1. 解释中断向量

向量就是确定确切位置的含义，中断向量的含义就是可以确定中断服务程序位置，也就是中断向量就是中断服务程序的首地址。中断服务程序的首地址，需要4个内存空间存储。

2. 解释中断类型码

我们把每个中断服务程序进行编号，这个号就代表一个中断服务程序，就是中断类型码。这个中断类型码是计算机用来查找中断向量用的。

中断指令的一般格式为 “INT n”其中，n称为“中断类型码”，n=0-255；

3. 解释中断向量表

存放所有的中断向量的地址空间。也就说中断向量表是一片内存空间，是一片专门用来存放中断向量的内存空间。中断向量表在内存单元的最低处，地址空间为00000H----003FFH(0-1024B)，这个正好可以和中断类型码有一种对应的关系，也就是说中断类型码*4(一个中断向量所占的空间)就等于这个中断向量的首地址。

4. 实模式下中断程序地址如何得到？

中断类型号×4=存放中断向量的首地址；

每一个中断向量所包含的地址以低位二字节存储偏移量，高位二字节存储段地址；

根据中断类型码n，从中断向量表中取得中断处理程序地址，取得的段地址存入CS，偏移量存入IP。从而使CPU转入中断处理程序运行。

5. 保护模式下中断程序地址如何得到？

在保护模式下，为每一个中断和异常定义了一个中断描述符，来说明中断和异常服务程序的入口地址的属性；

由中断描述符表(IDT)取代实地址模式下的中断向量表

中断描述符中的：

低地址的0和1两个字节是中断代码的偏移量A15～A0；

高地址的6和7两个字节是中断代码的偏移量A31～A16

2和3两个字节是段选择符，段选择符和偏移量用来形成中断服务子程序的入口地址。

6. 中断向量的地址如何得到？

实模式只要使用调用号N×4即可找到该向量的首地址，由此处再转移到中断服务程序。保护模式下要借助中断门描述符来获取中断子程序这个目标段的描述符，也就是说必须经过两次查表才能获得中断服务子程序的入口地址。

7. 实模式下如何根据中断向量的地址得到中断程序地址？

每一个中断向量所包含的地址以低位二字节存储偏移量，高位二字节存储段地址；

根据中断类型码n，从中断向量表中取得中断处理程序地址，取得的段地址存

入 CS，偏移量存入 IP。从而使 CPU 转入中断处理程序运行。

8. 解释中断描述符

中断描述符表的每一个项目（称作门描述符、中断描述符）除了含有中断处理程序地址信息外，还包括许多属性和类型位；每个中断描述符占用连续的 8 个字节；

中断描述符分为三类：任务门、中断门和自陷门，CPU 对不同的门有不同的处理方式。

低地址的 0 和 1 两个字节是中断代码的偏移量 A15～A0；

高地址的 6 和 7 两个字节是中断代码的偏移量 A31～A16；

2 和 3 两个字节是段选择符，段选择符和偏移量用来形成中断服务子程序的入口地址；

4 和 5 两个字节称为访问权限字节，它标识该中断描述符是否有效、服务程序的特权级和描述符的类型等信息

9. 保护模式下中断描述符表如何得到？

CPU 切换到保护模式之前，运行于实模式下的初始化程序必须使用 LIDT 指令装载中断描述符表 IDT，将 IDT 基地址与段界值装入 IDTR。

10. 保护模式下中断门如何得到？

查中断描述符表：以 IDTR 指定的中断描述符表的基地址为起始地址，用调用号 N×8 算出偏移量，即为 N 号中断门描述符的首地址，由此处取出中断门的 8 个字节。

10. 保护模式下如何根据中断门得到中断处理程序地址？

根据中断门中的选择子和偏移量得到中断处理程序入口：

根据段寄存器的内容(选择子)，首先判断描述符是在 GDT 中还是在 LDT 中，如果是在 GDT 中，根据 GDTR 以及该段寄存器的内容找到相应的“描述符”；如果是在 LDT 中，根据 LDTR（选择子）以及 GDTR 的内容找到 LDT 的描述符，得到 LDT 的地址，然后再根据段寄存器内容找到相应的“描述符”。

从描述符中得到基地址

将指令中发出的地址作为位移，与描述符中界限相比，看是否越界；

将指令的性质与描述符中的访问权限来确定是否越权；

将指令中发出的地址作为位移，与基地址相加得出实际的“物理地址”

11. 中断的分类，举例不同类型的中断？

- 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**。
- 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。

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。

- 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。

13. 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

保护模式下的中断处理与实模式下的中断处理最大区别在于寻找中断处理代码入口的方式；

在保护模式下，为每一个中断和异常定义了一个中断描述符，来说明中断和异常服务程序的入口地址的属性；由中断描述符表取代实地址模式下的中断向量表；

- 实模式
  e.g.通过 int 15h 得到计算机内存信息，然后在保护模式下把它们显示出来
- 保护模式
  中断向量表->IDT（中断描述符表）

15. 如何识别键盘组合键（如 Shift+a）。是否还有其他解决方案？

交给你去回答了。

用一个全部变量 caps，记录 shift 键是否被按下。当按下 shift+a 时，进行两次读键盘按键操作，读到 shift 被按下时，将 caps 置为 true。第二次再读 a 被按下，此时判断如果 caps 被置为了 true，那么就取 keymap 中的第二列的值，即对应字母的大写 A。

16. IDT 是什么，有什么作用？

在保护模式下，为中断服务提供中断/陷阱描述符，这些描述符构成中断描述符表（IDT），为每一个中断和异常定义了一个中断描述符，来说明中断和异常服务程序的入口地址的属性；由中断描述符表取代实地址模式下的中断向量表；

IDT 的作用是将每一个中断向量和一个描述符对应起来。

并引入一个 48 位的全地址寄存器存放 IDT 的内存地址。理论上 IDT 表同样可以有 8K 项，可是因为 80x86 只支持 256 个中断，因此 IDT 实际上最大只能有 256 项（2K 大小）。

17. IDT 中有几种描述符？

中断描述符分为三类：任务门、中断门和自陷门，CPU 对不同的门有不同的处理方式

18. 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 fault 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. 用户态和内核态的特权级分别是多少？

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)。
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。

20. 中断向量表中，每个中断有几个字节？里面的结构是什么？

- 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

21. 中断异常共同点（至少两点），不同点（至少三点）

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

不同点：

- 见 13