微机原理和接口技术

第五讲 指令系统与汇编程序2

提 纲

1. 指令系统概述
2. 寻址方式
3. 数据传送类指令
4. 算术运算类指令
5. 逻辑运算类指令
6. 控制转移类指令
7. 位操作指令
8. 查表指令的应用
9. 堆栈操作指令的应用
10. 十进制调整指令的应用
11. 逻辑指令与字节状态操作
12. 转移指令的应用

提 纲

3. 数据传送类指令

数据传送类指令

数据传送类指令是最基本、使用最多的一类指令，共有29条。主要用于数据的传送、保存以及交换等场合。

该类指令中，除给A赋值（A内容发生变化）会影响P标志外，其余标志不受影响。可分为以下五组：

- 内部RAM数据传送类指令：　16条（MOV）
- 程序存储器访问类指令：　2条（MOVC)
- 外部RAM访问类指令：　4条（MOVX）
- 堆栈操作类指令：　2条
- 数据交换类指令：　5条

数据传送类指令

1. 内部RAM数据传送指令(16条)：（助记符：MOV（MOVe））

该组指令实现8051MCU内部工作寄存器、存储单元、SFR之间的数据传送。

指令格式： MOV <目的操作数>, <源操作数>

根据指令中4种不同的目的操作数（A、Rn、direct和@Ri）分别讨论如下：

- **（1）以累加器A为目的操作数的指令(4条)**

  MOV A, Rn ;(A) ← (Rn)

  MOV A, direct ;(A) ← (direct)

  MOV A, @Ri ;(A) ← ((Ri))

  MOV A, #data ;(A) ← data

➢ **功能：**把源操作数指定的内容送入A中，**即A的赋值指令。**

数据传送类指令

1. 内部RAM数据传送指令(16条)：（助记符：MOV （MOVe ））

- **（2）以直接地址为目的操作数的指令（5条）**

```
MOV direct, A          ; (direct) ← (A)
MOV direct, Rn         ; (direct) ← (Rn)
MOV direct1, direct2   ; (direct1) ← (direct2)
MOV direct, @Ri        ; (direct) ← ((Ri))
MOV direct, #data      ; (direct) ← data
```

➢ **功能：**把源操作数送到内部RAM的direct单元，即**内存单元赋值指令。**

- **（3）以寄存器Rn为目的操作数的指令(3条)**

```
MOV  Rn, A             ;(Rn) ← (A)
MOV  Rn, direct        ;(Rn) ← (direct)
MOV  Rn, #data         ;(Rn) ← data
```

➢ **功能：**把源操作数送到工作寄存器Rn中，即**工作寄存器赋值指令。**

数据传送类指令

1. 内部RAM数据传送指令(16条)：（助记符：MOV（MOVe））

- **（4）以间接地址为目的操作数的指令（3条）**

```
MOV @Ri,A          ;((Ri)) ← (A)
MOV @Ri,direct     ;((Ri)) ← (direct)
MOV @Ri,#data      ;((Ri)) ← data
```

➢ **功能：**把源操作数送到以Ri的内容为地址的内部RAM中，给Ri间接寻址的内存单元赋值。

例：（A）＝08H，（R1）＝10H，执行MOV @R1, A后，10H单元的内容变为08H，而R1的内容仍为10H。

数据传送类指令

1. 内部RAM数据传送指令(16条)：（助记符：MOV（MOVe ））

- **（5） 16位立即数传送指令（1条）**

MOV DPTR,#data16 ;(DPTR) ← data16

➢ **功能：**将一个16位的立即数送入DPTR，其中高8位送入DPH，低8位送入DPL。

例：MOV DPTR，#1234H 的执行结果，与执行下面2条指令的结果相同。

MOV DPH，#12H

MOV DPL，#34H。

数据传送类指令

2.外部RAM访问指令 (4条)： （助记符：MOVX（MOVe eXternal RAM））

该组指令实现8051MCU与外部RAM的数据传送。

MOVX A, @DPTR	;(A) ← ((DPTR))
MOVX A, @Ri	;(A) ← ((Ri))
MOVX @DPTR, A	;((DPTR)) ← (A)
MOVX @Ri, A	;((Ri)) ← (A)

- **功能：**对外部RAM进行读或写操作，采用寄存器间接寻址方式。前2条为读指令，后2条是写指令。

- **注意：**

- 对外部RAM的读写必须通过A累加器。（对内部RAM的读写，指令丰富）
- 对外部RAM的读写，只能用寄存器间接寻址方式，用DPTR或Ri作为地址指针。（对内部RAM操作，有多种寻址方式）

数据传送类指令

问题：如果想把外部RAM的数据导入内部RAM或者SFR中，应该怎么做？

数据传送类指令

问题：如果想把外部**RAM**的数据导入内部**RAM**或者**SFR**中，应该怎么做？

答：应该利用累加器**A**为跳板；

（1） MOVX A, @DPTR ;(A) ← ((DPTR))

外部RAM64K地址中的内容 到 A

MOVX A, @Ri ;(A) ← ((Ri))

外部RAM256B地址中的内容 到 A

（2）MOV direct, A ; (direct) ← (A)

A 到 内部RAM 00H~7FH地址中 或者 SFR中

MOV @Ri,A ;((Ri)) ← (A)

A 到 内部RAM 00H~FFH地址中

3. 查表指令（2条）：（助记符：MOVC（MOVe Code））

该组指令的功能是从ROM中读取数据，通常是对存放在ROM中的数据表格进行查找读取。

（1）远程查表指令

MOVC A, @A+DPTR ； (A)← ((A)+(DPTR))

- **功能：**将DPTR的内容与A的内容相加后形成一个ROM单元地址，将该ROM单元的内容送至A。DPTR内容不变。

优点：可以查找存放在64KROM中任何地址的数据表格，因此称为**远程查表指令。**

缺点：要占用DPTR寄存器。

数据传送类指令

（2）近程查表指令：

MOVC A, @A+PC ；（PC）←（PC）+1

；（A）←（（A）+（PC））

- **功能：**将A和当前PC值相加，形成要寻址的ROM单元地址，将该ROM单元中的内容送到A。

注意：当前PC值，应为该指令所在地址加1。

优点：不占用其他的SFR，不改变PC的值。根据A的内容就可查到数据。

缺点：只能查找该指令后256字节范围内的数据表格，因此称为**近程查表指令。**

数据传送类指令

- **4.堆栈操作指令(2条)**
- 该组指令采用**直接寻址方式**，入栈操作是把直接寻址单元的内容传送到堆栈指针SP所指的单元中，出栈操作是把SP所指单元的内容送到直接寻址单元中。

➢ **助记符：** 进栈操作指令PUSH（PUSH onto stack）

- 出栈操作指令POP（POP from stack）
- PUSH direct ;(SP) ←(SP)+1，((SP)) ←(direct)
- POP direct ;(direct) ← ((SP)) ，(SP) ← (SP)-1

➢ **功能**

- **PUSH direct：**先修改SP指针，再将内部RAM direct单元的内容压入堆栈。
- **POP direct：**将堆栈栈顶的内容弹出，送到内部RAM direct单元，再修改SP指针。
- **堆栈指针SP的内容随着栈顶的变化而变化，即总是指向堆栈的顶部**

数据传送类指令

- **5.数据交换指令(5条)**
  - 该组指令是把A中的内容与源操作数所指的数据相互交换。有整字节交换和半字节交换。

➢ **助记符：**XCH（eXCHange，字节交换）；

- XCHD（eXCHange low-order Digit，低半字节交换）；
- SWAP（SWAP，A的低四位与高四位交换）。

● **字节交换：**

XCH A,Rn ;(A) ←→ (Rn)

XCH A,direct ;(A) ←→ (direct)

XCH A,@Ri ;(A) ←→ ((Ri))

● **半字节交换：**

XCHD A,@Ri ;(A)3~0 ←→ ((Ri))3~0

SWAP A ;(A)3~0 ←→ (A)7~4

➢ **功能：**把累加器A中的内容与源操作数所指出的数据相互交换。

5.数据交换指令(5条)

例：(A)=56H　执行SWAP A后，(A)=65H

例：(A)=34H，(R0)=20H，(20H)=78H:

执行：XCHD A,@R0后，(A)=38H,(20H)=74H

6. 数据传送类指令举例

- **例3-1：**将外部RAM100H单元中的内容送入外部RAM200H单元中。
- 程序如下：
- MOV DPTR，#0100H ；（DPTR）←#0100H
- MOVX A，@DPTR ；（A）←（（DPTR）），DPTR间址单元的内容读到A
- MOV DPTR，#0200H ；（DPTR）←#0200H
- MOVX @DPTR，A ；（（DPTR））←（A），A的内容写到DPTR间址单元

数据传送类指令

- **6.** 数据传送类指令举例

例3-2：(A)=5BH,(R1)=10H,(R2)=20H,(R3)=30H,(30H)=4FH,执行以下指令后，R1、R2、R3的结果分别是多少？

```
MOV  R1,A
MOV  R2，30H
MOV  R3，#83H
```

结果：(R1)=5BH,(R2)=4FH,(R3)=83H

提 纲

4. 算术运算类指令

算数运算类指令

- 算术运算指令是通过算术逻辑运算单元ALU进行数据运算与处理的指令，主要完成加、减、乘、除四则运算，以及加1、减1、BCD码运算和调整等。除加1、减1运算外，这类指令大多数要影响PSW中的标志位。24条指令可分为6组。

  - 不带进位加法指令: 4条
  - 带进位加法指令: 4条
  - 带借位减法指令: 4条
  - 加1指令: 5条
  - 减1指令: 4条
  - 乘法、除法、十进制运算调整指令： 3条

算数运算类指令

- **1.不带进位加法指令(4条) 助记符ADD**(Addition)

```
ADD   A,   #data    ;(A) ← (A)+ #data
ADD   A,   direct   ;(A) ← (A)+(direct)
ADD   A,   @Ri      ;(A) ← (A)+((Ri))
ADD   A,   Rn       ;(A) ← (A)+ (Rn)
```

➢ **功能：**将源操作数（Rn、direct、@Ri或立即数）和目的操作数（在A中）相加后，结果存放到A中。

算数运算类指令

- **1.不带进位加法指令(4条)：助记符ADD**(Addition)

例3-3：设(A) =4AH ,(R0)= 5CH。 执行指令：ADD A，R0

```
    01001010B
+   01011100B
-------------
    10100110B
```

执行结果：(A)=A6H，C=0，

OV=1，AC=1，P=0；

OV的判断：D6有进位而C7无进位，故OV=1；即出现二个正数相加，结果为负数的错误。

标志位的意义与操作数是带符号数还是无符号数有关：

➢无符号数相加时，如果Cy被置位，说明累加和超过了8位无符号数的最大值（255），此时OV虽受影响但无意义；

➢带符号数相加时，若溢出标志OV位被置位，说明累加和超出了8位带符号数的范围（-128~+127）。即出现了两个正数相加，和为负数；或两个负数相加，和为正数的错误结果。此时Cy虽受影响但已不需关注。

算数运算类指令

- **2.带进位加法指令(4条)：助记符ADDC**(Addition with Carry)

```
ADDC   A,  #data      ; (A) ← (A)+ data+(C)
ADDC   A,  direct     ; (A) ← (A)+ (direct)+ (C)
ADDC   A,  @Ri        ; (A) ← (A)+((Ri)) +(C)
ADDC   A,  Rn         ; (A) ← (A)+(Rn)+ (C)
```

  - **功能：**把源操作数、A和当前Cy的值相加，结果保存到A。主要用于多字节加法中。

- **例3-4：**(A)=AEH,(R0)=81H, C =1,执行指令 ADDC A,R0

```
    1 0 1 0 1 1 1 0
    1 0 0 0 0 0 0 1
+)                1
-----------------------
  1 0 0 1 1 0 0 0 0
```

执行结果：(A)=30H，C=1，OV=1，AC=1，P=0；

算数运算类指令

- **3.带借位减法指令(4条) ： 助记符SUBB**(Subtract with Carry)

```
SUBB   A, #data     ;(A) ← (A)-data-(C)
SUBB   A, direct    ;(A) ← (A)-(direct)-(C)
SUBB   A, @Ri       ;(A) ← (A)-((Ri))-(C)
SUBB   A, Rn        ;(A) ← (A)-(Rn)-(C)
```

➢**功能：**将A中的值减去源操作数指定的值，以及借位位Cy，结果存放在A中。

➢**若D7有借位则C置1，否则C清0；若D3有借位，则AC置1，否则AC清0。**

➢**若D7和D6中有一位有借位而另一位没有，则OV置1； 表示正数减负数结果为负，或负数减正数结果为正，结果错误。**

- **3.带借位减法指令(4条)**

例**3-5**：已知（A）=C9H, C=1，执行SUBB A, #54H的结果。

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

执行结果：(A)=74H ,C=0, OV=1, AC=0, P=0；（负数减正数得到了正数的结果，表示溢出）

- **在进行减法运算前，如果不清楚借位标志位Cy的状态，则应先对Cy进行清0。**

算数运算类指令

- **4.加1指令(5条)：助记符INC**(Increment)

```
INC     Rn          ;  (Rn) ← (Rn)＋1
INC     direct      ;  (direct) ← (direct)＋1
INC     @Ri         ;  ((Ri)) ← ((Ri))＋1
INC     A           ;  (A) ← (A)＋1
INC     DPTR        ;  (DPTR) ←(DPTR)＋1
```

➢ **功能：**将指令中的操作数加1。

算数运算类指令

- **5.减1指令(4条)：助记符DEC**(Decrement)

```
DEC   A        ; (A) ← （A）－1
DEC   direct   ; (direct) ← (direct)－1
DEC   @Ri      ; ((Ri)) ← ((Ri))－1
DEC   Rn       ; (Rn) ← (Rn)－1
```

➢ **功能：**指令中的操作数减1。若原操作数为#00H，则减1后为#0FFH。

INC A和DEC A，加1指令和减1指令不影响除P外的标志位。

➢**注意：无DPTR减1指令。若要使(DPTR)－1，必须要用一段程序来实现。**

```
   DPH  DPL        如:      01 00
－  00    01                －  00 01
   DPH  DPL                  00 FF
```

- **5.减1指令(4条)**

DPTR指针减1的程序：

```
DPTRSUB1:  CLR     C
           MOV     A,DPL
           SUBB    A,#1
           MOV     DPL,A
           MOV     A,DPH
           SUBB    A,#0
           MOV     DPH,A
```

如(DPTR)=0580H，执行后结果为057FH；

如(DPTR)=1000H，执行后结果为0FFFH。

Thank you!