

**计算机与信息 学院实验报告**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 实验课程： | 操作系统实验 | | | | |
| 实验编号： | 实验五 | | | | |
| 实验名称： | 地址变换机构及其变换方法 | | | | |
| 实验人员： | 学号 | 18111207248 | | | |
| 姓名 | 吴钰 | | | |
| 班级 | 18级计算机科学与技术创新班 | | | |
| 实验日期： | 2020.6.3 | | | | |
| 实验室： |  | | | | |
|  |  | | | | |
| 实验评价： |  | | | | |
| 实验成绩： | |  | 评价日期： |  |
|  | 指导教师： | |  | | |

# 一、实验目的

掌握页面、页表和段等概念和基本分页和基本分段的地址变换，

模拟基本分页和分段的地址变换机构及通过页表和段表等实现地址

变换，并显示变换过程

# 二、实验要求

给出页面结构、页表（包括具体数据），根据输入的逻辑地址空间中的地址A，通过地址变换机构计算并输出其对应的物理地址。 输入 逻辑地址A 输出 对应物理地址或错误提示

提示

自己设计页面结构和数据、页表结构和数据，对逻辑地址A计算出其页号P和页内地址d，并根据页表计算出其对应的物理地址。 自己设计分段结构和数据、段表结构和数据，根据段表计算出其对应的物理地址。

注意

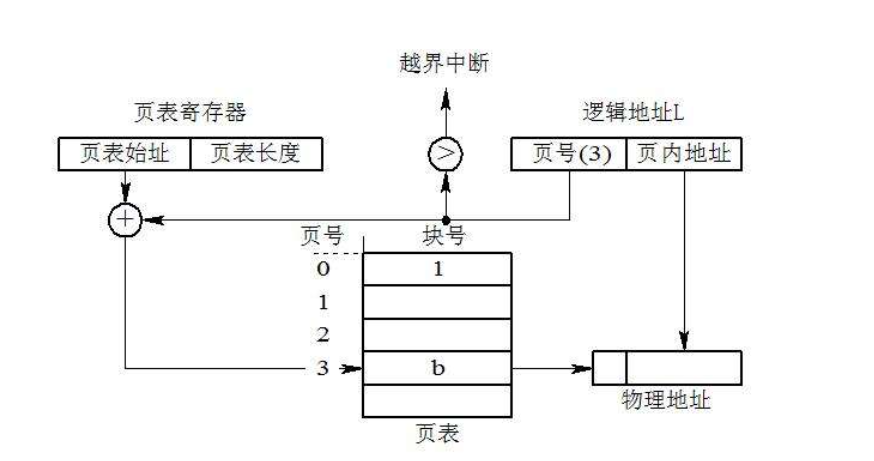
根据逻辑地址结构确定是哪种（分页或分段）地址变换，并判断

逻辑地址是否越界，如果是，输出提示信息，否则输出对应的物理

地址。

# 三、设计

【定义所有抽象数据类型，自定义函数的功能详细描述（伪代码表示），以及主程序的流程图。】



#include<iostream>

#include<stdio.h>

using namespace std;

typedef struct{

int blocknum;

}Page\_table;//页结构

typedef struct{

int length;

int baddress;

}Seg\_table;//段结构

void create\_page(int &n,int &size,Page\_table tab[]){

int i;

cout<<"输入页表长度：";

cin>>n;

cout<<"输入每页大小(KB)：";

cin>>size;

cout<<"设计页表："<<endl;

for(i=0;i<n;i++){

cout<<"页号"<<i<<"对应的块号为：";

cin>>tab[i].blocknum;

}

}

void fpaging(){

cout<<"========================================"<<endl;

cout<<"== 分页存储管理 =="<<endl;

cout<<"========================================"<<endl;

Page\_table tab[100];

int n,size,lg\_address,py\_address,i;

int P,W;

char sign;

create\_page(n,size,tab);

while(1){

cout<<"输入逻辑地址(B)：";

cin>>lg\_address;

P=lg\_address/(size\*1024);

W=lg\_address%(size\*1024);

if(P>=n)

cout<<"越界中断！"<<endl;

else{

cout<<"页号为："<<P<<" 页内位移量为："<<W<<endl;

for(i=0;i<n;i++){

if(i==P){

py\_address=tab[i].blocknum\*(size\*1024)+W;

cout<<"物理地址为(B)："<<py\_address<<" = "<<tab[i].blocknum<<" \* "<<size\*1024<<" + "<<W<<endl;

break;

}

}

}

cout<<"是否继续(继续输入Y,退出输入N)？"<<endl;

getchar();

cin>>sign;

if(sign=='Y')

continue;

else

break;

}

}

void create\_seg(int &n,int &size,Seg\_table tab[]){

int i;

cout<<"请输入段表长度：";

cin>>n;

cout<<"输入最大段长(KB)：";

cin>>size;

cout<<"设计段表："<<endl;

for(i=0;i<n;i++){

cout<<"第"<<i<<"段的段长(B)和基址(B)分别为：";

cin>>tab[i].length>>tab[i].baddress;

}

}

void fsubsection(){

cout<<"========================================"<<endl;

cout<<"== 分段存储管理 =="<<endl;

cout<<"========================================"<<endl;

Seg\_table tab[100];

int n,i,lg\_address,py\_address,size;

int P,W;

char sign;

create\_seg(n,size,tab);

while(1){

cout<<"输入逻辑地址(B)：";

cin>>lg\_address;

P=lg\_address/(size\*1024);

W=lg\_address%(size\*1024);

if(P>=n)

cout<<"越界中断！"<<endl;

else{

cout<<"段号为："<<P<<" 段内地址为："<<W<<endl;

for(i=0;i<n;i++){

if(i==P){

if(W>tab[i].length){

cout<<"越界中断！"<<endl;

break;

}

py\_address=tab[i].baddress+W;

cout<<"物理地址为(B)："<<py\_address<<" = "<<tab[i].baddress<<" + "<<W<<endl;

break;

}

}

}

cout<<"是否继续(继续输入Y,退出输入N)？"<<endl;

getchar();

cin>>sign;

if(sign=='Y')

continue;

else

break;

}

}

#include"define.h"

int main(){

int symbol;

while(1){

cout<<"========================================"<<endl;

cout<<"== 地址变换机构及其变换方法 =="<<endl;

cout<<"========================================"<<endl;

cout<<"== 1)分页存储管理 2)分段存储管理 =="<<endl;

cout<<"== 3)退出 =="<<endl;

cout<<"========================================"<<endl;

cout<<"选择变换方法：";

cin>>symbol;

if(symbol==1){

fpaging();

}

else if(symbol==2){

fsubsection();

}

else if(symbol==3)

break;

}

return 0;

}

# 四、调试分析和思考

【(1) 调试过程中遇到的问题是如何解决的以及对设计与实现中关键点的回顾讨论和分析；(2) 算法的时空分析(包括基本操作和其他算法的时间复杂度和空间复杂度的分析)和改进设想；(3) 经验和体会等。】

地址变换的整个过程可以分为以下六个步骤

1.首先将逻辑地址分为页号和页内地址

2.以页号为索引去检索页表

3.将页号与页表长度进行比较（若页号大于页表长度则表示，所访问的地址已经超越进程的地址空间，出现越界中断）

4.由硬件执行查找操作

5.将页表始址与页号和页表项长度的乘积相加，得到该表项在页表中的位置（可从中得到该页的物理块号）

6.拼接之前的页内地址，形成物理地址

注意以下情况：

1.首先将逻辑地址的段号S与段表长度TL进行比较，若S>TL，表示段号太大，是访问越界，于是产生越界中断信号。

2.未越界则根据段表的始址和该段的段号，计算出该段对应段表项的位置。

# 五、测试数据与结果

【列出你的测试结果，明确输入和输出数据。测试数据应该完整和严格，可以直接贴结果图。】

输入：3c6f#

注意：要以#结束。

截图见下方：

# C:\Users\wuyu\Documents\Tencent Files\1124274751\FileRecv\MobileFile\Image\}$D~O(JUL_9$367138_C(3D.png

