% 实验代码

switch\_to:

pushl %ebp

中。本次实验参考了

## "操作系统原理与实践"实验报告

## 基于内核栈切换的进程切换

本实验完整代码在 http://git.shiyanlou.com/gaoyuanhezhihao/shiyanlou\_cs115/src/master/Lab5

qiuy\_mac同学 (https://www.shiyanlou.com/user/34029 )的实验报告和代码,在此表示衷心的感谢。 1.首先改写 switch\_to函数,因为linux-0.11使用的是一个宏,所以我们这里需要用汇编编写一个switch\_to函数。之前我把 switch\_to 以及first\_retum\_from\_kemel写在一个新建的文件中,结果有问题,在参考别的同学的实验报告后,把这两 个函数写到了kemel/system\_call.s中,同时需要注意的是,为了能够在schedule函数,copy\_process函数中"看到"这 两个函数,我们需要用.global声明这两个函数符号,同时在sched.c,fork.c中声明这两个函数。具体可以参考源代码 .globl switch\_to,first\_return\_from\_kernel .align 2

```
movl %esp, %ebp
pushl %ecx
pushl %ebx
pushl %eax
mov1 8(%ebp),%ebx
cmpl %ebx, current
je 1f
movl %ebx, %eax
xchgl %eax, current
movl tss, %ecx
addl $4096, %ebx
movl %ebx, ESPO (%ecx)
movl %esp, KERNEL_STACK (%eax)
movl 8(%ebp),%ebx
mov1 KERNEL_STACK(%ebx), %esp
movl 12(%ebp), %ecx
11dt %cx
movl $0x17,%ecx
mov %cx, %fs
```

struct task\_struct { /\* these are hardcoded - don't touch \*/

2.修改任务数据结构: task\_struct, 在其中加入内核栈指针成员。

```
long state; /* -1 unrunnable, 0 runnable, >0 stopped */
                                                                同时需要修改的是0号进程的初始化
        long counter;
        long priority;
        long kernelstack;
数据:
```

```
/* state etc */ { 0, 15, 15, PAGE_SIZE+( long )&init_task,\
 /* signals */ 0,{{},}, 0, \
 /* ec,brk... */ 0,0,0,0,0,\
3.修改copy_process函数。首先申请一页内存:
```

p = (struct task\_struct \*) get\_free\_page();

p = (struct task\_struct \*) get\_free\_page();

进程的内核栈。所以我们把这页内存最后的地址作为栈顶:

= cs & 0xffff ;

= ds & 0xffff

= eip;

#define INIT TASK \

与父进程不同的用户代码走向。

后返回到用户程序中。

\*( -- krnstack)

\*( -- krnstack)

\*( -- krnstack)

```
这段代码就是新建立的进程保存信息的地方,原本的copy_process中从父进程复制了几乎所有的寄存器保存在这里,
其中有一个很大的不同就是eax被写为了0,起始这就是fork()函数的返回值,这样在最后fork()返回的时候就可以实现
```

p->tss.eax = 0;

```
现在我们不在使用tss进程切换方式,所以这里保存tss内容的代码可以不要。重要的是在我们的栈中手动压入一些信
息。这一点在实验提示中已经有了详细的说明。需要注意的是,其实我们的这个栈就在这个新建的p指向的页中,前面
```

获得了一页内存,大小为4096字节。前面保存任务数据结构只用了很少的一部分,剩下的内容完全可以用来作为新建

```
long *krnstack;
krnstack = (long)(PAGE_SIZE+(long)p);
```

然后再这个内核栈中手动"压入"一些内容,使得它在后面切换的时候能够顺利的进入first\_retum\_from\_kemel函数,最

\*( -- krnstack) = ss & 0xffff ; \*( -- krnstack) = esp; \*( -- krnstack) = eflags;

```
*( -- krnstack)
                       = es & 0xffff
       *( -- krnstack)
                       = fs & 0xffff
                       = gs & 0xffff
       *( -- krnstack)
       *( -- krnstack)
                       = esi;
       *( -- krnstack)
                       = edi;
       *( -- krnstack)
                       = edx;
       *( -- krnstack)
                       = (long )first_return_from_kernel;
       *( -- krnstack)
                       = ebp;
       *( -- krnstack)
                       = ecx;
       *( -- krnstack)

    ebx;

       *( -- krnstack)
                       = 0:
这个时候,栈顶的地址就可以保存到任务数据结构中了
       p->kernelstack
                        krnstack;
                  Bochs x86 emulator, http://bochs.sourceforge.net/
```

linux-0.11\$ cd .

cd Reperence/

ls

rungdb

linux-0.11\$ cd ..

http://www.nongnu.org/vgabios Bochs VBE Display Adapter enabled Bochs BIOS - build: 02/13/08

ata0 master: Generic 1234 ATA-6 Hard-Disk ( 60 MBytes)

\$Revision: 1.194 \$ \$Date: 2007/12/23 19:46:27 \$

Options: apmbios pcibios eltorito rombios32

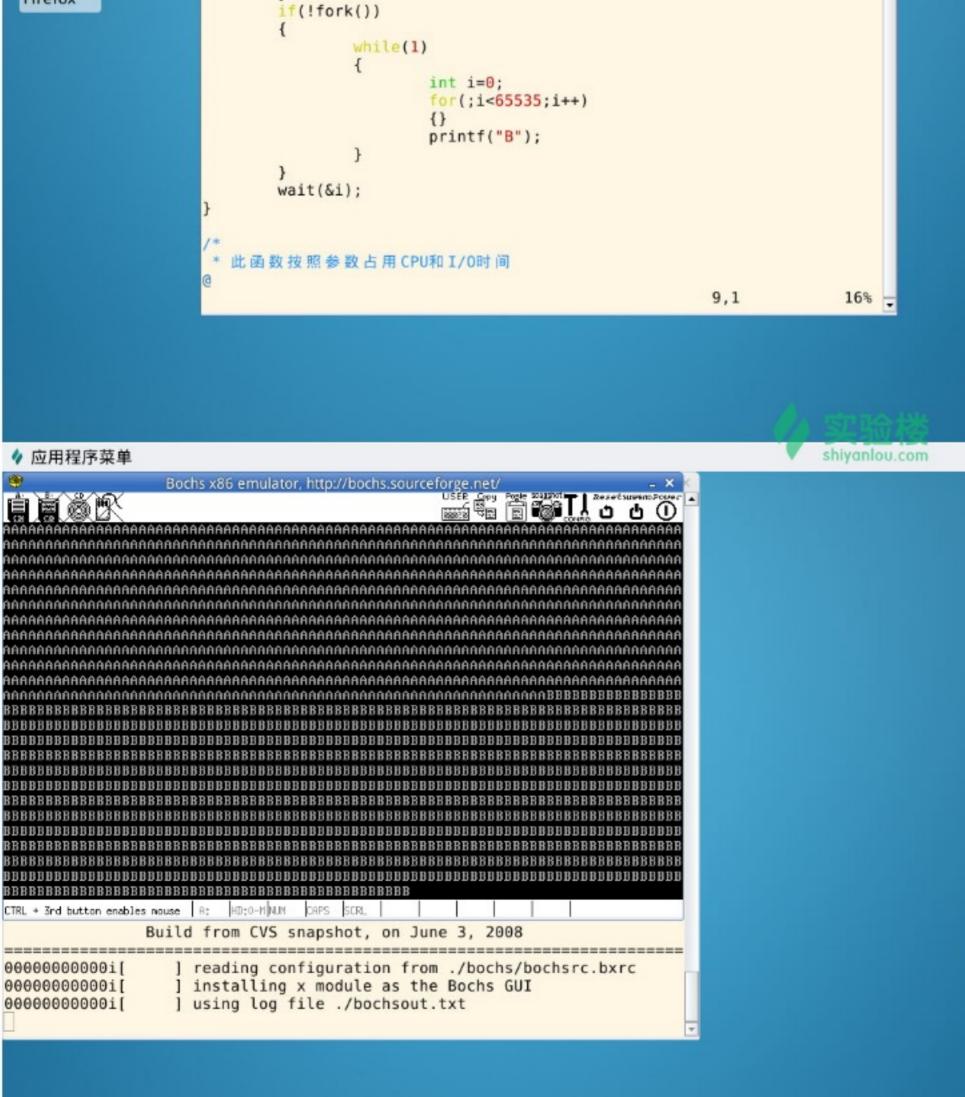
http://bochs.sourceforge.net

か 应用程序菜单

```
Booting from Floppy...
                                                                        Reperence$ ls
Loading system ...
                                                                        Reperence$ cd shiyanl
Partition table ok.
39065/62000 free blocks
                                                                        Reperence/shiyanlou_c
19520/20666 free inodes
3454 buffers = 3536896 bytes buffer space
ree mem: 12582912 bytes
                                                                        Reperence/shiyanlou c
Ok.
[/usr/root]# ls
              hello
                                                                         Reperence$ cd ..
README
                            hello.o
                                           linux0.tgz
                                                          shoe
                             linux-0.00
              hello.c
gcclib140
                                           mtools.howto
                                                          shoelace.tar.Z
                                                                         ./run
[/usr/root]#
CTRL + 3rd button enables nouse A: HD:0-MINUM CAPS SCRL
                                    Build from CVS snapshot, on June 3, 2008
                     000000000000i[
                                       ] reading configuration from ./bochs/bochsrc.bxrc
                     0000000000001[
                                       ] installing x module as the Bochs GUI
                     0000000000001
                                       ] using log file ./bochsout.txt
                                                                                       shiyanlou.com

か 应用程序菜单

                                   shiyanlou@591994b238bb: ~/Code/shiyanlou_cs115/Lab5
                     int main(int argc, char * argv[])
                             int i;
   Home
                             if(!fork())
                                     while(1)
                                             int i=0;
  Terminat
                                             for(i=0;i<65535;i++)
  or
                                             printf("A");
  Firefox ···
                              f(!fork())
                                     while(1)
                                             int i=0;
                                              for(;i<65535;i++)
                                             printf("B");
                             wait(&i);
                        此函数按照参数占用CPU和I/0时间
                                                                           9,1
                                                                                          16%
```



0

shiyanlou.com