第二章　MCS51(汇编)指令系统

- 指令分类、格式及一般说明
- 寻址方式
- 指令系统介绍
- 指令效率分析

MCS-51指令系统的分类及一般说明

汇编语言程序设计的意义

什么是程序？

完成某项特定任务的指令的集合。

计算机按程序一条一条地依次执行指令，从而完成指定任务。

要让计算机完成各项任务，就应设计各种程序。

程序设计语言：

机器语言：用二进制代码表示指令和数据。

汇编语言：用助记符表示指令操作功能，用符号表示操作对象。

高级语言：独立于机器，面向过程，接近自然语言和数学表达式。

汇编语言程序的每一条语句都与计算机的某一条指令对应，所以必需熟悉指令系统。

指令系统与指令格式

指令：指令是使计算机内部执行一定动作的一种控制命令。指令的集合构成指令系统。

指令格式：指令的表示方法称为指令格式。

指令 = 操作码 + 操作数

- 操作码：规定指令的操作功能
- 操作数：指令操作的具体对象（地址、数据）

MCS-51指令格式：

[标号：] 操作码助记符 [（目的操作数），（源操作数）]；注释

MCS-51的指令分类：

- 单字节指令
- 双字节指令
- 三字节指令

- 1个机器周期
- 2个机器周期
- 4个机器周期

……

指令分类

- 按指令功能分：数据传递与交换、算术运算、逻辑运算、程序转移、布尔（位）处理操作、CPU控制等6类。
- 按指令空间效率分：单字节指令（49条）、双字节指令（45条）、三字节指令（17条）。
- 按指令时间效率分：单周期指令（64条）、双周期指令（45条）、四周期指令（2条）。
- 按操作数数量分：无操作数指令（NOP、RET、RETI）、单操作数指令、双操作数指令。

MCS-51单片机指令系统特点

（1）指令执行时间快。

（2）指令短，约有一半的指令为单字节指令。

（3）支持乘法或除法指令（4个机器周期）。

（4）具有丰富的位操作指令。

（5）可直接用传送指令实现端口的输入输出操作。

指令系统的寻址方式

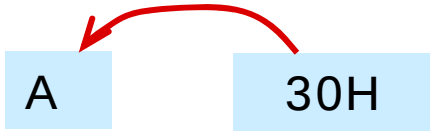
MCS-51单片机的指令系统共有111条指令，寻址方式如下：

- 立即寻址方式
- 直接寻址方式
- 寄存器寻址方式
- 寄存器间接寻址方式
- 基址寄存器加变址寄存器间接寻址方式（变址寻址方式）
- 相对寻址方式（变址寻址）
- 位寻址（直接寻址）

指令系统的寻址方式

一、立即寻址

立即寻址是指在指令中直接给出其操作数，该操作数称为立即数。为了与直接寻址指令中的直接地址相区别，在立即数前面必需加上前缀“#”，立即数是放在ROM中的常数（8位或16位数），立即数是指令代码的一部分。



例如：MOV A, #30H; （A）←30H

再例如：

```
MOV   A, # 60H          ;  （A）←60H
MOV   DPTR, # 3400H     ;  （DPTR）←3400H
MOV   30H, # 40H        ;  （30H）←40H
```

- 上述三条指令执行完后，累加器A中数据为立即数60H，DPTR寄存器中数据为3400H，30H单元中数据为立即数40H。

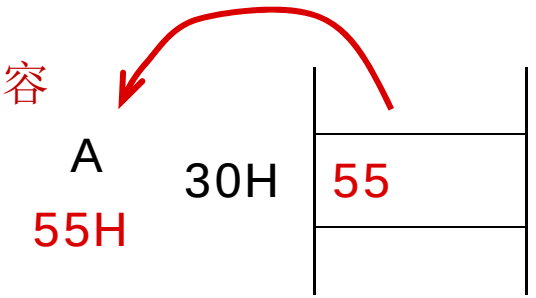
指令系统的寻址方式

二、直接寻址

直接寻址是指在指令中直接给出存放数据的地址（注意：不是立即数，并且只限于片内RAM范围）。直接寻址只能访问特殊功能寄存器、内部数据存储器和位地址空间。

例如：MOV A，30H ；（A）←内部RAM 30H单元中的内容

（30H为直接给出的内部RAM的地址。）



再例如：

MOV PSW，# 20H；　　（PSW）←20H

MOV DPTR，# 3400H；　（DPTR）←3400H

（PSW、DPTR为直接寻址寄存器的符号地址。

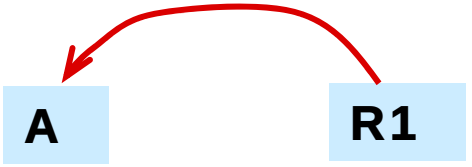
PSW（0D0H），DPTR——DPH，DPL（83H，82H））

指令系统的寻址方式

三、寄存器寻址

寄存器寻址是指指令中的操作数为通用寄存器中的内容。通用寄存器指四个寄存器区的R0～R7 。

例如：MOV A, R1; （A）←（R1）



再例如：设当前PSW的RS1、RS0均为 0，则下列操作涉及0区的R0-R7。

```
CLR     R0          ;  (R0) ←0
INC     R7          ;  (R7) ← (R7) +1
ADD     A,  R5      ;  (A) ← (A) + (R5)
```

指令系统的寻址方式

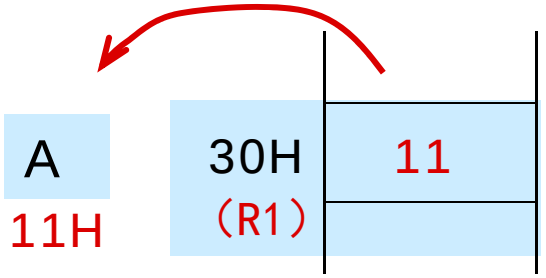
四、寄存器间接寻址

- 以寄存器中内容为地址，以该地址单元中内容为操作数的寻址方式。
  间接寻址的存储器空间包括内部数据RAM和外部数据RAM。
- 能用于寄存器间接寻址的寄存器有：R0，R1，DPTR，SP。

例如：

MOV R1，#30H；（R1）←30H（把立即数30H送R1寄存器）

MOV A，@R1；（A）←（30H）（把30H单元中的数送A中）

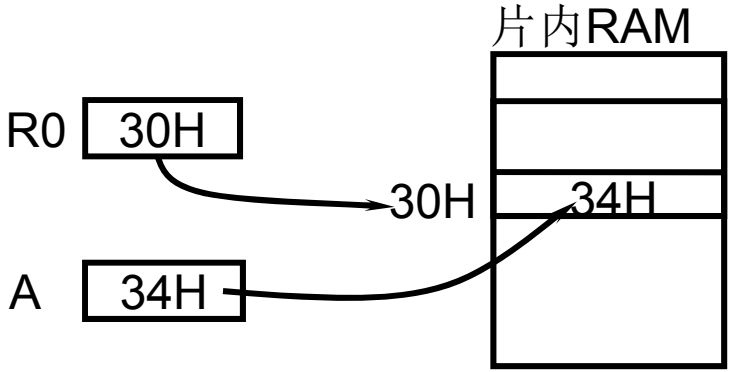


再例如：

已知（R0）=30H，（A）=34H；

MOV @R0，A；（(R0))←（A）

（A的内容送R0寻址的内部RAM单元中）



MOV @R0，A 间接寻址示意图

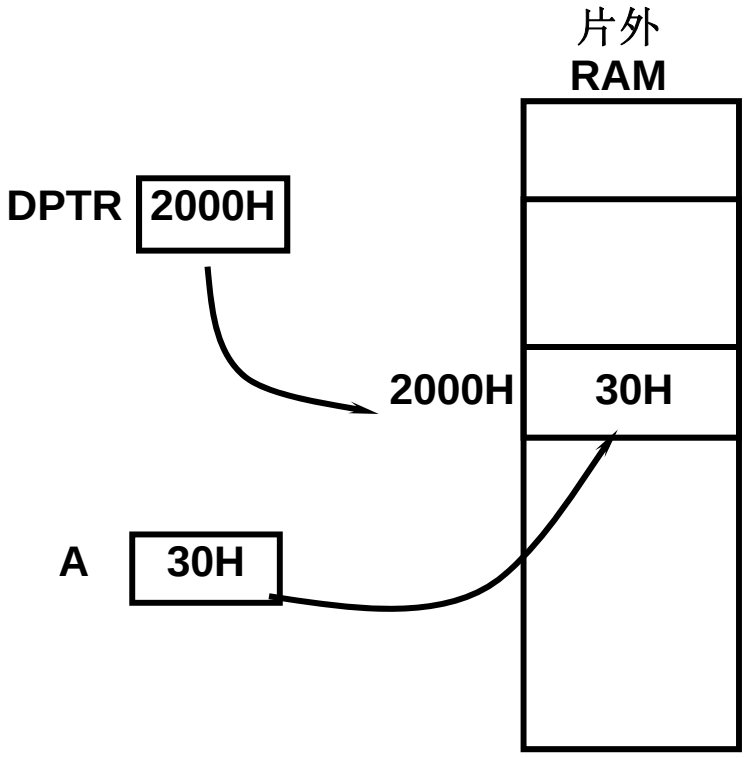
四、寄存器间接寻址

例如，已知（DPTR）=2000H，

（A）=30H。

MOVX @DPTR，A；((DPTR)) ← （A）

（A的内容送DPTR寻址的外部RAM单元中）



MOVX @DPTR，A　间接寻址示意图

指令系统的寻址方式

五、基址寄存器加变址寄存器间接寻址方式（变址寻址方式）

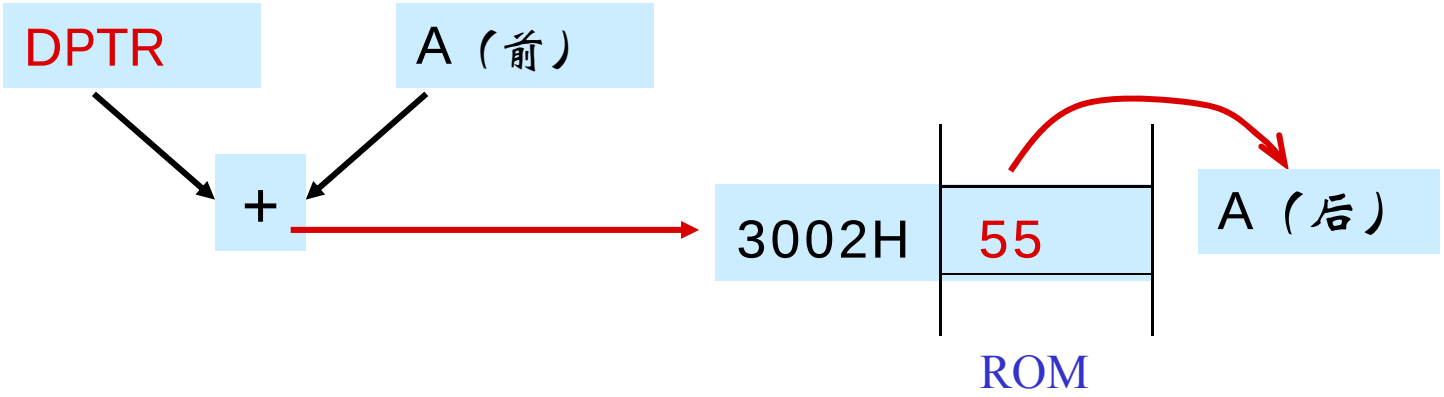
变址寻址用于访问程序存储器中的一个字节，该字节的地址是：基址寄存器（DPTR或PC）的内容与变址寄存器（A）中的内容之和。由于程序存储器是只读的，因此变址寻址只有读操作而无写操作，采用MOVC形式的指令。

MOVC A，@A+DPTR；（A）←（（A）+（DPTR））

MOVC A，@A+PC ；（A）←（（A）+（PC））

这二条指令通常又称为“查表指令”

例如：
```
MOV  DPTR, #3000H ; 立即数3000H送DPTR
MOV  A,  #02H     ; 立即数02H送A
MOVC A, @A+DPTR   ; 取ROM中3002H单元中的数55H送A
```



指令系统的寻址方式

六、相对寻址

以PC当前值为基准，加上相对偏移量rel （补码）形成转移地址。

转移范围：以PC当前值起始地址，相对偏移在-128～+127字节单元之间。

相对寻址方式为相对转移指令所采用，转移的目的地址为：

目的地址 = PC当前值起始地址+rel

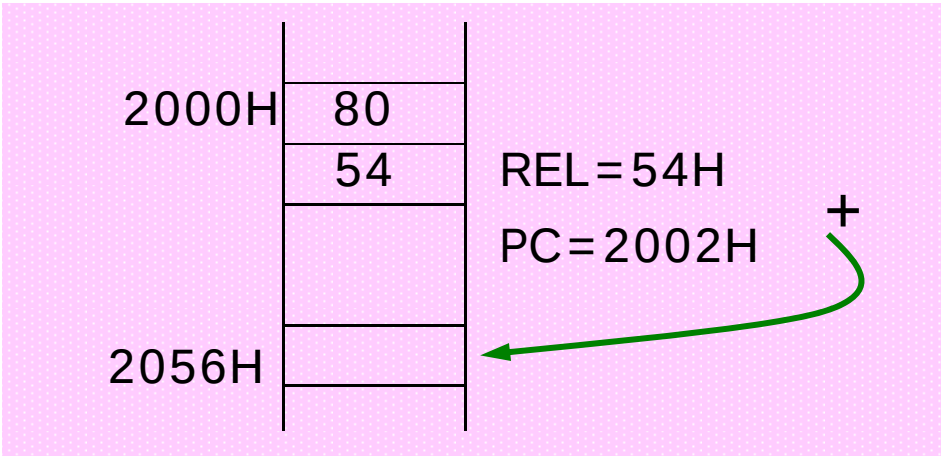
= 转移指令所在地址+转移指令字节数+rel

例：2000H　　SJMP　LP

　　.....

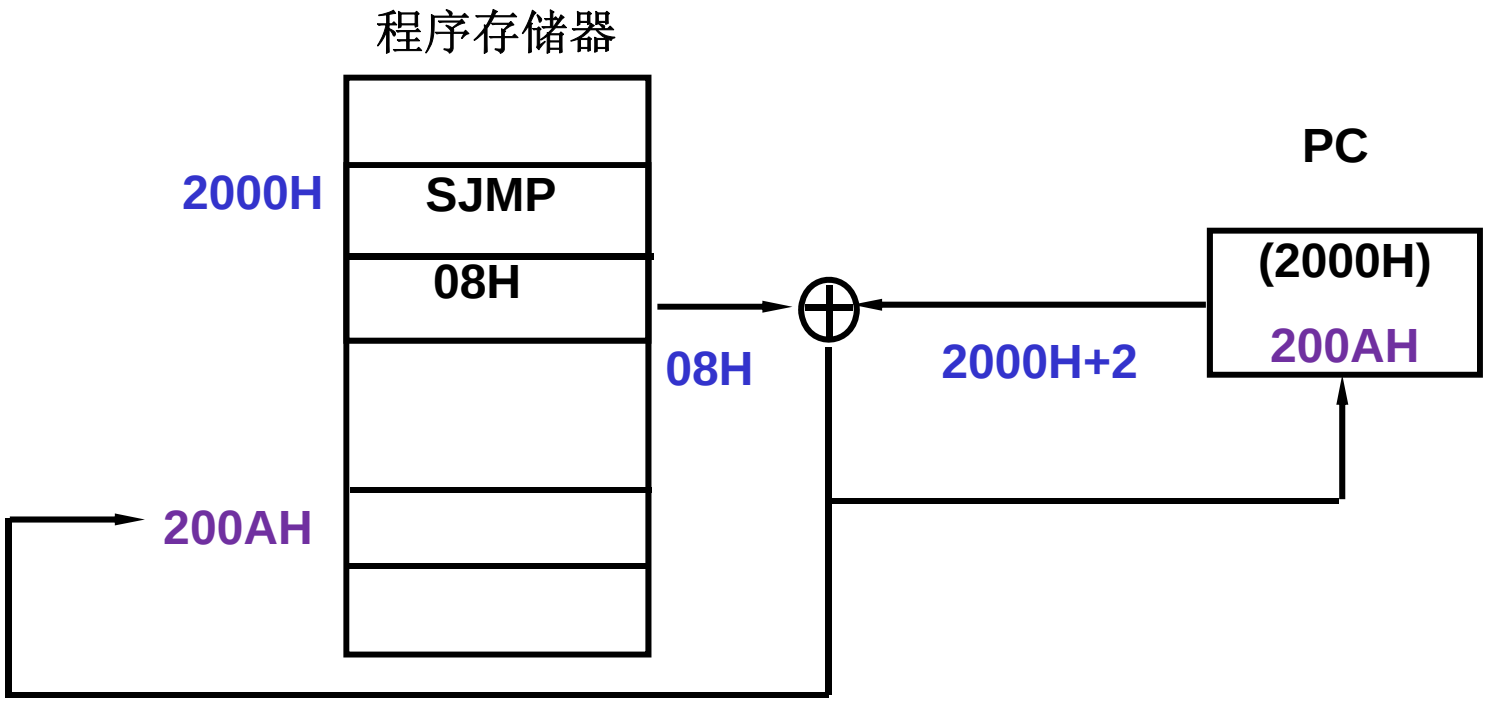
　　2056H　　LP:　.....

偏移量rel的计算由汇编程序完成



六、相对寻址

例如，SJMP 08H　　；（PC）←（PC）+2+08H



相对寻址示意图

指令系统的寻址方式

七、位寻址

位寻址，是指对片内RAM的位寻址区（20H~2FH）、可位寻址的特殊功能寄存器（SFR）的各位，进行位操作的寻址方式。例如：

```
MOV   C, 00H;    把直接位地址00H位（20H单元中D0位）的值送C（位累加器）。
MOV   P1.3, C;   把C位中的值送P1口的D3位。
SETB  2AH.7;     把位寻址区2AH单元D7位（直接位地址为57H）置1。
```

- 位寻址，只能对有位地址的单元作位寻址操作。
- 位寻址，其实是一种直接寻址方式，不过其地址是位地址，只能用在位操作指令之中。

位地址的表示方法：

位名称；	例：Cy、RS0、RS1
寄存器名加序号；	例：ACC.1、P0.3
字节地址加序号；	例：20H.3
直接位地址；	例：07H、 65H

操作数的7种寻址方式和寻址的空间

寻址方式	相关寄存器	源操作数寻址的空间
立即寻址		程序存储器中（立即数存储于ROM中）
直接寻址		片内低128字节RAM和SFR
寄存器寻址	R0~R7，A，B、DPTR	R0~R7，A，B，DPTR
寄存器间接寻址	@R0，@R1，@SP	片内RAM
	@R0，@R1，@DPTR	片外数据存储器
变址寻址	@A+PC，@A+DPTR	程序存储器
相对寻址	PC	程序存储器
位寻址	可位寻址的SFR	片内RAM20H~2FH、SFR中可寻址位

MCS-51指令系统介绍

MCS-51单片机的指令系统共分为五大类：

数据传送类指令，
算术运算类指令，
逻辑运算类指令，
控制转移类指令，
位操作类指令。

指令格式如下：

操作码［操作数1］［，操作数2］［，操作数3］

- 方括号，“[　]”内的字段表示可有可无。
- 操作码，表示指令进行何种操作，即操作性质。一般为英语单词缩写。
- 操作数，指出了参加操作的数据或数据存放的地址，即操作对象。它以一个或几个空格与操作码隔开；根据指令功能的不同，操作数可以有3个、2个、1个或没有，操作数之间以逗号“，”分开。

MCS-51指令系统介绍

MCS-51指令中，常用的符号：

Rn：当前寄存器组的8个工作寄存器R0~R7。

Ri：可用作间接寻址的工作寄存器，只能是R0、R1。

direct：直接地址，即8位的内部数据存储器或者特殊功能寄存器的地址。

#data、#data16：分别表示8位、16位立即数。

rel：相对转移指令中的偏移量，为8位带符号数（补码形式）。

addr11、addr16：分别表示11位、16位地址码。

bit：片内RAM中（可位寻址）的位地址。

MCS-51指令系统介绍

MCS-51指令中，常用的符号：

DPTR：数据指针，用作16位的地址寄存器。

A：累加器A；ACC则表示累加器A的地址。

C或Cy：进位标志位或位处理机中的位累加器。

@：间接寻址的前缀标志。

/：位操作数的前缀，表示对该位操作数取反，如：/bit。

$ ：当前指令存放的地址。

(X)：X中的内容。

((X))：由X间接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。

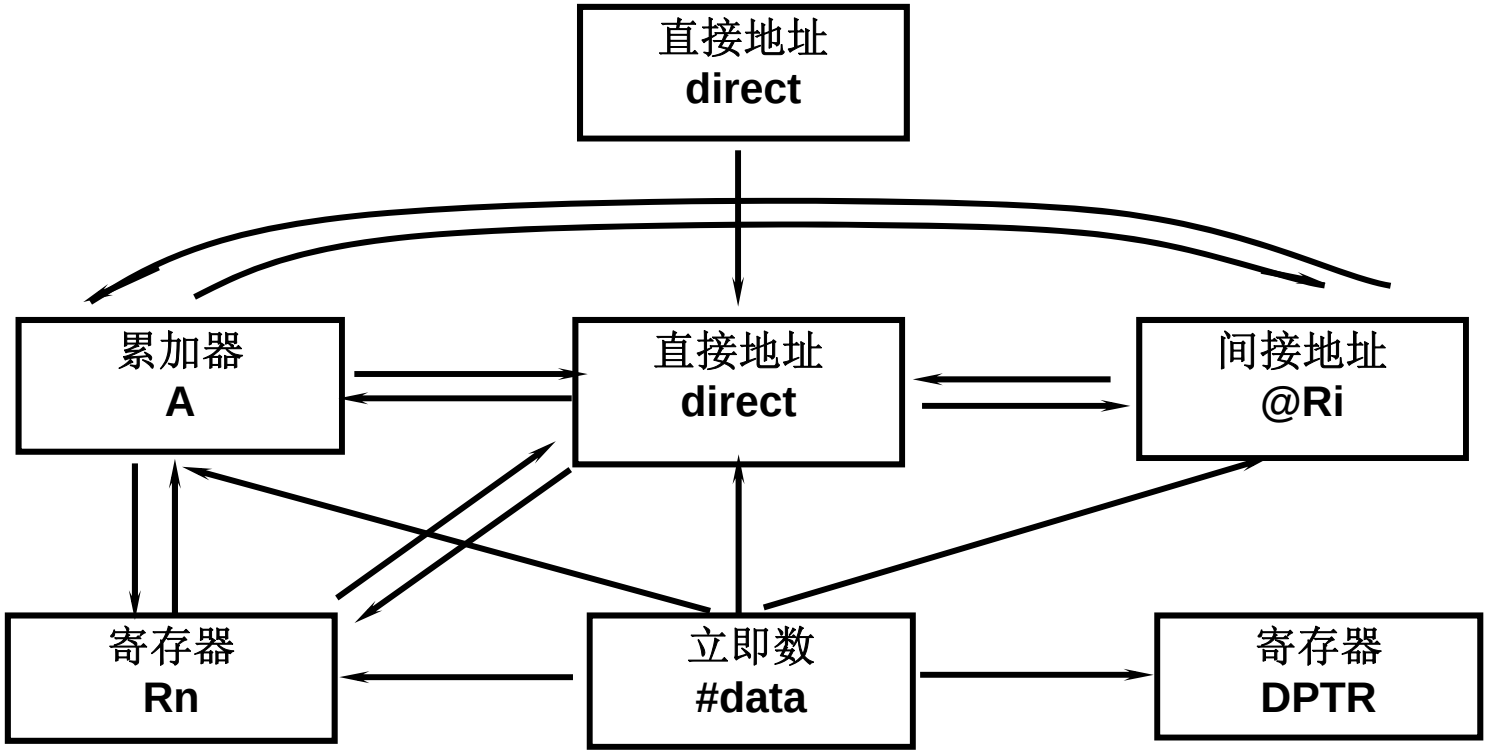
一、数据传送类指令

- ✓ 数据传送类指令共28条，是将源操作数送到目的操作数。指令执行后，源操作数不变，目的操作数被源操作数取代。数据传送类指令用到的助记符有MOV、MOVX、MOVC、XCH、XCHD、SWAP、PUSH、POP8种。

- ✓ 源操作数可采用寄存器、寄存器间接、直接、立即、变址5种寻址方式寻址，目的操作数可以采用寄存器、寄存器间接、直接寻址3种寻址方式。

- ✓ MCS-51单片机片内数据传送途径如下图所示。

一、数据传送类指令



MCS-51单片机片内数据传送图

一、数据传送类指令

功能：实现寄存器、存储器之间的数据传送。

内部传送指令：	片内数据存储器数据传送。
外部传送指令：	片外数据存储器数据传送。
查表指令：	程序存储器数据传送。
交换指令：	片内数据存储器数据传送。
堆栈操作指令：	片内数据存储器数据传送。

注意：只有指令表中的指令才有对应指令代码，计算机才能执行。编程时，不能随意创造发明指令。

一、数据传送类指令

特点：除第一操作数为A的指令，影响PSW中的标志P位之外，并不影响其他标志位。

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：顺序执行下列指令序列，求每一步执行结果。

```
MOV  A，  #30H      ；(A)= 30H
MOV  4FH，A         ；(4FH)= 30H
MOV  R0，  #20H     ；(R0) = 20H
MOV  @R0，4FH       ；(20H)= 30H
MOV  21H，20H       ；(21H)= 30H
```

一、数据传送类指令

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
```

3. 以直接地址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     ; (direct2)→（direct1）
MOV  direct，  @Ri        ; ((Ri))→（direct），i=0,1
MOV  direct，#data        ; data→（direct ）
```

一、数据传送类指令

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))，i=0,1**

MOV @Ri， #data ；**data→((Ri))，i=0,1**

例如：设（30H）=6FH，（R1）=40H，执行 MOV @R1，30H 后，

30H单元中数据6FH取出，送入R1间接寻址的40H单元，（40H）=6FH。

5. 16位数据传送指令

MOV DPTR，#data16 ；**(DPTR) ← data16，即：**
（DPH）←立即数高字节
（DPL）←立即数低字节

例如：**MOV 30H，#7AH** ；**将立即数7AH送片内RAM 30H单元中**

MOV R0，#30H ；**将立即数30H送R0寄存器**

MOV A，@R0 ；**将R0指定的30H中的数7AH送A中**

MOV DPTR，#1000H ；**将1000H送DPTR寄存器**

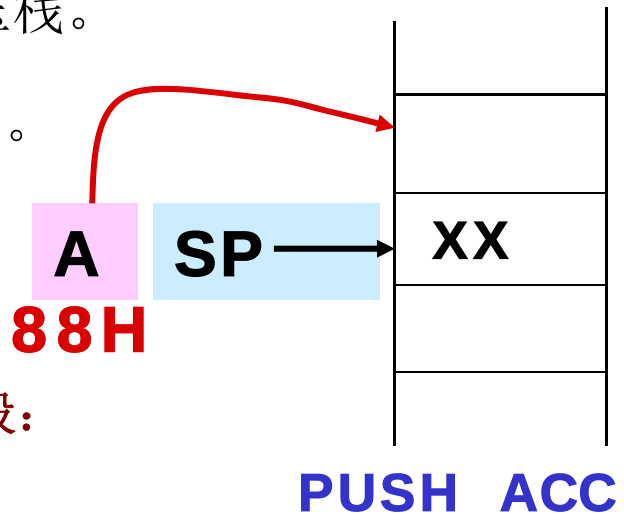
一、数据传送类指令

6. 堆栈操作指令

堆栈是在内部RAM中开辟的一个数据的暂存空间，遵守“后进先出”原则操作，其地址指针为SP，它指出栈顶的位置，复位时（SP）=07H。

入栈：PUSH direct ；（SP）先加1，再将数据压栈。

出栈：POP direct ；数据先出栈，再（SP）减1。



例如，已知（A）=44H，（30H）=55H，执行以下程序段：

```
MOV   SP，#5FH   ；堆栈起点地址设置为5FH
PUSH  ACC        ；A中的44H压到60H中保存
PUSH  30H        ；30H中的55H压到61H中保存
POP   30H        ；把61H中的55H弹出，送到30H
POP   ACC        ；把60H中的44H弹出，送到A中
```

一、数据传送类指令

指令运用举例:

例：将片内RAM的30H单元与40H单元中的内容互换。

方法1（直接地址传送法）：

```
MOV   31H，30H
MOV   30H，40H
MOV   40H，31H
SJMP    $
```

方法2（间接地址传送法）：

```
MOV   R0，#40H
MOV   R1，#30H
MOV   A，@R0
MOV   B，@R1
MOV   @R1，A
MOV   @R0，B
SJMP    $
```

一、数据传送类指令

指令运用举例:

例：将片内RAM的 30H单元与 40H单元中的内容互换。

方法3（字节交换传送法）：

```
MOV   A，30H
XCH   A，40H
MOV   30H，A
SJMP  $
```

方法4（堆栈传送法）：

```
PUSH    30H
PUSH    40H
POP     30H
POP     40H
SJMP    $
```

一、数据传送类指令

7. 累加器A与外部数据存储器传送指令

采用DPTR间接寻址；外部数据存储器的地址为16位（寻址空间64Kbyte）。采用Ri（i=0,1）间接寻址，外部数据存储器的地址为8位（寻址空间256byte）。

（1）64KB外部RAM单元与A之间的传送

MOVX A，@DPTR ；**((DPTR))→（A），读外部RAM或I/O口**

MOVX @DPTR, A ；**(A)→((DPTR))，写外部RAM或I/O口**

(2) 外部RAM低256字节单元与A之间的传送

MOVX A， @Ri ；**((Ri))→（A），读外部RAM或I/O口**

MOVX @Ri， A ；**(A)→ ((Ri))，写外部RAM或I/O口**

一、数据传送类指令

例: **MOV R0，#80H**

MOVX A， @R0 ；将外部RAM的80H单元内容 →（A）

例: **MOV DPTR, #2000H**

MOVX A，@DPTR ；将外部RAM中2000H单元内容 →（A）

例：试编写一程序段，实现将外部RAM 0FAH单元中的内容，传送到外部RAM 04FFH单元中。

```
MOV     DPTR，  #04FFH
MOV     R0，  #0FAH
MOVX    A，@R0
MOVX    @DPTR,    A
```

注意：片外数据存储器不能直接寻址。下列为非法指令：

```
MOVX    A，     2000H
MOVX   2100H，2000H
```

一、数据传送类指令

8. 查表指令

查表指令共两条，均为单字节指令，用于读取程序存储器中的数据表格。

```
MOVC  A，@A+DPTR        ；((A)+(DPTR))→(A)
MOVC  A，@A+PC          ；(PC)+1 →(PC)；((A)+(PC))→(A)
                        ；以PC的当前值为基址，A为变址。
```

例1：在程序存储器的ROM中从1000H开始存有5个字节数，编程将第2个字节数取出送片内RAM 30H单元中。程序段如下：

```
    MOV   DPTR，#1000H      ；置ROM地址指针（基址），(DPTR)=1000H;
    MOV   A，#01H           ；表内序号送A（变址）,(A)=01H;
    MOVC  A，@A+DPTR        ；从ROM 1001H单元中取数67H送到A;
    MOV  30H，A             ；再存入内RAM 30H中;
        ORG    1000H        ；ORG伪指令，定义数表起始地址为1000H;
TAB:  DB   55H，67H，9AH，…  ；DB伪指令，在ROM内TAB（1000H）开始的
                            空间中定义 5 个单字节数。
```

一、数据传送类指令

例2：设某数N已存于20H单元（N≤10），查表求N平方值，存入21H单元。

```
        MOV     A，20H                  ；取数N
        ADD     A，#02H                 ；增加PC定位到TAB的偏移量
        MOVC    A，@A+PC                ；查表
        SJMP    Store
TAB:    DB  00H，01H，04H，09H，……      ；定义平方值数表
Store:  MOV     21H，A                  ；结果（ N平方值）存入21H单元
```

由于PC为程序计数器，总是指向下一条指令的地址，在执行指令“MOVC A，@A+PC”前，应在A累加器中加上查表偏移量，即本例中SJMP指令的字节数（2）。

用DPTR查表时，表格可以放在ROM的64KB范围；用MOVC A，@A+PC 指令查表时，表格只能放在该查表指令所在地址的+256个单元之内。

一、数据传送类指令

9. 字节/半字节交换指令指令

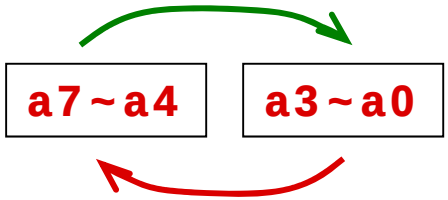
有3条XCH指令为整个字节相互交换，XCHD指令为低4位相互交换，SWAP指令为ACC中的高、低4位互换。

(1）字节交换

```
XCH  A, Rn          ; (A)←→(Rn)
XCH  A, direct      ; (A)←→(direct)
XCH  A, @Ri         ; (A)←→((Ri))
```

(2）半字节交换

```
XCHD  A,  @Ri   ; 低4位内容相互交换
SWAP  A         ; A的高4 位与低4位交换
```



SWAP A

思考：已知 (A)=34H，(R6)=29H，执行以下指令后，(A)=?

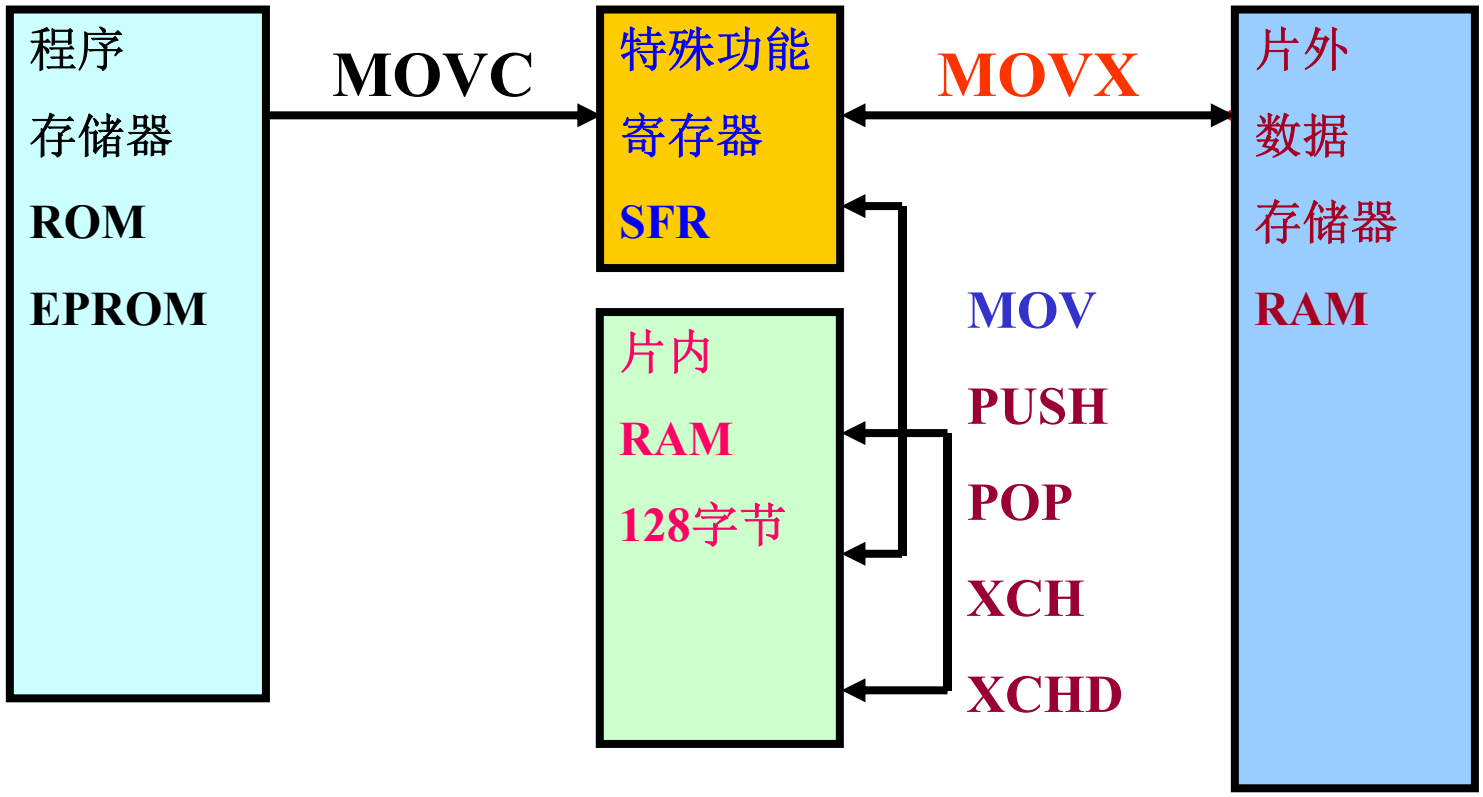
```
XCH   A, R6;
SWAP  A ;
```

一、数据传送类指令

传送规律：（1）程序存储器只读不可写，且只能传送至SFR；

（2）片外数据存储器可读、写，但只能与SFR交换数据；

（3）内部数据传送只能在SFR、片内RAM之间交换数据。



MCS-51单片机内外部数据传送图

二、算术操作类指令

算术运算类指令有加、减、乘、除法指令、增1和减1指令、十进制调整指令，共24条。使用时应注意判断各种结果对哪些标志位（Cy、OV、Ac、P）产生影响。

1. 不带进位的加法指令 ------- 指令助记符为 ADD

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

二、算术操作类指令

对标志位的影响：

- 如果位7有进位，则进位标志位Cy置“1”，否则Cy清“0”；
- 如果位3有进位，则辅助进位标志位Ac置“1”，否则Ac清“0”；
- 如果位6有进位而位7没有进位，或者位6没有进位而位7有进位，则溢出标志位OV置“1”，否则OV清“0” 。

例：（A）=53H，（R0）=0FCH，执行指令 ADD A，R0。

```
      0101 0011   (A)
+     1111 1100   (R0)
-------------------------
  1← 0100 1111
```

结果为：（A）=4FH，Cy=1，Ac=0，OV=0，P=1（偶校验）

无符号数相加时：若Cy = 1，说明有溢出（其值 > 255）；

有符号数相加时：若OV = 1，说明有溢出（其值大于127或小于 -128）；

OV＝1表示：两个正数相加，和变为负数；或两个负数相加，和变为正数的错误结果。

例：有符号数， 两个正数相加

120＋100＝220>127，

01111000 （120）

＋ 01100100 （100）

11011100 （溢出）

OV =1

位6有进位，位7无进位，符号位由0变1时，结果变负。➜结果错误

例：有符号数 ，两个负数相加

(-120)+(-100)＝ -220 < -128

	10001000 （-120）	（补码）
+	10011100 （-100）	（补码）
1←	00100100	（溢出）

OV = 1

位6无进位，位7有进位，符号位由1变为0，结果变正→**错误。**
在带符号的加减法中，OV是一个重要的编程标志。

二、算术操作类指令

2. 带进位的加法指令 ------- 指令助记符为 ADDC

ADDC 比 ADD 多了加 Cy 位的值（之前指令留下的Cy值），主要用于多字节的加法运算，结果也送A。影响Ac、Cy、OV、P位。

Cy为来自PSW状态寄存器中的进位位Cy 。

```
ADDC   A, Rn          ; (A)+(Rn)+(Cy)  →(A)
ADDC   A, direct      ; (A)+(direct)+(Cy)  →(A)
ADDC   A, @Ri         ; (A)+((Ri))+(Cy)  →(A)
ADDC   A, #data       ; (A)+data +(Cy) →(A)
```

对标志位的影响和ADD指令一样。

上述四条指令多用于多字节数相加。

二、算术操作类指令

例：把存放在R1R2和R3R4中的两个16位数相加，
结果存于R5R6中。

解：参考程序如下：

```
MOV     A，R2       ；取第一个数的低8位
ADD     A，R4       ；两数的低8位相加
MOV     R6，A       ；保存和的低8位
MOV     A，R1       ；取第一个数的高8位
ADDC    A，R3       ；两数的高8位相加，并把
                      低8位相加时的进位C加进来
MOV     R5，A       ；把相加的高8位存入R5寄存器中
SJMP  $
```

二、算术操作类指令

3. 增1指令 ------ 指令助记符为 INC

指令的功能是将操作数中的内容加1。除对A操作影响P外，不影响任何标志。

INC A ；(A)+1→(A) 以下类同

INC Rn

INC direct

INC @Ri

INC DPTR

若原来的内容为FFH，则加1后变为00H。

二、算术操作类指令

4. 带借位的减法指令 ------ 指令助记符为 SUBB

指令的功能是第一操作数A的内容减去第二操作数的内容，再减去Cy值，然后把结果存入A中，同时产生新的Cy、Ac、OV、P位的值。

```
SUBB   A, Rn          ; (A)-(Rn)-(Cy) →(A)
SUBB   A, direct      ; (A)-(direct)-(Cy) →(A)
SUBB   A, @Ri         ; (A)-((Ri))-(Cy) →(A)
SUBB   A, #data       ; (A)- data-(Cy) →(A)
```

二、算术操作类指令

SUBB对标志位的影响：

- 如果位7需借位，则标志位Cy置“1”，否则Cy清“0”；
- 如果位3需借位，则标志位Ac置“1”，否则Ac清“0”；
- 如果位6需借位而位7不需要借位，或者位6不需借位而位7需借位，则溢出标志位OV置“1”，否则OV清“0” 。
- 在带符号数运算时，只有当符号不同的两数相减时，才会发生溢出。所以OV置1，表示发生了正数减去负数差为负，或负数减去正数差为正的情况。

例：（A）=C9H，（R2）=54H，Cy=1，执行指令 SUBB A，R2。

```
      1100 1001    (A)
      0101 0100    (R2)
-           1      (Cy)
-------------------------
      0111 0100
```

结果为：（A）=74H，Cy=0，Ac=0，OV=1，P=0。

二、算术操作类指令

5. 减1指令 ------ 指令助记符为 DEC

指令的功能是将操作数中的内容减1。除对A操作影响P外，不影响任何标志。

```
DEC   A          ；(A)-1→(A)        以下类同
DEC   Rn
DEC   direct
DEC   @Ri
```

若原来的内容为00H，则减1后变为FFH。

注意：没有对DPTR的减1操作指令

二、算术操作类指令

6. 乘法指令------ 指令助记符为 MUL

MUL AB ；(A)×(B) →(B)(A)

说明： 实现8位无符号数乘法，乘积为16位，低8位放A中，高8位放B中；当乘积大于255（0FFH）时，置OV=1，否则OV=0；Cy位总是0，运算结果影响P标志位，执行时间为4个机器周期。

---

7. 除法指令 ------ 指令助记符为 DIV

DIV AB ；(A)/(B)：商→(A)，余数→(B)

说明：无符号数相除，当除数（B）=0时，结果为无意义，并置OV=1，否则，OV=0；Cy位总是0，运算结果影响P标志位，执行时间为4个机器周期。

例：试将A中的二进制数转换为3位BCD码，其中，百位数存放于31H单元，十位数和个位数压缩后存于30H单元中。

```
MOV   B，   #100
DIV   AB          ；得到商（百位数）放A中，余数 放B中；
MOV   31H，A      ；（31H）=百位数
MOV   A，   #10
XCH   A，   B     ；余数放A中，10放B中
DIV   AB          ；得到十位数和个位数，分别放在A和B的低4位中；
SWAP  A           ；将十位数放到A的高4位中
ADD   A，   B     ；将B中的个位数压缩到A的低4位中
MOV   30H，A
```

二、算术操作类指令

8. 十进制调整指令

ADD、ADDC指令都是对8位二进制数进行加法运算，当两个BCD码按二进制数相加后，必须经过DA A指令调整，才能得到正确的BCD码的“和”数，否则结果就会出错。【$(00100101)_{BCD}=25$，$(00100101)_{B}=37$】

DA A

指令的调整原则是：

- 若[$(A_{3\sim0})>9$]或[(Ac)=1]，则低4位 $(A_{3\sim0})$ +6调整；（产生低4位正确的BCD码）
- 若[$(A_{7\sim4})>9$]或[(Cy)=1]，则高4位 $(A_{7\sim4})$ +6调整；（产生高4位正确的BCD码）
- 若[$(A_{7\sim4})=9$]且 [$(A_{3\sim0})>9$] ，则高4位 $(A_{7\sim4})$ +6 和低4位 $(A_{3\sim0})$ +6调整。

注意：1）DA指令只能跟在加法指令后面使用；
2）调整前参与运算的两数是BCD码数；
3）DA指令不能与减法指令配对使用。