Application Note

Document No.: AN1125

G32R5xx SDK Quick Start Guide

Version: V1.2

1 Introduction

The G32R5xx SDK (Software Development Kit) is a set of development software and documents designed for the G32R5xx series real-time control MCU, intended to reduce software development time. From device drivers and libraries to peripheral examples, the G32R5xx SDK provides a foundation for evaluation and development.

Materials mentioned in this application note can be obtained from Geehy official channels.

To quickly evaluate the G32R5xx SDK, prepare the following environment or items:

- Windows 10/11
- Keil MDK v5.40 or later
- IAR Embedded Workbench for Arm (G32R501: refer to 9.60.2 or later; **G32R502 series requires 9.70.4 or later**)
- Eclipse 4.35 or later
- LLVM-ET-Arm 19.1.1 or later
- Arm GNU Toolchain 14.2 or later
- xpack-windows-build-tools 4.4.1-3 (Windows build helper; install and add to PATH for Eclipse / GNU toolchain builds)
- G32R501 EVAL and/or G32R502 evaluation board

Note: The G32R5xx SDK provides device_support/g32r501 and device_support/g32r502 in the same package (and the corresponding driverlib, libraries paths). First-run steps for G32R501 and G32R502 are in Chapters 10 and 11.

Contents

2 SDK structure

The main hierarchy of the G32R5xx SDK is as follows:

- boards: board hardware design materials
- device_support: device support (including g32r501, g32r502)
- docs: user documents (application note PDFs)
- driverlib: driver library and evaluation-board examples (including g32r501, g32r502)
- examples: cross-module examples
- libraries: DSP / Math / Flash API / SFO and related libraries
- kernel: FreeRTOS and related items
- package: CMSIS-Pack, SVD, and related files
- utilities: debug tools, geehy_tool, IAR Add-On, and related tools

For finer directory details, refer to this AN and the topic-specific AN PDFs.

3 About boards

This directory contains hardware design schematics and related materials for G32R5xx series boards.

Note: The boards content described in this chapter is under G32R5xx_SDK/boards.

4 About device_support

This directory contains device-specific support files (including Arm core-related files), bit-field headers, and device development notes. Devices are separated by subdirectory:

Note: The device_support content described in this chapter is under G32R5xx_SDK/device_support.

- device_support/g32r501/: G32R501 device tree (CMSIS, startup, linker script masters, examples, and so on)
- device_support/g32r502/: G32R502 device tree (CMSIS, startup, linker script masters, examples, and so on)

CMSIS-related directories include headers and startup support for the Arm Cortex-M52 core. Typical files under the Geehy device trees include:

- Device header and register definitions
- startup_g32r501. */ startup_g32r502.* and related startup files
- system_.c/.h and related system initialization files
- Common headers such as device.h under include/
- Linker script directories such as icf/, sct/, and ld/
- examples/: device_support-style examples

Exact file names follow the contents of the current SDK package.

5 About docs

This directory contains G32R5xx SDK user guides and software package documents (PDF).

Note: The docs content described in this chapter is under G32R5xx_SDK/docs.

6 About driverlib

This directory contains device-specific driver libraries and driver-based peripheral examples. Examples of device subdirectories:

Note: The driverlib content described in this chapter is under G32R5xx_SDK/driverlib.

- driverlib/g32r501/
- driverlib/g32r502/

Each device tree typically includes:

- driverlib/: peripheral driver API headers and sources (ADC, CAN/CANFD, GPIO, PWM, UART, Flash, and so on)
- examples/: evaluation-board examples (for example, eval/)

For example documentation, refer to *G32R5xx SDK Examples User Manual* (AN1127).

7 About examples

This directory holds cross-peripheral / application examples. Before formal use:

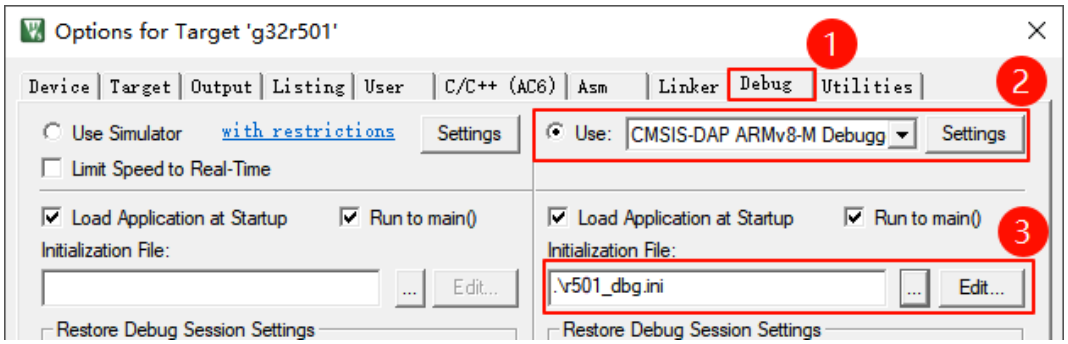
- Read the corresponding example notes and AN1127.

To run an example, prepare as follows:

- Software environment (compiler, Debugger, IDE).
- Hardware environment (Debugger such as GEEHY-LINK / J-Link, and the matching evaluation board).
- When debugging under MDK, build the program once, then select the appropriate Debugger and debug script under **Options for Target** → **Debug**, as shown in Figure 1.

Note: Screenshots below use a G32R501 example (target name often g32r501). For G32R502, select the matching G32R502 device/target, replace g32r501 with g32r502 in paths, and use r502_dbg.ini as the MDK initialization file.

Figure 1 Select the Debugger and debug script



8 About libraries

This directory contains DSP, math, Flash API, calibration, and related libraries. Most libraries are split under g32r501 / g32r502 paths. Common categories:

Note: The libraries content described in this chapter is under G32R5xx_SDK/libraries/.

8.1 Calibration

Contains HRPWM-related calibration libraries (SFO). For usage, refer to AN1134.

8.2 DSP

Typically includes Fixed Point, FPU, VCU, and related sub-libraries, providing FFT, filtering, vector math, CRC, and related capabilities.

8.3 Flash API

Provides Flash erase/program APIs. For usage, refer to the Flash API User Manual.

8.4 math

Typically includes Fix32math, FPUfastRTS, FastIntDiv, and related math libraries.

8.5 Other

The package may also include CoreMark, PMBus, Secure ROM APIs, and related items; follow the current SDK tree.

9 About utilities

This directory contains development helper tools. The current package mainly includes:

Note: The utilities content described in this chapter is under G32R5xx_SDK/utilities.

9.1 geehy_tool

Path: utilities/geehy_tool/. Main programs are listed below; for full CLI parameters and examples, see AN1131 *Tool User Manual*.

Main programs in the directory:

- Geehy_Bin.exe: firmware format conversion and CMAC-related processing (supports boot / UART / SPI / I2C options, and primary / extended secure boot modes; use --cmac to specify a key file)
- Geehy_Serial_flash_Programmer.exe / Geehy_Serial_flash_Programmer_appln.exe: serial Flash download tools
- hex2bin.exe: hexadecimal/binary conversion helper
- key.txt, .cmd: sample key and command scripts

Note: Before use, run one sample command from AN1131 or the tool “--help” output in the same directory.

9.2 keil_dbg_tool

MDK debug helper tools (see AN1129). For G32R502 r502_dbg.ini setup points, see Chapter 11.

9.3 G32R5xx_AddOn

IAR device-support installer (see AN1126).

10 G32R501 first run (summary)

1. Install the CMSIS-Pack / IAR Add-On (per AN1126).
2. Open an evaluation-board LED or similar starter example under driverlib/g32r501/examples/ (MDK / IAR / Eclipse).
3. Confirm device selection and debug scripts, then build and download to the G32R501 EVAL.
4. For detailed IDE and toolchain steps, see AN1126; for the example list, see AN1127.

11 G32R502 first run

This chapter is organized from the G32R502 device tree and evaluation-board paths so you can run a 502 example for the first time inside the G32R5xx SDK package.

11.1 Environment and toolchain reference

SDK examples target **G32R502MC** (Cortex-M52). Align with the reference toolchain versions in Table 1 (newer versions may work; if issues occur, align to the reference versions first).

Table 1 Reference toolchain versions

IDE / toolchain	Reference version
Keil MDK (uVision)	v5.40 or later (v5.43 recommended)
Arm Compiler for Embedded 6	v6.24
CMSIS-Pack	Geehy.G32R5xx_DFP (follow the pack under package/ in the package)
IAR Embedded Workbench for Arm	**G32R502 series requires 9.70.4 or later**
IAR Add-On	G32R5xx_AddOn (follow the version under utilities/G32R5xx_AddOn/)
Eclipse Embedded CDT	2023-12 or later
Arm LLVM (Eclipse)	19.1.x

11.2 Install device support

1. **Keil MDK:** Use Pack Installer to install the G32R5xx DFP pack under package/ (includes SVD and Flash algorithms). If direct install stalls, rename the extension to .zip, extract, then use **Import Folder**.
2. **IAR Embedded Workbench for Arm:** Close IAR, run the installer under utilities/G32R5xx_AddOn/, and select the IAR **installation root** (do not select a subdirectory such as config under the install tree). Select the IAR installation root (not a subdirectory such as config). After installation, restart IAR. **When developing for the G32R502 series, the IAR version must be 9.70.4 or later.**
3. **Debug (optional):** For J-Link Devices install and pyOCD registration, see **AN1126** (J-Link device support install and Chapter 7 pyOCD).

11.3 Open the first example (LED)

Recommended path:

```
driverlib/g32r502/examples/eval/led/led_ex1_blinky/project/
├── MDK/project.uvprojx
```

```
├── IAR/project.ewp
└── Eclipse/.cproject
```

Using MDK as an example:

1. Open project.uvprojx with Keil.
2. Confirm the Target name is **G32R502**.
3. Confirm the G32R5xx DFP is installed and the Device is a G32R502 series device.
4. Per Section 11.4, confirm that r502_dbg.ini and the debug options for this MDK project are configured correctly.
5. Build and download to the G32R502 evaluation board.

Note: This example lights evaluation-board **LD2 (GPIO49)** and does not occupy JTAG TDI. See the example source file header for details.

UART printf example path: driverlib/g32r502/examples/eval/uart/uart_ex5_printf/project/.

11.4 MDK debug initialization file (r502_dbg.ini)

In the current SDK, G32R502 MDK example project directories usually **already include** r502_dbg.ini (for example, driverlib/g32r502/examples/eval/adc/adc_ex1_soc_software/project/MDK/r502_dbg.ini). Projects reference it by relative path. The first time you open a 502 example in Keil, verify the following configuration.

1. Confirm that r502_dbg.ini exists in this project's MDK directory. If it is missing, copy it from the template and name it r502_dbg.ini:

```
Source: utilities/keil_dbg_tool/r502_dbg.ini.template
Destination: <example>/project/MDK/r502_dbg.ini
```

2. Under **Options for Target → Debug → Initialization File**, confirm .\r502_dbg.ini.
3. Confirm **User → After Build/Rebuild** is configured to call keil_dbg_tool (follow the command already in the project) so MSP/PC are updated after build.
4. Run a full Build; after the tool reports success, Debug / Download.
5. Use one r502_dbg.ini per MDK example project directory; do not share one physical file across multiple projects.

For fuller tool options, see AN1129. Minimum check: template “utilities/keil_dbg_tool/r502_dbg.ini.template” copied to “<example>/project/MDK/r502_dbg.ini”, and Options for Target → Debug → Initialization File set to “.\r502_dbg.ini”.

11.5 Build and download notes

- **Target name:** Unified as G32R502 across IDE projects.
- **Linker scripts:** eval examples default to Flash execution with a RAM vector table layout.
- **Compiler options:** Must include -mcpu=cortex-m52+nomve.fp+nofp.dp+cdecp0 (follow the options already in the project).

11.6 Known toolchain limitations

- **IAR (G32R502):** For G32R502 series devices, the IAR development environment **must** be IAR Embedded Workbench for Arm **9.70.4 or later**; earlier versions do not meet the requirements for this series.
- **Keil MDK register display:** On MDK 5.40, Debugger display of 16-bit register fields is usually normal; on MDK 5.43.1, similar fields may display incorrectly. If register-window values are required, temporarily use MDK 5.40, or cross-check in the Memory window.
- **Keil MDK source navigation:** Below MDK 5.43, when Device is set to Cortex-M52, F12 (Go to Definition) may be unavailable; use MDK 5.43 or later if source navigation is required.
- **Keil MDK + J-Link V12 (Cortex-M52):** When using MDK with J-Link V12 to debug a Cortex-M52 target, breakpoints may not hit as expected, single-step may land incorrectly, or variable inspection may be wrong. If this occurs, first rule out common causes such as an overly aggressive optimization level, a downloaded image that does not match the sources, unstable probe wiring, and an abnormal target reset state. Current information indicates a possible incompatibility between the J-Link driver and MDK Debug settings for M52 (for example MVE and PACBTI). This applies to that combination only; it does not mean every J-Link, every MDK version, or every M52 project will show the same issue.

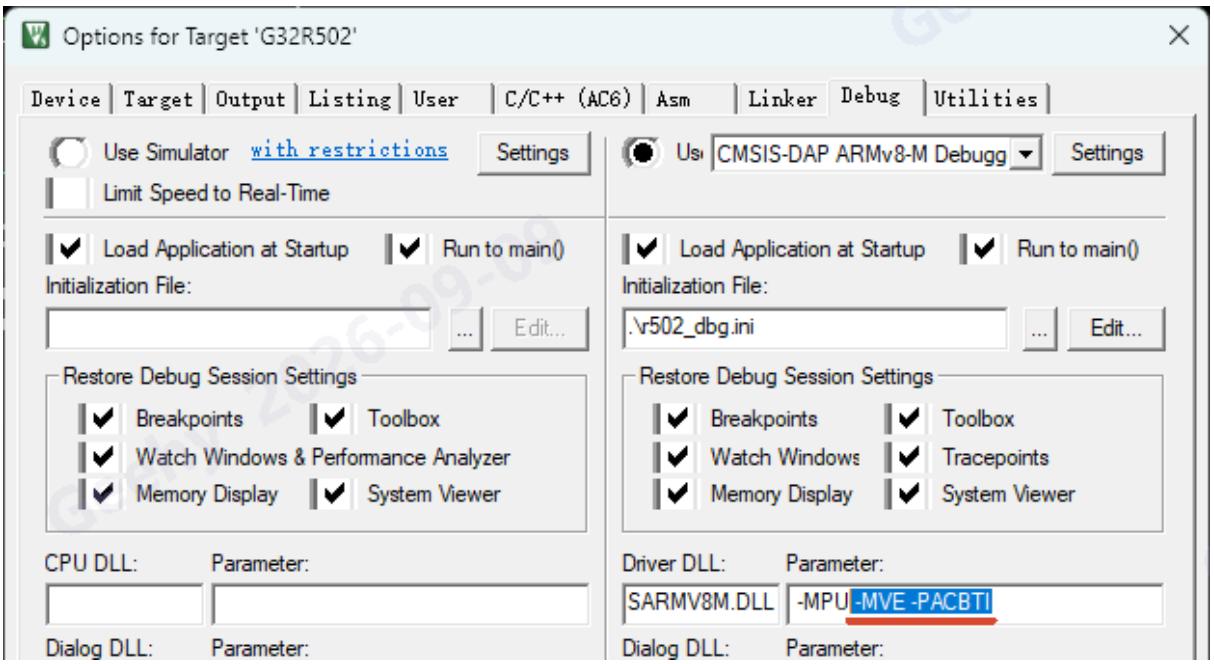
As a temporary check, keep a project copy or record the original options, then remove optional Driver DLL parameters such as “-MVE” and “-PACBTI”, rebuild, and download again to see whether the issue eases. Steps:

1. Open the target project in MDK and confirm the CPU is Cortex-M52 (for example G32R501 / G32R502).
2. Open **Options for Target** and open the **Debug** page.
3. Record the Debugger, interface, download settings, and optional Driver DLL Parameter values such as “-MVE” and “-PACBTI” for restore and comparison. Figure 2 shows where these parameters appear (the Debugger type in the screenshot is only an example; with J-Link, inspect and edit the same Parameter box).
4. After recording the original values, remove optional parameters such as “-MVE” and “-PACBTI” from Parameter (keep unrelated parameters if present, for example “-MPU” in the

sample).

5. Save the project, run Rebuild, and download the new image.

Figure 2 MVE and PACBTI in the MDK Debug Driver DLL Parameter field



12 Revision

The Document revision is shown in Table 2.

Table 2 Document revision

Date	Version	Description
2025.1	1.0	● Initial release
2025.4	1.1	● Corrections
2026.8	1.2	● Listed applicable products as G32R501 and G32R502 ● Added G32R502 first-run steps ● Stated that G32R502 requires IAR 9.70.4 or later.

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 “TM” 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.