## 《操作系统》实验报告

学号： 姓名： 上课时间：周三上午

|  |  |
| --- | --- |
| **实验名称：** | 处理器调度-实时调度算法EDF和RMS |
| **实验目的：** | 1）深入理解处理器调度算法，了解硬实时概念，掌握周期性实时调度算法EDF和RMS的可调度条件。  2）将样例代码中的RMS改为抢占式条件。 |
| 实验内容： | 1）在Linux环境中采用用户及线程模拟实现EDF和RMS两种实时调度算法，给定一组实时任务，按照EDF算法和RMS算法分别判断是否可调度。   1. 在可调度情况下，创建一组用户及线程，分别代表各个实时任务。在终端上画出字符Gantt图。 |
| **程序框图：**   |  |  | | --- | --- | | 主程序 | Proc()函数 | |  |  | | |
| **程序关键代码和解读：**   |  |  | | --- | --- | |  | 初始化并获取main\_wait和idle\_wait两个锁。 | |  | 求sum=ΣCi/Ti，用来判断是否能够成功调度。 | |  | 循环用来模拟系统的运行和任务调度。每个循环代表一个时间单位。  按照调度算法选一个线程运行，如果返回-1，也就是不存在能够调度的线程，那么选择闲逛线程。否则释放对应线程的锁，让其运行一个时间片。  这里不管是选择闲逛线程还是任务线程，都需要释放对应的锁，然后给主程序上锁。当闲逛线程或任务线程执行完一个时间周期后，会释放主函数锁main\_wait，让主函数继续执行，然后锁定自己的线程，等待主函数唤醒。 | |  | Select\_proc()函数，每次选择一个任务序号返回。对于EDF算法，就选一个剩余时间ti\_left最短的；对于RMS算法，选择一个周期时间ti最短的。 | |  |  | |  | 如果是RMS调度，这里判断若上一个任务还是活跃状态（flag==2）就直接返回，继续执行该任务。只需要将这部分代码注释掉，就可以改为抢占式。 | |  |  | |  |  | |  |  | |  |  | |  |  | |  |  | | |
| **实验结果分析与体会：**   |  |  | | --- | --- | |  | 2个任务，执行EDF调度方法，结果和预期相符。 | |  | 3个任务，执行RMS调度方法，结果和预期相符。 | |  | 3个任务，执行RMS调度方法，结果和预期相符。 | | |
| **思考题：**   |  | | --- | | （1）如何实现抢占式的RMS算法？  **在RMS方法下，选择下一个调度任务时，不考虑前一个任务是否活跃，总是从所有活跃任务中选择一个周期最小的进行调度即可。因此只需要注释掉图中部分即可。** | |  | | |