控制转移指令及分支循环结构

信息工程学院　　陈梅

第4讲-3 控制转移指令和分支循环调用结构要点

要点

- 控制转移指令
- 调用返回指令
- 位操作指令中的控制转移指令
- 分支结构、循环结构、子程序调用结构举例

控制转移类指令

1. 长转移指令 LJMP addr16

；机器码

02H	addr8h	addr8l

（1）16位地址是64K范围内的跳转。

（2）先执行 PC+3→PC，

然后 addr16→PC。

2. 绝对转移指令 AJMP addr11

；机器码

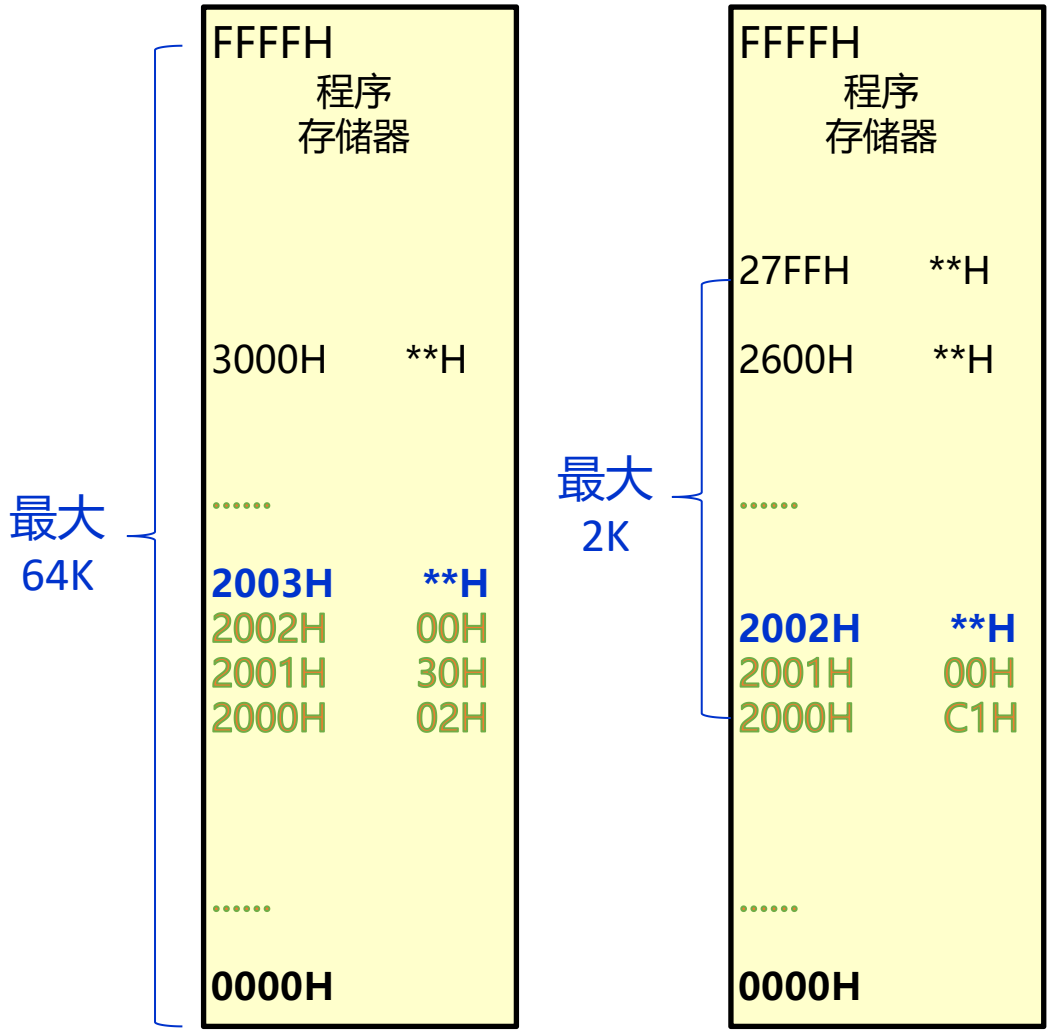
A10A9A8 0 0001	A7~A0

（1）11位地址A10～A0是2K范围的跳转。

（2）先执行PC+2→PC，

然后addr11→ PC10～PC0，

PC15～PC11不变。



3. 相对转移指令　SJMP　rel

；机器码

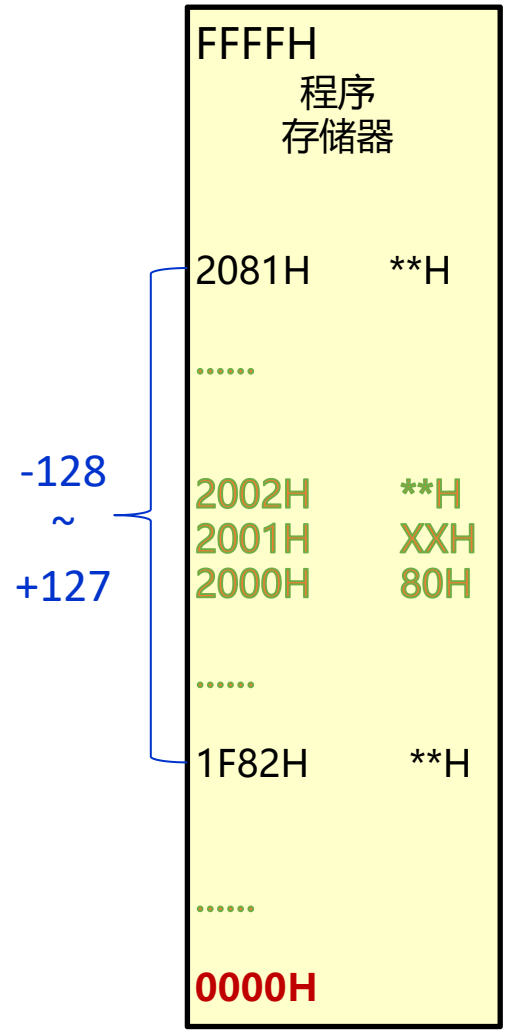
80H	rel补码

（1）rel8位有符号数是-128～+127范围的跳转。

（2）先执行PC+2 →PC，

然后PC=PC+rel，rel为正数

或　PC=PC-(FFH+1-rel)，rel为负数



空操作指令　NOP

操作：PC+1→PC

特点：CPU不进行任何实际操作，只消耗1个机器周期的时间。常用于程序中的等待或时间的延迟。

无条件跳转指令

1. 长转移指令　　LJMP addr16
2. 绝对转移指令　AJMP addr11
3. 相对转移指令　SJMP rel

实际编程中无论LJMP、AJMP、SJMP，都只需写上目的地址标号。Addr16、Addr11、rel都由汇编程序自动计算。

例：LOOP: MOV　　A，R6

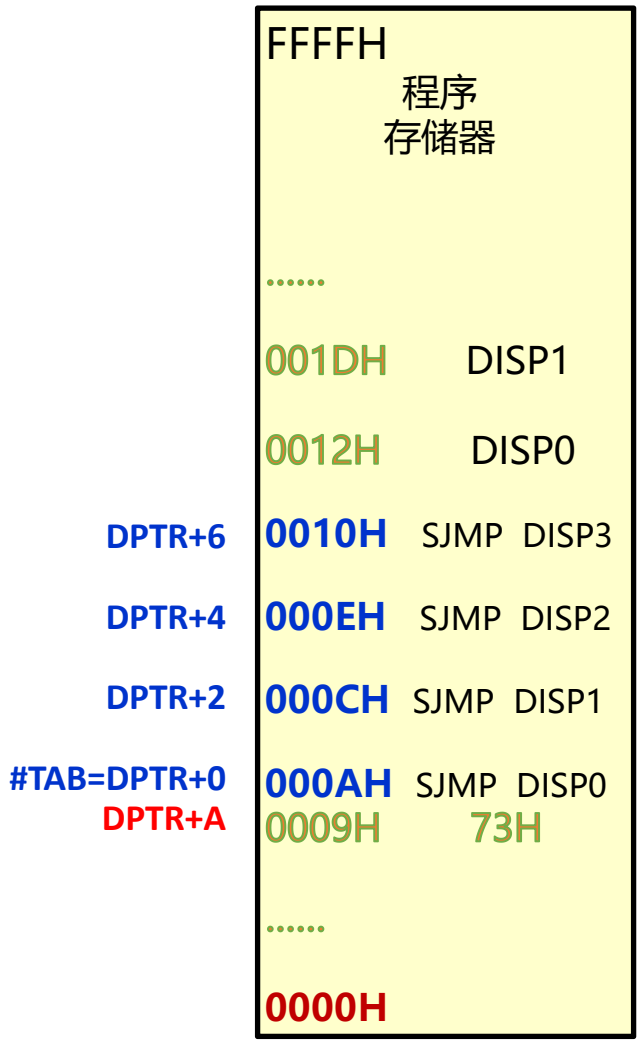
…

SJMP　　LOOP　或　AJMP　　LOOP　或　LJMP　LOOP

4. 间接跳转指令　JMP　@A+DPTR （又称散转指令）；机器码 73H

- 功能：跳转到由PC=A+DPTR计算所得地址处。
- 注意：计算PC不会改变A和DPTR原有的内容，也不影响标志。
- 用法：DPTR固定，A变化，可实现多分支转移。

```
        ……
        MOV     R0,#00H
NEXT:   MOV     A,R0
        MOV     DPTR,#TAB
        RL      A
        JMP     @A+DPTR
TAB:    SJMP    DISP0
        SJMP    DISP1
        SJMP    DISP2
        SJMP    DISP3
DISP0:  ……
DISP1:  ……
DISP2:  ……
DISP3:  ……
        ……
```



条件转移指令

条件转移指令：条件满足时转移，不满足时顺序执行下一条指令。

- 判断累加器A是否为0

  JZ rel　　；若A=0，则转移

  JNZ rel　　；若A≠0，则转移

- 减1不为0转移指令

  - DJNZ Rn, rel
  - DJNZ direct, rel
  - ；(Rn或direct)-1 → (Rn或direct), 若结果不为0则转移.

  - 用法：用于控制程序循环。预先装入循环次数，以减1后是否为“0”作为转移条件，即实现按次数控制循环。

- 比较不相等转移指令

  CJNE A,direct,rel

  CJNE A,#data,rel

  CJNE Rn,#data,rel

  CJNE @Ri,#data,rel

特点：

1）前两个操作数相减，若值不相等则转移，但不保留相减结果，也不改变任何一个操作数的内容；

2）影响标志位。

当第一操作数<第二操作数，则进位标志位CY=1；

当第一操作数≥第二操作数，则进位标志位CY=0。

用法：比较前面两个操作数大小，若值不相等则转移。常用于循环结构。

位操作指令——控制转移类

- 针对位地址空间的操作指令
- 只影响本身的CY（写作C），不影响其它标志位。
- 注意操作数的字节地址和位地址的区分

条件转移类位操作指令

```
JC      rel             ；Cy=1，则转移
JNC     rel             ；Cy=0，则转移
JB      bit, rel            ；bit=1，则转移
JNB     bit, rel        ；bit=0，则转移
JBC     bit, rel        ；先bit=1，则转移，再清bit
```

条件转移类字节操作指令

```
JZ  rel                 ；若A=0，则转移
JNZ  rel        ；若A≠0，则转移
```

分支程序结构

- 无条件转移：LJMP addr16 、SJMP rel、AJMP addr11

- 有条件分支转移

1. 单分支选择结构

如： JNB Acc.7,RETURN

```
if(条件) { 语句1; }
else     { 语句2; }
```

2. 多分支选择结构

间接转移指令“JMP @A+DPTR”由数据指针DPTR决定多分支转移程序的首地址，由A的内容选择对应分支。

```
switch(条件)
 { case 值1: 语句1; }
   case 值2: 语句2;
          ……          }
```

循环程序结构

（1）计数循环控制结构

如：

```
    MOV   R7, #**H  或  MOV   30H, #**H
LP:       ……                  ……
    DJNZ   R7, LP     或   DJNZ   30H, LP
```

for(循环初值；循环次数控制; 循环变量增减)
{ 循环体语句； }

（2）条件控制结构

循环控制中，设置一个条件，判断是否满足该条件，
如满足，则循环结束；如不满足该条件则循环继续。

循环初值;
while(循环条件控制)
{ 循环体语句;
循环变量增减； }

循环结构举例

【例3-23】在12MHz晶振系统中，写出50ms延时子程序。

说明：12MHz晶振的一个机器周期为1μs，DJNZ指令两个机器周期为2μs

50ms=2μs*25000=2μs*200*125

```
DELAY:   MOV    R7,#200   ;1μs （置外循环次数）
DEL1:    MOV    R6,#125   ;1μs （置内循环次数）
DEL2:    DJNZ   R6,DEL2   ;2μs×125次=250μs
         DJNZ   R7,DEL1   ;2μs
         RET              ;2μs
```

实际延时时间为 [1+(1+2*125+2)×200+2]μs=50.603ms

软件延时程序的特点：

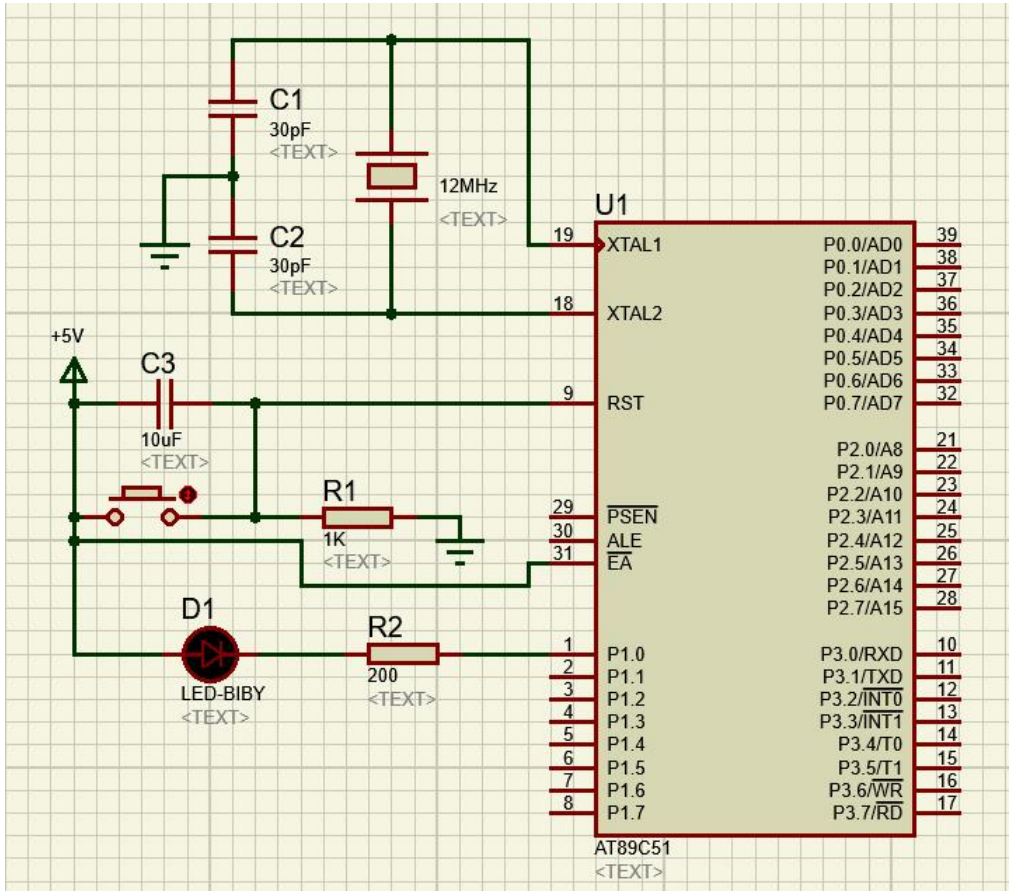
- 软件延时程序是根据指令循环执行次数拼凑出的时间不精确。
- 软件延时程序与单片机硬件系统的晶振有关。
  因此，同一段代码在不同的单片机系统里，延时时间不同。

循环结构举例

【例】在12MHz晶振系统中,将P1.0口接LED发光二极管，使按10Hz频率闪烁。

说明：10Hz周期为100ms，即每50ms变化

```
        ORG     0000H
        LJMP    MAIN
        ORG     0100H
MAIN:   SETB    P1.0        ;LED灭
        MOV     R7,#200     ;延时
DEL11:  MOV     R6,#125
DEL12:  DJNZ    R6,DEL12
        DJNZ    R7,DEL11
        CLR     P1.0        ;LED亮
        MOV     R7,#200     ;延时
DEL21:  MOV     R6,#125
DEL22:  DJNZ    R6,DEL22
        DJNZ    R7,DEL21
        LJMP    MAIN
        END
```

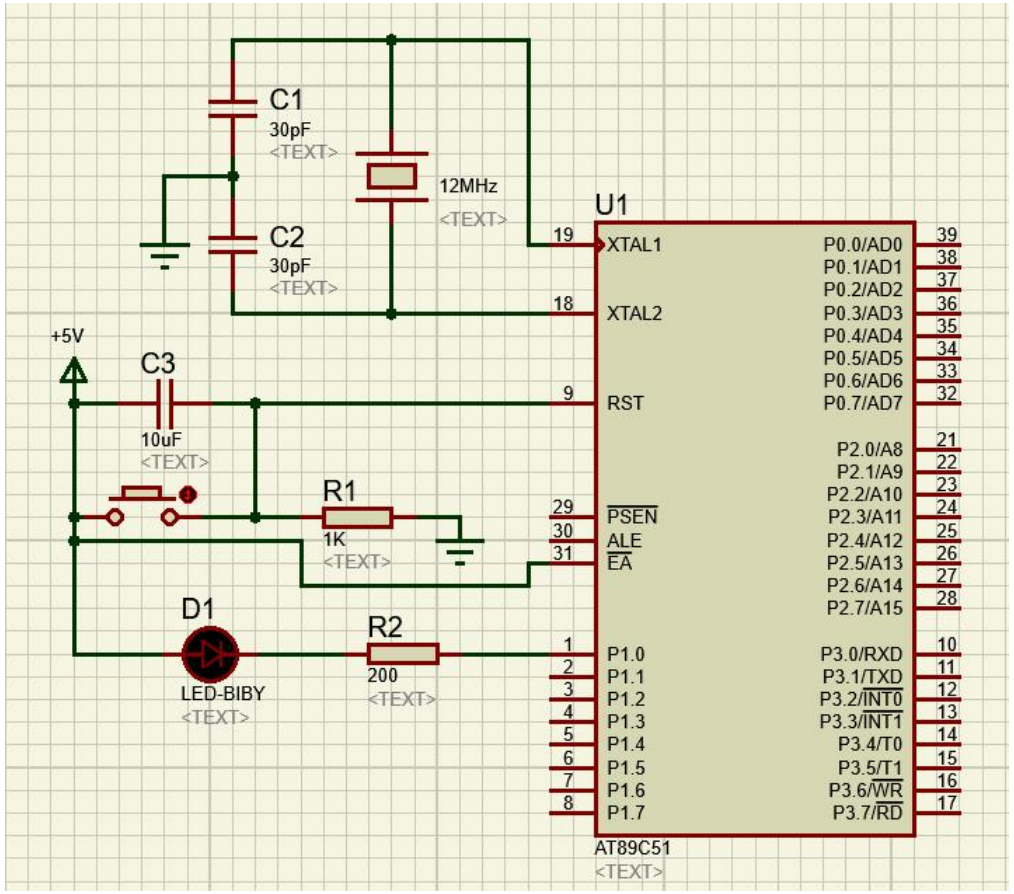


循环结构举例

【例】在12MHz晶振系统中,将P1.0口接LED发光二极管，使按10Hz频率闪烁。

说明：10Hz周期为100ms，即每50ms变化

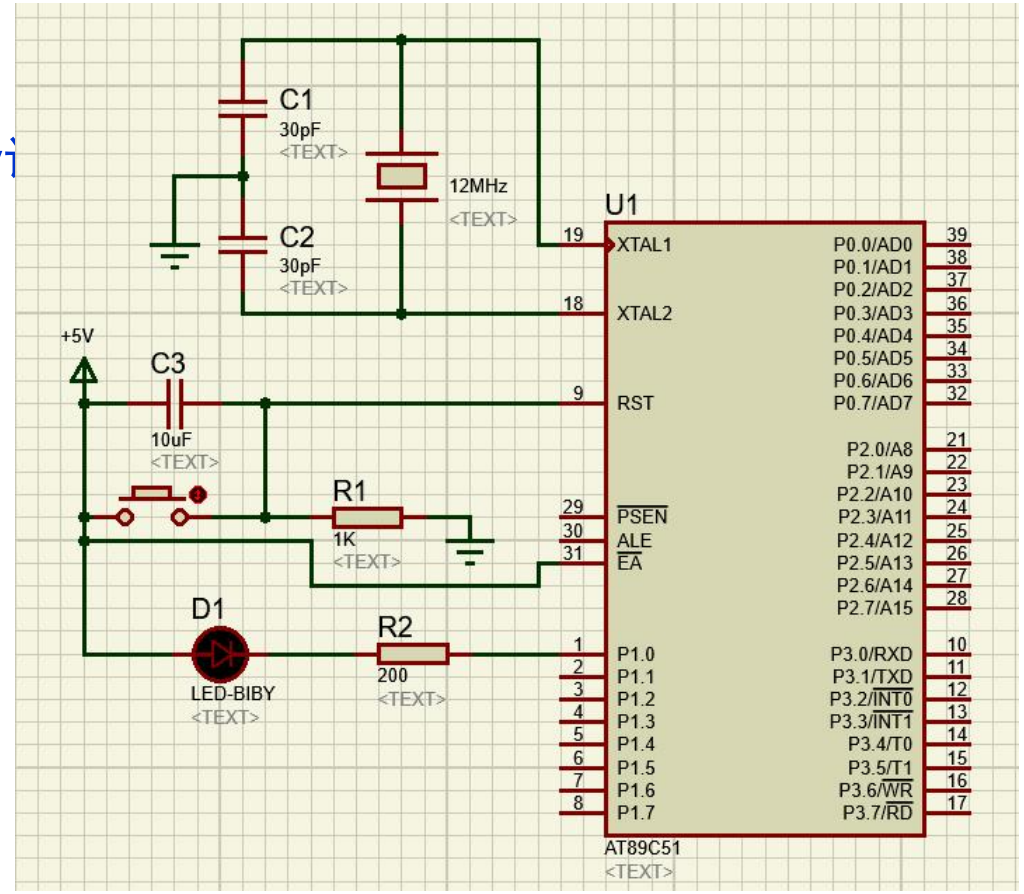
```
#include <reg52.h>
sbit  P10=P1^0;
void  main(void)
{    unsigned int i,j ;
      while(1){
            P10=1;                  //发光二极管灭
            for(i=50; i>0; i--){
               for(j=0;j<333;j++){ ; }
            }
            P10=0;                  //发光二极管亮
            for(i=50; i>0; i--){
               for(j=0;j<333;j++){ ; }
            }
  }
}
```



【例】在12MHz晶振系统中,将P1.0口接LED发光二极管，使按10Hz频率闪烁。

说明：10Hz周期为100ms，即每50ms变化

```
#include <reg52.h>
sbit  P10=P1^0;
void  Delay(unsigned int i){        //延时函数Delay
      unsigned int j;
      for(; i>0; i--){
            for(j=0;j<333;j++){;}
            }
}
void  main(void){
   while(1) {
            P10=1;                  //发光二极管灭
            Delay(50);              //延时50ms
            P10=0;                  //发光二极管亮
            Delay(50);              //延时50ms
     }
 }
```



典型的子程序结构

```
        ORG   0000H
        LJMP  MAIN
        ORG   ****H
MAIN:   ……                      ;主程序
        LCALL LOOP              ;子程序调用
        ……

LOOP:   PUSH      PSW           ;现场保护
        PUSH      Acc
        ……                      ;子程序功能
        POP       Acc           ;现场恢复
        POP       PSW
        RET                     ;子程序返回
        END
```

- 长调用指令　　LCALL　addr16
- 绝对调用指令　　ACALL　addr11

- 长转移指令　　LJMP　addr16
- 绝对转移指令　　AJMP　addr11

注意：LCALL与LJMP的区别？

调用子程序指令

区别：PC在改变前是否进行断点保护？

- **长转移指令　LJMP　addr16**

；机器码

02H	addr8h	addr8l

（1）16位地址是64K范围内的跳转。

（2）先执行 PC+3→PC，

然后　addr16→PC，跳转到新PC处。

- **绝对转移指令　AJMP addr11**

；机器码

A10A9A8 0 0001	A7~A0

（1）11位地址A10～A0是2K范围的跳转。

（2）先执行PC+2→PC，

然后addr11→ PC10～PC0，

PC15～PC11不变，跳转到新PC处。

- **长调用指令　LCALL　addr16**

；机器码

12H	addr8h	addr8l

（1）16位地址是64KB范围内的调用。

（2）先执行 PC+3→PC，即获得断点地址

再执行压栈保护断点

SP+1→SP，PCL→(SP)

SP+1→SP，PCH→(SP)

然后 addr16→PC，转到新PC子程序处

。

- **绝对调用指令　ACALL　addr11**

；机器码

A10A9A8 0 1001	A7~A0

（1）11位地址是2KB范围内的调用。

（2）先执行PC+2→PC，即获得断点地址；

再压栈保护断点：

SP+1→SP，PCL→(SP)；

SP+1→SP，PCH→(SP)；

然后addr11→PC.10～PC.0，

PC15～PC11不变，调用新PC处子程序。

子程序调用

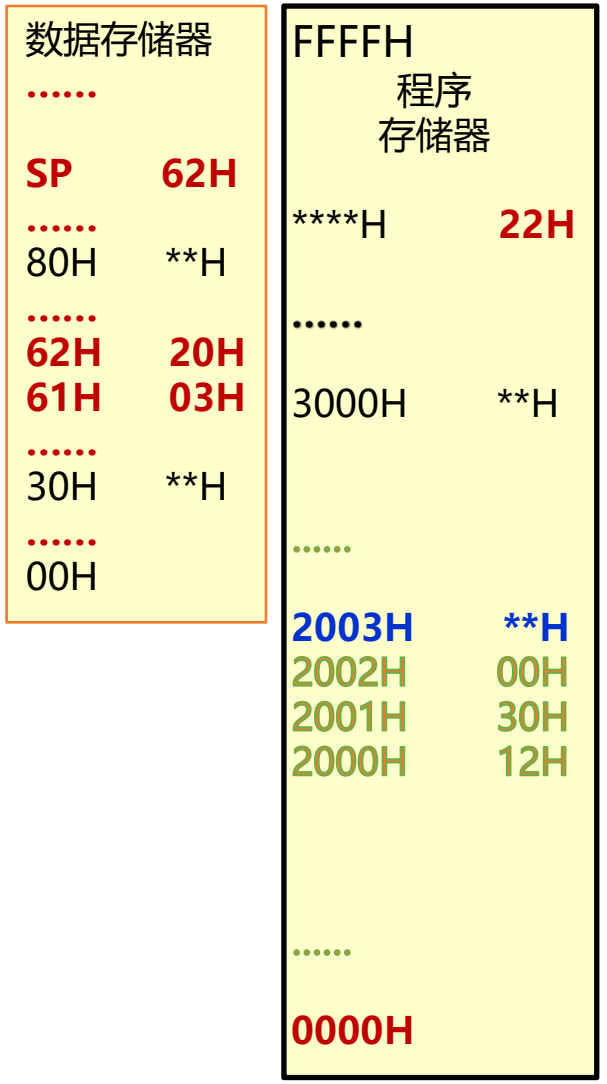
```
例:    ORG   0000H
       LJMP  MAIN
       ORG   ****H
MAIN:  MOV   SP,#60H
       ……
       LCALL LOOP   (假设在2000H处)
       ……           (2003H处)

LOOP:  ……           (假设在3000H处)
       RET
       END
```

1）PC+3=2003H,

2）SP=61H,(61H)=03H, SP=62H, (62H)=20H

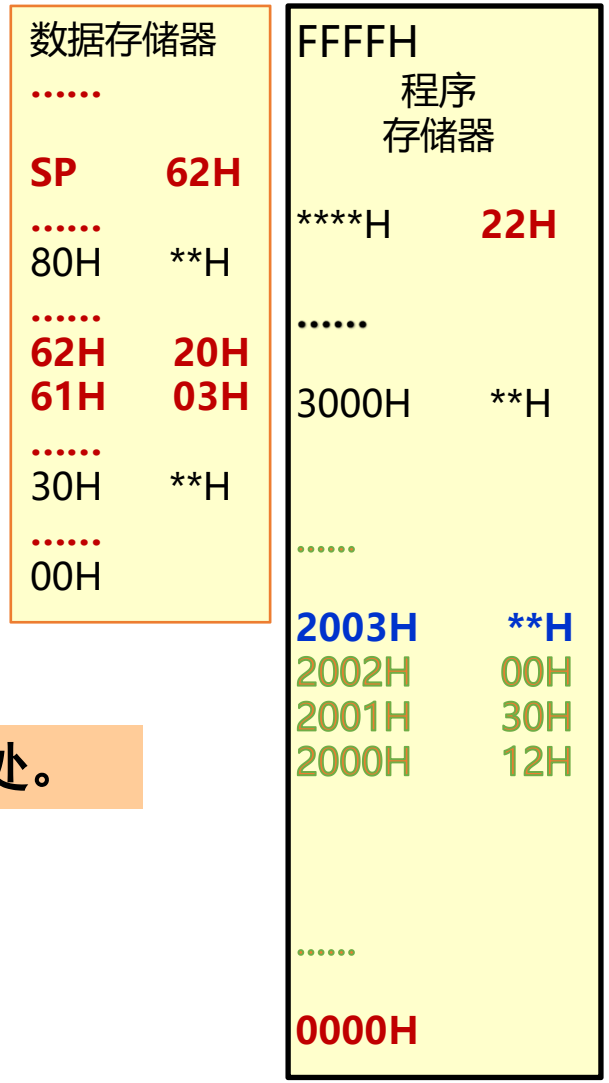
3）PC=3000H

PC由2000H->2003H->3000H



调用返回指令

```
例:     ORG   0000H
        LJMP  MAIN
        ORG   ****H
MAIN:   MOV   SP,#60H
        ……
        LCALL LOOP   (假设在2000H处)
        ……           (2003H处)

LOOP:   ……           (假设在3000H处)
        RET
        END
```



当运行到RET处，2003H出栈，使PC=2003H，返回到断点处。

- **子程序返回指令　RET　;机器码22H**

执行：　(SP)→PCH，(SP)-1→SP；

(SP)→PCL，(SP)-1→SP；

子程序结构举例

【例】将P1.0口接LED发光二极管，使按10Hz频率闪烁。
说明：10Hz周期为100ms，即每50ms变化

```
        ORG     0000H
        LJMP    MAIN
        ORG     0100H
MAIN:   MOV     SP,#60H
        SETB    P1.0  ;LED灭
        LCALL   DELAY
        CLR     P1.0  ;LED亮
        LCALL   DELAY
        LJMP    MAIN
DELAY:  MOV     R7,#200;延时子程序
DEL1:   MOV     R6,#125
DEL2:   DJNZ    R6,DEL2
        DJNZ    R7,DEL1
        RET
        END
```

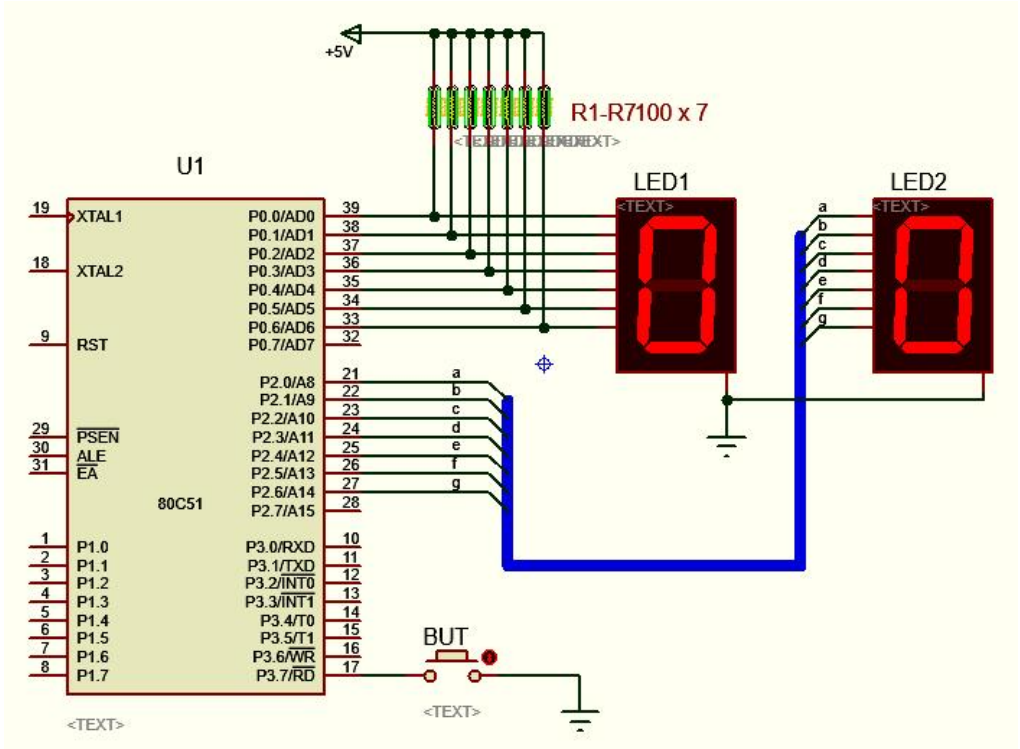
```
        ORG     0000H
        LJMP    MAIN
        ORG     0100H
MAIN:   SETB    P1.0
        MOV     R7,#200
DEL11:  MOV     R6,#125
DEL12:  DJNZ    R6,DEL12
        DJNZ    R7,DEL11
        CLR     P1.0
        MOV     R7,#200
DEL21:  MOV     R6,#125
DEL22:  DJNZ    R6,DEL22
        DJNZ    R7,DEL21
        LJMP    MAIN
        END
```

显示程序——子程序结构

例 统计按键次数并直接显示在P0口所接的数码管上。

设按次数值存放在累加器A中，设统计到3次。

```
        ORG   0000H
        LJMP  MAIN
        ORG   0100H
MAIN:   MOV   A,#03H
        MOV   DPTR,#TAB
        MOVC  A,@A+DPTR
        MOV   P0,A
        SJMP  $
TAB:    DB   3FH,06H,5BH,4FH,66H,6DH,7DH
        DB   07H,7FH,6FH,7FH,7CH,39H,5EH
        DB   79H,71H,00H  ;共阴极段码表
        END
```



显示程序——子程序结构

例 统计按键次数并直接显示在P0口所接的数码管上。

设按次数值存放在累加器A中，设统计到3次。

```
        ORG   0000H
        LJMP  MAIN
        ORG   0100H
MAIN:   MOV   A,#03H
        MOV   DPTR,#TAB
        MOVC  A,@A+DPTR
        MOV   P0,A
        SJMP  $
TAB:    DB  3FH,06H,5BH,4FH,66H,6DH,7DH
        DB  07H,7FH,6FH,7FH,7CH,39H,5EH
        DB  79H,71H,00H  ;共阴极段码表
        END
```

思考：如果去掉PUSH ACC 和POP ACC可以吗？
如果换成ACALL，程序要如何改？

```
        ORG   0000H
        LJMP  MAIN
        ORG   0100H
MAIN:   MOV   A,#03H
        LCALL DISPLAY
        SJMP  $
DISPLAY:PUSH  ACC
        MOV   DPTR,#TAB
        MOVC  A,@A+DPTR
        MOV   P3,A
        POP   ACC
        RET
TAB:    ……  ;共阴极段码表
        END
```

实验举例

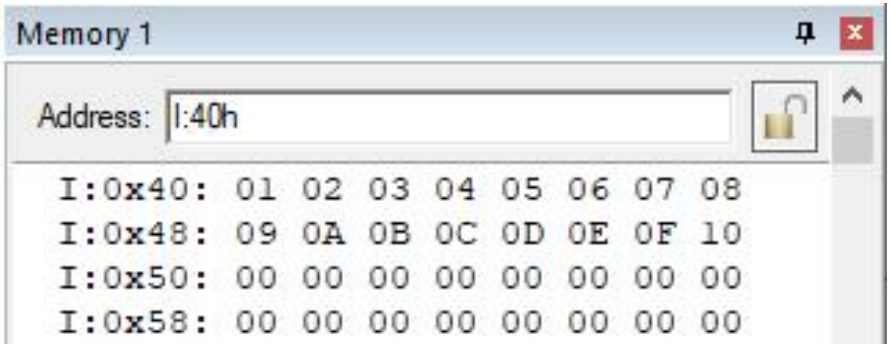
【例3- 24&3-31】设单片机片内数据存储器首地址为40H，片外数据存储器首地址为4000H，存取数据字节个数16个。先将片内数据存储区的这16个字节的内容设置为01H~10H，再将片内首地址为30H开始的16个单元的内容传送到片外首地址为4000H开始的数据存储区中保存。

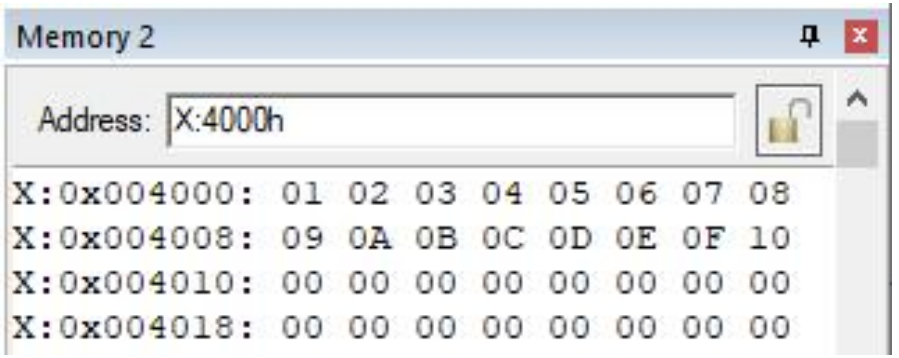
要求：汇编语言实验或C51编程使用绝对地址访问的三种方式之一实现。

①使用指针　　②使用C51运行库中预定义宏　　③使用C51扩展关键字_at_

不同存储空间的数据的查看在Memory窗口输入相应的字母和数字：

查看片内数据存储单元，输入：i:0x40 d:0x40或i:40H d:40H
查看片外数据存储单元，输入：x:0x4000 或x:4000H
查看程序存储单元，输入：c:0x0000 或c:0000H





第4讲-3 控制转移指令和分支循环结构要点

重点

- 控制转移指令中PC的变化
- 调用返回指令中堆栈的变化
- 位操作指令中的控制转移常用判断条件
- 分支结构中每条分支的走向
- 多重循环的循环控制条件和控制变量的使用
- 子程序调用的入口参数、出口参数和现场保护问题