智能家居控制系统

毕业论文

目　　录

引言

21 世纪是信息化的世纪，各种电信和互联网新技术推动了人类文明的巨大进步。本文介绍的数字化家居控制系统可以使得人们可以通过手机或电话在任何时候、任意地点对家中的任意电器（空调、热水器、电饭煲、灯光、音响、DVD 录像机）进行远程控制；也可以在下班途中，预先将家中的空调打开、让热水器提前烧好热水、电饭煲煮好香喷喷的米饭……；而这一切的实现都仅仅是打一个简单的电话。此外，该系统还可使家庭具有多途径报警、远程监控等多种功能，如果不幸出现某种险情，您和 110 可以在第一时间获得通知以便进一步采取行动。舒适、时尚的家居生活是社会进步的标志，智能家居控制系统能够在不改变家中任何家电的情况下，对家里的电器、灯光、电源、家庭环境进行方便地控制，使人们尽享高科技带来的简便而时尚的现代生活。

实现智能化离不开运算和控制单元，本系统采用 MCU(SM8952AC25P)作为主控器件，单片机应用系统由硬件和软件组成。硬件由单片机扩展的存储器、输入/出设备以及各种实现单片机系统控制要求的接口电路和有关的外围电路芯片或部件组成；软件由单片机应用系统实现其特定控制功能的各种工作程序和管理程序组成。在单片机应用系统开发的过程中，应不断调整软、硬件，协调地进行软、硬件设计，以提高工作效率，当系统硬件和软件紧密配合、协调一致，就可以组成高性能的单片机应用系统。本课题完成了单片机应用系统其开发过程的系统的总体设计、硬件设计、软件设计和系统调试，根据开发的实际需要，相互协调、交叉，有机的进行。本设计的 MCU 与各个芯片和模块的接口、各项标准都严格遵循国家有关标

准，为以后的产品化提供了良好的基础。

本系统的电话远程控制是基于电话交换网络的国际双音频通信标准DTMF通信方式，程控交换信令作为系统控制命令，采用MT8870双音频编解码电路实现，单片机通过MT8870识别来自电话程控交换机的网络的控制信号，用户只需拨通家中的电话可以根据系统的语音提示进行按键选择以实现用户身份的识别、远程控制和安防操作；各种传感器的检测是利用数据采集系统将多路被测量值转换成数字量，再经过单片机进行数据处理，实现实时测控；短消息发送部分采用基于SIEMENS TC35 GSM模块TC35 modem和TI公司的电平转换芯片MAX3238等器件构成的移动终端的硬件电路可以完成短消息收发等功能。

在设计本系统时，面对各种检测对象和大量控制单元，需要利用各种接口标准和MCU进行连接，再经过MCU进行数据处理，实现实时测控。而此时采用单片机来实现智能家居控制系统不仅具有采集控制方便、简单、灵活等优点，而且可以大幅度提高采各模块和芯片的协调性，从而大大提高系统的可利用性。此次系统设计统正是把MT8870、TC35 modem与SM8952AC25P单片机有机的结合起来，顺利的完成了本设计的要求。并且实现了学习型远程红外遥控功能，为控制红外家电和设备提供了良好的基础。本系统也可应用于工农业中,实现对无人值守岗位的远程监控等。

1　绪 论

1.1 课题背景

21世纪是信息时代,各种电信新技术推动了人类文明的进步。自从1876年，Alexander Graham Bell（贝尔）发明电话以来，世界各国的电话网络发展非常迅速，近十年来，中国的固定电话业务呈现出举世瞩目的快速增长。1997年8月局用电话交换机总容量突破1亿门，网络规模跃居世界第二位，2006年初固定电话用户总数达到35539.2万户，移动电话用户达到40407.2万户，现代电话网络是由程控交换机进行交换传输，移动通信也从模拟时代走向了高度数字化时代，它们的性能已经有了很大的进展，而且可靠性非常高。

正是因为通信技术、计算机技术、网络技术、控制技术的迅猛发展与提高，促使了家庭实现了生活现代化，居住环境舒适化、安全化。这些高科技已经影响到人们生活的方方面面，改变了人们生活习惯，提高了人们生活质量，家居智能化也正是在这种形势下应运而生的。智能家居控制系统的主要功能包括通信、设备自动控制、安全防范三个方面。

随着新技术和自动化的发展，传感器的使用数量越来越大，功能也越来越强，各种传感器都已经标准化、模块化这给智能家居控制系统的设计

提供极大方便。

电话远程控制作为一较新的课题与常规的遥控方式相比，显示出一定的优越性，不需进行专门的布线，不占用无线电频率资源，避免了电磁污染。同时，由于电话线路各地联网，可以充分利用现有的电话网，因此遥控距离可跨省市，甚至跨越国家。另外电话属双工通信手段。因此，这可以大大体现出利用电话进行遥控的更大优越性。操作者可以通过各种提示音即时了解受控对象的有关信息，从而进行进一步的操作。电话遥控部分课题目前已有涉足者，但是只是还只限于实验室阶段，因而距离实际应用,尤其是对于日常生活尚有一定的差距，并不能完全体现出电话遥控方式的双工通信特点。本设计正是针对这一点进行了较大改进，采取单片机智能控制，利用不同的提示音达到对于不同操作的提示及对受控方状态的信息反馈，从而使操作者能够及时了解受控方信息，使产品达到交互式与智能化。而且本设计的调试都是在线调试，已经在电信、铁通的交换机实验并且能够成功的使用移动电话进行操作。

短信息服务 (Short Message Service，SMS）是 GSM (Global System for Mobile Comm unication）系统中提供的一种 GSM 终端（手机）之间，通过服务中心(service center)进行文本信息收发的应用服务，其中服务中心完成信息的存储和转发功能。短信息服务作为 GSM 网络的一种基本业务，已得到越来越多的系统运营商和系统开发商的重视，基于这种业务的各种应用也蓬勃发展起来。由于 GSM 网络在全国范围内实现了联网和漫游，具有网络能力强的特点，用户无需另外组网，在极大提高网络覆盖范围的同时为客户节省了昂贵的建网费用和维护费用。同时，他对用户的数量也没有限制，克服了传统的专网通信系统投资成本大、维护费用高、且网络监控的覆盖范围和用户数量有限的缺陷。比传统的集群系统在无线网络覆盖上具有无法比拟的优势，加上 GSM 的 SMS 本身具备的数据传送功能，都使得这些应用得到迅速的普及。利用 GSM 短信息系统进行无线通信还具有双向数据传输功能，性能稳定，为远程数据传送和监控设备的通信提供了一个强大的支持平台。在此以 GSM 网络作为数据无线传输网络，它可以应用在银行、储蓄点机房监控、电信机房动力环境监控、通信行业远端无人值守站机房监控和远程维护（如移动通信基站、微波站、光纤中继站等）及其他无人值守点（如仓库、办公楼等）监控及城市公用事业实时监控维护系统像煤气调压站、自来水、污水管网和热力系统 、电力系统城市中电网等情况中。在此本系统采用了 Siemens 公司新一代无线通信 GSM 模块 TC35 是，它设计小巧、功耗很低很大程度上方便了智能家居控制系统的设计。

1.2 智能家居控制系统的概述

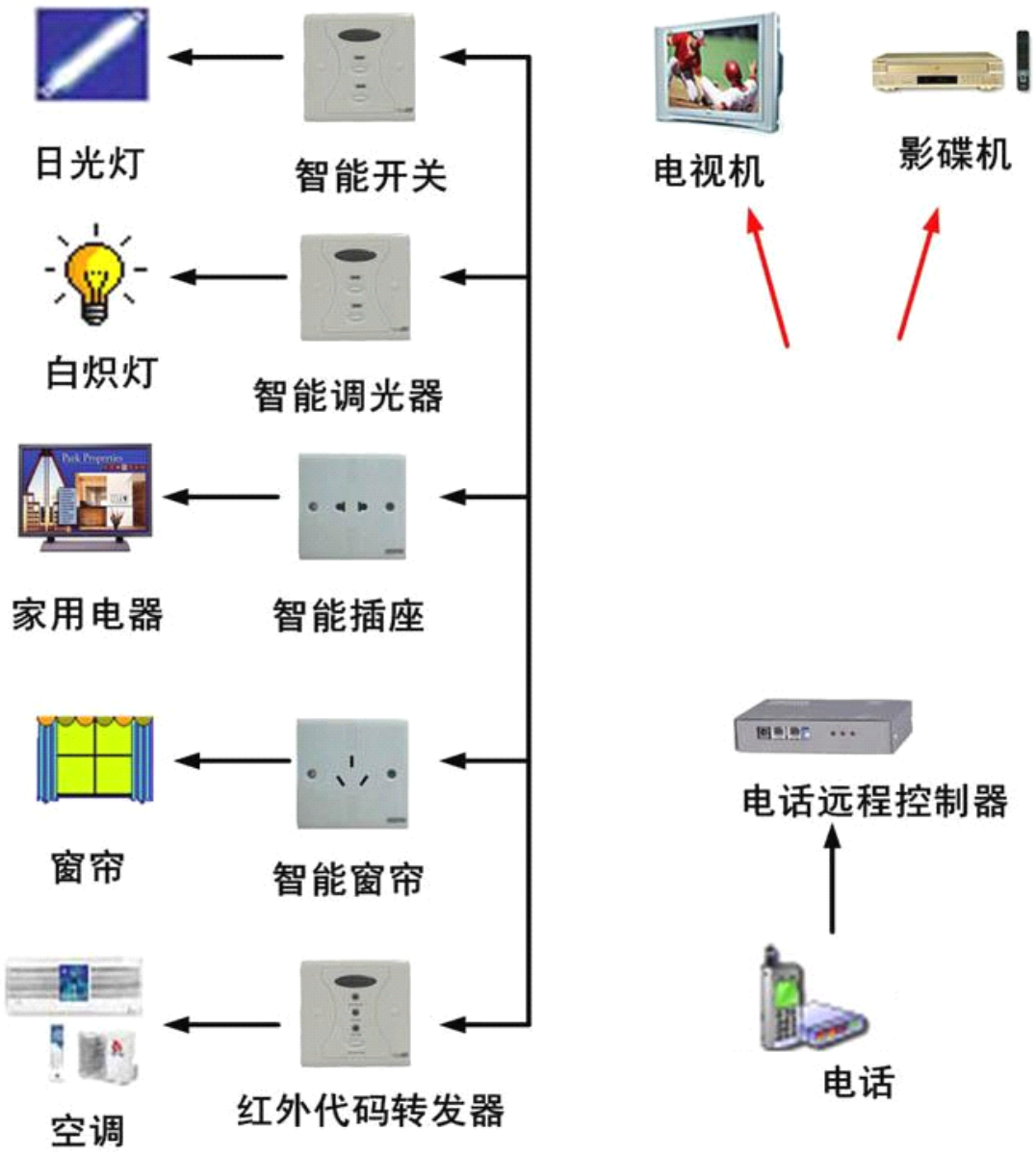


图 1.1-1 智能家居控制系统示意图

随着社会信息化的加快，人们的工作、生活和通讯、信息的关系日益紧密。信息化社会在改变人们生活方式与工作习惯的同时，也对传统的住宅提出了挑战，社会、技术以及经济的进步更使人们的观念随之巨变。人们对家居的要求早已不只是物理空间，更为关注的是一个安全、方便、舒适的居家环境。家居智能化技术起源于美国，它是以家为平台进行设计的。

智能家居控制系统是以 HFC、以太网、现场总线、公共电话网、无线网的传输网络为物理平台，计算机网络技术为技术平台，现场总线为应用操作平

台，构成一个完整的集家庭通信、家庭设备自动控制、家庭安全防范等功能的控制系统。

智能家居控制系统的总体目标是通过采用计算机技术、网络技术、控制技术和集成技术建立一个由家庭到小区乃至整个城市的综合信息服务和管理系统，以此来提高住宅高新技术的含量和居民居住环境水平。

大型的智能家居控制系统通常由系统服务器、家庭控制器(各种模块)、各种路由器、电缆调制解调器头端设备 CMTS、交换机、通讯器、控制器、无线收发器、各种探测器、各种传感器、各种执行机构、打印机等主要部分组成。

1.3 课题研究的目的及意义

智能家居控制系统可以定义为一个过程或者一个系统。利用先进的计算机技术、网络通讯技术、综合布线技术、将与家居生活有关的各种子系统，有机地结合在一起，通过统筹管理，让家居生活更加舒适、安全、有效。与普通家居相比，智能家居不仅具有传统的居住功能，提供舒适安全、高品位且宜人的家庭生活空间。还将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换畅通，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，甚至为各种能源费用节约资金。

系统的网络化功能可以提供遥控、家电（空调，热水器等）控制、照明控制、室内外遥控、窗帘自控、防盗报警、电话远程控制、可编程定时控制及计算机控制等多种功能和手段。使生活更加舒适、便利和安全。因智能家居控制系统布线简单、功能灵活，扩展容易而被人们广泛接受和应用。

1.4 系统设计主要任务

本文利用 SM8952AC25P、MT8870、TC35 modem 和各类家居传感器设计制作一款智能家居控制系统样机。实验样机的设计包括：系统硬件的设计与调试和控制软件的编写与调试。

(1)硬件部分

智能家居控制系统其硬件部分主要由五大部分构成，即电话交换网程控交换信令部分、控制单元、传感器数据采集系统、GSM 模块 TC35 modem 与接口和电源部分。电话交换网程控交换信令部分主要由 MT8870 和电压检测元件组成，它是系统中控制部分关键的元件，它与控制单元组成控制部分功能；传感器数据采集系统、GSM 模块 TC35 modem 和控制单元完成报警等信号的处理和发送；电源部分则为各个部分提供工作电源。

(2)软件部分

软件设计部分主要由五大部分构成：即数据采集与数据分析部分、电

话交换网程控交换信令识别与分析部分、GSM 模块 TC35 modem 接口程序部分、分析控制部分。其中数据采集与数据分析部分和电话交换网程控交换信令识别需要作实时处理；GSM 模块 TC35 modem 接口程序部分和分析控制部分则是根据采集和电话交换网交换信令进行分时操作有利于提高系统效率。

2 方案设计

2.1 系统总体设计与分析

本设计属于单片机应用系统。确定单片机控制系统总体方案，是进行系统设计最重要、最关键的一步。总体方案的好坏，直接影响整个控制系统的性能及实施细则。总体方案的设计主要是根据被控对象的任务及工艺要求而确定的。设计方法大致如下：根据系统的要求，首先确定出系统是采用开环系统还是闭环系统，或者是数据处理系统。选择检测元件，在确定总体方案时，必须首先选择好被测参数的测量元件，它是影响控制系统精度的重要因素之一。选择执行机构，执行机构是微型机控制系统的重要组成部件之一。执行机构的选择一方面要与控制算法匹配，另一方面要根据被控对象的实际情况确定。 选择输入/输出通道及外围设备。选择时应考虑以下几个问题：被控对象参数的数量；各输入/输出通道是串行操作还是并行操作；各通道数据的传递速率；各通道数据的字长及选择位数；对显示、打印有何要求；画出整个系统流程图和原理图。

2.1.1 单片机控制部分

本系统是单片机在系统检测以及工程控制方面的应用，其特点是体积小，成本低，功能强，功耗低，是微机应用产品化的最佳机种之一，它已广泛地应用在产品智能化和工业自动化上。而把单片机面向工控领域对象,嵌入到工控应用系统中，实现嵌入式应用的计算机称之为嵌入式计算机系统，简称嵌入式系统。嵌入式系统一般分为四种:工控机，通用 CPU 模块，嵌入式微机处理，单片机。嵌入式系统具有以下特点：

(1)面对控制对象。如传感信号输入、人机交互操作,伺服驱动等。

(2)嵌入到工控应用系统中的结构形态。

(3)能在工业现场环境中可靠运行的品质。

(4)突出控制功能。如对外部信息的捕捉、对控制对象实时控制和有突出控制功能的指令系统(I/O 控制、位操作和转移指令等)。

单片机有惟一的专门为嵌入式应用系统设计的体系结构与指令系统，最能满足嵌入式应用要求。单片机是完全按嵌入式系统要求设计的单芯片

形态应用系统，能满足面对控制对象、应用系统的嵌入、现场的可靠运行及非凡的控制品质等要求，是发展最快、品种最多、数量最大的嵌入式系统。

2.1.2 系统工作流程部分

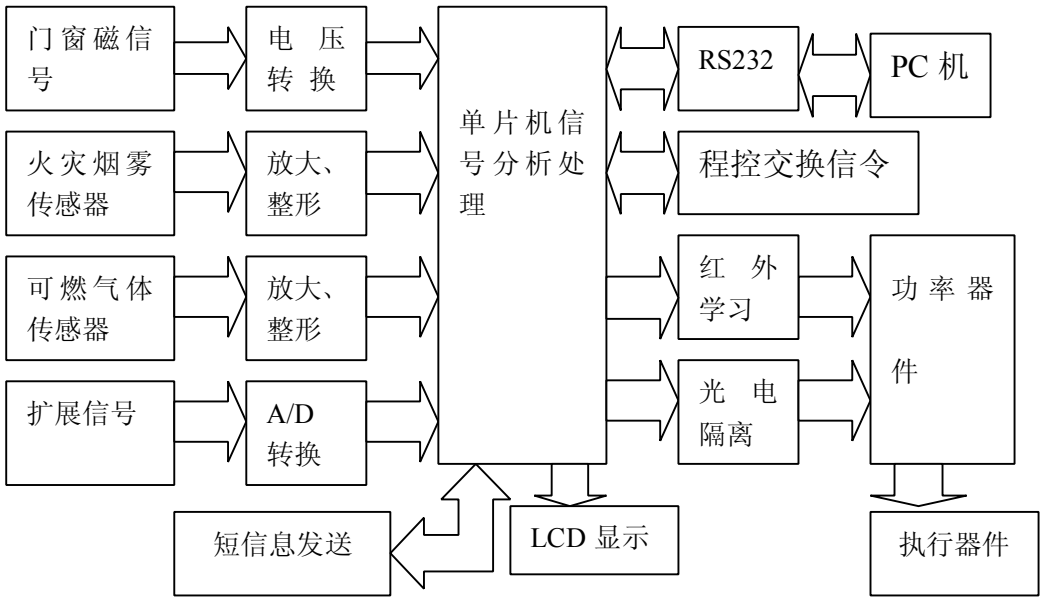


图 2.1-1 系统工作流程

2.2 远程控制设计与分析

2.2.1 控制系统设计分析

系统控制器由 MT8870 接收远端发送来的 DTMF 信号、并对其进行解码，解码后的信号由中央处理单元采集处理。为了方便用户使用，系统设计了语音提示界面。电话远程控制系统一般工作在无人值守环境，所以应具有自动离线、上线、复位功能。为了符合

智能化要求，系统采用 SM8952AC25P 作为中央处理器。同时，电话远程控制系统正常工作还需电源供电电路、驱动电路等辅助电路。系统组成框图如图 2.2-1 示。由图可知，系统主要由振铃检测电路、模拟摘挂机电路、DTMF 音频解码电路、语音提示电路、中央处理单元（SM8952AC25P）、控制电路、电源电路等组成。

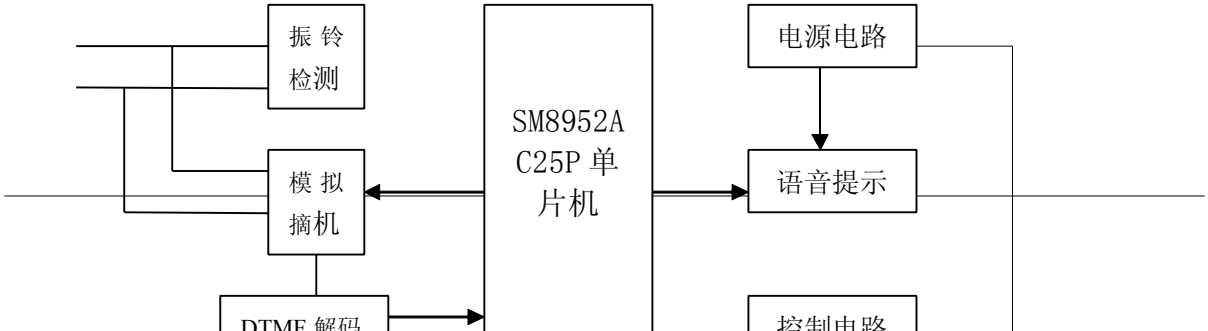


图 2.2-1 远程控制系统

控制器并联于电话机的两端，不会影响电话机的正常使用。用户通过异地电话拨通控制器所连接外线的电话号码，通过程控交换机向电话机发出振铃信号。控制器检测到六次振铃后，即六次响铃后无人接听时电话自动摘机，进入密码检测阶段，输入正确密码后选择被控电器，然后输入开或关密码遥控被控电器，完成后返回。

2.2.2 控制要求

完成远程控制部分必须完成一下要求：

(1) 通过电话网对异地的电器实现控制（开/关）。

(2) 控制器可以实现自动模拟摘挂机。

(3) 控制器设置密码校验。

2.2.3 单元功能模块

本设计此系统必须具有以下单元功能模块：

(1) 铃音检测、计数；

(2) 自动摘挂机；

(3) 密码校验；

(4) 在线修改密码；

(5) 双音频信号解码；

(6) 输入信息分析；

(7) 控制电器开关；

(8) 电器状态查询；

2.2.4 软硬件方案确定

根据电话机和交换机发出的不同信号音以及电话线各种状态的不同要求，本设计方案结合实际情况对具体的单元功能模块作出软件或硬件上作了以下分工：

交换机所发出的各种信号音都大多数可以通过软件编程来识别，即通过单片机发出的脉冲信号来检测信号音单位时间内的脉冲个数计算出其频率，从而完成信号音的识别。但从系统的可靠性和程序的结构设计上分析，用硬件来解决振铃音检测、双音频信号解码等功能相对稳定可靠。 设计中自动摘挂机和电器的控制必须使用具体硬件电路来实现。而振铃音计数、密码校验、在线修改密码、输入信息分析、电器状态查询等功能模块使用软件编程方式要比硬件电路简单的多，易于实现。

综上所述，本设计信号音检测、自动摘挂机、控制电器、双音频解码等功能模块使用硬件电路实现。而信号音计数、密码校验、在线修改密码、信息分析、电器状态查询等功能模块使用软件编程完成。

2.3 传感器信号采集设计与分析

2.3.1 防火灾发生传感器

火灾烟雾传感器是一种在消防管理、安全防范系统中常用的报警器材，它工作可靠、体积小巧，火灾烟雾传感器是把烟雾信号转换为电信号，电路设计时可以在背后设计专用的卡口以便地固定在墙体或者天花板上，使用时通过设置在厨房的感温探测器和设置在客厅、卧室等的感烟探测器，监视各个房间内有无火灾的发生。如有火灾发生控制器发出声光和短信报警信号，通知家人及小区物业管理部门。家庭控制器还可以根据有人在家或无人在家的情况，调节感温探测器和感烟探测器的灵敏度。本系统采用 NIS-09C 型烟雾传感器。

2.3.2 可燃气体泄漏传感器

通过设置在厨房的可燃气体探测器，监视燃气管道、灶具有无燃气泄漏。如有燃气泄漏家庭控制器发出声光报警信号，并联动关闭燃气管道上的电磁阀，同时短信通知家人及小区物业管理部门。这里采用 TP-2 高温型一氧化碳传感器报警器需在一氧化碳浓度达到设定制值时系统应启动报警。

2.3.3 防盗传感器

防盗报警的防护区域分成两部分，即住宅周界防护和住宅内区域防护。住宅周界防护是指在住宅的门、窗上安装门磁开关，在对外的玻璃窗、门附近安装玻璃破碎探测器；住宅内区域防护是指在主要通道、重要的房间

内安装被动红外探测器或被动红外／微波双技术探测器。当家中有人时，住宅周界防护的防盗报警设备(门磁开关、玻璃破碎探测器)设防，住宅内区域防护的防盗报警设备(红外探测器或被动红外／微波双技术探测器)撤防。当家人出门后，住宅周界防护的防盗报警设备(门磁开关、玻璃破碎探测器)和住宅内区域防护的防盗报警设备(被动红外探测器或被动红外／微波双技术探测器)均设防。当有非法侵入时，家庭控制器发出声光报警信号，并短信通知家人及小区物业管理部门。另外，通过程序可设定报警装置的等级和报警器的灵敏度。

2.3.4 信号采集设计与分析

传感器是将外界信息转换成电信号的装置，其中的电信号包括数字信号和模拟信号，数字信号可以通过电气隔离直接送给单片机或微型计算机进行处理，模拟信号则利用数据采集系统将多路被测量值转换成数字量，再经过单片机或微型计算机进行数据处理，实现实时测控。

2.4 GSM 模块的接口与设计

2.4.1 TC35 模块组成

Siemens 公司的 TC35 模块主要由 GSM 基带处理器、GSM 射频模块、供电模块(ASIC)、闪存、ZIF 连接器、天线接口六部分组成。作为 TC35 的核心，基带处理器主要处理 GSM 终端内的语音、数据信号，并涵盖了蜂窝射频设备中的所有的模拟和数字功能。在不需要额外硬件电路的前提下，可支持 FR、HR 和 EFR 语音信道编码。

2.4.2 TC35 模块通信电路

数据通信电路主要完成短消息收发、与微机机通信、软件流控制等功能。TC35 的数据接口采用串行异步收发，符合 ITU-T RS-232 接口电路标准，工作在 CMOS 电平(2.65V)。数据接口配置为 8 位数据位、1 位停止位、无校验位，可以在 300bps～115kbps 的波特率下运行，支持的自动波特率为 4.8kbps～115kbps（14.4kbps 和 28.8kbps 除外）。TC35 模块还支持 RTSO/CTSO 的硬件握手和 XON/XOFF 的软件流控制。

2.4.3 TC35 模块与 MCU 连接方式

单片机与 TC35 一般采用串行异步通信接口，具有红外和通信电缆两种连接方式其中电平转换及串口通信功能以 TI 公司的 MAX3238 芯片为核心，通信速度可设定，通常为 19200bps。采用红外接口的优点是单片机系统与手机电气隔离，相互不干扰，接口各自独立，使用方便；缺点是通信距离较短，红外传播的方向性对接口相对位置有要求。采用电缆连接时，数据传输的可靠性较好；其主要缺点是接口的电气参数不兼容，设计不当时就会对手机的通信质量产生影响。这个单片机系统其实就是一个具有 GPRS 功

能的GSM手机模块加上单片机控制系统,GSM手机模块主要是用来建立无线信道，接收和发出短消息。单片机系统用来控制手机模块，并且对收到的短消息信息进行解释并执行，目前全国共有8种短消息格式包含从互联网平台发出的短消息。本系统采用通信电缆的连接方式。

2.5 红外学习遥控设计

2.5.1 红外学习遥控的设想

对于空调、电视等需要红外遥控器才可控制的红外控制类家电仅靠接通交流电源是无法使其进入工作状态的。因此本人提出使用红外遥控进行对空调以及其他红外电气的控制方案，使系统具有的红外学习功能可实现对这类家电的有效控制。又如红外插座、红外开关等，虽然红外遥控在家电产品中有广泛应用，但各产品的遥控器不能相互兼容，目前市面上常见的万能遥控器只能对某几种产品进行控制，不是真正的“万能”。本系统可以学习并记忆各种红外控制类家电的遥控指令，利用单片机对遥控器的发射信号的波形进行测量，然后将测量的数据回放，由于只关心发射信号波形中的高低电平的宽度，不管其如何编码，因此做到了真正的“万能”。也为整个控制器的实现提供了良好的基础。当用户可以通过任意操作方式对这类家电进行控制时，红外插座或开关会向受控电器发送相应的红外遥控指令，从而控制电器的运行状态。

2.5.2 红外学习遥控的实现

经过充分的论证和研讨，本系统采用各设备集中控制的方式实现。集中各设备的方法是首先对各设备的红外遥控信号进行识别并存储，然后在需要时进行还原，以控制对应设备动作。由单片机构成集中控制器，它是自学习与还原的核心部分，红外接收部分由CX20106解调电路或一体化红外接收头组成，发射部分有红外发光管极其驱动部分组成。

通常，红外遥控器将遥控信号调制在38KHz的载波上，经缓冲放大后送至红外发光二极管，转化为红外信号发射出去。二进制脉冲码的形式有多种，其中最为常用的是PWM码(脉冲宽度调制码)和PPM码(脉冲位置调制码)。PWM码以宽脉冲表示1，窄脉冲表示0。PPM码脉冲宽度一样，但是码位的宽度不一样，码位宽的代表1，码位窄的代表0。

遥控编码脉冲信号（以PPM码为例）通常由引导码、系统码、系统反码、功能码、功能反码等信号组成。引导码也叫起始码，由宽度为9ms的高电平和宽度为4.5ms的低电平组成（不同的遥控系统在高低电平上有一定的区别)，用来标志遥控编码脉冲的开始。系统码也叫识别码，它用来指示遥控系统的种类，以区别其他遥控系统，防止各种遥控系统的误操作。功能码也叫指令码，它代表了相应的控制功能，接收机中的微控制器可根

据功能码的数值完成各种功能操作。系统反码与功能反码是系统码与功能码的反码，反码的加入是为了能在接收端校对传输过程中是否产生差错。为了提高抗干扰性能和降低电源消耗，将上述的遥控编码脉冲对频率为38KHz（周期为26.3ms）的载波信号进行脉幅调制（PAM），再经缓冲放大后送到红外发光管，将遥控信号发射出去。

由于遥控器的二进制编码脉冲有一定的宽度，而且它的高低电平均不断的交替变化，因此容易让我们想到一种容易且方便的方法，脉宽测量。虽然它的二进制脉冲的高低电平的宽度有所不同，但它们大都是毫秒级的，因而，采用单片机的定时器来测量它的脉冲宽度，然后存储，还原是完全可以实现的。实际证明这种思路是可行的，而且电路简单，容易实现。

3 硬件电路设计

3.1 相关芯片及模块简介

3.1.1 MCU SM8952AC25P 简介

SM8952C25的最高主频为25MHz,内带8KB闪存的MCU，SM8951/8952系列产品是内嵌4/8K字节闪存的8 位单片微控制器它具有多达32个I/O口其4K/8K的闪存既可作程序存储空间也可以作数据存储空间或程序数据混合空间这些硬件特征和其强大的指令系统和其自带可编程看门狗使它能应用于不同的场合，因此是一种通用的和性能价格比高的控制器，SM8951/8952允许用户还可以通过置位SCONF寄存器的位0(ALE1)来降低EMI，其中的看门狗定时器(WDT)是1个16位自运行计数器在计数器溢出时会产生复位信号。WDT对那些易受噪声干扰电压波动或放电现象影响的系统很有用在程序跑飞或死机的情况下，WDT可以使用户程序脱离不正常状态。WDT不同于8052系统的定时器0定时器1和定时器2。通过软件周期性的清除WDT计数器的值可以防止WDT产生复位信号。片上闪存可以使用商用编程器进行编程。

3.1.2 双音多频收发器 MT8870 简介

MT8870 是一种带呼叫进展过滤器的单片双音多频收发器。它包括一个带增益可调放大器的 DTMF 接收器和一个 DTMF 发送器。其中滤波电路采用高频群和低频群两个六阶开关电容带通滤波器，解码采用数字计数器技术来确定输入的 DTMF 音调的频率，并将其译成标准的四位二进制码。发送器采用开关电容 D/A 变换器。片内使用了一个脉冲计数器，能合成精确的音调脉冲，保证音调脉冲准确的定时发送。MT8870 提供了一个标准的微处理器总线接口，可以直接与 MCS-51 系列 MCU 和微机接口。它还可以选用呼叫进展方式工作，通过呼叫进展滤波器来检测特定通带内的信号频率，供微处理机或计数器电路分析，以确定检测到的呼叫进展音的性质。它从接收

端接收来自电话机的双音多频脉冲信号该双音多频信号先经其内部的拨号音滤波器，滤除拨号音信号，然后经前置放大后送入双音频滤波器，将双音频信号按高，低音频信号分开，再经高，低群滤波器，幅度检测器送入输出译码电路，经过数字运算后，在其数据输出端输出相对应的8421码。

3.1.3 ISD2500系列单片语音录放简介

美国ISD公司的2500芯片，按录放时间60秒、75秒、90秒和120秒分成ISD2560、2575、2590和25120四个品种。ISD2500系列和1400系列语音电路一样，具有抗断电、音质好，使用方便等优点。它的最大特点在于片内E2PROM容量为480K(1400系列为128K)，所以录放时间长；有10个地址输入端(1400系列仅为8个)，寻址能力可达1024位；2500系列最多可分为600段，只要在分段录/放音操作前(不少于300纳秒)，给地址A0~A9赋值，录音及放音功能均从设定的起始地址开始，录音结束由停止键操作决定，芯片内部自动在该段的结束位置插入结束标志（EOM）；而放音时芯片遇到EOM标志即自动停止放音，设有OVF（溢出）端，便于多个器件级联。

3.1.4 固态继电器(SSR)简介

继电器是本系统的执行机构，本系统选用D4810型固态继电器，固态继电器(SSR)与机电继电器相比，是一种没有机械运动，不含运动零件的继电器，但它具有与机电继电器本质上相同的功能。SSR是一种全部由固态电子元件组成的无触点开关元件，他利用电子元器件的点，磁和光特性来完成输入与输出的可靠隔离，利用大功率三极管，功率场效应管，单项可控硅和双向可控硅等器件的开关特性，来达到无触点，无火花地接通和断开被控电路。 固态继电器有三部分组成:输入电路，隔离(耦合)和输出电路。安输入电压的不同类别，输入电路可分为直流输入电路，交流输入电路和交直流输入电路三种。有些输入控制电路还具有与TTL/CMOS兼容，正负逻辑控制和反相等功能。固态继电器的输入与输出电路的隔离和耦合方式有光电耦合和变压器耦合两种。固态继电器的输出电路也可分为直流输出电路，交流输出电路和交直流输出电路等形式。交流输出时，通常使用两个可控硅或一个双向可控硅，直流输出时可使用双极性器件或功率场效应管。固态继电器有寿命长、可靠性高、灵敏度高、控制功率小、电磁兼容性好、快速转换、电磁干扰小的优点。D4810型继电器输入电流为5-40mA，电压3—30V，额定输出电流10A，输出电压范围宽为20-220V，满足项目的要求。

3.2 远程控制电路设计

3.2.1 振铃检测电路

振铃检测电路如图 3.2-1 所示，图中二极管有 2 种作用：

（1）将不确定的线路供电正负变为固定的正负输出；

（2）将交流的振铃信号变为脉动直流以供检测。

当没有振铃信号时，线路上的供电电压为 48V（部分交换机为 60V），经四个二极管构成的全桥整流后，不足以使 62V 稳压管导通，振铃信号输出端电压接近 0V，当振铃信号到来时，线路上的 90V 交流振铃信号经全桥整流变换后 90V 的脉动直流电，其峰值足以击穿耐压值为 62V 稳压管，经电阻 R1 给 U1 提供电压，从光电耦合器输出的波形是时通时断的方波方波信号，可以直接输出至单片机的中断输入口， CPU 可以根据振铃信号光电转换后的高低电平检测有无振铃。其中通过光电耦合一次侧的输入及的电流为：

$$I_{\max} = \frac{U\sqrt{2} - U_D - U_G}{R_1}$$

（3-1）

$$I_{\max} = \frac{90 \times \sqrt{2} - 62 - 0.7}{5100_{\ 1}}$$

$$I_{\max} = 0.013 \ A$$

即 R1 取 5.1K 满足设计要求。

电路中光电耦合 U1 隔离了振铃信号和单片机的直接连接，光电耦合器以光电转换原理传输信息，它不仅使信息发出端（一次侧）与信息接收并输出端（二次侧）是绝缘的，从而对地电位差干扰有很强的抑制能力，而且有很强的抑制电磁干扰能力。保护单片机也提高了稳定性。

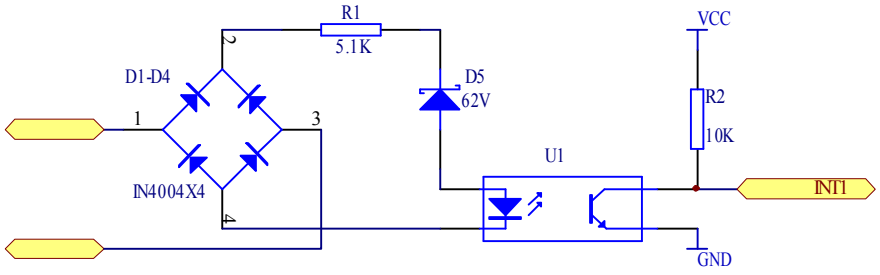


图 3.2-1 振铃检测电路

3.2.2 模拟摘挂机电路

设计主要思路：

根据 ITU 及国家标准规定：不论任何电话机，摘机状态的直流电阻应≤300Ω，有“R”键的电子电话机的摘机状态直流电阻应≤350Ω。在挂机状态下，其漏电流≤5μA。当用户摘机时，电话机通过叉簧接上约 250Ω 的

负载，使整个电话线回路流过约 30mA 的电流。交换机检测到该电流后便停止铃流发送，并将线路电压变为十几伏的直流，完成接续。

模拟摘挂机电路如图 3.2-2 所示。平时电话挂机时，两条电话线处于开路状态，两电话线的电压为 48V（部分电话为 60V），加到电话机的振铃电路两端，当摘机时振铃电路断开，两电话线接通，阻值大约 250Ω。当拨打电话号码时，来自电话线的高压振铃信号经铃流检测电路，通过光电耦合器在其二次侧形成方波脉冲信号送到单片机进行振铃脉冲进行计数，当振铃次数达到设定次数时，由单片机内部软件程序控制，控制摘挂机口输出一个低电平，然后电平送到三极管 PNP1 的基极，使三极管饱和此时，+5V 电源经三极管，再通过继电器线圈接地，继电器线圈得电，使继电器的常开触电闭合，250Ω 的电阻接入电路当中，电话接通。当用户输入密码错误，或者是操作结束后，系统由软件控制 PNP1 截止，继电器线圈失电，常开触点断开，电话线又处于开路状态，从而实现模拟挂机。

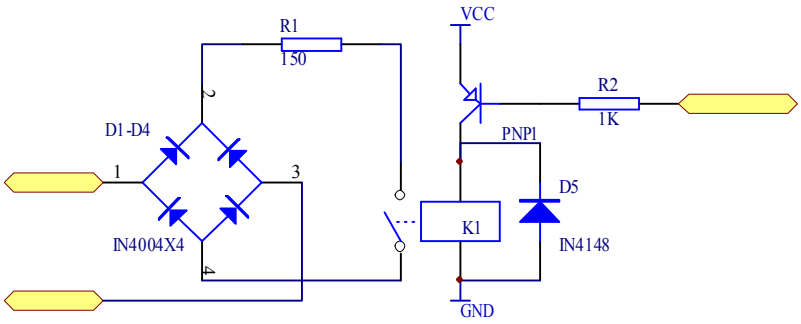


图 3.2-2 模拟摘挂机电路

3.2.3 双音频解码电路

DTMF（Dual Tone Multi Frequency）双音多频信号解码电路是目前在按键电话（固定电话、移动电话）、程控交换机及无线通信设备中广泛应用的集成电路。它包括 DTMF 发送器与 DTMF 接受器，前者主要应用于按键电话作双音频信号发送器，发送一组双音多频信号，从而实现音频拨号。双音多频信号是一组由高频信号与低频信号叠加而成的组合信号，CCITT 和我国国家标准都规定了电话键盘按键与双音多频信号的对应关系如表 1 所示：电话远程控制系统采用 MITEL 公司生产的 MT8870 DTMF 接受器作为 DTMF 信号的解码核心器件。MT8870 主要用于程控交换机、遥控、无线通信及广播系统，实现 DTMF 信号的分离滤波和译码功能，输出相应 16 种频率组合的四位并行二进制码。MT8870 具有拨号音抑制和模拟信号输入可调功能，所以在设计 MT8870 DTMF 解码电路时，只需外加一些阻容元件即可。

表 1　电话键盘与 DTMF 频率对应关系表

高频 低频	1209Hz	1336Hz	1477Hz	1633Hz
697Hz	1	2	3	A
770Hz	4	5	6	B
852Hz	7	8	9	C
941Hz	*	0	#	D

本系统的双音多频 DTMF 信号解码电路由 MT8870 主要承担。MT8870 的连线如图 3.2-3 所示，其的 2、3 脚接收来自电话机的双音多频脉冲信号该双音多频信号先经其内部的拨号音滤波器，滤除拨号音信号，然后经前置放大后送入双音频滤波器，将双音频信号

按高，低音频信号分开，再经高，低群滤波器，幅度检测器送入输出译码电路，经过数字运算后，在其数据输出端（11~14 脚）输出相对应的 8421 码。MT8870 的数据输出端 Q4 ~ Q1 连到单片机，单片机识别 4 位代码。电话按键与相应译码（Q4~Q1）输出。其中，A，B，C，D 4 个按键常被当作 R/P，REDIAL，HOLD，HANDSFREE 等功能使用。为了使单片机及时获取有效数据，MT8870 的 CLD 有效端经反相后接 CPU 的 INT0 引脚。当 MT8870 获取有效双音多频信号后，CLD 电平由低变高，再反相为低，CPU 检测后，指示输入口接收有效二进制代码。而无效的双音频信号（电话线路杂音、人们的语音信号等）是不会引起 MT8870 的 CLD 端变化的。DTMF 接收器的外围电路如图 3.2-3 所示。

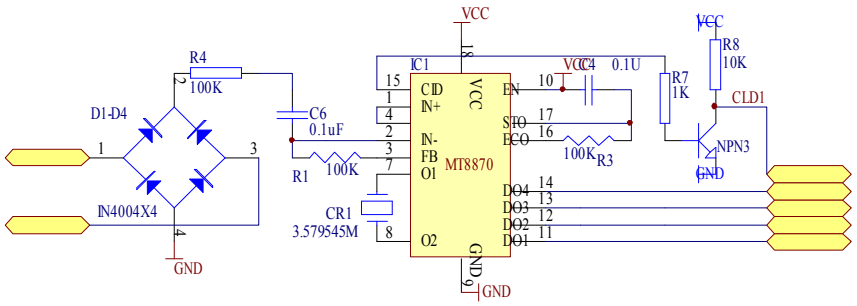


图 3.2-3 双音频解码电路

3.2.4 语言提示电路

ISD2560 芯片采用多电平直接模拟量存储专利技术，每个采样值直接存储在片内单个 E2PROM 单元中，不需另加 A/D 或 D/A 变换来存储和重放，能够非常自然、真实地再现语音，音乐，音调和效果声，避免了一般固体录音电路因量化和压缩造成的量化噪声和"金属声"，因此，作为目前国内外较为理想的语音芯片，在许多领域得到了广泛地应用。语音芯片音质的优劣、功能的强弱决定了报站器的语音效果和性能。如图 3.2-4 所示，语音控制电路由单片机完成。其中，系统中单片机 P0 端口为单片机与语音芯片的通讯口。ISD2560 中，模拟电源(VCCA)和数字电源(VCCD)使用不同的电源总线，分别引到外封装上；模拟地(VSSA)和数字地(VSSD)也使用不同的地线。

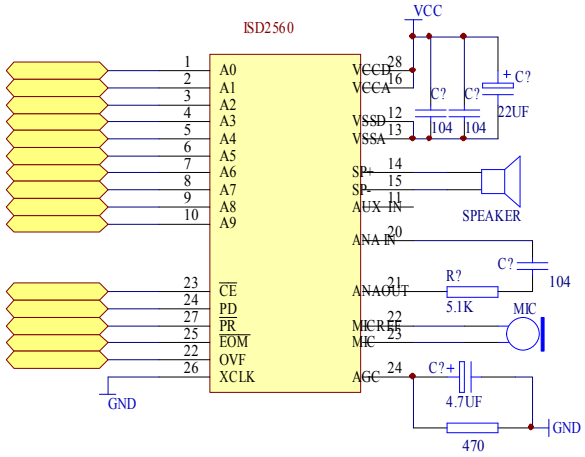


图 3.2-4 语言录放电路

3.3 电源电路设计

3.3.1 5V 开关电源稳压器电路

由于本控制系统单元电路较多对且对 5V 电源的要求比较高，其中 TC35 模块的突发耗电电流峰值可达 2.5A，故外加的稳压器件必须达到足以提供 TC35 和其他电路额定电流的条件。在本系统中，采用了开关电源芯片 LM2576 完成从 12V 到 5V 的转换，作为对 TC35 终端和其他 5V 单元的供电。必须特别注意的是，如图 3.3-1 由 LM2576 芯片完成开关电源转换需要大功率的电感(100uH)和电容，以提高储能的能力，达到单元电路的耗电需求。LM2576 为 5.0V3A 开关电源稳压器。

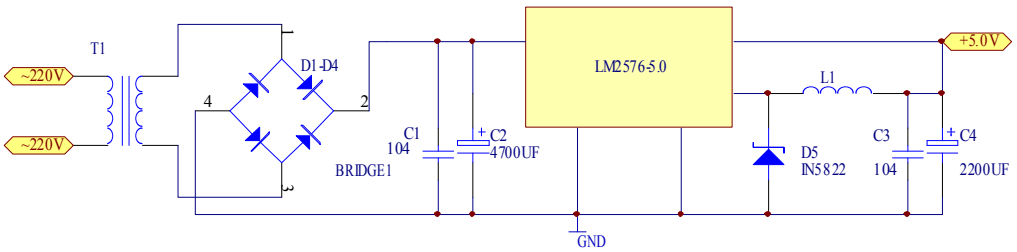


图 3.3-1　5V 开关电源稳压器电路

3.3.2 其他电源稳压器电路

本系统的设计还涉及到传感器信号采集与信号放大，其中放大器等部分需要用到±12V 电源，因此根据设计要求增加了 7812/7912 三端稳压 IC 提供的±12V 电源，如图 3.3-2 由 220V 的交流电经交流变压器 T 降压，变成 12V 交流电，低压交流电再经过全桥整流变成脉动直流 17V 左右为三端稳压提供电源，脉动直流电经 4700u 电解电容和 0.1u 瓷片电容滤波，送到稳压块输入端，进行稳压，然后由三端稳压输出端输出±12V 直流电压，±12V 直流电再由 2200u 和 0.1u 电容进行滤波，输出比较稳定的±12V 直流电压，给模块供电源。

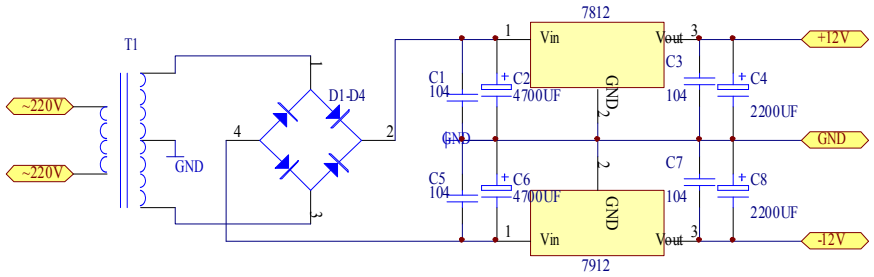


图 3.3-2　±12V 电源稳压器电路

3.4 TC35 短消息模块电路设计

3.4.1 TC35 短消息模块接口电路

TC35 短消息模块是 RS-232C 标准接口，RS-232C 标准（协议）的全称是 EIA-RS-232C 标准，其中 EIA(Electronic Industry Association)代表美国电子工业协会，RS（recommeded standard）代表推荐标准，232 是标识号，C 代表 RS232 的最新一次修改（1969），在这之前，有 RS232B、RS232A。它规定连接电缆和机械、电气特性、信号功能及传送过程。目前在 PC 机上的 COM1、COM2 接口就是 RS-232C 接口。TI 公司的 MAX3238 芯片如图 3.4-1 供电电压为 3～5.5V，符合 TIA/EIA-232-F 和 ITU v.28 标准。具有独特的±15KV 人体静电保护措施，兼容 5V 逻辑输入，内含 3 路接收、5 路发送串行通信接口，最大数据传输速率可达 250 kbps。该芯片的最大特点是，在串行口无数据输入的情况下，可以灵活的进行电源管理，即当 FORCEON(13 脚)为低电平、/FORCEOFF(14 脚)为高电平时，Auto-Powerdown Plus 功能有效。在正常运行模式下，约 30 秒事件内若芯片在接收和发送引脚没有检测到有效信号，将自动进入 Powerdown 模式，此时耗电 1uA。如果 FORCEON 和/FORCEOFF 引脚均为高电平，那么 Auto-Powerdown Plus 功能失效。在 Auto-Powerdown Plus 功能有效的时，如果检测到接收或发送引脚有信号输入，该芯片自动被激活，转入正常工作状态。如果任一接收通道的输入电

压高于 2.7V 或小于-2.7V，或者位于-0.3V～0.3V 的时间小于 30uS，则/INVALID(15 脚)引脚为高电平(数据有效)。如果所有接收通道的输入电压位于-0.3V～0.3V 的时间大于 30uS，则/INVALID(15 脚)引脚为低电平(数据无效)。 该芯片的以上特性，满足了 TC35 作为移动终端的接收和发送电路连接要求。

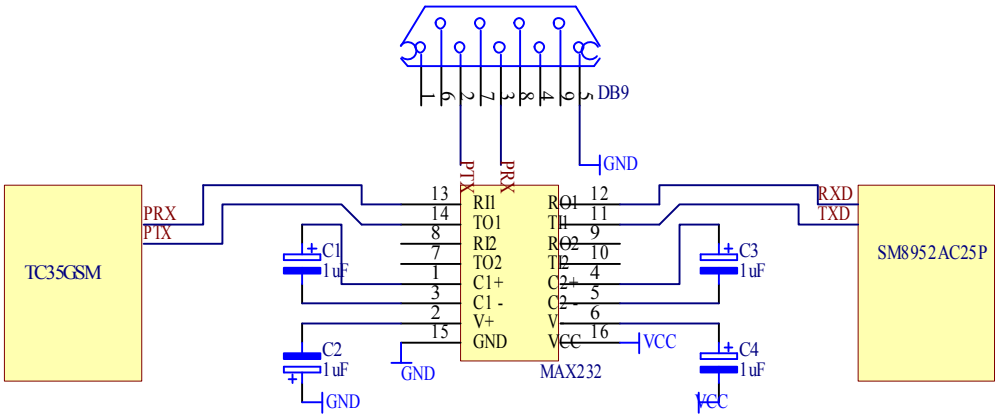


图 3.4-1　RS232 接口电路

3.4.2 TC35 短消息模块控制设计

SM8952AC25P 与 TC35 之间通过异步串行接口进行通信，通信速率为 9600b / s,具有 1 位起始位、8 位数据位、1 位停止位，无奇偶校验。由于 TC35 的数据接口工作在 CMOS 电平,接口电平为 3.3V～5.5V,而 SM8952AC25P 单片机串口工作在 TTL 电平，工作电压范围一般比 TC35 宽，故应在 SM8952AC25P 和 TC35 之间加电平转换电路和电平限制电路。若无电平转换,则有可能使智能模块的性能不太稳定。ZIF 连接器给 SIM 卡接口提供 6 个引脚,其中 CCIN 用来检测 SIM 卡是否插好,其他 5 个引脚分别为 ccvcc(电源,2．9v)、CCGND(地)、CCRST(复位)、CCCLK(时钟)和 CCIO(数据)。ZIF 连接器的 SYNC 脚控制灯的状态，以此判断 TC35 的工作状态。系统加电后，为使 TC35 进入工作状态，必须给 IGT 加一延时大于 100ms 的低脉冲，电平下降持续时间不可超过 lms；启动后，IGT 应保持高电平(3.3V)：驱动 IGT 时,TC35 的供电电压不能低于 3.3V．否则 TC35 不能被激活。

3.5 红外学习遥控电路设计

3.5.1 红外学习遥控接收电路设计

所有红外遥控器的输出都是用编码后的串行数据对 38kHz～40kHz 的方波进行脉冲幅度调制而产生的。如果直接对已调波进行测量，而其脉宽只有 20 多微秒，由于单片机的指令周期是微秒级，会产生很大的误差。因此,先要对已调波进行解调，对解调后的波形进行测量。将 CX20106 或一体化红外接收头解调出的遥控编码脉冲直接连入 SM8952AC25P 单片机的 INT0 和

T0 脚，定时器 T0 和 T1 都初始化为定时工作方式 1，T0 的 GATE 位置位。每次外部中断首先停止定时，记录 T0、T1 的计数值，然后将 T0、T1 的计数值清零，并重新启动定时。T0 的值即为高电平脉宽，T1～T0 的值为低电平脉宽。T0、T1 与红外编码信号脉宽的对应关系并且存储到外部储蓄器 24C256 中等待发送调用。

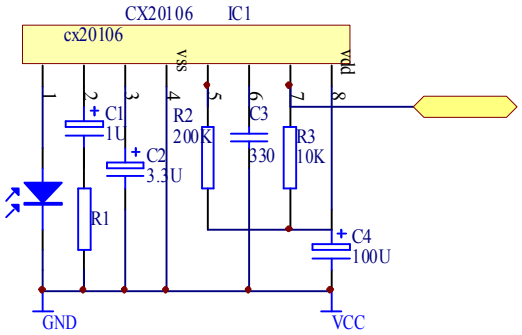


图 3.5-1　红外学习遥控接收电路设计

3.5.2 红外学习遥控发送电路设计

遥控信号的还原是通过 SM8952AC25P 的 T2 特殊输出口进行二进制脉冲码的高电平与低电平的调制输出，其中调制为利用单片机特殊功能进行内部调制这也是本设计的一个创新点，调制后的信号如图 3.5-2 驱动红外发光管工作。该设计的硬件电路相对简单，因此系统的调试重点在软件上。

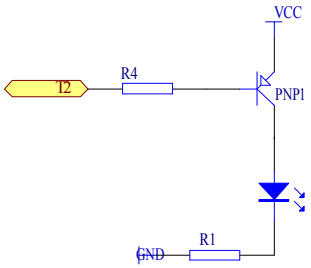


图 3.5-2　红外学习遥控发送电路设计

4　软件部分

4.1　下位机编程

4.1.1 主控单片机系统软件设计

主控单元部分由于任务多、对可靠性的要求高，本系统的主控部分软件设计为分时操作系统控制，采用 RTX51 Tiny 版操作系统。作为嵌入式系统主控单元的单片机，其软件一般是一个微观的实时操作系统，是为某种应用而专门设计的。系统程序有实时过程控制或实时信息处理的能力，要求能够及时响应随机发生的外部事件并对该事件做出快速处理。分时操作系统是把 CPU 的时间划分成长短基本相同的时间区间，即“时间片”，通过操作系统的管理，把这些时间片依次轮流地分配给各个用户使用。如果

某个作业在时间片结束之前，整个任务还没有完成，那么该作业就被暂停下来，放弃 CPU，等待下一轮循环再继续做。此时 CPU 又分配给另一个作业去使用。由于计算机的处理速度很快，只要时间片的间隔取得适当，那么一个用户作业从用完分配给它的一个时间片到获得下一个 CPU 时间片，中间有所“停顿”；但用户察觉不出来，好像整个系统全由它“独占”似的。分时操作系统主要具有以下 3 个特点：①多路性。用户通过各自的终端，可以同时使用一个系统。②及时性。用户提出的各种要求，能在较短或可容忍的时间内得到响应和处理。③独占性。在分时系统中，虽然允许多个用户同时使用一个 CPU，但用户之间操作独立，互不干涉。

RTX51 Tiny 可以配置成使用 round-robin 多任务。Round-robinp 容许 quasi-parallel 执行多任务。任务并不是连续执行的，而是分时间片执行的（本系统采用 10 毫秒作为时间片，由 RTX51 Tiny 把时间片分配给各个任务)。由于时间片的时间很短，因此任务看起来像连续执行一样。

在本系统中 RTX51 Tiny 首先执行信号采集 job0 的任务 0。本函数创建了另一个任务报警分析 job1。信号采集执行完它的时间片后，RTX51 Tiny 开始执行报警分析 job1，这个函数又创建了另一个任务短信发送 job2。如此类推，最后执行完它的时间片后，RTX51 Tiny 又返回到 job0 开始执行。然后再切换到 job1，如此循环。

```
void job0 (void) _task_ 0

{

   os_create (1);

   while (1)

   {

    job0（）;

    }

 }


void job1 (void) _task_ 1

 {
```

```
os_create (2);
while (1)
 {
  job1 ( ) ;
  }
}
```

4.1.2 远程控制程序设计

远程控制软件设计主要分为系统初始化、振铃检测计数、控制摘挂机、双音频信号分析处理、控制电器、信号音提示等部分。下面， 为整体流程图：

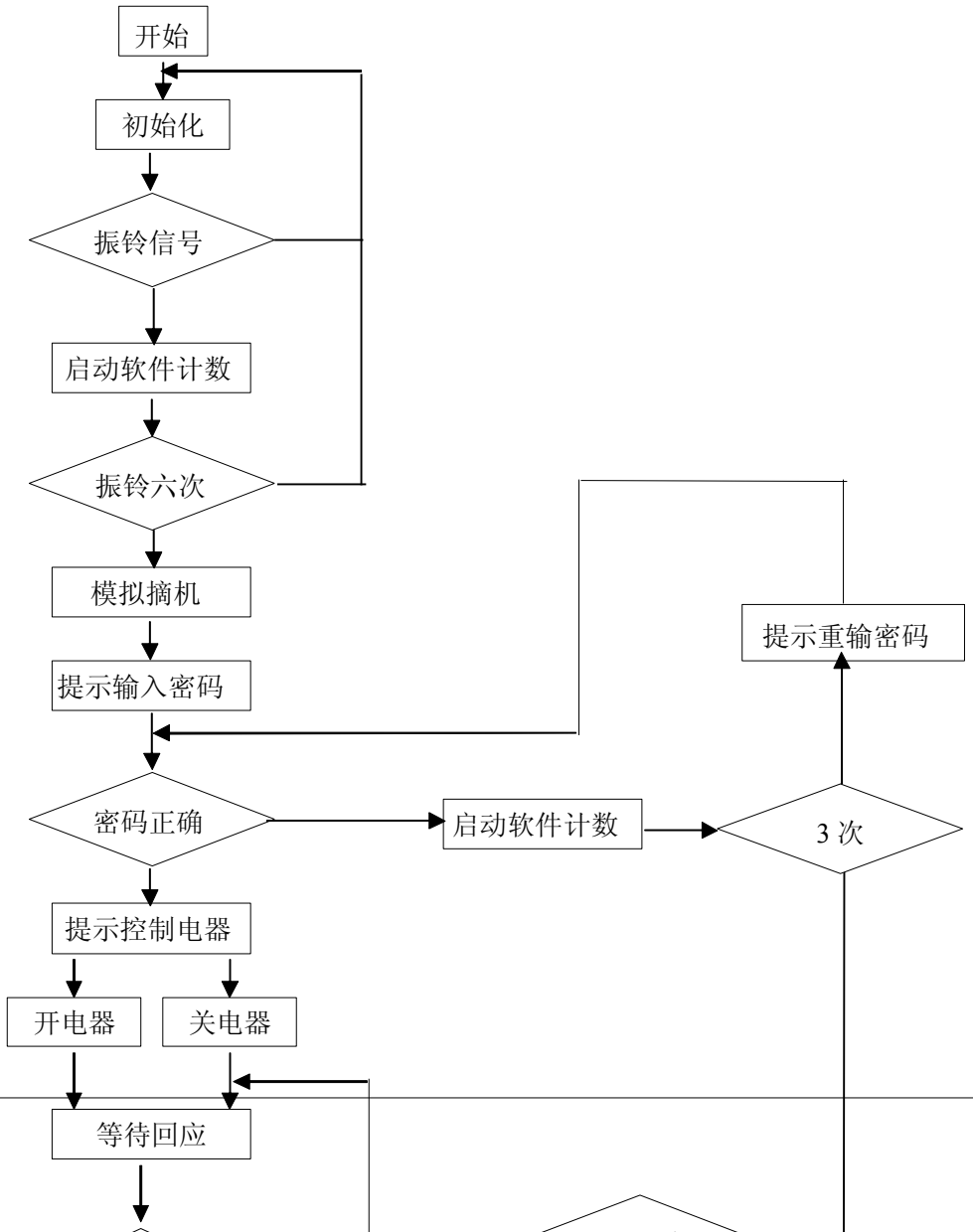


图 4.1-1 远程控制程序设计流程图

4.1.3 短信息发送程序设计

TC35 模块支持 8 位数据位,无奇偶校验位,位停止位数据传输。传输速率可以在 4.8kbit/s 到 115kbit/s 间自适应。对 TC35 模块控制,IGT 信号非常地重要,只有正确的 IGT 信号才可以使 TC35 模块正常地运行。IGT 的下降沿启动 TC35,并且 IGT 的低电平应该至少保持 100ms 如图 4.1-2,TC35 然后正常工作。

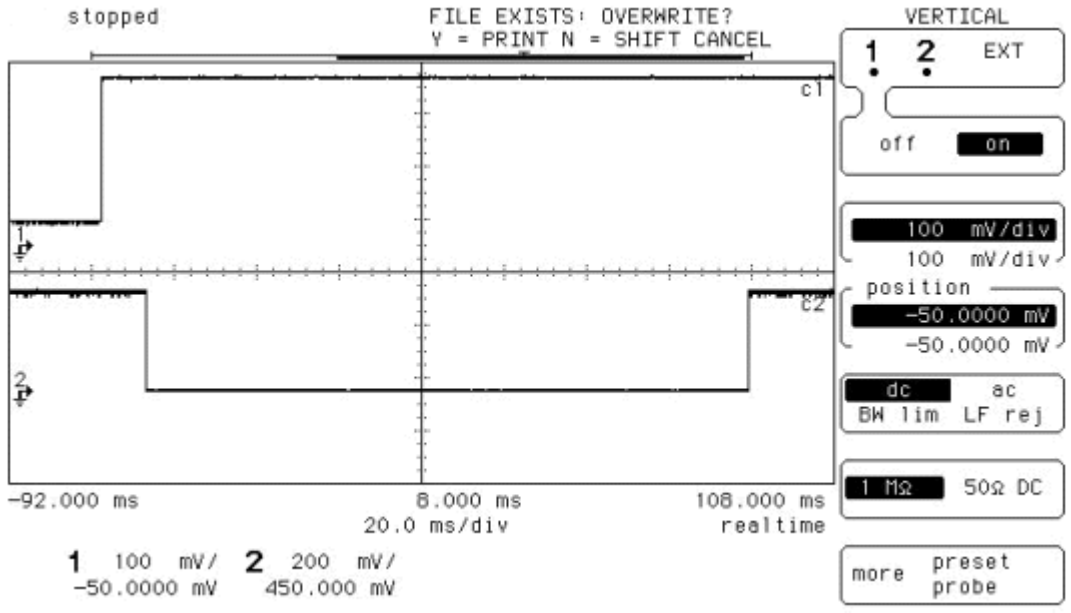


图 4.1-2 存储示波器采集到的启动信号

单片机对 TC35 模块的控制 TC35 模块采用 AT 贺氏指令。单片机可以通过正确的 AT 指令对 TC35 模块进行初始化和短消息的接收发送。对短消息的控制共有三种模式:Block 模式!PDU 模式和 Text 模式。使用 Block 模式需要手机生产厂家提供驱动支持。目前,PDU 模式已取代 Block 模式,而 Text 模式不支持中文,因此本系统使用 PDU 模式进行短消息的发送。单片机通过以下系列 AT 指令对短消息进行控制。

TC35 初始化:首先设置短消息发送格式 AT+CMGF=1<CR>,设置 1 代表 PDU 模式,<CR>是回车符号,也就是 0x0d。指令正确则模块返回<CRLF>OK<CRLF>,<CRLF>是回车换行符号。其次设置短消息中心 AT+CSCA=d+8613800531500d(短消息中心)<CR>,设置正确则模块返回<CRLF>OK<CRLF>。注意短消息中心号码可能会因不同手机或不同区域而不同。如果读取短消息服务中心则使用命令 AT+CSCA=?<CR>,模块应该返回<CRLF>+CSCA:d8613800531500d<CRLF>。最后设置短消息到达自动提示 AT+CNMI=1,1,0,0,1<CR>,设置正确则模块返回<CRLF>OK<CRLF>。设置此命令可使模块在短消息到达后向单片机发送指令<CRLF>+CMTI:dSMd,IN2DEX(信息存储位置)<CRLF>。

发送短消息在 PDU 模式,如果发送短消息,则首先发送短消息数据的长度。AT+CMGS=<length><CR>。等待 TC35 模块返回 ASCII 字符/>0,则可以将 PDU 数据输入,PDU 数据以<Z>(也就是 0x1a)作为结束符。短消息发送成功,模块返回<CRLF>OK<CRLF>。发送数据格式例如,需要发送汉字“一氧化碳的浓度超标,排气扇已打开”到手机 13878305396,则首先发送数据串 AT+CMGS=19<CR>,然后等待 ASCII 字符/>0,然后输入 PDU 数据。

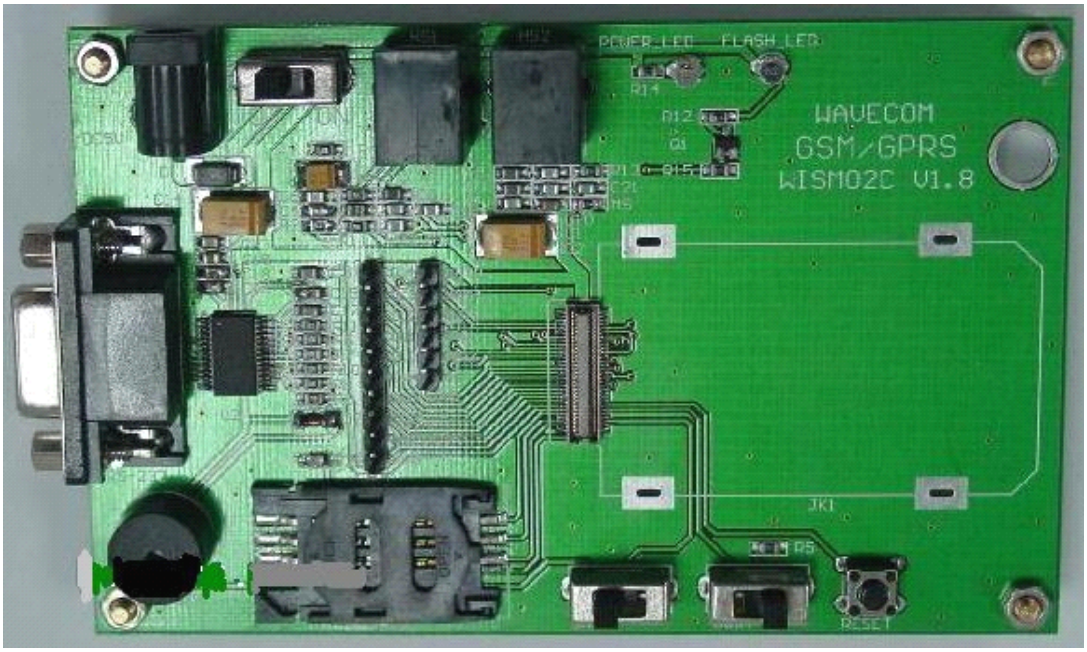


图 4.1-3 TC35 模块 PCB 实物图

4.1.4 红外学习遥控程序设计

设计的主要思路是通过对红外二进制脉冲的宽度进行测量，从而获得红外遥控的波形信息，进而实现存储、还原。根据遥控信号编码和发射过程，遥控信号的识别（也叫解码过程）是去除 38KHz 载波信号后识别出二进制脉冲码中的 0 和 1。红外接收头的解调信号送到 INT0 引脚，由内部定时器完成高低电平长度的采集，然后进行数据保存，由单片机进行高电平与低电平的宽度的测量。遥控信号的还原是通过 P1.0（T2）输出二进制已调脉冲。对遥控信号还原的调制在单片机的内部进行，利用了定时器 2 的捕获工作方式，定时器 2 是一个 16 位定时/计数器。它既可当定时器使用，也可作为外部事件计数器使用，其工作方式由特殊功能寄存器 T2CON 的 C/T2 位选择。定时器 2 有三种工作方式：捕获方式，自动重装载（向上或向下计数）方式和波特率发生器方式，工作方式由 T2CON 的控制位来选择，在捕获方式下，通过 T2CON 控制位 EXEN2 来选择两种方式。如果 EXEN2＝0，定时器 2 是一个 16 位定时器或计数器，计数溢出时，对 T2CON 的溢出标志 TF2 置位，同时激活中断。如查 EXEN2＝1，定时器 2 完成相同的操作，而当 T2EX 引脚外部输入信号发生 1 至 0 负跳变时，也出现 TH2 和 TL2 中的值分别被捕获到 RCAP2H 和 RCAP2L 中。另外，T2EX 引脚信号的跳变使得 T2CON 中的 EXF2 置位，与 TF2 相仿，EXF2 也会活中断。本系统正是利用了捕获方式的 RCAP2H 和 RCAP2L 自动重装载，产生 38K 调制信号，准备还原的 0，1 信号的调制通过 TF2 标志进行调制，该方案不仅合理的利用硬件资源，而且还提高了系统的可靠性。

4.2 上位机（PC 机）编程

本系统为了给用户提供一个良好的人机对话界面还利用 RS232 通信标准还特别设计了与 PC 机软件，传输波特率为 57600 kbps。

4.2.1 用户界面的设计

用户界面是一个应用程序最重要的部分，对用户而言，界面就是应用程序，用户感觉不到幕后正在执行的代码。不论花多少时间和精力来编制和优化代码，应用程序的可用性仍然在很大程度上依赖于界面的好坏。首先针对本系统要开发的应用程序做出初步规划设计，在设计用户界面时，参照了 Microsoft 和其他公司的一些应用程序，使用了通用的设计方案，比如：工具栏、状态条、工具提示、上下文菜单以及标记对话框等。

本设计用户界面如图图 4.2-1。

图 4.2-1 系统用户界面

4.2.2 串行通信的实现

利用 VB 开发通信程序主要有两种方法，一是利用 VB 本身提供的控件，另一种是利用 Windows API 应用程序接口，Windows API 主要提供了三个动态链接库 KERNEL.EXE、USER.EXE、GDI.EXE 供开发人员调用，其中 KERNEL.EXE 主要包括一些底层操作函数，如通信、菜单、消息等以及绝大多数非显示函数，GDI.EXE 图形设备接口库，主要内容为与设备输出有关的函数。和串口通信有关的函数均在 Windows\System 子目录下的 USER.EXE 动态链接库中。在本设计中用 VB 控件实现通信的方法比调用 SDK 的 API 动态链接库的方法更加快捷，且用较少的代码可以实现相同的功能，因此本设计使用此控件来完成串口的通信。

4.2.3 控件 MSComm 使用方法

MSComm 控件通过串行端口传输和接收数据，为应用程序提供串行通讯功能。MSComm 控件在串口编程时非常方便，程序员不必去花时间去了解较为复杂的 API 函数，而且在 VC、VB、Delphi 等语言中均可使用。 Microsoft Communications Control（以下简称 MSComm）是 Microsoft 公司提供的简化 Windows 下串行通信编程的 ActiveX 控件，它为应用程序提供了通过串

行接口收发数据的简便方法。具体的来说，它提供了两种处理通信问题的方法：一是事件驱动(Event－driven)方法，一是查询法。

MSComm 控件提供下列两种处理通讯的方式：事件驱动方式和查询方式。

①事件驱动方式

事件驱动通讯是处理串行端口交互作用的一种非常有效的方法。在许多情况下，在事件发生时需要得到通知，例如，在串口接收缓冲区中有字符，或者 Carrier Detect(CD)或 Request To Send (RTS)线上一个字符到达或一个变化发生时。在这些情况下，可以利用 MSComm 控件的 OnComm 事件捕获并处理这些通讯事件。OnComm 事件还可以检查和处理通讯错误。所有通讯事件和通讯错误的列表，参阅 CommEvent 属性。在编程过程中，就可以在 OnComm 事件处理函数中加入自己的处理代码。这种方法的优点是程序响应及时，可靠性高。每个 MSComm 控件对应着一个串行端口。如果应用程序需要访问多个串行端口，必须使用多个 MSComm 控件。

②查询方式

查询方式实质上还是事件驱动，但在有些情况下，这种方式显得更为便捷。在程序的每个关键功能之后，可以通过检查 CommEvent 属性的值来查询事件和错误。如果应用程序较小，并且是自保持的，这种方法可能是更可取的。

MSComm 控件有很多重要的属性，常用的如下。

CommPort 设置并返回通讯端口号。

Settings 以字符串的形式设置并返回波特率、奇偶校验、数据位、停止位。

PortOpen 设置并返回通讯端口的状态。也可以打开和关闭端口。

Input 从接收缓冲区返回和删除字符。

Output 向传输缓冲区写一个字符串。

下面分别描述。

CommPort 属性　设置并返回通讯端口号。

语法　object.CommPort[value]（value 一整型值，说明端口号。）

说明　在设计时，value 可以设置成从 1 到 16 的任何数（缺省值为 1）。但是如果用 PortOpen 属性打开一个并不存在的端口时，MSComm 控件会产生错误 68（设备无效）。必须在打开端口之前设置 CommPort 属性。

RThreshold 属性　在 MSComm 控件设置 CommEvent 属性为 comEvReceive 并产生 OnComm 之前，设置并返回的要接收的字符数。

语法　object.Rthreshold [= value]（value 整型表达式，说明在

产生 OnComm 事件之前要接收的字符数。）

说明　当接收字符后，若 Rthreshold 属性设置为 0（缺省值）则不产生 OnComm 事件。例如，设置 Rthreshold 为 1，接收缓冲区收到每一个字符都会使 MSComm 控件产生 OnComm 事件。

CTSHolding 属性　确定是否可通过查询 Clear To Send(CTS)线的状态发送数据。Clear To Send 是调制解调器发送到相联计算机的信号，指示传输可以进行。该属性在设计时无效，在运行时为只读。

语法　object.CTSHolding（Boolean）

说明　如果 Clear To Send 线为低电平(CTSHolding=False)并且超时时，MSComm 控件设置 CommEvent 属性为 comEventCTSTO(Clear To Send Timeout) 并产生 OnComm 事件。

Clear To Send 线用于 RTS/CTS(Request To Send/Clear To Send)硬件握手。如果需要确定 Clear To Send 线的状态，CTSHolding 属性给出一种手工查询的方法。

SThreshold 属性　MSComm 控件设置 CommEvent 属性为 comEvSend 并产生 OnComm 事件之前，设置并返回传输缓冲区中允许的最小字符数。

语法　object.SThreshold[=value]

value 整形表达式，代表在 OnComm 事件产生之前在传输缓冲区中的最小字符数。

说明　若设置 Sthreshold 属性为 0（缺省值），数据传输事件不会产生 OnComm 事件。若设置 Sthreshold 属性为 1，当传输缓冲区完全空时，MSComm 控件产生 OnComm 事件。如果在传输缓冲区中的字符数小于 value，CommEvent 属性设置为 comEvSend，并产生 OnComm 事件。comEvSend 事件仅当字符数与 Sthreshold 交叉时被激活一次。例如，如果 Sthreshold 等于 5，仅当在输出队列中字符数从 5 降到 4 时，comEvSend 才发生。如果在输出队列中从没有比 Sthreshold 多的字符，comEvSend 事件将绝不会发生。

PC 上位机程序见附录。

5　系统制作及调试

本系统的制作调试主要分为硬件调试、软件调试和联机调试等三大部分。

经过初步的分析设计后，在设计制作硬件电路的同时，调试穿插进行，应用系统的硬件调试和软件调试是分不开的，许多硬件故障是在调试软件时才发现的。但通常是先排除系统中明显的硬件故障后才和软件结合起来调试，如此有利于问题的分析和解决，不会造成问题的积累，从而可以节

约大量的调试时间。软件编程中，我是首先完成单元功能模块的调试，然后进行系统调试，整体上采用硬件调试的调试方法。联机调试是最重要的一部分，同时也是本系统成功的关键。

5.1 使用的仪器仪表及工具

TOSHIBA A10 移动 PC 一台；
TKS66S 单片机仿真器一台；
TDS210 60MHz 双踪示存储波器一台；
WYK—302Bz 型直流稳压电源一台；
TC-108H“多路通”程控交换机一台；
HA8188(9)P/T 双音多频电话机一台；
HA119(6)P/T 双音多频电话机一台；
MODEL HC—F1000C 型频率计一台；
EE1641B1 型函数发生器/计数器一台；
MF 47 型机械万用表一个；
DT 9208 型数字万用表一个；
YEAR 200 型体育竞赛秒表一个；
TLW-T 调温烙铁一把；
Keil 开发软件一套；
Protel 开发工具一套；
Visual Basic 6.0 中文版工具一套；

5.2 硬件制作与调试

5.2.1 系统 PCB 板的设计

PCB 即印刷电路板，是电子电路的承载体。在现代电子产品中，几乎都要使用 PCB。PCB 板的设计是电路设计的最后一个环节，也是对原理电路的再设计。因此 PCB 板的设计是理论设计到实际应用一个十分重要的内容。印制电路板(PCB)是电子产品中电路元件和器件的支撑件. 它提供电路元件和器件之间的电气连接。PCB 设计的好坏对抗干扰能力影响很大. 因此，在进行 PCB 设计时. 必须遵守印制电路板设计原则和抗干扰措施的一般原则，并应符合抗干扰设计的要求。本次设计采用 Altium 公司 PROTEL 系列设计完成 SCH 到 PCB 的设计，并且手工完成电路焊接以及整机的装配。

5.2.2 系统硬件调试

本系统的硬件调试分为以下阶段进行调试：

(1)逻辑错误调试

样机硬件的逻辑错误是由于设计错误和加工过程中的工艺性错误所造成的。这类错误包括：错线、开路、短路等几种，其中短路是最常见的故

障。

(2)器件调试

元器件失效的原因有两个方面：一是器件本身已损坏或性能不符合要求；二是由于组装错误造成的元器件失效，如电解电容、二极管的极性错误，集成块安装方向错误等。

(3)可靠性调试

引起系统不可靠的因素很多，如金属化孔、接插件接触不良会造成系统时好时坏；内部和外部的干扰、电源纹波系数过大、器件负载过大等造成逻辑电平不稳定；另外，走线和布局的不合理等也会引起系统可靠性差。

(4)电源故障

若样机中存在电源故障，则加电后将造成器件损坏。电源的故障包括：电压值不符合设计要求，电源引出线和插座不对应，电源功率不足、负载能力差。

在本次调试在调试样机加电之前，先用万用表和示波器,根据硬件电气原理图和装配图仔细检查样机线路的正确性，并核对元器件的型号、规格和安装是否符合要求。还特别注意电源的走线，防止电源之间的短路和极性错误，并重点检查扩展系统总线是否存在相互间的短路；或其它信号线的短路，由于本设计的印刷电路板布线密度较高，出现了两处因工艺原因造成短路，短路点用刻刀断开。

对于样机所用的电源事先做了单独调试，调试好后，检查其电压值、负载能力、极性等均符合要求，然后加到系统的各个部件上。在不插片子的情况下，加电检查各插件上引脚的电位，仔细测量各地点电位是否正常,还特别注意单片机插座上的各点电位是否正常，防止了联机时会损坏仿真器。

5.3 软件及联机调试

5.3.1 主控程序调试

软件调试与所选用的软件结构和程序设计技术有关。本系统采用模块程序设计技术，逐个模块调好以后，再进行系统程序总调试。由于采用了实时多任务操作系统，采用是逐个任务进行调试，下面进一步予以说明。在调试第一个任务时，同时也调试相关的子程序、中断服务程序和操作系统的程序。等逐个任务调试好以后，再使各个任务同时运行，在本次调试中操作系统中没有错误，在单步和断点调试后，进行了连续调试，因为单步运行只能验证程序的正确与否，而不能确定定时精度、CPU 的实时响应等问题。等全部完成后，反复运行多次，除了观察稳定性之外，还观察了用户系统的操作是否符合设计要求的操等，部分程序作了适当修正后系统能

够正常运行。

5.3.2 短消息发送调试

监控软件在平时不断检测各报警点的信号，当有异常情况时，系统通过 TC35 模块自动发出报警信息 ，在 TC35 初始化之前要用定时器延时约 5s，等待 TC35 自检完毕，然后检查 SIM 卡，如果检查到无 SIM 卡，系统就会调用提示出错程序；有 SIM 卡则继续检查移动运营商，之后再对 TC35 进行初始化，主要是用 AT 命令初始化发送方式、设置短信中心号码和登录网络的测试。特别需要注意的是：不能给 SIM 卡设置开机密码，否则不能正常登录到 GSM 网络，还有在收到短消息命令后必须先判断是否是手机预设号码，如果是就处理，否则删除。由于 GSM 网络有较好的安全及保密性，所以在软件设计时不需要考虑安全方面的问题，监控软件在外界干扰强烈的情况下有可能跑飞，为了使跑飞的程序恢复正常，可采用定时计数器 T1 来完成软件看门狗的功能. 定时时间设置为监控软件完成 1 次全过程时间的 3 倍。TC35 采用 AT 命令，单片机可以通过正确的 AT 指令对 TC35 进行初始化和短消息的接收与发送。对短消息的控制有三种模式：Block 模式、PDU 模式和 Text 模式。使用 Block 模式需要手机生产厂家提供驱动支持，目前已被 PDU 模式所取代；Text 模式比较简单，可以实现数字和字符的直接收发，但 Text 模式不支持中文；PDU 模式是将 GB2312 的中文编码转换为 Unicode 编码，容易实现中文编解码。本设计主要传送中文信息，本系统为了编程方便，使用 PDU 模式完成短消息的发送。

6 结论

本次毕业设计根据设计任务，提出了并且论证了设计方案，详细地阐述了电话远程控制原理、GSM 短消息发送的实现方法、以及相关电路的设计原理，设计中充分利用了系统的硬件和软件资源，实现了各个模块的协调控制，提高了系统的可靠性和通用性。

原理样机经过设计方案论证，设计了相应的硬件电路和系统软件，制作了电路原理样机并进行单机调试和与 PC 机联机调试，结果表明，所设计的电路和软件能完成基本的测试功能。

本系统中的电话远程控制，关键在于利用标准程控交换信令结合软件编程，实现了语音界面及安全认证机制，其中 GSM 短消息平台充分借助于 GSM 网络的短消息业务实现了短消息远程报警，具有投资少、成本低、可靠性高等特点，还具有良好可扩展性和实用价值，符合了未来家电的智能化、网络化发展方向。

本设计完成的工作超出了任务书中规定的设计任务。系统配合学习遥

控功能，解决了控制空调等红外遥控电器难的问题，并且取得了较好的效果。符合家居智能化系统是创造一个舒适的生活环境设计理念。在制作原理样机后，由于时间限制，还没有进行长时间可靠性和实际安装测试，这是系统产品化必须做的工作。

本系统还可以应用于工农业生产中，实现对无人值守岗位的远程控制和安全报警等。

谢　辞

本设计从开题到方案的设计和具体电路试验的实施始终是在导师许敏老师和郝卫东老师的精心指导和周密安排下进行的。感谢他们长期在学习和生活中给予我的帮助，使我受益非浅，同时，培养了我处理问题和我解决问题的能力。

此外，感谢童有为老师和孙安青老师多年来对我的关心和帮助，也感谢我在参加各类比赛的队友在合作的过程中给我信心和勇气，使我跨过一道道难关，和积累了很多实际经验。感谢在大学期间关心和支持我的所有老师和朋友。

最后，感谢在百忙之中给我审稿的诸位老师。

参考文献

[1] 朱世华.程控数字交换原理与应用.西安：西安交通大学出版社，1993.
[2] 李延文.中文版Visual Basic 6.0控件高级编程.北京：人民邮电出版社，2002年.
[3] 万福君.单片微机原理系统设计与开发.合肥：中国科技大学出版社，1995.
[4] 谢自美.电子线路设计.实验.测试.华中科技大学出版社，2000.
[5] 胡大可.基于单片机8051的嵌入式开发指南：电子工业出版社，2001.
[6] DALAS Semiconductor.Atomatic Identification Data Book[M].1995.
[7] 蓝贤芳.新型电话机的使用、原理与维修.广州：广州科技出版社，1994.
[8] Siemens TC35/TC37 Hardware Interface Description Vision 03.10.
[9] AT Command Set for TC35, TC37 and TC35 Terminal 03.10.
[10] Siemens AG Developer′s Guide SMS with the sms PDUmo de 1997.

[11] 王琴放.张凡.单片机原理及应用[M].北京:电子工业出版社:1997.8:3~203.
[12] 徐顺成.实用电子技术与电子产品汇编.北京：电子工业出版社，1993.
[13] 郝建国.赵英杰.通用集成电路大全.北京：人民邮电出版社，1997.
[14] 沙占友.集成化智能传感器原理与应用[M].北京：电子工业出版社.2004.1：198~222.
[15] 徐爱钧.彭秀华.单片机高级语言C51应用程序设计.北京：电子工业出版社,1999.
[16] 刘艳玲.采用MAX232实现MCS-51单片机与PC机的通信[J].天津理工学院学报，1999，15（2）：57～61.
[17] 何利民.MCS-51系列单片机应用系统设计系统配置与接口技术[M].北京：北京航空航天大学出版社，2003.

附录 1

下位机调试程序

```
#include <AT89X55.H>

bit r_flag;

enum                                    eepromtype
    {M2401,M2402,M2404,M2408,M2416,M2432,M2464,M24128,M24256};
```

```
extern bit      RW24XX(unsigned char *DataBuff,unsigned char
    ByteQuantity,unsigned int Address,
                    unsigned char ControlByte,enum eepromtype
    EepromType);


sbit key   = P2^0;
bit key_flag;

sbit dog   = P0^7;
sbit LED   = P0^0;
sbit TELA   = P3^6;
sbit TEL  = P3^7;
sbit RING  = P3^2;
sbit switching =P0^4;


unsigned char mode;
bit passwordflag;
unsigned char password[6];
unsigned char passwordtest[6];

void dlms(unsigned int x);
void reset(void);
void open(void);

//*************************************************************
    **  //
//              INT1中断服务程序
    //
//*************************************************************
    **  //
unsigned char ring_fluctuation;
unsigned char ring_count;
bit ring_flag;
```

```
void service_int0() interrupt 0 using 1
{
 if(ring_fluctuation<10)
   {
    ring_fluctuation++;
    }
 }
//*****************************************************************
     **  //
//                        t1 定时中断
     //
//*****************************************************************
     **  //
unsigned char ring_dlsm;
unsigned char ring_time;
unsigned int ring_reset_time;
unsigned char dlms_time;
void timer1 (void) interrupt 3 using 0
{
  TH1=0x3c;
  TL1=0xb0;
  dlms_time++;
  if(ring_fluctuation>5)
    {
      EX0=0;//INT1_OFF;
      ring_dlsm++;
      if(ring_dlsm==20)//等待一秒
        {
         ring_dlsm=0;
         EX0=1;//INT1_ON;
        ring_fluctuation=0;
      ring_count++;//震铃次数

       if(ring_flag)//有效振铃
         {
```

```
        if(ring_count==6)
          {
           ring_count=7;
              mode=1;
           }
        }
     else//无效振铃
       {
        if(ring_count>2)
          {
           ring_reset_time=800;
           }
        }
       }
  }


if((ring_count==2)|(ring_count==1))
  {
    ring_time++;
    if(ring_time==200)
      {
    ring_flag=1;
    }
    }
if((ring_count>0)&(ring_count<6))
  {
   LED=INT0;
    }
  else if(passwordflag)
   {
    LED=~LED;
     }

if(ring_count>0)
```

```
    {
      ring_reset_time++;
      if(ring_reset_time==1100)//55s
        {
        reset();
      LED=0;
         dlms(100);
      LED=1;
         dlms(100);
      LED=0;
         dlms(100);
      LED=1;
         dlms(100);
      LED=0;
         dlms(100);
      LED=1;
         dlms(100);
      LED=0;
         dlms(100);
      LED=1;
         dlms(100);
      LED=0;
         dlms(100);
      LED=1;
         dlms(100);
      LED=0;
         dlms(100);
      LED=switching;
      }
      }

}

//*************************************************************
    **  //
```

```
//                  INT1中断服务程序
     //
//*****************************************************************
     **  //

unsigned char DTMF;
void service_int1() interrupt 2 using 1
{
 EX1=0;
 DTMF=P1;
 DTMF=DTMF>>4;
}



void main(void)
{
  TMOD=0x11;
  TH1=0x3c;
  TL1=0xb0;

  TR1=1;
  ET1=1;

  IT0=1;
  EX0=1;

  IT1=0;
  EX1=1;




  reset();
  EA=0;
```

```
r_flag=RW24XX(password,6,0x0000,0xa1,M2402);//R

if(P2_0==0)
  dlms(20);
   if(P2_0==0)
    {
     ring_flag=1;
      mode=1;
     passwordflag=1;
      ring_count=7;
      ring_fluctuation=6;
      }
EA=1;
while(1)
    {
      open();
      dog=~dog;
     switch(mode)
          {
          case 0:;

                  break;
          case 1:
              TEL=1;
              LED=0;//
              dlms_time=0;
             ring_reset_time=0;
                  while(dlms_time<40)
                      {
                      dlms(1);
                       TELA=~TELA;
                       }
             ring_reset_time=0;
              LED=1;//
             TELA=0;
```

```
    mode++;
         DTMF=255;
         EX1=1;
         LED=switching;
         break;
case 2://password
       if(DTMF!=255)
      {
      passwordtest[0]=DTMF;
      if(INT1==1)
        {
        dlms_time=0;
        while(dlms_time<2);
        if(INT1==1)
          {
           DTMF=255;
           mode++;
             dlms_time=0;
           while(dlms_time<10)
                       {
                       dlms(2);
                        TELA=~TELA;
                        }
          EX1=1;
              TELA=0;
           }
        }
      }
         break;
case 3://password
       if(DTMF!=255)
      {
      passwordtest[1]=DTMF;
      if(INT1==1)
        {
```

```
            dlms_time=0;
            while(dlms_time<2);
            if(INT1==1)
              {
               DTMF=255;
               mode++;
                 dlms_time=0;
               while(dlms_time<10)
                           {
                           dlms(2);
                            TELA=~TELA;
                            }
              EX1=1;
              TELA=0;
               }
            }
          }
             break;
      case 4://password
            if(DTMF!=255)
          {
          passwordtest[2]=DTMF;
          if(INT1==1)
            {
            dlms_time=0;
            while(dlms_time<2);
            if(INT1==1)
              {
               DTMF=255;
               mode++;
                 dlms_time=0;
               while(dlms_time<10)
                           {
                           dlms(2);
                            TELA=~TELA;
```

```
                        }
          EX1=1;
          TELA=0;
           }
        }
      }
         break;
case 5://password
       if(DTMF!=255)
      {
      passwordtest[3]=DTMF;
      if(INT1==1)
        {
        dlms_time=0;
        while(dlms_time<2);
        if(INT1==1)
          {
           DTMF=255;
           mode++;
             dlms_time=0;
           while(dlms_time<10)
                       {
                      dlms(2);
                       TELA=~TELA;
                       }
          EX1=1;
          TELA=0;
           }
        }
      }
         break;
case 6://password
       if(DTMF!=255)
      {
      passwordtest[4]=DTMF;
```

```
        if(INT1==1)
          {
          dlms_time=0;
          while(dlms_time<2);
          if(INT1==1)
            {
             DTMF=255;
             mode++;
               dlms_time=0;
             while(dlms_time<10)
                         {
                        dlms(2);
                         TELA=~TELA;
                         }
            EX1=1;
            TELA=0;
             }
          }
        }
           break;
    case 7://password
          if(DTMF!=255)
        {
        passwordtest[5]=DTMF;
        if(INT1==1)
          {
          dlms_time=0;
          while(dlms_time>2);
          if(INT1==1)
            {
             DTMF=255;
             mode++;
            //EX1=1;
             }
          }
```

```
            }
               break;
        case 8:
            if(passwordflag==0)
             {

if((passwordtest[0]==password[0])&(passwordtest[1]==passwor
d[1])&(passwordtest[2]==password[2])&(passwordtest[3]==pass
word[3])&(passwordtest[4]==password[4])&(passwordtest[5]==p
assword[5]))
               {
                    mode++;
               dlms_time=0;
               }
             else
               {
                    reset();
               }
            }
          else
            {
             dlms_time=0;
                while(dlms_time<10)
                        {
                        dlms(2);
                         TELA=~TELA;
                         }
              TELA=0;
             password[0]=passwordtest[0];
             password[1]=passwordtest[1];
             password[2]=passwordtest[2];
             password[3]=passwordtest[3];
             password[4]=passwordtest[4];
             password[5]=passwordtest[5];
            EA=0;
```

```
r_flag=RW24XX(password,6,0x0000,0xa0,M2402);//W
              EA=1;
              reset();
             }

               break;
        case 9:
               while(dlms_time<20)
                   {
                   dlms(1);
                    TELA=~TELA;
                    }
               while(dlms_time<25)
                   {

                    }
               while(dlms_time<45)
                   {
                   dlms(1);
                    TELA=~TELA;
                    }
           TELA=0;
           EX1=1;
           mode++;
                break;
        case 10:
                    switch(DTMF)
                   {
                   case 1:
                        if(ring_reset_time<1050)
                         {
                           dlms_time=0;
                          switching=0;
                          LED=0;
```

```
                    while(dlms_time<10)
                      {

                       }
                    while(dlms_time<20)
                      {
                      dlms(2);
                       TELA=~TELA;
                       }
                    while(dlms_time<30)
                      {

                       }
                    while(dlms_time<40)
                      {
                      dlms(2);
                       TELA=~TELA;
                       }
          TELA=0;
        DTMF=255;
        EX1=1;
       }

     break;
case 2:
     if(ring_reset_time<1050)
      {
        dlms_time=0;
        switching=1;
      LED=1;
                    while(dlms_time<10)
                      {

                       }
                    while(dlms_time<20)
```

```
                                        {
                                        dlms(4);
                                         TELA=~TELA;
                                         }
                                       while(dlms_time<30)
                                        {

                                         }
                                       while(dlms_time<40)
                                        {
                                        dlms(4);
                                         TELA=~TELA;
                                         }
                                    TELA=0;
                                  DTMF=255;
                                  EX1=1;
                                 }

                              break;
                         case 12:
                              reset();
                              break;

                         }

                    break;
//-------------------------------------------------------------
     ------//

          default:;
//-------------------------------------------------------------
     ------//
          }
        }
```

```
 }

 void reset(void)
 {
       TEL=0;
       TELA=0;
         ring_fluctuation=0;
         ring_count=0;
         ring_flag=0;
         ring_dlsm=0;
         ring_time=0;
         dlms_time=0;
         ring_reset_time=0;
       key_flag=0;
       mode=0;
       EX1=1;
       DTMF=255;
       LED=switching;
       passwordflag=0;
       passwordtest[0]=255;
       passwordtest[1]=255;
       passwordtest[2]=255;
       passwordtest[3]=255;
       passwordtest[4]=255;
       passwordtest[5]=255;
  }
void open(void)
{
 if((key==0)&(key_flag==0)&(passwordflag==0))
    {
      dlms_time=0;
     while(dlms_time<2);
      if(key==0)
         {
          reset();
```

```
        key_flag=1;
        switching=~switching;
        LED=switching;
        dlms_time=0;
        }
    }
 if((key==1)&(dlms_time>10))
   {
         key_flag=0;
    }
 }

void dlms(unsigned int x)
{
     unsigned int i;
     while (x-->0)
     {
         dog=~dog;
      for (i=0;i<163;i++)
          {;}
     }
}




//24cxx

//-----------------------------------------START----------
----------------------------------//
//enum                                                eepromtype
{M2401,M2402,M2404,M2408,M2416,M2432,M2464,M24128,M24256};
//extern  bit     RW24XX(unsigned  char  *DataBuff,unsigned  char
ByteQuantity,unsigned int Address,
```

```
//                          unsigned char ControlByte,enum eepromtype
EepromType);

//  flag=RW24XX(&x,1,0x0000,0xa0,M24256);//W  24C256
//  flag=RW24XX(&x,1,0x0000,0xa1,M24256);//R  24C256

#include <AT89X55.H>
//#pragma  ot(6,SIZE)
#include <intrins.h>
#define  ERRORCOUNT 10

sbit     SDA=P2^7;//对应硬件
sbit     SCL=P2^6;//对应硬件

enum                                                    eepromtype
{M2401,M2402,M2404,M2408,M2416,M2432,M2464,M24128,M24256};
enum  eepromtype EepromType;

/*************************************************************
*********************/
//DataBuff为读写数据输入/输出缓冲区的首址
//ByteQuantity 为要读写数据的字节数量
//Address 为EEPROM的片内地址
//ControlByte  为 EEPROM 的 控 制 字 节 ， 具 体 形 式 为
(1)(0)(1)(0)(A2)(A1)(A0)(R/W),其中R/W=1,
//表示读操作,R/W=0为写操作,A2,A1,A0为EEPROM的页选或片选地址;
//EepromType为枚举变量,需为M2401至M24256中的一种,分别对应24C01至
24C256;
//函数返回值为一个位变量，若返回1表示此次操作失效，0表示操作成功;
//ERRORCOUNT为允许最大次数，若出现ERRORCOUNT次操作失效后，则函数
中止操作，并返回1
/*************************************************************
*********************/
extern  bit      RW24XX(unsigned  char  *DataBuff,unsigned  char
ByteQuantity,unsigned int Address,
```

```
                    unsigned  char  ControlByte,enum  eepromtype
EepromType)
{
void Delay(unsigned char DelayCount);
void IICStart(void);
void IICStop(void);
bit  IICRecAck(void);
void IICNoAck(void);
void IICAck(void);
unsigned char IICReceiveByte(void);
void IICSendByte(unsigned char sendbyte);
unsigned char data j,i=ERRORCOUNT;
bit      errorflag=1;
while(i--)
   {
   IICStart();
   IICSendByte(ControlByte&0xfe);
   if(IICRecAck())
       continue;
   if(EepromType>M2416)
      {
      IICSendByte((unsigned char)(Address>>8));
      if(IICRecAck())
         continue;
      }
   IICSendByte((unsigned char)Address);
   if(IICRecAck())
       continue;
   if(!(ControlByte&0x01))
      {
      j=ByteQuantity;
      errorflag=0;                        //********clr errorflag
      while(j--)
        {
        IICSendByte(*DataBuff++);
```

```
        if(!IICRecAck())
           continue;
        errorflag=1;
        break;
        }
      if(errorflag==1)
        continue;
      break;
      }
   else
     {
     IICStart();
     IICSendByte(ControlByte);
     if(IICRecAck())
       continue;
     while(--ByteQuantity)
       {
       *DataBuff++=IICReceiveByte();
       IICAck();
       }
     *DataBuff=IICReceiveByte();        //read last byte data
     IICNoAck();
     errorflag=0;
     break;
     }
  }
IICStop();
if(!(ControlByte&0x01))
  {
  Delay(255);
  Delay(255);
  Delay(255);
  Delay(255);
  }
return(errorflag);
```

```
}


/*****************以下是对IIC总线的操作子程序***/
/*****************启动总线*********************/
void IICStart(void)
{
SCL=0;                    //
SDA=1;
SCL=1;
_nop_();
_nop_();
_nop_();
SDA=0;
_nop_();
_nop_();
_nop_();
_nop_();
SCL=0;
SDA=1;                    //
}

/*****************停止IIC总线***************/
void IICStop(void)
{
SCL=0;
SDA=0;
SCL=1;
_nop_();
_nop_();
_nop_();
SDA=1;
_nop_();
_nop_();
_nop_();
```

```
SCL=0;
}

/**************检查应答位******************/
bit IICRecAck(void)
{
SCL=0;
SDA=1;
SCL=1;
_nop_();
_nop_();
_nop_();
_nop_();
CY=SDA;                    //因为返回值总是放在CY中的
SCL=0;
return(CY);
}

/***************对IIC总线产生应答******************/
void IICACK(void)
{
SDA=0;
SCL=1;
_nop_();
_nop_();
_nop_();
_nop_();
SCL=0;
_nop_();
SDA=1;
}

/*****************不对IIC总线产生应答***************/
void IICNoAck(void)
{
```

```
SDA=1;
SCL=1;
_nop_();
_nop_();
_nop_();
_nop_();
SCL=0;
}

/*******************向IIC总线写数据*********************/
void IICSendByte(unsigned char sendbyte)
{
unsigned char data j=8;
for(;j>0;j--)
   {
   SCL=0;
   sendbyte<<=1;          //无论C51怎样实现这个操作，始终会使
CY=sendbyte^7;
   SDA=CY;
   SCL=1;
   }
SCL=0;
}

/**********************从IIC总线上读数据子程序**********/
unsigned char IICReceiveByte(void)
{
register receivebyte,i=8;
SCL=0;
while(i--)
   {
   SCL=1;
   receivebyte=(receivebyte<<1)|SDA;
   SCL=0;
   }
```

```
return(receivebyte);
}

/***************一个简单延时程序***********************/
void Delay(unsigned char DelayCount)
{
while(DelayCount--);
}
```

附录 2

PC上位机调试程序

```
Private Declare Function TextToSms Lib "SMSDLL.dll" (ByVal csc As
     String, ByVal ToNum As String, ByVal smsnr As String, ByVal
     flash As Integer, ByVal reportit As Integer, ByRef sms_len As
     Integer, ByVal retSms As String) As Integer
Private Declare Function About Lib "SMSDLL.dll" ()
Private Declare Function SmsToText Lib "SMSDLL.dll" (ByVal sms As
     String, ByVal csca As String, ByRef caca_len, ByVal ToNum As
     String, ByRef ToNum_len As Integer, ByVal sendtime As String,
     ByRef time_len As Integer, ByVal smsnr As String) As Integer



Dim i(6) As Byte 'tl
Dim display() As Byte 'tl
Dim test As Byte 'tl
```

```
Dim j As Byte



Private Sub Command1_Click()
   '将信息编码成一条短消息
   Dim s As String
   Dim s1 As String
   Dim s2 As String
   Dim r As Integer
   Dim f As Integer
   Dim sms_len As Integer
   r = CInt(Check1.Value)
   f = CInt(Check2.Value)

   Dim rsms As String * 400

   If Text3.Text = "" Then
     MsgBox "请输入短消息内容"
     Exit Sub
   End If

   'rsms = "11111"
    ret = TextToSms(Text1.Text, Text2.Text, Text3.Text, r, f,
     sms_len, rsms)
    Text5.Text = "短消息长度:" + CStr(sms_len) & vbCrLf & "PDU内
     容:" + rsms
    MsgBox "总字数:" & ret & vbCrLf & "短消息长度:" + CStr(sms_len)
     + vbCrLf & "PDU内容:" & rsms

'    Dim i
'    i = 98
'     testit s, s1
'     MsgBox s1
End Sub
```

```
Private Sub Command2_Click()
'About
Dim csca As String * 30
Dim csca_len As Integer
Dim num As String * 30
Dim num_len As Integer
Dim sendtime As String * 30
Dim time_len As Integer
Dim nr As String * 300
Dim nr_len As Integer

Dim i As Integer
i = SmsToText(Text4.Text, csca, csca_len, num, num_len, sendtime,
     time_len, nr)
MsgBox "返回值:" & i & vbCrLf & "短消息内容:" & Left(nr, i)
End Sub




Private Sub Command3_Click()

End Sub

Private Sub Command4_Click()
  On Error Resume Next
   Dim comm As Integer '端口号
   comm = GetSetting(App.Title, "setting", "Com", 1)
   MSComm1.CommPort = comm
   Dim s As String
```

```
Dim s1 As String
Dim s2 As String
Dim r As Integer
Dim f As Integer
r = 0
f = 0
Dim sms_len As Integer
If Check1.Value = vbChecked Then
  r = 1
  Else
  r = 0
End If
 If Check2.Value = vbChecked Then
  f = 1
 ' MsgBox "f=1"
  Else
  f = 0
 End If



'MsgBox "r= " + r + "  f= " + f

'Exit Sub
Dim i As Integer
Dim rsms As String * 400

If Text3.Text = "" Then
  MsgBox "请输入短消息内容"
  Text3.SetFocus
  Exit Sub
End If
If Text2.Text = "" Then
  MsgBox "请输入对方手机号码"
  Text2.SetFocus
```

```
    Exit Sub
  End If
  'rsms = "11111"
   ret  =  TextToSms(Text1.Text,  Text2.Text,  Text3.Text,  r,  f,
    sms_len, rsms)
   If ret = 0 Then
     MsgBox ("短消息编码错误")
     Exit Sub
   End If
   Text5.Text = "短消息长度:" + CStr(sms_len) & vbCrLf & "PDU内
    容:" + rsms
   '开始发送
Dim time1

With MSComm1
If .PortOpen = False Then
  .PortOpen = True
End If
If .PortOpen = True Then
   time1 = Now()
   .Output = "at+cmgs=" + CStr(sms_len) + vbCrLf
   s = .Input
   Do Until (InStr(s, ">") > 0 Or DateDiff("S", time1, Now()) >
   5)  '5秒钟内是否能检测到>,如果没有检测到则可能没有接
   TC35modem
     DoEvents
     s = s + .Input
   Loop

   If InStr(s, ">") > 0 Then '如果发送接收到>符 则继续发送内容
    s = ""
    .Output = Left(rsms, ret)  '去掉空字符
    DoEvents
    For i = 1 To 100
    '少等片刻.
```

```
 DoEvents
 Next

 .Output = Chr(26) '发送^Z

ElseIf InStr(UCase(s), "ERROR") > 0 Then '如果发送接收到>符 则继续
  MsgBox "发送出错", vbInformation, "发送出错"
Else
  MsgBox "TC35 MODEM 没有响应，请检查断口是否正确", vbInformation, "发送出错"

End If
time1 = Now()
s = .Input
Do Until (InStr(s, "OK") > 0 Or DateDiff("S", time1, Now()) > 5) '检查是否发送成功
  DoEvents
  s = s + .Input
Loop
If InStr(s, "+CMGS:") > 0 Then
' MsgBox "报警发送"
Else
  MsgBox "指定时间内未检测到发送成功标志," + vbCrLf + "发送可能失败，请检查连接状况或网络状态。", vbInformation, "发送出错"
End If
.PortOpen = False


End If

End With
Exit Sub
```

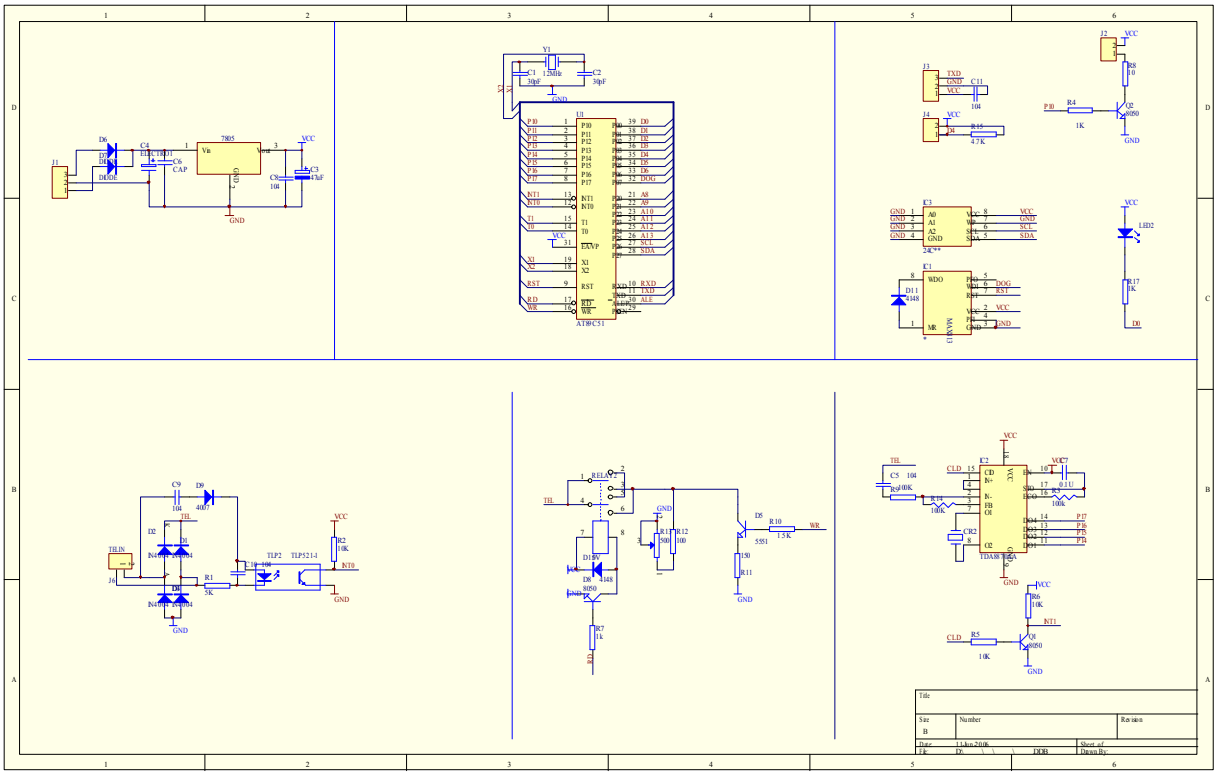
```
err1:

  comm = InputBox("请确认端口号是否正确", "打开端口出错",
     MSComm1.CommPort)
  MSComm1.CommPort = comm
  SaveSetting App.Title, "setting", "Com", comm


End Sub

Private Sub Command5_Click()

  Dim comm As Integer
  comm = InputBox("请输入端口号", "设置通信端口",
     MSComm1.CommPort)
  If MSComm1.PortOpen = True Then MSComm1.PortOpen = False
  MSComm1.CommPort = comm
  SaveSetting App.Title, "setting", "Com", comm
End Sub




Private Sub Form_Load()
Text3_Change
'Text2.Text    =    GetSetting(App.Title,    "setting",    "num",
     "17000165165123")
End Sub

Private Sub Form_Unload(Cancel As Integer)
SaveSetting App.Title, "setting", "num", Text2.Text
End
End Sub
```

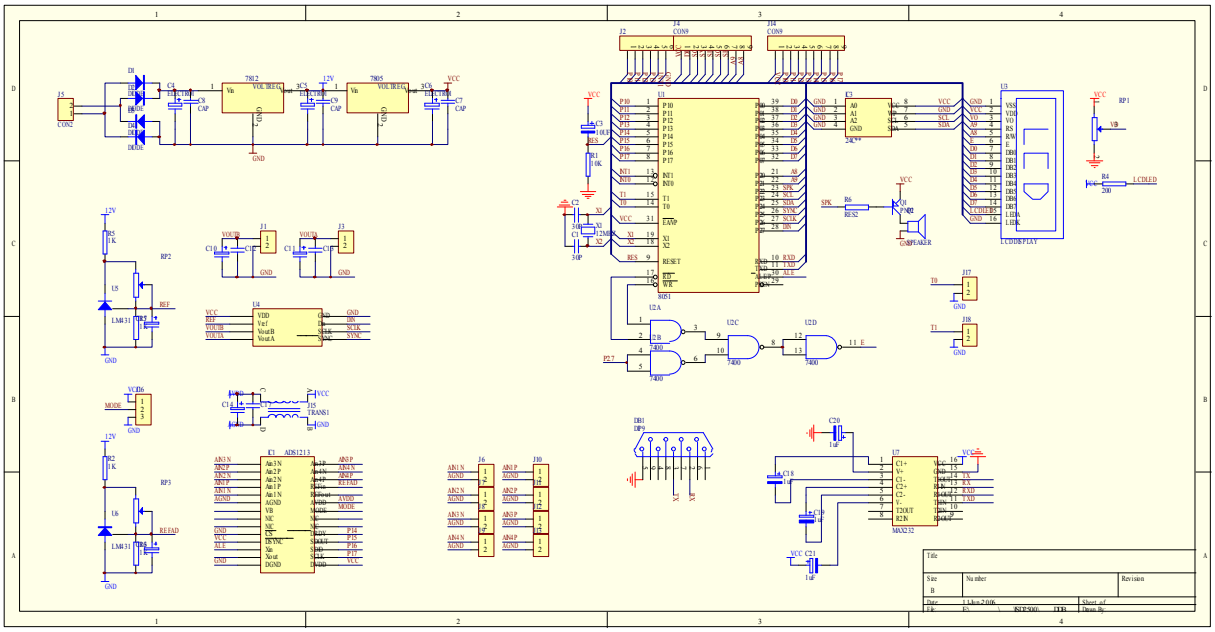
```
Private Sub Frame1_DragDrop(Source As Control, X As Single, Y As
     Single)

End Sub

Private Sub Frame3_DragDrop(Source As Control, X As Single, Y As
     Single) 'tl

End Sub


Private Sub Label6_Click() 'tl

End Sub

Private Sub Label7_Click()

End Sub

Private Sub MSComm2_OnComm() 'tl


End Sub

Private Sub Text3_Change()
Label5.Caption = "字数 : " & Len(Text3.Text)
End Sub

Private Sub Timer1_Timer() 'tl

Label7.ForeColor = RGB(255, 0, 0)

If MSComm2.PortOpen = False Then
        MSComm2.PortOpen = True
   End If
```

```
If MSComm2.InBufferCount > 0 Then
    display = MSComm2.Input
    Label6.Caption = display(0) 'Abs(display(0))
     Label8.Caption = Abs(display(1)) & " " & Abs(display(2)) & "
     " & Abs(display(3)) & " " & Abs(display(4)) & " " &
     Abs(display(5)) & "" & Abs(display(6)) & "" & Abs(display(7))
     & " " & Abs(display(8)) & " " & Abs(display(9))
    If display(0) = 254 Then
        If display(6) = 0 Then
        Text3.Text = "报警"
            Else
         Text3.Text = "提示"
            End If
            For j = 0 To 9
               display(j) = 0
            Next j
        'Command4_Click
        End If
    End If

End Sub
```

附录 3

系统原理图





附录 4

系统PCB图