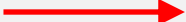


# 逻辑运算指令

# 逻辑运算

- 基本逻辑运算：
  - 与、或、非
  - 异或
- 逻辑运算指令  实现逻辑操作的指令
  - 与、或、非、异或

# 逻辑运算指令

## ■ 对操作数的要求：

- 大多与MOV指令相同。
- “非”运算指令要求操作数不能是立即数；

## ■ 对标志位的影响

- 除“非”运算指令，其余指令的执行都会影响除AF外的5个状态标志；
- 无论执行结果任何，都会使标志位OF=CF=0。
- “非”运算指令的执行不影响标志位。

# 1. “与”指令：

- 格式：

- AND OPRD1, OPRD2

- 操作：

- 两操作数相“与”，结果送目标地址。

# “与”指令的应用

- 实现两操作数按位相与的运算
  - `AND BL, [SI]`
- 使目标操作数的某些位不变，某些位清零
  - `AND AL, 0FH`
- 在操作数不变的情况下使CF和OF清零
  - `AND AX, AX`

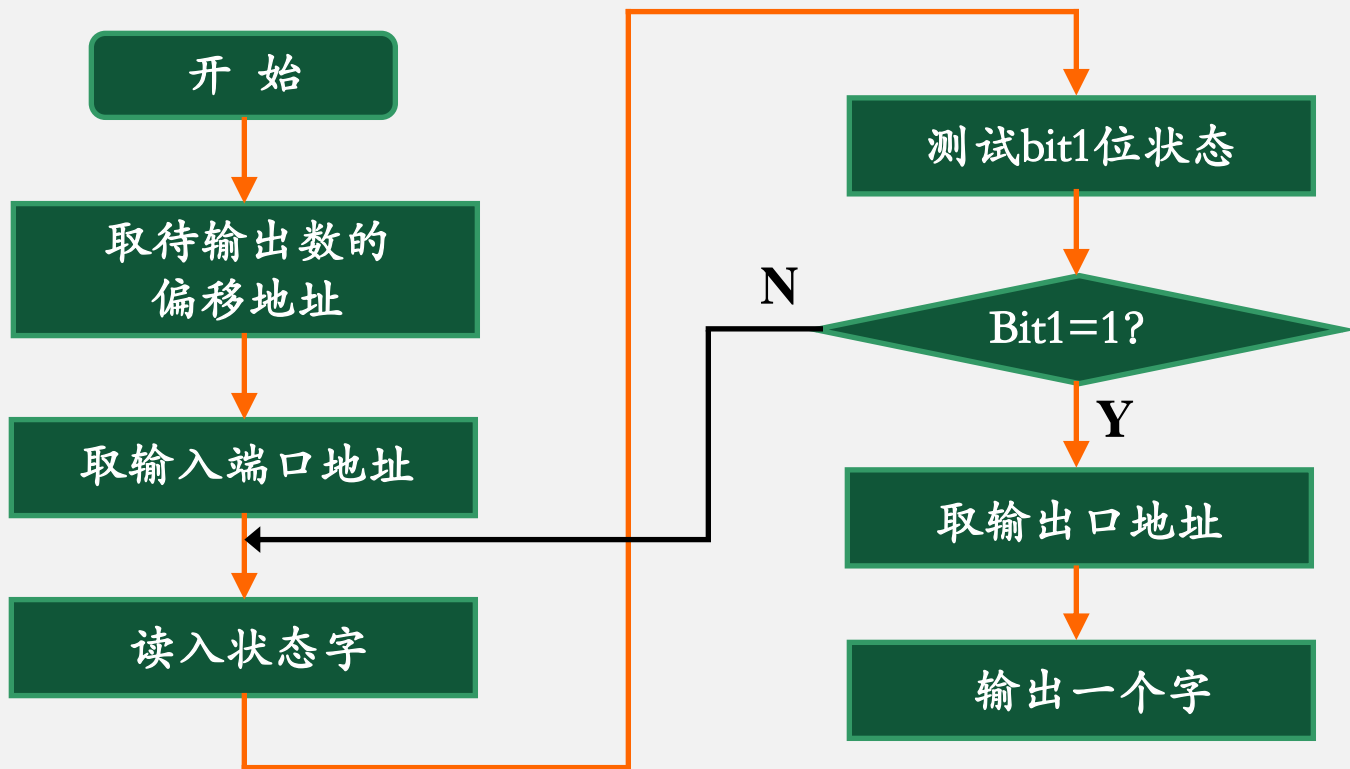
# “与”指令应用例

- 从地址为3F8H端口中读入一个字节数，如果该数bit1位为1，则可从38FH端口将DATA为首地址的1个字输出，否则就不能进行数据传送。
- 要求：
  - 编写相应的程序段。



状态字

# “与”指令应用例



# “与”指令应用例

**MOV DX, 3F8H**

**WATT: IN AL, DX**

**AND AL, 02H**

**JZ WATT**  **ZF=1转移**

**MOV DX, 38FH**

**MOV AX, DATA**

**OUT DX, AX**



## 2. “或” 运算指令

- 格式:

- OR OPRD1, OPRD2

- 操作:

- 两操作数相“或”，结果送目标地址

# “或”指令的应用

- 实现两操作数相“或”的运算
  - **OR AX, [DI]**
- 使某些位不变，某些位置“1”
  - **OR CL, 0FH**
- 在不改变操作数的 情况下使OF=CF=0
  - **OR AX, AX**

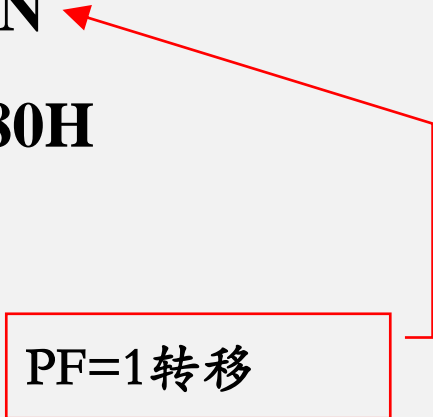
# “或”指令的应用例

**OR AL, AL**

**JPE GOON**

**OR AL, 80H**

**GOON: ....**



**PF=1 转移**

# “或”指令的应用

将一个二进制  
数9变为字符  
'9'

```
MOV AL, 9  
OR AL, 30H
```

如何实现？

### 3. “非” 运算指令

- 格式：
  - NOT OPRD
- 操作：
  - 操作数按位取反再送回原地址
- 注：
  - 指令的执行对标志位无影响
- 例：
  - **NOT BYTE PTR[BX]**

## 4. “异或” 运算指令

- 格式:

- XOR OPRD1, OPRD2

- 操作:

- 两操作数相“异或”，结果送目标地址

- 例:

- XOR BL, 80H

- XOR AX, AX

## 5. “测试” 指令

- 格式:

- TEST OPRD1, OPRD2

- 操作:

- 执行“与”运算，但运算的结果不送回目标地址。

- 应用:

- 常用于测试某些位的状态

## 例：

- 从地址为3F8H的端口中读入一个字节数，当该数的bit1，bit3，bit5位同时为1时，可从38FH端口将DATA为首地址的一个字输出，否则就不能进行数据传送。
- 编写相应的程序段。



# 源程序代码：

```
LEA SI, DATA
MOV DX, 3F8H
WATT: IN AL, DX

AND AL, 2AH
CMP AL, 2AH
JNZ WATT

MOV DX, 38FH
MOV AX, [SI]
OUT DX, AX
```

