计算机系统应用 ISSN 1003-3254, CODEN CSAOBN
Computer Systems & Applications,2017,26(8):55−59 [doi: 10.15888/j.cnki.csa.006052]

E-mail: csa@iscas.ac.cn
http://www.c-s-a.org.cn
Tel: +86-10-62661041

基于安卓移动终端的无线传感照明控制系统①

方伟骏

(淮安信息职业技术学院 电子工程学院, 淮安 223003)

摘　要: Android 移动终端的 APP 采用 socket 通信技术向服务器发送各种 LED 灯的控制命令; 服务器程序通过 USB 串口通信将控制命令发送给 ZigBee 协调器, 解决了 TCP/IP 协议与 ZigBee 协议栈的融合问题; ZigBee 协调器解读控制命令, 并将相应的控制信号以广播形式转发给所有 ZigBee 终端节点; 而终端节点则根据收到的控制信号驱动以 HV9910B 为核心的 PWM 调光电路, 实现多种 LED 的照明效果. 实验证明, 该系统能够适应多种照明控制系统的需求, 具有较好的可拓展性、实用性和健壮性.

关键词: 照明系统; ZigBee 无线传感网; Android; LED 驱动; Socket

引用格式: 方伟骏.基于安卓移动终端的无线传感照明控制系统.计算机系统应用,2017,26(8):55–59. http://www.c-s-a.org.cn/1003-3254/6052.html

Wireless Sensor Lighting Control System Based on the Android Mobile Terminal

FANG Wei-Jun

(School of Electronic Engineering, Huaian Vocational College of Information Technology, Huaian 223003, China)

Abstract: The Android mobile terminal App applies the socket technology to the communication with the server. The server program sends commands to the ZigBee coordinator by USB serial communication, which has solved the fusion problem between TCP/IP protocol and ZigBee protocol stack. The ZigBee coordinator analyzes the commands, and transmits the corresponding control signal in the form of broadcasting to all ZigBee terminal nodes.The terminal nodes drive the HV9910B PWM dimming circuit, and implement a variety of LED lighting effects according to the control signal received. Experiments show that the system has good practicability, scalable and robustness.

Key words: lighting system; ZigBee wireless sensor network; Android; LED driver; Socket

照明系统在人民生活、工业生产以及市政工程等领域具有举足轻重的作用. 传统的照明系统大多采用有线连接, 具有布线麻烦、能耗损失、扩展不易、维护困难等缺点. 文献[1]指出, 照明控制系统发展到智能化控制阶段, 主要以计算机和网络技术为核心, 利用微处理器技术和存储技术, 通过一定的程序指令控制照明电路中的设备. 调用不同的程序, 执行不同的功能, 就可以有不同的照明水平, 营造出不同的氛围和环境. ZigBee 无线传感网技术基于 IEEE 802.15.4 国际标准协议, 能够提供自由、灵活的组网机制, 具有低复杂度、低功耗、低成本、高可靠性、高安全性等特点; Android 是基于 Linux 的自由及开放源代码的操作系统, 主要应用于移动终端设备, 如设法让 Android 移动终端的 APP 程序嵌入 ZigBee 无线网络, 再通过 ZigBee 终端节点实现对照明驱动电路的控制; 二者结合, 可以实现 Android 移动终端对照明系统的无线远程控制.

1　系统方案设计与论证

基于 ZigBee 协调器和终端节点的硬件模块, 并在此基础上通过 ZigBee 协议栈编程, 控制 LED 驱动电路, 可以组建成无线传感照明网络; 但较难解决的是如

① 基金项目: 江苏省教育厅高校科研成果产业化推进项目(JHB2011-75)
收稿时间: 2016-10-26; 采用时间: 2016-12-08

何实现 ZigBee 无线传感网与 Android 应用程序之间的通信. 方案设计与论证如下:

方案 1. 文献[2]提出 Android 平台与 Bluetooth(蓝牙)模块实现通信系统的观点, 基于 Ubuntul0.04 Linux 操作系统搭建底层源码和应用程序开发环境, 进而实现蓝牙驱动、蓝牙协议栈、BlueZ 适配层和应用层 Android 蓝牙的通信. 然而这种通信系统的蓝牙模块很难实现节点较多的无线传感网络.

方案 2. 文献[3]提出以采用 ARM920T 核的 S3C2440A 芯片为处理器, 搭载 Android 移动操作平台, 并嵌入 ZigBee 无线模块, 通过底层模块驱动设计实现 ZigBee 网络通用化的移动基站和信息处理转发中心; 通过采用 Android 平台开发的上层应用开发调用完成不同 ZigBee 节点的数据收集、处理与转发功能. 此种方案可以将不同协议深度融合, 系统稳定; 但 ARM 与 ZigBee 的系统组建会增加开发周期和难度, 在不是必要的情况下, 不是当前系统构成方案的首选.

方案 3. 文献[4]提出系统采用 GPRS 与 ZigBee 两级网络结构模式, 通过 GPRS 远程通信技术实现 Android 应用程序与 ZigBee 协调器的连接. 此系统较为复杂, 需额外增加路由器模块及相应的 GPRS 网关通信程序. 然而 ZigBee 传感网中协调器和终端节点本来就已是无线通信, 所以没有必要再增加 GPRS 环节.

方案 4. 使用 PC 机作为服务器, 管理和监控功能大大增强, 通过短距离 USB 数据线实现 PC 机与 ZigBee 协调器的连接. 此方案与方案 3 的不同之处在于省略了 GPRS 远程通信的环节, 取而代之的是 PC 机与 ZigBee 协调器之间的串行通信, 开发工作量适中, 能轻松实现 ZigBee 协议栈与 TCP/IP 协议的融合.

2 系统的硬件设计与开发

2.1 CC2530 片上系统节点模块与 I/O 口资源分配

TI 公司的 ZigBee 节点模块目前基于的是 CC2530 片上系统解决方案. 通过星型或树型拓扑结构, 可以组建由不同节点构成的 ZigBee 无线传感网络. 相邻的 Zigbee 节点之间的通信距离为 10-100 m, 能够满足照明系统的设计要求. 节点的类型分为协调器、路由器和终端节点, 为了简化系统选用星型拓扑, 网络中协调器发送命令控制终端节点.

对于 CC2530 来说, I/O 口是重要的内部资源, 以 P1.0 作为驱动 LED 灯光亮度的 PWM 波输出引脚; P1.2 和 P1.3 作为 OLED 显示屏的“时钟”和“数据”引脚, 通过 I^2C 方式实现 CC2530 与 OLED 的通信.

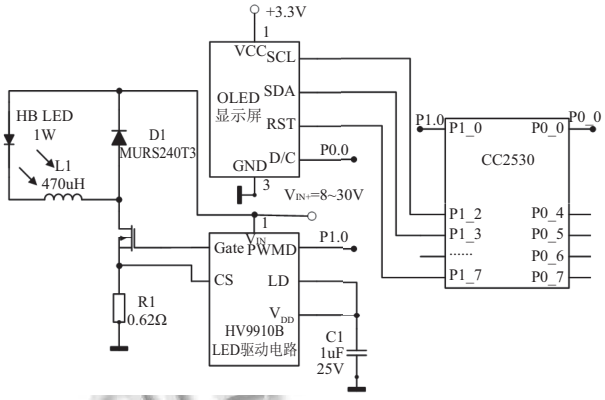

图 1 CC2530 的 I/O 口资源分配电路图

2.2 PWM 调光的 LED 驱动电路

终端节点如果接收到协调器的命令, 则根据不同的命令输出不同占空比的 PWM 波至 LED 驱动电路; LED 驱动电路以高亮度 LED 驱动芯片 HV9910B 为核心, 目的是将脉冲宽度调制 PWM 信号转换为精准的恒定电流, 从而驱动高功率 LED.

降压恒流 LED 驱动电路的设计与参数如图 2 所示. 工作过程及原理描述为: 输入可选用 8-30V 的直流电压, 正极接到 HV9910B 的 VIN 引脚, 输入电压通过电容 C2 滤波. 场效应管 IRFL014 起到开关的作用: 当其打开时, 输入电流经过负载高亮 LED、电感 L1、开关管 IRFL014、检测电阻 R1 流入到地. 在这个过程中, 电感不断存储能量并且电流线性上升, 检测电阻 R1 上的电压也跟着不断上升; 当 R1 上的电压达到 HV9910B CS 引脚的检测电压时, 开关管 IRFL014 关闭. IRFL014 关闭时, 电感中的电流继续通过续流二极管 MURS-240T3 为负载 LED 供电, 这个过程中电感电流从峰值线性下降. 其中输出电流即为电感电流的平均值.

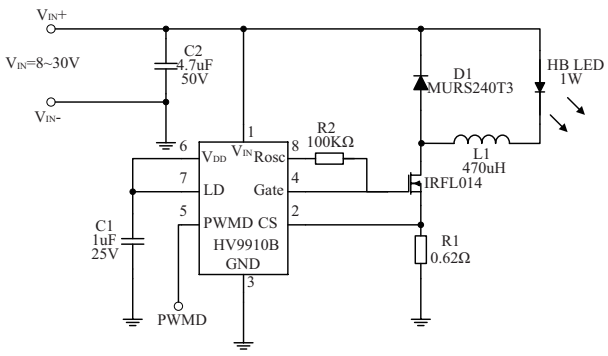

图 2 基于 HV9910B 的降压恒流 LED 驱动电路

3 系统的软件设计与开发

3.1 终端节点控制 PWM 波程序设计

PWM(pulse width modulation)即脉冲宽度调制, ZigBee 终端节点输出 PWM 波, 通过变化 PWM 波的占空比从而达到控制 LED 灯光亮度的目的.

在 SampleApp.C 文件中, SerialApp_Init()函数承担着对定时器、串口等各种资源的初始化工作. 初始化定时器的函数设定为 InitT1(), 而改变 PWM 波的占空比, 就是通过对定时器等资源进行相应的初始化完成的.

在 InitT1()函数中, PERCFG |=0x40 指定时器 1 的 I/O 位置选择在“备用位置 2”; P2SEL &= ~0x10 指当 PERCFG 分配定时器 1 和定时器 4 到相同的引脚的时候, 确定定时器 1 优先; P2DIR |=0xC0 指第 1 优先级是“定时器 1 通道 2-3”; P1DIR |=0xff 确定 P1.0 使用的是“通道 2”. 综合以上为: 通过 P1.0 引脚可以 “在通道 2 优先使用定时器 1”输出具有一定占空比的 PWM 波.

在此基础之上, 就可以通过 LedEvent(uint16 bright)函数实现对 LED 灯亮度的调节, 其实质性作用就是改变 PWM 波的占空比. T1CCTL2=0x1c 确定了定时器 1 工作在“输出比较模式”; T1CTL=0x02 确定了定时器 1 工作在“模模式”.

与定时器 1 有关的寄存器有“捕获/比较值” T1CC2, 将 T1CC2H 初始化为 0x00, 将形参 bright 值传递给 T1CC2L, 通过形参值的不断变化, 就可以改变 PWM 波的占空比. 从图 3 中可见, 定时器 1 从 0000H 开始不断增加计数值, 当计数值达到 T1CC2(实参 bright 的值)时, P1.0 引脚输出脉冲置 1; 当计数值达到 T1CC0(当前设置为 0xFF=255)时, P1.0 引脚输出脉冲清 0. 所以 PWM 波的占空比计算为:

$$占空比 = \frac{T1CC2}{T1CC0} = \frac{bright}{255} \quad (1)$$

在调用 LedEvent(uint16 bright)函数时, 只要改变实参 bright 的值, 就可以方便地通过改变 PWM 波的占空比实现对 LED 灯亮度的调节.

3.2 ZigBee 无线传感网调光控制程序设计

ZigBee 协调器是否向终端节点发送调光的控制信号, 取决于 PC 机(服务器)发送给它的控制命令, 控制命令中有预先设计好的功能位, 如功能位为 0x00 则增加亮度, 否则减小亮度. 在增加亮度时, 如果协调器第一次接收到服务器发来的控制命令, 则在函数 AF_DataRequest()中把标记 LED 状态的变量 state_led 以值 0x31 发送出去, 通知终端节点增加 LED 亮度; 如果协调器第二次接收到服务器发来的控制命令, 则以 0x32 发送出去, 通知终端节点停止增加 LED 亮度. 减小亮度时原理如前.

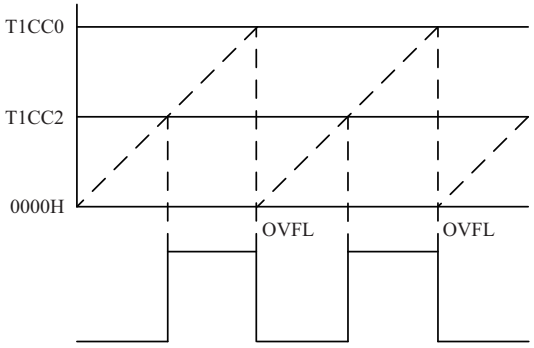

图 3 P1.0 引脚在“模模式”输出 PWM 波

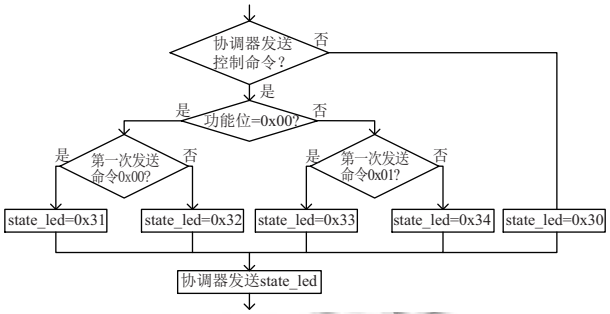

图 4 协调器控制调光程序算法流程图

需要说明的是: 协调器在没有发送 0x31、0x32、0x33 和 0x34 时, 都一直在发送心跳信号 0x30, 因为终端节点需要通过每一次接收到的 0x30 来逐渐增加或减少 LED 的亮度.

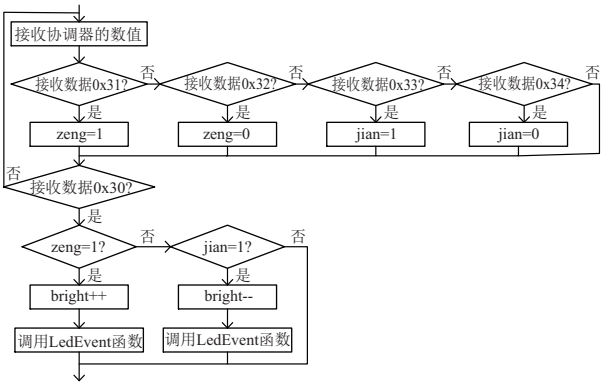

图 5 终端节点驱动调光程序算法流程图

通过调用函数 SampleApp_MessageMSGCB(), 终端节点接收协调器发来的信号 state_led, 并通过变量“zeng”和“jian”的值为 1 还是为 0 来标记增加亮度还是停止增加亮度, 亦或是减少亮度还是停止减少亮度, 最后通过调用 LedEvent(bright)函数来实施具体动作. 实参 bright 的值上限为 255、下限为 0, 增大到最大值或是减小到最小值, 都将不再变化.

3.3 7 档 LED 调光与 LED 的定时控制程序设计

由于调用 LedEvent(bright)函数的效果, 就是将 LED 以 bright/255 为占空比点亮, 所以 bright 为 0 时 LED 熄灭, bright 为 255 时 LED 最亮. 要想将 LED 的亮度分为 7 段, 每调亮 1 段, 就将 bright 的值增加 255/7≈36(近似); 每调暗 1 段, 就将 bright 的值减小 36. 需要注意的是, bright 的值最小为 0, 最大为 255, 不能超过其数值范围.

CC2530 的定时器可以工作在查询模式和中断模式, 相比之下, 中断模式更能节省 CPU 资源, 所以设置定时器 3, 使其工作在中断模式, 从而实现 LED 的定时控制.

首先将 T3CTL |=0x08 开溢出中断, T3IE=1 开总中断和 T3 中断; 接着将 T3CTL|=0XE0, 意味着 128 分频, 并有数值关系: 128/16000000*N=0.5(秒), 经计算可得 N=65200. T3CTL &= ~0X03 是将定时器 3 设置为从 0x00 到 0xff 进行自动重装. 所以如果要实现 0.5 秒时长的定时, 其实就是要进行 65200/256=254(次)的定时中断, 于是在中断函数中, 定义变量 count, 对定时器 3 产生的中断进行计数; 很显然, 定时 0.5 小时需中断 254*2* 60*30=914400 次, 1 小时需中断 1828800 次, 以此类推, 这样就实现了对 LED 进行不同时长的定时控制.

3.4 服务器程序设计

Android 移动终端难以直接与 ZigBee 协调器进行通信, 所以借助 PC 机(服务器)作为间接远程通信的桥梁. 服务器应用程序基于 Visual C++ 6.0 环境开发, 需要实现的重点功能是“打开串口”和“启动网络服务”.

打开串口的动作需要启动串口监视线程 Start-Monitoring(), 并初始化串口, 将波特率设置为 38400; 启动网络服务需要用套接字启动命令 WSAStartup()对 TCP 协议初始化, 从而可以将操作系统指定的 Socket 库绑定到应用程序中, 使应用程序能调用任何 Winsock API 函数; 再用 bind()函数绑定到本地的一个端口上, 并用 listen()函数实现服务器的监听. 在服务器监听的过程中, 如果收到来自 Android 移动终端的控制命令, 则将命令存放在一维数组 strTxBuf[]中, 并将该命令通过 USB 串口发送给 Zigbee 协调器.

3.5 Android 终端 APP 设计

Android 移动终端与服务器之间的通信, 基于的是 TCP 协议, 而 socket 连接是在 Java 或 VC++环境中针对 TCP 协议常用的网络通信方式, 所以 Android 移动终端与服务器通信时需要建立起成对的 socket 连接. Socket 工作时使用的类主要是 ServerSocket 和 Socket.

① 服务器端 socket 通信的核心程序为:

```
ServerSocket=socket(2, STREAM, TCP);
//创建服务器端套接字
sockaddr_in localaddr; //设置端口 33333 将套接字绑定到端口
localaddr.sin_port=htons(33333);
bind(ServerSock, localaddr, sizeof(sockaddr));
Listen(ServerSock, 5); //服务器监听移动终端
switch (selectevent )
//监听的过程中对不同情况作不同处理
{Case ACCEPT :
OnAccept (ClientSock ); break;
//如有移动终端连接请求则与客户端建立连接
Case READ:
OnReceive (ClientSock ); break;
//如收到移动终端的控制命令则按要求回应
Case CLOSE:
closesocket (ClientSock); break; }
//如与移动终端连接断开则关闭客户端套接字
```

② 移动终端 socket 通信的核心程序为:

```
socketAddress=new InetSocketAddress(strIP, SERVER_PORT);
//创建服务器 socket, IP 地址和端口号
socket=new Socket(); //创建 socket 实例
socket.connect(socketAddress, SERVER_PORT);
//根据地址和端口号建立 socket 连接
inputStream=socket.getInputStream();
//获得数据的输入流和输出流
outputStream=socket.getOutputStream();
inputStream.read(RxBuf);
//从输入流读取服务器数据存入 RxBuf 数组
```

socket 通信程序设计的基本算法与流程是:

① 服务器创建 ServerSocket 套接字, 设置端口 33333, 并将套接字绑定到端口, 然后实施对 Android 终端的监听. 在监听的过程中, 如果收到终端的连接请求, 则建立连接.

② 移动终端通过 WIFI 或 GPRS 流量的形式, 直接以服务器端的 IP 地址和监听端口号为参数实例化 Socket 类, 从而连接服务器.

③ 移动终端与服务器通过“流”的形式进行数据交互, 调用 getInputStream()方法得到输入流接收服务器发送来的数据; 调用 getOutputStream()方法得到输出流, 向其中写入数据并将流传输到服务器.

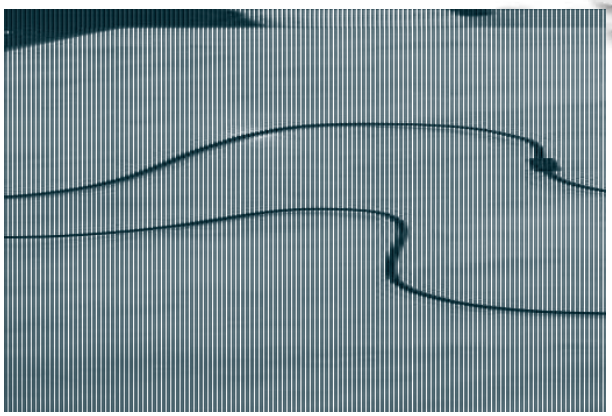

图 6 无线传感照明控制系统实物图

在 MainActivity 中, 设计了“网络设置”的功能模块, 用户可以输入服务器的 IP 地址, 程序根据地址建立并启动移动终端的线程. 主线程消息处理中心 initMainHandler()的作用是: 当 Android 移动终端需要进行灯光控制时, 可以点击不同的按钮控件 Button 完成, 操作功能丰富, 界面友好; 其实质是向服务器发送各种 LED 灯的控制命令, 可以实现:

① LED 亮度由暗变亮或是由亮变暗;

② 实现市面上常见的 LED 台灯 7 档调光;

③ 对照明进行不同时长的定时控制, 到达设定时间后自动熄灭.

4 结语

基于 Android 移动终端和 Zigbee 无线传感网融合的照明控制系统, 通过实验证明, 该无线传感网在需要距离延伸和节点接力时, 通过修改 ZigBee 网络的拓扑结构, 可以拓展为 N 个终端节点, 能够适应各种照明控制系统的需求; 该系统不再通过传统的按键或者服务器软件来管理照明系统, 而是采用随处可见的 Android 手机实现对 LED 的多样化控制, 克服了传统照明系统的诸多缺陷, 具有较好的实用性、健壮性和良好的市场前景.

参考文献

1 杨卫桥. 智能照明控制系统发展及趋势. 仪表技术, 2014, (9): 46–48.
2 罗富财. 基于 Android 平台的蓝牙通信系统的研究与实现[硕士学位论文]. 保定: 华北电力大学, 2013.
3 李睿. 基于 ZigBee 的移动智能终端在物联网智能家具中的应用[硕士学位论文]. 北京: 北京邮电大学, 2011.
4 杨世江. 基于 ZigBee 和 GPRS 技术的智能照明控制系统的研究与实现[硕士学位论文]. 吉首: 吉首大学, 2015.
5 田亚辉. 基于 ZigBee 的城市路灯无线监控系统研究[硕士学位论文]. 大连: 大连理工大学, 2013.
6 冯智磊. 基于 WiFi 网络和单片机的智能室内 LED 照明系统设计. 电气应用, 2015, 34(2): 40–44.
7 章洁, 秦会斌, 毛祥根. 基于 ZigBee 的室内智能照明系统设计. 物联网技术, 2013, (9): 22–24.
8 李丽莉. 基于 ZigBee 技术的无线传感器网络节点的设计与实现[硕士学位论文]. 成都: 西南交通大学, 2011.