摘 要

控制器是铁路公寓电脑叫班系统的核心组成部分。目前，国内所使用的控制器均为模拟系统。本文所讨论的数字式控制器的核心部分是数字系统，它采用数字信号处理理论及混合信号微控制器，实现了控制器的数字化，是一个完全技术创新的产品。

本文主要从数字控制器的硬件结构、软件结构、所使用的新技术几个方面分别对控制器的设计进行了阐述。

第一章介绍控制器的功能，以及数字系统相对于模拟系统的优点。第二章和第三章分别从硬件设计和软件设计方面探讨控制器的构成及实现。

本文重点从三个方面讨论了实现控制器的数字化技术。首先，在信号的调制解调模块，主要讨论了混合信号的滤波算法——基于信号幅度特征的时域滤波算法以及控制信号的调制解调技术。基于信号幅度特征的时域滤波算法是一种适合于微控制器的、能从数模混合信号中提取数字信号的时域滤波算法。其次，在系统的控制模块中，主要讨论了软件实现模拟信号的数字化音量控制、模拟信号的多通道数字化切换和 A/D 的动态取样频率的实现技术。最后，研究了在控制器上实现 IAP 功能的技术及实现方法。通过对 C8051FXXX 微控制器的 FLASH 程序存储器的结构和功能的分析，采用中断入口地址的重定位技术，通过 UART 串口实现了 IAP。

在论文的附录中给出了 IAP 功能中主机和端机之间的通信握手协议、软件实现 UART 的相关源代码、软件实现模拟信号的数字化音量控制的源代码以及数字式控制器的整体结构。

关键词：数字式控制器，滤波，IAP，数字化，动态取样，软件模拟

Abstract

Controller is the important component of Apartment Building Call System. The controller which is used now is analog system. The core of the digital controller discussed in this paper is a digital system, it make analog controller become a digital one by using digital signal processing theory and mixed signals MCU, it is completely technological innovation.

The paper mainly narrated design of the digital controller from several respects: hardware structure、software structure and new technologies used.

In chapter 1, the functional of digital controller and advantages of using digital system were narrated than using analog system. Chapter 2 and chapter 3 respectively introduced the composing of digital controller from two respects: hardware design and software design.

Three respects were mainly emphasized in this paper to narrate development digital technology. Firstly, in module of signals modulation and demodulation, we mainly discussed the mixed signals filter algorithms— the time domain filter algorithms based on signal range features and coding technology of control signals. The time domain filter algorithms based on signal range features is fit for MCU, it could extract digital signals from digital and analog mixed-signals exactly. Secondly, in system control module, we narrated software simulation of digital volume control of analog signals、multi-center digital switch of analog signals and A/D dynamic sampling frequency technology. At the end, the technology

and implementation way of IAP were introduced on MCU. After researching the structure and functional of C8051FXXX MCU, using the interruption vector re-positioning address technology, we made IAP available really with UART.

In the appendix of paper, the communications handshake agreement between the PC and terminals, source codes of software simulation of digital volume control of analog signals、source codes of the software simulation of UART and the whole structure chart of digital controller were listed.

Keywords: digital controller, filter, IAP, digital, dynamic sample, software simulation

独创性声明

本人声明，所呈交的学位论文是我个人在导师指导下进行的研究工作及取得的研究成果。尽本人所知，除了文中特别加以标注和致谢的地方外，论文中不包含其他人已经发表或撰写过的研究成果，也不包含为获得北京交通大学或其他教学机构的学位或证书而使用过的材料。与我一起工作的同志对本研究所做的任何贡献已在论文中作了明确的说明并表示了谢意。

本人签名：张亮生

日期：2006年3月___日

关于论文使用授权的说明

本人完全了解北京交通大学有关保留、使用学位论文的规定，即：学校有权保留送交论文的复印件，允许论文被查阅和借阅；学校可以公布论文的全部或部分内容，可以采用影印、缩印或其他复制手段保存论文。论文中所有创新和成果归北京交通大学计算机与信息技术学院所有。未经许可，任何单位和个人不得拷贝。版权所有，违者必究。

本人签名：张彦生

日期：2006年3月___日

第 1 章 绪论

1.1 引言

近代科学技术的高度发展，尤其是计算机技术的突飞猛进，影响了许多领域。各个领域的工作设备与计算机技术相结合来提高工作效率，降低工作强度成为了顺理成章的事情。铁路系统作为一个国家的重要部门，结合高科技技术实现管理的信息化，高效率运作，是势在必行的。上世纪八十年代，铁道部便开始着手实现铁道系统的信息技术化。铁路公寓的叫班工作原来一直是人工进行的，必须靠人工来按时呼叫车乘工作人员，工作效率低、强度大。所以，结合计算机技术实现自动化叫班是一个理想可行的方法。

铁路公寓电脑叫班系统运用当前最先进的微机管理技术及现代高科技的通讯手段，结合现代化公寓的管理模式，研制而成。具有自动化程度高、操作简单、系统可靠性高、叫班准确快速、语音清晰宏亮，电脑自动录放音等特点，大大减轻了公寓管理人员的劳动强度。为乘务人员正点出乘提供了可靠的保障，是计算机技术与语音技术的完美结合。

目前，国内有多家企业生产提供公寓电脑叫班系统，如北京完美科学技术研究所、郑州畅想自动化设备有限公司、北京弘扬电子有限公司、中国浙江鼎泰科技有限公司等多家高新技术企业提供成熟的产品。这些产品已经被广泛应用于全国铁路系统的各个铁路段、分局。如北京弘扬电子公司生产的公寓电脑叫班系统，已在全国多个铁路分局安装使用。铁道系统部门反映良好，一致评价：其性能稳定、功能

全面、容易使用。

根据调查，目前国内所有的公寓电脑叫班系统的一个重要组成部分——控制器，是完整的模拟系统。模拟系统中各元器件都有一定的温度系数，且电平是连续变化的，易受温度、噪声、电磁感应等的影响。由于铁路公寓基本上位于铁路沿线，环境较差，噪声、电磁感应等影响较大，使得控制器可靠性不高。

为此，结合当前电子硬件和软件技术发展，我们研制了数字式控制器 JH2000。该款数字式控制器采用先进的数字信号处理理论，以及目前微控制器市场中流行的混合处理器，实现了公寓电脑叫班系统控制器的完全数字化。从而在不提高价格成本的前提下大幅度提高了整个公寓电脑叫班系统的可靠性，实现了较高的性价比，使得产品在市场上具有较强的市场竞争力。

1.2 控制器功能简介

公寓电脑叫班系统主要由 PC 机软件、控制器、端机三部分组成。其中控制器是整个系统的核心，PC 机所发出的指令以及终端发送到 PC 机的应答命令都是经过控制器处理识别的。如图 1.1 所示：

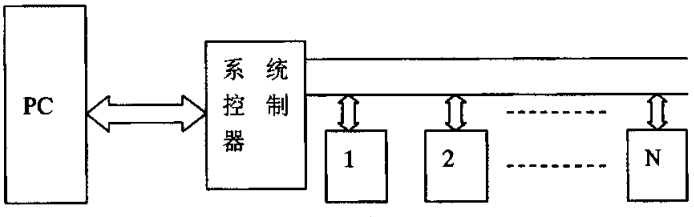


图 1.1 系统结构示意图

在系统中，控制器是 PC 机和终端之间交换命令的关键部件。任

何 PC 机发出的命令，要被终端正确识别并执行，就必须经过控制器的处理，调制为终端能够接收的波形；对于终端发送给 PC 机的信号，同样需要经过控制器的解调，转换为 PC 机能识别的命令。另外，控制器还有控制着系统的外围设备的功能，如电话、PC 机声卡录放音、麦克的声道切换问题以及系统正处于何种命令模式下。所以控制器性能的好坏直接决定了系统性能的优劣。

当前市场上所使用的公寓电脑叫班系统的控制器是采用的模拟信号的处理技术。它的整个系统工作过程中间所处理的信号都是连续变化的模拟信号，是一个完整的模拟系统。在精度、灵活性、以及系统集成化方面也存在着精度不高、灵活性不大、集成度不高的缺点。

数字信号处理技术以及硬件技术的高速发展正好解决了以上问题。所以，我们考虑采用数字信号处理理论，将系统信号由模拟信号转化为数字信号进行处理。数字信号处理与模拟信号处理相比有很多优点，主要表现在以下几个方面：

✓ 精度高：模拟处理系统中元件的精度要达到10^{-3}已不容易，而数字处理系统在采用 17 位字长时即可达到10^{-5}的精度。对于频率分辨力，模拟系统只能做到 1 *Hz*，而数字系统则可做到 2 *MHz*。

✓ 可靠性强：模拟系统中各种元件参数都有一定的温度系数，而会随着环境条件的变化而改变，并且容易出现感应、杂散效应，甚至振荡等。数字系统只有两个信号电平“0”、“1”，因而受周围环境、温度、噪声以及电磁感应的影响都比模拟系统小，所以可靠性强。

✓ 灵活性大：具有不同特性的模拟系统，其系统的各个元件一般都是不相同的。就是说，要改变模拟系统的特性，必须改变系统的各个元件。然而，对于数字系统，只要改变系统存储器中的数据，就

可以改变系统的参数，从而得到具有不同特性的系统。

✓ 便于大规模集成化：数字部件对电路参数的要求不太严格，且具有高度的规范性，因此易于实现大规模的集成化。此外，在低频电路中，它可以取代大电感、大电容，制成比模拟电路体积小、重量轻且价廉的电路。

✓ 可以做到时分复用：数字系统可以按时间顺序同时处理几路独立的信号，这就是所谓的时分复用。当各路输入信号同时输入序列值时，同步系统控制他们在时间上前后错开，由并行变为串行进入处理器。处理器在处理完第一路序列的第一个值之后，再处理第二路序列的第一个值，直至各路信号序列的第一个值都处理一遍，然后经串行变为并行分路输出。如此反复进行，就可通过一个系统处理多路信号。处理器速度越高，它所能处理的信号路数就越多[4]。

以上表明，数字处理有着模拟处理无可比拟的优点。我们可以选用合适的 CPU 来提高系统的集成度；同时，软件方面采用数字信号处理理论，控制器处理过程采用数字化处理。用 A/D 转换器将模拟信号变为数字信号，进行相应的处理。最后，再将需要的语音信号经过 D/A 转换器变为模拟信号。

控制器硬件上采用目前微控制器市场上功能较强，技术成熟，价格比较便宜的混合信号处理器；在软件方面，我们采用数字信号处理理论，利用数字信号的高可靠性实现系统的数字化。为了提高系统集成度，减小控制器体积，我们采用软件模拟技术实现一部分硬件的功能，实现系统的集成化和维护的简单化。

第 2 章 系统硬件设计

2.1 模拟式控制器硬件体系结构

在确定数字式控制器的硬件设计方案时，我们首先研究了现有的模拟式控制器的硬件体系结构。以北京弘扬电子公司的系统模拟式控制器为例，整体分为两大部分：调制解调模块和系统控制模块。

2.1.1 调制解调模块硬件体系结构图

调制解调模块示意图如 2.1 所示。

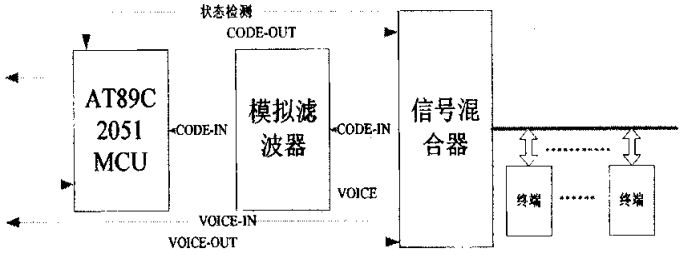


图 2.1 调制解调模块示意图

AT89C2051 是一个 51 内核的 8 位单片机。在调制解调模块中，AT89C2051 是核心功能部件，主要处理系统控制信号，信号为控制信号方波。AT89C2051 根据控制信号方波的频率不同，对控制信号进行识别处理。对从 PC 机发送过来的控制信号，AT89C2051 直接将其送给信号混合器，与系统的语音信号混合后发送到终端；对于终端发送过来的混合命令，先经过高通模拟滤波器，滤除语音信号，再对控制命令进行处理。

模拟滤波器主要对控制信号和语音信号的混合信号进行滤波，滤除混合信号中的低频语音信号。它是一个模拟高通滤波器。

信号混合器为北京弘扬电子公司的研制产品，主要作用是对系统中的语音信号和线路房间控制信号进行调制，在线路上进行传输。因为系统采用信号混合传输的总线结构，对于同时在线路上传输的语音信号和控制信号，我们必须首先经过信号混合器，才能在线路上进行传输。

2.1.2 系统控制模块硬件体系结构图

系统控制模块结构示意图如图 2.2 所示。

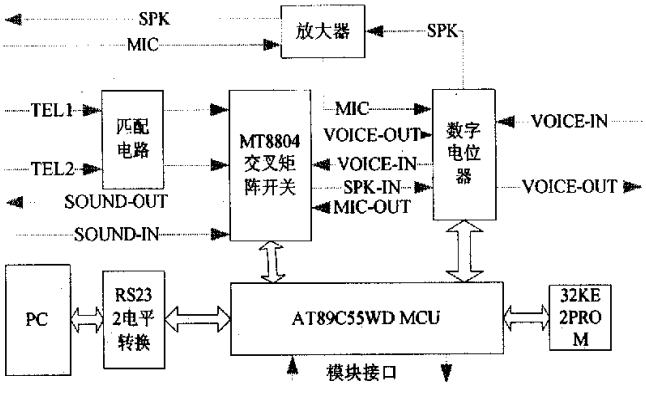


图 2.2　系统控制模块结构示意图

AT89C55WD 是一个 51 内核的 8 位单片机。MT8804 和数字电位器均由 AT89C55WD 控制。AT89C55WD 与 PC 机通过串口连接通信，它从 PC 机接收系统命令，控制 MT8804 和数字电位器切换到相应的工作状态，读取 E^2PROM 上的系统信息，并通过模块接口与调制解调模块 AT89C2051 通信，对命令进行调制后发送到端机。当终

端有信号到来时，AT89C55WD 从 AT89C2051 接收解调后的命令进行相应状态的控制并上传给 PC 机。

MT8804 是一个 8x4 模拟集成开关阵列，控制着系统中的 2 路电话、PC 机的声卡输入信息以及从终端发送过来的语音信息的声道的切换，决定系统工作在各个状态时的声道工作状态，如系统是录音还是放音。

数字电位器主要用来控制声音信号的大小，实现音量的调节。包括从麦克、PC 机声卡和电话各声源来的语音信号。

E^2PROM 是一个 32K 的电可擦写的可编程存储器。它是一个扩展的存储器，在系统中主要用于存储系统信息。

2.2 数字式控制器硬件体系结构

2.2.1 数字系统结构分析

经过对模拟式控制器体系结构的分析，我们要对控制器实现数字化，就必须使用数字系统，但我们处理的信号仍是模拟信号，数字系统不能直接处理模拟信号，所以我们必须首先将模拟信号变为数字信号。完成这种变换作用的器件叫 A/D 转换器。它具有三方面的功能：第一，根据取样定理所要求的速率对模拟信号 $f(t)$ 进行取样，使之变成离散信号 $f[kt]$；第二，把 $f[kt]$ 进行量化变成数字信号 $f[k]$；第三，把数字信号 $f[k]$ 编成一串二进制数码，然后送到数字系统进行处理。数字系统处理后的输出信号仍是一串二进制编码。如果我们希望处理后的输出信号还是模拟信号，尚需再附加一个叫做 D/A 的变换器，它

首先把二进制编码变成数字信号 $y[k]$，然后使其再变成阶梯状的连续信号。最后，可再用一个模拟低通滤波器对阶梯波进行平滑，从而获得模拟输出信号 $y[t]$。以上用数字系统代替模拟系统对模拟信号进行处理的简要过程如图 2.3 所示。

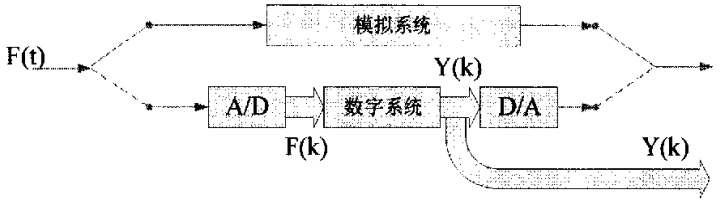


图 2.3 模拟信号的数字处理系统

根据上面理论的要求，我们对模拟系统的数字化必须有 A/D、D/A 转换器，中间的数字系统主要是 CPU 数字化技术处理，如对信号的滤波，提取和增强信号的有用分量，消弱无用的分量；或是估计信号的某些特征参数。总之，我们对系统数字化，就是用数字方式对信号进行滤波、变换、增强、压缩、估计、识别等。所有这些都取决于 CPU 的功能特性，所以，CPU 的选择尤为重要。

2.2.2 数字式控制器体系结构设计

根据 2.2.1 节对数字系统体系结构的分析，结合 2.1 节我们对模拟控制器的体系结构的研究，我们设计出了如图 2.4、2.5 所示的数字式控制器硬件体系结构示意图。

在调制解调模块中我们用数字系统替换掉 AT89C2051 芯片和模拟滤波器。数字系统既取代模拟滤波器的功能，还实现线路控制信号的调制解调。这样，既实现了系统调制解调模块的数字化，又减少了

外围硬件的使用，提高了系统的集成度。如图 2.4 中虚线框内的部分被数字处理系统替换。

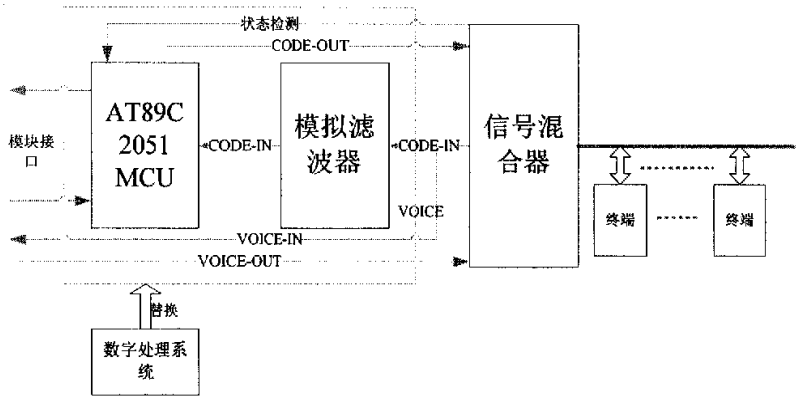


图 2.4 数字式控制器调制解调模块示意图

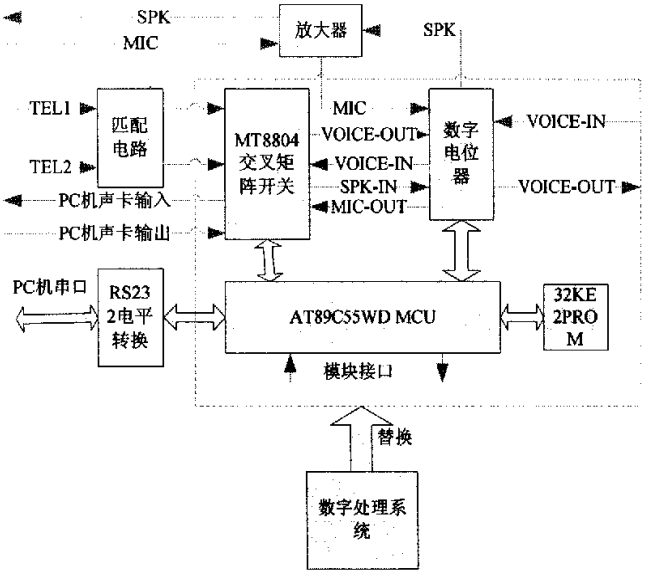


图 2.5 数字式控制器系统控制模块示意图

系统控制模块中，我们用设计用数字系统来替换掉 AT89C55WD MCU、MT8804 交叉矩阵开关、数字电位器和 32K E^2PROM。如图 2.5 所示，虚线框内的部分被数字系统替换。该模块中要求数字信号处理系统芯片不但具有较强的处理能力，还要有丰富的片上资源，能够实现模拟系统中的 MT8804 交叉矩阵开关的功能。

2.3 数字控制器 CPU 的选择

在 2.2.2 节，我们给出了数字式控制器的硬件体系结构示意图。由图中数字处理系统所起到的功能我们可以看出 CPU 的选择非常重要。它关系着整个系统性能的好坏。

对于 CPU 的选择，我们主要基于以下几个原则：

1、CPU 具有数字信号处理能力

如图 2.3 所示，信号经过 A/D 转换器转换为数字信号，为二进制数码表示的数字信号序列。要对这个数字信号序列进行处理，那么系统 CPU 必须是具有数字信号处理能力的处理器。目前，能够进行数字处理的 CPU 有很多，如数字信号处理器 DSP、具有混合信号处理能力的 MCS-96 系列和 C8051 系列单片机以及 ARM 处理器等。

2、集成度高

由于硬件技术的高速发展，数字部件的高度规范性使得硬件电路集成度提高，电路参数要求不严。在模拟控制器中，使用的模拟器件的电路参数要求严格、体积大、功耗大、价格较高，由于系统扩展的器件较多，焊接点也较多，信号就容易受到干扰，使得系统可靠性不高。所以，在数字系统中，根据数字集成电路的特点，我们要尽量使用集成度较高的芯片。DSP、MCS-96 系列单片机、ARM 处理器和

C8051 系列单片机都具有非常丰富的片上资源。

3、性价比要高

对于铁路公寓电脑自动叫班系统控制器的数字化的同时，我们还要考虑到实际的市场经济效益问题。我们不能对控制器数字化后使得系统成本大幅度提高，影响产品的市场竞争力。

我们要选用一款体积小、功耗小、价格低、处理能力相对较强的高性价比的CPU。所以，我们不采用高性能但价格昂贵的DSP和ARM处理器，而是根据系统的需求，选用性能较高但价格较低的具有混合信号处理能力的8051系列单片机。

C8051FXXX作为CPU具有如下的优点：第一点，拥有丰富的片上资源；第二点，功耗低、采用3V供电、且具有两种节电模式；第三点，处理能力强；第四点，性价比高。

C8051系列单片机使用CYGNAL的专利：CIP-51微控制器内核CIP-51。CIP-51工作在最大系统时钟频率25MHz时，它的峰值速度可以达到25MIPS。在本系统的软件实现中，对于时间性能要求较高的功能即实时的软件数字滤波功能，经实际测试，CIP-51的处理速度是完全能够满足要求的。图2.6给出了几种8位微控制器内核在最大系统时钟时的峰值速度的比较关系，由此图可见，在8位微控制器内核中CIP-51是属于处理能力较强的。

所以，我们使用了 Silicon Laboratories 公司的 C8051F300 和 C8051F015 两款芯片。下面我们介绍一下 C8051FXXX 系列芯片的功能特性，以及我们采用这两款芯片的原因。

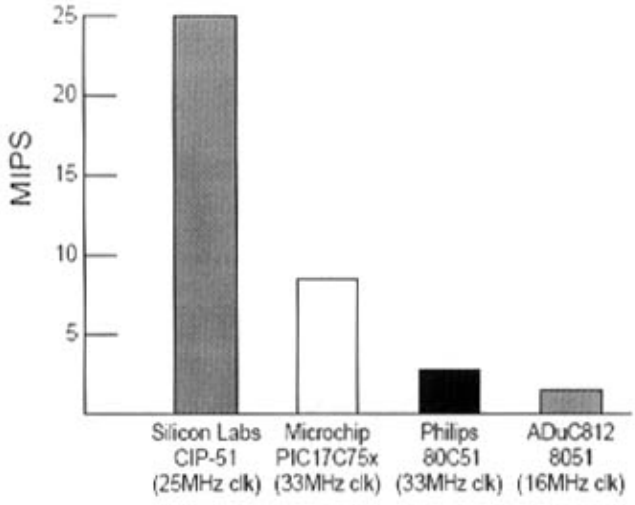


图 2.6 单片机峰值速度比较

2.3.1 信号调制解调模块 CPU 的选择——C8051F300

信号调制解调模块 CPU 主要用于信号的调制解调，所以它必须具有以下功能：第一，较高的 A/D 转换速率。系统要将语音信号数字化，所以必须有 A/D 转换器且 A/D 转换器必须较高的转换速率。第二，执行模拟滤波器的功能，对模拟信号进行滤波，滤除模拟信号中的干扰。而滤波算法一般都比较复杂，这就要求芯片具有较强的处理能力。并且滤波算法一般需要保存多个中间数据，这就要求芯片具有较大的 RAM 来暂时存储这些数据。第三，支持数据的串行传输，因在完成信号的调制解调后，必须通过串行口与系统控制模块进行通信，传输命令。而 C8051F300 可满足这些要求。图 2.7 为 C8051F300 的芯片结构示意图。

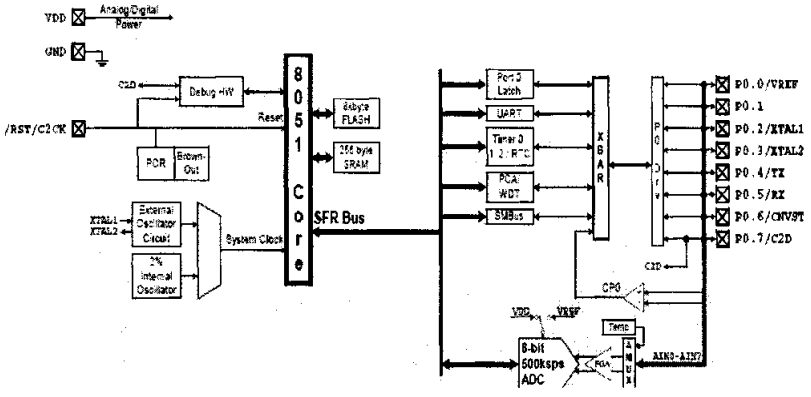


图 2.7 C8051F300 芯片结构图

(1) **C8051F300 的 A/D 资源**

C8051F300单片机的ADC子系统包括1个9通道的可配置模拟多路开关AMUX、1个可编程增益放大器PGA和1个500ksps8位分辨率的逐次逼近型ADC从而提供对线路控制信号采集功能的支持。只有当ADC的控制寄存器ADC0CN中的ADCEN位被置为1时，ADC子系统ADC跟踪保持器和PGA才允许工作。当ADCEN位为0时，ADC子系统处于低功耗关断方式。

AMUX中的8个通道用于外部测量，第9通道在内部被接到片内温度传感器上。可将AMUX输入对编程为工作在差分或单端方式，并允许用户对每个通道选择最佳的测量方式，甚至可以在测量过程中改变方式。在系统复位后，AMUX的默认方式为单端输入。AMUX输出信号的放大倍数可以用软件编程为0.5，1，2，4，8或16。

ADC使用VREF来确定它的满度电压，因此在进行一次转换之前必须正确设置这个参考电压。ADC的最高转换速度为500ksps。转换时钟来源于系统时钟。可以设置转换时钟的速度为系统时钟的1/2，1/4，1/8或1/16。这一功能用于根据不同的系统时钟速度调整转换速度。

有 4 种 A/D 转换启动方式。由 ADC0CN 的 ADC 启动转换方式位

ADSTM1 和 ADSTM0 的状态决定转换触发源有：①写 1 到 ADCOCN 的 ADBUSY 位；②定时器 3 溢出；③外部 ADC 转换启动信号的上升沿；④定时器 2 溢出。本系统采用第 1 种启动方式。

(2) C8051F300 的 UART 接口

C8051F300单片机提供的UART串行接口为我们提供了与PC机进行数据交互的通道(须借助MAX232C芯片进行电平转换)。

UART是一个能进行异步传输的串行口。UART可以工作在全双工方式。在所有方式下，接收数据被放在一个保持寄存器，这就允许在软件尚未读取前一个数据字节的情况下，开始接收第2个输入数据字节。

UART在特殊功能寄存器中有一个相关的串行控制器SCON和一个串行数据缓冲器SBUF。用一个SBUF地址可以访问发送寄存器和接收寄存器。读操作访问接收寄存器，写操作访问发送寄存器

如果允许UART能产生中断。UART有2个中断源，1个发送中断标志TI和1个接收中断标志RI。当UART转向中断服务程序时，硬件不清除中断标志，必须用软件清除。这就是允许软件查询UART中断的原因。

表2.1 UART的工作方式

方式	同步性	波特率时钟	数据位	起始/停止位
0	同步	SYSCLK/12	8	无/无
1	异步	定时器1或定时器2溢出	8	1/1
2	异步	SYSCLK/32或SYSCLK/64	9	1/1
3	异步	定时器1或定时器2溢出	9	1/1

UART提供4种工作方式：1种同步方式，3种异步方式。通过设置SCON寄存器中的配置位选择这4种方式，提供不同的波特率和通信协

议。表2.1表述了这4种方式。

(3) C8051F300 的高速处理能力

在本节开始部分，我们就已经讨论了C8051F300芯片的高速处理能力。CIP-51工作在最大系统时钟频率25MHz时，它的峰值速度可以达到25MIPS。所以对于运行在单片机上的经过简化的滤波算法而言，是可以满足的。

2.3.2 系统控制模块 CPU 的选择——C8051F015

系统控制模块芯片主要用于控制器的系统功能控制。它主要实现的功能有：第一，对终端命令的解调；第二，对 PC 机命令的调制；第三，音量调节的软件数字化实现；第四，动态化 A/D 采样频率技术的实现以及交叉模拟开关的软件实现；第五，IAP 软件装载技术的实现。C8051F015 的体系结构示意图如 2.8 所示：

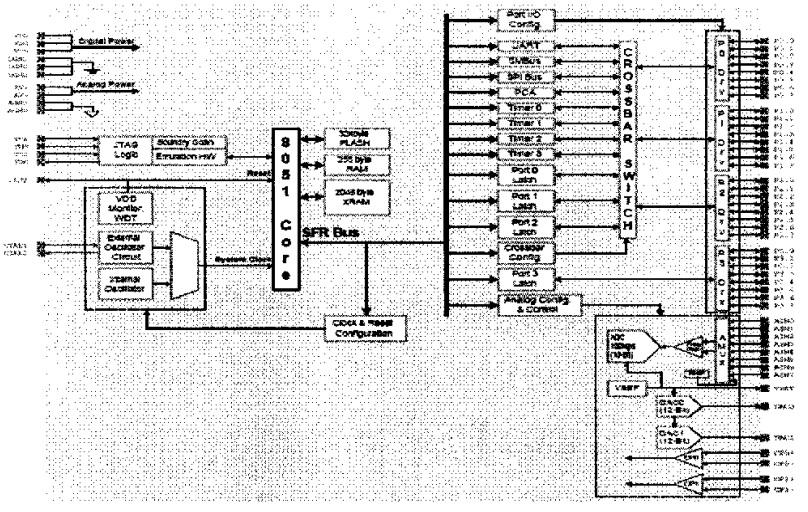


图 2.8 C8051F015 芯片结构示意图

(1) C8051F015 的 SPI 接口

C8051F015单片机提供的SPI串行接口提供了访问4线全双工串行总线的能力，从而实现外围D/A模拟的简单化。

C8051F015单片机的4线分别指MOSI、MISO、SCK、NSS四线。MOSI即主出从入信号，是主器件的输出和从器件的输入，用于从主器件到从器件的串行数据传输。MISO即主入从出信号，是从器件的输出和主器件的输入，用于从从器件到主器件的串行数据传输。SCK即串行时钟信号，是主器件的输出和从器件的输入，用于同步主器件和从器件之间在MOSI和MISO线上的串行数据传输。NSS即从选择信号是一个输入信号。主器件用它来选择处于从方式的SPI模块。

在全双工操作中，SPI主器件在MOSI线上向从器件发送数据，被寻址的从器件可以同时在MISO线上向主器件发送其移位寄存器中的内容，所接收到的来自从器件的数据替换主器件数据寄存器中的数据。两个方向上的数据传输由主器件产生的串行时钟同步。图2.9所示为一个SPI主器件和一个SPI从器件的全双工操作。

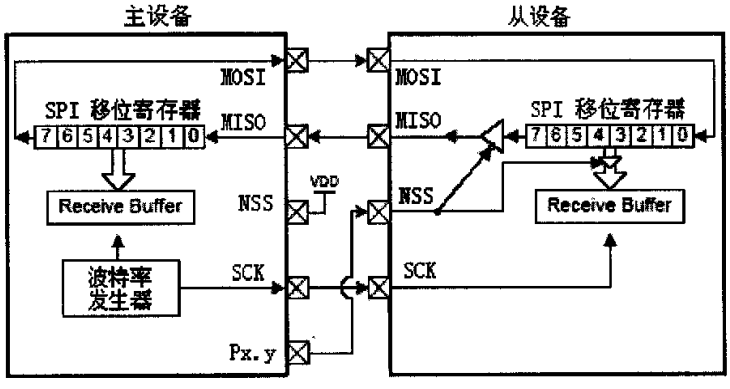


图 2.9 全双工操作

(2) C8051F015 的 UART 接口

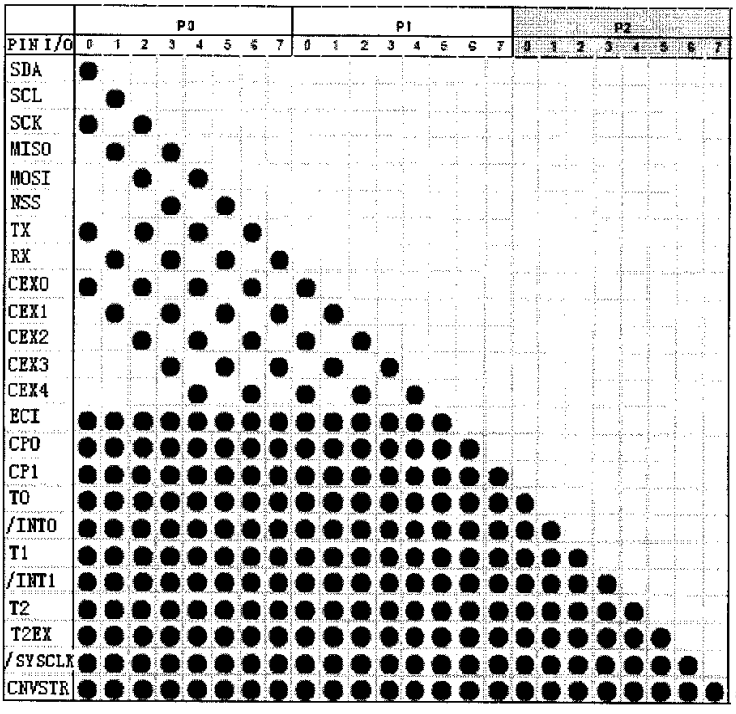
C8051F015 单片机提供的 UART 串行接口完全符合通用的与 PC 机通信的标准，其结构和工作方式与 F300 完全相同。这里不再赘述。有需要的读者请参阅 2.3.1 节。

(3) 端口交叉网络与 I/O 设置功能

C8051F015单片机中有大量的数字资源需要通过数字I/O端口P0，P1，P2和P3才能使用。端口P0，P1，P2中的每个引脚既可以定义为对应的I/O端口，又可以分配给一个内部数字资源。 设计者对功能可以完全控制，只受所选封装的可用引脚数的限制。这种资源分配的灵活性是通过使用优先权交叉开关译码器实现的

1．优先权交叉开关译码器。

表2.2优先权交叉开关译码表



优先权交叉开关译码器为连接内部数字资源到物理I/O引脚提供了很好的解决方案。交叉开关根据优先权译码表(表2.4)将所选择的内部数字资源分配到I/O引脚，寄存器XBR0，XBR1和XBR2用于选择内部数字功能或让I/O引脚默认为I/O端口。

优先权交叉开关译码表为每个I/O功能分配一个优先权。从顶端SMBus开始，如该表所示，当被选择时，它的两个信号被分配到I/O端口0的引脚0和引脚1。译码器总是从端口0开始按LSB到MSB的顺序填充I/O位，然后是端口1。如果需要的话，最后填充端口2；如果选择不用某个资源，则表中的下一个功能将填充这个优先权位置。这样就可以只选择设计中用到的功能，充分利用可用的I/O引脚。另外，任何未分配的I/O端口将被分配到一起，便于在应用代码中使用。

2. I/O端口设置。

我们可以通过设置相应的寄存器位从而允许交叉开关网络、将相应的端口配置为推挽方式或漏极开路方式、将所有配置为漏极开路方式的端口的弱上拉打开。

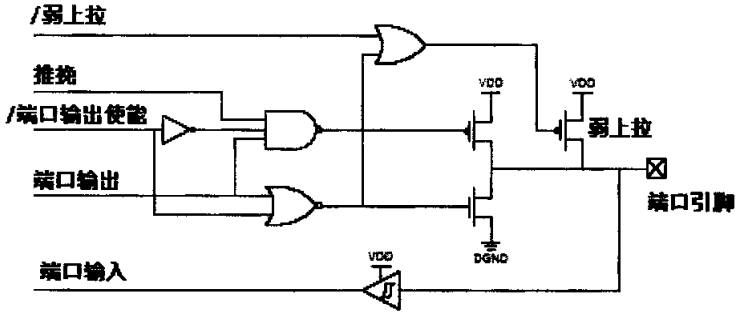


图2.10 I/O端口单元框图

从I/O端口的单元框图(图2.10)可以总结出I/O端口的配置方法为:①输出端口有两种配置方式：第一种为将该端口配置为推挽方式；

第二种为将该端口配置为漏极开路方式且使能弱上拉。①输入端口的配置方式：将该端口配置为漏极开路方式，同时向该端口写1。

(4) C8051F015的A/D资源

C8051F015单片机的ADC子系统包括1个9通道的可配置模拟多路开关AMUX、1个可编程增益放大器PGA和1个100ksps10位分辨率的逐次逼近型ADC从而提供对系统语音信号采集功能的支持。其功能及使用方法与F300芯片一样，这里不再赘述，可参见2.3.1节的AD部分。

与F300相同，F015有4种A/D转换启动方式。本系统采用第4种启动方式。

(5) C8051F015 的 D/A 资源

我们利用C8051F015的DAC来实现语音信号的转换功能，将数字化的语音信号转换为模拟信号，用音箱播放监控。C8051F015系列单片机有2个12位的电压方式DAC(DAC0如图2.11所示)，每个DAC的输出摆幅均为0V - VREF-1LSB,对应的输入码范围是0x000 - 0xFFF。

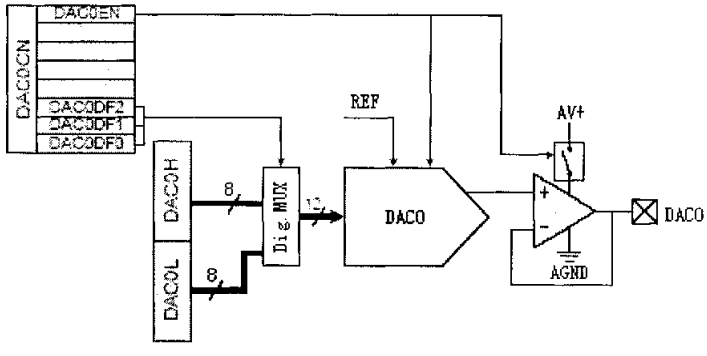


图2.11 DAC0功能框图

以DAC0为例，12位的数据字写到低字节(DAC0L)和高字节(DAC0H)数据寄存器。在写DAC0H寄存器时，数据被保存到DAC0。如果需要12位分辨率，应在写入DAC0L后写DAC0H。DAC可用于8位方式，这种情况

是将数据左移后只写入DAC0H,而在DAC0L中写入0x00。DAC0的允许/禁止功能由DAC0EN位控制。向DAC0EN位写1，允许DAC0工作，向DAC0EN写0,则禁止。DAC0在被禁止时，DAC0的输出保持在高阻状态，DAC0的供电电流降到1微安或更低。

(6) C8051F015的片内存储器

CIP51有标准的8051程序和数据地址配置。它包括256B的数据RAM，其中高128B为2个地址空间：用间接寻址访问通用RAM的高128B；用直接寻址访问128B的SFR地址。数据RAM的低128B可用直接或间接寻址方式访问。前32字节为4个通用工作寄存器区，接下来的16字节既可以字节寻址又可以位寻址。(见图2.12)

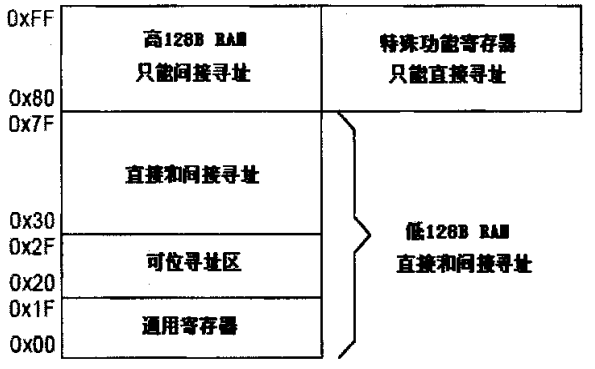


图2.12内部数据地址空间

C8051F015中的CIP-51还另有位于外部数据存储器地址空间的2048B的RAM块。这个2048B的RAM块可以在整个64KB外部数据存储器地址空间中被寻址。

C8051F015单片机的程序存储器包含32KB的FLASH。该存储器以512B为一个扇区，可以在系统编程，且无须在外提供编程电压。从0x7E00-0x7FFF的512B被保留，由工厂使用。(见图2.13)

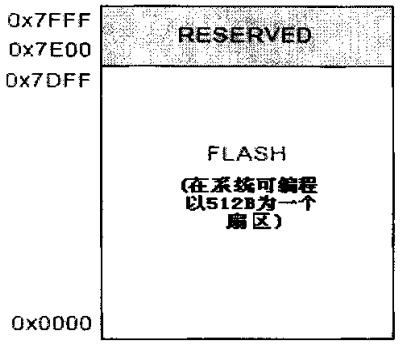


图2.13 程序存储器

在系统控制模块中，我们使用F015芯片的一个主要功能就是利用它的UART串行接口与PC进行通信，而PC机发送过来命令有时是以9600波特连续发送过来的，单片机来不及完全处理。这时，我们就把这些命令保存在一个队列里面，让单片机以后处理。所以，这个命令缓存队列会占用较大的RAM，我们使用时为这个缓存队列分配了128BRAM。

另外，在模拟系统中，有一个扩展的32K的E^2PROM，主要用来存储日期和房间信息。在数字系统中，我们利用F015芯片的32K的FLASH存储这些信息以及应用程序。另外，我们为了以后软件升级方便，在数字式控制器系统中增加了IAP程序装载技术，该IAP的功能实现在后面章节详细讲述。

2.3.3 数字控制器体系结构

通过前面几节对数字式控制器设计思想的考虑，结合对C8051F系列单片机——C8051F300和C8051F015片上资源的分析，我们设计出了如图2.14和2.15的控制器硬件体系结构。控制器整体结构图见附录5。

1、数字式控制器调制解调模块体系结构

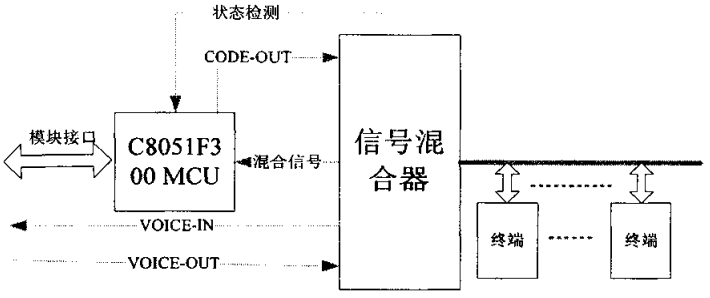


图2.14 数字式控制器调制解调模块体系结构简图

在此模块中，C8051F300芯片主要实现线路控制信号的调制解调和模拟滤波器的功能。当终端有应答信号送回时，混合信号在C8051F300内先经过软件滤波，再从控制信号中解调出控制信息送往C8051F015处理。当有命令从系统控制模块过来时，控制命令经过C8051F300调制，送往信号混合器；语音信号直接由外围设备送往信号混合器，信号混合器处理后送往终端。

2、数字式控制器系统控制模块体系结构

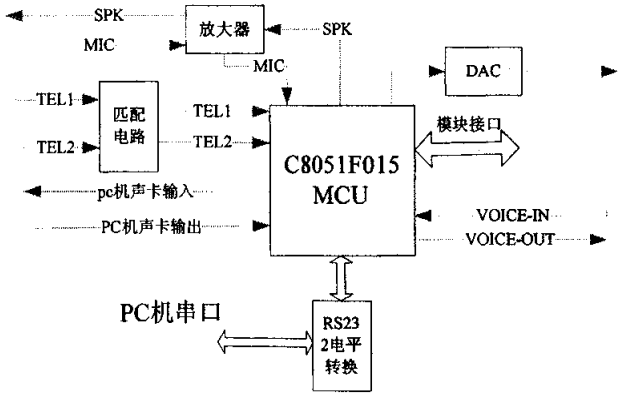


图2.15 数字式控制器系统控制模块体系结构简图

在数字式控制器系统控制模块中，C8051F015芯片主要实现对终端命令和对PC机命令的“翻译”、音量调节的软件数字化实现、动态化A/D采样频率技术的实现、交叉模拟开关的软件实现以及IAP软件装载技术的实现。

由图2.15可以看出，C8051F015芯片主要控制多个声道的切换，这主要是实现MT8804交叉模拟矩阵的功能；与PC机的通信，发送或接受各种叫班命令，是否对叫班录音；对于音量的软件控制，动态化A/D采样频率技术的实现，IAP软件装载技术的实现全采用软件技术实现。

2.4 C8051F 微控制器的程序存储器

2.4.1 C8051F 微控制器的 FLASH 存储器的组成

C8051F 微控制器内部有数 KB 至数百 KB 的 FLASH 存储器，以页（PAGE）/扇区（SECTOR）为单位组织，页/扇区大小为 512B、1024B 等。

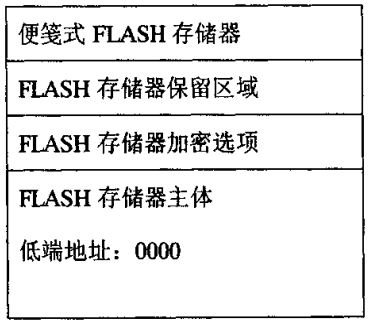


图 2.16 C8051F 微控制器的 FLASH 存储器结构示意图（容量小于 64KB）

有些 C8051F 微控制器另有仅供存放数据的数百 B 的便笺式 FLASH 存储器，其页大小为 128B。C8051F 微控制器的 FLASH 存储

器位于程序存储器空间内。C8051F 微控制器的 FLASH 存储器的结构如图 2.16 所示。（这里我们仅讨论容量小于 64KB 的情况）

C8051F 微控制器的 FLASH 存储器的主要特性如表 2.3 所示。

表 2.3 C8051F 微控制器的 FLASH 主要特性

参数	最小值	典型值	最大值
使用寿命（擦/写次数）	10K	100K	
擦除时间/ms	10	12	14
写入时间/us	40	50	60

2.4.2 作为程序存储器的使用方法

作为程序存储器使用时，对于 FLASH 存储器容量小于 64KB 的 C8051F 微控制器，其程序地址和物理地址是直接对应的，操作规则和方法与标准的 8031 完全相同。需要了解的读者请参阅相关文献。另外，由于 C8051F015 微控制器的 FLASH 大小为 32KB，所以，对于 FLASH 存储器容量大于 64KB 的 C8051F 微控制器，这里也不再讨论。

2.4.3 作为非易失性数据存储器的使用方法

当将 FLASH 存储器作为非易失性数据存储器使用，比许可以对 FLASH 存储器进行写/擦除操作。由于 8031 无写程序存储器的指令，因此在进行写/擦除操作时，必须将 FLASH 存储器映射为外部数据存储器，这样才可以使用写外部数据存储器的指令完成写 FLASH 存储器的操作。

对于 FLASH 存储器容量小于 64KB 的 C8051F 微控制器，采用的

是直接映射，即映射后的程序地址和物理地址是直接对应的，因此操作规则和方法标准的 8031 完全相同。

FLASH 存储器内的每个数据位可以由“1”改写为“0”，但反之却不行，必须先按页进行擦除操作（即将该页内的所有数据位置为“1”）。为修改某页/扇区中的某个字节，必须首先把整个页/扇区保存到一个临时存储区，然后整个页/扇区擦除，更新数据后再写回到原来的页/扇区。

FLASH 存储器的写/擦除操作由硬件自动定时，不需要进行数据查询来判断写/擦除操作何时结束。在写和擦除操作期间处理器不再从 FLASH 存储器中读取程序，对于无指令预取寄存器/程序 CACHE 的 C8051F 微控制器，处理器将暂停运行；对于有指令预取寄存器/程序 CACHE 的 C8051F 微控制器，处理器将执行完指令预取寄存器/程序 CACHE 中的指令后暂停。

为满足对 FLASH 存储器正确操作的时间要求，必须根据系统时钟频率设置正确的 FLASH 存储器的操作参数（FLSCL 中的相关位）。

对 C8051F 微控制器的 FLASH 存储器编程的最简单的方法是使用由 CYGNAL 或第三方供应商提供的编程工具通过 JTAG 接口编程，这是对未初始化器件的唯一的编程方法。或将 FLASH 存储器作为非易失性数据存储器，在程序中使用 MOVX 指令对 FLASH 存储器进行在系统编程。后一种方法即可以实现 ISP 或 IAP，为实现由远端控制来更新软件提供了必要的条件。

2.4.4 C8051F 微控制器程序存储器的安全和保密

1、安全选项

为了保护 FLASH 存储器中的数据/程序，防止被软件意外地修改，C8051F 微控制器提供了相应的保护手段。

首先，FLASH 存储器的写/擦除操作过程必须严格按有关规定操作步骤进行。除此之外，当进行写操作时，写允许位（PSWE 中的 PSCTL.0）必须有效，即将 FLASH 存储器映射为外部数据存储器；当进行擦除操作时，写允许位（PSWE 中的 PSCTL.0）和擦除允许位（PSEE 的 PSCTL.1）必须有效。

2、保密选项

(1) 对 JTAG 接口的保密选项

FLASH 存储器中的某些字节被用来作为读锁定标志、写/擦除锁定标志，可选择部分存储块读锁定（或只能全部读锁定）、部分存储块写/擦除锁定（或只能全部写/擦除锁定），有些 C8051F 微控制器简化为单一的锁定标志。这些锁定标志的作用是防止通过 JTAG 接口读取其中的数据/程序、写/擦除其中的数据/程序。

一旦安全字节所在的存储块被设置成写/擦除锁定，解锁的唯一办法是通过 JTAG 接口进行整个程序存储器空间擦除。对任何一个写/擦除锁定的存储块执行 JTAG 接口擦除操作，将自动启动对整个程序存储器空间的擦除（保留区除外）。

(2) 软件读限制（SRL，Software Read Limit）

软件读限制实际上是在 FLASH 存储器的低端设定了一个区域，运行在该区域外的程序无法读取该区域内的数据/程序，但可调用该区域内的程序。有些 C8051F 微控制器的软件访问限制可单独进行设置，而有些就简单地将锁定区域规定为有类似特性的访问限制[1]。

第 3 章 系统软件设计

3.1 嵌入式系统软件的特点

3.1.1 嵌入式系统的软件与硬件的关系

嵌入式系统的软件是实现嵌入式系统功能的关键，对嵌入式系统软件的要求与通用计算机是不同的。嵌入式系统的软件运行在独一无二的硬件上，最后的机器语言始终都要依赖于实际机器。

嵌入式系统开发人员必须控制开发工具为特定的硬件编译源代码，定义整个运行环境，以及软件怎样被打包或调度。因此，嵌入式系统开发人员必须对执行环境、开发工具、运行时软件包有更多的了解。在编写软件时必须知道以下内容：

① 存储器结构：包括系统的程序空间（ROM 或闪存）、数据空间（RAM）和 I／O 空间。存储器的编址方式与所采用的处理器有关：有些处理器的此三个空间是相互独立的；有些处理器的程序空间和数据空间统一编址；而有些处理器的此三个空间是统一编址的。

② 系统启动：一旦复位信号有效后处理器就从程序空间特定的地址开始运行程序空间中的程序。首先需运行初始化程序，包括对处理器和各种硬件资源进行初始化。

③ 中断：尽管在所有的计算机体系结构中，中断的切换机制是相同的，但不同的处理器在实现细节上还是有一些不同之处。需注意各硬件的功能手册。
各硬件的功能手册。

3.1.2 嵌入式系统的软件的特点

1．控制硬件：

嵌入式系统程序员经常需要编写一些直接控制外设的代码。对于不同的计算机体系结构，设备可能是端口映射的（I/O 与存储器独立编址），也有可能是内存映射的（I/O 与存储器统一编址）。

如果系统采用 I/O 与存储器独立编址，开发人员可能别无选择，只能使用汇编语言完成实际操作，这是由于 C 语言没有真正的“端口”概念。

如果系统采用 I/O 与存储器统一编址，事情就简单多了。访问 I/O 与访问存储器方法相同，只是需对 I/O 进行地址的绝对定位。

2．按位操作：

嵌入式系统编程经常需要操作硬件寄存器内单个二进制位。在大多数情况下，最好的方法是读出整个寄存器值，改变二进制位，然后把整个值写回到设备寄存器中。

3．软件要求固态化存储：

为了提高系统可靠性，嵌入式系统中的软件一般都固化在存储器芯片或处理器中，而不是存贮于磁盘等载体中。

4．执行速度和效率

尽管有一种只要有可能就要使用高级语言编程的趋势，但在大多数嵌入式系统中，进行操作所需的时间和 ROM 大小经常得不到充分满足，所以效率就成为关键因素，ISR 和其他有严格时序限制的例程应当由汇编语言编写。开发人员在使用 C 语言编程时与直接使用汇编语言编程时一样要小心翼翼，需要注意节约每一点存储空间与执行时间。

5．中断与中断服务例程

嵌入式系统如果花费太多的处理器时间检查某个轮询循环的单个状态位就会严重拖累系统。中断是无法回避的事实，中断需要按其重要性排出优先顺序。

从概念上讲，一个 ISR 只是编写的一小段代码。外设（对微控制器而言，外设可能与之同处一块芯片中，但在处理器核心之外）发出中断信号到处理器的中断输入，如果处理器能接受这个中断，它就开始执行硬件 ISR 响应周期。

6．可嵌套的中断与可重入性

如果高优先级的中断可以中断低优先级中断而得到优先响应，此中断系统即为可嵌套的，在可嵌套的中断系统中，编程就更复杂了。为了避免这种复杂性，简单的系统可在响应当前中断时屏蔽其他中断。当中断服务例程完成时，它才重新开中断。

如果允许中断嵌套，程序员必须特别小心，要确认所有在中断服务例程运行期间被调用的函数都是可重入的。如果 ISR 调用了不可重入函数，程序最终可能会出现相互访问或同步方面的错误，此时系统将崩溃。

7．测量执行时间

在许多嵌入式应用产品中，程序员需要精确了解程序执行时间。许多时间关键性 ISR 尽管在逻辑上是正确的，但可能会由于执行时间太长而造成错误。尽管在理论上计算汇编语言程序运行时间能精确到每个机器周期，但随着系统体系结构变得越来越复杂，计算过程也越来越复杂。

用汇编语言编写的程序可以为每条指令计算确切的执行时间，而用 C 语言编写程序就没有什么简单方法能计算出其确切的执行时间。

3.2 嵌入式系统软件的开发方法

嵌入式系统的编程环境可采用基于实时多任务操作系统的编程或基于处理器的直接编程。

基于实时多任务操作系统的编程以操作系统内核为基础，只需完成相关任务的编程，其实时性和可靠性由操作系统保障，较适用于功能复杂的应用系统。

基于处理器的直接编程需完成全部软件设计，其实时性和可靠性与编程人员的水平密切相关，适用于功能较简单的应用系统。

3.2.1 基于处理器的直接编程

目前在中国大多数嵌入式软件开发还是基于处理器的直接编程，没有采用商品化的RTOS，不能将系统软件和应用软件分开处理。这种软件开发方法对于较小的系统或对软件代码长度敏感的系统是合适的，特别是后一种系统必须采用这种软件开发方法,这种系统在实际应用中占了相当大的比例。基于处理器直接编程主要缺点是开发相对较复杂，软件的修改和维护也比较困难，要求编写者有相当丰富的经验。为今后对软件进行修改和维护，必须做好详细的文档。

基于处理器的直接编程通常把嵌入式程序分为两部分，即前台程序和后台程序。前台程序通过中断服务程序（ISR）来处理事件；后台程序则掌控整个嵌入式系统软、硬件资源的分配、管理及任务的调度，是一个系统管理调度程序。在任务运行时，后台程序检查每个任务是否具备运行条件，通过一定的调度算法来完成相应的操作。

1、前/后台系统

直接编程采用的是前/后台（或称为超循环）系统，其结构如图 3.1 所示。

在前/后台系统中，后台按时间顺序执行每个任务，也可称后台为任务级，前台用于处理异步事件（中断），也可称前台为中断级。一般系统中，中断处理得到的信息需由后台处理，因为中断程序不能长时间占用 CPU，否则将影响其他中断的进行。

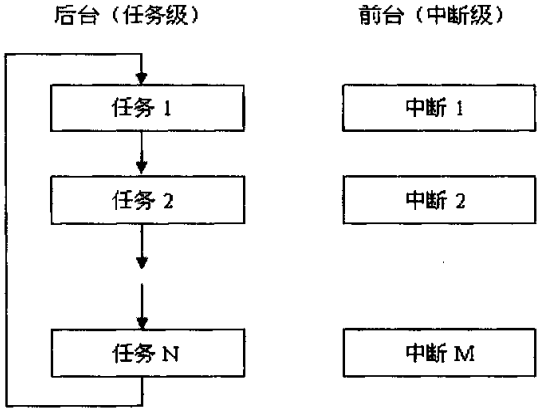


图 3.1 前/后台系统示意图

2、可抢占式任务调度的前/后台系统

前/后台系统的实时性较差，信息处理的最大时延为后台的整个循环的执行时间，而且此执行时间不是固定的常数，这对于时间关键性任务是不允许的。可仿照 RTOS 的任务可抢占式任务调度方式，为前/后台系统构造一个任务抢占机制来提前执行时间关键性任务。实现具有可抢占式任务调度的前/后台系统，其关键在于如何使某一中断得到的信息能马上进行处理。具有可抢占式任务调度的前/后台系统的实现原理如图 3.2 所示。

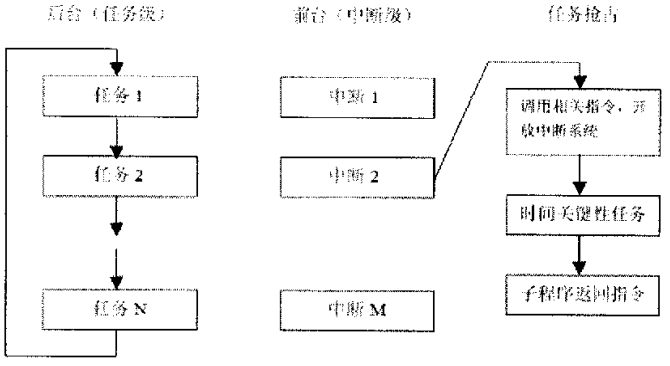


图 3.2 具有可抢占式任务调度的前/后台系统示意图

在 CPU 进入前台任务中断 2 时，应对被中断的后台任务的现场进行保护，在执行关键性任务前还应开放中断系统，以免影响 CPU 对其他中断的响应。通过调用中断返回、开中断等指令，可完成开放中断系统的操作。随后可执行时间关键性任务。当时间关键性任务完成以后，执行子程序返回指令即能正确地返回到后台系统的断点。

3、可抢占式和优先级任务调度的前/后台系统

还可实现具有可抢占式和优先级任务调度的前/后台系统，其实现原理如图 3.3 所示。

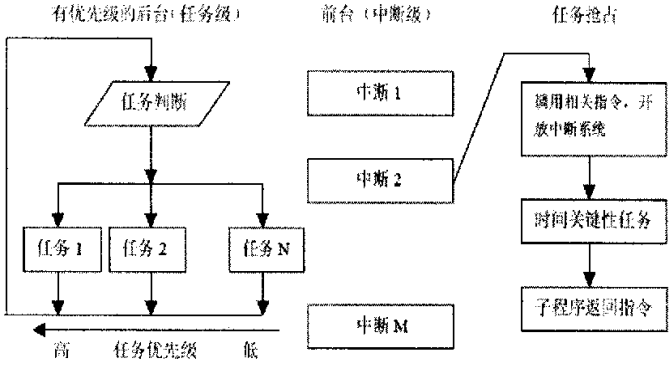


图 3.3 具有可抢占式和优先级任务调度的前/后台系统示意图

3.2.2 基于 RTOS 的编程

任何基于RTOS的多任务应用程序的基本模块都是一个个的任务。一个任务就是一个函数，一定数量的资源或时间被分配给它来完成一定的功能。对于大多数的 RTOS，每个任务都有一定的优先级。

在上述前/后台系统中，各任务是按顺序进行的，因此任务的切换很简单，不必为每个任务安排堆栈，也不必保存每个任务所使用的 CPU 寄存器。而在基于 RTOS 的编程时，任务的切换是在 RTOS 控制下按一定规则进行的，RTOS 必须谨慎地处理每个任务的堆栈、任务控制块和所使用的 CPU 寄存器。

在 RTOS 环境下，每个任务都有各自的堆栈和任务控制块，分时使用的 CPU 寄存器，如图 3.4 所示。

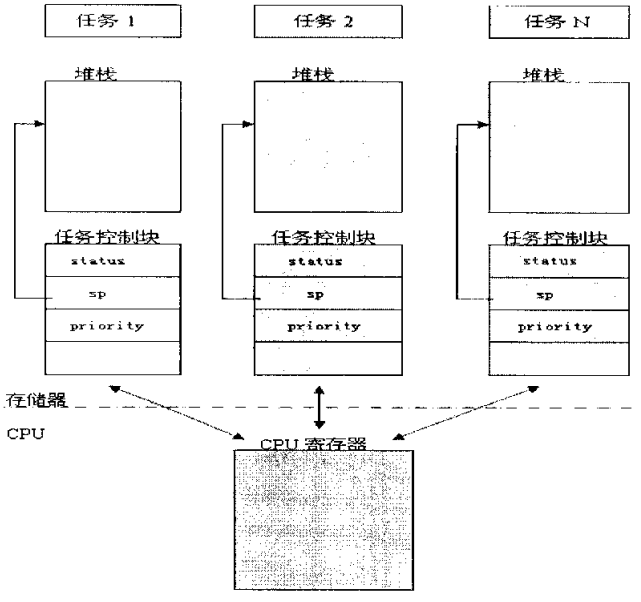


图 3.4 RTOS 环境下的任务

每个任务的状态流图如图 3.5 所示。

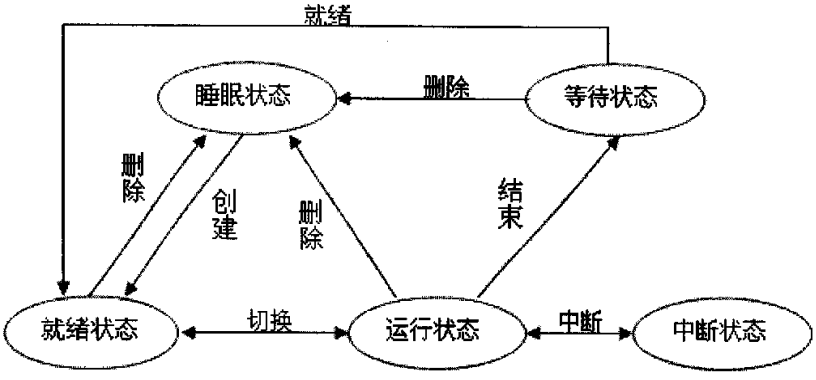


图 3.5 RTOS 环境下的任务的状态流图

3.3 IAP 技术与系统升级

在数字式控制器中，采用了数字信号处理器，提高了系统的集成度和软件含量，为以后系统的升级提供了方便。在无需更改硬件的情况下直接升级软件就可以了。为了我们升级软件的方便，我们在系统中增加了 IAP 程序装载技术。

3.3.1 IAP 功能简介

IAP（In Application Programming）是应用在 Flash 程序存储器的一种编程模式。简单地说就是在应用程序控制下，对程序某段存储空间进行读取、擦除、写入操作。它不需要从电路板上取下芯片用编程器烧写再安装上去运行新程序，即具有在线编程功能。IAP 可以在某段程序的控制下对另外一段程序 Flash 进行读写操作，可以控制对某段、某页甚至某个字节的读写操作。

3.3.2 IAP 与 ISP 区别

目前，编程模式均为在线编程，而在线编程有两种实现方法：在系统编程（ISP，In System Programming）和在应用编程（IAP，In Application Programming）。ISP 一般是通过单片机专用的串行编程接口对单片机内部的 Flash 存储器进行编程，而 IAP 技术是从结构上将 Flash 存储器映射为两个存储体，当运行一个存储体上的用户程序时，可对另一个存储体重新编程，之后将控制从一个存储体转向另一个。ISP 的实现一般需要很少的外部电路辅助实现，而 IAP 的实现更加灵活，通常可利用单片机的串行口接到计算机的 RS232 口，通过专门设计的固件程序来编程内部存储器。

IAP 和 ISP 在操作方式、结果和应用场合都存在很大区别：

1、 **操作方式**

ISP：用写入器将 code 烧入，不过，芯片可以在目标板上，不用取出来，在设计目标板的时候就将接口设计在上面,所以叫“在系统编程”，即不用脱离系统；

IAP：在应用编程,有芯片本身(或通过外围的芯片)可以通过一系列操作将 code 写入，比如一款支持 IAP 的单片机，内分 3 个程序区，1 作引导程序区，2 作运行程序区，3 作下载区，芯片通过串口接收到下载命令，进入引导区运行引导程序，在引导程序下将 new code 内容下载到下载区，下载完毕并校验通过后再将下载区内容复制到 2 区，运行复位程序，则 IAP 完成。

IAP 在系统运行的过程中动态编程，这种编程是对程序执行代码的动态修改，而且毋须借助于任何外部力量，也毋须进行任何机械操作。ISP 在进行加载程序以前，需要设置某些功能引脚，迫使 IC 转入

自举状态。而 IAP 则不需要作硬件上的任何动作，只要有合法的数据来源。数据源既可以是内部程序运行的结果，也可以来自 UART，I/O 口或者总线。换个角度来来说，IAP 不仅提供现场或者远程软件修改升级，也可以把它理解成 IDATA，PDATA 或者 XDATA，替代 I2C 之类的外部 E2PROM，存储并加密数据。

2、 **硬件要求**

ISP 通常是整片擦除、编程，在手工操作下通过 PC 串口下载程序到 Flash，需要简单的硬件资源——串口 RX、TX 和 RS232 驱动芯片。

IAP 是在某段程序的控制下对另外一段程序 Flash 进行读写操作，可以控制对某段、某页甚至某个字节的读写操作。

3、 **应用场合**

(1)ISP 程序升级需要到现场解决；

(2)IAP 不必到现场，通过网线等任何通信方式就可进行系统升级。

3.3.3 IAP 与系统应用

在我们的数字式控制器中就是把 IAP 模式融入到控制器的软件系统中，采用 C/S 结构，用 PC 机结合单片机串口升级指定控制器芯片的应用软件。

我们在控制器系统中对 IAP 的设计主要是根据嵌入式系统的 Boot Loader 原理来实现。Boot Loader 就是引导加载程序，系统加电后运行的第一段软件代码。在嵌入式系统中，通常并没有像 PC 机 BIOS 那样的固件程序（有的嵌入式 CPU 也会内嵌一段短小的启动程序），

因此整个系统的加载启动任务就完全由 Boot Loader 来完成。系统在上电或复位后，CPU 从复位地址处开始执行，而在这个地址处安排的通常就是系统的 Boot Loader 程序。

IAP 的本质就是，MCU 可以灵活地运行一个常驻 Flash 的引导加载程序（Boot Loader Program），实现对用户应用程序的在线自编程更新。引导加载程序的设计可以使用任何的可用的数据接口和相关的协议读取代码，或者从程序存储器中读取代码，然后将代码写入（编程）到 Flash 存储器中。

C8051 家族中的 C8051F015 芯片的 FLASH 为可分区可擦除的 FLASH，可将其划分为大小为 512B 的页，具备实现引导加载支持的用户程序自编程功能(In-System Programming by On-chip Boot Program)，它可以提供一个真正的由 MCU 本身自动下载和更新（采用读/写同时"Read-While-Write"进行的方式）程序代码的系统程序自编程更新的机制。利用这个功能，可以实现在应用编程（IAP）以及实现系统程序的远程自动更新的应用。

Boot Loader 程序的设计是实现 IAP 的关键，它采用 Client/Server 体系结构，通过一个通信接口，采用某种协议正确的接收数据，将完整的数据写入到软件程序存储区中。具体过程将在第六章详细介绍。

3.4 软件总体结构

由于 51 系列单片机的内存空间较小(仅有 256B,且有一大部分有特殊用途)，资源有限，因此并不适合移植操作系统。那么，我们又应该如何进行软件的总体设计，从而使得我们的软件具有有操作系统支持的软件系统的结构清晰，调试容易等的优点呢？我们采用了如下

的设计思路：仿照多任务运行软件(有操作系统的支持的嵌入式系统的软件)的设计思想，将系统功能划分为多个任务，结合单片机的前后台软件设计思路进行软件的总体设计。根据我们在 3.3 节介绍的嵌入式系统软件的设计方法，我将控制器所执行的功能划分为几个任务，并对它们的紧急系数做了分析和归纳，制定了各个任务的优先级。总体均采用前/后台系统的编程方法。前台运行中断级任务，后台运行信息处理任务。

3.4.1 任务划分及任务间的同步与通信

因为我们系统使用了两款芯片，它们均有自己的功能模块，所以下面我们分别对它们的任务划分作一下介绍。

1、 **调制解调模块任务划分**

表 3.1 调制解调模块任务划分表

任务名称	前/后台	功能描述
MCU_INIT	后台	MCU 初始化
FILTER_INIT	后台	滤波变量初始化
_FILTER	后台	滤波编码函数
SEND_COMMAND	前台	命令发送函数

因为是基于处理器直接编程，我们没有严格让它们按优先级运行。由于 SEND_COMMAND 任务是必须在_FILTER 任务后运行的，所以我们采用了中断触发方式来发送已经解调出来的命令。_FILTER 函数一直处于循环检测状态，当有控制信号到来时，它就进行解码；没有信号时它就一直处于检测状态。

2、 **系统控制模块任务划分**

系统控制模块比较复杂，执行的任务比较多，所以在任务间的通信与同步也比较复杂。。表 3.2 简单列举出了系统控制模块的任务划分。其中系统所有可能处于的 8 种执行状态以及叫班的音量控制任务都处于后台执行；而系统线路自动检测、数据串行发送函数、数据串行接受函数、电路电压监视函数、环境初始化函数等任务都在中断中实现，属于前台任务。

表 3.2 系统控制模块任务划分表

任务名称	前/后台	功能描述
MCU_INIT	后台	MCU 初始化
STATE_IDLE_TASK	后台	空闲状态
STATE_AUTO_TASK	后台	自动叫班
STATE_TEL1_TAPE_TASK	后台	电话 1 录音
STATE_TEL1_CALL_TASK	后台	电话 1 录音/通话
STATE_TEL2_TAPE_TASK	后台	电话 2 录音
STATE_TEL2_CALL_TASK	后台	电话 2 录音/通话
STATE_SBON_TASK	后台	放音
STATE_TALK_TASK	后台	普通对讲
VOICE_CONTROL_TASK	后台	音量大小控制函数
INIT_AUTO_DETECTION	前台	系统线路自动检测
COMT_SEND_COMMAND	前台	数据串行发送函数
INT300_REC_TASK	前台	数据串行接受函数
COMPARATOR_ISR_TASK	前台	电路电压监视函数
VOICE_STARTUP	前台	环境初始化函数

3.4.2 系统软件总体结构

根据前/后台系统的软件设计方法，我们设计了系统的软件整体结构示意图。

1、调制解调模块软件结构

本模块由于信号的调制解调涉及到硬件寄存器和位操作比较多，所以为处理方便采用了汇编编程实现。

```
    ACALL   MCU_INIT
    LCALL   _FILTER_INIT
TASK_CIRCLE:
    LCALL   _FILTER
    LCALL   _SEND_CODE
    JMP   TASK_CIRCLE
```

2、系统控制模块软件结构

```
void main (void)
{
   MCU_INIT();
   if(TERMINAL_TYPE_NUMBER == 0xF0)
   {
      while(LINE_RIGHT_OR_WRONG)
      {
         INIT_AUTO_DETECTION();
      }
   }
```

```
while(1)
{
    if(BUFFER_HEAD != PRE_TAIL)
    {
        COMMAND_DEAL();
    ..............................
    命令的串行发送和接受函数;
    }
    }
}
```

其中，COMMAND_DEAL()函数中调用了系统工作的8种状态，这里仅示意表示，不再详细列出。

前台任务由于是中断级任务，所以，由定时器等触发器控制，这里不再细讲。

第 4 章 信号的调制解调

在公寓叫班系统中，最主要的任务就是按时叫班。所以我们的数字式控制器的一个重要任务就是处理线路上的控制信号和语音信号。因为我们整体所采用是数字化系统，要处理这些信号，我们必须将其转换为数字信号。所以本章我们就讨论系统的难点之一，信号的数字化滤波问题，以及我们对终端命令的调制解调技术。

4.1 系统信号简介

在叫班系统中，终端连接结构有两种：二线制和四线制。在二线制体系结构中，所有房间终端均并联在两根线上，线路控制信号和语音信号混合在线路上线路上传输，两根线即供电又传输信号，信号即可上行传输，也可下行传输；在四线制体系结构中，两根线用来供电，另外两根线用来传输信号，一根线上行传输信号，一根线下行传输信号，线路上传输的信号也是控制信号与语音信号的混合信号，但信号调制方法与二线制不同。

对于一般的从数模混合信号中提取数字信号的方法，在我曾发表的学术论文《一种从数模混合信号中提取数字信号的时域算法》中有详细讨论。对于我们控制系统实际工程中的混合信号，我们进行了算法的优化改进。

图 4.1 所示为语音信号与控制信号的频率关系。由图上可知，语音信号频率范围为 200Hz～4000Hz，系统控制信号频率范围为 6000Hz～10000Hz。

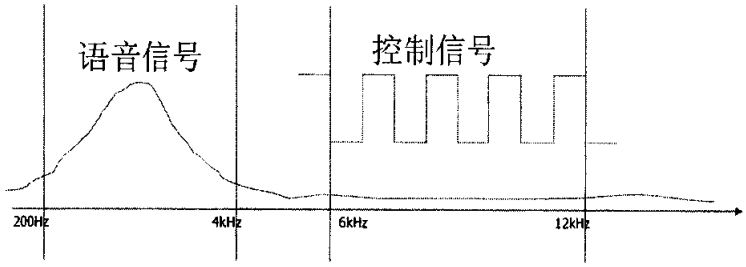


图 4.1 语音信号与控制信号频率关系图

二线制终端发送的控制信号波形如图 4.2、4.3 所示。

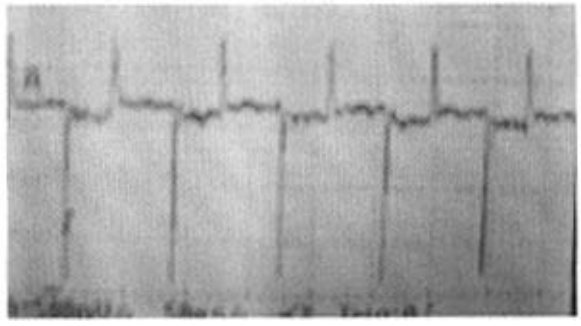


图 4.2 二线制终端发送的控制信号高频部分波形图一（a）

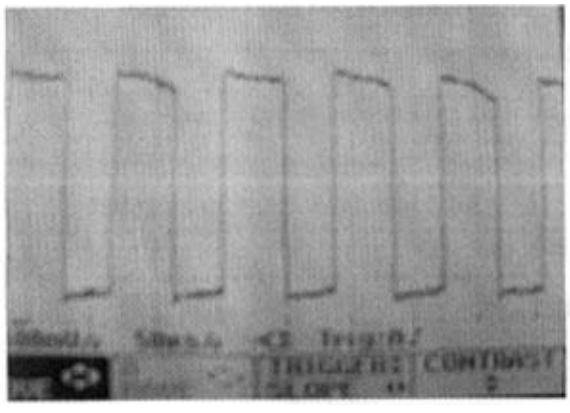


图 4.3 二线制终端发送的控制信号高频部分波形图一（b）

图 4.2、4.3 中的波形图为二线制终端发送的两种波形的高频部分，低频部分波形周期是高频波形的 2 倍，波形相同。

图 4.2 的尖峰波形为 PC 机叫班时终端应答的控制信号。图 4.3 为终端直接呼叫时发送的控制信号。二者是由于终端端机的电路阻抗特性的因素而产生的波形不同。在对它们进行滤波处理和解调信息时的算法应兼顾这两种情况。

四线制终端发送的控制信号波形如图 4.4 所示。

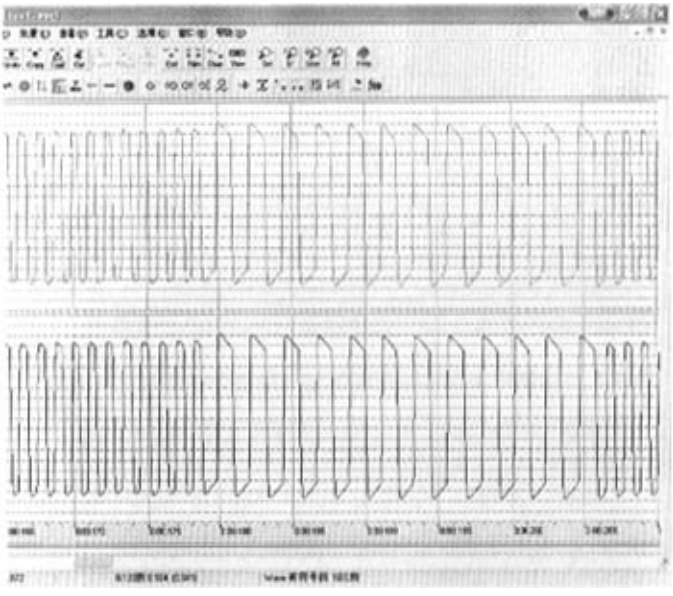


图 4.4 四线制终端发送的控制信号波形图

二线制系统和四线制系统端机发送的波形不同、编码方式也不同，所以在后面的信号滤波算法、命令解码编码算法方面我们都分开讨论。

4.2 数字信号处理基本理论

我们的混合信号在经过滤波以后，线路控制信号进入 F300 芯片进行处理。要运用数字系统处理从控制信号中提取命令，就必须把信号数字化，然后进行时域或频域的变换滤波处理，从而解调出控制命令。

模拟信号在时域空间中表现为幅值随时间连续变化的曲线；数字信号强调在时间上和幅值上都不连续，描述方式采用二进制数量来表示。系统中的信号是混合信号，我们首先在数字系统中对信号进行数字化滤波，最后进行解调处理。

4.2.1 信号数字化基本理论

▲采样周期

两次采样的时间间隔大小叫做采样周期，用 T_s 表示

▲采样频率

单位时间内的采样次数.用 f_S 表示.并有:

$$f_s = \frac{1}{T_S} \qquad (4\text{-}1)$$

▲采样频率的选择

①与采样精度和采样后的数据量大小有关。

在单位时间内采样次数越多,则对信号的描述越细腻,越接近真实信号，即采样频率 f_S 应尽量高。但一味提高采样频率，增大数据量，给数据处理带来了麻烦，增加了技术实现上的困难。

②与被测信号的变化速度有关。

在过短的时间里反复测量体温或是河流水位的变化是完全没有必要的。这就是说，采样频率的选择必须考虑被采样信号变化的快慢程度,f_S 是一个相对值。

▲采样定理

采样频率 f_S 必须高于被采样信号所含最高频率的两倍。该定理指出：当对连续变化的信号波形进行采样时，若采样频率 f_S 高于该信号所含最高频率的两倍，那么可以由采样值通过插补技术正确地恢复原信号的波形，否则将会引起频谱混叠（Aliasing）产生混叠噪声（Aliasing Noise）,而重叠的部分是不能恢复。

f_S 低于信号中最高频率的两倍，将出现频谱混叠，原信号的频谱与下边带无法分开，破坏了原信号的频谱，原信号将无法恢复。

4.2.2 控制信号的数字化

芯片 F300 片上集成了一个 8－bit 的 ADC，最高采样速率可达 500ksps。它在信号幅度方向有 256 个量化阶。在要对控制信号提取信息编码之前，信号经过 ADC 的采样、量化、编码，转化为一系列的十六进制的表示幅度的数字序列。我们的整个数字系统要处理的数字信号就是这个数字信号序列。

4.3 基于信号幅度特征的时域滤波算法

在模拟系统中，在 MCU 处理控制信号之前，混合信号先经过了一个高通滤波器，滤除了低频的语音信号。而在数字化系统中，去掉了高通滤波器，改用软件实现高通滤波器的功能。

混合信号在经过 ADC 处理后，接下来的任务就是从获得的数据中提取控制命令。由于混合有语音信号的数据，我们需要采取滤波算法滤除它。

F300 单片机最高处理性能可达 25MIPS，但仍不能满足复杂的滤波算法运行，所以我们只能新研究一种适合的滤波算法来达到高通滤波器的功能。现有的数字滤波算法有多种：限幅滤波法（又称程序判断滤波法）、中位值滤波法、算术平均滤波法、递推平均滤波法（又称滑动平均滤波法）、中位值平均滤波法（又称防脉冲干扰平均滤波法）、限幅平均滤波法等。经过对这些算法的分析研究，发现这些算法均不适合我们的情况。我们滤波的目的是要滤除低频的语音信号，对高频的控制信号解码，所以在经过仔细研究后，提出了一个新的滤波算法——基于信号幅度特征的时域滤波算法。

基于信号幅度特征的时域滤波算法不采用标准的数字信号滤波方法，而是在时域进行信号幅度特征的提取，以达到提取数字信号的目的。因为所需处理的数模混信号为语音信号（模拟信号，频率范围为 200Hz～4000Hz）和数字信号（频率范围为 6000Hz～10000Hz）的迭加。此数模信号的幅度特征是：在变化相对缓慢的模拟信号上迭加了快速变化的数字信号，可根据此幅度特征完成数字信号的信息的提取。

4.3.1 基于信号幅度特征的时域滤波算法原理

对于幅度变化相对缓慢的模拟语音信号，其幅值变化在一个采样间隔内一定小于某个范围，而幅度快速变化的数字信号与之相反。基于信号幅度特征的时域滤波算法的原理正是基于这一点。只要给出模

拟语音信号在一次采样周期内幅值变化的最大值，就可以利用这个值为判断界限进行数字信号的提取。

下面以正弦波为例计算出在一次采样周期内幅值变化的最大值，如公式（4—2）所示。

$$MAX = A_1 * |Sin[\omega(\tau_0 + \Delta t)] - Sin(\omega\tau_0)| \qquad (4-2)$$

公式（1）中：A_1 是正弦波的振幅值；Δt 是 ADC 采样周期。

由于正弦波的最大变化率出现在 0 角度点，所以就计算 0 角度点的一次采样周期内正弦的最大变化值。如图 4.5 所示，横坐标 $\tau_0 = 0.2$ 或者 $\tau_0 = 0.7$ 开始的一次 ADC 采样。以此值为算法的提取数字信号信息的判定门限。数字信号在同一时间的变化幅值在判定门限之上，语音信号的变化幅值在判定门限之下。

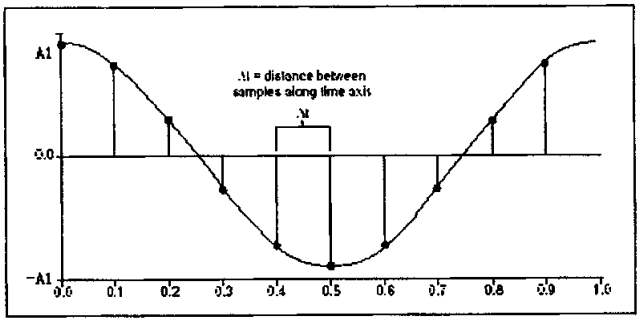


图 4.5 正弦波采样示意图

通过判断连续两次所取的采样值差值是否超出这个判定门限来决定是否提取出数字信号和滤除其中缓慢变化的噪声。算法流程示意图如图 4.6 所示：

图 4.6 基于信号幅度特征的时域滤波算法示意图

4.3.2 基于信号幅度特征的时域滤波算法分析

基于信号幅度特征的时域滤波算法完全在时域对信号进行处理，它根据相邻两次采样的幅度差值判断是否为模拟语音信号或数字控制信号，边滤波边提取数字信号信息。

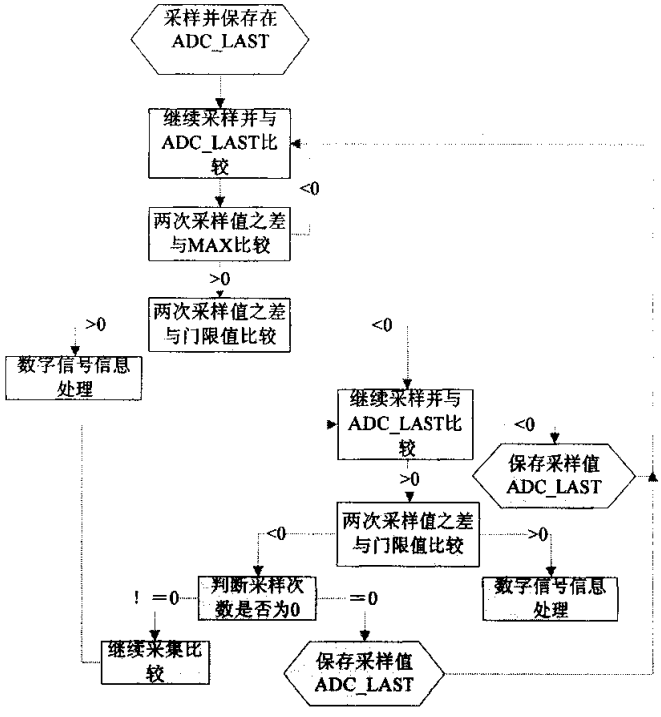


图 4.7 基于信号幅度特征的时域滤波算法的流程示意图

基于信号幅度特征的时域滤波算法的实现过程如下：取连续的两

次采样值 ADC_LAST、ADC0，判断其差值（ADC0－ADC_LAST）是否大于 *MAX*，如果成立,就保存 ADC_LAST。后面连续采样 N 次，只要某一次的采样值与 ADC_LAST 的差值大于我们所确定的数字信号门限，就认为提取到了数字信号，并且干扰也被滤除。不存储和计算 N 次采样的平均值，节省了存储 N 次采样值的 RAM 空间，减少了算法的运算周期，做到了信号采集和滤波处理并行执行，达到了实时性强和准确性高的目的。基于信号幅度特征的时域滤波算法的流程示意图如图 4.7 所示。

4.4 系统控制信号编码

在滤除了语音信号、得到了控制信号的信息后，要根据它们的调制方式对信息进行编码，才能得到控制命令。由于四线制结构的系统终端和二进制结构的系统终端的信号调制方式不一样，所以应采取不同的信息编码方式。

4.4.1 二线制结构系统终端编码

对于二线制结构系统终端发送的波形如图 4.8 所示。

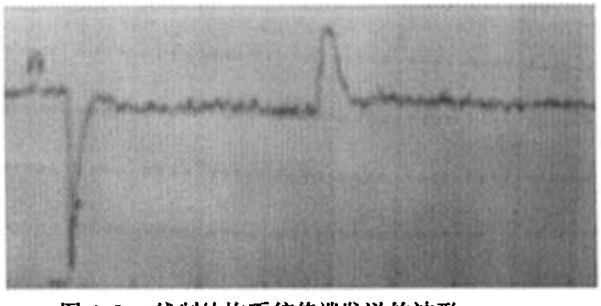


图 4.8 二线制结构系统终端发送的波形

由于电路的阻抗特性，二线制结构系统终端发送的波形比较奇特一点。除了正规方波，还有如 4.8 图的尖峰波形，图中一个波峰和一个波谷为一个波形周期。由示波器观察得知，单个波峰或波谷的周期太小，而 F300 单片机的 ADC 最高采样频率为 500ksps，周期为 20μs，所以无法进行采样。只有对信号进行展宽才能进行处理。展宽后波形波峰和波谷变宽，但信号周期不便。

端机采用 FSK 方式，12KHz 表示“1”，8.4KHz 表示“0”。数据的帧格式为 0FEAXH，其中 0FE 为数据帧的起始头，AH 为数据的特征符，XH 为数据。

根据波形的特征，我们设计了如下编码方法。原理如图 4.9、4.10 所示。

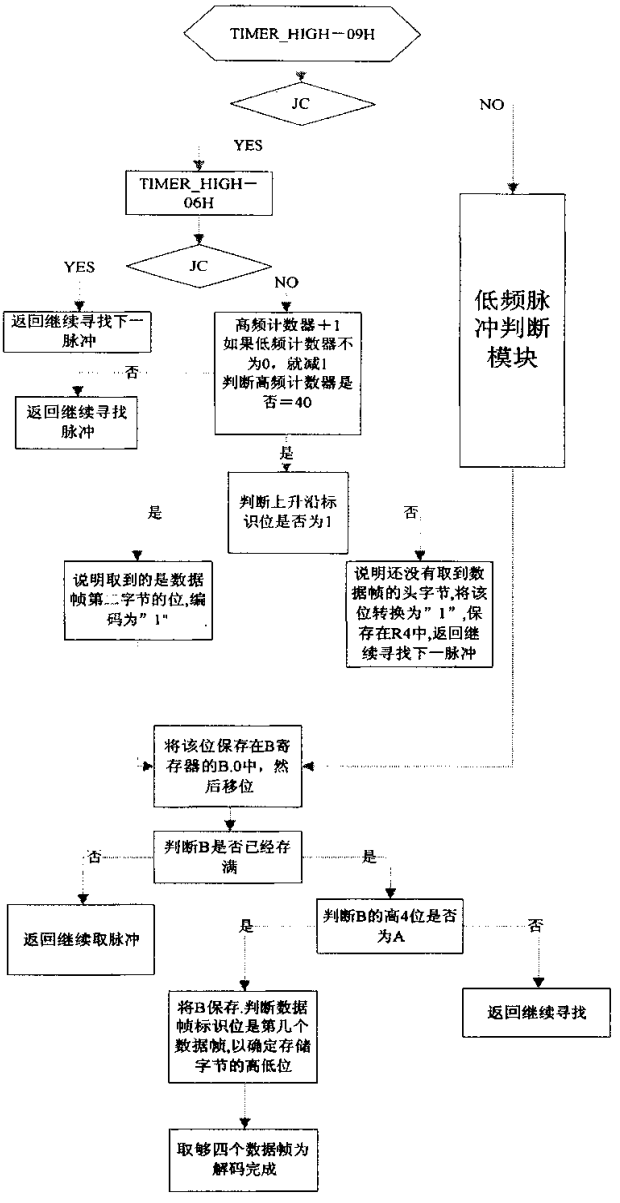


图 4.9 二线制系统编码方法

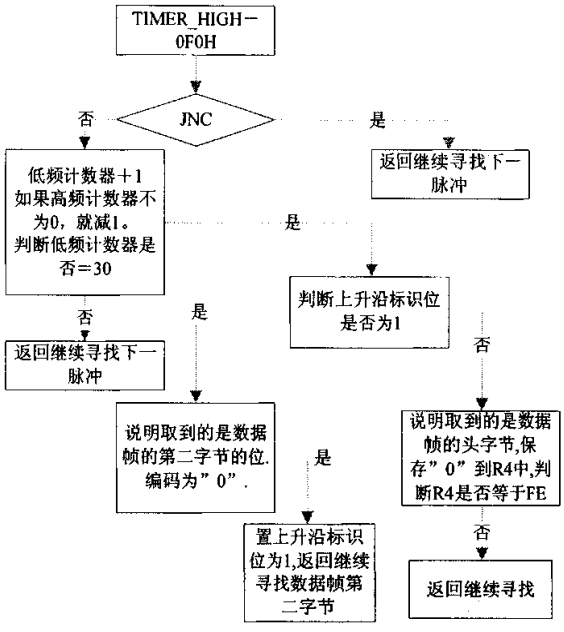


图 4.10　二线制系统编码算法低频脉冲判断模块

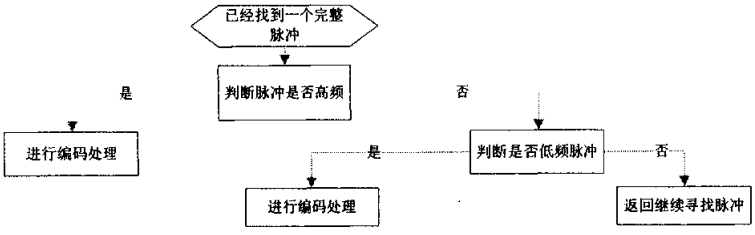
编码原理很简单，根据波形特征，我们用基于信号幅度特征的时域滤波算法取到波形的周期，根据周期判断是高频脉冲还是低频脉冲。当取到 40 个高频脉冲后就认为得到一个“1”bit；当取到 30 个低频脉冲后认为得到一个“0”bit。这样依次取，得到 8bit 后转换为一个十六进制数据。取够 2 个字节后判断是否符合 0FE AXH 的格式，连续取四次。最后把所有的 X 组合起来就得到我们的线路控制信息。详细算法请分析图 4.9 过程。

4.4.2 四线制结构系统终端编码

对于四线制结构系统终端发送的波形如图 4.3 所示。

端机先发送波形引导序列，三个高频脉冲三个低频脉冲连续

发送，发送多次后，接着发送带有特征的调制波形。根据波形特征，先得到 50 个高频脉冲，接着得到 35 个低频脉冲的特征为“1” Bit；先得到 35 个低频脉冲，接着得到 50 个高频脉冲的特征信息为“0” bit。这样依次取，得到 8bit 后转换为一个十六进制数据。连续取两次，最后得到两个字节的命令即为线路控制信息。详细算法如图 4.11 和 4.12 所示。



4.11 四线制系统编码方法

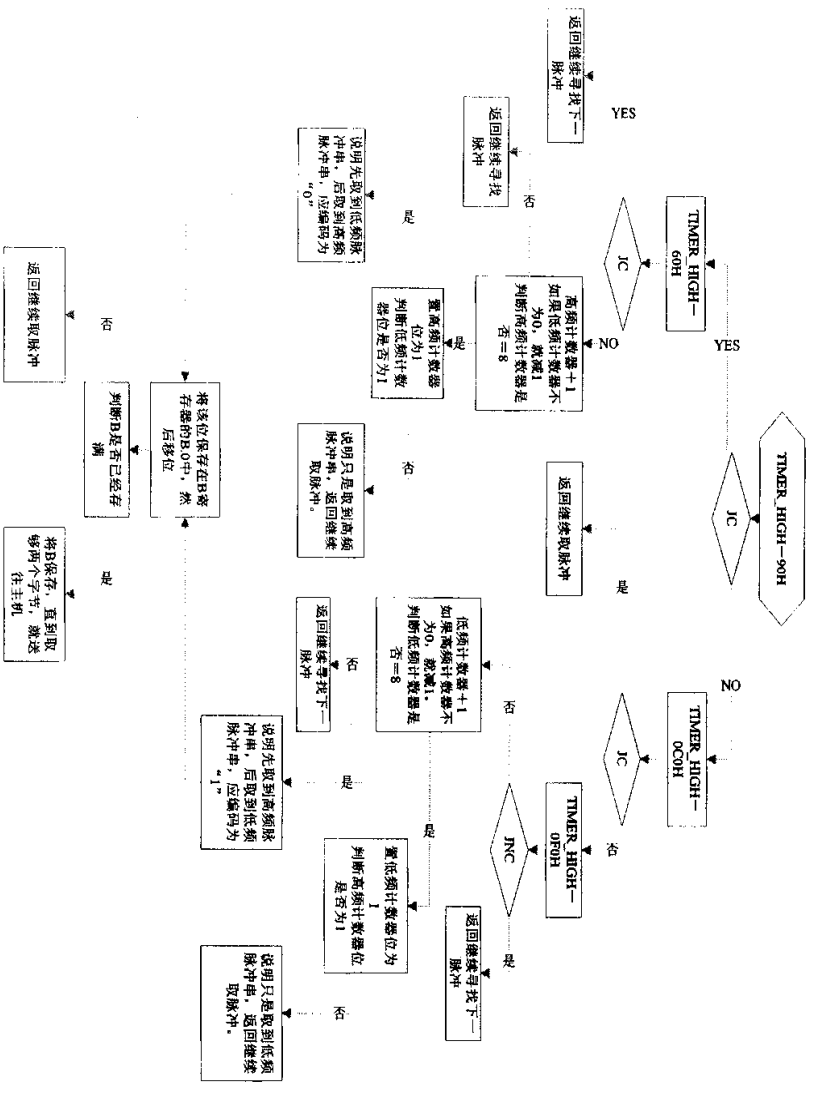


图 4.12 四线制系统编码算法－编码处理

第 5 章 控制模块中相关技术探讨

在本章中，我们主要讨论一下在系统控制模块中我们所使用的几项数字化技术。它们是我们的数字化技术在控制器系统上能够实现的关键，也是我们在数字式控制器研制过程中的技术创新点。

5.1 软件实现音量控制

在模拟控制器中，我们对系统音量的控制采用的是一个数字电位器。它具有调节较准确，不易受振动和污染等以外事件影响的优点。但为节省空间、增强系统可靠性、节省生产成本、易于系统装配和调试，在数字式控制器中，我们用软件实现音量的数字化控制。

对音量的控制，就是对信号波形的振幅的调整。模拟语音信号的音量控制，通过数字电位器进行调节加载的电压大小来调节。在不使用数字电位器的情况下，模拟语音信号经过 A/D 后转换为数字信号，要对音量大小进行控制，就要对这一系列数字进行同等比例的缩小或放大，也即调整振幅。

在单片机中进行这种操作，就要充分使用位操作来提高运行效率。8051 系列为 8 位 MCU，左移一位，相当于乘以 2，数据放大一倍；右移一位，相当于除以 2，数据减小一倍。我们通过这种原理对得到的语音数据都放大或缩小，从而实现音量的控制。

但在操作过程中需注意，F015 为 8 位 MCU，所以无论左移还是右移，均不能大于 8 位。

放大或缩小的数据经过 D/A 后转换为模拟信号，振幅比原来会放大或缩小一定比例，但波形不变，保持了声音的保真度。为了使声音

尽可能清晰，减少噪声干扰，可以简单进行滤波。单片机数字滤波有多种：

1、 程序判断滤波法：根据经验确定两次采样允许的最大偏差，再与采集进来的数据进行比较。

2、 中位值滤波法：连续采样 N 次（奇数次），然后把 N 次采样值按大小排列，取中间值为本次采样值。

3、 算术平均滤波法：连续取 N 个值进行采样，然后进行算术平均。

4、 递推平均滤波法：把数据看成一个队列，每进行一次新的测量，把测量结果放入队尾，扔掉原来队首一个，再进行测量，计算滤波值时，只要把队列中的 N 个数据进行平均，就可以得到新的滤波值。

5、 防脉冲干扰值滤波法：先去掉 N 个数据中的最大值和最小值，然后计算 N-2 个数据的算术平均值。

6、 限幅平均滤波法：相当于“限幅滤波法”+“递推平均滤波法”，每次采样到的新数据先进行限幅处理，再送入队列进行递推平均滤波处理。

以上均为适用于单片机的简单的数字滤波，可以根据需要选择适合的方法来采用。在数字式控制系统中，我采用了递推平均滤波法。

关于音量控制的软件程序，我们在附录 2 中给出，有兴趣的读者请参阅附录 2。

5.2 交叉矩阵开关的软件实现

在模拟控制器体系结构的系统控制模块中，MT8804 交叉矩阵开

关的作用非常重要，它主要用来根据命令切换声道。在数字式控制器中，我们采用软件方式实现交叉矩阵开关的功能。

在第三章中，我们给出了 C8051F015 任务划分，其中有八个后台任务：STATE_IDLE_TASK、STATE_AUTO_TASK、STATE_TEL1_TAPE_TASK、STATE_TEL1_CALL_TASK、STATE_TEL2_TAPE_TASK、STATE_SBON_TASK、STATE_TEL2_CALL_TASK、STATE_TALK_TASK。这八个任务根据系统命令相互之间进行切换。任务命令如表 5.1 所示。

表 5.1 任务命令表

命令	任务名称
0X06	STATE_IDLE_TASK
0X16	STATE_AUTO_TASK
0X26	STATE_TEL1_TAPE_TASK
0X36	STATE_TEL1_CALL_TASK
0X46	STATE_TEL2_TAPE_TASK
0X56	STATE_TEL2_CALL_TASK
0X66	STATE_SBON_TASK
0X76	STATE_TALK_TASK

八个任务要利用的声道各有不同，根据命令要实时地切换声道。为了达到这一要求，我们充分利用了 C8051F015 芯片的 A/D 的九个采样通道中的七个：通道 AIN0 对电话 1 采样，通道 AIN1 对电话 2 采样，通道 AIN2 对终端语音采样，通道 AIN3 对 MIC 采样，通道 AIN4 对声霸采样，通道 AIN5 对电话 1 进行状态监控，通道 AIN6 对电话 2 进行状态监控。详细情况可以参见表 5.2 的 AD/DA 的通道控制表。在表中可以看到，八个不同的叫班任务状态使用的声道不同。当控制器收到

新的任务命令或跳转到其它任务状态，需要对声道进行切换时，我们通过修改芯片特殊寄存器 AMXOSL（AMXOSL.AMXAD3-0）进行。这样，通过对不同声道的控制实现对外围设备的控制，从而实现了交叉矩阵开关 MT8804 的功能。

表 5.2 AD/DA 通道控制表

A/D TASK	T1IN	T2IN	LIN	MIC	SBP	T1V&T2V	DAC0	DAC1	DAC2
State_idle_task						×			
State_auto_task			×	×	×	×	×	×	×
State_tel1_tape_task	×					×			×
State_tel1_call_task	×		×	×		×	×	×	×
State_tel2_tape_task		×				×			×
State_tel2_call_task		×	×	×		×	×	×	×
State_sbon_task					×	×	×		
State_talk_task			×	×		×	×	×	

表中有 X 表示在本状态中该通道需要采样。其中，DAC0、DAC1、DAC2 三个通道是用来播放声音，不在采样通道之列。LIN 和 MIC 是不同时工作，当 LIN 讲时，MIC 听；当 MIC 讲时，LIN 听。T1V 和 T2V 是两路电话的监控通道，所以，它们必须一直处于采样状态。

对于任务切换的命令格式我们是这样设计的：（如图 5.1）

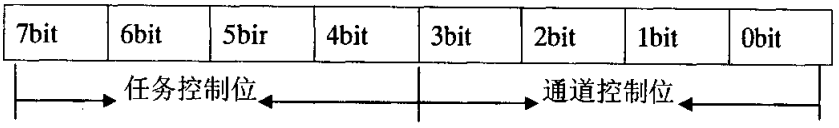


图 5.1 系统任务命令格式图

命令的高四位是任务控制位，通过识别这四位，系统可以调转到需要运行的任务状态。低四位是通道控制位，通过判断这四位，系统可以对需要采样的通道进行采样。功能与交叉矩阵开关相同。

通过命令格式的设计和 A/D 通道的有效利用，成功实现了交叉矩阵开关的软件实现。

5.3 A/D 动态频率取样技术探讨

在 5.2 节开始，我们给出了 C8051F015 的任务划分和各个任务所需要的采样通道。由表 5.2 可以看出，八个任务中，最多的需要控制 5 个采样通道。对于采样频率最高为 100KSPS 的 C8051F015 微控制器，如何控制设置采样方式使系统的语音达到最大程度的清晰，提高系统音质，将是本节所要讨论的内容。

根据奈奎斯特采样定理：要从抽样信号中无失真的恢复原信号，采样频率应大于 2 倍信号最高频率，即奈奎斯特采样频率为信号频率的两倍。工程上的采样频率一般为奈奎斯特采样频率的 2——3 倍。对于语音信号，采样频率高低决定了声音失真程度的大小，高采样频率意味着其存储音频的数据量越大，失真越小，反之，则失真越大。但是，高采样频率意味着其存储音频的数据量越大，采样频率的高低是根据奈奎斯特采样定理和声音信号本身的最高频率决定的。

目前在多媒体系统中捕获声音的标准采样频率定为 44.1kHz、22.05kHz 和 11.025kHz 三种。而人耳所能接收声音频率范围大约为

20Hz--20KHz，但在不同的实际应用中，音频的频率范围是不同的。例如根据 CCITT 公布的声音编码标准，把声音根据使用范围分为以下三级：

◆电话语音级：300Hz-3.4kHz

◆调幅广播级：50Hz-7kHz

◆高保真立体声级：20Hz-20kHz

因而采样频率 11.025kHz、22.05kHz、44.1kHz 正好与电话语音、调幅广播和高保真立体声（CD 音质）三级使用相对应。

在我们的数字式控制器系统中，对语音信号的处理，按要求采用 11.025KHz 的采样频率，根据工程上的应用原则，我们应尽可能对系统进行优化，提高对信号的采样频率，优化系统的音质。

根据表 5.2，我们进行分析，在八个任务中，需要控制采样通道最少的任务是 STATE_IDLE_TASK，它只需要对两路电话进行状态检测。需要控制采样通道最多的任务是 STATE_TEL1_CALL_TASK 和 STATE_TEL2_CALL_TASK，它们需要控制五个采样通道进行工作。但由于 LIN 和 MIC 不同时工作，所以可以认为是最多有四路通道同时工作，即 4 X 11.025 ＝ 44.1KHz。

而 C8051F015 芯片的 AD 采样频率最高可达 100KSPS，所以它可以满足采样的要求。但由于工程环境、系统质量等要求，我们需要最大程度提高采样频率，以保证语音的清晰度。由于微控制器硬件的性能已经提到最高，无法改善，我们就从软件方面着手，通过控制采样通道的采样时间分布来相对地优化 AD 采样频率。

由于微控制器的 AD 最高采样频率一定，所以，采样通道数目越少，分配给每个采样通道的采样时间就越多，采样频率就越高。而且

对于各采样通道，我们要最大程度的利用采样时间。当有一个通道需要工作时，我们就只对这一个通道进行循环采样；当有两个采样通道需要工作时，我们就对这两个通道进行循环采样；依次类推，使各通道的采样频率随通道数目的多少动态变化，但一直保持通道采样频率为最高值。下面进行详细分析。

■对两路电话的监控通道进行改善。由于电话的接听、挂断均是人为反应，而人的行为反应单位基本为秒级，所以，这两个通道根本无需太高的采样频率，设置为秒级中断取样即可。这样，相当于这两个通道几乎是其它通道采样频率的万分之一，大大降低了采样频率的占有分量。几乎可以认为最多时只有 3 个通道需要采样，每个通道的采样频率可高达 100/3 = 33.3KHz。

■各个状态的采样通道数目都不一样，各个通道采样频率要求也不尽相同，我们采用了动态取样频率的采样技术。即实时调整各通道的采样频率，使各通道的采样频率随通道数目的多少动态变化，一直保持通道采样频率为最高值。在最少需要两个通道的工作状态，采样频率是 100/2 = 50KHz。在最多五个采样通道工作的状态，根据需求设计不同通道不同的采样时间间隔。如在 STATE_TEL1_CALL_TASK 状态，T1IN、LIN、MIC、T1V、T2V 五个通道工作，T1IN 对电话 1 录音，需要至少 11.025KHz 的采样频率，所以我们设计它要求最高。LIN 和 MIC 是对讲使用通道，为单工通道，可以在设计上合为一个通道的采样需求。T1V 和 T2V 根据前面讨论我们把它们采样频率要求降低一万倍，即 T1IN 采样一万次，它们采样一次。而 AD 转换器一直运行在最高速率，并处于繁忙的工作状态。只要一个通道采样完成，立即切换到其它通道进行采样。通道之间的切换通过修改芯片特殊寄存器

AMXOSL（AMXOSL.AMXAD3-0）进行。这种具有优先级的、各通道采样频率不断变化但却一直处于最高的采样技术，我们称之为动态取样频率的采样技术。

5.4 UART 软件实现

F015 作为系统控制中心，不但需要与 PC 机进行双向通信，还需要与 F300 进行双向通信。但由于 8051 系列单片机片上资源的有限性，每个芯片上只有一个 UART 通信串口，这就需要我们自己用软件模拟一个 UART。

F015 上的 UART 串口我们用来与 PC 机通信，从主机上接收命令。我们模拟一个与 F300 进行通信的 UART 串口。

5.4.1 软件 UART 接收功能的实现

线路上一个字节信号的波形如图 5.2 所示，数据发送速率为 2400bps。

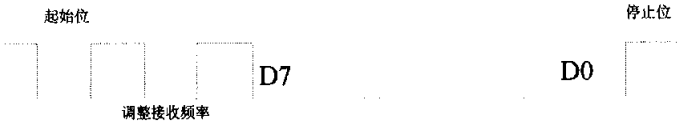


图 5.2 数据发送波形图示

波形图示，为我们所模拟的 UART 口所要接收的数据波形，它有一个起始位和一个停止位，紧跟在起始位后的是调整接收频率的“101”电平，后面是从高位到低位的一个字节数据 D7---D0，最后是一个停止位。

我们采用两个中断对其进行提取，一个外部中断 0，一个定时器 0

中断。外部中断 0 采用下降沿触发方式，当一个字节数据的起始位到来时，触发外部中断，外部中断服务子程序初始化定时器 0 中断参数并启动定时器 0。定时器 0 采用溢出方式来定时从串行线路上提取数据。定时器的溢出频率设置为 1/2400Hz。具体示例程序参见附录 3。

5.4.2 软件 UART 发送功能的实现

软件 UART 发送的波形命令如图 5.1 所示，发送的波特率为 2400bps。相比较起来，发送比接收容易实现。根据图 5.1 的波形特征，我们使用了定时器 2 的中断服务功能，每一次中断移位发送命令的 1bit。

我们将定时器 2 的中断频率设置为 1/2400Hz。当要向终端发送命令时，把命令按字节分开，每一字节作为一次发送。首先，将一个字节的命令送入一个寄存器，进行发送参数的初始化。接着启动定时器 2，设置为溢出方式。根据字节命令的每一位为“1”还是“0”置线路上的电平高低即可。具体示例程序参见附录 4。

5.5 D/A 扩展

在 F015 芯片上，有两个 D/A 转换器：DAC0、DAC1。DAC0 用来连接 PC 机声卡，实现叫班的记录录音；DAC1 用来接 F300，进行正常叫班工作。但由于我们必须监控是否线路叫班正确，还要在本地连接一个音箱，进行同步播放，以确定叫班成功。所以我们还需要一个 D/A 通道，这就需要我们外围扩展一个 D/A 转换设备。

我们采用了芯片 MAX550，它是一款串行的、8bit 电平输出的数模转换器（DACs）。它内部集成一个 DACs，同步输入。它的引脚示

意图如图 5.3 所示。

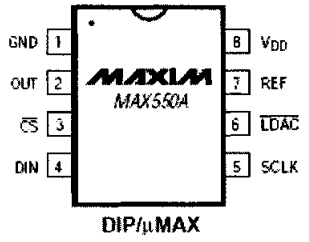


图 5.3 MAX550 管脚示意图

它有八个引脚，/CS 为片选引脚，DIN 为数据输入管脚，SCLK 为时钟控制管脚，OUT 为信号输出管脚。

MAX550 的操作时序图如图 5.4 所示。

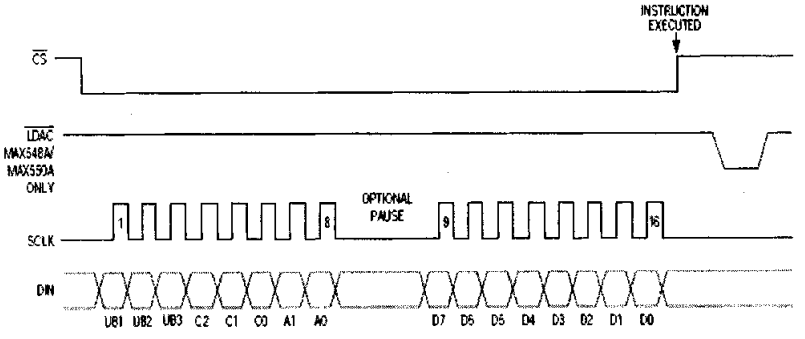


图 5.4 MAX550 工作时序图

由图 5.4 可以看出，当要只有/CS 为低电平有效时，芯片才能工作。每传送一字节数字之前，必须先传输一字节的命令设置 MAX550 的工作模式。

MAX550 串行接口编程命令表如图 5.5 所示。

CONTROL BYTE								DATA BYTE	LDAC	COMMAND (Commands executed on CS's rising edge)
Loaded First								Loaded Last		
UB1	UB2	UB3	C2	C1	C0	A1	A0	D7.....D0	Pin 6	
UNASSIGNED COMMANDS										
X	X	X	0	0	X	X	0	XXXXXXXX	X	Unassigned command
X	X	X	1	X	X	X	0	XXXXXXXX	X	Unassigned operation
COMMANDS LOADING INPUT REGISTER ONLY										
X	X	X	0	0	X	X	1	8-Bit DAC Data	X	Load DAC input register. DAC register unchanged.
COMMANDS LOADING DAC REGISTER										
X	X	X	0	1	0	X	0	XXXXXXXX	X	Update DAC register with current contents of input register. Input register unchanged.
X	X	X	0	1	0	X	1	8-Bit DAC Data	X	Load DAC input register and update DAC register.
X	X	X	0	1	1	X	0	XXXXXXXX	0	Update DAC register with current contents of input register. Input register unchanged.
X	X	X	0	1	1	X	1	8-Bit DAC Data	0	Load DAC input register and update DAC register.
COMMANDS UTILIZING THE ASYNCHRONOUS LOAD FUNCTION										
X	X	X	0	1	1	X	0	XXXXXXXX	1	After CS's rising edge and on LDAC's falling edge, update DAC register with current contents of input register. Input register unchanged.
X	X	X	0	1	1	X	1	8-Bit DAC Data	1	Load DAC input register. After CS's rising edge and on LDAC's falling edge, update DAC register.
COMMAND POWERING DOWN AND LOADING INPUT REGISTER ONLY										
X	X	X	1	0	X	X	1	8-Bit DAC Data	X	Load DAC input register and power down DAC.
COMMANDS POWERING DOWN AND UPDATING DAC REGISTER										
X	X	X	1	1	0	X	1	8-Bit DAC Data	X	Load DAC input register, power down DAC, and update DAC register.
X	X	X	1	1	1	X	1	8-Bit DAC Data	0	Load DAC input register, power down DAC, and update DAC register.
COMMAND POWERING DOWN AND UTILIZING THE ASYNCHRONOUS LOAD FUNCTION										
X	X	X	1	1	1	X	1	8-Bit DAC Data	1	Load DAC input register and power down DAC. While powered down, on LDAC's falling edge, update DAC register.

图 5.5 MAX550 编程命令表

例如图 5.6 为 MAX550 的一个命令，它长度为两个字节。

CONTROL BYTE								DATA BYTE							
Loaded First								Loaded Last							
UB1	UB2	UB3	C2	C1	C0	A1	A0	D7	D6	D5	D4	D3	D2	D1	D0
X	X	X	0	1	0	0	1	1	0	0	0	0	0	0	0

X = Don't care

图 5.6 MAX550 编程命令例子

根据 MAX550 芯片使用手册说明，有两种方式进行时钟控制的工作方式：一个是直接基于时钟；一个是基于 SPI。由 MAX550 工作时序图看出，采用直接基于时钟的方式，我们需要控制到位级别，即我们需要编程控制数据的每一位的传输；但对于 SCLK 所要求的频率，我们用编程语句无法做到精确控制；而基于 SPI 的方式，只设置 SPI

的工作频率和工作模式即可。所以，我们采用基于 SPI 的模式编程。

C8051F015 的片上 SPI 的时钟控制方式有四种，由 SPI0CFG.7、SPI0CFG.6 设置：00、01、10、11；我们选择 00 方式。工作频率设置为系统频率的 1/2（SPI0CKR = 0X00），SPI 的工作模式设置为主模式（SPI0CN.1 = 1）。

MAX550 的工作命令如图 5.5 所示，我们采用控制命令 09H。每次装入的命令和数据格式如图 5.6 所示。

根据 MAX550 的功能特性及编程命令，我们设计了它的基于 SPI 方式的驱动程序。这种方式我们控制到字节级别，即位的移位控制由 SPI 完成，我们无需参与。

基于 SPI 编程：

```
EXTERN_ADC_DRIVER:
    CLRCS
    CLR     SPIF
    MOV     SPI0DAT,#09H
WAIT_FINISH_0:
    JNB SPIF,WAIT_FINISH_0
    CLRSPIF
    MOV     SPI0DAT,ADCDATA_HIGH
WAIT_FINISH_1:
    JNB SPIF,WAIT_FINISH_1
    SETB    CS
    RET
```

第 6 章 IAP 技术及在系统中的应用

在第三章的 3.4 节我们已经对 IAP 技术做了简单的阐述，介绍了 IAP 技术与 ISP 技术的相同点和区别，它们各自的适用场合以及在系统中的应用。本章将详细介绍在我们的数字式控制器系统中是如何实现 IAP 技术的。

6.1 IAP 实现方法探讨

我们的数字式控制器中使用的是 Silicon Laboratories 公司的 C8051F015 芯片，该公司关于 IAP 技术也在相应文献中有些涉及。下面对使用 IAP 技术通过 UART 串口提供对 FLASH 进行重新编程的方法的实现进行简单的介绍。它有几处必须注意的地方：

1、在装载程序代码之前，必须擦除 FLASH 上原有的数据。

2、Intel Hexadecimal Object File 是一个 ASCII 文件，它包括可编程设备的程序代码空间（FLASH）的一个完整的或部分的映象。另外，在使用 Silicon Laboratories 公司的 IDE 进行文件转换时，我们必须向 Intel HEX 转换器提供一个 OMF-51 文件（二进制连接输出文件）。

3、必须能够通过一个 PC 机或任何一个其它的具有 UART 接口的嵌入式设备运行终端程序来控制 UART 代码装载器。

为了通过 UART 接口将代码装载进设备，这个设备必须得运行一个应用程序，来控制代码从主机到程序装载区的传输。这个应用程序必须能够完成以下任务：

1、设置设备参数，使其按照固定的波特率进行 UART 通信。

2、在接收代码之前擦除程序装载区（FLASH）。

3、下载新软件，并将其存储到程序装载区。

4、执行新装入的代码。

下面分别介绍一下这四个步骤，Silicon Laboratories 公司是如何实现的。

当通过 UART 接口在两个设备之间通信时，必须把它们设为相同的波特率，8bit 或 9bit 数据模式带不带奇偶校验位。例如，我们使用 8bit 不带奇偶校验的模式，波特率为 115200bits 每秒。如果我们的终端程序在一台主机上，这台主机就必须按下表进行设置：

表 6.1 终端设置表

Bits per second	115200
Data bits	8
Parity	None
Stop bits	1
Flow control	None

通常来说，在存储新的下载代码时，代码装载器必须擦除一个或多个 512 字节的 FLASH 页。擦写 FLASH 的方法根据设备的型号不同而可能不同。设备型号的详细信息需要参阅设备说明书的 FLASH 存储器部分。

一旦代码装载器已经擦除了一个或多个 FLASH 页，它就会提示用户发送新代码。主机可以使用多种方式对新代码进行加密编码，只要代码装载器能够解码和解释这些信息。一种好的方法就是使用 Intel Hexadecimal Object File 格式。Intel HEX 文件是一个 ASCII 文件，包括了可编程设备的程序代码空间（FLASH）的一个完整的或部分的映

象。这个文件由连接输出文件经过 Silicon Labs IDE 安装所带的 OH51 而产生的。

当新代码被存储到 FLASH 以后，就能使用一个函数指针来调用它。函数指针根据不同的编译器由不同的实现方式。具体信息可参看所使用编译器的编译文档。在 KEIL 51 编译器里，函数指针是一个三字节的通用指针。通用指针的第一个字节指出了存储器的段，后两个字节指出了地址。

任何的代码装载的实现必须至少包括两个项目，一个是代码装载器，另一个被装载的代码，让这两个不相关的项目共享同样的资源是有一定的难度的。在开始任何的多项目的工作之前确信你已经熟悉了编译器可连接器的文献，特别是与定位段相关的章节。当使用多项目时，最主要的事情是阻止数据段和代码段的重叠。代码段不允许在共享的 FLASH 页在没有擦除 FLASH 的情况下下载第二个项目。数据段不能重叠，以免出现执行的混乱。如果多个项目的变量被存储在一个共同区域，它们将互相破坏数据。阻止段代码的重叠相当简单；但要阻止数据段的重叠却相当具有挑战性，并且很难编译。下面是阻止段重叠的三个方法。

1、第一就是用代码连接器命令行参数“CODE”绝对定位代码段。为了阻止“DATA”段的重叠，函数使用“OVERLAY”命令行连接参数手工编译，这个方法相当复杂，对于需要额外存储器的大项目为了避免段重叠需要预留较大的存储容量。

2、第二个方法就是在为所有的项目声明并绝对定位一系列的全局变量，这些全局变量仅被要装载的项目所使用。实现这个的一个简单的方法就是定义一个头文件，来保存这些全局变量，然后再把这个

头文件包含到各个项目中去。代码段用连接参数“CODE”来绝对定位。

3、第三个方法是声明所有的局部变量为静态的。一旦被装载代码的项目建立，MAP 文件就检查数据段大小，并为这个段声明并绝对定位一个相同大小的空间。需要特别注意的是数据段必须用“DATA”参数绝对定位，以确保它不会四处移动。代码段同样需要用参数“CODE”绝对定位。

根据上面的应用分析，一般的 IAP 实现都采用技术 3 实现。但这种方法同样存在难以实际应用的缺点，它没有对 FLASH 低端的中断向量表进行处理，故在把装载程序固化到低端以后，中断向量将无法使用。所以我们需要对其进行改进。

6.2 系统中 IAP 的实现

我们对上面的 IAP 技术的实现进行改进的一个重要技术就是客户端微控制器的中断向量入口地址的重定位技术。下面详细介绍一下这一技术。

6.2.1 微控制器复位和中断入口地址的重新定位

根据 C8051F 微控制器锁定和访问限制特性，我们的装载软件只能固化在片上 FLASH 程序存储器的低端，而 8051 内核的微控制器的复位向量和中断入口地址是由硬件决定的，也位于程序存储器的低端，因此必须将复位和中断入口的地址重新定位。复位和中断入口的地址重新定位的方法如下。

(1) 8051 内核的微控制器的复位向量和中断入口地址

标准8051微控制器的复位向量为0X0000，中断入口地址分布情况如表6.2所示。

表6.2 8051中断向量地址分布情况

中断号	中断名称	中断入口地址
0	外部中断0	0003H
1	定时器0	000BH
2	外部中断1	0013H
3	定时器1	001BH
4	串口UART	0023H

表6.3 C8051F015微控制器中断入口地址的分布情况

中断号	中断名称	中断入口地址	中断号	中断名称	中断入口地址
	复位	0000H	10	COFE	0053H
0	外部中断0	0003H	11	CORE	005BH
1	定时器0	000BH	12	C1FE	0063H
2	外部中断1	0013H	13	C1RE	006BH
3	定时器1	001BH	14	定时器3	0073H
4	UART	0023H	15	ADC0 END	007BH
5	定时器2	002BH	16	外部中断4	0083H
6	SPI	0033H	17	外部中断5	008BH
7	SMBUS	003BH	18	外部中断6	0093H
8	ADC0 WC	0043H	19	外部中断7	009BH
9	PCA0	004BH	21	EC OSC R	00ABH

在表6.2中，中断入口地址和中断号之间的关系为：

中断入口地址 = 8 X 中断号 + 3

对于内核为 8052 的微控制器，包括 C8051F 微控制器，其中断入口地址和中断号之间的关系也满足上式。表 6.3 所示的是 C8051F015 微控制器中断入口地址的分布情况。

(2) C51 编译器产生的中断函数的入口地址

表 6.4 C51 编译器生成的中断函数入口地址的分布情况表

中断号	中断入口地址	中断号	中断入口地址
0	0003H	16	0083H
1	000BH	17	008BH
2	0013H	18	0093H
3	001BH	19	009BH
4	0023H	20	00A3H
5	002BH	21	00ABH
6	0033H	22	00B3H
7	003BH	23	00BBH
8	0043H	24	00C3H
9	004BH	25	00CBH
10	0053H	26	00D3H
11	005BH	27	00DBH
12	0063H	28	00E3H
13	006BH	29	00EBH
14	0073H	30	00F3H
15	007BH	31	00FBH

C51 编译器最多可支持 32 个中断函数，并支持重新定位。中断函

数入口地址和中断号之间的关系为：

中断函数入口地址 = interval X 中断号 + offset + 3

在默认的情况下，interval 等于 8、offset 等于 0，此时 C51 编译器生成的中断函数入口地址的分布情况如表 6.4。

(3) 复位和重新定位的编程实现

在下面的说明中，interval 等于 8，而 offset 为锁定和访问限制的最高地址＋1，在 C8051F015 微控制器中 offset 必须为 512 的整数倍。

①微控制器的复位向量重新定位的方法

```
ORG   0
JMP   START
.....
ORG   offset
START:
```

②微控制器的中断入口地址重新定位的方法

```
ORG   8 X 中断号 + 3
JMP   int_中断号
.....
ORG   8 X 中断号 + offset + 3
int_中断号:
    .....
```

(4) 应用程序的启动和中断入口地址重新定位

对于用 A51 进行编程时，实现应用程序的启动和中断入口地址的重新定位是相当容易的，只要使用 CSEG AT 绝对段定义即可。

对于用 C51 进行编程时，实现应用程序的启动和中断入口地址的重新定位也并不复杂。编译器生成的中断函数入口地址重新定位非常简单，只要加入#pragma iv（offset）编译参数。有了此编译参数，C51 编译器在生成中断函数入口时将中断函数入口地址重新定位至 8X 中断号 +offset + 3 处，正好与 C8051F015 微控制器重新定位的中断入口地址重合。而对于应用程序启动的重新定位必须一段初始化程序。

6.2.2 IAP 客户端在 C8051F015 微控制器上的实现

本节我们将详细介绍我们在单片机端所采用的编程技术及实现方法。结合 6.2.1 节我们提出的中断向量的入口地址的重定位技术，以及根据单片机与 PC 机通过 UART 通信的要求，我们设计的两者之间的通信握手协议，我们设计了客户端微控制器的任务划分和软件框架。二者之间的握手协议我们将在下节介绍。表 6.5 所示为单片极端的任务划分。

表 6.5 IAP 单片机端任务划分表

任务名称	任务描述
MCU_INI ()	MCU 初始化
Tick_wdt ()	定时器控制
Recvive_ok ()	接收正确处理函数
Recvive_err ()	接收错误处理函数
Receive ()	接收处理函数
Transmit ()	发送函数

软件总体框架为：

```
void main(void)
{
    MCU_INI ();
    LCD_FUNCTION ();
    Recvive_ok ();
    while (1)
    {
        Tick_wdt ();
        if (RI) Receive ();
        if (TI) Transmit ();
        if (TF2) Recvive_err ();
    }
}
```

中断移址技术的实现如程序 6－1 所示：

程序 6－1：

```
INT_INI MACRO
        IRPX,<0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,
        20,21>
         ORG X*8+3
         JMP X*8+3+1000H
         ENDM
    ENDM
EXTRN CODE (?C_START)              ;外部函数，连接程序引导 MAIN 函数
PUBLIC ?C_STARTUP
```

```
    NAME    ?C_STARTUP
?C_C51STARTUP   SEGMENT CODE
?STACK      SEGMENT IDATA
        RSEG    ?STACK
        DS  1

    cseg at 0
?C_STARTUP: jmp ?C_STARTUP1
    INT_INI  ;中断向量重新定位于 X*8+3+1000H, X 为中断号,;X=0-21
    RSEG ?C_C51STARTUP
?C_STARTUP1:
    MOV FLSCL,#087h
    MOV PSCTL,#1
    mov dptr,#7dfeh
    mov a,#0feh
    clr a
    inc dptr
    MOV PSCTL,#0
    MOV FLSCL,#8fh
    clr a
    mov dptr,#8029h
    movc a,@a+dptr
    cjne a,#0a5h,CLEAR_IDATA0
    sjmp ?C_STARTUP2
```

```
CLEAR_IDATA0:
    MOV R0,#0FFH
    CLR A
CLEAR_IDATA:
    MOV @R0,A
    DJNZ R0,CLEAR_IDATA
    MOV DPTR,#0
    MOV R7,#0
    MOV R6,#8
CLEAR_XDATA:
    MOVX @DPTR,A
    INC DPTR
    DJNZ R7,CLEAR_XDATA
    DJNZ R6,CLEAR_XDATA
    MOV SP,#?STACK-1
    jmp ?C_START
?C_STARTUP2:
    jmp 1000h              ;复位向量重新定位于1000H

    cseg at 8029h
    db 0
    END
```

如程序6－1所示，黑体表示了中断中断向量的入口地址的重定位技术的实现。复位向量被我们重新定位在1000H。此地址以下的低端

地址我们固化 IAP 协议下载程序。中断向量入口地址也被我们通过宏 INI_INI 重新定位到 1000H 以上。我们还通过单片机的加密解密位对低端存储的固件所在的 FLASH 区域进行了锁定，以防外人读写和串改程序。我们在第三章介绍过这方面的问题，读者可以参阅。

6.2.3 IAP 服务器端在 PC 机上的实现

在 PC 机端我们根据制定的一套握手协议制作了一个交互式界面的软件下载器。它可以在 WINDOWS 环境下使用。界面如图 6.1 所示。

图 6.1　PC 机端软件下载器用户界面

这个软件具有文件加密、端口选择、密码管理、产品序列号输入以及文件传输等功能。数传终端发送升级程序信令，PC 机收到并回复确认信息后，发送握手信息；收到正确应答后后开始下载代码到数传终端。按页顺序号发送、接收数据，校验帧的正确性，采用握手数

据流保证每页完全正确接收、存储。代码发送完毕后，发送下载完毕信令。每阶段都采取延时处理：延时时间到继续重发过程三次，还没收到就报告出错信息，强行下线，切换到待升级用户程序运行。如果通信正常，就完成了我们软件在线编程和升级。图 6.2 为我们软件下载器的版本信息。

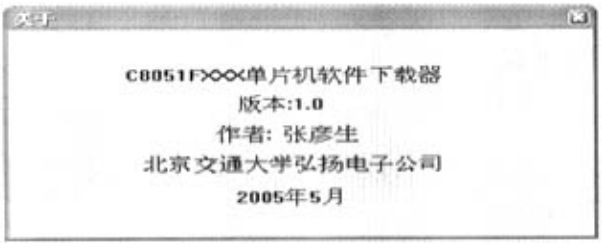


图 6.2 单片机软件下载器版本信息

附录 1 为单片机与 PC 机之间的握手协议，其过程采用类似于 TCP/IP 的三次握手形式。PC 机按每页 512B 大小发送数据信息。使用 VC 通信控件 MSComm 与单片机通信，利用 PC 机的串口，波特率为 56000bps，8 位数据位，握手时无奇偶校验，当连接建立后开始传输数据时将串口设为偶校验。对数据传送的验证，采用数据和的校验方式，对每次传送的 512 个字节计算它们的总和，在数据末尾发送过去。单片机端同样对数据进行计算和。再与传送过来的和进行比较，相同则数据传送正确，发送＃OK 到 PC 机；不同则数据传送错误，发送＃ER 到 PC 机。PC 机收到收到＃OK，就发送下一页数据；收到＃ER，则重发本页数据。每页传送间隔采用定时器控制，当发送超时或等待校验信息超时时，也重发本页数据。具体协议请参阅附录 1。

结束语

铁路公寓叫班系统的数字式控制器采用先进的数字信号处理理论，实现了控制器系统的数字化，提高了系统可靠性。在硬件方面，它采用价格较低但片上资源丰富的混合处理器－C8051F 系列单片机，并采用软件模拟技术，减少系统外围器件的使用，减小了体积，提高了系统的集成度，具有较高的性价比。相对于模拟控制器，数字控制器具有以下优点：

可靠性提高。用数字处理系统替代了模拟处理系统，增强了系统的抗干扰性，提高了系统的可靠性。

性价比高。数字式控制器中我们选用价格相对便宜的 C8051F 系列单片机，没有采用功能较强的数字信号处理器 DSP，却大大减少了外围元器件的使用，使得控制器体积减小，集成度增强，可靠性提高。与原来的模拟控制器相比，性价比较高。

产品更新方便。在数字式控制器中，我们增加了 IAP 程序装载技术，使用嵌入式系统 BOOTLOADER 原理，在存储器的低端固化一个接收引导装载程。当单片机芯片上电时，通过一整套的握手协议与主机建立通信通道，下载单片机系统程序，并写至 FLASH。完成后，将控制权转交给系统程序，自动引导运行系统程序。这样我们可以轻松地完成控制器的软件更新，进行功能升级。

维护方便。外围电路的减少及可靠性的提高，使得产品的售后维护简单易行。

由于研究时间及市场推广的紧迫性，使得数字式控制器的设计还存在不是太理想的地方。如在控制器与 PC 机通信的接口方式上，我

们使用了简单的串口通信方式，使得对通话录音时，每次只能对一个通道录音。由于串行口传输速率的限制，不能同时对几个通道同时录音。所以，我们可以考虑使用传输速率较高的USB通信方式。利用它的高速率传输性能，可以同时对多路声道进行监控录音。

这有待于以后的研究工作中进一步探讨。

致 谢

经过一个多月的撰写，论文终于完成了。本论文是在我的导师陈连坤副教授的耐心指导下完成的。本论文从选题、撰写、字斟句酌的修改，一直到最后定稿，倾注了陈老师大量的心血。他渊博的知识、创新的思想、严谨的治学态度和精益求精的工作精神使我获益菲浅。

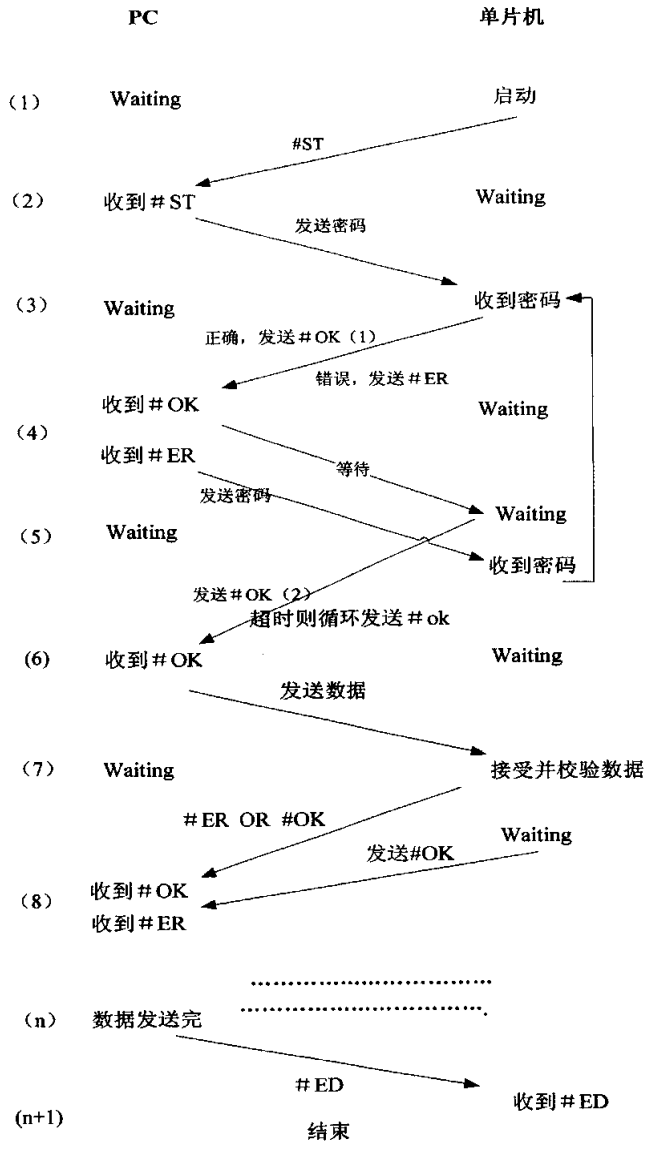
在两年半的研究生学习期间，陈老师在生活、科研方面，都给予我无微不至的关怀。在我学习研究遇到困难时耐心指导我，为我提供了难得的学习和锻炼机会，使我在理论水平和实践能力各方面都得到了很大的提高，衷心感谢陈老师！

感谢交大，感谢她在我的研究生学习期间提供的良好学习氛围和生活环境；感谢图书馆，感谢她提供了如此丰富的学习资料供我查阅汲取。

真心地感谢我实验室的每一位同学。在日常生活和科研学习过程中，他们与我互相学习，共同探讨，解决科研过程中的难点；在生活中给予了我无私的帮助和关怀。

最后，感谢所有的在研期间曾给予我支持、关心和帮助的各位老师、同学，感谢他们对我的学习和论文工作所给予的无私帮助。攻读硕士学位期间我所获得的每一点进步都与他们的深切关怀和热情帮助分不开的。再次感谢他们!

附录 1：IAP 通信握手协议



附录 2：软件实现数字化音量控制程序

```
;*********************************************************
;**  Copyright (a)2004-2005  HYC                        **
;**  All rights reserved.                               **
;**  文件名称：CONTROL_VOICE.ASM                        **
;**  简要描述：音量控制函数                             **
;**  作者：张彦生                                       **
;**  完成日期：JUNE 26 2005                             **
;*********************************************************
#include  <C8051F000.INC>
NAME    VOICE_CONTROL

PUBLIC   _VOICE_CONTROL
PUBLIC    _CONTROL_VOICE_INIT
PUBLIC   DACDATA_HIGH
PUBLIC   DACDATA_LOW
MIDDLE_VOLTAGE       EQU    0X7D00
EXTRN    DATA (ADCDATA_LOW)
EXTRN    DATA (ADCDATA_HIGH)

VOICE_CONTROL_CODE       SEGMENT     CODE
VOICE_CONTROL_DATA       SEGMENT     DATA
RSEG        VOICE_CONTROL_DATA
DACDATA_HIGH:    DS      1
```

```
DACDATA_LOW:        DS      1
RSEG        VOICE_CONTROL_CODE
USING       2
;音量控制函数采用第二组寄存器完成，其中用到了 R7,R6,R4,R2
;R7 保存得是要将音量放大的倍数。R4R6 保存的是转换后的音频数据，R4 保存；高位，R6 保存低位。R2 用来作为对音量放大倍数进行操作的单元。
;-------------------------------------------------------------------------------
_VOICE_CONTROL:
    PUSH    PSW
    SETB    RS1
    CLR     RS0
    MOV     R6,ADCDATA_LOW
    MOV     R4,ADCDATA_HIGH
    MOV     A,R7
    MOV     R2,A
    CJNE    A,#00,CONTINUE
    JMP     FINISH
CONTINUE:
    JNB     ACC.7,VOICE_LOUDER
    CPL     A
    INC     A
    MOV     R2,A
    JMP     VOICE_SMALLER
VOICE_SMALLER:
```

```
    MOV    A,R4
    CLR    C
    SUBB   A,#HIGH(MIDDLE_VOLTAGE)
    MOV    R4,A
SHIFT_2_RIGHT:
    MOV    A,R4
    MOV    C,ACC.7
    RRC    A
    MOV    R4,A
    MOV    A,R6
    RRC    A
    MOV    R6,A
    DJNZ   R2,SHIFT_2_RIGHT
    MOV    A,R4
    ADD    A,#HIGH(MIDDLE_VOLTAGE)
    MOV    R4,A
    JMP    FINISH

VOICE_LOUDER:
    MOV    A,R4
    CLR    C
    SUBB   A,#HIGH(MIDDLE_VOLTAGE)
    MOV    R4,A
    JC  SHIFT_2_LEFT2
```

```
SHIFT_2_LEFT1:
    MOV     A,R6
    RLC     A
    MOV     R6,A
    MOV     A,R4
    RLC     A
    JC      MAX_DEAl
    MOV     R4,A
    DJNZ    R2,SHIFT_2_LEFT1
    ADD     A,#HIGH(MIDDLE_VOLTAGE)
    JC      MAX_DEAL
    MOV     R4,A
    JMP     FINISH

MAX_DEAL:
    MOV     R4,#0FFH
    MOV     R6,#0FFH
    JMP     FINISH
SHIFT_2_LEFT2:
    MOV     A,R6
    RLC     A
    MOV     R6,A
    MOV     A,R4
    RLC     A
```

```
    JNC     MIN_DEAl
    MOV     R4,A
    DJNZ    R2,SHIFT_2_LEFT2
    ADD     A,#HIGH(MIDDLE_VOLTAGE)
    JNC     MIN_DEAL
    MOV     R4,A
    JMP     FINISH

MIN_DEAL:
    MOV     R4,#00H
    MOV     R6,#00H
FINISH:
    MOV      DACDATA_LOW,R6
    MOV      DACDATA_HIGH,R4
    POP     PSW
    RET

_CONTROL_VOICE_INIT:
    PUSH    PSW
    MOV     17H,R7        ;传递参数。R7 为放大倍数
    POP      PSW
    RET
;-------------------------------------------------------------------------------
    END
```

附录 3：软件 UART 串行接收程序

```
**************************************************************
**Copyright(a)2004-2005   HYC                                **
**All rights reserved.                                       **
**文件名称：INT300_REC.ASM                                   **
**简要描述：串行数据接收函数，模拟 UART 接收功能,接收串行数据**
**作者：张彦生                                               **
**完成日期：SEP 18   2005                                    **
**************************************************************
#include <c8051f000.inc>
NAME   INT300_ISR_MODEL

EXTRN  BIT(INT300)
INT0_ISR_CODE      SEGMENT      CODE   UNIT
TIMER0_ISR_CODE      SEGMENT      CODE   UNIT
TIMER0_ISR_BIT       SEGMENT      BIT
RSEG       TIMER0_ISR_BIT
INT300_RECEIVE_FLAG:  DBIT       1       ;0:DELAY T/2
INT300_REC_FINISH_FLAG: DBIT     1   ;一个字节接收完成标识
INT300_HEAD_FINISH_FLAG:DBIT     1   ;特征头接收完成标识位
INT300_HIGH_OR_LOW_FLAG:DBIT     1   ;0:高位字节,1:低位字节
TIMER0_ISR_DATA      SEGMENT      DATA
             RSEG       TIMER0_ISR_DATA
INT300_COUNTER:      DS      1
```

```
INT300_DATA_HEAD:       DS      1
INT300_DATA_HIGH:       DS      1
INT300_DATA_LOW:        DS      1
INT300_DATA_SHIFT_COUNT:DS      1
INT300_DATA_REC_BUFFER: DS      1

        CSEG        AT      0003H
                JMP     _INT0_ISR
        CSEG        AT      000BH
                JMP     _TIMER0_ISR
;-------------------------------------------------------------------------------
RSEG    INT0_ISR_CODE
_INT0_ISR:
    CLR     EX0             ;关闭 int0 中断
    CLR     IE0
    MOV     INT300_COUNTER,#03
    CLR     INT300_RECEIVE_FLAG
    MOV     INT300_DATA_SHIFT_COUNT,#08  ;每个字节八位
    SETB    TR0             ;启动 timer0 中断
    RETI
;-------------------------------------------------------------------------------
RSEG    TIMER0_ISR_CODE
_TIMER0_ISR:
    PUSH    PSW
```

```
    PUSH    ACC
    JB   INT300_RECEIVE_FLAG,INT300_DATA_RECEIVE
                                ;数据字节是起始位还是数据位。
INT300_CONTINUE_DETECT:         ;如果是起始位，检测是否为低电平
    JB   INT300,NEXT_INT300_INT0_ISR            ;连续检测三次
    DJNZ    INT300_COUNTER,INT300_CONTINUE_DETECT
    SETB    INT300_RECEIVE_FLAG
    JMP     INT300_EXIT
INT300_DATA_RECEIVE:
    MOV     C,INT300
    MOV     A,INT300_DATA_REC_BUFFER
    RRC     A
    MOV     INT300_DATA_REC_BUFFER,A
    DJNZ    INT300_DATA_SHIFT_COUNT,INT300_EXIT
    ;一个字节已经接收完成
    JNB INT300_HEAD_FINISH_FLAG,INT300_HEAD_DEAL_CODE
    JB   INT300_HIGH_OR_LOW_FLAG,INT300_LOW_BYTE_CODE
    MOV     INT300_DATA_HIGH,INT300_DATA_REC_BUFFER
    SETB    INT300_HIGH_OR_LOW_FLAG
    JMP     NEXT_INT300_INT0_ISR

INT300_HEAD_DEAL_CODE:
;   MOV     A,INT300_DATA_REC_BUFFER
    ANL     A,#1111$0000B
```

```
    CJNE    A,#0F0H,INT300_HEAD_WRONG_CODE
    MOV     INT300_DATA_HEAD,INT300_DATA_REC_BUFFER
    SETB    INT300_HEAD_FINISH_FLAG
INT300_HEAD_WRONG_CODE:
    JMP     NEXT_INT300_INT0_ISR

INT300_LOW_BYTE_CODE:
    MOV     INT300_DATA_LOW,INT300_DATA_REC_BUFFER
    CLR     INT300_HIGH_OR_LOW_FLAG
    SETB    INT300_REC_FINISH_FLAG
NEXT_INT300_INT0_ISR:
;接收完一个字节，要重新启动 INT0 中断并复位定时器 0
    CLR     TR0
    MOV     TL0,#0D8H
    MOV     TH0,#0B0H
    SETB    EX0
INT300_EXIT:
    POP     ACC
    POP     PSW
    RETI
;-------------------------------------------------------------------------------
    END
```

附录 4：软件 UART 串行发送程序

```
;*****************************************************************
;**Copyright(a)2004-2005   HYC                                   **
;**All rights reserved.                                          **
;**文件名称：COMT_SEND_COMMAND.ASM                               **
;**简要描述：串行数据发送函数，模拟 UART 发送功能，发送串行数据**
;**作者：张彦生                                                  **
;**完成日期：SEP 31   2005                                       **
;*****************************************************************
#include   <c8051f000.inc>
NAME       COMT_SEND_MODEL

EXTRN  DATA(TR_TERM_DATA0)
EXTRN  DATA(TR_TERM_DATA1)
EXTRN  DATA(TR_TERM_DATA2)
EXTRN  BIT(COMT)
COMT_SEND_CODE      SEGMENT   CODE   UNIT
TIMER2_ISR_CODE     SEGMENT   CODE   UNIT
COMT_SEND_DATA      SEGMENT       DATA   UNIT
            RSEG        COMT_SEND_DATA
TR2_SHIFT_TIME:     DS      1
COMT_SEND_MULDATA: DS       1
COMT_ADJUST_FREQUENCY:      DS      1
COMT_SEND_BIT       SEGMENT       BIT
```

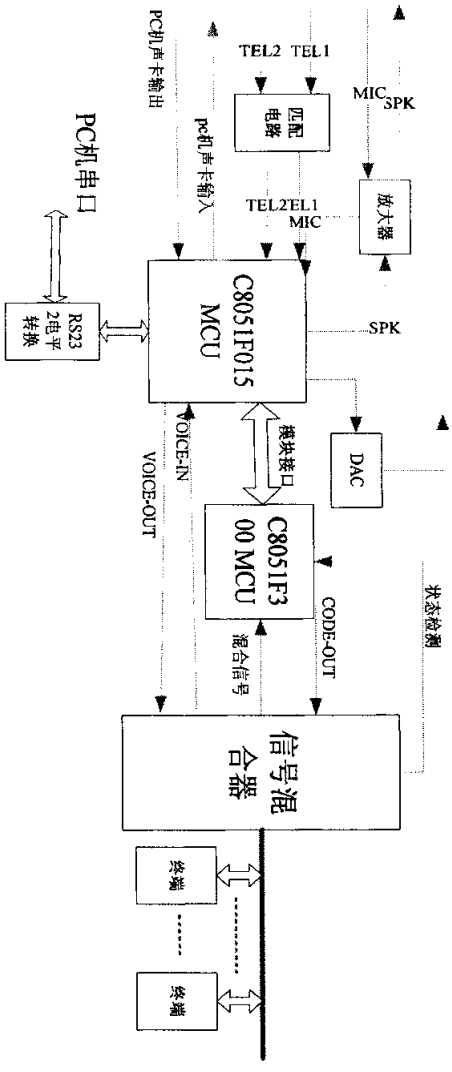
```
RSEG        COMT_SEND_BIT
COMT_SENDBYTE_FINISH_FLAG:DBIT        1
COMT_ADJFRE_FLAG:    DBIT        1
;-------------------------------------------------------------------------
        CSEG    AT  002BH
        JMP     _TIMER2_ISR
;-------------------------------------------------------------------------
RSEG    COMT_SEND_CODE
_COMT_SEND_COMMAND:
    MOV     COMT_SEND_MULDATA,R7
    SETB    COMT_ADJFRE_FLAG
    MOV     TR2_SHIFT_TIME,#09
    CLR     COMT_SENDBYTE_FINISH_FLAG
    MOV     COMT_ADJUST_FREQUENCY,#04
    SETB    TR2
    RET

RSEG    TIMER2_ISR_CODE
_TIMER2_ISR:
    PUSH    PSW
    PUSH    ACC
    CLR     TF2
    MOV     A,COMT_ADJUST_FREQUENCY
    CJNE    A,#00,COMT_ADJUSTFREQU_CODE
```

```
    JMP     COMT_DATA_TRANSMIT
COMT_ADJUSTFREQU_CODE:
    DEC     COMT_ADJUST_FREQUENCY
    JNB     COMT_ADJFRE_FLAG,FRE_COMT0_DEAL
    SETB    COMT
    CLR     COMT_ADJFRE_FLAG
    JMP     TR2_ISR_EXIT
FRE_COMT0_DEAL:
    CLR     COMT
    SETB    COMT_ADJFRE_FLAG
    JMP     TR2_ISR_EXIT
COMT_DATA_TRANSMIT:
    DJNZ    TR2_SHIFT_TIME,NOT_FINISH_BYTE
    CLR     TR2
    CLR     COMT
    MOV     TL2,#00H            ;2400bps
    MOV     TH2,#0E2H
    SETB    COMT_SENDBYTE_FINISH_FLAG
    JMP     TR2_ISR_EXIT
NOT_FINISH_BYTE:
    MOV     A,COMT_SEND_MULDATA
    RLC     A
    CPL     C
    MOV     COMT,C
```

```
    MOV     COMT_SEND_MULDATA,A
TR2_ISR_EXIT:
    POP     ACC
    POP     PSW
    RETI
    END
```

附录 5：控制器系统整体结构图



参考文献

[1] 陈连坤. 嵌入式系统的设计与开发[H]. 北京：清华大学出版社，北京交通大学出版社，2005

[2] 张树京. 通信系统原理[H]. 北京：中国铁道出版社，1992.

[3] 吴湘淇. 信号、系统与信号处理的软硬件实现[H]. 北京：电子工业出版社，2002.

[4] 程佩青. 数字信号处理教程[H]. 北京：清华大学出版社，1995

[5] 王瑞英等. 数字信号处理基础[H]. 北京：中国铁道出版社，1997.

[6] 田社平等. 基于数字陷波滤波的正弦波测量方法[J]. 计量技术，2002(9)

[7] 求是科技. 单片机典型模块设计实例导航[H]. 北京：人民邮电出版社，2004.

[8] 余相俊. 微机检测与控制应用系统设计[H]. 北京：北方交通大学出版社，2001.

[9] 孙涵芳，徐爱卿. MCS-51/96 系列单片机原理及应用[H]. 北京：北京航空航天大学出版社，1987.

[10] 李刚，林凌. 与 8051 兼容的高性能、高速单片机—C8051Fxxx[H]. 北京：北京航空航天大学出版社，2001.

[11] 徐爱钧，彭秀华. 单片机高级语言 C51 应用程序设计[H]. 电子工业出版社，2005

[12] 王田苗. 嵌入式系统设计与实例开发[H]. 北京：电子工业出版社，1999.

[13] 张彦生，陈连坤. 一种从数模混合信号中提取数字信号的时域算

法[J]．现代计算机，2005.7

[14]郭慧青．便携式心电测试仪的软件设计及其相关理论的研究[J]．北京交通大学，2005.4

[15]王文辉，刘淑英.电路与电子学[H].电子工业出版社，2000.1

[16]王永军，从玉珍.数字逻辑与数字系统[H]．电子工业出版社，2000.1