
目 录

一、 工具集下载	3
1.1、 vs code 下载	3
1.2、 clang 下载	3
1.3、 git 下载	4
1.4、 openocd 下载	4
1.5、 arm-none-eabi-gcc 下载	5
1.6、 arm 的交叉编译工具集, GNU toolchain 下载	5
1.7、 cubemx 下载	5
1.8、 st cubeprogrammer 下载	5
1.9、 gnu-mcu-eclipse 下载	5
1.10、 参考教程	6
二、 工具安装注意事项	6
2.1 vscode	6
2.2 clang/LLVM	6
2.3 git	6
2.4 Openocd	6
2.5 arm-none-eabi-gcc	6
2.6 cubemx	7
2.7 gnu mcu eclipse 插件	7
2.8 stm32cubeprogrammer	7
三、 软件安装检查	7

四、	Cubemx 生成独立工程	8
4.1	建立工程	8
4.2	使用 vscode 打开工程	10
五、	Vscode 配置	10
5.1	添加配置文件	11
六、	连接测试	13
6.1	在终端使用 stm32cubeprogrammer 连接 stm32	13
6.2	Stlink 连接 stm32 芯片测试:	14
6.3	Makefile 修改	14
七、	Vscode 下 json 文件配置	15
7.1	c_cpp_properties.json 文件	15
7.2	launch.json	16
7.3	tasks.json	17
附件	(命令行下载程序代码):	18

使用 vscode 开发 stm32 实战

Vscode+clang+openocd+gcc-arm-none-eabi+cubemx+stlink+gdb

开源越来越强，传统开发软件授权壁垒，是这个向导的主因。

学习新知识、天天向上，生命在于折腾，是次因。

习惯了 windows 下傻瓜式的 IED 环境，对 linux 有恐惧，想战胜这份恐惧。

所以结合网上丰富的非正规教程，就成了这篇备忘录。

一、工具集下载

1.1、vs code 下载

源于微软的一款文本编辑器，由于免费和丰富的插件生态，主要用来程序编写、调试。

软件链接 <https://code.visualstudio.com/Download>

安装过程 windows 风格，注意有个复选框，添加“右键打开文件和文件夹”选项即可。

1.2、clang 下载

其另一个名字交 LLVM,用于提供语法补全代码格式化等功能，正常完成安装后将软件的安装目录下的 bin 文件夹设置进入环境变量 PATH 中。

链接: <http://releases.llvm.org/9.0.0/>

后边是版本号，直接修改版本号即可进入对应版本的下载链接。进入网页后根据自己系统和需求下载对应的文件。我直接下载了 exe 安装包。安装后添加 bin 目录到系统 path 环境变量。

← → ↻ ⓘ 不安全 releases.llvm.org/9.0.0/		
Index of /9.0.0/		
../	19-Sep-2019 15:00	-
docs/	19-Sep-2019 15:00	-
projects/	19-Sep-2019 15:00	-
tools/	19-Sep-2019 15:00	-
LICENSE.TXT	19-Sep-2019 15:00	15141
LLVM-9.0.0-win32.exe	20-Sep-2019 11:30	145281242
LLVM-9.0.0-win32.exe.sig	20-Sep-2019 11:30	566
LLVM-9.0.0-win64.exe	20-Sep-2019 11:30	157915277
LLVM-9.0.0-win64.exe.sig	20-Sep-2019 11:30	566
cfe-9.0.0.src.tar.xz	19-Sep-2019 15:00	13533024
cfe-9.0.0.src.tar.xz.sig	19-Sep-2019 15:00	566
clang-tools-extra-9.0.0.src.tar.xz	19-Sep-2019 15:00	2183436
clang-tools-extra-9.0.0.src.tar.xz.sig	19-Sep-2019 15:00	566
clang-tools-extra-doxxygen-9.0.0.tar.xz	19-Sep-2019 15:00	7012160
clang-doxxygen-9.0.0.tar.xz	19-Sep-2019 15:00	66921096
compiler-rt-9.0.0.src.tar.xz	19-Sep-2019 15:00	1993084
compiler-rt-9.0.0.src.tar.xz.sig	19-Sep-2019 15:00	566
hans-eps-key.asc	19-Sep-2019 15:00	3151
libcxx-9.0.0.src.tar.xz	19-Sep-2019 15:00	1814388
libcxx-9.0.0.src.tar.xz.sig	19-Sep-2019 15:00	566
libcxxabi-9.0.0.src.tar.xz	19-Sep-2019 15:00	552088
libcxxabi-9.0.0.src.tar.xz.sig	19-Sep-2019 15:00	566
libunwind-9.0.0.src.tar.xz	19-Sep-2019 15:00	90372
libunwind-9.0.0.src.tar.xz.sig	19-Sep-2019 15:00	566
lld-9.0.0.src.tar.xz	19-Sep-2019 15:00	1100608
lld-9.0.0.src.tar.xz.sig	19-Sep-2019 15:00	566
lldb-9.0.0.src.tar.xz	19-Sep-2019 15:00	9846624
lldb-9.0.0.src.tar.xz.sig	19-Sep-2019 15:00	566
llvm-9.0.0.src.tar.xz	19-Sep-2019 15:00	32994768
llvm-9.0.0.src.tar.xz.sig	19-Sep-2019 15:00	566
llvm-doxxygen-9.0.0.tar.xz	19-Sep-2019 15:00	157170300
openmp-9.0.0.src.tar.xz	19-Sep-2019 15:00	939036
openmp-9.0.0.src.tar.xz.sig	19-Sep-2019 15:00	566
polly-9.0.0.src.tar.xz	19-Sep-2019 15:00	8719928
polly-9.0.0.src.tar.xz.sig	19-Sep-2019 15:00	566
test-suite-9.0.0.src.tar.xz	19-Sep-2019 15:00	166742228
test-suite-9.0.0.src.tar.xz.sig	19-Sep-2019 15:00	566

1.3、 git 下载

提供 Git 支持和 MINGW64 指令终端。我主要用 git 带的终端了, 名叫 bash.exe

官网下载链接: <https://gitforwindows.org/>

1.4、 openocd 下载

为什么牛 X 的都是老外, 是一个学校的开源项目, 十来年了, 它的进化版本叫 visualGDB, 好像是, 收费。提供一个 pc 和调试器之间的一个接口--gdb 服务。用 keil 哪有这么多事儿。这个文件下载下来不需要安装, 直接放到一个文件夹下, bin 目录添加系统 path 环境变量就行。

下载链接: <http://gnutoolchains.com/arm-eabi/openocd/> (还有好多其他的工具也在这里) (有的给的是这个链接, 这个里边的是源码, 还需要自己编译。

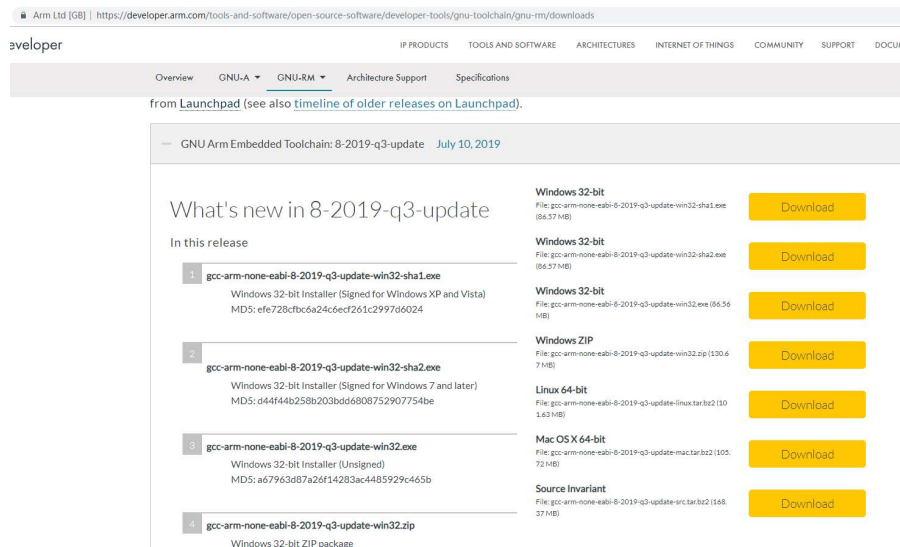
<https://sourceforge.net/>就是这里)

1.5、 arm-none-eabi-gcc 下载

1.6、 arm 的交叉编译工具集，GNU toolchain 下载

下载链接：<https://developer.arm.com/tools-and-software/open-source-software/developer-tools/gnu-toolchain/gnu-rm/downloads>

我下载的 gcc-arm-none-eabi-8-2019-q3-update-win32.exe 这个。



1.7、 cubemx 下载

stm32 程序初始化神器，ST 官网直接找吧。

1.8、 st cubeprogrammer 下载

用于 stm32 程序下载，ST 官网直接找吧。

1.9、 gnu-mcu-eclipse 下载

stm32 make 工具 gnu-mcu-eclipse, 是一个 eclipse 的插件, 可以在 windows 下构建 makefile 环境, 适合直接在终端界面 make 工程。可以作为 vscode 的辅助使用。

下载链接：<https://github.com/gnu-mcu-eclipse/windows-build-tools/releases>

1.10、参考教程

其他参考教程: <https://www.jianshu.com/u/b1ffe963c188>

<https://blog.csdn.net/zhengyangliu123>

写的比较好的两位博主。

<https://www.brobwind.com/archives/1291> (有关 gdb 和 openocd 的)

二、工具安装注意事项

2.1 vscode

添加“右键打开文件和文件夹”选项，安装好后添加一些常用的插件。

```
Chinese (Simplified) Language Pack for Visual Studio Code 中文界面
c/c++
c/c++ clang command adapter
c/c++ snippets
c++ intellisense
Chinese language... vsc 汉化
Cortex-debug (不在本教程范围, 后边再研究, 也可以用于调试)
Cortex-debug: device support
Gbk2utf8
Visual studio intellicode 语法支持、智能补全、颜色
```

2.2 clang/LLVM

bin 目录添加到环境变量

2.3 git

正常安装 在 vsc 中使用它的终端

2.4 Openocd

解压后放入自定义目录，然后其 bin 目录添加到环境变量

2.5 [arm-none-eabi-gcc](#)

添加“右键打开文件和文件夹”选项

2.6 cubemx

安装好后，下载相应 STM32 芯片的 pack 包

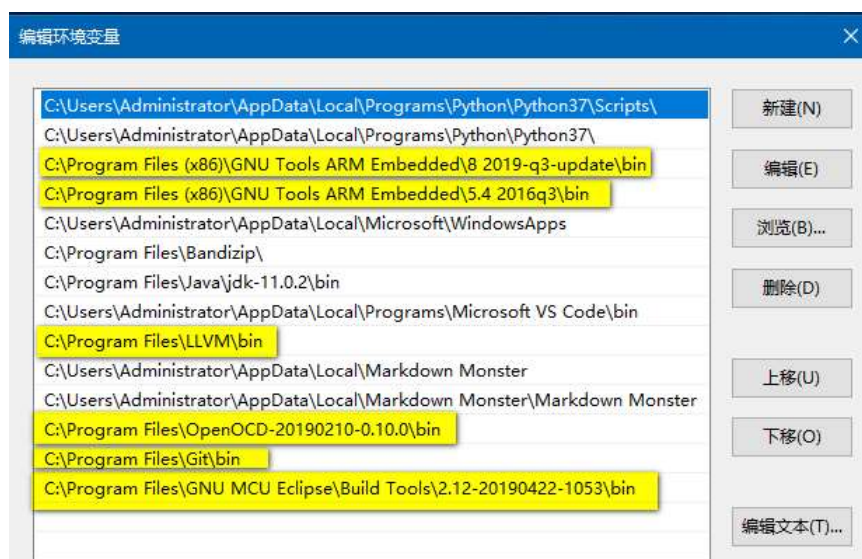
2.7 gun mcu eclipse 插件

解压到自定义文件夹，并添加 bin 目录到系统环境变量

2.8 stm32cubeprogrammer

安装好后，将 STM32_Programmer_CLI.exe 所在目录添加至系统环境变量。

仅供参考，系统环境变量如下：



三、 软件安装检查

Win+R 输入 cmd, 打开终端窗口, 验证 make、arm-none-eabi-gcc、openocd

功能。正常查出版本号，说明正确安装。随便一个终端工具都可以用。

```
管理员: C:\Windows\system32\cmd.exe
Microsoft Windows [版本 10.0.17763.737]
(c) 2018 Microsoft Corporation。保留所有权利。

C:\Users\Administrator>openocd -v
Open On-Chip Debugger 0.10.0 (2019-02-10) [https://github.com/sysprogs/openocd]
Licensed under GNU GPL v2
For bug reports, read
    http://openocd.org/doc/doxygen/bugs.html

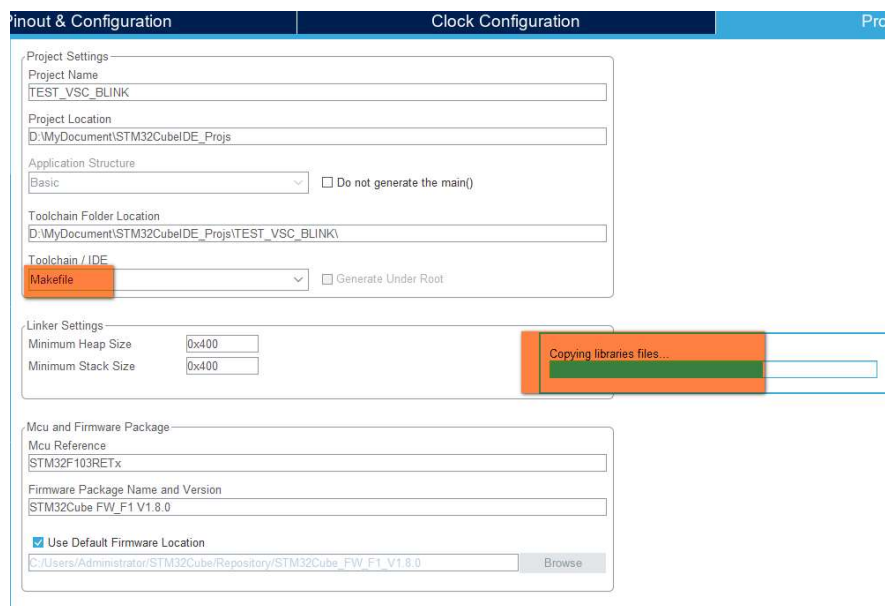
C:\Users\Administrator>make -v
GNU Make 4.2.1
Built for x86_64-w64-mingw32
Copyright (C) 1988-2016 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.

C:\Users\Administrator>arm-none-eabi-gcc --version
arm-none-eabi-gcc (GNU Tools for ARM Embedded Processors) 5.4.1 20160919 (release) [ARM/
3]
Copyright (C) 2015 Free Software Foundation, Inc.
This is free software; see the source for copying conditions.  There is NO
warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.
```

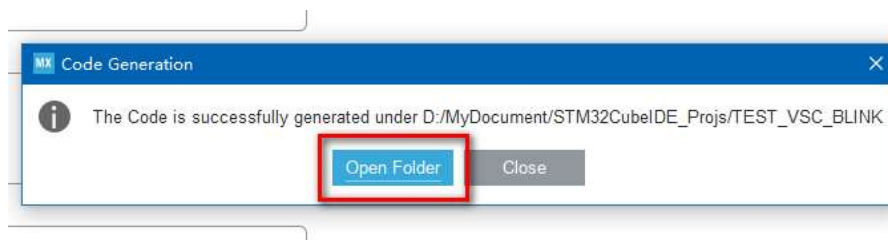
四、Cubemx 生成独立工程

4.1 建立工程

建工程的过程就略了，选择生成的工程类型：**makefile**



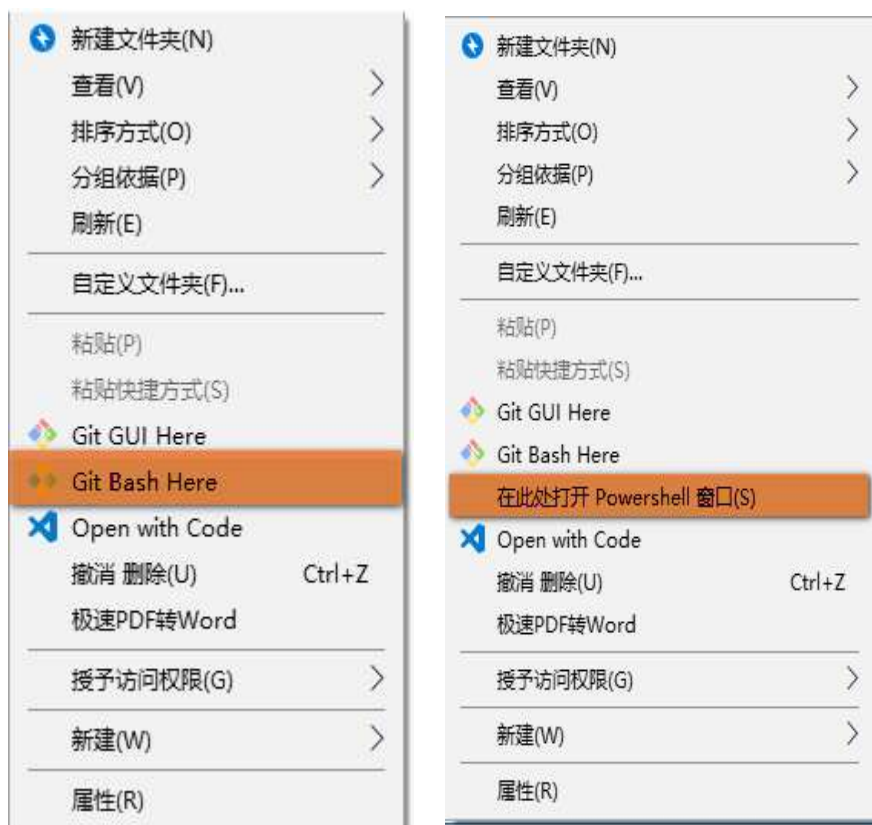
完成之后，选择 openfolder



生成的文件夹+文件如下(自动生成了 makefile):

32CubeIDE_Projs > TEST_VSC_BLINK				
名称	修改日期	类型	大小	
Drivers	2019-09-23 15:03	文件夹		
Inc	2019-09-23 15:05	文件夹		
Src	2019-09-23 15:05	文件夹		
.mxproject	2019-09-23 15:05	MXPROJECT 文件	6 KB	
Makefile	2019-09-23 15:05	文件	6 KB	
startup_stm32f103xe.s	2019-07-19 14:59	S 文件	13 KB	
STM32F103RETx_FLASH.ld	2019-09-23 15:05	LD 文件	6 KB	
TEST_VSC_BLINK.ioc	2019-09-23 15:05	STM32CubeMX	4 KB	

然后再该文件夹下，shift+右键，在弹出的菜单中选择 **git bash here** 或者在此处打开 **powershell** 窗口，输入 **make -j4**



```

MINGW64:/d/MyDocument/STM32CubeIDE_Projs/TEST_VSC_BLINK
Administrator@DESKTOP-15BM81N MINGW64 /d/MyDocument/STM32CubeIDE_Projs/TEST_VSC_
BLINK
$ make -j4
mkdir build
arm-none-eabi-gcc -c -mcpu=cortex-m3 -mthumb -DUSE_HAL_DRIVER -DSTM32F103xE -D
USE_HAL_DRIVER -DSTM32F103xE -IInc -IDrivers/STM32F1xx_HAL_Driver/Inc -IDrivers/
STM32F1xx_HAL_Driver/Inc/Legacy -IDrivers/CMSIS/Device/ST/STM32F1xx/Include -IDr
ivers/CMSIS/Include -IDrivers/CMSIS/Include -Og -Wall -fdata-sections -ffunction
-sections -g -gdwarf-2 -MMD -MP -MF"build/main.d" -Wa,-a,-ad,-alms=build/main.ls
t Src/main.c -o build/main.o
arm-none-eabi-gcc -c -mcpu=cortex-m3 -mthumb -DUSE_HAL_DRIVER -DSTM32F103xE -D

```

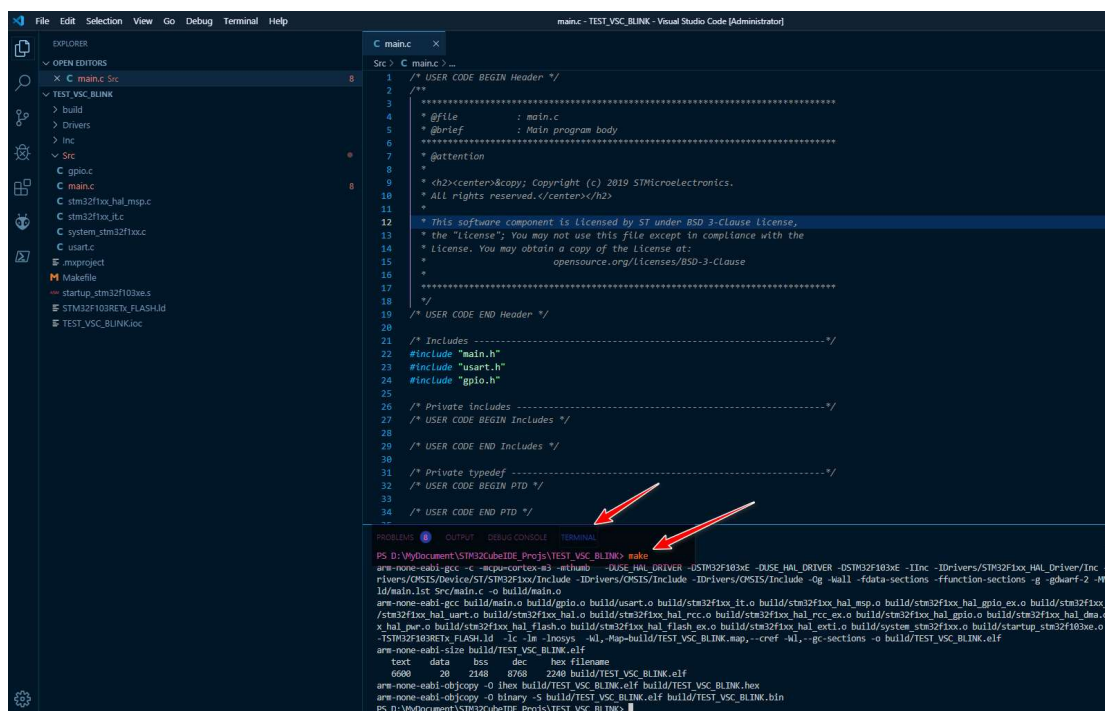
```
build/stm32f1xx_hal_dma.o build/stm32f1xx_hal_cortex.o build/stm32f1xx_hal_pwr.o build/stm32f1xx_hal_flash.o build/stm32f1xx_hal_flash_ex.o build/stm32f1xx_hal_exti.o build/system_stm32f1xx.o build/startup_stm32f103xe.o -mcpu=cortex-m3 -mtthumb -specs=nano.specs -TSTM32F103RETx_FLASH.ld -lc -lm -lnosys -Wl,-Map=build/TEST_VSC_BLINK.map,--cref -Wl,--gc-sections -o build/TEST_VSC_BLINK.elf
arm-none-eabi-size build/TEST_VSC_BLINK.elf
text data bss dec hex filename
6600 20 2148 8768 2240 build/TEST_VSC_BLINK.elf
arm-none-eabi-objcopy -O ihex build/TEST_VSC_BLINK.elf build/TEST_VSC_BLINK.hex
arm-none-eabi-objcopy -O binary -S build/TEST_VSC_BLINK.elf build/TEST_VSC_BLINK.bin

Administrator@DESKTOP-15BM81N MINGW64 /d/MyDocument/STM32CubeIDE_Projs/TEST_VSC_BLINK
$
```

见到如上的信息，说明已配置好 make 环境，并生成了正常的可用的程序。

4.2 使用 vscode 打开工程

在工程文件夹空白处右键，选择“**open with code**”



在下边箭头所指向的窗口选择 **terminal**，然后输入 **make**，正常的话，回正确输出编译过程和最后输出 elf 文件、hex 文件、bin 文件。

那么使用 vscode 开发 stm32 已经迈出了成功的第一步，剩下还有 99 步。

五、Vscode 配置

Vscode 配置 json 文件实现编译、下载、硬件复位、调试等功能

5.1 添加配置文件

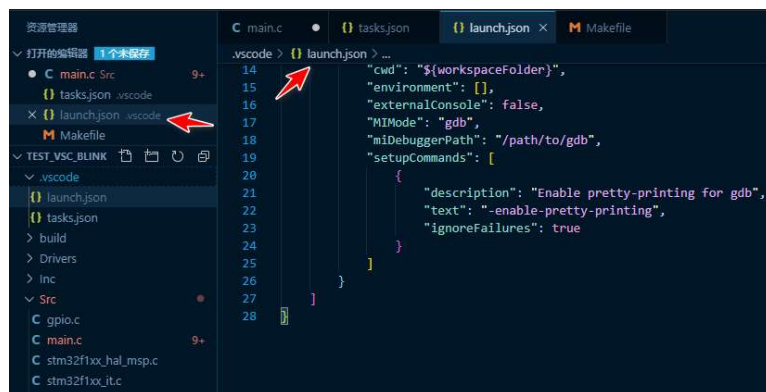
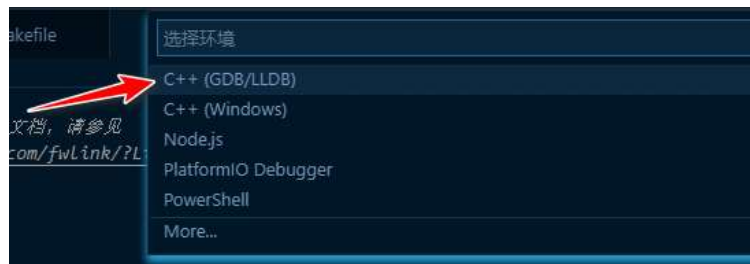
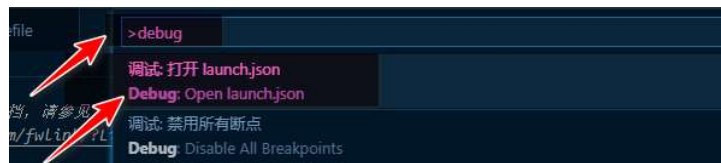
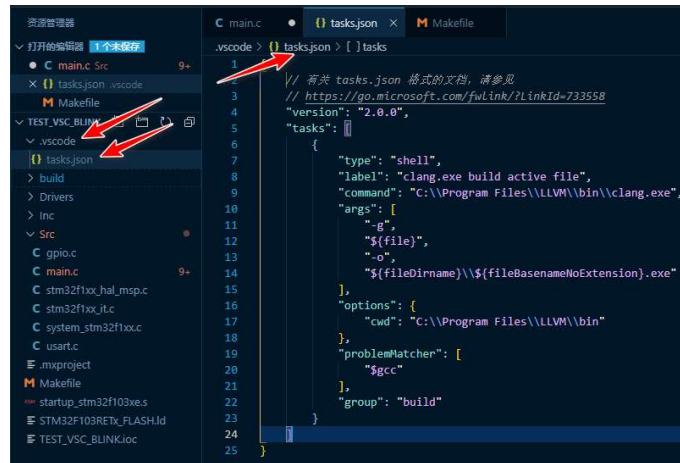
默认情况下，工程下是不含.vscode 的文件夹的，需要在 vscode 下生成这样的文件或者直接从别处拷贝一份过来。

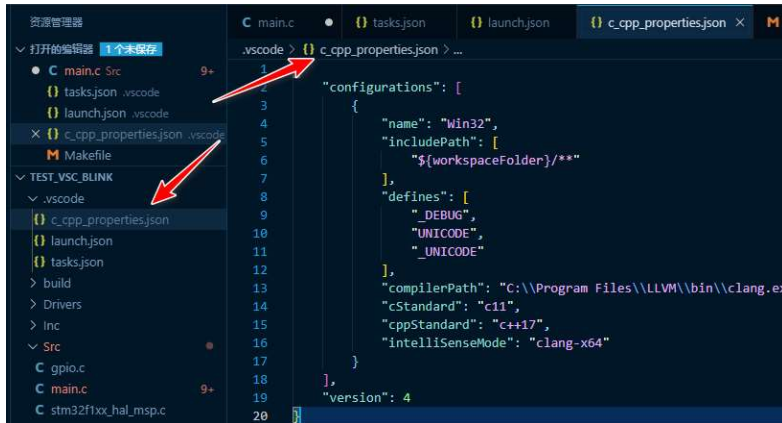
步骤：`ctrl+shift+p`，输入 `tasks`，选择 `configure task`，然后选择 `C/C++:clang.exe...`，这时候 vsc 会自动创建文件夹，并生成 `tasks.json` 模板文件，后期进行修改即可。

再次使用 `ctrl+shift+p`，输入 `debug`，选择 `open launch.json`，然后选择 `C++...`，结果会生成一个 `launch.json` 的文件。

同样的方式输入 `c c++`，找到图示的选择或者设置都可以，最终生成一个 `c_cpp_properties.json` 的文件。







六、 连接测试

6.1 在终端使用 stm32cubeprogrammer 连接 stm32

主要是使用它的 STM32_Programmer_CLI.exe 这个命令行工具。当然，安装 jlink、st-link utility 等也可以。

下图为 st-link utility 和 STM32cubeProgrammer 的 CLI 工具帮助信息，相对而言，Programmer 的功能更丰富，所以以这个为例来演示。

```
Microsoft Windows [版本 10.0.17134.960]
(c) 2018 Microsoft Corporation. 保留所有权利。

C:\Users\adminfish>ST-LINK_CLI -v
STM32 ST-LINK CLI v3.2.0.0
STM32 ST-LINK Command Line Interface

C:\Users\adminfish>ST-LINK_CLI -h
STM32 ST-LINK CLI v3.2.0.0
STM32 ST-LINK Command Line Interface

Available commands:
=====
-c Connect to the device using JTAG or SWD.
  Syntax: -c [ID=<id>/SN=<sn>] [JTAG/SWD SWCLK=<f>] [UR/HOTPLUG] [LPM]
  [ID=<id>] : id (Identifier) of ST-LINK [0..9] to use when multiple
            probes are connected to the host
  [SN=<sn>] : sn (Serial Number) of the chosen ST-LINK probe
  [UR] : Connect to target without reset
  [HOTPLUG] : Connect to target without halt or reset
  [LPM] : Activate debug in Low Power mode
  [Hrst] : Activate Hardware Reset mode
  [Srst] : Activate Software system Reset mode
  [Crst] : Activate Core Reset mode
  [Freq=<frequency>] : Frequency value in KHz
Example: -c ID=1 SWD SWCLK=5 UR LPM
Example: -c ID=1 JTAG JTAGCLK=6 UR
Note: When [ID=<id>] and [SN=<sn>] are not specified, the first
      ST-LINK with ID=0 will be selected
      Selection of ST-LINK by ID or SN should be used with:
      * V113Sx or greater ST-LINK firmware version
      * V2120Sx or greater ST-LINK/V2 firmware version
      * V2120Mx or greater ST-LINK/V2-1 firmware version
      [UR] available only with ST-LINK/V2 and in SWD mode
      For JTAG mode, connect under reset is available since
      ST-LINK/V2 firmware Version V2J15Sx
      The RESET pin of the JTAG connector (pin 15) should be connected
      to the device reset pin
      [HOTPLUG] available in SWD mode
      For JTAG mode, HotPlug Connect is available since
      ST-LINK/V2 firmware Version V2J15Sx
      [SWCLK=<f>] available only with ST-LINK/V2 and in SWD mode

-l List the corresponding firmware version and the unique Serial Number
  of every ST-LINK probe connected to the computer
Note: To have a correct SN the ST-LINK firmware version should be:
      * V113Sx or greater for ST-LINK
      * V2120Sx or greater for ST-LINK/V2
      * V2120Mx or greater for ST-LINK/V2-1

-r8 Read memory. Syntax: -r8 <Address> <NumBytes>
-r16 Read memory. Syntax: -r16 <Address> <NumHalfWords>
-r32 Read memory. Syntax: -r32 <Address> <NumWords>
-w8 Write 8-bit data. Syntax: -w8 <Address> <data>
-w32 Write 32-bit data. Syntax: -w32 <Address> <data>
-w64 Write 64-bit data. Syntax: -w64 <Address> <data>

- Core commands =====
```

```
C:\Users\Administrator>STM32_Programmer_CLI -v
STM32CubeProgrammer v2.1.0

Error: Connection to target must be established first.

C:\Users\Administrator>STM32_Programmer_CLI -h
STM32CubeProgrammer v2.1.0

Usage :
STM32_Programmer_CLI.exe [command_1] [Arguments_1] [command_2] [Arguments_2]...

Generic commands:
-?, -h, --help : Show this help
--version, -version : Displays the tool's version
-l, --list : List all available communication interfaces
  <uart> : UART interface
  <usb> : USB interface
  <st-link> : st-link interface
-q, --quietMode : Enable quiet mode. No progress bar displayed
-log, --log : Store the detailed output in log file
  [file_path.log] : Path of the log file, default path = %HOME%/STM32Programmer/trace.log
-vb, --verbosity : Specify verbosity level
  <Level> : Verbosity level, value in [1, 2, 3]

Available commands for STM32 MCU:
--skipErase : Skip sector erase before programming
-sl, --safelib : Add a segment into a firmware file (elf, bin, hex, srec) containing computed CRC values
  To use only with the safety lib component
  <file_path> : File path to be modified
  <start address> : Flash memory start address
  <end address> : Flash memory end address
  <slice_size> : Size of data per CRC value
-ms, --mergesbfsu : Add a binary header and a sbfsu segment to an elf file
  <elf_file_path> : File path to be modified
  <header_file_path> : Header file path
  <sbfsu_file_path> : SBFSU file path
-c, --connect : Establish connection to the device
  <port=<PortName> : Interface identifier. ex COM1, /dev/ttySO, usb1, JTAG, SWD...

UART port optional parameters:
[baudrate] : Baudrate, ex: 115200, 9600, etc, default 115200
[p=parity] : Parity bit, value in {NONE, ODD, EVEN}, default EVEN
[db=<data_bits>] : Data bit, value in {6, 7, 8} ..., default 8
[st=<stop_bits>] : Stop bit, value in {1, 1.5, 2} ..., default 1
[fc=<flowControl>] : Flow control
  Value in {OFF, Hardware, Software} ..., default OFF
  Not supported for STM32MP
[noinit=<noinit_bit>] : Set No Init bits, value in {0, 1} ..., default 0
[console] : Enter UART console mode

JTAG/SWD debug port optional parameters:
[freq=<frequency>] : Frequency in KHz, Default frequencies:
  4000 SWD 9000 JTAG with STLINKv2
  24000 SWD 21333 with STLINKv3
[index=<index>] : Index of the debug probe, default index 0
[sn=<serialNumber>] : Serial Number of the debug probe
[ap=<accessPort>] : Access Port index to connect to, default ap 0
```

终端输入命令：`STM32_Programmer_CLI -c port=SWD`，结果如图所示，可以显示连接到的芯片的信息，说明已正确连接。前提是硬件正常、正确连接、供电正常。

使用终端成功下载程序

使用 vsc 修改 makefile 后再终端下载程序

Makefile 添加代码如下:

14 / 18

```
@STM32_Programmer_CLI -c port=SWD -d $(BUILD_DIR)/$(TARGET).hex
-v -s 0x08000000
```

添加 update 段，功能是-c 连接 device，port=SWD，使用 swd 接口，-d 下载，后边为下载的文件，-v Verify，-s start，后边是下载首地址。

```
C main.c      C main.h      tasks.json      launch.json      c_cpp_properties.json      M Makefile •  
M Makefile  
192 -include $(wildcard $(BUILD_DIR)/*.d)  
193  
194 #####  
195 # update  
196 #####  
197 update:  
198     @STM32_Programmer_CLI -c port=SWD -d $(BUILD_DIR)/$(TARGET).hex -v -s 0x08000000  
199  
200 # *** EOF ***
```

问题 10 输出 调试控制台 终端

```
Administrator@DESKTOP-15B8IIN MINGW64 /d/MyDocument/STM32CubeIDE_Projs/TEST_VSC_BLINK  
$ make update
```

```
-----  
STM32CubeProgrammer v2.1.0  
-----  
  
ST-LINK SN   : 53FF6E064887565361292287  
ST-LINK FW   : V2J3S57  
Voltage      : 3.25V  
SWD freq     : 4000 KHz  
Connect mode : Normal  
Reset mode   : Software reset  
Device ID    : 0x414  
Device name   : STM32F101/F103 High-density  
Flash size   : 512 KBytes  
Device type   : MCU  
Device CPU    : Cortex-M3  
  
Memory Programming ...  
Opening and parsing file: TEST_VSC_BLINK.hex  
File          : TEST_VSC_BLINK.hex  
Size          : 6620 Bytes  
Address       : 0x08000000  
  
Erasing memory corresponding to segment 0:  
Erasing internal memory sectors [0 3]  
Download in Progress:  
██████████ 100%  
  
File download complete  
Time elapsed during download operation: 00:00:00.732  
  
Verifying ...
```

七、 Vscode 下 json 文件配置

7.1 c_cpp_properties.json 文件

主要有添加 include 路径, 编译器路径, 宏定义等

```
{
  "configurations": [
    {
      "name": "Win32",
      "includePath": [
        "${workspaceFolder}Drivers/STM32F1xx HAL Driver/Inc",

```

```

        "${workspaceFolder}Drivers/STM32F1xx_HAL_Driver/Inc/Legacy",
        "${workspaceFolder}Drivers/CMSIS/Device/ST/STM32F1xx/Include",
        "${workspaceFolder}Drivers/CMSIS/Include",
        "${workspaceFolder}Drivers/CMSIS/Include",
        "C:/Program Files (x86)/GNU Tools ARM Embedded/8 2019-q3-update/arm-none-eabi/include"
    ],
    "defines": [
        "USE_HAL_DRIVER ",
        "STM32F103xE ",
        "USE_HAL_DRIVER ",
        "STM32F103xE"
    ],
    "compilerPath": "C:\\Program Files (x86)\\GNU Tools ARM Embedded\\8 2019-q3-update\\bin\\arm-none-eabi-gcc.exe",
    "cStandard": "c11",
    "cppStandard": "c++17",
    "intelliSenseMode": "gcc-x86"
    }
},
"version": 4
}

```

7.2 launch.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
    // 问: https://go.microsoft.com/fwlink/?linkid=830387
    "version": "0.2.0",
    "configurations": [
        {
            "name": "stm32 Debug",
            "cwd": "${workspaceRoot}",
            "executable": "${workspaceFolder}/build/TEST_VSC_BLINK.elf"
        },
        {
            "request": "launch",
            "type": "cortex-debug",
            "serverType": "stutil",
            "device": "STM32F103RE",
            "preLaunchTask": "生成并下载",
            "postDebugTask": "复位设备"
        }
    ]
}

```



```
}  
]  
}
```

7.3 tasks.json

```
{  
  // See https://go.microsoft.com/fwlink/?LinkId=733558  
  // for the documentation about the tasks.json format  
  "version": "2.0.0",  
  "tasks": [  
    {  
      "label": "生成",  
      "type": "shell",  
      "command": "make -j6",  
      "problemMatcher": [],  
      "group": {  
        "kind": "build",  
        "isDefault": true  
      }  
    },  
    {  
      "label": "生成并下载",  
      "type": "shell",  
      "command": "make -j6 && make update",  
      "problemMatcher": []  
    },  
    {  
      "label": "重新生成",  
      "type": "shell",  
      "command": "make clean && make -j6",  
      "problemMatcher": []  
    },  
    {  
      "label": "复位设备",  
      "type": "shell",  
      "command": "STM32_Programmer_CLI -c port=SWD -hardRst",  
      "problemMatcher": []  
    }  
  ]  
}
```

附件（命令行下载程序代码）：

命令行下使用各种软件下载方法

```
1. STM32_Programmer_CLI -c port=SWD -d $(BUILD_DIR)/$(TARGET).hex -
   v -s 0x08000000
2. st-flash write TEST_VSC_BLINK.bin 0x8000000
3. ST-LINK_CLI -c SWD -
   P E:/wo4fisher/Documents/keil_proj__stm32/GCC_VSCODE_TEST/build/GCC
   _VSCODE_TEST.hex -V "after_programming"
```

第一种：STM32_Programmer，第二种 git-hub 开源软件 st-link (release 地址：

<https://github.com/texane/stlink/releases/tag/1.3.0>)，第三种：st link utility

Launch 原版

```
{
    // 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": [
        {
            "name": "stm32 Debug",
            "cwd": "${workspaceRoot}",
            "executable": "${workspaceFolder}/build/TEST_VSC_BLINK.elf"
        },
        {
            "request": "launch",
            "type": "cortex-debug",
            "serverType": "stutil",
            "device": "STM32F103RE",
            "preLaunchTask": "生成并下载",
            "postDebugTask": "复位设备"
        }
    ]
}
```