|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 武汉大学国家网络安全学院教学实验报告 | | | | |
| 课程名称 | 操作系统设计与实践 | | 实验日期 | 2023.10.15 |
| 实验名称 | 中断与异常 | | 实验周次 | 第四周 |
| 姓名 | 学号 | | 专业 | 班级 |
| 聂森 | 2021302191536 | | 网络空间安全 | 9班 |
| 程子洋 | 2021301051114 | | 网络空间安全 | 9班 |
| 王卓 | 2021302191791 | | 网络空间安全 | 9班 |
| 刘虓 | 2021302121234 | | 网络空间安全 | 9班 |
| 1. 实验目的及实验内容   （本次实验所涉及并要求掌握的知识；实验内容；必要的原理分析） | | | | |
| **实验内容：**   * 1. **本次实验内容：**  1. 学习中断与异常的机制 2. 调试8259A的编程基本例程 3. 调试时钟中断例程 4. 实现一个自定义的中断向量，功能可自由设想    1. **思考问题：** 5. 什么是中断，什么是异常 6. 8259A的工作原理是怎么样的？怎么给这些中断号的处理向量初始化值 7. 如何建立IDT，如何实现一个自定义的中断 8. 如何控制时钟中断，为什么时钟中断的时候，没有看到int的指令？ 9. 简要解释一下IOPL的作用与基本机理   **原理分析：**   1. 8259A：   8259A是一个可编程中断控制器（ProgrammableInterruptController，PIC），通常与Intel8086/8088系列微处理器一起使用，用于处理中断请求。调试8259A编程的基本例程旨在设置和管理中断，以便有效地处理外部设备的中断请求。  我们需要了解8259A的内部结构，包括寄存器和端口，以便了解如何与它通信。掌握8259A的编程模式，包括初始化、中断向量号的设置、级联模式等。明白如何编写中断服务程序，以响应8259A触发的中断。 | | | | |
| 1. 实验环境及实验步骤   （本次实验所使用的器件、仪器设备等的情况；具体的实验步骤） | | | | |
| **实验环境：**  虚拟机：  VMwareWorkstationPro/VMwareWorkstation16player  操作系统：Ubuntu16.04  模拟系统软件：Bochs2.7  **实验步骤：**   1. **学习中断与异常的机制** 2. 中断和异常的概念：   中断和异常都是软件或者硬件发生了某种情形而通知处理器的行为，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。每一种中断都对应一个中断向量号，这个向量号通过IDT与对应的中断程序对应起来。    图1 中断向量到中断处理程序的对应过程   * 1. 中断   这里的中断指的是外部中断。中断的引入是为了支持CPU和设备之间的并行操作。中断可以分为可屏蔽中断和不可屏蔽中断。   * 可屏蔽中断：指通过可屏蔽中断请求线INTR向CPU发出中断请求，CPU可以通过在中断控制器中设置响应的屏蔽字来屏蔽他或不屏蔽他，被屏蔽的中断请求将不被送至CPU。 * 不可屏蔽中断：指通过专门的不可屏蔽请求线NMI向CPU发出的中断请求，通常是非常紧急的硬件故障。   注：外部中断是异步的，意思是所有中断来的信号都是记录在中断寄存器中的，当CPU执行完一道指令后，如果是开中断状态，就会检查中断寄存器中有没有中断，如果有中断，就会选择一个中断优先级比较高的中断先处理，等到处理完之后再继续执行；如果是关中断，则不会检查，而直接执行下一条指令。   * 1. 异常   异常也叫内部中断，它不需要硬件支持。异常的引入表示 CPU 执行指令时本身出现的问题。异常可分为以下三种类型：   * Fault：一种可以可以被更正的异常。而且一旦被更正，程序可以不失连续性地继续执行。当一个 fault 发生，处理器会把产生 fault 指令之前的状态保存起来，异常处理程序的返回地址将会是产生 fault 的指令，而不是其后的那条指令。 * Trap：是一种在发生 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。异常处理程序的返回地址将会是产生 trap 的指令之后的那条指令。 * Abort：是一种不总是报告准确异常发生位置的异常，它不允许程序或者任务继续执行，而是用来报告严重错误的。   注：异常是同步的，这是指异常发生的时候，CPU立即处理本次异常，直到异常处理结束后才能继续进行接下来的任务。   1. 处理机制    1. 实模式下的中断处理：   实模式下，中断转移方法与8086相同，即通过中断向量号直接去中断向量表中找到中断处理程序入口，然后跳转到指定位置执行中断处理程序。    图2 实模式中断处理   * 1. 保护模式下的中断处理：   不同于实模式，在保护模式下，中断向量表被IDT代替。IDT的作用是将每一个中断向量和一个描述符对应起来。IDT的描述符有三类：中断门描述符、陷阱门描述符、任务门描述符。   * 中断门和陷阱门  1. 结构   中断门结构  图3 中断门和陷阱门结构  其中灰色部分表示保留，不使用。其中段选择码和偏移用来定位中断处理程序，其余标志该描述符的属性。   1. 处理机制   任务门  图4 中断门和陷阱门处理机制  注意：中断门和陷阱门的区别是对中断允许标志IF位的影响。中断门向量引起中断时会复位IF，此时其他中断干扰会被屏蔽，最终通过iret从堆栈上恢复出IF的原值。   * 任务门  1. 结构   任务门结构  图5 任务门结构  同样，灰色部分表示空闲不使用。任务门不需要提供段内偏移，因为任务门不指向某一个子程序的入口，TSS本身是作为一个段来对待的。   1. 处理机制   任务门  图6 任务门处理机制   1. **调试8259A的编程基本例程** 2. 外部中断   中断产生的原因有两种，一种是外部中断，也就是由硬件产生的中断，另一种是由指令intn产生的中断。  对于外部中断，需要建立硬件中断与向量号之间的对应关系。  外部中断分为不可屏蔽中断(NMI)和可屏蔽中断两种，分别由CPU的两根引脚NMI和INTR来接收。    图7 8259A  NMI不可屏蔽，因为它与IF是否被设置无关。NMI中断对应的中断向量号为2。  可屏蔽中断与CPU的关系是通过对可编程中断控制器8259A建立起来的。可以认为它是中断机制中所有外围设备的一个代理，这个代理不但可以根据优先级在同时发生中断的设备中选择应该处理的请求，而且可以通过对其寄存器的设置来屏蔽或打开相应的中断。  与CPU相连的是两片级联的8259A，每个8259A有8根中断信号线，于是两片级联总共可以挂接15个不同的外部设备。  在BIOS初始化的时候，IRQ0～IRQ7被设置为对应向量号08h～0Fh，而通过中断向量表我们知道，在保护模式下向量号08h-0Fh已经被占用了，所以需要重新设置主从8259A。  通过向相应的端口写入特定的ICW（InitializationCommandWord）来实现。主8259A对应的端口地址是20h和21h，从8259A对应的端口地址是A0h和A1h。  ICW共有4个，每一个都是具有特定格式的字节。初始化过程：  1.往端口20h（主片）或A0h（从片）写入ICW1。  2.往端口21h（主片）或A1h（从片）写入ICW2。  3.往端口21h（主片）或A1h（从片）写入ICW3。  4.往端口21h（主片）或A1h（从片）写入ICW4。  这4步的顺序是不能颠倒的。  4个ICW的格式如下图所示，可以看到，在写入**ICW2**时涉及与**中断向量号**的对应，这便是窍门所在了。    图8 ICW的格式   1. 设置8259A   通过函数Initial8259A初始化8259A。  k  l  图9 Initial8259A  这段代码分别往主、从两个8259A各写入了4个ICW。在往主8259A写入ICW2时，我们看到IRQ0对应了中断向量号20h，于是，IRQ0～IRQ7就对应中断向量20h～27h；类似地，IRQ8～IRQ15对应中断向量28h～2Fh。查询表格我们知道，20h～2Fh处于用户定义中断的范围内。  在这段代码的后半部分，我们通过对端口21h和A1h的操作屏蔽了所有的外部中断，这一次写入的不再是ICW了，而是OCW（OperationControlWord）。OCW共有3个，OCW1、OCW2和OCW3。由于我们只在两种情况下用到它，因此并不需要了解所有的内容。这两种情况是：  1.屏蔽或打开中断。  2.发送EOI给8259A以通知它中断处理结束。  若想屏蔽或打开外部中断，只需要往8259A写入OCW1就可以了，OCW1的格式如图所示。    图10 OCW1  可见，若想屏蔽某一个中断，将对应那一位设成1就可以了。实际上，OCW1是被写入了中断屏蔽寄存器（IMR，全称InterruptMaskRegister）中，当一个中断到达，IMR会判断此中断是否应被丢弃。  当每一次中断处理结束，需要发送一个EOI给8259A，以便继续接收中断。而发送EOI是通过往端口20h或A0h写OCW2来实现的。    图11 OCW2   1. 建立IDT   以pmtest9c.asm为例，把第80h号中断单独列出来，并新增加一个函数来处理这个中断：UserIntHandler——屏幕右上角出现红色的“I”。  使用int080h指令进行中断的调用。  o  图12 IDT初始化  结果如图所示，可见屏幕右上角出现红色的“I”:  j  图13 pmtest9c.asm运行结果  至此，完成8259A的编程基本例程。   1. **调试时钟中断例程** 2. 可屏蔽中断，设置8259A,IDT  * 打开时钟中断（IRQ0）   可屏蔽中断与NMI的区别在于是否受到IF位的影响，而8259A的中断屏蔽寄存器（IMR）也影响着中断是否会被响 应。所以，外部可屏蔽中断的发生就受到两个因素的影响，只有当IF位为1，并且IMR相应位为0时才会发生。  想打开时钟中断的话，一方面不仅要设计一个中断处理程序，另一方面还要设置IMR，并且设置IF位。设置IMR 可以通过写OCW2来完成，而设置IF可以通过指令sti来完成。   * 修改初始化8259A的代码，不再屏蔽IR(0)(主8259AIR0为时钟中断)。（11111110，第0位0，开启接收定时器中断）     图14 初始化8259A的代码   * 修改IDT，20h为时钟中断程序     图15 修改IDT的代码   1. 时间中断程序     图16 时间中断程序   1. 设置IF   调用80h号中断之后执行sti来打开中断（STI指令置IF为1，接收可屏蔽中断），时钟中断程序的效果就应该可以看到了。设置死循环原因是有一个问题：程序马上会继续执行，可能没等第一个中断发生程序已经执行完并退出了。    图17 设置IF   1. 时钟向量初始化   在IDT初始化的时候.20h中断设置为对应自己写的时钟中断程序。在8529A初始化的时候，设置IR(0)对应.20h中断。所以每次有外部时钟信号，都会调用时钟中断程序。   1. 效果展示       图18 时间中断程序效果展示   1. **实现一个自定义中断向量，功能自定**    1. 思路设计：   可以基于pmtest9.asm进行修改，在pmtest9.asm中已经实现了时钟中断，其所占用的中断向量是20h，那么可以编写一个新的中断使用中断向量21h。为了使得实验不繁琐复杂，这里编写的新的中断功能为：每次敲击键盘，右上角字符的颜色发生变化。   * 1. 编写中断处理程序：     图19 编写中断处理程序  该中断处理程序实现的功能就是，每次敲击键盘，右上角字符的颜色发生变化。该程序的编写并不复杂主要是想体验中断设计的过程。   * 1. 添加IDT表项：     图21 添加IDT表项  这里分配的是021h这个中断向量，除中断向量和偏移的不同之外，其余部分均相同，注意对齐格式。   * 1. 初始化8259A，并打开相应的中断     图22 初始化8259A，并打开相应的中断  翻阅教材，发现021h对应的是IRQ1，所以只需要修改OCW1的值即可以实现中断的打开和屏蔽。具体来说（见下图可知），修改第二位值为0即可。    图23 初始化OCW1  同时为了屏蔽其他中断而只去执行我们编写的中断程序，我们修改初始化8259A的程序如下所示：    图24 修改初始化8259A代码   * 1. 敲击键盘验证实验结果       图25-27 实验结果展示  可以看到三次不同的颜色，说明编写的新的中断非常成功！ | | | | |
| 1. 实验过程分析   （实验分工，详细记录实验过程中发生的故障和问题，进行故障分析，说明故障排除的过程及方法。根据具体实验，记录、整理相应的数据表格等） | | | | |
| **实验问题与故障分析：**   1. 编写了中断程序但是系统并不执行该中断     图28 故障1  可以看到系统并没有执行该段程序，只是跳过执行下面的程序。  解决：  查阅教材之后发现这是因为没有正确设置OCW1导致的。应该要将OCW1的相应位置改为0，而不是完全照搬11111110，这样的话中断会被屏蔽而不是执行。说明需要仔细阅读教材，按照步骤一步一步解决。 | | | | |
| 1. 实验结果总结   （对实验结果进行分析，完成思考题目，并提出实验的改进意见） | | | | |
| 1. **什么是中断，什么是异常？**   中断和异常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。  中断是CPU暂停当前工作，有计划地去处理其他的事情，它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。  每一种中断（异常）都会对应一个中断向量号，而这个向量号通过IDT就与相应的中断处理程序对应起来。   1. **8259A的工作原理是怎样的？怎么给这些中断号的处理向量初始化值？** 2. 8258A的工作原理：   先通过ICW1∼ICW4来初始化8259A，使芯片处于一个规定的基本工作方式。初始化完成后，开中断，一个外部中断请求信号通过中断请求线IRQ，传输到IMR（中断屏蔽寄存器），IMR根据所设定的中断屏蔽字（OCW1），决定是将其丢弃还是接受。  如果可以接受，则8259A将IRR（中断请求暂存寄存器）中代表此IRQ的位置位，以表示此IRQ有中断请求信号，并同时向CPU的INTR管脚发送一个信号，但CPU这时可能正在执行一条指令，因此CPU不会立即响应，而当这CPU正忙着执行某条指令时，还有可能有其余的IRQ线送来中断请求，这些请求都会接受IMR的挑选，如果没有被屏蔽，那么这些请求也会被放到IRR中，也即IRR中代表它们的IRQ的相应位会被置1。  当CPU执行完一条指令时后，会检查一下INTR管脚是否有信号，如果发现有信号，就会转到中断服务，此时，CPU会立即向8259A芯片的INTA（中断应答）管脚发送一个信号。当芯片收到此信号后，判优部件开始工作，它在IRR中，挑选优先级最高的中断，将中断请求送到ISR（中断服务寄存器），也即将ISR中代表此IRQ的位置位，并将IRR中相应位置零，表明此中断正在接受CPU的处理。同时，将它的编号写入中断向量寄存器IVR的低三位。这时，CPU还会送来第二个INTA信号，当收到此信号后，芯片将IVR中的内容，也就是此中断的中断号送上通向CPU的数据线。    图29 8259A的内部结构   1. 给这些中断号的处理向量初始化值：   要给这些中断号的处理向量初始化值，需要对8259A进行初始化编程，即向它写入四个初始化命令字ICW1-ICW4。这些命令字可以指定8259A的基本工作参数，如中断向量码的起始地址，工作模式，级联方式等。具体的编程步骤如下：  1.向8259A的命令寄存器写入ICW1，指定8259A是否级联，是否有ICW4等。  2.向8259A的数据寄存器写入ICW2，指定中断向量码的高5位（低3位由8259A自动提供）。  3.如果有级联，则向8259A的数据寄存器写入ICW3，指定主片和从片之间的连接方式。  4.如果有ICW4，则向8259A的数据寄存器写入ICW4，指定特殊全嵌套模式，缓冲模式，自动结束模式等。   1. **如何建立IDT，如何实现一个自定义的中断** 2. 如何建立IDT：  * **分配内存**：首先，你需要在内存中分配一块空间来存储IDT。这可以通过使用操作系统提供的内存分配函数或手动分配内存来完成。 * **定义IDT条目**：IDT是由一系列的描述符（Descriptor）组成的表格。每个描述符对应一个中断或异常类型，并包含了相应的处理程序的地址和其他相关信息。 * **设置IDT条目**：将中断处理程序的地址和其他相关信息设置到IDT的相应条目中。 * **加载IDT**：将IDT的地址加载到IDTR寄存器中，以告诉处理器IDT的位置和大小。  1. 如何实现一个自定义的中断：    * **定义中断处理程序**：首先需要定义要执行的中断处理程序。这可以是你自己编写的汇编或C代码，用于处理特定中断类型的操作。    * **初始化IDT**：IDT是一个存储中断处理程序地址的表格。你需要在内存中分配一块空间来存储IDT，并将其初始化为全零。    * **设置中断处理程序地址**：将你定义的中断处理程序的地址设置到IDT的相应条目中。每个中断类型都有一个唯一的中断向量值，用于索引IDT的相应条目。    * **加载IDT**：通过将IDT的地址加载到IDTR寄存器中，将IDT告诉处理器。    * **启用中断**：根据你的操作系统或应用程序的需求，启用或禁用特定的中断。 2. **如何控制时钟中断，为什么时钟中断时候，没有看到int的指令**   控制时钟中断一般情况下就由以下步骤：   1. 设置时钟中断向量：时钟中断的响应通常是通过设置一个中断向量来实现的。中断向量是一个指向特定中断处理程序的地址。在设置中断向量时，需要告诉计算机在发生时钟中断时跳转到哪个地址执行相应的处理代码。 2. 启用时钟中断：在启动系统或程序时，需要确保时钟中断被启用。这通常涉及设置相关的中断控制器或寄存器，使其能够接收和响应时钟中断请求。 3. 编写中断处理程序：当发生时钟中断时，控制权将转移到设置的中断处理程序的地址。在中断处理程序中，可以执行特定的操作，如更新系统时间、执行调度算法或处理其他与时间相关的任务。   在时钟中断的时候，没有int指令是因为处理器在中断发生时会自动执行相应的中断处理代码。"int"指令是用于软件触发和调用中断的指令，而时钟中断是由硬件定期触发的，在硬件层面上直接跳转到预设的中断处理程序，而不需要使用"int"指令来触发。这是因为中断处理是一种硬件级别的操作，处理器会自动进行相关的状态保存和恢复操作，以确保中断处理的正确性。   1. **简要解释一下IOPL的作用与基本原理**  * IOPL的作用   IOPL是I/O保护机制的关键之一，位于寄存器eflags的12、13位。  IOPL  图30 eflags结构  指令in、ins、out、outs、cli、sti只有在CPL≤IOPL的时候才能执行。前面说的指令被称为I/O敏感指令，如果低特权级的指令试图访问这些I/O敏感指令将会导致常规保护错误（#GP）   * IOPL的基本原理   处理器不限制0特权级程序的I/O访问，它总是允许的。但是，可以限制低特权级程序的I/O访问权限。这是很重要的，操作系统的功能之一是设备管理，它可能不希望应用程序拥有私自访问外设的能力。  可以改变IOPL的指令只有popf和iretd，但只有运行在ring0的程序才能将其改变。运行在低特权级下的程序无法改变IOPL，不过，如果试图那样做的话并不会产生任何异常，只是IOPL不会改变，仍然保持原样。  另一个与I/O操作特权级有关的概念是I/O位图。I/O位图基址是一个以TSS 的地址为基址的偏移，指向的便是I/O许可位图。之所以叫做位图，是因为它的每一位表示一个字节的端口地址是否可用。如果某一位为0，则表示此位对应的端口号可用，为1则不可用。由于每一个任务都可以有单独的TSS，所以每一个任务可以有它单独的I/O许可位图。  I/O位图与IOPL是或的关系，即只用满足一个条件即可。   |  |  |  | | --- | --- | --- | | 指令 | 功能 | 条件 | | cli | 清除IF位 | CPL≤IOPL | | sti | 设置IF位 | CPL≤IOPL | | in | 从I/O读数据 | CPL≤IOPL或I/O位图许可 | | ins | 从I/O读字符串 | CPL≤IOPL或I/O位图许可 | | out | 向I/O写数据 | CPL≤IOPL或I/O位图许可 | | outs | 向I/O写字符串 | CPL≤IOPL或I/O位图许可 |   **六．实验改进意见**  **王卓：**   1. 提供更详细的实验指导：在每个实验步骤中， 提供更详细的指导和说明， 比如预期结果等，这样能帮助我们更好地完成实验。 2. 介绍实验目的和背景：实验开始之前提供实验的目的和背景，解释一下为什么需要进行该实验以及其与操作系统的关系，这能帮助我们更好理解实验的意义和重要性。   **程子洋**：  错误排除指南：提供学生在遇到常见问题时进行自我排除的指南。这可以包括常见错误消息的解释以及如何解决这些问题的步骤。  **聂森：**  1. 可以先让同学阅读教材资料后再介绍实验内容和实验相关的知识，可以帮助同学更快更好的了解实验的目标和实验的原理及任务  2. 给出具体的实验指导获给出更加丰富的实验参考资料，可以帮助同学更快上手实验并在有问题时找到解决方式  **刘虓：**  如果能有详细的操作教学可以帮助学生更快地上手，并且希望老师能讲解汇编代码的一些关键部分，学生自己阅读汇编码容易忽略一些问题。 | | | | |
| 1. 各人实验贡献与体会（每人各自撰写） | | | | |
| **同学：刘虓：**  本次实验为本人独立完成大部分实验内容，并负责实验内容第3题，思考题第4题实验报告的撰写，并且在最后整合本组的实验报告。  本次实验过程中，我深入地分析、理解并掌握了：中断和异常的机制及其在保护模式下的实现、8259A、IDT、中断门的概念及它们在中断处理中的作用、时钟中断例程的调试、IOPL的作用和运行原理。同时，对汇编代码的阅读和分析过程，也对我自己汇编的语法和代码编写的知识和能力带来的极大的锻炼和提升。  **同学：程子洋**  此次实验为本人独立完成大部分实验内容，并主要负责实验内容第2题，思考题第2题实验报告的撰写。  实验体会：  通过学习中断和异常机制，以及调试8259A编程基本例程，我可以深入理解计算机系统的中断处理和外设管理。自定义中断和IDT的实现可以帮助我更好地理解中断向量和处理程序的映射。理解IOPL的概念有助于我理解操作系统如何管理特权级别以及如何保护系统的安全性。学习和实践这些主题是深入了解操作系统内部工作的重要一步。  感觉自己收获颇丰，极大地加深了我对中断与异常的理解。  **同学：王卓**  在本次实验中， 本人完成大部分实验内容， 负责实验内容的第1题、思考题的第1题和第5题的实验报告撰写。  本次实验主要研究保护模式下异常与中断处理机制，整体来说，实验不算过于难。通过对pmtest9a.asm的调试，我掌握了IDT的建立以及保护模式下的中断调用；通过对pmtest9.asm的调试，我知悉了时钟中断的触发机理。除此之外，通过阅读教材，我也掌握了8259A、IDT、中断门的概念及它们在中断处理中的作用、IOPL的作用与运行原理，并最后自己编写了一个中断处理函数。  总的来说，此次实验让我对中断异常的机制和其各种相关知识都有了一定程度的理解和自我的掌握，也进一步锻炼了我的汇编代码的编写。  **同学：聂森**  在本次实验中，本人完成全部题目，并主要负责实验代码编写和实验内容第四小问和实验思考第三小问实验报告的撰写。  在本次实验中，我更加深刻的认识到中断在实模式和保护模式下的不同，在实模式下中断的调用十分方便，但是保护模式下更加复杂，具体来说需要使用IDT并设置8259A。由于汇编的代码基础薄弱，在编写代码的过程中发生了很多错误，使得汇编没有通过，这个实属不应该。另外，由于基础知识没有深入了解，所以在实现中断的过程中，出现了一些意想不到的错误。最后通过网上查阅资料和翻看教材，我最终搞懂了这一切。可以说实践是最好的老师，只有自己亲手编写代码去实践才能真正的掌握原理！希望自己能在实践中获得更多成长。 | | | | |
| 1. 教师评语 | | | | |
|  | | | | |
| **教师评分（请填写好姓名、学号）** | | | | |
| 姓名 | | 学号 | 分数 | |
| 聂森 | | 2021302191536 |  | |
| 程子洋 | | 2021301051114 |  | |
| 王卓 | | 2021302191791 |  | |
| 刘虓 | | 2021302121234 |  | |
| 教师签名：  年月日 | | | | |