操作系统实验三
中断和异常

南京大学软件学院

广义上的中断

- 中断(广义的)是指程序执行过程中，遇到急需处理的事件时，暂时中止CPU上现行程序的运行，转去执行相应的事件处理程序，待处理完成后再返回原程序被中断处或调度其他程序执行的过程。

中断的分类

- 从中断源的角度分类

① 由计算机硬件异常或故障引起的中断，也称为**内部异常中断**。

② 由程序中执行了中断指令引起的中断，也称为**软中断**。由程序员通过INT或INT3指令触发，通常当做trap处理，用处：实现系统调用。

③ 外部设备(如输入输出设备)请求引起的中断，也称为**外部中断或I/O中断**。

中断的分类

- 主要有两类：

① 由CPU以外的事件引起的中断

如I/O中断、时钟中断、控制台中断等。

——中断

② 来自CPU的内部事件或程序执行中的事件引起的过程。

如由于CPU本身故障、程序故障和请求系统服务的指令引起的中断等。

——异常

异常的分类

- Fault，是一种可被更正的异常，而且一旦被更正，程序可以不失连续性地继续执行。返回地址是产生fault的指令。
- Trap，一种在发生trap的指令执行之后立即被报告的异常，它也允许程序或任务不失连续性地继续执行。返回地址是产生trap的指令之后的那条指令。
- Abort，不总是报告精确异常发生位置的异常，不允许程序或任务继续执行，而是用来报告严重错误的。

外部中断的分类

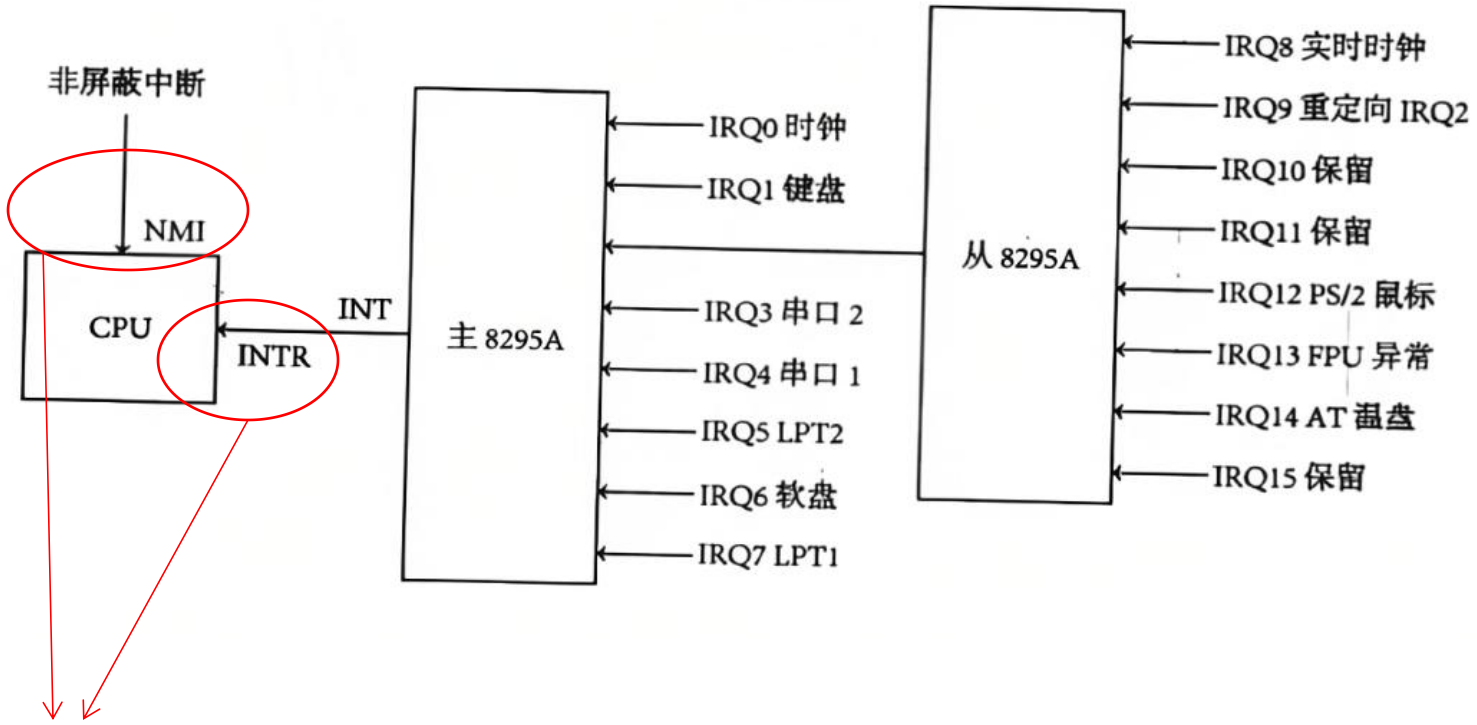
① 可屏蔽中断：

禁止响应某个中断，保证在执行一些重要的程序中不响应中断，以免造成迟缓而引起错误。

② 不可屏蔽中断

重新启动、电源故障、内存出错、总线出错等影响整个系统工作的中断是不能屏蔽的。

中断控制器和中断通道



可屏蔽中断与不可屏蔽中断(NMI)分别由CPU的两根引脚INTR 和NMI来接收。

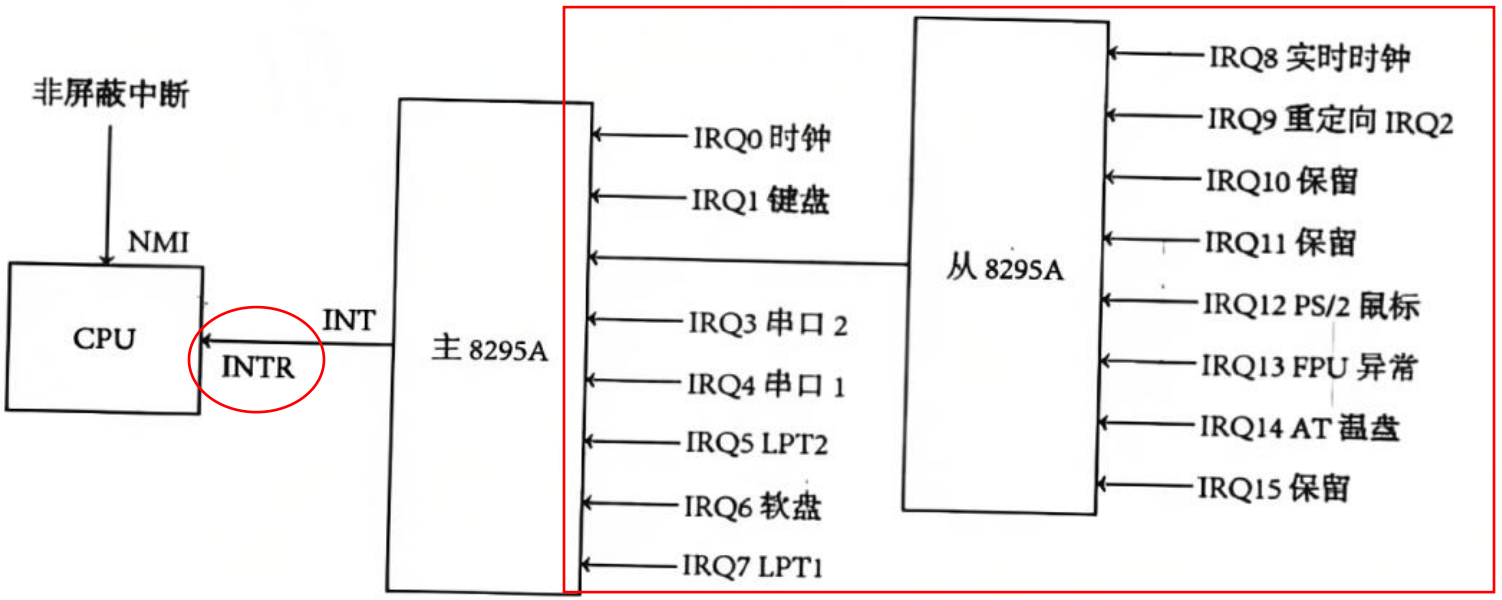
8259A

- 8259A是一个可编程中断控制器

  (Programmable Interrupt Controller, PIC)。它是中断的管理者。

- 主要功能：
  - 设置外部中断的优先级
  - 屏蔽某些外部中断等

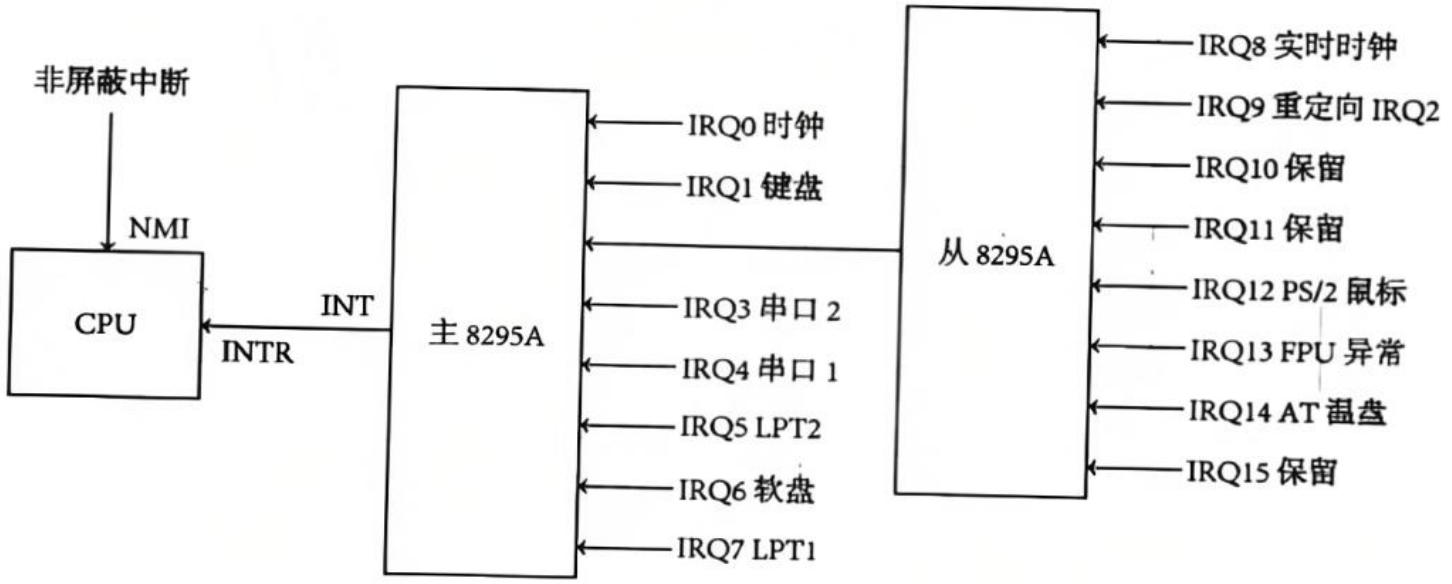
中断控制器和中断通道



- 一个INTR引脚——多个设备

因为每个设备都要使用中断，每个设备也就需要一个传送中断请求的通道。而CPU中只有一条接收可屏蔽中断请求的引脚，因此需要有一个机构来收集各个设备产生的各种中断请求，并按优先级排列送给CPU。这个机构称为中断控制器。

中断控制器和中断通道



- 早期的中断控制器是一片8259集成芯片，可以接收8个中断请求信号，也就是可以有8个中断通道。PC机允许使用15个中断通道，因此需要两片8259芯片。现在的微机仍然维持了这个结构，不过8259芯片已不是独立的芯片，而被进一步集成到其它的大规模芯片中了。
- 两片8259之间用级联的方法连接起来，即一片8259的输出连接到另一片8259的输入端。因此实际可以使用的中断通道只有15个。
- 如此少的中断请求线显然不够用，Linux系统中，多个设备可共用一条中断线，不同设备触发中断时，执行同一个中断处理程序，但是调用不同的中断处理例程。

中断与异常的共同点

- 都是程序执行过程中的强制性转移，转移到相应的处理程序。
- 都是软件或者硬件发生了某种情形而通知处理器的行为。

中断与异常的区别

- 1、中断，是CPU所具备的功能。通常因为“硬件”而随机发生。

  异常，是“软件”运行过程中的一种开发过程中没有考虑到的程序错误。

- 2、中断是CPU暂停当前工作，有计划地去处理其他的事情。中断的发生一般是可以预知的，处理的过程也是事先制定好的。处理中断时程序是正常运行的。

  异常是CPU遇到了无法响应的工作，而后进入一种非正常状态。异常的出现表明程序有缺陷。

中断与异常的区别

- 3、中断是异步的，异常是同步的。

中断是来自处理器外部的I/O设备的信号的结果，它不是由指令流中某条指令执行引起的，从这个意义上讲，它是异步的，是来自指令流之外的。

异常是执行当前指令流中的某条指令的结果，是来自指令流内部的，从这个意义上讲它们都是同步的。

中断与异常的区别

- 4、中断或异常的返回点

良性的如中断和trap，只是在正常的工作流之外执行额外的操作，然后继续干没干完的活。因此处理程序完了后返回到原指令流的下一条指令，继续执行。

恶性的如fault和abort，对于可修复fault，由于是在上一条指令执行过程中发生(是由正在执行的指令引发的)的，在修复fault之后，会重新执行该指令；至于不可修复fault或abort，则不会再返回。

中断与异常的区别

- 5、中断是由于当前程序无关的中断信号触发的，CPU对中断的响应是被动的，且与CPU模式无关。既可以发生在用户态，又可以发生在核心态。

  异常是由CPU控制单元产生的，大部分异常发生在用户态。

补充：

处理器状态(处理器模式)可分为***核心态***和***用户态***。

当处理器处于核心态时，CPU运行可信软件，硬件允许执行全部机器指令。

当处理器处于用户态时， CPU运行非可信软件，程序无法执行特权指令，且访问权限仅限于当前CPU上进程的地址空间。

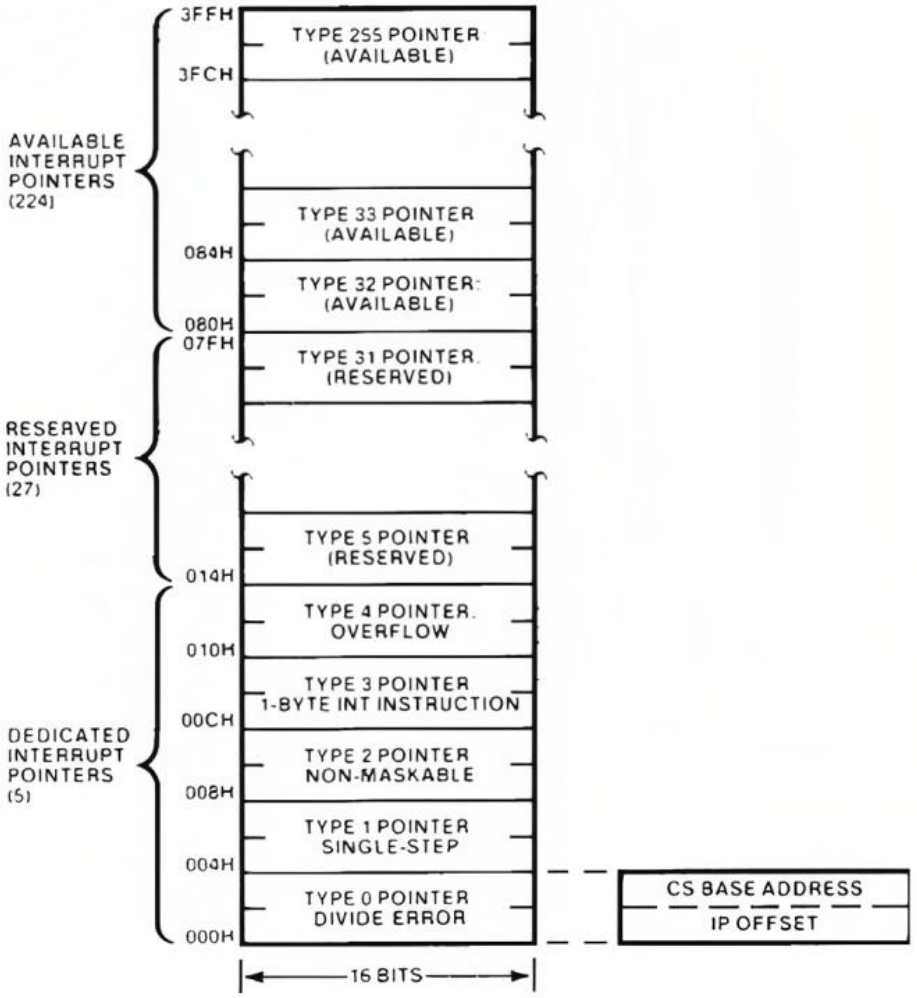
中断向量

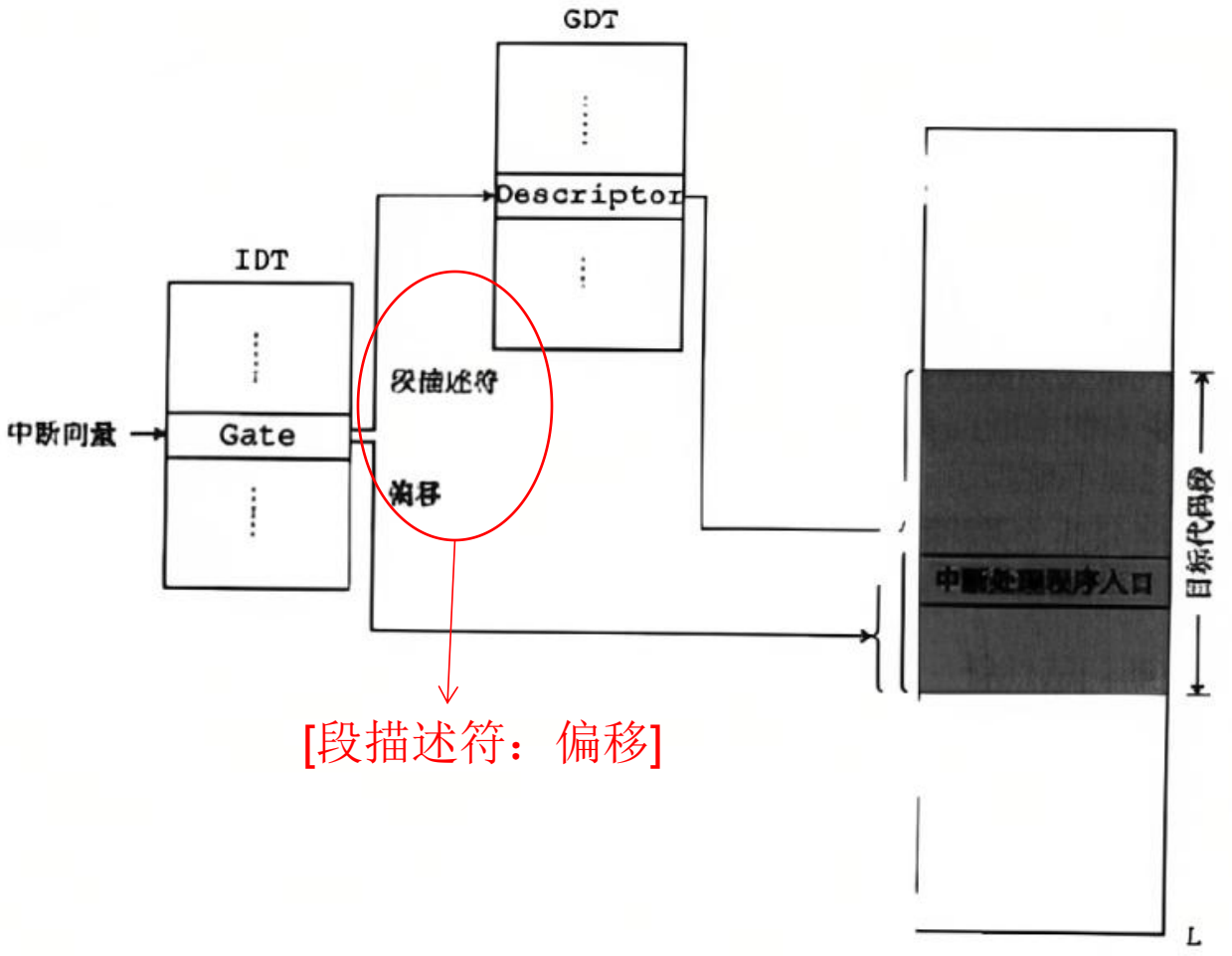
- 把中断/异常与相应的处理方法对应起来
- 每种中断都会对应一个中断向量号，而这个向量号通过IDT(中断描述符表)就与相应的中断处理程序对应起来了。
- 注意中断向量表和中断描述符表的区别

中断向量表

- 起始地址：0
- 每个中断向量包含4 Bytes
- 低地址两个Byte放偏移
- 高地址两个Byte放段描述符
- 最多256个中断向量

- BIOS中断调用说明：
  http://blog.csdn.net/regionyu/article/details/1708084



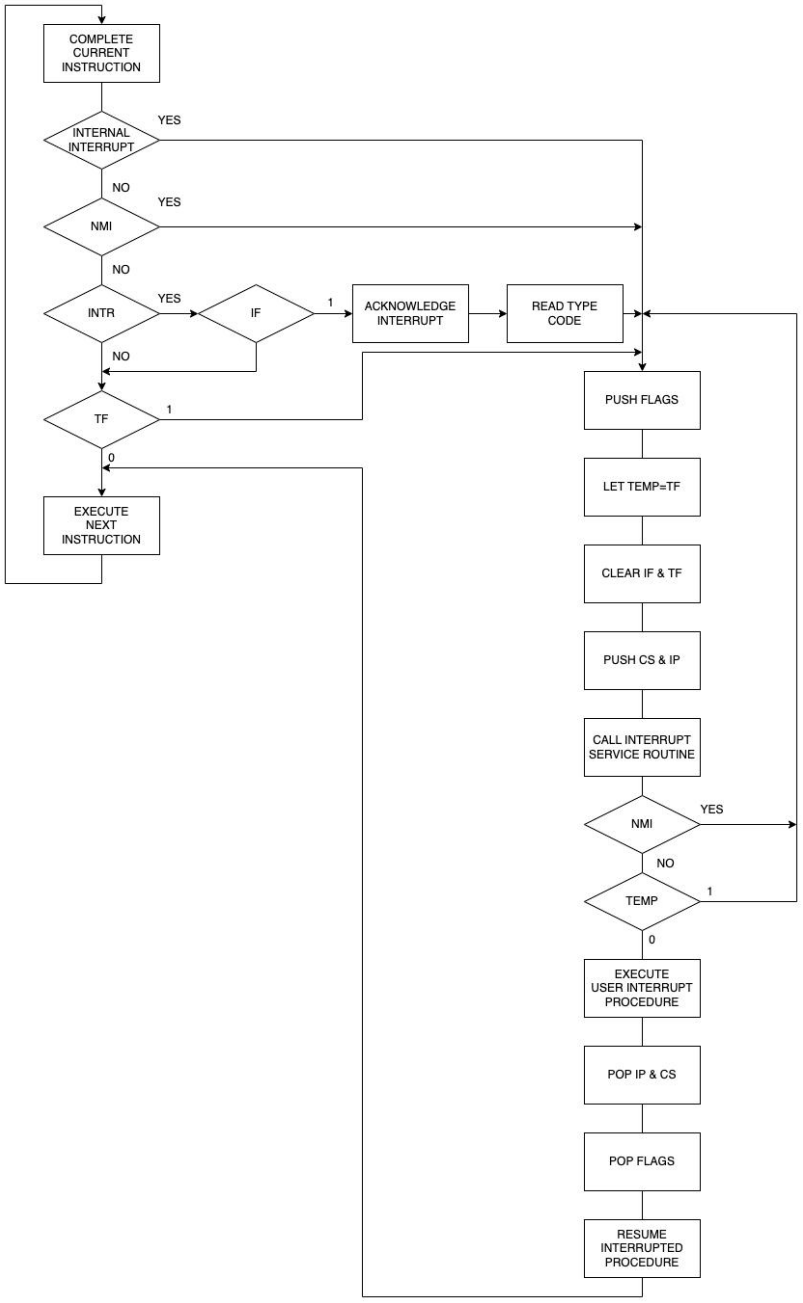


[段描述符：偏移]

中断过程

CPU每条指令运行完后检查一下是否有中断。

- 如果遇到无法避免的(如内部中断等)，会把FLAGS，CS，IP依次压栈，接着关中断(中断位为FLAGS的IF位)，进入对应的例程。
- 遇到可以暂时不管的中断(可屏蔽中断)，就先检查一下IF位。倘使它处于“开”状态，就执行上述过程，反之不管。



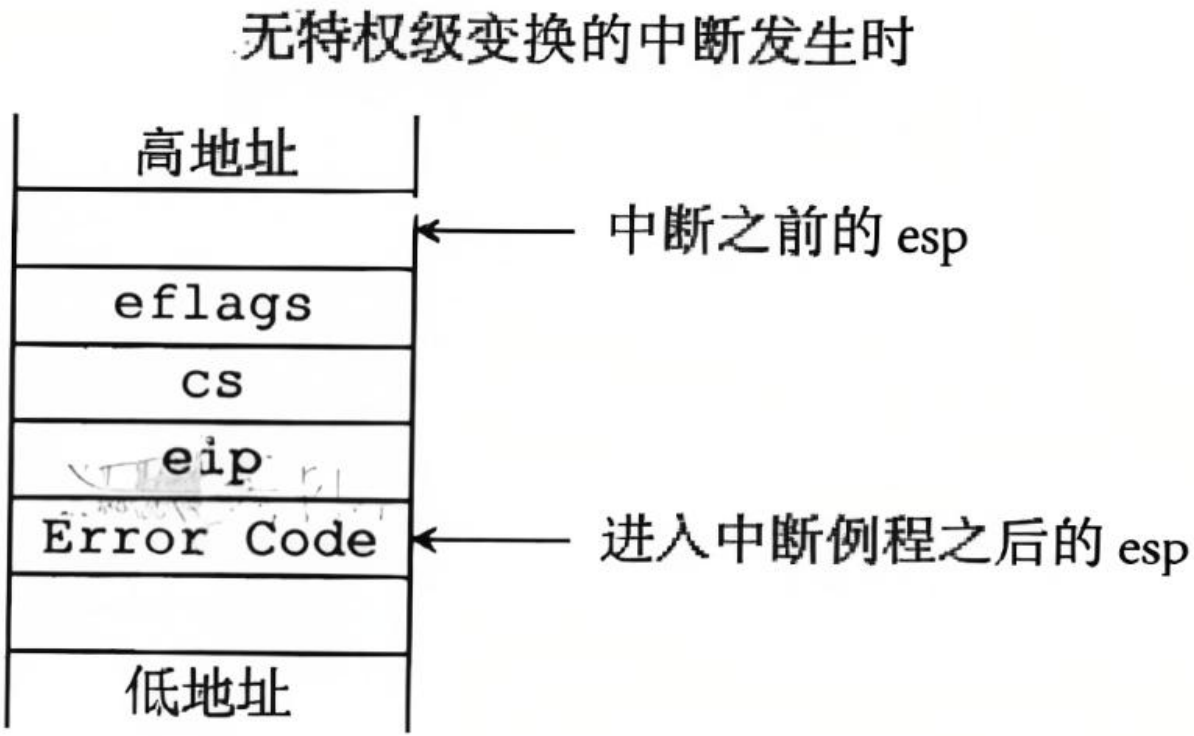
从硬件的角度看：CPU如何处理中断和异常

- 假定内核已被初始化，CPU已从实模式转到保护模式。
- 当CPU执行了当前指令之后，在对下一条指令执行前，CPU先要判断在执行当前指令的过程中是否发生了中断或异常。如果发生了一个中断或异常，那么CPU将做以下事情：
  - ① 确定所发生中断或异常的向量i(在0～255之间)
  - ② 通过IDTR寄存器找到IDT表，读取IDT表第i项(或叫第i个门)。进行有效性检查、特权级变化检查。

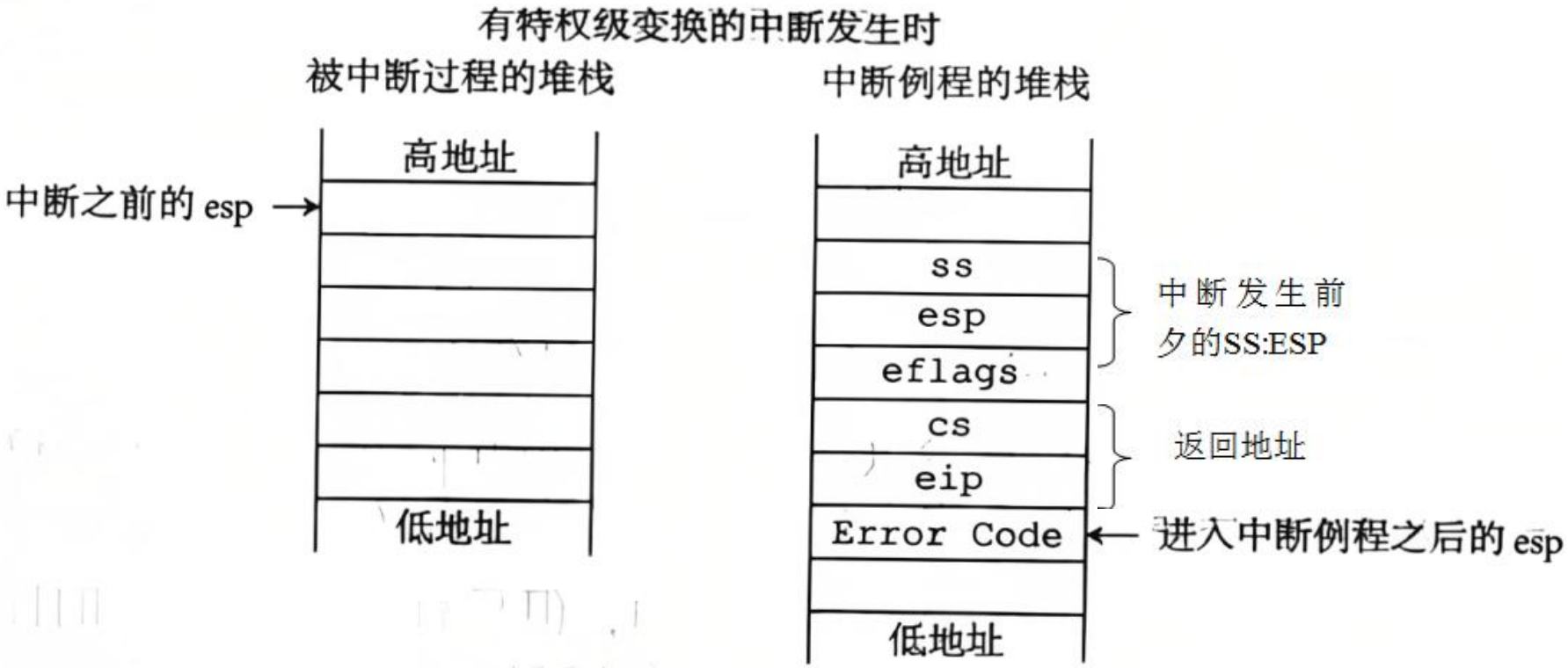
特权级变换

- 当中断发生在用户态(特权级为3)，而中断处理程序运行在内核态(特权级为0)，特权级发生了变化，所以会引起堆栈的更换。也就是说，从用户堆栈切换到内核堆栈。
- 而当中断发生在内核态时，即CPU在内核中运行时，则不会更换堆栈。

中断或异常发生时的堆栈变化



中断或异常发生时的堆栈变化



Thanks !