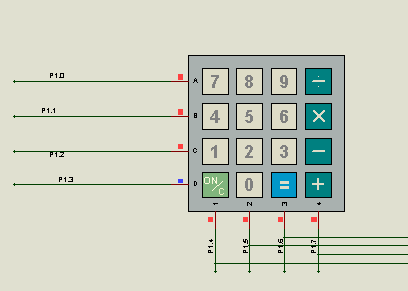
# 第5章 电子称的仿真实现

## 5.1对电子称模块的介绍

## 5.11 键盘输入模块



键盘输入模块主要完成向电子称输入单价。当用户使用电子称时，物品放到电子秤上时需要设置单价，这时显示界面会直接显示出这次的消费金额。

用4X4键盘输入，键盘对应名称如下：

1 2 3 A

4 5 6 B

7 8 9 C

\* 0 # D

下面为按键的功能介绍：

【0—9】为数字键

【\*】号键为无定义

【#】号键为小数点

【A】键为去皮

【B】键为清除单价

【C】键校准按键

【D】键校准按键

单价输入：输入数字，就可以输入单价，需要输入小数时，就按一下#键输入小数点，然后再输入一位小数。

单价的清除：当前的单价不需要时，就按一下B键将当前的单价数据清零。

去皮按键用法：放上需要去皮的物体，然后按一下A键，数值会显示0，就是把皮重去掉了，当不需要去皮时，就再按一下A按键，取消去皮。

下面介绍程序的具体实现方法：

按键函数的设计方法时通过中断处理函数来实现的，当用户按下按键时，就产生一个中断，程序跳转到中断处理函数中。开始扫描矩阵键盘的数值，最终把把按下的数值存入一个变量中去，供主函数调用。下面为具体的函数介绍：

void temer1() interrupt 3

{ uchar temp;

EX1=0;

P1=0xfe;

temp=P1;

temp=temp&0xf0;

if(temp!=0xf0)

{

delay(5);

temp=P1;

temp=temp&0xf0;

while(temp!=0xf0)

{

temp=P1;

switch(temp)

{

case 0xee:num=1;n=7;k=1,s++;break;

case 0xde:num=2;n=8;k=1,s++;break;

case 0xbe:num=3;n=9;k=1,s++;break;

case 0x7e:num=4;k='/',s++;break;

}

while(temp!=0xf0)

{

temp=P1;

temp=temp&0xf0;

}

//DisplayOneChar((s+6),1,table2[num-1]);

}

}

P1=0xfd;

temp=P1;

temp=temp&0xf0;

if(temp!=0xf0)

{

delay(5);

temp=P1;

temp=temp&0xf0;

while(temp!=0xf0)

{

temp=P1;

switch(temp)

{

case 0xed:num=5;n=4;k=1;s++;break;

case 0xdd:num=6;n=5;k=1;s++;break;

case 0xbd:num=7;n=6;k=1;s++;break;

case 0x7d:num=8;k='\*';s++;break;

}

while(temp!=0xf0)

{

temp=P1;

temp=temp&0xf0;

}

//DisplayOneChar(k+6,1,table2[num-1]);

}

}

P1=0xfb;

temp=P1;

temp=temp&0xf0;

if(temp!=0xf0)

{

delay(5);

temp=P1;

temp=temp&0xf0;

while(temp!=0xf0)

{

temp=P1;

switch(temp)

{

case 0xeb:num=9;n=1;k=1;s++;break;

case 0xdb:num=10;n=2;k=1;s++;break;

case 0xbb:num=11;n=3;k=1;s++;break;

case 0x7b:num=12;k='-';s++;break;

}

while(temp!=0xf0)

{

temp=P1;

temp=temp&0xf0;

}

// DisplayOneChar(k+6,1,table2[num-1]);

}

}

P1=0xf7;

temp=P1;

temp=temp&0xf0;

if(temp!=0xf0)

{

delay(5);

temp=P1;

temp=temp&0xf0;

while(temp!=0xf0)

{

temp=P1;

switch(temp)

{

case 0xe7:num=13;k=' ';break;

case 0xd7:num=14;n=0;k=1;s++;;break;

case 0xb7:num=15;k='=';s++;break;

case 0x77:num=16;k='+';s++;;break;

}

while(temp!=0xf0)

{

temp=P1;

temp=temp&0xf0;

}

//DisplayOneChar(k+6,1,table2[num-1]);

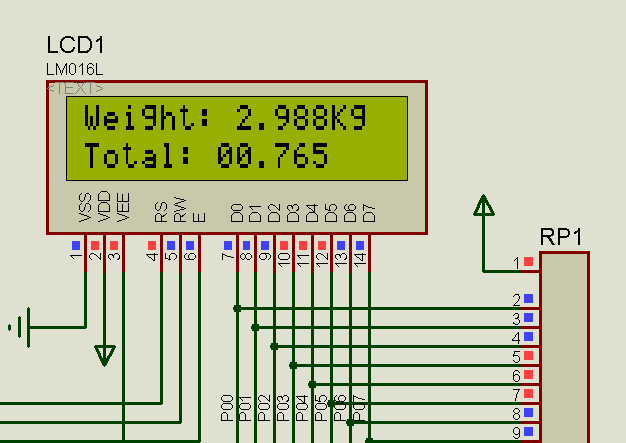
}

}

EX1=1;

}

## 5.12液晶显示模块



液晶显示模块的功能时显示物体的重量。当物体放在电子称上之后，通过键盘输入的单片机，在液晶的第二行显示这个物品的总价格。

下面介绍液晶程序的具体实现方法：

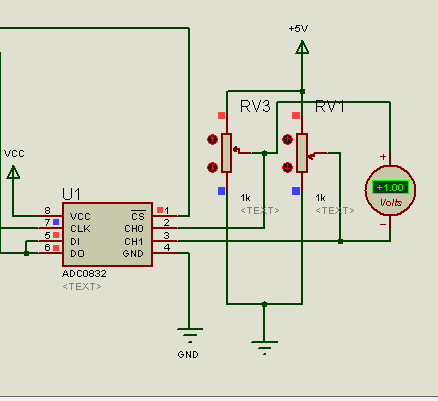
本次设计使用的是1602液晶，通过对液晶进行写命令和读命令实现液晶的显示效果。首先要对液晶进行初始换任务，在液晶初始化完成之后，就可以对液晶进行数据以及位置的写入操作，下面为具体的函数介绍：

void initLCM( )；此函数为液晶的初始化函数，通过这个函数设置液晶的显示模式，把液晶设置为开启状态，在显示一个数据后光标自动移动。

void DisplayListChar(uchar X,uchar Y,uchar code \*DData)；该函数的作用是显示指定坐标的一串字符子函数。

void DisplayOneChar(uchar X,uchar Y,uchar DData)；该函数的作用是显示指定坐标的一个字符子函数。

## 5.13 A/D转换模块



A/D显示模块的主要功能是模拟电压到数字量的转换。单片机通过A/D转换的数字量来计算出这个物体的重量，A/D输入的是一个电压值，物体越重转化的电压值越大，通过这种方法来测量物体的重量。

下面介绍A/D程序的具体实现方法：

在使用A/D芯片之前要对A/D进行初始化任务，确定使用A/D转换的哪一个通道，以及转换电压值得精度大小和A/D的装换时间的设置。下面为具体的函数的实现：

uchar Adc0832(unsigned char channel) //AD转换，返回结果

{

uchar i=0;

uchar j;

uint dat=0;

uchar ndat=0;

if(channel==0)channel=2;

if(channel==1)channel=3;

ADDI=1;

\_nop\_();

\_nop\_();

ADCS=0;//拉低CS端

\_nop\_();

\_nop\_();

ADCLK=1;//拉高CLK端

\_nop\_();

\_nop\_();

ADCLK=0;//拉低CLK端,形成下降沿1

\_nop\_();

\_nop\_();

ADCLK=1;//拉高CLK端

ADDI=channel&0x1;

\_nop\_();

\_nop\_();

ADCLK=0;//拉低CLK端,形成下降沿2

\_nop\_();

\_nop\_();

ADCLK=1;//拉高CLK端

ADDI=(channel>>1)&0x1;

\_nop\_();

\_nop\_();

ADCLK=0;//拉低CLK端,形成下降沿3

ADDI=1;//控制命令结束

\_nop\_();

\_nop\_();

dat=0;

for(i=0;i<8;i++)

{

dat|=ADDO;//收数据

ADCLK=1;

\_nop\_();

\_nop\_();

ADCLK=0;//形成一次时钟脉冲

\_nop\_();

\_nop\_();

dat<<=1;

if(i==7)dat|=ADDO;

}

for(i=0;i<8;i++)

{

j=0;

j=j|ADDO;//收数据

ADCLK=1;

\_nop\_();

\_nop\_();

ADCLK=0;//形成一次时钟脉冲

\_nop\_();

\_nop\_();

j=j<<7;

ndat=ndat|j;

if(i<7)ndat>>=1;

}

ADCS=1;//拉低CS端

ADCLK=0;//拉低CLK端

ADDO=1;//拉高数据端,回到初始状态

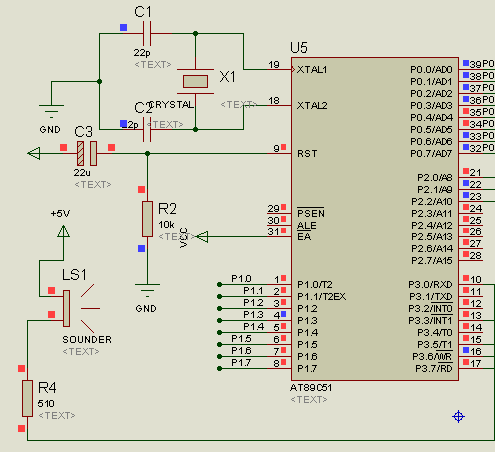
dat<<=8;

dat|=ndat;

return(dat); //return ad k

}

## 5.14 单片机最小系统和报警模块



单片机最小系统的作用是完成程序运行的基本功能。通过单片机的控制作用，完成数据的装换。

由于单片机最小系统是为程序提供运行的一个环境，它本身不需要程序驱动，下面只介绍一下报警程序的实现：

void alarm(void)

{

if(ad\_data>=256)

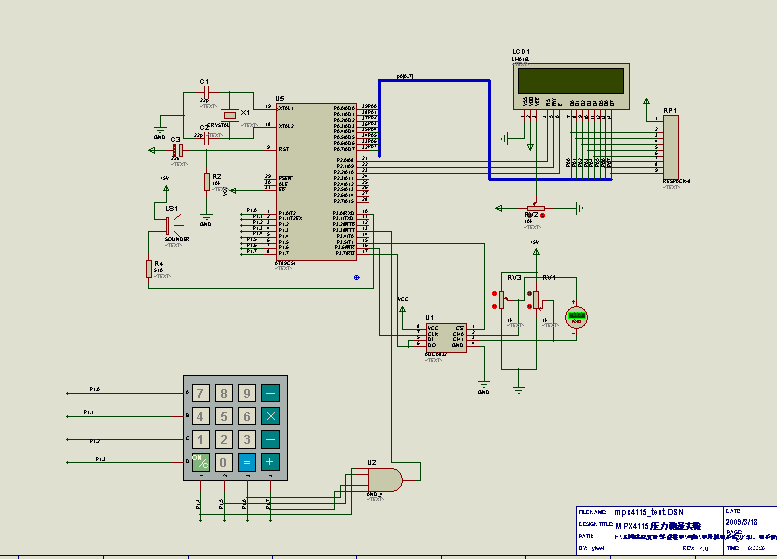
beep=0; //则启动报警

else

beep=1;

}

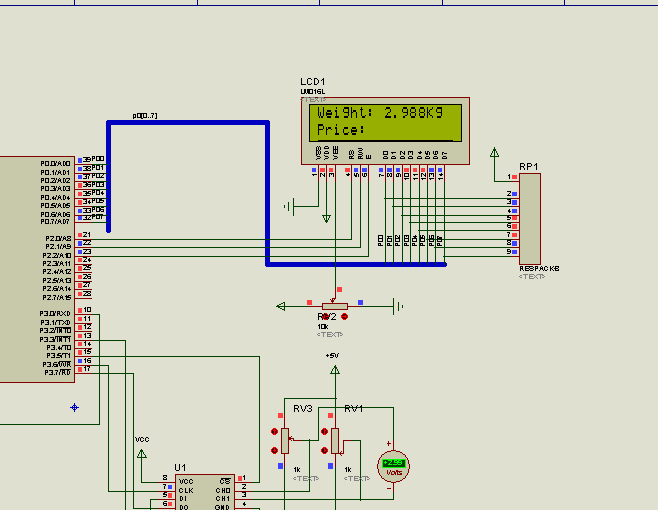
## 5.15 整体仿真电路图



上图是整体的电路图连接。通过对各个模块的连接，达到实现电子称功能的目的。从上图中可以看出本次设计的详细电路连接过程，以及模块之间和模块与单片机之间的连接关系。从而可以清晰的看出，电子称硬件的详细实现过程。

## 5.2对电子称各个功能的介绍

## 5.21 电子称的称重功能



上图所示为电子称称重的显示界面。本次设计的思路是通过采集物体的重量所产生的模拟电压量值来获取物体的重量参数。当物体放在电子称上时，电子称是一个可以感受压力的电阻，当压力变化时，电阻的两端电压值也就发生变化。通过A/D芯片采集电阻两边的电压值，通过这个电压值乘以一个称重的常数，就可以测得本次物体的称重的重量。

下面为具体的程序实现方法：

if(k=='\*')

{

data\_pro();

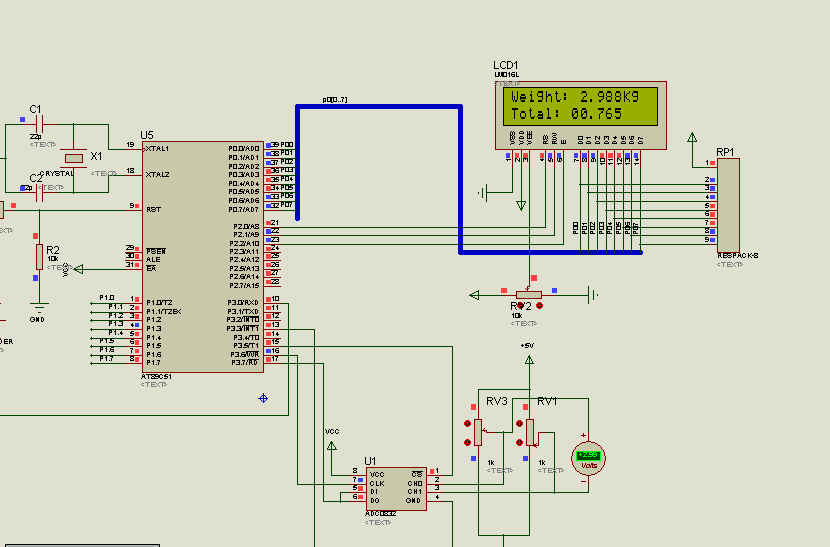
WriteCommandLCM(0x01,1);

weishu(z1);

k=0;

}

## 5.22 电子称的价格计算功能



上图所示为电子称物体价格的显示界面。本次设计的思路是同过A/D采集到的电压值的大小，乘以一个物体重量的常数的到物体的具体质量。通过这个质量的变量再乘以用户输入该物体单价的数值，处理机通过数值的计算，会在液晶显示屏上打印出该物体的总价钱。

下面为具体的程序实现方法：

if(k=='=')

{

z1=z1\*temp1;

WriteCommandLCM(0x01,1);

DisplayListChar(0,1,str3);

s=0;

weishu1(765);

k=0;

}

# 参考文献：

1任治刚．《电子信息工程专业英语教程》[M]北京.电子工业出版社.2006．129-15”

2 张万奎．《模拟电子技术》[M]．湖南．湖南大学出版社，2005：226-229．

3 崔建明编．《电工电子EDA技术》[M]．西安．高等教育出版社，2006：23-26．”

4 孙志勇，赵砚江．《数控与电控技术》[M]．北京：机械工业出版社，2002：178-200．”

5 潘新民，王燕芳．《微型计数器机控制技术》[M]．北京.电子工业出版社.2006，”

6 张克农．《数字电子技术基础》[M]．西安．高等教育出版社.2005：145．

7 席德勋．现代电子技术[D]．北京．高等教育出版社，2002：121-125．

8 谢运祥．可编程逻辑器件的发展及其应用前景[R].西安．高等教育出版社，2002

9 沈德金，陈粤初．MCS-51系列单片机接口电路与应用程序实例．北京航空航天大学出版社,1990

10 胡汉才.单片机原理及接口技术.清华大学出版社,1996

11 何立民．MCS-5系列单片机应用系统设计．北京航空航天大学出版社，1990．

12 张毅刚、彭喜元、姜守达、乔立岩．新编MCS-5系列单片机应用设计．哈尔滨工业大学出版社，2003.6

13 V. Yu. Teplov,A. V. Anisimov. Thermostatting System Using a Single-Chip Microcomputer and Thermoelectric Modules Based on the Peltier Effect[J] ,2002   
 14 Yeager Brent.How to troubleshoot your electronic scale[J].. Powder and Bulk Engineering. 1995   
 15 Meehan Joanne,Muir Lindsey.SCM in Merseyside SMEs:Benefits and barriers[J].. TQM Journal. 2008

# 附录：

#include<reg51.h>

#include<intrins.h>

#include <absacc.h>

#include <math.h>

#define uchar unsigned char

#define uint unsigned int

#define BUSY 0x80 //常量定义

#define DATAPORT P0

sbit ADCS =P3^5;

sbit ADDI =P3^7;

sbit ADDO =P3^7;

sbit ADCLK =P3^6;

sbit LCM\_RS=P2^0;

sbit LCM\_RW=P2^1;

sbit LCM\_EN=P2^2;

uint x1,y1,z1=0,w1,temp1;

uchar ad\_data,k,n,m,e,num,s; //采样值存储

sbit beep =P3^0;

char press\_data; //标度变换存储单元

unsigned char ad\_alarm; //报警值存储单元

unsigned char press\_ge=0; //显示值百位

unsigned char press\_shifen=0; //显示值十位

unsigned char press\_baifen=0; //显示值个位

unsigned char press\_qianfen=0; //显示值十分位

uchar code str0[]={"Weight: . Kg "};

uchar code str2[]={"Price: "};

uchar code str3[]={"Total: "};

uchar code table2[]={0x37,0x38,0x39,0xfd,0x34,0x35,0x36,0x78,0x31,0x32,0x33,0x2d,0x3d,0x30,0x2e,0x2b}; //键盘码

void delay(uint);

void lcd\_wait(void);

void delay\_LCM(uint); //LCD延时子程序

void initLCM( void); //LCD初始化子程序

void lcd\_wait(void); //LCD检测忙子程序

void WriteCommandLCM(uchar WCLCM,uchar BusyC); //写指令到ICM子函数

void WriteDataLCM(uchar WDLCM); //写数据到LCM子函数

void DisplayOneChar(uchar X,uchar Y,uchar DData); //显示指定坐标的一个字符子函数

void DisplayListChar(uchar X,uchar Y,uchar code \*DData); //显示指定坐标的一串字符子函数

void weishu(uint m);

void weishu1(uint m);

void display(void);

uchar Adc0832(unsigned char channel);

void alarm(void);

void data\_pro(void);

/\*\*\*\*\*\*\*\*\*\*main funcation\*\*\*\*\*\*\*\*\*\*\*\*/

void main(void)

{

delay(500); //系统延时500ms启动

//ad\_data=0; //采样值存储单元初始化为0

initLCM( );

WriteCommandLCM(0x01,1); //清显示屏

DisplayListChar(0,0,str0);

DisplayListChar(0,1,str2);

while(1)

{

ad\_data =Adc0832(0); //采样值存储单元初始化为0

alarm();

data\_pro();

display();

if(k==1)

{

DisplayOneChar((s+7),1,table2[num-1]);

x1=m;

y1=n;

y1=y1\*10+x1;

}

if(k=='\*')

{

data\_pro();

WriteCommandLCM(0x01,1);

weishu(z1);

k=0;

}

if(k=='=')

{

z1=z1\*temp1;

WriteCommandLCM(0x01,1);

DisplayListChar(0,1,str3);

s=0;

weishu1(765);

k=0;

}

if(k==' ')

{

WriteCommandLCM(0x80+0x40,1);

WriteCommandLCM(0x01,1);

z1=0;

s=0; //防止清零时指针后移动

}

}

}

/\*\*\*\*\*\*\*\*\*延时K\*1ms,12.000mhz\*\*\*\*\*\*\*\*\*\*/

void delay(uint k)

{

uint i,j;

for(i=0;i<k;i++)

for(j=0;j<100;j++);

}

/\*\*\*\*\*\*\*\*\*\*写指令到ICM子函数\*\*\*\*\*\*\*\*\*\*\*\*/

void WriteCommandLCM(uchar WCLCM,uchar BusyC)

{

if(BusyC)

lcd\_wait();

DATAPORT=WCLCM;

LCM\_RS=0; // 选中指令寄存器

LCM\_RW=0;

LCM\_RW=0; // 写模式

LCM\_EN=1;

\_nop\_();

\_nop\_();

\_nop\_();

\_nop\_();

\_nop\_();

\_nop\_();

\_nop\_();

LCM\_EN=0;

}

/\*\*\*\*\*\*\*\*\*\*写数据到LCM子函数\*\*\*\*\*\*\*\*\*\*\*\*/

void WriteDataLCM(uchar WDLCM)

{

lcd\_wait( ); //检测忙信号

DATAPORT=WDLCM;

LCM\_RS=1; // 选中数据寄存器

LCM\_RW=0; // 写模式

LCM\_EN=1;

\_nop\_();

\_nop\_();

\_nop\_();

\_nop\_();

\_nop\_();

\_nop\_();

\_nop\_();

\_nop\_();

LCM\_EN=0;

}

/\*\*\*\*\*\*\*\*\*\*\*lcm内部等待函数\*\*\*\*\*\*\*\*\*\*\*\*\*/

void lcd\_wait(void)

{

DATAPORT=0xff; //读LCD前若单片机输出低电平,而读出LCD为高电平,则冲突,Proteus仿真会有显示逻辑黄色

LCM\_EN=1;

LCM\_RS=0;

LCM\_RW=0;

LCM\_RW=1;

\_nop\_();

\_nop\_();

\_nop\_();

\_nop\_();

\_nop\_();

\_nop\_();

\_nop\_();

while(DATAPORT&BUSY)

{ LCM\_EN=0;

\_nop\_();

\_nop\_();

\_nop\_();

\_nop\_();

\_nop\_();

\_nop\_();

\_nop\_();

\_nop\_();

LCM\_EN=1;

\_nop\_();

\_nop\_();

\_nop\_();

\_nop\_();

\_nop\_();

\_nop\_();

\_nop\_();

}

LCM\_EN=0;

}

/\*\*\*\*\*\*\*\*\*\*LCM初始化子函数\*\*\*\*\*\*\*\*\*\*\*/

void initLCM( )

{

LCM\_EN=0;

DATAPORT=0;

delay(15);

WriteCommandLCM(0x38,0); //三次显示模式设置，不检测忙信号

delay(5);

WriteCommandLCM(0x38,0);

delay(5);

WriteCommandLCM(0x38,0);

delay(5);

WriteCommandLCM(0x38,1); //8bit数据传送，2行显示，5\*7字型，检测忙信号

WriteCommandLCM(0x08,1); //关闭显示，检测忙信号

WriteCommandLCM(0x01,1); //清屏，检测忙信号

WriteCommandLCM(0x06,1); //显示光标右移设置，检测忙信号

WriteCommandLCM(0x0c,1); //显示屏打开，光标不显示，不闪烁，检测忙信号

TMOD=0x11;

EA=1;

ET1=1;

TR1=1;

k=0;

x1=0;

y1=0;

z1=0;

}

/\*\*\*\*显示指定坐标的一个字符子函数\*\*\*\*/

void DisplayOneChar(uchar X,uchar Y,uchar DData)

{

Y&=0x01;

X&=0x0f;

if(Y)

X|=0x40; //若y为1（显示第二行），地址码+0X40

X|=0x80; //指令码为地址码+0X80

WriteCommandLCM(X,1);

WriteDataLCM(DData);

}

/\*\*\*\*\*\*\*显示指定坐标的一串字符子函数\*\*\*\*\*/

void DisplayListChar(uchar X,uchar Y,uchar code \*DData)

{

uchar ListLength=0;

Y&=0x01;

X&=0x0f;

while(X<16)

{

DisplayOneChar(X,Y,DData[ListLength]);

ListLength++;

X++;

}

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*系统显示子函数\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

void display(void)

{

WriteCommandLCM(0x0c,1); //显示屏打开，光标不显示，不闪烁，检测忙信号

DisplayListChar(0,0,str0);

//DisplayListChar(0,1,str2);

DisplayOneChar(8,0,press\_ge+0x30);

DisplayOneChar(10,0,press\_shifen+0x30);

DisplayOneChar(11,0,press\_baifen+0x30);

DisplayOneChar(12,0,press\_qianfen+0x30);

delay(1000); //稳定显示

}

/\*\*\*\*\*\*\*\*\*\*\*\*

读ADC0832函数

\*\*\*\*\*\*\*\*\*\*\*\*/

//采集并返回

uchar Adc0832(unsigned char channel) //AD转换，返回结果

{

uchar i=0;

uchar j;

uint dat=0;

uchar ndat=0;

if(channel==0)channel=2;

if(channel==1)channel=3;

ADDI=1;

\_nop\_();

\_nop\_();

ADCS=0;//拉低CS端

\_nop\_();

\_nop\_();

ADCLK=1;//拉高CLK端

\_nop\_();

\_nop\_();

ADCLK=0;//拉低CLK端,形成下降沿1

\_nop\_();

\_nop\_();

ADCLK=1;//拉高CLK端

ADDI=channel&0x1;

\_nop\_();

\_nop\_();

ADCLK=0;//拉低CLK端,形成下降沿2

\_nop\_();

\_nop\_();

ADCLK=1;//拉高CLK端

ADDI=(channel>>1)&0x1;

\_nop\_();

\_nop\_();

ADCLK=0;//拉低CLK端,形成下降沿3

ADDI=1;//控制命令结束

\_nop\_();

\_nop\_();

dat=0;

for(i=0;i<8;i++)

{

dat|=ADDO;//收数据

ADCLK=1;

\_nop\_();

\_nop\_();

ADCLK=0;//形成一次时钟脉冲

\_nop\_();

\_nop\_();

dat<<=1;

if(i==7)dat|=ADDO;

}

for(i=0;i<8;i++)

{

j=0;

j=j|ADDO;//收数据

ADCLK=1;

\_nop\_();

\_nop\_();

ADCLK=0;//形成一次时钟脉冲

\_nop\_();

\_nop\_();

j=j<<7;

ndat=ndat|j;

if(i<7)ndat>>=1;

}

ADCS=1;//拉低CS端

ADCLK=0;//拉低CLK端

ADDO=1;//拉高数据端,回到初始状态

dat<<=8;

dat|=ndat;

return(dat); //return ad k

}

void data\_pro(void)

{

unsigned int;

float press;

if(0<ad\_data<256)

{

int vary=ad\_data;

press=(0.019531\*vary);

temp1=(int)(press\*1000); //放大1000倍，便于后面的计算

press\_ge=temp1/1000; //取压力值百位

press\_shifen=(temp1%1000)/100; //取压力值十位

press\_baifen=((temp1%1000)%100)/10; //取压力值个位

press\_qianfen=((temp1%1000)%100)%10; //取压力值十分位

}

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*报警子函数\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

void alarm(void)

{

if(ad\_data>=256)

beep=0; //则启动报警

else

beep=1;

}

void weishu(uint m)

{

uchar wei4,wei3,wei2,wei1,wei0;

wei4=m/10000;

wei3=m%10000/1000;

wei2=m%1000/100;

wei1=m%100/10;

wei0=m%10;

DisplayOneChar(7,1,0x30+wei4);

DisplayOneChar(8,1,0x30+wei3);

DisplayOneChar(10,1,0x30+wei2);

//DisplayOneChar(10,1,'.');

DisplayOneChar(11,1,0x30+wei1);

DisplayOneChar(12,1,0x30+wei0);

}

void weishu1(uint m)

{

uchar wei5,wei4,wei3,wei2,wei1,wei0;

wei5=m/100000;

wei4=m%100000/10000;

wei3=m%10000/1000;

wei2=m%1000/100;

wei1=m%100/10;

wei0=m%10;

DisplayOneChar(7,1,0x30+wei4);

DisplayOneChar(8,1,0x30+wei3);

DisplayOneChar(9,1,'.');

DisplayOneChar(10,1,0x30+wei2);

//DisplayOneChar(10,1,'.');

DisplayOneChar(11,1,0x30+wei1);

DisplayOneChar(12,1,0x30+wei0);

}

void temer1() interrupt 3

{ uchar temp;

EX1=0;

P1=0xfe;

temp=P1;

temp=temp&0xf0;

if(temp!=0xf0)

{

delay(5);

temp=P1;

temp=temp&0xf0;

while(temp!=0xf0)

{

temp=P1;

switch(temp)

{

case 0xee:num=1;n=7;k=1,s++;break;

case 0xde:num=2;n=8;k=1,s++;break;

case 0xbe:num=3;n=9;k=1,s++;break;

case 0x7e:num=4;k='/',s++;break;

}

while(temp!=0xf0)

{

temp=P1;

temp=temp&0xf0;

}

//DisplayOneChar((s+6),1,table2[num-1]);

}

}

P1=0xfd;

temp=P1;

temp=temp&0xf0;

if(temp!=0xf0)

{

delay(5);

temp=P1;

temp=temp&0xf0;

while(temp!=0xf0)

{

temp=P1;

switch(temp)

{

case 0xed:num=5;n=4;k=1;s++;break;

case 0xdd:num=6;n=5;k=1;s++;break;

case 0xbd:num=7;n=6;k=1;s++;break;

case 0x7d:num=8;k='\*';s++;break;

}

while(temp!=0xf0)

{

temp=P1;

temp=temp&0xf0;

}

//DisplayOneChar(k+6,1,table2[num-1]);

}

}

P1=0xfb;

temp=P1;

temp=temp&0xf0;

if(temp!=0xf0)

{

delay(5);

temp=P1;

temp=temp&0xf0;

while(temp!=0xf0)

{

temp=P1;

switch(temp)

{

case 0xeb:num=9;n=1;k=1;s++;break;

case 0xdb:num=10;n=2;k=1;s++;break;

case 0xbb:num=11;n=3;k=1;s++;break;

case 0x7b:num=12;k='-';s++;break;

}

while(temp!=0xf0)

{

temp=P1;

temp=temp&0xf0;

}

// DisplayOneChar(k+6,1,table2[num-1]);

}

}

P1=0xf7;

temp=P1;

temp=temp&0xf0;

if(temp!=0xf0)

{

delay(5);

temp=P1;

temp=temp&0xf0;

while(temp!=0xf0)

{

temp=P1;

switch(temp)

{

case 0xe7:num=13;k=' ';break;

case 0xd7:num=14;n=0;k=1;s++;;break;

case 0xb7:num=15;k='=';s++;break;

case 0x77:num=16;k='+';s++;;break;

}

while(temp!=0xf0)

{

temp=P1;

temp=temp&0xf0;

}

//DisplayOneChar(k+6,1,table2[num-1]);

}

}

EX1=1;

}