Chapter 8

王拓为 2018011917

一、实现

- 在 ProcessControlBlock 中添加 is_enable_deadlock_detect 表示是否开启死锁检测，以及分别对应 mutex 和 semaphore 的三个数据结构 available，allocation 和 need
- 在 sys_thread_create 中初始化 available，allocation 和 need

  在 sys_mutex_create 和 sys_semaphore_create 中更新 available，allocation 和 need
- 在 sys_mutex_lock 和 sys_semaphore_down 中执行死锁检测算法同时更新相关数据结构
- 在 sys_mutex_unlock 和 sys_semaphore_up 中更新自身和等待队列中待唤醒线程的相关数据结构
- 在 sys_enable_deadlock_detect 中设置 is_enable_deadlock_detect 的值

二、问答

1. 需要回收的资源有：
   - 对应进程的 ProcessControlBlock
   - 主线程和其他线程的 TaskControlBlock
   - 地址空间
   - 文件描述符表

   其他线程的 TaskControlBlock 可能在主线程的 ProcessControlBlock 的 tasks 中被引用，也可能在进程的地址空间中被引用；

   需要首先在回收地址空间前回收各线程的 TaskControlBlock，防止重复释放。
2. 在 Mutex1 的实现中，当前线程在移交等待队列中的其他线程前就在内核态将锁的状态设为解锁，可能出现两点问题：
   - 等待队列中的线程在释放锁时会出现 assert 报错的情况，因为此时锁的状态为解锁；一种解决的方法是修改 lock 实现补上上锁操作；
   - 此外锁在移交后本应是上锁的状态，但此时在内核态中实际为解锁状态，会出现状态不统一的问题。