习题二

1.思考题

（1）什么是程序状态字（PSW）？它有何作用?

答案：程序执行的处理器现场的动态行为，用于记录当前运行程序的动态信息。

（2）为什么将机器指令分成特权指令和非特权指令？

答案：从系统资源管理和保护的角度出发

（3）试根据中断请求信号的来源对中断事件进行分类。

答案：

硬件故障中断

程序性中断

自愿性中断

I/O中断

外部中断

（4）什么是进程？操作系统为什么要引入进程的概念?

答案：进程是具有独立功能的程序在某个数据集上的一次执行活动，也是操作系统进行资源分配和保护的单位

（5）进程最基本的状态有哪些？哪些事件可能引起不同状态间的转换?

答案：就绪，运行和等待（阻塞）

**运行态**

**出现等待事件**

**被调度程序选中**

**落选**

**等待事件结束**

**就绪态**

**等待态**

注意：落选的事件可能是因为时间片到，也可能是出现高优先权的进程等

（6）什么是进程控制块（PCB）？它包含哪些基本信息?

答案：

1. 进程控制块PCB是OS用于记录和刻画进程状态及环境信息的数据结构
2. 借助PCB，OS可以全面管理进程的物理实体，刻画进程的执行现状，控制进程的执行
3. 进程控制块PCB，是进程所有信息的聚集

进程控制块中的信息：

1. 标识信息：唯一的标识一个进程，包括用户使用的外部标识符和系统使用的内部标识符。
2. 现场信息：进程运行时的处理器现场信息。常用的现场信息：通用寄存器、控制寄存器、用户堆栈指针、系统堆栈指针等
3. 进程控制信息：进程状态，进程优先级，进程调度中与调度算法有关的信息，事件信息，程序和数据的地址，进程同步和通信机制，资源清单，连接指针等

（7）什么是进程上下文？试述其主要内容。

答案：

进程物理实体和支持进程运行的环境（硬件寄存器、程序状态字、页表等）合称为进程上下文。

进程上下文由以下三部分组成：

用户级上下文：由用户程序块、用户数据块和用户堆栈组成的进程地址空间。

系统级上下文：包括进程的控制块（进程的状态）、内存管理信息、核心栈等操作系统管理进程所需要的信息组成。

寄存器上下文：由PSW寄存器和各类控制寄存器、地址寄存器、通用寄存器等组成。

（8）什么是进程切换？什么是模式切换?试述两种切换的主要步骤并对他们进行比较。

答案：

进程切换：保存老进程的上下文，装入新进程的上下文，以便新进程运行。进程切换发生在内核态。

进程切换切换步骤：

（1）保存被中断进程的处理器现场信息，修改被中断进程的进程控制块的有关信息，如进程状态等，把被中断进程的进程控制块加入有关队列

（2）选择下一个占有处理器运行的进程，修改被选中进程的进程控制块的有关信息，根据被选中进程设置操作系统用到的地址转换和存储保护信息，根据被选中进程恢复处理器现场

模式切换包括正向模式切换：用户态到内核态和逆向模式切换：内核态到用户态。正向模式切换由中断/异常/系统调用中断用户进程执行而触发； OS执行中断返回指令将控制权交还用户进程而触发逆向模式切换。

处理器模式切换工作流程

中断装置完成正向模式切换，包括：

处理器模式转为内核模式

保存当前进程的PC/PSW值到核心栈

转向中断/异常/系统调用处理程序

中断返回指令完成逆向模式转换，包括：

从待运行进程核心栈中弹出PSW/PC值

处理器模式转为用户模式

（9）什么是内核级线程？什么是用户级线程?什么是混合式线程?试对他们进行比较。

答案：

内核级线程KLT, Kernel-Level Threads：线程管理的所有工作由OS内核来做；OS提供了一个应用程序设计接口API，供开发者使用KLT；OS直接调度KLT。

用户级线程KLT, Kernel-Level Threads，用户空间运行的线程库，提供多线程应用程序的开发和运行支撑环境；任何应用程序均需通过线程库进行程序设计，再与线程库连接后运行；线程管理的所有工作都由应用程序完成，内核没有意识到线程的存在。

混合式线程：线程创建是完全在用户空间做的，单应用的多个ULT可以映射成一些KLT，通过调整KLT数目，可以达到较好的并行效果。

对比：

内核级线程KLT的优点：能调度同一进程的其它线程占有处理器运行，多处理器环境中，内核能同时调度同一进程中多个线程并行执行，内核自身也可用多线程技术实现，能提高操作系统的执行速度和效率

内核级线程KLT的缺点：

应用程序线程在用户态运行，线程调度和管理在内核实现，在同一进程中，控制权从一个线程传送到另一个线程时需要模式切换，系统开销较大(用户态-内核态-用户态)

用户级线程KLT的优点：

所有线程管理数据结构均在进程的用户空间中，线程切换不需要内核模式，能节省模式切换开销和内核的宝贵资源

允许进程按应用特定需要选择调度算法，甚至根据应用需求裁剪调度算法

能运行在任何OS上，内核在支持ULT方面不需要做任何工作

用户级线程KLT的缺点：不能利用多处理器的优点，OS调度进程，仅有一个ULT能执行；一个ULT的阻塞，将引起整个进程的阻塞

混合式线程将会结合纯粹用户级线程方法和内核级线程方法的优点，同时减少它们的缺点

2.应用题

（1）指出下列指令中的特权指令和非特权指令：①读时钟日期；②陷入指令；③设置时钟日期；④加载PSW；⑤置特殊寄存器；⑥启动1/0指令。

特权指令：3,4,5,6

非特权：1,2

（2）画2.1和2.2节的思维导图。

（3）百度并阅读《我是一个线程》，对比分析进程与线程。

进程与线程的比较

1、管理进程的开销大

2、线程是进程的一部分，是进程中的执行体

3、进程拥有资源，线程只能从父亲（进程）那里获取资源

4、进程（父）的CPU归线程（子）所有

5、并发时，线程（子）更好