

2017zyw

by 张益玮

一

1. x address物理内存地址
2. x frame pointer是为了debug的时候能够trace，没有也行
3. x 不共享寄存器
4. v
5. v
6. x 子进程0 父进程childPID
7. v
8. v
9. x 硬件只设置TLB 页表由操作系统设置
10. x 先段表后页表
11. v
12. x 时钟中断
13. v
14. v 必须有MMU
15. x x86IDT存着段描述符
16. v
17. x 不是一定段页式
18. x 用户态有没有异常处理机制，没有就杀死
19. v
20. v

二

1. $CPL \leq DPL[i]$ & $CPL \geq DPL[段]$; $MAX(CPL, RPL) \leq DPL[段]$
2. 创建; 就绪; 运行; 等待; 退出; 挂起; 等待挂起; 就绪挂起

三

1. (1) a.尽可能减少页面的换入换出次数 b.将未来不再访问或者短期内不访问的页面换出
(2) 在页表项中增加访问位，用来记录页面访问情况。页面换入内存时，将访问位初始化为0，访问页面的时候将访问位置为1，缺页时，从指针当前位置开始顺序检查环形链表，访问位为0则置换该页，为1则将访问位置0并将指针移动到下一个页面。
(3) LRU: 7 CLOCK: 7
(4) LRU、OPT有 CLOCK、FIFO没有
2. (1)不返回，`exit()`，结束进程
(2)1个返回值，`fork()`，用来将一个进程复制为2个进程，子进程的`fork()`返回0，父进程的`fork()`返回子进程的PID
(3)2个返回值，`ssize_t read(int fd, void *buf, size_t count)`，用来从一个文件中读取指定字节数的数据，buf中返回了读取的数据，同时返回了实际读取的字节数

3. (1) 虚拟地址为15位，最低5位是页内偏移。从PDBR中获得页目录表的基地址，加上虚拟地址的最高5位得到了对应的1Byte的页目录项地址。该页目录项中的VALID位为1，则将页目录项中的低7位左移5位，得到对应的页表基地址；VALID位为0进行缺页异常处理。该基地址加上虚拟地址的中间5位得到了对应的1Byte的PTE的地址。该PTE中的VALID位为1，则将该页表项的第7位左移5位得到物理页帧号，物理页帧号加上页内偏移就得到了物理地址；VALID位为0进行缺页异常处理。

(2) 0x4a10: 对应物理页:0xcd0, pde_index:0x12, pde_content:0xa7, pte_index:0x10, pte_content:0xe6

0x1ebe: 对应物理页:无(非法), pde_index:0x7, pde_content:0xe8, pte_index:0x15, pte_content:0x7f

0x135c: 对应物理页:0x4bc, pde_index:0x4, pde_content:0xe2, pte_index:0x1a, pte_content:0xa5

(3)

```
char translate(int16 virtualAddr){
    int16 offset = virtualAddr&0x1f;
    int16 pteIndex = virtualAddr&0x3e0;
    int16 pdeIndex = virtualAddr&0x7c00;
    char pdeContents = memory[PDBR+pdeIndex];
    if(pdeContents&0x80 == 0)
        return 0x81;
    int16 pageTableBase = (pdeContents&0x7f)<<5;
    int16 pteContents = memory[pageTableBase+pteIndex];
    if(pteContents&0x80 == 0)
        return 0x80;
    return (pteContents&0x7f)<<5+offset;
}
```