论文题目：　　温湿度检测系统的设计与实现

目　　录

温湿度检测系统的设计与实现

学生：徐祥（指导老师：王留留）

（淮南师范学院电气信息工程学院）

摘 要：温湿度测量系统的测量的使用领域是宽广的，在仓库中、果园中、医院内都有着重要的作用。这次的毕业设计是对温湿度测量系统的研究、仿真和实现，对它以后发展和推动起了重要作用。这次的毕业设计，仔细的分析了国外与国内关于温湿度检测系统的发展情况与研究方向，阐述了当今现实生活中、工业中、农业中其存在的一些问题，在经过探讨这些问题并提出合理的解决方案的之后，系统的设计一类关于单片机的温湿度检测系统，能够比较稳定、长时间、准确的对那些有着特别要求的场所进行测量其温度与湿度。硬件电路部分与软件电路部分是该次毕业设计的两大组成的部分，所设计的系统的基本原理如下：在某环境中，给予温湿度传感器模拟的温度与湿度，这些模拟信号会通过温湿度的检测系统所涉及的电路，利用传感器把这些处理的信号传输给核心部件单片机，然后单片机在处理这些信号，再传输到 LCD 显示出数字，从而实现对温湿度的测量。

关键词：温湿度；SHT10 传感器；单片机

前言

当下的生活中，温度与湿度的技术着重的被利用于特定的环境、环境温度湿度要求比较高的区域，其使用的范围与频率还是比较多的。

在以前，各种仓库、蔬菜大棚、车间等相对环境空间内的温度和相对湿度的信号采集即温度和相对湿度的检测，是利用传统的具有指示温度和湿度的检测仪表。但是利用这种方法则需要工作人员来检测控制，然后对那些不符合条件的相对环境空间进行温湿度处理，比如通风、干燥、升降温度等。当然这种方法对于工作人员来说是很浪费时间和精力的及工作效率就很低，而且通过这些仪表检测出来的温度和相对湿度误差会很大。现如今随着工业技术和微电计算机技术的飞快发展和不断创新，仓库这类的温湿度检测智能控制的仪器仪表已经开始慢慢地推广使用了，正在逐步的代替以往的那些检测仪器，其主要是以单片机为主的温湿度智能控制，这类的温湿度监测仪器的从成本和使用角度方面考虑，是最适合仓库类的相对环境的温湿度的检测，更重要的是测量的温湿度值比较准确。

现如今的测量和控制方面的实现需要利用到传感器，它是测量和控制类系统的重要部分。我们都知道那些原始被测信号若不利用仪器对其进行准确的捕捉及进行模数转换，那么我们所需要的智能监测类系统将无法运行，而这方面问题的解决则是传感器的功劳。如果在仓库管理系统中增加温湿度智能监测系统，那么对于工作人员来说，可以相对的减轻其工作量、仓库货物的储备量和质量、减少管理方面的资金等，这对广大仓库管理者来说很有帮助。

这次的毕业设计是通过一个与温度与湿度都有关的仪器对某特定的环境下测量出的温湿度,在多次的实验之后系统的对所设计的方案进行误差分析和改良;同时相关设备处理出现的温湿度方面的问题，从而来达到对某一温度和相对湿度的智能监控目的，操作人员能够明确的观察到温湿度的变化，这样温湿度一直保持在设定的温湿度范围内。

1 温湿度检测的简介

1.1 系统的概述

温湿度测量技术在当今的工厂加工、医疗区域、农业区域中已经起来重要的位子，例如资源的节约、产品质量的提高、产品数目的提高，这些问题现在已经越来越受到外界的关注了。当今，知识信息和知识的工业化已经开始了飞一般的进步，温度与湿度的问题影响的范围距离已经不再之前谈到的那些方面，它还体现在科技发展、卫生用品、医药卫生、国家安全基础等多种方面。就上述几个问题和情况，温湿度检测的准确性、稳定性、快速性、安全性这些方面的设计要求变得尤其重要。在最近几年中，使用 SHT10 控制的温湿度传感器和温湿度数据的网上直接检验技术现已成为当下的一种发展方向和追求。本次毕业设计介绍和实现了一种单片机与自动化温湿度传感器互相结合，它们两就组成了一种简单的温湿度检测器系统。这种检测系统具有以下的特点：易操作、制作成本低、准确性较高、持续时间长、较为稳定。

1.2 系统设计选题的背景

1.2.1 国内外研究现状

关于我国国内温湿度研究的时间相对于国外还是比较晚的，毕竟我国对于温湿度检测技术的研究才刚刚起步 。初期我国只运用了相对落后的温湿度的微机控制测量技术，而这门技术还是在参考当时国外发展国家的检测技术的基础上，这门技术局限于测量单方面环境因素，不支持复杂、多项的环境控制。我国关于温湿度检测技术从对国外发达技术的学习，经过慢慢时间的不断地实验，现已经发展到微测量计算机应用的层次上。目前，国内用的技术基本上包括单片机，这种技术是利用单片机控制的温湿度检测的系统，过程与步骤都比较简单，还不能实现多参数多回路的温湿度控制系统，相对于那些发达的国家，技术还是比较落后。我国的温湿度测量存在着下列问题：实现功能少、产量水平低，操作检修步骤繁琐。

1.2.2 国外外研究现状

关于国外温湿度研究的时间相对于国内来说还是较早。国外初期首先设计出通过组合的形式的模拟式器件，运用了就地取材的方法，将其收集的信号进行一系列的指示并加以记录。近阶段世界各国都在研究与开发基于计算机的控制温湿度系统，此系统受多因子的控制，其主要特点为精确性高、稳定性强。以后温湿度发展趋势向着无人操作

化、精度稳定化发展。

1.3 系统的分类

水汽压型：测出大气中对某一装置的总压力，然后再测出大气中的水汽对同一装置的压力，将测出的两个压力进行百分比的对比压力，即可以得出温湿度的大小值。

电阻式湿度片：通过外界温湿度变化与电阻值的关系的来设计出的测量仪器。当外界的温湿度改变时，与其用电路连接的电阻也随之改变。温湿度片就是这里的核心器件，它可以感应到外界温湿度的变化。

干湿球温度表：通过两只完全相同的温度表，使他们并列在一起，其中用一只温度表测量气温，另外一支温度表表头需要缠绕着浸透过纯蒸馏水的脱脂纱布，这两种温度表结合起来就是干湿球温度表。

1.4 系统设计的内容与要求

对某一特定环境下用温室度传感器感受到温度和湿度变化，把这种变化转化为电信号输入到单片机中，然后进行各端口的控制使其数据显示在 LCD 显示屏上，完成了对仓库额的温室与湿度的测量。要求误差在上下 10%之内。

2 系统设计方案

2.1 温湿度检测系统方案制定

方案一

温湿度的检测与温湿度的显示构成了温湿度传感器，温湿度的传感器有好多种，设计方案一中温湿度传感器选用的是 SHT10。为了营造无人看守状态,本次设计中还应用了远程通信系统。方案一中系统的控制核心是 AT89C52 单片机，它的主要作用是读取温湿度传感器工作时的内部参数，测试的结果可以显示在 LCD 上面。

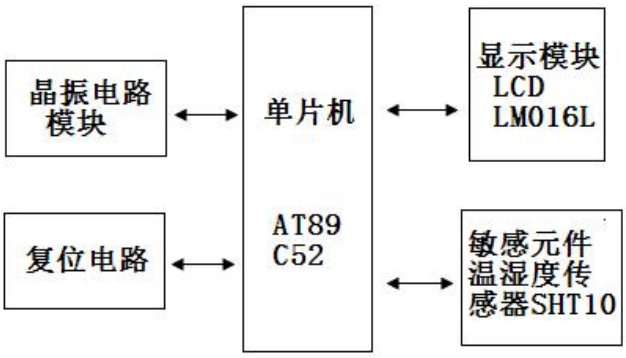

图 1 仓库温湿度检测系统原理图

方案二

此方案是温度和湿度电路的设计，我们可以使用热敏电阻和湿敏电阻之类的仪器来检测，主要是利用它们的感温效应和感湿效应。第一步是利用温敏电阻与热敏电阻的原理特性感应到外界湿度与温度的变化并对其进行电流与电压的采集，第二步是通过 A/D 模数的变化，第三步运行到了关键的部分单片机，最后是通过 LCD 屏幕显示出测量的结果来，与此同时已经测量出来的温湿度和之前设定的温湿度进行比较。如果采用方案二的话，该方案涉及到了 A/D 数模转换电路设计、感湿电路的设计和感温电路的设计，这样设计的步骤与程序比较繁琐。

综合这两种方案，方案一较为简单明朗，运用所学的知识较多，故选择方案一来实现温湿度检测系统的设计与实现。

2.2 系统功能模块分析

2.2.1 中央控制单元

本次温湿度检测系统设计中，由单片机组成的中央控制单元有十分重要的作用。这是整个系统的大脑，它发出操作命令指挥系统工作。该单片机不仅可以控制 LCD 显示屏幕的工作状态，还可以时时刻刻管理着监测着外部环境的温湿度的变化的温湿度传感器的工作状态。依照所需设计的要求和控制的目的，本次毕业设计选择了 AT89C52 芯片，该芯片里面包含 4k Bytes ISP 的能多次烧入的 Flash 器件，是一类简单高效率的 CMOS 8 位芯片。AT89C52 芯片是使用了 ATMEL 公司厂家中的较为先进高级的控制与制作技术做为支持动力。AT89C52 芯片还包括 MCS-52 系统的操作命令与 89C52 管脚的排列，其中较重要的 8 位 CPU 和 ISP Flash 存储单元是它的核心部件。

AT89C52 芯片的系统功能具有巨大的优势，它可以满足设计中系统稳定运行的基本要求。AT89C52 芯片具 16 位可编程定时计数器 3 个，有引脚 40 个，全双工串行通信口 2 个，外部双向输入/输出（I/O）端口 32 个，外中断口 2 个，读写口线 2 个，AT89C52 芯片的管脚结构如下图所示：

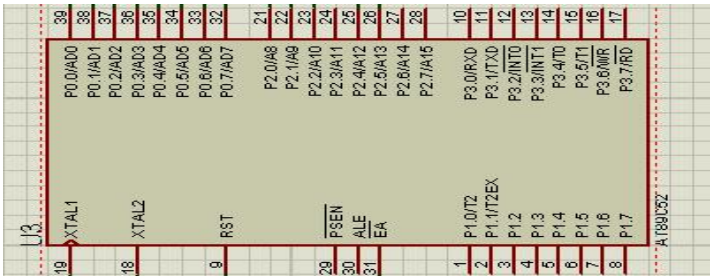

图 2 单片机

2.2.2 晶振电路模块与复位电路模块

晶振电路模块：单片机的工作条件是要在时钟驱动的作用下才可以稳定的进行工作,所需的电容大小通常为 30PF。单片机工作时需要一个信号脉冲，晶振的作用就是提供这个信号脉冲。在时钟驱动作用下，晶振电路所提供的信号脉冲就是单片机的工作速度。举个例子来说明，一个频率为 12MHZ 的晶振电路芯片，它的工作速度是 12MHZ 每秒的运行速度,和我们使用的电脑手机的 CPU 一个道理。就于多大的频率才能使单片机更好的更稳点的工作的问题，一般情况下其工作时所需要的频率在 24MHZ 左右，超过这个值，系统工作就不稳定了。单片机系统的工作速度取决于时钟信号，其内部镶有时钟振荡电路，在单片机的外部接通一个振荡源就可以工作了。

复位电路模块:复位电路在设计的系统中起着重要的作用，它保障了设计的系统可以在稳定的环境下工作，复位电路的主要作用功能就是上电复位。当复位信号消除的时候，系统微机电路才可以稳定高效的工作，消除复位信号的条件是 VCC 的电压在 4.7V 与 5.2V 之间，只有在提供稳定无误差的时钟信号才能实现本次的设计。下图为其仿真图：

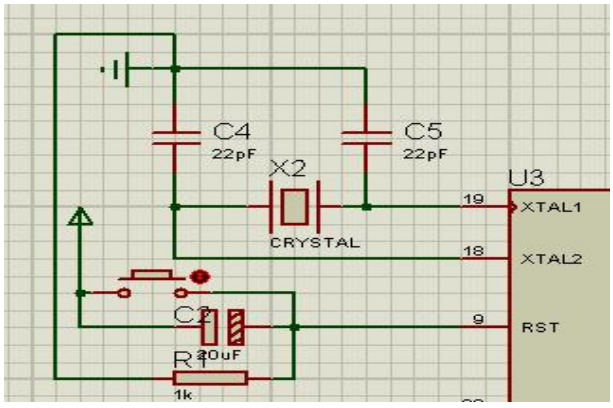

图 3 晶振电路和复位电路

2.2.3 显示模块

LCD 显示电路是本系统的功能具体体现的重要模块,实现了对温湿度检测的液晶屏控制的功能。温湿度显示电路的组成有 SHT10 温湿度传感器、LCD 液晶显示屏幕。其显示模块先接受来自单片机处理后的信号，再将其结果显示在液晶屏幕。下图为其仿真图：

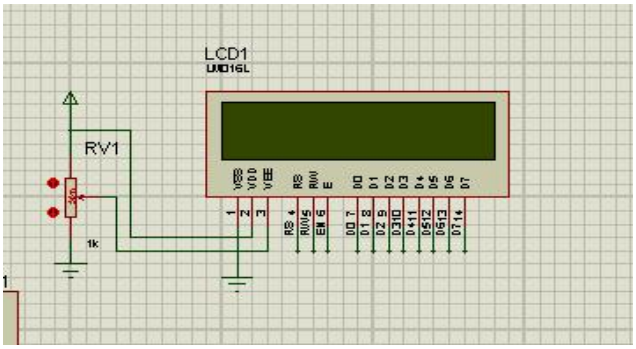

图 4 显示模块

2.2.4 温湿度传感器

利用型号为 SHT10 的温湿度传感器来测试仓库的温度和湿度。下图为其仿真图：

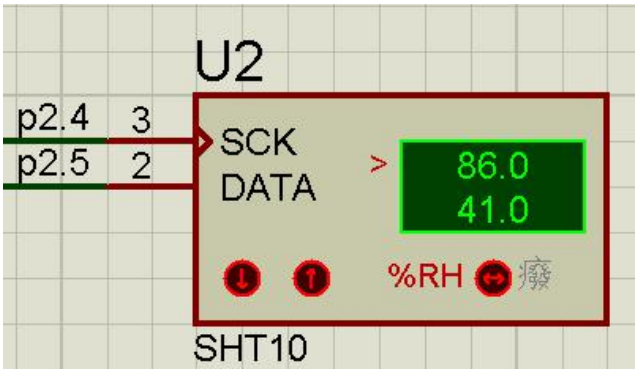

图 5 温湿度传感器

仿真图上三个按键：↑ ↓ ↔来控制操作。当↔打到左边时为湿度的调节，当↔打到右边的时候为温度的调节;↑是增大按键，↓是减小按键。

2.3 仿真器件

2.3.1 温湿度传感器的选择及介绍

选择 SHT10 温湿度传感器.可同时测量温度和湿度。精确度,高测量范围大,便于远距离测量。SHT10 是瑞士 Scnsirion 公司推出的一款数字温湿度传感器芯片。SHT10 温湿度传感器的接口是由 SCK 与 DATA 两个串行接口组成的，它可以实现 CRC 的校验传输，而且准确性高。SHT10 温湿度传感器利用的是 SMD 表面贴片设计之后的封装形式，管脚排列如图 6 所示，SHT10 的引脚说明如下：NC 是空管脚，GND 是接地端线，SCK 是串行时钟输入 DATA，—双向串行数据线，VDD 电源端是 0.5V 至 5.5V 电源端。

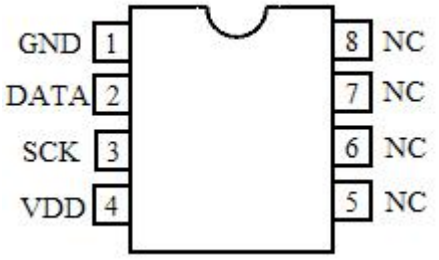

图 6 SHT10 外形及引脚排列

SHT10 功能齐全，将温度检测电路、湿度检测电路、数模转化器、微信号处理这些功能全部集成到 SHT10 芯片上面。讲这些功能具体、聚集化，用起来较为方便、快捷。

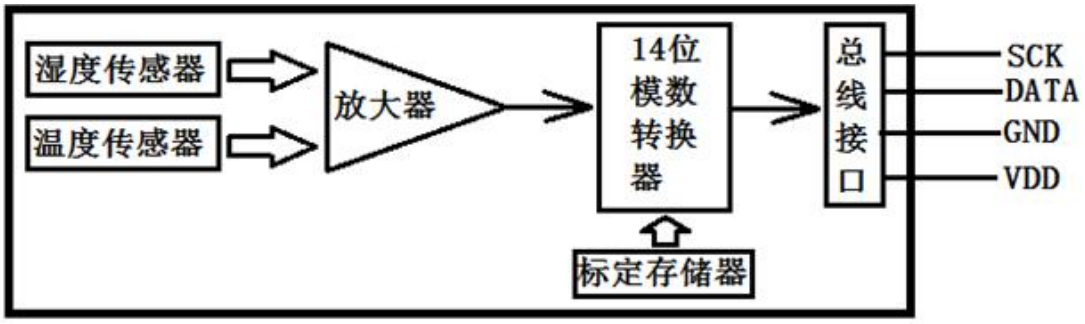

图 7 SHT10 温湿度传感器**原理图**

2.4 本章小结

本章首先进行系统的方案论证。6 根据对系统功能的定义，初步完成了系统软硬件的框图设计。接着分别介绍和分析中央控制单元模块、晶振电路模块、复位电路模块、显示电路模块以及主要器件的选型。

3 系统仿真调试

3.1 PROTEUS 对系统仿真

3.1.1 软件 Proteus 概述

Proteus 设计软件是由英国 Labcenter 公司针对模拟电路单独设计的一种仿真软件。此软件能在电脑系统中进行操作，可以有效的仿真出集成电路中许多的模拟器件。该软件能实现各种单片机电路的仿真，具有 A/D 转换、D/A 转换电路与 LED 液晶屏仿真具备许多虚拟仪器的功能，例如数字信号发生器数字示波器、与非门电路逻分析仪、数字示波器。

Proteus 是一种针对于单片机的实现与仿真的一款软件。其中它支持很以下单片机的系列：PIC16 类型、8051 类型、HC11 类型、PIC12 类型、Z80 类型、AVR 理性等多种芯片。

Proteus 软件还具有软件调试的功能。在许多硬件的仿真过程中拥有着多种调试功能例如设置断点、全速、单步等，同时可以仔细的观察到每个变量的状态，用该软件仿真电路中，同样的也具有此项功能，带动着第三方的软件编译与调试所需要的环境。

Proteus 软件如今是全世界内最全面、最权威的仿真平台。

3.1.2 Proteus 对系统仿真

根据设计要求，从 Proteus 元件库中找到所需要用到的元件，画好电路图并且检查

有无错误。最后通过 keil uVision 软件编写的 C 语言程序，转换成 HEX 文件下载到画好的的电路上进行调试。以下是系统的仿真电路图：

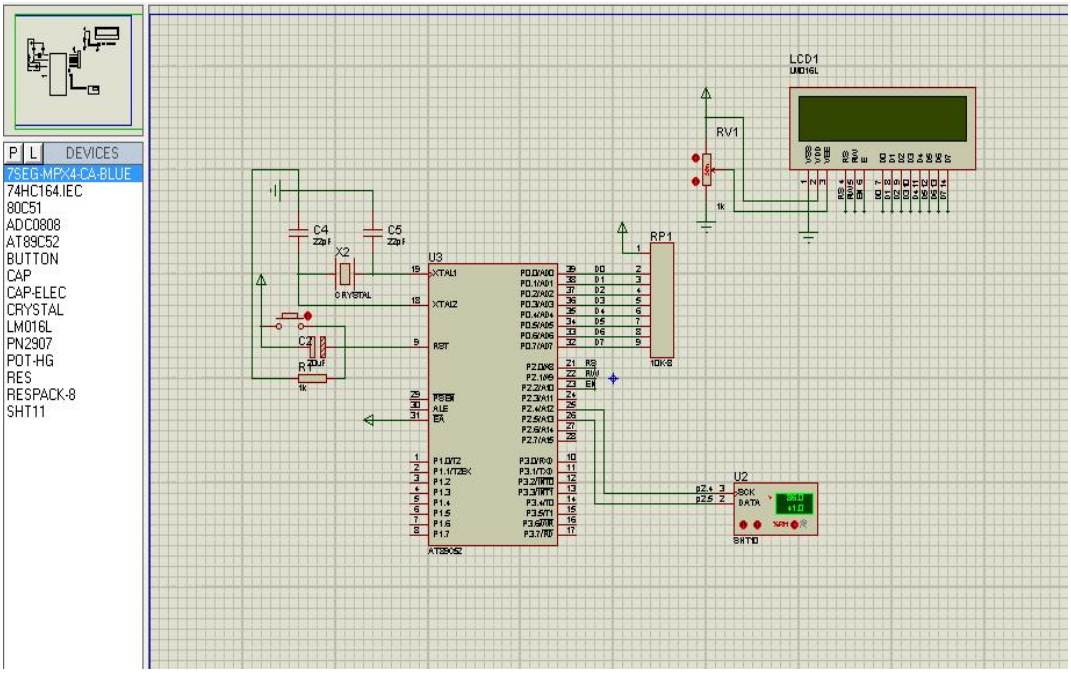

图 8 系统仿真总电路图

通过调节 SHT10 的温湿度传感器两个按钮“↑”“↓”来调节给定的模拟的温度和湿度的大小，调节之后，可以在 LCD 屏幕上观察显示屏出现的温度湿度的测量值。

在一定的环境行下，给予温湿度传感器模拟温度与湿度，其演示效果如图 8 所示;在经过单片机的处理分析后，测量的值会出现在 LCD 屏幕上，其效果图如图 9 所示。

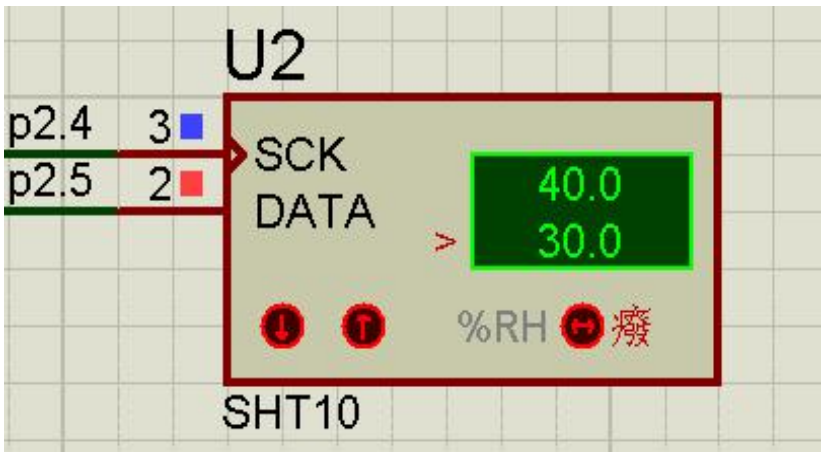

图 9 温湿度传感器的显示

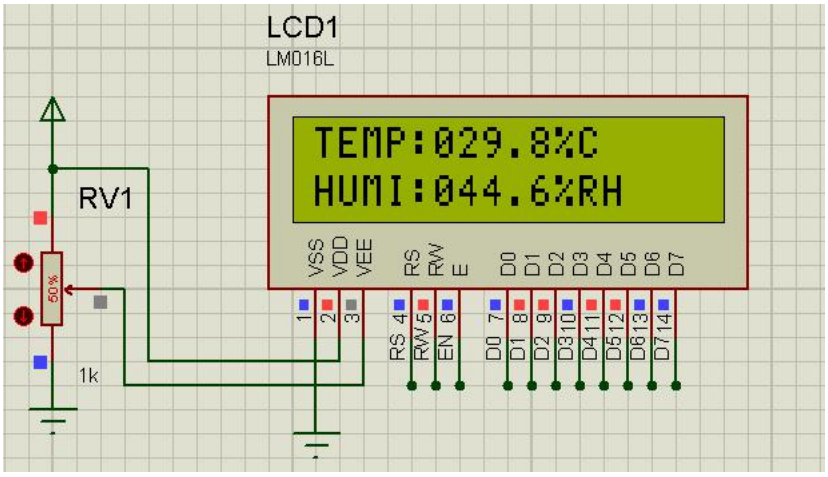

图 10 LCD 屏幕的显示

为了进一步研究系统的分别对温度湿度进行了 5 组数据的实现，其记录数据如下表：

测试对象与次数	1	2	3	4	5
实际温度℃	15	20	25	30	35
测量温度℃	15.2	20.1	24.9	29.8	34.6
误差大小	0.2	0.1	0.1	0.2	0.4

表一 温度的测量

测试对象与次数	1	2	3	4	5
实际湿度	35	40	45	50	55
测量湿度	40.5	45.8	51.1	56.5	59.8
误差大小	5.5	5.8	6.1	6.5	4.8

表二 湿度的测量

3.2 误差分析

a. SHT10 的温湿度传感器在进行测量时存在一定范围的误差。

b. 单片机的编程程序出现一些错误。

通过以上几种仿真的结果可以说明：主要用中央控制核心器件 AT89C52 单片机可以实现关于温湿度测量系统所需要的设计要求，再根据所记录表格可看出，虽然存在一定的误差，但在许可的误差范围 10%之内，所以这次的设计可以用来测量温度与湿度两项指标。

3.3 本章小结

本章是整个毕业设计的核心章节，要熟练的掌握 PROTEUS 的基本仿真功能，要通过多组实验来验证该系统的正确性、准确性与稳点性，认真的将数据填入表格中，合理对所测量的数据进行误差分析，得出相应的结论。通过仿真的设计与实现，本次设计系统是可以测量温湿度的，是满足设计的需要。

总结

本设计简单分析了温湿度控制系统，并按照有关要求完成了以高效单片机 AT89C52 作为核心，从而实现温湿度智能监测控制的系统设计。设计中的温湿度传感器 SHT10 集温度传感器和湿度传感器于一体来进行采集与测量，它自带 A/D 转换器，因而该温湿度控制仪器具有精度高、体积小、良好的抗干扰能力，故该系统具有很高的实用性。

原理图的绘制使我从新学习了一次 proteus，对软件种元器件更加熟悉，画仿真图时更为流畅。在进行设计之前有着很多要解决的问题，比如元器件的选择问题、各个模块的设计和主程序的编程。通过这次设计，我从到图书馆的网站查找相应的资料应用到对应电路参与设计的思考。每个模块都要经过多次的设计，不断的试验，让我对之前在学校所学的书本上的理论知识有了更为深刻的了解。在完成毕业设计的过程是一次难得的理论与实际相结合的过程，在这段时间我更为深刻的理解和掌握了大学期间所学的一些知识，例如 C 语言的编程、数字模拟电路、单片机的简单应用、proteus 和 keil 软件的使用与设计

该系统可以在许多环境下进行对温度、湿度的检测。这次毕业设计比中的系统中的显示模块可以设计的更加合理化，针对一些日常检查的工作，显示模块可以记录一天内需要测量的环境下的温湿度，这样更有利于各种的需要，鉴于本人的设计能力和设计要求，这里就不将该设计思路具体化了。

参考文献

[1] 王海宁.关于单片机的温度控制系统的研究[D].合肥工业大学硕士学位论文,2008:32~40.

[2] 黄贤武,郑筱霞,曲波等.传感器实际应用电路设计[M].第一版.成都:电子科技大学出社,1997:4~10.

[3] 陈曾平.《电路设计基础》[M].第二版.北京:北京高等教育出版社,2003:100~110.

[4] 余永权. 《单片机原理及应用》[M].第二版.北京:电子工业出版社,1997:47~48.

[5] 刘春恰.数字温度传感器 DS18B20 测温的应用.电器时代[J],2010（10):18~23.

[6] 周月霞,孙传友.DS18B20 硬件连接及软件编程.传感器世界[J],2001(12):25~29.

[7] 鹿红玉,戴彦,江培蕾.基于 PROTEUS 的 DS18B20 数字温度计的仿真实验[M].第一版.福建:福建电脑出版社,2010:4~20.

[8] 张毅刚.新编《MCS-51 单片机应用设计》[M].哈尔滨:哈尔滨工业大学出版社,2003:7~25.

[9] 张义和等.例说 8051[M].北京:人民邮电出版社,2006:14~29.

[10] 朱滨峰,徐桂云,李俊敏.单片机在温湿度测量系统中的应用.仪器仪表标注化与计量[J].2006(1):13~20.

[11] 贾振国.DS1820 及高精度温度测量的实现.电子技术应用[J],2000(1):58~59.

[12] 赵娜是,赵刚,于珍珠等.基于 51 单片机的温度测量系统.微计算机信息[J],2007(1-2):146~148.

[13] 刘同法,陈忠平等.《单片机基础与最小系统实践》[M].北京航空航天大学出版社,2007:23~34.

[14] 陈忠华.《基于单片机的温度智能控制系统的设计与实现》[D].大连理工大学硕士学位论文,2006:46~57.

附录 1

```
include<reg51.h>
#include <intrins.h>

#define uchar unsigned char
#define noACK 0                                    //继续传输数据，用于判断是否结束通讯
#define ACK   1             //结束数据传输；
                                                                  //地址  命令  读/写
#define STATUS_REG_W 0x06   //000   0011    0
#define STATUS_REG_R 0x07   //000   0011    1
#define MEASURE_TEMP 0x03   //000   0001    1
#define MEASURE_HUMI 0x05   //000   0010    1
#define RESET        0x1e   //000   1111    0

enum {TEMP,HUMI};

sbit DATA = P2^5;
sbit SCK = P2^4;

sbit RS  = P2^0;
sbit RW  = P2^1;
sbit E  = P2^2;
sfr DBPort  = 0x80;      //P0=0x80,P1=0x90,P2=0xA0,P3=0xB0.数据端口

/********     DS1602 函数声明      ********/
void LCD_Initial();
void GotoXY(unsigned char x, unsigned char y);
void Print(unsigned char *str);
void LCD_Write(bit style, unsigned char input);


/********     SHT10 函数声明       ********/
void s_connectionreset(void);
char s_measure(unsigned char *p_value, unsigned char *p_checksum, unsigned char mode);
void calc_sth10(float *p_humidity ,float *p_temperature);
//float calc_dewpoint(float h,float t);



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

```
//写字节程序
char s_write_byte(unsigned char value)
{
                unsigned char i,error=0;
                for (i=0x80;i>0;i>>=1)             //高位为 1，循环右移
                {
                                                if (i&value) DATA=1;          //和
要发送的数相与，结果为发送的位
                else DATA=0;
                SCK=1;
                _nop_();_nop_();_nop_();        //延时 3us
                SCK=0;
                }
                DATA=1;                            //释放数据线
                SCK=1;
                error=DATA;                        //检查应答信号，确认通讯
正常
                _nop_();_nop_();_nop_();
                SCK=0;
                DATA=1;
                return error;                      //error=1 通讯错误
}

//读字节程序
char s_read_byte(unsigned char ack)
//---------------------------------------------------------------------------
---------
{
                unsigned char i,val=0;
                DATA=1;                            //释放数据线
                for(i=0x80;i>0;i>>=1)              //高位为 1，循环右移
                {
                                                SCK=1;
                if(DATA) val=(val|i);          //读一位数据线的值
                SCK=0;
                }
                DATA=!ack;                         //如果是校验，读取完后结
束通讯;
                SCK=1;
                _nop_();_nop_();_nop_();           //延时 3us
                SCK=0;
                _nop_();_nop_();_nop_();
                DATA=1;                            //释放数据线
```

```
                return val;
}

//启动传输
void s_transstart(void)
// generates a transmission start
//       _____         ________
// DATA:      |_______|
//           ___     ___
// SCK : ___|   |___|   |______
{
                DATA=1; SCK=0;                    //准备
                _nop_();
                SCK=1;
                _nop_();
                DATA=0;
                _nop_();
                SCK=0;
                 _nop_();_nop_();_nop_();
                SCK=1;
                _nop_();
                DATA=1;
                _nop_();
                SCK=0;
}


//连接复位
void s_connectionreset(void)
// communication reset: DATA-line=1 and at least 9 SCK cycles followed by
transstart
//                    ___________________________________________________
_______
// DATA:                                                        |_______|
//          _    _    _    _    _    _    _    _    _        ___     ___
// SCK : __| |__| |__| |__| |__| |__| |__| |__| |__| |______|   |___|   |______
{
                unsigned char i;
                DATA=1; SCK=0;                    //准备
                for(i=0;i<9;i++)                   //DATA 保持高，SCK 时钟触
发 9 次，发送启动传输，通迅即复位
                {
                                          SCK=1;
                SCK=0;
```

```
                }
                s_transstart();                    //启动传输
}


//软复位程序
char s_softreset(void)
// resets the sensor by a softreset
{
                unsigned char error=0;
                s_connectionreset();               //启动连接复位
                error+=s_write_byte(RESET);        //发送复位命令
                return error;                      //error=1 通讯错误
}


/*读状态寄存器
char s_read_statusreg(unsigned char *p_value, unsigned char *p_checksum)
//---------------------------------------------------------------------------
---------
// reads the status register with checksum (8-bit)
{
                unsigned char error=0;
                s_transstart();                    //transmission start
                error=s_write_byte(STATUS_REG_R); //send command to sensor
                *p_value=s_read_byte(ACK);          //read status register
(8-bit)
                *p_checksum=s_read_byte(noACK);   //read checksum (8-bit)
                return error;                       //error=1 in case of no
response form the sensor
}

//写状态寄存器
char s_write_statusreg(unsigned char *p_value)
// writes the status register with checksum (8-bit)
{
                unsigned char error=0;
                s_transstart();                    //transmission start
                error+=s_write_byte(STATUS_REG_W);//send command to sensor
                error+=s_write_byte(*p_value);     //send value of status
register
                return error;                      //error>=1 in case of no
response form the sensor
}
```

```
                                                        */

//温湿度测量
char s_measure(unsigned char *p_value, unsigned char *p_checksum, unsigned char
mode)
// 进行温度或者湿度转换，由参数 mode 决定转换内容；
{
//                enum {TEMP,HUMI};          //已经在头文件中定义
                  unsigned error=0;
                  unsigned int i;

                  s_transstart();                    //启动传输
                  switch(mode)                      //选择发送命令
    {
                                          case            TEMP            :
error+=s_write_byte(MEASURE_TEMP); break;    //测量温度
                  case HUMI : error+=s_write_byte(MEASURE_HUMI); break;
//测量湿度
                  default     : break;
                  }
                  for (i=0;i<65535;i++) if(DATA==0) break; //等待测量结束
                  if(DATA) error+=1;                // 如果长时间数据线没有拉
低，说明测量错误
                  *(p_value) =s_read_byte(ACK);     //读第一个字节，高字节
(MSB)
                  *(p_value+1)=s_read_byte(ACK);    //读第二个字节，低字节
(LSB)
                  *p_checksum =s_read_byte(noACK); //read CRC 校验码
                  return error;                        // error=1 通讯错误
}

//温湿度值标度变换及温度补偿
void calc_sth10(float *p_humidity ,float *p_temperature)
{
                  const float C1=-4.0;              // 12 位湿度精度 修正公式
                  const float C2=+0.0405;           // 12 位湿度精度 修正公式
                  const float C3=-0.0000028;        // 12 位湿度精度 修正公式
                  const float T1=+0.01;              // 14 位温度精度 5V 条件
修正公式
                  const float T2=+0.00008;           // 14 位温度精度 5V 条件
修正公式

                  float rh=*p_humidity;             // rh:      12 位 湿度
                  float t=*p_temperature;           // t:       14 位 温度
```

```
                float rh_lin;                       // rh_lin: 湿度 linear 值
                float rh_true;                      // rh_true: 湿度 ture 值
                float t_C;                          // t_C   : 温度 ℃

                t_C=t*0.01 - 40;                    //补偿温度
                rh_lin=C3*rh*rh + C2*rh + C1;       //相对湿度非线性补偿
                rh_true=(t_C-25)*(T1+T2*rh)+rh_lin;   //相对湿度对于温度依
赖性补偿
                if(rh_true>100)rh_true=100;         //湿度最大修正
                if(rh_true<0.1)rh_true=0.1;         //湿度最小修正

                *p_temperature=t_C;                 //返回温度结果
                *p_humidity=rh_true;                //返回湿度结果
}

//从相对温度和湿度计算露点
/*float calc_dewpoint(float h,float t)
{
                float logEx,dew_point;
                logEx=0.66077+7.5*t/(237.3+t)+(log10(h)-2);
                dew_point = (logEx - 0.66077)*237.3/(0.66077+7.5-logEx);
                return dew_point;
}                                                                    */

/*************************************************************************
*************************************************************************
******/
//DS1602 程序（1602.c）：

//#include<tou.h>

//          内          部          等          待          函          数
*****************************************************************
unsigned char LCD_Wait(void)
{
    RS=0;
    RW=1;    _nop_();
    E=1;    _nop_();
    E=0;
    return DBPort;
}
//      向      LCD      写      入      命      令      或      数      据
*************************************************************
#define LCD_COMMAND         0       // Command
```

```
#define LCD_DATA            1       // Data
#define LCD_CLEAR_SCREEN    0x01        // 清屏
#define LCD_HOMING          0x02        // 光标返回原点
void LCD_Write(bit style, unsigned char input)
{
    E=0;
    RS=style;
    RW=0;         _nop_();
    DBPort=input;    _nop_();//注意顺序
    E=1;         _nop_();//注意顺序
    E=0;         _nop_();
    LCD_Wait();
}

//设置显示模式*****************************************************************
#define LCD_SHOW            0x04    //显示开
#define LCD_HIDE            0x00    //显示关

#define LCD_CURSOR          0x02    //显示光标
#define LCD_NO_CURSOR       0x00    //无光标

#define LCD_FLASH           0x01    //光标闪动
#define LCD_NO_FLASH        0x00    //光标不闪动

void LCD_SetDisplay(unsigned char DisplayMode)
{
    LCD_Write(LCD_COMMAND, 0x08|DisplayMode);
}

//设置输入模式*****************************************************************
#define LCD_AC_UP       0x02
#define LCD_AC_DOWN         0x00        // default

#define LCD_MOVE            0x01        // 画面可平移
#define LCD_NO_MOVE         0x00        //default

void LCD_SetInput(unsigned char InputMode)
{
    LCD_Write(LCD_COMMAND, 0x04|InputMode);
}

//初始化 LCD*******************************************************************
void LCD_Initial()
{
```

```
    E=0;
    LCD_Write(LCD_COMMAND,0x38);            //8 位数据端口,2 行显示,5*7 点阵
    LCD_Write(LCD_COMMAND,0x38);
    LCD_SetDisplay(LCD_SHOW|LCD_NO_CURSOR);     //开启显示, 无光标
    LCD_Write(LCD_COMMAND,LCD_CLEAR_SCREEN);    //清屏
    LCD_SetInput(LCD_AC_UP|LCD_NO_MOVE);        //AC 递增, 画面不动
}

//液晶字符输入的位置**********************
void GotoXY(unsigned char x, unsigned char y)
{
    if(y==0)
        LCD_Write(LCD_COMMAND,0x80|x);
    if(y==1)
        LCD_Write(LCD_COMMAND,0x80|(x-0x40));
}

//将字符输出到液晶显示
void Print(unsigned char *str)
{
    while(*str!='\0')
    {
        LCD_Write(LCD_DATA,*str);
        str++;
    }
}

/*****************************************************************************
*****************************************************************************
******/
//主函数 (main.c):

//#include<tou.h>

typedef union                                   //定义共用同类型
{
                unsigned int i;
                float f;
} value;

//延时函数
void delay(int z)                           //z 为毫秒数
{
                int x,y;
```

```
                for(x=z;x>0;x--)
                                        for(y=125;y>0;y--);
}


void main()
{
                unsigned int temp,humi;
                value humi_val,temp_val; //定义两个共同体，一个用于湿度，一
个用于温度
//              float dew_point;                //用于记录露点值
                unsigned char error;          //用于检验是否出现错误
                unsigned char checksum;       //CRC
                uchar wendu[6];                     //用于记录温度
                uchar shidu[6];                     //用于记录湿度


                LCD_Initial();                          //初始化液晶
                GotoXY(0,0);                            //选择温度显示位置
                Print("TEMP:     %C");          //5 格空格
    GotoXY(0,1);                                  //选择湿度显示位置
                Print("HUMI:     %RH");         //5 格空格
                s_connectionreset();            //启动连接复位
                while(1)
                {
                                        error=0;                    //初始
化 error=0，即没有错误
                                        error+=s_measure((unsigned
char*)&temp_val.i,&checksum,TEMP); //温度测量
                                        error+=s_measure((unsigned
char*)&humi_val.i,&checksum,HUMI); //湿度测量
                if(error!=0) s_connectionreset();               ////如果
发生错误，系统复位
                else
                {
                                        humi_val.f=(float)humi_val.i;
//转换为浮点数
                                        temp_val.f=(float)temp_val.i;
//转换为浮点数

                calc_sth10(&humi_val.f,&temp_val.f);            //修正相对
湿度及温度
//
                dew_point=calc_dewpoint(humi_val.f,temp_val.f); // 计 算 e
```

```
dew_point
                                        temp=temp_val.f*10;
                                        humi=humi_val.f*10;
                                        GotoXY(5,0);                    //
设置温度显示位置
                                        wendu[0]=temp/1000+'0';         //
温度百位
                                        wendu[1]=temp%1000/100+'0';
                    //温度十位
                                        wendu[2]=temp%100/10+'0';
                    //温度个位
                                        wendu[3]=0x2E;                  //
小数点
                                        wendu[4]=temp%10+'0';
                    //温度小数点后第一位
                                        Print(wendu);                   //
输出温度
                                        GotoXY(5,1);                    //
设置湿度显示位置
                                        shidu[0]=humi/1000+'0';
                    //湿度百位
                                        shidu[1]=humi%1000/100+'0';
                    //湿度十位
                                        shidu[2]=humi%100/10+'0';
                    //湿度个位
                                        shidu[3]=0x2E;                  //
小数点
                                        shidu[4]=humi%10+'0';
                    //湿度小数点后第一位
                                        Print(shidu);                   //
输出湿度
                    }
                                        delay(800);
//等待足够长的时间，以现行下一次转换
                    }
```