1. 解释中断向量

**中断向量是中断的识别标志**。对应每个中断源设置一个向量。这些向量顺序存在主存储器的特定存储区。向量的内容是相应**中断服务程序的起始地址和处理机状态字**。在响应中断时，由中断系统硬件提供向量地址，处理机根据该地址取得向量，并转入相应的中断服务程序

**(中断是指在计算机执行程序的过程中，当出现异常情况或者特殊请求时，计算机停止现行的程序的运行，转而对这些异常处理或者特殊请求的处理，处理结束后再返回到现行程序的中断处，继续执行原程序。)**

1. 解释中断类型码

对每个中断服务例程进行编号——中断类型码，代表一个中断服务例程，用来查找中断向量。

Int N中N即为中断类型码

1. 解释中断向量表

指**中断服务程序入口地址的偏移量与段基值**

一个中断向量占据4字节

**按中断类型号顺序存储了中断向量**

共256个中断向量，占据内存的最低1KB

前32个为硬件系统预留，其他的用户自定义

每一个中断向量：**低2B：偏移量 | 高2B：段地址(助教有问，答错了，傻了，一般偏移都是在低位)**

中断类型号和中断向量之间有下面的关系： **中断类型号×4=存放中断向量的首地址(因为一个中断向量占4B)** 有了存放中断向量的首地址，从该地址开始的4个存储单元中取出的就是中断服务程序的入口

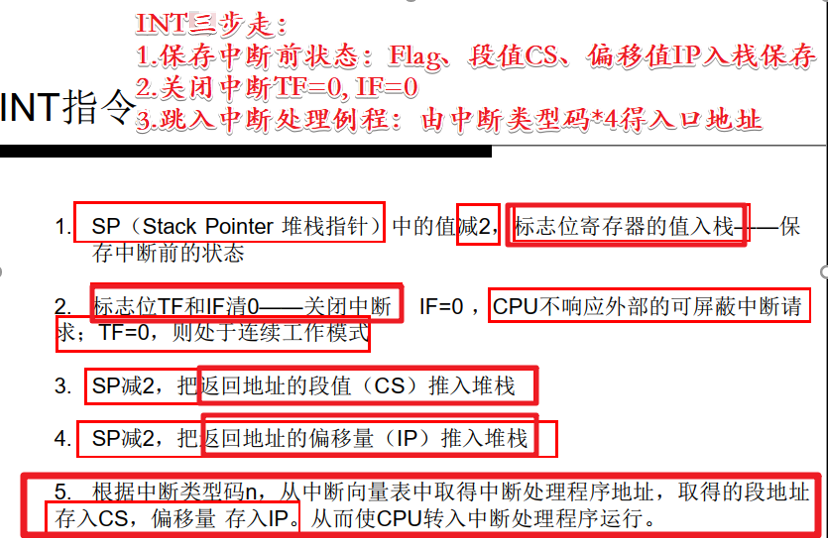
总结一下：

中断向量：中断处理例程的入口地址

中断类型码：中断处理例程编号

中断向量表：按中断类型码对中断向量排序

1. 实模式下中断程序地址如何得到?



执行中断返回指令**IRET(**Interrupt return)返回主程序断点继续执行，IRET就是恢复中断前状态——从栈中取出Flag、CS、IP

1. 保护模式下中断程序地址如何得到?

和实模式最大的不同在于中断处理程序入口地址的寻找

保护模式下，每一个中断/异常都有一个中断描述符ID，IDT取代中断向量表

Interrupt Descriptor Table中断描述符表

每一个IDT8字节

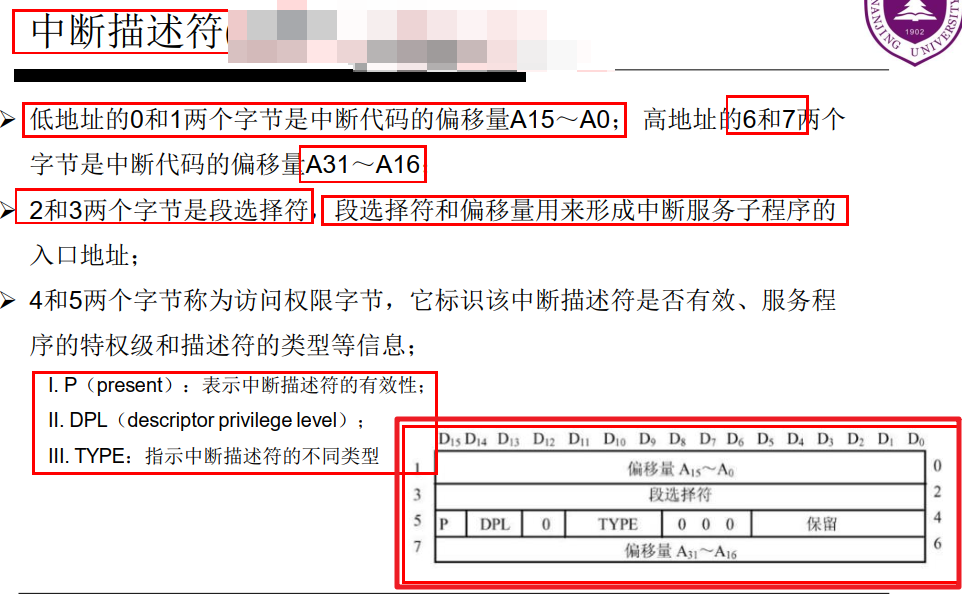
IDT分3类：

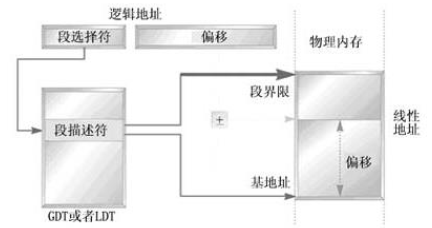
1. **中断门描述符：中断使用**

2. **陷阱门描述符**

3. **任务门描述符**（在Linux中没有使用）

CPU对不同门有不同处理





**48位寄存器IDTR描述IDTs的存放地址**

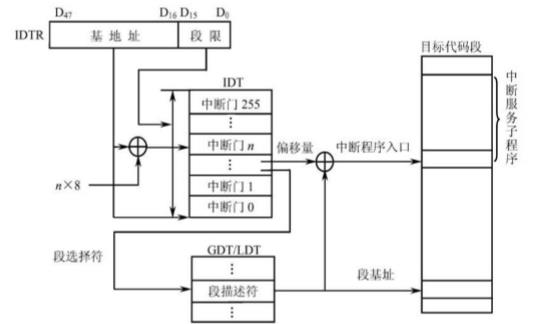
IDTR：32位基地址+16位段界限

**保护模式中断过程**：

根据中断门描述符获取中断处理程序的段描述符，再通过段描述符得到地址

即中**【中断门描述符->段描述符->GDT/LDT -> 物理地址】**

* Step1：**装载IDTR**(中断描述符寄存器)。CPU进入保护模式前，必须执行LIDT装载中断描述符表IDT。回到实模式时会装入000000H的基地址值与03FFH的段界值
* Step2：**查找IDT**。使用IDTR的基地址，N\*8偏移即得到N号IDT的首地址。
* Step3：**查找GDT/LDT得到地址**。工具IDT中的选择子(段描述符)查G/LDT得到入口地址。

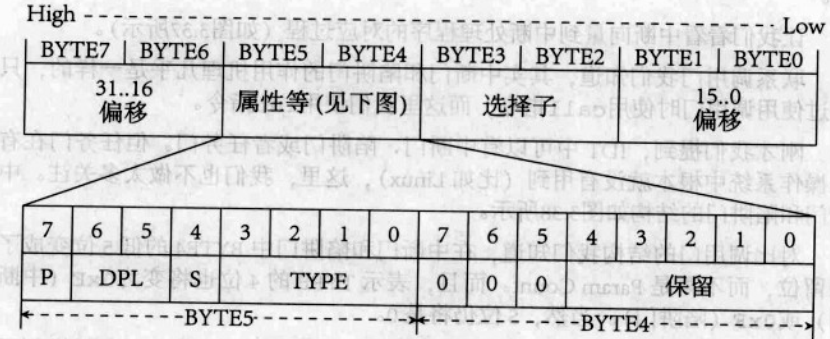


1. 中断向量的地址如何得到?

IDTR基地址+N\*8偏移

1. 实模式下如何根据中断向量的地址得到中断程序地址?
2. 解释中断描述符

IDT



1. 保护模式下中断描述符表如何得到?
2. 保护模式下中断门如何得到?
3. 保护模式下如何根据中断门得到中断处理程序地址?
4. 中断的分类，举例不同类型的中断?

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。

从中断源分类：

1. 内部异常中断：计算机硬件异常/故障引起
2. 软中断：执行了中断指令(如INT、INT3)引起，用于实现系统调用
3. 外部中断(IO中断)：外部设备(如输入输出设备)请求引起。外部中断又分为：
   1. 可屏蔽
   2. 不可屏蔽

广义中断主要分为两类：

1. **中断：CPU以外的事件引起的，如IO中断，时钟中断**
2. **异常：CPU内部异常/程序执行中的事件引起。**
3. 中断与异常的区别?
   1. 共同点：
      1. 都是强制性转移
      2. 都是软件/硬件发生某种情形而通知处理器的行为
   2. 不同：
      1. 中断是CPU具备的功能。而异常是软件开发中没有考虑到的错误
      2. 中断的发生是可以预知的，是暂停CPU当前任务，**有计划地处理其他事情**。异常是CPU遇到无法响应的事件，进入非正常状态，表明程序的缺陷
      3. 中断是异步的，是来自于指令流外部。而异常是同步的，发生于指令流内部。
      4. 良性：中断、trap；恶性：异常中的fault\abort
      5. 中断是由与当前程序无关的中断信号触发的(中断是**其他来打断当前**)，CPU对中断的响应是被动的，与模式无关。**异常是CPU控制单元产生，大部分发生在用户态。**
4. 实模式和保护模式下的中断处理差别
5. 如何识别键盘组合键 (如 Shift+a) 是否还有其他解决方案?

先检测第一个键，如果输入是Shift，存起来，再输入第二个键，进行转换

1. IDT 是什么，有什么作用?
2. IDT 中有几种描述符?
3. 异常的分类?

异常分为三类：

* **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。

1. 用户态和内核态的特权级分别是多少?
   1. 用户态：3
   2. 内核态：0

如果特权级变化，需要切换栈堆：用户栈堆🡸🡺内核栈堆

1. 中断向量表中，每个中断有几个字节? 里面的结构是什么?
2. 中断异常共同点 (至少两点)，不同点 (至少三点)