深圳大学
Shenzhen University

操作系统

操作系统大作业：part I-1

谭舜泉
计算机与软件学院

大作业Part I

- 大作业Part I 由六部分组成。六部分均为阅读xv6中文文档，理解对应的xv6源代码，并回答大作业中提出的代码理解问题。
  1. （第二周）第0章：操作系统接口；
  2. （第四周）第1章：第一个进程；
  3. （第六周）附录A/B：PC硬件及引导加载器；
  4. （第八周）第2章：页表；
  5. （第十周）第4章：锁；
  6. （第十二周）第3章：陷入、中断和驱动程序。

大作业Part I

- 注意上半学期的大作业分为两部分汇总提交。
- 其中1-3提交一份实验报告，而4-6提交另一份实验报告。

实验报告

- 实验报告要求使用深大实验报告模板填写。
- 允许两至三人组成一个小组，撰写实验报告。
- 请按照实验要求，完成实验并撰写实验报告。把实验报告电子版上传到BLACKBOARD。 本实验分组完成。每个实验小组完成一份实验报告即可。但要求小组中每个成员都把实验报告的副本上传到BLCKBOARD，并注明同组成员的名字和学号。

深圳大学
Shenzhen University

第0章操作系统接口

- Unix 里机制结合良好的窄接口提供了令人吃惊的通用性。
- 进程通过系统调用使用内核服务。系统调用会进入内核，让内核执行服务然后返回。所以进程总是在用户空间和内核空间之间交替运行。
- 内核提供的一系列系统调用（见系统调用列表）

第0章进程和内存

- 一个 xv6 进程由两部分组成：
  - 一部分是用户内存空间（指令，数据，栈），
  - 另一部分是仅对内核可见的进程状态。
- 一个进程可以通过系统调用 fork 来创建一个新的进程。
- 系统调用 exec 将从某个文件（通常是可执行文件）里读取内存镜像，并将其替换到调用它的进程的内存空间。

第0章I/O 和文件描述符

- **文件描述符**是一个整数，它代表了一个进程可以读写的被内核管理的对象。
- 进程从文件描述符0读入（标准输入），从文件描述符1输出（标准输出），从文件描述符2输出错误（标准错误输出）。

第0章管道

- 管道是一个小的内核缓冲区。
- 它以文件描述符对的形式提供给进程，一个用于写操作，一个用于读操作。
- 从管道的一端写的数据可以从管道的另一端读取。
- 管道提供了一种进程间交互的方式。

代码理解问题

- :8024-:8026.
  1. if(fork1() == 0)

     为什么fork1()返回值为0时才进入if语句内部？
  2. runcmd(parsecmd(buf));
     - 阅读runcmd的代码，其中：
     - $echo README对应的cmd->type是哪个？
     - 相应的，$ls; echo “hello world“ 对应的cmd->type是哪个？
     - 而ls | wc 对应的cmd->type是哪个？给出你的答案，并从代码中给出解释。

代码理解问题

```
char *argv[2];
argv[0] = "cat";
### argv[1] = 0;
if(fork() == 0) {
    close(0);
    open("input.txt", O_RDONLY);
    exec("cat", argv);
}
```

- 子进程关闭文件描述符0后， 我们可以保证 open 会使用0作为新打开的文件 input.txt 的文件描述符（因为0是 open 执行时的最小可用文件描述符） 。 之后 cat 就会在标准输入指向 input.txt 的情况下运行。
- 问题：阅读:7930对应的switch分支及相关代码，请说明如何确保rcmd->fd为标准输入的呢？

代码理解问题（sh.c :7950-:7972）

- 管道的例子：$ls –l | wc –l (不是xv6中的shell语句)
- 第二、三个if语句中，管道的读端口和写端口都通过close语句关闭了，请问还怎么保证pcmd->left的输出进入管道的写端口，而pcmd->right的输入进入管道的读端口？
- 为什么在父进程这里，还需要有两个close语句？以及两个wait语句？

```
case PIPE:
  pcmd = (struct pipecmd*)cmd;
  if(pipe(p) < 0)
    panic("pipe");
  if(fork1() == 0){
    close(1);
    dup(p[1]);
    close(p[0]);
    close(p[1]);
    runcmd(pcmd->left);
  }
  if(fork1() == 0){
    close(0);
    dup(p[0]);
    close(p[0]);
    close(p[1]);
    runcmd(pcmd->right);
  }
  close(p[0]);
  close(p[1]);
  wait();
  wait();
  break;
```

代码理解问题

- 这些问题，如果不能理解的话也不用担心。我将在两周后的课堂上（在实验报告提交前）给予解答。