嵌入式 刘海涛

单线程程序模型（轮询）

多线程程序模型（处理过程并行 缺点：资源竞争）

轮询：有一个有限的最差时间

无优先级，序列化运行

响应时间：所有任务的和

变化的影响比较大，改变任务的执行时间或增加任务会影响所有任务

简单，没有数据共享问题

有限状态机（计算机、通信领域中协议的实现 状态图）

类似于轮询，但只有当前状态得到执行

每一个状态决定下一个状态（非顺序执行）

优先级：每个状态决定下一个状态的优先级

响应时间：所有任务总和

变化影响：显著（同上）

简单，无共享数据问题

有中断的轮询

优先级：中断

响应时间：所有任务的执行时间+中断执行时间

变化影响：对中断部分影响较小，轮询部分影响同上

共享数据（中断服务进程）

实时性高的采用中断，实时性不高的进行轮询

补充：前后台系统 只能用于单用户，不能用于高速i/o设备，中断开销太大

函数队列任务：

函数指针被加入到队列中

主循环在队列中进行循环并执行任务

任务或中断在函数队列中加入新的任务

函数队列改进：

抢占式多任务、多线程

微内核microkernels

建立一个实时操作系统的方向

每个系统的功能看做服务，其他需要使用功能的

可扩展性、移植性、维护性（性能上不如linux单核结构）

单内核

系统服务平等地位互相调用（网络）

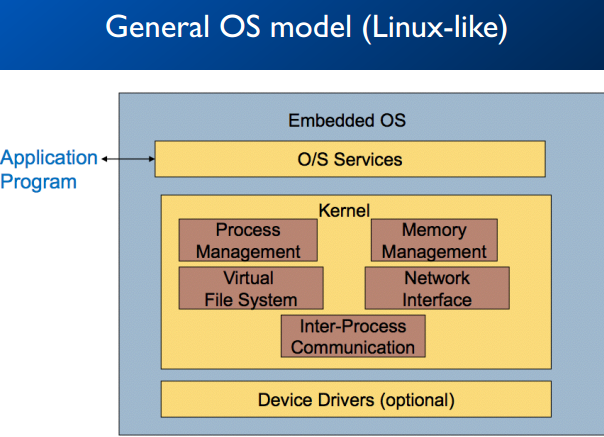
性能最高，组织混乱，可移植性较差

层次化

Io模式、底层、硬件相关/无关、只需要关注接口

实时操作系统（RTOS）

通用操作系统（GPOS）的体系结构



RTOS

1.保证在一定时间限制内完成特定功能的操作系统

当外界时间或数据产生时，能够接受并以足够快的速度予以处理，其处理的结果又能在规定时间内控制生产过程或对外处理 作出快速响应，并控制所有任务协调一致运行的嵌入式操作系统。

2.特点：高精度计时系统

多级中断机制

实时调度机制

3.选取原则：任务调度机制

内存管理、务实模式与保护模式

最小内存开销

时间阻碍：如果一个进程没有在deadline之前结束：

对于硬件deadline: 系统会失败

软件deadline： 用户会获得提醒，但是系统不一定会失败

实时系统中任务调度

调度目标：所有任务都在deadline之前完成

嵌入式&通用调度

通用调度的目标：提供最大存储量，使系统性能保持最优，让当前 合理使用资源

嵌入式目标：（与通用最大的差别：可预测性）系统行为是可知的

操作系统进程管理：多进程（MMU、MPU为其 硬件支持）

多任务操作系统（栈、任务控制块（TCB））

栈的大小自己设定

任务控制块（TCB）task activation records

派生新进程的时机：抢占式、非抢占式

如何决定调度哪个进程

抢占/非抢占

周期/非周期

固定优先级/动态优先级

优先级反转

调度策略度量标准

CPU利用率

满足所有deadline的能力

调度开销（时间复杂度，越复杂，越能提高利用率，但是开销会更大）

延迟（响应时间-实现时间）可行调度，这个值<=0

总完成时间

抢占式调度（Preemptive Scheduling）

静态分配（县城的整个过程中都是固定不变的）

动态分配（执行过程中可以动态改变）

RMS（rate Monotonic scheduling）

Ti表示每个任务的周期，任务之间没有依赖

Worst-case execution times(WCET) C1, …Cn（假设没有互斥，不阻塞I/O）响应时间

没有优先级约束

固定优先级

假设上下文切换没有时间

抢占式调度

按照当前周期，来设定优先级，周期最小的任务具有最高的优先级->可行调度

RMS是可行性最优的

提示：操作系统中对不同任务的优先级如何设定

RMS的可行性：每一个任务只要在它的执行时间，都可以执行完毕

rMS说明：只针对可行性最优 chap5 page69

实现：①时钟中断（中断周期：所有任务周期的最大公约数）

②多任务？？？

利用率（70%左右）

最早最？？

Deadline Driven Scheduling

1.Earliest Due Data：EDD针对任务急，按照n个任务之间截止时间大小来设定优先级



特点：具有最小 最大延迟，对任务急中的所有任务，最大延迟最小化

2.Earliest deadline first(EDF)最早截止时间优先：n个任务，任意到达（到达时间随机），已到达任务中最早的实现 优先级 最高

这是一个动态的

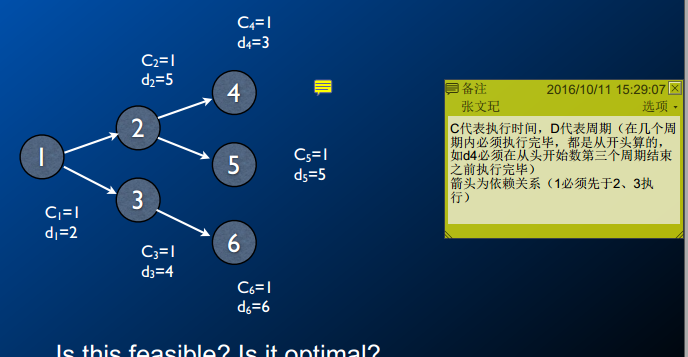
最优动态调度算法

可用于周期性调度算法

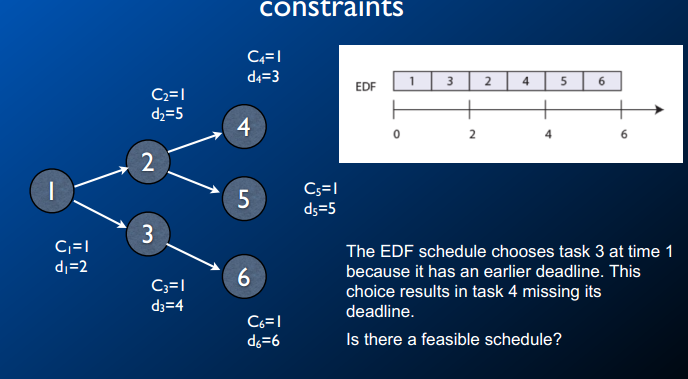
CPU利用率几乎100%

RMS&EDf比较

RMS复杂度较低，EDF性能较好（70%<100%） chapter5 page77

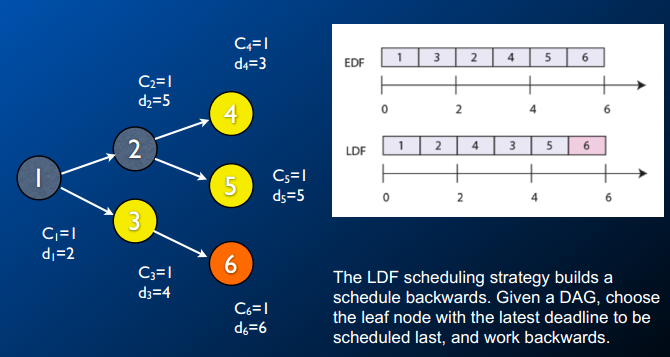


EDF方法：



4必须在第三个周期结束前执行完毕，所以这个方案是不可行的

改进；LDF（Latest Deadline First））



如果两个任务截止时间一致，都可以

LDF在最大延迟最小化中是最优的

它不要求抢占式（EDF是抢占式的）

静态的，必须在所有任务执行前知道所有的进程

改进二：EDF with prcedences

资源占用太多

优先级反转：

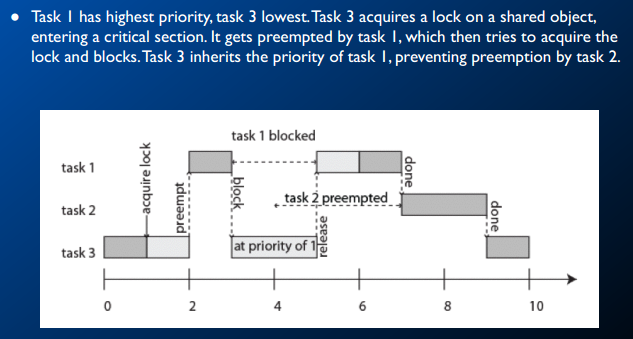
高优先：从共享内存中检索数据

中：交互任务

低：收集气象数据的线程

解决优先级反转的问题：

优先级继承(Priority Inheritance Protocol PIP)：当低优先级持有资源而阻塞高优先级任务时，它会暂时获得阻塞队列中的最高优先级，从而尽快执行完毕释放资源，使高优先级可以尽快的使用

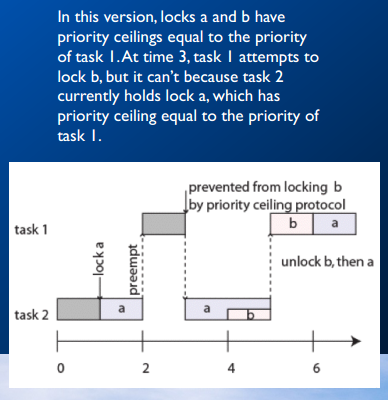


难以解决：死锁问题

解决方案二：优先级天花板Priority Ceiing Protocol(PCP)

每个锁或共享量 会被分配一个与最高优先级任务相同的优先级 作为一个优先级天花板

一个任务可以获得一个锁当且仅当它的优先级严格高于其它左右当前被持有的锁的优先级天花板



基于优先级的调度时 “贪心策略（每个处理器都只看自己的局部最优，不看全局）”

μC/OS-II

空闲任务/统计任务

任务控制块TCB（栈指针+链表指针）

时描述一个任务的核心数据结构，存放了它的各种管理信息

所有的任务控制块分属于两条不同的链表，，单项的空闲链表和双向的使用链表

任务级的任务切换

μC/OS-||作业

1.动态优先级

2.按照到达时间确定作业紧急程度

增加对于EDF的支持

TCB扩展

最小侵入式修改

创建周期为p，执行时间为c的任务

？？？？？

Chapter7 model

建模、设计、分析（完善系统模型，修正系统设计）

什么是建模：通过模拟来整个理解系统的特性

组成系统的元素以及元素之间的关联

好的模型：简单

经得起检验，利用仿真方式通过相关理论来验证当前设计

更高的表现力

可执行

可整合

。。。

典型模型：面向状态的（适合实时控制的）：有限状态机

面向活动的：数据流：

面向结构的

面向数据的：实体关系图

异构的模型（前四个的组合模型）

基于模型的设计：设计嵌入式系统的数学模型：

常见：对于控制对象和被控对象分别进行建模

模型：为实现而服务 目标：自动化

分类：常规vs事件驱动

实时

反应式

模型、语言和工具

Chap7 page 20

有限状态机：

完全指定

区分米粒机和摩尔机：摩尔机：输出完全取决于当前状态

米粒机：输出取决于当前状态和事件

等价的有限状态机：在同样输入的情况下产生同样的输出（结构不同），实用价值：可以互换，简化

优化：时间复杂度和 状态和转换的数目相关，可以使用等价的思想来优化

非确定的有限状态机

优势：形式更简单，行为不确定

用途：环境建模

摩尔机：不依赖输入，只基于当前状态，当前输入没有作用于当前状态中，实时性较差，状态数较多，实现更简单

米粒机：当前输入影响当前输出，更简洁

层次化的有限状态机

缺省状态：第一次跳入状态机时要有起始

历史状态：保留上一次跳出的状态

缺省和历史保留的都是状态机中的一个子状态

条件迁移



Event：触发事物的事件

Condition：监管事物

Action：当事物完成时的操作

每个都是可选的

Chap7 page65

交换机协议

作业：用有限状态机设计