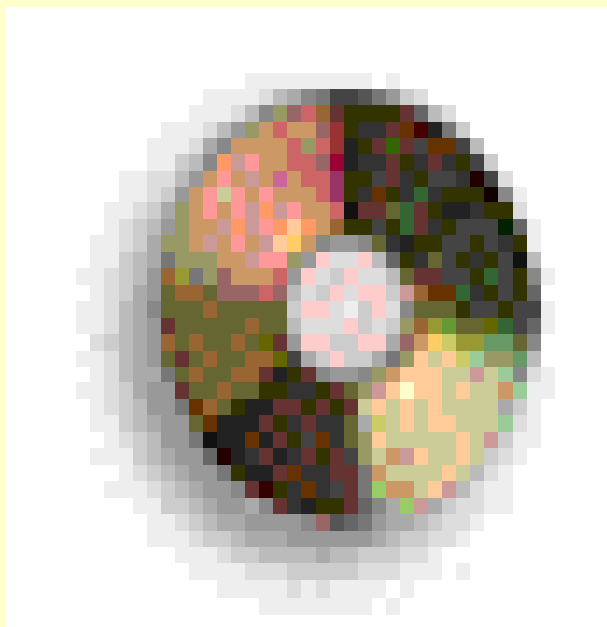


第三章 80x86的寻址方式与指令系统分类



一、8088(8086)的寻址方式

二、8088(8086)指令的分类

计算机是通过执行指令来管理计算机并完成一系列给定功能的。因而，每种计算机都有一组指令集提供给用户使用，这组指令集叫做计算机的指令系统。不同的计算机指令不同，指令集中指令的数量也不同，大体上在几十种到百余种。

指令的一般格式为：

操作码 操作数 ... 操作数

操作码 —— 告诉计算机要执行的操作是什么，如：加、减、逻辑与等。

操作数 —— 执行操作过程所要操作的数，如加运算的两个加数。

（一）8088的指令格式：

8088 CPU采用了一种较为灵活的指令格式，它由**1至6**个字节组成，每个字节都有特定的功能，指令字节长度随指令而异。通用格式如下：



操作码场		操作数场			
字节1	字节2	字节3	字节4	字节5	字节6
操作码 字节	寻址方式 字节	操作数的 低位地址 或 低位数据 字节	操作数的 高位地址 或 位数据 字节	操作数的 低位数据 字节	操作数的 高位数据 字节

如:	43	汇编语言指令 与 机器指令 一一对应	INC	BX
	89 08		MOV	AX, BX
	B8 00 10		MOV	AX, 1000
	02 85 00 20		ADD	AL, [DI+2000]
	C6 06 00 30 12		MOV	BYTE PTR[3000], 12
	C7 06 00 30 34 12		MOV	WORD PTR[3000], 1234
			操作码场	操作数场

(二) 汇编语言

汇编语言是一种符号语言,它用:

助记符——表示操作码; 符号或符号地址——表示操作数或操作数地址.



(三) 操作数存在方式

在微型计算机中，操作数可能以以下四种方式存在：

- 操作数包含在指令中——即指令的操作数场就包含着操作数本身。

MOV AX, 1234 ; ADD AL, 2

- 操作数包含在CPU的某一个内部寄存器中——这时指令中的操作数场是CPU内部寄存器的一个编码。 **MOV DS, AX**

- 操作数在内存的数据区中——这时指令中的操作数场包含着此操作数的地址。

MOV AX,[2000] ; MOV buffer[SI],AX

- 操作数在I/O端口中——这时指令中的操作数场包含着此操作数的端口号。

IN AL, DX; OUT 83H, AL



(四) 寻址方式

1.立即寻址方式

2.寄存器寻址

3.存储器寻址方式

1) 直接寻址方式

2) 寄存器间接寻址

3) 寄存器相对寻址

4) 基址加变址寻址

5) 相对基址加变址寻址

4.I/O寻址方式



二、IBM PC(8086/8088) 指令系统的指令分类

8086/8088 指令系统可分为六大类：

数据传送指令（Data transfer）；

算术指令（Arithmetic）；

逻辑指令（Logic）；

串处理指令（String manipulation）；

控制转移指令（Control transfer）；

处理器控制指令（Processor control）；



数据传送类指令

分类	名称	格式	功能	OSZ APC
通用 数据 传送 指令	基本传送指令	MOV DST, SRC	字、字节传送	不影响
	进栈指令	PUSH OPRD	字 压入堆栈	不影响
	出栈指令	POP OPRD	字 弹出堆栈	不影响
	交换指令	XCHG DST, SRC	字、字节交换	不影响
地址 传送 指令	有效地址送寄存器指令	LEA REG, SRC	有效地址→寄存器	不影响
	指针送寄存器及DS指令	LDS REG, SRC	地址指针→寄存器, DS	不影响
	指针送寄存器及ES指令	LES REG, SRC	地址指针→寄存器, ES	不影响
累加 器专 用指 令	输入指令	IN AL, PORT	外设数据→AL	不影响
	输出指令	OUT PORT, AL	AL → 外设数据	不影响
	换码指令	XLAT	AL中数据转换	不影响
标志 寄存 器传 送指 令	读取标志指令	LAHF	Flag 低字节→AH	不影响
	设置标志指令	SAHF	AH → Flag 低字节	Z A P C
	标志寄存器入栈指令	PUSHF	把Flag内容压入堆栈	不影响
	标志寄存器出栈指令	POPF	把Flag内容弹出堆栈	O S Z A P C



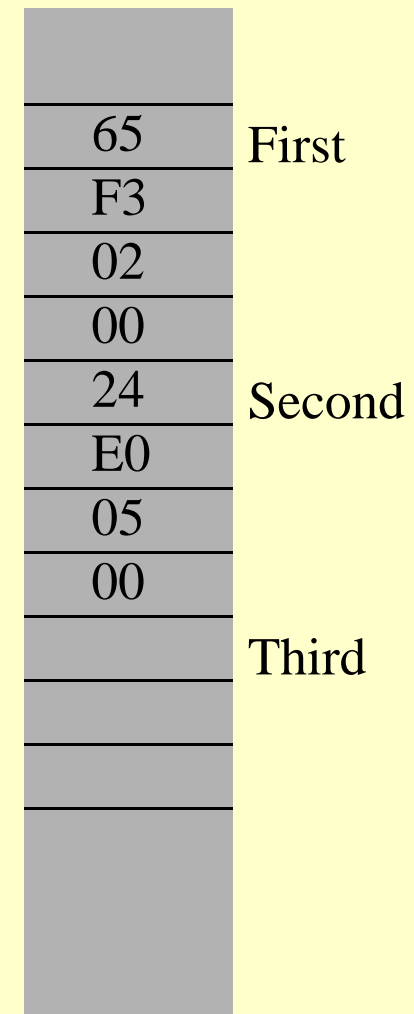
算术运算类指令

分类	名称	格式	功能	OSZAPC
加法指令	加法指令	ADD DST, SRC	加法（字、字节）	OSZAPC
	带进位加法指令	ADC DST, SRC	带进位加法(字、字节)	OSZAPC
	加 1 指令	INC OPRD	加 1（字、字节）	OSZAP
减法指令	减法指令	SUB DST, SRC	减法（字、字节）	OSZAPC
	带借位减法指令	SBB DST, SRC	带借位减法(字、字节)	OSZAPC
	减 1 指令	DEC OPRD	减 1（字、字节）	OSZAP
	比较指令	CMP DST, SRC	比较（字、字节）	OSZAPC
	求补指令	NEG OPRD	求补码	OSZAPC
乘法指令	无符号数乘法	MUL SRC	不带符号数乘法(字,字节)	O C
	带符号数乘法	IMUL SRC	带符号数乘法(字,字节)	O C
除法指令	无符号数除法	DIV SRC	不带符号数除法(字,字节)	
	带符号数乘法	IDIV SRC	带符号数除法(字,字节)	
	符号扩展指令	CBW	扩展AL中的符号	不影响
	符号扩展指令	CWD	扩展AX中的符号	不影响
十进制调整指令				



例：在内存的First和Secontd开始的区域中分别存放着2F365H和 5E024 H两个数，要求求其和，并存入Third中。

$$\begin{array}{r} 2F365 \\ + 5E024 \\ \hline 8D389 \end{array}$$



$$\begin{array}{r}
 2F365 \\
 + 5E024 \\
 \hline
 8D389
 \end{array}
 \rightarrow
 \begin{array}{r}
 \textcircled{1} \quad F365 \\
 + E024 \\
 \hline
 D389
 \end{array}$$

$$\begin{array}{r}
 \textcircled{2} \quad 0002 \\
 + 0005 \\
 \hline
 0008
 \end{array}$$

```

MOV AX, First
ADD AX, Second
MOV Third, AX
MOV AX, First+2
ADC AX, Second+2
MOV Third+2, AX

```

65	First
F3	
02	
00	Second
24	
E0	
05	Third
00	
89	
D3	
08	
00	



逻辑运算类指令

分类	名称	格式	功能	OSZAPC
逻辑运算指令	逻辑与指令	AND DST, SRC	与（字、字节）	OSZPC
	逻辑或指令	OR DST, SRC	或（字、字节）	OSZPC
	逻辑非指令	NOT OPRD	非（字、字节）	不影响
	逻辑异或指令	XOR DST, SRC	异或（字、字节）	OSZPC
	测试指令	TEST DST, SRC	测试（字、字节）	OSZPC
移位指令	逻辑左移指令	SHL OPRD, COUNT	逻辑左移（字、字节）	OSZPC
	算术左移指令	SAL OPRD, COUNT	算术左移（字、字节）	OSZPC
	逻辑右移指令	SHR OPRD, COUNT	逻辑右移（字、字节）	OSZPC
	算术右移指令	SAR OPRD, COUNT	算术右移（字、字节）	OSZPC
循环移位指令	循环左移指令	ROL OPRD, COUNT	循环左移(字,字节)	OC
	循环右移指令	ROR OPRD, COUNT	循环右移(字,字节)	OC
	带进位循环左移指令	RCL OPRD, COUNT	带进位循环左移(字,字节)	OC
	带进位循环右移指令	RCR OPRD, COUNT	带进位循环右移(字,字节)	OC



例：屏蔽AL寄存器的低四位。

MOV AL, FF

AND AL, F0

$$\begin{array}{r} 11111111 \\ \wedge 11110000 \\ \hline 11110000 \end{array}$$

使一立即数的该位为0即可

例：屏蔽AL寄存器的高两位。

MOV AL, FF

AND AL, 3F

$$\begin{array}{r} 11111111 \\ \wedge 00111111 \\ \hline 00111111 \end{array}$$

使一立即数的该位为0即可

AND通常用于使某些位置0，其它位不变的情况。



例：使某数的第4、5位置1。

```
MOV AL, 03H
OR  AL, 30H
```

	0	0	0	0	0	0	1	1
✓	0	0	1	1	0	0	0	0
<hr/>								
	0	0	1	1	0	0	1	1

使立即数的该位为1即可

OR指令通常用于
将某些位置1



例：使某数的D₁、D₀位取反，其它位不变。

MOV AL, 11H

XOR AL, 03H

$$\begin{array}{r} 00010001 \\ \oplus 00000011 \\ \hline 00010010 \end{array}$$

使一立即数的
相应位为1即可

例：测试AX中的数与42H是否相同，相同则转移。

XOR AX, 0042H

JZ Next

⋮

Next: MOV BX, 30H

⋮

给寄存器清0

测试两操作数
是否相等

例：给某寄存器清0。

XOR AX, AX

AX=0

结果

使一操作数
若干位维持不变，
若干位取反



TEST DST, SRC —— 测试

源操作数：通用寄存器、存储器、立即数

目的操作数：通用寄存器、存储器

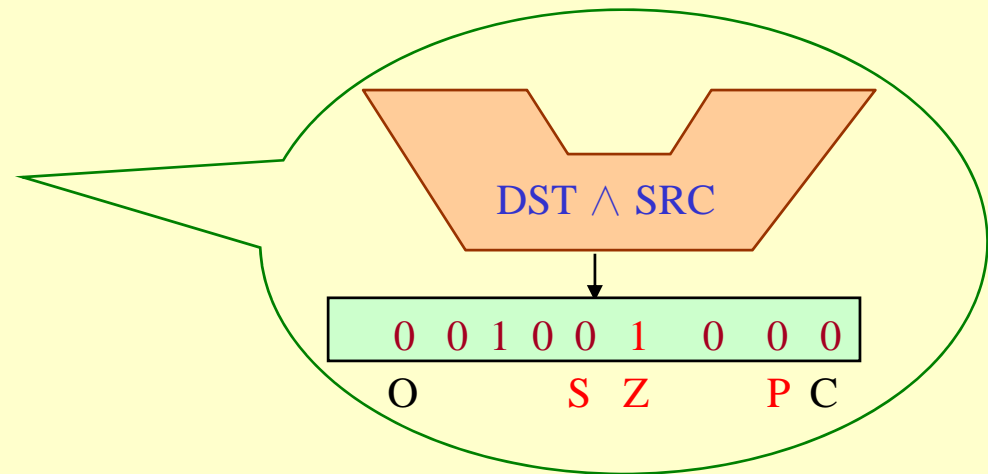
执行的操作：DST and SRC

功能：实现两个操作数的按位进行与运算，结果不保存，只影响标志位。

注：该指令影响标志位

使 O=0,C=0,

P,S,Z反映操作的结果。



例：检测AL中的最低位是否位1，若为1则转移。

```
MOV AL, 41H
```

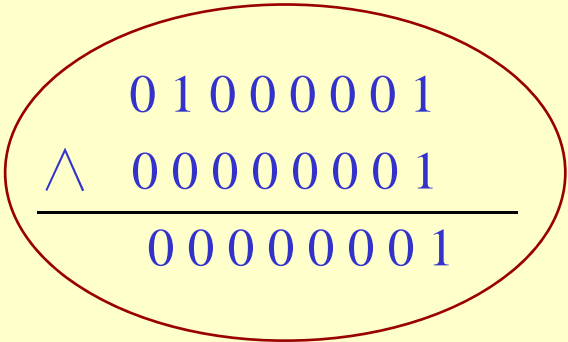
```
TEST AL, 01H
```

```
JNZ Next
```

```
⋮
```

```
Next: MOV BL, 0
```

```
⋮
```


$$\begin{array}{r} 01000001 \\ \wedge 00000001 \\ \hline 00000001 \end{array}$$

TEST 通常用于检测一些条件是否满足，但又不希望改变原来操作数的情况，该指令后通常带有条件转移指令。

例：检测CX内容是否位 0，为 0 则转移。

① **TEST** CX, 0FFFFH

JZ Next

⋮

Next: MOV AX, 1

⋮

② **CMP** CX, 0

JZ Next

⋮

Next: MOV AX,1

⋮



串操作类指令

分类	指令	功能	控制标志D
串传送指令	MOVSB MOVSW	字节传送 字传送	设置D
串比较指令	CMPSB CMPSW	字节串比较 字串比较	设置D
串扫描指令	SCASB SCASW	字节串扫描 字串扫描	设置D
存入串指令	STOSB STOSW	字节串存入 字串存入	设置D
从串取指令	LODSB LODSW	取字节串 取字串	设置D
重复前缀	REP REPE/REPZ REPNE/REPNZ	无条件重复 当相等/为零重复 当不相等/不为零重复	



例：将数据段中起始地址

为Source的100个字节
的数据传送到附加段
的Dest指向的单元中。

LEA SI, Source

LEA DI, Dest

MOV CX, 100

CLD

Again: **MOVSB**

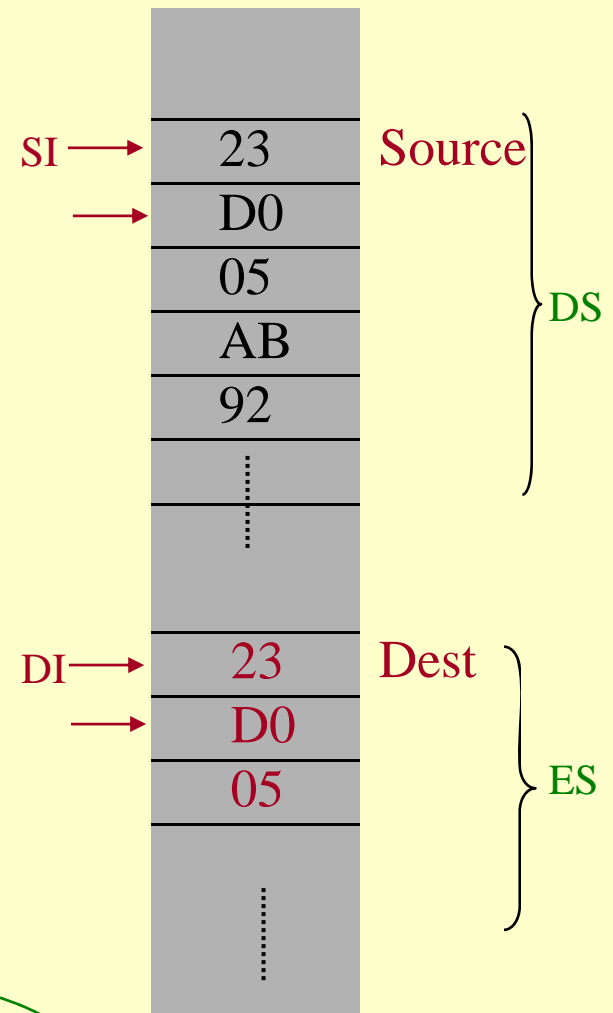
DEC CX

JNZ Again

HLT

增址传
送

[SI] → [DI]
SI+1 → SI
DI+1 → DI



控制转移类指令

分类	指令	功能
无条件转移指令	JMP OPRD	无条件转移
条件转移指令	JNZ OPRD (等)	根据上一条指令设置的标志位的情况转移
循环指令	LOOP OPRD	计数非零循环
	LOOPE/LOOPZ OPRD	计数非零循环且结果为0循环
	LOOPNE/LOOPNZ OPRD	计数非零循环且结果不为0循环
子程序调用及返回指令	CALL OPRD	调用子程序
	RET	从子程序返回
中断指令	INT N	软中断
	INTO	溢出时中断
	IRET	中断返回



处理器控制指令

标志处理指令

名称	格式	功能（对标志位的影响）
进位标志清 0 指令	CLC	$C = 0$
进位标志置 1 指令	STC	$C = 1$
进位标志取反	CMC	$C = \overline{C}$
方向标志清 0 指令	CLD	$D = 0$
方向标志置 1 指令	STD	$D = 1$
中断标志清 0 指令	CLI	$I = 0$
中断标志置 1 指令	STI	$I = 1$



其它处理指令

名称	格式	功能	状态标志位
处理器等待指令	WAIT	处理器等待	不影响
处理器交权指令	ESC	处理器交权	不影响
总线封锁前缀	LOCK	封锁总线	不影响
处理器暂停指令	HLT	使处理器暂时处于停机状态	不影响
空操作指令	NOP	使CPU不进行任何操作	不影响