分类 _____ 密级_____

西北师范大学

硕士学位论文

基于 ARM 与 Android 的智能家居系统设计与实现

吴志君

导师姓名职称：段富海　教授

专业名称：计算机软件与理论 研究方向：嵌入式系统

论文答辩日期：2012 年 5 月　学位授予日期：2012 年 6 月

答辩委员会主席：

评　　阅　　人：

二〇一二年五月

硕士学位论文

M.D Thesis

基于 ARM 与 Android 的智能家居系统设计与实现

Design and Implementation of Smarthome System Based on ARM and Android

吴志君

Wu Zhi-jun

独创性声明

本人声明所呈交的论文是我个人在导师指导下进行的研究工作及取得的研究成果。尽我所知，除了文中特别加以标注和致谢的地方外，论文中不包括其他人已经发表或撰写过的研究成果，也不包含为获得西北师范大学或其他教育机构的学位或证书而使用过的材料。与我一同工作的同志对本研究所做的任何贡献均已在论文中作了明确的说明并表示了谢意。

签名：_____ 日期：_____

关于论文使用授权的说明

本人完全了解西北师范大学有关保留、使用学位论文的规定，即：学校有权保留送交论文的复印件，允许论文被查阅和借阅；学校可以公布论文的全部或部分内容，可以采用影印、缩印或其他复制手段保存论文。

（保密的论文在解密后应遵守此规定）

签名：_____导师签名：_____日期：_____

摘　要

物联网的发展为智能家居带来了新的机遇。随着国内人民生活水平的提高，智能家居不仅仅是别墅、高档小区的独有之物，越来越多的中低档户型对智能家居的需求日益增加。本文通过对各种智能家居系统进行分析后，设计了一种可以满足中低档户型智能家居设计的整体解决方案。

本文首先设计了智能家居系统的整体架构，以 S3C6410 作为智能家居系统的主控制器，家庭内网 Zigbee 将家中环境和家用电器有机组合在一起，并与 Internet 和 GPRS 网络共同组成了一个安全、方便、舒适的家居环境。然后分别从硬件和软件两方面对智能家居系统进行阐述。硬件主要包括主控制器的选择和节点的设计，重点介绍了前端传感器的硬件原理图。软件方面分别从功能、设计和实现三方面对主机系统软件进行了详细的分析。最后对该系统进行了测试。该设计的创新点：在环境监测设计中使用了新型的、技术领先的传感器；使用 Android 系统作为智能家居终端设备的操作系统，并在 Android 系统上使用了 MVC 模式设计了智能家居主程序，极大地降低了系统各层的依赖，有利于组件的重用。本文主要研究内容：

(1)根据用户需求提出了本文的总体设计方案，满足中低档户型对智能家居的需求。

(2)通过对环境监测节点和执行节点功能的研究和分析，设计了相应的硬件电路；深入分析了 Z-stack 协议栈，完成了主机节点和终端节点的应用程序设计。

(3)深入研究了 Android 系统架构，剖析了 Android 应用组件和 Activity 生命周期，对 Android 平台底层组件软件开发和应用层软件开发过程进行了深入研究，掌握了从底层驱动的开发到 Android 应用程序的开发流程；在 Android 系统上设计了智能家居主程序，实现了室内环境与家电的连动模式，并且能够在系统监测到异常情况下，及时处理各种异常。

将主机系统、各节点模块分别上电，通过对该智能家居系统进行实际测试和验证，结果表明系统工作正常，达到了前期的设计要求，验证了电路设计的合理性。

关键词：智能家居；Android操作系统；Zigbee；ARM；JNI

Abstract

The development of IOT has brought new opportunities for the SmartHome. With the improvement of living standards of people in China, the SmartHome is not just villas, upscale district of the unique things, more and more units of middle and low demand for SmartHome growingly. In this paper, analysis of a variety of the SmartHome system, design the overall solution to meet the medium and low size SmartHome design.

Firstly, this paper design the overall architecture of the SmartHome system, the S3C6410 as the main controller , within the family network Zigbee combinate the home environment and household appliancest together, organic combined with the Internet and the GPRS network together to form a safe, convenient and comfortable home environment. Then describe the SmartHome system from hardware and software respectively. Hardware mainly includes the choice of the main controller and the design of the node, and focuses on the hardware schematic diagram of the front-end sensor. Software carries on the detailed analysis to host system software from the function, design and realization of three aspects respectively. Finally the system is tested. The scheme has the following innovation points:the use of new, technologically advanced sensors; use Android as the operating system for the SmartHome terminal equipment,and use the MVC pattern to design the SmartHome main program on the Android system,greatly reduces the dependence of the system layers, conducive to the reuse of components.This paper studies the content:

(1) Put forward the general design based on user needs, which meet the needs of middle and low units on the SmartHome.

(2) Research and analysis of environmental monitoring nodes and execution node function, design the corresponding hardware circuit; deeply analysis of the Z-stack protocol stack and complete the application design of the host node and terminal nodes.

(3) In-depth study of the Android system architecture, analysis of the Android application components and life cycle of Activity , deeply studies the methods of

kernel-level component software development and application software development,master the process from the bottom-driven development to Android the program development; design the main program of SmartHome based on the Android system, realize the indoor environment with appliances even mode, and timely process of a variety of abnormalities with the system monitoring to exceptional circumstances.

Power on the host system, each node module, the SmartHome system by the actual testing and validation, the results show that the system works well, and reached the pre-design requirements, verify the rationality of the circuit design.

Keywords: SmartHome; Android Operating System; Zigbee; ARM; JNI

目　录

第 1 章 绪论

1.1 嵌入式处理器的发展

嵌入式系统是计算机技术、通信技术、半导体技术、微电子技术、语音图像数据传输技术，甚至传感器等先进技术和具体应用对象相结合后的更新换代产品，反映当代最新技术的先进水平。嵌入式系统是当今非常热门的研究领域，在PC 市场已趋于稳定的今天，嵌入式系统市场的发展速度却正在加快。由于嵌入式系统所依托的软硬件技术得到了快速发展，因此嵌入式系统自身获得了快速发展。

嵌入式微处理器的快速发展，使得嵌入式系统已经广泛地应用我们的生活的各个领域。随着应用领域的不断扩大和深入，对于嵌入式处理器速度要求越来越高。所以，未来微处理器的发展趋势：集成度越来越高、主频越来越高、机器字长越来越大、总线越来越宽、同时处理的指令条数越来越多。

单从 ARM 体系的发展来看，从最初的 ARM1 原型到现在的四核 Cortex-A15，CPU 处理器的速度从 9MHZ 到现在的 2.5GHZ，这足以和 x86 处理器相抗衡。目前市场上常见的 ARM 处理器系列有 ARM7、ARM9、ARM11、Cortex 系列[1]，而处理器数字的不断递增表明了处理器的性能和复杂度的提高。ARM 处理器以低功耗、底层本、实用性强为特点，并以卓越的产品性能著称于世，如今 ARM 处理器及技术的应用在无线通信、网络应用、消费类电子产品等领域随处可见[2]。目前在移动设备市场，ARM 处理器的市场份额超过 90%；在计算机领域，微软公司宣布，下一版 Windows 将正式支持 ARM 处理器[3]。因此，ARM 很有可能会超越 Intel 成为主流。

1.2 研究背景

二十世纪九十年代后期，智能家居的概念开始在国内出现。2000 年智能家居开始在国内推广。虽然经过十几年的发展，智能家居在我国的一些小区已经得到了一定程度的应用，但是这并不代表智能家居在我国得到了良好的推广和发展。相反，随着智能家居的推广，出现了一系列问题：缺乏规范的、统一的行业

标准；成本费用高；功能华而不实；缺乏资金支持；缺乏完善的社会合作体系；跨产业的合作困难重重；个人隐私等[4]。这些因素严重阻碍了智能家居的发展。

物联网的快速发展为智能安防注入了新的活力。物联网的智能家居系统需要一个支撑平台，需要建立服务器，需要跟运营商合作，是面向更为广阔的客户群体，可以一对多进行统计的管理，数据及信息可以得以保存。对于用户而言，管理起来更为方便，只需要查看终端或手机，就可以访问控制家庭中的智能家居系统，为用户提供舒适安全、宜人的家庭生活空间;还由原来的被动静止结构转变为具有能动智慧的工具，提供全方位的信息交换功能，帮助家庭与外部保持信息交流畅通，优化人们的生活方式，帮助人们有效安排时间，增强家居生活的安全性，甚至节约各种能源费用。这种智能家居的方式更为便利，更加智能化。在国内已经开始在全国不少城市普及，几乎新建的每个小区和家庭都会配置一定的智能化系统。伴随科技的发展，在人类丰富的想象空间里，智能化住宅的发展必将带来人类居住的一场大变革。

2009 年 8 月，温家宝总理郑重提出建立“感知中国”的概念，开启了我国物联网建设大幕。物联网的发展势必会推动智能家居的向前发展。针对目前市场上国内智能家居行业出现的诸多问题：缺乏规范的、统一的行业标准；成本费用高等等，解决这些问题的突破口就在于物联网。

1.3 课题研究的目的和意义

目前市场上还没有一款使用 Android 操作系统的智能家居系统，而按 Android 系统目前的发展趋势，Android 很有可能成为便携式设备的主流系统。Android 系统的优势在于它不仅可以在手持设备上运行，还可以在平板电脑上运行，还可以作为智能家居的家庭网关，这是其他系统无法达到的。鉴于以上优点，如果在 Android 系统上开发一款智能家居软件，不仅可以缩短研发周期，而且便于日后的升级、维护。Zigbee 是一种功耗少、距离近的无线通信技术。Zigbee 主要是在远程控制领域和自动控制领域中运用，它的主要特点是成本低、距离短、功耗少、复杂度小、数据速率低、自组网。Zigbee 技术已经成为智能家居内建子网的首选。本系统家庭内网也采用 Zigbee 组网。

随着 Android 系统和 Zigbee 技术的不断发展，它们很有可能成为智能家居系统中操作系统和内建网络的标准。本文通过设计一个小型的智能家居系统，实现

了智能家居系统中关键的功能，成为智能家居设计的原型系统，为中低档住宅用户提供了一套完整的智能家居系统解决方案。

1.4 国内外发展现状与趋势

国外智能家居行业发展较早，通过市场调研发现，在中国从事智能家居行业的外国公司主要是美国的 Control4 和 Honeywell。Control4 是一家专业从事智能家居产品的研发、生产、销售的知名企业[5]，提供一整套的有线和无线系列控制产品，先进的连接和控制方式，通过 Composer 管理软件，可以对整个家庭自动化系统进行快速设置以及个性化设定。Honeywell 是一家在多元化技术和制造业方面占世界领导地位的跨国公司[6]，其网络型智能家居系统在功能上是一个高集成化的系统，基本涵盖了智能家居系统所有的功能，在以家庭网关为核心的同一平台上分别集成了可视对讲功能、门禁控制功能、家居安防功能、信息管理发布功能、网络家电智能控制、远程抄表、网络远程控制等众多功能，让用户在一套智能家居系统平台上就充分感受到了舒适、安全、信息化所带来的享受。智能家居在美国、德国、新加坡、日本、韩国等国家都有了大规模的应用。

虽然国外智能家居行业有着领先的技术和较完善的整体解决方案，但是其配置过程相对繁琐，以 Control4 的 Composer 软件为例，首先需要手动选择模块驱动，然后对相关联模块进行编程，当模块数量较多时，很难在短时间内完成其预定的配置。另外，国外智能家居系统价格高昂。Control4 的一台家庭主机市场价是 7000 元左右，而像各种调光开关模块，单个售价大都在 300 元以上；而 Honeywell 的一套 HRIS 无线套装售价为 26000 元，而这样的价格只是其基本配置。

相比国外，国内智能家居的发展较晚，但在国内智能家居行业发展相当迅速。目前，国内的智能家居品种有数百种之多。

在国内具有代表性的智能家居系统有海尔的“U-home”、安居宝的“数字终端”、索博（SUPER）的“全球控”和“全宅控”等。而这些系统有一个共同的特点就是偏重于家电控制，而且各子系统基本上不能统一控制，由于缺乏行业标准，各个厂家的产品互不兼容，价格也不容易被用户接受。 到目前为止，没有一家智能家居产品能够占领国内 10%的市场份额，这些问题也极大地降低了消费者的购买力。

在智能家居发展过程中，标准成为了智能家居的发展瓶颈，未来智能家居的

发展首先要解决的问题是标准。

1.5 研究内容和论文结构安排

目前，国内智能家居系统所用的处理器大多为 ARM9 或同频率的处理器，少数公司采用 ARM11，还有一部分采用单片机进行控制。受处理器主频的限制，使得安防系统有些功能无法实现，而且当系统需要进行较大的数据通信时，现有功能也不能很好的发挥其性能。因此，为解决现有智能家居系统的不足，决定在本项目上采用的处理器为 ARM11，操作系统为 Android 及组网采用当前流行的 Zigbee 无线网络。

智能家居是以住宅为平台安装有智能家居系统的居住环境，利用综合布线技术、网络通信技术、安全防范技术、自动控制技术、音视频技术将家居生活有关的设备集成，通过系统控制，能够遥控家中电器，能够随心所欲支配家居生活。因此，智能家居的建设是一个高技术含量的系统工程，任何一个环节出现问题都可能影响整个系统的性能。

所以在对智能家居系统进行开发之前，对与目前市场上常见的智能家居系统进行了调研，在此基础上设计和实现了一款适合于低端家庭用户的智能家居系统。本文主要的研究内容如下：

(1) 明确系统所要实现的功能，设计系统的整体框架。在这个过程中，首先对系统的功能进行描述，明确系统所要实现的功能，然后需要确定整个系统各功能模块之间的相互关系，然后细化各个模块所要实现的主要功能。

(2) 完成了系统的硬件设计，包括主控制器和传感器的选择、环境监测节点和执行节点的硬件设计。

(3) 深入分析 Android 体系架构，了解 Android 内部实现原理，掌握了 Android 平台下开发软件的流程，完成了节点软件和智能家居系统软件的设计和实现。

基于以上研究内容，本论文主要分为以下七章：

第 1 章是绪论，本章简要介绍了嵌入式处理的发展，然后介绍了研究该智能家居系统的目的、意义以及主要的研究内容。

第 2 章是智能家居系统总体设计，介绍了该系统所要实现的功能，给出了该系统的整体架构。

第 3 章是智能家居系统硬件设计，主要从主控制器的选择和节点的设计两方面进行了详细的分析。

第 4 章是智能家居系统软件设计，首先介绍了节点的软件设计，对系统中各模块的功能进行了详细的分析，然后分别对各模块进行设计。

第 5 章是智能家居系统软件实现，这里首先介绍系统开发环境的搭建，然后详细地叙述了系统的实现过程。

第 6 章是智能家居系统软硬件综合调试与验证，首先介绍了 Android 平台下如何进行软件调试，然后通过在开发板上运行该软件系统进行验证。

第 7 章是总结和展望，首先对自己所做的工作进行了总结，然后提出了系统的改进和后期进一步工作。

1.6 本章小结

本章首先介绍了嵌入式处理器的发展状况，接着介绍了智能家居系统的研究背景，然后阐述了该智能家居系统的研究目的及意义，分析了当前国内外智能家居的发展现状，最后说明了论文做的主要工作，最后简要介绍了本论文的章节安排。

第 2 章 智能家居系统总体设计

2.1 智能家居系统所要实现的功能

2.1.1 功能模块介绍

在本智能家居系统中，主要完成如图 2-1 所示的 9 个功能模块设计和实现，这 9 个模块分别是家居安防、家电控制、语音留言、可视门镜、电话功能、电子秘书、访客记录，状态监控和系统设置等。这些模块要实现的基本功能为：

图 2-1 系统框架图

家居安防：这里设置了 3 中常用的情景模式，居家模式、离家模式和夜间模式。用户可根据需要选择相应的情景模式，或者对已选择的模式进行撤销或切换等操作，当用户选择了一种情景模式后即可查看该模式下传感器的工作状态。

家电控制：通过主机发送学习命令，万能遥控器学习家电遥控器控制模式后，用户可以通过主机命令、手机短信命令等方式，控制相应家电的开关。

语音留言：用户可以进行离家留言的录制、播放和删除。

可视门镜：它与人体感应模块配合使用，安装在门上的摄像头可自动感应，当感应到物体后自动录像并存储，物体消失后自动停止，用户可选择开启摄像头来观察当前门外的状况或关闭摄像头的捕获影像窗口。

电话功能：当用户使用该模块时，此时的设备相当于一部固定电话，可对外

进行电话的拨打、挂断、接听等基本功能。

电子秘书：用户可进行语音留言、短信内容编写，设定备忘提醒的时间，用户预设的备忘提醒可选择以短信的形式发送到预设号码上，也可以通过拨打预设号码以语音的形式提示；

访客记录：用户通过该模块可查看可视门禁所拍摄的视频，及时了解来客信息。

状态监控：此模块的功能是定时检测系统中各传感器的工作状态，查看各传感器节点是否正常工作。

系统设置：在系统设置模块中，用户可以对系统设备、预设电话号码、预设短信命令、情景模式、家电控制及主机时间、日期等进行设置。

此外，在智能家居系统中还有一个重要的功能就是远程遥控。通过该功能用户可通过手机、电脑等网络设备对家居安防中的情景模式设置、设防/撤防、家电控制等。

2.1.2 功能模块关系分析

本节主要针对家居安防系统所要实现的功能进行分析，确定各模块之间调用关系，为系统软件的设计和实现提供逻辑基础。

在系统设置模块中，用户可以对系统设备、预设电话号码、预设短信命令、情景模式、家电控制及主机时间、日期等进行设置，这样系统其它模块都需要先调用系统设置模块进行设置；在家居安防中，除了访客记录和电子秘书外，其它模块都与之关联；与可视门禁关联的模块有家居安防、电话功能和访客记录；家电控制既可以在家居安防中控制，又可以用短信控制；与电话功能关联的模块有家居安防、可视门禁、电子秘书和家电控制。该系统中主要的功能模块关系如图 2-2 所示。

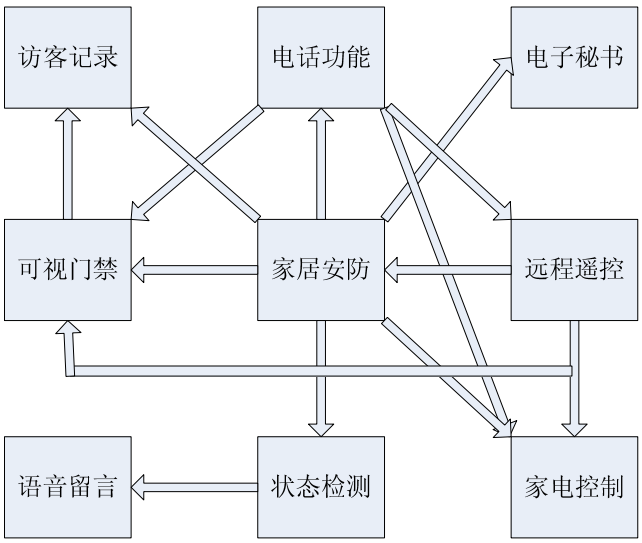

图 2-2 智能家居模块功能关系图

2.2 系统整体架构

该系统主要由主控制器、Zigbee 传感网络、便携多媒体终端以及 PC 这部分构成。主控制器使用的微处理器是采用三星的 S3C6410 芯片。Zigbee 传感网络是基于 Zigbee 技术的自组织网，在这个网络中包含了系统中所有的无线节点模块，比如负责环境监测的节点、负责家电控制的执行节点以及信息汇集的协调器节点。便携多媒体终端可以是手机、平板电脑等。当系统上电运行时，前端传感器采集到的环境状态信息发送到协调器节点上，再由该节点传递到主控制器进行处理。系统整体架构如图 2-3 所示。

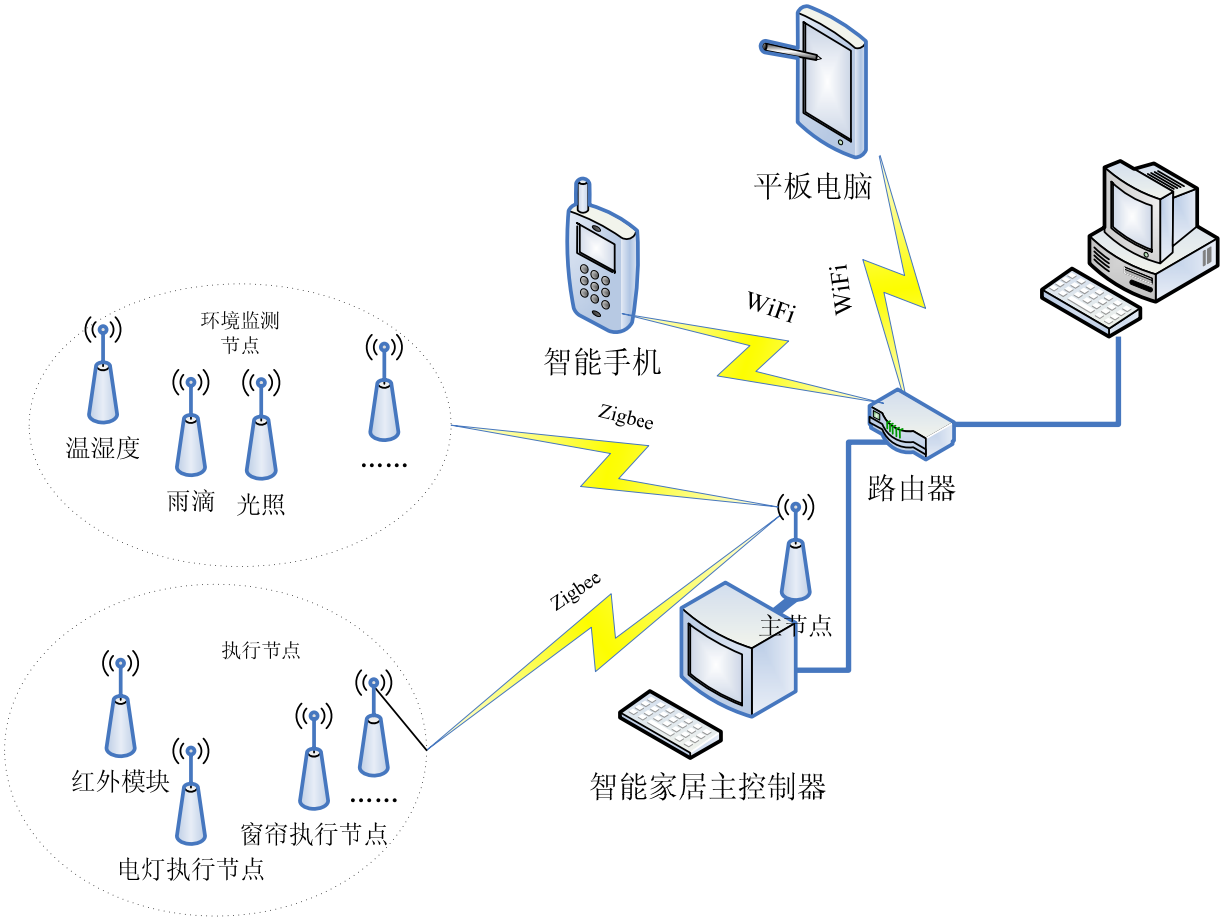

图 2-3 系统整体架构图

2.3 本章小结

本章先对智能家居系统的 9 大功能模块进行了简要的阐述，概述了各模块所能实现的基本功能，然后讨论了系统各模块之间的调用关系。最后给出了该系统的整体架构。

第 3 章 智能家居系统硬件设计

3.1 硬件设计概述

智能家居硬件设计主要分为两大部分，一部分是主控制器的选择，一部分是相关节点的硬件设计。主控制器选择原则是能够流畅运行 Android2.1。节点设计又可分为环境监测节点硬件设计和执行节点硬件设计，这两种节点都需要通过 CC2530 射频模块与主机上的主节点通信，环境监测节点在设计中主要考虑传感器的选择与接线；执行节点则根据要控制的终端设备设计不同的硬件电路。

3.2 主控制器

根据实验测试，在 2440 上跑 Android2.1，虽说可以运行，但是由于受到硬件的限制，系统会经常出现“假死机”现象；而在 6410 上测试则不会此现象。再考虑成本，此 S3C6410 开发板大都不超过 1000 元。而对于性能更强的 Cortex 系列来说，目前市场上的资料较少，而且价格较高，并不是很适合本系统。综合考虑，本系统采用 S3C6410 开发板。其硬件资源如图 3-1 所示，主控制器是智能家居系统的核心部件，他负责对终端节点传来的数据进行分析和处理。智能家居系统的所有软件和协议都在此处运行，主要采用三星的 S3C6410 作为核心处理器，256M SDRAM 的内存，1G NAND FALSH 作为系统软件的存储介质，最高主频可达 800MHz。此外，该主控器周围接有丰富的外围设备：Zigbee 无线网络通信接口；多路输入和输出的音频设计，与 GSM 模块构成完美 Phone 解决方案；WM9713audio，AC97 接口；有线视频采集模块；无线视频采集模块；大容量可更换的存储设备 SD 卡接口；两路 TTL Uart 接口；调试程序用的 JTAG 口、串口、USB Host/Device；板载 10/100M DM9000AEP 以太网；100M Ethernet RJ45 接口；Camera 接口，常用总线接口，如 CAN、IIC 实现可视化人机交互的 7 寸液晶接口；GSM/GPRS 模块 SIM300 板载接口；SDIO WIFI 接口； VGA 接口；预留大容量 MoviNand 接口，可扩展更大容量的 MoviNand flash；AV 输出端子和 S 端子输出；两路可切换 AVIN 视频输入接口；板载 MIC，板载 1W 喇叭等等，这些外设足以满足智能家居系统的硬件要求。

图 3-1 S3C6410 开发板

这里的 Zigbee 无线通信模块位于开发板的背面，他通过串口 Uart3 与 S3C6410 连接，通过拨码开关我们可以将 Zigbee 无线功能开启。其接线图原理图如图 3-2 所示：

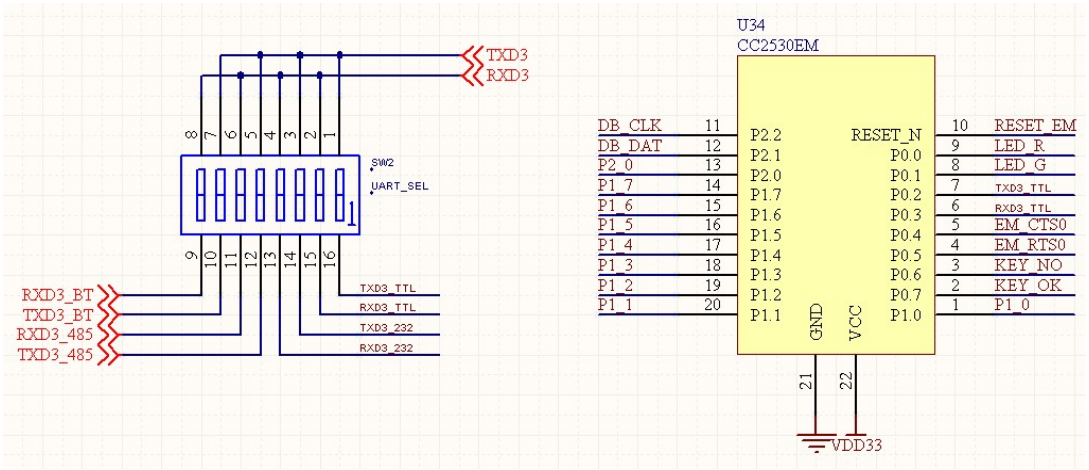

图 3-2 Zigbee 主节点与开发板接线图

3.3 无线收发模块

CC2530 是一个兼容 IEEE 802.15.4 的真正的片上系统，支持专有的 802.15.4 市场以及 Zigbee、Zigbee PRO 和 ZigbeeRF4CE 标准。CC2530 提供了 101dB 的链路质量，优秀的接收器灵敏度和健壮的抗干扰性，四种供电模式，多种闪存尺寸，以及一套广泛的外设集——包括 2 个 USART、12 位 ADC 和 21 个通用 GPIO，

以及更多。除了通过优秀的 RF 性能、选择性和业界标准增强 8051MCU 内核，支持一般的低功耗无线通信，CC2530 还可以配备 TI 的一个标准兼容或专有的网络协议栈（RemoTI, Z-Stack, 或 SimpliciTI）来简化开发，使你更快的获得市场。CC2530 可以用于的应用包括远程控制、消费型电子、家庭控制、计量和智能能源、楼宇自动化、医疗以及更多领域。

这里我们选择的版本是 CC2530F256，它具有 256KB 的闪存。而且他结合了德州仪器的业界领先的黄金单元 Zigbee 协议栈，提供了一个强大和完整的 Zigbee 解决方案。

CC2530 内部原理图[7]和引脚说明分别如图 3-3 和 3-4 所示。

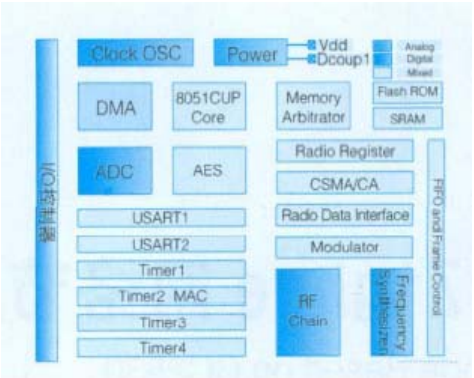

图 3-3 CC2530 内部原理图

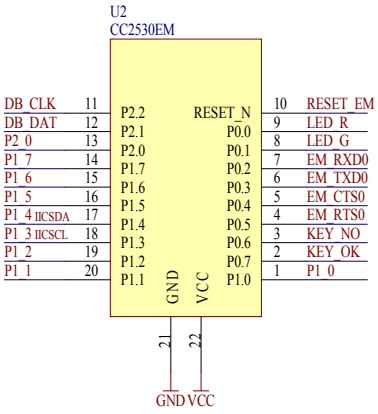

图 3-4 引脚说明

3.4 节点设计

按照节点功能划分，本系统的节点可分为主节点和终端节点两类。主节点主要负责接收终端节点发送来的数据以及发送主机向终端节点发送的控制命令。终端节点又可以分为两类：一类是环境监测节点，另一类是执行节点。

3.4.1 环境监测节点和执行节点设计

图 3-5 为环境监测节点的硬件结构框图，此类型的节点包括射频通信模块、传感器模块、电源模块、调试和烧写接口模块以及其他辅助电路。射频通信模块由 CC2530EM 与 PCB 天线构成它是整个节点的核心。传感器模块主要是通过温湿度、雨滴、气体等各种传感器监测家中温湿度是否适宜，下雨后窗户是否已经关窗，家中一氧化碳浓度是否超标等等，然后将监测的数据传递给 CC2530EM 内置的微处理器中。微处理器单元主要由 8051 单片机、存储器以及嵌入式操作

系统构成的模块，它的功能是控制本身的传感器节点正常休眠、工作，及储存相关数据，并对数据进行一定地处理[8]。如果传感器监测实际的数据超出或者低于设定的值，则将该数据以一定的格式发送的主节点中，再由主节点发送到主机系统中，由系统进行处理。射频通信模块还负责接收主节点发来的控制命令，如查询命令，主要用于查询节点是否工作正常，以及时更换节点，保证系统的正常运行。电源模块分为两种供电方式：一种是电池供电，2 节 5 号干电池即可满足；另一种是电源供电。无论哪种方式都可为整个系统提供正常工作所需的电能。调试和烧写接口模块用来调试和烧写完成上述功能的软件。此模块的体积小,功耗低,安装、维护都很简单。图 3-6 是带有雨滴传感器的终端节点。

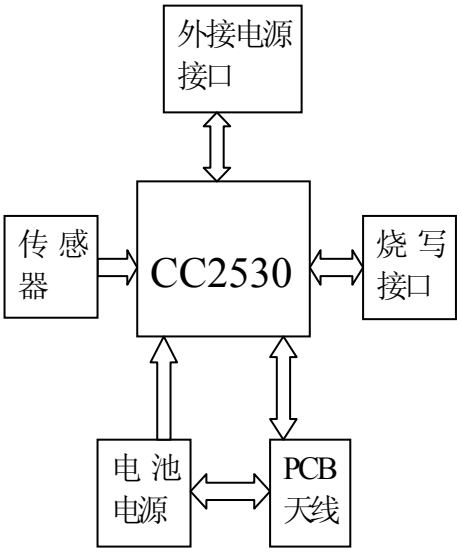

图 3-5 环境监测节点的硬件结构框图

图 3-6 带有雨滴传感器的终端节点

执行节点与环境监测节点的区别在于，执行节点负责终端设备的操作而环境监测节点仅对家居内环境监测，将监测到的数据进行发送。由图 3-7 硬件结构框图可见，执行节点主要由射频通信模块、继电器模块、过载保护模块和两组端子构成。这里需要说明的是根据终端设备的不同执行节点的硬件结构有所不同。如，继电器模块，他所能提供的是一个开关控制量，所以适用于灯光、浇花等系统的控制。若无继电器模块，该执行节点又可以控制电机，从而控制窗帘的开关。图 3-8 为无继电器模块的执行节点。

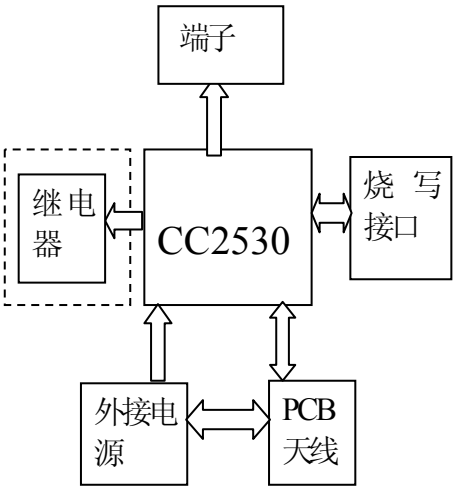

图 3-7 执行节点的硬件结构框图

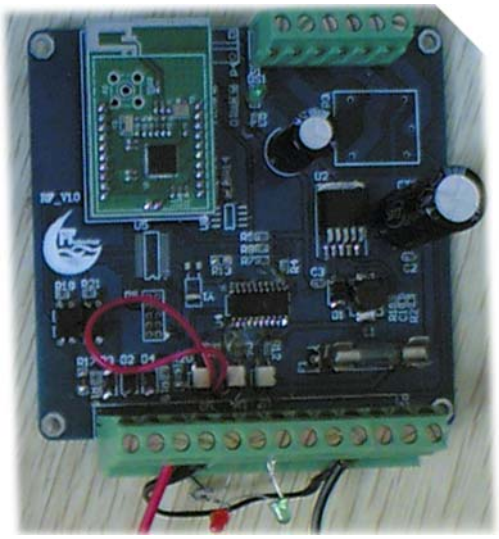

图 3-8 无继电器模块的执行节点

在执行节点中有一种特殊的节点那就是红外学习的 Zigbee 节点，这种节点在带有红外遥控器的家电中都能使用。本系统设计的具有红外学习功能的红外遥控器特性：在红外解码方面，采用单片机中断或者查询方式采集红外信号；在红外发射方面，通过实验发现红外发射距离受载波占空比和红外二极管贯通电流影响，通过调试将 38KHz 载波红外信号发射距离提高到 10 米；在红外接收方面，进行了红外干扰测试；提高了系统的稳定性[9]。红外遥控模块整体框图如下图 3-9 所示：

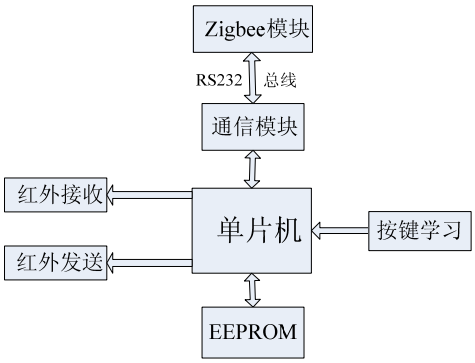

图 3-9 红外遥控模块整体框图

图 3-10 SAN-IR005A 红外控制模块

SAN-IR005A 型串口红外学习模块由学习模块、红外发射棒组成，实物图见上图 3-10 所示。设备通过输出 RS232 的指令来控制学习模块进行学习和控制红外设备，可以控制电视电视、DVD、投影机、空调，控制距离大于 7m。控制接口和电源输入通过一个 4PIN 的排针，定义分别是：PIN1-GND、PIN2-TXD、PIN3-RXD、PIN4-VCC。3 个 3.5 的插座分别是 RS232 通讯口，红外输出 1，红外输出 2。

下面对该模块的命令格式进行说明：

学习命令： 01 03 06 80 00 89 0D 0A

01 03 06 不变，80 表示学习操作，00 学习的组别（0--254）

89 是从 03—00 的和校验，89=03+06+80+00，0D，0A 回车换行是固定值

发送命令：01 03 06 81 00 8A 0D 0A

81 表示发送红外命令，其他定义如上。

学习命令发送后，学习指示灯亮，有红外遥控信号输入，学习成功指示灯熄灭，无遥控信号 30 秒后学习指示灯熄灭，自动退出学习状态，发送命令发送后，发射灯闪烁并通过红外发射棒发送红外信号。

3.4.2 传感器选择

环境监测节点中一个重要的模块就是传感器模块。因为只有传感器能正常工作，环境监测节点才有存在的意义。对于传感器的选择应符合以下几点要求：性能稳定；测量精度高；价格适中。下面将介绍本系统所采用的传感器：

(1)温湿度传感器

SHTxx 系列单芯片传感器是一款含有已校准数字信号输出的温湿度复合传感器。传感器包括一个电容式聚合体测湿元件和一个能隙式测温元件，并与一个 14 位的A/D 转换器以及串行接口电路在同一芯片上实现无缝连接。该传感器具有体积小、功耗低、响应速度快、接口简单、性价比高等特点，适合应用于自动控制、医疗、家电和消费品等领域[10]。综合考虑，该传感器的价格适中，测量精度高（温度测量精度：±0.5℃；湿度测量精度：±4.5%RH），完全满足智能家居中温湿度监测要求。图3-11为此传感器与CC2530的接线原理图：

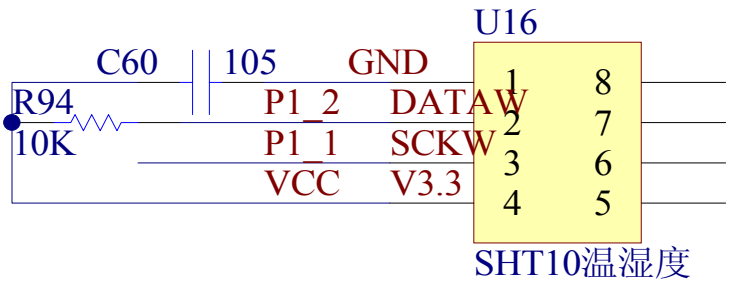

图 3-11 温湿度传感器与 CC2530 的接线原理图

(2)光照传感器

BH1750 是半导体制造商 ROHM 为适应以移动电话手机为首的便携式机器和液晶电视等的要求而开发出的具有优良光谱灵敏度特性、16bit 串行输出的单片数字照度传感器。该传感器的工作电压为 3.3V，内置了 16bit AD 转换器，无需外部部件。而且这种传感器无论是暗处还是在光照较强的地方都与人类视觉感应相似，能够进行大范围的亮度测定。该芯片使用 I2C 总线传输数据，消耗功率低，因此可作为智能家居中的光强度监测器件[11,12]。图 3-12 为该传感器的接线原理。

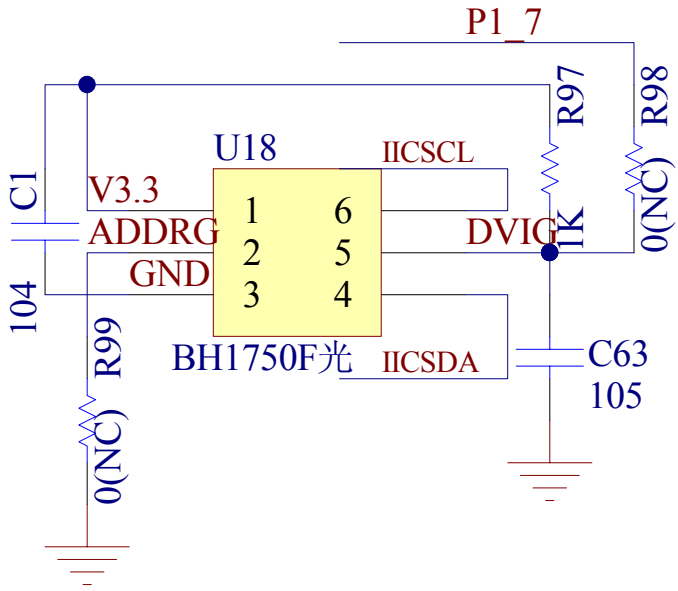

图 3-12 光照传感器与 CC2530 的接线原理图

(3)雨点传感器

雨滴传感器采用日本进口的特殊电子浆料和先进的厚膜技术制作的专门用于监测雨滴的一种新型传感元件。该元件广泛用于需要监测雨滴的各种场所，如：无人职守的机房、宾馆高楼的门窗，高级轿车、客车的门窗，以及各种货场等等的自动控制，以防止雨水的浸蚀。雨滴传感器可以在规定的工作条件下设计在控制的电路做传感之用，以接通各种控制电路。根据传感器的工作电压和电流选取适当的限流电阻以保证其正常工作。将传感器放在适当的位置，保证能在刚下雨时就能接受到雨滴，当传感器接收到雨滴后，发出信号接通控制器，通过控制器使执行机构动作而关好门窗[13]。图 3-13 为该传感器的接线原理。

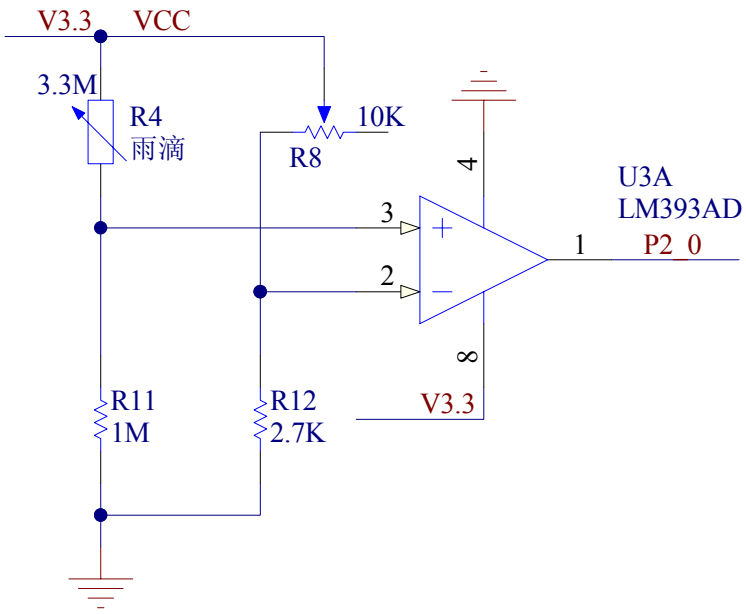

图 3-13 雨滴传感器与 CC2530 的接线原理图

(4)其他常用传感器

其他常用传感器是指已经在市场上广泛应用的传感器，如玻璃破碎传感器、气体传感器、烟雾报警传感器、人体感应等，下面对部分传感器的原理和特性进行说明。

玻璃破碎传感器：PA-456 枫叶玻璃破碎传感器是专门用来探测玻璃破碎的探测器，当不法分子打碎玻璃试图入侵作案时，即可发出报警信号。他的工作原理是将带通放大器的带宽选在 10～15KHz 的范围内（此频率刚好是玻璃破碎时发出的频率），就可将玻璃破碎时产生的高频声音信号取出，从而触发报警。探测器同时具有灵敏度调节功能，根据不同的现场环境，调节其反应灵敏度，以降低外界对探测器的干扰，最终达到较理想的监测效果。

气体传感器：MQ-7 一氧化碳气体传感器所使用的气敏材料是在清洁空气中电导率较低的二氧化锡(SnO2)。采用高低温循环监测方式低温（1.5V 加热）检一氧化碳，传感器的电导率随空气中一氧化碳气体浓度增加而增大，高温 5.0V 加热）清洗低温时吸附的杂散气体。

烟雾报警传感器：离子烟雾报警器有一个电离室，离子室所用放射元素——镅 241（Am241），强度约 0.8 微居里左右，正常状态下处于电场的平衡状态，当有烟尘进入电离室会破坏这种平衡关系，报警电路监测到浓度超过设定的阈值发出报警光电烟雾报警器内有一个光学迷宫，安装有红外对管，无烟时红外接收管

收不到红外发射管发出的红外光，当烟尘进入光学迷宫时，通过折射、反射，接收管接收到红外光，智能报警电路判断是否超过阈值，如果超过发出警报。

3.5 本章小结

本章系统地介绍了智能家居系统的主控制器、无线收发模块和节点的硬件结构。首先介绍了系统主控制器的硬件资源，这些资源已完全满足智能家居系统的硬件要求。接着介绍了无线收发模块的硬件结构和引脚说明，便于与传感器的接口对应。最后从节点的分类、结构到传感器的选择，详细介绍了无线节点的硬件结构。

第 4 章 智能家居系统软件设计

4.1 软件设计概述

在智能家居系统中，无论是终端节点还是主控制器，都需要软件系统的支持。在终端节点与主控制器通信过程中，为了实现数据的能够在终端节点和协调器之间准确无误的传输，需要设计一定的数据格式进行通信。而对于主控制器来说，我们需要设计整个智能家居软件系统。

4.1.1 节点软件整体设计

本系统中所使用的节点所使用的软件结构为 Z-Stack+OSAL 操作系统。Z-Stack 是德州仪器在 2007 年 4 月推出业界领先的 Zigbee 协议栈。Z-Stack 符合 Zigbee 2006 规范，支持多种平台，包括基于 CC2530 收发器以及 TI MSP430 超低功耗单片机的平台等。Z-Stack 包含了网状网络拓扑的几近于全功能的协议栈，在竞争激烈的 Zigbee 领域占有很重要地位。OSAL 可以看做是一种机制,一种任务分配资源的机制,从而形成了一个简单多任务的操作系统[14]。

Z-stack 协议栈中的无线接收和无线发送过程如下[15,16]：

(1)无线接收

当有数据通过无线发送到应用层时，应用层会发送 1 个 AF_INCOMING_MSG_CMD 消息事件。

```
case AF_INCOMING_MSG_CMD:
Gener icApp_Messag eMSGCB( MSGpkt ) ;
break;
```

这里表示收到 AF_INCOMING_MSG_CMD 消息事件, 然后调用收到消息事件的信息处理函数 GenericApp_MessageMSGCB(MSGpkt) , 开始接收数据并通过调用串口 HalUARTWrite(uint8 port , uint8 * buf, uint16 len)写函数发送接收到的数据。

(2) 无线发送

在串口回调函数中当串口有数据输入时，应用层会发送 1 个 GENERICAPP_SEND_MSG_EVT 消息事件。

```
if ( events & GENERICAPP_SEND_MSG_EVT ) {
GenericApp_SendTheMessage( ) ;
return ( events^ GENERICAPP_SEND_MSG_EVT) ;
}
```

调用 GenericApp_SendTheMessage 数据发送函数,即为 AF_DataRequest 函数,具体形式如下:

```
afStatus_t AF_DataRequest(
    afAddrType_t * dstAddr,
    endPointDesc_t * srcEP,
    uint16 cID,
    uint16 len,
    uint8 * buf,
    uint8 * transID,
    uint8 opt ions,
uint8 radius )
```

本系统中 Zigbee 节点软件的结构图如图 4-1 所示，这里我们主要通过修改 Application Layer 层来实现系统所要完成的功能。

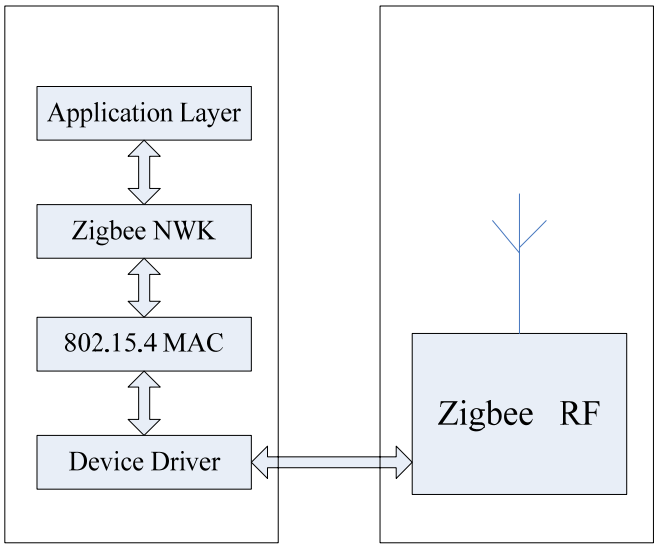

图 4-1 Zigbee 节点软件的结构图

4.1.2 智能家居系统软件整体设计

软件整体结构基本上符合 MVC 模式[17,18]。MVC 是一个设计模式，它强制性的使应用程序的输入、处理和输出分开。使用 MVC 应用程序被分成三个核心部件：模型（M）、视图（V）、控制器（C）。而 Android 下的应用程序可采用分层的模块化结构设计，分为表示层，控制层，业务层以及数据处理，其软

件总体架构如图 4-2 所示。

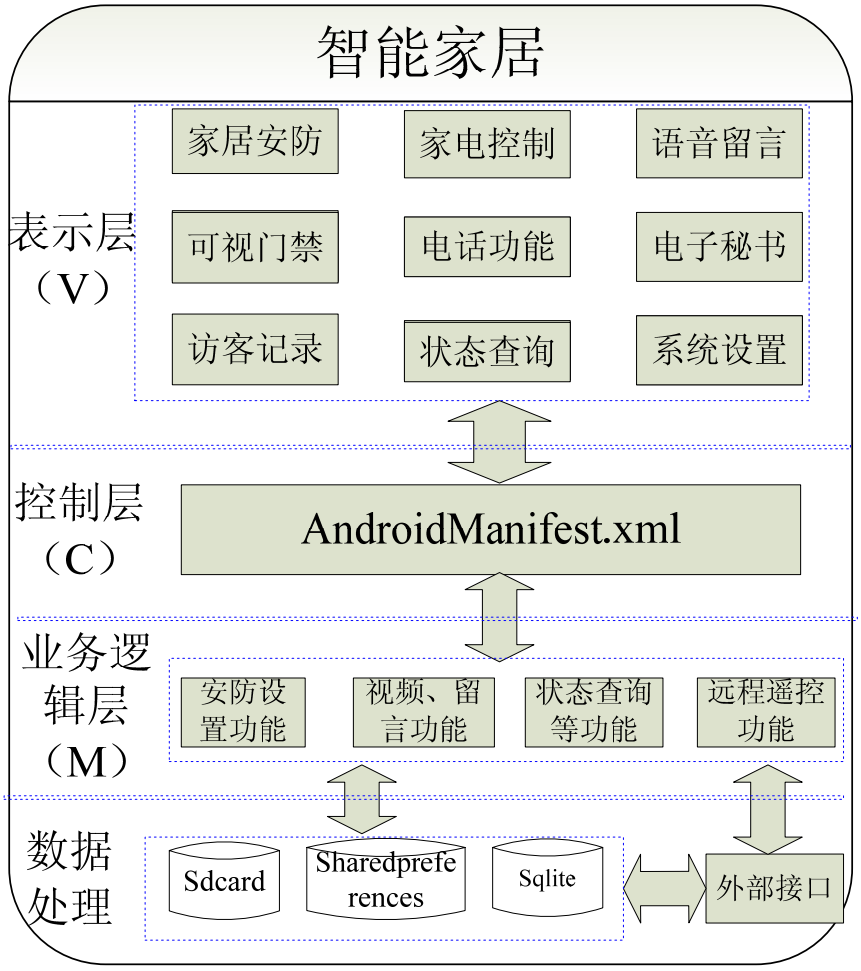

图 4-2 智能家居系统软件层次架构图

表示层：展现给用户图形操作界面，用于显示模型的状态信息、控制信息以及其他信息。这里向用户展示了智能家居系统中个模块的操作界面。用户操作的每个界面都对应于一个 Activity，其实现方法为 layout 下的 XML 布局文件生成以及在程序代码中硬实现。

控制层：负责表示层和业务逻辑层之间的流程控制。在 Android 系统中主要是通过监听事件、Activity 之间的跳转以及 Androidmanifest.xml 的控制信息完成。AndroidManifest.xml 是每个 Android 程序中必须的文件。它位于整个项目的根目录，描述了 package 中暴露的组件（activities, services, 等等），他们各自的实现类，各种能被处理的数据和启动位置。除了能声明程序中的四大组件（Activities, ContentProviders, Services, 和 Intent Receivers），还能指定 permissions 和 instrumentation（安全控制和测试）。

业务逻辑层：对于接受的数据进行相应的处理。这里主要通过 Android 中的

Activity 和一些专门负责数据处理的类实现。Activity 会将处理后的结果显示给用户，并且将需要记录的数据写入相应的存储器中。

数据处理：对 Sharedpreferences、数据库 Sqlite，文件进行操作。这里对于系统中的各种状态都使用 SharedPreferences 存储，而对于像需要存储空间大的视频等数据都存在 Sdcard 中。

4.2 节点软件详细设计

4.2.1 无线节点开发环境搭建

TI 官网上为我们提供了不同版本的 Z-stack 所对应的 IAR 版本。我们所选用的无线收发模块是 CC2530，与之对应的版本为 IAR EW8051 7.51A。IAR EW 集成开发环境支持多达 35 种以上的 8 位/16 位/32 位的 ARM 结构处理器。为了能够在 IAR 上进行 CC2530 的开发，还需要对其进行必要的设置。这里应该注意以下几点：

(1)在新建工程中，选择 8051 作为该工程工具链。这是因为 CC2530 内置的是微处理器是 8051。

(2)设置工程选项参数。这里为了使IAR所编译的目标文件能在CC2530中运行，需要设置一些参数，比如device项应选择“devices/Texas Instruments/CC2530.i51”；数据格式采用“大端”等等。

此外还需要的软件为 SmartRF Flash programmer，用于测试和烧写 CC2530 程序。对于 IAR 和 SmartRF Flash programmer 安装和使用这里不再做详细的介绍。

节点软件设计是基于 TI / Chipcon 公司免费提供的 Zigbee2006 协议栈，以 ZStack-CC2530-2.3.1-1.4.1 版本中 GenericApp 例程为基础。GenericApp 例子基本功能很齐全，而且在 Z-Stack 上实现了无线网络数据传输。例程没有多余的功能，所以是典型的 Z-Stack 模板，即为用户提供了一个通用模板，可以通过它建立自己的应用程序。因此，本系统中节点软件设计主要是通过应用层中的主要是改动应用层中的 GenericApp.c 文件实现的。

4.2.2 Zigbee 数据传输格式

1．Zigbee 帧格式设计

无论是从主节点到终端节点，还是从终端节点到主节点传送数据，为了使节点之间能够识别出双方所发信息的含义，就需要制定 Zigbee 数据传输时的格式。根据信息功能的不同，这里可以对 Zigbee 传输数据分成以下 2 种类型：

数据请求帧：当需要查询前端传感器的状态或者是要对家用电器进行控制时，都需要发送该种帧。

数据应答帧：当发送了数据请求帧后，需要有返回信息要求的节点则发送该种帧[19]。

表4.1 主控制器与节点通信帧格式

格式说明	标志	长度	父节点地址	原地地址	类型	数据	校验和
占用byte数	1	M	2	2	1	N	1

下面对此帧格式进行详细说明：

标志：共占用 1 个字节，标明该帧的传输方式。

长度：共占用 M 个字节，M 的值由该帧的类型、数值和校验和这三者决定的，其值为该三者之和。

父节点地址：该地址是相对于接收数据的节点来说的，在本系统中即为协调器节点地址。

原地地址：该地址为标示数据发送的节点地址，通常为终端节点地址。

类型：标示了该节点属于那类节点。

数据：记录了传感器采集到的数据或者是用于家电控制的各种编码。数值 N 是由前端传感器所采集数据所决定的

校验和：用来检验帧数据的正确性，其值为类型与数据和值的补码。

例如温湿度传感器所传的值需要占用 4 个 byte，N=4。而类型和校验和各占一个字节，所以相对温度来说，M=6。其具体的帧格式为：

0xFD, 0x06, 0x00,0x00, 0x00,0x01, A, 0x00,0x00(湿度)%RH ,0x00,0x00(温度)， 0x00(校验和)

表 4.2 和 4.3 分别列出了本系统中传感器及家用电器所使用的帧格式，其中在表中，类型 1 与类型 2 与 Zigbee 帧格式中的数据项对应，这里的类型 2 分别代表了具体的功能。这里还需要说明的是对于电视和空调的控制，还需要将该命令对应到红外命令中才能完成具体的控制功能。

表 4.2 传感器帧格式

传感器	长度	类型
温湿度	0X06	0XA1
光照	0X03	0XB1
雨滴	0X03	0XC1
玻璃	0X03	0XD1
气体	0X03	0XE1
烟雾	0X03	0XF1
人体	0X03	0XG1

表 4.3 家用电器帧格式

电器	类型1	类型2	功能说明
电饭煲	0XH1	0X01 0X02	开 关
电灯	0XJ1	0X01 0X02	开 关
窗户	0XL1	0X01 0X02	开 关

表 4.3(续 1) 家用电器帧格式

电器	类型1	类型2	功能说明
空调	0XI1	0X01 0X02	开 关
		0X03 0X04	制冷 制热
		0X05	抽湿

表 4.3(续 2) 家用电器帧格式

电器	类型1	类型2	功能说明
电视	0XK1	0X01 0X02	开 关
		0X03 0X04	上一个 下一个
		0X05 0X06	调高 调低

2．Zigbee 数据传送方式

Zigbee 模块数据传输功能非常简单易用，有二种数据传送方式：

(1)数据透明传输方式：

只要传送的第一个字节不是 0xFE，0xFD 或 0xFC，则自动进入数据透明传输方式；Coordinator 从串口接收到的数据，会自动发送给所有的节点；某个节点从串口接收到的数据，会自动发送到 Coordinator。

(2)点对点数据传输方式：

Zigbee 网络内的任意节点之间，可通过点对点传输指令，传送数据；指令格式：

0xFD + 数据长度 + 目标地址 + 数据

所以，我们在执行特定节点操作时都是使用的点对点数据传输方式，而对于状态查询操作则采用的是透明传输。对于透明传输将会在后面的章节中介绍。

4.2.3 节点软件设计

1．主节点软件设计

主节点即 Zigbee 网络中的协调器，它的主要功能是发起网络，分配网络地址，实现绑定，接受/发送信息等事件。组建一个完整的 Zigbee 网状网络包括两个步骤：网络初始化、节点加入网络。

在网络初始化中，Zigbee 网络的建立是由网络协调器发起的，任何一个 Zigbee 节点要组建一个网络必须要满足以下两点要求：节点是 FFD 节点，具备 Zigbee 协调器的能力；节点还没有与其他网络连接，当节点已经与其他网络连接时，此节点只能作为该网络的子节点，因为一个 Zigbee 网络中有且只有一个网络协调器[20,21,22]。本系统设计的采用网络拓扑结构的是星型拓扑结构。

节点加入网络是指终端节点通过与协调器连接入网。其基本过程为：查找网络协调器➔发送关联请求命令➔等待协调器处理➔发送数据请求命令➔回复。具体的组网过程这里不再介绍，请参考相关资料。下图 4-3 为主节点工作流程图：

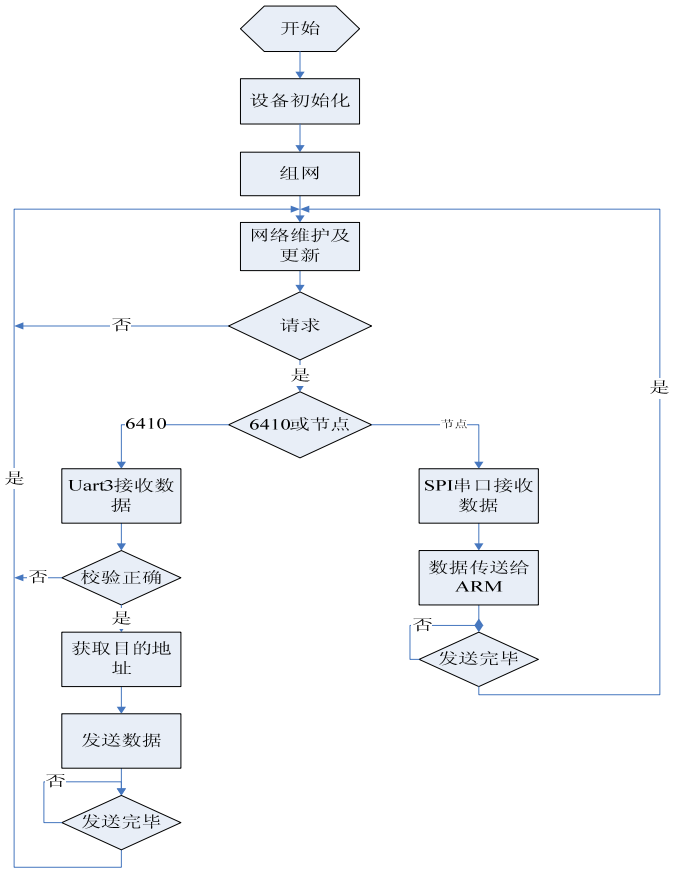

图 4-3 主节点工作流程图

当主机上的协调器从空中捕获到信号后，在应用层上首先收到信息的就是 GenericApp_ProcessEvent 这个函数了，它收到一个AF_INCOMING_MSG_CMD 的事件，并通知GenericApp_ProcessMSGCB，这样就将从空中获取到的信息，传

给了串口Uart3。GenericApp_ProcessMSGCB中的主要代码：

```
void GenericApp_MessageMSGCB( afIncomingMSGPacket_t *pkt )
{
  switch ( pkt->clusterId )
  {
    case GENERICAPP_CLUSTERID:
      // "the" message
#if defined(SERIAL_DEBUG_SUPPORTED)
      HalUARTWrite(HAL_UART_PORT_0, (uint8*)pkt->cmd.Data,pkt->cmd.Data[1]+6);
      HalLedSet ( HAL_LED_1, HAL_LED_MODE_TOGGLE );
#elif defined( WIN32 )
      WPRINTSTR( pkt->cmd.Data );
#endif
       break;
  }
}
```

2．终端节点软件设计

图 4-4 为终端节点通用流程图，从图中可以看到环境监测节点和执行节点的工作流程。

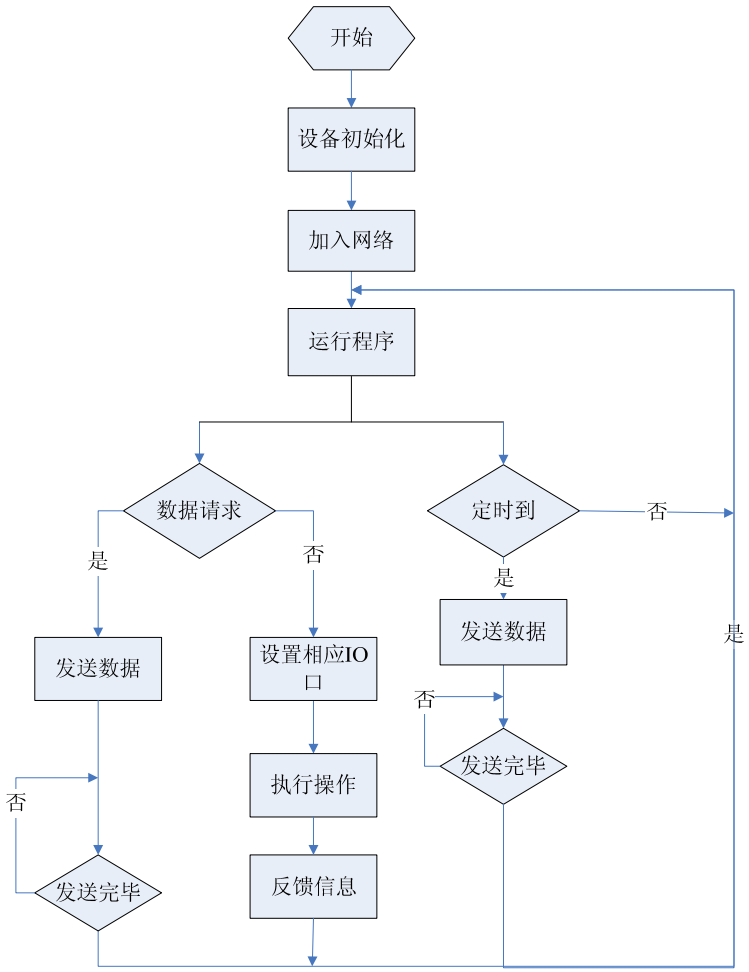

图4-4 终端节点通用流程图

(1)状态查询节点软件实现

状态查询节点主要用于监测家里的环境状态，他们的其实现原理大致相同，所以我们只以其中一种传感器进行说明。这里以温湿度传感器为例说明如何实现发送数据请求帧。在第 3 章 3.3.2 传感器选择这一小节中详细介绍了 SHT10 传感器的硬件特性。我们在使用 SHT10 时，就要了解它的各种命令以及相对应的时序[23,24]，只有我们正确使用了该时序才能读取正确的数值。比如我们常用的命令有测量温度、测量湿度、读或写状态寄存器以及复位命令等。在本系统中各种命令的实现函数的文件是 SHT1X.c。

下面代码是 Zigbee 工程文件 GenericApp.c 中的 GenericApp_application 函数，它主要负责温湿度数据请求帧的组织，并通过 AF_DataRequest 函数发送到空气中，具体实现如下：

```
void GenericApp_application( void )
{
  Send_data[2]=NLME_GetCoordShortAddr()/256; //父节点地址
```

```
  Send_data[3]=NLME_GetCoordShortAddr()%256; //父节点地址
  Send_data[4]= NLME_GetShortAddr()/256;
  Send_data[5]= NLME_GetShortAddr()%256;
  error=0;
      error+=s_measure( &Send_data[7],&checksum,HUMI);    //measure humidity
      error+=s_measure( &Send_data[9],&checksum,TEMP);    //measure temperature
      if(error!=0) s_connectionreset();                    //in case of an error: connection
      else
      { humi_val.i=Send_data[7]*256+Send_data[8];
        temp_val.i=Send_data[9]*256+Send_data[10];
        humi_val.f=(float)humi_val.i;                      //converts integer to float
        temp_val.f=(float)temp_val.i;                      //converts integer to float
        calc_dht90(&humi_val.f,&temp_val.f);               //calculate humidity, temperature
  Send_data[6]=0XA1;//传感器类型
  data=(uint)(100*humi_val.f);
  Send_data[7]=(uint8)(data/ 100);
  Send_data[8]=(uint8)(data % 100);
  data=(uint)(100*temp_val.f);
  Send_data[9]=(uint8)(data / 100);
  Send_data[10]=(uint8)(data % 100);

Send_data[11]=(Send_data[6]+Send_data[7]+Send_data[8]+Send_data[9]+Send_data[10])%256;

      if ( AF_DataRequest( &GenericApp_DstAddr, &GenericApp_epDesc,
                           GENERICAPP_CLUSTERID,
                           (Send_data[1]+6),
                           Send_data,
                           &GenericApp_TransID,
                           AF_DISCV_ROUTE,      AF_DEFAULT_RADIUS      )      ==
afStatus_SUCCESS );
}
}
```

(2)执行节点软件实现

执行节点主要用于控制系统中的家用电器[25]，其实现原理大致相同，所以我们只以其中一种家电进行说明。这里以控制空调的执行节点为例进行说明，当该节点收到主机发来的帧后，在 GenericApp_MessageMSGCB 函数中进行处理，然后将该信息传递到红外遥控模块中进行相应的处理。下面给出空调处理的主要代码：

```
    if((pkt->cmd.Data[0]==0xFD)&&(pkt->cmd.Data[6]=='H'))
      { //send_data4,5=0xFF,0xFF
          if((pkt->cmd.Data[7]==Send_data[4])&&(pkt->cmd.Data[8]==Send_data[5]))
```

```
        {GenericApp_applicationbuf=1;
         if(pkt->cmd.Data[9]==0xI1)//控制空调
           {
              if(pkt->cmd.Data[10]==0x01)
                 P1_2=1; //开
              if(pkt->cmd.Data[10]==0x02)
                 P1_3=1; //  关
              if(pkt->cmd.Data[10]==0x03)
                 P1_4=1; //制冷
              if(pkt->cmd.Data[10]==0x04)
                 P1_5=1; //  制热
              if(pkt->cmd.Data[10]==0x04)
                 P1_6=1; //  抽湿
           }
}
```

4.3 智能家居系统软件设计

在第 2 章中我们已经分析了智能家居系统所要实现的功能，本节首先对智能家居系统中的各模块功能进行分析，然后在此基础上对各模块进行详细的设计。

4.3.1 各模块功能分析

1．家居安防模块功能分析

家居安防模块主要是情景模式选择和对已选择的模式进行撤销或切换等操作，当用户选择了一种情景模式后即可查看该模式下传感器的工作状态。用例如图 4-5 所示。

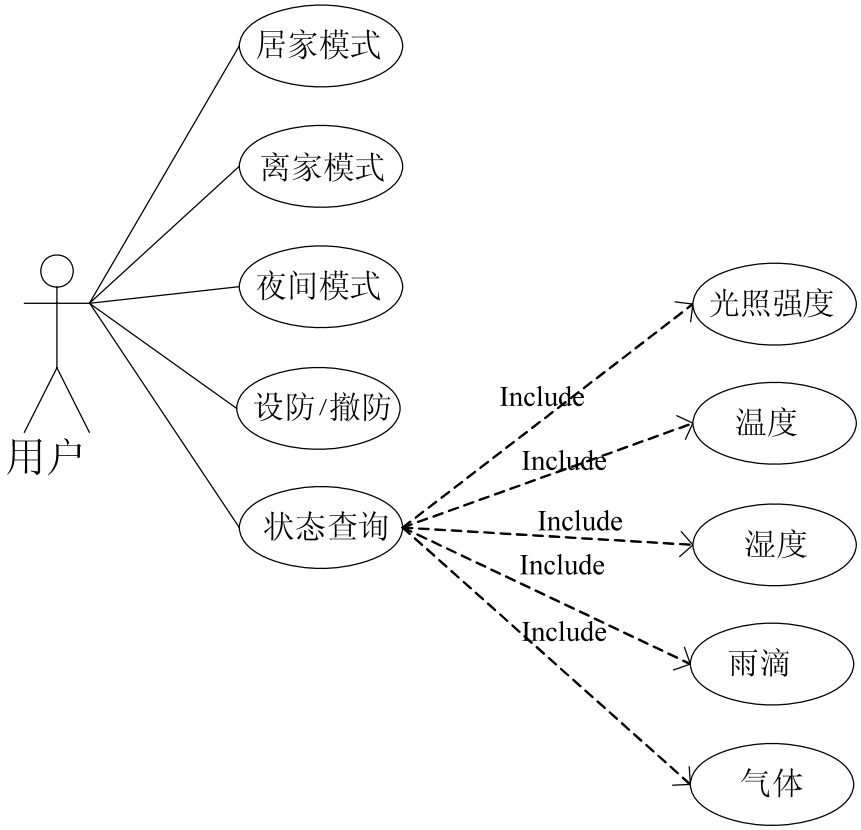

图 4-5 家居安防模块用例图

如图 4-6 所示，本软件系统有三种可选择的情景模式，当用户选择相应的模式后即可进行设防/撤防和状态查询操作。图 4-7 为用户设置了居家模式后单击了“状态显示”后的主界面。

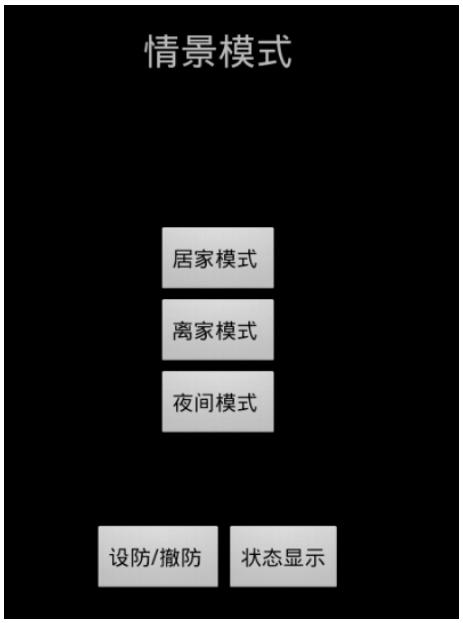

图 4-6 情景模式界面

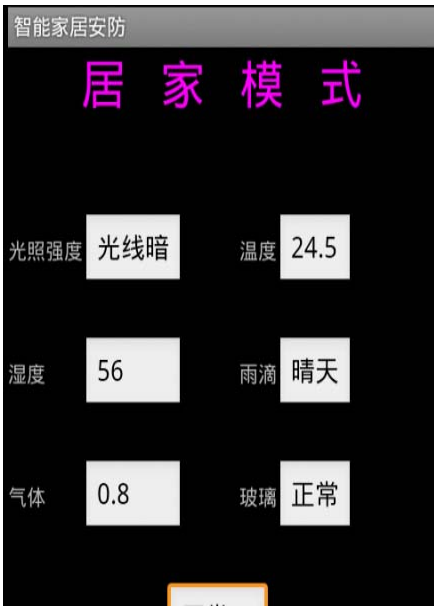

图 4-7 居家模式的状态显示主界面

2．家电控制模块功能分析

家电控制是智能家居系统中一个重要的组成部分。用户可以通过该模块对家

里的各种电器进行统一管理，这里我们所实现的功能都是常用家电的功能，其用例如图 4-8 所示。

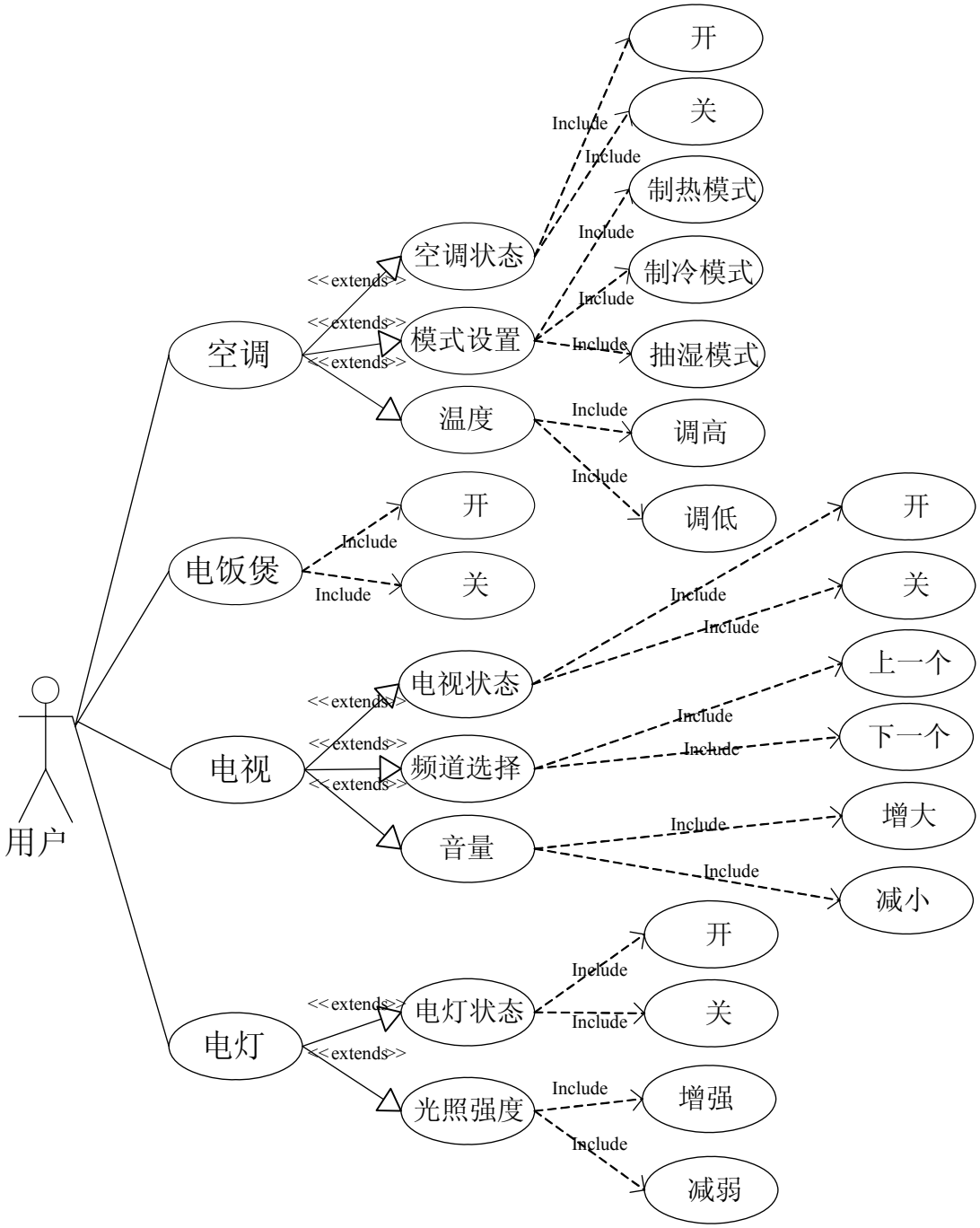

图 4-8 家电控制模块用例图

当用户选择任意一个标签后即可显示当前可操作的按钮，这些按钮用于控制与标签相对应得家电。由于空调和电视机上的功能众多，全部实现比较困难，因此这里根据用户习惯，对常用的功能进行实现，下面对各家电的功能分别介绍：

空调：最主要的功能就是制冷和制热，其次是抽湿。因此在本系统中只实现这三种功能，而温度选项是相对于制冷和制热模式而言的对与抽湿模式不起作用。此外还可控制空调的开和关。

电饭煲：这里对电饭煲的控制只有两种状态：开和关。

电视：电视遥控器的主要功能是开关电视，更换频道以及调节音量大小等，这也是在本系统所要实现的基本功能。

家电控制的基本界面如图 4-9 所示：

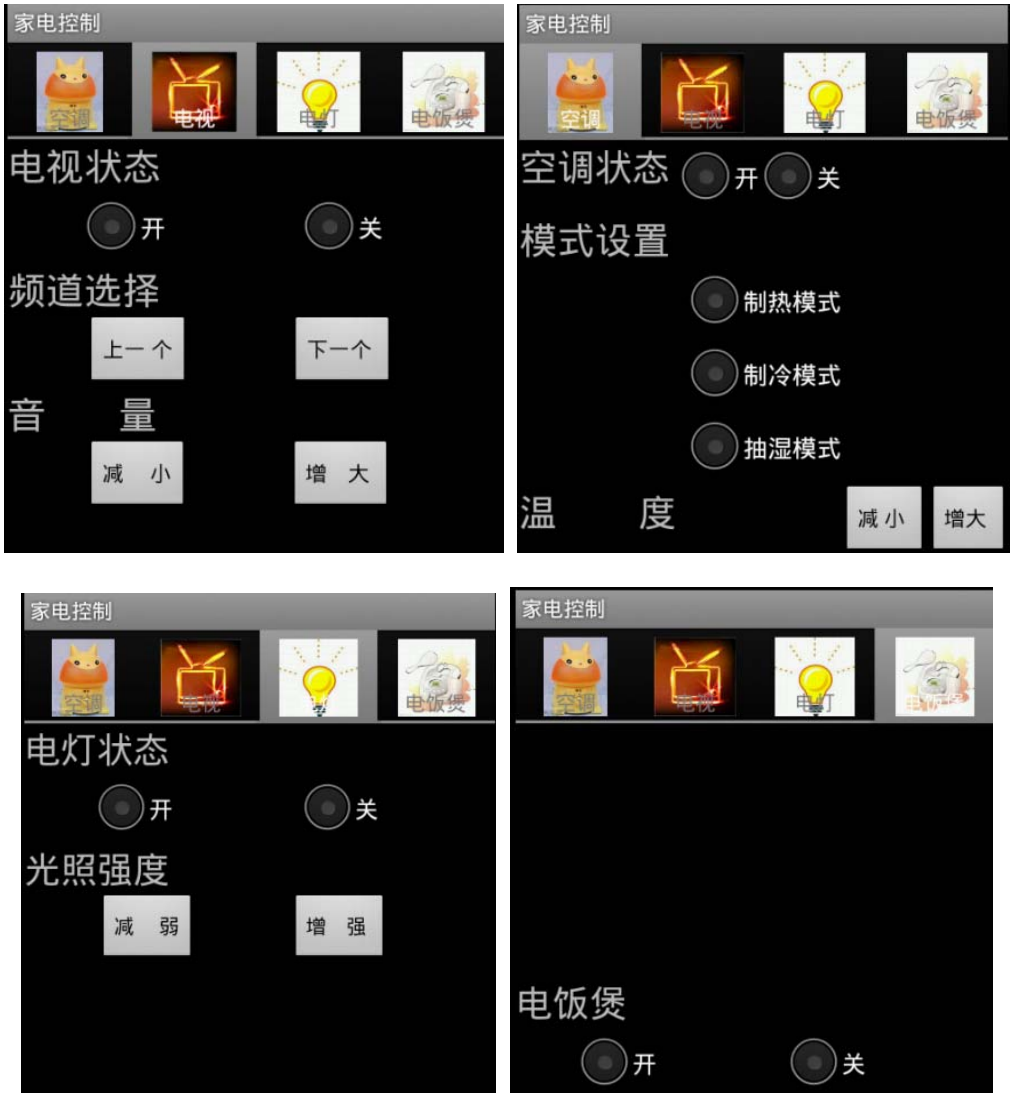

图 4-9 家电控制的基本界面

3．电话功能模块功能分析

电话功能模块的作用除了常用的拨打电话、发送短信等功能外，在智能家居系统中还用它可以设置报警功能、短信控制家电以及查询传感器工作状态等功能。用例图如图 4-10 所示：

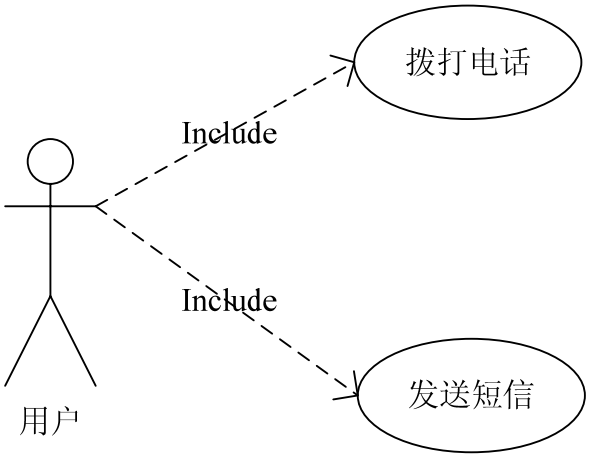

图 4-10 电话功能模块用例图

图 4-11 电话功能模块的基本界面

当用户点击了主界面中的“电话功能”按钮后，系统就会跳转到电话功能主界面。电话功能模块的基本界面如上图 4-11 所示。

4．状态检测模块功能分析

这个模块的功能主要是查看传感器和各种控制模块的工作状态，动态显示各模块是否能正常工作，用例图如图 4-12 所示。状态检测模块的基本界面如上图 4-13 所示。

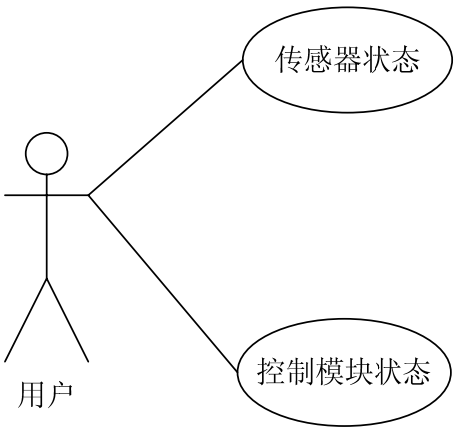

图 4-12 状态检测模块用例图

图 4-13 状态检测模块的基本界面

5．系统设置模块功能分析

系统设置主要为其他功能模块提供必要的设置信息，通过初始化和设置一些相关的参数，使得系统运行在用户所指定的最佳模式。系统设置主要功能是设置

情景模式、学习家电命令、为家电控制设置短信命令，设置报警时所要拨打或发短信的手机号码以及系统一些常规设置，包括系统的声音、系统时间设置等，用例图如图 4-14 所示。系统设置的主界面如图 4-15 所示。

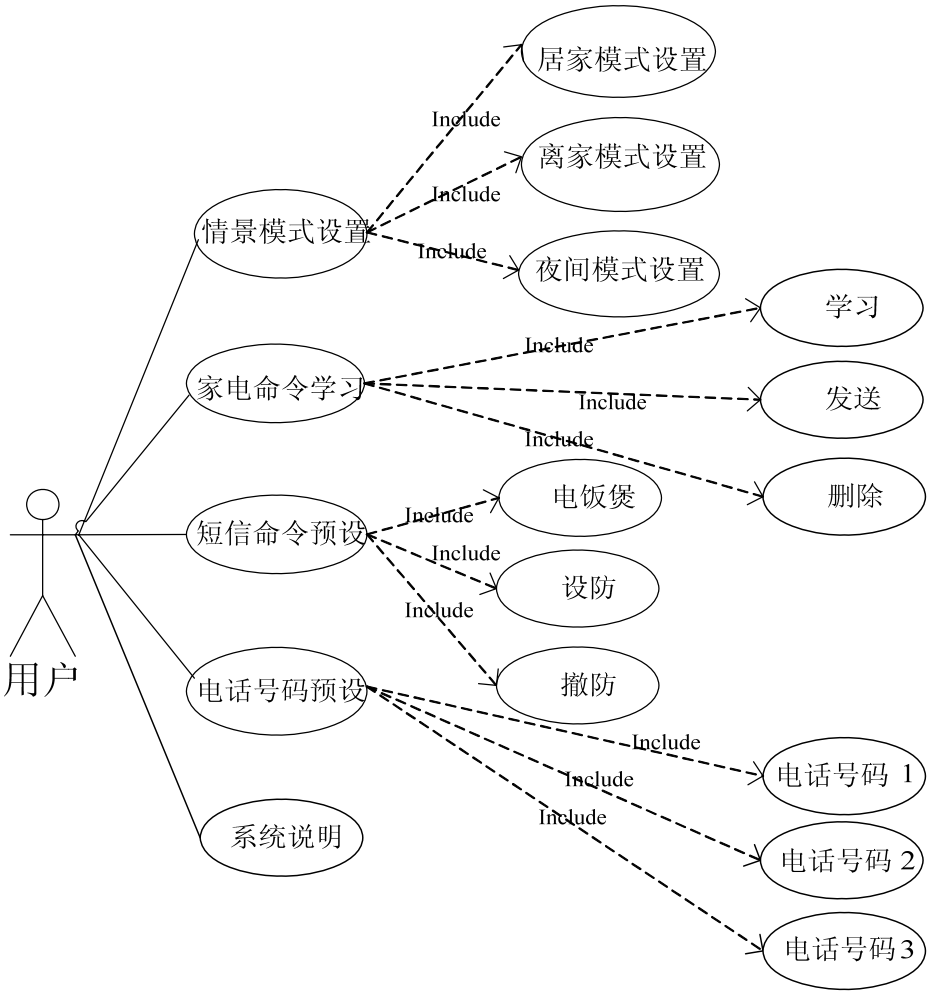

图 4-14 系统设置模块用例图

情景模式设置：它的主要功能是分别配置 3 种情景模式的各自的组合状态，使其工作在不同的情景模式下。如图 4-16 所示，在居家模式下用户分别可以对家电和传感器选择。当用户同时选择了空调和温湿度传感器后，并且在家居安防模块选择了居家模式后，系统就进入了连动状态。若室内温度低于系统所设定的温度后就会开启空调进入制热模式。

家电命令学习：这里通过点击“学习”按钮，发送命令给红外遥控模块，红外遥控即可学习遥控器命令，然后通过“发送”按钮验证该命令是否学习成功。选中其中一条命令，点击“删除”按钮，即可删除该命令。

短信命令预设：它的主要功能是通过短信控制和查询各种信息。我们可以为空调、电饭煲等设置开关命令；为各种情景模式设置布防和撤防命令；还可以查

询各个传感器的工作状态。

电话号码预设：这里我们设置了 5 路电话号码。若前端传感器检测到异常则会根据系统设定的号码循环拨打这 5 路电话号码，直到接通为止，接通后会播放系统预先设定好的语音记录，并通过短信通知用户。

主机设置：这里我们可以设置系统的音量以及系统的时间日期，以便在系统出现问题时能及时调整。

系统说明：它是系统的整体说明，包括系统的所用的开发环境，版本等信息。

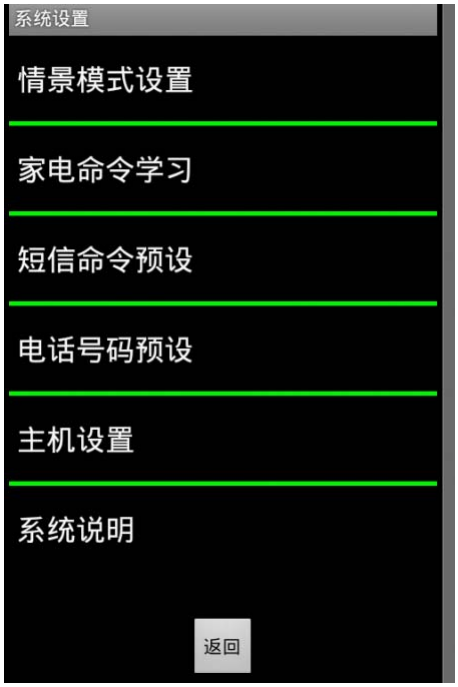

图 4-15 系统设置模块主界面

图 4-16 系统设置模块主界面

6．其他模块功能简要分析

可视门禁和访客记录：这两个模块的主要功能是通过可视门禁功能通知用户是否有客人来访，访客记录是将客人来访的情况一一记录，以供用户调阅。

电子秘书和语音留言：电子秘书主要用来提醒用户在特定的时间该做的事情；语音留言时当用户不在家时，当有人拨打该主机号码时，就会进入语音留言功能。

4.3.2 软件详细设计

1．家居安防模块设计

家居安防是整个系统的核心。为了实现上述功能，设计此模块的层次结构如

图 4-17 所示：

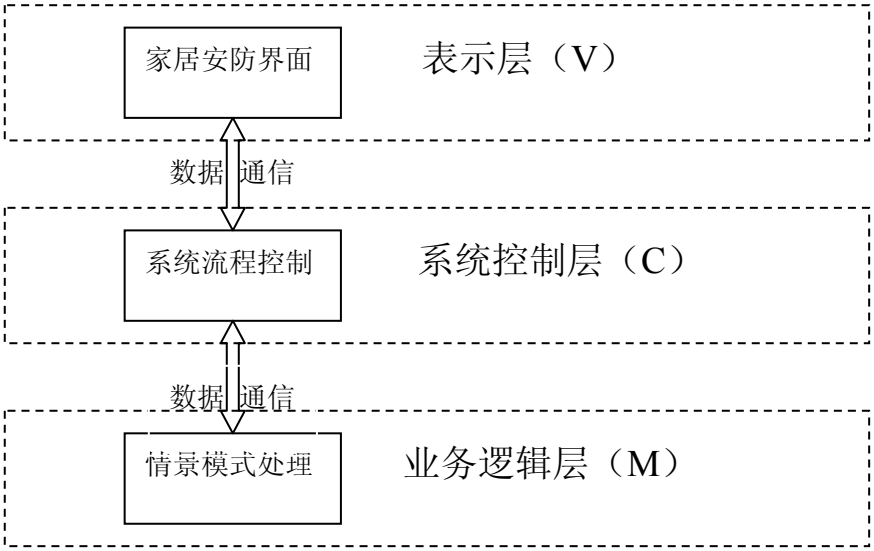

图 4-17 家居安防模块层次结构图

业务逻辑层：接收用户选择的数据，对相应的数据进行分析处理。这里涉及到类有 contextualdisplay、dataserver、athomedisplay、leavedisplay 和 nightdisplay 等。下面分别对家居安防中各个类的功能进行介绍：

Contextualdisplay 类：它是一个 Activity，用于处理用户选择的按钮，以及各 activity 之间的跳转。

Dataserver 类：这是一个服务类，它继承于 SerialPortService 类，这里是数据的处理中心。

Athomedisplay、leavedisplay 和 nightdisplay 类：这三个类都是 Activity 类，当用户设置了相应的情景模式后，若单击“状态查询”按钮，就会跳转到相应的状态显示，实时显示当前传感器的所测得的数据。例如用户设定了居家模式后，单击“状态查询”，系统就会跳转到 athomedisplay 界面中。

2．家电控制模块设计

这里首先需要对家庭常用的家电控制方式进行说明。对于不同类型的家庭电器，存在不同的控制方式，像电灯、电动窗帘、电饭煲这些家电，都只是给一个开关命令，通过执行节点来控制，比较容易实现；有需要专门的模块来控制，比如电视机、空调等这些有遥控器的家电，因为他们的功能比较多，因而需要通过万能遥控器先学习遥控器本身的命令然后通过该遥控器控制家电；而有些智能家电像智能冰箱因为其本身提供网络接入功能，只要接到家庭网络中就能实现远程

遥控。

在本系统中只涉及到前面两种家电的控制，分别是电视、空调、电灯及电饭煲。下面详细介绍这两类家电的控制方式：

(1)对于电饭煲、电灯这一类的开关量家庭电器，我们使用的是以继电器为核心的执行节点来控制，具体的硬件电路见第三章的 3.3.1 环境检测节点和执行节点。具体原理如图 4-18 所示。图形按钮、控制组件对应于家电控制中各种开关按钮和单选按钮，用户若点击了相应的按钮，则系统检测到该信息后将信息传递到处理该信息的服务中，发送相应的控制命令。当命令经过 Zigbee 网络传递到执行节点后，根据命令格式解码，设置相应的 IO 口输出高低电平，进而控制继电器的吸合、断开，然后由继电器来控制家用电器的开关。

(2)对于空调、电视机这类具有红外遥控器的家家电，我们可以通过万能红外遥控器控制其具体的设计方案如图 4-19 所示。这里涉及到两次编码：一种是空调、电视各种控制按钮与后台的各种发送命令之间的编码，一种是后台的发送命令与红外模块存储的命令之间的编码，这两种编码其实就是要找到一个对应关系，比如我们要开电视，当用户选择了开按钮后，监听器就会将该事件通过一种对应关系传递到相应的服务程序中，服务程序识别出这是一条开电视的信息，就会执行相应的开电视命令，将此命令发送到红外模块上，红外模块根据对应关系，找到该命令对应的红外命令，然后发送该命令即可。这里需要说明的是，在使用空调电视相关功能之前，首先需要通过系统设置中的“家电命令学习”模块，学习该家电的控制命令，然后存储到该红外模块中[26]。

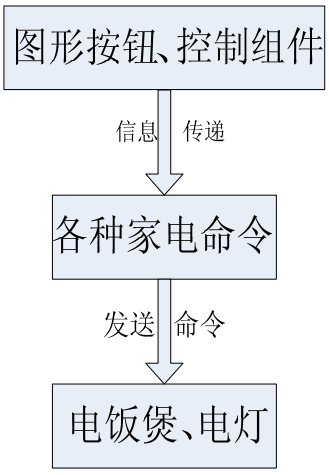

图 4-18 具有开关量家电控制原理图

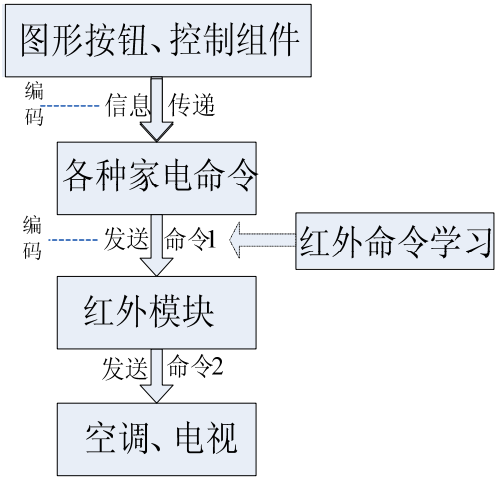

图 4-19 使用红外模块的家电控制原理图

3．电话功能模块设计

GPRS 模块是通过 Uart 连接到处理器的，因此该模块与主系统主要通过串口通信，需要设置、编写相应的串口程序。而且该模块实际上就是手机模块，所以需要查看相关手机模块的 AT 指令[27,28,29]。它能实现收发短信，拨打电话，发送彩信等功能，可作为实时监控报警系统的无线终端。为了实现 GPRS 在本系统中的报警和电话功能，在该模块中应该至少需要设计 3 个相关的类：GPRS 类、telephonecomm 类及 telecallmain 类。GPRS 类用来实现手机模块的基本功能(主要用来发送、接收 AT 指令)；telephonecomm 类是用来实现手机模块与主系统的串口通信（主要用来实现对家庭安防里与手机功能相关功能的实现）；telecallmain 类主要实现地是系统里的电话功能，通过该功能，用户可以直接拨打电话。电话功能模块类之间的关系如图 4-20 所示：

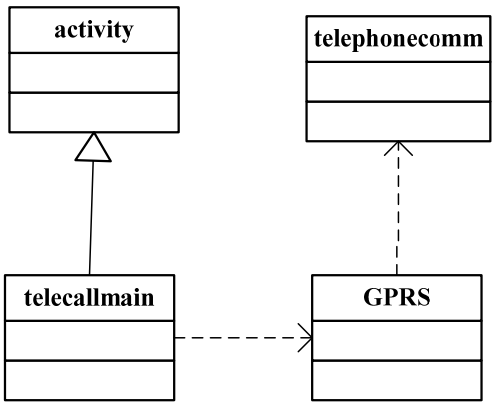

图 4-20 电话功能模块类图

Telecallmain 类：它是一个 Activity 类，继承自 Activity，该类负责电话功能图形界面显示。当用户拨打电话号码后，将相应的数据传递到 GPRS 类中处理。

GPRS 类：该类主要包括、整个 GPRS 模块的初始化、AT 命令的处理等功能。

Telephonecomm 类：该类负责数据的接收和发送。当用户发送 AT 命令后，就会通过 telephonecomm 类中相关函数发送到串口中。

4．状态检测模块设计

该模块主要用来检测传感器及控制模块的工作状态。那么为了实现这一功能，我们可以在设计节点命令中设计一条特殊命令：当主机发送该命令后，具有 Zigbee 模块的节点都要返回一条自己工作状态命令，这条命令就相当于网络中广播。该界面的布局文件为 conditionctlmain.xml，而 conditionctlmain.java 负责该界

面的显示。

5．系统设置模块设计

系统设置是该系统的关键模块，只有用户相关功能进行设置后才能使系统进入正常工作。该模块的结构框图如下图 4-21 所示：

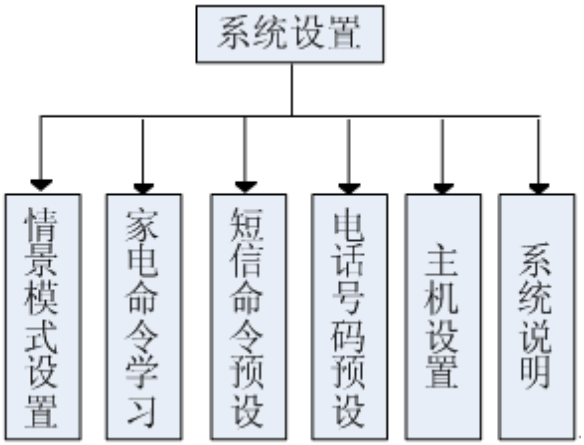

图 4-21 系统设置结构框图

系统设置界面用到的类为 systemsetmain.java。这里使用 ListView 组件为系统设置和情景模式设置设计图形界面。创建 ListView 有两种方式：一种是直接使用 ListView 进行创建，一种是让 Activity 继承 ListActivity。无论哪种哪种方式创建，都要为 ListView 设置它要显示的列表，内容 Adapter 就是负责提供需要显示的列表项。

- 情景模式设置和短信命令预设

在系统设置中我们需要保存一些系统配置选项和各种状态，SharedPreferences 是 Android 平台上一个轻量级的存储类，主要是保存一些常用的配置比如窗口状态，一般在 Activity 中重载窗口状态 onSaveInstanceState 保存一般使用 SharedPreferences 完成，它提供了 Android 平台常规的 Long 长整形、Int 整形、String 字符串型的保存。在情景模式中我们需要保存的是三种模式的选择状态相当于 Int 整形量，在短信命令预设中，我们要存储是一条条简单命令格式，相当于字符串变量，故这两个模块都可以使用这种方式进行存储信息。在 Android 系统中，这些信息以 XML 文件的形式保存在 /data/data/PACKAGE_NAME /shared_prefs 目录下。当点击某个 Preference 时的调用流程是 AdapterView.performItemClick－－>PreferenceScreen.onItemClick－

－>Preference.performClick －－>PreferenceActivity.onPreferenceTreeClick，当 onPreferenceTreeClick 返回 true 的时候就直接 return 了，没有走下面启动 Activity 的地方了，因此要使一个 Preference 能够正常跳转到另外一个 Activity 有两个条件，一是 xml 里面是否设置正确，第二是调用该 xml 的 java 类是否在 onPreferenceTreeClick 这个函数需要返回 false[30,31,32]。

- 家电命令学习

这个模块主要是针对带有红外遥控的模块而言的，该模块一条命令的学习流程如下图 4-22 所示：

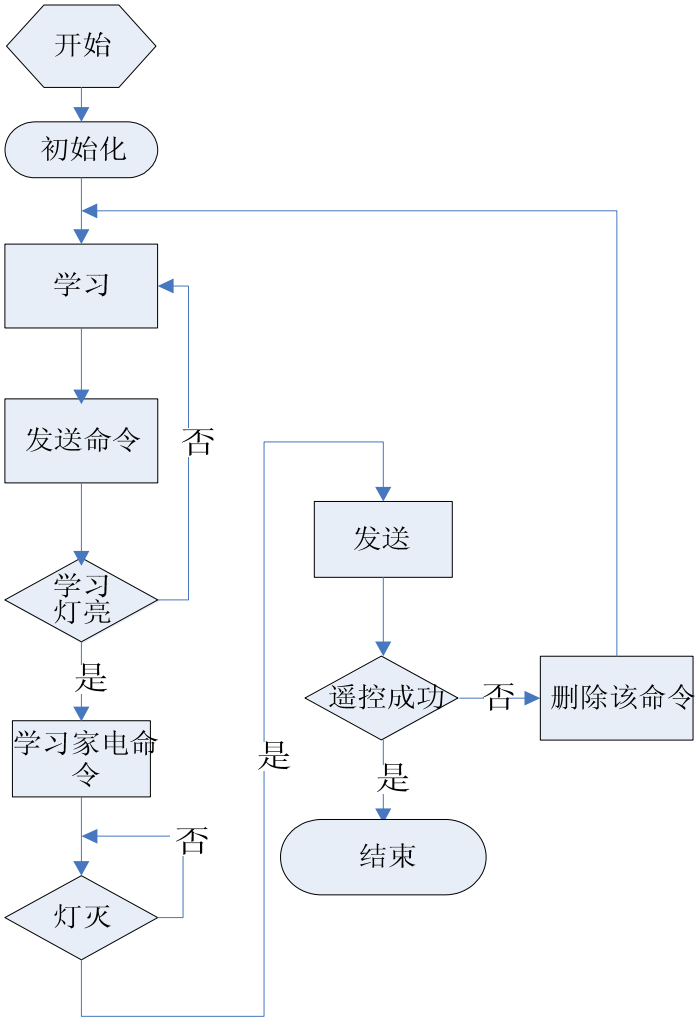

图 4-22 家电命令学习流程图

- 主机设置与系统说明

主机设置与系统说明的设计比较简单，他们用到的布局文件为 systemdata.xml 、 systemillustrate.xml ， 显示界面为 systemdata.java 、 systemillustrate.java。

7．其他模块设计方案

(1)这里给出可视门禁和访客记录的业务处理流程分别如图 4-23 和图 4-24 所示。

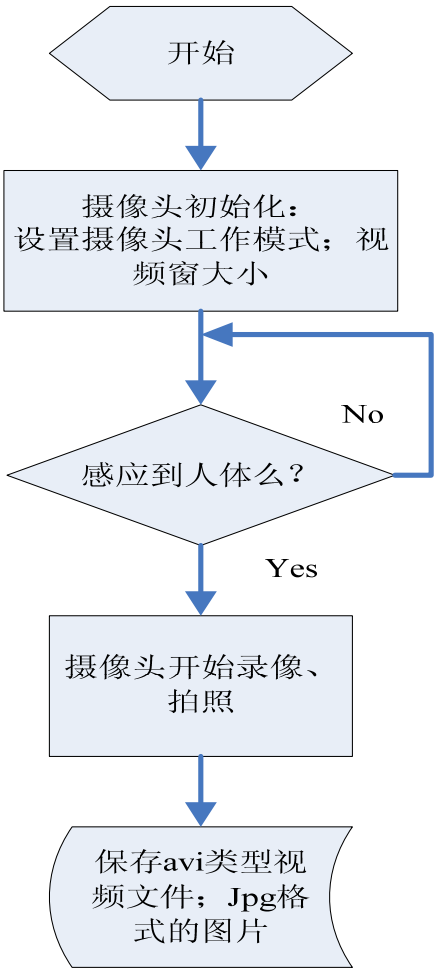

图 4-23 可视门禁流程图

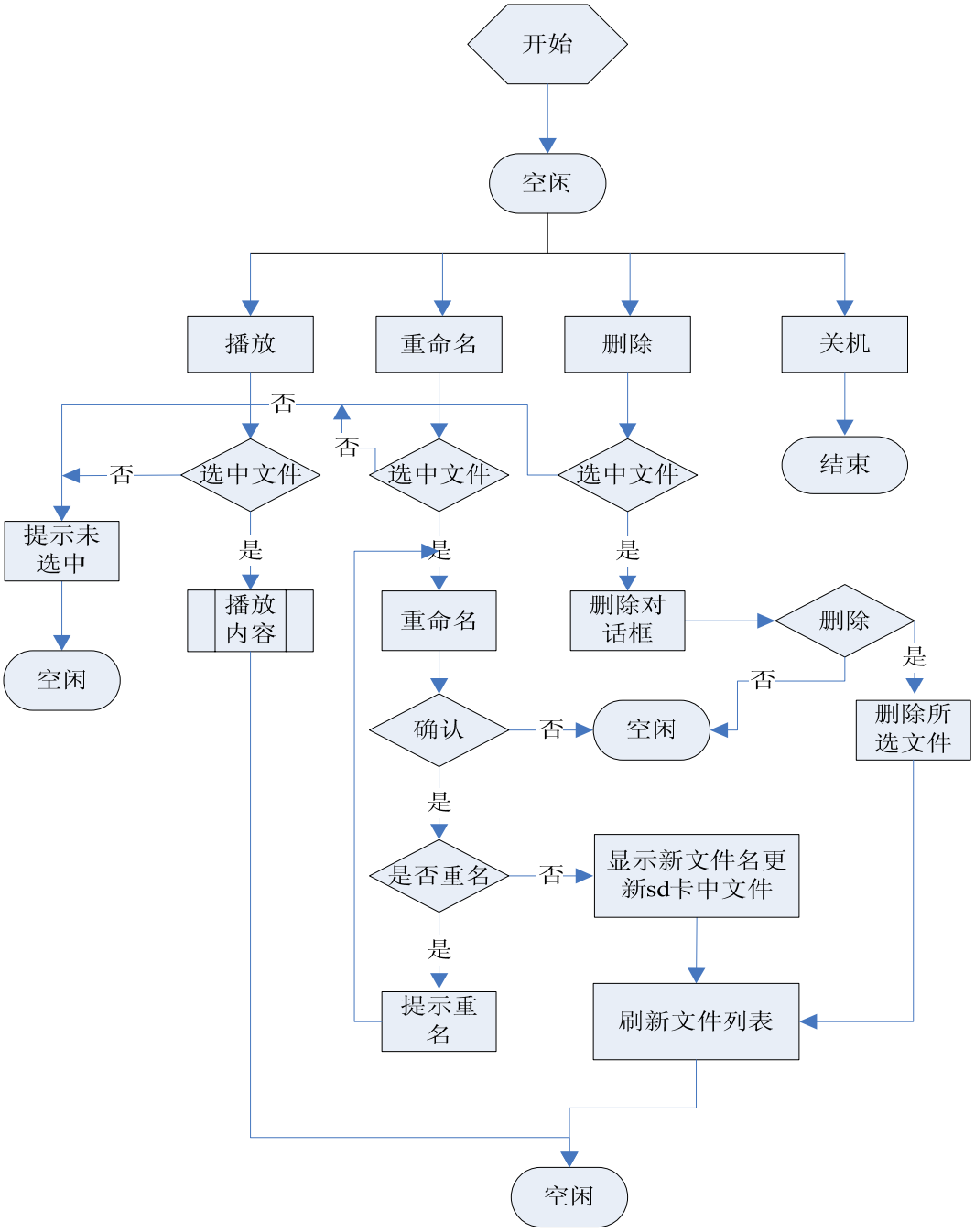

图 4-24 访客记录流程图

(2)电子秘书和语音留言，他们的处理过程基本相同，这里给出语音留言的处理流程如图 4-25 所示。

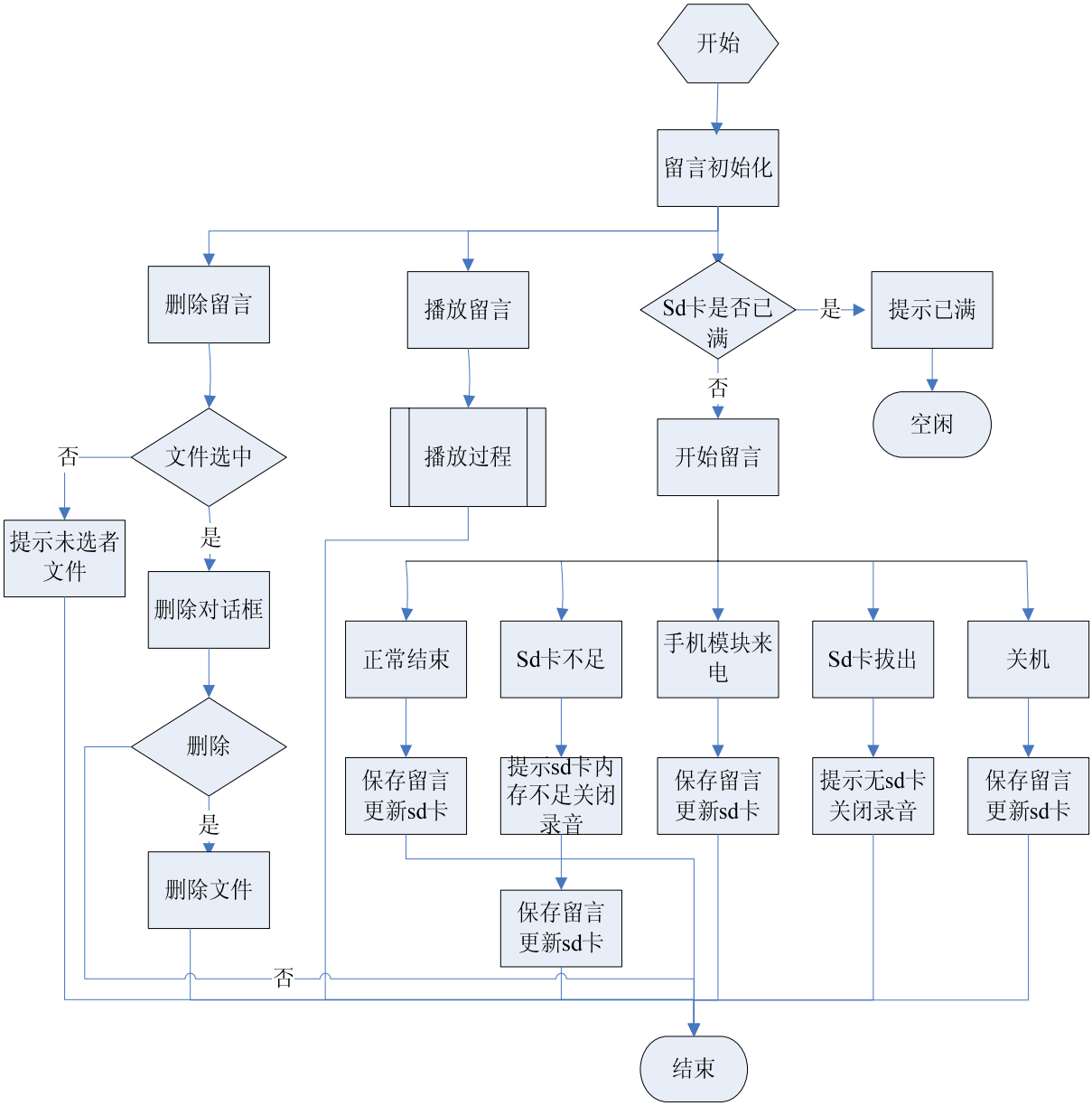

图 4-25 语音留言处理流程

4.4 本章小结

本章首先对系统的软件进行简要概述，系统的软件设计分为两部分：一部分是节点软件设计，另一部分是智能家居软件设计。而节点软件又细分为环境监测节点和执行节点的软件设计，这里主要涉及的是在 Z-stack 栈上开发应用程序；详细分析了智能家居系统中各模块的具体功能，在此基础上对各模块进行相应的设计。

第 5 章 智能家居系统软件实现

5.1 主系统软件开发环境搭建

为了实现系统所设定的功能，不仅需要编写 Android 应用程序，而且需要根据智能家居系统的功能描述重新编译系统的内核。这两者的开发需要不同的开发工具。

Java 是 Android 应用层软件的开发语言，而 Eclipse 是一种基于 Java 的可扩展开源开发平台。只要为 Eclipse 安装了 Android 插件，开发者就可以使用 Eclipse 开发 Android 应用程序。而 Eclipse 的安装和使用都很方便，所以本文选择 Eclipse 作为 Android 应用程序的开发工具。修改及重新编译内核，需要对内核源代码操作，这在 windows 下是无法完成的，除非安装了 Cygwin[33,34]。这里我们采用 google 所推荐的软件环境 Ubuntu。这里需要指出的是我们完全可以在 Ubuntu 下完成该软件系统从底层到应用层的开发，在 Windows 下开发 Android 应用程序只是习惯问题。下面就详细介绍 Linux 下开发环境的搭建。

Linux 下搭建 Android 开发环境的一般步骤为：

①安装 Ubuntu 及修改 root 权限

②安装 Java 运行环境 jdk

③安装 Android sdk

④安装 Eclipse 及其 Android 插件 ADT

⑤配置 Android avd

1．安装 Ubuntu 操作系统及修改 root 权限

在 VMware 中安装 Ubuntu。虚拟机的好处是它可以安装 Windows 平台下模拟真实的电脑环境，不需要重装 Windows 系统，安全高效实用。目前市场上最著名的虚拟机就是 VMware Workstation 了。网上可下载到 VM 最新版，这里我们用到的版本是 7.1.3 build-324285。具体安装过程可参考相关书籍。

安装完成后我们发现，Ubuntu 默认禁止 root 用户登录，但是，后面我们所做的工作大都需要 root 权限，为了开发方便我们需要修改一下设置，以后就可

以直接在 root 账号下操作了。

在终端窗口中输入：sudo vim /etc/gdm/gdm.conf

打开 gnome 的配置文件，找到 AllowRoot＝false，把 false 改为 true，保存后退出。重启系统后，就可以用 root 登录了。

2．安装 Java 运行环境 jdk

到 sun 官方主页下载 jdk6 离线安装包 jdk-1_6_26-linux-i586.bin 来进行安装：

chmod 777 jdk-1_6_26-linux-i586.bin

./ jdk-1_6_26-linux-i586.bin

执行后，便能在当前目录下得到 jdk1.6.26 目录。

安装结束后还要设置好环境变量让 Android 找到这个路径，编辑/etc/profile, 增加下面的语句：

export JAVA_HOME=/ jdk1.6.26

export ANDROID_JAVA_HOME=$JAVA_HOME

export PATH= $JAVA_HOME/bin:$PATH

这里我们要根据实际情况修改路径，保存退出。

3．安装 Android sdk

Android SDK 是 Android 专属的软件开发工具包，有了它我们才能开发 Android 程序。

最新版本的 Sdk 可从官网 http://developer.Android.com/sdk/index.html 上下载。本系统所用的版本是 Android-sdk_r13-linux_x86.tar。

在终端中输入以下命令，即可解压到当前目录。

tar –xvf Android-sdk_r13-linux_x86.tar

配置 sdk 的环境变量

从 Android 2.3 开始，sdk 的目录结构发生了变化，多了一个 plthforms-tools 文件夹，adb 文件有 tools 文件夹移动到了这个目录，因为不能只配置 tools 文件夹的环境变量，还需配置 platforms- tools 文件夹的环境变量，配置方式和配置 Java 的环境变量一样，顺序在 Path 后依次添加两个文件夹的路径即可，配置完成后在命令行中输入 adb 可能不管用，提示没有这个命令，这时可以重启系统，然后

就可以了。

4．安装 Eclipse 及其 Android 插件 ADT

在 Eclipse 官网上下载软件包 eclipse-cpp-helios-SR2-linux-gtk.tar.gz，在 Ubuntu 根目录下解压：

tar –zxvf eclipse-cpp-helios-SR2-linux-gtk.tar.gz

进入解压后的文件夹 eclipse

cd eclipse/

然后在终端输入

eclipse

这样 Eclipse 就启动起来了。这时就需要安装 ADT 插件了。运行 Eclipse，设置工作区，然后在 Help 菜单——Install New Software，点击右边的 Add，在 Location 里输入 **http://dl-ssl.google.com/Android/eclipse** （http 后面有无 s 都可以），Name 随便输个 ADT，点击 OK。然后列表内会出现条目，勾选全部，然后 Next，等待。然后会出现一个新的窗口，让你 Accept 协议，点击 Finish 继续。然后就是真正的安装 ADT 插件过程了，中途可能会跳出“Security Warning”窗口，点击 OK，再等两分钟会提示你重新启动 Eclipse。

5．配置 Eclipse

重新启动 Eclipse 后，在菜单 Window——Preference，然后点击左边的 Android，在右边的 SDK Location 输入你的 SDK 所在目录，我这里输入 /Android-SDK ，点击 Apply，列表出现 Android2.1，点击 OK 完成设置。

若要每次都在终端启动软件需要配置各软件的环境变量，将自己安装的软件路径添加到相应的配置文件中（不同环境变量之间的分隔符是冒号）。在 /etc/profile 中配置的环境变量：export PATH= $JAVA_HOME/bin:$PATH:/Android-sdk-linux_x86/tools:/eclipse。

至此，Android 在 Ubuntu 下的开发环境已经搭建成功。

5.2 Android 应用程序四大组件

Android 应用程序通常由一个或多个基本组件组成，前面我们介绍智能家居系统中所提到的 Activity 就是 Android 应用程序最常用的组件之一，其他组件分别是 Service、Broadcastreceiver、Contentprovider 组件[35,36,37]。

Activity：Activity 为 Android 应用提供了可视化的用户界面，一个 Activity 代表了一个单独的屏幕[38]。Activity 作用是应用程序与用户交互的接口；控制整个屏幕的的显示和布局。从视觉效果来看，一个 Activity 占据当前的窗口，响应所有窗口事件，具备有控件，菜单等界面元素。从内部逻辑来看，Activity 需要为了保持各个界面状态，需要做很多持久化的事情，还需要妥善管理生命周期，和一些转跳逻辑。

(1)Actvity 堆栈：每个 Actvity 的状态由它所在 Activity 栈中的位置所决定，所有当前正在运行的 Actvity 将遵循照后进先出的原则。Android 应用程序一般会包括多个 activity，这些 activity 组成 activity 栈，当前活动的 activity 位于栈顶。

(2)Activity 的生命周期：完整的 Activity 生命周期之间从 OnCreate 方法开始，到 onDestroy 方法结束。有可能在某些情况下，一个 Activity 被终止时并不调用 onDestroy 方法。使用 OnCreate 方法来初始化你 Activity，初始化的用户界面，分配引用类变量，绑定数据控件，并创建服务和线程。在 OnCreate 方法传递的对象 Bundle 包含最后一次调用 onSaveInstanceState 保存的 UI 状态。你可以使用这个 Bundle 恢复用户界面到以前的状态，无论是在 OnCreate 方法或通过覆盖 onRestoreInstanceStateMethod 方法。

(3)Activity 的启动方式：一种是调用 startActivity 方法，这个 intent 或者指明了一个特定的 Activity 或者是里面包含选定的 Activity 的信息。但是具体的启动哪个 Activity 是由系统决定的 Android 系统负责挑选一个最满意匹配挑选条件的 Activity。使用这个方法，当新的 Activity 运行结束的时候，不必执行原来的 Activity 里的回调函数，也不必返回值；一种是调用 startActivityForResult 方法，当新的 Activity 运行结束的时候，必须执行原来的 Activity 里的回调函数。而当退出 Activity 时，常用的方法是 finish 方法。

Service：服务没有可视化的用户界面，而是在一定的时间内在后台运行。后台服务于 Activity，封装有一个完整的功能逻辑实现，接受上层指令，完成相关的事务，定义好需要接受的 Intent 提供同步和异步的接口, Service 的启动有两种方式：context.startService() 和 context.bindService()。

BroadCastReceiver：广播接收器是一个专注于接收广播通知信息，并做出对应处理的组件。是 Android 提供的第三方应用数据的访问方案，可以派生 Content Provider 类，对外提供数据，可以像数据库一样进行选择排序，屏蔽内部数据的存储细节，向外提供统一的接口模型，大大简化上层应用，对数据的整合提供了更方便的途径。广播接收器没有用户界面。然而，它们可以启动一个 Activity 来响应它们收到的信息，或者用 NotificationManager 来通知用户。通知可以用很多种方式来吸引用户的注意力——闪动背灯、震动、播放声音等等。一般来说是在状态栏上放一个持久的图标，用户可以打开它并获取消息。

Content Provider：内容提供者将一些特定的应用程序数据供给其它应用程序使用。数据可以存储于文件系统、SQLite 数据库或其它方式。内容提供者继承于 ContentProvider 基类，为其它应用程序取用和存储它管理的数据实现了一套标准方法。然而，应用程序并不直接调用这些方法，而是使用一个 ContentResolver 对象，调用它的方法作为替代。ContentResolver 可以与任意内容提供者进行会话，与其合作来对所有相关交互通讯进行管理。

Intent 和 IntentFilter： Intent 在 Android 中是一个十分重要的组件，它是连接不同应用的桥梁和纽带。当 Intent 在组件间传递时，组件如果想告知 Android 系统自己能够响应和处理哪些 Intent，那么就需要用到 IntentFilter 对象。Intent 和 IntentFilter 就是 Android 的一种消息通信机制，可以让系统的组件之间进行通信。信息的载体就是 Intent，它可以是一个要完成的动作请求，也可以一般性的消息广播，它可以由任意一个组件发出。消息，也就是 Intent，最终也是要被组件来进行处理和消化。消息由组件发出，通常在消息的里面也会有会标记有目标组件的相关信息，另外目标组件也需要告诉系统，哪些消息是它所感兴趣的，它需要设置一些过滤器，以过滤掉无关的消息。

5.3 Android JNI 机制及其实现

5.3.1 JNI 机制简介

JNI 是 Java Native Interface 的缩写，中文为 Java 本地调用。JNI 标准是 Java 平台的一部分，它允许 Java 代码和其他语言写的代码进行交互，它使得在 Java 虚拟机 (VM) 内部运行的 Java 代码能够与用其它编程语言(如 C、C++ 和汇编语言)编写的应用程序和库进行交互操作[39]。

1．JNI 产生的原因[40]。

（1）应用程序需要使用系统相关的功能，而 Java 代码不支持或难以办到。

（2）已有其他语一言写好的类库或程序，希望 Java 程序可以使用它们。

（3）出于更高的性能要求，希望使用汇编或是 C/C++语一言来实现部分功能。

基于上述几点原因，Android 平台也支持 JNI 机制，而且本身 Android 的组件库就是用 C/C++编写的，并且一些底层与本地硬件交互的功能就是通过 JNI 实现的。所以这种方式并不违背 Android 的整个架构，在理论上我们也可以进行这种方式的开发，在实际应用中如果上层要调用底层的驱动时，若上层软件没有提供接口时，这就需要我们自己调用底层驱动，这时就会用到 JNI 机制。图 5-1 展示了 JNI 机制的作用，经过 JNI 机制，上层 Java 就可和其他语言进行交互了。

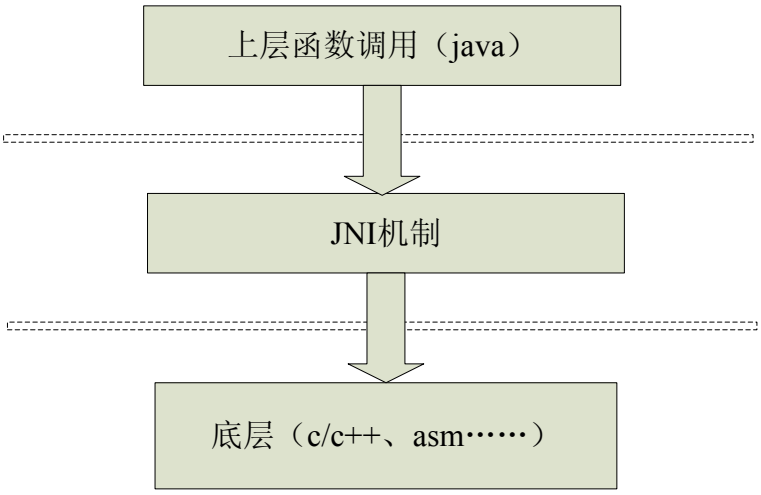

图 5-1 Android JNI 机制作用

2．Android JNI 程序开发一般步骤[41]。

(1) 编写带有 native 声明的方法的 Java 类。

(2) 使用 javac 或 IDE(JBuilder,eclipse 等)编译所编写的 java 类。

(3) 使用 javah -jni java 类名生成扩展名为 h 的头文件。

(4) 使用 C++ 实现本地方法，实现后使用 NDK 工具进行编译。

(5) 将编译好的库加载到项目中，在 Java 中 load 动态链接库文件，调用 native 方法。

5.3.2 Android JNI 的实现——NDK

Android NDK[42]提供了一系列的工具，帮助开发者快速开发 C（或 C++）的动态库，并能自动将 so 和 Java 应用一起打包成 APK，这些工具对开发者的帮助是巨大的。 NDK 集成了交叉编译器，并提供了相应的 mk 文件隔离 CPU、平台、ABI 等差异，开发人员只需要简单修改 mk 文件（指出“哪些文件需要编译”、“编译特性要求”等），就可以创建出 so。

Android NDK 作用主要用来编译本地库(由 C,C++文件编译后运行于目标 CPU 的库)给 Android java 调用,同时,可以用来打包编译动态库和静态库,并且能将本地库和 Java 文件打包成 APK。

以下是 NDK 工具的一般用法：

(1)项目中 C/C++源代码应该放在<ndk>/sources/<my src>/.…下。也可以将源代码链接到其他目录。这些源代码并不是完全与某一个共享库或 Android 应用绑定在一起，相反，它们可通过不同的配置文件生成对应不同 Android 应用的共享库。

(2)创建<ndk>/sources/<my src>/Android.mk 来指定要编译的源代码。

(3)创建<ndk>/apps/<my app>/Application.mk 来指定对应的目标 Android 应用。这个文件将一个 Android SDK 应用工程与<ndk>/sources/中的共享库关联起来，并指定了接收共享库的应用工程所在目录。

(4)从 NDK 根目录运行 make 指令进行源代码编译:$make APP=<my app>编译工具将拷贝共享库到应用工程的相应目录。

(5)最后，用 SDK 工具编译 Android 应用。SDK 编译工具将把之前编译好的共享库打到 apk 包中。

5.4 系统文件结构

智能家居系统是在 Android 平台上开发的，因此所用的主要语言是 Java，应

用程序是在 Eclipse 下开发的，整个工程文件名为 smarthome。整个工程项目目录结构如图 5-2 所示：

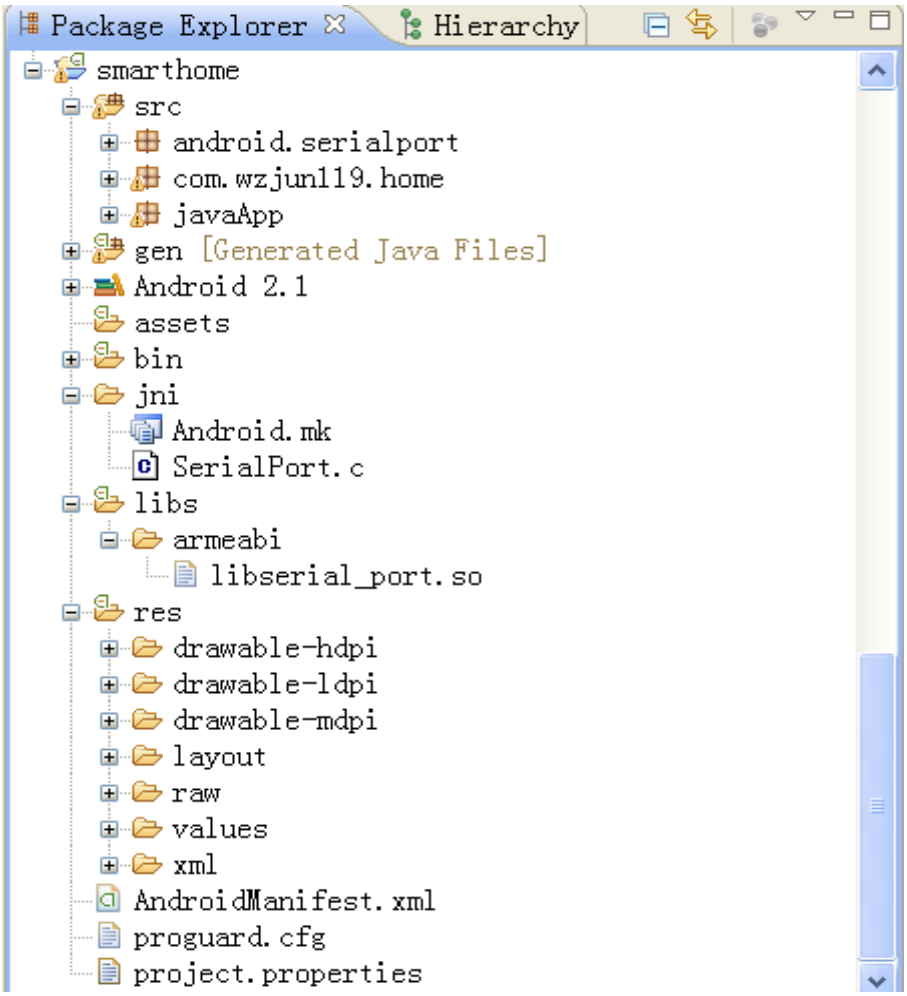

图 5-2 智能家居系统工程目录结构

下面对整个工程的目录结构及其各个文件作用进行简要分析：

Src 和 gen 文件夹下是工程项目的源文件，所有的 Java 代码都在这两个文件夹下。其中 src 文件夹存放的是本项目所开发的 Java 文件。gen 下面的 R.java 文件是 Eclipse 根据应用中的资源文件自动生成的，所以 R.java 就相当于 Android 应用程序的资源字典[43]。

Res 是存放整个系统中所用到的所有资源， Drawble 文件夹存放的是本文开发过程中所用到的所有图标、图片。Drawble 包括 drawable-hdpi, drawable-ldpi, drawable-mdpi 三个文件夹是 Android2.1 版本的改进，是为了适应 Android 系统不同的屏幕大小而设计的具有很好的兼容性；layout 文件夹下存放的是系统中各个界面的布局文件；raw 与 xml 文件夹下都存放的是资源文件，他们的区别是 xml 下的资源文件可以被 AAPT 打包进应用程序包，如果需要被打包，放置在/res/xml

目录下，如果不需要被打包，则放置在/res/raw 下；values 文件夹下存放的是 xml 格式的资源文件（比如字符串资源文件：strings.xml，颜色资源文件：colors.xml，尺寸资源文件：dimens.xml）。

Libs 文件夹下存放的是通过 NDK 工具编译出的动态库，供系统调用。

AndroidManifest.xml 是 Android 项目的系统清单文件，每个 Android 程序中所必需的，它位于 Android 工程的根目录中，描述了应用程序中的所有组件，包括 Package 中的组件(Activities,Services 等)以及他们各自的实现类，以及各自的使用权限。Manifest 文件的主要功能是声明应用程序的组件。例如:一个 Activity 必须要有一个<Activity>标记对应，无论它是供外部使用还是本工程使用。如果一个 Activity 没有在 AndroidManifest.xml 中注册，则系统无法调用。

5.5 串口收发程序设计与实现

S3C6410 与主节点 CC2530 之间是通过串口进行通信的，主机 S3C6410 通过串口向主节点发送相应的命令来控制家庭网络中的各个终端子节点。主节点将各终端节点的传来的数据通过串口传递给 S3C6410。

Android 是在 Linux 内核上建立一个硬件抽象层(Android HAL)，通过 Dalvik 以及各种库来执行 Android 应用的。在开发板上电启动时，首先需要由 Bootloader(uboot)引导 Linux 及开发板上各个硬件设备的驱动程序，之后才启动 Android 系统。在 Android 系统中，由于 Linux 内核的保护机制，应用程序是无法直接对串口设备进行操作，必须要通过设备的驱动程序来实现对设备的控制。所谓驱动程序就是一组由系统内核的相关子例程和数据组成的 IO 设备软件接口。每当内核意识到要对某个设备进行特定的操作时，它就调用相应的驱动接口函数。当用户调用驱动时，系统就从用户进程转移到了驱动空间，当驱动调用完成后又返回到了用户进程[44]。

这里涉及的类有 serial_port.java、SerialPortFinder.java、myApplication.java 和 serialportservice.java,下面对这几个类分别介绍。

(1)串口通信——动态链接库的生成

本实验用的开发环境是 Ubuntu10.10，使用 Android-ndk-r7 编译动态库[45]。其大致过程如下：

将 Android-ndk-r7-linux-x86.tar.bz2 解压到当前的工作目录下（本实验用的是/home/wzjun119 ），在 /etc/profile 中设置环境变量，export /home/wzjun119/Android-ndk-r7。重启系统后就可以使用 ndk-build 命令了。

在 eclipse 里创建一个新工程 smarthome，在工程文件夹下建立 jni 目录，再在该目录下创建 SerialPort.c 文件，在 src 目录下编写 serial_port 类，在该类中对调用 native 的方法进行声明。其关键代码如下所示：

```
// JNI
private native static FileDescriptor open(String path, int baudrate);
public native void close();
static {
    System.loadLibrary("serial_port");
        }
```

在终端窗口中使用 javac serial_port.java 命令编译后，使用 javah 指令编译 serial_port.class 文件，生成 Java 与 C/C++之间进行通信的约定接口，它规定了 Java 中 nativemethod 在 C/C++的具体接口。然后在 jni 下的 SerialPort 文件中实现该接口。编写相应的 Android.mk 文件。进入到工程主目录中，运行 ndk-build 命令后，在 lib/eambi 目录下会自动生成 libserial_port.so 动态库。

(2)串口通信——串口助手类

该类主要用于获取设备下面所有串口设备和路径。关键代码如下所示：

```
try {
        itdriv = getDrivers().iterator();
        while(itdriv.hasNext()) {
            Driver driver = itdriv.next();
            Iterator<File> itdev = driver.getDevices().iterator();
            while(itdev.hasNext()) {
                String device = itdev.next().getName();
                String value = String.format("%s (%s)", device, driver.getName());
                devices.add(value);
                }
}
```

上面是 getAllDevices()函数中的主要代码，现在对部分代码进行分析：getDrivers().iterator()首先给 getDrivers 加上一个迭代器，遍历所有的驱动。即把每遍历到一个符合条件的驱动就通过 mDrivers.add(new Driver(w[0], w[1]))函数添加到 driver 类中。然后调用 getDivices().iterator()，到/dev 下面去搜索前缀相同的设备名。在我们所要实现的功能中，就会在/dev 下搜索有没有 tty0，tty1，tty2，

等等。 这样，就能在初始化的地方映射到/dev/ttyS0，dev/s3c2410_serial0、dev/s3c2410_serial 1、 dev/s3c2410_serial 2 等等。

(3)串口通信——application 类

新建一个 myApplication 类继承 Application，用来对串口进行初始化和关闭串口，另外该类将搜索到的设备地址和波特率保存到 SharePreferences 中。这样设计就可以在不同的开发板上打开不同的串口。这样设计后只需选择相应的串口号和波特率即可在不同的开发板中使用该程序。

(4)串口通信——serialportservice 类

在 serialportservice 类的 onCreat()方法中首先调用 mApplication.getSerialPort()打开串口，获取该串口的输入输出流。

```
public void run() {
        super.run();
        while(!isInterrupted()) {
            int size;
            try {
                //byte[] buffer = new byte[15];/*64*/
                byte[] buffer = new byte[15];
                if (mInputStream == null) return;
                size = mInputStream.read(buffer);
                if (size > 0) {
                    onDataReceived(buffer, size);
                }
    ……
            }
```

在该段代码中可以看到一个 onDataReceived(buffer, size)方法，该方法负责对接收到的数据进行处理，具体的实现过程在 dataservcie 类（它继承了 serialportservice 类）实现。

5.6 智能家居系统实现

1．智能家居系统主界面

在第二章第一节我们对系统所要的实现的功能进行了整体的概述，在此基础上对智能家居主界面进行了设计，并做了美化处理[46,47]。实现后的界面如下图 5-3 所示：

图 5-3 智能家居系统主界面

实现该界面的类为 SmarthomeActivity.java，前面我们提到一个 Android 应用程序通常都会包含多个 Activity，但只有一个 Activity 会作为程序的入口，该 SmarthomeActivity 就是整个系统的入口，其他 Activity 都由该入口 Activity 启动。该 Activity 在 AndroidManifest.xml 文件中的配置如下：

```
<activity Android:name=".SmarthomeActivity"
          Android:label="@string/app_name">
  <intent-filter>
    <action Android:name="Android.intent.action.MAIN" />
    <category  Android:name="Android.intent.category.LAUNCHER" />
  </intent-filter>
</activity>
```

为了实现上图所示的界面，我们需要在layout文件夹下设置该界面的布局文件main.xml。在程序中通过setContentView方法来加载，再通过findViewById方法来获得每个组件的引用。在此布局中我们使用的是在线性布局中嵌套了表格布局，具体实现如下所示：

```
<LinearLayout xmlns:Android="http://schemas.Android.com/apk/res/Android"
```

```
    Android:orientation="vertical"
    Android:layout_width="fill_parent"
    Android:layout_height="fill_parent"
    Android:gravity="center|bottom"
    Android:background="@drawable/mainpic"
    >

    Android:orientation="vertical"
    Android:layout_width="fill_parent"
    Android:layout_height="wrap_content"
    >
   <TableRow
   Android:gravity="center"
   >
   <ImageButton
   Android:id="@+id/homeset"
   Android:layout_width="wrap_content"
   Android:layout_height="wrap_content"
   Android:src="@drawable/homeset"
   Android:background="#00ffffff"
   Android:text="家庭安防"
   />
   ……
  </TableLayout>
</LinearLayout>
```

在该界面中用每一个功能都是一个图形按钮组件，若用户单击其中任一按钮都会跳转到相应的Activity，这里以家居安防进行说明，当用户点击了该界面的“智能家居”按钮后，就会从SmarthomeActivity跳转到contextualdisplay所示的界面中，其主要实现代码为：

```
homesetbuttom.setOnClickListener(new OnClickListener() {
        @Override
        public void onClick(View v) {
            // TODO Auto-generated method stub
            Intent intent = new Intent();
            intent.setClass(SmarthomeActivity.this, contextualdisplay.class);
            intent.setFlags(Intent.FLAG_ACTIVITY_SINGLE_TOP);
            startActivity(intent);
        }
    });
```

2．家居安防模块实现

在家居安防设计这一小节中已经提到该界面的实现类为contextualdisplay，这

里涉及到的类有dataserver.java、athomedisplay.java、leavedisplay.java和nightdisplay.java等，他们之间的关系如下图5-4所示：

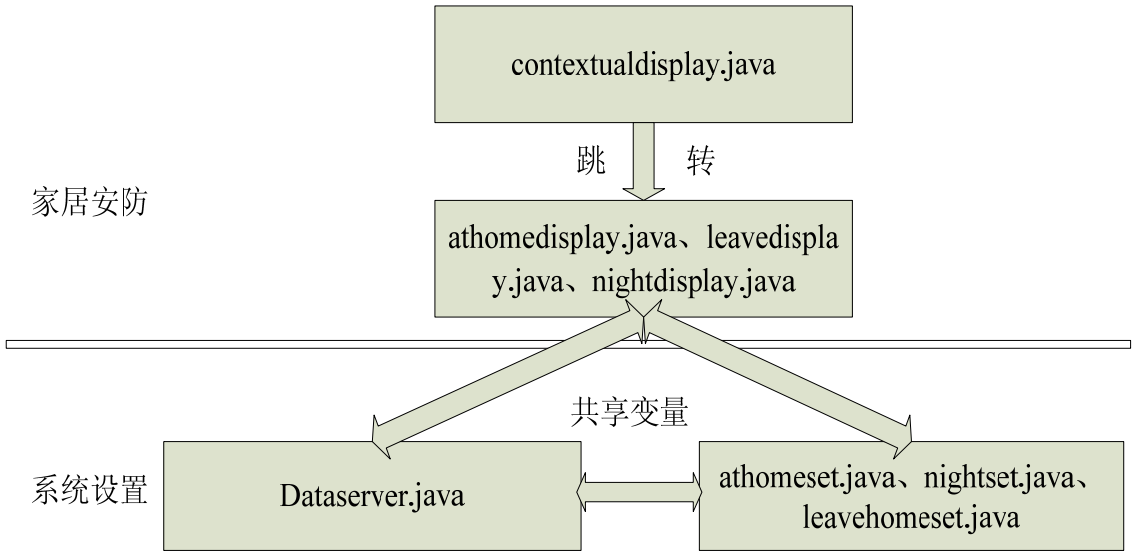

图5-4 家居安防与系统设置中类的调用关系图

这个模块与系统设置模块中的情景模式设置模块紧密相连，在athomeset.java、nightset.java和leavehomeset.java文件中保存的是用户分别在居家模式、夜间模式和离家模式下所设定的各种组合，比如在athomeset.java文件中系统会默认为用户选择相关的功能，这里有系统默认的配置，基于表5.1中所示的原则：

表5.1 情景模式与节点模块组合原则

设备	居家模式	离家模式	夜间模式	连动条件
空调	可控	关	可控	温湿度传感器
电视	可控	关	关	红外遥控
电饭煲	可控	可控	关	——
灯	可控	可控	可控	光照传感器、人体感应
温度传感器	开	关	开	关联湿度传感器
湿度传感器	开	关	开	关联温度传感器
光照传感器	开	关	关	——
雨滴传感器	开	开	开	——
玻璃破碎传感器	关	开	开	——
气体传感器	开	开	开	——
烟雾传感器	开	开	开	——
人体感应	开	关	开	——
红外遥控	开	关	关	——

. 这里对表格中的连动关系进行说明，根据居家模式的原则关系可知若用户在athomeset.java中只选择了温湿度传感器或者只选择了空调，那么系统将无法进行连动操作，其他传感器与其他电器组合与此类似，只有同时选择才能发挥智能家居系统的安防连动作用。下图5-5为居家模式工作流程。

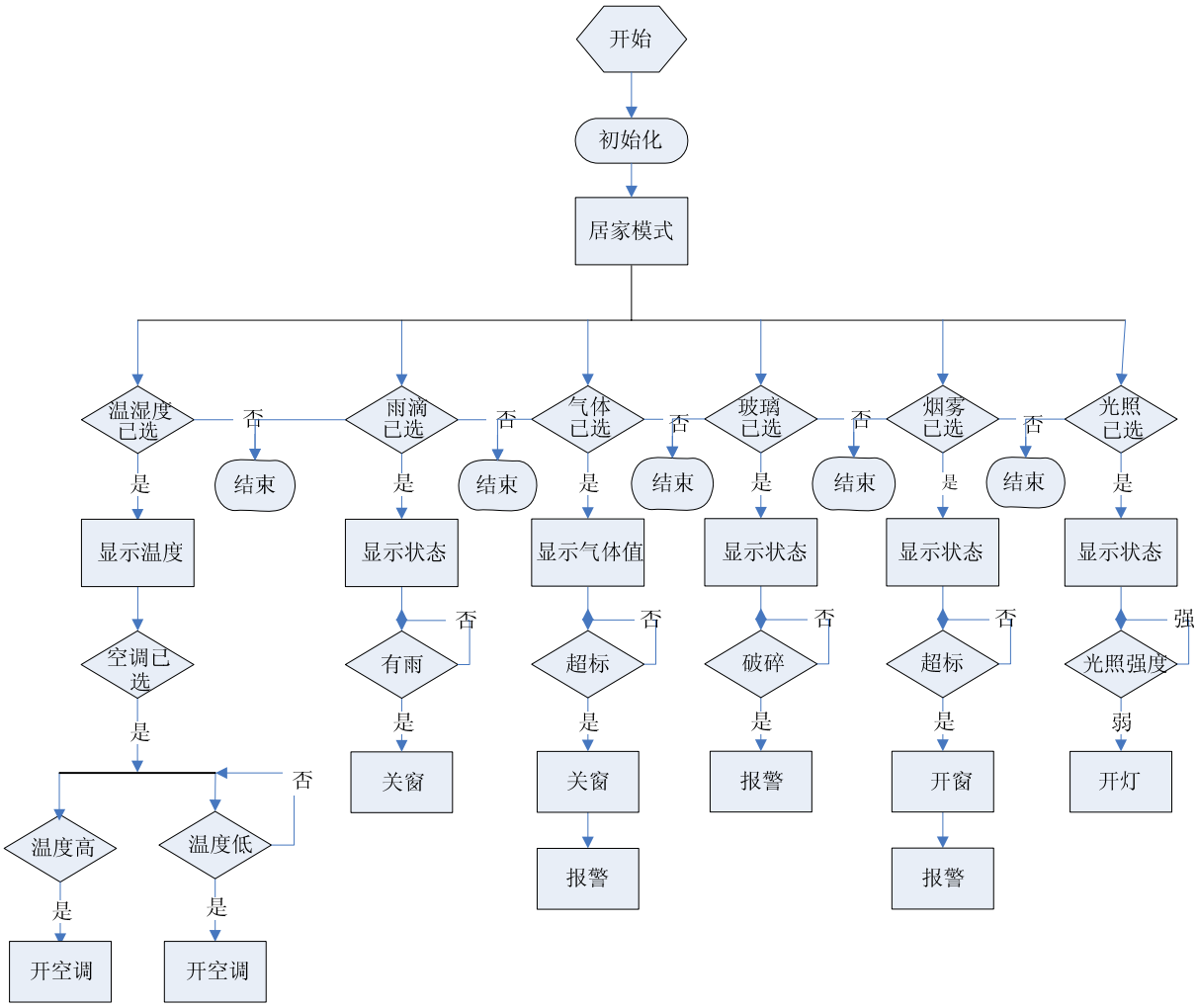

图5-5 居家模式工作流程图

在家居安防模块中contextualdisplay.java的实现比较简单只涉及到此界面的布局文件contextualdisplay.xml，在此activity中处理过程由一个switch选择语句和一个能弹出提示对话框的方法来实现的。在switch语句中主要处理用户选择的3中情景模式，当用户单击居家模式后，会跳转到athomedisplay界面中，该Activity主要负责各传感器的工作状态的显示，为了能实时地显示从传感器传来的状态在athomedisplay类中我们使用了定时器，将用户显示界面定时刷新，动态显示各种状态，具体关键代码如下所示：

```
mHandler = new Handler(){
    @Override
     public void handleMessage(Message msg) {
       switch (msg.what){
             case UPDATE_TEXT:
              /*数据显示*/
              ..................
};
```

```
            mTimerTask = new TimerTask() {
                @Override
                public void run() {
                    mHandler.sendEmptyMessage(UPDATE_TEXT);
                }
            };
            mTimer.schedule(mTimerTask, 1000, 1000);
        }
```

而具体的数据处理过程主要在dataserver.java的服务中实现。其居家模式的关键代码如下：

```
    //居家模式
    if(myApplication.modelselect==1){
        if(myApplication.pamerment1[4])   //温度
            temhandle(myApplication.tem1);
        if(myApplication.pamerment1[5])   //光照
            lighthandle(myApplication.illumination);
        ……
    }
```

以温度为例进行说明，在居家模式中调用了temhandle方法对温度值进行处理，该方法有一个参数，就是温度值。在该函数中主要是对该温度值进行判断，先判断是否连接了空调,然后再判断温度是否超标，若超标则开启空调，反之亦然。下面的代码是此方法具体的实现：

```
    double temdata1=Double.parseDouble(temdata);
    if(ischeckok()){
        if(myApplication.pamerment1[0]){
            if(temdata1>30.0){
                sendcommand();
                sendconditionopen();
                sendzhilengmoshi();
                }
            else if(temdata1<15.0){
                sendcommand();
                sendconditionopen();
                sendjiaremoshi();
                }
            else {
                ……
                    }
    }
```

3．家电控制模块实现

选项卡 TabHost 是整个 Tab 的容器，包括两部分，TabWidget 和 FrameLayout。TabWidget 就是每个 tab 的标签，FrameLayout 则是 tab 内容。

(1)当我们的实现类继承了 TabActitivty 时，如同 ListActivity，TabHost 必须设置为@android:id/tabhost；

(2)TabWidget 必须设置 android:id 为@android:id/tabs；

(3)FrameLayout 需要设置 android:id 为@android:id/tabcontent。

该家电控制模块具体实现中就使用了TabHost。该模块的实现类为electrlmain.java，布局文件为electrlmain.xml。该类继承了TabActivity，然后通过调用TabActivity的getTabHost方法获取TabHost对象，再通过该对象的方法来创建选项卡、添加选项卡。部分关键代码如下所示：

```
TabHost tabHost = getTabHost();
    //设置使用TabHost布局
    LayoutInflater.from(this).inflate(R.layout.electrlmain,
            tabHost.getTabContentView(), true);
    //添加第一个标签页
    tabHost.addTab(tabHost.newTabSpec("tab1")
        .setIndicator("空调"
            , getResources().getDrawable(R.drawable.condition))
            .setContent(R.id.tab01));
```

然后为标签切换事件处理添加监听器：setOnTabChangedListener。在此方法中处理用户所传递的数据，与系统设置中的家电学习命令相关联，从而实现对家电的控制。

```
tabhost.setOnTabChangedListener(new OnTabChangeListener(){
        @Override
        public void onTabChanged(String tabId) {
            // TODO Auto-generated method stub
                ……
        }
    });
```

4．电话功能模块实现

实现该界面的Activity为telecallmain，其布局文件为telecall.xml，该界面的布局设计是在线性布局中嵌套线性布局实现的。Telecallmain的主要代码为：

```
    public void onCreate(Bundle savedInstanceState) {
        ……
        bDial.setOnClickListener(//为拨号按钮添加监听器
       //OnClickListener为View的内部接口，其实现者负责监听鼠标点击事件
```

```
                new View.OnClickListener(){
                    public void onClick(View v){
                        String num=et.getText().toString();
                        //根据获取的电话号码创建Intent拨号
                        Intent dial = new Intent();
                        dial.setAction("Android.intent.action.CALL");
                        dial.setData(Uri.parse("tel://"+num));
                        startActivity(dial);                                    }
                });
            numListener=new    View.OnClickListener(){
                public void onClick(View v){ //为0-9数字按钮创建监听器
                                et.append(tempb.getText());
                }
            };
            for(int id:numButtonIds){//为所有数字按钮添加监听器
                            Button tempb=(Button)this.findViewById(id);
                        tempb.setOnClickListener(numListener);
            }
            ……
        }
```

5．状态检测模块实现

在4.3.2这一小节中我们提到主要是通过发送一条广播命令实现该模块的功能，下面介绍该命令格式见表5.2所示：

表5.2 广播查询命令格式

格式说明	标志	长度	父节点地址	原地地址	类型	数据	校验和
占用byte数	1	1	2	2	1	4	1

上表是具体的命名格式，从表中我们可以清楚看出该命令各字节的含义，该命令具体实现为：

{0X00,0X02,0XFF,0XFF,0X00,0X00,‘Z’,0X00,0X00,0X00,0X00,0X5A}

这里的目的地址为0XFFFF，这个地址代表的是广播地址，而原地地址0X0000代表的是协调器的地址，因此当该命令被广播后会通知相应的节点对相关字节的判断，返回预设的信息。

发送该命令的方法为：

```
private void sendbroadcastcommand() {
// TODO Auto-generated method stub
```

```
//发送广播地址
        bstr[0]=(byte)0x00;
        bstr[1]=(byte)0x02;
        bstr[2]=(byte)0xFF;
        bstr[3]=(byte)0xFF;
        bstr[4]=(byte)0x00;
        bstr[5]=(byte)0x00;
        bstr[6]=(byte)'Z';
        bstr[7]=(byte)0x00;
        bstr[8]=(byte)0x00;
        bstr[9]=(byte)0x00;
         bstr[10]=(byte)0x00;
         bstr[11]=(byte)((bstr[6]+bstr[7]+bstr[8]+bstr[9]+bstr[10])%256);
          try
             {
                  mOutputStream.write(bstr);
                }
                catch (IOException e)
                {
                        e.printStackTrace();
                  }
          }
```

6． 系统设置模块实现

首先是该界面的实现类systemsetmain.java，他的实现方式为ListView设置Adapter，然后为ListView添加事件监听器，主要实现代码为：

```
ArrayAdapter<String> arrayAdapter = new ArrayAdapter<String>(
        this , Android.R.layout.simple_list_item_1 , arr);
        //为ListView设置Adapter
 list2.setAdapter(arrayAdapter);
 list2.setOnItemClickListener(new OnItemClickListener() {
     @Override
     public void onItemClick(AdapterView<?> arg0, View arg1, int arg2,
                 long arg3) {
            //系统设置----情景模式
             if(list2.getItemAtPosition(arg2).equals(arr[0]))
                 {

                  startActivity(new Intent(systemsetmain.this, contextualmodel.class));
                     }
            ……
     }
```

情景模式设置界面的实现方式与系统设置实现方式基本相同，这里不再介

绍。无论是系统默认配置还是用户自己配置的情景模式，都需要对其中的各种状态进行保存，前面章节已经提到对于3种情景模式各种状态的保存我们使用Android系统中的SharedPreferences，其主要实现代码如下所示。

```
public boolean onPreferenceClick(Preference preference) {
    // TODO Auto-generated method stub
    if (preference == mCheckbox1){
        if(myApplication.pamerment1[0]==true)
            myApplication.pamerment1[0]=false;
        else
            myApplication.pamerment1[0]=true;
        }
    else if (preference == mCheckbox2){
        if(myApplication.pamerment1[1]==true)
            myApplication.pamerment1[1]=false;
        else
            myApplication.pamerment1[1]=true;
        }
      ……
    return true;
        }
```

5.7 本章小结

本章是智能家居系统软件的实现与调试。首先介绍了开发环境的搭建，然后对 Android 应用程序的四大组件和 JNI 机制进行了简要分析，最后重点介绍了系统各模块实现过程。

第 6 章 智能家居系统软硬件综合调试与验证

6.1 系统调试

6.1.1 节点软件调试

这里我们需要用到的工具有 IAR、串口调试助手以及 CC2530 烧写器。IAR 主要用来编辑、调试节点程序。串口调试助手用来调试节点收发数据的正确性。CC2530 烧写器用来烧写编译好的节点程序。图 6-1 显示了在 IAR 中温湿度环境监测节点调试界面。

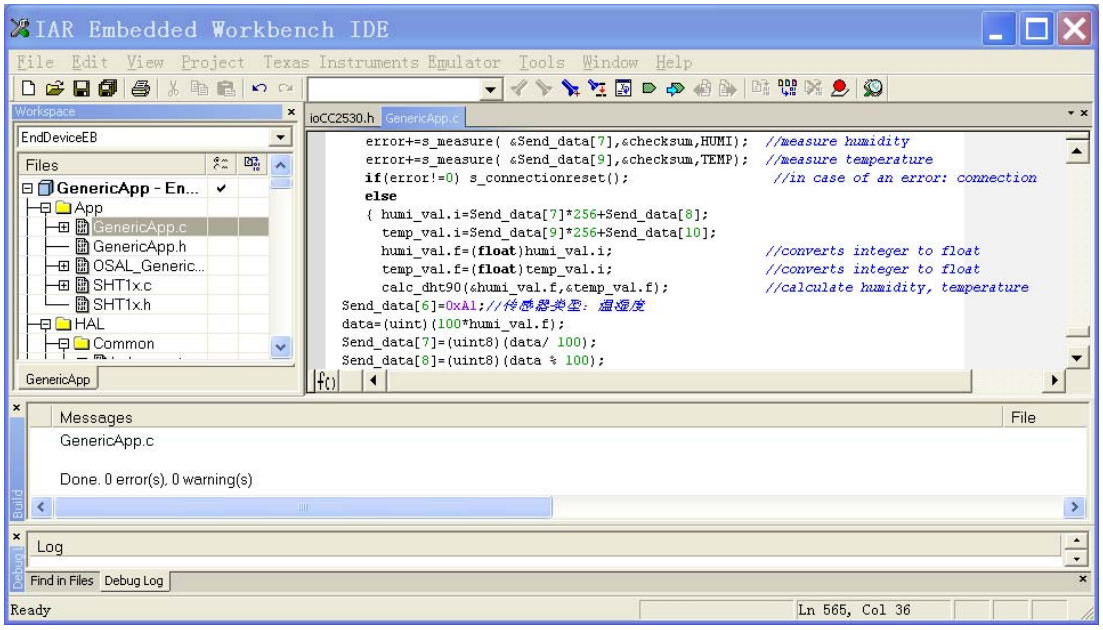

图 6-1 温湿度节点调试界面

6.1.2 主系统软件调试

Eclipse 允许用户在编辑过程中设置程序断点用户启动调试器之后，断点即被激活。断点可设置为条件表达式，变量或存储器访问，断点被触发后，调试器命令或调试功能即可执行。该系统的主机软件是在 Eclipse 开发环境下，通过 DDMS 直接与 S3C6410 开发板连接进行测试。其连接过程如下所示：

(1)将计算机和 S3C6410 用数据线相连，启动开发板。

(2)在用户第一次使用时，计算机会提示安装驱动。当驱动安装好后重启开发

板。

(3)打开 Windows 下的 dos 窗口输入以下命令：

adb devices

(4)若看到如图 6-2 所示的设备，这表示连接成功，这时就可以使用 Eclipse 里的各种调试工具进行调试了，其部分调试界面如图 6-3 所示。

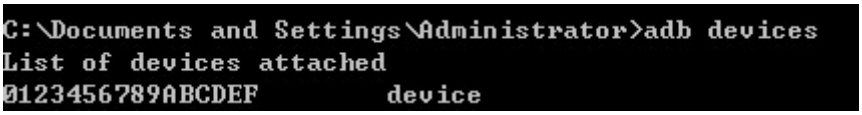

图6-2 设备连接成功

这里要强调一点是在与设备连接过程中，通常并不是一次就能连接成功，需要多次连接，在连接过程中需不断输入以下三个命令 adb kill-server、adb start-server 和 adb devices，直到连接成功为止。

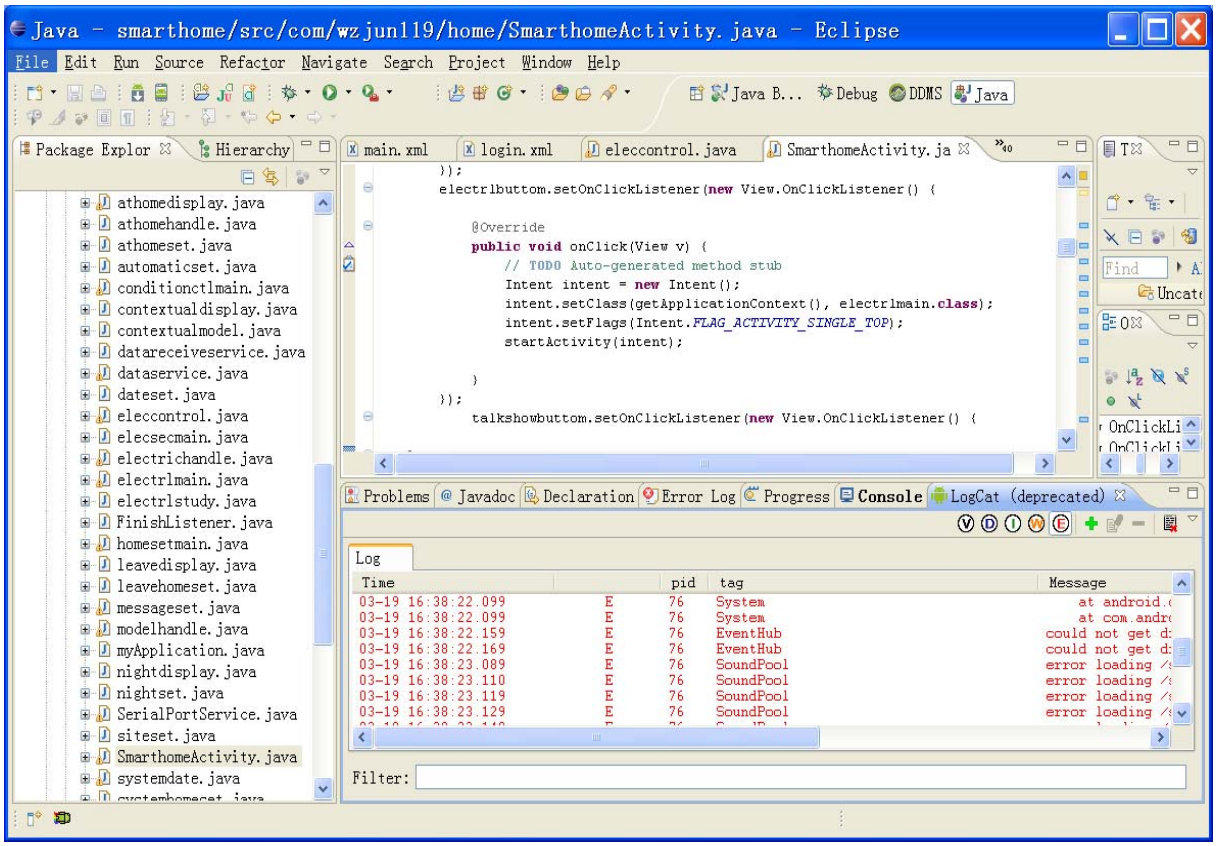

图6-3 Eclipse调试界面

6.2 系统运行结果与验证

将各节点模块分别上电，通过对该智能家居系统软件进行实际测试和验证，结果表明系统工作正常，基本上达到了前期的设计要求，验证了电路设计的合理

性。系统与主节点通过串口通信可以正常进行。通过 GPRS 模块，用户可以在终端手机上看到家居安防的各种提示信息。通过环境监测节点可以可查看前端节点的工作状态等等。

这里重点测试了家居安防、家电控制、状态检测这三个功能模块的实现。

(1)家居安防和家电控制模块测试。家居安防中以居家模式为例进行了测试，这里重点对家居模式中的连动功能进行测试。当系统设定的温度低于室内温度时，空调自动开启进入加热模式；当光照度低于系统设定的强度时，电灯自动打开。其测试结果如图 6-4 所示。

(2)状态检测模块测试。正常情况下，所有节点都是加入到该系统中的，若有节点出现异常，则表明该节点需要更换电池或维修。其测试结果如图 6-5 所示。

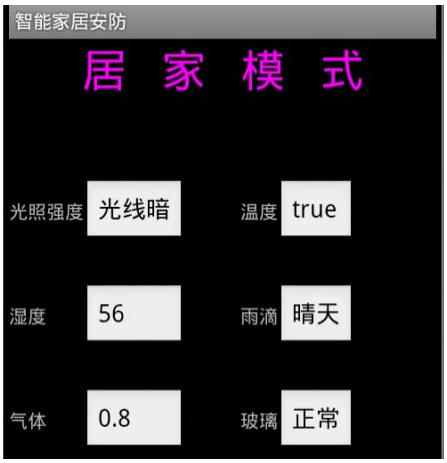

图 6-4 居家模式测试结果图

图 6-5 状态检测测试结果图

在调试和测试过程中存在的问题：

(1)对与前端节点的耗能进行测试时发现，根据前端传感器的功耗不同，其实际可正常工作时间有所区别，但所有节点在待机一个月后，节点电池供电不足，均不能正常工作。这里需要从软硬件设计两方面改进来降低电池功耗[48,49]。

(2)对家居安防和家电控制模块测试过程中发现，有时会出现以下情况：当家电命令以验证成功，在进行居家模式中温度过高需要开启空调时，虽然系统命令已发出，但是空调并未开启。初步判断应该是红外模块的学习代码有待于改进，其次应该考虑该硬件电路设计问题。

(3)在进行状态检测过程中发现，若节点个数超过 20 个（这些节点都能正常工作），系统中状态检测中有些节点会显示不正常。出现该问题应该是系统发出广播地址后，所有节点同时发送信息造成了帧的丢失，该问题可以通过系统轮训

方法得到解决。

6.3 本章小结

本章是对智能家居系统的整体测试。首先介绍了主机软件的调试方法，通过该方法就可以达到软硬件同步调试效果，然后介绍了该系统的部分功能的调试结果，最后在对系统测试过程中出现的问题进行总结分析，并提出了相应的解决办法。

第 7 章 总结和展望

7.1 论文工作总结

本文首先介绍了嵌入式处理器的发展状况，预测了 ARM 很有可能会超越 Intel 成为主流。接着介绍了国内外智能家居的现状，基于国外智能家居高昂的价格和国内智能家居混乱的局面，提出了在 Android 平台上开发出一套完整的智能家居系统。首先建立了本课题智能家居系统的整体结构，然后分别从硬件和软件两方面对智能家居系统进行阐述。硬件主要包括主控制器的选择和节点的设计，重点介绍了前端传感器的硬件原理图。软件方面分别从功能、设计和实现三方面对主机系统软件进行了详细的分析。最后对该系统进行了测试。这对于国内智能家居系统的设计起到了很好的借鉴作用。

本文的研究成果有以下几点：

(1)在现有知识的基础上，查阅了大量的参考文献和书籍，较为深入地学习系统编程所用的 Java 语言，在此基础上提出了本文的总体设计方案。

(2)通过对环境监测节点和执行节点功能的研究和分析，设计了相应的硬件电路。

(3)深入分析了 Z-stack 协议栈，完成了主机节点和终端节点的应用程序设计。

(4)深入研究了 Android 系统架构，掌握了从底层驱动的开发到 Android 应用程序的开发流程。

(5)在 Android 系统上设计了智能家居主程序，实现了室内环境与家电的连动模式，并且能够在系统监测到异常情况下，及时处理各种异常。

本文的创新点在于：

(1) 使用 Android 系统作为运行于智能家居主控制器和手持设备上的操作系统。目前市场上没有一款智能家居软件系统是基于 Android 的，就 Android 目前的发展前景来看，Android 很有可能成为智能家居主流操作系统。这样使得用户无论在主控制器还是在手持设备上都有统一的操作界面，使用更加方便。

(2)使用了 MVC 设计模式。该模式使得数据处理、业务处理和界面显示无论一旦哪一层的需求发生了变化，就只需要更改相应的层中的代码而不会影响到其

它层中的代码，降低了他们之间的依赖性，有利于组件的重用，提高了系统的可扩展性和可维护性。

(3)新型传感器的使用。本文将最新的雨滴、光照、温湿度传感器应用到智能家居的环境监测中，这些传感器都是采用国际领先技术生产的，其本身的具有很高的精确性和可靠性。

7.2 不足与展望

该智能家居系统已经实现了智能家居系统的大部分功能，基本完成了系统预设的目标。在对系统测试过程中发现的主要的问题有：当发送状态查询命令后，若节点过多，会导致节点信息丢失，造成状态显示错误；在设置了情景模式后，在传感器与家电进行连动控制时，有时系统发出的命令并未得到执行。

所以，本文的下一步工作应从以下几个方面入手：

(1)实现该系统中未完成的功能模块，在实现本模块的同时要注意与其他模块之间的关系。

(2)完善系统功能，找出并解决系统中所出现的问题。对于状态查询出现错误，应该是同一时间节点发送的数据过多导致串口接收缓冲区溢出，对于该问题的解决办法是主机程序中应该使用轮训方式对节点进行查询，从而避免信息在同一时间接收问题。对于情景模式中出现的问题应该从软硬件两方面入手，找出问题根源，确保智能家居系统的可靠性和稳定性。

(3)Android 系统中一个重要的优点是其图形界面美观，所以在完成系统功能后，应对其界面进行大量的美化工作，毕竟界面美观是用户使用该产品的首要因素。

参考文献

[1] 贾勇. 浅谈嵌入式处理器及其发展趋势[J]. 科技信息,2011,(22) :486-487.

[2] 王洪辉. 嵌入式系统 Linux 内核开发实战指南[M]. 北京:电子工业出版社,2009.

[3] http://baike.baidu.com/view/11200.htm

[4] http://www.jiaboohui.com/list-news/news/2011-11-16/5325.html

[5] http://baike.baidu.com/view/3550303.html?wtp=tt

[6] http://baike.baidu.com/view/77594.htm

[7] 李俊斌,胡永忠. 基于 CC2530 的 Zigbee 通信网络的应用设计[J]. 电子设计工程,2011,(16):108-111.

[8] 李志威,刘寿强,孟敬. 基于 CC2430/CC2431的无线传感器数据采集的方案研究与设计[J]. 现代计算机,2009,(32):133-135.

[9] 黎琼. 智能家居中红外遥控系统的设计与实现[D]. 华中科技大学,2007,1.

[10] Sensirion. SHT10 Humidity Sensor Dat asheet, 2007.

[11] 何安科. 基于 STM32 与光强度传感器 BH1750 的无线路灯控制系统[J]. 企业科技与发展.2011,(314):15-17.

[12] 王建,毛腾飞,陈英革. 基于 BH1750芯片的测光系统设计与实现[J]. 常熟理工学院学报,2011,(25):117-120.

[13] http://www.aykjw.com/product.asp?id=136.

[14] 李军,黄岚,王忠义. 基于 Z-Stack 协议栈的 WSN 能量管理策略[J]. 计算机工程,2011,4,(7):121-124.

[15] 曾宝国. Z-Stack 协议栈应用开发分析[J]. 物联网技术,2011,5,(1):71-73.

[16] 赖联有. Zigbee 协议分析及其实现[J]. 齐齐哈尔大学学报:自然科学版,2010,1,(1):47-50.

[17] 吴想想. 基于Android平台软件开发方法的研究与应用[D]. 北京邮电大学,2011,1.

[18] 孙晓宇. Android 手机界面管理系统的设计与实现[D]. 北京邮电大学,2009,5.

[19] 芦宁. Zigbee 无线技术在智能家居中的应用[D]. 哈尔滨工业大学,2006,06.

[20] 王健,刘忱. Z igBee 组网技术的研究[J]. 仪表技术,2008,(4):10-12.

[21] http://www.digi.com/technology/rf-articles/wireless-zigbee.jsp

[22] Zhao F.,L.Guibas. Wireless sensor networks:an information processing approach[M]. Morgan Kaufmann Publishers, 2004.

[23] 吴国宏. 新型温湿度传感器SHT10的原理及应用[J].单片机与嵌入式系统应用,2009,(4):53-55.

[24] 庞娜. 网状结构 Zigbee 无线传感器网络研究[D]. 吉林大学,2012, 6.

[25] 沈淀. 基于 Zigbee 技术和 Android 系统的智能家居系统设计[D]. 武汉理工大学,2011,5.

[26] 朱旭萍. 基于无线网络技术构建家电智能控制的研究[D]. 浙江工业大学,2010,3.

[27] 郭晖. 基于 ARM 的智能家居管理系统设计[D]. 东华大学,2011,1.

[28] 黄玉兰,刘静,王洪革,李志军. 基于 AT 指令集的 GPRS 智能通信系统[J]. 吉林大学学报:信息科学版,2009,7,(4):424-429.

[29] 李志伟. 基于 AT 指令的串行通信程序的设计[J]. 微计算机信息, 2007,(3) : 272-274.

[30] 盖索林．Android 开发入门指南[M]．北京:人民邮电出版社,2009.

[31] Ed Burnette. Hello, Android 3rd Edition[M].2010.

[32] Wallace Jackson . Android Apps for Absolute Beginners[M].2010.

[33] Cygwin user guide[EB/OL].www.cygwin.com.

[34] Satria, H. Wibowo, B. Kwon, J.B. Lee, J.B. Hwang, Y.S. A virtual development environment for embedded software using open source software [J]. This paper appears in: IEEE Transactions on Consumer Electronics ,May 2009.

[35] 杨丰盛. Android 应用开发揭秘[M]. 北京:机械工业出版社,2010.

[36] http://developer.Android.com/guide/index.html

[37] Android Kernel Issues.http://www.kAndroid.org

[38] 差沙. Android 开发手机应用[J]. 程序员，2008,(01):56-61.

[39] 韩超,梁泉. Android 系统原理及开发要点详解[M]. 电子工业出版社,2010.

[40] 叶炳发. Android 操作系统移植及关键技术研究[D]. 暨南大学,2010,5.

[41] 高晶,王建华. JNI 技术在嵌入式软件开发中的应用[J]. 哈尔滨师范大学自然科学学报,2007,(06):62-65.

[42] 赵宏伟.Android NDK 开发环境实现与应用[J].电脑知识与技术,2010,(35):10055-10060.

[43] 李刚. 疯狂 Android 讲义[M]. 北京:电子工业出版社,2011.

[44] (美)科波特著,魏永明译. Linux 设备驱动程序[M]. 中国电力出版社,2006.

[45] 金智义,张戟. 基于 Android 平台的串口通信实现[J]. 电脑知识与技术,2011,(13):2983-2990.

[46] LiXuDong, YanGaoshi, TangHai. Android Based Wireless Location and Surrounding Search System Design[J]. This paper appears in: 2010 Ninth International Symposium on Distributed Computing and Applications to Business, Engineering and Science, September 2010.

[47] Bin Peng, Hui Yan, Minghui Wu. Environment Research of Supporting Mobile Internet Practical Education Based on Android[J]. This paper appears in: 2010 Third International Conference on Education Technology and Training (ETT), November 2010.

[48] Amre El-Hoiydi, Jean-Dominique Decotignie, Jean Hernandez. Low Power MAC Protocols for Infrastructure Wireless Sensor Networks. In Proc. European Wireless (EW'04), 2004, 563-569.

[49] Andre Barroso, Utz Roedig, and Cormac J. Sreenan. UMAC: An Enemy-Efficient Medium Access Control for Wireless Sensor Networks. InProceedings of the 2nd IEEE European Workshop on Wireless Sensor Networks (EWSN2005), IEEE Computer Society Press, 2005:15-18.

致　谢

首先我要感谢我的导师段富海教授和马满福副教授，在整个研究生阶段无论是生活上还是学习上对我的悉心指导和帮助。感谢段老师在这三年中对我的关怀，使我有机会接触到嵌入式系统领域最新的技术和科研成果，在我撰写论文过程中，你的不断督促是我能在规定的时间内完成论文的重要保障。感谢马老师在我论文开题中给我的指导。在此，谨向两位导师致以我最诚挚的谢意。

感谢无锡泛太科技有限公司给我提供了实习的机会和我在做毕业论文时所用到的各种硬件设备。在无锡泛太实习期间，不仅学到了更多的专业知识，而且学会了如何为人处事，这段经历使我受益匪浅。感谢无锡泛太的各位领导对我的关怀和理解，感谢无锡泛太的各位同事，感谢徐升辉、郑晓、袁媛和毕海顺在我设计和实现该系统时对我的帮助，同时感谢我的师弟王诚瑞和赵奎斌给我的建议。

感谢计算机院张贵仓老师、王彩芬老师、冯百明老师、张明新老师、杨勇老师、马满福老师、蒋芸老师、王治和老师、党小超老师、任小康老师给予的指导、关心和帮助。在三年的研究生学习生涯中，各位老师给了我无价的知识和真诚的关爱，让我见识了众多研究领域的精华，在此表示衷心的感谢。

感谢我的室友王志强、周向前和孙春虎以及班里的每一位同学在学习和生活上给予的帮助。他们良好的生活作风和刻苦求学的专研精神时刻提醒我要不断向前进步，正是由于他们的帮助和鼓励，我才能克服一个个困难与疑惑。与他们在生活中友好相处，使我受益匪浅，在此致以诚挚的谢意。

最后，感谢我的父母和亲友，他们默默地理解和支持，是我能够顺利完成硕士学业的坚强后盾，感谢他们在生活上对我莫大的帮助，我的每一次成功，都离不开他们的关心和支持。

再次感谢所有关心和帮助我的人！

吴 志 君

二〇一二年三月于西北师范大学

攻读硕士学位期间学术成果

[1] 吴志君,段富海. 嵌入式 Linux 系统内存优化使用方法研究[J]. 甘肃科学学报,2012,3(1):119-122.

基于ARM与Android的智能家居系统设计与实现

作者：吴志君
学位授予单位：西北师范大学