Linux内存管理实验

一、实验内容：

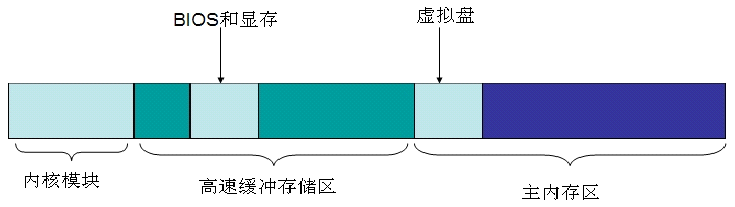
1.利用boches观测 linux0.11 下的 GDT 表和 LDT 表内容。

2.利用bochs观测 linux0.11 下的内存地址映射过程以及分页机制的实现。

3.利用**bochs**修改虚拟地址所对应的物理内存中存放的数值，观测程序运行情况的变化。

二、Linux内存管理机制分析

1.物理内存使用划分：为了有效使用物理内存，**Linux**将内存划分为几个功能区域，其中包括：内核模块区，高速缓冲区，主内存区，如下图：

 2.地址映射：

逻辑地址：该地址是指由程序产生并与代码段相关的偏移地址，分段分页机制对于程序员是透明的，程序员仅需和逻辑地址打交道。

线性地址：是指由逻辑地址到物理地址之间变换的中间层地址。程序代码会产生逻辑地址（即段内偏移地址），加上相应的段基址就产生了一个线性地址。若启用分页机制，那么线性地址还要经过一系列变换生成物理地址，若没有启用分页机制，则线性地址就是物理地址（在实地址模式下，线性地址就是物理地址，但是线性地址加大了系统的不安全因素）。

物理地址：是指出现在cpu外部地址总线上的寻址物理内存的地址信号，即地址变换的最终结果。当采用分页机制时，线性地址通过页目录，页表中的项来变换成物理地址，否则线性地址就直接是物理地址。

3.段描述符和段选择符：

段描述符：

i386虚模式下的地址是32位，而段寄存器却只有16位，因此处理器要求系统在内存中创建一系列的表来存放每个段的首地址。

这些表中的内容用来描述一个段的信息，因此这些表叫段描述符表，包括（GDT：全局描述符表，LDT：局部描述符表，IDT：中断描述符表）表中的内容就称为段描述符。

而程序的逻辑地址就是用段和段内的偏移地址表示。在linux中，程序的逻辑地址到线性地址的变换就是使用了cpu的全局描述符表GDT和局部描述符表LDT。由GDT映射的地址空间称为全局地址空间，由LDT映射的地址空间称为局部地址空间，其中LDT是GDT的二级表，即GDT中存储了LDT的地址。

段选择符：

为了和8086兼容，i386虚模式下的程序仍然用段寄存器来访问段，但是由于段描述符所在的地址为32位，而段寄存器只有16位，所以它只能作为段描述符表内的索引（即表内偏移来使用）。即用描述符表寄存器（GDTR）加上段寄存器中的偏移量来形成描述符的32位地址。因为段寄存器中存放的是索引，所以我们把段寄存器称为段选择符。

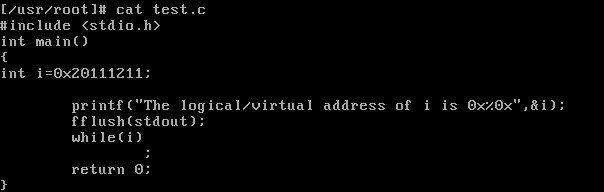
3.Linux中的分页机制：

**Linux**的内存分页管理机制是通过页目录表和内存表所组成的二级表进行，其运行过程如下图所示。



三、实验步骤及结果

1.搭建linux实验平台，并在搭建好的平台中用vi编辑器编写一个简单C程序，该程序的功能是打印局部变量**i**的逻辑地址，并使得进程能够一直处于运行状态。截图如下：



运行结果如下：

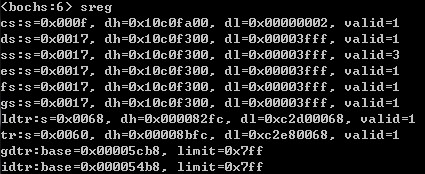
C:\Users\lenovo\AppData\Roaming\Tencent\Users\1197066450\QQ\WinTemp\RichOle\IHPOJ2TH86}9A`U96L7E7MW.jpg

在bochs命令窗口输入“ctrl+c“使得系统进入调试状态，则此时在bochs显示窗口会出现暂停，可以在命令窗口输入命令，并调试系统。在命令窗口中输入命令“u/7”反汇编出当前指令。如下图所示：



2.利用boches观测linux0.11下的GDT表和LDT表内容：

1)继续在调试窗口输入“sreg”命令，获取数据段，ldtr段选择符，gdtr寄存器。然后通过该信息，可以找到我们所需要的GDT表和LDT表地址。如下图所示：



**2)**由ldtr的值转换成二进制表示形式：0x0068=0000000001101000（二进制），表示LDT表存放在GDT表的1101(二进制)=13（十进制）号位置。又知道GDT表的基地址是gdtr=0x00005cb8.因此可以用命令“xp/2w 0x00005cc8+13\*8**”**来查找该进程对应的LDT表的基地址。如下图所示：

C:\Users\lenovo\AppData\Roaming\Tencent\Users\1197066450\QQ\WinTemp\RichOle\AY$K]%7Y$PMX9ZD0~E(YL4S.jpg

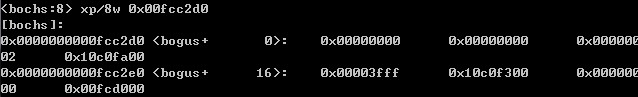
由上面显示的结果可得出LDT的地址为：0x00fcc2d0.

3）利用“xp/8w 0x00faa2d0**”**命令打印出LDT表中的内容，

由命令“sreg”可知ds段的值为0x0017=0000000000010111（二进制），故可知该段基址在LDT表中的索引值为10（二进制）=2（十进制）

所以LDT表中的第3个段描述符（从0开始编号）就是要寻找的ds段的基址.因此可以查找打印内容第三项，并将其内容与先前ds段的信息进行比较，从而验证LDT表和GDT表的功能。

如下图所示：



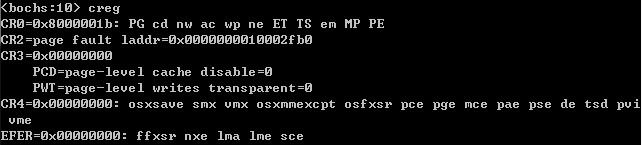
由LDT的内容可得出ds段的基址为：0x10000000.

4）由ds段的段基址+段内偏移，就可以获取线性地址，段内偏移为0x3fffef4。根据刚才汇编指令中变量i的线性地址ds：0x3004，我们可以输入命令**”**calc ds:0x3004**”**验证这个结果。

C:\Users\lenovo\AppData\Roaming\Tencent\Users\1197066450\QQ\WinTemp\RichOle\2M{UNW@E_[QT~6RW8_3`39L.jpg

3.利用bochs观测linux0.11下的内存地址映射过程以及分页机制的实现

1)输入**“**creg**“**指令来查看CR3寄存器的值，从而获取页目录表基地址。如下图所示：



可以看出页目录表的基地址为0。

10011111111111111111011110100

2)取线性地址ds:0x3004然后经过页变换映射后可知对应的页目录号为79，页号是1023，页内偏移量是**3828**。

故可以输入命令**”**xp /w 0+79\*4**”**来查看页目录表的第**79**项内容。如下如所示：

C:\Users\lenovo\AppData\Roaming\Tencent\Users\1197066450\QQ\WinTemp\RichOle\DC~%CE[ME]N609J74)}Z$ZC.jpg

3）输入命令**”**xp /w 0x00fb0000+1023\*4**“**。然后得到第1024个页表项的地址。则得到如下结果：

C:\Users\lenovo\AppData\Roaming\Tencent\Users\1197066450\QQ\WinTemp\RichOle\USU@W)13W`{HKWG~%`~~)SY.jpg

用以下命令则可得出逻辑地址所映射的物理地址：

C:\Users\lenovo\AppData\Roaming\Tencent\Users\1197066450\QQ\WinTemp\RichOle\F6F5KIGW6O{(E(%C{ZCBJRV.jpg

C:\Users\lenovo\AppData\Roaming\Tencent\Users\1197066450\QQ\WinTemp\RichOle\N%TJVY@K]VY)WWA{%5]JLDX.jpg

3.利用**bochs**修改虚拟地址所对应的物理内存中存放的数值，观测程序运行情况的变化：

1)利用setpmem 0x00fae004 4 0命令，将物理地址0x00fae004开始的四个字节内存置为0，然后输入c，让虚拟机继续运行

C:\Users\lenovo\AppData\Roaming\Tencent\Users\1197066450\QQ\WinTemp\RichOle\D%0HO7AN$LPK8691Y9Z8U}I.jpg

2)由虚拟机运行情况可以得出，我们对物理内存的改动起了作用，先前的**while(i)**由于**i**不为**0**得以不断执行，现在，**i**置**0,**则循环不再执行，程序正常退出，显示结果如下：

