第3章　寻址方式

罗文坚
中国科大 计算机学院

http://staff.ustc.edu.cn/~wjluo/mcps/

1

本章内容

- 指令格式的回顾
- 数据寻址方式
- 程序存储器寻址方式
- 堆栈存储器寻址方式
- I/O端口寻址

2

指令的组成

- 指令通常应提供的信息
  - 做什么操作、操作数从哪里来、操作结果放在哪里
  - 对于调用和转移指令，还要涉及转移或调用地址的提供方式
- 指令的组成

操作码	操作数	……	操作数

  - 操作码字段（Field）：标明计算机要执行什么操作。
    - 相对简单：对每一种操作指定相应的二进制代码即可。
  - 操作数字段：指出指令在执行过程中所需要的操作数（值为多少 或者放在什么地方或者控制转移到什么地方），以及操作结果送到哪里。
    - 比较复杂：寻址方式！

3

寻址方式

- *定义1*：指令中操作数的表示方式。
- *定义2*：规定如何对地址字段作出解释以找到操作数。
- 程序转移时需提供转移地址，这与提供操作数地址在方法上没有本质区别，因此也归入寻址方式的范畴。
- 一个指令系统能够提供哪些寻址方式，能否为编制程序提供方便，这是指令系统设计的关键。
- 高效开发微处理器软件，需通晓每条指令采用的寻址方式。

4

本章内容

- 指令格式的回顾
- 数据寻址方式
- 程序存储器寻址方式
- 堆栈存储器寻址方式
- I/O端口寻址

5

数据寻址方式

1. 寄存器寻址
2. 立即寻址
3. 直接数据寻址
4. 寄存器间接寻址
5. 基址加变址寻址
6. 寄存器相对寻址
7. 相对基址加变址寻址
8. 比例变址寻址（80386及更高档）
9. RIP相对寻址（Pentium 4和Core 2的64位模式）

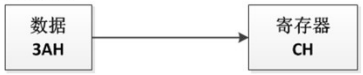
6

以MOV指令为例

- 注意：
  1. MOV指令是把源操作数复制到目的操作数，源操作数并不改变。
  2. 除MOVS指令外，任何其他指令都不允许储存器到存储器的传送。

寄存器寻址 MOV AX, BX

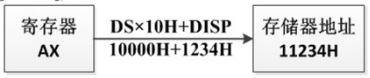
立即寻址 MOV CH, 3AH



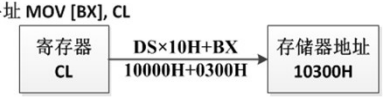
7

以MOV指令为例

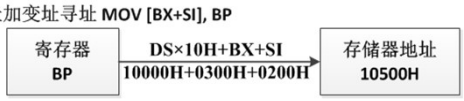
直接寻址 MOV [1234H], AX



寄存器间接寻址 MOV [BX], CL



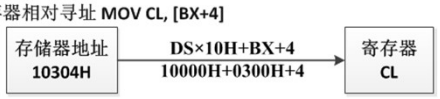
基址加变址寻址 MOV [BX+SI], BP



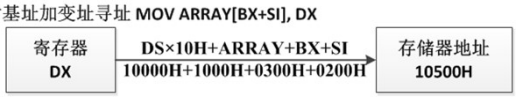
8

以MOV指令为例

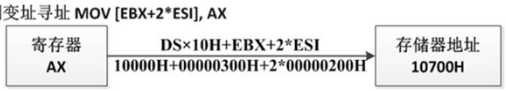
寄存器相对寻址 MOV CL, [BX+4]



相对基址加变址寻址 MOV ARRAY[BX+SI], DX



比例变址寻址 MOV [EBX+2*ESI], AX



9

寄存器寻址

- 寄存器寻址：操作数在寄存器中。
  - 只要记住寄存器名，就很容易使用。
  - 8位、16位、32位、64位寄存器
- 指令中使用相同长度的寄存器。
  - 正确：MOV AX, BX
  - 错误：MOV AX, BL
  - 少数指令例外。
    - 例如，移位指令SHL：SHL DX，CL
- 就MOV指令而言：
  - 不允许段寄存器到段寄存器的MOV指令。
  - CS不能作为MOV指令的目的操作数。

10

Example

- 0000 8B C3 MOV AX, BX；把BX的内容复制到AX
- 0002 8A CE MOV CL, DH
- 0004 8A CD MOV CL, CH
- 0006 66 | 8B C3 MOV EAX, EBX
- 0009 66 | 8B D8 MOV EBX, EAX
- 000C 66 | 8B C8 MOV ECX, EAX
- 000F 66 | 8B D0 MOV EDX, EAX
- 0012 8C C8 MOV AX, CS
- 0014 8E D8 MOV DS, AX；CS→DS分两步实现
- 0016 8E C8 MOV CS, AX；编译通过，运行有问题

11

立即寻址

- 立即数：在存储器中，数据紧接着放在操作码后面。
- 立即寻址可操作字节、字数据、双字数据（32位微处理器）、64位数据（64位模式）。
  - MOV AX, 1234H
  - MOV EAX, 123456H
  - MOV RAX, 123456780A311200H

12

Example

- MOV BL, 44；十进制数44送入BL寄存器
- MOV AX, 44H；十六进制数44送入AX
- MOV CL, 11001110B；二进制数11001110B送入CL

- 说明：
  1. 字母H表示16进制数。如果十六进制数以字母开头，则汇编程序要求前面加0，如0F2H。
  2. 如果用撇号将ASCII码括起来，也可以表示立即数，如MOV BX，'AB'。
  3. 对于二进制数，在数字后面加字母B来表示。

13

Example

1. .MODEL TINY；选择TINY模型，把程序汇编成一个代码段。TINY程序将会被成.COM程序，适用于DOS系统
2. .CODE；指示代码段的开始
3. .STARTUP；指示程序的开始
4. MOV AX, 0
5. MOV BX, 0
6. MOV SI, AX
7. MOV DI, AX
8. MOV BP, AX
9. .EXIT；返回到DOS操作系统
10. END；程序结束

14

汇编语言的语句格式简介

- 汇编语言程序中的每条语句由4个字段组成。
  - 标号、操作码、操作数、注释

LABEL	OPCODE	OPERAND	COMMENT
DATA1	DB	23H	;定义DATA1为字节23H
	……		
START1：	MOV	AL, BL	;把BL的内容复制AL
	MOV	CX, 200	;把200装入CX
	……		

15

.LST文件

- 程序被汇编后，可以生成程序清单，即.LST文件。

```
                        .MODEL TINY          ; 选择 TINY 模型
0000                    .CODE                ; 指示代码段的开始

                        .STARTUP             ; 指示程序的开始

0100  B8 0000           MOV     AX, 0        ; 把 0000H 放入 AX
0103  B8 0000           MOV     BX, 0000H    ; 把 0000H 放入 BX
0106  B9 0000           MOV     CX, 0        ; 把 0000H 放入 CX

0109  8B F0             MOV     SI, AX       ; AX 复制到 SI
010B  8B F8             MOV     DI, AX       ; AX 复制到 DI
010D  8B E8             MOV     BP, AX       ; AX 复制到 BP

                        .EXIT                ; 返回到 DOS
                        END                  ; 文件结束
```

16

Visual C++内嵌汇编程序

- 汇编程序也可以内嵌在Visual C++程序中。

```
int MyFunction(int temp)
{
      _asm
      {
            mov  eax,temp
            add  eax,20h
            mov  temp,eax
      }
      return temp;      // return a 32-bit integer
}
```

17

直接数据寻址

- 直接数据寻址：把位移量加到默认的段基址或其他段基址上形成地址。

- 直接数据寻址有两种形式：
  - 直接寻址（Direct addressing）
  - 位移量寻址（displacement addressing）

18

直接数据寻址—直接寻址

- 直接寻址：用于存储单元与AL、AX、EAX寄存器之间传送数据。
  - 这类指令通常是3个字节长。对于80386及更高型号的微处理器，指令前面可能出现一个表示寄存器长度的前缀，从而超过3个字节。
- 例：
  - MOV AL, DATA；假定DATA是存储单元的符号地址
  - MOV AL, NUMBER；NUMBER指向数据段存储单元
  - MOV ES：[2000H]，AL

19

直接数据寻址—位移量寻址

- 位移量寻址指令：将数据从存储单元移动到寄存器（不含AL、AX、EAX）的指令，称为位移量寻址指令。
- 例如：
  - MOV CL, DS:[1234H]

```
0000 A0 1234 R          MOV AL,DS:[1234H]
0003 BA 0E 1234 R       MOV CL,DS:[1234H]
```

  - MOV DATA1, EAX
  - MOV EDI, SUM1；SUM1已定义

20

Example

- SMALL模型：允许包含一个数据段和一个代码段。

```
                        .MODEL SMALL        ;choose small model
0000                    .DATA               ;start data segment

0000 10           DATA1 DB 10H              ;place 10H into DATA1
0001 00           DATA2 DB 0                ;place 00H into DATA2
0002 0000         DATA3 DW 0                ;place 0000H into DATA3
0004 AAAA         DATA4 DW 0AAAAH           ;place AAAAH into DATA4

0000                    .CODE               ;start code segment
                        .STARTUP            ;start program

0017 A00000 R           MOV AL,DATA1        ;copy DATA1 into AL
001A 8A 26 0001 R       MOV AH,DATA2        ;copy DATA2 into AH
001E A3 0002 R          MOV DATA3,AX        ;copy AX into DATA3
0021 8B 1E 0004 R       MOV BX,DATA4        ;copy DATA4 into BX

                        .EXIT;              exit to DOS
                        END;                end program listing
```

21

寄存器间接寻址

- 寄存器间接寻址允许寻址任何存储单元的数据。
- 16位微机：BP、BX、DI、SI。
  - 例、MOV AX, [BX]，如果[DS]=0100H，[BX]=1000H，则将存储器中02000H中的内容送入AX，低地址低字节，高地址高字节。
- 80386及更高档微处理器：除了用BP、BX、DI、SI寄存器做间接寻址寄存器外，允许使用除ESP以外的任何扩展寄存器。
  - 说明：ESP归入堆栈寻址。
- 64位模式下，可以使用任意64位寄存器来保存一个64位线性地址。

22

Example

- MOV CX, [BX]；数据段
- MOV [BP], DL；使用BP、EBP时，默认段为堆栈段
- MOV [DI], BH；数据段
- MOV [DI], [BX]；错！串指令外，不允许Mem→Mem
- MOV AL, [EDX]；数据段
- MOV ECX, [EBX]；数据段
- MOV RAX, [RDX]；64位模型

23

寄存器间接寻址

- BX、DI、SI：默认段寄存器为DS
- BP：默认段寄存器为SS
  - 注意：SP
- EAX、EBX、ECX、EDX、EDI、ESI：默认段寄存器为DS
- EBP：默认段寄存器为SS
  - 注意：ESP
- 在实模式下，32位寄存器寻址存储器时，32位寄存器的内容不允许超过0000FFFFH。

24

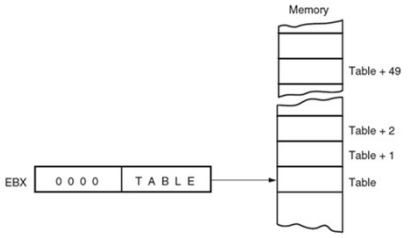
寄存器间接寻址

- 在保护模式下，只要不访问权限字节规定的段之外的存储单元，任何值都可以在用于间接寻址寄存器的32位寄存器中使用。
  - 例如：MOV EAX, [EBX]
- 可用汇编伪指令规定传送数据的长度。
  - BYTE PTR, 或WORD PTR, 或DWORD PTR，或QWORD PTR
  - 例，MOV [DI], 10H；传送字节？字？
  - 例，MOV BYTE PTR [DI], 10H；传送字节
  - 例，MOV DWORD PTR [DI], 10H；传送双字

25

寄存器简介寻址

- 寄存器间接寻址常用于引用存储系统中的数据表。



26

基址加变址寻址

- 基址加变址寻址：类似于间接寻址，用于间接地访问存储器。
- 8086~80286：
  - 基址寄存器BX或BP + 变址寄存器SI或DI。
  - 通常，基址寄存器用于保持数组的起始地址，变址寄存器用于保持数组元素的相对位置。
  - 注意：使用BP时，默认段寄存器是SS。

27

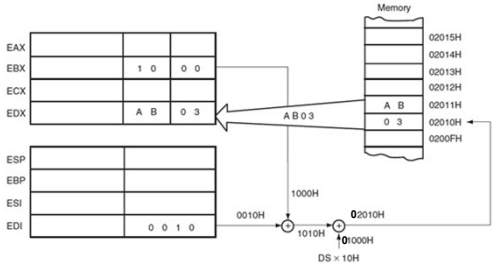
基址加变址寻址

- 80386及更高档微机
  - 允许除了ESP以外的任意两个32位扩展寄存器组合使用。
  - 例如，MOV DL, [EAX+EBX]
  - 注意：使用EBP时，默认段寄存器是SS。

28

基址加变址寻址

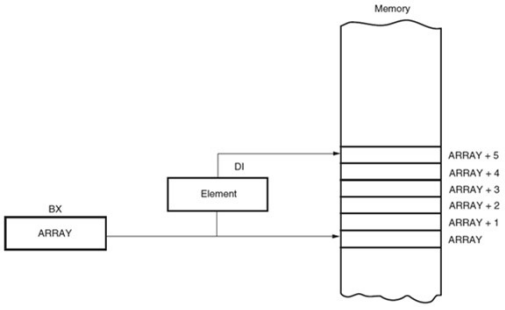
- 例，实模式下，用“基址加变址寻址”定位数据
  - BX=1000H，DI=0010H，DS=0100H



29

基址加变址寻址

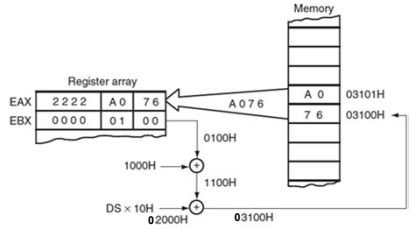
- 例，用“基址加变址寻址”定位数组数据



30

寄存器相对寻址

- 寄存器相对寻址：位移量+基址寄存器或变址寄存器。
- 例，MOV AX, [BX+1000H]
  - 已知DS=0200H



31

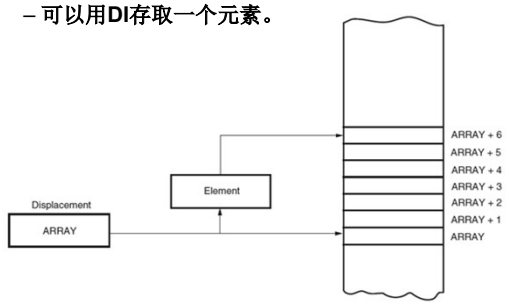
寄存器相对寻址

- 16位微机中，BX、DI或SI寻址数据段，BP寻址堆栈段。
- 80386以上微处理器中，位移量可以是32位的数字，寄存器可以是除了ESP外的任何32位寄存器。
- 位移量的形式：
  - MOV AL, [DI+2]
  - MOV AL, [SI-1]
  - MOV AL, DATA[DI]
  - MOV AL, DATA[DI+3]

32

寄存器相对寻址

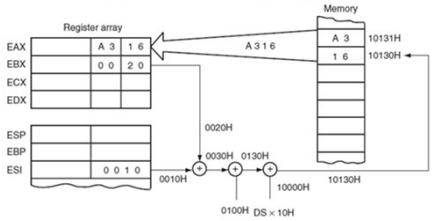
- 用寄存器相对寻址方式寻址数组数据
  - 可以用DI存取一个元素。



33

相对基址加变址寻址

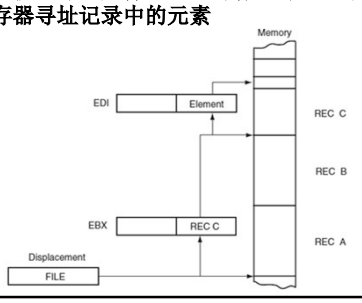
- 相对基址加变址寻址
  - 基址寄存器+变址寄存器+位移量
- 例．MOV AX, [BX+SI+100H]．假设DS=1000H



34

相对基址加变址寻址

- 用相对基址加变址寻址访问数组
  - 用位移量寻址文件，基址寄存器寻址记录，变址寄存器寻址记录中的元素



35

比例变址寻址

- 比例变址寻址是80386以上的微处理器所特有的寻址方式。
  - 有效地址EA由基址、变址、位移量和比例因子4部分组合而成。

EA=[基址寄存器]+[变址寄存器]×[比例因子]+[位移量]

  - 通用寄存器EAX、EBX、ECX、EDX、EBP、ESP、ESI和EDI都可以用作基址寄存器，用于修改内存地址（用EBP/ESP时，段寄存器为SS）。
  - 除ESP外，通用寄存器都可以用作变址寄存器。
  - 能以1、2、4、或8的比例因子对变址值进行换算，以便于对数组结构的寻址。

36

比例变址寻址

- MOV EAX, [EBX+4*ECX]
- MOV [EAX+2*EDI+100H], CX
- MOV AL, [EBP+2*EDI+2]
- MOV EAX, ARRAY[4*ECX]
- MOV RAX, [8*RDI]（64位模式）

37

RIP相对寻址

- RIP相对寻址是64位模式下，采用64位指令指针寄存器来寻址平展内存模式下的线性位置。
  - 在64位方式中，用一带符号的32位位移量进行相对寻址，由符号扩展的32位位移量的值加在RIP中的64位值以计算下一条指令的有效地址。
  - 偏移是有符号的，因此位于指令±2GB范围内的数据都可以通过这一寻址模式访问。
- 例1，CMP WORD PTR [RIP+ffff5b89H], 5a4dH
- 例2，如果RIP=1000000000H，一个32位偏移为300H，那么被访问的位置为1000000300H。

38

16位微处理器的数据寻址方式

- 立即寻址
- 寄存器寻址
- 存储器寻址
  - 物理地址PA：段基址SBA（Segment Base Address）与有效地址EA（Effecitive Address） 组合。
  - 有效地址EA：基址寄存器（BX和BP）、变址寄存器（SI和DI）、指令中给出的8位或16位位移量的组合。

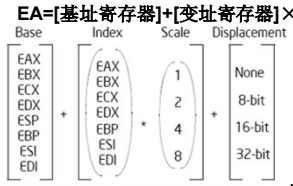
    EA=[基址寄存器]+[变址寄存器]+[位移量]
  - 不同组合方式，形成不同的寻址方式：
    1. 直接寻址
    2. 寄存器间接寻址
    3. 基址加变址寻址
    4. 寄存器相对寻址
    5. 相对基址加变址寻址

39

32位微处理器的数据寻址方式

- 立即寻址
- 寄存器寻址
- 存储器寻址
  - 物理地址PA：段基址SBA（Segment Base Address）与有效地址EA（Effective Address） 的组合。
  - 有效地址EA：基址、变址、位移量和比例因子的组合。

    EA=[基址寄存器]+[变址寄存器]×[比例因子]+[位移量]

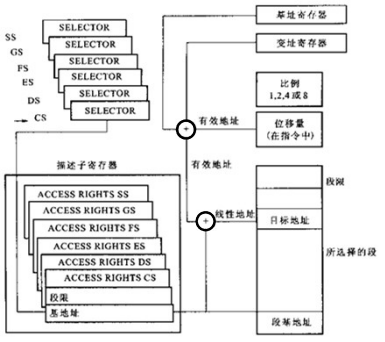


1. 直接寻址
2. 寄存器间接寻址
3. 基址加变址寻址
4. 寄存器相对寻址
5. 相对基址加变址寻址
6. 比例变址寻址

没有变址字节时，不允许ESP。

40

32位微处理器的存储器寻址方式计算



41

64位微处理器的数据寻址方式

- 立即寻址
- 寄存器寻址
- 存储器寻址
  - 物理地址PA：段基址SBA（Segment Base Address）与有效地址EA（Effective Address） 的组合。
  - 有效地址EA：基址、变址、位移量和比例因子的组合，或RIP与位移量的组合。

EA=[基址寄存器]+[变址寄存器]×[比例因子]+[位移量]

或

EA=[RIP]+[位移量]

42

64位微处理器的数据寻址方式

- 有效地址EA为基址、变址、位移量和比例因子的组合时，各部分的要求如下。
  - 位移量：一个8位、16位或32位值。
  - 基址：在一个32位(若REX.W设置，为64位)通用寄存器中的值（即基址寄存器中的值）。
  - 变址：在一个32位(若REX.W设置，为64位)通用寄存器中的值（即变址寄存器中的值）。
  - 比例系数：值2、4或8，用于与变址值相乘。
  - 在大多数情况下，基址寄存器和变址寄存器能在16个可用的通用寄存器之一中规定。

43

指令中带方括号的地址表达式遵循的规则

1. 立即数可以出现在方括号内，表示直接地址，例如[2000H]
   - 注意区别：MOV AX, 2000H 和 mov AX, [2000H]
2. 16位微处理器中，只有BX/BP，SI/DI四个寄存器可以出现在[ ]内。它们可以单独出现，也可以组合出现（只能相加），或以寄存器与常数相加的形式出现。但BX和BP寄存器不允许同时出现在一个[ ]内，SI和DI也不能同时出现。
   - 正确：[BX+SI－2]
   - 不正确：[BX－SI－2]
3. 由于方括号有相加的含义，下面几种写法等价：
   - 6[BX][SI]
   - [BX+6][SI]
   - [BX+SI+6]

44

指令中带方括号的地址表达式遵循的规则

4. 若方括号内包含BP，则蕴含使用SS来提供基地址。

   物理地址=16×SS+EA。

   - 8086 CPU中，包含BP的操作数有下列三种形式：

     DISP[BP+SI]

     DISP[BP+DI]

     DISP[BP]

     注：DISP是8位或16位的位移量

   - 允许用段超越前缀将SS修改为CS、DS或ES中的一个。

45

指令中带方括号的地址表达式遵循的规则

5. 其余情况蕴含使用DS来提供基地址。物理地址计算方法为：

   物理地址＝16×DS＋EA

   - 8086 CPU中，操作数可以有以下几种形式：

     [DISP]

     DISP[BX] 或 DISP[SI] 或 DISP[DI]

     DISP[BX+SI]

     DISP[BX+DI]

   - 同样允许用段超越前缀将DS修改为CS、ES或SS中的一个。

46

本章内容

- 指令格式的回顾
- 数据寻址方式
- 程序存储器寻址方式
- 堆栈存储器寻址方式
- I/O端口寻址

47

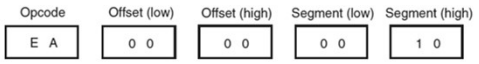
程序存储器寻址

- 用于JMP（转移）和CALL（调用）指令的程序存储器寻址方式有三种形式：
  - 直接程序存储器寻址
  - 相对程序存储器寻址
  - 间接程序存储器寻址

48

直接程序存储器寻址

- 直接程序存储器寻址：转移指令的目的地址和操作码一同存储。
- 例，程序要跳转到存储单元10000H处执行下一条指令，则地址10000H在存储器中被放在操作码的后面。



JMP [10000H]
1000H装入CS，0000H装入IP

49

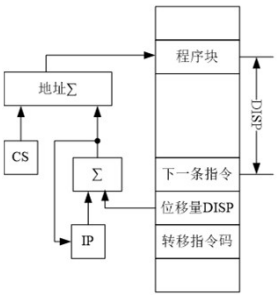
直接程序存储器寻址

- 直接转移通常是远转移（far jump），可以转移到任何存储单元执行下一条指令。
  - 远转移又称为段间转移。
- 在实模式下，远转移通过改变CS和IP的内容，可以访问1M存储器内的任何单元。
- 在保护模式操作中，远转移访问描述符表里的新的代码段描述符，允许转移到80386~Core2的整个4GB地址范围内的任何存储单元。
- 在64位模式下，CS寄存器包含一个指向描述符的指针，该描述符包含了访问权限和特权级，但不包含jump或call指令的地址。

50

相对程序存储器寻址

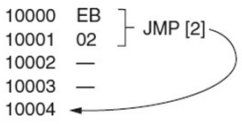
- 相对程序存储器寻址：相对于指令指针（IP/EIP）。



51

相对程序存储器寻址

- 例，JMP [2]，JMP指令跳过后面的两个存储器字节，即指令指针和2相加，就得到下一条指令的地址。



52

相对程序存储器寻址

- 通常，在16位微处理器中，JUMP指令的格式是1字节操作码加上1或2字节的位移量。
  - 1字节的位移量：短转移（short jump）
  - 2字节的位移量：近转移（near jump）
  - 短转移和近转移均属于段内转移。段内转移是指转移到当前代码段中的任何位置。
- 在80386及更高型号的微处理器中，位移量可以是4个字节（32位数），允许用相对转移到达4GB代码段内的任何位置。

53

相对程序存储器寻址

- 相对于JMP和CALL指令包含的带符号的位移量，允许向前或向后访问存储器。
- 所有的汇编程序能够自动地用位移量计算距离，并选择合适的1、2或4字节形式。
- 在16位微处理器中，如果距离太远，超出2个字节的位移量，有些汇编程序就使用直接转移。

54

间接程序存储器寻址

- 间接程序存储器寻址：利用寄存器寻址方式、存储器操作数的寻址方式求得的操作数为转移地址。
- 段内寻址的例子：
  - JMP AX；转移到当前代码段AX内容所指的位置
  - JMP NEAR PTR [BX]；地址在存储器中
  - JMP NEAR PTR [DI+2]
  - JMP TABLE [BX]
  - JMP ECX
  - JMP RDI

55

间接程序存储器寻址

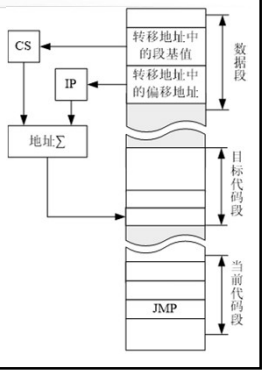
- 如果用16位寄存器存放JMP的目的地址，则是近转移。
- 例1，JMP BX。如果BX=1000H，则转移到段内偏移地址1000H处。
- 例2，JMP NEAR PTR [BX]。此时用寄存器间接寻址方式。
  - JMP [BX]的默认情况是JMP NEAR PTR [BX]。
- 例3，JMP TABEL [BX]。若BX=4，则转移到哪一个地址？

```
TABLE  DW  LOC0
       DW  LOC1
       DW  LOC2
       DW  LOC3
```

56

间接程序存储器寻址

- 段间间接转移寻址：不仅要求改变IP中的指令偏移地址，还要改变CS中的段基值。
- 例，16位微处理器中，JMP DWORD PTR [EA]



57

本章内容

- 指令格式的回顾
- 数据寻址方式
- 程序存储器寻址方式
- 堆栈存储器寻址方式
- I/O端口寻址

58

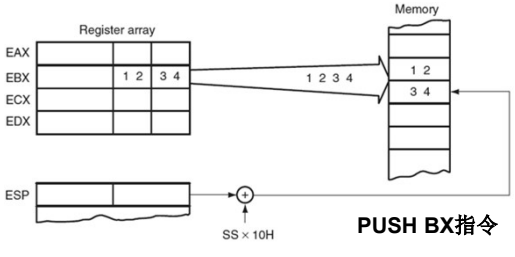
堆栈存储器寻址方式

- 堆栈在微处理器起着重要的作用，用来暂存数据，为程序保存返回地址。
- 部分相关指令：
  - PUSH / POP指令：数据入栈 / 出栈
  - CALL / RET指令：CALL用堆栈保存返回地址，RET从堆栈取出返回地址
- 堆栈存储器用两个寄存器维护：堆栈指针（SP或ESP），堆栈段寄存器（SS）。
  - 堆栈指针总是指向栈顶数据，因而是满堆栈。

59

堆栈存储器寻址方式

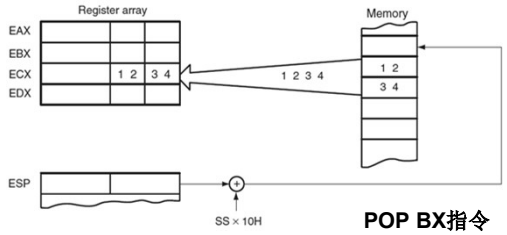
- 当字数据压入堆栈时，高8位数据放入SP-1寻址的单元，低8位数据放入SP-2寻址的单元，然后SP中的值减2。



60

堆栈存储器寻址方式

- 当字数据出栈时，低8位数据从SP寻址的单元取出，高8位数据从SP+1寻址的单元取出，然后SP中的值加2。



POP BX指令

61

堆栈存储器寻址方式

- 实模式中，SP/ESP寄存器加上SS×10H形成堆栈存储器地址。
- 保护模式中，SS寄存器的内容是用于访问一个描述符的选择子。
- 在8086~80286中，PUSH和POP总是按字（不是字节）进行数据出栈和入栈的。
- 在80386以上微处理器中，允许字或双字入栈/出栈。
- 在64位模式下，允许64位寄存器入栈/出栈。
- 思考题：写一个C语言程序，判断栈的增长方向。

62

本章内容

- 指令格式的回顾
- 数据寻址方式
- 程序存储器寻址方式
- 堆栈存储器寻址方式
- I/O端口寻址

63

I/O端口寻址

- I/O端口的范围是0000H~FFFFH（2^{16}个8位端口）。
  - 使用地址总线的低16位。
- I/O端口的寻址有2种方式。
  - 直接端口寻址方式
  - 间接端口寻址方式

64

直接端口寻址方式

- 直接端口寻址方式：在I/O指令中，端口地址以8位立即数的形式出现。
  - 仅适合于访问地址00~FFH的端口。
- 例，IN AL, 80H
  - 从地址为80H的端口读取一个字节数据到AL中。

65

间接端口寻址方式

- 间接端口寻址方式：I/O端口地址预先存放在DX寄存器中。
  - 适合于地址0000H~FFFFH的全部端口。
- 例，MOV DX, 2000H
  OUT DX, AX
  - 将AX中的16位数据发送到DX、DX+1确定的端口2000H和2001H。

66

本章小节

- 数据寻址方式
  - –立即寻址、寄存器寻址、存储器寻址
- 程序存储器寻址方式
  - –直接、相对、间接
  - –段内、段间
- 堆栈存储器寻址方式
  - –栈是递减型的满堆栈
- I/O端口寻址
  - –直接、间接

67

作业（1）

- 习题7
- 习题23
- 习题27
- 习题33
- 习题35

68

作业（2）

1. 8086 CPU中，设DS=1000H，ES=2000H，SS=3500H，SI=00A0H，DI＝0024H，BX=0100H，BP=0200H，数据段中变量名为VAL的偏移地址值为0030H，试说明下列源操作数字段的寻址方式是什么？
   1) MOV AX, [100H]
   2) MOV AX, VAL
   3) MOV AX, [BX]
   4) MOV AX, ES:[BX]
   5) MOV AX, [SI]
   6) MOV AX, [BX+10H]
   7) MOV AX, [BP]
   8) MOV AX, VAL[BP][SI]
   9) MOV AX, VAL[BX][DI]
   10) MOV AX, [BP][DI]

69

作业（3）

- 80386 CPU中，下列指令的源操作数的寻址方式是什么？
  1) MOV EAX, EBX
  2) MOV EAX, [ECX][EBX]
  3) MOV [ESI], [EDX×2]
  4) MOV EAX, [ESI×8]
  5) MOV EDX, [ESI][EBP+0FFF0000H]

70