PY32™ Microcontroller Development with GCC and EIDE

Introduction

This document describes the installation and use of the editor VSCode, the compiler GCC, and the debugger OpenOCD for the PY32 microcontroller.VSCode can realize the project management function similar to MDK/IAR and other IDE software by installing the EIDE extension, which can be configured visually, without the need to manually configure the Makefile.VSCode can realize the debugging function by installing the VSCode can realize debugging function by installing Cortex-Debug extension with JFlash or OpenOCD.

Table 1.

Type	Product Series
Microcontrollers	PY32F002A, PY32F002B, PY32F003, PY32F030 PY32F072, PY32F071, PY32F403

Contents

1   Installing GCC

1.1   downloading

Latest gcc-arm-none-eabi compiler download link: https://developer.arm.com/downloads/-/gnu-rm

1.2   mounting

Figure 1.2-1. Double-click gcc-arm-none-eabi-10.3-2021.10-win32.exe to start installation
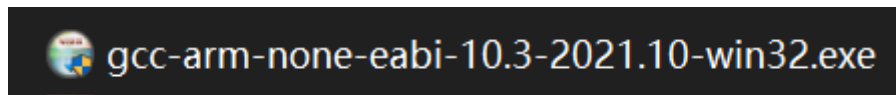

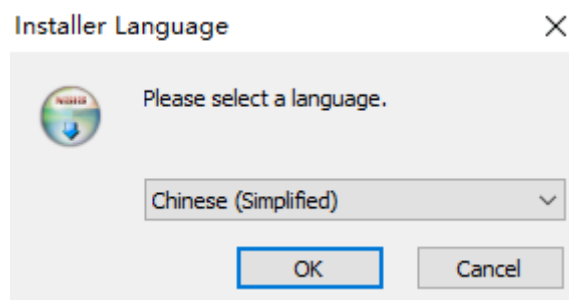Figure 1.2-2. Select Chinese (Simplified) and click OK.

Figure 1.2-3. Click the "Next" button.
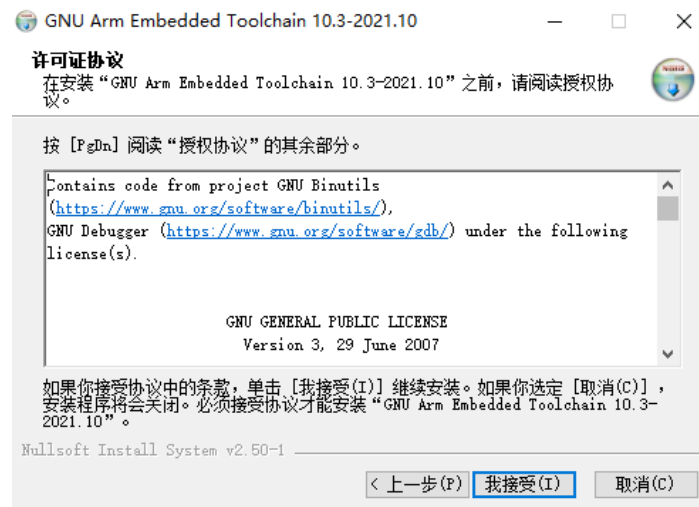
Figure 1.2-4. Click the "I Accept" button.

Figure 1.2-5. After selecting the path, click the "Install" button.
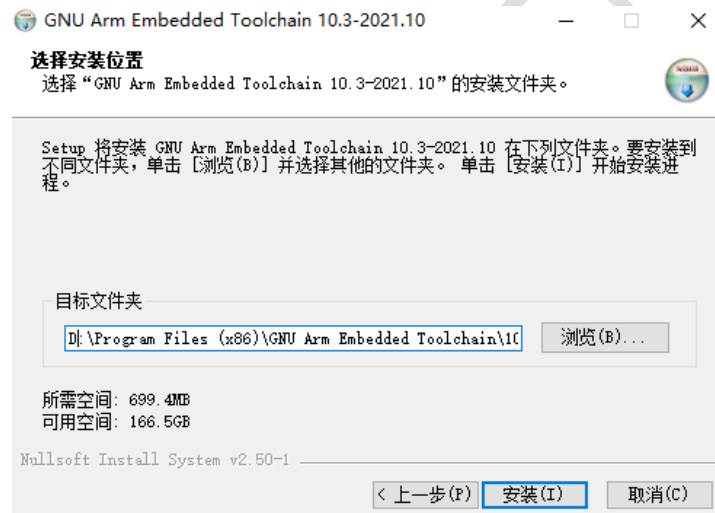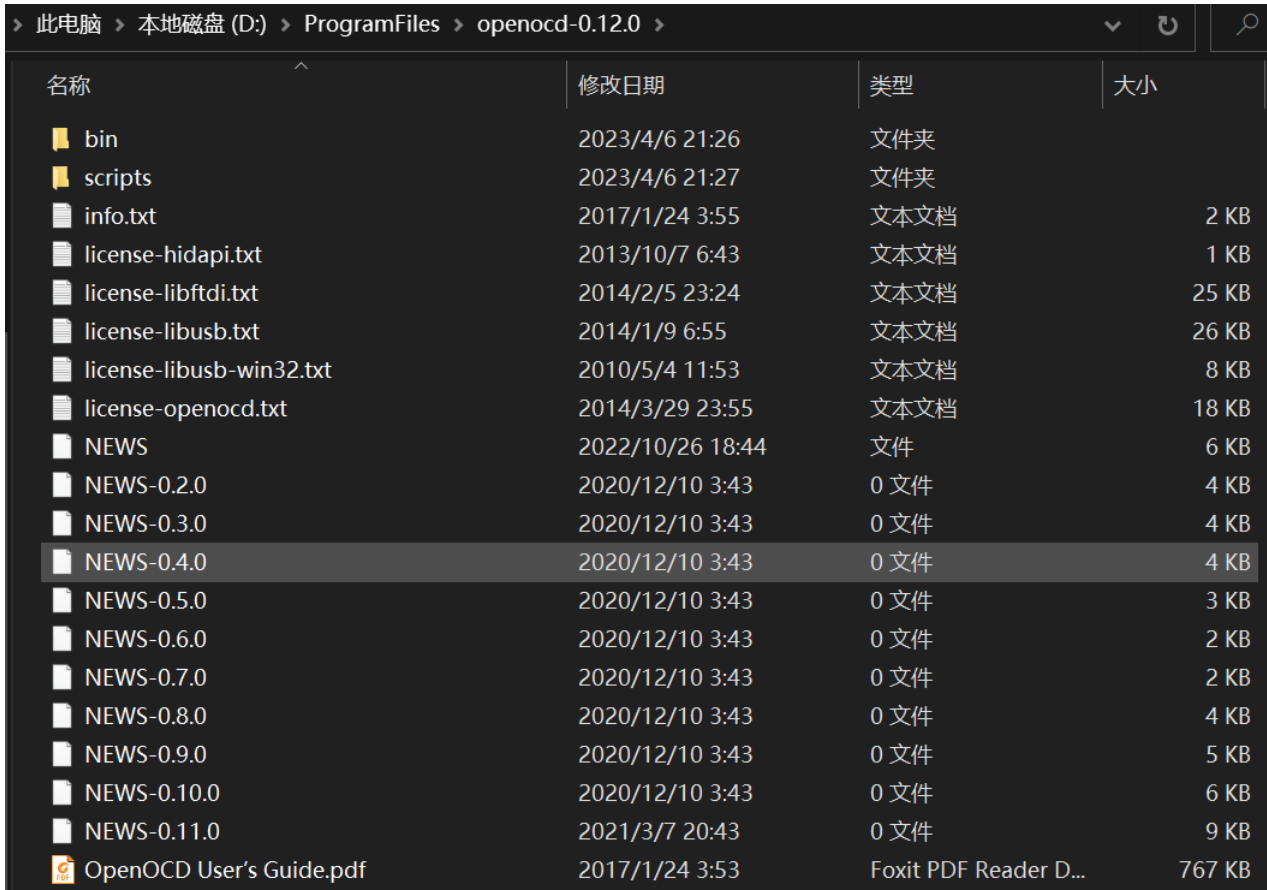

Figure 1.2-6. Check "Add path to environment variable", click "Finish" button.

2 Install openocd

This software is green installation-free software, can be used after unzipping, pay attention to unzip the path do not have spaces, Chinese and other special characters, such as D:\ProgramFiles.

Figure 2-1. Unzipping openocd-0.12.0.rar

3 Installing VSCode

3.1 Download VSCode
Latest VSCode software download link: https://code.visualstudio.com/Download

3.2 Installing VSCode

Figure 3.2-1. Double-click VSCodeUserSetup-x64-1.73.1.exe and follow the installation wizard to complete the installation.

名称	修改日期	类型	大小
VSCodeUserSetup-x64-1.73.1.exe	2022/11/11 10:45	应用程序	90,429 KB

3.3 Installing the Cortex-Debug extension

Figure 3.3-1. Enter "Cortex-Debug" in the search box of the VSCode Extension Store and click "Install".
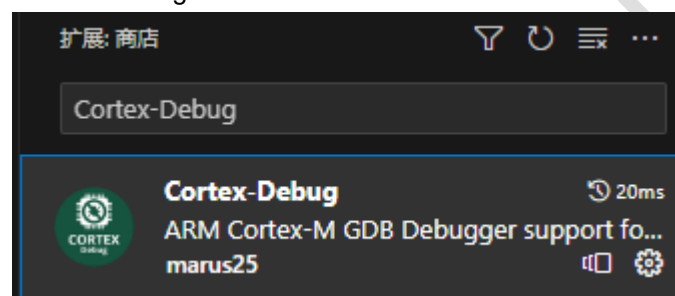
3.4 Installation of EIDE extensions

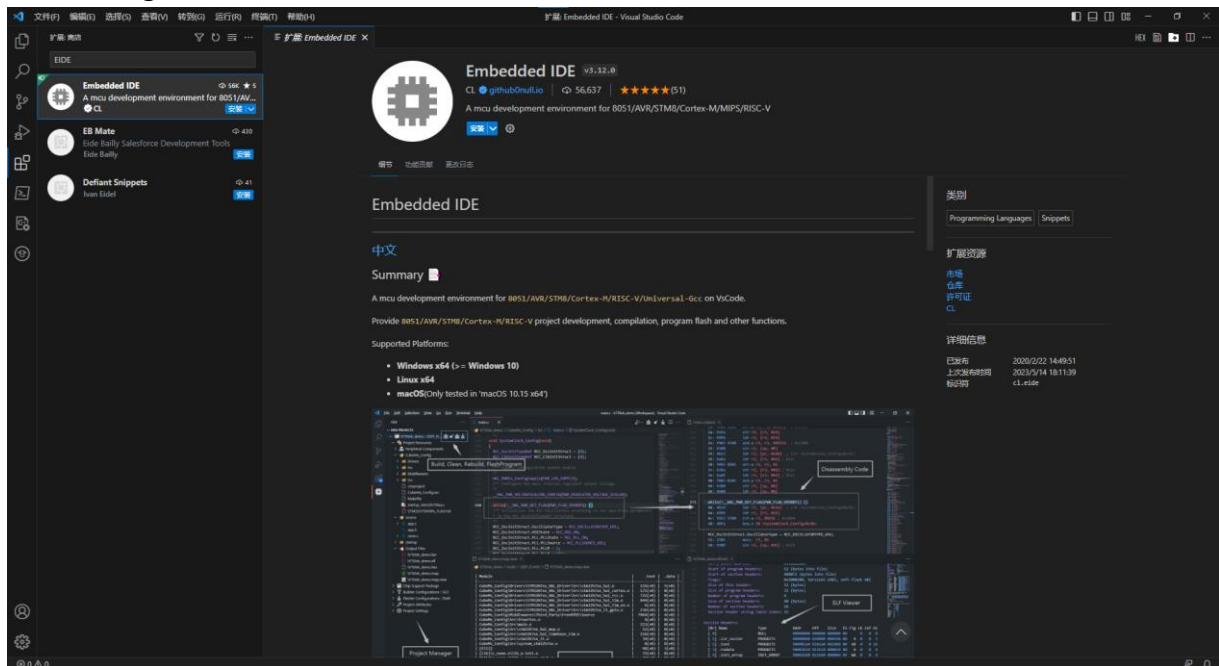Figure 3.4-1. Enter "EIDE" in the search box of VSCode Extension Store and click "Install".
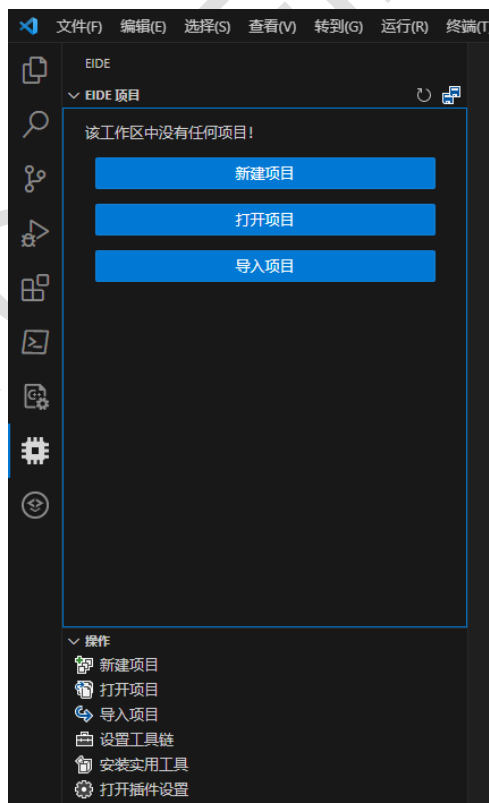

Figure 3.4-2. The "EIDE" option appears on the left side after the installation is completed.

4   Adding Environment Variables

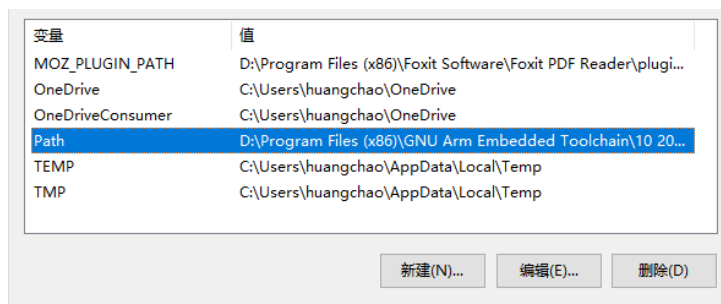Figure 4-1. Select the user variable "Path" and click the Edit button.

Figure 4-2. After adding the paths of gcc, mingw, openocd, and JFlash, click "OK".
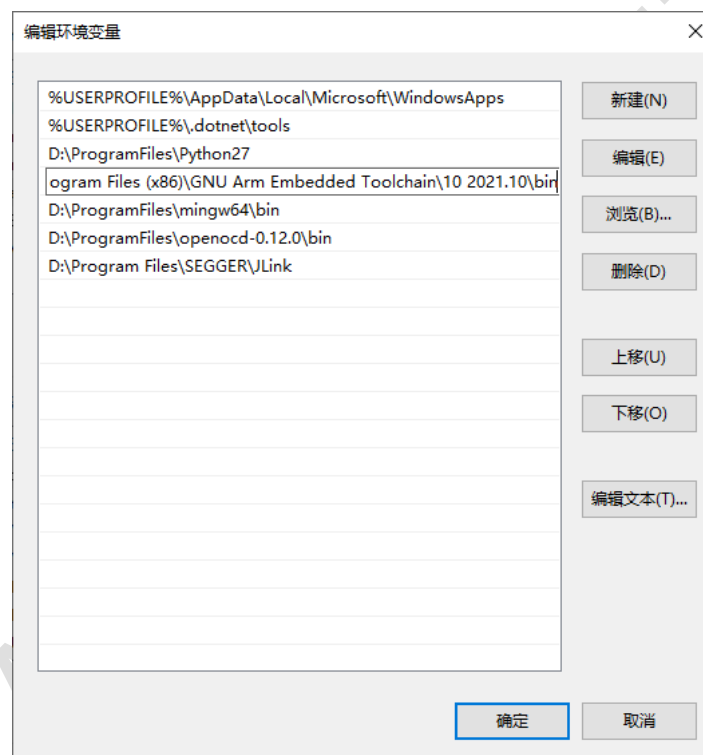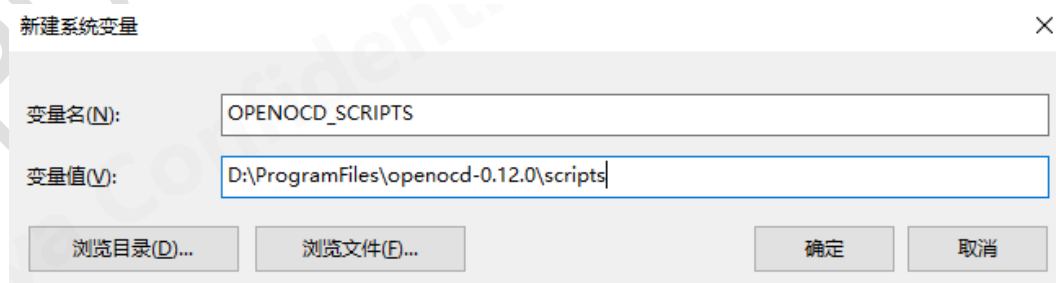
Figure 4-3. Adding System Variable，Name: OPENOCD_SCRIPTS，Value: openocd scripts folder

5   Software Usage

5.1   New EIDE project

Figure 5.1-1. Click the "New Project" button and select "Empty Project".
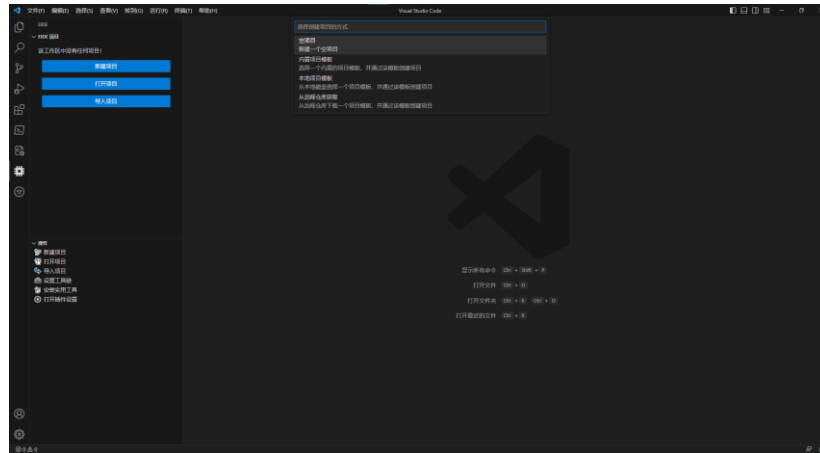

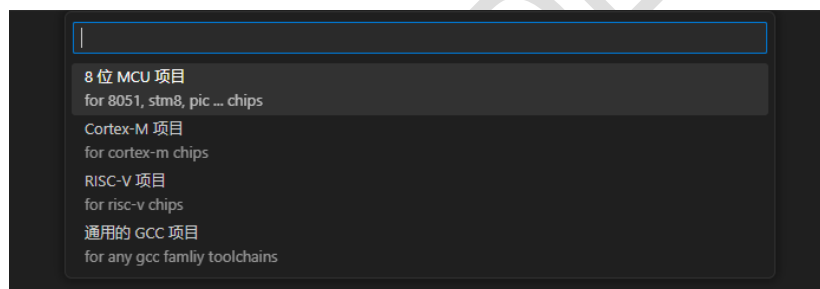Figure 5.1-2. Selecting the "Cortex-M Project".

Figure 5.1-3. Enter a project name to be created and press "Enter" to confirm.
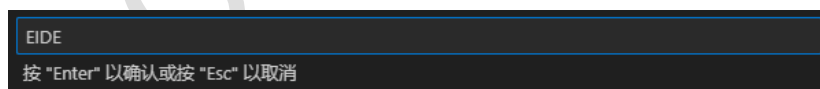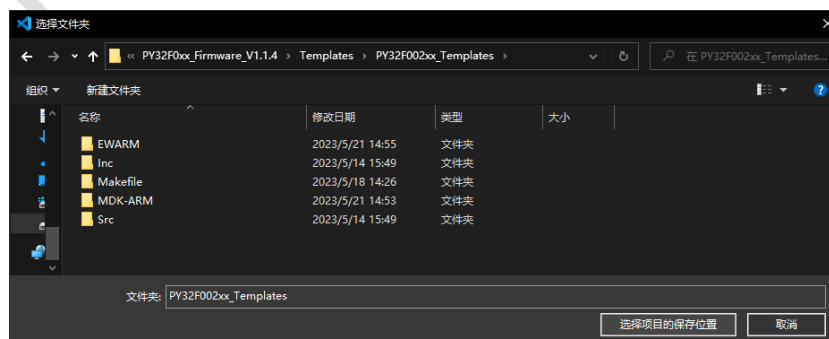

Figure 5.1-4. Selecting the save location for the project

5.2   Configuring the EIDE Project

Figure 5.2-1. Add the *.ld and *.s files provided by Puya to the EIDE folder.

5.2.1 Project resources

Figure 5.2.1-1. Adding a "virtual folder" to project resources.

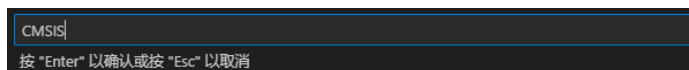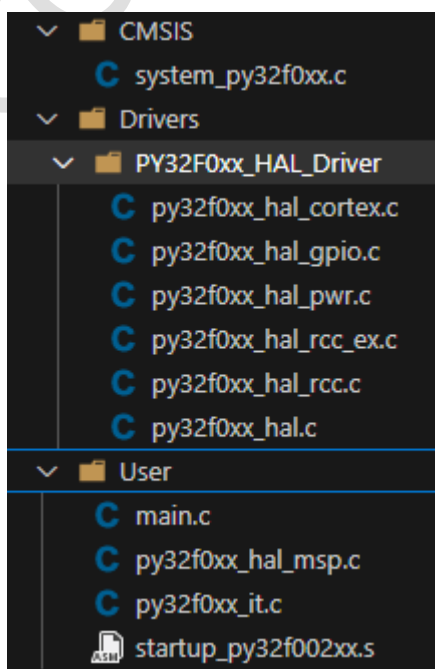Figure 5.2.1-2. Adding "CMSIS" virtual folder to the project resources.

Figure 5.2.1-3. Creating the Drivers, PY32F0xx_HAL_Driver, and User virtual folders in turn

Figure 5.2.1-4. Adding source files from "Src, EIDE, Drivers\PY32F0xx_HAL_Driver\Src" folder.

5.2.2 Chip Support Package

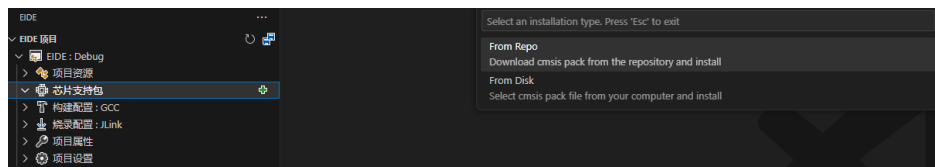Figure 5.2.2-1. Right-click on "Chip Support Package" and select "From Disk" on the right side.

Figure 5.2.2-2. Select MDK chip support package in the pop-up folder
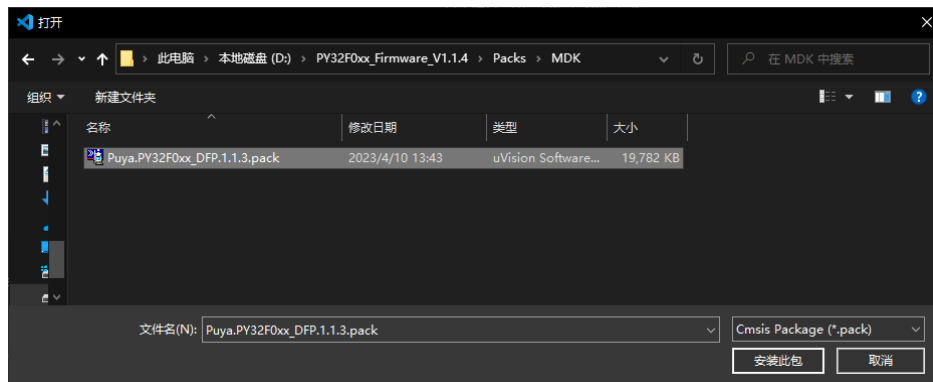

Figure 5.2.2-3. Click the "PY32F0xx_DFP" directory in the "Chip Support Packages" directory and
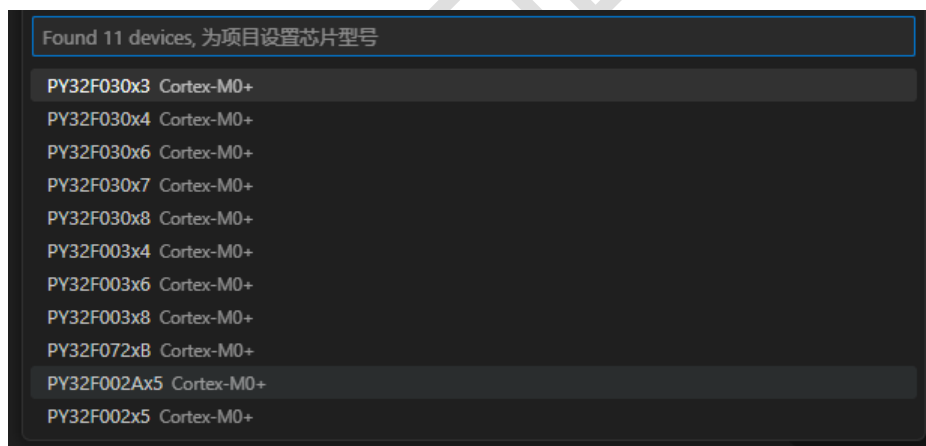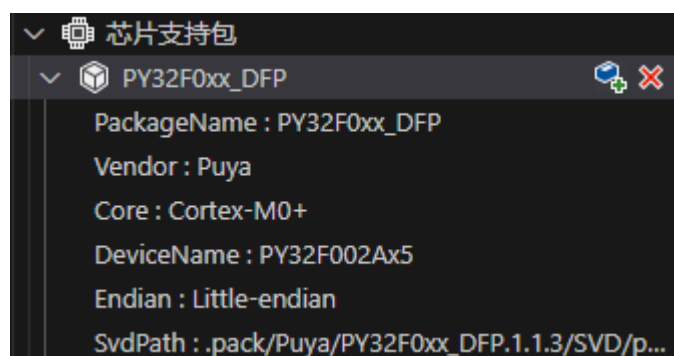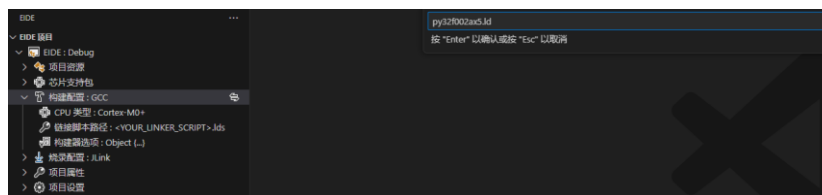
select the chip model.

Figure 5.2.2-4. Chip Support Package Configuration Complete

5.2.3 Build configuration: GCC

Figure 5.2.3-1. Click "Link Script Path:" and enter the link script file name.

5.2.4 Project Properties

Figure 5.2.4-1. Project Properties Configuration Include Directory
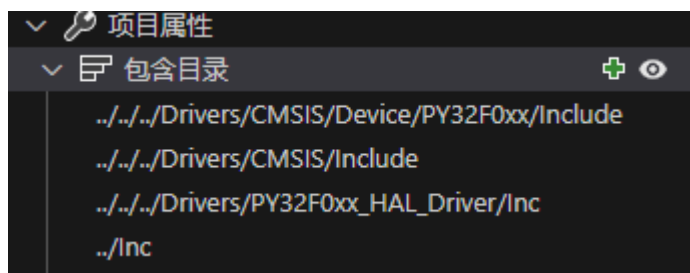
Figure 5.2.4-2. Project Properties Configuration Preprocessing Macro Definition
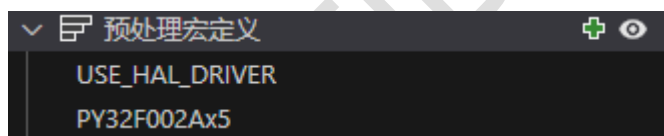
5.2.5 Builder Options

Figure 5.2.5-1. Additional compiler options can be added to the global options.

Add -mfloat-abi=hard -mfpu=vfpv4-d16 to make gcc generate floating point instructions (note:

only chips with FPUs are supported)

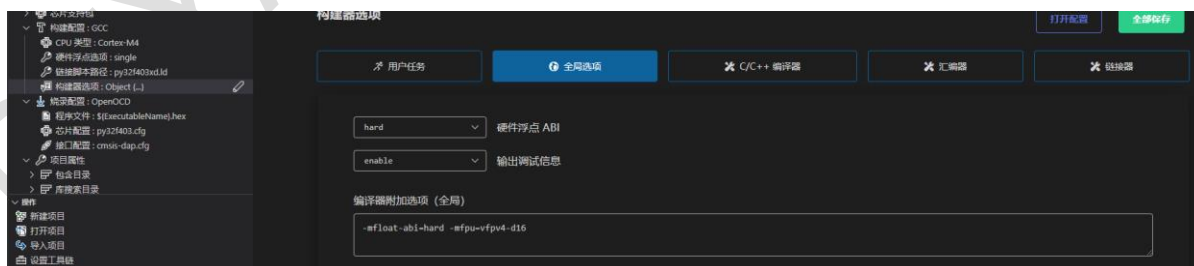Figure 5.2.5-2. Linker add-on options can be added to the connector

Add -u _printf_float to enable support for floating-point printing

✈ 用户任务	🌐 全局选项	✖ C/C++ 编译器	✖ 汇编器	✖ 链接器

ELF ⌄ 输出格式

☐ 不生成 Hex/Bin 文件

☑ Remove Unused Input Sections

☐ Not Print Memory Usage (disable: -Wl,--print-memory-usage)

链接器附加选项

```
--specs=nosys.specs --specs=nano.specs -u _printf_float
```

5.3   Compiling EIDE projects

Figure 5.3-1. Click the "Build" button or press "F7" to start compiling; the compilation

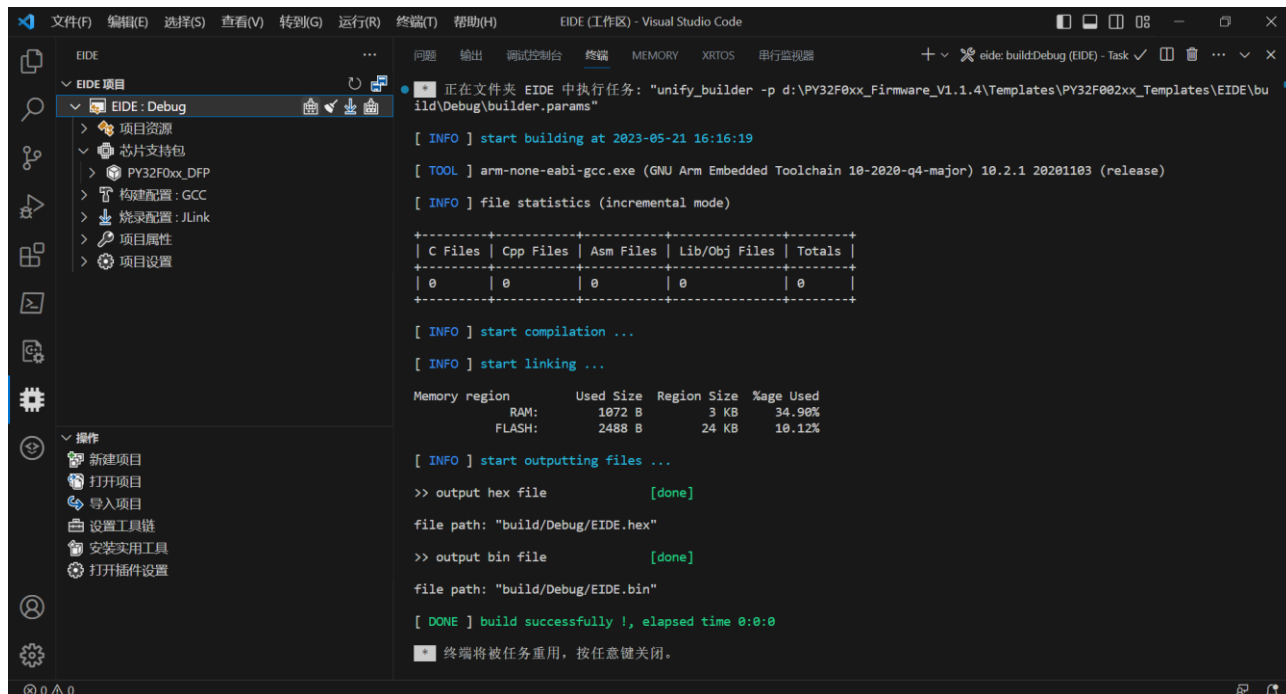information will be displayed when the compilation is completed.

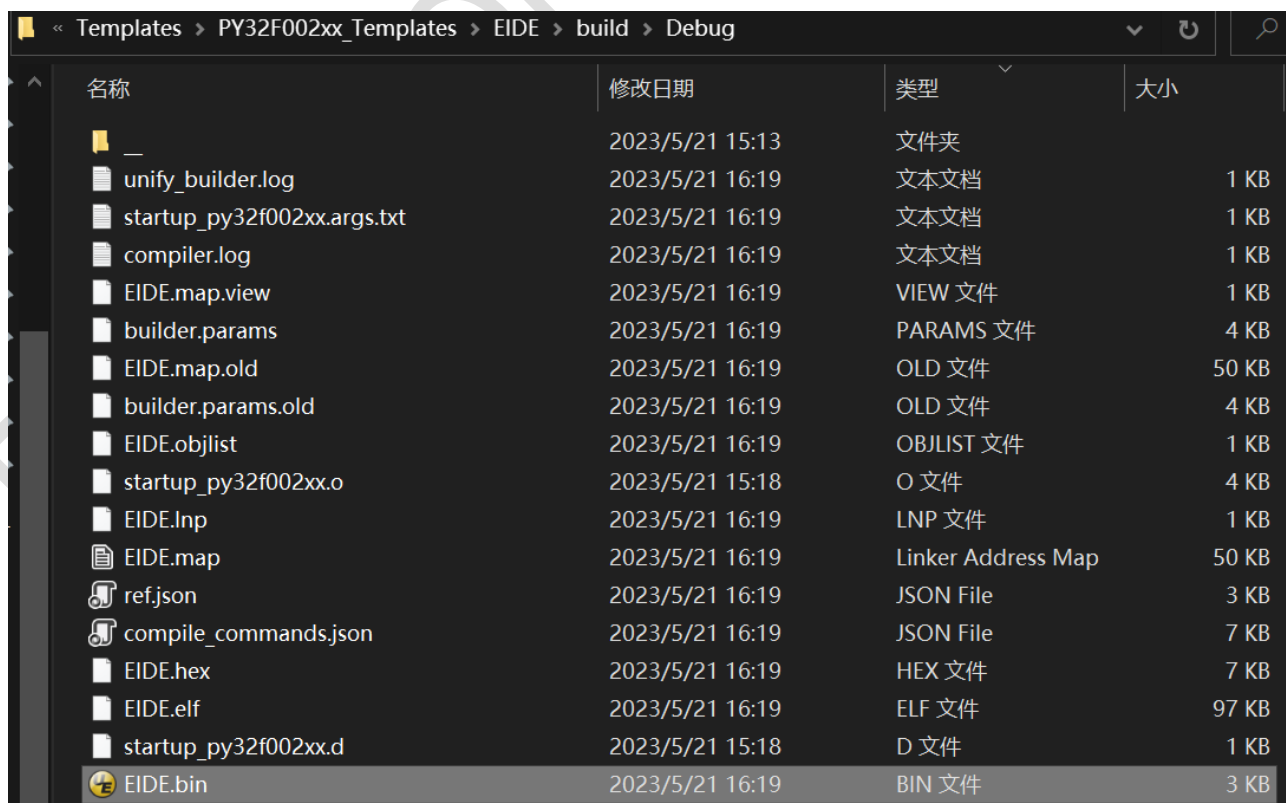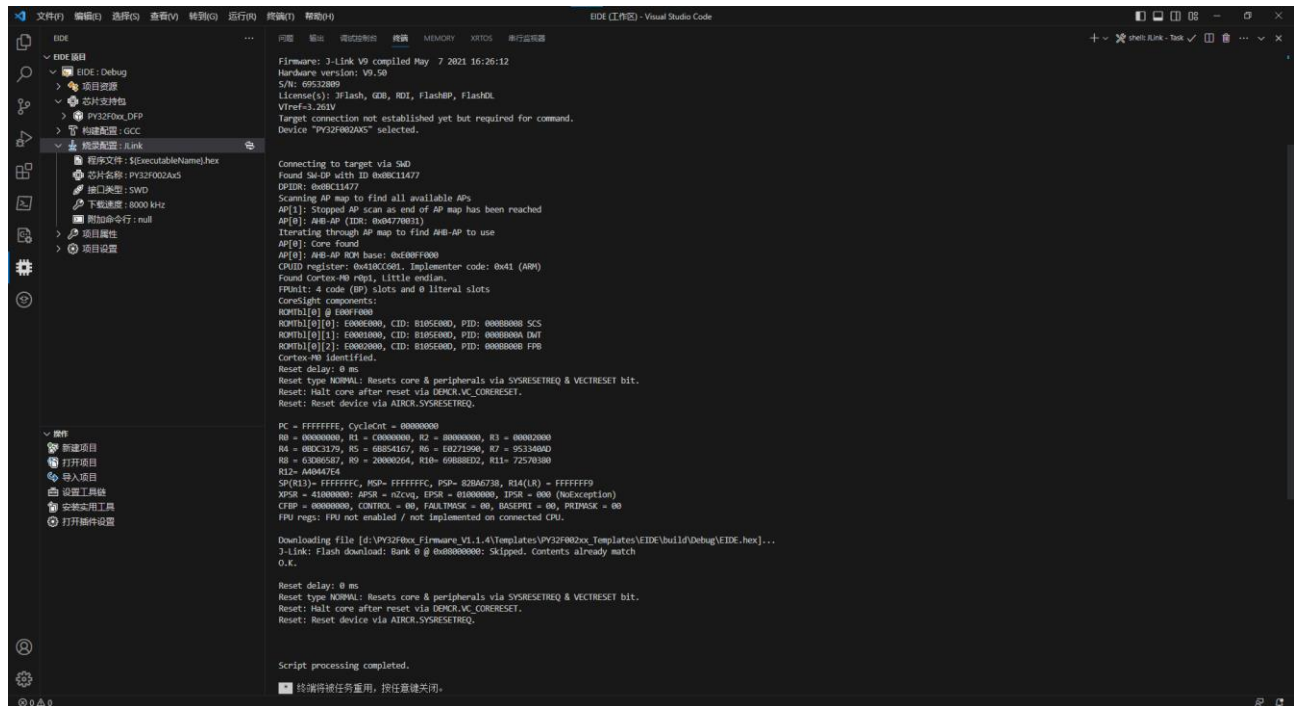Figure 5.3-2. The build directory generates target files in *.hex, *.elf, and *.bin formats.
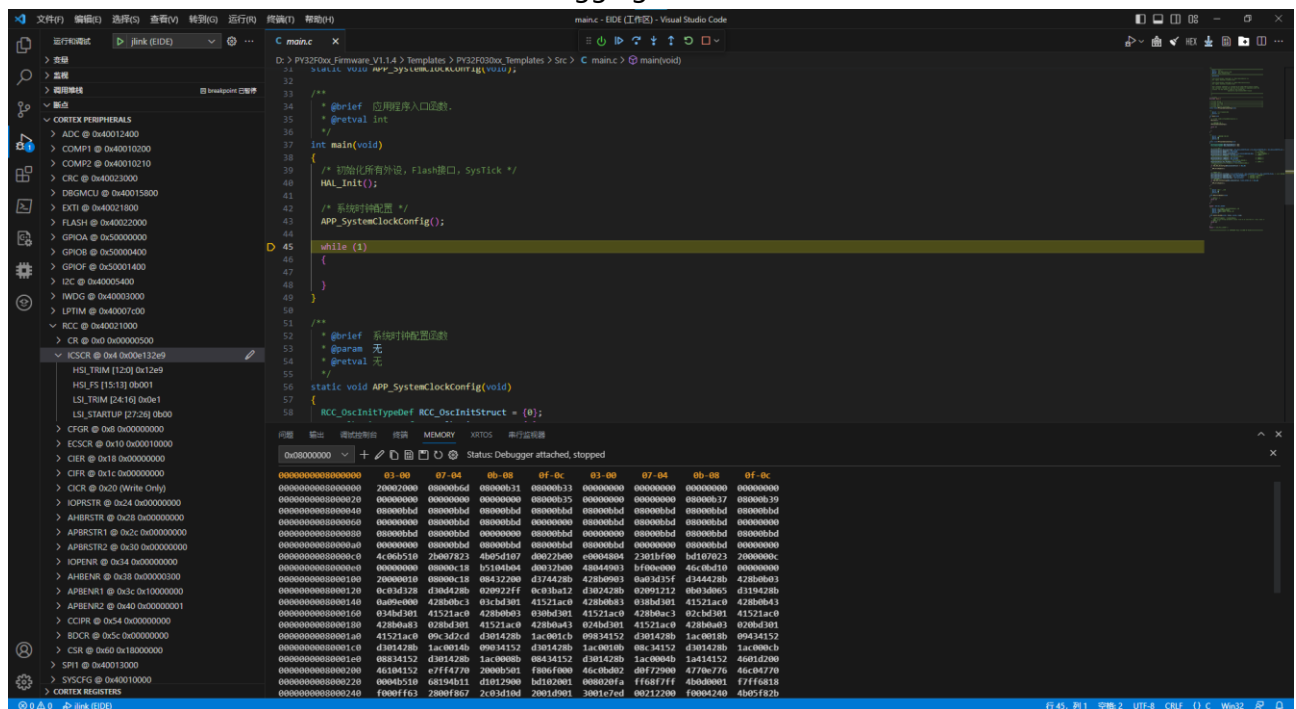
5.4 JLink Burning

Figure 5.4-2. Burning Configuration Select JLink, select the chip name, click the "Burn" button or

"Ctrl+Alt+D" to start burning.

5.5   JLink Debugging

Figure 5.5-1. Click "Run and Debug" and select the green triangle button in front of jlink(EIDE) to enter the debugging interface.

5.6   OpenOCD burning

Figure 5.6-1. The targets and interfaces folders in the openocd\scripts folder.

and mem_helper.tcl files to the EIDE project directory
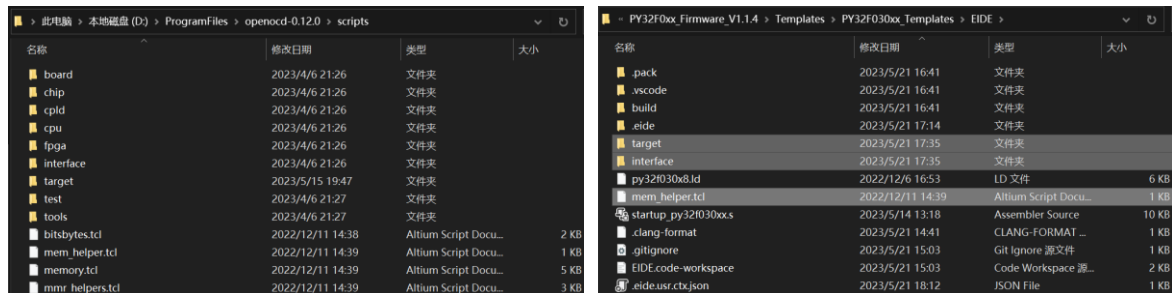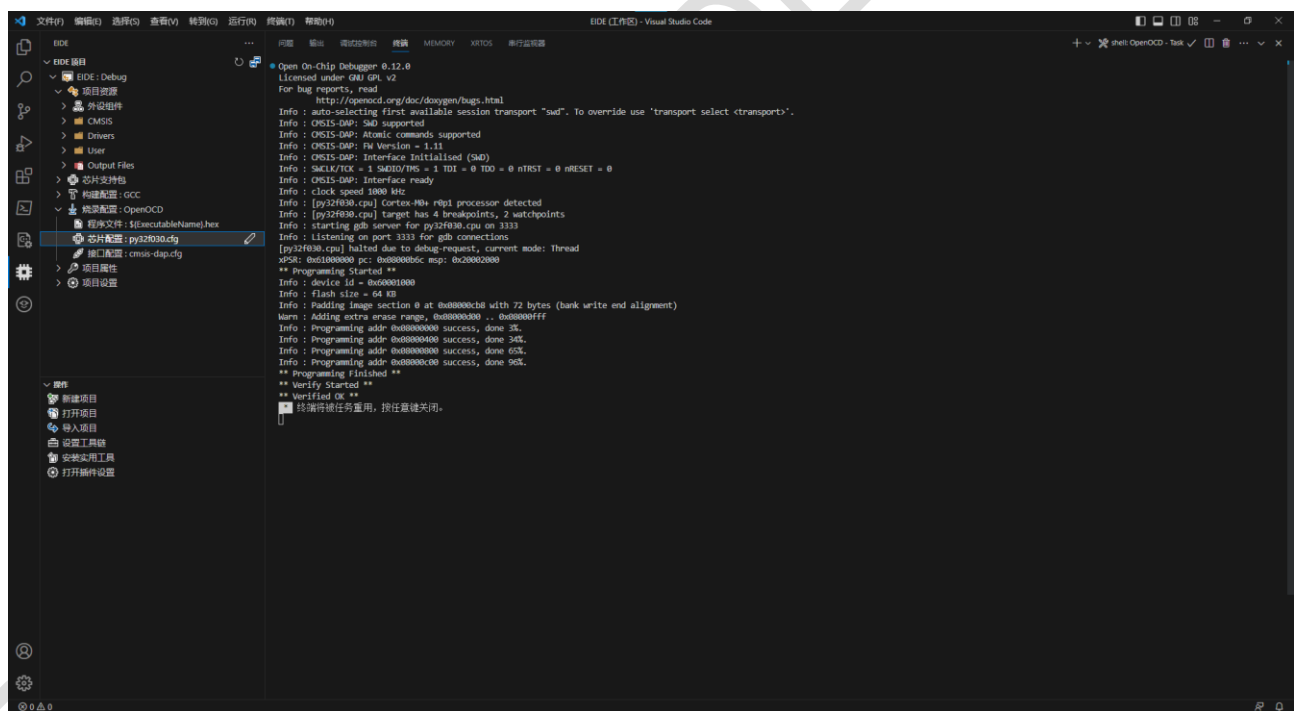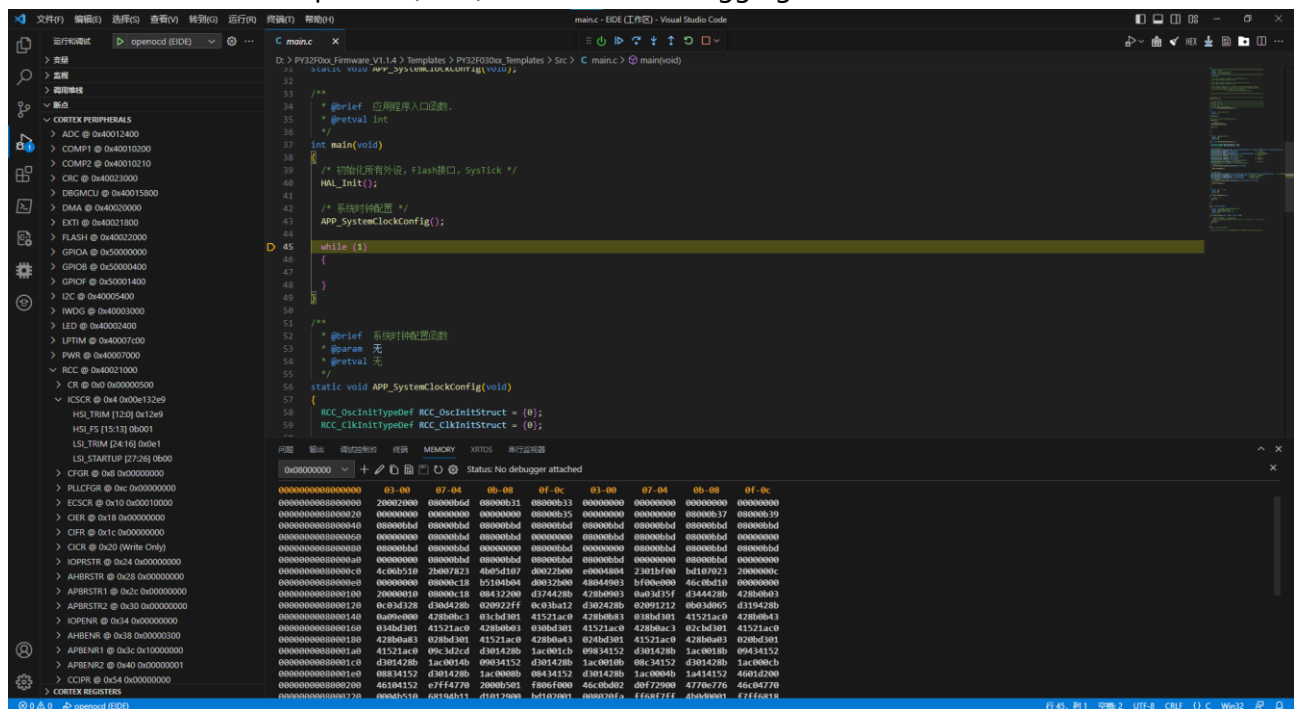

Figure 5.6-2. Burning Configuration Select OpenOCD and configure the "Chip Configuration"

and "Interface Configuration" according to the actual application.

Click the "Burn" button or "Ctrl+Alt+D" to start burning.

5.7 OpenOCD Debugging

Figure 5.7-1. Click "Run and Debug" and select the green triangle button in front of openocd(EIDE) to enter the debugging interface.

6   Version History

releases	dates	update a record
V1.0	2023.6.8	first edition

Puya Semiconductor Co.

IMPORTANT NOTICE
Puya Semiconductor reserves the right to make changes without further notice to any products or specifications herein. Puya Semiconductor does not Puya Semiconductor does not assume any responsibility for use of any its products for any particular purpose, nor does Puya Semiconductor assume any liability arising out of the application or use of any its products or circuits. Puya Semiconductor does not assume any responsibility for use of any of its products for any particular purpose, nor does Puya Semiconductor