哈爾濱工業大學(深圳)

HARBIN INSTITUTE OF TECHNOLOGY, SHENZHEN

汇编语言程序设计
第10讲：输入输出程序设计

裴文杰

I/O设备的数据传送方式

程序直接控制I/O方式

中断传送方式

- 8086/8088中断系统概述
- 中断分类
- 中断向量表
- 中断程序设计方法

wenjiecoder@outlook.com

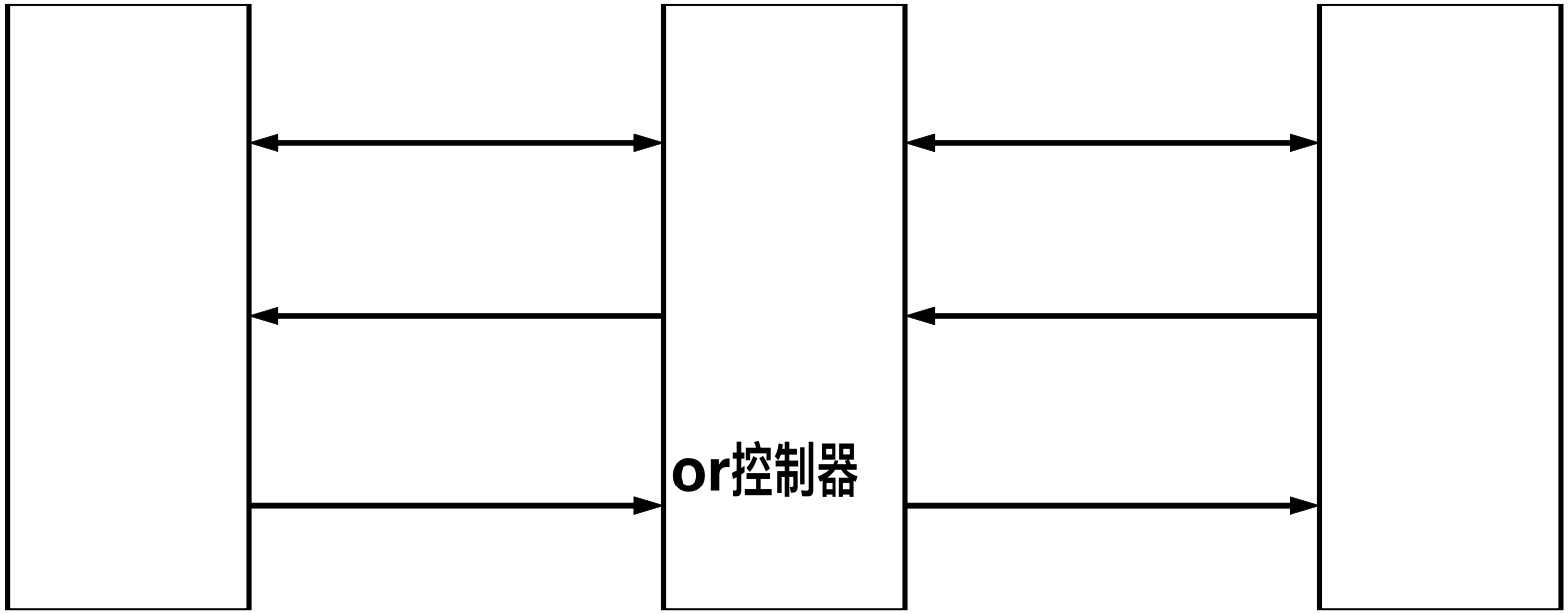
I/O设备的数据传送方式

程序直接控制I/O方式

中断传送方式

- 8086/8088中断系统概述
- 中断分类
- 中断向量表
- 中断程序设计方法

I/O设备的数据传送方式



主机与外设间的信息传送

I/O设备的数据传送方式

CPU与外设之间的接口信号：

1. 数据信息

这是CPU和外设之间真正要交换的信息。

2. 状态信息

它用来反映外设接口电路或外设的状态，CPU可根据这些状态信息决定对外设的操作或控制。

3. 控制信息

用于控制输入输出设备的启动或停止，设备的工作方式等。

以上三种不同性质的信息通过不同的端口传送，每个端口都有自己的地址，CPU寻址的是端口地址，而不是笼统的外设。

I/O设备的数据传送方式

主机与外设之间的数据传送(控制)方式：

- 直接存储器存取(DMA)方式
- 程序直接控制I/O方式（查询方式）
- 中断传送方式

直接存储器存取(DMA)方式（成组数据传送方式）

DMA（Direct Memory Access）方式能摆脱CPU的直接干预，利用硬件控制设备：DMA控制器(DMAC)，实现外部设备与内存间的直接数据传送，以提高数据传输速率。

主要用于一些高速I/O设备，如磁盘，模数转换器（A/D）等。

I/O设备的数据传送方式

因DMA方式主要有DMA控制器实现其传送功能，这里只介绍其基本概念，相关知识在“计算机组成原理”等课程中会有详细介绍。

在DMA模式下，CPU只须向DMA控制器下达指令，让DMA控制器来处理数据的传送，数据传送完毕再把信息反馈给CPU，这样就很大程度上减轻了CPU资源占有率，可以大大节省系统资源。

DMA控制器一般包括四个寄存器：控制寄存器（设置控制字：输入/输出、启动DMA等）、状态寄存器（DMAC状态）、地址寄存器（要传送数据块的首地址）、和字节计数器（要传送的数据字节数），这些寄存器在数据传送之前应该初始化。

在实现DMA传输时，由DMA控制器直接掌管总线，因此，存在着一个总线控制权转移问题。即DMA传输前，CPU要把总线控制权交给DMA控制器，而在结束DMA传输后，DMA控制器应立即把总线控制权再交回给CPU。

wenjiecoder@outlook.com

I/O设备的数据传送方式

系统完成DMA传送的步骤：

总线请求：DMA控制器向CPU申请使用总线

总线控制转移：CPU同意DMA控制器管理总线

数据传输：外设接口和存储器之间传输数据

传输数据块的首地址（在地址寄存器中）通过地址总线发出

传输的数据字节通过数据总线进行传送

地址寄存器加1，以指向下一个要传送的字节

字节字数器减1，如字节计数器非0，则继续传送

结束处理：DMA控制器放弃对总线的控制权

DMAC：
控制寄存器
状态寄存器
地址寄存器
字节计数器

wenjiecoder@outlook.com

I/O设备的数据传送方式

程序直接控制I/O方式

中断传送方式

8086/8088中断系统概述

中断分类

中断向量表

中断程序设计方法

wenjiecoder@outlook.com

程序直接控制I/O方式

数据寄存器
状态寄存器
命令寄存器

I/O端口：

- 外设都是通过接口连接到计算机系统上，每个接口由一组寄存器组成，这些寄存器都分配有一个称为I/O端口的地址编码。
- CPU就是通过不同的端口号来选择各种外部设备的。
- 在80X86微机中，I/O端口编址在一个独立的地址空间中，这个空间允许设置64K个8位端口，或32K个16位端口。
- 有两种寻址方式。
  - 直接寻址：只用于寻址00H～0FFH
    前256个端口，操作数表示端口号。
  - 间接寻址：可用于寻址全部64K个端口，DX寄存器的值就是端口号，
    对大于0FFH的端口只能采用间接寻址方式。

程序直接控制I/O方式

所有I/O端口与CPU之间的通信都由IN和OUT指令来完成。其中：

IN指令完成从I/O端口到CPU的数据传送（输入）。

OUT指令完成从CPU到I/O端口的数据传送（输出）。

输入指令IN：（只限使用AX或AL）

长格式：　IN　AL, PORT　（字节）

　　IN　AX, PORT　（字）

执行操作：(AL) ← (PORT)　（字节）

　　(AX) ← (PORT+1, PORT)（字）

短格式：　IN　AL, DX　（字节）

　　IN　AX, DX　（字）

执行操作：(AL) ← ((DX))　（字节）

　　(AX) ← ((DX)+1, (DX))（字）

这里的端口号或DX的内容均为地址，而传送的是端口中的信息。

程序直接控制I/O方式

注意：

- 不影响标志位
- 前256个端口号00H~FFH可直接在指令中指定（长格式）
- 如果端口号？256，端口号？DX（短格式）

IN　AL，　60H　；从端口60H读入一个字节到AL中

IN　AX，20H　；把端口21H、20H按“高低”组成的字读入AX

MOV　DX，2F8H；

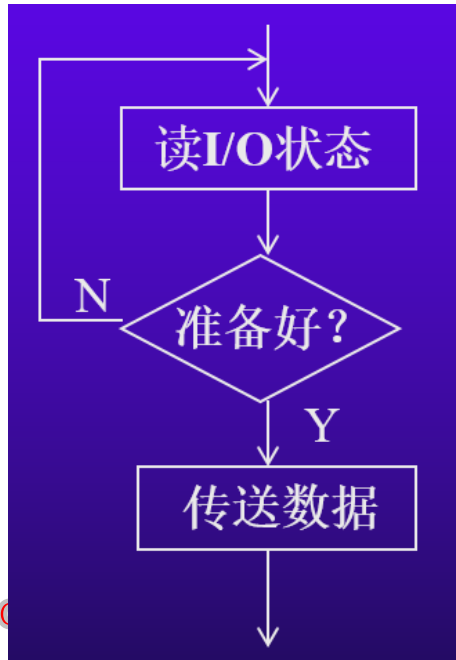
IN　AX，DX；把端口2F9H、2F8H按“高低”组成的字读入AX

MOV　DX，2F8H

IN　AL，DX；从端口2F8H读入一个字节到AL中

例：测试某状态寄存器的第2位是否为1，若为1则传送数据。

```
AGAIN: IN    AL, STATUS_PORT
       TEST  AL, 00000100B
       JZ    AGAIN
       MOV   AL, DATA
       OUT   DATA_PORT, AL
```



程序直接控制I/O方式

输出指令 OUT （CPU → I/O）

长格式： OUT PORT, AL （字节）

OUT PORT, AX （字）

执行操作：(PORT) → (AL) （字节）

(PORT+1, PORT) → (AX) （字）

短格式： OUT DX, AL （字节）

OUT DX, AX （字）

执行操作：((DX)) → (AL) （字节）

((DX)+1, (DX)) → (AX) （字）

字节输出

```
mov dx, 3fch
mov al, 80h
out dx, al
```

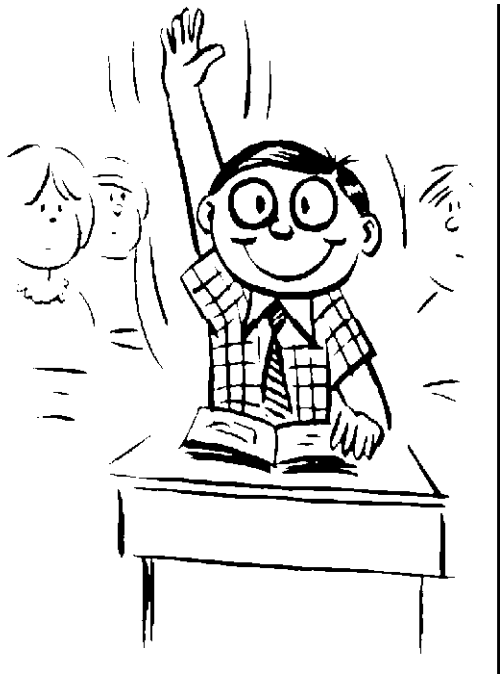
```
mov al, 80h
out 3fch, al
    (X)
```


wenjiecoder@outlook.com

思考题：

判断下列各条语句的对错：

IN AX, 20H	√
IN AL, DX	√
IN AH, 33H	×
IN AX, 200H	×
OUT AX, DX	×
OUT 21H, AL	√
OUT DL, AL	×
OUT DX, AL	√
OUT DX, AX	√



wenjiecoder@outlook.com

程序直接控制I/O方式

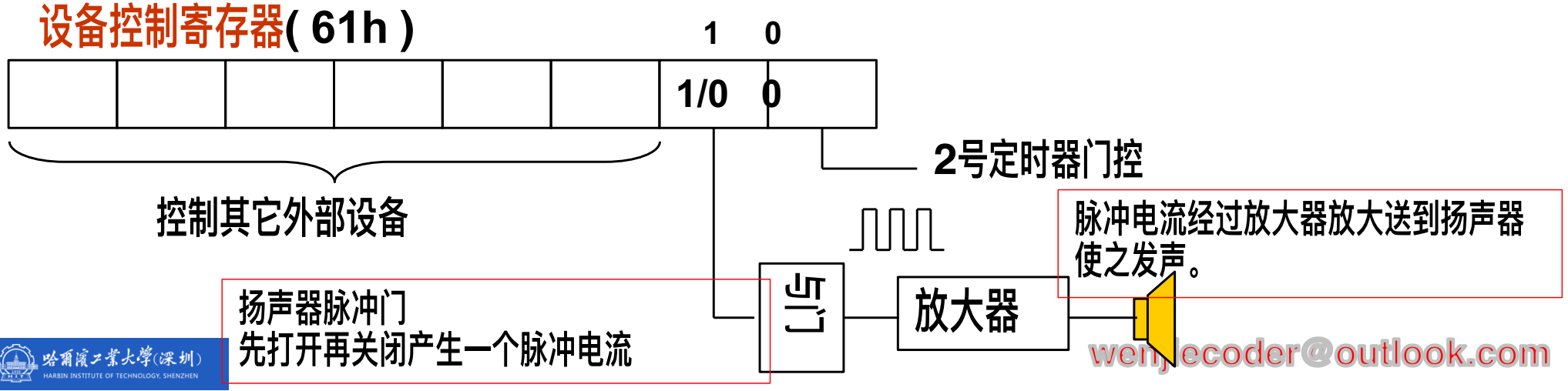
例：Sound 程序

通过直接控制扬声器来发出声音

。

```
CODES SEGMENT
START:
        mov   dx,  0ffffH       ;响铃持续多长
        in    al,  61h
        and   al,  11111100b
sound:
        xor   al,  2
        out   61h, al
        mov  cx,  140h          控制声音频率
wait1:
        loop  wait1             ;延时
        dec   dx
        jne   sound

        MOV AH,4CH
        INT 21H
CODES ENDS
   END START
```



I/O设备的数据传送方式

程序直接控制I/O方式

中断传送方式

8086/8088中断系统概述

中断分类

中断向量表

中断程序设计方法

中断传送方式

中断系统概述：

- 当CPU正在执行某程序时，由于外界事件的需要向CPU发出申请，CPU暂停现行程序的执行而转去处理临时发生的事件，处理完后再返回到被中断程序的断点处，继续向下执行，这个过程称为**中断**。
- 中断是CPU和外部设备进行I/O的有效方法。

  它可以避免因反复查询外部设备的状态而浪费时间，提高CPU的效率。
- 在中断过程中执行的事件处理程序称为**中断服务程序**或者**中断处理程序**。

wenjiecoder@outlook.com

中断传送方式

中断请求与中断源：

I/O设备或者事件需要CPU中断处理时，必须向CPU发出中断请求信号。当CPU收到该信号时，可引起中断。

引起中断的原因，或者说发出中断请求信号的源称为中断源。通常，中断源有以下几种：

（1）一般的输入输出设备。如键盘、打印机、通信接口等。

（2）数据通道中断源，如磁盘。

（3）实时时钟。

（4）故障源。

（5）为调试程序而设置的中断源。

wenjiecoder@outlook.com

中断传送方式

中断分类：

8086可以管理256个中断，每个中断有相应的中断类型号。

中断源可能来自外部设备的I/O请求，也可能是计算机的一些异常事故或其他内部原因。主要分为：

软件中断（内中断）：由程序安排的中断指令产生的中断、或CPU的某些错误结果产生的中断。

由中断指令INT引起；

由CPU的某些错误而引起；

为调试程序（DEBUG）设置的中断。

硬件中断（外中断）：由外设控制器或协处理器引起的中断（如键盘、鼠标等）。

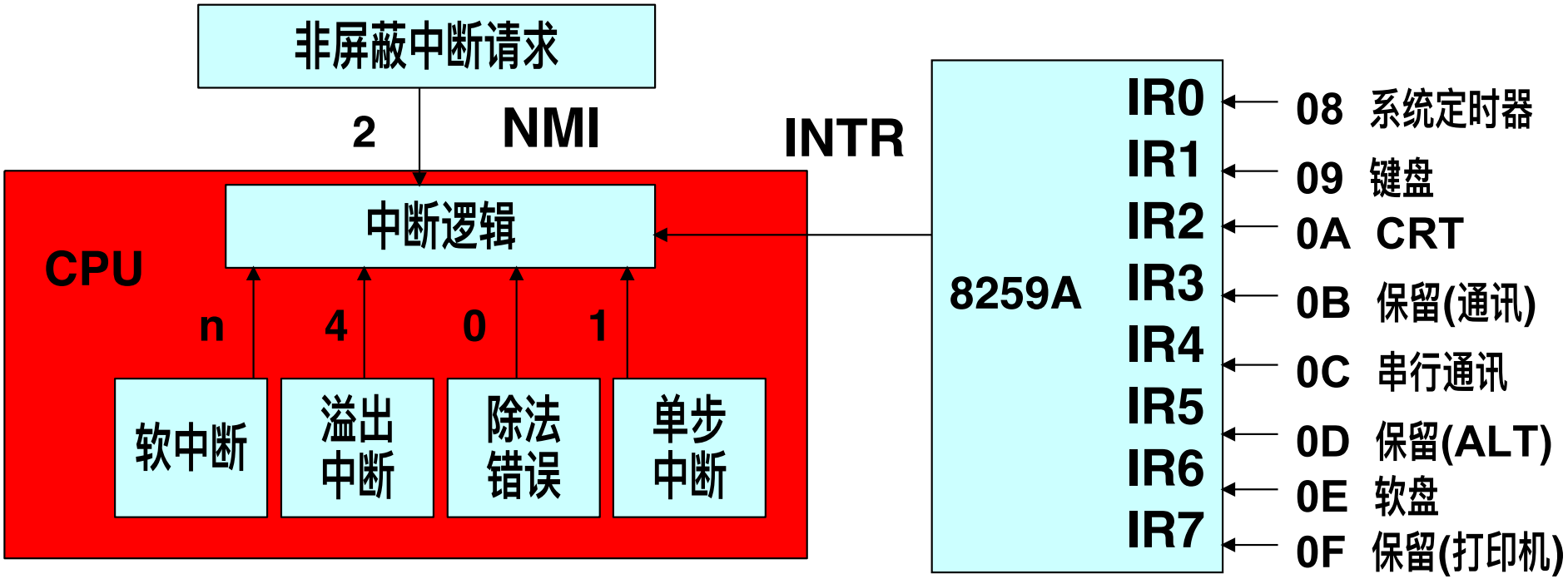
可屏蔽中断：可被中断允许标志IF屏蔽。

非屏蔽中断（NMI）：为电源错、内存或I/O总线的奇偶校验等异常事件保留的中断。它不受中断允许标志IF的屏蔽。整个系统只能有一个NMI（中断类型号为2）。

wenjiecoder@outlook.com

中断传送方式

8086/8088的中断系统



8259A：可编程中断控制器
负责外部设备的可屏蔽中断，
CPU通过一组I/O端口控制8259A

内中断

内中断——CPU内部执行程序引起的中断，又可以分成：

（1）中断0——除法错中断；除数为0或商超过了寄存器的范围

（2）中断1——单步中断； TF=1，允许单步调试

（3）中断3——断点中断；用于断点调试（INT 3）

（4）中断4——溢出中断(INTO)；执行溢出中断指令，OF=1时产生

（5）用户定义的软件中断(指令中断)：执行中断调用指令INT n产生的n号中断，如INT 21h

所有的内部中断都具有下述特点：

1）中断向量码或者包含在指令中，或者是预定的；

2）除单步中断外，内部中断都无法禁止；

3）除单步中断外，任何内部中断的优先级都比任何外部中断的高。

1. INT 中断调用指令

当cpu处理完中断请求返回原程序时，要保证原程序工作的连续性和正确性，所以中断发生时的Flags内容也要保存起来。

指令汇编格式：INT n

操作：SP←SP－2，（SP+1，SP）←Flags

IF ←0，TF ←0

SP ←SP－2，（SP+1，SP）← CS

SP ←SP－2，（SP+1，SP）← IP

IP ←（n*4+1，n*4）

CS ←（n*4+3，n*4+2）

自动清除IF和TF，cpu转入中断处理程序后，不允许再响应新的中断和单步调试，如果此时还想允许外部中断，可以通过STI再把IF置为1.

受影响的标志位：IF，TF

说明：n称为中断类型号，必须是0~255之间的立即数。

2. IRET 中断返回指令

指令汇编格式：**IRET**

操作：IP ← (SP+1，SP)，SP ←SP+2

CS ← (SP+1，SP)，SP ←SP+2

Flags ← (SP+1，SP)，SP ←SP+2

受影响的标志位：所有状态标志位。

说明：

IRET指令是任何中断服务程序的最后一条要执行的指令，它使CPU从中断服务程序返回被中断程序的断点处继续执行。

中断传送方式

外中断：可屏蔽中断和非屏蔽中断。

- 可屏蔽中断（INTR）：用户能控制的中断。用户可通过软件禁止可屏蔽中断的中断源发出的中断请求信号，或者关闭中断而对发出的请求不作响应。
- 非屏蔽中断（NMI）：用户不能禁止的中断。这种中断一旦出现，CPU必须立即响应，所以非屏蔽中断总是用于处理紧急事件。

整个系统只能有一个NMI（中断类型号为2）。

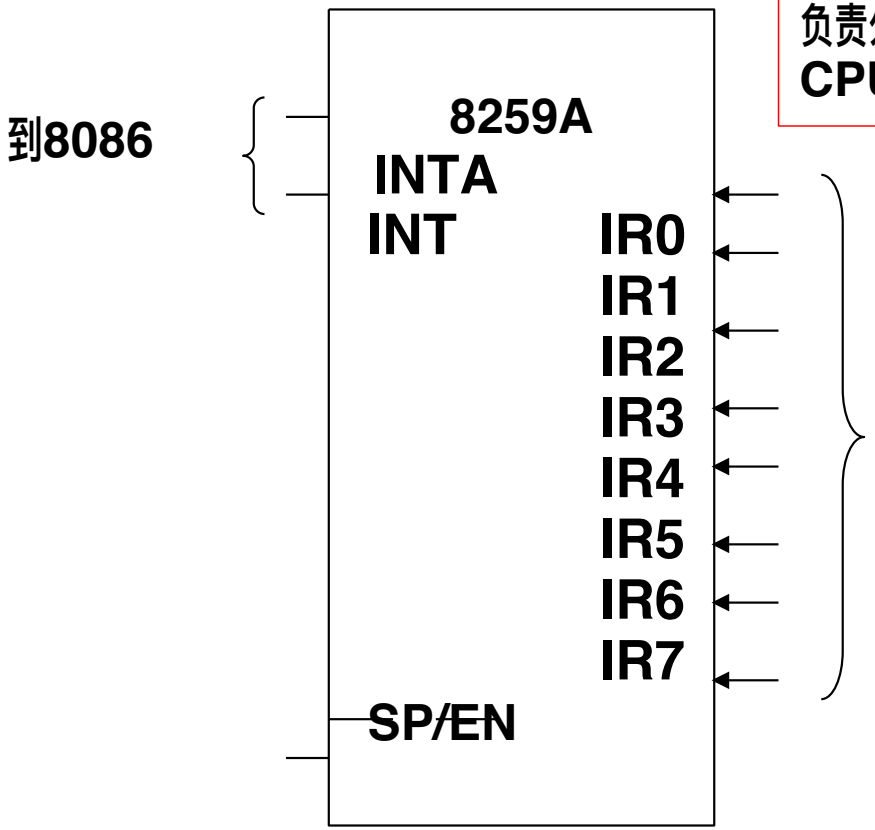
1）可屏蔽中断

- 其主要任务是接收外部设备的中断请求，然后根据优先级的高低和预先规定的排优规则决定哪个设备能够申请中断，由8259A向CPU发中断请求信号。
- 如果CPU响应此中断请求，就自动转入相应的中断处理程序。

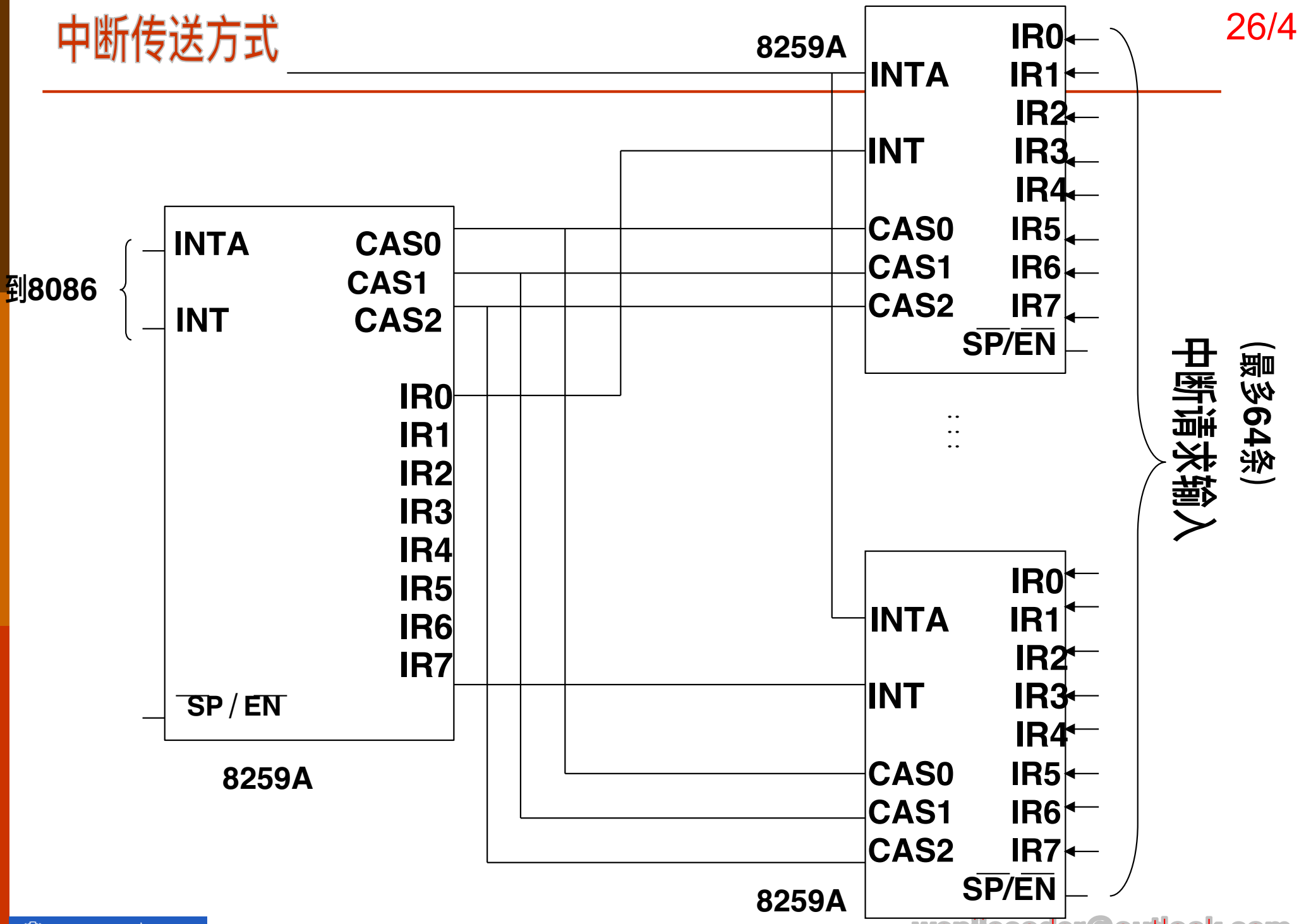
中断传送方式

每个8259A有8个中断请求输入端，因此单个8259A可以处理8级中断，通过级连8259A最多可以管理64级中断。

8259A：可编程中断控制器
负责外部设备的可屏蔽中断，
CPU通过一组I/O端口控制8259A



中断传送方式



中断传送方式

注意：从外设发出的中断请求到CPU响应中断，有两个控制条件起决定作用：

1)外设的中断请求是否被屏蔽，

2)CPU是否允许响应中断。

这两个条件分别由8259A的中断屏蔽寄存器（IMR）和标志寄存器中的中断允许位IF控制。

CLI	IF=0	关中断
STI	IF=1	开中断

中断屏蔽寄存器的I/O地址是21H，它的8位对应控制8个外部设备，某位0表示允许中断，1表示禁止中断。

上述2个条件需同时满足，才允许中断。

中断传送方式

中断的条件：

设置CPU中断允许位：

FLAGS 中的 IF 位 = 1 允许中断 （开中断指令：STI）

= 0 禁止中断 （关中断指令：CLI）

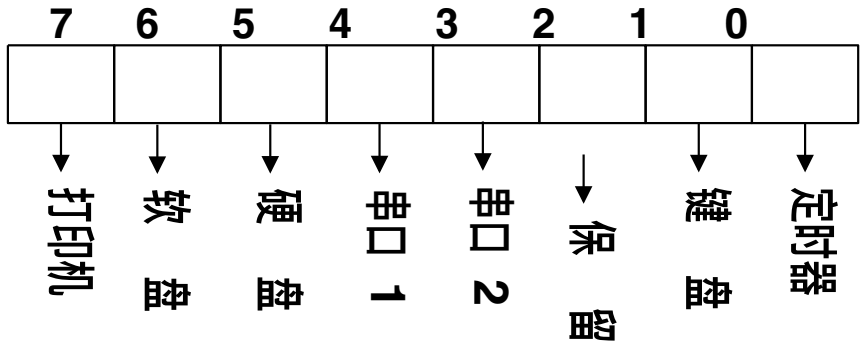
设置中断屏蔽位：

中断屏蔽寄存器的中断屏蔽位 = 0 允许I/O设备请求中断

= 1 禁止I/O设备请求中断

8259A

中断屏蔽寄存器21H



如：允许增设键盘中断

```
IN    AL, 21H
AND AL, 0FDH
OUT  21H, AL
```

中断传送方式

中断结束位为0：以后将屏蔽掉对同级中断或者低级中断的响应。
将EOI置为1，不是结束正在处理的中断处理，而是允许对同级中断或者低级中断的响应。

中断命令寄存器：

- 在一次中断处理结束之前，还应给8259A的中断命令寄存器发出中断结束命令（End of Interrupt, EOI, 第5位）。中断命令寄存器的I/O端口地址为20H。
- 当EOI=1时，当前正在处理的中断请求被清除。所以在中断处理完成后，必须把EOI为置1。
- L2~L0指定IR0-IR7中具有最低优先级的中断请求，第6、7位控制IR0-IR7中断优先级的顺序。

中断结束位EOI在程序的什么时候发出，取决于是否在中断处理过程中允许同级或者低级中断。一般设备希望一次中断的处理过程最好是完整的，所以在中断处理结束之前发出EOI命令。

8259A中断命令寄存器20H

7	6	5	4	3	2	1	0
R	SL	EOI	0	0	L2	L1	L0

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

0010 0000

端口号

CPU对外部中断的响应步骤

当外设通过8259A向CPU提出申请，且CPU的IF=1时，CPU就挂起正在处理的任务，进行中断响应和处理。整个过程如下：

响应中断；
将标志寄存器的内容压栈；
清除中断允许标志位IF和陷阱标志位TF；
将代码段寄存器CS和指令指针IP的内容压栈；
根据中断码找到服务程序入口，且调用服务程序；
执行用户中断服务程序；
将保存在栈中的IP和CS的内容从栈中弹回到IP和CS；
将保存在栈中的标志寄存器的内容从栈中弹回到标志寄存器；
从中断返回。

以上过程的1~6由硬件自动完成，7~9执行IRET指令实现。

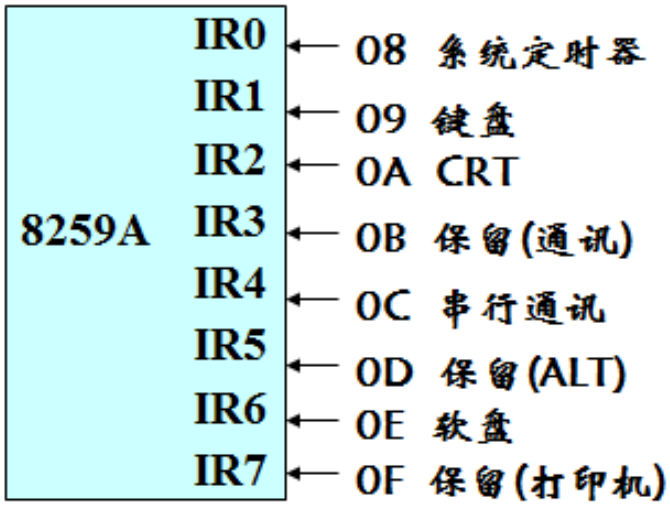
中断传送方式

中断优先级

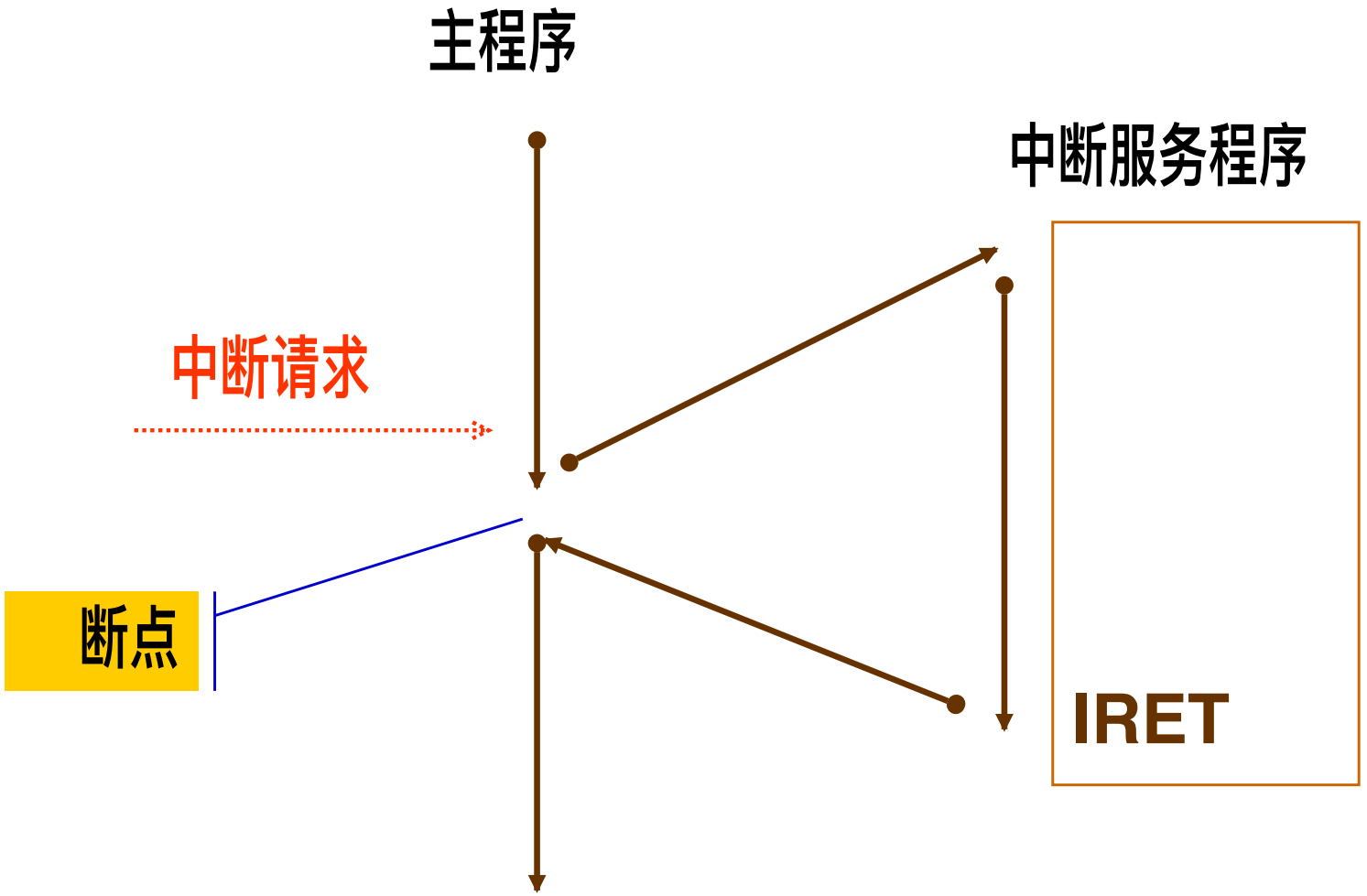
中　　断	优先级
除法错、INT n、INTO	最高
NMI	
INTR	
单步中断	最低

可屏蔽中断（INTR）的优先级又分为八级，正常情况下优先级由高到低依次是：

IR0，IR1，IR2，IR3，IR4，IR5，IR6，IR7



中断传送方式



中断服务程序和子程序的调用方式类似。

中断传送方式

中断和子程序的比较

中断和子程序调用之间有其相似和不同之处。它们的工作过程非常相似，即：暂停当前程序的执行，转而执行另一程序段，当该程序段执行完时，CPU都自动恢复原程序的执行。它们的主要差异有：

子程序调用一定是程序员在编写源程序时事先安排好的，是可知的，而中断是由中断源根据自身的需要产生的，是不可预见的（用指令INT引起的中断除外）；

子程序调用是用CALL指令来实现的，但没有调用中断的指令，只有发出中断请求的事件（指令INT是发出内部中断信号，而不要理解为调用中断服务程序）；

子程序的返回指令是RET，而中断服务程序的返回指令是IRET。中断处理程序会将Flag入栈，并清除IF、TF标志。

在通常情况下，子程序是由应用软件的开发者编写的，而中断服务程序是由系统软件设计者编写的。

wenjiecoder@outlook.com

中断传送方式

中断向量表：中断服务程序的入口地址。

- 8086/8088在其内存的前1K字节（地址为0000H~03FFH）建立了一个中断向量表。
- 表中内容分为256项，对应于类型号0~255。
- 每一项占用4个字节单元，用来存放相应类型的中断服务程序的入口地址。
- 高两个字节存放服务程序的入口地址的段地址部分。低两个字节存放偏移地址。

中断传送方式

中断向量表：

若中断号为N，则其中断子程序入口地址在向量表中的位置为：

N*4

IP=(N*4+1, N*4)

CS =(N*4+3, N*4+2)

采用中断向量表的方法，大大加快了中断处理的速度。因为计算机可直接通过中断向量表转向相应的中断处理（服务）程序，而不需要cpu去逐个检测和确定中断原因。

例：若中断号为20H，则中断向量为：

IP=2010H

CS=4030H

wenjiecoder@outlook.com

中断传送方式

例：以BIOS中断INT 4AH为例，表示中断操作的5个步骤：

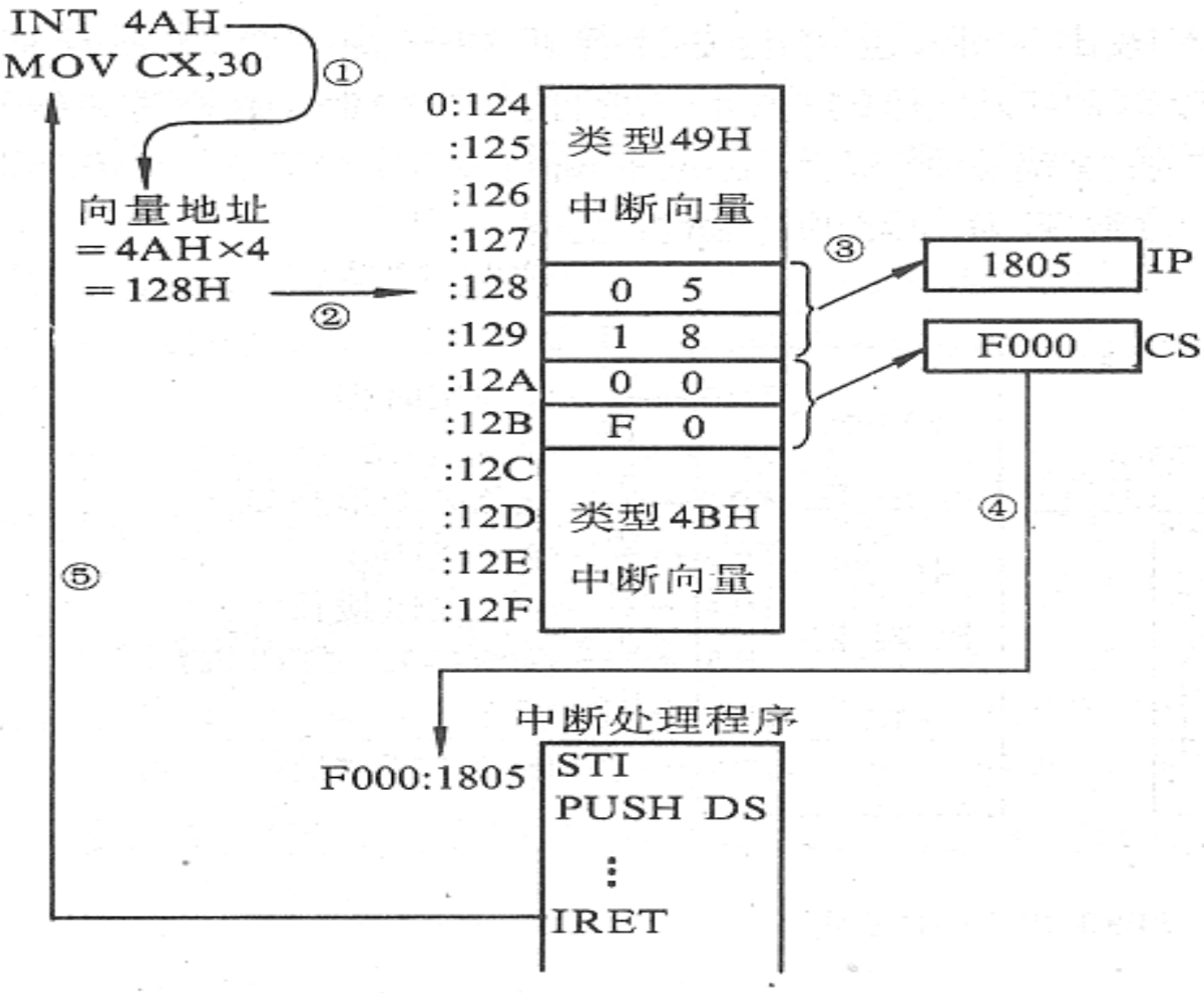
取中断类型号；

计算中断向量地址；

取中断向量，偏移地址送IP，段地址送CS；

转入中断处理程序；

中断返回到INT指令的下一条指令。



wenjiecoder@outlook.com

表 8.2　中断向量分配

地址	中断		地址	中断	
0—7F	0—1F	BIOS 中断向量	1C0—1DF	70—77	I/O 中断向量
80—FF	20—3F	DOS 中断向量	1E0—1FF	78—7F	保留
100—17F	40—5F	扩充 BIOS 中断向量	200—3C3	80—FD	BASIC 中断向量
180—19F	60—67	用户中断向量	3C4—3FF	F1—FF	保留
1A0—1BF	68—6F	保留			

如：DOS功能调用INT　21H

BIOS功能调用INT　10H（显示）、INT　14H（串口通信）等

中断传送方式

- 用户可以利用保留的中断类型号扩充自己需要的中断功能，对新增加的中断功能要在中断向量表中建立相应的中断向量。
- 如果中断功能只供自己使用，或者自己编写的中断处理程序代替系统原有的某个中断处理功能时，需要注意保存原中断向量。按保存-设置-恢复的顺序操作。
- 可调用DOS功能调用（21H）来存取中断向量：

设置中断向量：把由AL指定的中断类型的中断向量DS：DX放入中断向量表。

预置：AH=25H，AL=中断类型号，DS：DX=中断向量

执行：INT 21H

读取中断向量：把由AL指定的中断类型的中断向量从中断向量表中取到ES：BX。

预置：AH=35H，AL=中断类型号

执行：INT 21H

返回：ES：BX=中断向量

例：为已有的某个中断类型 N 设置中断向量

```
                MOV  AL, N
                MOV  AH, 35H
                INT   21H
                PUSH ES
                PUSH BX                          ;保存原来的中断向量
                PUSH DS
                MOV  DX, OFFSET  INTHAND         ;偏移地址=>DX
                MOV  AX, SEG     INTHAND
                MOV  DS, AX                       ;段地址=>DS
                MOV  AL, N
                MOV  AH, 25H
                INT   21H                        ;设置新的中断向量
                POP   DS
                ……
                POP   DX
                POP   DS
                MOV  AL, N
                MOV  AH, 25H
                INT   21H                        ;恢复原来的中断向量
                RET
                ……
INTHAND:              ;中断处理程序
                ……
                IRET
```

中断传送方式

中断处理程序的结构：

与子程序(即过程)相似，可用定义过程的方式来定义中断处理程序。所有编写过程的一些规定和要求均适用于中断处理程序，包括用伪指令PROC/ENDP定义过程，类型为远类型。

中断程序的调用和编写步骤：

主程序：

(1) 设置中断向量

(2) 设置 CPU 的中断允许位 IF

(3) 设置设备的中断屏蔽位

中断处理子程序：

(1) 保存寄存器内容

(2) 如允许中断嵌套，则开中断 (STI)

(3) 中断处理功能

(4) 关中断

(5) 送中断结束命令(EOI)给中断命令寄存器

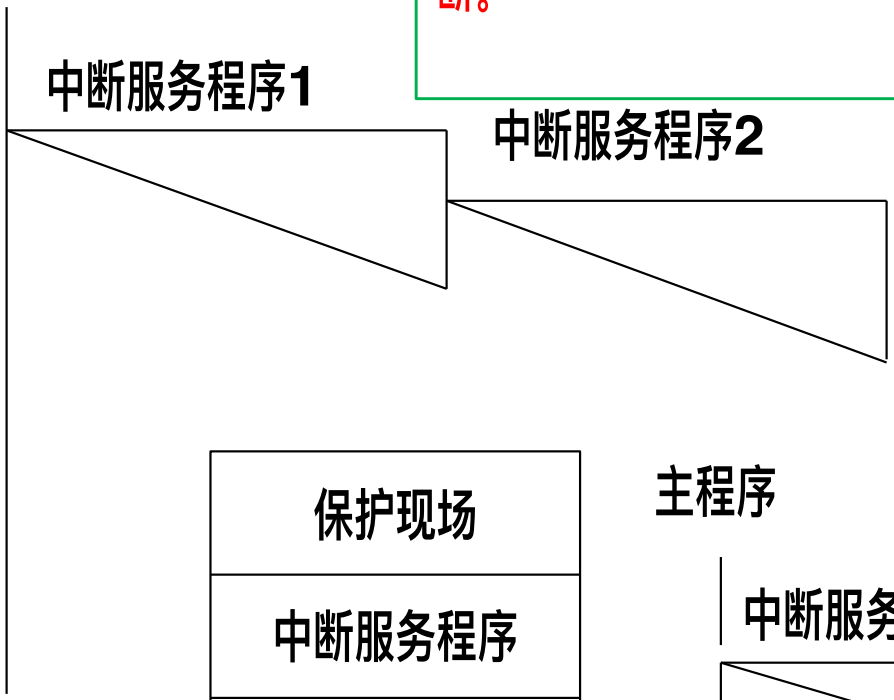
(6) 恢复寄存器内容

(7) IRET中断返回

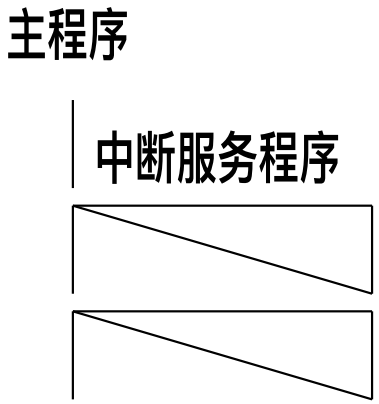
中断传送方式

一个正在执行的中断处理程序，在开中断（IF=1）的情况下,能被优先级更高的中断源中断，但如果要被同级或低级的中断源中断，则必须发出EOI命令，清除正在执行的中断请求，才能响应同级或者低级的中断。

中断服务程序设计：



多级中断服务程序结构



单级中断服务程序结构

中断传送方式

例：中断处理程序的一般结构的程序如下。

```
INTPRG   PROC  FAR
         STI                          ；若允许中断嵌套
         PUSH  DS
         PUSH  DX
         PUSH  AX
         PUSH  BX
         ……                         ；中断处理
         CLI                          ；关中断
         MOV  AL, 20H                 ；发中断结束命令EOI
         OUT  20H, AL
         POP  BX                      ；恢复现场
         POP  AX
         POP  DX
         POP  DS
         IRET                         ；中断返回
 INTPRG   ENDP
```

由于IRET将恢复中断前的标志，故IF也被恢复。

第10讲作业：

Page 313-314：8.9，8.10

wenjiecoder@outlook.com