# 会议记录

会议时间：2016/4/29

会议主持人：李雨霜

与会人员名单：韩慧敏，张然殊，张慧昕

会议记录：李雨霜

会议主题：解决测试过程中发现的程序bug，并添加提高要求。

自由发言

1. 发言人：李雨霜，议题：多级页表

解决方法：

源码中采用一级页表，直接可查询页表项得到真实地址。修改为二级页表后，前三位表示页目录，中间三位表示二级页表，最后两位表示页内偏移（一页大小4B）。即对于每个进程64\*4B的虚拟空间，有8个页目录，每个页目录下8个页表项，每个页表项对应一个物理地址。在寻址时，由原本的寻址方法变为先得到页目录号，再得到页号，由此得到物理块号，最后由物理块号和页内偏移得到真实的地址，再继续进程操作。

2.发言人：张慧昕，议题：do\_request实现在不同进程

解决方法：

利用命名管道fifo，在新终端调用./request，如无参数则随机生成，否则根据输入参数生成请求，将请求写入fifo，在vmm中创建fifo并以只读打开，每次循环中读取一条请求并加入链表中。

注：关于do\_response()由于组内有不同见解，故目前上交的代码中在每次循环中响应请求。如想只在新终端下执行./response,可对循环中以下代码进行修改，去掉switch中三行注释，并把switch后的do\_response注释掉，则每次循环中根据在新终端的输入进行请求的产生和响应。

3.发言人：韩慧敏，议题：.实现页面老化算法

解决方案：

void do\_PA(Ptr\_PageTableItem ptr\_pageTabIt)

{

unsigned int i,k,s, min, page,index,processNum, c, temp\_count;

printf("没有空闲物理块，开始进行页面老化算法页面替换...\n");

for(s=0,min=0xFFFFFFFF,page=0,index=0,processNum=0;s<Process\_SUM;s++)

for(k=0;k<INDEX\_SUM;k++)

for (i = 0; i < INDEX\_PAGE; i++)

{

for(c = 0, temp\_count = 0;c < 8;c++){

temp\_count = temp\_count \* 10 +

pageIndex[s][k].index[i].count[c] - '0';

}

if (temp\_count < min &&

pageIndex[s][k].index[i].filled==TRUE)

{

min = temp\_count;

processNum=s;

page = i;

index=k;

}

}

printf("选择第%u个目录第%u页进行替换\n",index,page);

if (pageIndex[processNum][index].index[page].edited)

{

/\* 页面内容有修改，需要写回至辅存 \*/

printf("该页内容有修改，写回至辅存\n");

do\_page\_out(&pageIndex[processNum][index].index[page]);

}

pageIndex[processNum][index].index[page].filled = FALSE;

pageIndex[processNum][index].index[page].visited = 0;

for(c = 0;c < 8;c++){

pageIndex[processNum][index].index[page].count[c] = '0';

}

/\* 读辅存内容，写入到实存 \*/

do\_page\_in(ptr\_pageTabIt,pageIndex[processNum][index].index[page].blockNum);

/\* 更新页表内容 \*/

ptr\_pageTabIt->blockNum =pageIndex[processNum][index].index[page].blockNum;

ptr\_pageTabIt->filled = TRUE;

ptr\_pageTabIt->edited = FALSE;

ptr\_pageTabIt->visited = 0;

for(c = 0;c < 8;c++){

ptr\_pageTabIt->count[c] = '0';

}

printf("页面替换成功\n");

}

4.发言人：张然殊，议题：实现多道程序的存储控制

解决方法：本实验中初始将进程数设置为4（在vmm.h中可调Process\_SUM）。将虚拟空间扩大到64\*4\*4，即每个进程对应赋存axuMem中64\*4的空间。将PageIndex数组扩充成二维数组，其中第一维度用来标明页表属于哪一进程。

PageIndexItem pageIndex[Process\_SUM][INDEX\_SUM];

在生成随机命令时，将随机生成进程号，标明该命令属于哪一进程。在手动输入命令时，也需要输入进程号。

如果进程号与该进程对应的赋存地址不匹配，将返回错误。

在寻址时，加入进程号，对应进程号进行下一步寻址，得到目录号、页号、物理块号、页内偏移，最终计算出实际地址。

会议结束

会议记录提交人：李雨霜

会议记录审阅人：韩慧敏