Xxxx学院

单片机原理及应用 实验报告

|  |  |
| --- | --- |
| 成绩 |  |

|  |  |  |  |
| --- | --- | --- | --- |
| 专业班级 |  | 实验日期 |  |
| 姓 名 |  | 学 号 |  |
| 实验名称 | AD应用-简易电压表 | 指导教师 |  |

（报告内容包括实验目的、实验设备及器材、实验内容和要求、实验步骤、实验小结等）

**一、实验目的**

1、熟悉模拟电压转化成数字的工作原理；

2、熟悉LCD1602的常用指令和编程方法；

3、熟悉ADC0804的使用和程序编写方法。

**二、实验内容及原理**

1、ADC0804芯片

ADC0804是一个8位CMOS型逐次比较式A/D转换器，具有三态锁存输出功能，最短转换时间为100us，其引脚图如下：

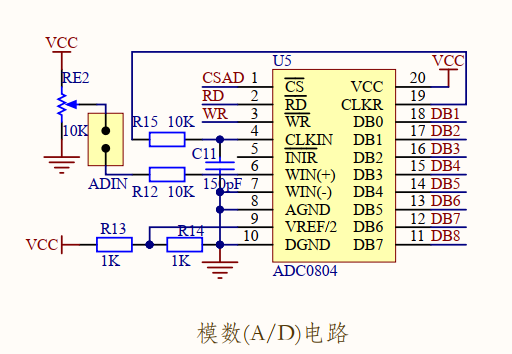


图1 ADC8040引脚图

① 芯片参数：

工作电压：+5V，即VCC=+5V。

模拟转换电压范围：0～+5V，即0≤Vin≤+5V。

分辨率：8位，即分辨率为1/28=1/256，转换值介于0～255之间。

转换时间：100us（fCK = 640KHz时）。

转换误差：±1LSB。

1/2参考电压：2.5V，即Vref = 5V。

② 各个引脚名称及作用：

Vin（+）、Vin（-）：两个模拟信号输入端，可以接收单极性、双极性和差模输入信号。

DB0-DB7：具有三态特性数字信号输出端，输出结果为八位二进制结果。

CLKIN：时钟信号输入端。

CLKR:内部时钟发生器的外接电阻端，与CLK端配合可由芯片自身产生时钟脉冲，其频率计算方式是：fck=1/(1.1RC）。

CS：片选信号输入端，低电平有效。

WR：写信号输入端，低电平启动AD转换。

RD：读信号输入端，低电平输出端有效。

INTR：转换完毕中断提供端，AD转换结束后，低电平表示本次转换已完成。

VREF/2：参考电平输入，决定量化单位。

VCC：芯片电源5V输入。

AGND:模拟电源地线。

DGND:数字电源地线。

③ ADC0804的转换原理

ADC0804是属于逐次逼近式（Successive Approximation Method）A/D转换器，转换速度快，分辨率高。

2、LCD1602液晶

LCD1602液晶模块是两行16个字符，用5×7点阵图形来显示字符的液晶显示器，属于16字×2行类型。内部具有字符发生器ROM( Character- Generator Rom, CG ROM)，可显示192种字符(160个5×7点阵字符和5×10点阵字符)。具有64B的自定义字符RAM( Character- Generator RAM, CG RAM)，可以定义8个5×8点阵字符或4个5×11点阵字符。具有64B的数据显示RAM(Data- Display RAM, DD RAM)。下图为LCD1602的引脚图。

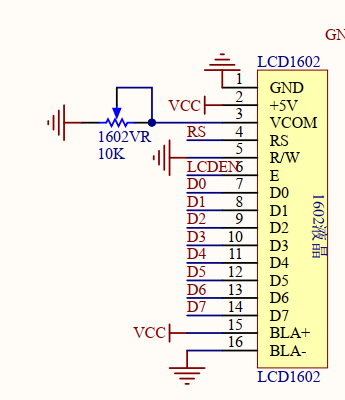


图2 LCD1602液晶引脚

① 模块引脚功能

表1 模块的引脚功能



② LCD1602指令集

表2 LCD1602指令集



**三、实验要求**

1、调节滑动变阻器，对其电压进行采样，采集到的是一个八位的二进制数。

2、LCD1602液晶上的第一行显示采集到的电压值（进制不限）。

3、（选做）LCD1602液晶上的第二行用十进制显示该采样值对应5V的实际电压值，且显示的最小分辨率为0.01V。

**四、实验源代码**

#include<reg52.h>

#include <intrins.h>

typedef unsigned char uchar;

#define AddWr 0x90 //PCF8591 地址

sbit en = P3 ^ 4; //lcd1602写数据

sbit rs = P3 ^ 5; //lcd1602写指令

sbit scl=P2^0; //I2C 时钟

sbit sda=P2^1; //I2C 数据

bit ack; //应答标志位

unsigned char date;

sbit C1=P2^6;//数码管1

sbit C2=P2^7;//数码管2

sbit Dp=P2^5;//小数点

uchar code table[] =

{

"0123456789."

};

uchar code table1[] =

{

"SMC1602 0x01..."

};

unsigned int data dis[3]={0x00,0x00,0x00};

unsigned int getData;

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

起动总线函数

函数原型: void Start\_I2c();

功能: 启动I2C总线,即发送I2C起始条件.

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

void Start\_I2c()

{

sda=1; /\*发送起始条件的数据信号\*/

\_nop\_();

scl=1;

\_nop\_(); /\*起始条件建立时间大于4.7us,延时\*/

\_nop\_();

\_nop\_();

\_nop\_();

\_nop\_();

sda=0; /\*发送起始信号\*/

\_nop\_(); /\* 起始条件锁定时间大于4μs\*/

\_nop\_();

\_nop\_();

\_nop\_();

\_nop\_();

scl=0; /\*钳住I2C总线，准备发送或接收数据 \*/

\_nop\_();

\_nop\_();

}

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

结束总线函数

函数原型: void Stop\_I2c();

功能: 结束I2C总线,即发送I2C结束条件.

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

void Stop\_I2c()

{

sda=0; /\*发送结束条件的数据信号\*/

\_nop\_(); /\*发送结束条件的时钟信号\*/

scl=1; /\*结束条件建立时间大于4μs\*/

\_nop\_();

\_nop\_();

\_nop\_();

\_nop\_();

\_nop\_();

sda=1; /\*发送I2C总线结束信号\*/

\_nop\_();

\_nop\_();

\_nop\_();

\_nop\_();

}

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

字节数据发送函数

函数原型: void I2C\_SendByte(UCHAR c);

功能: 将数据c发送出去,可以是地址,也可以是数据,发完后等待应答,并对

此状态位进行操作.(不应答或非应答都使ack=0)

发送数据正常，ack=1; ack=0表示被控器无应答或损坏。

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

void I2C\_SendByte(unsigned char c)

{

unsigned char i;

for(i=0;i<8;i++) /\*要传送的数据长度为8位\*/

{

if((c<<i)&0x80)sda=1; /\*判断发送位\*/

else sda=0;

\_nop\_();

scl=1; /\*置时钟线为高，通知被控器开始接收数据位\*/

\_nop\_();

\_nop\_(); /\*保证时钟高电平周期大于4μs\*/

\_nop\_();

\_nop\_();

\_nop\_();

scl=0;

}

\_nop\_();

\_nop\_();

sda=1; /\*8位发送完后释放数据线，准备接收应答位\*/

\_nop\_();

\_nop\_();

scl=1;

\_nop\_();

\_nop\_();

\_nop\_();

if(sda==1)ack=0;

else ack=1; /\*判断是否接收到应答信号\*/

scl=0;

\_nop\_();

\_nop\_();

}

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

字节数据接收函数

函数原型: UCHAR I2C\_RcvByte();

功能: 用来接收从器件传来的数据,并判断总线错误(不发应答信号)，

发完后请用应答函数应答从机。

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

unsigned char I2C\_RcvByte()

{

unsigned char retc=0,i;

sda=1; /\*置数据线为输入方式\*/

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

{

\_nop\_();

scl=0; /\*置时钟线为低，准备接收数据位\*/

\_nop\_();

\_nop\_(); /\*时钟低电平周期大于4.7μs\*/

\_nop\_();

\_nop\_();

\_nop\_();

scl=1; /\*置时钟线为高使数据线上数据有效\*/

\_nop\_();

\_nop\_();

retc=retc<<1;

if(sda==1)retc=retc+1; /\*读数据位,接收的数据位放入retc中 \*/

\_nop\_();

\_nop\_();

}

scl=0;

\_nop\_();

\_nop\_();

return(retc);

}

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

应答子函数

函数原型: void Ack\_I2c(bit a);

功能: 主控器进行应答信号(可以是应答或非应答信号，由位参数a决定)

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

void Ack\_I2c(bit a)

{

if(a==0)sda=0; /\*在此发出应答或非应答信号 \*/

else sda=1; /\*0为发出应答，1为非应答信号 \*/

\_nop\_();

\_nop\_();

\_nop\_();

scl=1;

\_nop\_();

\_nop\_(); /\*时钟低电平周期大于4μs\*/

\_nop\_();

\_nop\_();

\_nop\_();

scl=0; /\*清时钟线，住I2C总线以便继续接收\*/

\_nop\_();

\_nop\_();

}

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

\* 函数名 : Pcf8591\_DaConversion

\* 函数功能 : PCF8591的输出端输出模拟量

\* 输入 : addr（器件地址），channel（转换通道），value（转换的数值）

\* 输出 : 无

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

bit Pcf8591\_DaConversion(unsigned char addr,unsigned char channel,unsigned char Val)

{

Start\_I2c(); //启动总线

I2C\_SendByte(addr); //发送器件地址

if(ack==0)return(0);

I2C\_SendByte(0x40|channel); //发送控制字节

if(ack==0)return(0);

I2C\_SendByte(Val); //发送DAC的数值

if(ack==0)return(0);

Stop\_I2c(); //结束总线

return(1);

}

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

\* 函数名 : Pcf8591\_SendByte

\* 函数功能 : 写入一个控制命令

\* 输入 : addr（器件地址），channel（转换通道）

\* 输出 : 无

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

bit PCF8591\_SendByte(unsigned char addr,unsigned char channel)

{

Start\_I2c(); //启动总线

I2C\_SendByte(addr); //发送器件地址

if(ack==0)return(0);

I2C\_SendByte(0x40|channel); //发送控制字节

if(ack==0)return(0);

Stop\_I2c(); //结束总线

return(1);

}

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

\* 函数名 : PCF8591\_RcvByte

\* 函数功能 : 读取一个转换值

\* 输入 :

\* 输出 : dat

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

unsigned char PCF8591\_RcvByte(unsigned char addr)

{

unsigned char dat;

Start\_I2c(); //启动总线

I2C\_SendByte(addr+1); //发送器件地址

if(ack==0)return(0);

dat=I2C\_RcvByte(); //读取数据0

Ack\_I2c(1); //发送非应答信号

Stop\_I2c(); //结束总线

return(dat);

}

/\*------------------------------------------------

串口初始化函数

------------------------------------------------\*/

void init\_com(void)

{

EA=1; //开总中断

ES=1; //允许串口中断

ET1=1;

TMOD=0x22; //定时器T1，在方式2中断产生波特率

PCON=0x00; //SMOD=0

SCON=0x50; // 方式1 由定时器控制

TH1=0xfd; //波特率设置为9600

TL1=0xfd;

TR1=1; //开定时器T1运行控制位

}

/\*------------------------------------------------

延时函数

------------------------------------------------\*/

void delay(unsigned char i)

{

unsigned char j,k;

for(j=i;j>0;j--)

for(k=125;k>0;k--);

}

/\*------------------------------------------------

把读取值转换成一个一个的字符，给串口显示

------------------------------------------------\*/

void To\_ascii(unsigned char num)

{

SBUF=num/100+'0';

delay(200);

SBUF=num/10%10+'0';

delay(200);

SBUF=num%10+'0';

delay(200);

}

/\* 写数据 \*/

void write\_date(uchar date)

{

//参考写操作时序图

rs = 1; //RS高电平,写数据

P0 = date; //写入数据,对应时序图t\_SP2数据建立

delay(5);

en = 1; //打开使能,给E一个高脉冲,开始写入数据

delay(5);

en = 0; //关闭使能,写入数据完成

}

/\* 写指令 \*/

void write\_com(uchar com)

{

//基本原理同写数据

rs = 0; //RS低电平,写指令

P0 = com;

delay(5);

en = 1;

delay(5);

en = 0;

}

/\* 初始化LCD \*/

void init\_lcd()

{

en = 1;

write\_com(0x38); //00111000 设置16\*2显示,5\*7点阵,8位数据接口

write\_com(0x0E); //00001110 开显示,显示光标,不闪烁

write\_com(0x06); //00000110 读写字符后地址指针+1,光标+1,屏幕不移动

write\_com(0x01); //清屏

write\_com(0x80 + 0x01); //设置数据地址指针从第一个开始

}

void Display()

{

write\_date(table[dis[1]]);//整数位

delay(20);

write\_date(table[10]);//小数点

delay(20);

write\_date(table[dis[0]]);//小数位

delay(20);

}

/\*void Display()interrupt 0 using 1

{

uchar i;

for (i = 0; i < 16; ++i) //遍历逐个写入第一行数据

{

write\_date(table1[i]);

delay(20);

}

}\*/

void Clear()

{

write\_com(0x01);

}

/\*------------------------------------------------

主函数

------------------------------------------------\*/

int main()

{

/\* 初始化LCD \*/

init\_lcd();

/\* 初始化中断 \*/

EA = 1;

EX0 = 1;

delay(100);

while(1)

{

PCF8591\_SendByte(AddWr,0); //启动转换

getData=PCF8591\_RcvByte(AddWr); //读转换完的数字信号

dis[1]=getData/51; //整数位

dis[2]=getData%51; //dis[2]位中间暂存数据位

dis[2]=dis[2]\*10;

dis[0]=dis[2]/51; //计算输出电压的小数值

Clear(); //清屏准备显示新内容

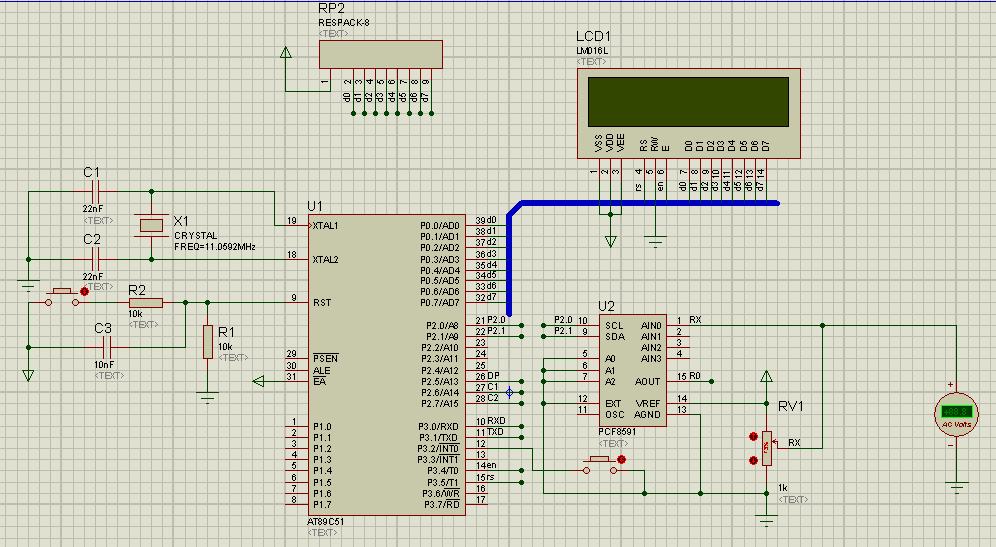
Display(); //显示内容

}

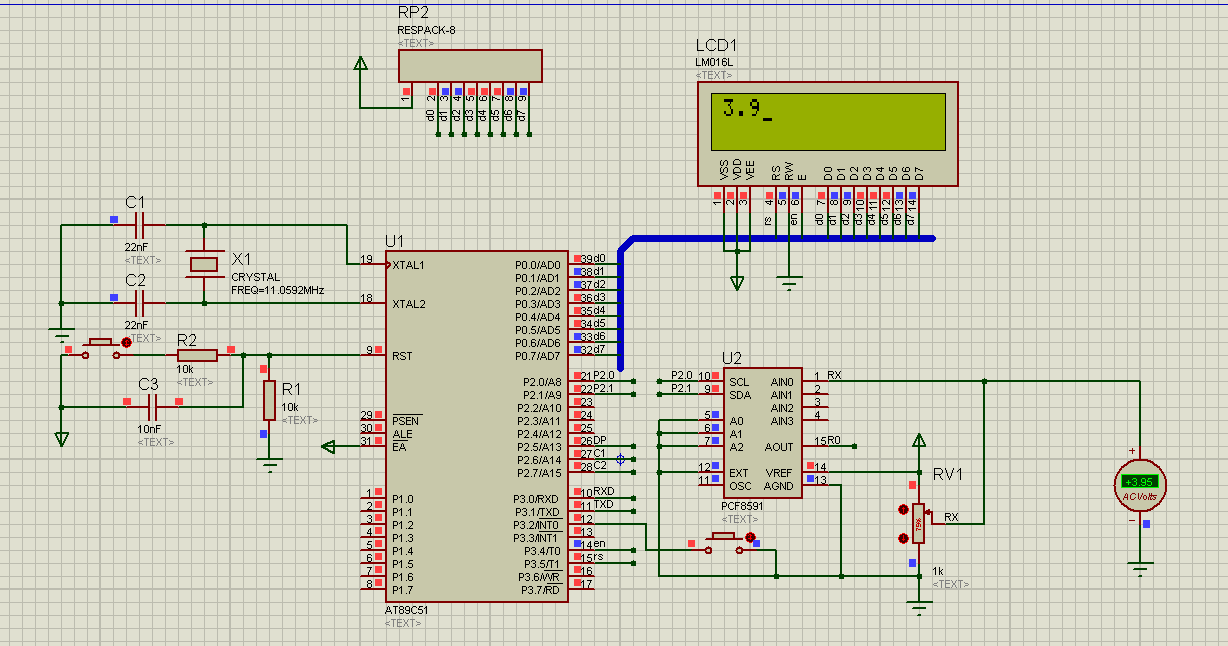
}

**五、实验结果**

1. 电路图：



1. 运行图：



**六、实验小结**

电压表数据生成及读取难度不大，读懂 ADC0804 的功能表即可。Lcd1602是一个比较通用的液晶屏幕，在github上有现成的基础开发库，加载或直接包含进源码后，进行调用即可。Lcd需要注意的是我接在P0口需要加一个上拉电源，否则无法正常使用，另外一个解决方法是改用P1口。

1. **参考文章**

[51单片机 | 实现SMC1602液晶屏显示实例](https://www.cnblogs.com/hughdong/p/6890337.html)

[【C51单片机】数字电压表设计](https://blog.csdn.net/qq_40602000/article/details/84102765)