7.1 机器指令

一、指令的一般格式

这只是一种逻辑的表达方式，实际上位置不是这么严格区分的。

操作码字段	地址码字段

1.操作码：反映机器做什么操作、对什么数据做操作等

（1）长度固定

用于**指令字长较长**的情况：RISC

（2）长度可变

操作码分散在**指令字**的**不同**字段中。

（3）扩展操作码技术

操作码的位数随地址数的减少而增加。可以采用**保留码点**来指明扩展操作码技术。**除非地址为零，否则操作码不可以全为 1！**扩展原则：**短操作码一定不是长操作码的前缀**。

三地址指令操作码每减少一种最多可多构成 2^4 种二地址指令。

二地址指令指令操作码每减少一种操作码最多可构成 2^4 种一地址指令。

一般来说，高频指令用短操作码表示，不常出现的用长操作码表示。

2.地址码

（1）四地址

OP	A1	A2	A3	A4

说明：

A1 是第一操作数地址；A2 是第二操作数地址；A3 是结果的地址；A4 是下一条指令的地址。

假设当前操作是(A1)OP(A2)→A3，取指令访存 1 次，然后根据地址取出两个操作数访存 2 次，最后将结果存入 A3 的地址访存 1 次，一共**访存 4 次**。假设指令字长为 32 位，操作码为 8 位，每个地址码长度就为 6 位，那么每个地址码寻址范围为 **64**，可访问内存空间非常小，这样的地址几乎是不可用的。

实际上，我们可以用 PC 代替 A4，那么就变成了 3 地址了。

（2）三地址

OP	A1	A2	A3

同样执行上述操作，还是 4 次访存，但是寻址范围变为了 **256**。

（3）二地址

OP	A1	A2

若结果存于 ACC，则只需要 3 次访存，若结果存于 A1 或者 A2 则依旧是 4 次访存。寻址范围变为了 2^{12}=**4K**。

（4）一地址

OP	A1

2 次访存，寻址范围为 2^{24}=**16M**。

（5）零地址。无地址码。

二、指令字长

指令字长决定于：**操作码的长度；操作数地址的长度；操作数地址的个数**。

1.指令字长固定：指令字长=存储字长。

2.指令字长可变：按字节的倍数变化。

小结

当用一些硬件资源代替指令字长中的地址码字段后：

（1）可扩大指令的寻址范围；

（2）可缩短指令字长；

（3）可减少访存次数。

当指令的地址字段为**寄存器**时：

（1）三地址：OP R1 R2 R3；

（2）二地址：OP R1 R2；

（3）一地址：OP R1。

这种方式**可缩短指令字长，指令执行阶段不访存**。

7.2 操作数类型和操作类型

一、操作数类型

（1）地址：无符号整数（绝对寻址）、有符号数（相对寻址）

（2）数字：定点数、浮点数、十进制数

（3）字符：ASCII

（4）逻辑数：逻辑运算

二、数据在存储器中的存储方式

1.字节编址：数据在存储器中的存储方式（存储字长 64 位，机器字长 32 位）

a.从任意位置开始存储

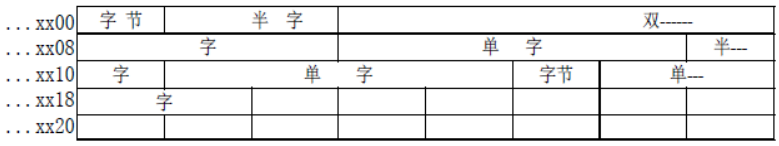

图 2.1 任意位置存储

优点：不浪费存储资源；缺点：除了访问一个字节之外，访问其它任何类型的数据都可能话费两个存储周期的时间。读写控制比较复杂。

b.从一个存储字的起始位置开始访问

图 2.2 从一个存储字的起始位置开始访问

优点：无论访问何种类型的数据，在一个周期均可完成，读写控制简单。但是缺点是浪费了宝贵的存储资源。

c.边界对准方式：从地址的整数倍位置开始访问

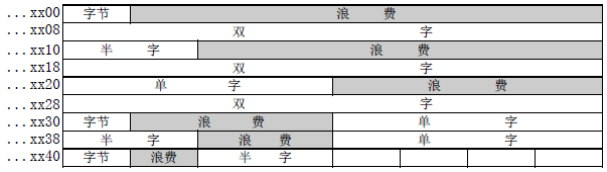

图 2.3 边界对准方式

数据存储的起始地址是数据长度（按照编址单位进行计算）的整数倍。本方案是前两个方案的折中，在一个周期内可以完成存储访问，空间浪费也不太严重。

三、操作类型

1.数据传送

源	寄存器	寄存器	存储器	存储器
目的	寄存器	存储器	寄存器	存储器
例子	MOVE	STORE/MOVE/PUSH	LOAD/MOVE/POP	MOVE

还有一种是置“1”，清“0”。

2.算术逻辑操作

加、减、乘、除、增 1、减 1、求补、浮点运算、十进制运算、与、或、非、异或。位操作、位测试、位清除、位求反。

3.移位操作

算术移位、逻辑移位、循环移位（带进位和不带进位）

4.转移

（1）无条件转移：JMP；

（2）条件转移：结果为 0 转：JZ，结果为溢出转：JO，结果又进位转：JC，跳过一条指令：SKP；

（3）调用和返回：CALL/RETURN；

（4）陷阱（Trap）与陷阱指令

意外事故的中断：

①一般不提供给用户直接使用。在出现事故时，由 CPU 自动产生并执行（指令）。

②设置供用户使用的陷阱指令。

如 8086 的 INT TYPE 软中断。提供给用户使用的陷阱指令，完成系统调试。

5.输入输出

入：端口中的内容→CPU 的寄存器。如：IN AL, n。

出：CPU 的寄存器→端口中的内容。如：OUT n, AL。

7.3 寻址方式

寻址方式是确定**本条**指令的**操作数地址**、**下一条**要执行指令的**指令地址**。

一、指令寻址

1.顺序寻址：(PC)+1→PC

2.跳跃寻址：由转移指令指出下一条指令地址。

二、数据寻址

指令格式：

操作码	寻址特征	形式地址 A

形式地址：指令字中的地址。

有效地址：操作数的真实地址。

我们做如下约定，指令字长=存储字长=机器字长

1.立即寻址

形式地址 **A 就是操作数**，直接参与操作码指定的运算。指令形式变成如下：

操作码	#	A

其中#是**立即寻址特征**，A 是立即数，可正可负（补码表示）。在**指令执行阶段不访存**，A 的位数限制了**立即数的范围**。

2.直接寻址

EA=A，有效地址由形式地址直接给出。如图 3.1 所示。

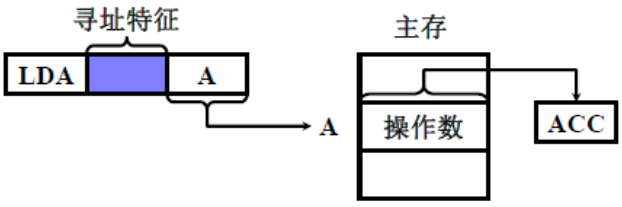

图 3.1 直接寻址方式

执行阶段访问一次存储器（就可以直接获取到操作数地址），A 的尾数决定了该指令操作数的**寻址范围**。操作数的地址不易修改（必须修改 A）。

3.隐含寻址

操作数地址隐含在操作码中。

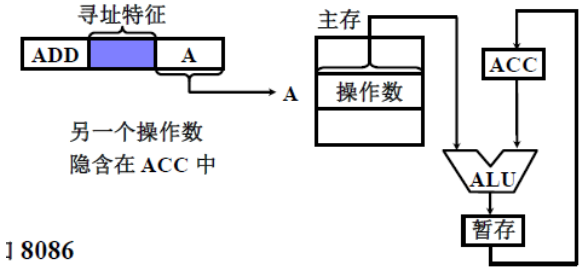

图 3.2 隐含寻址

如 8086，

MUL 指令	被乘数隐含在 AX（16 位）或 AL（8 位）中
MOVS 指令	源操作数的地址隐含在 SI 中，目的操作数的地址隐含在 DI 中。

指令字中少了一个地址字段，可缩短指令字长。

4.间接寻址

EA=(A)，有效地址由形式地址间接提供。如图 3.3 所示。

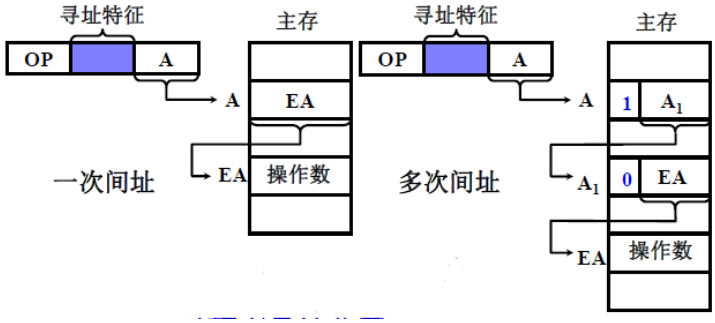

图 3.3 间接寻址

对于一次间接寻址来说，它执行指令阶段 2 次访存，可扩大寻址范围，便于编制程序。对于多次间接寻址来说，其不同是需要多次访存。

5.寄存器寻址

EA=Ri，有效地址即为寄存器编号。如图 3.4 所示。它的特点是：

①执行阶段不访存，只访问寄存器，执行速度快；

②寄存器个数有限，可缩短指令字长。

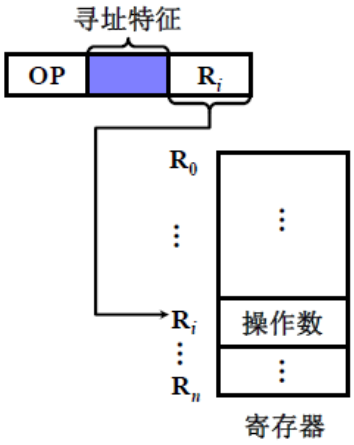

图 3.4 寄存器寻址

6.寄存器间接寻址

EA=(Ri)，有效地址存在寄存器中。如图 3.5 所示。

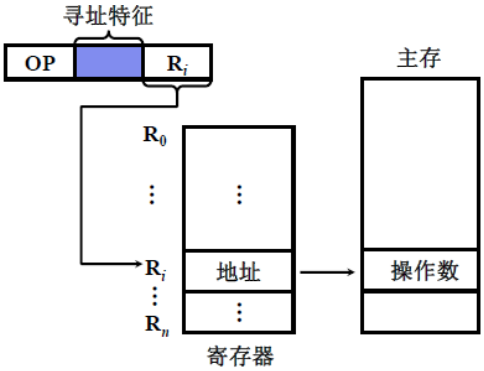

图 3.5 寄存器间接寻址

它有以下特点：

①有效地址在寄存器中，操作数在存储器中，执行阶段访存；

②便于编制循环程序。

7.基址寻址

（1）采用专用寄存器作基址寄存器

EA=(BR)+A，BR 为基址寄存器。如图 3.6 所示。

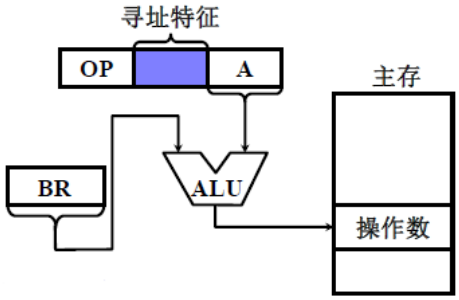

图 3.6 专用寄存器作基址寄存器

它有以下特点：

①可扩大寻址范围；

②有利于多道程序；

③BR 内容由操作系统或管理程序确定；

④在程序的执行过程中 BR 内容不变，形式地址 A 可变。

（2）采用通用寄存器作基址寄存器

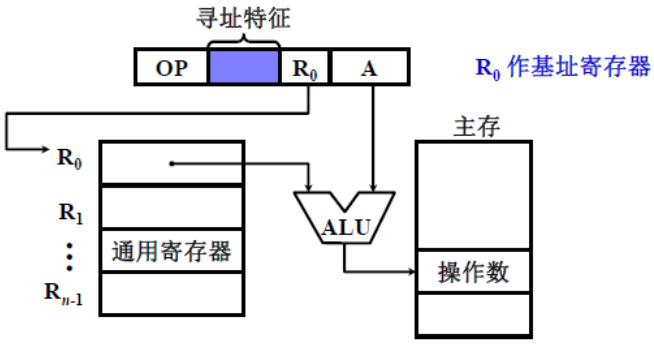

图 3.7 通用寄存器作基址寄存器

它有以下特点：

①由用户指定哪个通用寄存器作基址寄存器；

②基址寄存器的内容由操作系统确定；

③在程序的执行过程中 R0 内容不变，形式地址 A 可变。

8.变址寻址

EA=(IX)+A

IX 为变址寄存器（专用），通用寄存器也可以作为变址寄存器。

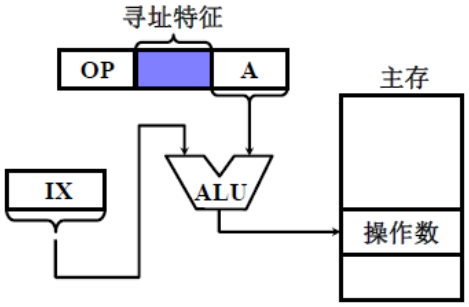

图 3.8 变址寻址

它有以下特点：

①可扩大寻址范围；

②IX 的内容由用户给定；

③在程序的执行过程中 IX 内容可变，形式地址 A 不变；

④便于处理数组问题。

例：设数据块首地址为 D，求 N 个数的平均值。

解答过程如图 3.9 所示。

图 3.9 例的图解

9.相对寻址

EA=(PC)+A

A 是相对于当前指令的位移量（可正可负，以补码形式保存）。如图 3.10 所示。

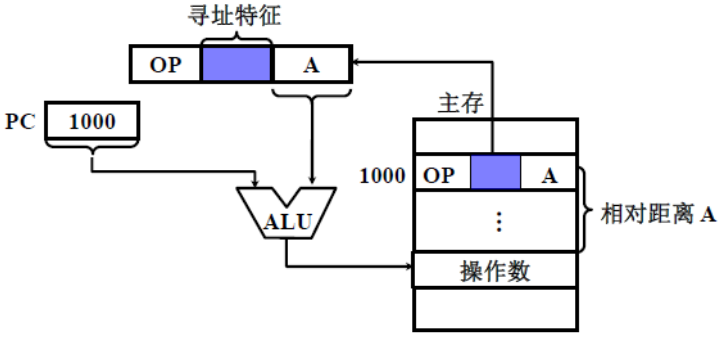

图 3.10 相对寻址

它有以下特点：

①A 的位数决定操作数的寻址范围；

②程序浮动（程序在内存当中位置变化）；

③广泛用于转移指令。

上例用相对寻址的实现，如图 3.11 所示。

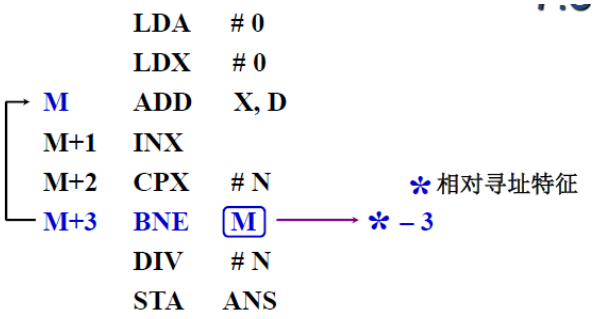

图 3.11 上例用相对寻址实现

我们注意 M 随程序所在存储空间的位置不同而不同，但是指令 BNE *-3 与指令 ADD X,D 相对位移量不变（*为相对寻址特征）。指令 BNE *-3 操作数的有效地址为 EA=(M+3)-3=M。

（1）按字节寻址的相对寻址举例

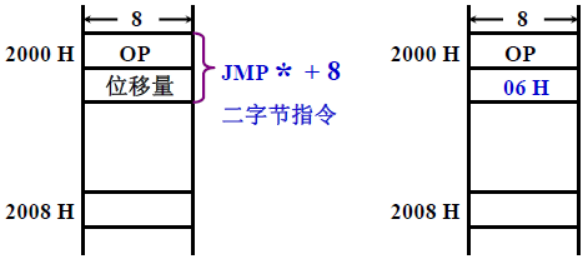

图 3.11 按字节寻址的相对寻址举例

设当前指令地址为 PC=2000H，转以后的目的地址为 2008H。因为取出 JMP*+8 后，PC 的地址已经跳转到 2002H 了，所以 JMP*+8 指令的第二字节为 2008H-2002H=06H。

10.堆栈寻址

（1）堆栈的特点

堆栈包括硬堆栈（多个寄存器）和软堆栈（指定的存储空间）。

先进后出（一个出入口）。栈顶地址由 SP 指出。进栈（SP）-1→SP，出栈（SP）+1→SP。

一般来说，栈底地址最高，栈顶地址最小，所以进栈才会-1，出栈才会+1。

（2）堆栈寻址举例

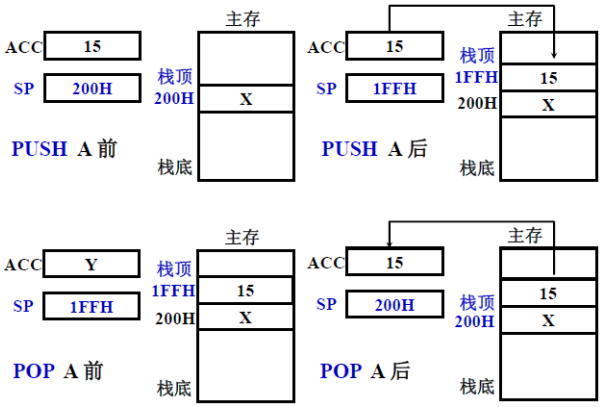

图 3.12 堆栈寻址举例

假如说要 PUSH A，在执行 PUSH 之前，栈顶是 SP=200H，保存着 X，ACC 为 15。PUSH 以后，先将 SP-1=1FFH，然后把 ACC 中的 15 压入到栈顶（1FFH）。

假如说要 POP A。POP A 之前，SP=1FFH，ACC 中保存的为 Y。先把 15 送入到 ACC，然后栈顶下压变为 200H。

（3）SP 修改与主存编址方法有关

①按**字**编址：进栈：(SP)-1→SP；出栈：(SP)+1→SP；

②按**字节**编址：

若存储字长 16 位，进栈：(SP)-2→SP；出栈：(SP)+2→SP；

若存储字长 32 位，进栈：(SP)-4→SP；出栈：(SP)+4→SP。

7.4 指令格式举例

一、设计指令格式时应考虑的各种因素

1.指令系统的**兼容性**。

2.其它因素：

操作类型	包括指令个数及操作的难易程度。
数据类型	确定哪些数据类型可参与操作。
指令格式	指令字长是否固定。操作码位数、是否采用扩展操作码技术，地址码位数、地址个数、寻址方式类型。
寻址方式	指令寻址、操作数寻址。
寄存器个数	寄存器的多少直接影响指令的执行时间。

二、指令格式举例

1.IBM 360

如图 4.1 所示。

图 4.1 IBM360 的寻址方式

2. Intel 8086

（1）指令字长：1~6 个字节

INC AX	1 个字节
MOV WORD PTR[0304], 0138H	6 字节

（2）地址格式

地址类型	地址举例	长度
零地址	NOP	1 字节
一地址	CALL 段间调用	5 字节
	CALL 段内调用	3 字节
二地址	ADD AX, BX	2 字节，寄存器-寄存器
	ADD AX, 3048H	3 字节，寄存器-立即数
	ADD AX, [3048H]	4 字节，寄存器・存储器

7.5 RISC 技术

一、关于 RISC

RISC（Reduced Instruction Set Computer）简单指令集计算机

CISC（Complex Instruction Set Computer）复杂指令集计算机

二-八规律：80%的程序使用 20%的指令。

二、RISC 的主要特征

1.选用使用频度较高的一些**简单指令**，复杂指令的功能由简单指令来组合；

2.指令**长度固定、指令格式种类少、寻址方式少**；

3.只有 **LOAD/STORE 指**令访存；

4.CPU 中有**多个**通用**寄存器**；

5.采用**流水技术，一个时钟周期**内完成一条指令；

6.采用**组合逻辑**实现控制器。

三、CISC 的主要特征

1.系统指令**复杂庞大**，各种指令使用频度相差大；

2.指令**长度不固定、指令格式种类多、寻址方式多**；

3.**访存**指令**不受限制**；

4.CPU 有**专用寄存器**；

5.大多数指令需要**多个时钟周期**执行完毕；

6、采用**微程序**控制器。

四、RISC 和 CISC 的比较

1.RISC 更能**充分利用 VLSI 芯片**的面积；

2.RISC 更能**提高计算机运行速度**，指**令数、指令格式、寻址方式**少，通用寄存器多，采用**组合逻辑**，便于实现**指令流水**；

3.RISC **便于设计，可降低成本，提高可靠性**；

4.RISC **不易**实现**指令系统兼容**。