文章编号:1001 - 9081(2008)04 - 1029 - 03

无线传感器网络软件动态加载技术

黄廷辉,崔更申,赵岭忠
(桂林电子科技大学 计算机与控制学院,广西 桂林 541004)
(glhth@guet.edu.cn)

摘 要:以 Arena 操作系统的设计技术为基础,构建了一个适合无线传感器网络软件动态加载的总体设计方案,详细给出了动态对象加载器的接口设计技术。该设计减小了系统运行时的内核镜像,实现了系统在运行状态下动态加载和卸载软件模块的功能,同时保证系统长时间可靠运转并具备高可配置能力,从而有利于系统软件的升级和后期维护。模块动态加载实验验证了该设计方案的可行性。

关键词:无线传感器网络;动态加载;模块;Arena ;可执行可链接格式
中图分类号: TP316.89 **文献标志码**:A

Dynamic loading technique for software of wireless sensor network

HUANG Ting-hui, CUI Geng-shen, ZHAO Ling-zhong
(*School of Computer and Control, Guilin University of Electronic Technology, Guilin Gaungxi* 541004, *China*)

Abstract: Based on the design of Arena operating system, an overall framework for dynamically loading software of a wireless sensor network was proposed. The techniques for designing interface to the dynamic object loader were described in detail. This design reduced the runtime mirror of the core, and allowed dynamic loading and unloading of function modules of the system, and guaranteed long-time reliable system service and high configurability. Experiment results show the feasibility of the framework, and the ease of post updating and maintenance of system software.

Key words: wireless sensor network; dynamic loading; module; Arena; Executable and Linkable Format (ELF)

0 引言

随着计算技术的快速发展,大量分布式信息的采集和获取手段已不能很好满足用户的需求,而融合传感器技术、嵌入式计算技术、现代网络及无线通信技术、分布式信息处理技术的无线传感器网络(Wireless Sensor Network,WSN)则显现了它在大量分布式信息获取系统中的显著优势。随着无线网络传感器应用的普及,对无线传感器网络的研究与开发已经成为信息领域的一个热点。

软件动态更新技术是保证系统长时间可靠运转和具备高可配置能力的关键技术,在软件系统愈趋复杂的今天,其重要性得到越来越多的认同。随着无线传感器网络的深入发展,系统的可扩展性和可维护性已经成为重要的设计目标。传统的软件更新技术已经不再适合无线传感器网络的需要,迫切需要一种在系统运行状态下能最大限度地自由配置所需功能模块的机制。即系统在运行状态下可以动态地加载和卸载软件模块,以方便系统软件升级和后期维护,降低系统的营运成本。

1 无线传感器网络

无线传感器网络是由部署在监测区域内大量廉价微型的传感器节点组成,并通过无线通信方式形成的一个多跳的自组织的网络,其目的是相互协作感知、采集和处理网络覆盖区域中感知对象的信息,并发送给观察者。传感器节点通常是一个微型的嵌入式系统,它的处理能力、存储能力和通信能力相对较弱,只能通过有限能量的电池供电。

无线传感器网络应用的领域越来越多,节点上的系统软件必须能够根据有限的内存、计算速度以及能量来满足应用特定的需求,并允许多种应用灵活地使用各种系统资源,如通信、计算和存储。在无线传感器网络中,单个节点系统存在两个突出的特点:一个是系统的并发性操作很密集,需要系统软件有效地处理这些发生频繁、并发程度高、执行过程比较短的逻辑控制流程;另一个是系统结构的模块性高,在保证不影响系统开销的情况下,可以灵活地动态配置应用程序中的各个功能模块。

新的无线传感器网络软件系统不仅要考虑到无线网络传感器节点的硬件资源非常有限和节点能源储备低等约束,还要充分考虑无线传感器网络中的节点密集、数量较大,针对不同的应用领域,其硬件配置和任务需求具有多样性,以及由于系统功能的改进,用户可能赋予网络传感器系统新的功能需求。本文给出了一种新的软件动态加载机制,该设计可以精简系统软件,降低能源消耗,最大限度地支持本地和远程动态加载软件模块,实现适应环境和资源变化的节点自适应功能和根据用户需求动态配置系统的重构功能,方便进行远程维护和扩展新的功能,保证系统长时间地可靠运转。

2 传感器网路软件加载原理及现状

图 1 是一个传感器网络软件动态加载的概念模型[9]。图 1(a)是一个连接到传感器网络的模块服务器,它不受资源限制;图 1(b)是组成传感器网络的节点,在节点上的每一个操

收稿日期:2007 - 08 - 27;**修回日期**:2007 - 11 - 26。
基金项目:国家自然科学基金资助项目(60563005);广西壮族自治区教育厅立项项目(Z20667)。
作者简介:黄廷辉(1970 -),男,广西桂林人,副教授,主要研究方向:嵌入式系统设计、无线传感器网络操作系统设计; 崔更申(1970 -),男,广西桂林人,讲师,主要研究方向:嵌入式系统; 赵岭忠(1977 -),男,河南南阳人,讲师,博士,主要研究方向:形式化技术、符号算法。

作均需要受到有限的能源、网络带宽、CPU 计算速度和存储空间约束。

一个典型的软件模块动态加载过程包括以下步骤:1)生成一个可以动态加载的软件模块;2)系统管理者使用用户接口计划修补一个安装在传感器节点的软件模块或传感器节点系统请求一个新的功能模块;3)用户接口从服务器的模块库选出需要的软件模块;4)由数据传送协议把模块传送到目标传感器节点;5)当新的模块传送到目标节点,执行环境使用这个模块来更新传感器节点系统,系统启用新的服务。

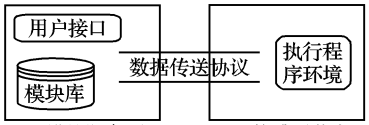

图1 软件模块动态加载模型

在无线传感器网络节点的硬件上,需要运行一个用来支持应用程序执行的执行环境软件,其设计是无线传感器网络实现软件模块动态加载的关键部分。目前采用的设计方式有三种:单体式设计、模块化设计和虚拟机设计。Berkeley 大学的 TinyOS 和 Colorado 大学的 MANTIS 属于单体式执行环境,不支持软件模块的动态加载,典型的升级处理必须重新构建一个新的镜像文件保存在辅助存储器(FLASH 或者 EEPROM)中,然后重新启动系统完成升级。UCLA 组设计的 SOS 和 Swedish 学院开发的 Contiki 属于模块化执行环境,可以实现应用软件模块的动态加载和灵活配置,软件更新相对简单,系统能耗低,但大部分系统核心功能模块不能动态更新。Berkeley 大学开发的 Mate 采用虚拟机执行环境,其更新将以新的应用脚本的形式出现,这些脚本非常小,实现网络开销也比较小。但由于虚拟机命令的解释执行比执行本地程序需要更多的处理时间,会导致系统更高的执行开销。三种方式都不能很好满足无线传感器网络动态加载软件模块的要求。

3 加载系统总体设计

3.1 纳内核设计

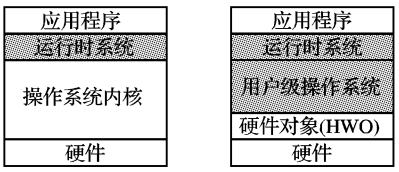

图2 单体式操作系统与 Arena 的比较

为满足传感器网络节点资源的限制和软件模块的灵活动态配置,设计的启动内核应该尽量小,大部分核心功能模块可由应用系统任意配置。Arena 是一个采用模块化结构设计的运行时用户级嵌入式操作系统。图2给出了 Arena 的层次结构。所有与硬件相关的代码放在一个称为“纳内核”的 HWO(HardWare Objective)实现。HWO 主要处理与底层硬件有关的操作,比如通常的存取寄存器内容和转换内存地址,主要是一个给上层用户级操作系统提供基本机器接口的运行时执行体。而其他与硬件无关的代码是由独立于硬件的用户级资源管理程序库中的可动态加载模块来实现。这样可以大大减小协系统启动内核镜像,提高系统运行的可靠性,很好满足系统运行时让用户任意配置和替换核心功能模块的需求。

3.2 动态加载模块设计

要实现软件模块的动态加载,必须设计一个能动态加载链接进应用系统的模块对象。在现代的类 Unix 操作系统中,ELF[12](Executable and Linkable Format)是目标文件和执行文件的一个标准的动态链接格式文件。一个 ELF 对象文件不仅包括要执行的程序代码和数据,同时也包括了符号表、所有外部未解释的符号名称和重定位表,非常适合于动态链接。

ELF 是由 GCC 编译器编译生成对象的默认格式,很多开源工具都支持对 ELF 文件的有效操作,例如调试器、链接器、文件转换器和计算程序代码和数据存储容量大小的工具。这些工具也被广泛地使用于不同的平台,包括 MS Windows,Linux,Solaris 和 FreeBSD。采用 ELF 文件作为加载对象不需要特定的开发环境和开发相应的工具,与其他动态加载系统的实现相比,这是一个很大的优点。本文设计的系统就采用 ELF 文件作为动态加载模块对象。

3.3 总体设计方案

Arena 是面向应用的嵌入式操作系统,它很好地将操作系统功能机制与策略进行分离实现,使得运行的系统镜像尽可能小。系统能灵活地根据环境应用的特殊需要,动态最优地配置系统管理策略。这些特点也很好地满足了无线传感器网络系统的要求。但传感器网络系统与传统嵌入式系统存在着一些不同,为此,本文以 Arena 操作系统设计技术为基础,给出了一个新的无线传感器网络操作系统的设计方案,它不仅能实现传感器网络系统软件动态加载的功能,而且能最大限度灵活地配置高效的应用系统。系统总体设计结构如图3所示。

从图3可以看到,在纳内核 HWO 中,装载协议(Application Loading Protocol,ALP)是一个轻型简单的传送协议,实现类似于简单文件传送协议(Trivial File Transport Protocol, TFTP)的功能,所不同的是 ALP 直接在网络 IP 层上实现,而 TFTP 建立在 UDP 层之上。ALP 主要是给动态对象加载器(Dynamic Object Loader,DOL)提供一个简单的发送数据和接收数据的接口,实现传感器网络模块服务器和传感器节点之间的模块传送。系统最初的主应用程序的加载是由 HWO 中的 App. Load 来完成。而以后传送到节点的软件模块被加载到应用程序和链接处理的工作是由在应用层的资源管理库中设计的 DOL[13] 来完成。

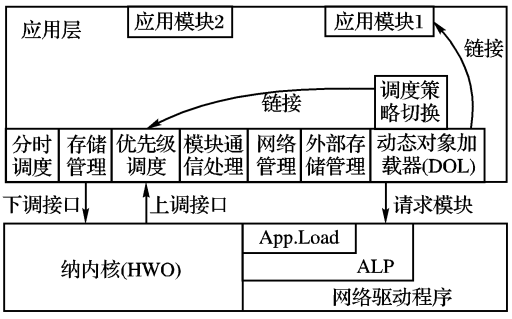

图3 可动态加载模块的系统结构

3.4 动态对象加载器接口设计

在 Arena 的应用层资源管理库设计了一个动态对象加载器,主要负责把软件模块动态加载到网络传感器节点并链接到应用系统,实现软件模块的动态加载。DOL 可以被应用程序直接调用,也可以通过一个资源管理层的管理模块间接调用它,如图3所示。

DOL 必须要记录哪些模块已加载到系统,以及主程序和

已加载的模块的相关信息。包括的信息有模块的名字,模块的符号表和字符串表的大小、位置,以及模块包括的所有节区信息。DOL 建立状态信息表的初始化操作通过以下系统接口函数来实现:

```
int dol_init ( char *name)
```

dol_init( )接口函数以主应用程序的名字作为参数,建立初始模块状态表。当系统需要一个应用层软件模块时通过调用如下接口函数来实现:

```
Int dol_load_module ( char *name , int type)
```

dol_load_module( )接口函数主要工作是加载特定的模块到应用程序的内存空间,在模块状态表中为新加载的模块建立一个对应表项,完成加载模块与其他模块之间的链接操作。一旦应用程序需要的模块在模块状态表中没有,就调用 dol_load_module( )接口,由 DOL 和远程模块服务器进行交互,服务器找到需要加载的模块通过网络传送给节点的 DOL。DOL 收到要加载的模块后,首先对 ELF 模块进行分析,得到模块的程序头部描述信息。系统调用存储管理器(Storage Manager, SM)分配相应的内存空间,依次加载所有可装入的节区,包括字符串和符号表等。然后 DOL 在模块状态表中为刚加载的模块设置相应的类型、标注符号表和字符串表的位置等初始化数据。

对于一个从主应用程序到某个加载模块的地址访问,DOL 提供一个接口函数来实现:

```
Void *dol_get_symbol ( char *name)
```

dol_get_symbol( )接口函数通过在模块状态表查找加载模块的符号表,返回一个指向需要符号的位置的指针,就可访问相应的内存单元。

卸载一个模块由以下接口函数实现:

```
Int dol_unload_moduler ( char *name);
```

这个接口函数以相应被加载模块的名字为参数,把原来在模块状态表中建立的信息注销掉。

4　模块动态加载实验

表 1　设计系统与其他操作系统的比较

系统	软件升级单位	动态加载应用模块	动态加载系统模块	系统运行内存空间
新设计系统	单个模块	可以	可以	少
SOS	单个模块	可以	不可以	少
Tiny OS	整个系统	不可以	不可以	多

为了验证软件动态加载机制的可行性,设计了如下实验系统。传感器节点系统配置为:S3C44B0 ARM7 微处理器,2 MB的 ROM,4 MB 的 RAM 和 10M 的以太网芯片。运行应用服务器和模块服务器的PC机配置如下:英特尔赛扬800MHz的处理器、256 MB RAM 和 10M 以太网卡,操作系统为 Redhat 7.0 Linux。服务器与节点系统之间通过有线电缆连接。在实验平台上成功进行了动态替换任务调度策略模块和动态加载 TCP 软件模块的两个实验。表 1 给出了新设计的软件系统和文献[2]介绍的操作系统的比较。

实验表明,新设计的系统不仅可以动态加载应用程序模块,还可以动态加载系统的核心功能模块。和静态加载系统相比,较大提高了系统可维护和可扩展性能,当系统没有加载 TCP 动态链接模块时,可节省 15 KB 的内存空间。但和静态加载系统相比,系统执行时需要一定的动态链接和重定位模块时间,当模块第一次需要时,还增加了模块的网络传输时间,这些会增加系统执行的时间开销。而一旦模块加载到系统,则执行效率与静态加载系统相当。

5　结语

本文给出的软件动态加载机制实现了在无线传感器网络系统中的模块动态加载与更新,使得在无线传感器网络软件开发过程中,开发人员可以进行更为有效的系统设计,共享资源,提高效率,使产品快速市场化。整个系统扩展性和可维护性得到较大的提高,较好地满足了实际需要。实验是在嵌入式环境中完成的,设计中对系统安全性、模块传输性能的考虑还不够,在设计加载模块中还可以进行优化。

参考文献:

[1] 李健中,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.

[2] HAN C C, KUMAR R, SHEA R, *et al*. Sensor network software update management: a survey[J]. International Journal of Network Management, 2005, 15(4): 283 - 294.

[3] MAYES K R, BRIDGLAND J. Arena-A run-time operating system for parallel applications[C]// Proceeding of the 5th Euromicro Workshop on Parallel and Distributed Processing. London: IEEE Computer Society Press, 1997: 253 - 258.

[4] Tool Interface Standards(TIS) Committee. Executable and Linking Format(ELF) specification version 1.2[EB/OL]. [2007 - 01 - 26]. http://x86.ddj.com/ftp/manuals/tools/elf.pdf

[5] BEYER S, MAYES K, WARBOYS B. Dynamic loading in an application specific embedded operating system[EB/OL]. [2006 - 05 - 20]. http:// www.asap.ecs.soton.ac.uk/steps/papers/1230_1245.pdf.

[6] GAUGER M, LACHENMANN A, MINDER D, *et al*. FlexCup: A flexible and efficient code update mechanism for sensor networks[C]// Proceedings of the 3rd European Workshop on Wireless Sensor Networks (EWSN 2006). Zurich, Switzerland: IEEE Computer Society Press, 2006: 212 - 227.

[7] 杨伟,罗雷. 嵌入式系统中的模块动态加载技术[J]. 单片机与嵌入式系统应用,2005(11):8 - 10.

(上接第 1028 页)

[6] COTA E, ZEFERINO C, KREUTZ M, *et al*. The impact of NoC reuse on the testing of core-based systems[C]// Proceedings of the 21st IEEE VLSI Test Symposium. Washington DC: IEEE Computer Society, 2003: 128 - 133.

[7] COTA E, LIU C. Constraint - driven test scheduling for NoC - based systems[J]. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 2006, 25(11): 2465 - 2478.

[8] COTA E, CARRO L, WAGNER F, *et al*. Power-aware NoC reuse on the testing of core-based systems[C]// Proceedings of International Test Conference. Washington DC: IEEE Computer Society, 2003: 612 - 621.

[9] ZEFERINO C, SUSIN A. SoCIN: A parametric and scalable network-on chip[C]// Proceedings of the 16th Symposium on Integrated Circuits and Systems Design. Washington DC: IEEE Computer Society, 2003: 169 - 174

[10] MARINISSEN E J, IYENGAR V, CHAKRABARTY K. A set of benchmarks for modular testing of SoCs[C]// International Test Conference. Washington DC: IEEE Computer Society, 2002: 521 - 528.