南京邮电大学

毕 业 设 计（论 文）

|  |  |
| --- | --- |
| 题 目 | 基于ZigBee的红外入侵智能报警器设计与实现 |
| 专 业 | 自动化 |
| 学生姓名 | 戚彬 |
| 班级学号 | B11050325 |
| 指导教师 | 王冬生 |
| 指导单位 | 自动化学院 |

日期： 年 月 日至 年 月 日

毕业设计（论文）原创性声明

本人郑重声明：所提交的毕业设计（论文），是本人在导师指导下，独立进行研究工作所取得的成果。除文中已注明引用的内容外，本毕业设计（论文）不包含任何其他个人或集体已经发表或撰写过的作品成果。对本研究做出过重要贡献的个人和集体，均已在文中以明确方式标明并表示了谢意。

论文作者签名：

日期： 年 月 日

摘 要

近年来，随着社会的发展和科学技术的进步，家庭、银行、监狱、医院、军事重地、重要物资仓库等区域都需要用入侵报警系统加以有效的防范，以确保人身和财产安全。因此，入侵报警系统逐渐成为社会关注的焦点，用户的需求也越来越迫切。这其中，红外线入侵报警器因其成熟的技术、出色的监控效果和相对低廉的价格，受到了许多用户尤其民用市场的追捧，占据了大量的市场份额。同时，无线传感器网络的飞速发展必将使报警器向着以下的方向发展——智能化、网络化、便于组网。

本文设计了一种红外入侵报警器，它以ZigBee无线通信技术为基础，使用MSP430单片机作为控制端，采用被动式热释电红外传感器作为入侵信号监测装置，通过声音及灯光报警。论文从研究现状、研究意义等出发，主要介绍ZigBee协议及软硬件的设计，最终得出研究结果以及结论。结果表明，本课题可以很好地在监控区域内实现入侵报警及解除报警功能，同时具备低功耗、高灵敏度以及易于组网等优点。

关键词：红外入侵报警器；ZigBee技术；MSP430F149单片机；无线传感器网络

**ABSTRACT**

In recent years, with the development of society and progress of science and technology, some areas such as family, banks, prisons, hospitals, military areas, important materials warehouse need to be effectively prevented. In order to ensure personal security and property safety, intrusion alarm systems are used in these areas. Thus, intrusion alarm system has become the focus of attention. User demands have become more and more urgent also. The infrared intrusion alarm technology has excellent monitoring results and relatively low product price. Since it has developed more mature, this technology has received numerous users’ pursuit. Especially in the civilian market, this alarm system accounts for a significant market share. Meanwhile, the wireless sensor network technology has become a hot spot. So the alarm devices advance towards the direction of intelligence, networking and easy network establishment.

This paper designs an infrared intrusion alarm. It is based on ZigBee technology and takes MSP430 MCU as the control terminal. It also uses passive pyroelectricity infrared sensor as the intrusion signal monitoring device, and alarms by sound and light. This design starts from the research status and significance, and mainly focuses on ZigBee protocol and the design of software and hardware. Finally, the research result and conclusion are obtained. The result shows that the project can realize the intrusion alarm function and disarm the alarm, along with the advantages of low power consumption, high sensitivity, easy network establishment and so on.

**Key words：**infrared intrusion alarm；ZigBee technology；MSP430F149 MCU；wireless sensor network

目 录

[第一章 绪论 1](#_Toc421708716)

[1.1课题研究现状 1](#_Toc421708717)

[1.1.1红外入侵智能传感器的研究进展 1](#_Toc421708718)

[1.1.2红外入侵智能传感器的组织结构 2](#_Toc421708719)

[1.1.3红外入侵智能传感器的特点 2](#_Toc421708720)

[1.2课题研究意义 3](#_Toc421708721)

[1.3技术要求及设计方案 3](#_Toc421708722)

[1.3.1技术要求 3](#_Toc421708723)

[1.3.2设计方案 3](#_Toc421708724)

[1.4内容安排 4](#_Toc421708725)

[第二章 ZigBee通信协议 5](#_Toc421708726)

[2.1 ZigBee协议概述 5](#_Toc421708727)

[2.2 ZigBee设备类型及其拓扑结构 5](#_Toc421708728)

[2.2.1 ZigBee的设备类型 5](#_Toc421708729)

[2.2.2 ZigBee的拓扑结构 5](#_Toc421708730)

[2.3 ZigBee协议栈架构 6](#_Toc421708731)

[2.3.1物理层 7](#_Toc421708732)

[2.3.2介质访问控制子层（MAC子层） 7](#_Toc421708733)

[2.3.3网络层 8](#_Toc421708734)

[2.3.4应用层 8](#_Toc421708735)

[2.4 ZigBee的技术特点 8](#_Toc421708736)

[2.5 ZigBee的应用前景 9](#_Toc421708737)

[第三章 硬件设计 10](#_Toc421708738)

[3.1硬件选材 10](#_Toc421708739)

[3.2电路设计 10](#_Toc421708740)

[3.2.1电源模块 10](#_Toc421708741)

[3.2.2复位电路 11](#_Toc421708742)

[3.2.3系统时钟 12](#_Toc421708743)

[3.2.4 JTAG接口 12](#_Toc421708744)

[3.2.5报警和消警电路 13](#_Toc421708745)

[3.2.6串口通信电路 14](#_Toc421708746)

[3.3制作过程 14](#_Toc421708747)

[3.3.1 PCB板制作 15](#_Toc421708748)

[3.3.2元件焊接及各模块连线 17](#_Toc421708749)

[3.4传感器模块 17](#_Toc421708750)

[第四章 软件设计 22](#_Toc421708751)

[4.1软件设计总体方案 22](#_Toc421708752)

[4.2控制模块编程 22](#_Toc421708753)

[4.2.1初始化模块 22](#_Toc421708754)

[4.2.2中断模块 24](#_Toc421708755)

[4.2.3主程序模块 25](#_Toc421708756)

[4.3 ZigBee模块编程 27](#_Toc421708757)

[4.3.1初始化模块 27](#_Toc421708758)

[4.3.2事件处理函数 30](#_Toc421708759)

[4.3.3终端工作过程 30](#_Toc421708760)

[4.3.4协调器工作过程 32](#_Toc421708761)

[第五章 调试及性能分析 34](#_Toc421708762)

[5.1调试软件介绍 34](#_Toc421708763)

[5.2 MSP430工程配置和调试 34](#_Toc421708764)

[5.2.1工程配置 34](#_Toc421708765)

[5.2.2软件编译和调试 34](#_Toc421708766)

[5.3 ZigBee模块工程配置和调试 35](#_Toc421708767)

[5.3.1工程配置 35](#_Toc421708768)

[5.3.2编译和调试 35](#_Toc421708769)

[5.4性能分析 35](#_Toc421708770)

[5.4.1调试过程中的问题 35](#_Toc421708771)

[5.4.2低功耗分析 37](#_Toc421708772)

[5.4.3实验结果 38](#_Toc421708773)

[结束语 39](#_Toc421708774)

[致 谢 40](#_Toc421708775)

[参考文献 41](#_Toc421708776)

[附录A 43](#_Toc421708777)

[附录B 45](#_Toc421708778)

第一章 绪论

在如今，以无线通信技术[1]及计算机通信技术发展最为迅速，红外入侵智能传感器[2]便在此背景下诞生。这个技术基于无线传感器网络，将应用广泛的无线通信与传感器的入侵检测相融合，同时以单片机作为控制模块，实现智能报警功能。

在当今信息社会，人们的信息交流因物联网[3]技术提供的通信平台变得十分快捷、方便。红外入侵智能传感器使人们获取信息的途径变得丰富，在发达的互联网技术的帮助下，可以智能地将现实世界中的物理信息通过虚拟的网络世界传输给我们，使人们工作效率和生活水平得到了极大地提高。

1.1课题研究现状

1.1.1红外入侵智能传感器的研究进展

（1）国外进展

在政府和社会各方的支持下，美国等其它科技较为发达国家于二十世纪末便开展了红外入侵智能传感器网络的研究和开发[4]。学术领域、民用领域和军事领域从二十一世纪开始重点关注该项目。

①学术领域

美国各种与传感器技术相关的机构及基金会每年都会拨发大量资金给各知名高校，如加州大学洛杉矶分校、麻省理工学院、哈佛大学等，在学校内建立传感器网络研究中心[5]，培养了大量拥有专业知识的人才。

②民用领域

在交通、医疗等领域，利用红外入侵传感器[6]提出了各种造福人民的规划和相关项目，方便了人们的日常出行，保障了居民的财产安全和身体健康。人们的家居环境将更加舒适便捷、更加智能，智能家居[7]将传感器节点与家电或家具相结合，利用无线网络连接，可以实现各种远距离控制。例如，智能灯光控制、智能电器控制和安防监控[8]系统。

③军事领域

传感器技术最早就是为军事应用所设计的，因此传感器在战场上的应用一直都受到军事发达国家的高度重视。在战场上，通过大量布设的传感器节点，将传感器收集到的各项数据和信息汇聚到情报分析机构，从而达到对战场情况的充分掌握。在美国，无论陆军还是海军，都已全面部署了该作战系统，为了今后的现代化战争打下了坚实的基础。

（2）国内进展

国内对于传感器技术的研究[9]正处于起步阶段，但在国家和各种研究机构的大力扶持下正飞速发展，已初步形成规模，与技术发达国家的差距正在不断缩小。因此，相信在不久的将来，传感器技术必将在我国广泛传播，使我国早日跻身世界先进行列。

1.1.2红外入侵智能传感器的组织结构

红外入侵智能传感器系统通常由以下几个部分组成：传感器节点（Sensor Node）、汇聚节点（Sink Node）和管理节点（Manage Node）[10]，它的结构体系如图1.1所示。

监测区域内包含大量传感器节点，它们在监测到数据后通过其它传感器节点逐跳地传输，并最终到达汇聚节点，最后再由互联网和卫星将数据发送到管理节点。用户可以通过管理节点向传感器节点发布监控任务[11]以及处理监控信息。

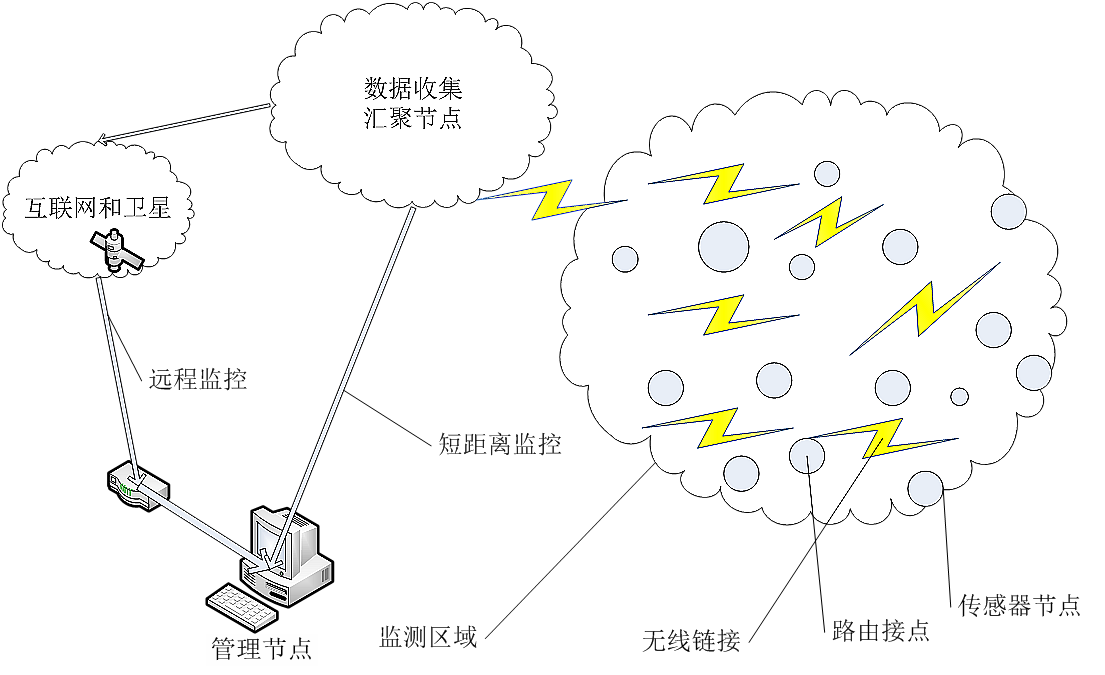


图1.1 红外入侵智能传感器的结构体系

1.1.3红外入侵智能传感器的特点

（1）节点数量众多，分布密集：

传感器网络中的节点数量繁多[12]，可能达到成百上千万，甚至更多。大量且密集的传感器节点不仅可以测量得到更详细、准确的数据，还可以保证整个网络的稳定；

（2）自组织网络：

在传感器网络应用中，传感器节点的位置是随机摆放的，相邻节点之间的位置关系也不得而知，这就要求传感器节点具有自组织能力[12]，能够自动进行配置和管理；

（3）动态性：

传感器电量耗尽或遭到外部破坏，再加上经常有新节点加入网络，这些原因使得传感器网络的节点数量不断变化[12]，就要求网络的拓扑结构能够根据节点数量的变化而不断做出调整；

（4）以数据为中心：

用户不会关心传感器的硬件问题，而是对传感器产生的数据[12]感兴趣，因此传感器必须重点关注数据问题；

（5）应用相关：

现实世界的信息种类繁多，不同的传感器能够收集不同的信息[12]。在开发传感器网络时，要根据具体的应用领域，定制相应的研究方法，专注于传感器网络的差异性。

1.2课题研究意义

红外入侵智能传感器因其高效可靠的性能、方便快捷的应用开发以及极其低廉的产品价格，正不断走进社会生活的各方面领域，在人们日常生活中的应用也越来越广泛[13]。尤其是目前数据处理、控制硬件技术的飞速发展和传感器硬件技术的发展以及新的技术不断成熟，红外入侵智能传感器技术必将在未来得到更广泛的应用。

1.3技术要求及设计方案

1.3.1技术要求

（1）初始化时，所有LED（包括入侵信号显示LED和报警LED）灭和蜂鸣器不响；

（2）当某个终端检测到入侵信号时，对应的显示LED亮；

（3）如果3个终端任意一个检测到入侵信号，报警LED亮，蜂鸣器响；

（4）报警后，按下清除报警按键，关闭报警LED和蜂鸣器。

1.3.2设计方案

红外入侵报警器[14]以MSP430单片机为控制器，通过串口与ZigBee协调器[15]模块互相通信，协调器不断接收和它组成同一网络的3个终端所发出的入侵信号。当控制器接收到报警信号时，控制蜂鸣器和LED报警，然后通过按键消除报警信号。

总体设计方案如下图1.2所示。

监控区域1

被动式

红外探测器

ZigBee

终端

ZigBee

协调器

MSP430

控制模块

监控区域2

监控区域3

图1.2 系统设计方案

1.4内容安排

本课题其它内容安排如下：

（1）第二章介绍ZigBee协议栈，主要包括协议栈简介、协议栈构架及协议栈特点等；

（2）第三章介绍硬件设计，主要包括各硬件模块的选择和电路设计、制作过程；

（3）第四章介绍软件设计，主要包括控制模块编程和ZigBee模块编程；

（4）第五章介绍调试及性能分析，主要包括调试软件的工程配置和系统性能分析；

（5）最后几部分是结束语、致谢、参考文献和附录。

第二章 ZigBee通信协议

2.1 ZigBee协议概述

ZigBee 技术是一种近距离、低复杂度、低功耗、低速率、低成本的[16]双向无线通信技术，它基于IEEE 802.15.4协议栈，主要用于距离短、功耗低且传输速率不高的设备之间组网通讯。

在自然世界，蜜蜂（bee）发现花蜜后会通过一种特殊的八字舞蹈动作，具体形式为“嗡嗡”（zigzag）地振动翅膀，来告知同伴具体的位置，它们就是通过这样一种方式来达到交换信息的目的[21]。“ZigBee”一词就来源于此，也寓意该协议近距离、低复杂度等特点。

2001年8月，ZigBee联盟成立[17]，它的主要工作是研发下一代红外入侵智能通信标准——“ZigBee”，所有这些公司都参与了开发ZigBee物理层和介质访问控制层（MAC层）的IEEE 802.15.4标准。

Z-stack[19]协议栈是由TI公司提出的一种ZigBee解决方案，现已被大部分ZigBee应用开发厂商所采用。该协议栈符合ZigBee2007/PRO规范，可以在CC2530[20]等芯片上实现完全的ZigBee。

2.2 ZigBee设备类型及其拓扑结构

2.2.1 ZigBee的设备类型

ZigBee网络的设备类型有三种，即协调器节点、路由节点和终端节点[20]，其中协调节点和路由节点为全功能设备（FFD：Full-Function Device），而终端设备只需要选用精简功能设备（RFD：Reduced-Function Device）。

协调器：ZigBee网络中至少要有一个协调器，并且只能有一个协调器。协调器主要负责网络的启动和配置，当这些工作完成后，协调器将实现路由节点的功能，协调器相当于整个网络中的大脑，需要足够的存储空间和计算能力；

路由器：路由器可以用来扩展网络，组建更加大型的网络，控制节点的路由信息。路由节点一般只有在树状网络和网状网络中存在，在星形网络中是不存在的。路由器能够判断是否让终端节点加入网络，并完成转发终端节点和协调器节点之间的数据。有扩展网络的能力，具有成为父节点的潜力，使用路由器来让网络加入更多的设备；

终端设备：终端设备主要负责测量数据和发送数据，或者执行接收到的指令，没有成为父节点或路由器的能力。终端设备作为网络的执行设备，直接与实际的监控对象相接触，是ZigBee无线传感网络的重要组成部分，ZigBee网络的功能主要是由终端设备来完成的。

2.2.2 ZigBee的拓扑结构

ZigBee网络层支持3种拓扑结构：星型、树型和网状型网络结构[21]，如图2.1所示。

树型网

网状网

星型网

图2.1 ZigBee协议的拓扑结构

本课题采用星型网络结构。在星型网中，整个网络的形成以及数据的传输由中心的网络协调器集中控制，所有设备只与中心的PAN协调器通信。协调器一般需要持续供电，而终端设备则采用电池供电。星型网络通信方式简单，建立网络的成本低廉，适合智能家居、公司安保以及健康监测等小范围应用。

2.3 ZigBee协议栈架构

ZigBee协议栈基于开放系统互连模型（OSI），网络体系结构如图2.2所示。每一层都实现一部分通信功能，并向高层提供服务。IEEE 802.15.4标准定义最下面的两层：物理层（PHY）和介质访问控制子层（MAC）。ZigBee联盟提供了最上层的网络层（NWK）和应用层（APL）框架的设计，其中应用层的框架包括了应用支持子层（APS）、ZigBee设备对象（ZDO）和由制造商定义的应用对象。ZigBee联盟还开发了安全层，防止在通信过程中泄露相关标识，保障信息安全。

|  |  |
| --- | --- |
| 应用层 | ZigBee联盟 |
| 网络层/安全层 |
| MAC层 | IEEE 802.15.4 |
| 物理层 |

图2.2 ZigBee网络体系结构图

2.3.1物理层

（1）物理层主要功能

物理层的主要功能[22]有：工作频段的分配、信道的选择、信道能量检测（ED）、空闲信道评估（CCA）、链路质量指示（LQI）的检测以及为MAC层服务提供数据服务和管理服务。

其中，信道能量检测用于为协调器提供当前频段所有信道的能量信息，以判断是否有空闲信道可以加入新的网络；空闲信道评估判断的是当前信道是否处于空闲状态，从而决定是否发送当前数据帧；链路质量指示为高层协议提供数据传输的质量来选择路由方式。

物理层定义了三个载波频段，分别是868MHz、915MHz和2450MHz（一般称为2.4GHz）。其中2.4GHz在全世界通用，868MHz和915MHz分别只在欧洲和北美使用。三个载波频段共包含了27个信道，这些信道的频率分别如下列公式2-1计算：

 式（2-1）

以上三式中k指的是信道号，fc的单位为MHz。

1. 物理层服务规范

物理层[22]主要有两种服务接口（SAP：Service Access Point），分别是物理层管理服务接口（PLME-SAP）和物理层数据服务接口（PD-SAP）。它们除了负责在物理层和MAC层之间提供管理和数据服务之外，还负责维护物理层PAN信息库（PHY PIB）。

2.3.2介质访问控制子层（MAC子层）

（1）MAC子层的主要功能[22]：

①如果是协调器就产生网络信标；

②在两个设备的MAC层之间提供可靠的通信链路；

③支持PAN网络的加入和退出；

④使用CSMA/CA机制来访问信道。

IEEE 802参考模型将数据链路层划分为MAC子层和逻辑链路控制（LLC：Logical Link Control）子层，MAC子层将LLC子层独立开来，解决了物理层与数据层共享信道的问题。

（2）MAC层服务规范

MAC层包括MAC层公共部分子层（MCPS）和MAC层管理服务（MLME），它们定义了数据服务接口（MCPS-SAP）和管理服务接口（MLME-SAP），保证MAC层数据帧与物理层数据服务的正确传输[22]和提供MAC层管理服务，还负责维护MAC层 PAN信息库（MAC PIB）。

2.3.3网络层

（1）网络层的主要功能

网络层主要实现建立网络，节点加入或离开网络，以及管理和配置网络[23]。

①协调器具有建立新网络的能力；

②协调器和路由器具备允许设备加入网络或者离开网络、为设备分配网络内部的逻辑地址、建立和维护邻居表等功能；

③终端节点只需要加入或离开网络的能力即可。

（2）网络层服务规范

网络层内部提供了网络层数据服务（NLDE）和网络层管理服务（NLME），它们通过访问数据服务接口（NLDE-SAP）和管理服务接口（NLME-SAP）为上层提供数据传输服务和管理服务[24]，还负责一些网络层的管理工作和管理网络信息中心（NIB）。

2.3.4应用层

（1）应用层的主要功能

应用层由三个部分组成，包括应用支持子层（APS）、ZigBee设备对象（ZDO）以及制造商定义的应用对象[25]。它们三部分协同工作，分别实现不同的功能，为用户开发ZigBee应用提供入口。

（2）应用支持子层服务规范

应用支持子层[26]通过应用支持子层数据服务（APSDE）和应用支持子层管理服务（APSME）来提供服务，其中APSDE通过数据服务接口（APSDE-SAP）提供应用层帧[27]协议数据单元的处理服务，并为APDU加入合适的协议头；APSME通过管理服务接口（APSME-SAP）提供设备发现[28]、设备绑定等服务，还提供应用层信息库（AIB）的管理和安全管理。

2.4 ZigBee的技术特点

目前，无线通信技术主要有ZigBee、wifi、蓝牙等几种模式[27]。下表2.1展示出了这几种通信技术的性能指标。

经过对上述几种无线通信技术的比较，我们可以明显地发现ZigBee技术在构建红外入侵智能传感器方面具有无可匹敌的优势。

（1）功耗低：当ZigBee设备处于低耗电待机模式时，两节5号电池就可以维持工作长达半年到两年左右的时间[28]，避免了频繁更换电池或充电的麻烦，从而减轻了网络维护负担；

（2）成本低：因为ZigBee协议栈结构简单[29]，开发设计非常方便，所以其研发和生产成本较低。并且ZigBee协议不用收取专利费，所以大大降低了成本。

（3）容量大：每个ZigBee网络最多可支持255个设备；

（4）时延短：通常时延都在15ms至30ms之间；

（5）安全性好：ZigBee使用载波检测多址访问/冲突避免（CSMA/CA）的方式传输数据[30]，数据都经过确认和检验，而且最高级的安全模式采用高级加密标准（AES），可以极大地提高网络安全性；

（6）工作频段灵活：使用的频段分别为868MHz（欧洲）、915MHz（美国）及2.4GHz（全球），均为免执照频段。

表2.1 红外入侵智能技术对比

|  |  |  |  |
| --- | --- | --- | --- |
| 种类 | ZigBee | 蓝牙 | Wi-Fi |
| 单点覆盖距离 | 10~200m | 10m | 50m |
| 扩展性 | 自动扩展 | 无 | 无 |
| 电池寿命 | 1～2年 | 1～2天 | 2～3小时 |
| 复杂性 | 简单 | 复杂 | 非常复杂 |
| 传输速率 | 250kbps | 1Mbps | 11Mbps |
| 频段 | 868/915MHz/2.4GHz | 2.4GHz | 2.4GHz |
| 节点数 | 65000 | 8 | 50 |
| 联网需要时间 | 30ms | 10s | 3s |
| 使用成本 | 低 | 低 | 高 |
| 安全性 | 128bitAES | 64bit,128bit | SSID |
| 可靠性 | 高 | 高 | 一般 |

2.5 ZigBee的应用前景

ZigBee因其低功耗、低成本、易于组网、工作稳定等一系列特点，正不断渗入到日常生活、国家安全、工业生产等各个领域[31]，具有广阔的应用前景。相信在未来的几年里，ZigBee技术必将在无线通信市场占据举足轻重的地位。ZigBee的应用领域如图2.3所示。

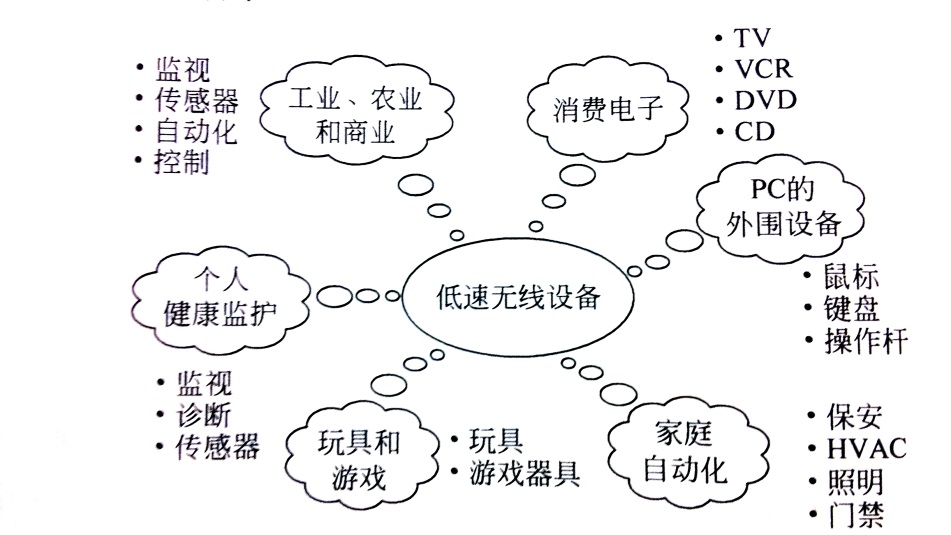


图2.3 ZigBee的应用场合

第三章 硬件设计

3.1硬件选材

TI公司的MSP430F149单片机[32]具有超低功耗、强大的CPU内核、灵活多样的外围模块、Flash型存储器、方便高效的开发环境等特点，因此采用MSP430F149 单片机作为控制器，接收报警信号以及控制报警模块。

同样是TI公司的CC2530芯片具有高性能的8051MCU内核、灵敏度极高和抗干扰能力强、支持硬件调试、低功耗、强大灵活的开发工具等特点，因此采用CC2530ZigBee模块进行入侵信号的发送（终端）和接收（协调器）以及终端和协调器之间的组网。

采用被动式热释电红外传感器进行入侵信号的探测；采用PCB板进行元件的焊接和连接；采用蜂鸣器和LED作为报警模块。

3.2电路设计

本课题的电路设计以MSP430F149芯片的相关电路为主，包括电源模块、复位电路、时钟电路、JTAG接口、报警电路和串口通信电路等。

3.2.1电源模块

稳定的电源使单片机正常工作的保障，要根据设计需求选择合适的供电方式和供电线路。本系统采用mini USB接口供电，输入电压为5V。然后通过开关导通后点亮电源指示灯后经过AM1117稳压器的电压转换功能，使电压变为3.3V，便可以供CPU（包括MSP430F149和CC2530芯片）和其它外设（包括蜂鸣器和LED）使用。

为了减小电源输出的纹波并降低其对系统的干扰，本系统在电源的输出端连接了一个电容，以滤除高频和低频的干扰。电路连接如下图3.1所示。

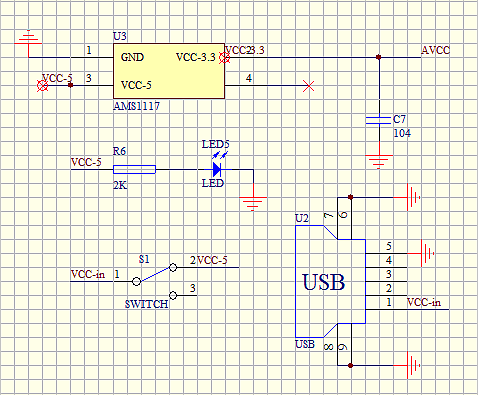


图3.1 电源模块原理图

其中，Vcc-in是USB接口的输入电压，Vcc-5是通过开关后的5V电压，Vcc-3.3是稳压后的数字电路电压，AVCC是滤波后的模拟电路电压。

3.2.2复位电路

复位电路存在于大部分单片机设备上，可以防止CPU一上电就进入工作状态，导致发出错误的指令、执行错误的操作，从而提高系统的稳定性和可靠性。复位电路较为简单，只需将轻触按键的两端分别接地和电源正极，然后让复位引脚与电源正极相连。

当按下复位按键后，复位引脚便于地导通，输入低电平使单片机复位。值得注意的是，按键必须与一个有极性电容相并联，否则会使系统无法正常启动。电路连接如下图3.2所示。

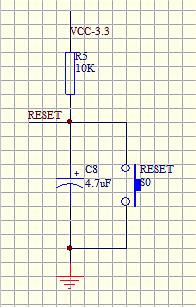


图3.2 复位电路原理图

图中，RESET即为MSP430单片机的复位引脚。

3.2.3系统时钟

系统时钟是单片机正常执行指令和各模块稳定工作的来源，MSP430F49的基本时钟模块由3个独立的振荡器时钟源[32]组成，它们分别是：

（1）LFXT1：

使用标准石英晶体或陶瓷谐振器构成的高低频兼容的振荡器，频率范围为450kHz~8MHz，本系统采用的是32.768KHz晶振；

（2）XT2：使用由标准石英晶体或陶瓷谐振器构成的高频振荡器，频率范围为450KHz ~8MHz，本系统采用的是8MHz晶振；

（3）DCO：使用RC电路构成的数控振荡器，频率范围为100kHz~10MHz。

用上述3个振荡器可以产生3种时钟信号分别供单片机内部的CPU和各个外设模块使用，这3个时钟信号分别是：

（1）ACLK辅助时钟信号：ACLK来自LFXT1时钟源，经过1/2/4/8分频后得到，可由软件选择供各个外设模块使用；

（2）MCLK主时钟信号：MCLK来自3 个时钟源，它们分别是 LFXT1CLK、 XT2CLK和 DCO时钟源信号，经1/2/4/8分频后用于CPU和系统做时钟；

（3）SMCLK子系统时钟信号：SMCLK来自2个时钟源，它们分别是 XT2CLK和 DCO时钟源信号，经1/2/4/8分频后供外设模块使用。

本系统外接了两个低频、高频晶振来提供稳定地时钟信号，电路连接如下图3.3所示。

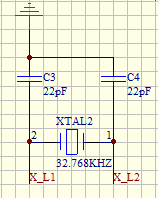
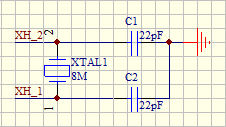
 

图3.3 可选的32.768KHz低频晶振和8MHz高频晶振

图中，X\_L和X\_H分别是高频和低频的时钟源输入信号。

3.2.4 JTAG接口

JTAG接口用于在线调试和下载程序，由于调试结果可以直接显示出来，可以极大地提高软件编程的效率。标准的JTAG接口是4线：[TMS](http://baike.baidu.com/view/1163589.htm)、[TCK](http://baike.baidu.com/view/160759.htm)、[TDI](http://baike.baidu.com/view/242148.htm)、[TDO](http://baike.baidu.com/view/3742112.htm)，各引脚的功能如表3.1所示。

表3.1 JTAG各引脚功能表

|  |  |
| --- | --- |
| 引脚 | 功能 |
| TMS | 测试模式选择 |
| TCK | 测试时钟输入 |
| TDI | 测试数据输入 |
| TDO | 测试数据输出 |

JTAG接口的电路连接如下图3.4所示。

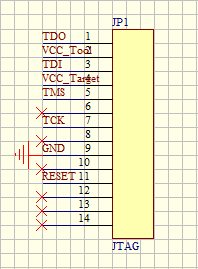


图3.4 JTAG接口电路原理图

图中，VCC\_Tool用于通过JTAG接口为系统供电。

3.2.5报警和消警电路

本系统采用蜂鸣器和LED声光报警，可以有效的发出报警信号。同时使用轻触按键清除报警，非常快捷方便。

蜂鸣器的输入信号由P6.4端口提供，高电平启动方式。由于单片机系统的电流不足以驱动蜂鸣器发出声音，因此在蜂鸣器输入端接入一个三极管放大电路，以保证其正常鸣响。4个LED分别连接P6.0~P6.3，为低电平启动方式。

这里需要注意的是，LED必须与一个电阻串联以减小电流，否则会导致LED烧毁。按键的输出信号连接到P1.0端口，工作原理与复位按键相同，不再赘述。电路连接如下图3.5所示。

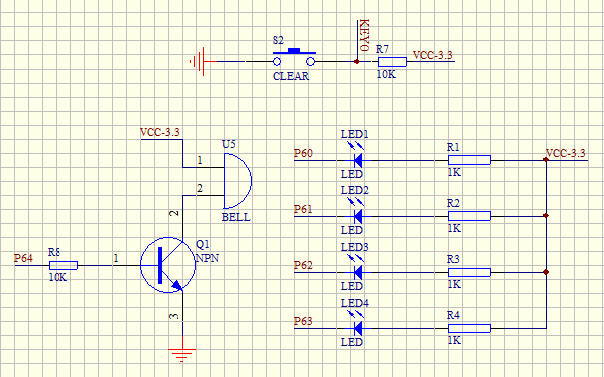


图3.5 报警和消警电路原理图

3.2.6串口通信电路

（1）MSP430系列单片机中的串行通信接口可以使用一个硬件模块支持两种通用串口通信模式：异步UART模式和同步SPI模式[32]。可以根据寄存器UxCTL的 SYNC位来选择USART的工作模式，当SYNC=0时 ，USART进入UART模式，通过两个引脚URXDx 、UTXDx与外部系统进行通信，这里的x是USART的编号，其数值为0或1。

（2）CC2530的串行通信接口[33]能够分别运行两种串口通信模式，分别是UART异步模式或SPI同步模式。可以通过配置寄存器UxCSR的MODE位选择这两种模式，当MODE=1时 ，USART进入UART模式，通过两个引脚RX、TX与外部系统进行通信，这里的x是USART的编号，其数值为0或1。

本系统设置MSP430芯片与CC2530芯片之间通过串口0的UART模式进行通信。电路连接如下图3.6所示。

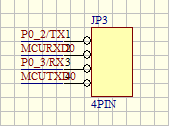


图3.6 串口通信电路

图中，MCUTXD0、MCURXD0分别和RX、TX通过跳线帽相连，实现两个芯片的串口通信。

3.3制作过程

3.3.1 PCB板制作

（1）制作软件介绍

本课题选择的电路板制作软件是Altium公司的Protel 99SE[36]。该软件功能强大，包括原理图设计、PCB图设计、电路模拟仿真和逻辑设计检查，界面人性化、提供各种快捷键和工具条，操作简单方便，使得电路板设计变得轻松高效。

（2）制作过程

一般PCB基本设计流程[36]如下：制作原理图库和封装库->绘制原理图->规划电路板->PCB布线->PCB布线->设计规则检查->制版。   
 第一：制作原理图库和封装库。准备好所需要原理图库和封装库是进行PCB设计的基础工作。软件自带的原理图库和PCB库通常没有我们设计电路所包含的元件，这就需要我们自己动手绘制。原理图库的要求较低，大致画好元件的外形并标号引脚的标号和数字即可；但是封装库要求较高，必须准确地制作得到元件的封装，否则会严重影响随后的元件焊接过程。需要注意的是两个库中的元件引脚要一一映射的关系；

第二：绘制原理图。根据具体的电路设计要求，将元件库中的元件布置在原理图上，并将各引脚相连接，同时规定好元件的标号、注释以及网络标号，尽量做到简洁美观。然后使用电器规则检查（ERC），确定没有错误后进行下一步工作。设计完成的原理图如图3.7所示。

第三：规划电路板。电路板的规划包括以下几个方面的内容。

①电路板选型：选择单面板、双面板或多面板。综合可靠性、实用性和经济性考虑，本课题选择双面板；

②确定电路板的外形，包括设置电路板形状、电路板尺寸等参数。本设计将电路板定义为矩形，大小约为9.3cm\*6.4cm；

③确定电路板与外界的接口形式，选择接插件的封装形式及确定接插件的安装位置和电路板的安装方式。  
 第四：PCB元件布局。首先在做好的原理图上生成网络表，之后在PCB图上导入网络表和元件封装库，需要强调的是在此之前必须先载入封装库。导入完成后就可以发现所有的元件都以封装的形式展示在PCB图上，通过飞线连接各元件，接着开始对器件布局。  
 需要特别注意，各元件之间一定要保持合适的距离，包括水平距离的和垂直高度，避免出现模块堆叠、开关按键或LED灯被遮挡的情况。切记将整块板规划好后再放置[元器件](http://www.baidu.com/s?wd=%E5%85%83%E5%99%A8%E4%BB%B6&hl_tag=textlink&tn=SE_hldp01350_v6v6zkg6)，否则将严重影响最后成品的质量，应该在保证上面原则的前提下，适当调整器件的位置，使板子整体看起来规范美观。  
 第五：PCB布线。整个PCB设计中最重要的工作就是布线，布线效果这将直接影响着PCB板的性能好坏。布线包括自动布线和手工布线，一般需要手动布线。根据布线规则布通之后，需要仔细对布线进行调整，使其能达到最佳的电器性能。接着是布线要整齐划一，不能纵横交错、杂乱无章。  
 第六：设计规则检查。在电路布线完成后，应当对电路板进行设计规则检查（DRC），以保证PCB图那满足符合电气规则要求和具体设计要求等。如果通过检查发现电路板上有不规范的地方，系统会报错以提示设计者进行修改。绘制完成的PCB图如图3.8所示  
 第七：制版。在此之前，最好还要有一个最终检查的过程。

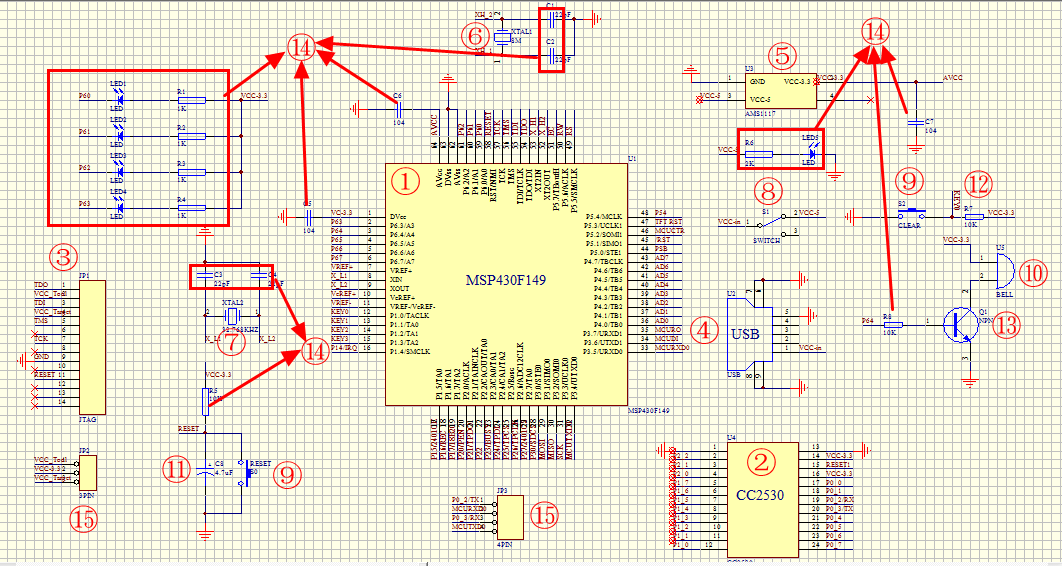


图3.7 设计完成的原理图

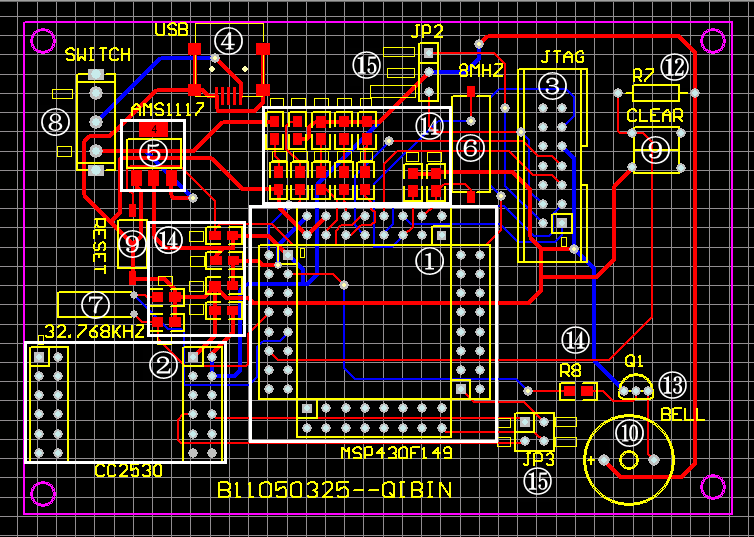


图3.8 绘制完成的PCB图

3.3.2元件焊接及各模块连线

最后一步硬件设计是焊接和各个模块的连接。完成后硬件成果如图3.9所示。

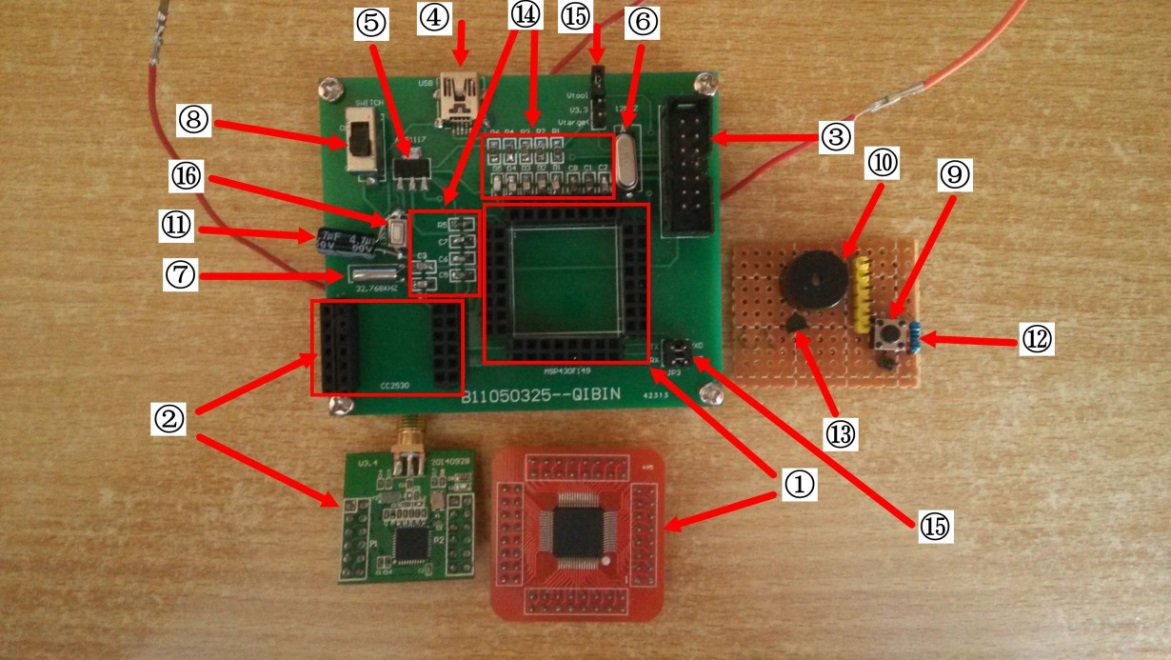


图3.9 硬件成果图

注：图3.7、3.8、3.9中数字代表的元件如下所示：

①MSP430F149单片机

②CC2530ZigBee模块

③JTAG接口

④USB接口

⑤AMS1117稳压芯片

⑥8M外部晶振

⑦32.768K外部晶振

⑧拨动开关

⑨微动开关

⑩蜂鸣器

⑪有极性电容

⑫直插电阻

⑬三极管

⑭贴片电阻电容LED

⑮排针

3.4传感器模块

传感器模块主要由光学系统、被动式热释电红外传感器、信号处理等几部分组成。其原理框图如图3.10所示，电路原理图如图3.11所示，实物图如图3.12所示。

待测

目标

菲涅耳

光学透镜

热释电

红外传感器

信号

处理

无线

发射

图3.10 传感器模块原理框图

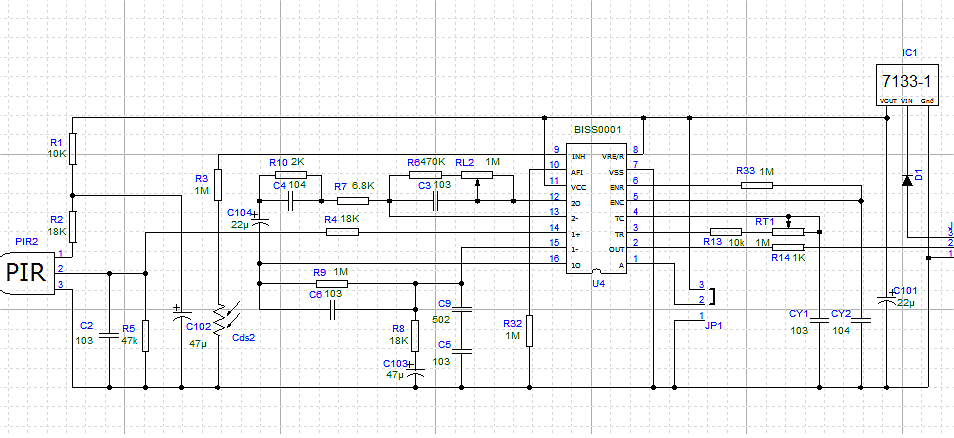


图3.11 传感器模块电路原理图

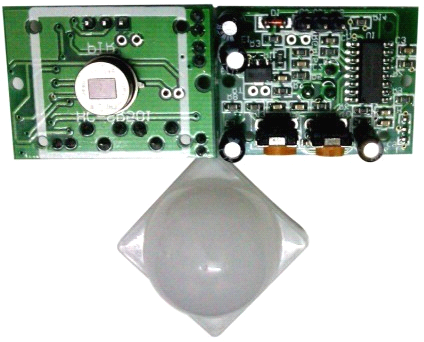


图3.12 传感器模块实物图



图3.13 ZigBee终端模块

（1）工作原理

①热释电效应

热释电效应[34]指的是在特定晶体的上、下表面设置电极，在上表面覆盖黑色膜，当红外线辐射到上表面时，会引起晶体上表面原子排列方式发生变化，从而导致电极发生变化，与下表面电极之间产生电压差△U。一般用钛酸钡、钽酸锂、硫酸三甘肽及钛铅酸铅等陶瓷氧化物和压电晶体制作此类外传感器上的晶体。

②热释电红外传感器

本系统采用的传感器是被动式热释电红外传感器，传感器本身不发出任何形式能量，由传感器直接探测来自移动目标的红外辐射[34]，因而有“被动”之称。人体正常体温一般稳定在37摄氏度，所以会发出10μm左右波长的红外线，而红外传感器的探测波长范围是8~14μm，因此能较好地探测到活动的人体。

热释电红外传感器是传感器模块中的最为重要的器件，它的结构图如图3.14所示。它的工作是把人体的红外信号转换为电信号以供信号处理部分使用，工作原理为感应探头呈长方形，两个感应元（A元、B元）位于探头窗口较长两端。当人体经过传感器时，红外光谱到达双元的时间、距离有差值，再根据热释电效应产生电信号。图3.15所示的是将待测目标、菲涅耳透镜[34]、热释电红外传感器相结合使用时的工作原理示意图。

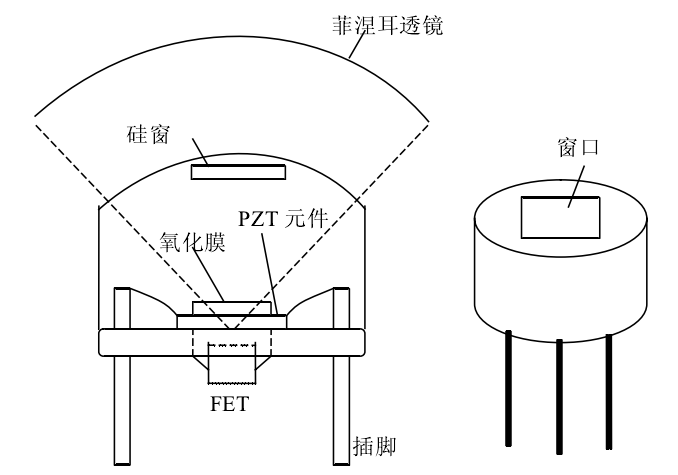


图3.14 热释电红外传感器结构图

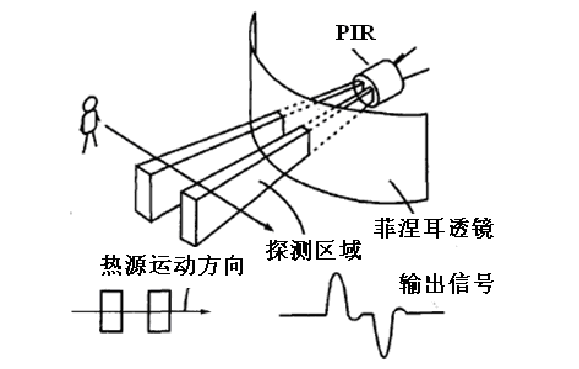


图3.15 人体通过传感器时产生的信号

③信号处理

传感器模块其它部分电路负责将传感器输出的微弱电信号进行放大、滤波、延迟、比较[35]等一系列处理，得到输出的入侵信号。

（2）感应范围

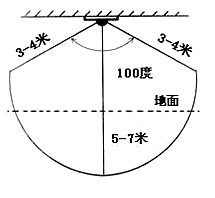
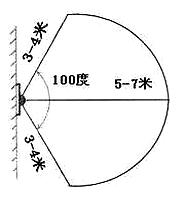


图3.16 传感器模块感应范围

传感器模块的感应范围如上图3.16所示，可以通过调节距离调节电位器改变，顺时针旋转感应距离增大（最远约为7m），逆时针旋转感应距离减小（最近约为3m）。

（3）注意事项

①传感器模块通电后会初始化一分钟左右的时间，在初始化期间模块会间断地输出几次信号，之后进入待机状态；

②应尽量避免光照近距离照射传感器模块表面的透镜，以免接收到干扰信号产生误动作；同时尽量避免气流对传感器造成干扰；

③当人体从水平方向走过时，红外光谱到达双元的时间、距离差值大[34]，感应灵敏；当人体从垂直方向走过时，红外光谱到达双元的时间、距离差值较小，甚至没有差值，因此传感器感应不灵敏或不工作。所以安装感应器时应使根据实际情况，使探头双元的方向与人体活动的方向尽量相平行，保证人体经过时先后被探头双元所感应 。

第四章 软件设计

4.1软件设计总体方案

ZigBee终端节点不断地向协调器节点发送当前监测情况，协调器接收到数据后通过串口发送给430控制端，430控制端对数据进行处理数，包括判断应该点亮哪个LED以及是否报警。若产生报警信号，则可以通过消除报警按键关闭报警。

4.2控制模块编程

4.2.1初始化模块

（1）串口初始化

串口初始化[37]通过对U0CTL、U0TCTL、U0BR、ME1、IE1、P3SEL、P3DIR这几个寄存器进行置位或复位操作，来配置串口相关参数。代码如下：

void UART\_Init()

{

U0CTL |= SWRST; //复位SWRST

U0CTL |= CHAR; //8位数据模式

U0TCTL |= SSEL1; //SMCLK为串口时钟

U0BR1 = baud\_h; //baud\_h和baud\_l定义在Config.h头文件中

U0BR0 = baud\_l;

U0MCTL = 0x00; //微调寄存器为0，波特率115200bps

ME1 |= UTXE0; //UART0发送使能

ME1 |= URXE0; //UART0接收使能

U0CTL &= ~SWRST; //初始化SWRST

IE1 |= URXIE0; //接收中断使能位

P3SEL &=~ 0x30; //设置IO口为普通I/O模式

P3DIR |= BIT4; //设置TXD0口方向为输出

}

串口初始化的步骤为：

①使能USART控制寄存器U0CTL的第0位SWRST，用于软件复位；将第4位CHAR置1，用于设置字符数据长度为8位；

②设置USART发送控制寄存器U0TCTL的第4、5位SSEL0为01，选择BRCLK的时钟源为SMCLK；

③MSP430的波特率计算公式4-1如下所示。

式（4-1）

BRCLK表示MSP 430串口时钟信号（一般由 8MHz外部晶振提供），分频因子N=UxBR+UMCTL/8，其中UxBR为由波特率控制寄存器UxBR中的16位数据值，用于调整整数部分，它由UxBR1 和UxBR0 组成，而UMCTL为调制寄存器UxMTCL 中的8位数据值，用于微调小数部分。在本系统中，我们设置串口通信的波特率为115200bps。

首先设置波特率控制寄存器U0BR，包括U0BR0和U0BR1为baud\_h和baud\_l。

这两个量定义在Config.h头文件中，代码如下：

//串口波特率计算

#define baud 115200 //设置波特率的大小

#define baud\_setting (uint)((ulong)CPU\_F/((ulong)baud)) //波特率计算公式

#define baud\_h (uchar)(baud\_setting>>8) //提取高位

#define baud\_l (uchar)(baud\_setting) //低位

根据，现令baud=115200，计算出baud\_setting，然后将它的高、低8位分别赋值给baud\_h和baud\_l。

微调寄存器U0MCTL用于波特率的微调，本设计不需要微调，因此设置为0；

④置位模块使能寄存器ME1的第6、7位，将USART0的接收和发送模块使能；

⑤置位中断使能寄存器IE1的第6、7位，将USART0的发送和接收中断使能；

⑥因为MSP430串口通信的UART模式使用P3的第4、5位UTXD0和URXD0，所以端口功能选择寄存器P3SEL的第4、5位都设置为0，即都设置为普通I/O，然后端口方向寄存器P3DIR的第4、5位分别设置为1和0，即UTXD0设置为输出，URXD0设置为输入。

（2）端口初始化

端口初始化[37]通过对寄存器 P6SEL、P6DIR、P6OUT、P1SEL、P1DIR 进行配置，从而控制LED、蜂鸣器和按键的输入输出。

void Port\_Init()

{

P6SEL = 0x00; //P6口普通IO功能

P6DIR = 0xFF; //设置IO口方向为输出

P6OUT = 0xEF; //初始设置为关

P1SEL = 0x00; //P1口普通IO功能

P1DIR = 0xF0; //设置IO口方向为输入

}

通过设置P6的端口功能选择寄存器P6SEL的所有位为0，使它的所有端口都设置为普通I/O，再设置端口方向寄存器P6DIR所有位为1，使其所有端口都设置为输出，同时通过设置输出寄存器P6OUT全为1，全部初始化输出1。其中P6.0~P6.2作为入侵信号显示LED的控制，P6.3控制报警LED，P6.4控制报警蜂鸣器。最后同样设置P1.0为普通I/O，方向需要设置为输入，用作按键清除报警信号的输入。

（3）看门狗和系统时钟初始化

这两个函数定义在Config.h头文件中，为系统默认配置。

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

// 系统时钟初始化，外部8M晶振

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

void Clock\_Init()

{

uchar i;

BCSCTL1&=~XT2OFF; //打开XT2振荡器

BCSCTL2|=SELM1+SELS; //MCLK为8MHZ，SMCLK为8MHZ

do{

IFG1&=~OFIFG; //清楚振荡器错误标志

for(i=0;i<100;i++)

\_NOP();

}

while((IFG1&OFIFG)!=0); //如果标志位1，则继续循环等待

IFG1&=~OFIFG;

}

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

// MSP430内部看门狗初始化

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

void WDT\_Init()

{

WDTCTL = WDTPW + WDTHOLD; //关闭看门狗

}

4.2.2中断模块

系统中断[37]为串口0接收中断，中断向量为#pragma vector = UART0RX\_ VECTOR。当串口接收到数据后，系统进入该中断。

#pragma vector=UART0RX\_VECTOR

\_\_interrupt void UART0\_RX\_ISR(void)

{

count++; //count自加1计数

data = U0RXBUF; //接收到的数据存放在data中

}

该中断的主要工作为存储串口接收到的数据到data中，同时计数全局变量count。

4.2.3主程序模块

（1）流程图



图4.1 主程序流程图

（2）程序运行步骤

void main(void)

{

WDT\_Init(); //看门狗设置

Clock\_Init(); //系统时钟设置

Port\_Init(); //LED端口初始化

UART\_Init(); //串口设置初始化

\_EINT(); //开中断

while(1) //无限循环

{

switch(data)

{

//接收到‘1’则点亮相应的LED

case '1':P6OUT &= ~(1<<(count%3));break;

//接收到‘0’则熄灭相应的LED

case '0':P6OUT |= (1<<(count%3));break;

}

//若三个LED中有任意一个被点亮，则点亮报警LED和蜂鸣器

if(((P6OUT & 0x01)==0)||((P6OUT & 0x02)==0)||((P6OUT & 0x04)==0))

{

P6OUT &= ~BIT3;

P6OUT |= BIT4;

}

//若报警LED被点亮的同时按下消除报警按键

if(((P1IN & BIT0)==0)&&((P6OUT & BIT3)==0))

{

delay\_ms(20); //延时20ms消除抖动

P6OUT |= BIT3; //熄灭报警LED和蜂鸣器

P6OUT &= ~BIT4;

}

else

P6OUT &= 0xFF; //否则报警LED和蜂鸣器保持不变

}

}

首先进行初始化，详见4.2.1，然后开总中断。若系统产生串口中断，则进入中断服务程序，详见4.2.2。退出中断后判断接受到的数据为‘1’还是‘0’，分别对对应的入侵信号显示LED进行开关操作。具体方法如下：

由于3个终端的数据是连续发送的，例如“110 001 100....”，所以在主程序中将count除以3，然后取余就可以将对应的入侵显示LED点亮。当count=时，取余得0，那么将P6OUT第0位设置为0，则点亮第1个LED；当count=1时，取余得1，那么将P6OUT第1位设置为0，则点亮第2个LED，其它情况以此类推。

若三个LED中任意一个被点亮，则产生报警信号，点亮报警LED和蜂鸣器。若之后按下消除报警按键，使P1的输入寄存器P1IN第1位为0，则关闭报警。随后程序进入无限循环。具体程序详见附录A。

4.3 ZigBee模块编程

由于Z-stack中大部分函数都被封装在库代码中，用户在进行具体的应用开发时只能够通过调用API[38]来进行，而无法知道ZigBee协议栈的具体细节。因此，用户的开发设计大多建立在应用层，即Z-Stack协议栈[38]的APP层。更为具体说，就是修改Sample Application.c文件中的各个函数，其它函数及代码基本上不必考虑。

4.3.1初始化模块

在ZigBee红外入侵智能射频芯片 CC2530加电启动之后，首先执行初始化，初始化内容如下图4.2所示:



图4.2 初始化模块图

在Sample Application.c文件中，调用任务初始化函数SampleApp\_Init()，主要处理以下几个内容：

1. 赋予任务ID号；
2. 设置寻址方式；
3. 端点描述符的初始化；
4. 调用函数afRegister()在AF层注册端点；

（5）热释电红外线传感器IO口初始化。

由于传感器与芯片的P0.4引脚相连，（5）对应的代码为：

P0SEL &=~0x10; //设置P0.4为普通IO

P0DIR &=~0x10; //P0.4定义为输入

其余代码详解见附录B。

在Z-main.c文件的函数中，调用了函数osal\_init\_system()对Z-Stack进行初始化，在函数osal\_init\_system()中调用了函数osalInitTask()[38]对任务进行初始化，在函数osalInitTask()中调用ZDApp\_Init()函数对ZDO层进行初始化，ZDO层是设备启动应该主要关注的一层。

图4.3~图4.4是协调器和终端设备从函数ZDOInitDevice(0)开始的启动过程。



图4.3 协调器启动过程



图4.4 终端启动过程

其中，devState为设备状态结构变量，ZDO为Z-stack的ZigBee设备对象层，NWK为Z-stack的网络层[38]。DEV\_INIT：初始化，DEV\_COOR\_STARTING：作为协调器启动，DEV\_ZB\_COORD：作为协调器启动，DEV\_NWK\_DISC：发现网络，DEV\_NWK\_JOINING：加入网络，DEV\_END\_DEVICE：终端加入。

4.3.2事件处理函数

SampleApp\_ProcessEvent()函数是对应用户任务的事件处理函数[38]。当应用层接收到消息时，先判断消息类型，分为两类：意识系统消息事件，二是用户自定义事件。

系统消息事件包括以下几类：

1. 按键事件KEY\_CHANGE；
2. 接收消息事件AF\_INCOMING\_MSG\_CMD；
3. 设备状态改变事件ZDO\_STATE\_CHANGE。

具体代码见附录B。

ZigBee设备通过不断触发不同的事件调用相应的函数，从而达到收发报警信息的目的，以下将从协调器和终端两种设备出发，分别介绍具体事件的调用过程以及作用。

4.3.3终端工作过程

当终端设备成功启动后，便开始不断接收传感器发送的入侵信号，从而触发事件ZDO\_STATE\_CHANGE[38]，事件处理如下：

case ZDO\_STATE\_CHANGE:

SampleApp\_NwkState = (devStates\_t)(MSGpkt->hdr.status);

if ( //(SampleApp\_NwkState == DEV\_ZB\_COORD)

|| (SampleApp\_NwkState == DEV\_ROUTER)

|| (SampleApp\_NwkState == DEV\_END\_DEVICE) )

{

//启用定时器，打开定时事件

osal\_start\_timerEx( SampleApp\_TaskID,

SAMPLEAPP\_SEND\_PERIODIC\_MSG\_EVT,

SAMPLEAPP\_SEND\_PERIODIC\_MSG\_TIMEOUT );

}

else

{

//设备不在网络中

}

break;

如果设备的网络状态为DEV\_ZB\_COORD、DEV\_ROUTER或者DEV\_END\_ DEVICE表明设备启动成功，则调用函数osal\_start\_timerEx()定时触发事件SAMPLEAPP \_SEND\_ PERIODIC\_ MSG\_ EVT。该事件的任务ID为Sample App\_TaskID，即该事件还是由SampleApp\_ ProcessEvent()函数[38]进行处理。定时长度为SAMPLEAPP \_SEND\_ PERIODIC \_ MSG\_TIMEOUT（定义在SampleApp.h头文件中，值为1s）。SAMPLEAPP \_SEND\_ PERIODIC\_ MSG\_ EVT的处理代码如下：

if ( events & SAMPLEAPP\_SEND\_PERIODIC\_MSG\_EVT )

{

//将协议栈中自带的发送周期信息函数替换为发送自定义信息函数

// SampleApp\_SendPeriodicMessage();

SampleApp\_Send\_P2P\_Message();

//定时再次触发事件SAMPLEAPP \_SEND\_ PERIODIC\_ MSG\_ EVT

osal\_start\_timerEx( SampleApp\_TaskID,

SAMPLEAPP\_SEND\_PERIODIC\_MSG\_EVT,

(SAMPLEAPP\_SEND\_PERIODIC\_MSG\_TIMEOUT

+(osal\_rand() & 0x00FF)) );

//返回没有处理完的事件

return (events ^ SAMPLEAPP\_SEND\_PERIODIC\_MSG\_EVT);

}

在处理该事件时，协议栈调用了事件处理函数SampleApp\_ Send\_ P2P\_ Message()，其代码如下：

void SampleApp\_Send\_P2P\_Message( void )

{

byte state;

//DATA\_PIN定义在Config.h头文件中，为传感器接入引脚P0.4

if(DATA\_PIN == 1)

{

MicroWait (10000); //等待10ms，函数定义在OnBoard.h头文件中

if(DATA\_PIN == 1)

{

state = 0x31; //有人进入，state为’1’

HalUARTWrite(0,"insecurity\n", 10); //串口显示

HalLcdWriteString( "insecurity", HAL\_LCD\_LINE\_3 ); //LCD显示

}

}

else

{

state = 0x30; //无人进入，state为’0’

HalUARTWrite(0,"security\n", 9); //串口显示

HalLcdWriteString( "security", HAL\_LCD\_LINE\_3 ); //LCD显示

}

if ( AF\_DataRequest( //发送目的地址和端点描述符

&SampleApp\_P2P\_DstAddr, &SampleApp\_epDesc,

//簇ID

SAMPLEAPP\_P2P\_CLUSTERID,

//发送的字节长度

1,

//发送的数据，为上述变量state

&state,

//发送的数据ID

&SampleApp\_TransID,

//设置路由发现

AF\_DISCV\_ROUTE,

//设置路由域

AF\_DEFAULT\_RADIUS ) == afStatus\_SUCCESS )

{

}

else

{

//请求发送错误

}

}

再利用定时器周期性间隔1秒定时采集热释电红外传感器监测到的入侵信号，以点播方式将采集到的信息发送给协调器。

在该函数中调用了数据发送函数AF\_DataRequest()发送数据。参数中地址为Sample App\_ P2P\_DstAddr在SampleApp\_Init()中被初始化，簇ID为SAMPLEAPP \_P2P\_ CLUSTERID。

4.3.4协调器工作过程

协调器的主要功能[38]是协调建立网络，其他功能还包括：传输网络信标、管理网络节点及存储网络节点信息，并且提供关联节点之间的路由信息；此外，网络协调器要存储一些基本信息，如节点数据设备、数据转发表及设备关联表等。

当协调器接收到终端发送的消息后会触发事件AF\_INCOMING\_MSG\_CMD进行处理，事件的处理如下：

case AF\_INCOMING\_MSG\_CMD:

//调用接收消息处理函数

SampleApp\_MessageMSGCB( MSGpkt );

break;

在处理该事件时调用了事件处理函数SampleApp\_MessageMSGCB()进行处理。该函数如下：

void SampleApp\_MessageMSGCB( afIncomingMSGPacket\_t \*pkt )

{

uint16 flashTime;

switch ( pkt->clusterId ) //提取接收信息的簇ID

{

case SAMPLEAPP\_P2P\_CLUSTERID: //自定义信息簇ID

//输出接收到的数据

HalUARTWrite(0, pkt->cmd.Data, pkt->cmd.DataLength);

break;

case SAMPLEAPP\_PERIODIC\_CLUSTERID: //协议栈周期信息簇ID

break;

case SAMPLEAPP\_FLASH\_CLUSTERID: //协议栈闪烁信息簇ID

break;

}

}

在该函数中根据簇ID的不同进行处理。在本设计中，除了自定义信息簇ID外，没有对其它簇ID进行任何处理。当接收到自定义信息簇ID时，函数调用了HalUARTWrite()函数[37]将接收到的数据通过串口发送给控制模块MSP430。

CC2530的波特率计算公式4-2如下所示。

式（4-2）

式中f是系统时钟频率，等于16MHzRC振荡器或者32MHz晶体振荡器，本系统选择32MHz晶振作为系统时钟源。因为MSP430和CC2530芯片的串口波特率必须相同才可以通信，所以设置CC2530的波特率也为115200bps。MT层的MT\_UartInit()函数[37]可以设置波特率，具体代码为uartConfig.baudrate=MT\_ UART\_ DEFAULT\_BAUDRATE，通过更改MT\_ UART\_ DEFAULT\_BAUDRATE变量的数值来设置波特率。

第五章 调试及性能分析

5.1调试软件介绍

本设计的两个芯片都使用IAR System公司的调试软件IAR Embedded Workbench for MSP430 5.30和IAR Embedded Workbench for 8051 8.10。

IAR EW开发工具操作简单、功能丰富，该集成开发环境包括IAR的C/C++优化编译器、汇编工具、链接器、库管理器、文本编辑器、工程管理器。最大的特点是可以根据工程配置的不同，针对每种芯片生成具有特性的可编译文件。使用IAR EW，可以有效地提高用户的编程效率，程序稳定可靠，便于移植。

5.2 MSP430工程配置和调试

5.2.1工程配置

（1）点击界面左侧窗口的项目名称，选中后点击右键，选择“option”选项，弹

出对属性设置对话框；

（2）在“Gneral Options”选项“Target”标签下的“Device”栏中选择器件类型为MSP430F149；

（3）在“Linker”选项“Output”标签下勾选“Allow C-SPY-specific extra output file”；

（4）在“Extra Output”标签中设置输出txt类型目标文件；

（5）在“Debugger”选项“Setup”标签下选择仿真器，本人有硬件仿真器，所以选择“FET Debugger”，

（6）如果选择了硬件仿真器，可以在“FET Debugger”中选择对应的工具类型，根据所拥有的仿真器自行选择，本人的仿真器为“Texas Instrument USB-IF”。

5.2.2软件编译和调试

配置完成后，点击菜单栏“Project”下“Make”项，也可以在快捷图标栏点击QQ截图20150525170828，即可编译并输出目标文件。

编译后如下图5.7所示。



图5.7 编译完成后界面

如果有错误提示，需根据错误提示更改然后重新编译。

编译无误后，将仿真器与电脑和JTAG口相连，点击菜单栏“Project”下“Download and Debug”项,也可以在快捷图标栏点击QQ截图20150525170714，即可下载和调试程序。

5.3 ZigBee模块工程配置和调试

5.3.1工程配置

（1）点击界面左侧窗口的项目名称，选中后点击右键，选择“option”选项，弹

出属性设置对话框；

（2）在“General Option”选项“Target”标签下的“Device”栏中选择“Texas Instruments”文件夹下的“CC2530F256.i51”；

（3）设置“Codemodel”为“Banked”、“Data model”为“Large”、“Calling convention”为“XDATA stack reentrant”；

（4）在“Stack/Heap”标签下“Stack sizes”栏中的“IDATA”、“IDATA”、“IDATA”分别为“0x40”、“0x80”、“0x1FF”；

（5）在“Linker”选项“Config”标签下勾选“Override default”，点击下面对话框最右边的按键，选择“lnk51ew\_cc2530F256\_banked.xcl”；

（6）在“Linker”选项“Output”标签下勾选“Allow C-SPY-specific extra output file”，如下图5.13所示；

（7）在“Extra Output”标签中设置输出hex类型目标文件；

（8）在“Debugger”栏中“Setup”标签下选择仿真器为“Texas Instruments”。

5.3.2编译和调试

ZigBee模块的编译和调试与MSP430模块相同，这里不再赘述。

需要注意的是，在下载程序时，协调器和终端对应不同的编译选项，分别为CoordinatorEB-Pro和EndDeviceEB-Pro，两者的区别是功能的全面与否，需要选择对应的编译选项下载程序。

一般无法调试都是由于调试软件没有与仿真器正确连接，主要问题就是上述调试配置不正确。另外需要注意的是，ZigBee模块仿真之前必须按下仿真器上的复位按键，否则也会出现连接错误。

5.4性能分析

5.4.1调试过程中的问题

（1）传感器感应范围

如第三章硬件设计传感器模块部分所述，调节传感器模块上的距离调节电位器可以改变感应范围。在调试过程中，我发现若感应范围过大，会导致传感器检测不灵敏的现象，甚至检测失效。这是由于热释电红外传感器的工作原理[38]所致的，详见传感器模块部分。当感应范围过大时，人体红外光谱达到双元的时间、距离的差值相对单个元的数值来说太小，则容易被忽略，从而导致检测不灵敏。简单示意图如下图5.16所示。

A元

B元

人体

lb

la

A元

B元

人体

Lb

La

图5.16 简单示意图

从图中，我们很明显可以看出，远距离情况下A、B元数值的差值与A元数值的比值要远小于近距离的情况，即，这就是感应范围过大导致传感器检测失效的主要原因。因此，选择适当的感应范围[40]关系到整个报警系统的功能好坏。结合实际情况，最终选择3m作为感应范围。

（2）传感器摆放位置

之后是传感器的摆放问题，由于感应范围是扇形，本人认为将监测区域设定为三角形可以充分利用传感器的性能[40]，覆盖较大的区域。具体摆放方式如下图5.17所示。

图5.17 3传感器摆放图

但在实际的测试中，我发现，由于传感器的感应角度较大（100°左右），使用现有的3个传感器无法组成一个形状较为完整的监控区域，若条件允许，使用4个传感器的话，应该可以组成一个正方形，如下图5.18所示。

图5.18 4传感器摆放图

（3）监控效果

在报警器网络工作之前，如果有人位于监控范围内，开机组网之后，报警器并不会报警。这也是由于热释电传感器的工作原理所致，它对静态物体的检测不够敏感，只有当人体在传感器的检测范围内移动时，红外线到达双元的距离和时间不断动态变化，使得传感器上下表面的电压差也随之改变，这样才能达到理想的监控效果。在实际需要报警监控的应用情况下，如果不法分子事先在监控区域内蹲点，而传感器却无法准确地报警，将会给个人乃至社会造成不可估量的灾难。

这也给我的毕业设计留下了一项改进计划，就是如何对于静止的物体也能够准确地检测到入侵信号，实现动静结合的良好效果。

5.4.2低功耗分析

从上文我们可以得知，MSP430和CC2530都具有超低功耗的特点，下面具体计算它们的功耗。

MSP430芯片在正常工作模式下的电流[39]是200μA，在最低功耗LMP4模式下电流是0.1μA，基本上可以忽略。由于单片机的工作电压为3.3V，计算出功耗为。一节普通5号电池的电压为1.5V，容量为2000mAH，总电量为。两者相除，约得4545h≈190天。由上述计算可以得出，两节5号电池大概可以使MSP430芯片持续工作超过1年。

CC2530芯片在正常工作模式下的电流[40]是0.23mA，在最低功耗下的电流是0.4μA.由于工作电压为3.3V，计算出功耗为。上文计算出一节普通 5号电池的电量为3wh。两者相除，约得3947h≈165天。由上述计算可以得出，两节5号电池大概可以使CC2530芯片持续工作将近1年。

5.4.3实验结果

（1）ZigBee设备组网

本设计共使用了4个ZigBee模块，共同组成一个星型网络。其中1个作为协调器，协调器的核心板与PCB板相连接，并以UART模式与MSP430单片机进行串口通信，还有3个ZigBee模块作为终端节点，分别监控相应的区域。由于ZigBee是自组网形式，所以它的组网非常简单，只需要将同一个协议栈的的程序通过不同的编译选项下载到协调器和终端节点中，然后通过ZigBee模块底板上的3个LED灯来判断是否组网成功。将终端节点通电后，如果协调器没有通电，则LED3会一直闪烁；直到协调器通电后，协调器和终端节点上的LED3先后熄灭，则说明两者已经完成组网。若需要将新的节点加入网络，只要把相应的程序下载到终端节点并通电，就可以实现自动组网。组网成功后，3个终端节点会周期性地发送监测信号到协调器，协调器会将数据处理好发送到MSP430单片机。

（2）报警系统具体工作流程：

将连接了传感器的3个终端摆放好，使它们的探测区域能够覆盖整个监测区域，传感器探头双元的方向与人体走动的方向尽量相平行，保证人体经过时先后被探头双元所感应。终端摆放好以后，与PCB板一同供电，三个终端自动与协调器组网；组网后，在控制端上观察传感器监测情况。当有人从监测区域通过后，监测区域对应的入侵显示LED被点亮，随后打开报警LED和蜂鸣器，等待操作；

如果按下消除报警按键，则关闭LED和蜂鸣器。

结束语

1.总结

本系统采用MSP430F149芯片作为控制芯片，ZigBeeCC2530作为信号处理和传输节点，被动式热释电红外传感器作为监控模块收集入侵信号，实现了入侵报警功能，同时具备低功耗、易组网等特点。

在做毕业设计的过程中，我遇到过许多困难。最开始做硬件的时候，由于之前没有经验，元件的封装设计得不符合要求，模块的摆放位置也不合理，导致焊接的时候很多元件无法插到PCB板上。最后进过多次努力，虽然板子不是很美观，但总算是焊接完成了。随后的软件编写过程也是充满曲折，先是ZigBee模块无法通信，原来是没有安装天线所致；接着是430控制程序无法下载，原因是仿真器驱动没有装好；最后是报警功能无法实现，经过不断的修改程序，终于是完成了设计要求。回想起这整个过程，我感触颇多。

（1）这次毕设使我认识到自己的相关专业知识掌握得还不够牢固，知识体系不太全面，主要是以前课堂上学习到的内容没有深入地理解，不够透彻。因此，我会在日后的工作、生活中不断充电，活到老，学到老；

（2）毕设不仅检测学生的书本知识，更是对学生的分析问题解决问题、动手能力最开始的时候觉得题目很难，无从下手，在自己的努力和老师同学的帮助下，我对题目越来越理解，思路也逐渐清晰，个人的能力得到了极大的提高；

（3）在这次毕业设计中，我体会到了同学之间深厚的感情。每当我遇到挫折，都会有热心的同学帮助我解决问题，度过难关；或者是给我提出一些宝贵的建议，激发我对毕设的不断完善和改进，所以在这里非常感谢帮助他们。

2.展望

关于本系统，我认为有下几个方面可以进行改进：

1. 要提高PCB板的设计水平，避免布线杂乱、封装不精确的现象。同时要提高自己的焊接工艺，尽量将元件焊接整齐，把握好焊锡的量，不多焊、虚焊；
2. 系统可以加入更多可定制的功能，可以根据实际的情况调整报警器的各项参数指标，接着完善系统的稳定性，提高程序的运行速度；
3. 供电方式从USB供电改为纽扣电池或干电池供电，使节点更加便于携带和移动，另一方面可以突出低功耗的特点；
4. 更高效、安全的路由算法，路由算法的优化是ZigBee乃至所有无线通讯技术着重研究的方向，一旦实现将对无线通信产生巨大的影响。

致 谢

这次毕业设计能够顺利完成离不开老师和同学们的帮助，是他们的支持和鼓励让我不断前进，坚持不懈。在毕设即将结束之际，我要对给予我大量关心和督促的王冬生老师表示衷心的谢意。王老师工作认真、治学严谨、学富五车，他的学术研究精神深深地感染和激励了我。此外，王老师人格高尚，待人和善，在毕设期间不仅教授了我相关学术知识和科研方法，还教会了我许多做人的道理，这些都将使我终生受益。当然，在[论文](http://biyelunwen.yjbys.com/)的写作过程中，王老师也给了我许多意见和建议，帮助我不断完善论文，提高质量。

同时，我也要向身边的同学表示感谢，当我遇到困难时，他们给予了我很大的帮助和鼓励，使最终我能走到这一步。

最后要感谢的是大学四年来所有的任课老师，没有他们的辛勤劳动、谆谆教导就没有我今天的成果。

在这即将离开学校、进入社会之际，我祝愿所有曾经关心、帮助和支持过我的人都能拥有一个绚烂缤纷的美好明天。

参考文献

[1] 孙利民, 李建中, 陈渝等. 无线传感器网络[M]. 北京:清华大学出版社, 2005: 109-132.

[2] 郑长风, 程光伟, 郭军. 一种基于红外探测转无线报警监控系统[J]. 现代电子技术, 2003,

(4):84-86.

[3] 张春红等. 物联网技术与应用[M]. 北京:人民邮电出版社, 2011: 35-42.

[4] 李晓延. 浅谈无线传感器网络[J]. 今日电子, 2006, (9): 57-59.

[5] 张少军. 红外入侵智能传感器技术及应用[M]. 北京:中国电力出版社, 2010: 63-70.

[6] 杨靖, 洪露, 李捍东. 基于WSN的家居安防系统中防入侵探测节点的设计[J]. 计算机测量

技术, 2012, 20(7): 348-352.

[7] 王芳, 马幼军, 蒋国平. 智能化住宅防盗防火报警系统设计[J]. 传感器技术, 2002, 21(10):

25-33.

[8] 阎子勤, 刘一江. 周边红外防盗报警系统[J]. 安防科技, 2009, (7): 45-48.

[9] 陈梦阳, 田杰, 罗明玉等. 用于周界入侵报警的红外超声波无线传感器网络节点设计[J]. 传感器与微系统, 2014, 33(7): 25-31.

[10] 杨毅, 陈超, 潘涛, 王先春. 新型多功能智能报警器的研究与开发[J]. 科技风, 2010, (19):

206-211.

[11] 朱惜辰. 红外探测器的进展[J]. 光子学报, 2001, (10): 30-34.

[12] 姜仲, 刘丹. ZigBee技术与实训教程——基于CC2530的无线传感器技术[M]. 北京:清华

大学出版社, 2014: 1-9.

[13] 吕立波. 浅析防盗入侵报警系统及其应用[J]. 中国公共安全(学术版), 2013, ( 13):

172-175.

[14] 李文仲, 段朝玉. 红外入侵智能单片机技术丛书——Zigbee红外入侵智能技术入门与实

战[M]. 北京:北京航空航天大学出版社, 2008: 178-190.

[15] 王权平, 王莉. ZigBee技术及其应用[J]. 现代电信科技, 2004, (1): 33-37.

[16] 韩华峰, 杜克明, 孙忠富, 赵伟, 陈冉梁. 基于ZigBee网络的温室环境远程监控系统设计

与应用[J]. 农业工程学报, 2009, (7): 267-272.

[17] 董亚超. 基于ZigBee技术的无线环境监测网络的开发[D]. 大连理工大学, 2008: 3-9.

[18] 王东, 张金荣, 魏延. 利用ZigBee技术构建无线传感器网络[J]. 重庆大学学报(自然科学

版), 2006, 29(8): 26-31.

[19] 赵芸, 张浩, 彭道刚. ZigBee无线网络技术的应用[J]. 机电一体化, 2007, (4): 34-38.

[20] 李文仲. Zigbee技术与应用——CC2530设计、开发与实践[M]. 北京:国防工业出版社,

2010: 35-50.

[21] Axel Sikora. Coexistence of IEEE802.15.4 (ZigBee) with IEEE802.11 (WLAN),Bluetooth,and

Microwave Ovens in 2.4 GHz ISM Band[M]. 2004, (3): 16-27.

[22] Han D M, Lim J H.Smart Home Energy Management System using IEEE 802.15.4 and

ZigBee[J]. IEEE Transactions on Consumer Electronics, 2010, 56(3): 1403-1410.

[23] 童丽丽. 基于ZigBee技术的协议开发与平台设计[D]. 沈阳工业大学, 2008: 14-18.

[24] 李文仲等. Zigbee2007红外入侵智能与红外入侵智能定位实战[M]. 北京:北京航空航天大

学出版社, 2008: 157-170.

[25] 金纯, 罗祖秋, 罗凤. Zigbee技术基础及案例分析[M]. 北京:国防工业出版社, 2008:

247-265.

[26] 陈以华. 基于ZigBee无线网络的在线监测系统的设计[J]. 中国新技术新产品, 2012, (8):

12-12.

[27] 广州致远电子有限公司. ZigBee技术实用开发参考指南[J]. 电子技术应用, 2011, 37(4):

106-110.

[28] Larranaga J, Muguira L, Lopez-Garde J -, et al.An environment adaptive ZigBee-based indoor

positioning algorithm[C]. International Conference on Indoor Positioning and Indoor

Navigation. IEEE, 2010: 1-8.

[29] 姜进, 王鑫, 杨慧中. 基于MSP430单片机的无线环境监测系统的设计[J]. 江南大学学报

(自然科学版), 2011, 32(1): 45-48.

[30] 青岛东合技术有限公司. ZigBee开发技术及实践[M]. 西安:西安电子科技大学出版社,

2014: 14-32.

[31] ZigBee Alliance. ZigBee Document 053474r06, Version 1.0[R]. 2004.

[32] Liang X U, Diao X M, Zhou H J, et al. Design of wireless environment monitoring sensor

system based on MSP430F149[J]. International Electronic Elements, 2006, (7): 19-25.

[33] 高守玮, 吴灿阳. Zigbee技术实践教程：基于CC2530的红外入侵智能传感器解决方案(附

盘)[M][CD]. 北京:北京航空航天大学出版社, 2009: 197-204.

[34] 刘舒棋, 施国梁. 基于热释电红外传感器的报警系统[J]. 国外电子元器件, 2005, (3):

157-161.

[35] 胡伟生, 方佩敏. 菲涅耳透镜与外壳[J]. 电子世界, 2004, (10): 47-48.

[36] 王青林, 张伟, 赵景波. 电路设计与制版——Protel 99SE基础教程(修订版)[M]. 北京:

人民邮电出版社, 2012: 237-250.

[37] 唐继贤, 杨扬. MSP430超低功耗16位单片机开发实例[M]. 北京:北京航空航天大学出版

社, 2014: 116-120.

[38] LAN/MAN Standards Committee of the IEEE Computer Society,Part 15.4.Wireless Medium

Access Control (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless

Personal Area Networks (LR-WPANs)[J]. IEEE Communications Magazine, 2002, 40(8):

70-77.

[39] 宋红霞. ZigBee无线网络通信系统开发与节能技术研究[D]. 北方工业大学, 2014: 37-46.

[40] Kartal B, Sokullu R, Suihko T. A new multicast routing algorithm for ZigBee networks[J].

Winsys, 2007: 16-20.

附录A

**控制端源程序**

#include <MSP430x14x.h>

#include "Config.h"

int count=-1; //定义全局变量count用于控制显示入侵信号LED

uchar data=0; //定义字符data存放串口接收到的数据

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

// MSP430串口初始化

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

void UART\_Init()

{

U0CTL |= SWRST; //复位SWRST

U0CTL |= CHAR; //8位数据模式

U0TCTL |= SSEL1; //SMCLK为串口时钟

U0BR1 = baud\_h; //baud\_h和baud\_l定义在Config.h头文件中

U0BR0 = baud\_l;

U0MCTL = 0x00; //微调寄存器为0，波特率115200bps

ME1 |= UTXE0; //UART0发送接收使能

ME1 |= URXE0; //UART0发送接收使能

U0CTL &= ~SWRST; //初始化SWRST

IE1 |= URXIE0; //接收中断使能位

P3SEL |= 0x30; //设置IO口为普通I/O模式

P3DIR |= BIT4; //设置TXD0口方向为输出

}

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

// MSP430端口初始化

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

void Port\_Init()

{

P6SEL = 0x00; //P6口普通IO功能

P6DIR = 0xFF; //设置IO口方向为输出

P6OUT = 0xEF; //初始设置为关

P1SEL = 0x00; //P1口普通IO功能

P1DIR = 0xF0; //设置IO口方向为输入

}

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

// 处理来自串口 0 的接收中断

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

#pragma vector=UART0RX\_VECTOR

\_\_interrupt void UART0\_RX\_ISR(void)

{

count++; //count自加1计数

data = U0RXBUF; //接收到的数据存放在data中

}

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

// 主函数

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

void main(void)

{

WDT\_Init(); //看门狗设置

Clock\_Init(); //系统时钟设置

Port\_Init(); //LED端口初始化

UART\_Init(); //串口设置初始化

\_EINT(); //开中断

while(1) //无限循环

{

switch(data)

{

case '1':P6OUT &= ~(1<<(count%3));break;//接收到‘1’则点亮相应的LED

case '0':P6OUT |= (1<<(count%3));break; //接收到’0‘则熄灭相应的LED

}

//若三个LED中有任意一个被点亮，则点亮报警LED和蜂鸣器

if(((P6OUT & 0x01)==0)||((P6OUT & 0x02)==0)||((P6OUT & 0x04)==0))

{

P6OUT &= ~BIT3;

P6OUT |= BIT4;

}

//若报警LED被点亮的同时按下消除报警按键

if(((P1IN & BIT0)==0)&&((P6OUT & BIT3)==0))

{

delay\_ms(20); //延时20ms消除抖动

P6OUT |= BIT3; //熄灭报警LED和蜂鸣器

P6OUT &= ~BIT4;

}

else

P6OUT &= 0xFF; //否则报警LED和蜂鸣器保持不变

}

}

附录B

**事件处理函数SampleApp\_ProcessEvent()源程序**

uint16 SampleApp\_ProcessEvent( uint8 task\_id, uint16 events )

{

//定义接收到的消息

afIncomingMSGPacket\_t \*MSGpkt;

//为了避免编译出错，将task\_id屏蔽掉

(void)task\_id;

//如果事件为系统消息事件

if ( events & SYS\_EVENT\_MSG )

{

//接收来自SampleApp\_TaskID任务的消息

MSGpkt = (afIncomingMSGPacket\_t \*)osal\_msg\_receive( SampleApp\_TaskID );

while ( MSGpkt )

{

//当接受的消息有事件发生是，判断事件的类型

switch ( MSGpkt->hdr.event )

{

//按键事件

case KEY\_CHANGE:

//调用按键事件处理函数

SampleApp\_HandleKeys(((keyChange\_t\*)MSGpkt)->state,

((keyChange\_t \*)MSGpkt)->keys );

break;

//接收消息事件

case AF\_INCOMING\_MSG\_CMD:

//调用接收消息事件处理函数

SampleApp\_MessageMSGCB( MSGpkt );

break;

//状态改变事件

case ZDO\_STATE\_CHANGE:

SampleApp\_NwkState = (devStates\_t)(MSGpkt->hdr.status);

if ( //(SampleApp\_NwkState == DEV\_ZB\_COORD)

|| (SampleApp\_NwkState == DEV\_ROUTER)

|| (SampleApp\_NwkState == DEV\_END\_DEVICE) )

{

//启用定时器，打开定时事件

osal\_start\_timerEx( SampleApp\_TaskID,

SAMPLEAPP\_SEND\_PERIODIC\_MSG\_EVT,

SAMPLEAPP\_SEND\_PERIODIC\_MSG\_TIMEOUT );

}

else

{

//设备不在网络中

}

break;

default:

break;

}

//释放内存

osal\_msg\_deallocate( (uint8 \*)MSGpkt );

//等待下一个数据帧的到来

MSGpkt=(afIncomingMSGPacket\_t \*)osal\_msg\_receive( SampleApp\_TaskID );

}

//返回没有处理完的事件

return (events ^ SYS\_EVENT\_MSG);

}

//定时事件

if ( events & SAMPLEAPP\_SEND\_PERIODIC\_MSG\_EVT )

{

//将协议栈中自带的发送周期信息函数替换为发送自定义信息函数

// SampleApp\_SendPeriodicMessage();

SampleApp\_Send\_P2P\_Message();

//定时再次触发事件SAMPLEAPP \_SEND\_ PERIODIC\_ MSG\_ EVT

osal\_start\_timerEx( SampleApp\_TaskID,

SAMPLEAPP\_SEND\_PERIODIC\_MSG\_EV,

(SAMPLEAPP\_SEND\_PERIODIC\_MSG\_TIMEOUT

+(osal\_rand() & 0x00FF)) );

//返回没有处理完的事件

return (events ^ SAMPLEAPP\_SEND\_PERIODIC\_MSG\_EVT);

}

}

**任务初始化函数SampleApp\_Init()源程序**

void SampleApp\_Init( uint8 task\_id )

{

//任务ID号赋值

SampleApp\_TaskID = task\_id;

//网络状态为初始化状态

SampleApp\_NwkState = DEV\_INIT;

//传输序列号赋值

SampleApp\_TransID = 0;

MT\_UartInit(); //串口初始化

MT\_UartRegisterTaskID(task\_id); //注册串口任务

P0SEL &= ~0x10; //设置P0.4为普通IO口

P0DIR &= ~0x10; //P0.4定义为输入口

#if defined ( BUILD\_ALL\_DEVICES )

if ( readCoordinatorJumper() ) //如果检测到跳线则设备为协调器

zgDeviceLogicalType = ZG\_DEVICETYPE\_COORDINATOR;

else //如果没有检测到跳线则设备为路由器

zgDeviceLogicalType = ZG\_DEVICETYPE\_ROUTER;

#endif //都没有检测到为所有设备

#if defined ( HOLD\_AUTO\_START )

//如果定义了HOLD\_AUTO\_START编译选项则执行以下程序

ZDOInitDevice(0);

#endif

//设置周期信息的地址，此地址为广播地址0xFFFF

SampleApp\_Periodic\_DstAddr.addrMode = (afAddrMode\_t)AddrBroadcast;

SampleApp\_Periodic\_DstAddr.endPoint = SAMPLEAPP\_ENDPOINT;

SampleApp\_Periodic\_DstAddr.addr.shortAddr = 0xFFFF;

//设置闪烁信息的地址，此地址为组1的地址

SampleApp\_Flash\_DstAddr.addrMode = (afAddrMode\_t)afAddrGroup;

SampleApp\_Flash\_DstAddr.endPoint = SAMPLEAPP\_ENDPOINT;

SampleApp\_Flash\_DstAddr.addr.shortAddr = SAMPLEAPP\_FLASH\_GROUP;

//设置自定义信息的地址，此地址为单播地址0x0000

SampleApp\_P2P\_DstAddr.addrMode = (afAddrMode\_t)Addr16Bit;

SampleApp\_P2P\_DstAddr.endPoint = SAMPLEAPP\_ENDPOINT;

SampleApp\_P2P\_DstAddr.addr.shortAddr = 0x0000;

//对端点SAMPLEAPP\_ENDPOINT进行描述

SampleApp\_epDesc.endPoint = SAMPLEAPP\_ENDPOINT;

SampleApp\_epDesc.task\_id = &SampleApp\_TaskID;

SampleApp\_epDesc.simpleDesc

= (SimpleDescriptionFormat\_t \*)&SampleApp\_SimpleDesc;

SampleApp\_epDesc.latencyReq = noLatencyReqs;

//注册端点描述符

afRegister( &SampleApp\_epDesc );

//注册按键，按键事件由应用层进行处理

RegisterForKeys( SampleApp\_TaskID );

//默认情况，所有设备都加入组1

SampleApp\_Group.ID = 0x0001; //设定组ID

osal\_memcpy( SampleApp\_Group.name, "Group 1", 7 ); //设定组名

aps\_AddGroup( SAMPLEAPP\_ENDPOINT, &SampleApp\_Group );//加入组

//如果定义了LCD\_SUPPORTED，在液晶上显示"SampleApp"

#if defined ( LCD\_SUPPORTED )

HalLcdWriteString( "SampleApp", HAL\_LCD\_LINE\_1 );

#endif

}