苏州大学

硕士学位论文

基于神经网络和模糊控制的车载厢体风机智能控制系统的设计与实现

姓名：陈立平

申请学位级别：硕士

专业：计算机技术

指导教师：徐汀荣

20070501

摘 要

本文设计并实现了一种基于神经网络和模糊控制的车载厢体风机智能控制系统。

首先，介绍了课题的研究现状、课题的研究目标、论文的主要工作及特色，以及论文的章节安排。

然后，对本论文中的关键技术进行详细的介绍。分别详述了神经网络 BP 算法、模糊控制、远程通信和嵌入式系统的基本概念、基本原理及其应用。

根据风机控制系统的功能要求，介绍了系统硬件设计与实现和系统软件设计与实现。本文设计并实现了基于神经网络和模糊控制的车载厢体风机智能控制系统，该系统以 AT89C55WD 为硬件平台，以 uC/OS－II 为实时操作系统。不同于一般的基于单片机的系统，该系统是一种多任务的风机控制系统。针对多点温度，该系统在 uC/OS－II 支撑平台上，创建和启动多项任务对这些温度进行连续检测，并依据参数变化采用模糊策略自动调整风机的工作状态。uC/OS－II 的运用，使系统具有较好的实时性、可扩展性和多任务处理能力；模糊控制策略的运用，使系统能较好地适应于温度参数的非线性和时变性；通过 USB 通信接口将数据和状态信息上传到上位机显示、存储，并采用前馈神经网络 BP 算法，分析温度的变化趋势，调整风机相应控制参数，使系统有效保护风机，延长风机寿命；通过无线接口在工作现场接受便携机或者 PC 机命令；同时本文提出了基于“伪固定”IP 地址的远程传输概念，以便中央服务器能集中各路数据，对温度数据做进一步集中分析。

最后是对车载厢体风机智能控制系统的厢体温度均衡度和风机工作强度进行测试，并对测试结果进行比较分析和总结。

本系统已经进入测试阶段，测试结果表明该系统已经达到了设计要求，对保护风机和延长风机寿命起到了较好的效果。

关键词：神经网络，BP 算法，模糊控制，嵌入式系统，实时操作系统，uC/OS－II，多任务，风机智能控制，伪固定 IP 地址。

Abstract

This article has designed and Implement one kind of air blower intelligence control system on compartment of automobile based on nerve network and fuzzy control.

First, briefed present research situation of the topic, the topic research aim, the paper main work and the characteristic, as well as paper arrangement of chapter.

Then, key technology in the present paper was introducted in detail. Described basic concept, basic principle and application of the nerve network BP algorithm, the fuzzy control, the long-distance correspondence and built-in system.

According to the air blower control system function requestion, introducing the system hardware design and realization , as well as the system software design and realization. This article has designed and realized the air blower intelligence control system on compartment by automobile based on the nerve network and the fuzzy control, this system take AT89C55WD as the hardware platform, take uC/OS - II as real-time operating system. Differ from the simple air blower intelligence control system base on singlechip, this system is a kind of air blower intelligence control system of many mission. Aim at the multi- spots temperature and in the platform of uC/OS—II, the system establish and start several mission to carry on a continuous examination to these temperatures, and adopt fuzzy strategy to adjust the work status of air blowe automatically according to the parameter variety. The uC/OS - II utilization enable the system to have a better performance of real-time, extention and the multitasking ability; The fuzzy control strategy utilization causes the system to adapt non-linearity and denaturation of the temperature parameter well; through the USB correspondence

interface, the data and the condition information are transmitted to Pc and saved. we can analysis temperature change tendency and adjusts the air blower corresponding control variablity by using the forward feed nerve network BP algorithm, which causes the system to protect the air blower and extend the air blower life effectively; Through wireless interface the control system can accept and take the order of PC or portables computers in the job site;Simultaneously, in order to concentrate various data on the central server and analy them furtherly, the article propose long-distance transmission concept based on disguised fixation IP address. ;

Finally the compartment temperature balanced degree and the air blower working strength of the compartment air blower intelligence control system has been tested , as well as the result of which has been compared, analyzed and summarized.

This system has already entered the test stage. The test result indicated that this system had already achieved the design request and taken out better effect in protecting the air blower and extending the air blower life

Keyword: nerve network , BP algorithm , fuzzy control , built-in system, real-time os, uC/OS—II, multitask, air blower intelligence control, disguised fixation IP address

苏州大学学位论文独创性声明及使用授权的声明

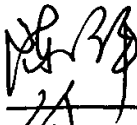
学位论文独创性声明

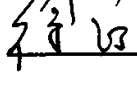
本人郑重声明：所提交的学位论文是本人在导师的指导下，独立进行研究工作所取得的成果。除文中已经注明引用的内容外，本论文不含其他个人或集体已经发表或撰写过的研究成果，也不含为获得苏州大学或其它教育机构的学位证书而使用过的材料。对本文的研究作出重要贡献的个人和集体，均已在文中以明确方式标明。本人承担本声明的法律责任。

研究生签名：  日 期：2007.5

学位论文使用授权声明

苏州大学、中国科学技术信息研究所、国家图书馆、清华大学论文合作部、中国社科院文献信息情报中心有权保留本人所送交学位论文的复印件和电子文档，可以采用影印、缩印或其他复制手段保存论文。本人电子文档的内容和纸质论文的内容相一致。除在保密期内的保密论文外，允许论文被查阅和借阅，可以公布（包括刊登）论文的全部或部分内容。论文的公布（包括刊登）授权苏州大学学位办办理。

研究生签名：  日 期：2007.5

导师签名：  日 期：

第一章 绪论

1.1 车载厢体风机控制的研究现状

风机控制是近期温度控制领域中的一个热点，应用领域非常广泛，如家庭、机房、锅炉、仓库、冷库、温室、航天、军事、交通运输等方面。随着社会发展，科技进步，人们对生活质量的要求逐步提高，人们希望在炎热的夏季能坐上空气清新而又凉爽的空调车，南方人能吃到新鲜的新疆水果，北方人能吃到新鲜的广东海鲜，等等。交通运输行业对实施恒温控制的广度和深度也相应日益提高。

在早期的车载厢体风机控制系统中一般采用单片机和多路开关、多点询问来控制温度和风机，缺点就是实时性较差。目前国内外采用了多任务嵌入式系统和变频技术，提高了实时性，节省了电能，但是两者都没有考虑风机之间的相互影响，没有预测温度取样点附近温度的变化趋势，没有保存温度变化历史数据，风机的起停与温度的变化是关系单纯，没有对温度变化趋势进一步分析。这些直接导致风机老化加速，厢体温度分布不均衡，成本开销增加。因此，设计出性价比高、有效延长风机寿命、能无线通信、保存温度变化历史数据的风机控制系统，具有深远的理论和现实意义。

1.2 嵌入式系统的现状与发展趋势

信息时代、数字时代使得嵌入式产品获得了巨大的发展机遇，为嵌入式市场展现了美好的前景，同时也对嵌入式生产厂商提出了新的挑战。从中可以刊出未来嵌入式系统的几大发展趋势[1]：

1、嵌入式开发是一项系统工程，因此要求嵌入式系统厂商不仅要提供嵌入式软硬件系统本身，同时还需要提供强大的硬件开发工具和软件包支持。

2、网络化、信息化的要求随着因特网技术的成熟、带宽的提高而日益提高，使得以往单一功能的设备如电话、手机、冰箱、微波炉等功能不再单一，结构更加复杂。

3、网络互联成为必然趋势。

4、精简系统内核、算法，降低功耗和软硬件成本。

5、提供友好的多媒体人机界面。

1.3 本文的研究目标

本课题的目标，就是要将嵌入式技术应用于风机控制系统，在这种系统的合理控制之下，能对温度变化作出快速反应，保存历史数据。并且这种嵌入式结构的系统的应用，有助于实现风机控制设备的小型化、智能化、集成化。

在车载厢体风机控制过程中，由于在不同时刻汽车处于位置不同，温度变化可能很大，导致不同时刻风机控制参数也大不相同。传统控制方法缺乏灵活性、应变性，不能根据环境温度的变化及时调整某些运行参数，难于实现自适应的自动控制。本文采用的模糊控制充分显示了自身的优越性。建立在模糊集合理论上的模糊控制，运用隶属函数来描述一些不确定的模糊变量，通过建立在模糊语言、模糊推理和模糊关系上的模糊控制规则，对非常复杂的动态规程进行模糊控制，能得到其他控制方式无法实现的控制结果。因为模糊控制算法无需建立精确的数学模型，在建立模糊控制规则时可利用人的知识和经验以及新的研究成果，让计算机模拟有经验的操作者实现对模糊事物的处理过程，进而完成传统控制方式无法处理的模糊信息，实现具有智能化的模糊控制。

同时本文利用前馈神经网络 BP 算法对保存的温度变化历史数据进行学习，获得温度变化趋势和有规律性的经验，与模糊控制有机结合，使模糊控制具有自学习、自适应能力。

为了使车载厢体能达到恒温效果，本风机控制系统涉及的温度变量也有多个，因此要求系统能根据多个取样点温度的变化情况，同时对各个风机起停进行模糊控制，这就需要将模糊控制算法与多任务机制较好地结合，以实现对多任务环境下的模糊控制。

1.4 本文的主要工作及特色

针对车载厢体风机控制，本文提出了一套嵌入式风机控制系统，该系统的原理是：在多任务实时操作系统 uC/OS－II 的调度下，创建和启动多个任务，对各点温度进行连续在线检测，并依据取样结果，运用模糊控制策略，实时调节风机的工作状态。由于采用 uC/OS－II 操作系统进行调度，使得各任务能够并行执行，大大提高了系统可靠性和实时性。uC/OS－II 是一个基于优先级、占先式、可裁减的多任务实时操作系

统，其实时性与内核的健壮性早已在大量的实际应用中得到了证实。

本文主要做了以下几项工作：

（1）设计了以 AT89C55WD 单片机为核心的嵌入式系统的硬件平台；

（2）研究了模糊控制策略的工作机理，并用 C51 语言在上述硬件平台上予以实现，运用模糊控制策略控制风机的工作状态；

（3）研究了前馈神经网络的工作机理，并用 C51 和 VC++语言在硬件平台和 PC 机上完成，实现了模糊规则的在线修改和隶属函数的自动更新；

（4）研究了 uC/OS－II 嵌入式实时操作系统，并将其修改、移植到以 AT89C55WD 为核心的硬件平台上运行；

（5）将风机控制要完成的功能细化为若干个核心任务，并根据其轻重缓急赋予不同的优先权，使各个任务在 uC/OS－II 的调度下实时、协调、高效运行；

（6）编写了嵌入式风机控制系统与 USB 存储器通信程序，为进一步的数据分析保存历史资料；

（7）设计并实现了控制系统与便携机或 PC 机之间的无线通信；

（8）提出基于“伪固定”IP 地址远通信程概念，通过网络将历史数据传输至中央服务器，编程并实现之。

本文工作的特色是：

（1）将嵌入式实时内核 uC/OS－II 应用于车载厢体风机控制系统中，对风机和温度进行多任务实时控制，大大提高了控制系统的实时性、可靠性和可扩展性；

（2）利用模糊控制算法实现了对各个风机的闭环控制；

（3）利用前馈神经网络，实现了模糊规则的在线修改和隶属函数的自动更新等；

（4）利用 USB 存储器保存大批量温度变化历史数据，为后期学习分析奠定基础；

（5）通过无线接口接受便携机或者 PC 机命令；

（6）提出并实现了基于伪固定 IP 地址远程通信；

（7）使用一个嵌入式控制器来控制多个风机和多点温度，性价比高，智能化程度高，易于实现车载厢体风机控制系统的小型化、集成化和自动化。

1.5 本文的章节安排

本文共分六章，各章内容安排如下：

第一章，概述。介绍了本课题国内外研究现状、嵌入式系统的发展趋势、本课题的研究目标以及本文的主要工作和工作特色。

第二章，系统关键技术研究。介绍前馈神经网络及其 BP 算法，阐述采用模糊控制算法控制风机状态的基本算法，简要介绍基于伪固定 IP 地址实现远程通信的基本思想，以及概述了嵌入式系统的基本原理。

第三章，系统硬件设计与实现。介绍了硬件设计方案，各单元电路的结构和原理，以及实现。

第四章，系统软件设计与实现。研究了 uC/OS－II 的结构、开发平台、uC/OS－II 移植、驱动程序、应用软件、远程通信等内容及其实现。

第五章，系统应用及性能分析。对系统的测试结果进行比较分析。

第六章，结束语。从总体上总结了本文所做的主要工作，并讨论了系统需要进一步完善的地方以及进一步的工作。

第二章 关键技术研究

2.1 神经网络 BP 算法

在车载风机控制系统中，风机的起停与转速控制是一个关键而又较难解决的问题，因为影响因素很多。风机的工作过程是，在某一时段 t 内，当检测点的温度大于额定温度时，风机启动，但是缺点很明显，检测点的温度还会继续急剧上升，到达峰值后开始下降，直至额定温度，风机停止，风机必须做更多的功来降温。本章就提出一种基于 BP 神经网络来预测未来 5 分钟各个检测点的温度变化情况的方法，并将预测结果提供给模糊控制模块。模糊控制模块采用模糊控制算法根据温度值变化趋势进行风机转速的模糊控制。这样在检测点温度上升之前就开始启动风机，风机就可以做更少的功完成同样的降温效果，既降低了风机转数，延长风机寿命，又节约电能。

2.1.1 BP 神经网络

在 1986 年 Rumelhart 等提出误差反向传播法，即 BP(error BackPropagation)法。BP 算法一直是自动控制上最重要、应用最多的有效算法之一。

1、单个神经元

单个神经元由线性元件和阀值元件组成，如图 2.1 所示神经元 J。

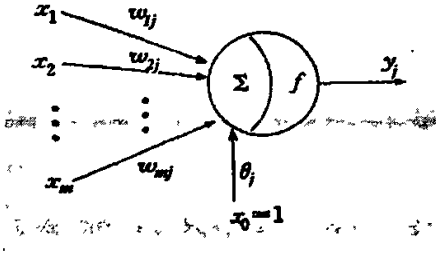


图 2.1 单个神经元

其中：xi(i=1,2,…,m)为多个输入，yj 为单个输出，θj 为阈值，wij 为从神经元 i 到神经元 j 的连接权重因子，f()为传递函数，或称激励函数，可以选择线性函数，但通常选用非线性函数（如阶跃函数、Sigmoid（S 型）函数、高斯型函数等）。

输入和输出的关系可表示为：

$$\begin{cases} s_j = \sum_{i=1}^{m} w_{ij} x_i - \theta_j \\ y_j = f(s_j) \end{cases}$$

2、BP 神经网络

BP 网络采用的是有监督学习，设一个 BP 神经网络有 L 层对应每层有（n1、n2、…nL）个神经元，如图 2.2 所示

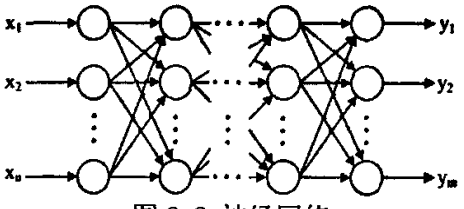


图 2.2 神经网络

给定 N 个样本（xk, yk），对每个训练样本网络训练误差为：

$$E_k = \frac{1}{2} \sum_{l=1}^{L} \sum_{j=1}^{n_l} (y_{lj} - \bar{y}_{lj})^2$$

其中 $\bar{y}_{lj}$ 为实际输出值，总误差为：

$$E = \frac{1}{2N} \sum_{k=1}^{N} E_k$$

反向传播算法就是不断修正网络权初值，直至总误差降低到可接受范围，修正权值公式如下：

$$\omega_{.ij} = \omega_{.ij} - \mu \frac{\partial E_k}{\partial \omega_{.ij}}$$

μ 为学习速率，一般 μ =0.01 至 1。

3、改进型 BP 神经网络

每次调整的网络权值都通过数据结构保存下来的 BP 算法。

2.1.2 传统温度问题描述及 BP 神经网络设计

1、传统温度问题描述

传统车载厢体的温度检测和控制可以通过单片机来实现。以三个检测点为例，分别放 3 个温度传感器，3 台风机，分别与单片机相连，并由单片机进行统一控制。如

图 2.3 所示。

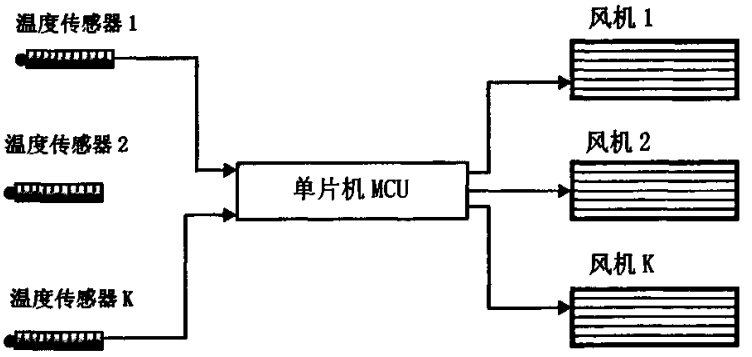


图 2.3 单片机温度控制系统

单片机每隔 1 分钟去按顺序读取各个检测点温度传感器的数据，然后分别与设定额定温度值 t0 进行比较，如果大于等于 t0 则启动相应风机，进行降温。

根据该厢体历史数据，绘制出 06 年 10 月 18 日上午 8：30 到 9：00 的 1#检测点温度变化折线图，如图 2.4 所示

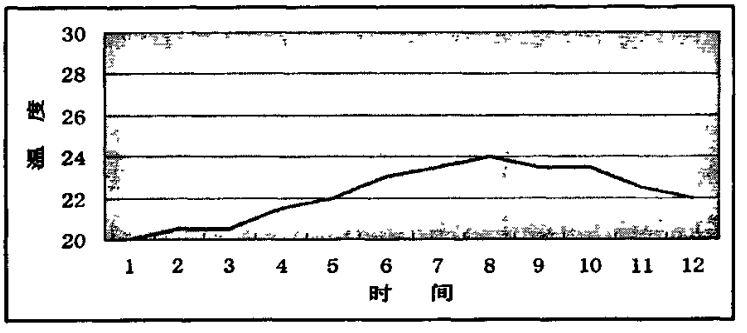


图 2.4　1＃检测点温度变化折线图

从图中可以分析得到，当 1#检测点的温度大于设定的额定温度值 70 摄氏度时，对应 1#风机启动。但是 1#点的温度并不立刻下降，而是继续上升至 75 度才开始下降。这样，一方面 1＃风机要做更多的功来降温 5 度，另一方面 75 度或者更高的温度会对特种金属的质量产生一定的影响。所以，提出了一种更优的智能解决方案。

2、BP 神经网络设计

用 t1(t+1)表示下 1 分钟 1＃检测点上的温度：

t1(t+1) = f1(t1(t),t1(t-1),t1(t-2), t1(t-3), t1(t-4), t1(t-5))

其中：t1(t)、t1(t-1)、t1(t-2)、t1(t-3)、t1(t-4)、t1(t-5)为 1#检测点上前 6 个时间单位的温度值，其他三个检测点的预测函数 f2, f3, f4 同理可得。对每一个 f 函数用一个 BP 网络逼近。

将 1#检测点上前 6 个时间单位的温度值作为 BP 神经网络的输入，下一时间单位的流量值作为输出。BP 神经网络选用三层网络即：输入层、单 Sigmoid 隐含层、线性输出层，隐含层上神经元个数的选择根据 Kolmogorov 定理选取输入层神经元个数的 2*6+1=13 个，这里为了增加预测准确性和收敛稳定性隐含层选取 15 个神经元。

2.2 模糊控制

2.2.1 模糊控制概述

自从 1965 年美国加利福尼亚大学控制论专家 L .A .zadeh 教授提出模糊数学以来”，吸引了众多的学者对其进行研究，使其理论与方法日臻完善，并且广泛地应用于自然科学和社会科学的各个领域，尤其是在第 5 代计算机研制和知识工程开发等领域占有特殊重要的地位。把模糊逻辑应用于控制领域则始于 1973 年”。1974 年英国的 E. H. Mamdani 成功地将模糊控制应用于锅炉和蒸汽机控制。此后 20 多年来，模糊控制不断发展并在许多领域中得到成功应用。由于模糊逻辑本身提供了由专家构造语言信息并将其转化为控制策略的一种系统的推理方法，因而能够解决许多复杂而无法建立精确数学模型系统的控制问题，所以它是处理推理系统和控制系统中不精确和不确定性的一种有效方法。从广义上讲，模糊控制是适于模糊推理，模仿人的思维方式，对难以建立精确数学模型的对象实施的一种控制策略。它是模糊数学同控制理论相结合的产物，同时也是智能控制的重要组成部分[19]。

2.2.2 隶属函数

用于表征模糊集合的数学工具。对于普通集合 A，它可以理解为某个论域 U 上的一个子集。为了描述论域 U 中任一元素 u 是否属于集合 A，通常可以用 0 或 1 标志。用 0 表示 u 不属于 A，而用 1 表示属于 A ，从而得到了 U 上的一个二值函数 xA(u)，它表征了 U 的元素 u 对普通集合的从属关系，通常称为 A 的特征函数，为了描述元素

u 对 U 上的一个模糊集合的隶属关系，由于这种关系的不分明性，它将用从区间[0，1] 中所取的数值代替 0，1 这两值来描述，记为 (u)，数值 (u) 表示元素隶属于模糊集的程度，论域 U 上的函数 μ 即为模糊集的隶属函数，而 (u) 即为 u 对的隶属度。

2.2.3 精确输入的模糊化

通常控制总是用系统的实际输出值与设定的期望值相比较，得到一个偏差 E,控制器根据这个偏差来决定如何对系统加以调整控制。很多情况下还需要根据该偏差的变化率 EC 来进行综合判断。无论是偏差还是偏差变化率，他们都是精确的输入值。要采用模糊控制的技术，首先把它们转换成模糊集合的隶属函数。每一个输入值都可以对应一个模糊集合，某个范围的连续变化值就可以有无限多个模糊集合，这在工程实践上是无意义的。为了便于工程实现，通常把变量范围人为的定义为离散的若干级，所定义的级数多少取决于输入量的分辨率。现在使用最多的为三角隶属函数。

为了实现模糊控制器的标准化设计，目前在实际中常用的处理方法是玛达提出的方法，将偏差 E 和 EC 的变化范围设定为[-6，+6]区间连续变化量，使之离散化，构成含 7 个整数元素的离散集合：

{NB，NM，NS，ZO，PS，PM，PB}

即：

{负大，负中，负小，不变，正小，正中，正大}

并将误差 E 和误差的变化量 EC 量化为 13 个等级

{-6，-5，-4，-3，-2，-1,0,1,2,3，4，5，6}

2.2.4 模糊控制规则和模糊推理

模糊规则一般采用最常用的模糊 if-then 规则。一个单独的模糊 if-then 规则形式如下：

If x is A then y is B

其中，A 和 B 是有模糊集合分别定义在 X、Y 范围（论域）上的语言值。模糊规则中的 if 部分“if x is A”被称为规则的前提或假设，同时 then 部分”y is B”被称为结果或结论。实质上，该表达式描述了变量 x 和 y 之间的关系。

模糊推理是建立在模糊规则之后的。它是对一个特定的表达式解释的过程。在建模时，只要将模糊输入集和模糊规则库定义为重叠的，模糊系统使用自己的归纳法则进行模糊判定，尤其是对相近的表述信息进行归纳、总结，使系统的判定变为连续的判定，从而达到连续的执行模糊法则的过程。

2.3 基于伪固定 IP 远程通信

随着计算机网络的进一步普及应用，企业又提出了新的需求，如能将温度历史数据通过网络上传到公司数据中心，做进一步分析；能在家里、办公室里或者外地看到各个控制点的相关参数，一方面是能实时了解各点的现状，另一方面又能对现场上的问题进行远程指导。这就涉及到远程通信。远程通信过程中很多情况下主服务器的 IP 地址是动态的，这是远程通信非常重要的一个问题。

为了解决远程通信问题，传统的解决方法是：在中央服务器上安装“花生壳”软件，然后去“花生壳”网站注册、申请域名，接着在服务器上运行“花生壳”软件，最后在客户端上通过域名连接中央服务器，即可。但是，这种解决方案对“花生壳”服务平台具有过分的依赖性，一旦该类公司不提供这些服务，企业就无法进行远程通信。

还有一部分企业在服务器上建立 Web 服务，将整个服务器都暴露在公网之中。这些都会对公司的各方面产生极大的影响和无形的损失。为了使企业能将主动权掌握自己的手中，免去后顾之忧，特提出以下“基于动态 IP 远程访问的解决方案”。

2.3.1 宽带路由器共享上网的工作原理

当前，很多企业和家庭都是使用宽带路由器共享上网，其最基本、最核心的技术就是 NAT 转换技术。

1)、公网 IP 地址和私有 IP 地址

在 TCP/IP 协议中，我们需要了解两种地址，一个就是可以直接访问 internet 的公网 IP 地址，另一个就是我们组建局域网时用到的私有 IP 地址。公网 IP 地址在全球网络中是唯一的，有了它，我们就可以访问 Internet 网络，而私有 IP 地址则是局域网中唯一，但对外是不可见的。

2)、公网 IP 地址与私有 IP 地址的转换

随着互联网主机的逐年增多，公网 IP 地址日益“枯竭”，很难在一个网络中申请到多个公网 IP 地址。为此，可以采用 NAT 转换技术，将局域网中使用的多个私有 IP 地址与公网 IP 地址进行转换，从而达到访问 Internet 的目的，如图 2.5 所示。

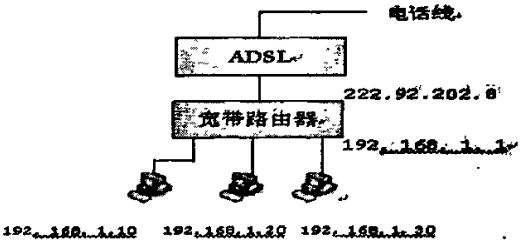


图 2.5 宽带路由共享上网

在图一宽带局域网中，ADSL 负责数据信号与模拟信号的相互转换，宽带路由器内嵌了 NAT 技术，其有 2 个 IP 地址，一个是 222.92.202.8 即对外的公网 IP 地址，另一是 192.168.1.1 即对内的私有 IP 地址，局域网中的每台机器的 IP 地址都是私有 IP 地址。假设，机器 192.168.1.10 向 Internet 中的某台公网主机发送 IP 数据包，该数据包所携带的源地址就是私有 IP 地址 192.168.1.10，当数据包到达宽带路由器以后，NAT 技术就用全球唯一的公网 IP 地址 222.92.202.8 替换掉数据包内私有 IP 地址。数据包就会使用 222.92.202.8 地址来访问外部网络了。

2.3.2 伪固定 IP 地址

目前，大多数宽带路由器对外的公网 IP 地址都是电信局动态分配的，这就使得内网中的主机对外 IP 地址一直在变化，其他公网主机就无法与之连接或者持久连接。为此，提出了“伪固定 IP 地址“解决方案。

1、端口绑定

在宽带路由器的 WEB 管理界面中，将内网中一台主机的私有 IP 地址与路由器的某个端口进行绑定，这样公网中所有发往该路由器端口的数据包，都会自动转发给与该端口绑定的内网主机。

如图 2.6 所示

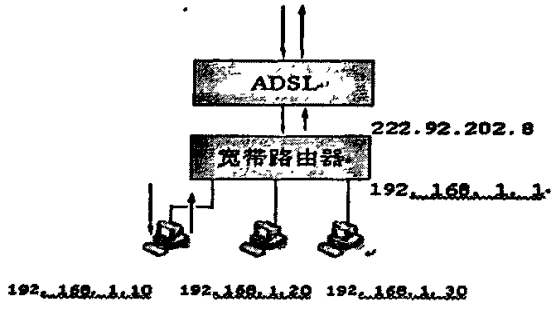


图 2.6 端口绑定访问流程示意图

2、“第三者”跳板

宽带路由器对外的公网 IP 地址是动态的，是经常变化的，其他内网主机如何获得该动态 IP 地址呢？解决方案是利用 “第三者“作为跳板，它必须具有全球唯一的公网 IP。如图 2.7 所示。

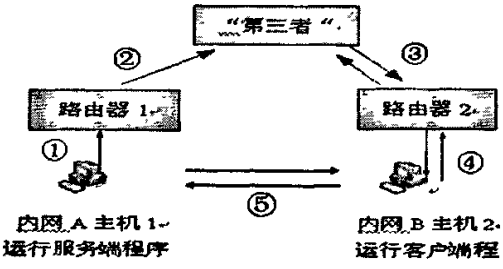


图 2.7 “第三者“访问流程示意图

（1）FTP 模式

将公网的一台预约 FTP 服务器作为“第三者“。在已绑定端口的内网 A 主机 1 上运行服务端程序，定时检测路由器 1 对外的公网 IP 地址，如果发现改变，则将此 IP 地址写入一文件，然后经由①、②上传文件至预约 FTP 服务器。在另一内网 B 主机 2 上运行客户端软件。开始连接前，先由④、③登录预约 FTP 服务器，再由③、④下载该文件，并读取 IP 地址，利用 UDP 协议与服务器进行连接、访问，即第⑤步。或者客户机通过 WINDOWS 系统自带的远程控制软件进行远程端口监控。

（2）Mail 模式

将 Internet 上一台预约 Mail 服务器作为“第三者“。在已绑定端口的内网 A 主机 1 上运行服务端程序，定时检测公网的 IP 地址，如果发现改变，然后经由①、②自动给预约 MAIL 服务器发电子邮件，该邮件标题具有一个特征值，以示唯一性和特殊性，邮件内容就是变化后的公网的 IP 地址。另一内网 B 主机 2 运行客户端软件，在连接前，先由④、③登录入预约 MAIL 服务器，再由③、④收取具有唯一特征值标题的邮件，并读取其中数据，获得内网 A 主机 1 对外的公网 IP 地址。至此就可以经由第⑤步通过如同方案一的软件进行远程访问和监控。

2.4 嵌入式系统

2.4.1 嵌入式系统概述

1、何为嵌入式系统

随着嵌入式技术的发展和应用，出现了不同的嵌入式系统的定义，目前国内一个普遍被认同的定义是：以应用为中心，以计算机技术为基础，软硬件可裁减，适应应用系统对功能、可靠性、成本、体积、功耗严格要求的计算机系统。

可以从以下几个方面来理解国内对嵌入式系统的定义：

◆ 嵌入式系统式面向用户、面向产品、面向应用的，它必须与具体的应用相结合才会具有生命力、才具有优势。即嵌入式系统是与应用紧密结合的，它具有很强的专用性，必须结合实际系统需求进行合理的裁减利用。

◆ 嵌入是系统式将先进的计算机技术、半导体技术和电子技术以及各个行业的具体应用相结合后的产物。这一点就决定了它必然是一个技术密集、资金密集、高度分散、不断创新的知识集成系统。所以，介入嵌入式系统的行业，必须有一个正确的定位。例如 Palm OS 之所以在 PDA 领域占有 70%以上的市场，就是因为其立足于个人电子消费品，着重发展图形界面和多任务管理；而风河（WindRiver）的 VxWorks 之所以在火星车上得以应用，则是因为其高实时性和高可靠性。

◆ 嵌入式系统必须根据应用需求可对软硬件进行裁减，满足应用系统的功能、可靠性、成本、体积等要求。所以，如果能建立相对通用的软硬件基础，然后在其上开发出适应各种需要的系统，是一个比较好的开发模式。目前，国内外有很多微内核大小不等的嵌入式操作系统，可以根据实际的使用进行功能扩展或者裁减，从而加速嵌入式系统的开发。

现在当讲到嵌入式系统时，一般是指近年来比较热的具有操作系统的嵌入式系统。其基本结构如图2.8所示。

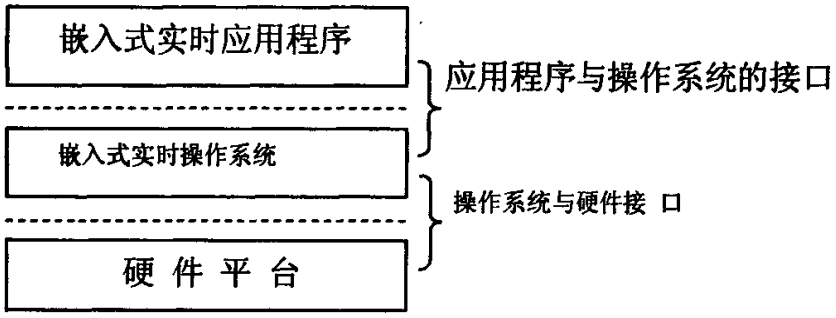


图 2.8　嵌入式系统的基本结构

2、嵌入式实时操作系统

实时操作系统（Real Time Operating System，RTOS）是根据操作系统的工作特性而言的，是指具有实时性，能支持实时控制系统工作的操作系统。它的首要任务是调度一切可利用的资源完成实时控制任务，提高可靠性，其次才着眼于提高系统的使用效率，要满足对时间的限制和要求。RTOS是嵌入式应用软件的基础和开发平台，它应具有如下的功能：

1)．任务管理（多任务和基于优先级的任务调度）；

2)．任务间的同步和通信（信号量、邮箱和消息队列）；

3)．存储器优化管理（含ROM 的管理）

4)．实时时钟服务

5)．中断管理服务

实时操作系统中的任务（Task）等同于分实操作系统中的进程（Process）的概念。系统中的任务有四种状态：运行（Executing）、就绪（Ready）、挂起（Suspended）、睡眠（Dormant）。

运行：获得CPU控制权；

就绪：进入任务等待队列，通过调度转为运行状态；

挂起：任务发生阻塞，从任务等待队列中移出，等待系统实时事件的发生而唤醒，从而转为就绪或运行状态；

睡眠：任务完成或者错误等原因被删除的任务。

在任意时刻，只有一个任务处于运行状态。

RTOS是操作系统研究的一个重要分支，它与一般商用多任务OS如Unix、Windows

等有共同的一面，也有不同的一面。对商用多任务OS，其目的是方便用户管理计算机资源，追求系统资源最大利用率；而RTOS追求的是调度的实时性、时间响应时间的可确定性、系统的高度可靠性。评价一个实时操作系统一般可以从任务调度、内存管理、任务通讯、内存开销、任务切换时间、最大中断禁止时间等几个方面来衡量。

3、嵌入式系统的组成

一个嵌入式系统是一个有特定功能或用途的计算机软硬件的集合体，其硬件的核心部件是嵌入式处理器，包括微控制器（MCU）、数字信号处理器（DSP）、嵌入式微处理器（MPU）、嵌入式片上系统（System On Chip）等。而软件的核心部件式嵌入式操作系统，目前流行的嵌入式操作系统有VxWorks、pSOS、QNX、Windows CE、Palm OS、QNX、Linux等。

（1）嵌入式微处理器

目前据不完全统计，全世界嵌入式处理器的品种总量已经超过1000多种，流行的体系结构有30多个系列。现在几乎每个半导体制造商都生产嵌入式处理器，根据其现状，大致分为以下几类：

◆ 嵌入式微处理器（Embedded Microprocessor Unit, EMPU）

嵌入式处理器目前主要有Aml86/88、386EX、SC-400、Power PC、68000、MIPS、ARM系列等。

◆ 嵌入式微控制器（Microcontroller Unit,MCU）

嵌入式微控制器目前的品种和数量最多，比较有代表性的通用系列有8051、P51XA、MCS-251/96、MC68HC05/11/16、68300等，

◆ 嵌入式 DSP 处理器（Embedded Digital Signal Processor, EDSP）

目前嵌入式DSP处理器比较有代表性的产品是Texas Instruments的TMS320系列和Motorola 的DSP56000系列。

◆ 嵌入式片上系统（System On Chip, SOC）

SOC分为通用和专用两类，通用系列包括Siemens的TriCore，Motorola的M-Core，某些ARM系列器件。专用SOC一般专用于某个或某类系统中，一个有代表性的产品是Philips的Smart XA，它将XA单片机内核和支持超过2048位复杂RSA算法的CCU单元制作在一块硅片上，形成一个可加载JAVA或者C语言的专用SOC，可用于公众互联网如

Internet安全方面。

（2）嵌入式操作系统

经过多年的发展，目前世界上已经有一大批十分成熟的实时嵌入式操作系统，比较流行的嵌入式操作系统如下：

- VxWorks
- pSOS
- Windows Embedded

Windows Embedded 产品家族是Microsoft的产品，主要是用于建立支持具有丰富应用程序和服务的32嵌入时系统，从而针对广泛的用户需求提供灵活解决方案。此外，同支持更快的“产品上市速度”并降低开发成本，Windows Embedded 产品家族还能保证开发人员立于竞争前沿。目前Windows Embedded产品家族主要有Windows CE 3.0和Windows NT Embedded 4.0。

- Palm OS
- OS-9
- LynxOS
- QNX
- 嵌入式 Linux

自由免费软件Linux的出现对目前商用嵌入式操作系统带来了冲击。它可以移植到多个有不同结构的CPU和硬件平台上，具有很好的稳定性、各种性能的升级能力强，而且开发更容易。

国际上许多大型跨国企业，已经选中了Linux操作系统作为开发嵌入式产品的工具。如韩国三星公司、美国Transmeta公司等。国内也有很多厂家推出了基于Linux的嵌入式系统，如中科红旗软件技术有限公司既开发了嵌入式Linux系统基本开发平台，有提供了可供裁减的嵌入式Linux图形用户界面、窗口系统和网络浏览器，并与其它厂商合作开发了许多产品，包括PDA、机顶盒、彩票机等。

- μC/OS-II

μC/OS-II是源代码公开的实时嵌入式内核，是由美国人Jean J. Labrosse撰写，其性能完全可以与商业产品竞争。它是基于μC/OS的，在1992年以来已经有很多成功的商业应用。它可在绝大多数8位、16位、32位甚至64位微处理器、微控制器、数字信号处理器（DSP）上运行。鉴于它的很多的优点，我在系统设计中选用了此操作系统，关于它的更详细的信息将在第下一章中论述。

另外，国内也有许多自主开发的实时操作系统，如科银京成（CoreTek）公司的嵌入式软件开发平台DeltaSystem,中科院推出的Hopen嵌入式操作系统，浙江大学自主研制开发的全中文的嵌入式操作系统HBOS系统等。

面对如此众多的嵌入式操作系统，嵌入式开发人员要根据自己的实际应用，进行合理的选择。进行选择时，一般主要考虑以下几个方面：

◆ 操作系统的硬件支持

这主要从两个方面考虑：是否支持目标硬件平台；可移植性；

◆ 开发工具的支持程度

选择实时操作系统时，要考虑与之相关的工具。微处理器、在线仿真器（ICE）、编译器、汇编器、连接器、调试器以及模拟器等都不同程度影响着操作系统。

◆ 能否满足应用需求

主要考虑以下几个方面：对操作系统性能的要求：内核存储空间要求、网络化支持等；中文内核支持，国内产品要考虑对中文的支持；标准兼容性，要考虑应用行业的标准性；技术支持，购买RTOS之后，还需要技术支持，要考虑供应商的技术支持渠道及时间性等；源代码还是目标码；许可，获得RTOS使用许可进行开发产品时，要考虑供应商的收费方式。

如果考虑了以上的各种因素之后，找不到合适的实时操作系统，可以自建一个。自建实时操作系统有两种方式，一种是完全从内核开始，写自己的RTOS，这对于一般的用户和开发人员而言，是不可想象的。另一种就是在免费的源代码公开的内核上写自己的RTOS，如Linux和μC/OS-II。

2.4.2 车载厢体风机嵌入式控制系统

早期风机控制系统，在硬件上采用单片机和多路开关控制，软件上使用常规的顺序程序方法加上中断方式或者端口查询方式来实现，这种系统实现复杂的微机控制系统是比较困难的，主要体现在以下几个方面：

(1).实时性差：由于计算机在处理中断时，一般不允许响应低级和同级中断，为了提高实时性，要求中断处理程序尽量短。但是有许多实时操作的处理比较复杂，需要较长的CPU执行时间。如果用中断来完成这些处理，则在处理时，无法响应低级或

同级中断。如果采用中断置标志的方法，让主程序来进行处理，则一方面会增加程序的复杂性，另一方面也难以做到实时处理，因为主程序不可能在执行其它程序时，随时去检查这些标志位而转向不同的处理程序。

(2).难以实现并行操作的相互通信：在功能较强的实时系统中，除了主程序有时需要与中断间进行信息交换外，各个并行操作之间有时也需相互通信。这些用常规方法是难以实现的。

(3).结构复杂、移植性差、维护困难：单片微机功能的复杂化，使软件越来越复杂，特别是为了实现并行操作，需使用大量的中断和标志，使程序结构十分混乱，难以设计和调试。同时由于程序采用线性结构，使得程序难于修改或者移植，因此缺乏灵活性、通用性和可维护性。

为了解决以上的问题，可以把应用软件按所完成的功能分成一个个独立的、但可以并行运行的任务，如串行口通信任务、数据采集任务、数据计算任务、定时打印任务等。这样，整个应用软件有各个任务所组成，设计、调试时可分别进行。修改时只可修改个别任务即可，从而提高了软件的可移植性。为了提高系统的可靠性，并有效地实现任务间的相互通信，当应用程序处理的任务较多，尤其要求同时执行两个以上的工作和任务时，在软件设计中引入实时多任务操作系统(Real Time Operating System,RTOS)将非常必要。

提倡在嵌入式应用中使用 RTOS 的最主要原因是提高系统的可靠性。长期以来，在国内传统的开发方式是：针对某一应用，画程序流程图、编制应用程序。通常是线性程序，此机制的优势在于流程直观。这种方法的缺点是：除中断服务程序以外，各程序模块没有优先级的区别，被主循环简单地轮转调用，实时性差，响应时间无法预料；而且，当一个任务申请不到资源，或循环过程中由于某种原因无法跳出循环时，其他任务将得不到响应， 当程序很小时，虽然可通过设置 Watchdog ，利用中断等方法来解决上述矛盾。如果程序变得较大，将大大增加开发时间和调试难度，复杂度不堪想象。正是上述的缺点，在干扰严重的情况下，系统安全性差。另一重要原因是提高开发效率，缩短开发周期。

系统中引入 RTOS 之后，有 RTOS 完成任务管理、任务间通信、中断管理等功能。嵌入式系统中的多任务操作系统在应用系统启动后，首先运行的是背景程序，用户的应用程序是运行于其上的各个具体任务，多任务操作系统允许灵活地分配系统资源

（中央处理器、存储器等等）给各个任务，各程序模块（或者任务）就如同中断程序一样并行运行，这样就可以简化那些复杂而且时间要求严格的工程的软件设计，同时也提高了可靠性。

目前较流行的嵌入式实时多任务操作系统国外主要有 VxWorks、QNX、pSOS、Windows CE 等。另外，国内也有许多自主开发的实时操作系统，如科银京成(CoreTek)公司的嵌入式软件开发平台 DeltaSystem，中科院推出的 Hopen 嵌入式操作系统，浙江大学自主研制开发的全中文的嵌入式操作系统 HBOS 系统等。这些操作系统性能优越，易于移植，但均属于商业操作系统，需支付昂贵的版税。另外也有两个优秀的自由软件操作系统是μC/OS-II 和嵌入式 Linux，它们也具有相当好的性能，且源代码开放，免费使用，以上这些操作系统大多都有完善的开发环境和工具。用户在进行嵌入式系统的设计时，根据具体应用和实际情况，选择适合自己的实时操作系统。

基于嵌入式的车载厢体风机控制系统具有自己的优越性，但是它与基于多路开关的控制系统一样，并没有通过软件去预测各个温度检测点温度变化趋势，风机的起停状态完全取决于静态的控制参数，这样就直接加速了风机老化，降低风机寿命。同时也没有考虑到温度变化历史数据的保存问题，后期无法对历史数据进行分析。

所以本文提出的嵌入式车载厢体风机控制系统，就是以风机为主要研究对象。通过 USB 存储器定期获得温度变化历史数据，采用前馈神经网络对大量历史数据进行知识学习，去预测各个检测点温度的变化趋势，并以此为依据，实现了模糊规则的在线修改和隶属函数的自动更新，从而使风机控制系统工作效率更高，有效保护风机，延长风机寿命。另外为了能将温度变化历史数据传输至中央服务器，本文提出了基于“伪固定”IP 地址的远程访问，通过第三者作为跳板，实现电信动态分配的 IP 地址成为“固定”IP 地址。以上这些都使得本系统具有独特的优越性。

第三章 系统硬件设计与实现

本章将介绍系统的总体设计方案，以及各单元电路的结构、原理以及实现。

3.1 总体设计

3.1.1 系统工作的基本原理

本系统实现的是车载厢体风机控制。即要实现厢体的恒温，又要能够有效的保护风机，提高风机的工作效率。工作流程比较简单，如图 3.1 所示。

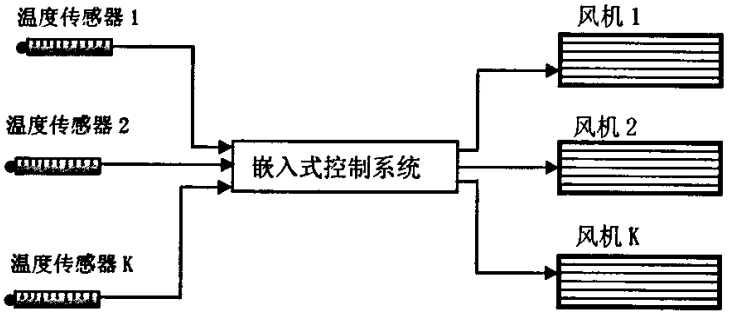


图 3.1 风机控制系统工作流程图

在图 3.1 中，嵌入式控制系统获取各个检测点温度值，由神经网络 BP 算法分析出温度变化趋势，模糊控制系统根据既定参数和规则决定各相应风机的工作状态。

根据图 3.1 的工作流程，要求嵌入式控制系统具备以下几项功能：

（1） 在线连续检测各个点温度值，有风机对温度进行调节；

（2） 具有多路开关输出，用于控制风机的起停；

（3） 具有键盘和液晶显示器，用于在控制器上设定参数及现实信息；

（4） 能通过 USB 接口定期保存温度变化历史数据；

（5） 能通过无线接口接受便携机或者 PC 机命令

（6） 有一定的抗干扰能力和自恢复能力，能适应运输过程中的不良环境。

3.1.2 系统硬件设计的主要目标

根据以上工作过程，可以制定出系统硬件设计的主要目标：

（1）为了让实时多任务操作系统μC/OS-II 在 MCU 上顺利运行，须选用符合要求的 MCU，并配备足够大的 RAM 和 ROM 空间；

（2）设计多通道的 D/A 转换电路，将 MCU 输出的控制量转换为 0~20mA 电流，通过变频器来控制风机的转速，为了提高安全性，还要用光电耦合器将弱电和强电完全隔离开；

（3）具有键盘和显示器，键盘主要用于参数设定，显示器主要用于多点温度和风机工作状态的动态显示；

（4）具有开关量的输入和输出功能，用于控制各风机的工作状态，或检测外设的工作状态；

（5）配备 USB 接口，定期保存温度变化历史数据；

（6）配备无线通信接口，能在工作现场与便携机进行数据通信。

3.1.3 系统硬件设计方案

系统的硬件结构如图 3.2 所示。

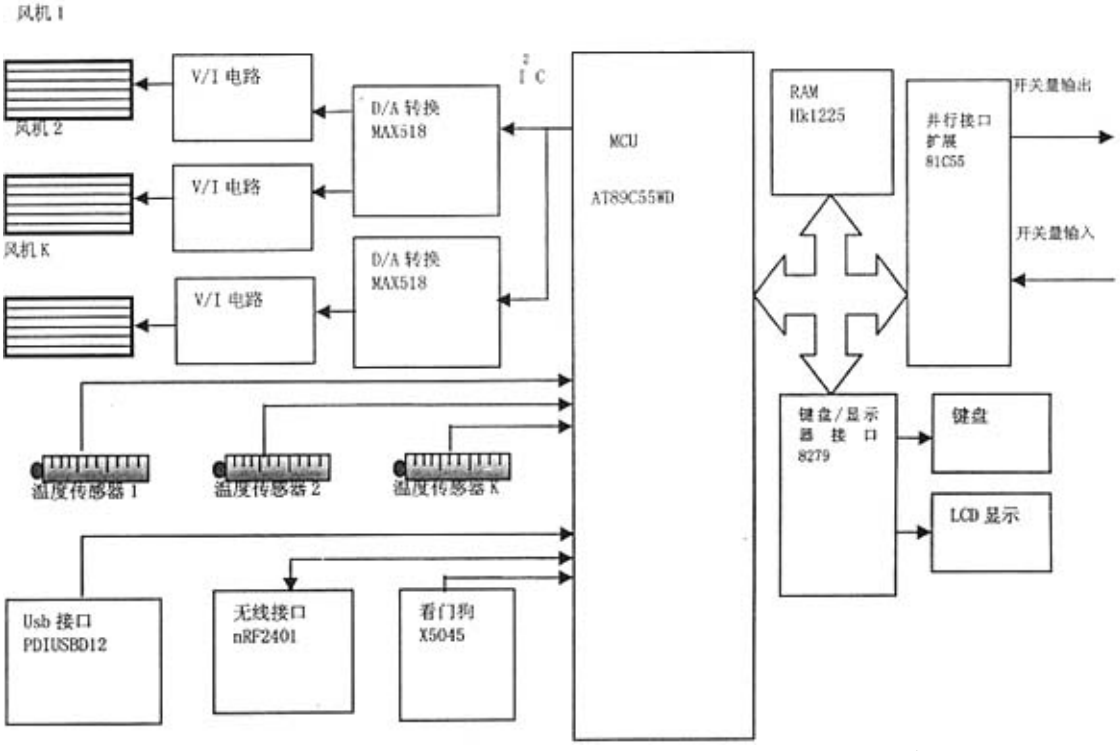


图 3.2 风机控制系统硬件结构图

系统的体系结构设计方案如下：

（1）MCU 采用 8 位 CISC 架构的微处理器 AT89C55WD，它具有 111 条功能指令，内置 20K 的可重写快闪存储器，1 个硬件看门狗定时器，256 字节内部 RAM，32 根 I/O 口线，3 个定时/计数器，8 个中断源和 1 个串口。另外，在 AT89C55WD 并行总线上扩展了内置锂电池的随机存储器 HK1225，可以存储参数，并行接口扩展芯片 81C55 以及键盘/显示接口 8279；

（2）温度采用串行数字温度传感器 DS18B20 检测，直接输出数字信号到 MCU；

（3）D/A 转换采用 8 位串行数模转换器 MAX518，AT89C55WD 通过 I2C 总线输出的控制量先经过 MAX518 转化成 0~5V 支流信号，再由 V/I 电路转成 0~20mA 电流信号，然后控制变频调速器调整风机的工作速度；

（4）配备了多功能监控芯片 X5045，该器件不仅具有看门狗和电源电压监控功能，而且内含了 4kb EEPROM；

（5）无线收发芯片 nRF2401；

（6）USB 接口芯片 PDIUSBD12

系统运行时，模拟温度信号经过温度传感器 DS18B20 转换成数字信号，进入 MCU，调用神经网络模糊控制算法求出输出量，经过 D/A 转换后驱动变频调速器改变风机的工作速度，从而实现温控的效果。同时定期将温度变化数据传输至 USB 存储器保存。

3.2 各单元电路的结构与原理

3.2.1 AT89C55WD MCU 概述

AT89C55WD 是一个低电压，高性能 CMOS 8 位单片机，片内含 20k bytes 的可反复擦写的 Flash 只读程序存储器和 256 bytes 的随机存取数据存储器（RAM），器件采用 ATMEL 公司的高密度、非易失性存储技术生产，兼容标准 MCS-51 指令系统，引脚兼容工业标准 89C51 和 89C52 芯片，采用通用编程方式，片内置通用 8 位中央处理器和 Flash 存储单元，内置功能强大的微处理器的 AT89C52 可为您提供许多高性价比的解决方案，适用于多数嵌入式应用系统[39]。

如图 3.3 所示，AT89C55WD 有 40 个引脚，32 个外部双向输入/输出（I/O）端口，同时内含 2 个外中断口，2 个 16 位可编程定时计数器，2 个全双工串行通信口，2 个

读写口线，片内时钟电路，AT89C55WD 采用两种软件控制其进入省电睡眠模式的静态逻辑工作闲置方式设计，可以用 RAM、定时/计数器、串行口和外部中断唤醒睡眠状态而继续工作，在睡眠模式下，RAM 被冻结，其他功能全部停止，直至下个外中断触发或硬件复位方可开始运行。特别是可反复擦写的 Flash 存储器可有效地降低开发成本。

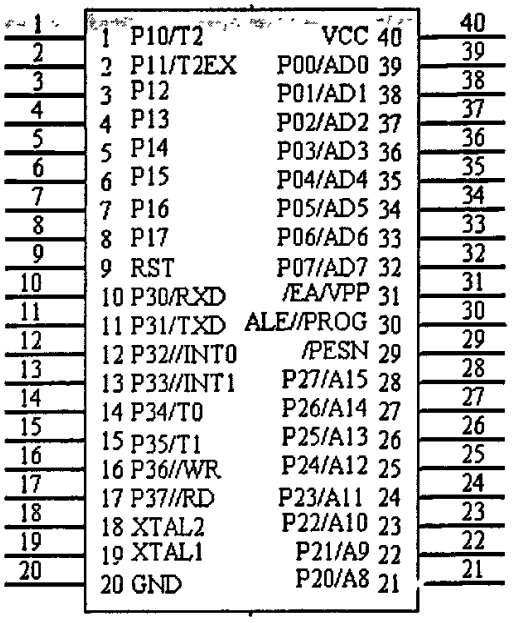


图 3.3 AT89C55WD 引脚图

3.2.2 数字温度传感器 DS18B20

DS18b20 是美国 DALLAS 半导体公司继 DS1820 之后最新推出的一种改进型智能温度传感器。与传统的热敏电阻相比，它能够直接读出被测温度并且可根据实际要求通过简单的编程实现 9-12 位的数字值读数方式。可以分别在 93.75ms 和 750ms 内完成 9 位和 12 位的数字量，而且从 DS18B20 读出的信息或写入 DS18B20 的信息仅需要一根口线（单线接口）读写,温度变换功率来源于数据总线，总线本身也可以向所挂接的 DS18B20 供电。而无需额外电源。因而使用 DS18B20 可使系统结构更趋简单，可靠性更高。它在测温精度、转换时间、传输距离、分辨率等方面较 DS1820 有了很大的改进，给用户带来了更方便的使用和更令人满意的效果[20]。

1. DS18B20 简介

（1）独特的单线接口方式：DS18B20 与微处理器连接时仅需要一条口线即可实现微处理器与 DS18B20 的双向通讯。

（2）在使用中不需要任何外围元件。

（3）可用数据线供电，电压范围：3.0~5.5V

（4）测温范围-55 摄氏度~+125 摄氏度。固有测温分辨率为 0.5 摄氏度

（5）通过编程可实现 9~12 位的数字读数方式

（6）用户可自设定非易失性的报警上下限值。

（7）支持多点组网功能，多个 DS18B20 可以并联在唯一的三线上，实现多点测温。

（8）负压特性，电源极性接反时，温度计不会因发热而烧毁，但不能正常工作。

2. DS18B20 的内部结构

DS18B20 采用 3 脚 PR—35 封装或 8 脚 SOIC 封装其内部结构框图如图 3.4 所示：

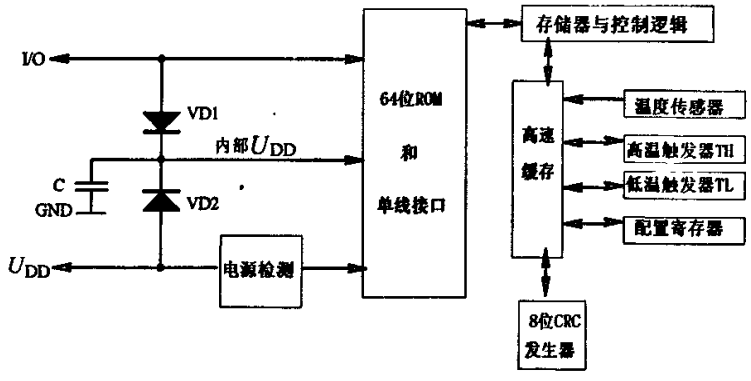


图 3.4　DS18B20 内部结构图

（1） 64bit 闪速 ROM 的结构如下：

开始 8 位是产品类型的编号，接着是每个器件的唯一的序号，共有 48 位，最后 8 位是前 56 位的 CRC 校验码，这也是多个 DS18B20 可以采用一线进行通信的原因

（2） 非易市失性温度报警触发器 TH 和 TL，可通过软件写入用户报警上下限。

（3）高速暂存存储器

DS18B20 温度传感器的内部存储器包括一个高速占存 RAM 和一个非易失性的可电擦除的 EERAM. 后者用于存储 TH、TL 值。数据先写入 RAM，经校验后再传给 EERAM。而配置寄存器为高速占存器中的第 5 个字节，它的内容用于确定温度值的数字转换分辨率，DS18B20 工作时按此寄存器中的分辨率将温度转换为相应精度的数值。该字节各位的定义如下：

TM	R1	R0	1	1	1	1	1

低五位一直都是 1，TM 是测试模式位，用于设置 DS18B20 在工作模式还是在测试模式。在 DS18B20 出厂时该位被设置为 0，用户不要去改动，R1 和 R0 决定温度转换的精度位数，即是来设置分辨率，如下表所示：（DS18B20 出厂时被设置为 12 位）

R1	R0	分辨率	温度最大转换时间
0	0	9 位	93.75mm
0	1	10 位	187.5mm
1	0	11 位	375mm
1	1	12 位	750mm

由表可见，设定的分辨率越高，所需要的温度数据转换时间就越长。因此，在实际应用中要在分辨率和转换时间权衡考虑。

高速暂存存储器除了配置寄存器外，还有其它 8 个字节组成，其分配如下图所示。其中温度信息（第 1、2 字节）、TH 和 TL 值第 3、4 字节、第 6、7、8 字节未用，表现为全逻辑 1；第 9 字节读出的是前面所有 8 个字节的 CRC 码，可用来保证通信正确

温度低位	温度高位	TH	TL	配置	保留	保留	保留	8 位 CRC

LSB　　MSB

当 DS18B20 接收到温度转换命令后，开始启动转换。转换完成后的温度值就以 16 位带符号扩展的二进制补码形式存储在高速暂存存储器的第 1、2 字节。单片机可通过单线接口读到该数据，读取时低位在前，高位在后，数据格式以 0.0625 摄氏度/LSB 形式表示。温度值格式图如下：

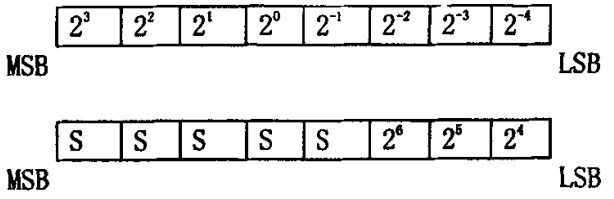


图 3.5　温度值格式图

对应的温度计算：当符号位 S=0 时，直接将二进制位转换为十进制；当 S=1 时，先将补码变换为原码，，再计算十进制值。表 2 是对应的一部分温度值。

表 2 温度转换

温度/°C	二进制表示	十六进制表示
+125	00000111 11010000	07D0H
+25.0625	00000001 10010001	0191H
+0.5	00000000 00001000	0008H
+0	00000000 00000000	0000H
-0.5	11111111 11111000	FFF8H
-25.0625	11111110 01101111	FE6FH
-55	11111100 10010000	FC90H

DS18B20 完成温度转换后，就把测得的温度值与 TH、TL 作比较，若 T>TH 或 T<TL，则将该器件内的告警标志置位，并对主机发出的告警搜索命令作出响应。因此，可用多只 DS18B20 同时测量温度并进行告警搜索，

（4） CRC 的产生

在 64 位 ROM 的最高有效字节中存储有循环冗余校验码（CRC）。主机根据 ROM 的前 56 位来计算 CRC 值，并和存入 DS18B20 中的 CRC 值作比较，以判断主机收到的 ROM 数据是否正确。

3. DS18B20 的测温原理：

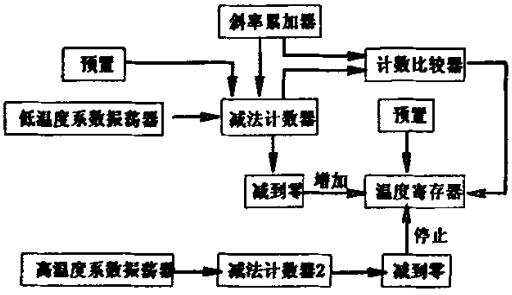


图 3.6　DS18B20 的内部测温电路框图

DS18B20 的测温原理如图 3.6 所示，图中低温度系数晶振的振荡频率受温度的影响很小[1]，用于产生固定频率的脉冲信号送给减法计数器 1，高温度系数晶振随温度变化其震荡频率明显改变，所产生的信号作为减法计数器 2 的脉冲输入，图中还隐含着计数门，当计数门打开时，DS18B20 就对低温度系数振荡器产生的时钟脉冲后进行计数，进而完成温度测量。计数门的开启时间由高温度系数振荡器来决定，每次测量前，首先将-55 摄氏度所对应的基数分别置入减法计数器 1、温度寄存器中，减法计数器 1 和温度寄存器被预置在-55 摄氏度所对应的一个基数值。减法计数器 1 对低温度系数晶振产生的脉冲信号进行减法计数，当减法计数器 1 的预置值减到 0 时 温度寄存器的值将加 1，减法计数器 1 的预置将重新被装入，减法计数器 1 重新开始对低温度系数晶振产生的脉冲信号进行计数，如此循环直到减法计数器 2 计数到 0 时，停止温度寄存器值的累加，此时温度寄存器中的数值即为所测温度。图 2 中的斜率累加器用于补偿和修正测温过程中的非线性，其输出用于修正减法计数器的预置值，只要计数门仍未关闭就重复上述过程，直至温度寄存器值达到被测温度值，这就是 DS18B20 的测温原理。

另外，由于 DS18B20 单线通信功能是分时完成的，它有严格的时隙概念，因此读写时序很重要。系统对 DS18B20 的各种操作必须按协议进行。操作协议为：初始化 DS18B20（发复位脉冲）→发 ROM 功能命令→发存储器操作命令→处理数据。各种操作的时序图与 DS1820 相同。

3.2.3 多通道 D/A 转换

在本系统中，控制风机转速的方法是：由 AT89C55WD 输出控制量，经过 D/A 转换

后向变频器发出控制信号，通过变频调速器来调整风机的转速，从而达到控制风机和温度的目的。系统采用 D/A 转换芯片是 Maxim 公司的 MAX518，该器件是一种串行 8 位 D/A 转换器，单 5V 供电，通过 I2C 总线与微处理器接口，具有 2 路模拟电压输出，输出范围为 0~5V。

MAX518 的引脚 AD0 和 AD1 用于确定该芯片的地址，两片 MAX518 的地址分别为 00 和 01；SCL 是串行时钟线，SDA 是串行数据线，SCL 和 SDA 构成的 I2C 总线上可同时连接多个 MAX518；OUT0 和 OUT1 是两个电压输出通道。AT89C55WD 作为总线主控制器件，产生调节数据流的时钟信号，MAX518 作为从控器件，根据主控器件的命令接收或发送数据。

以三台风机为例。由于要控制三台风机的转速，须三组模拟输出来控制三台变频器，因此需要两片 MAX518。AT89C55WD 发出的控制数据由 81C55 扩展并口的四个引脚输出到 MAX518 的 SCL 和 SDA 端，芯片内部把输入数据中地址位与 AD0、AD1 所表示的地址比较，如符合则把数据存放 8 位数据缓冲寄存器，然后经 D/A 转换和运放后分别从 OUT0、OUT1 输出 0～5V 的电压信号，再经运放 LM358 和三极管 8050 转换成三路 0～20mA 电流信号后，分别控制三台变频器。为了防止强电部分对系统的干扰和冲击，在 81C55 与 MAX518 之间设置了 4N25 进行光电隔离。

3.2.4 看门狗

车载厢体风机控制系统一般工作在运输过程中，各种因素容易对系统造成干扰和冲击，甚至导致系统死机。为了使系统在这些场合可稳定地工作，就必须外加监控电路。本系统采用的复位监控芯片 X5045 是一种集看门狗定时器、电源电压监控、上电复位和带块锁功能串行 EEPROM 四项功能于一体的多功能芯片。该芯片的应用简化了系统结构，减少对电路板的空间需求，降低了成本，增加了系统可靠性。

X5045 使用三线串行总线（SPI）与 AT89C55WD 接口，对芯片进行操作的所有操作码、字节地址及写入的数据都从 SI 引脚输入，写入的数据在串行始终 SCK 的上升沿被锁定；从芯片读取的数据从 SO 引脚串行移出，并在 SCK 的下降沿被读出。芯片的看门狗定时器和电压监视器都对 AT89C55WD 提供独立的保护。当电源电压降到 4.5V 以下时，RST 引脚立即自动产生高电平复位信号，并一直保持高电源电压恢复正常；

当系统上电或掉电时，RST 引脚也自动产生一个高电平复位信号；当系统发生故障时，只要看门狗定时器达到其可编程的超时极限，RST 引脚立即自动产生一个持续 200ms 的高电平复位信号。这样，就可有效地防止死机、数据误写及误操作等故障的发生。

X5045 芯片内部有 2K×8 位的串行 EEPROM，可以擦写 10 万次以上，内部数据可以保存 100 年以上。应用时可以通过编程对指定的块进行锁定，以防止由于错误操作等原因破坏保存的数据。AT89C55WD 必须每隔一段时间向 X5045 发一个触发信号，否则 X5045 将从 RST 发信号使 AT89C55WD 复位，以保证系统不死机。

3.2.5 键盘输入

键盘是一组按键的组合，它是常用的输入设备，可以通过键盘输入数据或者命令，实现简单的人机对话。键盘可分为独立联接式和行列式（矩阵式）两类，每类按其译码方式又分为编码式及非编码式两类。设计中使用的是独立联接非编码式键盘。

每个按键使用的是一个瞬时接触开关，这种联接方式可以容易被微处理器检测，但由于按键会产生机械抖动，在按键被按下或者抬起的瞬间，一般持续 5~15ms，因此设计中要去除键抖动。可以通过硬件双稳态电路或者软件延时来实现，设计中采用延时 20ms 实现的。

对于串键，采用无限处理方法。同时为了防止按一次键而产生多次处理的情况（键扫描和键处理速度较快而此时键还没释放），在有键按下时，作一次键处理后还要检测按下的键是否释放。

3.2.6 LCD 显示

对于LCD MGLS-12864，内置HD61202图形液晶显示模块，厂家为其设置了7条指令来完成对它的控制，有两条指令用于显示状态的设置，其余指令用于数据读/写操作，在此不对其进行详细的说明。

MGLS-12864与微处理器的连接方式有两种：一种是直接访问方式，一种为间接控制方式。直接访问方式就是将液晶显示模块的接口作为存储器或者I/O设备直接挂在计算机总线上，计算机以访问存储器或者I/O设备的方式操作液晶显示模块的工作。而间接控制方式是计算机通过自身的或者系统中的并行口与液晶显示模块连接，通过

对接口的操作达到对液晶显示模块的控制。

设计中我采用了间接控制方式，这种方式的特点是电路简单，控制时序有软件实现，可以实现高速计算机与液晶显示模块的接口。

3.2.7 扩展并行接口

由于本系统要控制的器件和外部设备比较多，光靠AT89C55WD的I/O引脚还不够，故使用一片81C55作为扩展I/O接口，它是一种可编程I/O和RAM扩展芯片，内部含有2个8位、1个6位的I/O口和256B静态RAM以及1个14位定时/计数器。在本系统中，81C55除了用于控制X5045、MAX518和状态指示灯等器件以外，还用来控制各个风机的起停。

81C55的AD0~AD7为地址/数据总线；CE为片选端，输入低电平有效；RD为读操作端，输入低电平有效，在CE有效并且该引脚有效时，按照IO/M端的电平极性，将I/O接口或者RAM单元内容读出到地址/数据总线；WR为写操作端，输入低电平有效，当CE有效并且该引脚有效时，按照IO/M端的电平极性，将地址/数据总线上的数据写入I/O接口或者RAM单元；IO/M为选择端，输入低电平时选中RAM单元，高电平时选中I/O接口；ALE为地址锁存允许信号，从AT89C55WD的ALE端引入，在下降沿时将AD0~AD7的地质、IO/M的状态以及CE的状态锁存至81C55内；TMRIN和TMROUT分别为定时器输入和输出端；PA、PB和PC均为通用I/O接口；RESET为复位端，复位信号从AT89C55WD的RESET端引入。

由于81C55片内有地址锁存器，所有单片机P0口输出的低8位地址不需另外加锁存器，而直接与81C55相连，既作低8位地址总线，又作数据数据总线，地址直接用ALE在81C55中锁存。高8位地址由CE及IO/M的地址控制线决定，因此在图 中的连接状态下，地址编号如表3.1所示

表3.1 81C55的I/O口地址

I/O名称	I/O口地址
命令/状态口	0x8f00
PA口	0x8f01
PB口	0x8f02
PC口	0x8f03
定时器低8位	0x8f04
定时器高8位	0x8f05
RAM单元	0x8e00~0x8eff

3.2.8 扩展数据存储器接口

由于AT89C55WD单片机片内RAM区很小，且掉电后数据会丢失，而本系统需要较大容量的RAM区来保存中间结果和参数，并且对于重要参数要求掉电后不会丢失。为达到此功能，采用8位非易失性存储器HK1225作为数据RAM。该芯片具有以下特点：

（1）容量为8K×8位，双列直插封装，与静态RAM芯片6264完全兼容；

（2）内置锂电池，在无外部供电情况下10年数据不会丢失；

（3）可以单字节读写，读写此处无限；

（4）内部集成抗冲击电路；

（5）单5V供电，功耗小于5mV。

在该系统的连接方式下，HK1225存储器的地址空间为0x4000～0x5fff，单片机AT89C55WD的P0口输出低8位地址信号，经74LS373锁存后送至HK1225的A0～A7端，P2口输出高位地址信号P2.0～P2.4送至HK1225的A8～A12端。向HK1225读写数据时，数据由D0～D7确定，而单元地址由A0～A12确定。

3.2.9 USB接口芯片PDIUSBD12

利用通用串行总线(USB，Universal Serial Bus)可为计算机和外设间的数据通信提供一个很好的解决方案，这种USB具有传输速度快、连接灵活、使用方便等特点。它作为一种高速的新型总线接口，可支持即插即用设备，并能为外设提供电流且易于扩展，因此，可广泛应用于打印机、扫描仪、大容量的外部数据存储器、数码相机和高速数据采集设备等多种设备中。而PDIUSBD12是Philips公司推出的一种价格便宜、功能完善的并行接口芯片，它支持多路复用、非多路复用和DMA并行传输。PDIUSBD12接口芯片遵从协议USB1.1，适合于不同用途的传输类型。PDIUSBD12需要外接微控制器(MCU)来进行协议处理和数据交换，它对MCU没有特殊要求，而且接口方便灵活。

PDIUSBD1.2的突出特点使它成为USB接口开发设计者的首选，它特别适用于便携式USB设备、产品的改型设计、以及需要高速传输数据的数据采集系统。PDIUSBD12与AT89C55WD的数据交换采用中断方式，其中断可通过外部中断0(INT0)来完成。

3.2.10 无线接口芯片 nRF2401

nRF2401 是单片射频收发芯片，工作于 2.4～2.5GHz ISM 频段，芯片内置频率合成器、功率放大器、晶体振荡器和调制器等功能模块，输出功率和通信频道可通过程序进行配置。芯片能耗非常低，以-5dBm 的功率发射时，工作电流只有 10.5mA，接收时工作电流只有 18mA，多种低功率工作模式，节能设计更方便。其 DuoCeiverTM 技术使 nRF2401 可以使用同--天线，同时接收两个不同频道的数据。

1、芯片结构和引脚说明

（1）芯片结构

nRF2401 内置地址解码器、先入先出堆栈区、解调处理器、时钟处理器、GFSK 滤波器、低噪声放大器、频率合成器，功率放大器等功能模块，需要很少的外围元件，因此使用起来非常方便。QFN24 引脚封装， 外形尺寸只有 5×5mm。

（2）引脚说明

表 3.2：nRF2401 引脚

引脚	名称	引脚功能	描述
1	CE	数字输入	使 nRF2401 工作于接收或发送状态
2	DR2	数字输出	频道 2 接收数据准备好
3	CLK2	数字 I/O	频道 2 接收数据时钟输入/输出
4	DOUT2	数字输出	频道 2 接收数据
5	CS	数字输入	配置模式的片选端
6	DR1	数字输出	频道 1 接收数据准备好
7	CLK1	数字 I/O	频道 1 接收数据时钟输入/输出
8	DATA	数字 I/O	频道 1 接收/发送数据端
9	DVDD	电源	电源的正数字输出
10	VSS	电源	电源地
11	XC1	模拟输出	晶振 1
12	XC2	模拟输入	晶振 2
13	VDD_PA	电源输出	给功率放大器提供 1.8V 的电压
14	ANT1	天线	天线接口 1
15	ANT2	天线	天线接口 2
16	VSS_PA	电源	电源地
17	VDD	电源	电源正端
18	VSS	电源	电源地
19	IREF	模拟输入	模数转换的外部参考电压
20	VSS	电源	电源地
21	VDD	电源	电源正端
22	VSS	电源	电源地
23	PWR_UP	数字输入	芯片激活端
24	VDD	电源	电源正端

2、工作模式

RF2401 有工作模式有四种：收发模式、配置模式、空闲模式和关机模式。nRF2401 的工作模式由 PWR_UP 、CE、TX_EN 和 CS 三个引脚决定，详见表 2。

表 3.3：nRF2401 工作模式

工作模式	PWR_UP	CE	CS
收发模式	1	1	0
配置模式	1	0	1
空闲模式	1	0	0
关机模式	0	×	×

3.3 风机控制系统实现

经过精心的选材和详细的设计，实现了风机控制系统的硬件连接。整个系统如图 3.7 所示。

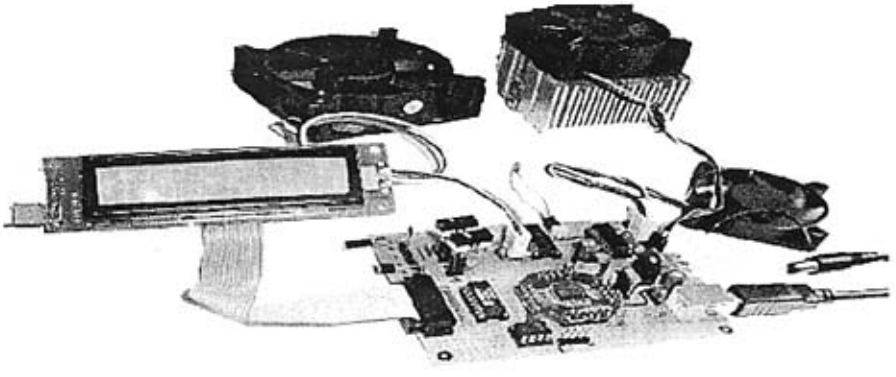


图 3.7 风机控制系统硬件连接图

第四章 系统软件设计与实现

本章将从软件设计的主要目标、软件需求、体系结构、开发平台、设计方案uC/OS-II 移植、驱动程序设计和基本算法等方面，研究车载风机控制系统的软件设计及实现方法。

4.1 软件设计的主要目标

软件设计的最终目标是控制车厢温度能符合恒温需求，具体而言，软件设计要达到以下3个目标：

（1）实现多任务的风机控制机制。风机控制过程中往往同时产生出许多任务，例如，在测量温度值的同时，不仅要根据温度值的变化来动态调整D/A输出值，以控制风机的转速，还要顾及到显示、键盘、看门狗、通信等任务，这些任务关系复杂，相互交错，因此需要设计一种多任务的控制机制来对其进行调度管理，使其避免冲突，有序进行；

（2）保证风机控制的实时性。风机控制具有很强的时变性，处理过程中采集到的厢体温度总是处于不断的变化之中，因此要求系统具有较高的实时性，与温度的变化保持同步，能根据温度的变化和预测及时调整风机的工作状态，将相应时间限制在尽可能短的时间之内；

（3）采用科学的算法来控制风机的起停和转速。风机起停和转速控制是风机控制中的重要环节，如果过早或过晚控制都会增加风机工作量，缩短风机寿命，增加运行成本。为了使得风机控制始终适应于温度的变化，需要设计出一种简洁高效的算法。

4.2 软件需求

根据上述的软件设计目的，嵌入式系统对软件有如下需求：

（1）为了实现风机控制系统的实时性、多任务运行，本系统需要采用多任务实时操作系统uC/OSII对各任务进行管理，该操作系统的优点是执行效率高、占用空间小、实时性优良和可扩展性强，非常适合于中小型嵌入式系统。因此，针对本文的嵌入式应用，由于其规模较小，故采用uC/OS-II是很合适的。

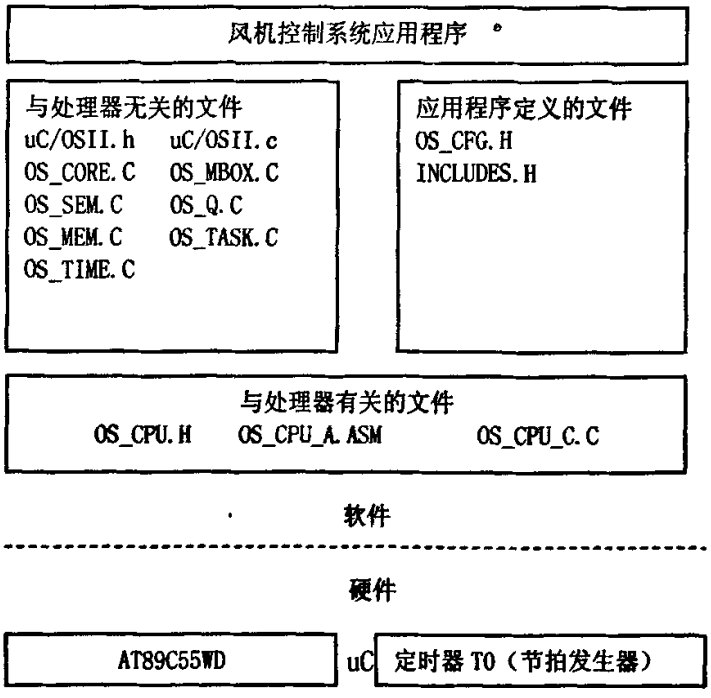
（2）uC/OS-II 作为一个免费的开放式内核，不象一些著名的商业内核那样，具有功能强大的软件包，它缺乏必要的支持，用户通常需要自己编写驱动程序，特别是如果用户使用的是不太常用的处理器，还必须自己编写移植程序。在本系统中，就要针对 AT89C55WD 微处理器编写 uC/OS-II 移植程序，并且自行编写硬件电路的驱动程序；

（3）由于风机控制系统的不稳定性和不确定性，其处理过程具有时变性和随机性，建立精确的数学模型比较困难，因此本系统需要设计出风机起停和风机转速控制的神经网络模糊控制器；

（4）要使风机控制的各环节正确、协调运行，须对风机控制过程进行任务划分，并确定各个任务的优先级别，根据其轻重缓急执行，另外还要妥善解决任务间的通信和共享资源的保护等问题。

4.3 uC/OS-II 的结构及其特点

uC/OS-II 的体系如图 4.1 所示。



uC/OS-II 具有如下特点：

（1）uC/OS-II 是一个占先式的内核，即已经准备就绪的高优先级任务可以剥夺正在运行的低优先级任务的 CPU 使用权。这个特点使得它的实时性比非占先式的内核要好。通常都是在中断子程序中使高优先级任务进入就绪状态（例如使用发送信号的方法），退出中断服务程序后，将进行任务切换，高优先级任务被执行。

（2）uC/OS-II 是一个基于优先级的实时操作系统。它和 Linux 等分时操作系统不同，不支持时间片轮转法。uC/OS-II 中每一个任务必须具有不同的优先级，如果优先级相同，任务将无法区分。进入就绪状态的优先级最高的任务首先得到 CPU 的使用权，只有等它交出 CPU 的使用权后，其他任务才可以被执行。所以只能说它是多任务操作系统，不能说它是多进程操作系统。

（3）uC/OS-II 对共享资源提供了保护机制。如前文所述，uC/OS-II 是一个支持多任务的操作系统。一个完整的程序可以划分成几个任务，不同的任务执行不同的功能。这样，一个任务就相当于模块化设计中的一个子模块。在任务中添加代码时，只要不是共享资源就不许担心相互之间有影响。对于共享资源（譬如串口），uC/OS-II 也提供了很好的解决办法。一般情况下使用的信号量的方法。简单说，先创建一个信号量并对它进行初始化。当一个任务需要使用一个共享资源时，它必须先申请得到这个信号量。而一旦它得到了此信号量，那就只有等它使用完了该资源，信号量才会被释放。在这个过程中即使有优先权更高的任务进入了就绪状态，因为无法得到此信号两，也不能使用该资源。

（4）在单片机系统中潜入 uC/OS-II 将增强该系统的可靠性，并使得调试程序变得简单起来。在传统的单片机开发工作中，调试程序时常会遇到程序跑飞或者陷入死循环的问题。对于前一种情况，可以加看门狗。对于后一种情况，尤其是其中如果牵扯到复杂数学计算的话，只能采用单步执行进行分析了。往往是结果分析出来了，开发人员的耐心也消耗得差不多了。如果在系统中潜入 uC/OS-II 的话，事情就简单多了。可以把整个程序分成许多任务，每个任务相互独立。仁厚在每个任务中设置超时函数，时间用完以后，任务必须交出 CPU 的使用权。即使一个任务发生问题，也不会影响其他任务的执行。这样既提高了系统的可靠性，同时也使得调试程序变得容易，有利于缩短开发周期。

（5）在单片机系统中嵌入 uC/OS-II 将占用系统一定的 RAM 和 ROM 空间。但由于 uC/OS-II 是可以裁剪的操作系统，其所需要的 RAM 和 ROM 依赖于对操作系统的一些

功能的选择，因此开发人员可根据项目的实际需要来对 uC/OS-II 进行功能裁剪，以减少系统开销。

4.4 软件开发平台

本嵌入式系统采用的软件开发平台是 Keil uVision2 集成环境，它是众多单片机应用开发软件中优秀的软件之一，该集成开发环境包含：编译器、汇编器、实时操作系统、调试器等，支持汇编、C 语言编译。uVision2 调试功能强大，具备源极调试、符号调试及历史跟踪、复杂断点等功能。它支持众多不同公司的 MCS51 架构的芯片，它的界面和常用的微软 VC++的界面相似，界面友好，易学易用，深受广大单片机工程师喜欢。该平台能很好地支持本文所用的 AT89C55WD 微处理器，因此采用 Keil uVision2 作为本系统的软件开发平台。软件在该平台上调试、编译成功后，通过 MEP300 编程器写入 AT89C55WD，即可脱机运行。

4.5 软件设计方案

在由 uC/OS-II 管理的多任务机制下，风机控制程序流程如图 4.2 所示。应用程序从函数 main( )开始。main( )内容如下：

```
Void  main(void)  {
        SysInit();              /*系统初始化*/
        OSInit();               /*初始化 uC/OS-II*/
        OSTaskCreate(TaskStart,(void*)0, (void*)&TaskStk[0][0],5);
                                /*建立起始任务*/
        OSStart();              /*开始多任务调度*/
```

其中，SysInit()对系统的初始化工作主要包括：建立相关参数和变量，设置各种中断（不含有 T0 中断），以及对各器件进行初始化（包括键盘/显示芯片 8279、看门狗 X5045、数字温度传感器 DS18B20、扩展并口 81C55、数模转换芯片 MAX518、无线接口芯片、USB 接口芯片等等）；OSIinit（）用于对 uC/OS-II 操作系统初始化。

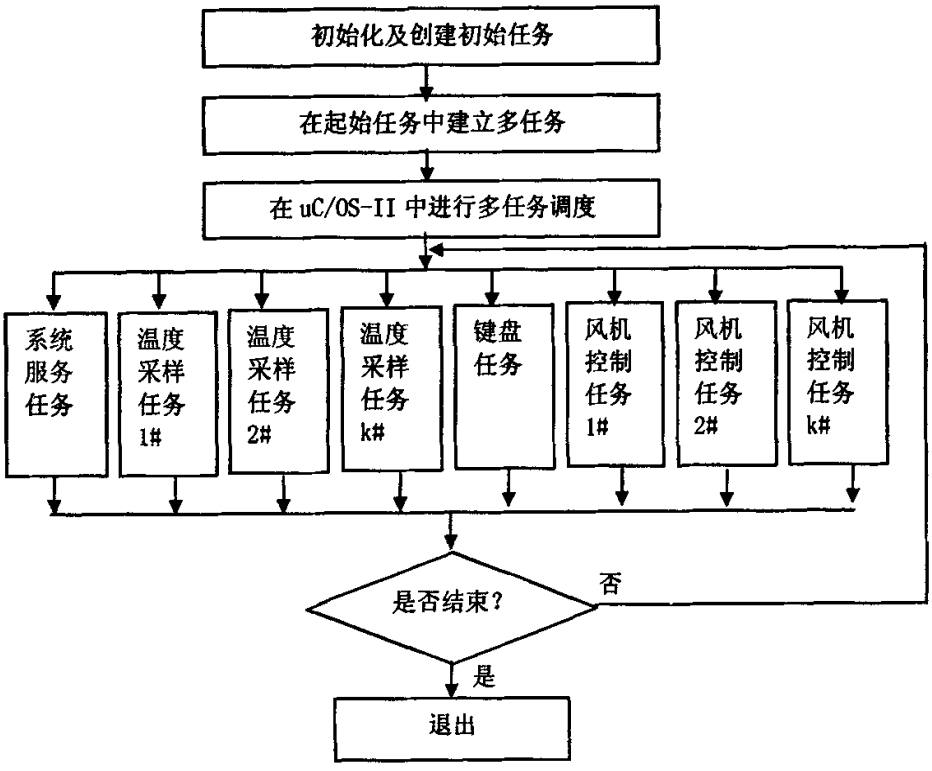


图 4.2 风机控制程序流程图

起始任务 TaskStart（）是一个建立其它任务的任务。首先，起始任务调用函数 InitTimer0（）对定时器 T0 进行初始化，设置时钟节拍率为每秒 50 拍，每拍中断一次，内核在发生中断时判断是否有任务切换或者其它操作。接着，建立以下八个邮箱用于任务间的通信：

```
Mbox_TempOne=OSMboxCreate((void)0);/*用于收发温度 1#消息*/
Mbox_TempTwo=OSMboxCreate((void)0);/*用于收发温度 2#消息*/
Mbox_TempThree=OSMboxCreate((void)0);/*用于收发温度 k#消息*/
Mbox_ air_blower_One =OSMboxCreate((void)0);/*用于收发风机控制 1#消息*/
Mbox_ air_blower_Two =OSMboxCreate((void)0);/*用于收发风机控制 2#消息*/
Mbox_ air_blower_Three =OSMboxCreate((void)0);/*用于收发风机控制 k#消息*/
```

```
Mbox_Key=OSMboxCreate((void)0);/*用于收发键值*/
Mbox_KeyValue=OSMboxCreate((void)0);/*用于收发键值*/
```

接下来，用 OSTaskCreate（）函数建立不同功能的八个任务：TempTaskOne（温度采样任务#1）、TempTaskTwo（温度采样任务#2）、TempTaskThree（温度采样任务#k），Air_BlowerOneTask（风机控制任务#1）、Air_BlowerTwoTask（风机控制任务#2）、Air_BlowerThreeTask（风机控制任务#3）、SysSrvTask（系统服务任务）和 KeyTask（键盘任务）。各任务的优先级依次为 6~13。

（1）温度采集任务 TempTaskOne、TempTaskTwo 和 TempTaskThree。该任务负责以一定的时序读取数字温度传感器 DS18B20 暂存器中的温度值，每收到一次温度值，便调用 OSMboxPost（Mbox_TempX,(void*)&temp）发送温度消息，以供风机控制任务读取。

（2）风机控制任务 TempTaskThree、Air_BlowerTwoTask 和 Air_BlowerThreeTask。该任务通过调用 OSMboxPend()函数，来捕获 TempTaskX 发送的温度消息。当消息收到后 TempTaskX 被唤醒，而后用 OSMboxAccept（）函数从邮箱 Mbox_Temp 中获得温度消息。由于不同的温度对风机控制有影响，故须对进行温度补偿，根据修正后的温度值，调用模糊控制算法，得到相应的控制输出量，经 D/A 转换后产生电流控制信号给变频调速器，以控制风机的工作状态，达到调节厢体温度的目的。

（3）键盘任务 KeyTask。有键按下则产生一次中断（外部中断 0），在中断服务函数中调用 OSMboxPost(Mbox_Key,(void *)&keyvalue)来唤醒键盘任务，键盘任务通过 key_msg=(uint *)OSMboxPend(Mbox_Key,0,&err)来获取键值消息，然后根据不同键值进行起停、清零、参数设置、显示切换等操作。

（4）系统服务任务 SysSrvTask。该任务完成诸如数码显示、点亮指示灯、喂看门狗以及处理系统错误等功能。

另外，本系统与上位机的无线通信是在无线中断服务函数中实现的，如果上位 PC 机或者便携机发来读数据命令，则将各温度值、各风机的工作状态数据发送到上位机；如果上位机发来控制命令（如启动或者停止等），则调用 OXMboxPost(Mbox_Key,(void *)&keyvalue）唤醒键盘任务，模拟手工按键进行启动、停止等操作。

4.6 uC/OS-II 在 AT89C55WD 上的移植

uC/OS-II 的移植对处理器有 5 个要求：

（1）处理器的 C 编译器能产生可重入代码；

（2）能用 C 语言打开或关闭中断；

（3）处理器支持中断，并且能产生定时中断；

（4）处理器支持足够大的 RAM 作为任务堆栈；

（5）处理器有将堆栈指针和其他 CPU 寄存器读出或存储到堆栈或内存中的指令。

本系统采用的 Keil C51 编译器符合条件（1），并且微处理器 AT89C55WD 在扩展了 8K RAM 后，符合条件（2）~（5），因此可以将 uC/OS-II 移植到 AT89C55WD 上。

移植和配置方法如下：

（1）在 OS_CPU。H 中定义相关的宏，声明 C51 能够识别的数据类型和堆栈增长方向。

（2）在 OS_CPU_C.C 中定义以下 6 个函数：OSTaskStkInit( )、OSTaskCreateHook( )、OSTskSwHook( )、OSTaskDelHook( )、OSTaskStatHook( )、OSTimeTickHook( )。实际上真正需要定义的只有 OSTaskStkInit( )，其余 5 个只需声明，不一定要有实际内容，这 5 个函数都是要由用户定义的接口函数。如果要使用这些函数，必须将文件 OS_CFG.H 中的#define constant OS-CPU-HOOKS-EN 设为 1。

（3）在 OS_CPU_A.ASM 中修改以下几个汇编函数：OSStartHighRdy、OSCtxSw、OSIntCtxSw、OSTickISR。OSStartHighRdy 的功能是运行优先级最高的就绪任务，该函数由 OSStart（）函数调用，OSStart（）的调用是建立在 OSInit 的调用并且至少建立一个任务的基础上的；OSCtxSw 是一个任务级的任务切换函数，该函数在任务中调用；OSIntCtxSw 是一个中断级的任务切换函数；OSTickISR 为时钟中断服务函数，时钟中断程序负责处理所有与定时相关的工作，如任务的延时、等待等，在时钟中断中将查询处于等待状态的任务，判断是否延时结束，否则将重新进行任务调度。

（4）在主头文件 INCLUDES.H 中增加 OS_CPU.H、OS_CPU_C.C 和 OS_CPU_A.ASM。因为 INCLUDES.H 是主头文件，它将被所有后缀名为.C 的文件所包含。在移植时，需要对该文件进行改写，在文件末尾增加以下文件：

```
#include<OS_CPU.H>
```

#include<OS_CPU_C.C>

#include<OS_CPU_A.ASM>

（5）在配置文件 OS_CFG.H 中，定义最大事件数，最多内存分块数，最多消息队列，最多任务数，最低任务优先级，是否允许信号量使能，是否允许邮箱使能，是否允许消息队列使能，时钟节拍数以及其他的一些配置。通过修改这些设置，可对 uC/OS-II 进行裁剪，使之适应本系统的具体需要。

4.6 驱动程序的设计与实现

4.6.1 数字测温驱动程序

1、数字测温程序步骤

编写 DS18B20 测温程序应按照以下三个步骤：

（1）初始化。所有操作都从初始化开始，相当于双方握手。AT89C55WD 发送复位脉冲（RESET），DS18B20 回应存在（PRESENCE）脉冲告知 AT89C55WD 自己准备就绪。

（2）发送 ROM 命令。DS18B20 有 5 个单字节 ROM 命令。AT89C55WD 检测到存在脉冲后，方可发送 ROM 命令，ROM 命令的功能包括：查询总线上有多少个 DS18B20，查询单个 DS18B20 的序列号，总线上是否有匹配特定序列号的 DS18B20 以及总线上的 DS18B20 器件是否处于报警状态。

（3）发送 DS18B20 功能命令。AT89C55WD 在发送了某种 ROM 命令以后才能发送 DS18B20 功能命令。通过 6 个字节功能命令，AT89C55WD 能够读写中间暂存器和 EEPROM、初始化温度转换以及判断当前电源供电模式。

中间暂存器有 9 个寄存器（9 个字节），其中第一、第二字节分别是温度值的低 8 位和高 8 位，DS18B20 总是从最低字节开始发送，直到 9 个字节全部发送完为止。在本系统中，只需要第一、第二字节的温度数据，故 AT89C55WD 在收到温度数据后，向 DS18B20 发复位脉冲，使其停止发送后续数据，从而节省了时间。

2、数字测温程序相关代码

（1）引脚的定义

```
Sbit DQ_18B20=P1^0; //由 AT89C55WD 的 P1.0 控制三个 DS18B20 的 DQ 引脚
```

（2）延时子程序

```
void  delay (uchar  delaycount)
{
      while (delaycount--)  {
      此处插入 26 条 "__nop__  ()" 指令；//延时约 15us
      }
}
```

（3）初始化子程序

```
uchar  rst18b20  (void)
{
      uchar  presence;
      DQ_18B20=0; //将 18B20 的 DQ 引脚置低电平
      Delay (36);
       DQ_18B20=1; //DQ 置高电平
       Delay (6);
       presence=DQ_18B20; //读 presence 信号
       Delay  (36);
       Return (presence); //若有则返回 1，无则返回 0
}
```

（4）向 DS18B20 写一字节子程序

```
void  WR18B20 (uchar  byte)
{
    uchar  i ;
    for  (i=0; i<=7; i++) {
          DQ_18B20=0; //开始
          Delay (1); //延时
          Byte>>=1; //第 byte 中的第 i 位移位到 CY
          DQ_18B20=CY; //将该位从 DQ 引脚发送至 DS18B20
```

```
            DQ__18B20=1; //结束
                }
}
```

（5）从 DS18B20 读一字节子程序

```
uchar  RD18B20 (void)
{
      uchar  i ;
      uchar  byte=0; //读入的字节将放在 byte 中
  for   (i=0; i<=7; i++) {
            DO_18B20=0; //开始
             _nop_(); //延时
             DQ__18B20=1; //拉高
             Delay  (1); //延时
             byte= (byte>>=1) | DQ__18B20; //从 DQ 引脚读入第 i 位数据
            }
            return (byte);
}
```

（6）启动一次温度转换子程序

```
void  convert__  temp  (void)
  {
    RST18B20();  //初始化
    WR18B20 (0x44); //启动温度转换
}
```

（7）读取温度转换值子程序

```
int  read_temp  (void)
{
    RST18B20 ();  //初始化
    WR18B20 (0xcc); //跳过多传感器识别
    WR18B20 (0xbe); //读 DS18B20 暂存器
```

```
    DPL=RD18B20 ();  //读温度低位字节
    DPH=  RD18B20 ();  //读温度高位字节
    Return  (DPTR);  //返回以上 2 字节温度值
}
```

有了以上子程序后，若要获取一温度采样，可先用 。onvert__ temp () 启动温度转换，再经过足够的转换时间 (大于 750ms)，调用 read temp ()，其返回值即是温度值。

4.6.2 D/A 转换驱动程序

1、D/A 转换初始化顺序

采用 MAX518 进行 D/A 转换时，由于数据输入是串行方式，所以程序编写要比并行输入的 D/A 转换器复杂，每进行一路 D/A 转换，都须向 MAX518 依次写入三个字节，其格式表 5.1 所示。

表 4.1 向 MAX518 顺序写入的三个字节

写入顺序	字节名称	字节格式							
1	地址字节	0	1	0	1	1	AD1	AD0	0
2	命令字节	0	0	0	0	0	0	0	A0
3	数据字节	X	X	X	X	X	X	X	X

地址字节中的 AD0 和 AD1 须和 MAX518 的引脚 AD0 和 AD1 的电平一致(见图 3.9)；命令字节中的 A0 用于对两路 D/A 通道的选择，如果 A0=0；则选中 DAC0 (即 OUT0)，如果 A0=1,则选中 DAC1(即 OUT1)；数据字节就是要被 D/A 转换的控制量，由 AT89C55WD 发出，范围从 0x00~0xff,该数据经 MAX518 转成模拟量后控制净水剂的投放量。

2、D/A 转换相关代码

(1) 定义控制接口

```
#define  PC8155  XBYTE  [0X8f03]//用 81C55 的 PC4 和 PC5 控制 SCL 和 SDA
```

(2) 写一字节至 MAX518 子程序

```
void  1R518 (uchar  x)
{
      Uchar i;
```

```
        Bdata j;
        for  (i=0; i<=7; i++)  {//依次输出该字节的8位
              j=PC8155;
              x<<=1;
              j^4=CY
              PC8155=j;
              J^5=1;
              PC8155=j;
               J^5=0;
               PC8155=j;
        }
        j"4=0;      //停止位
        PC8155=j;
        J^5=1;
        PC8155=j;
        J^5=0;
        PC8155=j;
      }
```

(3)　D/A转换子程序

```
void  da_convert (uchar  address, uchar  command, uchar  data)
{ //三个参数分别为地址字节、命令字节和数据字节
        bdata  k;
        k=PC8155;
        k^4=1;
        PC8155=k;
        K^5=1;
       PC8155=k;
       K^4=0;
       PC8155=k;
```

```
        K^5=0;
        PC8155=k;
        WR518 (address); //发送地址字节
        WR518 (command): //发送命令字节
        WR518 (data);  //发送数据字节
        K^5=1;
        PC8155=k;
        K^4=1;
        PC8155=k;
}
```

4.6.3 看门狗驱动程序

1、看门狗功能、指令及寄存器

为了提高风机控制系统的稳定性和可靠性，本系统配备了看门狗芯片 X5045。X5045 的稳定性体现在以下三点：（1）在正常工作情况下，系统每隔一段时间会对 X5045 发送一个“喂狗”信号，如果 X5045 在限定时间内收到该信号，便不会发送复位信号，确保程序的运行正常，一旦系统因干扰等原因导致程序跑飞或死机时，“喂狗”信号也随之中断，如果 X5045 爱限定时间内没收到“喂狗”信号，便会立即发出复位信号，使系统重新启动，从而解除死锁等异常状态；（2）在电源电压过低的状态下，X5045 也会向系统发出抚慰信号，直到电压恢复正常为止；（3）X5045 内置带写保护功能的 EEPROM 单元，存贮在其中的重要数据不会因掉电等原因而丢失，从而对污水处理控制系统的重要参数进行保护。

X5045 有 7 条指令（如表 1），对芯片所有操作都是通过对指令寄存器写命令来完成的。所有指令、地址、数据均以高位在前的方式串行传送。X5045 的指令及操作如表 4.2 所示。

X5045 内有一个 8 位状态寄存器，在任何时间都可以通过指令访问其中的内容。复位时为 0，其格式如表 4.3 所示。

表 4.2 X5045 的指令集

指令名	指令格式	操作
WREN	00000110	设置写使能锁存器（允许写操作）
WRDI	00000100	复位写使能锁存器（禁止写操作）
RDSR	00000101	读状态寄存器
WRSR	00000001	写状态寄存器（看门狗和块锁）
READ	0000A011	从选定的地址开始读存储器阵列的数据
WRITE	0000A010	从选下的地址开始写入数据至存储器阵列（1 至 16 字节）

表 4.3　X5045 的状态寄存器

7	6	5	4	3	2	1	0
X	X	WD1	WD0	BL1	BL0	WEL	WIP

在表 4.3 中，WIP 为只读位，用于指示芯片是否正忙于内部非易失性的写操作；WEL 为写使能，指示当前写使能锁存器状态；BL1、BL0 用于设置 EEPROM 块保护的地址范围，见表 4.4

表 4.4 块保护的地址范围

BL1	BL0	被保护的地址范围
0	0	无
0	1	0x180~0x1ff
1	0	0x100~0x1ff
1	1	0x000~0x1ff

表 4.3 中的 WD1 和 WD0 用于设置看门狗定时器的超时周期，见表 4.5。

表 4.5　看门狗定时器的超时周期

BL1	BL0	被保护的地址范围
0	0	无
0	1	0x180~0x1ff
1	0	0x100~0x1ff
1	1	0x000~0x1ff

2、看门狗相关代码

（1）变量、引脚、接口的定义

```
#ifndef  nop2 ()
#define  nop2 ()  __nop__  (); _nop_( );_nop_( ) //用于短暂延时
#endif
```

```
#define  WREN      0x06//允许写操作指令
#define  WRDI      0x04//禁止写操作指令
#define  RDSR      0x05//读状态寄存器
#define  WRSR      0x01//写状态寄存器
#define  READ      0x03//读存储器指令
#define  WRITE     0x02 //写存储器指令
sfr      PORT=0x90; //本系统中，X5045 的 4 根 io 脚都接在 P1 端口
sbit__SI=p1^//si 接在 p1.  6
sbit_SCK=p1^//sck 接在 p1.  7
sbit__SO=p1^   //so 接在 p1.  5
sbit_CS=p1^ //cs 接在 p1.  4
```

（2）位写子程序

```
void _w_byte (Data) //将 Data 按位从 SI 引脚写入 X5045
   char  Data;
{
   char  i;
   PORT  &=  (__SCK^0xff);
   for (i=0; i<8; i++)
    {
       nop2  ()  ; nop2  ();
       if (Data&0x80)     PORT|=__SI;
       else  PORT&=(__SI^0xff);
       nop2  ()  ; nop2  ();
        PORT|=  SCK;
        nop2  ()  ; nop2 ();
        Data=Data<<  1;
        nop2 (); nop2  ();
        PORT&=(  SCK^0xff) ;
         nop2( )  ;   nop2  ();
```

```
        }
}
```

(3) 位读子程序

```
char __r__byte (void) //按位从 X5045 的 So 引脚读出一个字节
{
 char i ;
 char result
 result=0;
 for (i=0; i<8; i++)
   {
       nop2( );   nop2   ();
       PORT | =  _SCK;
       result=result<<1;
       nop2( ); nop2  ();
          if   ((PORT&__SO)   ! =  0)
              result} =0x01;
          nop2( );   nop2( );
          PORT  &= (_SCK ^ 0xff);
          nop2( );   nop2( );
 }
 return (result); //返回值即为所读内容
}
```

(4) 写状态寄存器

```
void      write__  status (char  status) //将 status 中的数据写入状态寄存器
{
 PORT&= (_CS^0xff);
 nop2 (); nop2( );
 _w_byte (WRSR); //发出写状态寄存器指令
 _w__byte (status);
```

```
  PORT | =_CS;
  nop2( )  ; nop2_ ();
  return;
}
```

（5）复位看门狗

```
void    rst_wchdog (void)
{
      _CS=0;
      nop2  ()  ; nop2  ();
      __CS=  1;
      nop2  ()  ; nop2  ();
}
```

（6）延时程序

```
void    wait__  free (void) //将在 (9) 中用到
{
  unsigned  int    t;
  t=3000;
  while (--t);
}
```

（7）从给定地址单元读数据

```
uchar  read__  byte (uint  address) //返回值为地址 address 的内容
{
      char     result;
      PORT  &= (_CS^0xff);  //选中该器件
      nop2 ()  ; nop2 ();
      _w_byte ((char) (address>255? (0x08 | READ): READ)  );
      _w_byte ((char) (address&NOW ));
      result=_r__byte ();
      nop2 ()  ;  nop2 ();
```

```
    PORT | =_CS;                    //不选中该器件
    return (result);
}
```

(8) 写一个数据到给定地址单元

```
void    write__byte (address, Data) //将 Data 写到地址为 address 的单元
  unsigned  int  address;
    char  Data;
{
    write__status (WREN); //使 X5045 允许写入
    nop2 (); nop2 ();
    PORT&= (__CS^0xff) ;
nop2 ()  ; nop2 ();
_w_byte ((unsigned  char) (address>255?  (0x08|WRITE): WRITE));
_w_byte ((unsigned  char) (address&M OM );
_w_byte (Data);
nop2( ); nop2( );
PORT | =_CS;
wait__free ();
return;
}
```

4.6.4 键盘输入驱动程序

键盘的相关驱动函数如下：

✧ GetKey

定义：INT8U GetKey ( );

功能：获得被按下的键数，返回值中指示出哪个键被按下。

✧ GetScanKey

定义：INT8U GetScanKey ( )

功能：对各个按键进行扫描，从而确定其状态。相应位指示其是否被按下（1 表示按下，0 表示断开）。

4.6.5 LCD 显示驱动程序

系统设计中选用的 LCD，内部有控制电路，在系统的内存里开辟了一块内存作为液晶屏显示的后台缓冲区 LCDBuffer,用于保存要显示的内容。对于不同的液晶屏显示只需要改动 LCD128.C 和 LCD128.H 中的程序即可。

液晶模块有两种工作模式：图形方式和文本方式。在图形方式下，模块上的缓冲区映射的是液晶屏上显示的图形点阵；在文本方式下，模块上的缓冲区对应的是液晶屏上显示的文本字符，包括英文字符和英文标点符号。在此对汉字显示仅作演示之用。

液晶屏的操作主要包括：初始化、设置工作模式（文本或者图形）、更新显示。接口函数如下：

✧ LCD_Init

定义：void LCD_Init(void);

功能：初始化 LCD，在系统启动时此函数被调用。

✧ LCD_Pritf

定义：void LCD_Printf(char *str,…);

功能：在 LCD 的文本方式下输出字符串。

参数说明：

str:所输出的字符串

✧ LCD_ChangeMode

定义：void LCD_ChangeMode(INT8U mode);

功能：改变 LCD 的工作模式

参数说明：

mode: 设定的 LCD 的显示模式，0 表示文本模式， 1 表示图形模式

✧ LCD_Refresh

定义：void LCD_Refresh (INT8U col,INT8U row ,char *str)

功能：更新 LCD 的显示，把后台缓冲区 LCDBuffer[][]中的内容更新到 LCD 的显

示屏上。

参数说明：

col：显示字符串的起始列数。

row：显示字符串的起始列数。

str：需要显示的内容

4.6.6 USB 驱动程序

为了使驱动软件可移植性强、易维护，采用分层的方法编写 PDIUSBD12 的驱动程序。

USB 驱动程序软件包提供给用户 6 个 API 函数，这 6 个函数都在 USB 应用层中定义，功能描述如表 4.6 所列。

表 4.6 USB 驱动软件包函数功能说明

函数名称	功能描述
Init_D12	设置 D12 与硬件的连接，初始化 D12，复位 D12，初始化相关变量
TaskSetup	控制传输处理任务，以便 USB 能及时完成传输
ReadPort1	从端点 1 读出字节
ReadPort2	从端点 2 读出字节
WritePort1	向端点 1 发送字节
WritePort2	向端点 2 发送字节

部分源代码如下：

```
#defineRW_NUMS1024//任务收发数据字节数
voidTaskRecl(void*pdata)
{#ifOS_CRITICAL_METHOD==3
//为 CPU 状态寄存器分配存储空间
OS_CPU_SRcpu_sr;
#endif
INT8UBuff[RW_NUMS]; //接收及发送缓冲区
```

```
INT8Uack=0x01; //应答主机数值
INT8Uerr; //函数返回值
pdata=pdata; //避免编译器警告
for(; ; ){
OSSemPend(TaskReel_Sere, 0, &-err); //等待 TaskStart 的命令
err=WritePort1(1, &ack, 200); //应答 USB 主机
if(err==USB_NO_ERR){ //应答正确
err=ReadPort2(RW_NUMS, Buff, 200); //接收数据
OSTimeDly(1); //延时一个时钟周期
if(err==USB_NO_ERR){ //接收正确
Buff[0]=OSPrioCur; //标识该任务
err=WritePort2(RW_NUMS, Buff, 200); //发送数据
}
}
}}
```

4.6.6 无线接口驱动程序

控制器 CY7C63231 控制 nRF2401 射频芯片，实现无线数据的接收和发送。以接收无线数据为例。主程序 UsbTaskLoop，它是一个无限循环，仅仅在中断的时候跳出。程序检测 nRF2401 的 DR1 和 DR2 引脚，当 DR1 上的电平为高时，产生中断跳到 Receivech1 子程序，当 DR2 上的电平为高时，产生中断跳到 Receivech2 子程序。USBSend 程序负责从端口 1 向上位机发送数据，WaitforAck 程序等待上位机对端口 1 的应答信号，Receivech1 和 Receivech2 程序分别从 nRF2401 的通道 1 和通道 2 接收数据。最后调用 DATAOUT 把数据传给上位机，然后调用 WaitforAck。Prg2401 程序段负责对 nRF2401 进行操作控制，能够通过设置不同的参数使 nRF2401 工作在三种不同的工作模式。

4.7 神经网络 BP 算法的设计与实现

4.7.1 神经网络 BP 算法流程图

BP 算法因其简单、易行、计算量小、并行性强等优点，是目前神经网络训练采用最多也是最成熟的训练算法之一。

其算法流程如图 4.3 所示

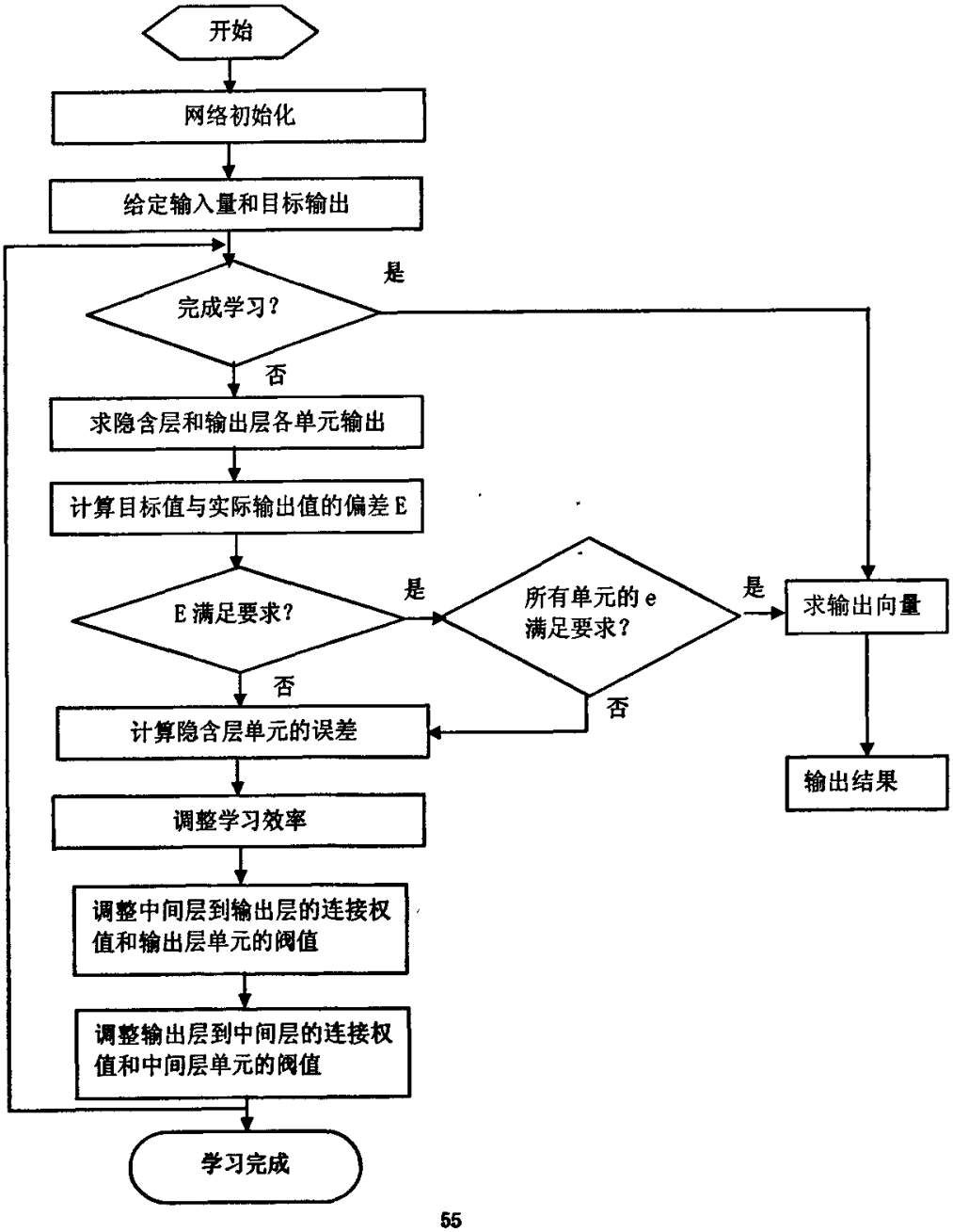


图 4.3 BP 算法流程图

4.7.2 风机控制系统中温度预测的 BP 网络建立

图 4.4 是三个温度传感器在 2007 年 4 月 1 日 19 点至 20 点 50 分之间的温度变化图。从图中可以看出，温度变化序列有一定的相关性，序列之间存在一定的关联性，所以从序列上可以预测未来的走势。姑且认为序列的某一个值的影响因素为序列的前几个值和其他两个温度传感器同一时刻的温度值。

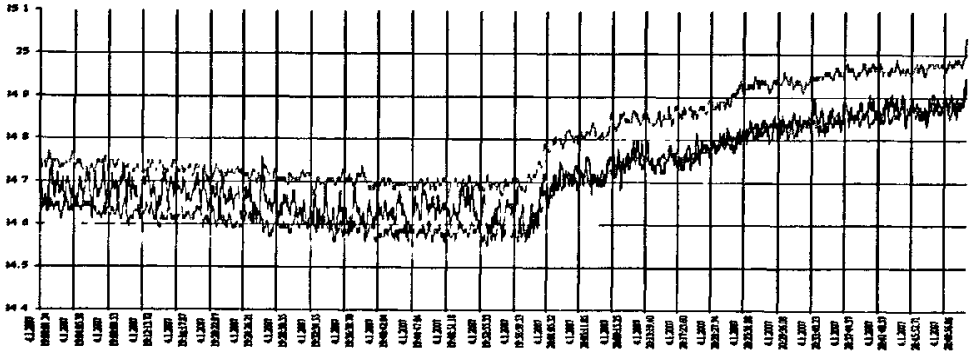


图 4.4 1#、2#和 3#温度检测点温度值折线图

据此，首先我们将建立三层的 BP 网络模型，分别为输入层、隐含层和输出层，如图 4.5 所示。

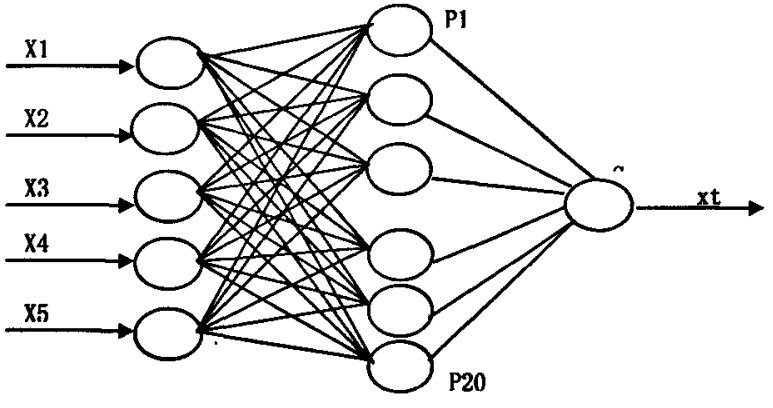


图 4.5 三层 BP 网络

输入层节点个数为 5，x1、x2 和 x3 分别对应该点前三个时刻的温度值，x4、x5 对应其他两个检测点的温度值。输出节点只有一个，反映了下一个时刻的可能温度值。隐含层节点的个数按照公式选择 5*4=20 个。网络共 26 个节点。各单元的输入与输出

的特征函数采用 Sigmoid 函数，即隐含层第 j 个神经元的输入为 $net_j=\sum W_{ji}x_i$，输出 O_j 为：$O_j=1/(1+e^{-(net_j+\theta_j)/\theta_0})$ 公式中 θ_j 表示阈值，θ_0 的作用是调节 Sigmoid 函数的形状。

4.7.3 BP 网络预测结果

以给定的模式作为网络的输入，要求网络通过调节所有的连接权系数和各神经元的阈值，使得在输出层神经元上得到的结果较为理想；取 2007 年 4 月 1 日 19：00 到 20：50 的温度数据为训练样本，计算结束后，网络达到收敛。

对 4 月 6 日的温度进行预测。测试结果如图 4.6 所示。

由此可见，与传统信息处理方法不同，人工神经网络是自适应和可以被训练的，它有自修改能力，从原理上就比传统的方法要快的多。把神经网络 BP 算法引入温度预测，并把预测的结果提供给模糊控制，可以有效的提高风机工作的效率，降低风机的工作强度，增加风机的寿命。

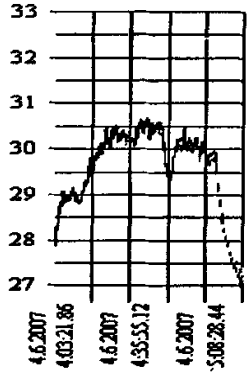


图 4.6 温度预测值与实际值比较

4.8 模糊控制算法的设计与实现

4.8.1 模糊控制模块的结构

模糊控制模块的设计包括精确量的模糊化、模糊控制规则的形成和解决模糊三部分，模糊控制器的结构如图 4.7 所示。

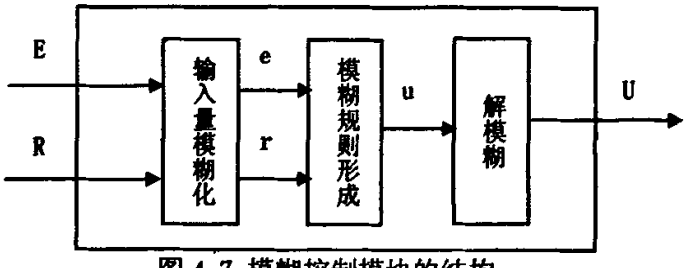


图 4.7 模糊控制模块的结构

控制温度值的模糊控制模块为双输入单输出的控制器，其两个输入变量为：E（温度值的偏差），R（温度值的偏差变化率）；输出量为 U（校正值）。其中 U 是指所测温度值与设定的温度值的差，R 是指 E 的变化速率，R=dE/dt。

4.8.2 输入变量的隶属度函数

输入变量 E 和 R 通过特定的映射规则映射到[-3，3]区间，对应的映射值分别为 En 和 Rn，设定 En 和 Rn 均分属于模糊集合{负大，负中，负小，零，正小，正中，正大}，7 个模糊子集分别记为 NL、NM、NS、ZK、PS、PM、PL。变量 En 和 Rn 属于子集 NL、NM、NS、ZK、PS、PM、PL 的隶属度函数采用三角形。图 4.8 给了 En 对各模糊子集的隶度函数，Rn 对各模糊子集的隶属度函数同此。在区间[-3，3]内，变量 En 和 Rn 有对应的隶属度。例如，En 和 Rn 组合为（-1.8，-2.1）时，En 的隶属度集为{0，0.8，0.2，0，0，0，0}，Rn 的隶属度集为{0.1，0.9，0，0，0，0}，即 En 隶属于负中（NM）和负小（NS）两个子集，Rn 隶属于负大（NL）和负中（NM）两个子集。

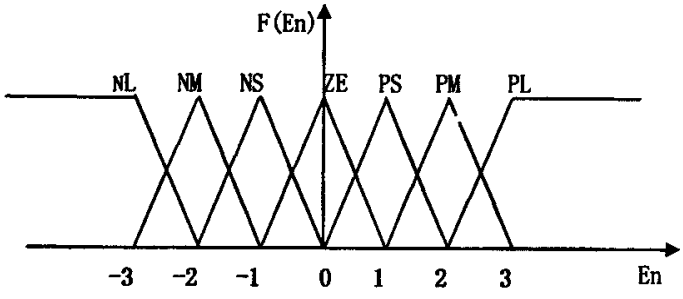


图 4.8 变量 En 对各模糊子集的隶属函数

4.8.3 模糊控制规则

分析风机控制过程并结合实际经验，可得出表 4.1 的控制规则，模糊控制模块系根据 En 和 Rn 对于各子集的隶属度确定所属子集，然后按照表 4.7 判定输出变量 U 所属的模糊子集。

表 4.7 输出变量 U 所属的模糊子集规则

Rn / En	NL	NM	NS	ZE	PS	PM	PL
NL	PL	PL	PL	PL	PM	PS	PS
NM	PL	PL	PM	PM	EZ	ZE	NS

NS	PL	PM	PM	PS	ZE	ZE	NM
ZE	PL	PM	PS	ZE	NS	NM	NL
PS	PM	ZE	ZE	NS	NS	NM	NL
PM	PS	ZE	ZE	NM	NM	NL	NL
PL	ZE	NS	NM	NL	NL	NL	NL

由表 4.7 可知，U=f（En，Rn），其关系又表 4.7 决定。变量 U 所属的模糊集同样被定义为 7 个模糊子集{负大，负中，负小，零，正小，正中，正大}，也记为 NL、NM、NS、ZK、PS、PM、PL；与其对应的量化值 X 区间为[-3，3]；隶属度按照取小原则确定，即：

$$f(X_k)=\min[fe(En), fr(Rn)] \qquad (4.1)$$

公式 4.1 中，$f(X_k)$为 U 属于X_k的隶属度；fe（(En)、fr(Rn)为与X_k对应的 En 和 Rn 隶属度。

仍以上节中 En 和 Rn 组合（-1.8，-2.1）为例，En 隶属度非 0 的子集为 i=2，3，即 En 隶属于负中（NM）和负小（NS）子集；Rn 隶属度非 0 的子集为 j=1，2，即 Rn 隶属于负大（NL）和负中（NM）子集。由表 4.1，U 隶属的子集有 4；k=1，2，3，属于 PL 子集（即 $X_1=X_2=X_3=X_4$）；k=4，属于 PM 子集（$X_4=2$）。有公式（4.1），隶属度

$$f(X_1)=\min(0.8, 0.1)=0.1$$
$$f(X_2)=\min(0.8, 0.9)=0.8$$
$$f(X_3)=\min(0.2, 0.1)=0.1$$
$$f(X_4)=\min(0.2, 0.9)=0.2$$

校正量 U 由公式（4.2）计算：

$$U=\sum_{K=1}^{4} X_k \bullet f(X_k) / \sum_{K=1}^{4} f(X_k) \qquad (4.2)$$

计算结果为 U=2.83。根据校正量 U 的实际量程进行反映射，得出数字量输出，该数字量通过 MAX518 转换成电压，再通过 LM358 等器件转换为 0~20mA 电流，变可控制执行机构调节风机的转速，实现自动控制的目的。控制风机的执行机构如图 4.9 所示。

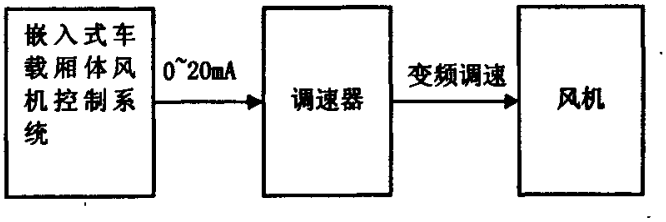


图 4.9　风机风速控制的执行机构框图

以上介绍了根据温度值控制风机转速的神经网络模糊控制方法。结果表明，与神经网络结合能有效的降低风机工作强度，增加风机寿命。

4.9　上位机程序的设计与实现

为了实现对系统的有效控制监控，在 PC 机需要最温度数据进一步分析，以获得风机控制系统的特点，总结经验，更好的调节参数，完成对系统更有效快速的控制。

4.9.1　远程通信的实现

随着计算机网络的进一步普及应用，企业又提出了新的需求，如能将温度历史数据通过网络上传到公司数据中心，做进一步分析；能在家里、办公室里或者外地看到各个控制点的相关参数，一方面是能实时了解各点的现状，另一方面又能对现场上的问题进行远程指导。这就涉及到远程通信。远程通信过程中很多情况下主服务器的 IP 地址是动态的，这是远程通信非常重要的一个问题。

公司数据中心服务器是通过电信局动态分配的 IP 地址进行 Internet 通信，为了能与客户机远程通信，必须将变化后的 IP 地址告诉通信客户端，整个过程如流程图 4.10 所示。

服务端和客户端程序使用的编程工具是 Borland Delphi 6.0，在 Windows 2000 professional 系统上编译、连接、测试成功。

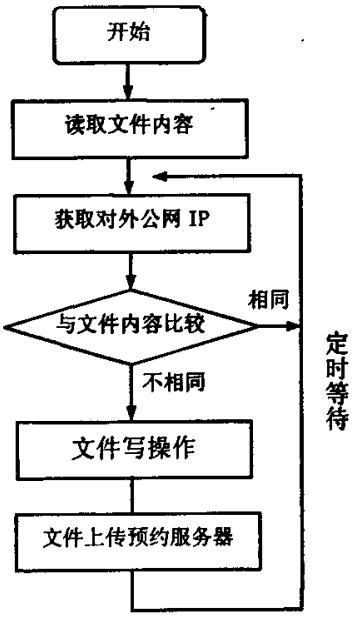


图 4.10 伪固定 IP 地址远程通信流程图

远程通信程序界面如图 4.10 所示：

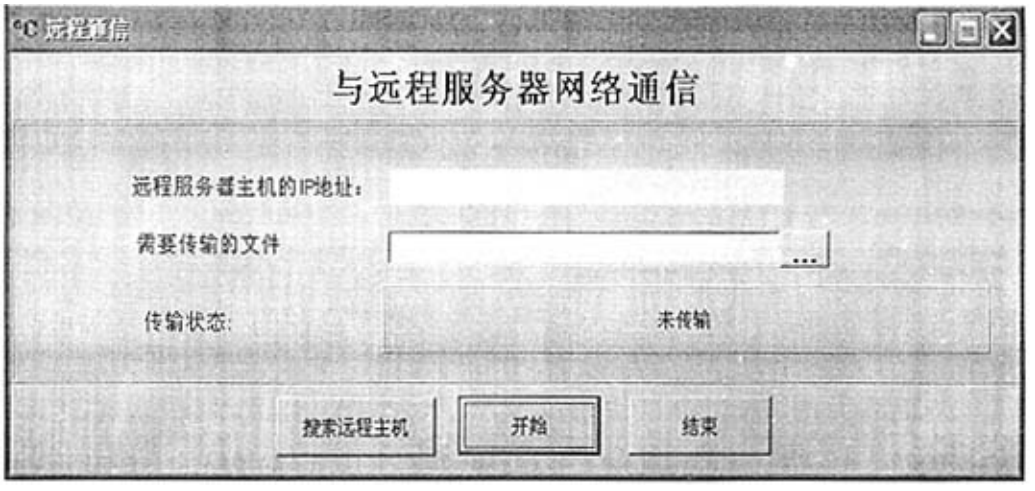


图 4.10 远程通信界面

界面上第一个按钮的功能是搜索远程服务器主机的 IP 地址，并将搜索结果填写在第一个文本框内。第二个按钮负责完成选种文件的上传，并将传输状态和传输结果显示在传输状态标签中。

4.9.2 下位机相关参数在上位机显示的实现

通过无线通信接口，下位机的相关参数如各个风机的实时转速、各个检测点的实时温度以及转速和温度的变化曲线，都可以在上位机进行显示，这样能形象直观的了解各个点的实时控制情况。

界面如图 4.11 所示，

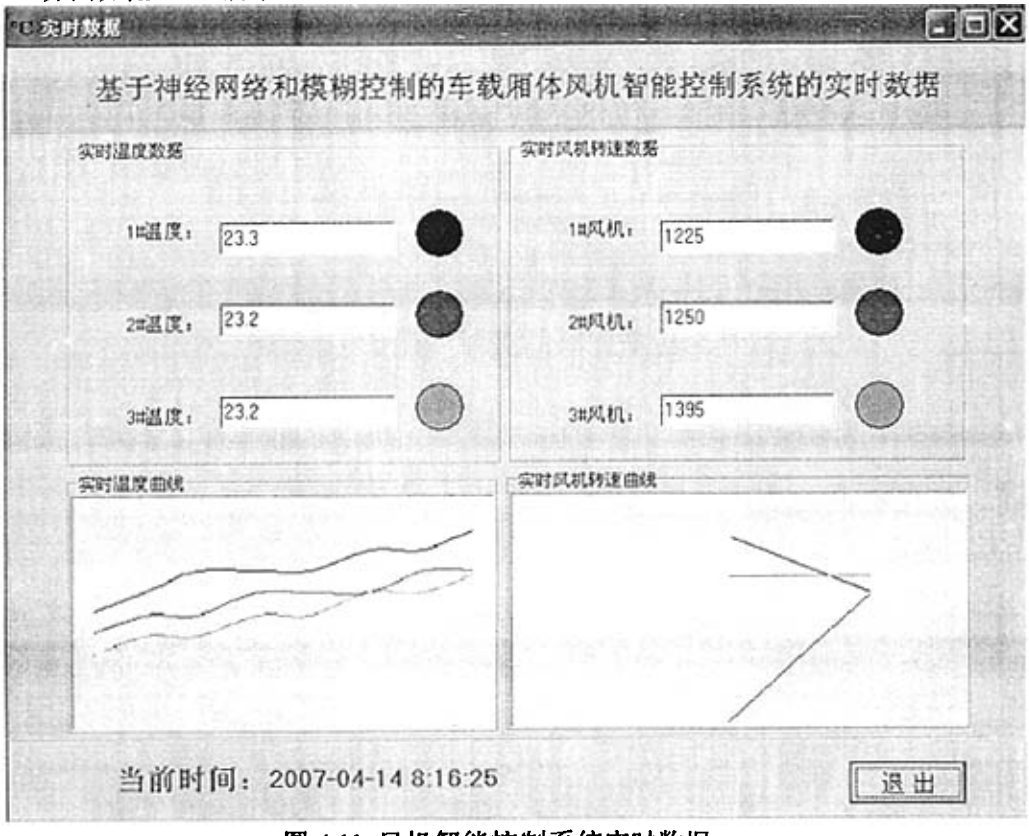


图 4.11 风机智能控制系统实时数据

第五章 系统应用及性能分析

5.1 系统应用

基于神经网络和模糊控制的车载厢体风机智能控制系统是针对大型车厢生产企业研究风机智能控制课题而设计。外设USB存储器通过USB接口与系统通信，可以保存海量的温度和风机转速历史数据，为进一步的分析研究做好了前期准备工作。

本系统的后代产品应用范围广泛，可应用于诸如冷藏、冶金、纺织等使用大型风机的企业，为企业带来较好的经济效益。

5.2 性能分析

本课题在模拟厢体中进行测试，仅仅以三个温度测试点和三个风机控制点的数据来进行相关性能分析。

5.2.1 厢体温度均衡度分析

设定参数模拟厢体恒温在20℃∽23.5℃，临界报警温度值分别为下界20℃，上界23.5℃。以4月14日8：05至8：14的实验数据为例，1#、2#和3#三个检测点的温度测量值如表5.1所示。

表5.1 实际温度测量值

时间 / 温度	8:05	8:06	8:07	8:08	8:09	8:10	8:11	8:12	8:13	8:14
1#	23.05	23.05	23.08	23.10	23.10	23.15	23.20	23.20	23.20	23.25
2#	22.95	22.95	23.00	23.05	23.05	23.10	23.10	23.15	23.15	23.15
3#	22.90	22.90	22.95	23.00	23.00	23.05	23.05	23.10	23.10	23.15

将上表中数据转换为平滑直线图5.1，可以很容易看出其在10分钟以内的变化趋势：三个检测点的温度一直在上升。

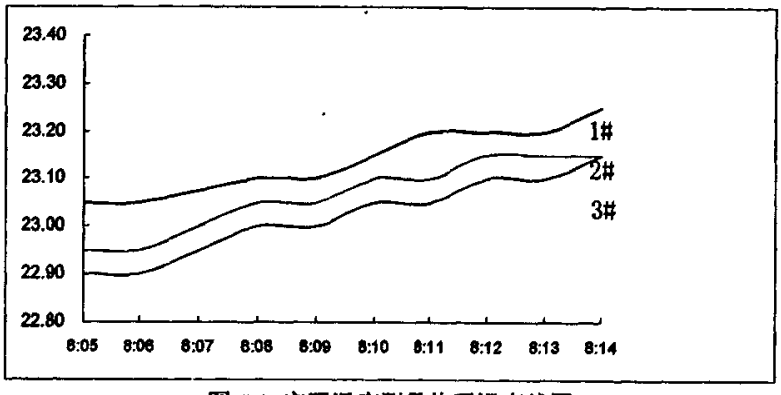


图 5.1 实际温度测量值平滑直线图

通过神经网络 BP 算法，预测三个点未来 10 分钟的可能温度值如表 5.2 所示。

表 5.2 温度预测值

时间 / 温度	8:15	8:16	8:17	8:18	8:19	8:20	8:21	8:22	8:23	8:24
1#	23.25	23.30	23.35	23.35	23.40	23.40	23.45	23.45	23.50	23.55
2#	23.20	23.20	23.25	23.30	23.30	23.35	23.35	23.40	23.45	23.50
3#	23.15	23.20	23.20	23.20	23.20	23.25	23.25	23.30	23.35	23.40

将上表中的预测温度值转换为平滑曲线图，如图 5.2 所示，可以很容易看出在未来 10 分钟内的变化趋势：三个检测点的温度仍然在上升，并且在 8：23 分时 1#点温度可能达到临界值 23.5℃，在 8：24 分时 2#可能达到临界值，3#以后可能会达到临界值。

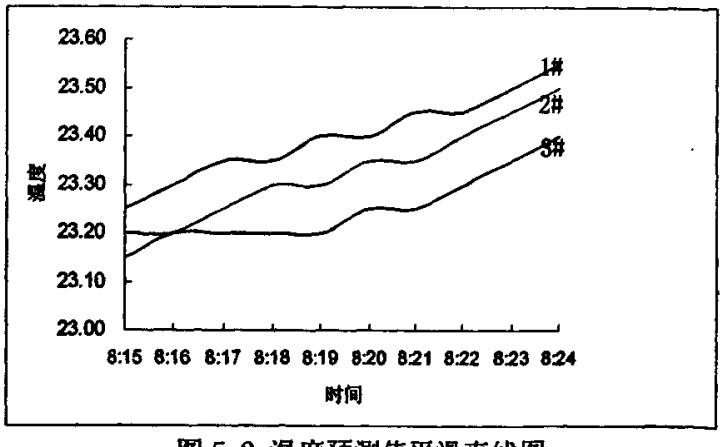


图 5.2 温度预测值平滑直线图

如果在临界值处启动风机，根据实验数据可知，温度值在临界值后会继续上升一段时间，这样风机势必做更多的功来降温。因此，在 8：15 分时，就启动风机，用预

测的可能温度值作为模糊控制的输入，来控制风机的转速。

三个风机启动以后，三个检测点的温度测量值如表 5.3 所示：

表 5.3 实际温度测量值

时间 / 温度	8:15	8:16	8:17	8:18	8:19	8:20	8:21	8:22	8:23	8:24
1#	23.25	23.20	23.15	23.15	23.10	23.10	23.10	23.15	23.10	23.10
2#	23.20	23.20	23.15	23.15	23.15	23.10	23.10	23.10	23.10	23.10
3#	23.15	23.10	23.10	23.05	23.10	23.10	23.10	23.15	23.10	23.10

将表中数据转换为平滑曲线图，如图 5.3 所示，可以看出三个检测的温度在经过 5 分钟左右的降温震荡之后趋于 23.1℃左右。这样各个点的温度在一段时间内能保持均衡，使得厢体取得较好的恒温效果。

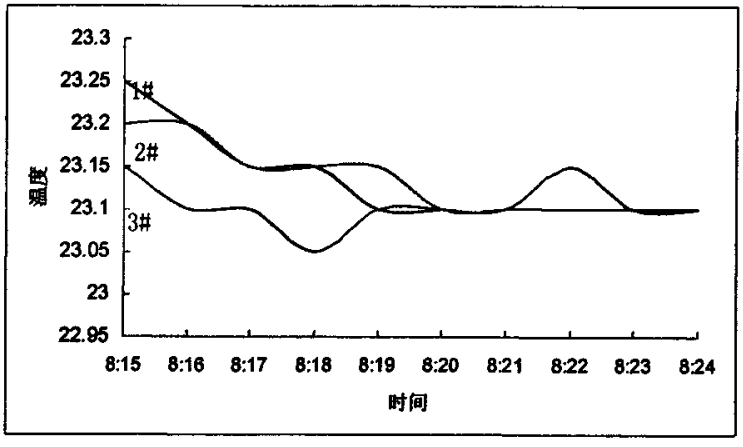


图 5.3 实际温度测量值平滑直线图

5.2.2 风机工作强度分析

正如上面所说，如果在临界值处启动风机，温度值在临界值后会继续上升一段时间，这样风机就要做更多的功来降温。因此，在 8：15 分时，就启动风机，用预测的可能温度值作为模糊控制的输入，来控制风机的转速。

表 5.4 中是采用本控制系统以后 8：15 到 8：24 三个风机转速情况。

表 5.4 风机转速

时间 / 温度	8:15	8:16	8:17	8:18	8:19	8:20	8:21	8:22	8:23	8:24
1#	1785	1225	1020	1020	0	0	0	1355	0	0
2#	0	1250	600	0	1235	1055	0	0	0	0
3#	1395	1395	1205	0	0	0	0	755	0	0

将表 5.4 中数据转化为平滑曲线如图 5.4 所示

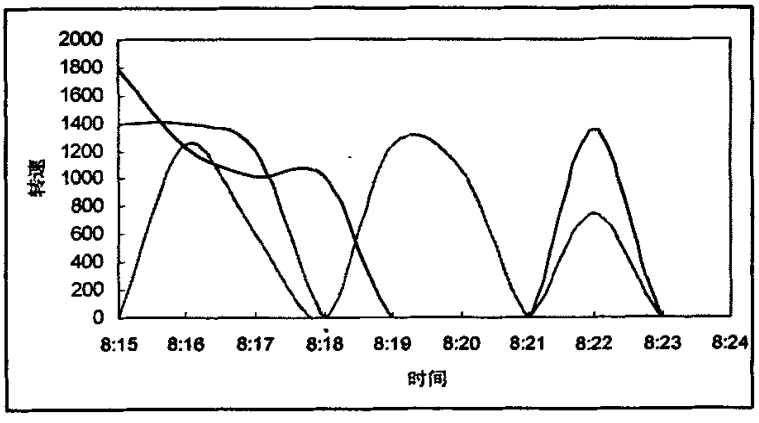


图 5.4 风机转速平滑直线图

根据表中数据可以分析出，在采用了基于神经网络和模糊控制的风机控制系统以后，风机使厢体温度趋于均衡所做的工作是：15295 转。在同等情况下没有采用该系统的普通风机，根据实验数据知道平均需要 18000 多转。可以看出风机的强度至少降低了 15%。有效减少了风机的转数，延长风机寿命，提高风机工作效率，节省了能源。为企业带来较好的经济效益。

第六章 结束语

6.1 工作总结

总结整个研究过程，可以得到以下结论：

（1）将实时多任务操作系统引入到车载风机过程控制，能大大提高风机处理系统的实时性、可靠性和可扩展性；

（2）模糊控制符合人的思考习惯，计算量小，易于实现，对难以精确建模的风机控制过程很有效。

（3）神经网络 BP 算法能预测各个检测点的温度变化趋势，与模糊控制进行结合，较有效提高风机寿命，减少风机转速。

（4）μC / OSII 是一个优秀的多任务实时内核，具有较好的可移植性，并且占用资源少，功能代码可根据需要而裁剪，工作稳定可靠。因此很适合在风机控制或其它工业过程控制中应用.

（5）将嵌入式系统应用于风机控制，易于实现风机设备设备小型化、智能化、集成化

（6）基于“伪固定”IP 地址远程通信的提出使得对数据的访问更加简单，有着广阔的实际应用价值。

6.2 进一步的工作

基于神经网络和模糊控制的车载风机智能控制系统有着广泛的实际应用价值，针对社会的发展、科技的进步和人们的需求，在以后的工作和学习中，还要加强以下几个方面的研究：

（1）深入学习、研究和分析算法

算法是软件的核心和灵魂，本论文中使用到神经网络 BP 算法和模糊控制算法，以后要进一步学习和研究这些算法，并优化该系统中的这两种算法。同时将其它控制算法也引入到该系统中，融入到该系统中，使得本控制系统能更有效的提高风机的工作效率，延长风机的寿命。

（2）图形用户界面

在本课题研究中,对于显示部分，只是根据实际情况为 LCD 编写了相对简单的驱动函数，这样足以满足课题的要求。但对于基于 32 位嵌入式处理器的系统，有着较高的运算速度和大容量的内存，可以设计更复杂友好的用户界面，为此需要为系统建立图形用户接口(GUI)函数，编制功能强大的绘图函数。使得用户可以容易、直观的明白系统的运行状态和分析的结果。

（3）网络功能

随着信息化的发展，也需要信息共享，可以在此基础上扩展风机控制系统的网络功能，使系统具有联网功能。在嵌入式系统中加入 TCP/IP 的协议栈实现联网功能。

参考文献：

[1]王田苗，嵌入式系统设计与实例开发，清华大学出版社，2002 年

[2]陈善广，BP 神经网络学习算法研究，应用基础与工程科学学报，1995 年 04 期

[3] 蔡宁， 赵英凯，BP 神经网络结构的改进及其在智能控制中的应用，化学工业与工程技术，1995 年 04 期

[4] 刘素芳， 基于模糊控制的洗衣机水位控制的仿真，山西电子技术，2007 年 01 期

[5] 张立志，王玲，高希彦，BP 神经网络在加热炉控制系统中的应用，大连理工大学学报，1996 年 05 期

[6] 刘伟，费仁元.，智能监控中改进 BP 神经网络模型的应用研究.，北京工业大学学报，1996 年 02 期

[7] 路桂明，周美兰，林治楠.，电锅炉智能控制系统的设计，机械研究与应用，2007 年 01 期

[8] 刘龙江，浅析模糊控制及其一种模糊控制器的设计，陕西国防工业职业技术学院学报， 2006 年 01 期

[9] 王振，柳志华，客车冷却系统模糊控制策略及仿真，轻型汽车技术，2007 年 01 期

[10] http://www.laogu.com/，老古开发网

[11] http://www.zlgmcu.com/01.htm，周立功单片机网

[12] http://www.embedlinux.cn/default.asp，嵌入式开发联盟论坛

[13] http://www.embedworld.com/，嵌入式世界网

[14] http://www.21control.com/bbs/，嵌入式控制研究室论坛

[15] http://www.jgchina.com/ednns/ednns.htm，智能控制

[16] http://www.itpub.net/，itpub 论坛

[17] http://www.madio.cn/BBS/dispbbs.asp，矩阵工作室

[18] http://bbs.matwav.com/，研学论坛

[19]http://www.ednchina.com/blog/nagao/13424/message.aspx，EDN 博客

[20] 赵海兰，智能温度传感器 DS18B20 的原理与应用，沙洲职业工学院院报

[21] 邵贝贝译，嵌入式实时操作系统 p C／OS-II（第 2 版），北京航空航天大学出版社，2003

[22] 刘洋，雷霆， 嵌入式系统软硬件功能分配的研究，计算机工程，2003

[23] 李海波，何熙文，89C51 单片机在模糊测控系统中的应用微处理机，2002

[24] 邵贝贝，龚光，μC／OS-II 使用中的几个热点问题，世界电子元器件，2002

[25] 王保进，刘恒禹，μC／OS-11 实时操作系统内存管理的改进，电子技术应用，2002

[26] 黄友锐，基于模糊推理的自整定 PID 控制器，微计算机信息，2003

[27] 韩志军，刘新民，数字温度传感器 DS18B20 及其应用，南京工程学院学报，自然科学版，2003

[28] 飞思科技产品研究中心，神经网络理论与 MATLAB 7 实现，电子工业出版社，2005 年

[29] 朱大奇，史慧，人工神经网络原理及应用，科学出版社，2006 年

[30] 周志华，曹存根，神经网络及其应用，清华大学出版社，2002 年

[31]. 楼顺天，施 阳，基于 MATLAB 的系统分析与设计——神经网络，西安电子科技大学出版社，2001 年

[32]. 尚宁，基于 BP 神经网络的路口短时交通流量预测方法，2006 年

[33]. 蔡自兴，徐光祐，人工智能及其应用（第二版），清华大学出版社，2004 年

[34]. 王万森，人工智能原理及其应用，电子工业出版社，2007 年

[35]. 马忠梅等，单片机的 C 语言应用程序设计，北京航空航天大学出版社，2003 年

[36] 陆锦军，黄忠良等，单片机应用新技术教程，电子工业出版社，2005 年

[37] Specially Designed for Engineers and Technicians of Electronics，西安电子科技出版社，2005

[38]. 王幸之等， AT89 系列单片机原理与接口技术，2004 年 5 月

[39]. 曹国华等，高速嵌入式单片机原理与接口技术，国防工业出版社，2005 年 7 月

[40]. 沈红卫，基于单片机的智能系统设计与实现，电子工业出版社，2005 年 1 月

[41] （美）Douglas E. Comer、 David L. Stevens，TCP/IP 网络互联技术（卷 3）——客户-服务器编程与应用（Windows 套接字版），清华大学出版社，2004 年 9 月

攻读硕士学位期间公开发表的论文

[1] 陈立平 徐汀荣. 基于 BP 神经网络模型的温度预测方法。沙洲职业工学院学报，已发表，2006 年，第九卷，第三期

[2] 陈立平 周颖平. 基于伪固定 IP 的远程监控管理系统。科技信息，已发表，2007 年，第六期

致谢

衷心感谢我的导师徐汀荣教授。在研究生学习期间，徐老师在公务繁忙之中仍给予了我很大的帮助和对论文的悉心指导。徐教授忘我的拼搏精神、一丝不苟的钻研精神、充满活力的创新精神、锲而不舍的执着精神等，会使我终生受益匪浅。另外，也要感谢苏大陆建德、王林、张广泉、王宜怀等各位教授，他们深厚的学术功底、清晰的条理、精彩的讲课令笔者深受教益，笔者从这些老师身上学到了很多东西。他们认真负责的工作态度，严谨的治学精神和深厚的理论水平都使笔者终生难忘。他们无论在理论上还是在实践中，都给予笔者很多的帮助，使笔者得到很大的提高，这对于笔者以后的工作和学习都有一种巨大的帮助，在此表示感谢。

感谢单位领导和同事们，是他们的支持和帮助使我能够顺利地完成研究生学业。

上位机和下位机的联合控制是我大学时的梦想，我带着这个梦想已经有多年了，今天在徐教授的指导下，我终于完成了自己的一个心愿。在今后的工作中我会倍加深入的学习这方面知识，完善课题的研究，让该系统有着更为广阔的应用前景。

最后，还要感谢所有关心和帮助过我的朋友。