中断技术

5.1 中断的基本概念

- 为什么要用程序中断?
  - 程序查询方式的缺点
    1. 在查询过程中，CPU长期处于等待状态
    2. 在一段时间内只能和一台外设交换信息，其他设备不能同时工作
    3. 不能发现和处理预先无法估计的错误和异常情况
  - 提高输入和输出能力和CPU的效率

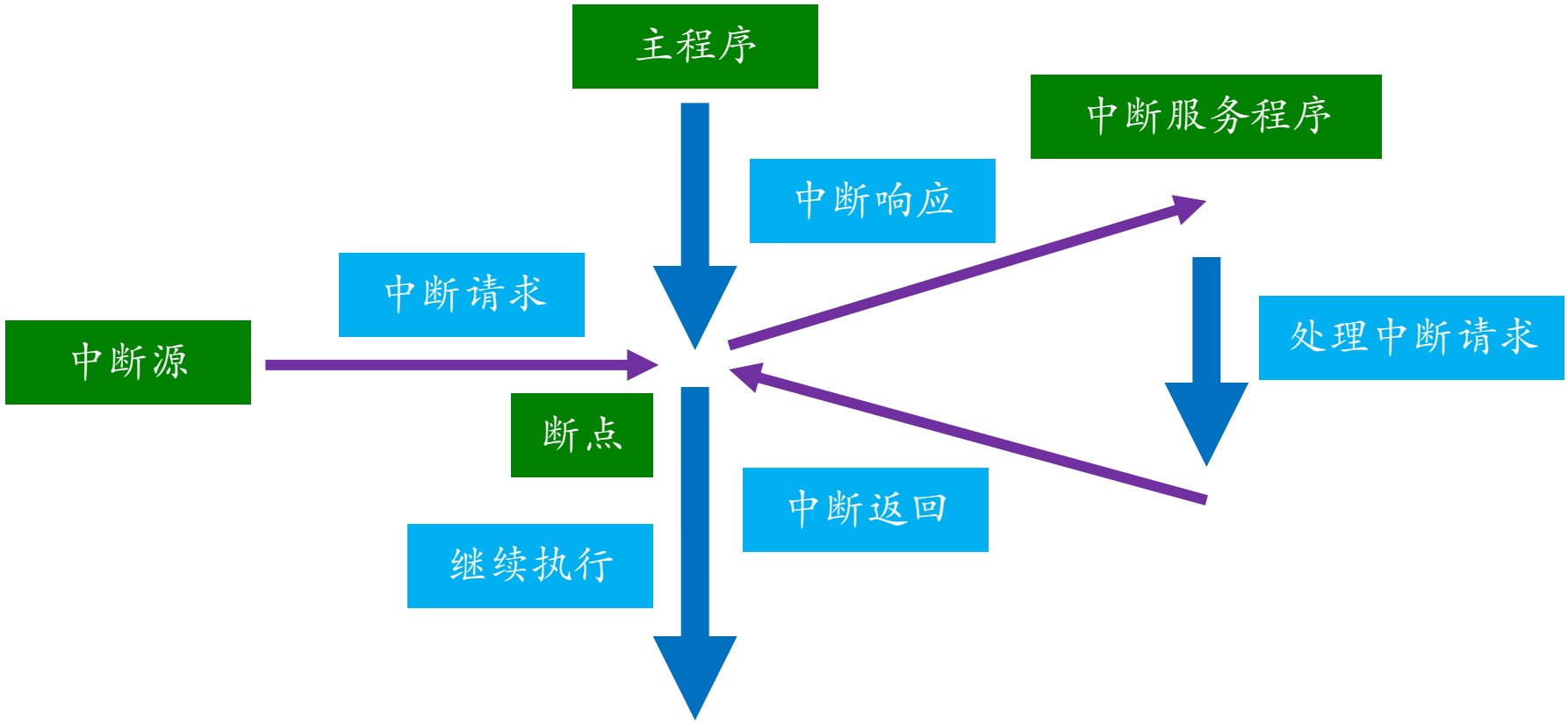
5.1 中断的基本概念

一、什么是中断?

- 1956年，美国IBM公司IBM 7049机上首先使用了中断处理技术，开始使用“中断”（Interrupt）这一术语
- 中断是作为**处理器与外部设备交换信息**的一种控制方式提出，最初的中断全部是对外部设备而言，称为**外部中断/硬件中断**
- 随着计算机技术的发展，中断的范围也随之扩大，出现了**内部软件中断**的概念，它是**为解决机器内部运行时出现的异常**以及**为编程方便**而提出
- 外部中断或硬件中断通常称为中断，软件中断或异常中断通常称为异常（Exception）

5.1 中断的基本概念

- 中断的概念
  - 中断是一个过程，指CPU在正常运行程序时，由于内部/外部事件或程序预先安排的事件，引起CPU中断正在运行的程序，而转到为中断事件服务的程序中去，服务完毕，再返回执行原程序
  - 不论哪种中断都遵循同样的中断处理过程



中断响应过程

5.1 中断的基本概念

- 中断服务程序和子程序区别
  - 中断请求取决于事件的发生时间，这个时间是随机的 → 在执行程序的过程中中断程序的插入也是随机的

5.1 中断的基本概念

- 中断服务程序和子程序区别
  1. 子程序的执行事先安排好的，而中断服务程序是随机的
  2. 子程序的执行受到主程序的控制或上级子程序的控制，而中断服务程序和主程序无关
  3. 不存在调用多个子程序的情况，但是有可能发生多个外设同时请求为自己服务的情况

5.1 中断的基本概念

- 中断的用途
  - I/O设备信息传送控制（键盘，打印机等）
  - 实时处理（数据采集系统），实时系统中现场参数变化，随时发出中断请求
  - 中断在处理一些紧急事件时特别有效，数据出错（奇偶校验错）、硬件故障（I/O错），产生不可屏蔽中断（NMI）
  - 内部事件引起，例如零除某个数
  - 程序发出中断指令（软件中断INT nH），系统资源的共享与利用，例如BIOS和DOS功能调用

5.1 中断的基本概念

二、中断源与中断识别

1. 中断源

发出中断请求的外设或引起中断的内部原因

- **外设中断**

  外部设备要求与CPU交换信息而产生的中断

- **指令中断**

  为使用系统资源或调试软件而设置的中断指令（调用I/O设备的BIOS及DOS系统功能的中断指令；设置断点中断等）

5.1 中断的基本概念

- **程序性中断**

  程序运行过程中出现错误而产生中断

  （溢出中断、被零除中断、地址越界中断、非法操作码中断、存储器空间不够中断）

- **硬件故障中断**

  机器在运行过程中，硬件出现偶然性或固定性错误而引起的中断（奇偶错中断、电源故障中断）

5.1 中断的基本概念

二、中断源与中断识别

2. 中断识别

- CPU响应中断后，只知道有中断请求，但不知道是哪一个中断源，寻找中断源的操作过程称为中断识别
- 目的是形成中断服务程序的入口地址，以便CPU将此地址置入CS:IP寄存器，从而实现程序的转移

5.1 中断的基本概念

3. CPU识别中断的方法

- 向量中断

  在CPU响应中断后，由中断控制器将服务程序入口地址送到CPU

- 查询中断

  用软件查询技术确定发出中断请求的中断源

5.1 中断的基本概念

- 查询中断方式

  通过查询各设备的中断状态标志，确定哪个设备发中断，并为之服务，具有以下特点

  （1）硬件简单

  （2）查询顺序决定优先权

  （2）查询频率决定响应速度

  （3）占用CPU较多的时间

  （4）中断嵌套不方便

- 查询中断流程


主程序断点
保存CPU状态
读A中断状态标志
A请求中断
Yes
No
清A中断标志，转A的中断服务
读B中断状态标志
B请求中断
Yes
No
清B中断标志，转B的中断服务
恢复CPU状态
中断返回
查询中断过程

5.1 中断的基本概念

- 向量中断方式

  向量中断方式是以硬件为基础，为每个中断源提供单独的中断入口标志（也称中断向量、中断矢量）

  在CPU响应中断时，由发中断申请的中断源将中断向量传递给CPU，CPU据此获得该中断源的服务入口地址（实现中断源的识别）

5.1 中断的基本概念

三、中断向量与中断向量表

- 中断向量，实际上就是中断服务程序的入口地址，每个中断类型对应一个中断向量
- 中断类型号，系统分配给每个中断源的代号，共256个，通过一个地址指针表与中断服务程序的入口地址相联，在实模式下，该表称为中断向量表（中断服务程序入口地址表）

p80/p63

5.1 中断的基本概念

- 中断向量表，包含256个中断向量，每个中断向量包含两个字（4个字节），高地址字为中断服务程序所在代码段的段基址，低地址字为代码段中中断服务程序第一条指令偏移量
- 实模式下，中断向量表存放在内存最低端的1K单元之中，物理地址00000H~003FFH

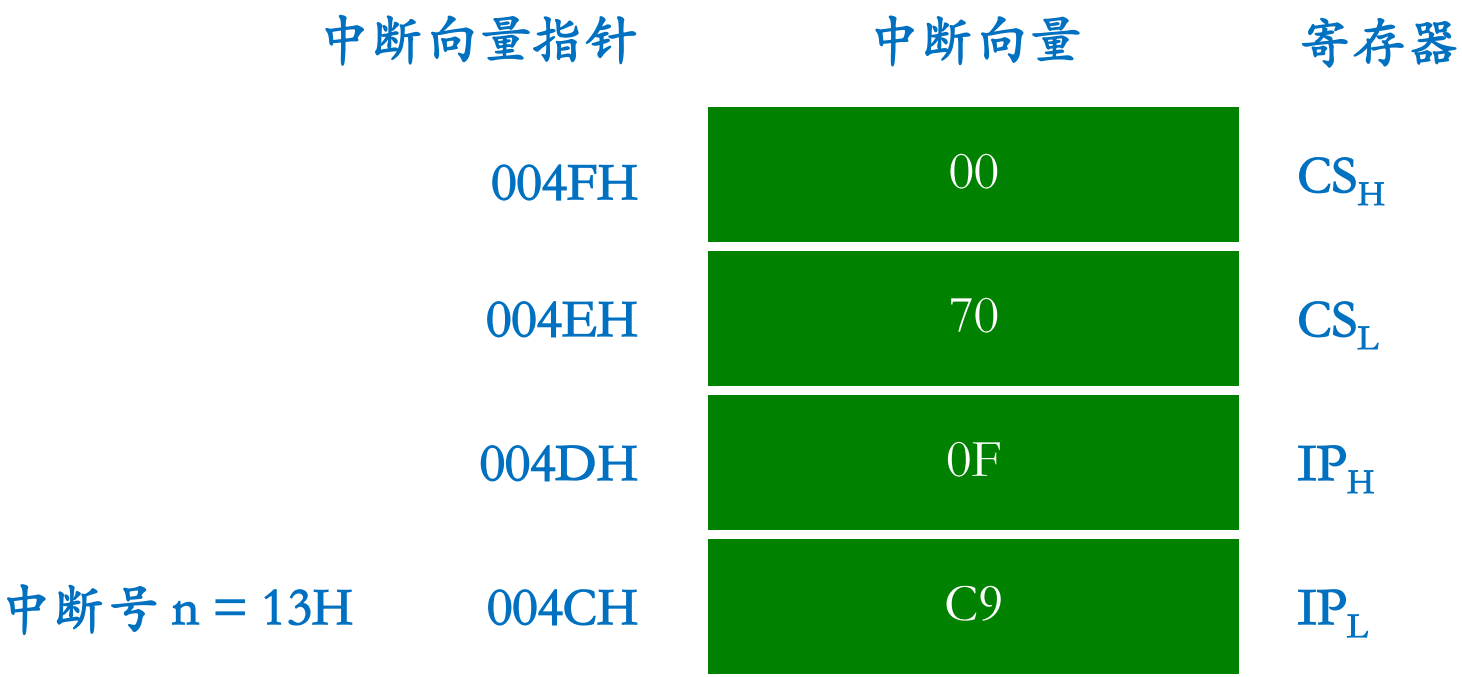
5.1 中断的基本概念

四、中断类型号与中断向量指针

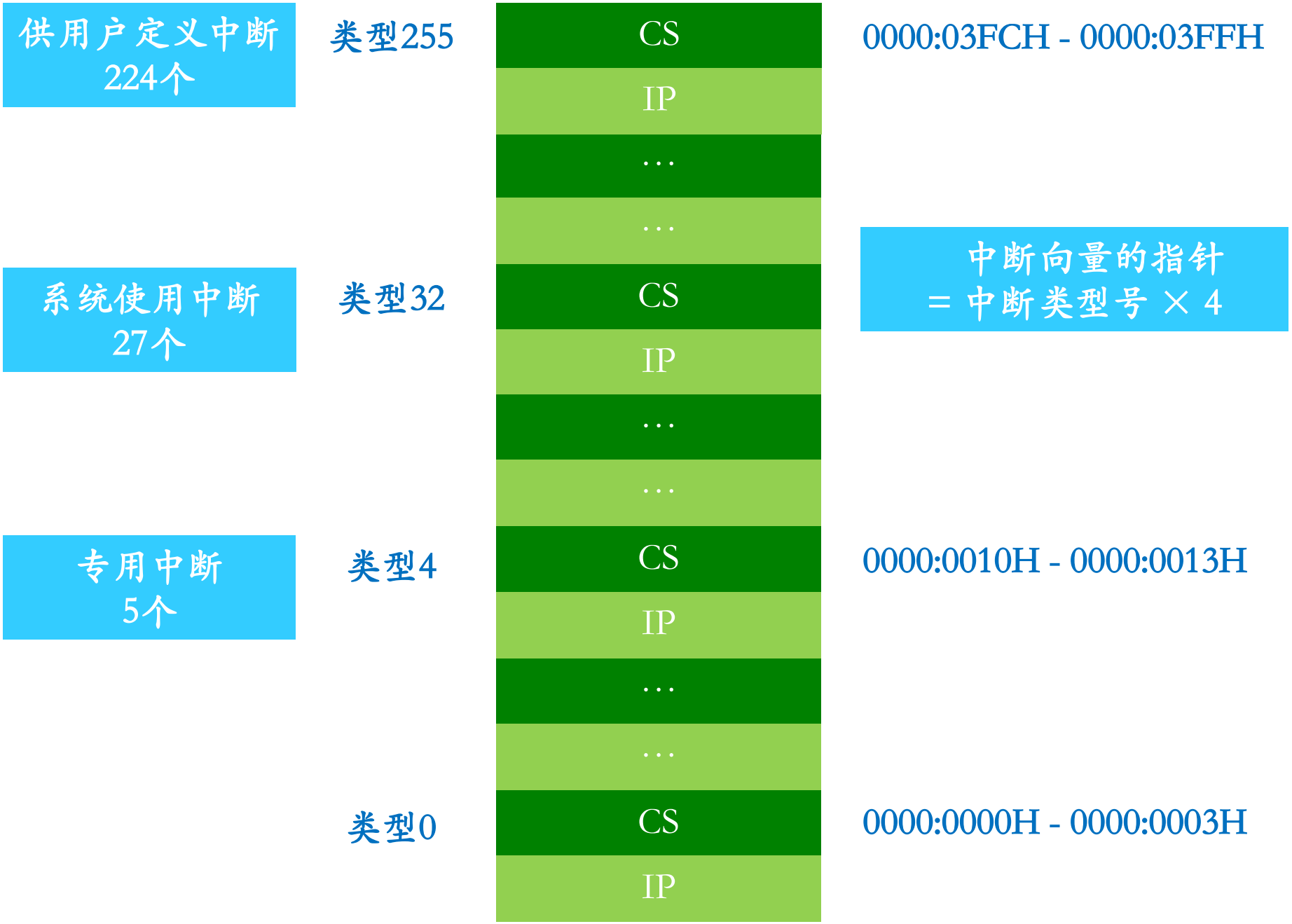
- 中断类型号
  - 统一编号（256个）
    外部中断/内部中断，硬中断/软终端，保留号
  - 在向量中断方式中，CPU通过中断号找到中断服务程序入口地址，实现程序转移
- CPU如何获取中断号
  - 可屏蔽中断 - 中断控制器8259提供
  - 指令中断INT nH - 中断指令直接给出
  - 不可屏蔽中断/特殊中断 - 系统预先设置（NMI，除零数等）

5.1 中断的基本概念

- 中断类型号与中断向量指针的关系
  - 中断类型号 n×4＝中断向量指针最低字节的指针
  - 中断号固定不变（系统分配指定）
  - 中断号对应的中断向量可以改变，便于使用系统资源
  - 系统专用中断，不允许用户随意修改中断向量


中断向量指针
中断向量
寄存器
004FH
00
CS_H
004EH
70
CS_L
004DH
0F
IP_H
中断号 n = 13H
004CH
C9
IP_L

中断向量指针


供用户定义中断
224个
类型255
CS
IP
…
…
0000:03FCH - 0000:03FFH
中断向量的指针
= 中断类型号 × 4
系统使用中断
27个
类型32
CS
IP
…
…
专用中断
5个
类型4
CS
IP
…
…
0000:0010H - 0000:0013H
类型0
CS
IP
0000:0000H - 0000:0003H

5.1 中断的基本概念

例　类型号为20H的中断所对应的中断向量存放在0000:0080H开始的4个单元中，则它所对应的中断服务程序的入口地址为：

4030:2010H。

40H	0000:0083H
30H	0000:0082H
20H	0000:0081H
10H	0000:0080H

例　一个系统中对应中断类型号17H的中断服务程序存放在2345:7890H开始的内存区域中，17H对应的中断向量存放于：

17H×4＝5CH。

23H	0000:005FH
45H	0000:005EH
78H	0000:005DH
90H	0000:005CH

5.1 中断的基本概念

五、中断向量的装入和修改

1. 中断向量的装入

一般由系统装入，但是在单片机由用户自己装入。

5.1 中断的基本概念

例1 用MOV指令填写中断向量表，类型60H　　p81/p66

```
…
CLI                     ;关中断
CLD                     ;清方向标志位
MOV AX, 0
MOV ES, AX              ;给ES赋值0
MOV DI, 4*60H           ;中断向量指针
MOV AX, OFFSET_INTR
STOSW
MOV AX, SEG_INTR
STOSW
STI
…
```

5.1 中断的基本概念

例2 将中断服务程序的入口地址直接写入中断向量 表p81/p66

…

```
MOV AX, 00H
MOV ES, AX
MOV BX, 60H*4
MOV AX, 006DH                ;偏移地址
MOV ES:[BX], AX
PUSH CS
POP AX
MOV ES:[BX+2], AX ;段地址
```

…

5.1 中断的基本概念

例3 采用DOS功能int 21h中的AH=25h装入中断向量

…

```
mov ax, N                ;使al=N
mov ah, 25h
mov dx, seg_intr
mov ds, dx
mov dx, offset_intr
int 21h
```

…

5.1 中断的基本概念

例4 采用DOS功能int 21h中的AH=35h取原中断向量

…

```
mov ah, 35h
mov al, N
int 21h
mov old_off, bx
mov bx, es
mov old_seg, bx
```

…

5.1 中断的基本概念

2. 中断向量的修改

利用DOS功能调用INT 21H的35H和25H号功能

三个步骤:

① 用35h号功能，获取原中断向量，并保存在字变量中;

② 用25号功能，设置新的中断变量，取代原中断向量，以便当中断发生后，转移到新中断服务程序中去;

③ 新中断服务程序完毕，利用25号功能，恢复原中断向量

5.1 中断的基本概念

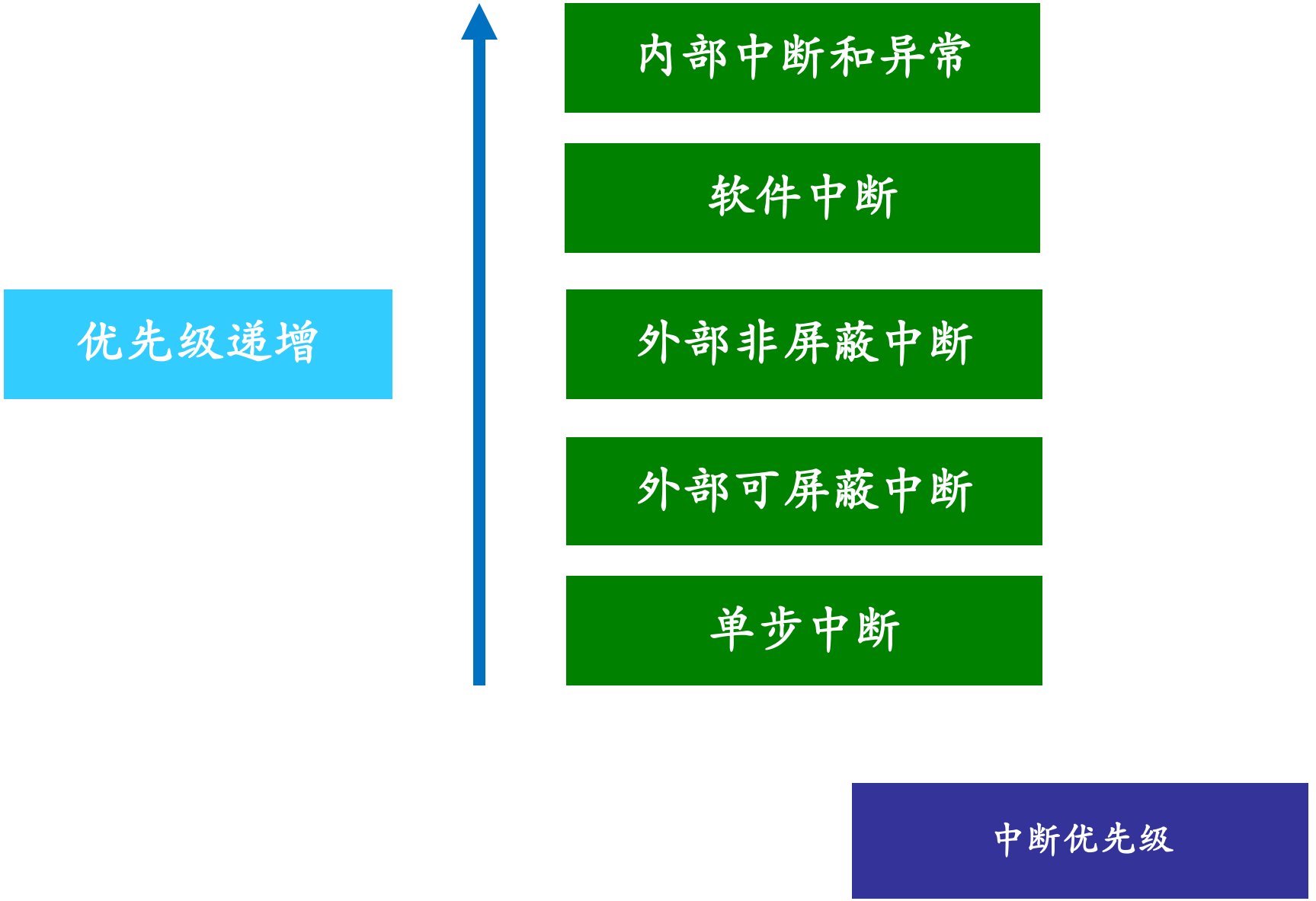
六、 中断的优先级排队方式

- 通常一个系统有多个中断源
- CPU同一时刻只能响应一个中断源的请求

当多个中断源同时请求中断服务时，按中断源的轻重缓急程度确定的优先级别，称为优先级

PC系列微机最多可以支持256种中断，排队方式有

- 优先级排队
- 循环轮流排队
- 其它排队方式


优先级递增
内部中断和异常
软件中断
外部非屏蔽中断
外部可屏蔽中断
单步中断
中断优先级

5.1 中断的基本概念

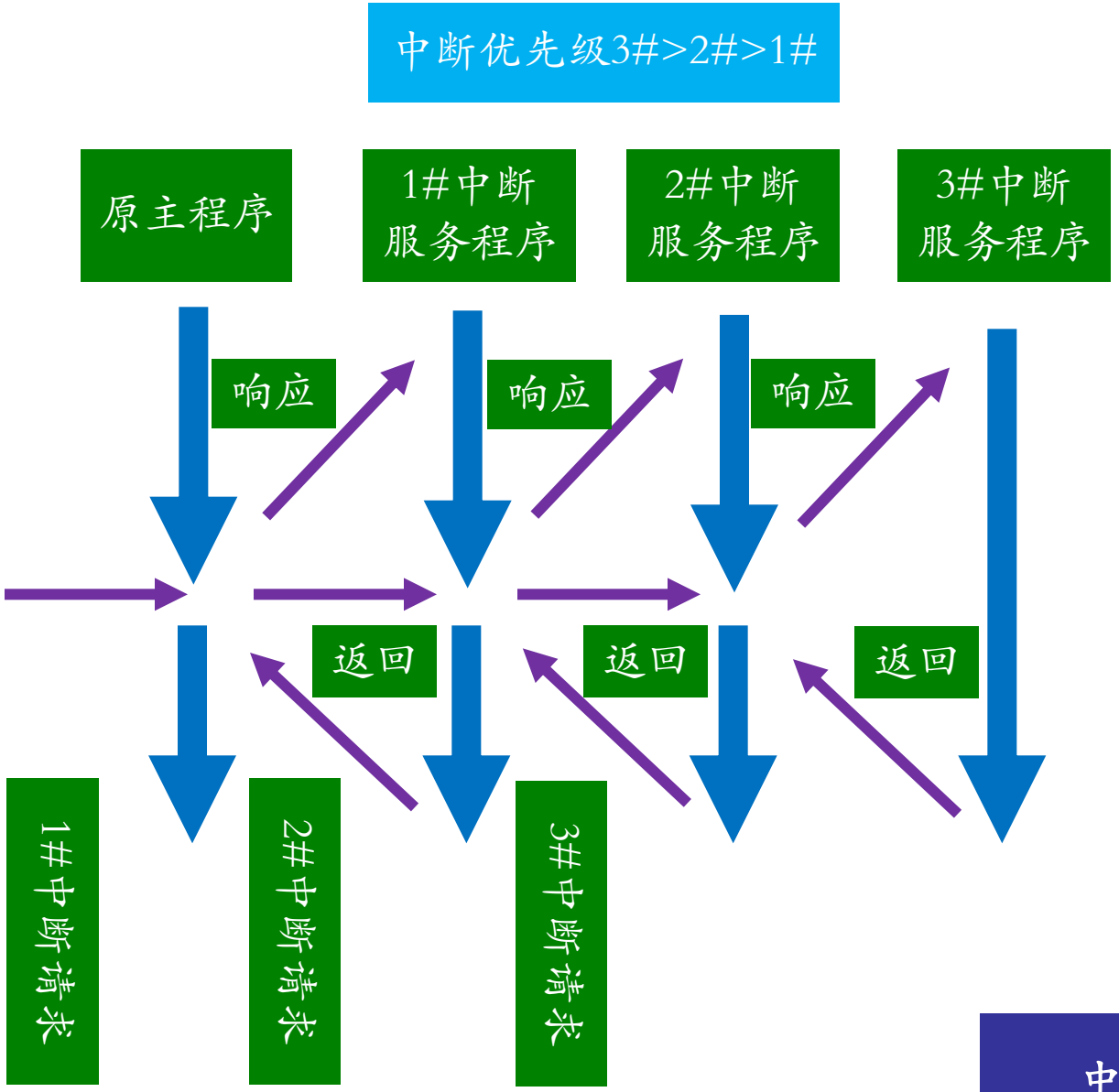
当系统具有多个中断源时，有可能同时发出请求，CPU按照重要性和急迫性（中断优先级别）择优响应，一般的处理原则：

- 不同优先级同时请求，按优先级别处理
- 低优先级中断正在处理，出现高优先级请求，转去处理高优先级请求
- 高优先级中断正在处理，出现低优先级请求，暂不响应
- 中断处理时，出现同级别请求，当前中断处理完以后再处理新的请求

5.1 中断的基本概念

七、中断嵌套

- 当CPU正在响应某一中断源的请求，执行为其服务的中断服务程序时，如果有优先级更高的中断源发出请求，CPU将中止正在执行的中断服务程序而转入新的中断源服务，等新的中断服务程序执行完后，再返回到被中止的中断服务程序，这一过程称为**中断嵌套**
- 中断嵌套可以有多级，具体级数原则上不限，只取决于堆栈深度


中断优先级3#>2#>1#
原主程序
1#中断
服务程序
2#中断
服务程序
3#中断
服务程序
响应
响应
响应
返回
返回
返回
1#中断请求
2#中断请求
3#中断请求

中断优先级与嵌套

5.1 中断的基本概念

八、中断指令

- STI（开中断指令）

  将标志寄存器FLAGS中的中断标志位IF置1，允许CPU响应来自INTR引脚的中断请求

- CLI（关中断指令）

  将标志寄存器中的中断标志位IF清0，使CPU不响应来自INTR引脚的中断请求

5.1 中断的基本概念

八、中断指令

- INT n（软件中断指令）

n为中断号，调用n号中断服务程序

① 将标志寄存器压栈

② 将TF置0，禁止单步操作，将IF置0，使CPU处于关中断状态

③ 断点的CS、IP压栈

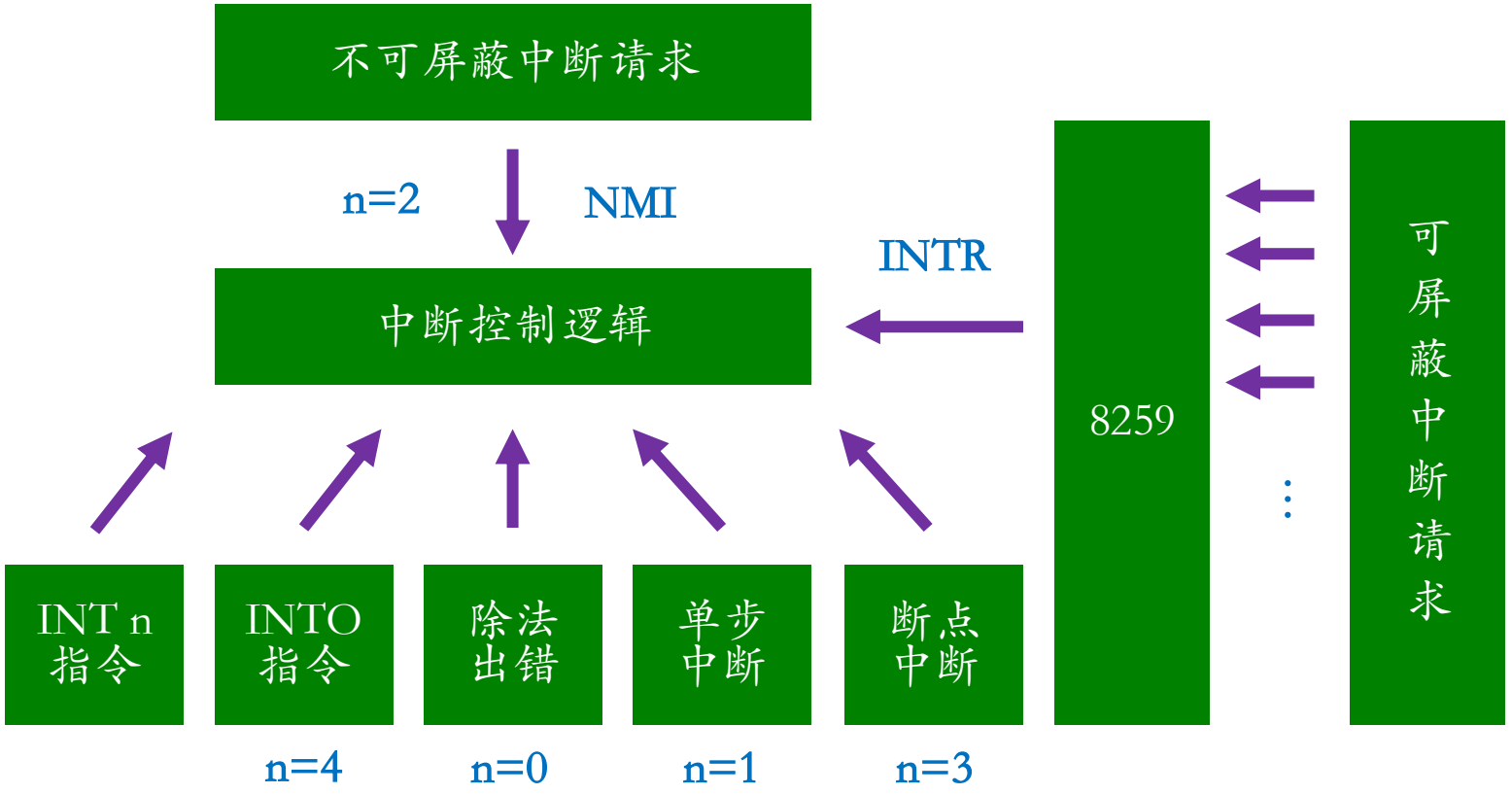
④ 从中断向量表取n号中断向量→IP、CS

⑤ 转向n号中断服务程序

5.1 中断的基本概念

八、中断指令

- IRET（中断返回指令）
  - 中断服务程序的出口指令，从栈顶弹出6个字节依次写入IP、CS和标志寄存器
  - 在执行IRET之前必须保证栈顶是断点地址，否则执行IRET指令将导致系统瘫痪


不可屏蔽中断请求
n=2
NMI
中断控制逻辑
INTR
8259
可屏蔽中断请求
INT n 指令
INTO 指令
除法出错
单步中断
断点中断
n=4
n=0
n=1
n=3

IBM-PC中断系统

5.2 IBM-PC微机中断系统

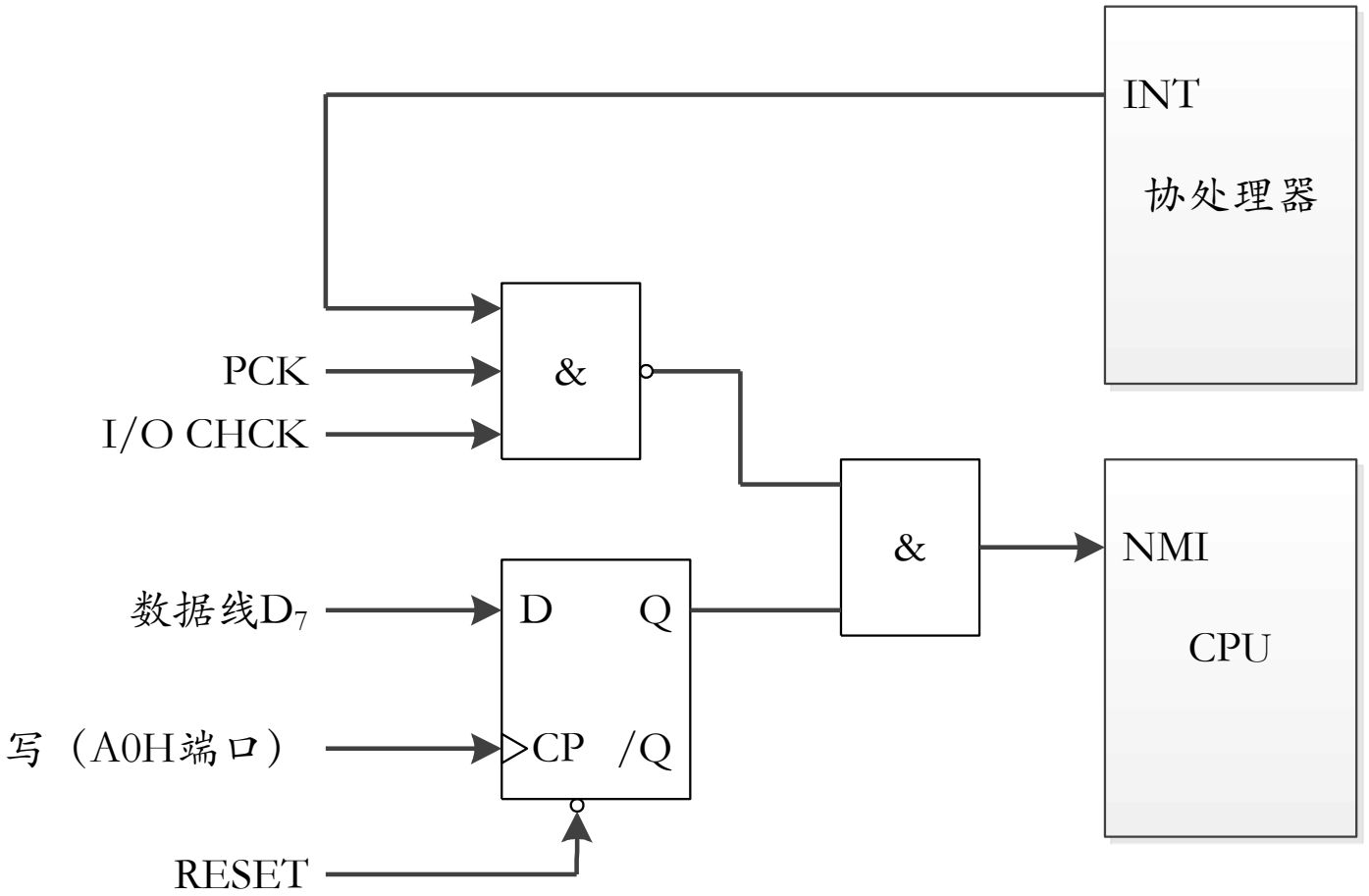
一、硬中断

1. 外部非屏蔽中断

- 为外部紧急请求提供服务的中断，通过处理器的NMI引脚产生
- NMI输入是上升沿触发的，只要NMI输入端上出现由0到1的跳变，一个中断服务请求就被锁存在Pentium中，与IF标志的状态无关。
- NMI有一个专用的类型号02H
- 使用非屏蔽中断的典型例子是电源故障、RAM奇偶校验错、I/O通道校验错、协处理器出错

系统硬中断表

中断号	IRQ	标准应用
02H	NMI	奇偶校验错、I/O检测错中断
08H	0	定时器OUT_0中断
09H	1	键盘输入中断
0AH	2	接收从片8259A的中断请求
70H	8	实时钟中断
71H	9	改向INT 0AH（以IRQ_2出现）
72H	10	保留
73H	11	保留
74H	12	保留
75H	13	协处理器中断
76H	14	硬磁盘控制器中断
77H	15	保留
0BH	3	串行通信（COM_1）中断
0CH	4	串行通信（COM_2）中断
0DH	5	打印机（LPT_2）中断
0EH	6	软磁盘控制器中断
0FH	7	打印机（LPT_1）中断



外部控制NMI信号产生

5.2 IBM-PC微机中断系统

2. 外部可屏蔽中断

- 外部可屏蔽中断是处理器响应各种外部硬件中断的最常用的方法，通过CPU的INTR引脚产生，外部可屏蔽中断受处理器内部的中断允许标志位IF的控制
- 处理器以电平触发方式接受INTR请求，当每条指令结束时，若INTR为高电平且IF=1，则CPU响应相应I/O接口的中断请求

5.2 IBM-PC微机中断系统

- 处理器只有一个INTR引脚可以接受外部可屏蔽中断请求，为了管理众多的外部中断源，微机系统中采用可编程中断控制器8259，PC系列机通过两片8259级连可以响应15个外部中断源
- 可屏蔽中断INTR由外部设备产生，中断向量表中08H~0FH和070H~077H

【说明】

- 两片可以扩充及兼容性的问题 p84

5.2 IBM-PC微机中断系统

二、软中断

- 软中断的中断号是在中断指令中直接给出的
- CPU不发出中断响应信号，也不要求中断控制器提供中断号

1. BIOS和DOS中断
2. 一些特殊的中断

   0号中断（除数为0中断）

   1号中断（单步中断）

   3号中断（断点中断）

   4号中断（溢出中断）

5.2 IBM-PC微机中断系统

三、中断的处理过程

1. 中断申请
2. 中断响应
3. 中断服务程序
4. 中断返回

5.2 IBM-PC微机中断系统

1. 中断申请

中断源向CPU发出的中断请求信号

- 硬件中断源通过专门的电路传送给CPU；CPU有专门的引脚接收中断请求信号

  8086/8088CPU用INTR引脚和NMI引脚接收硬件中断请求信号

- 对于软件中断源，在CPU内部由中断指令或程序出错直接引发中断

5.2 IBM-PC微机中断系统

- 中断允许标志

  CPU的标志寄存器中的IF标志，表示是否可以响应外设的中断请求，通常用1来表示允许；

  在8086/8088系统中可以用CLI和STI指令来设置IF，禁止或允许来自INTR可屏蔽中断请求引脚的请求

- 中断请求标志

  8259A对应每个外设有一位，用来记录外设的中断请求状态，有请求时置1，中断处理完后清0

  在PC机中，用8259A来管理外设的中断请求；在8051系列的单片机中，该标志在CPU内部

5.2 IBM-PC微机中断系统

- 中断屏蔽

  有些硬件中断源的请求可以根据IF标志决定是否响应，而有些硬件中断源的请求需要一定被响应

  在8086/8088系统中，INTR是可屏蔽中断请求引脚；NMI是不可屏蔽的中断请求引脚

  在PC机系统中，通过8259A管理的外设中断源连在CPU的INTR引脚上；

  在8259A内部有8位中断请求寄存器和8位的中断屏蔽寄存器，可以对应于8个外设的中断申请和中断屏蔽

5.2 IBM-PC微机中断系统

2. 中断响应

- CPU在每条指令执行的最后一个时钟周期检测中断请求，如果以下之一，CPU自动进入中断响应周期。
  （1）有软件中断　（2）有NMI中断信号
  （3）有INTR中断信号，且CPU允许响应中断
- 进入中断响应周期以后
  如果是INTR，则产生INTA信号给中断源，让中断源通过数据总线的低8位送出中断类型码给CPU；
  如果是NMI，类型码是2；
  中断指令中有类型码；
  程序出错有默认类型码
- 标志寄存器进栈，清除TF和IF标志，返回地址的CS和IP进栈

5.2 IBM-PC微机中断系统

- 形成中断服务程序入口地址，转入中断服务程序执行
每个中断的有自己的处理程序，各种微处理器形成中断服务程序入口地址的方法也不相同

8086/8088系列

中断服务程序的入口地址称为中断向量，将中断向量送入CS:IP即可转入中断处理程序；

在内存的000H~3FFH的1KB空间中存有256个中断中断向量对应于256种中断类型码，称为中断向量表；

在表中，按类型码×4即可得到中断向量的位置，取出4个字节的中断向量

8051系列

CPU知道中断地址，不同中断直接对应于各固定位置的指令

5.2 IBM-PC微机中断系统

3. 中断服务程序

进入中断服务程序的步骤

- 保护现场，开中断
- 执行中断服务程序
- 关中断，恢复现场
- 开中断，返回主程序

5.2 IBM-PC微机中断系统

4. 中断返回

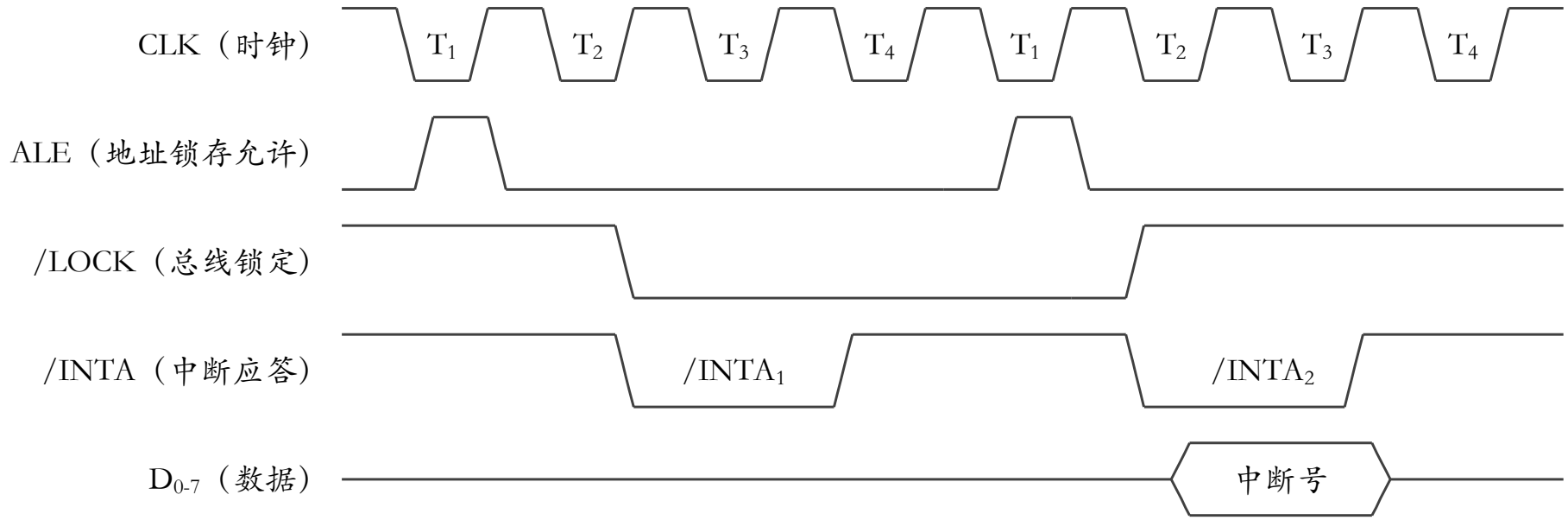
- 断点处的指令是中断处理结束后，返回时恢复执行的第一条指令的地址称为**返回地址**

【注意】

- 从栈顶弹出6个字节依次写入IP、CS和标志寄存器（注意顺序）
- 在执行IRET之前必须保证栈顶是断点地址，否则执行IRET指令将导致系统瘫痪

5.2 IBM-PC微机中断系统

- 中断响应周期及$INTA_2$信号的作用


CLK（时钟）
T1
T2
T3
T4
T1
T2
T3
T4
ALE（地址锁存允许）
/LOCK（总线锁定）
/INTA（中断应答）
/INTA1
/INTA2
D0-7（数据）
中断号

中断响应周期时序

5.3 可编程中断控制器8259A

一、8259A中断功能

- 优先级排队

  一片Intel 8259可管理8个中断请求，并把当前优先级最高的中断请求送到CPU的INTR端；

  8个外部中断的优先级排列方式：通过对8259编程进行指定；也可以通过编程屏蔽某些中断请求；或者通过编程改变中断类型码

- 接受和扩充外部设备的中断请求

  允许8片8259级联，构成64级中断系统

  在PC/AT系列微机中，使用两片8259级联，构成15级中断

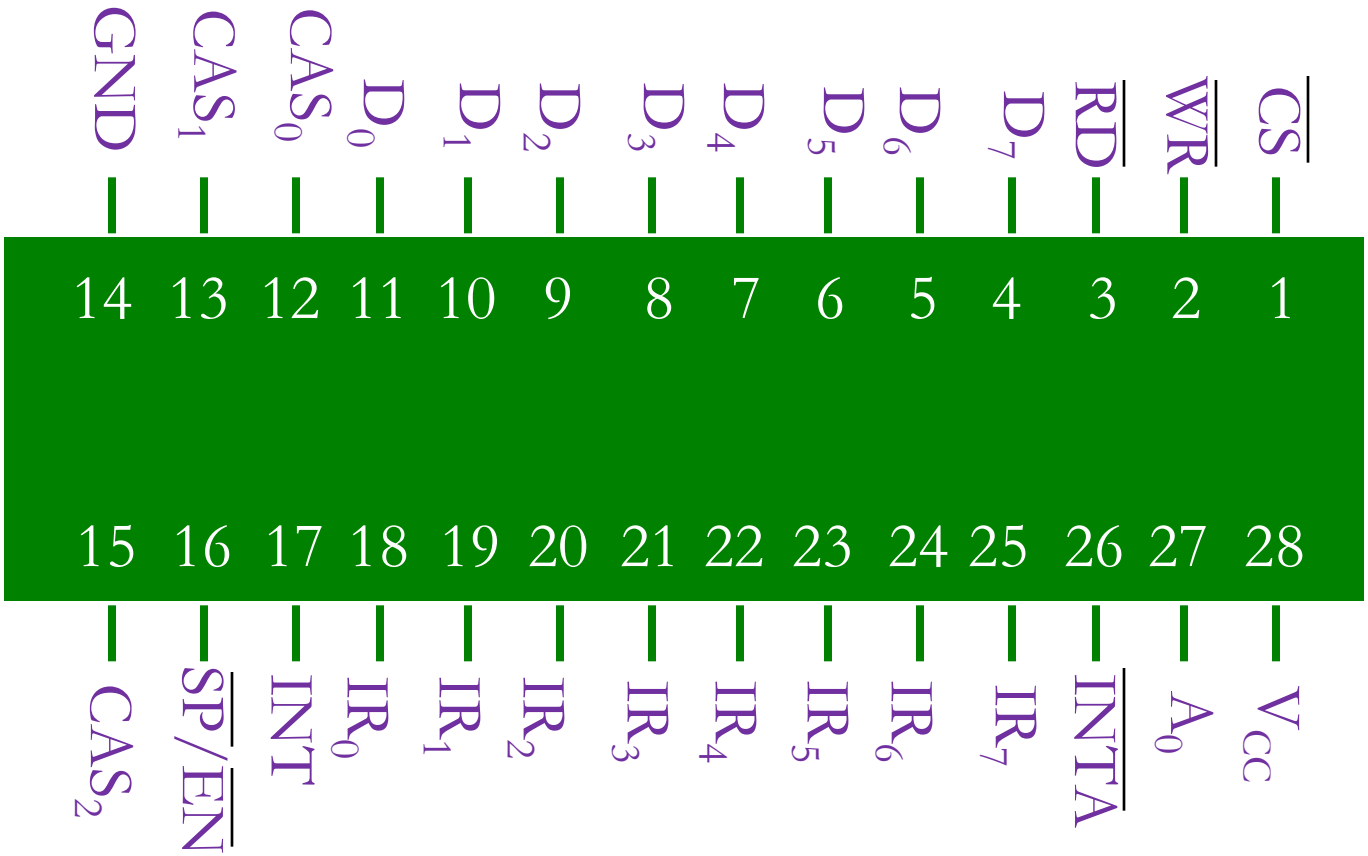
5.3 可编程中断控制器8259A

- 提供中断类型号
  当CPU响应中断时，为CPU提供中断类型码
- 进行中断请求屏蔽和开放

可用于8086/8088和8080/8085CPU

5.3 可编程中断控制器8259A

二、8259A外部特性和内部结构


GND
CAS1
CAS0
D0
D1
D2
D3
D4
D5
D6
D7
RD
WR
CS
14 13 12 11 10 9 8 7 6 5 4 3 2 1
15 16 17 18 19 20 21 22 23 24 25 26 27 28
CAS2
SP/EN
INT
IR0
IR1
IR2
IR3
IR4
IR5
IR6
IR7
INTA
A0
VCC

8259外部引脚

5.3 可编程中断控制器8259A

- D_7~D_0　　数据总线，双向，三态，用于与CPU之间传送命令、状态、中断类型码
- /RD　读信号，输入，用来通知8259把某个内部寄存器的值送数据线D_7~D_0
- /WR　写信号，输入，用来通知8259把数据线D_7~D_0上的值写入内部某个寄存器
- /CS　片选信号，输入，通过地址译码逻辑电路与地址总线相连
- A_0　　地址线，输入，用来指出当前8259的哪个端口被访问，选择内部寄存器的端口地址

  在标准AT机中，使用两片8259构成主从式中断系统，主8259的端口地址： 20H，21H，从8259的端口地址：A0H，A1H

5.3 可编程中断控制器8259A

- INT　中断请求，输出，把IR7~IR0上的最高优先级请求传送到CPU的INTR引脚，向CPU发中断请求
- INTA 中断响应，接收CPU的中断应答信号，CPU发出的中断响应信号为两个负脉冲，第一个负脉冲作为中断应答信号，第二个负脉冲到来时，8259从数据线D_7~D_0上发出中断类型码
- IR_7~IR_0　外设中断请求输入，在含有多片8259的复杂系统中，主片的IR_7~IR_0分别与从片的INT端相连，用来接收来自从片的中断请求
- CAS_2~CAS_0　级联线，用来指示主片到从片
- SP#/EN#　从设备编程/缓冲器允许，双向，输入时，用来决定8259是主片还是从片（1-主片，0-从片，非缓冲方式）；输出时，使数据总线驱动器启动（缓冲方式）

5.3 可编程中断控制器8259A

8259A的内部结构

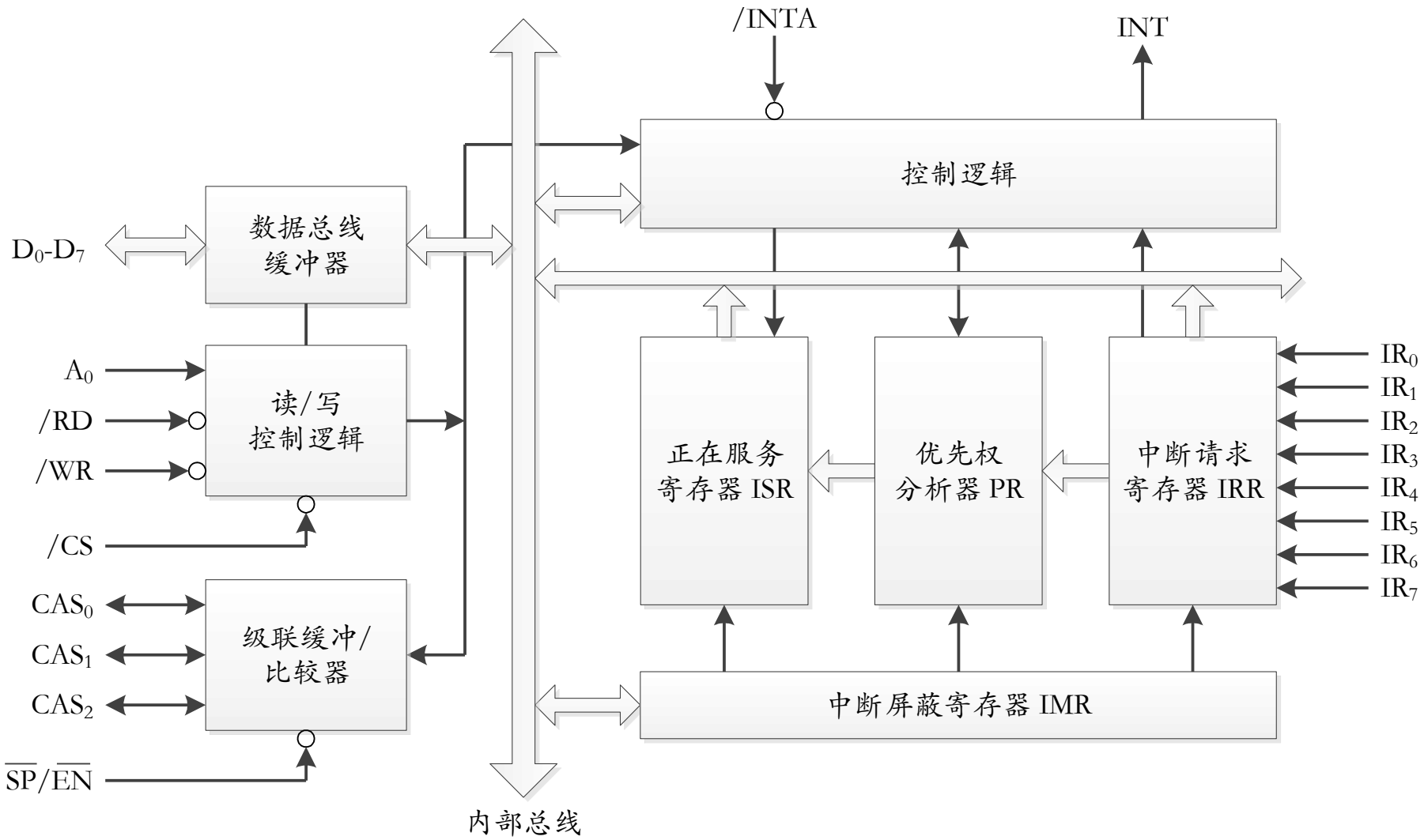
- 数据总线缓冲器

  三态，双向8位缓冲器，D_7~D_0用于和CPU的数据总线相连，CPU通过数据总线缓冲器向8259传送命令码，或从8259读取状态字和中断类型号

- 中断请求寄存器（IRR）

  用来保存正在请求服务的中断源发出的中断请求信号IR，共8级IR_7~IR_0

  若某一位输入线有请求，该位就置1，具有锁存功能，其内容可用OCW_3命令读出



8259A内部逻辑框图

5.3 可编程中断控制器8259A

- 正在服务寄存器（ISR）

  中断服务状态寄存器用于保存已被CPU响应并处理的中断信号IR

  包括尚未服务完而中途被别的中断所打断的中断级，其内容可以用OCW_3命令读出

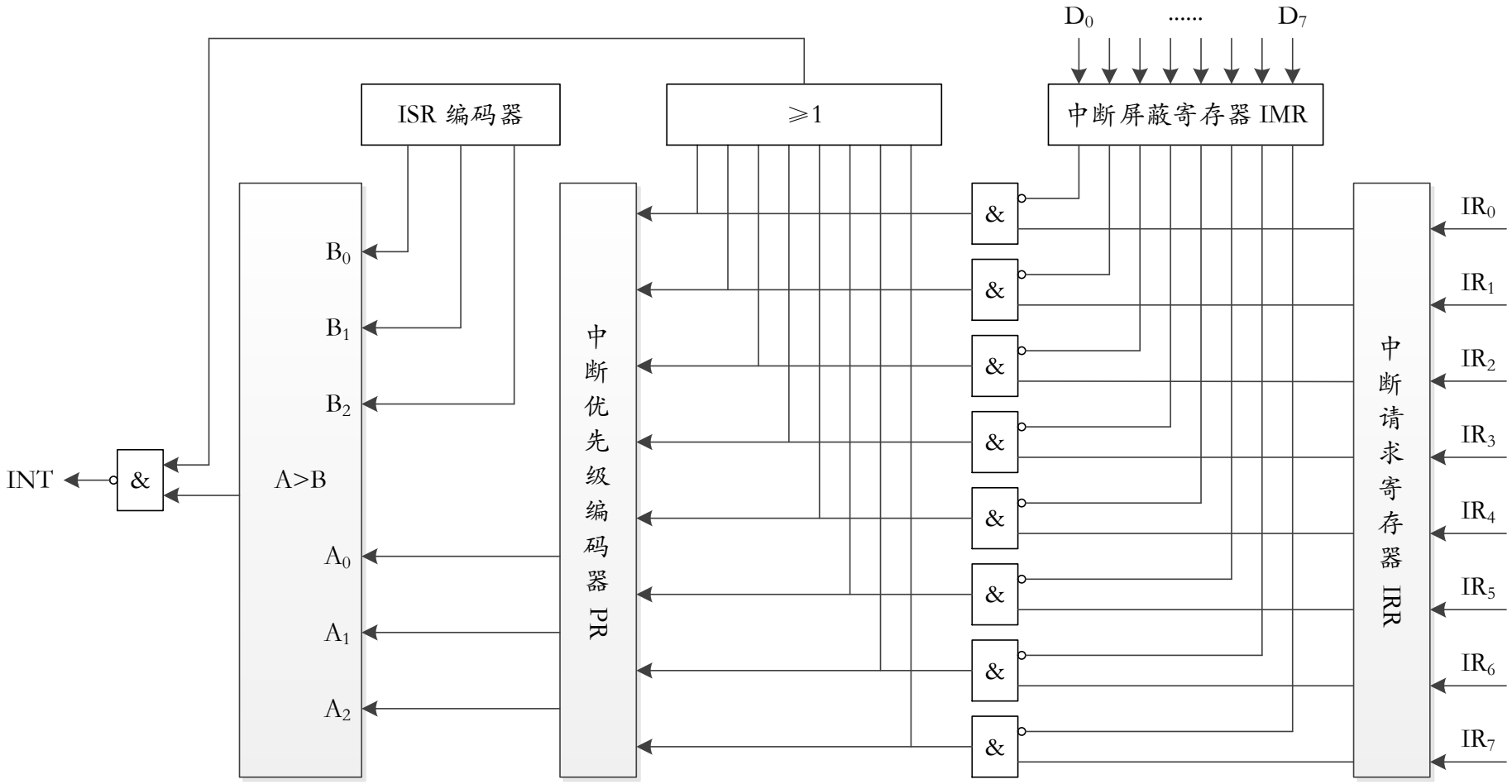
- 中断屏蔽寄存器（IMR）

  对IRR起屏蔽作用，中断屏蔽寄存器则通过屏蔽某级中断信号对应位来改变中断服务的优先顺序，IMR存放屏蔽位信息，这些屏蔽位禁止IRR中相应位的中断请求进入服务

5.3 可编程中断控制器8259A

- 优先权分辨器（PR）

  优先权分辨器确定发出中断请求信号的优先级别，当中断请求寄存器中有中断信号时，优先权分辨器选出其中最高级别者，然后和正在服务的中断比较，若比后者高，则使INT变成高电平，向CPU发出中断请求


D0
.....
D7
ISR 编码器
≥1
中断屏蔽寄存器 IMR
B0
B1
B2
A>B
A0
A1
A2
INT
&
中断优先级编码器 PR
中断请求寄存器 IRR
IR0
IR1
IR2
IR3
IR4
IR5
IR6
IR7

中断优先级分析器

5.3 可编程中断控制器8259A

- 控制逻辑和读写逻辑
  - 控制逻辑 - 根据8259的工作方式产生内部控制信号，产生外部输出信号，在适当时刻向CPU发出中断请求信号INT，请求CPU响应；

    当控制逻辑收到中断响应信号$INTA_2$时，将中断号送到数据总线
  - 读/写逻辑 - 为8259与CPU通过数据总线缓冲器传送数据信息，提供了译码寻址信号

8259A的读写操作I/O端口地址

/CS	/WR	/RD	A_0	读写操作	PIC（主）	PIC（从）
0	0	1	0	写ICW_1，OCW_2，OCW_3	20H	0A0H
0	0	1	1	写$ICW_{2\text{-}4}$，OCW_1	21H	0A1H
0	1	0	0	读IRR，ISR，查询字	20H	0A0H
0	1	0	1	读IMR	21H	0A1H

5.3 可编程中断控制器8259A

- 级联缓冲/比较器
  - 用于构成8259的主/从级联控制结构；
  - 多个8259级联时，用级联缓冲器/比较器，寄存、比较各级8259芯片标志；
  - CAS_2~CAS_0在8259作为主设备时为输出端，作为从设备时为输入端

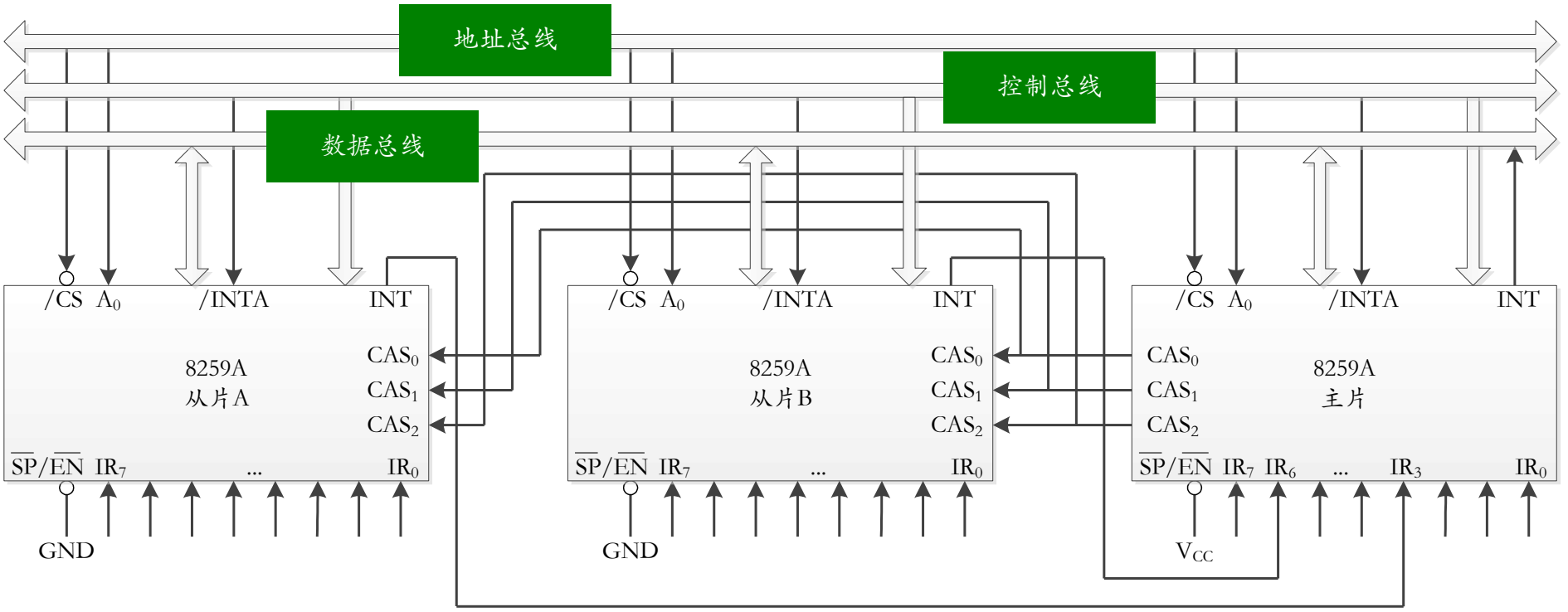
5.3 可编程中断控制器8259A

三、8259的工作方式

- 引入中断请求的方式
  - 边沿触发
  - 电平触发
  - 中断查询
- 连接系统总线的方式
  - 缓冲方式
  - 非缓冲方式

5.3 可编程中断控制器8259A

- 屏蔽中断源的方式
  - 通用屏蔽方式
  - 特殊屏蔽方式
- 优先级排队方式
  - 全嵌套方式
  - 特殊嵌套方式
  - 优先级自动轮换方式
  - 优先级指定轮换方式
- 结束中断的处理方式
  - 自动中断结束方式
  - 非自动中断结束方式


地址总线
控制总线
数据总线
/CS
A0
/INTA
INT
8259A
从片A
CAS0
CAS1
CAS2
SP/EN
IR7
...
IR0
GND
/CS
A0
/INTA
INT
8259A
从片B
CAS0
CAS1
CAS2
SP/EN
IR7
...
IR0
GND
/CS
A0
/INTA
INT
CAS0
CAS1
CAS2
8259A
主片
SP/EN
IR7
IR6
...
IR3
IR0
VCC

8259A主从级联

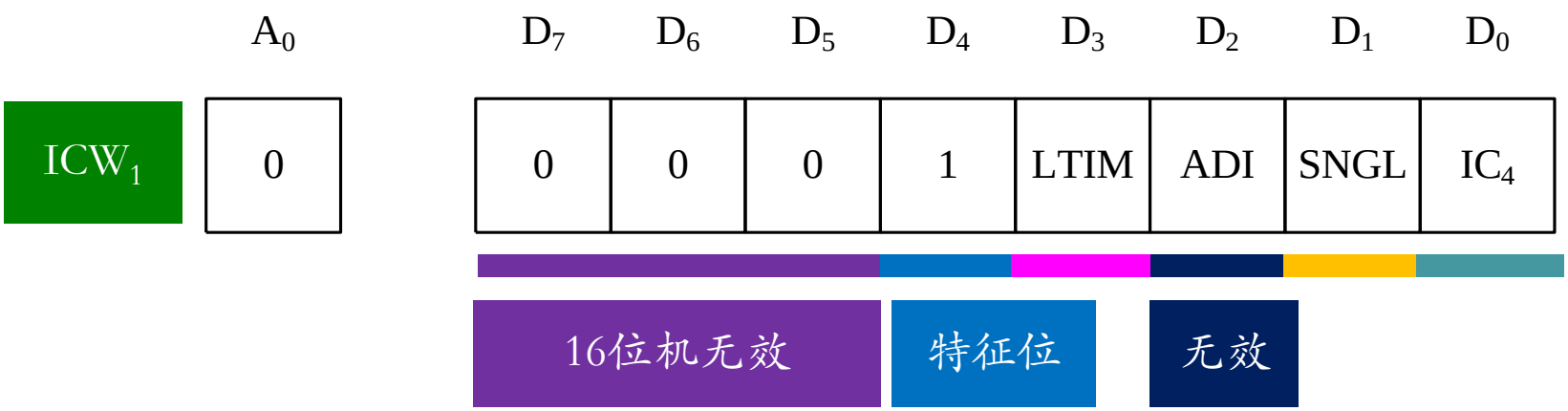
5.3 可编程中断控制器8259A

四、8259A的中断操作功能及其命令

- 8259A的工作状态和操作方式，由CPU命令指定
- 命令有两种
  - 设置工作方式：初始化命令字 ICW_1~ICW_4
  - 控制操作：操作命令字OCW_1~OCW_3
- 每片8259A有2个片内地址A_0=0和A_0=1，所有的命令都是通过这两个端口来实现

5.3 可编程中断控制器8259A

- 8259的初始化命令字（预置命令字ICW）
  - ICW_1~ICW_4在初始化程序中设定，且在整个工作过程中保持不变；
  - ICW_1~ICW_4必须按顺序设定；
  - ICW_1写入8259偶地址中（A_0=0，在AT机中为20H/A0H）；
  - ICW_2~ICW_4写入8259奇地址中（A_0=1，在AT机中为21H/A1H）



A_0=0 & D_4=1表示初始化编程开始

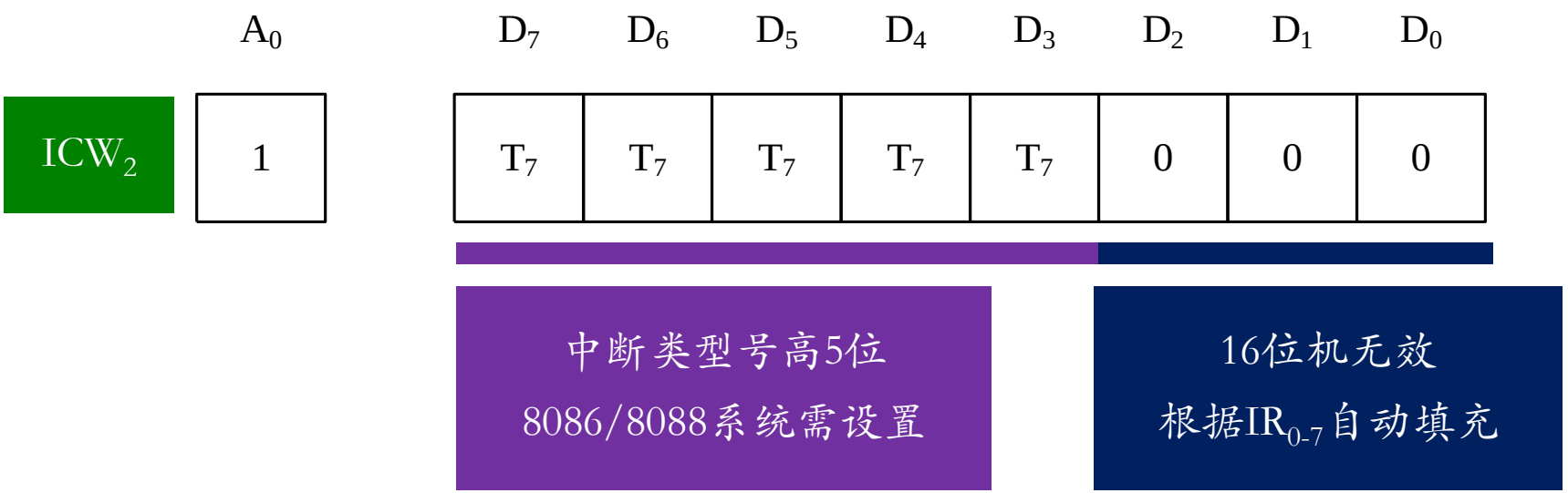
中断信号的触发方式 0边沿/1高电平

是否单片方式 0多片级联/1单片

是否有ICW_4（16位以上微机）0无/1有

8086/8088系统$D_{5\text{-}7}$和D_2不用/通常置0
D_0置1

触发方式/级联


A0
D7 D6 D5 D4 D3 D2 D1 D0
ICW2
1
T7 T7 T7 T7 T7 0 0 0
中断类型号高5位
8086/8088系统需设置
16位机无效
根据IR0-7自动填充

中断类型号

5.3 可编程中断控制器8259A

例　在PC/XT中ICW_2为00001000B

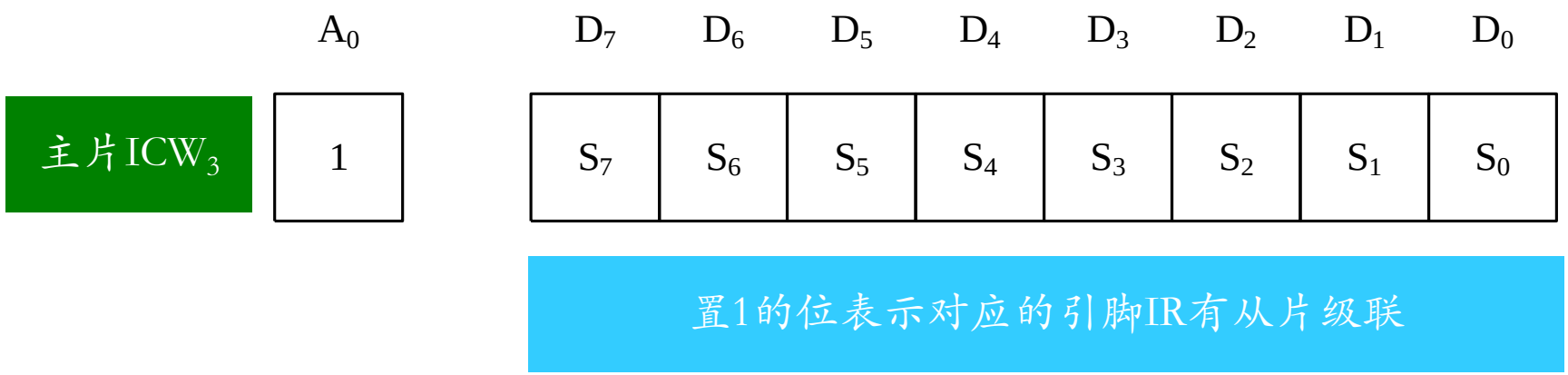
中断号：类型号		中断号：类型号	
IR0：08H	时钟中断	IR4：0CH	COM1
IR1：09H	键盘中断	IR5：0DH	硬盘
IR2：0AH	保留	IR6：0EH	软盘
IR3：0BH	COM2	IR7：0FH	LPT1

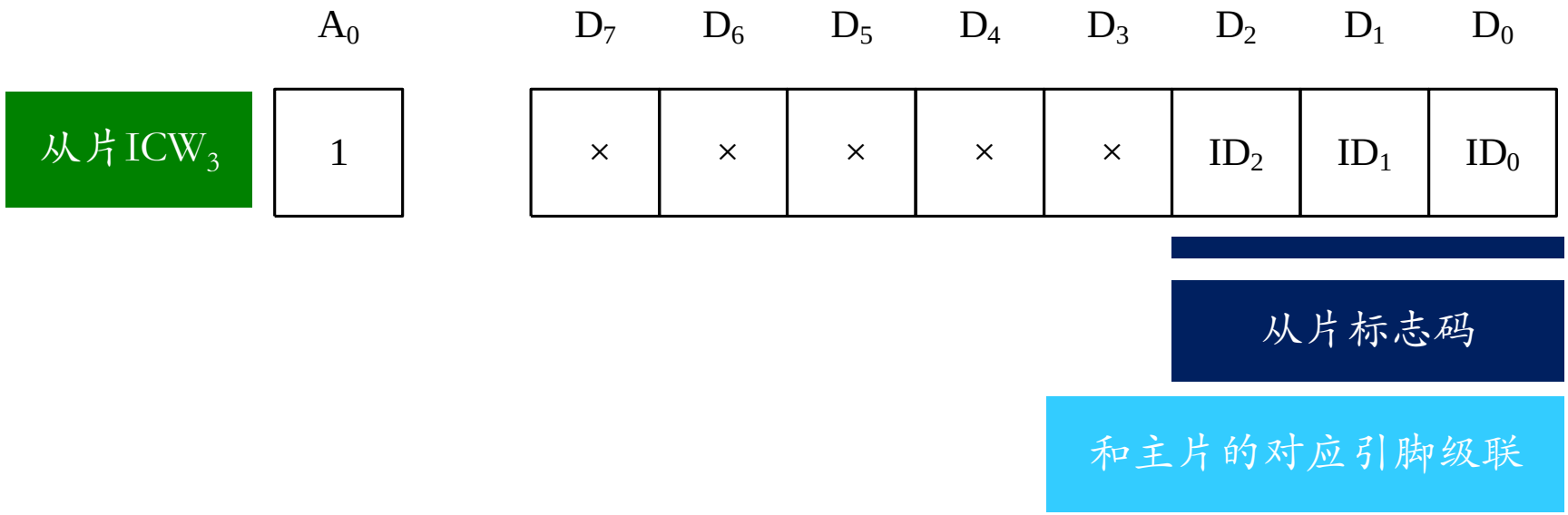
主片8259A 8级硬中断源的中断号

中断源	中断号高5位	低3位	中断号
日时钟	08H	0/IR_0	08H
键盘	08H	1/IR_1	09H
保留	08H	2/IR_2	0AH
通信（二）	08H	3/IR_3	0BH
通信（一）	08H	4/IR_4	0CH
硬盘	08H	5/IR_5	0DH
软盘	08H	6/IR_6	0EH
打印机	08H	7/IR_7	0FH

从片8259A扩充8级硬中断源的中断号

中断源	中断号高5位	低3位	中断号
实时钟	070H	0/IR_0	70H
改向INT 0A	070H	1/IR_1	71H
保留	070H	2/IR_2	72H
保留	070H	3/IR_3	73H
协处理器	070H	4/IR_4	74H
保留	070H	5/IR_5	75H
硬盘	070H	6/IR_6	76H
保留	070H	7/IR_7	77H

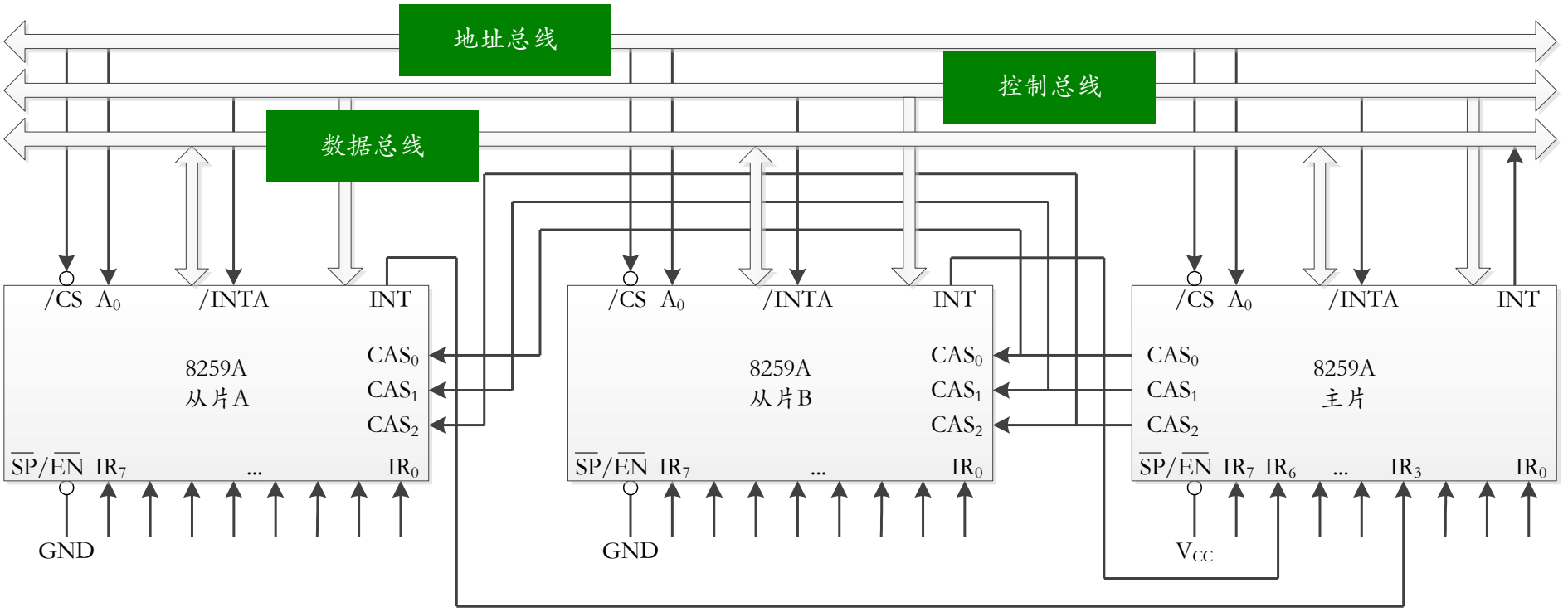

A0
D7 D6 D5 D4 D3 D2 D1 D0
主片ICW3
1
S7 S6 S5 S4 S3 S2 S1 S0
置1的位表示对应的引脚IR有从片级联


A0
D7 D6 D5 D4 D3 D2 D1 D0
从片ICW3
1
× × × × × ID2 ID1 ID0
从片标志码
和主片的对应引脚级联

级联结构

5.3 可编程中断控制器8259A

- 级联方式下从片的申请和响应
  - 中断申请
  - 中断响应
  - 中断结束


地址总线
控制总线
数据总线
/CS
A0
/INTA
INT
8259A
从片A
CAS0
CAS1
CAS2
SP/EN
IR7
...
IR0
GND
/CS
A0
/INTA
INT
8259A
从片B
CAS0
CAS1
CAS2
SP/EN
IR7
...
IR0
GND
/CS
A0
/INTA
INT
CAS0
CAS1
CAS2
8259A
主片
SP/EN
IR7
IR6
...
IR3
IR0
VCC

8259A主从级联

	A_0	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
ICW_4	1	0	0	0	SFNM	BUF	$M/\overline{S}$	AEOI	μPM

16位机无效　主片嵌套　0从片/1主片（BUF=1）

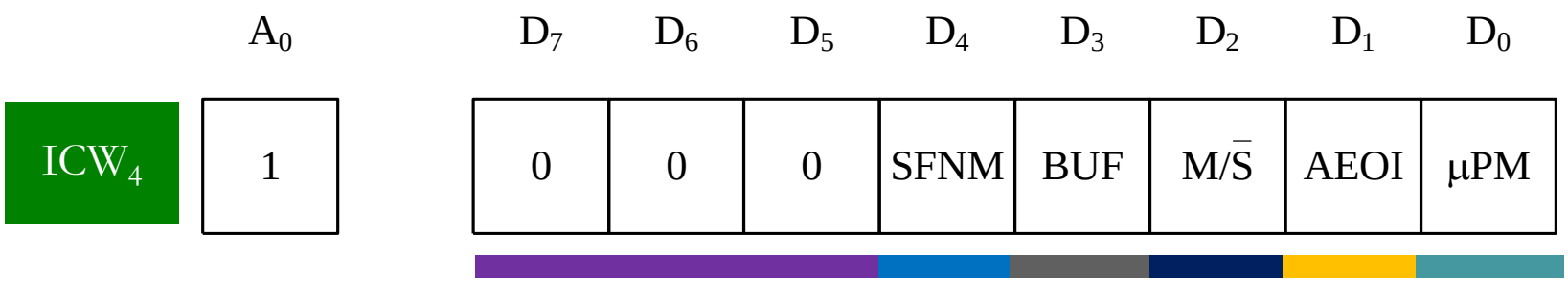
ICW1.IC4=1时写ICW4

0 一般嵌套（优先级IR_0>IR_7）/1 特殊的全嵌套

1 8259通过数据缓冲器和总线相连
SP#/EN#引脚输出 控制缓冲器的数据传送方向（接收/发送）

0 无缓冲 SP#/EN#引脚输入 作主/从片选择端

嵌套/缓冲/主从

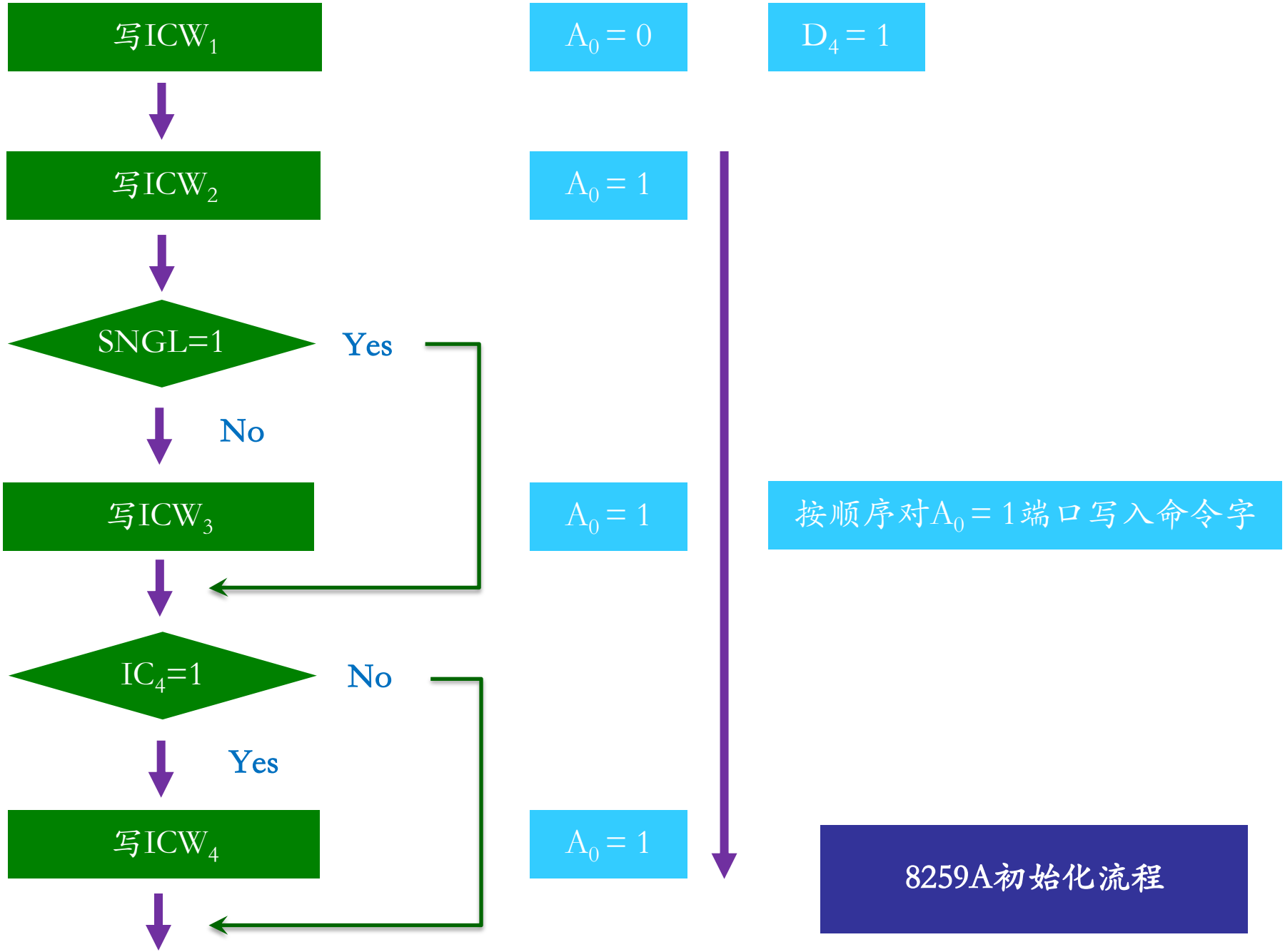


自动结束中断方式（由OCW_2实现）
0 不自动清除ISR/1 CPU响应中断后自动清除ISR

0 8位微机8080 8085 Z80/1 16位以上微机8086 8088

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

自动结束/处理器


写ICW1
A0 = 0
D4 = 1
写ICW2
A0 = 1
SNGL=1
Yes
No
写ICW3
A0 = 1
按顺序对A0 = 1端口写入命令字
IC4=1
No
Yes
写ICW4
A0 = 1
8259A初始化流程

5.3 可编程中断控制器8259A

例1 PC/XT机中8259A的端口地址是20H、21H，初始化程序

```
MOV AL, 13H    ;ICW1：00010011
OUT 20H, AL    ;单片、上升沿触发、使用ICW4
MOV AL, 8      ;ICW2：00001000
OUT 21H, AL    ;中断类型码是08H~0FH
MOV AL, 1      ;ICW4：000000001
OUT 20H, AL    ;一般全嵌套，非缓冲，
               ;非AEOI方式，16位以上微机
```

5.3 可编程中断控制器8259A

例2　PC/AT中8259A的定义

- 主片定义为：
  上升沿触发、有ICW_4
  中断类型码为08H~0FH
  在IR_2级联从片
  一般的中断嵌套方式、非AEOI方式
- 从片定义为：
  上升沿触发、有ICW_4
  中断类型码为70H~77H
  级联到主片的IR_2
  一般的中断嵌套方式、非AEOI方式

5.3 可编程中断控制器8259A

- 初始化主片

  PC/AT机中8259A主片的端口地址是20H、21H

```
MOV AL, 11H
OUT 20H, AL     ;ICW1
MOV AL, 08H
OUT 21H, AL     ;ICW2
MOV AL, 04H
OUT 21H, AL     ;ICW3
MOV AL, 01H
OUT 21H, AL     ;ICW4
```

- 初始化从片

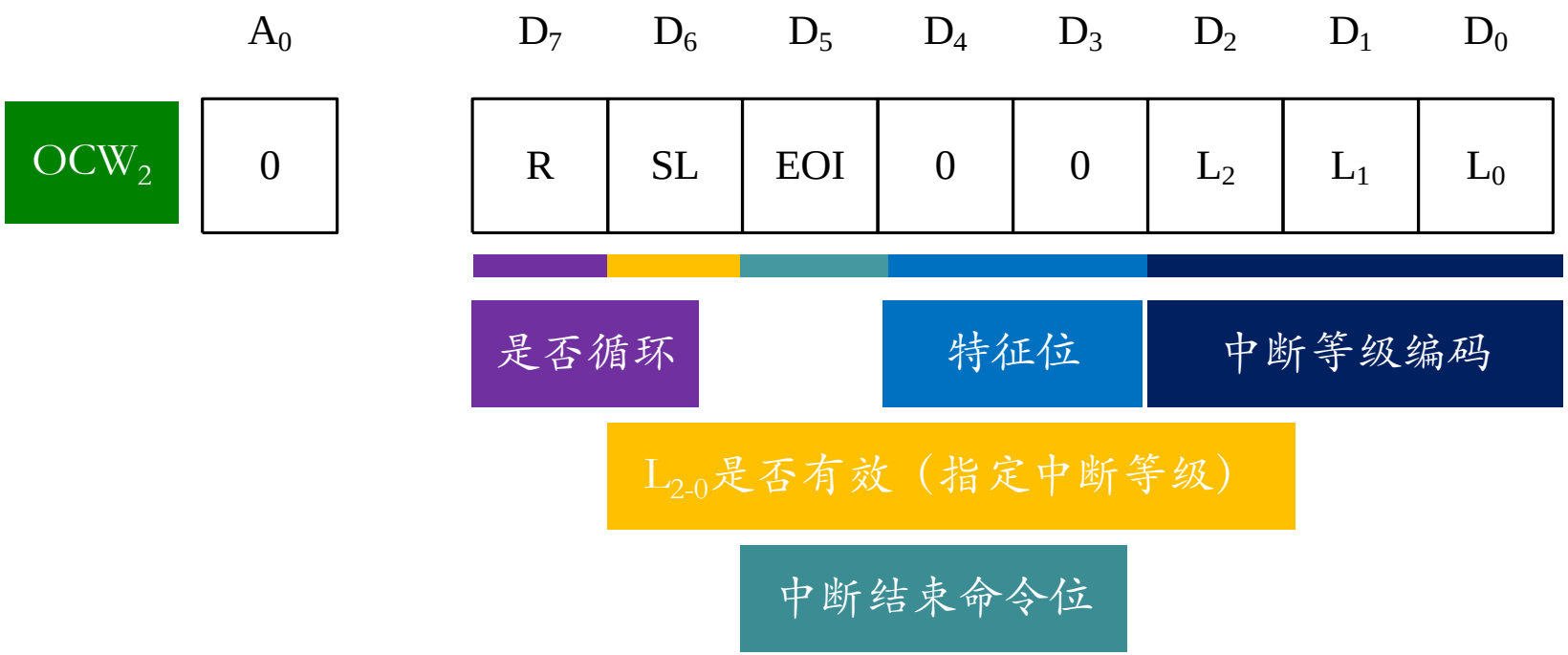
  PC/AT机中8259A从片的端口地址是A0H、A1H

```
MOV AL, 11H
OUT 0A0H, AL    ;ICW1
MOV AL, 70H
OUT 0A1H, AL    ;ICW2
MOV AL, 02H
OUT 0A1H, AL    ;ICW3
MOV AL, 01H
OUT 0A1H, AL    ;ICW4
```

	A_0	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
OCW_1	1	M_7	M_6	M_5	M_4	M_3	M_2	M_1	M_0

M_x=1屏蔽中断源IR_x
设置和清除中断屏蔽寄存器

中断屏蔽寄存器



PC机中常用EOI命令

```
MOV AL, 20H    ;00100000
OUT 20H, AL
```

中断排队方式

5.3 可编程中断控制器8259A

- OCW_2作用
  - 作中断结束控制，在初始化$ICW_4.D_1$时选用非自动结束方式，就利用OCW_2来控制中断结束

    分为两种方式：不指定的中断结束（默认最高优先级）/指定中断结束方式
  - 作中断优先级排队控制

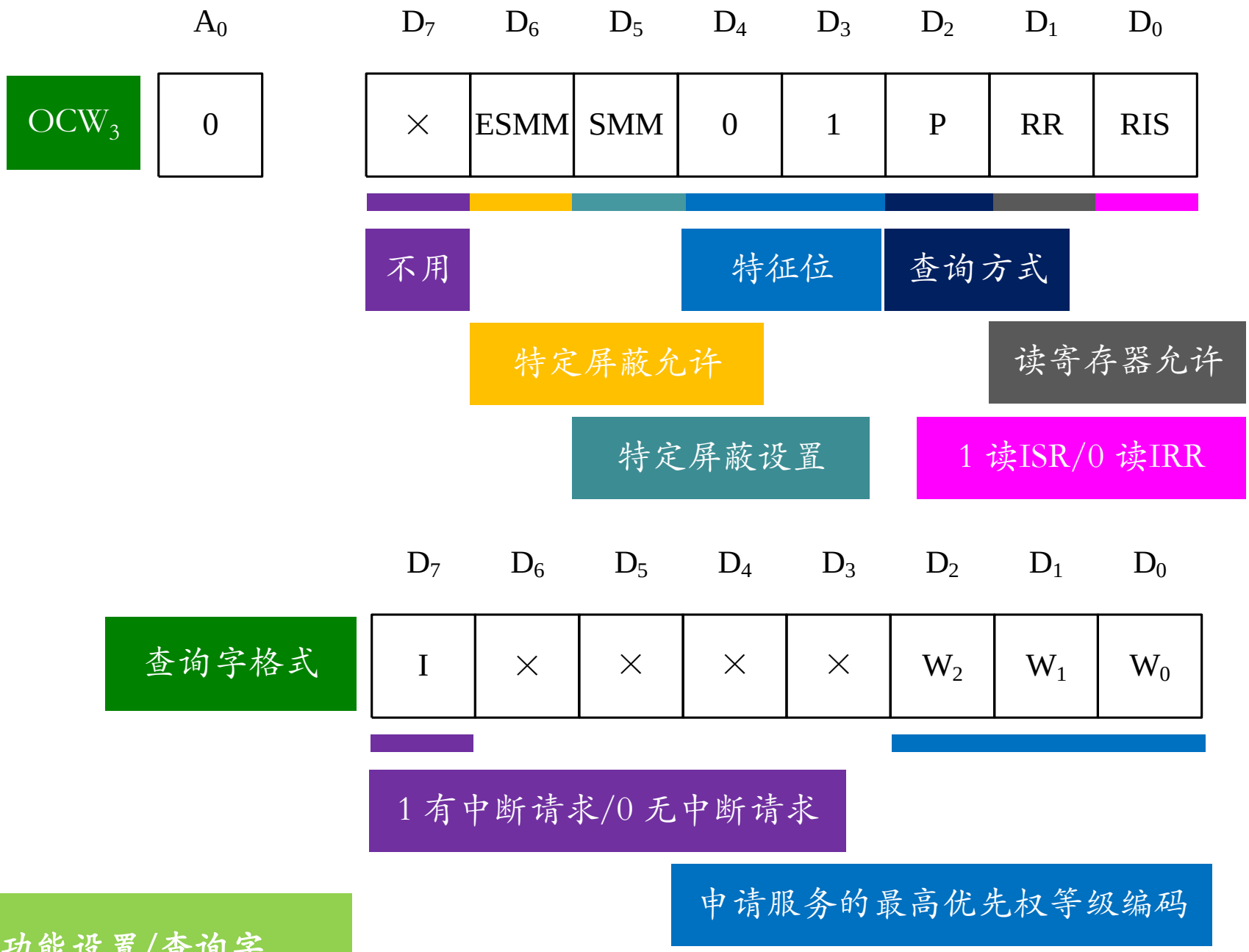
    优先权自动轮换/指定轮换

OCW_2的功能

R	SL	EOI	0	0	L_2	L_1	L_0	功 能
0	0	1	0	0	0	0	0	不指定EOI方式命令
0	1	1	0	0	L_2	L_1	L_0	指定EOI方式命令
1	0	1	0	0	0	0	0	在不指定EOI方式中轮换命令
1	0	0	0	0	0	0	0	在自动EOI方式中轮换置位命令
0	0	0	0	0	0	0	0	在自动EOI方式中轮换复位命令
1	1	1	0	0	L_2	L_1	L_0	在指定EOI方式中轮换命令
1	1	0	0	0	L_2	L_1	L_0	直接置优先级轮换命令

5.3 可编程中断控制器8259A

- OCW_3的功能
  - 设置和撤销特殊屏蔽方式
  - 设置中断查询方式
  - 设置对8259A内部寄存器的读出


A0
D7 D6 D5 D4 D3 D2 D1 D0
OCW3
0
× ESMM SMM 0 1 P RR RIS
不用
特征位
查询方式
特定屏蔽允许
读寄存器允许
特定屏蔽设置
1 读ISR/0 读IRR
D7 D6 D5 D4 D3 D2 D1 D0
查询字格式
I × × × × W2 W1 W0
1 有中断请求/0 无中断请求
申请服务的最高优先权等级编码
功能设置/查询字

5.3 可编程中断控制器8259A

例　PC机中常用命令

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；

读最高级别的中断请求IR - 先向20H端口写0CH，再读20H端口；

读出IMR寄存器 - 不需要事先发指定命令，直接读奇地址端口

8259A的读写操作I/O端口地址

/CS	/WR	/RD	A_0	读写操作	PIC（主）	PIC（从）
0	0	1	0	写ICW_1，OCW_2，OCW_3	20H	0A0H
0	0	1	1	写$ICW_{2\text{-}4}$，OCW_1	21H	0A1H
0	1	0	0	读IRR，ISR，查询字	20H	0A0H
0	1	0	1	读IMR	21H	0A1H

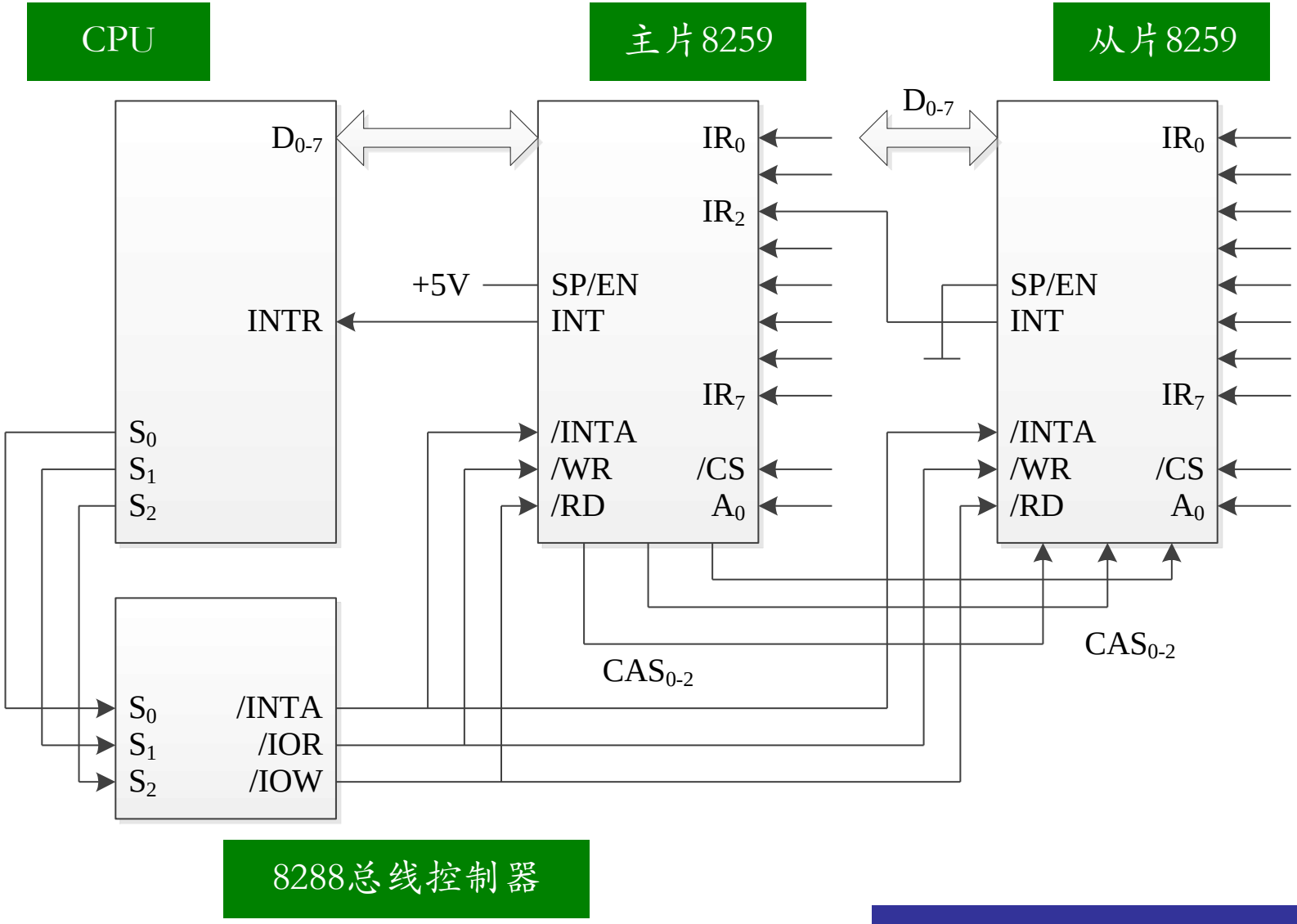
5.4 8259A在微机中的应用

- 单片使用8259A初始化编程

```
INTA00 EQU 020H
INTA01 EQU 021H
MOV AL, 13H          ;ICW1: 边沿触发/单片/需要ICW4
OUT INTA00, AL
MOV AL, 8            ;ICW2: 中断类型号高5位
OUT INTA01, AL
MOV AL, 1            ;ICW4: 全嵌套，16位微机/非自动结束
OUT INTA01, AL
```

5.4 8259A在微机中的应用

- PC/AT中8259的使用
  - 2片8259级联，提供15级向量中断，从片的INT接主片的IR2；
  - 端口地址：主片20H、21H，从片A0H、A1H；
  - 主片和从片均采用边沿触发；
  - 采用全嵌套优先级排列方式；
  - 采用非缓冲方式，主片SP/EN接+5V，从片SP/EN接地；
  - 主片的类型码为08H~0FH，从片的类型码为70H~77H


CPU
主片8259
从片8259
D0-7
INTR
S0
S1
S2
+5V
SP/EN
INT
IR0
IR2
IR7
/INTA
/WR
/RD
/CS
A0
CAS0-2
/IOR
/IOW
8288总线控制器
PC/AT中8259A连线图

5.4 8259A在微机中的应用

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
```

5.4 8259A在微机中的应用

主片8259A

```
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
```

5.4 8259A在微机中的应用

从片8259A

```
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
```

5.4 8259A在微机中的应用

- 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；

5.4 8259A在微机中的应用

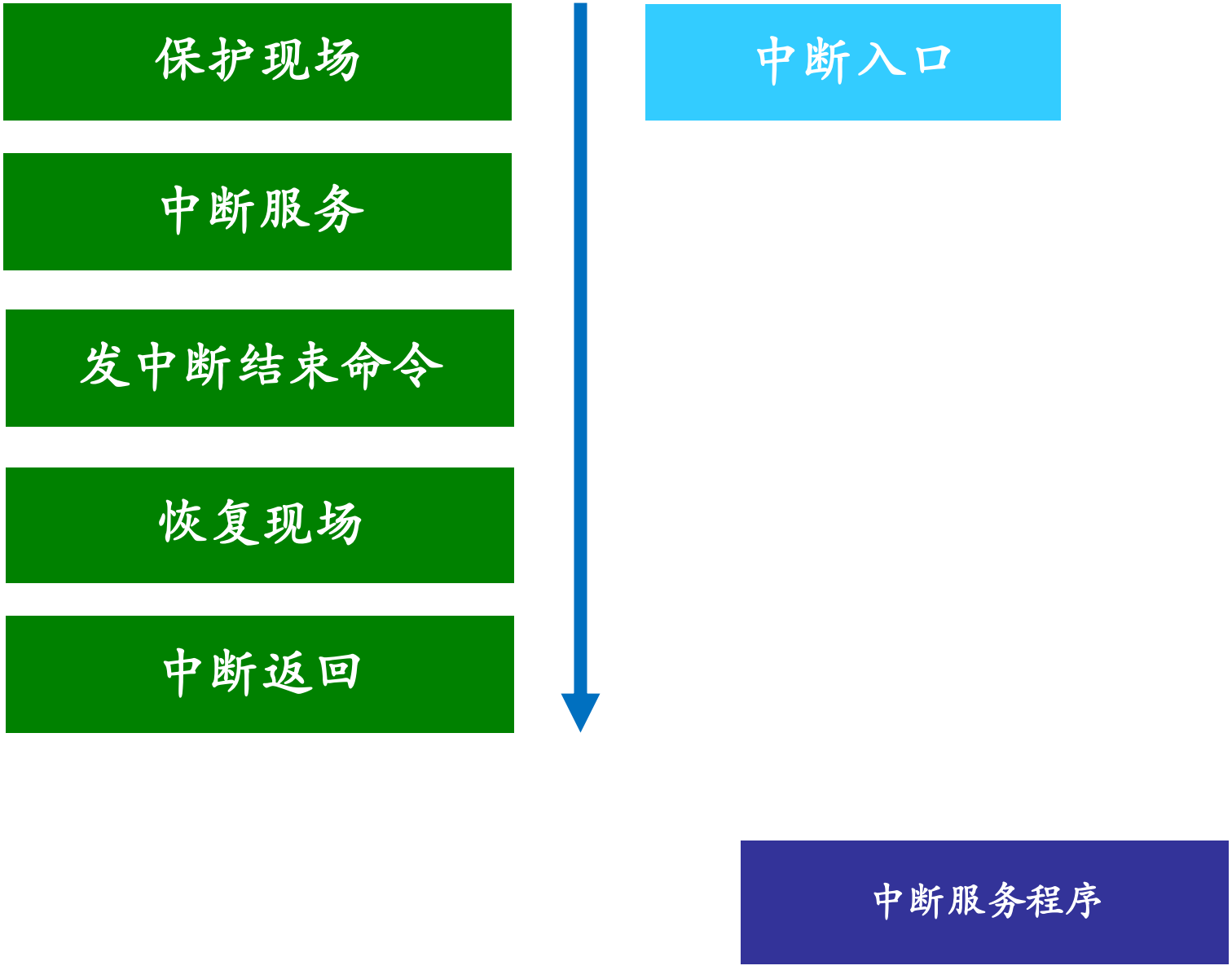
5. PC机中8259采用非自动结束中断，在中断服务程序执行IRET返回前应向8259发结束中断命令EOI

```
MOV AL, 20H          ;使当前ISR中的对应位复位
                     ;OCW2:00100000
OUT 20H, AL
IRET
```

6. 当编写替代系统原有中断服务程序时，应保存好原中断向量的内容，在应用程序终止前恢复原中断向量；
7. 中断服务程序不要使用DOS系统功能调用（INT 21H），因为DOS不允许重入；

5.4 8259A在微机中的应用

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


保护现场
中断服务
发中断结束命令
恢复现场
中断返回
中断入口
中断服务程序

5.4 8259A在微机中的应用

- 中断向量表的操作
  1. 读写中段向量表不使用MOV指令，而应调用相应的DOS功能。
  2. 取中断向量

     预置AH=35H, AL=中断类型号

     执行INT 21H

     把类型号为AL的中断向量取出到ES:BX中
  3. 设置中断向量

     预置AH=25H, AL=中断类型号，DS:DX=中断向量

     执行INT 21H

     把DS:DX指向的中断向量放置到中断向量表中类型号为AL的中断向量处

5.4 8259A在微机中的应用

例　填写中断向量表，实现用户所定义的60H中断

```
…...
CLI
PUSH DS
MOV AX, SEG INT60
MOV DS, AX
MOV DX, OFFSET INT60
MOV AH, 25H
MOV AL, 60H
INT 21H
POP DS
STI
…...
```

中断服务程序

```
INT60   PROC FAR

        ……

        IRET
INT60   ENDP
```

5.4 8259A在微机中的应用

例　用用户定义的中断服务程序置换系统原有的中断服务程序

```
DATA   SEGMENT
          OLD_INT_SEG  DW ?
          OLD_INT_OFF  DW ?
          …
DATA   ENDS
          ….
          MOV AL, N
          MOV AH, 35H
          INT 21H
          MOV OLD_INT_SEG, ES
          MOV OLD_INT_OFF, BX
          CLI
          PUSH DS
          MOV AX, SEG NEW_INT
          MOV DS, AX
          MOV DX, OFFSET NEW_INT
          MOV AH, 25H
          MOV AL, N
          INT 21H
          POP DS
          STI
          ……
          CLI
          PUSH DS
          MOV AX, SEG OLD_INT_SEG
          MOV DS, AX
          MOV DX, OFFSET OLD_INT_OFF
          MOV AH, 25H
          MOV AL, N
          INT  21H
          STI
          ….…
```