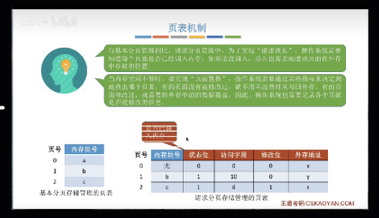
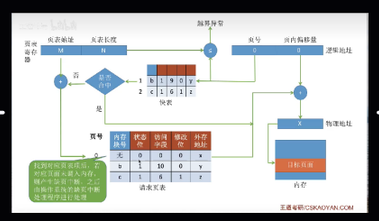
请求分页管理方式

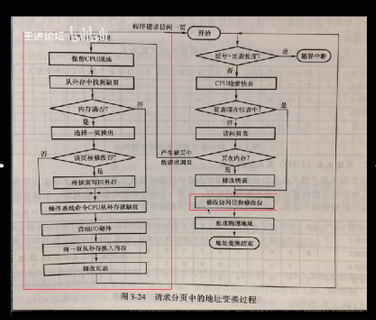
- 新功能
  - 请求调页功能 —— 缺失的页面从外存调入内存
  - 页面置换功能 —— 暂时用不到的页面换出外存
- 页表机制
  - 思路



  - 请求页表
    - 内存块号
    - 状态位 —— 是否调入内存
    - 访问字段 —— 记录最近访问几次，上次访问时间，供置换算法选择换出页面
    - 修改位 —— 调入内存后，是否被修改
    - 外存地址 —— 页面在外存中的位置
- 缺页中断机构
  - 例子
    - 访问的页面不在内存中，产生缺页中断，操作系统进行中断处理
    - 此时缺页的进程阻塞，放入阻塞队列。调页完成后再将其唤醒，放回就绪队列
    - 如果内存中有空闲块，为进程分配一个空闲块，所缺页面放入该快，修改页表中相应的页表项
    - 内存中没有空闲块，页面置换算法选择一个页面淘汰，若该页面被修改过，写回外存。没有修改，无需写回外存。
  - 缺页中断
    - 指令想要访问的目标未调入内存产生的，内中断 —— 输入故障 fault 可被修复
    - 一条指令执行中可能产生多次缺页中断
- 地址变换机构
  - 新增步骤
    - 请求调页 —— 查到页表项时进行判断
    - 页面置换 —— 需要调入页面，但没有空闲内存块
    - 修改页表中新增的表项
  - 过程



    - 快表中的页面一定是在内存中的
      若某个页面被换出外存
      快表中的相应表项也要被删除
      否则会出现访问错误的页面
  - 流程图



    - 只有执行写指令，才需要修改 修改位。 —— 一般来说只需要修改快表中的数据，只有快表项被删除后才需要写回内存中的慢表
    - 产生缺页中断后，需要保留CPU现场
    - 内存满了，需要选择页面换出 —— 页面置换算法
    - 换入 换出页面 启动I/O操作(慢速) —— 不能太频繁
    - 页面调入内存后，需要修改慢表，同时复制到快表
- 回顾与思考