

汇编语言程序设计

逻辑运算指令



位操作类指令

- 计算机中最基本的数据单位是二进制位
- 针对二进制位进行操作、实现位控制的指令
 - ▶ 逻辑运算指令
 - ▶ 移位指令
 - ▶ 循环移位指令
 - ▶
- 进行一位或若干位处理，采用位操作类指令



逻辑运算指令

- 逻辑与指令 **AND**
- 逻辑或指令 **OR**
- 逻辑非指令 **NOT**
- 逻辑异或指令 **XOR**
- 测试指令 **TEST**

除**NOT**指令不影响标志外，其他逻辑指令

▶ 使**OF=CF=0**

▶ 根据结果按定义影响**ZF**、**SF**和**PF**



逻辑与指令AND

➤ 逻辑与(逻辑乘)运算规则

▶ 两位都是逻辑**1**，则结果是**1**；否则，结果是**0**

➤ 逻辑与指令**AND**

▶ 按位进行逻辑与，结果返回目的操作数

AND reg,imm/reg/mem

;reg←reg ∧ imm/reg/mem

AND mem,imm/reg

;mem←mem ∧ imm/reg

	01000101
^	00110001
<hr/>	
	00000001



逻辑或指令OR

➤ 逻辑或(逻辑加)运算规则

▶ 两位都是逻辑**0**，则结果是**0**；否则，结果是**1**

➤ 逻辑或指令**OR**

▶ 按位进行逻辑或，结果返回目的操作数

OR reg,imm/reg/mem

;reg←reg ∨ imm/reg/mem

OR mem,imm/reg

;mem←mem ∨ imm/reg

01000101

∨ 00110001

01110101



逻辑非指令NOT

- 逻辑非(逻辑反)运算规则:
 - ▶ 原来为**0**的位变成**1**，原来为**1**的位变成**0**
 - 逻辑非指令**NOT**:
 - ▶ 按位进行逻辑非，结果返回操作数
- NOT reg/mem**
- ;reg/mem ← ~reg/mem**
- **NOT**指令不影响状态标志位

$$\begin{array}{r} \sim \quad 01000101 \\ \hline 10111010 \end{array}$$



逻辑异或指令XOR

- 逻辑异或(逻辑半加)运算规则:
 - ▶ 两位不同(相异), 则结果是**1**; 否则, 结果是**0**
- 逻辑异或指令**XOR**:
 - ▶ 按位进行逻辑异或, 结果返回目的操作数

XOR reg,imm/reg/mem

;reg←reg ⊕ imm/reg/mem

XOR mem,imm/reg

;mem←mem ⊕ imm/reg

01000101

⊕ 00110001

01110100



逻辑运算程序—1

； 数据段

varA dword 110010100001111001010101001101b

varB dword 00110111010110100011010111100001b

varT1 dword ?

varT2 dword ?



逻辑运算程序—2

mov eax,varA	;EAX=...1101B
not eax	;EAX=...0010B
and eax,varB	;EAX=...0000B
mov ebx,varB	;EBX=...0001B
not ebx	;EBX=...1110B
and ebx,varA	;EBX=...1100B
or eax,ebx	;EAX=...1100B
mov varT1,eax	;varT1=...1100B

varT1

$$= \overline{A} \cdot B + A \cdot \overline{B}$$



逻辑运算程序—3

```
mov eax,varA      ;EAX=...1101B
xor eax,varB       ;...1101B ⊕ ...0001B
mov varT2,eax      ;varT2=...1100B
```

$$\text{varT2} = A \oplus B$$

$$A \oplus B = \overline{A} \cdot B + A \cdot \overline{B}$$

varT1=varT2

```
mov eax,varT1
call dispbpd      ;显示varT1
call dispCrLf     ;换行显示
mov eax,varT2
call dispbpd      ;显示varT2
```



逻辑运算的屏蔽作用

➤ **AND**复位某些位(同0与), 不影响其他(同1与)

and bl,11110110b ;BL中D0和D3清0, 其余位不变

➤ **OR**置位某些位(同1或), 不影响其他(同0或)

or bl,00001001b ;BL中D0和D3置1, 其余位不变

➤ **XOR**求反某些位(同1异或), 不影响其他(同0异或)

xor bl,00001001b ;BL中D0和D3求反, 其余位不变

置位Set: 置1

复位Reset: 清0, 清除Clear



逻辑运算用于大小写转换

大写 = 小写 AND DFH

小写 = 大写 OR 20H

字母

ASCII值

'A'

01000001B

'B'

01000010B

...

...

'a'

01100001B

'b'

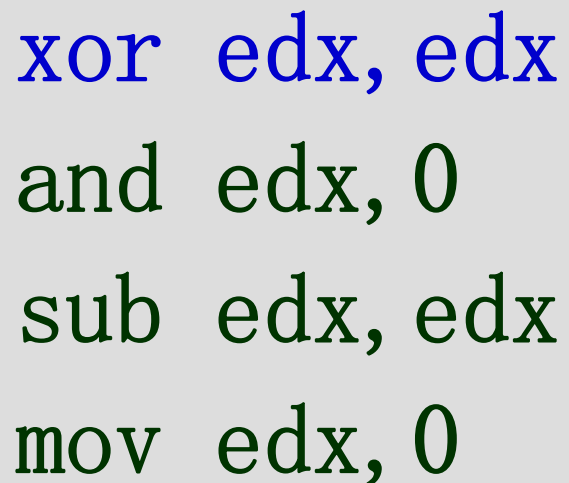
01100010B

...

...



逻辑运算用于清零



```
xor edx, edx  
and edx, 0  
sub edx, edx  
mov edx, 0
```

如何清零?
哪个最好?



本讲总结

- 逻辑运算指令属于位操作类指令
 - ▶ 逻辑与**AND**、逻辑或指令**OR**、逻辑非指令**NOT**
 - ▶ 逻辑异或**XOR**、测试指令**TEST**
- 除用于对数据进行逻辑运算外
 - ▶ 还常用于复位、置位或求反若干位
 - ▶ 异或指令常用于对寄存器清零
 - ▶



汇编语言程序设计

移位指令



位操作类指令

- 计算机中最基本的数据单位是二进制位
- 针对二进制位进行操作、实现位控制的指令
 - ▶ 逻辑运算指令
 - ▶ 移位指令
 - ▶ 循环移位指令
 - ▶
- 进行一位或若干位处理，采用位操作类指令



移位指令

- 逻辑左移指令 **SHL**
- 逻辑右移指令 **SHR**
- 算术左移指令 **SAL**
- 算术右移指令 **SAR**

S: Shift

L: Left

R: Right

A: Arithmetic

- 逻辑 (**Logical**)
 - 算术 (**Arithmetic**)

- 左移 (**Left**)
- 右移 (**Right**)



逻辑左移指令SHL / 算术左移指令SAL

SHL **reg/mem, i8/CL**

移位数据

SAL **reg/mem, i8/CL**

移位位数

;各位同时左移，最低位补**0**，最高位进入**CF**



2个助记符
同1条指令

i8 ;移动**i8**位（**i8**是8位数值）
CL ;移动**CL**寄存器指定的位数

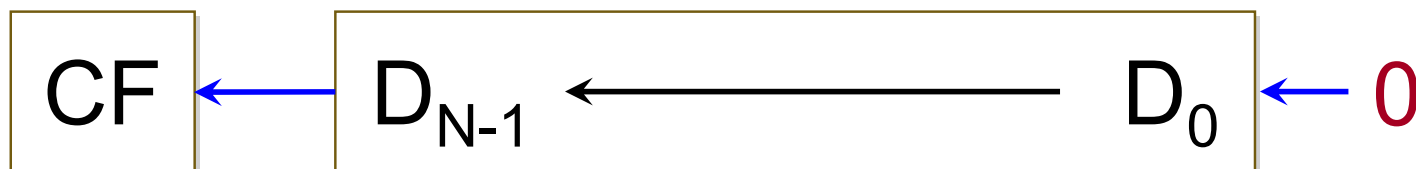


左移位指令SHL / SAL

SHL reg/mem,i8/CL

SAL reg/mem,i8/CL

;各位同时左移，最低位补**0**，最高位进入**CF**



;执行前 **AL = 11110101B**

SHL AL,1

;执行后 **AL = 11101010B, CF = 1**

逻辑右移指令SHR

SHR reg/mem,i8/CL

;各位同时右移，最高位补**0**，最低位进入**CF**



;执行前 **AL = 11110101B**

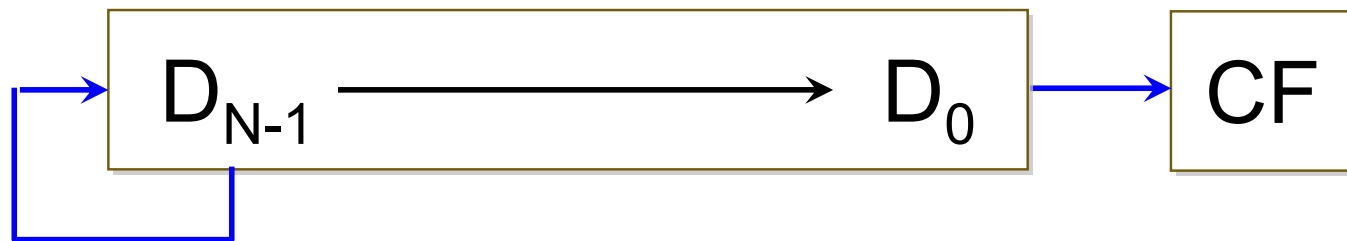
SHR AL,1

;执行后 **AL = 01111010B, CF = 1**

算术右移指令SAR

SAR reg/mem,i8/CL

;各位同时右移，最高位不变，最低位进入**CF**



;执行前 **AL = 11110101B**

SAR AL,1

;执行后 **AL = 11111010B, CF = 1**

本讲总结

- 二进制数的最基本单位是比特位 (**bit**)
- 移位操作也是最基本的数据处理方法
- 移位是以位为单位将数据向左或向右的移动
 - ▶ 左移指令**SHL**、**SAL**，两者相同
 - ▶ 右移指令**SHR**、**SAR**，两者不同
- 算术右移针对有符号数（补码）移位
 - ▶ 以便保持最高位不变、即数据符号不变



汇编语言程序设计

循环移位指令



位操作类指令

- 计算机中最基本的数据单位是二进制位
- 针对二进制位进行操作、实现位控制的指令
 - ▶ 逻辑运算指令
 - ▶ 移位指令
 - ▶ 循环移位指令
 - ▶
- 进行一位或若干位处理，采用位操作类指令



循环移位指令

- 不带进位循环左移指令 **ROL**
- 不带进位循环右移指令 **ROR**
- 带进位循环左移指令 **RCL**
- 带进位循环右移指令 **RCR**

R: Rotate

L: Left

R: Right

C: Carry flag

➤ 带进位 (**Carry flag**)

➤ 不带进位

➤ 左移 (**Left**)

➤ 右移 (**Right**)

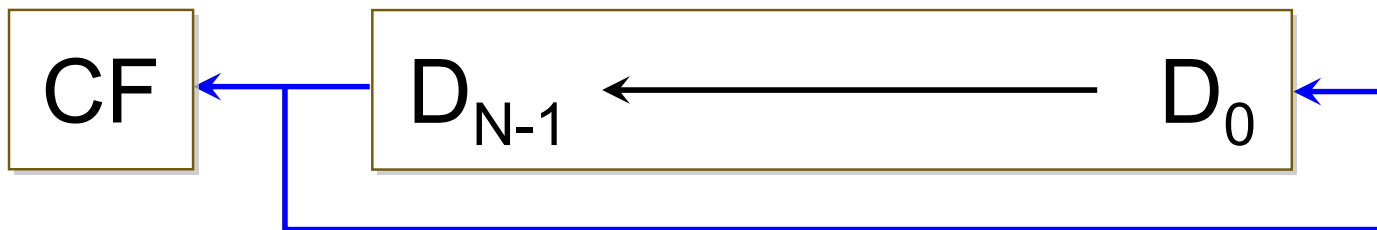


不带进位循环左移指令ROL

ROL reg/mem, i8/CL

移位位数

;各位同时左移，最高位循环进入最低位和**CF**



移位数据

i8 ;移动i8位（i8是8位数值）

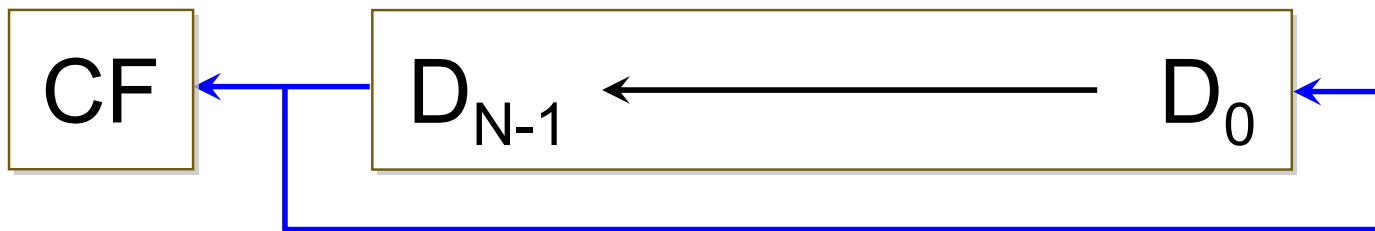
CL ;移动CL寄存器指定的位数



不带进位循环左移指令ROL

ROL reg/mem,i8/CL

;各位同时左移，最高位循环进入最低位和**CF**



;执行前 **AL = 11100101B**

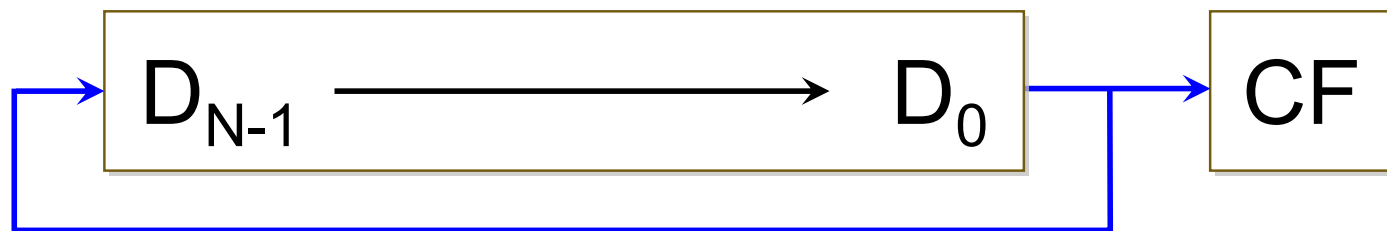
ROL AL,1

;执行后 **AL = 11001011B, CF = 1**

不带进位循环右移指令ROR

ROR reg/mem,i8/CL

;各位同时右移，最低位进入最高位和**CF**



;执行前 **AL = 11100101B**, **CL = 2**

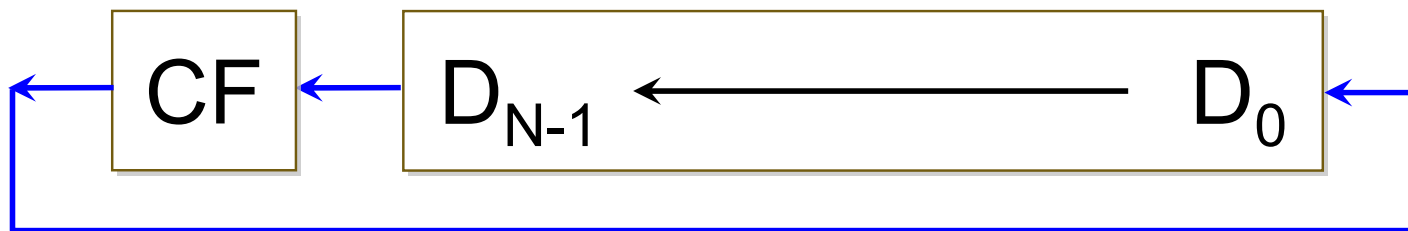
ROR AL,CL

;执行后 **AL = 01111001B**, **CF = 0**

带进位循环左移指令RCL

RCL reg/mem,i8/CL

;CF作为附加位，各位同时左移，**CF**进入最低位



;执行前AL = 11100101B, CF=0

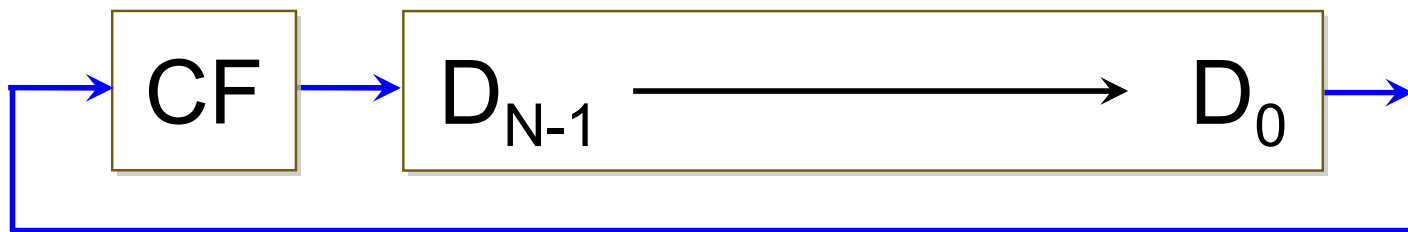
RCL AL,1

;执行后AL = 11001010B, CF = 1

带进位循环右移指令RCR

RCR reg/mem,i8/CL

;CF作为附加位，各位同时右移，最低位进入**CF**



;执行前AL=11100101B, CF=0

RCR AL,4

;执行后AL=10101110B, CF=0

本讲总结

- **IA-32**处理器不仅有移位指令
 - ▶ 逻辑左移**SHL**、逻辑右移**SHR**
 - ▶ 算术左移**SAL**、算术右移**SAR**
- **IA-32**处理器还有循环移位指令
 - ▶ 不带进位的循环左移**ROL**、右移**ROR**
 - ▶ 带进位的循环左移**RCL**、右移**RCR**
- 应用中，要根据问题灵活运用适当的移位操作

