单片微机 MCS-51 用户手册

涂时亮 张友德 编译
陈章龙 审校

复旦大学出版社

单片微机 MCS-51 用户手册

涂时亮　张友德　编译

陈章龙　　审校

复旦大学出版社

(沪)新登字202号

单片微机 MCS-51 用户手册

涂时亮　张友德　编译

责任编辑　陆盛强

复旦大学出版社出版

(200433上海国权路579号)

新华书店上海发行所发行　崇明晨光印刷厂印刷

开本787×1092　1/16　印张25.25　字数614千字

1990年9月第1版　　1992年5月第2次印刷

印数　13800

ISBN7-309-00515-5/T・20　　定价：10.00元

前　言

我室编译的《MCS-51用户手册》已发行了几万册，受到了广大读者的欢迎。

INTEL公司1989年的《8-Bit Embedded Controller Handbook》与原先的版本在叙述上有很大的不同，并增加许多新的内容。在INTEL公司的支持下，我们根据最新的1989年版本重新进行了编译。

新的用户手册不但包含了原有的8031/8051/8751、8032/8052/8752、80C31/80C51/87C51的内容，还包括了80C51FA(FB)/83C51FA(FB)/87C51FA(FB)、80C51GA/83C51GA/87C51GA、80C152/83C152、80C451/83C451和80C452/83C452的内容，使MCS-51成为真正应用最广的八位单片机。本书中，第一章、第二章、第三章和第六章由涂时亮同志编译；第四章、第五章、第七章和第八章由张友德同志编译；全书由陈幸龙同志校审。在本书的编译过程中得到了徐君毅同志和复旦大学计算机科学系微机室同志的支持和帮助。

由于编译者水平有限，如有错漏疏误之处，敬请指正。

编　者

1989.9

目　录

第一章　MCS-51 微控制器系列总体结构

§1.1　引　　言

8051 是 MCS-51 的始祖，也是所有 MCS-51 器件的核心。8051 核心具有如下特性：

- 适于控制应用的 8 位 CPU
- 扩展的布尔处理(单一位逻辑)能力
- 64K程序存贮器空间
- 64K数据存贮器空间
- 4K字节片内程序存贮器
- 128 字节片内数据 RAM
- 32根双向和可单独寻址的 I/O 线
- 两个16位定时/计数器
- 全双工UART
- 6源/5向量中断结构，具有两个优先级
- 片内时钟振荡器。

8051 核心的基本总体结构见图 1.1。

MCS-51 系列的每一种器件具有 8051 核心的特性和一些附加的特性。所有MCS-51器件的特性对比见表 1.1。

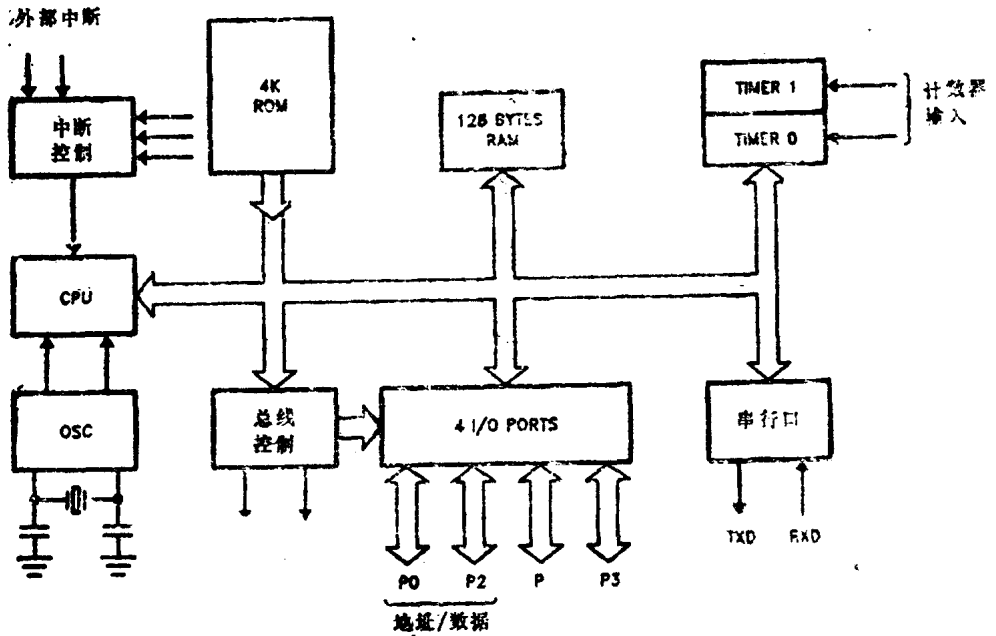

图 1.1　8051核心结构框图

表 1.1 MCS-51 器件的特性对比

器件	无ROM型号	EPROM型号	ROM字节数	RAM字节数	8位I/O口	16位定时/计数器	可编程计数器阵列(PLA)	UART	串行扩展口(SEP)	全局串行通道(GSC)	DMA通道	A/D通道	中断源/向量数	掉电和空闲方式	电路工艺
8051	8031	—	4K	128	4	2		✓					6/5		HMOS
8051AH	8031AH	8751H 8751BH	4K	128	4	2		✓					6/5		HMOS
8052AH	8032AH	8752BH	8K	256	4	3		✓					8/6		HMOS
80C51BH	80C31BH	87C51	4K	128	4	2		✓					6/5	✓	CHMOS
83C51FA	80C51FA	87C51FA	8K	256	4	3	✓	✓					14/7	✓	CHMOS
83C51FB	80C51FA	87C51FB	16K	256	4	3	✓	✓					14/7	✓	CHMOS
83C51GA	80C51GA	87C51GA	4K	128	4	2		✓	✓			8	8/7	✓	CHMOS
83C152JA	80C152JA	—	8K	256	5	2		✓		✓	2		19/11	✓	CHMOS
	80C152JB	—	—	256	7	2		✓		✓	2		19/11	✓	CHMOS
83C152JC	80C152JC	—	8K	256	5	2		✓		✓	2		19/11	✓	CHMOS
—	80C152JD	—	—	256	7	2		✓		✓	2		19/11	✓	CHMOS
83C451	80C451	—	4K	128	7	2		✓					6/5	✓	CHMOS
83C452	80C452	87C152P	8K	256	5	2		✓					9/8	✓	CHMOS

CHMOS 器件

从功能上来说，CHMOS 器件(器件名称的中间有一个"C"字母)与 8051 完全兼容，但作为 CMOS，它消耗的电流比 HMOS 少得多。为进一步利用 CMOS 电路的省电特性，加入了两种掉电工作方式：

·软件启动空闲方式

在该方式中，CPU 停止工作，但 RAM 和其他片内外围电路继续工作。在这种方式中，消耗的电流降低为正常工作时的大约15%。

·软件启动掉电方式

在该方式中，停止芯片的一切工作。片内 RAM 继续保持数据。在这种方式中，器件消耗的电流一般小于 10uA。

虽然 80C51BH 等 CMOS 器件与 HMOS 在功能上完全兼容，但如果想使 HMOS 和 CHMOS 器件能完全互换，则在设计应用电路时必须注意这两种器件之间的不同之处：

(1) 用 CHMOS 工艺制成的为动态电路，所以它的工作频率不能太低，一般应在 0.5 MH_z 以上。但 CHMOS 器件中的片内 RAM 不是动态的，仅是 CPU 为动态。

(2) CHMOS 电路的功耗与频率有关，一般来说频率越高功耗越大。功耗最低的工作频率为 1.5MH_z 左右。

(3) CHMOS 的电平与 HMOS 不同：

在驱动 CMOS 电路时，CHMOS(80C51BH 等)的输出高电平大于 4.5V，低电平小于 0.45V，可直接驱动 74HC 电路(输入高电平应大于 3.5V，输入低电平应小于 1.0V)。而 HMOS(8051)的输出高电平大于 2.4V，低电平小于 0.45V。

CHMOS 要求的输入高电平为 1.9V，低电平为 0.9V。而 HMOS 则分别为 2.0V 和 0.8V。

(4) P0口电平：

CMOS 电路的输入脚不能处于浮空状态。80C51BH 等的 P1、P2、P3 口均有内部拉电电阻，不会处于浮空状态。但 P0 口作输入时可能处于浮空状态。而在掉电方式时，即使是 80C31BH 的 P0 口也将处于浮空状态。为了防止这种情况，可把 P0 口通过电阻接高电平或接地，电阻应为 10K 左右。如果与 P0 口相连的电路在掉电时没有电源，则该电阻以接地为好。

关于表 1.1 中各种器件的详细介绍可参看本书中它们的硬件描述和技术手册。

§1.2 MCS-51 的存贮器组织

1.2.1 逻辑上分开的程序和数据存贮器

所有的 MCS-51 器件都有分开的程序和数据存贮器，见图 1.2。程序和数据存贮器的逻辑上的分开允许用 8 位地址访问数据存贮器，对 8 位 CPU 来说，8 位地址可更快地存放和形成。然而，也能用 DPTR 寄存器产生 16 位数据存贮器地址。

程序存贮器只能读出，不能写入。最多可有64K程序存贮器。对8051、8051AH、80C51BH 和它们的 EPROM 型号，程序存贮器的低 4K 字节在片内。8052AH、83C51FA 和

83C152 具有 8K 字节片内程序存贮器。对无 ROM 的型号所有的程序存贮器均在外部。外部程序存贮器的读出脉冲是 $\overline{PSEN}$ 信号(程序存贮允许)。

数据存贮器位于与程序存贮器分开的地址空间。在外部数据地址空间最多可寻址 64K 外部 RAM,在访问外部数据存贮器时,CPU 按需产生读出信号 $\overline{RD}$ 和写入信号 $\overline{WR}$。

在需要时通过把 $\overline{RD}$ 和 $\overline{PSEN}$ 信号接到一个与门上，与门的输出当作外部程序/数据存贮器的读脉冲，可把外部程序存贮器和外部数据存贮器合并起来。

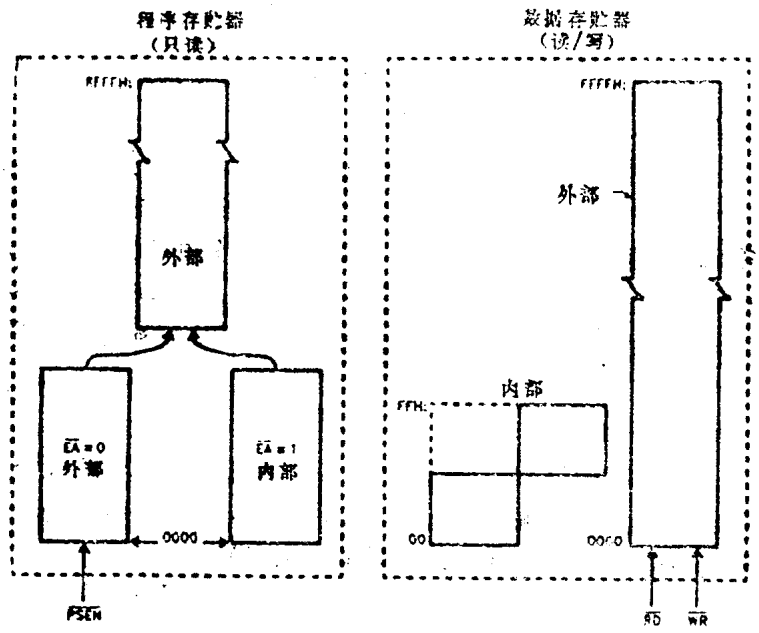

图 1.2 MCS-51存贮器结构

1.2.2 程序存贮器

图 1.3 显示了程序存贮器低部分的结构。复位后 CPU 从地址 0000H 开始执行。

如图 1.3 所示，每一个中断赋于程序存贮器中的一个固定地址。中断使 CPU 转向该地址，并从这里开始执行服务程序。例如，外部中断 0 对应于地址 0003H,如果使用外部中断 0，则服务程序必须从地址 0003H 开始。如果不使用该中断，则该地址可用作一般的程序存贮器。

各中断服务程序入口地址的间隔为 8 字节：外部中断 0 为0003H,定时器 0 为000BH，外部中断 1 为 0013H，定时器 1 为 001BH 等。如果一个中断服务程序足够短(如控制应用场合)，它可全部位于这 8 字节间隔中。如果使用其他中断时，长的服务程序可用跳转指令跳过后面的中断入口地址。

低 4K(或 8K，对 8052AH、83C51FA 和 83C152)字节程序存贮器可以在片内ROM 也可以在外部 ROM 中。这可通过把 $\overline{EA}$ (外部访问)脚接 V_{CC} 或 V_{SS} 来选择。

对8051和它的派生产品，如果 $\overline{EA}$ 脚指 V_{CC}，则从地址 0000H 到 0FFFH 取指将对内部 ROM 进行。从地址 1000H 到 FFFFH 取指总是对外部 ROM 进行。

对 8052AH 和其他 8KROM 产品，$\overline{EA}$ = V_{CC} 选择地址 0000H 到 1FFFH 为内部

ROM，而地址 2000H 到 FFFFH 为外部 ROM。

如果 $\overline{EA}$ 脚接 V_{SS}，则所有的取指操作均对外部 ROM 进行。对无 ROM 的产品(8031、8032AH 等)，$\overline{EA}$ 脚必须接 V_{SS} 以允许它们从外部程序存贮器执行程序。

外部 ROM 的读脉冲 $\overline{PSEN}$ 用于所有的外部程序取指操作。在读取内部程序时，不产生 $\overline{PSEN}$。

执行外部程序的硬件结构见图 1.4。在外部程序存贮器寻址时，16 根I/O线(P0和P2)用作总线功能。P0 用作多路地址/数据总线。它输出程序计数器的低位字节(PCL)作为地址，然后变为浮空状态等待从程序存贮器读出的指令码。在 P0 口上程序计数器低位字节有效期间，信号 ALE(地址锁存允许)把该字节打入地址锁存器中。同时，口 2(图 1.4 中 P2)输出程序计数器的高位字节(PCH)。然后 $\overline{PSEN}$ 选通 EPROM 并把指令码读入微控制器中。

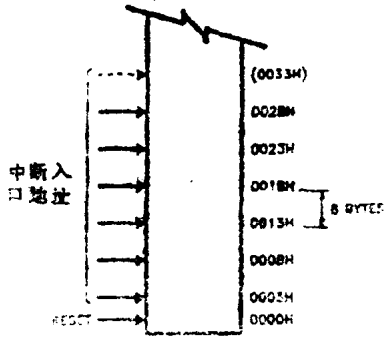

图 1.3 MCS-51程序存贮器

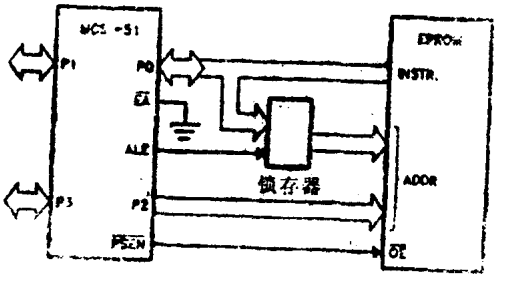

图 1.4 扩展外部程序存贮器

即使实际使用的程序存贮器少于 64K 字节，程序存贮器地址总是 16 位宽。执行外部程序牺牲了两个 8 位口，P0 和 P2，用作程序存贮器寻址。

1.2.3 数据存贮器

图 1.2 的右半部列出了 MCS-51 用户可使用的内部和外部数据存贮器空间。

图 1.5 给出了可访问2K字节外部RAM 的硬件结构。图中 CPU 执行内部 ROM 中的程序。口 0 用作 RAM 的多路地址/数据总线，口 2 的 3 根线用作 RAM 的页寻址。在访问外部 RAM 时 CPU 按需产生 $\overline{RD}$ 和 $\overline{WR}$信号。

最多可外接 64K 字节数据存贮器。外部数据存贮器地址可以是一或二个字节。一个字节地址常与用一根或几根 I/O 线的 RAM 页寻址配合使用，如图 1.5 所示。也可使用二字节地址，这时高位地址从 P2 口输出。

内部数据存贮器的映象图见图1.6。如图所示，存贮器空间分为三块：低 128 字节，高 128 字节的和 SFR 空间。

内部数据存贮器地址总是一个字节，所以地址空间为 256 字节。然而，使用一个简单的技巧，外部 RAM 寻址方式可访问 384 字节。高于 7FH 的直接寻址访问一个存贮空间，而高于 7FH 的间接寻址可访问另一个存贮空间。这样如图 1.6 所示，虽然高 128 字节和 SFR 空间为物理上不同的实体，但它们可占据同一个地址空间 80H 到 0FFH。

所有的 MCS-51 器件都有低 128 字节 RAM，其映象图见图 1.7。最低的 32 个字节分

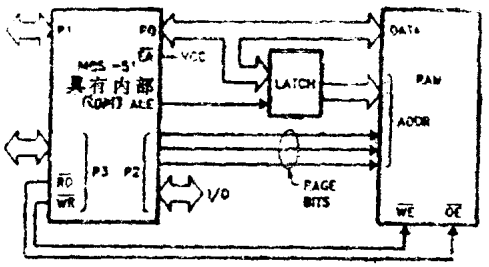

图 1.5 访问外部数据存贮器(假如程序存贮器在内部，P2的其他位可用作I/O)

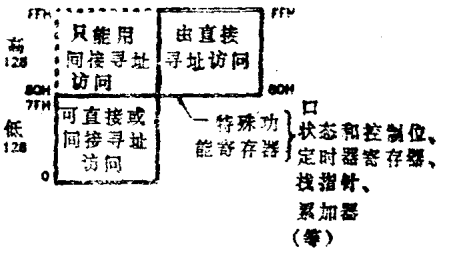

图 1.6 内部数据存贮器

成 4 个 8 寄存器区。程序指令可用R0到 R7 来访问这些寄存器。程序状态字 (PSW) 中的二位可选择使用哪一个寄存器区。这可提高编码空间的效率，因为寄存器指令比使用直接寻址的指令短。

寄存器区上面的 16 字节为位可寻址存贮器空间。MCS-51 指令系统包括各种位操作指令，它们可直接访问前述空间中的 128 位，位地地为 00H 到 7FH。

低 128 的所有字节都可由直接或间接寻址所访问。高 128 字节只能由间接寻址所访问。8051 没有高 128 字节 RAM，而 8052AH、83C51FA 和 83C152 有这些 RAM。

图 1.8 给出了特殊功能寄存器(SFR)空间的简单示意图。SFR 包括口锁存器、定时器、外围设备控制等。这些寄存器只能用直接寻址来访问。一般来说，所有的MCS-51微控制器都具有与 8051 相同的 SFR，并且地址也相同。但是8051的增强型各有一些新的、8051 没有的 SFR。

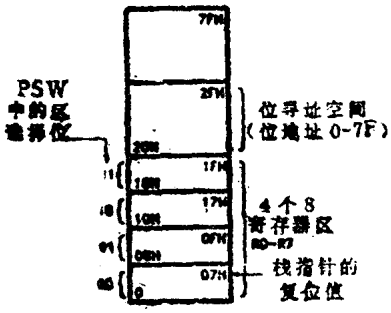

图 1.7 低128字节内部RAM

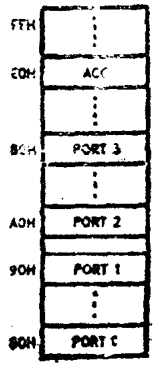

图 1.8 SFR空间

SFR 空间中的 16 个地址既可字节寻址又可位寻址。位可寻址的 SFR 的地址的低 3 位为 000B。该区间中的位地址为 80H 到 0FFH。

§1.3 MCS-51 指令系统

所有的 MCS-51 成员都具有相同的指令系统。MCS-51 指令系统特别适于 8 位控制应用。它有各种访问内部 RAM 的快速寻址方式以方便对小的数据结构进行操作。该指令系

统可支持一种独立的数据类型——位变量，允许在需要布尔处理的控制和逻辑系统中对位的操作。

下面全面介绍MCS-51的指令系统，并简述某些指令的使用方法。文中提到的"汇编器"是Intel的MCS-51宏汇编器——ASM51。关于指令系统的更详细的介绍可参看MCS-51宏汇编用户手册。

1.3.1 程序状态字

程序状态字(PSW)包括一些反映CPU当前状态的状态位。PSW处于SFR空间，它包括进位位、辅助进位位(用于BCD操作)、两个寄存器区选择位、溢出标志位、奇偶位和两个用户定义状态标志位。

进位位除了用作算术运算的进位标志外，还是各种位操作的"累加器"。

位RS0和RS1用于选择图1.7中四个寄存器区之一。许多指令可用R0到R7来访问这些RAM单元。选择这四个区中的哪一个由运行时RS0和RS1的状态决定。

奇偶位反映累加器中"1"的数目：如累加器包含奇数个1则P=1；如累加器包含偶数个1则P=0。这样累加器和P中1的数目始终是偶数。

PSW中另外两位是独立的，可用作通用状态标志位。

1.3.2 寻址方式

MCS-51指令系统具有如下寻址方式：

1. 直接寻址

在直接寻址方式里操作数由指令中的一个8位地址场所指定。只有内部数据RAM和SFR可以直接寻址。

2. 间接寻址

在间接寻址方式里指令指出的一个寄存器包含操作数的地址。内部RAM和外部RAM都可以间接寻址。

8位地址的地址寄存器可以是选中寄存器区的R0或R1，也可以是堆栈指针。16位地址的地址寄存器只能是16位"数据指针"寄存器DPTR。

3. 寄存器指令

寄存器区包含寄存器R0到R7，能使用某些指令来访问。这些指令的操作码中有三个寄存器选择位。使用这种方法访问寄存器的指令的编码效率比较高，因为它不需要地址字节。执行这些指令时，访问选中的寄存器区的8个寄存器之一。运行时可通过PSW中的两个区选择位来选择四个区中的任一个。

4. 特定寄存器指令

有些指令指定于某个寄存器。例如，有些指令总是对累加器或数据指针等进行操作，故不需要指向它的地址字节，指令码本身隐含了操作对象。指定累加器为A的指令汇编成累加器操作指令码。

5. 立即数寻址

常数值能放于程序存贮器的操作码后。例如：MOV A，#100

把十进制数100装入累加器中。这个数也能写成16进制数64H。

6. 变址寻址

只有程序存贮器能用变址寻址访问，并且只能读出。这种寻址方式用于读出程序存贮器中的检索表。16位基寄存器(DPTR 或程序计数器)指向表格的基地址，累加器存放表格入口序号。程序存贮器中的表格入口地址可通过把累加器的数据加到基指针上得到。

1.3.3 算术指令

算术指令见表 1.2。表中指出每种指令可使用的访问<byte>操作数的寻址方式。例如，ADD A，<byte>指令能写成：

```
ADD   A, 7FH      (直接寻址)
ADD   A, @R0      (间接寻址)
ADD   A, R7       (寄存器寻址)
ADD   A, #127     (立即常数)
```

表 1.2 所列的执行时间为 12MHz 时钟频率下的执行时间。除了 INC DPTR 和乘、除指令外，所有的算术指令的执行时间为 1μs，INC DPTR 为 2μs，而乘、除指令为 4μs。

表 1.2 MCS-51算术指令

助记符	操作	寻址方式				执行时间(μs)
		直接	间接	寄存器	立即	
ADD A, <byte>	A = A + <byte>	×	×	×	×	1
ADDC A, <byte>	A = A + <byte> + C	×	×	×	×	1
SUBB A, <byte>	A = A − <byte> − C	×	×	×	×	1
INC A	A = A + 1	仅累加器				1
INC <byte>	<byte> = <byte> + 1	×	×	×		1
INC DPTR	DPTR = DPTR + 1	仅数据指针				2
DEC A	A = A − 1	仅累加器				1
DEC <byte>	<byte> = <byte> − 1	×	×	×		1
MUL AB	B：A = B×A	仅ACC和B				4
DIV AB	A = INT[A/B] B = MOD[A/B]	仅ACC和B				4
DA A	十进制调整	仅累加器				1

注意，内部数据存贮器空间中的任一字节可以不通过累加器实现加1或减1操作。

一个INC指令对16位数据指针进行操作。数据指针用于产生外部存贮器的16位地址，因此该16位的加1操作非常有用。

MUL AB指令把累加器和寄存器B中的数据相乘，16位的积放于B和累加器相连而成的寄存器对中。

DIV AB指令把累加器除以B寄存器的数据，8位商放于累加器中，8位余数在B寄存器中。

非常奇怪，DIV AB指令在算术除法程序中用处不大，相反它常用于数制转换和可程控移位。后面特给出DIV AB在数制转换中的应用例子。在移位操作中，除以2^n相当于把它右移几位。使用DIV AB执行除法可在4μs内完成移位，在B寄存器中存放移出的位。

DA A指令用于BCD算术操作。在BCD运算中，ADD和ADDC指令应后按DAA操作，以保证结果也为BCD码。注意，DA A不能将二进制数转换成BCD码。DA A只有在用于两个BCD字节相加后的第二步操作时才有意义。

1.3.4 逻辑指令

表1.3列出了MCS-51的逻辑指令，对字节执行布尔运算(AND，OR，异或，NOT)的字节按位进行操作。这就是说，如果累加器内容为00110101B，而<byte>内容为01010011B，则执行

```
ANL A, <byte>
```

后，累加器将存放00010001B。

表1.3给出了访问<byte>可使用的寻址方式。这样，ANL A，<byte>指令可能为如下形式之一：

```
ANL   A, 7FH     (直接寻址)
ANL   A, @R1     (间接寻址)
ANL   A, R6      (寄存器寻址)
ANL   A, #53H    (立即常数)
```

所有的累加器操作逻辑指令的执行时间为1μs(使用12MHz时钟)，其他为2μs。

注意布尔操作可对内部数据存贮器空间的任一字节进行，不必要通过累加器。例如，XRL <byte>，#data指令可快速方便地取反口的位，如

```
XRL  P0, #0FFH
```

如果该操作是响应中断时执行，则不使用累加器在服务程序中可节省保护它的指令和时间。

移位指令(RL A、RLC A等)把累加器左移或右移一位。对左环移操作，最高位(MSB)移入最低位(LSB)中。对右环移操作，LSB移入MSB位置。

SWAP A交换累加器的高4位和低4位。在BCD操作中这是很有用的。例如，累加器中包含一个小于100的二进制数，可以用以下代码很快地转换成BCD码：

```
MOV  B, #10
DIV  AB
```

表 1.3 MCS-51逻辑指令

助记符	操作	寻址方式				执行时间(μs)
		直接	间接	寄存器	立即	
ANL A, <byte>	A = A.AND.<byte>	×	×	×	×	1
ANL <byte>, A	<byte> = <byte>.AND.A	×				1
ANL <byte>, #data	<byte> = <byte>.AND.#data	×				2
ORL A, <byte>	A = A.OR.<byte>	×	×	×	×	1
ORL <byte>, A	<byte> = <byte>.OR.A	×				1
ORL <byte>, #data	<byte> = <byte>.OR.#data	×				2
XRL A, <byte>	A = A.XOR.<byte>	×	×	×	×	1
XRL <byte>, A	<byte> = <byte>.XOR.A	×				1
XRL <byte>, #data	<byte> = <byte>.XOR.#data	×				2
CLR A	A = 00H	仅累加器				1
CPL A	A = .NOT.A	仅累加器				1
RL A	ACC左移一位	仅累加器				1
RLC A	ACC通过进位左移一位	仅累加器				1
RR A	ACC右移一位	仅累加器				1
RRC A	ACC通过进位右移一位	仅累加器				1
SWAP A	ACC半字交换	仅累加器				1

SWAP A

ADD A, B

把一个数除以 10 则十位数字在累加器的低 4 位中，个位数字在B寄存器中。SWAP和ADD 指令把十位数字移至累加器的高 4 位，并把个位数字移至累加器的低 4 位。

1.3.5 数据传送

1. 内部 RAM