第 5 章 STC 单片机 CPU 指令系统

CHAPTER 5

本章介绍 STC15 系列单片机 CPU 指令系统，主要内容包括 STC 单片机 CPU 寻址模式和 STC 单片机 CPU 指令集。

CPU 指令系统反映了中央处理单元 CPU 的结构。当指令系统确定后，CPU 内核的结构就确定了。通过对 STC 单片机 CPU 寻址模式和指令系统的学习，进一步掌握 STC 单片机 CPU 的结构和接口功能，为后续学习汇编语言和 C 语言程序设计打下基础。

特别需要强调的是只有掌握了 STC 单片机 8051 CPU 的结构及其指令系统后，才能使所编写的软件程序在运行时达到最高的性能要求。

5.1 STC 单片机 CPU 寻址模式

一条机器指令包含两部分，即操作码和操作数。操作码的目的是要对被操作对象进行处理。例如，对被操作对象实现逻辑与或非运算、加减乘除运算等。在机器/汇编语言指令中，将操作对象称为操作数。

在 STC 8051 单片机中，这些被操作的对象（操作数）可以保存在 CPU 的内部寄存器、片内 Flash 程序存储器、片内基本 RAM、片内扩展 RAM 或者片外扩展存储器中，也可能仅是一个常数（它作为指令的一部分存在）。

因此，就需要预先确定一些规则，一方面，使得操作数可以保存在这些区域内；另一方面，CPU 可以找到它们。在 STC 8051 单片机中，将 CPU 寻找操作对象（操作数）所在存储位置的方式，称为寻址模式。

在 STC 8051 单片机中，操作对象包括立即数、直接位地址、程序地址、直接地址、间接地址和特殊的汇编器符号。这些操作对象和寻址模式相关。

需要说明的是，特殊汇编器符号用来表示 8051 CPU 的内部功能寄存器，不可以修改这些符号。在 8051 单片机常用的寄存器符号有：

（1）A 表示 8051 的累加器 ACC。

（2）DPTR 表示 16 位的数据指针，指向外部数据空间或者代码存储空间。

（3）PC 表示 16 位的程序计数器，指向下一条将要执行指令的地址。

（4）C 表示进位标志 CY。

（5）AB 表示 A 和 B 寄存器对，用于乘和除操作。

（6）R0～R7 表示当前所使用寄存器组内的 8 个 8 位的通用寄存器。

(7) SP 表示堆栈指针。

(8) DPS 数据指针选择寄存器。

STC15 系列单片机采用的是 8051 CPU 内核,所以其寻址模式和传统 8051 单片机的寻址模式完全一样。

5.1.1 立即数寻址模式

一些指令直接加载常数的值,而不是地址。例如指令:

```
MOV A, #3AH
```

功能是将 8 位的十六进制立即数 3A 送给累加器,如图 5.1 所示。

图 5.1 立即数寻址模式

5.1.2 直接寻址模式

操作数由一个直接 8 位地址域指定。当使用这种模式时,只能访问片内 RAM 和特殊功能寄存器 SFR。例如指令:

```
MOV A,3AH
```

功能是将片内 RAM 中 3AH 单元中的数据送给累加器 A,如图 5.2 所示。

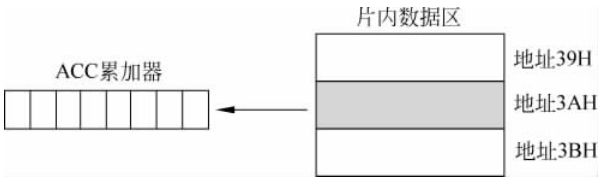

图 5.2 直接寻址模式

注:MOV A,3AH 和 MOV A,#3AH 指令的区别:如果操作数前带“#”符号,则操作数表示的是一个立即数,是立即数寻址方式;而操作数前面不带“#”符号,则操作数表示的是存储器的地址,3A 是存储器的地址,表示把存储器地址为 3A 的内容送到寄存器 A 中。

5.1.3 间接寻址模式

由指令指定一个寄存器,该寄存器包含操作数的地址。寄存器 R0 和 R1 用来指定 8 位地址,数据指针寄存器(DPTR)用来指定 16 位的地址。例如指令:

```
ANL A,@R1
```

假设累加器 A 的内容为 31H,R1 寄存器的内容为 60H,即(R1)=60H,则以 60H 作为存储器的地址,将 60H 地址单元的内容与累加器 A 中的数 31H 进行逻辑与运算,运算结果保存在累加器 A 中,如图 5.3 所示。

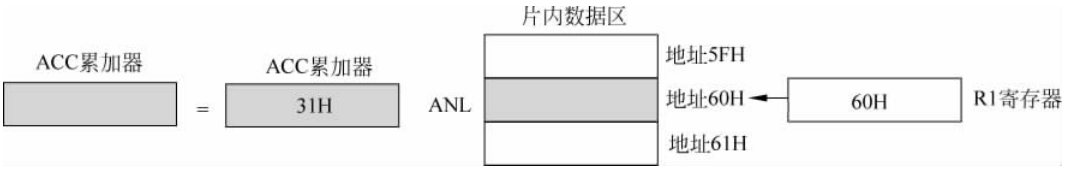

图 5.3　间接寻址模式

5.1.4　寄存器寻址模式

某些特定指令用来访问寄存器组中的 R0～R7 寄存器、累加器 A、通用寄存器 B、地址寄存器和进位 CY。由于这些指令不需要地址域，因此这些指令访问效率更高。例如指令：

```
INC R0
```

功能是将寄存器 R0 的内容加 1，再送回 R0，如图 5.4 所示（假设当前寄存器 R0 中的数为 50H）。

图 5.4　寄存器寻址模式

5.1.5　相对寻址模式

相对寻址时将程序计数器（PC）中的当前值与指令中第二字节给出的数相加，其结果作为转移指令的目的转移地址。PC 中的当前值为基地址，指令第二字节给出的数作为偏移量。由于目的地址是相对于 PC 中的基地址而言，所以这种寻址方式称为相对寻址。偏移量为带符号的数，取值范围为－128～＋127。这种寻址方式主要用于跳转指令，例如指令：

```
JC 80H
```

注：该指令为两字节指令，操作码 JC 占用一字节，80H 占用另一字节。

功能是当进位标志为 1 时，则进行跳转，如图 5.5 所示。

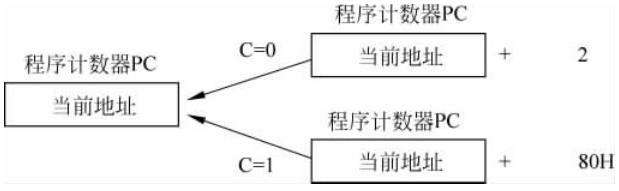

图 5.5　相对寻址模式

5.1.6　变址寻址模式

变址寻址模式使用数据指针作为基地址，累加器值作为偏移地址来读取程序存储器。例如指令：

```
MOVC A,@A + DPTR
```

功能是将 DPTR 和 A 的内容相加所得到的值作为程序存储器的地址，并将该地址单元的内容送 A，如图 5.6 所示。

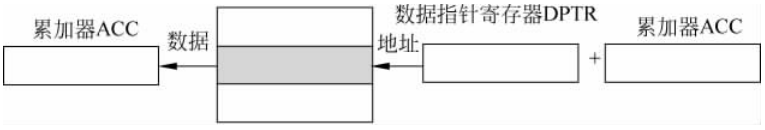

图 5.6　变址寻址模式

5.1.7　位寻址模式

位寻址是对一些内部数据存储器 RAM 和特殊功能寄存器 SFR 进行位操作时的寻址模式。在进行位操作时，指令操作数直接给出该位的地址，然后根据操作码的类型对该位进行操作。在这种模式下，操作数是 256 比特中的某一位。例如指令：

```
MOV C,2BH
```

功能是把位寻址区位地址为 2BH 的位状态送进位标志 C，如图 5.7 所示。

图 5.7　位寻址模式

思考与练习

5-1　在 STC 8051 单片机中，共有_____种寻址模式。

5-2　请说明在 STC 8051 单片机中，操作数可以存放的位置。

5-3　在 STC 8051 单片机中，寻址模式是指_____。

5-4　参考 STC 寻址模式和 STC 单片机 CPU 指令集，说明下面指令的寻址方式：

(1) MOV DRTR，#1234H，寻址方式_____。

(2) MUL AB，寻址方式_____。

(3) SETB C，寻址方式_____。

(4) MOV A，12H，寻址方式_____。

(5) MOVC A，@A＋PC，寻址方式_____。

(6) LJMP 100H，寻址方式_____。

(7) MOV A，@R1，寻址方式_____。

5.2　STC 单片机 CPU 指令集

STC15 系列单片机内的 8051 CPU 指令集包含 111 条指令，这些指令与传统的 8051 指令完全兼容，但是大幅度提高了执行指令的时间效率。表现在：

(1) 采用了 STC-Y5 超高速 CPU 内核，在相同的时钟频率下，速度又比 STC 早期的 IT 系列单片机(如 STC12 系列/STC11 系列/STC10 系列)快 20%。

(2) 典型的,其中 INC DPTR 和 MUL AB 指令的执行速度大幅提高 24 倍。

(3) 在指令集中有 22 条指令,这些指令可以在一个周期内执行完,平均速度提高 8～12 倍。

(4) 将 111 条指令全部执行完一遍所需的时钟周期数为 283,而传统的 8051 单片机将 111 条指令全部执行完所需要的时钟周期数为 1944。

按照所实现的功能,将 STC15 单片机内 8051 CPU 指令集分为算术指令、逻辑指令、数据传输指令、布尔指令和程序分支指令。

5.2.1　算术指令

算术指令支持直接、间接、寄存器、立即数和寄存器指定指令寻址方式。算术模式用于加、减、乘、除、递增和递减操作。

1. 加法指令

1) ADD A,Rn

该指令将寄存器 Rn 的内容和累加器 A 的内容相加,结果保存在累加器 A 中,如表 5.1 所示,并设置 CY 标志、AC 标志以及溢出标志 OV。

表 5.1　ADD A,Rn 指令的内容

助记符	操作	标志	机器码	字节数	周期数
ADD A,Rn	(PC)← (PC) + 1 (A)← (A) + (Rn)	CY,AC,OV	00101rrr	1	1

注：rrr 为寄存器的编号,因此机器码范围是 28H～2FH。

(1) 当和的第 3 位和第 7 位有进位时,分别将 CY、AC 标志置位,否则置 0。

(2) 对于带符号运算数,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,否则清 0。或者可以这样说,当两个正数相加时,相加的结果为负数;或当两个负数相加时,相加的结果为正数时,在这两种情况下设置 OV 为 1。

【例 5-1】 假设累加器 A 中的数据为 C3H,R0 寄存器中的数据为 AAH。当执行指令:

```
ADD A,R0
```

结果:

(A)＝6DH,(AC)＝0,(CY)＝1,(OV)＝1

注：()表示内容。

计算过程为

$$\begin{array}{r} 1100,0011 \\ +\quad 1010,1010 \\ \hline 1,0110,1101 \end{array}$$

2) ADD A,direct

该指令将直接寻址单元的内容和累加器 A 的内容相加,结果保存在累加器 A 中,如表 5.2 所示。CY、AC、OV 标志的设置同上。

表 5.2 ADD A,direct 指令的内容

助记符	操作	标志	操作码	字节数	周期数
ADD A,direct	(PC) ← (PC) + 2 (A) ← (A) + (direct)	CY,AC,OV	00100101	2	2

注：在操作码后面跟着一字节的直接地址。

3) ADD A,@Ri

该指令将间接寻址单元的内容和累加器 A 的内容相加，结果保存在累加器 A 中，如表 5.3 所示。CY、AC、OV 标志的设置同上。

表 5.3 ADD A,@Ri 指令的内容

助记符	操作	标志	操作码	字节数	周期数
ADD A,@Ri	(PC)← (PC) + 1 (A)← (A) + ((Ri))	CY,AC,OV	0010011i	1	2

注：i 表示 R0 或者 R1。当 i=0 时，表示 R0 寄存器；当 i=1 时，表示 R1 寄存器。

4) ADD A,#data

该指令将一个立即数和累加器 A 的内容相加，结果保存在累加器 A 中，如表 5.4 所示。CY、AC、OV 标志的设置同上。

表 5.4 ADD A,#data 指令的内容

助记符	操作	标志	操作码	字节数	周期数
ADD A,#data	(PC)← (PC) + 2 (A)← (A) + data	CY,AC,OV	00100100	2	2

注：在操作码后面跟着一字节的立即数。

5) ADDC A,Rn

该指令将寄存器 Rn 的内容与累加器 A 的内容及进位标志 CY 的内容相加，结果保存在累加器 A 中，如表 5.5 所示。CY、AC、OV 标志的设置同上。

表 5.5 ADDC A,Rn 指令的内容

助记符	操作	标志	操作码	字节数	周期数
ADDC A,Rn	(PC)← (PC) + 1 (A)← (A) +(C)+ (Rn)	CY,AC,OV	00111rrr	1	1

注：rrr 为寄存器的编号，因此机器码范围是 38H～3FH。

【例 5-2】 假设累加器 A 中的数据为 C3H，R0 寄存器中的数据为 AAH，进位标志为 1 时，执行指令：

```
ADDC A,R0
```

结果：

(A)=6EH，(AC)=0，(CY)=1，(OV)=1

注：计算过程为

$$\begin{array}{r} 1100,0011 \\ 1010,1010 \\ +\qquad\quad 1 \\ \hline 1,0110,1110 \end{array}$$

6）ADDC A,direct

该指令将直接寻址单元的内容与累加器 A 的内容及进位标志 CY 中的内容相加，结果保存在累加器 A 中，如表 5.6 所示。CY、AC、OV 标志的设置同上。

表 5.6　ADDC A,direct 指令的内容

助记符	操作	标志	操作码	字节数	周期数
ADDC A,direct	(PC)←(PC) + 2 (A)←(A) +(C) + (direct)	CY,AC,OV	00110101	2	2

注：在操作码后面跟着一字节的直接地址。

7）ADDC A,@Ri

该指令将间接寻址单元的内容与累加器 A 的内容及进位标志 CY 中的内容相加，结果保存在累加器 A 中，如表 5.7 所示。CY、AC、OV 标志的设置同上。

表 5.7　ADDC A,@Ri 指令的内容

助记符	操作	标志	操作码	字节数	周期数
ADDC A,@Ri	(PC)←(PC) + 1 (A)←(A) +(C) + ((Ri))	CY,AC,OV	0011011i	1	2

注：i 表示 R0 或者 R1。当 i=0 时，表示 R0 寄存器；当 i=1 时，表示 R1 寄存器。

8）ADDC A,#data

该指令将一个立即数与累加器 A 的内容及进位标志 CY 中的内容相加，结果保存在累加器 A 中，如表 5.8 所示。CY、AC、OV 标志的设置同上。

表 5.8　ADDC A,#data 指令的内容

助记符	操作	标志	操作码	字节数	周期数
ADDC A,#data	(PC)←(PC) + 2 (A)←(A) +(C)+ data	CY,AC,OV	00110100	2	2

注：在操作码后面跟着一字节的立即数。

2. 减法指令

1）SUBB A,Rn

该指令从累加器 A 中减去寄存器 Rn 和进位标志 CY 内的内容，将结果保存在累加器 A 中，如表 5.9 所示。

表 5.9　SUBB A,Rn 指令的内容

助记符	操作	标志	操作码	字节数	周期数
SUBB A,Rn	(PC)←(PC) + 1 (A)←(A) − (C) − (Rn)	CY,AC,OV	10011rrr	1	1

注：rrr 为寄存器的编号，因此机器码范围是 98H～9FH。

(1) 如果第 7 位需要一个借位，则设置进位(借位)标志，否则清除 CY 标志。

注：如果在执行指令前，已经设置了 CY 标志，则表示前边的多个步骤需要一个借位。这样，从累加器中减去进位标志以及源操作数。

(2) 如果第 3 位需要一个借位，则设置 AC 标志；否则清除 AC 标志。

(3) 如果第 6 位需要借位，而 7 位没有借位；或者第 7 位有借位，而第 6 位没有借位，在这两种情况下都会设置 OV 标志。或者可以这样说，当减去有符号的整数时，当一个正数减去一个负数，产生一个负数结果时；或者一个负数减去一个正数时，产生一个正数结果时，设置 OV 标志。

【例 5-3】 假设累加器 A 中的数据为 C9H，R2 寄存器中的数据为 54H，进位标志为 1 时，执行指令：

```
SUBB A,R2
```

结果：

(A)＝74H，(AC)＝0，(CY)＝0，(OV)＝1

注：计算过程为

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

2) SUBB A,direct

该指令从累加器 A 中减去直接寻址单元的内容和进位标志 CY 的内容，然后结果保存在累加器 A 中，如表 5.10 所示。CY、AC、OV 设置同上。

表 5.10　SUBB A,direct 指令的内容

助记符	操作	标志	操作码	字节数	周期数
SUBB A,direct	(PC)←(PC) + 2 (A)←(A) − (C) − (direct)	CY,AC,OV	10010101	2	2

注：在操作码后面跟着一字节的直接地址。

3) SUBB A,@Ri

该指令从累加器 A 中减去间接寻址单元的内容和进位标志 CY 的内容，然后结果保存在累加器 A 中，如表 5.11 所示。CY、AC、OV 设置同上。

表 5.11　SUBB A,@Ri 指令的内容

助记符	操作	标志	操作码	字节数	周期数
SUBB A,@Ri	(PC)←(PC) + 1 (A)←(A) − (C) − ((Ri))	CY,AC,OV	1001011i	1	2

注：i 表示 R0 或者 R1。当 i=0 时，表示 R0 寄存器；当 i=1 时，表示 R1 寄存器。

4) SUBB A,#data

该指令从累加器 A 中减去一个立即数和进位标志 CY 的内容，然后结果保存在累加器 A 中，如表 5.12 所示。CY、AC、OV 设置同上。

表 5.12　SUBB A,#data 指令的内容

助记符	操作	标志	操作码	字节数	周期数
SUBB A,#data	(PC)←(PC) + 2 (A)←(A) − (C) − data	CY,AC,OV	10010100	2	2

注：在操作码后面跟着一字节的立即数。

3. 递增指令

1) INC A

该指令将累加器 A 的内容加 1，结果保存在累加器 A 中，如表 5.13 所示。若累加器 A 的结果为 0xFF 时，在执行完该指令后，将其内容设置为 0。

表 5.13　INC A 指令的内容

助记符	操作	标志	操作码	字节数	周期数
INC A	(PC)←(PC) + 1 (A)←(A) + 1	N	00000100	1	1

2) INC Rn

该指令将寄存器 Rn 的内容加 1，结果保存在 Rn 中，如表 5.14 所示。若 Rn 的结果为 0xFF 时，在执行完该指令后，将其内容设置为 0。

表 5.14　INC Rn 指令的内容

助记符	操作	标志	操作码	字节数	周期数
INC Rn	(PC)←(PC) + 1 (Rn)←(Rn) + 1	N	00001rrr	1	2

注：rrr 为寄存器的编号，因此机器码范围是 08H～0FH。

3) INC direct

该指令将直接寻址单元的内容加 1，结果保存在直接地址单元中，如表 5.15 所示。若直接地址单元的结果为 0xFF 时，在执行完该指令后，将其内容设置为 0。

表 5.15　INC direct 指令的内容

助记符	操作	标志	操作码	字节数	周期数
INC direct	(PC)←(PC) + 2 (direct)←(direct) + 1	N	00000101	2	3

注：在操作码后面跟着一字节的直接地址。

4) INC @Ri

该指令将间接寻址单元的内容加 1，结果保存在间接地址单元中，如表 5.16 所示。若间接地址单元的结果为 0xFF 时，在执行完该指令后，将其内容设置为 0。

表 5.16　INC @Ri 指令的内容

助记符	操作	标志	操作码	字节数	周期数
INC @Ri	(PC)←(PC) + 1 ((Ri))←((Ri)) + 1	N	0000011i	1	3

注：i 表示 R0 或者 R1。当 i=0 时，表示 R0 寄存器；当 i=1 时，表示 R1 寄存器。

5) INC DPTR

该指令将 DPTR 的内容加 1，结果保存在 DPTR 中，如表 5.17 所示。若 DPTR 的结果为 0xFFFF 时，在执行完该指令后，将其内容设置为 0x0000。

表 5.17　INC DPTR 指令的内容

助记符	操作	标志	操作码	字节数	周期数
INC DPTR	(PC)←(PC) + 1 (DPTR)←(DPTR) + 1	N	10100011	1	1

【例 5-4】 假设寄存器 R0 中的数据为 7EH，内部 RAM 地址为 7EH 和 7FH 单元的数据分别为 FFH 和 40H，即(7E)=FFH，(7F)=40H，则当执行指令：

```
INC @R0        ;内部 RAM 地址为 7EH 单元的内容加 1，变成 00H
INC R0         ;寄存器 R0 中的数据变为 7FH
INC @R0        ;内部 RAM 地址为 7FH 单元的内容加 1，变成 41H
```

结果：

(R0)=7FH，内部 RAM 地址为 7EH 和 7FH 单元的数据变为 00H 和 41H。

4. 递减指令

1) DEC A

该指令将累加器 A 的内容减 1，结果保存在累加器 A 中，如表 5.18 所示。如果累加器 A 中的内容为 0，在执行完该指令后，则变为 0xFF。

表 5.18　DEC A 指令的内容

助记符	操作	标志	操作码	字节数	周期数
DEC A	(PC)←(PC) + 1 (A)←(A) − 1	N	00010100	1	1

2) DEC Rn

该指令将寄存器 Rn 的内容减 1,结果保存在寄存器 Rn 中,如表 5.19 所示。如果 Rn 的内容为 0,在执行完该指令后,则变为 0xFF。

表 5.19　DEC Rn 指令的内容

助记符	操作	标志	操作码	字节数	周期数
DEC Rn	(PC)← (PC) + 1 (Rn)← (Rn)−1	N	00011rrr	1	2

注：rrr 为寄存器的编号,因此机器码范围是 18H～1FH。

3) DEC direct

该指令将直接寻址单元的内容减 1,结果保存在直接地址单元中,如表 5.20 所示。如果直接寻址单元的内容为 0,在执行完该指令后,则变为 0xFF。

表 5.20　DEC direct 指令的内容

助记符	操作	标志	操作码	字节数	周期数
DEC direct	(PC)← (PC) + 2 (direct) ← (direct)−1	N	00010101	2	3

注：在操作码后面跟着一字节的直接地址。

4) DEC @Ri

该指令将间接寻址单元的内容减 1,结果保存在间接地址单元中,如表 5.21 所示。如果间接寻址单元的内容为 0,在执行完该指令后,则变为 0xFF。

表 5.21　DEC @Ri 指令的内容

助记符	操作	标志	操作码	字节数	周期数
DEC @Ri	(PC)← (PC) + 1 ((Ri)) ← ((Ri))−1	N	0001011i	1	3

注：i 表示 R0 或者 R1。当 i=0 时,表示 R0 寄存器；当 i=1 时,表示 R1 寄存器。

【例 5-5】 假设寄存器 R0 中的数据为 7FH,内部 RAM 地址为 7EH 和 7FH 单元的数据分别为 00H 和 40H,即(7F)=00H,(7E)=40H,则当执行指令：

```
DEC @R0        ;内部 RAM 地址为 7FH 单元的内容减 1,变成 FFH
DEC R0         ;寄存器 R0 中的数据变为 7EH
DEC @R0(此时 R0 中的数据为 40H);内部 RAM 地址为 7EH 单元的内容减 1,变成 3FH
```

结果：

(R0)=7EH,内部 RAM 地址为 7EH 和 7FH 单元的数据变为 FFH 和 3FH。

5. 乘法指令

MUL AB 指令将累加器 A 和寄存器 B 中的两个无符号 8 位二进制数相乘,所得的 16 位乘积的低 8 位结果保存在累加器 A 中,高 8 位结果保存在寄存器 B 中,如表 5.22 所示。

如果乘积大于 255,则溢出标志 OV 置 1；否则 OV 清零。在执行该命令时,总是清除进位标志 CY。

表 5.22 MUL AB 指令的内容

助记符	操作	标志	操作码	字节数	周期数
MULAB	(PC)←(PC) + 1 (A)←(A) × (B)结果的第 7 位～第 0 位 (B)←(A) × (B)结果的第 15 位～第 8 位	CY,OV	10100100	1	2

【例 5-6】 假设累加器 A 中的数据为$(80)_{10}$=50H,寄存器 B 中的数据为$(160)_{10}$=A0H,则执行指令:

```
MUL AB
```

结果:

乘积为$(12800)_{10}$=3200H,(A)=00H,(B)=32H,(CY)=0,(OV)=1。

注:

$$
\begin{array}{r}
01010000 \\
\times\quad 10100000 \\
\hline
0\ 0000000 \\
0\ 0000000 \\
0\ 0000000 \\
0\ 0000000 \\
0\ 0000000 \\
01010000 \\
00000000 \\
+\quad 01010000 \\
\hline
0011001000000000
\end{array}
$$

6. 除法指令

DIV AB 指令用累加器 A 中的无符号整数除以寄存器 B 中无符号整数。所得的商保存在累加器 A 中,余数保存在寄存器 B 中,如表 5.23 所示。当除数(B 寄存器的内容)为 0 时,结果不定,溢出标志 OV 置 1。在执行该指令时,清除进位标志 CY。

表 5.23 DIV AB 指令的内容

助记符	操作	标志	操作码	字节数	周期数
DIV	(PC)←(PC) + 1 $(A)_{15\text{-}8}$←(A)/(B) $(B)_{7\text{-}0}$←(A)/(B)	CY,OV	10000100	1	6

【例 5-7】 假设累加器 A 中的数据为$(251)_{10}$=FBH,寄存器 B 中的数据为$(18)_{10}$=12H,则执行指令:

```
DIV AB
```

结果:

(A)=0DH,(B)=11H,(CY)=0,(OV)=0。

注：计算过程为

```
          00001101
00010010 ) 11111011
         - 10010
           11010
         - 10010
           100011
         - 010010
             10001
```

7. BCD 调整指令

DA A 指令的功能是对 BCD 码的加法结果进行调整。两个压缩型 BCD 码按十进制数相加后，需经该指令的调整才能得到压缩型 BCD 码的和数，如表 5.24 所示。

表 5.24　DA A 指令的内容

助记符	操作	标志	操作码	字节数	周期数
DA A	(PC)←(PC) + 1 如果{[(A_{3-0}) > 9] V [(AC) = 1]}，则 (A_{3-0}) ←(A_{3-0}) + 6 如果{[(A_{7-4}) > 9] V [(C) = 1]}，则 (A_{7-4}) ←(A_{7-4}) + 6	CY	11010100	1	3

本指令是根据 A 的最初数值和程序状态字 PSW 的状态，决定对 A 执行加 06H、60H 或 66H 操作。

注：如果前面没有使用加法运算，则不能直接使用 DA 指令。此外，如果前面执行的是减法运算，则 DA 指令也不起任何作用。

【例 5-8】 假设累加器 A 中的数据为 56H，表示十进制数 56 的 BCD 码。寄存器 R3 的内容为 67H，表示十进制数 67 的 BCD 码。进位标志为 1，则执行指令：

```
ADDC   A,R3      ;累加器 A 的结果为 BEH,(AC) = 0,(CY) = 0
DA     A         ;表示十进制数的 124
```

结果表示为(A)＝124。

注：

因为在执行完 ADDC 指令后，(A)＝BEH，

($A)_{3-0}$＞9，所以($A)_{3-0}$＋6→($A)_{3-0}$＝4H，向第 4 位有进位。

($A)_{7-4}$＞9，所以($A)_{7-4}$＋6＋进位→($A)_{7-4}$＝2H，最高位有进位。

思考与练习

5-5　假定(A)＝66H，(R0)＝55H，(55H)＝FFH 时，执行指令：

```
ADD A, @R0
```

(A)＝_____，(CY)＝_____，(OV)＝_____。

5-6　如果(A)＝60H，(B)＝73H，(CY)＝1 时，执行指令：

```
ADDC A, B
```

(A)＝_____，(CY)＝_____，(OV)＝_____。

5-7　如果(A)=3DH,(B)=4EH,执行指令：

```
MUL AB
```

(A)=_____,(B)=_____。

5.2.2　逻辑指令

逻辑指令执行布尔操作,比如逻辑与,逻辑或,逻辑异或操作,对累加器内容进行旋转和累加器半字交换。

1. 逻辑与指令

1) ANL A, Rn

该指令将累加器 A 的内容和寄存器 Rn 的内容做逻辑与操作,结果保存在累加器 A 中,如表 5.25 所示。

表 5.25　ANL A,Rn 指令的内容

助记符	操作	标志	操作码	字节数	周期数
ANL A,Rn	(PC)←(PC) + 1 (A)←(A) ∧ (Rn)	N	01011rrr	1	1

注：rrr 为寄存器的编号,因此机器码范围是 58H～5FH。

【例 5-9】　假设累加器 A 中的数据为 C3H,寄存器 R0 的内容为 55H,则执行指令：

```
ANL A,R0
```

结果：

累加器 A 中的数据为 41H。

注：

$$\begin{array}{rr} & 11000011 \\ \wedge & 01010101 \\ \hline & 01000001 \end{array}$$

【例 5-10】　执行指令

```
ANL P1, #01110011B
```

结果：

将端口 1 的第 7 位、第 3 位和第 2 位清零。

2) ANL A, direct

该指令将累加器 A 的内容和直接寻址单元的内容做逻辑与操作,结果保存在累加器 A 中,如表 5.26 所示。

表 5.26　ANL A,direct 指令的内容

助记符	操作	标志	操作码	字节数	周期数
ANL A,direct	(PC)←(PC) + 2 (A)←(A) ∧ (direct)	N	01010101	2	2

注：在操作码后面跟着一字节的直接地址。

3) ANL A，@Ri

该指令将累加器 A 的内容和间接寻址单元中的内容做逻辑与操作，结果保存在累加器 A 中，如表 5.27 所示。

表 5.27　ANL A,@Ri 指令的内容

助记符	操作	标志	操作码	字节数	周期数
ANL A,@Ri	(PC)←(PC) + 1 (A)←(A) ∧ ((Ri))	N	0101011i	1	2

注：i 表示 R0 或者 R1。当 i=0 时，表示 R0 寄存器；当 i=1 时，表示 R1 寄存器。

4) ANL A，#data

该指令将累加器 A 的内容和立即数做逻辑与操作，结果保存在累加器 A 中，如表 5.28 所示。

表 5.28　ANL A,#data 指令的内容

助记符	操作	标志	操作码	字节数	周期数
ANL A,#data	(PC)←(PC) + 2 (A)←(A) ∧ data	N	01010100	2	2

注：在操作码后面跟着一字节的立即数。

5) ANL direct，A

该指令将累加器 A 的内容和直接寻址单元的内容做逻辑与操作，结果保存在直接寻址单元中，如表 5.29 所示。

表 5.29　ANL direct,A 指令的内容

助记符	操作	标志	操作码	字节数	周期数
ANL direct，A	(PC)←(PC) + 2 (direct)←(direct) ∧ (A)	N	01010010	2	3

注：在操作码后面跟着一字节的直接地址。

6) ANL direct，#data

该指令对立即数和直接寻址单元的内容做逻辑与操作，结果保存在直接寻址单元中，如表 5.30 所示。

表 5.30　ANL direct,#data 指令的内容

助记符	操作	标志	操作码	字节数	周期数
ANL direct，#data	(PC)←(PC) + 3 (direct)←(direct) ∧ data	N	01010011	3	3

注：在操作码后面跟着一字节的直接地址和一字节的立即数。

2. 逻辑或指令

1) ORL A，Rn

该指令将累加器 A 的内容和寄存器 Rn 中内容做逻辑或操作，结果保存在累加器 A

中，如表 5.31 所示。

表 5.31　ORL A,Rn 指令的内容

助记符	操作	标志	操作码	字节数	周期数
ORL A,Rn	(PC)←(PC) + 1 (A)←(A) ∨ (Rn)	N	01001rrr	1	1

注：rrr 为寄存器的编号，因此机器码范围是 48H～4FH。

【例 5-11】 假设累加器 A 中的数据为 C3H，寄存器 R0 的内容为 55H，则执行指令：

```
ORL A,R0
```

结果：

累加器 A 中的数据为 D7H。

注：计算过程为

$$\begin{array}{rr} & 11000011 \\ \vee & 01010101 \\ \hline & 11010111 \end{array}$$

【例 5-12】 执行指令

```
ORL P1, #00110010B
```

结果：

将端口 1 的第 5 位、第 4 位和第 1 位置 1。

2) ORL A, direct

该指令将累加器 A 的内容和直接寻址单元的内容做逻辑或操作，结果保存在累加器 A 中，如表 5.32 所示。

表 5.32　ORL A,direct 指令的内容

助记符	操作	标志	操作码	字节数	周期数
ORL A,direct	(PC)←(PC) + 2 (A)←(A) ∨ (direct)	N	01000101	2	2

注：在操作码后面跟着一字节的直接地址。

3) ORL A, @Ri

该指令将累加器 A 的内容和间接寻址单元中内容做逻辑或操作，结果保存在累加器 A 中，如表 5.33 所示。

表 5.33　ORL A,@Ri 指令的内容

助记符	操作	标志	操作码	字节数	周期数
ORL A,@Ri	(PC)←(PC) + 1 (A)←(A) ∨ ((Ri))	N	0100011i	1	2

注：i 表示 R0 或者 R1。当 i=0 时，表示 R0 寄存器；当 i=1 时，表示 R1 寄存器。

4) ORL A，#data

该指令将累加器 A 的内容和立即数做逻辑或操作，结果保存在累加器 A 中，如表 5.34 所示。

表 5.34　ORL A，#data 指令的内容

助记符	操作	标志	操作码	字节数	周期数
ORL A，#data	(PC)←(PC) + 2 (A)←(A) ∨ data	N	01000100	2	2

注：在操作码后面跟着一字节的立即数。

5) ORL direct，A

该指令将直接寻址单元的内容和累加器 A 中内容做逻辑或操作，结果保存在直接寻址单元中，如表 5.35 所示。

表 5.35　ORL direct，A 指令的内容

助记符	操作	标志	操作码	字节数	周期数
ORL direct，A	(PC) ←(PC) + 2 (direct) ←(direct) ∨ (A)	N	01000010	2	3

注：在操作码后面跟着一字节的直接地址。

6) ORL direct，#data

该指令将直接寻址单元中内容和立即数做逻辑或操作，结果保存在直接寻址单元中，如表 5.36 所示。

表 5.36　ORL direct，#data 指令的内容

助记符	操作	标志	操作码	字节数	周期数
ORL direct，#data	(PC)←(PC) + 3 (direct)←(direct) $\dot{\vee}$ data	N	01000011	3	3

注：在操作码后面跟着一字节的直接地址和一字节的立即数。

3. 逻辑异或指令

1) XRL A，Rn

该指令将累加器 A 的内容和寄存器 Rn 的内容做逻辑异或操作，结果保存在累加器 A 中，如表 5.37 所示。

表 5.37　XRL A，Rn 指令的内容

助记符	操作	标志	操作码	字节数	周期数
XRL A，Rn	(PC)←(PC) + 1 (A)←(A) ∨(Rn)	N	01101rrr	1	1

注：rrr 为寄存器的编号，因此机器码范围是 68H～6FH。

【例 5-13】 假设累加器 A 中的数据为 C3H，寄存器 R0 的内容为 AAH，则执行指令：

```
XRL A,R0
```

结果：

累加器 A 中的数据为 69H。

注：计算过程为

$$\begin{array}{rr} & 11000011 \\ \forall & 10101010 \\ \hline & 01101001 \end{array}$$

【例 5-14】 执行指令

```
XRL P1, #00110001B
```

结果：

将端口 1 的第 5 位、第 4 位和第 0 位取反。

2) XRL A，direct

该指令将累加器 A 的内容和直接寻址单元的内容做逻辑异或操作，结果保存在累加器 A 中，如表 5.38 所示。

表 5.38 XRL A,direct 指令的内容

助记符	操作	标志	操作码	字节数	周期数
XRL A,direct	(PC)←(PC) + 2 (A)←(A) ∀(direct)	N	01100101	2	2

注：在操作码后面跟着一字节的直接地址。

3) XRL A,@Ri

该指令将累加器 A 的内容和间接寻址单元的内容做逻辑异或操作，结果保存在累加器 A 中，如表 5.39 所示。

表 5.39 XRL A,@Ri 指令的内容

助记符	操作	标志	操作码	字节数	周期数
XRL A,@Ri	(PC)←(PC) + 1 (A)←(A) ∀((Ri))	N	0110011i	1	2

注：i 表示 R0 或者 R1。当 i=0 时，表示 R0 寄存器；当 i=1 时，表示 R1 寄存器。

4) XRL A，#data

该指令将累加器 A 的内容和一个立即数做逻辑异或操作，结果保存在累加器 A 中，如表 5.40 所示。

表 5.40 XRL A,#data 指令的内容

助记符	操作	标志	操作码	字节数	周期数
XRL A,#data	(PC)←(PC) + 2 (A)←(A) ∀ data	N	01100100	2	2

注：在操作码后面跟着一字节的立即数。

5) XRL direct,A

该指令将直接寻址单元的内容和累加器 A 的内容做逻辑异或操作,结果保存在直接寻址的单元中,如表 5.41 所示。

表 5.41　XRL direct,A 指令的内容

助记符	操作	标志	操作码	字节数	周期数
XRL direct, A	(PC)← (PC) + 2 (direct)← (direct) ∀ (A)	N	01100010	2	3

注：在操作码后面跟着一字节的直接地址。

6) XRL direct, #data

该指令将直接寻址的内容和一个立即数做逻辑异或操作,结果保存在直接寻址的单元中,如表 5.42 所示。

表 5.42　XRL direct, #data 指令的内容

助记符	操作	标志	操作码	字节数	周期数
XRL direct, #data	(PC)← (PC) + 3 (direct)← (direct) ∀ data	N	01100011	3	3

注：在操作码后面跟着一字节的直接地址和一字节的立即数。

4. 清除指令

CLR A 指令将累加器 A 中的各位清 0,如表 5.43 所示。

表 5.43　CLR A 指令的内容

助记符	操作	标志	操作码	字节数	周期数
CLR A	(PC)← (PC) + 1 (A)← 0	N	11100100	1	1

【例 5-15】 假设累加器 A 中的数据为 5CH,则执行指令:

```
CLR A
```

结果:

(A)=00H。

5. 取反指令

CPL A 指令将累加器 A 按位取反,将累加器 A 各位中的逻辑 1 变成逻辑 0,逻辑 0 变成逻辑 1,如表 5.44 所示。

表 5.44　CPL A 指令的内容

助记符	操作	标志	操作码	字节数	周期数
CPL A	(PC)← (PC) + 1 (A)← ($\overline{A}$)	N	11110100	1	1

【例 5-16】 假设 P1 端口的数据为 5BH,则执行指令:

```
CPL P1.1
```

```
CPL P1.2
```

结果：

将 P1 端口设置为 5DH＝01011101B。

6. 移位指令

1) RL A

该指令将累加器 A 中的内容循环左移，如表 5.45 所示。

表 5.45 RL A 指令的内容

助记符	操作	标志	操作码	字节数	周期数
RL A	(PC)←(PC) + 1 (A_{n+1}) ←(A_n),n＝0～6 (A_0) ←(A_7)	N	00100011	1	1

【例 5-17】 假设累加器 A 的数据为 C5H(11000101B)，则执行指令：

```
RL A
```

结果：

累加器 A 的数据变成 8BH＝10001011B。

2) RLC A

该指令将累加器 A 的内容和进位标志 CY 一起循环左移，如表 5.46 所示。

表 5.46 RLC A 指令的内容

助记符	操作	标志	操作码	字节数	周期数
RLC A	(PC)←(PC) + 1 (A_{n+1}) ←(A_n),n＝0～6 (A_0) ←(CY) (CY) ←(A_7)	CY	00110011	1	1

【5-18】 假设累加器 A 的数据为 C5H(11000101B)，进位标志(CY)＝0，则执行指令：

```
RLC A
```

结果：

累加器 A 的数据变成 8AH＝10001010B，进位标志(CY)＝1。

3) RR A

该指令将累加器 A 的内容循环右移，如表 5.47 所示。

表 5.47 RR A 指令的内容

助记符	操作	标志	操作码	字节数	周期数
RR A	(PC)←(PC) + 1 (A_n) ←(A_{n+1}),n＝0～6 (A_7) ←(A_0)	N	00000011	1	1

【例 5-19】 假设累加器 A 的数据为 C5H(11000101B)，则执行指令：

```
RR A
```

结果：

累加器 A 的内容变成 E2H＝11100010B。

4）RRC A

该指令将累加器 ACC 的内容和进位标志 CY 一起循环右移，如表 5.48 所示。

表 5.48　RRC A 指令的内容

助记符	操作	标志	操作码	字节数	周期数
RRC A	(PC)←(PC) ＋ 1 (A_n) ←(A_{n+1})，n＝0～6 (A_7) ←(CY) (CY) ←(A_0)	CY	00010011	1	1

【例 5-20】 假设累加器 A 的数据为 C5H(11000101B)，进位标志(CY)＝0，则执行指令：

```
RRC A
```

结果：

累加器 A 的内容变成 62H＝01100010B，进位标志(CY)＝1。

7. 半字节交换指令

SWAP A 指令将累加器 A 中的半字节互换，即将累加器 A 的高、低半字节互换，如表 5.49 所示。

表 5.49　SWAP A 指令的内容

助记符	操作	标志	操作码	字节数	周期数
SWAP A	(PC)←(PC) ＋ 1 ($A_{3\text{-}0}$) ←($A_{7\text{-}4}$) ($A_{7\text{-}4}$) ←($A_{3\text{-}0}$)	N	11000100	1	1

【例 5-21】 假设累加器 A 的数据为 C5H(11000101B)，则执行指令：

```
SWAP A
```

结果：

累加器 A 的内容变成 5CH＝01011100B。

思考与练习

5-8　如果(A)＝AAH，(R0)＝55H，则

(1) 执行指令

```
ANL A,R0
```

(A)＝_____。

(2) 执行指令

```
ORL A,R0
```

(A)=_____。

(3) 执行指令

```
XRL A,R0
```

(A)=_____。

(4) 执行指令

```
RL A
```

(A)=_____。

5.2.3 数据传送指令

STC 单片机中的数据传送指令包括数据传输指令、堆栈操作指令和数据交换指令。

1. 数据传输指令

STC 单片机中的数据传输指令包括内部数据传输指令、外部数据传输指令和查找表传输指令。

1) 内部数据传输指令

该类型数据传输指令是在任何两个内部 RAM 或者 SFR 间实现数据传输。这些指令使用直接、间接、寄存器和立即数寻址。

(1) MOV A,Rn

该指令将寄存器 Rn 中的内容复制到累加器 A 中,且 Rn 的内容不发生变化,如表 5.50 所示。

表 5.50 MOV A,Rn 指令的内容

助记符	操作	标志	操作码	字节数	周期数
MOV A, Rn	(PC)← (PC) + 1 (A)← (Rn)	N	11101rrr	1	1

注:rrr 为寄存器的编号,因此机器码范围是 E8H～EFH。

(2) MOV A,direct

该指令将直接寻址单元的内容复制到累加器 A 中,且直接寻址单元的内容不发生变化,如表 5.51 所示。

表 5.51 MOV A,direct 指令的内容

助记符	操作	标志	操作码	字节数	周期数
MOV A, direct	(PC)← (PC) + 2 (A)← (direct)	N	11100101	2	2

注:在操作码后面跟着一字节的直接地址。

(3) MOV A,@Ri

该指令将间接寻址单元中的内容复制到累加器 A 中,且间接寻址单元的内容不发生变化,如表 5.52 所示。

表 5.52　MOV A,@Ri 指令的内容

助记符	操作	标志	操作码	字节数	周期数
MOV A, @Ri	(PC)← (PC) + 1 (A)← ((Ri))	N	1110011i	1	2

注：i 表示 R0 或者 R1。当 i=0 时，表示 R0 寄存器；当 i=1 时，表示 R1 寄存器。

(4) MOV A,＃data

该指令将立即数复制到累加器 A 中，且立即数的内容不发生变化，如表 5.53 所示。

表 5.53　MOV A,＃data 指令的内容

助记符	操作	标志	操作码	字节数	周期数
MOV A, ＃data	(PC)← (PC) + 2 (A)← data	N	01110100	2	2

注：在操作码后面跟着一字节的立即数。

(5) MOV Rn, A

该指令将累加器 A 的内容复制到寄存器 Rn 中，且累加器 A 的内容不发生变化，如表 5.54 所示。

表 5.54　MOV Rn,A 指令的内容

助记符	操作	标志	操作码	字节数	周期数
MOV Rn, A	(PC)← (PC) + 1 (Rn)← (A)	N	11111rrr	1	1

注：rrr 为寄存器的编号，因此机器码范围是 F8H～FFH。

(6) MOV Rn, direct

该指令将直接寻址单元的内容复制到寄存器 Rn 中，且直接寻址单元的内容不发生变化，如表 5.55 所示。

表 5.55　MOV Rn,direct 指令的内容

助记符	操作	标志	操作码	字节数	周期数
MOV Rn, direct	(PC)← (PC) + 2 (Rn)← (direct)	N	10101rrr	2	3

注：rrr 为寄存器的编号，因此机器码范围是 A8H～AFH；在操作码后面跟着一字节的直接地址。

(7) MOV Rn, ＃data

该指令将立即数复制到寄存器 Rn 中，且立即数的内容不发生变化，如表 5.56 所示。

表 5.56 MOV Rn，#data 指令的内容

助记符	操作	标志	操作码	字节数	周期数
MOV Rn，#data	(PC)←(PC) + 2 (Rn)← data	N	01111rrr	2	2

注：rrr 为寄存器的编号，因此机器码范围是 78H～7FH。在操作码后面跟着一字节的立即数。

(8) MOV direct，A

该指令将累加器 A 的内容复制到直接寻址单元中，且累加器 A 的内容不发生变化，如表 5.57 所示。

表 5.57 MOV direct，A 指令的内容

助记符	操作	标志	操作码	字节数	周期数
MOV direct，A	(PC)←(PC) + 2 (direct)←(A)	N	11110101	2	2

注：在操作码后面跟着一字节的直接地址。

(9) MOV direct，Rn

该指令将寄存器 Rn 的内容复制到直接寻址单元中，且 Rn 的内容不发生变化，如表 5.58 所示。

表 5.58 MOV direct，Rn 指令的内容

助记符	操作	标志	操作码	字节数	周期数
MOV direct，Rn	(PC)←(PC) + 2 (direct)←(Rn)	N	10001rrr	2	2

注：rrr 为寄存器的编号，因此机器码范围是 88H～8FH。在操作码后面跟着一字节的直接地址。

(10) MOV direct，direct

该指令将直接寻址单元的内容复制到另一个直接寻址单元中，且源直接寻址单元的内容不发生变化，如表 5.59 所示。

表 5.59 MOV direct，direct 指令的内容

助记符	操作	标志	操作码	字节数	周期数
MOV direct，direct	(PC)←(PC) + 3 (direct)←(direct)	N	10000101	3	3

注：在操作码后面跟着两字节的直接地址，一个是源操作数地址，另一个是目的操作数地址。

(11) MOV direct，@Ri

该指令将间接寻址单元的内容复制到直接寻址单元中，且间接寻址单元的内容不发生变化，如表 5.60 所示。

表 5.60　MOV direct,@Ri 指令的内容

助记符	操作	标志	操作码	字节数	周期数
MOV direct, @Ri	(PC)←(PC) + 2 (direct)←((Ri))	N	1000011i	2	3

注：i 表示 R0 或者 R1。当 i=0 时，表示 R0 寄存器；当 i=1 时，表示 R1 寄存器。在操作码后面跟着一字节的直接地址。

(12) MOV direct, #data

该指令将立即数复制到直接寻址单元中，且立即数的内容不发生变化，如表 5.61 所示。

表 5.61　MOV direct,#data 指令的内容

助记符	操作	标志	操作码	字节数	周期数
MOV direct,#data	(PC) ← (PC) + 3 (direct) ← data	N	01110101	3	3

注：在操作码后面跟着一字节的直接地址和一字节的立即数。

(13) MOV @Ri, A

该指令将累加器 A 的内容复制到间接寻址的单元中，且累加器 A 的内容不发生变化，如表 5.62 所示。

表 5.62　MOV @Ri,A 指令的内容

助记符	操作	标志	操作码	字节数	周期数
MOV @Ri,A	(PC)←(PC) + 1 ((Ri))←(A)	N	1111011i	1	2

注：i 表示 R0 或者 R1。当 i=0 时，表示 R0 寄存器；当 i=1 时，表示 R1 寄存器。

(14) MOV @Ri, direct

该指令将直接寻址单元的内容复制到间接寻址的寄存器中，且直接寻址寄存器内容不发生变化，如表 5.63 所示。

表 5.63　MOV @Ri,direct 指令的内容

助记符	操作	标志	操作码	字节数	周期数
MOV @Ri, direct	(PC)←(PC) + 2 ((Ri))←(direct)	N	1010011i	2	3

注：i 表示 R0 或者 R1。当 i=0 时，表示 R0 寄存器；当 i=1 时，表示 R1 寄存器。在操作码后面跟着一字节的直接地址。

(15) MOV @Ri, #data

该指令将立即数内容复制到间接寻址单元中，且立即数的内容不发生变化，如表 5.64 所示。

表 5.64　MOV @Ri，#data 指令的内容

助记符	操作	标志	操作码	字节数	周期数
MOV @Ri，#data	(PC)← (PC) + 2 ((Ri))← data	N	0111011i	2	2

注：i 表示 R0 或者 R1。当 i=0 时，表示 R0 寄存器；当 i=1 时，表示 R1 寄存器。在操作码后面跟着一字节的立即数。

(16) MOV DPTR，#data 16

该指令将一个 16 位的立即数复制到数据指针寄存器 DPTR 中，且 16 位立即数的内容不发生变化，如表 5.65 所示。

表 5.65　MOV DPTR，#data16 指令的内容

助记符	操作	标志	操作码	字节数	周期数
MOV DPTR，#data 16	(PC)← (PC) + 3 DPH← $data_{15-8}$ DPL← $data_{7-0}$	N	10010000	3	3

注：在操作码后面跟着两字节(16 位)的立即数。

【例 5-22】 假设内部 RAM 地址为 30H 的单元的内容为 40H，而 40H 单元的内容为 10H。端口 1 的数据为 CAH(11001010B)，则执行指令：

```
MOV R0,#30H       ;将立即数 30H 送到寄存器 R0,(R0) = 30H
MOV A,@R0         ;将 30H 作为指向内部 RAM 的地址,内部 RAM 地址为 30H
                  ;单元的内容 40H 送到累加器 A 中
MOV R1,A          ;将累加器 A 的内容 40H 送到寄存器 R1 中,(R1) = 40H
MOV B,@R1         ;将 40H 作为指向内部 RAM 的地址,内部 RAM 地址为 40H
                  ;单元的内容 10H 送到寄存器 B 中
MOV @R1,P1        ;将 P1 端口的内容送到 R1 寄存器所指向的内部 RAM 的
                  ;地址单元中,即内部 RAM 地址为 40H 单元的内容变为 CAH
MOV P2,P1         ;将 P1 端口的内容送到 P2 端口中,P2 端口的内容变为 CAH
```

2) 外部数据传输指令

该类型传输指令是在累加器和 8051 片内扩展 RAM 和片外扩展 RAM 地址空间实现数据传输数据，这种传输只能使用 MOVX 指令。

注：对于 STC 8051 单片机来说，它的片内扩展 RAM 和外部扩展 RAM 都看作外部数据存储区。

(1) MOVX A，@Ri

该指令将外部数据存储区的一字节的内容复制到累加器 A 中。8 位外部数据存储区地址由 R0 或 R1 确定，且外部数据存储器单元的内容不发生变化，如表 5.66 所示。

表 5.66　MOVX A，@Ri 指令的内容

助记符	操作	标志	操作码	字节数	周期数
MOVX A，@Ri	(PC)← (PC) + 1 (A)←((Ri))	N	1110001i	1	3

注：i 表示 R0 或者 R1。当 i=0 时，表示 R0 寄存器；当 i=1 时，表示 R1 寄存器。

【例 5-23】 假设有一个时分复用地址/数据线的外部 RAM 存储器，容量为 256B，该存储器连接到 STC 单片机的 P0 端口上，端口 P3 用于提供外部 RAM 所需要的控制信号。端口 P1 和 P2 用作通用输入/输出端口。R0 寄存器和 R1 寄存器中的数据分别为 12H 和 34H，外部 RAM 地址为 34H 的单元内容为 56H，执行指令：

```
MOVX A,@R1      ;将外部 RAM 地址为 34H 单元的内容 56H 送到累加器 A
MOVX @R0,A      ;将累加器 A 的内容 56 送到外部 RAM 地址为 12H 的单元中
```

(2) MOVX A,@DPTR

该指令将外部数据存储区的一字节的内容复制到累加器 A 中。16 位外部数据存储区单元的地址由 DPTR 寄存器确定，且外部数据存储器单元的内容不发生变化，如表 5.67 所示。

表 5.67　MOVX A,@DPTR 指令的内容

助记符	操作	标志	操作码	字节数	周期数
MOVX A, @DPTR	(PC)←(PC) + 1 (A)←(DPTR)	N	11100000	1	2

(3) MOVX @Ri, A

该指令将累加器 A 的内容复制到外部数据存储单元中。8 位外部数据存储区地址由 R0 或 R1 确定，且累加器 A 中的内容不发生变化，如表 5.68 所示。

表 5.68　MOVX @Ri,A 指令的内容

助记符	操作	标志	操作码	字节数	周期数
MOVX @Ri, A	(PC)←(PC) + 1 ((Ri))←(A)	N	1111001i	1	4

注：i 表示 R0 或者 R1。当 i=0 时，表示 R0 寄存器；当 i=1 时，表示 R1 寄存器。

(4) MOVX @DPTR,A

该指令将累加器 A 的内容复制到外部数据存储单元中。16 位外部数据存储区单元的地址由 DPTR 寄存器确定，且累加器 A 中的内容不发生变化，如表 5.69 所示。

表 5.69　MOVX @DPTR,A 指令的内容

助记符	操作	标志	操作码	字节数	周期数
MOVX @DPTR, A	(PC)←(PC) + 1 (DPTR)←(A)	N	11110000	1	3

3) 查找表传输指令

只在累加器和程序存储器之间实现数据传输，这种传输只能使用 MOVC 指令。

(1) MOVC A,@A+DPTR

该指令将数据指针寄存器 DPTR 和累加器 A 的内容相加所得到的存储器地址单元的内容复制到累加器 A 中，如表 5.70 所示。

表 5.70 MOVC A,@A+DPTR 指令的内容

助记符	操作	标志	操作码	字节数	周期数
MOVC A,@A+DPTR	(PC)←(PC)+1 (A)←((A)+(DPTR))	N	10010011	1	5

【例 5-24】 假设累加器 A 的值为 0～4,下面的子程序将累加器 A 中的值转换为用 DB 伪指令定义的 4 个值之一。

```
REL_PC:  INC A
         MOVC A,@A+PC
         RET
         DB 66H
         DB 77H
         DB 88H
         DB 99H
```

如果在调用该子程序之前累加器的值为 02H,执行完该子程序后,累加器的值变为 88H。MOVC 指令之前的 INC A 指令是为了在查表时跨越 RET 而设置的。如果 MOVC 和表格之间被多字节隔开,则为了正确地读取表格,必须将相应的字节数预先加到累加器 A 上。

(2) MOVC A,@A+PC

该指令将程序计数器 PC 和累加器 A 的内容相加所得到的存储器地址单元的内容复制到累加器 A 中,如表 5.71 所示。

表 5.71 MOVC A,@A+PC 指令的内容

助记符	操作	标志	操作码	字节数	周期数
MOVC A,@A+PC	(PC)←(PC)+1 (A)←((A)+(PC))	N	10000011	1	4

2. 堆栈操作指令

1) POP direct

该指令将堆栈指针 SP 所指向栈顶的内容保存到直接寻址单元中,然后执行(SP)－1→(SP)的操作,此操作不影响标志位,如表 5.72 所示。

表 5.72 POP direct 指令的内容

助记符	操作	标志	操作码	字节数	周期数
POP direct	(PC)←(PC)+2 (direct)←((SP)) (SP)←(SP)－1	N	11010000	2	2

注：在操作码后面跟着一字节的直接地址。

【例 5-25】 假设堆栈指针的初值为 32H,内部 RAM 地址 30H～32H 单元的数据分别为 20H、23H 和 01H,则执行指令：

```
POP DPH
POP DPL
```

结果：

堆栈指针的值变成30H，(DPH)＝01H，(DPL)＝23H。

如果继续执行指令：

```
POP SP
```

则在这种特殊情况下，在写入出栈数据20H之前，栈指针减小到2FH，然后再随着20H的写入，(SP)＝20H。

2) PUSH direct

该指令将指针执行后堆栈指针(SP)＋1→SP指向栈顶单元，将直接寻址单元的内容送入SP所指向的堆栈空间，此操作不影响标志位，如表5.73所示。

表5.73　PUSH direct指令的内容

助记符	操作	标志	操作码	字节数	周期数
PUSH direct	(PC)←(PC)＋2 (SP)←(SP)＋1 ((SP))←(direct)	N	11000000	2	3

注：在操作码后面跟着一字节的直接地址。

【例5-26】 假设在进入中断服务程序之前堆栈指针的值为09H，数据指针DPTR的值为0123H，则执行下面的指令：

```
PUSH DPL
PUSH DPH
```

结果：

堆栈指针变成0BH，并把数据23H和01H分别保存到内部RAM的0AH和0BH的存储单元中。

3. 数据交换指令

1) XCH A，Rn

该指令将累加器A的内容和寄存器Rn中的内容互相交换，如表5.74所示。

表5.74　XCH A，Rn指令的内容

助记符	操作	标志	操作码	字节数	周期数
XCH A，Rn	(PC)←(PC)＋1 (A)↔(Rn)	N	11001rrr	1	2

注：rrr为寄存器的编号，因此机器码范围是C8H～CFH。

2) XCH A，direct

该指令将累加器A的内容和直接寻址单元的内容互相交换，如表5.75所示。

表 5.75　XCH A,direct 指令的内容

助记符	操作	标志	操作码	字节数	周期数
XCH A,direct	(PC)←(PC)＋2 (A)↔(direct)	N	11000101	2	3

注：在操作码后面跟着一字节的直接地址。

3) XCH A,@Ri

该指令将累加器 A 的内容和间接寻址的内容互相交换，如表 5.76 所示。

表 5.76　XCH A,@Ri 指令的内容

助记符	操作	标志	操作码	字节数	周期数
XCH A,@Ri	(PC)←(PC)＋1 (A)↔((Ri))	N	1100011i	1	3

注：i 表示 R0 或者 R1。当 i=0 时，表示 R0 寄存器；当 i=1 时，表示 R1 寄存器。

【例 5-27】 假设 R0 的内容为地址 20H，累加器 A 的内容为 3FH。内部 RAM 地址为 20H 单元的内容为 75H，执行指令：

```
XCH  A,@R0
```

结果：

执行该指令后，将 20H 所指向的内部 RAM 的单元的数据 75H 和累加器 A 的内容 3FH 进行交换，结果是累加器 A 的内容变成 75H，而内部 RAM 地址为 20H 单元的内容变成 3FH。

4) XCHD A,@Ri

该指令将累加器 A 的内容和间接寻址单元内容的低半字节互相交换，如表 5.77 所示。

表 5.77　XCHD A,@Ri 指令的内容

助记符	操作	标志	操作码	字节数	周期数
XCHD A,@Ri	(PC)←(PC)＋1 $(A_{3\text{-}0}) \leftrightarrow ((Ri)_{3\text{-}0})$	N	1101011i	1	3

注：i 表示 R0 或者 R1。当 i=0 时，表示 R0 寄存器；当 i=1 时，表示 R1 寄存器。

【例 5-28】 假设寄存器 R0 的内容为 20H，累加器 A 的内容为 36H。内部 RAM 地址为 20H 的单元内容为 75H，执行指令：

```
XCHD A,@R0
```

结果：

将 20H 所指向内部 RAM 单元的数据 75H 和累加器 A 内容为 36H 的低四位数据进行交换，结果是累加器 A 的内容变成 35H，而内部 RAM 地址为 20H 单元的内容变成 76H。

思考与练习

5-9　假设(30H)=40H，(31H)=5DH，(SP)=15H，则执行下面的指令：

```
PUSH 30H
PUSH 31H
```

其目的是_____,(SP)=_____。

5.2.4　布尔指令

8051单片机有独立的位可寻址存储区域。它有128比特的位可寻址RAM和SFR。

1. 清除指令

1) CLR bit

该指令将目的比特位清0,如表5.78所示。

表5.78　CLR bit指令的内容

助记符	操作	标志	操作码	字节数	周期数
CLR bit	(PC)←(PC)+2 (bit)←0	N	11000010	2	3

注：在操作码后面跟着一字节的位地址。

【例5-29】 假设端口P1的数据为5DH(01011101B),执行指令：

```
CLR  P1.2
```

结果：

端口P1的数据为59H(01011001B)。

2) CLR C

该指令将进位标志位CY清0,如表5.79所示。

表5.79　CLR C指令的内容

助记符	操作	标志	操作码	字节数	周期数
CLR C	(PC)←(PC)+1 (C)←0	CY	11000011	1	1

2. 设置指令

1) SETB bit

该指令将目标比特位置1,如表5.80所示。

表5.80　SETB bit指令的内容

助记符	操作	标志	操作码	字节数	周期数
SETB bit	(PC)←(PC)+2 (bit)←1	N	11010010	2	3

注：在操作码后面跟着一字节的位地址。

2) SETB C

该指令将进位标志CY置1,如表5.81所示。

表 5.81 SETB C 指令的内容

助记符	操作	标志	操作码	字节数	周期数
SETB C	(PC)← (PC) + 1 (C)← 1	CY	11010011	1	1

【例 5-30】 假设端口 P1 的数据为 34H(00110100B),执行指令:

```
SETB C
SETB P1.0
```

结果:

进位标志(CY)=1,端口 P1 的数据变成为 35H(00110101B)。

3. 取反指令

1) CPL bit

该指令将目标比特位取反,如表 5.82 所示。

表 5.82 CPL bit 指令的内容

助记符	操作	标志	操作码	字节数	周期数
CPL bit	(PC)← (PC) + 2 (bit)← ($\overline{bit}$)	N	10110010	2	3

注:在操作码后面跟着一字节的位地址。

【例 5-31】 假设端口 P1 的数据为 5BH(01011011B),执行指令:

```
CPL P1.1
CPL P1.2
```

结果:

端口 P1 的数据变成为 5DH(01011101B)。

2) CPL C

该指令将进位标志 CY 取反。如果 CY 为 1,执行该指令后 CY 为 0;反之亦然,如表 5.83 所示。

表 5.83 CPL C 指令的内容

助记符	操作	标志	操作码	字节数	周期数
CPL C	(PC)← (PC) + 1 (C)← ($\overline{C}$)	CY	10110011	1	1

4. 逻辑与指令

1) ANL C, bit

该指令对进位标志 CY 和一比特位做逻辑与操作,结果保存在 CY 中,如表 5.84 所示。

表 5.84　ANL C,bit 指令的内容

助记符	操作	标志	操作码	字节数	周期数
ANL C, bit	(PC)← (PC) + 2 (CY) ← (CY) ∧ (bit)	CY	10000010	2	2

注：在操作码后面跟着一字节的位地址。

2) ANL C, /bit

该指令对进位标志 CY 和一个比特位取反后做逻辑与操作，结果保存在 CY 中，如表 5.85 所示。

表 5.85　ANL C,/bit 指令的内容

助记符	操作	标志	操作码	字节数	周期数
ANL C, /bit	(PC)← (PC) + 2 (CY) ← (CY) ∧ $\overline{(bit)}$	CY	10110000	2	2

注：在操作码后面跟着一字节的位地址。

【例 5-32】 假设 P1 端口的第 0 位为 1，且累加器 A 的第 7 位为 1，同时溢出标志 OV 的内容为 0，执行指令：

```
MOV C,P1.0          ;进位标志 CY 设置为 1
ANL C,ACC.7         ;进位标志 CY 设置为 1
ANL C,/OV           ;进位标志 CY 设置为 1
```

5. 逻辑或指令

1) ORL C, bit

该指令把进位标志 CY 的内容和比特位内容做逻辑或操作，结果保存在 CY 中，如表 5.86 所示。

表 5.86　ORL C,bit 指令的内容

助记符	操作	标志	操作码	字节数	周期数
ORL C,bit	(PC)← (PC) + 2 (CY) ← (CY) ∨ (bit)	CY	01110010	2	2

注：在操作码后面跟着一字节的位地址。

2) ORL C, /bit

该指令把进位标志 CY 的内容和比特位内容取反后做逻辑或操作，结果保存在 CY 中，如表 5.87 所示。

表 5.87　ORL C,/bit 指令的内容

助记符	操作	标志	操作码	字节数	周期数
ORL C,/bit	(PC)← (PC) + 2 (CY) ← (CY) ∨ $\overline{(bit)}$	CY	10100000	2	2

注：在操作码后面跟着一字节的位地址。

【例 5-33】 假设 P1 端口的第 0 位为 1，或者累加器 A 的第 7 位为 1，或者溢出标志 OV 的内容为 0，执行指令：

```
MOV C,P1.0          ;进位标志 CY 设置为 1
ORL C,ACC.7         ;进位标志 CY 设置为 1
ORL C,/OV           ;进位标志 CY 设置为 1
```

6. 传输指令

1) MOV C，bit

该指令把一比特位的值复制到进位标志 CY 中，且比特位的值不发生变化，如表 5.88 所示。

表 5.88 MOV C,bit 指令的内容

助记符	操作	标志	操作码	字节数	周期数
MOV C,bit	(PC)← (PC) + 2 (CY) ← (bit)	CY	10100010	2	2

注：在操作码后面跟着一字节的位地址。

2) MOV bit,C

该指令把进位标志 CY 的内容复制到一个比特位中，且进位标志 CY 的值不发生变化，如表 5.89 所示。

表 5.89 MOV bit,C 指令的内容

助记符	操作	标志	操作码	字节数	周期数
MOV bit,C	(PC)← (PC) + 2 (bit)← (C)	N	10010010	2	3

注：在操作码后面跟着一字节的位地址。

【例 5-34】 假设进位标志 CY 的初值为 1，端口 P2 中的数据为 C5H(11000101B)，端口 P1 中的数据为 35H(00110101B)，执行指令：

```
MOV P1.3,C          ;P1 端口的值变为 3DH(00111101B)
MOV C,P2.3          ;进位标志 CY 设置为 0
MOV P1.2,C          ;P1 端口的值变为 39H(00111001B)
```

7. 跳转指令

1) JB bit,rel

该指令判断 bit 位中的数据是否为 1，如果为 1，则跳转到(PC) + rel 指定的目标地址；否则，程序转向下一条指令，该操作不影响标志位，如表 5.90 所示。

表 5.90 JB bit,rel 指令的内容

助记符	操作	标志	操作码	字节数	周期数
JB bit,rel	(PC)← (PC) + 3 如果(bit) = 1，则(PC) ← (PC) + rel	N	00100000	3	5

注：在使用助记符编写汇编语言程序时，Keil μVision 将助记符中的 rel 转换成程序存

储空间内的一个目标地址。而在生成所对应的机器指令时，并不是直接使用 rel 所表示的目标地址，而是将其转换成一个相对偏移量 rel.address，对于该条指令而言，相对偏移量的计算方法表示为

rel.address＝rel(助记符所表示的目标地址)－PC－3

其中，rel.address 对应于操作中(PC)＋rel 中的 rel。

因此，读者一定要正确理解助记符中 rel 的含义，以及操作中 rel 所表示的含义。对于本条机器指令而言，操作码后面跟着一字节的位地址和一字节的偏移量 rel.address，即表示为下面的形式：

0 0 1 0	0 0 0 0	bit address	rel.address

【例 5-35】 假设端口 1 的数据为 CAH(11001010B)，累加器 A 的内容为 56H(01010110B)。则执行指令：

```
JB P1.2,LABEL1     ;跳转条件不成立
JB ACC.2,LABEL2    ;跳转条件成立
```

结果：

程序跳转到标号 LABEL2 的地方执行。

2) JNB bit，rel

该指令判断 bit 中的数据是否为 0，如果为 0，则程序跳转到(PC) ＋ rel 指定的目标地址去；否则，程序转向下一条指令，该操作不影响标志位，如表 5.91 所示。

表 5.91　JNB bit，rel 指令的内容

助记符	操作	标志	操作码	字节数	周期数
JNB bit，rel	(PC)← (PC) ＋ 3 如果(bit) ＝ 0，则(PC) ← (PC) ＋ rel	N	00110000	3	5

注：在操作码后面跟着一字节的位地址和一字节的偏移量 rel。

【例 5-36】 假设端口 1 的数据为 CAH(11001010B)，累加器 A 的内容为 56H(01010110B)。则执行指令：

```
JNB P1.3,LABEL1    ;跳转条件不成立
JNB ACC.3,LABEL2   ;跳转条件成立
```

结果：

程序跳转到标号 LABEL2 的地方执行。

3) JC rel

该指令判断进位标志位 CY 是否为 1，如果为 1，则跳转到(PC)＋rel 指定的目标地址；否则，程序转向下一条指令，该操作不影响标志位，如表 5.92 所示。

表 5.92　JC rel 指令的内容

助记符	操作	标志	操作码	字节数	周期数
JC rel	(PC)← (PC) ＋ 2 如果(CY) ＝ 1，则(PC) ← (PC) ＋ rel	N	01000000	2	3

注：在操作码后面跟着一字节的偏移量 rel。

【例 5-37】 假设进位标志为 CY 为 0，则执行指令：

```
JC LABEL1      ;跳转条件不成立
CPL C          ;取反，进位标志 CY 变为 1
JC LABEL2      ;跳转条件成立
```

结果：

程序跳转到标号 LABEL2 的地方执行。

4) JNC rel

该指令判断进位标志位 CY 是否为 0，如果为 0，则跳转到(PC)＋rel 指定的目标地址；否则，程序转向下一条指令，该操作不影响标志位，如表 5.93 所示。

表 5.93 JNC rel 指令的内容

助记符	操作	标志	操作码	字节数	周期数
JNC rel	(PC)← (PC) ＋ 2 如果(CY) ＝ 0，则(PC) ← (PC) ＋ rel	N	01010000	2	3

注：在操作码后面跟着一字节的偏移量 rel。

【例 5-38】 假设进位标志为 CY 为 1，则执行指令：

```
JNC LABEL1     ;跳转条件不成立
CPL C          ;取反，进位标志 CY 变为 0
JNC LABEL2     ;跳转条件成立
```

结果：

程序跳转到标号 LABEL2 的地方执行。

5) JBC bit，rel

该指令判断指定 bit 位是否为 1，如果为 1，则将该位清零，并且跳转到(PC) ＋ rel 指定的目标地址；否则，程序转向下一条指令，该操作不影响标志位，如表 5.94 所示。

表 5.94 JBC bit，rel 指令的内容

助记符	操作	标志	操作码	字节数	周期数
JBC bit，rel	(PC)← (PC) ＋ 3 如果(bit) ＝ 1，则 bit← 0，(PC) ← (PC) ＋ rel	N	00010000	3	5

注：在操作码后面跟着一字节的位地址和一字节的偏移量 rel。

【例 5-39】 假设累加器 A 的内容为 56H(01010110B)，则执行指令：

```
JBC ACC.3,LABEL1     ;跳转条件不成立
JBC ACC.2  LABEL2    ;跳转条件成立，并且将 ACC.2 清零
```

结果：

程序跳转到标号 LABEL2 的地方执行，累加器 A 的内容变为 52H(01010010B)。

5.2.5　程序分支指令

8051 CPU 支持有条件和无条件的跳转指令，这些跳转指令用于改变程序的执行顺序。

1. 调用指令

1) ACALL addr11

该指令无条件地调用在指定地址处的子程序。目标地址由递增 PC 的高 5 位、操作码的第 7 位～第 5 位和指令第 2 字节并置组成。所以所调用的子程序的首地址必须与 ACALL 后面指令的第一字节在同一个 2KB 区域内，如表 5.95 所示。

表 5.95　ACALL addr11 指令的内容

助记符	操作	标志	操作码	字节数	周期数
ACALL addr11	(PC) ← (PC) + 2 (SP) ← (SP) + 1 ((SP)) ← ($PC_{7\text{-}0}$) (SP) ← (SP) + 1 ((SP)) ← ($PC_{15\text{-}8}$) ($PC_{10\text{-}0}$) ←页面地址	无	$a_{10}a_9a_8$10010	2	4

注：$a_{10}a_9a_8$是 11 位目标地址的 A10～A8 位。在操作码后面带着一字节目标地址的 A7～A0 位。

【例 5-40】 假设堆栈指针的初值为 07H，标号 SUBRTN 位于程序存储器地址为 0345H 的位置，如果执行位于地址 0123H 处的指令：

```
ACALL SUBRTN
```

结果：

堆栈指针的内容变成 09H，内部 RAM 地址为 08H 和 09H 的位置所保存的内容为 25H 和 01H，PC 值变为 0345H。

2) LCALL addr16

该指令无条件地调用首地址为 addr16 处的子程序。执行该指令时，将 PC 加 3，以获得下一条指令的地址。然后，将指令第 2 字节和第 3 字节所提供的 16 位目标地址加载到 $PC_{15\text{-}0}$，程序转向子程序的首地址执行，如表 5.96 所示。所调用的子程序的首地址可以在 64KB 的范围内。

表 5.96　LCALL addr16 指令的内容

助记符	操作	标志	操作码	字节数	周期数
LCALL addr16	(PC)← (PC) + 3 (SP)← (SP) + 1 ((SP))← ($PC_{7\text{-}0}$) (SP)← (SP) + 1 ((SP))← ($PC_{15\text{-}8}$) (PC)← $addr_{15\text{-}0}$	N	00010010	3	4

注：在操作码后面带着一字节目标地址的 A_{15}～A_8 位和一字节目标地址的 A_7～A_0 位。

【例 5-41】 假设堆栈指针的初值为 07H，标号 SUBRTN 位于程序存储器地址为 1234H 的位置，如果执行位于地址 0123H 处的指令：

```
LCALL SUBRTN
```

结果：

堆栈指针的内容变成 09H，内部 RAM 地址为 08H 和 09H 的位置所保存的内容为 26H 和 01H，PC 值变为 1234H。

2. 返回指令

1) RET

该指令将栈顶高地址和低地址字节连续地送给 PC 的高字节和低字节，并把堆栈指针减 2，返回 ACALL 或 LCALL 的下一条指令，继续往下执行，该指令的操作不影响标志位，如表 5.97 所示。

表 5.97 RET 指令的内容

助记符	操作	标志	操作码	字节数	周期数
RET	$(PC_{15\text{-}8}) \leftarrow ((SP))$ $(SP) \leftarrow (SP) - 1$ $(PC_{7\text{-}0}) \leftarrow ((SP))$ $(SP) \leftarrow (SP) - 1$	N	00100010	1	4

【例 5-42】 堆栈指针的内容为 0BH，内部 RAM 地址为 0AH 和 0BH 的位置保存的内容为 23H 和 01H，如果执行指令：

```
RET
```

结果：

堆栈指针的内容变成 09H，程序将从地址为 0123H 的地方继续执行。

2) RETI

该指令将从中断服务程序返回，并清除相应的内部中断状态寄存器。CPU 在执行 RETI 后，至少要再执行一条指令，才能响应新的中断请求，如表 5.98 所示。

表 5.98 RETI 指令的内容

助记符	操作	标志	操作码	字节数	周期数
RETI	$(PC_{15\text{-}8}) \leftarrow ((SP))$ $(SP) \leftarrow (SP) - 1$ $(PC_{7\text{-}0}) \leftarrow ((SP))$ $(SP) \leftarrow (SP) - 1$	N	00110010	1	4

【例 5-43】 堆栈指针的内容为 0BH，结束在地址 0123H 处的指令执行结束期间产生中断，内部 RAM 地址为 0AH 和 0BH 的位置保存的内容为 23H 和 01H，如果执行指令：

```
RETI
```

结果：

堆栈指针的内容变成09H，中断返回后继续从程序代码地址为0123H的位置继续执行。

3. 无条件转移指令

1) AJMP addr11

该指令实现无条件跳转。绝对跳转操作的目标地址是由PC递增两次后的高5位，操作码的第7～5位和第二字节并置而成，如表5.99所示。目标地址必须包含AJMP指令后第一条指令的第一字节在内的2KB范围内。

表5.99　AJMP addr11指令的内容

助记符	操作	标志	操作码	字节数	周期数
AJMP addr11	(PC)←(PC)＋2 ($PC_{10\text{-}0}$)←页面地址	N	$a_{10}a_9a_8$00001	2	3

注：$a_{10}a_9a_8$是11位目标地址的A_{10}～A_8位。在操作码后面带着一字节的目标地址的A_7～A_0位。

【例5-44】 假设标号JMPADR位于程序存储器的0123H的位置，如果指令：

```
AJMP JMPADR
```

位于程序存储器地址为0345H的位置。

结果：

执行完该指令后，PC的值变为0123H。

2) LJMP addr16

该指令实现无条件长跳转操作，跳转的16位目的地址由指令的第2字节和第3字节共同组成，如表5.100所示。因此，程序指向的目标地址可以包含程序存储器的整个64KB空间。

表5.100　LJMP addr16指令的内容

助记符	操作	标志	操作码	字节数	周期数
LJMP addr16	(PC)← addr15…addr0	N	00000010	3	4

注：在操作码后面带着一字节的目标地址的A_{15}～A_8位和一字节目标地址的A_7～A_0位。

【例5-45】 假设标号JMPADR位于程序存储器的1234H的位置，如果指令：

```
LJMP JMPADR
```

位于程序存储器地址为1234H的位置。

结果：

执行完该指令后，PC的值变为1234H。

3) SJMP rel

该指令实现无条件短跳转操作，跳转的目的地址是由PC递增两次后的值和指令的第2字节带符号的相对地址相加而成的，如表5.101所示。

表 5.101　SJMP rel 指令的内容

助记符	操作	标志	操作码	字节数	周期数
SJMP rel	(PC)←(PC) + 2 (PC)←(PC) + rel	N	10000000	2	3

注：在操作码后面带着一字节的偏移量 rel。

【例 5-46】 假设标号 RELADR 位于程序存储器的 0123H 的位置，如果指令：

```
SJMP RELADR
```

位于程序存储器地址为 0100H 的位置。

结果：

执行完该指令后，PC 的值变为 0123H。

注：在上面这个例子中，紧接 SJMP 的下一条指令的地址是 0102H，因此跳转的偏移量为 0123H－0102H＝21H

4) JMP @A+DPTR

该指令实现无条件的跳转操作，跳转的目标地址是将累加器 A 中的 8 位无符号数与数据指针 DPTR 的内容相加得到。相加运算不影响累加器 A 和数据指针 DPTR 的原内容，如表 5.102 所示。若相加结果大于 64KB，则从程序存储器的零地址往下延续。

表 5.102　JMP @A+DPTR 指令的内容

助记符	操作	标志	操作码	字节数	周期数
JMP @A+DPTR	(PC)←(A) + (DPTR)	N	01110011	1	5

【例 5-47】 假设累加器 A 中的值是偶数(0～6)。下面的指令序列将使程序跳转到位于跳转表 JMP_TBL 的 4 条 AJMP 指令中的某一条去执行：

```
        MOV DPTR, #JMP_TBL
        JMP @A + DPTR
JMP_TBL: AJMP LABEL0
        AJMP LABEL1
        AJMP LABEL2
        AJMP LABEL3
```

如果开始执行上面指令时，累加器 A 中的值为 04H，那么程序最终会跳到标号为 LABEL2 的地方执行。

注：AJMP 是一个 2 字节指令，所以在跳转表中，各个跳转指令的入口地址依次相差 2 字节。

4. 有条件转移指令

1) JNZ rel

该指令实现有条件跳转。判断累加器 A 的内容是否不为 0，如果不为 0，则跳转到(PC) + rel 指定的目标地址；否则，程序转向下一条指令，如表 5.103 所示。

表 5.103　JNZ rel 指令的内容

助记符	操作	标志	操作码	字节数	周期数
JNZ rel	(PC)← (PC) + 2 如果(A) ≠ 0,则(PC) ← (PC) + rel	N	01110000	2	4

注：在操作码后面带着一字节的偏移量 rel。

【例 5-48】 假设累加器 A 的内容 00H,则执行指令：

```
JNZ LABEL1      ;跳转条件不成立
INC A           ;累加器的内容加 1
JNZ LABEL2      ;跳转条件成立
```

结果：

程序跳转到标号 LABEL2 的地方执行。

2) JZ rel

该指令实现有条件跳转。判断累加器 A 的内容是否为 0,如果为 0,则跳转到(PC) + rel 指定的目标地址；否则,程序转向下一条指令,如表 5.104 所示。

表 5.104　JZ rel 指令的内容

助记符	操作	标志	操作码	字节数	周期数
JZ rel	(PC)← (PC) + 2 如果(A) = 0,则(PC) ← (PC) + rel	N	01100000	2	4

注：在操作码后面带着一字节的偏移量 rel。

【例 5-49】 假设累加器 A 的内容 01H,则执行指令：

```
JZ LABEL1       ;跳转条件不成立
DEC A           ;累加器的内容减 1
JZ LABEL2       ;跳转条件成立
```

结果：

程序跳转到标号 LABEL2 的地方执行。

3) CJNE A, direct, rel

该指令对累加器 A 和直接寻址单元内容相比较,若它们的值不相等,则程序转移到(PC) + rel 指向的目标地址。若直接寻址单元的内容小于累加器内容,则清除进位标志 CY；否则,置位进位标志 CY,如表 5.105 所示。

表 5.105　CJNE A, direct, rel 指令的内容

助记符	操作	标志	操作码	字节数	周期数
CJNE A, direct,rel	(PC)← (PC) + 3 如果(A) ≠ (direct),则 (PC)← (PC) + rel 如果(A) < (direct),则(CY) ← 1 否则(CY) ← 0	CY	10110101	3	5

注：在操作码后面跟着一字节的直接地址和一字节的偏移量 rel。

4) CJNE A，＃data，rel

该指令将比较累加器 A 的内容和立即数，若它们的值不相等，则程序转移(PC) ＋ rel 指向的目标地址。进位标志 CY 设置同上，该指令不影响累加器 A 的内容，如表 5.106 所示。

表 5.106　CJNE A，＃data，rel 指令的内容

助记符	操作	标志	操作码	字节数	周期数
CJNE A，＃data，rel	(PC)← (PC) ＋ 3 如果(A) ≠ data，则 (PC)← (PC) ＋ rel 如果(A) ＜ data，则 (CY) ← 1 否则(CY) ← 0	CY	10110101	3	4

注：在操作码后面跟着一字节的立即数和一字节的偏移量 rel。

5) CJNE Rn，＃data，rel

该指令将寄存器 Rn 的内容和立即数进行比较，若它们的值不相等，则程序转移到(PC) ＋ rel 指向的目标地址，如表 5.107 所示。

表 5.107　CJNE Rn，＃data，rel 指令的内容

助记符	操作	标志	操作码	字节数	周期数
CJNE Rn，＃data，rel	(PC)← (PC) ＋ 3 如果(Rn) ≠ data，则 (PC)← (PC) ＋ rel 如果(Rn) ＜data，则(CY) ← 1 否则(CY) ← 0	CY	10111rrr	3	4

注：rrr 为寄存器的编号，因此机器码范围是 B8H～BFH。在操作码后面跟着一字节的立即数和一字节的偏移量 rel。

6) CJNE @Ri，＃data，rel

该指令将间接寻址的内容和立即数相比较，若它们的值不相等，则程序转移到(PC) ＋ rel 指向的目标地址，如表 5.108 所示。

表 5.108　CJNE @Ri，＃data，rel 指令的内容

助记符	操作	标志	操作码	字节数	周期数
CJNE @Ri，＃data，rel	(PC)← (PC) ＋ 3 如果((Ri))≠data，则 (PC)← (PC) ＋ rel 如果((Ri)) ＜data，则 (CY) ← 1 否则(CY) ← 0	CY	1011011i	3	5

注：i 表示 R0 或者 R1。当 i＝0 时，表示 R0 寄存器；当 i＝1 时，表示 R1 寄存器。在操

作码后面跟着一字节的立即数和一字节的偏移量 rel。

【例 5-50】 假设累加器 A 的内容 34H，寄存器 R7 的内容为 56H。则执行指令：

```
CJNE R7,#60H,NOT_EQ
…                              ;R7 的内容为 60H
NOT_EQ:  JC     REQ_LOW        ;如果 R7<60H
…                              ;R7>60H
```

结果：

第一条指令将进位标志 CY 设置为 1，程序跳转到标号 NOT_EQ 的地方。接下来测试进位标志 CY，可以确定寄存器 R7 的内容大于还是小于 60H。

7) DJNZ Rn,rel

该指令实现有条件跳转。每执行一次指令，寄存器 Rn 的内容减 1，并判断其内容是否为 0。若不为 0 则转向(PC) + rel 指向的目标地址，继续执行循环程序，否则，结束循环程序，执行下一条指令，如表 5.109 所示。

表 5.109　DJNZ Rn,rel 指令的内容

助记符	操作	标志	操作码	字节数	周期数
DJNZ Rn, rel	(PC)←(PC) + 2 (Rn)←(Rn) − 1 如果(Rn)≠0,则 (PC)←(PC) + rel	N	11011rrr	2	4

注：rrr 为寄存器的编号，因此机器码范围是 D8H～DFH。在操作码后面跟着一字节的偏移量 rel。

8) DJNZ direct,rel

该指令实现有条件跳转。每执行一次指令，直接寻址单元的内容减 1，并判断其内容是否为 0。若不为 0，则转向(PC) + rel 指向的目标地址，继续执行循环程序，否则，结束循环程序，执行下一条指令，如表 5.110 所示。

表 5.110　DJNZ direct,rel 指令的内容

助记符	操作	标志	操作码	字节数	周期数
DJNZ direct, rel	(PC)←(PC) + 3 (direct)←(direct) − 1 如果(direct) ≠ 0,则 (PC)←(PC) + rel	N	11010101	3	5

注：在操作码后面跟着一字节的目标位地址和一字节的偏移量 rel。

【例 5-51】 假设内部 RAM 地址为 40H、50H 和 60H 的单元分别保存着数据 01H、70H 和 15H，则执行指令：

```
DJNZ 40H,LABEL_1
DJNZ 50H,LABEL_2
DJNZ 60H,LABEL_3
```

结果：

程序将跳转到标号 LABEL_2 处执行，且相应的 3 个 RAM 单元的内容变成 00H、6FH 和 15H。

5. 空操作指令

NOP

该指令表示无操作，PC＋1，如表 5.111 所示。

表 5.111　NOP 指令的内容

助记符	操作	标志	操作码	字节数	周期数
NOP	(PC)←(PC) ＋ 1	N	0x00	1	1

【例 5-52】 假设期望在端口 P2 的第 7 位引脚上输出一个长时间的低电平脉冲，该脉冲持续 5 个机器周期(精确)。若仅仅使用 SETB 和 CLR 指令序列，生成的脉冲只能持续一个机器周期。因此，需要设法增加 4 个额外的机器周期，可以按照下面的方式实现所要求的功能(假设在此期间没有使能中断)：

```
CLR P2.7
NOP
NOP
NOP
NOP
SETB P2.7
```