name, and click "Save".



Figure 21 Create New Project

Tool chain:	Arm	~	·
Project template	es:		
👘 🖹 Empty	project		^
🚊 💼 asm			
- 🖹 as	m (ARM 32-bit)		
- 🖹 a:	m (Cortex-M)		
🚊 💼 C++			
🛛 🖾 🖹 m	ain		
🚊 💼 C 👘			
- 🖹 📶	ain		
🖳 🖹 sł	ared object (.so)		
🛛 🗠 🖹 DLIB	C (with some runtime s	support for C++)	
🛛 🖻 DLIB	C++ (C++14 with exce	ptions and RTTI)	
Extern	ally built executable		~
Description			
C project using	default tool settings in	ncluding an empty main.c	: file.

- 3. Select the MCU:
 - Right-click the project name and select "Options...".
 - Select "Device" under "Target" in "General Options".
 - Select the appropriate G32R501 series MCU model, and click "OK".

Figure 22 Select MCU

Options for node "project"

Category:							
General Options	1						
Static Analysis							
Runtime Checking	II						
C/C++ Compiler		Library Configura	tion	Library	Options 1	Librar	y Options 2
Assembler		Target	32	- bit	64-bit		Output
Output Converter		langer	32	DIL	04-010		Output
Custom Build		Processor varian	+				
Linker		Processor varian	·				
Build Actions		⊖ Core	O Core Cortex-M52 V				
Debugger							—
Simulator		Device	Geeh	y G32R501	IVY	F	1
CADI		Openice		, 			
CMSIS DAP			None				
E2/E2 Lite		O CMSIS-Pack	None				
GDB Server							
G+LINK							
I-jet		-Execution mode					
J-Link/J-Trace		C as L's					
TI Stellaris		32-bit					
Nu-Link		∩ 64-bit					
PE micro		0.01.01					
ST-LINK							
Third-Darty Driver							



3.4.3. File import

- 1. Open the project view:
 - Make sure your new project is already open in the Project window.
- 2. Add an existing file:
 - Right-click the project name, select "Add", and then select "Add Files".

□ ● project - Deb └──	Options		
	Make		
	Compile		
	Rebuild All		
	Clean		
	C-STAT Static Analysis	>	
	Stop Build		
	Add	>	Add Files
	Remove		Add Group

Figure 23 File Extensions

- In the pop-up file browser, find and select the source files to be imported (e.g. .c and .h files), and then click "Add".
- 3. New file (if needed):
 - If a new source file needs to be created, click the "File" on the toolbar and select "New File".
 - A file will appear in the IDE; press "CTRL+S" to save the file.
 - Enter the file name, select the file type (e.g. C file or assembly file), and then refer to the steps for "Adding files".

3.4.4. Configure header file path and macro definition

- 1. Right-click the project name and select "Options...".
- 2. Select "Preprocessor" under the "C/C++Compiler" tab.



C/C++ Compiler		Languag	ie 1	Languag	e 2	Code	Optimizati	ons	Output
Assembler		List	Pret	processor	Dia	anostics	Encodings	Evt	a Options
Output Converter		LIST				gnosues	Encounigs	LAU	a options
Custom Build					P.				
Linker			e star	idard includ	e aire	ctories			
Build Actions		Addition	al incl	ude directo	ries: (one per lin	e)		
Debugger		Addition			1031 (one per in	0,		
CADI		\$PROJ_		\\include					<u> </u>
CMSIS DAP		\$PROJ_			.\\de	vice_suppo	ort\g32r501\cc	ommo	r
E2/E2 Lite		\$PROJ_			.\\de	vice_suppo	ort\g32r501\cc	ommo	r
GDB Server		\$PKOJ_DIK\$\\\\\\\\\\device_support\g32r501\commor							
G+LINK		SPROJ_	лкэ\	\ \ \ \ \ \ \.	.\\ae	vice_suppo	ort/g32r501/cc	ommo	r 🗸
I-jet		Preinclue	de						
J-Link/J-Trace									
TI Stellaris									
Nu-Link		D.C.J		1	P	_			
PE micro		Defined	symb	ols: (one pe	r line)				
ST-LINK		_G32R	501XX	(<u>∧</u> ⊔P	reprocessor o	utput	to file
Third-Party Driver		_ARM_	ARCH	_8_1M_MAII	N		Preserve cor	nmen	ts
II MORATE I	×	ARM_	TARG	ET_COPROC		~	Generate #li	ne dir	ectives
		G32R	501			×			
		_			_	*			

Figure 24 Configure Header File Path and Macro Definition

- 3. Add all necessary library file paths in "Additional include directories: (one per line)".
- 4. In Defined Symbols: The macro definition required for (one per line).

3.4.5. Compilation optimization level settings

IAR EW for Arm provides multiple optimization level settings, which can be adjusted at the global, single-file, and single-function levels:

1. Global optimization level settings: Open the "C/C++ Compiler" tab and then select the appropriate optimization level in "Optimizations".

Options for node "proje	ct"						
options for hode proje	o.						
Category: General Options A Static Analysis Runtime Checking	Multi-file	e Compilation card Unused Publics				Fac	tory Settings
C/C++ Compiler	List	Preprocessor	Dia	gnostics	Encodings	Extr	a Options
Assembler Output Converter Custom Build	Langu	age 1 Languag	je 2	Code	Optimizati	ons	Output
Linker Build Actions Debugger Simulator	○ N	one		Commor Loop un	n subexpressio rolling	n elim	ination
CADI CMSIS DAP E2/E2 Lite	Ом	⊖ Medium ⊖ High			otion sed alias analy stering	sis	
GDB Server G+LINK I-jet 1+ ink/1-Trace	E	alanced v	nte	Instructio Vectoriza	on scheduling ation		

Figure 25 Global Optimization Level Settings

2. Single-file optimization level settings: Right-click a specific source file, select "Options...", check "Override inherited settings", and then select the appropriate optimization level in



"Optimizations".

Figure 26 Single-file Optimization Level Settings

Exclude from build								
Category: Static Analysis Runtime Checking C/C++ Compiler		inherited	d settings				Fact	tory Setting:
Custom Build	List	Prep	processor	Dia	gnostics	Encodings	Extra	a Option
	Language 1 Language 2			e 2	Code	Optimizati	ons	Output
	Level No O No Me O Hig Bz	one w edium gh alanced	I v	E [[[[[[[[[[Common Coop un Function Code ma Type-ba Static clu Instructio Vectoriza	n subexpressic rolling inlining otion sed alias analy istering on scheduling ation	on elim /sis	ination

3. Single-function optimization level settings: Specific compiler instructions can be used before the function to set the optimization level, e.g. declaring no optimization using Pragma("optimize=none").

3.4.6. Program compilation

1. In the menu bar, click "Project" ->"Make" (or directly click the "Make" button on the toolbar, or press the "F7" button).



2. After the compilation process is completed, check for any errors or warning messages in the output window. If there are errors, make corresponding modifications according to the prompt

3.4.7. Program simulation and download

- 1. Select the simulator:
 - Right-click the project name and select "Options...".
 - In the pop-up dialog box, select the "Debugger" tab.
 - In the "Driver" drop-down menu under the "Setup" tab, select the appropriate

Figure 27 Compiler Program



debugging simulator ((e.g. Geehy-Link (CMSIS DAP), J-Link, etc.).

Category:						Factory Settings
General Options	^					
Runtime Checking						
C/C++ Compiler		Authenticat	ion	Extra	a Options	Plugins
Assembler						i lugino
Output Converter		Setup	Downlo	bad	Images	Multicore
Custom Build						
Linker		Driver		Run	to	
Build Actions						
Debugger		CMSIS DAP	~	ma	in	
Simulator		Simulator				
CADI		CADI				
CMSIS DAP		CMSIS DAP				
E2/E2 Lite		E2/E2 Lite				
GDB Server		GDB Server				
G+LINK		G+UNK	_			
I-jet		List	-			
J-Link/J-Trace		I Cale/I Tasas				•••
TI Stellaris		J-Link/J-Trace				
Nu-Link		TI Stellaris				
PE micro		INU-LINK				
ST-LINK		PE micro				
Third-Party Driver		ST-LINK				
TI MSP-FET	Υ.	Third-Party Drive	er d	lebugger\@	ieehy\G32R501VE	.ddt
		TI MSP-FET				
		TI XDS				

Figure 28 Select the Simulator

2. Configure simulation instructions:

As the G32R5 series MCU supports DCS encryption, corresponding instructions need to be configured for normal simulation.

- Right-click the project name and select "Options...".
- In the pop-up dialog box, select the "Debugger" tab.
- Check "Use command line options" under the "Extra Options" tab, and add instructions to "Command line options"
 - "--macro_param _DCS_ZONE1_CSM0=0xFFFFFFF, setting the DCS key.
 - "--macro_param _SET_PC_BOOTADDR=0x08000000", setting the boot address

C/C++ Compiler	Setup	Download Images		Images	Multicore			
Assembler Output Converter	Authentica	Authentication		tra Options	Plugins			
Custom Build Linker Build Actions Debugger	<u> U</u> se command line options <u>C</u> ommand line options: (one per							
Simulator CADI CMSIS DAP E2/E2 Lite GDB Server G+LINK I-jet J-Link/J-Trace TI Stellaris Nu-Link PE micro	macro_para macro_para macro_para macro_para macro_para macro_para macro_para	am _DCS_Z(am _DCS_Z(am _DCS_Z(am _DCS_Z(am _DCS_Z(am _DCS_Z(am _DCS_Z(am _DCS_Z(am _SET_PC	DNE1_CSM DNE1_CSM DNE1_CSM DNE1_CSM DNE2_CSM DNE2_CSM DNE2_CSM DNE2_CSM DNE2_CSM DNE2_CSM	0=0xFFFFFFF 1=0xFFFFFFF 2=0xFFFFFFFF 3=0xFFFFFFF 0=0xFFFFFFF 1=0xFFFEDFFF 2=0xFFFFFFF 3=0xFFFFFFFF DR=0x08000000	^			

Figure 29 Configure Simulation Instructions

3. Erase and download programs:



- On the toolbar, click "Project" and select "Download" from the menu
 - "Erase memory": Erases the chip Flash.
 - "Download active application": download the program of this project to the chip.
 - "Download file...": Download other programs to the chip
- 4. Program simulation:
 - Click the "Download and Debug" button on the toolbar or press "Ctrl+D" to start the simulation.

Figure 30 IAR Download and Debug



- In debugging mode, such operations as setting breakpoints, viewing variables, and single-step execution can be performed.
- 5. Problems in Debugging
 - During debugging, the Flash content in the Memory window does not update.

Reason: The AccType of the Flash corresponding Memory region in the ddf (device description file) is set to "R", which means the debugger has read-only access to the Flash and cannot modify its content. Therefore, the debugger will not update the values of the corresponding Memory region during the debugging process.

Figure 31 ddf file (Excerpt)

[Memory]						
;;	Name	AdrSpace	StartAdr	EndAdr	АссТуре	Width
Memory =	Boot_ROM	Memory	0x10000000	0x1001FFFF	R	
Memory =	Secure_ROM	Memory	0x10020000	0x1002FFFF	R	
Memory =	ITCM_Flash	Memory	0x00100000	0x0019FFFF	R	
Memory =	BusMatrix_Flash	Memory	0x08000000	0x0809FFFF	R	
Memory =	CPU0_ITCM	Memory	0x00000000	0x0000BFFF	RW	
Memory =	CPU1_ITCM	Memory	0x00000000	0x00001FFF	RW	
Memory =	CPU0_DTCM	Memory	0x20000000	0x20003FFF	RW	

Solution:

1) Enable "Disable Debugger Cache"



Figure 32 Disable Debugger Cache



2) Change the Cache Type of the Flash corresponding Memory region to RAM

Click on Memory Configuration to enter the Memory Configuration window and modify the Cache Type.

ictory ranges									
ielected dev	9 60 20 arrive and in	Project Options:	2000E01D.M.7.44						
	fellowing default me	Jebuggei (deeniy (332H301DX1X7.00						
specilies the	Tollowing detault the	moly ranges.							
Zone	Name		Edit Memory Range	•					
Memory	CPU0_ITCM		- Memoru range						
Momory	CPU1_LICM		Zone:			OK			
Memory	BusMatrix Fl	lash	Memory	\sim		Cancel			
Memory	Boot ROM		Start address:	End address:		Gangor			
Memory	Secure ROM		0x800'0000	0x809'ffff					
Memory	CPU0_DTCM		640 khutes						
	COUR DITCH						-	*	
			Cache type:		Extra attributes				
ed ranges –			• RAM		Access width (8	,16,32,64)			
his is the me	mory configuration th	iat will be used. Yo	○ ROM/Flash		width				
Zone	Start	End	O Uncached/SFR					^	New
Memory	0×00000000	0x0000BFFF						ſ	Edit
Memory	0×00000000	0x00001FFF							Earc.
Memory	0x00100000	0x0019FFFF	ROM/Flash	640 kbytes					Remove
Memory	0x08000000	0x0809FFFF	ROM/Flash	640 kbytes 1					Use Factor
Memory	0×10000000	0x1001FFFF	ROM/Flash	128 kbytes					
Memory	0x10020000	0x1002FFFF	ROM/Flash	64 kbytes					
Memory	0×20000000	0x20003FFF	RAM	16 kbytes				~	

Figure 33 Cache type

3) Modify the AccType of the Flash corresponding Memory region in the ddf file to RW.



Figure 34 ddf file AccType

[Memory]						
11	Name	AdrSpace	StartAdr	EndAdr	АссТуре	Width
Memory =	Boot_ROM	Memory	0x10000000	0x1001FFFF	R	
Memory =	Secure_ROM	Memory	0x10020000	0x1002FFFF	R	
Memory =	ITCM_Flash	Memory	0x00100000	0x0019FFFF	R	
Memory =	BusMatrix_Flash	Memory	0x08000000	0x0809FFFF	RW	
Memory =	CPU0_ITCM	Memory	0x00000000	0x0000BFFF	RW	
Memory =	CPU1_ITCM	Memory	0x00000000	0x00001FFF	RW	
Memory =	CPU0_DTCM	Memory	0x20000000	0x20003FFF	RW	
Memory =	CPU1_DTCM	Memory	0x20000000	0x20001FFF	RW	
Memory =	SRAM1	Memory	0x20100000	0x2011FFFF	RW	
Memory =	SRAM2	Memory	0x20200000	0x2021FFFF	RW	
Memory =	SRAM3	Memory	0x20300000	0x2031FFFF	RW	
Memory =	APB0_Peripherals	Memory	0x40000000	0x4000FFFF	W	
Memory =	APB1_Peripherals	Memory	0x40010000	0x4001FFFF	W	
Memory =	APB2_Peripherals	Memory	0x40020000	0x4002FFFF	W	
Memory =	DEMUX0_AHB	Memory	0x40030000	0x4003FFFF	W	
Memory =	APB3_Peripherals	Memory	0x50000000	0x5000FFFF	W	
Memory =	DEMUX1_AHB0	Memory	0x50010000	0x50027FFF	W	
Memory =	DEMUX1_AHB1	Memory	0x60000000	0x6FFFFFFF	W	
Memory =	APB4_Peripherals	Memory	0x50100000	0x50103FFF	W	
Memory =	APB5_Peripherals	Memory	0x50104000	0x5010FFFF	W	
Memory =	DEMUX2_AHB	Memory	0x50110000	0x5011FFFF	W	
Memory =	PPB	Memory	0xE0000000	0xE00FFFFF	W	

More detail: 调试时 Memory 窗口中 Flash 内容不更新

3.5. C Language Compatibility

Please refer to Chapter 2 "C Language Compatibility".

3.6. Assembly Compatibility

There may be significant differences in instruction set and assembly syntax among different target platforms, so the existing assembly code needs to be rewritten and adapted to ensure correct operation on the new platforms

3.6.1. File format support

IAR Embedded Workbench uses the specific IAR assembly syntax. Assembly file formats supported:

• .s file: IAR-style assembly file format.

Meanwhile, it supports the use of inline assembly in C functions.

3.6.2. Assembly code format requirements

• Single assembly file: Here is a simple assembly code example, which defines an assembly function add, used to add two integers. The content of the file "add.s" is as follows:

SECTION .text:CODE ; Define code section PUBLIC add ; Declare global symbol



add:

```
; Function entry
; Parameters: r0 and r1
; Return value: r0
ADD r0, r0, r1 ; Add r0 and r1, store result in r0
BX Ir ; Return to calling function
```

END

• Use inline assembly in C function: The following is an example of using inline assembly in C function, and an inline assembly function add_inline is defined to add two integers:

```
// Inline assembly function
static inline int add_inline(int a, int b) {
    int result;
    __asm volatile (
        "adds %0, %1, %2\n"
        : "=r" (result) // Output operand
        : "r" (a), "r" (b) // Input operands
        : "cc" // Clobbered registers
    );
    return result;
}
```

3.7. Linker Script Files

The linker script files are used to define the memory layout and section allocation of the program. In the migration process, it is necessary to use linker script files in the corresponding format according to different target platforms and development environments. G32R501 uses the linker script files in the ".icf" format in the IAR EW for Arm development environment, which comply with IAR Company's specifications.

Differences in linker script files

- File format:
 - The G32R501 uses linker script files in the ".icf" format, which comply with IAR Company's specifications.
 - Txx320F28004x uses the linker script files in ".CMD" format, which comply with the company's specifications.
- Memory layout and section allocation:
 - The ".icf" files of G32R501 allocate memory attributes by defining different "regions"



and their attributes.

 The ".CMD" file of Txx320F28004x arranges memory allocation by defining the Memory section (MEMORY) and section allocation (SECTIONS).

3.8. RAM Operation

Please refer to Chapter 2.7.



4. Eclipse

4.1. Emulator Support

- Geehy-Link (WinUSB), DAP Link (firmware version CMSIS-DAP V2 or above)
- J-Link V12 (J-Link V7.94g or above)

4.2. IDE Version

Ensure the use of Eclipse 4.35 or a newer version of the IDE.

4.3. LLVM_For_ARM_Toolchain

- 1. Download the LLVM_For_ARM_Toolchain compiler
- From the LLVM_For_ARM_Toolchain repository: <u>https://github.com/ARM-software/LLVM-embedded-toolchain-for-Arm</u>, download the LLVM-ET-Arm-19.1.1-Windows-x86_64 compiler.
- 2. Add Environment Variables
- Extract the downloaded compressed package (example extraction path: D:\desktop\clang\utilities\LLVM-ET-Arm-19.1.1-Windows-x86_64)
- Add the "bin" folder from the extracted directory to the system environment variable.

Figure 35 Add the system environment variable

:\Program Files (x86)\Pico Technology\PicoScope6\
:\xpack-windows-build-tools-4.3.0-1-win32-x64\xpack-window
:\GCC\10 2021.10\bin
:\Program Files\Git\cmd
:\Program Files\TortoiseSVN\bin
:\Program Files\MATLAB\R2022a\runtime\win64
:\Program Files\MATLAB\R2022a\bin
:\desktop\clang\utilities\LLVM-ET-Arm-19.1.1-Windows-x86_6
SystemRoot%\System32
system courses and a

4.4. GDB Service

It is recommended to use the arm-none-eabi-gdb.exe provided by Arm. The version of armnone-eabi-gdb.exe used in the example is 14.2.

4.5. Project Operations

4.5.1. Opening the Example Project

1. Run Eclipse 4.35.



 Click "File" -> "Import..." -> "General" -> "Existing Project into Workspace" in the menu bar.

C Import		\times
Select Create new projects from an archive file or directory.	Ľ	1
Select an import wizard:		
type filter text		
 ✓ ➢ General ♥ Archive File ♥ Existing Projects into Workspace ₽ File System Projects from Folder or Archive > ➢ C/C++ > ➢ Git > ➢ Install > ➢ Comph > ➢ RPM > ➢ RPM > ➢ Tubebug 		~
? < Back Next > 3 Finish	Canc	el

Figure 36 Import project

3. Click "Select root directory", navigate to the path of the SDK project file you provided, select the corresponding project folder, and then click "Finish".

Note: Please complete the above steps after finishing the LLVM compilation configuration in section 4.3.

4.5.2. Project Creation

- 1. Run Eclipse:
- Open Eclipse 4.35.
- 2. Create a New Project:
- Under "File->New," select to create a new C/C++ Project, and choose C Managed Build.
- Enter the project name, configure the project type, and it is recommended to place the project under the Project directory. Configure the toolchain, selecting Arm Cross GCC as the toolchain.



Figure 37 Configure project

•	— 🗆 X
C Project Directory with specified name already exists 	
Project name: Demo_project	
Use default location	Browse
Choose file system: default v	Toolchains:
 GNU Autotools Executable Empty Project Hello World ANSI C Project Hello World Arm C Project Hello World RISC-V C Project ADuCM36x C/C++ Project Show project types and toolchains only if the second se	Arm Cross GCC Cross GCC RISC-V Cross GCC
? < Back	Next > Finish Cancel

 If the Arm Toolchains in the Eclipse IDE are correctly configured, the path will be automatically selected here. If not properly configured, you can click Browse to select the corresponding absolute path.

Figure 38 Set Arm To	olchains Path
----------------------	---------------

GNU Arm Cross To Select the toolchain	olchain and configure	e path			Ď
Toolchain name: xPa	ck GNU Arm I	Embedded GCC	(arm-none-ea	bi-gcc)	~
Toolchain path: C:\a	ırm\gcc\bin				Browse
?		< Back	Next >	Finish	Cancel

• Click "Finish", completed setting Project.

4.5.3. File Import

1. Right-click on the project folder to create a virtual folder.



Figure 39 New file

Project Explorer 🗡	<	🖻 😫 🏹 🍃 🕴 🗖 🗖			
✓ ⅔ Demo_project (i	n ecli	ipse_demo)			
> 🗊 Includes		New	>	B	Project
		Go Into		¢	C/C++ Project
		Open in New Window		Ľ	File
		Show In Alt+Shift+	w >	C ^{\$}	File from Template
		Show in Local Terminal	>	Ċ	Folder
	ľ	Copy Ctrl+	۲C	¢	Create a new folder resource

- 2. Add files:
 - Select the folder, right-click and choose the Import option to directly import files.

🖨 Import — 🗆	×
Select	
Import resources from the local file system into an existing project.	
Select an import wizard:	
type filter text	
🖌 🗁 General	
🕼 Archive File	
Existing Projects into Workspace	- U
🛱 File System	- U
Preferences	- H.
Projects from Folder or Archive	
> > > C/C++	
> 🔁 Git	
California Californ	ancel

Figure 40 Import file

• When importing files, select "File System" as the import method. In the pop-up file path selection dialog, choose the path of the files to be imported, then check the files you need to import.



Figure 41 Select file

File system Import resources from the local file system.	
From directory: s\eval\empty_projects\empty_driverlib_project\source v	rowse
Source	
Filter Types Select All Into folder: Demo_project/user	ro <u>w</u> se
Options Overwrite existing resources without warning Create top-level folder << <u>A</u> dvanced	
 ✓ Create links in workspace ✓ Create virtual folders ✓ Create link locations relative to: PROJECT_LOC 	
? < Back Next > Finish C	Cancel

- 3. New file (If needed):
 - If you need to create a new source file, click "File" -> "New" and select the source file to create.
 - - Choose the folder and filename for the new source file.\

4.5.4. Compilation Configuration

- 1. Right-click on the project name and select "Properties".
- 2. Select the "External builder" configuration:



Figure 42 External builder

Properties for Demo_pro	ject	- 🗆 X
type filter text	C/C++ Build	⟨¬ ▼ ¬ ⇒ %
 > Resource Builders > C/C++ Build > C/C++ General 	Configuration: G32R501 [Active]	✓ Manage Configurations
Linux Tools Path MCU Project Natures	🗐 Builder Settings 💿 Behavior 🤣 Refresh Policy	
Project References Run/Debug Settings > Task Repository	Builder Builder type: External builder ⊡ Use default build command	~
Task Tags	Build command: \${cross_make}	Variables

- 3. Under the "C/C++ Build" tab, select "Settings" -> "Tool Settings" -> "Target Processor"
 - Set all dropdown configuration items to default, then manually add the command line: "--target=armv81m-none-eabi -mcpu=cortex-m52 -mfpu=none -fno-exceptions -fno-rtti -lcrt0-semihost –lsemihost" (These parameters can be modified according to the actual chip specifications.)

e filter text Settings		↓ ↓
Active Builders C/C++ Build Build Variables Environment ISON Compilation Dat	e] III Devices 1 Container Settings 🌶	Manage Configurations Build Steps 😤 Build Artifact 📷 Binary Parsers 🕢 Error Parsers
Logging <u>Settings</u> Tool Chain Editor C/C++ General Git Linux Tools Path MCU Project References Run/Debug Settings Task Tags Validation WikiText	Arm family (-mcpu) Architecture (-march) Instruction set Thumb interwork (- Endianness Float ABI FPU Type Unaligned access TrustZone (-mcmse Adrch64 family Feature crc Feature crp Feature rp Feature fp Feature simd Code model Strict align (-mstric Other target flags	Toolchain default Toolchain default Toolchain default mthumb-interwork() Toolchain default Toolchain default <
>		

Figure 43 Target Processor

 Under the "C/C++ Build" tab, select "Settings" -> "Tool Settings" -> "GNU Arm Cross C Compiler" -> "Preprocessor"



Figure 44 Add Macro Definitions

Properties for led_ex1_blin	ky					3	×
type filter text	Settings			Ģ	• • •	> +	8
> Resource Builders	🛞 Tool Settings 🛞 Toolchains 🔳 Devices 🚺	🕽 Container Settings 🎤 Build Steps 🙅 Build Artifact 🗟 Binary Parsers 🥹 Error Parsers					^
✓ C/C++ Build Build Variables	Arget Processor Optimization	Do not search system directories (-nostdinc) Preprocess only (-E)					
Environment JSON Compilation Dat	👺 Warnings 👺 Debugging	Defined symbols (-D)	 i 		상 🐇	<u>k</u>	
Logging Settings Tool Chain Editor	S GNU Arm Cross Assembler Preprocessor And Includes	ARM_ARCH_BM_MAINARCH_BM_MAINARM_ARCH_BM_MAINARM_TARGET_COPROC					
> C/C++ General Git	🖉 Warnings 🎥 Miscellaneous	CORE_CPU0 G32R501_EVAL					
> MCU Project Natures	Soft Arm Cross C Compiler Preprocessor Includes						
Project References	Optimization						

 Under the "C/C++ Build" tab, select "Settings" -> "Tool Settings" -> "GNU Arm Cross C Compiler" -> "Includes".



Figure 45 Configure head file path

 Under the "C/C++ Build" tab, select "Settings" -> "Tool Settings" -> "GNU Arm Cross C Linker" -> "General".



Figure 46 Add link file



 Under the "C/C++ Build" tab, select "Settings" -> "Tool Settings" -> "GNU Arm Cross C Linker" -> "Libraries".

Properties for Fix32math_sa	ample		- 🗆 X
type filter text	Settings		<> ▼ ⇒ ▼ §
type filter text Sett > Resource Builders > C(-++Build Build Variables Environment JSON Compilation Dat JSON Compilation Dat Image: Compilation Dat Settings Tool Chain Editor > C/C++ General Image: Compilation Dat Git Linux Tools Path > MCU Project Natures Project References Run/Debug Settings > Task Repository Task Rags > Validation WikiTest	Configuration: G32R501 [Active] Tool Settings Toolchains Devices Target Processor Optimization Warnings Debugging Debugging Settings Preprocessor Preprocessor Optimized Location Debugging Configuration Debugging Debugging Debugging	Container Settings A Build Steps 2	المعاملة الم المعاملة المعاملة الم المعاملة المعاملة الم
	 ➢ Maccos ➢ Miscellaneous ➢ GNU Arm Cross C Compiler ➢ Preprocessor ➢ Includes ➢ Optimization ➢ Warnings ➢ Warnings 		
	✓ S GNU Arm Cross C Linker Ø General	Group libraries (-Wl,start-group, Library search path (-L)	,end-group) 🔕 📦 🗞 중 욧
	[避 Libraries]	*\$(ProjDirPath)////lib*	

Figure 47 Add libraries

8. Configure "ToolChain".

type filter text > Resource Builders ~ C/C++ Build	Settings	
> Resource Builders V C/C++ Build		
Build Variables Environment JSON Compilation Dat Logging Settings Tool Chain Editor > C/C++ General Git Linux Tools Path > MCU Project Natures Project References Run/Debug Settings > Task Repository Task Tags > Validation WikiText	Configuration: G32R501 [Active]	Manage Configurations

Figure 48 Configure ToolChain

- 1) Use the built-in clang compiler from llvm.
- 2) Built-in tools from the llvm toolchain.
- 3) Use the built-in make compiler from xpack.



- 9. Configure "Build Step":
 - Under the "C/C++ Build" tab, select "Settings" -> "Build Step" -> "Post-build steps"
 -> "Command".
 - Add the command: `llvm-objcopy -O binary "\${ProjName}.elf" "\${ProjName}.bin"; llvm-objdump -D "\${ProjName}.elf" > "\${ProjName}.dump"`.

Properties for led_ex1_blin	- 🗆	×
type filter text	Settings 🗇 👻	÷ -
 > Resource Builders ✓ C/C++ Build Build Variables 	Configuration: G32R501 [Active] V Manage Configuration	ns
Environment JSON Compilation Dat Logging Settings Tool Chain Editor	 Tool Settings Toolchains Devices Container Settings Build Steps Pre-build steps Command: 	
 C/C++ General Git Linux Tools Path MCU 	Description:	-
Project Natures Project References Run/Debug Settings > Task Repository Task Tags > Validation	Post-build steps Command: llvm-objcopy -O binary *\$(ProjName).elf* *\$(ProjName).bin*; llvm-objdump -D *\$(ProjName v Description:	-

Figure 49 Build Step

4.5.5. Compilation Optimization Level Settings

Eclipse provides multiple optimization level settings that can be adjusted at the global, single-file, and single-function levels:

 Global Optimization Level Setting: Under the "C/C++ Build" tab, select "Settings" -> "Tool Settings" -> "Optimization" -> "Optimization Level".

Properties for led_ex1_blin	ky		- D X
type filter text	Settings		
 > Resource Builders C/C++ Build Build Variables Environment JSON Compilation Dat 	Configuration: G32R501 [Active]	3 Container Settings 🎤 B	✓ Manage Configurations Build Steps ⊕ Build Artifact >>2
Logging Settings I ool Chain Editor > C/C++ General Git Linux Tools Path > MCU	 Target Processor Optimization Warnings Debugging SONU Arm Cross Assembler Preprocessor Includes 	Optimization Level Message length (-fme 'char' is signed (-fsign Function sections (-ffu Data sections (-fdata- No common unitialize	None (-O0) ssage-length=0) ned-char) inction-sections) sections) d (-fno-common)
Project Natures	🖄 Warnings	Do not inline functions	s (-fno-inline-functions)

Figure 50

2. Single-Function Optimization Level Setting: Specific compiler directives can be used before



functions to set optimization levels, such as using "_Pragma("optimize=none")" to declare no optimization.

4.5.6. Program Compilation

1. Click on **Project**, then select **Build Project** to compile the current project.

Note: Using **Build Project** only compiles the current project, while **Build All** compiles all projects in the current workspace.

	Open Project Close Project	
010	Build All	Ctrl+B
	Build Configurations	>
	Build Project	
	Build Working Set	>
	Clean	
	Build Automatically	
	Build Targets	>
	C/C++ Index	>
	Prepare Changelog	Ctrl+Alt+P
	Properties	

Figure 51 Program Compilation

Note: Before compiling, be sure to save the current project. Otherwise, the compilation may be based on the last saved version rather than the latest modifications. To ensure compilation accuracy, perform a **Clean** operation on the project after making changes, then proceed with compilation. After compilation is complete, corresponding.elf, .hex, and .bin files will be generated.

2. Wait for the compilation process to complete and check the output window for any error or warning messages. If errors are present, make the necessary corrections based on the prompts.

4.5.7. Program Simulation and Download

4.5.7.1. J-Link Simulation

- 1. Open the Debug Configuration Interface
 - In the project, click **Run** in the menu bar, then select **Debug Configurations** from the pop-up options to enter the Debug configuration interface.



Figure 52 Debug configuration interface

: 📸 🕶 🔚 🕼 🥸 🕶 🐔 🕶 🔜 : 🞺 😓 : 🖶 : 🝬 💵		Resume	
Project Explorer X		Suspend	
> 🚰 Demo_project (in eclipse_demo)		Terminate	
> 🔐 Fix32math_sample (in Eclipse) [g32r501_v0.6 lijian]	1-1	Disconnect	
✓ ⅔ led_ex1_blinky (in Eclipse) [g32r501_v0.6 lijian]	7.	Step Into	
> 🐉 Binaries	P	Step Over	
> 🔊 Includes	_P2	Step Return	
> 🔁 device	=>]	Run to Line	
> 🔁 driverlib	ষ্ট	Use Step Filters	Shift+F5
> 🔁 G32R501	0	Run Last Launched	Ctrl+F11
V 🔁 user	杰	Debug Last Launched	F11
> 🔄 lod ov1 bliplovs	õ	Profile Last Launched	
main c	₫≡		
⇒ g32r501dxy cpu0 cbus flash.ld	_	Profile History	>
	₿≣	Profile As	>
		Profile Configurations	
		Run History	>
	0	Run As	>
	-	Run Configurations	
		Debug History	>
	*	Debug As	>
		Debug Configurations	
		Breakpoint Types	>

 In the new window, select "GDB SEGGER J-Link Debugging", right-click, and choose "New Configuration" to create a new simulation configuration.

Figure 53 New simulation J-Link configuration



- 2. Configure the Main Tab
 - Name the current simulation configuration at the top.
 - Click **Browse...** to select the project corresponding to the current simulation configuration.
 - Choose the corresponding simulation ELF file (e.g., `G32R501\led_ex1_blinky.elf`). Relative paths (relative to the project file) are used here, but absolute paths are also supported.



Figure 54 Configure Main

Name:	led_ex1_blinky G32R501	01	
Ma	ain 💖 Debugger 🕨 Sta	tartup 💱 Source 🔲 Common 🛃 SVD Path	
Projec	:t:		
led_e	x1_blinky		Browse
C/C+·	+ Application:		
G32R	501\led_ex1_blinky.elf		
		Variables	Search Project Browse

- 3. Configure the Debugger Tab
 - Select the corresponding simulation chip (e.g., G32R501).
 - Choose the GDB service—it is recommended to use the Arm-provided arm-none-eabigdb.exe.

net: Fix32math_sample G32R501 Main Debugger Startup Devolution Common C	Name: Fix32math_sample G32R501 Image: Main Image: Startup Image: Startup Image: J-Link GDB Server Setup Image: Connect to running target Executable path: S(jlink_path)/s(jlink_gdbserver) Browse Variables Actual executable: C/Program Files/SEGGER/JLink/GDBServerCLexe Browse Variables Image: Image: Supported device name: Supported device name: Supported device name: Image: Image: Image: Image: Supported device name: Supported device name: Connection: Image: Image: Image: Supported device name: Supported device name: Image: Image: Image: Image: Supported device name: Supported device name: Image: Image: Image: Image: Supported device name: Supported device name: Image: Image: Image: Image: Supported device name: Supported device name: Image: Image: Image: Image: Supported device name: Image: SWO port: 2332 Image: Image: Image: <td< th=""><th></th><th></th><th></th><th></th><th></th><th></th><th></th><th></th><th></th><th></th><th></th><th></th></td<>												
Main Debugger Startup Startup Startup Startup Connect to running target -Link GDB server Setup Start the J-Link GDB server locally Connect to running target Start the J-Link GDB server locally Browse Variables Actual executable path: \$(jlink_path)/\$(jlink_gdbserver) Browse Variables Actual executable: C/Program Files/SEGGER/JLink_V796j/JLinkGDBServerCLexe Variables Variables Device name: G32R501 Supported device names Supported device names Endianness: ● Little Big Supported device names Connection: ● USB IP (USB serial or IP name/address) Supported device names Initial speed: ○ Auto ○ Adaptive @ Fixed 1000 kHz GDB port: 2331 SWO oprt: 2332 Verify downloads Initialize registers on start Telnet port: 2333 Initial speed: © Silent Browse Device Other options: -singlerun -strict -timeout 0 -nogui // Allocate console for semihosting and SWO Pariation of the GDB server Allocate console for semihosting and SWO SWO	Main Startup Source Common Start he J-link GDB Server Setup Start the J-Link GDB server locally Connect to running target Executable path: \$(jlink_path)/\$(jlink_gdbserver) Browse Variables Actual executable: C:/Program Files/SEGGER/JLink_V796j/JLinkGDBServerCLexe Browse Variables Interface: Itel Big Supported device name Supported device name Initial speed: O Auto O Adaptive Fixed 1000 kHz GDB port: 2331 SWO port: 2332 Initialize registers on sta Telnet port: 2333 Initialize console for the GDB server Allocate console for semihosting and SWO GDB Client Setup Executable name: arm-none-eabi-gdb Browse Variables.	ame: Fix32math_sa	mple G32R5(01									
Link GDB Server Setup Start the J-Link GDB server locally Connect to running target Corperative the global or workspace preferences pages or the project properties page) Device names: Start the OB g Connection: Start the OB g Connection: Start the OB gerver Start GDB port: 2331 SWO port: 2332 SWO port: 2333 SWO port: 2334 SWO port: 2335 SWO port: 2336 SWO port:	J-Link GDB Server Setup Start the J-Link GDB server locally Start the J-Link GDB server locally Start the J-Link GDB server locally Connect to running target Executable path: \$(jiink_path)/\$(jiink_gdbserver) Actual executable: C/Program Files/SEGGER/JLink_V796j/JLinkGDBServerCLexe (to change it use the global or workspace preferences pages or the project properties page) Device name: G32R501 Supported device nam Endianness: GLttle Big Connection: GUSB IP (USB serial or IP name/address) Interface: GSWD OJTAG Initial speed: Auto OAdaptive @ Fixed 1000 kHz GDB port: 2331 SWO port: 2332 Verify downloads / Initialize registers on sta Telnet port: 2333 Cuber options: -singlerun -strict -timeout 0 -nogui Allocate console for semihosting and SWO GDB Client Setup Executable name: arm-none-eabi-gdb Actual executable: arm-none-eabi-gdb	📄 Main 🕸 Debugg	ger 🕨 Start	up 🦤 Source	Commo	on 🔀 SVD Par	th						
Start the J-Link GDB server locally □ Connect to running target Executable path: \$(jlink_path)/\$(jlink_gdbserver) Browse Variables Actual executable C/Program Files/SEGGER/JLink_V796j/JLinkGDBServerCLexe Image: Connect to running target Image: Connect to running target Device name: G32R501 Supported device names Endianness: Image: Connect ion: BUSB IP (USB serial or IP name/address) Interface: Image: Connect ion: Image: Connect ion: Image: Connect ion: Image: Connect ion: SUDD JTAG Image: Connect ion: Image: Connect io	☑ Start the J-Link GDB server locally □ Connect to running target Executable path: §tjlink_path/\$tjlink_gdbserver} Browse Variables Actual executable: C:/Program Files/SEGGER/JLink_V796j/JLinkGDBServerCLexe Supported device name to change it use the global or workspace preferences pages or the project properties page) Supported device name Endianness: © Little Big Connection: © USB IP (USB serial or IP name/address) Interface: © SWD JTAG Initial speed: Auto Adaptive Fixed SWO port: 2332 ☑ Verify downloads ☑ Initialize registers on sta Telnet port: 2333 ☑ Local host only Silent Log file:	J-Link GDB Server	Setup										
Executable path: \$(jlink_path)/\$(jlink_gdbserver}) Browse Variables Actual executable: C:/Program Files/SEGGER/JLink_V796j/JLinkGDBServerCLexe Image: Comparison of the global or workspace preferences pages or the project properties page) Supported device names Device name: G32R501 Supported device names Endianness: Image: Comparison of the good or workspace preferences pages or the project properties page) Supported device names Endianness: Image: Comparison of the good or workspace preferences pages or the project properties page) Supported device names Connection: Image: Output of the good or workspace preferences pages or the project properties page) Supported device names Interface: Image: Output of the good of the	Executable path: \$(jlink_path)/\$(jlink_gdbserver) Browse Variables Actual executable: C/Program Files/SEGER/JLink_V796j/JLinkGDBServerCLexe Supported device name: Supported device name: Device name: G32R501 Supported device name: Supported device name: Endianness: ILitle Big Connection: USB IP (USB serial or IP name/address) Interface: © SWD JTAG Initial speed: Auto Adaptive Fixed 1000 kHz GDB port: 2331 SWO port: 2332 Verify downloads Initialize registers on sta Telnet port: 2333 ILocal host only Silent Iog file: Browse Other options: -singlerun -strict -timeout 0 -nogui Allocate console for semihosting and SWO SWO GDB Client Setup Executable name: arm-none-eabi-gdb Browse Variables.	Start the J-Link	GDB server	locally				Connect to r	running targe	et			
Actual executable: C:/Program Files/SEGGER/JLink_V796j/JLinkGDBServerCLexe to change it use the global or workspace preferences pages or the project properties page) Device name: G32R501 Supported device names Endianness: ILittle Big Connection: ILITTLE Big Connect	Actual executable: C:/Program Files/SEGER//Link_V796j/ItinkGDBServerCLexe (to change it use the global or workspace preferences pages or the project properties page) Supported device name: Device name: G32R501 Supported device name: Endianness: Little Big Connection: USB IP (USB serial or IP name/address) Interface: SWD JTAG Initial speed: Auto Adaptive Fixed 1000 kHz GDB port: 2331 SWO port: 2332 Verify downloads Initialize registers on sta Telnet port: 2333 Local host only Silent Log file: Prowse. Other options: -singlerun -strict -timeout 0 -nogui Allocate console for the GDB server Allocate console for semihosting and SWO GDB Client Setup Executable name: arm-none-eabi-gdb Actual executable: arm-none-eabi-gdb arm-none-eabi-gdb arm-none-eabi-gdb arm-none-eabi-gdb<	Executable path:	\${jlink_path	}/\${jlink_gdbse	erver}							Browse	Variables
to change it use the global or workspace preferences pages or the project properties page) Device name: [G32R501	Ito change it use the global or workspace preferences pages or the project properties page) Supported device name Device name: G32R501 Supported device name Endianness: Ititle Big Connection: USB IP (USB serial or IP name/address) Interface: SWD JTAG Initial speed: Auto Adaptive Fixed 1000 kHz GDB port: 2331 SWO port: 2332 Verify downloads Initialize registers on statistication of the project of the projec	Actual executable:	C:/Program	1 Files/SEGGER	JLink_V796	ij/JLinkGDBSen	/erCL.exe						
Device name: G32R501 Supported device names Endianness: ● Little ● Big Connection: ● USB □ P (USB serial or IP name/address) Interface: ● SWD ◯ JTAG Initial speed: ○ Auto ○ Adaptive @ Fixed 1000 GDB port: 2331 ✓ Verify downloads ✓ Initialize registers on start Telnet port: 2333 ✓ Verify downloads ✓ Initialize registers on start Conference ✓ Device to nongui ✓ Silent Dther options: -singlerun -strict -timeout 0 -nogui ✓ ✓ Allocate console for the GDB server ✓ Allocate console for semihosting and SWO ✓	Device name: G32R501 Supported device name Endianness: ● Little ● Big Connection: ● USB ○ IP (USB serial or IP name/address) Interface: ● SWD ○ JTAG Initial speed: ○ Auto ○ Adaptive ● Fixed 1000 KHz GDB port: 2331 SWO port: 2332 SWO port: 2332 ○ Verify downloads ○ Initialize registers on sta Telnet port: 2333 ○ Local host only Silent Log file: ● Browse. ● Allocate console for the GDB server Other options: -singlerun -strict -timeout 0 -nogui ○ Allocate console for semihosting and SWO GDB Client Setup ● Browse Variables. Actual executable: arm-none-eabi-gdb ● Browse Variables.		(to change i	t use the <u>glob</u>	al or <u>works</u> p	pace preferenc	es pages or th	e <u>project</u> prope	rties page)				
Endianness: Endianness: Endianness: Endianness: En	Endianness: ● Little ● Big Connection: ● USB ○ IP (USB serial or IP name/address) Interface: ● SWD ○ JTAG Initial speed: ○ Auto ○ Adaptive ● Fixed 1000 KHz GDB port: 2331 SWO port: 2332 ○ Verify downloads ○ Initialize registers on sta Telnet port: 2333 ○ Local host only ○ Silent Log file: ● ● ● Other options: -singlerun -strict -timeout 0 -nogui ● ✓ Allocate console for the GDB server ○ Allocate console for semihosting and SWO GDB Client Setup Executable name: arm-none-eabi-gdb ● ● Actual executable: arm-none-eabi-gdb ●	Device name:	G32R501									Supported	device names
Connection: IVSB IP (USB serial or IP name/address) Interface: IVSD JTAG Initial speed: O Adaptive Initial speed: IVERIA 1000 GDB port: 2331 Verify downloads Initialize registers on start SWO port: 2332 Verify downloads Initialize registers on start Telnet port: 2333 IVERIA 1000 Configure file: IVERIA 1000 IVERIA 1000 Other options: -singlerun -strict -timeout 0 -nogui INFORMER Console for semihosting and SWO IVEN Close to console for the GDB server Allocate console for semihosting and SWO	Connection: • USB IP (USB serial or IP name/address) Interface: • SWD)JTAG Initial speed: ∩ Auto ∩ Adaptive • Fixed 1000 kHz GDB port: 2331 SWO port: 2332 ✓ Verify downloads Initialize registers on state Telnet port: 2333 SUcal host only Slient Log file: Browse. Browse. Other options: -singlerun -strict -timeout 0 -nogui Allocate console for semihosting and SWO GDB Client Setup Executable name: arm-none-eabi-gdb Browse Variables. Actual executable: arm-none-eabi-gdb Browse Variables.	Endianness:	Little	OBig									
Interface: SWD OJTAG Auto Adaptive SFixed 1000 kHz ADD port: 2331 SWO port: 2332 SWO port: 2333 SWO port: 2433 SWO port: 243	Interface: <pre> SWD OJTAG JAG Initial speed:</pre> O Auto O Adaptive @ Fixed 1000 kHz SWO port: 2331 SWO port: 2332 Verify downloads // Initialize registers on sta Telnet port: 2333 Verify downloads // Initialize registers on sta Telnet port: 2333 Verify downloads // Initialize registers on sta Telnet port: 2333 Verify downloads // Initialize registers on sta Telnet port: 2333 Verify downloads // Initialize registers on sta Telnet port: 2333 Verify downloads // Initialize registers on sta Telnet port: 2333 Verify downloads // Initialize registers on sta Telnet port: 2333 Verify downloads // Initialize registers on sta Verify downloads // Initialize registers on sta Telnet port: 2333 Verify downloads // Initialize registers on sta Verify downloads // Initialize registers	Connection:	USB	OIP			(USB serial o	r IP name/addre	ess)				
Initial speed: O Auto O Adaptive @ Fixed 1000 kHz GDB port: 2331 SWO port: 2332 Verify downloads I Initialize registers on start Telnet port: 2333 Verify downloads I Initialize registers on start Clocal host only Silent Log file: Silent Browse Other options: -singlerun -strict -timeout 0 -nogui Allocate console for the GDB server Allocate console for semihosting and SWO	Initial speed: O Auto O Adaptive @ Fixed 1000 kHz GDB port: 2331 SWO port: 2332 Verify downloads I Initialize registers on sta Telnet port: 2333 ILocal host only Silent Log file: Browse. Other options: -singlerun -strict -timeout 0 -nogui Allocate console for the GDB server Allocate console for semihosting and SWO GDB Client Setup Executable name: arm-none-eabi-gdb Actual executable: arm-none-eabi-gdb	Interface:	SWD	⊖ JTAG									
GDB port: 2331 SWO port: 2332 Telnet port: 2333 Image: State of the construction of the	GDB port: 2331 SWO port: 2332 2332 Verify downloads Initialize registers on sta Telnet port: 2333 Uog file: Islent Other options: -singlerun -strict -timeout 0 -nogui Allocate console for the GDB server Allocate console for semihosting and SWO GDB Client Setup Executable name: arm-none-eabi-gdb Actual executable: arm-none-eabi-gdb Image: Strict Strice Strict Strict Strice Strict Strict Strict Strict Strict Strice	Initial speed:	O Auto	○ Adaptive	Fixed	1000 kH:	z						
SWO port: 2332	SWO port: 2332	GDB port:	2331										
Telnet port: 2333 Image: Construction of the CDB server Browse Other options: -singlerun -strict -timeout 0 -nogui Browse Allocate console for the GDB server Allocate console for semihosting and SWO	Telnet port: 2333 Image: Construction of the	SWO port:	2332						6	✓ Verify download	s ⊡ Ir	nitialize regis	ters on start
Log file: Browse Other options: -singlerun -strict -timeout 0 -nogui Allocate console for the GDB server Allocate console for semihosting and SWO	Log file:	Telnet port:	2333						E	Local host only	🗌 <mark>S</mark>	ilent	
Other options: -singlerun -strict -timeout 0 -nogui Incate console for the GDB server Allocate console for semihosting and SWO	Other options: -singlerun -strict -timeout 0 -nogui Image: Allocate console for the GDB server Image: Allocate console for semihosting and SWO GDB Client Setup Executable name: arm-none-eabi-gdb Actual executable: arm-none-eabi-gdb Image: Arm-none-eabi-gdb	Log file:											Browse
Allocate console for the GDB server Allocate console for semihosting and SWO	☑ Allocate console for the GDB server ☑ Allocate console for semihosting and SWO GDB Client Setup Executable name: Executable name: arm-none-eabi-gdb Actual executable: arm-none-eabi-gdb	Other options:	-singlerun ·	-strict -timeout	t 0 -noqui								
	GDB Client Setup Executable name: arm-none-eabi-gdb Variables. Actual executable: arm-none-eabi-gdb	Allocate consol	le for the GDI	B server				Allocate console	for semihos	sting and SWO			
JUB Client Setup	Executable name: arm-none-eabi-gdb Variables. Variables. Variables.	GDB Client Setup											
Executable name: arm-none-eabi-odb Browse Variables	Actual executable: arm-none-eabi-gdb	Executable name:	arm-none-	eabi-odb								Browse	Variables
	Actual executable: anni-none-eabi-gub	Actual everytables		aabi adb									
autual executavie, anti-nivite-eau-yuu		Actual executable:	arm-none-e	saur-gub									

Figure 55 Debugger Tab

- 4. Configure the Startup Tab
 - Add **DCS KEY** (must match the chip-side configuration) in the **Initialization Commands** and **Run/Restart Commands** fields.
 - The content added in both fields should be identical.

set {unsigned int}0x50024020 = 0xFFFFFFF

set {unsigned int}0x50024024 = 0xFFFFFDC

set {unsigned int}0x50024028 = 0xFFFFFFF

set {unsigned int}0x5002402C = 0xFFFFFFF



et {unsigned int}0x500240A0 = 0xFFFFFFF
et {unsigned int}0x500240A4 = 0xFFFEDFFF
et {unsigned int}0x500240A8 = 0xFFFFFFF
et {unsigned int}0x500240AC = 0xFFFFFFF
et \$t0 = *(unsigned int *)0x08000000
et \$sp=\$t0
et \$t1 = *(unsigned int *)0x08000004
et \$pc=\$t1
et \$xpsr=\$xpsr (1<<24)

5. Configure the SVD Tab

Figure 56 Configure SVD path

I	Name: led_ex1_blinky G32R501					
	📄 Main 🕸 Debugger 🕨 Startup 🦃 Source 🔲 Common 🔀 SVD Path					
	SVD file (used by the peripheral registers viewer; may be .svd.zip)					
	File path: D:\desktop\r501pack\Geehy.G32R5xx_DFP.1.0.0\SVD\G32R501.svd	Browse Variables				

6. Finally, click the **Apply** button at the bottom right of the tab to save all configurations.

4.5.7.2. GEEHY LINK Simulation

Use Geehy-Link (WinUSB) to download and debug the project, and use pyocd to download and simulate the project. For details, refer to pyocd 适配 G32R501.

4.6. C Language Compatibility

Refer to Section 2 "C Language Compatibility".

4.7. Assembly Compatibility

The instruction sets and assembly syntax of different target platforms may vary significantly. Existing assembly code may need to be rewritten and adapted to ensure correct operation on new platforms.

4.7.1. File Format Support

Eclipse uses a specific GCC assembly syntax. The assembly syntax of GCC (GNU Assembler, abbreviated as GAS) is consistent with MDK (Keil's ARM assembler, i.e., ARMASM) in terms of core instruction sets, but there are significant differences in pseudo-instructions, syntax formats,



and symbol definitions. Below is a detailed comparison.

• .S files: GCC-style assembly file format.

Additionally, it supports inline assembly within C functions.

4.7.2. Assembly Coding Format Requirements

• Single assembly file: Here is a simple example of assembly code that defines an assembly function `add` to add two integers. The content of the file "add.s" is as follows:

.section .text	<pre>//Define code section</pre>
.global add	// Declare global symbol

add:

```
//Function entry
//Parameters: r0 and r1
//Return value: r0
add r0, r0, r1 // Add r0 and r1, store result in r0
bx lr // Return to calling function
```

 Using inline assembly in C functions: The following is an example of using inline assembly in a C function, defining an inline assembly function add_inline to add two integers:

```
// Inline assembly function
static inline int add_inline(int a, int b) {
    int result;
    asm volatile (
        "ADD %0, %1, %2"
        : "=r" (result)
        : "r" (a), "r" (b)
        :
      );
    return result;
}
```

4.8. Linker Script Files

Linker script files are used to define the memory layout and section allocation of a program. During the migration process, it is necessary to use linker script files in the appropriate format according to the target platform and development environment. The G32R501 uses ".ld" format linker script files under the Eclipse development environment, adhering to Eclipse's



specifications.

Differences in Linker Script Files

- File Format:
 - The G32R501 uses ".ld" format linker script files.
 - The Txx320F28004x uses ".CMD" format linker script files, following its company's specifications.
- Memory Layout and Section Allocation:
 - The ".ld" file of the G32R501 allocates memory attributes by defining different "MEMORY" and its "SECTIONS."
 - The ".CMD" file of the Txx320F28004x arranges memory allocation by defining memory segments (MEMORY) and section allocations (SECTIONS).

4.9. RAM Operating

Refer to Chapter 2.7.



5. pyocd Adaptation for G32R501

5.1. Background

To enable customers to perform operations such as program downloading and debugging for the G32R501 in an open-source environment, the G32R501 series MCUs need to support PyOCD.

5.2. PyOCD Adaptation Modifications

The current PyOCD (<u>https://github.com/pyocd/pyOCD/releases/tag/v0.36.0</u>) release version is 0.36, which does not support the M52 core chips or the G32R501 chip. Source code modifications are required to enable support.

5.2.1. Adding M52 Core Support

- 1. Add M52 core support in PyOCD. The main files to be modified are:
 - pyocd\coresight\component_ids.py

Under class CmpInfo(NamedTuple):, add:

# Designer Factory	Component Class Part	Type Archid	Name	Product
(ARM_CHINA_ MC2', None	ID, CORESIGHT_CLASS, (),	0xD24, 0x31, 0x0a3 ⁻	1) : CmpInfo('MTB',	'Star-
(ARM_CHINA_ MC2', ITM.facto	ID, CORESIGHT_CLASS, (pry),	0xD24, 0x43, 0x1a0 ⁻	1) : CmpInfo('ITM',	'Star-
(ARM_CHINA_ MC2', DWTv2.f	ID, CORESIGHT_CLASS, (actory),	0xD24, 0x00, 0x1a02	2) : CmpInfo('DWT',	'Star-
(ARM_CHINA_ MC2', FPB.fact	ID, CORESIGHT_CLASS, (ory),	0xD24, 0x00, 0x1a03	3) : CmpInfo('BPU',	'Star-
(ARM_CHINA_ MC2', None	ID, CORESIGHT_CLASS, (),	0xD24, 0x14, 0x1a14	4) : CmpInfo('CTI',	'Star-
(ARM_CHINA_ MC2', CortexM_	ID, CORESIGHT_CLASS, (_v8M.factory),	0xD24, 0x00, 0x2a04	4) : CmpInfo('SCS',	'Star-
(ARM_CHINA_ MC2', None	ID, CORESIGHT_CLASS, (),	0xD24, 0x13, 0x4a13	3) : CmpInfo('ETM',	'Star-
(ARM_CHINA_ MC2', TPIU.factory	ID, CORESIGHT_CLASS, (),	0x132, 0x11, 0)	: CmpInfo('TPIU',	'Star-



Figure 57 Modified component_ids.py

94	## Map from (designer, class, part	devtype, archid) to compone	nt name, product name, and fa	actory.
95	COMPONENT_MAP: Dict[Tuple[int, int	Optional[int], Optional[int], int], CmpInfo] = {	
96	<pre># Archid-only entries</pre>			
97	# Designer Component C	lass Part Type Archid	Name Product	Factory
98	(ARM_CHINA_ID, CORESIGHT_CLASS	0xD24, 0x31, 0x0a31) : CmpI	nfo('MTB', 'Star-MC2',	None),
99	(ARM_CHINA_ID, CORESIGHT_CLASS	0xD24, 0x43, 0x1a01) : CmpI	nfo('ITM', 'Star-MC2',	ITM.factory),
100	(ARM_CHINA_ID, CORESIGHT_CLASS	0xD24, 0x00, 0x1a02) : CmpI	nfo('DWT', 'Star-MC2',	DWTv2.factory),
101	(ARM_CHINA_ID, CORESIGHT_CLASS	0xD24, 0x00, 0x1a03) : CmpI	nfo('BPU', 'Star-MC2',	FPB.factory),
102	(ARM_CHINA_ID, CORESIGHT_CLASS	0xD24, 0x14, 0x1a14) : CmpI	nfo('CTI', 'Star-MC2',	None),
103	(ARM_CHINA_ID, CORESIGHT_CLASS	0xD24, 0x00, 0x2a04) : CmpI	nfo('SCS', 'Star-MC2',	CortexM_v8M.factory),
104	(ARM_CHINA_ID, CORESIGHT_CLASS	0xD24, 0x13, 0x4a13) : CmpI	nfo('ETM', 'Star-MC2',	None),
105	(ARM_CHINA_ID, CORESIGHT_CLASS	0x132, 0x11, 0) : CmpI	nfo('TPIU', 'Star-MC2',	TPIU.factory),
106	# Designer Component Class Part	Type Archid Na	me Product	Factory
107	(ARM_ID, CORESIGHT_CLASS, None	None, 0x0a00) : CmpInfo('F	ASv1', None,	None),
108	(ARM_ID, CORESIGHT_CLASS, None	None, 0x1a01) : CmpInfo('I	TMv2', None,	ITM.factory),
109	(ARM_ID, CORESIGHT_CLASS, None	None, 0x1a02) : CmpInfo('I	WTv2', None,	DWTv2.factory),
110	(ARM_ID, CORESIGHT_CLASS, None	None, 0x1a03) : CmpInfo('F	PBv2', None,	FPB.factory),
111	(ARM_ID, CORESIGHT_CLASS, None	None, 0x2a04) : CmpInfo('v	8-M Debug', None,	CortexM_v8M.factory),
112	(ARM TD. CORFSTGHT CLASS. None	None. 0x6a05) · CmnInfo('v	8-R Debug'. None.	None).

- pyocd\coresight\core_ids.py
 - 1) Add the core ID after # CPUID PARTNO values

ARM_China_StarMC2 = 0xD24

2) Add the core name after CORE_TYPE_NAME: Dict[Tuple[int, int], str]

(CPUID_ARM_CHINA, ARM_China_StarMC2): "Star-MC2",

Figure 58 Modiffied core_ids.py

```
39 ARM CortexM55 = 0xD22
40
           ARM CortexM85 = 0xD23
41
             ARM China StarMC1 = 0x132
            ARM China StarMC2 = 0xD24
42
43
44
              # pylint: enable=invalid_name
45
            ## @brief User-friendly names for core types.
46
             CORE TYPE_NAME: Dict[Tuple[int, int], str] = {
47
                                        PE_NAME: Dict[Tuple[int, int], str] = {
  (CPUID_ARM, ARM_SC000):
  (CPUID_ARM, ARM_SC300):
  (CPUID_ARM, ARM_CortexM0):
  (CPUID_ARM, ARM_CortexM1):
  (CPUID_ARM, ARM_CortexM3):
  (CPUID_ARM, ARM_CortexM4):
  (CPUID_ARM, ARM_CortexM7):
  (CPUID_ARM, ARM_CortexM0p):
  (CPUID_ARM, ARM_CortexM3):
  (CPUID_ARM, ARM_CortexM3):
  (CPUID_ARM, ARM_CortexM3):
  (CPUID_ARM, ARM_CortexM3):
  (CPUID_ARM, ARM_CortexM35):
  (CPUID_ARM_CHINA, ARM_CHINA_STARMC1):
  (CPUID_ARM_CHINA, ARM_CHINA_STARMC1):
  (CPUID_ARM_CHINA_ARM_CHINA_STARMC1):
  (CPUID_ARM_CHINA_ARM_CHINA_STARMC1):
  (CPUID_ARM_CHINA_ARM_CHINA_STARMC1):
  (CPUID_ARM_CHINA_ARM_CHINA_STARMC1)

                                                                                                                                                                                 "SecurCore SC000",
48
                                                                                                                                                                              "SecurCore SC300",
49
                                                                                                                                                                              "Cortex-M0",
50
                                                                                                                                                                              "Cortex-M1",
51
                                                                                                                                                                                  "Cortex-M3",
52
                                                                                                                                                                                  "Cortex-M4",
53
                                                                                                                                                                                  "Cortex-M7"
54
                                                                                                                                                                                  "Cortex-M0+",
55
                                                                                                                                                                                  "Cortex-M23",
56
                                                                                                                                                                                  "Cortex-M33",
57
58
                                                                                                                                                                                   "Cortex-M35P",
                                                                                                                                                                                   "Cortex-M55",
59
                                                                                                                                                                                   "Cortex-M85",
60
                                           (CPUID ARM CHINA,
                                                                                                                                                                                  "Star-MC1",
61
                                                                                                           ARM China StarMC1):
                                           (CPUID ARM CHINA, ARM China StarMC2): "Star-MC2",
62
63
```

. .



5.2.2. Add G32R501 Chip Support

- Add the G32R501 download algorithm support file in pyocd\target\builtin: target_G32R501xx.py. This file has been added in SDK/device_support/g32r501/common/pyOCD/target_G32R501xx.py.
- 2. To add G32R501 support, include the following in pyocd\target\builtin__init__.py:

from . import target_G32R501xx

'g32r501dxx': target_G32R501xx.G32R501Dxx,

'g32r501xx': target_G32R501xx.G32R501xx,

Figure 59 Modified __init__.py

138	from	. impor	rt target Ai:	r32F103	xx		32B1ME0,	,
139	from	. impor	t target_G3	2R501xx			32B1MD1,	
140				_				
315		'a	air32f103xb'	: targe	t_Air32F1	03xx.A	ir32F103	3xB,
316		'a	air32f103xc'	: targe	t_Air32F1	03xx.A	ir32F103	3xC,
317		'a	air32f103xp'	: targe	t_Air32F1	03xx.A	ir32F103	}x₽,
318		'a	air32f103xe'	: targe	t_Air32F1	03xx.A	ir32F103	3xE,
319		'a	air32f103xg'	: targe	t Air32F1	03xx.A	ir32F103	3xG,
320		' ç	g32r501xx': †	target_	G32R501xx	.G32R5	01xx,	1
321		' e	g32r501dxx':	target	G32R501x	x.G32R	501Dxx,	J
322		}						
323								

5.2.3. pyocd_user.py

pyocd supports parsing the corresponding script using "--script=<path>" to add custom commands and additional operations before or after connection.

The pyocd_user.py file has been added in SDK/device_support/g32r501/common/pyOCD/target_G32R501xx.py.



5.2.3.1. DCS Key

Due to the features of the G32R501, the chip includes DCS functionality, requiring the transmission of a key to the aforementioned target_G32R501xx.py during the connection process for use during the download operation.

If users need to modify the DCS key, they can directly edit NEW_DECRYPT_KEYS:

```
NEW_DECRYPT_KEYS = [
   (0x50024020, 0xFFFFFFFF,),
   (0x50024024, 0xFFFFFFFDC),
   (0x50024028, 0xFFFFFFFF,),
   (0x5002402C, 0xFFFFFFFF,),
   (0x500240A0, 0xFFFFFFFF,),
   (0x500240A4, 0xFFFFFFFF,),
   (0x500240A8, 0xFFFFFFFF,),
   (0x500240AC, 0xFFFFFFFF,),
```

]

Note: The front part is the write address, and the back part is the DCS KEY content to be written.

5.2.3.2. Connection Phase

The connection phase requires the following steps:

- 1. Perform DCS decryption
- 2. Initialize the CPU

The above operations are implemented in "def did_connect(board) -> None:".

5.3. pyocd Installation

5.3.1. Windows

5.3.1.1. Install Python

pyocd 支持需要在 python 环境下,请至 python 官网(<u>https://www.python.org</u>)下载最新的



python 安装包进行安装。

注意:安装完成后,请确保将 Python 添加到系统的 PATH 环境变量中,以便在命令行中使用。

验证:使用 Win+R 键,输入 CMD,在命令行窗口上输入:python,然后回车。显示已安装的 Python 版本号等内容,并进入 Python 的交互式命令行(REPL)中。如需退出输入 exit()或 quit(),然后按下回车键。

Figure 60 Python install check

C:\Users\apex800691>python Python 3.13.1 (tags/v3.13.1:0671451, Dec 3 2024, 19:06:28) [MSC v.1942 64 bit (AMD64)] on win32 Type "help", "copyright", "credits" or "license" for more information.

5.3.1.2. Install pyocd

pyocd is a component package for Python that supports online installation (online installation is recommended due to various dependency packages that can be resolved during the process).

The installation method is as follows:

1. Install using the pip command:

Figure 61 Install pyocd

:\Users\apex800691>pip install pyocd

Note: After installation, add PyOCD's installation path to the system's PATH environment variable for command-line usage. For example:C:\Users\Geehy\AppData\Local\Programs\Python\Python313\Scripts

2. Verify the installation by pressing **Win + R**, entering **CMD**, and running the following command in the terminal: pyocd -h, this will display the command help.

Figure 62 pyocd install check

```
C:\Users\apex800691>pyocd -h
usage: pyocd [-h] [-V] [--help-options] ...
PyOCD debug tools for Arm Cortex devices
options:
-h, --help show this help message and exit
-V, --version show program's version number and exit
--help-options Display available session options.
```

5.3.2. Ubuntu



5.3.2.1. Installing Python

1. Ubuntu usually comes with Python pre-installed. You can check the Python and pip versions by running the following command in the terminal:

python --version

2. If Python or pip is not installed, use the following command to install them:

sudo apt update

sudo apt install python3 python3-pip

5.3.2.2. python3-venv

Some Ubuntu systems come with externally managed Python3 environments, which prevent direct installation of packages in the global environment using pip. To avoid this issue, you can use Python's built-in venv module to create a virtual environment.

1. Use the following command to install the python3-venv package:

sudo apt install python3-venv

2. Use the following command to create a virtual environment (assuming you name it "venv"):

python3 -m venv venv

- 3. Use the following command to activate the virtual environment:
 - Activate the virtual environment to install packages within it.

source venv/bin/activate

• To deactivate the virtual environment later, use the following command:

deactivate

5.3.2.3. Install pyocd

1. In the activated virtual environment, install pyOCD using pip:

www.geehy.com



pip install pyocd

2. Verify the installation results

pyocd --version

3. Query the installation location of pyocd (for subsequent modification of pyocd source code)

pip show pyocd

5.3.2.4. USB Permissions for pyocd

If pyocd cannot access the debugger under a regular user, consider adding appropriate permissions for the current user. Use udev rules to ensure USB device access without sudo. Follow these steps:

1. Create a new rule file by executing the following command in the terminal to create a new udev rule file (e.g., named 99-pyocd.rules):

sudo nano /etc/udev/rules.d/99-pyocd.rules

2. Add the following content to the file (the Geehy-Link device ID is 314B):

SUBSYSTEM=="usb", ATTR{idVendor}=="314b", MODE="0666"

In the nano editor, press Ctrl + O to save the file, then press **Enter** to confirm. Next, press Ctrl + X to exit the editor.

3. After saving the file, reload the udev rules.

sudo udevadm control --reload-rules

sudo udevadm trigger

4. Verify whether pyocd can correctly recognize Geehy-Link.

pyocd list





(venv) kai@Ubuntu:~\$ pyocd lis	t Unique TD	Taraat
#	Ргореувоаго		Target
Θ	Geehy CMSIS-DAP WinUSB	00350043500000144e544859000258	n/a

5.3.3. Modify Replacement Item Content

To support G32R501, please refer to Chapter 5.2 to modify the downloaded pyocd content.

To verify whether g32r501 support has been successfully added, press Win+R, enter CMD, and in the command prompt window, input: "pyocd list –targets". This will display all supported chips. Check if "g32r501xx" is listed among them.

🛤 命令提示符				—	×
cy8c64xx_cm4	Cypress	cy8c64xx_cm4	builtin		~
cy8c64xx_cm4_ful1_flash	Cypress	cy8c64xx_cm4_ful1_f1ash	builtin		
cy8c64xx_cm4_nosmif	Cypress	cy8c64xx_cm4_nosmif	builtin		
cy8c64xx_cm4_s25hx512t	Cypress	cy8c64xx_cm4_s25hx512t	builtin		
cy8c6xx5	Cypress	CY8C6xx5	builtin		
cy8c6xx7	Cypress	CY8C6xx7	builtin		
cy8c6xx7_nosmif	Cypress	CY8C6xx7_nosmif	builtin		
cy8c6xx7_s25fs512s	Cypress	CY8C6xx7_S25FS512S	builtin		
су8сбхха	Cypress	CY8C6xxA	builtin		
g32r501dxx	Geehy	G32R501Dxx	builtin		
g32r501xx	Geehy	G32R501xx	builtin		
hc32a460xe	HDSC	HC32F460xE	builtin		

Figure 64 Support chips list

5.4. Command Line Usage

pyocd supports operation via the CMD command line. The following steps outline the usage (ensure pyocd has been added to PATH before use):

- 1. Connect the board or debugger to the chip and then to the PC.
- 2. In the working directory, add a `pyocd.yaml` configuration file. This file specifies the default target chip, connection method, and other settings. Users can refer to the official documentation: <u>https://pyocd.io/docs/configuration.html</u>.

For the G32R501, the reference `pyocd.yaml` configuration file is located at `SDK/device_support/g32r501/common/pyOCD/target_G32R501xx.py`.

- 3. In the working directory, add a `pyocd_user.py` file. Refer to Chapter 2.3 for its content.
- 4. Open CMD in the working directory and enter: `pyocd commander`. You can then input relevant commands in the command window to perform corresponding operations.



Figure 65 pyocd commander

#	Probe/Board	Unique ID	Target
0	Geehy CMSIS-DAP WinUSB	00330057500000144e544859000258	n/a
1	Geehy CMSIS-DAP WinUSB	00480051500000054e544859000258	n/a
nter 0105 onne yocd yocd 8000 yocd	the number of the debug p 56 W Invalid coresight con- cted to G32R501Dxx [Halter > erase > rw 0x08000000 0x10 000: ffffffff ffffffff f >	probe or 'q' to quit> 1 mponent, cidr=0x0 [rom_table] d]: 00480051500000054e54485900025 ffffffff ffffffff	

5.5. Integration with Eclipse

Currently, Eclipse+pyocd has version requirements for Eclipse. It is recommended to use Eclipse 202503.

Additionally, it is advised to configure the pyocd path globally in Eclipse (as some computers may encounter issues with Eclipse retrieving the PATH). The steps are as follows:

- 1. Right-click on the "Windows" menu bar to display all configurations.
- 2. Select "Preference" from the displayed configurations.
- 3. In the new window, navigate to the sub-options under the "MCU" option.
- 4. Select the sub-option "Global pyOCD Path."
- 5. On the right side, choose the directory where the corresponding pyocd.exe is located.
- 6. Finally, click "Apply and Close."



Figure 66 Global pyOCD Path Step 1-2



Figure 67 Global pyOCD Path Step 3-6

Preferences					\times
type filter text	Global pyOC	CD Path	¢	• 🔿	₩ 8
> General > C/C++ ChangeLog	Configure the they are used	e location where pyOCD is installed. The values are stored within Eclipse d for all projects in all workspaces.	. Unless	redefin	ned mc
> Docker > Help	After installin configure the	ig pyOCD updates, restart Eclipse for the defaults to be re-evaluated an e new location.	d use th	e Resto	re Def
> Install/Update	Executable:	pyocd.exe	5		
> Language Servers > Library Hover	Folder:	$\label{eq:c:Users} C: Users a pex800691 \\ AppData \\ Local \\ Programs \\ Python \\ Python \\ 313 \\ Scription \\ Python \\ Py$	ots	Browse	ə
Global Arm Toolchains Paths					
Global Build Tools Path Global OpenOCD Path					
Global pyOCD Path Global QEMU Paths					
Global RISC-V Toolchains Path Global SEGGER J-Link Path					
Workspace Arm Toolchains Pa			Rest	tore Def	faults
Workspace Build Tools Path ✓	<				>
? è l' 0		Apply and Close		Cancel	

5.5.1. Single-core Simulation Configuration

After importing the project into Eclipse and ensuring it compiles without errors, follow the steps below to configure the simulation tab.

- 1. Create a New Simulation Configuration
 - 1) Left-click the Debug icon to display the Debug configurations.
 - 2) Select the displayed "Debug Configurations..." option.
 - 3) In the new window, choose "GDB PyOCD Debugging" and right-click.
 - 4) Select "New Configuration" to proceed with the simulation configuration.

Figure 68 New Configuration Step1-2





Figure 69 New Configuration Step 3-4



- 2. Configure the Main Tab
 - 1) Name the current simulation configuration at the top.
 - 2) Select "Browse..." to choose the project corresponding to the current simulation configuration.
 - 3) Select the corresponding simulation ELF file, e.g., `G32R501\led_ex1_blinky.elf`. The example uses a relative path to the project file, but absolute paths are also supported.

Name: led_ex1_blinky G32R501				
📄 Main 🎋 Debugger 🕨 Startup 🧤 Source 🔲 Common 🔀 SVD Path				
Project:				2
led_ex1_blinky				Browse
C/C++ Application:				
G32R501\led_ex1_blinky.elf				
		Variables	Search Project	Browse
Build (if required) before launching				
Build Configuration: Select Automatically				~
○ Enable auto build	◯ Disable auto build			
● Use workspace settings	Configure Workspace Settings			

Figure 70 Configure Main

- 3. Configure the Debugger Tab
 - 1) Select the Geehy-Link emulator to be used. The characters in parentheses represent the emulator's serial number.
 - 2) Choose the corresponding simulation chip, e.g., 'Geehy > G32R501xx (g32r501xx)'.
 - 3) For the reset mode, select "Software (SYSRESETREQ)."
 - 4) For the configuration file, select the previously prepared `pyocd_user.py`. It is recommended to use an absolute path without Chinese characters or spaces.
 - 5) For the GDB service, it is recommended to use the Arm-provided `arm-gnu-toolchain-14.2.rel1\bin\arm-none-eabi-gdb.exe`.



Figure 71

Name: led_ex1_blinky G32R501					
📄 Main 🕸 Debugger 🕨 Startup 🤤 Source 🔲 Common 🗟 SVD Path					
pyOCD Setup					
☑ Start pyOCD locally					
Executable path: C:\Users\apex800691\AppData\Local\Programs\Python\Python313\Scripts\pyocd.ex Browse Variable	es				
Actual executable: C:\Users\apex800691\AppData\Local\Programs\Python\Python313\Scripts\pyocd.exe					
(to change it use the global or workspace preferences pages or the project properties page)					
GDB port: 3333 Allocate console for pyOCD					
Semihosting port: 4444 Allocate console for semihosting					
Debug probe: Geehy CMSIS-DAP WinUSB (00330057500000144e544859000258)	esh				
Default target: Generic > CoreSightTarget (cortex_m) 2					
✓ Override target: Geehy > G32R501xx (g32r501xx)					
Bus speed: 1000000 ~ Hz					
Connect mode: Halt ~					
Reset type: Software (VECTRESET) V 3					
Flash mode: Sector erase \checkmark Smart flash					
☐ Halt at hard fault					
☑ Enable semihosting □ Use GDB syscalls for semihosting					
Other options: script E:\GIT\G32R501\g32r501_v0.6\driverlib\g32r501\examples\eval\\ed\\ed_ex1_blinky\project LEclipse\pyocd_user.py	^				
	\vee				
GDB Client Setup					
Executable name: C:\GCC\10 2021.10\bin\arm-none-eabi-gdb.exe Browse Variabl	C:\GCC\10 2021.10\bin\arm-none-eabi-gdb.exe Browse Variables				
Actual executable: C:\GCC\10 2021.10\bin\arm-none-eabi-qdb.exe	C\GCC\10 2021.10\bin\arm-none-eabi-gdb.exe				

- 4. Configure the Startup Tab
 - 1) Add the DCS KEY in the Initialization Commands and Run/Restart Commands sections (this needs to correspond with pyocd_user.py and the chip side).
 - 2) The content added in both sections should be consistent. Refer to the following:

set {unsigned int}0x50024020 = 0xFFFFFFFF
set {unsigned int}0x50024024 = 0xFFFFFDC
set {unsigned int}0x50024028 = 0xFFFFFFFF
set {unsigned int}0x5002402C = 0xFFFFFFF
set {unsigned int}0x500240A0 = 0xFFFFFFFF
set {unsigned int}0x500240A4 = 0xFFFEDFFF
set {unsigned int}0x500240A8 = 0xFFFFFFFF
set {unsigned int}0x500240AC = 0xFFFFFFF
set \$t0 = *(unsigned int *)0x08000000
set \$sp=\$t0



set \$t1 = *(unsigned int *)0x08000004

set \$pc=\$t1

set \$xpsr=\$xpsr|(1<<24)</pre>

Figure 72 Configure Startup
📄 Main 🕸 Debugger 🕨 Startup 🦃 Source 🔲 Common 🔀 SVD Path
Initialization Commands
set {unsigned int}0x50024020 = 0xFFFFFFF set {unsigned int}0x50024024 = 0xFFFFFFDC set {unsigned int}0x50024028 = 0xFFFFFFFF cet {unsigned int}0x5002402C = 0xEEEEEEE
Load Symbols and Executable
✓ Load symbols
Use project binary: led_ex1_blinky.elf
O Use file: Workspace File System
Symbols offset (hex):
✓ Load executable
Use project binary: led_ex1_blinky.elf
OUse file: Workspace File System
Executable offset (hex):
Runtime Options
Debug in RAM
Run/Restart Commands
Pre-run/Restart reset Type: halt (always executed at Restart)
set {unsigned int}0x50024020 = 0xFFFFFFFF set {unsigned int}0x50024024 = 0xFFFFFFDC set {unsigned int}0x50024028 = 0xFFFFFFFF cet {unsigned int}0x50024026 = 0xEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEE
Set program counter at (hex):
✓ Set breakpoint at: main

5. Finally, click the "Apply" button at the bottom right corner of the tab to apply all configuration items.

5.5.2. Dual-Core Emulation Configuration

For dual-core emulation configuration, please refer to the <u>KAN1128_G32R501 Dual-Core</u> <u>Emulation Guide</u>.

5.5.3. Program Fails to Run After Exiting Emulation in Ubuntu

5.5.3.1. Cause

In Ubuntu, Eclipse terminates the current thread before arm-none-eabi-gdb sends "resuming



core 0 [cortex_m]", preventing the chip from resetting properly.

5.5.3.2. Solution

Refer to the dual-core emulation configuration and start the pyOCD gdbserver solution in the terminal.

Start pyOCD gdbserver in the terminal:

Fiaure	73 F	Run	pvOCD	adbserver
				9

0001289 I Patched target_G32R501xx [bCRYPT KEYS via pyocd_user.py! [pyocd_user] 0001327 I Target type is g32r501xx [board] 0001366 I DP IDR = 0x6ba02477 (v2 rev6) [dap]
0001327 I Target type is g32r501xx [board] 0001366 I DP IDR = 0x6ba02477 (v2 rev6) [dap]
0001366 I DP IDR = 0x6ba02477 (v2 rev6) [dap]
$\frac{1}{1}$
0001302 I AHRADATING IDK - GARATAGONI (AHRAD VICE OF GARAGONI)
0001305 I ALBA DE TIDE - 0X04770001 (ALBA VIO TEVO) [discovery]
0001/01 T AHR.APM Class (V1 POM table #0 0 Avendeffene (designer=a75.0rm (bina part=4d2) [rom table]
1 [Alcondon All a Citada Contract and Contra
0001400 I [1]_00001000.WJ Star.WC2 class=9 designer=a75:Arm China part=424 devtype=00 archid=1302 devtd=0:0:05 [rom_table]
Montal I [] conditional and the construction of the construction o
Wold 41 1 [3] color 000 000 000 10 Star MC2 classes designer answer thing part-day devide-out and contraction devide-out of the frame table of the star of the sta
Wold 44 T [5] color 100 cm star MC2 class-9 docigor = 375:47m (bina part=424 doutypo=13 archid=4a13 douid=0:0.05 [com_table]
0001421 I [6] 2003000:202 classed decision characterization and characterization and device of the control for catteri
0001422 I [7]c000/2000:TI Star.WC2 classes designed article developer develo
0001423 I [8] a 004(600.000 C 100 c 100 c 2000-0 C 2000-0 C 100 c 200 c
0001435 I URL 011 Class 0x1 ROM table #0.0 Aven0ff000 (designer=a75.4rm China part=402) [rom_table]
0001447 I [0]c000000000 CS Star.MC2 classes designered start developed architector projectore
VOINTE I [] COMPANIE (CONTRESS AND CONTRES
(00145) I [2] ee0002000: RPI Star-MC2 class=9 designer=a75:Arm China part=d24 devtype=00 archid=1a03 devid=0:0:08 [com_table]
(001455 I [3]ee0000000001TM Star-MC2 class=9 designer=a75:Arm China part=d24 devtype=d3 archid=1a01 devid=0:0:08 [form table]
0001459 I [5]ee0041000:FTM Star-MC2 class=9 designer=a75:Arm China part=d24 devtype=13 archid=4a13 devid=0:0:08 [form table]
0001464 I [6]<0003000:222 class=9 designer=a75:Arm China part=d24 devtype=16 archid=0a06 devid=0:0:0> [rom_table]
0001467 I [7] <e0042000:cti archid="1a14" china="" class="9" designer="a75:Arm" devid="40800:0:0" devtype="14" part="d24" star-mc2=""> [rom table]</e0042000:cti>
0001470 [[8]<00046000:PMC-100 class=9 designer=43b:Arm part=9ba devtype=55 archid=0a55 devid=145509d6:c105c04:0> [rom table]
0001478 I CPU core #0: Star-MC2 c001, v7.0M architecture [cortex m]
0001478 I Extensions: [DSP, FPU, FPU DP, FPU V5, MPU] [cortex m]
0001478 I FPU present: FPv5-D16-M [cortex m]
0001479 I core is created and initialized. [target G32R501xx]
0001484 I 4 hardware watchpoints [dwt]
0001486 I 8 hardware breakpoints, 1 literal comparators [fpb]
0001493 I 4 hardware watchpoints [dwt]
0001495 I 8 hardware breakpoints, 1 literal comparators [fpb]
r501 connect in did_connect
0001550 I Semihost server started on port 4444 (core 0) [server]
0001569 I GDB server started on port 3333 (core 0) [gdbserver]

Eclipse Debugger Tab Configuration:



Figure 74

	Debug Configurations	×
Create, manage, and run configurations		*
🗄 🖻 🐌 🖹 🗮 🖻 🎖 🗸	Name (led_ext_blinkyC32R301	
type filter text	🗈 Main 🔅 Debugger 🕨 Startup 🦭 Source 🖾 Common 🐉 SVD Path	
Type Filter text CIC/E+- Application CIC/E+- Ap	D Main (P Addoger) (* Statu (P Server, C Common, Y, NOPut) Processing (* Statu (P Server, C C Server, Ser	Browse, Vurlahter,
	Host name or IP address localhost Port number: 3333	<u>Restore defaults</u>
Filter matched 20 of 20 items		Revert Apply
0		Close Debug

Note: The terminal path used to start the pyOCD gdbserver should include the single-core pyocd.yaml file.



6. Revision

Table 4 Document Revision History

Date	Version	Change History
January 2025	1.0	New
April 2025	1.1	Add Chapter 2.3.10.1, 4, 5



Statement

This document is formulated and published by Geehy Semiconductor Co., Ltd. (hereinafter referred to as "Geehy"). The contents in this document are protected by laws and regulations of trademark, copyright and software copyright. Geehy reserves the right to make corrections and modifications to this document at any time. Read this document carefully before using Geehy products. Once you use the Geehy product, it means that you (hereinafter referred to as the "users") have known and accepted all the contents of this document. Users shall use the Geehy product in accordance with relevant laws and regulations and the requirements of this document.

1. Ownership

This document can only be used in connection with the corresponding chip products or software products provided by Geehy. Without the prior permission of Geehy, no unit or individual may copy, transcribe, modify, edit or disseminate all or part of the contents of this document for any reason or in any form.

The "极海" or "Geehy" words or graphics with "®" or "[™]" in this document are trademarks of Geehy. Other product or service names displayed on Geehy products are the property of their respective owners.

2. No Intellectual Property License

Geehy owns all rights, ownership and intellectual property rights involved in this document.

Geehy shall not be deemed to grant the license or right of any intellectual property to users explicitly or implicitly due to the sale or distribution of Geehy products or this document.

If any third party's products, services or intellectual property are involved in this document, it shall not be deemed that Geehy authorizes users to use the aforesaid third party's products, services or intellectual property. Any information regarding the application of the product, Geehy hereby disclaims any and all warranties and liabilities of any kind, including without limitation warranties of non-infringement of intellectual property rights of any third party, unless otherwise agreed in sales order or sales contract.

3. Version Update



Users can obtain the latest document of the corresponding models when ordering Geehy products.

If the contents in this document are inconsistent with Geehy products, the agreement in the sales order or the sales contract shall prevail.

4. Information Reliability

The relevant data in this document are obtained from batch test by Geehy Laboratory or cooperative third-party testing organization. However, clerical errors in correction or errors caused by differences in testing environment may occur inevitably. Therefore, users should understand that Geehy does not bear any responsibility for such errors that may occur in this document. The relevant data in this document are only used to guide users as performance parameter reference and do not constitute Geehy's guarantee for any product performance.

Users shall select appropriate Geehy products according to their own needs, and effectively verify and test the applicability of Geehy products to confirm that Geehy products meet their own needs, corresponding standards, safety or other reliability requirements. If losses are caused to users due to user's failure to fully verify and test Geehy products, Geehy will not bear any responsibility.

5. Legality

USERS SHALL ABIDE BY ALL APPLICABLE LOCAL LAWS AND REGULATIONS WHEN USING THIS DOCUMENT AND THE MATCHING GEEHY PRODUCTS. USERS SHALL UNDERSTAND THAT THE PRODUCTS MAY BE RESTRICTED BY THE EXPORT, RE-EXPORT OR OTHER LAWS OF THE COUNTRIES OF THE PRODUCTS SUPPLIERS, GEEHY, GEEHY DISTRIBUTORS AND USERS. USERS (ON BEHALF OR ITSELF, SUBSIDIARIES AND AFFILIATED ENTERPRISES) SHALL AGREE AND PROMISE TO ABIDE BY ALL APPLICABLE LAWS AND REGULATIONS ON THE EXPORT AND RE-EXPORT OF GEEHY PRODUCTS AND/OR TECHNOLOGIES AND DIRECT PRODUCTS.

6. Disclaimer of Warranty

THIS DOCUMENT IS PROVIDED BY GEEHY "AS IS" AND THERE IS NO WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE, TO THE EXTENT PERMITTED BY APPLICABLE LAW.



GEEHY'S PRODUCTS ARE NOT DESIGNED, AUTHORIZED, OR WARRANTED FOR USE AS CRITICAL COMPONENTS IN MILITARY, LIFE-SUPPORT, POLLUTION CONTROL, OR HAZARDOUS SUBSTANCES MANAGEMENT SYSTEMS, NOR WHERE FAILURE COULD RESULT IN INJURY, DEATH, PROPERTY OR ENVIRONMENTAL DAMAGE.

IF THE PRODUCT IS NOT LABELED AS "AUTOMOTIVE GRADE," IT SHOULD NOT BE CONSIDERED SUITABLE FOR AUTOMOTIVE APPLICATIONS. GEEHY ASSUMES NO LIABILITY FOR THE USE BEYOND ITS SPECIFICATIONS OR GUIDELINES.

THE USER SHOULD ENSURE THAT THE APPLICATION OF THE PRODUCTS COMPLIES WITH ALL RELEVANT STANDARDS, INCLUDING BUT NOT LIMITED TO SAFETY, INFORMATION SECURITY, AND ENVIRONMENTAL REQUIREMENTS. THE USER ASSUMES FULL RESPONSIBILITY FOR THE SELECTION AND USE OF GEEHY PRODUCTS. GEEHY WILL BEAR NO RESPONSIBILITY FOR ANY DISPUTES ARISING FROM THE SUBSEQUENT DESIGN OR USE BY USERS.

7. Limitation of Liability

IN NO EVENT, UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL GEEHY OR ANY OTHER PARTY WHO PROVIDES THE DOCUMENT AND PRODUCTS "AS IS", BE LIABLE FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, DIRECT, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE DOCUMENT AND PRODUCTS (INCLUDING BUT NOT LIMITED TO LOSSES OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY USERS OR THIRD PARTIES). THIS COVERS POTENTIAL DAMAGES TO PERSONAL SAFETY, PROPERTY, OR THE ENVIRONMENT, FOR WHICH GEEHY WILL NOT BE RESPONSIBLE.

8. Scope of Application

The information in this document replaces the information provided in all previous versions of the document.

© 2025 Geehy Semiconductor Co., Ltd. - All Rights Reserved

Geehy Semiconductor Co.,Ltd.