“操作系统原理与实践”实验报告

proc文件系统的实现

这是8个实验中，最后的一个实验，以下是本人实验中 proc.c 的源码

```
#include <linux/kernel.h>
#include <sys/stat.h>
#include <errno.h>
#include <linux/fs.h>
#include <asm/segment.h>
#include <string.h>
#include <a.out.h>
#include <linux/sched.h>

static char *ps_content = NULL;
static char *hd_content = NULL;

int proc_read struct char int off_t
{
        int i, j, k, l;
        char *tmp = buf;
        char container[ 256 ];

        struct task_struct **p;
        struct super_block *sb;
        struct buffer_head *bh;
        char *bit_map;
        int free_block = 0;
        int used_block = 0;

        if ((inode - >i_zone[ 0] != FILE_PSINFO) && (inode - >i_zone[ 0] != FILE_HDINFO)){
                printk( "This proc file isn't supported yet.\n\r" );
                return - EINVAL;
        }

        if (inode - >i_zone[ 0] == FILE_PSINFO){
                //printk("f_pos=%d\n", *f_pos);
                if ( ! ( *f _pos ) ) {
输入代码           ps _cont ent = ( char *) malloc (PAGE_SIZE);
                        memset (ps_content, 0, sizeof (ps_content));
                        sprintf (ps_content,
"pid\tstate\tfather\tcounter\tstart_time\n" );
                        for (p = &FIRST_TASK; p <= &LAST_TASK; p++){
                                if (*p){
                                        memset (container, 0, sizeof (container));
                                        sprintf (container, "%ld\t%ld\t%ld\t%ld\t%ld\n" ,
                                                (*p) - >pid, (*p) - >state, (*p) - >father,
(*p)->counter, (*p) - >start_time);
                                        strncat (ps_content, container, strlen
(container));
                                }
                        }
                }
                for (i= 0; i<count; i++){
                        if (ps_content[*f_pos+i] == 0){
                                break ;
                        }
                        put_fs_byte(ps_content[i], tmp);
                        tmp++;
                }
                if (!i){
                        //printk("free ps_content\n");
                        free (ps_content);
                        *f_pos = 0;
                }
                else
                        (*f_pos) += i;
        }

        else if (inode - >i_zone[ 0] == FILE_HDINFO){
                if (!(*f_pos)){
                        sb = get_super(inode - >i_dev);
                        //printk("s_zmap_blcoks %d\n", sb - >s_zmap_blocks);
                        l= 0;
                        for (i= 0; i<sb - >s_zmap_blocks; i++){
                                bh = sb - >s_zmap[i];
                                bit_map = ( char *)bh - >b_data;
                                for (j= 0; j<BLOCK_SIZE; j++){
                                        for (k= 0; k<8; k++){
                                                l++;
                                                if (l>sb - >s_nzones)
                                                        goto CALCULATE_OVER;
                                                if (bit_map[j] & ( 1<<k))
                                                        used_block++;
                                                else
                                                        free_block++;
                                        }
                                }
                        }
CALCULATE_OVER:
                        hd_content = ( char *) malloc (PAGE_SIZE);
                        memset (hd_content, 0, sizeof (hd_content));
                        sprintf (hd_content, "total inode:\t%d\ntotal blocks:\t%d\nfree
blocks:\t%d\nused blocks:\t%d\n" ,
                                sb - >s_ninodes, sb - >s_nzones, free_block, used_block);
                }
                for (i= 0; i<count; i++){
                        if (hd_content[*f_pos+i] == 0){
                                break ;
                        }
                        put_fs_byte(hd_content[i], tmp);
                        tmp++;
                }
                if (!i){
                        //printk("free ps_content\n");
                        free (hd_content);
                        *f_pos = 0;
                }
                else
                        (*f_pos) += i;
        }

        return i;
}
```

如果大家对 内核级线程实现的实验有兴趣 可以加Q群 258178702 一起探讨实验或者其他操作系统实现问题 也附有本人实验1~8的源码