泊松教育

指令格式

指令(机器指令)是指示计算机执行某种操作的命令。一台计算机的所有指令的集合构成该机的指令系统，也称指令集。指令系统是计算机的主要属性，位于硬件和软件的交界面上。

1:指令的基本格式

一条指令就是机器语言的一个语句，它是一组有意义的二进制代码。一条指令通常包括操作码字段和地址码字段两部分:

操作码字段	地址码字段

其中，操作码指出指令中该指令应该执行什么性质的操作以及具有何种功能。操作码是识别指令、了解指令功能及区分操作数地址内容的组成和使用方法等的关键信息。例如，指出是算术加运算还是算术减运算，是程序转移还是返回操作。

地址码给出被操作的信息(指令或数据)的地址，包括参加运算的一个或多个操作数所在的地址、运算结果的保存地址、程序的转移地址、被调用的子程序的入口地址等。

指令的长度是指一条指令中所包含的二进制代码的位数。指令字长取决于操作码的长度、操作数地址码的长度和操作数地址的个数。指令长度与机器字长没有固定的关系，它可以等于机器字长，也可以大于或小于机器字长。通常，把指令长度等于机器字长的指令称为单字长指令，指令长度等于半个机器字长的指令称为半字长指令；指令长度等于两个机器字长的指令称为双字长指令。

在一个指令系统中，若所有指令的长度都是相等的，则称为定长指令字结构。定字长指令的执行速度快，控制简单。若各种指令的长度随指令功能而异，则称为变长指令字结构。然而，因为主存一般是按字节编址的，所以指令字长多为字节的

泊松教育

整数倍。 ☆ 指令长度是字节（字节）的整倍数。

根据指令中操作数地址码的数目的不同，可将指令分成以下几种格式。

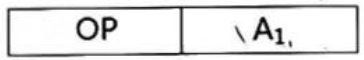
1)零地址指令

只给出操作码 OP，没有显式地址。这种指令有两种可能:

(1)不需要操作数的指令，如空操作指令、停机指令、关中断指令等。

(2)零地址的运算类指令仅用在堆栈计算机中。通常参与运算的两个操作数隐含地从栈顶和次栈顶弹出，送到运算器进行运算，运算结果再隐含地压入堆栈。

2)一地址指令



这种指令也有两种常见的形态，要根据操作码的含义确定究竟是哪一种。

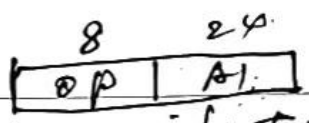
(1)只有目的操作数的单操作数指令，按 A_1 地址读取操作数，进行 OP 操作后，结果存回原地址。

指令含义:$OP(A_1) \rightarrow A_1$

如操作码含义是加 1、减 1、求反、求补等。

(2)隐含约定目的地址的双操作数指令，按指令地址 A_1 可读取源操作数，指令可隐含约定另一个操作数由 ACC(累加器)提供，运算结果也将存放在 ACC 中。

泊松教育 24位机器字长，指令字长



32位 寻址方向划：16M
↳直接寻址范围：0 ~ $2^{24}-1$

指令含义:(ACC)OP(A_1)→ACC

★ 若指令字长为 32 位，操作码占 8 位，1 个地址码字段占 24 位，则指令操作数的直接寻址范围为 2^{24}=16M。

3)二地址指令

8 12 12

二地址	OP	A_1	A_2

指令含义:(A_1)OP(A_2)→A_1

寻址：2^{12}（4K） 用2^{12}（4K）

对于常用的算术和逻辑运算指令，往往要求使用两个操作数，需分别给出目的操作数和源操作数的地址，其中目的操作数地址还用于保存本次的运算结果。

若指令字长为 32 位，操作码占 8 位，两个地址码字段各占 12 位，则指令操作数的直接寻址范围为 2^{12}=4K。

4)三地址指令

8 8 8 8

三地址	OP	A_1	A_2	A_3（结果）

指令含义:(A_1)OP(A_2)→A_3

若指令字长为 32 位，操作码占 8 位，3 个地址码字段各占 8 位，则指令操作数的直接寻址范围为 2^8=256。若地址字段均为主存地址，则完成一条三地址需要 4 次访问存储器（取指令 1 次，取两个操作数 2 次，存放结果 1 次）。

5)四地址指令

四地址	OP	A_1	A_2	A_3（结果）	A_4（下址）

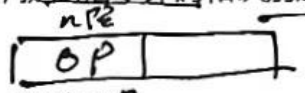
泊松教育

指令含义:$(A_1)OP(A_2)\rightarrow A_3$ A_4=下一条将要执行指令的地址。

若指令字长为 32 位，操作码占 8 位，4 个地址码字段各占 6 位，则指令操作数的直接寻址范围为 $2^6=64$。

2:定长操作码指令格式

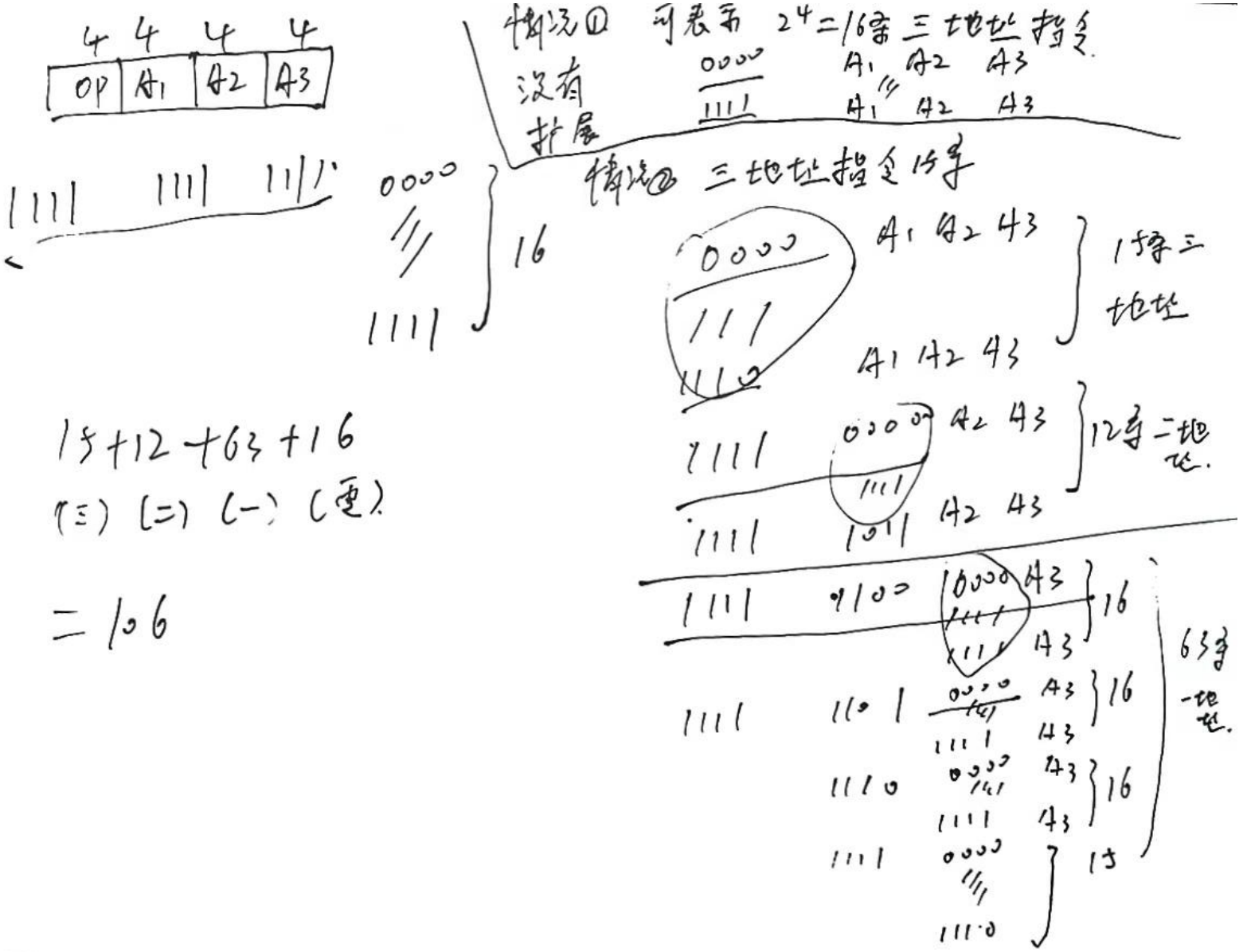
定长操作码指令在指令字的最高位部分分配固定的若干位(定长)表示操作码。一般 n 位操作码字段的指令系统最大能够表示 2" 条指令。定长操作码对于简化计算机硬件设计，提高指令译码和识别速度很有利。当计算机字长为 32 位或更长时，这是常规用法。



3:扩展操作码指令格式

为了在指令字长有限的前提下仍保持比较丰富的指令种类，可采取可变长度操作码，即全部指令的操作码字段的位数不固定，且分散地放在指令字的不同位置上。显然，这将增加指令译码和分析的难度，使控制器的设计复杂化。

最常见的变长操作码方法是扩展操作码，它使操作码的长度随地址码的减少而增加，不同地址数的指令可具有不同长度的操作码，从而在满足需要的前提下，有效地缩短指令字长。图 1 所示即为一种扩展操作码的安排方式。


4 4 4 4
OP A1 A2 A3
情况① 可表示 2^4=16条三地址指令
没有扩展
0000 A1 A2 A3
1111 A1 A2 A3
情况② 三地址指令15条
1111 1111 1111
0000
1111
16
15+12+63+16
(三) (二) (一) (更)
=106
0000 A1 A2 A3
1111
1条三地址
1111 0000 A2 A3
1111
12条二地址
1111 1011 A2 A3
1111 1100 0000 A3
1111
16
1111 1101 0000 A3
1111 A3
16
1111 1110 0000 A3
1111 A3
16
1111 0000 A3
1111
1111 0000
1110
1条
63条一地址

泊松教育

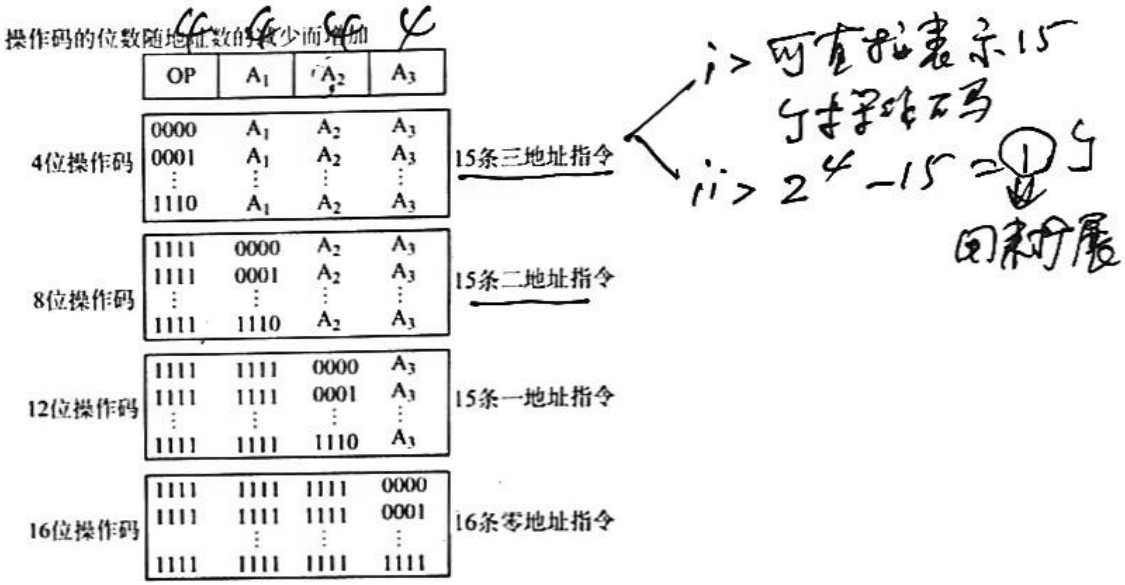


图 1 扩展操作码技术

在图 1 中，指令字长为 16 位，其中 4 位为基本操作码字段 OP，另有 3 个 4 位长的地址字段 A1、A2 和 A3。4 位基本操作码若全部用于三地址指令，则有 16 条。图 1 中所示的三地址指令为 15 条，1111 留作扩展操作码之用;二地址指令为 15 条，1111 1111 留作扩展操作码之用;一地址指令为 15 条，1111 1111 1111 留作扩展操作码之用;零地址指令为 16 条。

除这种安排外，还有其他多种扩展方法，如形成 15 条三地址指令、12 条二地址指令、63 条一地址指令和 16 条零地址指令，共 106 条指令。

在设计扩展操作码指令格式时，必须注意以下两点:

泊松教育

(1)不允许短码是长码的前缀，即短操作码不能与长操作码的前面部分的代码相同。

(2)各指令的操作码一定不能重复。

通常情况下，对使用频率较高的指令分配较短的操作码，对使用频率较低的指令分配较长的操作码，从而尽可能减少指令译码和分析的时间。

4:指令的操作类型

设计指令系统时必须考虑应提供哪些操作类型，指令操作类型按功能可分为以下几种。

1)数据传送

传送指令通常有寄存器之间的传送(MOV)、从内存单元读取数据到 CPU 寄存器(LOAD)、从 CPU 寄存器写数据到内存单元(STORE)等。

2)算术和逻辑运算

这类指令主要有加(ADD)、减(SUB)、比较(CMP)、乘(MUL)、除(DIV)、加 1(INC)、减 1(DEC)、与(AND)、或(OR)、取反(NOT)、异或(XOR)等。

3)移位操作

移位指令主要有算法移位、逻辑移位、循环移位等。

4)转移操作

泊松教育

转移指令主要有无条件转移(JMP)、条件转移(BRANCH)、调用(CALL)、返回(RET)、陷阱(TRAP)等。无条件转移指令在任何情况下都执行转移操作，而条件转移指令仅在特定条件满足时才执行转移操作，转移条件一般是某个标志位的值，或几个标志位的组合。

调用指令和转移指令的区别:执行调用指令时必须保存下一条指令的地址(返回地址)，当子程序执行结束时，根据返回地址返回到主程序继续执行;而转移指令则不返回执行。

5)输入输出操作

这类指令用于完成 CPU 与外部设备交换数据或传送控制命令及状态信息。

16 < 20 < 32

2^4 2^5

2017>. 计算机系统按字节编址，指令字长固定，其中三地址指令 20 条，二地址指令 107 条，每个地址字段 6 位，则指令字长至少是 ______

24 位. 26 位. (X) 28 位. (X) 32 位.

按字节编址.

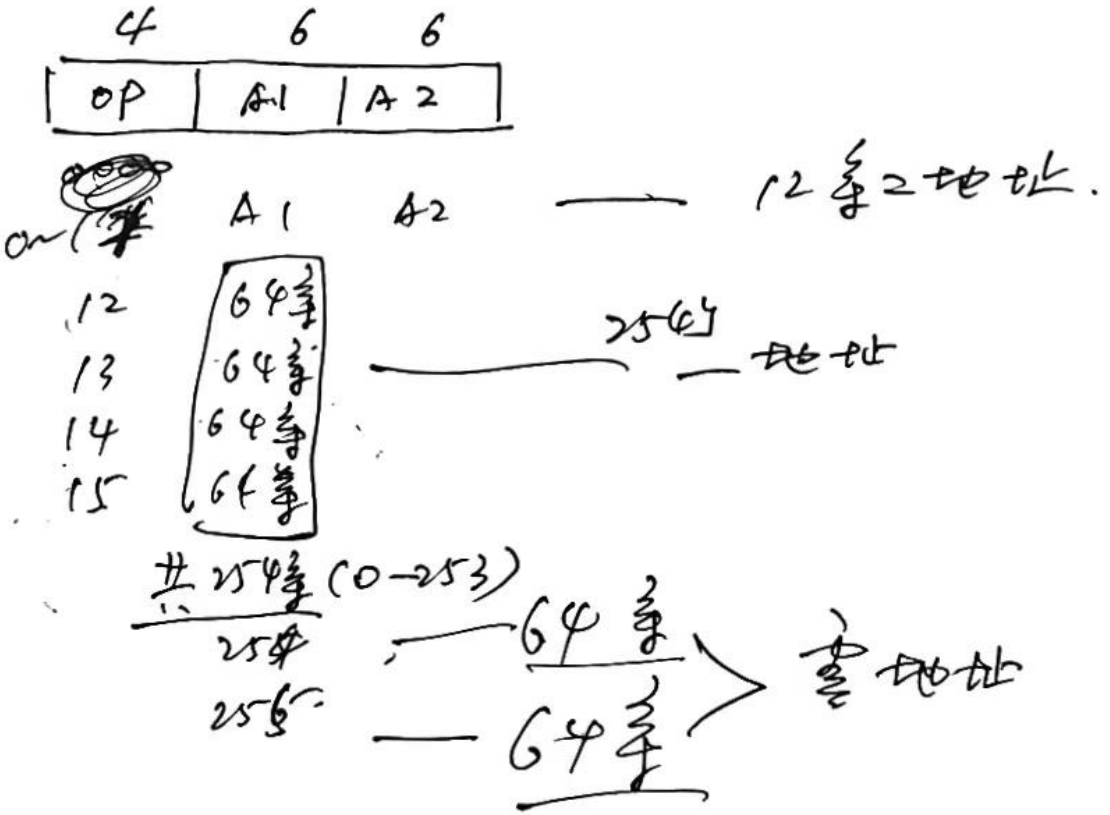
i> 直接排除 B、C —— 原因：必须字长 8 的倍数.

ii>.

5	6	6	6
OP	A1	A2	A3
00000	A1	A2	A3
10011			

【2022 统考真题】设计某指令系统时，假设采用16位定长指令字格式，操作码使用扩展编码方式，地址码为6位，包含零地址、一地址和二地址3种格式的指令。若二地址指令有12条，一地址指令有254条，则零地址指令的条数最多为( )。

A.0 B.2 C.64 D.128



ISA —— 指令集体系结构

- 1) 软、硬件之间接口
- 2) 指令格式：指令长度、格式
  - 操作数存放方式
  - 可访问的寄存器数、字数、编号
  - 存储器寻址方式、地址空间大小
  - 指令执行的控制方式

泊松教育

指令的寻址方式

寻址方式是指寻找指令或操作数有效地址的方式，即确定本条指令的数据地址及下一条待执行指令的地址的方法。寻址方式分为指令寻址和数据寻址两大类。

指令中的地址码字段并不代表操作数的真实地址，这种地址称为形式地址(A)。形式地址结合寻址方式，可以计算出操作数在存储器中的真实地址，这种地址称为有效地址(EA)。

注意，(A)表示地址为 A 的数值，A 既可以是寄存器编号，也可以是内存地址。对应的(A)就是寄存器中的数值，或相应内存单元的数值。例如，EA=(A)意思是有效地址是地址 A 中的数值。

1:指令寻址和数据寻址

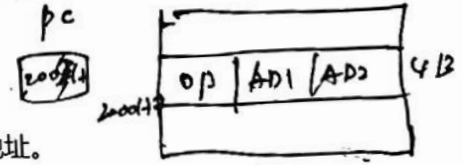
寻址方式分为指令寻址和数据寻址两大类。寻找下一条将要执行的指令地址称为指令寻址;寻找本条指令的数据地址称为数据寻址。

找指令寻址 —— 找下一条指令.

找数据寻址 —— 找当前指令操作数.

1)指令寻址

指令寻址方式有两种:一种是顺序寻址方式，另一种是跳跃寻址方式。



(1)顺序寻址

通过程序计数器 PC 加 1(1 个指令字长)，自动形成下一条指令的地址。

(2)跳跃寻址

泊松教育

通过转移类指令实现。所谓跳跃，是指下条指令的地址不由程序计数器 PC 自动给出，而由本条指令给出下条指令地址的计算方式。而是否跳跃可能受到状态寄存器和操作数的控制，跳跃的地址分为绝对地址(由标记符直接得到)和相对地址(相对于当前指令地址的偏移量)，跳跃的结果是当前指令修改 PC 值，所以下一条指令仍然通过 PC 给出。

2)数据寻址

数据寻址是指如何在指令中表示一个操作数的地址，如何用这种表示得到操作数或怎样计算出操作数的地址。

数据寻址的方式较多，为区别各种方式，通常在指令字中设一个字段，用来指明属于哪种寻址方式，由此可得指令的格式如下所示:

操作码	寻址特征	形式地址 A

2:常见的数据寻址方式

不同寻址方式

1)隐含寻址

这种类型的指令不明显地给出操作数的地址，而在指令中隐含操作数的地址。例如，单地址的指令格式就不明显地在地址字段中指出第二操作数的地址，而规定累加器(ACC)作为第二操作数地址，指令格式明显指出的仅是第一操作数的地址。因此，累加器(ACC)对单地址指令格式来说是隐含寻址，如图 1 所示。

泊松教育

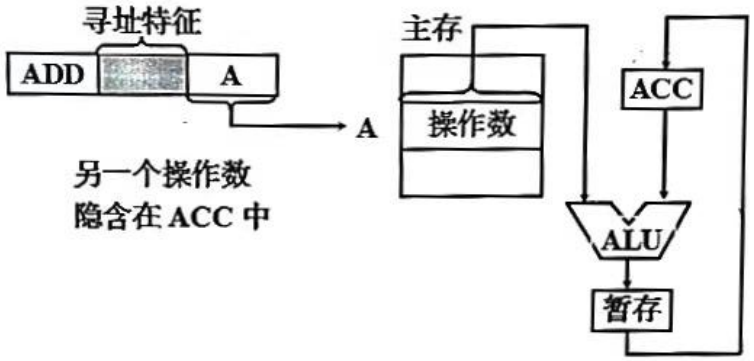


图 1　隐含寻址

隐含寻址的优点是有利于缩短指令字长;缺点是需增加存储操作数或隐含地址的硬件。

2)立即(数)寻址

这种类型的指令的地址字段指出的不是操作数的地址，而是操作数本身，又称立即数，采用补码表示。图 2 所示为立即寻址示意图，图中#表示立即寻址特征，A 就是操作数。

立即寻址的优点是指令在执行阶段不访问主存，指令执行时间最短;缺点是 A 的位数限制了立即数的范围。

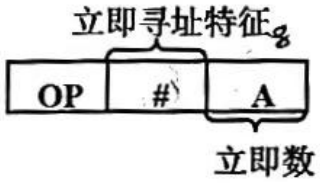


图 2　立即寻址

泊松教育

3)直接寻址

指令字中的形式地址 A 是操作数的真实地址 EA，即 EA=A，如图 3 所示。

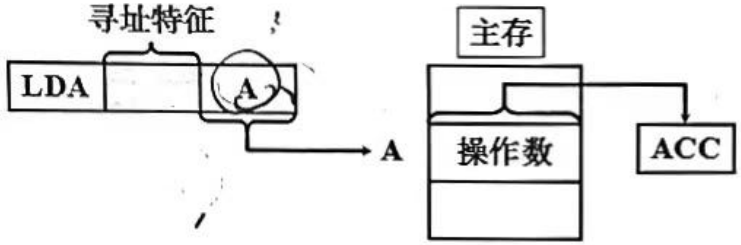


图 3　直接寻址

直接寻址的优点是简单，指令在执行阶段仅访问一次主存，不需要专门计算操作数的地址;缺点是 A 的位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。

4)间接寻址

间接寻址是相对于直接寻址而言的，指令的地址字段给出的形式地址不是操作数的真正地址，而是操作数有效地址所在的存储单元的地址，也就是操作数地址的地址，即 EA=(A)，如图 4 所示。间接寻址可以是一次间接寻址，还可以是多次间接寻址。

泊松教育

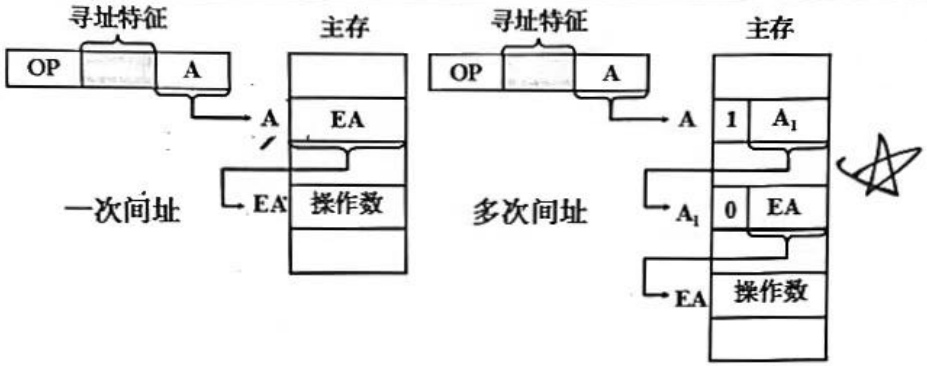


图 4　间接寻址

在图 4 中，主存字第一位为 1 时，表示取出的仍不是操作数的地址，即多次间址;主存字第一位为 0 时，表示取得的是操作数的地址。

间接寻址的优点是可扩大寻址范围(有效地址 EA 的位数大于形式地址 A 的位数)，便于编制程序(用间接寻址可方便地完成子程序返回);缺点是指令在执行阶段要多次访存(一次间接寻址需两次访存，多次间接寻址需根据存储字的最高位确定访存次数)。由于访问速度过慢，这种寻址方式并不常用。一般问到扩大寻址范围时，通常指的是寄存器间接寻址。

5)寄存器寻址

寄存器寻址是指在指令字中直接给出操作数所在的寄存器编号，即 EA=R_i，其操作数在由 R_i所指的寄存器内，如图 4.6 所示。寄存器寻址的优点是指令在执行阶段不访问主存，只访问寄存器，因寄存器数量较少，对应地址码长度较小，使得指令字短且因不用访存，所以执行速度快，支持向量/矩阵运算;缺点是寄存器价格昂贵，计算机中的寄存器个数有限。

泊松教育

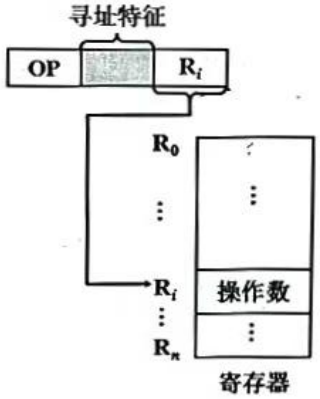
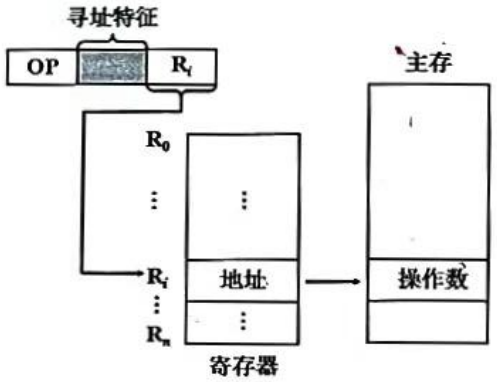


图 5　寄存器寻址

先寄存器
再主存

6)寄存器间接寻址

寄存器间接寻址是指在寄存器 R_i，中给出的不是一个操作数，而是操作数所在主存单元的地址，即 $EA=(R_i)$，如图 6 所示。



寄存器主存取操作数在主存中的地址.

一次访存

泊松教育

图 6　寄存器间接寻址

寄存器间接寻址的特点是，与一般间接寻址相比速度更快，但指令的执行阶段需要访问主存(因为操作数在主存中)。

7)相对寻址

相对寻址是把 PC 的内容加上指令格式中的形式地址 A 而形成操作数的有效地址，即 EA = (PC) + A 其中 A 是相对于当前指令地址的位移量，可正可负，补码表示，如图 7 所示。

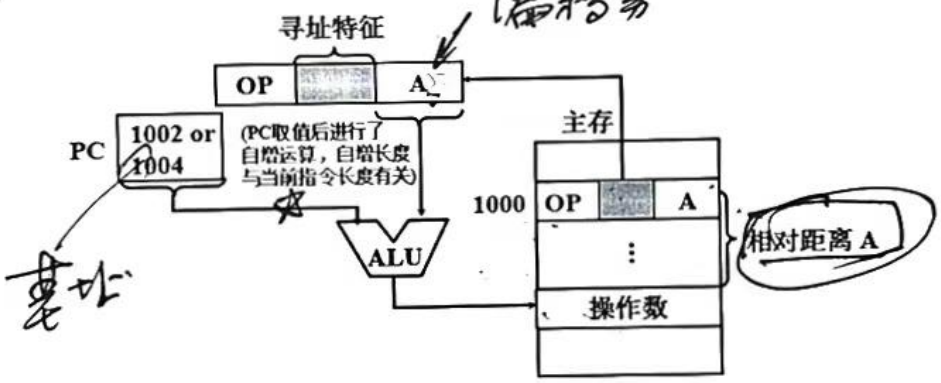


图 7　相对寻址

在图 7 中，A 的位数决定操作数的寻址范围。相对寻址的优点是操作数的地址不是固定的，它随 PC 值的变化而变化，且与指令地址之间总是相差一个固定值，因此便于程序浮动。相对寻址广泛应用于转移指令。

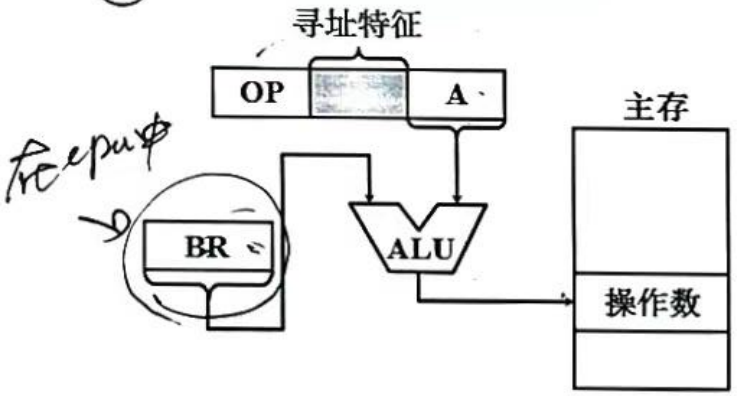
注意，对于转移指令 JMPA，当 CPU 从存储器中取出一字节时，会自动执行(PC)+1→PC。若转移指令的地址为 X，且占 2B，在取出该指令后，PC 的值会增 2，即(PC) =X+2，这样在执行完该指令后，会自动跳转到 X+2+A 的地址继续

泊松教育

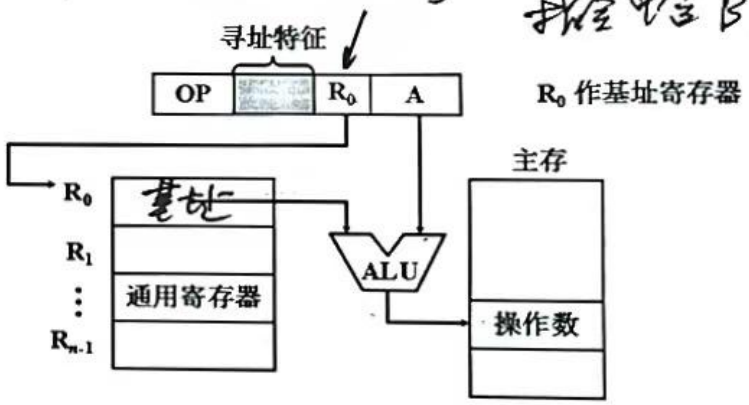
执行。

8)基址寻址

基址寻址是指将 CPU 中基址寄存器(BR)的内容加上指令格式中的形式地址 A 而形成操作数的有效地址，即 EA = (BR) + A。其中基址寄存器既可采用专用寄存器，又可采用通用寄存器，如图 8 所示。



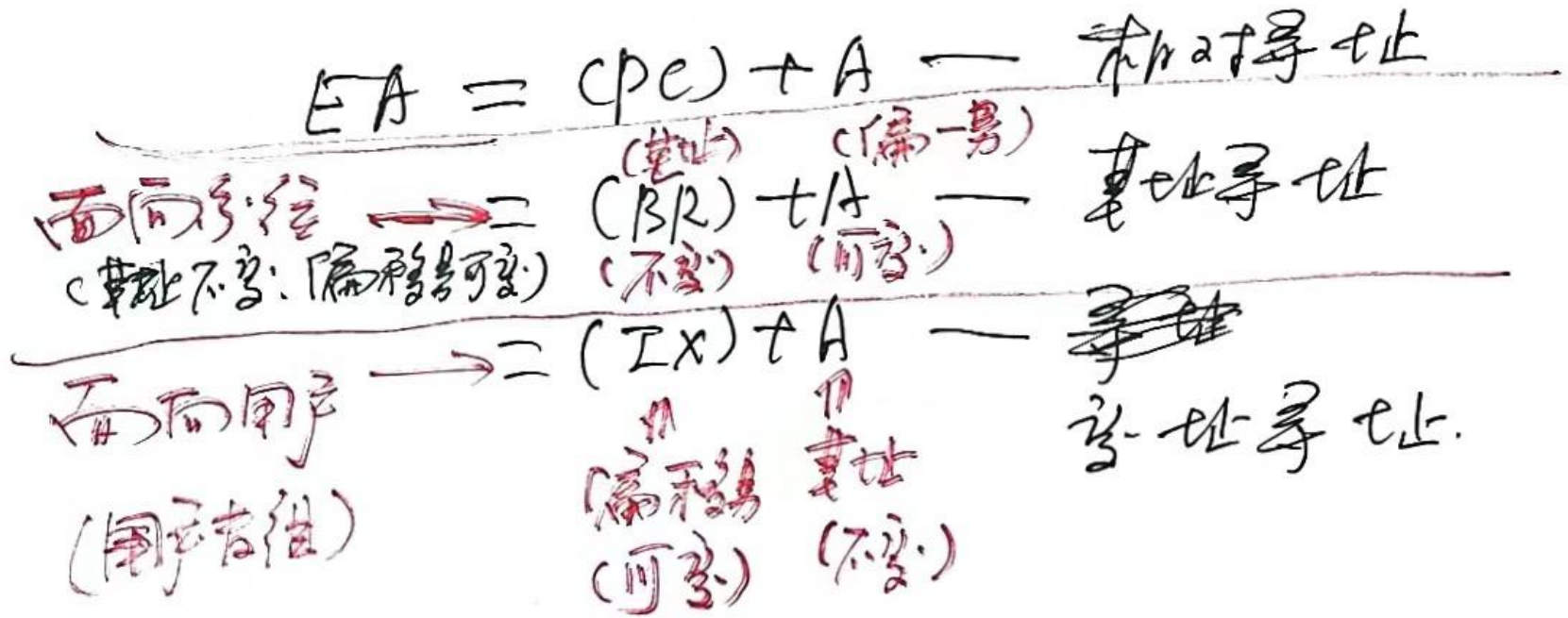
(a)采用专用寄存器 BR 作为基址寄存器



(b)采用通用寄存器 R_0 作为基址寄存器

图 8　基址寻址

基址寄存器是面向操作系统的，其内容由操作系统或管理程序确定，主要用于解决程序逻辑空间与存储器物理空间的无关性。在程序执行过程中，基址寄存器的内容不变(作为基地址)，形式地址可变(作为偏移量)。采用通用寄存器作为基址寄存器时，可由用户决定哪个寄存器作为基址寄存器，但其内容仍由操作系统确定。


EA = (PC) + A —— 相对寻址
(基址) (偏移量)
面向对象 ⟹ = (BR) + IA —— 基址寻址
(基址不变、偏移量可变) (不变) (可变)
= (IX) + A —— 变址寻址
面向用户
(用户自行)
(偏移量 可变)
(基址 不变)

泊松教育

基址寻址的优点是可扩大寻址范围(基址寄存器的位数大于形式地址 A 的位数);用户不必考虑自己的程序存于主存的哪个空间区域，因此有利于多道程序设计，并可用于编制浮动程序，但偏移量(形式地址 A)的位数较短。

9)变址寻址

变址寻址是指有效地址 EA 等于指令字中的形式地址 A 与变址寄存器 IX 的内容之和，即 EA= (IX) + A，其中 IX 为变址寄存器(专用)，也可用通用寄存器作为变址寄存器。图 4.10 所示为采用专用寄存器 IX 的变址寻址示意图。

变址寄存器是面向用户的，在程序执行过程中，变址寄存器的内容可由用户改变(作为偏移量)，形式地址 A 不变(作为基地址)。

变址寻址的优点是可扩大寻址范围(变址寄存器的位数大于形式地址 A 的位数);在数组处理过程中，可设定 A 为数组的首地址，不断改变变址寄存器 IX 的内容，便可很容易形成数组中任一数据的地址，特别适合编制循环程序。偏移量(变址寄存器 IX)的位数足以表示整个存储空间。

显然，变址寻址与基址寻址的有效地址形成过程极为相似。但从本质上讲，两者有较大区别。基址寻址面向系统，主要用于为多道程序或数据分配存储空间，因此基址寄存器的内容通常由操作系统或管理程序确定，在程序的执行过程中其值不可变，而指令字中的 A 是可变的;变址寻址立足于用户，主要用于处理数组问题，在变址寻址中，变址寄存器的内容由用户设定，在程序执行过程中其值可变，而指令字中的 A 是不可变的。

泊松教育

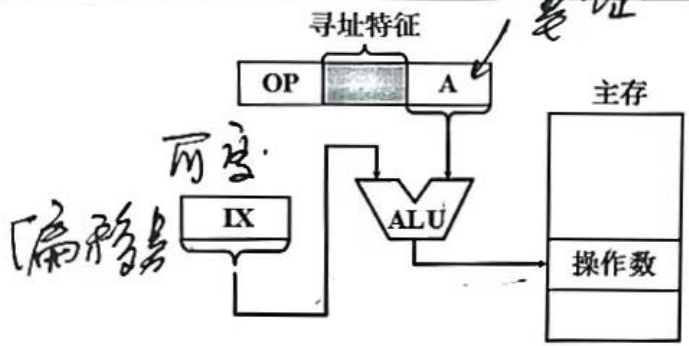


图 9　变址寻址

10)堆栈寻址

堆栈是存储器(或专用寄存器组)中一块特定的、按后进先出(LIFO)原则管理的存储区，该存储区中读/写单元的地址是用一个特定的寄存器给出的，该寄存器称为堆栈指针(SP)。堆栈可分为硬堆栈与软堆栈两种。

寄存器堆栈又称硬堆栈。寄存器堆栈的成本较高，不适合做大容量的堆栈;而从主存中划出一段区域来做堆栈是最合算且最常用的方法，这种堆栈称为软堆栈。

在采用堆栈结构的计算机系统中，大部分指令表面上都表现为无操作数指令的形式，因为操作数地址都隐含使用了 SP。

通常情况下，在读/写堆栈中的一个单元的前后都伴有自动完成对 SP 内容的增量或减量操作。

下面简单总结寻址方式、有效地址及访存次数(不含取本条指令的访存)，见表 1。

泊松教育

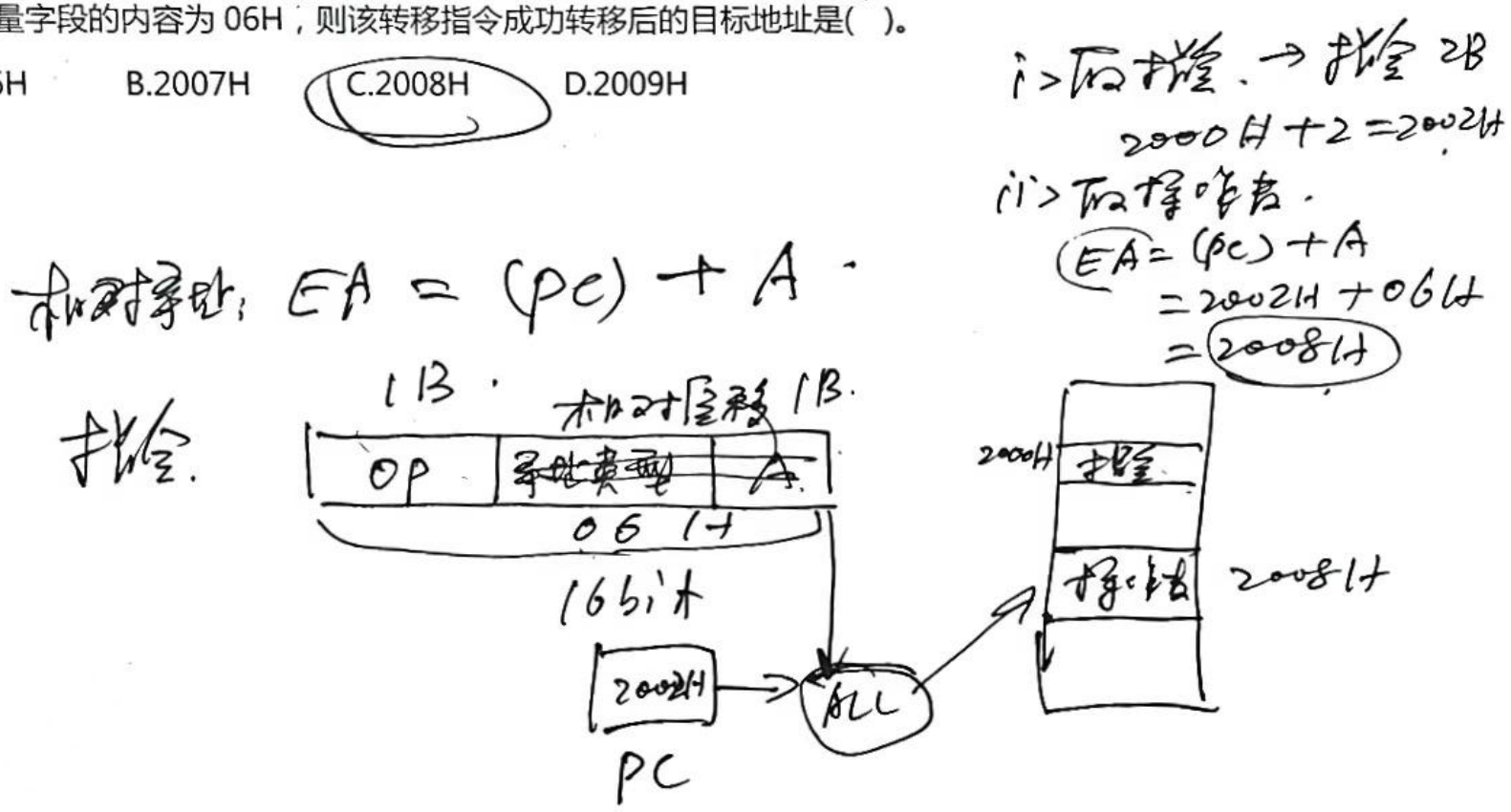
表1 寻址方式、有效地址和访问次数

寻址方式	有效地址	访问次数
隐含寻址	程序指定	0
立即寻址	A即是操作数	0
直接寻址	EA=A	1
一次间接寻址	EA=(A)	2
寄存器寻址	~~EA=R_i~~	0
寄存器间接一次寻址	EA=(R_i)	1
相对寻址	EA=(PC)+A	1
基址寻址	EA=(BR)+A	1
变址寻址	EA=(IX)+A	1

操作数=R_i

【2009 统考真题】某机器字长为 16 位，主存按字节编址，转移指令采用相对寻址，由 2 字节组成，第一字节为操作码字段，第二字节为相对位移量字段。假定取指令时，每取一字节 PC 自动加 1。若某转移指令所在主存地址为 2000H，相对位移量字段的内容为 06H，则该转移指令成功转移后的目标地址是(　)。

A.2006H　　B.2007H　　C.2008H　　D.2009H



【2013 统考真题】假设变址寄存器 R 的内容为 1000H，指令中的形式地址为 2000H；地址 1000H 中的内容为 2000H，地址 2000H 中的内容为 3000H，地址 3000H 中的内容为 4000H，则变址寻址方式下访问到的操作数是(　)。

A.1000H　　B.2000H　　C.3000H　　D.4000H