

# 实验六报告讲义

陈晨曦

2023 年 6 月 16 日

## 1 实验内容

1. 本实验在自己所完成的实验5的基础上进行，不提供框架
2. 调度算法，至少2种（不含FCFS，必须含有抢占式调度算法）
3. hook机制的实现

```
typedef struct scheduler {
    unsigned long type; // the type of the scheduler
    int preemptive_or_not; //if True, the scheduler is preemptive
    myTCB* (*nextTsk_func)(void);
    void (*enqueueTsk_func)(myTCB *tsk);
    void (*dequeueTsk_func)(myTCB *tsk);
    void (*schedulerInit_func)(myTCB* idleTsk);
    int (*createTsk_hook)(void (*tskBody)(void),tskPara para);
    void (*tick_hook)(void); //if set, tick_hook will be called every tick
}scheduler;
```

4. 自编测试用例

## 2 实验结果展示

助教自己的实验封装了 testPRIORITY 作为命令，（自己决定测试方法，不需要和助教的一致）。

```
addNewCmd("testPRIORITY\0", testPRIORITY, NULL, "Test_PRIORITY.\0");

int testPRIORITY(int argc, unsigned char **argv){
    SchedulerPolicySet(SCHEDULER_PRIORITY);
    tskPara myTsk0_para={2,2,2}; // Priority exeTime arrTime
    createTsk(myTsk0,myTsk0_para); //myTsk0 will create myTsk3 which is {1,1,1}
    tskPara myTsk2_para={3,3,3};
    createTsk(myTsk2,myTsk2_para);
    tskPara myTsk1_para={3,3,3};
    createTsk(myTsk1,myTsk1_para);
    tskPara myTsk4_para={3,3,3};
    createTsk(myTsk4,myTsk4_para);
    return 0;
}
```

测试抢占式的 PRIORITY 调度算法，在 tsk0 中创建的 tsk3 的 Priority 大于 tsk0，所以先输出 tsk3 再输出 tsk0，运行结果合理。

```
xlanchen >:testPRIORITY
testPRIORITY
xlanchen >:
*****
*      Tsk3: HELLO WORLD!      *
*****

*****
*      Tsk0: HELLO WORLD!      *
*****

*****
*      Tsk2: HELLO WORLD!      *
*****

*****
*      Tsk1: HELLO WORLD!      *
*****

*****
*      Tsk4: HELLO WORLD!      *
*****

xlanchen >:|
```

图 1: testPRIORITY

### 3 实验报告要求

1. 自编测试用例，详细说明测试用例的设置依据。
2. 实验结果截图，是否和预期一致。