GigaDevice Semiconductor Inc.

GD-Link V2 Adapter

User Guide

Revision 1.2

(Nov. 2025)

Table of Contents

List of Figures

List of Tables

1. Introduction

GD-Link V2 is a rich-featured, easy-to-use, and portable debugging and programming tool developed by GigaDevice for GD32 series MCU, which has the following characteristics:

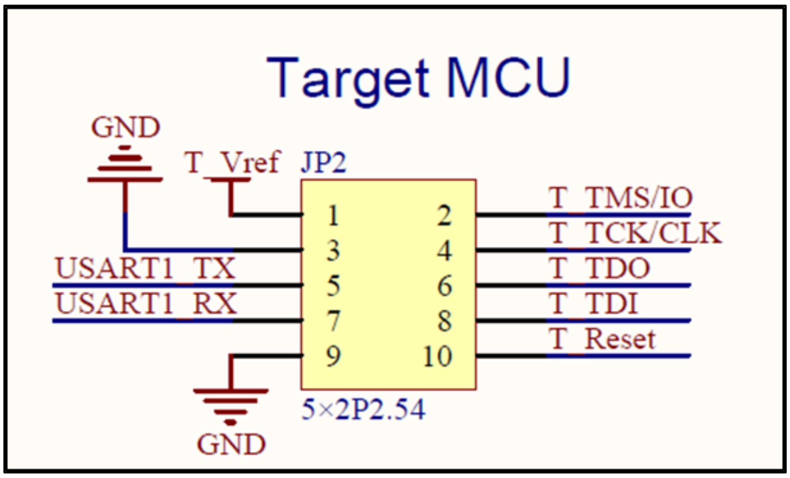
- USB2.0 high-speed interface
- Provide 5V or 3.3V power supply for the target chip
- Support firmware update through the GD-Link Programmer software
- Support SWD / JTAG debugging and programming interface
- Support GD32 ARM / RISC-V core full series of chips
- Support KEIL / IAR / Eclipse debugging and programming
- Support offline programming
- Support virtual USB disk drag and drop programming
- Support SWO function
- Support virtual serial port printing

2. Hardware introduction

2.1. Pin definitions and wiring methods

To enable programming, debugging, serial communication, and printing functions, connect the GD-Link V2 pins to the SWD (SWO), JTAG, or USART interface of the target chip using DuPont wires or ribbon cables. The pinout of GD-Link V2 is illustrated in ***Figure 2-1. GD-Link V2 pinout diagram***.

Figure 2-1. GD-Link V2 pinout diagram



The functions of each GD-Link V2 pin are described as shown in ***Table 2-1. GD-Link V2 pin function definitions***.

Table 2-1. GD-Link V2 pin function definitions

Pin Number	Pin Name	Description
1	T_Vref	Target chip power supply, providing 3.3V / 5V
2	T_TMS/IO	JTAG TMS pin / SWD SWDIO pin
3	GND	Power ground
4	T_TCK/CLK	JTAG TCK pin / SWD CLK pin
5	USART1_TX	Serial transmission pin
6	T_TDO	JTAG TDO pin / SWO pin
7	USART1_RX	Serial reception pin
8	T_TDI	JTAG TDI pin
9	GND	Power ground
10	T_Reset	JTAG / SWD target chip reset pin

The diagram of GD-Link V2 hardware connection to the target chip is illustrated in ***Figure 2-2. SWD interface connection diagram***, ***Figure 2-3. JTAG interface connection diagram***, ***Figure 2-4. SWD + SWO interface connection diagram*** and ***Figure 2-5. Serial interface connection diagram***.

Figure 2-2. SWD interface connection diagram

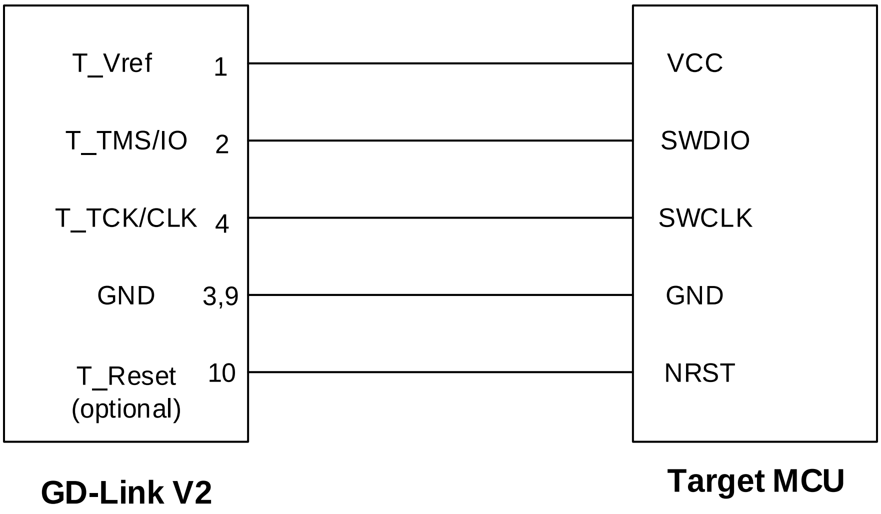


Figure 2-3. JTAG interface connection diagram

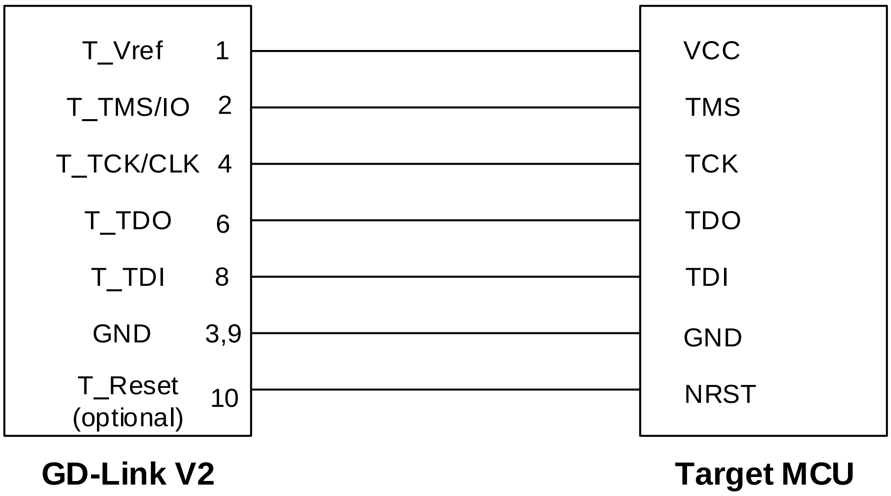


Figure 2-4. SWD + SWO interface connection diagram

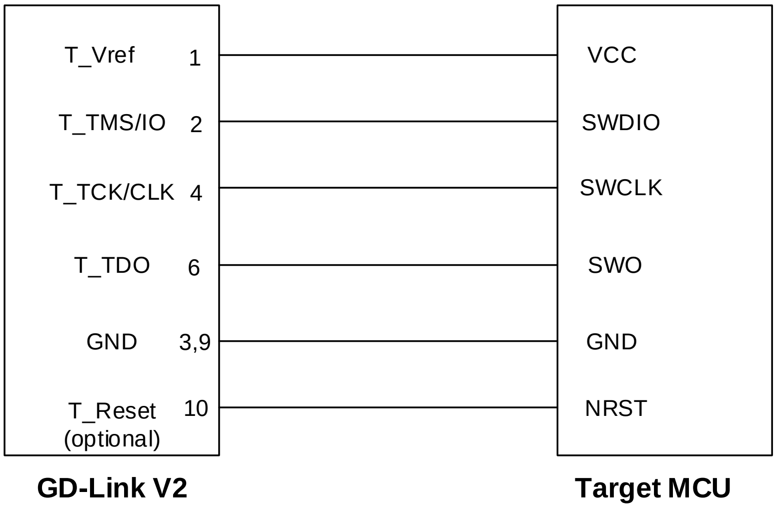
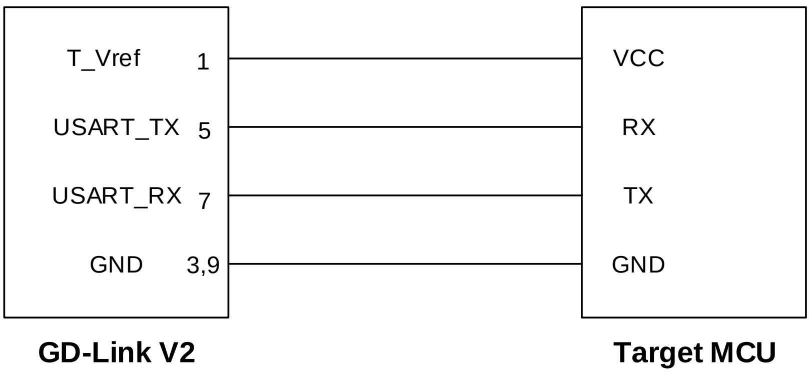


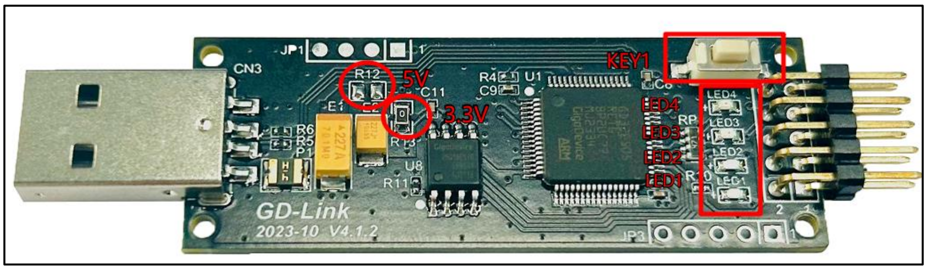
Figure 2-5. Serial interface connection diagram



2.2. Button, LEDs and Buzzer

GD-Link V2 features a single button (K1), a buzzer (BZ1) and four LEDs (LED1/2/3/4) as indicators. The physical representation of GD-Link V2 is shown in ***Figure 2-6. GD-Link V2 adapter hardware***. The button K1 is used for firmware updates and offline programming. For specific usage instructions, please refer to the firmware update and offline programming section.

Figure 2-6. GD-Link V2 adapter hardware



During offline programming and drag-and-drop programming from a virtual USB disk, when the target chip has been successfully programmed with the desired file, the buzzer will beep, indicating a successful programming status. The on-off and blinking of the LED indicate different working states of GD-Link V2. ***Table 2-2. Working status of GD-Link V2*** provides a description of the different status of these LEDs which indicate the status of programming and debugging tool.

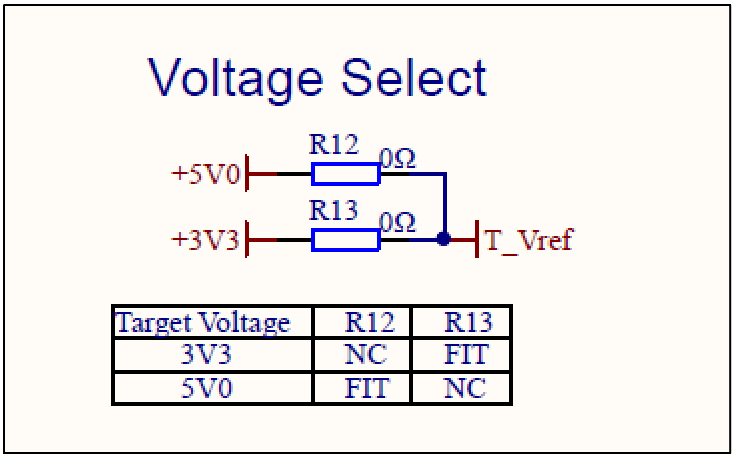
Table 2-2. Working status of GD-Link V2

LED	LED status	GD-Link V2 working status
LED1	always bright	Offline programming or drag-and-drop programming successful
	flashing	performing offline programming or drag-and-drop programming
LED2	flashing fast	USB connection successful
	flashing slow	USB not connected
LED3	always bright	Firmware update status
LED4	always bright	Power supply is normal

2.3. Output voltage

The debugger provides 5V and 3.3V output voltages for users to choose. The output voltage can be modified by short-connecting the R12 and R13 resistors in the hardware through the 0Ω resistor. The schematic diagram of the voltage selection is shown in ***Figure 2-7. GD-Link V2 output voltage select***. Reference ***Figure 2-6. GD-Link V2 adapter hardware***, when the 0Ω resistor is welded at R13, the T_Vref voltage is 3.3V, when the 0Ω resistor is welded at R13, and the output T_Vref voltage is 5V.

Figure 2-7. GD-Link V2 output voltage select



Target Voltage	R12	R13
3V3	NC	FIT
5V0	FIT	NC

3. Software features

3.1. Firmware updates

GD-Link V2 provides firmware update functionality. Firmware updates are used to:

- Support the latest MCUs released by GD32.
- Fix issues present in the firmware.

GD-Link V2 can be updated using the GD-Link Programmer software. Users can visit the GD32MCU official website to obtain the latest version of the GD-Link Programmer software, unzip it after downloading, and follow these firmware update steps:

1. Disconnect GD-Link V2 from the computer's USB port.
2. While holding down button K1, plug GD-Link V2 back into the computer's USB port. At this time, LED3 is always on, indicating that the programmer is in firmware upgrade mode.
3. Release button K1 and click the "GD-Link" menu in the GD-Link Programmer software. Choose "Update Firmware" to start the firmware update process.
4. A progress bar will pop up in the GD-Link Programmer software, indicating the progress of the update. Wait for it to reach 100% and show a successful update message.

Refer to *Figure 3-1. GD-Link V2 firmware update step 1*, *Figure 3-2. GD-Link V2 firmware update step 2* and *Figure 3-3. GD-Link V2 firmware update step 3* for visual guidance on the firmware update process.

Figure 3-1. GD-Link V2 firmware update step 1

Figure 3-2. GD-Link V2 firmware update step 2

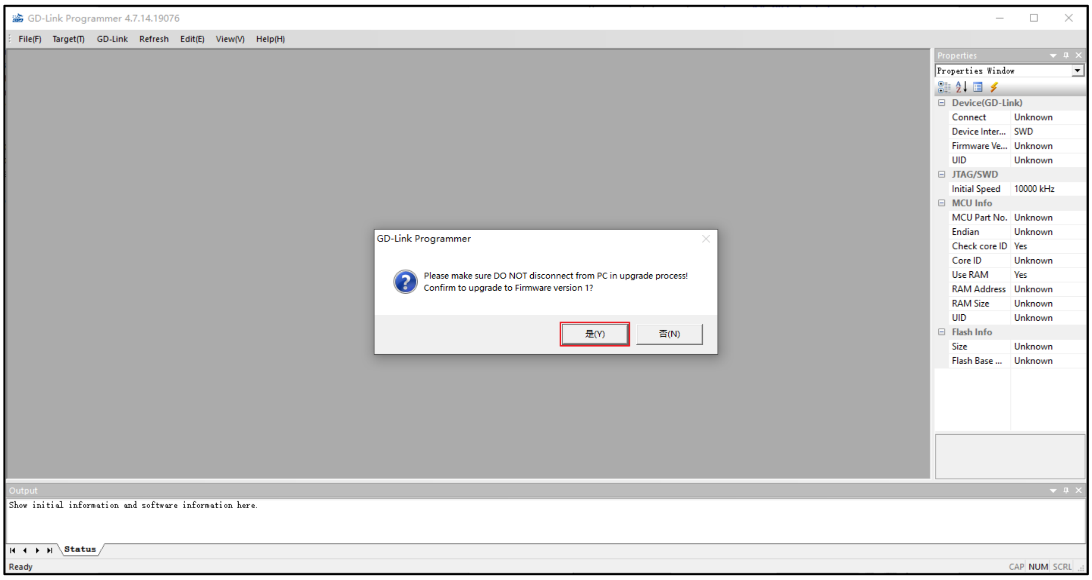
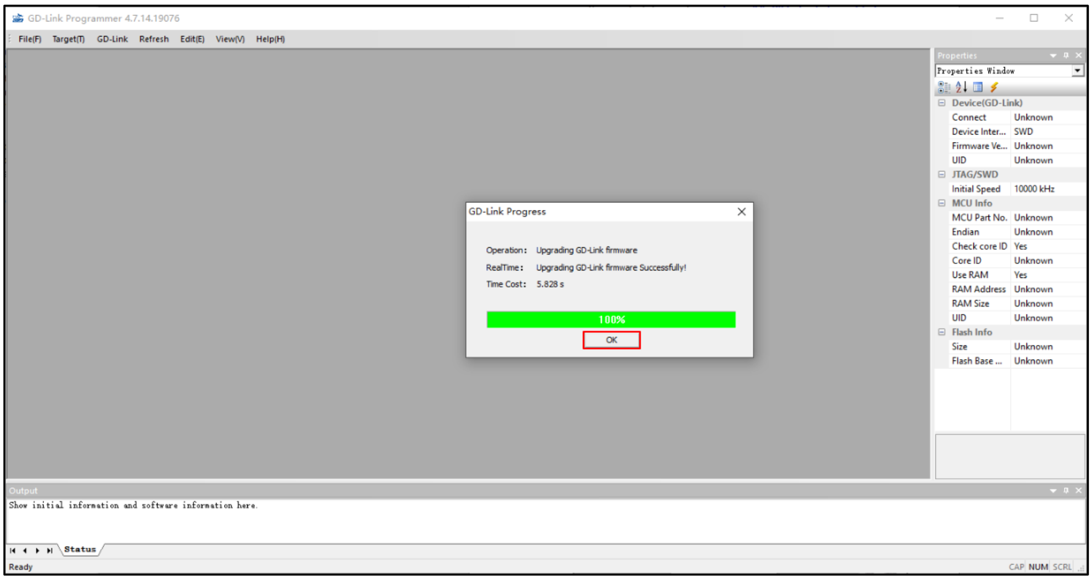


Figure 3-3. GD-Link V2 firmware update step 3



After the update is completed, user can check the current firmware version number in the properties pane, as shown in ***Figure 3-4. GD-Link V2 firmware update step 4***.

Figure 3-4. GD-Link V2 firmware update step 4

Note: During the firmware update process, do not unplug GD-Link V2 from the computer's USB port.

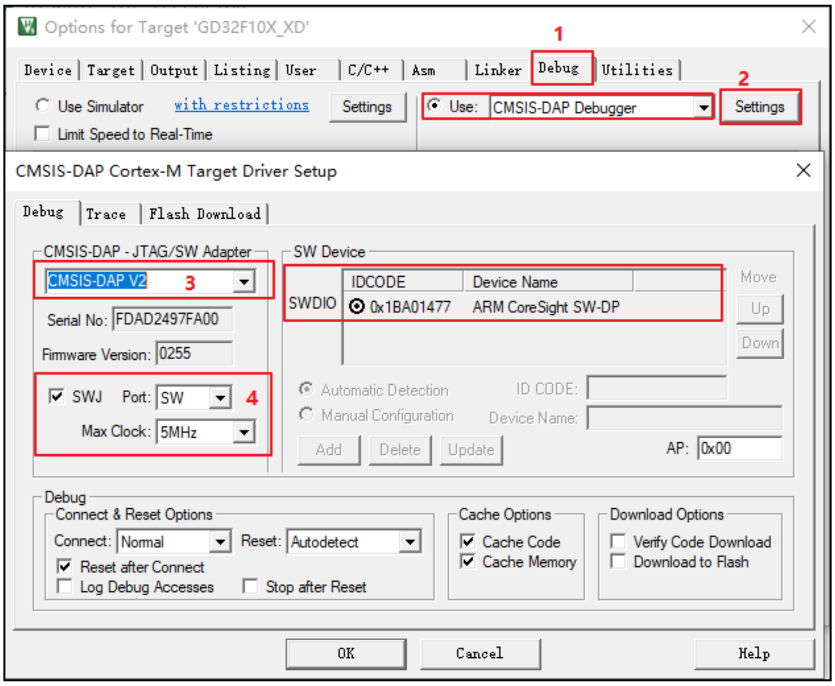
3.2. Programming function

3.2.1. IDE programming

Programming with KEIL (version 5.27 and above)

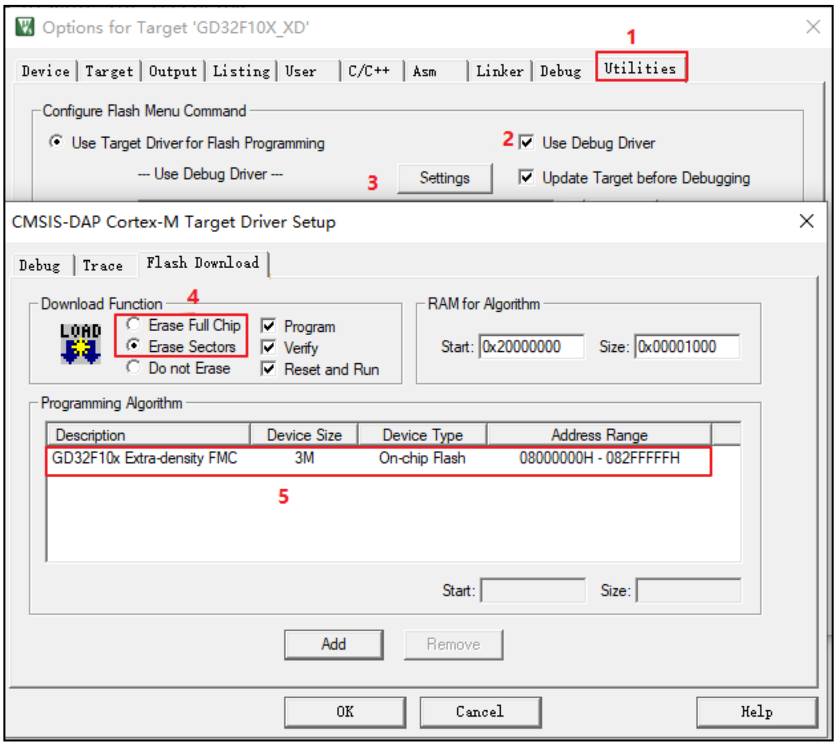
Connect GD-Link V2 to the target chip according to the hardware connection described in ***Pin definitions and wiring methods*** section. Connect the USB interface of GD-Link V2 to the PC, and wait for LED2 to enter rapid blinking mode. Open KEIL software, in the KEIL Debug tab, select "CMSIS-DAP Debugger" or "CMSIS-DAP ARMv8-M Debugger" in the "Debug" option, as shown in ***Figure 3-5. KEIL debug configuration***.

Figure 3-5. KEIL debug configuration



In the "Utilities" tab, select "Use Debug Driver" and click the "Setting" button to choose the MCU download algorithm and configure the erase mode and other settings, as shown in ***Figure 3-6. KEIL utilities configuration***.

Figure 3-6. KEIL utilities configuration



Click the "Download" icon in the KEIL menu bar. In the "Build Output" window, the programming progress can be monitored, as shown in ***Figure 3-7. KEIL Download Icon*** and

Figure 3-8. Build output window - programming successful.

Figure 3-7. KEIL Download Icon

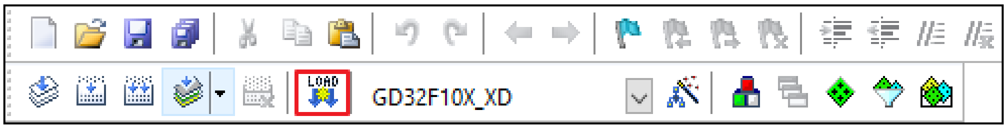


Figure 3-8. Build output window - programming successful

```
Full Chip Erase Done.
Programming Done.
Verify OK.
Flash Load finished at 16:06:29
```

Programming with IAR (version 8.50 and above)

Connect GD-Link V2 to the target chip according to the the hardware connection described in *Pin definitions and wiring methods* section. Connect the USB interface of GD-Link V2 to the PC, and wait for LED2 to enter rapid blinking mode. Open IAR software. In the IAR "Project" menu, choose "Options." In the "Debugger" tab, choose "CMSIS-DAP" as the debugger driver, as shown in *Figure 3-9. IAR debugger configuration*. In the "Setup" tab, choose the MCU type, download algorithm, and other configurations according to the target chip's requirements, as shown in *Figure 3-10. IAR CMSIS DAP configuration*.

Figure 3-9. IAR debugger configuration

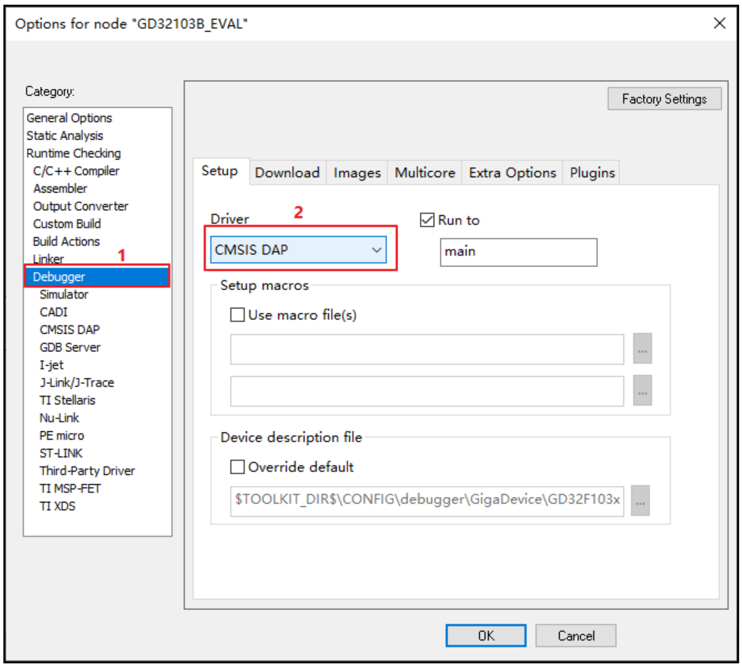
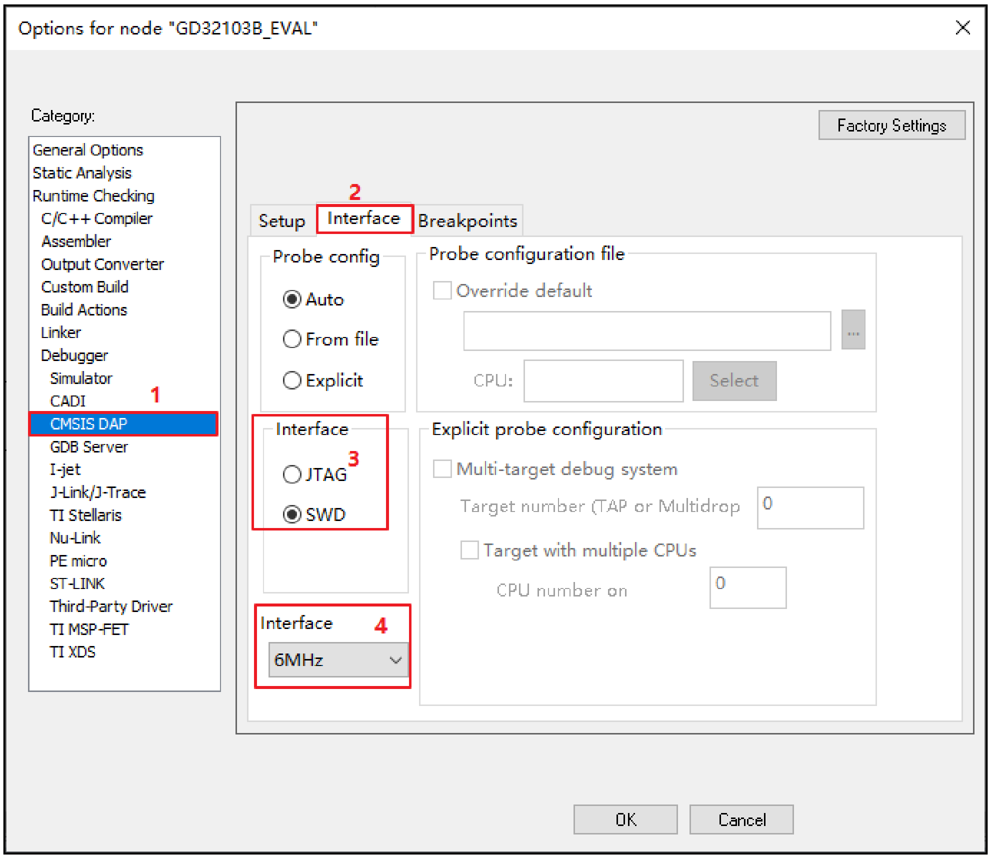


Figure 3-10. IAR CMSIS DAP configuration



In the menu bar "Project" drop-down option "Download", click "Download active application" and wait for the progress bar to complete the burning, as shown in ***Figure 3-11. IAR download button*** and ***Figure 3-12. IAR download progress bar***.

Figure 3-11. IAR download button

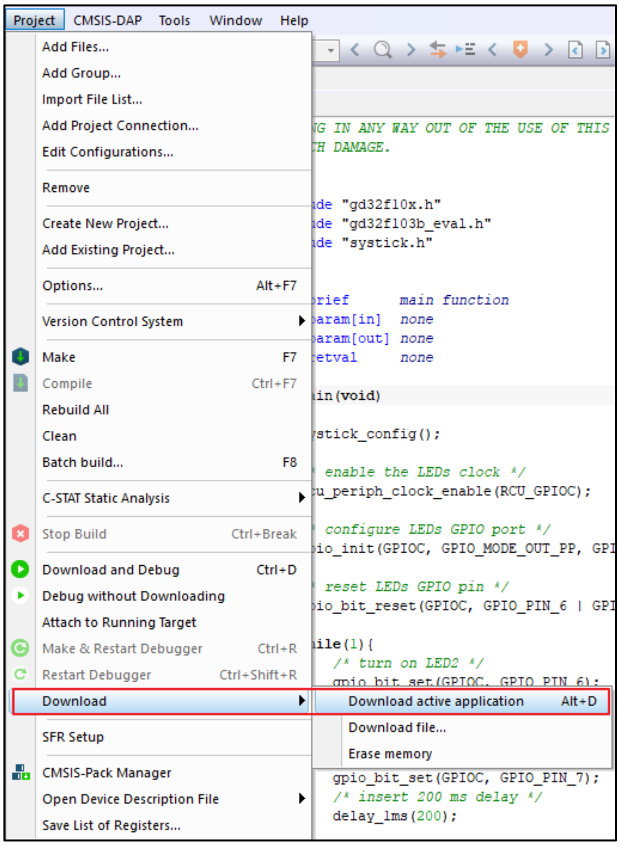
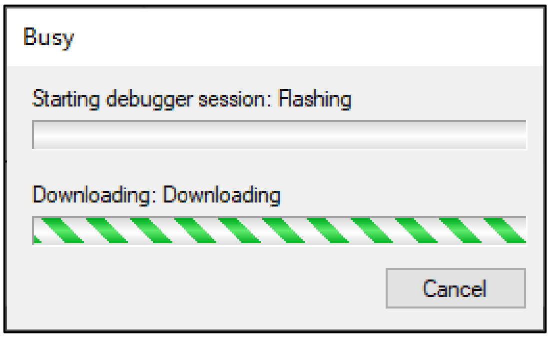


Figure 3-12. IAR download progress bar



Programming with Eclipse

Connect GD-Link V2 to the target chip according to the hardware connection described in the ***Pin definitions and wiring methods*** section. Connect the USB interface of GD-Link V2 to the PC, and wait for LED2 to enter rapid blinking mode. Open the Eclipse software and click "RUN" menu and select the dropdown option "Debug Configurations..." to enter the "Debugger" tab, as shown in ***Figure 3-13. Access the "Debug Configurations" interface***. Configure the OpenOCD path correctly and fill in the cfg file to be used in the "Config options" section, as demonstrated in ***Figure 3-14. Configure the "Eclipse Debug" tab*** in the Eclipse Debug Configuration interface.

After completing the configuration, click the "Apply" button to save the settings. Then, select the "Debug" button, and when the "Confirm Perspective Switch" window appears, click "YES" to confirm. This will initiate the code download and take to the debugging interface, as illustrated in *Figure 3-15. Enter the debugging interface in Eclipse*.

Figure 3-13. Access the "Debug Configurations" interface

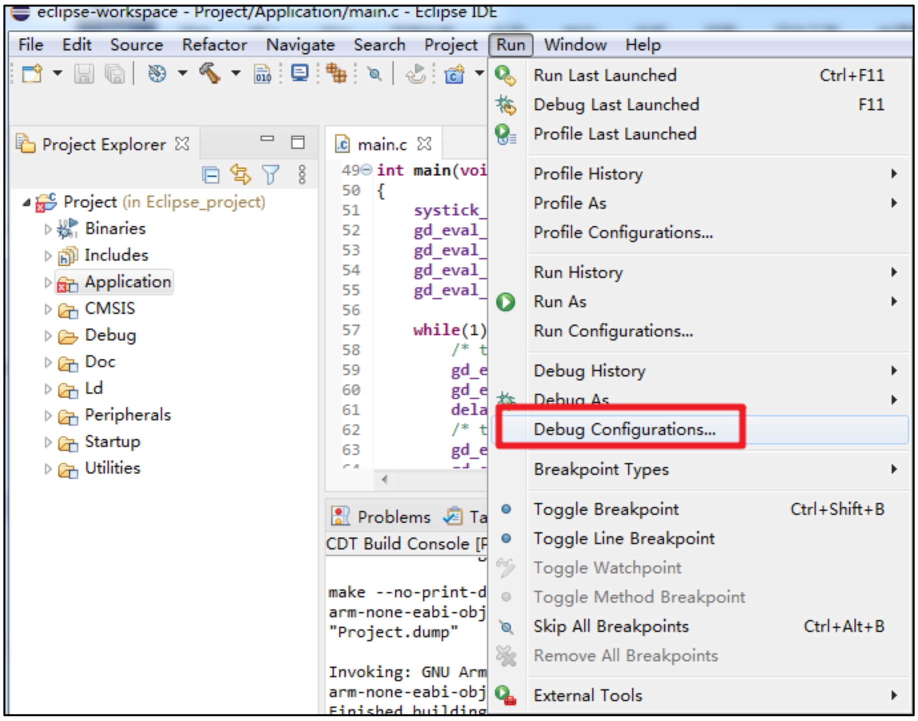


Figure 3-14. Configure the "Eclipse Debug" tab

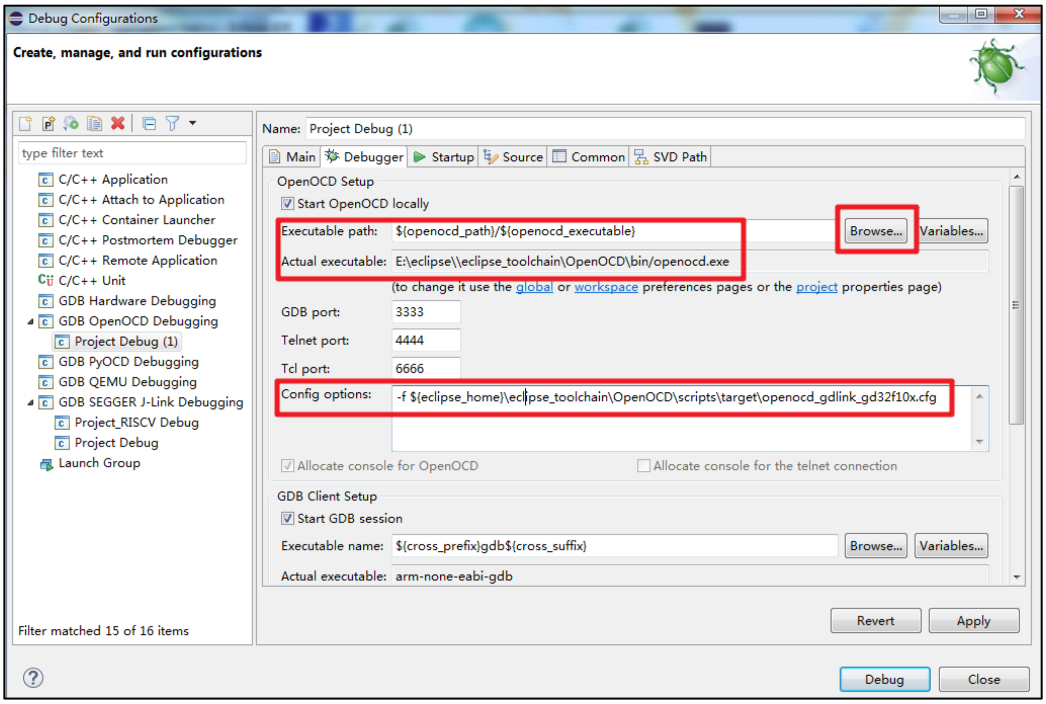
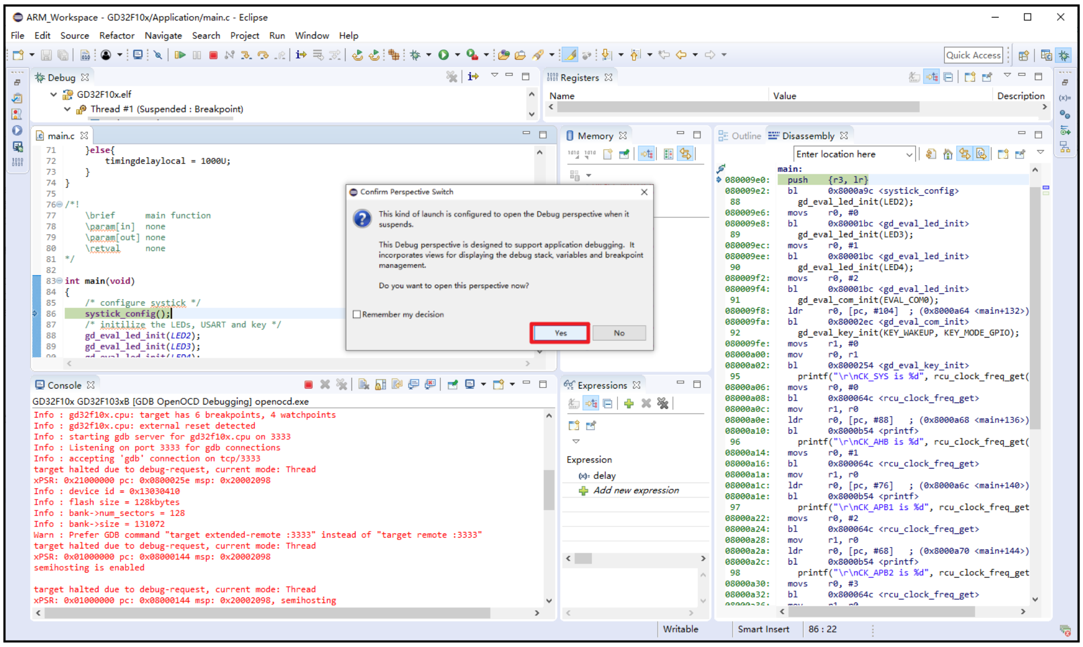


Figure 3-15. Enter the debugging interface in Eclipse



3.2.2. GD-Link Programming

Connect GD-Link V2 to the target chip according to the hardware connection described in ***Pin definitions and wiring methods*** section. Connect the USB interface of GD-Link V2 to the PC, and wait for LED2 to enter rapid blinking mode. Open the GD-Link Programmer software and select the JTAG / SWD programming interface and configure the communication speed in the "Properties" window. Refer to ***Figure 3-16. GD-Link programmer programming options configuration*** for an illustration of GD-Link Programmer programming options.

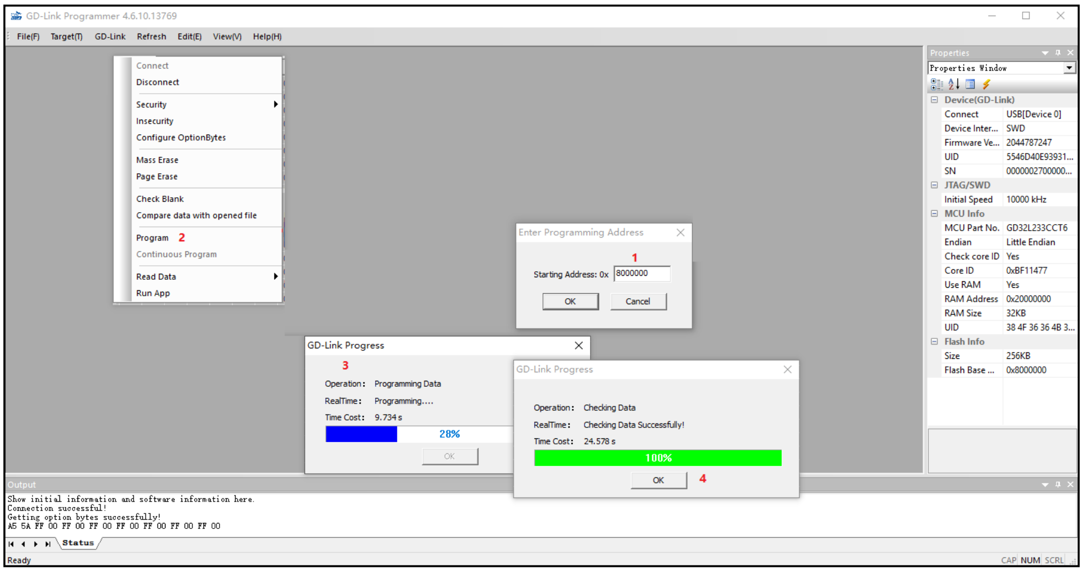
Figure 3-16. GD-Link programmer programming options configuration

Click the "Target" dropdown menu and choose the "Connect" option. Check the "Output" window for a message indicating "Connection successful." At the same time, the detailes information about the connected target chip, including its specific type are listed in the "Properties" window. Refer to ***Figure 3-17. Connecting the target chip in GD-Link Programmer*** for an illustration of GD-Link Programmer successfully connecting to the target chip.

Figure 3-17. Connecting the target chip in GD-Link Programmer

Drag and drop the binary file, "xxx.bin" or the executable file "xxx.hex" into the GD-Link Programmer software. When using the "xxx.bin" file for programming, a dialog will appear on the host computer's software, prompting to enter the starting address for the download. After entering the correct download address, click the "OK" button. Then, select the "Target" dropdown menu and choose the "Program" option. The software will start downloading the program to the target chip. Wait for the progress bar to reach 100%, and a message will confirm the successful download, as shown in ***Figure 3-18. GD-Link Programmer burns target chip***.

Figure 3-18. GD-Link Programmer burns target chip



3.2.3. Offline programming

Connect the USB interface of GD-Link V2 to the PC, and wait for LED2 to enter rapid blinking mode. Open the GD-Link Programmer software. Click "GD-Link" menu bar and then choose "Configuration" to configure the parameters of offline programming, referring to ***Figure 3-19. GD-Link V2 offline download parameter configuration***. The following configurations can be performed using this interface:

- Whether to enable read protection after offline programming completion.
- Erase method selection: full chip erase or page erase.
- Limit the number of offline programming downloads.

Click the "OK" button in the offline programming parameter configuration interface to save the settings. After configuration, in the menu bar, click "GD-Link" and then "Update File" to enter the file update interface. Referring to ***Figure 3-20. GD-Link V2 offline download file update configuration***. Select the specific part number of the target MCU, add the xxx.bin file, specify the download address to the target chip, and click the "Update" button. Wait for the progress bar to reach 100% to complete the file update, as shown in ***Figure 3-21. Offline download file updated to GD-Link V2***.

File updating supports one-time burning for BOOT+APP functionality. The user can continue to click the "Add" button to add a second bin file, specify the burning address. The file update allows to add a maximum of 8 bin files. The addition process is illustrated in ***Figure 3-22. Simultaneously adding BOOT+APP offline download file update to GD-Link V2***.

For GD32W515 series MCU, offline programming also supports option byte configuration. When selecting the MCU part number as GD32W515 series MCU in the "GD-Link Update File Configuration" window, click the "Configure OptionBytes" button. In the pop-up window, perform the option byte configuration. After configuration, click the "OK" button to save the relevant settings, as shown in ***Figure 3-23. Offline download configuration option byte feature***.

Figure 3-19. GD-Link V2 offline download parameter configuration

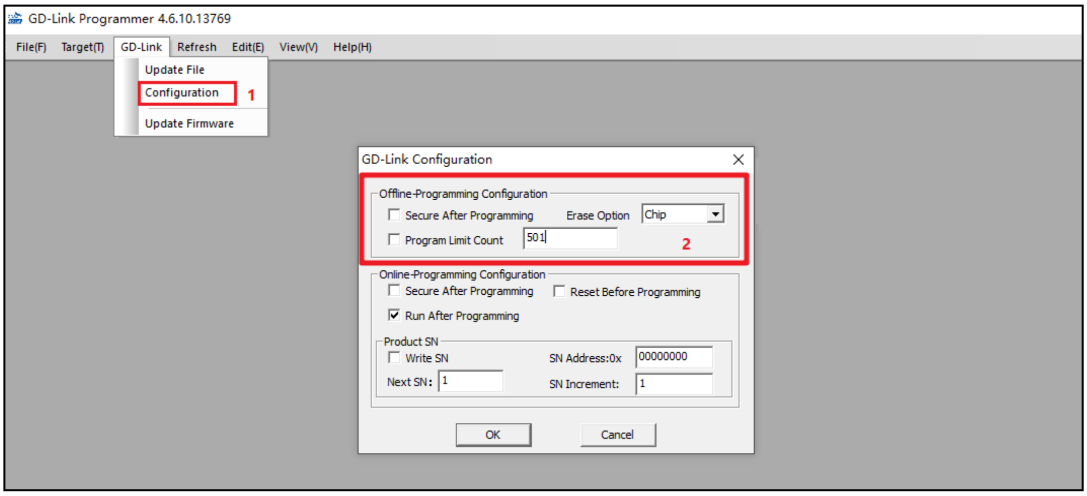


Figure 3-20. GD-Link V2 offline download file update configuration

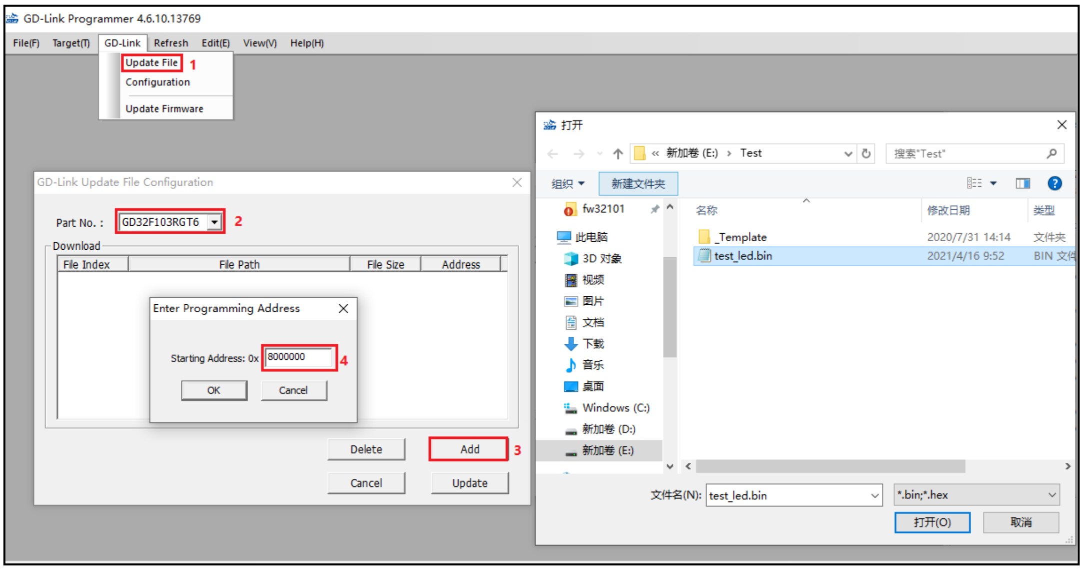


Figure 3-21. Offline download file updated to GD-Link V2

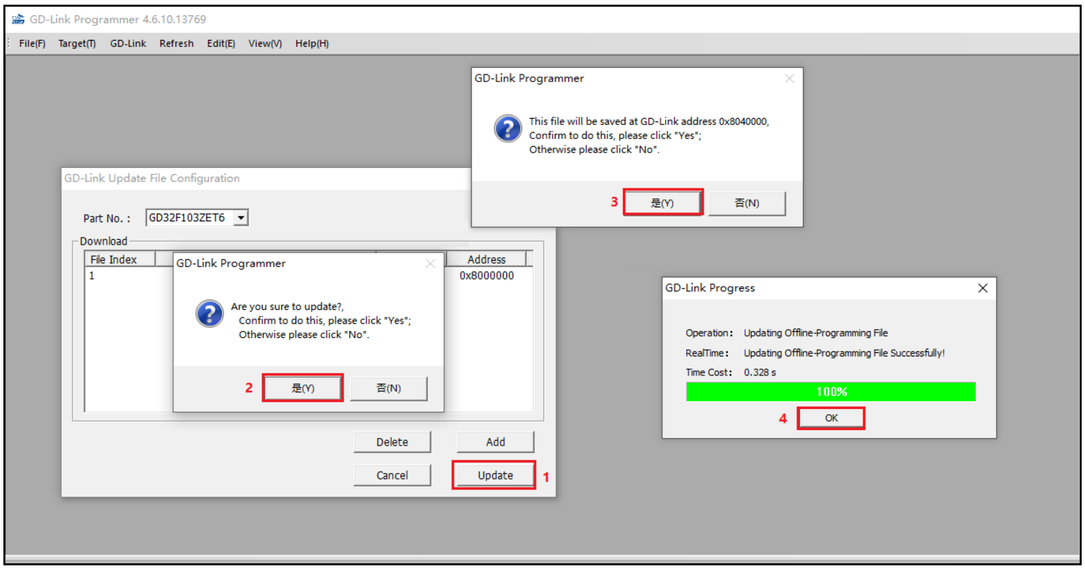


Figure 3-22. Simultaneously adding BOOT+APP offline download file update to GD-

Link V2

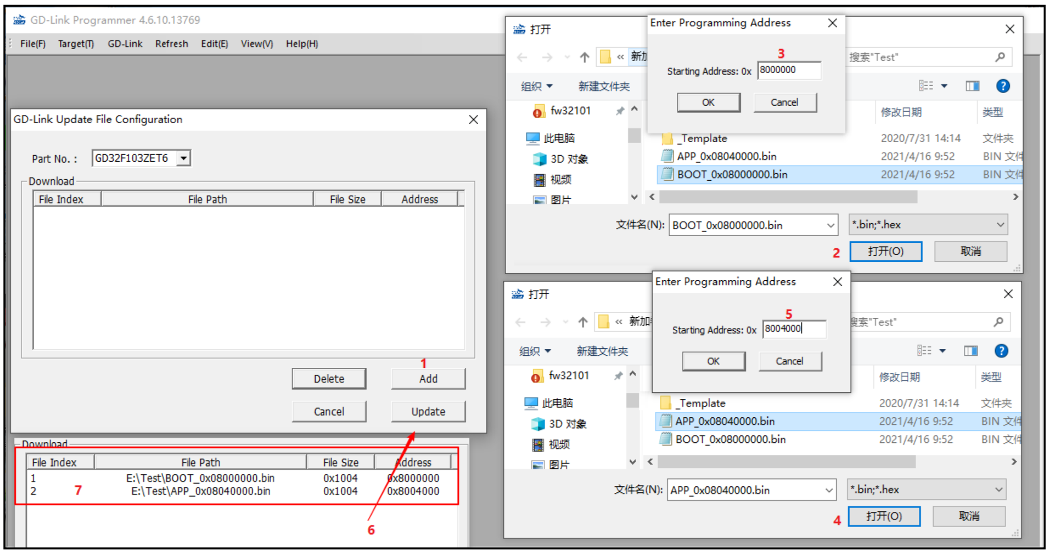
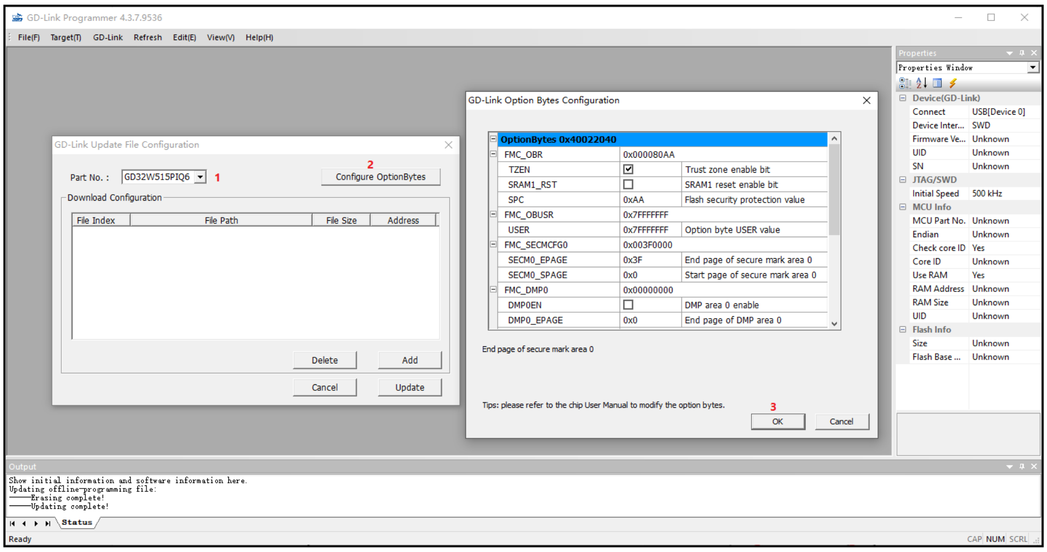


Figure 3-23. Offline download configuration option byte feature



After updating the offline burning file to GD-Link V2, refer to the ***Pin definitions and wiring methods*** section related to hardware connection with the target chip. Manually press the K1 button, if LED1 entering rapid blinking mode, indicating that the offline burning process is ongoing. When the buzzer beeps, it signifies the completion of the offline burning. At this time, LED1 is always bright. If the buzzer does not beep, and LED1 is turned off after blinking, it indicates an offline burning failure.

3.2.4. Machine singal triggered programming

GD-Link V2 offers machine-triggered programming functionality. The signal interface pinout diagram is shown in ***Figure 3-24. Machine signal programming pin distribution schematic diagram***. The functions of each pin for the machine-triggered programming interface are described in ***Table 2-1. GD-Link V2 pin function definitions***. After updating the programming file into the programmer as described in the offline programming section, users can initiate the programming process by providing a 100ms low-level pulse signal to the T_START pin. During the programming process, the T_BUSY pin remains at a low-level signal. When the programming is successful, the T_GOOD pin generates a low-level signal, while a low-level signal on the T_NG pin indicates a programming failure.

Figure 3-24. Machine signal programming pin distribution schematic diagram

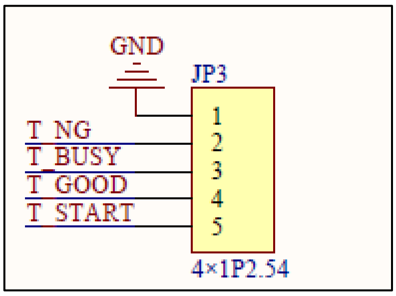


Table 3-1. Machine signal programming pin function definition

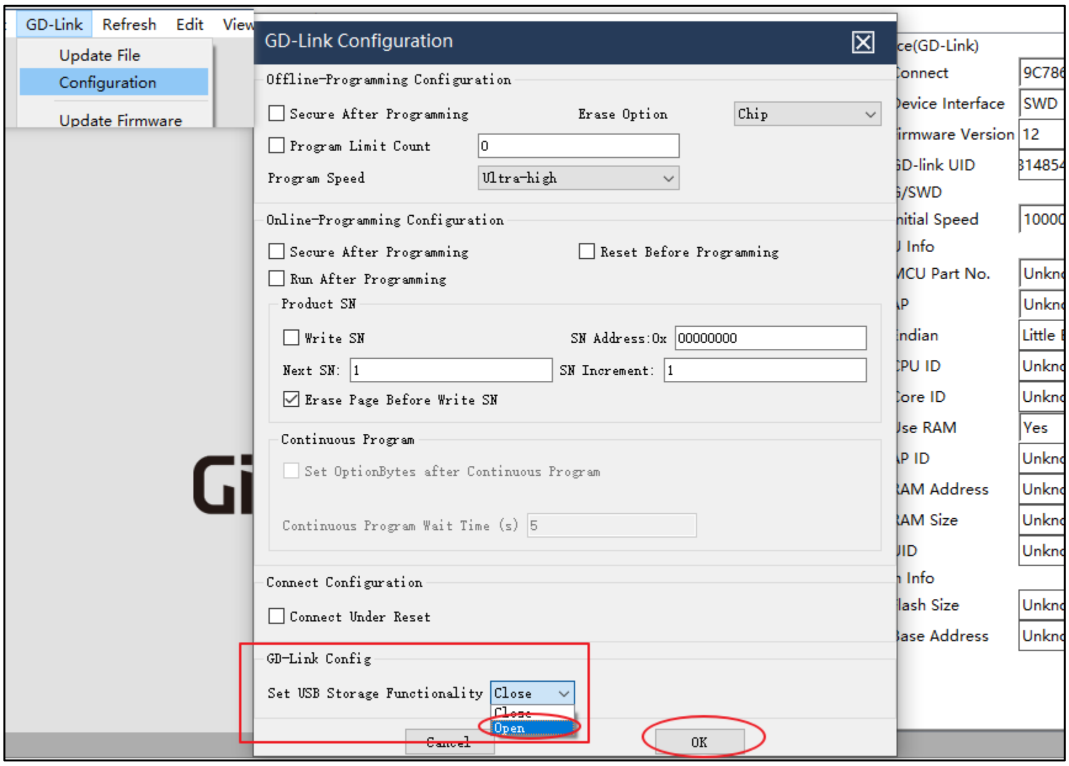
Pin Number	Pin Name	Description
1	GND	Power ground
2	T_NG	Defaults to a high level. When burning fails, this pin goes to a low level.
3	T_BUSY	Defaults to a high level. When burning is in progress, this pin goes to a low level.
4	T_GOOD	Defaults to a high level. When burning is successful, this level goes to a low level.
5	T_START	Defaults to a high level. When this pin receives a low-level signal with a width of 100ms, burning starts.

3.2.5. Virtual USB disk drag and drop programming

Virtual USB disk drag and drop programming function configuration

Starting from firmware version 11, the drag-and-drop functionality of the virtual USB disk is disabled by default. After connecting GD-Link, user can check whether this function is enabled or not through the host software (host software version GD-LinkUtilityProgrammer_win_I_v2.0.2.34758 or later). To enable this function, navigate to "GD-Link -> Configuration -> GD-Link Config -> Set USB Storage Functionality" and select "Open".

Figure 3-25. Virtual USB disk drag and drop programming function configuration



Insert the GD-Link V2 USB into the PC port. There will be a USB mass storage device in the PC device manager, and a GigaDevice disk with the GD logo will appear in the local disk. As shown in ***Figure 3-26. USB mass storage device*** and ***Figure 3-27. Virtual USB drive***.

Figure 3-26. USB mass storage device

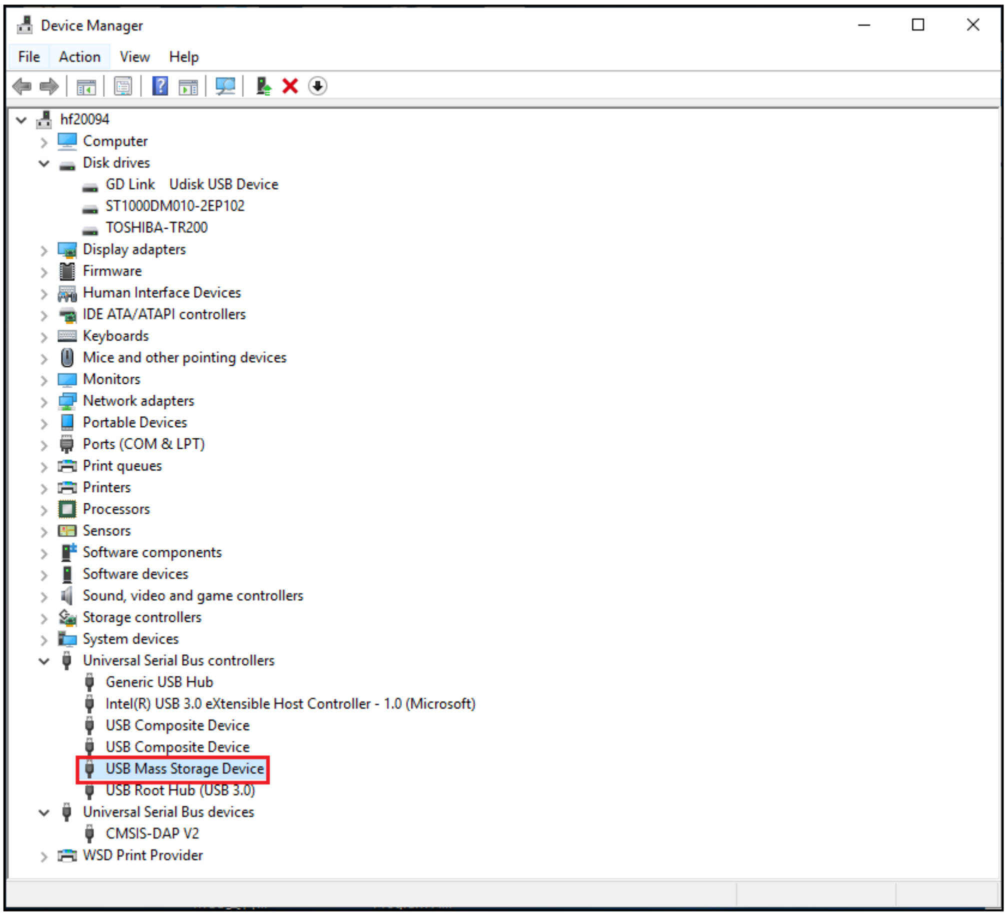
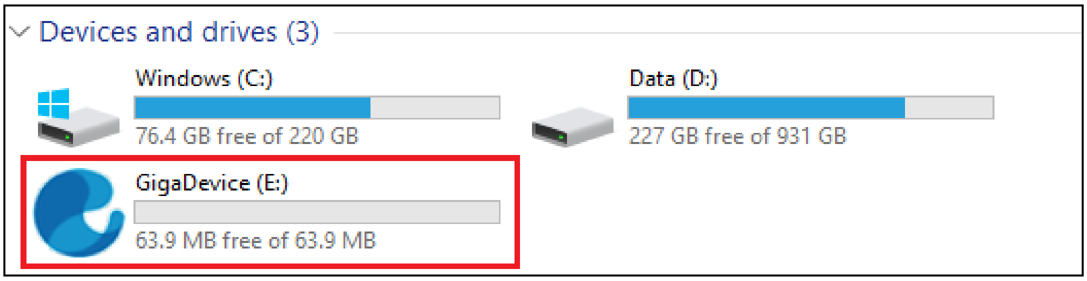


Figure 3-27. Virtual USB drive



Double-click to open the disk. Inside the disk, there is a CONFIG.TXT file. By modifying the content of this file and saving it, the initialize the programming parameter can be configured. The content of the CONFIG.TXT file is shown in ***Table 3-2. CONFIG.TXT file content***.

Table 3-2. CONFIG.TXT file content

Program config:
Target CPU: ARM
Program Flash start address: 0x08000000
Erase method: Page
Reset method: Software
Read protection: Disable
Sram start address: 0x20000000

```
Note: Keep the format of this TXT UTF-8. Please configure programming parameters follow the format
before programming. E.g:
# Program config:
Target CPU: ARM
Program start address: 0x08000000
Erase method: Chip
Reset method: Software
Read protection: Disable
Sram start address: 0x20000000
```

The options and descriptions for each parameter configuration are as shown in ***Table 3-3. Drag-and-Drop programming configuration parameter definitions***.

Table 3-3. Drag-and-Drop programming configuration parameter definitions

Parameter	Options	Description
Target MCU core architecture	ARM	Select ARM as the target chip core
	RISC-V	Select RISC-V as the target chip core
Program flash start address	0x08XXXXXX	Program flash start address 0x08XXXXXX
	0x0CXXXXXX	Program flash start address 0x0CXXXXXX
Erase method	Page	Flash erasing method is page erasing
	Chip	Flash erasing method is full chip erasing
Reset method	Software	Reset method after completing chip download is software reset
	Hardware	Reset method after completing chip download is hardware reset
Sram start address	0x2XXXXXXX	Target chip's SRAM start address is 0x2XXXXXXX
	0x3XXXXXXX	Target chip's SRAM start address is 0x3XXXXXXX
Debug interface	SWD	Select SWD as the download interface(only for ARM)
	JTAG	Select JTAG as the download interface(only for RISC-V)

After configuring the programming parameters, save and close the file. Refer to the ***Hardware introduction*** section, connect GD-Link V2 to the target chip via SWD (GD Cortex-M core MCU) or JTAG interface (GD RISC-V core MCU) correctly, then copy or drag the binary xxx.bin or executable file xxx.hex generated by the IDE or compiler toolchain to the recognized GigaDevice disk device. The programmer will automatically identify the target chip and complete the file programming.

After programming is complete, the virtual USB device will unmount and then remount from the disk. Once mounting is complete, open the GigaDevice disk. If the disk contains only the

CONFIG.TXT file, it indicates a successful file programming. If a FAIL.TXT file appears in the disk, it indicates a programming failure. Double-click to open FAIL.TXT and check the reason for the programming failure.

Note:

1. When the debugger loses power and is unplugged and reconnected, the previous programming parameters will revert to default values.

2. The binary xxx.bin file should be generated by the compiler and the corresponding download target address should be filled correctly, otherwise, programming failure may occur.

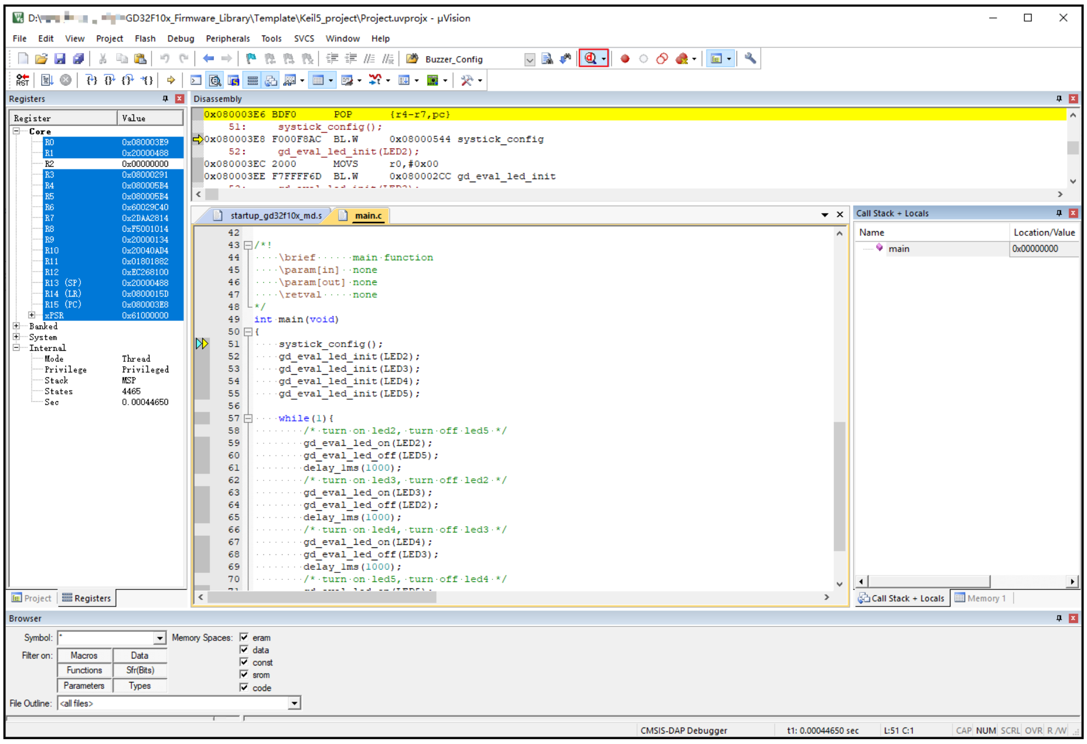
3.3. Debug function

3.3.1. SWD /JTAG debugging

Debugging with KEIL (version 5.27 and above)

Complete the KEIL configuration according to ***IDE programming*** chapter, click the icon button of "Start/Stop Debug Session" in the KEIL menu bar to enter the debugging interface, as shown in ***Figure 3-28. KEIL debugging interface***.

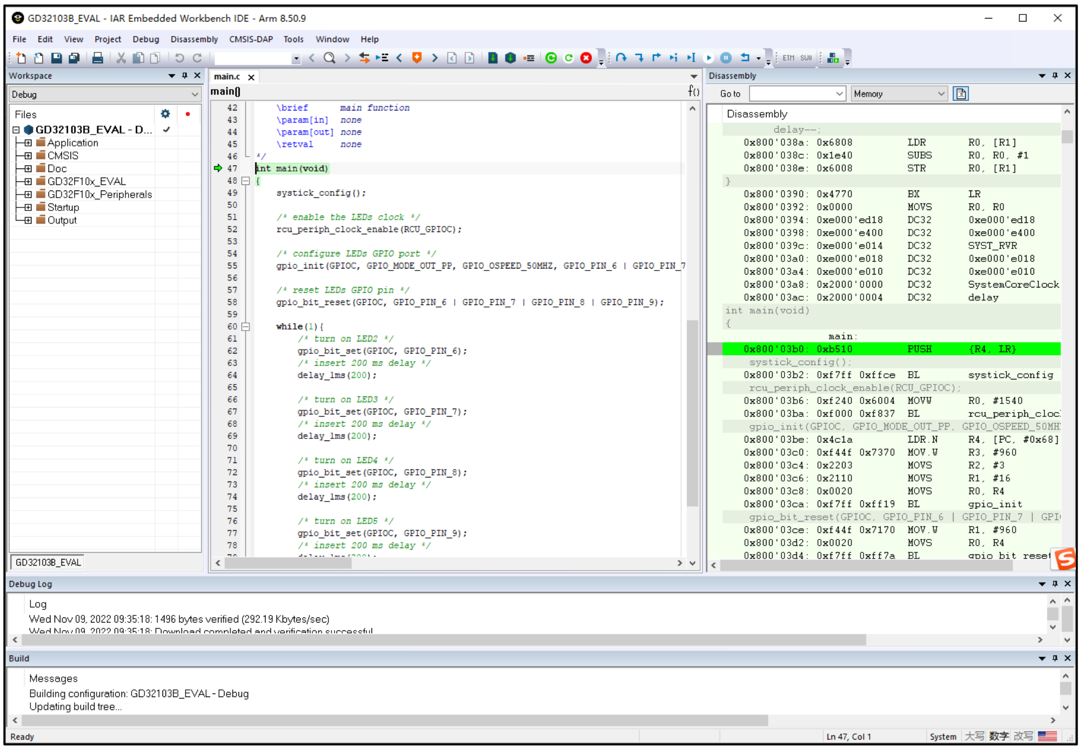
Figure 3-28. KEIL debugging interface



Debugging with IAR (version 8.50 and above)

Complete the IAR configuration according to ***IDE programming*** chapter, click the icon button of "Download and Debug" in the IAR menu bar to enter the debugging interface, as shown in ***Figure 3-29. IAR debugging interface***.

Figure 3-29. IAR debugging interface



Debugging with Eclipse

Complete the Eclipse configuration and debugging according to ***IDE programming*** chapter**.**

3.3.2. SWO function

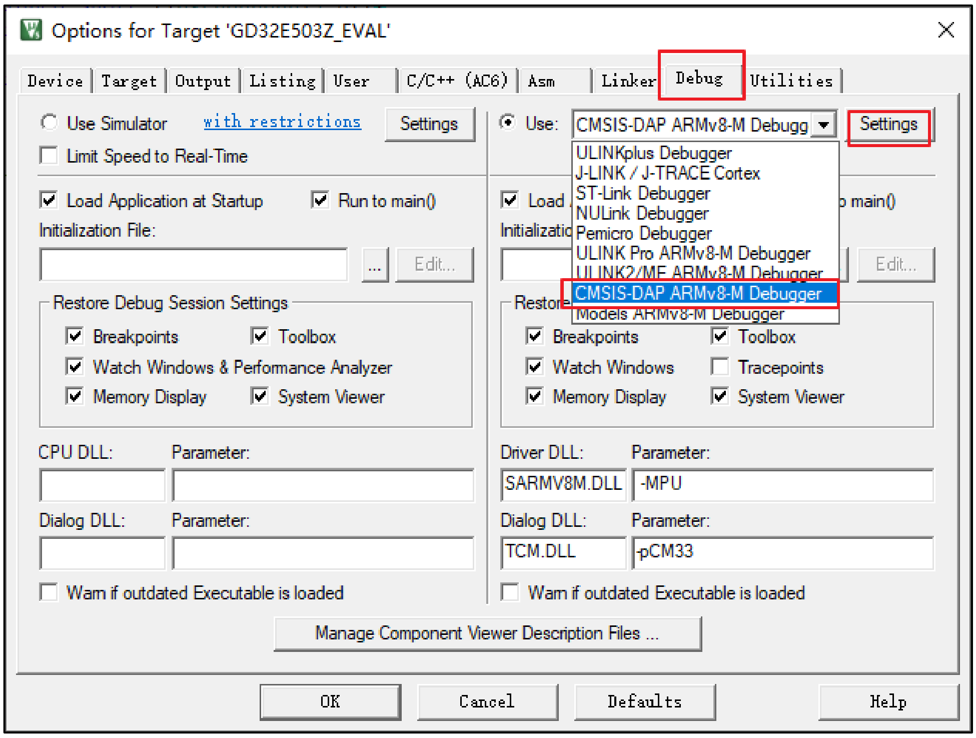
The Serial Wrie Output (SWO) function uses the ITM (Instrumenttaton Trace Marcrocell) module in the Cortex-M kernel to output debugging information in the kernel through the SWO pin of the chip. The connection mode between the burner and the chip is referred to ***Figure 2-4. SWD + SWO interface connection diagram***.

Note: For details about whether the chip supports the SWO function, see the corresponding user manual and datasheet.

SWO configuration in KEIL

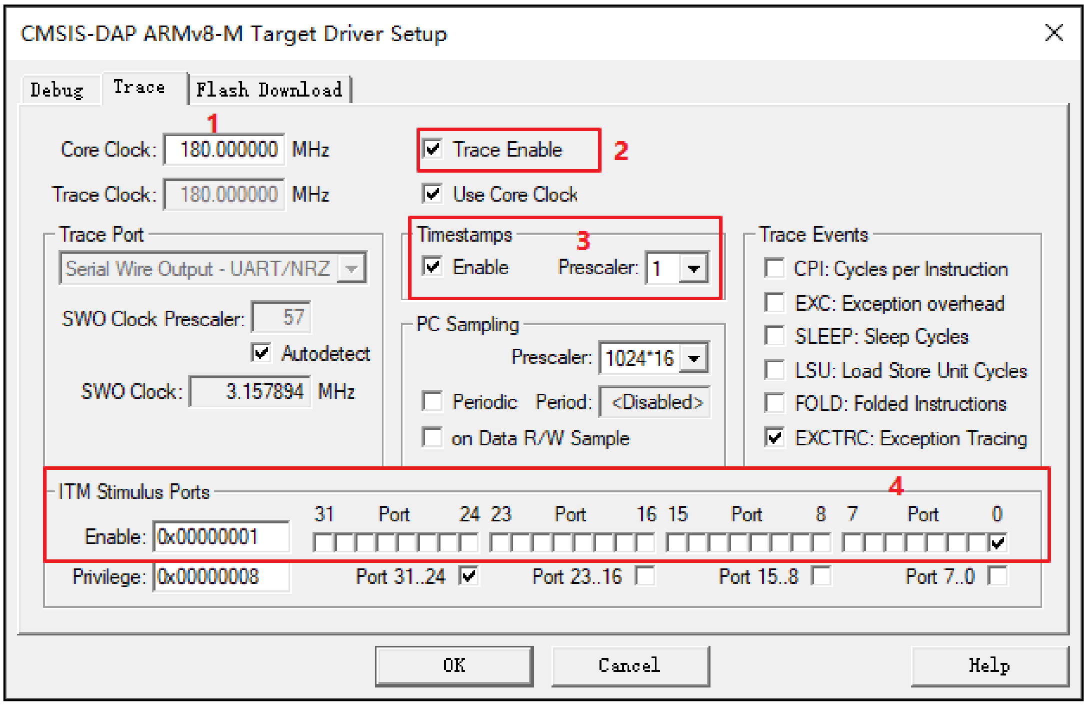
Select the Debug tab in Options for Target and select CMSIS-DAP ARMv8-M Debugger from the drop-down list, referring to ***Figure 3-30. SWO configuration step 1 in KEIL***.

Figure 3-30. SWO configuration step 1 in KEIL



Select the Trace tab in Settings, and the configuration interface is shown in ***Figure 3-31. SWO configuration step 2 in KEIL***.

Figure 3-31. SWO configuration step 2 in KEIL



The Trace pin is enabled in the code. For an MCU with Trace mode configuration, the Trace mode needs to be configured as asynchronous mode, as shown in ***Table 3-4. Trace mode enable***. For details about how to enable Trace mode, see the Debug chapter in the user manuals of each series of MCUs.

Table 3-4. Trace mode enable

```
DBG_CTL |= DBG_CTL_TRACE_IOEN;
```

In the code, the serial printf output is redirected to the ITM output, and the added code is shown in ***Table 3-5. Printf retarget***.

Table 3-5. Printf retarget

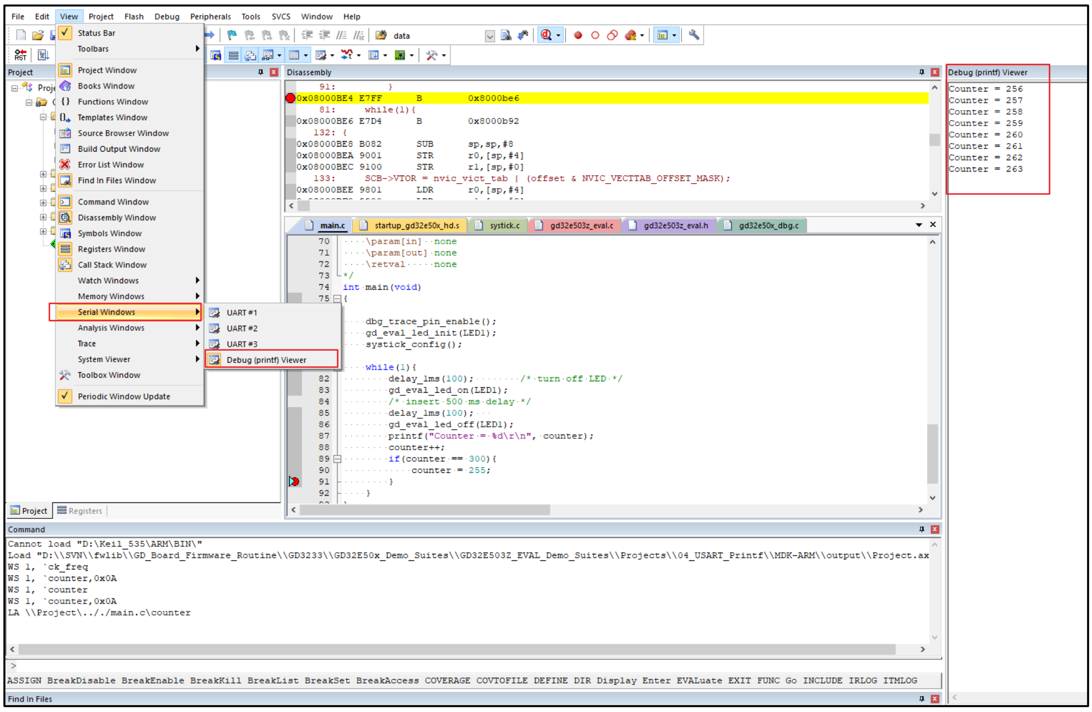
```
#define ITM_Port8(n)      (*((volatile unsigned char *)(0xE0000000+4*n)))
#define ITM_Port16(n)     (*((volatile unsigned short*)(0xE0000000+4*n)))
#define ITM_Port32(n)     (*((volatile unsigned long *)(0xE0000000+4*n)))

#define DEMCR             (*((volatile unsigned long *)(0xE000EDFC)))
#define TRCENA            0x01000000

int fputc(int ch, FILE *f)
{
    if (DEMCR & TRCENA)
    {
        while (ITM_Port32(0) == 0) {};
        ITM_Port8(0) = ch;
    }
  return(ch);
}
```

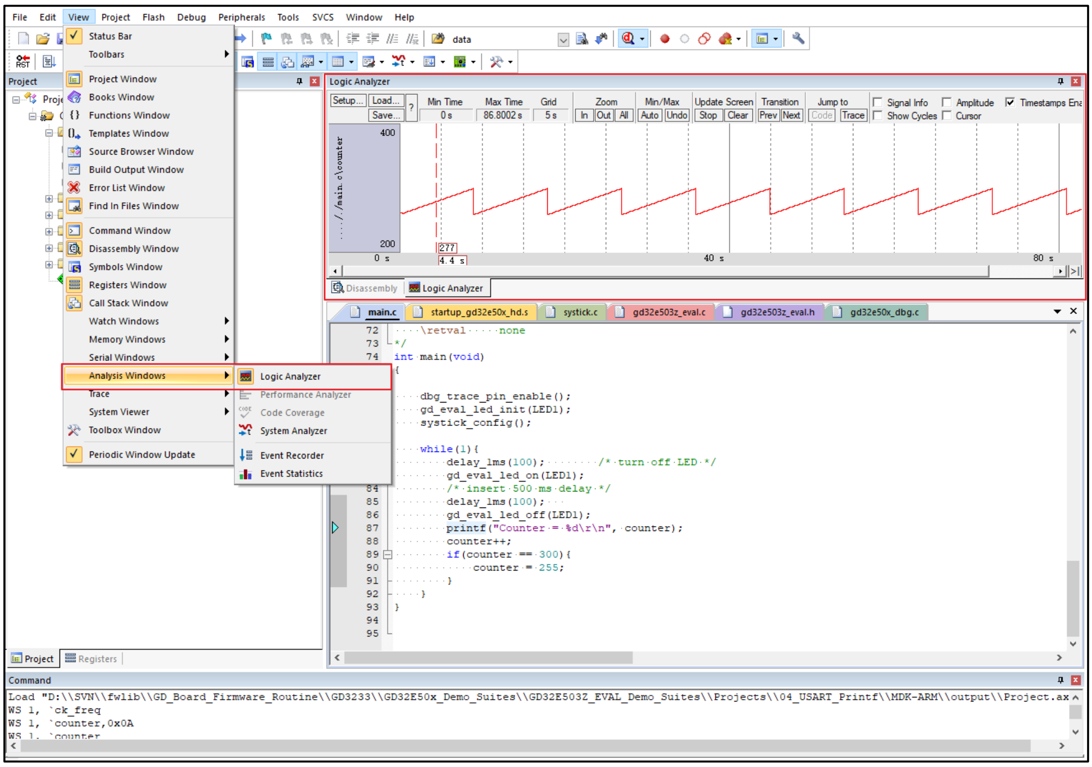
Enter the debugging interface, select "View" -> "Serial Windows" -> "Debug(printf)Viewer", open the serial port printing interface, run the code at full speed, and the printed information will be displayed in the Debug(printf)Viewer window. The Debug(printf)Viewer window in is shown in ***Figure 3-32. Debug (printf) viewer window in KEIL***.

Figure 3-32. Debug (printf) viewer window in KEIL



Enter the debugging interface, choose "View" -> "Analysis Windows" -> "Logic Aanlyzer", open the logic analyzer window, add the variables to be observed, run the code at full speed, and the value of the variable will be displayed in the logic analyzer window through the waveform. The Logic Aanlyzer window is shown in *Figure 3-33. Logical Analyzer window in KEIL*.

Figure 3-33. Logical Analyzer window in KEIL



3.3.3. Dual-core microcontroller debugging

This user guide takes GD32A72XX as an example to introduce how to perform dual-core debugging in different IDEs.

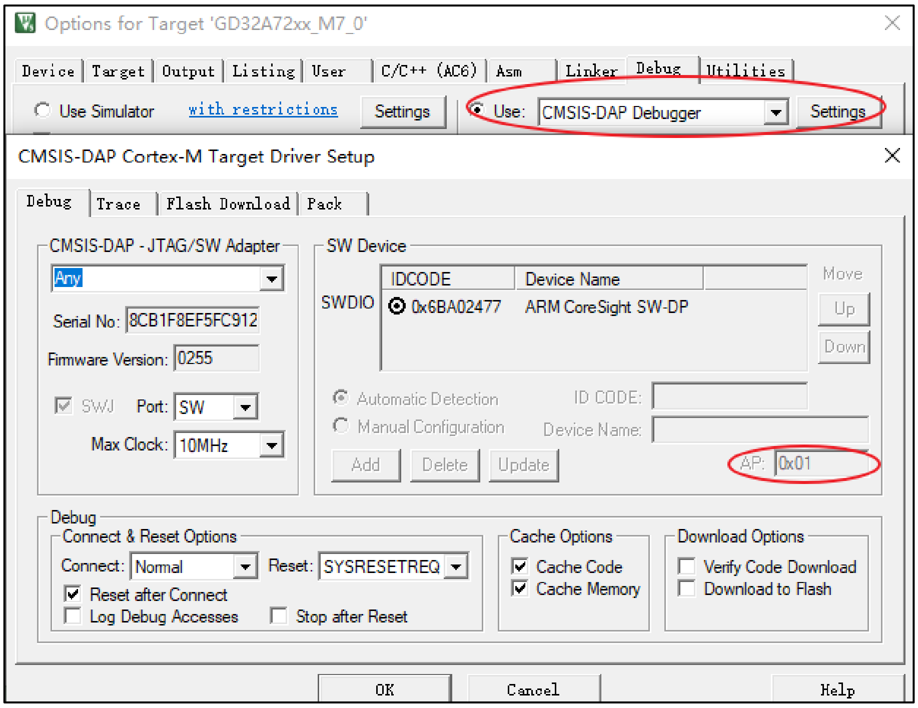
Debugging with KEIL (version 5.30 and above)

1. Download the IVT project to the target microcontroller. Open the IVT project and configure it to use the CMSIS-DAP Debugger, with the code settings as shown in ***Figure 3-34. KEIL IVT config***. Configure the BCW to 7 to release the cores CM7_0, CM7_1, and CM23. By default, only CM7_0 is active on dual-core microcontrollers. After the IVT is downloaded, the microcontroller must be reset.

Figure 3-34. KEIL IVT configuration

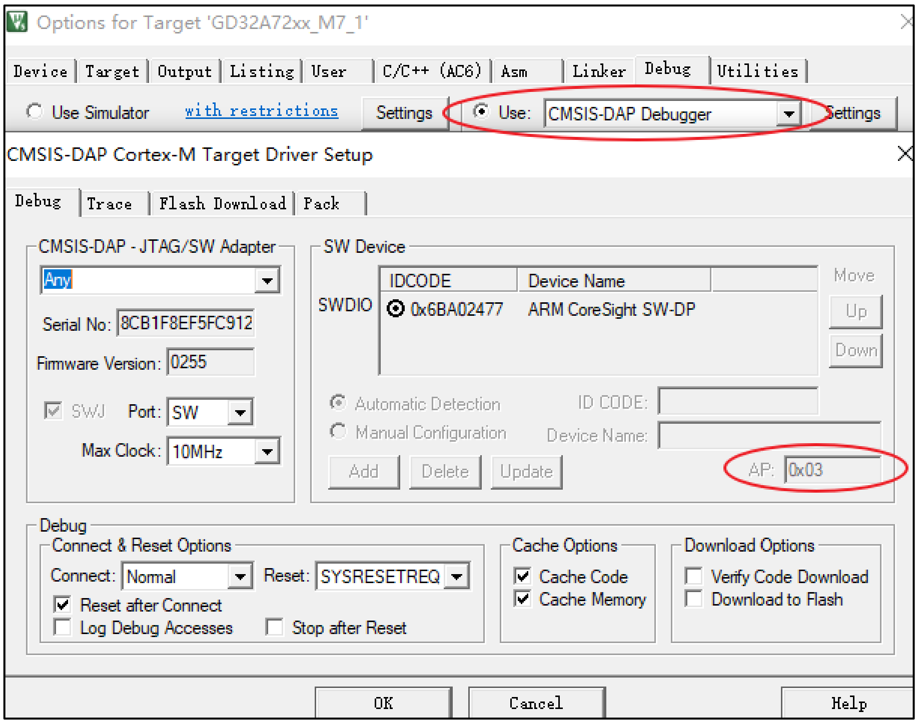
2. Open the M7_0 project and configure it to use the CMSIS-DAP Debugger, with the AP item set to 0x01. Compile and download the M7_0 project to the target chip. (If the pack has been installed and selected correctly, the AP is automatically configured as 0x01 in the pack, and users do not need to configure it manually.)

Figure 3-35. KEIL core0 debug setting



3. Open the M7_1 project and configure it to use the CMSIS-DAP Debugger, with the AP item set to 0x03. Compile and download the M7_1 project to the target chip. (If the pack has been installed and selected correctly, the AP is automatically configured as 0x03 in the pack, and users do not need to configure it manually.)

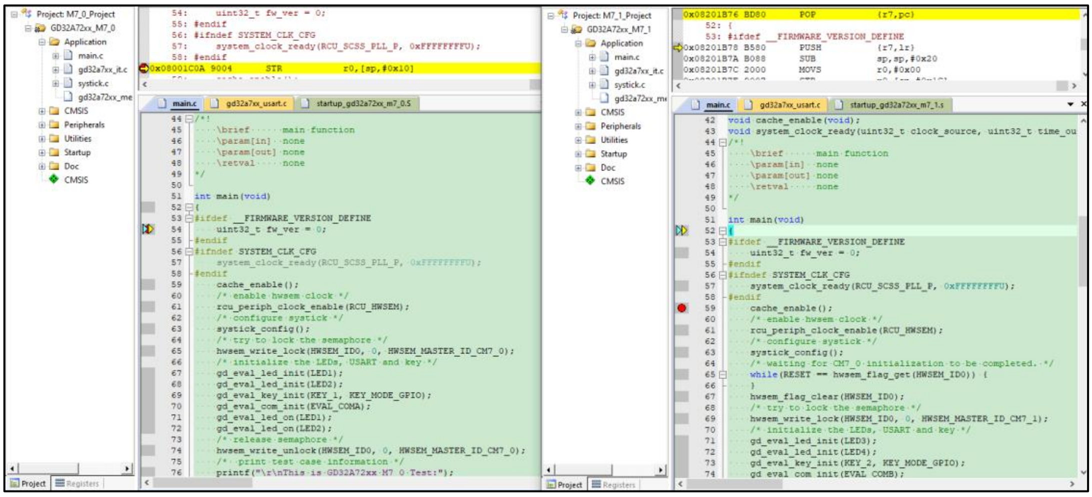
Figure 3-36. KEIL core1 debug setting



4. Launch the debugging of the main core CM7_0 project.

5. the debugging of the slave core CM7_1 project. Dual-core debugging can be performed simultaneously.

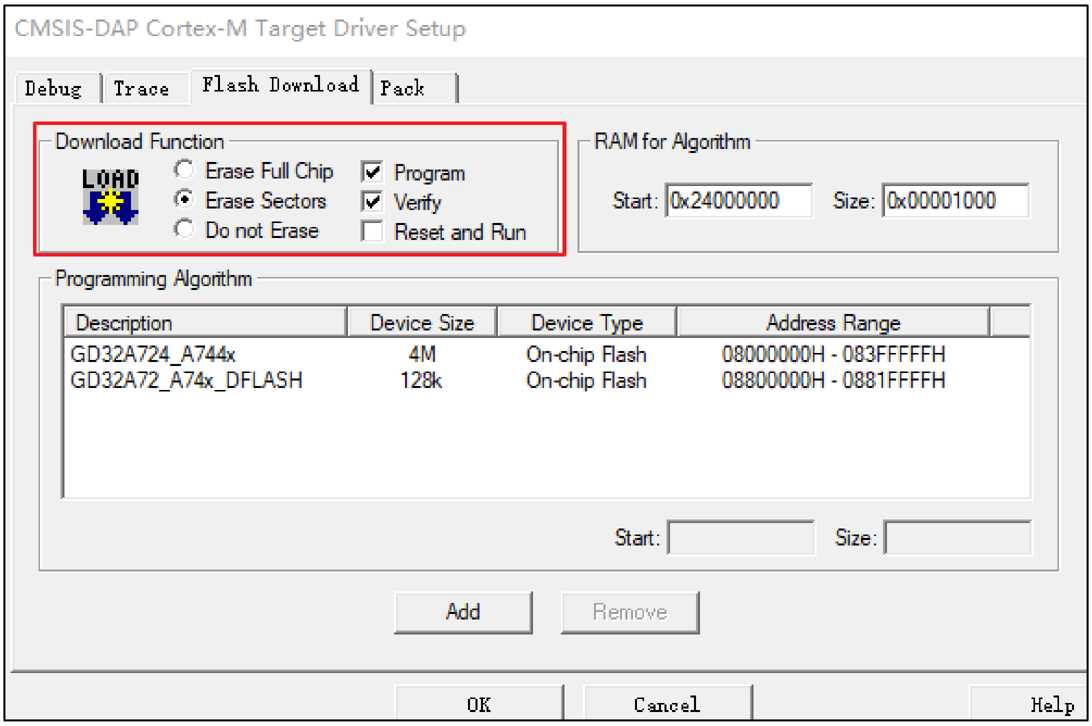
Figure 3-37. KEIL dual-core debugging interface



Note:

1. Configure the "Flash Download" as shown in ***Figure 3-38. KEIL flash download configuration***。

Figure 3-38. KEIL flash download configuration



2. When debugging and downloading to different cores, it is necessary to correctly configure the AP (Access Port).

Debugging with IAR (version 9.50 and above)

1. Download the IVT project to the target microcontroller. Open the IVT project and configure it to use the CMSIS-DAP Debugger, with the code settings as shown in ***Figure 3-39. IAR IVT configuration***. Configure the BCW to 7 to release the cores CM7_0, CM7_1, and CM23. By default, only CM7_0 is active on dual-core microcontrollers. After the IVT is downloaded, the microcontroller must be reset.

Figure 3-39. IAR IVT configuration

```
#include "gd32a72xx_IVT_config.h"
#include "gd32a72xx_memory_map.h"

extern const ivt_config_srtuct gd32a72x_ivt;

#if defined (__ICCARM__) 
__root const ivt_config_srtuct gd32a72xx_ivt __attribute__((section(".IVT_HEADER"),used)) = {
#else 
const ivt_config_srtuct gd32a72x_ivt __attribute__((section(".IVT_HEADER"),used)) = {
#endif /* defined ( __ICCARM__ ) */
        .tag         = 0x5AA55AA5U,
        .bcw_config  = 7U,
        .boot_addr_0 = CM7_0_FLASH_START,
        .boot_addr_1 = CM7_1_FLASH_START,
        .boot_addr_2 = 0x08880000U,
        .lc_addr     = 0U
};
```

2. Open the M7_0 project, set the “Debugger” for the CM7_0 project to CMSIS DAP, and configure the “Multicore” option as shown in ***Figure 3-40. IAR core0 debugger multicore***

setting. Additionally, set the reset method to Core reset, and specify the core as CM7_0.

Figure 3-40. IAR core0 debugger multicore setting

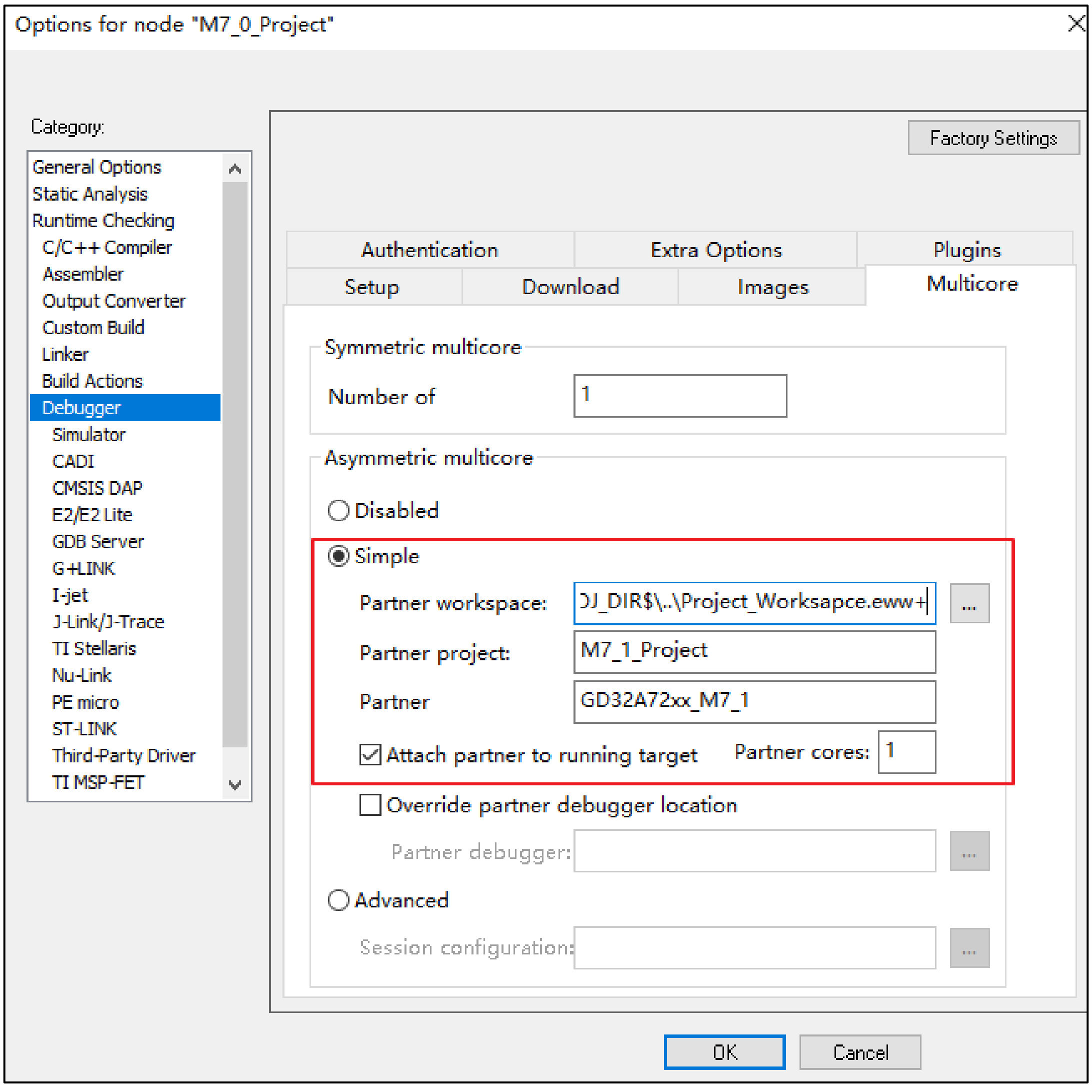


Figure 3-41. IAR core0 debugger reset setting

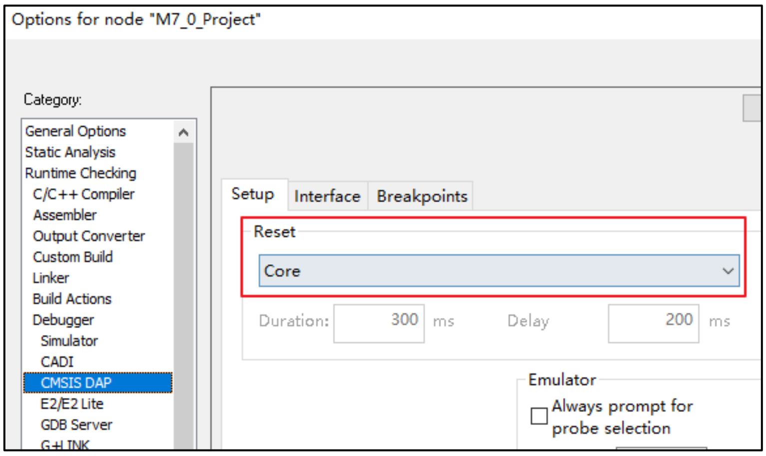
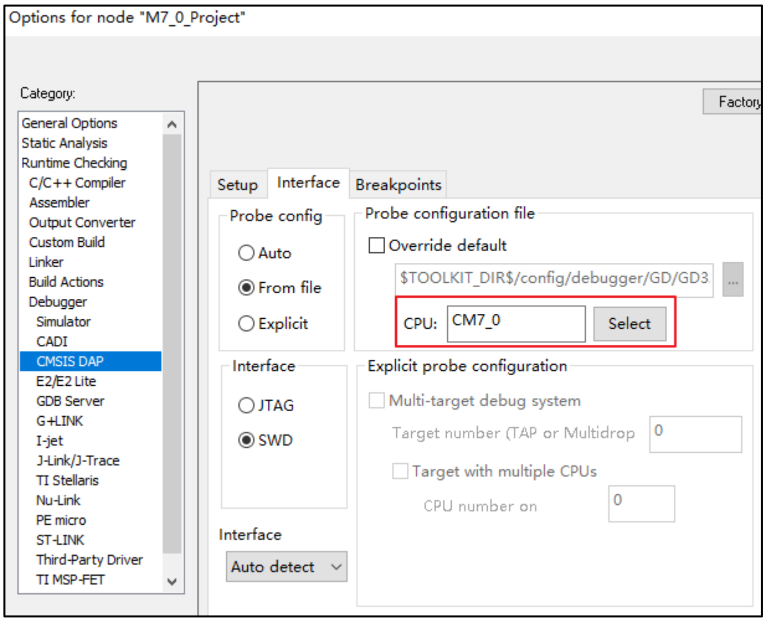


Figure 3-42. IAR core0 debugger interface setting



3. Open the M7_1 project, set the "Debugger" for the CM7_1 project to CMSIS DAP, and configure the reset method to Core reset, specifying the core as CM7_1.

Figure 3-43. IAR core1 debugger reset setting

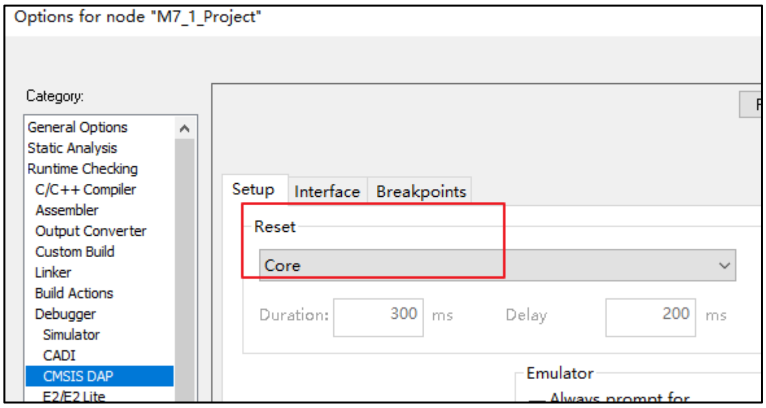
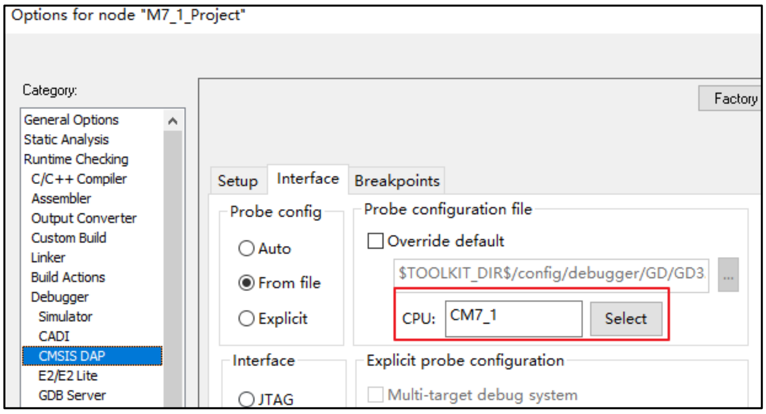


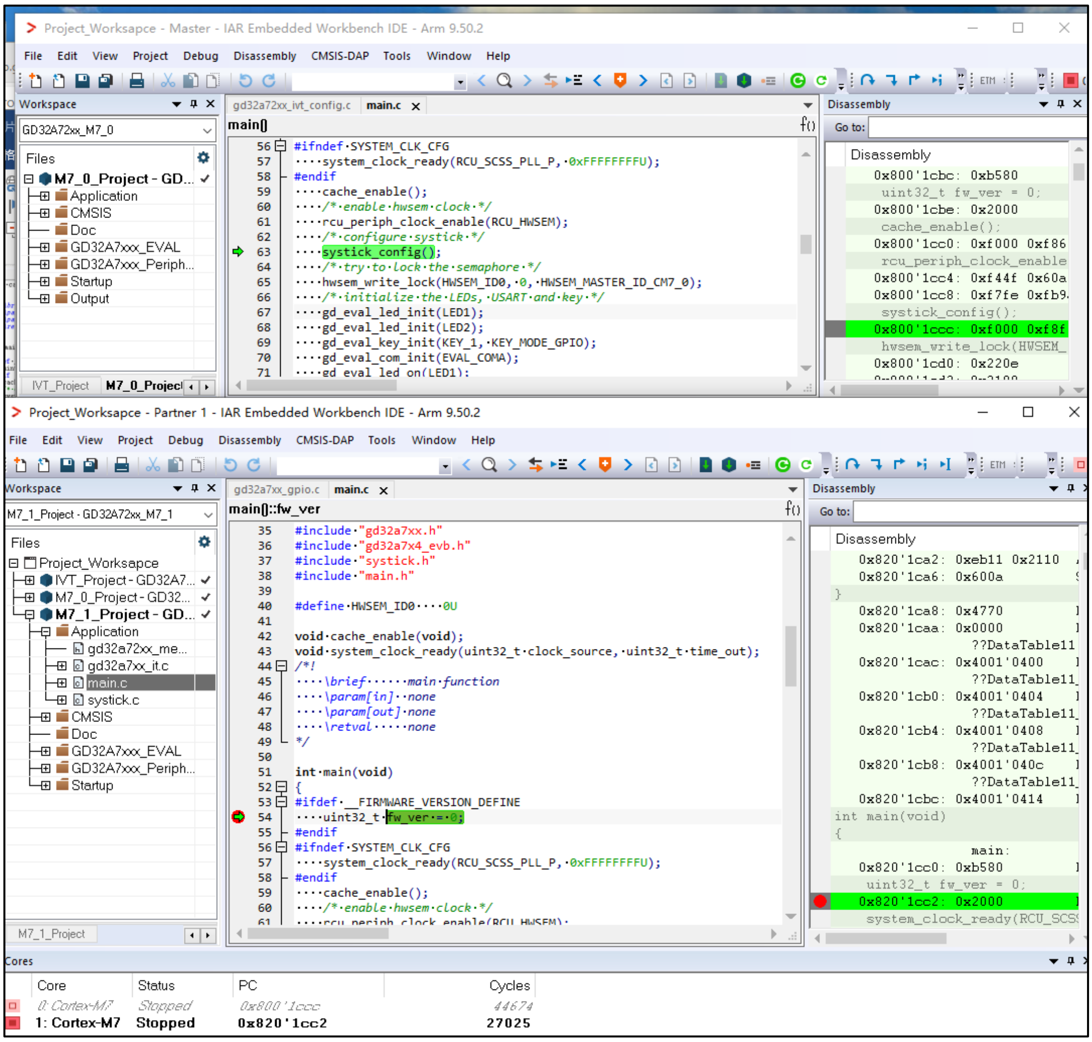
Figure 3-44. IAR core1 debugger interface setting



4. Download the M7_0 project and the M7_1 project to the target chip.

6. Start debugging the main core CM7_0 project, and simultaneously start debugging the slave core CM7_1 project. Dual-core debugging can be performed simultaneously.

Figure 3-45. IAR dual-core debugging interface



Debugging with Eclipse

1. Download the IVT to the target chip. The code configuration is as shown in ***Figure 3-46. Eclipse IVT configuration***. Set the BCW to 7 to release CM7_0, CM7_1, and CM23. By default, only CM7_0 is active on dual-core microcontrollers. After the IVT is downloaded, the microcontroller must be reset.

Figure 3-46. Eclipse IVT configuration

```
#include "gd32a72xx_IVT_config.h"
#include "gd32a72xx_memory_map.h"

extern const ivt_config_srtuct gd32a74x_ivt;

const ivt_config_srtuct gd32a74x_ivt = {
        .tag         = 0x5AA55AA5U,
        .bcw_config  = 7U,
        .boot_addr_0 = CM7_0_FLASH_START,
        .boot_addr_1 = CM7_1_FLASH_START,
        .boot_addr_2 = DATA_FLASH_START,
        .lc_addr     = 0U
};
```

Figure 3-47. Eclipse IVT debug\run configuration – Main tab

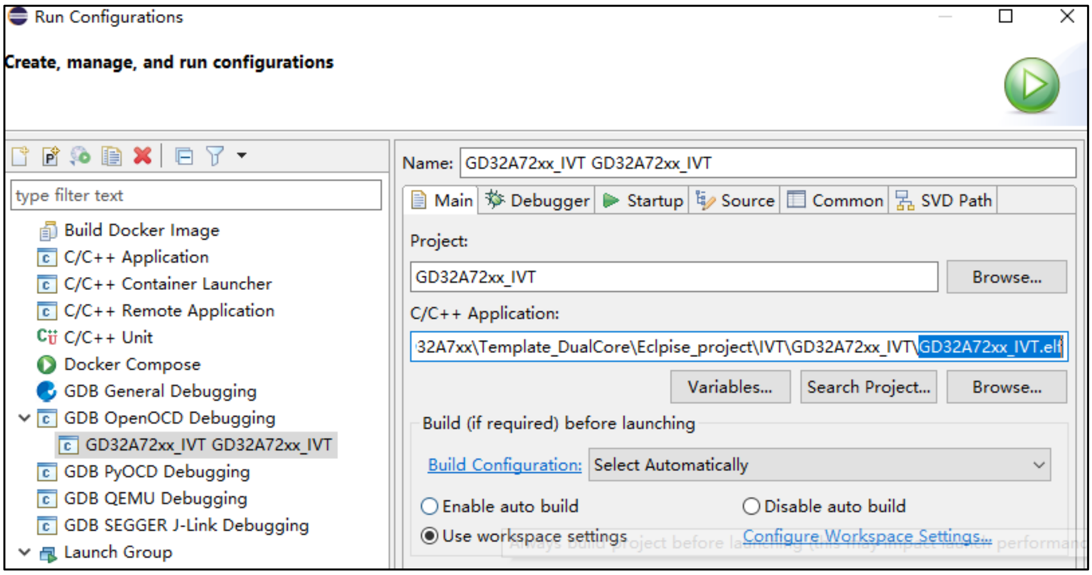
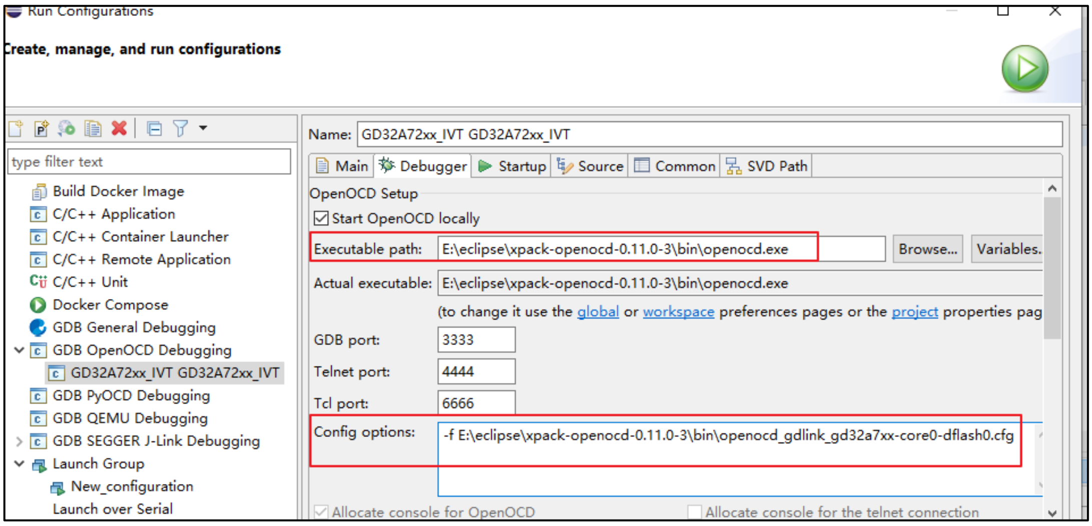


Figure 3-48. Eclipse IVT debug\run configuration – Debugger tab



2. Configure the external tool OpenOCD.

Figure 3-49. Eclipse external tool configuration

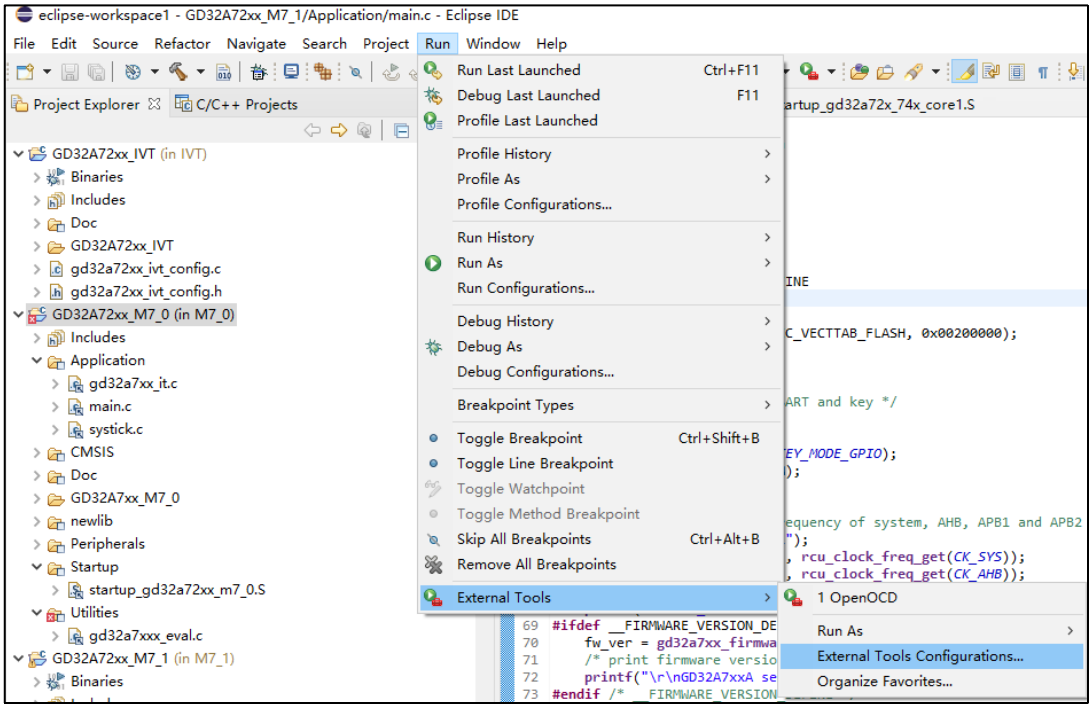
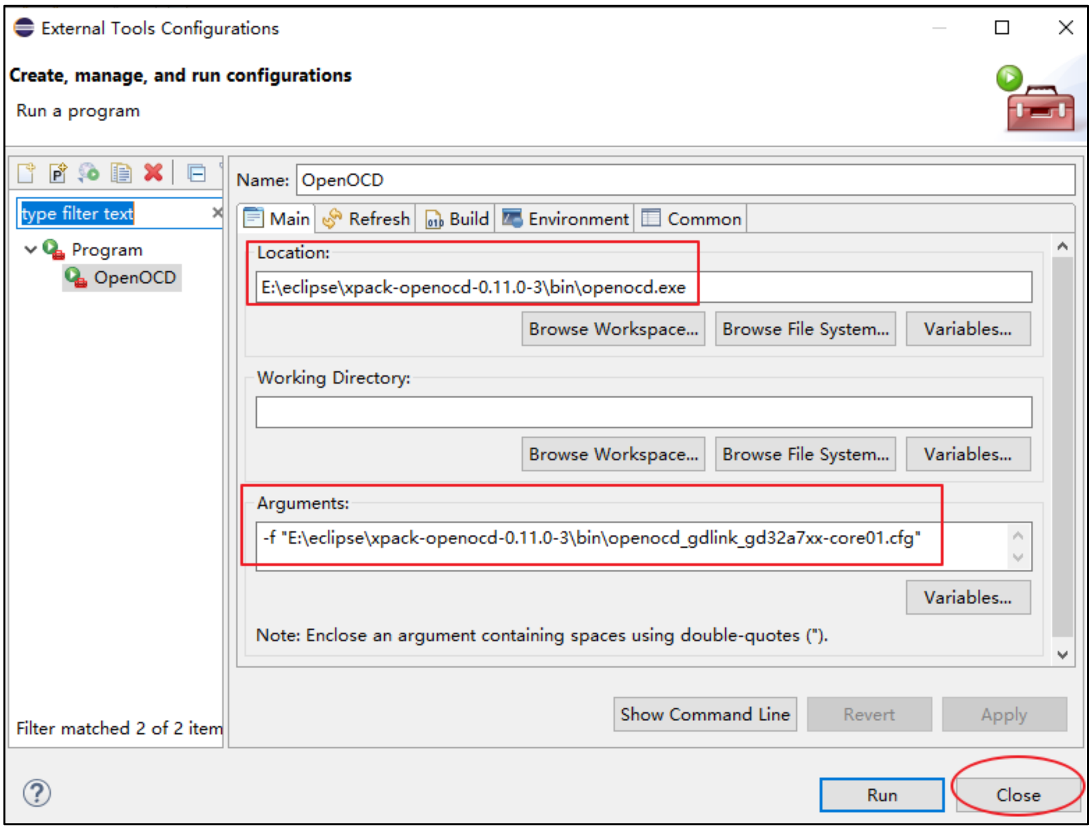


Figure 3-50. Eclipse external tool configuration – OpenOCD



3. Configure the GDB port 3333 for debugging the CM7_0 core and the GDB port 3334 for debugging the CM7_1 core.

Figure 3-51. Eclipse core0 debug configuration – Main tab

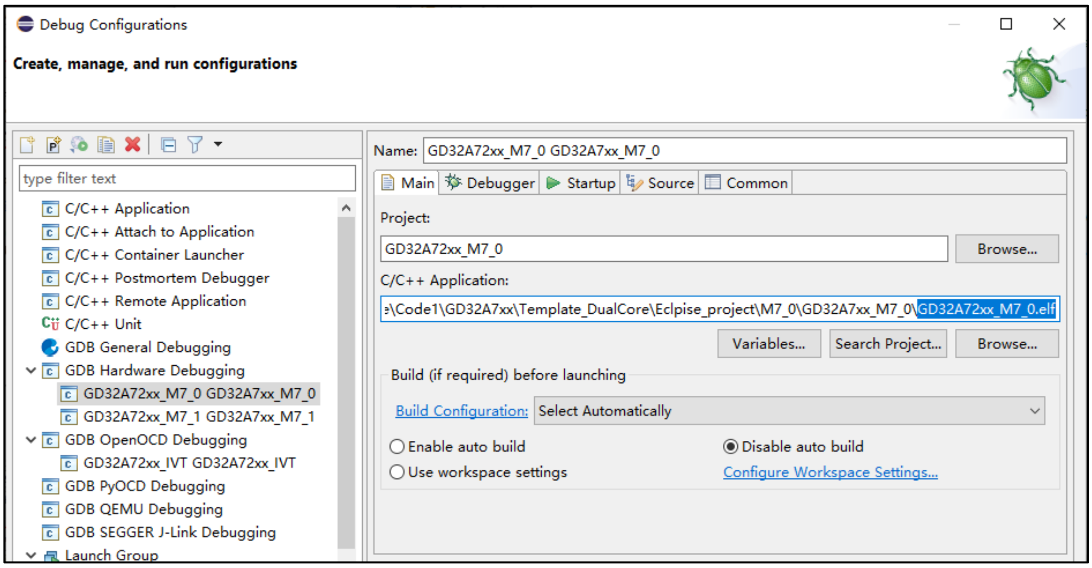


Figure 3-52. Eclipse core0 debug configuration – Debugger tab

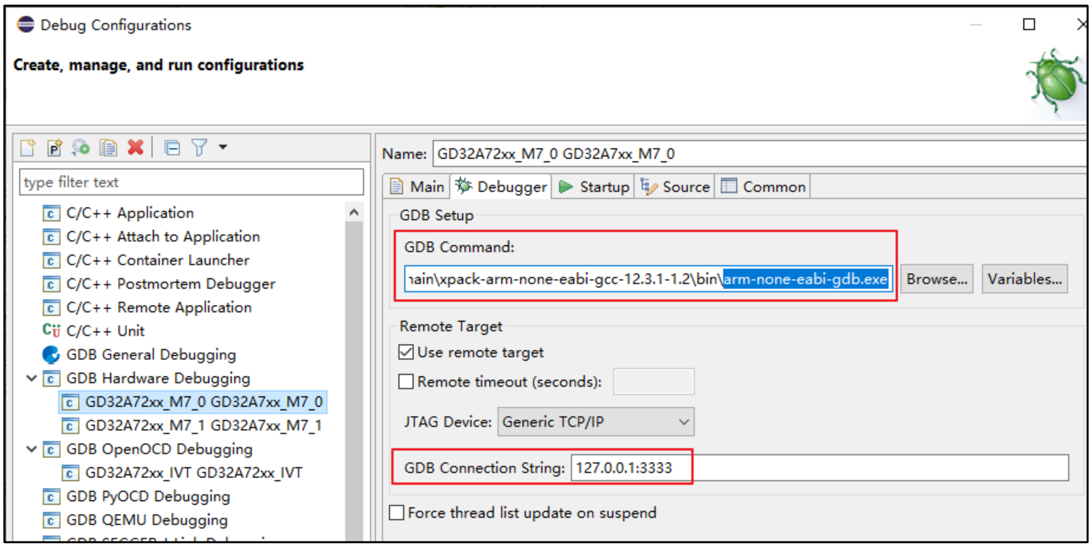


Figure 3-53. Eclipse core1 debug configuration – Main tab

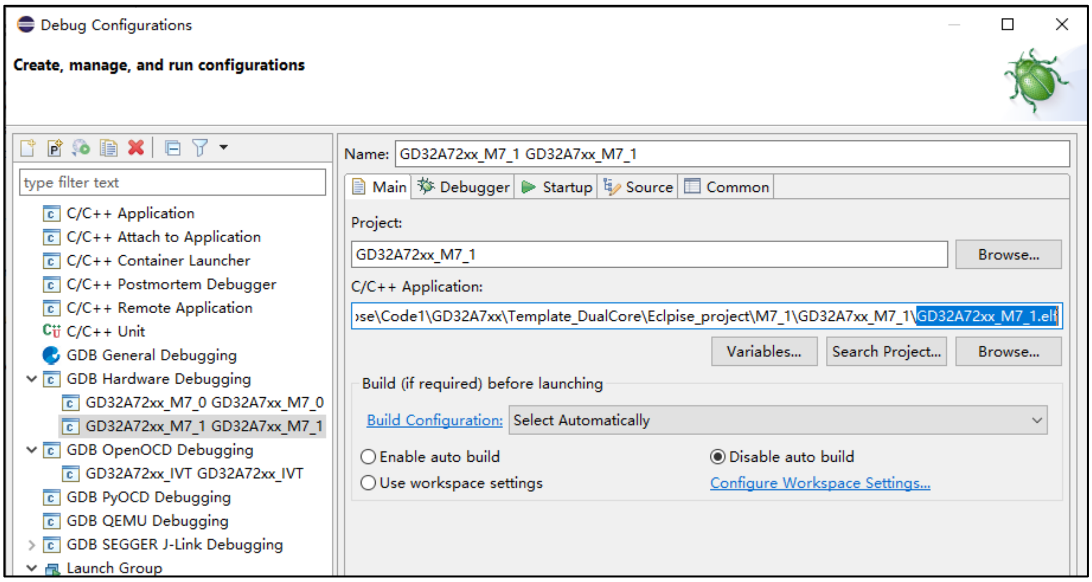
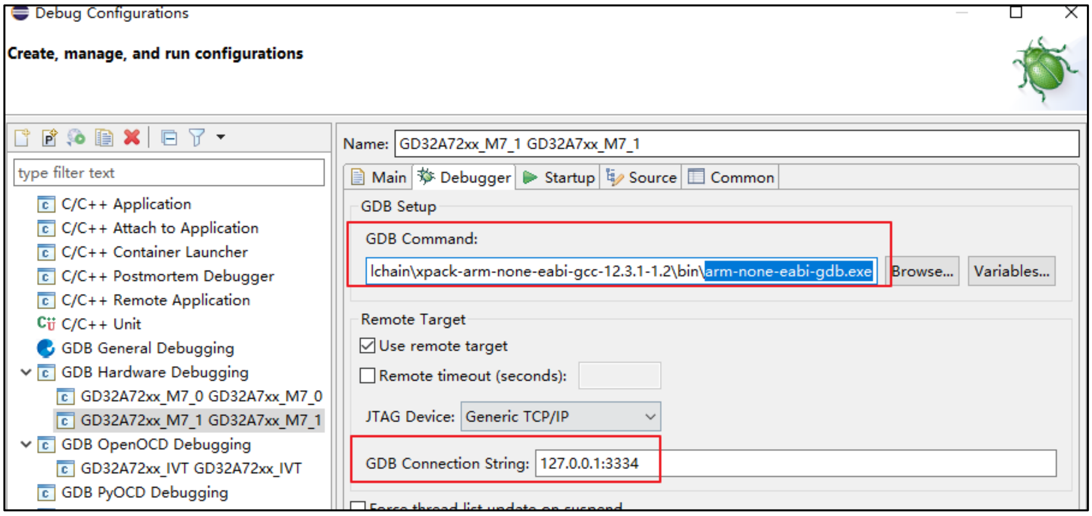


Figure 3-54. Eclipse core1 debug configuration – Debugger tab



4. Combine the launch of OpenOCD and GDB in Eclipse.

Figure 3-55. Eclipse lauch group configuration – Program option

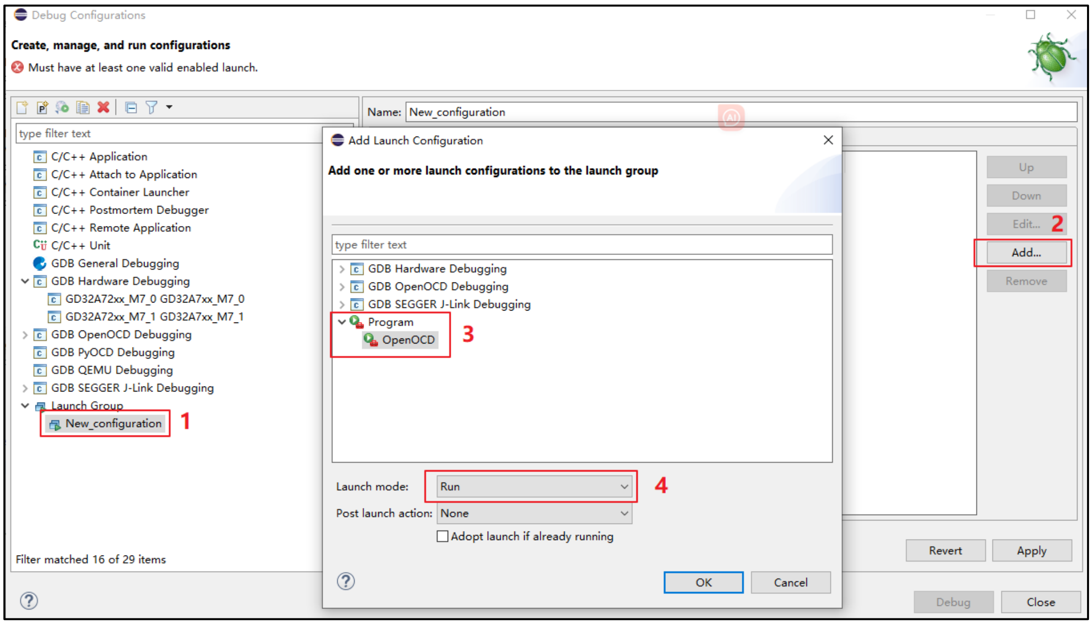
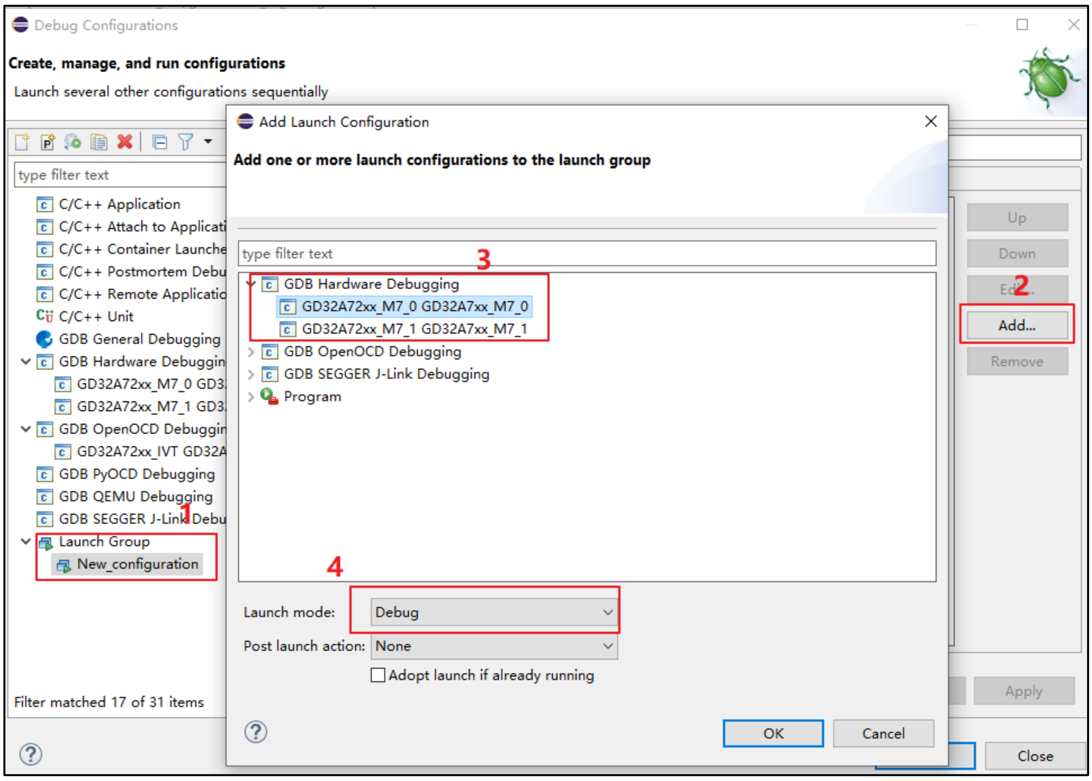


Figure 3-56. Eclipse lauch group configuration – GDB Hardware Debugging option



After the configuration is completed, as shown in ***Figure 3-57. Eclipse lauch group configuration***. Click the "Debug" button to enter dual-core debugging.

Figure 3-57. Eclipse lauch group configuration

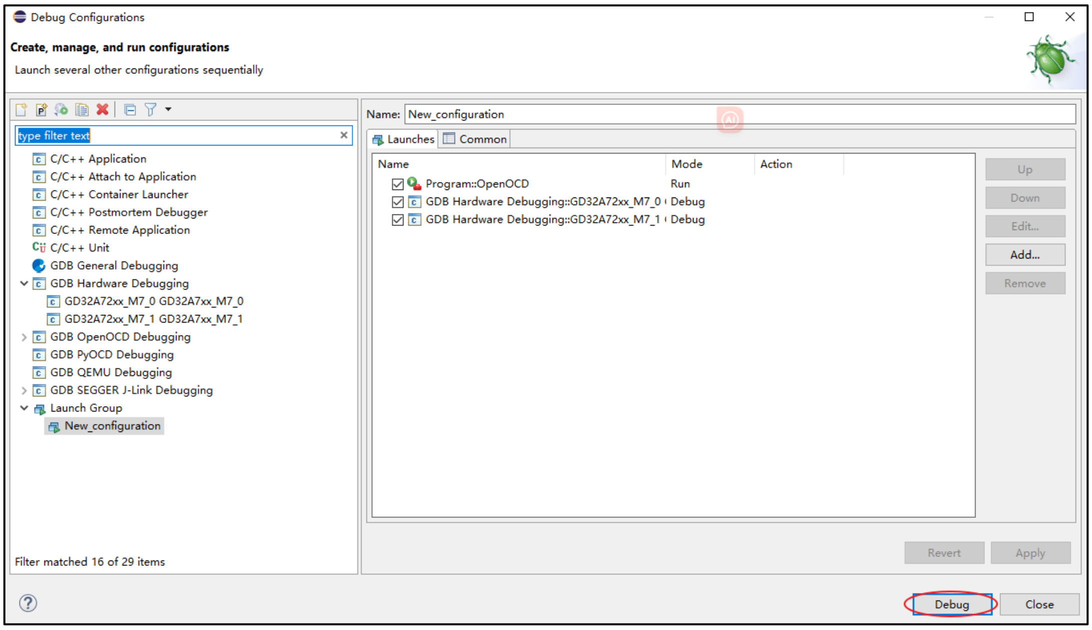
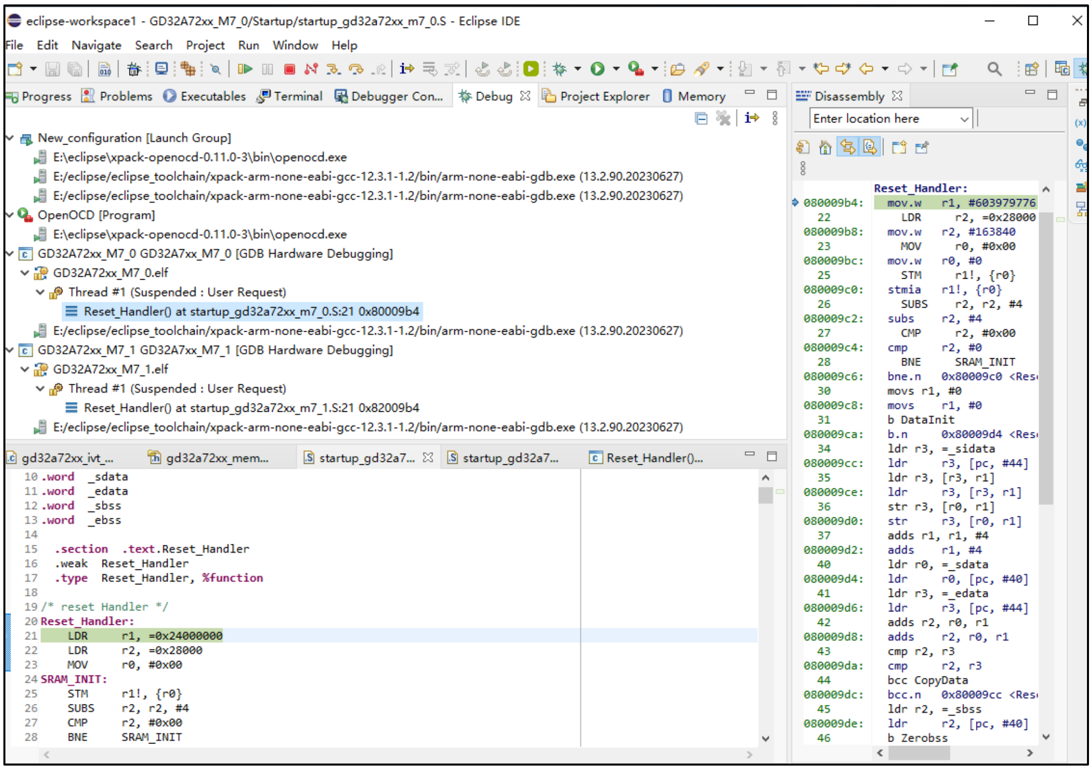


Figure 3-58. Eclipse dual-core debugging interface



Note:

When using OpenOCD for downloading and debugging in Eclipse, user need to configure the cfg file correctly according to the core and address, and user need to correctly select the corresponding flash programming algorithm.

3.4. Virtual serial port printing

When the GD-Link V2 USB is inserted into the PC port, a USB serial device will appear on the PC Device Manager port (COM and LPT) interface (there is no driver for WIN10 system, and the corresponding driver should be installed for win7 system), as shown in ***Figure 3-59. USB serial device***, refer to ***Figure 2-5. Serial interface connection diagram***. Connect GD-Link V2 to the serial port pin hardware of the target chip, configure the correct serial port baud rate and other information in the serial port debugging assistant, and write the data to be sent to the target MCU serial port receiver through the serial port debugging assistant. The target MCU can also print the information to be printed through the USB port of the burner to the upper computer interface of the serial debugging assistant through the serial port transmitter and display it, as shown in ***Figure 3-60. USB virtual serial printing***.

Figure 3-59. USB serial device

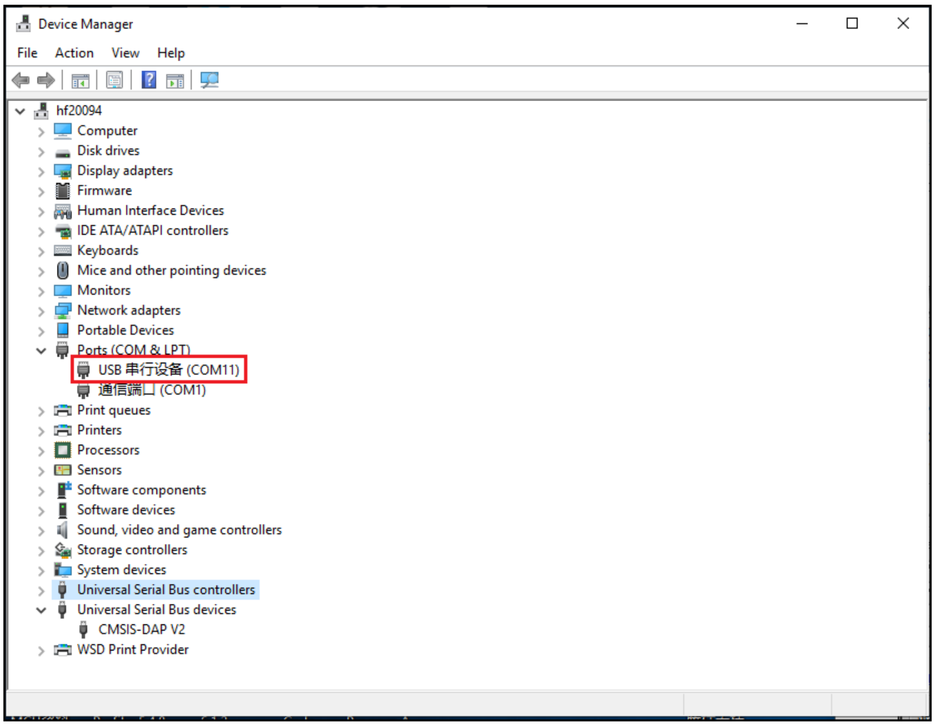
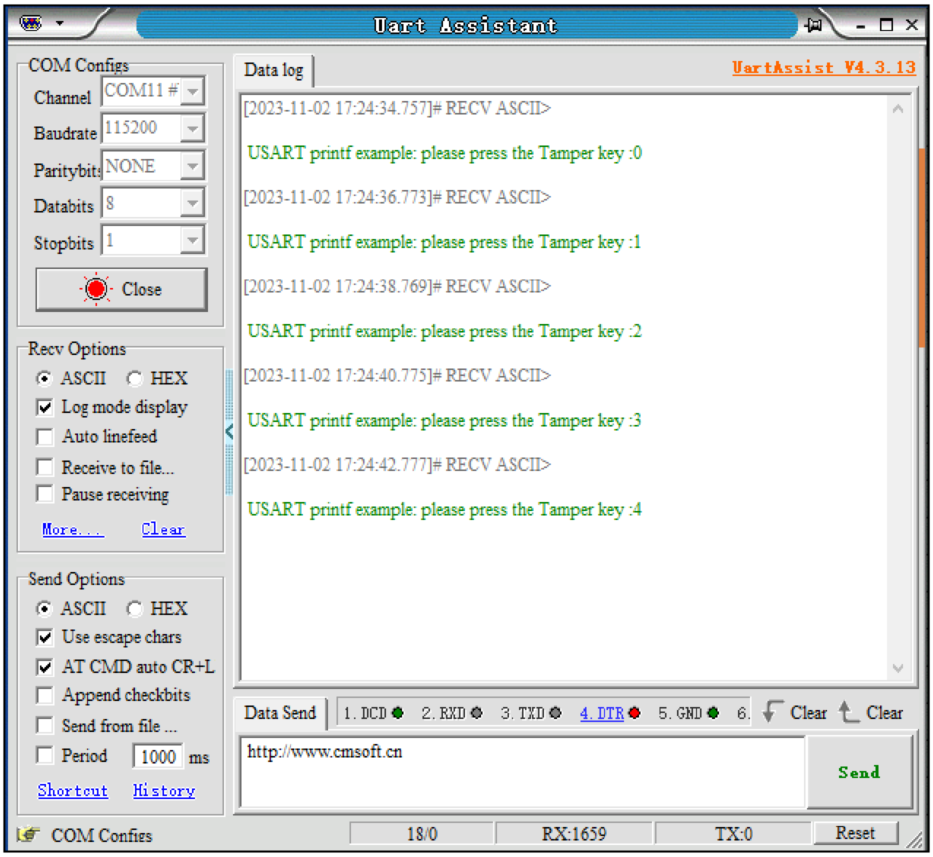


Figure 3-60. USB virtual serial printing



4. Q&A

4.1. Unable to recognize GD-Link V2 device

3IN1 GD-Link V2 requires driver installation on the WIN7 system. For WIN10 and WIN11, no driver installation is needed. Normally, it should appear as follows in the WIN10 Device Manager.

If GD-Link V2 cannot be recognized successfully, such as in ***Figure 4-1. Unable to recognize 3IN1 GD-Link V2 device in GD-Link Programmer***, where it appears as "unknown" in the GD-Link Programmer or cannot be identified in IDEs like Keil, please follow the steps below to attempt to fix:

1. Check whether USB HUB is being used. If USB HUB is used, remove it and connect directly to the computer.
2. Uninstall the driver and reconnect the device. Refer to ***Figure 4-3. Uninstall the driver***.
3. Modify the hardware resistance, reconnect the device, refer to ***Hardware issue***, and repeat step 2.
4. If none of the above methods resolve the issue, please contact the FAE.

If the device is not recognized or appears as an unknown device in GD-Link Programmer or IDE (GD-Link Programmer failed to correctly recognize the device serial number of GDLink.), please also uninstall the driver, and reconnect the device.

Figure 4-1. Unable to recognize 3IN1 GD-Link V2 device in GD-Link Programmer

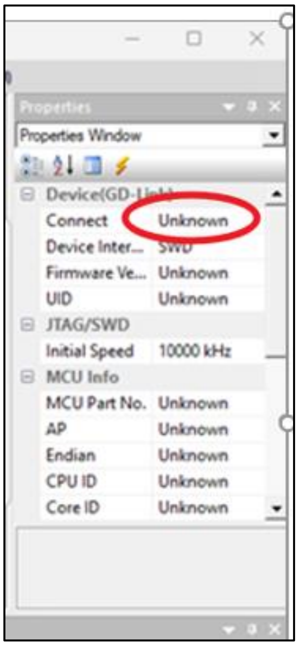


Figure 4-2. 3IN1 GD-Link V2 in Device Manager

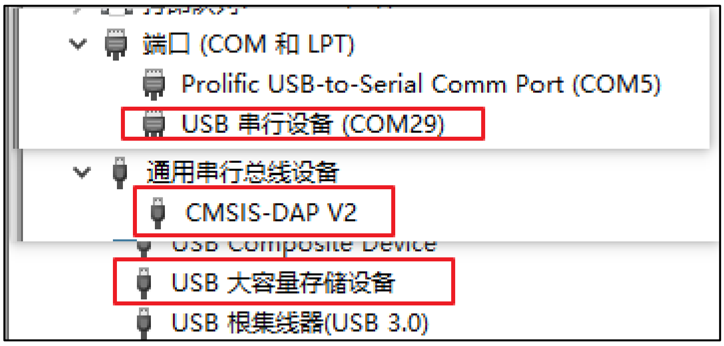
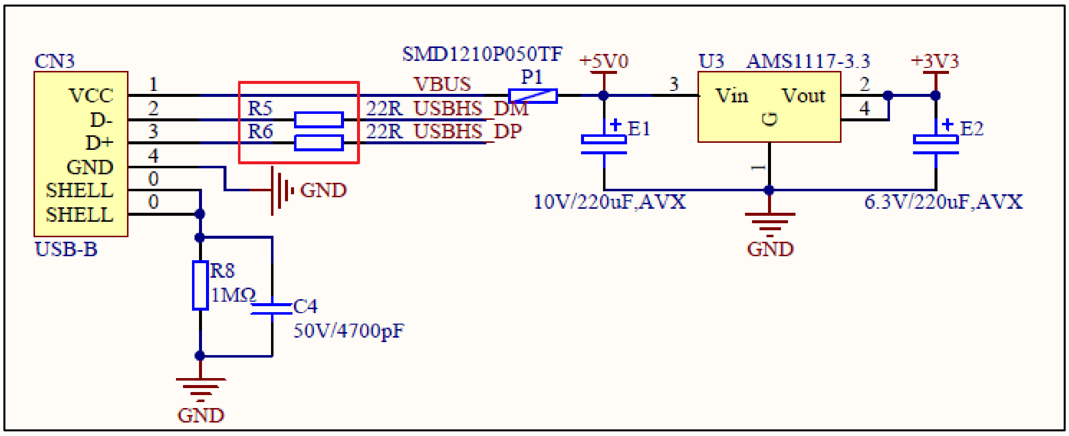


Figure 4-3. Uninstall the driver

Hardware issue

The two resistors of the USBHS_DM and USBHS_DP pins should use 0Ω resistors instead of 22Ω ones. Please remove these two resistors. This hardware issue may cause USB signal instability, and Windows may stop trusting the current driver.

Figure 4-4. Hardware issue



4.2. Unable to debug using with OpenOCD, when multiple CMSIS-DAP devices are connected to the PC

When multiple CMSIS-DAP devices are connected to the PC, the serial number of the GD-Link must be specified in the CFG file when debugging with OpenOCD software. The serial number can be obtained using tools such as GDLink_CLI.

Figure 4-5. GD-Link SN

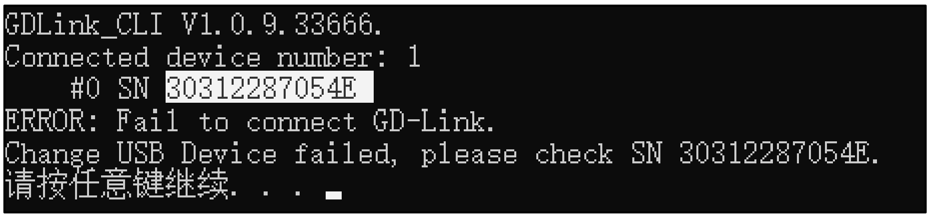
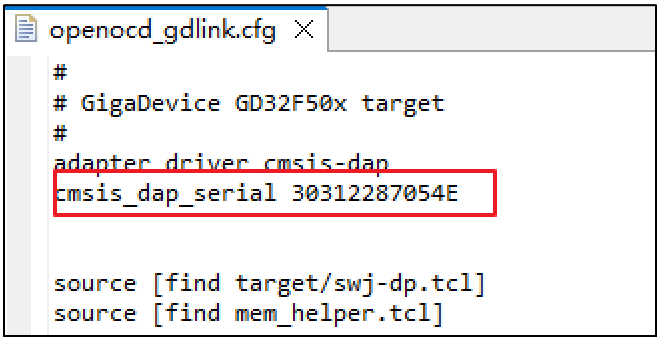


Figure 4-6. OpenOCD cfg file



4.3. Can I use a USB HUB to connect GD-Link and the computer

We strongly recommend against using a USB HUB to connect GD-Link and the computer. If you must use a USB HUB, please choose a high-quality USB HUB with independent power supply and closely monitor the connection stability during use. If connection issues occur, first try direct connection to troubleshoot the problem.

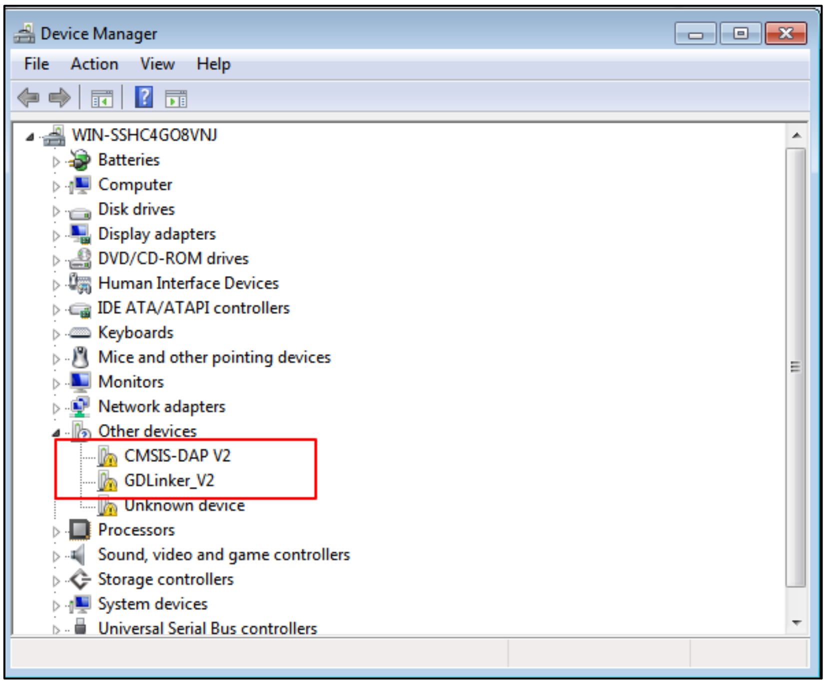
4.4. How to install drivers on a Windows 7 computer

1. Open the GD32MCU official website and download the Gigadevice GD-Link Win7 Driver file. Address: [***https://www.gd32mcu.com/cn/download?kw=GD-Link&lan=cn***](https://www.gd32mcu.com/cn/download?kw=GD-Link&lan=cn).
2. Insert GD-Link V2 into the computer's USB port, and Open Device Manager on computer.
3. In the "Other devices" category, two unrecognized devices can be seen. As shown in

Figure 4-7. The two unrecognized devices.

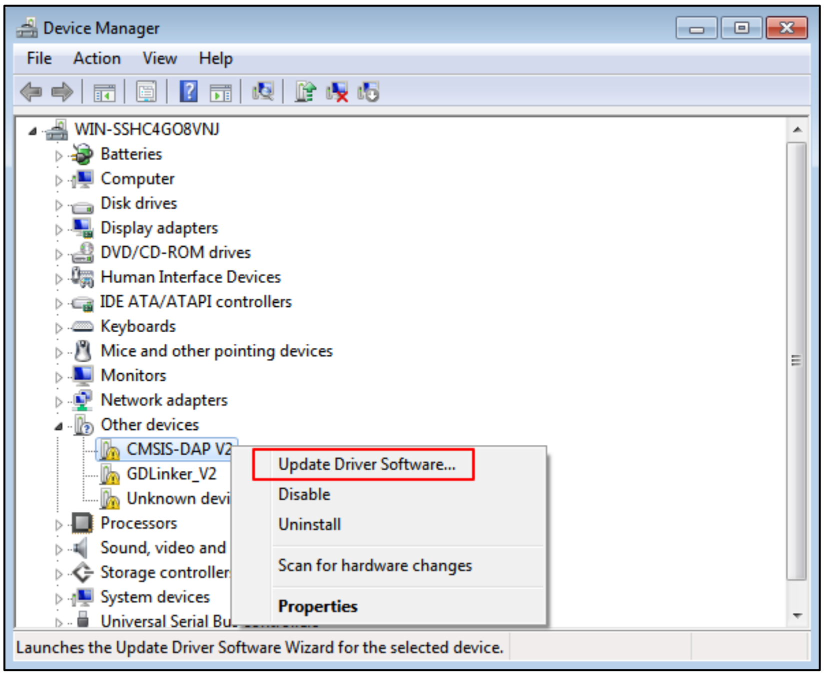
- CMSIS-DAP V2 - Debugger interface device
- GDLinker_V2 - Virtual serial port device

Figure 4-7. The two unrecognized devices



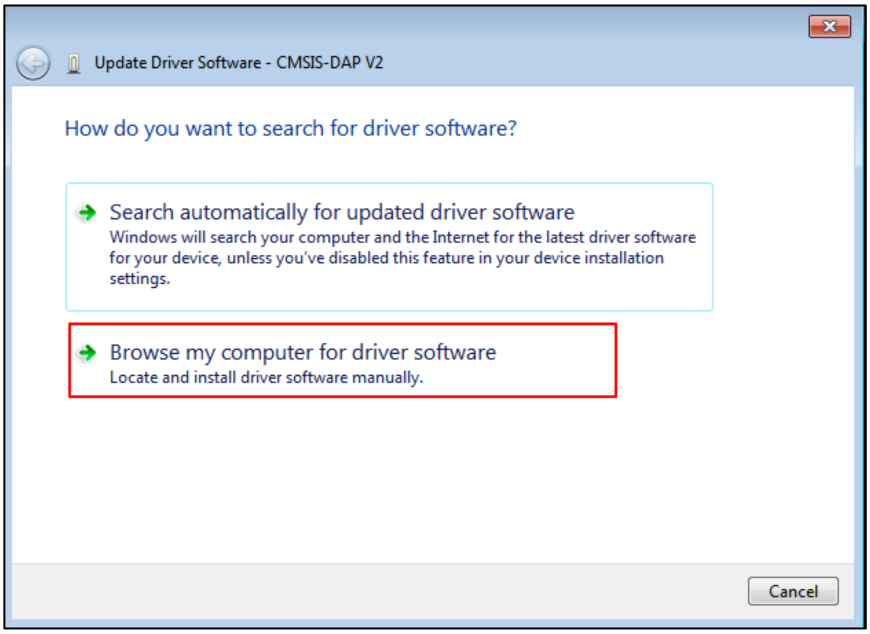
4. Right-click "CMSIS-DAP V2" → Select "Update Driver Software...". As shown in *Figure 4-8. Step 1: Install the driver*.

Figure 4-8. Step 1: Install the driver



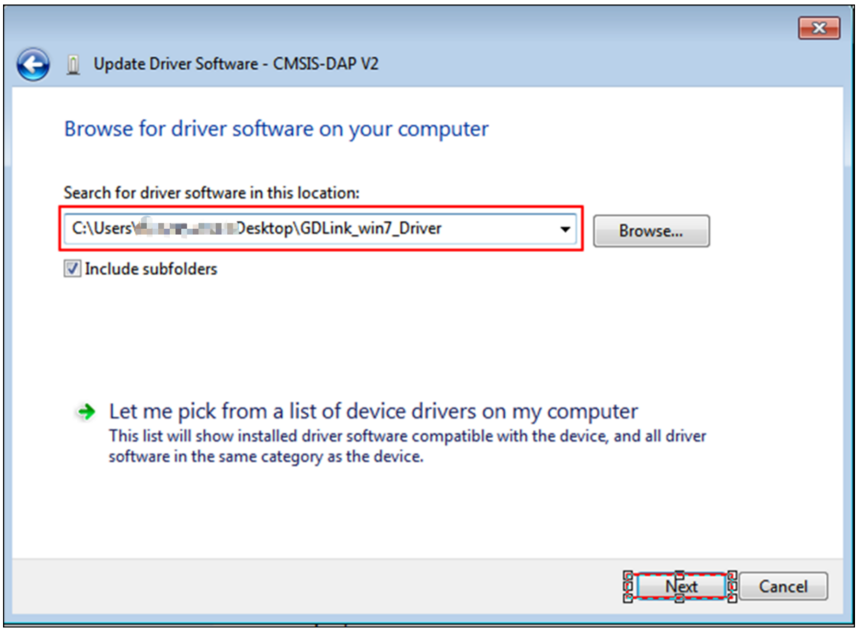
5. Select "Browse my computer for driver software". As shown in *Figure 4-9. Step 2: Install the driver*.

Figure 4-9. Step 2: Install the driver



6. Browse to the downloaded GDLink_win7_Driver folder path → Click "Next". As shown in *Figure 4-10. Step 3: Install the driver*.

Figure 4-10. Step 3: Install the driver

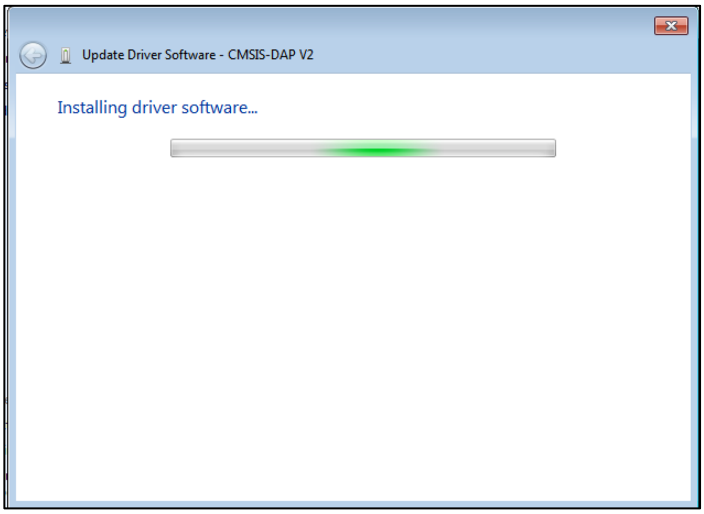


7. Select "Install this driver software anyway" (if security warning appears). As shown in *Figure 4-11. Step 4: Install the driver*.

Figure 4-11. Step 4: Install the driver

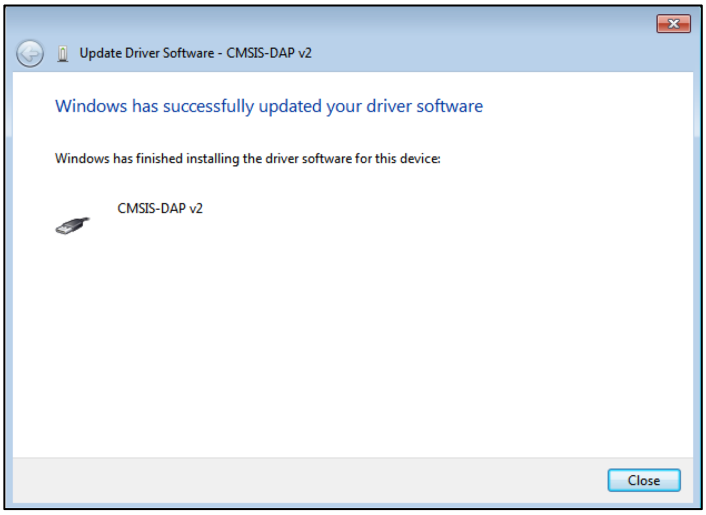
8. Wait for driver installation.... As shown in ***Figure 4-12. Step 5: Install the driver***.

Figure 4-12. Step 5: Install the driver



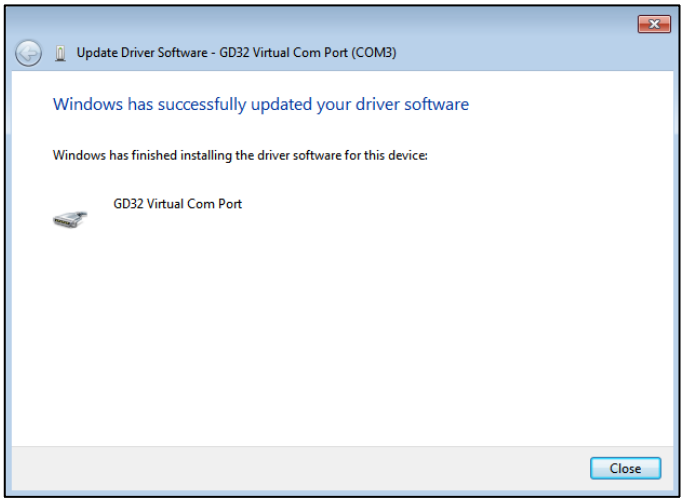
9. Installation complete when window shows "Windows has successfully updated your driver software". As shown in ***Figure 4-13. Step 6: Install the driver***.

Figure 4-13. Step 6: Install the driver



10. Repeat Same Process for GDLinker_V2. As shown in ***Figure 4-14. Step 7: Install the driver***.

Figure 4-14. Step 7: Install the driver



After installation, check in Device Manager:

- "Ports (COM & LPT)" - Should show GD32 Virtual Com port
- "Universal Serial Bus devices" - Should show CMSIS-DAP v2 related device
- Ensure no yellow warning signs or unknown devices remain

If displayed normally, user can use GD-Link for debugging and serial communication in development environment.

5. Revision history

Table 5-1. Revision history

Revision No.	Description	Date
1.0	Initial Release	Nov.1 2023
1.1	Add the output voltage section	Jan.2 2024
1.2	1. Add the Q&A section to explain common issues such as driver installation and debugger recognition failure. 2. Add description for enabling or disabling the Virtual USB disk drag and drop programming function via the GD-Link Programmer software in ***Virtual USB disk drag and drop programming***.	Nov.18 2025

Important Notice

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

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

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