



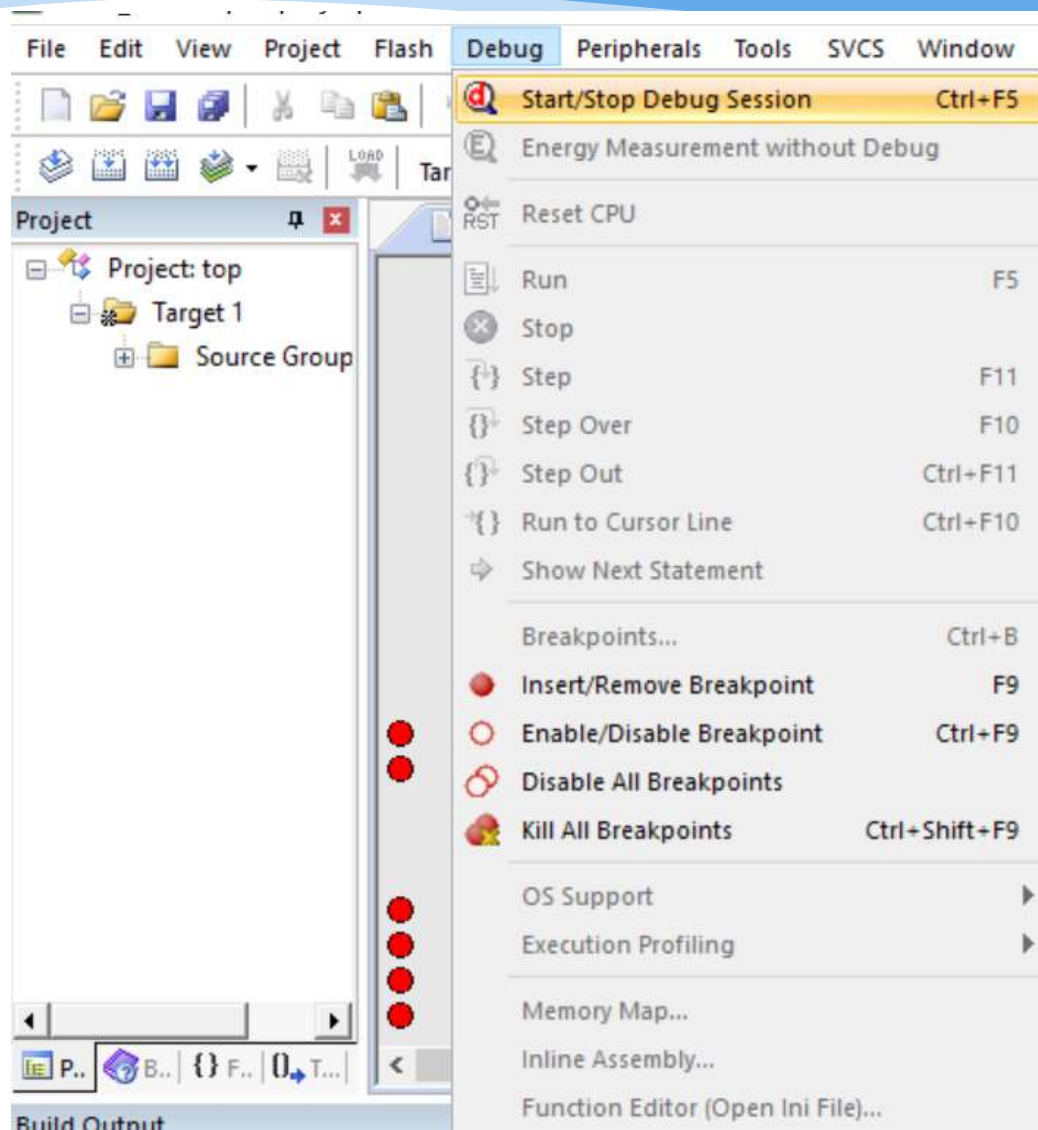
第3章 STC单片机软件开发环境

何宾

2018.03

Keil μ Vision5基本开发流程实现

--软件仿真



Keil μ Vision5基本开发流程实现

--软件仿真

F:\stc_class\top.uvproj - μ Vision

File Edit View Project Flash Debug Peripherals Tools SVCS Window Help

Registers Disassembly

Register	Value
r0	0x00
r1	0x00
r2	0x00
r3	0x00
r4	0x00
r5	0x00
r6	0x00
r7	0x00
sys	
a	0x00
b	0x00
sp	0x00
sp_max	0x07
PC	0x00
auxr1	0x00
dptr	0x000
states	0
sec	0.000
psw	0x00

Disassembly

Address	Disassembly
22: P4_7=~a;	
C:0x0024 A200 MOV C,0x20.0	
C:0x0026 B3 CPL C	
C:0x0027 92C7 MOV P4_7(0xC0.7),C	
23: }	
C:0x0029 22 RET	
C:0x002A 787F MOV R0,#0x7F	
C:0x002C E4 CLR A	
C:0x002D F6 MOV @R0,A	
C:0x002E D8FD DJNZ R0,C:002D	
C:0x0030 758120 MOV SP(0x81),#0x20	
C:0x0033 020003 LJMP main(C:0003)	

main.c

```
11 sbit P4_7=P4^7;
12 void main()
13 {
14     volatile bit a=1,b=0;
15     P1M0=0;
16     P1M1=0;
17     P4M0=0;
18     P4M1=0;
19     P1_6=a & b;
20     P1_7=a | b;
```

Command

Load "F:\\stc_class\\Objects\\top"

ASM ASSIGN BreakDisable BreakEnable BreakKill BreakList BreakSet BreakAccess

Call Stack + Locals

Name	Location/Value	Type
------	----------------	------

Simulation t1: 0.00002217 sec L:22 C:1 CAP NUM SCRL OVR R/W

Keil μ Vision5基本开发流程实现

--硬件在线调试

STC15系列单片机开始提供的一个**重要的功能**，通过**硬件在线调试**使得程序开发人员能够发现软件仿真时不能检测到的一些更深层次的设计问题。

- 当程序不能响应外部中断的时候，可能有以下几种情况，
 - 全局中断没有使能？
 - 对应的外部中断没有使能？
 - 中断服务程序代码有问题？（没有进入中断服务程序？没有从中断服务程序正常返回？）

Keil μ Vision5基本开发流程实现

--硬件在线调试

- 这些可能性只有通过硬件在线调试功能才能确认。
- 软件仿真绝不能代替硬件在线调试。

Keil μ Vision5基本开发流程实现

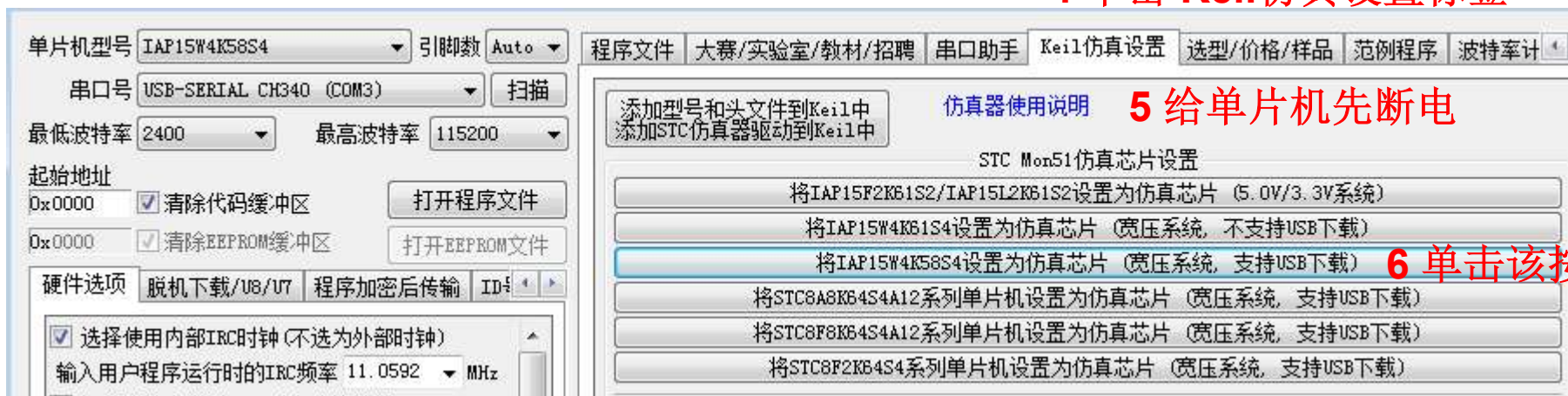
--硬件在线调试

1 通过**USB**接口，连接单片机目标系统和**PC**/笔记本电脑

2 确认单片机型号和串口号

3 通过打开程序文件按钮，确认**HEX**文件

4 单击 **Keil**仿真设置标签



7 给单片机上电

Keil μ Vision5基本开发流程实现

--硬件在线调试

- STC-ISP软件右下方的界面中，显示下载程序的相关信息。



Keil μ Vision5基本开发流程实现

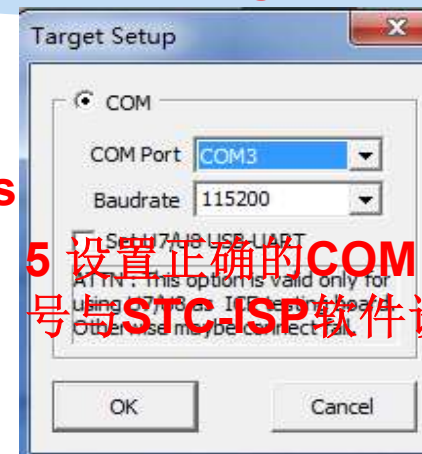
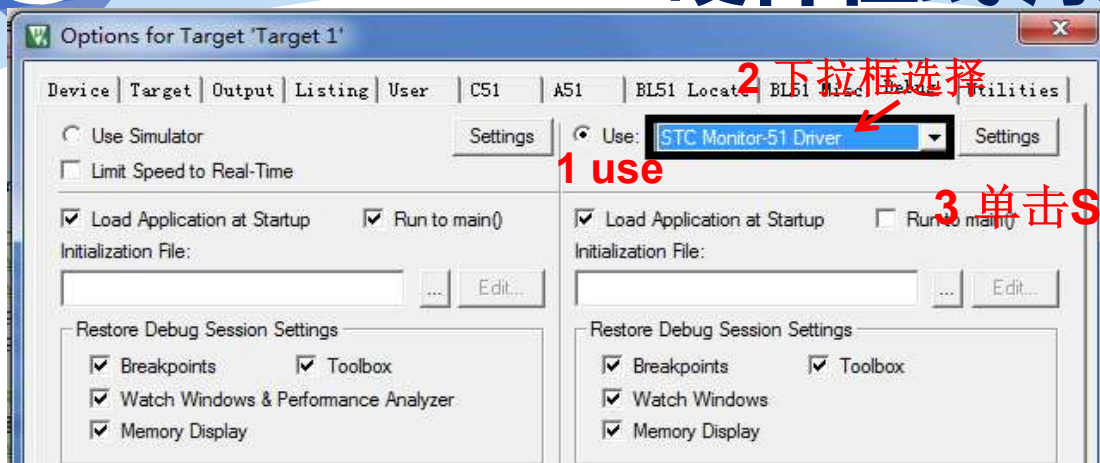
--硬件在线调试

- 用Keil μ Vision5打开与烧写文件对应的设计工程top.uvproj。
- 在Keil μ Vision5集成开发环境左侧的Project窗口中，选中Target 1并单击右键，出现浮动菜单，选择Options for Target 'Target 1' 选项。
- 出现Options for Target 'Target 1' 对话框界面，如图所示。在该界面中，单击Debug标签。在该标签界面右侧窗口中，按如下设置参数：

Keil μ Vision5基本开发流程实现

--硬件在线调试

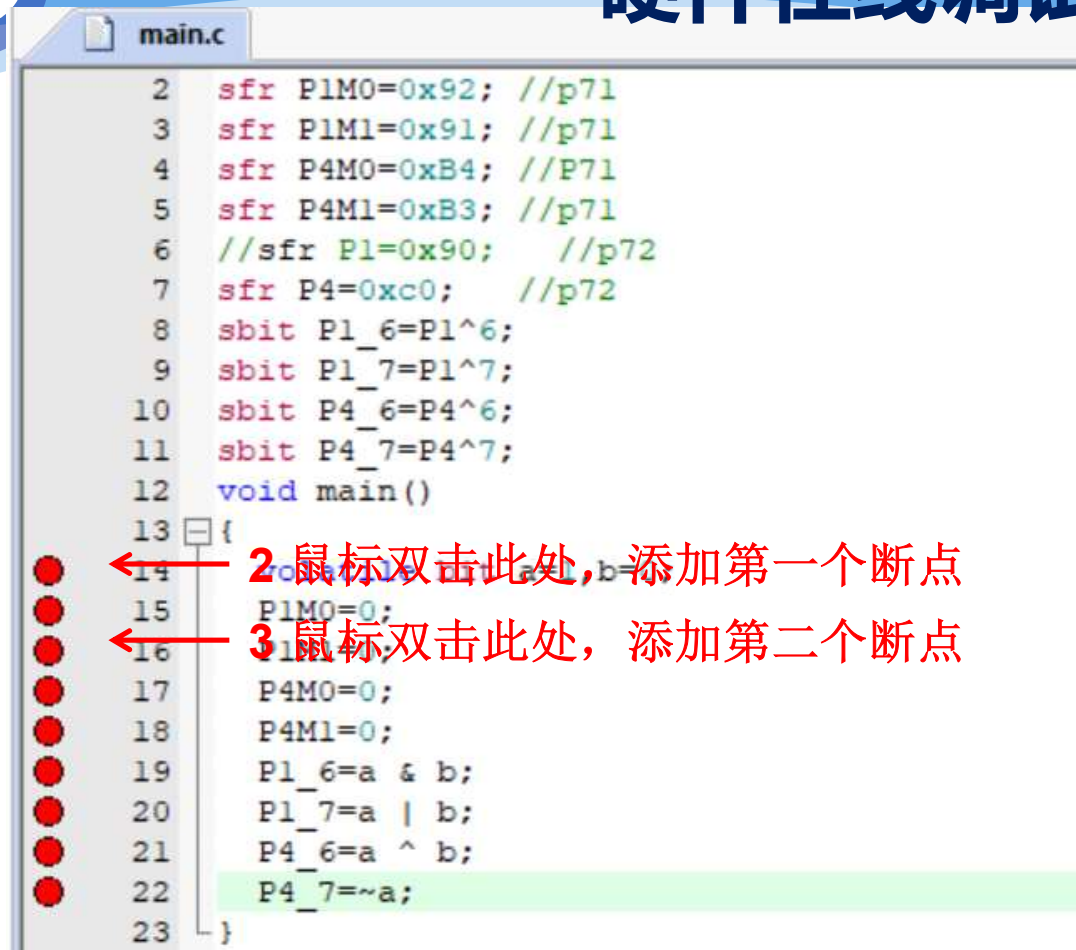
4 出现Target Setup



7 单击OK退出Options for Target对话框界面

Keil μ Vision5基本开发流程实现

--硬件在线调试



```
main.c
2  sfr P1M0=0x92; //p71
3  sfr P1M1=0x91; //p71
4  sfr P4M0=0xB4; //P71
5  sfr P4M1=0xB3; //p71
6  //sfr P1=0x90;    //p72
7  sfr P4=0xc0;    //p72
8  sbit P1_6=P1^6;
9  sbit P1_7=P1^7;
10 sbit P4_6=P4^6;
11 sbit P4_7=P4^7;
12 void main()
13 {
14     //sfr P1=0x90;    //p72
15     P1M0=0;
16     P1M1=0;
17     P4M0=0;
18     P4M1=0;
19     P1_6=a & b;
20     P1_7=a | b;
21     P4_6=a ^ b;
22     P4_7=~a;
23 }
```

2 鼠标双击此处，添加第一个断点

3 鼠标双击此处，添加第二个断点

- 在Keil μ Vision主界面主菜单下，选择Debug->Start/Stop Debug Session，进入调试器模式。

Keil μ Vision5基本开发流程实现

--硬件在线调试

F:\stc_class\top.uvproj - μ Vision

File Edit View Project Flash Debug Peripherals Tools SVCS Window Help

Registers

Register	Value
r0	0x00
r1	0x00
r2	0x00
r3	0x00
r4	0x00
r5	0x00
r6	0x00
r7	0x00
sp	0x00
sp_max	0x07
PC	0x00
auxr1	0x00
dptr	0x000
states	0
sec	0.000
psw	0x00

Disassembly

Address	Disassembly
22: P4_7=~a;	
C:0x0024 A200	MOV C,0x20.0
C:0x0026 B3	CPL C
C:0x0027 92C7	MOV P4_7(0xC0.7),C
23: }	
C:0x0029 22	RET
C:0x002A 787F	MOV R0,#0x7F
C:0x002C E4	CLR A
C:0x002D F6	MOV @R0,A
C:0x002E D8FD	DJNZ R0,C:002D
C:0x0030 758120	MOV SP(0x81),#0x20
C:0x0033 020003	LJMP main(C:0003)

main.c

```
11 sbit P4_7=P4^7;
12 void main()
13 {
14     volatile bit a=1,b=0;
15     P1M0=0;
16     P1M1=0;
17     P4M0=0;
18     P4M1=0;
19     P1_6=a & b;
20     P1_7=a | b;
```

Command

Load "F:\stc_class\Objects\top"

ASM ASSIGN BreakDisable BreakEnable BreakKill BreakList BreakSet BreakAccess

Call Stack + Locals

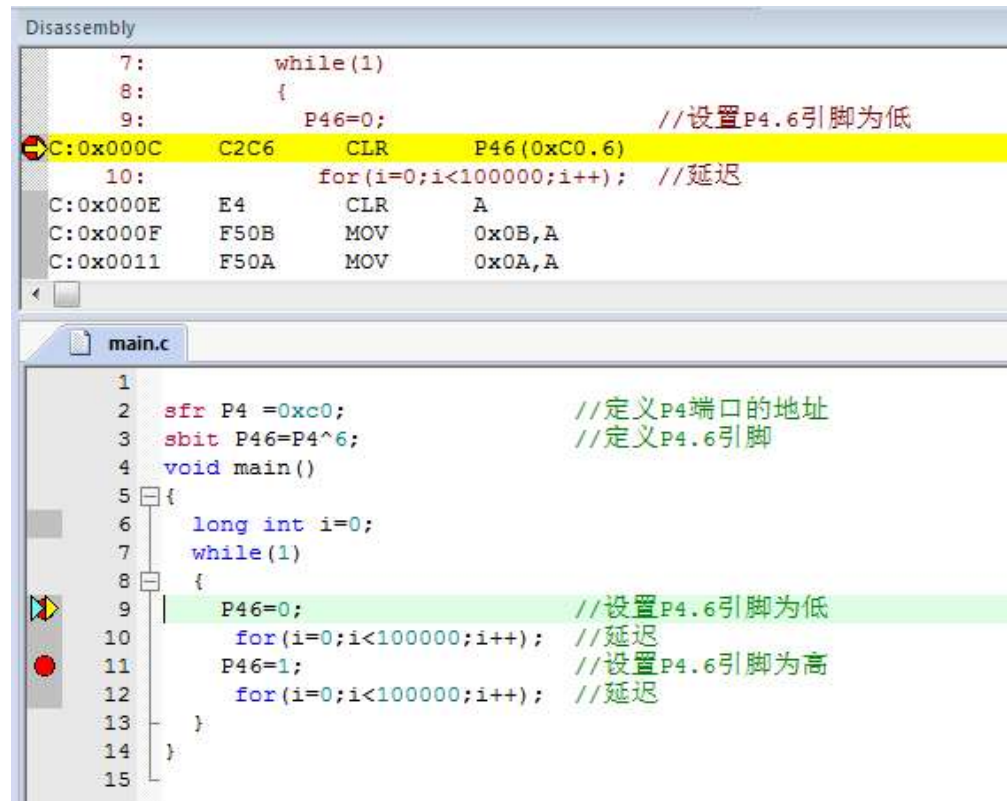
Name	Location/Value	Type
------	----------------	------

Simulation t1: 0.00002217 sec L:22 C:1 CAP NUM SCRL OVR RV

Keil μ Vision5基本开发流程实现

--硬件在线调试

- 单步运行，观察LED灯的状态。



The screenshot displays the Keil μ Vision5 IDE interface. The top window, titled 'Disassembly', shows the assembly code for the program. The bottom window, titled 'main.c', shows the corresponding C source code. The assembly code is as follows:

```
7:      while(1)
8:      {
9:          P46=0;          //设置P4.6引脚为低
C:0x000C  C2C6  CLR      P46(0xC0.6)
10:         for(i=0;i<100000;i++); //延迟
C:0x000E  E4      CLR      A
C:0x000F  F50B  MOV      0x0B,A
C:0x0011  F50A  MOV      0x0A,A
```

The C source code is as follows:

```
1
2  sfr P4 =0xc0;          //定义P4端口的地址
3  sbit P46=P4^6;          //定义P4.6引脚
4  void main()
5  {
6      long int i=0;
7      while(1)
8      {
9          P46=0;          //设置P4.6引脚为低
10         for(i=0;i<100000;i++); //延迟
11         P46=1;          //设置P4.6引脚为高
12         for(i=0;i<100000;i++); //延迟
13     }
14 }
15
```