# 3.实验项目三：Nachos内存管理实验

## 3.1实验目的：

完成Nachos的内存管理模块的扩展，掌握操作系统中虚拟内存管理的扩展实现。

## 3.2实验环境：

描述所用实验环境，包括计算机硬件和软件资源的情况，如选用的操作系统、机器配置、编译器版本等。

[在以下Add你的描述信息]

## 3.3实验内容：

对实践过程的详细说明，如对Nachos平台的哪些代码进行了什么样的修改，采用了何种算法或思想等。

List核心的代码，如以文件为单位一一进行描述。

可以结合适当的流程图或者类图来辅助描述。

[在以下Add你的描述信息]

1. **修改addrSpace.h与addrSpace.cc**

修改addrSpace.h中addrSpace类的定义，增加s\_reg和FileAddr分别用于存储寄存器的值与加载程序的元信息

修改addrSpace的构造器，使得其不再构造时申请页表，而在加载程序的load时再申请页表

同时修改addrSpace的析构函数，改变其地址空间所占用的物理页的全局页表项

在load函数中申请页表，使用machine.h中的findFreeFrame来寻找一个空闲的页。如果没有，该物理页暂时被置为-1

存储当前加载的程序的元信息

修改读入程序的代码，将虚拟地址进行转换。注意，这里的读取是逐字节的以避免某一字节处于没有分配物理页的位置

修改restoreState恢复现场方法和saveState存储现场方法，添加将暂存的寄存器内容恢复的代码和将寄存器中的值暂存的代码

**重要的一步：将Execute方法中的restoreState方法注释掉，改为直接赋值页表和其他信息。**因为在execute之前，需要初始化寄存器，如pc，stack等，但是如果使用了restore方法，将会把一系列0值覆盖掉初始化的值，导致错误。

1. **machine.h与machine.cc的修改**

首先在machine.h中添加全局页表项数据结构的声明

RefPageTable是为了在发生缺页时，将之前引用牺牲页的地址空间的虚拟页表对应项置为失效，因为它的空间已经被其他的虚拟页占领。

声明全局页表，寻找空闲物理页方法，利用LRU算法寻找牺牲页的算法和加载程序元信息等数据项

然后修改Machine类的构造器，使得其初始化全局页表项

在machine.cc中添加对应方法的实现：

LRU算法是比较时间戳useStamp来得到最少使用的牺牲页的物理页号的。

寻找物理内存中的空闲Frame的方法是查看全局页表GlobalPageTable中的引用项是否为空来判断是否已经由虚拟页引用本物理页，如果找不到将会返回-1

1. **修改translate.h与translate.cc**

首先修改translate.h中TranslationEntry类的定义，增加存储加载程序名称的数据项

然后修改translate.cc中readMem方法和writeMem方法，使得其在发生缺页中引发缺页中断的同时再度读取和写入

修改Translate方法中，对缺页中断的判定。在本实现中，如果程序页表对应项的valid置为false或者物理页分配为-1.代表缺页

最后，如果每次translate访问到某一个物理页，则将对应的物理页的时间戳加1代表最近使用过一

1. **exception.cc的修改**

首先在exceptionHandler中添加缺页对应的case：PageFaultException。声明变量，从BadVAddrReg寄存器中读取缺页的虚拟地址，计算缺页地址的虚拟页号等

然后先寻找是否有空闲物理页，如果没有再使用LRU算法寻找

如果是利用LRU寻找牺牲页，则应该先将牺牲页写回磁盘。写回时，先判断该页是否被修改，即查看dirty变量，然后读取程序的元信息以便从磁盘读入文件

然后，逐字节的，将本虚拟页对应的信息从磁盘中读入内存。由于不同的虚拟地址可能对应到不同的读取位置，因此需要使用程序元信息来判断是代码部分还是数据部分的信息。该位置的计算公式为（代码部分为例）：

当前读取的虚拟地址 - 代码起始虚拟地址 + 代码在磁盘中的起始位置 = 当前虚拟地址代码在磁盘中的位置

最后，更新全局页表，将被替换的虚拟页的valid置为false，同时更新程序页表，将对应的物理页置为牺牲页的页号，valid置为true等。最后，释放openfile的空间，case结束

1. **main.cc,add.c与ksyscall.h中的修改**

在threads/main.cc中，如ppt上所示，改为申请两个空间，但是只运行第二个地址空间的程序

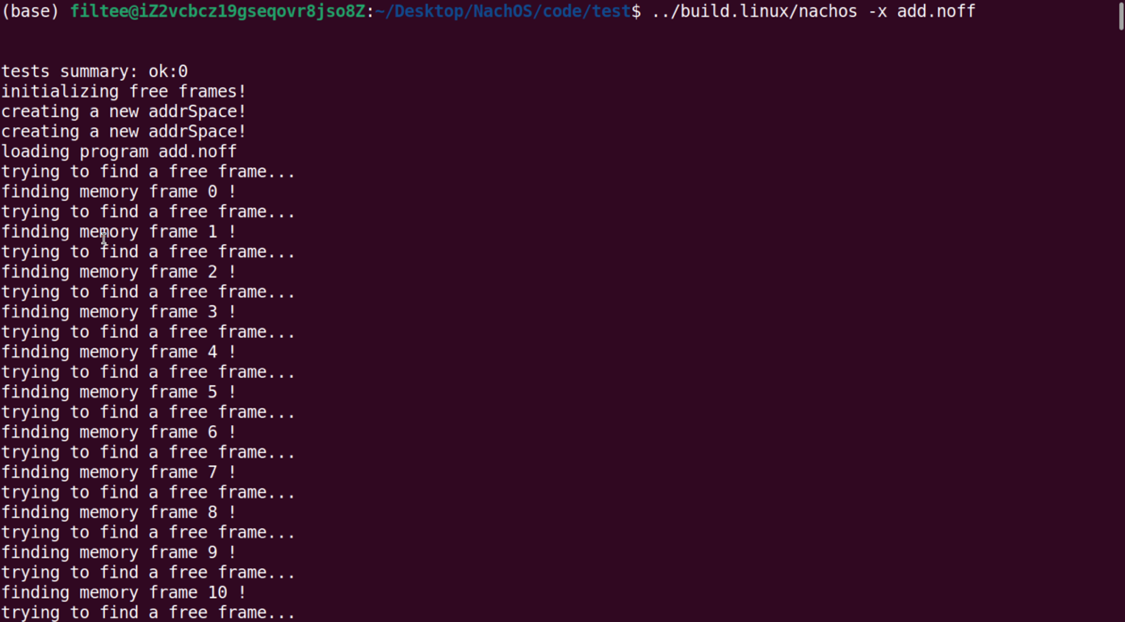
同时，修改code/test/add.c，使其更复杂一些以便测试

最后，修改usrprog/ksyscall.h文件，使得系统调用的结果可见

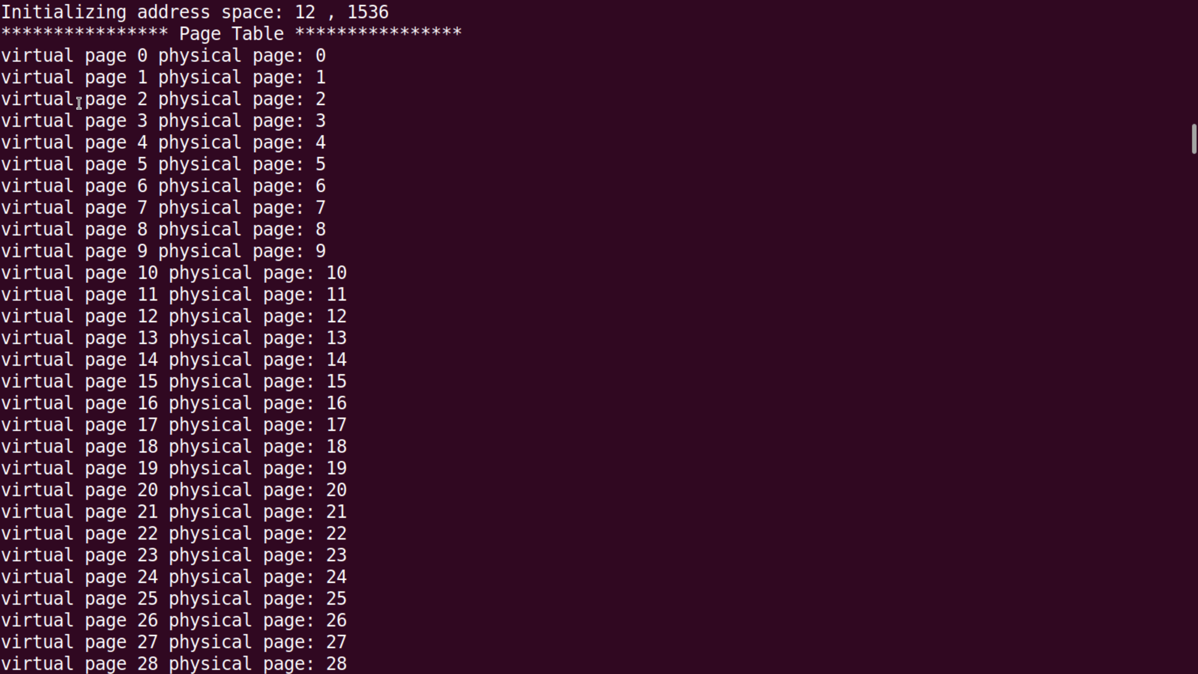
## 3.4实验结果：

截图说明运行效果，并且辅助相关描述信息。

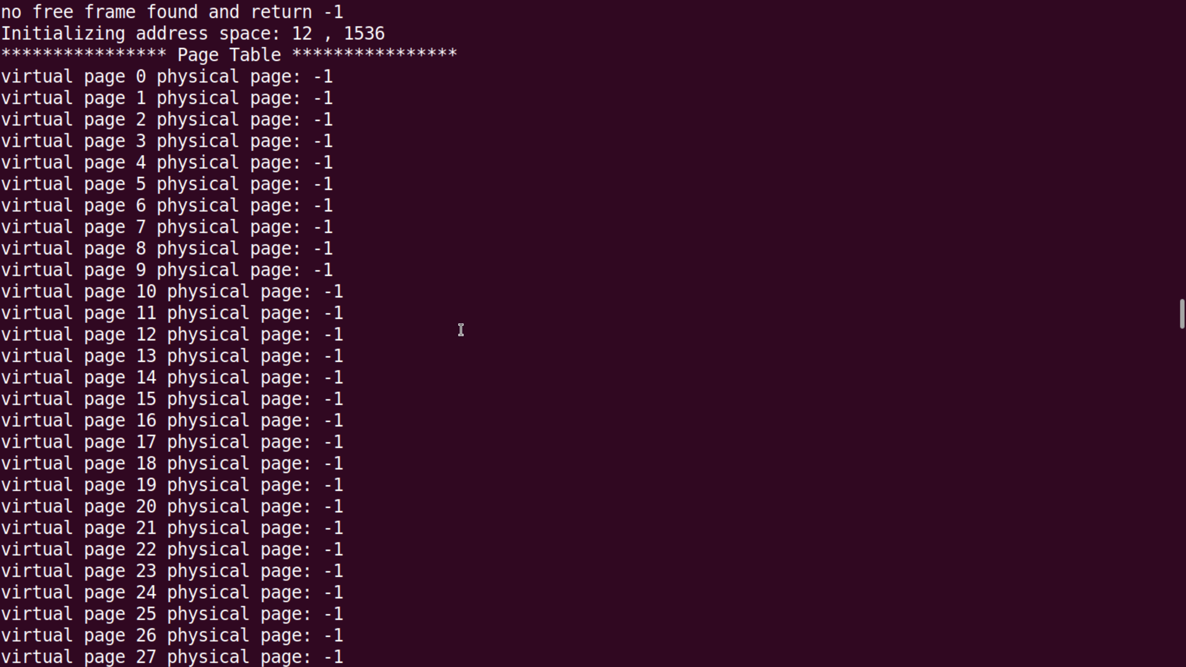
1. **使用./nachos -x ../test/add.noff 指令运行add.noff**



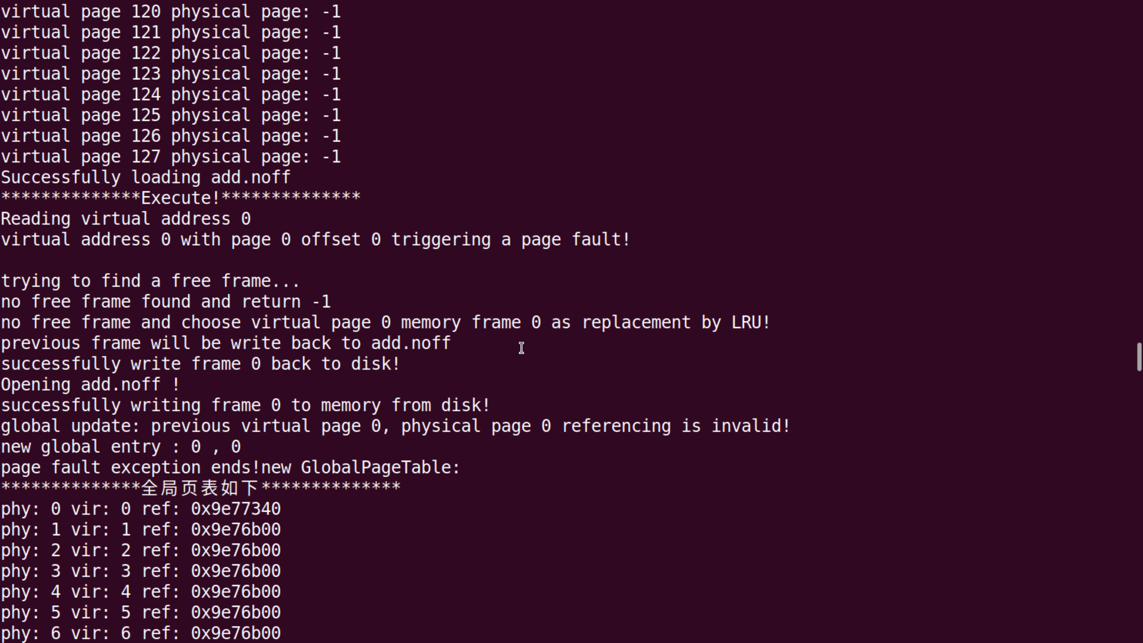
第一个地址空间生成的页表：



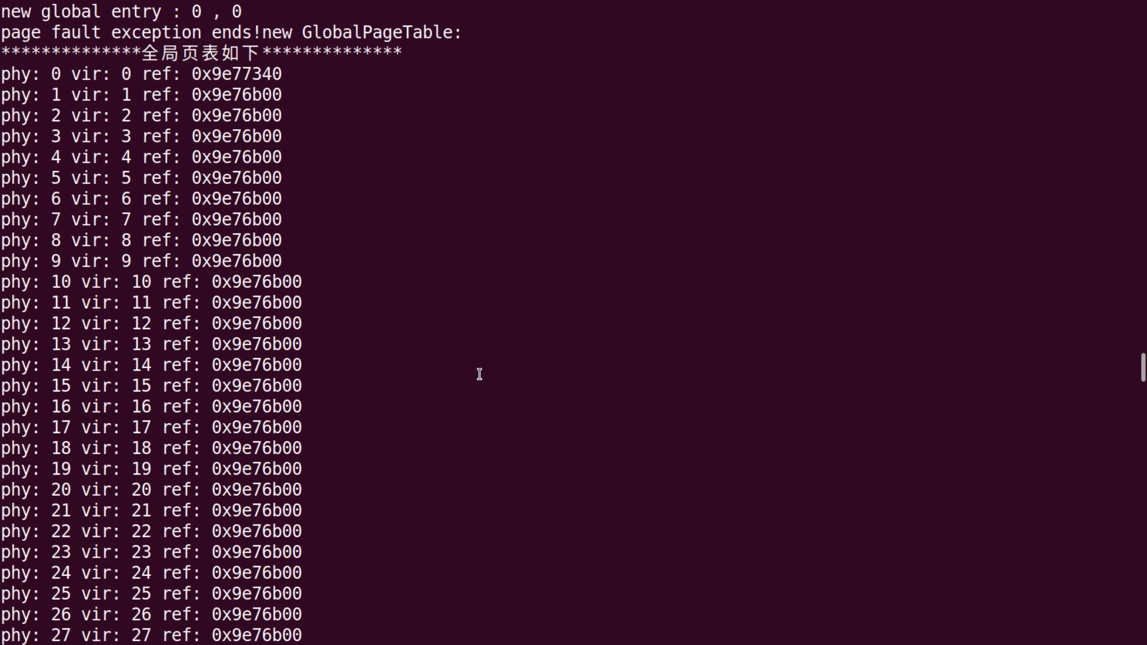
第二个地址空间运行时发生缺页：



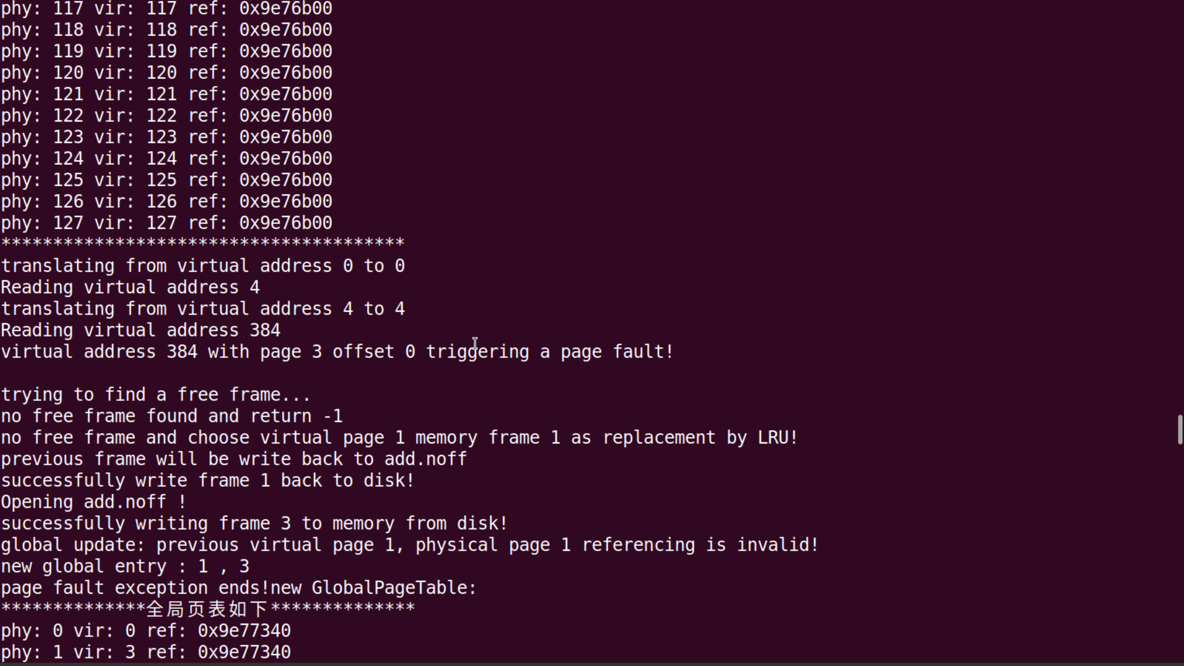
通过LRU算法找到牺牲页：



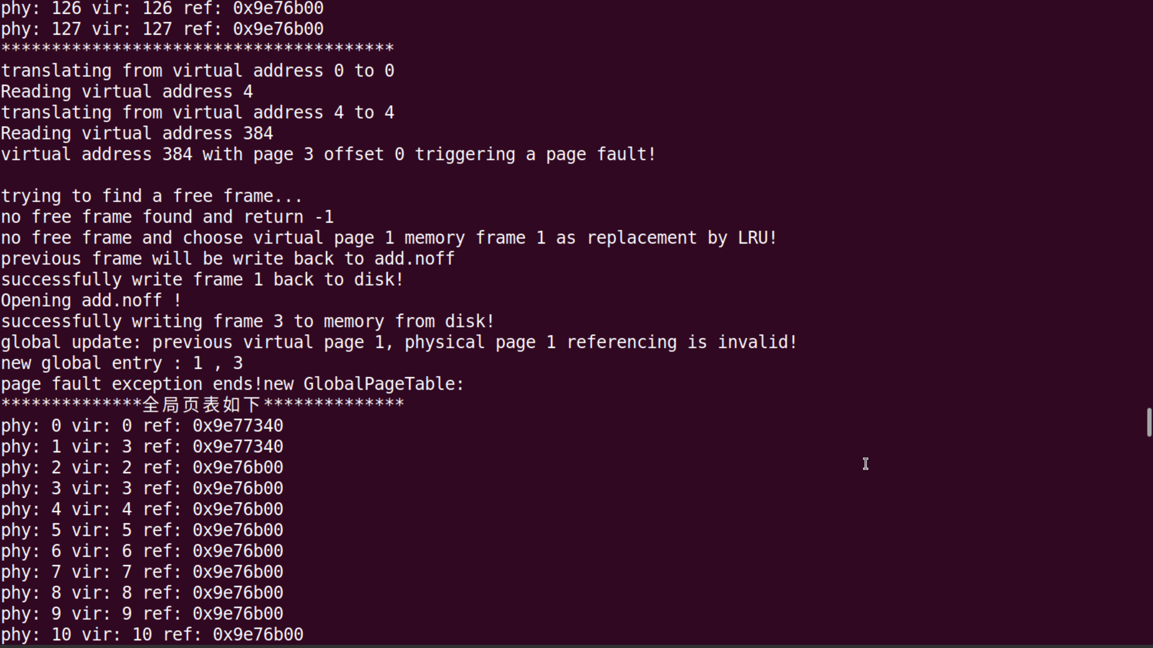
缺页中断处理后，全局页表更新：



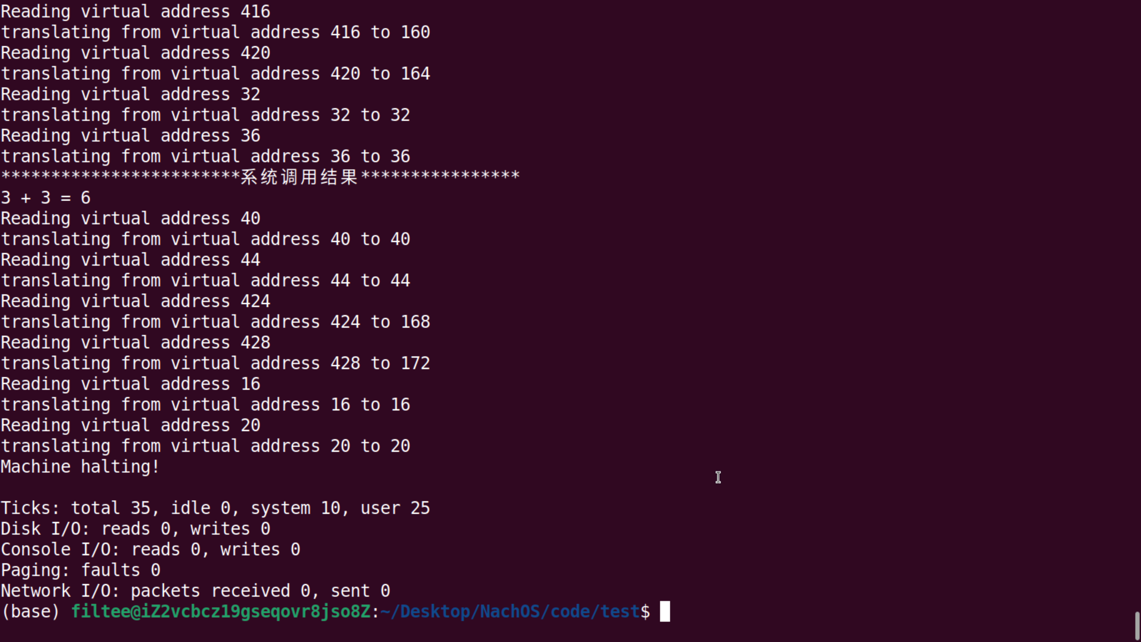
原来发生缺页中断的虚拟地址0已经被正确转换为物理地址0了



再次发生缺页中断，处理方式同上：



最后，程序打印出正常结果，并正常退出：



## 3.5实验总结：

总结本实验的完成情况，包括代码是否编写完成，是否调试通过，能否正常运行，实现了实验要求中要求的哪些项（对实验要求的满足程度；）；

总结本实验所涉及到的知识点。

💩实验，狗屎课；毁我青春，害我年华