**OS实验三说明文档**

**161250098 彭俊杰**

该说明文档写于实验代码完成后（在实验进行过程中忘记进行详细记录了……），若有些许遗漏，望见谅！

首先，本次作业是在《Orange’s》第七章源码m的基础上完成的。接下来不妨依照实验要求回想修改与添加的地方吧……

1. MakeFile中修改内容：

（1）CFLAGS = -I include/ -c -fno-builtin -fno-stack-protector（禁用堆栈保护。）

（2）添加execute及run命令

.PHONY : everything final image clean realclean disasm all buildimg run execute

execute:

bochs -f bochsrc

run : image execute

1. 删除Test A, Test B, Test C三个进程，仅留下task\_tty 作为唯一进程

删除kernel/main.c中对应的三个进程体及优先级，在kernel/global.c中删除task\_table中的对应进程，在include/proc.h进行适当修改：

/\* Number of tasks \*/#define NR\_TASKS 1

/\* stacks of tasks \*/

#define STACK\_SIZE\_TTY 0x8000

#define STACK\_SIZE\_TOTAL (STACK\_SIZE\_TTY)

删除include/proto.h中对应的函数声明，同时关闭kernel/clock.c中的进程调度，将其注释掉//schedule();

1. 将控制台个数设置为一，去除多余控制台

在include/const.h中修改宏

/\* TTY \*/ #define NR\_CONSOLES 1 /\* consoles \*/

1. 增加console属性并修改控制台初始化过程

在include/console.h中往struct console添加属性：

查找模式对应的状态flag及关键字key相关：

unsigned int search\_flag;

unsigned int key\_len;

char key[20];

初始时间戳以及清屏flag：

unsigned int start\_tick;

unsigned int clear\_flag;

在kernel/console.c中修改init\_screen（属性初始化及清屏）：

p\_tty->p\_console->search\_flag = 0;

p\_tty->p\_console->key\_len = 0;

p\_tty->p\_console->clear\_flag = 0;

for(int i=0;i<20;i++)

p\_tty->p\_console->key[i]='\0';

if (nr\_tty == 0) {

disp\_pos=0;

for(int i=0;i<80\*25;i++)

disp\_str(" ");

disp\_pos = 0;

p\_tty->p\_console->cursor = p\_tty->p\_console->original\_addr;

}

1. 修改回车键换行、退格键删除，添加Tab、Esc键处理，补充查找功能

kernel/tty.c中添加:

case TAB:

put\_key(p\_tty, '\t');

break;

case ESC:

put\_key(p\_tty, '\e');

break;

对kernel/console.c中进行大量修改，代码就不贴出来了。。。

修改out\_char中的所有处理，添加

PUBLIC int checkLine(CONSOLE\* p\_con,int line) //检测’\0’

PUBLIC void display\_key(CONSOLE\* p\_con,int status) //将key word改变颜色

PUBLIC void clear\_screen(CONSOLE\* p\_con) //清屏

PS:将’\0’作为’\n’指示性字符，对于Tab键实现纵向对齐，填充黑色空格，清屏是会将clear\_flag改为0不过只会在search\_flag为0时执行清屏操作，search\_flag三种状态：0普通文本编辑，1关键字录入，2查找锁定

1. 添加自动清屏功能：

在kernel/tty.c的task\_tty进程中添加clear\_flag的更改（利用的是时间中断产生的时间戳）

while (1) {

for (p\_tty=TTY\_FIRST;p\_tty<TTY\_END;p\_tty++) {

if(((get\_ticks() - p\_tty->p\_console->start\_tick) \* 1000 / HZ) % 100000 == 0){

p\_tty->p\_console->clear\_flag=1;

clear\_screen(p\_tty->p\_console);

}

tty\_do\_read(p\_tty);

tty\_do\_write(p\_tty);

}

}