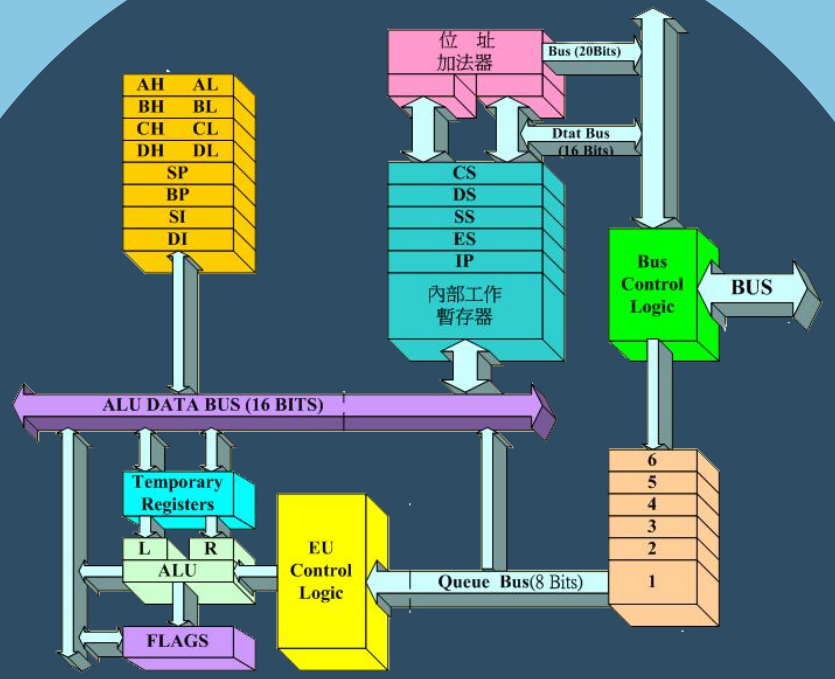
单步中断

贺利坚　主讲



汇编语言程序设计

Assembly Language

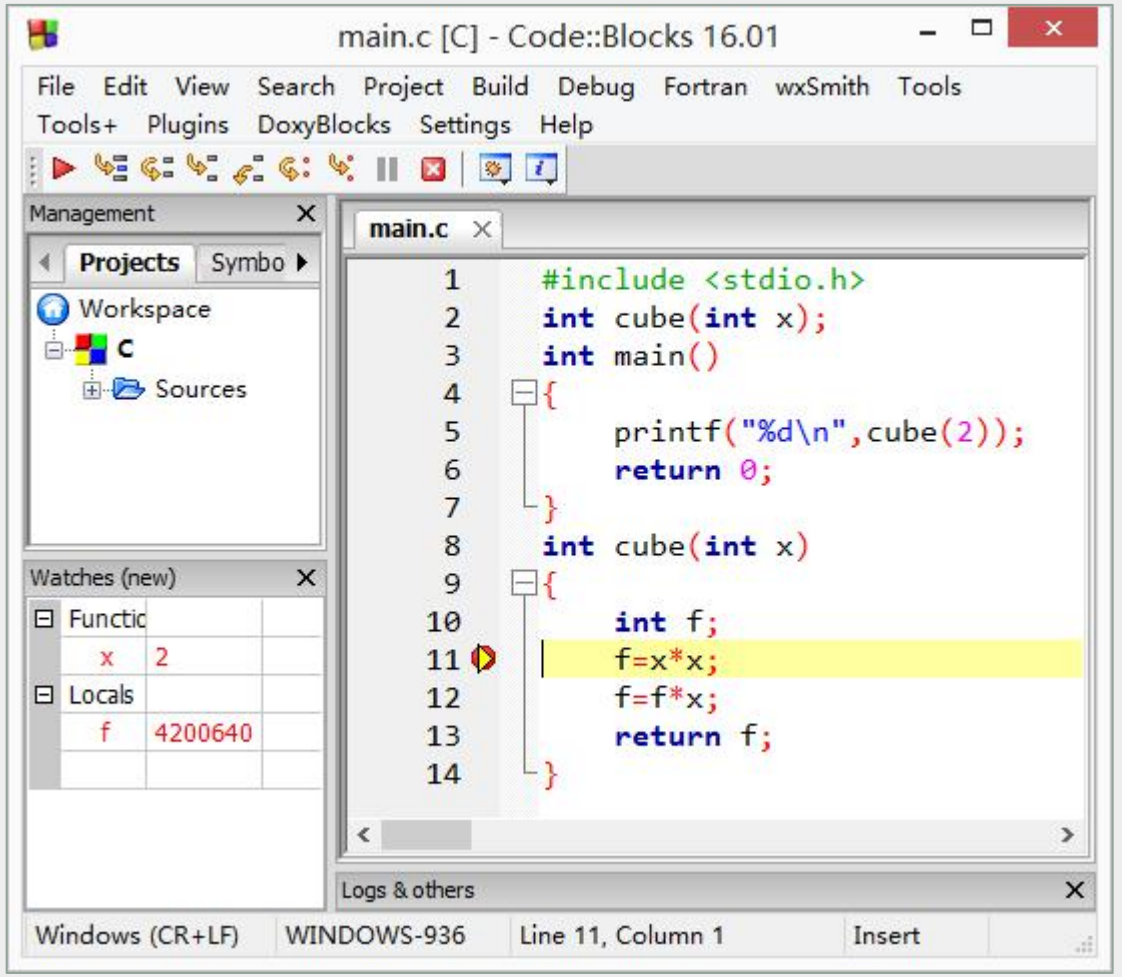
由Debug中的t命令说起……

- 程序的正常执行：取指令、改变CS:IP、执行指令、取指令……
- Debug提供了单步中断的中断处理程序，功能为显示所有寄存器中的内容后等待输入命令。

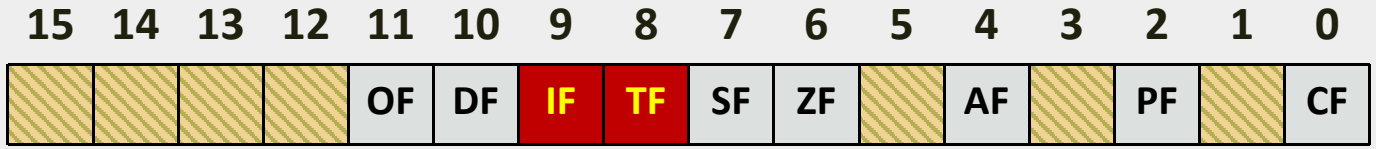
```
C:\>debug p12-1.exe
-t

AX=1000  BX=0000  CX=002C  DX=0000  SP=0000  BP=0000  SI=0000  DI=0000
DS=075A  ES=075A  SS=0769  CS=076C  IP=0003   NV UP EI PL NZ NA PO NC
076C:0003 B310          MOV     BL,10
-
```

- 是什么，让CPU能执行一条指令就停下来？
  - Debug利用了**CPU提供的**单步中断的功能
  - 使用t命令时，Debug将TF标志设为1，使CPU工作在单步中断方式下……
- 自定义单步中断处理程序，还可以实现特殊的功能。

- 让指令停下来，能做不少事



单步中断过程与处理



- 两个和中断相关的寄存器标志位
  - TF-陷阱标志(Trap flag)：用于调试时的单步方式操作。当TF=1时，每条指令执行完后产生陷阱，由系统控制计算机；当TF=0时，CPU正常工作，不产生陷阱。
  - IF-中断标志(Interrupt flag)：当IF=1时，允许CPU响应可屏蔽中断请求；当IF=0时，关闭中断。

- CPU在执行完一条指令之后，如果检测到标志寄存器的TF位为1，则产生单步中断（中断类型码为1），引发中断过程，执行中断处理程序。
- 中断过程：
  （1）取得中断类型码1；
  （2）标志寄存器入栈，TF、IF设置为0；
  （3）CS、IP入栈；
  （4）(IP)=(1*4)，(CS)=(1*4+2)。

为什么？

- 中断处理程序也由一条条指令组成的。
- 如果在执行中断处理程序之前，TF=1，则CPU在执行完中断处理程序的第一条指令后，又要产生单步中断，转去执行单步中断的中断处理程序的第一条指令……
- 上面的过程将陷入一个永远不能结束的循环，CPU永远执行单步中断处理程序的第一条指令。
- 所以，在进入中断处理程序之前，设置TF=0。

应用：中断不响应的情况

- 一般情况下，CPU在执行完当前指令后，如果检测到中断信息，就响应中断，引发中断过程。
- 在有些情况下，CPU 在执行完当前指令后，即便是发生中断，也不会响应。
- 例：在执行完向 ss寄存器传送数据的指令后，即便是发生中断，CPU 也不会响应。
  - 原因：ss:sp联合指向栈顶，而对它们的设置应该连续完成。
  - 以此保证对栈的正确操作！
- 例：设置栈

```
mov ax,1000
mov ss,ax
mov sp,10
```

```
C:\>debug
-a
073F:0100 mov ax, 1000
073F:0103 mov ss, ax
073F:0105 mov sp, 10
073F:0108 mov cx, 2
073F:010B
-t

AX=1000  BX=0000  CX=0000  DX=0000  SP=00FD  BP=0000  SI=0000  DI=0000
DS=073F  ES=073F  SS=073F  CS=073F  IP=0103   NV UP EI PL NZ NA PO NC
073F:0103 8ED0          MOV     SS,AX
-t

AX=1000  BX=0000  CX=0000  DX=0000  SP=0010  BP=0000  SI=0000  DI=0000
DS=073F  ES=073F  SS=1000  CS=073F  IP=0108   NV UP EI PL NZ NA PO NC
073F:0108 B90200        MOV     CX,0002
-_
```

```
mov ax,1000
mov ss,ax
mov ax,0
mov sp,10
```