摘要

随着数字技术的快速发展，数字技术被广泛应用于智能控制的领域中。单片机以体积小、功能全、价格低廉、开发方便的优势得到了许多电子系统设计者的青睐。它适合于实时控制，可构成工业控制器、智能仪表、智能接口、智能武器装置以及通用测控单元等。

本文以STC89C52RC单片机为核心设计了全自动洗衣机控制系统，本系统实现了对洗衣机整个洗衣过程的控制，包括时间控制、模式切换、洗衣、水位控制和结束报警等功能。控制系统主要由单片机主控系统组成。单片机主控系统负责控制洗衣机的工作过程，主要由STC89C52RC单片机、按键、蜂鸣器、水位传感器、LCD1602显示屏、继电器、三极管、电动机、步进电机驱动、红外接收器等组成。

本系统的电路并不复杂，给STC89C52RC单片机载入软件程序后，能够实现全自动洗衣机的基本功能。虽然不能与电器市场上的洗衣机控制系统媲美，但也具有一定的实用性。

关键词：单片机、洗衣机、控制

目录

课程设计报告的主要内容应包括：电路原理图、工作原理详述、电路印制板图、程序流程图及程序、系统调试结果及结论分析等；内容结构包含：封面、中文摘要与关键词、目录、论文正文、结束语、参考文献。

一、设计目的

二、设计要求

三、功能分析

四、整体结构分析

五、程序设计

六、硬件设计

七、调试结果

八、总结

九、参考文献

一、设计目的

“微机应用系统综合设计”课程设计是自动化、测控专业本科生的必修课。本课程设计综合了模电/数电技术、微机原理与接口技术、嵌入式微控制器技术、电机与电力拖动和自动控制理论等相关知识应用，通过基于嵌入式微控制器应用系统的硬件原理和软件编程的分析、设计与调试，让学生对嵌入式微控制器系统应用有一个较全面的理解，掌握基于嵌入式微控制器应用系统软硬件设计的基本方法和技术，同时，也为下一阶段相关专业课程的学习和应用设计实验，提供良好基础和平台。

二、设计要求

### **全自动洗衣机控制系统设计 （同类微控制器限1-2人选）**

设计要求：

（1）、使用汇编语言或C语言；

（2）、程序功能要求：四个按键：切换、开始/暂停、时间加、时间减；两种洗涤模式：标准和轻柔模式，或按键切换；电机具备正转、反转功能；水位自动测量功能；数码管（液晶）显示剩余时间，时间可以通过按键设置；简单故障诊断与报警。

（3）、上位机监控功能要求：通过串口或USB口实现上/下位机通信，能够显示洗衣机当前状态及运行参数（数据显示或动画显示），能够从上位机控制下位机之洗衣机工作。

三、功能分析

3.1基本功能

1.lcd显示洗衣机的运行状态；通过串口向上位机显示洗衣机的运行状态

2.按键控制洗衣机的运行

独立按键 K1开始/暂停

矩阵按键 S1模式切换、S2增时S3减时、S5低水位信号、S6高水位信号

3.标准模式电机转速快，轻柔模式电机转速慢

4.电机运作时正转5s，反转5s，循环

5.时间为0时，电机停止，蜂鸣器响

3.2进阶功能：

1.串口控制$mode=1; $mode=0; $start=0; $start=1; $add; $sub;

2.红外控制

3.水位监测：

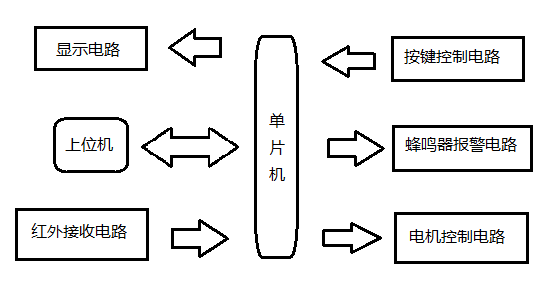
低水位时，若电机正在运转则停止电机，开启抽水模式，抽水至信号正常，10s后若仍为低水位，则停止抽水，报警

高水位时，若电机正在运转则停止电机，开启放水模式，放水至信号正常，10s后若仍为高水位，则停止放水，报警

水位恢复后，返回至原有状态

四、整体结构分析

控制系统主要由单片机控制系统和外部硬件电路构成。单片机控制系统负责控制洗衣机的工作过程，主要由STC89C52RC单片机、LCD1602显示屏、按键、蜂鸣器组成；外部硬件电路有继电器、三极管、电动机、步进电机驱动、红外接收器组成。

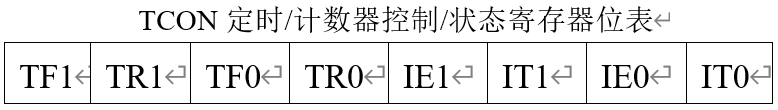


五、程序设计

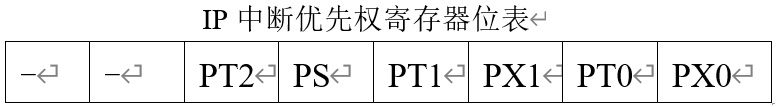
5.1端口地址定义



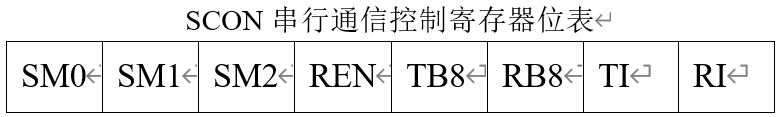




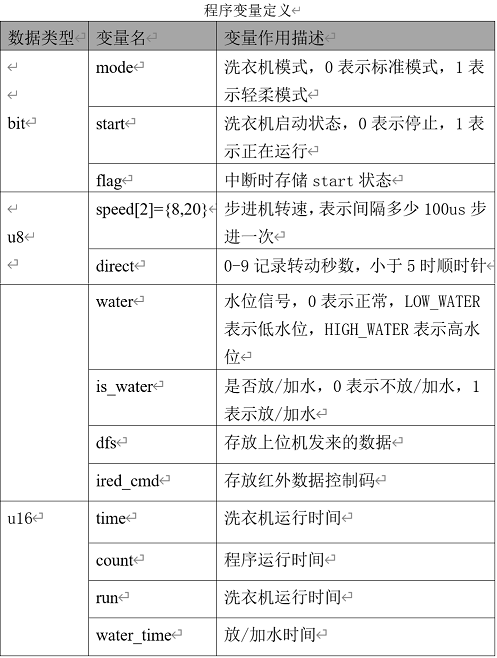


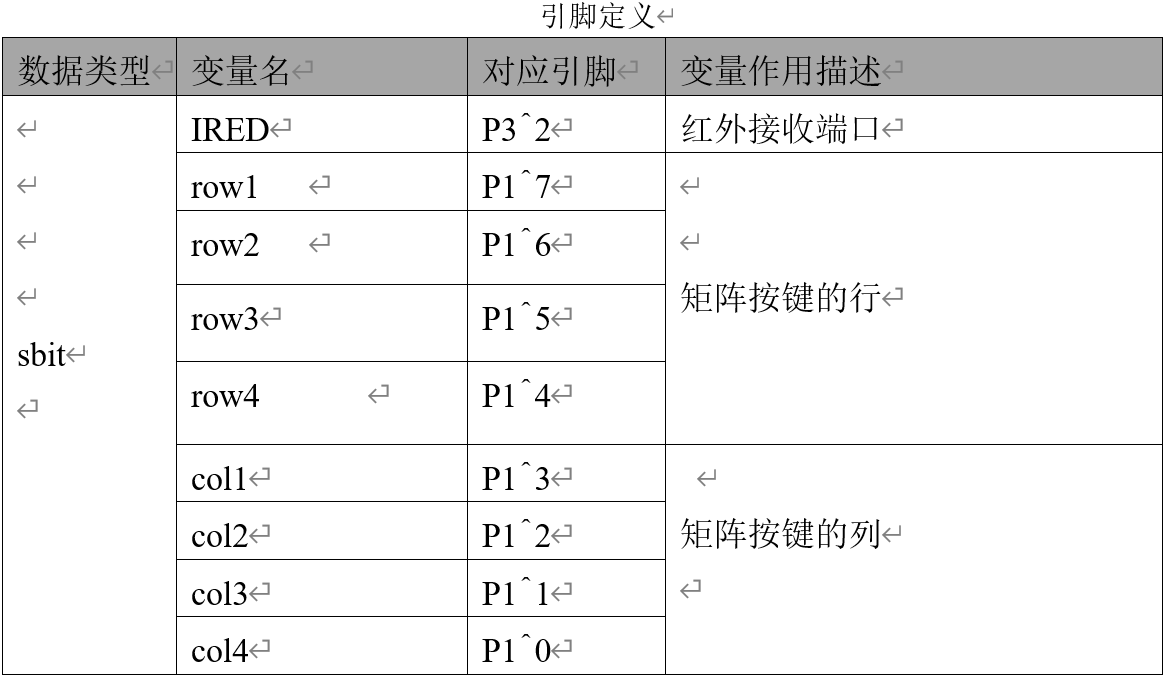






5.2程序变量定义







5.3主程序设计

洗衣机通电之后单片机上电首先进行程序的初始化包括定时器0、串口、lcd1602和红外接收的初始化以及各参数初值的设定。默认为“标准洗”模式，洗涤时间为60s。当启动键按下洗衣机从待命状态进入工作状态。当洗衣结束时控制蜂鸣器发声。

5.4延迟程序设计

本设计采用11.0592MHz的晶振

在程序中的不同情况需要延时的时间段长度不同，因此设计了三种不同时间基数的延时子函数,delay\_ms函数的单位是ms，delay\_100us函数的单位是100us，delay\_10us函数是延迟10us

*void* delay\_ms(*u16* *xms*)

{

*u16* i,j;

    for(i=0;i<*xms*;i++)

        for(j=0;j<110;j++);

}

*void* delay\_100us(*u16* *xus*)

{

*u16* i,j;

    for(i=0;i<*xus*;i++)

        for(j=0;j<11;j++);

}

*void* delay\_10us()

{

*u8* i=2;

    while(--i);

}

5.5 Lcd程序设计

Lcd显示功能在程序上实现了两种函数，lcd1602\_write\_string函数显示字符串，lcd1602\_write\_int函数显示整数

*void* lcd1602\_write\_string(*u8* *x*,*u8* *y*,*u8* \**s*)

{

    /\*

        写字符串

        x:列0-15

        y:行0-1

    \*/

*u8* i=0;

    if(lcd1602\_set\_cursor(*x*,*y*)==0)//设置光标位置

        return;

    while(\**s*!='\0')

    {

        lcd1602\_write\_data(\**s*);//逐个显示字符

        i++;

*s*++;

        if(i==16-*x*)

        {

            if(*y*==0)

                lcd1602\_write\_cmd(0xc0);//换到第二行

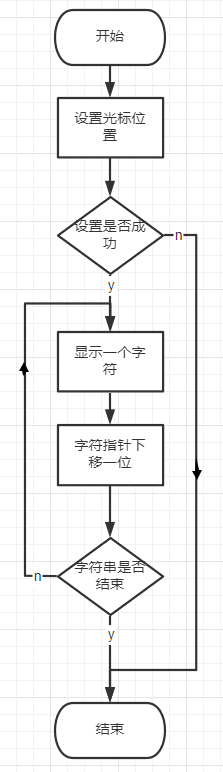
            else

                return;

        }

    }

}



*void* lcd1602\_write\_int( *u8* *x*, *u8* *y*,*int* *dat*,*u8* *len*)

{

    /\*

        写整数,可以是负数,最大值32767

        x:列0-15

        y:行0-1

        len:数字长度,最大5

     \*/

*u16* new\_dat=0;

*u8* i=0;

    if(lcd1602\_set\_cursor(*x*,*y*)==0)//设置光标位置

        return;

    if(*len*>5) *len*=5;

    if(*dat*>=0)

    {

        new\_dat=*dat*;

    }

    else

    {

        lcd1602\_write\_char(*x*,*y*,'-');//负数显示负号

        new\_dat=-*dat*;

*x*++;

    }

    for(i=*len*;i>0;i--)

    {

        if(*len*-i==16-*x*)

        {

            if(*y*==0)

                lcd1602\_write\_cmd(0xc0);//换到第二行

            else

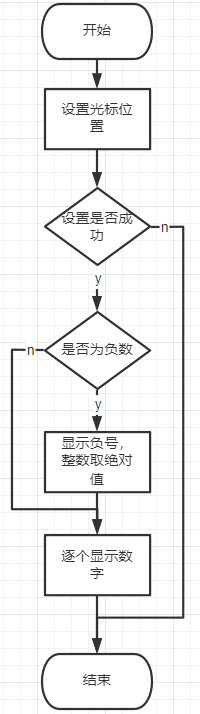
                return;

        }

        lcd1602\_write\_data(new\_dat/lcd1602\_pow(10,i-1)%10+'0');//逐个显示数字

    }

}



5.6按键扫描程序设计

按键扫描程序有两个函数，key\_independent\_scan函数扫描并返回独立按键，key\_matrix\_scan函数扫描并返回矩阵按键，两个函数需要在定时器中断服务中调用。

*u8* key\_independent\_scan()

{

    static *u8* key\_state=0;

*u8* i;

    if(key\_state==0)//上个时刻无按键被按下

    {

        for(i=0;i<4;i++)

        {

            if(key[i]==0)

            {

                key\_state=i+1;

                return NONE\_PRESS;

            }

        }

        return NONE\_PRESS;

    }

    //上个时刻有按键被按下

    if(key[key\_state-1]==0)//按键仍被按下

    {

        return NONE\_PRESS;

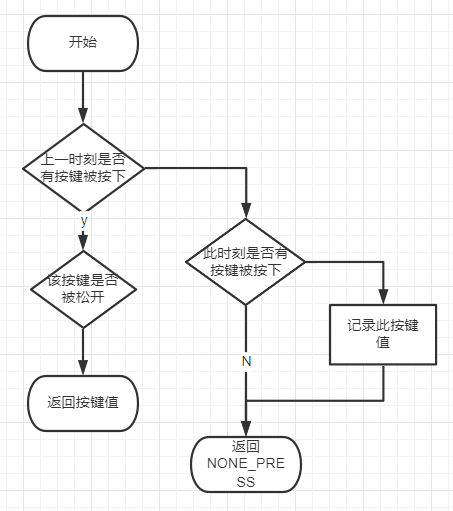
    }

    //按键被松开

    key\_state=0;

    return key\_state;

}



*u8* key\_matrix\_scan()

{

*u8* re;

    KEY\_MATRIX\_PORT=0x7f;//第一行低电平，其他行高电平

    re=key\_row\_scan(1);//扫描第一行

    if(re!=NONE\_PRESS)

    {

        return re;//返回按键号

    }

    KEY\_MATRIX\_PORT=0xbf;//第二行低电平，其他行高电平

    re=key\_row\_scan(2);

    if(re!=NONE\_PRESS)

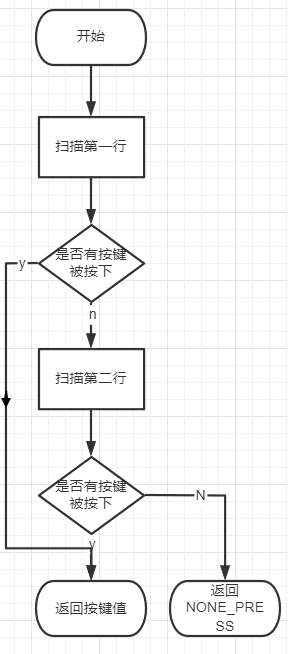
    {

        return re;

    }

    return NONE\_PRESS;

}



5.7洗涤程序设计

在洗涤的实现过程中，要涉及到对正反转步进电机的控制。一个洗涤的基本单元包括一次正转和一次反转，一次正转或一次反转时间为5s。

下面是步进电机控制函数stepper\_motor

*void* stepper\_motor(*u8* *dir*)

{

    /\*dir:转动方向，0表示顺时针，1表示逆时针\*/

    static *u8* i=0;

    STEPPER\_PORT &= 0xf0;

    STEPPER\_PORT |= step[i];

    if(*dir*==0)

        i++;

    else i--;

    if(i==8)

    {

        i=0;

    }

    else if(i==255)

    {

        i=7;

    }

}

下面是洗涤函数，通过定时器中断实现全局变量direct每秒增一，0到10循环，判断direct的范围控制步进电机的正反转

*void* start\_washing()

{

    while(start==1)

    {

        if(start==0)//按下停止按钮

            break;

        if(direct<5)

        {

            stepper\_motor(1);

        }

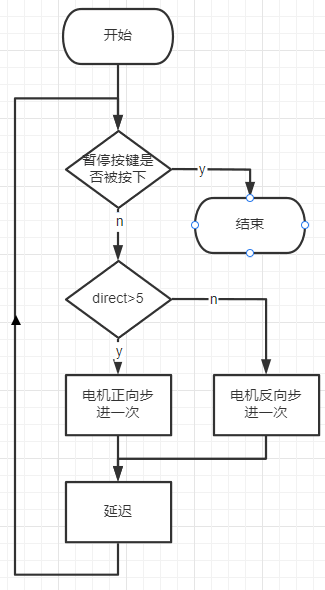
        else

            stepper\_motor(0);

        delay\_100us\_11M(speed[mode]);

    }

}



5.8报警程序设计

BEEP为蜂鸣器方波输入引脚

*void* buzzer\_play(*u16* *xms*, *u16* *freq*)

{

    /\*

        xms:持续时间ms

        freq:频率0-5000

    \*/

*u16* i;

    if(*freq*==0||*freq*>5000||*xms*==0)

        return;

    for(i=0;i<*xms*\*1000/*freq*;i++)

    {

        BEEP = !BEEP;

        delay\_100us\_11M(5000/*freq*);

        BEEP = !BEEP;

        delay\_100us\_11M(5000/*freq*);

    }

    BEEP = 0;

}

5.9串口通讯程序设计

下面是串口初始化函数，定时器1负责计时

*void* serial\_init(*bit* *flag*)//9600bps@11.0592MHz

{

    /\*

    串口初始化函数,flag=1,打开串口中断,即接收数据

    \*/

    if(*flag*==1)

    {

        ES=1;               //打开串口中断

        EA=1;               //打开总中断

    }

    SCON = 0x50;        //8位数据,可变波特率

    TMOD &= 0x0F;       //设置定时器模式

    TMOD |= 0x20;       //设置定时器模式

    TL1 = 0xFD;         //设置定时初始值

    TH1 = 0xFD;         //设置定时重载值

    ET1 = 0;            //禁止定时器中断

    TR1 = 1;            //定时器1开始计时

}

串口发送数据功能实现了三个函数，serial\_send\_char发送字符，serial\_send\_string发送字符串，serial\_send\_int发送整数。

*void* serial\_send\_char(*u8* *d*)

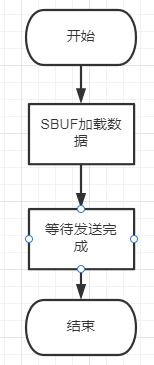
{

    SBUF=*d*;

    while(TI==0);//等待发送完成

    TI=0;

}



*void* serial\_send\_string(*u8*\* *s*)

{

    while(\**s*!='\0')

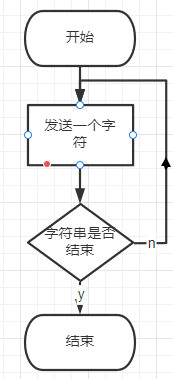
    {

        serial\_send\_char(\**s*);

*s*++;

    }

}



*void* serial\_send\_int(*int* *dat*,*u8* *len*)

{

    /\*

    发送一个整数,可以是负数,最大值32767

    len:数字长度,最大5

    \*/

*u16* new\_dat=0;

*u8* i;

    if(*len*>5) *len*=5;

    if(*dat*>=0)

    {

        new\_dat=*dat*;

    }

    else

    {

        serial\_send\_char('-');//负数发送负号

        new\_dat=-*dat*;

    }

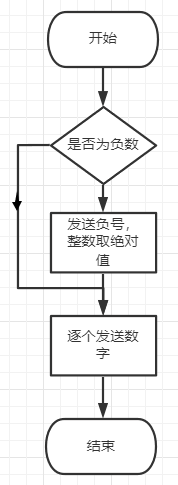
    for(i=*len*;i>0;i--)

    {

        serial\_send\_char(new\_dat/serial\_pow(10,i-1)%10+'0');//逐个发送数字

    }

}



接收数据通过串口中断函数实现，同时对数据进行处理，实现上位机控制下位机的功能

*void* serial\_interrupt() interrupt 4

{

    if(RI==1)//只处理接收数据

    {

        RI=0;

        dfs=SBUF;//获得数据

        check\_dfs();//处理数据

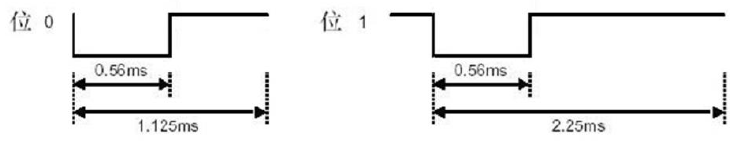
    }

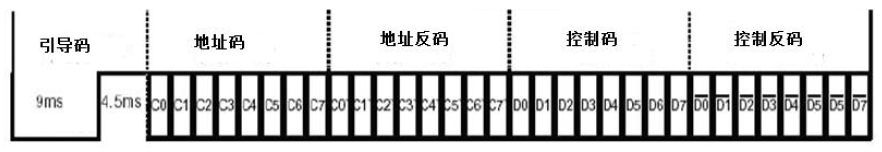
}

5.10红外接收程序设计

红外接收端口与单片机的INT0引脚连接，因此可以通过外部中断服务函数进行红外接收解码。

下面简述NEC协议





下面是红外接收初始化函数

*void* ired\_init(*void*)

{

    /\*

        初始化红外接收器，使用外部中断0

    \*/

    IT0=1;  //下降沿触发

    EX0=1;  //打开中断0

    EA=1;   //打开总中断

    IRED=1;

    PX0=1;//高优先级

}

下面是外部中断服务函数，其中cmd\_flag和addr\_flag分别为数据码标识位和地址码标识位，0表示未接受，1表示接收成功；IRED为红外接收端口，即P3^2；ired\_data为四个元素的字符数组，用于接收数据。

*void* ired\_int() interrupt 0

{

    /\*解码\*/

    u16 t=0,high\_time=0;

    u8 i,j;

    if(IRED==0)

    {

        t=1000;

        while(IRED==0&&t>0)//等待引导信号9ms低电平结束，若超过10ms则认为是干扰信号

        {

            delay\_10us();

            t--;

            if(t==0)

            {

                addr\_flag=0;

                cmd\_flag=0;

                return;

            }

        }

        t=500;

        while(IRED==1&&t>0)//等待引导信号4.5ms高电平结束，若超过5ms则认为是干扰信号

        {

            delay\_10us();

            t--;

            if(t==0)

            {

                addr\_flag=0;

                cmd\_flag=0;

                return;

            }

        }

        for(i=0;i<4;i++)//读取4个字节数据

        {

            for(j=0;j<8;j++)//读取8位数据

            {

                t=600;

                while(IRED==0&&t>0)//等待数据0/1前的0.56ms低电平结束，若超过1ms则认为是干扰信号

                {

                    delay\_10us();

                    t--;

                    if(t==0)

                    {

                        addr\_flag=0;

                        cmd\_flag=0;

                        return;

                    }

                }

                while(IRED==1)//计算高电平时间

                {

                    delay\_100us\_11M(1);

                    high\_time++;

                    if(high\_time>20)//若超过2ms则认为是干扰信号

                    {

                        addr\_flag=0;

                        cmd\_flag=0;

                        return;

                    }

                }

                ired\_data[i]>>=1;//右移一位

                if(high\_time>=8)//若高电平时间大于0.8ms，则认为是数据1

                    ired\_data[i]|=0x80;

                else ired\_data[i]&=0x7f;//否则是数据0

                high\_time=0;

            }

        }

        //检验数据是否正确

        if(ired\_data[0]!=~ired\_data[1] || ired\_data[2]!=~ired\_data[3])

        {

            //数据错误

            for(i=0;i<4;i++)

                ired\_data[i]=0;

            addr\_flag=0;

            cmd\_flag=0;

            return;

        }

        addr\_flag=1;

        cmd\_flag=1;//数据接收完成

    }

}

5.11水位报警程序设计

洗衣机在运行时须保证水位的正常，若出现低水位信号（按键模拟），则停止工作并开始添水，信号正常后继续工作，若添水时间过长，则报警；若出现高水位信号（按键模拟），则停止工作并开始排水，信号正常后继续工作，若排水时间过长，则报警。

u8 water\_interrupt()

{

    start=0;//停止洗衣机

    if(count==200)

        water\_time++;//每1s加1

    if(water\_time==0)

    {

        lcd1602\_clear();

        water\_time=1;

        if(water==LOW\_WATER)

        {

            //加水

            lcd1602\_write\_string(0,0,"adding water...");

        }

        else if(water==HIGH\_WATER)

        {

            //放水

            lcd1602\_write\_string(0,0,"watering...");

        }

    }

    if(water\_time%10==0)//10s后仍异常

    {

        lcd1602\_clear();

        lcd1602\_write\_string(0,0,"water erro!");

        if(water==LOW\_WATER)

        {

            //放水

            lcd1602\_write\_string(0,1,"low water!");

        }

        else if(water==HIGH\_WATER)

        {

            //加水

            lcd1602\_write\_string(0,1,"high water!");

        }

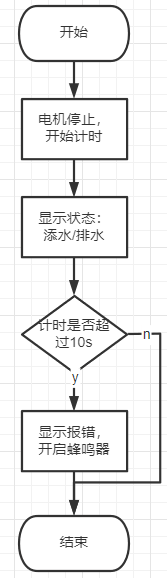
        //开启蜂鸣器

        return 1;

    }

    return 0;

}



5.12定时器中断程序设计

定时器0是程序运行的关键，它的计时周期是5ms，中断提供按键扫描、电机运行计时和处理红外数据等服务，保证程序按预期进行，不会被其它延迟函数或死循环影响。

下面是定时器0初始化函数

*void* init\_timer0()//5毫秒@11.0592MHz

{

    TMOD &= 0xF0; // 清除定时器0的工作方式

    TMOD = 0x01; // 选择定时器0工作在方式1

    TH0 = 0xee;TL0 = 0x00;//5ms

    EA = 1; // 开启总中断

    ET0 = 1; // 开启定时器0中断

    TR0 = 1; // 开启定时器0

}

下面是定时器0中断函数

*void* timer0\_interrupt() interrupt 1  //0表示外部中断0，1表示定时器0中断

//2表示外部中断1，3表示定时器1中断，4表示串口中断

{

    /\*5ms\*/

    static u8 re=0;//接收water\_interrupt()的返回值

    count++;

    TH0 = 0xee; // 定时器0的高8位

    TL0 = 0x00; // 定时器0的低8位

    //每10ms扫描一次按键

    if(count%2==0)

    {

        if(water==0)//水位正常

            independent\_keys();//独立按键扫描

            matrix\_keys();//矩阵按键扫描

    }

    if(water==0)

    {

        ired\_cmd=ired\_get\_cmd();//获得红外数据控制码

        if(ired\_cmd!=0)

        {

            ired();//处理红外数据

        }

    }

    if(water==0 && is\_water==1)

    {

        //水位正常，停止放/加水

        is\_water=0;

        water\_time=0;

        re=0;

        start=flag;//恢复洗衣机启动状态

        lcd1602\_clear();

        send\_data();

    }

    if(water!=0)

    {

        //水位异常，放/加水

        if(re!=1)

            re=water\_interrupt();

        else

            buzzer\_play\_11M(4,1000);

        is\_water=1;

    }

    if(count==200) count=0;

    if(start==1)

        run++;

    //每1s更新一次显示数据

    if(run == 200)

    {

        run = 0;

        direct++;

        if(direct==10)

            direct=0;

        if(time < 2)

        {

            // 关闭洗衣机

            start=0;

            time=0;

            flag=0;

            send\_data();

            buzzer\_play\_11M(500,1000);

            return;

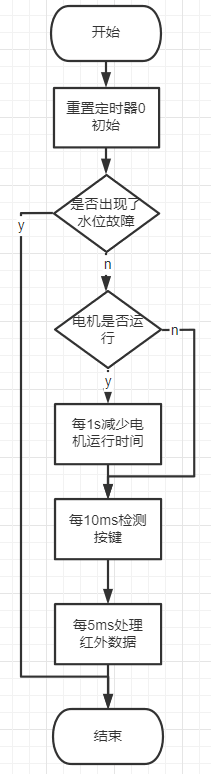
        }

        time--;

        send\_data();

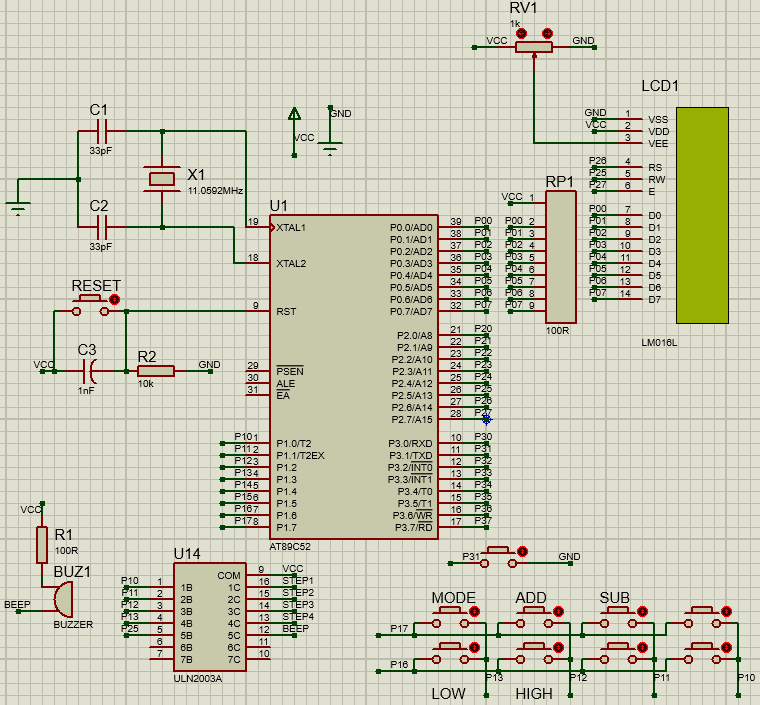
    }

}



六、硬件设计

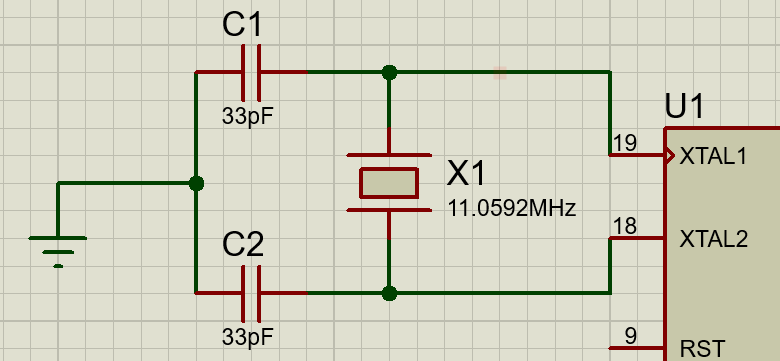
6.1全局电路图



6.2时钟电路

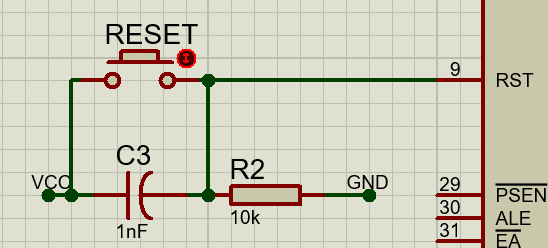
时钟电路是计算机的心脏，它控制着计算机的工作节奏。本次设计选用的单片机时钟频率为11.0592MHz。

STC89C52RC中有一个构成内部震荡器的高增益反向放大器，引脚XTAL1和XTAL2分别是该放大器的输入端和输出端。这个放大器与作为反馈元件的片外石英或陶瓷震荡器一起构成自激震荡器震荡电路。外接石英晶体（或陶瓷震荡器）及电容C1、C2接在放大器的震荡回路中构成并联震荡电路。本次设计两个电容都选取33pF。



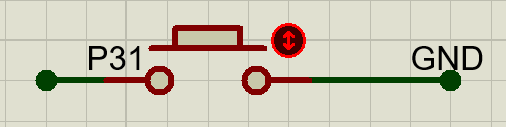
6.3复位电路

复位有两种方式，一种是上电复位，即开启电源后自动复位，另一种是按键复位。RESET按键按下后VCC直接与单片机RST引脚相连，松开后VCC与RST断开，RST被电阻拉为低电平，这个过程实现了复位功能；电源接通瞬间，电容相当于短路，RST为高电平，自动复位，电容充满电后，相当于开路，RST为低电平，程序开始运行。

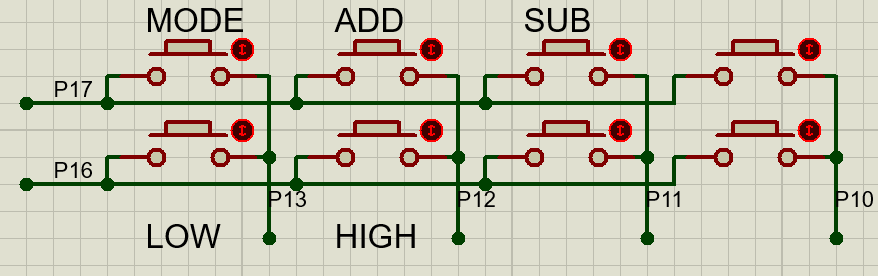


6.4按键电路

独立按键为开始/暂停按键

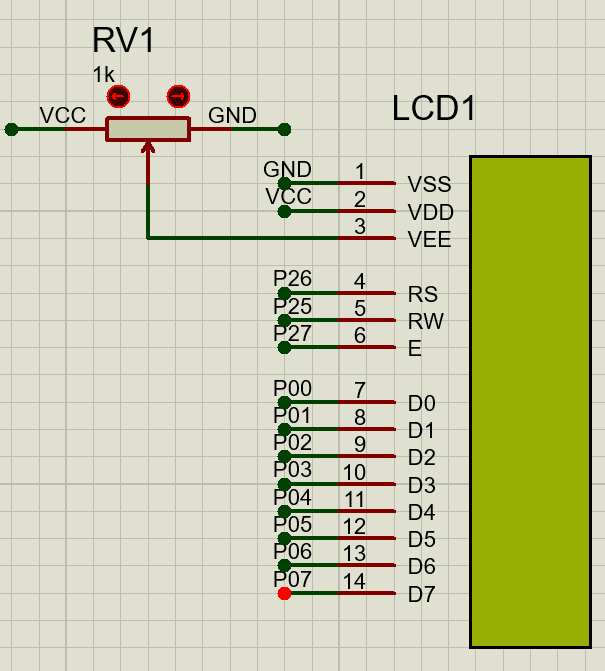


下图为矩阵按键，MODE按键切换洗涤模式，ADD按键添加时间，SUB按键减少时间，LOW按键模拟低水位信号，HIGH按键模拟高水位信号

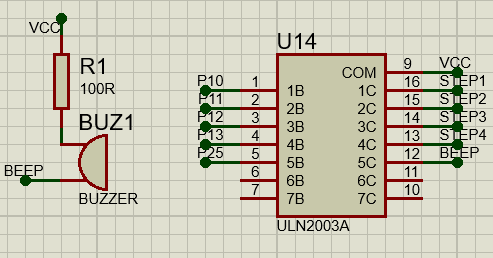


6.5显示电路

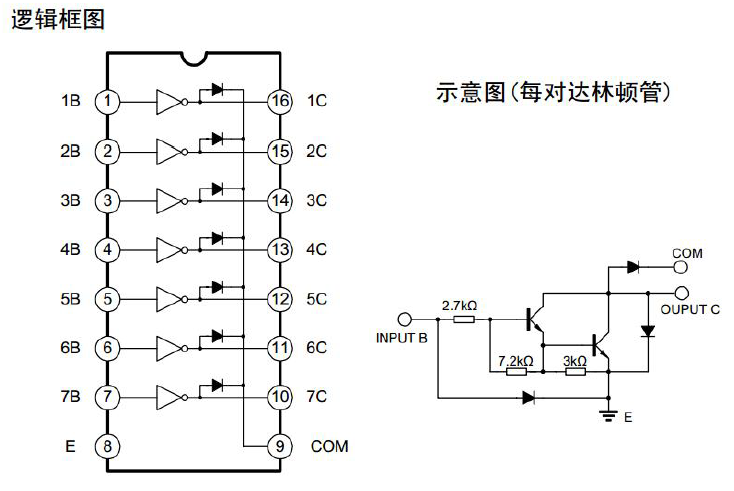
LCD1602的八位数据口DB0-DB7与单片机的P0.0-P0.7管脚连接，LCD1602的RS、RW、E脚分别与单片机的P2.6、P2.5、P2.7管脚连接。RV1是一个电位器，用来调节LCD1602对比度即显示亮度。



6.6蜂鸣器电路



由于单片机IO口驱动能力较弱，即使外加上拉电阻也无法驱动蜂鸣器运作，因此本次设计采用了ULN2003芯片。该芯片是一个单片高电压、高电流的达林顿晶体管阵列集成电路。其逻辑图与电路图如下



由此可知输入为低电平时，输出为高电平，反之。

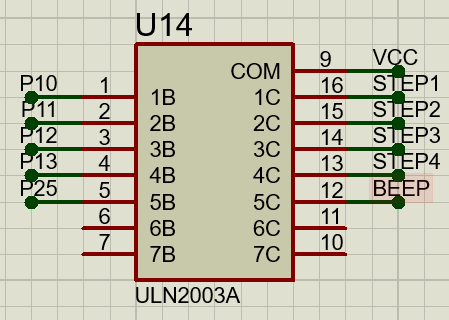
本次设计采用的是无源蜂鸣器，它需要一定频率的脉冲（高低电平）才会发

声，因此需要让P25 脚以一定频率不断输出高低电平信号才能控制蜂鸣器发出声

音。

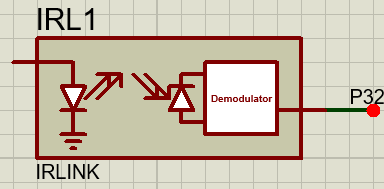
6.7电机驱动电路

电机驱动也是采用ULN2003芯片。控制口为P1.0-P1.3



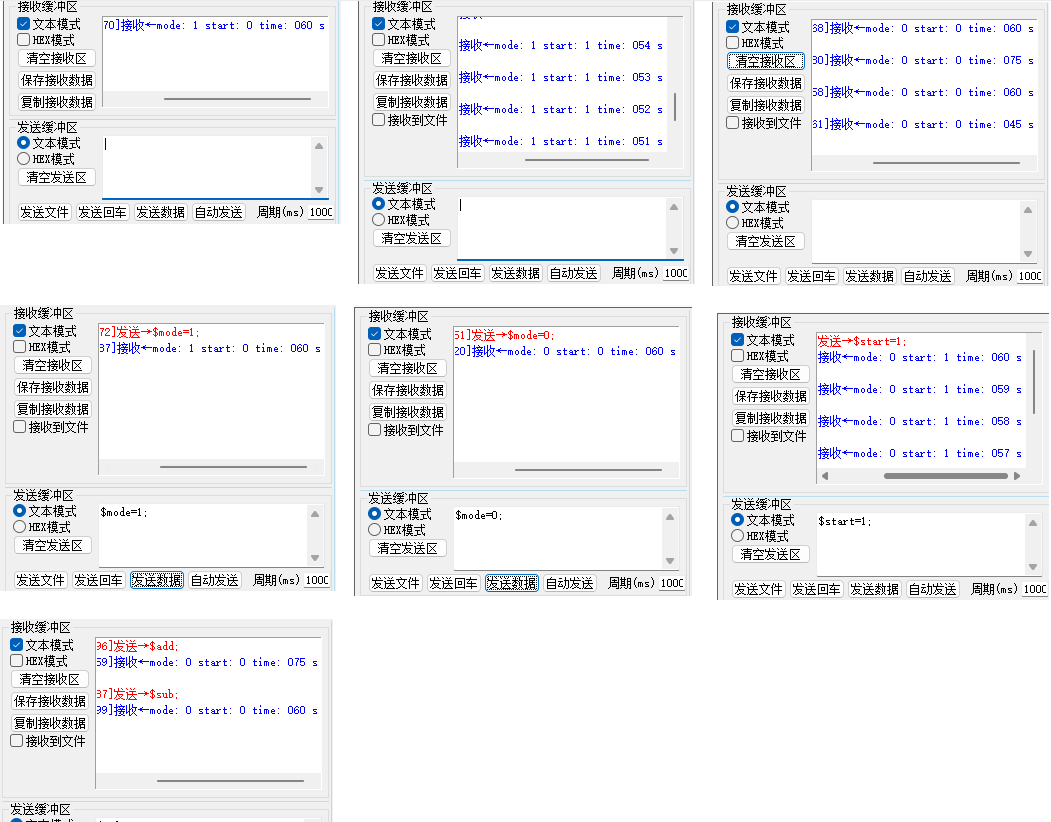
6.8红外接收器电路

由于红外接收头在没有脉冲的时候为高电平，当收到脉冲的时候为低电平，所以可以通过外部中断的下降沿触发中断，在中断内通过计算高电平时间，来判断接收的数据是0还是1.

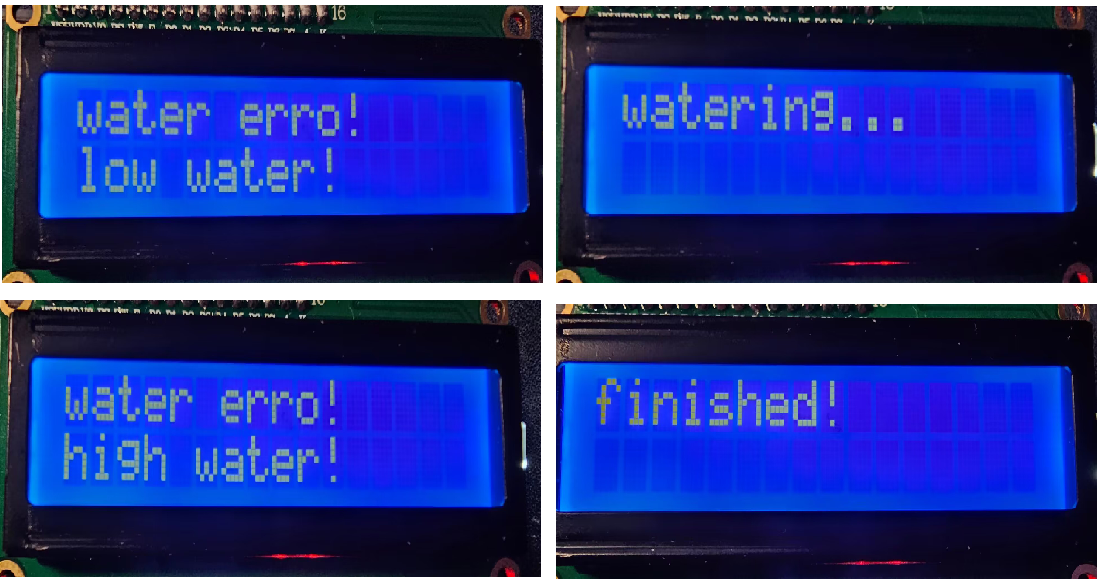


七、调试

7.1串口调试结果



7.2硬件调试结果

八、总结

单片机的广泛应用使得现在的电子产品设计越来越方便、功能越来越好、集成度也越来越高。通过对单片机的重新编程可以很方便的改变洗衣机的功能。单片机和微电子等技术是密切相关的，它们如何应用将直接对设计结果产生影响，应用不同的元器件和设计方法可以使洗衣机的设计结果截然不同，洗衣机的性能也将有差别，同时在本设计中应用的元器件都是单片机的应用，使该设计只要稍加修改就可以很方便地开发出其他基于单片机的洗衣机控制系统。

本次设计的洗衣机功能不够完善，全自动洗衣机发展趋势是不断向着高度自动化、智能化、健康化、节能、大容量、微型化、安全化等方向发展。本次设计没有考虑这些细节功能。在今后的课程设计中，要多联系所学专业课知识，同时要与时俱进，了解国内外的研究历程，这样才能完善自己的设计思路。