无线传感器网络节点操作系统研究*

李 晶，王福豹，段渭军，王建刚
(西北工业大学 宽带网络技术研究所，陕西 西安 710072)

摘 要：无线传感器网络是一种全新的信息获取和处理技术，能够实时监测、感知和采集各种环境或监测对象的信息，而网络节点上的嵌入式操作系统是其大多数应用的基础。在综合分析大量无线传感器网络体系结构的技术文献和最新研究结果的基础上，提出了无线传感器网络嵌入式操作系统的设计目标，对通用的多任务操作系统 μC/OS-II 和事件驱动的操作系统 TinyOS 进行了对比分析，指出 TinyOS 在一些应用中的局限性及拓展。

关键词：无线传感器网络；嵌入式操作系统；TinyOS；μC/OS-II
中图法分类号：TP316 **文献标识码**：A **文章编号**：1001-3695(2006)08-0028-03

Research on Node Operation System of Wireless Sensor Networks

LI Jing, WANG Fu-bao, DUAN Wei-jun, WANG Jian-gang
(Institute of Broadband Network, Northwestern Polytechnical University, Xi'an Shanxi 710072, China)

Abstract: As a novel information acquirement and processing technology, Wireless Sensor Networks(WSN) can inspect, apperceive and collect the information of all kinds of environment and surveillance objects in a real-time way, and the embedded operation system which is operated in network nodes is the base of most of applications of WSN. After the analyses of the newest productions of WSN architecture, this paper brings forward the design targets of embedded operation systems of WSN, analyzes the general-purpose multi-tasking OS μC/OS-II and event-driven OS TinyOS, points out the limitations of TinyOS in some applications, and the improvement schemes.

Key words: Wireless Sensor Networks; Embedded Operation System; TinyOS; μC/OS-II

1 引言

微机电系统(Micro-Electro-Mechanism System, MEMS)、无线通信和数字电子技术的发展孕育了无线传感器网络(Wireless Sensor Network, WSN)[1]。WSN 是一种不需要固定网络支持，具有快速展开、抗毁性强等特点，可广泛应用于军事、工业、交通环保等领域，引起了人们广泛关注[1~4]。WSN 作为一个全新的研究领域，向科技工作者提出了大量的挑战性研究课题，微型化的嵌入式操作系统就是其中之一。

无线传感器网络是由大量集成有传感器和无线通信的网络节点组成。网络节点除了从外界环境采集数据外，还要接收邻近节点的数据，对数据进行处理、融合、转发。为了维护这个网络的拓扑结构，节点间需要定期交互更新路由信息。而网络节点的硬件能力是非常有限的，因此节点上的嵌入式操作系统必须满足在有限的物理空间内实现对硬件的高效管理[5]。

根据实现机制可以把现有的嵌入式操作分为两类，即 General-purpose Multi-tasking OS 通用的多任务操作系统和 Event-driven OS 事件驱动的操作系统，前者多用于便携式智能设备(如手机、PDA 等)和工业控制中。对于支撑几个独立的应用运行在一个虚拟机上的并行操作是高效的，在处理过程中任务的运行和挂起很好地支撑多任务或者多线程。但是，随着内部任务切换频率的增加将产生非常大的开销，典型代表如 μC/OS-II[6]、嵌入式 Linux、WinCE、Mantis[7]。而后者支持数据流的高效并发，并且考虑了系统的低功耗要求[8]，在功耗、运行开销等方面具有优势，因此备受关注。典型的代表如 TinyOS[5]，Contiki[9]。

由于无线传感器网络应用的多样性，节点的操作系统必须能够根据内存、处理器以及能量等满足应用的严格需求，也必须能够灵活地允许多种应用同时使用系统资源，如通信、计算和存储[8]。以下先就面向无线传感器网络节点的嵌入式操作系统设计目标进行讨论，据此对 μC/OS-II 和 TinyOS 两种典型操作系统进行对比分析。

收稿日期：2005-07-13；修返日期：2005-08-30
基金项目：国家自然科学基金资助项目(60472074)；西北工业大学研究生创业种子基金资助项目(Z200568)

2 无线传感器网络节点及其嵌入式操作系统设计目标

2.1 无线传感器网络节点

在不同应用中，传感器网络节点的组成不尽相同，但一般都由数据采集单元、数据处理单元、数据存储单元、数据传输单元、电源和嵌入式操作系统等部分组成[3]，如图 1 所示。被监测物理信号的形式决定了数据采集单元的类型；数据处理单元通常选用嵌入式 CPU，负责协调节点各部分的工作，如对数据采集单元获取的信息进行必要的处理、保存，控制数据采集单元和电源的工作模式等；数据传输单元主要由低功耗、短距离的无线通信模块组成[1]；电源为网络节点提供正常工作所必需的能源；嵌入式操作系统为网络节点提供必要的软件支持，负责管理节点的硬件资源，对不同应用的任务进行调度与管

理。在欧洲的 EYES Project 项目中[10]节点使用的处理器主要是 MSP430F149, 而无线通信模块则采用的是 RFM 公司的 TR1001。美国加州大学伯克利分校开发的 Motes[11]系列节点的处理器的发展是从 Atmel 公司的 AT90LS8535 到 Mega128L, 无线收发模块则是采用 RFM 公司的 TR1000 到 Chipcon 公司的 CC1000。

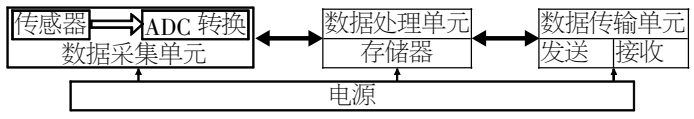

图 1 节点的基本构成

2.2 节点嵌入式操作系统设计目标

无线传感器网络具有能量有限、计算能力有限、传感器节点数量大、分布范围广、网络动态性强以及网络中的数据量大等特征[3], 这就决定了网络节点的操作系统设计应满足如下要求:

(1) 小代码量。由于节点的内存有限, 因此操作系统核心代码量必须比较小, 使其可以在有限的空间中具备高效管理硬件的能力[12]。

(2) 模块化。无线传感器网络设计的趋势是针对特定的应用而并不是普遍的应用, 不同的应用所需要的硬件平台是不相同的。随着无线传感器网络的广泛应用, 节点构成的变化是巨大的。在特定的硬件平台上, 根据不同的应用快速便利地结合软件模块实现应用是非常重要的[5]。

(3) 低功耗。WSN 的大多数节点采用电池供电。由于节点数量众多以及节点被散布的环境使更换节点的电池是不可行的, 甚至是不可能的, 因此低功耗的操作将延长整个网络的生命周期, 是操作系统设计必须满足的条件。

(4) 并发操作性。在传感器网络的节点上存在着大量的并发操作, 如数据采样、数据处理、数据转发可能同时进行。操作系统需要具备支持严格并发操作的能力[12]。

(5) 健壮性。WSN 节点数量众多以及运行环境特殊, 要求运行在单个节点上的操作系统不但健壮, 而且应该便利地适应于可靠的分布式应用的发展[8]。

3 μC/OS-Ⅱ与 TinyOS 的分析

μC/OS-II 操作系统是一种性能优良、源码公开且被广泛应用的免费嵌入式操作系统[6]。2002 年 7 月, μC/OS-II 在一个航空项目中得到了美国联邦航空管理局(Federal Aviation Administration) 对于商用飞机的、符合 RTCA DO-178B 标准的认证。它是一种结构小巧、具有可剥夺实时内核的实时操作系统, 内核提供任务调度与管理、时间管理、任务间同步与通信、内存管理和中断服务等功能[13], 具有可移植性、可裁减、可剥夺性、可确定性等特点[6]。

TinyOS(Tiny Micro Threading Operating System) 是一个开源的嵌入式操作系统, 它是由加州大学伯克利分校开发出来的, 主要应用于无线传感器网络方面[12]。目前在世界范围内, 有超过 500 个研究小组或者公司正在 Berkeley/Crossbow 的节点上使用 TinyOS[14]。它是基于一种组件(Component-based) 的架构方式[8], 能够快速实现各种应用。TinyOS 采用模块化设计, 程序核心往往很小, 能够突破传感器存储资源少的限制, 这能够让其很有效地运行在无线传感器网络上并去执行相应的管理工作等[15]。

以下是对传统的基于多任务的嵌入式操作系统 μC/OS-II 和事件驱动的嵌入式操作系统 TinyOS 从调度策略、堆栈分配、实时性、能量消耗以及并发操作等方面的对比分析。

3.1 调度策略

事件驱动的 TinyOS 采用两级调度[5]: 任务和硬件事件处理句柄(Hardware Event Handlers)。任务是一些可以被抢占的函数, 一旦被调度, 任务运行完成彼此之间不能相互抢占。硬件事件处理句柄被执行去响应硬件中断, 可以抢占任务的运行或者其他硬件事件处理句柄。TinyOS 的任务调度队列只是采用简单的 FIFO 算法[8]。任务事件的调度过程如图 2 所示。TinyOS 的任务队列如果为空, 则进入极低功耗的 Sleep 模式。当被事件触发后, 在 TinyOS 中发出信号的事件关联的所有任务被迅速处理。当这个事件和所有任务被处理完成, 未被使用的 CPU 循环被置于睡眠状态而不是积极寻找下一个活跃的事件[12]。

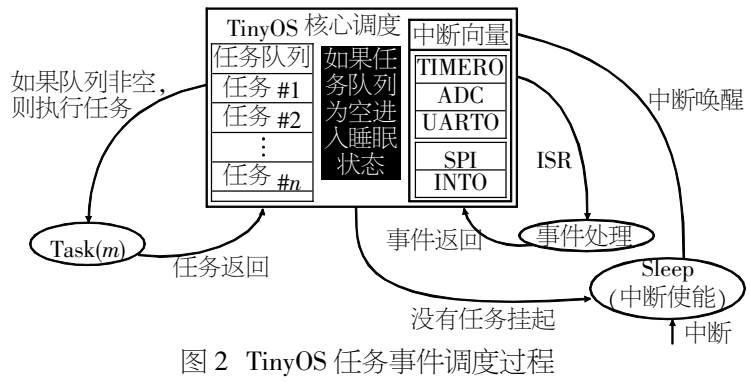

图 2 TinyOS 任务事件调度过程

基于多任务的 μC/OS-II 采用基于优先级的调度算法, CPU 总是让处于就绪态的、优先级最高的任务运行, 而且具有可剥夺型内核, 使得任务级的响应时间得以优化, 保证了良好的实时性[13]。其任务的切换状态如图 3 所示。在 μC/OS-II 中, CPU 要不停地查询就绪表中是否有就绪的高优先级任务, 如果有则做任务切换, 运行当前就绪的优先级最高的任务[6]; 否则运行优先级最低的空闲任务(Idle Task)。

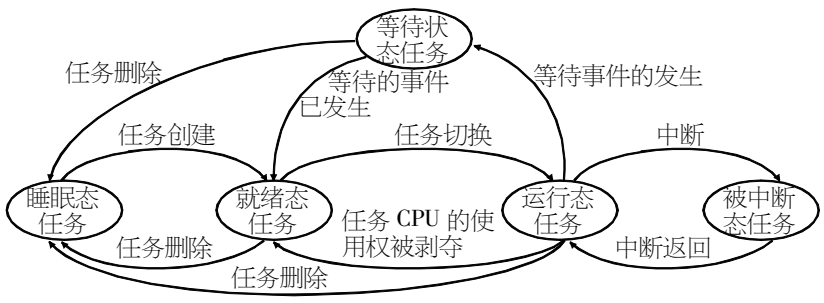

图 3 μC/OS-II 中的任务状态

从任务调度策略上, TinyOS 这种事件驱动的操作系统采用的两级调度策略、事件触发方式的机制让 CPU 在大多数时间处在极低功耗的睡眠模式。

3.2 运行空间

由于多任务的系统需要进行任务切换或者中断服务与任务间的切换。而每次切换就是保存正在运行任务的当前状态, 即 CPU 寄存器中的全部内容, 这些内容保存在运行任务的堆栈内。入栈工作完成后, 就把下一个将要运行的任务的当前状况从任务的堆栈中重新装入 CPU 的寄存器中, 并开始下一个任务的运行。因此, μC/OS-II 需要预先为每个任务分配堆栈空间[6]。在 μC/OS-II 中的任务堆栈空间可以根据任务的实际需求分配合适的大小。在事件驱动的 TinyOS 中, 由于对任务的特殊的语义定义运行—完成(Run to Completion)[15], 任务之

间是不能切换的,所以任务的堆栈是共享的,并且任务堆栈总是当前正在运行的任务在使用[8]。从运行空间方面看,多任务系统需要为每个上下文切换预先分配空间,而事件驱动的执行模块则可以运行在很小的空间中。因此多任务系统的上下文开销要明显高于事件系统,TinyOS 更好地减少了系统对 ROM 的需求量,满足了节点内存资源有限的限制。

3.3 实时性与低功耗操作

基于多任务的 μC/OS-II 总是运行进入就绪态任务中优先级最高的任务,使用的是可剥夺型内核。这样,最高优先级的任务何时可以执行,何时就可以得到 CPU 的使用权[6],实时性好。而在 TinyOS 中,采用的是简单的 FIFO 队列,不存在优先级的概念。

事件驱动的 TinyOS,如果任务队列为空,则进入睡眠态,直到有事件唤醒才去处理事件以及与事件相关的所有任务,然后再次进入睡眠态[8]。因而这种事件驱动的驱动系统,保证节点大多数时期都处在极低功耗的睡眠态,有效地节约了系统的能量消耗,延长了传感器网络的生命周期。而基于多任务的 μC/OS-II 实际上不考虑低功耗的应用。

3.4 其他方面

μC/OS-II 适合小型控制系统,最小内核可编译至 2KB[6]。一般来说 TinyOS 核心代码和数据大概在 400Bytes 左右[5]。μC/OS-II 源码绝大部分是用移植性强的 ANSI C 写的,与微处理器相关的部分是用汇编写的[13];而 TinyOS 代码则是由 NesC[12] 和 C 编写的,底层与硬件相关部分使用了大量的宏定义。因而 μC/OS-II 的可移植性要好于 TinyOS。μC/OS-II 与 TinyOS 的对比总结如表 1 所示。

表 1 TinyOS 与 μC/OS-II 的对比结果

操作系统	通用性	核心代码量	运行空间	能量消耗	并发操作	移植性	实时性
μC/OS-II	通用	大	大	不考虑	无	好	好
TinyOS	事件驱动系统	小	小	低	支持	差	差

4 TinyOS 的局限性与扩展

通过以上部分的比较,事件驱动的 TinyOS 相对于多任务的 μC/OS-II 能够更好地适用于无线传感器网络,但是它有一定的局限性。在 TinyOS 中对任务的操作依赖于对一个循环队列的操作,任务队列所能容纳的最大任务数为 7。而在传感器网络中需要处理的任务数量依赖于节点是发送原始数据给基站还是在发送到基站之前进行数据融合和处理。在前者节点将处理大量的路由任务,而在后者节点则需要为数据融合作更多的本地处理,当节点的任务量超过它的处理能力时,超负荷将会周期性地发生。在节点发送原始数据的情况下,节点的数据发送率非常高或者网络节点分布密集也将导致高的网络流量可能发生超负荷。而在节点发送的是融合过的数据后,本地的数据处理量非常大也将发生超负荷。节点在发生超负荷任务后,按照先后次序,后到的任务将被丢弃而不考虑任务的重要程度,也就是说 FIFO 调度算法对某些应用不适合。为了增强 TinyOS 的适用性,我们正在对 TinyOS 进行以下扩展,完善其调度部分及能量管理机制。

(1) TinyOS 目前的调度程序是由一个单一的 C 文件控制执行,可以考虑将调度程序模块化,这样可以根据不同的应用选择不同调度算法,如基于优先级、时间片轮转调度等。

(2) 通过在最底层增加一个事件队列减少事件的丢失,进而增强系统的健壮性。

(3) 增加全局的能量控制机制,使得微控制器可以实时监测当前电池的电压,进而控制控制器进入相应的操作模式。

5 结束语

本文阐述了适用于无线传感器网络的微型嵌入式操作系统的特征,并且对基于多任务的 μC/OS-II 与事件驱动的 TinyOS 针对无线传感器网络进行了分析对比,TinyOS 要比 μC/OS-II 更适合无线传感器网络。但 TinyOS 存在一些应用的局限性,需要研究者对其进行必要的扩展。

参考文献:

[1] 任丰原,黄海宁,林闯. 无线传感器网络[J]. 软件学报, 2003, 14(2): 1148-1157.

[2] 马祖长,孙怡宁, 梅涛. 无线传感器网络综述[J]. 通讯学报, 2004, 25(4): 114-124.

[3] 李建中,李金宝,石胜飞. 传感器网络及其数据管理的概念、问题与进展[J]. 软件学报, 2003, 14(10): 1717-1727.

[4] 孙雨耕,张静,孙永进,等. 无线自组传感器网络[J]. 传感技术学报, 2004, 17(2): 331-335, 348.

[5] Jason Hill, Robert Szewczyk, Alec Woo, *et al*. System Architecture Direction for Network Sensors[C]. Cambridge: International Conference on Architectural Support for Programming Languages and Operating Systems, 2000.

[6] Labrosse J J. μC/OS-Ⅱ ——源码公开的实时嵌入式操作系统[M]. 邵贝贝. 北京: 中国电力出版社, 2001.

[7] H Abrach, S Bhatti, J Carlson, *et al*. Mantis: System Support for Multimodel Networks of In-Situ Sensors[C]. Proc. of WSNA '03, 2003.

[8] Jason L Hill. System Architecture for Wireless Sensor Networks[D]. Berkeley University, 2003.

[9] Adam Dunkels, Bj rn Gr nvall, Thiemo Voigt. Contiki: A Lightweight and Flexible Operating System for Tiny Networked Sensors[C]. Tampa: Proceedings of the 1st IEEE Workshop on Embedded Networked Sensors, 2004.

[10] EYES Project[EB/OL]. http://eyes.eu.org, 2005-04.

[11] I F Akyildiz, W Su, Y Sankarasubramaniam, *et al*. A Survey on Sensor Networks[J]. IEEE Communications Magazine, 2002, 40(8): 102-114.

[12] David Gay, Phil Levis, Rob Von Behren, *et al*. The NesC Language: A Holistic Approach to Networked Embedded Systems[C]. Proceedings of Programming Language Design and Implementation(PLDI), 1996.

[13] 黄涛,徐宏喆,陈宁,等. 嵌入式实时操作系统移植技术的分析与应用[J]. 计算机应用, 2003, 23(9): 88-89, 98.

[14] Mission Statement[EB/OL]. http://www.tinyos.net, 2005-03.

[15] Philip Levis, Sam Madden, David Gay, *et al*. The Emergence of Networking Abstractions and Techniques in TinyOS[C]. Proceedings of the 1st USENIX/ACM Symposium on Networked Systems Design and Implementation, NSDI, 2004.

作者简介:

李晶(1980-),女,黑龙江人,硕士研究生,主要研究方向为传感器网络与分布式计算技术;王福豹(1963-),男,山西人,教授,博士,主要研究方向为计算机网络、流媒体、传感器网络等;段渭军(1962-),男,陕西人,高工,博士,主要研究方向为流媒体、无线通信等;王建刚(1974-),男,陕西人,硕士研究生,主要研究方向为传感器网络分布式计算与节点定位技术。