GW1NS-2C MCU IDE

Software Reference Manual

RN519-1.1E,2018-11-21

Copyright©2018 Guangdong Gowin Semiconductor Corporation. All Rights Reserved.
No part of this document may be reproduced or transmitted in any form or by any denotes, electronic, mechanical, photocopying, recording or otherwise, without the prior written consent of GOWINSEMI.

Disclaimer

GOWINSEMI®, LittleBee®, Arora™, and the GOWINSEMI logos are trademarks of GOWINSEMI and are registered in China, the U.S. Patent and Trademark Office and other countries. All other words and logos identified as trademarks or service marks are the property of their respective holders, as described at www.gowinsemi.com.cn. GOWINSEMI assumes no liability and provides no warranty (either expressed or implied) and is not responsible for any damage incurred to your hardware, software, data, or property resulting from usage of the materials or intellectual property except as outlined in the GOWINSEMI Terms and Conditions of Sale. All information in this document should be treated as preliminary. GOWINSEMI may make changes to this document at any time without prior notice. Anyone relying on this documentation should contact GOWINSEMI for the current documentation and errata.

Revision

Date	Version	Description
2018/08/21	1.0E	Initial version.
2018/11/21	1.1E	● Optimize the Eclipse installation process ● Add project instance operations

Contents

List of Figures

1 ARM KEIL Software

1.1 Software Installation

Please refer to *MDK Getting Started* and *uVision User Guide* provided by ARM. (These two manuals are available in the following path: GW1NS-2C_package\release\rt...).

1.2 Project template

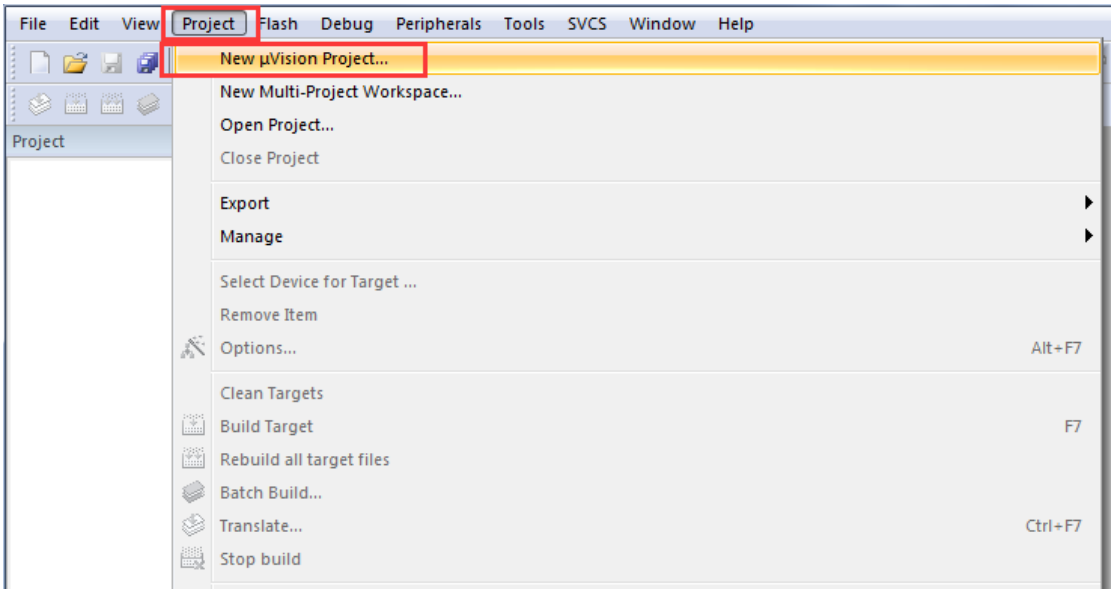
Using ARM KEIL software to develop embedded software requires creating project, configuring project options, coding, compiling, programming and debugging.

1.2.1 Configuration options

Project creation

Open the ARM KEIL software and select New uVision Project... in Project of the menu bar to create the project, as shown in Figure 1-1.

Figure 0-1 Creating a project

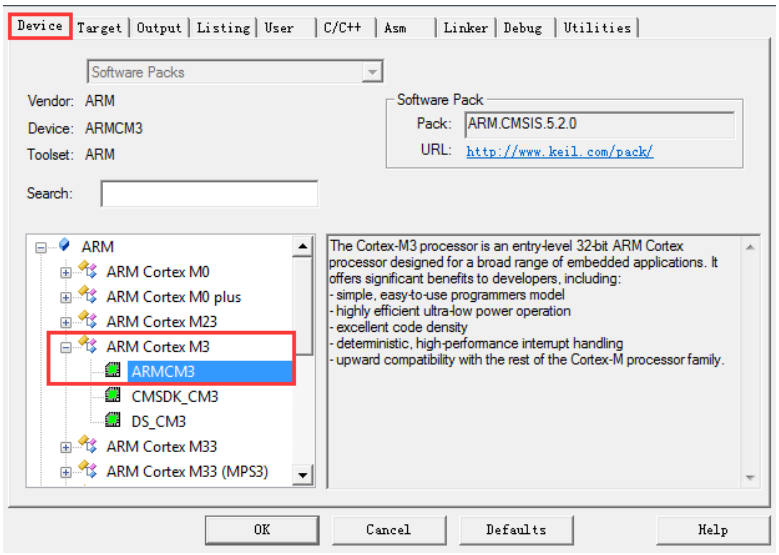


1.2.2 Project Configuration

Configuration of device

The GW1NS-2C MCU is the ARM Cortex-M3 core, so for the device, select "ARMCM3" of the ARM Cortex-M3, as shown in Figure 1-2.

Figure 0-2 Configuration of device



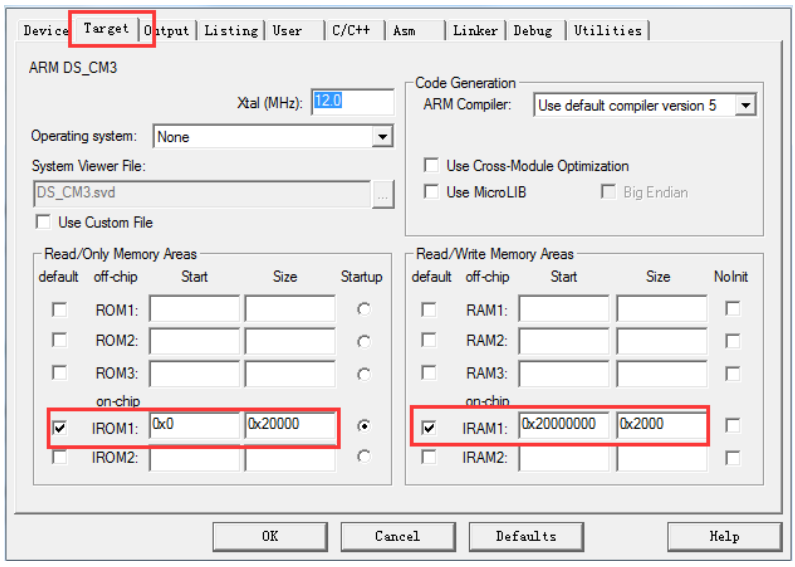
Configuring ROM and RAM

Configure the starting address and capacity of the ROM and RAM.

The starting address of the GW1NS-2C Flash-Rom is 0x20000000 and the capacity is 128K Byte.

The starting address of the GW1NS-2C Sram is 0x20000000 and the capacity is 8K Byte, as shown in Figure 1-3.

Figure 0-3 Configuration of ROM and RAM

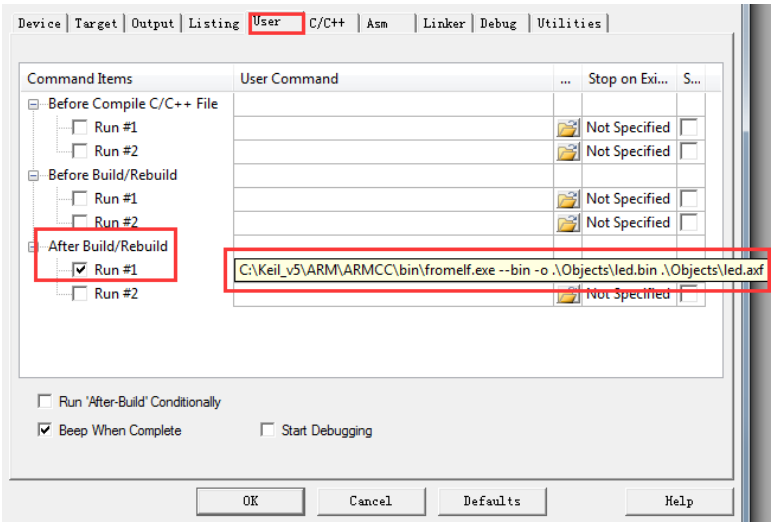


Configuring output file format

Gowin Programmer supports BIN binary file programming format, so the configuration output file format is BIN.

The *.axf file is converted to the *.bin file in the User command line option, as shown in Figure 1-4.

Figure0-4 Configuration of output file format



The command format is:

*:\Keil_v5\ARM\ARMCC\bin\fromelf.exe --bin

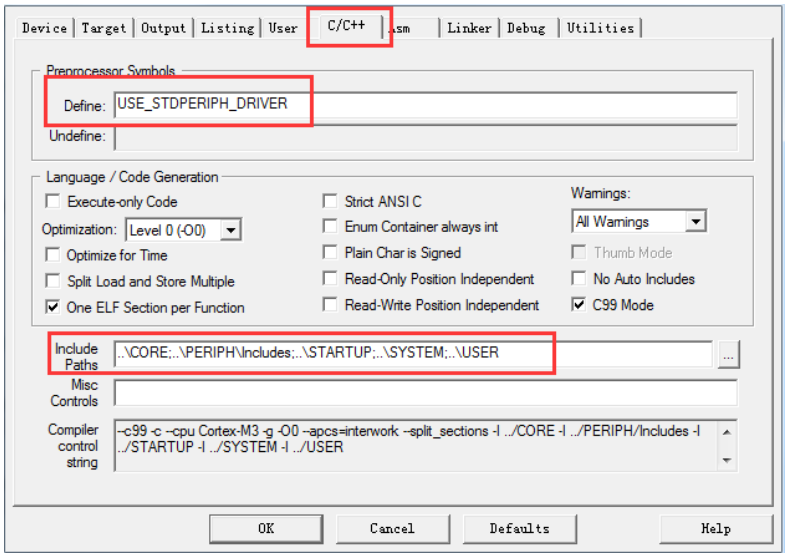
-o ./Objects/*.bin ./Objects/*.axf

Configure macro definitions and header file paths

Configure macro definition "USE_STDPERIPH_DRIVER" to call standard peripherals.

Configure the header file path to call the header file during compilation. The configuration is shown in Figure 1-5.

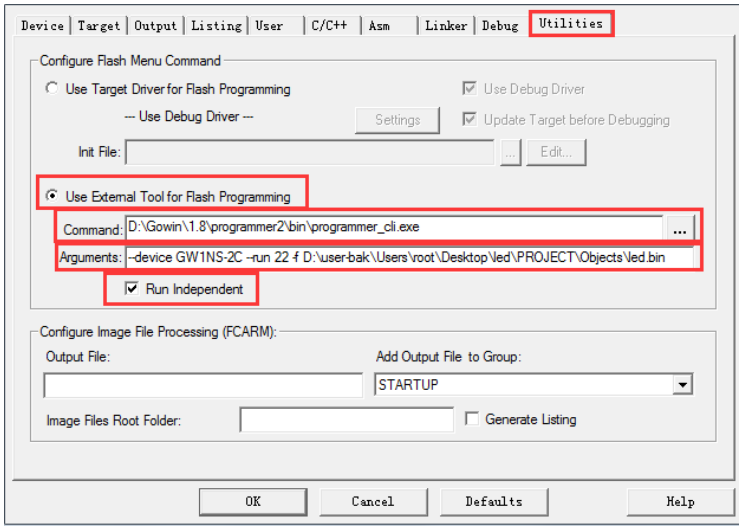
Figure 0-5 Configure macro definitions and header file paths



Configuration of programmer tool

Configure the Flash download tool as an external tool and use Gowin Programmer, as shown in Figure 1-6.

Figure 1-6 configuration of download options



Command is the Programmer path.

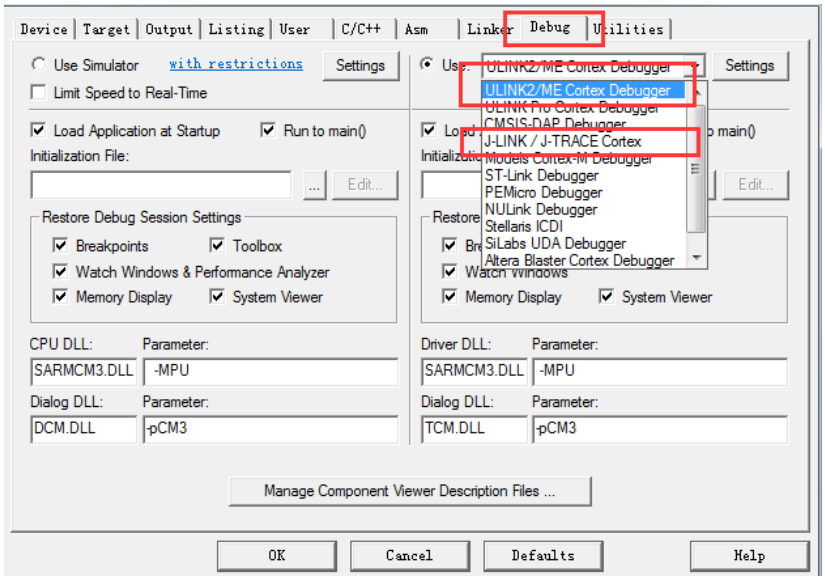
Arguments are options for the Programmer command, such as --device GW1NS-2C --run 22 -f *.bin.

Configuring debug options

If you choose to use the U-LINK emulator, the Debug option is configured as "ULNK2/ME Cortex Debugger".

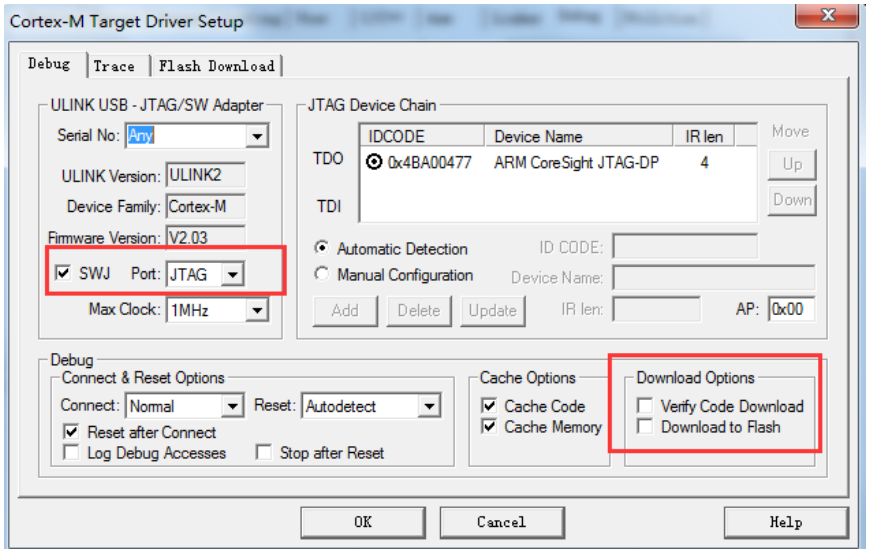
If you choose to use the J-LINK emulator, the Debug option is configured as "J-LINK/J-TRACE Cortex", as shown in Figure 1-7.

Figure 1- 7 Configuring debug options



The debug interface type is configured as JTAG, as shown in Figure 1-8.

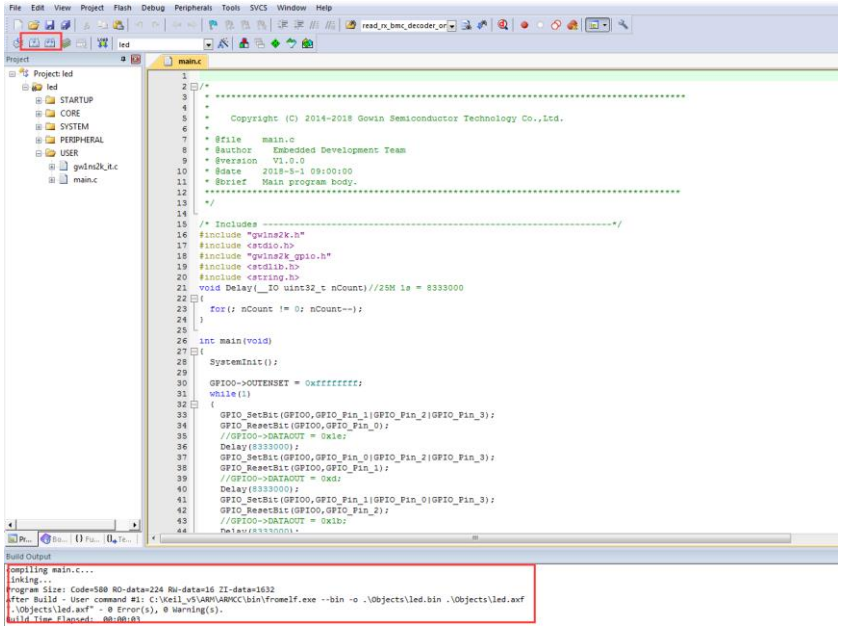
Figure 0-6 Debug interface type



1.2.3 Project compilation

After the encoding and project configuration are completed, the BIN format image file is compiled and generated, as shown in Figure1-9.

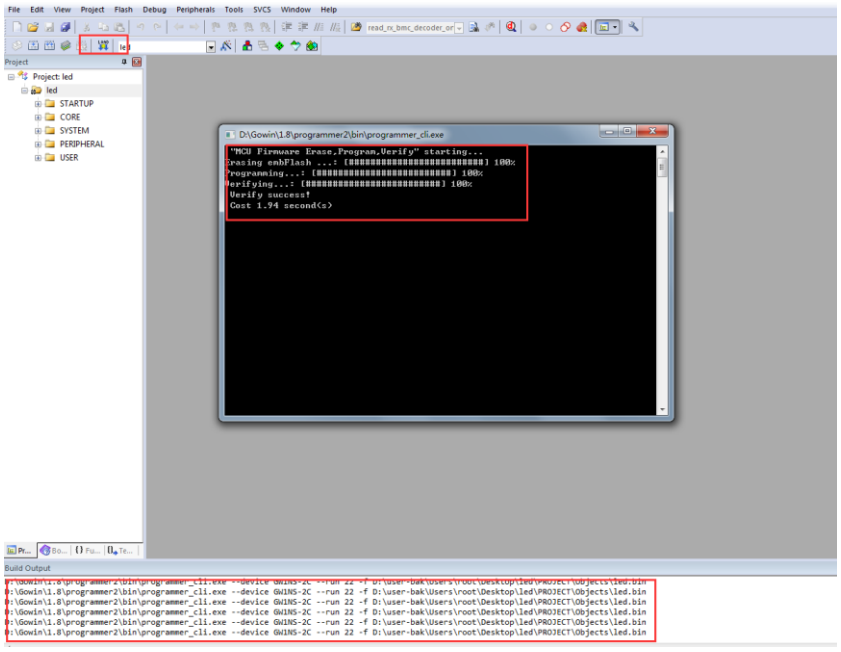
Figure 0-7 Project compilation



1.2.4 Project Programmer

After completing the project compilation and configuring Gowin Programmer software as the programmer tool, select Download to complete the program, as shown in Figure 1-10.

Figure 0-8 Programmer



1.2.5 Project debugging

After the MCU image file is programmed, if there is a problem with the user design, you can connect to the U-LINK or J-LINK emulator to use the

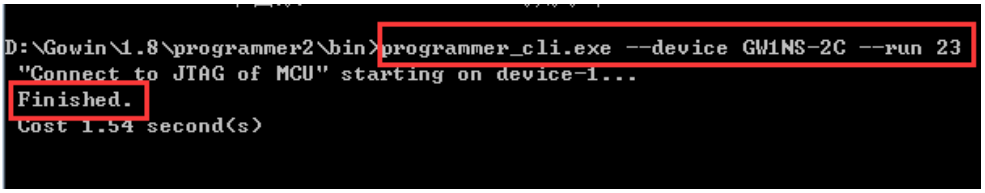
debugging function.

JTAG mode switching

As shown in Figure 1-11, use Gowin Programmer command line to manually switch the JTAG mode from download mode to debug mode. The command is:

programmer_cli.exe --device GW1NS-2C --run 23

Figure 0-9 Switching command in JTAG mode



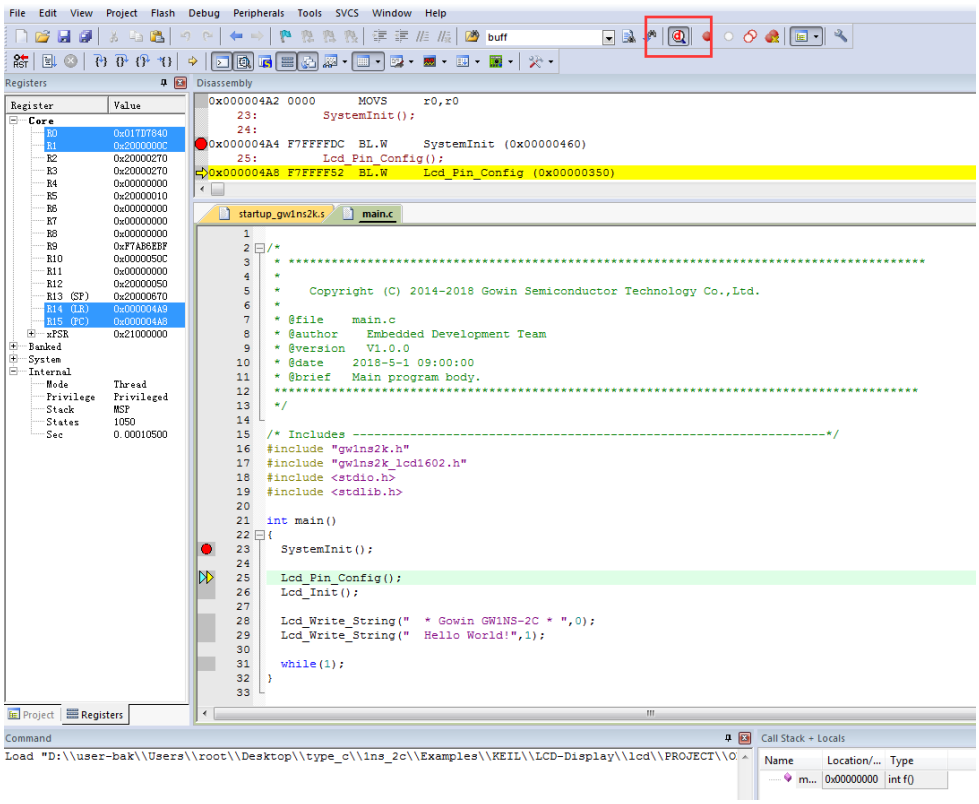
JTAG interface switching

If you use the development board "DK-EVAL-GW1NS V1.1", you need to manually switch the JTAG interface TMS, TCK, TDI and TDO jumper caps from FDTI programming to ARM programming, that is, switch from (1, 3) port to (3, 4) port.

Start debugging

Connect the U-LINK or J-LINK emulator, select the Debug button on the toolbar, and start debugging, as shown in Figure 1-12.

Figure 0-10 Start debugging



1.3 Reference design

Gowin provides a reference design based on ARM KEIL software: Gowin GW1NS-2C_MCU_PACK\Gowin_GW1NS-2C_MCU_RefDesign\MCU_RefDesign\Keil

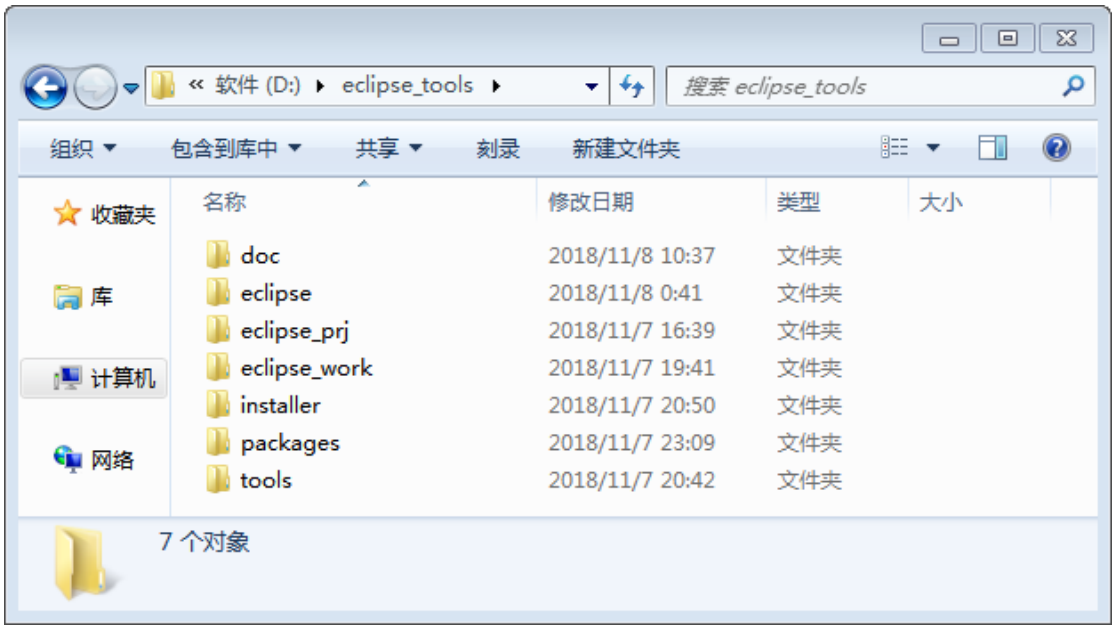
2GNU MCU Eclipse software

2.1 Software Installation

Gowin Semiconductor offers GNU MCU Eclipse installation package.

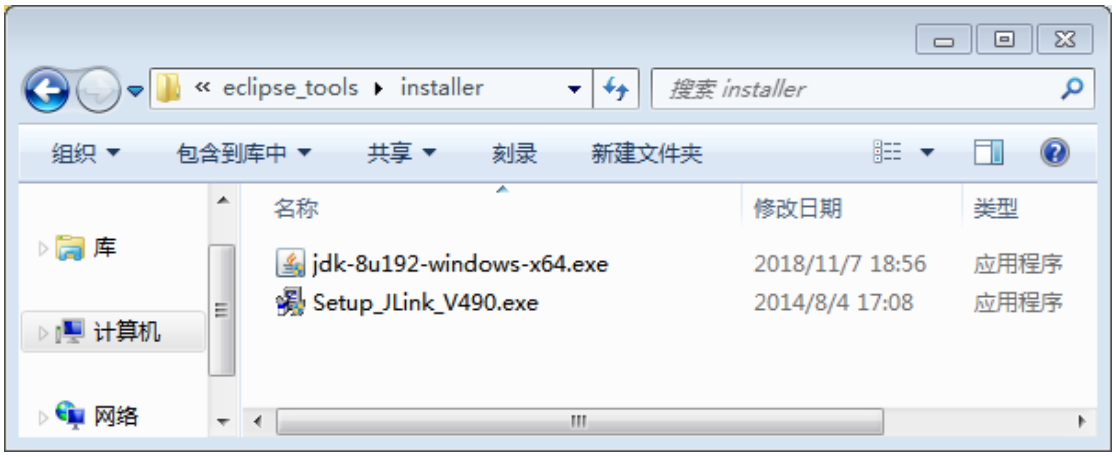
Obtain the GNU MCU Eclipse installation package and extract it to the root directory of the local D drive, as shown in Figure 2-1.

Figure 2-1Directory structure of GNU MCU Eclipse installation package



Select the installer directory to get the JDK installation file and the J-LINK emulator driver installation file, as shown in Figure 2-2.

Figure 2-2 Directory structure of JDK and J-LINK installation



2.1.1 Install JDK

Select the JDK installation file

Select the installation file jdk-8u192-windows-x64.exe and double-click it to open it, as shown in Figure 2-3.

Figure 2-3 JDK Installation Wizard

Select optional features and installation directories to install

Select the optional features and installation directory to install, as

shown in Figure 2-4. The default installation path is C:\Program Files\Java\jdk1.8.0_192.

Figure 2-4 Select to install the JDK optional features

Select the JRE installation directory

Select the JRE installation directory, as shown in Figure 2-5. The default path is C:\Program Files\Java\jre1.8.0_192.

Figure 2-5 Select the JRE installation path

Complete the JDK installation

As shown in Figure 2-6, JDK is successfully installed and you can choose to close it.

Figure 2-6 Complete JDK installation

Test JDK installation

Select the start menu locally, select the cmd command line, and enter the command "java-version". If you can output the Jave version information, the installation is successful, as shown in Figure 2-7.

Figure 2-7 Test JDK installation

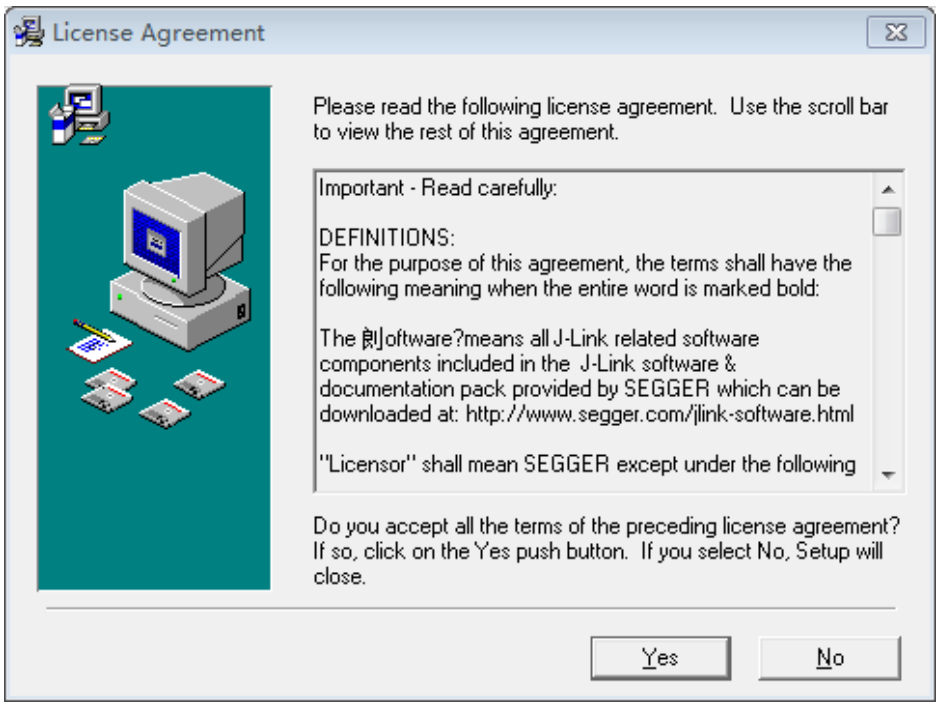
2.1.2 Install J-LINK

Select installation file

Select the installation file Setup_JLink_V490.exe and double-click it to open it, as shown in Figure 2-8, confirm the License Agreement.

If you have J-LINK installed locally, you don't need to install J-LINK again, just set the path of the locally installed J-LINK in Eclipse.

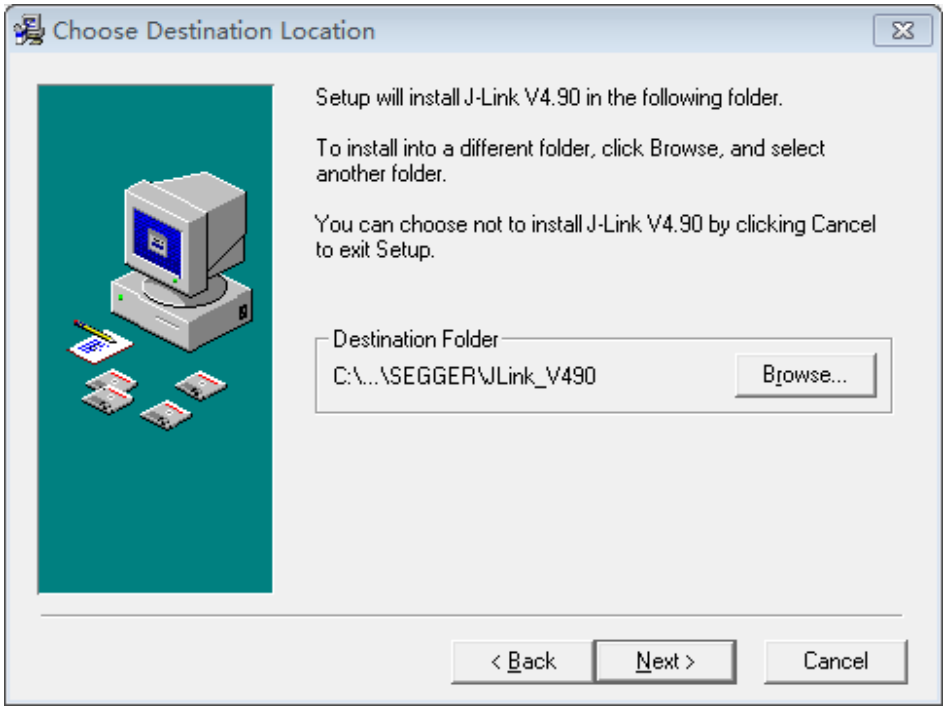
Figure 2-8 J-LINK License Agreement



Select the directory to install

Select the directory to install, as shown in Figure 2-9. The default path is C:\Program Files (x86)\SEGGER\JLink_V490.

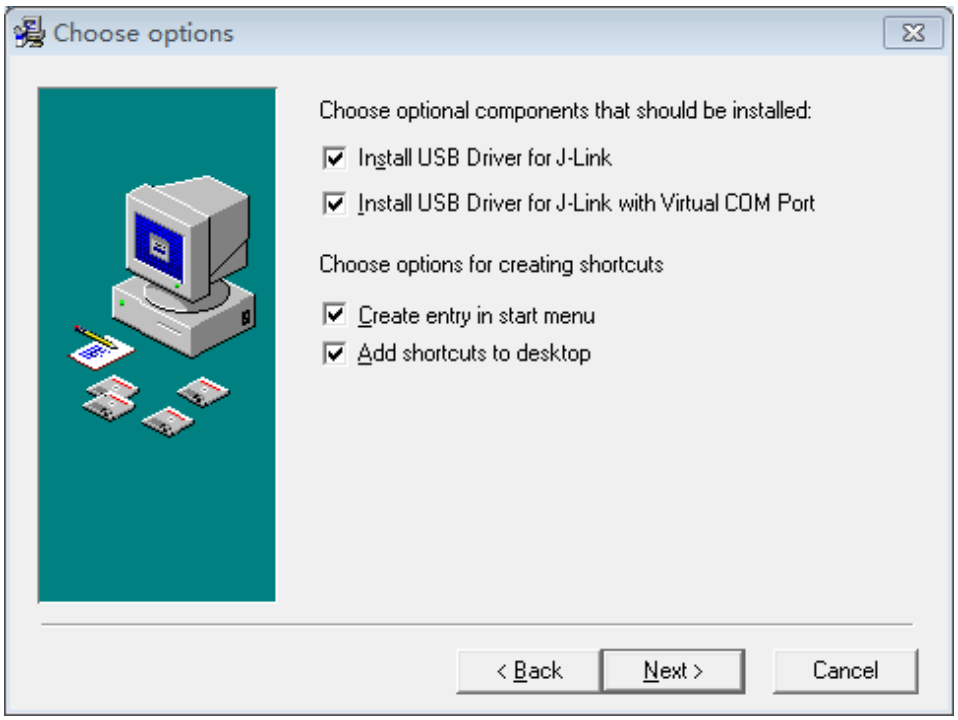
Figure 2-9 Select the installation path



Select the functional components to install

Select the functional components to install, as shown in Figure 2-10.

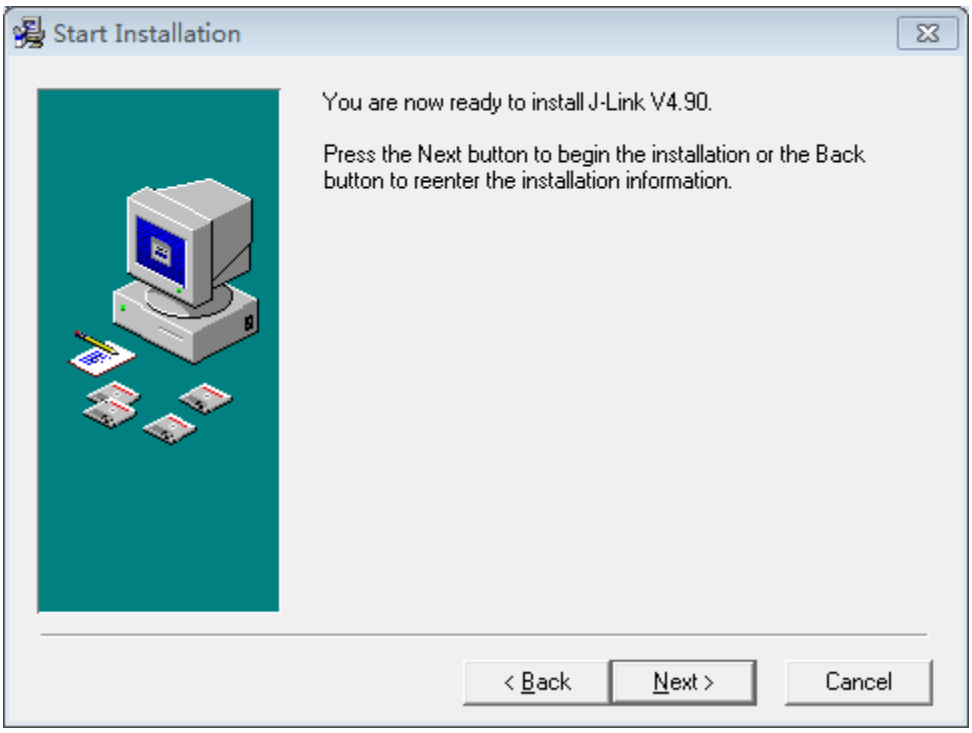
Figure 2-10 Selecting a functional component



Choose to start the installation

Start the installation, as shown in Figure 2-11.

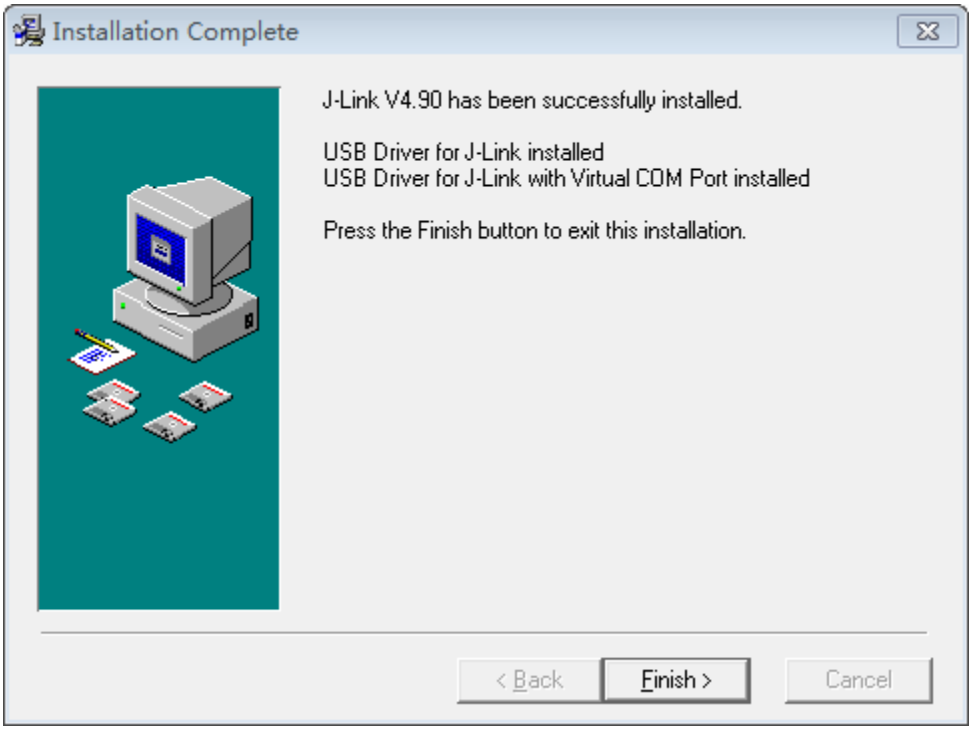
Figure 2-11 Start the installation wizard



Finish installation

Finish the installation, as shown in Figure 2-12.

Figure 2-12 Complete the installation



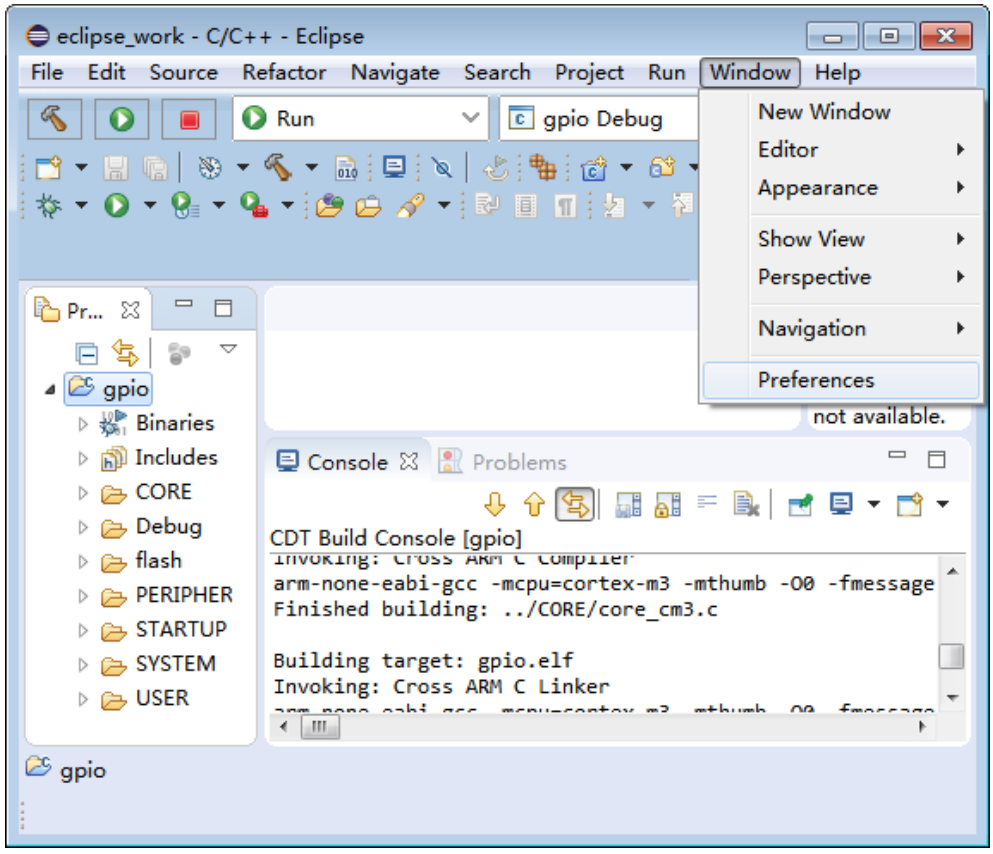
2.1.3 Install the MCU device package

MCU device packages are provided in the GNU MCU Eclipse installation package, as shown in Figure 2-1, and the packages are MCU device packages.

Specify the MCU device package path

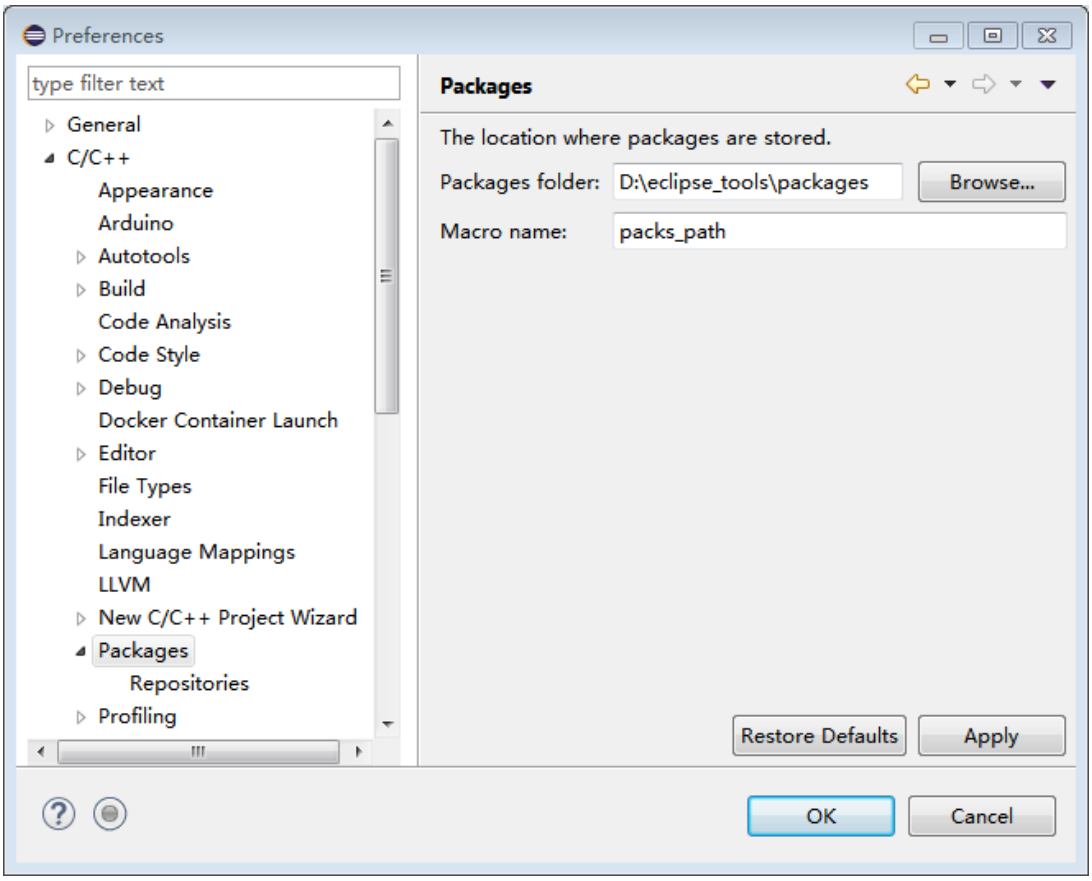
Select Preferences in Window of the menu bar, as shown in Figure 2-13.

Figure 2-13 Select Preferences



Select the Packages option in the C/C++ tab in Preferences and specify the Packages folder and Macro name, as shown in Figure 2-14.

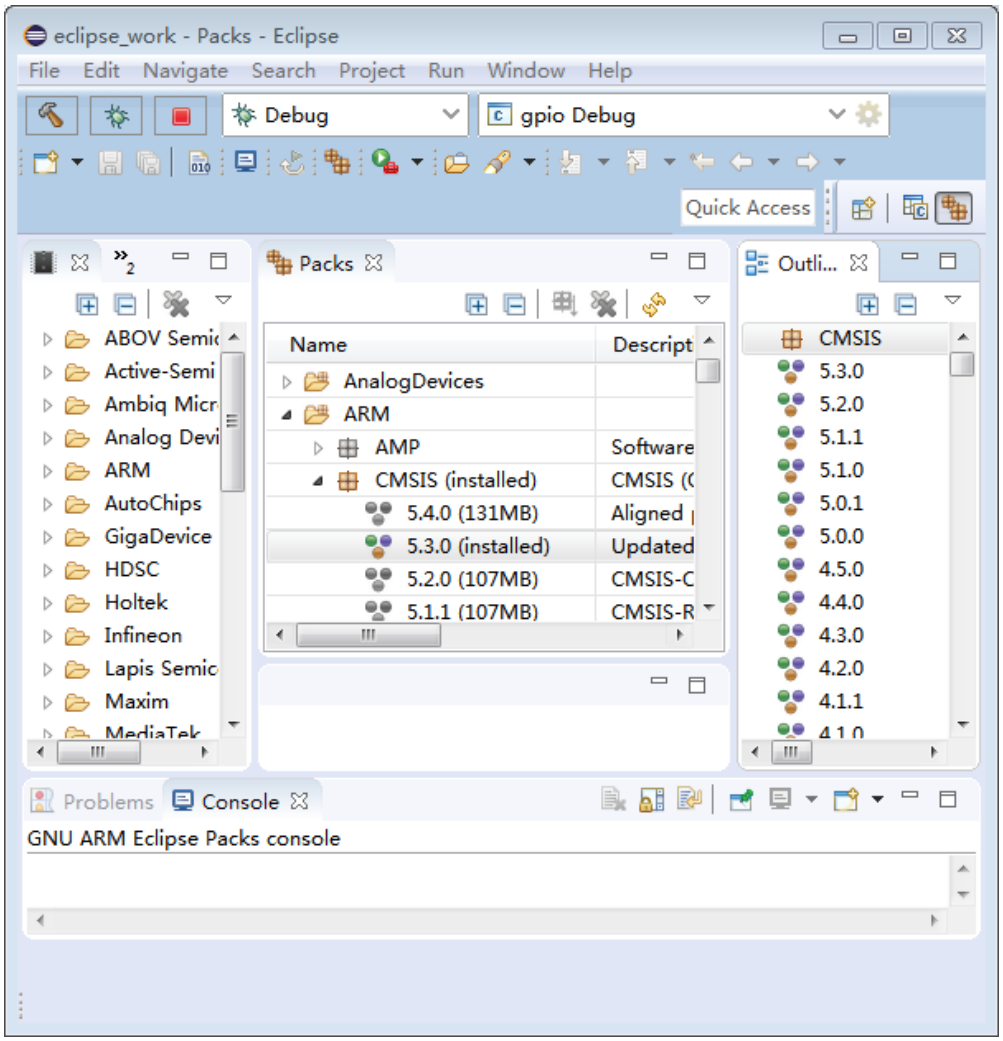
Figure 2-14 Specify the Packages path



Update MCU device list online

Select the Packs button in the toolbar to update the MCU device package list online, as shown in Figure 2-14. Since the GNU MCU Eclipse installation package already provides the ARM.CMSIS.5.3.0 MCU device package, Figure 2-15 shows that the device package is installed.

Figure 2-15 Update device package list



2.2 Software configuration

The configured template project is provided in the GNU MCU Eclipse software installation package. The eclipse_proj\demo\gpio template project is imported into the software installation package to configure the Eclipse software.

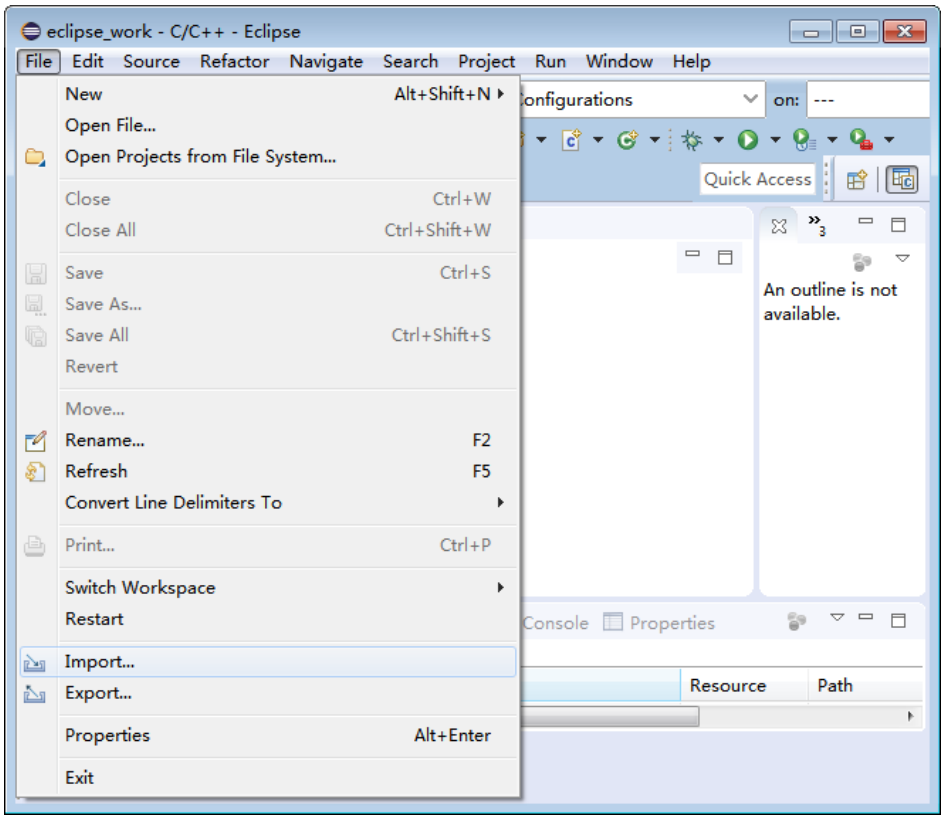
If the GNU MCU Eclipse installation package path, JDK installation path, J-LINK installation path, and Gowin YunYuan Software installation path are the same as described above, you can use Eclipse software to compile, link, program, and debug without configuring Eclipse.

2.2.1 Import template project

As shown in Figure 2-16, select Open Projects from File System in File

of the menu bar to import the template project eclipse_proj\demo\gpio.

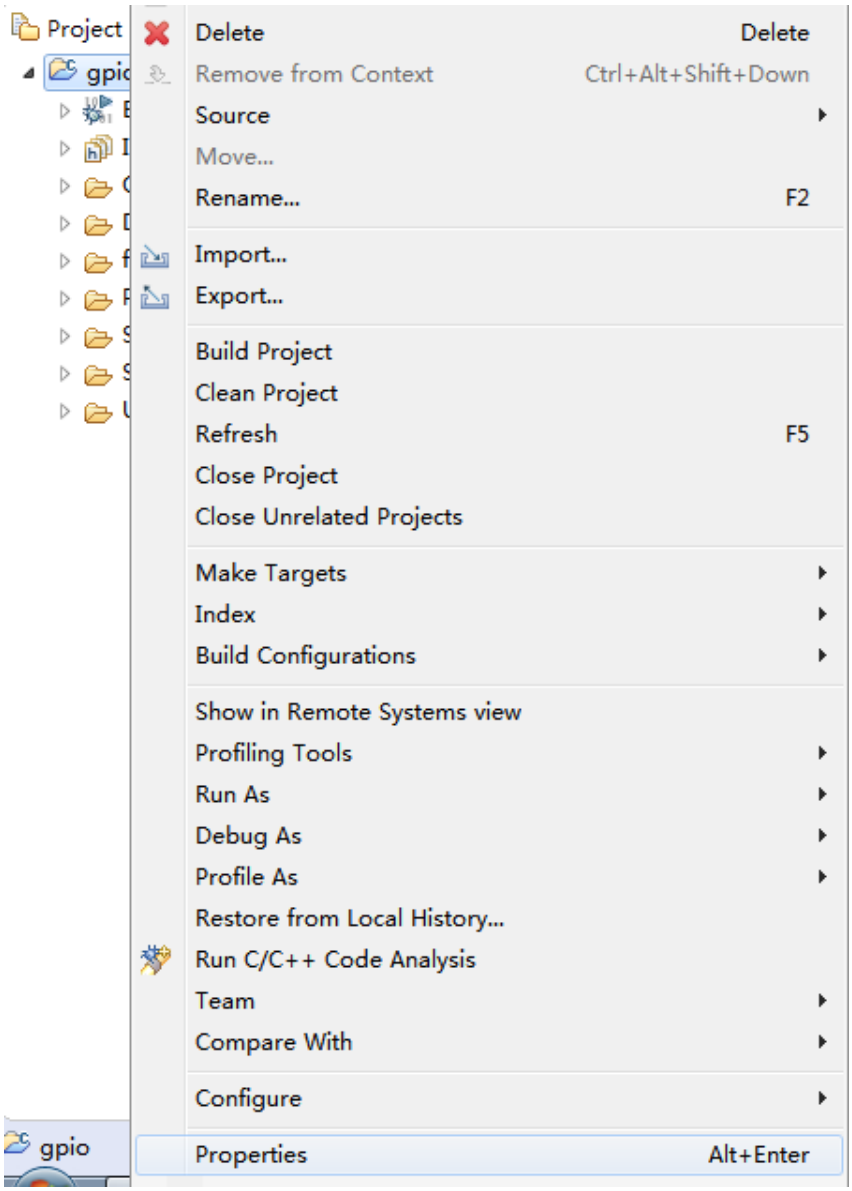
Figure 2-16 Import template project



2.2.2 Select Properties

After importing the template project, select the template project gpio, right click and select Properties, as shown in Figure 2-17.

Figure 2-17 Select Properties

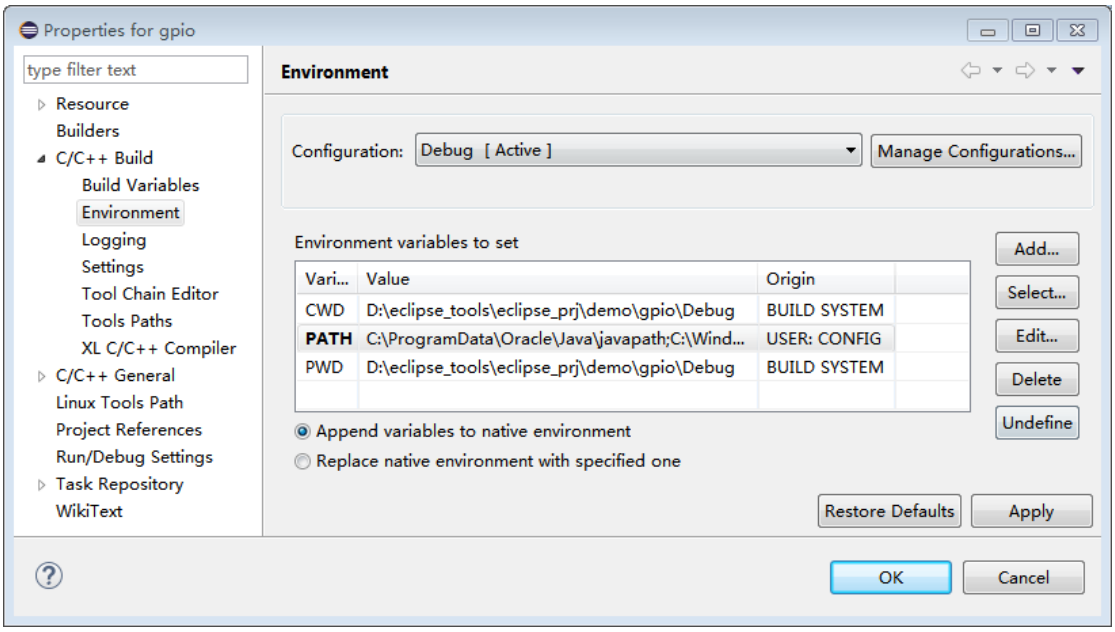


2.2.3 Configuring environment variables

Select the Environment option in the C/C++ Build tab, as shown in Figure 2-18, and configure the following environment variables:

- C:\ProgramData\Oracle\Java\javapath;
- C:\Windows\system32;C:\Windows;
- C:\Windows\System32\Wbem;
- C:\Windows\System32\WindowsPowerShell\v1.0\;
- D:\eclipse_tools\tools\GNU Tools ARM Embedded\5.4 2016q2\bin;
- D:\eclipse_tools\tools\GNU MCU Eclipse\Build Tools\2.11-20180428-1604\bin

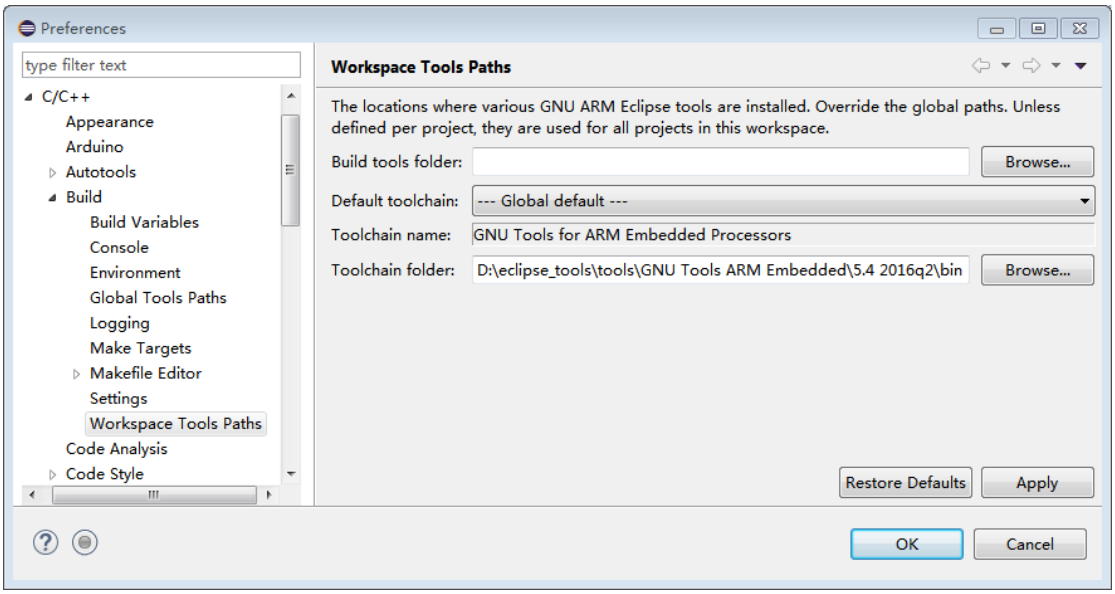
Figure 2-18 Configuring environment variables



2.2.4 Configuring the tool chain

Select Preferences in Window of the menu bar, select Workspace Tools Paths in Build in the C/C++ tab, and configure the toolchain path, as shown in Figure 2-19.

Figure 2-19 Configuring the tool chain



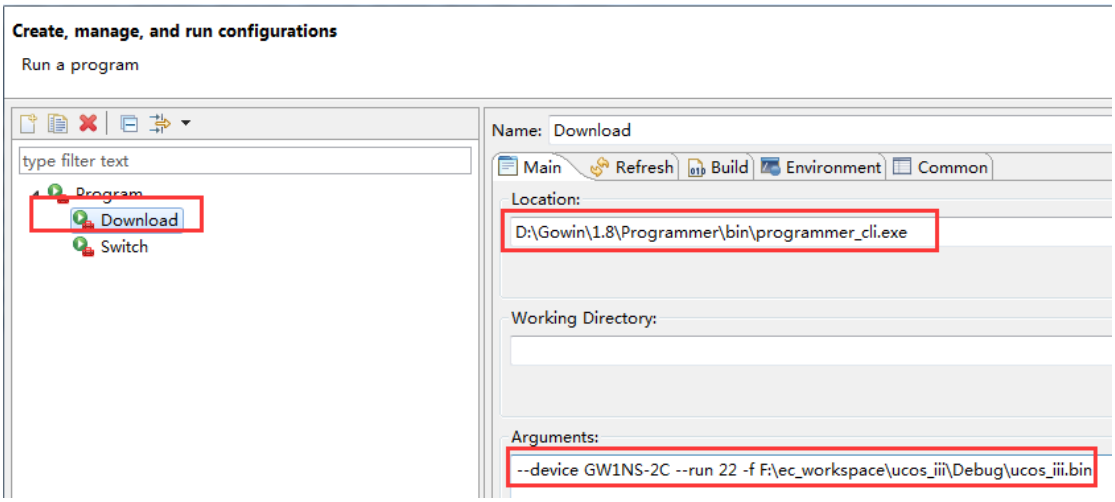
2.2.5 Configuring the Programmer Tool

The GNU MCU Eclipse software uses Gowin Programmer software to

program MCU image files.

In the GNU MCU Eclipse software, configure Gowin Programmer software as the MCU image file programmer tool. Select External Tools Configurations...in External Tools in Run of the menu bar to configure the programmer tool, as shown in Figure 2-20.

Figure 2-20 Configuring the Programmer Tool



2.2.6 Configuring the switching tool in JTAG mode

If the user needs to debug the project, the JTAG mode needs to be switched from the programming mode to the debugging mode.

In the GNU MCU Eclipse software, configure Gowin Programmer software as the JTAG mode switching tool. Select External Tools Configurations... in External Tools in Run of the menu bar to configure the switching tool, as shown in Figure 2-21.

Figure 2-21 Configuring the Switch Tool in JTAG Mode

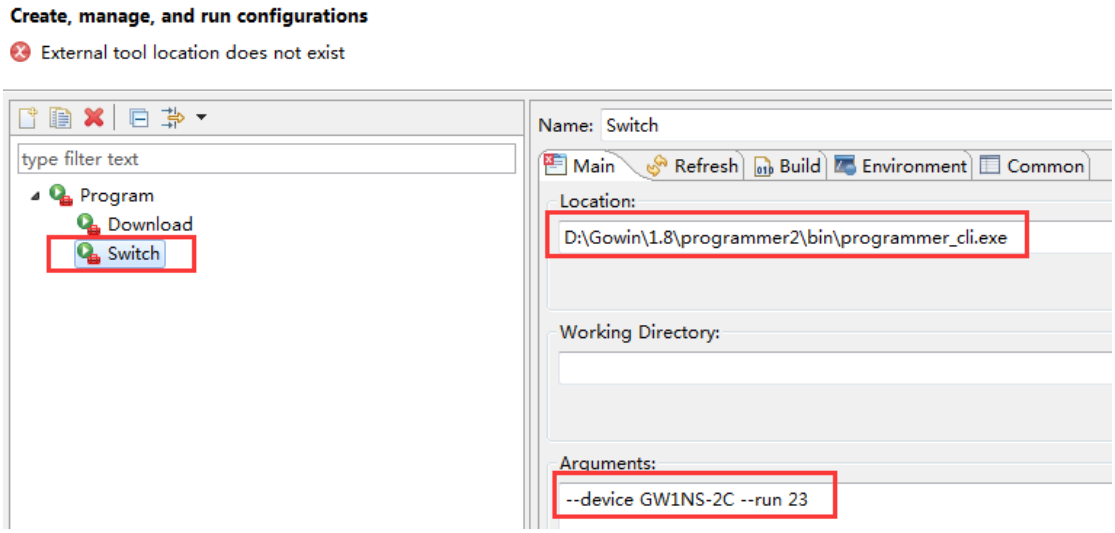
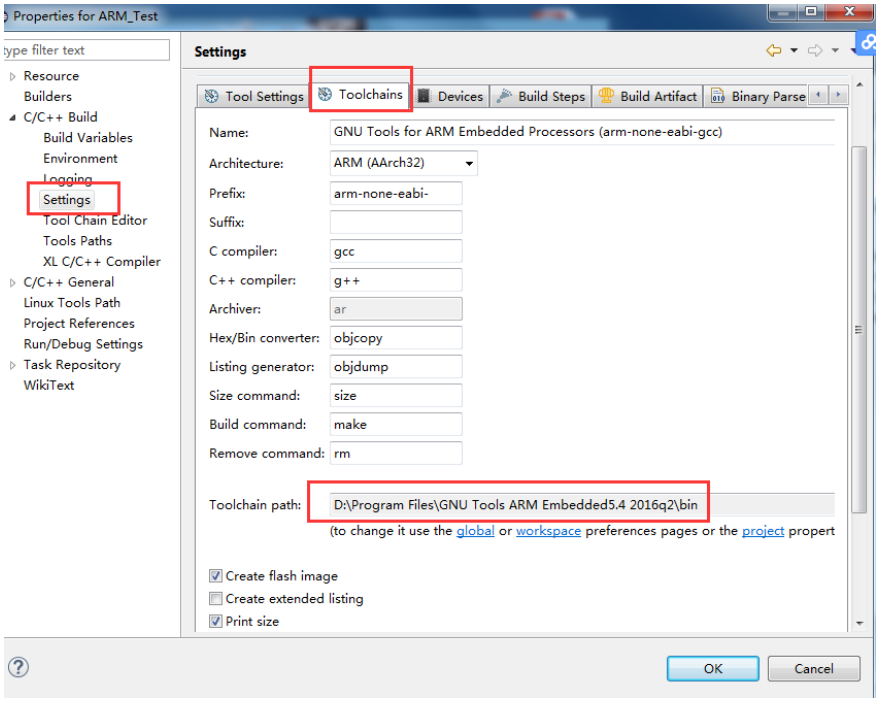


Figure 2-22 Cross-compilation configuration



2.3 Project template

2.3.1 Project creation

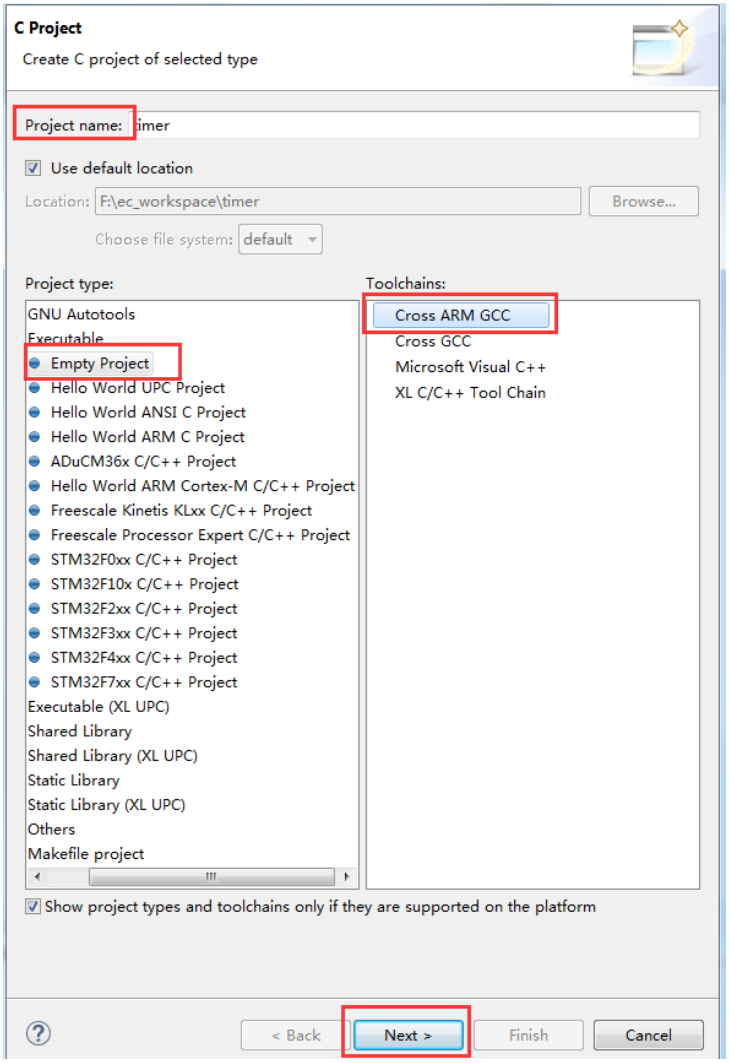
New Project

Select C Project in New in File of the menu bar, as shown in Figure 2-23.

- Create project name
- Select the project type, Empty Project

- Select the tool chain, Cross ARM Gcc

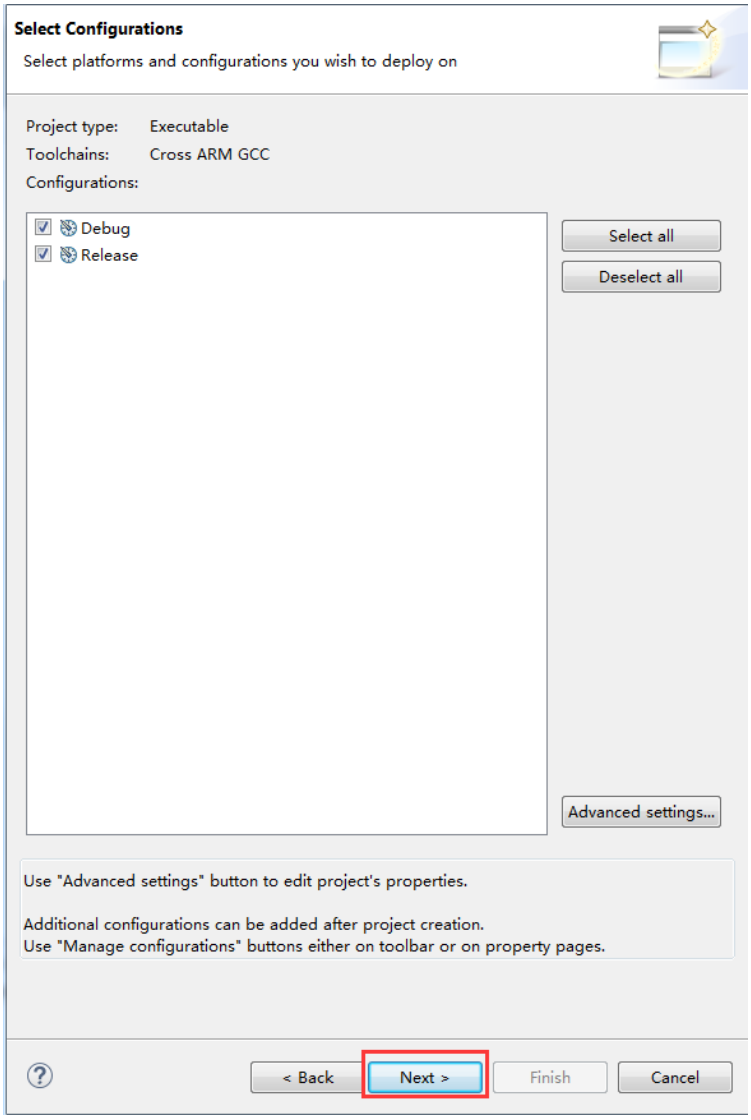
Figure 2-23 New Project



Select configuration

Select the project configuration type, Debug and Release, as shown in Figure 2-24.

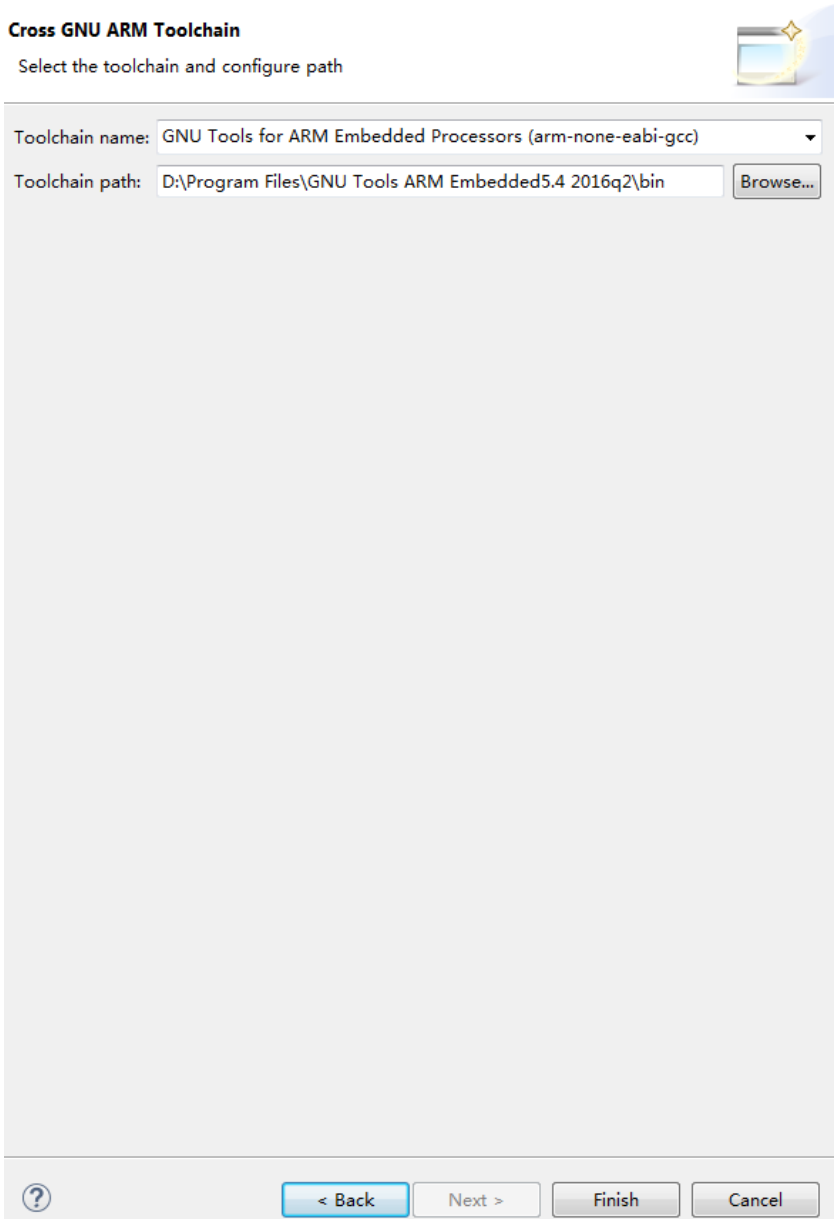
Figure 2-24 Select project type configuration



Select toolchain and path

Select the cross-compilation toolchain, arm-none-eabi-gcc, as shown in Figure 2-25.

Figure 2-25 Select tool chain

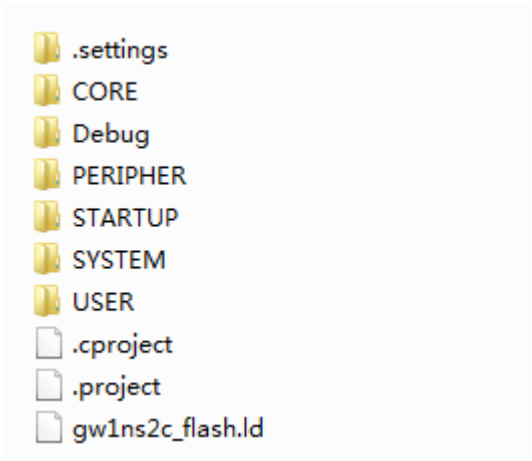


Establish project structure

After completing the new project, select the new project in the Eclipse workspace, and add the project structure and code, as shown in Figure 2-26.

- CORE： ARM Cortex-M3 kernel definition
- PERIPHER： Peripheral driver library
- STARTUP： Boot startup file
- SYSTEM： Register definition, system initialization, and system clock definition
- USER： User design
- gw1ns2c_flash.ld： Flash link script

Figure 2-26 Project Structure



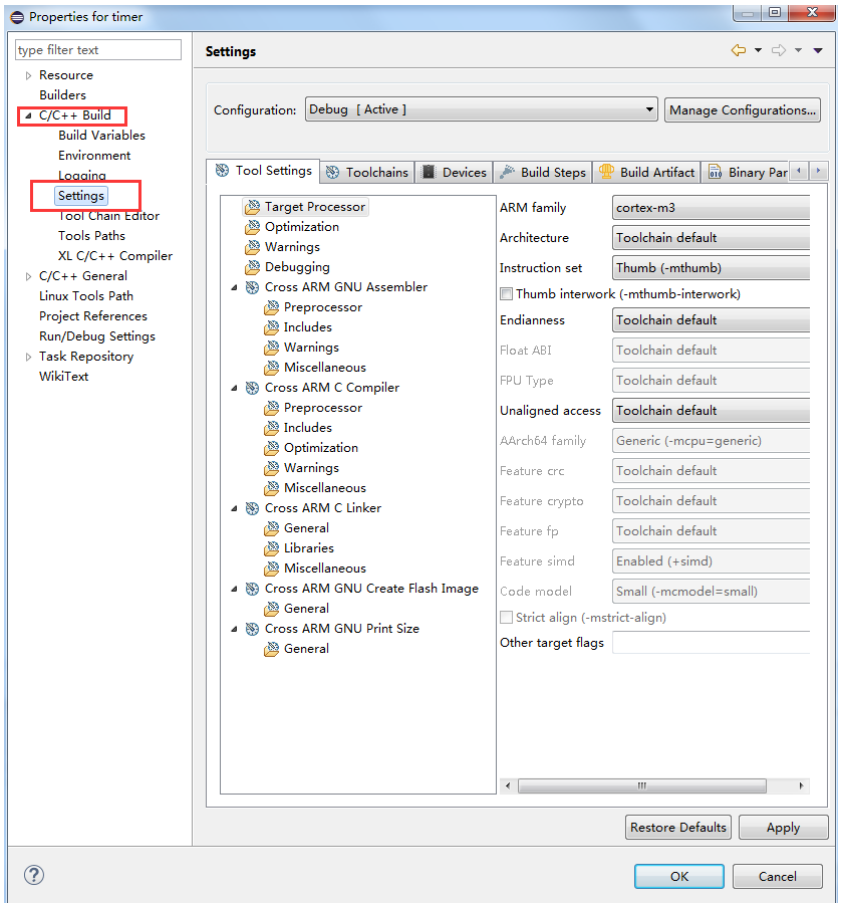
Importing files

In the Eclipse software, select the current project, right-click to select Refresh, and automatically import the project structure and code as shown in Figure 2-24.

2.3.2 Project creation

In the Eclipse software, select the current project, right-click on Properties, select the tab C/C++ Build, and select the Setting option, as shown in Figure 2-27.

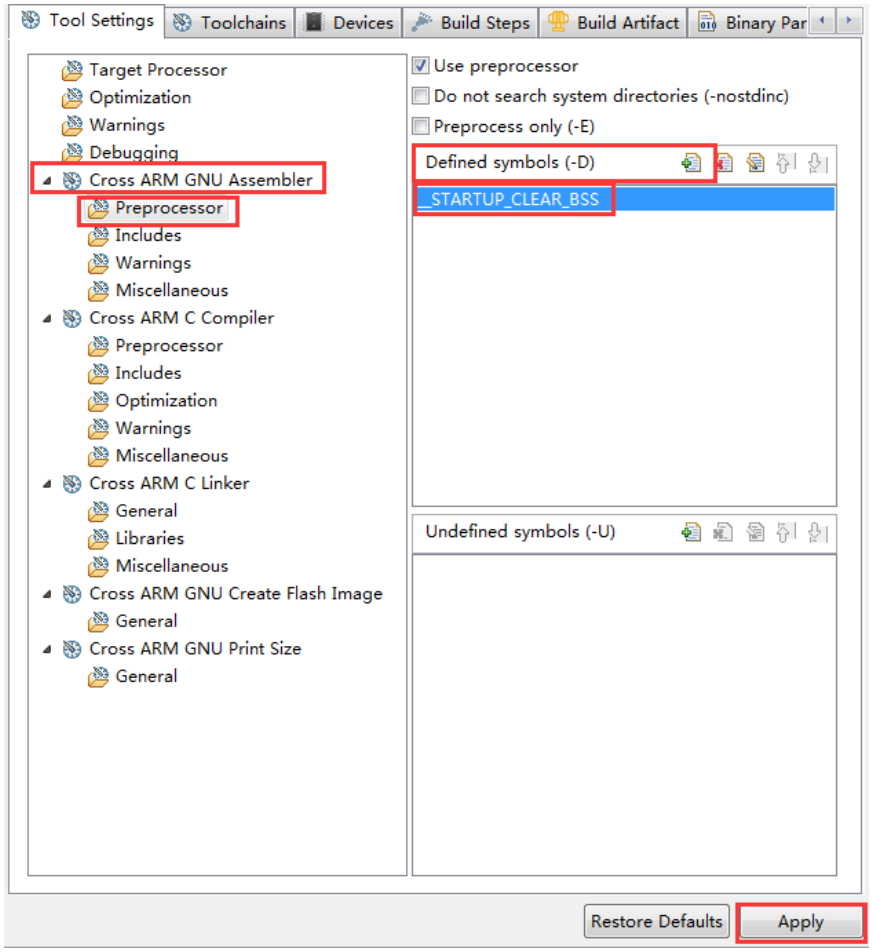
Figure 2-27 Select Properties



Configure Cross ARM GNU Assembler Preprocessor

Select the Preprocessor in the Cross ARM GNU Assembler and configure the assembly macro definition __STARTUP_CLEAR_BSS, as shown in Figure 2-28.

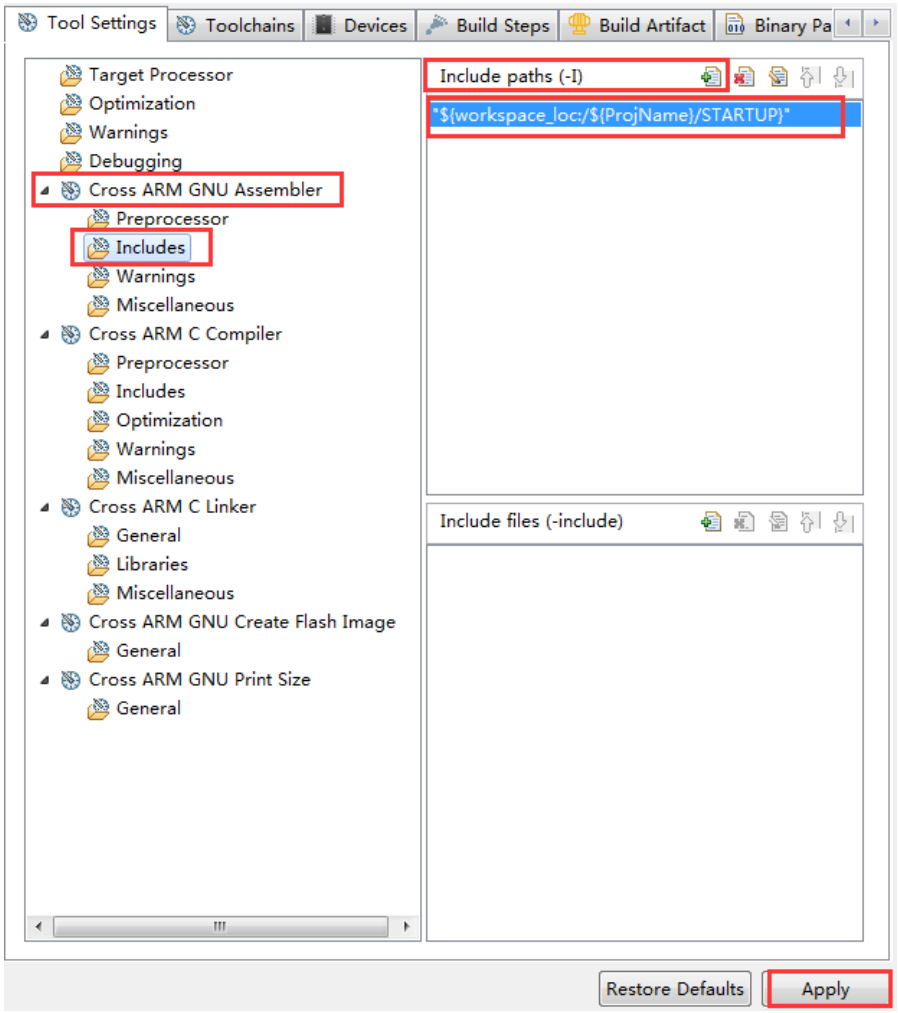
Figure 2-28 Configure the Cross ARM GNU Assembler Preprocessor



Configure Cross ARM GNU Assembler Includes

Select Includes in the Cross ARM GNU Assembler to configure the assembly reference file path, as shown in Figure 2-29.

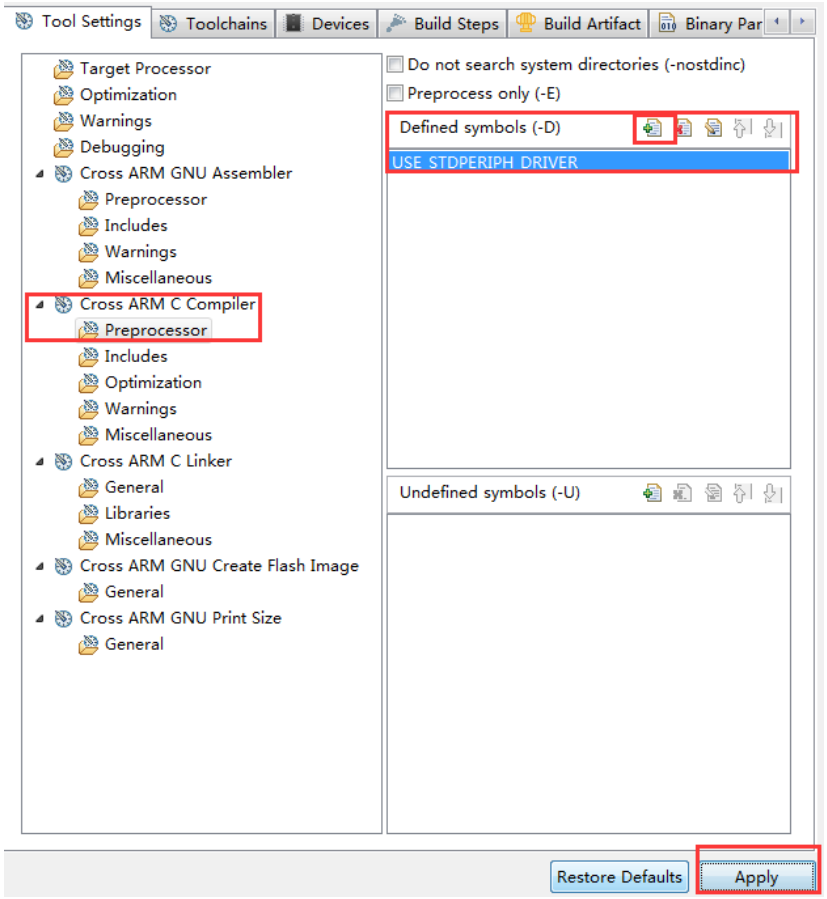
Figure 2-29 Configuring Cross ARM GNU Assembler Includes



Configuring Cross ARM C Compiler Preprocessor

Select Preprocessor in Cross ARM C Compiler and configure C macro definition USE_STDPERIPH_DRIVER, as shown in Figure 2-30.

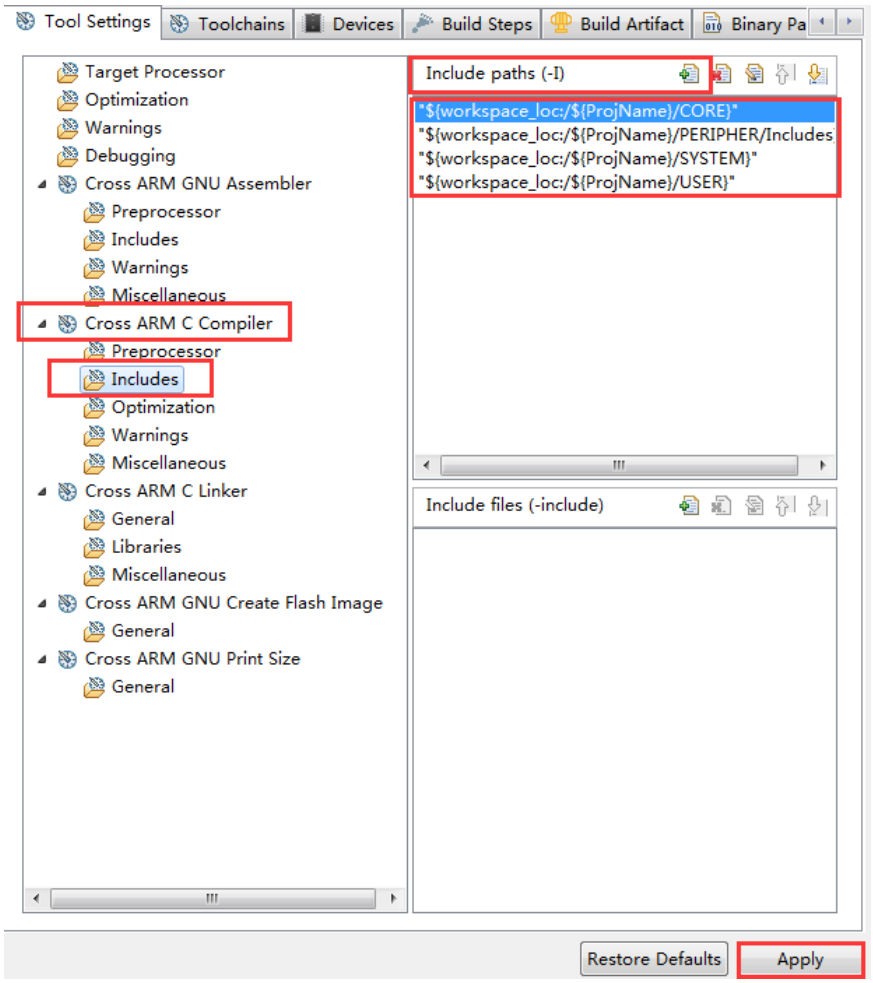
Figure 2-30 Configuring Cross ARM C Compiler Preprocessor



Configure Cross ARM C Compiler Includes

Select Includes in the Cross ARM C Compiler to configure the C reference file path, as shown in Figure 2-31.

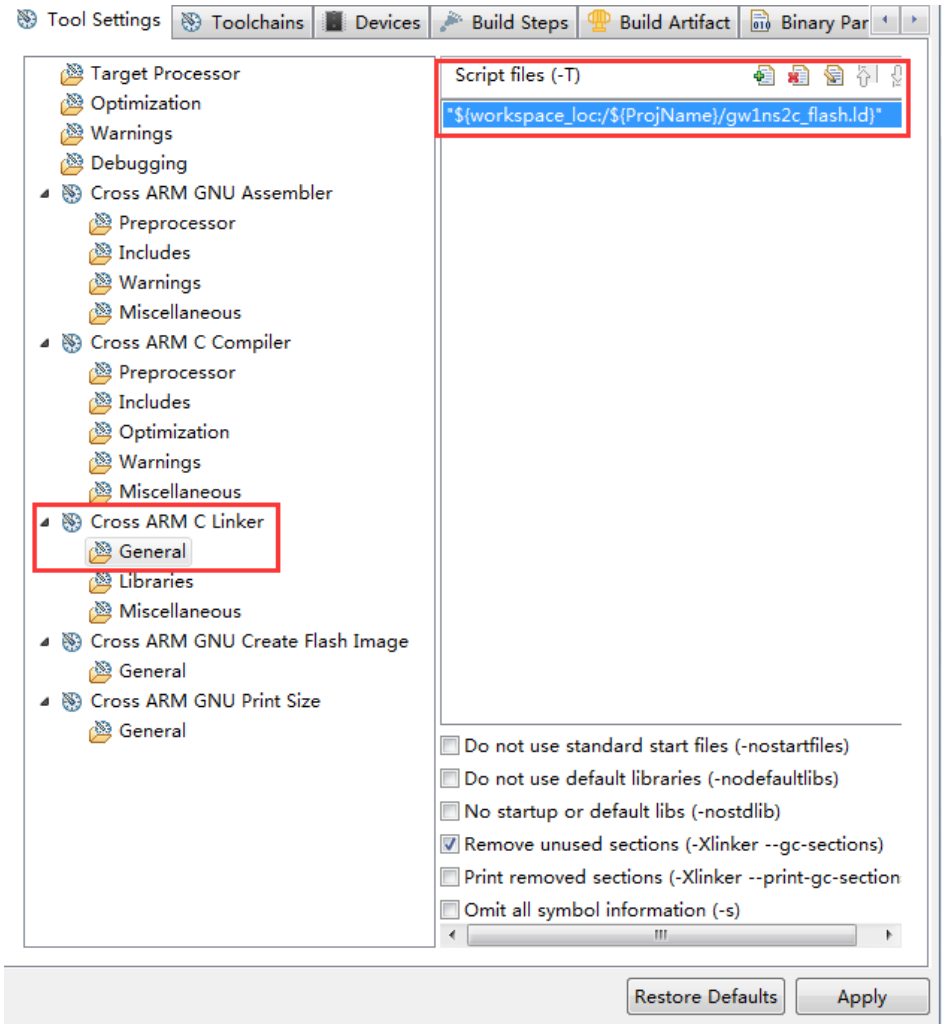
Figure 2-31 Configuring Cross ARM C Compiler Includes



Configuring Cross ARM C Linker

Select General in the Cross ARM C Linker to configure the Flash link script, as shown in Figure 2-32.

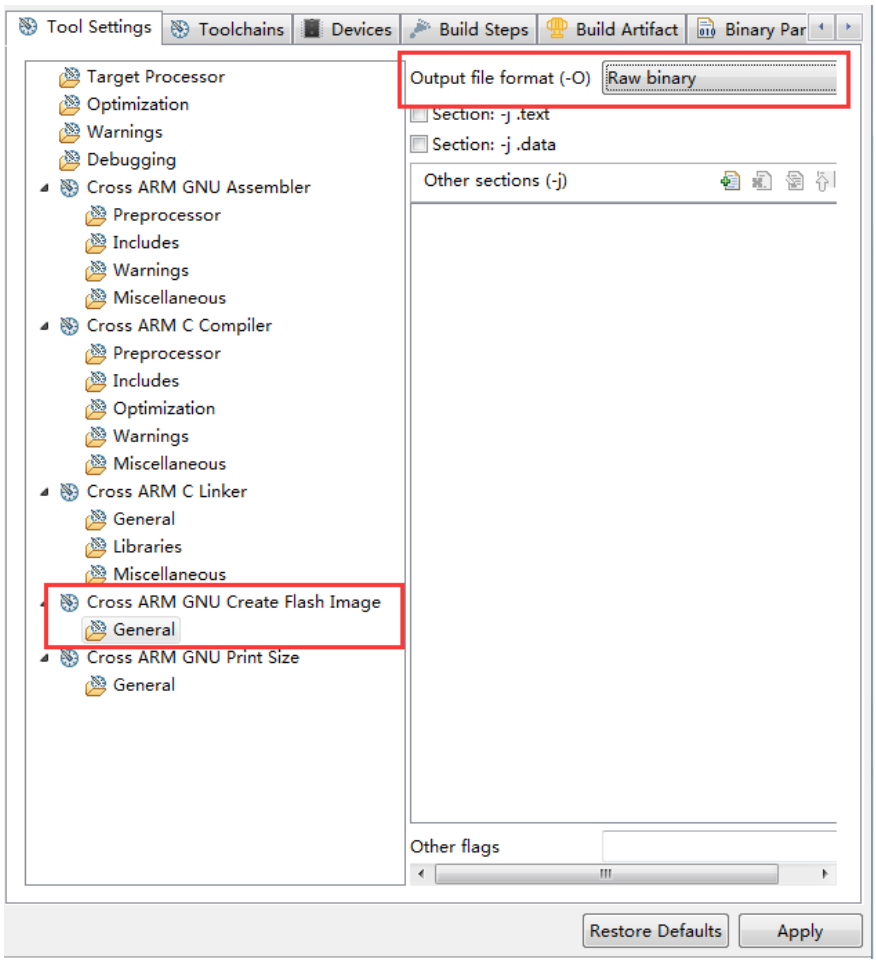
Figure 2-32 Configuring Cross ARM C Linker



Configure Cross ARM GNU Create Flash Image

Select General in the Cross ARM GNU Create Flash Image and configure the MCU image file format to be RAW binary, as shown in Figure 2-33.

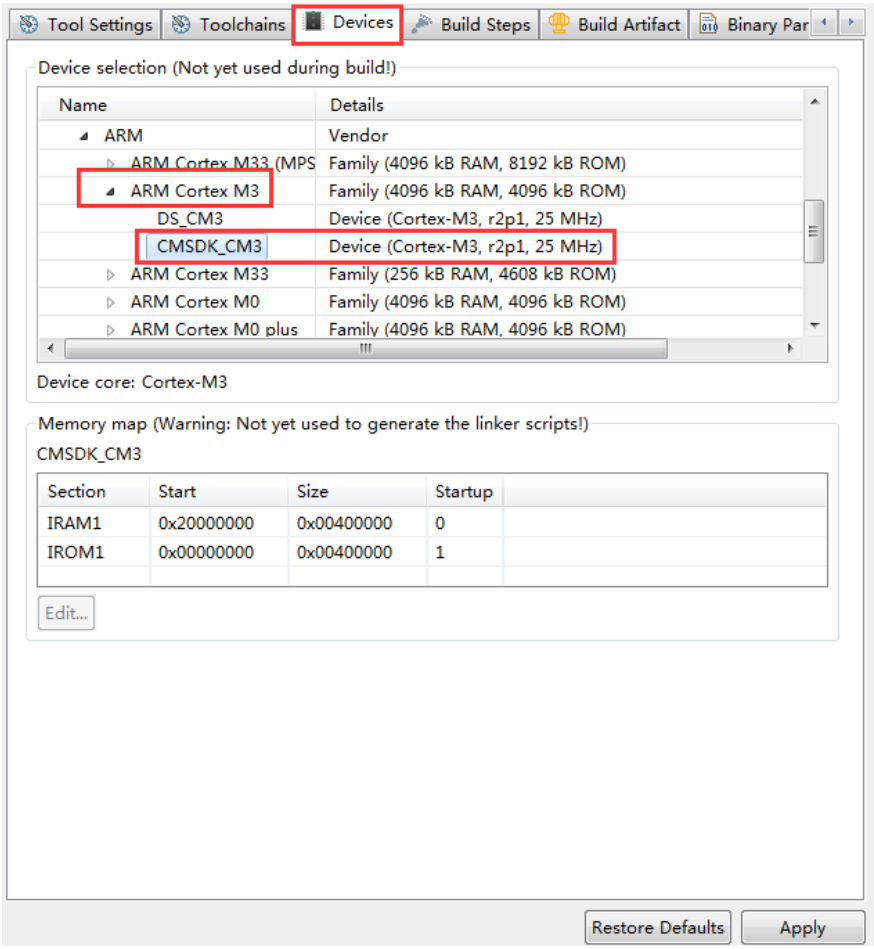
Figure 2-33 Configure Cross ARM GNU Create Flash Image e



Configure Devices

Select the Devices tab and select the device ARM Cortex-M3 “CMSDK_CM3” as shown in Figure 2-34.

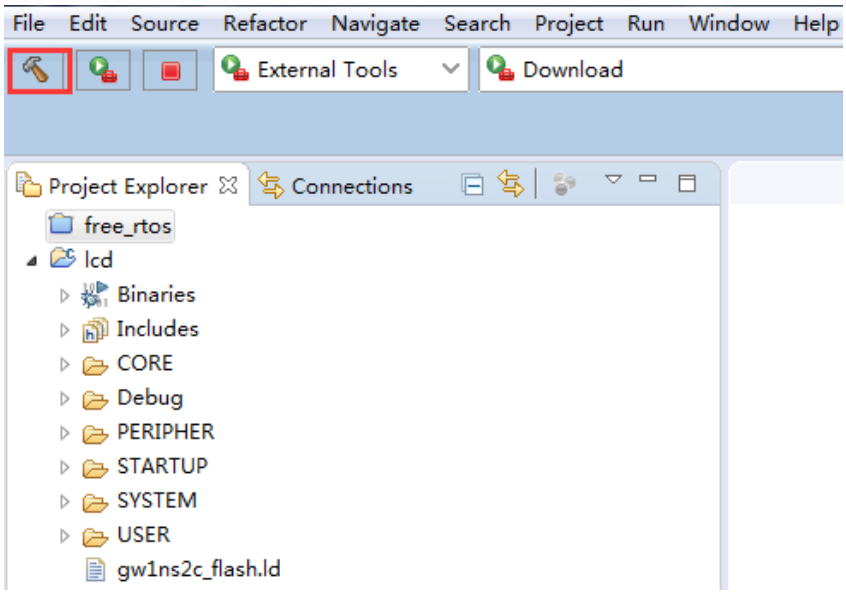
Figure 2-34 Configuring Devices



2.3.3 Project compilation

After completing the project configuration and coding, compile the project, select the toolbar and compile the button, and start compiling, as shown in Figure 2-35.

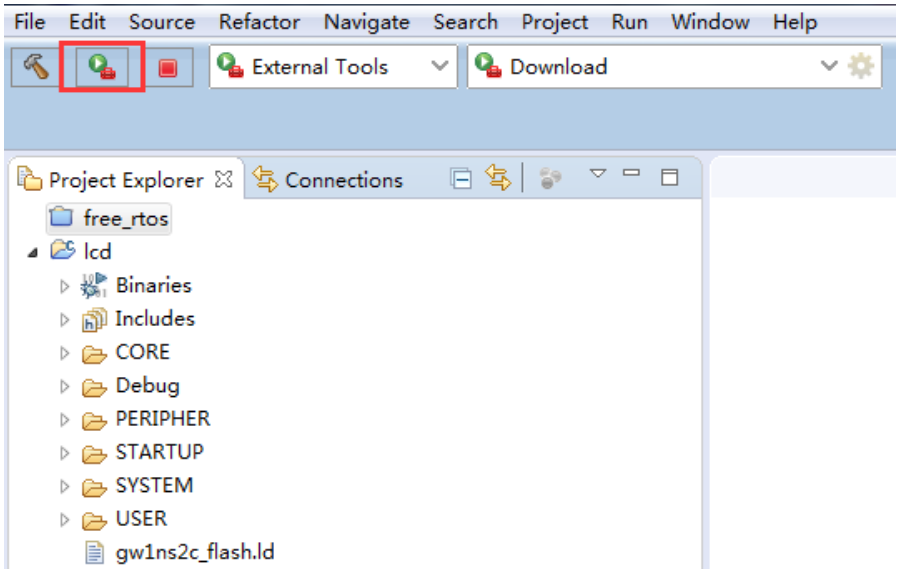
Figure 2-35 Compilation project



2.3.4 Project programmer

After the project is compiled, the generated image file is programmed into the chip, and the toolbar running button is selected to invoke the configured programmer tool, as shown in Figure 2-36.

Figure 2-36 Project programmer



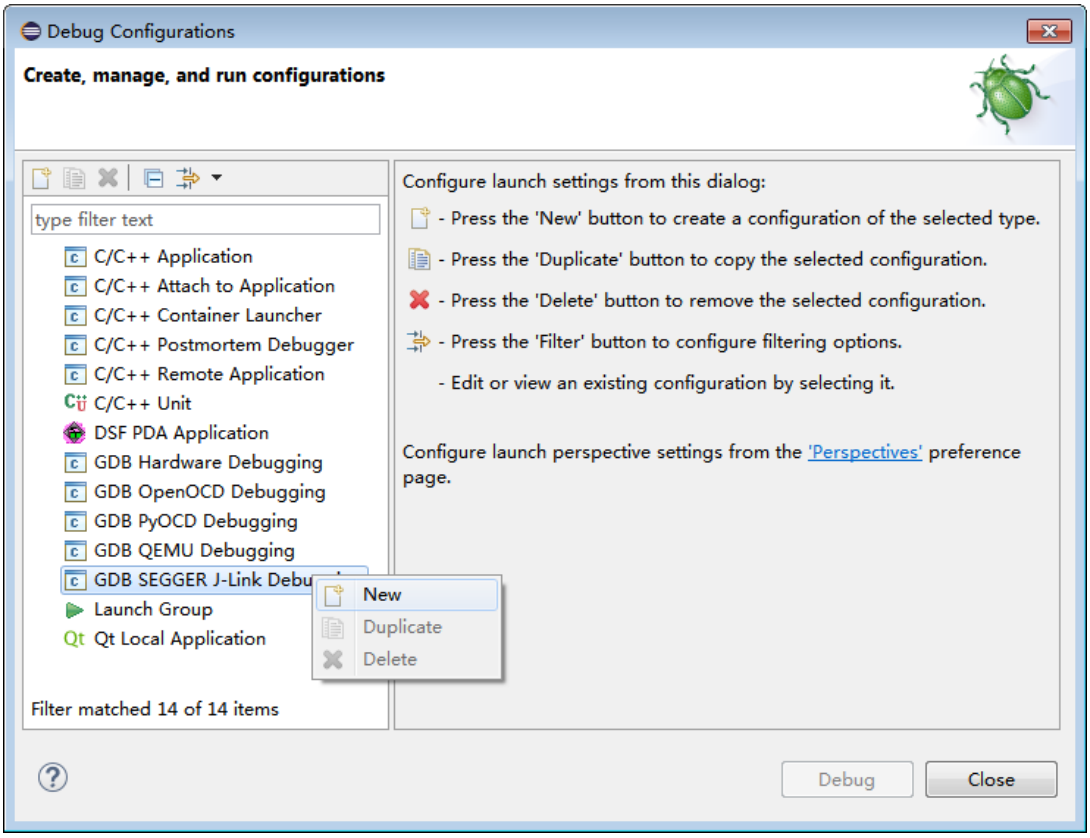
2.3.5 Project debugging

Configuring debug options

Select the GDB SEGGER J-Link Debugging tab in Debug

Configurations of Run in the menu bar, right-click and select New to create the debug configuration options for the project, as shown in Figure 2-37.

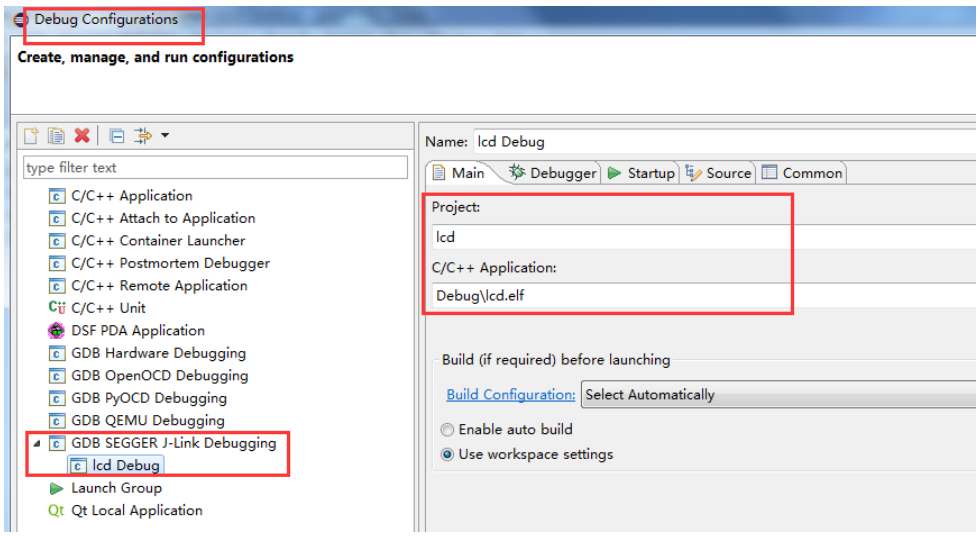
Figure 2-37 Establish debug configuration options



Select the established debug configuration option.

1. Select the Main tab and configure the path to the image file, as shown in Figure 2- 38.

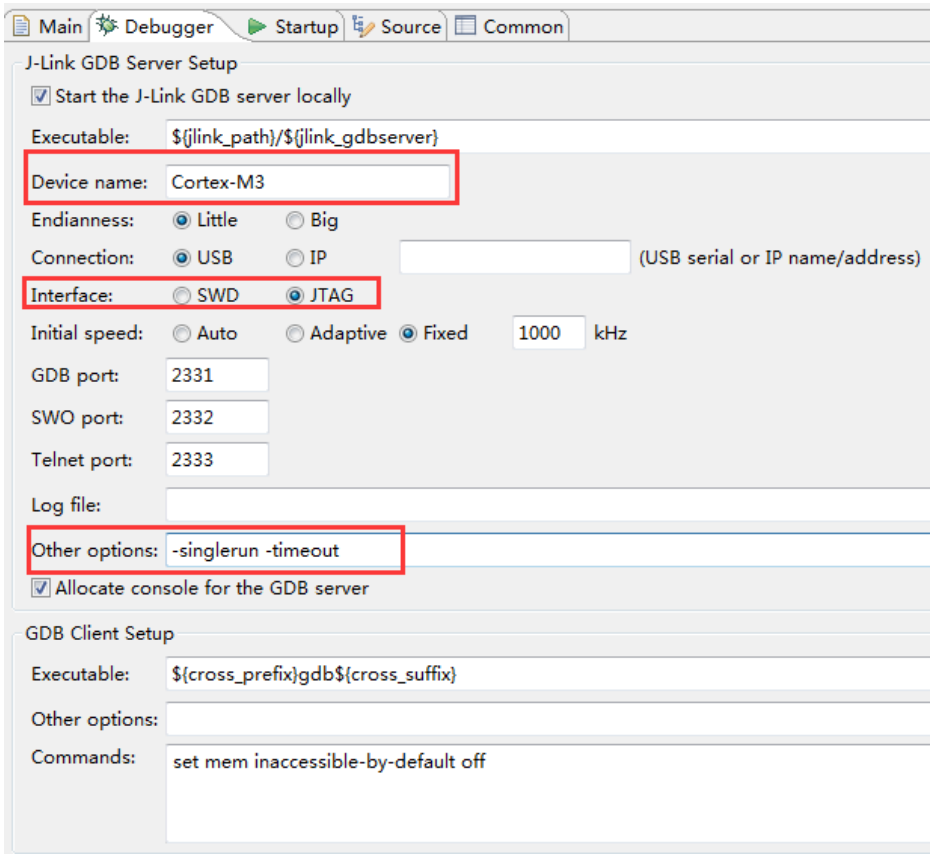
Figure 2-38 Configuring the Image Path



2. Select the Debugger tab and configure the Debugger device name and debug interface to JTAG, as shown in Figure 2-39.

- Device Name：Cortex-M3
- Interface：JTAG
- Other options：-singlerun -timeout

Figure 2-39 Configuring the Debugger



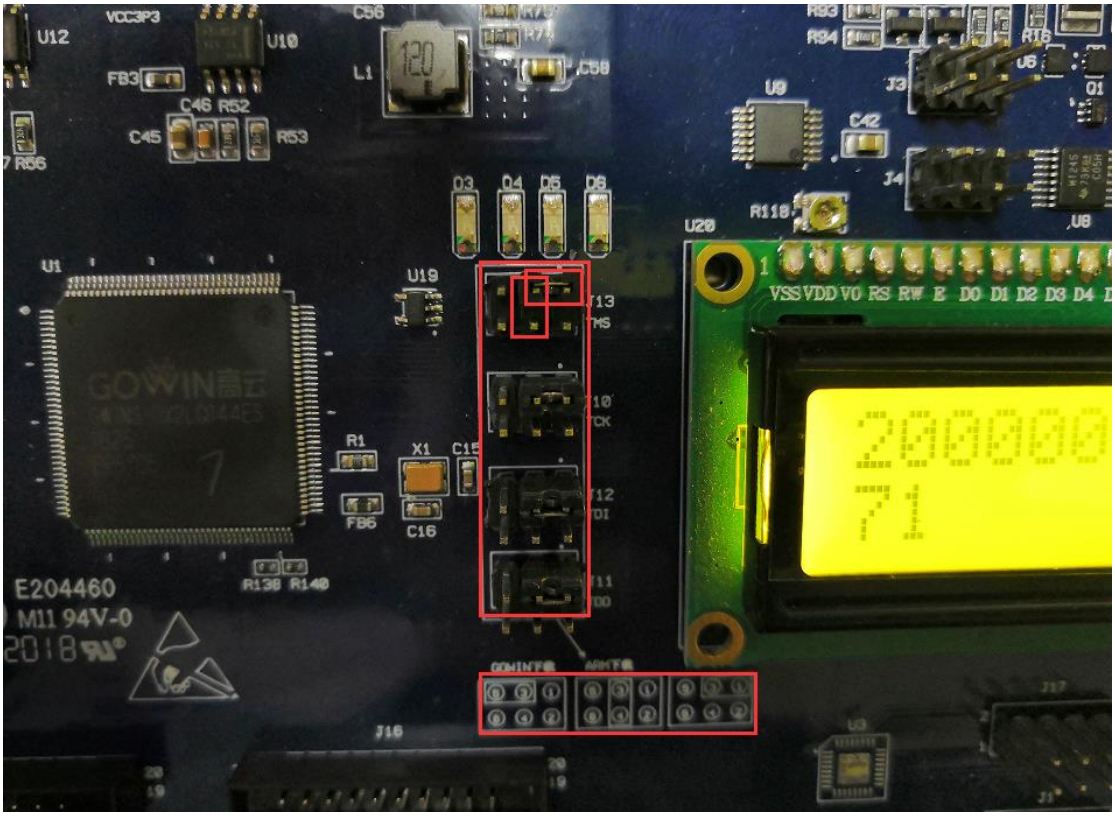
JTAG mode switching

After the debug mode configuration is completed, the JTAG mode switching tool Switch configured in the above Eclipse software configuration is invoked, and the JTAG software mode is switched from the programming mode to the debugging mode.

JTAG interface switching

After completing the JTAG software mode switch, if you use Gowin development board "DK-EVAL-GW1NS V1.1", it is necessary to switch the four jumper caps of the JTAG interface TMS, TCK, TDI, and TDO on the development board from the (1, 3) port to the (3, 4) port to complete the JTAG interface switching, as shown in Figure 2-40.

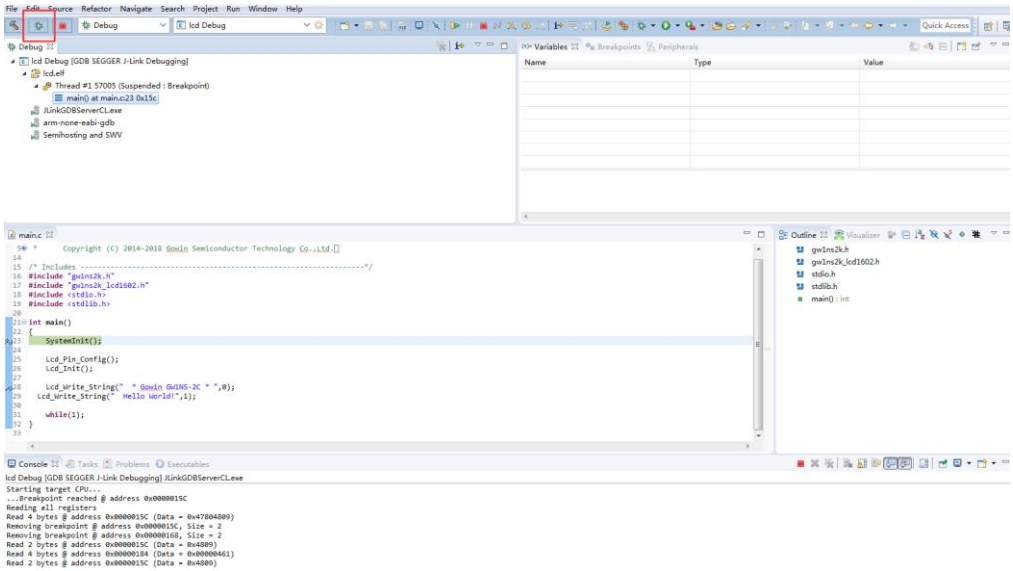
Figure 2-40 JTAG interface switching



Debugging

Connect the J-LINK emulator and select the Debug button on the toolbar to start debugging, as shown in Figure 2-41.

Figure 2-41 Startup debugging



2.4 Reference design

Gowin provides a reference design based on the GNU MCU Eclipse software:

Gowin GW1NS-2C_MCU_PACK\Gowin_GW1NS-2C_MCU_RefDesign\MCU_RefDesign\GNU


GOWIN
PROGRAMMING FOR FUTURE