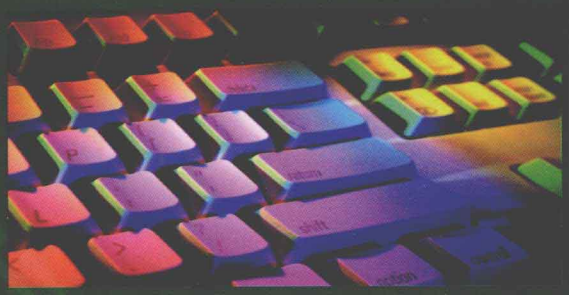
普通高等教育物联网工程专业规划教材

物联网工程实训教程
——实验、案例和习题解答

王志良　王新平　主编



附光盘

机械工业出版社
CHINA MACHINE PRESS

普通高等教育物联网工程专业规划教材

物联网工程实训教程——实验、案例和习题解答

王志良　王新平　主编

机械工业出版社

本书是一本讲述物联网工程实训的教材，从物联网实验和案例两大方面入手，它与《物联网工程概论》一书联合使用，构成了物联网工程的完善知识体系。实验包含 TinyOS、NS2 仿真及云计算的系统实验和 RFID、ZigBee 和 M2M 的基础实验。案例包括认识案例和设计案例两部分，使读者从感性认识物联网到了解项目实施有一个实际体验过程。本书图文并茂，在写作构思和结构编排上力争为读者提供全面、系统的讲述。为方便读者，本书还提供了《物联网工程概论》习题与思考题的详细解答（见配套光盘）。

本书可作为物联网工程专业及其相关专业的教材使用，供需要掌握物联网基础知识的高年级本科生学习和研究生选读，还可作为希望了解物联网知识的企业管理者、科研人员、高等院校教师等读者朋友的参考用书。

图书在版编目（CIP）数据

物联网工程实训教程：实验、案例和习题解答/王志良，王新平主编. —北京：机械工业出版社，2011.9

普通高等教育物联网工程专业规划教材

ISBN 978-7-111-35702-5

Ⅰ.①物… Ⅱ.①王…②王… Ⅲ.①互联网络-应用-高等学校-教学参考资料②智能技术-应用-高等学校-教学参考资料 Ⅳ.①TP393.4②TP18

中国版本图书馆 CIP 数据核字（2011）第 170742 号

机械工业出版社（北京市百万庄大街 22 号 邮政编码 100037）

策划编辑：牛新国 责任编辑：朱 林 责任校对：刘怡丹

封面设计：路恩中 责任印制：乔 宇

三河市国英印务有限公司印刷

2011 年 9 月第 1 版第 1 次印刷

184mm×260mm · 14 印张 · 345 千字

0001— 3000 册

标准书号：ISBN 978-7-111-35702-5

ISBN 978-7-89433-098-7（光盘）

定价：39.80 元（含 1CD）

凡购本书，如有缺页、倒页、脱页，由本社发行部调换

电话服务

社 服 务 中 心：（010）88361066

销 售 一 部：（010）68326294

销 售 二 部：（010）88379649

读者购书热线：（010）88379203

网络服务

门户网：http://www.cmpbook.com

教材网：http://www.cmpedu.com

封面无防伪标均为盗版

前　言

本书是《物联网工程概论》的姊妹篇，包括3个部分：第一部分讲述物联网实验，包括系统仿真实验和物联网基础实验；第二部分讲述物联网应用案例，包括认识案例和设计案例；第三部分是《物联网工程概论》的习题解答（见配套光盘）。

全书共分为14章，第一部分共分6章，第1章为基于SNAP平台的TinyOS实验；第2章为NS2网络仿真实验；第3章为使用VS2010开发部署Windows Azure应用程序；第4章为基于RFID射频识别实验；第5章为ZigBee实验；第6章为M2M实验。第二部分共分8章，前6章是认识案例，后2章是设计案例，第7章为智慧城市；第8章为灾害监测；第9章为智能电网；第10章为车联网；第11章为智慧校园；第12章为智能家居；第13章为家庭智能终端机器人；第14章为基于ZigBee的老人身体状态监测。

本书有如下特点：内容完整：实验、案例和习题解答3个部分和《物联网工程概论》构成了物联网教学的完整体系。

层次分明：实验分为系统仿真实验和基础实验，涵盖了RFID、ZigBee、M2M和云计算实验。案例分为认识案例和设计案例，设计案例以项目分析为框架，从需求分析到设计到制作，形成了一个完整的工程，从中可以体会到工程项目实施的过程。

课件丰富：为配合教学，本书提供所有实验和案例课件。

本书由王志良、王新平任主编，王志良制定了本书大纲、内容安排并指导文字写作，王新平负责全书的统稿和组织工作。王志良、史建雷参与了第1章的编写工作；王志良、李云龙参与了第2章的编写工作；王粉花、李明参与了第3章的编写工作；王粉花、于浩雷参与了第4章的编写工作；王新平、王鲁参与了第5章的编写工作；王新平、周翔参与了第6章的编写工作；王新平、胡余参与了第7章的编写工作；王新平、董佃超参与了第8章的编写工作；王新平、张华伟参与了第9章的编写工作；王新平、牛晓鹏参与了第10章的编写工作；王志良、晓虎尔参与了第11章的编写工作；王志良、毛昌参与了第12章的编写工作；王志良、王嵘参与了第13章的编写工作；王志良、王玉财参与了第14章的编写工作。

感谢微软公司软件架构资深顾问方国伟先生对于Azure云计算平台实验给予的支持与帮助。感谢研华公司培训中心李嘉先生对于智能电网和智慧校园案例提供的支持与写作指导。

作为全国高校物联网及其相关专业教学指导小组和物联网工程专业教学研究专家组成员，作者在其组织的物联网工程教学研讨活动中，汲取了物联网工程的教学理念，对于此书的编辑写作受益匪浅。本书的出版还得到了机械工业出版社的大力支持，同时得到第七批国家级高等学校（物联网工程）特色专业建设点项目、中央高校基本科研业务费专项资金、学校研究型教学项目的支持和资助，在此一并表示感谢。

因为时间有限，有些内容本书未能全部涵盖。同时，由于作者的认识领悟能力有限，书中难免存在的缺点与疏漏，敬请各位专家以及广大读者批评指正。

王志良、王新平
于北京科技大学
2011年6月

目　录

前言

实　验　篇

第 1 章　基于 SNAP 平台的 TinyOS 实验 …… 1
1.1　无线传感器网络简介 …… 1
1.2　TinyOS 的由来 …… 1
1.3　SNAP 平台介绍 …… 2
1.4　软件平台介绍 …… 2
1.4.1　TinyOS 介绍 …… 2
1.4.2　NesC 语言介绍 …… 3
1.5　硬件平台介绍 …… 4
1.5.1　节点介绍 …… 4
1.5.2　网关介绍 …… 5
1.6　TinyOS 实验 …… 6
1.6.1　串口控制 LED 实验 …… 6
1.6.2　点对点无线通信实验 …… 10
1.6.3　传感器数据采集实验 …… 12
1.6.4　组网协议实验 …… 15
习题与思考题 …… 17
参考文献 …… 17

第 2 章　NS2 网络仿真实验 …… 18
2.1　NS2 基础知识 …… 18
2.1.1　NS2 概述 …… 18
2.1.2　Tcl 和 OTcl …… 18
2.1.3　NS2 的功能模块 …… 19
2.1.4　NS2 的仿真元素 …… 20
2.1.5　NS2 的仿真过程 …… 20
2.2　NS2 实验部分 …… 21
2.2.1　NS2 的实验环境安装与基本操作 …… 21
2.2.2　NS2 的 Tcl 脚本生成及仿真结果分析 …… 29
习题与思考题 …… 39
参考文献 …… 39

第 3 章　使用 VS2010 开发部署 Windows Azure 应用程序 …… 40
3.1　Windows Azure Platform …… 40
3.2　开发部署 Azure 应用程序环境要求 …… 45
3.3　微软云计算实验 …… 45
3.3.1　开发部署“Hello Azure”云计算应用程序 …… 45
3.3.2　开发部署云存储应用程序 …… 49
3.4　本章小结 …… 57
习题与思考题 …… 58
参考文献 …… 58

第 4 章　基于 RFID 射频识别实验 …… 59
4.1　HBE-RFID-REX 射频识别系统 …… 59
4.1.1　HBE-RFID-REX 的整体结构 …… 59
4.1.2　HBE-RFID-REX 的编码模块 …… 60
4.1.3　HBE-RFID-REX 的解码模块 …… 63
4.1.4　HBE-RFID-REX 的调制解调模块 …… 63
4.2　HBE-RFID-REX 的嵌入式系统 …… 64
4.3　基于 RFID 射频识别实验 …… 66
4.3.1　RFID 数据编码实验 …… 66
4.3.2　HBE-RFID-REX 应答器读写实验 …… 67
4.3.3　HBE-RFID-REX 系统 ASK 调制解调 …… 70
习题与思考题 …… 72
参考文献 …… 72

第 5 章　ZigBee 实验 …… 73
5.1　概要 …… 73
5.2　目标 …… 73
5.3　基础知识 …… 73
5.3.1　ZigBee 和 IEEE 802.15.4—2006 标准 …… 73
5.3.2　IEEE 802.15.4 技术 …… 74
5.3.3　ZigBee 技术 …… 76
5.4　Z-Stack 程序 …… 78
5.4.1　IAR 编译器 …… 78
5.4.2　Z-Stack 的运行结构 …… 79

5.4.3 通过 IAR 编译器编译的方法 …… 84
5.4.4 通过 IAR 编译器下载的方法 …… 86
5.5 ZigBee 实验 …… 86
5.5.1 MyApp 实验 …… 86
5.5.2 基于 Packet Sniffer 的数据分析练习 …… 87
5.5.3 基于 Z-Tool 的定位实验 …… 89
习题与思考题 …… 92
参考文献 …… 92
第 6 章 M2M 实验 …… 93
6.1 M2M 基础知识 …… 93
6.1.1 M2M 概述 …… 93
6.1.2 GSM/GPRS 技术 …… 94
6.1.3 常用的 AT 指令 …… 94
6.1.4 实验硬件简介 …… 95
6.2 M2M 实验部分 …… 95
6.2.1 实验目的 …… 95
6.2.2 实验内容 …… 95
6.2.3 实验所用仪表及设备 …… 95
6.2.4 实验步骤 …… 96
6.2.5 实验报告要求 …… 101
习题与思考题 …… 102
参考文献 …… 102

案 例 篇

第 7 章 智慧城市 …… 103
7.1 智慧城市概述 …… 103
7.1.1 智慧城市的起源和发展 …… 103
7.1.2 什么是智慧城市 …… 105
7.1.3 智慧城市的架构 …… 106
7.1.4 物联网与智慧城市 …… 110
7.2 数字城管呼叫中心 …… 110
7.2.1 系统需求分析 …… 111
7.2.2 系统功能 …… 112
7.2.3 系统整体框架 …… 113
习题与思考题 …… 115
参考文献 …… 115
第 8 章 灾害监测 …… 116
8.1 地质灾害 …… 116
8.1.1 地质灾害综述 …… 116
8.1.2 地质灾害预警的意义 …… 116
8.1.3 地质灾害系统框架 …… 116
8.2 野外信息采集和发送系统 …… 116
8.2.1 采集系统硬件实现 …… 116
8.2.2 北斗通信型用户机 …… 118
8.2.3 野外信息发送系统的运行机制 … 118
8.2.4 野外信息发送系统的具体实现 … 120
8.3 北斗卫星定位导航系统 …… 120
8.3.1 北斗卫星导航系统组成 …… 120
8.3.2 北斗卫星导航定位原理 …… 120
8.3.3 北斗卫星导航系统三大功能 …… 121
8.3.4 北斗卫星导航系统应用 …… 121
8.4 地质灾害监测管理中心软件系统及支撑技术 …… 121
8.4.1 通信控制子系统 …… 121
8.4.2 数据库存储子系统 …… 123
8.4.3 滑坡监测数据处理软件系统 …… 126
8.5 小结 …… 127
8.5.1 增加传感器的种类与数量，扩大监测内容 …… 127
8.5.2 扩大监测范围，实现多区域联合测试 …… 127
习题与思考题 …… 127
参考文献 …… 127
第 9 章 智能电网 …… 128
9.1 智能电网简介 …… 128
9.1.1 物联网与智能电网 …… 128
9.1.2 智能电网基本架构 …… 128
9.1.3 物联网在智能电网中的应用 …… 129
9.1.4 统一坚强智能电网 …… 130
9.1.5 智能电网支撑技术 …… 131
9.2 分布式发电与微电网技术 …… 133
9.3 智能用电关键技术 …… 134
9.4 研华智能抄表系统 …… 136
9.4.1 需求分析 …… 136
9.4.2 系统功能 …… 137
9.4.3 系统结构及工作原理 …… 137
9.4.4 研华解决方案 …… 138
习题与思考题 …… 139
参考文献 …… 139
第 10 章 车联网 …… 140
10.1 智能交通系统 …… 140
10.1.1 智能交通系统综述及服务框架 …… 140
10.1.2 智能交通与物联网的融合 …… 140

10.2 车联网的相关概念 …… 140
10.2.1 车联网的体系架构 …… 140
10.2.2 车联网的定义 …… 141
10.2.3 车联网的工作原理 …… 141
10.3 车联网的研究领域 …… 141
10.4 车联网的研究意义 …… 142
10.4.1 交通问题日益严重 …… 142
10.4.2 车联网带来的效益 …… 143
10.5 GPS 车载定位终端的设计 …… 144
10.5.1 需求分析 …… 144
10.5.2 系统功能 …… 144
10.5.3 系统软硬件选型 …… 144
10.5.4 系统参数设定 …… 145
10.5.5 车载导航系统硬件设计 …… 145
10.5.6 车载导航系统软件设计 …… 148
10.6 小结 …… 150
习题与思考题 …… 150
参考文献 …… 151
第 11 章 智慧校园 …… 152
11.1 智慧校园概述 …… 152
11.1.1 数字化校园的发展历程 …… 152
11.1.2 什么是智慧校园 …… 154
11.2 智慧校园的架构与技术核心 …… 154
11.2.1 智能教育管理体系 …… 154
11.2.2 智能化教学环境 …… 157
11.3 智慧校园的应用 …… 159
习题与思考题 …… 164
参考文献 …… 164
第 12 章 智能家居 …… 166
12.1 智能家居概述 …… 166
12.1.1 智能家居的概念 …… 166
12.1.2 智能家居国内外发展现状 …… 166
12.1.3 智能家居发展的特点和方向 …… 171
12.2 智能家居的功能、结构和特点 …… 172
12.2.1 智能家居的功能 …… 172
12.2.2 智能家居的体系结构 …… 173
12.2.3 智能家居的平台特点 …… 176
12.3 智能家居的关键技术 …… 177
12.3.1 家庭网络内部组网技术 …… 177
12.3.2 家庭网络中间件技术 …… 177
12.3.3 智能家居远程控制技术 …… 178
习题与思考题 …… 179
参考文献 …… 180
第 13 章 家庭智能终端机器人 …… 181
13.1 设计背景 …… 181
13.2 功能设计 …… 182
13.3 外形设计 …… 184
13.4 机械设计 …… 186
13.5 硬件设计 …… 187
13.5.1 底层电路板 …… 188
13.5.2 嵌入式硬件环境 …… 190
13.6 通信 …… 192
13.6.1 串口通信 …… 192
13.6.2 网络通信 …… 193
13.7 软件设计 …… 194
13.7.1 软件界面开发 …… 194
13.7.2 网络传输开发 …… 196
13.7.3 视频采集开发 …… 196
13.8 小结 …… 198
习题与思考题 …… 198
参考文献 …… 199
第 14 章 基于 ZigBee 的老人身体状态监测 …… 200
14.1 需求分析 …… 200
14.2 硬件和网络选择 …… 202
14.2.1 MMA7260 加速度传感器 …… 202
14.2.2 WTW240-28P 语音模块 …… 204
14.2.3 微处理器 …… 205
14.2.4 无线网络 …… 206
14.3 系统设计 …… 207
14.3.1 系统整体构架 …… 207
14.3.2 终端监测仪 …… 207
14.3.3 ZigBee 协调器 …… 208
14.3.4 监控中心软件 …… 208
14.4 装置实现 …… 210
14.4.1 网关硬件系统 …… 210
14.4.2 终端传感器系统 …… 211
14.4.3 射频电路注意事项 …… 211
14.4.4 编写软件程序 …… 211
14.5 数据分析与调试经验 …… 212
14.5.1 数据分析 …… 212
14.5.2 调试经验 …… 214
14.5.3 实际测试及问题分析 …… 215
14.6 小结 …… 216
习题与思考题 …… 217
参考文献 …… 217

实　验　篇

第 1 章　基于 SNAP 平台的 TinyOS 实验

1.1　无线传感器网络简介

无线传感器网络（Wireless Sensor Network，WSN）是由密集部署于监控区域内的微型传感器节点组成的一种无中心节点的全分布系统。这些低成本、低功耗，具有感知、数据处理和通信能力的节点通过无线信道相连，自组织构成了网络系统。传感器节点借助于其内置的形式多样的传感器，探测包括温度、湿度、噪声、光强度等众多人们感兴趣的物理现象。由于每个传感器节点都可以直接嵌入到相应的设备或环境中，所以它们具有很强的灵活性和移动性。节点之间的通信采用多跳、对等的方式，可以有效地避免信息在长距离无线传播的过程中遇到干扰和自身信号衰减等问题。通过相应的网络设备，无线传感器网络还可以与现有的网络基础设施相连，将采集到的信息通过 Internet 或移动通信网络发送至远程终端。

典型的无线传感器网络一般包括传感器节点（Sensor node）、汇聚节点（Sink node）和任务管理节点。

1.2　TinyOS 的由来

针对无线传感器网络的硬件节点相对简单的特点，一些学者认为对现有的嵌入式操作系统，如 VxWorks、WinCE 和 Linux，进行必要的裁剪定制后，就可以在节点上运行。但上述嵌入式操作系统设计时没有考虑到无线传感网络节点的系统资源十分有限和运行的特点，很难在无线传感器网络节点上取得好的运行效果。

研究人员通过实验发现无线传感器网络节点有两个比较突出的特点：一个特点是并发性很密集，即可能存在多个需要同时执行的逻辑控制，需要操作系统能有效提供处理这种发生频繁、并发程度高、执行过程比较短的逻辑控制流的能力；另一个特点是无线传感器节点模块化程度高，一个节点通常划分为几个主要的部分，在不影响整体性能的情况下，要求操作系统能够让程序方便地对模块进行控制，使程序中的各个部分能方便地进行组合。

针对无线传感器网络节点系统资源有限和运行特点，美国加州大学伯克利分校科研人员在设计 TinyOS（微型操作系统）的过程中，引入轻量线程、主动消息、事件驱动模式、基于组件编程、硬件抽象层和并行处理的研究成果，更好地满足了无线传感器网络节点运行的特点。

1.3 SNAP 平台介绍

1. 无线传感器网络辅助平台

无线传感器网络辅助平台（Wireless Sensor Network Assistant Platform，Wireless SNAP）是由广州市香港科大霍英东研究院数字生活研究中心研发的，专为无线传感器网络（WSN）教学和研究服务的系统级辅助平台。SNAP 基于用户控制和分析软件，通过实地部署传感器节点和数据转发处理网关，实时详尽地了解 WSN 内部节点的通信数据、节点内部状态信息、网络通信拓扑以及网络中节点的能耗信息等。同时，SNAP 可以对大规模 WSN 中的节点进行控制、批量更新编译好的 NesC 程序以及对所有数据进行保存回放和自定义解析处理。

SNAP 以服务 WSN 的教学和研究为目的，面向教学实验，教师可以基于 SNAP 建立一整套全面的针对 WSN 课程的教学演示和实验，有效地丰富 WSN 课程的教学手段，提高学生学习的兴趣和主动性。学生可以从硬件和软件两个方面掌握 WSN 的采集、路由、能耗、拓扑控制、定位以及安全等相关技术，并在实际应用中方便地开发和加深对 WSN 的理解，对其有更加全面的认识。面向 WSN 的研究，旨在帮助 WSN 的设计者彻底地从繁琐的测试数据采集工作中解放出来，并降低 WSN 研究成本、减短研发周期，尤其在 WSN 的大规模应用中，可以使用该平台来完成应用系统部署前的评估和应用系统部署后的监控维护工作。

2. SNAP 套装软件配置及相关说明

1）SNAPMonitor：用于 SNAP 系统显示和控制的人机交互软件。

2）D-SNAP-2. x：基于 TinyOS-2. 0. 8 版本的传感器节点微型操作系统，用于编译传感器节点 NesC 程序。

3）VMware Player + ubuntu10. 04 镜像：基于 VMware 的 Linux 操作系统镜像，为 D-TinyOS-2. x 提供运行环境。

4）NTP 安装包：安装网络时间协议（Network Time Protocol，NTP），用于 SnapGate 与 PC 终端的时间同步。

5）Microsoft. NET framework 3. 5 SP1 安装包：SNAPMonitor 运行环境。

1.4 软件平台介绍

1.4.1 TinyOS 介绍

TinyOS 是美国加州大学伯克利分校专门为 WSN 开发的一种微型操作系统。确切地说，TinyOS 是一个适用于网络化嵌入式系统的编程框架，通过在这个框架内链接一组必要的组件，就能方便地编译出面向特定应用的操作系统，这对于存储资源极为有限的系统来说非常重要。针对 WSN 内节点众多，以及多并发操作的工作方式，该操作系统采用事件驱动的体系结构。与经典的操作系统相比，完整的系统由一个调度器和一些组件组成，应用程序与组件一起编译成系统。组件由下到上可分为硬件抽象组件、综合硬件组件和高层软件组件，高层组件向底层组件发出命令，底层组件向高层组件报告事件。调度器具有两层结构：第一层维护着命令和事件，它主要是在硬件中断发生时对组件的状态进行处理；第二层维护任务

（负责各种计算），只有当组件状态维护工作完成后，任务才能被调度。

TinyOS 的组件层次结构就如同一个网络协议栈，如图 1-1 所示，底层的组件负责接收和发送最原始的数据位，而高层的组件对这些位数据进行编码、解码，更高层的组件则负责数据的打包、路由和传输。

传感器节点底层的硬件细节由硬件抽象层屏蔽，这样使上层程序员可以更关注基于应用的开发，底层的系统程序员则可以不断增加对不同平台的支持，使得 TinyOS 可以适用于更广泛的硬件平台。基于 TinyOS 的上层应用程序的开发使用 NesC 语言，用组件-接口的模型和设计方法来快速便捷地搭建部署应用。但底层基于硬件设备驱动的开发还是使用 C 语言或者汇编语言，然后通过系统组件的方式封装，以提供给其他开发者使用。

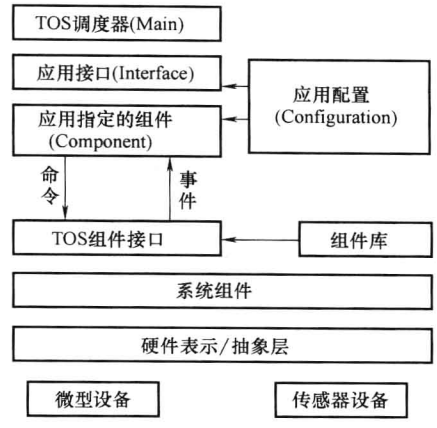


图 1-1　TinyOS 的组件层次结构

TinyOS 的设计基于组件的架构方式，可以很简单方便地编写程序，使得能够快速实现各种应用。在传感器网络天生就有内存严格限制的条件下，这可以用最小代码来创新和实现。TinyOS 的组件库包括网络协议、分布式服务、传感器驱动和数据获取工具。所有这些都可以像这样的使用或者进一步精练到用户自己的应用中。可以把 TinyOS 看成是一个可以与传感器进行交互的应用程序接口（API），它们之间可以进行各种通信。TinyOS 基于事件驱动执行架构（Event Driven Architecture），应用程序都是基于事件驱动模式的，采用事件触发去唤醒传感器工作，这使得更细密的功耗管理成为可能。

1.4.2　NesC 语言介绍

TinyOS 最初是用 C 语言和汇编语言编写的，但是科研人员的进一步研究发现，C 语言不能有效、方便地支持面向传感器网络的应用和操作系统的开发，于是美国加州大学伯克利分校在 C 语言的基础上进行了一定的扩展，开发了支持组件化编程的 NesC 语言。TinyOS 以及基于 TinyOS 的应用程序都是用 NesC 编写的。

NesC 语言是一个提供了包含组件机制、事件驱动机制和并发型等特征的编程模式，满足了面向传感器网络的操作系统和应用程序的设计要求，并降低了复杂度。应用程序由一组可重用的系统组件和专门的应用程序代码组成，不明确区分软硬件界线，随应用程序和硬件平台而变化。用 C 语言实现的目标代码比较长，而使用 NesC 语言产生的目标代码相对较短。一个 NesC 语言的应用程序只使用相关的组件，而不需要操作系统的整体运行。所以只要连接需要使用的组件就可以完成功能，生成的代码短且高效。

TinyOS 以及基于 TinyOS 的应用程序是由许多功能独立且相互联系的组件（Component）组成的。一个 NesC 程序由一个或多个组件组合（Assembled）或连接（Wired）而成。组件定义了两种：配置（Configuration）和模块（Module），配件定义了程序使用的组件以及组件间的连接关系，模块则是组件的具体实现。一个组件使用（Use）或者提供（Provide）若干接口（Interface），组件的接口是实现组件间联系的通道，如果组件实现的函数没有在它的接口中说明，就不能被其他组件使用，这实际上也是组件化编程的一个重要特征。接口的

使用者需要实现的一组功能函数称为事件（Event），接口的提供者需要实现一组功能函数，称为命令（Commands）。可以说接口是一系列声明的有名函数集合，同时接口是连接不同组件的纽带。

我们可以将 NesC 的应用程序概括为以下 3 种类型：①接口定义文件——app. ncc ；②模块文件——app_P. nc 或 app_M. nc；③配置文件——appC. nc。

NesC 模块调用关系如图 1-2 所示。配置文件中“模块 A 接口 a- > 模块 B 接口 b”，即表示模块 A 中的接口 a 调用了模块 B 中的接口 b，格式为“调用者- > 提供者”。接口定义文件中定义了接口的成员，但成员的具体实现还需在模块文件的 implementation 里完成，同时模块中设置了接口类型，以便使用。

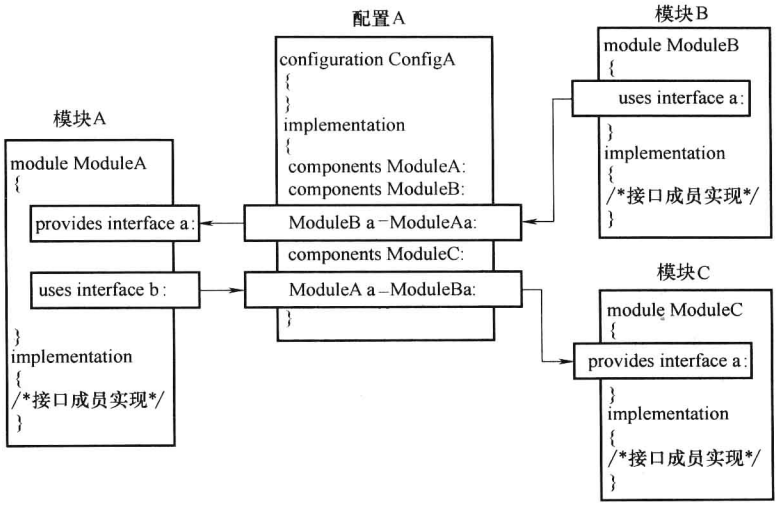


图 1-2 NesC 模块调用关系

1.5 硬件平台介绍

1.5.1 节点介绍

Telosb 节点是目前在研究领域应用非常广泛的开源设计节点，将所有无线传感器教学和研究必要的模块全部集中在一个平台上。它包括串口转通用串行总线（USB）编程和数据接口，一个基于 IEEE 802. 15. 4/ZigBee 协议、工作在 2. 4G 频段的收发芯片，一个内置的射频天线，一个带有扩展内存的低功耗微处理器以及一些可以选择的传感器。它可选择安装湿度、温度和光线感应器。Telosb 可以通过两节 AA 电池供电，同时也可以通过 USB 端口对其进行供电。如果通过 USB 插进电脑，就不需要 AA 电池。Telosb 平台提供完整的外设，包括一个 12 位 A-D 转换器和 D-A 转换器、定时器、I^2C、SPI（同步并行接口）、UART（通用异步收发器）及 DMA（直接存储器访问）控制器，拥有外置闪速存储器（Flash memory），能够保存 1024KB 的数据。在源代码开放的 TinyOS 支持下，Telos Rev B 应用最新的无线协议和开放源代码软件。Telos Rev B 携带一个容易使用的 USB 协议，使用它可以直接对 Telosb 平台进行编程、调试及数据采集。Telosb 节点实物图，如图 1-3 和图 1-4 所示。

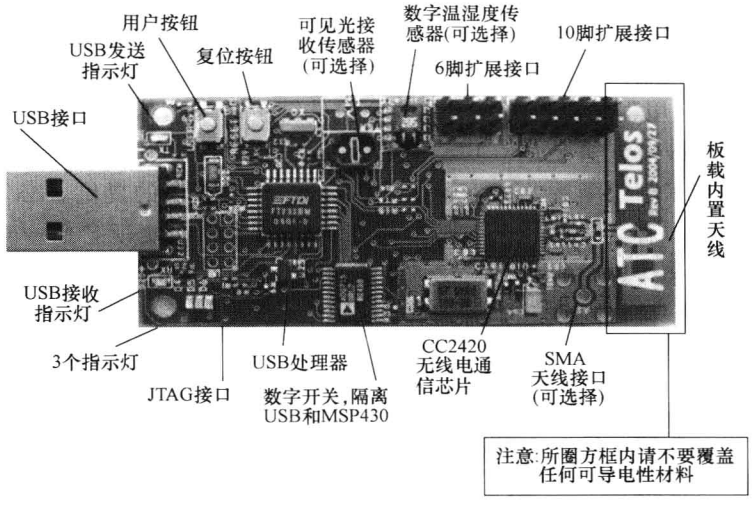


图 1-3　Telosb 节点的正面

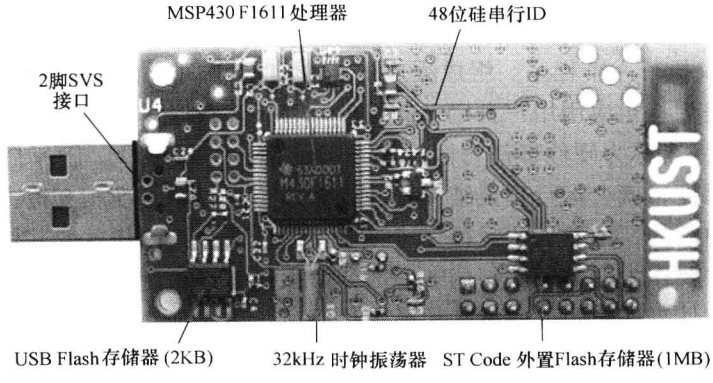


图 1-4　Telosb 节点的反面

Telosb 为使用者提供了连接外部硬件的能力，这两个连接器和板载跳线可以配置来控制模拟传感器和数字的外围设备以及液晶显示器（LCD）的显示。

1.5.2　网关介绍

Snapgate 是针对 SNAP 自主开发的基于嵌入式 Linux 的一款数据采集和转发网关。作为一个高性能的嵌入式微系统，Snapgate 提供完善的连接端口，可支持 7 个 USB 接口与传感器节点的连接和通信，1 个专属 USB 接口可以连接无线通信扩展。同时提供 RJ45 网络接口，通过 10/100（Mbit/s）的以太网与外界通信。SNAP 网关如图 1-5 所示。

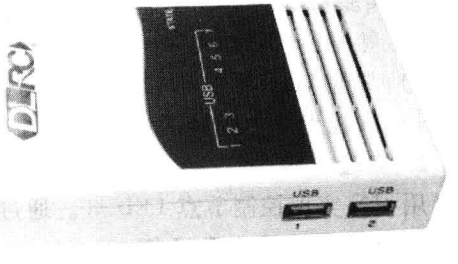


图 1-5　SNAP 网关

1.6 TinyOS 实验

任何一个 NesC 应用程序都是由一个或多个组件链接起来，从而形成一个完整的可执行程序。组件提供（Provide）并使用（Use）接口。这些接口是组件的唯一访问点，并且它们是双向的。接口声明了一组函数，称为命令（Command），接口的提供者必须实现它们；还声明了另外一组函数，称为事件（Event），接口的使用者必须实现它们。对于一个组件而言，如果它要使用某个接口中的命令，它必须实现这个接口的事件。一个组件可以使用或提供多个接口以及同一个接口的多个实例。

1. 实现

在 NesC 中有两种类型的组件，分别称为模块（Module）和配置（Configuration）。模块提供应用程序代码，实现一个或多个接口；配置则是用来将其他组件装配起来的组件，将各个组件所使用的接口与其他组件提供的接口连接在一起。这种行为称为导通（Wiring）。每个 NesC 应用程序都由一个顶级配置所描述，其内容就是将该应用程序所用到的所有组件导通起来，形成一个有机整体。

NesC 的所有源文件，包括 interface、Module 和配置，其文件扩展名都是“.nc”。

2. 并发模型（Concurrency Model）

TinyOS 一次仅执行一个程序。组成程序的组件来自于两个方面：一部分是系统提供的组件；另一部分是为特定应用用户自定义的组件。程序运行时，有两个执行线程：一个称为任务（Task）；另一个称为硬件事件句柄（Hardware event handler）。任务是被延期执行的函数，它们一旦被调度，就会运行直至结束，并且在运行过程中不准相互抢占。硬件事件句柄是用来处理硬件中断的，虽然也要运行完毕，但它们可能会抢占任务或其他硬件事件句柄的执行。命令和事件要作为硬件事件句柄的一部分而执行，必须使用关键字 async 来声明。

因为任务和硬件事件句柄可能被其他异步代码所抢占，所以 NesC 程序容易受到特定竞争条件的影响，导致产生不一致或不正确的数据。避免竞争的办法通常是在任务内排他地访问共享数据，或访问所有数据都使用原子语句。NesC 编译器会在编译时向程序员报告潜在的数据争用，这里面可能包含事实上并不可能发生的冲突。如果程序员确实可以担保对某个数据的访问不会导致麻烦，可以将该变量使用关键字 norace 来声明，但使用这个关键字一定要格外小心。

1.6.1 串口控制 LED 实验

1. 实验目的

1）熟悉传感器网络节点程序结构和编译环境，掌握 NesC 程序编译和上传过程。

2）掌握 TinyOS 中的定时器组件。

3）掌握无线传感器的串口通信方法，并能熟练运用于程序调试。

2. 实验内容

用程序实现控制节点 LED 灯。通过串口控制 LED 灯的闪烁或熄灭的周期等。

3. 实验所用仪表及设备

硬件：PC 一台，SNAP 套件提供了 Telosb 型号的传感器节点和 SnapGate 数据采集和转

发网关。

软件：SnapMonitor 软件（集成了串口调试助手）一套。

4. 实验原理

在 TinyOS 下，定时器的组件一般为通用组件，可以通过 new 来实例化最多 255 个定时器，Timer1 提供了 TimerMilliC 等组件，这些组件可供用户调用，其命名规则为定时器类型、精度、位数。本实验是用定时器通用组件 TimerMilliC，它由 TimerMilliC 组件提供 Timer<Tmilli>接口。

先看看 Timer 的配置文件：

```
configuration TimerAppC
{
}
implementation
{
  components MainC,TimerC,LedsC;
  components new TimerMilliC() as Timer0;
  TimerC→MainC.Boot;
  TimerC.Timer0→Timer0;
  TimerC.Leds→LedsC;
}
```

关键字 configuration，它表明这是一个配置文件。开头的两行

```
configuration TimerAppC
  {
  }
```

只是简单地声明了该配置名为 TimerAppC。跟模块一样，在声明后的这个花括号内可以指定 uses 子句和 provides 子句。这一点非常重要：配置可以提供和使用接口。配置的实际内容是由跟在关键字 implementation 后面的花括号部分来实现的。

Components 这一行指定了该配置要引用的组件集合，此例中是 MainC、TimerC、LedsC 和 TimerMilliC（）。实现的剩余部分将这些组件使用的接口与提供这些接口的其他组件连接起来，即是“导通”操作。

Main 是在 TinyOS 应用程序中首先被执行的一个组件。确切地说，在 TinyOS 中执行的第一个命令是 Main.StdControl.init（），接下来是 Main.StdControl.start（）。因此，TinyOS 应用程序在其配置中必须要有 Main 组件。接口 StdControl 是用来初始化和启动 TinyOS 组件的一个公共（通用）接口，它的源文件位于 tos/interfaces/StdControl.nc，StdControl 接口定义了 3 个命令（command），分别是 init（）、start（）及 stop（）。

当组件第一次初始化时调用 init（）命令，启动时调用 start（）命令。stop（）命令是在组件停止时调用，例如将其控制的设备的电源断开。init（）命令可以被调用多次，但如果调用了 start（）命令或 stop（）命令以后就再也不能被调用。

Timer 配置中有如下三行：

```
TimerC→MainC.Boot;
```

```
TimerC. Timer0→Timer0;
TimerC. Leds→LedsC;
```

NesC 使用箭头（→）来指示和标识接口间的关系，其意义为“绑定”，即左边的接口绑定到右边的实现上。换言之，使用接口的组件在左边，提供接口的组件在右边。

“TimerC→MainC. Boot;”这一句话的意思是将组件 TimerC 所使用的接口 Boot 与组件 MainC 所提供的接口 Boot 导通起来，相当于“TimerC. Boot→MainC. Boot;”，TimerC 组件省略了 Boot 接口。箭头左边的 BlinkM. Timer 引用名为 Time 的接口（tos/interfaces/Timer. nc），而箭头右边的 SingleTimer. Timer 则指向 Timer 的实现（tos/lib/SingleTimer. nc）。箭头的作用就是将其左边的接口与其右边的实现绑定起来。其作用是将 TimerC 组件的 Boot 接口与 MainC 组件中的 Boot 接口，将 TimerC 中的 Timer0 和 SingleTimer 中的 StdControl 接口导通起来，将 TimerC 中的 Leds 与 LedsC 中的接口导通起来。这里第一行省略了 Boot。再来看看 BlinkM. nc 模块：

```
#include "Timer. h"
module TimerC( )
{
  uses   interface Timer < TMilli > as Timer0;
  uses   interface Leds;
  uses   interface Boot;
}
implementation
{
  event void Boot. booted( )
  {
    call Timer0. startPeriodic(1000);
  }
  event void Timer0. fired( )
  {
      call Leds. led0Toggle( );
  }
}
```

Timer 模块还使用了 3 个接口，分别是 Leds、Timer 和 Boot。这意味着它可能调用这些接口中声明的任何命令以及必须实现这些接口中声明的任何事件。

Leds 接口（tos/interfaces/Leds. nc）定义了多个命令，如 redOn（）、redOff（）等，其作用是将微粒上的 LED（红、绿、黄）灯打开或关闭。由于 TimerC 组件使用 Leds 接口，因此它可调用其中任一命令。请注意，Leds 仅仅只是一个接口，其实现由与使用它的组件对应的配置文件指定。此例中，在 TimerAppC. nc 中指定为 LedsC，即是要由 LedsC 来实现 Leds 接口。LedsC 位于 tos/system/LedsC. nc，与 TimerMilliC. nc 一样，同属于 TinyOS 的系统组件。

可以看出，Timer 接口除了定义了两个命令 start（）和 stop（）以外，还定义了一个事

件 fired（）。

应用程序是通过 event result_t fired（）事件知道定时器时间到的，事件是当某个事情发生时接口的实现发出信号（signal）的函数。在本例中，当指定的时间间隔到达时，fired（）事件就被触发。这是一个双向接口的例子，不仅提供被该接口使用者调用的命令，而且触发事件，该事件再调用接口使用者的处理函数。可以认为事件是接口的实现者将会调用的一个回调函数。使用接口的模块必须实现接口使用的事件。

此外，本实验还采用了串口发送数据，因此使用了 PlatformSerialC 组件，该组件提供了串口发送、串口接收功能，从而便于节点和计算机等设备通过串口交互。该组件提供了 UartStream、UartByte 等接口，分别负责发送字符串和字节。具体程序请查看源程序。

5. 实验步骤

1）将 PC、SNAP、Telosb 节点和虚拟机正确连接。

2）打开软件，并将操作目录切换到文件所在目录下，编译程序。查看 SNAP 使用说明，将节点程序上传到 SNAP 中。

3）触发节点，根据菜单显示在串口中的输入命令，查看 LED 灯的变化和串口接收到的信息。

6. 实验结果及分析

触发节点后，会看到 LED 灯闪烁，输入 A 会打开所有的 LED，输入 B 会关闭所有的 LED 灯，输入 C 会有一行提示符，让输入设定定时器的时间，然后控制 LED 灯闪烁的时间，输入 D 会关闭定时器。串口显示如图 1-6 所示，输入 C 选项如图 1-7 所示，输入定时器的时间如图 1-8 所示。

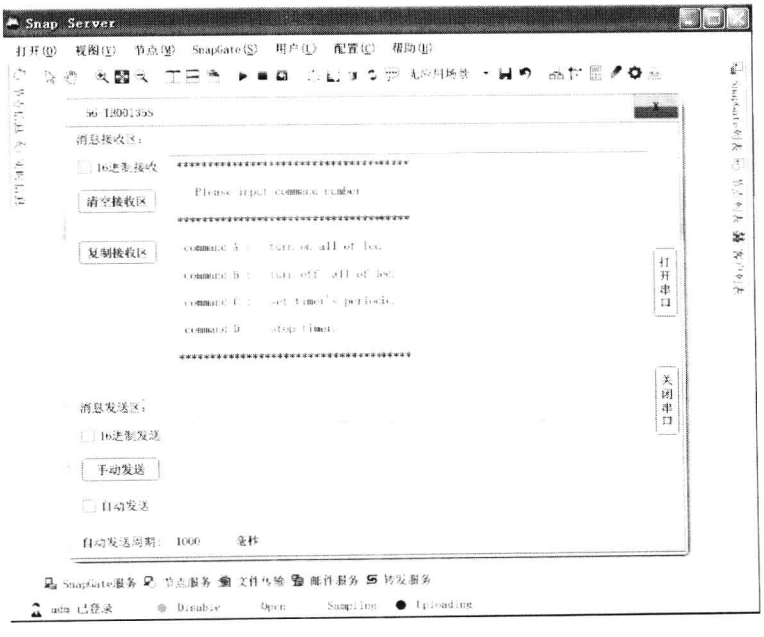


图 1-6　串口显示

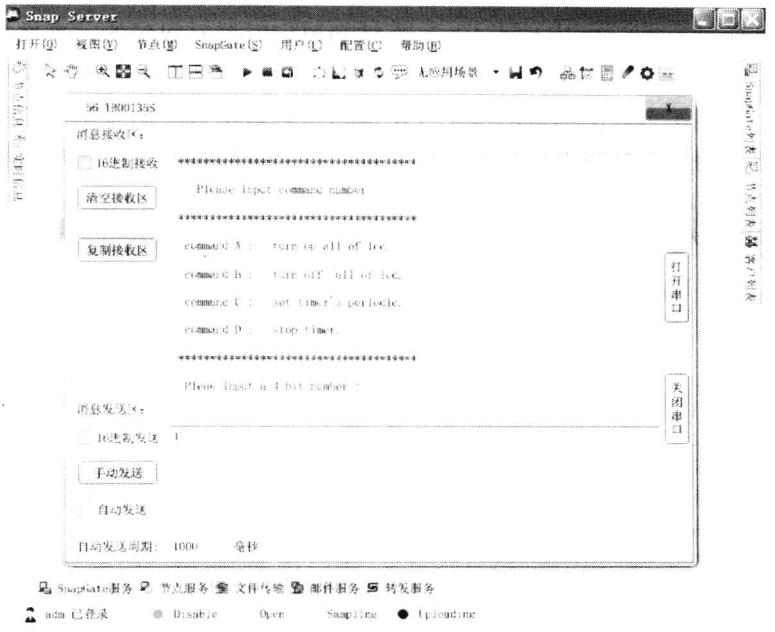


图 1-7　输入 C 选项

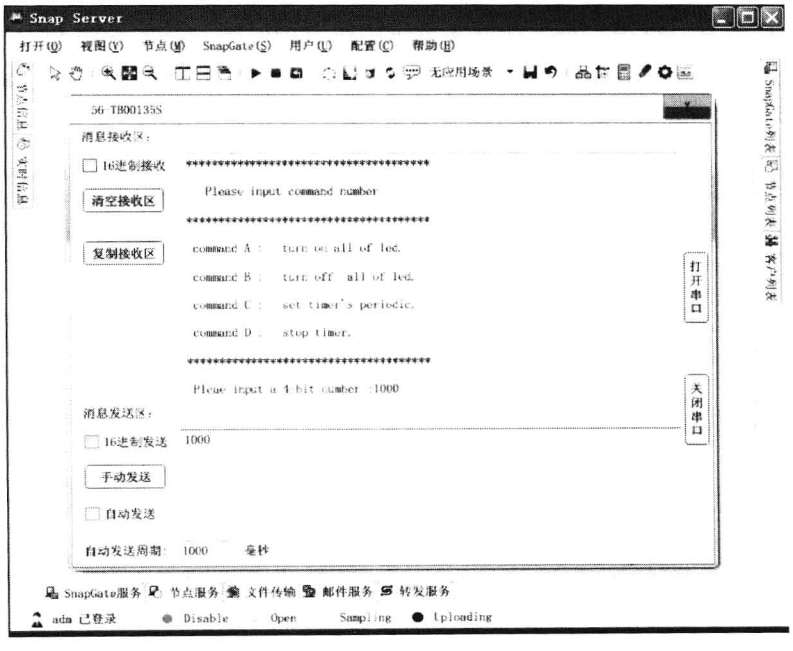


图 1-8　输入定时器的时间

1.6.2　点对点无线通信实验

1. 实验目的

学习使用传感器节点的无线模块来进行通信。