CHAPTER 3 第3章

80C51单片机的指令系统

计算机系统包括硬件系统和软件系统两部分。一台计算机要正常工作，除硬件系统外，还必须有相应的软件与之配合，才能充分发挥其功能。而软件的基础则是计算机指令系统。80C51单片机指令系统共有111条指令。本章将详细介绍80C51系列单片机指令系统的寻址方式，以及各类指令的功能和格式。

3.1 指令系统简介

3.1.1 指令的格式

指令的表示方法称为指令格式。51系列单片机指令主要由操作码助记符字段和操作数字段组成。格式如下：

[标号：] 操作码 [,操作数或操作数地址] [;注释]

[]中的内容是可选的，其包含的内容因指令的不同可有可无。

标号是用户根据编程需要为指令设定的符号地址，当需要时加上。标号由1～8个字符组成，第一个字符必须是英文字母，不能是数字或其他符号，标号后面必须用冒号。

操作码表示指令的操作功能，是指令必不可少的部分。例如，MOV表示数据传送操作，MUL表示乘法运算等。

操作数或操作数地址表示参加运算的数据或数据的有效地址。操作数一般有以下几种形式：有的指令没有操作数项，操作数隐含在操作码中，如RETI指令；有的指令只有一个操作数，如“CLR C”指令；有的指令有两个操作数，如“MOV A，#20H”指令，操作数之间以逗号相隔，前者为目的操作数，后者为源操作数；有的指令有3个操作数，如“CJNE A，20H，LOOP”指令，多个操作数之间也用逗号分开。

注释是对指令的注解，是为了便于人们阅读程序而设定的，用来提高程序的可读性。计算机的汇编程序在汇编时对注释不作处理。汇编语言要求在注释前必须加分号。

3.1.2 指令的三种表示形式

指令的3种表示形式为：

二进制形式——直接为CPU执行；

十六进制形式——阅读和书写；

汇编形式——编写程序。

具体形式见表 3-1。

表 3-1 指令的三种表示形式

二进制形式	十六进制形式	汇编形式
01110100 00001010	740AH	MOV A,#0AH
00100100 00000100	2404H	ADD A,#04H
10000000 11111110	80FEH	SJMP $

3.1.3 指令的字节数

80C51 单片机的机器语言根据指令的长短又将指令分为一字节指令、二字节指令、三字节指令 3 种格式。一字节指令即在程序存储器中需要一个字节的单元来存储；二字节指令即在程序存储器中需要两个字节的单元来存储；三字节指令即在程序存储器中需要三个字节的单元来存储。

1. 单字节指令

单字节指令格式由 8 位二进制编码表示。在 80C51 指令系统中，单字节指令可分为以下两类。

(1) 无操作数单字节指令。这类指令只有操作码字段，操作数隐含在操作码中。例如，空操作“NOP”的指令码为

0	0	0	0	0	0	0	0

(2) 含有操作数寄存器编号的单字节指令。指令码由操作码字段和指示操作数所在寄存器编号的字段组成。例如，“MOV A,R*n*”的指令码为

1	1	1	0	1	*r*	*r*	*r*

其中，*rrr* 为寄存器 R*n* 的编号，假设 *n*=0，则 *rrr*=000。

2. 双字节指令

双字节指令格式中，指令的编码由两个字节组成，该指令存放在存储器时需占用两个存储器单元。例如，“MOV A,#23H”的指令码为

第一字节	0 1 1 1 0 1 0 0	操作码
第二字节	0 0 1 0 0 0 1 1	操作数

这条指令的功能是将立即数 23H 送到累加器 A 中去。

3. 三字节指令

三字节指令格式中第一个字节为操作码，第二和第三个字节为操作数或操作数地址。

例如，“MOV 74H，#0FFH”的指令码为

第一字节	1 0 1 1 0 1 0 0	操作码
第二字节	0 1 1 1 0 1 0 0	第一操作数(目的操作数)
第三字节	1 1 1 1 1 1 1 1	第二操作数(立即数)

这条指令是指立即数0FFH送到地址为74H的单元中去。

3.1.4 指令的分类

80C51单片机有111条指令，指令的分类方法不同，所分类型也不同。

1. 按指令的功能划分

数据传送指令：把数据、地址传送到寄存器或存储单元中。

算术运算指令：完成加、减、乘、除等各种算术运算。

逻辑运算指令：实现与、或、非、异或等逻辑运算。

控制转移指令：包括无条件转移、条件转移等指令，可用于分支和循环程序中。

位操作指令：对位寻址区地址单元进行操作，实现位控制。

2. 按指令的字节数划分

单字节指令：只有一个字节，即指令的操作码。

双字节指令：有两个字节，第二个字节为地址码或操作数。

三字节指令：首字节为操作码，后两个字节为地址码或操作数。

3. 按指令的机器周期划分

可分为单机器周期指令、双机器周期指令、4个机器周期指令。

3.2 寻址方式

在带有操作数的指令中，数据可能就在指令中，也有可能在寄存器或存储器中，甚至在I/O接口中。对这些设备内的数据要正确进行操作就要在指令中指出其地址，寻找操作数地址的方法称为寻址方式。寻址方式的多少及寻址功能强弱是反映指令系统性能优劣的重要指标。

3.2.1 符号约定

任何一种计算机语言都有自己的语法规则。80C51单片机采用汇编语言，其常用符号含义见表3-2。

3.2.2 寻址方式说明

80C51单片机的寻址方式有七种，即立即寻址、寄存器寻址、直接寻址、寄存器间接寻址、变址寻址、相对寻址和位寻址。这些寻址方式所对应的寄存器和存储空间见表3-3。

表 3-2 80C51 单片机汇编语言常用符号的含义

符 号	含 义	符 号	含 义
R*n*	内部工作寄存器 R0～R7 之一	B	B 寄存器
R*i*	工作寄存器 R0 或 R1	bit	位地址
＃data	8 位立即数	DPTR	数据指针
Direct	片内 RAM 直接地址	＃	立即数前缀
Addr16	16 位目的地址	/	位操作的取反操作前缀
Addr11	11 位目的地址	(×)	表示×地址单元或寄存器内容
rel	补码形式表示的 8 位偏移量	((×))	表示×地址单元或寄存器内容为地址所指定单元的内容
@	间址或基址寄存器前缀	←	数据传送方向
A	累加器	←→	数据交换

表 3-3 寻址方式所对应的寄存器和存储空间

寻 址 方 式	寻 址 范 围
立即寻址	ROM
寄存器寻址	R0～R7,A,AB,DPTR,C(CY)
直接寻址	内部 RAM 低 128B,SFR,位寻址区
寄存器间接寻址	片内 RAM(@R0,@R1,SP——仅 PUSH、POP) 片外 RAM 或 I/O 接口(@R0,@R1,@DPTR)
变址寻址	ROM(@A＋DPTR,@A＋PC)
相对寻址	ROM(－128B ～ ＋127B)
位寻址	可寻址位(内部 RAM 20H ～ 2FH 单元的位和部分 SFR 的位)

1. 立即寻址

立即寻址方式是操作数包含在指令字节中,指令操作码后面字节的内容就是操作数本身。汇编指令中,在一个数的前面冠以“＃”符号作前缀,就表示该数为立即数寻址。由于立即数是一个常数,所以只能为源操作数。

例如:

```
机器码     助记符          注释
74  5F     MOV  A,#5FH   ;5FH→A
```

该指令功能是将立即数 5FH 送入累加器 A,指令执行如图 3-1 所示。这条指令为双字节指令,操作数本身 5FH 跟在操作码 74H 后面,以指令形式存放在程序存储器内。

2. 寄存器寻址

操作数存放在寄存器中,指令中直接给出该寄存器名称的寻址方式称为寄存器寻址。采用寄存器寻址可以提高指令执行的速度。

例如:

```
MOV  A,R0
```

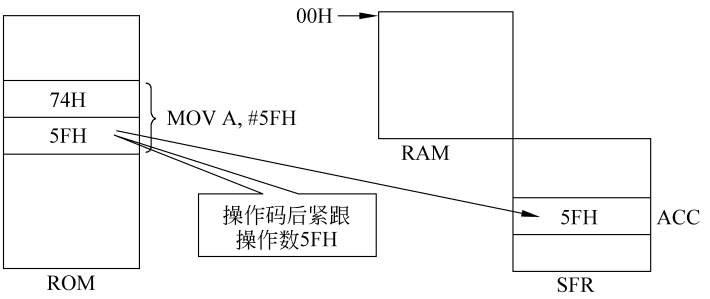

图 3-1　指令“MOV A，＃5FH”的执行示意图

该指令功能是将工作寄存器 R0 中的内容送到累加器 A 中，这里的操作数采用寄存器寻址。指令执行如图 3-2 所示。指令“MOV A，R0”在 ROM 中的编码为单字节：E8H。指令执行时，若当前工作寄存器组为 0 组（假定此时 RS1RS0 为 00），则 R0 指的就是 RAM 的 00H 单元。指令执行后，R0 中的数据（即“50H”）被复制到累加器 A 中，A 中原来的内容被覆盖。

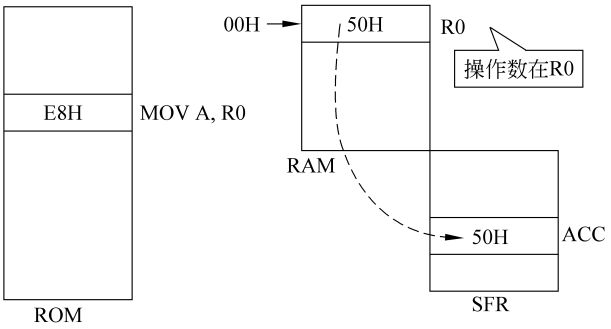

图 3-2　指令“MOV A，R0”的执行示意图

可使用寄存器寻址方式的寄存器或地址单元有：累加器 ACC、寄存器 B、位累加器 C、AB 双字节、工作寄存器 R0～R7、DPTR 等。

3. 直接寻址

参加运算的操作数地址在指令中直接给出，这种寻址方式就叫直接寻址。

例如：

```
MOV  A,3AH
```

在指令中，3AH 就是操作数的地址，要传送的操作数就存放在片内 RAM 3AH 字节单元中。执行指令如图 3-3 所示。

指令“MOV A，3AH”在 ROM 中的编码为双字节：E5H、3AH。指令执行时，若 3AH 单元的内容为“03H”，则指令执行后，3AH 单元的数据（即“03H”）被复制到累加器 A 中，A 中原来的内容被覆盖。

直接寻址方式不但可以访问内部 RAM 的低 128 个单元，还可以访问 SFR 区域。这时 SFR 常采用符号形式表示。如“MOV A，90H”可以写成“MOV A，P1”。这里的“P1”

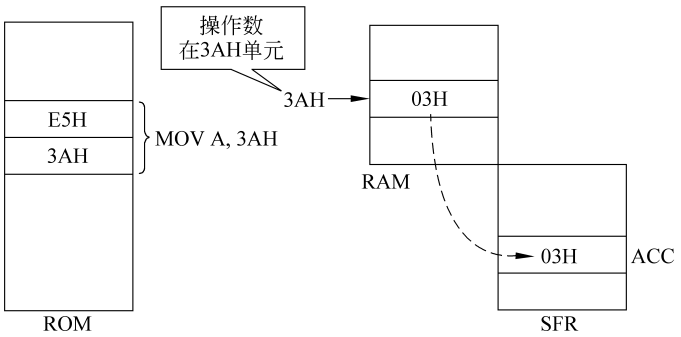

图 3-3　指令“MOV A，3AH”的执行示意图

与地址“90H”是等同的。

4. 寄存器间接寻址

寄存器间接寻址是指由指令指出某一个寄存器的内容作为操作数的地址的寻址方式，寄存器的内容不是操作数，而是操作数所在的存储器地址，操作数是通过寄存器间接得到的。

为了区别寄存器寻址，在寄存器间接寻址方式中，应在寄存器的名称前面加前缀“@”。例如：

```
MOV  A,@R0
```

该指令的功能是把 R0 所指出的内部 RAM 单元中的内容送累加器 A。若 R0 的内容为 60H，而内部 RAM 60H 单元中的内容是 3BH，则指令“MOV A，@R0”的功能是将 3BH 这个数送到累加器 A，如图 3-4 所示。

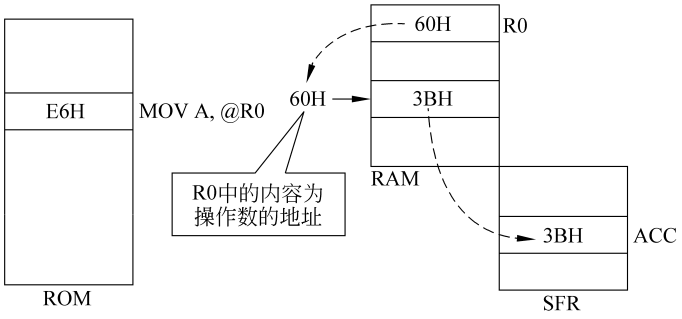

图 3-4　指令“MOV A，@R0”的执行示意图

寄存器间接寻址的寻址范围为：

① 片内 RAM 低 128B，这里只能用 R0 和 R1 作为间接寻址寄存器。

② 片外 RAM 低 64KB，使用 DPTR 作为间接寻址寄存器。

③ 片外 RAM 低 256B，可以使用 DPTR、R0 和 R1。

④ 堆栈区，以堆栈指针 SP 作为间接寻址寄存器。

5. 变址寻址

变址寻址方式用于访问程序存储器中的数据表格，它把基址寄存器(DPTR 或 PC)和变址寄存器 A 的内容作为无符号数相加形成 16 位的地址，访问程序存储器中的数据表格。操作时是以某个寄存器的内容为基础，然后在这个基础上再加上地址偏移量，形成真正的操作数地址。需要特别指出的是，用来作为基址的寄存器可以是 PC 或是 DPTR，地址偏移量存储在累加器 A 中。变址寻址用于两种情况：一是用于对 ROM 中的数据进行寻址(MOVC 类)；二是用于跳转指令(如 JMP @A＋DPTR)。

例如：

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

A 中为无符号数，该指令的功能是 A 的内容和 DPTR 的内容相加得到程序存储器的有效地址，把该存储器单元中的内容送到 A，如图 3-5 所示。

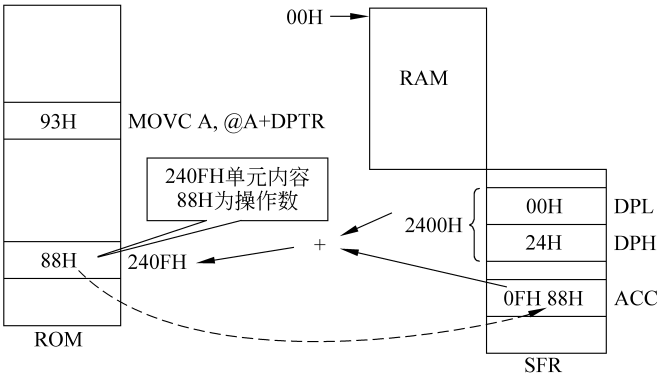

图 3-5　指令“MOV A，@A＋DPTR”的执行示意图

变址寻址的寻址空间为 ROM 空间(采用@A＋DPTR，@A＋PC)。

6. 相对寻址

相对寻址主要是针对跳转指令而言的。对于跳转指令，跳转去的目标指令的地址是通过正在执行的指令地址来确定的，一般是采用正在执行的指令地址加上偏移量的方式。

这类寻址方式是以当前 PC 的内容作为基地址，加上指令中给定的偏移量所得结果作为转移地址，它只适用于双字节转移指令。偏移量是带符号数，在－128～＋127 范围内用补码表示。

例如：

```
JC  rel
```

指令“JC rel”在 ROM 中的编码为双字节，若偏移量 rel 为 06H，则编码为 40H、06H。第一字节为操作码，第二字节就是相对于程序计数器 PC 当前地址的偏移量 rel。若转移指令操作码存放在 1000H 单元，偏移量存放在 1001H 单元，该指令执行后 PC 已为 1002H，则转移到的目标地址为 1002H＋06H＝1008H，即当 C＝1 时，将去执行 1008H 单元中的指令，如图 3-6 所示。

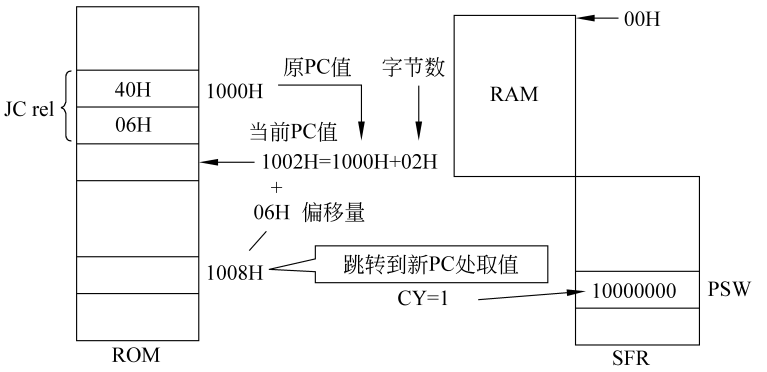

图 3-6　指令“JC rel”的执行示意图

7. 位寻址

对位地址的内容进行操作的寻址方式称为位寻址。采用位寻址指令的操作数是 8 位二进制数中的某一位。指令中给出的是位地址。位寻址方式实质上属于位的直接寻址。

例如：

```
MOV  C,00H
```

该指令的功能是把位地址为 00H 的位状态送至进位标志位 C。

位寻址方式的寻址范围包括：

(1) 片内 RAM 中的位寻址区。单元地址为 20H～2FH，共 16 个单元 128 位，位地址是 00H～7FH。对这 128 个位的寻址用直接位地址表示，例如，指令 MOV C,2BH 的功能是把位寻址区的 2BH 位状态送至进位标志位。

(2) 专用寄存器的可寻址位。可供位寻址的专用寄存器共有 11 个，实有寻址位 83 位，对这些寻址位在指令中有以下 4 种表示方法：

① 直接位地址(00H～0FFH)，如 20H。

② 字节地址带位号，如 20H.3 表示 20H 单元的 3 位。

③ 特殊功能寄存器名带位号，如 P0.1 表示 P0 口的 1 位。

④ 位符号地址，如 TR0 是定时器/计数器 T0 的启动位。

3.3　80C51 指令系统

3.3.1　数据传送类指令

80C51 指令系统中，数据传送指令共有 29 条。这类指令将源操作数传送到指定的目的地址，传送后源操作数保持不变。

数据传送类指令用到的助记符有 MOV、MOVX、MOVC、XCHD、PUSH、POP、XCH、XCHD、SWAP 等。数据传送类指令见表 3-4。

表 3-4 数据传送类指令

指令助记符	说 明	机器码字节	字节数	周期数
MOV A,R*n*	A←(R*n*)	E8H(～EFH)	1	12
MOV A,direct	A←(direct)	E5H direct	2	12
MOV A,@R*i*	A←((R*i*))	E6H(～E7H)	1	12
MOV A,＃data	A←data	74H data	2	12
MOV R*n*,A	R*n*←(A)	F8H(～FFH)	1	12
MOV R*n*,direct	R*n*←(direct)	A8H(～AFH) direct	2	24
MOV R*n*,＃data	R*n*←data	78H(～7FH) data	2	12
MOV direct,A	direct←(A)	F5H direct	2	12
MOV direct,R*n*	direct←(R*n*)	88H(～8FH) direct	2	24
MOV direct1,direct2	direct1←(direct2)	85H direct2 direct1	3	24
MOV direct,@R*i*	direct←((R*i*))	86H(～87H) direct	2	24
MOV direct,＃data	direct←data	75H direct data	3	24
MOV @R*i*,A	(R*i*)←(A)	F6H(A7H)	1	12
MOV @R*i*,direct	(R*i*)←(direct)	A6H(～A7H) direct	2	24
MOV @R*i*,＃data	(R*i*)←data	76H(～77H) data	2	12
MOV DPTR,＃data16	DPTR←data16	90H dataH dataL	3	24
MOVC A,@A＋DPTR	A←((A)＋(DPTR))	93H	1	24
MOVC A,@A＋PC	A←((A)＋(PC))	83H	1	24
MOVX A,@R*i*	A←((R*i*))	E2H(～E3H)	1	24
MOVX A,@DPTR	A←((DPTR))	E0H	1	24
MOVX @R*i*,A	(R*i*)←(A)	F2H(～F3H)	1	24
MOVX @DPTR,A	(DPTR)←(A)	F0H	1	24
PUSH direct	SP←(SP)＋1,(SP)←(direct)	C0H direct	2	24
POP direct	direct←((SP)),SP←(SP)－1	D0H direct	2	24
XCH A,R*n*	(A)←→(R*n*)	C8H(CFH)	1	12
XCH A,direct	(A)←→(direct)	C5H direct	2	12
XCH A, @R*i*	(A)←→((R*i*))	C6H(～C7H)	1	12
XCHD A,@R*i*	(A)3～0←→((R*i*))3～0	D6H(～D7H)	1	12
SWAP A	(A)3～0←→(A)7～4	C4H	1	12

数据传送类指令源操作数和目的操作数的寻址方式及传送路径如图 3-7 所示。

1. 内部 RAM 和特殊功能寄存器(SFR)数据传送指令

(1) 以累加器 A 为目的操作数的指令：

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

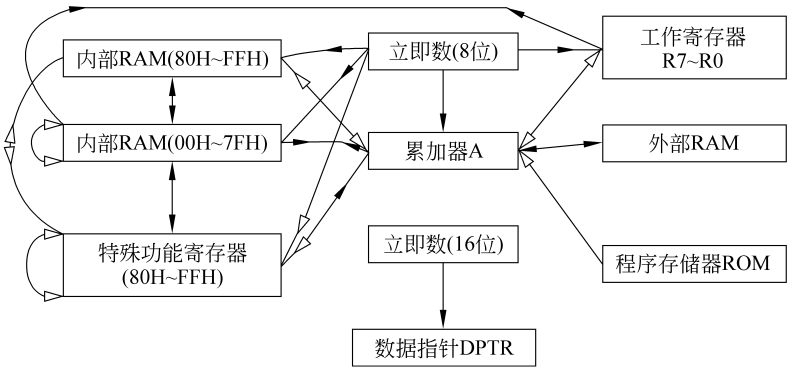

图 3-7　数据传送类指令源操作数和目的操作数的寻址方式及传送路径

这组指令的功能是把源字节送入累加器中。源字节的寻址方式分别为寄存器寻址、直接寻址、寄存器间接寻址和立即寻址四种基本寻址方式。

由于 80C51 单片机仍然属于以累加器为中心的结构，多数数据的处理都要通过累加器完成。所以，以累加器 A 为目的操作数的指令在整个指令系统中是使用最为频繁的指令。

【例 3-1】 若(R1)＝20H，(20H)＝55H，执行指令“MOV A，@R1”后，(A)＝55H。

(2) 以 R*n*、DPTR 为目的操作数的指令：

```
MOV  Rn,    A          ;Rn←(A)
MOV  Rn,    direct     ;Rn←(direct)
MOV  Rn,    #data      ;Rn←data
MOV  DPTR,  #data16    ;DPTR←data16
```

这组指令的功能是把源地址单元中的内容送入工作寄存器，源操作数不变。最后一条指令是唯一的一条 16 位立即数传送指令，其功能是将一个 16 位的立即数送入 DPTR 中。其中高 8 位送入 DPH，低 8 位送入 DPL。

【例 3-2】 指令“MOV R6，＃5AH”执行后，(R6)＝5AH。

【例 3-3】 指令“MOV DPTR，＃1234H”执行后，(DPH)＝12H，(DPL)＝34H。

(3) 以直接地址 direct 为目的操作数的指令：

```
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
```

这组指令的功能是把源字节送入 direct 中。源字节的寻址方式分别为寄存器寻址、直接寻址、寄存器间接寻址和立即寻址。

【例 3-4】 若(R1)＝0FH，(0FH)＝12H，执行指令“MOV 30H，@R1”后，(30H)＝12H。