文章编号: 1005-3026(2006)04-0394-04

WebitOS内核的实现机制及性能分析

张希元[1], 赵 海[1], 孙佩刚[1,2], 罗玎玎[1]
(1. 东北大学 信息科学与工程学院, 辽宁 沈阳 110004; 2. 沈阳炮兵学院, 辽宁 沈阳 110162)

摘 要: 描述了WebitOS的体系结构,WebitOS采用模块化分层结构设计,包含设备驱动、实时内核及轻型TCP/IP协议栈等模块,功能完备;分析了WebitOS内核的实现机制:实时调度采用了基于优先级抢占的多任务调度机制,内存管理采用了最佳匹配的分配算法和边回收边整理的回收算法.在此基础上,从内核运行的时空开销、支持EI应用等角度对WebitOS内核的性能进行了测试.结果表明,WebitOS内核是一个实时、精简且高效的内核,特别适用于资源受限环境下开发嵌入式实时应用.

关 键 词: 嵌入式操作系统;硬实时;嵌入式Internet;抢占式调度;最佳分配算法
中图分类号: TP 316.2 **文献标识码**: A

设计嵌入式实时操作系统有着诸多困难,因为不仅要提供所要求的能力,还要提供可靠性和预见性,但这些都以吝啬地使用有限的资源为前提.现有的大多数嵌入式操作系统要么占用大量空间和资源,实时性不高;要么支持的协议模块少,功能单一[1~4].

WebitOS是辽宁省嵌入式技术重点实验室为实现非PC设备联网使用的因特网接入服务器[5](Internet access server,IAS)专门设计的嵌入式实时操作系统.WebitOS能够支持硬实时应用,占用小的程序空间和数据空间,采用C语言编写,便于开发和移植,特别适用于为各类嵌入式应用提供可靠的系统服务.

1 WebitOS的体系结构

WebitOS提供了一个抢占式实时多任务内核、设备驱动程序、TCP/IP网络协议栈和基于Flash的小型文件系统等功能模块.其体系结构如图1所示.

各层的功能模块是独立开发的,WebitOS为用户提供可调用的API函数.设备驱动层为WebitOS管理底层物理硬件提供了逻辑接口;实时内核中仅保留最基本、最重要的系统服务,主要完成任务管理、任务间通信、内存管理、时钟管理等功能;thin TCP/IP协议栈、文件系统和输入输出I/O管理是运行于系统内核之上的外围模块,thin TCP/IP协议栈为嵌入式应用提供网络通信支持,文件系统为嵌入式应用提供一定的文件管理能力,输入输出模块负责为用户提供I/O服务.将编译后的内核,下载到IAS的MCU中,占用不到4 KB的程序空间,较好地满足了嵌入式环境下资源受限的特点.

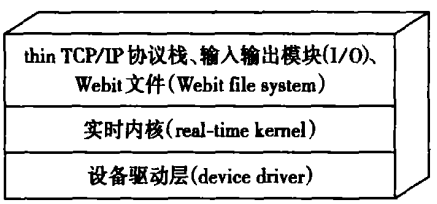

图1 WebitOS的系统结构
Fig.1 System architecture of WebitOS

2 WebitOS内核的实现机制

2.1 任务调度机制

WebitOS采用的是一种基于静态优先级的抢占式实时内核,它的调度机制是基于优先级的抢占式调度算法[6,7].

WebitOS最多支持256个优先级,对应0~255,数值越小,优先级越高,相同优先级的任务轮转调度.WebitOS中的任务切换是基于时钟/事件驱动的,定时器管理支持一次性和周期性任务的定时,分辨率为1 ms;使用事件机制来保证任务

收稿日期: 2005-05-23
基金项目: 国家发改委高技术产业化示范工程资助项目(20012167); 国家科技部火炬计划项目(2002EB010154).
作者简介: 张希元(1981-),男,辽宁铁岭人,东北大学博士研究生; 赵 海(1959-),男,辽宁沈阳人,东北大学教授,博士生导师.

间的同步和通信.

由于嵌入式环境下硬件资源的相对有限,WebitOS 中的任务创建采用静态创建方法:由应用开发人员预先定义好任务的堆栈大小和优先级等参数,系统在编译时为每个任务分配堆栈并创建任务控制块(task control block,TCB).值得注意的是任务堆栈的大小除了要满足用户要求以外,还应留出足够的空间为任务切换时保存程序计数器 PC 和环境寄存器变量[8],如果堆栈空间不足,在任务切换时会造成系统崩溃.

在 WebitOS 中,每个任务可能处于 5 种状态之一:就绪态(ready)、运行态(running)、挂起态(suspend)、睡眠态(dormant)及中断服务态(ISR running).在系统内核的管理下,各个任务可以实现运行状态的跃迁,如图 2 所示.

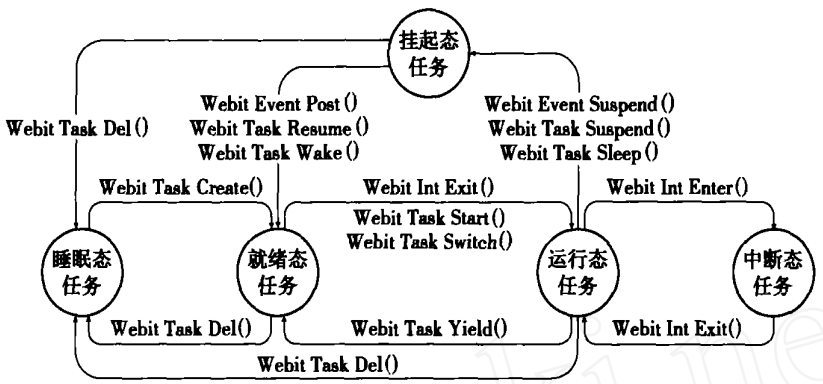

图 2　WebitOS 任务的状态跃迁

Fig. 2　State transition of WebitOS task

在 WebitOS 中,所有的就绪任务按照优先级由高至低的顺序排队,形成一个由 TCB 组成的就绪队列,一旦任务成为就绪队列中优先级最高的,便可在内核管理机制的调度下获得 CPU,进入运行态.为了任务调度的需要,运行态任务的 TCB 并不从就绪队列中删除,仍位于就绪队列的首部.正在运行的任务一旦遇到中断,就会进入中断服务态,如果因为等待某种资源、信号或消息而无法继续运行,便会转入挂起状态,处于挂起态的任务在获得所需的资源之后,会重新进入就绪态,当就绪队列中有比当前运行任务优先级更高的任务存在时,内核就会进行任务切换,将当前处于运行态任务的 CPU 寄存器值压入堆栈,然后将就绪队列中最高优先级任务的 CPU 寄存器值从堆栈恢复到通用寄存器中,从而完成将一个高优先级任务调入到内核中运行.

2.2　内存管理机制

对于嵌入式 Web 服务器(embedded web server,EWS)来说,由于资源受限,一般没有自己的内存管理单元(memory management unit,MMU),因此内存管理大多不采用虚地址寻址方式,每次分配给一个任务的内存空间必须是一个连续的、完整的一段空间.对于片内一般只有几 KB 存贮空间的 MCU 来说,如果没有一个简单、高效的内存管理算法,在多个任务运行一段时间之后,系统最终将会陷入瘫痪[9,10].

为了确保内存管理算法的有效性,同时兼顾系统的实时性,WebitOS 采用了最佳匹配的内存分配算法和边回收边整理的内存回收算法.

(1) 最佳匹配的内存分配算法

WebitOS 将内存中所有不连续的、未使用的内存块,用一个空闲链表链接起来.在每次任务创建时,根据任务所要求的内存块大小,遍历整个空闲节点链表,如果用户的需求对系统来说可以接受,那么内核总能把既满足要求、又是最小的空闲分区分配给任务,避免了"大块小用".为了加速查找,该算法将空闲链表按节点大小依次递增的顺序排列.这样,第一次找到的满足要求的空闲节点,必然是最佳匹配的,如果该匹配的节点在分配给任务后,仍然有较大剩余且剩余部分超过了规定的阈值(10 个字节),那么 WebitOS 就会将此节点进行分割,一部分分给申请空间的任务,剩余部分依照节点尺寸大小重新插入到空闲链表中的合适位置,以备下次分配时查找使用.

(2) 边回收边整理的内存回收算法

每当一个任务执行完毕,WebitOS 都要对该任务所释放的内存空间进行回收,采用的方法是边回收边整理.首先是检查所释放的内存块大小,将其按照节点块的大小链入到空闲链表中的相应位置,接着检查回收的这块内存节点是否和其前后相邻的空闲节点的地址连续,如果发现和其中的一个节点地址连续,或者和前后两个节点的地址都连续,则将其合并成一块大的空闲块,将此大

空闲块重新插入到空闲链表的恰当位置上去.简言之,回收时整理策略的重点是,在回收时须对节点进行检查与合并.此算法可以保证空闲链表在任何时刻都是规整的、最优的,在任何时候都不存在地址连续而没有被合并的节点,因此它能够极大地提高内存空间的利用率.

3 内核性能分析

针对嵌入式实时操作系统,要对其性能进行评估时,应考虑以下三方面的衡量尺度:①RTOS对应用的支持程度,包括内核调度机制、最大任务数,内核所占用的程序存贮空间和数据存贮空间的大小等;②RTOS的实时效率,如任务的切换延迟、中断延迟时间及内存分配的效率等.③RTOS主要协议模块(如设备驱动、thin TCP/IP协议模块)完成特定需求的能力,这能反映出实时系统的综合效能.

本文分别就上述几项衡量尺度,系统化地测量了对于嵌入式实时操作系统至关重要的几个性能参数.测量平台是基于实验室自主开发的IAS开发板,使用的MCU是Atmega128,处理器时钟频率为8 MHz.对每个测量项目,采样1 000次,以达到相应的准确性,测量的数据如表1～表4所示.

表1 WebitOS与μC/OS-Ⅱ的性能比较(相同的硬件平台)

Table 1 Comparison between WebitOS and μC/OS-Ⅱ with the same hardware platform

系统指标	WebitOS	μC/OS-Ⅱ
调度方法	时钟/事件驱动 优先级抢占	事件驱动 优先级抢占
相同优先级	支持	不支持
优先级个数	256(2个保留)	64(8个保留)
最大任务数	不受限制	56
任务间通信	事件、信号	信号量、消息、邮箱
最小代码	3.817 KB	3.914 KB
最小RAM	0.284 KB	0.792 KB

表2 WebitOS的任务调度参数(就绪队列中有10个任务)

Table 2 task scheduling parameters of WebitOS (10 tasks in ready queue of system)

任务调度参数	平均值/μs	标准差/μs	最大值/μs
任务切换时间	20.382	1.158	22.215
中断延迟时间	34.854	3.258	53.266
任务释放时间	148.708	6.314	220.043
任务启动时间	101.904	5.279	182.827

由表1知,WebitOS的调度机制采用的是静态优先级抢占式内核,支持256个优先级,相同优先级主动退让,其内核程序代码空间不到4 KB,占用数据空间不到300字节,其所支持的任务数、内核代码占用空间等指标优于相同测试环境中的μC/OS-Ⅱ实时内核,能较好地满足嵌入式环境下资源受限的要求.

由表2,对于运行在MCU频率只有8 MHz的WebitOS而言,其任务切换时间仅为20 μs左右,CPU主要完成了将现任务挂起,保存堆栈指针,查找就绪队列,将优先级最高的任务调入运行;最大中断响应时间约为50 μs左右,在这段时间里,CPU主要完成对临界资源进行保护,执行任务唤醒,进行任务切换;任务释放时间和启动时间与当前系统中运行的任务数有关.

表3 WebitOS的内存分配效率

Table 3 Memory allocation efficiency of WebitOS kernel

块大小/byte	平均值/μs	标准差/μs	最大值/μs
32	25.325	0.703	27.057
64	26.214	0.595	27.385
128	26.367	0.730	27.218
256	26.130	0.717	27.462
512	26.136	0.645	28.139
1K	25.213	0.623	27.236
2K	25.306	0.635	27.619
4K	25.333	0.670	27.163
8K	25.667	0.618	27.545
16K	27.018	0.703	28.320

由表3,WebitOS内核的内存分配时间对于不同大小的内存块基本相同;由标准差的大小,可以认为对于同等大小的内存块,WebitOS的内存分配具有确定性.

表4 IAS的部分网络指标

Table 4 Typical network indices of IAS

典型网络指标	平均值	标准差值	最大值
TCP建立时间/ms	3.09	0.23	3.37
WFTP上传速率/Kbps	46.71	1.62	48.75
HTTP下载速率/Kbps	58.75	2.84	62.00

由表4,因特网接入服务器文件上传速率较快,接近50 Kbps,TCP建立时间为3 ms左右,HTTP的下载速率也较高,这对于一般的非PC设备服务器要求已经足够了,能够满足嵌入式Internet下对设备进行访问和控制需求.

4 结 论

WebitOS是为非PC设备实现网络互联而开发的嵌入式实时操作系统.作为一个轻量级的RTOS,WebitOS能在8位微控制器平台上为实现各类嵌入式应用提供了底层系统级支持.其抢占式内核和高效的内存管理策略,保证了应用的确定性

和实时性.对 WebitOS 内核性能测试的结果表明,WebitOS 内核能较好地满足一般嵌入式应用开发需求,是一个精简、完善且高效的实时内核.

由于 WebitOS 内核的良好性能,目前已成功应用于传统产品的智能化改造过程中.应用案例有春兰静博士网络空调、台湾冠宇不间断电源、网络智能化循检器、母婴监护仪等.将来,还需进一步优化代码以提高系统的整体性能,研究操作系统如何完成对无线传输协议的支持以及如何有效地进行能耗管理等.

参考文献:

[1] 赵海.嵌入式 Internet —— 21 世纪的一场信息技术革命[M].北京:清华大学出版社, 2002. 28 - 91.
(Zhao H. *Embedded Internet —a revolution of information technique in 21st century*[M]. Beijing: Tsinghua University Press, 2002. 28 - 91.)

[2] Halang W A, Stoyenko A D. Next generation of real-time operating systems: industrial perspective[A]. *Proceedings of the NATO Advanced Study Institute on Real-Time Computing*[C]. New York, 1994. 595 - 596.

[3] Han G J, Zhao H, Wang J D, *et al*. Webit: a minimum and efficient Internet server for non-PC devices[A]. *Proceedings of the IEEE Global Telecommunications Conference (IEEE GLOBECOM '03)*[C]. San Francisco, 2003. 2928 - 2931.

[4] Scott A B. Performance analysis of dynamic soft real-time systems [A]. *IEEE International Performance, Computing, and Communications Conference (IPCCC 2001)*[C]. Arizona, 2001. 379 - 386.

[5] Riihijarvi J. Providing network connectivity for small appliances: a functionally minimized embedded web server [J]. *IEEE Communications Magazine*, 2001, 39(3): 74 - 79.

[6] Liu J W. *Real-time systems*[M]. New Jersey: Prentice Hall Incorporation, 2002. 85 - 113.

[7] 关沫,赵海.一个支持 EI 应用的嵌入式实时操作系统 WebitX[J].东北大学学报(自然科学版), 2004, 25(7): 649 - 652.
(Guan M, Zhao H. WebitX: a real-time operating system for embedded Internet applications[J]. *Journal of Northeastern University(Natural Science)*, 2004, 25(7): 649 - 652.)

[8] Jean J L. 嵌入式实时操作系统μC/OS-Ⅱ[M].第 2 版.邵贝贝,译.北京:北京航空航天大学出版社, 2003. 34 - 71.
(Jean J L. *Embedded real-time operating system μC/OS-Ⅱ*[M]. 2nd ed. Translated by Shao B B. Beijing: Beijing Aeronautics & Astronautics University Press, 2003. 34 - 71.)

[9] Garcey A, Lessey V. Design to time real-time scheduling [J]. *IEEE Transactions on Systems, Man and Cybernetics*, 1993, 23(6): 58 - 67.

[10] Atul A, Jon H, Marvin T, *et al*. Cooperative task management without manual stack management [A]. *Proceedings of the 2002 Usenix ATC*[C]. Monterey, 2002. 1123 - 1128.

Realization Mechanism and Performance of WebitOS Kernel

ZHANG Xi-yuan[1], *ZHAO Hai*[1], *SUN Pei-gang*[1,2], *LUO Ding-ding*[1]
(1. School of Information Science & Engineering, Northeastern University, Shenyang 110004, China; 2. Shenyang Artillery Academy, Shenyang 110162, China. Correspondent: ZHANG Xi-yuan, E-mail: zhangxy@neuera.com)

Abstract: Describes the architecture of WebitOS. As a hierarchically modularized system with perfect function, WebitOS consists of several modules for device driver, real-time kernel and TCP/IP protocol, etc. Analyzes the realization mechanism of WebitOS kernel: employs a priority-based preemptive multi-task scheduler for real-time scheduling and uses the best-fit allocation algorithm and collection-coalition algorithm for memory management. Then, tests the performance of WebitOS kernel in view of space-time costs and support to EI application. The results show that WebitOS kernel is a real-time, simplified and effective one that is specially adoptable to embedded applications in resource-constrained environment.

Key words: embedded operating system, hard real-time, embedded Internet, preemptive scheduling, best-fit allocation algorithm

(Received May 23, 2005)