



# 第7章 中断控制接口



# 第7章 中断控制接口



## 学习重点

- ◇ 8088 CPU 的中断系统
- ◇ 中断控制器8259A
- ◇ 中断服务程序的编写

## 7.1 8088中断系统

- ◇ 8088能够处理256个中断，用中断类型号0 ~ 255标识
- ◇ 8088的中断系统采用向量中断机制

0000	0000
0000	0001
0000	0010
	⋮
	⋮
1111	1110
1111	1111

## 7.1.1 8088的中断类型

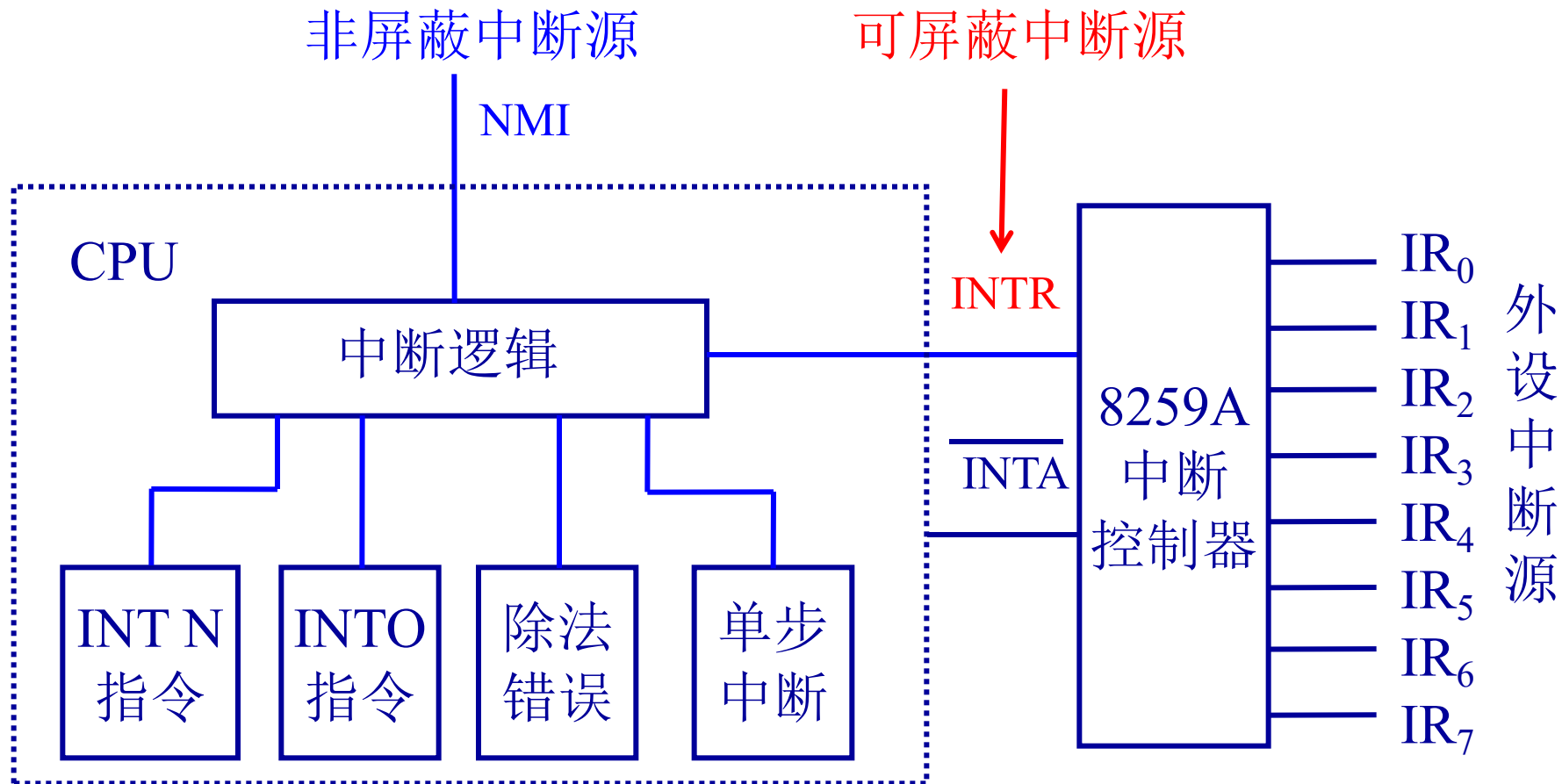
### ◇ 内部中断

- ◆ 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(NMI)
- ◆ 可屏蔽中断(INTR)

## 7.1.1 8088的中断类型



# 1. 内部中断

- ◇ 内部中断是8088因执行程序出现异常而产生的中断。
- ◇ 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发生的异常情况的有效途径。

# (1) 除法错中断

- ◇ 8088执行除法指令时，若除数为0或商超过了约定的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。

例如：

```
mov bl,0
```

```
idiv bl      ; 除数bl=0，产生除法错中断
```

```
mov ax,200h
```

```
mov bl,1
```

```
div bl      ; 商=200H，不能用AL存储  
            ; 产生除法错中断
```

## (2) 单步中断

- ◇ 若单步中断标志**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

15	12	11	10	9	8	7	6	5	4	3	2	1	0
		<b>OF</b>	<b>DF</b>	<b>IF</b>	<b>TF</b>	<b>SF</b>	<b>ZF</b>		<b>AF</b>		<b>PF</b>		<b>CF</b>



### (3) 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 (指令代码: 11001100) 常用于程序调试, 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。

## (4)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
```

15    12    11    10    9    8    7    6    5    4    3    2    1    0

**OF**

**DF**

**IF**

**TF**

**SF**

**ZF**

**AF**

**PF**

**CF**

## 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。
- ◇ 外部中断是处理器外部随机产生的，所以是真正的中断 (*Interrupt*) 。
- ◇ 内部中断是处理器执行程序出现异常导致的，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输入引脚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。
- ◇ 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

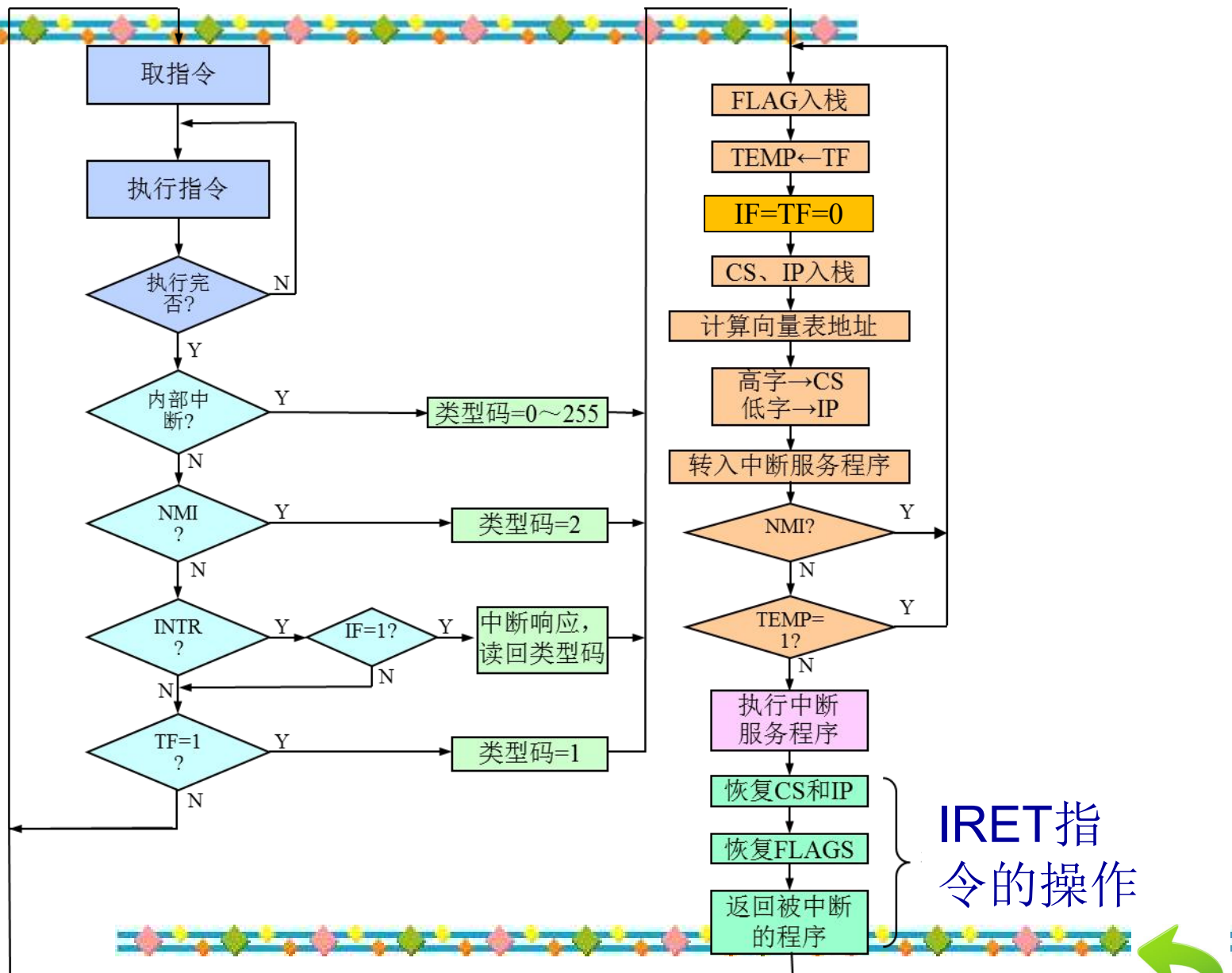
## (2) 可屏蔽中断

- ◇ IO设备接口电路通过可屏蔽中断请求信号输入引脚(INTR)向微处理器发出的中断请求，微处理器在允许可屏蔽中断的条件下(**IF=1**)，在当前指令执行结束时予以响应，同时输出可屏蔽中断响应信号(INTA\*)。
- ◇ 8088用中断允许标志**IF**控制是否响应可屏蔽中断请求；如果响应，由外部中断控制器接口电路提供中断向量号。
- ◇ 8088通常需要在中断控制器8259A的配合下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。

# 中断允许标志IF的状态

- ◇ **IF = 0**：可屏蔽中断不会被响应
  - ◆ 系统此时称为关中断、禁止中断或中断屏蔽；
  - ◆ 系统复位时，**IF = 0**；
  - ◆ 任何一个中断被响应，使**IF = 0**；
  - ◆ 执行指令**CLI**，使**IF = 0**。
- ◇ **IF = 1**：可屏蔽中断会被响应
  - ◆ 系统此时称为开中断、允许中断或中断开放；
  - ◆ 执行指令**STI**，使**IF = 1**。
- ◇ 执行指令**IRET**恢复原**IF**状态

## 7.1.2 8088的中断响应过程



## 7.1.2 8088的中断响应过程

查询中断源的顺序，决定了各种中断的优先级别。

8088的各种中断按优先级从高到低排列如下：

◇ 软件中断

◆ 除法错中断

◆ 指令中断

◆ 溢出中断

这3种为同级中断，不会同时发生

◇ 非屏蔽中断

◇ 可屏蔽中断

◇ 单步中断



### 7.1.3 8088的中断向量表

- ◇ 中断向量：中断服务程序的入口地址。
  - ◆ 入口地址为逻辑地址，包括段地址和偏移地址(CS:IP)。
  - ◆ 每个中断向量在内存需占用4个字节，其中低地址字存储偏移地址、高地址字存储段地址。
- ◇ 8088微处理器在内存物理地址00000H~003FFH的1KB的空间内，依次安排256个中断的中断向量，形成中断向量表。

向量号为N的中断向量的物理地址 =  $N \times 4$

## 7.2 中断服务程序

- ◇ 编写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**
  - ◆ 最后用**IRET**指令实现中断返回
  - ◆ 通常采用寄存器传递参数
  - ◆ 第1条指令通常为开中断指令**STI**(允许中断嵌套)
- ◇ 主程序可以调用中断服务程序
  - ◆ 利用**INT n**指令调用中断服务程序
  - ◆ 设置必要的入口参数
  - ◆ 处理出口参数
  - ◆ 调用前, 需要设置中断向量

## 例7.1 内部中断服务程序

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



A Instruction Interrupt !

## Function 0Eh Write Character

Input	AH = 0Eh AL = ASCII Code of the character BH = Active video page number. BL = Foreground color (graphics modes)
Output	AL = No registers set.
Description	Function 0Eh writes a character to the current video page at the current cursor position. The cursor column position is incremented after writing the character. If the end of a line is reached, the cursor row position is also incremented and the column position is set to 0. The ASCII control characters are: 07h = beep, 08h = backspace, 0Ah = line feed, and 0Dh = carriage return.


## 例7.1的中断服务程序（1）

- ； 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 = 缓冲区首地址


```
new80h      proc          ; 过程定义
             sti           ; 开中断
             push ax       ; 保护寄存器
             push bx
             push si
```




## 例7.1的中断服务程序（2）




```
mov si,dx
dispchar: mov al,[si]           ; 读取一个显示字符
          cmp al,0             ; 为结尾“0”，则结束
          jz  finish         ; 输出结束，转中断返回
          mov bx,0             ; 采用ROM-BIOS功能调用
          mov ah,0eh           ; 0eh号子功能，输出al中的字符
          int 10h
          inc si               ; 准备显示下一个字符
          jmp dispchar
```



## 例7.1的中断服务程序（3）



```
finish:      pop si      ; 恢复寄存器  
              pop bx  
              pop ax  
              iret       ; 中断返回  
new80h      endp        ; 过程（中断服务程序）结束
```



## 例7.1的主程序

； 数据段

intoff dw ? ; 用于保存偏移地址

intseg dw ? ; 用于保存段基地址

intmsg db 'An Instruction Interrupt !', 0dh,0ah,0

回车、换行

以“0”结尾





## 例7.1的获取原中断向量

； 代码段

mov ax,3580h ; 利用DOS功能35H号

int 21h ; 获取原80H中断向量

mov intoff,bx ; 保存偏移地址

mov intseg,es ; 保存段基地址


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的设置新中断向量



```
push ds
mov dx,offset new80h      ; 取中断程序偏移地址
mov ax,seg new80h         ; 取中断程序段地址
mov ds,ax
mov ax,2580h
int 21h
pop ds
```

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 中断调用

； 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



A Instruction Interrupt !

## 例7.1 主程序返回



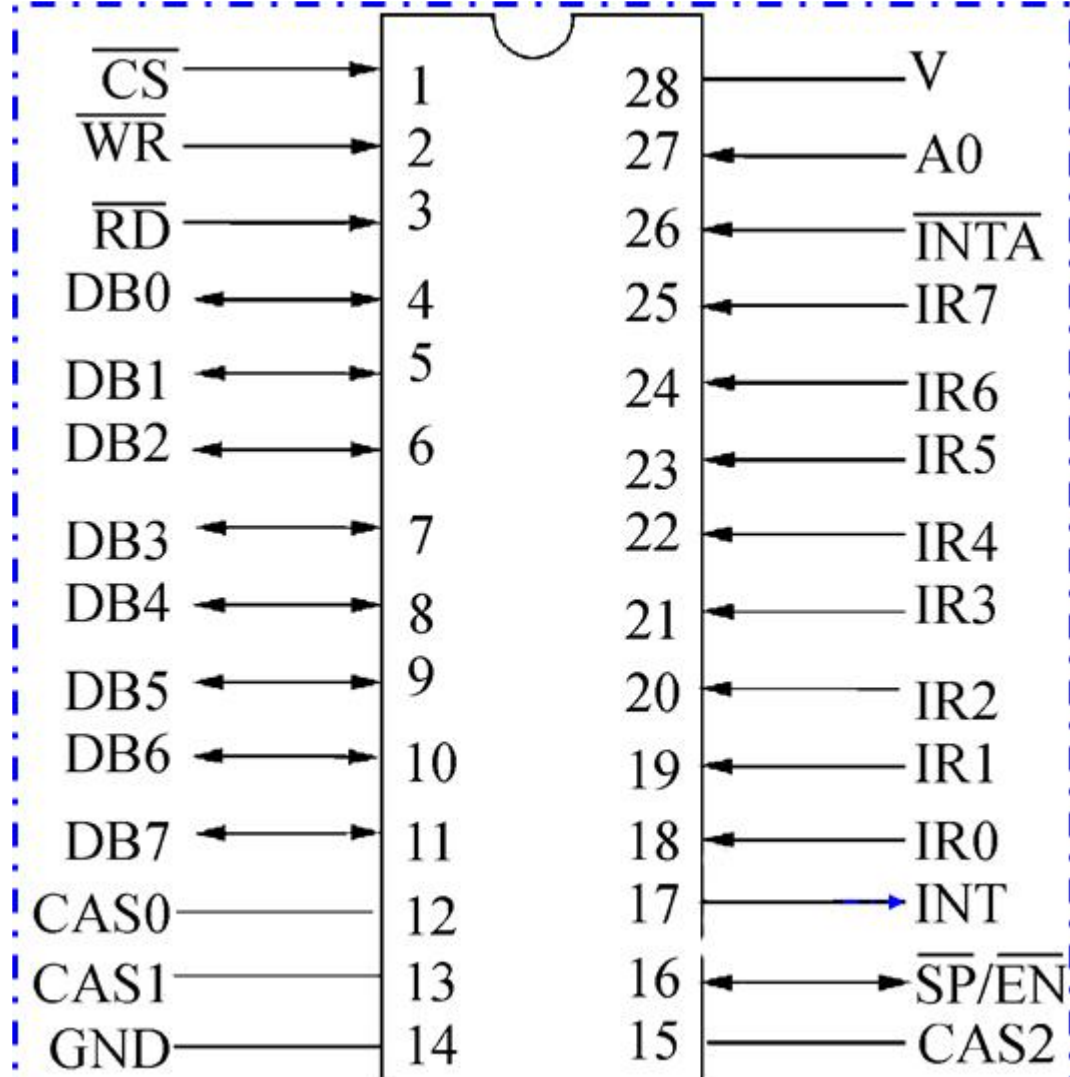
```
mov dx,intoff           ; 恢复原中断向量
mov ax,intseg
mov ds,ax                 ; 改变DS
mov ax,2580h
int  21h                  ; 因紧接着返回DOS
mov ax,4c00h              ; 故无需恢复DS
int  21h
```

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）

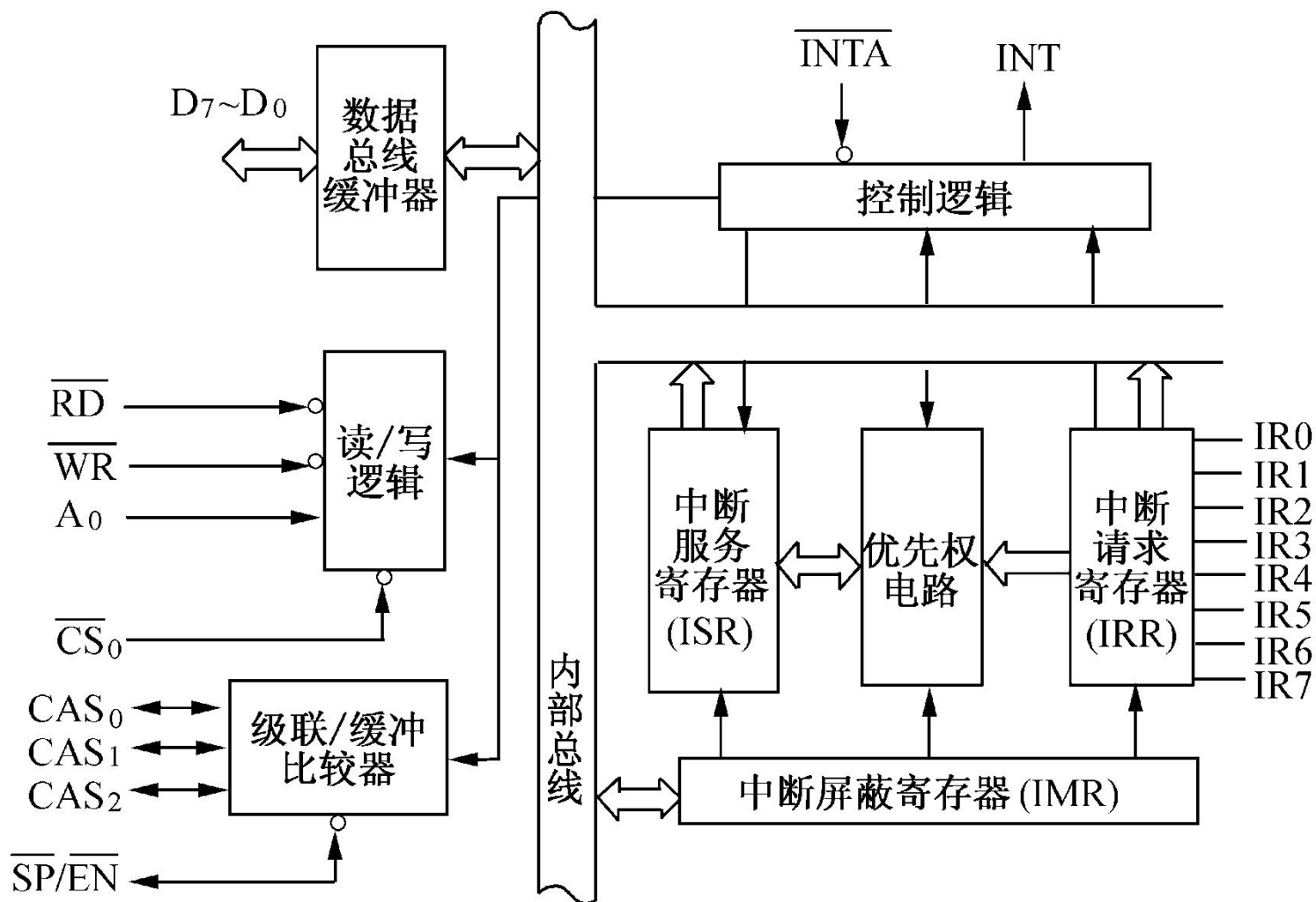
## 7.3 8259A中断控制器

- ◇ Intel 8259A是可编程中断控制器PIC
- ◇ 用于管理Intel 8080/8085、8086/8088、80286/80386的可屏蔽中断
- ◇ 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，向CPU提供相应的中断类型号
  - ◆ 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



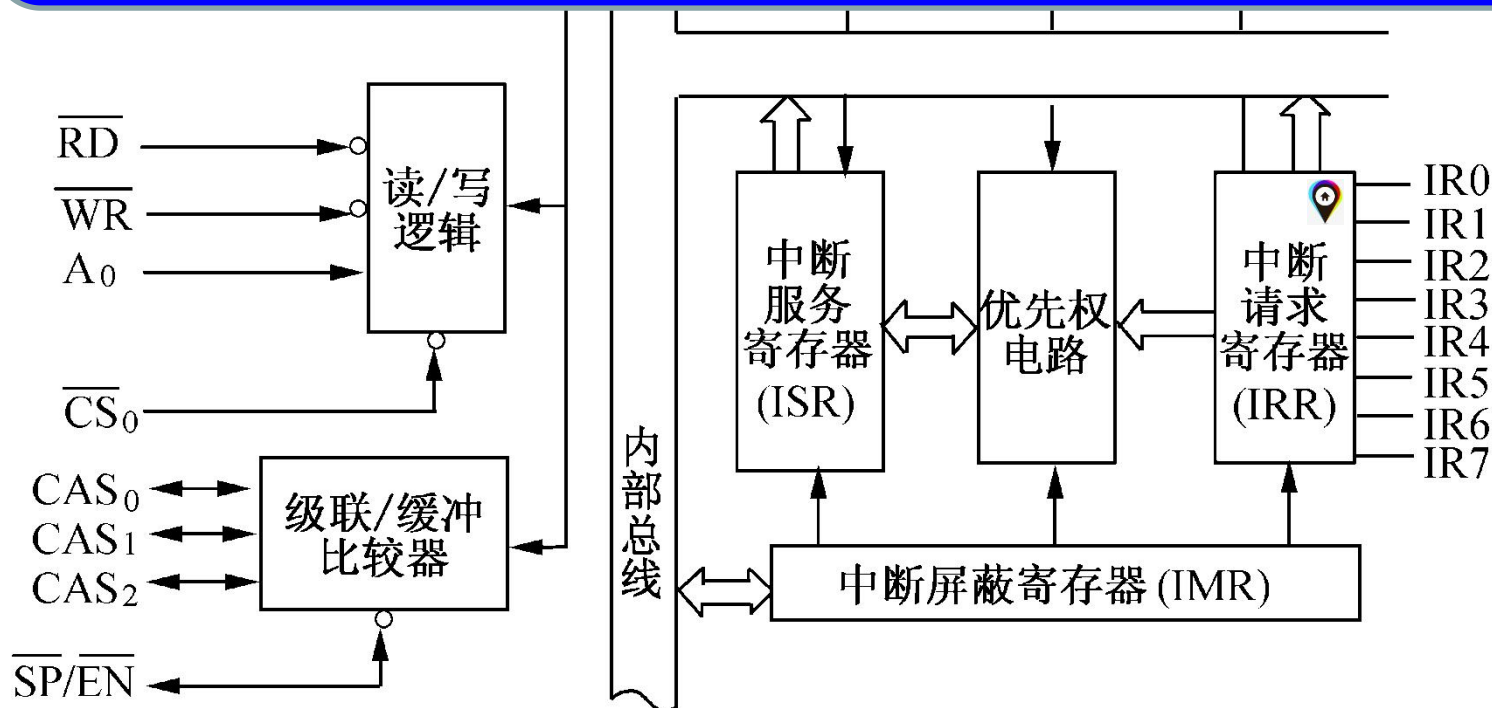
## 7.3.1 8259A的内部结构和引脚



## 7.3.1 8259A的内部结构和引脚

### 中断请求寄存器IRR

- ✧ 保存8条外部中断请求信号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

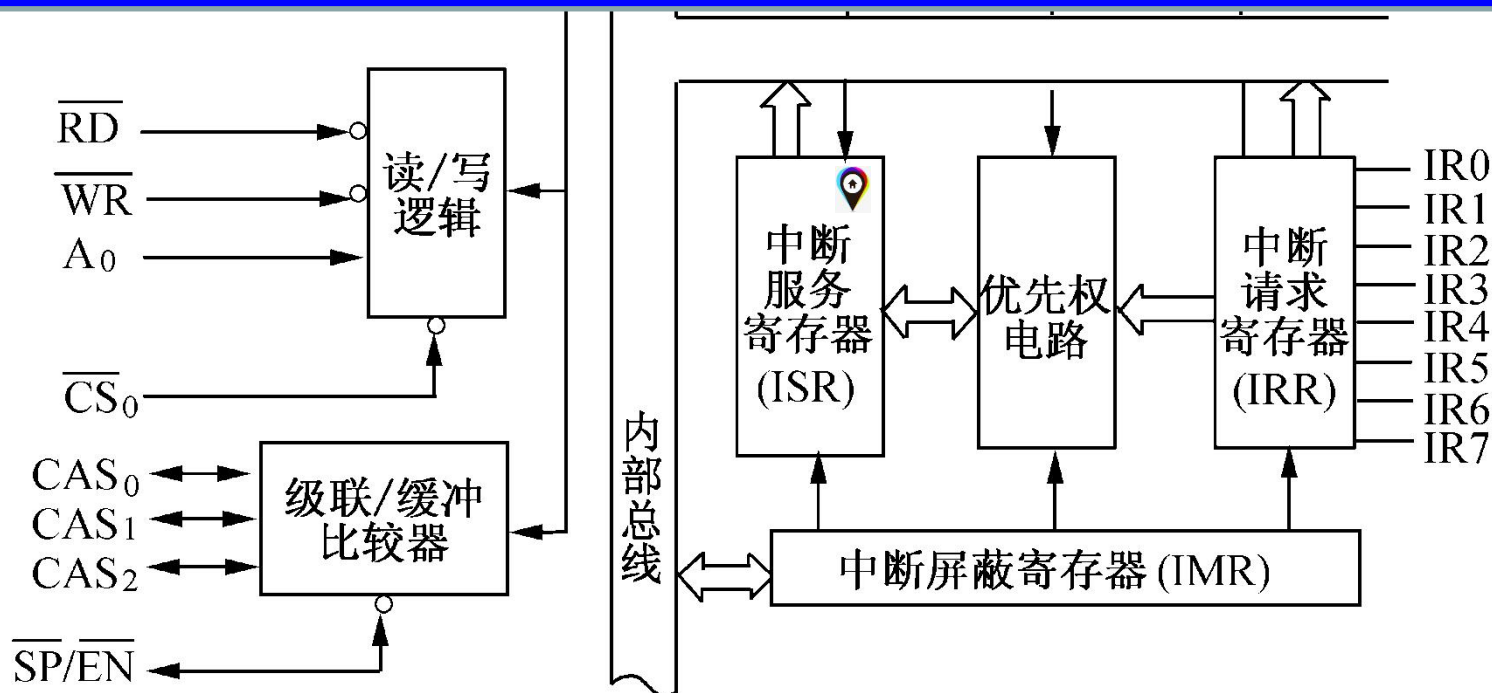




## 7.3.1 8259A的内部结构和引脚

### 中断服务寄存器ISR

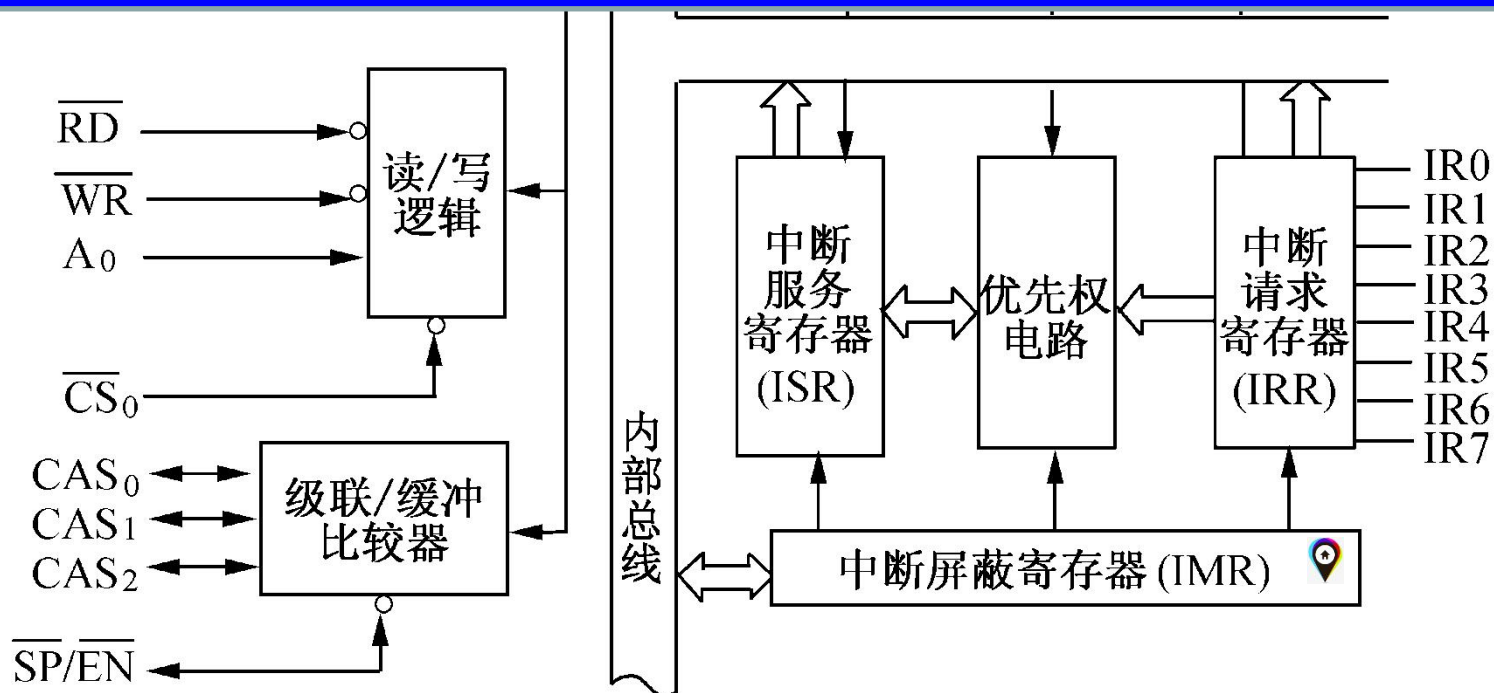
- ✧ 标记正在处理过程中的中断请求
- ✧  $D_i$ 位为1表示 $IR_i$ 中断正在服务中，为0表示没有被服务



## 7.3.1 8259A的内部结构和引脚

### 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



# 1. 中断控制相关寄存器及引脚

## ◇ 中断请求寄存器IRR

- ◆ 保存8条外部中断请求信号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

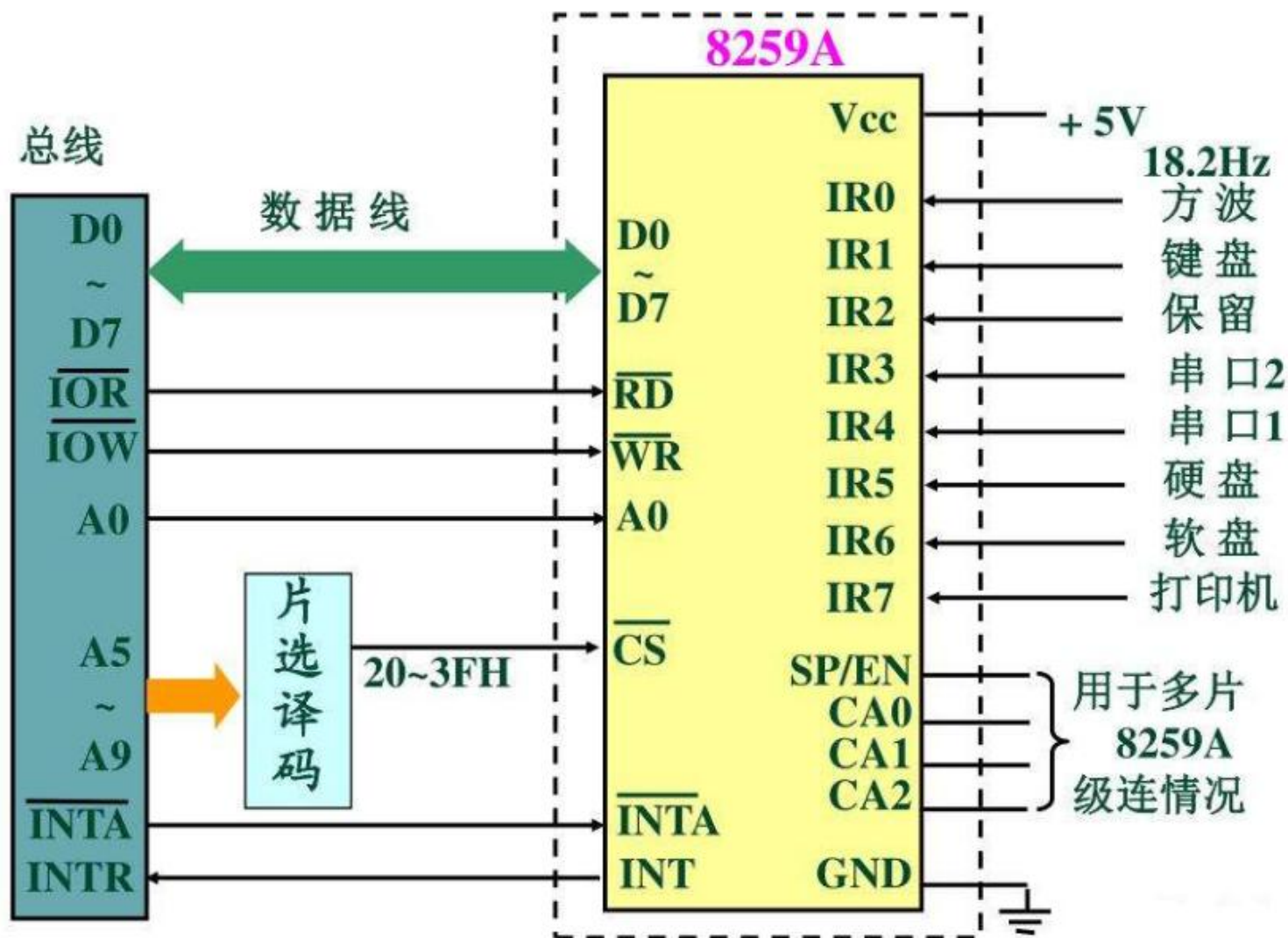
## ◇ 中断服务寄存器ISR

- ◆ 标记正在处理过程中的中断请求
- ◆ 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. 8259A与处理器的接口



## 2. 8259A与处理器的接口

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

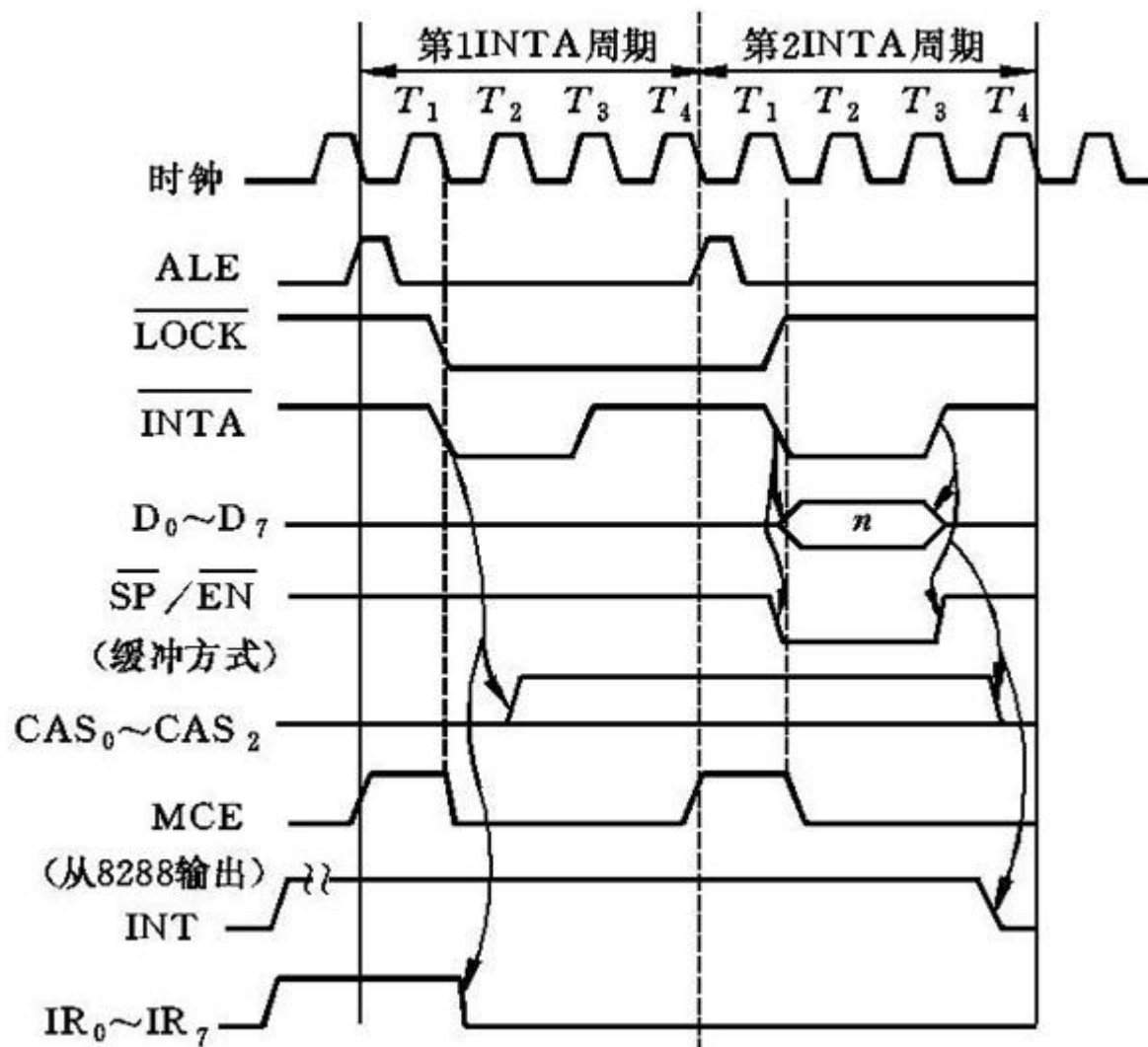
### 3. 中断级连

- ◇ 一个系统中，8259A可以级联，一个主8259A，可联接多达8个从8259A；
- ◇ 级联时，主8259A的三条级联线CAS0~CAS2作为输出线，连至每个从8259A的CAS0~CAS2；
- ◇ 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）。

示例

动画

## 7.3.2 8259A的中断过程



动画

## 7.3.3 8259A的工作方式

1. 中断触发方式
2. 屏蔽中断源方式
3. 设置优先权方式
4. 结束中断处理方式
5. 数据线连接方式

小 结



# 1. 中断请求方式



- (1) 边沿触发方式
- (2) 电平触发方式
- (3) 中断查询方式



# 1. 中断请求方式

## (1) 边沿触发方式



将IR7~IR0出现的由低电平向高电平的跳变作为中断请求信号。

**优点：**申请中断的IRi端可以一直保持高电平而不会被误判为又发生一次中断申请。

**注意：**在CPU响应中断，发回第一个INTA\*应答信号前，已申请过中断的IRi端不要发生又一次的由低电平到高电平的跳变。

**实现方法：**使初始化控制字ICW1的D3位置0。

# 1. 中断请求方式

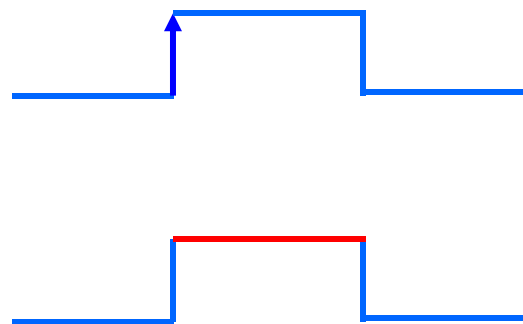
## (2) 电平触发方式

将IR7~IR0上出现的高电平作为中断请求信号。

**优点：**可靠，不会因IRi端引入干扰信号而引起误操作。

**注意：**中断申请信号需保持到第一个INTA信号的前沿，若时间过短会丢失该信号；另外，在CPU响应中断且ISR的相应位置1后，必须撤除中断申请信号，否则会发生第二次中断申请。

**实现方法：**初始化命令字ICW1的D3位置1。



# 1. 中断请求方式

## (3) 中断查询方式

当CPU内部的中断标志位 $IF=0$ 时，CPU可采用查询方式了解 $IR7\sim IR0$ 哪一端有中断申请并为它服务。

- ◇ 实现办法：CPU先执行一条输出指令，对8259A发出查询命令。查询命令是通过对OCW3的D2位置1来实现的；查询命令发出后，执行一条输入指令，可得到查询字。用查询字的I位判断当前有无中断请求，用W2、W1、W0位指出当前发出中断请求级别最高的中断信号。

## 2. 屏蔽中断源的方式

- (1) 普通屏蔽方式
- (2) 特殊屏蔽方式

## (1) 普通屏蔽方式

- ◇ 通过将操作控制字OCW1的相应位置1来对特定的中断源进行屏蔽。
- ◇ 应用
  - ✧ 当CPU执行主程序时，可以将不希望响应的中断源屏蔽；
  - ✧ 当CPU执行某一个中断服务程序时，可以将不希望响应的比此中断优先级高的中断源屏蔽。

IR7				IR0			
0	0	1	1	0	0	0	0

## (2) 特殊屏蔽方式

◇ CPU正在处理某一级中断时，通过设置IMR和ISR实现只对本级中断进行屏蔽，允许级别比它高或比它低的中断源申请中断。

◇ 应用


在中断处理过程中动态改变系统的中断优先级结构。

## (2) 特殊屏蔽方式（续）

- ◇ 实现方法：在某级中断服务程序中首先将操作控制字OCW3的D6、D5位置1，进入特殊屏蔽方式，然后通过设置控制字OCW1使该级的中断申请被屏蔽。
- ◇ 若想退出特殊屏蔽方式，通过将操作控制字OCW3的D6、D5位分别设置为1、0即可。



### 3. 设置优先权方式

- 
- (1) 普通全嵌套方式
  - (2) 特殊全嵌套方式
  - (3) 优先权自动循环方式
  - (4) 优先权特殊循环方式

# (1) 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
- ◇ 有中断请求发生时，8259A对当前中断请求中优先权最高的中断IR<sub>i</sub>予以响应，将其向量号送上数据总线，对应ISR的D<sub>i</sub>位置位，直到中断结束（ISR的D<sub>i</sub>位复位）
- ◇ 在ISR的D<sub>i</sub>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
- ◇ 该方式是8259A被初始化后自动进入的基本工作方式

## (2) 特殊全嵌套方式

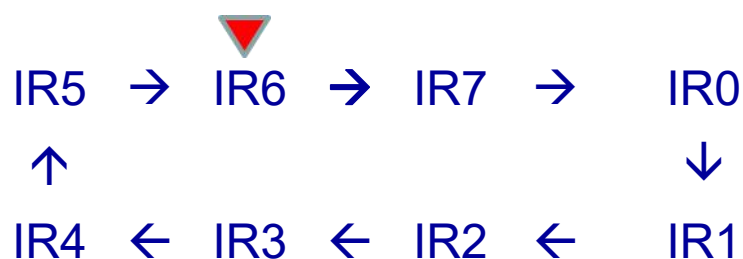
- ◇ 特殊全嵌套方式中，中断源IR0优先级最高，IR7最低。但在CPU处理某一中断请求时，不但响应比该级中断优先级别高的中断申请，而且响应同级的中断申请。
- ◇ 应用：特殊全嵌套方式一般用于级连方式中的主8259A。
- ◇ 操作控制字ICW4中的D4位置1，为特殊全嵌套方式，置0为普通全嵌套方式。

### (3) 优先权自动循环方式

- ◇ 是改变中断请求优先级别的策略之一
- ◇ 基本思想：在优先级自动循环方式下，初始优先级顺序为IR0最高，IR7最低，从高到低的顺序依次为IR0, IR1, IR2, IR3, IR4, IR5, IR6, IR7，当某一个中断源受到服务后，它的优先级别改为最低级，而将最高优先级赋给比它低一级的中断源，其他级别依次类推。



初始优先级顺序



当IR5得到中断服务以后

### (3) 优先权自动循环方式（续）

- ◇ **应用**：当IR0 ~ IR7端引入的中断源有相同的优先级时，使用自动循环方式，能够使得每个中断源有同等的机会得到CPU的服务。
- ◇ **实现方法**：通过将操作控制字OCW2的D7、D6位置为1、0实现。

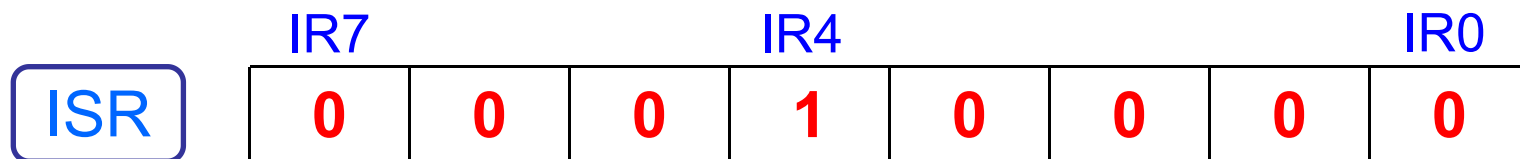
## (4) 优先权特殊循环方式

- ◇ **特点**：该循环方式和优先级自动循环方式基本相同，不同点仅在于可以根据用户要求将最低优先级赋于某一中断源。
- ◇ **实现方法**：通过将操作控制字OCW2的D7、D6位置1，可设置为优先级特殊循环方式，同时用OCW2中的D2、D1、D0位指出哪个中断源的级别最低。

## 4.中断结束处理方式

什么是8259A的中断结束处理？

- ◇ 使中断服务寄存器**ISR**的相应位清0的操作。
- ◇ 中断服务寄存器（**ISR**）的状态反应出当前**CPU**正在为哪个中断源服务和还未完成哪个中断源的服务；
- ◇ 中断结束处理并不是结束中断服务程序，只是使该中断服务寄存器的相应位清0。



## 4. 结束中断处理方式



- (1) 自动中断结束方式
- (2) 普通中断结束方式
- (3) 特殊中断结束方式





## (1) 自动中断结束方式

- ◇ 自动中断结束是指中断服务寄存器的相应位清零是由硬件自动完成的。
- ◇ 在此方式下，当某一级中断被CPU响应后：
  - ①CPU送回的第一个INTA\*中断应答使中断服务寄存器ISR的相应位置1；
  - ②第二个INTA\*负脉冲结束时自动将ISR的相应位清0。

## (1) 自动中断结束方式



- ◇ 应用：中断自动结束方式适用于只有一块8259A，并且各级中断不会发生嵌套的情况。
- ◇ 实现方法：通过将初始化控制字ICW4的D1位设置为1实现。



## (2) 普通中断结束方式

- ◇ 程序中通过向8259A发出普通中断结束命令（EOI），使得8259A将所有正在服务的中断中优先级最高的ISR位复位。
- ◇ 配合普通全嵌套优先级方式使用。

### (3) 特殊中断结束方式

- ◇ 通过向8259A发送特殊中断结束命令来清除指定的ISR位
- ◇ 配合循环优先级方式使用

## 5. 数据线连接方式



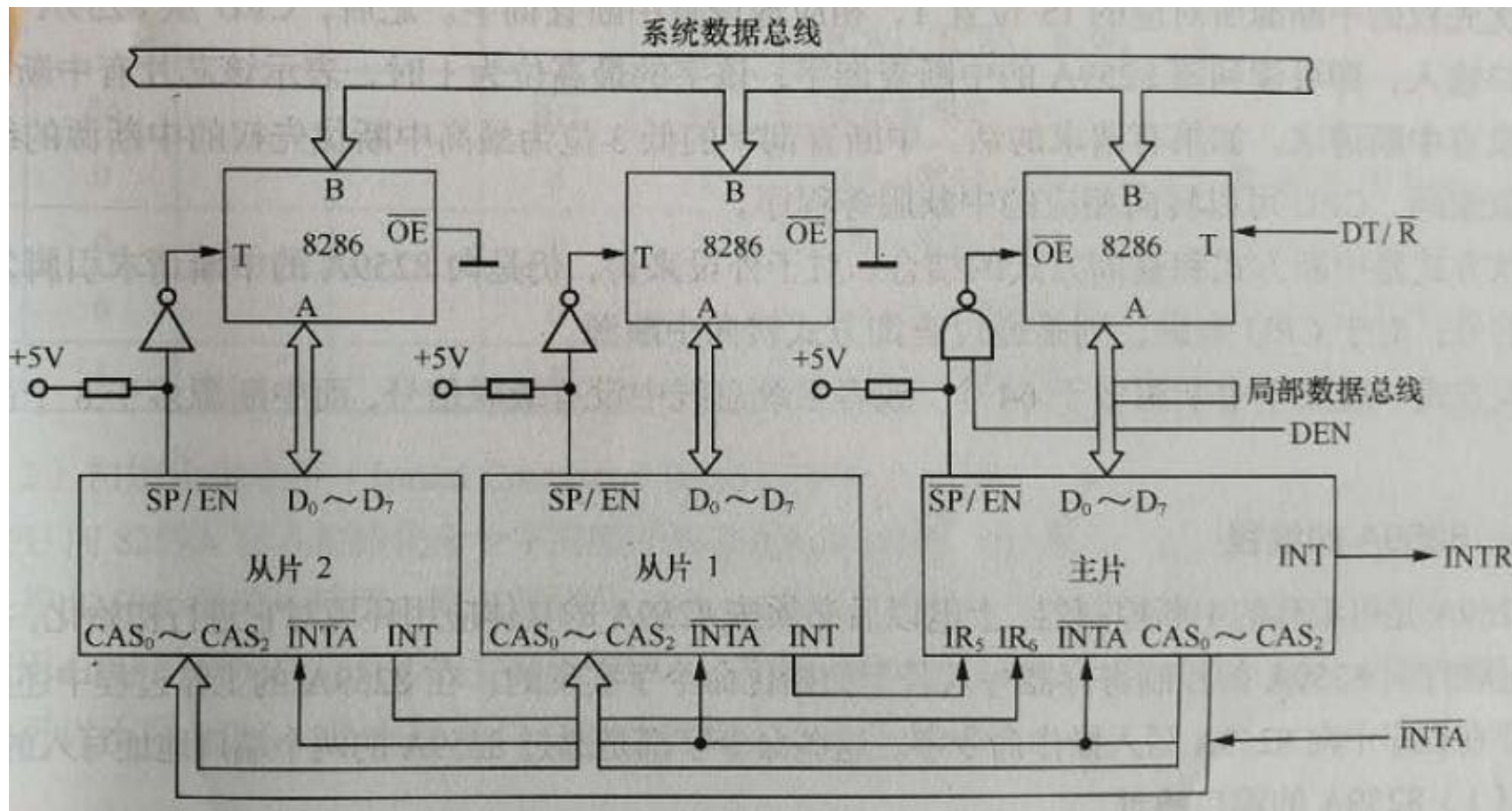
### (1) 缓冲方式

特点：8259A通过总线驱动器和数据总线相连。

应用：多片8259A级连的大系统中。

实现方法：将8259A的初始化控制字ICW4的D3位置1，设置为缓冲方式，并把8259A的SP\*/EN\*端输出一个低电平信号作为总线驱动器的启动信号。

# 8259A的级联缓冲方式



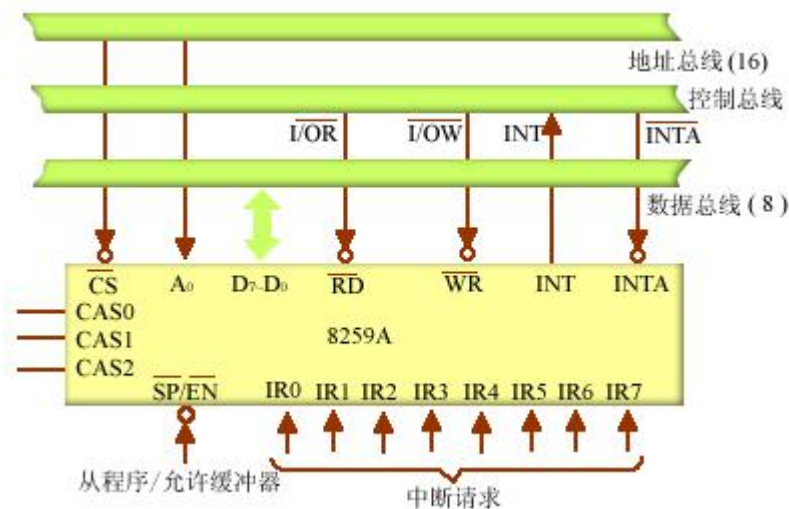
## 5. 数据线连接方式

### (2) 非缓冲方式

特点：8259A直接和数据总线相连。

应用：这种方式用于单片8259A或片数不多的8259A组成的系统中。

实现方法：将初始化控制字ICW4的D3位置0，设置为非缓冲方式。在非缓冲方式时，对于单片8259A， $SP^*/EN^*$ 端为输入，接高电平，对于多片8259A的级连系统，主8259A的 $SP^*/EN^*$ 端接高电平，从8259A的 $SP^*/EN^*$ 端接低电平。

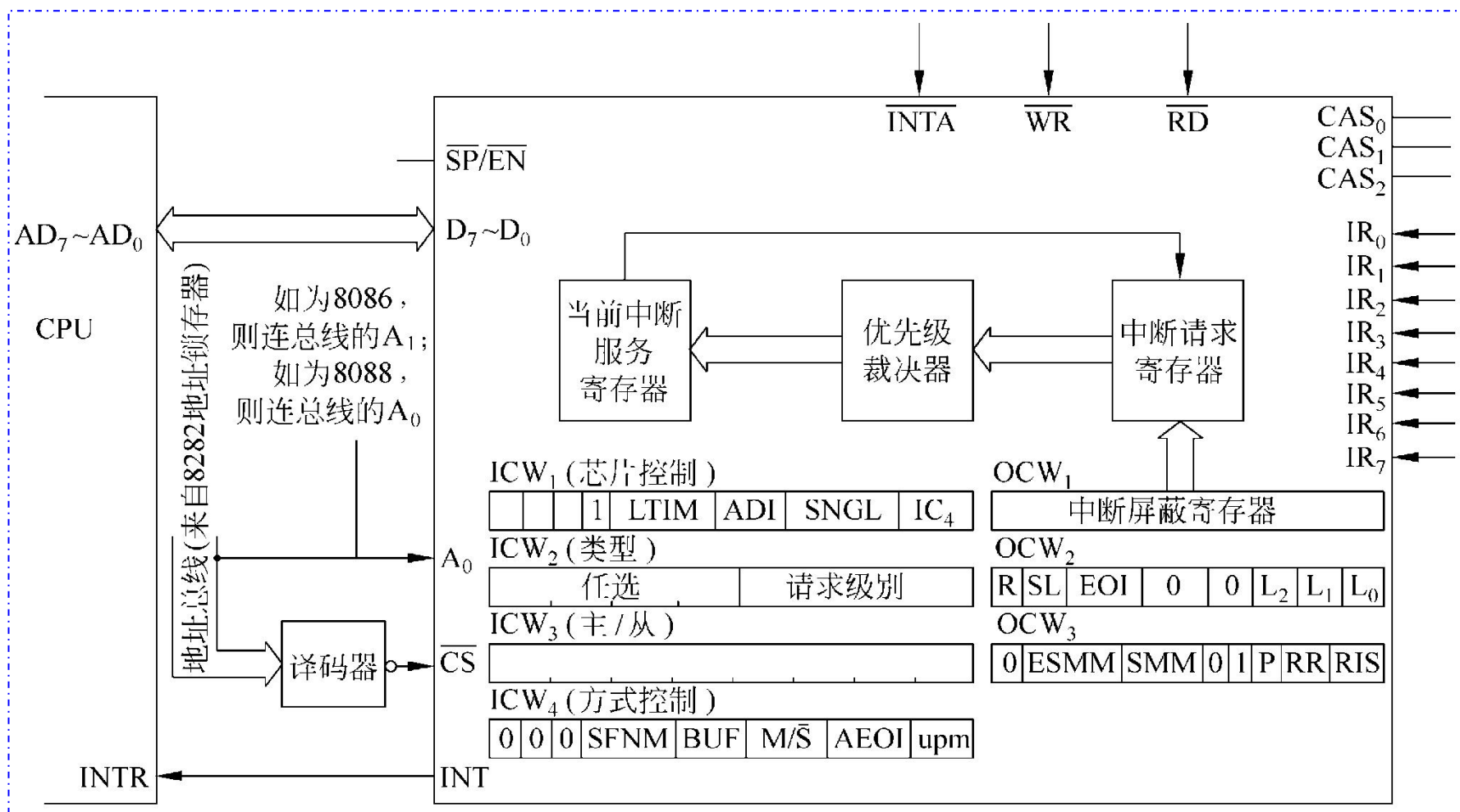


## 7.3.4 8259A的编程

- ◇ 8259A的工作状态和操作方式由CPU的命令设定。
- ◇ 命令有两种
  - ◆ 初始化命令：初始化命令字 ICW1~ICW4
  - ◆ 操作控制命令：操作命令字 OCW1~OCW3
- ◇ 每片8259A有2个片内地址A0=0和A0=1，所有的命令都是通过这两个端口来设置的。




# 8259A的编程结构



# 8259的初始化命令字

- ◇ 初始化命令字ICW最多有4个，8259A在开始工作前必须写入。
- ◇ ICW1和ICW2必须设置，ICW3和ICW4由工作方式决定
- ◇ ICW1写入8259偶地址中 ( $A_0=0$ ，在AT机中为20H/A0H)
- ◇ ICW2~ICW4写入8259奇地址中 ( $A_0=1$ ，在AT机中为21H/A1H)。
- ◇ ICW<sub>1</sub>~ICW<sub>4</sub>必须在初始化程序中按顺序设定，且在整个工作过程中保持不变。

# ICW1(Initialization Command Word)




D7	D6	D5	D4	D3	D2	D1	D0
×	×	×	1	LTIM	×	SNGL	IC4

例1



# ICW1




D7	D6	D5	D4	D3	D2	D1	D0
×	×	×	1	LTIM	×	SNGL	IC4

×——表示可以任意  
为1为0都可以（建议为0）

例1



# ICW1



D7	D6	D5	D4	D3	D2	D1	D0
×	×	×	1	LTIM	×	SNGL	IC4




1——只能为1，作为标志

例1



# ICW1



D7	D6	D5	D4	D3	D2	D1	D0
×	×	×	1	LTIM	×	SNGL	IC4

中断触发方式：


LTIM = 1，电平触发方式

LTIM = 0，边沿触发方式

例1



# ICW1



D7	D6	D5	D4	D3	D2	D1	D0
×	×	×	1	LTIM	×	SNGL	IC4

规定单片或级连方式：


SNGL = 1，单片方式

SNGL = 0，级连方式

例1



# 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

例1





## ICW2 设置中断类型码

A <sub>0</sub>	D <sub>7</sub>	D <sub>6</sub>	D <sub>5</sub>	D <sub>4</sub>	D <sub>3</sub>	D <sub>2</sub>	D <sub>1</sub>	D <sub>0</sub>
1	?	?	?	?	?	X	X	X

- ◇ 在写ICW<sub>1</sub>之后，对A<sub>0</sub>=1的端口第一次写入的数据是ICW<sub>2</sub>。
- ◇ 在8086/8088系统中，设置D<sub>7</sub>~D<sub>3</sub>，D<sub>2</sub>~D<sub>0</sub>无效（由8259A根据IR<sub>0</sub>~IR<sub>7</sub>自动填充为000~111）。
- ◇ 应用实例：在PC/XT中ICW<sub>2</sub>为00001000B，则

中断号：类型号

IR<sub>0</sub>: 08H 时钟中断

IR<sub>1</sub>: 09H 键盘中断

IR<sub>2</sub>: 0AH 保留

IR<sub>3</sub>: 0BH COM2

中断号：类型号

IR<sub>4</sub>: 0CH COM1

IR<sub>5</sub>: 0DH 硬盘

IR<sub>6</sub>: 0EH 软盘

IR<sub>7</sub>: 0FH LPT1

例1

## ICW3 设置级联

◇ 系统中有级联时 ( $ICW_1.SNGL=0$ )，在 $ICW_2$ 之后写 $ICW_3$ 。


◇ 对于主片：置1的位表示对应的引脚IR有从片级联。

$A_0$	$D_7$	$D_6$	$D_5$	$D_4$	$D_3$	$D_2$	$D_1$	$D_0$
1	0	0	0	0	0	1	0	0

◇ 对于从片：用 $D_2 \sim D_0$ 表示和主片的对应引脚级联。

$A_0$	$D_7$	$D_6$	$D_5$	$D_4$	$D_3$	$D_2$	$D_1$	$D_0$
1	X	X	X	X	X	0	1	0

# ICW4 模式设置



$A_0$	$D_7$	$D_6$	$D_5$	$D_4$	$D_3$	$D_2$	$D_1$	$D_0$
1	0	0	0	SFNM	BUF	M/S	AEOI	$\mu$ PM


- ◇  $ICW_1, IC_4=1$ 时, 有 $ICW_4$ 。
- ◇  $D_4$ : SFNM 中断的嵌套方式  
0: 一般嵌套      1: 特殊的全嵌套  
在一般嵌套方式下, 优先级 $IR_0 \sim IR_7$ 从高到低
- ◇  $D_1$ : AEOI 自动结束中断方式  
0: 不自动清除ISR  
1: CPU响应中断后, 自动清除ISR
- ◇  $D_0$ :  $\mu$ PM 微处理器类型  
0: 8080/8085/Z80      1: 8086/8088



例1



## ICW4 模式设置



A <sub>0</sub>	D <sub>7</sub>	D <sub>6</sub>	D <sub>5</sub>	D <sub>4</sub>	D <sub>3</sub>	D <sub>2</sub>	D <sub>1</sub>	D <sub>0</sub>
1	0	0	0	SFNM	BUF	M/S	AEOI	μ PM

### ◇ D3: BUF 缓冲

1: 8259通过数据缓冲器和总线相连, SP\*/EN\*引脚输出, 缓冲器选通端。

0: 无缓冲, SP\*/EN\*引脚输入, 用作主片、从片选择端。

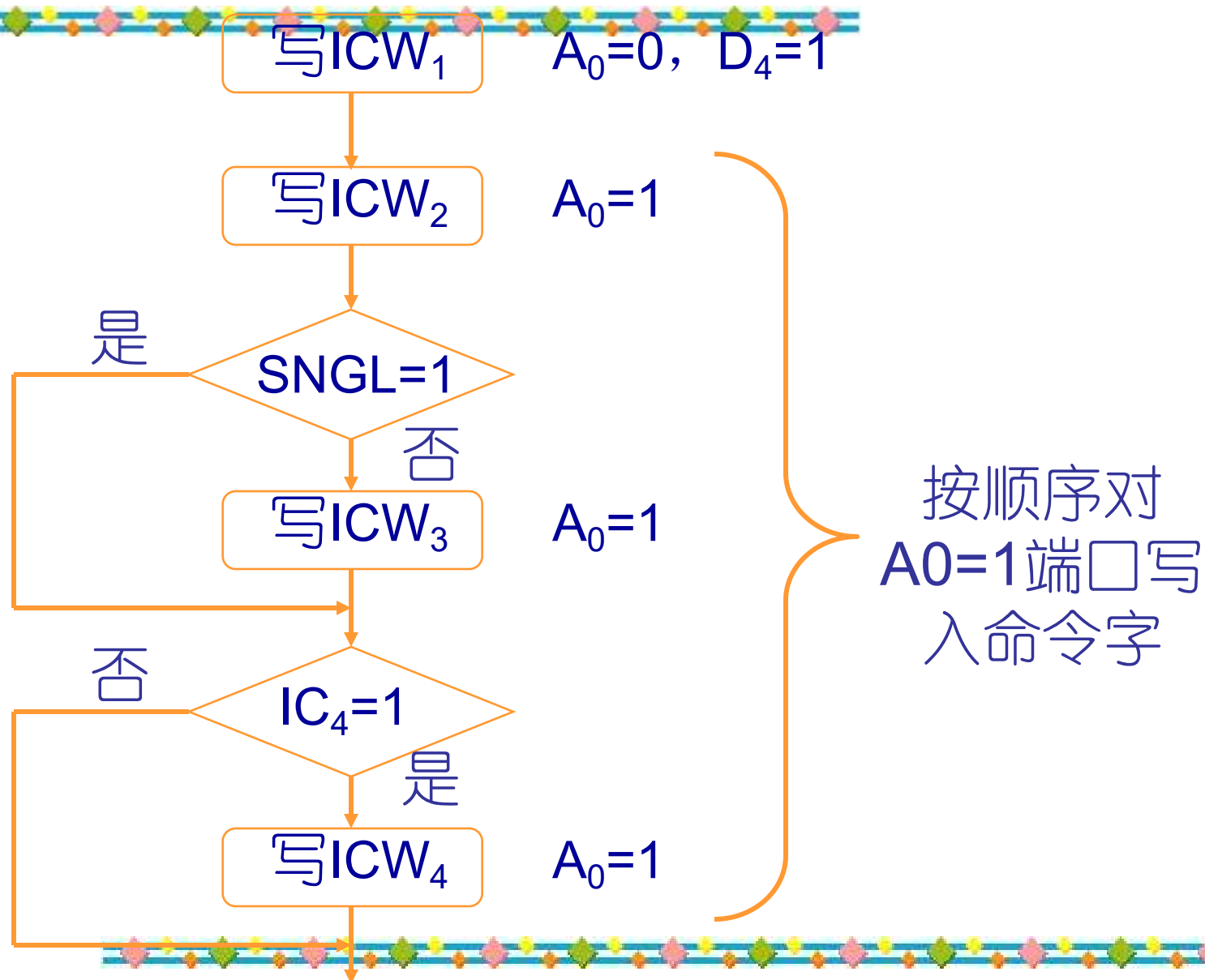
### ◇ D2: M/S\* 主片/从片选择 (BUF=1时, 有效)

0: 从片      1: 主片

◇ 初始化编程一般在系统启动时进行, 初始化以后系统才可以接收中断请求信号。



# 8259A芯片的初始化流程



## 4. 接口电路中多个端口寄存器的区分方法



- (1) 利用地址信号区别不同I/O地址的寄存器
- (2) 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
- (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

# 8259A初始化实例

例1 PC/XT机中8259A的端口地址是20H、21H，请分析如下初始化代码序列的功能。

```
MOV    AL, 13H
OUT    20H, AL
```

ICW<sub>1</sub>: 单片、上升沿触发、  
使用ICW<sub>4</sub>

```
MOV    AL, 08H
OUT    21H, AL
```

ICW<sub>2</sub>: 中断类型码是  
08H~0FH

```
MOV    AL, 0DH
OUT    21H, AL
```

ICW<sub>4</sub>: 非AEIOI方式

例2：PC/AT机中8259A主片的端口地址是20H、21H，  
从片的端口地址是A0H、A1H。初始化序列如下：

• 初始化主片

• 初始化从片

ICW<sub>1</sub>

MOV AL, 11H

OUT 20H, AL

ICW<sub>2</sub>

MOV AL, 08H

OUT 21H, AL

ICW<sub>3</sub>

MOV AL, 04H

OUT 21H, AL

ICW<sub>4</sub>

MOV AL, 01H

OUT 21H, AL

MOV AL, 11H

OUT 0A0H, AL

MOV AL, 70H

OUT 0A1H, AL

MOV AL, 02H

OUT 0A1H, AL

MOV AL, 01H

OUT 0A1H, AL

ICW<sub>1</sub>

ICW<sub>2</sub>

ICW<sub>3</sub>

ICW<sub>4</sub>





上述初始化程序设置的结果为：

◇ 主片定义为：

上升沿触发、在 $IR_2$ 级联从片、有 $ICW_4$ 、非  
**AEOI**方式、中断类型码为08H~0FH、一般的中  
断嵌套方式

◇ 从片定义为：

上升沿触发、级联到主片的 $IR_2$ 、有 $ICW_4$ 、非  
**AEOI**方式、中断类型码为70H~77H、一般的中  
断嵌套方式

# 8259的操作命令字OCW(Operation Command Word)


- ◇ 系统初始化完成以后，可以在应用程序中进行操作编程。
- ◇  $OCW_1$  设置和清除中断屏蔽寄存器

$A_0$	$D_7$	$D_6$	$D_5$	$D_4$	$D_3$	$D_2$	$D_1$	$D_0$
1	$M_7$	$M_6$	$M_5$	$M_4$	$M_3$	$M_2$	$M_1$	$M_0$

- ◇  $M_x=1$ 表示屏蔽中断源 $IR_x$

```
IN    AL, 21H
OR    AL, M 或 AND AL, M
OUT   21H, AL
```

## OCW2 设置优先级循环方式和中断结束方式



A <sub>0</sub>	D <sub>7</sub>	D <sub>6</sub>	D <sub>5</sub>	D <sub>4</sub>	D <sub>3</sub>	D <sub>2</sub>	D <sub>1</sub>	D <sub>0</sub>
0	R	SL	EOI	0	0	L <sub>2</sub>	L <sub>1</sub>	L <sub>0</sub>

对A<sub>0</sub>=0端口写入D<sub>4</sub>D<sub>3</sub>=00的数据，表示是OCW<sub>2</sub>

- ◇ R：表示优先级是否循环；
- ◇ SL：表示L<sub>2</sub>~L<sub>0</sub>是否有效；
- ◇ EOI：中断结束命令位。

在PC机中常用的是EOI命令：

```
MOV AL, 20H  
OUT 20H, AL
```

R SL EOI 0 0 L2 L1 L0	功能
0 0 0 0 0 0 0 0	自动EOI、优先级固定命令
0 0 1 0 0 0 0 0	优先级固定方式时的中断结束命令
0 1 1 0 0 L2 L1 L0	特殊EOI命令
1 0 0 0 0 0 0 0	优先级自动轮转命令
1 0 1 0 0 0 0 0	非自动EOI[不指定]、优先级自动轮转命令
1 1 1 0 0 L2 L1 L0	非自动EOI[指定]、优先级指定轮转命令
1 1 0 0 0 L2 L1 L0	自动EOI、优先级指定轮转命令

# OCW3



A <sub>0</sub>	D <sub>7</sub>	D <sub>6</sub>	D <sub>5</sub>	D <sub>4</sub>	D <sub>3</sub>	D <sub>2</sub>	D <sub>1</sub>	D <sub>0</sub>
0	0	ESMM	SMM	0	1	P	RR	RIS

◇ 对A<sub>0</sub>=0端口写入D4D3=01的数据，表示是OCW<sub>3</sub>

OCW<sub>3</sub> 的功能有三个方面

- ◇ 设置和撤销特殊屏蔽方式
- ◇ 设置对8259A内部寄存器的读出
- ◇ 设置中断查询方式



# OCW3 设置和撤销特殊屏蔽方式



$A_0$	$D_7$	$D_6$	$D_5$	$D_4$	$D_3$	$D_2$	$D_1$	$D_0$
0	0	ESMM	SMM	0	1	P	RR	RIS

ESMM	SMM	功能
0	X	无效
1	0	取消特殊屏蔽方式
1	1	设置特殊屏蔽方式

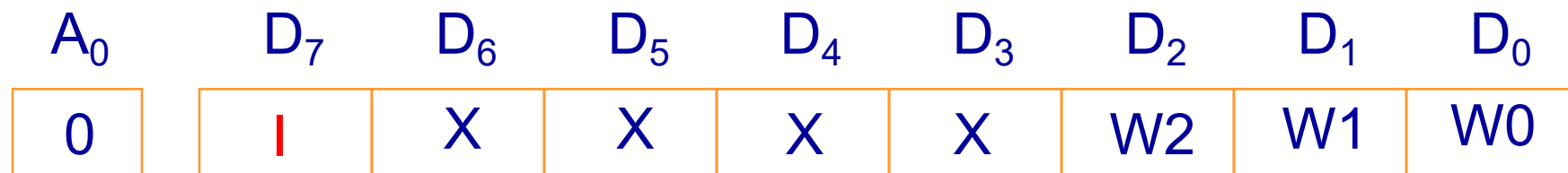


## OCW3 对8259A内部寄存器的读出

P RR RIS	功能	应用
0 1 0	读取IRR内容	MOV AL, 0AH OUT 20H, AL IN AL, 20H
0 1 1	读取ISR内容	MOV AL, 0BH OUT 20H, AL IN AL, 20H
0 0 *	不读取任何寄存器内容	
1 * *	读取查询字信息 状态字格式见下页	MOV AL, 0CH OUT 20H, AL IN AL, 20H

# OCW3 中断查询

## 中断状态字格式



0:表示无中断请求  
1:表示有中断请求

中断源编码



# OCW3

在PC机中常用的是：

读出 IRR：先向20H端口写0AH, 再读20H端口

读出 ISR：先向20H端口写0BH, 再读20H端口

读出 最高级别的中断请求 IR：

先向20H端口写0CH, 再读20H端口

A <sub>0</sub>	D <sub>7</sub>	D <sub>6</sub>	D <sub>5</sub>	D <sub>4</sub>	D <sub>3</sub>	D <sub>2</sub>	D <sub>1</sub>	D <sub>0</sub>
0	0	ESMM	SMM	0	1	P	RR	RIS

例 设8259的□地址为280H和281H

PORT EQU 280H

P EQU 0CH

RDIRR EQU 0AH

RDISR EQU 0BH

MOV DX,PORT

MOV AL,P

OUT DX,AL

IN AL,DX

MOV AL,RDIRR

OUT DX,AL

IN AL,DX

MOV AL,RDISR

OUT DX,AL

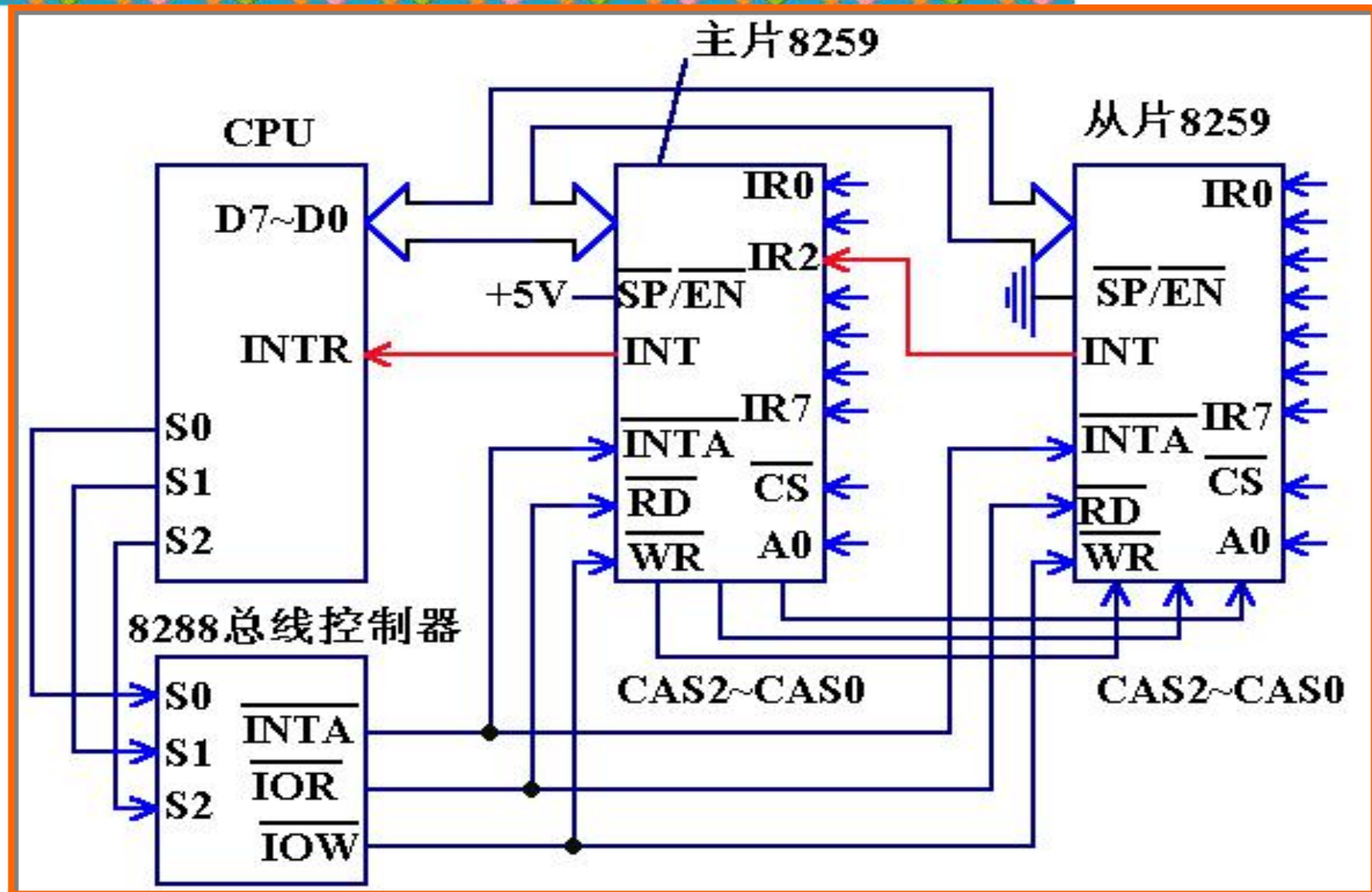
IN AL,DX

# 8259A应用举例

PC/AT中，8259的使用情况为：

1. 2片8259级联，提供15级向量中断。从片的INT接主片的IR2(图)。
2. 端口地址：主片20H、21H，从片A0H、A1H。
3. 主片和从片均采用边沿触发。
4. 采用全嵌套优先级排列方式
5. 采用非缓冲方式，主片 $\overline{SP}/\overline{EN}$ 接+5V,从片 $\overline{SP}/\overline{EN}$ 接地。
6. 主片的类型码为08H~0FH，从片的类型码为70H~77H。

# PC/AT中8259A连线图



# PC/AT机主、从8259的初始化程序



ICW1A EQU 20H ; 主片端口地址

ICW2A EQU ICW1A+1

ICW3A EQU ICW2A

ICW4A EQU ICW2A

ICW1B EQU 0A0H ; 从片端口地址

ICW2B EQU ICW1B+1

ICW3B EQU ICW2B

ICW4B EQU ICW2B



## ; ----- 主片8259A Initialization-----



```
MOV AL, 11H      ; ICW1, 边沿触发, 多片, 需ICW4
OUT ICW1A, AL
NOP              ; I/O端口延时
MOV AL, 08H      ; ICW2, 中断类型码
OUT ICW2A, AL
NOP
MOV AL, 04H      ; ICW3, IR2接从片
OUT ICW3A, AL
NOP
MOV AL, 01H      ; ICW4, 非缓冲, 全嵌套, 非自动结束
OUT ICW4A, AL
NOP
```

## ; ----- 从片8259A Initialization-----

MOV AL, 11H ; ICW1, 边沿触发, 多片, 需ICW4

OUT ICW1B, AL

NOP

MOV AL, 70H ; ICW2, 中断类型码

OUT ICW2B, AL

NOP

MOV AL, 02H ; ICW3, INT接主片的IR2

OUT ICW3B, AL

NOP

MOV AL, 01H ; ICW4, 非缓冲, 全嵌套, 非自动结束

OUT ICW4B, AL

NOP

## 第三节 中断服务程序的编程

### 1. 中断服务程序的编程原则

- (1) 中断是异步发生的，进入响应时并不考虑当前运行状态。因此中断服务程序必须保护现场。
- (2) 在进入具体中断处理之前要先初始化中断向量，使其指向相应的中断服务程序，但在此之前要先关中断，以防接管中断过程中发生中断。
- (3) 在中断服务程序入口处要立即开中断，以允许较高优先级的中断产生。
- (4) 中断服务程序的服务时间要尽量压缩，以免干扰同级或低级中断设备的工作。



# 1. 中断服务程序的编程原则



(5) 由于PC机中的8259采用普通中断结束方式工作，因此在中断服务程序中IRET指令之前应向8259发结束中断命令EOI，具体代码如下。

```
MOV AL, 20H
```



```
OUT 20H, AL
```

```
IRET
```

(6) 当需覆盖系统原有的中断向量时，应保存好原中断向量的内容，并在应用程序终止前恢复原有的中断向量。



# 1. 中断服务程序的编程原则

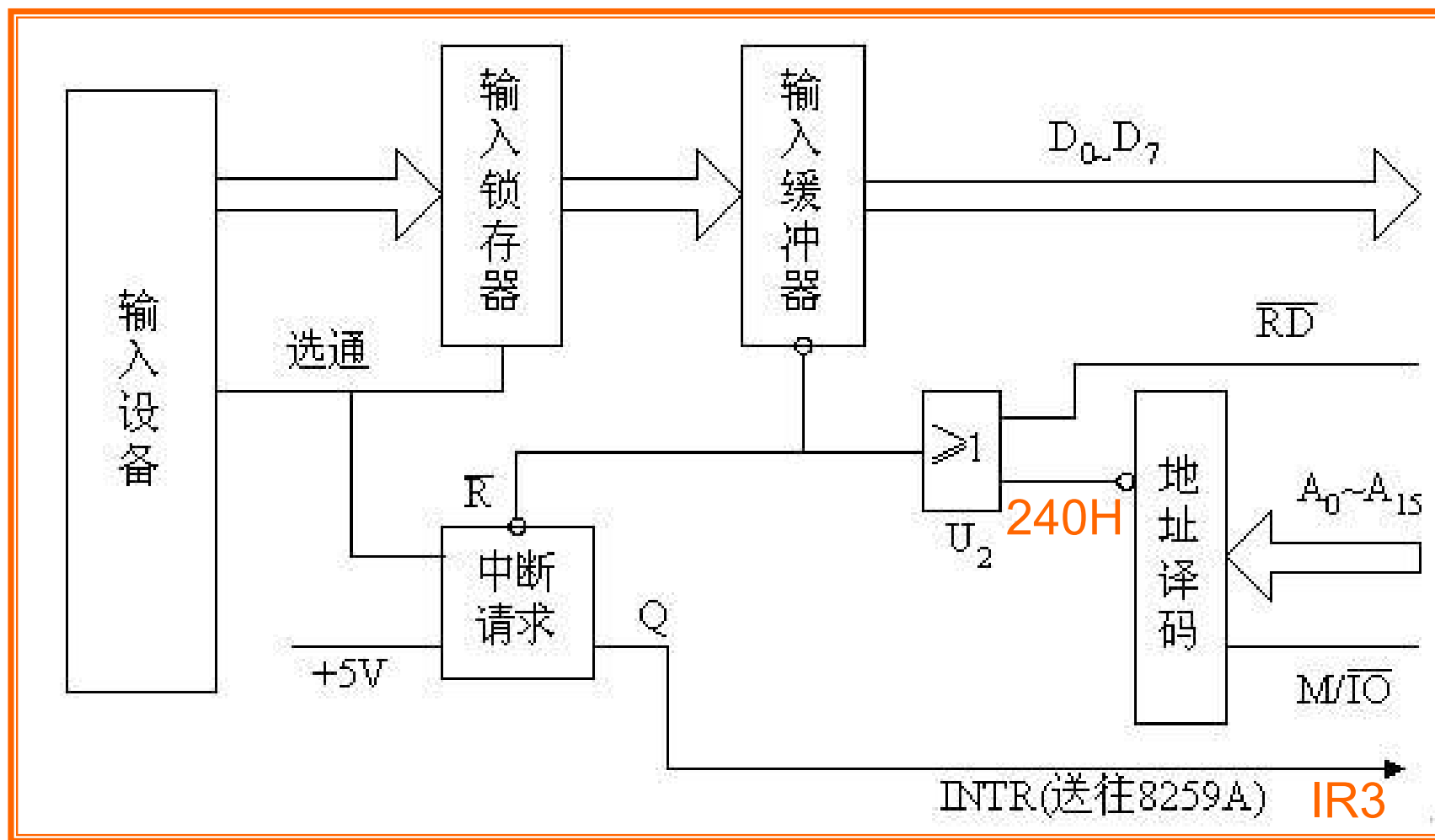
- 
- (7) 中断服务程序应尽量避免使用**DOS**系统功能调用(**INT 21H**)，因为**DOS**不允许重入。
  - (8) 若中断服务程序只为某个应用程序服务，则中断服务程序可以和主程序组装成一个程序一起装入内存，随主程序结束而一起退出内存。
  - (9) 若中断服务程序为多个应用程序服务，则中断服务程序可以与一个初始化程序组装成一个程序一起装入内存，通过初始化程序的执行而将中断服务程序驻留内存。
- 

## 例题：

以下页图中的输入设备为例，给出一个完整的中断方式的输入程序。

设该输入设备的数据端口地址为**240H**，使用**8259A**的**IR3**引脚申请中断，中断类型**0BH**。**8259A**端口地址为**20H**，**21H**。输入以“回车”字符表示结束。

# 中断方式输入设备接口



# 源程序

wj0704.asm

```
.model small
```

```
.stack
```

```
.data
```

； 定义接收缓冲区，假设一次输入不超过100字节

```
IN_BUFFER DB 100 DUP ( ? )
```

； 定义接收缓冲区指针

```
IN_POINTER DW ?
```

； 定义完成标志，=1表示输入已完成

```
DONE DB 0
```

```
；
```



## 源程序（续1）

.code

BEGIN: CLI

MOV AX, SEG IN\_INTR ;IN\_INTR是中断服务程序名

MOV DS, AX

LEA BX, IN\_INTR

MOV AX, 250BH ;AH 中为功能号, AL中为中断类型

INT 21H ;装载0BH中断向量

MOV AX, @data

MOV DS, AX ;装载数据段段基址

LEA IN\_POINTER, IN\_BUFFER ;设置指针初值

MOV DONE, 0 ;设置完成标志为“未完成”

## 源程序（续2）

IN AL, 21H

AND AL, 11110111B

OUT 21H, AL

;清除IR3的屏蔽位

STI

;开放中断

W: CMP DONE, 0

JZ W

;等待完成

:

;结束处理

MOV AX, 4C00H

INT 21H

## 源程序（续3）

;输入中断服务程序

IN\_INTR PROC FAR

PUSH DS ;保护现场

PUSH AX

PUSH BX

PUSH DX

STI ;开放中断，允许响应更高级中断

MOV AX, @data

MOV DS, AX ;在中断服务程序中重新装载DS寄存器


MOV BX, IN\_POINTER ;装载缓冲区指针

MOV DX, 240H


IN AL, DX ;从输入设备读取一个数据,同时清除中断请求



## 源程序 (续4)



```
MOV    [BX],AL           ;数据存入缓冲区
INC     BX
MOV     IN_POINTER,BX    ;修改指针, 存入内存单元
CMP     AL,0DH           ;判输入是否结束
JNE     EXIT
IN      AL,21H
OR      AL,00001000B     ;输入结束, 置IR3屏蔽位
OUT     21H,AL
MOV     DONE,1           ;置完成标志
EXIT:
CLI     ;关闭中断, 准备中断返回
MOV     AL,20H
OUT     20H,AL           ;向8259发中断结束命令
```



## 源程序（续5）



POP DX                   ;恢复现场

POP BX

POP AX

POP DS

IRET                   ;中断返回

IN\_INTR   ENDP

END   BEGIN



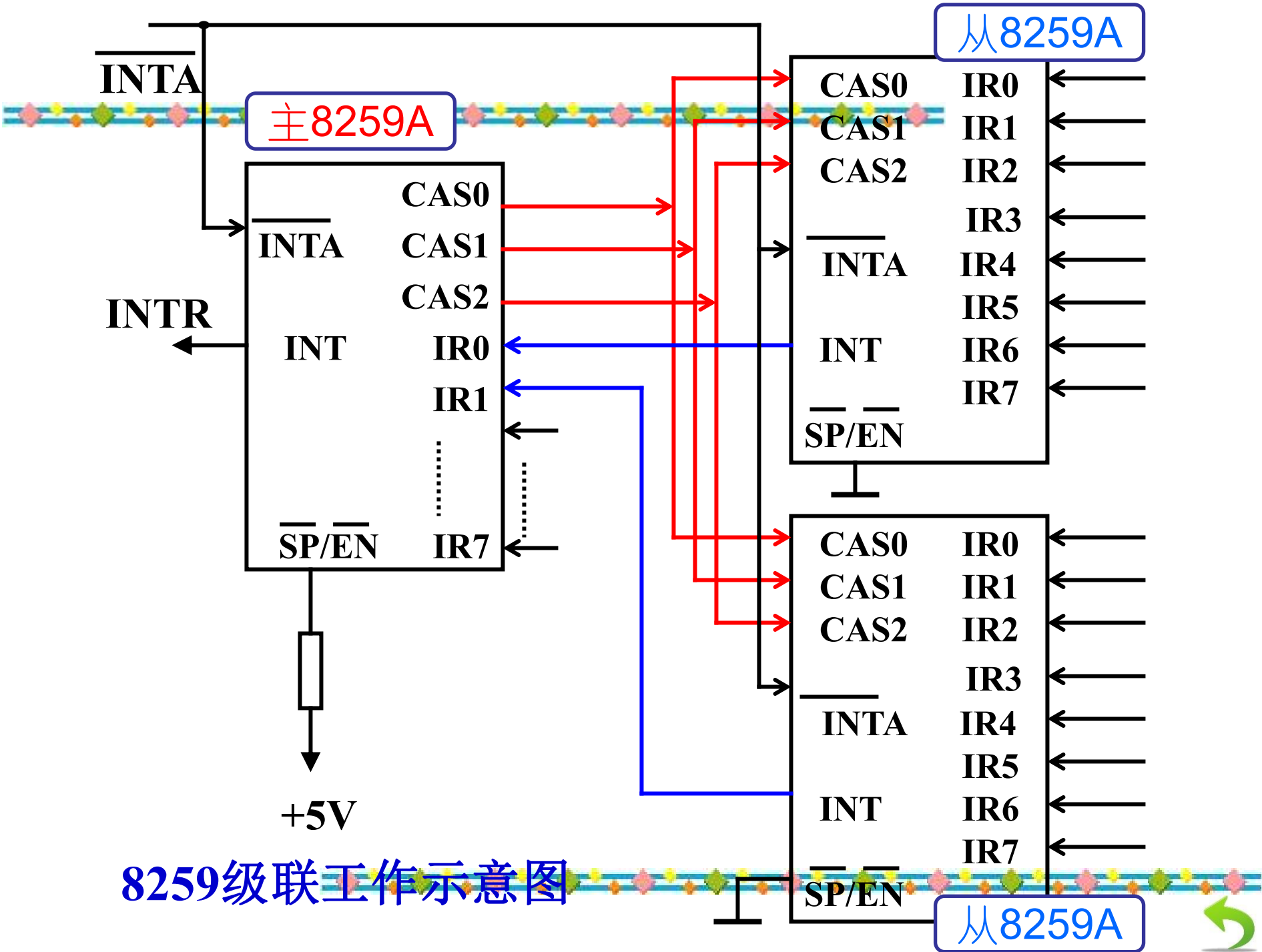
# 习题

1. 假设8259的端口地址为50H、51H，试编写一段程序，将8259中的IRR、ISR、IMR的内容读出送至存储器中REG\_ARR开始的内存单元中
2. 某微机系统采用三片8259级连使用，一片为主，两片为从，从片分别接入主片的IR2和IR4，试画出该系统的硬件连接图。
3. 某微机系统只有一片8259，其端口地址为02C0H、02C1H，试编写初始化程序，要求1) 中断请求输入采用电平触发，2) IR0的中断类型码是16，3) 采用缓冲器方式，4) 采用普通的EOI命令
4. 什么是中断向量和中断向量表？中断类型码和中断向量表的关系是什么？
5. 什么是中断？



*The end*  
*Thank you!*





现假设有三块芯片构成了22级主从式中中断响应系统，下面的图表示了此类系统的工作原理：

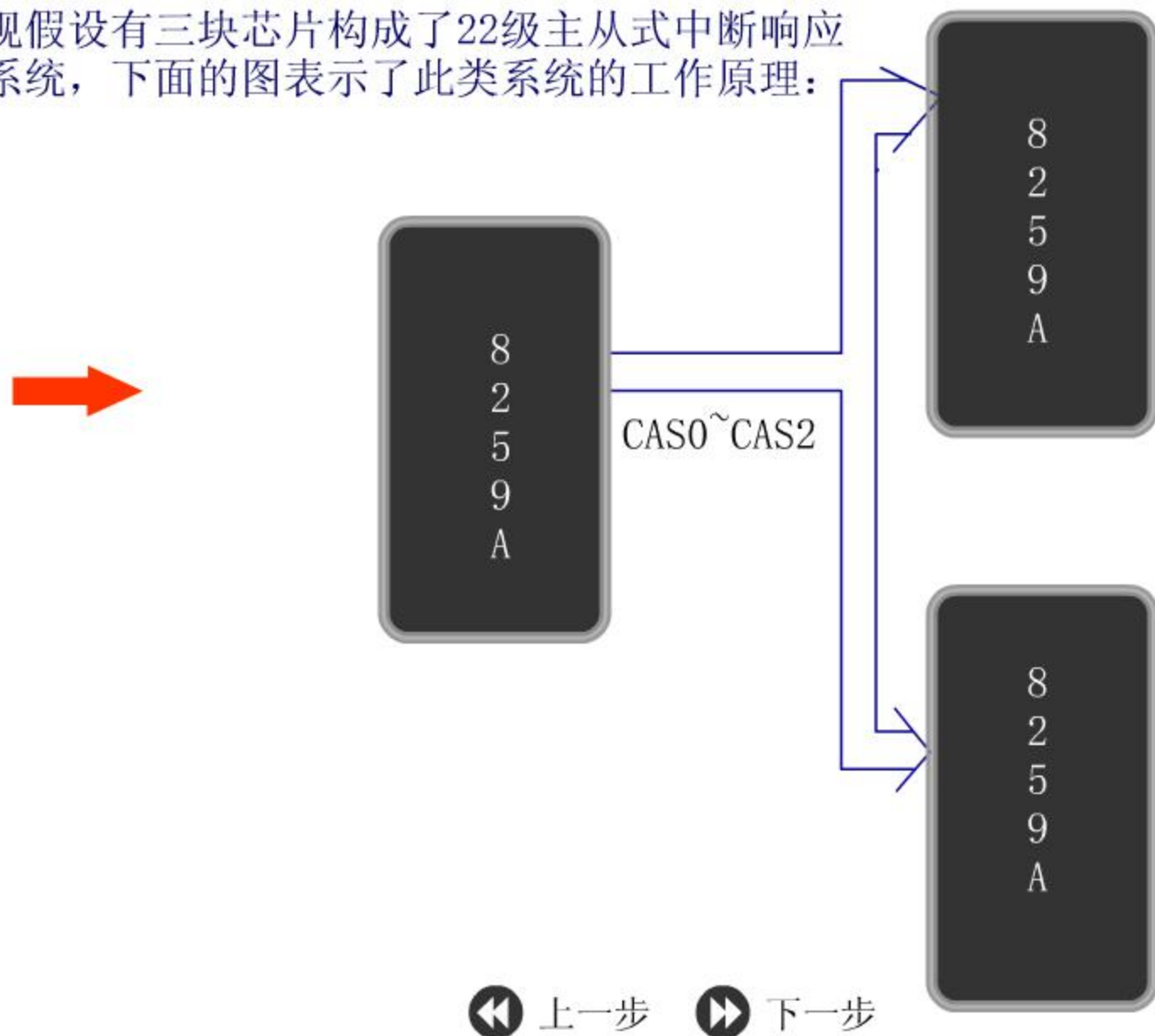
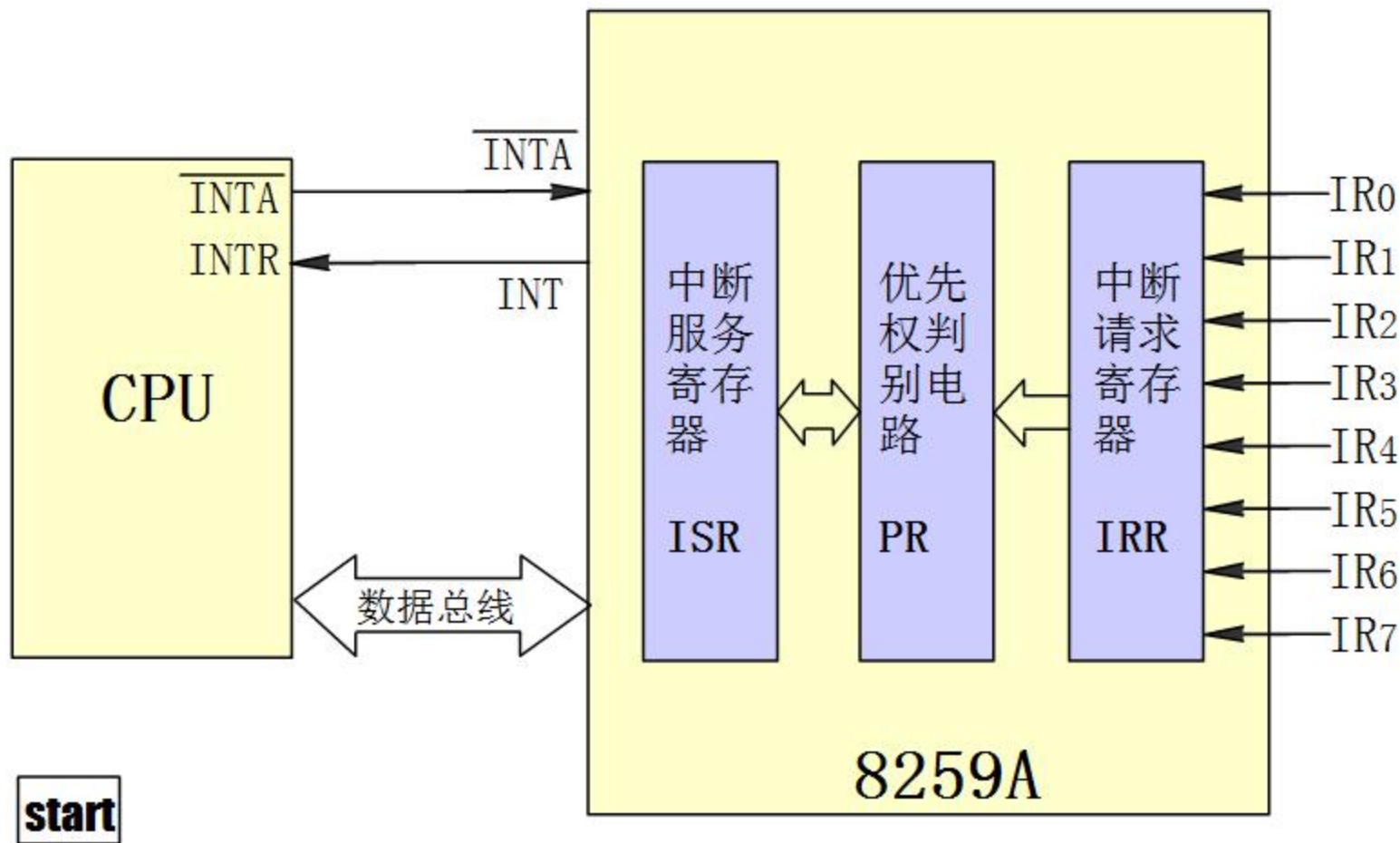


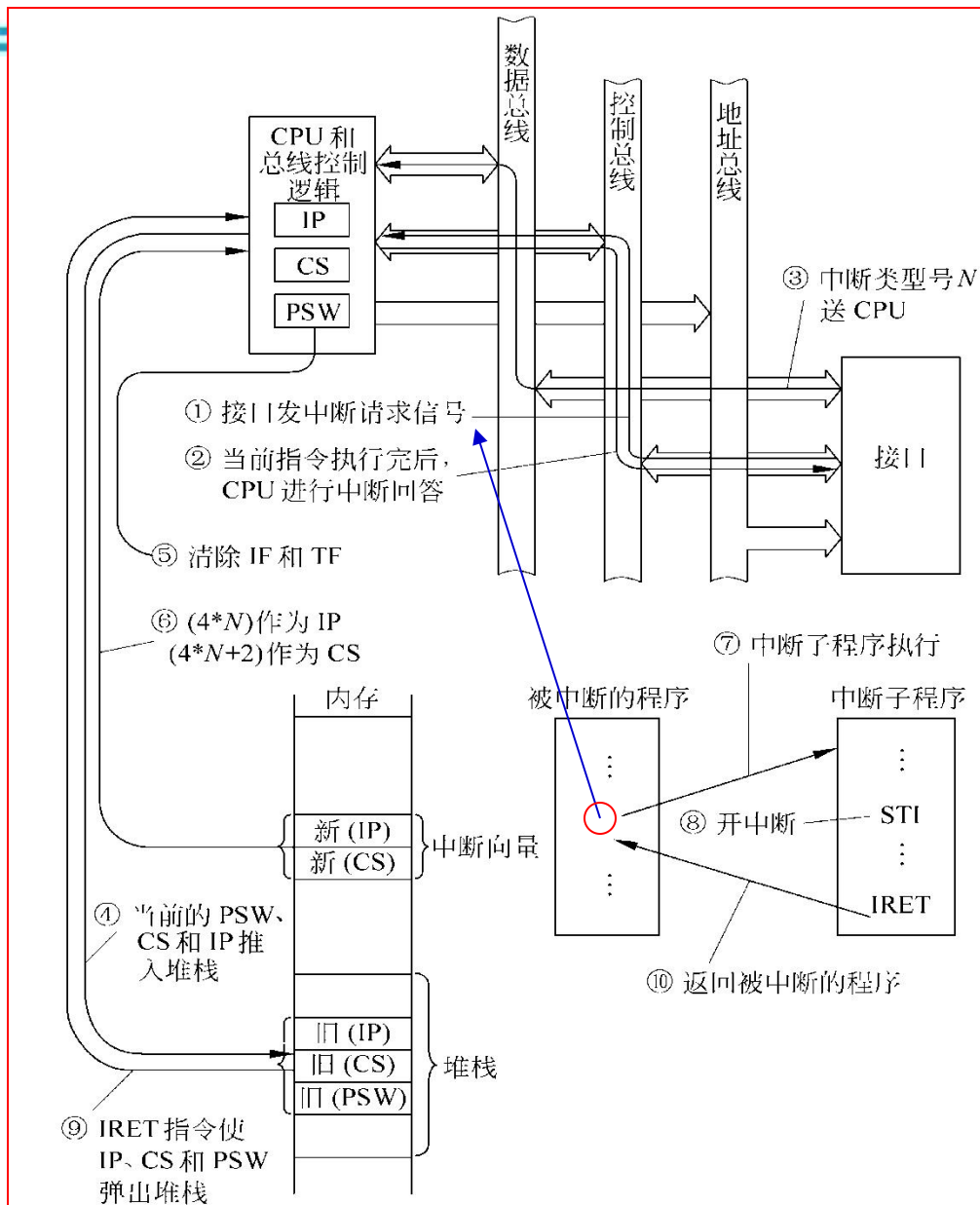
图5-16

8259级联工作示意图



8259A中断处理演示

# 中断控制方式





播放 停止

外设接口

60h

61h

内存

MAIN LEA BX, BUF

.....

.....

ADD AL, [BX]

ADC AH, 0

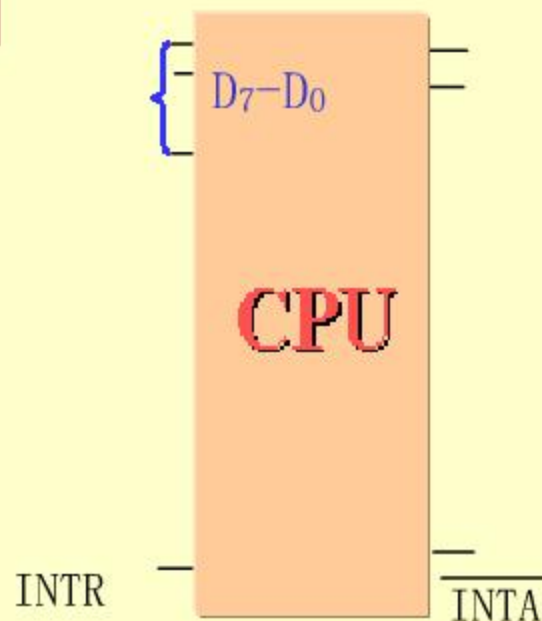
U\_INT

MOV AL, 50

OUT 60H, AL

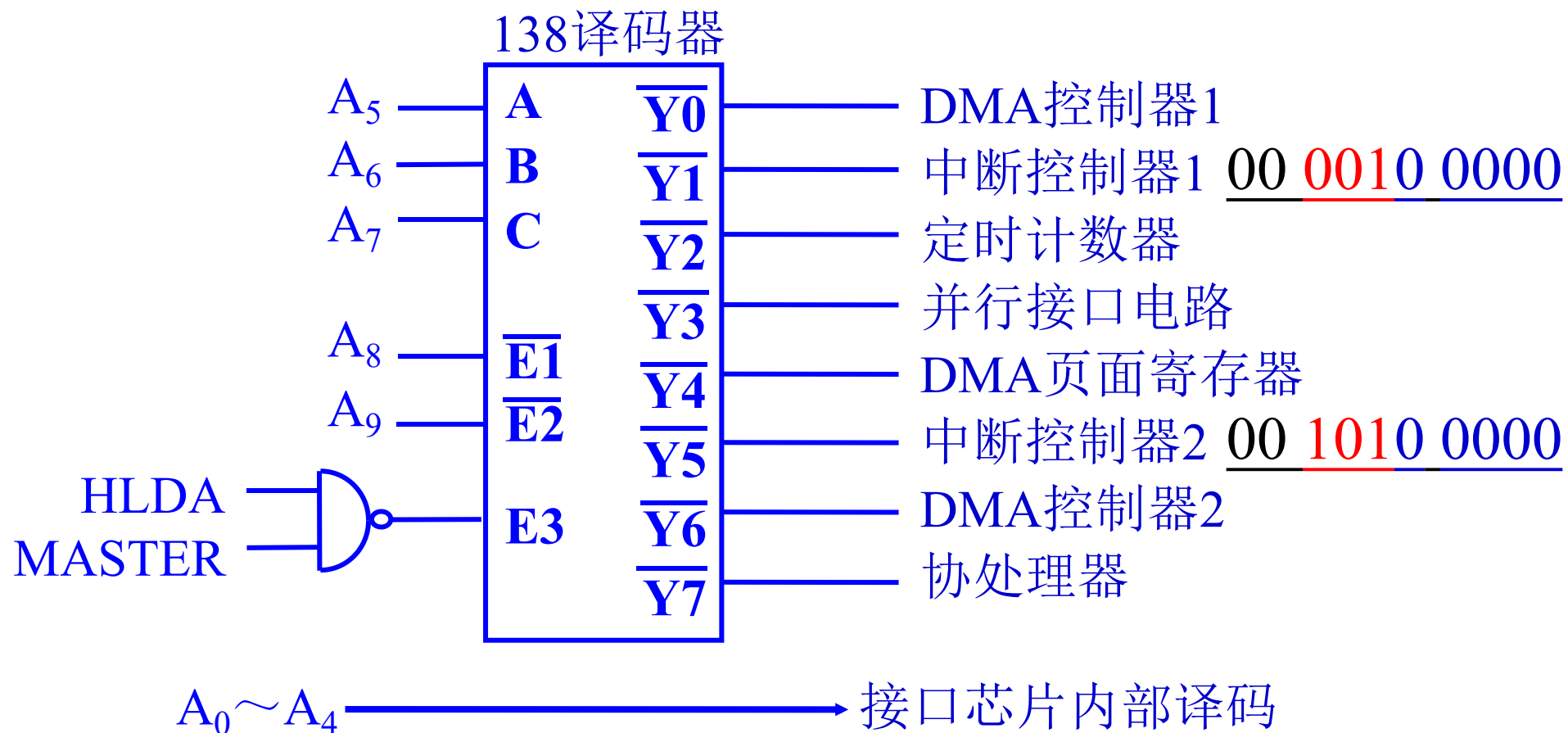
IN AL, 61H

中断服务程序




中断传送流程

# IBM PC/AT主机板的I/O译码电路



## 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. 了解8259A在IBM PC系列机上的应用情况

习题7（第189 ~ 190页）——7.1 7.2 7.4 7.14

补充习题：

1. 8088中断允许标志IF在什么情况是0，如何使其为1？
2. 8088CPU各种中断的优先权顺序是什么？
3. 说明IRET指令的功能。
4. 如何利用DOS功能调用设置中断向量？
5. 如何利用DOS功能调用获取中断向量？

# 8088的中断向量表

03FCH	向量号255的CS值	用户中断 (向量号255)
	向量号255的IP值	
	.....	
0008H	向量号2的CS值	非屏蔽中断 (向量号2)
	向量号2的IP值	
0004H	向量号1的CS值	单步中断 (向量号1)
	向量号1的IP值	
0000H	向量号0的CS值	除法错中断 (向量号0)
	向量号0的IP值	

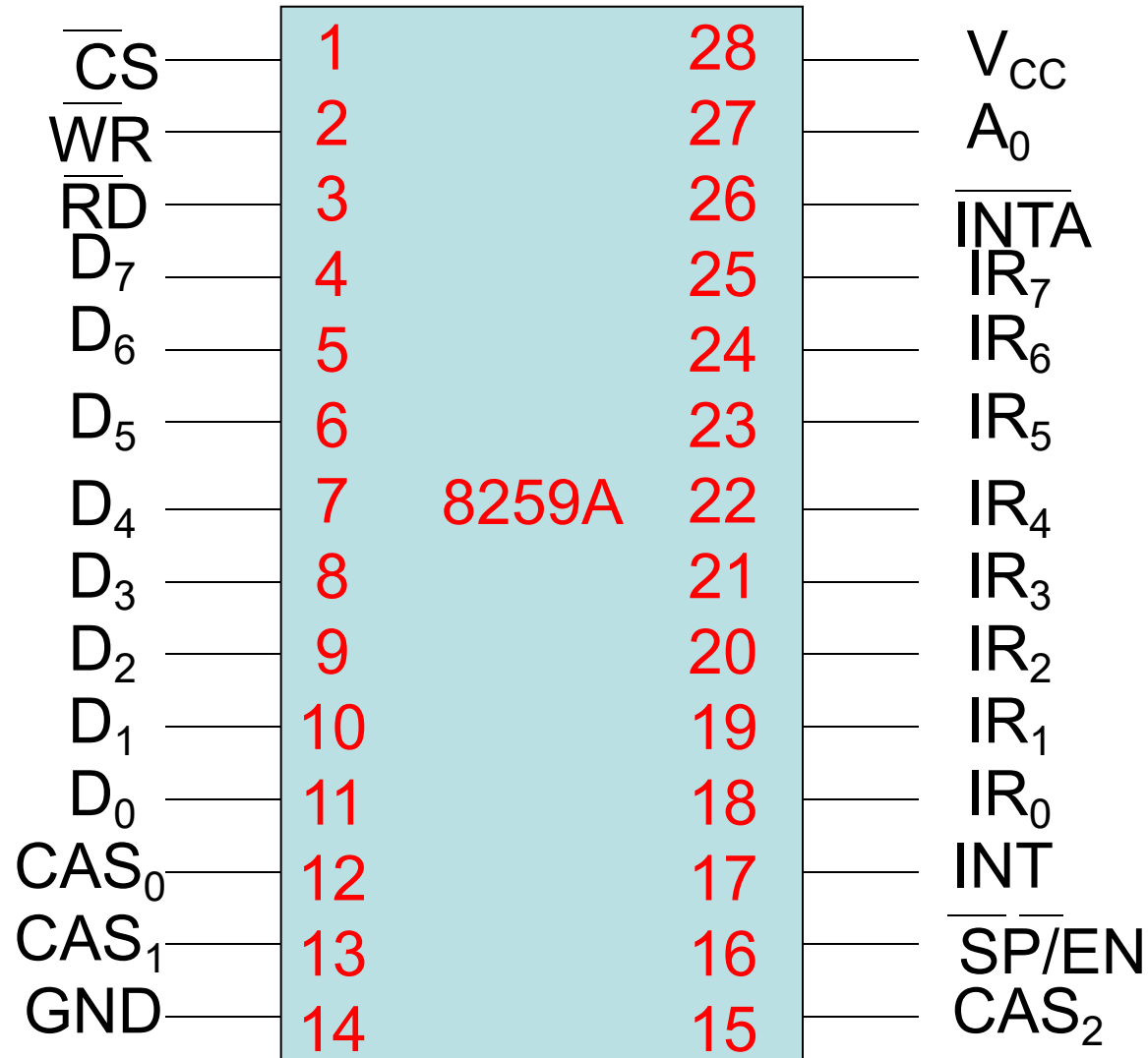


# 8088引脚

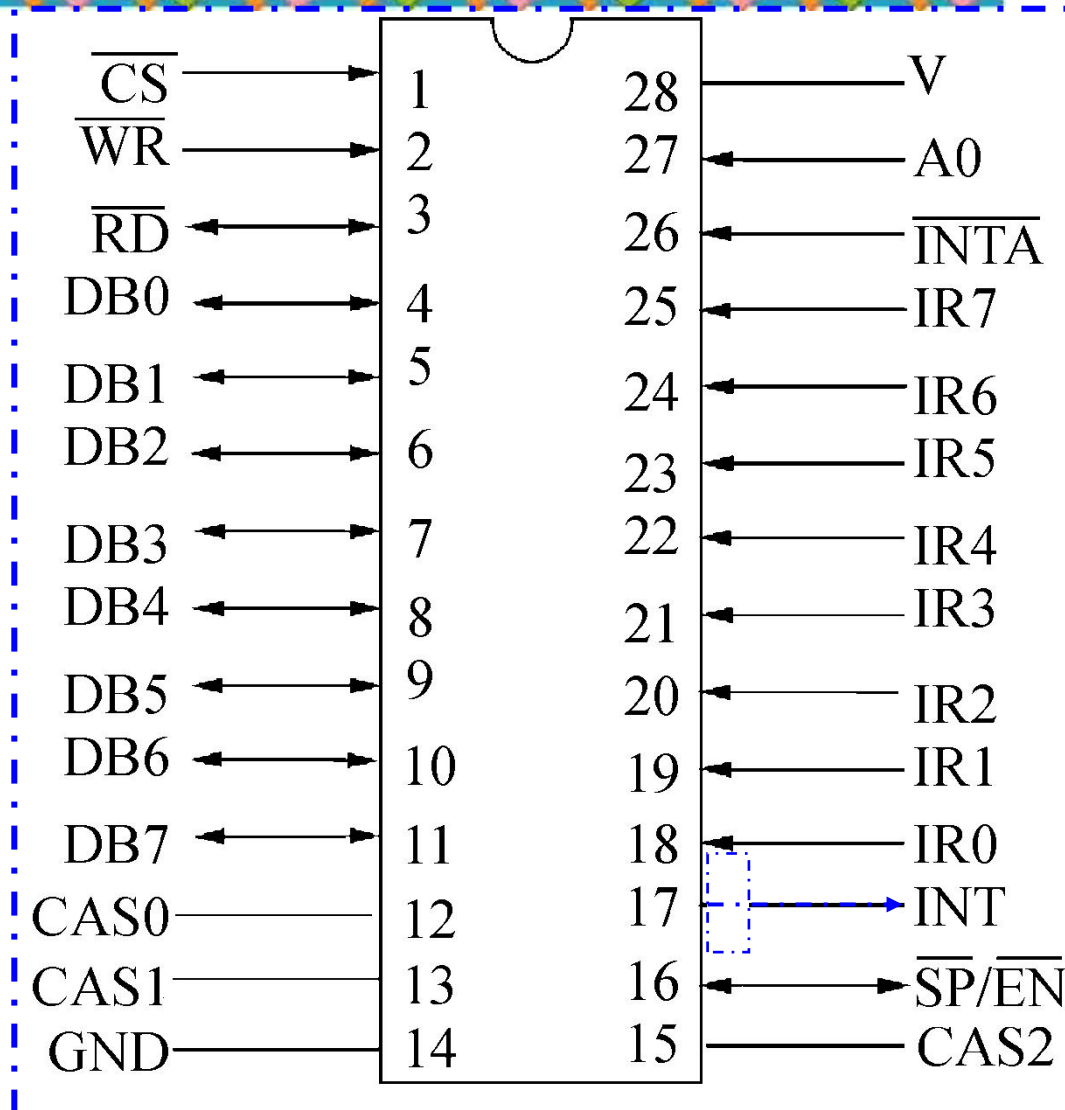
GND	1	40	VCC
A <sub>14</sub>	2	39	AD <sub>15</sub>
A <sub>13</sub>	3	38	AD <sub>16</sub> / S <sub>3</sub>
A <sub>12</sub>	4	37	AD <sub>17</sub> / S <sub>4</sub>
A <sub>11</sub>	5	36	AD <sub>18</sub> / S <sub>5</sub>
A <sub>10</sub>	6	35	AD <sub>19</sub> / S <sub>6</sub>
A <sub>9</sub>	7	34	SS0* (HIGH)
A <sub>8</sub>	8	33	MN / MX*
AD <sub>7</sub>	9	32	RD*
AD <sub>6</sub>	10	31	HOLD (RQ)* / GT0*
AD <sub>5</sub>	11	30	HLDA (RQ1* / GT1*)
AD <sub>4</sub>	12	29	WR* (LOCK*)
AD <sub>3</sub>	13	28	M / IO (S2*)
AD <sub>2</sub>	14	27	DT / R* (S1*)
AD <sub>1</sub>	15	26	DEN (S0)
AD <sub>0</sub>	16	25	ALE
NMI	17	24	INTA*
INTR	18	23	TEST*
CLK	19	22	READY
GND	20	21	RESET

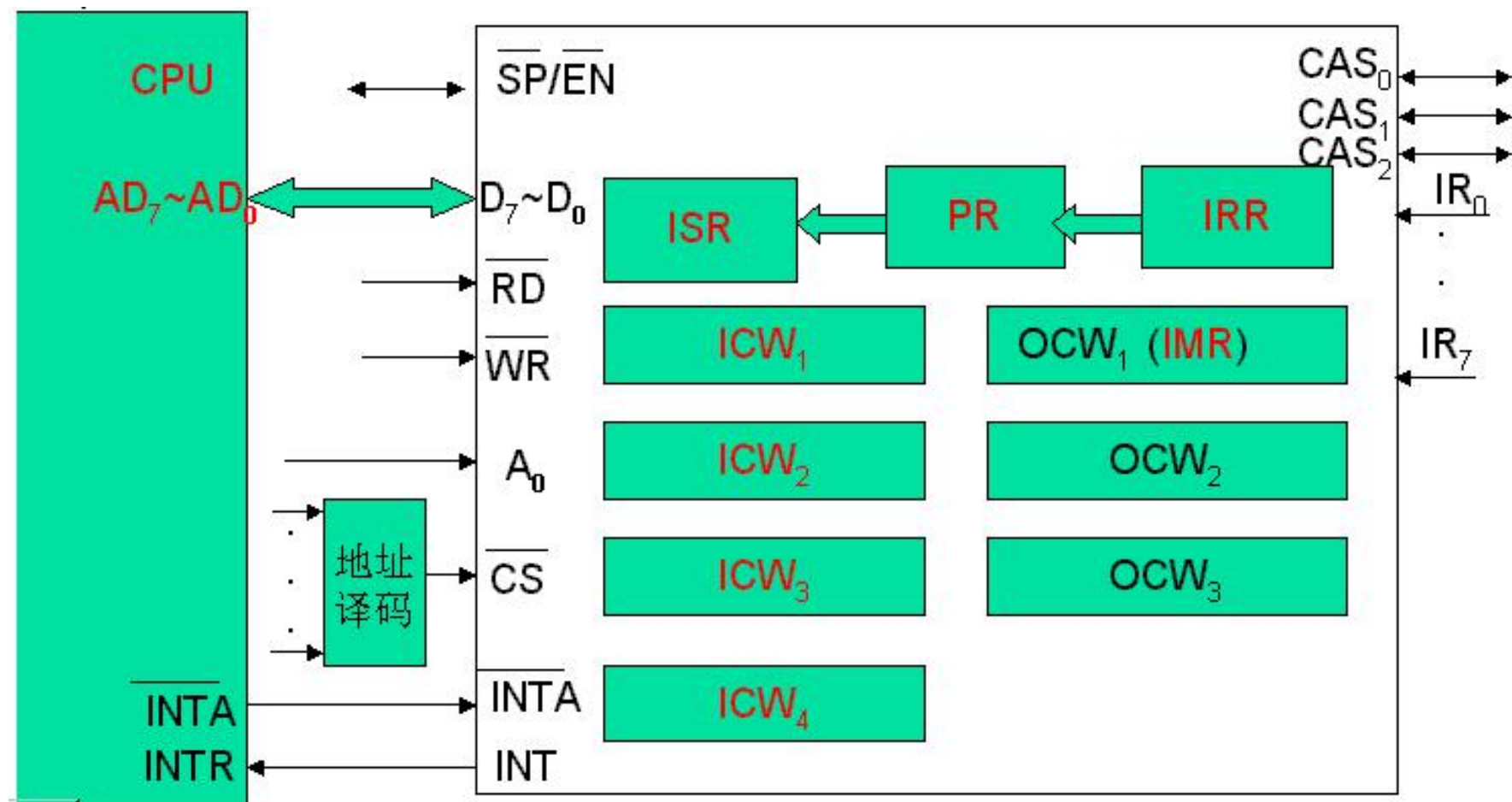


# 8259A引脚



# 8259A引脚







## 二. 8259A的工作方式

中断触发方式 { 边沿触发方式  
电平触发方式

屏蔽中断源方式 { 普通屏蔽方式  
特殊屏蔽方式

设置优先级方式 { 优先级固定方式 { 普通全嵌套方式  
特殊全嵌套方式  
优先级循环方式 { 自动循环方式  
特殊循环方式

结束中断处理方式 { 自动中断结束方式  
非自动中断结束方式 { 普通中断结束方式  
特殊中断结束方式

数据线连接方式 { 缓冲方式  
非缓冲方式

返回