**问题清单**

1.解释中断向量

中断向量，即将中断号及中断处理程序位置（首地址）对应起来的4字节字段

2.解释中断类型码

中断类型码，即中断服务程序的编号，用于查找中断向量

3.解释中断向量表

中断向量表是一片专门用于存放所有中断向量的内存空间（全表共256个中断向量，每个向量4字节，地址空间为00000H-003FFH（1024B）），提供了“中断类型码”与中断处理代码段入口地址的映射（中断向量首地址=中断类型码\*4）

4.实模式下中断程序地址如何得到？

由中断类型码\*4得到中断向量（段值：偏移），实模式下中断程序地址=段值\*16+偏移量

5.保护模式下中断程序地址如何得到？

首先利用IDTR中的IDT基址及中断类型码（基址+中断号\*8）取得中断描述符，然后通过描述符中的选择子查表（GDT或LDT）得到段描述符，最后将段基址与中断描述符中的偏移量相加即有保护模式下中断程序地址

6.中断向量的地址如何得到？

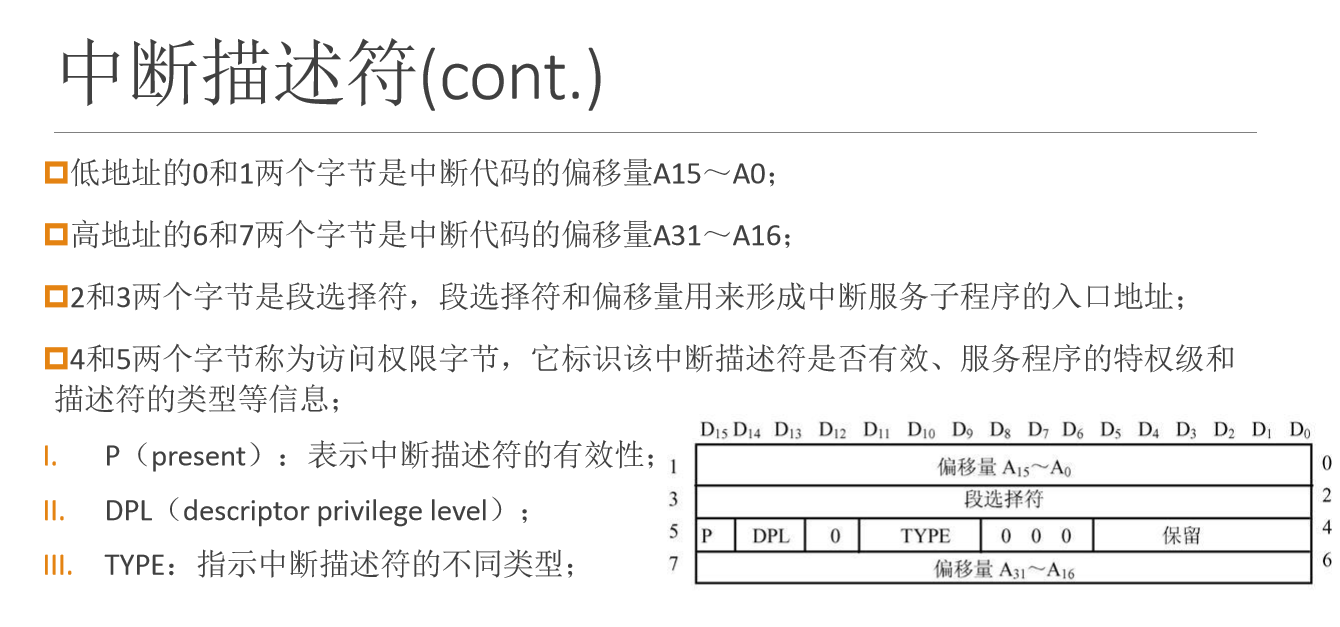
中断向量基址=中断类型码\*4

7.实模式下如何根据中断向量的地址得到中断程序地址？

由地址获得32位中断向量，内容为段值：偏移（各16位），中断程序地址=段值\*16+偏移量

8.解释中断描述符

中断描述符，又称门描述符，分任务门、中断门、自陷门三类，占用连续的8字节，包含中断处理程序地址信息及其他类型属性信息，具体见下图：



9.保护模式下中断描述符表如何得到？

48位的全地址寄存器IDTR中存放了32位的段基址和16位的段界限，32位的段基址即为中断描述符表在存储器中的起始点

10.保护模式下中断⻔如何得到？

中断门=IDT基址+中断类型码\*8（每个中断描述符占8个字节，IDT基址存储在IDTR中）

11.保护模式下如何根据中断⻔得到中断处理程序地址？

先利用中断门中的段选择符（2、3字节）查表（GDT、LDT）得到段描述符，继而取得中断处理程序所在段基址，再加上中断门中的偏移（0、1及6、7字节）即可获得中断处理程序的地址

12.中断的分类，举例不同类型的中断？

从中断源的角度分类：

➀ 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**。

➁ 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。

➂ 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**。

13.中断与异常的区别？

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。

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。

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。

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。

14.实模式和保护模式下的中断处理差别

实模式和保护模式下中断处理的最大差别在于寻找中断处理程序入口地址的方式

实模式下通过中断向量（段值：偏移）指示中断处理程序入口的地址，而

保护模式下通过中断描述符来说明中断处理程序的入口地址属性，这时亦

通过IDT对实模式下的中断向量表进行取缔

15.如何识别键盘组合键（如Shift+a）是否还有其他解决⽅案？

键盘组合键对应着一个扫描码序列，以左Shift+a为例，为0x2A0x1E0x9E0xAA，分别是左Shift键的MakeCode，a的MakeCode，a的BreakCode，左Shift键的BreakCode，可以将该扫描码序列作为整体识别键盘组合键

其他解决方案：

可以为组合键中的前置单键设置Flag，出现标识键位则将Flag置1，继而通过检查后续扫描码来识别键盘组合键

16.IDT是什么，有什么作⽤？

IDT即中断描述符表，存放着所有的中断描述符，它用于取代实模式下的中断向量表，提供了中断类型码到中断处理程序入口地址的映射

17.IDT中有⼏种描述符？

IDT中有三种描述符，分别为任务门、中断门以及自陷门

18.异常的分类？

根据保存在eip寄存器中的值,分为3种：

* 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。

19.⽤⼾态和内核态的特权级分别是多少？

一般而言，用户态的特权级为3，内核态的特权级为0

20.中断向量表中，每个中断有⼏个字节？⾥⾯的结构是什么？

中断向量表中，每个中断向量有4个字节，高地址两个字节放段描述符，低地址两个字节放偏移

21.中断异常共同点(⾄少两点)，不同点(⾄少三点)

中断异常共同点：

1. 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。
2. 都是软件或者硬件发生了某种情形而通知处理器的行为

不同点：

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。

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。

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。

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。