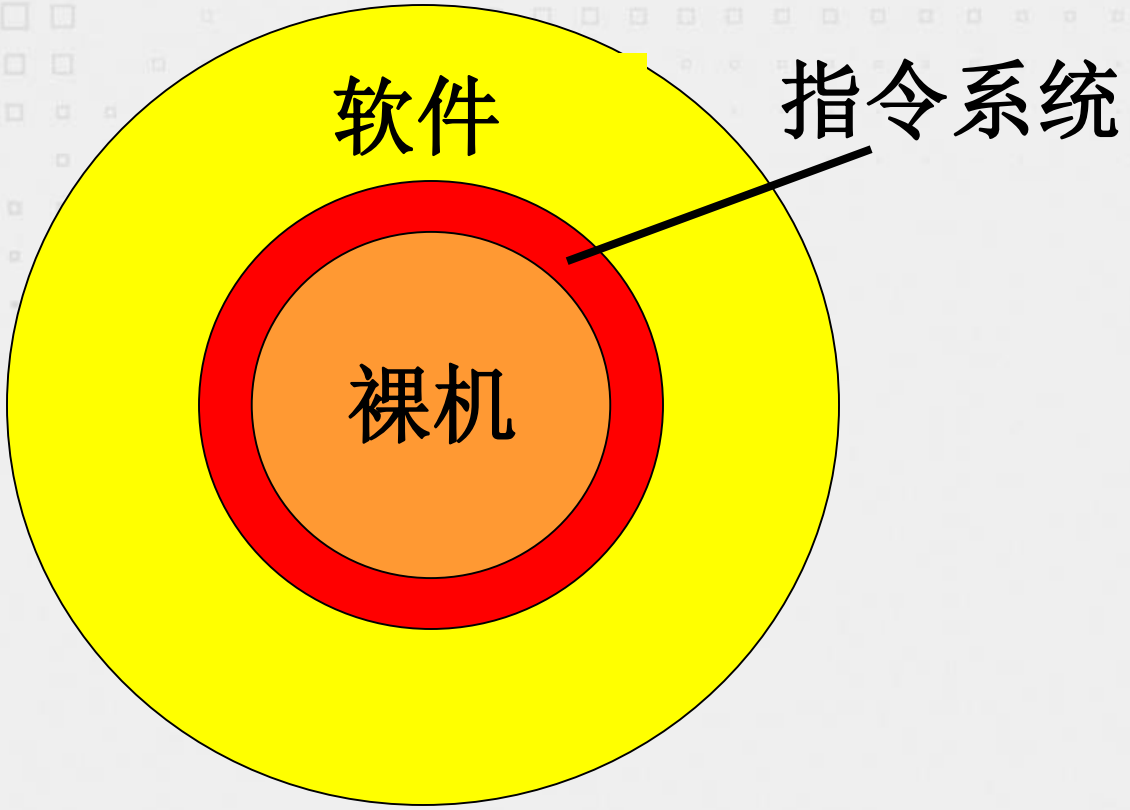
第三章 指令系统

前 言

指令和指令系统是计算机中最基本的概念。指令是指示计算机执行某些操作的命令，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。本章将讨论一般计算机的指令系统所涉及的基本问题。


软件
指令系统
裸机

目 录

3.1.1 机器指令的基本格式

- 操作码：指明操作的性质及功能。
- 地址码：指明操作数的地址，特殊情况下也可能直接给出操作数本身。

3.1.1 机器指令的基本格式

- 指令长度可以等于机器字长，也可以大于或小于机器字长。
- 在一个指令系统中，若所有指令的长度都是相等的，称为定长指令字结构；若各种指令的长度随指令功能而异，称为变长指令字结构。

3.1.2 地址码结构

- 一条双操作数指令除操作码之外，还应包含以下信息：
  - 第一操作数地址，用A1表示；
  - 第二操作数地址，用A2表示；
  - 操作结果存放地址，用A3表示；
  - 下条将要执行指令的地址，用A4表示。
- 这些信息可以在指令中明显的给出，称为显地址；也可以依照某种事先的约定，用隐含的方式给出，称为隐地址。

3.1.2 地址码结构

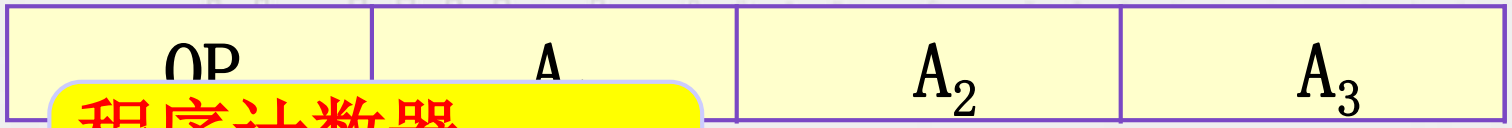
- 1.四地址指令

OP	A_1	A_2	A_3	A_4

  - $(A_1)OP(A_2)\rightarrow A_3$
  - A_4=下条将要执行指令的地址
  - 执行一条四地址指令需 4 次访问主存。

3.1.2 地址码结构

- 2.三地址指令

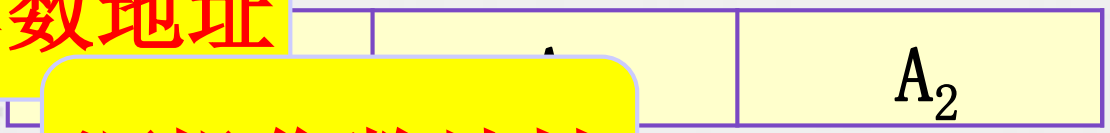


程序计数器：
存放当前指令地址

- (A_1) OP $(A_2)\rightarrow A_3$
- (PC)+1=下条将要执行指令的地址
- 执行一条三地址指令需 3 次访问主存。

3.12 地址码结构

- 3.二地址指令

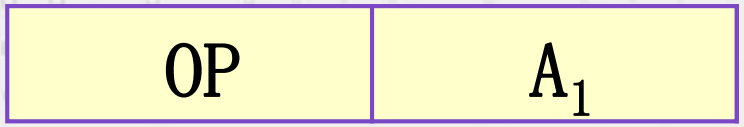


  - $(A_1)OP(A_2)\rightarrow A_1$
    - A1中原存内容在指令执行后被破坏。
  - (PC)+1=下条将要执行指令的地址
  - 执行一条二地址指令需 3 次访问主存。

3.1.2 地址码结构

- 4.一地址指令

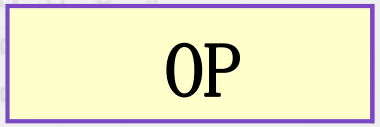
累加寄存器



  - $(A_{cc})OP(A_1)\rightarrow A_{cc}$
  - (PC)+1=下条将要执行指令的地址
  - 执行一条一地址指令需 1 次访问主存。

3.1.2 地址码结构

- 5.零地址指令



  - 操作数地址是隐含的。
  - 参加运算的操作数放在堆栈中，运算结果也放在堆栈中。有关堆栈的概念将在稍后讨论。

3.1.3 指令的操作码

- 指令操作码的编码可以分为规整型和非规整型两类：
  - 规整型（定长编码）
  - 非规整型（变长编码）

3.1.3 指令的操作码

- 1.规整型
  - 操作码字段的位数和位置是固定的。
  - 假定：指令系统共有m条指令，指令中操作码字段的位数为N位，则有如下关系式：

$$N \geq \log_2 m$$

  - 定长编码对于简化硬件设计，减少指令译码的时间是非常有利的，但存在着信息冗余。

3.1.3 指令的操作码

- 2.非规整型
  - 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。
  - 操作码字段的位数和位置不固定将增加指令译码和分析的难度，使控制器的设计复杂化。
  - 最常用的非规整型编码方式是扩展操作码法。让操作数地址个数多的指令（如：三地址指令）的操作码字段短些，操作数地址个数少的指令（如一或零地址指令）的操作码字段长些。

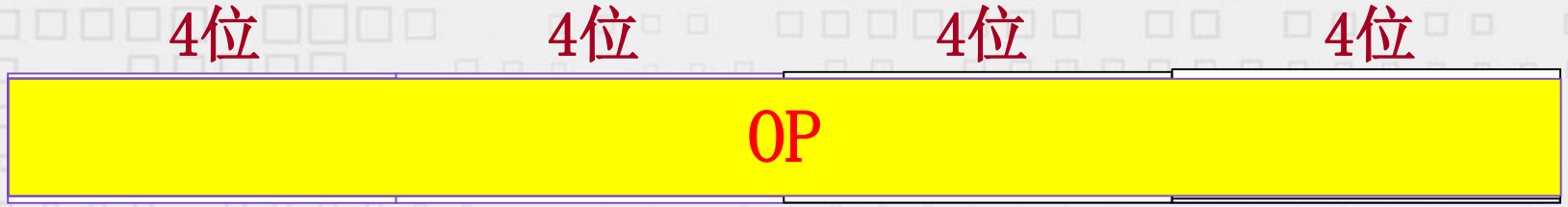
3.1.3 指令的操作码

- 例如：设某机的指令长度为16位，操作码字段为4位，有三个4位的地址码字段，其格式为：

4位	4位	4位	4位
OP	A_1	A_2	A_3

  - 如果按照定长编码的方法，4位操作码字段最多只能表示16条不同的三地址指令。

3.1.3 指令的操作码



扩展窗口

扩展窗口

扩展窗口

0000 XXXX YYYY ZZZZ
⋮
1110 XXXX YYYY ZZZZ
1111 0000 XXXX YYYY
⋮
1111 1110 XXXX YYYY
1111 1111 0000 XXXX
⋮
1111 1111 1110 XXXX
1111 1111 1111 0000
⋮
1111 1111 1111 1111

15条三地址指令

15条二地址指令

15条一地址指令

16条零地址指令

课堂习题

- 1. 一地址指令中另一个操作数存放在(累加器)。
- 2. 零地址指令的操作数存放在(堆栈)。
- 3. 下条指令的地址存放在(PC/程序计数器)。

课堂习题

- 4.某机器字长32位，指令单字长，指令系统中具有二地址指令、一地址指令和零地址指令各若干条，已知每个地址长12位，采用扩展操作码方式，问该指令系统中的二地址指令、一地址指令、零地址指令各最多能有多少条？

课堂习题

- 5.指令字长为12位，每个地址码为3位，采用扩展操作码的方式，设计4条三地址指令、16条二地址指令、64条一地址指令和16条零地址指令。
  - （1）给出一种操作码的扩展方案
  - （2）计算操作码的平均长度

课堂习题

- 6. 若某机为定长指令字结构，要求：三地址指令4条，二地址指令8条，单地址指令180条。设指令字长为12位.每个地址码长为3位。问能否以扩展操作码为其编码?如果其中单地址指令为254条呢?说明其理由。

答案：∵指令字长 12 位，每个地址码占 3 位;
∴三地址指令最多是 2^(12-3-3-3)=8 条，　现三地址指令需 4 条，
∴可有 4 条编码作为扩展码，
∴二地址指令最多为 4×2^3=32 条，
现要求二地址指令 8 条，∴可有 24 个编码作扩展码
∴单地址指令最多为 24×2^3=192 条

目 录

寻址技术

- 寻址，指的是寻找操作数的地址或下一条将要执行的指令地址。
- 寻址技术包括编址方式和寻址方式。

3.2.1 编址方式

- 1. 编址
  - 通常，指令中的地址码字段将指出操作数的来源和去向，操作数则存放在相应的存储设备中。
  - 在计算机中需要编址的设备主要有CPU中的通用寄存器、主存储器和输入输出设备等3种。

3.2.1 编址方式

- 2. 编址单位
  - (1)字编址
    - 编址单位=访问单位
    - 每个编址单位所包含的信息量（二进制位数）与读或写一次寄存器、主存所获得的信息量是相同的。
    - 早期的大多数机器都采用这种编址方式。

3.2.1 编址方式

- 2. 编址单位
  - (2)字节编址
    - 编址单位 < 访问单位
    - 通常主存的访问单位是编址单位的若干倍。
  - (3)位编址
    - 部分计算机系统采用位编址方式。

3.2.1 编址方式

- 3. 指令中地址码的位数
  - 指令格式中每个地址码的位数是与主存容量和最小寻址单位（即编址单位）有关联的。主存容量越大，所需的地址码位数就越长。

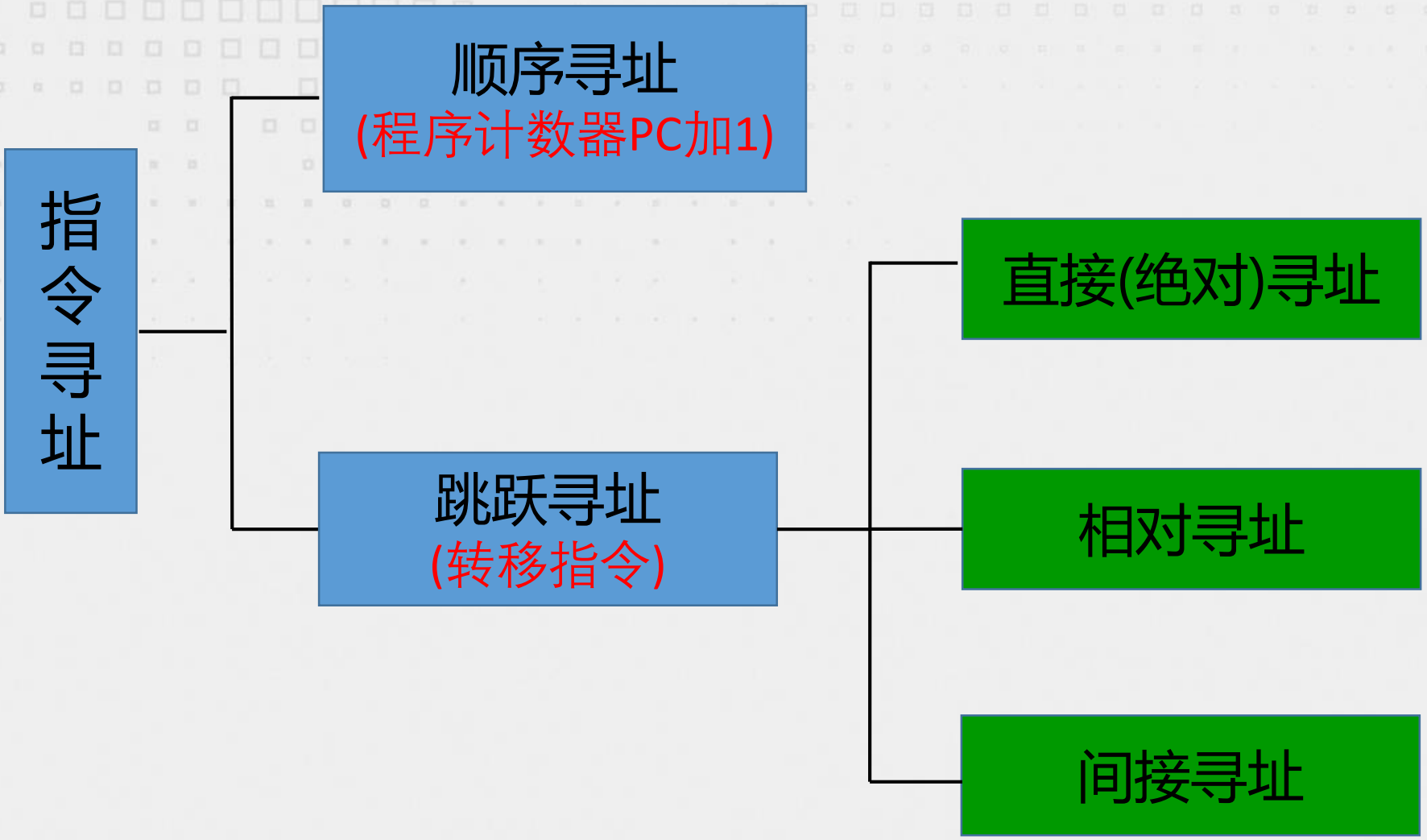
课堂练习

- 设某机主存容量为2^{20}个字节，机器字长32位。若最小寻址单位为字节（按字节编址），其地址码应为(**20**)位；若最小寻址单位为字（按字编址），其地址码只需(**18**)位。

3.2.2 数据寻址和指令寻址

- 寻找操作数的地址称为数据寻址，数据寻址方式较多，其最终目的都是寻找所需要的操作数。
- 寻找下一条将要执行的指令地址称为指令寻址。

3.2.2 数据寻址和指令寻址



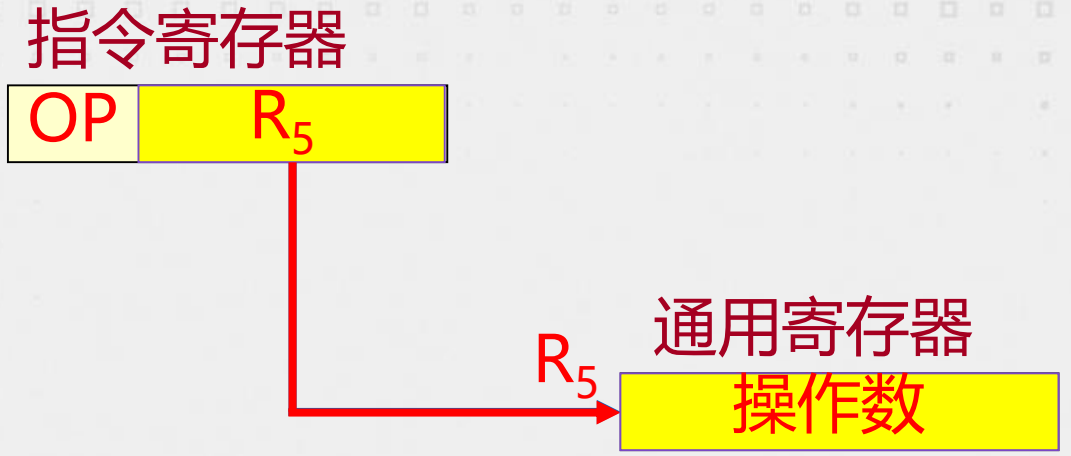
3.2.2 基本的数据寻址

- 寻址方式是根据指令中给出的地址码字段寻找真实操作数地址的方式。
  - 指令中的形式地址A —寻址方式→ 有效地址EA
- 1.立即寻址

  - 速度快
  - 立即数大小受限

3.2.2 基本的数据寻址

- 2.寄存器寻址
  - 指令中地址码部分给出某一通用寄存器的编号，所指定的寄存器中存放着操作数。
  - 优点：
    - 从寄存器存取数据比主存快得多；
    - 由于寄存器的数量较少，其地址码字段比主存单元地址字段短得多。

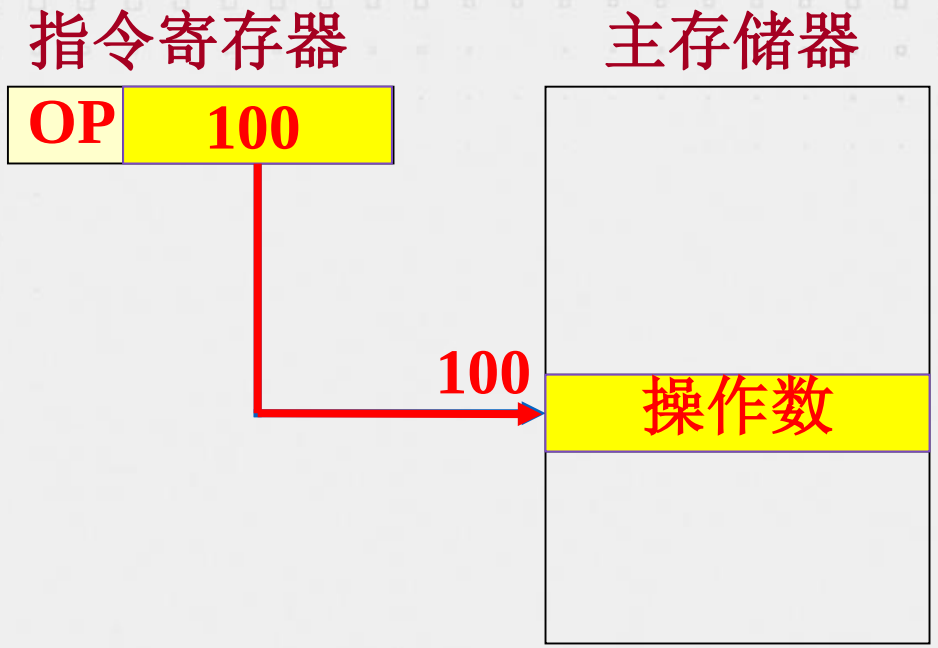


$EA = R_i$

操作数$S = (R_i)$

3.2.2 基本的数据寻址

- 3.直接寻址
  - 指令中地址码字段给出的地址A就是操作数的有效地址。
  - 由于操作数地址是不能修改的，与程序本身所在的位置无关，所以又叫做绝对寻址方式。

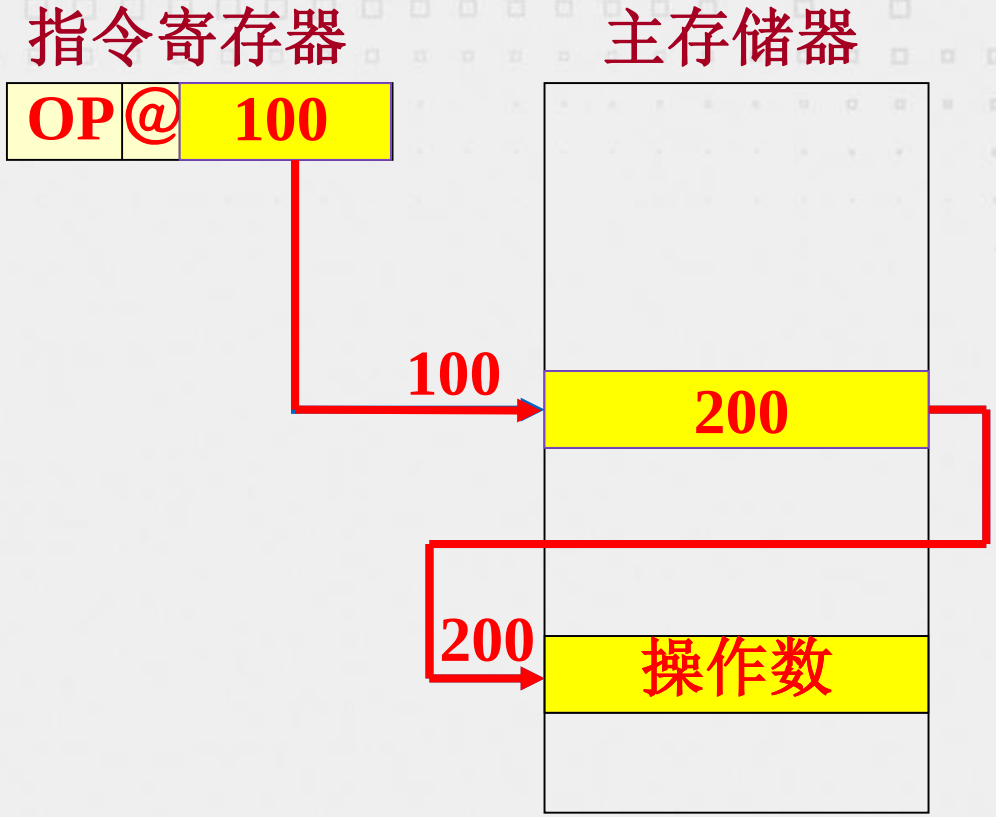


EA=A

操作数S=(A)

3.2.2 基本的数据寻址

- 4.间接寻址
  - 指令中给出的地址A是存放操作数地址的地址。
  - 通常在指令格式中划出一位@作为标志位。
    - @=0 直接寻址
    - @=1 间接寻址



EA=(A)

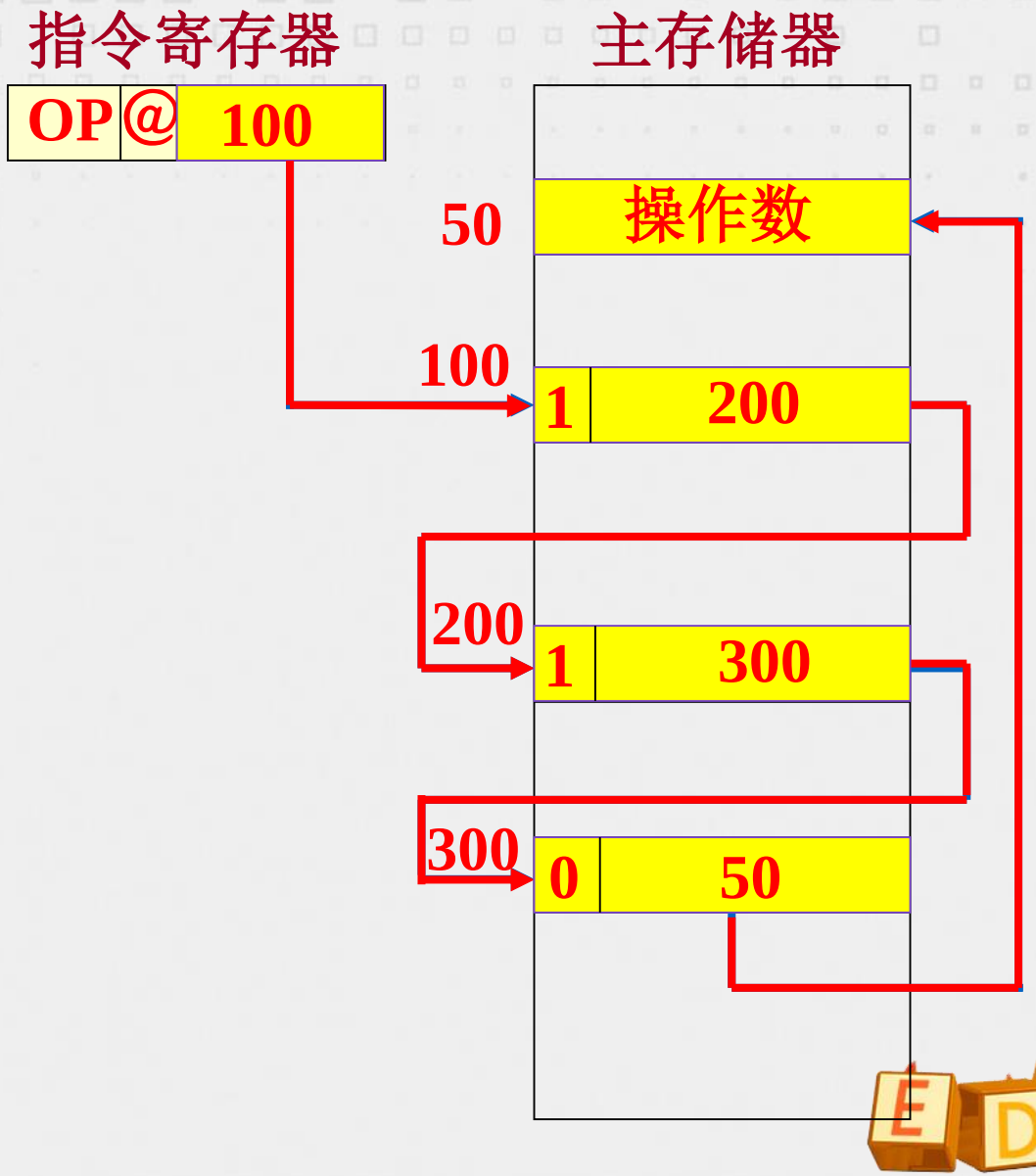
操作数S=((A))

3.2.2 基本的数据寻址

- 间接寻址要比直接寻址灵活得多，它的主要优点为：
  - 扩大了寻址范围，可用指令的短地址访问大的主存空间。
  - 可将主存单元作为程序的地址指针，用以指示操作数在主存中的位置。当操作数的地址需要改变时，不必修改指令，只需修改存放有效地址的那个主存单元（间接地址单元）的内容就可以了。

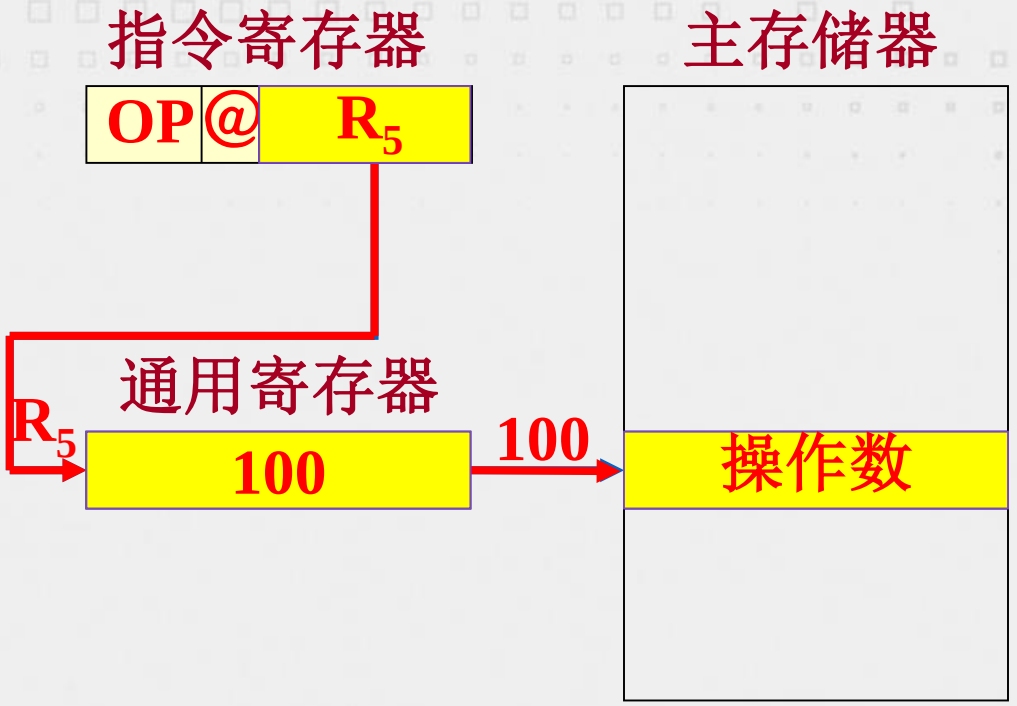
3.2.2 基本的数据寻址

- 多级间接寻址：取得操作数需要多次访问主存，即使在找到操作数有效地址后，还需再访问一次主存才可得到真正的操作数。
- 多级间接标志：
  - 0：找到有效地址
  - 1：继续间接寻址



3.2.2 基本的数据寻址

- 5.寄存器间接寻址
  - 指令中的地址码给出某一通用寄存器的编号，被指定的寄存器中存放操作数的有效地址，而操作数则存放在主存单元中。
  - 这种寻址方式的指令较短，并且在取指后只需一次访存便可得到操作数。

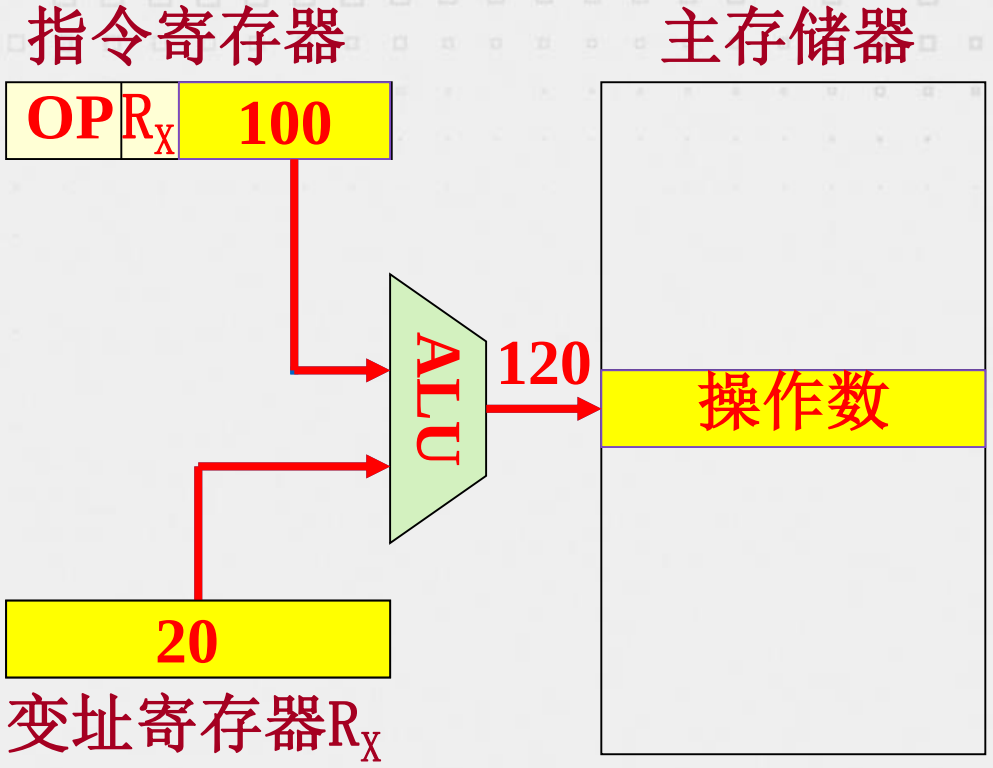


$EA=(R_i)$

操作数$S=((R_i))$

3.2.2 基本的数据寻址

- 6.变址寻址
  - 把指令给出的形式地址A与变址寄存器R_X的内容相加，形成操作数有效地址。
  - R_X 的内容为变址值



$EA= (R_X) +A$

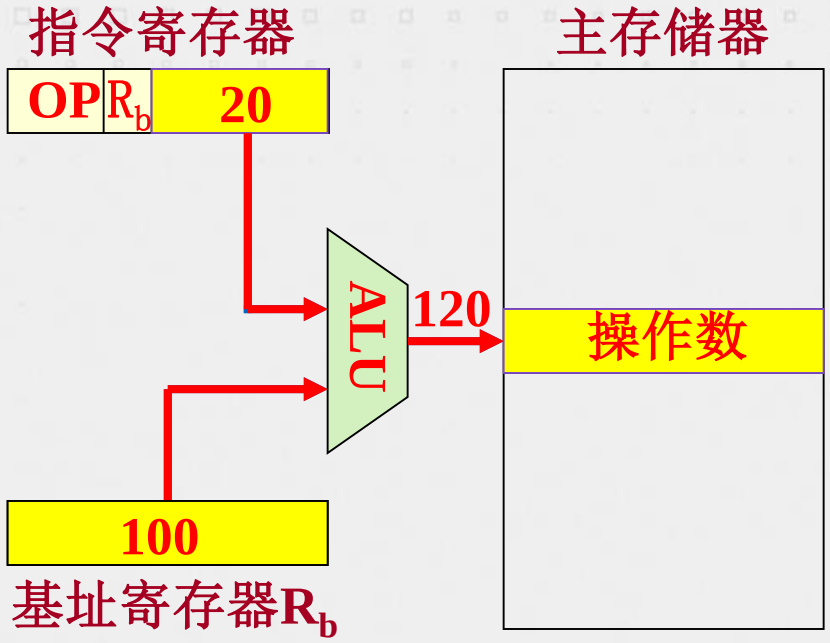
操作数$S=(A+(R_X))$

3.2.2 基本的数据寻址

- 变址寻址是一种广泛采用的寻址方式，通常指令中的形式地址作为基准地址，而R_X的内容作为修改量。在遇到需要频繁修改地址时，无须修改指令，只要修改变址值就可以了。

3.2.2 基本的数据寻址

- 7.基址寻址
  - 将基址寄存器R_b的内容与位移量D相加，形成操作数有效地址。
  - 基址寄存器的内容称为基址值，指令的地址码字段是一个位移量，位移量可正可负。



$$EA=(R_b)+D$$

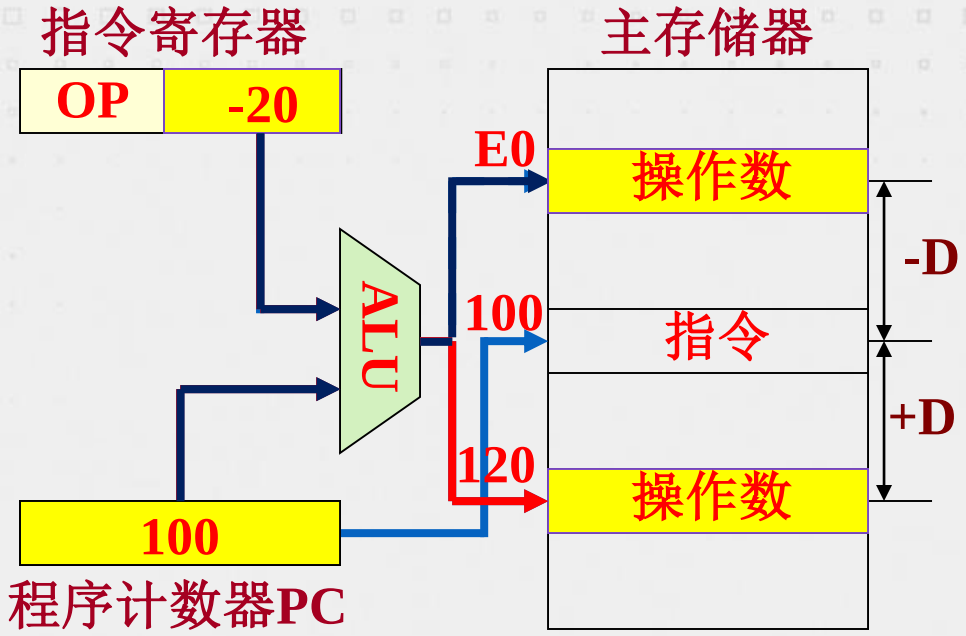
$$操作数S=((R_b)+D)$$

3.2.2 基本的数据寻址

- 基址寻址和变址寻址方式应用的场合不同，变址寻址是面向用户的，用于访问字符串、向量和数组等成批数据；而基址寻址面向系统，主要用于逻辑地址和物理地址的变换，用以解决程序在主存中的再定位和扩大寻址空间等问题。在某些大型机中，基址寄存器只能由特权指令来管理，用户指令无权操作和修改。

3.2.2 基本的数据寻址

- 8.相对寻址
  - 相对寻址是基址寻址的一种变通，由程序计数器PC提供基准地址。
  - 位移量指出的是操作数和现行指令之间的相对位置。



$$EA=(PC)+D$$

$$操作数S=((PC)+D)$$

3.2.2 基本的数据寻址

- 9.页面寻址
  - 页面寻址相当于将整个主存空间分成若干个大小相同的区，每个区称为一页，每页有若干个主存单元。每页都有自己的编号，称为页面地址；页面内的每个主存单元也有自己的编号，称为页内地址。这样，操作数的有效地址就被分为两部分：前部为页面地址，后部为页内地址。

页面地址	页内地址

3.2.2 基本的数据寻址

- 根据页面地址的来源不同，页面寻址又可以分成3种不同的方式：
  - ① 基页寻址，又称零页寻址。由于页面地址全等于0，所以有效地址EA=0//A（//在这里表示简单拼接），操作数S在零页面中，基页寻址实际上就是直接寻址。
  - ② 当前页寻址。页面地址就等于程序计数器PC的高位部分的内容，所以有效地址EA=(PC)H//A，操作数S与指令本身处于同一页面中。
  - ③ 页寄存器寻址。页面地址取自页寄存器，与形式地址相拼接形成有效地址。
- 有些计算机在指令格式中设置了一个页面标志位（Z/C）。当Z/C=0，表示零页寻址，当Z/C=1，表示当前页寻址。

3.2.2 基本的数据寻址

- 各种数据寻址方式获得数据的速度（由快到慢）
  - 立即寻址
  - 寄存器寻址
  - 直接寻址
  - 寄存器间接寻址
  - 页面寻址
  - 变址寻址（基址寻址、相对寻址）
  - 一级间接寻址
  - 多级间接寻址

3.2.2 基本的数据寻址

- 为了能区分出各种不同寻址方式，必须在指令中给出标识。标识的方式通常有两种：显式和隐式。
  - 显式的方法就是在指令中设置专门的寻址方式（MOD）字段，用二进制代码来表明寻址方式类型。
  - 隐式的方式是由指令的操作码字段说明指令格式并隐含约定寻址方式。

显式

OP	MOD	A

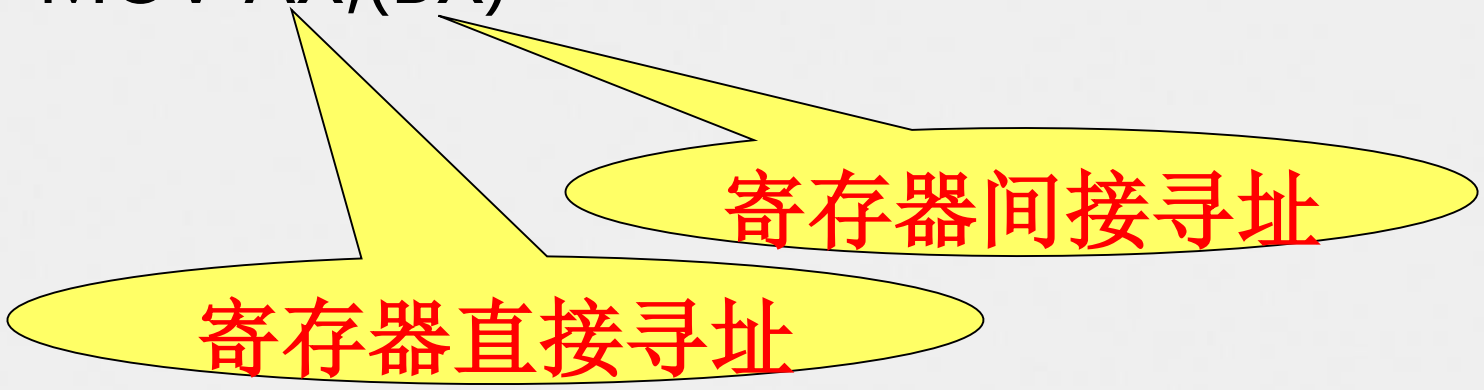
隐式

OP	A

3.2.2 基本的数据寻址

- 注意，一条指令若有两个或两个以上的地址码时，各地址码可采用不同的寻址方式。例如，源地址采用一种寻址方式，而目的地址采用另一种寻址方式。

MOV AX,(BX)

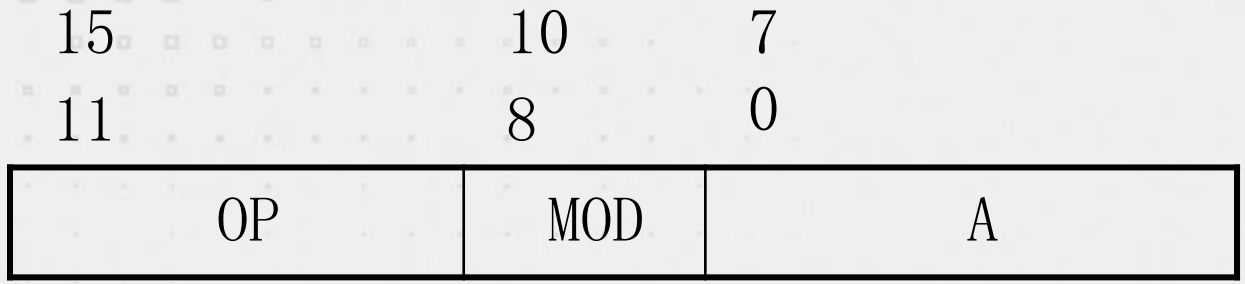


课堂习题

- 1.根据操作数所在位置，指出其寻址方式：
  - 操作数在寄存器中，称为__寄存器__寻址方式；操作数地址在寄存器中，称为__寄存器间接__寻址方式；操作数在指令中，称为__立即__寻址方式；操作数地址在指令中，为__直接__寻址方式。操作数地址为某一寄存器的内容与位移之和，则可以是__变址寻址__、__基址寻址__、__相对寻址__寻址方式。

课堂习题

- 2.某计算机的字长为16位，存储器按字编址，访存指令格式如图所示：



其中，OP是操作码，MOD定义寻址方式，A为形式地址。设PC和Rx分别为程序计数器和变址寄存器，字长为16位，问：

1.该格式能定义多少种指令？

2.各种寻址方式的寻址范围是多少字？

3.写出各种寻址方式的有效地址EA的计算公式。

课堂习题

- 3. (2011)偏移寻址通过将某个寄存器内容与一个形式地址相加而生成有效地址。下列寻址方式中，不属于偏移寻址方式的是(A)

  A.间接寻址　　B.基址寻址　　C.相对寻址　　D.变址寻址

- 4. (2013)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(D)

  A. 1000H　　B. 2000H　　C. 3000H　　D. 4000H

课堂习题

- 5. 指令系统中采用不同的寻址方式的目的主要是(B)

  A. 实现存储程序和程序控制

  B. 缩短指令长度，扩大寻址空间，提高编程灵活性

  C. 可以直接访问外存

  D. 提供扩展操作码的可能性并降低指令译码难度

目 录

3.3 堆栈与堆栈操作

- 堆栈是一种按特定顺序进行存取的存储区，这种特定顺序可归结为“后进先出”（LIFO）或“先进后出”（FILO）。
- 堆栈结构：寄存器堆栈、存储器堆栈

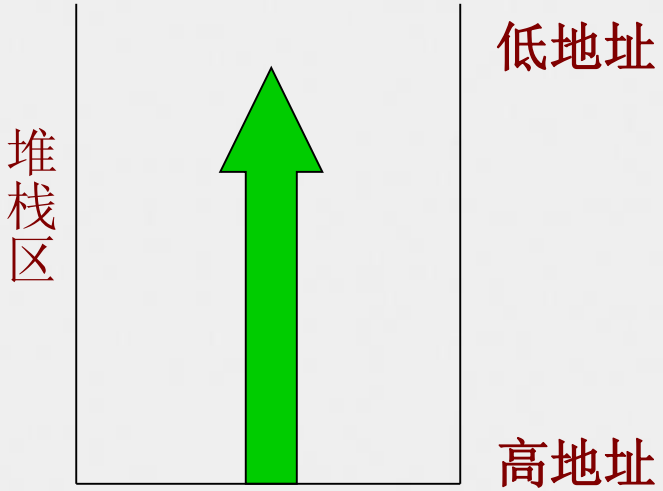
3.3.1 堆栈结构

- 1.寄存器堆栈
  - 用一组专门的寄存器构成寄存器堆栈，又称为硬堆栈。这种堆栈的栈顶是固定的，寄存器组中各寄存器是相互连接的，它们之间具有对应位自动推移的功能，即可将一个寄存器的内容推移到相邻的另一个寄存器中去。

3.3.1 堆栈结构

- 2.存储器堆栈
  - 从主存中划出一段区域来作堆栈，这种堆栈又称为软堆栈，堆栈的大小可变，栈底固定，栈顶浮动，故需要一个专门的硬件寄存器作为堆栈栈顶指针SP，简称栈指针。栈指针所指定的主存单元，就是堆栈的栈顶。

自底向上生成方式的堆栈

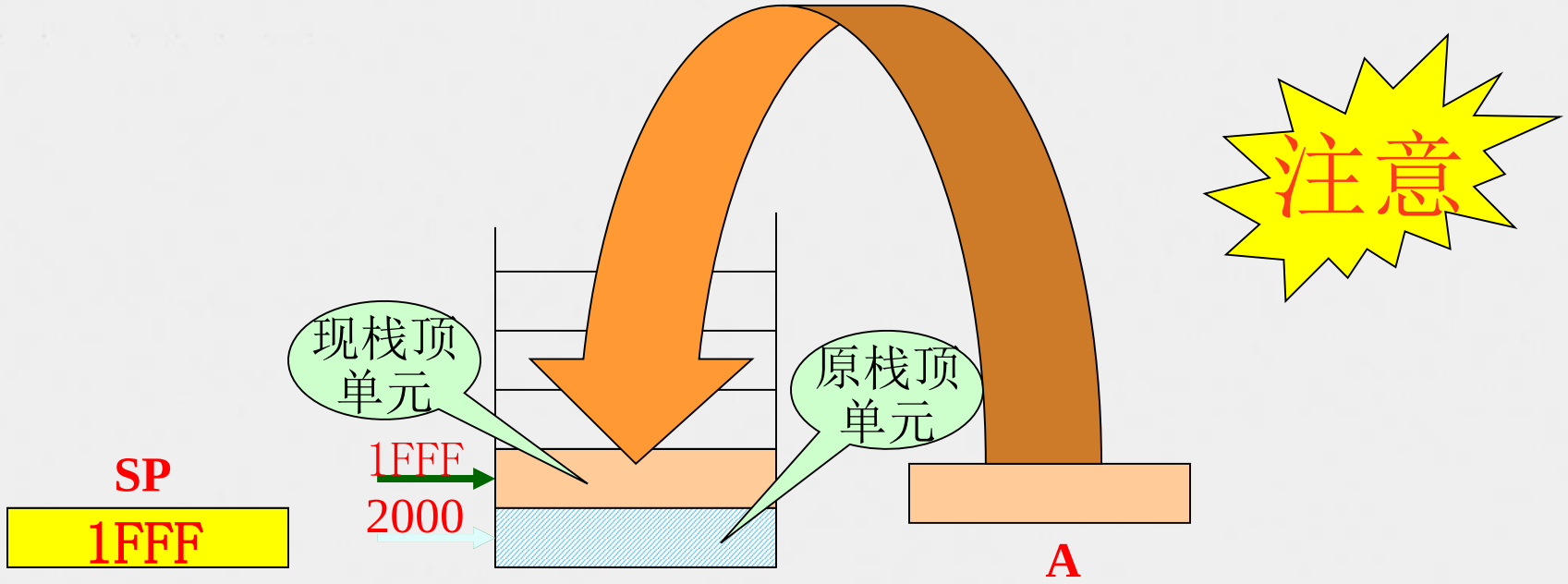


33 堆栈与堆栈操作（进栈）

- 堆栈的栈底地址大于栈顶地址，通常栈指针始终指向栈顶的满单元。进栈时，SP的内容需要先自动减1，然后再将数据压入堆栈。

  (SP)-1→SP　　修改栈指针

  (A)→(SP)　　将A中的数据压入堆栈

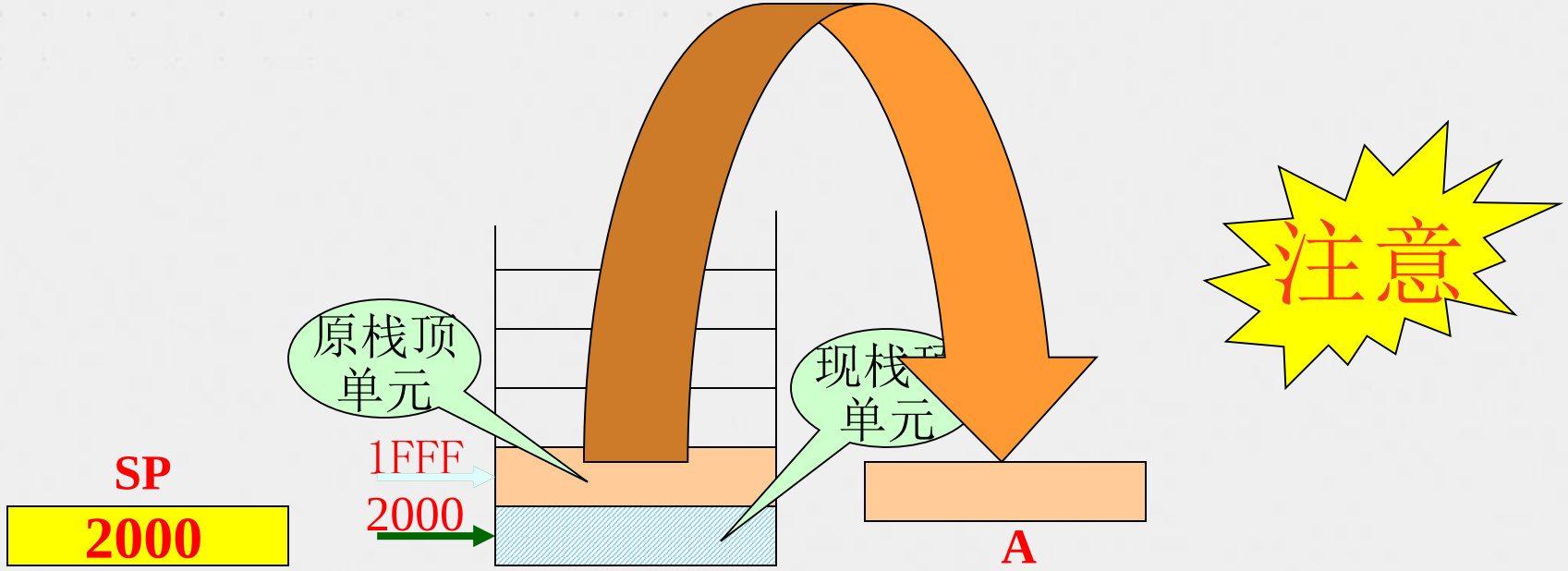


3.3 堆栈与堆栈操作（出栈）

- 出栈时，需要先将堆栈中的数据弹出，然后SP的内容再自动加1。

((SP))→A　　将栈顶内容弹出，送入A中

(SP)+1→SP　　修改栈指针



3.3.2 堆栈操作

- 在一般计算机中，堆栈主要用来暂存中断断点、子程序调用时的返回地址、状态标志及现场信息等，也可用于子程序调用时参数的传递，所以用于访问堆栈的指令只有进栈（压入）和出栈（弹出）两种。

目录

3.4.1 数据传送类指令

- 数据传送类指令主要用于实现寄存器与寄存器之间，寄存器与主存单元之间以及两个主存单元之间的数据传送。数据传送类指令又可以细分为：
  - 1.一般传送指令
  - 2.堆栈操作指令
  - 3.数据交换指令

3.4.1 数据传送类指令

- 1.一般传送指令
  - (1)主存单元之间的传送
    - MOV mem2,mem1，其含义为 (mem1)→mem2
  - (2)从主存单元传送到寄存器
    - MOV reg,mem，其含义为 (mem)→reg
    - 在有些计算机中，该指令用助记符LOAD表示，又称为取数指令。
  - (3)从寄存器传送到主存单元
    - MOV mem,reg，其含义为 (reg)→mem
    - 在有些计算机里，该指令用助记符STORE表示，又称为存数指令。
  - (4)寄存器之间的传送
    - MOV reg2,reg1，其含义为 (reg1)→reg2

3.4.1 数据传送类指令

- 2.堆栈操作指令
  - 堆栈指令是一种特殊的数据传送指令，分为进栈（PUSH）和出栈（POP）两种。
  - 因为堆栈（指软堆栈）是主存的一个特定区域，所以对堆栈的操作也就是对存储器的操作。
- 3.数据交换指令
  - 前述指令的传送都是单方向的，然而，数据传送也可以是双方向的，即将源操作数与目的操作数（一个字节或一个字）相互交换位置。

3.4.2 运算类指令

- 1.算术运算指令
  - 算术运算指令主要用于进行定点和浮点运算。这类运算包括加、减、乘、除以及加1、减1、比较等，有些机器还有十进制算术运算指令。
  - 绝大多数算术运算指令都会影响到状态标志位，通常的标志位有进位、溢出、全零、正负和奇偶等。

3.4.2 运算类指令

- 2.逻辑运算指令
  - 一般计算机都具有与、或、非、异或等逻辑运算指令。这类指令在没有设置专门的位操作指令的计算机中常用于对数据字（字节）中某些位（一位或多位）进行操作。
  - (1)按位测（位检查）：按位与运算

$$\begin{array}{r} XXXXXXXX \\ \wedge\ \ 00010000 \\ \hline 000X0000 \end{array}$$

AND AL,10H

3.4.2 运算类指令

- (2)按位清（位清除）：按位与运算

$$\begin{array}{rl} & XXXXXXXX \\ \wedge & 11111101 \\ \hline & XXXXXX0X \end{array}$$

AND AL,FDH

- (3)按位置（位设置）：按位或运算

$$\begin{array}{rl} & XXXXXXXX \\ \vee & 01000000 \\ \hline & X1XXXXXX \end{array}$$

OR AL,40H

3.4.2 运算类指令

- (4)按位修改：按位异或运算
  - 利用“异或”指令可以修改目的操作数的某些位，只要源操作数的相应位为“1”，其余位为“0”，异或之后就达到了修改这些位的目的（因为A⊕1=A，A⊕0=A）。

$$\begin{array}{r} XXXXXXXX \\ \oplus\ \ 00001000 \\ \hline XXXX\overline{X}XXX \end{array}$$

- (5)判符合
  - 若两数相符合，其异或之后的结果必定为“0”。
- (6)清0（与本身异或）
  - XOR AL,AL

3.4.2 运算类指令

- 3.移位指令
  - 分为算术移位、逻辑移位和循环移位3类，它们又可分为左移和右移两种。
  - 算术移位的对象是带符号数，算术移位过程中必须保持操作数的符号不变，左移一位，数值 × 2，右移一位，数值 ÷ 2。
  - 逻辑移位的对象是没有数值含义的二进制代码，因此移位时不必考虑符号问题。
  - 循环移位又按进位位是否一起循环分为两类：
    - 小循环（不带进位循环）
    - 大循环（带进位循环）

3.4.3 程序控制类指令

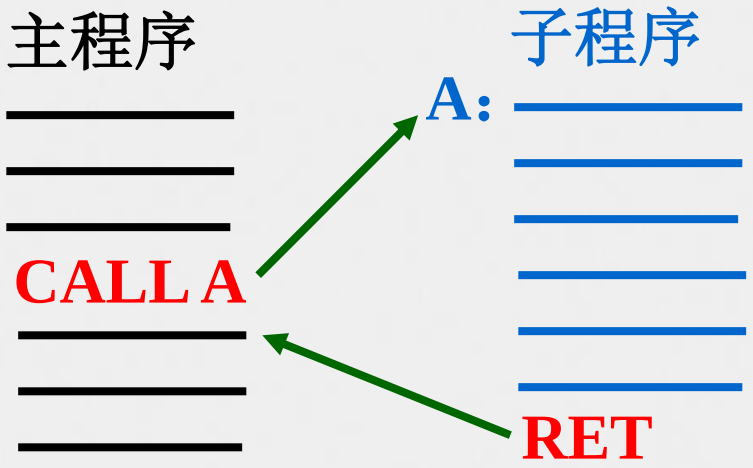
- 程序控制类指令用于控制程序的执行方向，并使程序具有测试、分析与判断的能力。
  - 1.转移指令
    - 在程序执行过程中，通常采用转移指令来改变程序的执行方向。转移指令又分无条件转移和条件转移两种。
    - 无条件转移指令（JMP）不受任何条件的约束，直接把程序转向新的位置执行。

3.4.3 程序控制类指令

- 1. 条件转移指令必须受到条件的约束，若条件满足时才转向新的位置执行，否则程序仍顺序执行。
  - 无论是条件转移还是无条件转移都需要给出转移地址。若采用相对寻址方式，转移地址为当前指令地址（即PC的值）和指令中给出的位移量之和，即：(PC)+位移量→PC；若采用绝对寻址方式，转移地址由指令的地址码直接给出，即：A→PC。

3.4.3 程序控制类指令

- 2.子程序调用指令
  - 子程序是一组可以公用的指令序列，只要知道子程序的入口地址就能调用它。
  - 从主程序转向子程序的指令称为子程序调用指令（CALL）；而从子程序转向主程序的指令称为返回指令（RET）。



3.4.3 程序控制类指令

- 主程序和子程序是相对的概念，调用其它程序的程序是主程序；被其它程序调用的程序是子程序。
- 转子指令安排在主程序中需要调用子程序的地方，转子指令是一地址指令。

3.4.3 程序控制类指令

- 子程序调用指令和转移指令都可以改变程序的执行顺序，但两者存在着很大的差别：
- 转移指令转移到指令中给出的转移地址处执行指令，不存在返回要求，没有返回地址问题；而子程序调用指令必须以某种方式保存返回地址，以便返回时能找到原来的位置。
- 转移指令用于实现同一程序内的转移；而子程序调用指令转去执行一段子程序，实现的是程序与程序之间的转移。

3.4.3 程序控制类指令

- 保存返回地址的方法除去堆栈以外还有：
  - (1) 用子程序的第一个字单元存放返回地址。
  - (2) 用寄存器存放返回地址。

3.4.3 程序控制类指令

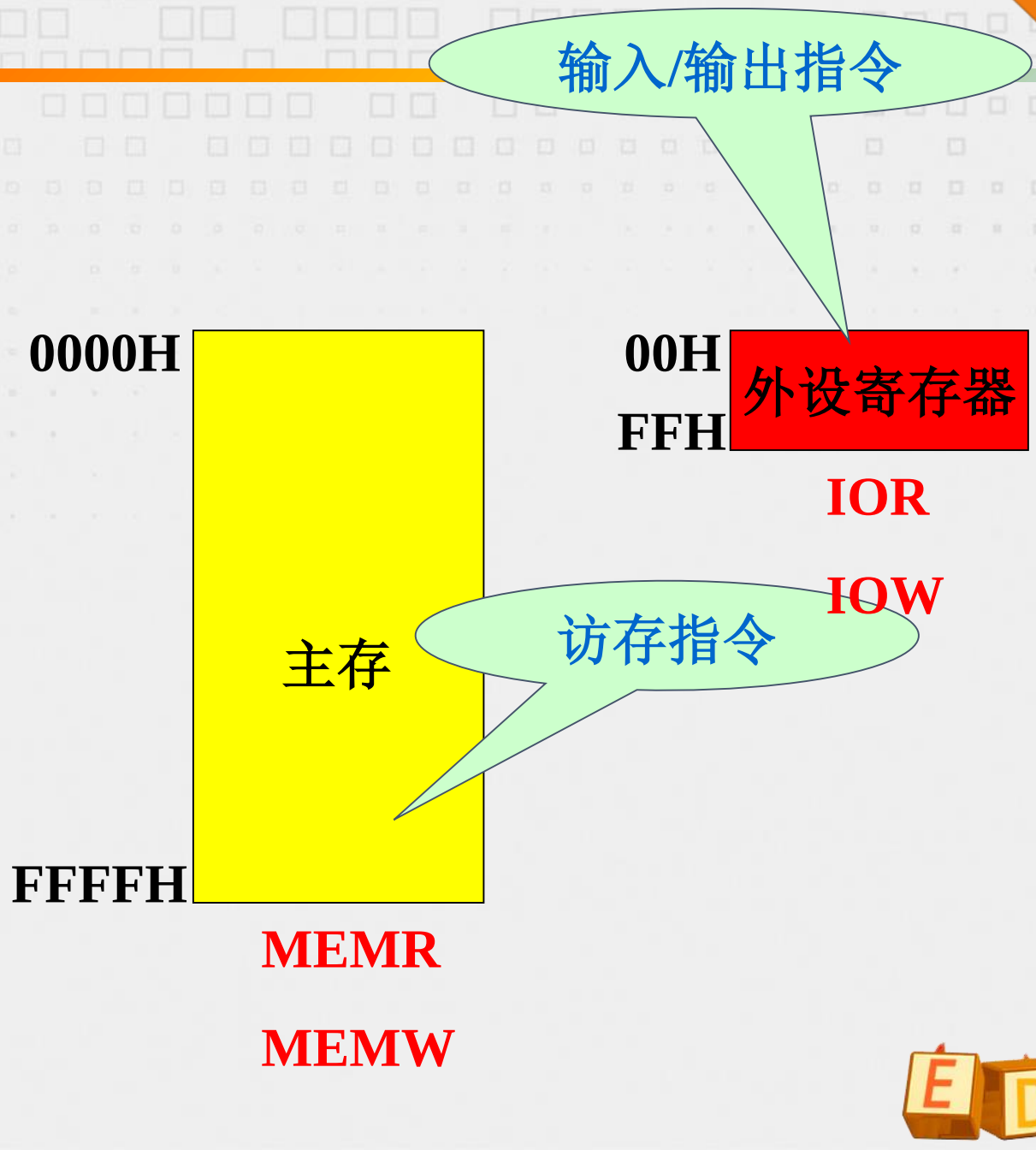
- 3.返回指令
  - 从子程序转向主程序的指令称为返回指令，其助记符一般为RET，子程序的最后一条指令一定是返回指令。
  - 返回地址存放的位置决定了返回指令的格式，如果返回地址保存在堆栈中，则返回指令是零地址指令，如果返回地址保存在某个主存单元中，则返回指令中必须是一地址指令。

3.4.4 输入输出类指令

- 输入/输出（I/O）类指令用来实现主机与外部设备之间的信息交换，包括输入/输出数据、主机向外设发控制命令或外设向主机报告工作状态等。从广义的角度看，I/O指令可以归入数据传送类。
- 各种不同的计算机的I/O指令差别很大，通常有两种方式：
  - 独立编址方式
  - 统一编址方式

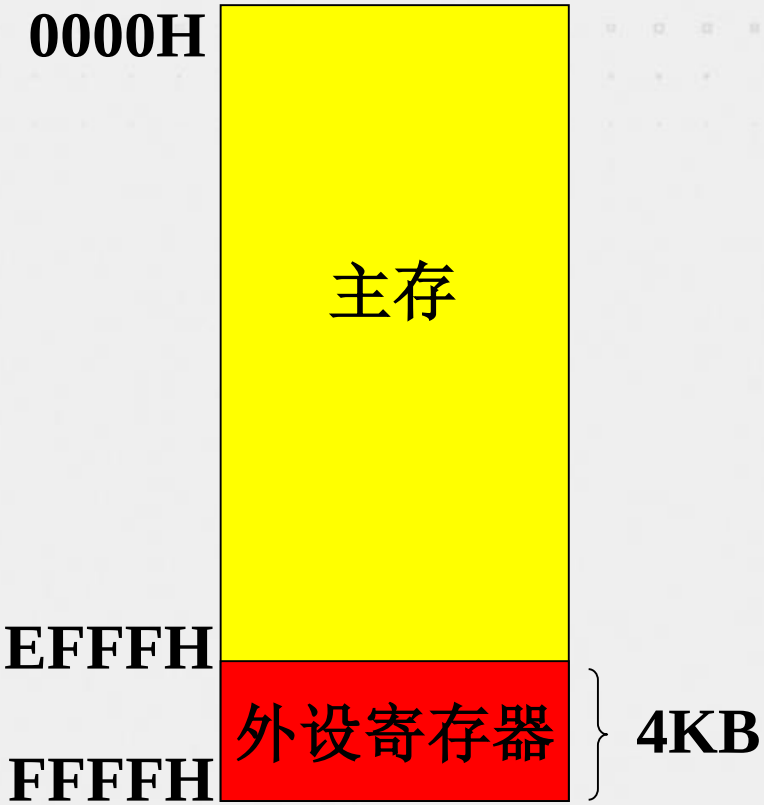
3.4.4 输入输出类指令

- 1.独立编址的I/O
  - 所谓独立编址就是把外设端口和主存单元分别独立编址。指令系统中有专门的IN/OUT指令。
  - 以主机为基准，信息由外设传送到主机称为输入，反之称为输出。指令中需要给出外设端口地址。这些端口地址与主存地址无关，是另一个独立的地址空间。



3.4.4 输入输出类指令

- 2.统一编址的I/O
  - 所谓统一编址就是把外设寄存器和主存单元统一编址。指令系统中没有专门的I/O指令，就用一般的数据传送类指令来实现I/O操作。



数据传送指令

输入指令

MOV R_0,(50H)

MOV R_0,(F000H)

课堂习题

- 1.某个主存单元，按照(A)编址时，需要的地址码位数最少。

  A. 字编址　　B. 位编址　C. 字节编址　　D. 以上选项一样多

- 2. 以下编址方式中外设有专用的I/O指令的是(C)。

  A.字节编址　　B. 字编址　C.独立编址　　D.统一编址

- 3. 某主存空间容量是1GB，字长32位，如果按照字编址，其地址码位数是（ D ）。

  A.15　　B. 30　　C. 25　　D. 28

课堂习题

- 4. 在统一编址方式下，下面的说法（D）是正确的。

  A. 一个具体的地址只能对应输入输出设备。

  B. 一个具体的地址只能对应内存单元。

  C. 一个具体地址既可对应输入输出设备又可对应内存单元。

  D.一个具体地址只对应I/O设备或者对应内存单元。

- 5. 在独立编址方式下，下面的说法（C）是正确的。

  A. 一个具体的地址只能对应输入输出设备。

  B. 一个具体的地址只能对应内存单元。

  C. 一个具体地址既可对应输入输出设备又可对应内存单元。

  D.一个具体地址只对应I/O设备或者对应内存单元。

课堂习题

- 6. 在独立编址方式下，存储单元和I/O设备是靠（ C ）来区分的。

  A .不同的地址代码　　B. 不同的地址总线

  C. 不同的指令或不同的控制信号　　D. 上述都不对

- 7. 在统一编址方式下，存储单元和I/O设备是靠（ A ）来区分的。

  A. 不同的地址代码　　B. 不同的地址总线

  C. 不同的指令或不同的控制信号　　D. 上述都不对

目 录

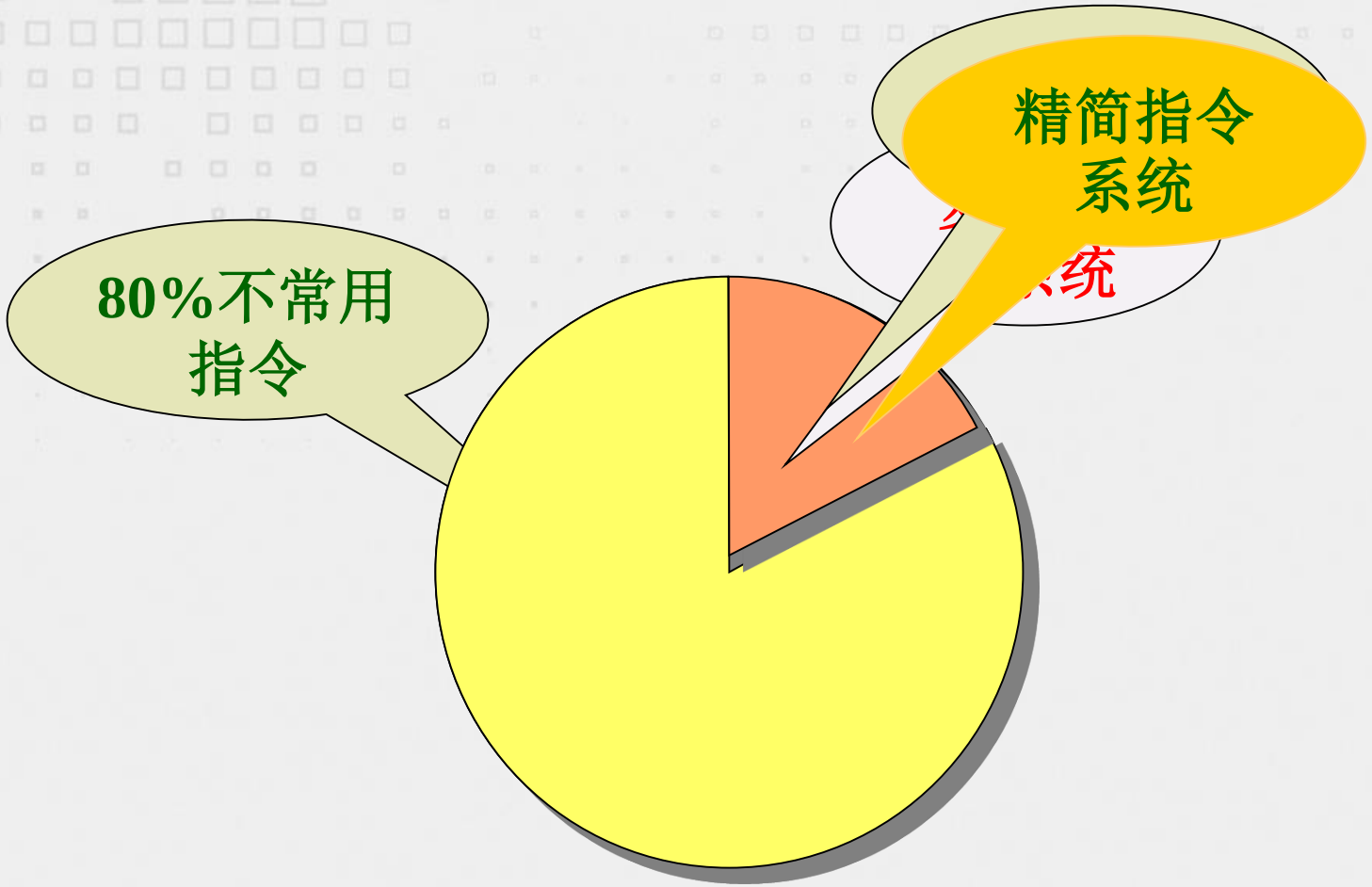
3.5.1 x86架构的扩展指令集

- MMX指令集
- SSE指令集
- 3DNow指令集
- SSE2指令集
- SSE3指令集

3.5.2 从复杂指令系统到精简指令系统

- 指令系统中指令丰富、功能强。这些计算机被称为复杂指令系统计算机，简称CISC。
- 大量测试表明，最常使用的是一些比较简单的指令，这类指令仅占指令总数的20%，但在各种程序中出现的频度却占80%，其余大多数指令是功能复杂的指令，这类指令占指令总数的80%，但其使用频度很低，仅占20%。因此，人们把这种情况称为“20%-80%律”。

3.5 指令系统的发展



3.5 指令系统的发展

- 从“20%-80%律”出发，人们开始了对指令系统合理性的研究，提出了精简指令系统的想法，出现了精简指令系统计算机，简称RISC。
- RISC技术将在后继课程中介绍。

指令系统的发展

- VLIW（Very Long Instruction Word，超长指令字
- EPIC（Explicit Parallel Instruction Code，显式并行指令代码）

课堂习题

- 1. RISC是精简指令计算机，CISC是复杂指令计算机。
- 2. 以下（B）计算机的CPU设计比较复杂，程序设计简单；（A）计算机的CPU设计比较简单，程序设计也比较复杂；（A）计算机使用流水线工作。

  A.RISC　　B.CISC

第3章 小结

- 3.1 指令格式
- 指令的基本格式
- 指令的地址码结构（3、2、1、0地址指令的区别）
- 非规整型指令的操作码（扩展操作码）

第3章 小结

- 3.2 寻址技术
  - 编址方式
    - ¤ 字编址、字节编址
  - 指令中地址码的位数
  - 主存容量、最小寻址单位
  - 数据寻址和指令寻址

第3章 小结

- 常见数据寻址方式的特点
  - 立即寻址、直接寻址、间接寻址、相对寻址、变址寻址、页面寻址
- 各种数据寻址方式的速度区别
- 有效地址EA的计算
  - 直接寻址、间接寻址、变址寻址、页面寻址

第3章 小结

- 3.3 堆栈与堆栈操作
  - 存储器堆栈
  - 存储器堆栈操作
  - 进栈、出栈时栈指针的修改和数据的压入和弹出
- 3.4 指令类型
  - 程序控制类指令
    - 转移、转子、返回指令的区别
  - 输入/输出类指令
    - 独立编址I/O、统一编址I/O

第3章 小结

- 3.5 指令系统的发展
  - CISC
  - RISC
  - VLIW
  - EPIC

THANK YOU