计算机组成原理

第五章 指令系统

5.3 操作数寻址方式

5.3 操作数寻址方式

- 形成操作数有效地址的方法
- 当数据在主存中时， 需要计算机其有效地址E
- S= (E)

5.3 操作数寻址方式

1 立即数寻址

- 地址码字段是操作数本身

- 例　MOV　AX,200H　　（AX ← 200H）
- S=D
- 特点：
  - 取指操作将数据与指令一并读入CPU内部的寄存器，指令执行速度快
  - 便于程序设计（变量赋初值）
  - 数据大小受字段位数限制

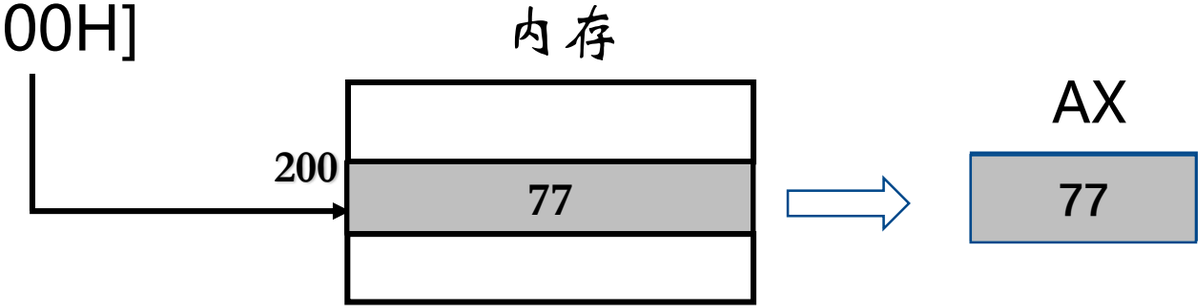
2 寄存器寻址

- 操作数在CPU的内部寄存器中

- 例　MOV　AX, BX　　（AX ← （BX））
- 特点：
  - 操作数在寄存器中，指令执行速度快
  - 能访问的数据大小一般与计算机字长有关
  - 地址字段的位数与计算机通用寄存器数量相关

3 直接寻址

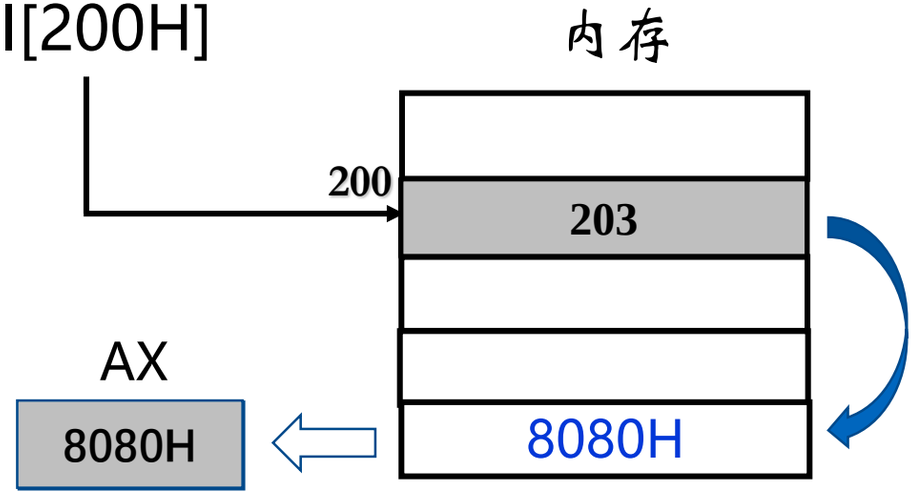
- 地址码字段直接给出操作数在内存的地址. E=D, S=(D)
- 例 MOV AX, [200H]



- 特点:
  - 提供访问主存的操作
  - 获得数据要访问主存，指令执行速度慢
  - 地址字段的位数决定了访存空间大小

4 间接寻址

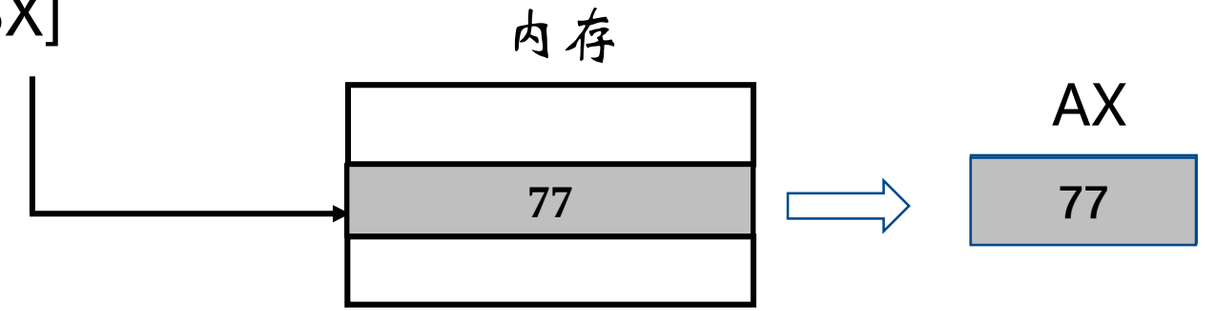
- 地址码字段给出的是操作数主存地址的地址. E=(D), S= ((D))
- 例 MOV AX, I[200H]



- 特点:
  - 解决了直接寻址方式下地址字段的位数限制访存范围大小的问题
  - 获得数据要访问主存2次，指令执行速度太慢

5 寄存器间接寻址

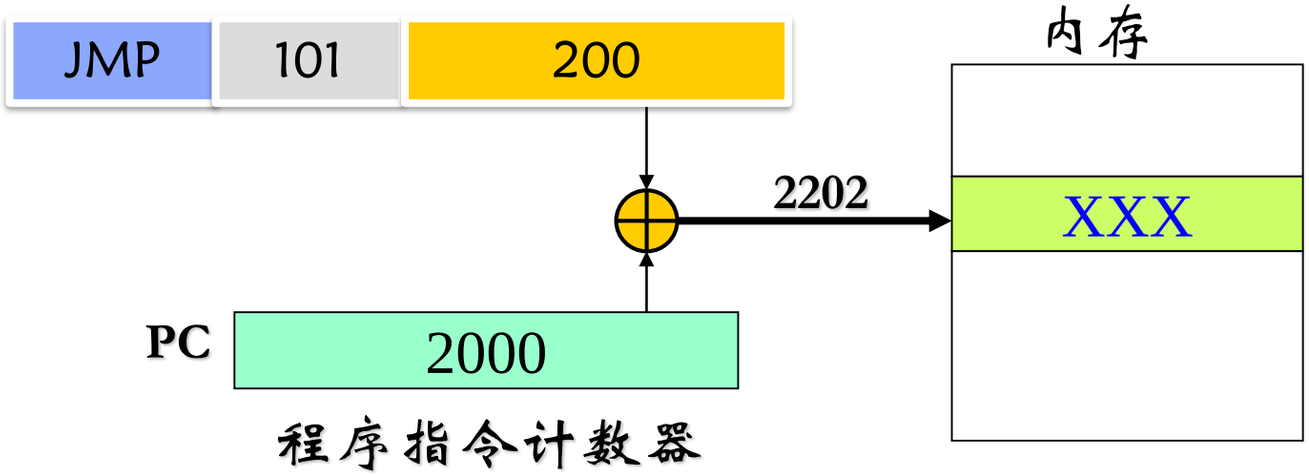
- 地址码字段给出的是寄存器编号R. E=(R), S= ((R))
- 例 MOV AX, [BX]



- 特点:
  - 解决了直接寻址方式下地址字段的位数限制访存范围大小的问题
  - 获得数据只需访问主存1次

6 相对寻址

- E=D + (PC)，D为指令中地址字段的值



- 特点：

◆可节省指令中的地址位数，便于程序在内存中成块移动

◆ 注意PC的改变对计算E的影响，如 本例中E = 200 + 2000 + 2

6 相对寻址

例 某计算机采用双字节长指令,内存基于字节寻址,指令中的数据采用补码表示,且PC的值在取指阶段完成修改。

1)若某采用相对寻址指令的当前地址为2003H,且要求转移后的目标地址为200AH,则该指令形式地址字段的值为多少?

2)若某采用相对寻址的指令的当前地址为2008H,且要求转移后的目标地址为2001H,则该指令的形式地址字段的值为多少?

解: 1)200AH – (2003H +2) = 5 (0000 0101)

2)2001H – (2008H +2) = -9 (1111 0111 即F7H)

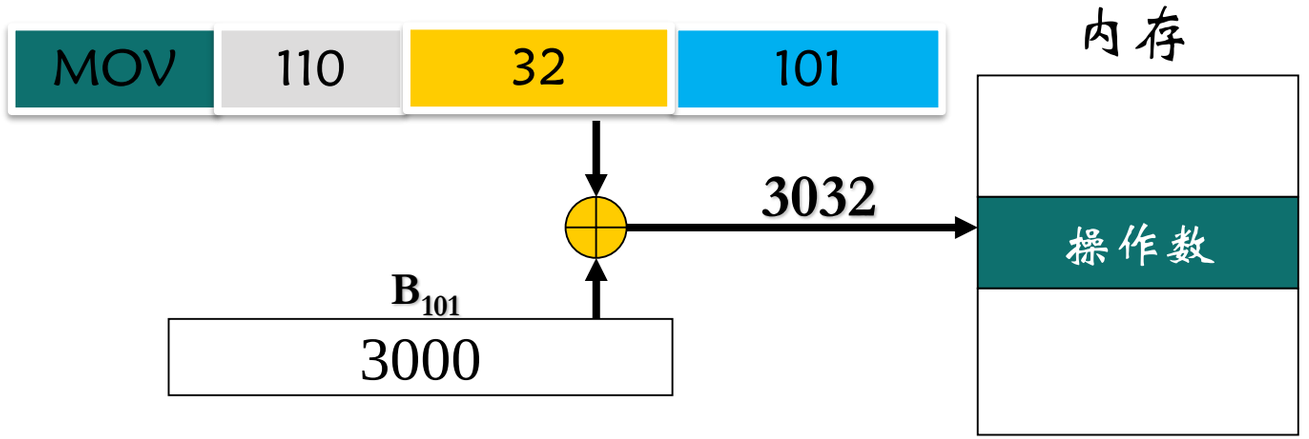
若计算机字长32位, 且PC的值在取指阶段修改,情况如何?

7 基址寻址

- 指定一个基址寄存器B，与本指令地址无关

  E= D + (B)，D为指令中地址字段的值

- MOV AX, 32[B]



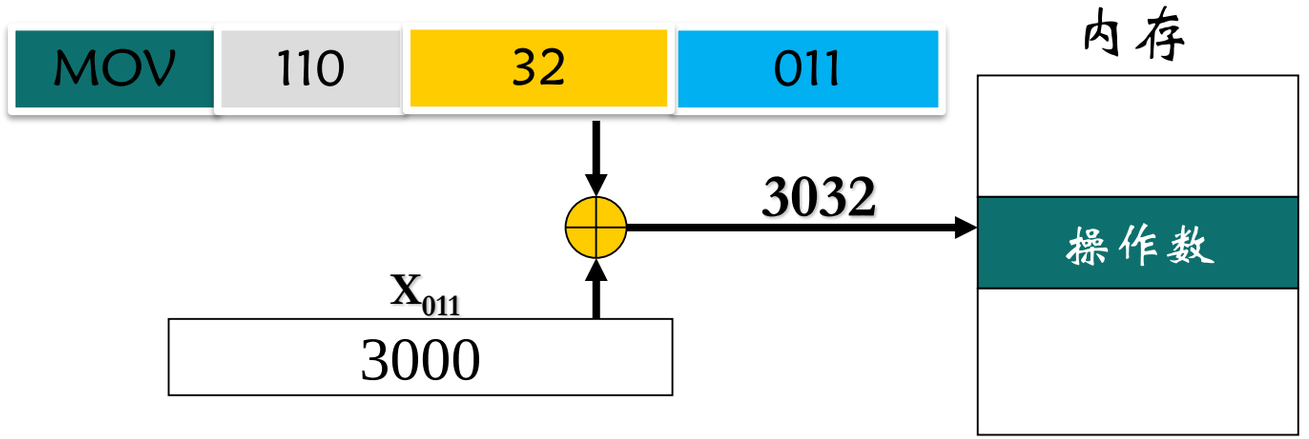
- 特点：
  - 使用基址寄存器可以访问更大的主存空间
  - 对某一程序而言，基址值设定后不变，故要访问不同数据需修改D

8 变址寻址

- 指定一个变址寄存器X，与本指令地址无关，内容可随要求改变，

  E= D + (X)，D为指令中地址字段的值

- MOV AX, 32[SI]　　SI,DI 都称为变址寄存器



- 特点：
  - ◆ 不改变指令即可改变数据的有效地址，可在循环中使用
  - ◆ 在字符串处理，向量运算等等成批数据处理中非常有用

9 数据寻址方式总结

立即寻址	S=D	快，便于程序设计，赋初值，初值大小受限
寄存器寻址	S=R	快，便于程序设计，不能访问主存
直接寻址	E=D	慢，便于程序设计，提供访存，范围受限
间接寻址	E=(D)	很慢，解决直接寻址访存范围受限的问题
寄存器间接	E=(R)	慢，便于程序设计，提供访存，范围增大
相对寻址	E=(PC)+D	慢，提供访存，不能在循环中使用
变址寻址	E=(X)+D	慢，便于程序设计，提供访存，可在循环中用
基址寻址	E=(B)+D	慢，提供更大的范围的访存能力，不能在循环中用

使用寻址方式的好处： 有利于缩短指令字长、方便程序设计、扩展访存空间！