电子科技大学 计算机科学与工程学院

实验指导书

（实验）课程名称： 计算机操作系统

电子科技大学教务处制表

# 内存地址转换实验

* 实验所属系列：操作系统课程实验
* 实验对象：本科
* 相关课程及专业：计算机操作系统，C语言，数据结构；计算机专业
* 实验类型：配套上机
* 实验时数： 2学时

## 1. 实验目的

（1）掌握计算机的寻址过程

（2）掌握页式地址地址转换过程

（3）掌握计算机各种寄存器的用法

## 2. 实验内容

本实验运行一个设置了全局变量的循环程序，通过查看段寄存器，LDT表，GDT表等信息，经过一系列段、页地址转换，找到程序中该全局变量的物理地址。

## 3. 实验环境

Linux内核（0.11）+Bochs虚拟机

## 4. 实验相关知识

4.1 逻辑地址到线性地址的转换

逻辑地址：Intel段式管理中：，“一个逻辑地址，是由一个段标识符加上一个指定段内相对地址的偏移量，表示为 [段标识符：段内偏移量]。”

段标识符: 也称为段选择符，段标识符是由一个16位长的字段组成，其中前13位是一个索引号。后面3位包含一些硬件细节：



图1 段选择符

索引号： 可以看作是段的编号，也可以看做是相关段描述符在段表中的索引位置。系统中的段表有两类：GDT和LDT。

GDT：全局段描述符表，整个系统一个，GDT表中存放了共享段的描述符，以及LDT的描述符（每个LDT本身被看作一个段）

LDT：局部段描述符表，每个进程一个，进程内部的各个段的描述符，就放在LDT中。

T1字段：Intel设计思想是：一些全局的段描述符，就放在“全局段描述符表(GDT)”中，一些局部的，例如每个进程自己的，就放在所谓的“局部段描述符表(LDT)”中。那究竟什么时候该用GDT，什么时候该用LDT呢？这是由段选择符中的T1字段表示的，T1=0，表示相应的段描述符在GDT中，T1=1表示表示相应的段描述符在LDT中。

段描述符：具体描述了一个段。在段表中，存放了很多段描述符。我们可以通过段标识符的前13位，直接在段描述符表中找到一个具体的段描述符，也就是说，段标识符的前13位是相关段描述符在段表中的索引位置。

段描述符

索引号

GDT或LDT

图2 GDT或LDT示例

每一个段描述符由8个字节组成，如图3：

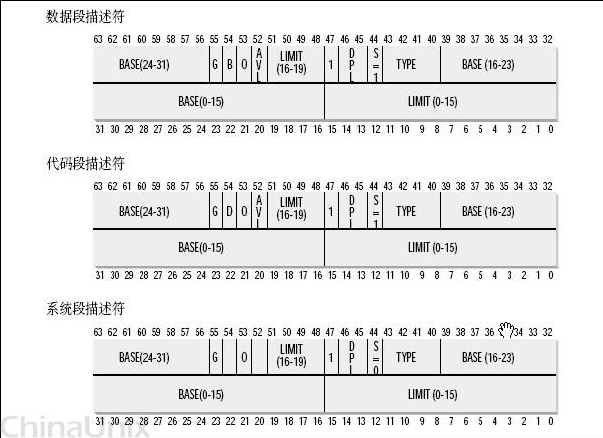


图3 段描述符

Base字段：它描述了一个段的开始位置：段基址。

相关寄存器：

GDTR：存放GDT在内存中的起始地址和大小

LDTR：分两种情况：

1. 当段选择符中的T1=1时，表示段描述符存放在LDT中，如何找到LDT呢，LDT本身也被看作一个段，LDT的起始地址存放在GDT中，此时LDTR存放的就是LDT在GDT中的索引。这也是本实验关注的情况。
2. 当段选择符中的T1=0时，表示段描述符存放在GDT中，通过GDTR找到GDT，此时LDTR存放的是LDT的起始地址，当T1=0时，不涉及对LDT和LDTR的使用。

段选择符：如在DS，SS等寄存器内存储，取高13位作为在相应段表（如上例中的DS的高13位为对应段在LDT）中的索引。

线性地址： 段标识符用来标明一个段的编号，具体的，我们需要通过段的编号，查找段表，来获得这个段的起始地址，即段基址。段基地址＋段内偏移量，就得到线性地址。

从逻辑地址到线性地址的转换过程，如图4所示（以T1=1为例，此时从段选择符中分离出段描述符和T1字段，T1=1，表明段描述符存放在LDT中）；

（1）从GDTR中获得GDT的地址，从LDTR中获得LDT在GDT中的偏移量，查找GDT，从中获取LDT的起始地址；

（2）从DS中的高13位获取DS段在LDT中索引位置，查找LDT，获取DS段的段描述符，从而获取DS段的基地址；

（3）根据DS段的基地址＋段内偏移量，获取所需单元的线性地址。

GDT

LDT

GDTR

LDTR

DS

段基址

段内偏移量

线性地址

(1)

(2)

(3)

图4 逻辑地址到线性地址的转换

**4.2 线性地址到物理地址的转换**

物理地址：分段是面向用户，而分页则是面向系统，以提高内存的利用率，简言之，内存空间是按照分页来管理的。一个32位的机器，支持的内存空间是4G，在页面大小为4KB的情况下，如果采用二级分页管理方式，线性地址结构如图5所示。

每一个32位的线性地址被划分为三部份， 页目录索引(10位)：页表索引(10位)：偏移(12位，因为页面大小为4K)。最终，我们需要根据线性地址，来获得物理地址。

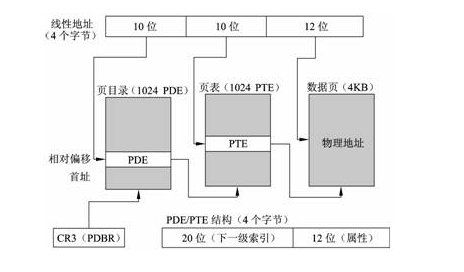


图5 线性地址结构

将线性地址转换成物理地址的步骤：

(1)、因为页目录表的地址放在CPU的cr3寄存器中，因此首先从cr3中取出进程的页目录表地址（操作系统负责在调度进程的时候，已经把这个地址装入对应寄存器）；

(2)、根据线性地址前十位，在页目录表中，找到对应的索引项，因为引入了二级管理模式，页目录中的项，不是页的地址，而是一个页表的起始地址。

(3)、查找页表，根据线性地址的中间十位，在页表中找到数据页的起始地址；

(4)、将页的起始地址与页内偏移量（即线性地址中最后12位）相加，得到最终我们想要的物理地址；

**5. 实验参考步骤**

（1）编写 实验使用的示例程序:

#include <stdio.h>

int j = 0x123456;

int main()

{

printf("the address of j is 0x%x\n", &j);

while(j);

printf("program terminated normally!\n");

return 0;

}

（2）理解X86计算机的寻址机制，理解全局描述符表GDT，局部描述符表LDT等数据结构的内容。

（3）使用sreg查看查看GDTR，LDTR，DS等寄存器内容，了解寄存器的数据格式。

（4）根据寄存器和相关的数据结构，计算变量j的线性地址。

（5）使用creg查看控制寄存器信息

（6）根据线性地址和页内偏移，基于页式地址转换，计算变量j在内存中的物理地址。

（7）根据j的物理地址，修改j的值，将j变为0，从而使得上述程序能够结束循环。

另参详细实验步骤

**6. 实验报告**

**独立完成，实验报告要求给出具体的实验原理分析，实验步骤描述中要有相关步骤的截图说明。**