2. STM32 development environment

2.1. Introduction to STM32CubeIDE

STM32CubeIDE is an all-in-one multi-OS development tool, an advanced C/C++ development platform with peripheral configuration, code generation, code compilation and debugging functions for STM32 microcontrollers and microprocessors. It is based on the Eclipse®/CDT™ framework and GCC toolchain for development, and GDB for debugging, selects an STM32 microprocessor and then creates a project and generates initialization code, supports graphical configuration of STM32 clocks and pins, etc. Content, support Windows, Linux, Mac mainstream platforms, very powerful and practical.
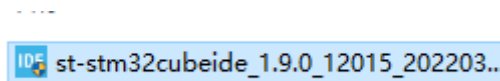
2.2. download the installation package

Open the following link with a computer browser:

https://www.st.com/en/development-tools/stm32cubeide.html

According to the download of the computer system, here we take the installation of version 1.9.0 in the Win10 system as an example, select the latest version to download.

If you have an account on my.st.com, you can log in directly and download the software. If you don't want to log in now, simply provide your name and email address in the form below to download the software.
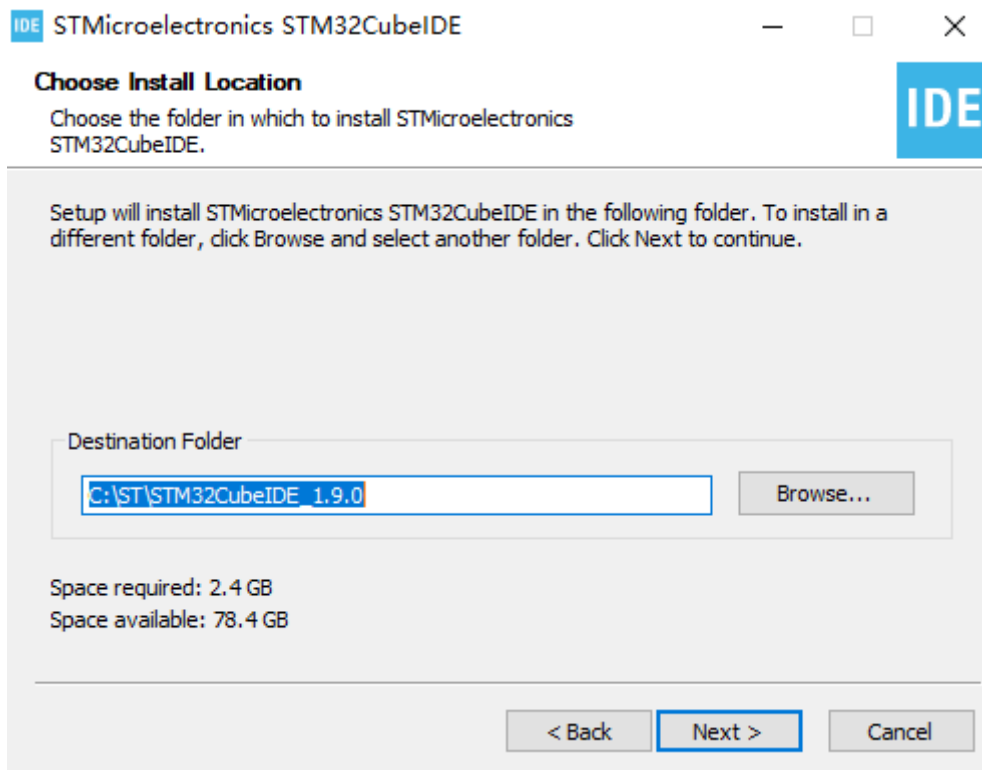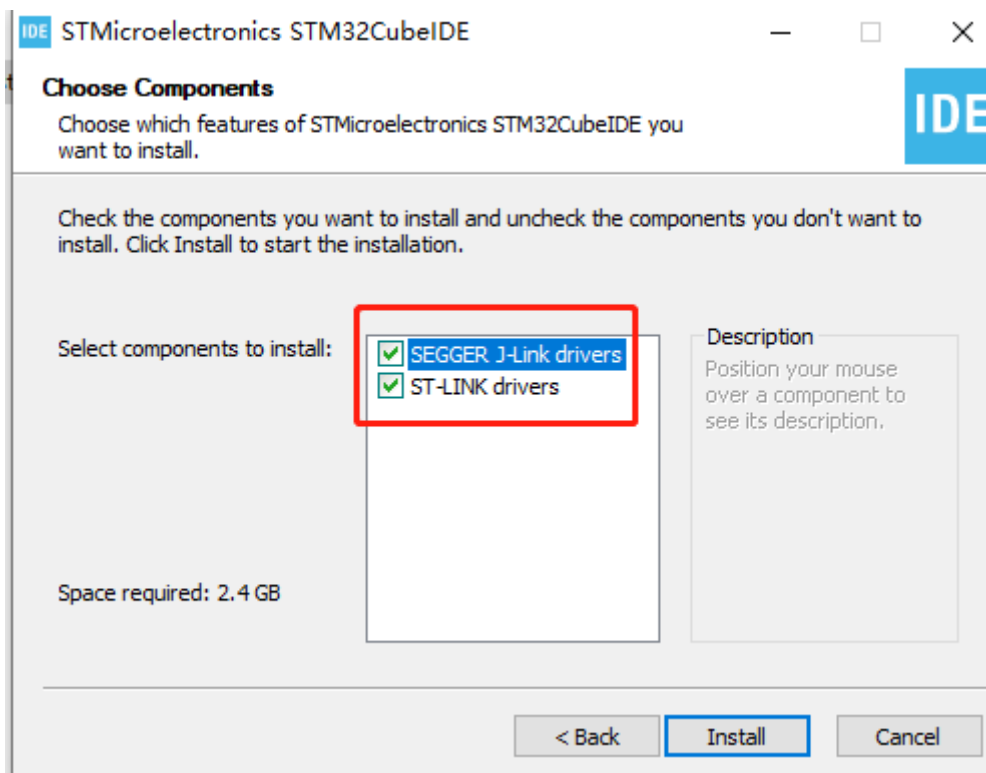

2.3. start the installation

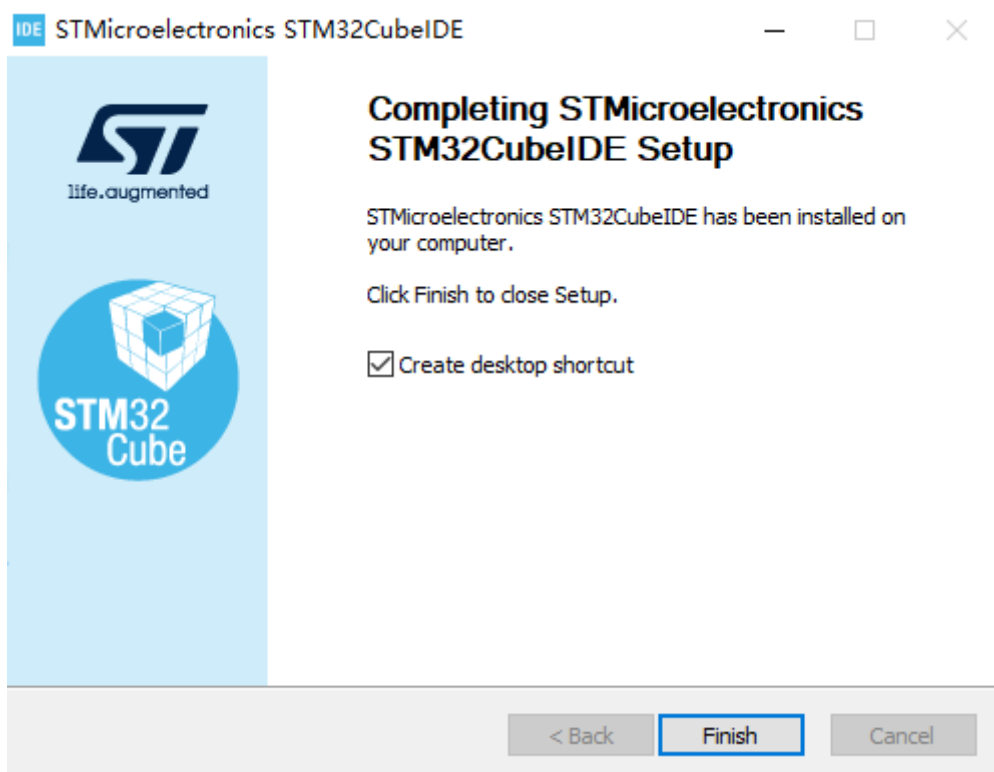Double-click to open the installation package. Then just follow the tutorial.
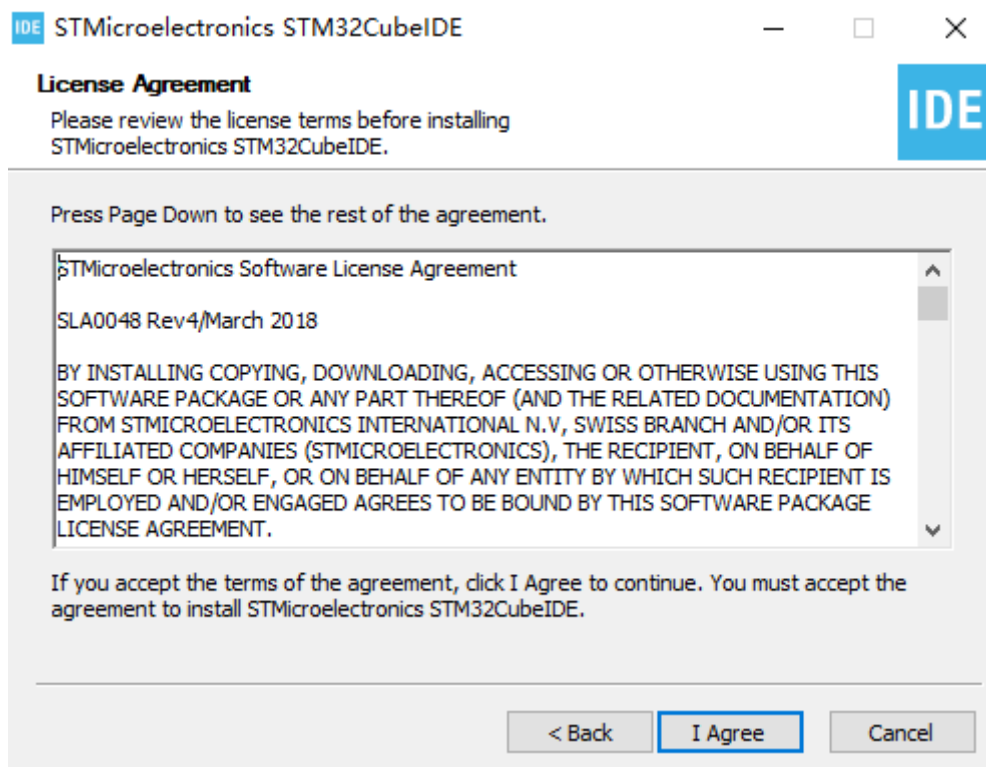
The installation path can be modified according to the actual situation, be careful not to have Chinese.

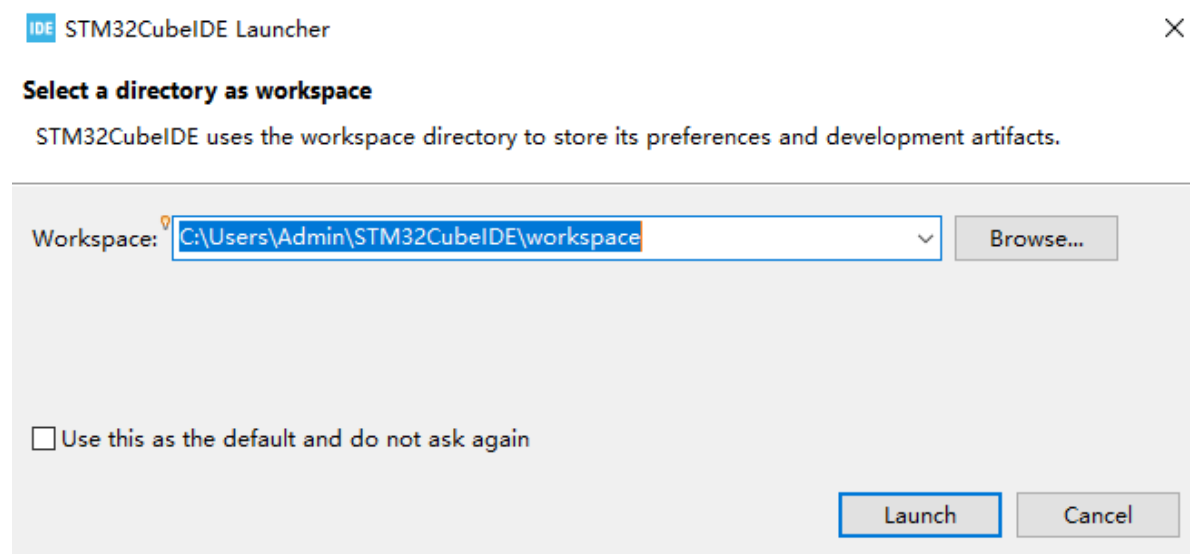Check the driver, and then click Install to install.

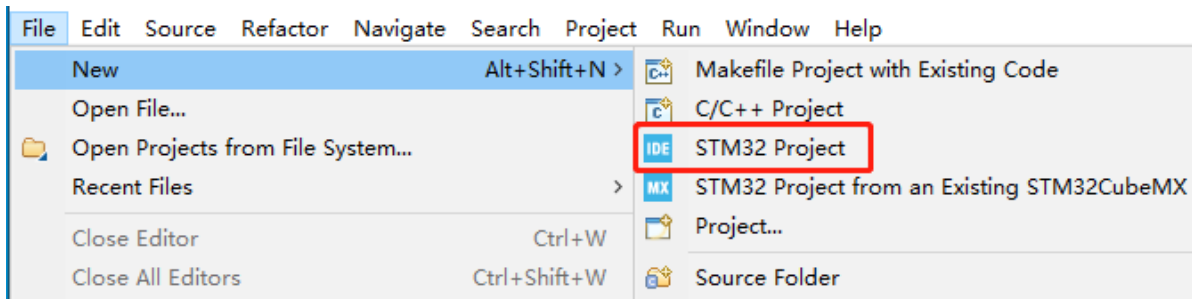Then wait for the installation to complete.

2.4. New construction

1. Double-click the shortcut on the desktop to open STM32CubeIDE. You need to select a workspace. The save path can be selected according to the actual path (do not bring Chinese).
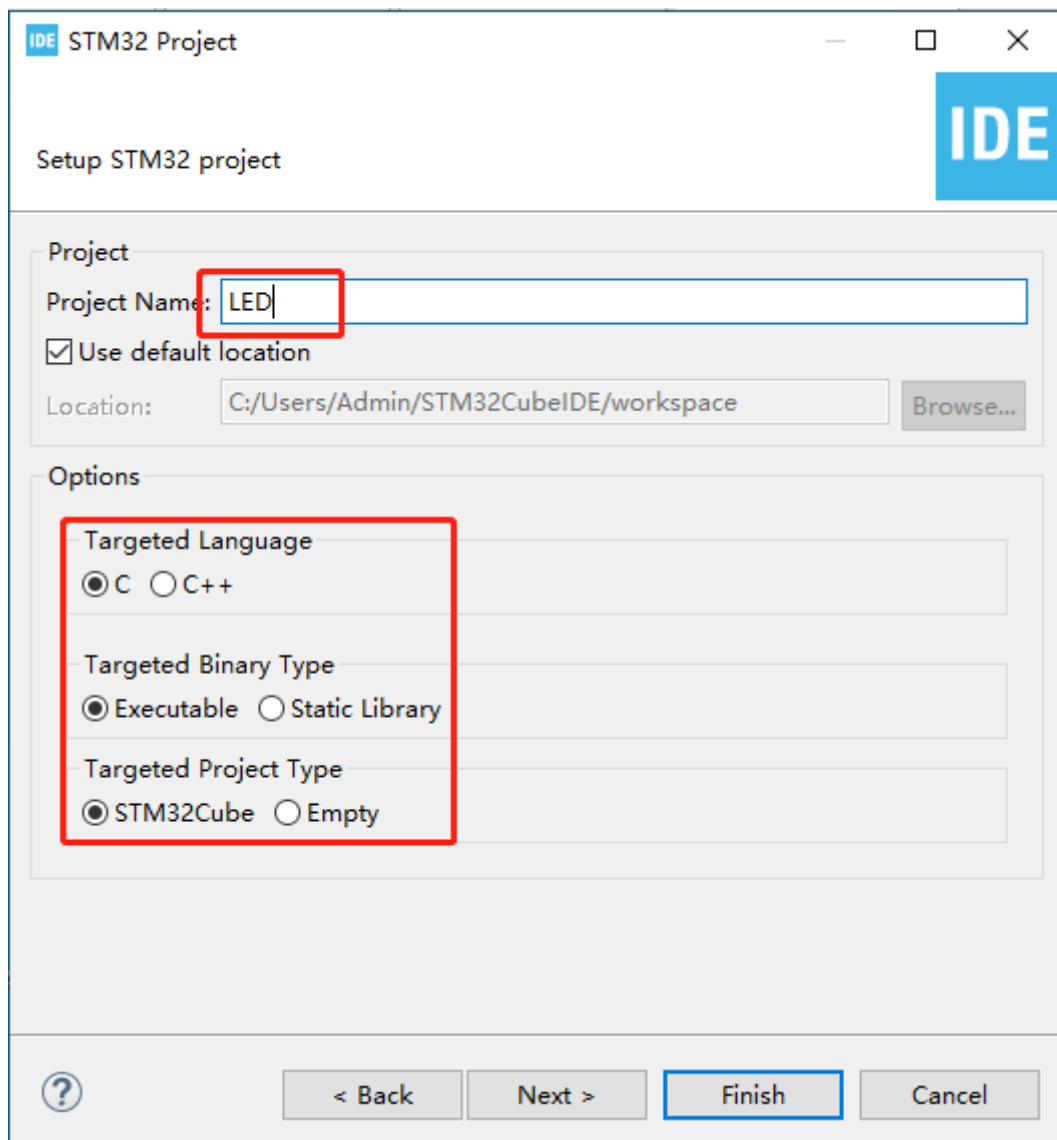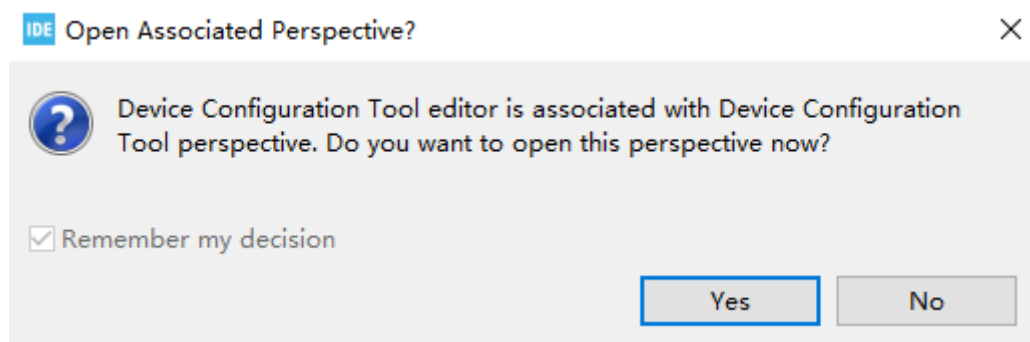

2. Click File->New->STM32 Project.

3. Search and select the chip STM32F103RCT6, then click Next in the lower right corner to enter the next step.
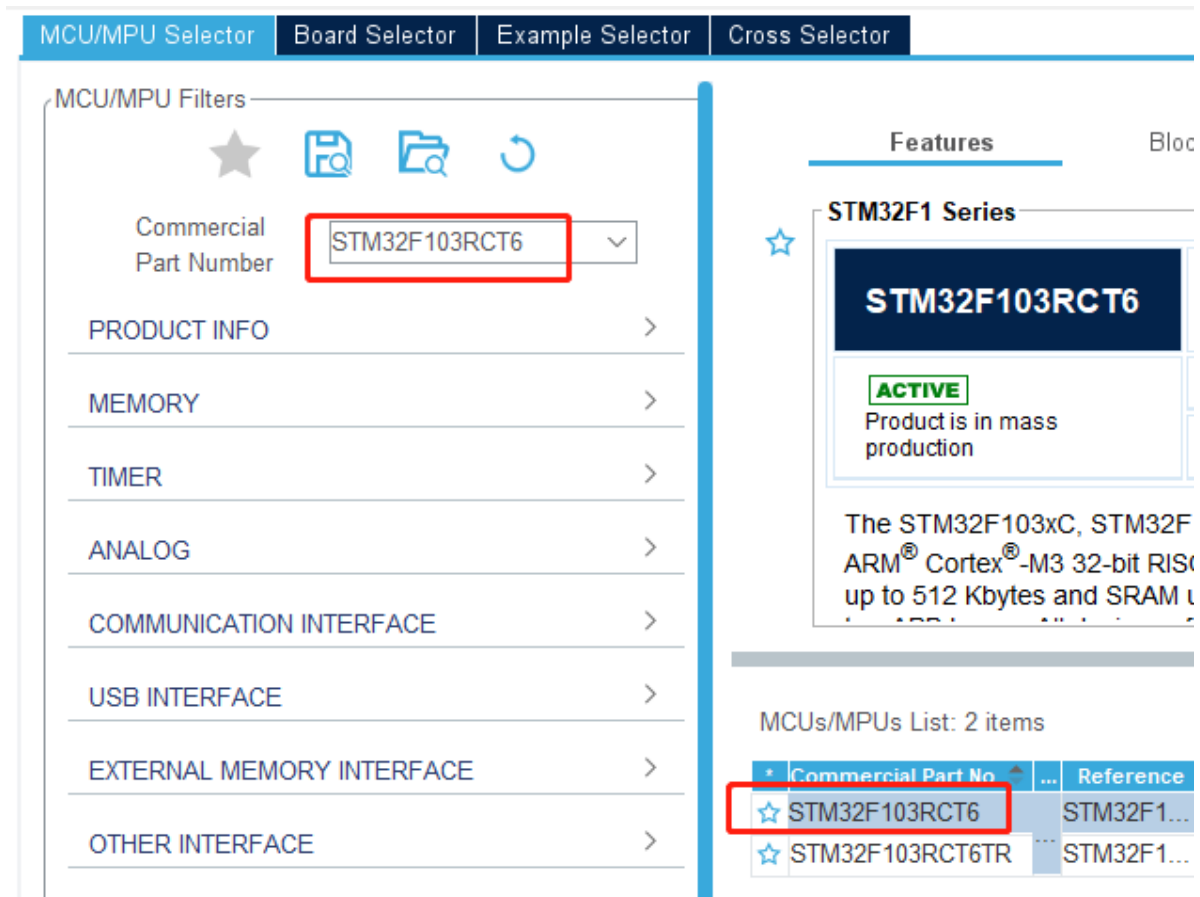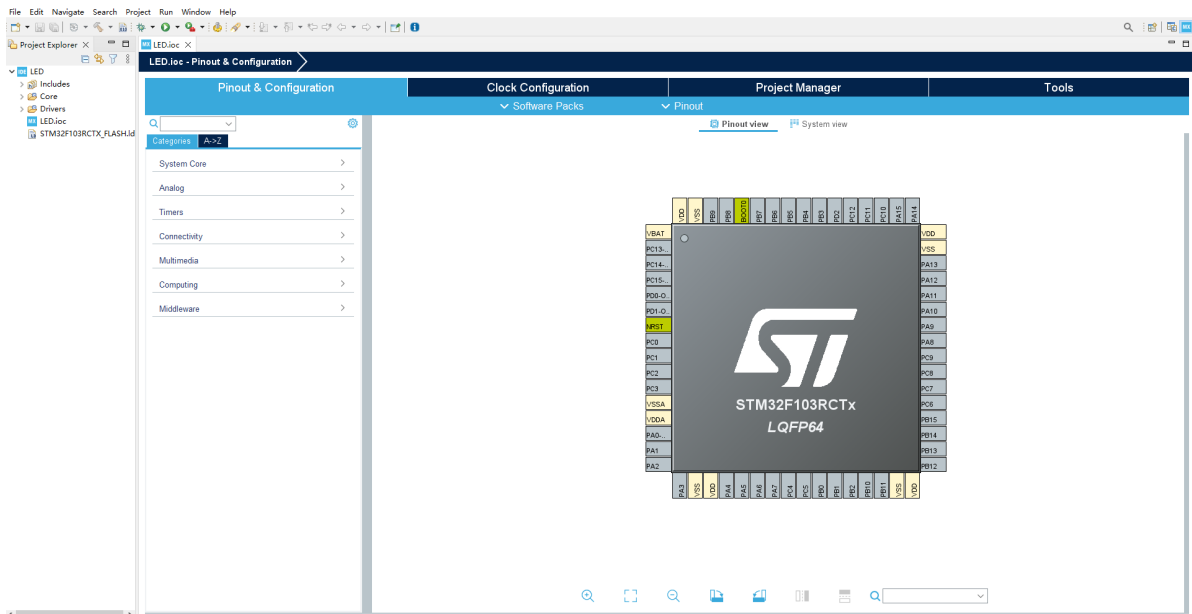

4. Enter the project name, take LED as an example here, other parameters can be defaulted.

5. Click Yes, and the graphical content will be loaded.

After completion, it will look like the following picture:

2.5. pin configuration

1. First need Debug information, click SYS->Debug under Pinout&Configuration and select Serial Wire.
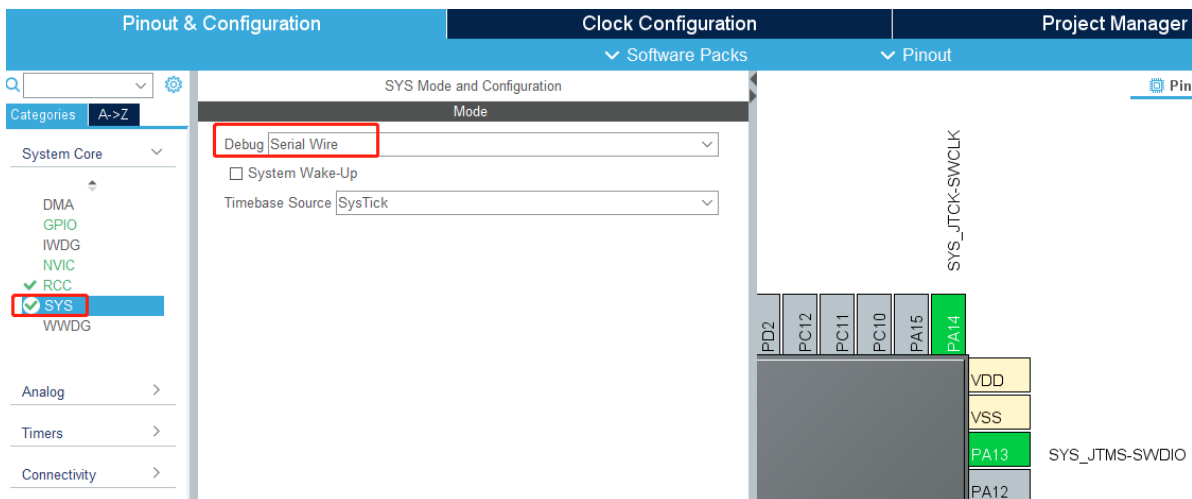
2. Modify the system clock of STM32. According to the schematic diagram, the external crystal oscillator is 8M frequency.
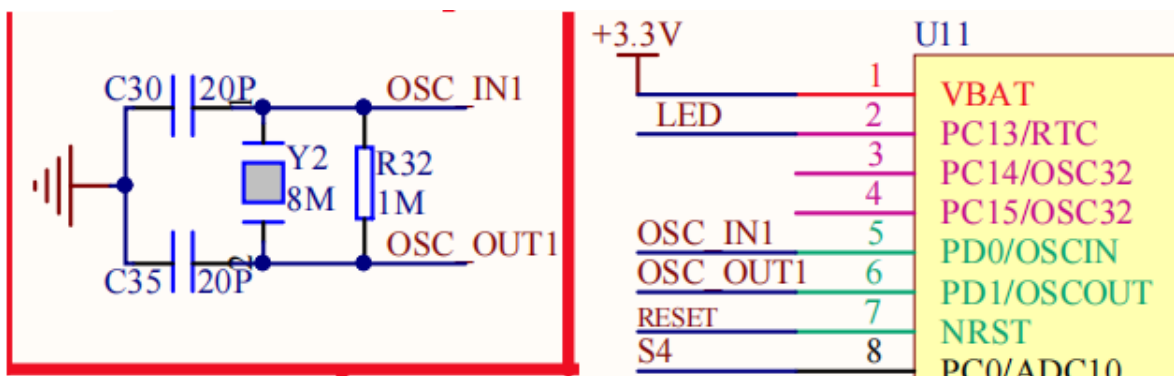
Select RCC->HSE in Pinout&Configuration and select Crystal/Ceramic Resonator. The HSE is the external clock, and the LSE is the internal clock. Using the external clock can be more stable and efficient than the internal clock.
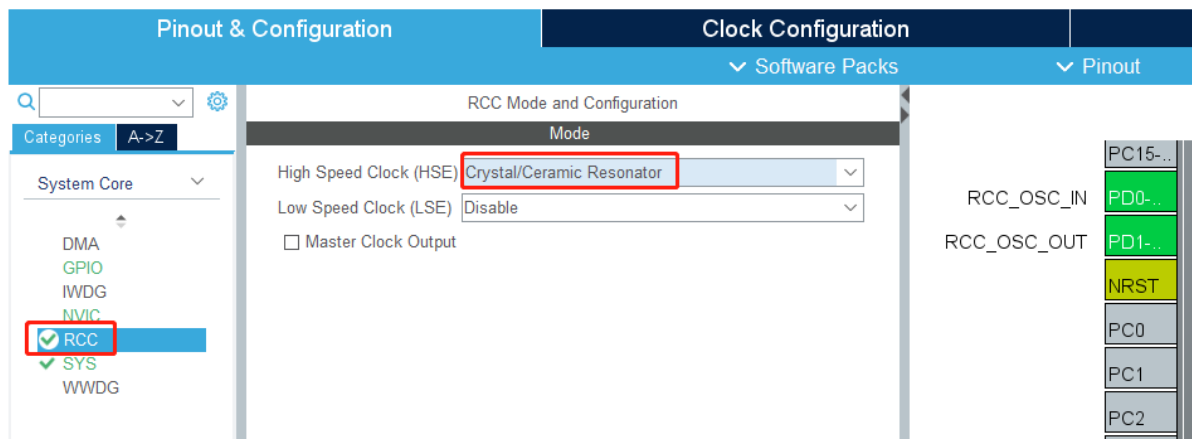
Switch to Clock Configuration to modify the frequency of HCLK to 72, and press Enter to confirm.

Clock Wizard ✕

? No solution found using the current selected sources.
Do you want to use other sources?

OK Cancel
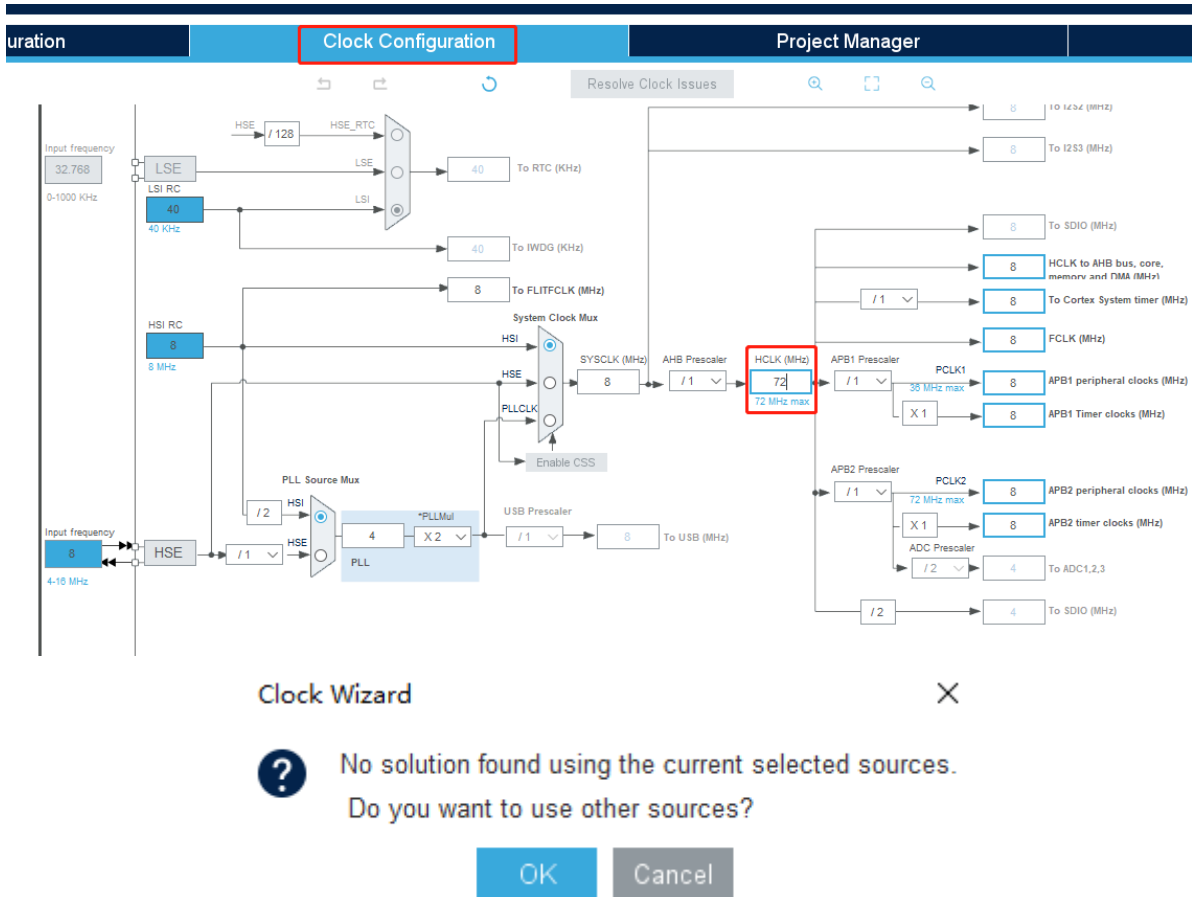
3. Increase the LED pin configuration. It can be seen from the schematic diagram that the LED is connected to the PC13 pin.
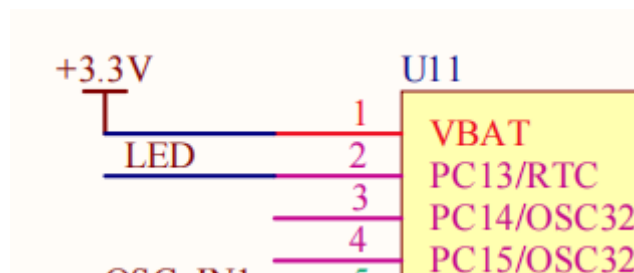
Set the PC13 pin to GPIO_Output, and modify the Label to LED for convenience.
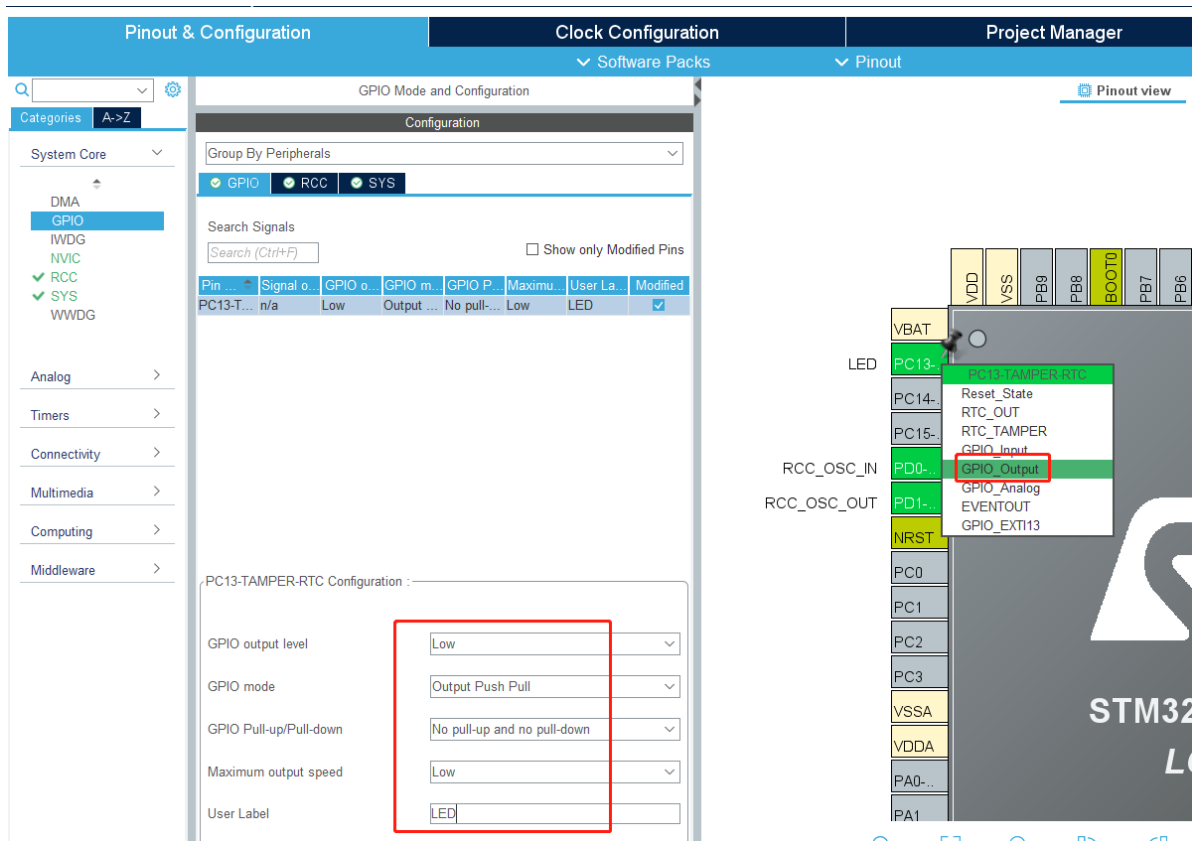
Then press Ctrl+S to save, tick Remember my decision, and click Yes. This will automatically generate the code every time you save.
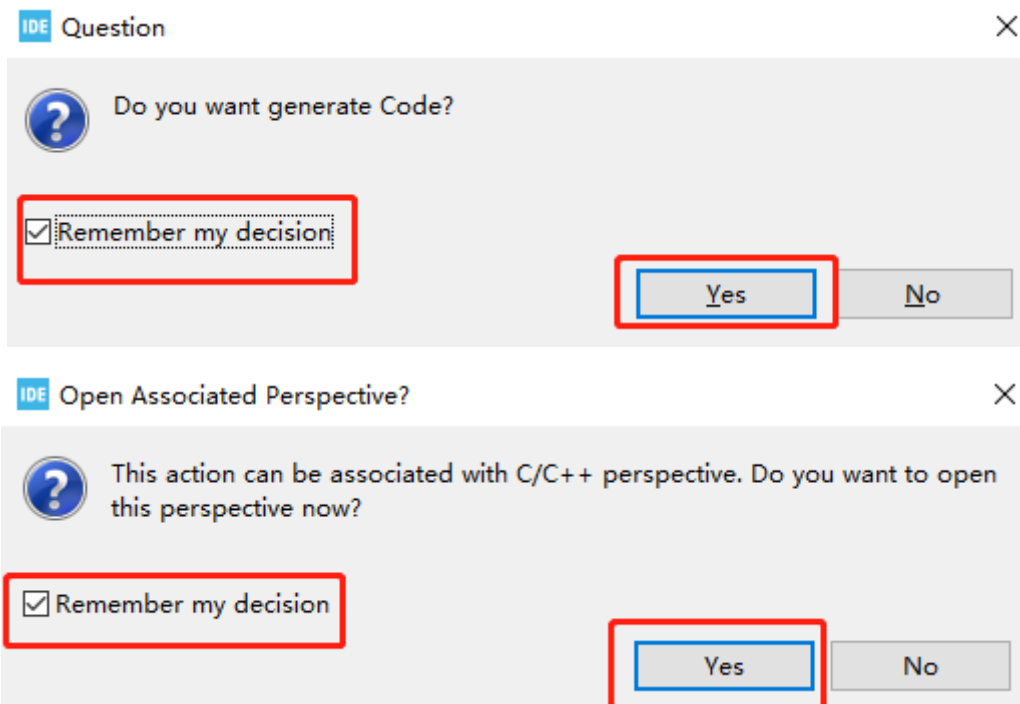
2.6. write code

1. Since the initialization code of the system has been generated by the graphical configuration in the previous step, we only need to add the functions to be implemented.

Find the main function in the main.c file, and add the content to control the LED light under while(1). The function is that the LED light flashes every 200 milliseconds. Press Ctrl+S to save the code.

```
/* Infinite loop */
/* USER CODE BEGIN WHILE */
while (1)
{
    HAL_GPIO_WritePin(LED_GPIO_Port, LED_Pin, SET);
    HAL_Delay(200);
    HAL_GPIO_WritePin(LED_GPIO_Port, LED_Pin, RESET);
    HAL_Delay(200);
    /* USER CODE END WHILE */

    /* USER CODE BEGIN 3 */
}
/* USER CODE END 3 */
```

Note: The code content needs to be added between USER CODE BEGIN and USER CODE END. Otherwise, the content of the code will be overwritten after the code is generated by the graphical tool next time, and the content added between USER CODE BEGIN and USER CODE END will not be overwritten. Don't write Chinese comments in it, there may be garbled characters.

2.7. Compiler

1. Add the function of generating HEX files.

Click Project->Properties->C/C++ Build->Settings->MCU Post build outputs in turn, and then check the box before Convert to Intel Hex file(-O ihex), as shown in the following figure.
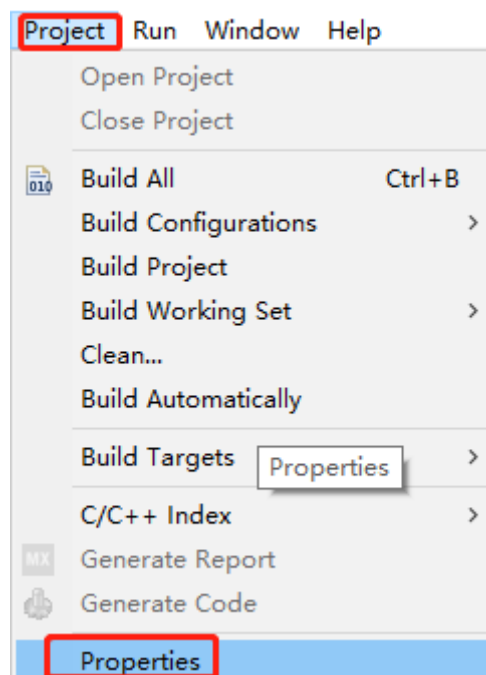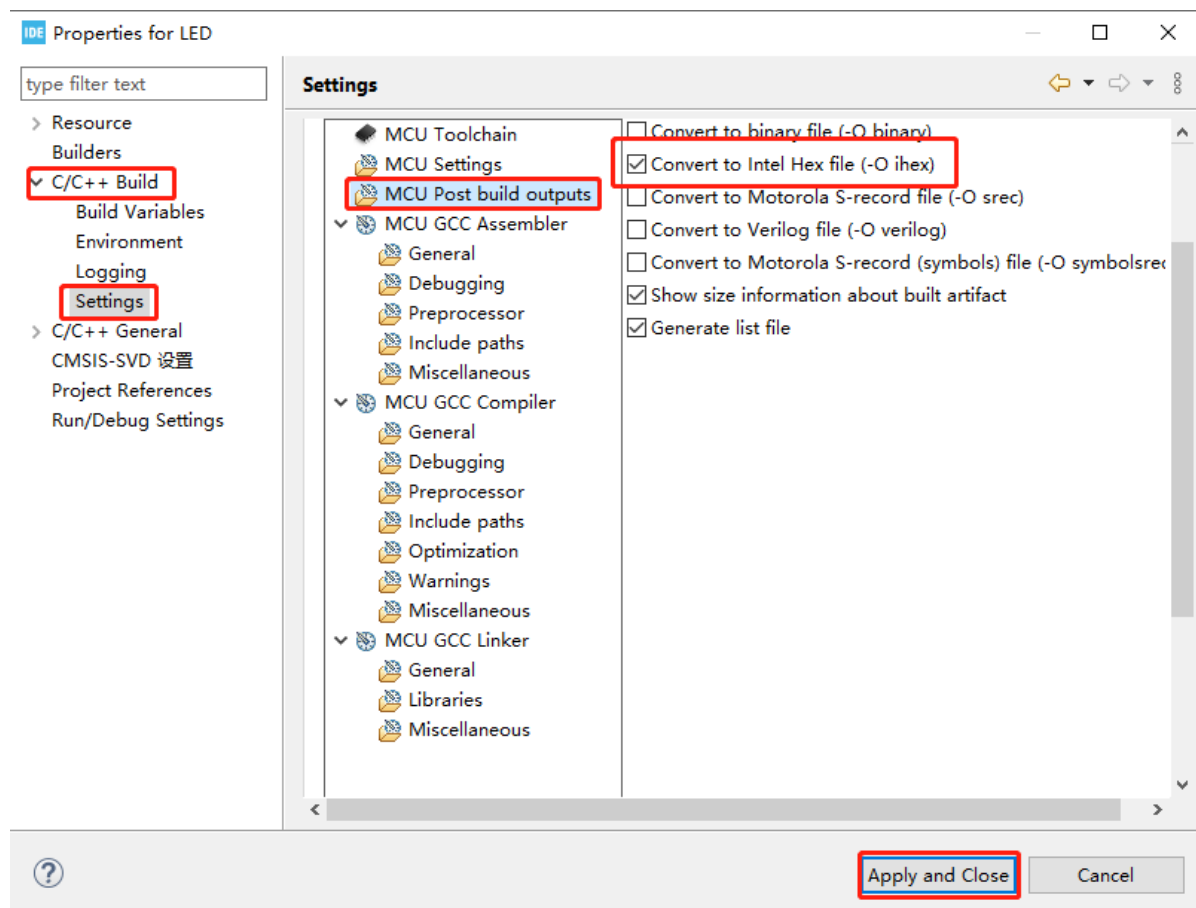
2. Click the hammer in the toolbar to start compiling the project.

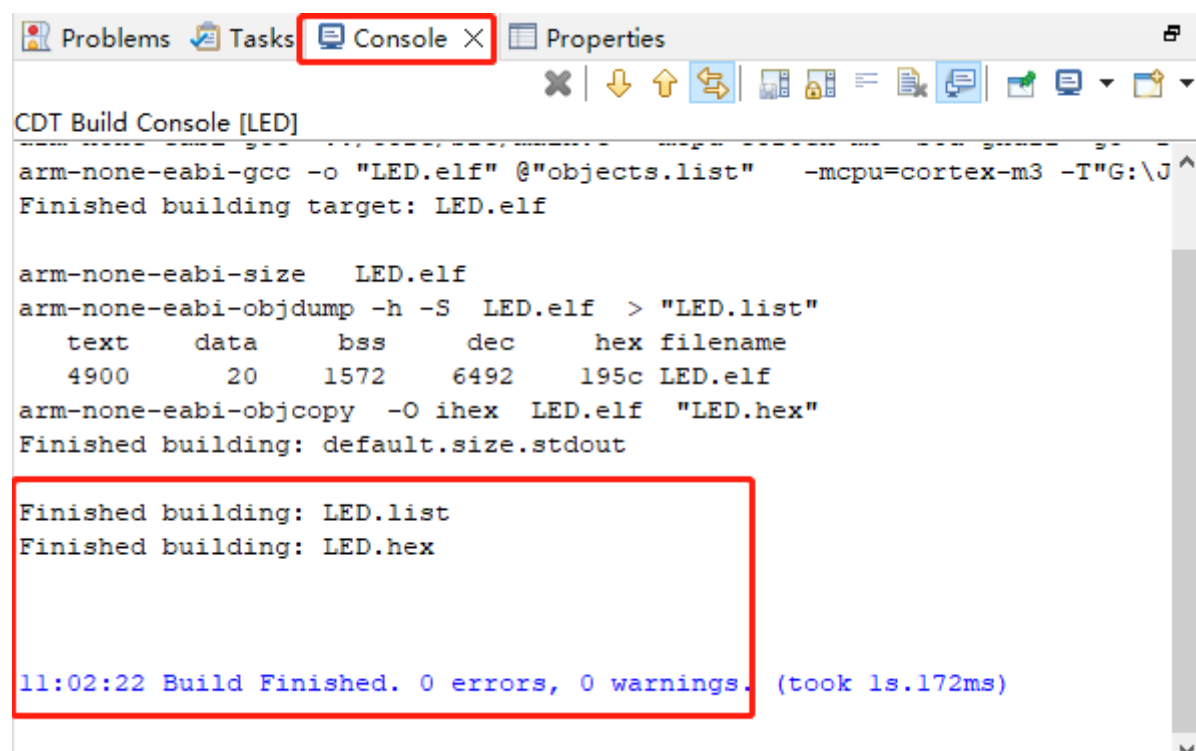STM32CubeIDE will pop up the Console console and see 0 errors in compilation, and 0 warnings means the compilation is successful. As shown in the figure below, the file name generated by the project is LED.hex, and this file is saved in the Debug folder of the project directory.

2.8. Burn program

1. Install the CH340 driver

Since the USB communication of the Rosmaster expansion board uses the CH340 chip, the driver of the CH340 chip needs to be installed. If the computer has already installed the CH340 driver, there is no need to install it again.

Unzip the [Uart drive (CH340).zip] in the course materials, double-click to open the CH341SER.EXE program
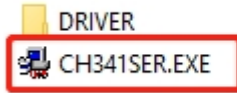
Click Install. After the installation is complete, you will be prompted that the installation was successful.
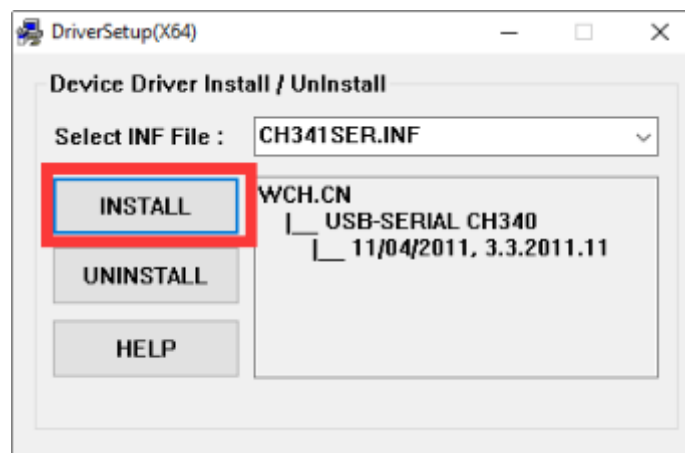
2. Download the burning software

This time burning Rosmaster expansion board microcontroller firmware requires mcuisp (or flymcu) burning software, please go to http://www.mcuisp.com website to download mcuisp (or flymcu) burning software; you can also use the data provided directly mcuisp software.

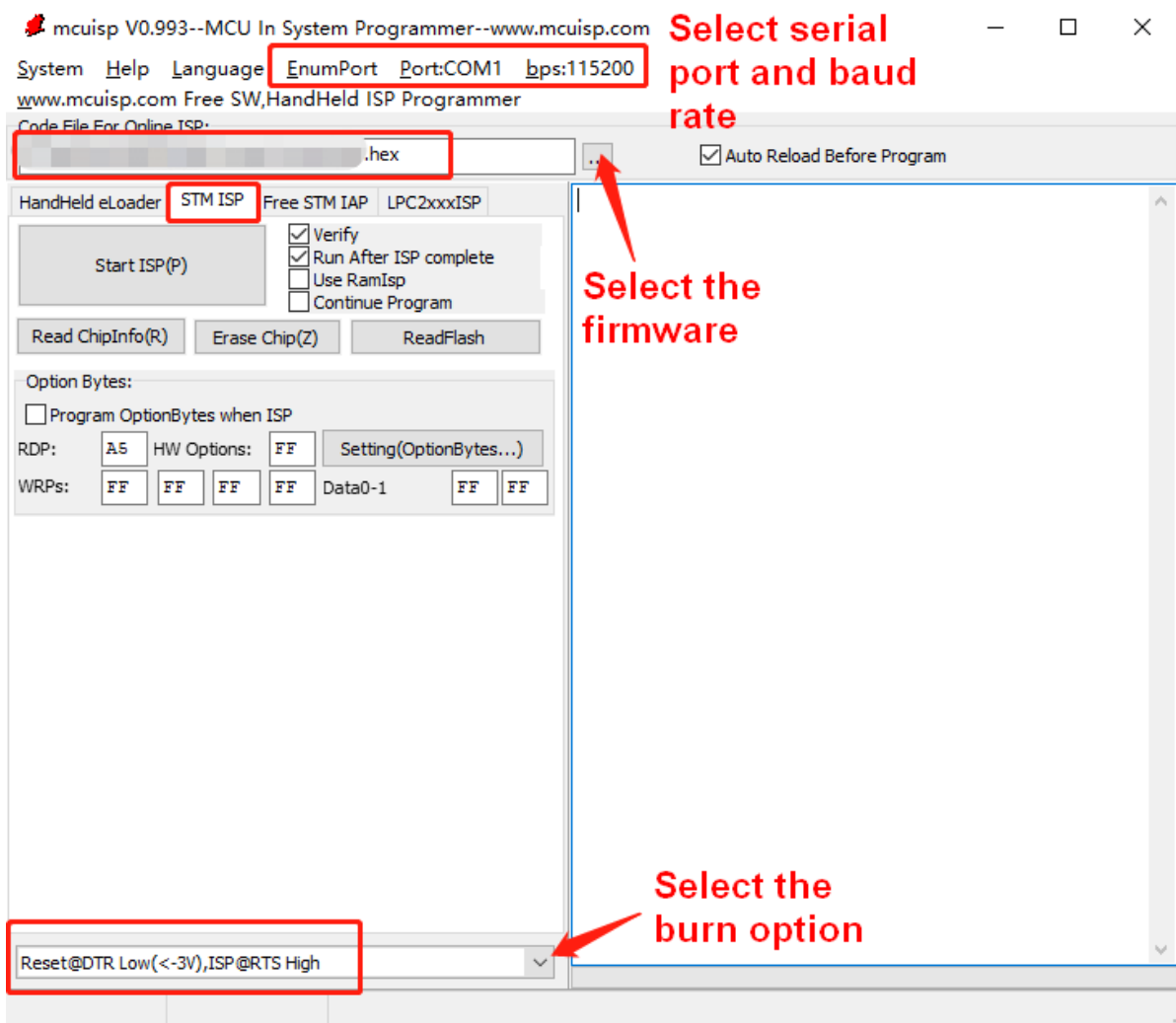The mcuisp software is a green version software, which does not need to be installed. Double-click to open it to use.

3. Connect the device

Before connecting the Rosmaster to the computer, please unplug the Micro USB data cable and power cable connecting the expansion board to the Jetson Nano.

Insert one end of the USB data cable into the USB port of the computer, and the other end into the Micro USB port of the Rosmaster expansion board.

The other end is connected to a computer
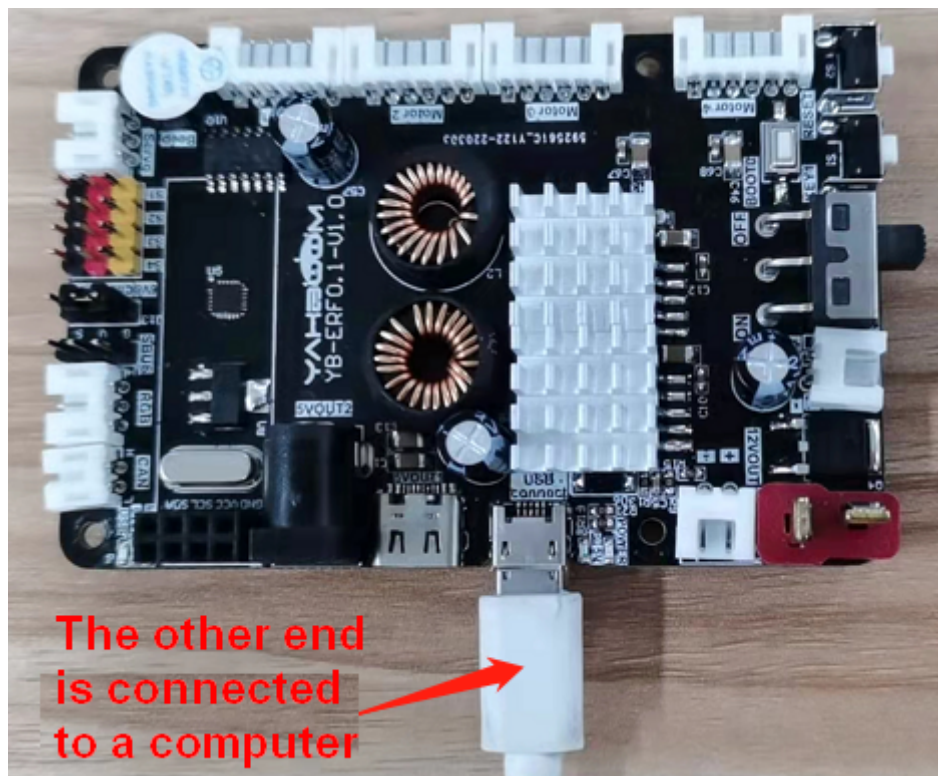
4. Configure the burning software

Select serial port and baud rate

Select the firmware

Select the burn option

When searching for a serial port, if there are multiple serial port numbers, it is not confirmed which one is the Rosmaster. Solution 1: Unplug other USB ports and search again; Solution 2: First unplug the Rosmaster USB data cable, click Search Serial Port, write down the searched serial port number, insert the Rosmaster USB data cable, search the serial port again, and compare before

and after Twice, the newly added serial port number is the serial port number of the Rosmaster.

When selecting firmware, you need to select the LED.hex file in the Debug folder of the project directory.

The last is the configuration selection at the bottom. Be sure to select the option of [DTR low-level reset, RTS high-level into BootLoader] option, otherwise the download may fail.

5. Burn program

Please put the microcontroller on the expansion board into the programming mode first:

First press and hold the BOOT0 key on the expansion board, then press the RESET key, and finally release the BOOT0 key.
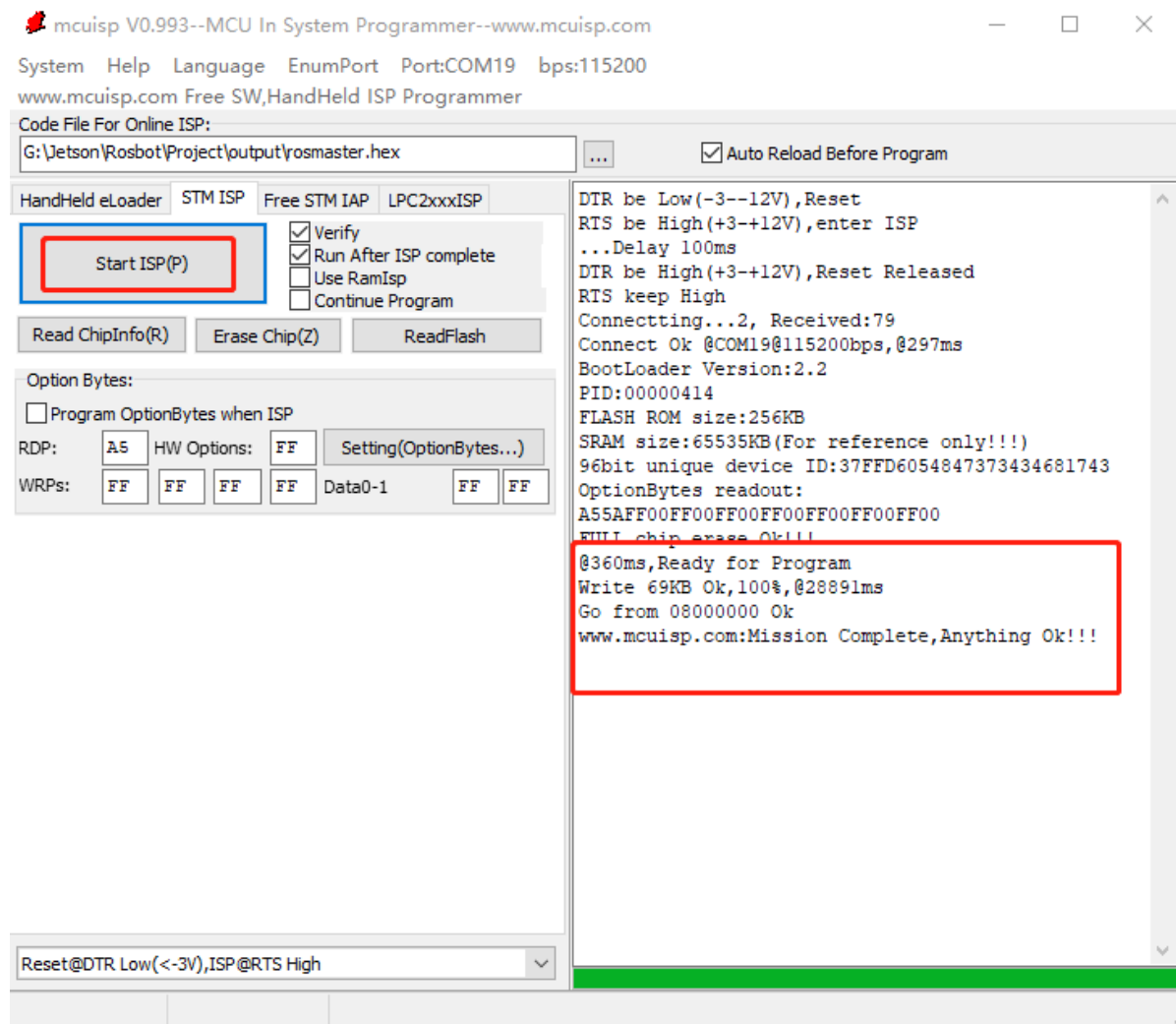
Click [Start Programming], and the mcuisp burning software will burn the firmware we selected in the previous step to the microcontroller on the Rosmaster expansion board. When the prompt appears on the right side [www.mcuisp.com:Mission Complete,Anything Ok!!!], it means that the download is successful.

Notice:

①Before starting programming, please confirm that the serial port number of the Rosmaster is accessible, that is, there is no serial port assistant occupying it.

②Rosmaster enters the burning mode operation, first press and hold the BOOT0 key on the expansion board, then press the RESET key, and finally release the BOOT0 key.

2.9. Program phenomenon

The LEDs on the expansion board flash every 200ms.