UM0987 User manual

PY32™ Microcontroller Development with GCC and Makefile

preamble

This document is used to describe the installation and use of the GCC compilation environment for the PY32 microcontroller. The editing function is realized by using VSCode software; the compilation function is realized by using gcc-arm-none-eabi software; the download and burn function is realized by using CooFlash software and PY-LINK emulator, JFlash software and J-Link emulator.

Table 1.

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

Contents

1 Installing GCC 3
1.1 Download 3
1.2 Installation 3
2 Installing mingw 5
3 Installing VSCode 6
3.1 Download 6
3.2 Installation 6
4 Adding Environment Variables 7
5 Makefile files 8
6 *.ld link file 9
7 Edit, compile, download 10
7.1 VSCode Editor 10
7.2 GCC Compilation 11
7.3 CooFlash Software Download 11
7.4 JFlash Software Download 12
7.5 JFlash Command Line Download 13
7.6 OpenOCD Command Line Download 14
8 Creating and Using VSCode Tasks 15
8.1 Creating compile and download tasks - tasks.json 15
8.2 Creating a debug task - launch.json 18
8.3 Copy task folder - .vscode 20
9 Version History 21

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

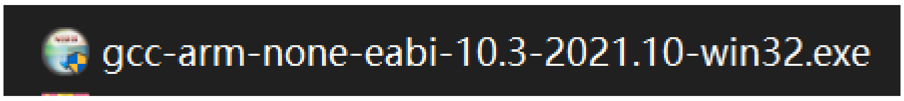
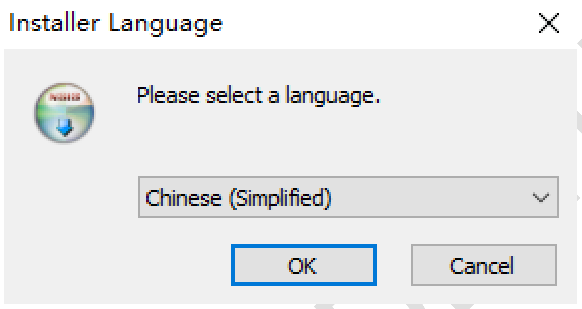

gcc-arm-none-eabi-10.3-2021.10-win32.exe

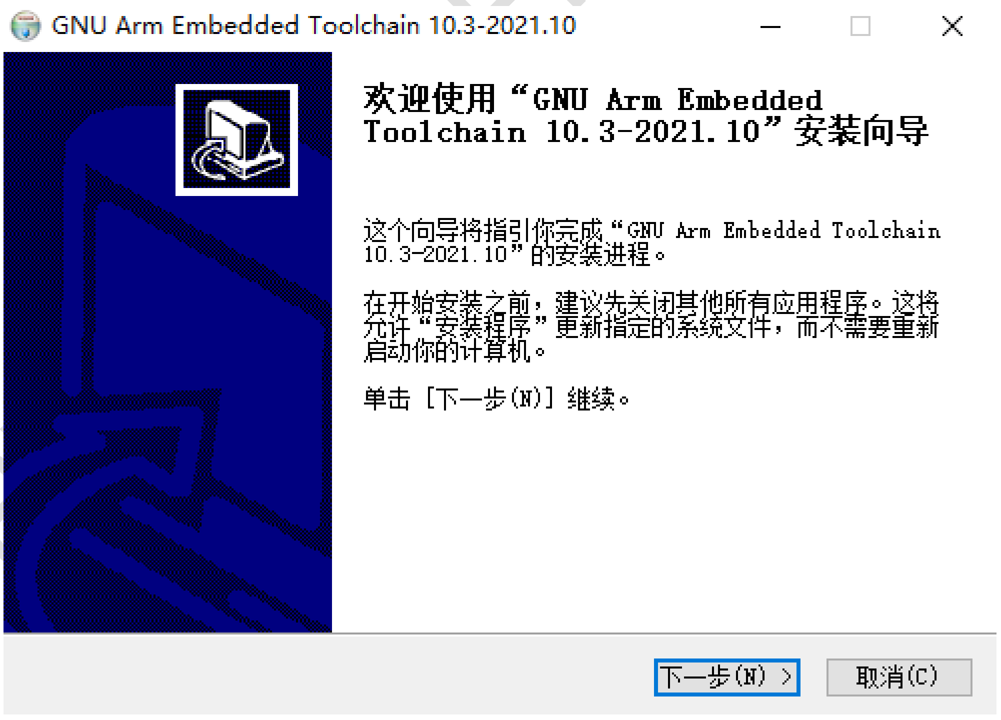
Figure 1.2-2. Select Chinese (Simplified) and click OK.


Installer Language

Please select a language.

Chinese (Simplified)

Figure 1.2-3. Click the "Next" button.


GNU Arm Embedded Toolchain 10.3-2021.10

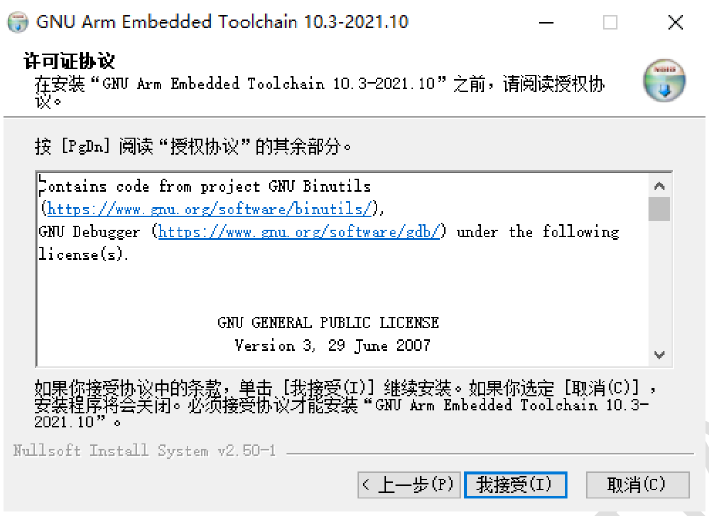
欢迎使用“GNU Arm Embedded Toolchain 10.3-2021.10”安装向导

这个向导将指引你完成“GNU Arm Embedded Toolchain 10.3-2021.10”的安装进程。

在开始安装之前，建议先关闭其他所有应用程序。这将允许“安装程序”更新指定的系统文件，而不需要重新启动你的计算机。

单击 [下一步(N)] 继续。

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



GNU Arm Embedded Toolchain 10.3-2021.10

许可证协议

在安装“GNU Arm Embedded Toolchain 10.3-2021.10”之前，请阅读授权协议。

按 [PgDn] 阅读“授权协议”的其余部分。

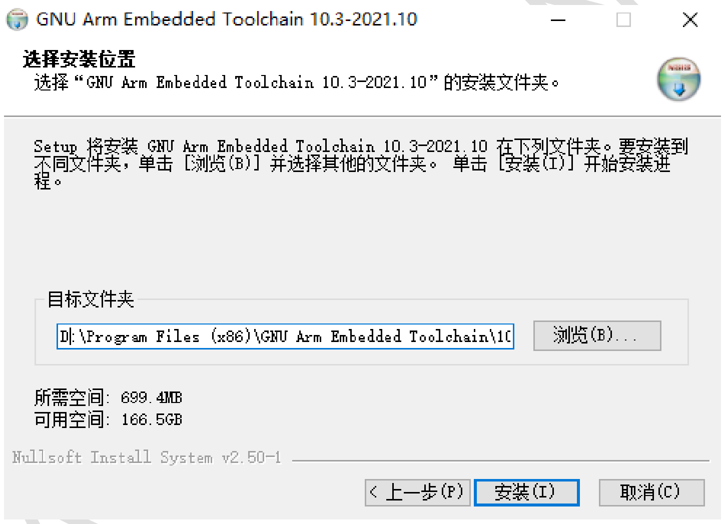
```
Contains code from project GNU Binutils
(https://www.gnu.org/software/binutils/),
GNU Debugger (https://www.gnu.org/software/gdb/) under the following
license(s).


                  GNU GENERAL PUBLIC LICENSE
                    Version 3, 29 June 2007
```

如果你接受协议中的条款，单击 [我接受(I)] 继续安装。如果你选定 [取消(C)]，安装程序将会关闭。必须接受协议才能安装“GNU Arm Embedded Toolchain 10.3-2021.10”。

Nullsoft Install System v2.50-1

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



GNU Arm Embedded Toolchain 10.3-2021.10

选择安装位置

选择“GNU Arm Embedded Toolchain 10.3-2021.10”的安装文件夹。

Setup 将安装 GNU Arm Embedded Toolchain 10.3-2021.10 在下列文件夹。要安装到不同文件夹，单击 [浏览(B)] 并选择其他的文件夹。 单击 [安装(I)] 开始安装进程。

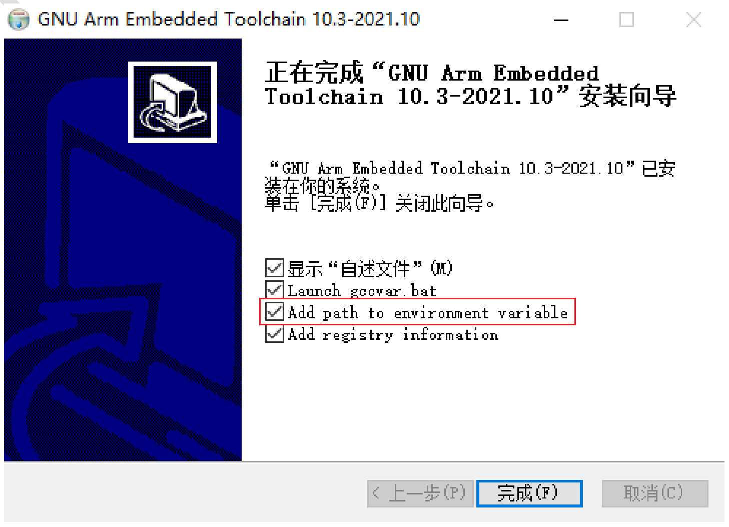
目标文件夹

D:\Program Files (x86)\GNU Arm Embedded Toolchain\1(

所需空间: 699.4MB
可用空间: 166.5GB

Nullsoft Install System v2.50-1

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



GNU Arm Embedded Toolchain 10.3-2021.10

正在完成“GNU Arm Embedded Toolchain 10.3-2021.10”安装向导

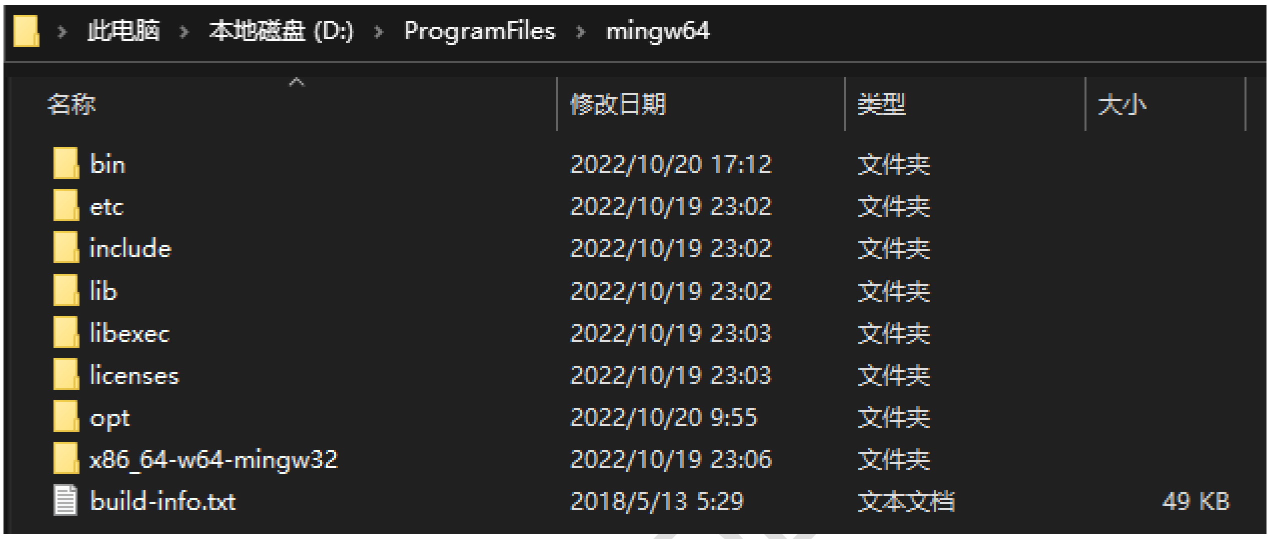
“GNU Arm Embedded Toolchain 10.3-2021.10”已安装在你的系统。
单击 [完成(F)] 关闭此向导。

- [✓] 显示“自述文件”(M)
- [✓] Launch gccvar.bat
- [✓] Add path to environment variable
- [✓] Add registry information

2 Install mingw

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. Extracting mingw64.rar



此电脑 › 本地磁盘 (D:) › ProgramFiles › mingw64

名称	修改日期	类型	大小
bin	2022/10/20 17:12	文件夹	
etc	2022/10/19 23:02	文件夹	
include	2022/10/19 23:02	文件夹	
lib	2022/10/19 23:02	文件夹	
libexec	2022/10/19 23:03	文件夹	
licenses	2022/10/19 23:03	文件夹	
opt	2022/10/20 9:55	文件夹	
x86_64-w64-mingw32	2022/10/19 23:06	文件夹	
build-info.txt	2018/5/13 5:29	文本文档	49 KB

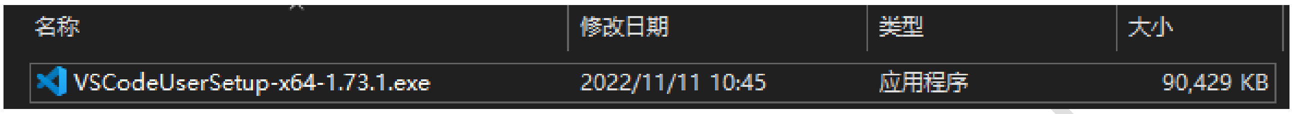
3 Installing VSCode

3.1 downloading

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

3.2 mounting

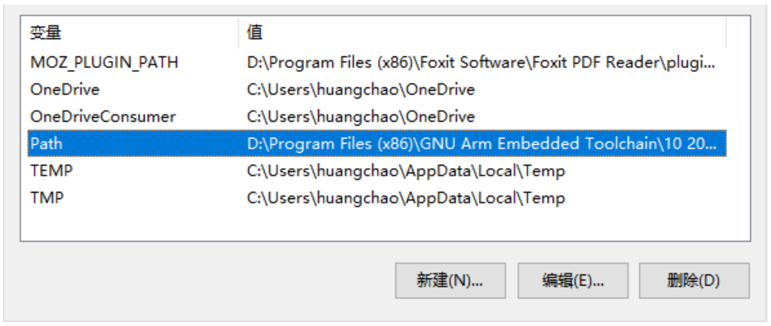
Figure 1.2-1. Double-click VSCodeUserSetup-x64-1.73.1.exe and follow the installation wizard to complete the installation.



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

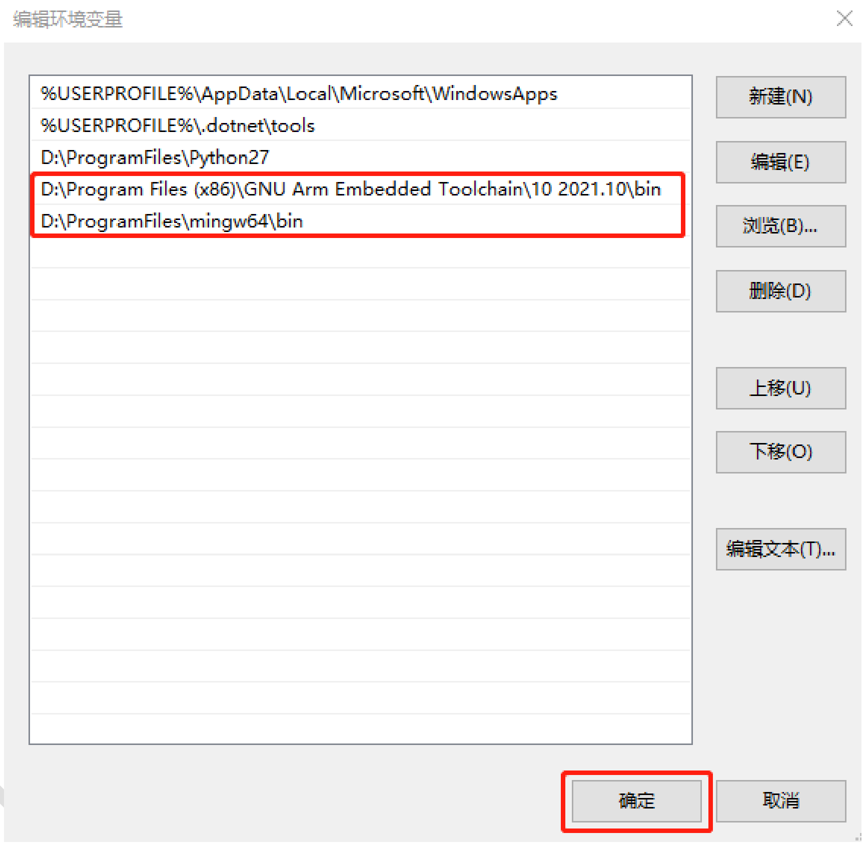
4 Adding Environment Variables

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



变量	值
MOZ_PLUGIN_PATH	D:\Program Files (x86)\Foxit Software\Foxit PDF Reader\plugi...
OneDrive	C:\Users\huangchao\OneDrive
OneDriveConsumer	C:\Users\huangchao\OneDrive
Path	D:\Program Files (x86)\GNU Arm Embedded Toolchain\10 20...
TEMP	C:\Users\huangchao\AppData\Local\Temp
TMP	C:\Users\huangchao\AppData\Local\Temp

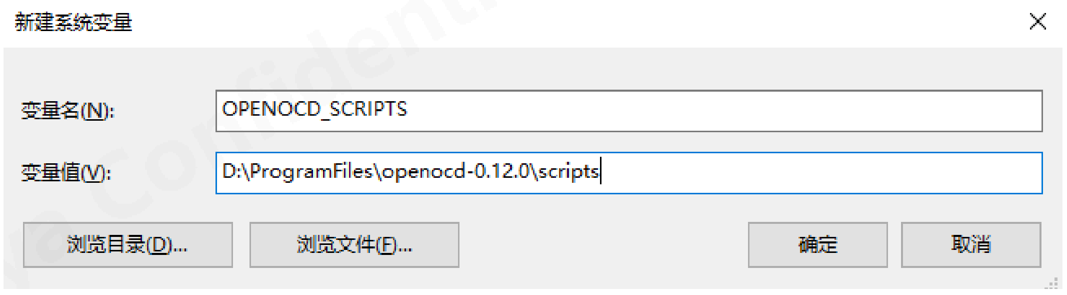
Figure 4-2. After adding the paths for gcc and mingw, click the "OK" button.



编辑环境变量

```
%USERPROFILE%\AppData\Local\Microsoft\WindowsApps
%USERPROFILE%\.dotnet\tools
D:\ProgramFiles\Python27
D:\Program Files (x86)\GNU Arm Embedded Toolchain\10 2021.10\bin
D:\ProgramFiles\mingw64\bin
```

Figure 4-3. Adding System Variable，Name: OPENOCD_SCRIPTS，Value: openocd scripts folder



新建系统变量

变量名(N):	OPENOCD_SCRIPTS
变量值(V):	D:\ProgramFiles\openocd-0.12.0\scripts

5 Makefile

name (of a thing)	clarification
TARGET	Name of the generated target file (hex/bin/elf)
OPT	Compile Optimization Level
BUILD_DIR	Path to the generated target file
C_SOURCES	List of C source files to be compiled
ASM_SOURCES	List of S assembly files to be compiled
C_DEFS	Preprocessing Macro Definitions
AS_INCLUDES	Compilation files contain directories
C_INCLUDES	C source files contain directories

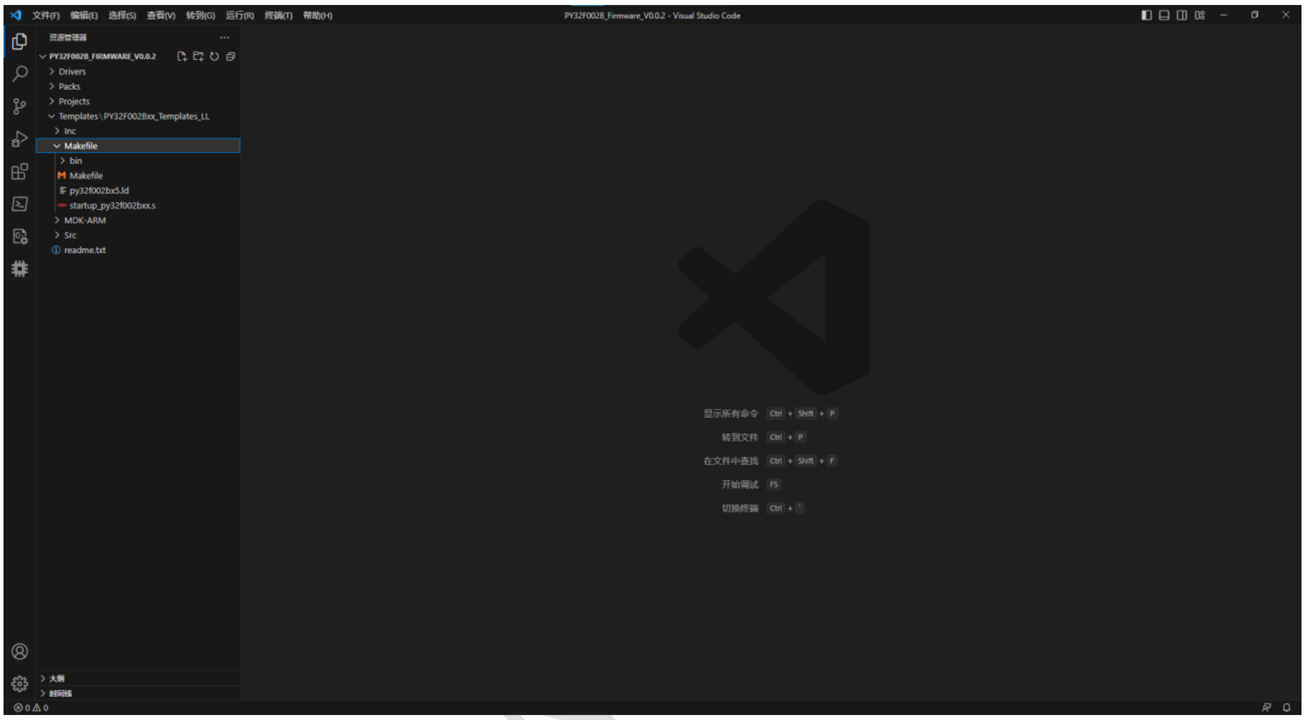
6 *.ld link file

name (of a thing)	clarification
_Min_Heap_Size	Setting the heap size
_Min_Stack_Size	Setting the stack size
RAM (xrw)	SRAM starting address and size
FLASH (rx)	FLASH start address and size

7 Edit, compile, download

7.1 VSCode Editor

Figure 7.1-1. Unzip PY32F002B_Firmware_V0.0.2.rar first, and then open the unzipped folder using VSCode


文件(F) 编辑(E) 选择(S) 查看(V) 转到(G) 运行(R) 终端(T) 帮助(H)
PY32F002B_Firmware_V0.0.2 - Visual Studio Code
资源管理器
PY32F002B_FIRMWARE_V0.0.2
Drivers
Packs
Projects
Templates \ PY32F002Bxx_Templates_LL
Inc
Makefile
bin
Makefile
py32f002bx5.ld
startup_py32f002bxx.s
MDK-ARM
Src
readme.txt
显示所有命令 Ctrl + Shift + P
转到文件 Ctrl + P
在文件中查找 Ctrl + Shift + F
开始调试 F5
切换终端 Ctrl + `
大纲
时间线

7.2 GCC Compilation

Figure 7.2-1. Right-click on the Makefile folder to be compiled and click "Open in Integrated Terminal".

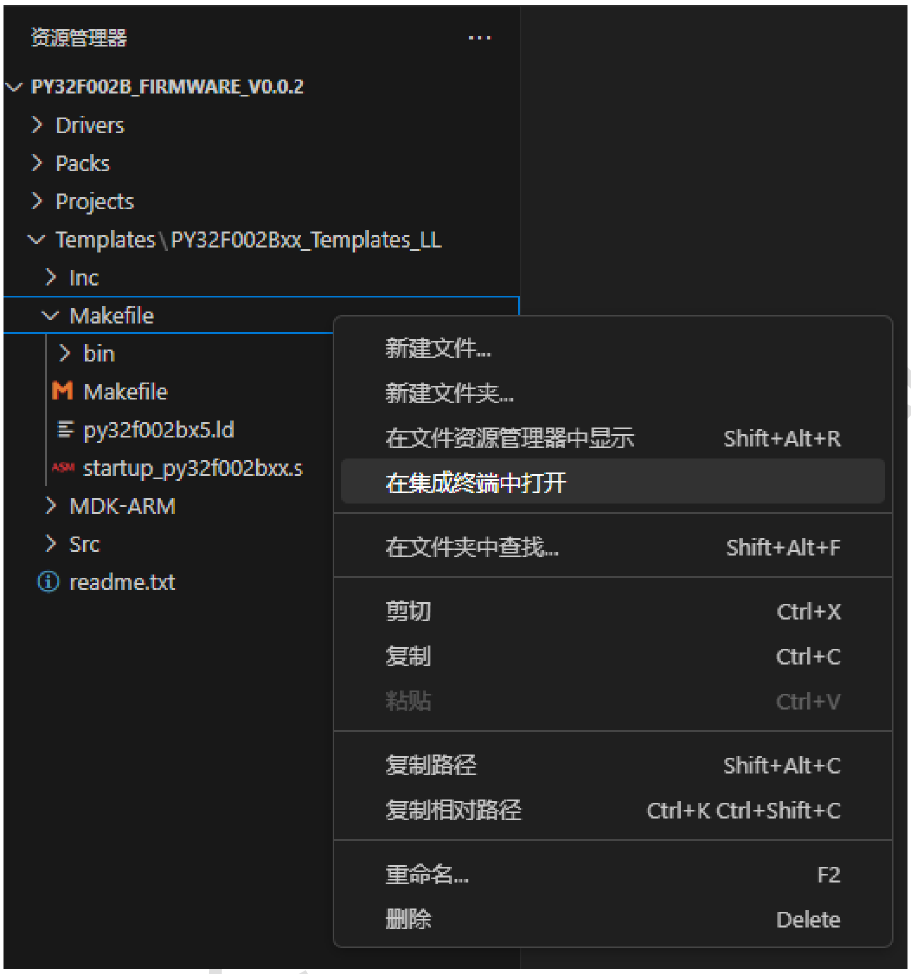
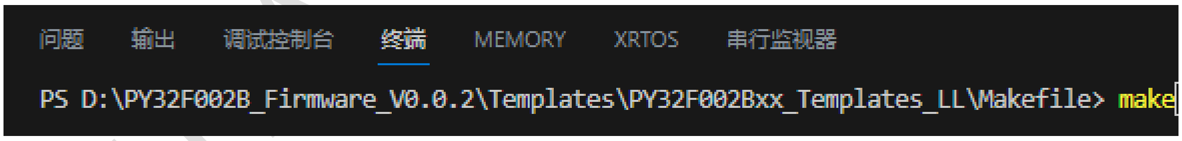
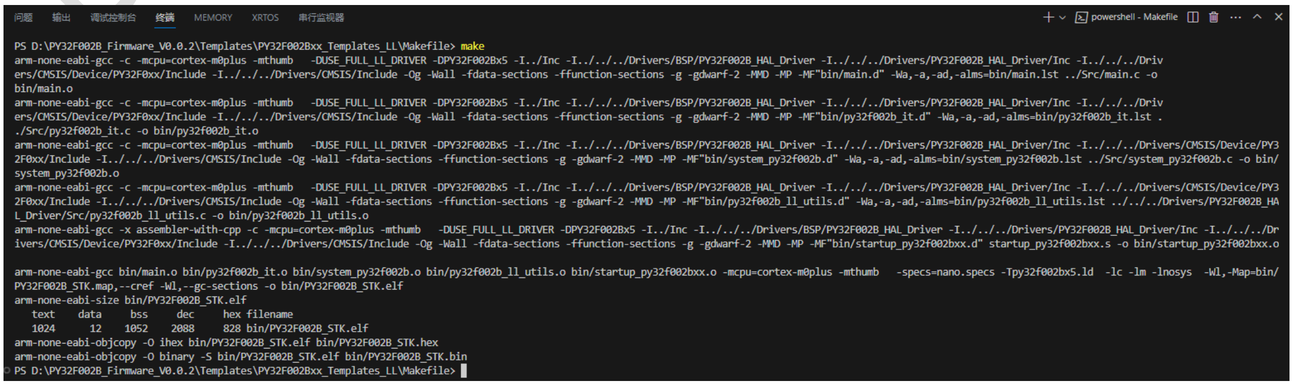


Figure 7.2-2. Type make in the open Powershell terminal and press "Enter".



```
PS D:\PY32F002B_Firmware_V0.0.2\Templates\PY32F002Bxx_Templates_LL\Makefile> make
```

Figure 7.2-3. Compile the target file in hex, bin and elf formats.



```
PS D:\PY32F002B_Firmware_V0.0.2\Templates\PY32F002Bxx_Templates_LL\Makefile> make
arm-none-eabi-gcc -c -mcpu=cortex-m0plus -mthumb   -DUSE_FULL_LL_DRIVER -DPY32F002Bx5 -I../Inc -I../../../Drivers/BSP/PY32F002B_HAL_Driver -I../../../Drivers/PY32F002B_HAL_Driver/Inc -I../../../Driv
ers/CMSIS/Device/PY32F0xx/Include -I../../../Drivers/CMSIS/Include -Og -Wall -fdata-sections -ffunction-sections -g -gdwarf-2 -MMD -MP -MF"bin/main.d" -Wa,-a,-ad,-alms=bin/main.lst ../Src/main.c -o
bin/main.o
arm-none-eabi-gcc -c -mcpu=cortex-m0plus -mthumb   -DUSE_FULL_LL_DRIVER -DPY32F002Bx5 -I../Inc -I../../../Drivers/BSP/PY32F002B_HAL_Driver -I../../../Drivers/PY32F002B_HAL_Driver/Inc -I../../../Driv
ers/CMSIS/Device/PY32F0xx/Include -I../../../Drivers/CMSIS/Include -Og -Wall -fdata-sections -ffunction-sections -g -gdwarf-2 -MMD -MP -MF"bin/py32f002b_it.d" -Wa,-a,-ad,-alms=bin/py32f002b_it.lst .
./Src/py32f002b_it.c -o bin/py32f002b_it.o
arm-none-eabi-gcc -c -mcpu=cortex-m0plus -mthumb   -DUSE_FULL_LL_DRIVER -DPY32F002Bx5 -I../Inc -I../../../Drivers/BSP/PY32F002B_HAL_Driver -I../../../Drivers/PY32F002B_HAL_Driver/Inc -I../../../Drivers/CMSIS/Device/PY3
2F0xx/Include -I../../../Drivers/CMSIS/Include -Og -Wall -fdata-sections -ffunction-sections -g -gdwarf-2 -MMD -MP -MF"bin/system_py32f002b.d" -Wa,-a,-ad,-alms=bin/system_py32f002b.lst ../Src/system_py32f002b.c -o bin/
system_py32f002b.o
arm-none-eabi-gcc -c -mcpu=cortex-m0plus -mthumb   -DUSE_FULL_LL_DRIVER -DPY32F002Bx5 -I../Inc -I../../../Drivers/BSP/PY32F002B_HAL_Driver -I../../../Drivers/PY32F002B_HAL_Driver/Inc -I../../../Drivers/CMSIS/Device/PY3
2F0xx/Include -I../../../Drivers/CMSIS/Include -Og -Wall -fdata-sections -ffunction-sections -g -gdwarf-2 -MMD -MP -MF"bin/py32f002b_ll_utils.d" -Wa,-a,-ad,-alms=bin/py32f002b_ll_utils.lst ../../../Drivers/PY32F002B_HA
L_Driver/Src/py32f002b_ll_utils.c -o bin/py32f002b_ll_utils.o
arm-none-eabi-gcc -x assembler-with-cpp -c -mcpu=cortex-m0plus -mthumb   -DUSE_FULL_LL_DRIVER -DPY32F002Bx5 -I../Inc -I../../../Drivers/BSP/PY32F002B_HAL_Driver -I../../../Drivers/PY32F002B_HAL_Driver/Inc -I../../../Dr
ivers/CMSIS/Device/PY32F0xx/Include -I../../../Drivers/CMSIS/Include -Og -Wall -fdata-sections -ffunction-sections -g -gdwarf-2 -MMD -MP -MF"bin/startup_py32f002bxx.d" startup_py32f002bxx.s -o bin/startup_py32f002bxx.o

arm-none-eabi-gcc bin/main.o bin/py32f002b_it.o bin/system_py32f002b.o bin/py32f002b_ll_utils.o bin/startup_py32f002bxx.o -mcpu=cortex-m0plus -mthumb   -specs=nano.specs -Tpy32f002bx5.ld  -lc -lm -lnosys  -Wl,-Map=bin/
PY32F002B_STK.map,--cref -Wl,--gc-sections -o bin/PY32F002B_STK.elf
arm-none-eabi-size bin/PY32F002B_STK.elf
   text    data     bss     dec     hex filename
   1024      12    1052    2088     828 bin/PY32F002B_STK.elf
arm-none-eabi-objcopy -O ihex bin/PY32F002B_STK.elf bin/PY32F002B_STK.hex
arm-none-eabi-objcopy -O binary -S bin/PY32F002B_STK.elf bin/PY32F002B_STK.bin
PS D:\PY32F002B_Firmware_V0.0.2\Templates\PY32F002Bxx_Templates_LL\Makefile>
```

7.3 CooFlash Software Download

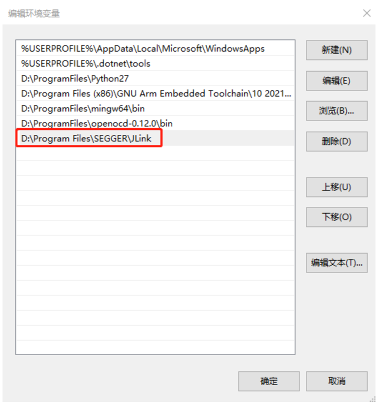
Please refer to the document PY32 Microcontroller CooFlash Download Software User Manual.

7.4 JFlash Software Download

Please refer to the document PY32 Microcontroller JFlash Download Software User Manual.

7.5 JFlash Command Line Download

Figure 7.5-1. Adding the JFlash installation directory to the user environment variable.

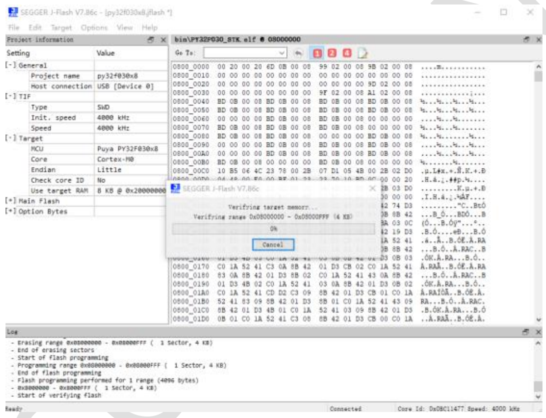


```
%USERPROFILE%\AppData\Local\Microsoft\WindowsApps
%USERPROFILE%\.dotnet\tools
D:\ProgramFiles\Python27
D:\Program Files (x86)\GNU Arm Embedded Toolchain\10 2021...
D:\ProgramFiles\mingw64\bin
D:\ProgramFiles\openocd-0.12.0\bin
D:\Program Files\SEGGER\JLink
```

- JFlash download command

JFlash.exe -openprjpy32f030x8.jflash -openbin\PY32F030_STK.elf,0x08000000 -auto -startapp -exit

Figure 7.5-2. JFlash download in progress



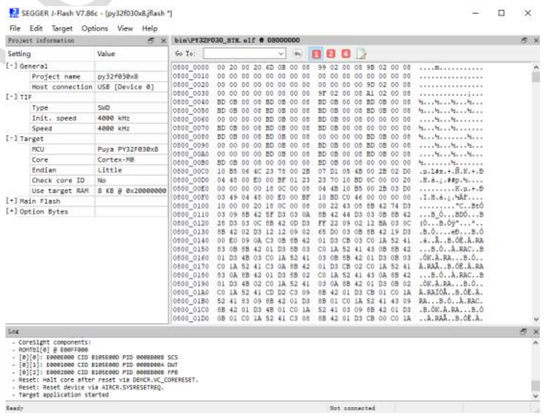
Setting	Value
[-] General	
Project name	py32f030x8
Host connection	USB [Device 0]
[-] TIF	
Type	SWD
Init. speed	4000 kHz
Speed	4000 kHz
[-] Target	
MCU	Puya PY32F030x8
Core	Cortex-M0
Endian	Little
Check core ID	No
Use target RAM	8 KB @ 0x20000000
[+] Main Flash	
[+] Option Bytes	

```
Verifying target memory...
Verifying range 0x08000000 - 0x08000FFF (4 KB)
0%
```

```
0800_0000  00 20 00 20 6D 0B 00 08  99 02 00 08 9B 02 00 08  ....m...........
0800_0010  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
0800_0020  00 00 00 00 00 00 00 00  00 00 00 00 9D 02 00 08  ................
0800_0030  00 00 00 00 00 00 00 00  9F 02 00 08 A1 02 00 08  ............¡...
0800_0040  BD 0B 00 08 BD 0B 00 08  BD 0B 00 08 BD 0B 00 08  ½...½...½...½...
0800_0050  BD 0B 00 08 BD 0B 00 08  BD 0B 00 08 BD 0B 00 08  ½...½...½...½...
0800_0060  00 00 00 00 BD 0B 00 08  BD 0B 00 08 00 00 00 00  ....½...½.......
0800_0070  BD 0B 00 08 BD 0B 00 08  BD 0B 00 08 00 00 00 00  ½...½...½.......
0800_0080  BD 0B 00 08 BD 0B 00 08  00 00 00 00 BD 0B 00 08  ½...½.......½...
0800_0090  00 00 00 00 BD 0B 00 08  BD 0B 00 08 BD 0B 00 08  ....½...½...½...
0800_00A0  00 00 00 00 BD 0B 00 08  BD 0B 00 08 BD 0B 00 08  ....½...½...½...
0800_00B0  BD 0B 00 08 00 00 00 00  BD 0B 00 08 00 00 00 00  ½.......½.......
0800_00C0  10 B5 06 4C 23 78 00 2B  07 D1 05 4B 00 2B 02 D0  .µ.L#x.+.Ñ.K.+.Ð
0800_0170  C0 1A 52 41 C3 0A 8B 42  01 D3 CB 02 C0 1A 52 41  À.RAÃ..B.ÓË.À.RA
0800_0180  83 0A 8B 42 01 D3 8B 02  C0 1A 52 41 43 0A 8B 42  ...B.Ó..À.RAC..B
0800_0190  01 D3 4B 02 C0 1A 52 41  03 0A 8B 42 01 D3 0B 02  .ÓK.À.RA...B.Ó..
0800_01A0  C0 1A 52 41 CD D2 C3 09  8B 42 01 D3 CB 01 C0 1A  À.RAÍÒÃ..B.ÓË.À.
0800_01B0  52 41 83 09 8B 42 01 D3  8B 01 C0 1A 52 41 43 09  RA...B.Ó..À.RAC.
0800_01C0  8B 42 01 D3 4B 01 C0 1A  52 41 03 09 8B 42 01 D3  .B.ÓK.À.RA...B.Ó
0800_01D0  0B 01 C0 1A 52 41 C3 08  8B 42 01 D3 CB 00 C0 1A  ..À.RAÃ..B.ÓË.À.
```

```
Log
 - Erasing range 0x08000000 - 0x08000FFF (  1 Sector, 4 KB)
 - End of erasing sectors
 - Start of flash programming
 - Programming range 0x08000000 - 0x08000FFF (  1 Sector, 4 KB)
 - End of flash programming
 - Flash programming performed for 1 range (4096 bytes)
 - 0x8000000 - 0x8000FFF (  1 Sector, 4 KB)
 - Start of verifying flash
Ready                                   Connected      Core Id: 0x0BC11477 Speed: 4000 kHz
```

Figure 7.5-3. JFlash download complete.


SEGGER J-Flash V7.86c - [py32f030x8.jflash *]
File
Edit
Target
Options
View
Help
Project information
bin\PY32F030_STK.elf @ 08000000
Go To:

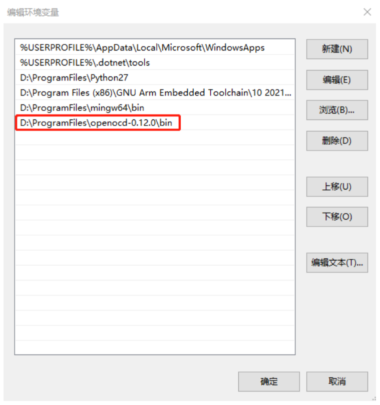
Setting	Value
[-] General	
Project name	py32f030x8
Host connection	USB [Device 0]
[-] TIF	
Type	SWD
Init. speed	4000 kHz
Speed	4000 kHz
[-] Target	
MCU	Puya PY32F030x8
Core	Cortex-M0
Endian	Little
Check core ID	No
Use target RAM	8 KB @ 0x20000000
[+] Main Flash	
[+] Option Bytes	

```
0800_0000  00 20 00 20 6D 0B 00 08  99 02 00 08 9B 02 00 08  ....m...........
0800_0010  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
0800_0020  00 00 00 00 00 00 00 00  00 00 00 00 9D 02 00 08  ................
0800_0030  00 00 00 00 00 00 00 00  9F 02 00 08 A1 02 00 08  ............¡...
0800_0040  BD 0B 00 08 BD 0B 00 08  BD 0B 00 08 BD 0B 00 08  ½...½...½...½...
0800_0050  BD 0B 00 08 BD 0B 00 08  BD 0B 00 08 BD 0B 00 08  ½...½...½...½...
0800_0060  00 00 00 00 BD 0B 00 08  BD 0B 00 08 00 00 00 00  ....½...½.......
0800_0070  BD 0B 00 08 BD 0B 00 08  BD 0B 00 08 00 00 00 00  ½...½...½.......
0800_0080  BD 0B 00 08 BD 0B 00 08  00 00 00 00 BD 0B 00 08  ½...½.......½...
0800_0090  00 00 00 00 BD 0B 00 08  BD 0B 00 08 BD 0B 00 08  ....½...½...½...
0800_00A0  00 00 00 00 BD 0B 00 08  BD 0B 00 08 BD 0B 00 08  ....½...½...½...
0800_00B0  BD 0B 00 08 00 00 00 00  BD 0B 00 08 00 00 00 00  ½.......½.......
0800_00C0  10 B5 06 4C 23 78 00 2B  07 D1 05 4B 00 2B 02 D0  .µ.L#x.+.Ñ.K.+.Ð
0800_00D0  04 48 00 E0 00 BF 01 23  23 70 10 BD 0C 00 00 20  .H.à.¿.##p.½....
0800_00E0  00 00 00 00 18 0C 00 08  04 4B 10 B5 00 2B 03 D0  .........K.µ.+.Ð
0800_00F0  03 49 04 48 00 E0 00 BF  10 BD C0 46 00 00 00 00  .I.H.à.¿.½ÀF....
0800_0100  10 00 00 20 18 0C 00 08  00 22 43 08 8B 42 74 D3  ........."C..BtÓ
0800_0110  03 09 8B 42 5F D3 03 0A  8B 42 44 D3 03 0B 8B 42  ...B_Ó...BDÓ...B
0800_0120  28 D3 03 0C 8B 42 0D D3  FF 22 09 02 12 BA 03 0C  (Ó...B.Óÿ"...º..
0800_0130  8B 42 02 D3 12 12 09 02  65 D0 03 0B 8B 42 19 D3  .B.Ó....eÐ...B.Ó
0800_0140  00 E0 09 0A C3 0B 8B 42  01 D3 CB 03 C0 1A 52 41  .à..Ã..B.ÓË.À.RA
0800_0150  83 0B 8B 42 01 D3 8B 03  C0 1A 52 41 43 0B 8B 42  ...B.Ó..À.RAC..B
0800_0160  01 D3 4B 03 C0 1A 52 41  03 0B 8B 42 01 D3 0B 03  .ÓK.À.RA...B.Ó..
0800_0170  C0 1A 52 41 C3 0A 8B 42  01 D3 CB 02 C0 1A 52 41  À.RAÃ..B.ÓË.À.RA
0800_0180  83 0A 8B 42 01 D3 8B 02  C0 1A 52 41 43 0A 8B 42  ...B.Ó..À.RAC..B
0800_0190  01 D3 4B 02 C0 1A 52 41  03 0A 8B 42 01 D3 0B 02  .ÓK.À.RA...B.Ó..
0800_01A0  C0 1A 52 41 CD D2 C3 09  8B 42 01 D3 CB 01 C0 1A  À.RAÍÒÃ..B.ÓË.À.
0800_01B0  52 41 83 09 8B 42 01 D3  8B 01 C0 1A 52 41 43 09  RA...B.Ó..À.RAC.
0800_01C0  8B 42 01 D3 4B 01 C0 1A  52 41 03 09 8B 42 01 D3  .B.ÓK.À.RA...B.Ó
0800_01D0  0B 01 C0 1A 52 41 C3 08  8B 42 01 D3 CB 00 C0 1A  ..À.RAÃ..B.ÓË.À.
```

```
Log
 - CoreSight components:
 - ROMTbl[0] @ E00FF000
 - [0][0]: E000E000 CID B105E00D PID 000BB008 SCS
 - [0][1]: E0001000 CID B105E00D PID 000BB00A DWT
 - [0][2]: E0002000 CID B105E00D PID 000BB00B FPB
 - Reset: Halt core after reset via DEMCR.VC_CORERESET.
 - Reset: Reset device via AIRCR.SYSRESETREQ.
 - Target application started
Ready                                   Not connected
```

7.6 OpenOCD Command Line Download

Figure 7.6-1. Adding the openocd bin installation directory to the user environment variable.

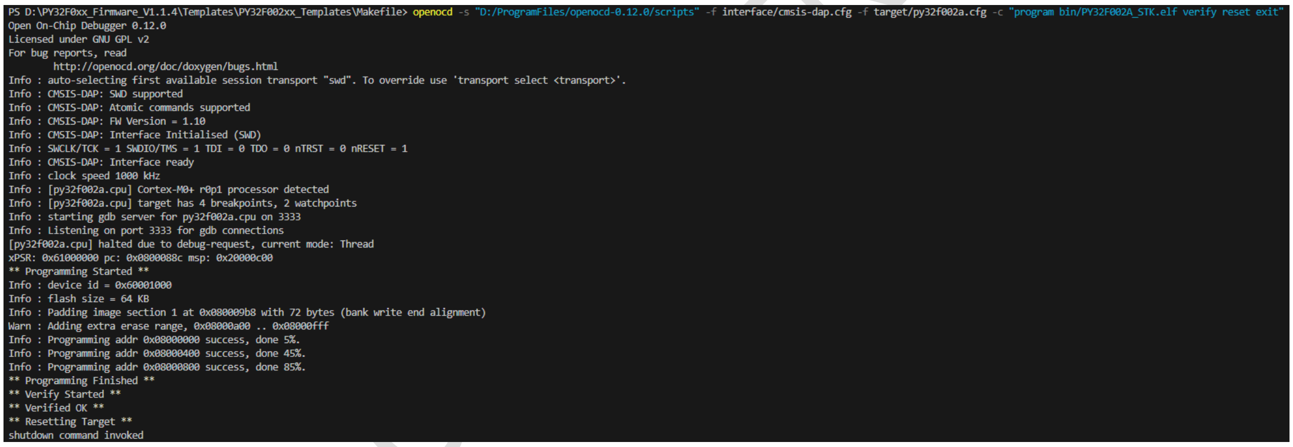


```
%USERPROFILE%\AppData\Local\Microsoft\WindowsApps
%USERPROFILE%\.dotnet\tools
D:\ProgramFiles\Python27
D:\Program Files (x86)\GNU Arm Embedded Toolchain\10 2021...
D:\ProgramFiles\mingw64\bin
D:\ProgramFiles\openocd-0.12.0\bin
```

- OpenOCD download command

openocd -s "D:/ProgramFiles/openocd-0.12.0/scripts" -f interface/cmsis-dap.cfg -f target/py32f002a.cfg -c "program bin/PY32F002A_STK.elf verify reset exit"

Figure 7.6-2. VSCode powershell terminal entering the above command to start the download



```
PS D:\PY32F0xx_Firmware_V1.1.4\Templates\PY32F002xx_Templates\Makefile> openocd -s "D:/ProgramFiles/openocd-0.12.0/scripts" -f interface/cmsis-dap.cfg -f target/py32f002a.cfg -c "program bin/PY32F002A_STK.elf verify reset exit"
Open On-Chip Debugger 0.12.0
Licensed under GNU GPL v2
For bug reports, read
        http://openocd.org/doc/doxygen/bugs.html
Info : auto-selecting first available session transport "swd". To override use 'transport select <transport>'.
Info : CMSIS-DAP: SWD supported
Info : CMSIS-DAP: Atomic commands supported
Info : CMSIS-DAP: FW Version = 1.10
Info : CMSIS-DAP: Interface Initialised (SWD)
Info : SWCLK/TCK = 1 SWDIO/TMS = 1 TDI = 0 TDO = 0 nTRST = 0 nRESET = 1
Info : CMSIS-DAP: Interface ready
Info : clock speed 1000 kHz
Info : [py32f002a.cpu] Cortex-M0+ r0p1 processor detected
Info : [py32f002a.cpu] target has 4 breakpoints, 2 watchpoints
Info : starting gdb server for py32f002a.cpu on 3333
Info : Listening on port 3333 for gdb connections
[py32f002a.cpu] halted due to debug-request, current mode: Thread
xPSR: 0x61000000 pc: 0x0800088c msp: 0x20000c00
** Programming Started **
Info : device id = 0x60001000
Info : flash size = 64 KB
Info : Padding image section 1 at 0x080009b8 with 72 bytes (bank write end alignment)
Warn : Adding extra erase range, 0x08000a00 .. 0x08000fff
Info : Programming addr 0x08000000 success, done 5%.
Info : Programming addr 0x08000400 success, done 45%.
Info : Programming addr 0x08000800 success, done 85%.
** Programming Finished **
** Verify Started **
** Verified OK **
** Resetting Target **
shutdown command invoked
```

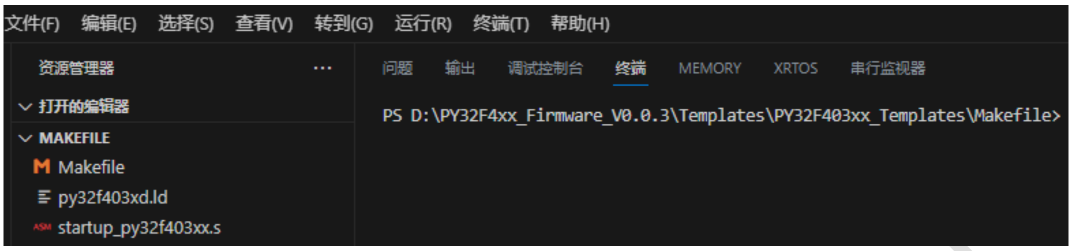
Table 7.6-1. list of openocd cfg files

chip series	target scripts	adapter	interface scripts
PY32F002A	py32f002a.cfg	DAP-LINK	cmsis-dap.cfg
PY32F002B	py32f002b.cfg	J-Link	jlink.cfg
PY32F003	py32f003.cfg	ST-LINK V2	stlink-v2.cfg
PY32F030	py32f030.cfg	ULINK	ulink.cfg
PY32F07X	py32f07x.cfg		
PY32F403	py32f403.cfg		
PY32L020	py32l020.cfg		

8 Creating and using VSCode tasks

8.1 Create compile and download tasks - tasks.json

Figure 8.1-1. Opening the Makefile folder using VSCode


文件(F) 编辑(E) 选择(S) 查看(V) 转到(G) 运行(R) 终端(T) 帮助(H)
资源管理器
打开的编辑器
MAKEFILE
Makefile
py32f403xd.ld
startup_py32f403xx.s
问题 输出 调试控制台 终端 MEMORY XRTOS 串行监视器

```
PS D:\PY32F4xx_Firmware_V0.0.3\Templates\PY32F403xx_Templates\Makefile>
```

Figure 8.1-2. The menu bar selects "Terminal", "Configure Tasks", and "Create tasks.json file using template".

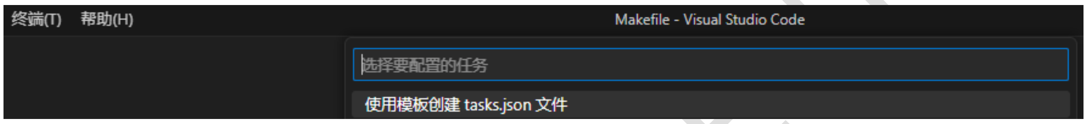


Figure 8.1-3. Select "Others to run an example of any external command".

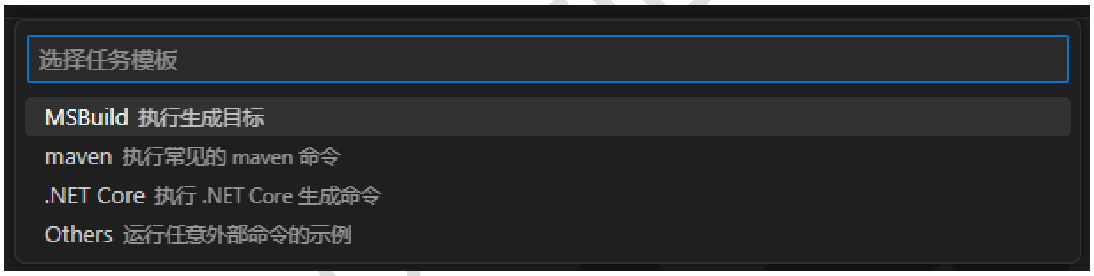
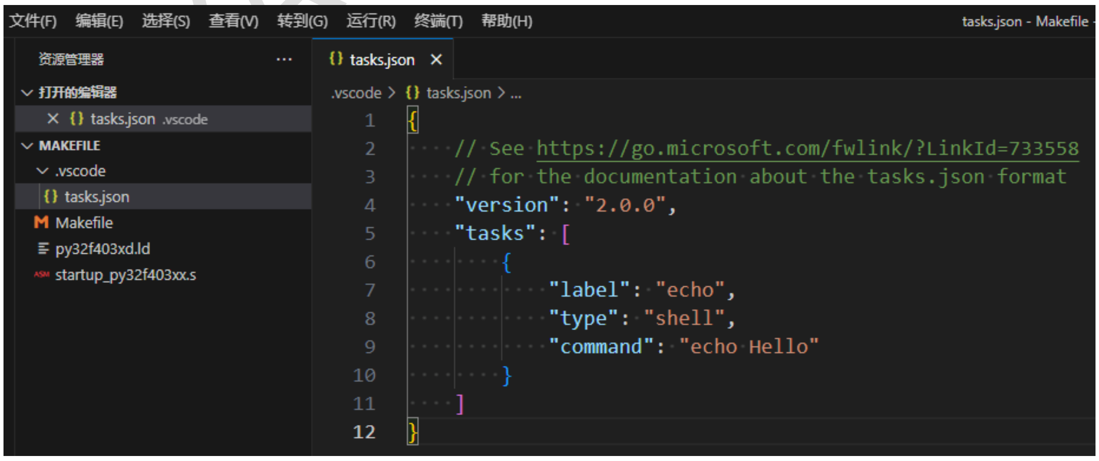
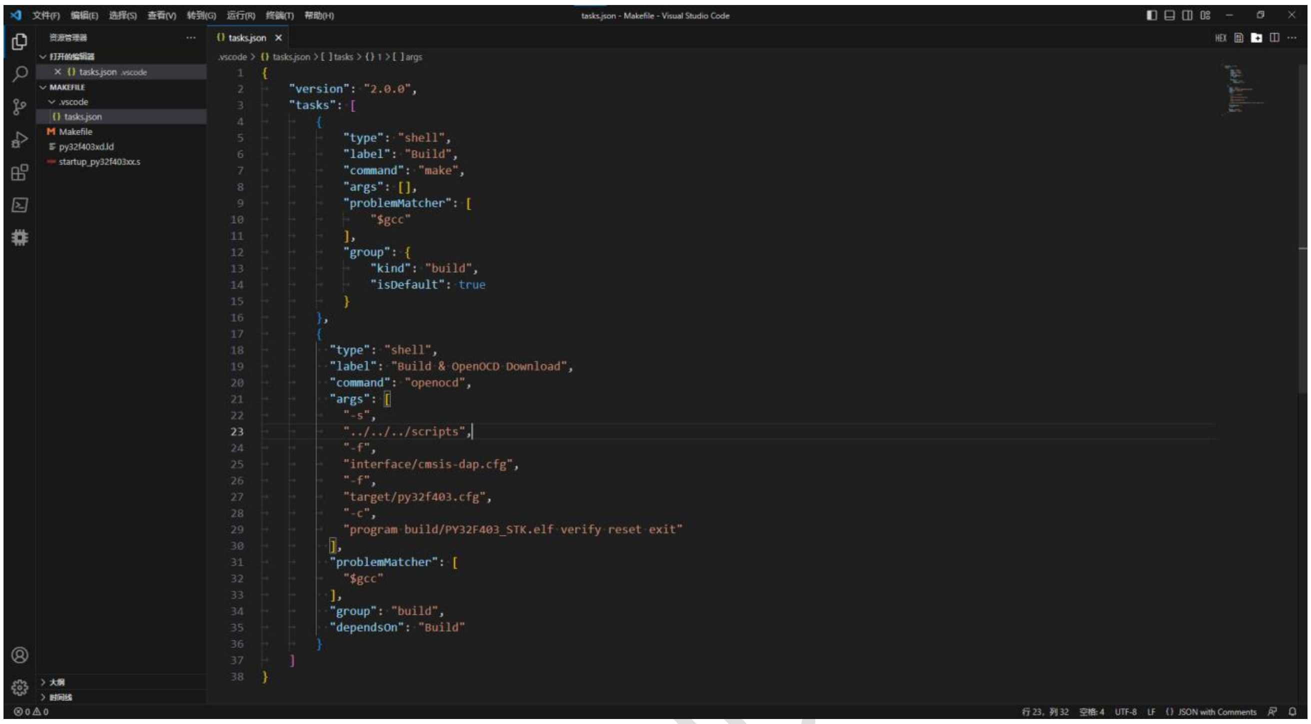

选择任务模板
MSBuild 执行生成目标
maven 执行常见的 maven 命令
.NET Core 执行 .NET Core 生成命令
Others 运行任意外部命令的示例

Figure 8.1-4. VSCode automatically generates tasks.json file



```
{
    // See https://go.microsoft.com/fwlink/?LinkId=733558
    // for the documentation about the tasks.json format
    "version": "2.0.0",
    "tasks": [
        {
            "label": "echo",
            "type": "shell",
            "command": "echo Hello"
        }
    ]
}
```

Figure 8.1-5. Modifying the tasks.json file and saving it



```
{
    "version": "2.0.0",
    "tasks": [
        {
            "type": "shell",
            "label": "Build",
            "command": "make",
            "args": [],
            "problemMatcher": [
                "$gcc"
            ],
            "group": {
                "kind": "build",
                "isDefault": true
            }
        },
        {
          "type": "shell",
          "label": "Build & OpenOCD Download",
          "command": "openocd",
          "args": [
            "-s",
            "../../../scripts",
            "-f",
            "interface/cmsis-dap.cfg",
            "-f",
            "target/py32f403.cfg",
            "-c",
            "program build/PY32F403_STK.elf verify reset exit"
          ],
          "problemMatcher": [
            "$gcc"
          ],
          "group": "build",
          "dependsOn": "Build"
        }
    ]
}
```

Figure 8.1-6. Select "Terminal" and "Run Tasks" from the menu bar.

Two new tasks appear: Build and Build & OpenOCD Download.

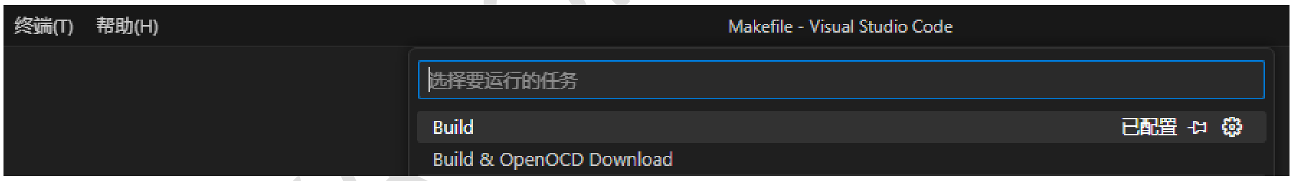
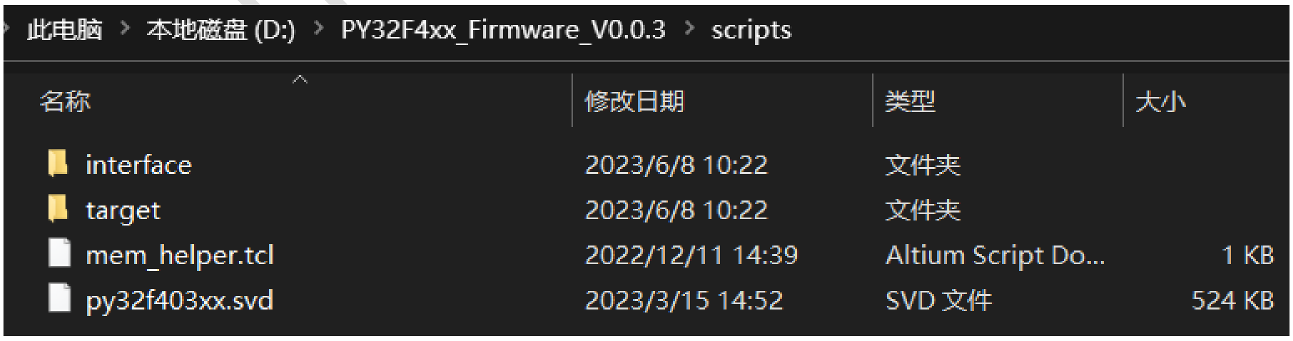


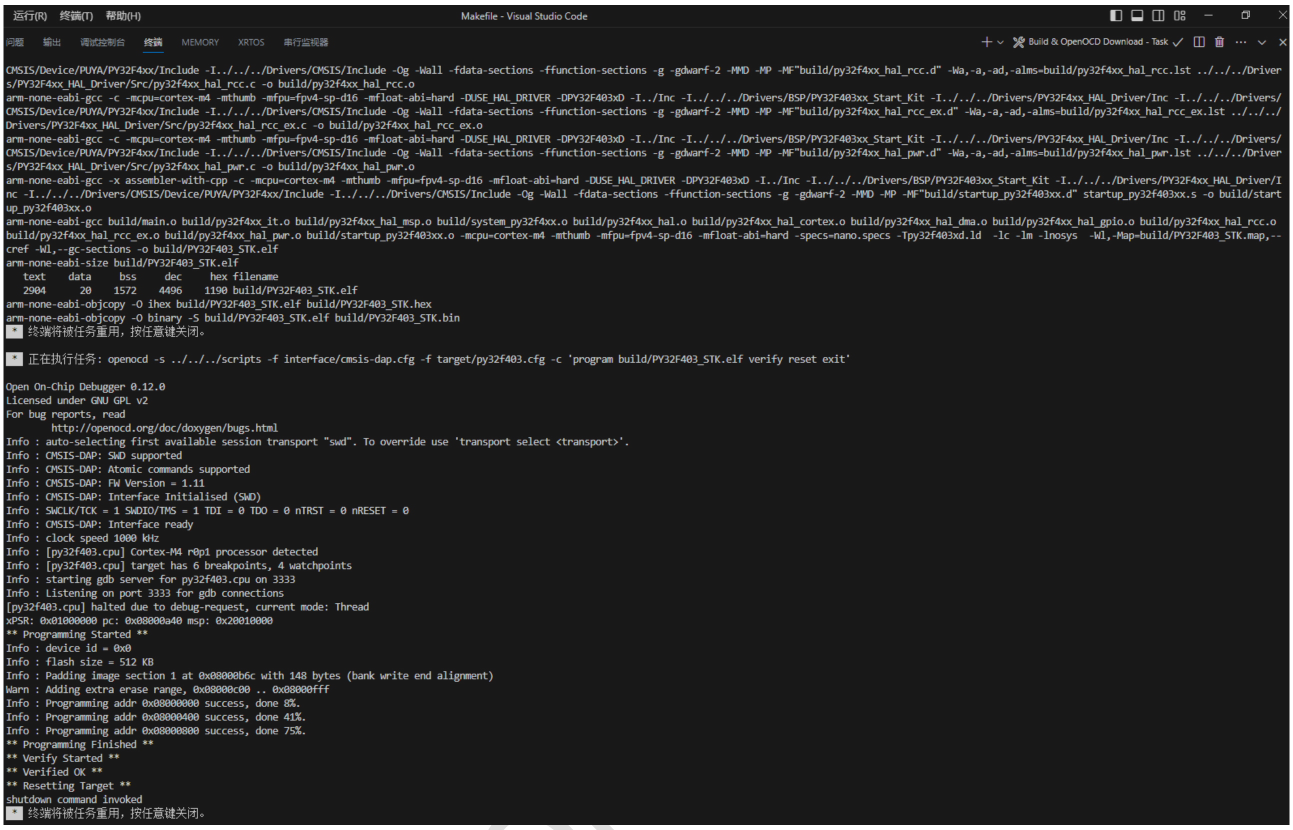
Figure 8.1-7. Copying the scripts script file



此电脑 > 本地磁盘 (D:) > PY32F4xx_Firmware_V0.0.3 > scripts

名称	修改日期	类型	大小
interface	2023/6/8 10:22	文件夹	
target	2023/6/8 10:22	文件夹	
mem_helper.tcl	2022/12/11 14:39	Altium Script Do...	1 KB
py32f403xx.svd	2023/3/15 14:52	SVD 文件	524 KB

Figure 8.1-8. Run the Build & OpenOCD Download task to perform the compilation and download in turn


运行(R) 终端(T) 帮助(H)
Makefile - Visual Studio Code
问题 输出 调试控制台 终端 MEMORY XRTOS 串行监视器
Build & OpenOCD Download - Task

```
CMSIS/Device/PUYA/PY32F4xx/Include -I../../../Drivers/CMSIS/Include -Og -Wall -fdata-sections -ffunction-sections -g -gdwarf-2 -MMD -MP -MF"build/py32f4xx_hal_rcc.d" -Wa,-a,-ad,-alms=build/py32f4xx_hal_rcc.lst ../../../Driver
s/PY32F4xx_HAL_Driver/Src/py32f4xx_hal_rcc.c -o build/py32f4xx_hal_rcc.o
arm-none-eabi-gcc -c -mcpu=cortex-m4 -mthumb -mfpu=fpv4-sp-d16 -mfloat-abi=hard -DUSE_HAL_DRIVER -DPY32F403xD -I../Inc -I../../../Drivers/BSP/PY32F403xx_Start_Kit -I../../../Drivers/PY32F4xx_HAL_Driver/Inc -I../../../Drivers/
CMSIS/Device/PUYA/PY32F4xx/Include -I../../../Drivers/CMSIS/Include -Og -Wall -fdata-sections -ffunction-sections -g -gdwarf-2 -MMD -MP -MF"build/py32f4xx_hal_rcc_ex.d" -Wa,-a,-ad,-alms=build/py32f4xx_hal_rcc_ex.lst ../../../
Drivers/PY32F4xx_HAL_Driver/Src/py32f4xx_hal_rcc_ex.c -o build/py32f4xx_hal_rcc_ex.o
arm-none-eabi-gcc -c -mcpu=cortex-m4 -mthumb -mfpu=fpv4-sp-d16 -mfloat-abi=hard -DUSE_HAL_DRIVER -DPY32F403xD -I../Inc -I../../../Drivers/BSP/PY32F403xx_Start_Kit -I../../../Drivers/PY32F4xx_HAL_Driver/Inc -I../../../Drivers/
CMSIS/Device/PUYA/PY32F4xx/Include -I../../../Drivers/CMSIS/Include -Og -Wall -fdata-sections -ffunction-sections -g -gdwarf-2 -MMD -MP -MF"build/py32f4xx_hal_pwr.d" -Wa,-a,-ad,-alms=build/py32f4xx_hal_pwr.lst ../../../Driver
s/PY32F4xx_HAL_Driver/Src/py32f4xx_hal_pwr.c -o build/py32f4xx_hal_pwr.o
arm-none-eabi-gcc -x assembler-with-cpp -c -mcpu=cortex-m4 -mthumb -mfpu=fpv4-sp-d16 -mfloat-abi=hard -DUSE_HAL_DRIVER -DPY32F403xD -I../Inc -I../../../Drivers/BSP/PY32F403xx_Start_Kit -I../../../Drivers/PY32F4xx_HAL_Driver/I
nc -I../../../Drivers/CMSIS/Device/PUYA/PY32F4xx/Include -I../../../Drivers/CMSIS/Include -Og -Wall -fdata-sections -ffunction-sections -g -gdwarf-2 -MMD -MP -MF"build/startup_py32f403xx.d" startup_py32f403xx.s -o build/start
up_py32f403xx.o
arm-none-eabi-gcc build/main.o build/py32f4xx_it.o build/py32f4xx_hal_msp.o build/system_py32f4xx.o build/py32f4xx_hal.o build/py32f4xx_hal_cortex.o build/py32f4xx_hal_dma.o build/py32f4xx_hal_gpio.o build/py32f4xx_hal_rcc.o
build/py32f4xx_hal_rcc_ex.o build/py32f4xx_hal_pwr.o build/startup_py32f403xx.o -mcpu=cortex-m4 -mthumb -mfpu=fpv4-sp-d16 -mfloat-abi=hard -specs=nano.specs -Tpy32f403xd.ld  -lc -lm -lnosys  -Wl,-Map=build/PY32F403_STK.map,--
cref -Wl,--gc-sections -o build/PY32F403_STK.elf
arm-none-eabi-size build/PY32F403_STK.elf
   text    data     bss     dec     hex filename
   2904      20    1572    4496    1190 build/PY32F403_STK.elf
arm-none-eabi-objcopy -O ihex build/PY32F403_STK.elf build/PY32F403_STK.hex
arm-none-eabi-objcopy -O binary -S build/PY32F403_STK.elf build/PY32F403_STK.bin
 *  终端将被任务重用，按任意键关闭。

 *  正在执行任务: openocd -s ../../../scripts -f interface/cmsis-dap.cfg -f target/py32f403.cfg -c 'program build/PY32F403_STK.elf verify reset exit'

Open On-Chip Debugger 0.12.0
Licensed under GNU GPL v2
For bug reports, read
        http://openocd.org/doc/doxygen/bugs.html
Info : auto-selecting first available session transport "swd". To override use 'transport select <transport>'.
Info : CMSIS-DAP: SWD supported
Info : CMSIS-DAP: Atomic commands supported
Info : CMSIS-DAP: FW Version = 1.11
Info : CMSIS-DAP: Interface Initialised (SWD)
Info : SWCLK/TCK = 1 SWDIO/TMS = 1 TDI = 0 TDO = 0 nTRST = 0 nRESET = 0
Info : CMSIS-DAP: Interface ready
Info : clock speed 1000 kHz
Info : [py32f403.cpu] Cortex-M4 r0p1 processor detected
Info : [py32f403.cpu] target has 6 breakpoints, 4 watchpoints
Info : starting gdb server for py32f403.cpu on 3333
Info : Listening on port 3333 for gdb connections
[py32f403.cpu] halted due to debug-request, current mode: Thread
xPSR: 0x01000000 pc: 0x08000a40 msp: 0x20010000
** Programming Started **
Info : device id = 0x0
Info : flash size = 512 KB
Info : Padding image section 1 at 0x08000b6c with 148 bytes (bank write end alignment)
Warn : Adding extra erase range, 0x08000c00 .. 0x08000fff
Info : Programming addr 0x08000000 success, done 8%.
Info : Programming addr 0x08000400 success, done 41%.
Info : Programming addr 0x08000800 success, done 75%.
** Programming Finished **
** Verify Started **
** Verified OK **
** Resetting Target **
shutdown command invoked
 *  终端将被任务重用，按任意键关闭。
```

8.2 Creating a debug task - launch.json

Figure 8.2-1. Select "Run and Debug" in the leftmost toolbar.

Then click on the blue font "Create launch.json" and select "Cortex Debug".

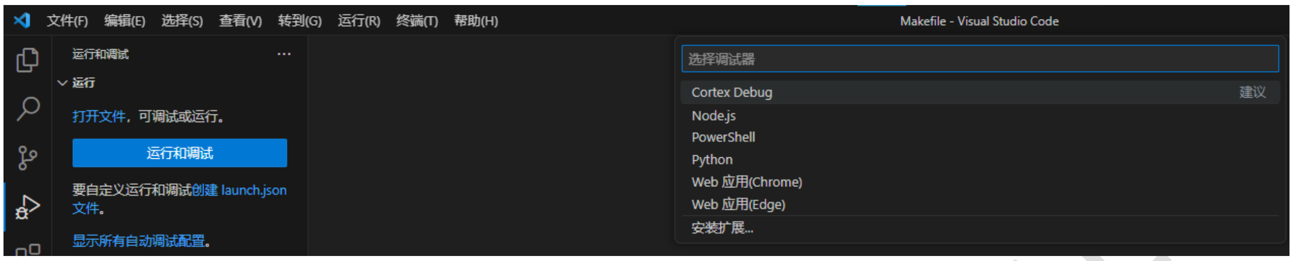
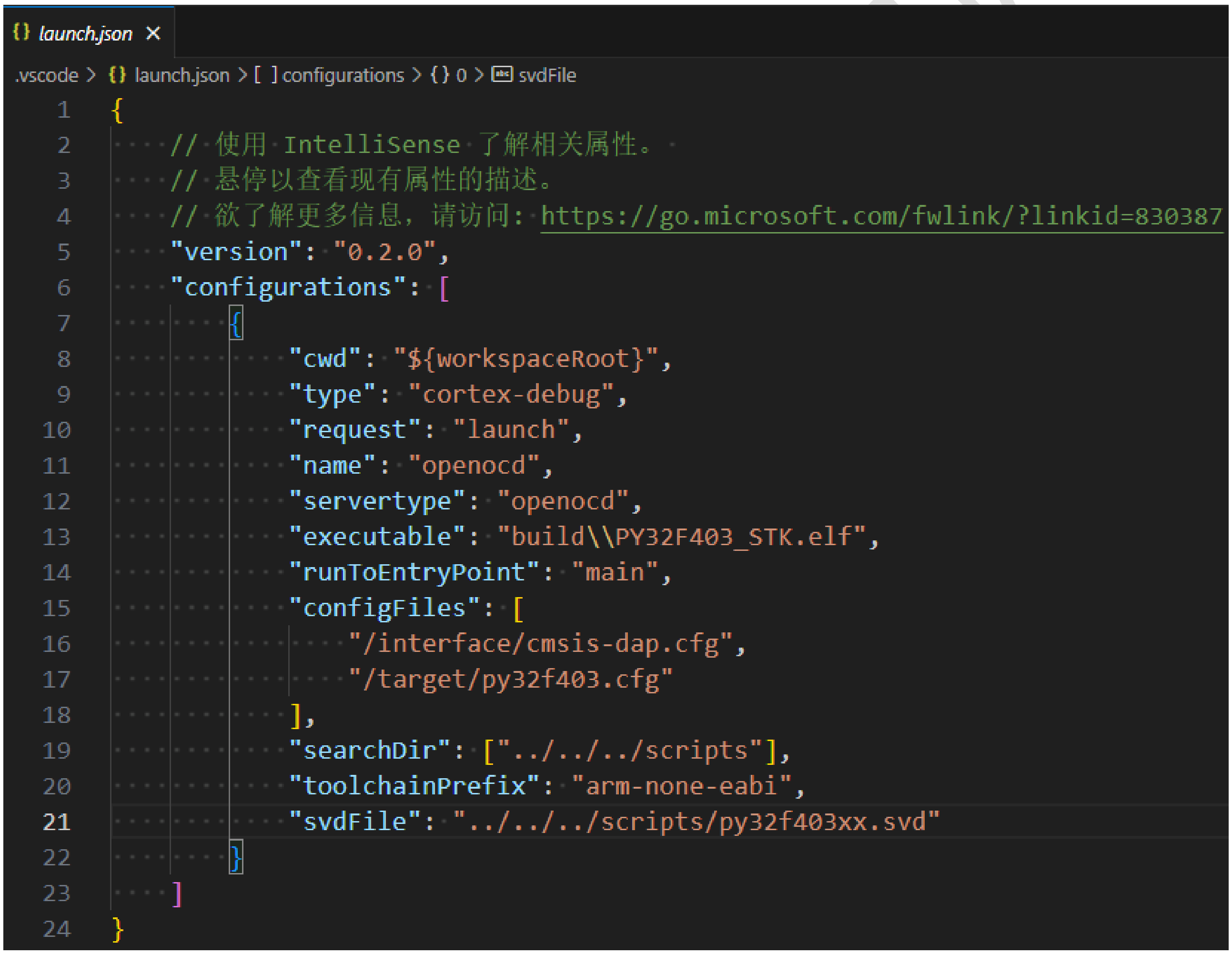
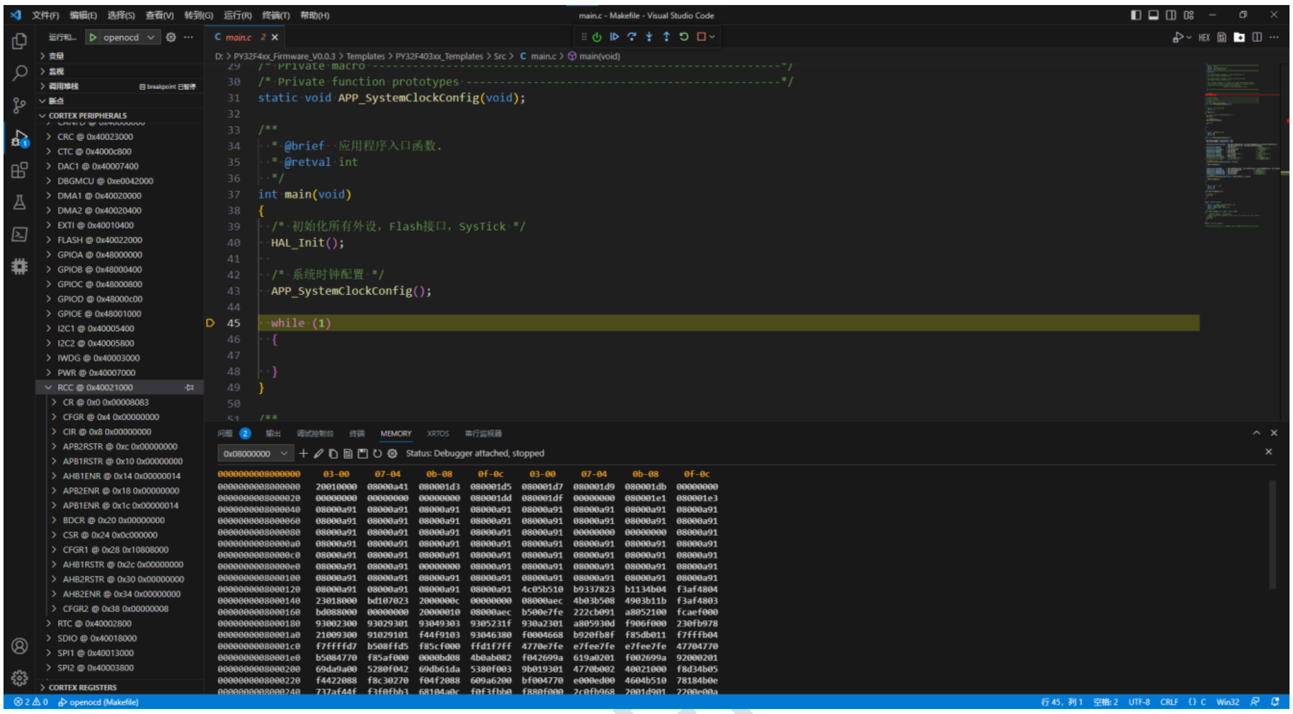


Figure 8.2-2. Modifying the launch.json file


launch.json
.vscode > launch.json > [ ] configurations > { } 0 > svdFile

```json
{
    // 使用 IntelliSense 了解相关属性。
    // 悬停以查看现有属性的描述。
    // 欲了解更多信息，请访问: https://go.microsoft.com/fwlink/?linkid=830387
    "version": "0.2.0",
    "configurations": [
        {
            "cwd": "${workspaceRoot}",
            "type": "cortex-debug",
            "request": "launch",
            "name": "openocd",
            "servertype": "openocd",
            "executable": "build\\PY32F403_STK.elf",
            "runToEntryPoint": "main",
            "configFiles": [
                "/interface/cmsis-dap.cfg",
                "/target/py32f403.cfg"
            ],
            "searchDir": ["../../../scripts"],
            "toolchainPrefix": "arm-none-eabi",
            "svdFile": "../../../scripts/py32f403xx.svd"
        }
    ]
}
```

Figure 8.2-3. Click on the green triangle button to the left of "openocd" in the left task window to start Cortex-Debug debugging.


文件(F) 编辑(E) 选择(S) 查看(V) 转到(G) 运行(R) 终端(T) 帮助(H)
main.c - Makefile - Visual Studio Code
运行和... openocd
main.c 2
变量
监视
调用堆栈 breakpoint 已暂停
断点
问题 2 输出 调试控制台 终端 MEMORY XRTOS 串行监视器
0x08000000
行 45, 列 1 空格: 2 UTF-8 CRLF {} C Win32
⊗ 2 ⚠ 0 openocd (Makefile)

D: > PY32F4xx_Firmware_V0.0.3 > Templates > PY32F403xx_Templates > Src > C main.c > main(void)

```c
/* Private macro -------------------------------------------------------------*/
/* Private function prototypes -----------------------------------------------*/
static void APP_SystemClockConfig(void);

/**
  * @brief  应用程序入口函数.
  * @retval int
  */
int main(void)
{
  /* 初始化所有外设，Flash接口，SysTick */
  HAL_Init();
  
  /* 系统时钟配置 */
  APP_SystemClockConfig();

  while (1)
  {

  }
}

/**
```

CORTEX PERIPHERALS

```
> CRC @ 0x40023000
> CTC @ 0x4000c800
> DAC1 @ 0x40007400
> DBGMCU @ 0xe0042000
> DMA1 @ 0x40020000
> DMA2 @ 0x40020400
> EXTI @ 0x40010400
> FLASH @ 0x40022000
> GPIOA @ 0x48000000
> GPIOB @ 0x48000400
> GPIOC @ 0x48000800
> GPIOD @ 0x48000c00
> GPIOE @ 0x48001000
> I2C1 @ 0x40005400
> I2C2 @ 0x40005800
> IWDG @ 0x40003000
> PWR @ 0x40007000
v RCC @ 0x40021000
   > CR @ 0x0 0x00008083
   > CFGR @ 0x4 0x00000000
   > CIR @ 0x8 0x00000000
   > APB2RSTR @ 0xc 0x00000000
   > APB1RSTR @ 0x10 0x00000000
   > AHB1ENR @ 0x14 0x00000014
   > APB2ENR @ 0x18 0x00000000
   > APB1ENR @ 0x1c 0x00000014
   > BDCR @ 0x20 0x00000000
   > CSR @ 0x24 0x0c000000
   > CFGR1 @ 0x28 0x10808000
   > AHB1RSTR @ 0x2c 0x00000000
   > AHB2RSTR @ 0x30 0x00000000
   > AHB2ENR @ 0x34 0x00000000
   > CFGR2 @ 0x38 0x00000008
> RTC @ 0x40002800
> SDIO @ 0x40018000
> SPI1 @ 0x40013000
> SPI2 @ 0x40003800
```

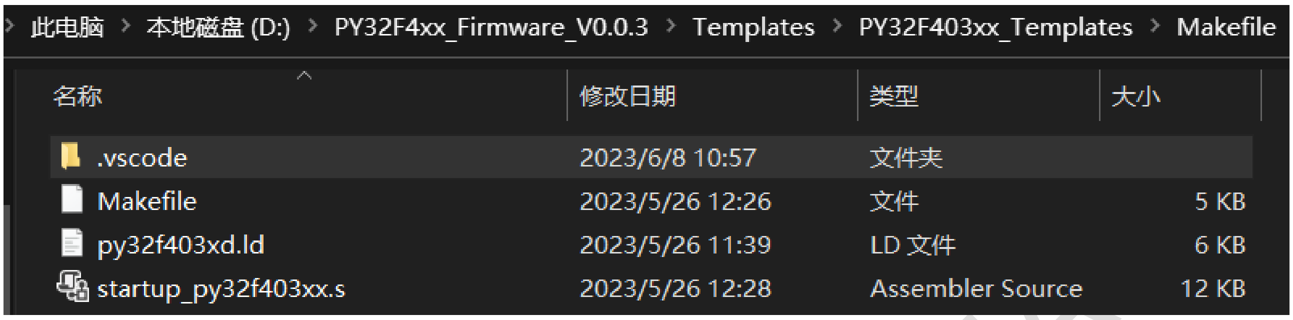
CORTEX REGISTERS

Status: Debugger attached, stopped

```
0000000008000000  03-00     07-04     0b-08     0f-0c     03-00     07-04     0b-08     0f-0c
0000000008000000  20010000  08000a41  080001d3  080001d5  080001d7  080001d9  080001db  00000000
0000000008000020  00000000  00000000  00000000  080001dd  080001df  00000000  080001e1  080001e3
0000000008000040  08000a91  08000a91  08000a91  08000a91  08000a91  08000a91  08000a91  08000a91
0000000008000060  08000a91  08000a91  08000a91  08000a91  08000a91  08000a91  08000a91  08000a91
0000000008000080  08000a91  08000a91  08000a91  08000a91  08000a91  00000000  00000000  08000a91
00000000080000a0  08000a91  08000a91  08000a91  08000a91  08000a91  08000a91  08000a91  08000a91
00000000080000c0  08000a91  08000a91  08000a91  08000a91  08000a91  08000a91  08000a91  08000a91
00000000080000e0  08000a91  08000a91  00000000  08000a91  08000a91  08000a91  08000a91  08000a91
0000000008000100  08000a91  08000a91  08000a91  08000a91  08000a91  08000a91  08000a91  08000a91
0000000008000120  08000a91  08000a91  08000a91  08000a91  4c05b510  b9337823  b1134b04  f3af4804
0000000008000140  23018000  bd107023  2000000c  00000000  08000aec  4b03b508  4903b11b  f3af4803
0000000008000160  bd088000  00000000  20000010  08000aec  b500e7fe  222cb091  a8052100  fcaef000
0000000008000180  93002300  93029301  93049303  9305231f  930a2301  a805930d  f906f000  230fb978
00000000080001a0  21009300  91029101  f44f9103  93046380  f0004668  b920fb8f  f85db011  f7fffb04
00000000080001c0  f7ffffd7  b508ffd5  f85cf000  ffd1f7ff  4770e7fe  e7fee7fe  e7fee7fe  47704770
00000000080001e0  b5084770  f85af000  0000bd08  4b0ab082  f042699a  619a0201  f002699a  92000201
0000000008000200  69da9a00  5280f042  69db61da  5380f003  9b019301  4770b002  40021000  f8d34b05
0000000008000220  f4422088  f8c30270  f04f2088  609a6200  bf004770  e000ed00  4604b510  78184b0e
0000000008000240  737af44f  f3f0fbb3  68104a0c  f0f3fbb0  f880f000  2c0fb968  2001d901  2200e00a
```

8.3 Copy task folder - .vscode

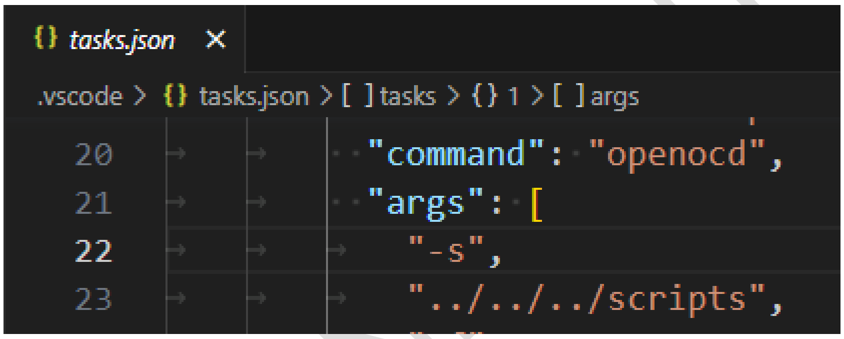
Figure 8.3-1. Copy all the .vscode folders under the Makefile folder to the Makefile folders of other projects.



此电脑 › 本地磁盘 (D:) › PY32F4xx_Firmware_V0.0.3 › Templates › PY32F403xx_Templates › Makefile

名称	修改日期	类型	大小
.vscode	2023/6/8 10:57	文件夹	
Makefile	2023/5/26 12:26	文件	5 KB
py32f403xd.ld	2023/5/26 11:39	LD 文件	6 KB
startup_py32f403xx.s	2023/5/26 12:28	Assembler Source	12 KB

Figure 8.3-2. Modify the path to the scripts folder inside the tasks.json file.

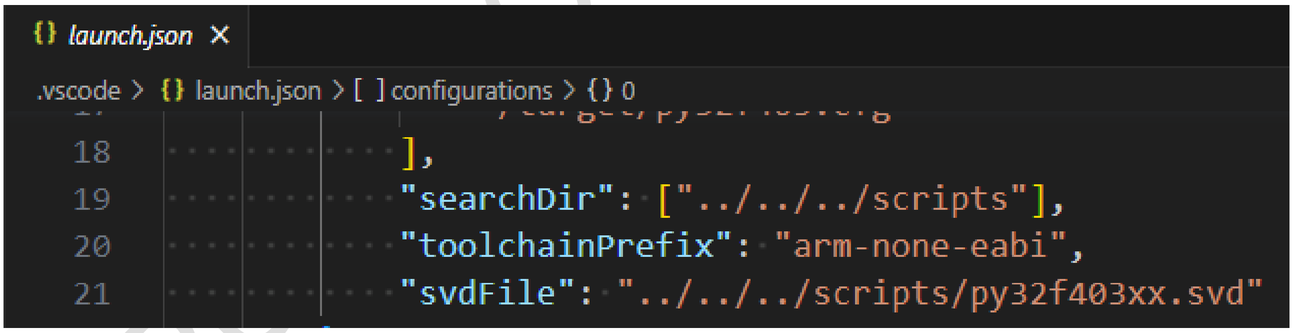


tasks.json

.vscode › tasks.json › [ ] tasks › {} 1 › [ ] args

```
        "command": "openocd",
        "args": [
            "-s",
            "../../../scripts",
```

Figure 8.3-3. Modify the path to the scripts folder and svd file inside the launch.json file.



launch.json

.vscode › launch.json › [ ] configurations › {} 0

```
            ],
            "searchDir": ["../../../scripts"],
            "toolchainPrefix": "arm-none-eabi",
            "svdFile": "../../../scripts/py32f403xx.svd"
```

9 Version History

releases	dates	update a record
V1.0	2023.6.8	first edition

Puya Semiconductor Co.