作业 7 异常与外设中断 参考答案

同济大学计算机系操作系统　国豪 2023　拔尖班 2023
姓名　学号

Part 1、异常，备考期末
1、（Ubuntu 系统）输入除数 0，描述异常响应过程

```c
#include      <unistd.h>
#include      <stdio.h>
int
main(void)
{
    int a,b,c;


    for ( ; ; )
    {
        printf("输入被除数\n");
        scanf("%d", &a);

        printf("输入除数\n");
        scanf("%d", &b);

        c=a/b;

        printf("%d / %d = %d\n",a,b,c);
    }
}
```

进程的运行环境：
应用程序没有定义 8#异常处理程序　∴u_signal[8]==0

除 0 异常响应过程：
（1）　b==0，执行语句 c=a/b 时，CPU 硬件抛出除 0 异常，压栈保护除 0 指令地址。
（2）　响应异常：内核（核心态运行的现运行进程）执行 0#异常处理程序，
- 向自己发 8#信号 SIGFPE
- 同步处理该信号：u_signal[8]是 0，执行 Exit 函数终止自己。

评论：
（1）　之后，没有以后了~~~　出错的应用程序在执行故障操作之前被内核杀死，系统有效阻止了错误蔓延，Unix 好漂亮的设计！
（2）　内核是谁？陷入核心态运行的现运行进程。它是联合国轮值主席。
需要访问系统资源时，自服务、执行系统调用处理程序。
中断发生时，代理内核执行中断处理程序，为系统和其它进程服务。

2、（Ubuntu 系统）输入除数 0，描述异常响应过程（突出用户栈、核心栈的变化）

进程的运行环境：

应用程序有定义 8#异常处理程序，u_signal[8]==信号处理程序 sig_dyzero( )的入口地址

除 0 异常响应过程：

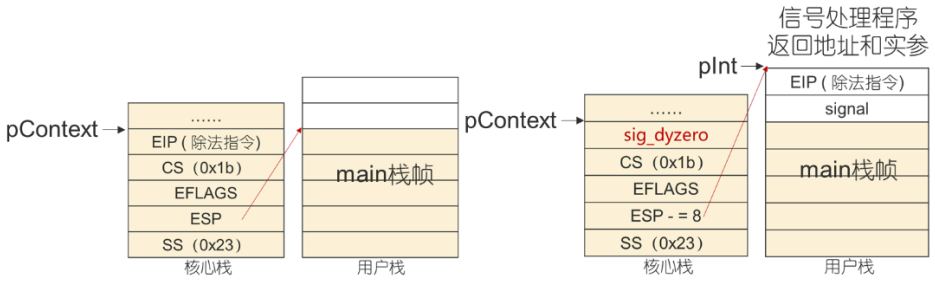
（1）b==0，执行语句 c=a/b 时，CPU 硬件抛出除 0 异常，压栈保护除 0 指令地址。

（2）响应异常：内核（核心态运行的现运行进程）执行 0#异常处理程序，

- 向自己发 8#信号 SIGFPE
- 同步处理该信号：u_signal[8]不是 0，篡改核心栈和用户栈，为执行信号处理程序做准备，具体操作：

  （核心栈底，将第一现场保护区 EIP 改为信号处理程序 sig_dyzero( )首地址）

  （用户栈顶，为信号处理程序 sig_dyzero( )布置栈帧：实参区放信号数值和返回地址是除法指令）



（3）异常处理程序返回，异常入口程序返回，现运行进程回用户态，执行信号处理程序 sig_dyzero( )，返回后重新执行除法指令，触发新的除 0 异常。步骤 1~3，周而复始，无法停止。

3、（Ubuntu 系统）debug 第二小问展示的这个程序

略。看 PPT

Part 2、外设中断

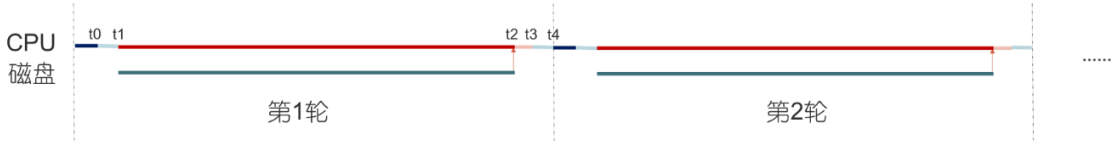
1、 情景分析，系统并发 copy 进程 和 matrix 进程（4096*4096 矩阵乘法）。

（1） 请画甘特图、描述这两个进程轮流使用 CPU 的情景（不考虑 write 系统调用）。

（2） 归纳总结：计算机系统怎样执行 IO 操作。

参考答案：

（1）



蓝色系，copy 进程。

- 深蓝，执行应用程序；浅蓝，执行 read 系统调用；青色，执行 IO 操作。

红色系，matrix 进程。

- 深红，执行应用程序（矩阵运算），浅红，执行磁盘中断处理程序，唤醒 copy 进程。

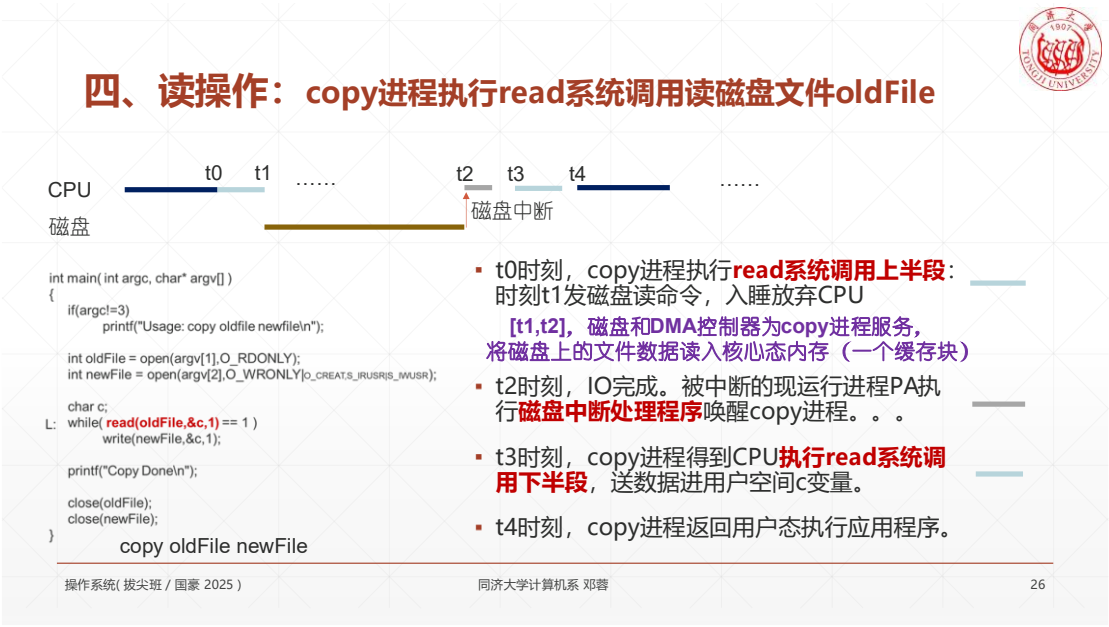
系统运行很规整，分若干轮[注 1]。每轮，copy 进程和 matrix 进程物理并行，前者使用磁盘 IO，后者使用 CPU 运算。IO 操作会耗时很久很久，所以，运行期间，matrix 进程几乎独占 CPU，只在 IO 完成时刻，腾出 CPU 执行与 copy 进程 IO 操作相关的代码。计算、IO 并行，系统性能很好。

[注 1] matrix 进程执行期间，每个时钟滴答执行时钟中断处理程序，维护系统时钟。整数秒会有例行调度，因为只有一个 SRUN 进程，所以下一秒还是 matrix 进程运行。。。这个是插曲，不影响计算、IO 并行大局。

[此外]，每轮有 2 个调度点。看第一轮，t1，copy 入睡主动放弃 CPU（无条件执行 Swtch，不和别人比谁优先级高）；t2，matrix 被剥夺，把 CPU 让给优先级更高的 copy 进程。。。

答题，不需要这两个细节。我们加深对系统的理解。

（2）略，自己归纳



延申：scanf 读键盘，进程执行的也是 read 系统调用：

read(0,***,***)读标准输入文件

用户敲键盘输入字符。所以，读键盘，CPU 是不需要向外设发命令的，read 系统调用上半段，进程直接 Sleep 入睡。

Part 3、（思考题，总评分加 5）Unix V6++系统，复现上述程序。

略