本节主题

x86指令简介

北京大学·慕课

计算机组成

制作人：陆俊林

指令的主要类别

运算类指令

例如：加、减、乘、除，与、或、非等

传送类指令

例如：从存储器到通用寄存器，从通用寄存器到I/O接口等

控制类指令

例如：暂停处理器、清除标志位等

转移类指令

例如：无条件转移、条件转移、过程调用等

指令的运行结果

改变通用寄存器的内容
*如 ADD AX, DX

改变存储器单元的内容
*如 MOV [10H], CX

其它
……

改变标志位
*如产生进位

改变指令指针
*如 JMP [BX]

改变外设端口的内容
*如访问显示端口

指令分类举例

1. 传送指令
2. 算术运算指令
3. 逻辑运算和移位指令
4. 转移指令
5. 处理器控制指令

传送指令

- 作用：把数据或地址传送到寄存器或存储器单元中

分组	助记符	功能	操作数类型
通用数据传送指令	MOV	传送	字节/字
	PUSH	压栈	字
	POP	弹栈	字
	XCHG	交换	字节/字
累加器专用传送指令	XLAT	换码	字节
	IN	输入	字节/字
	OUT	输出	字节/字
地址传送指令	LEA	装入有效地址	字
	LDS	把指针装入寄存器和DS	4个字节
	LES	把指针装入寄存器和ES	4个字节
标志传送指令	LAHF	把标志装入AH	字节
	SAHF	把AH送标志寄存器	字节
	PUSHF	标志压栈	字
	POPF	标志弹栈	字

（1）传送指令

MOV指令（传送）

- 格式：MOV　DST, SRC
- 操作：DST←SRC

- 说明：
  - DST表示目的操作数，SRC表示源操作数
  - MOV指令把一个操作数从源传送至目的，源操作数保持不变

MOV指令和寻址方式的示例

```
MOV  EBX, 40
MOV  AL, BL
MOV  ECX, [1000H]

MOV  [DI], AX

MOV  WORD PTR[BX+SI*2+200H], 01H
```

直接给出操作数

给出存放操作数的寄存器名称

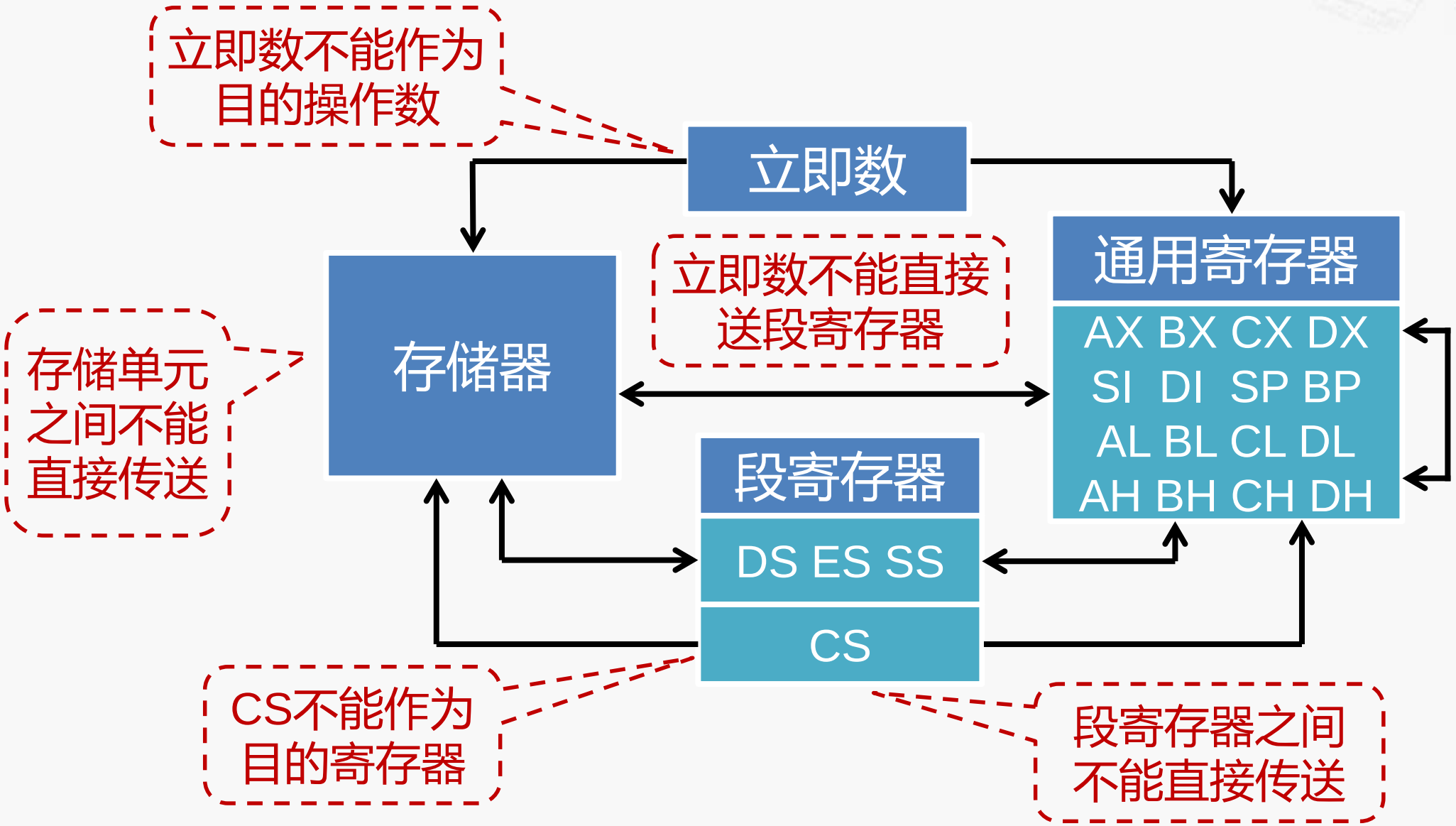
给出存放操作数的存储器地址

给出存放“存放操作数的存储器地址”的寄存器名称

给出“存放操作数的存储器地址”的计算方法

注：BYTE PTR ：字节长度标记
WORD PTR ：字长度标记
DWORD PTR ：双字长度标记

MOV指令的传送方向和限制



不同类型的MOV指令编码

7 6 5 4 3 2 1 0	7 6 5 4 3 2 1 0	76543210	76543210	76543210	76543210
1 1 0 0 0 1 1 w	mod 000 r/m	DISP-LO	DISP_HI	data	data if w=1
1 0 0 0 1 0 d w	mod reg r/m	DISP-LO	DISP_HI		
1 0 0 0 1 1 1 0	mod 0 SR r/m	DISP-LO	DISP_HI		
1 0 0 0 1 1 0 0	mod 0 SR r/m	DISP-LO	DISP_HI		
1 0 1 0 0 0 0 w	addr-lo	addr-hi			
1 0 1 0 0 0 1 w	addr-lo	addr-hi			
1 0 1 1 w r e g	data	data if w=1			

MOV指令编码示例

76543210	76543210	76543210	76543210	76543210	76543210
1011 w reg	data	data if w=1			
1011 1 000	11101110	00010000			
B8	EE	10			

立即数到寄存器

MOV AX,10EEH

76543210	76543210	76543210	76543210	76543210	76543210
100010dw	mod reg r/m	DISP-LO	DISP_HI		
10001011	000 011 11	00000100	00010000		
8B	0F	04	10		

存储器到寄存器

MOV CX,[BX+1004H]

（2）栈操作指令

PUSH指令（压栈）

- 格式：PUSH　SRC
- 说明：
  - SRC表示寄存器操作数或存储器操作数

POP指令（弹栈）

- 格式：POP　DST
- 说明：
  - DST表示寄存器操作数或存储器操作数
  - DST也可以是除CS寄存器以外的段寄存器

PUSH和POP指令操作示意

(a) 空堆栈

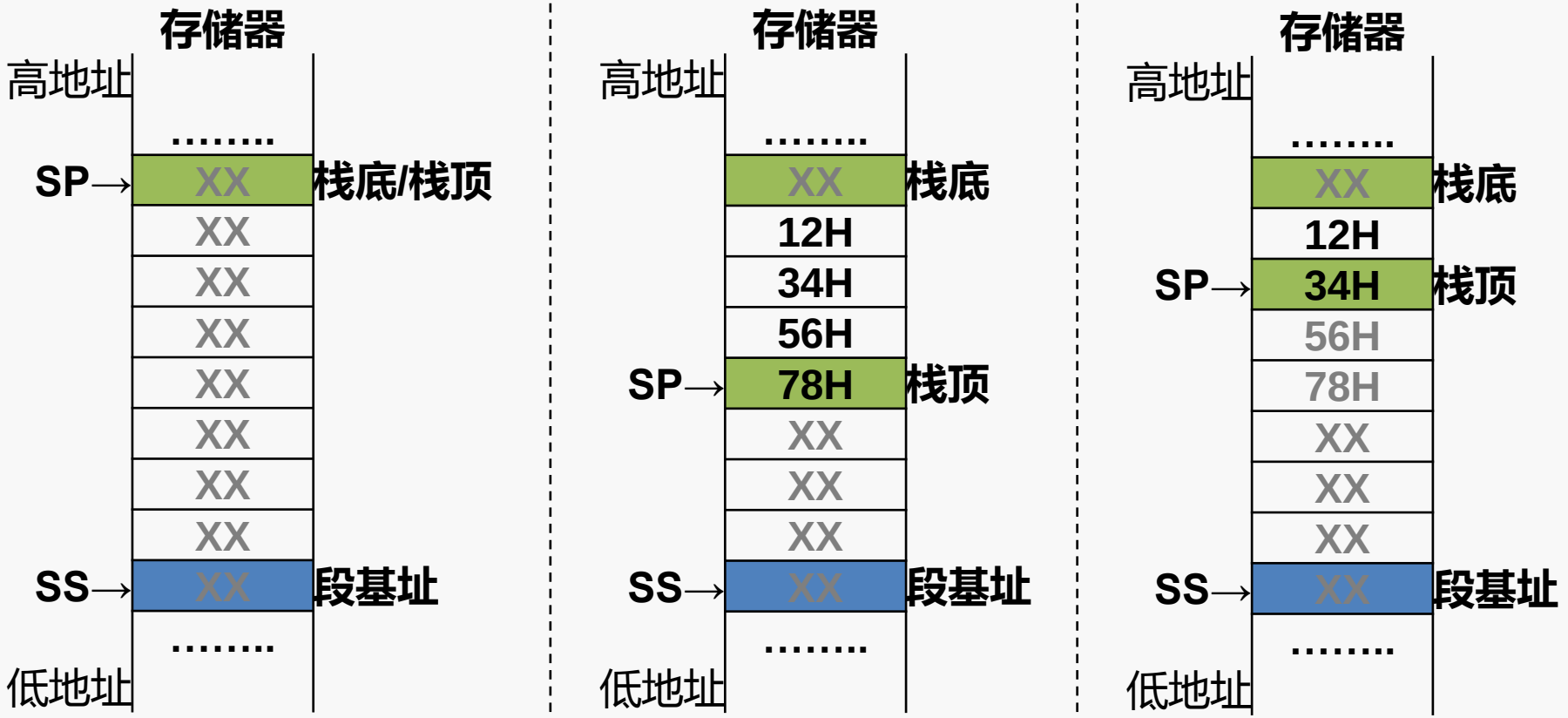
(b) 压栈

PUSH AX; AX=1234H

PUSH BX; BX=5678H

(c) 弹栈

POP CX



注：堆栈结构是所谓“向下生长”，即栈底在堆栈的高地址端。而堆栈段的段基址（由SS寄存器确定）并不是栈底。

（3）LEA指令说明

LEA指令（Load Effective Address）

- 格式：LEA　REG, SRC
- 操作：
  - 把源操作数（SRC）的有效地址（即偏移地址）装入指定寄存器（REG）
- 说明：
  - 源操作数必须是存储器操作数
  - 目的操作数必须是通用寄存器

LEA指令示例

```
LEA BX,[BX+DI*4+6H]
```

注意区别于：

```
MOV BX,[BX+DI*4+6H]
```

思考：

LEA指令的巧妙用途

指令分类举例

1. 传送指令
2. **算术运算指令**
3. 逻辑运算和移位指令
4. 转移指令
5. 处理器控制指令

算术运算指令

作用

- 完成加、减、乘、除等算术运算
- 提供运算结果调整、符号扩展等功能

操作数的限制

- 目的操作数不能是立即数或CS寄存器
- 两个操作数不能同时为存储器操作数

分组	助记符	功能
加法	ADD	加
	ADC	加（带进位）
	INC	加1
减法	SUB	减
	SBB	减（带借位）
	DEC	减1
	NEG	取补
	CMP	比较
乘法	MUL	乘（不带符号）
	IMUL	乘（带符号）
除法	DIV	除（不带符号）
	IDIV	除（带符号）

算术运算指令

- 作用
  - 完成加、减、乘、除等算术运算
  - 提供运算结果调整、符号扩展等功能

- 操作数的限制
  - 目的操作数不能是立即数或CS寄存器
  - 两个操作数不能同时为存储器操作数

分组	助记符	功能
符号扩展	CBW	将字节扩展为字
	CWD	将字扩展为双字
十进制调整	AAA	加法的ASCII调整
	DAA	加法的十进制调整
	AAS	减法的ASCII调整
	DAS	减法的十进制调整
	AAM	乘法的ASCII调整
	AAD	除法的ASCII调整

（1）加法类指令

ADD指令（加）

- 格式：ADD　DST, SRC
- 操作：DST←DST+SRC

ADC指令（带进位的加）

- 格式：ADC　DST, SRC
- 操作：DST←DST+SRC+CF

INC指令（加1）

- 格式：INC　OPR
- 操作：OPR←OPR+1

```
ADD BL, 8
ADD WORD PTR[BX], DX

ADD EAX, ECX
ADC EBX, EDX

INC CL
```

示例

INC Reg
指令长度为1字节

7	6	5	4	3	2	1	0
0	1	0	0	0	r	e	g

（2）减法类指令

SUB指令（减）

- 格式：SUB　DST, SRC
- 操作：DST←DST-SRC

SBB指令（带借位的减）

- 格式：SBB　DST, SRC
- 操作：DST←DST-SRC-CF

DEC指令（减1）

- 格式：DEC　OPR
- 操作：OPR←OPR-1

```
SUB BL, 8
SUB WORD PTR[BX], DX

SUB EAX, ECX
SBB EBX, EDX

DEC ECX
```

示例

（2）减法类指令

CMP指令（比较）

- 格式：CMP　DST, SRC
- 操作：DST - SRC
- 说明：减法操作，但不回写结果，仅影响标志位

指令分类举例

1. 传送指令
2. 算术运算指令
3. **逻辑运算和移位指令**
4. 转移指令
5. 处理器控制指令

逻辑运算和移位指令

作用

- 实现对二进制位的操作和控制
- 又称“位操作指令”

操作数的限制

- 对于单操作数指令，操作数不能是立即数
- 对于双操作数指令，限制与MOV指令相同

分组	助记符	功能
逻辑运算	NOT	逻辑非
	AND	逻辑与
	OR	逻辑或
	XOR	逻辑异或
	TEST	逻辑测试
移位	SHL	逻辑左移
	SAL	算术左移
	SHR	逻辑右移
	SAR	算术右移
循环移位	ROL	循环左移
	ROR	循环右移
	RCL	带进位循环左移
	RCR	带进位循环右移

（1）逻辑运算指令

NOT指令（逻辑非）

- 格式：NOT　OPR
- 操作：OPR按位求反，送回OPR

```
MOV   AL, 10101010B
NOT   AL
;now: AL=01010101B
```

示例

AND指令（逻辑与）

- 格式：AND　DST, SRC
- 操作：将DST和SRC的内容按位进行“与”操作，结果送到DST中

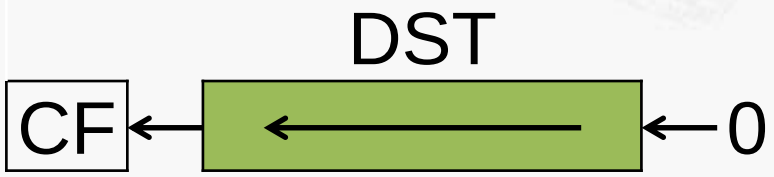
```
MOV   BL, 11111010B
AND   BL, 0FH
;now: BL=00001010B
```

示例

（2）移位指令

SHL指令（左移）

- 格式：SHL　DST, CNT
- 操作：
  - 将DST的内容按图中方式移动
  - 移动位数由CNT指定
- 说明：
  - DST可以是寄存器或存储器操作数
  - CNT可以是立即数1或CL寄存器
  - 相当于无符号数乘以2^n的运算

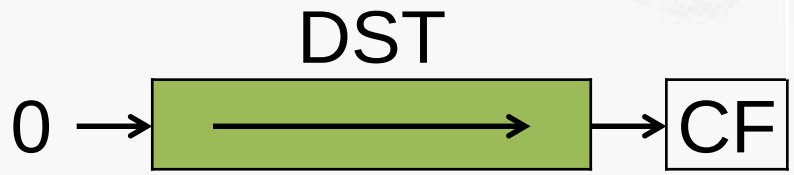


示例：设AL中有一个无符号数X，用移位指令求10X

```
MOV     AH, 0
SHL     AX, 1   ;得到2X
MOV     BX, AX
MOV     CL, 2
SHL     AX, CL ;得到8X
ADD     AX, BX ;得到10X
```

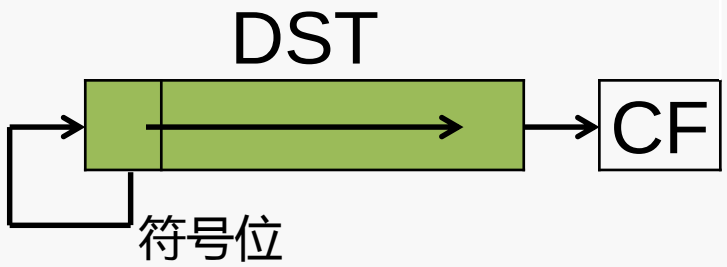
（2）移位指令

SHR指令（逻辑右移）



- 格式：SHR　DST, CNT
- 说明：相当于无符号数除以2^n的运算

SAR指令（算术右移）



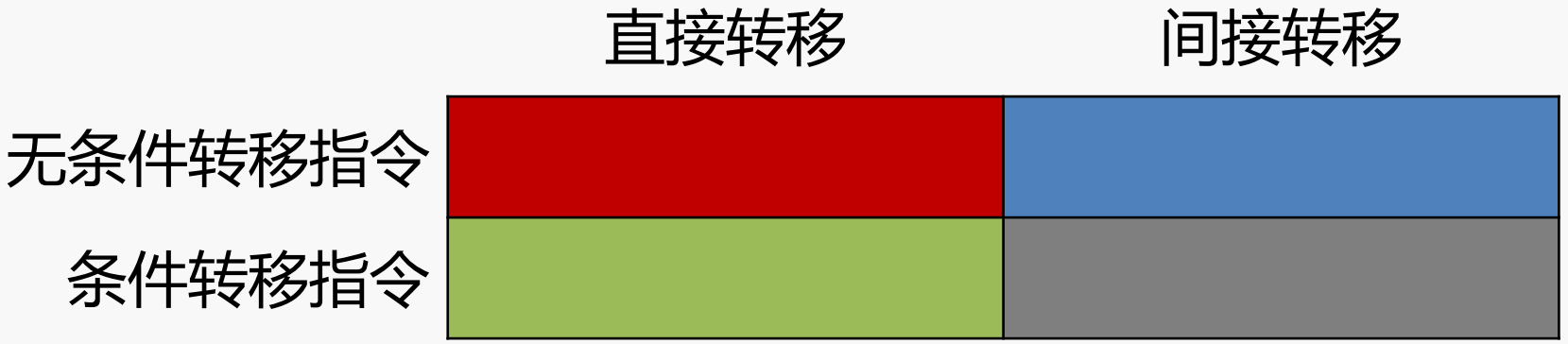
- 格式：SAR　DST, CNT
- 说明：相当于带符号数除以2^n的运算

指令分类举例

1. 传送指令
2. 算术运算指令
3. 逻辑运算和移位指令
4. **转移指令**
5. 处理器控制指令

转移指令

- 作用
  - 改变指令执行顺序
- 说明
  - 根据是否有判断条件，分为无条件转移指令和条件转移指令两大类
  - 根据转移目标地址的提供方式，可分为直接转移和间接转移两种方式



（1）无条件转移指令 - 直接转移

- 短转移：JMP　SHORT　LABEL
  - 操作：IP←IP+8位的位移量（-128~127Byte）
- 近转移：JMP　NEAR PTR　LABEL
  - 操作：IP←IP+16位的位移量（±32KByte）
- 远转移：JMP　FAR PTR　LABEL
  - 操作：IP←LABEL的偏移地址；CS ←LABEL的段基值

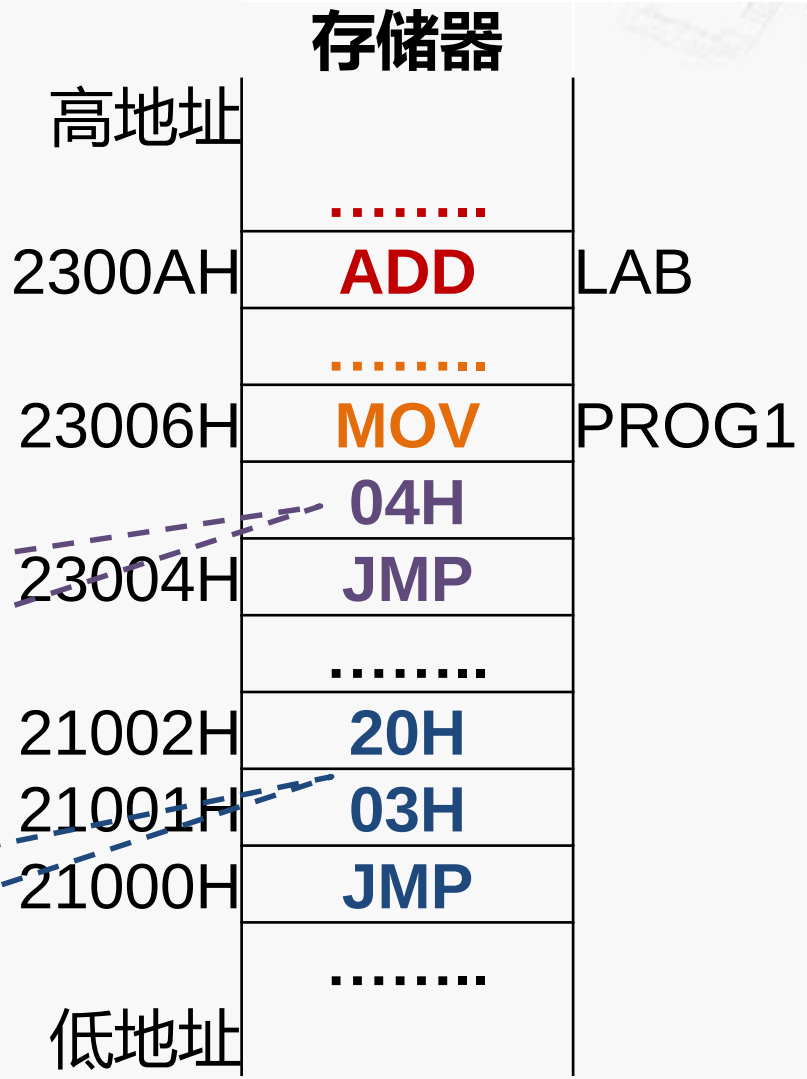
说明

1. 位移量是一个带符号数，为LABEL的偏移地址与当前EIP/IP值之差
2. 从80386开始，近转移可以使用32位的位移量

	操作码				
短转移	EB	8-bit位移量			
近转移	E9	16-bit位移量			
远转移	EA	IP	IP	CS	CS

短/近转移的执行过程

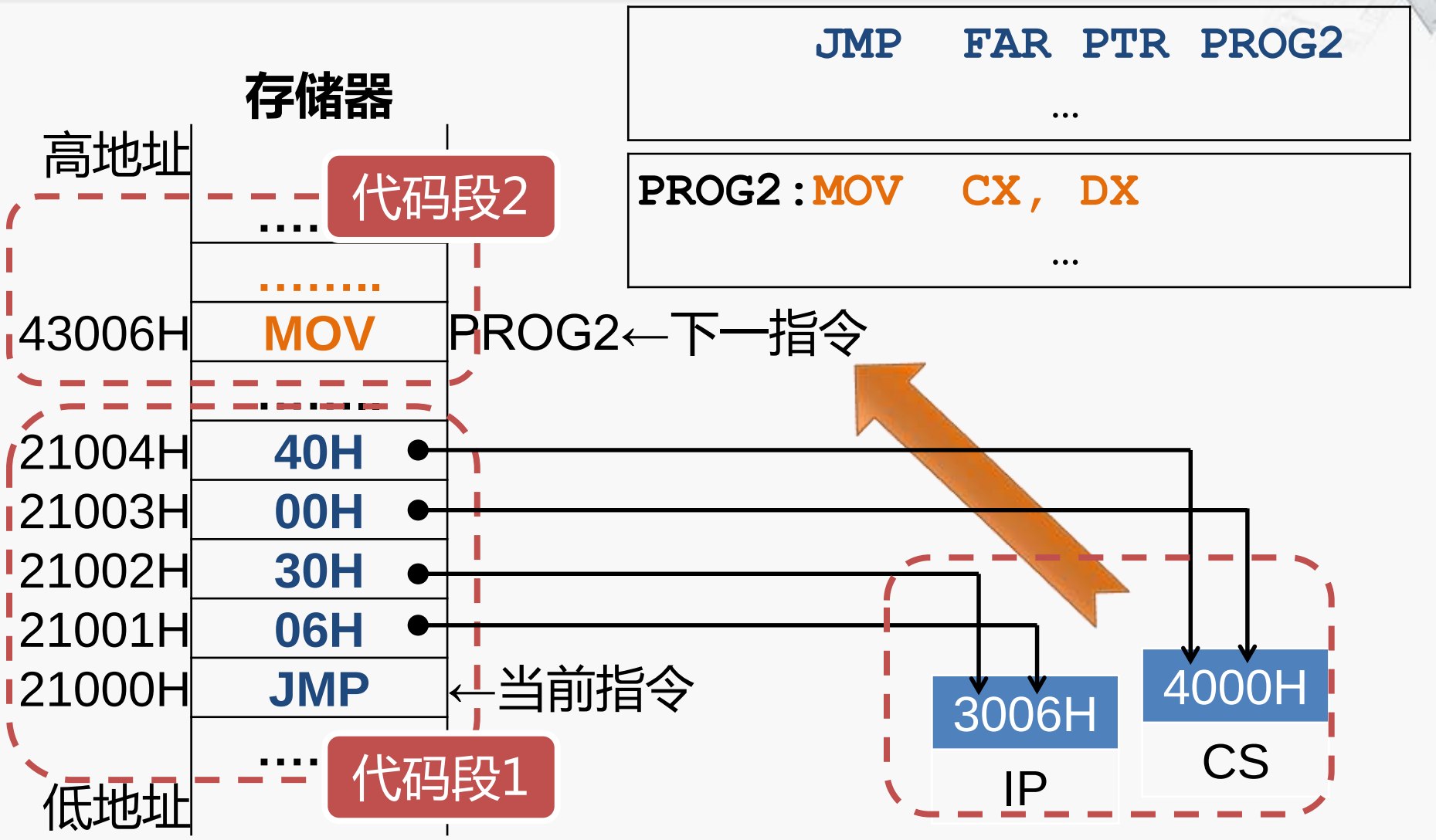
```
        JMP   NEAR PTR PROG1
                ...
        JMP   SHORT LAB
PROG1:  MOV   CX, DX
LAB:    ADD   AX, BX
                ...
```



LAB-IP
=2300AH-(23004H+2)
=04H

PROG1-IP
=23006H-(21000H+3)
=2003H

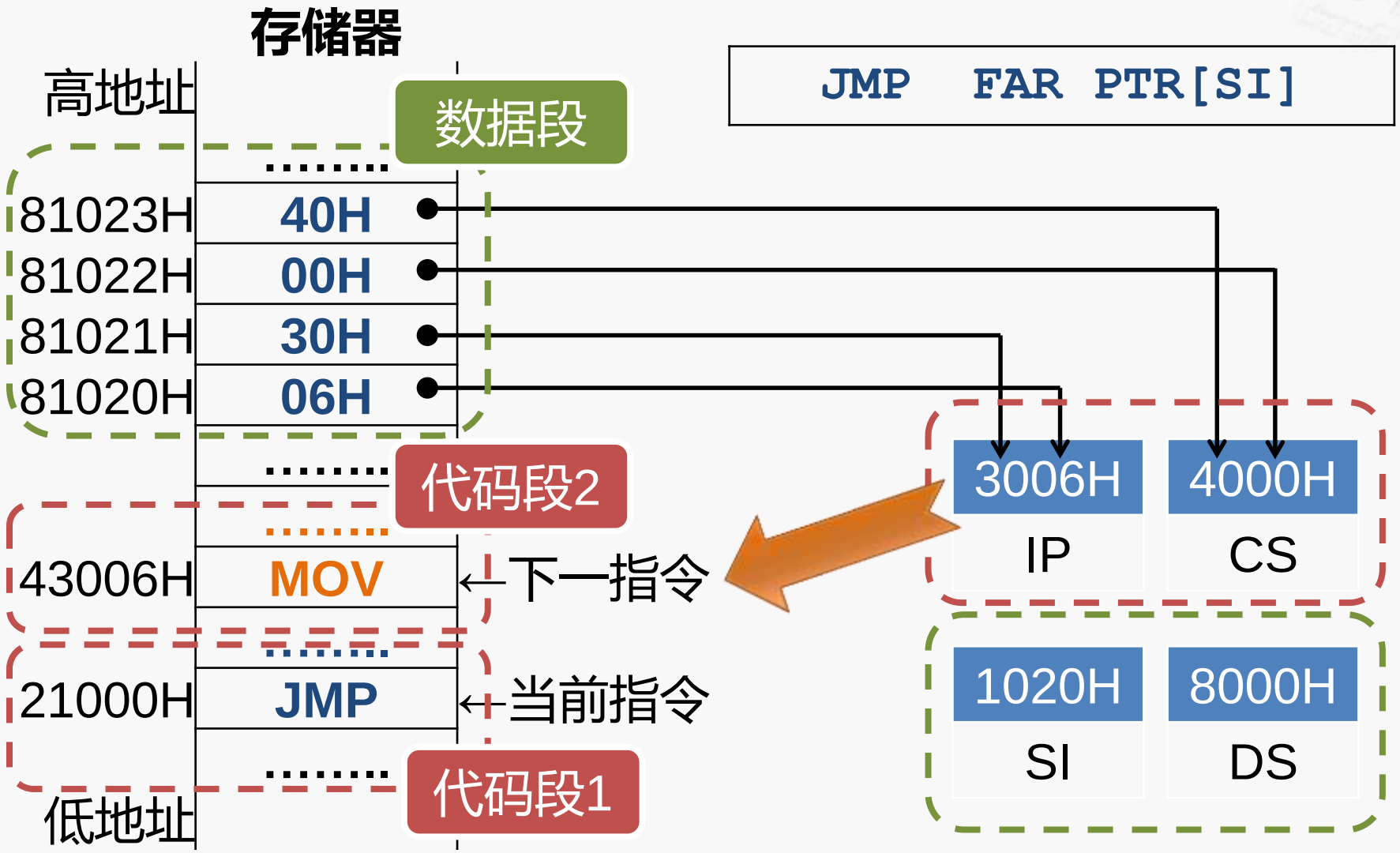
远转移的执行过程


JMP FAR PTR PROG2
...
PROG2: MOV CX, DX
...
存储器
高地址
代码段2
43006H
MOV
PROG2←下一指令
21004H
40H
21003H
00H
21002H
30H
21001H
06H
21000H
JMP
←当前指令
代码段1
低地址
3006H
IP
4000H
CS

（2）无条件转移指令 - 间接转移

- 转移目标地址在寄存器中
  - `JMP AX ; AX → IP`
  - `JMP EAX ; EAX → EIP`
- 转移目标地址在存储器中
  - `JMP [SI] ; [SI] → IP`
  - `JMP FAR PTR[SI] ; [SI] → IP, [SI+2] → CS`

间接转移示例


存储器
高地址
数据段
81023H
40H
81022H
00H
81021H
30H
81020H
06H
代码段2
43006H
MOV
下一指令
21000H
JMP
当前指令
代码段1
低地址
JMP FAR PTR[SI]
3006H
IP
4000H
CS
1020H
SI
8000H
DS

（3）条件转移指令

- 操作
  - 根据当前的状态标志位决定是否发生转移

- 说明
  - 一般在影响标志位的算术或逻辑运算指令之后
  - 8086中，所有的条件转移都是短转移
    - 同一代码段内，-128~127字节范围内
  - 从80386起，条件转移指令可以使用32位的长位移量
    - 同一代码段内，±2G字节范围

条件转移指令

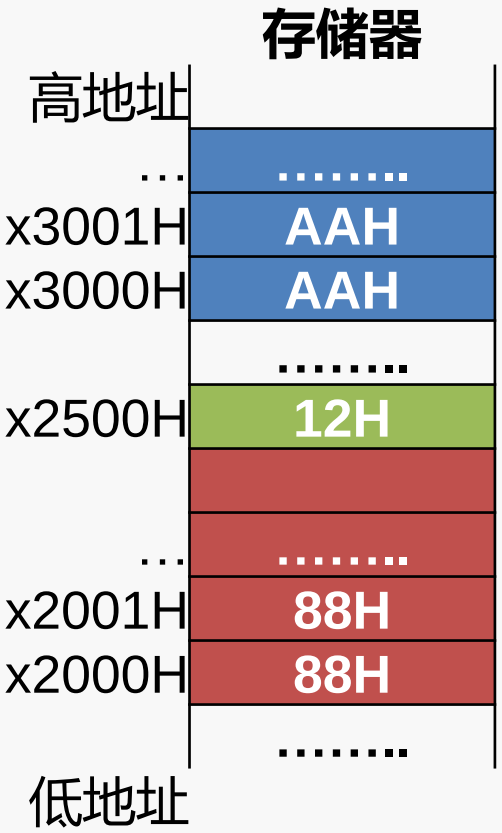
分组		格式		功能	测试条件
条件转移指令	根据某一状态标志转移	JC	LABEL	有进位时转移	CF=1
		JNC	LABEL	无进位时转移	CF=0
		JP/JPE	LABEL	奇偶位为1时转移	PF=1
		JNP/JPO	LABEL	奇偶位为0时转移	PF=0
		JZ/JE	LABEL	为零/相等时转移	ZF=1
		JNZ/JNE	LABEL	不为零/不相等时转移	ZF=0
		JS	LABEL	负数时转移	SF=1
		JNS	LABEL	正数时转移	SF=0
		JO	LABEL	溢出时转移	OF=1
		JNO	LABEL	无溢出时转移	OF=0

条件转移指令

分组		格式	功能	测试条件
条件转移指令	对无符号数	JB/JNAE LABEL	低于/不高于等于时转移	CF=1
		JNB/JAE LABEL	不低于/高于等于时转移	CF=0
		JA/JNBE LABEL	高于/不低于等于时转移	CF=0且ZF=0
		JNA/JBE LABEL	不高于/低于等于时转移	CF=1或ZF=1
	对有符号数	JL/JNGE LABEL	小于/不大于等于时转移	SF≠OF
		JNL/JGE LABEL	不小于/大于等于时转移	SF=OF
		JG/JNLE LABEL	大于/不小于等于时转移	ZF=0且SF=OF
		JNG/JLE LABEL	不大于/小于等于时转移	ZF=1或SF≠OF

程序示例

计算存储器中[2000H]和[3000H]起始的两个数之和，数的长度存放在[2500H]字节单元




```
        MOV   CL, [2500H] ;
        MOV   SI, 2000H
        MOV   DI, 3000H
        CLC   ; 将标志位CF清零
LOOP1:  MOV   AX, [SI]
        ADC   AX, [DI]
        MOV   [SI], AX
        INC   SI
        INC   SI    ; 可否使用“ADD SI，2”
        INC   DI
        INC   DI
        DEC   CL
        JNZ   LOOP1 ; 循环执行[2500]次
        MOV   AX, 0H
        ADC   AX, 0H
        MOV   [SI], AX
```

指令分类举例

1. 传送指令
2. 算术运算指令
3. 逻辑运算和移位指令
4. 转移指令
5. **处理器控制指令**

处理器控制指令

- 作用
  - 控制CPU的功能
  - 对标志位进行操作

分组	格式	功能
标志操作指令	STC	把进位标志CF置1
	CLC	把进位标志CF清0
	CMC	把进位标志CF取反
	STD	把方向标志DF置1
	CLD	把方向标志DF清0
	STI	把中断标志IF置1
	CLI	把中断标志IF清0
外同步指令	HLT	暂停
	WAIT	等待
	ESC	交权
	LOCK	封锁总线（指令前缀）
空操作	NOP	空操作

本节小结

x86指令简介

北京大学·慕课
计算机组成
制作人：陆俊林