

# lab3实验报告

陈德丹 221220159 邮箱: 221220159@smail.nju.edu.cn

## 实验进度

完成所有实验内容

## 实验过程

**完善lab3/lib/syscall.c 中的三个库函数**，分别实现创建子进程、休眠指定时间、结束进程的运行

```
pid_t fork() {return syscall(SYS_FORK, 0, 0, 0, 0, 0);}
int sleep(uint32_t time) {return syscall(SYS_SLEEP, time, 0, 0, 0, 0);}
int exit() {return syscall(SYS_EXIT, 0, 0, 0, 0, 0);}
```

### 时钟中断处理

1. 遍历 pcb，将状态为 STATE\_BLOCKED 的进程的 sleepTime 减一，如果进程的 sleepTime 变为0，重新设为 STATE\_RUNNABLE
2. 将当前进程的 timeCount 加一，如果时间片用完（timeCount==MAX\_TIME\_COUNT）且有其它状态为 STATE\_RUNNABLE 的进程，切换，否则继续执行当前进程  
\*若没有可执行的进程，则切换 pid 为0的idle进程

```

void timerHandle(struct StackFrame *sf)
{
    // TODO
    int i;
    for (i = 0; i < MAX_PCB_NUM; i++) {
        if (pcb[i].state == STATE_BLOCKED) {
            pcb[i].sleepTime--;
            if(pcb[i].sleepTime == 0)
                pcb[i].state = STATE_RUNNABLE;
        }
    }
    if (pcb[current].state == STATE_RUNNING) {
        if (pcb[current].timeCount < MAX_TIME_COUNT)
            pcb[current].timeCount++;
        else {
            pcb[current].timeCount = 0;
            pcb[current].state = STATE_RUNNABLE;
        }
    }
    if (pcb[current].state != STATE_RUNNING) {
        for (i = 0; i < MAX_PCB_NUM; i++){
            if (i != current && pcb[i].state == STATE_RUNNABLE) break;
        }
        if (i == MAX_PCB_NUM) i = 0;

        current = i;
        pcb[current].state = STATE_RUNNING;

        uint32_t tmpStackTop = pcb[current].stackTop;
        tss.esp0 = pcb[current].prevStackTop;
        pcb[current].stackTop = pcb[current].prevStackTop;
        asm volatile("movl %0, %%esp"::"m"(tmpStackTop));
        asm volatile("popl %gs");
        asm volatile("popl %fs");
        asm volatile("popl %es");
        asm volatile("popl %ds");
        asm volatile("popal");
        asm volatile("addl $8, %esp");
        asm volatile("iret");
    }
    return;
}

```

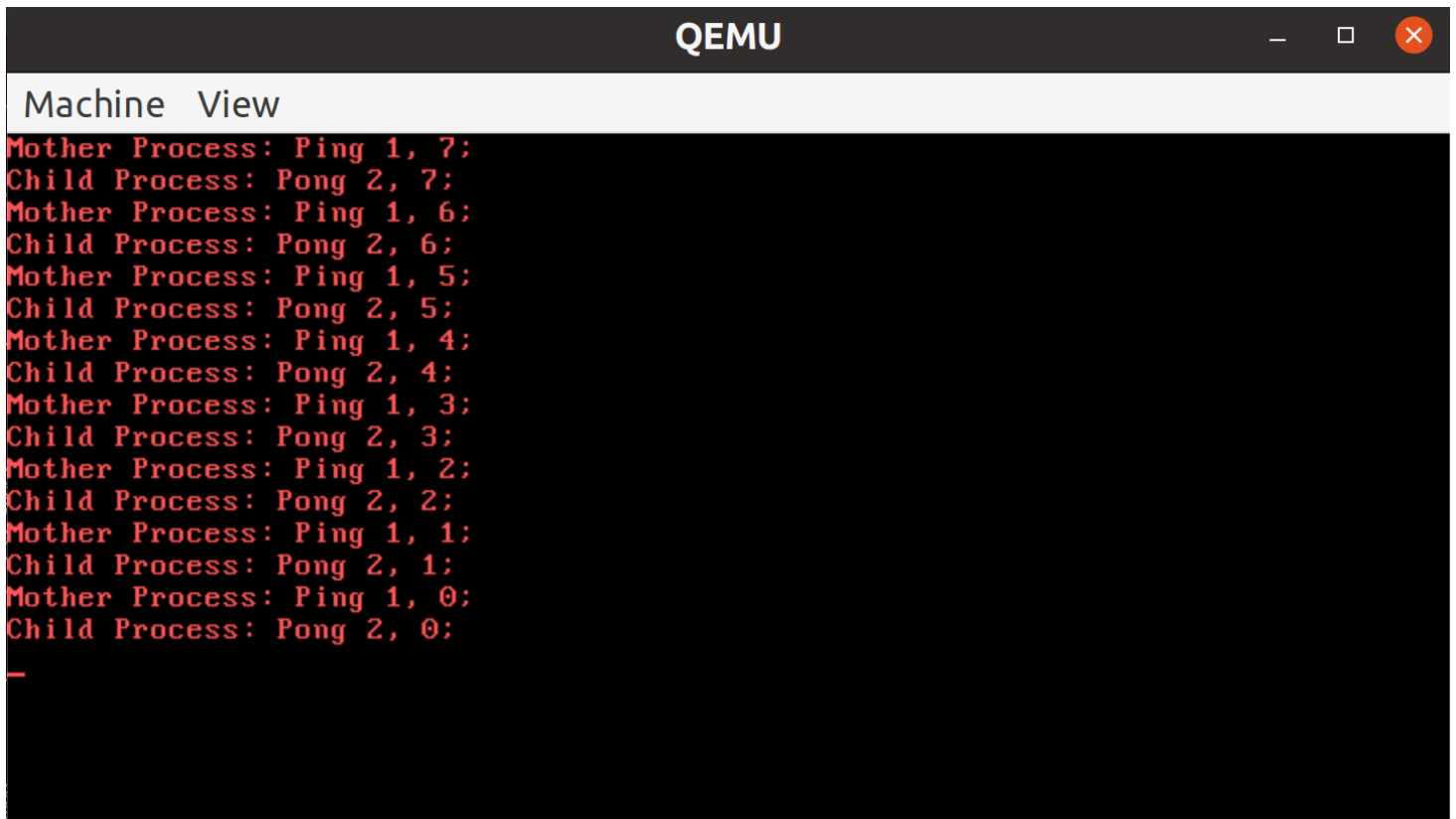
## 实现syscallFork、syscallSleep、syscallExit

syscallFork 先找一个空闲的 pcb(STATE\_DEAD) 作为子进程的进程控制块，拷贝相应空间资源，在进行 pcb 的复制时，要注意有些可以直接复制，有些则需要经过计算，比如 stackTop 就是根据两进程的 stackTop 地址和实际值之差相等得到

syscallSleep 要注意的是判断传入参数的合法性(>0)，将当前进程的状态设置为 STATE\_BLOCKED，利用 timerHandle 进行进程切换

syscallExit 将当前进程的状态设置为STATE\_DEAD，然后模拟时钟中断进行进程切换

## 实验结果



```
Machine View
Mother Process: Ping 1, 7;
Child Process: Pong 2, 7;
Mother Process: Ping 1, 6;
Child Process: Pong 2, 6;
Mother Process: Ping 1, 5;
Child Process: Pong 2, 5;
Mother Process: Ping 1, 4;
Child Process: Pong 2, 4;
Mother Process: Ping 1, 3;
Child Process: Pong 2, 3;
Mother Process: Ping 1, 2;
Child Process: Pong 2, 2;
Mother Process: Ping 1, 1;
Child Process: Pong 2, 1;
Mother Process: Ping 1, 0;
Child Process: Pong 2, 0;
```

鉴于今天是母亲节，将Father改成Mother，希望以后parent process的默认翻译不是父进程，更不希望在框架代码中把英语原版也改成了father process，望修改，谢谢！