第 4 章 指令系统

主要内容：

- 指令系统概述
- 指令格式
- 操作数类型
- 指令和数据的寻址方式
- 典型指令

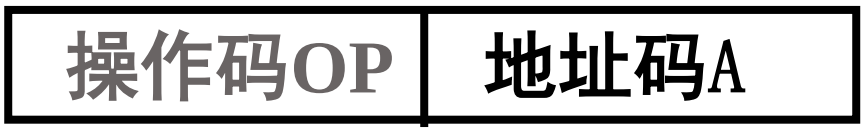
4.1 指令系统概述

- 指令：指示计算机执行某种操作的命令。
  - 微指令：硬件指令
  - 机器指令：硬件与软件的接口。
    - 硬件的任务是执行指令。
    - 指令的序列构成程序。
  - 宏指令：软件指令（语句）
- 指令系统：一台计算机所有机器指令的集合。

  它代表了一台计算机的硬件功能。
  - 复杂指令系统（CISC）
  - 精简指令系统（RISC)

4.2 指令的格式

- 指令字：机器指令用机器字表示,称为指令字，简称指令。
- 指令的格式：指令字用二进制代码表示的结构形式：
  - 操作码字段：表示指令的操作特性与功能。
  - 地址数字段：表示参与操作的操作数的地址，含被操作数地址、操作数地址和操作结果地址。
- 指令的功能：根据操作码对地址码提供的操作数完成某种操作。



4.2.1 操作码

- 每条指令都要规定一个操作码
- 不同的指令用（操作码字段的）不同编码表示，操作码的位数与指令规模有关。
  - 固定长度操作码
    - 优点：简化硬件设计，减少译码时间。
    - 缺点：操作码的平均长度长，需要指令字长长。
    - 一般用于大中型机和超级小型机。
  - 可变长度操作码
    - 优点：根据地址码长度调整，可以压缩操作码的平均长度。
    - 缺点：控制器的设计相对较复杂，指令的译码时间也较长。
    - 为提高速度，一般使用频度高的指令分配短的操作码；使用频度低的指令分配长的操作码。
    - 一般用于微、小型机。

4.2.2 地址码

根据指令中有几个操作数可分成三地址、二地址、一地址和零地址几种结构形式。

1、零地址指令

格式：

- 这种指令字不含地址码字段，有两种可能：
  - 不需要操作数的指令；
  - 所需操作数都是隐含指定。

2、一地址指令

格式：

•使用隐地址可以减少指令中的地址数，简化地址结构。

功能：

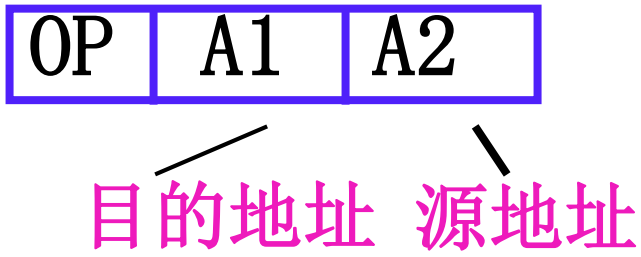
单操作数：　OP (A)　→ A

双操作数：　(AC) OP (A)　→ AC

隐含约定

3、二地址指令

格式：



功能：　(A1) OP (A2)　→ A1

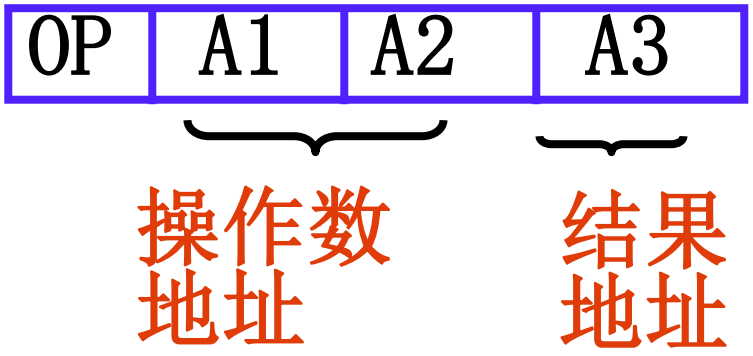
OP (A2)　→ A1

➢二地址指令格式中，根据操作数的物理位置可分为三类：

- *存储器-存储器（SS）型*：参与操作的数都放在内存里。
- *寄存器-寄存器（RR）型*：参与操作的数都放在寄存器里。
- *寄存器-存储器（RS）型*：一个操作数在内存，一个在寄存。

4、三地址指令

格式：



功能：(A1) OP (A2) → A3

4.2.3 指令字长度

指令字长度为一个指令字中包含二进制代码的位数。

- 机器字长：计算机能直接处理的二进制数据的位数，它决定了计算机的运算精度。
- 根据指令字长与机器字长关系将指令分为：
  单字长指令、半字长指令、双（多）字长指令。
- 一个指令系统根据指令字长可分为：
  变长指令格式、固定长指令格式

4.2.4 指令助记符

指令助记符是指令代码的符号化。

- 不同计算机助记符规定不一样；
- 助记符必须转换成代码机器才可识别。

例1　指令格式如下所示，其中OP为操作码，试分析指令格式的特点。

- 解：

(1)单字长指令。

(2)操作码字段OP可以设计128条指令。

(3)二个地址码，源寄存器和目标寄存器都是通用寄存器（可分别指定16个），所以是RR型指令，两个操作数均在寄存器中。

例2　指令格式如下所示，其中OP为操作码，试分析指令格式的特点。

解：

（1)双字长指令。

(2)操作码字段OP为6位，可以设计64种操作。

(3)二个地址码，一个操作数在源寄存器（共16个），另一个操作数在存储器中（由变址寄存器和位移量决定）所以是RS型指令。

4.3 操作数的类型

- 地址数据
- 数值数据
- 字符数据
- 逻辑数据

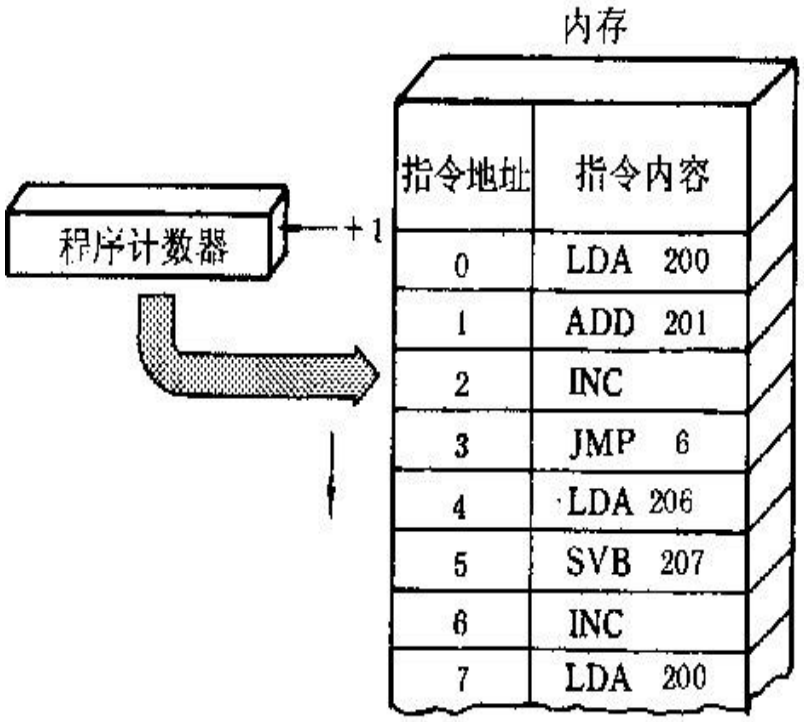
4.4 指令和数据的寻址方式

- 存储器数据读写方式：地址指定方式、相联存储方式和堆栈存取方式。
- 寻址方式：地址指定方式中，形成操作数或指令地址的方式。
- 冯诺依曼型计算机中，内存中指令的寻址和数据的寻址是交替进行的。

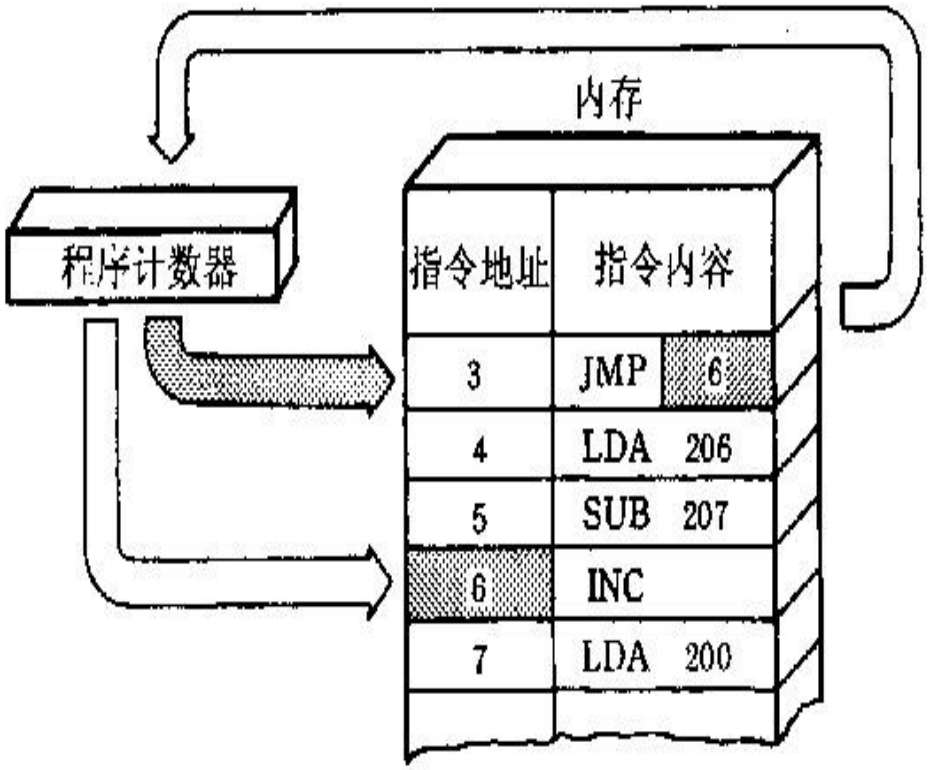
地址码给出的地址值不一定能直接使用，需根据寻址方式得到可用的地址

4.4.1 指令的寻址方式

- 顺序寻址方式
- 跳跃寻址方式



(a) 指令的顺序寻址方式



(b) 指令的跳跃寻址方式

4.4.2 操作数的寻址方式

操作数的寻址方式说明了形成操作数的有效地址的方法。

- 寻址方式的含义有二个：一是要表示指令所需的操作数在何处（如在寄存器中或主存单元中）；二是要给出获取操作数地址的方法。

- 指令中表达寻址方式的方法
  - 操作码隐含说明不同寻址方式
  - 指令中设置专门字段说明寻址方式

例如右所示的一种单地址指令结构

操作码 OP	寻址模式 X	间址标志 I	形式地址 A

- 关于地址的术语：
  - 有效地址（物理地址EA）：可以直接取数的地址；
  - 形式地址（A）（偏移量）：地址须变换才可取地址；

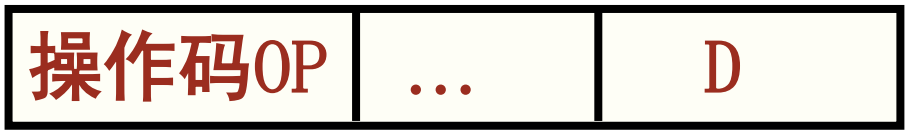
1、隐含寻址

操作数的地址不由地址码指明，而是隐含在操作码中。不访问存储器

2、立即寻址

指令地址段直接给出操作数。

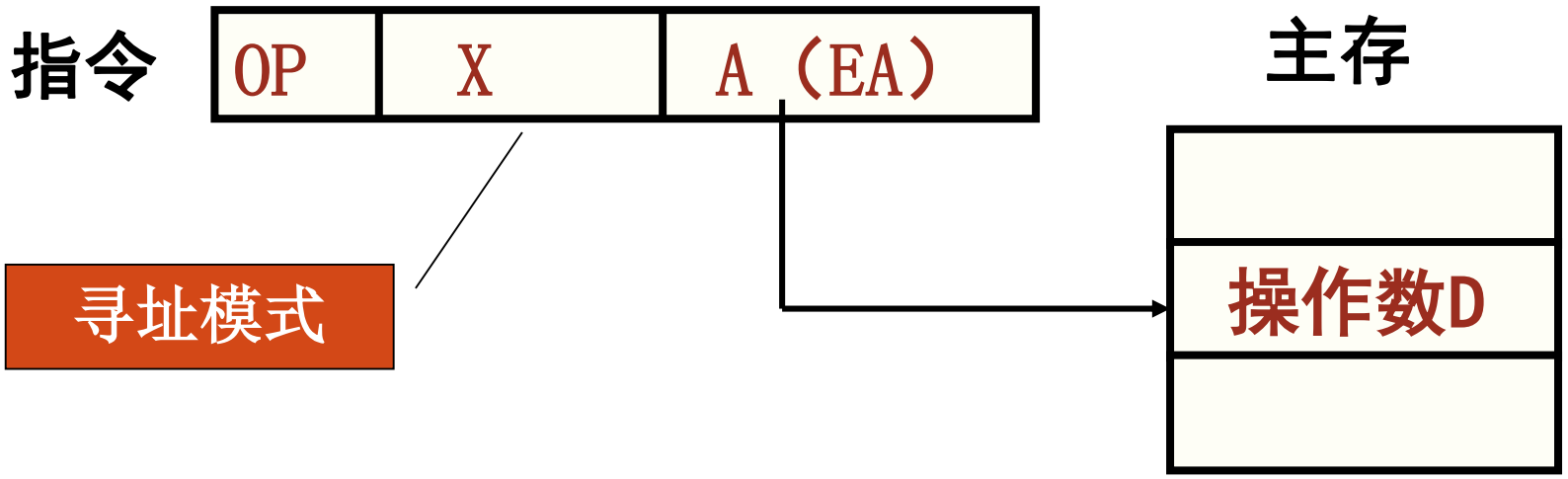
不访问存储器



D是操作数

3、直接寻址

指令直接给出操作数地址，根据该地址可从主存单元中读取操作数。寻址过程可描述为：

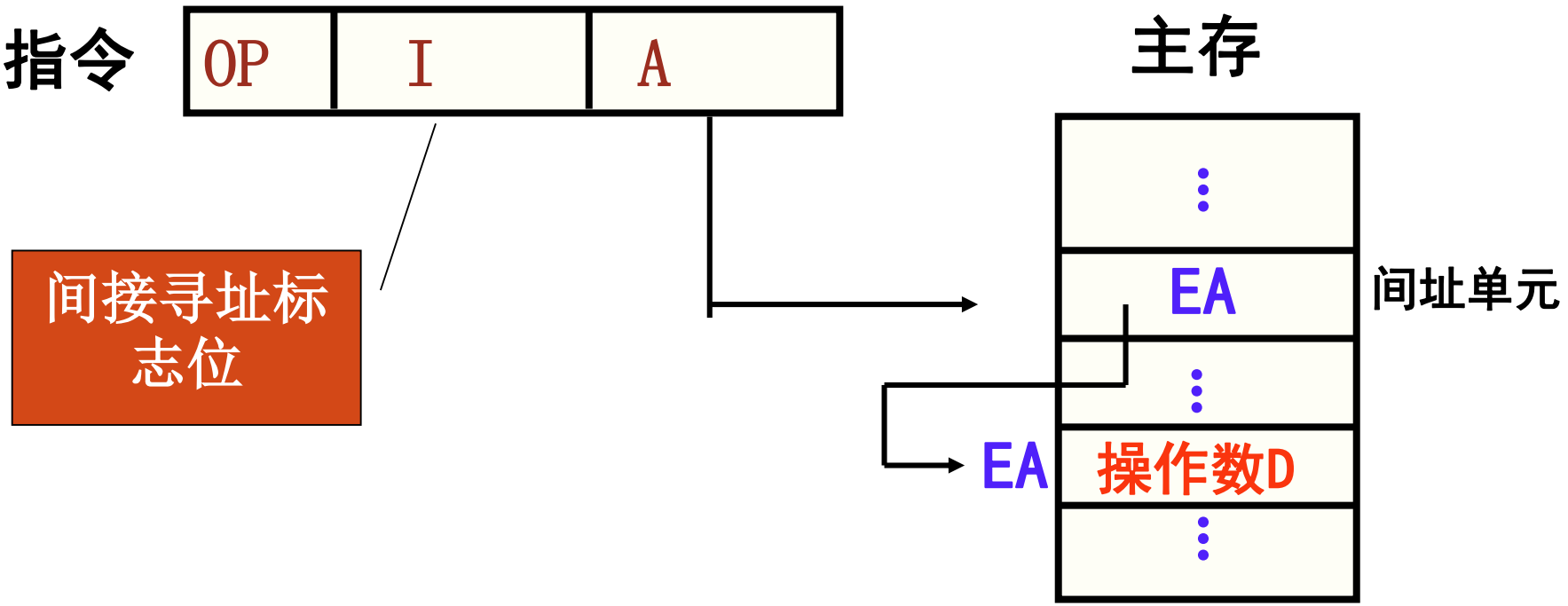


也可表示为：D=（A）

访问存储器一次

4、间接寻址

指令给出存放操作数地址的主存单元地址，即操作数的间接地址。寻址过程可描述为：



也可表示为：D=（EA）=（（A））

访问存储器多次

5、寄存器寻址

指令中给出寄存器号（也称寄存器地址），从寄存器中获取操作数。寻址过程可描述为：

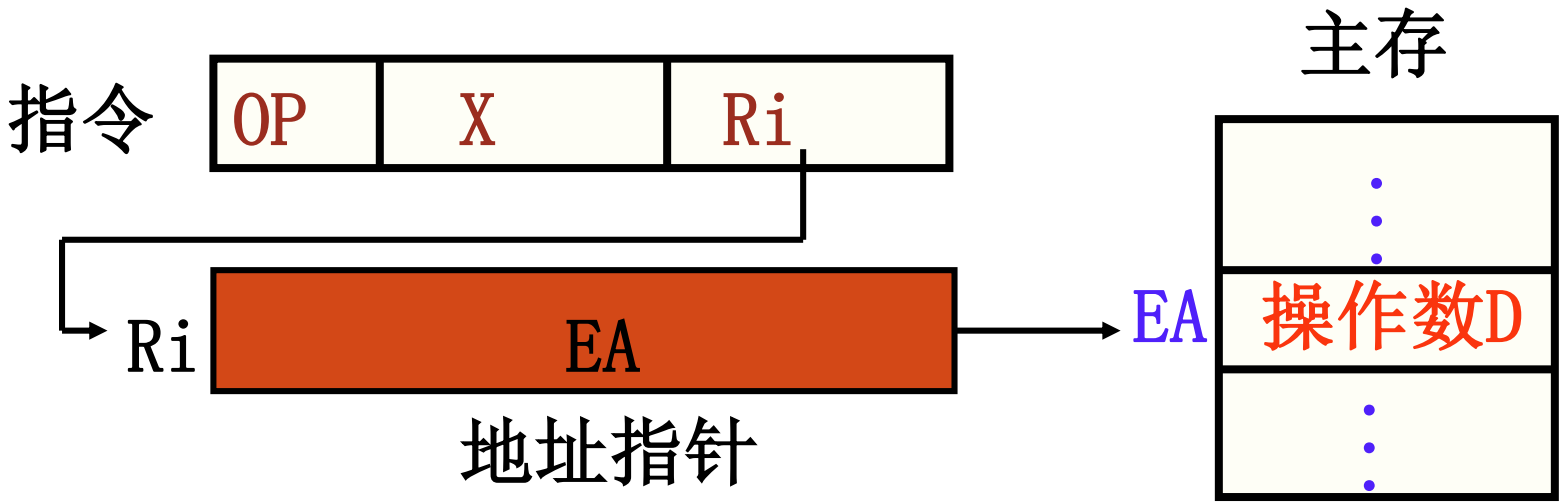
也可表示为：D=（R）

- 该寻址方式的优点：
  - 寻址速度快
  - 可减少一个操作数地址的位数

不访问存储器

6、寄存器间接寻址

操作数在主存单元中，由指令给出寄存器号，该寄存器存放操作数地址。寻址过程可描述为：



也可表示为：D=((Ri))

- 该寻址方式的优点：
  - 寻址速度比间址寻址快
  - 可减少一个操作数地址的位数

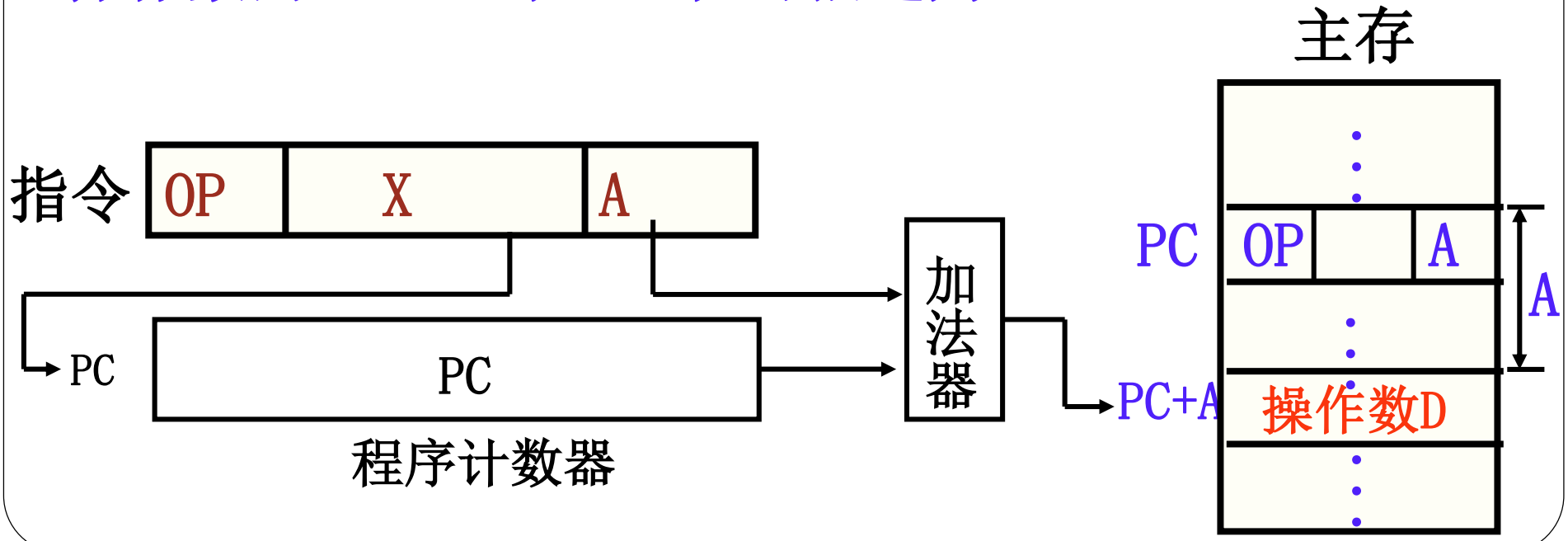
访问存储器一次

7、偏移寻址

有效地址由寄存器内容加一偏移量组成

（1）相对寻址

用程序计数器PC的内容作为基准地址，指令中给出的形式地址作为位移量（可正可负），二者相加后形成操作数的地址。寻址过程可描述为：



寻址过程也可表示为：

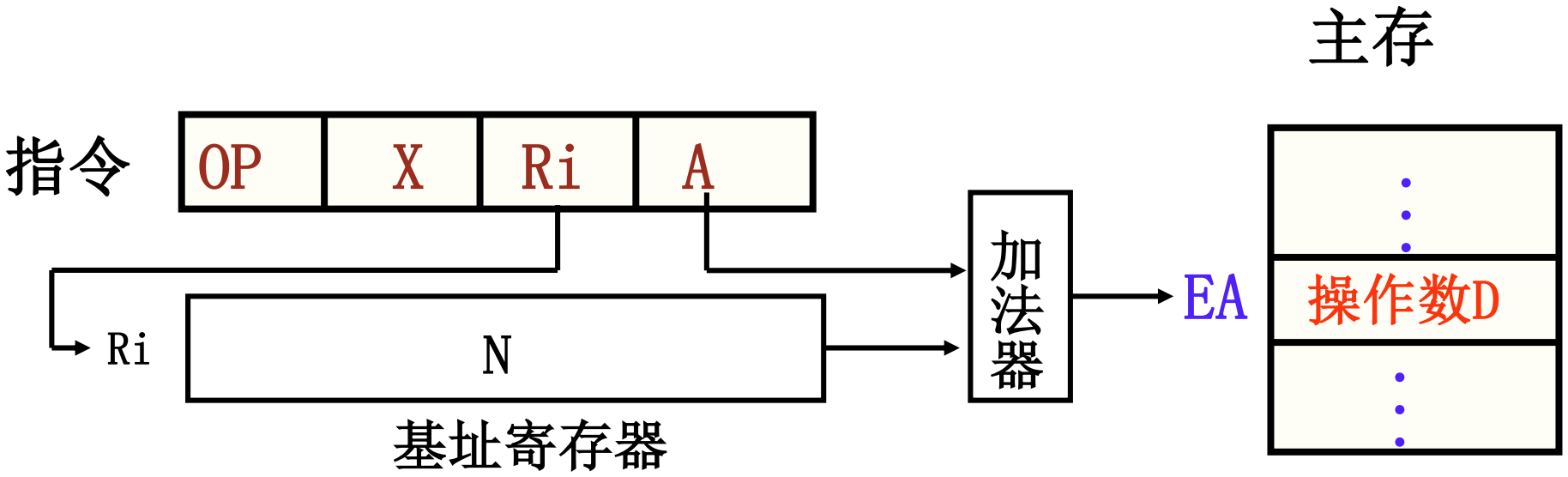
$$D=((PC)+A)$$

特点：

- 隐含引用专用寄存器PC
- 操作数地址随PC内容变化而改变，但二者之间的距离不变，可使操作数与指令在主存中一起移动；
- 位移量A可正可负，表示操作数地址可以在指令地址之后或之前。

（2）基址寻址

指令给出一个形式地址，指定CPU中一个寄存器作为基址寄存器，将基址寄存器内容（作为基准量）与形式地址相加得到操作数地址。寻址过程可描述为：



也可表示为：D=((Ri)+A)　　A一般为无符号数

（3）变址寻址

指令给出一个形式地址，并给出变址寄存器号；形式地址（作为基准量）与变址寄存器内容相加得到操作数地址。

基址寻址与变址寻址在形成操作数地址的方法上很相似，但主要应用目的不同：

- 变址寻址面向用户，用于访问字符串、线形表、一维数组等；
- 基址寻址面向系统，用来解决程序在主存中重定位的问题，以及在有限字长指令中扩大寻址空间等。

（8）段寻址

指令给出一个形式地址，CPU中设有专用寄存器作为段基址寄存器，先对段寄存器进行固定操作后再与形式地址相加得到操作数地址。

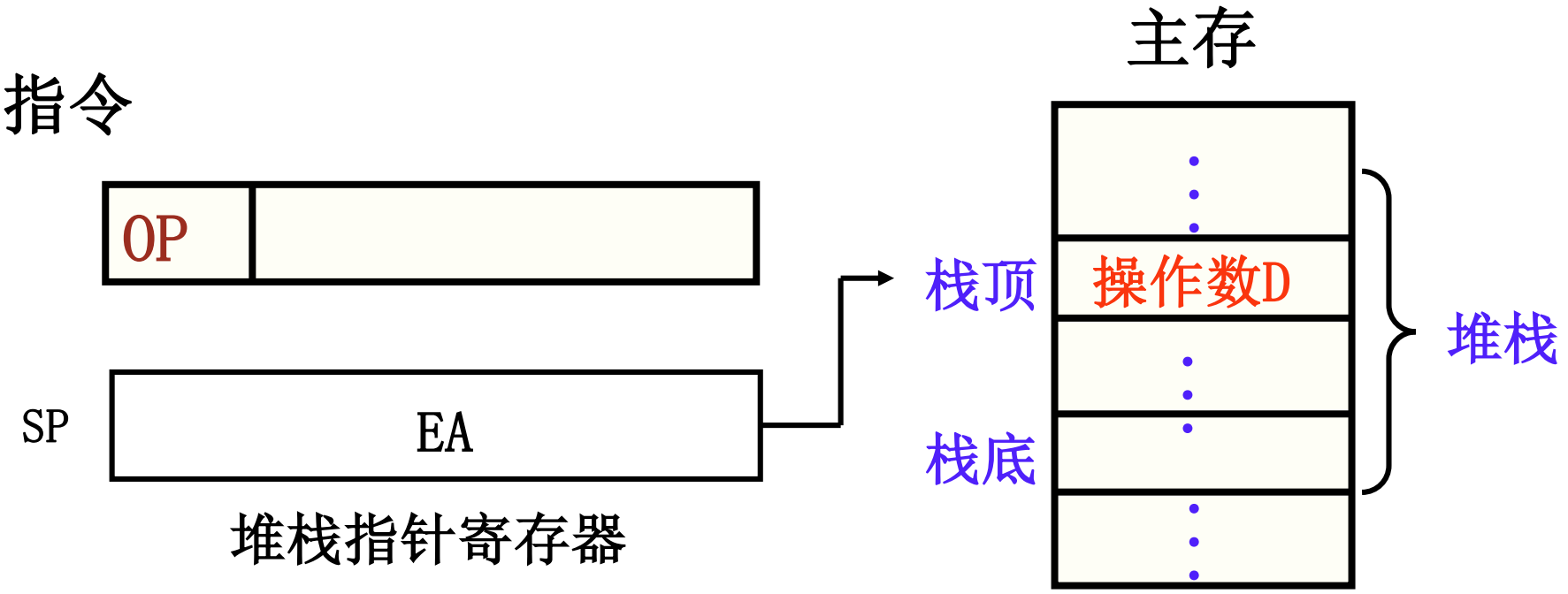
例x86机寻址为：$S=((R)_{段}x16+A)$

在段地址寄存器仅有16位的情况下，有1M的寻址能力

（9） 堆栈寻址方式

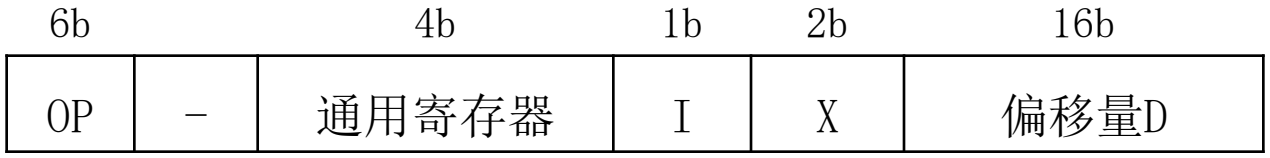
- 堆栈：计算机中暂时存储数据的存储单元，分成寄存器堆栈和存储器堆栈。
  - 存储数据特点：后进先出。
- 寄存器堆栈：在CPU中设有一组专门的寄存器，采用串联方式进、出栈。
  - 进、出栈地址不变，数据位置改变。
- 存储器堆栈：在主存中分配一块存储单元，指令隐含约定由堆栈指针专用寄存器（SP）提供堆栈栈顶单元地址，进行读出或写入。
  - 进、出栈的地址变化，数据位置不变。

存储器堆栈寻址过程：



存储器的堆栈区有二端，作为起点的一端固定称为栈底；另一端称为栈顶。对堆栈的读出（弹出）或写入（压入）都是对栈顶单元进行，因此CPU中设具有加减计数功能的SP指示栈顶的位置。

例3. 一种二地址RS型指令的结构如下所示：



其中I为间接寻址标志位，X为寻址模式字段，D为偏移量字段。通过I,X,D的组合，可构成如下所示的寻址方式。

寻址方式	I	X	有效地址E的算法	说明
（1）	0	0	E=D	
（2）	0	01	E=(PC)±D	PC为程序计数器
（3）	0	10	E=(R2)±D	R2为变址寄存器
（4）	1	11	E=(R3)	
（5）	1	00	E=(D)	
（6）	0	11	E=(R1)±D	R1为基址寄存器

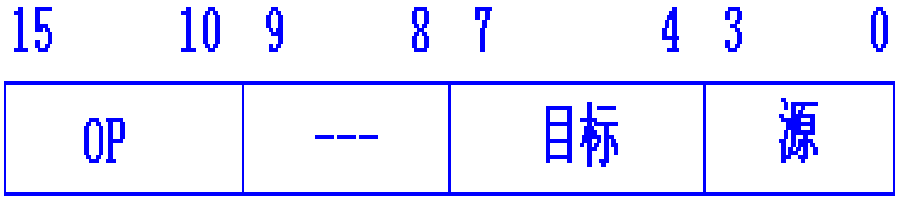
请写出6种寻址方式的名称。

- 例[4] 某16位机器所使用的指令格式和寻址方式如下所示，该机有两个20位基址寄存器，四个16位变址寄存器，十六个16位通用寄存器，指令汇编格式中的S（源），D（目标）都是通用寄存器，M是主存中的一个单元。三种指令的操作码分别是MOV（OP）＝（A）H，STO（OP）＝（1B）H，LDA（OP）＝（3C）H。MOV是传送指令，STO为存数指令，LDA为取数指令。
- 要求：⑴分析三种指令的指令格式与寻址方式特点。⑵CPU完成哪一种操作所花时间最短？哪一种操作所花时间最长？第二种指令的执行时间有时会等于第三种指令的执行时间吗？ ⑶下列情况下每个十六进制指令字分别代表什么操作？其中如果有编码不正确，如何改正才能成为合法指令？

①(F0F1)H (3CD2)H ②(2856)H ③（6FD6)H ④（1C2)H

单字长，RR型

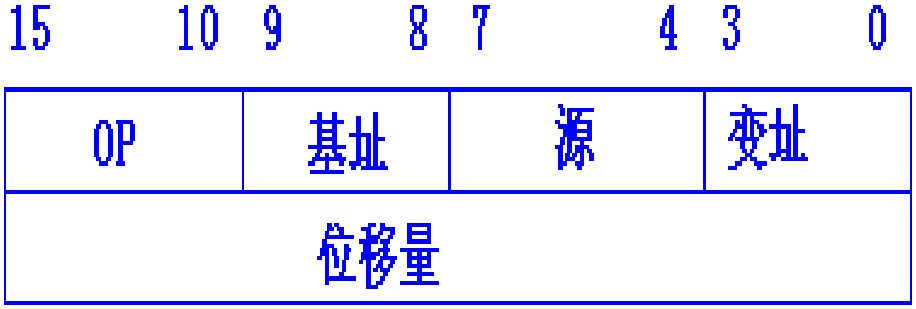
最短



MOV S, D

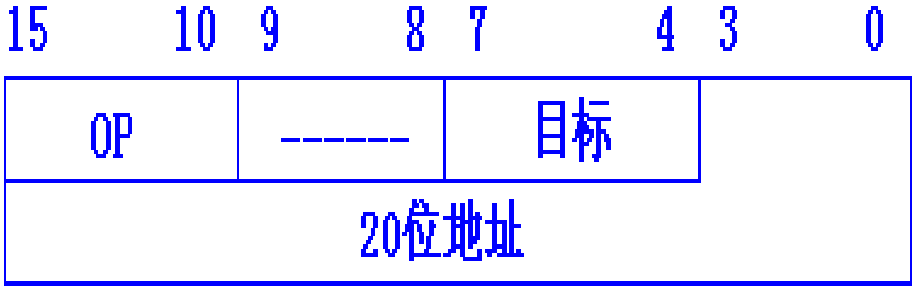
双单字长，RS型

最长



STA S, I

双字长，RS型



LDA S, I

① (F0F1)H (3CD2)H

（ 1111 0000 $)_2$ （ F13CD2) H

根据指令长度和op，该指令功能：

（13CD2H）-> R_F

② (2856)H

（ 00101000$)_2$ （ 56) H

根据指令长度和op，该指令功能：

R_6-> R_5

③（6FD6)H

（ 01101111 $)_2$ （ D6) H

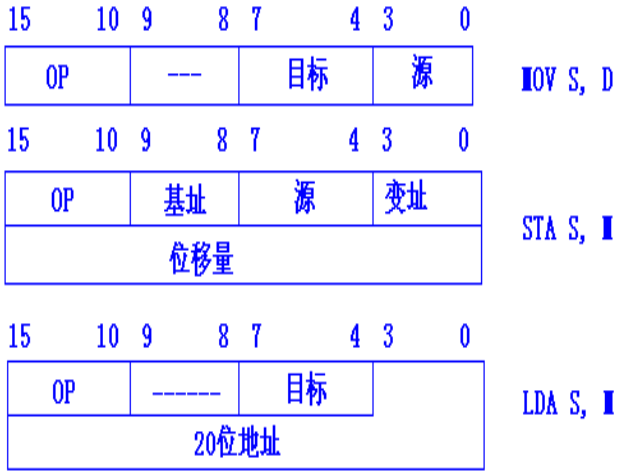
根据指令长度和op可知编码错误，可改为

（ 00101011 $)_2$ （ D6) H

R_6-> R_D

④（1C2)H

根据指令长度可知编码错误



MOV（OP）=（A）H = 001010
STO（OP）=（1B）H = 011011
LDA（OP）=（3C）H = 111100

思考：下列指令中，各个操作数采用何种寻址方式？

1　MOV AX，[0050H]　　MOV [AX]，0050H

2

```
ARR  DW  0000H, 0001H...
…
MOV SI, 0
MOV CX, 10
NEXT :  MOV  AX，ARR[SI]
…
INC  SI
LOOP NEXT
```