电子科技大学

计算机专业类课程

实验报告

课程名称：操作系统

学　　院：计算机科学与工程

专　　业：计算机科学与技术

学生姓名：韩会彬

学　　号：2013060105004

指导教师：薛瑞尼

日　　期：2016 年 06 月　05 日

电 子 科 技 大 学

实 验 报 告

实验四

一、实验名称：混合索引逻辑地址到物理地址映射

二、实验学时：4

三、实验内容和目的：

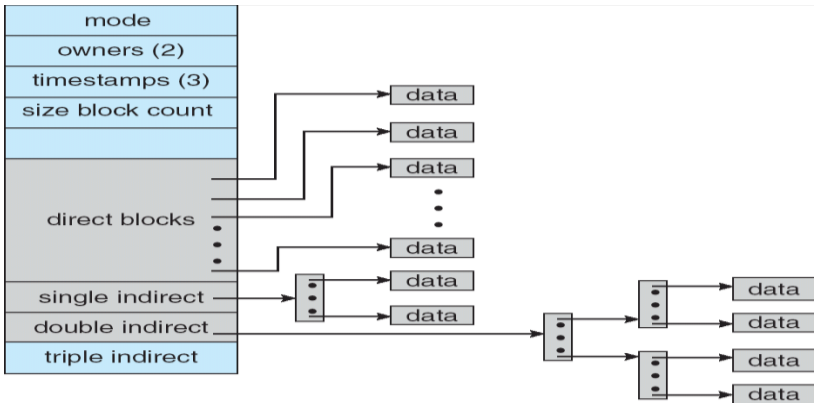
构建 inode 表并编写程序实现逻辑地址到物理地址的转换，实验要求：

- 条件：自定义混合索引 inode 结构
  - 必须包括一次，二次，和三次间接块
  - 逻辑块 n 对应物理块 n
- 输入：文件逻辑地址
- 输出
  i. 输出 inode 详细信息（间接块不展开）
  ii. 物理地址（物理块号，块内偏移）

四、实验原理：

一个数据块容纳不了一个文件的所有分区时，需要若干个索引结点进行存储；

系统既采用了直接地址，又采用了一级索引分配方式，或两级索引分配方式，甚至还采用了三级索引分配方式，将多种索引分配方式相结合而形成的一种分配方式。成为混合索引。



五、实验过程及结果：

实验代码：

```
        #include<stdio.h>
#include<malloc.h>
#include<math.h>

int getBlockNum(const long long int addr);

int main(){

    long long int addr = 0;
    int blocknum = 0;
    int yeneinum = 0;
    printf("输入逻辑地址(十进制)：");
    scanf_s("%d", &addr);

    printf("inode表结构：\n");
    printf("直接索引：\n\t数据地址-->数据\n一级索引：\n\t一级索引地址-->数据地址-->数据\n二级索引：\n\t一级索引地址-->二级索引地址-->数据地址-->数据\n三级索引：\n\t一级索引地址-->二级索引地址-->三级索引地址-->数据地址-->数据\n");

    blocknum = getBlockNum(addr);
    yeneinum = addr % 1024;
    printf("物理块号：%d",blocknum);
    printf("块内偏移：%d",yeneinum);

    printf("over");
    return 0;
}

int getBlockNum(const long long int addr){
    int blocksize = 1024;
    int numofblock = 0;
    /*建表*/
    int Zero_Block[8];   //0-7
    int *Ind_Block;      //8-263一级
    int **Dou_Block;      //264-65799二级
    int ***Tri_Block;     //65800-16843015三级
    int sign1 = 0;        //记录每级索引上限
    int sign2 = 0;
    int sign3 = 0;
    /*初始化*/
    for (int zi = 0; zi < 8; zi++){         //0
        Zero_Block[zi] = numofblock;
        numofblock = numofblock + 1;
```

```
    }
    Ind_Block = (int *)malloc(1024);        //1
    for (int fi = 0; fi < 256; fi++){
        *(Ind_Block + fi) = numofblock;
        numofblock = numofblock + 1;
    }
    sign1 = numofblock;
    Dou_Block = (int **)malloc(1024);       //2
    for (int si = 0; si < 256; si++){
        *(Dou_Block + si) = (int *)malloc(1024);
        for (int sj = 0; sj < 256; sj++){
            *(*(Dou_Block + si) + sj) = numofblock;
            numofblock = numofblock + 1;
        }
    }
    sign2 = numofblock;
    Tri_Block = (int ***)malloc(1024);       //3
    for (int ti = 0; ti < 256; ti++){
        *(Tri_Block + ti) = (int **)malloc(1024);
        for (int tj = 0; tj < 256; tj++){
            *(*(Tri_Block + ti) + tj) = (int *)malloc(1024);
            for (int tk = 0; tk < 256; tk++){
                *(*(*(Tri_Block + ti) + tj) + tk) = numofblock;
                numofblock = numofblock + 1;
            }
        }
    }
    sign3 = numofblock;

    int reblock = 0;
    int blocknum = 0;
    blocknum = addr / 1024;
    /*判断块号*/
    if (blocknum < 8 && blocknum >= 0)
        reblock = Zero_Block[blocknum];
    else if (blocknum >= 8 && blocknum < sign1)
        reblock = *(Ind_Block + blocknum - 8);
    else if (blocknum >= sign1&&blocknum < sign2)
        reblock = *(*(Dou_Block + (blocknum - sign1) / 256) + (blocknum - sign1) %
256);
    else if (blocknum >= sign2&&blocknum < sign3)
        reblock = *(*(*(Tri_Block + (blocknum - sign2) / (256 * 256)) + (blocknum -
sign2) / 256) + (blocknum - sign2) % 256);
    else{
```

```
            printf("地址越界");
            reblock = -1;
        }

        return reblock;
}
```

实验结果：

六、实验结论和心得：

通过本次实验，我更深入的了解了文件存储的机制，更加熟悉了 inode 表的构造机制和工作机制。