

前言

本文档用于描述 PY32 微控制器 GCC 编译环境的安装以及使用。通过使用 VSCode 软件实现编辑功能；通过 gcc-arm-none-eabi 软件实现编译功能；通过 CooFlash 软件和 PY-LINK 仿真器、JFlash 软件和 J-Link 仿真器实现下载烧录功能。

PUYA CONFIDENTIAL

目录

1	安装 GCC	3
1.1	下载	3
1.2	安装	3
2	安装 mingw	5
3	安装 VSCode	6
3.1	下载	6
3.2	安装	6
4	添加环境变量	7
5	Makefile 文件	8
6	*.ld 链接文件	9
7	编辑、编译、下载	10
7.1	VSCode 编辑	10
7.2	GCC 编译	11
7.3	CooFlash 软件下载	12
7.4	JFlash 软件下载	12
7.5	JFlash 命令行下载	13
7.6	OpenOCD 命令行下载	14
8	创建和使用 VSCode 任务	15
8.1	创建编译和下载任务 – tasks.json	15
8.2	创建调试任务 – launch.json	18
8.3	拷贝任务文件夹 – .vscode	20
9	版本历史	21

1 安装 GCC

1.1 下载

最新 gcc-arm-none-eabi 编译器下载链接: <https://developer.arm.com/downloads/-/gnu-rm>

1.2 安装

图 1.2-1. 双击 gcc-arm-none-eabi-10.3-2021.10-win32.exe 开始安装

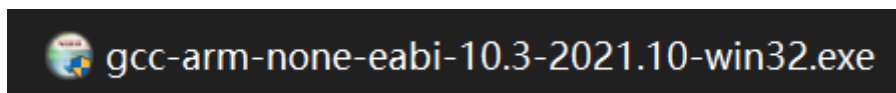


图 1.2-2. 选择 Chinese(Simplified), 点击 OK

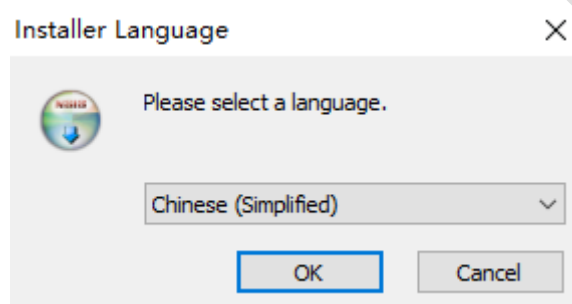


图 1.2-3. 点击“下一步”按钮



图 1.2-4. 点击“我接受”按钮

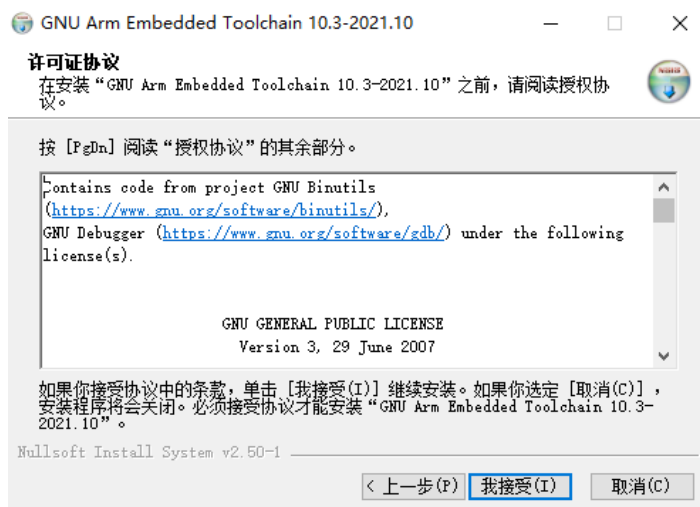


图 1.2-5. 选择好路径后，点击“安装”按钮

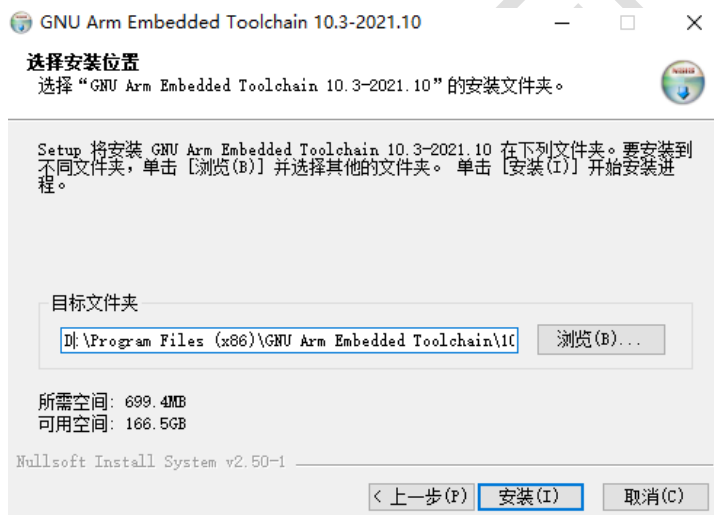


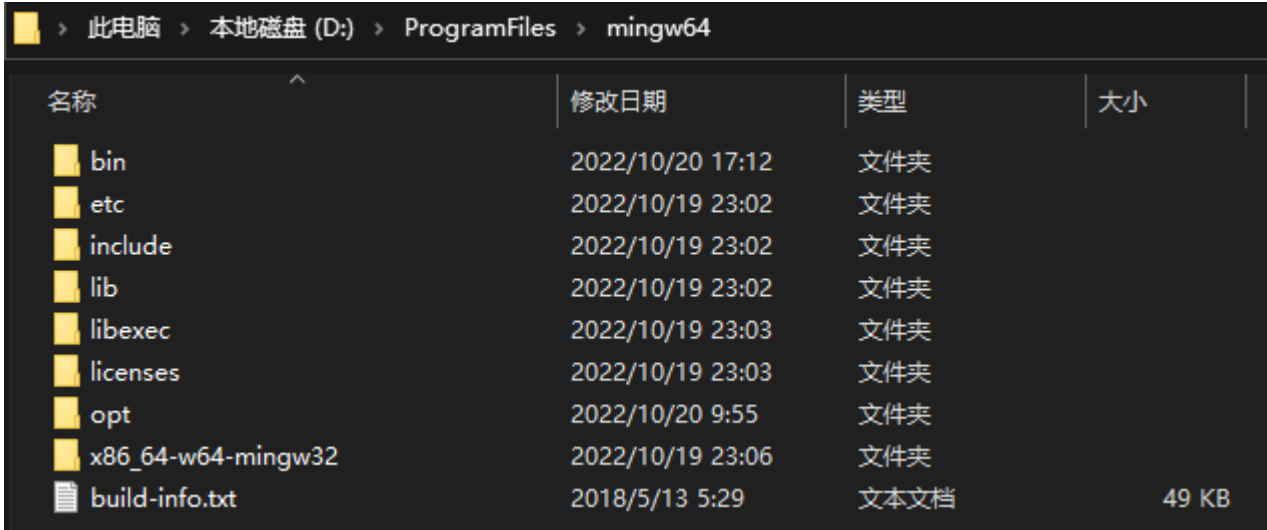
图 1.2-6. 勾选“Add path to environment variable”，点击“完成”按钮



2 安装 mingw

此软件为绿色免安装软件，解压后即可使用，注意解压路径不要有空格、中文等特殊字符，如 D:\ProgramFiles。

图 2-1. 解压 mingw64.rar



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 安装 VSCode

3.1 下载

最新 VSCode 软件下载链接: <https://code.visualstudio.com/Download>

3.2 安装

图 1.2-1. 双击 VSCodeUserSetup-x64-1.73.1.exe, 根据安装向导提示完成安装

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 编辑环境变量

图 4-1. 选中用户变量“ Path” 这一栏， 点击编辑按钮

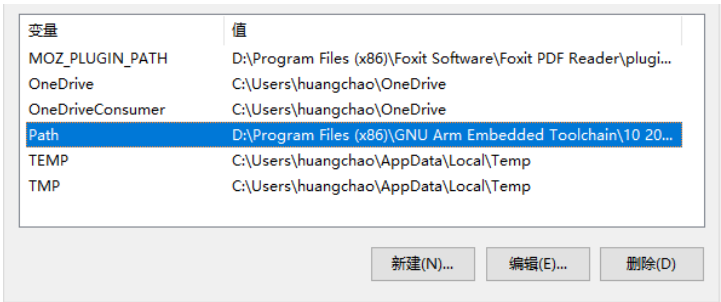


图 4-2. 添加好 gcc 和 mingw 的路径后， 点击 “确定” 按钮

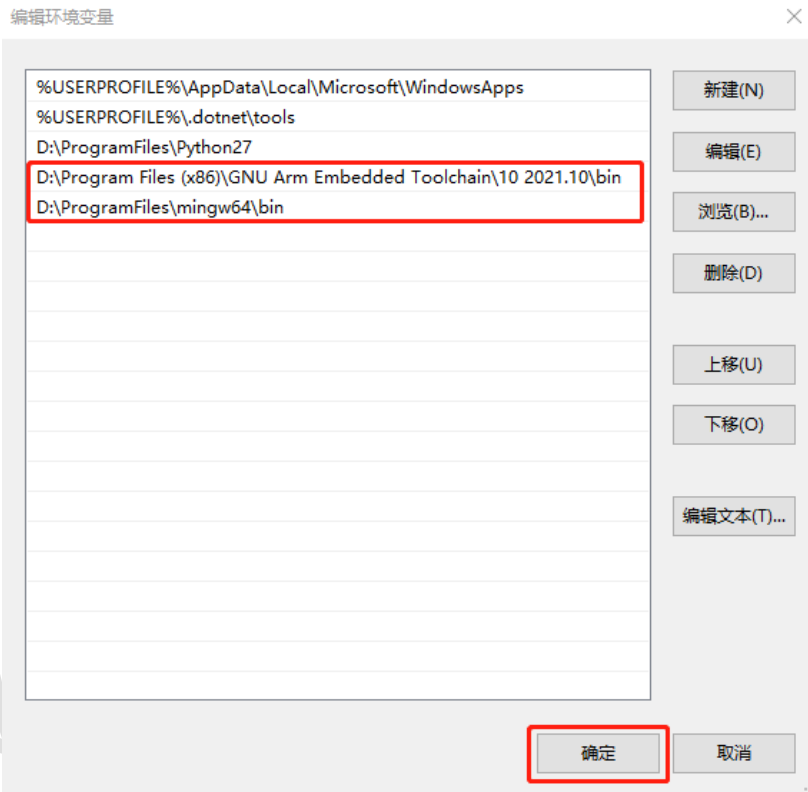
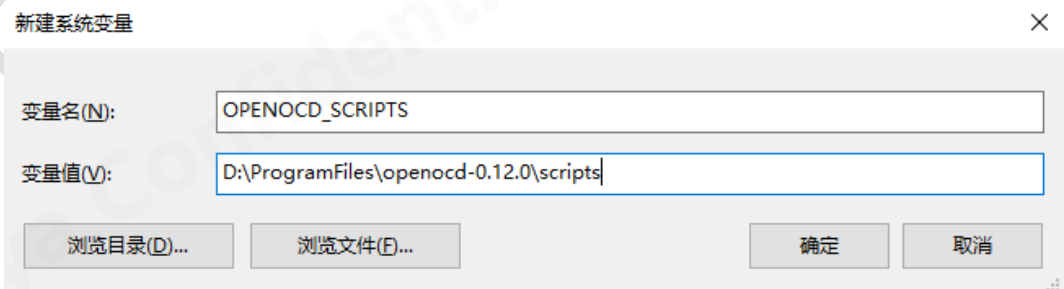


图 4-3. 新建系统变量， 变量名为“ OPENOCD_SCRIPTS” ， 变量值为 openocd scripts 目录



5 Makefile 文件

名称	说明
TARGET	生成的目标文件(hex/bin/elf)名字
OPT	编译优化等级
BUILD_DIR	生成的目标文件路径
C_SOURCES	待编译的 C 源文件列表
ASM_SOURCES	待编译的 S 汇编文件列表
C_DEFS	预处理宏定义
AS_INCLUDES	汇编文件包含目录
C_INCLUDES	C 源文件包含目录

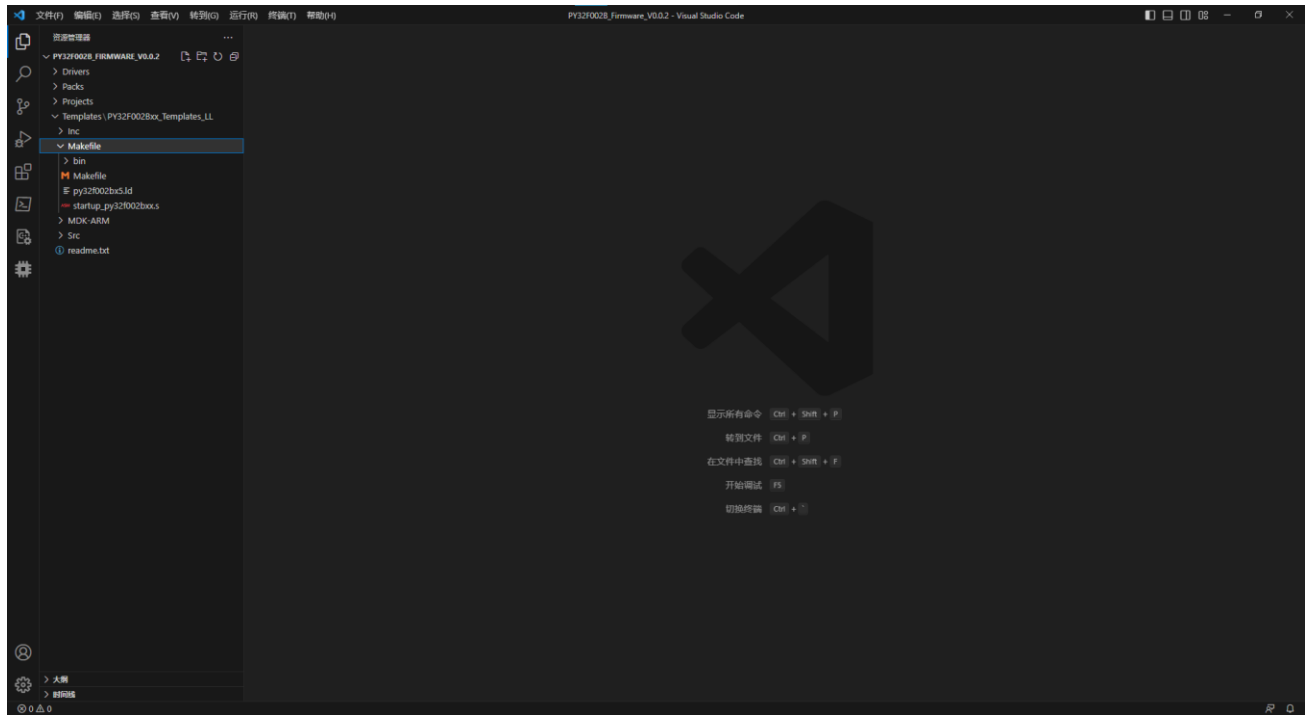
6 *.ld 链接文件

名称	说明
_Min_Heap_Size	设置堆大小
_Min_Stack_Size	设置栈大小
RAM (xrw)	SRAM 起始地址及大小
FLASH (rx)	FLASH 起始地址及大小

7 编辑、编译、下载

7.1 VSCode 编辑

图 7.1-1. 先解压 PY32F002B_Firmware_V0.0.2.rar，然后使用 VSCode 打开解压后的文件夹



7.2 GCC 编译

图 7.2-1. 右击需要编译的 Makefile 文件夹，点击“在集成终端中打开”

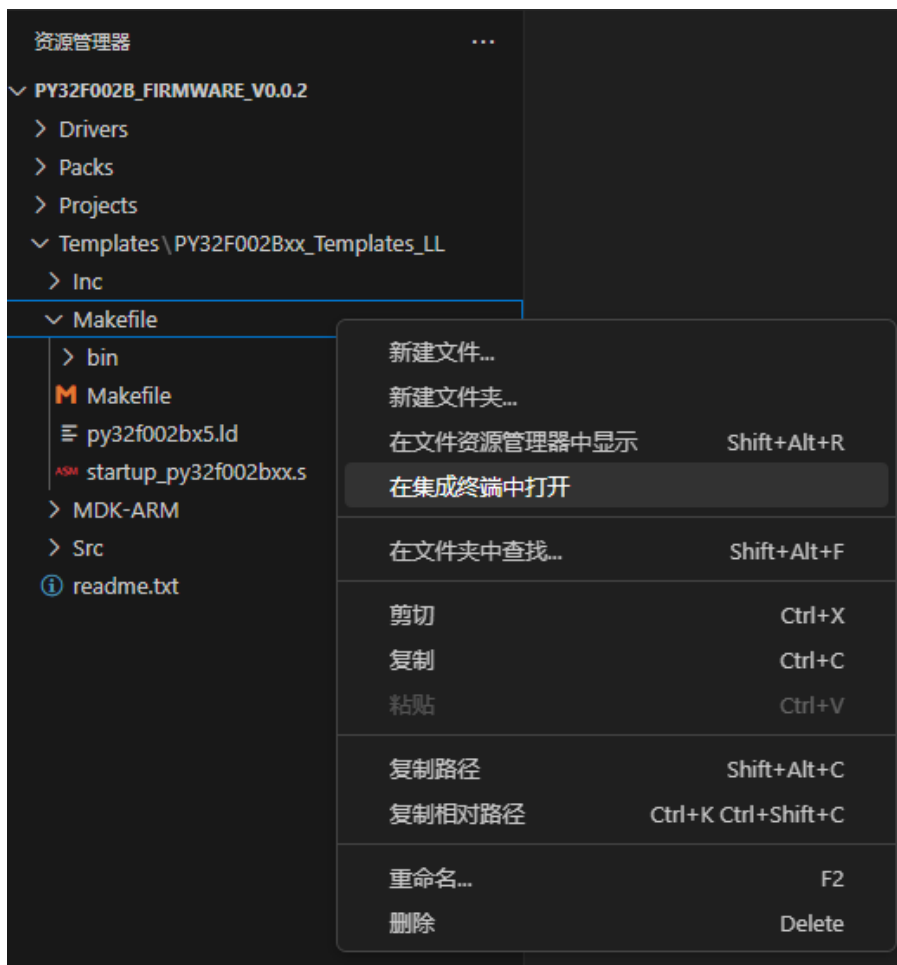


图 7.2-2. 在打开的 Powershell 终端中输入 make 后按键盘“Enter”键

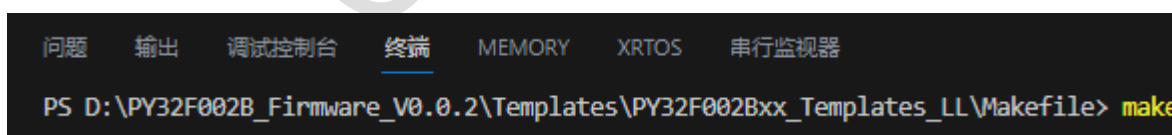
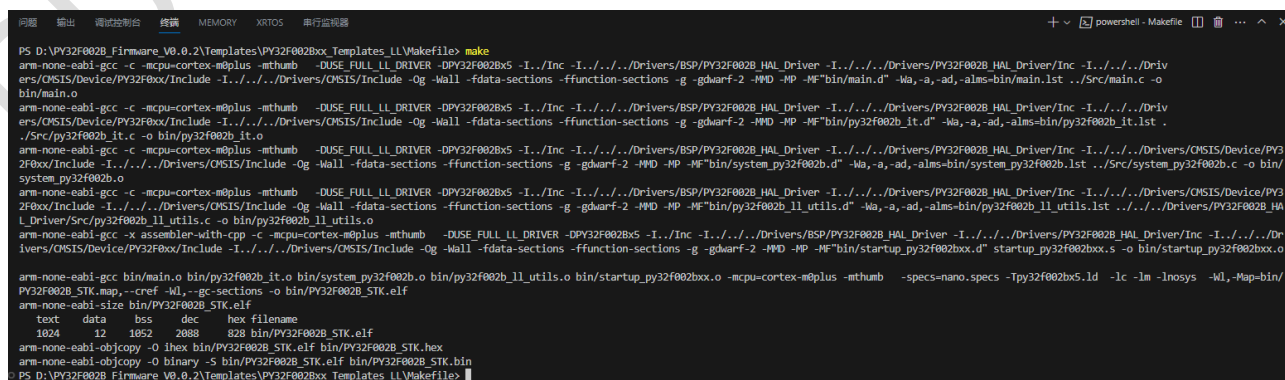


图 7.2-3. 编译完成生成 hex,bin,elf 三种格式的目标文件



7.3 CooFlash 软件下载

请参考文档《PY32 微控制器 CooFlash 下载软件用户手册》。

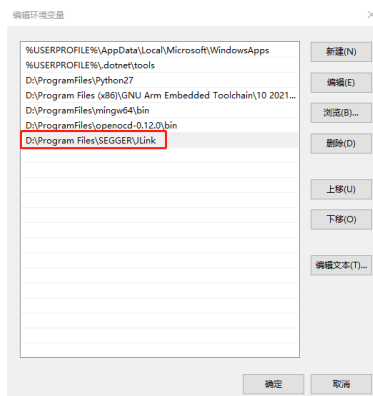
7.4 JFlash 软件下载

请参考文档《PY32 微控制器 JFlash 下载软件用户手册》。

PUYA CONFIDENTIAL

7.5 JFlash 命令行下载

图 7.5-1. 添加 JFlash 安装目录至用户环境变量



- JFlash 下载命令

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

图 7.5-2. JFlash 下载过程中

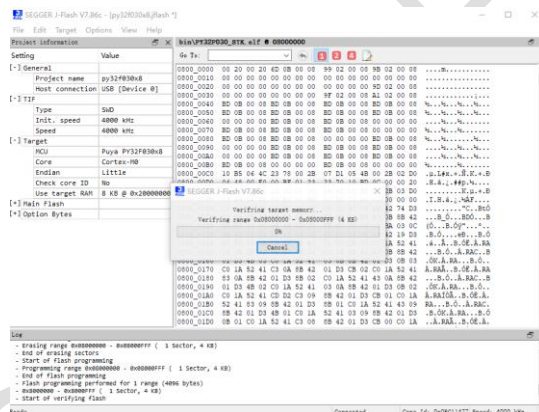
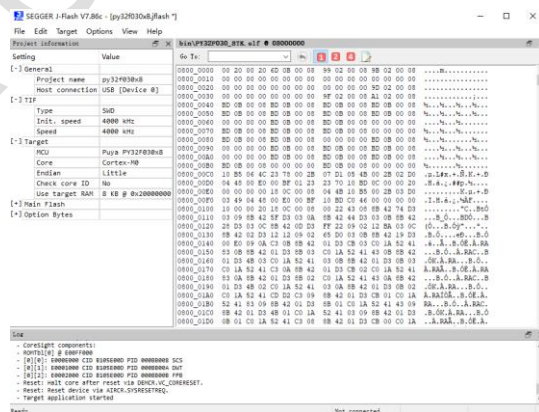
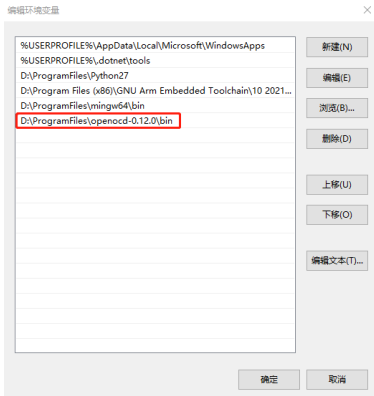


图 7.5-3. JFlash 下载完成



7.6 OpenOCD 命令行下载

图 7.6-1. 添加 openocd bin 安装目录至用户环境变量



● OpenOCD 下载命令

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"

图 7.6-2. VSCode powershell 终端中输入上面的命令开始下载

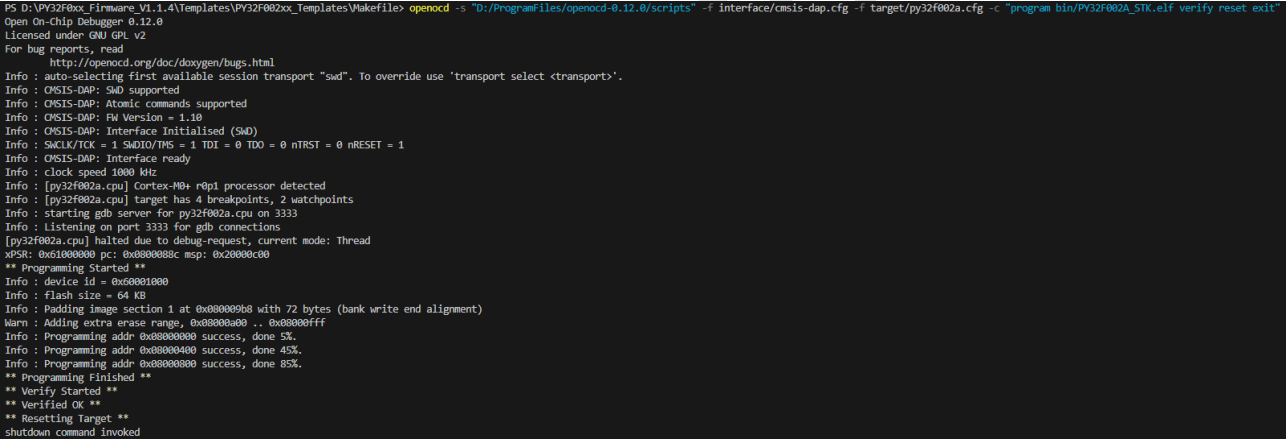


表 7.6-1. openocd cfg 文件列表

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 创建和使用 VSCode 任务

8.1 创建编译和下载任务 – tasks.json

图 8.1-1.使用 VSCode 打开 Makefile 文件夹



图 8.1-2.菜单栏依次选择 “终端”、“配置任务”、“使用模板创建 tasks.json 文件”

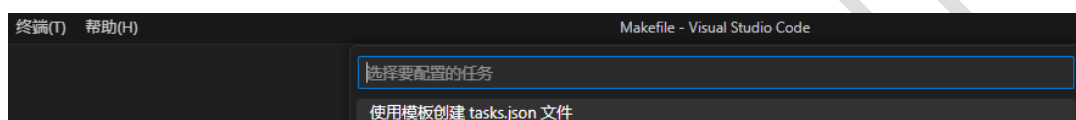


图 8.1-3.选择 “Others 运行任意外部命令的示例”

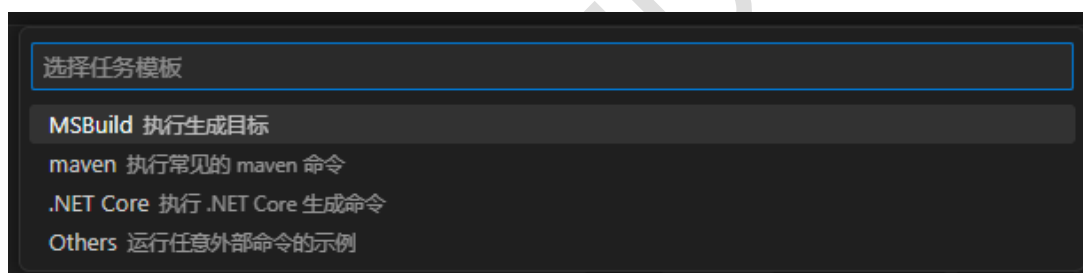


图 8.1-4.VSCode 自动生成 tasks.json 文件

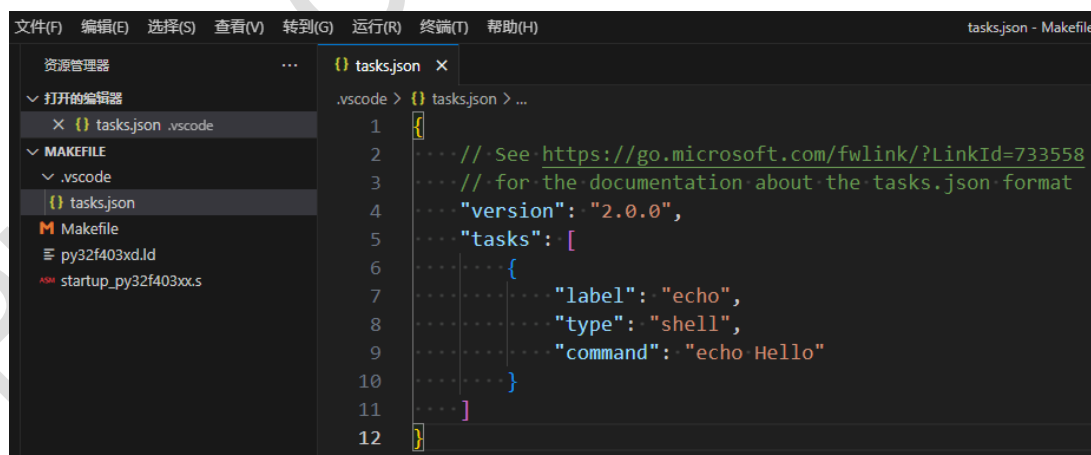


图 8.1-5.修改 tasks.json 文件并保存

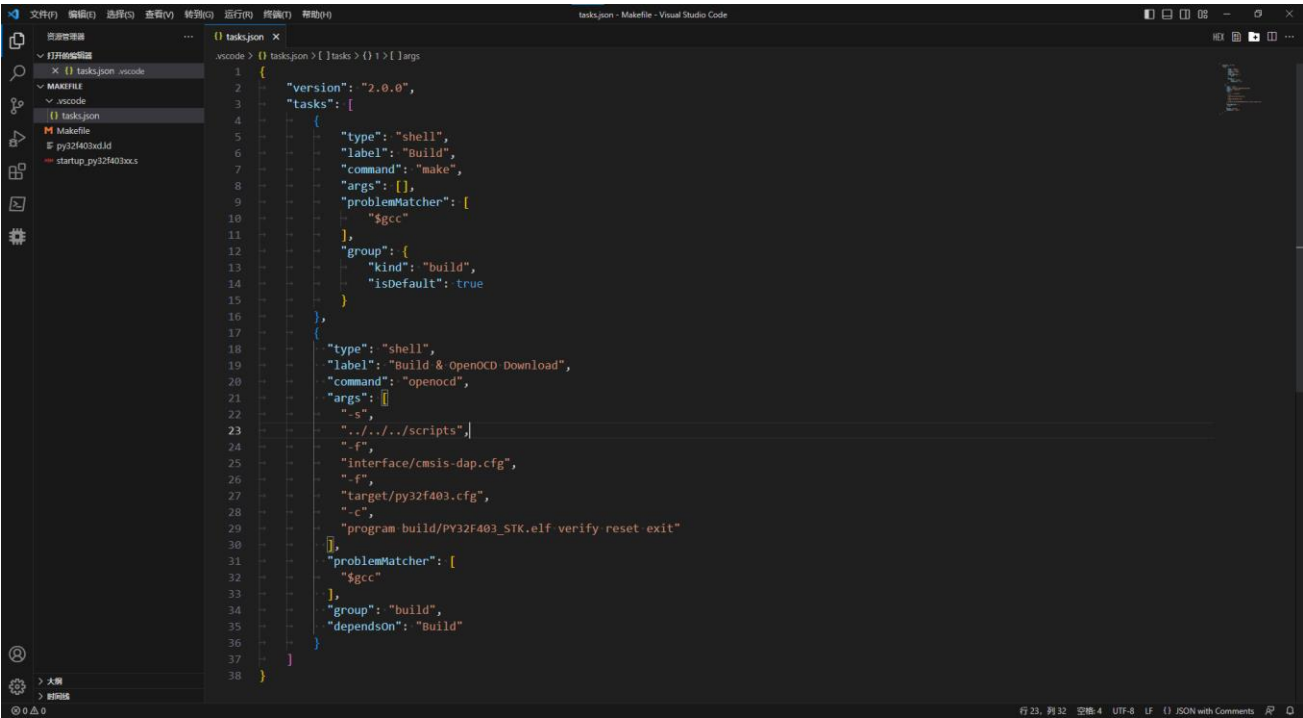


图 8.1-6.菜单栏依次选择 “终端 ”、“ 运行任务 ”

出现两个新建的任务: Build 和 Build & OpenOCD Download

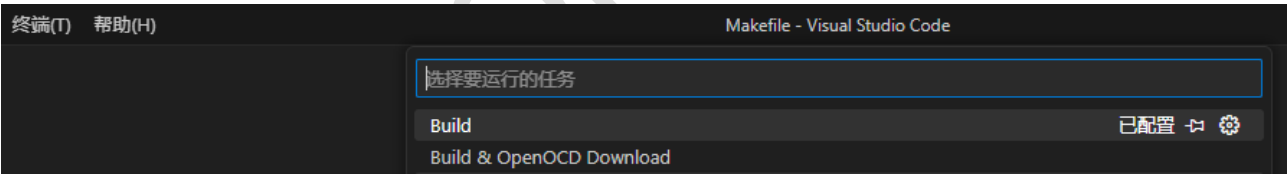
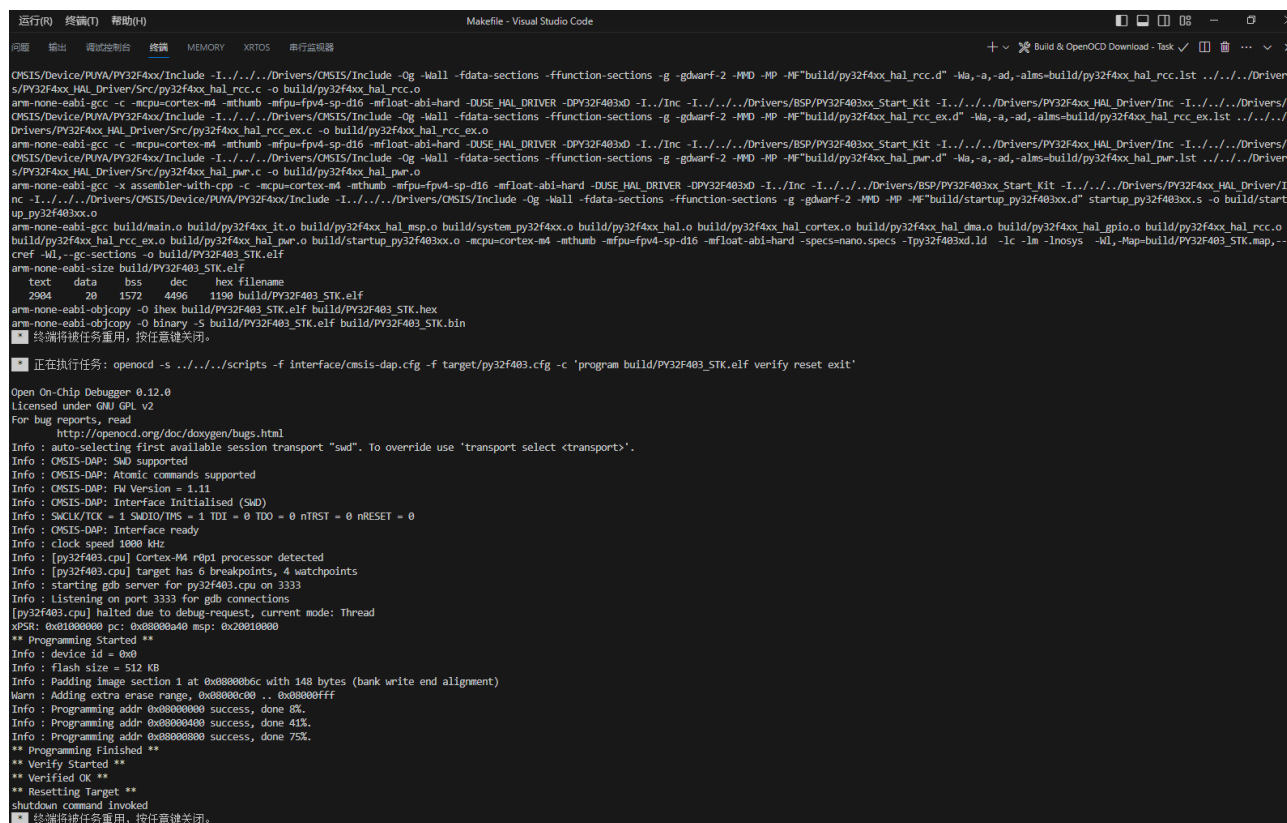


图 8.1-7.拷贝 scripts 脚本文件

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

图 8.1-8.运行 Build & OpenOCD Download 任务，依次执行编译和下载



```
运行(内) 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 ✓ 0 ... X
CMSIS/Device/PANA/PY32F40x/Include -I../Drivers/CMSIS/Include -Og -Wall -fdata-sections -ffunction-sections -g -gdwarf-2 -MD -MF"build/py32f40x_hal_rcc.d" -Ma,-a,-ad,-alms-build/py32f40x_hal_rcc.lst ../Drivers/
s/PY32F40x_HAL_Driver/Src/py32f40x_hal_rcc.c -o build/py32f40x_hal_rcc.o
arm-none-eabi-gcc -c -mcpu=cortex-m4 -mthumb -mfpv4-sp-d16 -mfloat-abi=hard -DUSE_HAL_DRIVER -DPY32F403x0 -I../Inc -I../Drivers/BSP/PY32F403xx_Start_Kit -I../Drivers/PY32F40x_HAL_Driver/Inc -I../Drivers/
CMSIS/Device/PANA/PY32F40x/Include -I../Drivers/CMSIS/Include -Og -Wall -fdata-sections -ffunction-sections -g -gdwarf-2 -MD -MF"build/py32f40x_hal_rcc_ex.d" -Ma,-a,-ad,-alms-build/py32f40x_hal_rcc_ex.lst ../Driver
s/PY32F40x_HAL_Driver/Src/py32f40x_hal_rcc_ex.c -o build/py32f40x_hal_rcc_ex.o
arm-none-eabi-gcc -c -mcpu=cortex-m4 -mthumb -mfpv4-sp-d16 -mfloat-abi=hard -DUSE_HAL_DRIVER -DPY32F403xx -I../Inc -I../Drivers/BSP/PY32F403xx_Start_Kit -I../Drivers/PY32F40x_HAL_Driver/Inc -I../Drivers/
CMSIS/Device/PANA/PY32F40x/Include -I../Drivers/CMSIS/Include -Og -Wall -fdata-sections -ffunction-sections -g -gdwarf-2 -MD -MF"build/py32f40x_hal_pwm.d" -Ma,-a,-ad,-alms-build/py32f40x_hal_pwm.lst ../Driver
s/PY32F40x_HAL_Driver/Src/py32f40x_hal_pwm.c -o build/py32f40x_hal_pwm.o
arm-none-eabi-gcc -x assembler-with-cpp -c -mcpu=cortex-m4 -mthumb -mfpv4-sp-d16 -mfloat-abi=hard -DUSE_HAL_DRIVER -DPY32F403xx -I../Inc -I../Drivers/BSP/PY32F403xx_Start_Kit -I../Drivers/PY32F40x_HAL_Driver/I
nc -I../Drivers/CMSIS/Device/PANA/PY32F40x/Include -I../Drivers/CMSIS/Include -Og -Wall -fdata-sections -ffunction-sections -g -gdwarf-2 -MD -MF"build/startup_py32f403xx.d" startup_py32f403xx.s -o build/start
up_py32f403xx.o
arm-none-eabi-gcc build/main.o build/py32f40x_it.o build/py32f40x_hal_msp.o build/system_py32f40x.o build/py32f40x_hal.o build/py32f40x_hal_cortex.o build/py32f40x_hal_dma.o build/py32f40x_hal_gpio.o build/py32f40x_hal_rcc.o
build/py32f40x_hal_rcc_ex.o build/py32f40x_hal_pwm.o build/startup_py32f403xx.o -mcpu=cortex-m4 -mthumb -mfpv4-sp-d16 -mfloat-abi=hard -specs=nano.specs -Tpy32f403xx.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      28    1572    4496    1180 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/IO = 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
yPSR: 0x01000000 pc: 0x00000040 msp: 0x20010000
** Programming Started **
Info: device id = 0x0
Info: flash size = 512 KB
Info: Padding image section 1 at 0x000000fc with 148 bytes (bank write end alignment)
Warn: Adding extra erase range: 0x00000000 - 0x000000ff
Info: Programming addr 0x00000000 success, done 0%.
Info: Programming addr 0x00000040 success, done 41%.
Info: Programming addr 0x00000080 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 创建调试任务 – launch.json

图 8.2-1.最左边工具栏选择 “运行和调试”，

然后点击蓝色字体 “创建 launch.json”，选择 “Cortex Debug”

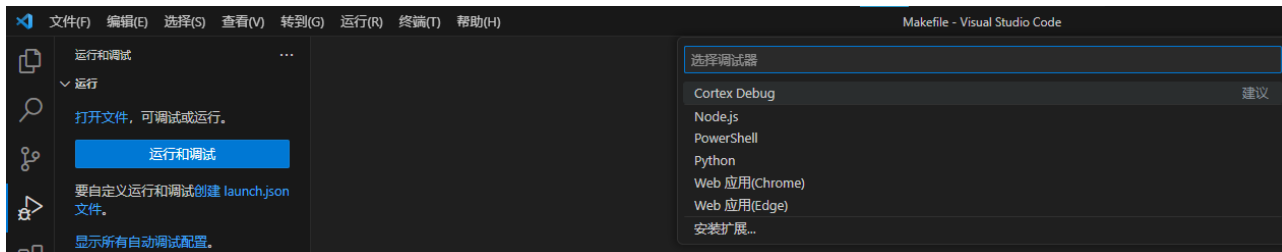


图 8.2-2.修改 launch.json 文件

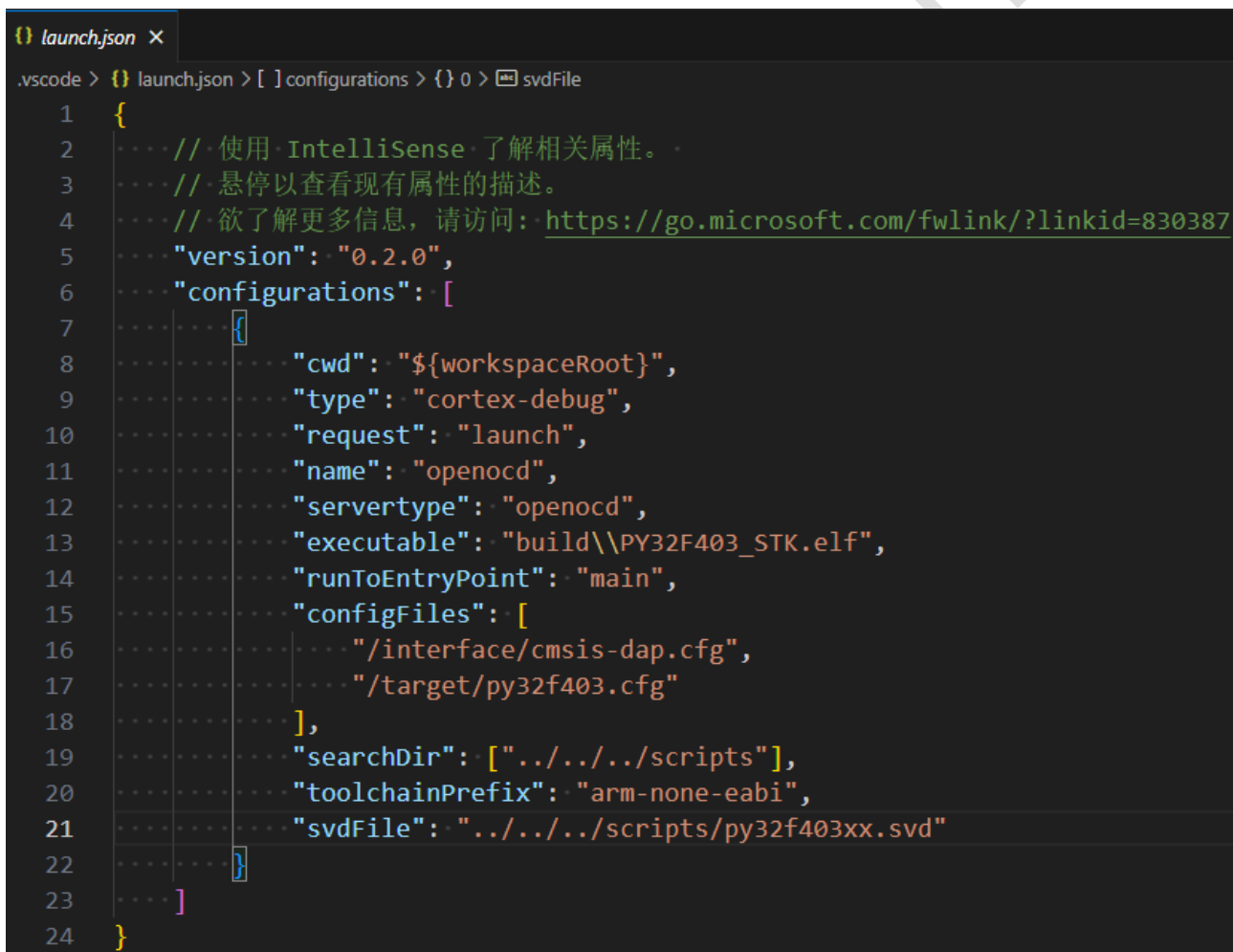
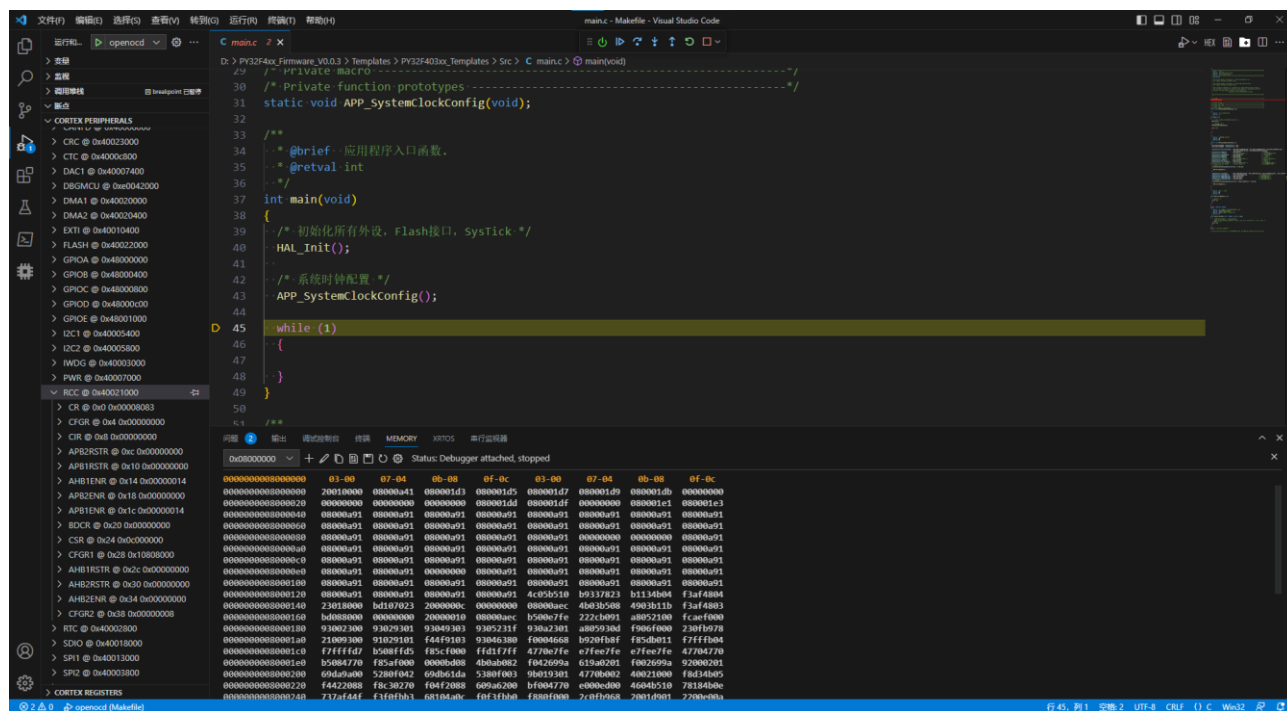
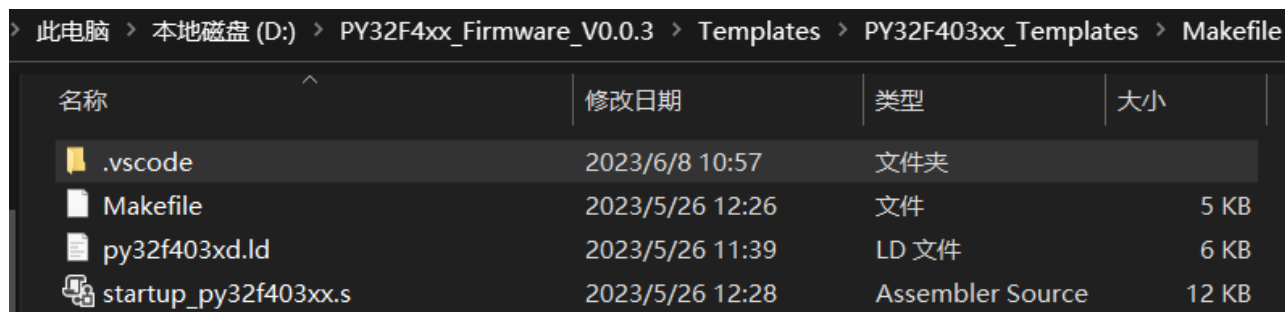


图 8.2-3. 点击左边任务窗口“openocd”左边的绿色三角按钮开始 Cortex-Debug 调试



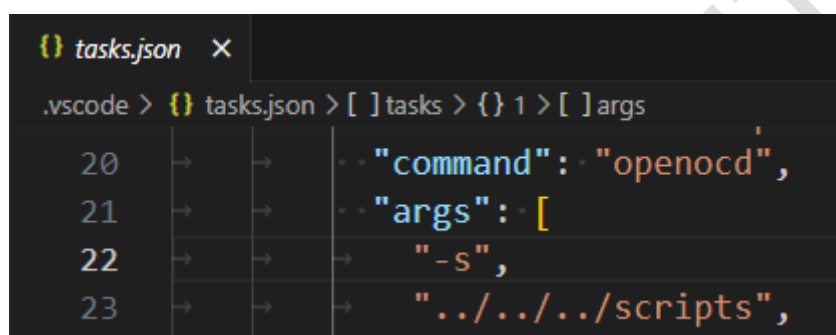
8.3 拷贝任务文件夹 – .vscode

图 8.3-1.将 Makefile 文件夹下的.vscode 文件夹全部拷贝至其它工程的 Makefile 文件夹下



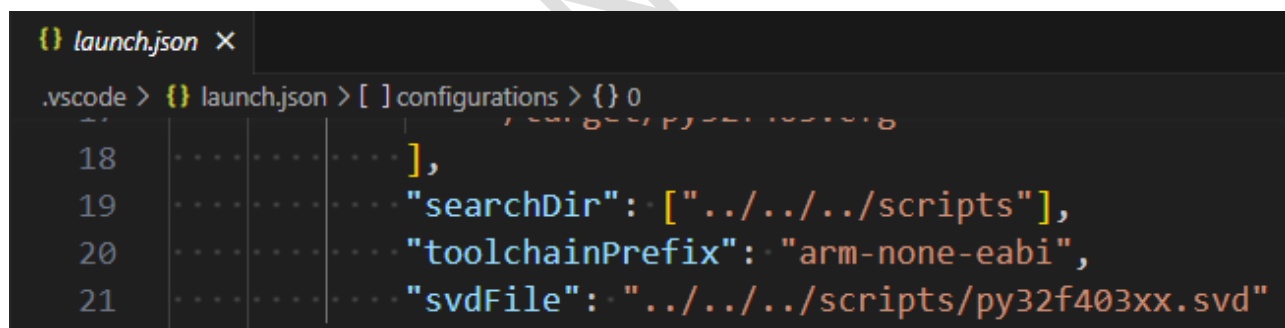
名称	修改日期	类型	大小
.vscode	2023/6/8 10:57	文件夹	
Makefile	2023/5/26 12:26	文件	5 KB
py32f403xd.ld	2023/5/26 11:39	LD 文件	6 KB
startup_py32f403xx.s	2023/5/26 12:28	Assembler Source	12 KB

图 8.3-2.修改 tasks.json 文件里面 scripts 文件夹的路径



```
{} tasks.json ×  
.vscode > {} tasks.json > [ ] tasks > {} 1 > [ ] args  
20 | | | | "command": "openocd",  
21 | | | | "args": [  
22 | | | |   "-s",  
23 | | | |   "../../../../../scripts",
```

图 8.3-3.修改 launch.json 文件里面 scripts 文件夹及 svd 文件的路径



```
{} launch.json ×  
.vscode > {} launch.json > [ ] configurations > {} 0  
18 | | | | ],  
19 | | | | "searchDir": ["../../../../scripts"],  
20 | | | | "toolchainPrefix": "arm-none-eabi",  
21 | | | | "svdFile": "../../../../scripts/py32f403xx.svd"
```

9 版本历史

版本	日期	更新记录
V1.0	2024-06-12	初版



Puya Semiconductor Co., Ltd.

IMPORTANT NOTICE

Puya Semiconductor reserves the right to make changes without further notice to any products or specifications herein. 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 convey any license under its patent rights or other rights nor the rights of others.