

# Program simulation

---

## Program simulation

- Open the project
- Compile the project
- Configure the debugger
  - Program download
- Program debugging
  - Start/stop debugging
  - Debug options
    - Breakpoints
    - Monitor variables
- Complete demonstration
  - Run the program

The tutorial mainly demonstrates the simple debugging steps of the development board using ST-Link.

Tutorial Demonstration Case: LED Control (GPIO)

Since we only provide the standard library version source code for a single case, the tutorial only demonstrates the simulation steps of MDK-ARM

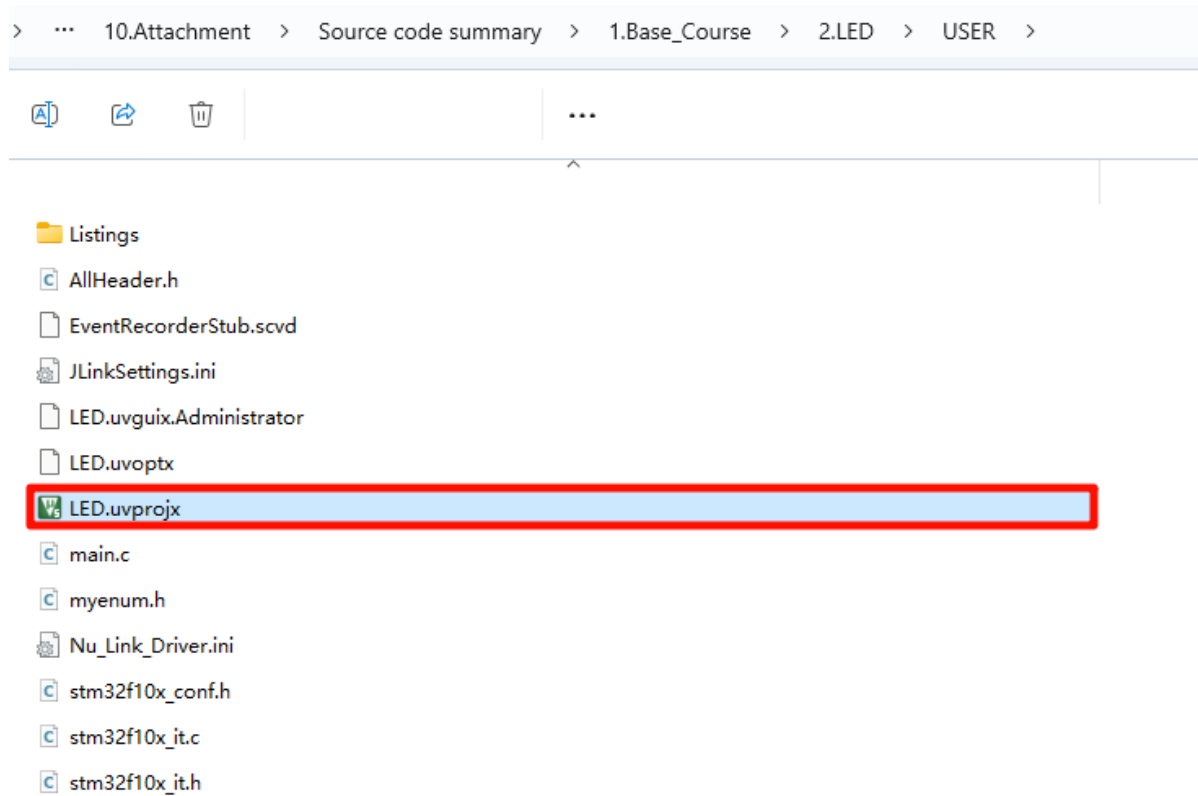
## Open the project

---

Product supporting materials source code path: Attachment → Source Code Summary → 1.Base\_Course → 2.LED

After decompressing the corresponding project file, find the file with the `.uvprojx` suffix and double-click it to open it.

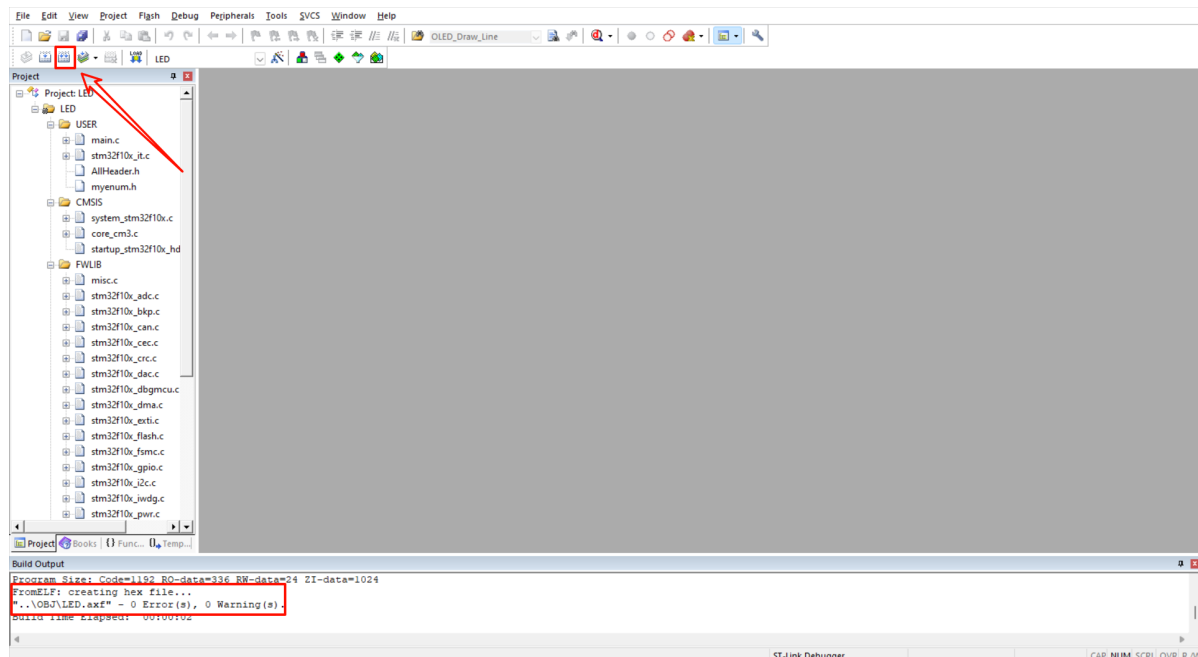
The `.uvprojx` suffix file is located in the USER directory of the project file



## Compile the project

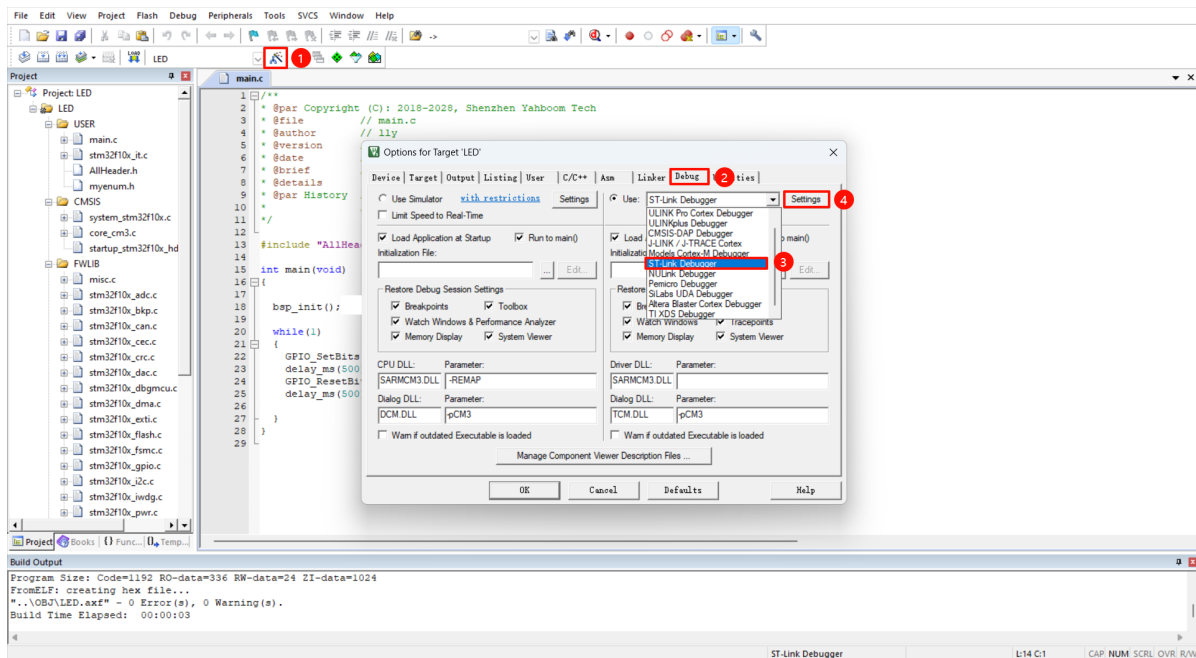
Click the **Rebuild** option on the toolbar to compile the project, and the compilation output bar will prompt the compilation result.

Compiling the project can confirm whether there are any syntax problems in the program you programmed

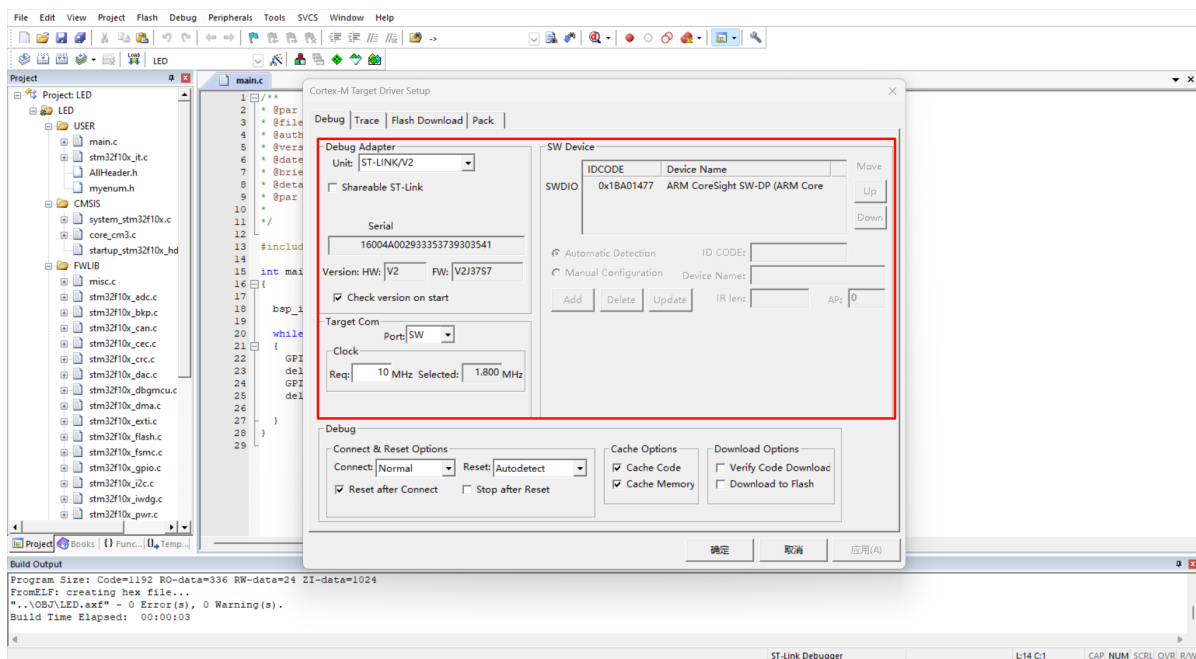


## Configure the debugger

Select **Options for Target...** → **Debug** → **ST-Link Debugger**

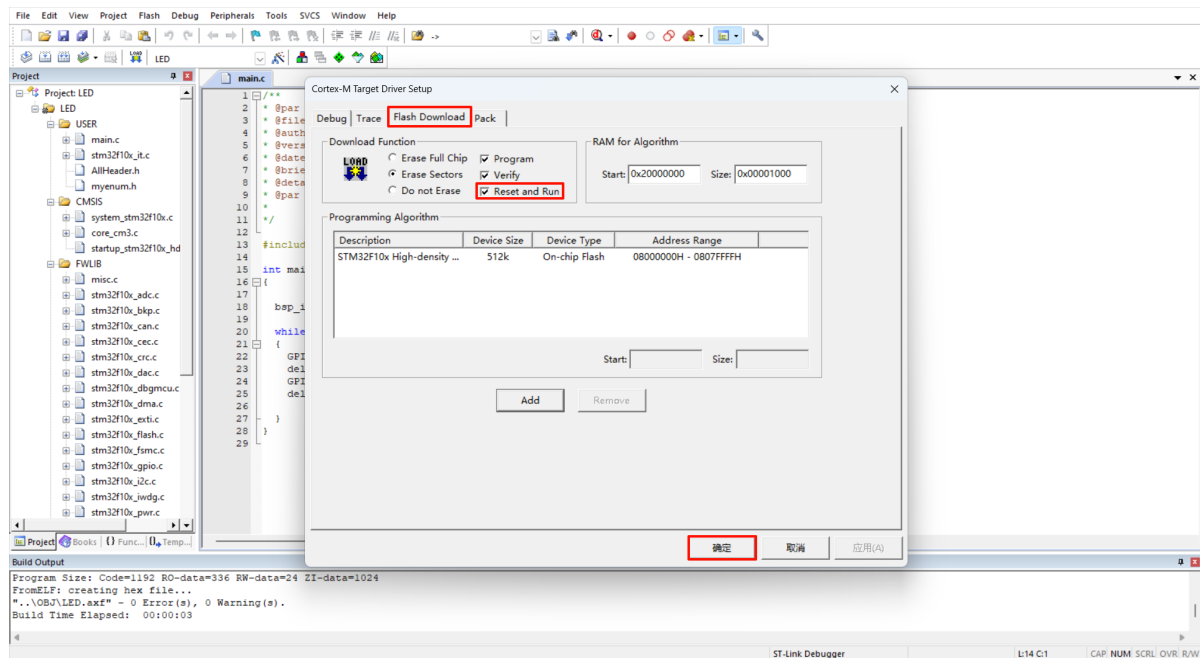


If the system detects ST-Link, you can see the hardware information when entering the debugger interface:

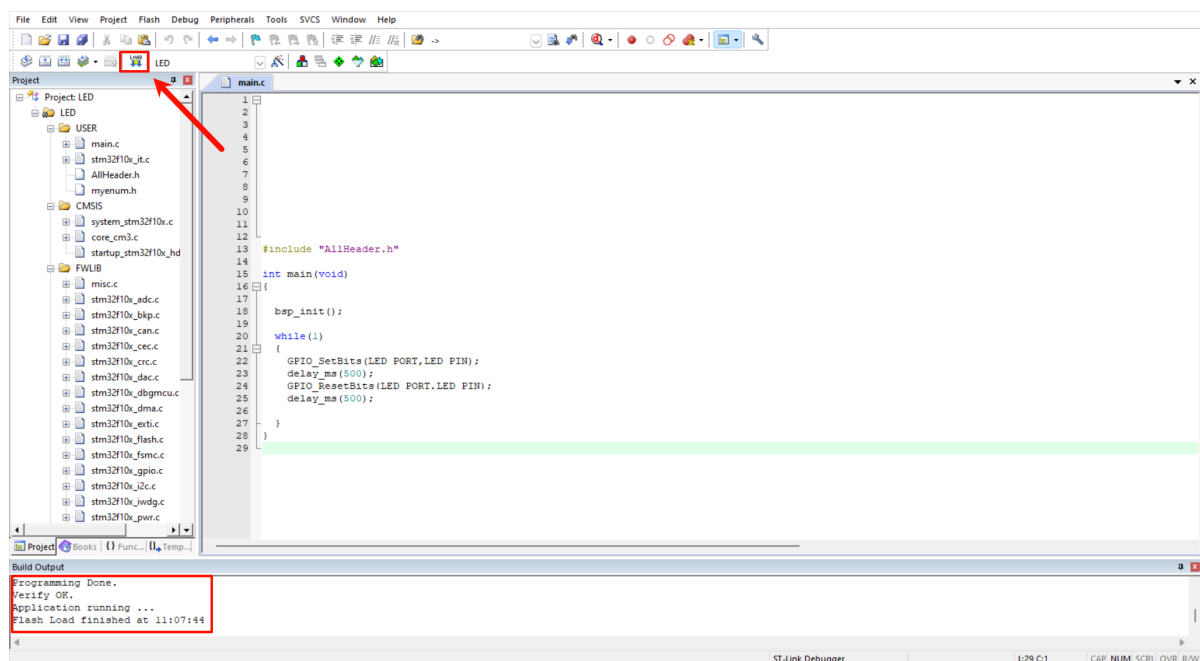


## Program download

Check **Reset** and **Run** to run automatically after downloading the program using ST-Link:



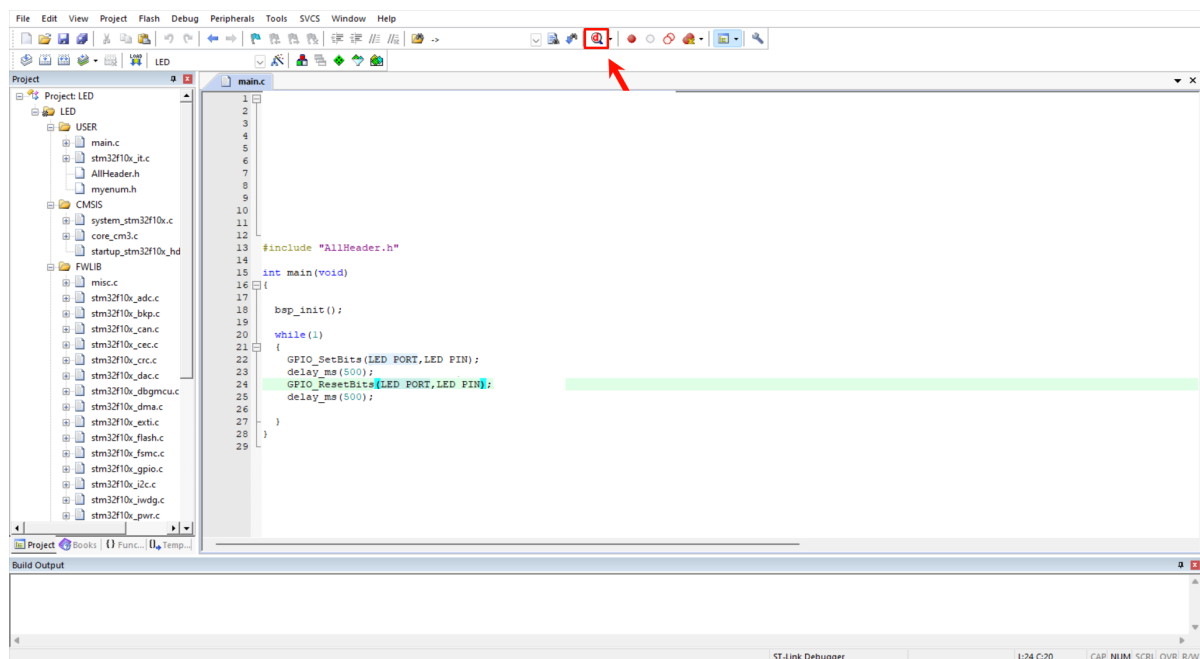
Click **Download** to download the program to the development board via ST-Link:



## Program debugging

### Start/stop debugging

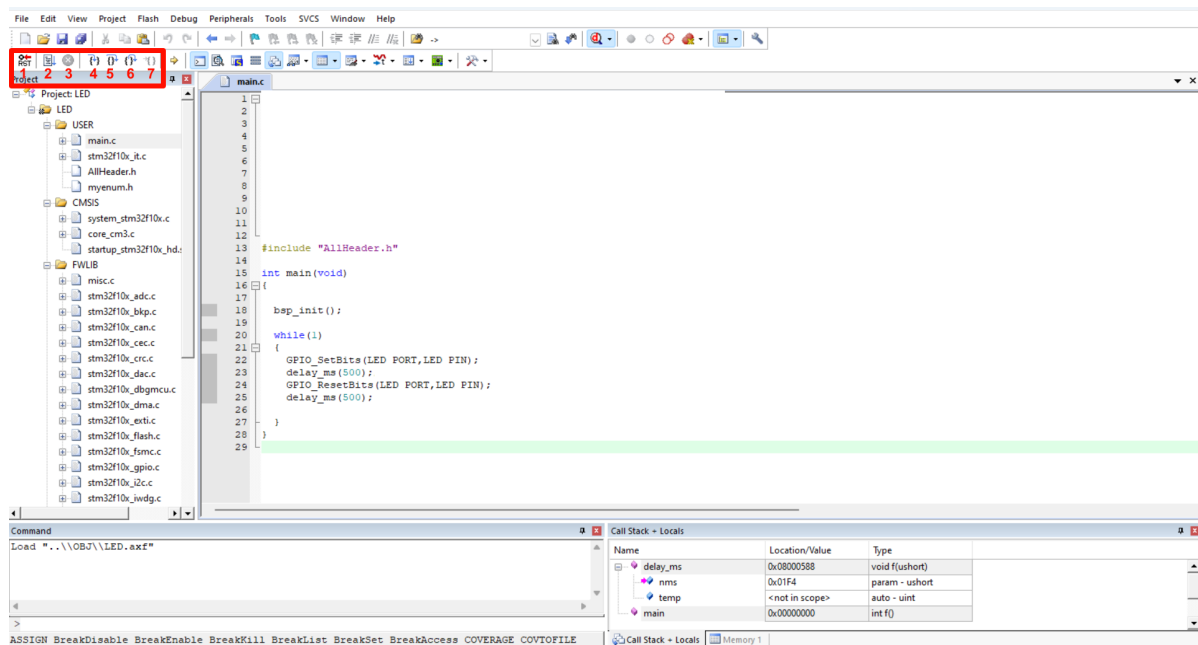
Click **Start/Stop Debug Session** to start and stop debugging mode:



## Debug options

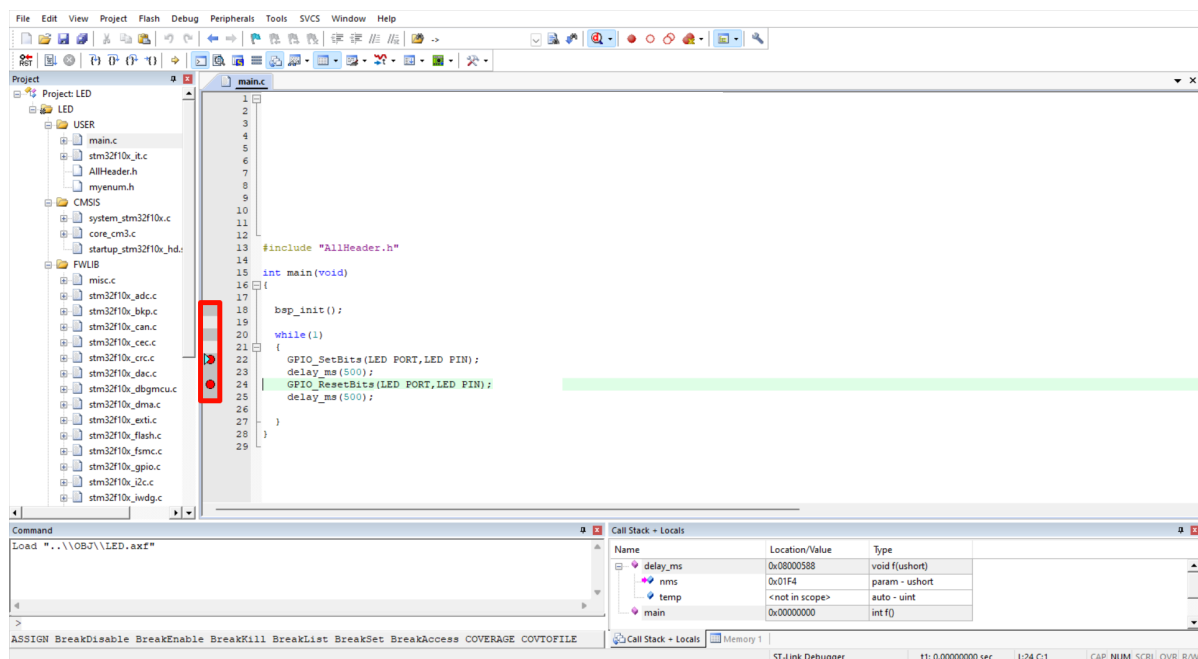
The tutorial only introduces commonly used options:

Serial number	Description
1	Reset: The program performs a reset operation
2	Run at full speed: The program starts to run normally at full speed until the program stops when it encounters a breakpoint
3	Stop running: The program stops running
4	Enter function to run: Run once per click
5	Run line by line: Do not enter the function, run single line
6	Jump out of function to run: Run other statements in the function, and then jump out of the current function to run
7	The program runs to the cursor



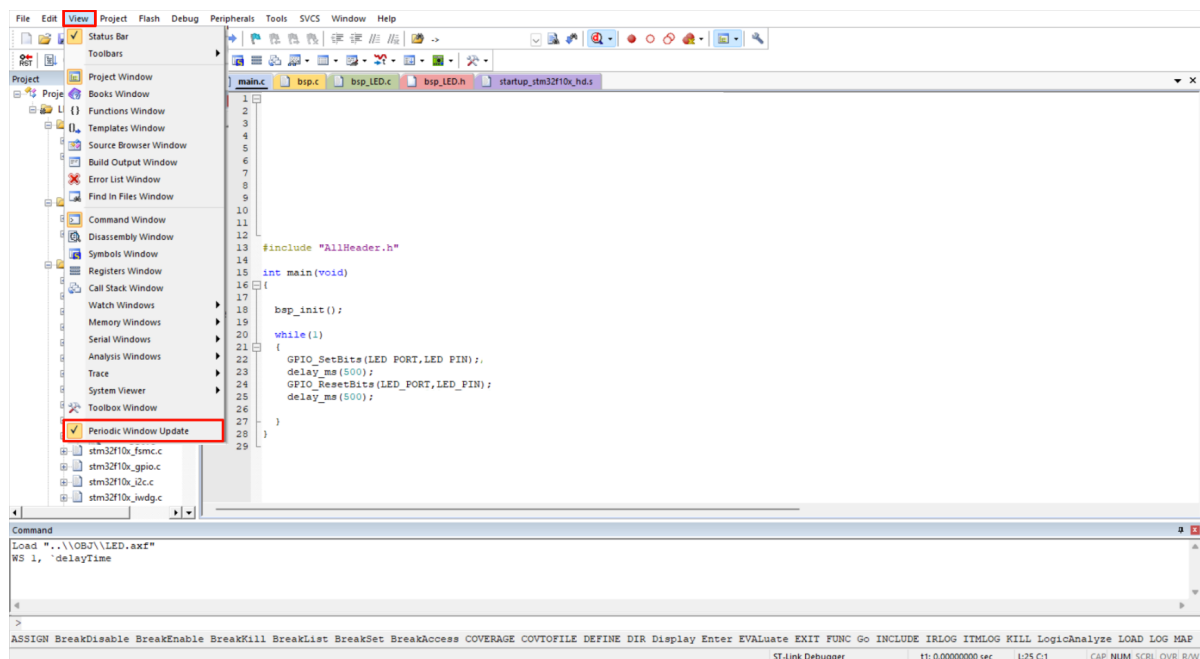
## Breakpoints

When the program reaches a breakpoint, it will stop running. Click the gray area of the picture to add and cancel breakpoints: click once to add, and click again to cancel the breakpoint



## Monitor variables

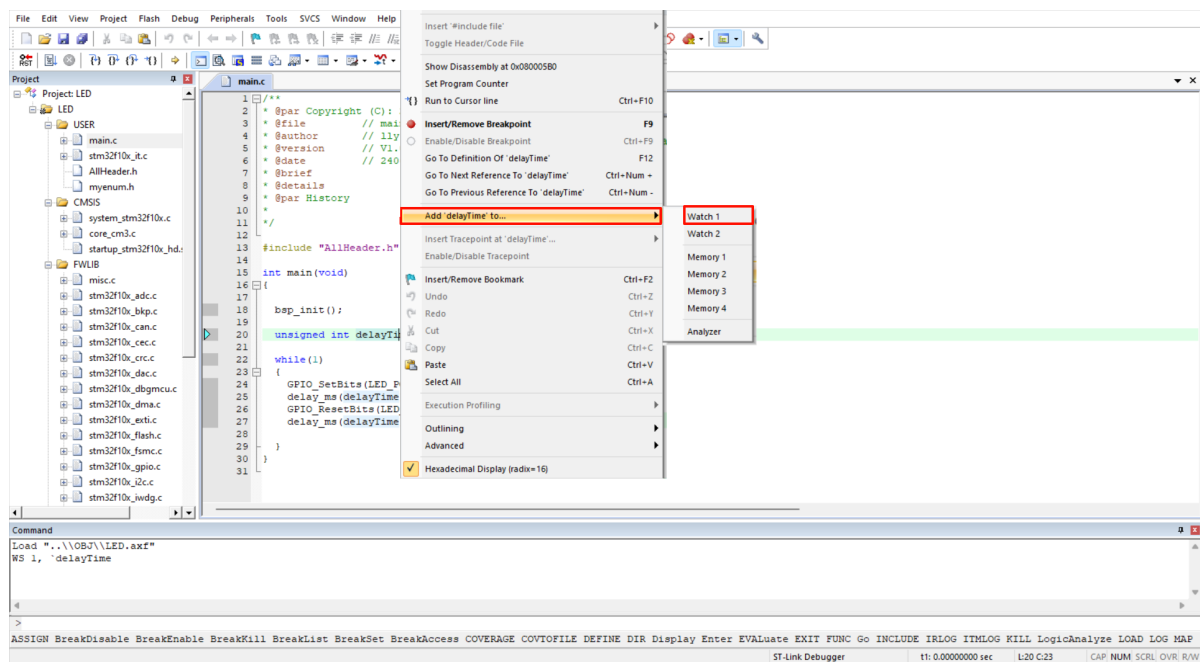
To view the program's variable data in real time, you need to check **view** → **Periodic window**  
Update:



Since there are no viewable variables in the program, we slightly adjust the code: To adjust the code, you need to exit Debug mode, then recompile the project, and re-enter Debug mode

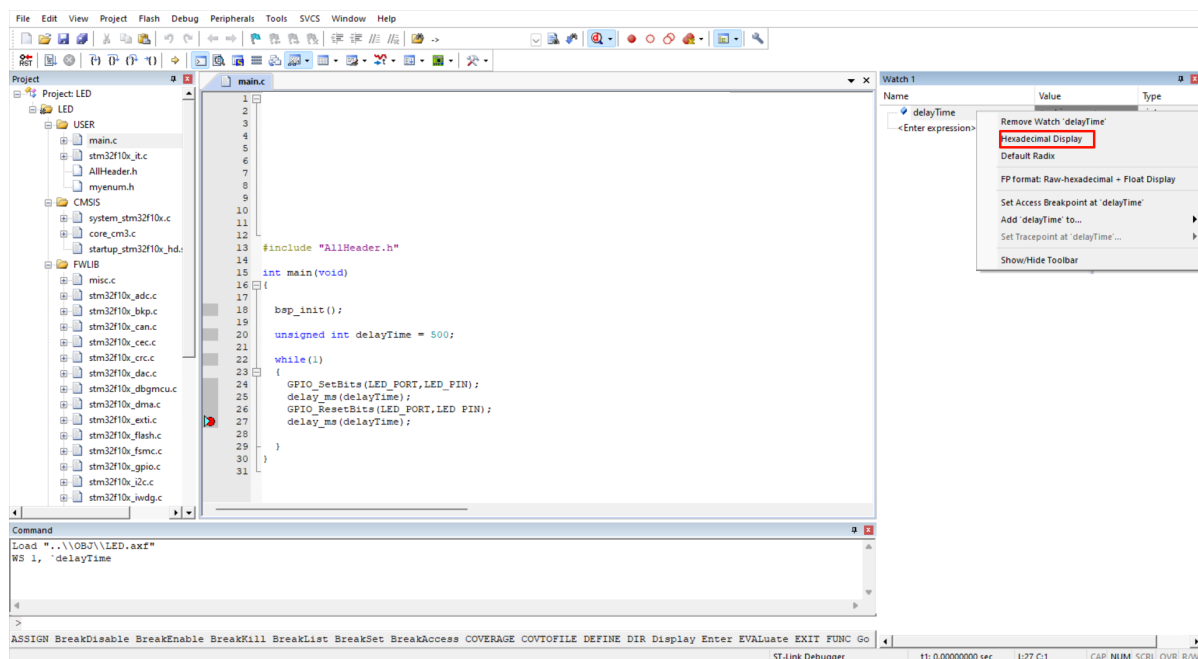
## Add variables

Select the variable in the program and right-click to add it to the monitoring window:



Select the variable in the monitoring window and right-click to change the data display format: cancel the hexadecimal display and use decimal for easy observation

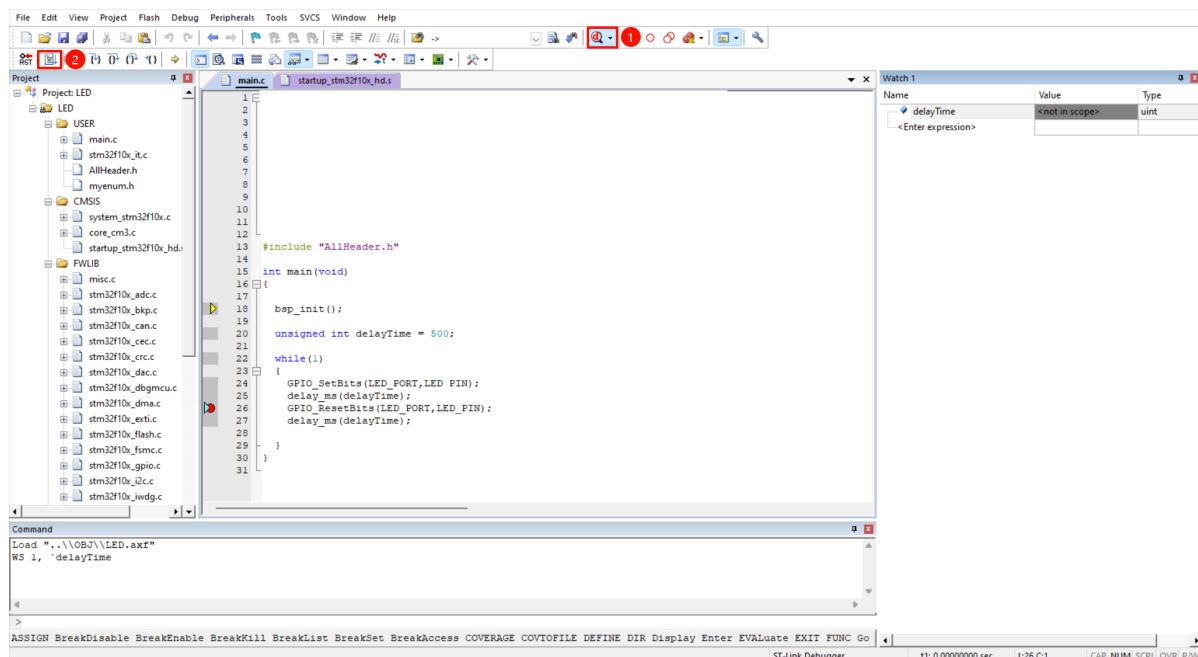
A breakpoint needs to exist in the program, otherwise the data will not be updated when the program is running, and it will be displayed as out of range



## Complete demonstration

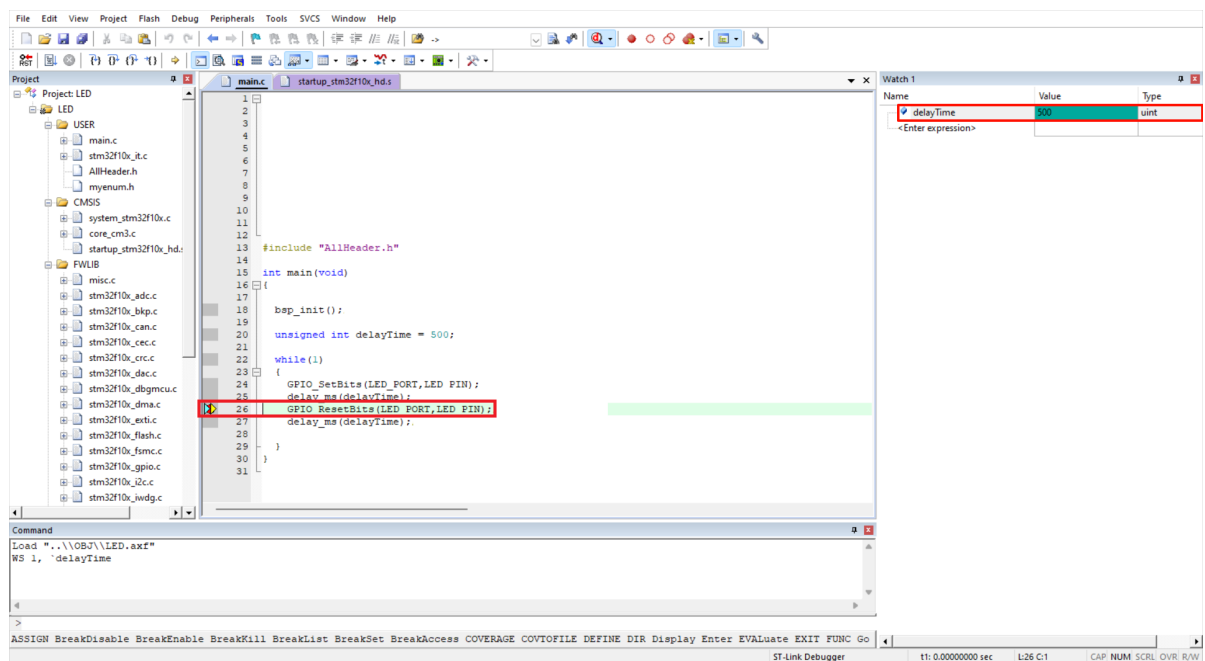
Use LED control (GPIO) project to demonstrate a simple simulation process.

## Run the program

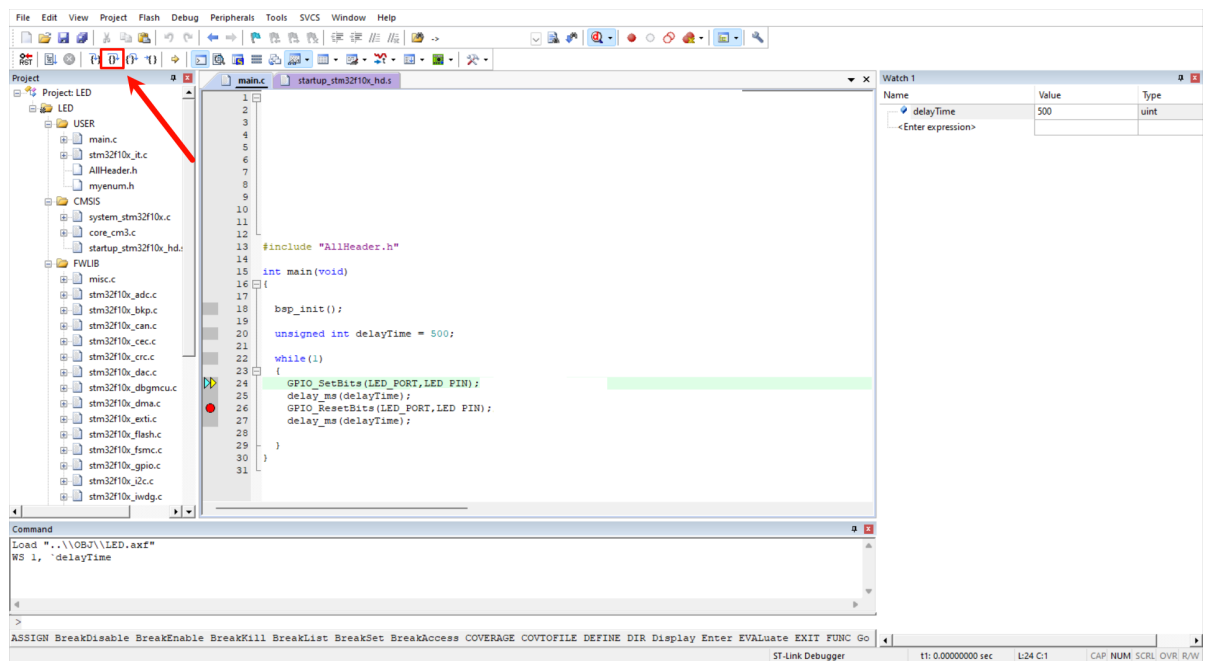


The program will run to the breakpoint and the delay time value will be updated to 500:





Click the line-by-line run icon to control the progress of code execution. At this time, you can observe the LED light phenomenon of the development board.



Click the Debug option to exit the debugging interface:

File Edit View Project Flash Debug Peripherals Tools SVCS Window Help

Project: LED

- LED
  - USER
    - main.c
    - stm32f10x\_it.c
    - AllHeader.h
    - myenum.h
  - CMSIS
    - system\_stm32f10x.c
    - core\_cm3.c
    - startup\_stm32f10x\_hd.c
  - FWLIB
    - misc.c
    - stm32f10x\_adc.c
    - stm32f10x\_bkp.c
    - stm32f10x\_can.c
    - stm32f10x\_cec.c
    - stm32f10x\_crc.c
    - stm32f10x\_dac.c
    - stm32f10x\_dbgmcu.c
    - stm32f10x\_dma.c
    - stm32f10x\_exti.c
    - stm32f10x\_flash.c
    - stm32f10x\_fsmc.c
    - stm32f10x\_gpio.c
    - stm32f10x\_i2c.c
    - stm32f10x\_iwdg.c

main.c startup\_stm32f10x\_hd.c

```
1
2
3
4
5
6
7
8
9
10
11
12
13 #include "AllHeader.h"
14
15 int main(void)
16 {
17     bsp_init();
18
19     unsigned int delayTime = 500;
20
21     while(1)
22     {
23         GPIO_SetBits(LED_PORT, LED_PIN);
24         delay_ms(delayTime);
25         GPIO_ResetBits(LED_PORT, LED_PIN);
26         delay_ms(delayTime);
27     }
28 }
29
30
31
```

Watch 1

Name	Value	Type
delayTime	500	uint
<Enter expression>		

Command

```
Load "...\\OBJ\\LED.axi"
WS i, 'delayTime'
```

ASSIGN BreakDisable BreakEnable BreakKill BreakList BreakSet BreakAccess COVERAGE COVTOFILE DEFINE DIR Display Enter EVALuate EXIT FUNC Go

ST-Link Debugger t1: 74.75905100 sec L:26 C:1 CAP: NUM: SCRL: OVR: RAW