



# PineconePi ONE

## Getting Start

日期	作者	内容	备注
2019.5.21	xdd	Finish Document	

# Catalog

## **I.About PineconePi ONE**

- 1.What is PineconePi ONE?
- 2.What can PineconePi ONE do ?

## **II, Development Environment Configuration**

- 1.Install KEIL PACK
- 2.Create Project
3. SVD-aided debugging

## **III, Use of Burning Software**

1. Use of ISP Burning Software
2. Use of J-link Burning Software

## **IV, Light a LED**

## **V, External Register View Modification Software Use**

## **VI, HardFault Analysis Software**

## **VII, Try to use MicroPython for development (novice recommendation)**

# I, About PineconePi ONE

## 1.What is the PineconePi ONE?

Pinecone opens your geek heart: equipped with SWM320 processor; Low cost, small size, only 48 mm x31 mm; Full lead extraction; Bread board straight insert; Support MicroPython; ARM - architecture (M4 kernel; Hardware single-period multiplication; 512 KB FLASH, 128 KB SRAM; Peripheral cross mapping; The highest resolution can support 1024\* 768tft-lcd driver; Abundant hardware resources: 1 set of 32-bit watchdog timers, 6 sets of 32-bit general timers, 1 set of 32-bit dedicated pulse width measurement timers, 12-channel 16-bit PWM generator, 2 8-channel 12-bit ADC modules of IMSPS; The onboard Ch330N. Abundant chip data; Multi-docking support.


## 2.What can PineconePi ONE do?

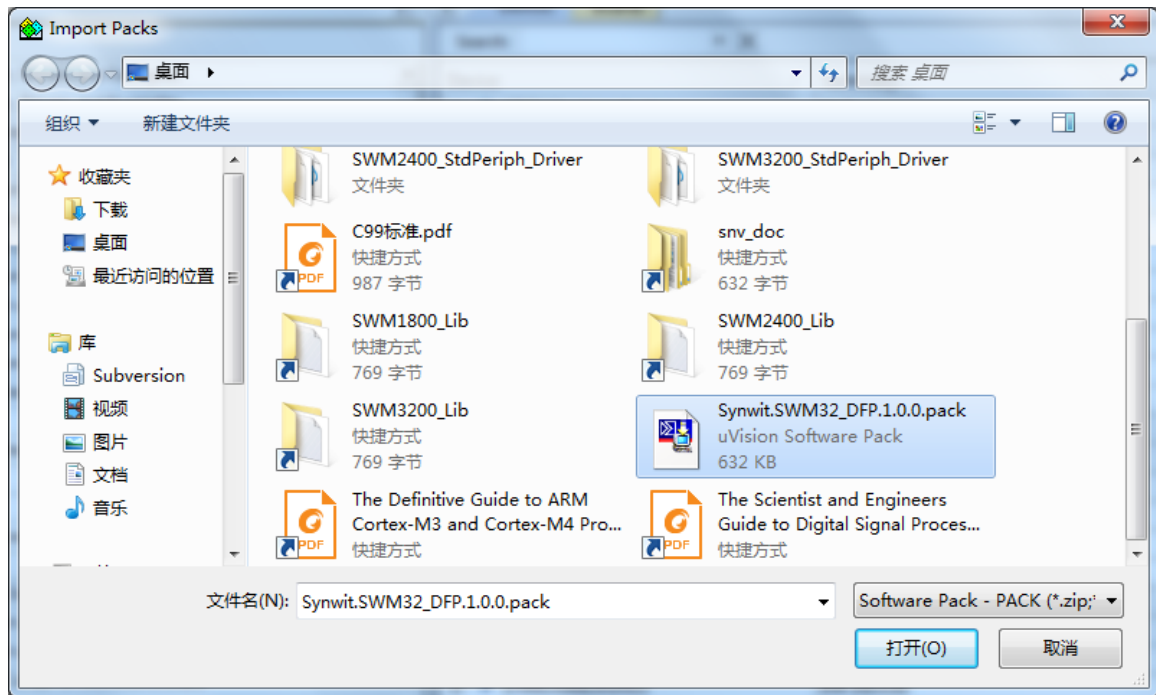
A starter ARM MCU magic development board;  
3D printer control panel;  
Laser engraving machine control panel;  
Writing machine control panel;  
A four-axis aircraft flight control;  
An NES handheld game console;

## II, Development environment configuration

### 1.Install KEIL PACK

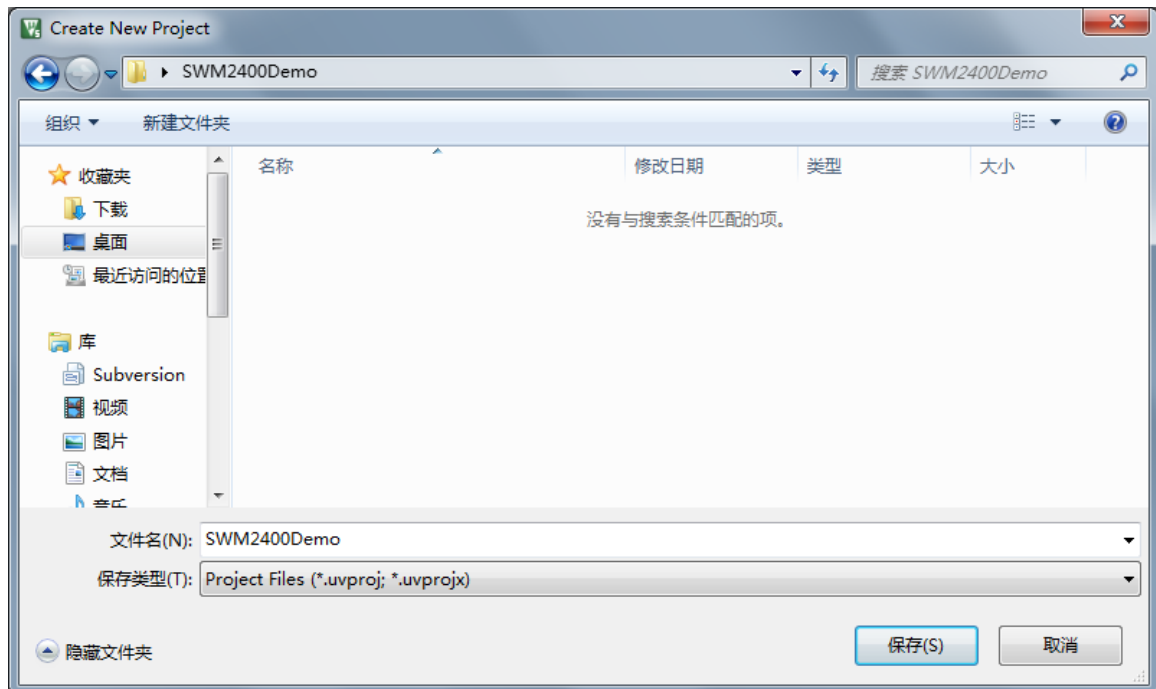
Keil provides support for the chip from the 5.0 version by installing the corresponding MCU package. After installing Synwit.SWM32\_DFP.1.0.0.pack, the new project in Keil 5 can directly select the Synwit SWM series chip, and the software will automatically according to the selected model. Correctly set the Flash/RAM size, program programming algorithm, SVD simulation file, etc.安装 **Synwit.SWM32\_DFP.1.0.0.pack**

Click the toolbar "Pack Installer" button  , In the pop-up window, select Synwit.SWM32\_DFP.1.0.0.pack through the menu "File Import..", click the "Open" button to install



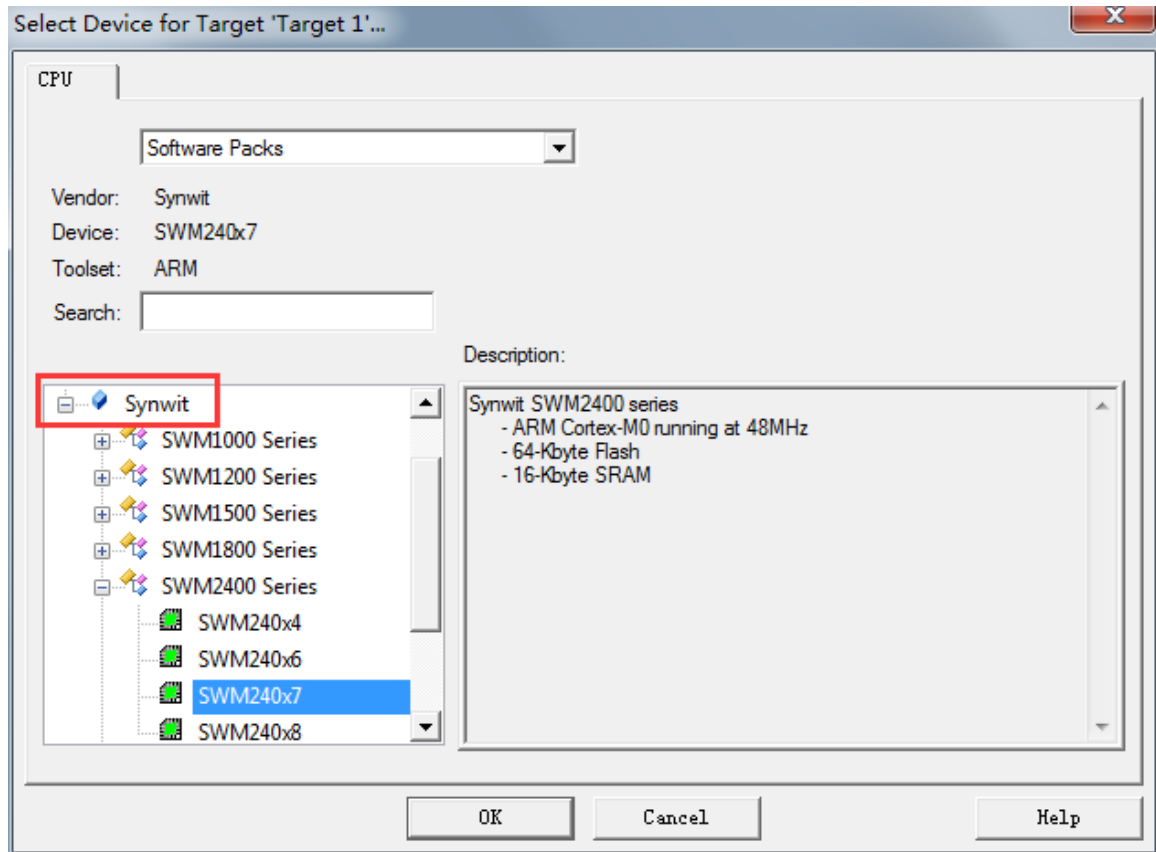
## 2.Create Project

Open the Keil 5 software and select the menu "Project New uVision Project."

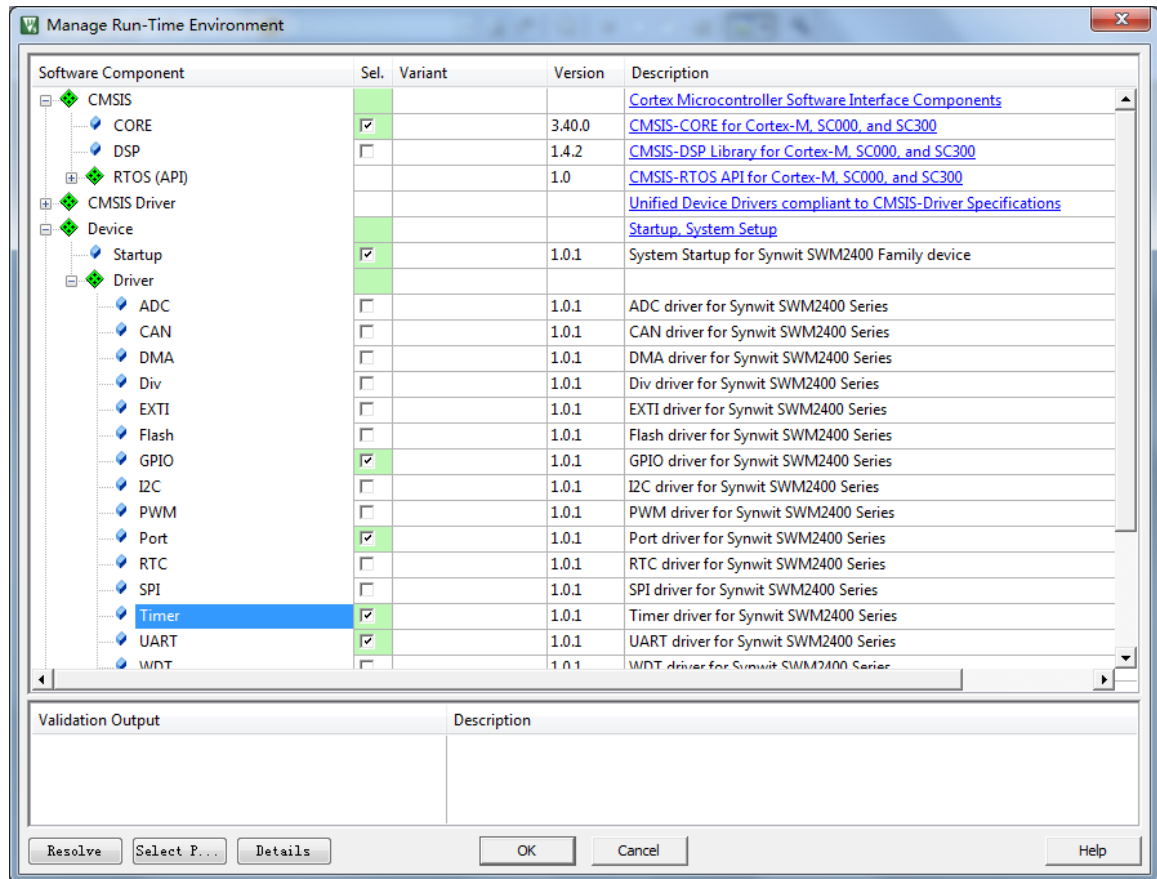


Click "Save" to pop up the device selection window and select

## SWM240x7underSynwit.



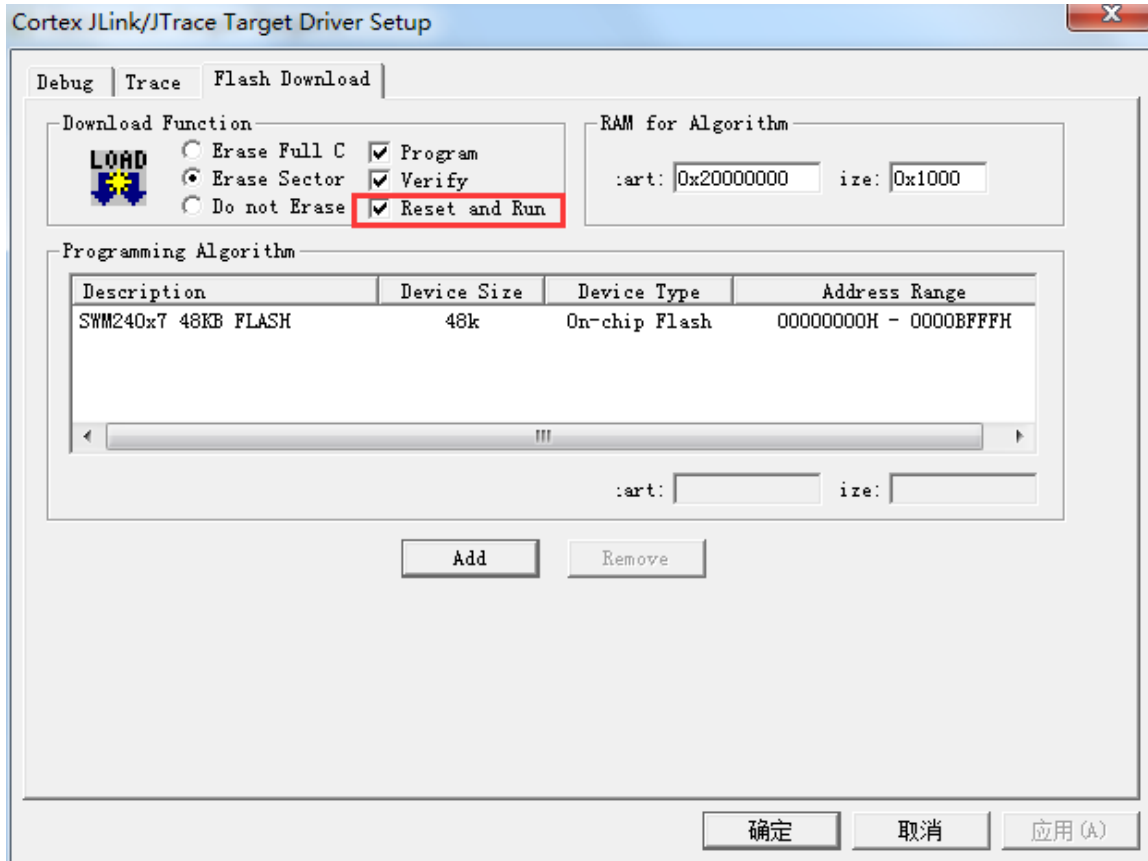
Click "OK" and select the required file in the "Runtime Environment Management" window that pops up.



Among them CORE and Startup, Port is mandatory. Click "OK" to complete project creation and configuration; however, there are still some settings that need to be manually checked.

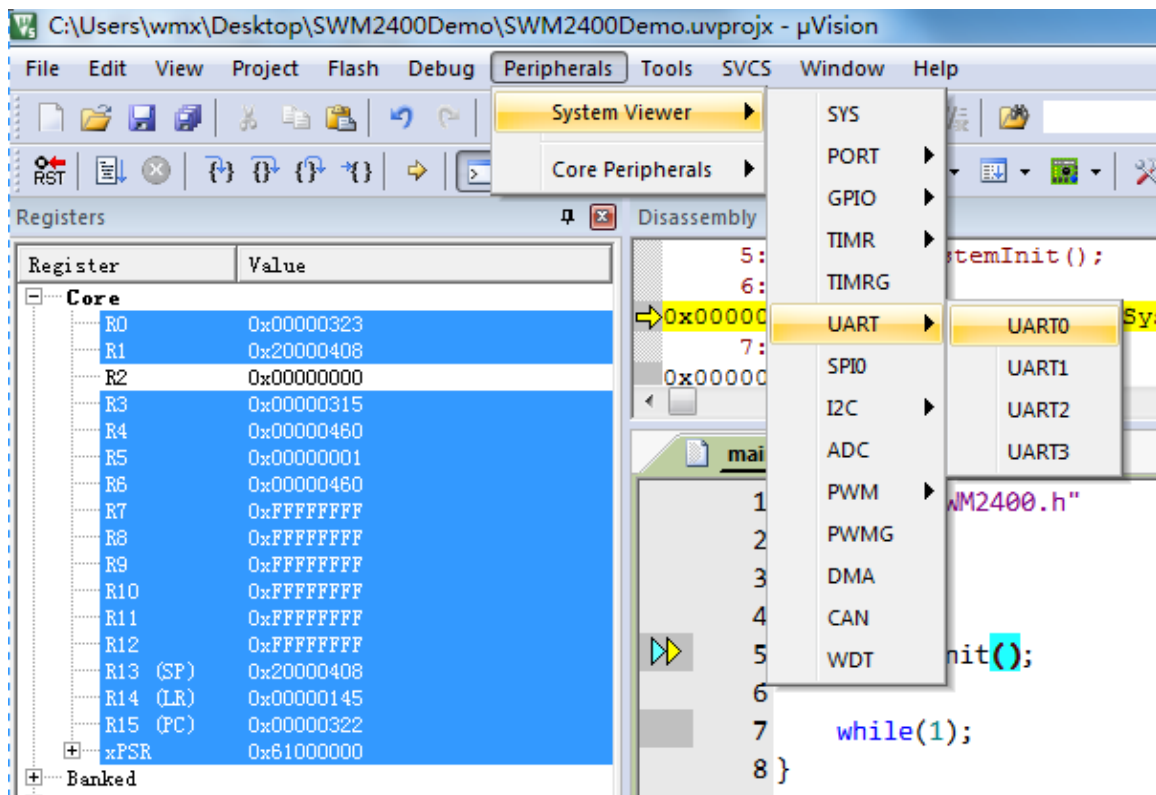


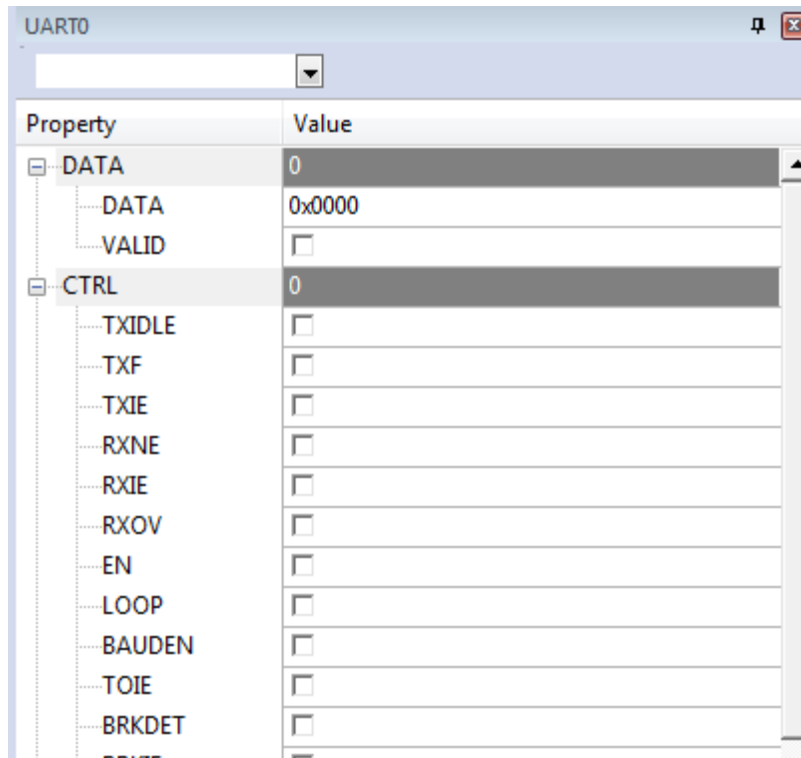
1. Open the "Option" window and manually check "Use MicroLib"
2. Check "Reset and Run"



### 3.SVD assisted debugging

Synwit.SWM32\_DFP.1.0.0.pack contains SVD files, which can display the status of peripherals by bit field during debugging and operate peripherals.





### III, Ues Of Burning software

#### 1. ISP burning software use

Function Description:

1. Automatically burn after handshake: Click one of the keys to automatically perform a one-click download.
2. Production mode: After the download is successful, the handshake operation is automatically performed until the handshake is successful.
- 3, all erase: erase the entire piece of Flash.
4. Sector Erase: Flash required to erase the selected program size.
- 5, one-click download: automatic sector erase and program download operation

other instructions:

1. Try not to perform other operations while loading the bin file.
2. To ensure stability, users are advised to follow the steps in the interface prompts.

Operating procedures:

1. Connect the download interface when the chip is powered off:

SWM180 -- RX-A0 、 TX-A1

SWM240 -- RX-B11、 TX-B12

SWM320 -- RX-A2 、 TX-A3

SWM400 -- RX-A6 、 TX-A5

2. Pull the BOOT pin high.

SWM180 -- BOOT-PB0

SWM240 -- BOOT-PD0

SWM320 -- BOOT-PB0

SWM400 -- BOOT-PB0

3. Power on and perform corresponding operations.

Let's start the burning process below!

**Step 1:** Open "SYNWIT\_ISP\_V2.2.3.exe", select the model

SWM320VET7-50, and click "Confirm".



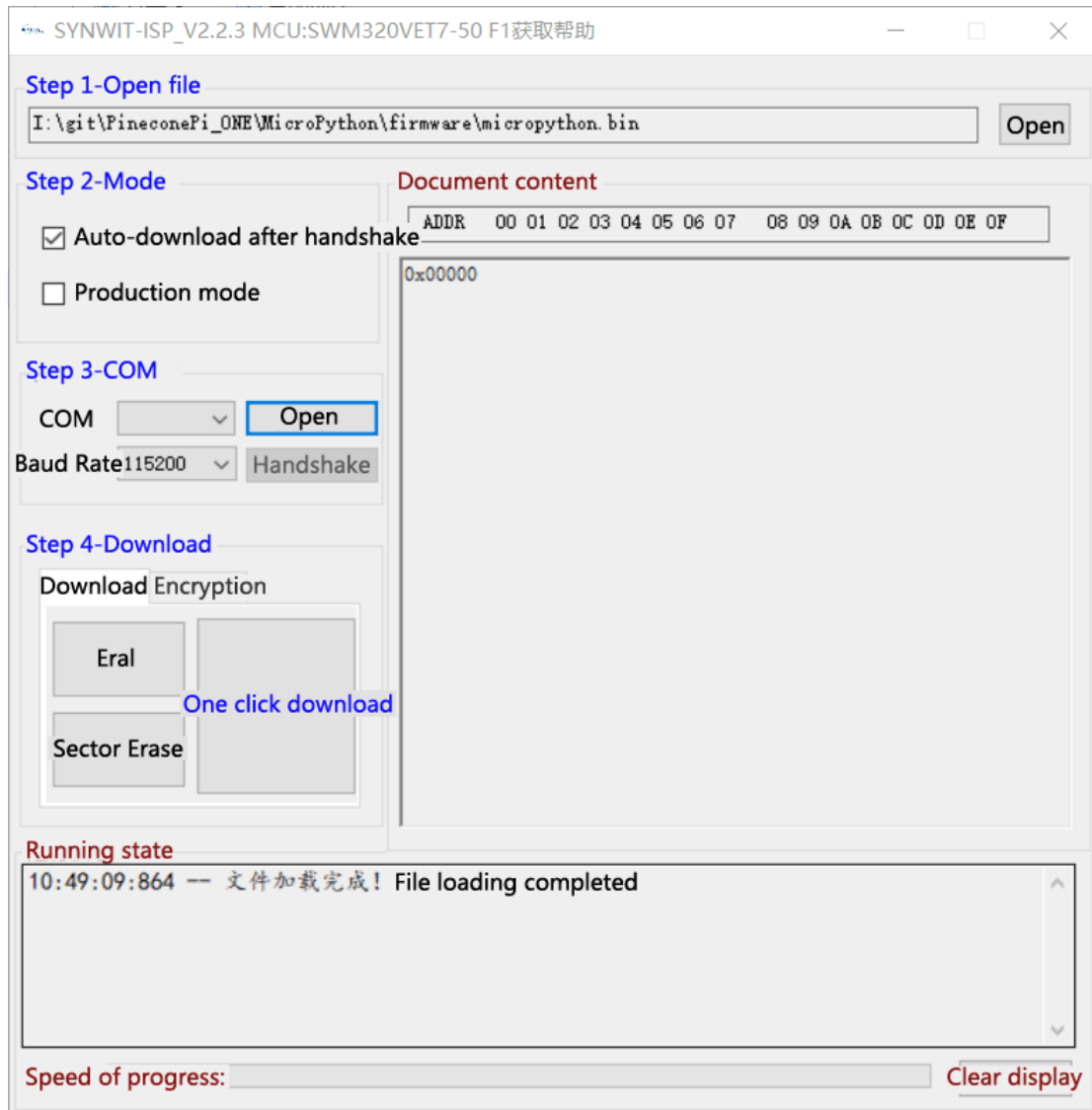
**Step 2:** Select the file and click Open;

**Step 3:** Select the operation mode (in general, check "Automatic download after handshake").

**Step 4:** Serial port operation, select the serial port number and baud rate, and click the "Open Serial Port" button.

**Step 5:** Download the operation and click "One-click download"

**Step 6:** Wait for the download prompt below to increase the "download success" and the progress bar becomes 100%.

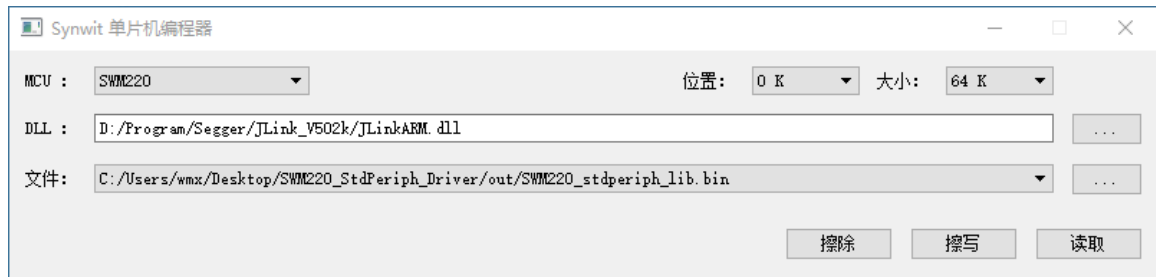


(PS: Because this software does not have an English version, we have made an English version of the interface map for your reference.)

## 2.J-Link burning software use

Burn \*.bin or \*.hex files to the chip specified location via JLINK

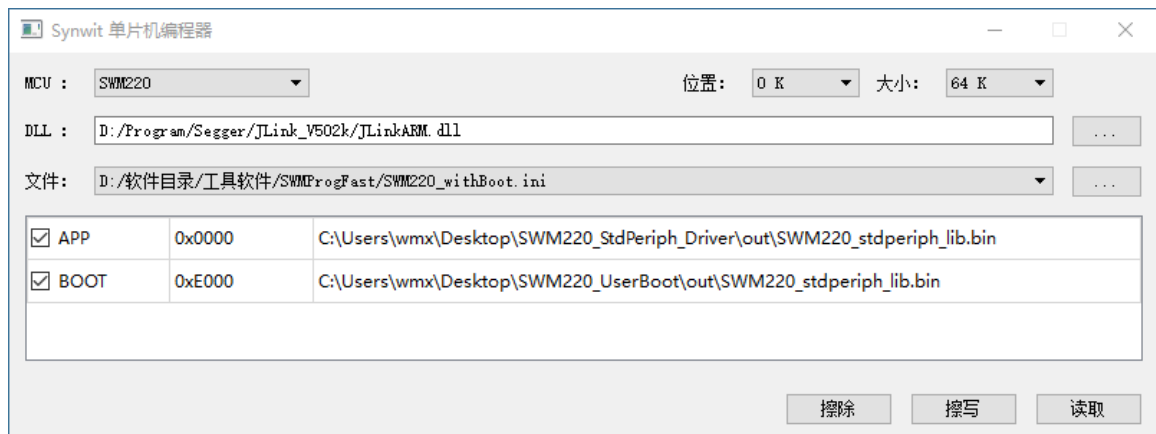
### Single file burning



Location: Start address of erase, erase, and read operations

Size: the size of the erase, read operation, the size of the erase operation is determined by the file size

### Multi-file burning



When the specified file is suffixed with ".ini", the program becomes the multi-file burning interface shown in the figure above.

You can double-click the file path in the table to replace the file and modify the ".ini" file.

## **.ini file format**

[APP]

addr = 0x0000

path

C:\Users\wmx\Desktop\SWM220\_StdPeriph\_Driver\out\SWM220\_stdperiph\_lib.bin

[BOOT]

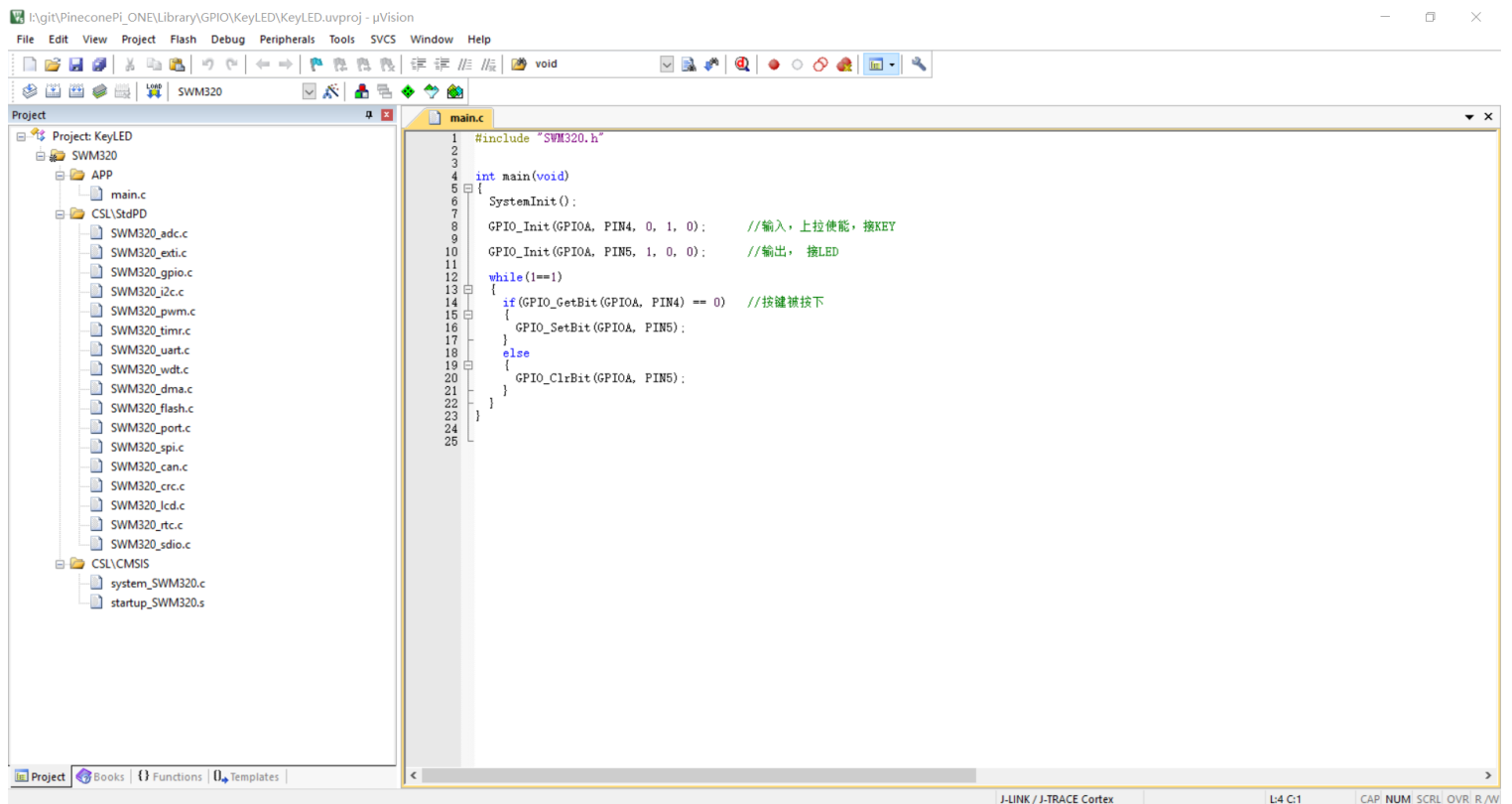
addr = 0xE000

path = C:\Users\wmx\Desktop\SWM220\_UserBoot\out\SWM220\_stdperiph\_lib.bin



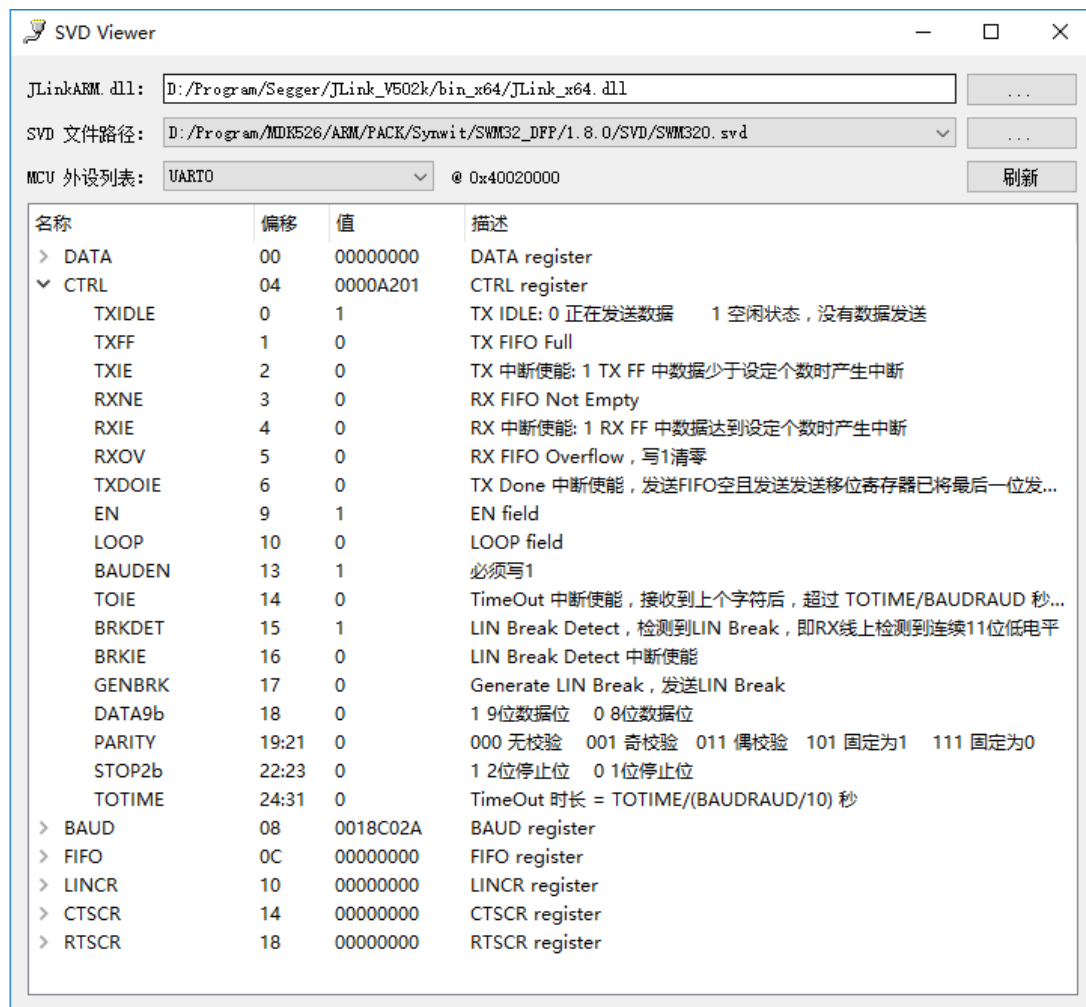
## IV, Light a LED

Turn on the KEIL project under  
"/Library/GPIO/KeyLED/" and complete the GPIO input and  
output pin definition according to the project definition to  
complete the lighting.



## V, Peripheral register view to modify the software usage

Overview: J-Link connection chip, this software reads the chip peripheral register through J-Link, and displays the value of the peripheral register and bit field according to the bit field definition of the SVD file parsing register, and can click the register and bit Real-time modification of the value of the domain to facilitate debugging of the microcontroller peripherals in a non-debug state



The first step is to ensure that J-Link can be connected to the chip. You can use J-Link Commander or Keil to test the connection.

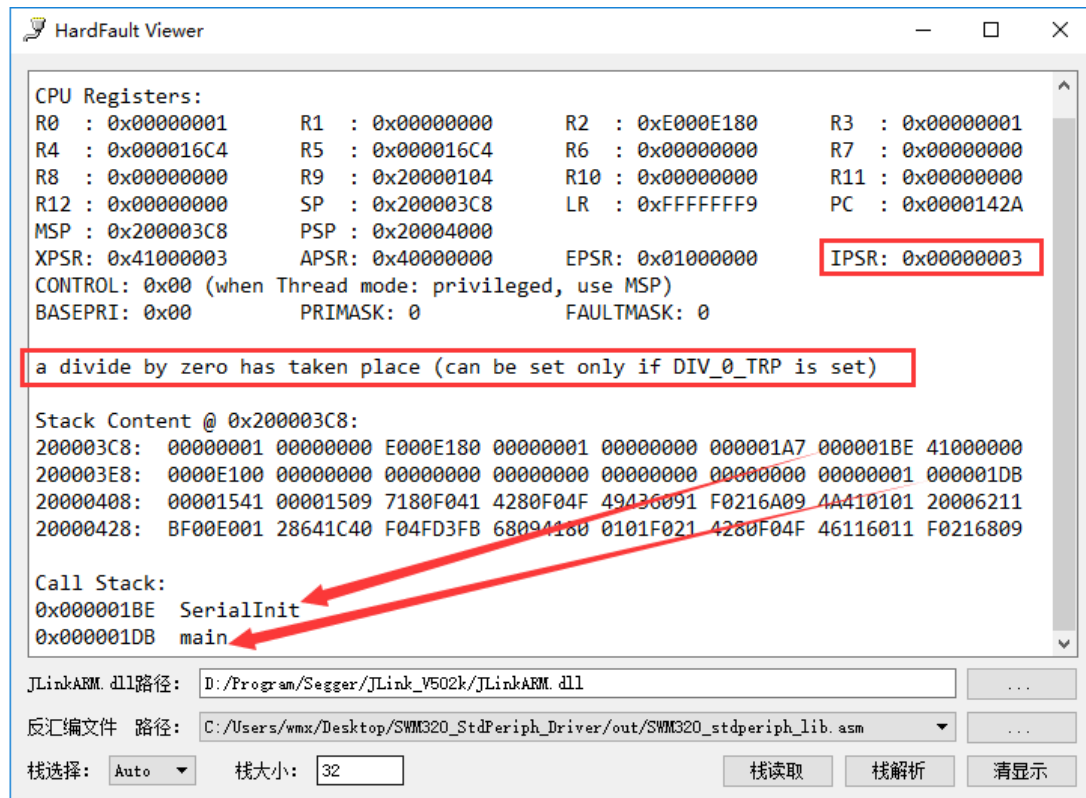
The second step, double-click SVDView.exe to run, set the correct JLink\_x64.dll and device SVD file path

The third step is to select the peripherals you want to view from the "Peripheral List" drop-down box.

The fourth step, click the "Refresh" button to display the current value of the peripheral register.

Step 5, click on the value of the register/bit field, and a dialog box will pop up to modify the value of the register/bit field.

## VI, HardFault analysis software use



After the HardFault occurs, the software can read the contents of the stack from the chip memory through JLink, and then parse the location and execution path of the HardFault according to the contents of the stack and the function call relationship parsed by the disassembly file. For cores with fault cause registers (such as Cortex-M3, M4), the software also analyzes the cause of the fault register and prints the cause of the fault.

In addition, the software will also save the function call relationship parsed from the disassembly file (such as which function is called by function funcA, which function calls function funcA) to the CallStack.txt file in the same directory of the software, the content format as follows:

```

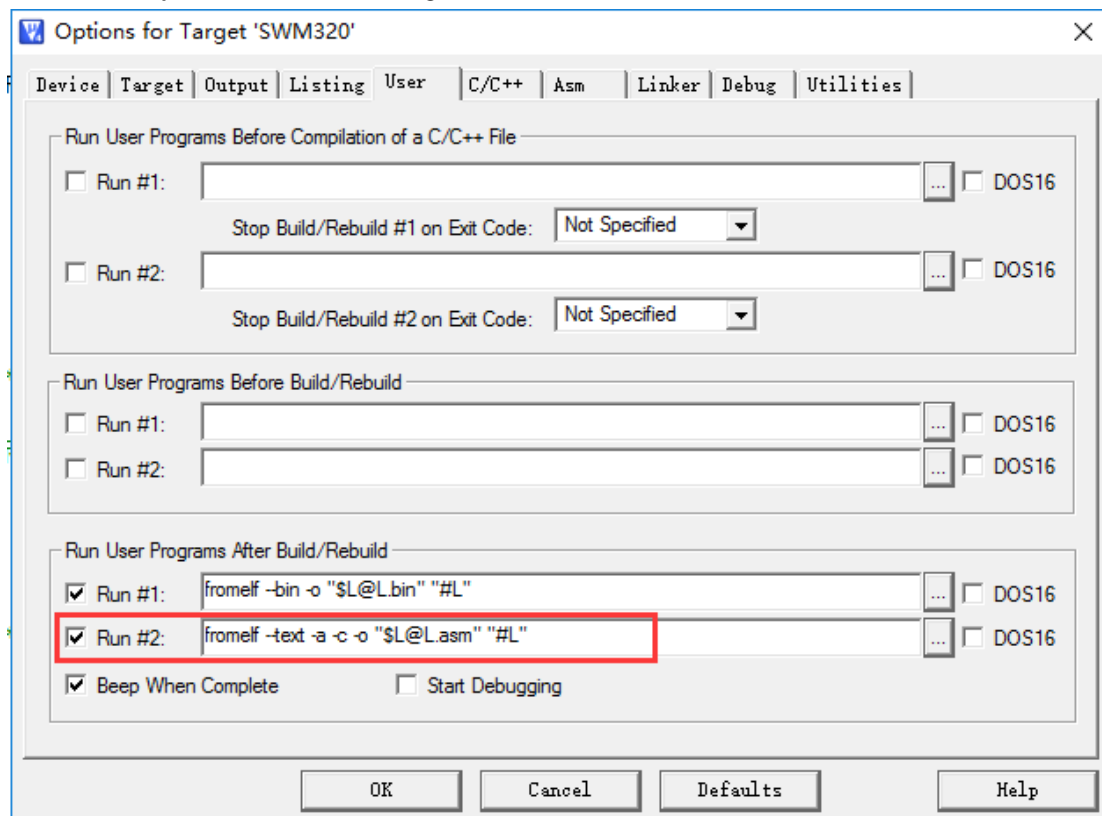
13 ▼ SerialInit                                @ 0x0000015C - 0x000001CE
14     0x00000168 PORT_Init
15     0x00000174 PORT_Init
16     0x0000019C UART_Init
17     0x000001A2 UART_Open
18
19 ▼ main                                        @ 0x000001D0 - 0x00000230
20     0x000001D2 SystemInit
21     0x000001D6 SerialInit
22     0x000001E6 GPIO_Init
23     0x000001F6 GPIO_Init
24     0x000001FE GPIO_ClrBit
25     0x00000206 GPIO_SetBit
26     0x00000210 GPIO_InvBit
27     0x00000218 GPIO_InvBit
28
29 ▼ fputc                                      @ 0x00000232 - 0x0000024E
30     0x0000023C UART_WriteByte
31     0x00000244 UART_IsTXBusy

```

Step 0, please ensure that there is only one infinite loop in HardFault\_Handler(), no extra code.

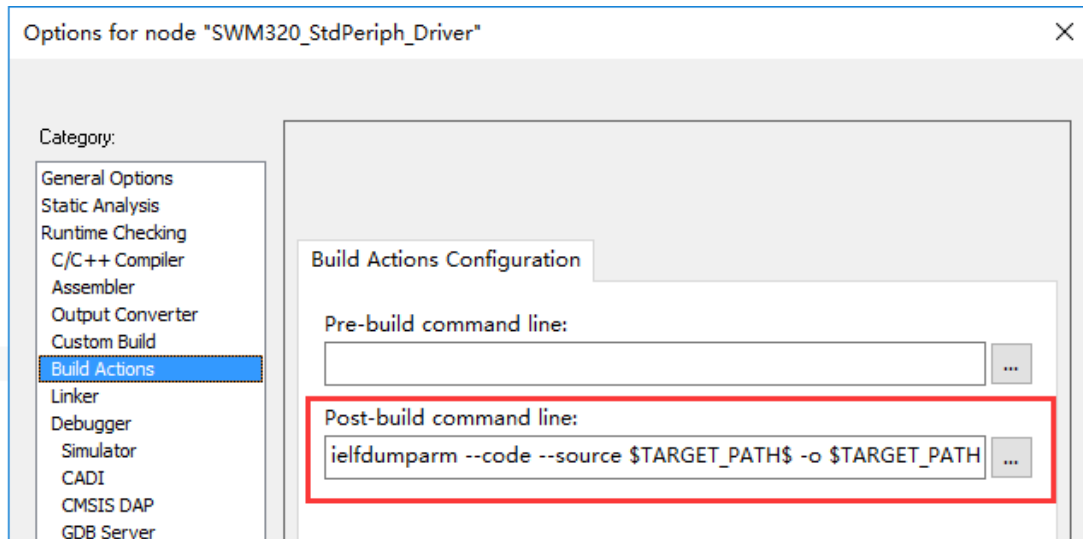
The first step is to ensure that J-Link can be connected to the chip. You can use J-Link Commander or Keil to test the connection.

The second step is to generate a program disassembly file when the program is compiled. The generation method is as follows:



Generate command: fromelf --text -a -c -o "\$L@L.asm" "#L"

IAR generates disassembly settings:



Generate command: `ielfdumparm --code --source $TARGET_PATH$ -o $TARGET_PATH$.dis`.

The third step, double-click HFView.exe to run, set the correct JLinkARM.dll file and the path of the generated disassembly file.

The fourth step, click the "stack read" button to view the value of IPSR. If it is 0x00000003, it means that HardFault has occurred. You can click "stack analysis" to analyze the cause and location of HardFault. If the value of IPSR is not 0x00000003, then the description is HardFault did not occur.

## VII, Try to develop with MicroPython (novice recommended)

See the Github, "/MicroPython/Teaching Document/" folder for MicroPython documentation.