ZigBee-2006 协议栈的实现及其应用

中文摘要

ZigBee 作为一种基于开放性国际标准的低成本、低功耗、低数据速率、低复杂度、高可靠性的新型短距离无线通信技术，在能源管理和效率、家庭自动化、楼宇自动化、工业自动化以及无线传感网络等领域具有十分广阔的应用前景，已经成为当前的一个研究和应用热点。

但是，ZigBee 技术在国内的发展和应用还不够。虽然也存在一些提供 ZigBee 软硬件模块的厂商，但是其 ZigBee 协议栈一般都是基于国外的商业协议栈或者功能简单的免费协议栈。上述状况所造成的学习和研究 ZigBee 技术细节平台的缺乏反过来也影响了 ZigBee 技术在国内的发展。

本文从技术研究与实际应用的角度出发，基于 Freescale 的 MC13213 芯片构建了单芯片的 ZigBee 硬件平台，深入研读了 IEEE 802.15.4 标准和 ZigBee-2006 协议规范，详细阐述了 ZigBee 协议物理层、MAC 层和网络层的设计和实现过程，最终实现了一个功能有所裁剪的 ZigBee 协议栈。剖析了底层驱动程序实现过程中所遇到的 SPI 事务操作问题的根源；针对 MAC 层众多属性提出了一种高效的管理方法；针对 ZigBee 网络层路由算法的复杂性，实现了原理简单易于理解但功能只是稍许减弱的 AODVjr 协议。由于本协议栈中采用了分布式地址分配机制，所以也支持树状层次路由。最后，在自主研发的软硬件平台上，给出了一个车间设备监控系统的应用实例，验证了软硬件平台的可行性和正确性，并分析和实现了系统中 ZigBee 节点的低功耗。

本文所实现的 ZigBee 软硬件平台可以直接作为一个独立的模块用于实际项目中，也可以作为研究 ZigBee 协议的基础平台，具有一定的参考价值和借鉴意义。

关键词：ZigBee 协议，IEEE 802.15.4，无线个域网，MC13213，低功耗

作　　者：倪敬飞

指导老师：王宜怀

Implementation and Application of ZigBee-2006 Protocol Stack

Abstract

ZigBee is an open international standards-based novel short-range wireless communication technology with cost-effective, low-power consumption, low data-rates, low-complexity and high reliability, which has very broad application prospects on Energy Management and Efficiency, Home Automation, Building Automation, Industrial Automation and Wireless Sensor Networks. ZigBee has become a research and application hotspot currently.

However, the domestic development of ZigBee is not good enough. Although there are some related hardware and software vendors, the protocol stacks running on their hardware are generally based on foreign commercial protocol stacks or simplified free protocol stacks. The lack of study and research platform for ZigBee technical details caused by the above-mentioned situation may also affect the domestic development of ZigBee technology in return.

This paper provides a implementation of ZigBee-2006 protocol stack with some function cut-off from the view of technical research and practical application. The ZigBee hardware platform is constructed with Freescale's SiP MC13213. IEEE 802.15.4 standard and ZigBee-2006 specification are deeply studied and analyzed. This paper describes and implements the ZigBee PHY, MAC and NWK layer in detail, and analyzes the the root causes of SPI transaction protocol problem occurred in implementing the hardware driver. A highly efficient method is put forward to solve the problem of the management of numerous MAC PIB attributes. AODVjr algorithm is addressed to reduce the complexity of routing algorithm in ZigBee NWK layer, and the hierarchical routing is also supported since the use of distributed address assignment mechanism. Finally, a workshop devices monitoring system is carried out to verify the feasibility and correctness of the independent research and development hardware and software platform, and the low power consumption of the system is also analyzed and solved.

The platform can be directly used as an independent module in practical project, and it also has a valuable reference value in offering as the basic platform for the study and research of ZigBee protocol.

Key words: ZigBee protocol, IEEE 802.15.4, Wireless Personal Area Network(WPAN), MC13213, low power

Written by Ni Jingfei

Supervised by Wang Yihuai

目　录

第一章 绪论

1.1 课题背景

1.1.1 ZigBee 概述及其发展历程

1. ZigBee起源

早在上世纪末，就已经有人在考虑发展一种新的通信技术，用于传感控制(sensor and control)以及自动化应用，这个想法后来在 IEEE 802.15.4 工作组中提出来，于是就成立了 TG4 工作组[1]，致力于开发一种可应用在固定、便携或移动设备上的低成本、低功耗以及多节点[2]的低速率无线个人区域网络(Low-Rate Wireless Personal Area Network，LR-WPAN)技术标准。但是 IEEE 802 的规范只专注于底层，要达到产品的互操作和兼容性，还需要定义高层的规范[1]，于是 2001 年 ZigBee 联盟(ZigBee Alliance)成立，正式有了“ZigBee”这个名词。

2. ZigBee概述

ZigBee 一词源于蜜蜂，蜜蜂通过 ZigZag 舞蹈与同伴传达花与蜜的位置、方向、距离等信息，因而藉此作为这个短距离无线通信新技术的命名[3]。

ZigBee 是一种开放性的低成本、低功耗、低数据速率、低复杂度、双向传输、高可靠性的新型短距离无线通信技术，其突出特点[4]是应用简单，电池寿命长，成本低，可靠性高，具有自组网和自恢复能力。

ZigBee 一般采用 IEEE 802.15.4 收发器与 ZigBee 协议栈的组合，在数千个节点之间相互协调实现通信。这些节点只需要很少的能量，以接力的方式通过无线电波将数据从一个节点传到另一个节点[5]，所以它们的通信效率非常高。因此，ZigBee 在传感器网络、智能家居、工业自动化等领域有着广泛的应用。市场研究机构 NSR(Northern Sky Research)曾预估[6]，到 2010 年，全球将可望部署 5.8 亿个 ZigBee 组件，成长非常快速。

3. ZigBee发展历程

ZigBee 的发展历程可以追溯到上个世纪末[7]。ZigBee 技术发展史中的里程碑事件如图 1.1 所示。

1999 年，就已经制定了 ZigBee 技术的初始的市场需求文档(Marketing Requirement Document，MRD)。接着又制定了技术需求文档(Technical Requirement Document，TRD)，并于 2000 年底讨论和通过了 V0.2 版本，然后在 2001 年初被提交

给 IEEE 802.15.4 工作组。

2000 年底，就已经制定了 IEEE 802.15.4 标准的项目授权申请书(Project Authorization Requests，PAR)并提交到 IEEE 802.15.4 工作组讨论。然后经过多次反复评审，IEEE 802.15.4-2003 标准终于在 2003 年 5 月获得通过[2]。另外，IEEE 802.15.4-2006 标准也在 2006 年 6 月通过，对前一版进行了改进和修正，增加了几种新的频率分配，澄清了一些模糊的概念，减少了不必要的复杂性[8]。但是最新的 ZigBee 协议规范中都还是参考 IEEE 802.15.4-2003 实现的。因此，在后文中如未明确说明，IEEE 802.15.4 均是指 IEEE 802.15.4-2003。

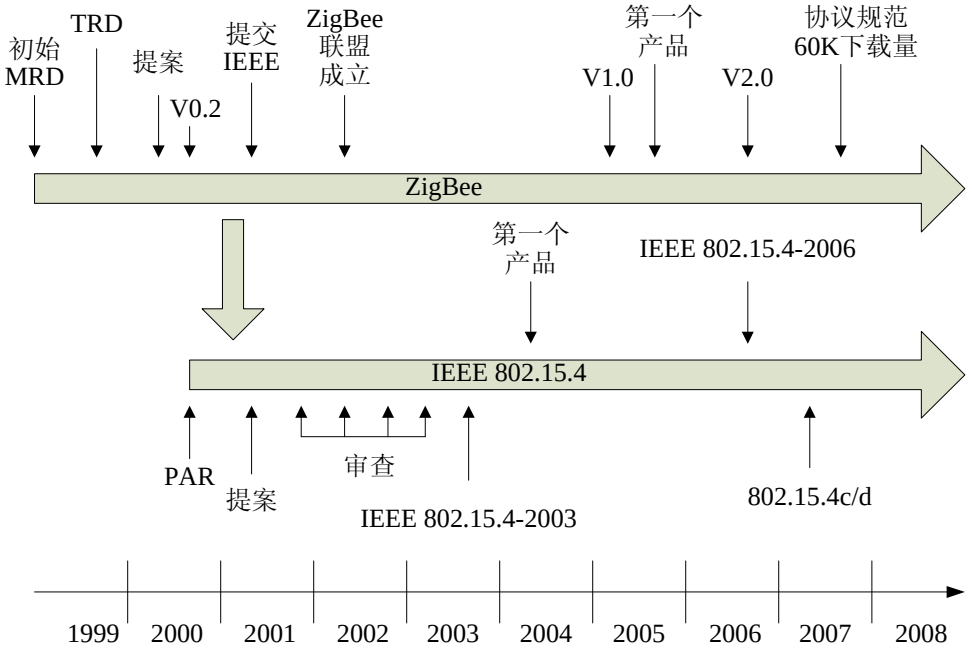


图 1.1 ZigBee 技术发展里程碑

2001 年 8 月，美国 Honeywell 等公司发起成立 ZigBee 联盟，他们提出的 ZigBee 技术被确认纳入为 IEEE 802.15.4 标准。2002 年 10 月，英国 Invensys 公司，美国摩托罗拉，荷兰飞利浦和日本三菱等重量级企业加盟 ZigBee 联盟[9]。

2004 年 12 月，ZigBee 联盟通过了 ZigBee 技术规范 V1.0 版本。这是第一个 ZigBee 规范公开版本，于 2005 年 6 月开放下载，文件内记载公布时间为 2005 年 6 月 27 日，内部文件编号为 r06，现称为 ZigBee-2004[10]。

2006 年 10 月，ZigBee 联盟通过了 ZigBee-2006 规范。这是第二个 ZigBee 规范公开版本，于 2007 年 1 月开放下载，文件内记载公布时间为 2006 年 12 月 1 日，内部文件编号为 r13，现称为 ZigBee-2006[11]。

2007 年 10 月，ZigBee 联盟通过了 ZigBee-2007 规范。这是第三个 ZigBee 规范公开版本，也是最新的版本，于 2008 年 1 月开放下载，文件内记载公布时间为 2008 年 1 月 17 日，内部文件编号为 r17，现称为 ZigBee PRO 或 ZigBee-2007[12]。

ZigBee 联盟认为目前关于 ZigBee 规范制定的主体工作已经趋近于完成，今后对于规范本身可能仅仅是执行一些维护和小的修补，预期将不再对 ZigBee 规范进行另外的更新。接下来 ZigBee 联盟将把大部分精力分配到制定公共应用规范(Public Application Profile，PAP)上面。

1.1.2 ZigBee 应用领域和目标市场

可以想象这样一种情形：当你需要开灯时，不需要走到墙壁开关处，而直接通过遥控便可；当你打开电视机时，灯光会自动减弱；当电话铃响起时或你拿起话机准备打电话时，电视机会自动静音。

你也许认为这是一个梦境，但 ZigBee 技术的出现及应用足以让这一切梦想成真。韩国第三大移动手持设备制造商 Curitel Communications 公司也已经开始研制世界上第一款 ZigBee 手机，该手机将可通过无线的方式将家中或是办公室内的个人电脑、家用设备和电动开关连接起来。这种手机融入了 ZigBee 技术，能够使手机用户在短距离内操纵电动开关和控制其他电子设备[4]。

1. 应用领域

ZigBee 技术为产品制造商和开发商提供了建立安全可靠、低功耗、具有成本效益(cost-effective)的无线控制产品的能力。这些产品可以满足住宅、商业和工业等领域的应用需求。ZigBee 最初的市场包括能源管理和效率、家庭自动化、楼宇自动化和工业自动化等。经过不断地市场探索，ZigBee 联盟确定了 ZigBee 技术最新的应用领域，如图 1.2 所示[13]。

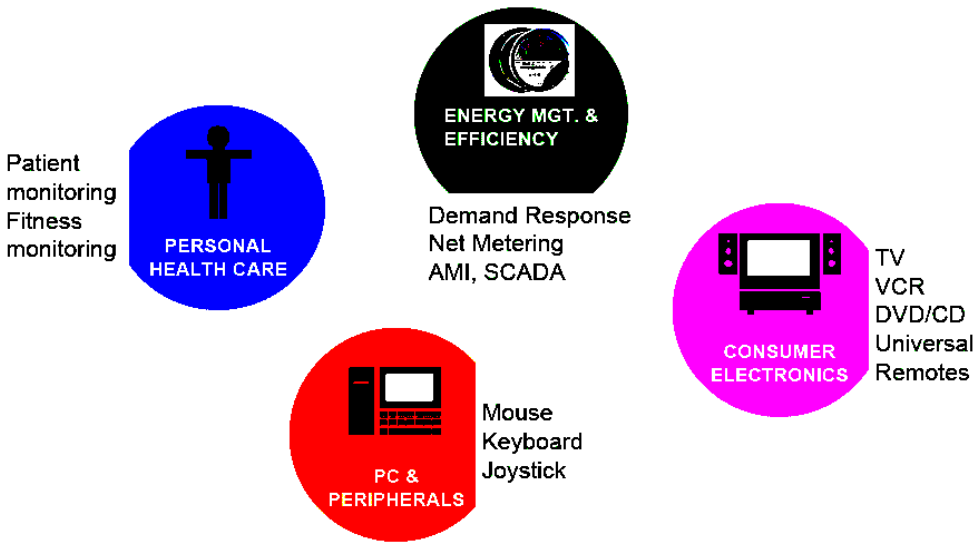


图 1.2 ZigBee 应用领域

从图 1.2 中可以看出，ZigBee 的应用领域目前主要包括 4 个方面：(1) 能源管理和效率：如先进抄表基础设施(Advanced Meter Infrastructure，AMI)、数据采集与监视

控制系统(Supervisory Control And Data Acquisition，SCADA)等；(2) 消费电子产品：如 TV、VCD、DVD/CD 等的通用遥控器；(3) 计算机及外设：如鼠标、键盘、摇杆等；(4) 个人医疗护理：如病人监护、健康监测等。

2. 目标市场

ZigBee 不是用来与蓝牙(Bluetooth)或其它已存在的无线技术标准竞争的，它主要用于现存的其它技术都不能满足的特定应用需求的市场，如节点数量巨大的传感器网络以及自动化应用等。

图 1.3 ZigBee 目标市场

根据 ZigBee 的应用领域，ZigBee 联盟进一步确立了 ZigBee 技术的目标市场[13]，如图 1.3 所示，主要包括楼宇和家庭自动化控制、能量管理、健康监测、电信应用等方面。ZigBee 联盟会针对图 1.3 中的每个具体应用，分别制定相应的公共应用规范(PAP)[14]。PAP 中定义了大家公认的在某个应用中必须遵守的一些使用规范或准则。运行于 ZigBee 设备中的 PAP 记录了如何与一个设备进行通信的具体细节。当不同的厂商实现同一 PAP 应用的产品时，PAP 可以保证产品的互操作性和兼容性，这样就方便了用户对产品的选择，也利于厂商开发通用性的产品，降低生产成本，提供效率和竞争力。

目前 ZigBee 联盟已经制定完成的 PAP 包括[13]：针对能源管理应用的 ZigBee 智能能源 PAP(ZigBee Smart Energy PAP)和针对家庭控制应用的 ZigBee 家庭自动化 PAP(ZigBee Home Automation PAP)。

2009 年 3 月，一个新的基于射频(RF)的消费电子远程控制标准化规范 ZigBee RF4CE PAP(ZigBee Radio Frequency for Consumer Electronics)也已经开始制定[15]，目前仅供联盟成员使用，这将是第一个可对家庭娱乐设备进行创新双向交互与控制的公共规范。

3. 潜在竞争对手

在 ZigBee 联盟确定的应用领域和目标市场中，尚有 Z-Wave、Insteon 与超低功耗

蓝牙(Ultra-Low-Power Bluetooth，ULP Bluetooth)等潜在竞争对手[16]。其中，丹麦公司 Zensys 一手所主导成立的 Z-Wave 联盟[17]，锁定的技术应用即是家庭自动化。

与相关行业和公司自行提出的的专有技术相比，ZigBee 技术是架构在 IEEE 802.15.4 标准上，有国际性标准组织的支持[16]，所以其应用较广泛。截至 2009 年 3 月，ZigBee 联盟已经有近 300 家会员厂商的支持。而且为确保各厂商所开发的 ZigBee 协议栈及应用规范能够实现彼此间互相兼容，ZigBee 联盟还专门制定了 ZigBee 相容平台与 ZigBee 认证产品的测试规范。

1.2 国内外研究现状

1.2.1 ZigBee 联盟

ZigBee 联盟是一个由近 300 家成员企业所组成的非营利性协会，致力于在全球各地推广 ZigBee 技术，使其能成为应用于家用电器、能源、住宅、商业和工业领域的领先无线网络连接、传感和控制标准[13]。

表 1.1 会员等级及费用

会员级别	会员数量	会员费用(美元/年)
促进者	13	50, 000
参与者	145	9, 500
采纳者	139	3, 500

截至 2009 年 3 月，根据 ZigBee 联盟官方网站上的数据显示，ZigBee 联盟目前有 297 家会员，分为促进者(promoter)、参与者(participant)和采纳者(adopter)三个会员等级[13,18]。其中促进者级别的会员有 13 家，包括飞思卡尔(Freescale)、华为、飞利浦、三星、西门子、意法半导体(ST)、施耐德、德州仪器(TI)等。

表 1.1、表 1.2、表 1.3 分别从会员等级及费用、会员所处行业、会员地区分布等方面对 ZigBee 联盟会员进行了归纳。而表 1.4 特别针对亚洲/太平洋地区的主要联盟会员进行了详细统计。表中所有数据均取自 ZigBee 联盟官方网站。需要说明的是，有些 ZigBee 联盟会员(约 30 个)在全球均设有分公司或办事处，并且同时涉足多个行业，所以在数据的统计过程中，同一会员可能在不同的统计条件下出现多次。通过分析表中数据可以得出以下结论：

表 1.2 会员行业分布情况

行业	会员数量
设计公司	65
硬件/模块	67
分销商	8
OEM	130
半导体	28
工具性软/硬件	59
能源供应商	28
ZigBee 协议栈	26

(1) ZigBee 技术在美洲的发展最为壮大，其会员占联盟总会员数的 1/2 强，而亚洲/太平洋地区和欧洲/中东/非洲地区则各占 1/3 左右。

(2) 联盟会员所从事的行业覆盖广泛，目前已经拥有包括芯片制造商、软件开发者、系统集成商、终端制造商以及服务提供商等在内的一条完整的产业链，而且这个

技术联盟还在不断地发展壮大。

(3) 亚洲/太平洋地区的联盟会员主要分布在日本、中国、韩国、印度、澳大利亚等 5 个国家，这几个国家的会员总数之和占该地区会员总数的 70%，其中，日本的会员数量为该地区之最。日本的许多企业十分看好 ZigBee 技术，并于 2005 年成立了日本 ZigBee 特别兴趣小组(ZigBee Special Interest Group Japan)。

(4) 对于中国来说，中国大陆和香港两地区的会员数之和才与台湾地区的会员数相当。台湾也已于 2003 年 10 月成立了台湾 ZigBee 特别兴趣小组。目前，中国大陆地区已经有 5 家企业加入了 ZigBee 联盟，包括：促进者级别的华为技术有限公司；参与者级别的深圳金勃实业有限公司；采纳者级别的曼博科技有限公司、锐拔科技(深圳)有限公司和北京 Inforson 技术有限公司。

当然这些数据是从联盟会员的角度进行统计的，若统计本地区所有从事 ZigBee 的企业，而不仅仅是联盟会员，那么统计结果也许会大不相同。

表 1.3 会员地区分布情况

地区	会员数量
美洲	168
亚洲/太平洋地区	99
欧洲/中东/非洲	108

表 1.4 亚太地区主要会员分布情况

地区	会员数量	地区	会员数量
日本	23	中国大陆	5
韩国	12	中国香港	3
印度	10	中国台湾	8
澳大利亚	7	总计	68

1.2.2 ZigBee 软硬件

1. ZigBee硬件

目前，绝大部分的 ZigBee 硬件都是由国外厂商设计和生产的。

早期的 ZigBee 硬件都是微控制器(Microcontroller Unit，MCU)和 IEEE 802.15.4 射频芯片分离的。随着计算机硬件技术的不断发展，出现了系统级芯片(System-on-a-Chip，SoC)和系统级封装(System-in-a-Package，SiP)[19]，即把多个硬件模块集成到一个单芯片中去。目前，ZigBee 硬件也发展到了这个阶段，在一个芯片内部集成了 MCU 和射频芯片。这不但降低了 ZigBee 开发者对硬件射频电路的要求，加速了 ZigBee 系统的开发，同时也对 ZigBee 系统的稳定性、可靠性、芯片体积等方面带来了积极影响。

目前很多公司都提供 SoC/SiP 封装的 ZigBee 硬件，其中使用者较多的芯片有：Freescale 公司的 MC13211，MC13212 及 MC13213；TI 公司的 CC2430 和 CC2431；Ember 公司的 EM250 和 EM260；Jennic 公司的 JN5121 和 JN5139 等等。

值得注意的是台湾的达盛电子也推出了 ZigBee 射频芯片 UZ2400。台湾工研院资通所也自行研发 ZigBee 平台 ITRI ZBnode[20]。另外，成立于 2007 年的苏州博联科技

有限公司生产的 BeeMote 系列产品硬件也全部兼容 Zigbee 标准[21]。

2. ZigBee软件

在 ZigBee 商业协议栈中，起步较早也最为著名的是 Figure 8 Wireless(F8W)公司所开发的 Z-Stack，后来它被 Chipcon 所并购，再后来 TI 又并购了 Chipcon 从此进入 ZigBee 行业。Freescale 也与印度公司 Mindteck 合作开发了 BeeStack。美商 Helicomm 与 Silicon Labs 研发的 IPLink，Ember 的 EmberZNet，台湾资策会 III 的 ZigBee Advanced Protocol，以及印度公司 Airbee，日商 NEC、OKI、Renesas 等也都推出了自主研发或与其它厂商合作开发的 ZigBee 平台，并且通过了 ZigBee 相容平台的相容性测试[20]。

另外，宁波中科集成电路设计中心(中科院计算所宁波分所)无线通信事业部也对 IEEE 802.15.4 MAC 层的软件开发进行了研究[22]。

也存在一些公司团体或个人开发的免费的 ZigBee 协议栈，如：TI 提供的二进制代码级别的 Z-Stack[23]；Microchip 提供的源代码级别的 MpZBee[24]；美国密西西比州立大学的 Robert B. Reese 教授实现的精简版的 MSSTATE_LRWPAN[25]等。

1.2.3 ZigBee 在中国

2006 年 6 月，华为技术有限公司的会员等级从参与者升级为促进者，并以电信应用规范任务小组(Telecom Application Profile Task Group，TA PTG)成员的身份参与到讨论和制定电信应用规范中去。该规范将提供安全移动支付、信息发布、医疗监控、对等小型数据共享和其他定位服务和功能[26]。

Ember 与中国自动仪表读取系统/方案供应商华立仪表的合作则可看作 ZigBee 在中国推广和应用的里程碑事件。2007 年 7 月，华立仪表与江西省供电公司合作，在鹰潭市启动新型的基于 ZigBee 方式跨台区集中抄表系统创新科研项目。该项目以自动抄表系统(Automatic Meter Reading System，AMRS)为核心平台，配套智能化的通信终端及改进电表，实现了数据自动采集、传输和处理；克服了传统人工抄表模式弊端，轻松解决了广大中小居民用户的电能量采集和分析难题，并于 2008 年 8 月通过了省级验收[27]。

2008 年 12 月，ZigBee 联盟独家授权吉林市曼博科技有限公司在中国成立 ZigBee 产业园区，曼博科技等 9 家公司达成了在吉林高新区共同建设 ZigBee 产业园区的协议[28]。

还有一些中国公司，如上海顺舟、成都无线龙、广州蓝科、上海城优等也进行了 ZigBee 技术的相关研发，但他们主要是与其它 ZigBee 软硬件厂商合作开发一些基于 ZigBee 的应用和解决方案，对 ZigBee 协议栈的实现研究的并不多。

1.3 本文研究内容与意义

1.3.1 本文研究内容

1. ZigBee硬件模块的设计与实现

虽然 ZigBee 芯片集成度的提高一定程度上降低了硬件设计的难度，但由于 ZigBee 硬件涉及到无线射频电路，对 ZigBee 硬件设计人员有较高的要求，尤其是天线模块的设计部分。通过设计并实现一个基于 MC13213 的比较通用的 ZigBee 硬件模块，可方便其它研究人员学习和实践 ZigBee 技术，进一步降低研究 ZigBee 技术的门槛。同时，通用性的硬件设计也能够使其很容易应用到实际项目中。

2. ZigBee协议栈的实现

作为本论文的主体工作，在分析 ZigBee 协议物理层、MAC 层、网络层规范的基础上，针对自制的 ZigBee 硬件模块实现了一个功能有所裁剪的 ZigBee-2006 协议栈，并进行了详细的测试。协议栈基于非信标网络，能同时提供 AODVjr(Ad Hoc On-Demand Distance Vector junior)路由[29-30]和树状层次路由[11]两种路由方式，分别适用于分布式和集中式两种控制需求，具有一定的通用性。

3. ZigBee在设备监控系统中的应用研究

基于自制的 ZigBee 软硬件平台，给出了一个基于 ZigBee 的车间设备监控系统的实例，详细阐述了系统结构、功能和低功耗等内容，可通过高端的数据管理中心软件以图形化的方式监测和控制每个设备。这个实例同时也验证了自制 ZigBee 软硬件平台的可行性和正确性。

1.3.2 研究意义

ZigBee 作为当前短距离无线通信技术的研究热点，有着十分广阔的应用前景，非常值得国内的企业、科研机构以及个人深入学习和研究。但从前面的 ZigBee 联盟会员组成的分析中可以发现，国内研究 ZigBee 技术的氛围不是很好，也相对缺乏系统和深入的学习研究资料，还没有出现比较有影响力或实用的协议栈。国内的大部分企业和科研机构或者在国外厂商的软硬件模块的基础上，进行应用推广和提供解决方案，或者对协议的某一层次进行了比较深入的研究，但对 ZigBee 协议栈本身的具体实现研究的并不多。

虽然国外厂商也提供功能强大的商业协议栈，但一般企业都无法承受其购买或授权费用。另外，也存在一些免费的协议栈，如：MpZBee、MSSTATE_LRWPAN 和 Z-Stack 等。其中，Z-Stack 是以二进制库文件的方式提供给用户使用，不方便用户对 ZigBee

技术的研究和学习；而 MpZBee 和 MSSTATE_LRWPAN 仅实现了 ZigBee 的一个较小的子集，如仅支持树状路由等，并且程序结构不够清晰，代码注释不够详细，不利于用户的进一步学习、研究或改进，并且当进行商业应用时会受到一些版权以及使用费用等方面的限制。

作者所在实验室从 ZigBee-2004 规范开始便对 ZigBee 技术进行了跟踪研究，也做了一些实际的应用方案，目前也有正在进行中的相关应用项目，已经积累了一定的技术和应用基础，但 ZigBee 技术本身也在不断地发展。本文一方面是从实验室的技术跟踪研究和实际应用的角度出发，继续跟进最新的 ZigBee 技术；另一方面也期望能够为其他 ZigBee 技术研究者提供一个便于学习和研究的软硬件平台，以为促进国内 ZigBee 技术的蓬勃发展贡献微薄之力。

1.4 论文结构

本文共分为七章，论文结构及各章主要内容如下：

第一章为绪论，给出了本课题的研究背景以及本文的主要研究内容和意义。

第二章为相关技术分析，给出了 IEEE 802.15.4 标准、ZigBee 协议规范以及扩频通信技术的基本知识，总结了几个易混淆的概念。

第三章为 ZigBee 物理层协议的实现，给出了 ZigBee 硬件和实现物理层协议所需要的基本知识点，详细设计与实现了 ZigBee 硬件，在自制硬件平台上实现了 ZigBee 物理层协议并进行了测试。

第四章为 ZigBee MAC 层协议的实现，概述了 MAC 层的基本功能和概念以及 MAC 帧结构定义，并在物理层协议的基础上实现了基于非信标网络的 ZigBee MAC 层协议。

第五章为 ZigBee 网络层协议的实现，概述了网络层的基本功能和网络层帧结构组织方式以及常用的路由协议，在 MAC 层协议的基础上实现了 ZigBee 网络层协议。

第六章阐述了一个基于 ZigBee 的车间设备监控系统的应用实例，验证了软硬件平台的可行性和正确性，深入分析了节点低功耗等内容。

第七章对全文进行总结，并就进一步研究的问题进行讨论。

第二章 相关技术分析

本章给出了服务原语的概念和类型，概括了 IEEE 802.15.4 标准的内容和特点，总结了 ZigBee 协议栈各层的功能以及各版本间的差异，并给出了扩频通信的基础知识，最后总结了几个容易混淆的概念。

2.1 IEEE 802.15.4 标准

2.1.1 服务原语简介

1. 原语概念

服务原语(service primitives)是指用户和协议实体间的接口。对于第 N 层用户来说，他可以通过服务原语来调用第 N 层协议实体所提供的一些服务，而调用过程中第 N 层协议实体也会调用服务原语给第 N 层用户返回一些状态信息。

2. 原语类型

通常原语可以分为 4 种类型[2]：(1) request：第 N 层用户调用 request 原语向第 N 层协议实体请求一个服务；(2) indication：第 N 层协议实体调用 indication 原语通知第 N 层用户在第 N 层协议实体内部发生的一些事件；(3) response：第 N 层用户调用 response 原语通知对第 N 层协议实体之前所调用的 indication 原语的执行结果；(4) confirm：第 N 层协议实体调用 confirm 原语通知第 N 层用户之前所调用一个或多个服务请求的执行结果。

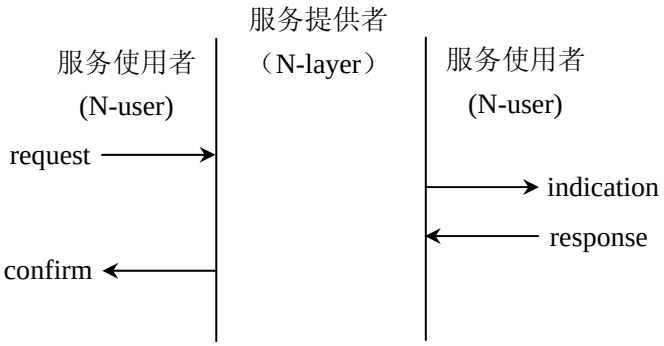


图 2.1 服务原语

这 4 种原语的形象化表示如图 2.1 所示。图中左右两边的 N-user 是两个处于对等关系的不同的 N-user，并且都位于 N-layer 对等协议实体之上。

IEEE 802.15.4 标准和 ZigBee 协议规范已经把协议各层应该实现的功能以服务原语的形式标准化了，所以实现 ZigBee 协议栈的大部分工作就是实现协议标准中的各

个原语。原语的实现过程中还会需要许多功能性函数的配合，所以实现协议栈的工作量还是很大的。

2.1.2 标准概述及特点

1. 概述

IEEE 802.15.4 工作组致力于制定低速率无线个人区域网络(LR-WPAN)标准。该标准把低能量消耗、低速率传输、低成本作为重点目标，旨在为个人或者家庭范围内不同设备之间的短距离低速互连提供统一标准。

IEEE 802.15.4 的低速率、低功耗和短距离传输等特点使它非常适宜支持简单器件。在 IEEE 802.15.4 中定义了 13 个物理层服务原语和 35 个 MAC 层服务原语，总共为 48 个，仅为蓝牙的 1/3[2,31]。这使它非常适用于存储能力和计算能力有限的简单器件。

IEEE 802.15.4 中定义了两种器件[2]：具有完整功能的全功能设备(Full Functional Device，FFD)和只具有部分功能的简化功能设备(Reduced Function Device，RFD)。对于 FFD，其资源比较丰富，功能也复杂，要求它支持所有的 48 个服务原语。而对于 RFD，由于其占用资源较少，应用相对简单，需要的存储容量也小，所以在最小配置时只要求它支持 37 个服务原语，以降低系统成本。

2. 特点

IEEE 802.15.4 标准所定义的 LR-WPAN 具有如下特点[2]：在不同的载波频率下实现了 20kbps、40kbps 和 250kbps 三种不同的传输速率；支持星型、点对点网络拓扑结构；支持 16 位和 64 位两种地址类型；支持带有冲突避免的载波监听多路访问(Carrier Sense Multiple Access with Collision Avoidance，CSMA-CA)信道访问算法；支持完整的确认机制，保证传输可靠性；低能量消耗。

2.2 ZigBee 协议规范

2.2.1 协议体系结构

ZigBee 技术具有统一的技术标准，主要由 IEEE 802.15.4 工作组与 ZigBee 联盟分别制定。其中，IEEE 802.15.4 工作组负责制定物理(Physical，PHY)层和媒体访问控制(Medium Access Control，MAC)层的标准，而 ZigBee 联盟则制定高层的网络(Network，NWK)层、应用(Application，APL)层和安全服务提供者(Security Services Provider，SSP)等标准。ZigBee 技术的整体协议架构如图 2.2 所示[11]。

PHY 层主要负责数据的调制与解调、发送和接收，向下直接操作物理传输介质(无

线射频)，向上为 MAC 层提供服务。

MAC 层负责为一个节点和它的直接邻居之间提供可靠的通信链路，提供冲突避免以提高通信效率，负责组装和分解 MAC 层帧。

NWK 层决定设备连接和断开网络时所采用的机制；执行设备间的路由发现和路由维护；完成一跳(one-hop)范围内的邻居设备的发现和相关信息的存储；创建新网络；为新入网设备分配网络地址等。

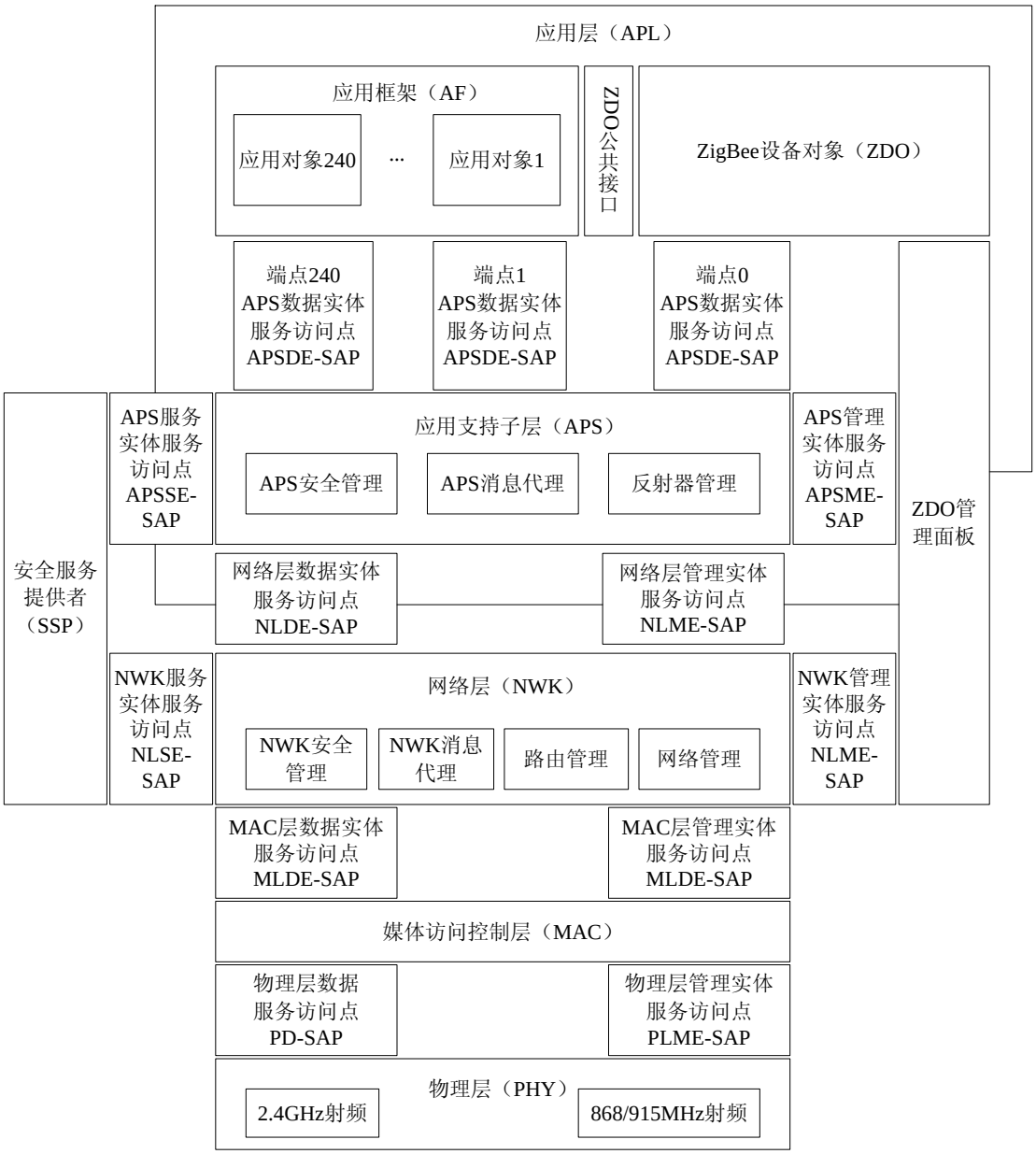


图 2.2 ZigBee 协议架构

APL 层由应用支持子层(Application Support Sub-layer，APS)、应用框架(Application Framework，AF)、ZigBee 设备对象(ZigBee Device Object，ZDO)组成。APS 用来维护绑定表以及在绑定设备间传送消息(所谓绑定就是基于两台设备的服务

和需求将它们匹配地连接起来)。AF 提供了如何建立一个规范的描述，以确保规范的前后一致性，它也规定了规范的一系列标准数据类型。ZDO 定义了一个设备在网络中的角色(协调者、路由器或终端设备)，可以发起和响应绑定请求，并可以在网络设备之间建立安全机制。

SSP 为使用加密的层(如 NWK 和 APS 等)提供安全机制。

2.2.2 各协议版本比较

目前存在 3 种公开的协议版本：ZigBee-2004、ZigBee-2006、ZigBee-2007(又称 ZigBee PRO)。各协议版本之间的比较[20,32-33]如表 2.1 所示。

表 2.1 ZigBee 协议各版本之间的比较

协议版本	ZigBee-2004	ZigBee-2006	ZigBee PRO
公开时间	2005 年 6 月	2007 年 1 月	2008 年 1 月
特征	8 位簇标识； 键值对(Key Value Pair，KVP)/MSG(Message)通信格式； 协调者绑定； 提供多种安全模式； 数据包≤128 字节； 分布式地址分配机制和树状路由； 容错路由算法；	16 位簇标识； 可选的协调者绑定； 移除了： KVP 通信格式； 增加了： ZCL 库、 多播、 群组设备、 无线配置；	在 2006 版本基础上， 增加了： 频率跳变、 分割/重组、 随机地址分配、 多对一/源节点路由、 安全层的一些改进； 移除了： 分布式地址分配机制和树状路由；
兼容性	原始 ZigBee 版本	不需与 ZigBee-2004 兼容	与 ZigBee-2006 兼容； 不需与 ZigBee-2004 兼容

对于 ZigBee 协议规范的演进，并非原来的设计有严重瑕疵，迫使 ZigBee 联盟改良规范。相反的，正是由于 ZigBee 应用越来越广泛，使原来的设计需针对不同的应用进行调整。例如，2004 版本主要用于单纯的家庭灯光控制，而 2006 以及 Pro 则扩展至楼宇自动化、工厂系统监控及家庭自动化。也正因为这些功能的调整，使厂商对于版本间的兼容性问题产生疑虑，也因此造成 ZigBee 联盟会员的反弹意见，目前正在寻求合理的解决方式[34]。

考虑到 ZigBee PRO 特性比较复杂，且某些新增功能如频率跳变、分割/重组等可在协议栈之上的应用程序中实现；而其它的一些功能如随机地址分配、高安全模式等是以网络通信开销增加和通信延迟变长为代价的。而 ZigBee 2006 版本应用最为广泛，并且在某些需要集中式控制和节点很少移动或几乎不移动的应用中，使用树状路由会比较合适。因为树状路由不需要路由发现和路由维护的过程，减少了控制包的发送，而在无线通信环境中，发送次数的降低意味着节点功耗的降低，也意味着整个网络的

生命周期的延长。所以在地址分配策略中没有采取随机地址分配策略，而仍采用分布式地址分配机制。这样可以采用 AODVjr 路由和树状路由相结合的路由策略，以适应多种应用环境。但这种地址分配方式要求用户对最终的网络事先进行规划，需要有一定的可预见性，在协议栈以后的扩展中可以考虑使用 ZigBee PRO 中的随机地址分配策略和路由策略。

2.3 扩频通信技术

2.3.1 基本概念和特点

扩频通信，即扩展频谱通信(spread spectrum communication)，是将待传送的信息数据用伪随机编码(又称扩频序列)调制，实现频谱扩展后再传输，接收端则采用相同的编码进行解调及相关处理，恢复原始信息数据。

这种通信方式与常规的窄带通信方式是有区别的：一是信息的频谱被扩展后形成宽带传输；二是经相关处理后能恢复成原始窄带信息数据。

正是由于这两大持点，使扩频通信有如下的优点：抗干扰、抗噪音、抗多径衰落；具有保密性；功率谱密度低，具有隐蔽性和低截获概率；可多址复用和任意选址；可用于精确定时和测距等[35-36]。

2.3.2 理论基础

扩频通信的可行性，是从信息论和抗干扰理论的基本公式中引申而来的。信息论中关于信息容量的香农(Shannon)公式为[35]：

$$C = B \cdot \log_2(1 + S/N) \tag{2.1}$$

其中，C 是信道容量(用传输速率度量)；B 是信号频带宽度；S 是信号功率；N 是加性噪声功率。

式(2.1)表明，在给定传输速率 C 不变的条件下，频带宽度 B 和信噪比 S/N 是可以互换的。即可通过增加频带宽度的方法，在较低的信噪比下传输信息。扩展频谱以换取信噪比要求的降低，正是扩频通信的重要特点，并由此为扩频通信的应用奠定了基础。

2.3.3 工作原理

扩频通信的一般工作原理[35-36]如图 2.3 所示。

在发送端，输入的信息先经过信息调制形成数字信号，然后由扩频码发生器产生的扩频码序列去调制数字信号以展宽信号的频谱。展宽后的信号再调制到射频中发送出去。

在接收端，会把收到的宽带射频信号先变频至中频，然后使用本地产生的与发送端相同的扩频码序列进行扩频解调，再经信息解调，恢复成原始信息输出。

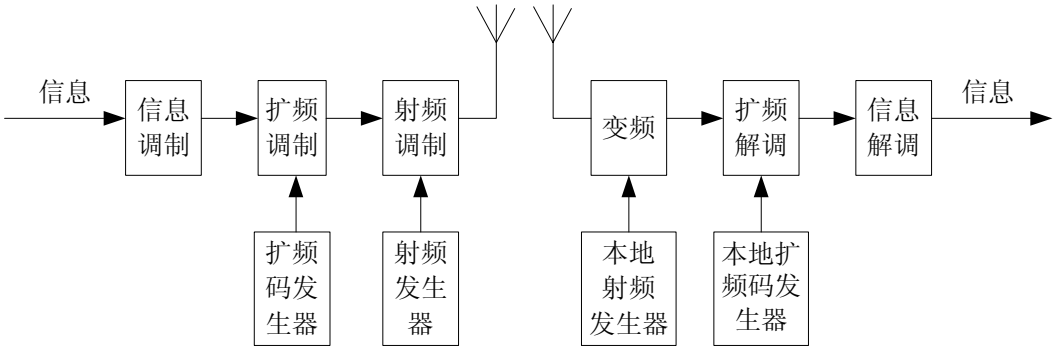


图 2.3 扩频通信的一般原理

与一般通信系统比较，扩频通信就是多了扩频调制和扩频解调部分。根据扩展频谱方式的不同，扩频通信系统可分为直接序列扩频(Direct Sequence Spread Spectrum，DSSS)、跳频扩频(Frequency Hopping Spread Spectrum，FHSS)、跳时扩频(Time Hopping Spread Spectrum，THSS)以及混合扩频等。

2.4 相关概念澄清

2.4.1 IEEE 802.15.4 与 ZigBee

从 ZigBee 的发展历史可以看到，ZigBee 与 IEEE 802.15.4 有着密切的关系，事实上 ZigBee 的底层技术就是基于 IEEE 802.15.4 的，因此有一种说法认为 ZigBee 和 IEEE 802.15.4 是同一个东西，或者说 ZigBee 只是 IEEE 802.15.4 的名字而已，其实这是一种误解。

实际上 ZigBee 和 IEEE 802.15.4 的关系，有点类似于 WiMAX 和 IEEE 802.16，Wi-Fi 和 IEEE 802.11，Bluetooth 和 IEEE 802.15.1。ZigBee 可以看作是一个商标，也可以看作是一种技术，当把它看作一种技术的时候，它表示一种高层的技术，而其物理层和 MAC 层直接引用 IEEE 802.15.4。事物是不断的发展变化的，尤其是通信技术，可以想象将来的 ZigBee 可能不会使用 IEEE 802.15.4 定义的底层，就像 Bluetooth 宣布下一代底层采用 UWB 技术一样，但是“ZigBee”这个商标以及高层的技术还会继续保留[1,20]。

2.4.2 设备类型

在 ZigBee 的 MAC 层，IEEE 802.15.4 标准定义了 2 种类型的设备[2]：全功能设备(FFD)和简化功能设备(RFD)。在 ZigBee 的网络层，ZigBee 协议规范定义了 3 种类型的设备[11]：ZigBee 协调者(ZigBee Coordinator，ZC)、ZigBee 路由器(ZigBee Router，

ZR)和 ZigBee 终端设备(ZigBee End Device，ZED)。其实这两种定义只是从 ZigBee 协议的不同层次出发对网络中设备的不同区分而已。

在 ZigBee 网络中，ZC 与 ZR 功能一致，但 ZC 与 ZR 在网络中的角色不同，本质上 ZC 是在网络组建过程中扮作一个领导者角色的 ZR。每个网络有且仅有一个 ZC。ZR 有能力转发数据包和参与网络组建。ZED 仅可以发送和接收为它自己本身所准备的数据包。一般地，ZC 和 ZR 均为 FFD，而 ZED 可以为 RFD。

2.4.3 协调者和 PAN 协调者

IEEE 802.15.4 把具有数据转发能力的 FFD 定义协调者(coordinator)。若一个协调者是个域网(Personal Area Network，PAN)内最主要的控制器，则称之为 PAN 协调者(PAN coordinator)。在每个 PAN 中，只能有一个 PAN 协调者[2]，主要负责该 PAN 的创建和维护等工作。

2.5 本章小结

本章给出了服务原语的概念和类型，简述了无线个域网标准 IEEE 802.15.4 的主要内容和特点。回顾了 ZigBee 技术的主要内容，给出了 ZigBee 协议栈的体系结构并归纳了协议各层的主要功能，详细对比分析了目前存在的 ZigBee 协议各版本。对扩频通信技术的基本概念和原理进行了必要的概述。最后针对 IEEE 802.15.4 标准和 ZigBee 协议规范中容易混淆或引起误解的概念进行了总结。

第三章 ZigBee 物理层的研究与实现

物理层是本文的关键工作之一。物理层通过操作底层硬件为上层提供服务接口，物理层的稳定可靠关系到整个协议栈的健壮性。本章阐述了物理层的基本内容，设计和实现了 ZigBee 硬件平台以及物理层协议并进行了详细的测试。在实现过程中从硬件和软件两个方面保证了物理层功能的可靠性。

3.1 ZigBee 物理层概述

3.1.1 工作频段和数据传输速率

ZigBee 工作在免申请的工业科学医疗(Industrial Scientific & Medical，ISM)频段。IEEE 802.15.4 标准中定义了两个可用的物理层：基于 2.4GHz 频段的“短距离(short-distance)”实现和基于 868/915MHz 频段的“长距离(long-distance)”实现，二者都使用 DSSS 扩频技术。ZigBee 的工作频段和数据传输速率如表 3.1 所示。

表 3.1 工作频段和数据传输速率

工作频段 (MHz)	适用地区	扩展参数		数据参数		
		码片速率 (kchip/s)	调制方式	比特速率 (kb/s)	符号速率 (ksymbol/s)	符号 (symbol)
868～868.6	欧洲	300	BPSK	20	20	二进制
902～928	北美	600	BPSK	40	40	二进制
2400～2483.5	全球	2 000	O-QPSK	250	62.5	16 相正交

对于固定设备区域市场，如户外测量网络(outdoor metering networks)，IEEE 802.15.4 工作组发布了两个新特色的可用物理层：适用于中国的 780MHz 频段 802.15.4c 和适用于日本的 950MHz RFID 频段 802.15.4d。这两个新的“长距离”的 ZigBee 频段将有利于扩展 ZigBee 在户外的应用[7]。

3.1.2 信道分配和编号

IEEE 802.15.4-2003 规范定义了 27 个物理信道，信道编号从 0 到 26，每个信道对应着一个中心频率，这 27 个物理信道覆盖了表 3.1 中的 3 个频段。其中，868MHz 频段定义了 1 个信道(信道 0)；915MHz 频段定义了 10 个信道(信道 1～10)；2.4GHz 频段定义了 16 个信道(信道 11～26)。这些信道的中心频率定义如下：

f_c = 868.3MHz　　当 k=0 时，

f_c = 906 + 2(k-1)MHz　　当 k=1，2，…，10 时，

f_c = 2405 + 5(k-11)MHz　　当 k=11，12，…，26 时。

其中，k 为信道编号，f_c 为信道中心频率。信道分布状况如图 3.1 所示。

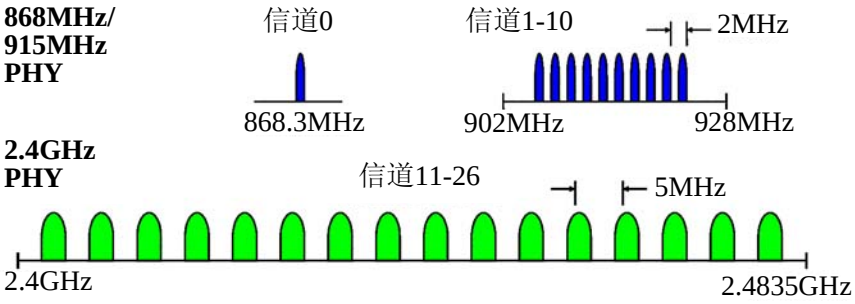


图 3.1 信道分布状况

通常 ZigBee 硬件设备不能同时兼容这 3 个频段，因此应根据当地无线电管理委员会的规定来选择 ZigBee 设备。中国目前的 ZigBee 工作频段为 2.4GHz[37]。

3.1.3 物理层模型和服务原语

1. 物理层模型

ZigBee 物理层通过射频固件(firmware)和射频硬件(hardware)为 MAC 层和物理无线信道之间提供了称作服务接入点(Service Access Point，SAP)的接口。

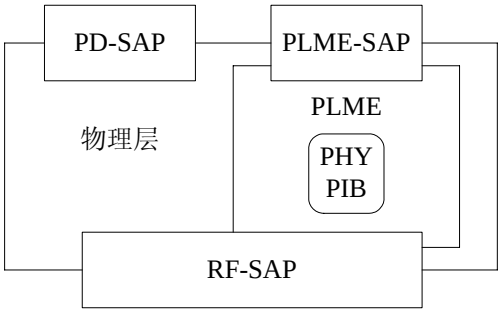


图 3.2 物理层参考模型

IEEE 802.15.4 定义的物理层参考模型如图 3.2 所示。其中 PD-SAP(PHY Data Service Access Point)是物理层提供给 MAC 层的数据服务接口；PLME-SAP(Physical Layer Management Entity-Service Access Point)是物理层提供给 MAC 层的管理服务接口；RF-SAP 是由底层无线射频驱动程序提供给物理层的接口。

2. 物理层原语

物理层主要完成以下工作：激活和禁用射频收发器；对信道进行能量检测(Energy Detect，ED)；提供所接收数据包的链路质量指示(Link Quality Indication，LQI)；空闲信道评估(Clear Channel Assessment，CCA)；信道频率选择；数据发送和接收等。关于物理层原语的详细内容请参考 IEEE 802.15.4 标准。

3.2 ZigBee 硬件设计

3.2.1 ZigBee 硬件选型

在嵌入式产品设计中，硬件选型是一个重要环节，它直接影响着产品的性能，以及产品的后续开发过程。为了选择满足系统功能需求的硬件，需要考虑到芯片内部集

成的功能模块、通用的 I/O 引脚数目、RAM 及 FLASH 大小、芯片以及开发环境的熟悉程度、芯片的可购买性以及性价比等等。

生产 ZigBee 芯片的公司中，最著名的莫过于 TI 和 Freescale 两家公司。这两家公司的 ZigBee 产品线非常丰富，不管是早期的“MCU+射频芯片”的组合，如 TI 的“MSP430+CC2420/CC2520”，Freescale 的“MC9S08GT60+MC13191/MC13192”等；还是目前主流的 SoC 或 SiP 的单芯片解决方案，如 TI 的 CC2430、CC2431，Freescale 的 MC1321x、MC1322x 等；这两家公司都为用户提供了多种选择。考虑到系统的稳定性、可靠性、芯片体积等方面因素，作者决定在这两家公司的产品中选择出一款单芯片的 ZigBee 解决方案。

CC2430 是 TI 的第二代 ZigBee 平台和真正的 SoC 解决方案[38]。它兼容 IEEE 802.15.4 标准，在单芯片上集成了 ZigBee 射频前端、存储器(8KB 的 RAM 和 32/64/128KB 可选容量的 FLASH)和微控制器(8051 核)。另外，CC2430 还包含了 AES-128 协处理器、看门狗、掉电检测电路以及 21 个通用 I/O 引脚。CC2431 与 CC2430 基本相同，只是增加了高精度无线定位跟踪引擎模块。

Freescale 公司针对 ZigBee 技术也推出了完整的单芯片解决方案，其主要代表是第二代的 MC1321x 系列产品[39]。MC1321x 内部集成了 HCS08 MCU 和遵循 IEEE 802.15.4 标准的第二代无线射频收发器 MC1320x。用户可以根据具体系统对 I/O 引脚数量和存储器大小(RAM/FLASH)的不同需求，选择 MC13211(1KB/16KB)、MC13212(2KB/32KB)、MC13213(4KB/60KB)等产品。MC1322x 是 Freescale 最新的第三代 ZigBee 封装内平台(Platform in a Package，PiP)产品[40]，其内部集成了完整的低功耗 2.4GHz 收发器和基于 32 位 ARM7 核的 MCU，以及用于 IEEE 802.15.4 MAC 和 AES 安全加密的硬件加速器，是高密度低元件数的 ZigBee 解决方案。

上述两家公司提供的 ZigBee 解决方案相差无几，本文主要从对芯片及其开发环境的熟悉程度等方面进行选择。在研究生阶段的学习过程中，作者对 Freescale 的 8/16/32 位系列 MCU 做了大量研究，熟悉其指令系统、编程结构和工作性能，实验室也有比较完备的开发环境、程序写入工具等条件。另外根据功能最合适原则[41]，最终选定了 Freescale 的 MC13213 作为 ZigBee 硬件平台。

3.2.2 MC13213 芯片简介

1. MC13213概述

MC13213 采用 SiP 技术在 9×9mm 的 LGA 封装内集成了 MC9S08GT 主控 MCU 和 MC1320x 射频收发器。后文中将用 MCU 和 Modem 分别代表 MC13213 中的主控 MCU 模块和射频收发器模块。

MC13213 拥有 4KB 的 RAM、60KB 的 FLASH，具有 1 个串行外设接口(Serial Peripheral Interface，SPI)，2 个异步串行通信接口(Serial Communications Interface，SCI)，1 个键盘中断模块(Keyboard Interrupt，KBI)，2 个定时器/脉宽调制模块(Timer/PWM，TPM)，1 个 8 通道 10 位的模数转换器(Analog/Digital Converter，ADC)，以及多达 32 个的 GPIO 口等[42]。MC13213 芯片的功能框图如图 3.3 所示，关于这款芯片的详细内容请参考 MC13213 的芯片手册。

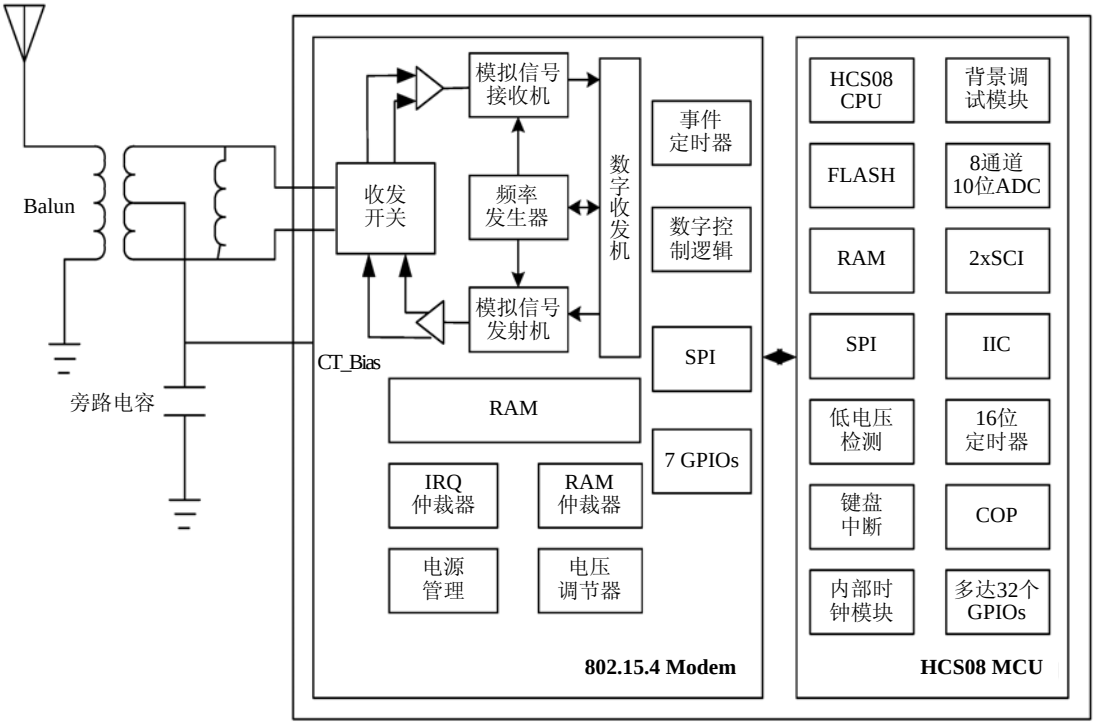


图 3.3 MC13213 的功能框图

2. Modem与MCU的交互方式

Modem 可以通过 SPI 接口、IRQ 中断请求以及几个状态和控制信号与主控 MCU 实现交互，如图 3.4 所示。在 MC13213 的 SiP 内部已互连了这些必需的连接，图中的箭头指示了状态或控制信号的接收方。

1) SPI 命令通道

SPI 命令通道(SPI command channel)是 Modem 与 MCU 之间的主要交互方式，使用标准的 4 线 SPI 进行通信。MCU 通过 SPI 命令结构可以读/写 Modem 的寄存器内容、设置 Modem 的初始化参数、读取 Modem 的状态和控制信息。

在 MCU 和 Modem 的 SPI 通信过程中，必须满足 Modem 的一些特殊要求：(1) MCU 必须作为 SPI 主机，而 Modem 作为 SPI 从机；(2) SPI 通信的时钟频率最大不能超过 8MHz；(3) 先传送最高有效位(Most Significant Bit，MSB)；(4) SPICLK 时钟格式必须配置为：时钟相位 CPHA=0，即在 SPICLK 的第一个沿跳变时就开始捕捉数

据；时钟极性 CPOL=0，即 SPICLK 在空闲时为低电平。

在 SPI 通信过程中，所有的 SPI 传输都是由主机发起的。虽然 SPI 通信是双向的数据传输过程，但 Modem 的 SPI 协议规定了每次的数据传输只在一个方向(主机到从机或从机到主机)上进行，具体细节请参考后面关于 SPI 事务的介绍。

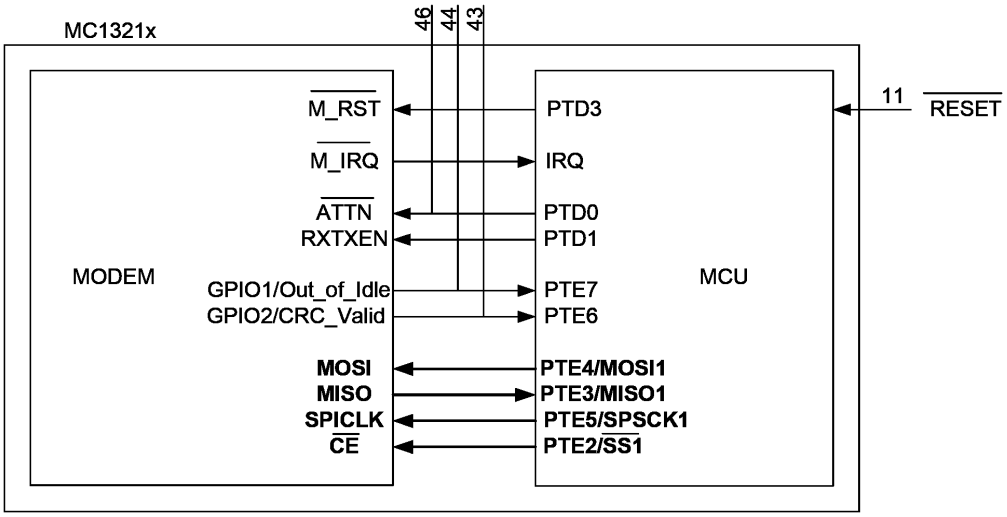


图 3.4 MC13213 的内部功能性互连

2) IRQ 中断请求

IRQ 中断为 Modem 提供了一种通知 MCU 有关 Modem 内部所发生事件的方法，这样就免除了 MCU 一直轮询 Modem，降低了 MCU 的运行开销。

3) Modem 控制信号

为使 Modem 能在 MCU 的控制下正确地工作，MCU 需要用两个输出引脚来提供控制信号，如图 3.4 中的 PTD0、PTD1。其中，$\overline{\text{ATTN}}$ 用来把 Modem 从低功耗模式唤醒；RXTXEN 用来允许 Modem 的发送、接收和 CCA 等操作。

4) Modem 状态信号

除了可以使用 SPI 事务来读取 Modem 的状态信息外，Modem 还为 MCU 提供了两个状态指示引脚，如图 3.4 中的 GPIO1 和 GPIO2。其中，GPIO1 引脚反映了 Modem 收发机是否忙；GPIO2 引脚可以反映所接收数据包的循环冗余校验(Cyclical Redundancy Check，CRC)是否有效或者反映 CCA 的结果。

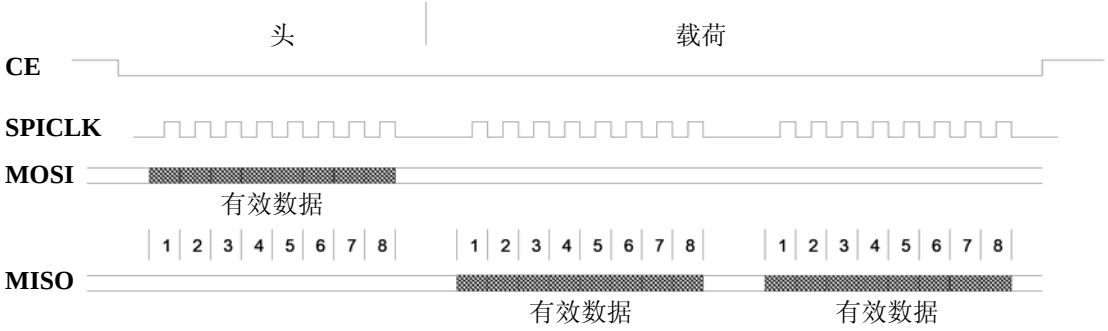
3. Modem的SPI事务操作

Modem 在标准 SPI 协议基础上实现的一个称作 SPI 事务的扩展 SPI 协议。由于 Modem 中的寄存器和 RAM 大小都配置为 16 位即一个字(word)宽度，所以它规定了每次 SPI 事务过程必须持续(1+2×N)个字节长度，每个字节对应一个 SPI 脉冲(SPI burst)，其中 1≤N≤64，且为整数，代表每个 SPI 事务中所包含的字(word)数。特别

地，当 N=1 时，称为 SPI 单次事务(SPI singular transaction)；其它情况称为 SPI 循环事务(SPI recursive transaction)。

在每个 SPI 事务中，由 1 字节的头(header)和 2×N 字节的载荷(payload)组成。其中，header 的最高位为 R/$\overline{W}$ 位，表示操作类型是读还是写；header 的低 6 位是寄存器地址，表示了 SPI 操作的 64 个可能的寄存器地址(注意，有一部分寄存器没有实现)；后续的 N 个字是 SPI 事务的数据部分。注意，R/$\overline{W}$ 是从 MCU 的视角衡量的，当 R/$\overline{W}$ =1 时，表示是读操作；当 R/$\overline{W}$ =0 时，表示是写操作。

每个 SPI 事务由$\overline{CE}$引脚来界定。事务开始时，需拉低$\overline{CE}$引脚来表示本次 SPI 事务开始；事务结束时，需拉高$\overline{CE}$引脚来表示本次 SPI 事务结束。在$\overline{CE}$引脚被拉低以后，SPICLK 必须至少提供 24 个上升沿(当是 SPI 单次事务时)，否则本次 SPI 事务不会改变寄存器的内容。SPI 单次读写事务的时序图如图 3.5 所示。



(a) SPI 单次读

(b) SPI 单次写

图 3.5 SPI 单次读写事务的时序图

4. Modem的数据传输模式

Modem 定义了两种数据传输模式：Stream 模式和 Packet 模式。在 Stream 模式中，数据的发送和接收是逐字(word-by-word)处理的。而在 Packet 模式中，发送时，发送方先将待发送数据缓存在 Modem 的发送缓冲区(TX RAM)中，然后再发送；接收时，接收方先在接收缓冲区(RX RAM)中缓存收到的整个数据包，然后再通知 MCU 来读取。

虽然 Packet 模式下数据的接收有稍许延迟，但其降低了对 MCU 的资源要求，在本协议栈实现过程中使用这种数据传输模式。

3.2.3 ZigBee 硬件设计与实现

MC13213 的硬件设计包含两部分：主控 MCU 和无线射频模块。下面将分 3 个部分来阐述 ZigBee 硬件平台的设计与实现。需要说明的是，最终实现的 ZigBee 硬件的对外接口将主要是使用嵌入式系统中最普遍的 RS232 串行通信接口，以后可能会根据应用需求提供其它接口。

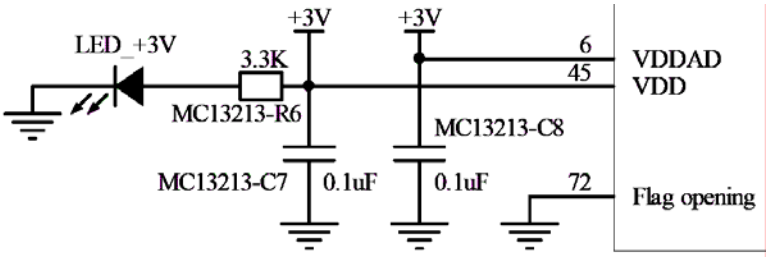
硬件设计参照实验室的硬件构件规范[41]进行，以保证硬件设计的模块化和可重用性。关于硬件构件的基本规则在天线模块中加以详述，其它模块也都遵循这些基本规则。

1. 主芯片模块

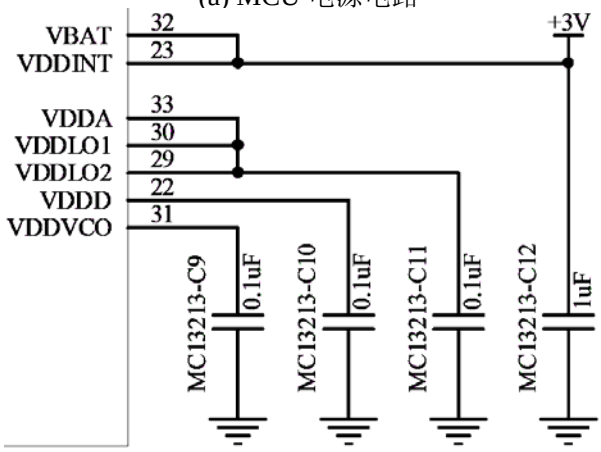
所谓主芯片模块即是 MCU 最小系统。MCU 最小系统是指可使 MCU 内部程序正常运行所必需的外围电路，主要包括系统电源与滤波电路、复位电路、晶振电路、调试写入接口等[43]。

MC13213 的最小系统主要包括电源模块、晶振模块、BDM 调试模块、复位模块等。下面将分别简述各个模块的设计及注意事项。

1) 电源模块



(a) MCU 电源电路



(b) Modem 电源电路

图 3.6 电源模块

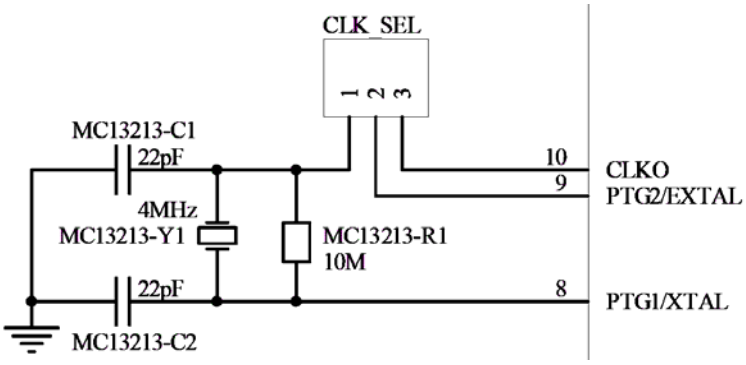
ZigBee 设备一般使用电池供电。根据 MC13213 芯片手册得知，其工作电压范围

为 2V～3.4V，日常生活中常用的 3V 纽扣电池完全可以满足 MC13213 的工作电压要求。电源模块的设计和实现时就直接使用纽扣电池作为电源输入，只需特别注意下相应电源引脚的滤波即可。这样不仅免除了使用额外的电压转换电路的麻烦，也降低了系统的总体成本。

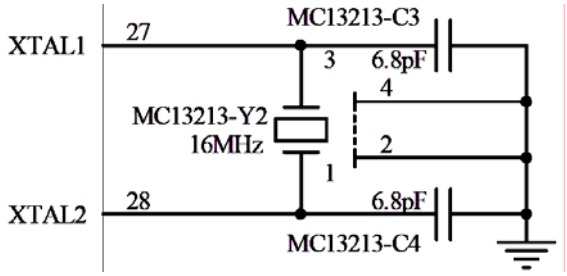
电源模块如图 3.6 所示，包括 MCU 和 Modem 两部分，图中的电容主要起滤波作用。另外在图(a)中设计了一个电源指示灯，但在最终的实际产品中，为最大程度地降低功耗，延长设备的使用时间，应尽量少用或不用指示灯(耗电量较大)，即使确实需要指示灯，也仅在需要时才点亮。

2) 晶振模块

晶振模块也包括 MCU 和 Modem 两个部分，如图 3.7 所示。其中 Modem 的晶振是必须的，且其对晶振精度也有特殊要求，根据 IEEE 802.15.4 的标准，必须使用 16MHz 的晶振，且其精度必须≥±40ppm(part per million)[2]。对于 MCU 的晶振电路则可以选择外部晶振、外部时钟源或内部晶振，在设计中为用户提供了多种选择。



(a) MCU 晶振电路



(b) Modem 晶振电路

图 3.7 晶振模块

使用外部晶振是一种高成本高精度的 MCU 时钟供给方案。

一般地，从降低系统成本因素考虑，会使用 Modem 的 CLKO 时钟输出作为 MCU 的外部时钟源，这是一种低成本高精度的 MCU 时钟供给方案。若采用这种方案，则在 Modem 初始化之前和 Modem 进入低功耗之后，MCU 需使用自时钟模式(Self-Clocked Mode，SCM)，这需要通过编程来实现时钟的切换。

另外，MC13213 的内部集成了一个大约 243KHz 的参考时钟，但其精度不高，未进行校准(trim)前只保证±25%的精度，但经过校准之后可达到±0.5%的精度[42]，完全可以满足 MCU 对时钟精度的要求。这也可以作为一种低成本较高精度的 MCU 时钟供给方案。考虑到成本以及程序结构的清晰性，在协议栈的实现过程中 MCU 采用这种时钟供给方案。在最终产品中，为保证时钟的高精度，可选择使用 Modem 的 CLKO 时钟输出作为 MCU 的外部时钟源。

3) BDM 调试写入接口及复位模块

BDM 调试写入接口及复位模块如图 3.8 所示。Freescale 的 HCS08 系列 MCU 都有专门的调试及写入接口，即 4 线的后台调试模块(Background Debug Module，BDM)，它仅使用了后台调试引脚(BKGD)、复位引脚(RST)、地以及电源 4 个引脚。为了与实验室的写入器接口兼容，本版硬件中的 BDM 接口使用 6 脚插针来实现。但是，考虑到 ZigBee 模块在很多应用环境下，都要求有很小的体积，这样一来，6 脚插针的 BDM 接口就显得臃肿好多。因此，将来的硬件设计中可考虑使用体积更小的接口(如 miniUSB 接口)来实现。

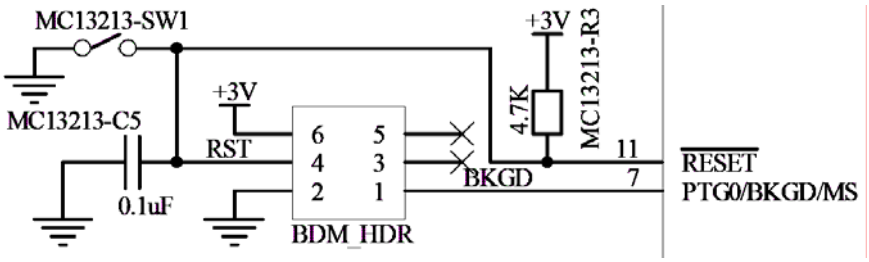


图 3.8 BDM 调试写入接口及复位模块

复位模块的电路比较简单，只需通过 4.7K 欧姆的电阻上拉到+3V，再利用一个小电容维持复位电平一定时间，以使 MCU 复位，同时也可以滤除一些杂波的干扰，防止 MCU 被错误复位。

2. 天线模块

天线模块设计的好坏关系到无线信号的收发质量，对于短距离无线通信设备来说，辐射模型、增益、阻抗匹配、带宽、尺寸和成本等因素，都会影响天线的选择和设计。由于 Modem 内部已经集成了功率放大器(Power Amplifier，PA)、低噪声放大器(Low Noise Amplifier，LNA)和收/发开关(T/R switch)，这在很大程度上降低了系统成本和射频电路的设计难度。

常用的天线有 PCB 天线、Chip 天线和 Whip 天线等几种[5]，具体形状如图 3.9 所示。PCB 天线一般有偶极子、平面倒 F 等形状，在频率高时尺寸可以做得较小。但 PCB 天线设计难度较大，有一定的方向性，通常还需要仿真工具的支持，实际设计中一般都是原封不动的按照芯片厂商给出的天线参数来做，但其成本最低。Chip 天

线一般就是陶瓷天线，体积很小，常用在天线空间比较小的场景中，其方向性没有 PCB 天线那么明显，但成本比较高。Whip 天线性能最好，但体积也很大，成本也较高。为了获得比较远的通信距离，本设计最终选择了标准 SMA(SubMiniature version A)接口的 Whip 天线。

(a) PCB 天线 (b) Chip 天线 (c) Whip 天线

图 3.9 常见天线形状

天线又可分为单端天线和差分天线。单端天线又称为非平衡天线，差分天线又称为平衡天线。单端天线主要依靠地(ground)作为参考信号，它的特征阻抗一般为 50 欧姆。但好多 Modem 芯片(如 MC13213)都具有差分的天线接口，因此，若使用单端天线，就需要有一个电路来改变这种特性，这个电路就称为平衡/不平衡变压器(Balun，也有人称之为巴伦)。

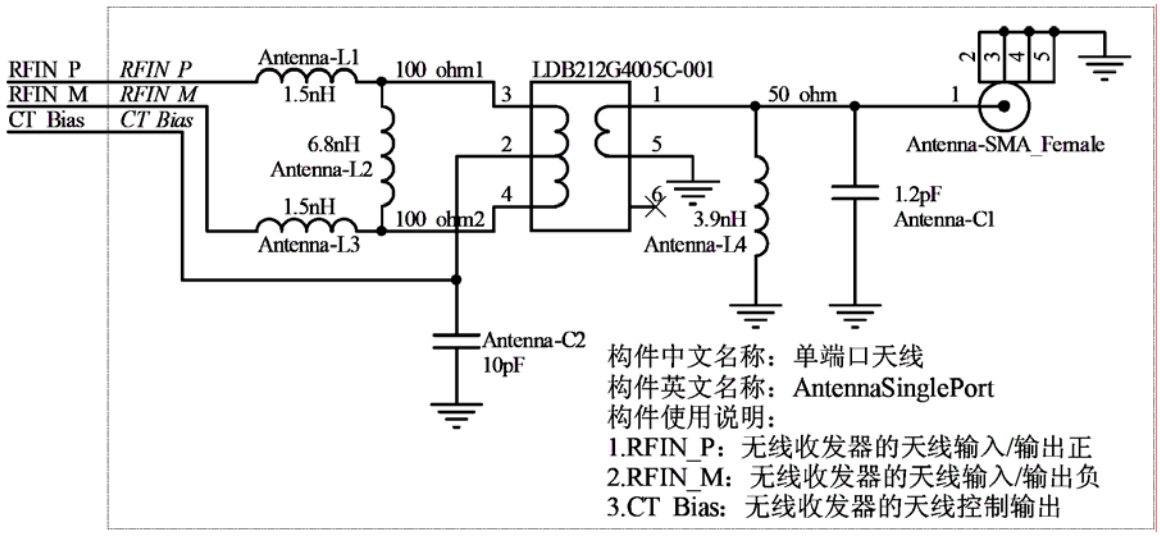


图 3.10 天线模块

天线模块的电路如图 3.10 所示，图中使用的 Balun 是日本村田(Murata)公司的 LDB212G4005C-001。模块中的 CT_Bias 引脚用来为 Balun 提供合适的偏压(无线发送时其值为电源电压 VDDA；接收时其值为参考地 GND)。这里需要注意的是，我们可以使用 Modem 的内部收/发开关，也可以自己另加外部收/发开关。外部收/发开关的收发效果一般会更好一点，但这样一来，成本就增加了，即需要增加 1 个收/发开关和 1 个 Balun。本版的硬件设计中使用的是内部收/发开关。

作为一个典型的硬件构件，在天线模块设计过程中遵循了以下的设计原则[41]：

(1) 构件命名及内部元器件命名要合理。构件的名称既要简洁，又要能体现构件的功能。为了避免当硬件构件移植到其他系统时出现元器件重名现象，构件内部元器件按照“构件名称-元器件名称”格式命名。

(2) 文字描述要详尽。应为硬件构件添加详细的文字描述，包括构件中英文名称、功能描述、接口描述、注意事项等，以增强硬件构件原理图的可读性，提高构件的友好度。

(3) 构件实体和描述要封装。构件如同一个“黑盒”，在使用构件“组装”一个具体的嵌入式硬件系统时，设计者不必关心构件内部具体的实现细节。因此，可用矩形虚线框将构件实体和文字描述封装起来。

(4) 接口标识要区分。接口是各构件之间连接的通道。构件的接口注释标于矩形虚线框内，为斜体字，是接口的解释性文字；而接口网标位于矩形虚线框外，且具有电气特性。

以上即为在实现 ZigBee 硬件过程中所遵循的嵌入式硬件构件设计原则。

3. 外围扩展模块

文中把除最小系统以外的所有其它模块统称为外围扩展模块。本版硬件中所实现的扩展模块主要包括 SCI 通信模块和系统运行状态指示灯。这些都是嵌入式系统中最常用和最基本的模块，限于篇幅，这里不加详述。

需要注意的是，在选择元器件和芯片时，要考虑其工作电压和电流，因为 ZigBee 节点的基本要求就是节能和高效。

4. 实现PCB的注意事项

在印刷电路板(Printed Circuit Board，PCB)的设计过程中一定要格外细心。一般应按模块将相关元器件尽量放在一起，特别是滤波电容，应靠近相应的引脚，否则就起不到滤波作用了。另外 PCB 板上应预留出一定量的测试点，以便以后测量使用。天线模块电路的设计需要较丰富的设计经验，对于元件摆放位置、线的走向及过孔等都有一定的要求，可用双面铺地来提高抗干扰性，这部分的设计主要是参考芯片厂商的相关资料，并在实验室原有的基础上设计实现的。

3.2.4 硬件的焊接和测试

在完成硬件电路设计并联系相关厂家完成 PCB 板的制作之后，就需对各硬件模块进行焊接并测试，以验证硬件电路的正确性和可靠性。

通过万用表可以测出电源模块是否正常供电。通过示波器检测晶振的频率，得知晶振运行情况。另外通过 BDM 写入口，将具有基本 I/O 操作的程序下载到芯片中，如能正常写入则 BDM 电路工作正常，然后通过检查 I/O 口是否受程序控制可以得出

芯片是否正常工作。

硬件的焊接和测试是一个复杂的过程。在首次焊接和测试一个新硬件电路时，千万不要将所有元件都焊接上再进行测试，而应按一定顺序，逐模块的焊接和测试，这样当硬件在某一阶段出现问题时，可以很快定位硬件故障并加以解决，这是在实践过程中得出的重要经验。作者在硬件焊接和测试过程中就遇到了由于芯片引脚虚焊而导致的串口通信时而不正常的问题，但由于使用了上述方法，所以很快就定位到错误，解决了问题。

3.3 物理层功能实现

3.3.1 底层硬件驱动程序的实现

硬件驱动程序介于底层硬件和 ZigBee 协议栈之间，可以使得运行于硬件之上的 ZigBee 协议栈更易于维护和移植。

1. 芯片初始化程序

芯片初始化程序对 MCU 的一些硬件模块进行正确的配置，以保证 MCU 可以正常工作。这里所做的主要配置如下：

(1) 关闭看门狗，防止其计数溢出而导致的芯片复位。一般在代码开发阶段会关闭看门狗，但在产品最终发布时应加入看门狗模块的代码，以提升产品稳定性。

(2) 设置内部时钟模块的校准(trim)值为 0x86。本文中，内部时钟模块的校准值是通过下面的方法得到的：首先设置欲得到的总线频率，然后利用 MCU 定时器模块的输出比较功能输出一个 1KHz 的方波(这里，定时器模块的时钟源一定要选择总线时钟)，然后不断调整 MCU 校准值并通过示波器观察方波频率变化情况。通过示波器观察发现，当 trim 值为 0x86 时，方波频率稳定在 1KHz。

(3) 配置 MCU 的时钟模块，这里选择 FLL 使能内部(FLL Engaged Internal，FEI)模式[42]。利用 MCU 内部的 243KHz 参考时钟和锁频环(Frequency Locked Loop，FLL)电路，产生 16MHz 的总线频率。

(4) 等待 FLL 电路稳定。当 MCU 时钟模块的 FLL 电路锁定到用户设置的频率时会设置一个状态位，可通过查询这个状态位以确定时钟模块是否正常工作。

2. SCI通信模块

实验室已经有非常成熟和规范的串口收发子程序，只需在此基础之上实现一个环形的串口接收缓冲区(循环队列)即可。缓冲区的大小可根据实际的通信需求以及 MCU 的内存资源而确定。

缓冲区是为了解决速率不匹配问题而引入的，通常串口通信速率相对于 MCU 的

处理速度而言是非常之慢，所以考虑到程序的执行效率，尽量不要在串口接收中断程序中等待接收后续的串口数据。另外通过这个接收缓冲区还可以防止因 MCU 正在处理其它事情而来不及接收串口数据时所引起的数据丢失。

3. SPI循环事务的实现

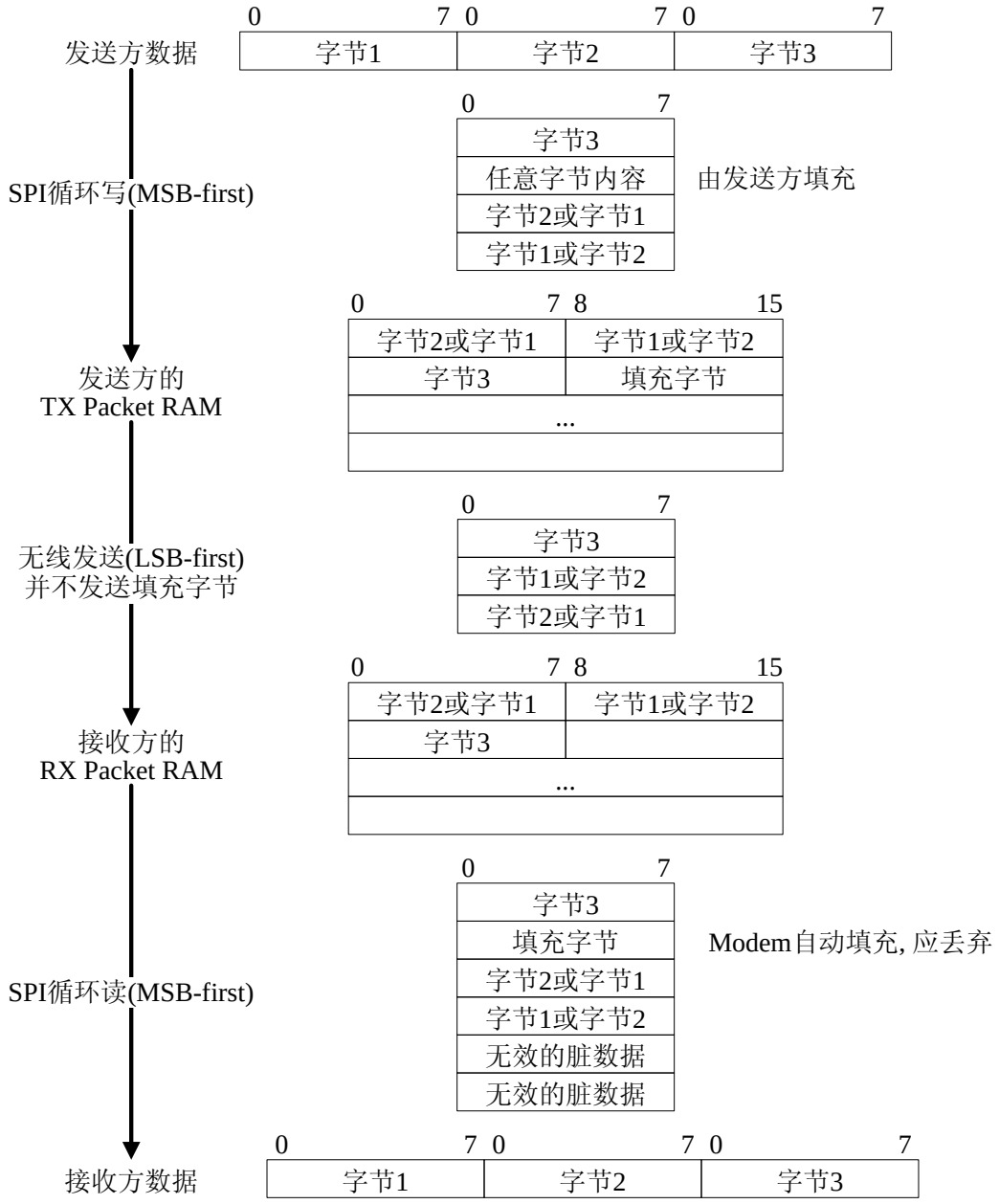


图 3.11 读写 Modem RX/TX RAM 的详细示意图

SPI 作为 Modem 与 MCU 间最主要的通信接口，提供了 SPI 单次事务和 SPI 循环事务操作。SPI 单次事务只需严格按照图 3.5 中的时序图进行实现即可，这里不作详述。下面以 Packet 模式下发送和接收 3 字节数据的完整过程来描述对 Modem RX/TX RAM 的 SPI 循环读写操作，如图 3.11 所示，这是经过多次失败和测试总结得到的。

其中，RX/TX RAM 的长度为 128 字节。图中假定 MCU 均是以字节数组的形式来保存待发送或接收到的数据。

从图 3.11 可以看出，读/写 RAM 时的 SPI 通信是最高有效位优先(MSB-first)的，而在无线发送/接收过程中是最低有效位优先(Least Significant Bit first，LSB-first)的，但在编写 SPI 循环读写操作时并不需要考虑上述两种顺序，那并不会导致接收方在接收发送方的数据时产生比特位顺序的改变。

需要特别注意的地方是，由于 SPI 事务要求所有的数据传输都是按 16 位宽度进行的，当发送数据是奇数个字节时，其最后一字节数据要进行特别处理，即需填充一个任意字节以凑满 16 位宽度，但是这个拼凑的字节和最后那个有效字节的发送顺序必须按照图 3.11 中的顺序进行，即先发填充字节，保证了在 TX RAM 中，最后一字节紧跟在前面的偶数个字节之后。而在最后一字节数据之前的偶数个字节数据由于是 16 位宽度的倍数，所以在发送每个 word 时对字节发送顺序没有特别要求，只要接收方和发送方按照同一种顺序收发各字节即可。但是为了保证与其它协议栈的兼容性，最好采用图中的第二种顺序。

1) 使用 SPI 循环写事务向 TX RAM 中写入待发送数据

执行这个操作之前，待发送数据长度应已经写入 TX_Pkt_Control 寄存器的 tx_pkt_length[6:0]字段。

MCU 向 TX RAM 中写入待发送数据的一般流程如下：

(1) 根据需要配置 TX_Pkt_Control 寄存器的 tx_ram2_select 位，以选择使用两块 TX RAM 中的哪一块；

(2) 计算写入待发送数据所需要的 SPI 脉冲个数，注意：

① CRC 字节不需写入到 TX RAM 中，它是由硬件自动产生的；

② 待发送数据的最大长度为 125 字节(去掉 2 字节的 CRC)；

③ 必须为偶数个字节，若数据长度为奇数个字节，应加 1 使其变为偶数；

(3) 做一个 SPI 循环写事务来写入数据：

① MCU 拉低 SPI 模块的片选信号 $\overline{CE}$，选中 Modem；

② MCU 向 Modem 发送第一个 SPI 脉冲，其中 R/$\overline{W}$ 位应为 0，表示写操作；

③ 按照(2)中计算的 SPI 脉冲个数，写入待发送数据；

④ MCU 拉高 $\overline{CE}$，使片选失效；

(4) 整个写操作结束。

2) 使用 SPI 循环读事务读取 RX RAM 中的已接收数据

MCU 读取 RX RAM 中的已接收数据的一般流程如下：

(1) MCU 读 Modem 的 RX_Status 寄存器 rx_pkt_latch[6:0]字段以获取数据长度；

(2) 计算读取 RX RAM 中的已接收数据所需要的 SPI 脉冲个数：

① 通常不读取 2 字节的 CRC，所以数据长度应减去 2；

② 若数据长度为奇数个字节，应加 1 使其变为偶数；

③ 按照 Modem SPI 事务协议的规定，应丢弃读到的第一个字(word)，因为在第一次读取时，内部 RAM 的地址还没有准备好，这样又导致了数据长度加 2；

(3) 做一个 SPI 循环读事务来读取数据：

① MCU 拉低 SPI 模块的片选信号 $\overline{CE}$，选中 Modem；

② MCU 向 Modem 发送第一个 SPI 脉冲，其中 R/$\overline{W}$ 位应为 1，表示读操作；

③ 按照(2)中计算的 SPI 脉冲个数读取所有数据。注意，协议规定应丢弃读到的第一个字(word)。当数据为奇数个字节时，应丢弃图 3.11 中的那个填充字节；

④ MCU 拉高 $\overline{CE}$，使片选失效；

(4) 整个读操作结束。

3.3.2 设置 Modem 运行模式

Modem 有多种运行模式，主要可分成两类[42]：活动模式和低功耗模式。其中活动模式包括 Idle 模式、Receive(RX)模式、Transmit(TX)模式和 CCA/ED 模式；低功耗模式包括 Off 模式、Hibemate 模式、Doze 模式。

Off 模式的功耗最低，当 Modem 的 $\overline{M\_RST}$ 引脚被拉低时会进入该模式；Hibernate 模式的功耗仅次于 Off 模式；Doze 模式下大部分硬件模块都不工作，但参考时钟和事件定时器仍工作。Hibernate 模式和 Doze 模式都可以通过 $\overline{ATTN}$ 引脚唤醒并进入 Idle 模式，Doze 模式下还可以通过事件定时器唤醒。

表 3.2 Modem 的模式定义及转换时间

模式	SPI	晶振	CLKO	RAM 数据是否丢失	数字引脚输出为三态	转换时间
Off	x	x	x	是	√	10～25 ms 到 Idle
Hibernate	x	x	x	否	x	8～20 ms 到 Idle
Doze	x	√	≤1MHz	否	x	(300+1/CLKO)μs 到 Idle
Idle	√	√	√	否	x	0
RX	√	√	√	否	x	144 μs 从 Idle
TX	√	√	√	否	x	144 μs 从 Idle
CCA/ED	√	√	√	否	x	144 μs 从 Idle

Idle 模式是 Modem 退出任何其它模式后的默认模式，也是进入任何其它模式的初始模式；RX、TX 模式分别为 Modem 接收、发送数据时所处的工作模式；CCA/ED 模式为空闲信道评估/能量检测时所处的工作模式，用来评估信道是否空闲或测量信

道的当前能量值。

Modem 在各工作模式下硬件模块的状态不尽相同，并且从 Idle 模式到其它模式或从其它模式到 Idle 模式的转换时间也有所不同，如表 3.2 所示[42]。注意，所有模式的转换都必须以 Idle 模式为初始模式，模式间的转换条件如图 3.12 所示[42]。

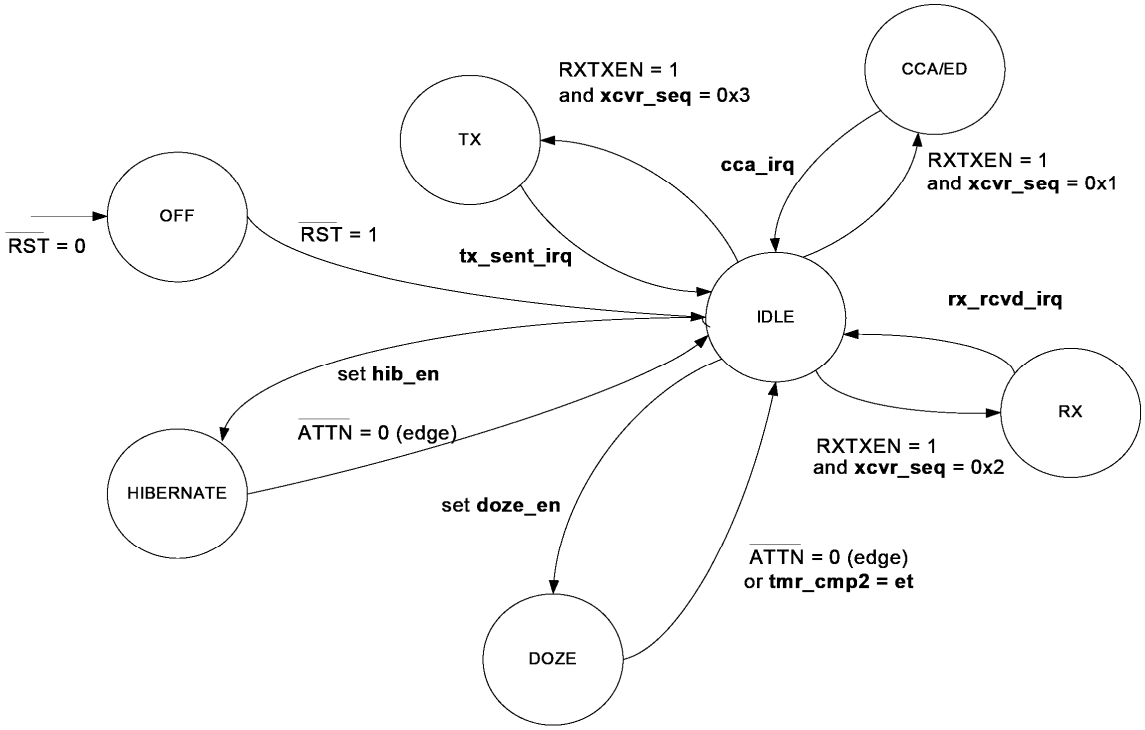


图 3.12 Packet 模式下的 Modem 工作状态转换图

收发机状态设置是通过调用设置收发状态原语实现的，其函数头如下：

```
//-------------------------------------------------------------------------*
//功   能: 设置收发机状态原语，把收发机设置成用户期望的状态
//参   数: nDesiredStatus        - 用户期望状态
//返   回: SUCCESS               - 成功设置成指定模式;
//         等于用户期望模式      - 收发机之前就处于用户期望状态;
//         其它                  - 表示执行失败;
//说   明: 无
//-------------------------------------------------------------------------*
INT8U PLMESetTRXState(INT8U nDesiredStatus);
```

3.3.3 物理层数据包的收发

物理层数据称为物理层数据单元(PHY Protocol Data Unit，PPDU)，包括同步包头、物理层包头和物理层载荷 3 个部分[2]，如图 3.13 所示。其中同步包头可以使得接收设备锁定在比特流上，并与比特流保持同步；物理层包头包含了数据包的长度信息，在 0 到 127 之间；物理层服务数据单元(PHY Service Data Unit，PSDU)，也称物理层载

荷，携带 MAC 层的帧信息，即 MAC 层协议数据单元(MAC Protocol Data Unit，MPDU)。注意，CRC 检验字节包含在 PSDU 中的最后两字节中。

4 字节	1 字节	7 位	1 位	变量
前同步码	帧定界符	帧长度	预留位	PSDU
同步包头		物理层包头		物理层载荷

图 3.13 PPDU 格式

物理层数据包的发送和接收比较简单，对于 MC13213 的 Modem 来说，用户所要做的只是调用 SPI 循环事务写入或读取物理层包头和载荷，然后拉高 RXTXEN 引脚使能 Modem 的发送和接收即可。PPDU 的同步包头是由硬件自动添加的。

1. 物理层数据包的发送

Packet 模式下发送数据包的一般过程如下：

(1) MCU 拉低 RXTXEN 引脚；

(2) MCU 设置 Modem 的 Control_A 寄存器的 tx_sent_mask 位为 1，以允许产生 TX 完成中断请求；

(3) MCU 加载数据长度(包括 2 字节 CRC)到 Modem 的 TX_Pkt_Control 寄存器的 tx_pkt_length[6:0]字段；

(4) MCU 使用 SPI 循环写事务预加载待发送数据至 TX RAM 寄存器中；

(5) MCU 设置 Control_A 寄存器中的 xcvr_seq[1:0]字段为 0x03，表示 Modem 将要转换到 TX 工作模式；注意，在这之前，Modem 应处于 Idle 模式，因为它是任何模式转换的初始模式；否则，发送过程应中止，并向上层通知出错信息；

(6) MCU 拉高 RXTXEN 引脚并保持，以促使 Modem 完成工作模式的转换，并启动数据发送过程；

(7) 当数据被成功发送后，IRQ_Status 寄存器的 tx_sent_irq 状态位置 1，以指示发送完成，并根据 2)中的设置，产生一个 IRQ 中断；

(8) 为响应 Modem 的中断请求，MCU 应读取相应的状态位来清除中断，并向上层通知数据发送成功的消息。

以上过程是在物理层数据请求原语中实现的，在原语实现过程中，应根据要求，向上层通知数据发送结果的状态信息。其函数头如下：

```
//------------------------------------------------------------------------*
//功    能: 数据请求原语，生成物理层协议数据单元(PPDU)并无线发送出去
//参    数: nPSDULength - 物理层 PSDU(即 MAC 层的 MPDU)中的字节数，
//                        长度要<=aMaxPHYPacketSize(物理层最大数据包容量)
//          pPSDU       - 指向物理层 PSDU 数据的指针
//返    回: SUCCESS - 发送成功；其它值 - 发送失败
//说    明: 由 MAC 层调用
```

```
//-------------------------------------------------------------------------*
INT8U PDDataRequest(INT8U nPSDULength, INT8U *pPSDU);
```

2. 物理层数据包的接收

一般情况下，Modem 的接收机是关闭的。当 Modem 接收机处于打开状态时有其它 Modem 在同一信道上发送数据，则 Modem 会接收到这些数据。

Packet 模式下接收数据包的一般过程如下：

(1) MCU 拉低 RXTXEN 引脚；

(2) MCU 设置 Modem 的 Control_A 寄存器的 rx_rcvd_mask 位为 1，以允许产生 RX 完成中断请求；

(3) MCU 设置 Control_A 寄存器中的 xcvr_seq[1:0]字段为 0x02，表示 Modem 将要转换到 RX 工作模式；注意，在这之前，Modem 应处于 Idle 模式，因为它是任何模式转换的初始模式；否则，接收过程应中止，并向上层通知出错信息；

(4) MCU 拉高 RXTXEN 引脚并保持，以促使 Modem 完成工作模式的转换，并启动数据接收过程；

(5) 当成功收到一个数据包后，Modem 将向 MCU 报告以下信息：

① RX_Status 寄存器的 rx_pkt_latch[6:0]字段保存了接收到的数据包的长度，长度中包含了 2 字节的 CRC；

② IRQ_Status 寄存器的 crc_valid 位指示了 CRC 检验是否合法，“1”为合法；

③ RX_Status 寄存器的 cca_final[7:0]字段保存了本次接收过程中的链路质量(LQI)值；

④ IRQ_Status 寄存器的 rx_rcvd_irq 状态位置 1，以指示接收完成，并根据 2)中的设置，产生一个 IRQ 中断；

(6) 为响应 Modem 的中断请求，MCU 应：

① 读取 rx_rcvd_irq 位和 crc_valid 位来验证数据的合法性；读取 rx_pkt_latch [6:0]字段来确定合法的数据长度；合法的数据长度应≥3，因为其中包含了 2 字节的 CRC；

② 使用 SPI 循环读事务来获取接收数据；

③ 向上层通知数据接收成功的消息。

关于对物理层数据包发送完成和接收完成中断请求的处理过程，在后面的 IRQ 中断处理程序部分有详细地阐述。

与串口类似的，在本协议栈的实现过程中，也为物理层数据包实现了一个环形的接收缓冲区，以保证数据的及时可靠接收，其结构定义如下：

```
//PSDU 最大数据包长度(不包括 CRC)
#define PSDUMaxLen                125
//PSDU 数据包定义
```

```
typedef struct PSDURxPacket_tag
{
    INT8U m_nLen;
    INT8U m_nData[PSDUMaxLen];
    INT8U m_nStatus;
    INT8U m_nLQI;
} PSDURxPacket_t;

//PSDU 数据接收缓冲区个数
#define PSDURxBufferNum          8
//PSDU 数据接收缓冲区定义
typedef struct PSDURxBuffer_tag
{
    INT8U m_nPSDUCount;
    INT8U m_nHead;
    INT8U m_nTrail;
    PSDURxPacket_t m_sPSDU[PSDURxBufferNum];
} PSDURxBuffer_t;

//定义 PSDU 环形接收缓冲区
static PSDURxBuffer_t s_sPSDURxBuffer;
```

开始时是使用 C 标准库中的动态内存分配函数 malloc 来实现接收缓冲区的，但在实际测试中发现，即使在 main 函数中仅调用 malloc 函数，然后查看编译器生成的 map 文件，RAM 也需占用 2KB，这应该是为支持 malloc 函数所维护的一个动态内存区，但对于 RAM 资源十分宝贵的 MCU 来说，这个代价有些过大。

为了实现上的方便，物理层中使用了静态数组来实现环形缓冲，并且 PSDU 数据包也是定长的。这样做虽然浪费了一些存储空间，但是在拥有 8 个 PSDU 缓冲的情况下仍比调用 malloc 函数少用了一半的存储空间。但是从去除不必要的 RAM 空间占用的角度出发，在协议栈今后的升级中，可以考虑自己实现一个动态内存分配函数，以解决存储空间浪费问题。

3.3.4 空闲信道评估/能量检测

空闲信道评估(CCA)用来判断信道是否空闲。能量检测(ED)用来测量目标信道中接收信号的功率强度，由于这个检测本身不进行解码操作，所以检测结果是有效信号功率和噪声信号功率之和。其实 CCA 和 ED 的基本过程是一样的，唯一的不同在于 CCA 多做了阈值比较操作，即 CCA 会把能量检测值与一个预先定义的阈值进行比较来决定信道空闲与否。

另外，链路质量指示(LQI)提供了接收数据包时无线信号的强度和信道质量信息。与能量检测不同的是，它要对信号进行解码，生成的是一个信噪比指标。这个信噪比指标和物理层数据单元一起提交给上层处理。

Modem 中 RX_Status 寄存器的 cca_final[7:0]字段保存了以上操作的结果值。

3.3.5 IRQ 中断处理程序

当 Modem 完成 MCU 指定的某个功能，如发送完成、接收完成、CCA/ED 完成等时，就会产生 IRQ 中断向 MCU 通知 Modem 的当前状态，然后 MCU 的 IRQ 中断处理程序会读取 Modem 的 IRQ 状态寄存器，针对不同的 IRQ 中断类型分别进行处理。目前处理的中断类型包括发送完成、接收完成、CCA/ED 完成、Modem 的 PLL 时钟失锁等，以后会根据系统需求再加入其它类型的中断处理。IRQ 中断程序处理流程如图 3.14 所示。

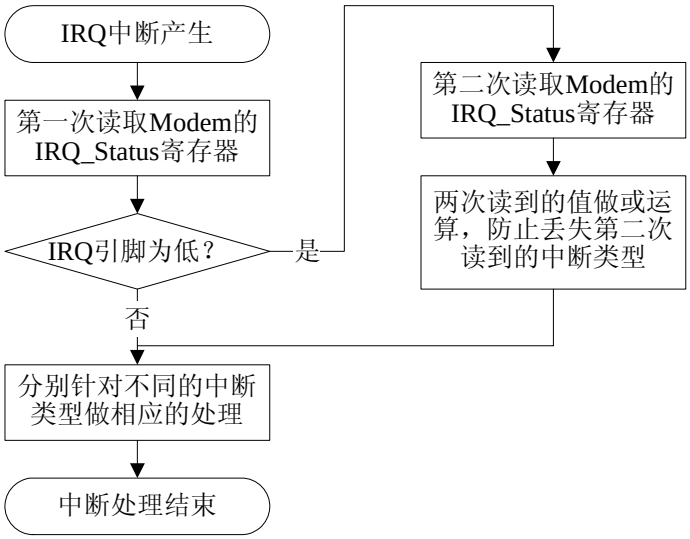


图 3.14 IRQ 中断程序处理流程

当 IRQ 引脚为低电平时，会产生 IRQ 中断。一般来说，MCU 读取 IRQ_Status 寄存器后，IRQ 引脚会变高，但当多个 IRQ 中断事件以非常短的时间间隔连续出现时，会出现 IRQ 引脚刚变高又被新的中断事件拉低的情况。所以在中断处理过程中，读完 IRQ_Status 寄存器后，会立刻判断 IRQ 引脚电平是否为低，若为低则再次读取 IRQ_Status 寄存器以使 IRQ 引脚变高。同时，这样做还可以防止 IRQ 引脚上的假信号的干扰。

3.3.6 物理层属性的设置和获取

物理层管理实体(PLME)维护了物理层正常工作所必须的一些属性参数，包括物理层支持的信道列表、当前用于发送和接收的信道、物理层的发射功率以及 CCA 模

式等 4 个属性。每个属性都有一个唯一的属性标识符，并且某些属性还有一些特定的取值范围。属性的读和写分别由属性设置和读取原语来实现，由于物理层的属性较少，直接通过 switch/case 语句实现即可。

3.3.7 遇到的问题及解决方法

正所谓万事开头难。SPI 通信模块作为第一个与 Modem 交互的模块，在其实现过程中遇到了一些实际困难。

1. 读Modem寄存器内容不正确

在实现 MC13213 的 SPI 单次读事务操作时，所读的寄存器内容一直不正确。开始还怀疑是硬件的原因，后经仔细检查发现罪魁祸首是 while(SPI1S & (1 << 7) == 0);，这条语句的功能是等待 SPI 数据接收缓冲区满标志，但由于==运算符的优先级高于&，导致了 while 循环的条件一直为 FALSE！而在编译过程中编译器也给出了“Condition always is FALSE”的警告信息，但是并没有引起关注。该语句正确的写法应为 while((SPI1S & (1 << 7)) == 0);。

从这个教训中可以得到几个经验：(1) 书写代码时一定要仔细，对于优先级不确定的运算符，可以通过加小括号来准确限定其运算顺序；(2) 编译器的警告信息一定要关注，并力求去除所有警告信息；(3) 类似的经常容易犯的错误是没有包含所调用函数的声明信息，如果对这个警告置之不理的话，程序就会得到一些莫名其妙的错误，需要引起特别注意。

2. 错误地清除了SPI状态寄存器

当 Modem 上电启动完成进入 Idle 模式时会产生一个 ATTN IRQ 中断，这时 MCU 可用一个 SPI 单次读事务，查看 Modem 的 IRQ_Status 寄存器中的 attn_irq 位是否置位，以确定是否是 ATTN IRQ 中断。正常的话读到的值应为 0x0400，但是实际读到的一直是 0x0004。经多次测试，最终发现问题出现在 SPIInit()函数中所调用的 ClearSPIStatusR()宏！其原始定义以及改正后的定义如下：

原始定义	改正后定义

//前两句清除接收缓冲区满(SPRF)状态位；
//后两句清除发送缓冲区空(SPTEF)状态位
#define ClearSPIStatusR() \
{ \
char temp = SPIStatusR; \
temp = SPIDataR; \
temp = SPIStatusR; \
SPIDataR = temp; \
}
	//清除接收缓冲区满(SPRF)状态位；
	//注意！！若无特殊原因，不要去
	//改变发送缓冲区空(SPTEF)状态位
	#define ClearSPIStatusR() \
	{ \
	char temp = SPIStatusR; \
	temp = SPIDataR; \
	}

这个宏本来用于清除 SPI 状态寄存器的，其实前两句话执行任意多次都不会出现问题，但当执行 SPIDataR = temp;后，会立即启动一次主从机的数据交换，并且交换完成后 SPRF 会置位。

从 MC13213 的芯片手册上得知，SPI 模块是双缓冲的，即它除了移位寄存器外，还存在单独的数据发送和接收缓冲，读数据寄存器会返回接收缓冲的数据，写数据寄存器会写数据到发送缓冲。这样在读取上次接收的数据之前，本次 SPI 传输就已经可以开始，提高了数据传输速度。

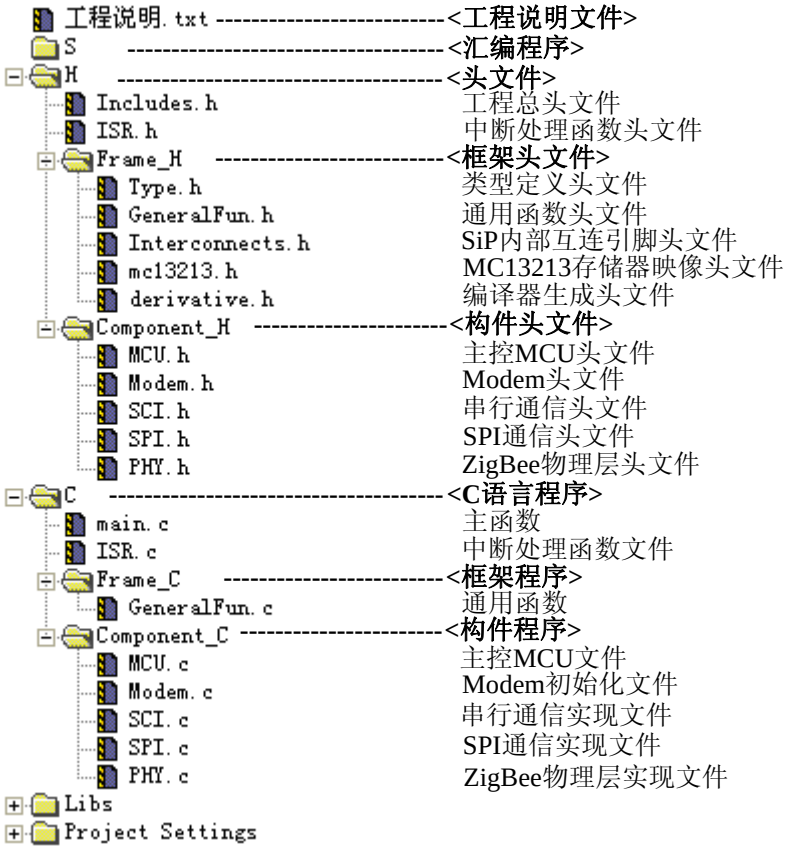


图 3.15 ZigBee 物理层工程组织结构图

由前面介绍可知，SPI 单次读事务由 3 个 SPI 脉冲组成，第 1 个脉冲用来传输寄存器的地址，后 2 个脉冲用来读取寄存器的内容，每个脉冲都会启动一次主从机的数据交换。第 1 个脉冲期间，正常情况下要等 8 个 SPI 时钟周期后，一字节数据才交换完成，SPRF 位才会置位，这时 MCU 才读取数据。但是由于之前 ClearSPIStatusR()中后两句的操作导致 SPRF 位早已置过位了，所以此时 MCU 无需等待 8 个 SPI 时钟周期就立即读取数据寄存器中上次的值。这样在第 1 个 SPI 脉冲结束后，MCU 确实是把寄存器地址传给了 Modem，但此时读到的数据并不是本次交换来的数据。根据 SPI 单次读事务协议，第 1 个脉冲获得的数据被丢弃。

在第 2 个 SPI 脉冲期间，指定寄存器的高字节内容会交换给 MCU，与第 1 个脉

冲一样，MCU 无需等待 8 个 SPI 时钟周期就读取数据，但是此时 MCU 获得的其实是在第 1 个 SPI 脉冲时所交换来的数据，经测试是 0x00。同样的，在第 3 个脉冲时读取的是在第 2 个 SPI 脉冲时交换来的数据 0x04，而第 3 个脉冲交换来的数据 0x00 并没有被读取。这样一来，导致了读到的数据一直是 0x0004。

总结经验教训的同时，从上面的分析中可以得出几条结论：(1) 为防止以前的 SPI 操作影响到即将进行的 SPI 操作，建议在每次 SPI 操作之前，先清除 SPI 状态寄存器中的 SPRF 状态位；(2) SPI 状态寄存器中的 SPTEF 状态位在程序中如无特殊需要，一般无需做额外处理；(3) Modem 交换给 MCU 的无效数据是 0x00。

3.3.8 ZigBee 物理层工程组织结构

图 3.15 给出了实现 ZigBee 物理层协议时的工程组织结构图。在协议栈功能实现过程中，遵循了本实验室的嵌入式软件构件编程规范，采用面向硬件对象和模块化封装的基本思想进行编程，尽量不使用全局变量，代码结构和注释清晰，易于理解和阅读，具有比较好的可移植性和可维护性。

3.4 物理层测试

3.4.1 SPI 单次读写事务的正确性测试

这部分的测试执行起来比较简单。在 Modem 复位完成之后，就可以接收 SPI 命令了。所以在对 Modem 的内部寄存器初始化之前，可利用 SPI 单次读事务获取 Modem 寄存器的内容并通过串口输出显示，将这些值与 Modem 寄存器的复位值对比，若二者一致则表示 SPI 单次读事务的实现是正确的。接着利用 SPI 单次写事务，对 Modem 进行初始化，初始化完成之后，再把修改后的 Modem 寄存器的内容输出到串口显示，然后与修改值比较，即可得出 SPI 写事务实现的是否正确。

3.4.2 物理层数据包的收发测试

物理层数据包的收发测试需要一个发送节点和一个接收节点相互配合。对于能否正确收发需要测试两种情况，发送节点分别发送奇数个和偶数个字节的数据，看接收节点能否正确收到。这部分的测试也是借助于串口调试工具来完成的，接收节点把收到的数据发往 PC 机串口显示。

在实际测试中确实也发现了一些问题，如 PLMESetTRXState()函数中设置 Modem 工作模式的语句，以设置 TX 模式为例：

```
content = (content & XCVR_SEQ_MASK) | XCVR_SEQ_PKT_TX;
```

正确的写法应为：

content = (content & ~(XCVR_SEQ_MASK)) | XCVR_SEQ_PKT_TX;

其中，XCVR_SEQ_MASK 是用来获取 Modem 当前模式的掩码。若需要设置为用户期望的新状态，应该使用 XCVR_SEQ_MASK 取反之后的值。这个错误会导致收发机状态不能正确设置，最终导致只能发送一次数据！

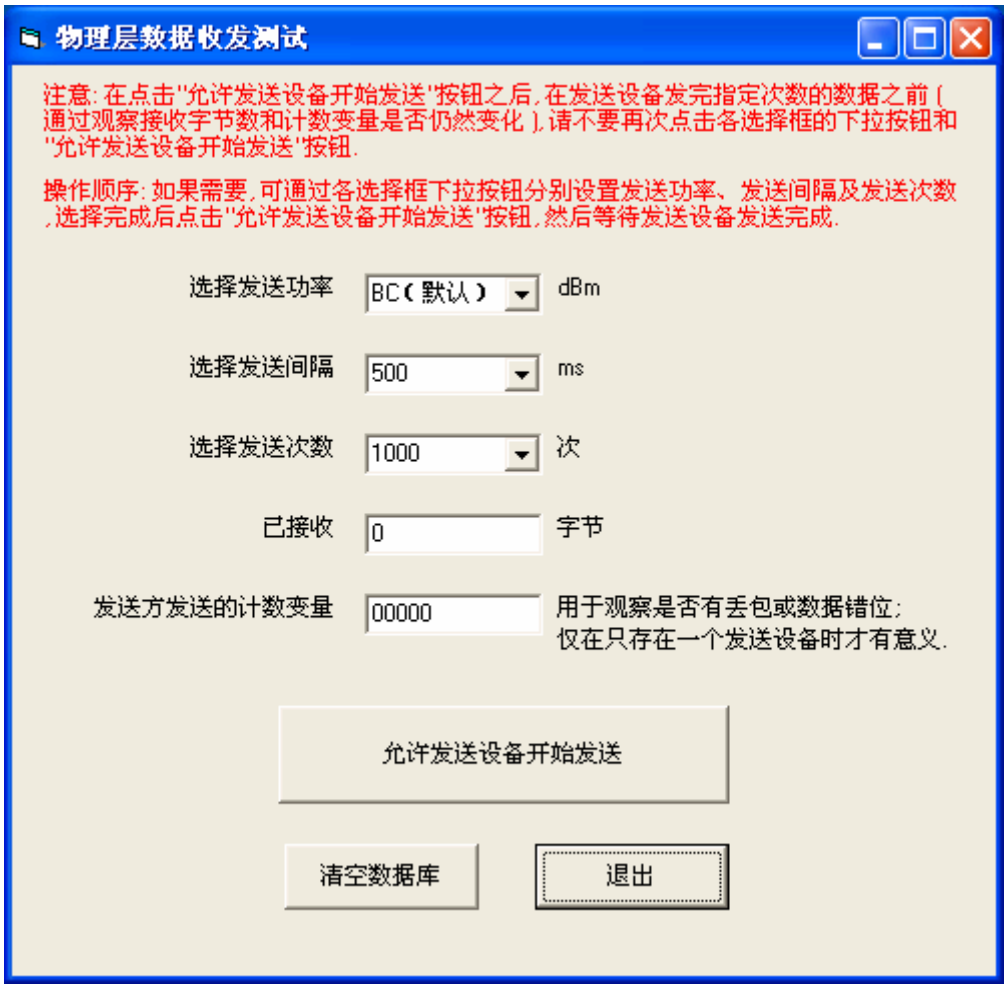


图 3.16 物理层数据收发测试软件界面

物理层数据包收发的可靠性或数据包丢失情况也应该进行重点测试，以保证物理层功能的可靠性。测试条件如下：一个发送节点和一个接收节点，二者相距 5 米左右，其中发送节点每次发送长度 20 字节长度的数据，并且数据中的最后两字节作为一个 16 位的整数，用来记录发送次数，每发送一次其值加 1。发送节点何时开始发送数据由 PC 方测试软件控制，接收节点负责接收数据并发给 PC 端测试软件显示，通过比较发送字节数与接收字节数以及数据中的发送次数字段，可以得出数据丢失情况，PC 端测试软件界面如图 3.16 所示，所有的测试数据会写入后台的 ACCESS 数据库中，以供将来进行数据的统计分析。试验中测试的一组数据如表 3.3 所示，达到了小于 1% 的丢包率。

表 3.3 物理层数据包收发测试结果

发送功率(dBm)	发送间隔(ms)	发送数据包个数	接收数据包个数
-0.66	500	100	100
-0.66	500	1000	1000
-0.66	200	100	100
-0.66	200	1000	999
3.4	500	100	100
3.4	500	1000	1000
3.4	200	100	100
3.4	200	1000	1000

3.5 本章小结

本章给出了 ZigBee 芯片的物理工作特性，概括了物理层的参考模型和功能。分析了几种常见的 ZigBee 硬件方案并选择了 MC13213 做为硬件平台，按照嵌入式硬件构件的原则，详细设计和实现了 ZigBee 硬件各模块，指出了在硬件设计、制板、焊接和测试过程中应该注意或避免的事项，并根据实验室积累的多年经验，给出了一种按顺序、分模块逐步测试硬件的方法。描述了 MC13213 中 MCU 与 Modem 的各种交互接口，研究和概括了实现协议栈过程中必须了解的知识点，如 SPI 事务协议、Modem 的工作模式和转换条件等。实现了自制硬件平台的底层硬件驱动程序，以此测试了自制硬件的可行性和正确性，针对在编写 SPI 事务操作程序中所遇到问题的深层次原因进行了剖析，总结了经验教训。仔细分析并实现了 Modem 工作模式设置、物理层数据收发、CCA/ED 等物理层功能，并进行了详细的功能性和可靠性测试。

第四章 ZigBee MAC 层的研究与实现

MAC 协议用于在各节点间公平有效地共享通信信道。本章概括了 MAC 层的基本功能、网络结构及 MAC 帧结构等内容。详细阐述了 MAC 层连接与响应、数据发送和接收的流程，提出了一种管理 MAC 层属性的有效方法。在物理层协议基础上实现了基于非信标网络的 MAC 层协议并进行了相应的测试。

4.1 ZigBee MAC 层概述

4.1.1 MAC 层简介

无线信道属于共享信道，在任何时间，若有一台设备正在使用该信道发送数据，则其它设备就不能使用该信道，否则就会因为冲突(碰撞)而产生错误。如何解决无线信道的合理共享是 MAC 层的关键技术之一，整个网络的性能如吞吐量、延迟、能量消耗等将取决于所采用的 MAC 协议。

4.1.2 MAC 层模型和服务原语

1. MAC层模型

IEEE 802.15.4 定义的 MAC 层参考模型如图 4.1 所示。其中，PD-SAP 是物理层提供给 MAC 层的数据服务接口；PLME-SAP 是物理层提供给 MAC 层的管理服务接口；MCPS-SAP(MAC Common Part Sublayer-Service Access Point)是 MAC 层提供给网络层的数据服务接口；MLME-SAP(MAC Sublayer Management Entity-Service Access Point)是 MAC 层提供给网络层的管理服务接口。

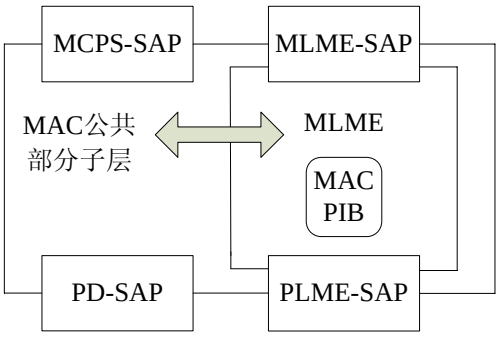


图 4.1 MAC 层参考模型

2. MAC层原语

MAC 层处理所有的对物理无线电信道的访问，它主要完成以下工作：若设备是协调者，则可以选择产生网络信标(beacon)；支持个域网(PAN)的连接和断开；支持设备安全性；采用 CSMA-CA 机制进行信道访问；处理和维护保护时隙(Guaranteed Time Slot，GTS)；为两个对等 MAC 实体之间提供一个可靠的链路。关于 MAC 层原语的详细内容请参考 IEEE 802.15.4 标准。

4.1.3 IEEE 802.15.4 的网络结构和地址类型

1. 信标网络与非信标网络

MAC 层定义了两种基本的操作模式：信标模式(beacon)和非信标(non-beacon)模式，它们对应的网络分别称为信标网络和非信标网络。

在信标网络中规定了一种称作超帧(superframe)的格式。超帧的活跃部分被均匀划分为 16 个等长的时隙，每个时隙的长度由协调者决定，并通过信标帧广播到整个网络。协调者在超帧的开始发送信标帧；紧接着是竞争访问时期(Contention Access Period，CAP)，在这段时间内各节点以竞争方式访问信道；再后面是免竞争期(Contention-Free access Period，CFP)，节点采用时分复用的方式访问信道；然后是非活跃期，节点进入休眠状态，等待下一个超帧周期的开始又发送信标帧。

非信标网络则比较灵活，协调者无需周期性地发送信标帧，而仅在设备请求时才发送，节点均以竞争方式访问信道。由于在信标网络中存在周期性的信标，整个网络的所有节点都能进行同步，但这种同步网络的规模不会很大，一般适用于低延迟设备间的通信。实际上，ZigBee 中用得更多的可能是非信标网络。

在信标网络中，信道的竞争访问使用时隙 CSMA-CA 算法，这时每个设备的退避周期边界都应与超帧时隙的边界一致。而在非信标网络中，使用非时隙 CSMA-CA 算法，对退避周期边界不作任何要求。本设计采用了非信标网络。

2. IEEE 802.15.4定义的网络拓扑结构

LR-WPAN 根据应用需要可以组织成星型(star)网络或对等(peer-to-peer)网络(也称点对点网络)，图 4.2 给出了这两种网络结构。每个网络都由一个 16 位长度的 PAN 标识符(PAN identifier，PAN ID)进行唯一区分。

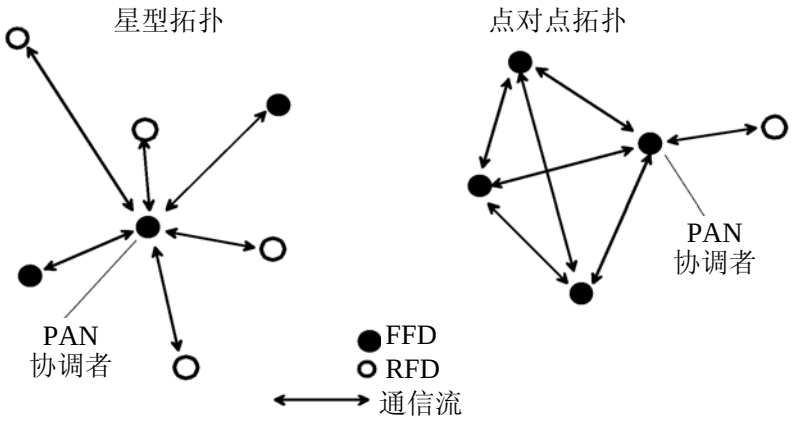


图 4.2 星型拓扑和点对点拓扑

星型网络由一个称为 PAN 协调者的中央控制器(必须为 FFD)和多个从设备(FFD 或 RFD)组成。在星型网络中，所有设备都只能与 PAN 协调者进行通信。星型网络

中的两个从设备如需相互通信，都是先把各自的数据包发送给 PAN 协调者，然后由 PAN 协调者转发给对方。

与星型网络不同，对等网络只要彼此都在对方的无线辐射范围之内，任何两个设备之间都可以直接通信。对等网络中也需要 PAN 协调者，负责实现管理链路状态信息、认证设备身份等功能。对等网络可以支持 Ad Hoc 网络，允许通过多跳路由的方式在网络中传输数据。对等网络可以构造更复杂的网络结构，如树状和网状网络等。

3. IEEE 802.15.4中的两种地址类型

IEEE 802.15.4 中定义了两种类型的地址：扩展地址(extended address)和短地址(short address)。其中，64 位扩展地址是全球唯一地址，也称为 IEEE 地址，它是在设备加入 PAN 之前就分配好的。16 位短地址是设备加入 PAN 后所分配得到的网内局部地址，也称为网络地址。16 位短地址只能保证在本 PAN 内部是唯一的，所以使用 16 位短地址通信时必须结合其所属网络的 16 位 PAN ID 才有意义。为降低帧控制信息的开销，在 ZigBee 网络中主要使用 16 位短地址。

4.1.4 MAC 层的几个概念

1. 数据传输类型

在 IEEE 802.15.4 中，有三种不同的数据转移：从设备到协调者、从协调者到设备、在对等网络中从一方到另一方。为了突出低功耗的特点，可以把数据传输分为以下三种方式[2]：

(1) 直接传输(direct transmission)：适用于以上所有三种数据转移。在使用直接传输时，发送设备需要准确知道接收设备的接收机何时打开，否则会由于接收设备正在休眠而导致数据丢失。

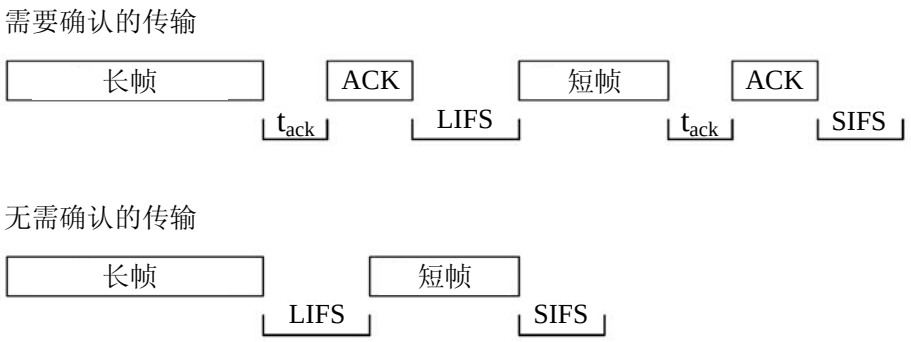
(2) 间接传输(indirect transmission)：仅适用于从协调者到设备的数据转移。间接传输不是由协调者而是由设备首先发起的。在这种方式中，协调者会先把待发送数据保存在自己的事务处理列表中，然后协调者在它发送的信标帧中表明了有某个设备的数据等待传输(信标网络中)或者由设备本身主动向协调者轮询是否有要传输给它的数据(非信标网络中)。这样一来，就可以让设备的接收电路大部分时间处于睡眠状态，从而大幅降低功耗。

(3) 保护时隙(GTS)传输：这仅适用于信标网络中的设备与其协调者之间的数据转移，既可以从设备到协调者，也可以从协调者到设备。

2. 信道访问方式

由于 ZigBee 网络中的所有节点都工作在同一个信道上，因此如果邻近节点同时

发送数据就有可能发生冲突。为此 MAC 层采用了 CSMA-CA 算法来解决无线信道的共享访问问题。简单来说，就是节点在发送数据之前先监听信道，如果信道空闲则可以发送数据，否则就要进行随机的退避，即延迟一段随机时间，然后再进行监听。所有节点竞争共享同一个信道。随机退避时间是指数增长的但有最大值，这样做的原因是如果多次监听信道都忙，则有可能表明信道上的数据量大，因此让节点等待更多的时间，避免繁忙的监听。



aTurnaroundTime ≤t_{ack}≤ (aTurnaroundTime + aUnitBackoffPeriod)

图 4.3 IFS 结构

由于 MAC 层需要一定的时间来处理由物理层接收来的数据，因此，在发送帧之后，应该留有一个帧间间隔(Interframe Spacing，IFS)，如果传输需要确认，IFS 应跟在确认帧之后。IFS 的结构如图 4.3 所示。

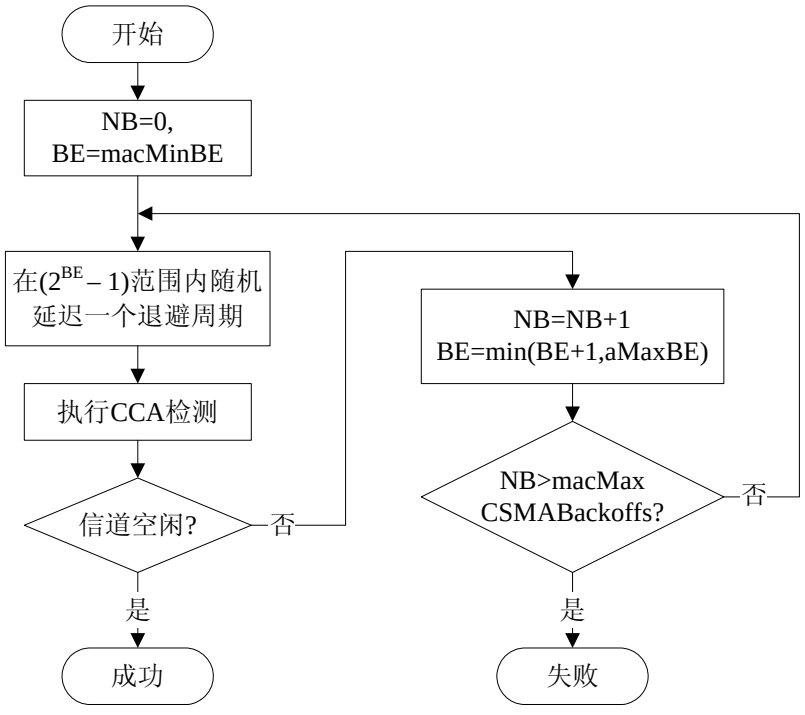


图 4.4 非时隙 CSMA-CA 算法基本流程

IFS 的长度依赖于所发送帧(MPDU)的长度。长度为 18 字节的帧后应跟随一个

至少持续 12 个符号(symbol)的短帧间间隔(short IFS)。长度大于 18 字节的帧后应跟随一个至少持续 40 个符号的长帧间间隔(long IFS)。其中，aTurnaroundTime 的值为 12 个符号时间，退避周期单位即 aUnitBackoffPeriod 的值为 20 个符号。由于 2.4GHz 的符号速率为 62500symbol/s，所以每个符号的持续时间为 16μs。

实现 CSMA-CA 算法时应考虑到 IFS 的要求。由于本协议栈只支持非信标网络，所以只给出了非时隙 CSMA-CA 算法的基本流程，如图 4.4 所示。每个设备为发送任务保持 2 个变量：NB 和 BE。NB 是在执行当前发送任务时，执行 CSMA-CA 算法所需要进行退避的次数，在每次执行新的发送任务之前，这个值初始化为 0；BE 是退避指数，与设备在试图访问信道之前需等待的退避周期有关，在非时隙系统中，BE 初始化为 macMinBE 的值。在 IEEE 802.15.4 中，macMinBE 的值为 3；aMaxBE 的值为 5；macMaxCSMABackoffs 的值为 4。

3. 信道扫描方式

(1) 能量检测扫描：能量检测扫描可以获得被扫描信道的峰值能量。有潜力成为 PAN 协调者的设备可以利用它，在启动一个新的 PAN 之前选择合适的信道。在能量检测扫描期间，MAC 丢弃物理层数据服务传来的所有帧。

(2) 主动扫描：主动扫描的目的是找出其工作范围内的发送信标帧的任何其它协调者。有潜力成为 PAN 协调者的设备可以利用它，在启动一个新的 PAN 之前，选择合适的 PAN ID；设备在连接网络前也可能会使用它。在主动扫描期间，MAC 丢弃物理层数据服务传来的所有非信标帧。

(3) 被动扫描：与主动扫描类似，被动扫描的目的也是找出其工作范围内发送信标帧的任何其它协调者。不过被动扫描不发送信标请求帧。设备在连接网络前会使用它。在被动扫描期间，MAC 丢弃物理层数据服务传来的所有非信标帧。

(4) 孤点扫描：所谓孤立节点，是指以前已经与某个协调者建立了连接，但现在与协调者失去同步的设备。节点在失去与其协调者的同步之后，可以利用孤点扫描尝试去重新找出其所连接的协调者。在孤点扫描期间，MAC 丢弃物理层数据服务传来的所有非协调者重新调整命令帧。

4.2 MAC 帧类型及格式

1. MAC帧类型

MAC 层协议数据单元(MPDU)以 MAC 帧(frame)的形式进行组织。MAC 层定义了 4 种类型的帧[2]：(1) 信标帧：协调者用来发送信标的帧；(2) 数据帧：用于所有数据传输的帧；(3) 确认帧：用于确认接收成功的帧；(4) MAC 命令帧：用于处理

所有 MAC 层对等实体间的控制传输。后面将分别给出 4 种帧格式的定义。限于篇幅，帧中各字段的具体含义请参考 IEEE 802.15.4 标准。

2. 通用MAC帧格式

首先给出 MAC 帧的通用格式定义，如图 4.5 所示[2]。对于不同类型的 MAC 帧，其帧头和帧尾都是一样的，只是帧的载荷部分有差异。每个 MAC 帧都具有 3 个基本部分：MAC 帧头(MAC header，MHR)、MAC 帧载荷(MAC payload，MPL)和 MAC 帧尾(MAC footer，MFR)。

<table>
<tr><td>字节：2</td><td>1</td><td>0/2</td><td>0/2/8</td><td>0/2</td><td>0/2/8</td><td>变长</td><td>2</td></tr>
<tr><td rowspan="2">帧控制</td><td rowspan="2">序列号</td><td>目的
PAN ID</td><td>目的
地址</td><td>源
PAN ID</td><td>源
地址</td><td rowspan="2">帧载荷</td><td rowspan="2">帧校验
序列</td></tr>
<tr><td colspan="4">地址域</td></tr>
<tr><td colspan="6">MHR</td><td>MPL</td><td>MFR</td></tr>
</table>

图 4.5 通用 MAC 帧格式

3. 信标帧格式

信标帧只是细化了通用 MAC 帧的帧载荷字段，如图 4.6 所示。

<table>
<tr><td>字节：2</td><td>1</td><td>2</td><td>2/8</td><td>2</td><td>1</td><td>0/1</td><td>变长</td><td>1</td><td>变长</td><td>变长</td><td>2</td></tr>
<tr><td rowspan="2">帧控制</td><td rowspan="2">序
列
号</td><td>源
PAN ID</td><td>源
地址</td><td rowspan="2">超帧
描述</td><td>GTS
描述</td><td>GTS
方向</td><td>GTS
列表</td><td>未处理
地址描述</td><td>地址
列表</td><td rowspan="2">信标
载荷</td><td rowspan="2">帧校验
序列</td></tr>
<tr><td colspan="2">地址域</td><td colspan="3">GTS 域</td><td colspan="2">未处理地址域</td></tr>
<tr><td colspan="4">MHR</td><td colspan="7">MPL</td><td>MFR</td></tr>
</table>

图 4.6 信标帧格式

4. 数据帧格式

数据帧格式与通用 MAC 帧格式完全一致，如图 4.5 所示。其帧控制字段的帧类型子字段应设置为数据帧所对应的类型，其余字段如地址域等，都应根据具体需求进行相应设置。数据帧的帧载荷字段是由上层请求 MAC 层传输的一组字节数据，如网络层数据。

5. 确认帧格式

确认帧在通用 MAC 帧格式的基础上进一步简化，是几种帧格式中最简单的，整个确认帧只有 5 字节长，如图 4.7 所示。确认帧的帧控制字段的帧类型子字段应设置为确认帧所对应的类型。

<table>
<tr><td>字节：2</td><td>1</td><td>2</td></tr>
<tr><td>帧控制</td><td>序列号</td><td>帧校验序列</td></tr>
<tr><td colspan="2">MHR</td><td>MFR</td></tr>
</table>

图 4.7 确认帧格式

注意，确认帧不存在地址域和帧载荷字段，其序列号是从需要确认的数据帧或命令帧的序列号字段中复制过来的。

6. MAC命令帧格式

MAC 命令帧只是细化了通用 MAC 帧的帧载荷字段，是几种帧格式中比较复杂的，如图 4.8 所示。MAC 命令帧对实现 MAC 层协议有重要意义，MAC 层通过各种 MAC 命令帧来实现网络的连接、断开等功能。

<table>
<tr><td>字节：2</td><td>1</td><td>0/2</td><td>0/2/8</td><td>0/2</td><td>0/2/8</td><td>1</td><td>变长</td><td>2</td></tr>
<tr><td rowspan="2">帧控制</td><td rowspan="2">序列号</td><td>目的
PAN ID</td><td>目的
地址</td><td>源
PAN ID</td><td>源
地址</td><td rowspan="2">命令帧
标识符</td><td rowspan="2">命令帧
载荷</td><td rowspan="2">帧校验
序列</td></tr>
<tr><td colspan="4">地址域</td></tr>
<tr><td colspan="6">MHR</td><td colspan="2">MPL</td><td>MFR</td></tr>
</table>

图 4.8 MAC 命令帧格式

4.3 MAC 层功能实现

4.3.1 MAC 层连接与响应过程

设备需要与协调者建立连接后，才能与网内的其它设备进行数据的收发。为建立连接，设备先经过主动或被动扫描，获得其工作范围内的协调者信息，然后选择一个协调者并向其提出连接请求，设备发起连接的详细过程如图 4.9 所示。协调者在收到设备的连接请求后会根据自身的资源情况决定是否允许设备连接并作出响应，设备成功连接后，在设备需要的情况下，协调者会给其分配一个 16 位短地址，用于以后的网内通信，协调者允许设备连接的详细过程如图 4.10 所示。

4.3.2 MAC 层数据的发送和接收

1. MAC层数据的发送

MAC 层发送数据的流程如图 4.11 所示。在 MAC 帧中，若源地址字段不存在，则认为是由协调者发送的；若目的地址字段不存在，则认为是发送给协调者的。这样做可以减少帧控制信息的开销，一定程度上降低了设备的功耗。

MAC 层在构造数据或命令帧时，会把 macDSN 的值复制到帧头的序列号字段中，然后将 macDSN 加 1。类似地，在构造信标帧时，MAC 层会把 macBSN 的值复制到帧头的序列号字段中，然后将 macBSN 加 1。

2. MAC层数据的接收

MAC 层接收数据的流程如图 4.12 所示。根据无线信道的特点，设备打开其接收机后，除接收和解析所有符合 IEEE 802.15.4 协议、工作在同一信道、处于设备无线通信范围内的、同一网络中的设备发送的信号外，也会接收到其它发射设备的信号。MAC 层应能对这些信号进行分析，滤出自己需要的帧，然后传给上层。

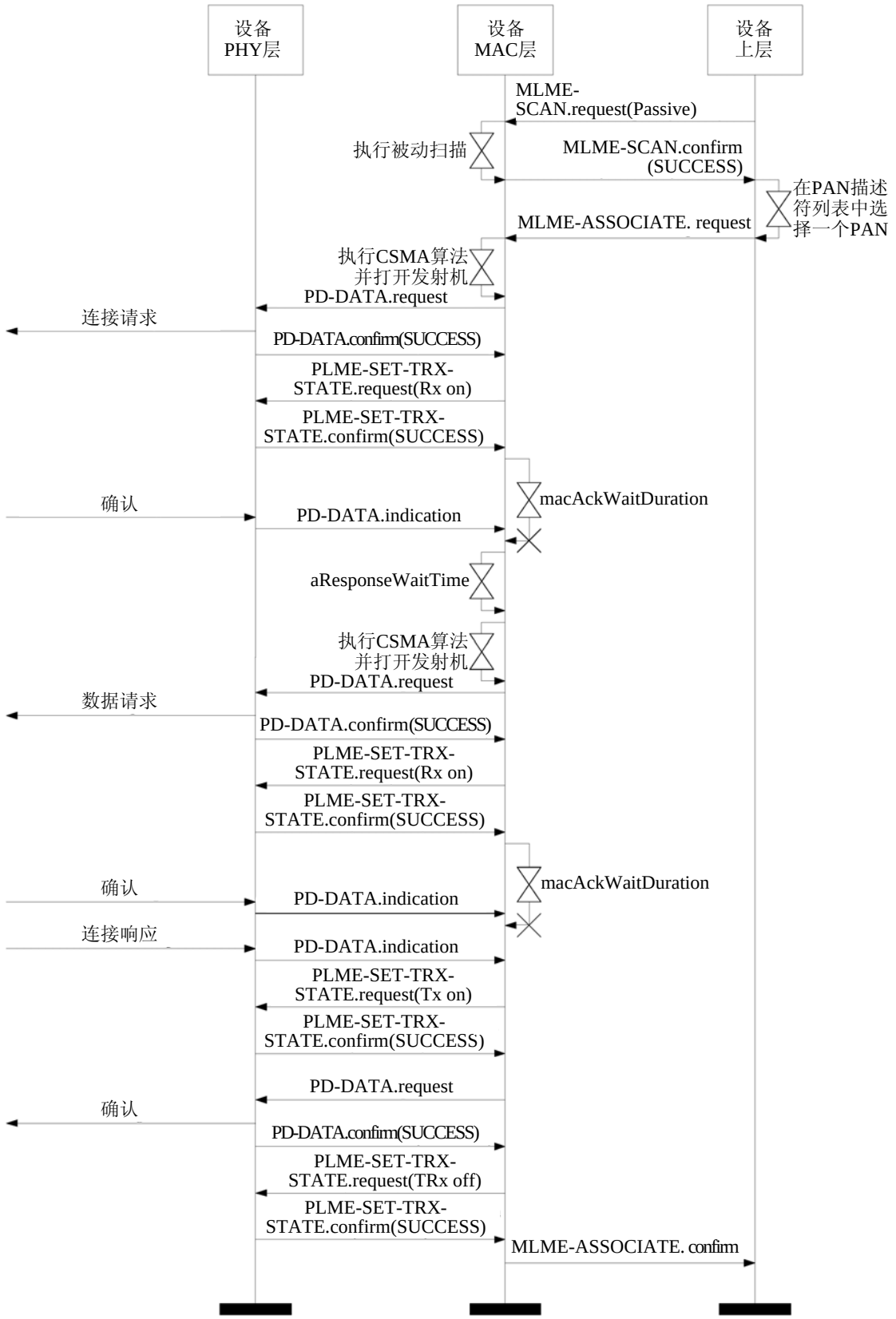


图 4.9 设备发起连接的过程

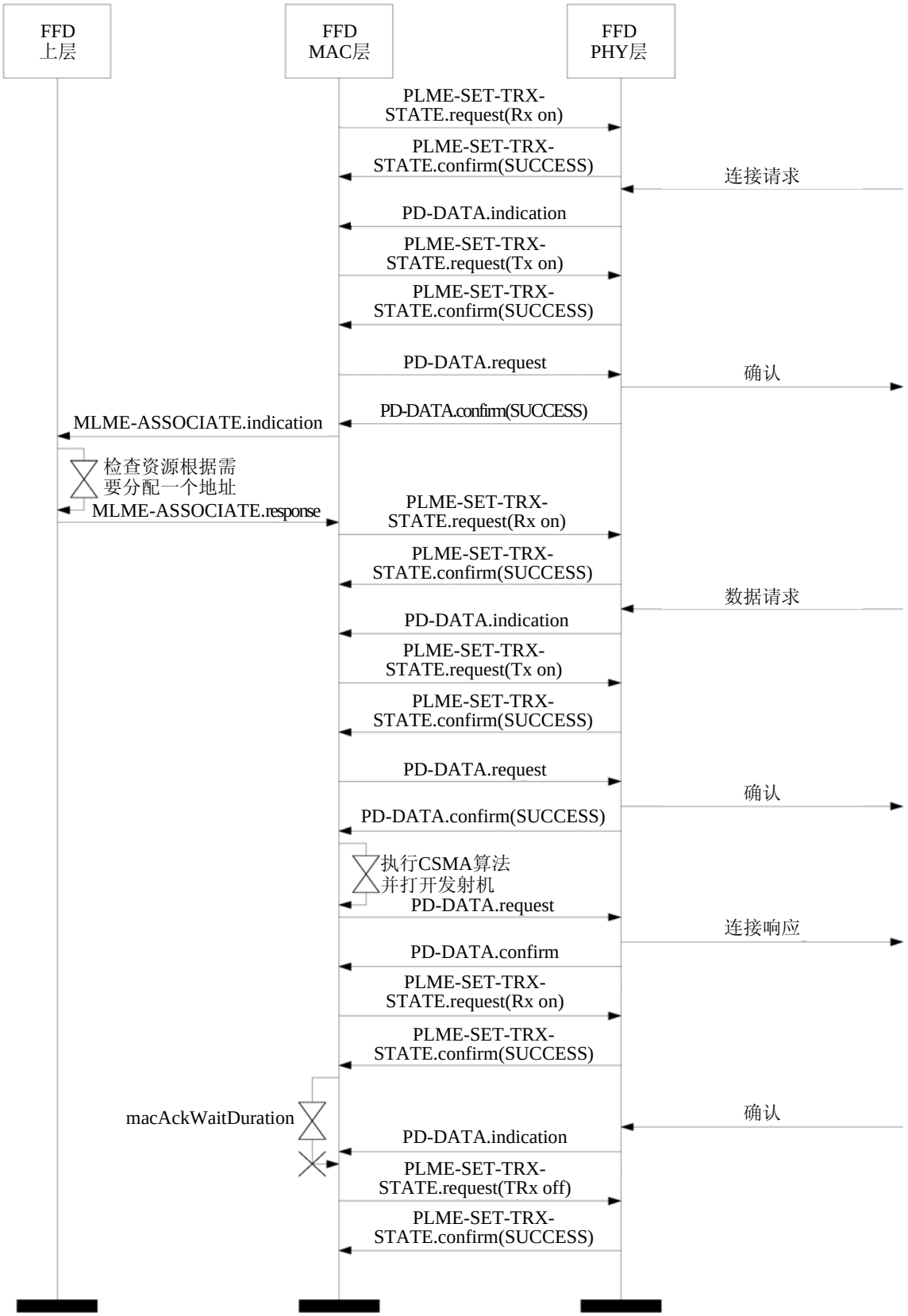


图 4.10 协调者允许连接过程

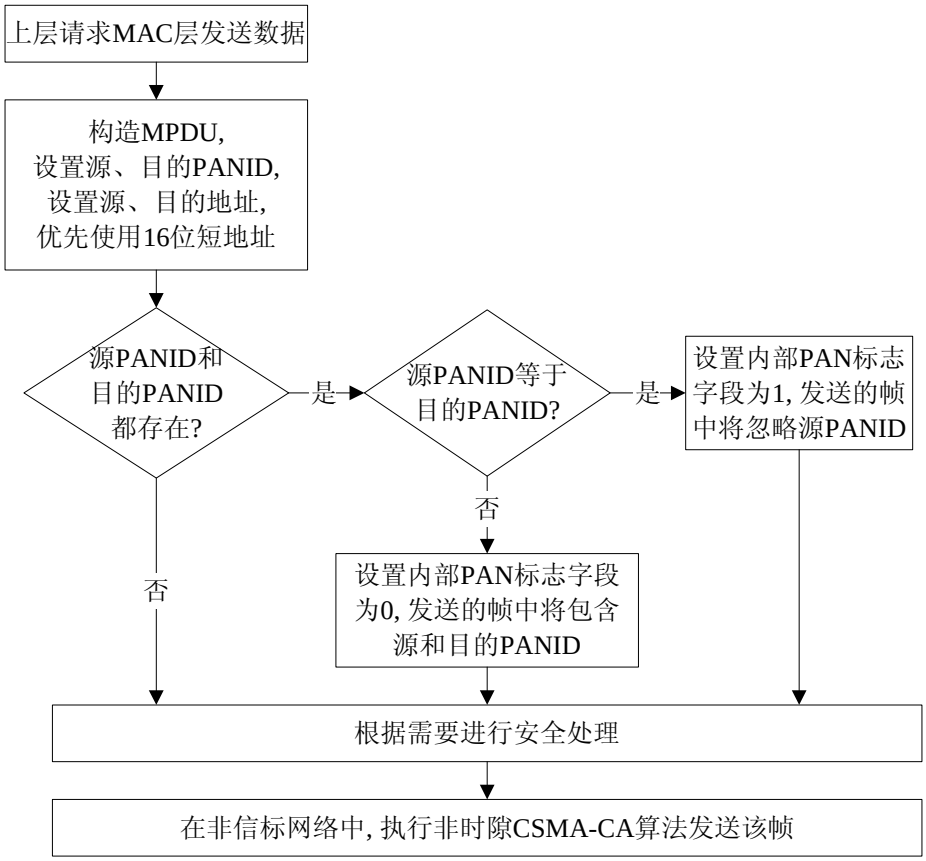


图 4.11 MAC 层数据发送流程

3. MAC层确认的使用

在发送数据帧或命令帧时，如果发送设备希望接收设备在收到有效帧后作出确认，则发送设备可将帧控制字段的请求确认子字段设为 1。这样，如果接收设备正确接收到该帧，则它将生成并发送确认帧。确认帧里的序列号应来自于所确认的数据帧或命令帧的序列号。

4.3.3 MAC 层属性的高效管理

在管理物理层属性时，由于物理层属性只有 4 个，所以直接使用 if 语句或 switch/case 语句实现即可。但 MAC 层属性多达 29 个，为实现 MAC 层的属性设置和读取原语，必须对这些属性进行一定形式的组织和管理，否则 29 个 switch/case 语句使得函数代码量非常庞大，占用大量空间，也不易于属性的维护和扩展，所以期望能够找出一种易于管理且可在常量时间内存取所有属性的方法。

虽然 MAC 层的属性的数据类型有差异，属性的取值范围也有差别，但也还是有一些规律可循。分析所有的属性可知其数据类型主要有 3 种：整型、布尔类型和字节数组。对于布尔类型的属性，只有真或假两种取值；对于整型的属性，其取值范围一般没有限制，可以取该类型所能表示的所有数值，但也有个别的整型属性只

取一定范围内的连续值或者只取离散的几个值，但是取离散值时其值个数在目前的所有属性中还没有超过 2 个的，并且这类整型属性都是 8 位无符号整型的；对于字节数组类型的属性主要就是一些节点地址的信息。最主要的是所有属性的属性标识符都是连续的，目前是从 0x40～0x55 以及 0x70～0x76。

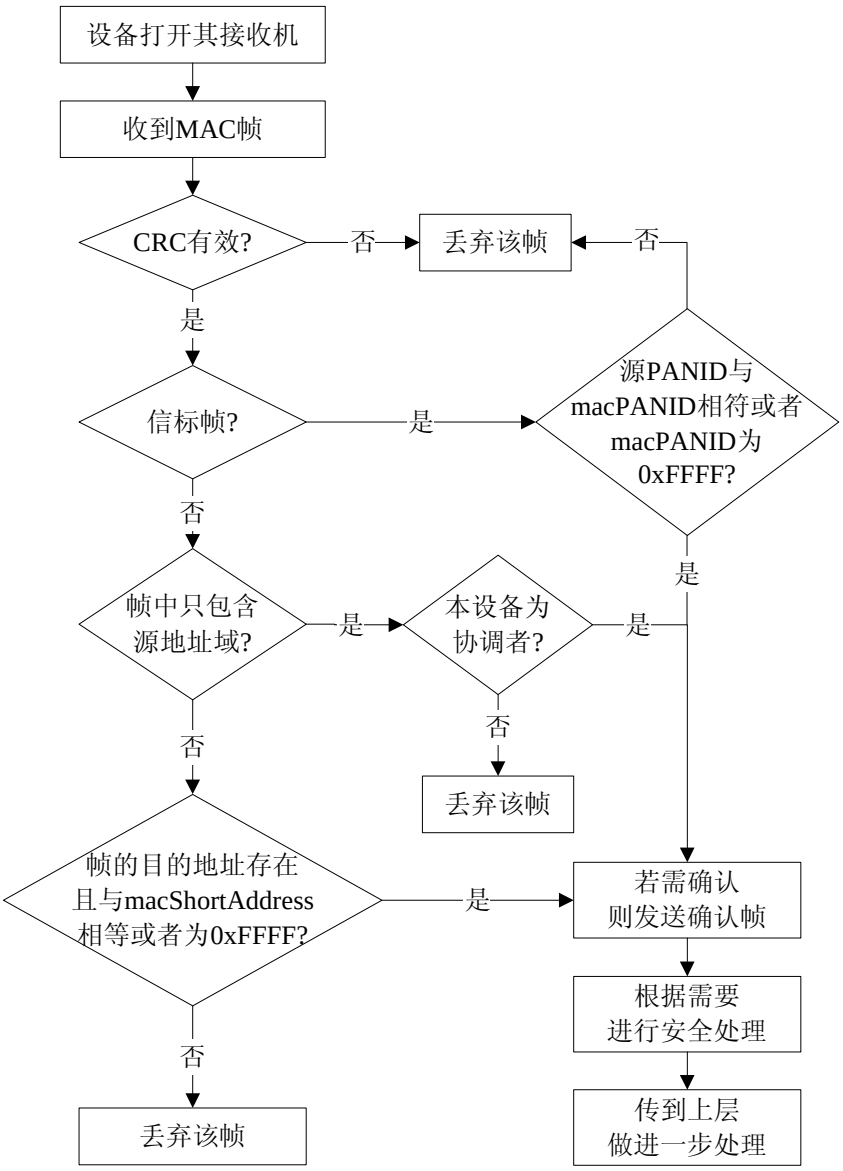


图 4.12 MAC 层数据接收流程

基于以上分析出的规律，一种直接的想法就是利用属性标识符连续这一特点，构造一个指针数组，数组的每个元素依次指向每个属性的开始地址，并在程序初始化时完成指针数组的各元素与 MAC 层各属性间的关联关系，然后利用查表的思想，把属性标识符转换为该属性在指针数组中的下标，这样就解决了在常量时间内存取任何属性的问题，但是属性的取值范围以及属性的类型如何得知呢？

这两个问题可以通过以下方法解决。首先，可以通过增加一个 8 位无符号整型

变量 m_nType 来记录属性的类型，这样也就知道了属性的字节长度，并定义规则如下：若 m_nType 的最高位为 1，表示是字节数组类型的属性，同时其低 7 位的值代表了字节个数；否则，即为非字节数组类型，在 MAC 层属性目前的定义中，只能是整型或布尔类型。对于非字节数组类型，目前的属性中都是当做无符号处理的，所以再定义如下规则：0x00 表示布尔类型；0x01~0x03 分别表示 8、16、32 位无符号整型；0x04~0x06 分别表示指向 8、16、32 位无符号整型的指针。这样就基本解决了属性类型的识别问题。其次，属性的取值范围可以通过增加两个 8 位无符号整型变量 m_nMax 和 m_nMin 来记录属性的最大值和最小值，并且定义如下规则：若 m_nMax=m_nMin=0，表示该属性取值无限制；若 m_nMax<m_nMin，表示该属性取离散值；若 m_nMax≥m_nMin，则表示该属性取值有一定的限制。这样定义的属性取值范围的规则，完全可以解决设置 MAC 属性值时的非法赋值检测的问题。

最终，使用了一个结构体数组来记录 MAC 层所有属性的地址及类型信息，并且保存在 FLASH 中(属性本身还是定义在 RAM 中的)，以节省 RAM 空间，相应结构体类型的定义如下：

```
#define NumberOfMACPIB          29
typedef struct MACPIBEntry_tag
{
    INT8U m_nType;
    INT8U m_nMax;
    INT8U m_nMin;
    void *m_pPIBValue;
} MACPIBEntry_t;
const MACPIBEntry_t MACPIB[NumberOfMACPIB];
```

4.3.4 MAC 命令帧的实现

MAC 命令帧的实现实际上就是根据提供的入口参数，按照相应的命令帧格式逐字段组装成帧。为降低协议实现的复杂度，在本协议栈中只实现了与非信标网络相关的命令，如连接请求、连接响应、数据请求、信标请求等，限于篇幅，有关的 MAC 命令帧及其格式请参考 IEEE 802.15.4 标准。

4.4 MAC 层测试

MAC 层的测试主要集中在测试信标帧和 MAC 命令帧的帧结构组织是否正确上。通常，比较方便的做法是使用与以太网中常用的抓包工具类似的专业工具来辅助分析帧结构，这需要购买配套的硬件以及抓包软件，目前市场上也有一些 ZigBee 相关厂商提供这些工具，但其价格一般不易接受。

在开始实现 MAC 层功能时是使用串口调试工具的 16 进制显示功能，把接收到的帧按照其帧格式的详细定义进行逐位的对比分析，这是一件相当费时和需要耐心的工作。后来随着实现的命令帧的增加，测试工作量也变得很大，于是编写了如图 4.13 所示的逐字段分解 MAC 帧结构的测试程序，大大提高了测试效率，减少了后面的测试工作量。

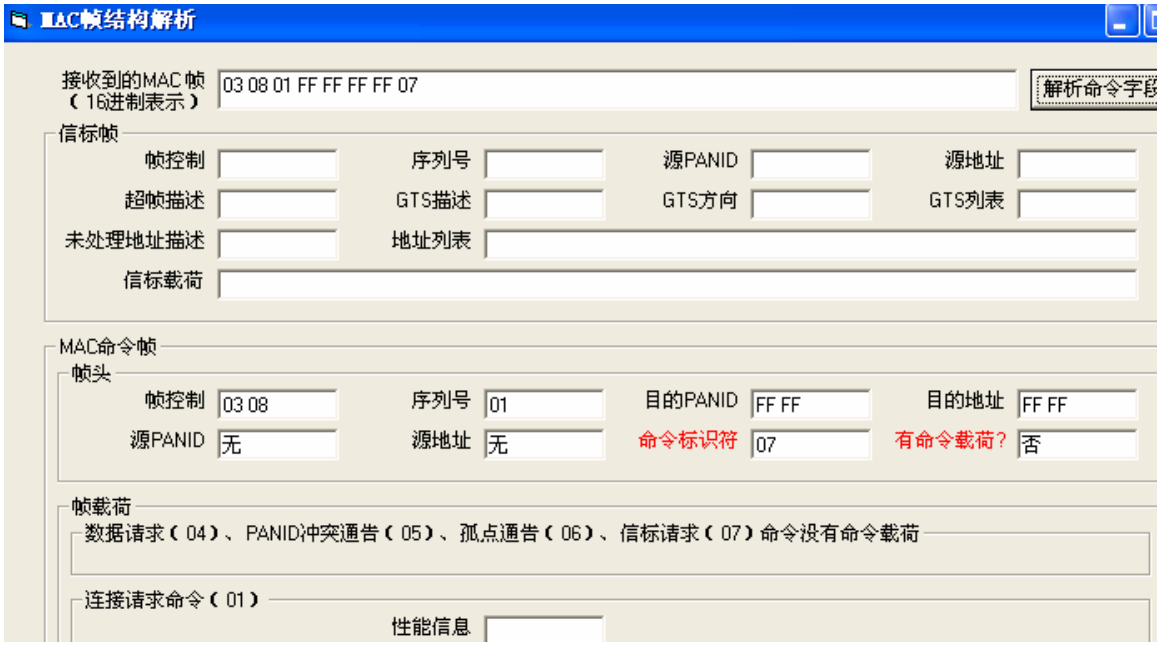


图 4.13 MAC 层帧结构解析程序

4.5 本章小结

本章概括了 MAC 层的基本功能和几个基本概念，接着给出了 MAC 帧类型及格式定义。详细描述了设备发起连接和协调者响应连接的流程，给出了数据收发的基本步骤以及 MAC 层确认的使用。针对 MAC 层属性的特有规律，提出了一种在常量时间内存取 MAC 层任一属性的易于维护和扩展的方法。最后在物理层协议的基础上实现了基于非信标网络的 ZigBee MAC 层协议并进行了测试。

第五章 ZigBee 网络层的研究与实现

网络层位于 MAC 层与应用层之间，它通过正确操作 MAC 层提供的功能来向应用层提供合适的服务接口。本章概括了网络层的基本功能和相关概念，给出了网络帧类型及格式定义。分析评价了无线自组网几种常用的路由协议，阐述了分布式地址分配机制以及树状层次路由和 AODVjr 路由的实现过程。最后在 MAC 层协议的基础上实现了 ZigBee 网络层协议并进行了相应的测试。

5.1 ZigBee 网络层概述

5.1.1 网络层简介

在 ZigBee 网络中，由于受能量限制，节点之间无法直接通信，通常需要借助中间节点以多跳路由的方式将源数据传送至目的节点。ZigBee 网络层主要负责路由的发现和维护，主要包括两个方面的功能：一是路径选择，即寻找源节点到目的节点的优化路径；另一个是数据转发，即将数据沿优化路径正确转发。

5.1.2 网络层模型和服务原语

1. 网络层模型

ZigBee 网络层的参考模型如图 5.1 所示。

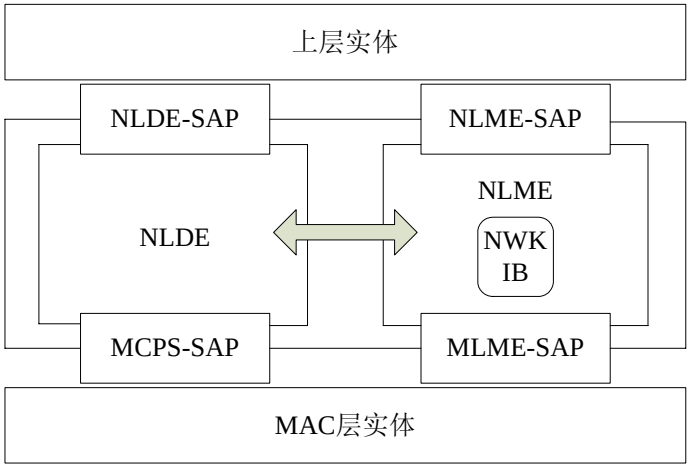


图 5.1 网络层参考模型

其中，MCPS-SAP 是 MAC 层提供给网络层的数据服务接口；MLME-SAP 是 MAC 层提供给网络层的管理服务接口；NLME-SAP(Network Layer Management Entity-Service Access Point) 是网络层提供给应用层的管理服务接口；NLDE-SAP(Network Layer Data Entity-Service Access Point)是网络层提供给应用层的

数据服务接口。NLDE-SAP 用于将应用层提供的数据组装成网络层协议数据单元(Network layer Protocol Data Unit，NPDU)，并将其传输给目的节点的网络层；或者将接收到的网络层协议数据单元进行解包，并将解包后得到的数据传送给本节点的应用层，也就是说 NLDE-SAP 实现两个对等应用层之间的数据传输。

2. 网络层原语

网络层主要完成以下工作：NLDE 应具有产生 NPDU 的能力；能针对具体网络拓扑结构进行路由选择；可以执行一些安全性操作等。NLME 应具有根据操作需要充分地配置协议栈的能力；可以建立一个新网络或加入和离开一个网络；可以为网络中的设备分配地址；能够执行邻居发现和路由发现操作等。关于网络层原语的详细内容请参考 ZigBee 协议规范。

5.1.3 ZigBee 的网络结构和通信方式

1. ZigBee定义的网络拓扑结构

ZigBee 网络层支持星型(star)、树状(tree)和网状(mesh)拓扑结构。树状和网状网络本质上就是 IEEE 802.15.4 中所支持的点对点网络。

在星型网络中，网络由一个称为 ZC 的设备控制，ZC 负责启动网络和维护网络中的设备，其它的所有设备都只直接与 ZC 通信。星型网络的覆盖范围和通信距离都比较小。在网状和树状网络中，ZC 负责启动网络并选择关键的网络参数，并且网络覆盖范围可以通过使用 ZR 进行扩展。树状网络可以用于 IEEE 802.15.4 中所描述的信标网络中。在树状网络中，ZR 使用层次路由策略(hierarchical routing strategy)来传输网络中的数据和控制信息，这种路由算法非常简单，只有两种选择，要么路由给其父节点，要么路由给其一个子节点，但是如果某一节点出现故障，其下的所有子节点与外部其他节点的通信就会中断，所以在实际应用中需重点考虑树状网络的鲁棒性。网状网络支持完整的点对点通信，并且具有自组织(self organizing)和自恢复(self healing)能力，所以网络的稳定性和可扩展性都非常好。

ZigBee 协议规范规定，网状网络中的 ZR 不应该发射 IEEE 802.15.4 中所定义的信标。本协议栈中以网状网络为设计目标。

2. ZigBee网络的通信方式

ZigBee 网络层定义了 3 种通信方式：单播(unicast)、广播(broadcast)和多播(multicast)。单播数据只传送给网络中的某个单一设备。广播数据传送给网络中的每个设备。多播数据传送给网络中的某个组内的所有设备，也称为组播。组是 ZigBee 网络层中的一个概念。一个组由一个或多个设备组成，每个组都用一个 16 位的组标识符(Group ID)进行唯一区分。在 ZigBee 网络层中，可以以组为单位进行寻址和数据

收发操作，这样就可以在一次操作中控制多个设备(即组内的所有成员)。本文实现的协议栈中暂未提供多播通信方式。

5.1.4 网络帧类型及格式

1. 网络帧类型

网络层协议数据单元(NPDU)以网络帧(frame)的形式进行组织。网络层定义了 2 种类型的帧[11]：数据帧和网络命令帧。后面会分别给出 2 种帧格式的详细定义。限于篇幅，帧中各字段的具体含义请参考 ZigBee 协议规范。

2. 通用网络帧格式

通用网络帧格式如图 5.2 所示[11]。对于不同类型的帧，其帧头都是一样的，只是帧的载荷部分有差异。每个网络帧都具有 2 个基本部分：网络帧头(NWK header，NHR)、网络帧载荷(NWK payload，NPL)。

<table>
<tr><td>字节：2</td><td>2</td><td>2</td><td>1</td><td>1</td><td>0/8</td><td>0/8</td><td>0/1</td><td>变长</td><td>变长</td></tr>
<tr><td rowspan="2">帧控制</td><td>目的
地址</td><td>源
地址</td><td rowspan="2">半径</td><td>序列
号</td><td>目的
IEEE 地址</td><td>源
IEEE 地址</td><td>多播
控制</td><td>源路由
子帧</td><td rowspan="2">帧载荷</td></tr>
<tr><td colspan="7">路由域</td></tr>
<tr><td colspan="9">NHR</td><td>NPL</td></tr>
</table>

图 5.2 通用网络帧格式

3. 网络命令帧格式

网络命令帧格式与通用网络帧格式基本一致，只是细化了帧载荷字段的具体内容，如图 5.3 所示，网络命令标识符字段区分了不同的网络层命令。

字节：2	变长	1	变长
帧控制	路由域	网络命令标识符	网络命令载荷
NHR		NPL	

图 5.3 网络命令帧格式

4. 数据帧格式

数据帧格式与通用网络帧格式完全一样。数据帧的帧控制字段的帧类型子字段应设置为数据帧所对应的类型，其余字段应根据具体需求进行相应设置。数据帧的路由域应包含适当的地址字段和广播字段的组合，这取决于帧控制字段的设置。数据帧的帧载荷字段是由上层请求网络层传输的一组字节数据。

5.2 无线自组网常用路由协议

ZigBee 网络作为一种自组织和自恢复的网络，具有多跳、动态拓扑、资源受限等特点，为路由算法的设计带来很大挑战。多年来，研究者已提出数十种自组网路由

协议方案，可从不同角度对这些协议进行分类[44-45]：根据路由建立时机与数据发送的关系，可分为表驱动路由(table driven protocols)、按需驱动路由(on-demand driven protocols)和混和路由；根据节点在路由过程中是否有层次结构、作用是否有差异，可分为平面路由(flat protocols)和层次路由(hierarchical protocols)；根据是否使用 GPS 定位系统作为路由辅助条件，可分为地理定位辅助路由和非地理定位辅助路由。以上几种协议间的关系如图 5.4 所示。

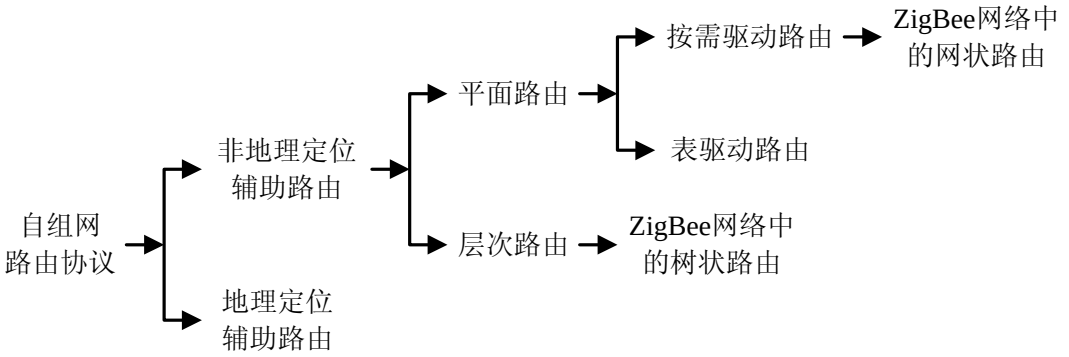


图 5.4 路由协议分类

5.2.1 表驱动路由协议

表驱动路由协议又称为主动式或先应式路由协议，是一种基于路由表的路由协议。在表驱动路由协议中，节点间通过周期性地交换路由信息来更新自身路由表，以便及时反映网络拓扑结构的变化。其最大优点是当节点间有通信请求时，能够立即获得所需路由。其主要缺点是维护路由表及更新路由会带来巨大的网络开销。下面介绍几种常见的表驱动路由协议。

1. DSDV

DSDV(Destination-Sequenced Distance-Vector)协议是一个基于传统的 Bellman-Ford 算法的路由选择协议。该协议用目的节点序列号来避免使用过时路由信息而产生无效路由，通过对路由编号等措施来避免路由环路的产生。DSDV 的主要优点是消除了路由环路，加快了收敛速度，同时减少了控制信息的开销。它的不足在于周期性的广播分组增大了网络开销，难以适应拓扑变化快的移动自组网[46]。

2. OLSR

OLSR(Optimized Link State Routing)协议采用和 DSDV 类似的逐跳路由，数据包不指明完整的路径，由各中间节点根据自己的路由表选择路径，各节点需要有规律的交换网络拓扑信息。该协议的优点在于：它只广播部分链路状态信息，且只在选出的部分节点中广播，节约了有限的网络资源[47]。

5.2.2 按需驱动路由协议

按需路由协议也称为反应式路由协议。与表驱动路由协议不同的是，按需驱动路由协议不需周期性地进行路由更新，仅当源节点有通信需要但又没有去往目的节点的路由时，才洪泛(flooding)一个路由请求分组来创建路由。其最大优点是无需周期性地广播路由信息，节省了有限的网络资源。其缺点在于等待路由的创建会造成通信延迟。下面介绍几种常见的按需驱动路由协议。

1. DSR

DSR(Dynamic Source Routing)协议使用源路由算法，网络中每个节点需要维护一个路由表，当发现新的路由时会更新该表。每个路由分组头都包含该分组从源节点到目的节点途中所经过的中间节点信息，各节点缓存路由分组头中携带的路径信息。DSR 的优点在于：使用缓存技术减少了路由发现的开销；一次路由发现过程可以产生多条到达目的节点的路由，有助于路由选择。DSR 的缺点在于：每个分组的头部都需携带完整的路由信息，数据包的额外开销较大；由于路由被缓存，过期路由会影响路由选择的准确性[48]。

2. AODV

AODV(Ad Hoc On-Demand Distance Vector)路由协议使用基于目的节点的路由发现机制。源节点广播路由请求分组，收到路由请求分组的节点创建到达源节点的反向路由。目的节点收到路由请求后回复路由应答分组，回复分组以单播方式向源节点传播，沿途建立到目的节点的正向路由[49]。它采用了 DSDV 的逐跳路由和顺序编号机制，同时借鉴了 DSR 的路由发现和路由维护机制。

AODV 协议主要具有以下优点[50-51]：通过引入序列号避免了出现路由环路；支持中间节点应答，能使源节点快速获得路由，有效减少了广播分组数；节点只存储需要的路由，减少了内存需求；能快速响应活跃路径上的断链；具有良好的可扩展性。AODV 协议也还存在着以下不足[52]：没有路由安全保护机制；需要相对较长的路由建立时延；需要周期性地广播 HELLO 分组以维持路由，而这会消耗一定的能量和带宽。

3. AODVjr

AODVjr(AODV junior)是 AODV 协议的简化版本。AODVjr 移除了 AODV 中的序列号(sequence numbers)和跳数(hop count)，只允许由目的节点对最先到达的路由请求做出应答，而不像 AODV 中知道到达目的节点路径的中间节点也可以作出应答。通过这种端到端(end-to-end)的策略，AODV 中的 HELLO 信包(HELLO messages)、路由错误(Route Error，RERR)信包以及前驱列表(precursor lists)在 AODVjr 中也无需考

虑，这使得路由发现更为节能高效[29-30]。

从文献[29]中对 AODVjr 和 AODV 的对比分析可以看出，二者具有几乎相同的性能，但是 AODVjr 相对更加简单，且易于理解和实现。而且在路由维护过程中，AODVjr 的控制包的开销远远小于 AODV，图 5.5 中给出了二者控制包的开销对比[29]。所以，在本协议栈的路由协议实现中，使用了 AODVjr 与树状层次路由相结合的方式。

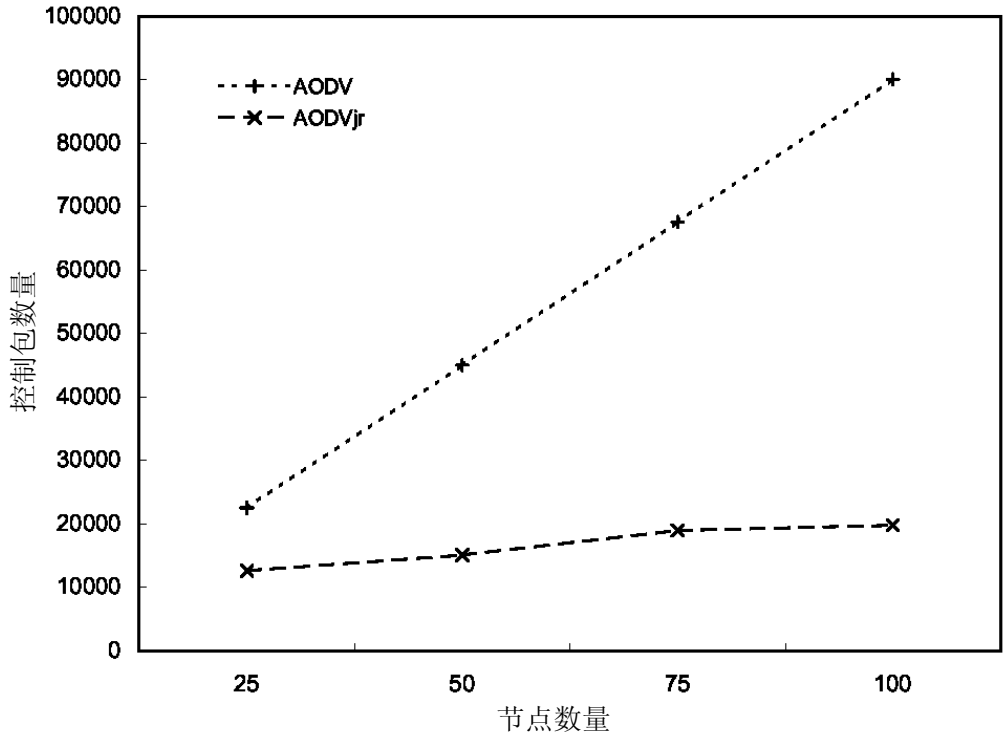


图 5.5 AODVjr 与 AODV 的控制包开销对比

5.3 网络层功能实现

5.3.1 网络的建立

ZigBee 网络只能由具有协调者能力的设备来实现，该设备在建立网络后会将其自身初始化为一个 ZigBee 协调者(ZC)，同一个 ZigBee 网络中只能有且仅有一个 ZC。ZigBee 网络的建立流程如图 5.6 所示。

上层通过 NLME-NETWORK-FORMATION.request 原语来启动建立新网络的过程。启动之后，NLME 首先执行能量检测扫描来选择可接受的信道，然后会在所有可接受的信道上执行主动扫描来选择准备在其上建立新网络的信道，然后 NLME 会为新网络选择一个小于 0x3fff 的 PANID，并设置自己的 16 位网络地址为 0x0000，然后检查并初始化 nwkExtendedPANID 属性的值。最后 NLME 使用 MAC 层的 MLME-START.request 原语来启动 PAN 并向上层通知建立网络的状态。

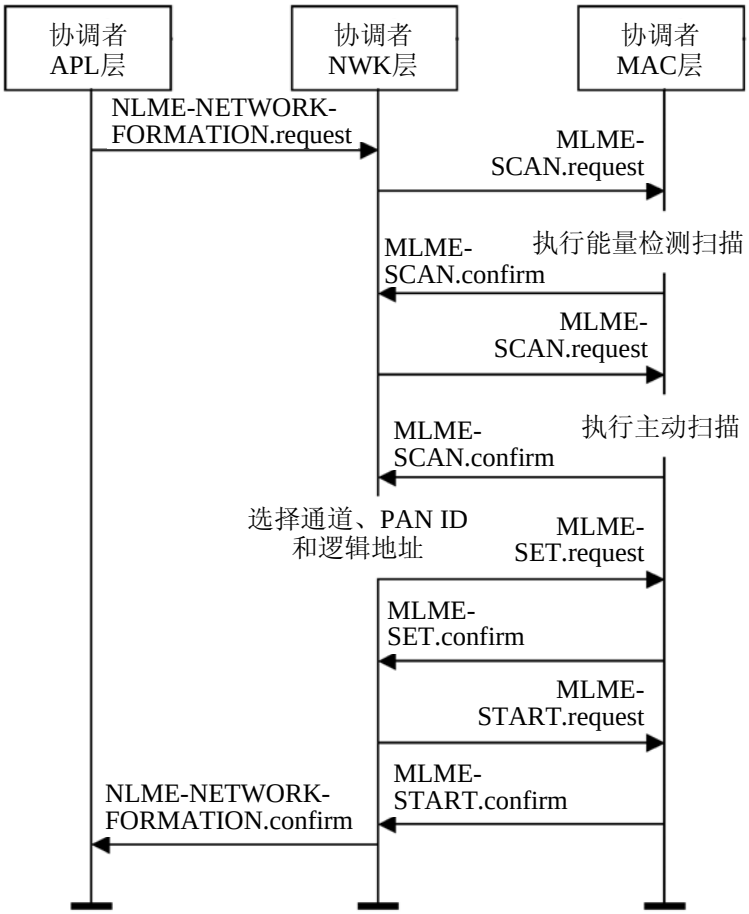


图 5.6 建立一个新网络的流程

5.3.2 节点加入网络

当处在网络内的一个设备允许一个新设备加入网络时，这两个设备就构成了父子关系。新加入的设备是子设备，第一个设备是父设备。一个子设备可以通过下面两种方式加入网络：通过 MAC 层的连接(association)过程加入网络，或者由先前指定的父设备直接加入网络。另外，一个孤点设备可以通过孤点方式加入或重新加入网络。限于篇幅，这里仅给出节点正常入网时所用的 MAC 层连接的过程，其余的方式请参考 ZigBee 协议规范。

1. MAC层连接加入网络时的子设备流程

通过 MAC 层连接加入网络时，子设备的一般流程如图 5.7 所示。

子设备通过向网络层发送 NLME-NETWORK-DISCOVERY.request 原语来启动 MAC 层连接加入网络的过程。启动之后，网络层首先会请求 MAC 层执行主动或被动扫描，然后从得到的信标信息中选择一个合适的网络执行 MAC 层的连接加入过程。若子设备加入网络成功，父设备会为其分配一个 16 位短地址用于之后的网内通信。如果设备是一个路由器，可能还会根据需要来设置超帧配置并在要求的时候开始发送

信标帧。

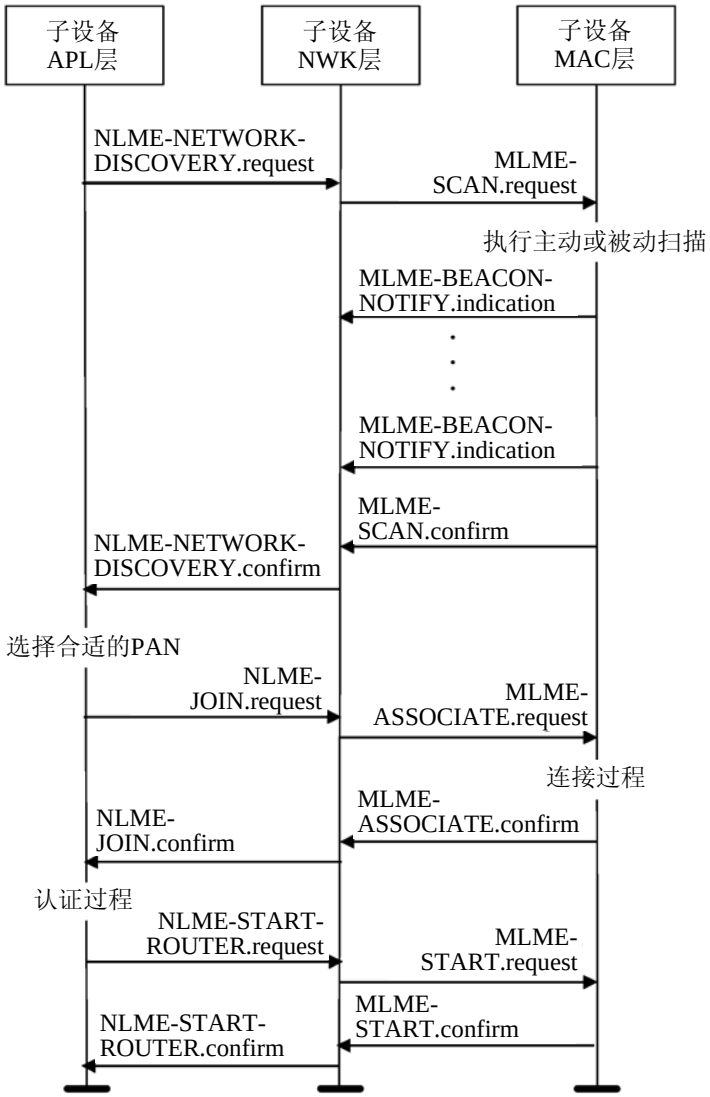


图 5.7 MAC 层连接加入网络时的子设备流程

2. MAC层连接加入网络时的父设备流程

通过 MAC 层连接加入网络时，父设备的流程如图 5.8 所示。只有 ZC 和 ZR 能够允许设备加入网络，而 ZED 则不能。

ZC 或 ZR 通过 MAC 层连接将一个设备加入到网络的过程是利用 MAC 层的 MLME-ASSOCIATE.indication 原语来初始化的。父设备的 NLME 会根据请求设备的 64 位扩展地址判断该设备是否已经在网络中，若已存在则获得原先的 16 位短地址，否则为新设备分配一个 16 位短地址，然后向 MAC 层发送连接响应。若父设备接受了新设备的入网请求，NLME 将在邻居表中为新加入的子设备增加一条新记录以保存设备相关信息。同时，NLME 会向上层发送 NLME-JOIN.indication 原语，告知有一个新设备加入到网络中。

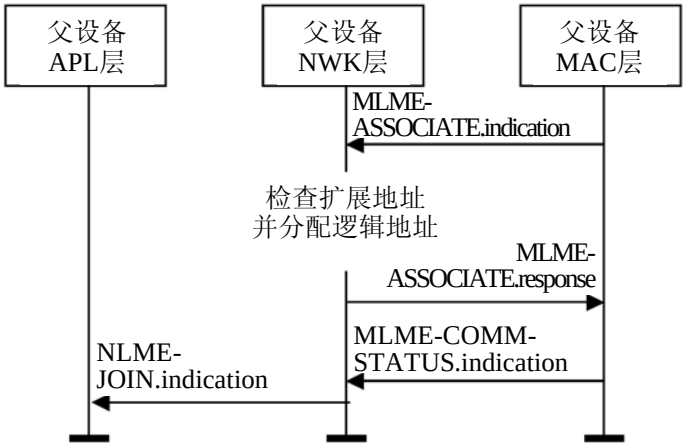


图 5.8 MAC 层连接加入网络时的父设备流程

5.3.3 网络地址分配机制

在 ZigBee 网络层中，采用的是分布式地址分配机制(distributed address assignment mechanism)，即每个父设备(只能是 ZC 或 ZR)拥有一个有限的网络地址段，并可以从这些地址段中选择一些地址分配给其子设备。

ZigBee 网络中的每个设备都有一个连接深度，定义为该设备到 ZC 的最小跳数，其中 ZC 自己的连接深度为 0，其直接子设备的连接深度为 1，其余设备的连接深度大于 1，ZC 决定网络的最大深度。

假如一个父设备所能拥有的最大子设备数量为 nwkMaxChildren(C_m)，其中所允许的最大路由器子设备数量为 nwkMaxRouters(R_m)；网络的最大深度为 nwkMaxDepth(L_m)，则函数 Cskip 是父设备所拥有的地址数量，其公式为：

$$\mathrm{Cskip}(d)=\begin{cases}1+C_m\cdot(L_m-d-1) & (R_m=1\text{时})\\ \dfrac{1+C_m-R_m-C_m\cdot R_m^{L_m-d-1}}{1-R_m} & (R_m>1\text{时})\end{cases} \tag{5.1}$$

其中，d 为父设备在网络中的连接深度。若一个设备的 Cskip(d)≤0，则表示该设备没有接受子设备连接的能力。

在父设备为子设备分配地址时，若该子设备是路由器，则会给其分配一组地址；否则为该终端子设备分配仅一个地址。假设父设备自己的地址为 A_{parent}，则它为具有路由能力的子设备分配地址的公式为：

$$A_n=\begin{cases}A_{parent}+1 & (n=1\text{时})\\ A_{n-1}+\mathrm{Cskip}(d) & (n>1\text{时})\end{cases} \tag{5.2}$$

其中，A_n 是第 n 个路由器子设备的地址且 $1\leqslant n\leqslant R_m$。

父设备的终端子设备的地址紧跟在最后一台路由器的后面安排，地址分配公式

为：

$$A_n = A_{parent} + Cskip(d) \cdot R_m + n \tag{5.3}$$

其中，A_n 是第 n 个终端子设备的地址且 $1 \leqslant n \leqslant C_m - R_m$。

5.3.4 网络层数据的发送和接收

1. 网络层数据的发送

只有已经与网络连接的设备才可以从网络层发送数据。网络层数据的发送非常简单，所有的网络帧都应包含一个目的地址和源地址，网络层发送的所有帧，包括数据帧和网络命令帧，都要按照前面介绍的帧格式进行组装。当构造好一个网络帧，并且已经准备好发送该帧时，网络层会将 NPDU 传送给 MAC 层数据服务实体，并请求 MAC 层发送 NPDU。MAC 层会把数据发送的结果返回给网络层。网络层发送数据的流程如图 5.9 所示。

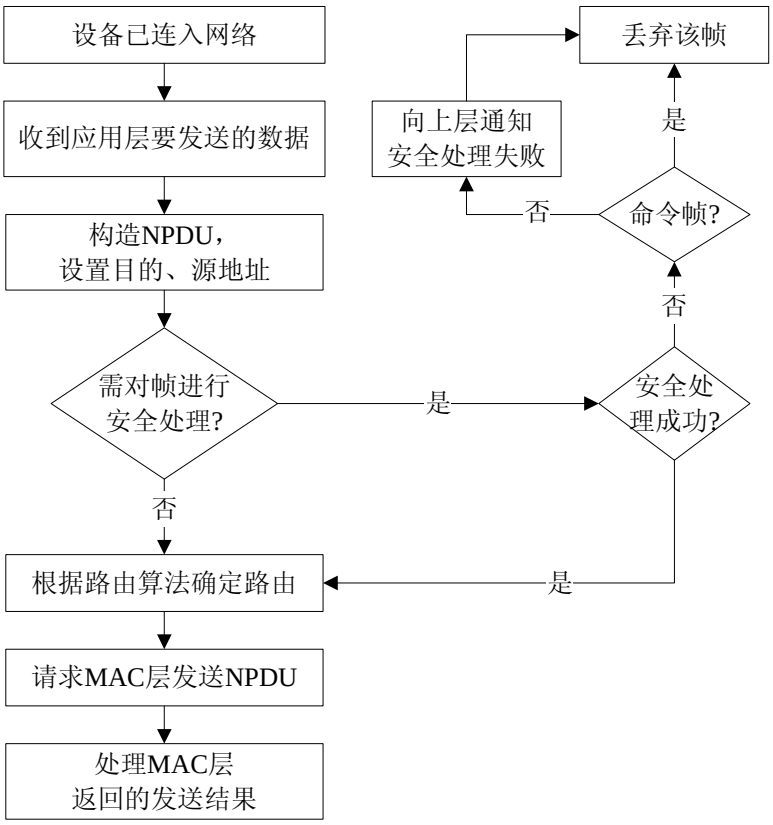


图 5.9 网络层数据发送流程

2. 网络层数据的接收

为接收数据，设备必须打开其接收机。当然，设备可以使用间接数据传输过程来完成数据的接收。在信标网络中，每个设备都会跟踪其父设备的信标，可以通过间接数据传输来获得自己的数据。而在非信标网络中，设备可以使用数据请求命令轮询其

父设备以获得自己的数据。

本协议栈中使用直接数据传输。在协议中设置了所有设备在空闲时打开其接收机，这样每个设备都可以使用直接数据传输方式获得自己的数据帧。不过这也增加了设备的功耗，在今后协议栈的升级中，可以予以改进。

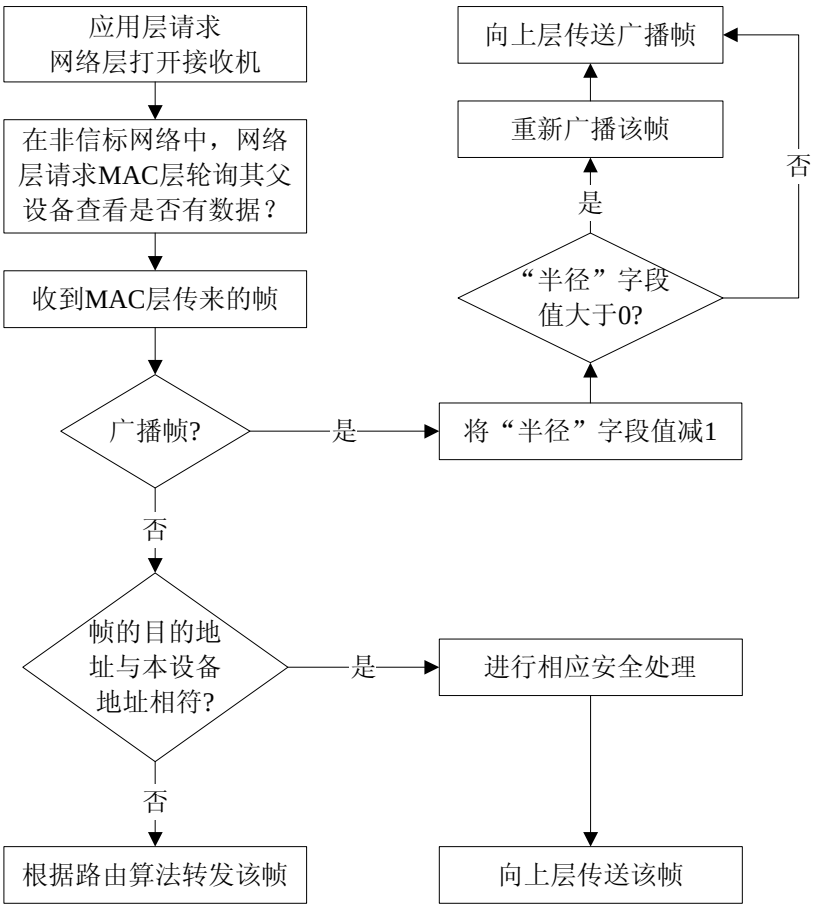


图 5.10 网络层数据接收流程

网络层数据的接收流程如图 5.10 所示。当收到 MAC 层传来的数据帧后，网络层会对帧进行一些合法性检查，查看该帧的目的地址和设备自己的网络地址是否相符合，若地址相符，则该帧会被传送到上层进行处理；若地址不符，并且设备具有路由能力，则设备网络层会查询路由表，选择合适的下一跳，进行数据转发工作。转发过程中，如果不存在到达目的地址的下一跳路由，则还可能启动路由发现过程。若该帧不是发往本设备，并且本设备也没有路由能力，则会使用树状层次路由算法进行数据转发。

5.3.5 网络命令帧的实现

网络命令帧的实现实际上就是根据提供的入口参数，按照相应的命令帧格式逐字段组装成帧，这里不作详细介绍。本协议栈中仅用到了路由请求和应答命令，关于相

关命令的格式请参考 ZigBee 协议规范。

5.3.6 网络层路由功能的实现

1. ZigBee的基本路由算法

当设备网络层从 MAC 层或其上层接收到一个单播数据帧时，将按照图 5.11 给出的 ZigBee 基本路由算法为帧安排路由。具有路由能力的设备应查找与该帧的目的地址对应的路由记录并转发该帧或在不存在相应记录的情况下启动路由发现过程。对于没有路由能力的设备，若设备的 NIB 属性 nwkUseTreeRouting 等于 TRUE，设备将采用层次路由沿着树转发帧。

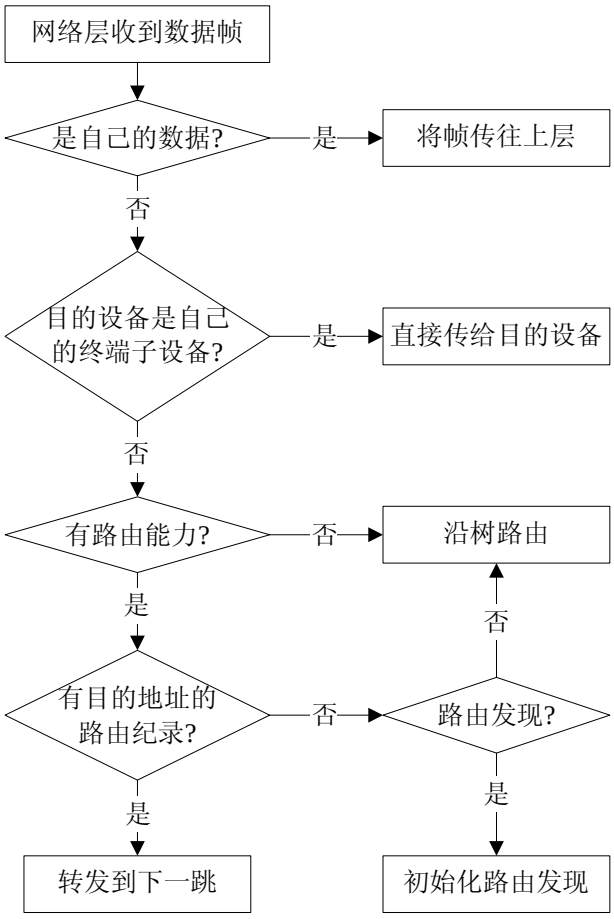


图 5.11 ZigBee 的基本路由算法

2. 树状层次路由算法

在树状层次路由算法中，若帧的目的设备是当前设备的后代设备，则设备就把帧转发给适当的子设备，否则就把该帧转发给其父设备。接下来讲述了如何判定本设备的后代设备以及如何确定层次路由算法的下一跳地址。

ZigBee 网络中除 ZC 之外的每一个设备都是 ZC 的后代设备，而任何一个 ZED 都没有后代设备。对一个地址为 A、深度为 d 的 ZR，如果下面的逻辑表达式成立，

那么地址为 D 的目的设备就是该 ZR 的后代设备：

$$A < D < A + Cskip(d-1) \tag{5.4}$$

当确定帧的目的设备是当前接收设备的后代时，那么下一跳地址 N 为：

$$N = \begin{cases} D & (\text{子设备为ZED，即} D > A + R_m \cdot Cskip(d)\text{时}) \\ A + 1 + \left\lfloor \dfrac{D-(A+1)}{Cskip(d)} \right\rfloor \cdot Cskip(d) & (\text{子设备为ZR}) \end{cases} \tag{5.5}$$

3. 路由表结构

ZR 或 ZC 维护了一个路由表以记录和维护到达某些目的设备的路由信息，其存储的信息格式如表 5.1 所示。ZR 或 ZC 还可能预留一些路由表条目专用于路由修复或在其它路由能力都耗尽的时候才使用。

表 5.1 路由表条目格式

字段名	字段长度	描述
目的地址	2 字节	16 位的网络地址或组标识符。若目的设备是 ZR 或 ZC，则该字段包含了实际的网络地址；若目的设备是 ZED，则包含其父设备的网络地址。
状态	3 位	路由状态。
多对一	1 位	标志目的设备是一个多对一路由请求的集中器。
需要路由记录	1 位	标志在发送下一个数据包之前需要发送一个路由记录命令帧。
组标识符标志	1 位	标志目的地址是一个组标识符(Group ID)。
下一跳地址	2 字节	去往目的地址的下一跳的 16 位网络地址。

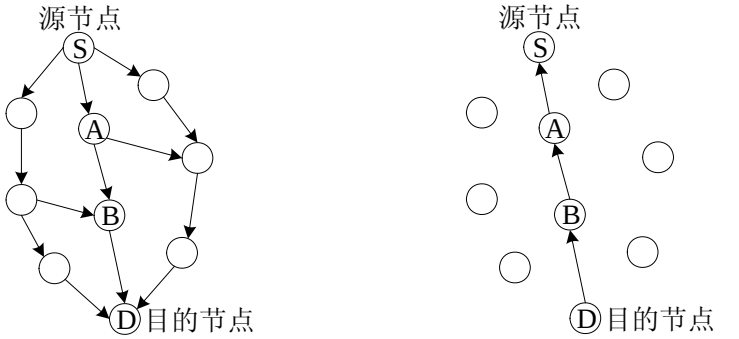
4. AODVjr的路由发现过程

路由发现是网络中的设备相互配合，发现并建立到达某一目的设备的路由的过程。路由发现总是针对特定的源设备和目的设备执行的。

AODVjr 算法是需求驱动型的，由源节点选择路由。考虑到节能、成本、应用方便性等因素，AODVjr 简化了 AODV 协议的一些特点，只用到了路由请求(Route Request，RREQ)命令和路由应答(Route Reply，RREP)命令，去掉了 AODV 中的序列号、中间节点回复、周期性的 HELLO 信包、路由错误(Route Error，RERR)命令以及前驱列表等[29,49]，但是仍然保留了 AODV 的主要功能。在 AODVjr 中，由于不存在序列号，所以只允许目的节点才能回复 RREP 命令，这样可以避免路由循环问题和无效 RREP 信包的出现，提高了通信效率。图 5.12 给出了 AODVjr 的路由发现过程[29,53]。

当源节点 S 要发送数据给目的节点 D 时，如果发现自己没有到目的节点 D 的路由，便通过网络层广播 RREQ 命令帧，请求其邻居帮忙查找到目的节点 D 的路径。每个接收到 RREQ 命令帧的节点，都维护一条到源节点 S 的路由信息，同时帮助源

节点 S 广播 RREQ 命令帧。通过这种洪泛方式，RREQ 命令帧最终会被转发至目的节点 D。目的节点 D 接收到 RREQ 命令帧后，根据 RREQ 命令帧的路由代价决定是否更新自己的路由表，同时使用路由代价最小的路径给源节点 S 回复 RREP 命令帧。本协议栈中使用 RREQ 命令帧的到达时间作为路由代价，选择最先到达的那条路径作为代价最小的路径。



(a) 源节点 S 广播 RREQ (b) 目的节点 D 单播 RREP

图 5.12 AODVjr 的路由发现过程

注意，源节点查找目的节点是通过广播 RREQ 进行的，在这个过程中建立了目的节点到源节点的反向路由；而目的节点给源节点回复 RREP 则是通过单播进行的，此时建立了源节点到目标节点的正向路由。正向路由建立之后，源节点就可以向目的节点发送数据信息了。

在路由维护时，正向路由在源节点发送数据时得以维持，如果目的节点在一段时间内都不需要给源节点发送数据，它可以选择向源节点发送一个连接信息包来维持反向路由。当源节点在一段时间内没有接收到目的节点发来的任何数据时，它就认为此条路径失效，必要时可以重新进行路由发现。

5.4 网络层测试

网络层的测试与 MAC 层类似，主要是要查看网络层接收到的数据帧的内容是否正确，以及在路由发现过程中路由表和路由发现表中的记录是否存在以及是否正确。由于协议栈中只实现了路由请求命令和路由应答命令，所以只是利用串口按一定的格式输出具体要查看的内容，然后对照网络层命令格式检查是否正确。

另外，对于网络建立是否成功、子设备是否成功加入到网络、设备当前的状态等等都有相应的状态指示灯来表明相应操作的结果。

5.5 本章小结

本章概括了网络层的参考模型和基本功能，总结了网络层的拓扑结构和常用通信

方式，给出了网络帧类型及其格式定义，实现了网络新建和节点加入、数据发送和接收等网络层功能。概括了无线自组网常见的路由协议，阐述了路由表的结构和分布式地址分配机制以及树状层次路由算法。实现了适用于 ZigBee 网状网络的原理简单易于理解但功能只是稍许减弱的 AODVjr 路由算法，并对网络层的功能进行了一定的测试。

第六章 ZigBee 在车间设备监控系统中的应用

本章概括了设备管理的必要性和现存系统存在的一些缺陷，设计和实现了基于 ZigBee 的车间设备监控系统中的设备监控节点和数据管理中心两个功能模块，并深入分析了 ZigBee 节点的低功耗策略。

6.1 系统概述及构成

6.1.1 设备管理的必要性

在复杂的经济环境和激烈的市场竞争条件下，越来越多的企业为了充分利用信息资源，提高管理水平和竞争能力，建立起车间设备监控系统[54]。

设备投入生产后，其价值能否充分和持续发挥，能否为企业生产出合格的产品，能否持续地节约原材料、能源和维修费用以降低生产成本，能否实现安全生产，满足环保要求；能否达到设备寿命周期费用最经济，设备综合效率最高，取得设备投资的效益，都依赖于对设备的管理[55]。

对企业来说，通过设备监控系统可以获得设备的当前工作环境和运行状况以及设备的相关历史数据，以此来为企业中不同层次的人员提供管理、维护以及决策支持等信息，对保证设备安全正常运行，充分发挥设备效能，提高企业经济效益有着极其重要的作用。因此，很有必要在企业内部搭建合理的设备监控系统。

6.1.2 现存系统的缺陷及 ZigBee 技术的优势

目前已经有许多企业构建了自己的车间设备监控系统，但是，它们大多数是利用 RS485 总线或工业以太网等有线方式组织实现的，这种有线方式存在一些缺点：系统的耗材多、造价高、功耗大；扩展能力差；设计、施工与维护复杂，安装时需穿墙打孔、预埋线路，布线繁琐；由于采用硬线连接，线路的老化、腐蚀、磨损及各种非自然损坏等问题严重，不易排查和修理；而且对于未安装监控系统的老厂房和老建筑，其施工更不方便[56-57]。

ZigBee 无线通信技术避免了有线方式的缺点，特别适合于此类监控应用，其优势在于：(1) 便于安装和重新布置，节约大量布线成本以及老系统改造成本，大大节约了安装时间。(2) 可扩充性好。节点可以固定安装，也可以灵活移动安装。在增加或删除节点、节点位置发生变动或节点发生故障等时，网络都能够自我修复，无需人工干预，保证整个系统仍然能正常工作。(3) 网络容量大。ZigBee 可采用具有强大出错自恢复能力的网状网络结构，最多可组成 65535 个节点的大网[11]。本系统中即采用

网状网络结构。(4) 可靠性高。硬件上采用扩频通信技术，在协议方面使用了握手确认机制和多次重发机制，并且通过 CRC 校验来保证数据传输的正确性。(5) 可双向传输数据信息及控制命令。不但可以从监控节点传出数据，而且双向通信能够将控制命令传送到与监控节点相连接的传感器和控制执行器件，这样就可以实现远程监测和控制设备。

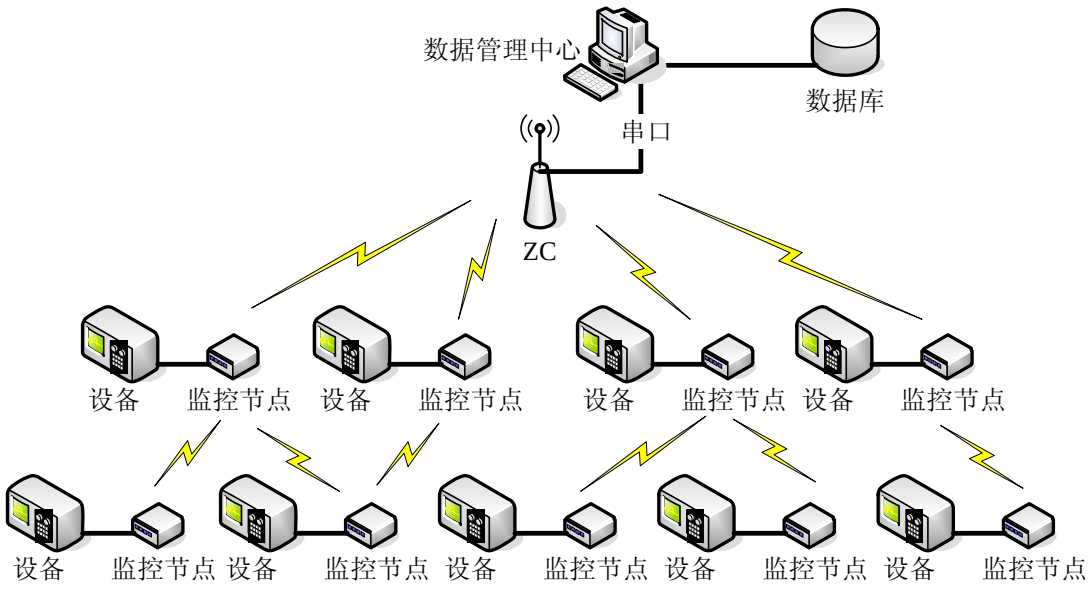


图 6.1 基于 ZigBee 的车间设备监控系统的整体结构

6.1.3 基于 ZigBee 的车间设备监控系统的整体结构

本章以车间设备监控系统为例，阐述如何将前文实现的 ZigBee 协议栈应用于车间设备监控系统中，同时可以验证自制 ZigBee 软硬件平台的可行性和正确性。

基于 ZigBee 的车间设备监控系统的整体结构如图 6.1 所示，包括设备监控节点和数据管理中心两个部分。系统中使用了网状拓扑结构，以保证网络强大的自我修复能力。限于篇幅，这里对设备信息采集及控制过程不作详细描述。

6.2 设备监控节点

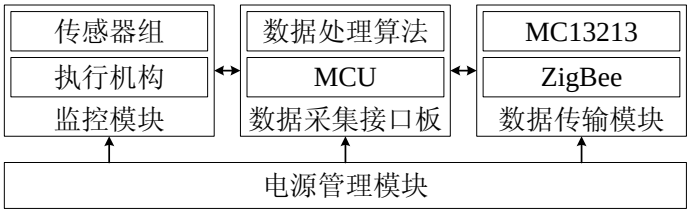


图 6.2 设备监控节点的结构

设备监控节点通过合适的方式如串口等与具体设备相连，用于采集设备的工作环境和运行状态信息并发送给数据管理中心，或者接收数据管理中心的控制命令并执行

相应的动作。设备监控节点的结构如图 6.2 所示，包括监控模块、数据采集接口板、数据传输模块和电源管理模块 4 个部分。

考虑到系统今后的扩展和升级，因此以 Freescale 的 MC9S08AW60(后文称 AW60)芯片为核心制作了一块数据采集接口板。AW60 是首个能支持 5V 且基于高性能 HCS08 核的系列成员，内部总线频率可达 20MHz，通用 I/O 引脚多达 56 个，是 Freescale 目前主推的 8 位 MCU。它包含众多有价值的特性：60K 的 FLASH、2K 的 RAM、16 通道的 10 位模数转换器、串行外设接口、内部集成电路总线、2 个串行通信模块、2 个共 8 通道的 16 位定时器/脉宽调制器模块以及键盘中断模块等[58]。AW60 丰富的片上资源和对外接口，较适合于实现数据采集接口板。图 6.3 给出了最终实现的数据采集接口板的功能框图，可满足较多种类和数量的数据采集需求。

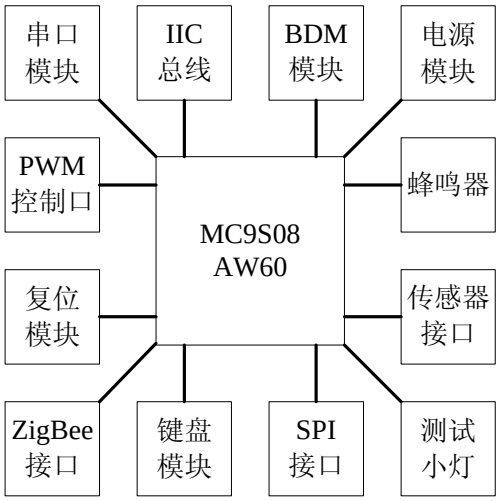


图 6.3 数据采集接口板的功能模块框图

设备监控节点中使用的数据传输模块主要是 ZR 和 ZED。其中，ZR 主要用于转发数据和扩展 ZigBee 网络的覆盖范围；而 ZED 仅用于发送采集数据或接收数据管理中心的命令，不参与数据转发。为降低成本，在保证网络可靠性和稳定性的条件下，应尽可能多的使用 ZED 节点。

6.3 数据管理中心

6.3.1 功能概述

数据管理中心通过与之相连的 ZC 来接收、处理、存储、显示各个设备监控节点的采集数据或者向它们发送配置和命令信息。它以图形化的方式显示了整个系统的结构，允许用户远程管理和维护各个设备监控节点，并提供了相应的查询和操作界面，如查询某个节点的实时数据或历史数据、设置采集周期等。

为把协议栈用于设备监控系统中，在协议栈网络层之上实现了一个简单的应用程序，以实现节点的自动组网与自动入网过程，减少了用户的工作量。

6.3.2 管理功能的实现

1. 系统的图形化显示与操作界面

数据管理中心软件的操作界面如图 6.4 所示。图中的“C”、“R”、“E”分别表示

该设备监控节点中使用的 ZigBee 节点的类型是 ZC、ZR、ZED，并用不同的颜色进行了区分。当鼠标右键单击每个节点时，会有一个适用于该节点的右键菜单，其中有针对该节点的各种功能性操作子菜单，如设备信息查询和修改、设备实时数据查询、设备历史数据查询、向该设备发送相关的控制命令等。当鼠标悬停在每个节点上方时，会有一个提示框显示该节点的简单信息。

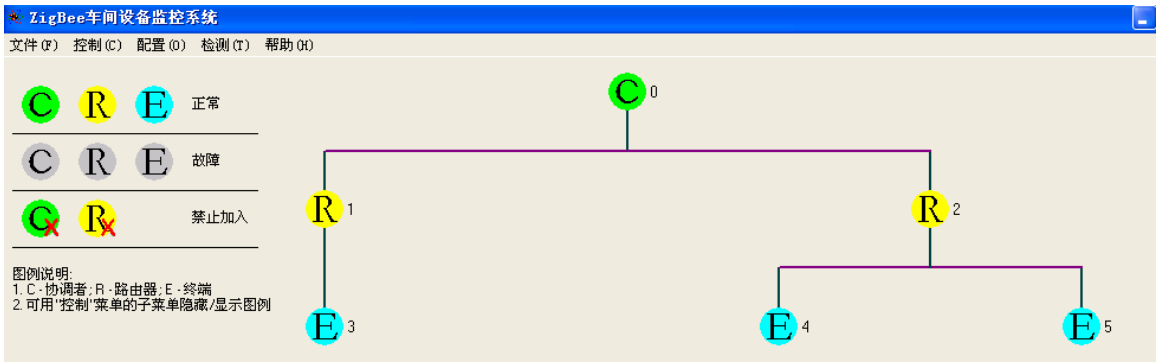


图 6.4　数据管理中心的软件操作界面

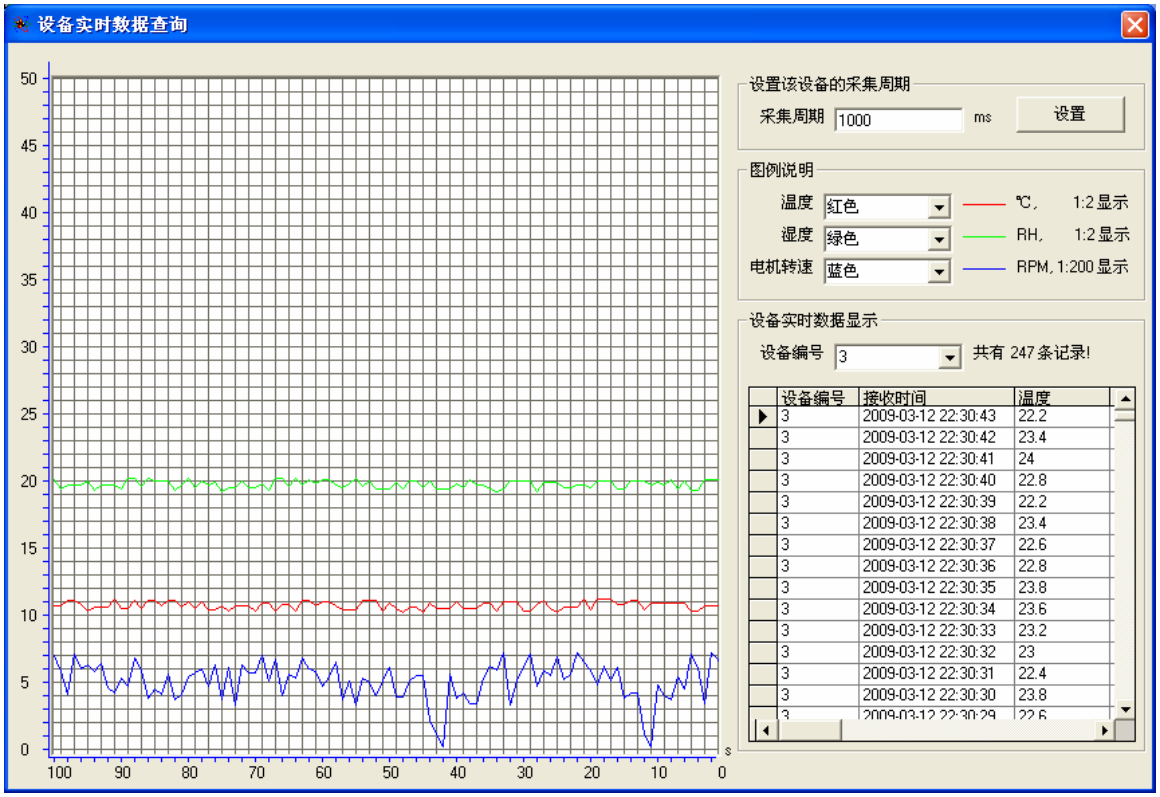


图 6.5　设备实时数据查询界面

2. 设备实时数据的查询

数据管理中心可以实时查询和显示单个设备监控节点的数据，如图 6.5 所示，这里仅测量和显示了温度、湿度和电机转速等 3 个物理量，分别用不同的颜色进行区分，并各自使用不同的比例尺使得可以在同一个界面中进行显示。

由于采用了专门的数据采集接口板，所以可以由用户自己决定采集哪些物理量，并可以根据需要添加新的物理量采集传感器。由于 ZigBee 网络具有自组织和自恢复特性，设备监控节点可以在网络的覆盖范围内随意移动，所以用户可以自由决定采集哪些设备的信息，只需把设备监控节点与欲进行监控的设备相连即可。整个系统具有一定的可定制性和较好的可扩展性。

3. 设备历史数据的查询

图 6.6 设备历史数据查询界面

数据管理中心同时也提供了查询和显示所有或某个设备监控节点的历史数据的功能，如图 6.6 所示，可以按照温湿度范围、时间段以及设备编号等多种条件查询设备的历史数据信息，以掌握设备的工作环境和运行状态，为更好地管理和维护设备提供相关的决策支持信息。

6.4 低功耗考虑

本系统中的 ZigBee 节点都采用电池供电，这样就无需考虑监控节点的供电问题，最大限度地减少了对原有设备和工作环境的影响和破坏。但由于是电池供电，所以不可避免的要考虑到电池的使用寿命。在非信标网络中，ZED 节点并不参与数据转发

工作，这里主要分析 ZED 节点的低功耗。当设备监控节点的数据采集周期较长(后面以 10s 为例进行讨论)时，没有必要让 ZED 节点一直工作在运行模式。

MC13213 的 MCU 和 Modem 模块都提供了相应的低功耗模式。表 6.1 给出了它们在各种工作模式下的典型电流值[42,59]。经实际测量，自制硬件的电流基本符合表中所列的典型值。

表 6.1 MC13213 各种工作模式下的电流典型值

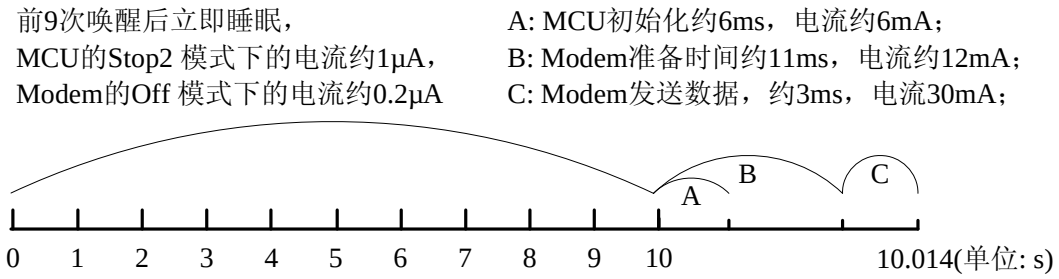
模块	工作模式		典型电流值 (室温，VDD=3V)
MCU	低功耗模式	Stop1	25 nA
		Stop2	550 nA
		Stop2(启用 RTI，内部时钟源)	850 nA
		Stop3	675 nA
		Stop3(启用 RTI，内部时钟源)	675 + 300 = 975 nA
		Stop3(启用 RTI，外部时钟源)	675 + 5000 = 5675 nA
		Wait(fBus = 1 MHz)	560 μA
	运行模式	Run(fBus = 1 MHz)	1.1 mA
		Run(fBus = 8 MHz)	6.5 mA
Modem	低功耗模式	Off	0.2 μA
		Hibernate	1 μA
		Doze(禁止 CLKO)	35 μA
	运行模式	Idle	500 μA
		CCA/ED	< 37 mA
		RX	< 37 mA
		TX	< 30 mA

对 MCU 而言，虽然 Stop1 模式功耗最低，但此模式下的所有寄存器和 RAM 内容都会丢失，且只能通过上电复位才能退出该模式，不满足节点定时唤醒的需求。在系统中选择了带实时中断(Real-Time Interrupt，RTI)的 Stop2 模式，通过周期性 RTI 中断来唤醒 MCU。MC13213 的 RTI 模块在使用内部时钟源时，其产生周期性中断的最长间隔约为 1024ms，所以 ZED 节点累计被唤醒 10 次后才真正运行并发送数据，前 9 次都是唤醒后就立即再次进入低功耗。

对于 Modem 而言，在空闲时可以选择 Doze 模式或 Off 模式。选择 Doze 模式的优势是其退出低功耗的恢复时间很短，只需约 300μs 即可；而退出 Off 模式则需要约 10ms～25ms 的恢复时间[42]。

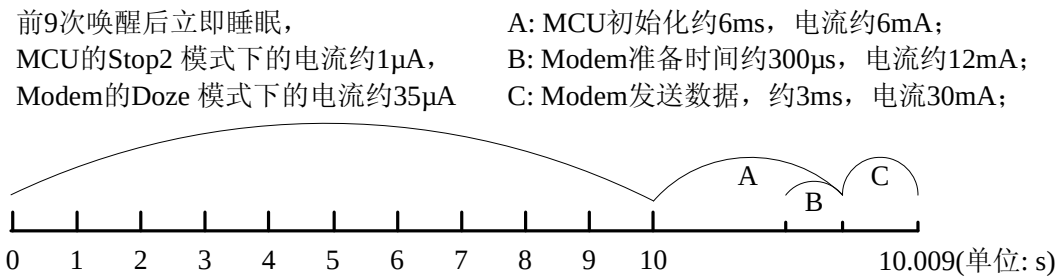
图 6.7 分别给出了 MCU 使用 Stop2 模式、Modem 分别使用 Off 和 Doze 模式时，ZED 节点在 10s 内的工作时间及电流分布情况。图中的工作时间是利用定时器模块实际测量代码的执行时间而得到的，并且 Modem 是使用 0dBm 的发射功率。可以看出

方案 1 具有比较大的优势。



在共计约10s内的功耗为：$10000*(1+0.2)*10^{-3}+(11+3)*6+11*12+3*30=318mA.ms$

(a) 方案 1：MCU 使用 Stop2 模式，Modem 使用 Off 模式



在共计10s内的功耗为：$10000*(1+35)*10^{-3}+(6+3)*6+0.3*12+3*30=507.6mA.ms$

(b) 方案 2：MCU 使用 Stop2 模式，Modem 使用 Doze 模式

图 6.7 两种低功耗方案的对比

在方案 1 条件下，若用 900mAh(毫安时)容量的电池供电，则电池的理论使用时间为((900*60*60*1000)/318)*10 秒，约(((900*60*60*1000)/318)*10)/(24*60*60)=1179 天，即 3.231 年。再次分析方案 1，发现在 MCU 初始化完成后，会等待 Modem 直到其正常工作，其实此时 MCU 完全可以再进入 Wait 低功耗模式，这样又可以进一步降低约 30mA.ms 的功耗，电池使用时间约为 3.567 年。当然上述计算只是理论上的，考虑到漏电、信道访问冲突引起的数据重发等因素，实际的使用时间可能会短一些，但也在可接受的范围之内。

6.5 本章小结

本章基于自制的 ZigBee 软硬件平台，给出了一个基于 ZigBee 的车间设备监控系统的应用实例，设计和实现了设备监控节点和数据管理中心两个功能模块，给出了数据管理中心的操作界面以及设备实时和历史数据的查询界面，并深入分析了 ZED 节点的低功耗策略，基本满足了实际的应用需求。

第七章 总结与展望

7.1 总结

本文基于 Freescale 的 MC13213 芯片设计和实现了自己的单芯片 ZigBee 硬件平台，然后在此基础上研究并实现了一个以 ZigBee-2006 协议规范为基准的经过一定裁剪的使用非信标网络的 ZigBee 协议栈。

本文的主要工作总结如下：

(1) 收集分析和归纳整理了有关 ZigBee 的历史、现状以及最新进展等方面的文献资料和数据，对 ZigBee 技术有了比较清楚的认识和了解。

(2) 深入研读了 IEEE 802.15.4 标准和 ZigBee 协议规范中的相关章节，从整体和细节上把握了 ZigBee 协议各层的脉络，并对相关的辅助性技术和知识进行了一定的了解，为后面的协议实现提供了相应的理论基础。

(3) 对比分析了目前存在的各种 ZigBee 硬件方案，以比较有代表性的 MC13213 为基础构建了自己的单芯片 ZigBee 硬件平台，阐述了各个模块的原理和实现中的注意事项，详细分析和实现了 ZigBee 硬件中的各个模块。

(4) 基于上述的 ZigBee 硬件，自底向上逐层实现了一个以 ZigBee-2006 协议规范为基准的协议栈，并在各层实现过程中分别进行了详细的测试。概括和分析了物理层协议，并对在实现物理层协议过程中所遇到的 SPI 事务操作等问题进行了深入剖析，找出了问题的深层次原因。MAC 层协议功能的实现在很多地方上都是针对 MAC 帧展开的，在 MAC 层协议实现过程中，提出了一种可高效管理 MAC 层众多属性的方法，具有较好的可维护性和可扩展性。路由转发是网络层的重要功能，在分析了 ZigBee 协议规范中的路由协议并结合无线自组网的现有路由协议的基础上，实现了针对网状网络的原理简单但功能并不弱的 AODVjr 协议。另外，由于本协议栈中采用了分布式地址分配机制，所以也支持树状层次路由。

(5) 在上述软硬件平台上，以车间设备监控系统为应用对象，给出了一个具体的应用实例，验证了软硬件平台的可行性和正确性，并深入分析了 ZED 节点的低功耗问题。

7.2 展望

下一步要做的重点工作就是测试协议栈的稳定性和可靠性。目前由于只是少量节点间的小数据量的数据通信测试，所以协议栈中可能还存在一些隐藏的问题，这需要

今后更进一步的测试。

本协议栈中只实现了在实际中可能要更为常用的非信标网络。虽然信标网络的网络规模不能很大(否则网络内各节点间的同步将会成为问题)，但对功耗要求非常严格的小规模网络的应用场合，一般可以使用信标网络和超帧结构，这样网络中的节点都可以在空闲时进入睡眠状态从而降低功耗。所以接下来可以考虑添加这部分功能。

为减少工作量和保证协议栈实现的清晰性，本协议栈中并没有实现各层中与安全方面相关的功能，而安全处理功能在实际应用中可能会有所需求，无线通信的特性使得安全和保密性尤为重要。这也是今后增强协议栈功能的一个方面。

另外，研究和推进 ZigBee 网络与其它现存标准网络间的互通和融合问题也是一个比较实际的热点问题。如通过实现一体化网关，完成 ZigBee 到其它标准协议间的协议转换，以接入到其它有线网络，毕竟一般情况下有线网络会更可靠。

参考文献

[1] 华中田.与蜂共舞——ZigBee 技术一瞥[J].电子产品世界,2007,(11):72-80.
[2] IEEE.Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks[S].IEEE Std 802.15.4-2003,October 2003.
[3] 蒋挺,赵成林.紫蜂技术及其应用[M].北京邮电大学出版社,2006.
[4] 闫富松,赵军辉,李秀萍.ZigBee 技术及其应用[J].广东通信技术,2006,26(4):48-51.
[5] 施炯.移动设备中 ZigBee 接口的实现[EB/OL].http://www.winbile.net/bbs/forums/threads/1037493.aspx,2009.
[6] 勾淑婉.Zigbee 前景仍可期待,市场起飞需等待时机[EB/OL].http://www.gkong.com/html/news/2008/4/20368.html,2009.
[7] Carcelle X,Heile B,Chatellier C,et al.Next WSN applications using ZigBee[C].IFIP International Federation for Information Processing,Boston:Springer,2008:239-254.
[8] IEEE.Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks[S].IEEE Std 802.15.4-2006,September 2006.
[9] 刘新,吴秋峰.无线个域网技术及相关协议[J].计算机工程,2006,32(22):102-103.
[10] ZigBee Alliance.ZigBee Specification[S].053474r06,December 2004.
[11] ZigBee Alliance. ZigBee Specification[S].053474r13,October 2006.
[12] ZigBee Alliance.ZigBee Specification[S].053474r17,January 2008.
[13] ZigBee Alliance.ZigBee_Tutorial.pdf[M].http://www.zigbee.org/,2009.
[14] ZigBee Alliance. What is a Public Application Profile[EB/OL].http://www.zigbee.org/About/FAQ/tabid/192/Default.aspx#14,2009.
[15] ZigBee Alliance.094928r00ZB_MWG-ZigBee_and_RF4CE_Set_New_Course_for_Consumer_Electronics_Rem.pdf[M].http://www.zigbee.org/,2009.
[16] 李俊贤.ZigBee 通讯堆叠与应用类别之相容性测试与认证[J].电脑与通讯,2008,(123):98-107.
[17] Z-Wave Alliance.Joining Together To Make Home Control A Reality[EB/OL].http://www.z-wavealliance.org/,2009.
[18] ZigBee Alliance. ZigBee Ecosystem Report[EB/OL].http://zigbee.org/imwp/web3/reports/ecosystems.aspx,2009.
[19] 缪彩琴,翁寿松.SIP 和 SOC[J].电子与封装,2005,5(8):9-12.
[20] 李俊贤.ZigBee 技术规格与测试方案之发展[J].电脑与通讯,2007,(119):32-41.
[21] 苏州博联科技有限公司.ZIGBEE 传输终端系列[EB/OL].http://www.beelinker.

com/,2009.
[22] 宁波高新区深联科技有限公司.GAINZ-2.4G 开发套件[EB/OL].http://www.wsn.org.cn/,2009.
[23] TI.Z-Stack - ZigBee Protocol Stack[EB/OL].http://focus.ti.com/docs/toolsw/folders/print/z-stack.html?DCMP=HPA_RFIC_General&HQS=Other+OT+z-stack,2009.
[24] Microchip.ZigBee 2006 Protocol Stack MpZBee[EB/OL].http://www.microchip.com/stellent/idcplg?IdcService=SS_GET_PAGE&nodeId=1824&appnote=en537767,2009.
[25] Reese R.A Zigbee-subset/IEEE 802.15.4 Multi-platform Protocol Stack [EB/OL].http://www.ece.msstate.edu/~reese/msstatePAN/,2009.
[26] ZigBee Alliance.074925r00ZB_MWG-ZigBee_Alliance_Members_Connect_With_Telecom_Market.pdf[M].http://www.zigbee.org/,2009.
[27] 华立仪表集团股份有限公司.华立仪表 Zigbee 跨台区集中抄表系统鹰潭试点项目,顺利通过江西省供电公司验收[EB/OL].http://www.holleymeter.com/gb/news_single.php?id=11,2009.
[28] 王丹.ZigBee 中国产业基地将落户高新区[N].江城日报,2008-12-8.
[29] Chakeres I,Klein-Berndt L.AODVjr,AODV Simplified[J].Mobile Computing and Communication Review,2002,6(3):100-101.
[30] 王芳,柴乔林,班艳丽.一种改进的 ZigBee mesh 网络路由算法[J].计算机应用,2008,28(11):2788-2794.
[31] 吕治安.ZigBee 网络原理与应用开发[M].北京航空航天大学出版社,2008.
[32] ZigBee Alliance.075299r02ZB_MWG-ZigBee_Specification_Features_&_Benefits.pdf[M].http://www.zigbee.org/,2009.
[33] Daintree Networks.Compare ZigBee Specification Versions[EB/OL].http:// www.daintree.net/resources/spec-matrix.php,2009.
[34] 张志龙.分析标准规范不同 ZigBee 1.1/Pro 相容问题有解[J].新电子,2007,(257):1-11.
[35] 李晓维,徐勇军,任丰原.无线传感器网络技术[M].北京理工大学出版社,2007.
[36] 陈卓,张正文.论扩展频谱通信技术在无线网安全性中的应用[J].计算机应用研究,2000,17(4):78-79.
[37] 信息产业部.中华人民共和国无线电频率划分规定[S].中华人民共和国信息产业部令第 40 号,2009.
[38] TI.CC2430.pdf[DB/OL].http://www.ti.com/,2009.
[39] Freescale.MC13213:2.4GHz RF transceiver and 8-bit MCU with 60K of Flash for ZigBee applications[EB/OL].http://www.freescale.com/webapp/sps/site/prod_summary.jsp?code=MC13213&nodeId=0106B9869925657103,2009.
[40] Freescale.MC13224V:MC1322x Platform in a Package[EB/OL]. http://www.freescale.com/webapp/sps/site/prod_summary.jsp?code=MC13224V&nodeId=0106B9869925657103,2009.

[41] 王宜怀,刘晓升.嵌入式技术基础与实践[M].清华大学出版社,2007.
[42] Freescale.MC13213RM.pdf[DB/OL].http://www.freescale.com/,2009.
[43] 荐红梅.基于硬件构件的嵌入式底层软件开发方法研究及其应用[D].苏州:苏州大学,2008.
[44] 唐勇,周明天,张欣.无线传感器网络路由协议研究进展[J].软件学报,2006,17(3):410-421.
[45] 关媛.移动自组网 AODV 协议优化及安全路由方案研究[D].苏州:苏州大学,2006.
[46] Perkins C,Bhagwat P.Highly dynamic destination-sequenced distance-vector routing (DSDV) for mobile computers[J].Computer Communication Review,1994,24(4):234-244.
[47] Clausen T,Jacquet P.Optimized Link State Routing Protocol (OLSR)[S].RFC3626,October 2003.
[48] Johnson D,Hu Y,Maltz D.The Dynamic Source Routing Protocol (DSR) for Mobile Ad Hoc Networks for IPv4[S].RFC4728,February 2007.
[49] Perkins C,Belding-Royer E,Das S.Ad hoc On-Demand Distance Vector (AODV) Routing[S].RFC3561,July 2003.
[50] Belding-Royer E,Perkins C.Evolution and future directions of the ad hoc on-demand distance-vector routing protocol[J].Ad Hoc Networks,2003,1(1):125-150.
[51] 郑相全等.无线自组网技术实用教程[M].清华大学出版社,2004.
[52] 史美林,英春.自组网路由协议综述[J].通信学报,2001,22(11):94-103.
[53] 朱向庆,王建明.利用捎带技术提高 ZigBee 网络性能的方法[J].微计算机信息,2008,24(3-2):56-58.
[54] 张华,张志胜,伏明明.车间设备监控系统的设计与开发[J].可编程控制器与工厂自动化,2008,(6):79-81.
[55] 暴庆兰,王建利.浅淡车间的设备管理[J].世界采矿快报,2000,16(12):464-465.
[56] 王瑛,卢修文,潘云.基于ZigBee和ARM的嵌入式智能楼宇无线火警系统设计[J].电子元器件应用,2008,10(7):32-34.
[57] 杨艳华,张凤登,马进明.ZigBee 技术在火灾自动报警系统中的应用[J].上海电力学院学报,2008,24(4):393-396.
[58] Freescale.MC9S08AW60.pdf[M].http://www.freescale.com/,2009.
[59] Freescale.AN2493:MC9S08GBGT Low Power Modes.pdf[M].http://www.freescale.com/,2009.

攻读学位期间本人公开发表的论文

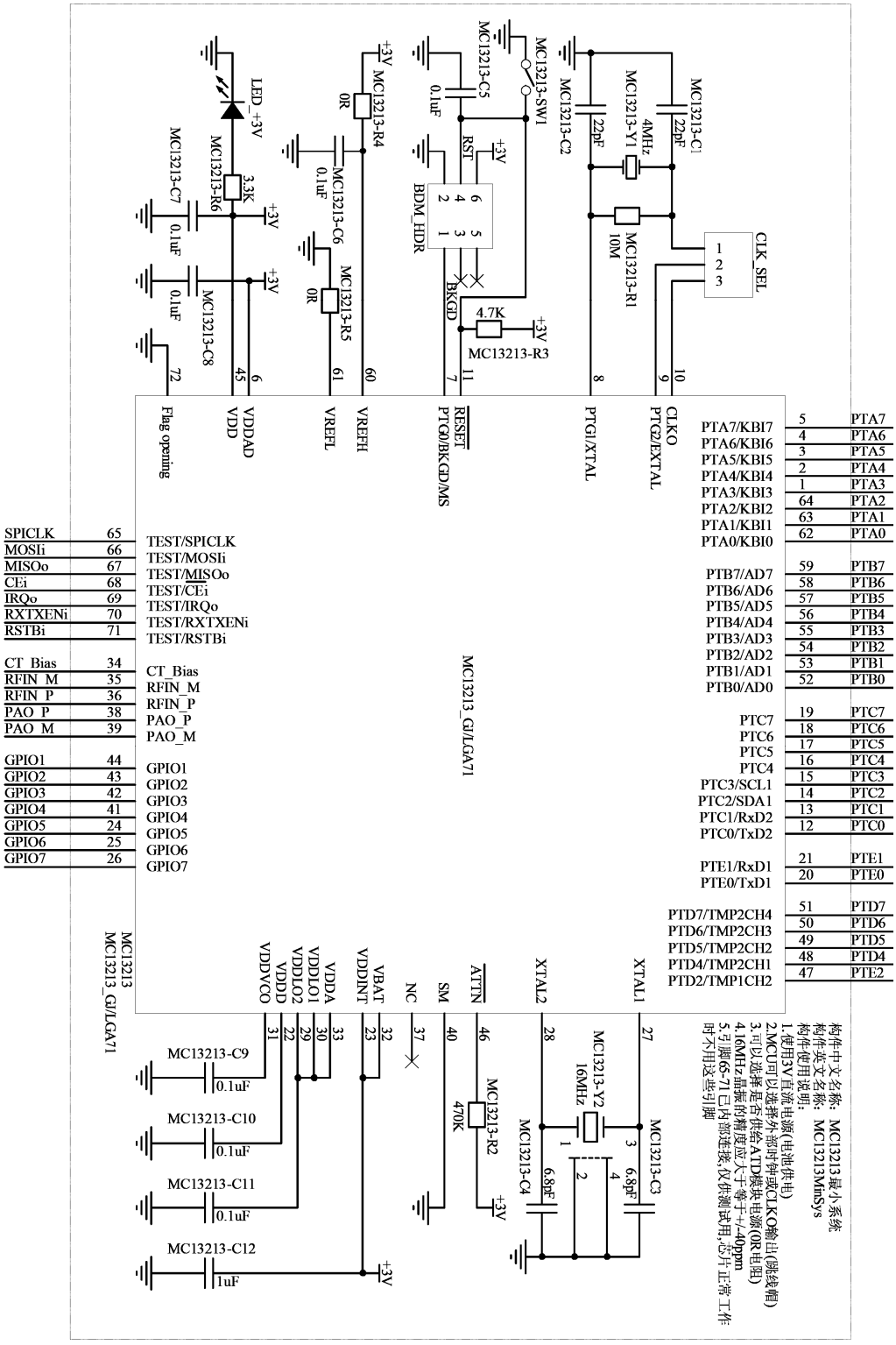
[1] 倪敬飞，王宜怀等，一种面向中小规模嵌入式应用的软件开发框架，微计算机信息，2009(已录用)。

[2] 作者参与研制的 SD-II 系列嵌入式系统开发平台于 2008 年 5 月通过江苏省科技厅鉴定。

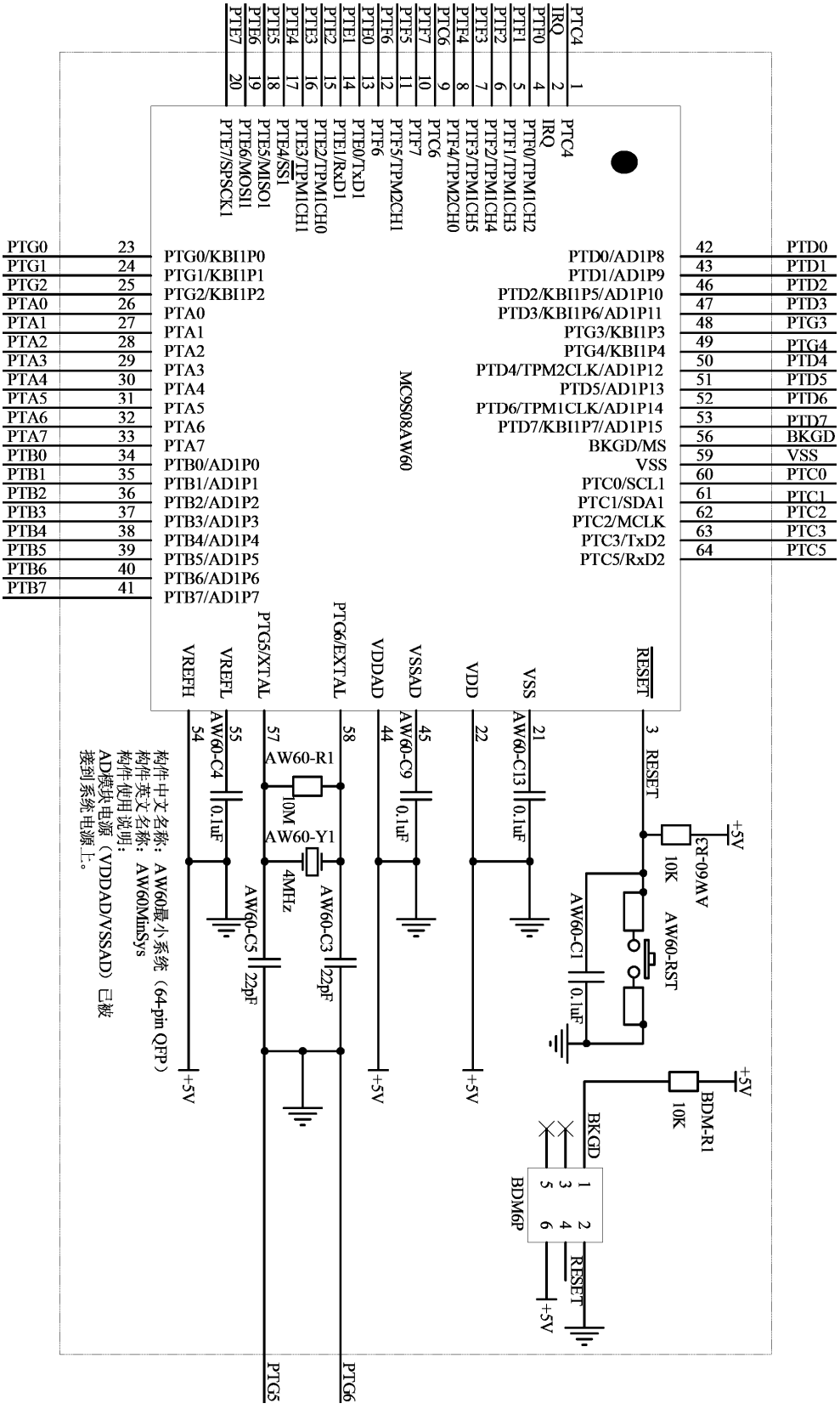
[3] 作者参与了 2007 年第二届“飞思卡尔”杯全国大学生智能汽车竞赛，并获全国总决赛二等奖。

附录 A 硬件原理图

A.1 ZigBee 硬件最小系统原理图

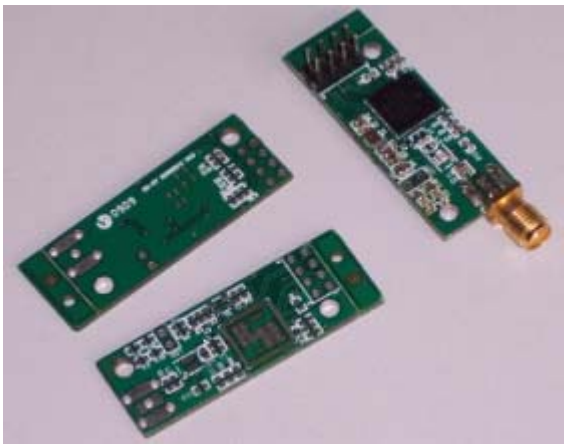


A.2 数据采集接口板最小系统原理图

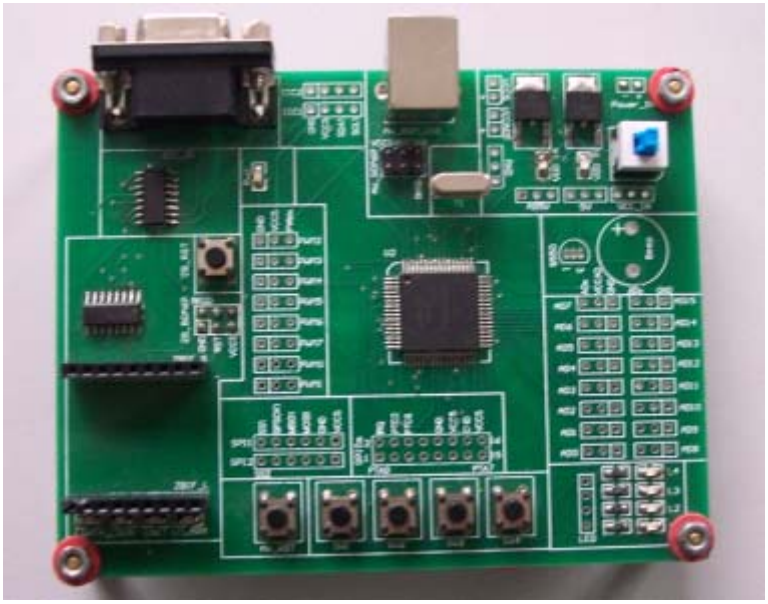


附录 B 硬件实物图

B.1 ZigBee 硬件实物图



B.2 数据采集接口板实物图



致 谢

本文是在导师王宜怀教授的悉心指导下完成的，在此，特向导师表示深深的感谢和崇高的敬意。在研究生期间的学习和生活中，您严谨的治学作风，谦逊儒雅的学者风范，以及细微处的言传身教将使我终生受益。

感谢刘晓升老师在平时的学习生活和相关的项目开发中给予的无私帮助，他对问题的独到见解使我受益匪浅，每次与他讨论问题总会有所收获。

三年中的大部分时间都是在实验室度过的，这是一个温馨融洽的集体，感谢实验室的兄弟姐妹们。特别感谢王凤林、王超艺、常赛，他们对我的毕业论文提出了宝贵的意见。

在这三年中有太多的老师、同学、朋友给了我无私的关心与帮助，在这里一并表达我真挚的谢意。

感谢我的家人，这么多年来毫无保留的物质支持和精神鼓励，使得我能够全身心的投入到学习中。

最后，诚挚地感谢为评阅本文而付出辛勤劳动的各位专家和学者。