第四章：存储器寻址，判断寻址方式，判断物理地址，找物理地址。指令系统，数据传送，堆栈过程明白，取地址，lea，lds。逻辑运算指令，按位进行逻辑运算，八个例子，填空。程序控制，带条件，无条件。

指令系统

一些指令：

- 双操作数指令，单操作数指令，无操作数指令

寻址方式

寻址方式

存储器既可以用来存放数据，又可以存放指令。

数据地址：当某个存储单元存放的是数据的时候，该存储单元序号的就是此数据的地址；

指令地址：当某个存储单元存放的是指令的时候，该存储单元序号的就是此指令的地址。

形成数据地址或者指令地址的方式，就被称为寻址方式。

寻址方式分类

数据寻址方式

指令寻址方式

数据寻址方式

有：立即数寻址方式、寄存器寻址方式、存储器寻址方式、I/O端口寻址

操作数包含在指令中：即指令的操作数部分就包含着操作数本身。
MOV AX, 1234
ADD AL, 2

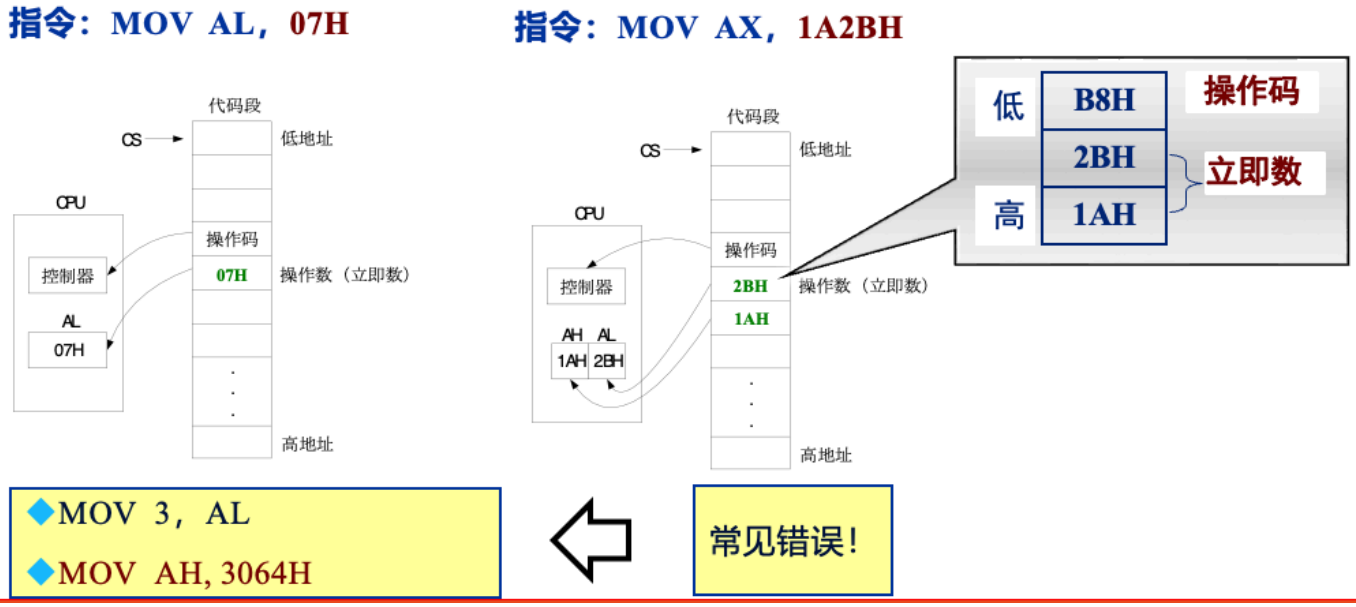
操作数包含在CPU内部寄存器中：指令中的操作数是CPU内部的某一个寄存器
MOV DS, AX

操作数在内存的数据区中：指令中的操作数包含着此操作数的地址
MOV AX, [2000]
MOV buffer[SI], AX

操作数在I/O端口寄存器中：指令中的操作数包含着此操作数所在端口地址
IN AL, n; n是端口号

立即寻址

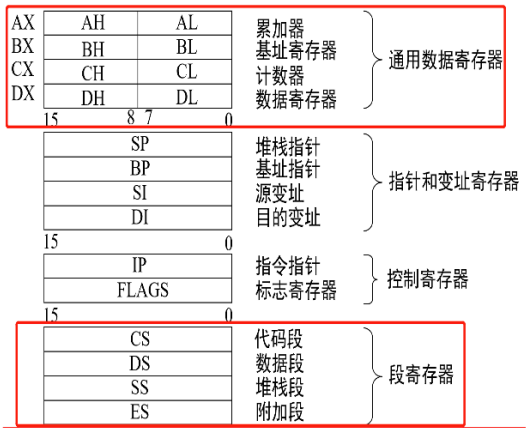
- **数据寻址方式**
  - **立即寻址（Immediate Addressing）**
    - 操作数直接存放在指令中，紧跟在操作码之后，作为指令的一部分，存放在代码段里，这种操作数称为**立即数**。常用来给寄存器赋值。



寄存器寻址

4.2 8086寻址方式

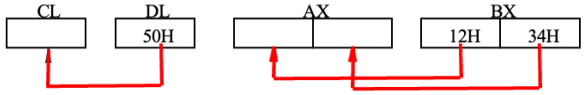
- **数据寻址方式**
  - **寄存器寻址（Register Addressing）**
    - 操作数存放在CPU的内部寄存器
    - 寄存器名表示其内容（操作数）
  - 8位寄存器：
    - AH、AL、BH、BL、CH、CL、DH、DL
  - 16位寄存器：
    - AX、BX、CX、DX、SI、DI、BP、SP
  - 4个段寄存器：
    - CS、DS、SS、ES



4.2 8086寻址方式

- **数据寻址方式**
  - **寄存器寻址（Register Addressing）**

【例】MOV CL，DL；　（DL）=50H
MOV AX，BX；　（BX=1234H）目的、源操作数采用寄存器寻址



▲**由于操作数在寄存器中**，指令执行时，操作就在CPU的内部进行，不需要访问存储器来取得操作数，因而**执行速度快。**

▲ 寄存器符号比内存地址短，汇编后**机器码长度最短。**

▲ 寄存器寻址方式既可用于DST，也可用于SRC，还可以两者都用寄存器寻址。

★ 当指令中的 DST和 SRC均为寄存器时， 必须采用同样长度的寄存器；

★ 两个操作数不能同时为段寄存器；

★ 目的操作数不能是代码段寄存器。

存储器寻址

4.2 8086寻址方式

中山大學 SUN YAT-SEN UNIVERSITY

- **数据寻址方式**
  - **存储器寻址（Memory Addressing）**

◆ 操作数存放在存储器中，CPU取出指令后，为了获得操作数（对于源操作数）或操作数的存放地址（对于目的操作数）还要再次访问存储器。

◆ 指令中以**逻辑地址（有效地址EA）**表示操作数存放的位置，可能存放在存储器的任意一个逻辑段中，CPU必须计算出操作数的**物理地址**才能完成存储单元的读、写。

◆ 存储器寻址既可用于源操作数，也可用于目的操作数，但**两者不能同时使用**。

◆ 用方括号对 **[]** 表示存储器寻址。

存储器操作数具有三个属性：

- **段 地 址**：操作数所在的逻辑段的段地址。
- **偏移地址**：相对段地址的偏移量。
- **数据类型**：操作数是一个字节/字。

指令中，不允许使用AX、CX、DX等作偏移地址

- **逻辑地址LA**：段地址:偏移地址
- **物理地址PA**：段地址×16 + 偏移地址
- 有效地址EA：偏移地址

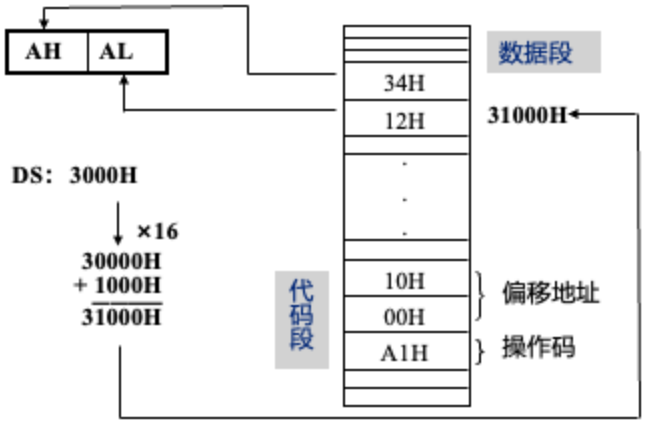
$$EA = \begin{pmatrix} BX \\ BP \end{pmatrix} + \begin{pmatrix} SI \\ DI \end{pmatrix} + DISP$$

存放在指令中的一个8位、16或32位的数，但它不是立即数，而是一个地址

直接寻址

- **存储器寻址：①直接寻址**

指令MOV AX，[1000H]的示意图



执行结果
(AX)=3412H

◆根据指令中给出的有效地址，得到存储单元的物理地址：

PA = DS×16 + 1000H = 31000H

◆把该内存单元中的内容送到AX中；

◆字在内存中占两个内存单元，低字节在前（**低地址**），高字节在后（**高地址**）；并以低字节的地址作为字的地址。

寄存器间接寻址

- **存储器寻址：②寄存器间接寻址**
  - 操作数据的偏移地址(EA)用SI、DI、BX、BP这4个寄存器之一来指定
  - 为了区别于寄存器寻址，寄存器名用“ [ ] ”括起。

$$EA=\begin{cases}BX\\BP\\SI\\DI\end{cases}$$

☆ **不同的寄存器所隐含对应的段地址不同**

- 采用**SI、DI、BX** 寄存器，数据存于**数据段**中；
- 采用 **BP** 寄存器，数据存于**堆栈段**中。

操作数的物理地址计算式为：

PA = DS *10H + SI / DI / BX

PA = SS *10H + BP

- **数据寻址方式**
  - **存储器寻址：②寄存器间接寻址**

例如：假设 DS=2000H，SI=1000H，

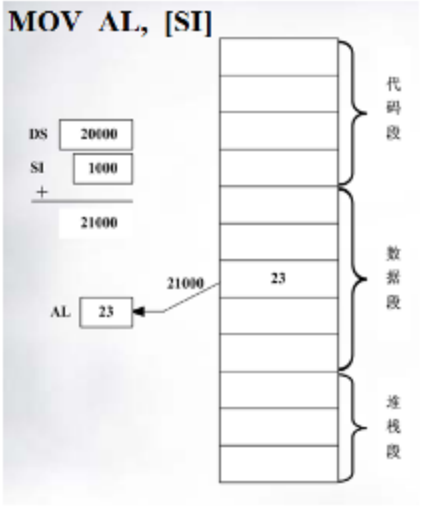
执行指令 MOV AL，[SI]

物理地址=DS*10H+SI

=2000H*10H+1000H

= 21000H

AL= (21000H) =23H



不允许使用AX、CX、DX、SP存放EA：（×）MOV AX, [CX]

源、目的操作数不能同时带方括号：（×）MOV [BX], [SI]

寄存器相对寻址

- **存储器寻址：③ 寄存器相对寻址**

☆ 寄存器相对寻址时，操作数的有效地址分为两部分：

▲ 一部分存于**寄存器**中，指令中给出该寄存器名；

▲ 另一部分以**偏移量**的方式直接在指令中给出。

$$EA=\begin{cases}BX\\BP\\SI\\DI\end{cases}+8位或16位偏移量$$

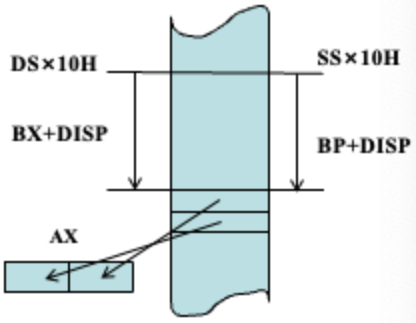
有效地址EA为基址寄存器（BX，BP）的内容加偏移量。

当EA = （BX） + DISP时：

PA = （DS）×16 + EA

当EA = （BP） + DISP时：

PA = （SS）×16 + EA



- **数据寻址方式**
  - **存储器寻址：③ 寄存器相对寻址**

例如：假设 DS=2000H，SI=1000H，count=100H

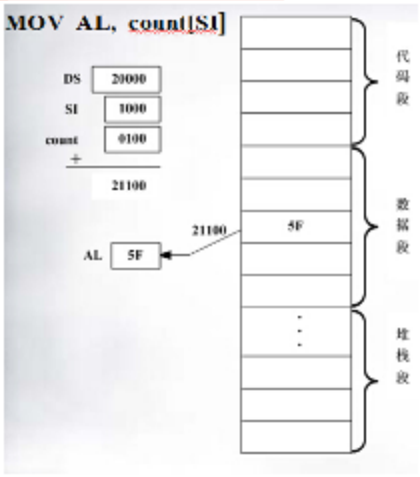
执行指令　　MOV AL, count[SI]

物理地址=DS*10H+SI+count

=2000H*10H+1000H+100H

= 21100H

AL= (21100H) =5FH



◆ IBM汇编允许用三种形式表示相对寻址：

```
MOV AX, [BX]+6    ; 标准格式
MOV AX, 6[BX]     ; 先写偏移值
MOV AX, [BX+6]    ; 偏移值写在括号内
```

基址加变址寻址

- **存储器寻址：④ 基址加变址寻址**

总结：有效地址EA为基址寄存器（BX，BP）加变址寄存器（SI，DI）的内容

$$EA = \begin{pmatrix} BX \\ BP \end{pmatrix} + \begin{pmatrix} SI \\ DI \end{pmatrix}$$

PA = (DS) ×16 + EA，当基址寄存器为BX时

PA = (SS) ×16 + EA，当基址寄存器为BP时

```
MOV [BX+DI], AX     ; 目的操作数在数据段   EA = (BX) + (DI)
                                           PA = (DS) ×16 + EA
MOV DX, [BX][SI]    ; 源操作数在数据段     EA = (BX) + (SI)
                                           PA = (DS) ×16 + EA
MOV [BP][DI], BX    ; 目的操作数在堆栈段   EA = (BP) + (DI)
                                           PA = (SS) ×16 + EA
MOV CX, [BP+SI]     ; 源操作数在堆栈段     EA = (BP) + (SI)
                                           PA = (SS) ×16 + EA
```

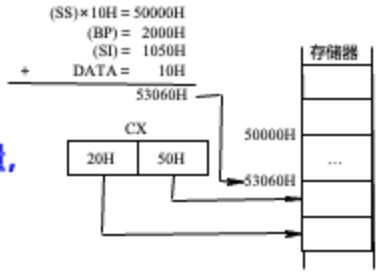
相对基址加变址寻址

- **存储器寻址：⑤相对基址加变址寻址**

【例】 MOV 10H [BP] [SI], CX

★ 相对基址变址寻址的寻址过程：

SS =5000H；BP =2000H； SI =1050H



① 由指令中给出寄存器名、寄存器内容及偏移量，得到存储单元的物理地址：

SS ×10H ＋BP＋SI＋10H＝53060H

② 把寄存器CX中的内容送到该地址内存单元中。

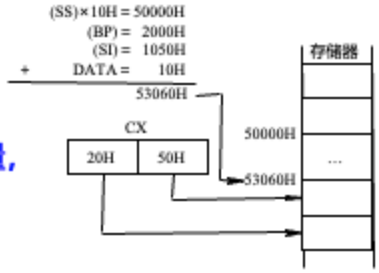
I/O端口寻址

- **存储器寻址：⑤相对基址加变址寻址**

【例】 MOV 10H [BP] [SI], CX

★ 相对基址变址寻址的寻址过程：

SS =5000H；BP =2000H； SI =1050H



① 由指令中给出寄存器名、寄存器内容及偏移量，得到存储单元的物理地址：

SS ×10H ＋BP＋SI＋10H＝53060H

② 把寄存器CX中的内容送到该地址内存单元中。

8086指令系统

- **数据传送指令**

1. 传送字或字节指令 MOV

格式：MOV dst, src ;

功能： src → dst

MOV指令的特点

➢这里的传送实际上是**复制**，把src的内容复制带dst，src内容不变；

➢src和dst必须类型**一致**（都是8位或者16位）；

➢**dst不能是立即数；**

➢两个操作数**不能都是存储器操作数；**

➢两个操作数**不能都是段寄存器操作数**；

➢**src是立即数时**，dst不能是段寄存器，必须通过通用寄存器作中介；

(1) 立即数传送给通用寄存器或存储器

```
MOV AL,12H
; 8位数据传送，将12H传送到寄存器AL中
MOV AX,3456H
; 16位数据传送，将3456H传送到寄存器AX中
```

(2) 通用寄存器之间相互传送

```
MOV AX,BX
; 16位数据传送，将BX中的数据传送到寄存器AX中
MOV CL,BH
; 8位数据传送，将BH中的数据传送到寄存器CL中
```

(3) 通用寄存器和存储器之间相互传送

```
MOV AX, [BX]
; 16位数据传送，将BX指定的连续2个字节中的数据传送到AX中
MOV [SI], DH
; 8位数据传送，将DH中的数据传送到由SI指定的内存单元中
```

(4) 段寄存器与通用寄存器、存储器之间的相互传送

```
MOV DS, AX
MOV ES, [SI]
MOV [DI], SS
```

2. 堆栈操作指令

入栈格式：**PUSH** src ;　　src: { MEM，REG，SEG }

功能：将src指示的字数据压入当前栈顶

出栈格式：**POP** dst ;　　dst: { MEM，REG，SEG }

功能：将当前栈顶的字弹出到 dst 中

注意几点：

（1）因为堆栈指针**SP**总是指向已经存入数据的**栈顶**（不是空单元），所以PUSH指令是先将（SP）减2，后将内容压栈（即先修改SP使之指向空单元，后压入数据），而POP是先从栈顶弹出一个字，后将堆栈指针SP加2。

（2）因为SP总是指向栈顶，而用PUSH和POP指令存取数时都是在栈顶进行的，所以堆栈是“先进后出”或叫“后进先出”的。栈底在高地址，堆栈是从高地址向低地址延伸的，所以栈底就是最初的栈顶。

（3）用PUSH指令和POP指令时只能按字访问堆栈，不能按字节访问堆栈。