|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 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 | | 2024.9.16 |
| 实验名称 | 基本实验环境搭建 | | 实验周次 | | 第一周 |
| 姓名 | 学号 | | 专业 | | 班级 |
| 王亚鹏 | 2022302181161 | | 信息安全 | | 5班 |
| 杨依磊 | 2022302181159 | | 信息安全 | | 5班 |
| 杜泓波 | 2022302181162 | | 信息安全 | | 5班 |
| 侯名扬 | 2022302181165 | | 信息安全 | | 5班 |
| 1. 实验目的及实验内容   （要求掌握的知识；实验内容；原理分析） | | | | | |
| **总目标：**  1. 理解x86架构下的段式内存管理  2. 掌握实模式和保护模式下段式寻址的组织方式、关键数据结构、代码组织方式  3. 掌握实模式与保护模式的切换  4. 掌握特权级的概念，以及不同特权之间的转移  5. 了解调用门、任务门的基本概念  **实验内容**：  1．认真阅读章节资料，掌握什么是保护模式，弄清关键数据结构： GDT、descriptor、selector、GDTR， 及其之间关系，阅读 pm.inc文件中数据结构以及含义，写出对宏Descriptor的分析  2. 调试代码，/a/ 掌握从实模式到保护模式的基本方法，画出代码流程图，特别注意跳转问题，如果把跳转直接改成jmp offset，而不用selector:offset形式，会是什么结果，反汇编比较一下区别。  3. 调试代码，/b/，掌握GDT的构造与切换，从保护模式切换回实模式方法  4. 调试代码，/c/，掌握LDT切换  5. 调试代码，/d/掌握一致代码段、非一致代码段、数据段的权限 访问规则，掌握CPL、DPL、RPL之间关系，以及段间切换的基本方法  6. 调试代码，/e/掌握利用调用门进行特权级变换的转移的基本方法  7. GDT、Descriptor、Selector、GDTR结构，及其含义是什么？他 们的关联关系如何？pm.inc所定义的宏怎么使用？  8. 从实模式到保护模式，关键步骤有哪些？为什么要关中断？为 什么要打开A20地址线？从保护模式切换回实模式，又需要哪些 步骤？  9. 解释不同权限代码的切换原理，call, jmp，retf使用场景如何， 能够互换吗？  10. 动手改： ① 自定义添加1个GDT代码段、1个LDT代码段，GDT段内要对一个内 存数据结构写入一段字符串，然后LDT段内代码段功能为读取并打 印该GDT的内容； ② 自定义2个GDT代码段A、B，分属于不同特权级，功能自定义，要 求实现A-->B的跳转，以及B-->A的跳转。 | | | | | |
|  | | | | | |
| 1. 实验过程分析   （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） | | | | | |
| **实验环境**：  **X86 32位Linux环境**  **实验步骤：**  **1.GDT、LDT、Descriptor、Selector、GDTR的结构及其含义**  **1-1、GDT和LDT**  描述符表的长度可变，每个描述符的长度都是8B，最多可以包含8k个这样的描述符（段选择子时16位的，其中的13bit用来做index）。有两种描述符表，GDT和LDT。结构如下：    段描述符表存放在有操作系统维护的特殊数据结构中，由处理器的内存管理部件来引用。虚拟地址空间被分裂成两部分，一部分由GDT来映射，一部分由LDT映射。整个虚拟地址空间还有个段，一半是GDT映射的全局虚拟地址空间，一半是LDT映射的全局虚拟地址空间。系统发生任务切换会更新任务的LDT。  每个系统必须定义一个LDT，用于系统的所有任务和程序。可选择性定义若干个LDT。GDT本身不是一个段，而是线性地址空间的一个数据结构；GDT的线性基地址和长度必须加载到GDTR中。由于段描述符长度是8，多以GDT长度为8n-1.同时每个描述符的长度是8，所以GDT的基地址最好8字节对齐。  LDT表存放在LDT类型的段之中，此时GDT必须含有LDT的段描述符。访问LDT需要使用段描述符，为了减少访问LDT时的段转换次数，LDT的段描述符，段基址，段限长都要放在LDTR寄存器中。查找GDT在线性地址中的基地址，需要借助GDTR；而查找LDT相应基地址，需要的是GDT中的段描述符。LDT本身是一个段，而GDT不是。  **1-2、段选择子Selector**  16bit，指向段描述符表的段描述符。结构如下：  （1）段寄存器只有六个，一个程序可以定义很多段，但只能同时访问六个  （2）我们提供隐藏部分，也就是描述符缓冲，来减少对描述符表的引用次数。  （3）有缓存，必然涉及到缓存一致性的维护，我们在描述符表中的描述符修改过后立即更新缓存。  （4）有两类加载指令可以改变段寄存器的值：MOV（pop/lds/les/lss/lgs/lfs）等显示指令和CALL（JMP/RET/IRET/INTn/INTO/INT3）等隐式加载指令。  **1-3、Descriptor**  （1）GDT描述符:32bit段基地址（B2、B3、B4、B7），20Bit段限长（B0、B1、B6的低四位），12bit段属性（B5、B6的高四位）。结构如下：  因此，存储段描述符结构如下：  -------------------------------------------  ；segment Descriptor  ;for code and data  ;usage Descriptor Base,Limit,Attr  ;Base dd  ;Limit dd(Low 20bit are available)  ;Attr dw(Lower 4bit of higher type are 0)  %macro Descriptor 3  dw %2&0FFFFh  dw %1&0FFFFh  db (%1 >> 16)&0FFh ;注意，运算以后超过8b，但是会被db截断  dw (%3&0F0FFh) | ((%2 >> 8)&F0FFh)  db （%1>> 24)&0FFh  %endmacro;8字节  （2）定义了这样的门描述符结构：  ;for gate  ;usage Gate Selector Selector,Offset,Dcount,Attr  ; selector dw  ; offset dd  ; Dcount dw  ; Attr dw  %macro Gate 4  dw (%2&FFFFh)  dw %1  dw (%3 & 1Fh) | (（%4<< 8) & FF00h)  dw (%2 >> 16) & FFFFh  %endmarco  （3）根据IA-32的有关说明定义了如下的常量：  ;描述符类型说明:  \_LDT EQU 82h ; 局部描述符表段类型值  DA\_TaskGate EQU 85h ; 任务门类型值  DA\_386TSS EQU 89h ; 可用386任务状态段类型值  DA\_386CGate EQU 8Ch ; 386 调用门类型值  DA\_386IGate EQU 8Eh ; 386 中断门类型值  DA\_386TGate EQU 8Fh ; 386 陷阱门类型值  ;---------------------------------------------------  ;DA\_32 EQU 4000h ;32b段  DA\_DPL0 EQU 00h ; DPL = 0  DA\_DPL1 EQU 20h ; DPL = 1  DA\_DPL2 EQU 40h ; DPL = 2  DA\_DPL3 EQU 60h ; DPL = 3  ;数据段和代码段描述符的低8位  ;---------------------------------------------------------------------------  DA\_DR EQU 90h ; 存在的只读数据段类型值  DA\_DRW EQU 92h ; 存在的可读写数据段属性值  DA\_DRWA EQU 93h ; 存在的已访问可读写数据段类型值  DA\_C EQU 98h ; 存在的只执行代码段属性值  DA\_CR EQU 9Ah ; 存在的可执行可读代码段属性值  DA\_CCO EQU 9Ch ; 存在的只执行一致代码段属性值  DA\_CCOR EQU 9Eh ; 存在的可执行可读一致代码段属性值  ;选择子类型说明  ------------------------------------------  ;SA\_selector attribute  SA\_RPL0 equ 0;  SA\_RPL1 equ 1;  SA\_RPL2 equ 2;  SA\_RPL3 equ 3;  SA\_TIG equ 0;  SA\_TIG equ 4;  **1-4、GDTR**  1-1中有解释。  **2．从实模式到保护模式及代码流程图。**  **2-1、从实模式到保护模式的过程。**  （1）完成对段描述符的初始化  （2）装填gdtptr  （3）关中断  （4）打开A20  （5）设置cr0寄存器  （6）jmp切换  **2-2、代码分析**   1. 进入保护模式前，为什么要关中断？   由于实模式和保护模式的中断机制不同。实模式下，CPU使用中断向量来处理中断，这个表固定在内存的特定位置。保护模式下使用LDT。  CPU的寄存器和内存管理机制会改变。  防止任务切换。   1. 为什么要打开A20地址线？   扩展内存寻址达到4GB，避免内存回绕，确保系统的稳定性和兼容性   1. 从保护模式切换回实模式，需要哪些步骤？   |要从16位代码段返回，因此新增Normal描述符，在返回之前把选择子selectornormal加载到ds、es、ss  |关闭中断：清除标志寄存器FLAGS中的中断位IF  cli  |清空GDT  lgdt [gdtr\_null]  |更新段寄存器为0  mov ax,SelectorNormal  mov ds,ax  mov es,ax  mov fs,ax  mov gs,ax  mov ss,ax  |更新指令指针为实模式的代码位置  jmp SelectorCode16:0  |清除CR0寄存器的保护模式位  mov eax,cr0  and ax,0xFFFE  mov cr0,eax  |关闭A20地址线  in al,92h  and al 11111101b  out 92h,al  |打开中断  sti  |回到Dos  mov ax,4c00h  int 21h   1. 把跳转改成jmp offset，结果反汇编对比：   下图依次是jmp selector：offset和jmp offset的反汇编代码，发现跳转指令0075行和最后jmp指令的跳转位置不同  使用 `jmp selector:offset` 指令来进行跳转。这个跳转不仅改变了指令指针（EIP），也改变了代码段寄存器（CS），从而将CPU置于保护模式。这个操作完成后，CPU将开始以保护模式执行代码。  `jmp selector:offset` 用于在保护模式下执行跳转，而 `jmp offset` 仅在实模式下使用。因此在进行模式转换时会导致CPU继续在实模式下运行。  **2-3、代码流程图**  初始化寄存器ax，ds，es，ss，sp  ->计算并设置GDT  ->准备GDTR  ->加载GDTR到CPU  ->设置Cr0寄存器，启用保护模式  ->跳转到32位代码段，进入保护模式  **-**>在32位代码段中，设置视频段选择子到GS  ->计算屏幕显示位置  ->将字符“p”和颜色属性写入显存  ->进入无限循环，停止程序  **2-4、实验结果截图**    **3. GDT的构造与切换，从保护模式切换回实模式。**  如下图，并不像**2-4**中出现死循环而是出现dos提示符，证明回到实模式。具体步骤在**2-2**（3）中解释。    **4.LDT切换**  在[SECTION .S32]中打印完“In Protect Mode now”字符串后，出现一个红色的字符L。  20231016172158  **5、掌握一致代码段、非一致代码段、数据段的权限访问规则，掌握CPL、DPL、RPL之间关系，以及段间切换的基本方法。**  20231016172428  **6、掌握利用调用门进行特权级变换的转移的基本方法**  20231016172612  **7、从实模式到保护模式，关键步骤有哪些？为什么要关中断？为什么要打开A20地址线？从保护模式切换回实模式，又需要哪些步骤？**  详见2相关解释  **8、阐述不同权限代码的切换方法，call, jmp，retf使用场景如何，能够互换吗？**  8-1切换方法：   * 1. call指令：call指令用于调用一个过程或函数，并将控制权转移到目标代码段中的指定地址。当使用call指令时，处理器会将当前代码段的返回地址（即下一条指令的地址）压入堆栈，并跳转到目标代码段中的指定地址。call指令可以在不同的权限级别之间进行切换，例如从用户态（Ring 3）切换到内核态（Ring 0）。   2. jmp指令：jmp指令用于无条件跳转到目标代码段中的指定地址。与call指令不同，jmp指令不会将返回地址压入堆栈，因此无法直接实现权限级别的切换。但是，通过在目标代码段中设置适当的段选择子，可以实现从一个权限级别的代码段跳转到另一个权限级别的代码段。   3. retf指令：retf指令用于从过程或函数返回，并将控制权转移到调用者的代码段中的指定地址。retf指令会从堆栈中弹出返回地址，并跳转到该地址。与call指令类似，retf指令可以在不同的权限级别之间进行切换，例如从内核态（Ring 0）返回到用户态（Ring 3）。   8-2使用场景：   * 1. call指令：call指令通常用于调用子程序或函数，将控制权转移到另一个代码段，并在返回时继续执行调用指令的下一条指令。它常用于实现函数调用、子程序调用和中断处理等场景。   2. jmp指令：jmp指令通常用于实现条件跳转或无条件跳转，将控制权直接转移到目标代码段中的指定地址。它常用于实现循环、条件语句和跳转表等场景。   3. retf指令：retf指令通常用于从过程或函数返回，并将控制权转移到调用者的代码段中的指定地址。它常用于实现函数返回、中断返回和任务切换等场景。   这些指令在某些情况下可以互换使用。例如，可以使用Jmp指令实现从一个权限级别的代码段跳转到另一个权限级别的代码段，然后使用Retf指令返回到原来的权限级别  9.动手改  9-1、**自定义添加1个GDT代码段、1个LDT代码段，GDT段内要对一个内存数据结构写入一段字符串，然后LDT段内代码段功能为读取并打印该GDT的内容**  参考pmtest3.asm进行修改，将打印的字符串更替为“randi from GDT”。具体修改如下：   1. 修改[SECTION.data1]中的内容为   PMMessage:db “randi from GDT”, 0  StrTest: db “randi from GDT”, 0  （2）修改GDT代码段  在pmtest3.asm的[SECTION .S32]代码段基础上，call DispReturn和；Load LDT之间加上call Read和call Write    Read部分用来从内存中读取数据并显示在屏幕上，Write负责将字符串写入内存。    （3）LDT代码段    (4)结果如下：     1. 从实模式jmp到保护模式进入[SECTION .s32]，显示“randi from GDT   ”，[SECTION .s32]中通过lldt加载LDT的ldtr，然后跳转到[SECTION .la]；[SECTION .la]段显示字符串StrTest即“randi from GDT”，然后jmp回实模式。  9-2、**自定义2个GDT代码段A、B，分属于不同特权级，功能自定义，要求实现A-->B的跳转，以及B-->A的跳转。**  **方法一：**  参考实验代码pmtest5.asm,高特权级通过retf跳到低特权级，低特权级通过调用门跳回高特权级。即ring0->ring3->ring0->ring3d。  20231016172612  **方法二：**  通过设置高特权级代码段为一致性代码段实现低特权级到高特权级的跳转**。**   1. 定义dpl=3的数据段并定义段描述符和段选择符，初始化段选择符      1. 定义ring3代码段描述符和段选择符，同时初始化段描述符。 2. 编写ring3代码段，实现输出字符串，通过call跳转到ring0代码段      1. 定义ring0代码段描述符和选择符，初始化段选择符 2. 编写ring0代码段，实现输出字符串      1. 实验结果： | | | | | |
| 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．开始的准备工作没有做全，导致后续bug频出，需要先下载freedos，并修改bochsrc**   1. 下载freedos，将解压后的a.img命名为freedos.img并复制到工作目录下，用bximage生成一个软盘映像pm.img     （2）修改bochsrc    （3）启动bochs，待freedos启动完毕后格式化B盘  20231016161223  这里的B盘为我们的pm.img镜像文件，格式化操作相当于将其转化为FAT16文件系统，我们通过终端查看pm.img内容时，可以看到其中已经写入了一些文件系统相关的东西。    （4）修改pmtest1.asm代码中的07c00h为0100h，并重新编译  （5）命令如下：  sudo mount -o loop pm.img /mnt/floppy  sudo cp pmtest1.com /mnt/floppy/  sudo umount /mnt/floppy  （6）重启bochs到B盘运行  20231016161928  **2.对于特权级之间跳转理解不当。**    不同特权级别段之间的代码转移 ，本质上是一个添加了属性的特殊入口地址。高特权级通过retf跳到低特权级，低特权级通过调用门跳回高特权级。 | | | | | |
| 1. 各人实验贡献与体会（每人各自撰写） | | | | | |
| 每个人均全程参与实验  **侯名扬：**参与实验内容，撰写实验报告。并与杜泓波合作提供动手写的第二个代码部分内容。通过本次实验，充分掌握实模式，保护模式的切换；LDT切换，不同特权级之间的跳转，掌握基本的调用门的方法。掌握一致代码段，非一致代码段，数据段的访问规则，学会在ubuntu环境下使用freedos。弄清关键数据结构： GDT、descriptor、selector、GDTR， 及其之间关系。通过反汇编对两种跳转方式生成的代码进行了比较。  **王亚鹏：**参与实验内容，撰写实验报告。完成了各个代码的调试并与杨依磊合作提供动手写的第一个代码部分内容。**段切换与保护模式的实现是这次实验最大的收获。还有对于bochs和freedos的使用方法有了进一步了解。**  **本实验的关键环节在于从实模式切换到保护模式的过程中加载GDT和LDT**，并通过适当的长跳转切换到不同的段。在代码实现中，使用lgdt指令加载GDT，lldt指令加载LDT，这些操作非常直观地展示了段式内存管理机制在实际程序中的应用。此外，段选择子在切换不同段时起到了关键作用，它们指向具体的段描述符，并通过组合索引、TI和RPL字段来定位段。在实验中，我实现了将一段字符串写入GDT的数据段，然后通过LDT代码段读取该字符串并显示到屏幕上。这个过程帮助我加深了对内存寻址的理解。在32位模式下，字符串的存取需要根据段的基地址和偏移量进行精确的地址计算。  **实验中的另一个关键部分是实模式与保护模式的切换。**在进入保护模式前，需要关闭中断并启用A20地址线，然后设置CR0寄存器的PE位进入保护模式。实验让我更清楚地认识到进入保护模式的步骤和实模式与保护模式的区别。而在实验结束时，通过从保护模式跳回实模式，实现了两种模式的来回切换，这一部分强化了我对CPU模式切换机制的理解。  **杨依磊：**与王亚鹏合作提供动手写的第一个动手改代码部分内容,对修改跳转内容前后的a文件夹中代码编译后反汇编进行比较，学会了freedos的使用，完成了b,c,d,e四个文件夹中的代码调试，在调试过程中掌握了实模式到保护模式，以及保护模式到实模式的切换方法，LDT切换，不同特权级间跳转，基本的调用门方法，以及不同段的访问规则。弄清 GDT,descriptor,selector,GDTR各自的概念及其关系。  **杜泓波：**与侯名扬合作提供动手写的第二个代码部分内容。本次实验我了解了保护模式及其数据结构GDT、descriptor、selector和GDTR。通过分析 pm.inc 文件我对宏 Descriptor 的构造有了清晰的认识，了解到它是如何将基址、段限制和访问权限结合在一起的，从而形成有效的段描述符。  在调试代码的过程中，我学习了从实模式到保护模式的基本转换方法，特别是如何使用正确的跳转方式。特别需要注意的是，如果将跳转方式直接改为 jmp offset 而不是使用 selector:offset，处理器会继续在实模式下执行代码，无法正确访问保护模式中的新段，从而导致程序崩溃或产生意外结果。我通过反汇编对两种跳转方式生成的代码进行了比较，验证了选择子在模式切换中的重要性。  在掌握 GDT 的构造与切换时，我进一步理解了如何加载新的 GDT，并使用指令切换回实模式的步骤。调试 LDT 切换的代码让我看到了局部描述符表在多任务环境中的作用，它能够为每个任务提供独立的段描述符，从而实现任务之间的隔离与资源共享。  学习了一致代码段、非一致代码段以及数据段的权限访问规则，掌握了当前CPL、DPL和RPL间的关系。在段间切换的基本方法中，我意识到只有在 CPL <= DPL 和 RPL <= CPL 的情况下，才能成功访问所请求的段。这种权限控制机制是确保系统安全和稳定的重要环节。  我学习了如何利用调用门进行特权级变换，这一机制让我认识到如何安全地从低特权级代码调用高特权级服务，确保系统的安全性与稳定性。整体而言，这次实验不仅加深了我对保护模式的理解，也让我熟悉了相关的编程调试技能，为我今后的学习和研究奠定了坚实的基础。我希望在以后的课程中，能够继续探索和应用这些知识，进一步提升自己对计算机系统和操作系统的理解。 | | | | | |
|  | | | | | |
| 1. 教师评语   （实验报告的考评：依据实验内容完整度、实验步骤清晰度、实验结果与分析正确性、实验心得与思考的全面性、实验报告文档的规范性等五个维度综合考评）   |  |  | | --- | --- | | 85-100 | * 实验内容完整或者有超出课程实验大纲的内容； * 实验步骤详尽，能够体现完整的实验过程； * 实验结果正确且实验数据分析得当； * 实验心得与思考全面并且有自己的独立思考； * 实验报告文档规范、排版整齐。 | | 75-84 | * 实验内容较为完整； * 实验步骤较为详尽，能够体现实验过程； * 实验结果正确且实验数据分析较为得当； * 实验心得与思考全面； * 实验报告文档规范、排版较为整齐。 | | 60-74 | * 实验内容有缺失； * 实验步骤不够详尽，不能够体现完整的实验过程； * 实验结果部分正确； * 实验心得与思考无或者不够深入； * 实验报告文档规范性有待增强。 | | 60以下 | * 实验内容严重缺失、实验态度不够端正 * 实验步骤不够详尽，不能够体现完整的实验过程； * 实验结果部分正确； * 实验心得与思考无或者不够深入； * 实验报告文档规范性有待增强。 | | | | | | |
|  | | | | | |
| **教师评分（请填写好姓名、学号）** | | 学号 | | 分数 | |
| 王亚鹏 | | 2022302181161 | |  | |
| 杨依磊 | | 2022302181159 | |  | |
| 侯名扬 | | 2022302181165 | |  | |
| 杜泓波 | | 2022302181162 | |  | |
|  | | | | | |
| 教师签名：  年 月 日 | | | | | |