无线传感器网络操作系统TinyOS的移植

程 龙 杨 波
(北京航空航天大学自动化科学与电气工程学院 北京 100191)

摘 要 TinyOS系统以其组件结构模型、事件驱动、简易的编程环境等优点成为目前最受关注的无线传感器网络操作系统,而CC2430以其高性能、低成本、低功耗被广泛应用于无线传感器网络节点,但TinyOS不支持CC2430芯片,因此在分析TinyOS基本原理、NesC编程语言实现机制及其编译过程的基础上,介绍了基于CC2430的TinyOS及NesC语言的移植方法,完成了NesC语言、CC2430的Timer组件、Uart组件、ADC组件和RF组件的移植,并测试了各组件的移植效果。在移植的基础上,进行了基于优先级的任务调度机制的改进和星形网的组建。实验测试结果表明,两项扩展应用均已开发成功。

关键词 无线传感器网络,TinyOS,CC2430,星形网
中图法分类号 TP316 **文献标识码** A

Transplantation of Embedded Operating System Tinyos for Wireless Sensor Network Nodes

CHENG Long YANG Bo
(School of Automation Science and Electrical Engineering, Beihang University, Beijing 100191, China)

Abstract The TinyOS system has received much attention as a wireless sensor network operation system for its syntactic model, event-driven architecture and easy programming environment. And the CC2430 chip is widely used for wireless sensor network nodes with high performance, low price and low power cost. But the CC2430 chip is not supported by TinyOS system. After an analysis of TinyOS principle, the programming language NesC and its compiling process, the transplantation was performed including porting NesC and CC2430's components like Timer, Uart, ADC etc, and its application was tested. The test result shows that the TinyOS system can run on the CC2430 chip, the components work well, and the transplantation is successful. Based on the transplantation, two extended applications were proposed and realized including an improved priority-based scheduling mechanism and the formation of star network.

Keywords Wireless sensor network, TinyOS, CC2430, Star network

无线传感器网络集传感器、嵌入式系统、无线通信和分布式信息处理技术于一体,广泛应用于军事、医疗、工业、农业及环境监测等领域,具有广阔的发展前景[1-3]。

无线传感器网络主要由无线传感器网络节点和无线传感器网络操作系统组成。随着技术发展,网络规模越来越大,节点处理的数据越来越多,任务越来越重,使得程序的编写和调试变得十分繁琐,这对无线传感器网络节点使用的芯片和无线传感器网络操作系统提出了更高的要求。目前,加州伯克利分校设计的无线传感器网络操作系统TinyOS在现阶段优势较为明显,但TinyOS系统不支持功能强大的无线传感器网络节点芯片CC2430,为此需要进行基于CC2430的TinyOS系统的移植,以促进国内基于CC2430的TinyOS的应用与研究。

本文详细介绍了基于CC2430的TinyOS系统的移植过程,编写相应组件进行了测试,开发了两项扩展应用,为进行基于CC2430的TinyOS系统的移植、应用与研究提供了借鉴奠定了基础。

1 TinyOS的简介

TinyOS系统是针对无线传感器网络的新型操作系统,它使用专门开发的NesC语言进行开发,采用模块化设计,具有组件构成、事件触发、并发模型、主动消息等特点。它运行在无线传感器网络节点上,是上层应用和协议运行的前提。TinyOS提供了一系列可重用的组件,这些组件为外界应用提供接口,同时把接口的实现封装到组件的内部,用户使用组件可以只知道接口,而无需了解其内部的实现。

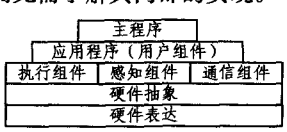

图1 TinyOS基于组件的程序模型

TinyOS主要由主程序、应用程序、硬件抽象层和硬件表达层构成,体系结构如图1所示。主程序是一个简单的先入先出任务调度器。应用程序结合具体的应用实例进行编写。

本文受航空科学基金(2008ZD51056)资助。

程 龙(1987—),男,硕士,主要研究方向为无线传感器网络,E-mail:egyptshizhe@126.com;杨 波(1972—),女,博士,副教授,主要研究方向为无线传感器网络的关键技术研究、新型微特电机的智能检测与智能控制等。

硬件抽象组件主要由执行组件、感知组件和通信组件组成，为应用层提供与硬件无关的各种功能。硬件表达层组件将硬件提供的各种功能以函数形式表达，供上层调用。层与层之间互访由接口文件完成[4]。

2 NesC 语言与移植

2.1 NesC 简介

NesC 语言是针对 TinyOS 程序开发而设计的一种 C 语言的扩展，在提供了程序开发人员熟悉的 C 语言作为编程语言的同时支持了 TinyOS 的模块化思想，方便了 TinyOS 应用程序和系统组件的开发。

用 NesC 语言编写的应用程序由一系列组件及接口构成，一个组件一般提供一些接口，接口可以看作是这个组件实现的一系列函数的声明。一个组件通过引用其他组件的相同类型的接口声明，就可以调用其他组件的对应接口的函数，从而实现组件间的功能相互调用。

NesC 语言编写的程序中有两种组件：模块组件和配置组件。配置组件用于将别的组件绑定在一起，并通过接口实现组件间的组合与互联。模块组件提供具体应用程序代码，以实现一个或多个接口的功能。接口则提供了两类函数，分别为接口的提供者需要实现的一组命令函数，和接口的使用者需要实现的一组事件函数[5]。

一个 TinyOS 程序就是由 NesC 语言编写的多个模块组件和配置组件组成的，其中各个组件间通过相同类型的接口相互连接。当用户调用某一接口的函数时，TinyOS 将通过配置组件的连接关系寻找实现该函数的模块组件，并最终调用该模块组件所提供的对应函数来实现用户要求的功能。

2.2 NesC 的移植

TinyOS 开发环境为 Unix，若要为 TinyOS 开发应用程序，首先要使用 NesC 进行编程，产生以“. nc”为扩展名的源文件；然后再调用 Unix 的 NCC(NesC Compiler)编译器将源文件编译成硬件可以执行的二进制/十六进制机器码[6,7]，如图 2 所示。

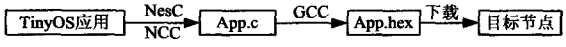

图 2 TinyOS 应用程序编译过程

但 NesC 的编译器 NCC 实际调用的是 Unix 的 GCC(GNU Compiler Collection)编译器，而 GCC 编译器并不支持 CC2430 所使用的 8051 处理器。为解决 NesC 程序的编译问题，需要在编译过程中使用 Perl 语言的正则表达能力，转换 C 语言编译器不能识别的 NesC 关键字，将 NesC 语言编写的“*. nc”文件先编译为常规的 C 语言源文件，然后绕过 Unix 的 GCC 编译器，改由支持 8051 的 Keil 进行编译，其过程如图 3 所示。

图 3 8051 平台 TinyOS 应用程序编译过程

具体的实现为编写脚本文件 mangleAppC. pl，在编译规则文件“. rules”中添加该 Perl 脚本的引用，以生成中间文件 App. preMangle. c，并通过建立批处理文件 CC2430-F128. bat，调用 Keil 程序编译生成目标文件 App. hex。

3 TinyOS 的移植

3.1 TinyOS 最小平台搭建

TinyOS 应用程序的编译是以平台为对象的，由于 TinyOS 不支持 CC2430 及其相关平台，因此编写针对 CC2430 的 TinyOS 应用程序必须先建立能被 TinyOS 承认的平台。通过分析 TinyOS 可知，一个新平台建立的首要步骤为建立一个不实现任何功能、只包含被 TinyOS 承认为合法平台所需要的目录和相关文件的最小平台。搭建 TinyOS 最小平台是整个移植的重点，根据 TinyOS 最小平台的定义，需要创建目录及编写的文件为：①/tos/platforms/cc2430 目录，及其下的“. platform”文件、platform. h、PlatformC. nc 和 PlatformP. nc 文件，这些文件包含平台初始化的实现代码和平台的环境变量；②/tos/support/make 下的 CC2430em. target 文件，包含移植代码的目标平台的识别信息；③/tos/support/ make/mcs51 下的“. rules”文件，包含生成目标平台二进制/十六进制可执行代码时的编译说明；④/tos/chips/ mcs51 下的 hardware. h 和 McuSleepC. nc 文件，包含平台硬件体系必备的宏定义和低功耗机制实现代码[8]。

3.2 CC2430 组件的编写

TinyOS 采用组件化思想，用户编写的应用程序由一系列组件连接而成，其中包括用户自己编写的用于实现应用程序功能的组件、系统提供的用于实现常用功能的组件和针对不同芯片的硬件表达和抽象组件。由于 TinyOS 本身并不含有针对 CC2430 的硬件表达和抽象组件，因此要编写相应组件来完成移植，由 TinyOS 的组件化思想可得出组件移植方案为：①找到 TinyOS 提供的最上层系统组件，了解移植所需要提供的接口与函数。②根据需要移植的具体硬件和需要实现的接口和函数，编写最底层的硬件描述组件，并提供供上层组件使用的接口。③编写上层组件和配置组件，将最底层组件和系统组件按需要的方式连接起来。

按照这样的步骤编写以下几个重要的组件：

1)通用 IO 口组件。通过建立 HplCC2430GeneralIOC 模块组件实现。提供两类系统接口 GeneralIO 和 Init。其中 GeneralIO 用于对 IO 口的设置和使用，Init 用于对 IO 口进行初始化。

2)Timer 组件。Timer 组件由 TimerMilli 组件、HilTimerMilliC 配置组件、HplCC2430 Timer1AlarmCounter 组件等组成，用于产生以毫秒为单位的计时器及实现计时等相关功能。

3)Uart 组件。Uart 组件由 StdOut 组件、UartC 配置组件、HalCC2430SimpleUart 组件等组成，用于实现通过串口接收字符及输出字符串的功能。

4)ADC 组件。ADC 组件由 AdcC 配置组件，AdcP 模块组件组成，用于实现 ADC 单次或连续采样。

5)RF 组件的支持，RF 组件主要由 SimpleMacC 配置组件、SimpleMacM 模块组件、HalCC2430Radio 组件和 HplCC2430InterruptsC 模块组件构成，用于实现基本无线数据传输功能。

3.3 移植结果测试

本文使用 SmartRF07DK 开发平台及 HFZ-CC2430EM-22 模块进行测试，测试方法为编写测试程序 TestAll，调用 Timer 组件、Uart 组件、ADC 组件和 RF 组件，并编写相应测

试代码，然后编译并烧录到开发平台的 CC2430 上运行，进行测试并分析。

经测试，能用 Timer 组件产生以 1ms 为周期的时钟，并控制指定 IO 口产生周期为 1ms 的矩形波，并能用示波器进行观察，如图 4 所示，表明 Timer 组件能正常使用。

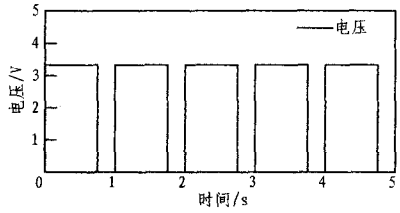

图 4　Timer 组件测试结果

测试表明能通过串口调试助手输入不同的字符来实现指定的功能并输出指定字符串，表明 Uart 组件能正常使用；能使用 ADC 组件对电位器电压进行单次采样和连续采样，并且与示波器测量电压相符，表明 ADC 组件可以正常使用；最后通过 RF 组件进行两个节点间的无线数据传输，并能通过串口调试助手输出指定数据，表明 RF 组件可以正常使用，

至此表明基于 CC2430 的 TinyOS 系统移植成功。

4　TinyOS 的扩展应用

4.1　TinyOS 任务调度机制的研究及改进

TinyOS 采用先入先出的调度模式确保了 TinyOS 的并发处理能力以及一定的实时响应能力。可是这种简单的调度策略在一些复杂情况下，如网络规模较大、节点数量较多时表现并不理想。为此需要针对具体的情况对 TinyOS 的任务调度策略做出一定的修改以满足应用的需求[9]。

为了克服 TinyOS 任务调度机制的不足，提高重要任务的执行能力，在先入先出队列的基础上加入了优先级机制。改进后的任务调度机制将给每个任务分配一个优先级，默认为 0，优先级参数越大，则优先级越高。

编写带优先级参数的新的任务接口 TaskPriority，并编写任务调度机制组件 SchedulerPriorityP，使其调用 TaskPriority 接口并实现基于优先级的任务队列排序，以完成高优先级先执行的要求。最后编写 TinySchedulerC 配置组件调用 SchedulerPriorityP 组件，完成基于优先级调度机制的改进。

为进行测试，设计 3 个任务，任务 1、任务 2、任务 3 分别每 30ms、50ms、70ms 执行一次，当有两个任务或三个任务同时执行时，只有队首的任务得到执行。并且分别设任务 1/2/3 的优先级为 8/4/1，进行 100s 的测试，实际测试结果如表 1 所列。

表 1　任务调度结果比较

任务调度机制	先入先出	基于优先级
任务 1	2285	3333
任务 2	1714	1333
任务 3	1428	762
总执行次数	5427	5428

从测试结果可以看出，在采用优先级的任务调度机制下，总执行次数基本保持不变，表明没有影响任务的正常执行。而在基于优先级的情况下，任务 1 的执行次数增幅 46%，并且与理论上 100s 中任务 1 的执行次数一致，同时在任务 2 与任务 3 的竞争中，基于优先级的情况下任务 2 与任务 3 的执行次数之比从 1.2 提升为 1.75。综上，基于优先级的任务调度机制改进成功，可以对不同的任务设置优先级，以决定竞争时任务的执行次序。

4.2　星形网的组建

对于无线传感器网络操作系统而言，能否方便的组网是判别其实用性的一个重要因素。本文通过使用 TinyOS 组建简单的星形网络，来研究 TinyOS 组网方面的扩展性和实用性。

编写协调器模块组件 SimpleMacCorM 和接口组件 SimpleMacCor，以及节点模块组件 SimpleMacMoteM 和接口组件 SimpleMacMote，以实现如下功能：(1)协调器对全体节点广播帧；(2)节点接收到协调器发送的数据帧后发送带有自身 ID 的数据帧；(3)协调器根据节点发送的数据帧获得节点 ID 并完成组网；(4)协调器对指定节点发送数据帧；(5)节点只接受协调器发送的数据帧。

通过 1 个协调器和 3 个节点进行实验，3 个节点接收到有效帧后分别向协调器发送 0/1/2，协调器接收后通过串口调试助手输出。

可以观察到，协调器发送广播帧后，各节点均接收到并返回含自身 ID 的数据帧，协调器从中得到各节点 ID 并输出“00 02 01”；协调器针对指定节点发送数据帧，则只有该节点接收并返回数据帧，协调器依次输出 0、2、1 三个值。说明星形网组网成功，协调器可与各节点通信，而各节点不能互相通信，只能与协调器通信。

结束语　本文在分析 TinyOS 系统及其编程语言 NesC 的基础上，进行了基于 CC2430 的 TinyOS 系统的移植，介绍了 NesC 语言的移植、TinyOS 最小平台搭建的过程，编写了 CC2430 的通用 IO 口、Timer、Uart、ADC 及 RF 等组件，并进行了相关测试。测试结果表明各组件可以正常使用，证明移植成功。在移植的基础上，进行了基于优先级的任务调度机制的改进和星形网的组建，并通过编写测试程序，测试并验证了扩展应用的成功开发。至此，本文基本上完成了基于 CC2430 的 TinyOS 的移植，并在此基础上改进了任务调度机制、支持了星形网的组建，为 CC2430 无线传感器网络提供了无线传感器网络操作系统 TinyOS，有助于降低未来以 CC2430 芯片为主体的无线传感器网络的应用程序的开发难度，对于缩短开发周期，减少开发成本，具有很重要的意义。

参 考 文 献

[1]　崔戴. WSN 的构成和应用[J]. 中国电子商情，2009(5)：41-44

[2]　崔学义. WSN 的由来以及在国外的研究和发展[J]. 中国电子商情，2009(5)：45-48

[3]　王峰，王健，郭忠文. 基于 uC/OS-II 的嵌入式无线传感器网络平台的设计[J]. 计算机科学，2010，37(6)：136-141

[4]　Berkeley U C. TinyOS: an open-source operating system designed for wireless embeddedsensor networks[EB/OL]. http://www. tinyos. net，2007-07-30

[5]　刘信新，邵明凯. 无线传感器网络操作系统 TinyOS 研究[J]. 计算机与数字工程，2007，35(7)：65-68

[6]　胥小武，田丰. 基于 CC2430 的 TinyOS 系统移植开发[J]. 沈阳航空工业学院学报，2009，26(5)：83-86

[7]　周迪，陶正苏，陈德富，等. CC2430 节点移植 TinyOS[J]. 机电产品开发与创新，2009，22(6)：97-98

[8]　Leopold M. Creating a New Platform for TinyOS 2. x[EB/OL]. http://www. tinyos. net/tinyos-2. 1. 0/doc/html/tep131. html，2007-09-06

[9]　赵建华，刘宁. TinyOS 调度机制的研究与改进[J]. 西华大学学报，2008，27(2)：8-10