Identification and Analysis of Performance Metrics for Real Time Operating System

组　员：

报告人：
时间：2011/5/30

在嵌入式软件设计和集成中，实时多任务操作系统的性能分析是至关重要的，它需要保证实际应用的时间限制得到满足，即是运行时间不能超过实际应用的截止时间限制。

为了选择满足用于特定应用的嵌入式系统的一个适当的操作系统,我们需要对操作系统服务进行分析，这些操作系统服务是由形成性能指标的参数确定的。

既定的性能指标包括:

上下文切换时间
抢占时间
中断延迟
信号混洗时间

什么是嵌入式系统

一个嵌入式系统是由硬件、软件和其他为了执行具体功能而设计的机械部件所组成，嵌入式系统通常包含可编程处理器和外围设备，以及特定的应用所需的硬件。底层硬件的应用复杂性，严格的性能和功率预算，以及开发进度都要求应用程序开发人员使用执行时支持软件。这种支持通常采用RTOS，运行库和设备驱动程序的形式。 RTOS被应用于包含软实时限制的嵌入式系统中，以及包含硬实时限制的正式实时系统。

RTOS为创建一个应用程序向嵌入式系统设计师提供了一系列服务，如任务管理，内存管理，以及资源管理。

什么是实时操作系统

定义：实时操作系统是一个程序，它按照时序方式调度执行任务，并管理系统资源，为开发应用程序代码提供一致的基础。

嵌入式系统的一个子类是实时嵌入式系统，实时系统即是有时间限制的系统。

实时系统的性能依据是能否及时运算并做出决策的能力，这些运算具有完成截止时间要求，错过了截止时间和得出错误结果是一样糟糕的。由这个错误造成的损失程度取决于应用程序。例如，当一个实时系统是飞机飞行控制系统的一部分，仅错过截止时间就足以危及乘客和工作人员的生命安全。

什么是实时操作系统

操作系统并不是任何一个计算机系统必须的组成部分。一个简单的微波炉并不需要操作系统，但是，随着应用的复杂性逐步扩展，应用操作系统的好处远远大于其成本。

RTOS的使用能通过任务分解进一步简化设计，通过抢占式处理，保证关键任务能够及时有效地得到处理。RTOS允许通过提供诸如信号量，邮箱机制，队列，时间延迟，时间戳等服务来更好的利用系统资源。

Components of RTOS

大部分RTOS内核组成：
调度程序(Scheduler)
对象(Object)
服务(Services)

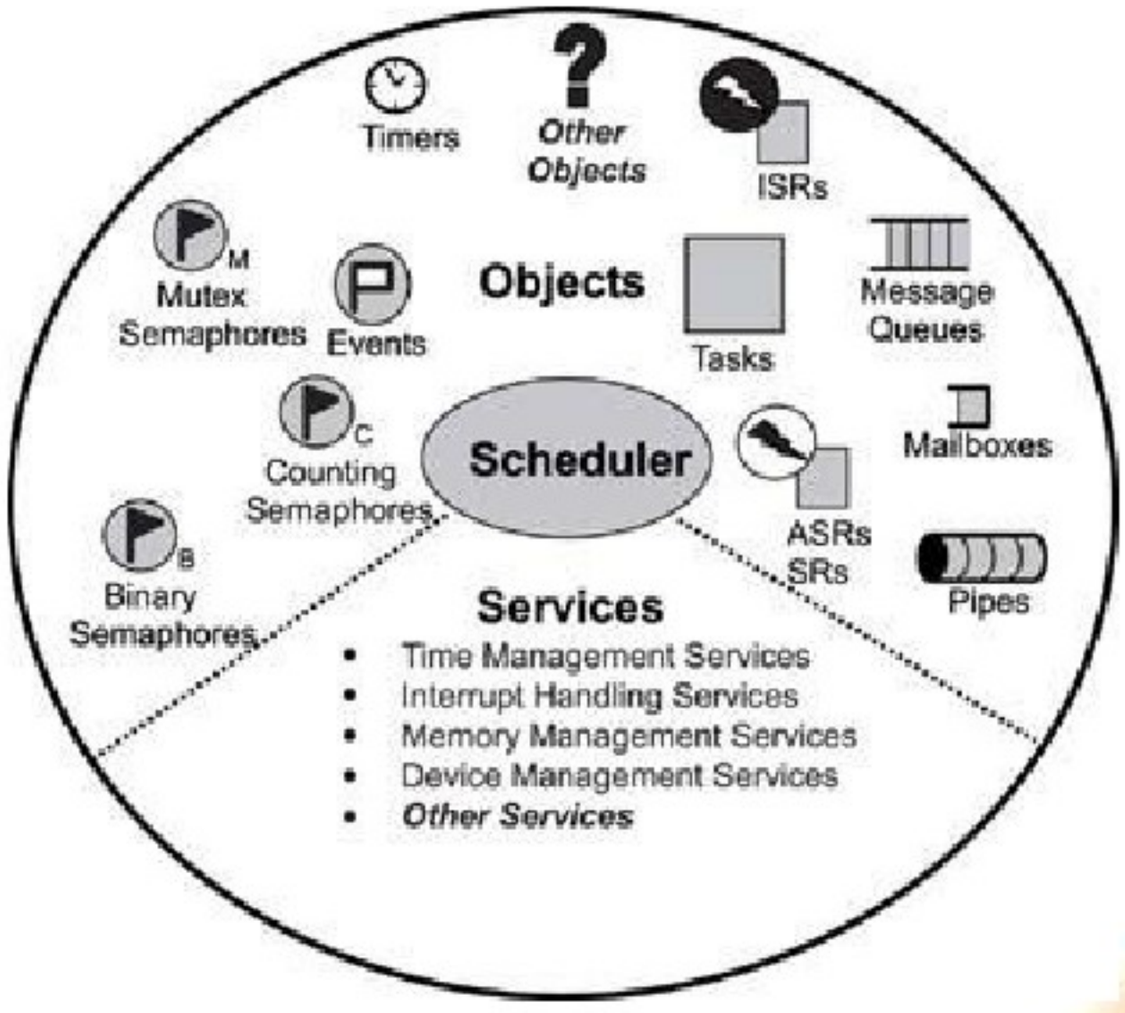


Figure 1: Components

Components of RTOS

调度程序

调度程序位于每个内核的核心部分，它提供算法来决定何时执行哪一个任务。

对象

最常见的RTOS内核对象有：

任务——是指在执行时能够竞争CPU执行时间的并发独立的线程。

信号量——一种令牌状对象，任务数的增加或减少进行同步或互斥执行。

消息队列——就像数据结构缓冲区，可用于通过互斥、同步和数据交换方式进行任务之间信息的传递。

Components of RTOS

服务

大部分的内核提供服务，帮助开发人员为实时嵌入式系统创建应用软件，这些服务包括API函数调用集，可用于对内核对象的操作，或者用于促进下述服务：

(1) 定时器管理
(2) 中断处理
(3) I/O设备
(4) 内存管理

嵌入式系统可用于各种各样的应用软件。这些应用软件主动或被动地依赖于接口、可扩展性、连接性等需求。如何选择适合嵌入式系统的OS依赖于OS自身性能分析和应用需求。

基准测试(Bench Marking)

基准测试是指通过设计科学的测试方法、测试工具和测试系统，实现对一类测试对象的某项性能指标进行定量的和可对比的测试。

基准的分类:

(1) 复合基准测试 Synthetic Benchmark

(2) 应用基准测试 Application Benchmark

(3) 派生基准测试 Derived Benchmark

基准测试(Bench Marking)

(1) 复合基准测试 Synthetic Benchmark

复合基准测试用于测量系统、处理器、编译器的多方面特性。复合基准试图模拟出真实世界中的应用的指令混合。它主要用于调试特定功能，但是它们并不涉及应用软件中的功能如何实现。所以它不是必须的基准测试。

基准测试(Bench Marking)

(2) 应用基准测试 Application Benchmark

应用基准也被称为“真实世界”基准点，它使用系统软件或者用户层软件代码形成真正的算法。应用基准在系统层基准测试中更常用，通常具有大量的代码和数据存储需求。

基准测试(Bench Marking)

(3) 派生基准测试 Derived Benchmark

派生基准也称为“基于算法的基准点”，它是前两种基准点的折中。顾名思义，派生基准是通过提起关键算法，从实际应用中生成真实数据集，这避免了 执行整个应用程序的需要，派生基准可以用于调试、内部构造，以及对比分析。派生基准基于实际应用，尤其适用于嵌入式环境。

实时系统的性能基准

一系列的“基准“适用于一部分RTOS。他们的目标是各不相同的，有些注重嵌入式微控制器I/ O处理，有些试图测量台式电脑的浮点性能。这里简要介绍四个测量基准方法：

(1)Whetstone

(2)Dhrystone

(3)Hartstone

(4)Rhealstone

Whetstone

Whetstone是综合性测试程序，主要包括浮点运算、整数算术运算、功能调用、数组变址、条件转移、超越函数。测试结果用单位Kwips表示，1Kwips表示机器每秒钟能执行1000条Whetstone指令。

Whetstone基准可用于嵌入式系统和编译器的比较。

Dhrystone

Dhrystone是测量处理器运算能力的最常见基准程序之一，常用于测量指令集执行时间。该指令集被认为是高级语言中调用函数集的典型代表。指令之间的分布是：53%用于分配，32%用于控制，15%用于程序和函数调用。

缺点：Dhrystone的重心是并不常用于嵌入式系统中的字符串任务。

Hartstone

Hartstone性能基准程序是用Ada语言编写，是依据硬实时测试需求构成的一组合成应用。Hartstone性能基准程序需要不同测试序列，每个测试序列中的测试结果只有2种情况：满足或者不满足时间限制。若任一个程序不满足时限，那么测试结论就是“不通过”。共有5种测试序列：

A. 任务周期有规律的周期性任务；
B. 任务周期无规律的周期性任务；
C. 任务周期有规律但非周期性处理的周期性任务；
D. 任务周期有规律有同步的周期性任务；
E. 任务周期有规律有同步但非周期性处理的周期性任务。

Rhealstone

Rhealstone性能基准程序是对实时系统的六个关键操作的时间量进行操作，这六个关键操作是：上下文切换时间、抢占时间、中断延迟、信号量洗牌时间、死锁解除时间、信息传递延迟。这六项操作作为Rhealstone的六个组件，每个组件被单独测量，然后将经验结果合并为单一的测量值，即是Rhealstone值。

有两种方式测量Rhealstone值：
一是通用Rhealstone；
二是每个组件应用于具体应用程序的特定Rhealstone。

Rhealstone有两个缺点：
一是它测量的是平均时间，而不是最坏值；
二是其最后的结论是加权平均值，没有给出确定权值的依据。

决定RTOS性能的重要指标

一个RTOS的实时性能的主要评测指标包括：

A. 上下文切换时间 **Context Switching Time**；

B. 抢占时间 **Preemption Time**；

C. 中断延迟时间 **Interrupt Latency**；

D. 信号量混洗时间 **Semaphore Shuffling Time**。

上下文切换时间

上下文切换时间也称任务切换时间(task switching time)，定义为系统在两个独立的、处于就绪态并且具有相同优先级的任务之间切换所需要的时间。它包括三个部分：

保存当前任务上下文的时间；
调度程序选中新任务的时间；
恢复新任务上下文的时间。

切换所需的时间主要取决于保存任务上下文所用的数据结构以及操作系统采用的调度算法的效率。任务切换是任一多任务系统中基本效率的测量点，它是同步的，非抢占的，实时控制软件实现了一种基于同等优先级任务的时间片轮转算法。影响任务切换时间效率的因素有：主CPU结构，指令集以及CPU特性。

上下文的保存对象为CPU 寄存器的内容，切换时数据会保存在任务自己的堆栈内。上下文恢复的操作正好相反。保存的寄存器数量越多，上下文切换的工作量越大。因此，不应以单位时间内可做的上下文切换次数来衡量此项指标。上下文切换时间是系统实时性能的一个重要指标，通常约为1 μs。

上下文切换时间

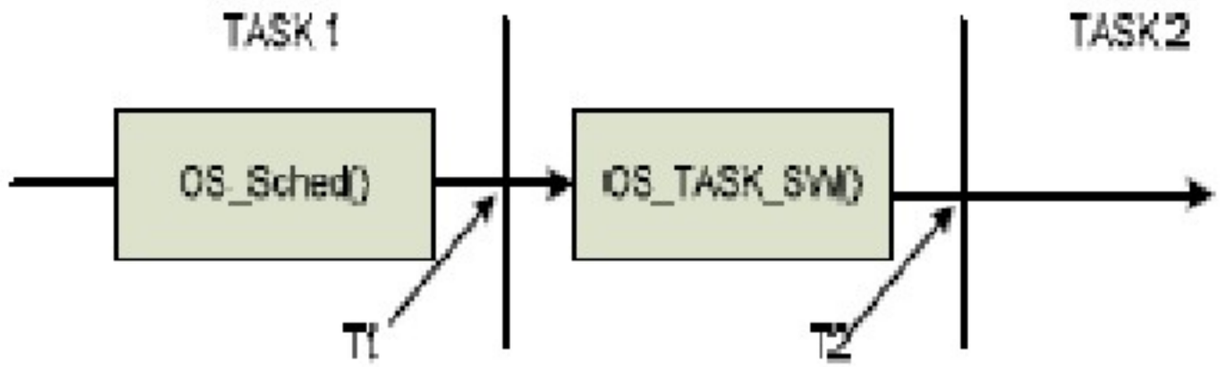


Figure 2: Context Switch Time

T1 = TIME START FOR FUNCTION “OS_TASK_SW()”
T2 = TIME END FOR FUNCTION “OS_TASK_SW()”
CST= T2-T1

抢占时间

抢占时间即系统将控制权从低优先级的任务转移到高优先级任务所花费的时间。为了对任务进行抢占，系统必须首先识别引起高优先级任务就绪的事件，比较两个任务的优先级，最后进行任务的切换。

它和任务切换有些类似，但是抢占时间通常花费时间更长。这是因为执行中首先要确认唤醒事件，并评估正在运行的任务和请求运行的任务的优先级高低，然后才决定是否切换任务。实质上，所有的多处理任务可以在执行期间动态分配优先级，所以，抢占时间也是衡量实时性能的重要指标。

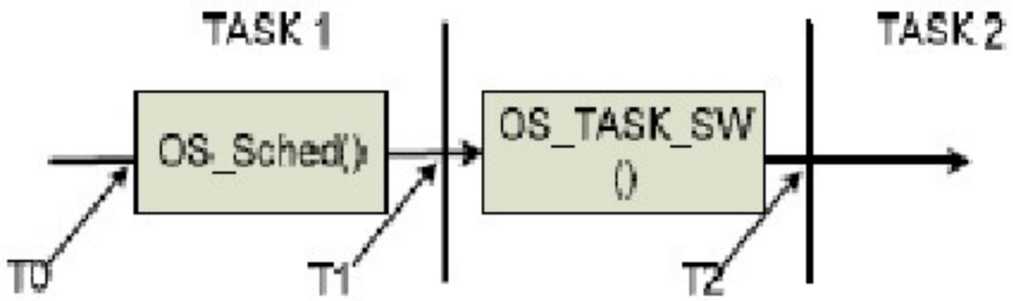


Figure 3: Preemption Time

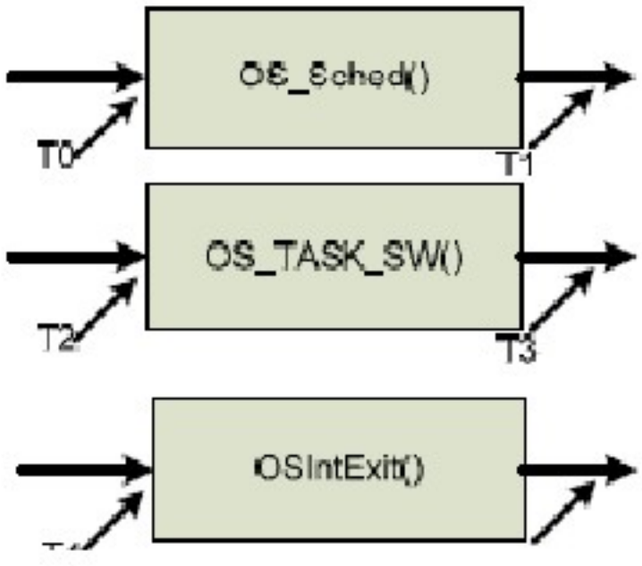
PT = T2 – T0

中断延迟

中断延迟时间是指从接收到中断信号到操作系统做出响应，并完成进入中断服务例程所需要的时间。多任务操作系统中，中断处理首先进入一个中断服务的总控程序，然后才进入驱动程序的ISR。

中断延迟时间＝最大关中断时间＋硬件开始处理中断到开始执行中断服务例程第一条指令之间的时间。

硬件开始处理中断到开始执行中断服务例程的第一条指令之间的时间由硬件决定，所以，**中断延迟时间的长短主要取决于最大关中断的时间**。硬实时操作系统的关中断时间通常是几微秒，而Linux最坏可达几毫秒。



Ta = T1-T0

Tb = T3-T2 **IL= Max(Ta,Tb,Tc)**

Tc = T5-T4

Figure 4: Interrupt Latency

信号量混洗时间

信号量混洗时间是指从一个任务释放信号量到另一个等待该信号量的任务被激活的时间延迟。在RTOS中，通常有许多任务同时竞争某一共享资源，基于信号量的互斥访问保证了任一时刻只有一个任务能够访问公共资源。

信号量混洗时间反映了与互斥有关的时间开销，因此也是衡量ERTOS实时性能的一个重要指标。

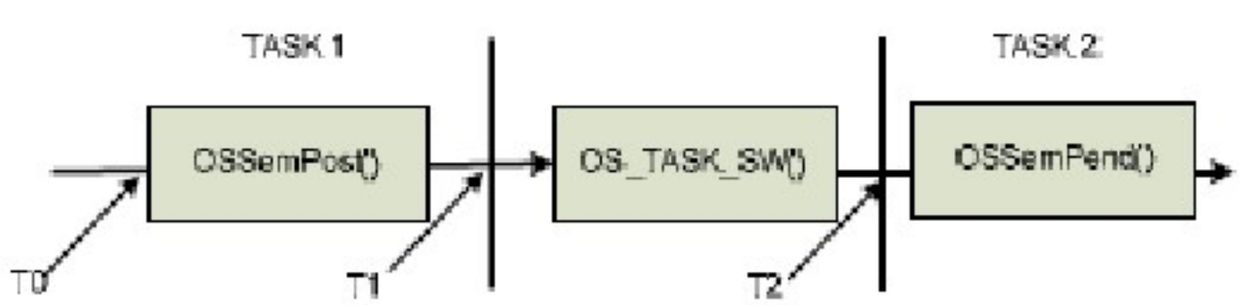


Figure 5: Semaphore Shuffling Time

SST = T0-T1

数据分析

TABLE 1
GENERATION OF METRICS NUMBER

Performance parameter	Observed values in microsecond	Number of times performance parameter occur	T X W	ΣTn Wn in microsec	METRICS NUMBER 1/ Σ Tn Wn
Context Switching Time	T1 =11	W1 = 38	418	24928	40.11
Preemption Time	T2 =15	W2 = 38	570		
Interrupt Latency	T3 = 18	W3 = 1286	23148		
Semaphore Shuffling Time	T4 = 24	W4 = 33	792		

结论：**Metric number**越大，则对应具体应用软件的操作系统越好。

发展方向

测试技术与设计技术的成熟度同比发展，随着设计技术的发展，相应的测试技术也需要有相应成熟的测试例程、特定应用领域的测试基准程序来作为测试的标准。可以预见，在未来几年里，测试基准程序和针对嵌入式计算机设计的性能测试基准程序，将会在系统设计过程中的更早阶段——系统级构架优化已基本性能度量方面取得更大的发展。

近十年来ERTOS得到飞速发展：从支持8 位微处理器到16位、32 位甚至64位；从支持单一品种的微处理器芯片到支持多品种微处理器芯片；从只有实时内核到除了内核外还提供其他功能模块如:文件系统TCP/IP，网络系统GUI 图行系统等等。目前嵌入式实时操作系统及其应用开发环境的发展动向是：

1. ERTOS正向实时超微内核开放发展；
2. ERTOS开发环境正向开放的集成化的方向发展。

THANK YOU