

基于 WSNs 平台的 Contiki 通用移植方法研究

冀宇鑫, 杨冬, 秦雅娟, 郑涛, 武尚青

(北京交通大学电子信息工程学院, 北京 100044)

摘要:无线传感器网络(WSNs)目前已成为国内外备受关注的研究热点,作为一种新的信息获取和处理模式,WSNs已广泛应用于军事、医疗、环境监测等各个领域。而 WSNs 节点的存储能力和功耗成为目前设计人员面临的巨大挑战。文中以此为出发点,研究比较了目前广泛采用的几种无线传感器网络操作系统(WSNsOS)的主要特点,选取小内存、低功耗,且易于移植的 Contiki 操作系统,对 Contiki 的系统架构及代码架构进行了深入分析;研究了基于 WSNs 平台移植 Contiki 的通用方法,并以微型传感路由器(MSR)作为测试平台,进行了 Contiki 移植的仿真测试,验证方法的可行性,一定程度上解决了文章提出的问题。

关键词:无线传感器网络;Contiki 操作系统;移植

中图分类号:TP31

文献标识码:A

文章编号:1673-629X(2012)11-0134-04

Research on General Porting Method of Contiki OS Based on Wireless Sensor Networks

Ji Yu-xin, YANG Dong, QIN Ya-juan, ZHENG Tao, WU Shang-qing

(School of Electronic Information Engineering, Beijing Jiaotong University, Beijing 100044, China)

Abstract: Wireless sensor networks (WSNs) has become a research focus at home and abroad as a new information acquisition and processing mode. WSNs has been widely used in various fields, such as the military, medical, environmental monitoring and so on. However, in wireless sensor networks, the memory and power-consumption are the biggest challenge for designers. Using a small and low-power operating system in the WSNs nodes can resolve this problem. As a starting point, in the paper, firstly choose Contiki as OS by studying various WSNsOSs. Then, do research on a general porting of Contiki for WSNs by a further studying on the system architecture and code structure. In order to verify the reliability, take MSR as test-platform and have a porting of the Contiki for MSR. It is ultimately verified through simulation testing and to some extent, solve the problems proposed previously.

Key words: wireless sensor networks; ContikiOS; porting

0 引言

目前,无线传感器网络^[1](WSNs)节点采用的操作系统有多种,包括 $\mu C/OS-II$, TinyOS, Contiki 等,这些操作系统可以满足 WSNs 的基本要求。为了解决内存、能量有限的 WSNs 应用需求,文中研究分析了各种 WSNsOS 的特点,选取了专为内存受限网络系统设计,开源、易移植的轻量级 Contiki 操作系统。通过对 Contiki-2.5 代码架构及系统架构的分析研究,设计了基于不同平台移植 Contiki 的通用方法,仿真测试验证了该方法的可行性。利用文中研究的 WSNs 上 Contiki

的通用移植方法,结合相应平台的特点,可以将 Contiki 移植到相应的 WSNs 平台上进行应用开发。

1 无线传感器网络平台及其操作系统简介

1.1 无线传感器网络平台简介

WSNs 节点硬件平台一般由数据采集单元、数据处理和控制单元、数据传输单元、电源等部分组成^[2],见图1。传感器采集环境中的相关信息数据,经过 ADC 转换为数字信息送往处理单元。处理和传输单元包括处理器、存储器和控制模块,实现数据的分析、处理和存储等功能,并控制整个传感器节点的运行。传输单元负责建立无线信道,实现节点间的短距离无线通信。电源为系统各单元提供运行所需的能量^[3]。

由图1可以看出,数据处理单元起着非常重要的作用。通常一个节点拥有 2~10KB 的 RAM, 4~16KB 的 EEPROM, 48~128KB 的 Flash 内存。可见,其存储空间非常有限。而且 WSNs 中通常是大范围部署节点

收稿日期:2012-03-07;修回日期:2012-06-12

基金项目:“新一代宽带无线移动通信网”科技重大专项(2012ZX03005003)

作者简介:冀宇鑫(1988-),女,硕士研究生,研究方向为无线传感器网络;秦雅娟,博士,教授,博士生导师,研究方向为下一代网络路由交换技术、移动互联网技术、宽带无线通信。

来长时间地监测“环境变量”,所以必须考虑其功耗等成本因素。

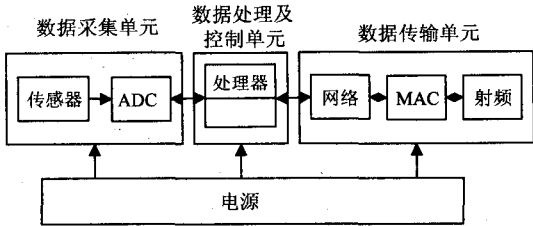


图 1 WSNs 节点硬件平台结构图

1.2 无线传感器网络操作系统简介

WSNsOS 是 WSNs 系统的基本软件环境,是其应用软件开发的基础。WSNs 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^[4]。根据实现机制可以把现有的嵌入式操作系统分为两类,即通用的多任务操作系统和事件驱动型操作系统。通用多任务操作系统对于支撑几个独立应用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。多任务操作系统的典型代表是 $\mu\text{C}/\text{OS-II}$ 。而事件驱动型操作系统支持数据流的高效并发,并且考虑了系统的低功耗要求,在功耗、运行开销等方面具有优势,典型代表如 TinyOS, Contiki^[5]。

由于 WSNs 应用的多样性,节点的操作系统必须在内存、处理器以及能量等方面严格符合应用的需求,也必须能够灵活地允许多种应用同时使用系统资源。表 1 对 $\mu\text{C}/\text{OS-II}$, TinyOS 和 Contiki 三种典型操作系统进行了比较。

表 1 $\mu\text{C}/\text{OS-II}$, TinyOS 和 Contiki 比较结果

| 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 |
|----------------------------|-------|-------|------|------|------|-----|-----|
| $\mu\text{C}/\text{OS-II}$ | 通用 | 大 | 大 | 不考虑 | 无 | 好 | 好 |
| TinyOS | 事件驱动型 | 小 | 小 | 低 | 支持 | 差 | 差 |
| Contiki | 事件驱动型 | 小 | 小 | 低 | 支持 | 好 | 好 |

基于以上部分对 WSNs 及 WSNsOS 的研究,文中最终选取 Contiki 作为 WSNs 节点操作系统。Contiki 只需要几 KB 的代码和几百字节的内存就能提供多任务的环境及 TCP/IP 支持。其一般的配置是 2KB 的 RAM 和 40KB 的 ROM,由一个事件驱动型的内核组成,在它上面可以动态地加载或者卸载应用程序。Contiki 进程使用一个轻量级的 protothreads,在事件驱动型内核之上提供了一个线性的、类线程的编程方式。Contiki 的这些轻便的特性使得其很适合做内存有限的微控制器。另外,Contiki 集成了很多独立的模块,例如, uIP 和 Rime 协议栈, uIP 可以理解为裁减后的 TCP/IP 协议栈,每层都考虑了低功耗的处理^[6]。可

见, Contiki 可以很好地满足 WSNs 的特点。此外, Contiki 是一个高度可移植的操作系统,适用于多种平台之上。文中通过对 Contiki 代码架构的研究分析,旨在进一步研究 Contiki 通用的移植方法,将 Contiki 移植到多种 WSNs 平台上来满足其需求。

2 Contiki 移植技术的研究

2.1 Contiki 系统架构及源代码架构分析

Contiki 操作系统遵循模块化架构,在内核中遵循事件驱动型模型,对每个单独的进程都提供可选的线程设施。每个应用程序都可以调用服务程序。Contiki 操作系统的系统架构如图 2 所示^[7]。

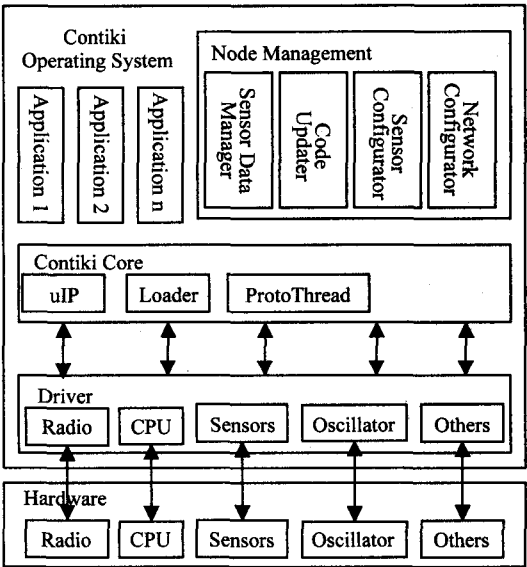


图 2 Contiki 系统架构图

文中以 Contiki-2.5 版本为例,对其源代码架构进行了研究分析。Contiki-2.5 主要有 apps、core、doc、examples、platform、tools 等目录,各部分目录分析见表 2。

表 2 Contiki-2.5 源代码架构

| 目录名称 | 内容分析 |
|----------|--|
| core | Contiki 的核心源代码,包括 net(网络)、文件系统(cfs)、外部设备(dev)、链接库(lib)等,并且包含了时钟、I/O、elf 装载器、网络驱动等的抽象 |
| cpu | Contiki 支持的微处理器,例如 arm、avr、msp430 等。如果需要支持新的微处理器,可以在这里添加相应的源代码 |
| platform | Contiki 的硬件平台,例如 mx231cc、micaz、sky、z1 等。Contiki 的移植平台主要在这个目录下完成,该部分代码与相应的硬件平台相关 |
| apps | Contiki 所添加的一些应用程序,如 ftp、shell、webserver 等。在项目程序开发中可以直接使用,只需将 Makefile 中定义 APPS=[应用程序名称]即可 |
| examples | Contiki 针对不同平台的示例程序 |
| doc | Contiki 的帮助文档目录,对 Contiki 应用程序开发极具参考价值,使用前需先使用 Doxygen 进行编译 |
| tools | Contiki 开发过程中常用到的一些工具,例如, cfs 相关的 makefsdata、网络相关的 tunslip、模拟器 cooja 和 mspsim 等 |

2.2 Contiki 在传感器平台上的通用移植方法研究

在针对不同平台进行 Contiki 移植^[8]之前,首先研究 Contiki 在 WSNs 平台上移植的通用框架^[9]。通过前面对 Contiki 源代码架构的分析可知,Contiki 的移植主要涉及 platform 目录及 cpu 目录,其它目录可以直接使用 Contiki 源代码中的相应目录^[10]。首先在 platform 目录下新建一个相应平台的子目录 platform_name,在 cpu 目录下新建一个相应平台所用 cpu 的子目录 cpu_name。然后针对不同平台的特点,在各个目录下添加相应的代码文件。下面对 platform_name 和 cpu_name 子目录下应该添加的代码文件作详细研究。

2.2.1 platform_name 子目录研究

Contiki-2.5 中 platform 目录主要包含了与平台有关的代码,例如,相应平台及编译环境的配置,编译路径等的配置。

通用的移植文件如下:

(1) 相应平台初始化文件 contiki-platform_name-main. c:该文件对相应平台的硬件进行初始化,并且调度和运行所定义的进程。

(2) 配置文件 contiki-conf. h:非常重要的文件,包含了 Contiki 对特定平台的诸多配置选项,如编译器配置、时钟配置、uIP 配置、底层内存地址配置、各层驱动配置、管脚配置等,主要是宏定义。

(3) platform-conf. h:相应平台的一些配置,主要是寄存器、管脚等的宏定义。

(4) Makefile. platform_name:该文件包含了特定平台在编译时所需的选项,它指定了底层硬件特定的代码、处理器类型以及代码路径等其他规则,以保证程序被正确编译。

(5) cfs-coffee-arch. c:Contiki 文件系统 coffee 的程序代码,主要包括对文件的一系列操作,例如,读、写、打开和关闭操作等。

(6) 应用程序文件夹 Apps:该目录下主要放置一些针对特定平台所设计的小应用程序,如摄像头程序等。

(7) 文件夹 Dev:该目录下主要放置特定平台下所接外围设备的一些配置封装函数,例如,LEDs 的各种操作程序等。另外,平台扩展的外围模块的配置代码都可以放置于此目录下。

2.2.2 cpu_name 子目录研究

Contiki-2.5 中 cpu 下各个子目录包含了相应平台 MCU 上相关模块的代码文件,相应 cpu 的配置文件以及射频芯片模块等,通用的移植文件应该包含以下几个模块:

(1) cpu 初始化模块:cpu_name. c,该文件主要包括时钟源(如 ACLK, MCLK, SMCLK)的初始化、I/O 口

的初始化、看门狗的初始化等。

(2) 时钟模块:clock. c,该模块主要是对相应 cpu 的定时/计数器进行初始化,例如基本时钟、定时/计数器、看门狗定时器等,主要是设置这些时钟的时钟源及中断服务请求程序。

(3) 串口模块:不同 cpu 有不同种类不同数量的串口,主要有 UART, SPI, I2C 等模式。可以分多个程序文件来分别配置各种串口模式,对其进行初始化,并且设置各种串口模式的中断服务请求程序。另外,在这个模块中,可以添加串口的读写等操作函数。

(4) Flash 模块:例如,flash. c, rom. c 等,该模块主要实现对存储器的操作。

(5) 射频芯片模块:该模块主要是对平台上所采用的射频芯片的设置,主要包括射频芯片的初始化函数、传输帧的格式设置、传输信道设置、传输功率设置、传输速率设置、数据检错纠错设置、传输状态设置等。

(6) 配置模块:Makefile. cpu_name,该文件配置了交叉编译环境及其规则,需指定源代码的路径。

(7) 传感器模块:该模块需参照相应 MCU 的原理图,对 MCU 上的传感器(例如,温湿度传感器、加速度仪等)进行设置。

(8) 多线程架构的实现模块:mtarch. c,该文件包含了所有底层多线程架构的具体实现,包括对操作系统中多线程的各种应用操作。

通过上述对 platform 以及 cpu 下各个文件的分析,研究了 platform 和 cpu 中子目录应该包含的文件。其它几个大目录下的内容只需要根据不同平台的特点添加相应的宏定义和更改某些细节代码即可。以上的研究可以作为 Contiki 移植的通用方法,对某个平台进行 Contiki 移植时,参考该方法并根据具体平台的特征增减相应的移植配置。

2.3 Contiki 在 MSR 节点上的移植

根据以上研究设计的移植 Contiki 的通用方法,文中在 MSR 节点上进行了 Contiki 的移植。MSR^[11]为文中移植试验的专门节点,由 ATmega128^[12]作为 cpu, CC2420^[13]作为射频芯片,并连接了几个简单传感器模块。结合 MSR 自身特点,文中设计的移植方案为:在 Contiki-2.5 源代码中的 platform 下新建子目录 MSR,在 cpu 下新建子目录 AVR128。各个目录下应该包含的文件如表 3 和表 4 所示,其它目录下只需添加相应的宏定义即可。

表 3 platform \ MSR 子目录

| | |
|-----------------------|---|
| \ platform \ MSR \ | contiki-msr-main. c, contiki-conf. h, platform-conf. h, Makefile. msr, cfs-coffee-arch. c, cfs-coffee-arch. h |
| | Dev \ leds-msr. c |

表 4 cpu \ AVR128 子目录

| | |
|---------|---|
| \ cpu \ | avr128. c, clock. c, spi. c, uart. c, flash. c, rom. c, |
| | Makefile. avr128, mtarch. c |
| | Radio \ cc2420. c, hal2420. c |
| | Dev \ sensor. c, sht11. c |

修改代码后,将代码导入 IAR^[14]新建的工程中进行编译,Contiki 是否移植成功,可以使用仿真软件 proteus 进行仿真测试。

3 Contiki 在 MSR 节点上的移植仿真测试

文中通过 proteus 设计模拟电路验证 Contiki 在 MSR 上移植的可行性。在 IAR 中编写测试程序,将编译好的 hex 文件下载到 proteus 的仿真电路中。MSR 上 LED 测试程序见图 3。

```
#include "contiki.h"
#include "dev/leds.h"
#include "avr/io.h"

/*
PROCESS (ledson _process, "ledson");
AUTOSTART _PROCESSES (&ledson _process);
*/
PROCESS _THREAD (ledson _process, ev_data)
{
    PROCESS_EXITHANDLER (goto exit);
    PROCESS_BEGIN();

    while(1)
    {
        static struct etimer et;
        etimer_set(&et, CLOCK_SECOND);
        PROCESS_WAIT_EVENT_UNTIL(etimer_expired(&et));
        leds(LEDS_ALL);

    }

exit:
    leds_off(LEDS_ALL);
    PROCESS_END();
}
```

图 3 仿真测试程序

在 proteus 设计的 MSR 仿真电路图中运行 hex 文件,显示 4 个 LED 全部亮,表明 Contiki 在 MSR 上移植成功,验证了文中提出的 Contiki 移植方法的可行性。

4 结束语

文中针对 WSNs 中节点内存和处理能力有限的问题,详细分析了节点操作系统的特点,选取 Contiki 作为研究的操作系统。文中深入研究了 Contiki 的系统架构和代码架构,设计了基于 WSNs 平台的 Contiki 通

用移植方法。

根据该通用移植方法,在实验平台 MSR 节点上设计了相应的移植方案,经过仿真测试,验证了该通用移植方法的可行性。

参考文献:

- [1] 王殊,阎毓杰,胡富平,等.无线传感器网络的理论及应用[M].北京:北京航空航天大学出版社,2007.
- [2] 杨树森,周小佳,阎斌.无线传感器网络在环境监测中的应用[J].计算机技术与发展,2008,18(9):170-172.
- [3] 孙利民.无线传感器网络[M].北京:清华大学出版社,2005.
- [4] 黄光燕,李晓维.无线传感器网络操作系统[J].信息技术快报,2005,3(3):23-24.
- [5] 李晶,王福豹,段渭军,等.无线传感器网络操作系统研究[J].计算机应用研究,2006,14(6):838-848.
- [6] Dunkels A, Gronvall B, Voigt T. Contiki - a lightweight and flexible operating system for tiny networked sensors[C]//Proc of the 29th Annual International Conference on Local Computer Networks. Washington, DC, USA: IEEE Computer Society, 2004:455-462.
- [7] Farooq M O, Kunz T. Operating Systems for Wireless Sensor Networks: A Survey[J]. Sensors, 2011, 11(6):5900-5930.
- [8] Oikonomou G, Phillips I. Experiences from porting the Contiki operating system to a popular hardware platform[C]//2011 International Conference on Distributed Computing in Sensor Systems and Workshops. [s.l.]: [s.n.], 2011:1-6.
- [9] Stan A. Porting the core of the Contiki operating system to the TelosB and MicaZ platforms[R]. [s.l.]: [s.n.], 2007.
- [10] Dokić A. MICAz and TelosB Sensor Device Driver Port to Contiki[R]. [s.l.]: [s.n.], 2007.
- [11] Zheng T, Qin Y, Gao D, et al. Environmental monitoring and air-conditioning automatic control with intelligent building wireless sensor network[C]//ICARCV. [s.l.]: [s.n.], 2010:2431-2436.
- [12] Atmel. Atmega128-128L microprocessor Datasheet[EB/OL]. [2012-05]. www.atmel.com/Images/doc2467.pdf.
- [13] Texas Instrument. CC2420 datasheet[EB/OL]. [2007-05-20]. http://www.ti.com/product/cc2420.
- [14] 唐思超.嵌入式系统软件设计实践-基于 IAR Embedded Workbench[M].北京:北京航空航天大学出版社,2010.

(上接第 133 页)

- threshold systems[J]. Phys. Rev. Lett, 2000, 84(11):2310-2313.
- [15] 王友国,刘洪伟,罗辑.基于互信息的多阈值系统中的随机谐振现象研究[J].计算机技术与发展,2010,20(6):89-92.

- [16] 杨祥龙,汪乐宇.随机共振技术在弱信号检测中的应用[J].电路与系统学报,2001,6(2):94-97.
- [17] 王利亚,蔡文生,印春生,等.自适应随机共振算法用于微弱信号检测[J].高等学校化学学报,2001,22(5):762-763.