哈爾濱工業大學 远程教育学院

第3章 MCS-51的指令系统

http://www.hitce.net

主要内容

概述

51指令系统的寻址方式

51的指令系统及一般说明

- 数据传送类
- 算数操作类
- 逻辑运算类
- 控制转移类
- 位操作类

3.1 指令系统概述

指令系统：指令的集合。
介绍MCS-51汇编语言的指令系统。
MCS-51的基本指令共111条，按指令所占的字节来分：
(1) 单字节指令49条；
(2) 双字节指令45条；
(3) 三字节指令17条。
按指令的执行时间来分：
(1) 1个机器周期（12个时钟振荡周期）指令64条
(2) 2个机器周期（24个时钟振荡周期）指令45条
(3)只有乘、除两条指令的执行时间为4个机器周期（48个时钟振荡周期）。

12MHz晶振：机器周期为1μs。

指令的长度是单字节并不一定执行起来就是一个机器周期，例如：

单字节双周期 RET　　单字节四周期　乘(MUL)　除（DIV）

双字节单周期 ANL A,#00H

51指令不区分大小写

3.2 指令格式

即指令的表示方法

两部分组成，即操作码和操作数。

操作码用来规定指令进行什么操作

操作数则是指令操作的对象

有单字节指令、双字节指令、三字节不同长度的指令，格式不同：

（1）单字节指令：指令只有一个字节，操作码和操作数同在一个字节中。

（2）双字节指令：一个字节为操作码，另一个字节是操作数。

（3）三字节指令：操作码占一个字节，操作数占二个字节。其中操作数既可能是数据，也可能是地址。

3.3 指令系统的寻址方式

寻址方式就是在指令中说明操作数所在地址的方法。

共7种寻址方式。

1. 寄存器寻址方式

操作数在寄存器中

MOV A，Rn ；（Rn）→A，n=0～7

表示把寄存器Rn的内容传送给累加器A

寻址范围包括：

（1）4组通用工作寄存区共32个工作寄存器。

（2）部分特殊功能寄存器，例如A、B 以及数据指针寄存器DPTR等。

2. 直接寻址方式

操作数直接以单元地址的形式给出：

```
MOV  A，40H
```

寻址范围：

(1) 内部RAM的128个单元

(2) 特殊功能寄存器。除了以单元地址的形式外，还可用寄存器符号的形式给出。例如：

MOV A，80H 与 MOV A，P0是等价的。

3. 寄存器间接寻址方式

寄存器中存放的是操作数的地址

在寄存器的名称前面加前缀标志“@”

访问内部RAM或外部数据存储器的低256个字节时，只能采用R0或R1作为间址寄存器。例如：

```
MOV   R1，#40H
MOV   A，@R1        ；
```

把内部RAM中地址为40H单元内容送A。

寻址范围：

（1）访问内部RAM区，51系列128个字节，52系列256个字节，其通用形式为@Ri

（2）对片外数据存储器的64K字节的间接寻址，例如：

```
MOVX   A，@DPTR
```

（3）片外数据存储器的低256字节，不建议使用。

例如：MOVX　A，@Ri

（4）堆栈区

堆栈操作指令PUSH（压栈）和POP（出栈）使用堆栈指针（SP）作间址寄存器

4．立即寻址方式

操作数在指令中直接给出，需在操作数前面加前缀“#”，若立即数的首位为A~F，前面还要加零。例如：

```
MOV　　A，#40H
MOV　　A，#0FFH
```

5．基址寄存器加变址寄存器间址寻址方式

本寻址方式是以DPTR或PC作基址寄存器，以累加器A作为变址寄存器。

例如：指令 MOVC A，@A+DPTR 其中A的原有内容为05H，DPTR的内容为0400H，该指令执行的结果是把程序存储器0405H单元的内容传送给A。

说明：

（1）本寻址方式是专门针对程序存储器的寻址方式，寻址范围可达到64KB。

（2）本寻址方式的指令只有3条：

```
MOVC    A，@A+DPTR
MOVC    A，@A+PC
JMP     @A+DPTR
```

6．位寻址方式

MCS-51有位处理功能，可以对数据位进行操作，例如：

MOV　C，40H

是把位40H的值送到进位位C。

寻址范围包括：

（1）内部RAM中的位寻址区。位有两种表示方法，例如，40H；另一种是单元地址加上位，例如，(28H).0，指的是28H单元中的最低位。它们是等价的。

（2）特殊功能寄存器中的可寻址位

可寻址位在指令中有如下4种的表示方法：

a. 直接使用位地址。例如PSW.5的位地址为0D5H。

b. 位名称的表示方法。例如：PSW.5是F0标志位，可使用F0表示该位。

c. 单元地址加位数的表示方法。例如 :(0D0H).5。

d. 特殊功能寄存器符号加位数的表示方法。例如:PSW.5。

7. 相对寻址方式

在相对寻址的转移指令中，给出了地址偏移量，以“rel”表示，即把PC的当前值加上偏移量就构成了程序转移的目的地址：

目的地址=转移指令所在的地址 + 转移指令的字节数 + rel

偏移量rel是一带符号的8位二进制数补码数 。

范围是：-128 ~ +127

向地址增加方向最大可转移（127+转移指令字节）个单元地址，向地址减少方向最大可转移（128-转移指令字节）个单元地址。

3.4 MCS-51指令系统分类介绍

111条指令，按功能分类，可分为下面5大类：

（1）数据传送类(28条)

（2）算术操作类(24条)

（3）逻辑运算类(25条)

（4）控制转移类(17条)

（5）位操作类(17条)

指令中符号的意义:

Rn 当前寄存器区的8个工作寄存器R0～R7(n=0～7)。

Ri 当前选中的寄存器区中可作间接寻址寄存器的2个寄存器R0、R1(i=0,1)。

Direct 直接地址，即8位的内部数据存储器单元或特殊功能寄存器的地址。

#data 包含在指令中的8位立即数。

#data16 包含在指令中的16位立即数。

rel 相对转移指令中的偏移量，为8位的带符号补码数

DPTR 数据指针，可用作16位的数据地址寄存器。

bit 内部RAM或特殊功能寄存器中的直接寻址位。

C（或Cy） 进位标志位或位处理机中的累加器。

addr11　11位目的地址

addr16　16位目的地址

@ 间接寻址寄存器前缀，如@Ri，@A+DPTR

(X) X中的内容。

((X)) 由X寻址的单元中的内容。

→ 箭头右边的内容被箭头左边的内容所取代。

3.4.1 数据传送类指令

使用最频繁的一类指令，通用格式：

MOV　<目的操作数>，<源操作数>

属“复制”性质，而不是“搬家”

数据传送类指令不影响标志位，Cy、Ac和OV，但不包括奇偶标志位P。

1．以累加器为目的操作数的指令

```
MOV   A,Rn        ; (Rn)→A，n=0～7
MOV   A,@Ri       ; ((Ri))→A,i=0,1
MOV   A,direct    ; (direct) →A
MOV   A,#data     ; #data→A
```

例如：

```
MOV    A,R6       ;(R6)→A，寄存器寻址
MOV    A,70H      ;(70H)→A，直接寻址
MOV    A,@R0      ;((R0))→A，间接寻址
MOV    A,#78H     ;78H→A，立即寻址
```

2. 以Rn为目的操作数的指令

```
MOV   Rn,A          ; (A)→Rn,n=0～7
MOV   Rn,direct     ;（direct）→Rn,n=0～7
MOV   Rn,#data      ; #data→Rn,n=0～7
```

功能：是把源操作数的内容送入当前一组工作寄存器区的R0～R7中的某一个寄存器。

3.以直接地址direct为目的操作数的指令

```
MOV   direct,A ; (A)→direct
MOV   direct,Rn; (Rn)→direct, n=0～7
MOV   direct1,direct2;
MOV   direct,@Ri ; ((Ri))→direct
MOV   direct,#data; #data→direct
```

功能：把源操作数送入直接地址指出的存储单元。direct指的是内部RAM或SFR的地址。

4.以寄存器间接地址为目的操作数的指令

```
MOV    @Ri,A          ;(A)→((Ri)),i=0,1
MOV    @Ri,direct     ;  (direct) →((Ri))
MOV    @Ri,#data      ; #data→((Ri))
```

5.16位数传送指令

```
MOV    DPTR,#data16     ; #data16→DPTR
```

唯一的16位数据的传送指令，立即数的高8位送入DPH，立即数的低8位送入DPL。

6. 堆栈操作指令

MCS-51内部RAM中可以设定一个后进先出（LIFO-Last In First Out）的区域称作堆栈.

堆栈指针SP指出堆栈的栈顶位置。

(1)进栈指令

PUSH direct

先将栈指针SP加1，然后把direct中的内容送到栈指针SP指示的内部RAM单元中。

例如： 当（SP）=60H,（A）=30H,（B）=70H时，

执行：

```
PUSH   Acc    ; (SP)+1=61H→SP,(A)→61H
PUSH   B      ;
```

(SP)+1=62H→SP,(B)→62H

结果：(61H)=30H,(62H)=70H,(SP)=62H

(2) 出栈指令

POP direct

SP指示的栈顶（内部RAM单元）内容送入direct字节单元中，栈指针SP减1.

例如： 当 (SP)=62H，(62H)=70H，(61H)=30H，

执行：

POP DPH ;((SP))→DPH,(SP)-1→SP

POP DPL ;((SP))→DPL,(SP)-1→SP

结果：(DPTR)=7030H，(SP)=60H

不遵循先进后出原则的问题

```
mov   SP，#60H
MOV   A，#0AAH
MOV   B，#00H
PUSH  A                （61H）为 #0AAH
PUSH  B                （62H）为 #00H
POP    A                A      为#00H
POP    B                B      为#0AAH
```

7. 累加器A与外部数据存储器传送指令

```
MOVX   A,@DPTR  ;((DPTR))→A,读外部RAM/IO
MOVX   A,@Ri    ;((Ri))→A,读外部RAM/IO
MOVX  @DPTR,A;(A)→((DPTR)),写外部RAM/IO
MOVX  @Ri,A     ;(A)→((Ri)),写外部RAM/IO
```

功能：读外部RAM存储器或I/O中的一个字节，或把A中一个字节的数据写到外部RAM存储器或I/O中。

注意：RD*或WR*信号有效。

采用DPTR间接寻址，高8位地址（DPH）由P2口输出，低8位地址（DPL）由P0口输出。

采用Ri（i=0,1）间接寻址，可寻址片外256个单元的数据存储器。Ri内容由P0口输出。

8位地址和数据均由P0口输出，可选用其它任何输出口线来输出高于8位的地址（一般选用P2口输出高8位的地址）。

MOV后“X”表示单片机访问的是片外RAM存储器或I/O。

8. 查表指令

共两条，用于读程序存储器中的数据表格的指令，均采用基址寄存器加变址寄存器间接寻址方式。

（1） MOVC　　A,@A+PC

以PC作基址寄存器，A的内容作为无符号整数和PC中的内容（下一条指令的起始地址）相加后得到一个16位的地址，该地址指出的程序存储单元的内容送到累加器A。

注意：PSEN*信号有效。

例如：（A)=30H, 执行地址1000H处的指令

1000H：MOVC　　A, @A+PC

本指令占用一个字节，执行结果将程序存储器中1031H的内容送入A。

优点：不改变特殊功能寄存器及PC的状态，根据A的内容就可以取出表格中的常数。

缺点：表格只能存放在该条查表指令后面的256个单元之内，表格的大小受到限制，且表格只能被一段程序所利用。

（2）MOVC　　A, @A+DPTR

以DPTR作为基址寄存器，A的内容作为无符号数和DPTR的内容相加得到一个16位的地址，把由该地址指出的程序存储器单元的内容送到累加器A.

例如　(DPTR)=8100H　(A)=40H　执行指令

MOVC　A,@A+DPTR

本指令的执行结果只和指针DPTR及累加器A的内容有关，与该指令存放的地址及常数表格存放的地址无关，因此表格的大小和位置可以在64K程序存储器中任意安排，一个表格可以为各个程序块公用。

两条指令是在MOV的后面加C，“C”是CODE的第一个字母，即代码的意思。

9．字节交换指令

XCH　A,Rn

XCH　A,direct

XCH　A,@Ri

例如：

(A)=80H，(R7)=08H，(40H)=F0H

(R0)=30H，(30H)=0FH

执行下列指令：

```
XCH    A,R7        ;(A)与(R7)互换
XCH    A,40H       ;(A)与(40H)互换
XCH    A,@R0       ;(A)与((R0))互换
```

结果：(A)=0FH,(R7)=80H,(40H)=08H,(30H)=F0H

10．半字节交换指令

```
XCHD    A,@Ri
```

累加器的低4位与内部RAM低4位交换。例如：

(R0)=60H,(60H)=3EH,(A)=59H

执行完 XCHD A,@R0 指令，

则(A)=5EH,(60H)=39H。

3.4.2 算术操作类指令

单字节的加、减、乘、除法指令，都是针对8位二进制无符号数。

执行的结果对Cy、Ac、OV 三种标志位有影响。

但增1和减1指令不影响上述标志。

1．加法指令

共有4条加法运算指令：

```
ADD   A,Rn          ;(A)+(Rn)→A，n=0～7
ADD   A,direct      ;(A)+(direct)→A
ADD   A,@Ri         ;(A)+((Ri))→A,i=0,1
ADD    A,#data   ;  (A)+#data→A
```

一个加数总是来自累加器A，而另一个加数可由不同的寻址方式得到。结果总是放在A中。

使用加法指令时，要注意累加器A中的运算结果对各个标志位的影响：

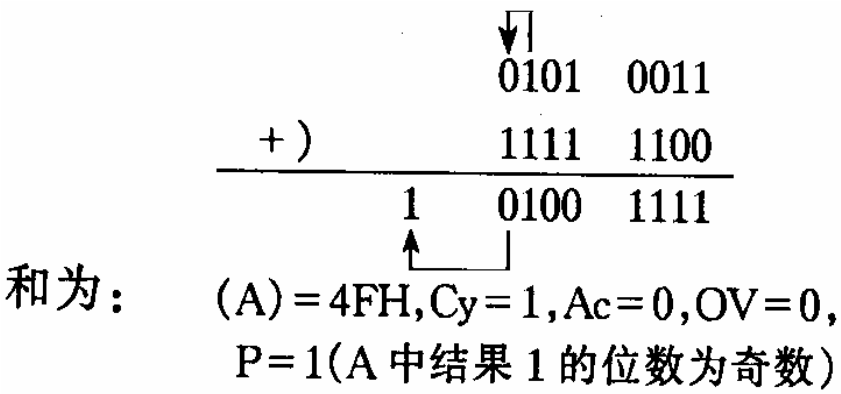
（1）如果位7有进位，则置“1”进位标志Cy，否则清“0”Cy

（2）如果位3有进位，置“1”辅助进位标志Ac，否则清“0”Ac（Ac为PSW寄存器中的一位）

（3）如果位6有进位，而位7没有进位，或者位7有进位，而位6没有，则溢出标志位OV置“1”，否则清“0”OV。

例 (A)=53H，(R0)=FCH，执行指令

ADD A, R0



注意：上面的运算中，由于位6和位7同时有进位，所以标志位OV=0。

例　(A)=85H,(R0)=20H,(20H)=AFH,执行指令：

ADD　A,@R0

结果为：

$$\begin{array}{rcc} & 1000 & 0101 \\ +) & 1010 & 1111 \\ \hline 1 & 0011 & 0100 \end{array}$$

和为：

(A)=34H,Cy=1,Ac=1,OV=1,P=1

注意：由于位7有进位，而位6无进位，所以标志位**OV=1**

2. 带进位加法指令

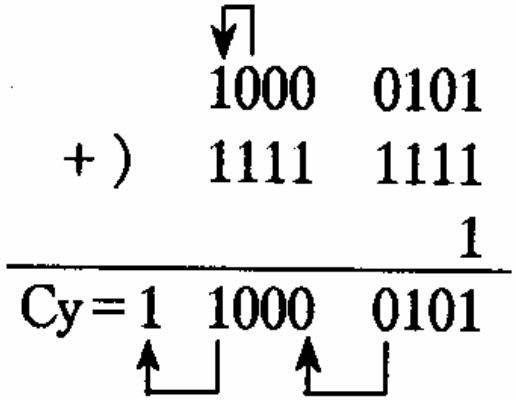
标志位Cy参加运算，因此是三个数相加。共4条：

```
ADDC    A,Rn        ;(A)+(Rn)+C→A，  n=0～7
ADDC    A,direct    ;(A)+(direct)+C→A
ADDC    A,@Ri       ;(A)+(Ri)+C→A，  i=0,1
ADDC    A,#data     ;(A)+#data+C→A
```

例：　（A）=85H,　(20H)=FFH,　Cy=1

执行指令：　ADDC　A,20H

结果为：



和为：

(A)=85H,Cy=1,Ac=1,OV=0,

P=1（A 中 1 的位数为奇数）

3. 增1指令

5条增1指令：

```
INC   A
INC   Rn          ;n=0～7
INC   direct
INC   @Ri         ;i=0,1
INC   DPTR
```

不影响PSW中的任何标志，若原变量中内容为**FFH**则程序执行后变量变为**00H**。

第5条指令INC DPTR，是16位数增1指令。指令首先对低8位指针DPL的内容执行加1的操作，当产生溢出时，就对DPH的内容进行加1操作，并不影响标志Cy的状态。

4. 十进制调整指令

用于对BCD码十进制数加法运算结果的内容修正。

指令格式：　　DA　A

两个压缩BCD码的数按二进制相加之后，必须经本指令的调整才能得到正确的和数（仍为压缩BCD码表示）。

应用背景：

（1）该指令执行前，一般有一条加法指令。

（2）加法指令中的两个加数，应该是用压缩BCD码表示的十进制数，和存放在A中。

（3）执行完DA指令后，A中存放的数是两个加数的十进制和，也使用压缩BCD码表示。

若(A)=56h (R5)=67h

ADD A,R5

DA A

结果（A）=23；Cy=1

5. 带借位的减法指令

4条指令：

```
SUBB   A,Rn     ;     （A）-（Rn）- Cy→A，n=0～7
SUBB   A,direct ;     （A）-（direct）- Cy→A
SUBB   A,@Ri;         （A）-（(Ri)）- Cy→A，  i=0,1
SUBB   A,#data  ;     （A）-#data - Cy→A
```

从累加器A中的内容减去指定的变量和进位标志Cy的值，结果存在累加器A中。

如果位7需借位则置“1” Cy，否则清“0”Cy；

如果位3需借位则置“1”Ac，否则清“0”Ac；

如果位6需借位而位7不需要借位，或者位7需借位，位6不需借位，则置“1”溢出标志位OV，否则清“0”OV。

例　(A)＝C9H (R2)＝54H,Cy＝1,执行指令

SUBB　A,R2

结果为

$$
\begin{array}{rr}
 & \downarrow \\
 & 1100\ \ 1001 \\
 & 0101\ \ 0100 \\
-) & \qquad\quad 1 \\
\hline
 & 0111\ \ 0100
\end{array}
$$

差为　(A)＝74H,Cy＝0,Ac＝0,
OV＝1(位6向位7借位)

6．减1指令

4条指令：

```
DEC   A          ；(A)-1→A
DEC   Rn         ；(Rn)-1→Rn，n=0～7
DEC   direct     ；(direct)-1→direct
DEC   @Ri        ；((Ri))-1→（Ri），i=0,1
```

减1指令不影响标志位。

7．乘法指令

```
MUL   AB           ；A×B→BA
```

如果积大于255，则置“1”溢出标志位OV

8. 除法指令

DIV　AB　　；A/B→A（商），余数→B

如果B的内容为“0”（即除数为“0”），则存放结果的A、B中的内容不定，并置“1”溢出标志位OV。

3.4.3 逻辑运算指令

1. 简单逻辑操作指令

（1）CLR　A

功能是累加器A清“0”。不影响Cy、Ac、OV等标志。

（2）CPL　A

功能是将累加器A的内容按位逻辑取反，不影响标志。

2．左环移指令

RL　A

功能是累加器A的8位向左循环移位，位7循环移入位0，不影响标志。

3．带进位左环移指令

RLC　A

功能是将累加器A的内容和进位标志位Cy一起向左环移一位，Acc.7移入进位位Cy，Cy移入Acc.0，不影响其它标志。

4．右环移指令

RR　A

功能是累加器A的内容向右环移一位，Acc.0移入Acc.7，不影响其它标志。

5．带进位环移指令

RRC　A

这条指令的功能是累加器A的内容和进位标志Cy一起向右环移一位，Acc.0进入Cy，Cy移入Acc.7。

6．累加器半字节交换指令

SWAP　A

将累加器A的高半字节（Acc.7～Acc.4）和低半字节（Acc.3～Acc.0）互换。

例　（A）=0C5H，执行指令：

SWAP　A

结果：（A）=5CH

7．逻辑与指令

```
ANL   A,Rn      ;  (A) ∧ (Rn) →A, n=0～7
ANL   A,direct  ;  (A) ∧ (direct) →A
ANL   A,#data   ;  (A) ∧#data→A
ANL   A,@Ri     ;  (A) ∧( (Ri) )→A, i=0～1
ANL   direct,A  ;  (direct) ∧(A)→direct
ANL   direct,#data
                ;  (direct) ∧#data→direct
```

运算结果存入第一操作数中，

例 (A)=07H, (R0)=0FDH,执行指令：

```
ANL A,R0
```

(A)= 00000111

(R0)=11111101

结果：(A)= 00000101=05H

8. 逻辑或指令

```
ORL A,Rn    ; (A)∨(Rn)→A , n=0～7
ORL A,direct  ; (A)∨(direct)→A
ORL A,#data   ; (A)∨ data→A
ORL A,@Ri ;  (A)∨((Ri))→A, i=0,1
ORL direct,A  ; (direct)∨(A)→direct
ORL direct,#data
              ; (direct)∨#data→direct
```

例 （P1）=05H，（A）=33H，执行指令

```
ORL   P1,A
```

（P1） =00000101

（A） =00110011

结果：（P1）=00110111=37H

9. 逻辑异或指令

```
XRL   A,Rn          ; (A) ⊕(Rn)→A
XRL   A,direct      ; (A) ⊕ (direct) →A
XRL   A,@Ri
                    ; (A)⊕( (Ri) )→A ,i=0,1
XRL   A,#data       ; (A)⊕#data→A
XRL   direct,A
                  ; (direct) ⊕ (A) →direct
XRL   direct,#data
                  ; (direct) ⊕ #data →direct
```

例　（A）=90H，（R3）=73H 执行指令：

XRL　A,R3

（A）= 10010000

（R3）=01110011

结果：（A）=11100011=E3H

3.4.4 控制转移类指令

1. 无条件转移指令

AJMP　addr11

2K字节范围内的无条件跳转指令，　64K程序存储器空间分为32个区，每区2K字节，转移的目标地址必须与AJMP下一条指令的地址（pc）的高5位地址码A15~A11相同。

**本指令是为能与MCS-48的JMP指令兼容而设的。不建议使用

2. 长跳转指令

LJMP addr16

指令执行时把指令的第二和第三字节分别装入PC的高位和低位字节中，无条件地转向addr16指出的目标地址。目标地址可以在64K程序存储器地址空间的任何位置。

3. 相对转移指令

SJMP rel

在编写程序时，直接写上要转向的目标地址标号就可以,由汇编程序自动计算和填入偏移量。

跳转目标地址处于当前PC值的

-128字节-- +127字节之间

例如：

```
LOOP:   MOV   A, R6
          ⋮
        SJMP  LOOP
          ⋮
```

跳转目的地址超过-128~127区间的处理

Sjmp loop

超过127字节

Loop:

编译会出错

Sjmp loop1

Ljmp loop

Loop:

编译正确

4. 间接跳转指令

```
JMP    @A+DPTR
```

由A中8位无符号数与DPTR的16位数内容之和来确定。以DPTR内容作为基址，A的内容作变址。

给A赋予不同的值，即可实现程序的多分支转移。

5. 条件转移指令

```
JZ    rel   ；如果累加器为“0”，则转移
JNZ   rel   ；如果累加器非“0”，则转移
```

规定的条件满足，则进行转移，跳转到相应标号处。跳转目的地址的要求同SJMP。

条件不满足则顺序执行下一条指令。

6. 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
```

比较前面两个操作数的大小，如果它们的值不相等则转移，转移的目的地址要求同SJMP。

**如果第一操作数（无符号整数）小于第二操作数（无符号整数），则置进位标志位Cy，否则清“0”Cy。

7. 减1不为0转移指令

这是一组把减1与条件转移两种功能结合在一起的指令。共两条指令：

```
DJNZ    Rn,rel        ;n=0～7
DJNZ    direct,rel
```

将源操作数（Rn或direct）减1，结果回送到Rn寄存器或direct中去。如果结果不为0则转移，转移的目的地址要求同SJMP。

**允许程序员把寄存器Rn或内部RAM的direct单元用作程序循环计数器。主要用于控制程序循环。以减1后是否为“0”作为转移条件，即可实现按次数控制循环。

8. 调用子程序指令

(1)短调用指令

ACALL addr11

与AJMP指令相类似，是为了与MCS-48中的CALL指令兼容而设的，不建议使用。

(2)长调用指令

LCALL addr16

该指令执行时，MCS51执行如下操作：

（1）当前PCL、PCH进栈

（2）addr 16 送入PC

9. 子程序的返回指令

RET

执行本指令时：

（SP）→PCH，然后（SP）-1→SP

（SP）→PCL，然后（SP）-1→SP

功能是从堆栈中退出PC的高8位和低8位字节，把栈指针减2，从PC值开始继续执行程序。

10. 中断返回指令

RETI

功能与RET指令相似，两指令不同之处，是本指令清除了中断响应时，被置“1”的MCS-51内部中断优先级寄存器的优先级状态。

11. 空操作指令

NOP

3.4.5 位操作指令

1.数据位传送指令

```
MOV   C,bit
MOV   bit,C
```

例 `MOV C,06H` ；(20H).6→Cy

06H是内部RAM 20H字节位6的位地址。

```
MOV   P1.0,C   ；Cy→P1.0
```

2. 位变量修改指令

```
CLR   C        ；清“0”Cy
CLR   bit      ；清“0”bit位
CPL   C        ；Cy求反
```

```
CPL   bit          ; bit位求反
SETB  C            ; 置“1” Cy
SETB  bit          ; 置“1” bit位
```

这组指令将操作数指出的位清“0”、求反、置“1”，不影响其它标志。

```
例    CLR   C      ; 0→Cy
      CLR   27H    ; 0→（24H）.7位
      CPL   08H    ; →（21H）.0位
      SETB  P1.7   ; 1→P1.7位
```

3. 位变量逻辑与指令

```
ANL   C,bit        ; bit∧Cy→Cy
ANL   C,/bit;      ; /bit ∧Cy→Cy
```

4. 位变量逻辑或指令

```
ORL    C,bit
ORL    C,/bit
```

5. 条件转移类指令

```
JC     rel          ; 如果进位位Cy=1，则转移
JNC    rel          ; 如果进位位Cy=0，则转移
JB     bit,rel      ; 如果直接寻址位=1，
                      则转移
JNB    bit,rel      ; 如果直接寻址位=0，
                      则转移
JBC    bit,rel      ; 如果直接寻址位=1，
                    则转移，并清0直接寻址位
```

下表列出了按指令功能排列的全部指令及功能的简要说明，以及指令长度、执行的时间以及指令代码（机器代码）。

读者可根据指令助记符，迅速查到对应的指令代码（手工汇编）。也可根据指令代码迅速查到对应的指令助记符（手工反汇编）。

应熟练地掌握这些指令表的使用，因为这是使用MCS-51汇编语言进行程序设计的基础。

按功能排列的指令表

一、数据传送类

助记符	说　　明	字节数	执行时间（机器周期）	指令代码（机器代码）
MOV　A,Rn	寄存器内容传送到累加器A	1	1	E8H～EFH
MOV　A,direct	直接寻址字节传送到累加器	2	1	E5H,direct
MOV　A,@Ri	间接寻址RAM传送到累加器	1	1	E6H～E7H
MOV　A,#data	立即数传送到累加器	2	1	74H,data
MOV　Rn,A	累加器内容传送到寄存器	1	1	F8H～FFH
MOV　Rn,direct	直接寻址字节传送到寄存器	2	2	A8H～AFH,direct
MOV　Rn,#data	立即数传送到寄存器	2	1	78H～7FH,data
MOV　direct,A	累加器内容传送到直接寻址字节	2	1	F5H,direct
MOV　direct,Rn	寄存器内容传送到直接寻址字节	2	2	88H～8FH,direct
MOV　direct1,direct2	直接寻址字节2传送到直接寻址字节1	3	2	85H,direct2,direct1
MOV　direct,@Ri	间接寻址RAM传送到直接寻址字节	2	2	86H～87H
MOV　direct,#data	立即数传送到直接寻址字节	3	2	75H,direct,data
MOV　@Ri,A	累加器传送到间接寻址RAM	1	1	F6H～F7H
MOV　@Ri,direct	直接寻址字节传送到间接寻址RAM	2	2	A6H～A7H,direct

讨 论

♣ 如何对sfr中的内容进行读写？

Mov a,80h；直接寻址方式访问

或者 mov a,p0；直接使用SFR的名称。

♣♣ 如何对52系列单片机的高128字节RAM进行读写？

Mov r0,#80h

Mov a,@r0；间接寻址方式访问

♣♣♣ 如何读出片外数据存储器字节地址为2000H中的内容？

Mov dptr,#2000h

Movx a,@dptr

习题

第3章（共15题）

1．判断以下指令的正误：

（1）MOV 28H，@R2 （2）DEC DPTR （3）INC DPTR （4）CLR R0

（5）CPL R5 （6）MOV R0，R1 （7）PHSH DPTR （8）MOV F0，C

（9）MOV F0，Acc.3 （10）MOVX A，@R1 （11）MOV C，30H （12）RLC R0

答：（1）× （2）× （3）√（4）×（5）× （6）× （7）× （8）√（9）×
（10）√ （11）√ （12）×

2．判断下列说法是否正确。

（A）立即寻址方式是被操作的数据本身在指令中，而不是它的地址在指令中。

（B）指令周期是执行一条指令的时间。

（C）指令中直接给出的操作数称为直接寻址。

答：（A）√ （B）√ （C）×

3．在基址加变址寻址方式中，以（ ）作变址寄存器，以（ ）或（ ）作基址寄存器。

答：A，DPTR或PC

4．MCS-51共有哪几种寻址方式？各有什么特点？

答：共有7种寻址方式，分别是直接寻址、寄存器寻址、寄存器间接寻址、立即寻址、基址加变址、位寻址、相对寻址

5．MCS-51指令按功能可以分为哪几类？每类指令的作用是什么？

答：分为5类，分别是数据传送类、算术操作类、逻辑运算类、控制转移类、位操作类

6．访问SFR，可使用哪些寻址方式？

答：直接寻址方式是访问特殊功能寄存器的唯一寻址方式。

7．指令格式是由（　　）和（　　）所组成，也可能仅由（　　）组成。

答：操作码，操作数，操作码

8. 假定累加器A中的内容为30H，执行指令：

1000H：MOVC A，@A+PC

后，把程序存储器（　　）单元的内容送入累加器A中。

答：1031H

9．在MCS-51中，PC和DPTR都用于提供地址，但PC是为访问（　　）存储器提供地址，而DPTR是为访问（　　）存储器提供地址。

答：程序存储器，数据存储器

10．在寄存器间接寻址方式中，其“间接”体现在指令中寄存器的内容不是操作数，而是操作数的（　　）。

答：地址

11．下列程序段的功能是什么？

```
PUSH  Acc
PUSH  B
POP   Acc
POP   B
```

答：A和B内容互换

12．写出完成如下要求的指令，但是不能改变未涉及位的内容。

（A）把ACC.3， ACC.4，ACC.5和 ACC.6清“0”。

（B）把累加器A的中间4位清“0”。

（C）使ACC.2和 ACC.3置“1”。

答：　　（A）ANL A,#87H

　　（B）ANL A,#C3H

　　（C）ORL A,#0CH

13．假设**A=55H，R3=0AAH**，在执行指令**ANL A,R5**后，**A=（　），R3=（　）**。

答：A=（00H），R3=（0AAH）。

14．如果**DPTR=507BH，SP=32H，（30H）=50H，（31H）=5FH，（32H）=3CH**，则执行下列指令后：

POP DPH

POP DPL

则：**DPH=（　　），DPL=（　　）**

答：DPH=（3CH），DPL=（5FH）

15．假定，**SP=60H，A=30H，B=70H**，执行下列指令：

PUSH Acc

PUSH B

后，**SP**的内容为（　　），**61H**单元的内容为（　　），**62H**单元的内容为（　　）。

答：SP的内容为（62H），61H单元的内容为（30H），62H单元的内容为（70H）。

谢谢大家