3.5 算术运算指令

51系列单片机的算术运算类指令共有24条，包括加、减、乘、除4种基本算术运算指令，这4种指令能对8位的无符号数进行直接运算，借助溢出标志，可对有符号数进行补码运算；借助进位标志，可实现多字节的加、减运算，同时还可对压缩的BCD码进行运算，其运算功能较强。

算术运算指令通过算术逻辑运算单元ALU进行数据运算处理。ALU仅执行无符号二进制整数的算术运算。对于带符号数则要进行其它处理。这类指令多数以A为源操作数之一，同时又使A为目的操作数。

助记符8种：ADD、ADDC、INC、DA、SUBB、DEC、MUL、DIV。

3.5 算术运算指令

算术运算指令执行结果将影响标志位（PSW中的OV、CY、AC和P等）。但是加1和减1指令不影响进位标志（Cy），辅助进位标志（AC）、溢出标志位（OV）这些标志。

进位（借位）标志CY为无符号整数的多字节加法、减法、移位等操作提供了方便；溢出标志OV可方便的控制补码运算；辅助进位标志AC用于BCD码运算。

指令 / 标志	ADD	ADDC	SUBB	DA	MUL	DIV
CY	√	√	√	√	0	0
AC	√	√	√	√	×	×
OV	√	√	√	×	√	√
P	√	√	√	√	√	√

注：符号√表示相应的指令操作影响标志；符号×表示相应的指令操作不影响标志；符号0表示相应的指令操作对该标志清0。另外，累加器加1（INC A）和减1（DEC A）指令影响P标志。

3.5 算术运算指令

3.5.1 加法指令

51系列的加法指令分为4类，共14条。

一、不带进位位的加法指令（ADD，Addition 4条）

ADD A，Rn ；（A）←（A）+（Rn）

ADD A，direct ；（A）←（A）+（direct）

ADD A，@Ri ；（A）←（A）+（（Ri））

ADD A，#data ；（A）←（A）+ data

功能：将两个操作数相加，结果再送回累加器中。

说明：这类指令将影响标志位AC、CY、OV、P。

CY：和的D7位有进位时，（CY）=1；否则，（CY）=0。

AC：和的D3位有进位时，（AC）=1；否则，（AC）=0。

OV：和的D7、D6位只有一个有进位时，（OV）=1；溢出表示运算的结果超出了数值所允许的范围。如：两个正数相加结果为负数或两个负数相加结果为正数时属于错误结果，此时（OV）=1。

P：累加器ACC中“1”的个数为奇数时，（P）=1；为偶数时，（P）=0。

3.5 算术运算指令

对于无符号数相加，若CY置“1”，说明和数溢出（大于255）。对于有符号数相加时，和数是否溢出（大于+127或小于-128），则可通过溢出标志OV来判断，若OV为“1”，说明和数溢出。

这4条指令使得累加器A可以和内部RAM的任何一个单元的内容进行相加，也可以和一个8位立即数相加，相加结果存放在A中。无论是哪一条加法指令，参加运算的都是两个8位二进制数。对用户来说，这些8位数可当作无符号数（0～255），也可以当作带符号数（-128～+127），即补码数。

例如，对于二进制数11010011，用户可认为它是无符号数，即为十进制数211，也可以认为它是带符号数，即为十进制负数-45。但计算机在作加法运算时，总按以下规定进行：

3.5 算术运算指令

（1）在求和时，总是把操作数直接相加，而无须任何变换。例如，若A=11010011B，R1=11101000B，执行指令ADD A，R1时，其算式表达为:

```
  11010011
+ 11101000
 110111011
```

相加后（A）=10111011B。若认为是无符号相加，则A的值代表十进制数187；若认为是带符号补码数相加，则A的值为十进制负数-69。

（2）在确定相加后进位标志CY的值时，总是把两个操作数作为无符号数直接相加而得出进位CY值。如上例中，相加后CY=1。若为无符号数相加CY代表十进制数256，但若是两个带符号数相加，CY没有意义。

（3）在确定相加后溢出标志OV的值时，计算机总是把操作数当作带符号数来对待。在作加法运算时，一个正数和一个负数相加是不可能产生溢出的，只有两个同符号数相加才有可能产生溢出，表示运算结果出错。

3.5 算术运算指令

例 设（A）＝C3H，（R0）=AAH。

执行指令：ADD A，R0

```
  1 1 0 0 0 0 1 1 B
+ 1 0 1 0 1 0 1 0 B
-----------------
1 0 1 1 0 1 1 0 1 B
```

执行结果：（A）=6DH，（CY）=1，（OV）=1，（AC）=0，P=1。

例 设（A）=85H，R0=20H，（20H）=0AFH，执行指令 ADD A，@R0

```
  10000101
+ 10101111
----------
100110100
```

结果：（A）=34H；（CY）=1；（AC）=1；（OV）=1；P=1。

对于加法，溢出只能发生在两个同符号数相加的情况。在进行有符号数的加法运算时，溢出标志OV是一个重要的编程标志，利用它可以判断两个有符号数相加，和数是否溢出。

3.5 算术运算指令

二、带进位位的加法指令（ADDC，Addition with Carry 4条）

ADDC A，Rn ；（A）←（A）+（Rn）+（CY）

ADDC A，direct ；(A）←（A）+（direct）+（CY）

ADDC A，@Ri ；（A）←（A）+（（Ri））+（CY）

ADDC A，＃data ；（A）←（A）+data+（CY）

功能：将工作寄存器Rn、片内RAM单元中的内容、间接地址存储器中的8位二进制数及立即数与累加器A的内容和当前进位标志CY的内容相加，相加的结果仍存放在A中。

说明：

1）指令的功能是把源操作数与累加器A的内容相加再与进位标志CY的值相加，结果送入目的操作数A中。加的进位标志CY的值是在该指令执行之前已经存在的进位标志的值，而不是执行该指令过程中产生的进位 。

2）常用于多字节数相加。

3）这类指令将影响标志位AC、CY、OV、P。情况与不带进位位的加法指令相同。

3.5 算术运算指令

当和的第3位有进位时，将AC标志置位，否则清0。

当和的第7位有进位时，将CY标志置位，表示和数溢出，否则清0。

对于带符号数运算，当和的第7位与第6位中有一位进位而另一位不产生进位时，溢出标志OV置位，否则为0。（OV）=1表示两个正数相加，和为负数；或两个负数相加而和为正数的错误结果。

例 设（A）=C3H，（R0）=AAH，（CY）=1。执行指令 ADDC A，R0

```
  1 1 0 0 0 0 1 1
+ 1 0 1 0 1 0 1 0
+               1   （CY）
-----------------
1 0 1 1 0 1 1 1 0
```

执行结果：（A）=6EH，（CY）=1，（OV）=l，（AC）=0。

对于带符号数的带进位相加，溢出标志为1，意味着出错，上例为两个负数相加，出现结果为正数的错误。

3.5 算术运算指令

例 已知（A）=B3H，（R1）=56H。执行指令ADD A，R1

B 3H	1 0 1 1 0 0 1 1
+ 5 6H	+ 0 1 0 1 0 1 1 0
1 0 9H	1 0 0 0 1 0 0 1
CY=1	CY=1，OV=0，AC=0

若两个数是无符号数，则B3H+56H=109H，答案正确。

若两个数是带符号数，则B3H的原码是（-77D），56H为86D，（-77D）+86D＝09D，答案也是正确的，OV＝0。

例 把存放在R1R2和R3R4中的两个16位数相加，结果存于R5R6中。

解： R2、 R4 、R6存放16位数的低字节，R1、 R3、 R5存放高字节。由于不存在16位数加法指令，所以只能先加低8位，后加高8位，而在加高8位时要连低8位相加时产生的进位一起相加。假设其和不超过16位，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 相加时的进位位加进来
MOV  R5，A   ；把相加的高8位存入R5寄存器中
SJMP  $
```

3.5 算术运算指令

三、增量指令（INC，Increase 5条）

INC A ；（A）←（A）+1

INC Rn ；（Rn）←（Rn）+1

INC direct ；（direct）←（direct）+1

INC @Ri ；（（Ri））←（（Ri））+1

INC DPTR ；（DPTR）←（DPTR）+1

功能：将工作寄存器Rn、片内RAM单元中的内容 、间接地址存储器中的8位二进制数、累加器A和数据指针DPTR的内容加 1，相加的结果仍存放在原单元中。

说明：这些指令仅 INC A 影响P标志。其余指令都不影响标志位的状态。若原来的内容为0FFH，则加1后将产生溢出，使操作数的内容变成00H，但不影响任何标志。最后一条指令是对16位的数据指针寄存器DPTR执行加1操作，指令执行时，先对低8位指针DPL的内容加1，当产生溢出时就对高8位指针DPH加1，也不影响任何标志。

3.5 算术运算指令

例 设（R0）=7EH，（7EH）=FFH.（7FH）=40H。

```
INC @R0    ；FFH＋1=00H 仍存入7EH单元
INC R0     ；7EH＋1=7FH 存入（R0）
INC @ R0   ；40H＋1=41H 存入（7FH）
```

执行结果为（R0）=7FH，（7EH）=00H.（7FH）=41H

例 设（A）=12H，（R3）=0FH，（35H）=4AH，（R0）=56H，（56H）=00H 。执行如下指令

```
INC    A     ;
INC    R3    ;
INC    35H   ;
INC    @R0   ;
```

结果为：（A）=13H ；（R3）=10H；（35H）=4BH；（56H）=01H。

3.5 算术运算指令

例 设（R0）=7EH，（7EH）=FFH，（7FH）=38H，
（DPTR）=10FEH，

```
INC  @R0    ；使7EH单元内容由FFH变为00H
INC  R0     ；使R0的内容由7EH变为7FH
INC  @R0    ；使7FH单元内容由38H变为39H
INC  DPTR   ；使DPL为FFH，DPH不变
INC  DPTR   ；使DPL为00H，DPH为11H
INC  DPTR   ；使DPL为01H， DPH不变
```

3.5 算术运算指令

四、十进制调整指令（Decimal Adjust for Addition 1条）

DA A

功能：对累加器A中刚进行的两个BCD码的加法的结果进行十进制调整。两个压缩的BCD码按二进制相加后，必须经过调整方能得到正确的压缩BCD码的和。

说明：BCD码采用4位二进制数编码，并且只采用了其中的10个编码，即0000～1001，分别代表BCD码0～9，而1010～1111为无效码。

当相加结果大于9，说明已进入无效编码区；

当相加结果有进位，说明已跳过无效编码区。

凡结果进入或跳过无效编码区时，结果是错误的，相加结果均比正确结果小6（差6个无效编码）。

十进制调整的修正方法：

（1）当累加器A中的低4位数出现了非BCD码（1010~1111）或低4位产生进位（AC=1），则应在低4位加6调整，以产生低4位正确的BCD结果。

（2）当累加器A中的高4位数出现了非BCD码（1010~1111）或高4位产生进位（CY=1），则应在高4位加6调整，以产生高4位正确的BCD结果。

3.5 算术运算指令

十进制调整指令执行后，PSW中的CY表示结果的百位值。

要点分析：

1）该指令必须紧跟在ADD或ADDC指令之后，且这里的ADD或ADDC的操作是对压缩的BCD数进行运算。

2）DA指令不影响溢出标志。

3） DA指令不能对减法进行十进制调整。

例 设（A）=0101 0110=56 BCD，（R3）= 0110 0111=67 BCD，（CY）=1。

执行指令： ADDC A，R3

DA A

执行 ADDC A，R3	（A）	0 1 0 1	0 1 1 0	（56 BCD）
	（R 3）	0 1 1 0	0 1 1 1	（67 BCD）
	+ （CY）		1	
		1 0 1 1	1 1 1 0	（高、低4位均大于9）
再执行 DA A	+	0 1 1 0	0 1 1 0	（加66H操作）
	CY=1	0 0 1 0	0 1 0 0	（124 BCD）

即BCD码数56+67+1=124。

3.5 算术运算指令

例 设计将两个4位压缩BCD码数相加程序。其中一个数存放在30H（存放十位，个位）、31H（存放千位，百位）存储器单元，另一个加数存放在32H（存放低位）、33H（存放高位）存储单元，和数存到30H, 31H单元。程序如下：

```
MOV R0，＃30H    ；地址指针指向一个加数的个位、十位
MOV R1，＃32H    ；另一个地址指针指向第二个加数的个位、十位
MOV A，@R0       ；一个加数送累加器
ADD A，@R1       ；两个加数的个位、十位相加
DA A             ；调整为BCD码数
MOV @R0，A       ；和数的个位、十位送30H单元
INC R0           ；两个地址指针分别指向两个加数的百位、千位
INC R1
MOV A，@R0       ；一个加数的百位、千位送累加器
ADDC A，@R1      ；两个加数的百位、千位和进位相加
DA A             ；调整为BCD码数
MOV @R0，A       ；和数的百位、千位送31H单元
```

3.5 算术运算指令

例 对累加器A中压缩BCD码数减1。

```
        ADD  A，#99H
        DA   A
HERE:   SJMP  HERE
```

说明：累加器A允许的最大BCD码数为99BCD，当对A实行加99BCD码数时，必然形成对BCD码百位数的进位，而剩在A中的内容正是压缩BCD码数减1。

如BCD的59H，经加99H和DA A的调整后，为58H且CY=1，不考虑进位CY，则BCD码59-1=58。

例 利用十进制加法调整指令DA作十进制减法调整。

分析: 由于DA指令不能直接对减法进行十进制调整，为了进行十进制减法运算，只能用加减数的补数来进行。两位十进制数是对100取补的，如60-30=30，也可改为补数相加：

60+（100-30）= 1 30

丢掉进位（模）100后，就得到正确的结果。

3.5 算术运算指令

在实际运算时，由于CPU为8位，不可能用9位二进制数表示十进制数100，但可用8位二进制数10011010（9AH）代替，因为这个二进制数经过十进制调整（加6）后就是100000000 BCD。 这样十进制无符号数的减法运算可按以下步骤进行：

(1) 求减数的补数（9AH-减数）；

(2) 被减数与减数的补数相加；

(3) 经DA指令调整后就得到所求的十进制减法运算结果。

这里用“补数”而不是“补码”是为了和带符号位的补码加以区别。 由于现在操作数都是正数，没有必要再加符号位，故称“补数”更为合适一些。

3.5 算术运算指令

设M1、 M2、 M3 分别为被减数、 减数和差的符号地址，相应的十进制减法运算程序如下：

```
CLR   C                ；CY清0
MOV  A，   ＃9AH    ；（A）←9AH
SUBB  A，   M2          ；求减数的补数
ADD  A，   M1           ；加补数完成减法
DA  A                  ；  十进制调整
MOV  M3，  A           ；  差存入M3单元
```

若程序执行前：（M1）=（91）BCD，（M2）=（36）BCD，则程序执行后：（M3）=（55）BCD。

3.5 算术运算指令

3.5.2 减法指令

一、带借（进）位减法指令带位减法指令（SUBB，Subtract with Borrow 4条）

SUBB A，#data ；（A）←（A）-data-CY

SUBB A，Rn ；（A）←（A）-（Rn）-CY

SUBB A，direct ；（A）←（A）-（direct）-CY

SUBB A，@Ri ；（A）←（A）-（（Ri））-CY

功能：累加器A中的内容减去源操作数中的内容及进位位CY，差值存入累加器A中。

CY：差的位7需借位时，（CY）=1；否则，（CY）=0。

AC：差的位3需借位时，（AC）=1；否则，（AC）=0。

OV：若位6有借位而位7无借位或位7有借位而位6无借位时，（OV）=1。

OV位用于带符号的整数减法。

51指令系统中没有不带借位的减法，如要用此组指令完成不带借位减法，只需先清CY为 0。 清CY有专门的指令，它属于位操作类指令（详见3.7节），指令为 CLR C ；（CY）← 0

3.5 算术运算指令

例 设（A）=49H，（CY）=1，分析执行指令SUBB A，#64H

0100 1001（49H）

0110 0100（64H）

\- 1

1 1110 0100

结果：（A）=E4H，（CY）=1，（P）=0，（AC）=0，（OV）=0。

例 设（A）=D9H，（R0）=87H，求减法操作后的结果。

解：程序为： CLR C ；清进位位

SUBB A，R0

分析， 11011001（D9H）

10000111（87H）

\- 0（CY）

01010010

结果：（A）=52H，（CY）=0，（AC）=0，（P）=1，（OV）=0。

3.5 算术运算指令

例 双字节无符号数相减（R0 R1）-（R2 R3）→（R4 R5）。R0、R2、R4存放16位数的高字节，R1、R3、R5存放低字节，先减低8位，后减高8位和低位减借位。由于低位开始减时没有借位，所以要先清零。其编程如下：

```
MOV  A，  R1   ；取被减数低字节
CLR  C         ；清借位位
SUBB  A，R3    ；低字节相减
MOV  R5，A     ；保存差低字节
MOV  A，  R0   ；取被减数高字节
SUBB  A，R2    ；两高字节差减低位借位
MOV  R4，A     ；保存差高字节
```

3.5 算术运算指令

二、减1指令（DEC，Decrease 4条）

DEC A ；（A）←（A）-1

DEC Rn ；（Rn）←（Rn）-1

DEC direct ；（direct）←（direct）-1

DEC @Ri ；（（Ri））←（（Ri））-1

功能：把操作数的内容减 1，结果再送回原单元。这组指令仅DEC A 影响P标志。其余指令都不影响PSW标志位的状态。

例 设（R0）=7FH，（7EH）=00H，（7FH）=40H。

执行指令：

DEC @R0 ；（7FH）-1=40H-1=3FH→（7FH）

DEC R0 ；（R0）-1=7FH-1=7EH→（R0）

DEC @R0 ；（7EH）-1=00H-1=FFH→（7EH）

执行结果：（R0）=7EH，（7EH）=FFH，（7FH）=3FH。

3.5 算术运算指令

3.5.3 乘法指令（MUL，Multiplication）

MUL　　AB　　；（B）(A)←（A）×（B）

功能：把累加器A和寄存器B中的8位无符号整数相乘，乘积为16位，乘积的低8位存于A中，高8位存于B中。指令执行对PSW的影响如下：

（1）当乘积大于FFH（255）时，溢出标志位（OV）=1，否则（OV）=0。

（2）标志CY总是被清0。

（3）P受累加器A中的内容影响。

例 设（A）＝50H（80D），（B）＝A0H（160D），执行指令：

MUL AB，即80×160＝12800＝3200H。

执行结果：乘积 3200H（12800），（A）＝00H，（B）＝32H，

（OV）＝1，（CY）＝0。

例 若（A）=4EH（78），（B）=5DH（93），执行指令：MUL AB

结果：积为（BA）=1C56H（7254）＞FFH（255），（A）=56H，

（B）=1CH，OV=1，CY=0，P=0。

3.5 算术运算指令

3.5.4 除法指令（DIV， Division）

DIV AB ；（A）←（A/B）的（商），（B）←（A/B）的（余数）

功能：把累加器A中的8位无符号整数除以寄存器B中8位无符号整数，商放在A中，余数放在B中。指令执行对PSW的影响如下：

（1）CY和OV置0。

（2）当除数（B）＝0时，除法运算没有意义，结果不定，则OV置1。CY总是清0。

（3）P受累加器A 中的内容影响。

例 设（A）＝FBH（251D），（B）＝12H（18D），执行指令：DIV AB

执行结果：（A）＝0DH（商为13），（B）＝11H（余数为17），

（OV）＝0，（CY）＝0。

例 若（A）=FBH（251），（B）=12H（18），执行指令 DIV AB 之后，

（A）=0DH，（B）=11H，（OV）=0，（CY）=0。

3.5 算术运算指令

乘除法指令要点分析：

（1）乘法指令和除法指令需要4个机器周期，是指令系统中执行时间最长的指令。

（2）MUL指令实现8位无符号数的乘法操作，两个乘数分别放在累加器A和寄存器B中，乘积为16位，低8位放在A中，高8位放在B中；DIV指令实现8位无符号数除法，被除数放在A中，除数放在B中，指令执行后，商放在A中而余数放在B中。

（3）在51单片机中，乘法和除法指令仅适用于8位数乘法和除法运算。如果被乘数、被除数和除数中有一个是16位数时，不能用两个指令。要进行多字节乘法和除法运算还需编写相应的程序。