微机原理与接口技术

第七章 中断控制接口

第7章 中断控制接口

- **教学重点**

- 8088 CPU的中断系统
- 8259A的中断工作过程和工作方式
- 中断服务程序的编写

7.1 8088中断系统

- 8088的中断系统采用向量中断机制
- 能够处理256个中断
- 用中断向量号0～255区别
- 可屏蔽中断还需要借助专用中断控制器Intel 8259A实现优先权管理

8088的中断向量表

- 中断向量：中断服务程序的入口地址（首地址）
- 逻辑地址含有段地址CS和偏移地址IP（32位）
- 每个中断向量的低字是偏移地址、高字是段地址，需占用4个字节

8088的中断向量表

- 8088微处理器从物理地址00000H开始，依次安排各个中断向量，向量号也从0开始
- 256个中断占用1KB区域，就形成**中断向量表**
- 向量号为N的中断向量的物理地址＝N×4

8088的中断向量表

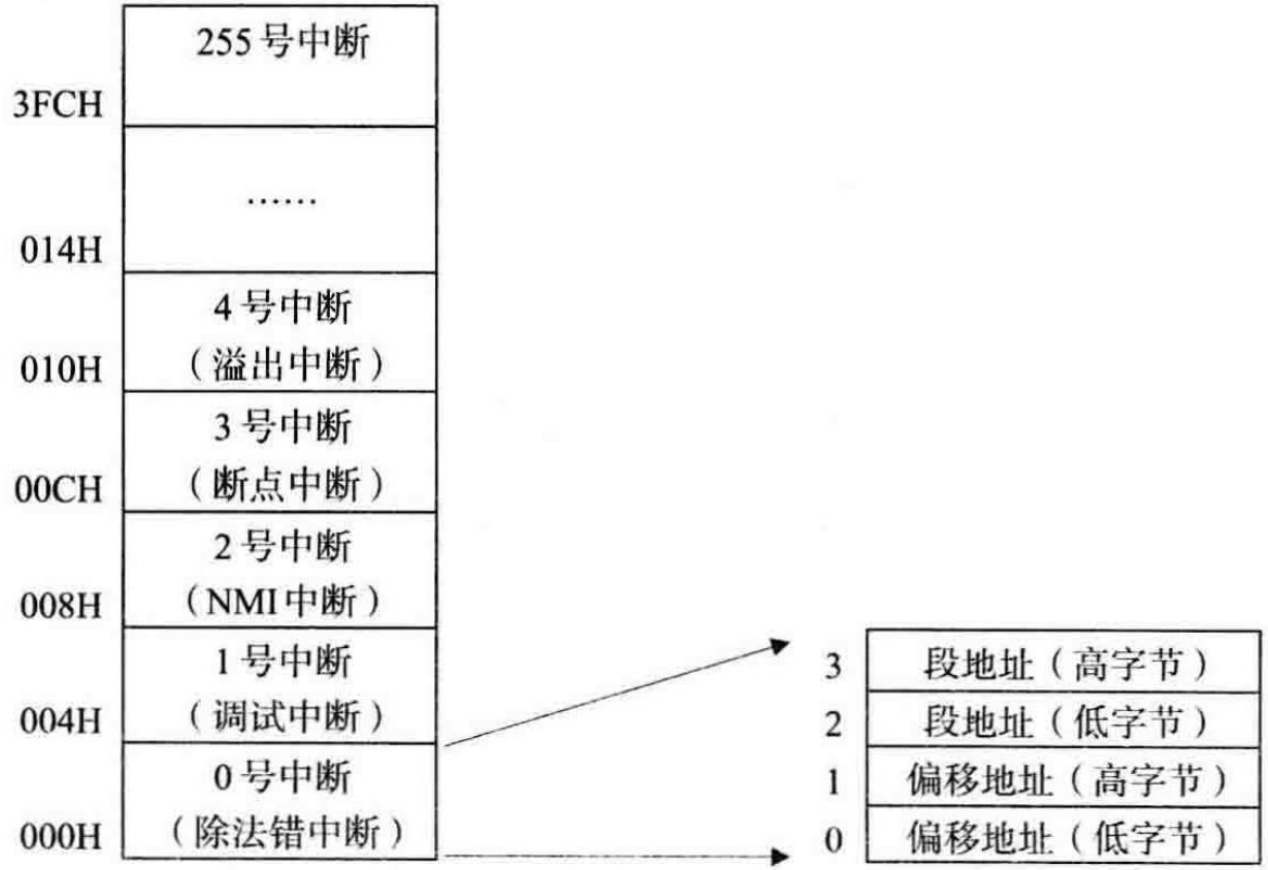
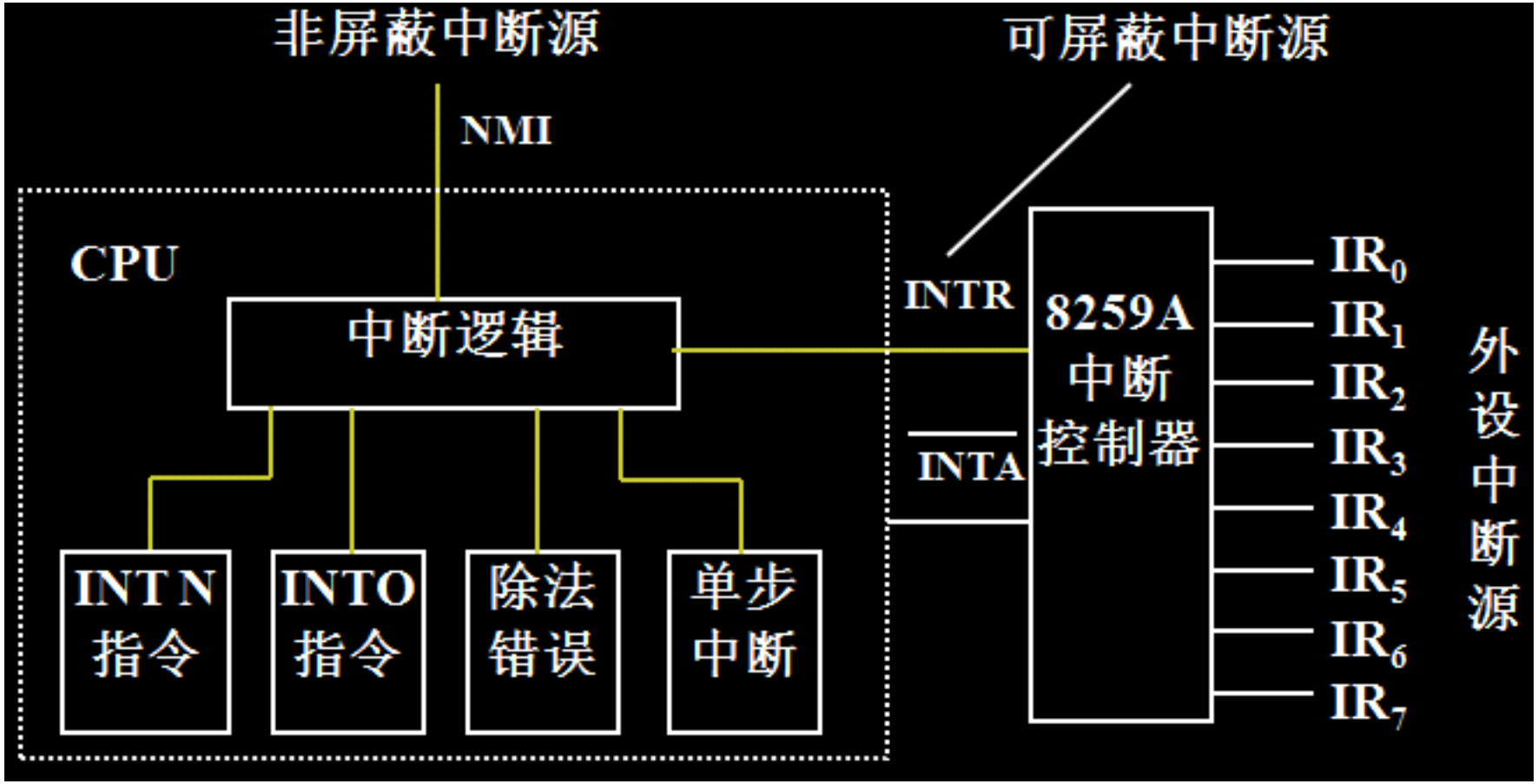


图7-3　8088的中断向量表

7.1.1 8088的中断类型



7.1.1 8088的中断类型

- 内部中断
  - 除法错中断
  - 指令中断
  - 溢出中断
  - 单步中断
- 外部中断
  - 非屏蔽中断
  - 可屏蔽中断

1. 内部中断

- 内部中断是由于8088**内部执行程序出现异常引起**的程序中断

- 利用内部中断，微处理器为用户提供了发现、调试并解决程序执行时异常情况的有效途径

- 此外，ROM-BIOS和DOS系统利用内部中断为程序员提供了各种功能调用

(1) 除法错中断

- 在执行除法指令时，若除数为0或商超过了寄存器所能表达的范围，则产生一个向量号为0的内部中断，称为除法错中断

- 例1.
- mov bl,0
- idiv bl　；除数BL＝0，产生除法错中断

(2) 指令中断

- 在执行中断调用指令INT n时产生的一个向量号为n（0～255）的内部中断，称为指令中断

- 其中向量号为3的指令中断比较特别（生成一个字节的指令代码：11001100），常用于程序调试，被称为断点中断

- 例如：DEBUG.EXE调试程序的运行命令G设置的断点，就是利用INT 3指令实现的

(3) 溢出中断

- 在执行溢出中断指令INTO时，若溢出标志OF为1，则产生一个向量号为4的内部中断，被称为溢出中断

例如：

```
mov ax,2000h
add ax, 7000h
；2000H＋7000H＝9000H，溢出：OF＝1
into        ；因为OF＝1，所以产生溢出中断
```

(4)单步中断

- 若单步中断TF为1，则在每条指令执行结束后产生一个向量号为1的内部中断，称为单步中断

- **例如：DEBUG.EXE调试程序的单步命令T就利用单步中断实现对程序的单步调试**

2. 外部中断

- 外部中断是由于8088**外部提出中断请求引起**的程序中断
- 利用外部中断，微机系统可以实时响应外部设备的数据传送请求，能够及时处理外部意外或紧急事件
- 外部中断的原因是处理器外部随机产生的，所以是真正的**中断**（Interrupt）
- 内部中断的原因是处理器执行程序出现异常，所以经常被称为**异常**（Exception）

(1) 非屏蔽中断

- 通过非屏蔽中断请求信号向微处理器提出的中断请求，微处理器无法禁止，将在当前指令执行结束予以响应，这个中断被称为非屏蔽中断

- 8088的非屏蔽中断的向量号为2，非屏蔽中断请求信号为NMI

(1) 非屏蔽中断

- 非屏蔽中断主要用于处理系统的意外或故障。例如：
  - 电源掉电前的数据保护
  - 存储器读写错误的处理

(2) 可屏蔽中断

- 外部通过可屏蔽中断请求信号向微处理器提出的中断，微处理器在允许可屏蔽中断的条件下，在当前指令执行结束予以响应，同时输出可屏蔽中断响应信号，这个中断就是可屏蔽中断

- 8088的可屏蔽中断请求和响应信号分别是INTR和INTA*；由IF标志控制可屏蔽中断是否允许响应；向量号来自外部中断控制器

(2) 可屏蔽中断

- 8088通常需要配合中断控制器8259A共同处理可屏蔽中断
- 可屏蔽中断主要用于主机与外设交换数据
- **IF控制可屏蔽中断的响应**

中断标志IF的状态

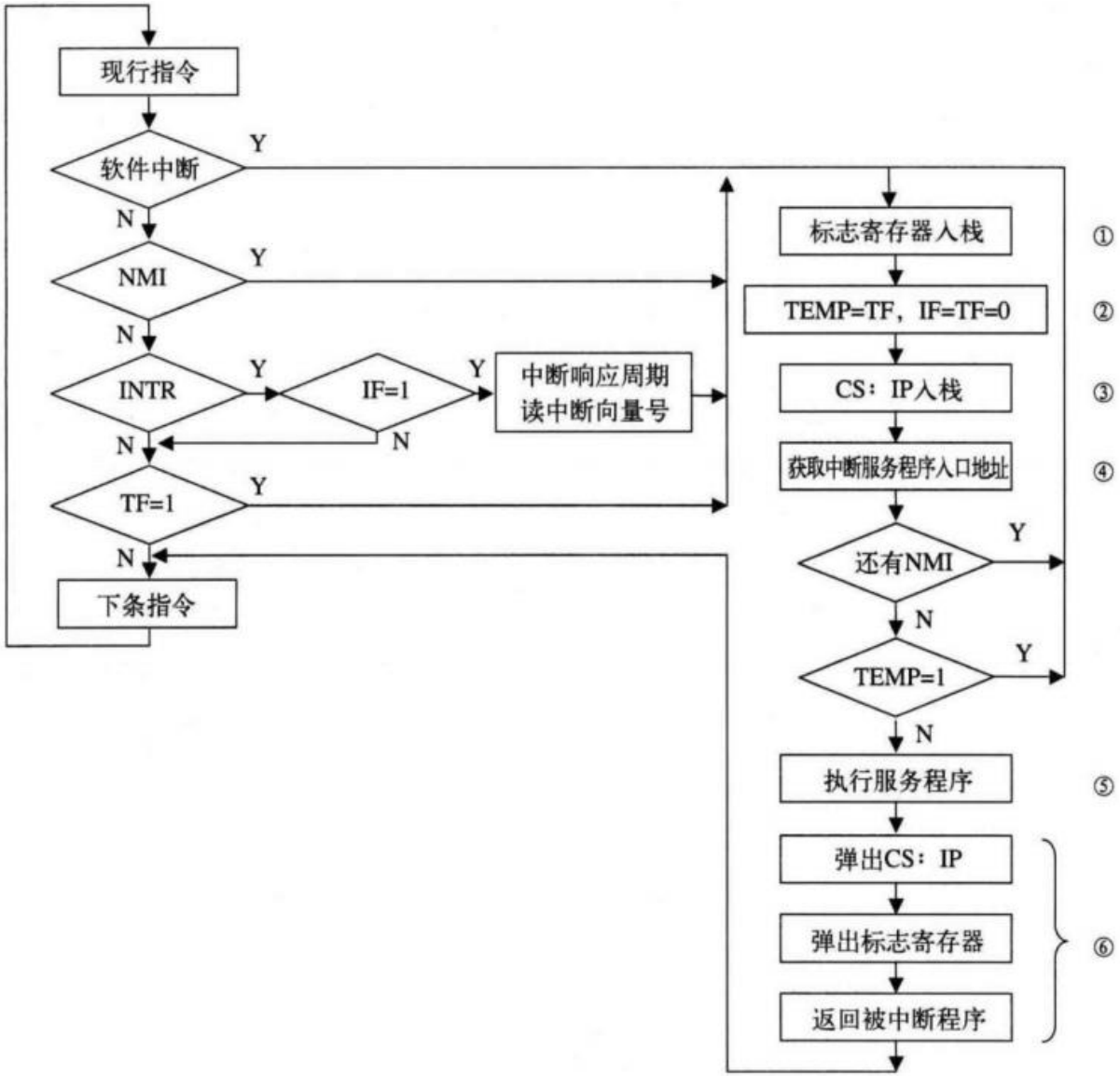
- IF＝0：可屏蔽中断不会被响应
  - 关中断、禁止中断、中断屏蔽
  - 系统复位，使IF＝0
  - 任何一个中断被响应，使IF＝0
  - 执行指令CLI，使IF＝0

中断标志IF的状态

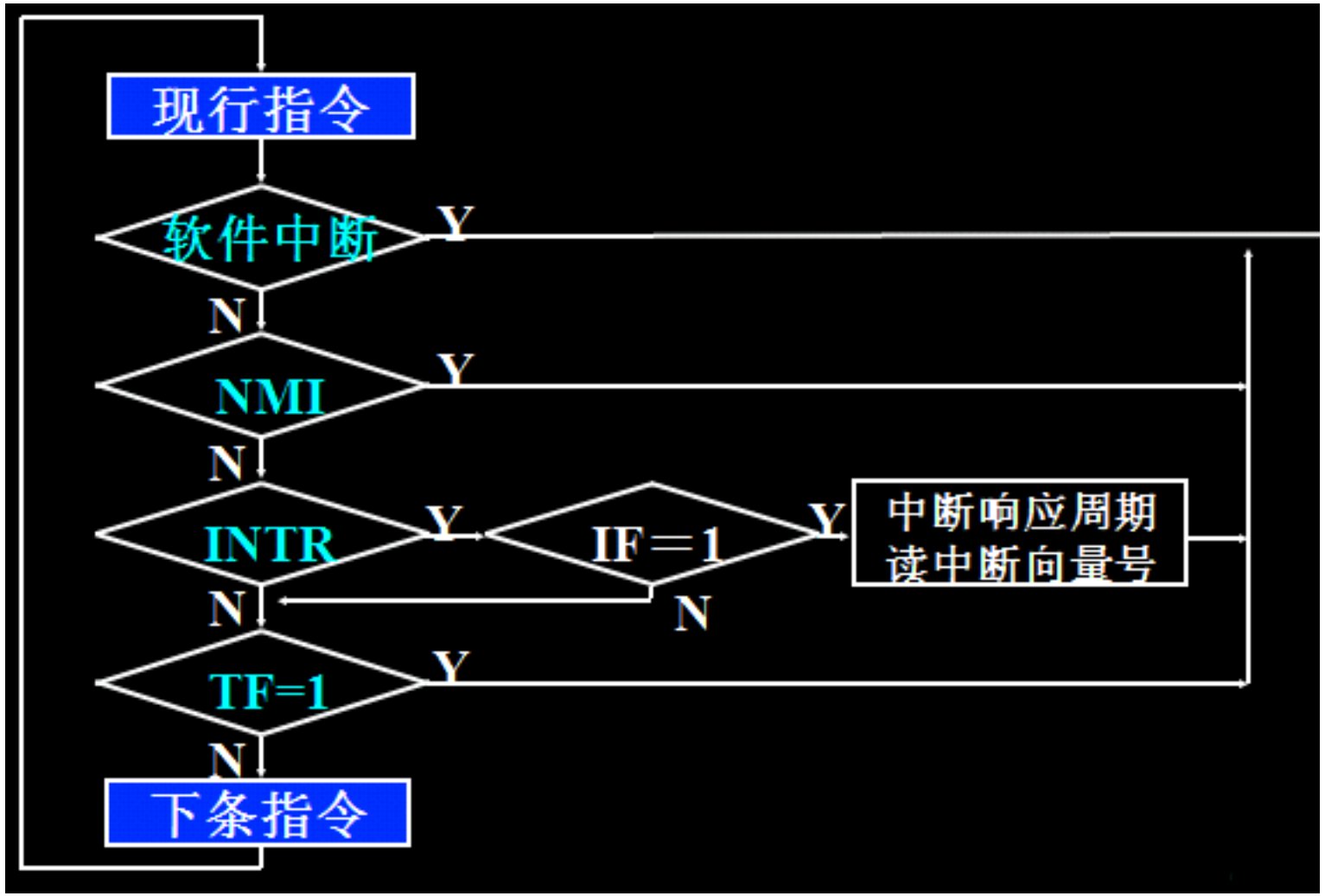
- IF＝1：可屏蔽中断会被响应
  - 开中断、允许中断、中断开放
  - 执行指令STI，使IF＝1
- 执行指令IRET恢复原IF状态

- **明确IF标志的状态是关键**

7.1.2 8088的中断响应过程



7.1.2 8088的中断响应过程



7.1.2 8088的中断响应过程

查询中断的顺序，
决定了各种中断源的优先权（**由高到低**）

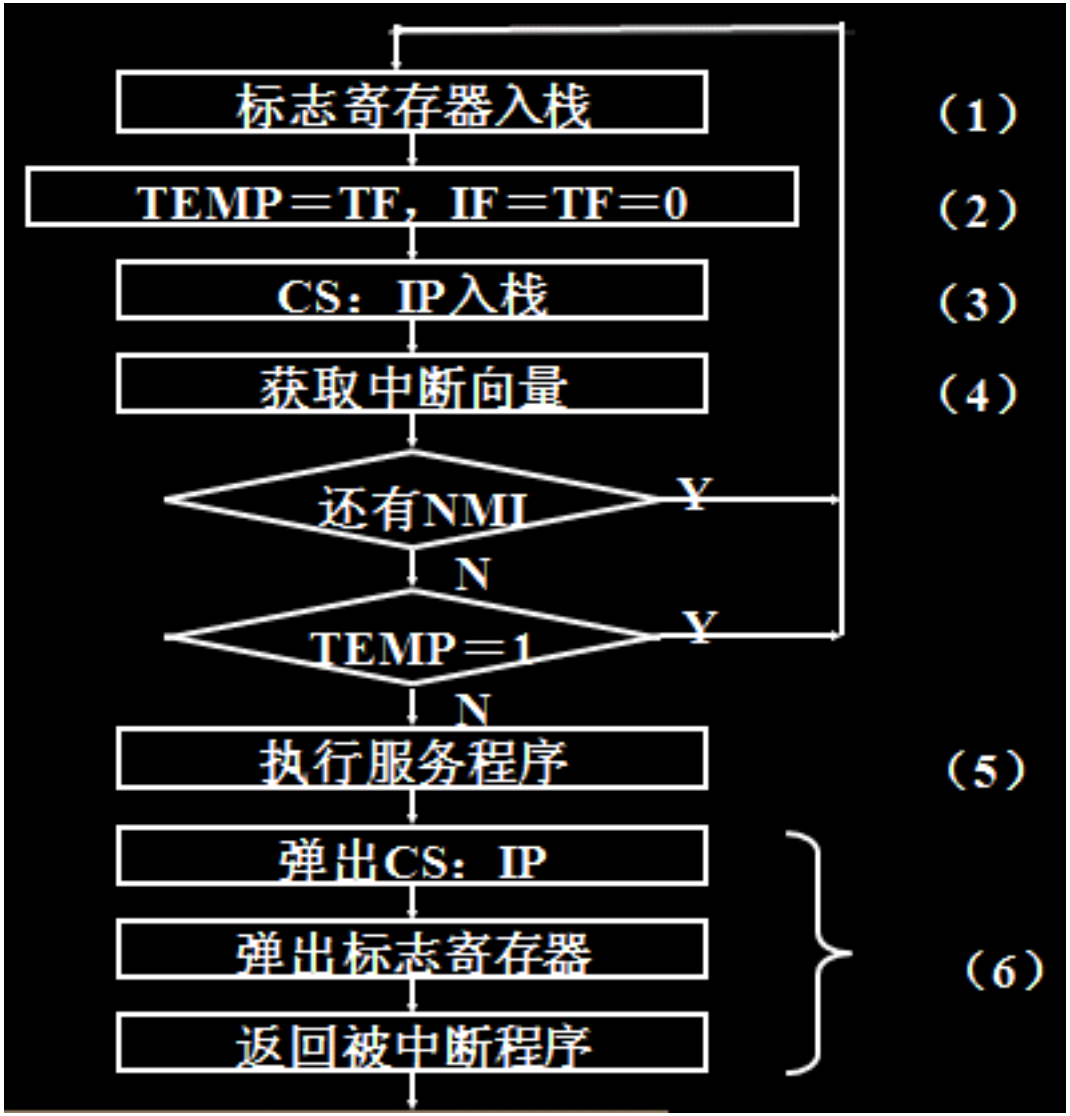
- 软件中断
  - 除法错中断
  - 指令中断
  - 溢出中断
- 非屏蔽中断
- 可屏蔽中断
- 单步中断

7.1.2 8088的中断响应过程

- 8088各种中断源的优先权，实际上是指被识别出来的先后

- 多种中断同时请求时，最先响应的则可能是单步中断或NMI中断

7.1.2 8088的中断响应过程



7.2 内部中断服务程序

- 编写内部中断服务程序与编写子程序类似
  - 利用过程定义伪指令PROC/ENDP
  - 第1条指令通常为开中断指令STI
  - 最后用中断返回指令IRET
  - 通常采用寄存器传递参数

7.2 内部中断服务程序

- 主程序需要调用中断服务程序
  - 调用前，需要设置中断向量
  - 利用INT n指令调用中断服务程序

例7.1 内部中断服务程序

- 编写80H号中断服务程序
- 功能：显示以“0”结尾字符串的功能
- 利用显示器功能调用INT 10H
- 字符串缓冲区首地址为入口参数
- 利用DS:DX（段地址：偏移地址）传递参数

例7.1 数据段

```
intoff    dw ？ ;保存原中断服务程序的偏移地址
intseg    dw ？ ;保存原中断服务程序的断基地址
intmsg    db ‘A Instruction Interrupt !’
          db 0dh,0ah,0
```

回车、换行

以 “0”结尾

例7.1 保存中断向量

获取中断向量（DOS功能调用INT 21H）

功能号：AH＝35H

入口参数：AL＝中断向量号

出口参数：
ES:BX＝中断向量（段地址：偏移地址）

例7.1 保存中断向量

```
mov ax,3580h
int 21h
mov intoff,bx     ;保存偏移地址
mov intseg,es     ;保存段基地址
```

例7.1 设置中断向量

设置中断向量（DOS功能调用INT 21H）

功能号：AH＝25H

入口参数：
AL＝中断向量号
DS:DX＝中断向量（段地址：偏移地址）

例7.1 设置中断向量

```
push ds
mov dx,offset new80h
mov ax,seg new80h
mov ds,ax
mov ax,2580h
int 21h
pop ds
```

例7.1 调用中断服务程序

```
; 设置入口参数：DS＝段地址（已设置）
;              DX＝偏移地址

    mov dx,offset intmsg
    int 80h                ；调用80H中断服务程序
```

例7.1 主程序结束

```
mov dx,intoff    ;恢复系统的原80H中断服务程序
mov ax,intseg
mov ds,ax
mov ax,2580h
int 21h
mov ax,4c00h
int 21h
```

例7.1进入中断服务程序

```
; 80H号内部中断服务程序:
; 显示字符串（以“0”结尾）
; 入口参数：DS:DX＝缓冲器首地址
new80h   proc
         sti              ;开中断
         push ax          ;保护寄存器
         push bx
         push si
```

例7.1显示字符串

```
        mov si,dx
new1:   mov al,[si]
        cmp al,0
        jz new2
        mov bx,0    ;显示一个字符
        mov ah,0eh
        int 10h
        inc si
        jmp new1
```

例7.1 退出中断服务程序

```
new2:    pop si        ;恢复寄存器
         pop bx
         pop ax
         iret          ;中断返回，注意不是ret
new80h   endp
```

7.3 8259A中断控制器

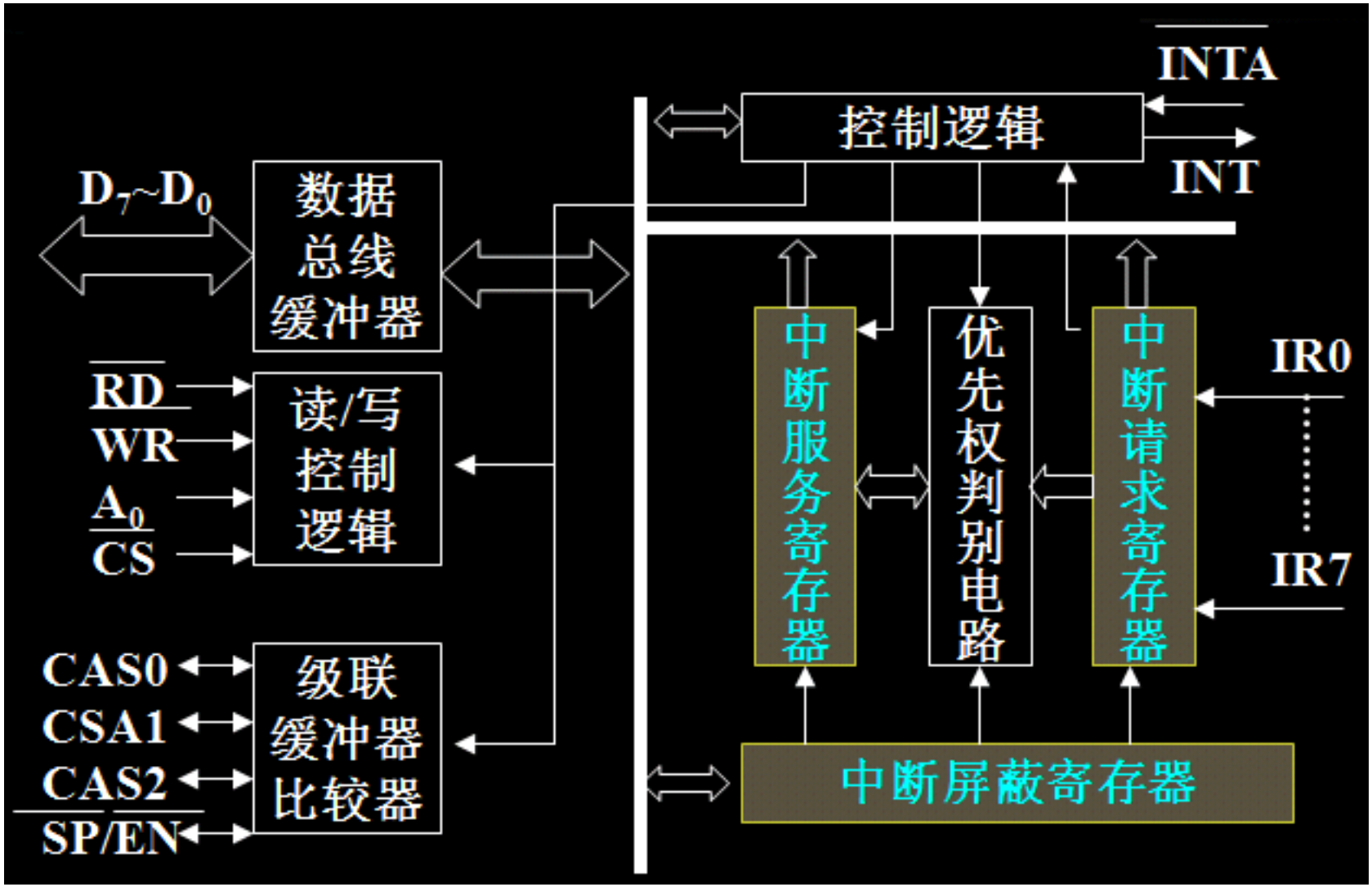
- Intel 8259A是可编程中断控制器PIC

- 可用于管理Intel 8080/8085、8086/8088、80286/80386的可屏蔽中断

7.3 8259A中断控制器

- 8259A的基本功能
  - 一片8259A可以管理8级中断，可扩展至64级
  - 每一级中断都可单独被屏蔽或允许
  - 在中断响应周期，可提供相应的中断向量号
  - 8259A设计有多种工作方式，可通过编程选择

7.3.1 8259A的内部结构和引脚



1. 中断控制

- 中断请求寄存器IRR
  - 保存8条外界中断请求信号IR0～IR7的请求状态
  - Di位为1表示IRi引脚有中断请求；为0表示无请求
- 中断服务寄存器ISR
  - 保存正在被8259A服务着的中断状态
  - Di位为1表示IRi中断正在服务中；为0表示没有被服务
- 中断屏蔽寄存器IMR
  - 保存对中断请求信号IR的屏蔽状态
  - Di位为1表示IRi中断被屏蔽（禁止）；为0表示允许

2. 与处理器接口

A_0	RD*	WR*	CS*	功能
0	1	0	0	写入ICW1、OCW2和OCW3
1	1	0	0	写入ICW2～ICW4和OCW1
0	0	1	0	读出IRR、ISR和查询字
1	0	1	0	读出IMR
×	1	1	0	数据总线高阻状态
×	×	×	1	数据总线高阻状态

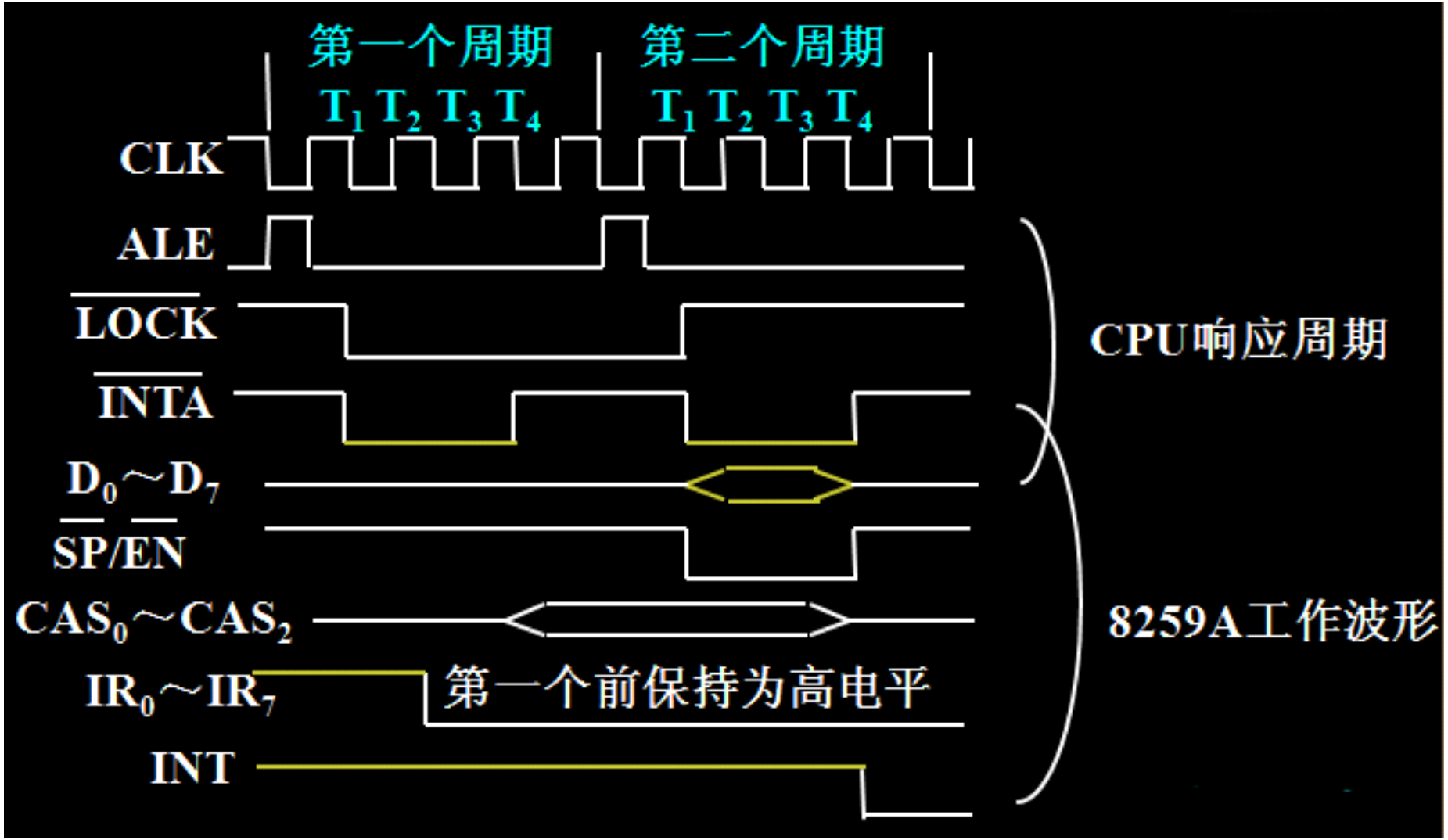
3. 中断级连

- 一个系统中，8259A可以级连，有一个主8259A，若干个（最多8个）从8259A
- 级连时，主8259A的三条级连线CAS0～CAS2作为输出线，连至每个从8259A的CAS0～CAS2

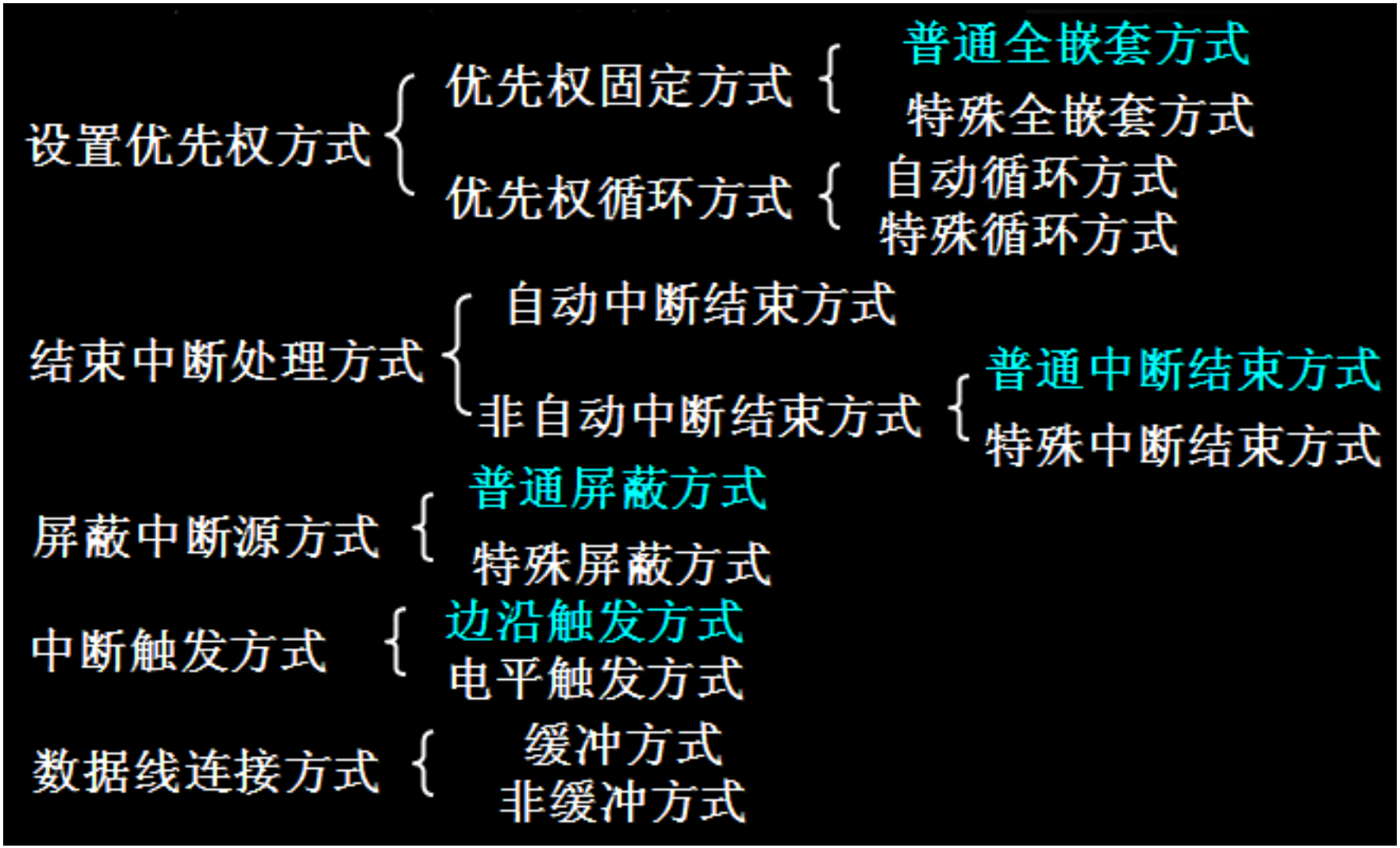
3. 中断级连

- 每个从8259A的中断请求信号INT，连至主8259A的一个中断请求输入端IR
- 主8259A的INT线连至CPU的中断请求输入端
- SP*/EN*在非缓冲方式下，规定该8259A是主片（SP*＝1）还是从片（SP*＝0）

7.3.2 8259A的中断过程



7.3.3 8259A的工作方式



1. 设置优先权方式

- 普通全嵌套方式
  - 8259A的中断优先权顺序固定不变，从高到低依次为IR0、IR1、IR2、……IR7
  - 中断请求后，8259A对当前请求中断中优先权最高的中断IRi予以响应，将其向量号送上数据总线，对应ISR的Di位置位，至到中断结束（ISR的Di位复位）
  - 在ISR的Di位置位期间，禁止再发生同级和低级优先权的中断，但允许高级优先权中断的嵌套

1. 设置优先权方式

- 特殊全嵌套方式

- 级联情况下，允许主片在某级从片进行中断服务时响应同一级从片来的另一级申请，发生嵌套

- 从片工作在普通完全嵌套方式

1. 设置优先权方式

- 优先权自动循环方式

  用于多个优先级相等中断源场合，服务完优先级降为最低

	76543210	76543210
IRR	00010100	00010000
ISR	00000100	00010000
Prio	76543210	43210765

1. 设置优先权方式

- 优先权特殊循环方式

最低优先级由编程时设置实现。若IR5为最低时，则IR6最高。

2. 结束中断处理方式

- 什么是8259A的中断结束？

- 8259A利用中断服务寄存器ISR判断：
  - 某位为1，表示正在进行中断服务；
  - 该位为0，就是该中断结束服务。
- 这里说明如何使ISR某位为0，
- 不反映CPU的工作状态。

2. 结束中断处理方式

- 自动中断结束方式

- 程序设定AEOI，用于单片，不会发生嵌套时，中断响应过程结束后，ISR中相应位清零，此时CPU刚开始执行中断服务程序。

2. 结束中断处理方式

- 普通中断结束方式
  - 配合全嵌套优先权方式使用
  - 当CPU用输出指令往8259A发出普通中断结束EOI命令时，8259A就会把所有正在服务的中断中优先权最高的ISR位复位

2. 结束中断处理方式

- 特殊中断结束方式
  - 配合循环优先权方式使用
  - CPU在程序中向8259A发送一条特殊中断结束命令，这个命令中指出了要清除哪个ISR位

3. 屏蔽中断源方式

- 普通屏蔽方式
  - 将IMR的Di位置1，则对应的中断IRi被屏蔽，该中断请求不能从8259A送到CPU
  - 如果IMR的Di位置0，则允许IRi中断产生
- 特殊屏蔽方式
  - 将IMR的Di位置1，对应的中断IRi被屏蔽的同时，使ISR的Di位置0

4. 中断触发方式

- 边沿触发方式
  - 8259A将中断请求输入端出现的上升沿作为中断请求信号

- 电平触发方式
  - 中断请求端出现的高电平是有效的中断请求信号

5. 数据线连接方式

- 缓冲方式
  - 8259A的数据线需加缓冲器予以驱动
  - 8259A把SP*/EN*引脚作为输出端，输出允许信号，用以锁存或开启缓冲器
- 非缓冲方式
  - SP*/EN*引脚为输入端
  - 若8259A级连，由其确定是主片或从片

7.3.4 8259A的编程

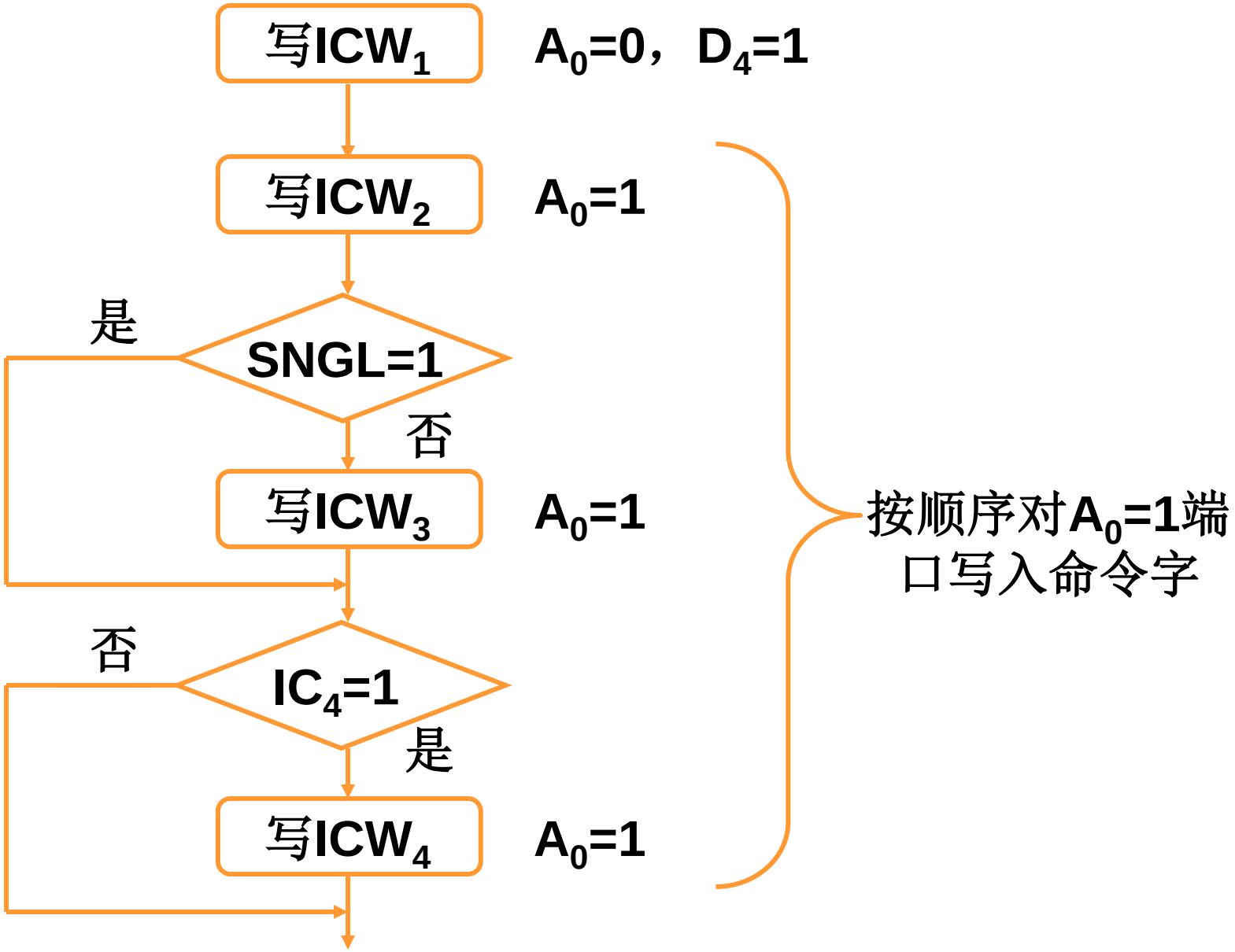
- 初始化编程
  - 8259A开始工作前，必须进行初始化编程
  - 给8259A写入初始化命令字ICW
- 中断操作编程
  - 在8259A工作期间
  - 可以写入操作命令字OCW将选定的操作命令传送给8259A，使之按要求完成操作
  - 还可以读取8259A的信息，以便了解他的工作状态

1. 初始化命令字ICW

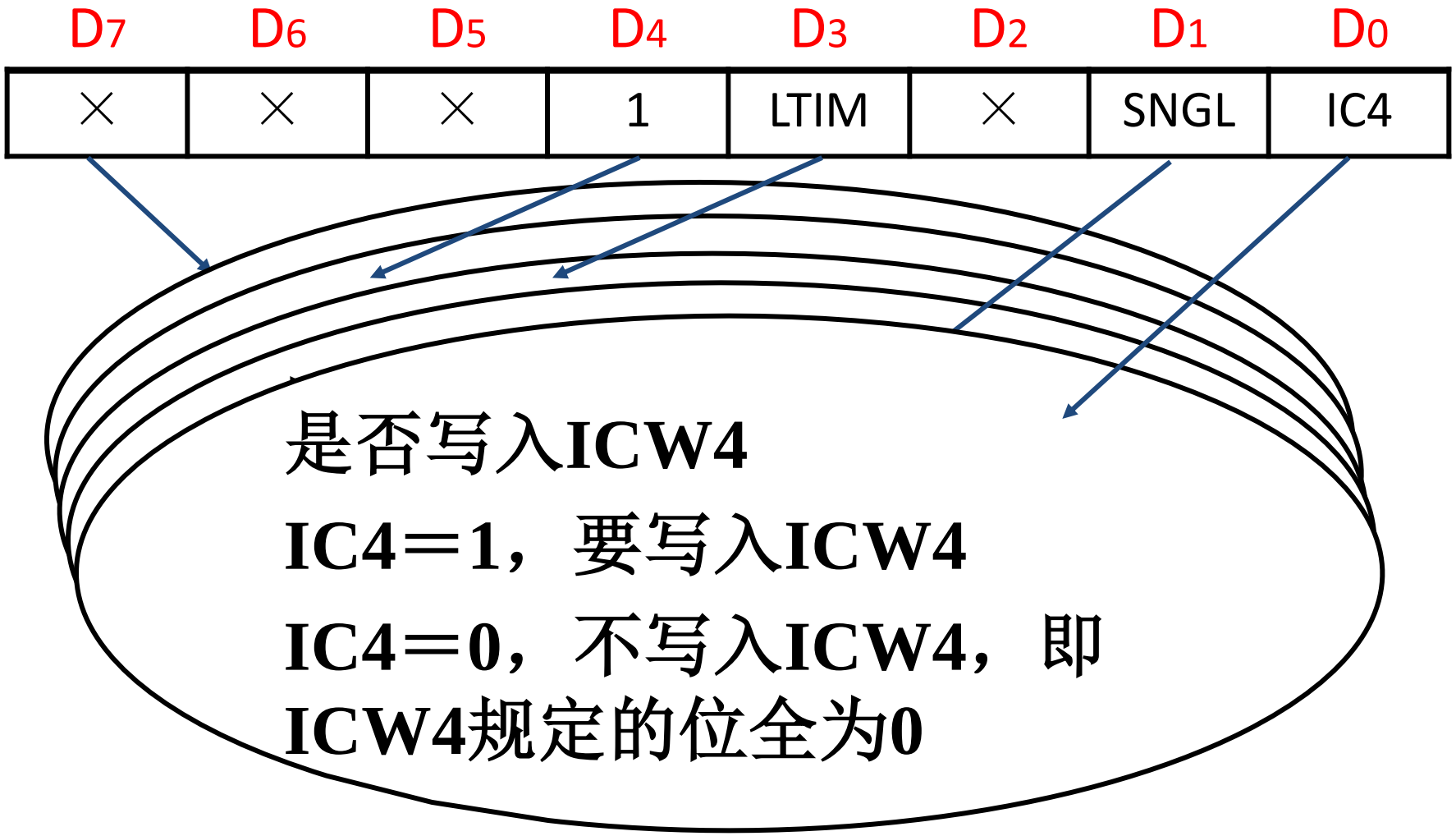
- 初始化命令字ICW最多有4个
- 8259A在开始工作前必须写入
- 必须按照ICW1～ICW4顺序写入

- ICW1和ICW2是必须送的
- ICW3和ICW4由工作方式决定

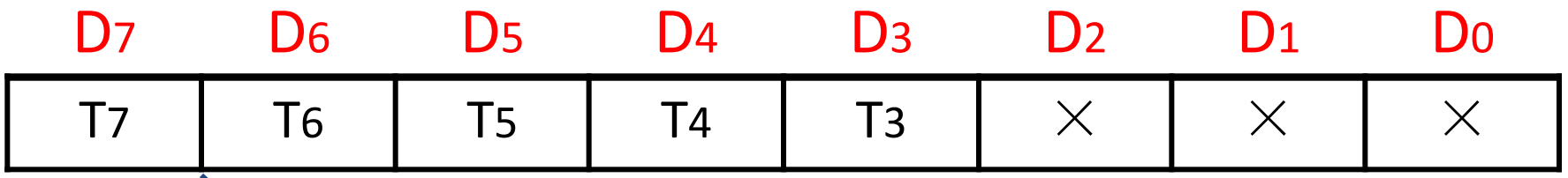
8259A芯片的初始化流程



ICW1


D7 D6 D5 D4 D3 D2 D1 D0
× × × 1 LTIM × SNGL IC4
是否写入ICW4
IC4＝1，要写入ICW4
IC4＝0，不写入ICW4，即ICW4规定的位全为0

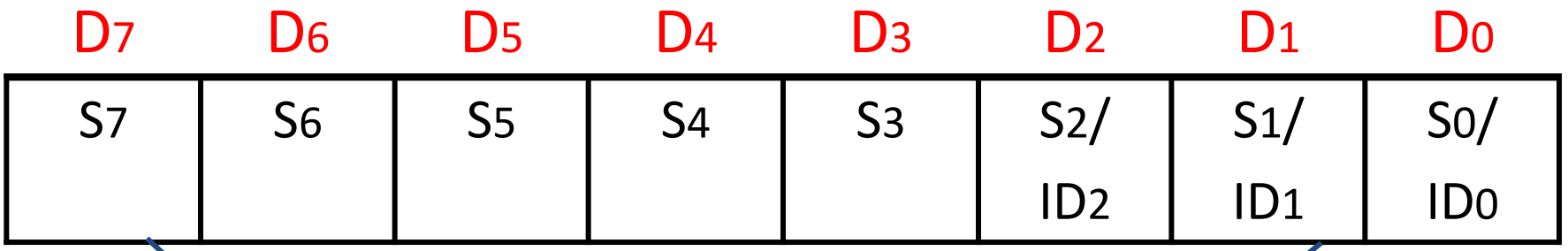
ICW2



设置中断向量号

- **T7～T3为中断向量号的高5位**
- **低3位由8259A自动确定：**
- **IR0为000、IR1为001、……、IR7为111**

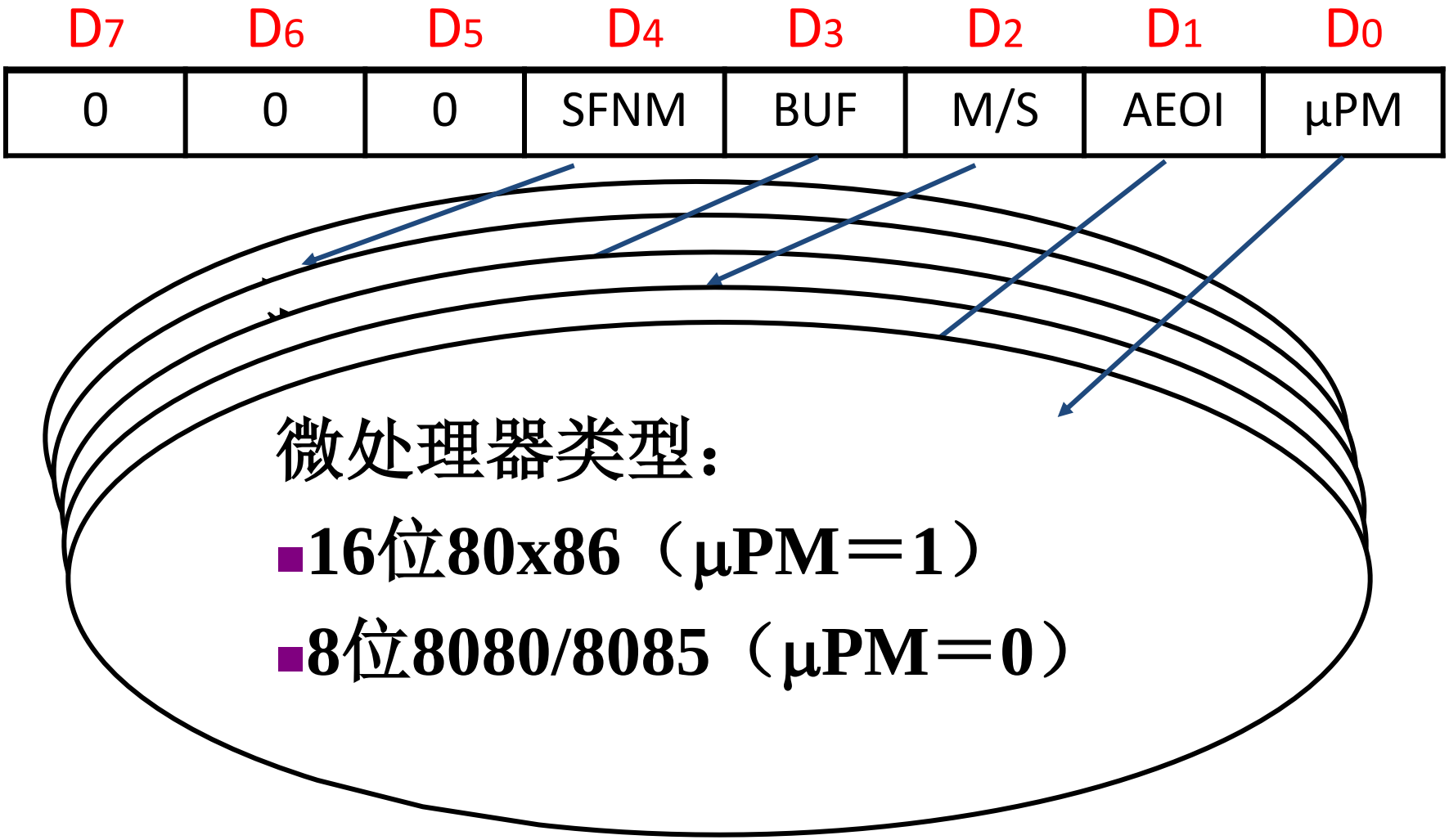
ICW3



级连命令字

- 主片**8259A**：**Si**＝**1**对应**IRi**接有从片；否则**IRi**没有连接从片
- 从片**8259A**：**ID0**～**ID2**编码说明从片**INT**引脚接到主片哪个**IR**引脚

ICW4


D7 D6 D5 D4 D3 D2 D1 D0
0 0 0 SFNM BUF M/S AEOI μPM
微处理器类型：
16位80x86（μPM＝1）
8位8080/8085（μPM＝0）

中断控制器的初始化程序段——初始化主片8259A

```
        mov al,11h      ;写入ICW1
        out 20h,al
        jmp intr1
intr1:  mov al,08h      ;写入ICW2
        out 21h,al
        jmp intr2
intr2:  mov al,04h      ;写入ICW3
        out 21h,al
        jmp intr3
intr3:  mov al,1h       ;写入ICW4
        out 21h,al
```

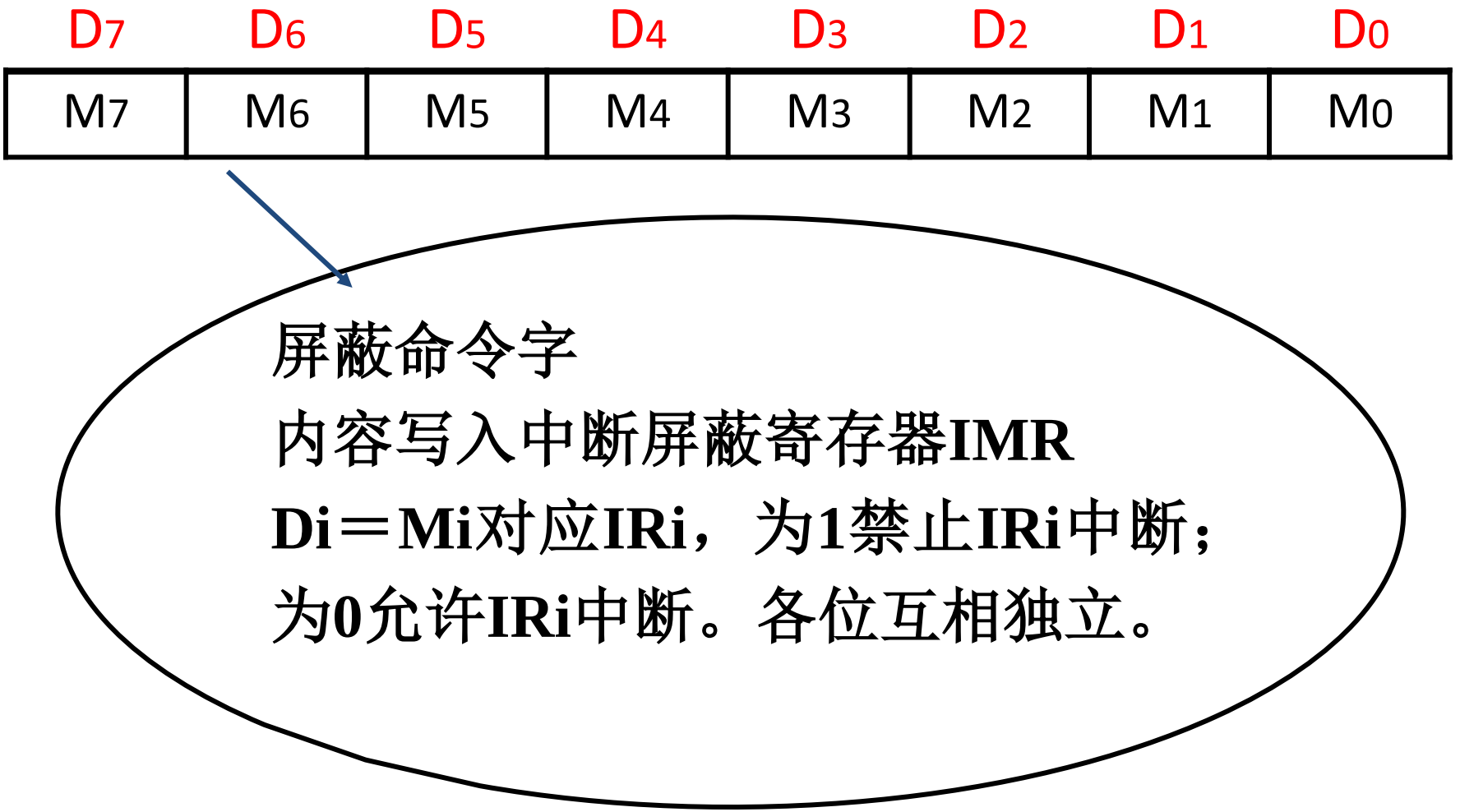
中断控制器的初始化程序段——初始化从片8259A

```
        mov al,11h      ;写入ICW1
        out 0a0h,al
        jmp intr5
intr5:  mov al,70h      ;写入ICW2
        out 0a1h,al
        jmp intr6
intr6:  mov al,02h      ;写入ICW3
        out 0a1h,al
        jmp intr7
intr7:  mov al, 01h     ;写入ICW4
        out 0a1h,al
```

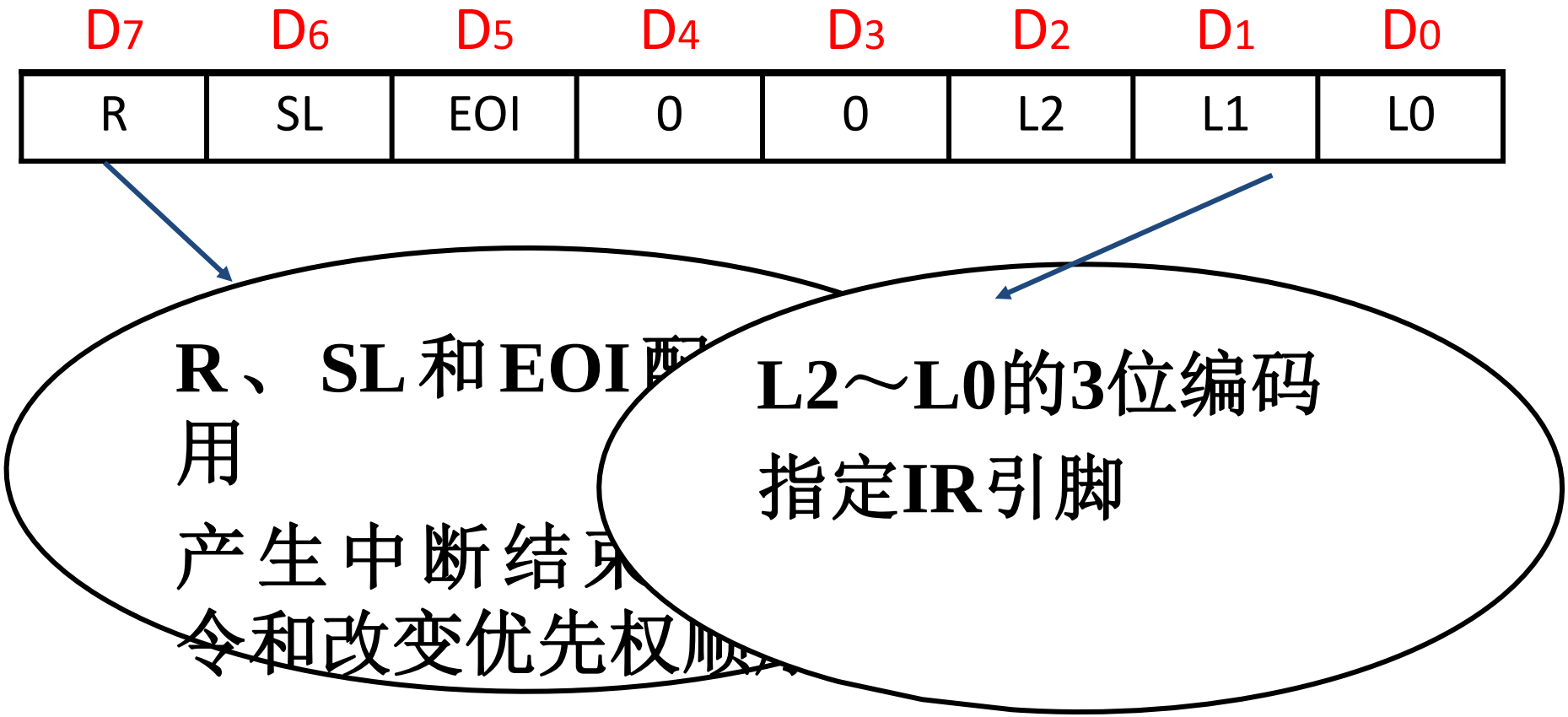
2. 操作命令字OCW

- 8259A工作期间，可以随时接受操作命令字OCW
- OCW共有3个：OCW1～OCW3
- 写入时没有顺序要求，需要哪个OCW就写入那个OCW

OCW1


D7 D6 D5 D4 D3 D2 D1 D0
M7 M6 M5 M4 M3 M2 M1 M0
屏蔽命令字
内容写入中断屏蔽寄存器IMR
Di＝Mi对应IRi，为1禁止IRi中断；
为0允许IRi中断。各位互相独立。

OCW2


D7 D6 D5 D4 D3 D2 D1 D0
R SL EOI 0 0 L2 L1 L0
R、SL和EOI配
用
产生中断结束
令和改变优先权顺
L2～L0的3位编码
指定IR引脚

OCW2

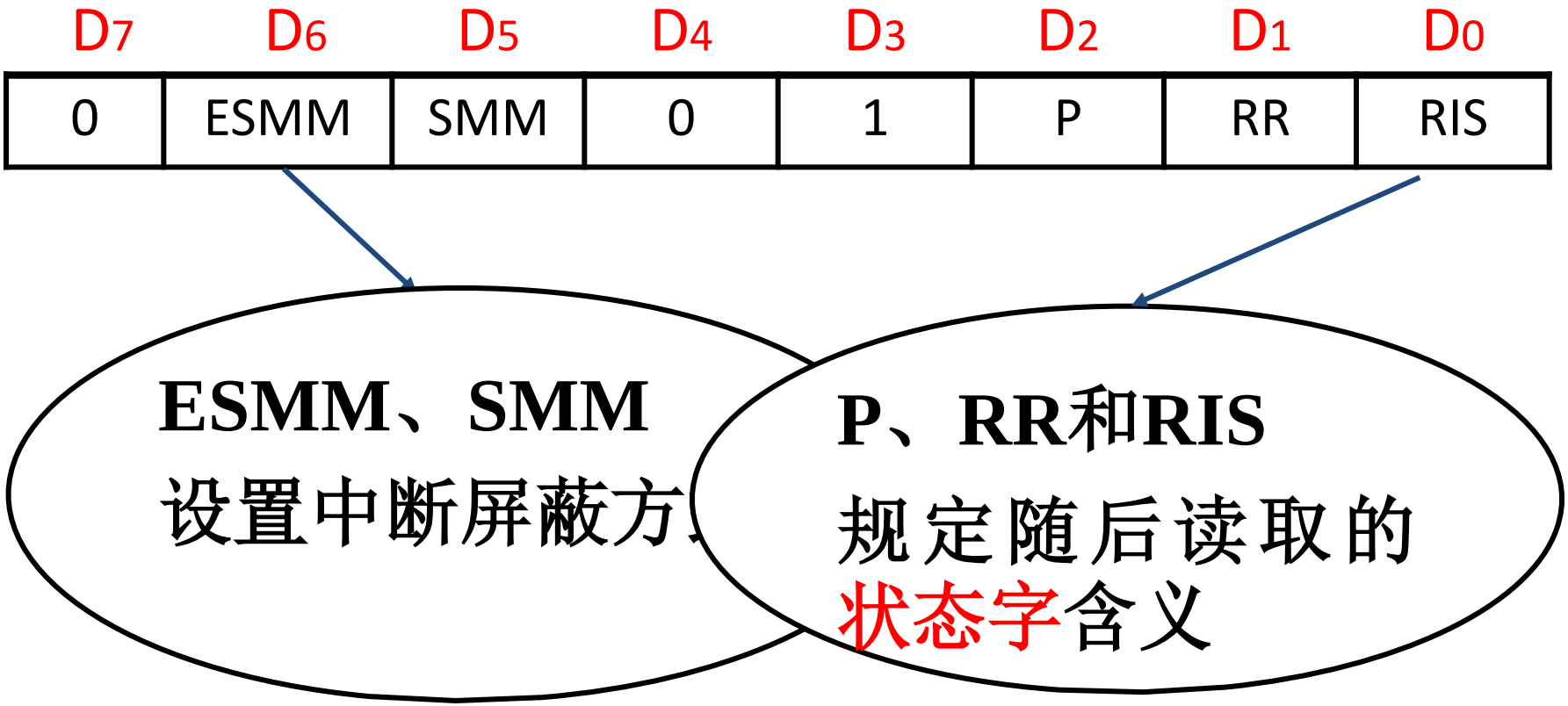
R=1 优先级按循环方式设置

SL（SET LEVEL）=1　L2，L1，L0有效

EOI=1 发中断结束命令，对当前指定ISR_i复位

0	0	1	一般EOI命令
0	1	1	特殊EOI命令，对L2~L0指定的ISR_i复位
1	0	1	一般EOI命令下的循环优先
1	0	0	设置自动的循环优先
0	0	0	清除自动循环优先
1	1	1	EOI命令后，按L2~L0指定级别循环优先
1	1	0	特殊循环优先方式中设置优先级
0	1	0	无操作

OCW3


D7 D6 D5 D4 D3 D2 D1 D0
0 ESMM SMM 0 1 P RR RIS
ESMM、SMM
设置中断屏蔽方
P、RR和RIS
规定随后读取的
状态字含义

OCW3

ESMM	SMM	
1	**1**	**特殊屏蔽方式置位**
1	**0**	**特殊屏蔽方式复位**
0	**x**	**无操作**

P（Polling）：**=1，查询方式**

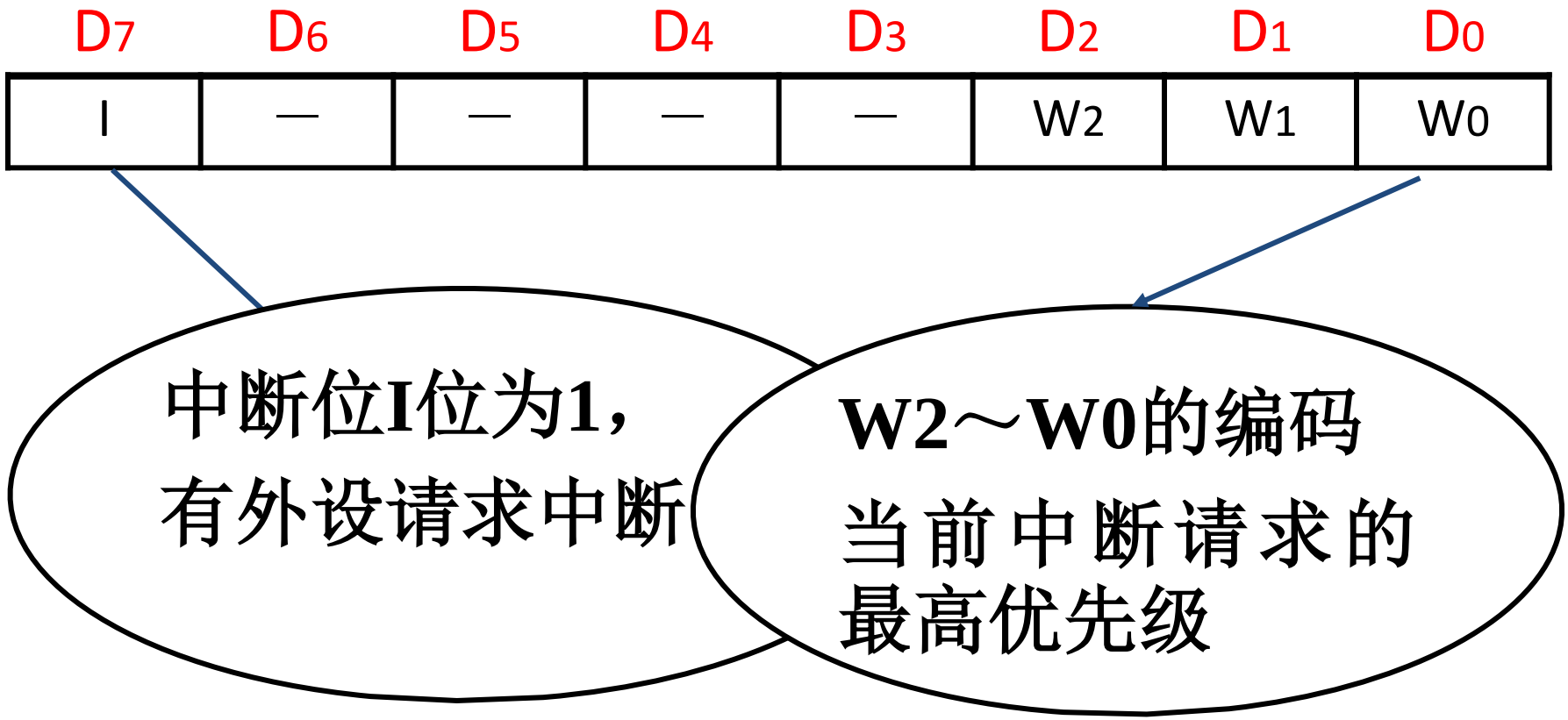
=0，读内部REG

RR	RIS	
1	**1**	**读ISR**
1	**0**	**读IRR**
0	**x**	**非读（无操作）**

3. 读取状态字

- CPU可读出IRR、ISR、IMR和查询字
- A0为低，由OCW3中RR和RIS位设定读取IRR或ISR，由OCW3中P位设定读取查询字
- 而A0引脚为高电平时读取的都是IMR
- 查询字反映8259A是否有中断请求

查询字


D7 D6 D5 D4 D3 D2 D1 D0
I — — — — W2 W1 W0
中断位I位为1，
有外设请求中断
W2～W0的编码
当前中断请求的
最高优先级

4. 命令字和状态字的区别方法

(1) 利用读写信号区别写入的控制寄存器和读出的状态寄存器

(2) 利用地址信号区别不同I/O地址的寄存器

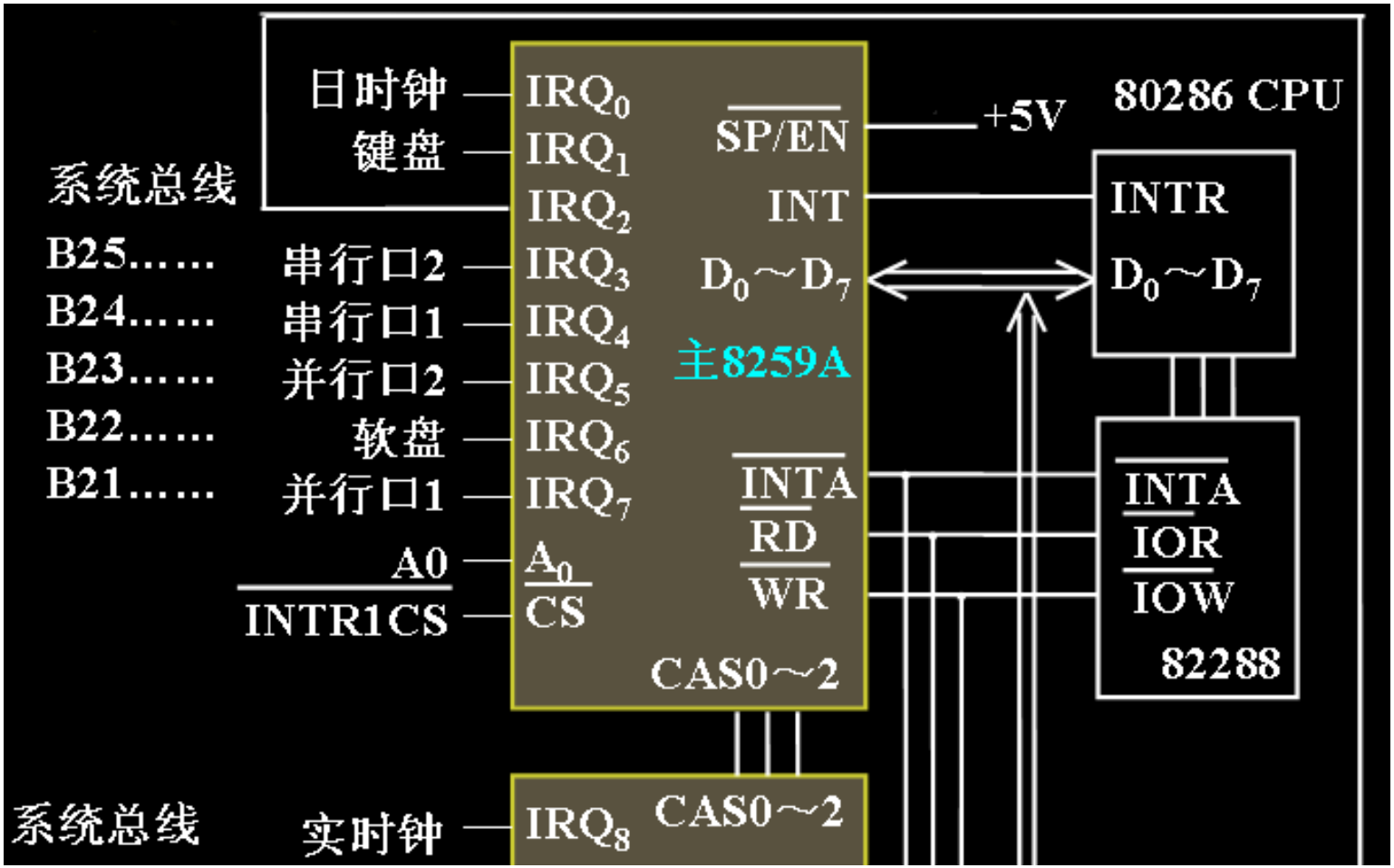
(3) 由控制字中的标志位说明是哪个寄存器

(4) 由芯片内顺序控制逻辑按一定顺序识别不同的寄存器

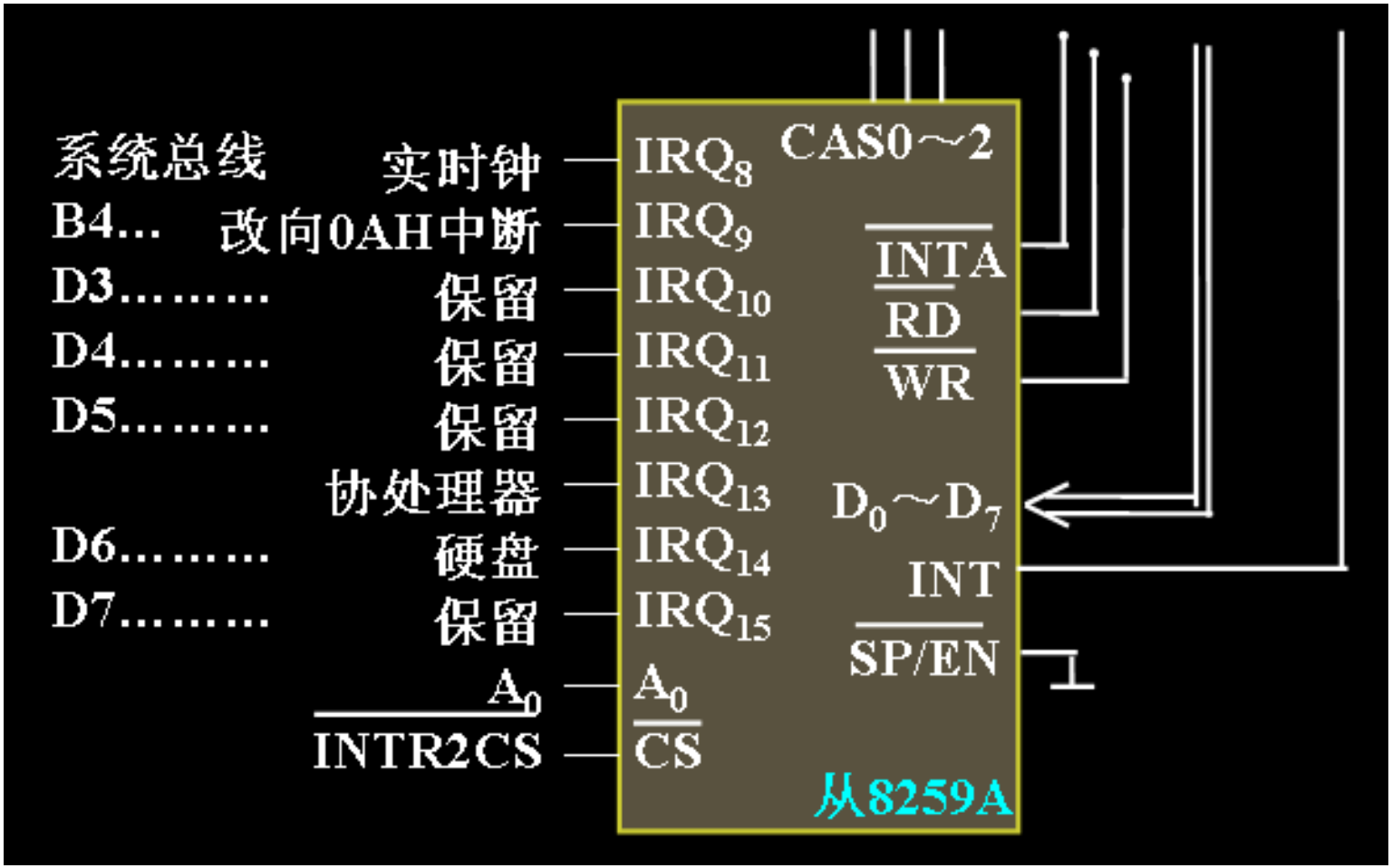
(5) 由前面的控制字决定后续操作的寄存器

这些都是接口电路中常用的方法。

7.4 8259A在IBM PC系列机上的应用



7.4 8259A在IBM PC系列机上的应用



应用注意事项

- 利用上升沿做为中断请求IRQ的有效信号
- IRQ0～IRQ7的中断向量号依次为08H～0FH，IRQ8～IRQ15依次为70H～77H
- 采用普通全嵌套优先权方式，中断优先权从高到低顺序为IRQ0～IRQ2、IRQ8～IRQ15、IRQ3～IRQ7，且不能改变

应用注意事项

- 采用普通中断结束EOI方式，需要在中断服务程序最后发送普通EOI命令；如果针对从片中断源，需要发送2个EOI
- 一般采用普通屏蔽方式，通过写入IMR允许中断，但注意不要破坏原屏蔽状态

7.5 外部中断服务程序

编写外部可屏蔽中断服务程序，需注意：

- 发送中断结束命令（非自动中断结束）
- 一般只能采用存储单元传递参数（随机性决定）
- 不要使用DOS系统功能调用（避免循环调用）
- 中断服务程序尽量短小（尽量少占据内存空间）

7.5 外部中断服务程序

编写主程序，需注意：

- 修改中断向量
- 控制CPU的中断允许标志
- 设置8259A的中断屏蔽寄存器

例题 可屏蔽中断服务程序

- 8259A的IRQ0（向量号为08H）中断请求来自定时器8253，每隔55ms产生一次
- 本程序的08H号中断服务程序，每次中断显示一串信息，显示10次
- 用内存单元（共享变量）在主程序与外部中断服务程序之间传递参数：中断次数
- 显示信息也安排在共同的数据段中

例1 数据段

```
intmsg    db ‘A 8259A Interrupt !’
          db 0dh,0ah,0
counter   db 0
```

例1 保存中断向量

```
mov ax,3508h
int 21h
push bx      ;保存偏移地址
push es      ;保存段基地址
```

例1 设置中断向量

```
cli
push ds
mov dx,offset new08h
mov ax,seg new08h
mov ds,ax
mov ax,2508h
int 21h
pop ds
```

例1 设置中断寄存器

```
in al,21h
push ax
and al,0feh          ;允许IRQ0
out 21h,al
mov counter,0        ;设置中断次数初值
sti                  ;开中断
```

例1 循环等待中断

```
start1:     cmp counter,10
            jb start1              ;中断10次退出
```

例1 主程序结束

```
cli
pop ax
out 21h,al
pop dx
pop ds
mov ax,2508h
int 21h
sti
mov ax,4c00h
int 21h
```

A 8259A Interrupt !
A 8259A Interrupt !
A 8259A Interrupt !
A 8259A Interrupt !
A 8259A Interrupt !
A 8259A Interrupt !
A 8259A Interrupt !
A 8259A Interrupt !
A 8259A Interrupt !
A 8259A Interrupt !

例1 进入中断服务程序

```
new08h    proc
          sti              ;开中断
          push ax          ;保护寄存器
          push bx
          push ds
          mov ax,data
          mov ds,ax  ;设置数据段DS
```

例1 中断处理

```
inc counter
mov si,offset intmsg   ;显示信息
call dpstri
```

例1 退出中断服务程序

```
        mov al,20h
        out 20h,al
        pop ds          ;恢复寄存器
        pop bx
        pop ax
        iret            ;中断返回
new08h endp
```

例1 显示字符串

```
dpstri proc        ;显示字符串子程序
        push ax
        push bx
dps1:   lodsb
        cmp al,0
        jz dps2
```

例1 显示字符串

```
        mov bx,0
        mov ah,0eh
        int 10h
        jmp dps1
dps2:   pop bx
        pop ax
        ret
dpstri endp
```

7.6 驻留中断服务程序

驻留TSR（Terminate and Stay Resident）程序

- 用户程序运行后仍然保存在主存中，可以让其他程序使用
- 利用DOS功能调用31H代替4CH终止程序
- 小型驻留程序常编写成COM程序
- 驻留程序也可以编写成EXE程序
- 需要驻留内存的程序段要写在前面

例题2 报时中断驻留服务程序

- 系统08H号中断服务程序调用1CH中断
- 每隔55ms调用这个报时中断，中断65543次就是时间过了一个小时
- 本例编写一个驻留内存的1CH内部中断服务程序
- 实现每过一小时就显示信息
- 执行此程序后，报时中断服务程序将驻留内存

例2 进入中断服务程序

```
new1ch proc
        sti                 ;开中断
        push si             ;保护寄存器
        push ds
        mov si,cs
        mov ds,si           ;设置数据段DS
        add countl,1
        adc counth,0
```

例2 中断处理

```
cmp countl,hourl
jnz   n1ch1
cmp counth,hourh
jnz   n1ch1
mov countl,0
mov counth,0
mov si,offset intmsg
call  dpstri
```

例2 退出中断服务程序

```
n1ch1:   pop ds   ;恢复寄存器
         pop si
         iret     ;中断返回
         countl   dw 0
         counth   dw 0
         intmsg   db 'One Hour Has Passed ! '
         db 0dh,0ah,0
new1ch endp
```

One Hour Has Passed !

例2 显示字符串

```
dpstri proc          ;显示字符串子程序
       push ax
       push bx
dps1:  lodsb
       cmp al,0
       jz dps2
```

例2 显示字符串

```
        mov bx,0
        mov ah,0eh
        int 10h
        jmp dps1
dps2: pop bx
        pop ax
        ret
dpstri endp
```

例2 主程序开始

```
start:  mov ax,cs
        mov ds,ax
        mov dx,offset new1ch
        cli
        mov ax,251ch
        int 21h
        sti
        mov dx,offset tsrmsg
        mov ah,09h
        int 21h
```

例2 主程序结束

```
mov dx,offset start
add  dx,15
mov cl,4
shr   dx,cl
add  dx,10h
mov ax,3100h        ;程序驻留
int 21h
 tsrmsg          db  ‘INT 1CH Program Installed !’
db 0dh,0ah,'$'
end start
```

第7章教学要求

- 1. 熟悉8088的中断类型、中断响应过程、中断向量表
- 2. 掌握内部中断服务程序的编写
- 3. 理解8259A的内部结构、寄存器作用、中断过程
- 4. 掌握8259A的普通全嵌套优先权、普通中断结束、边沿触发方式

第7章教学要求

- 5. 了解的8259A的ICW和OCW，注意命令字和状态字的区别方法
- 6. 了解8259A在IBM PC系列机上的应用情况
- 7. 掌握外部中断服务程序的编写