实验六　51 单片机 IO 接口仿真

一、实验目的

掌握用并行口设计 LED 数码显示和键盘电路的方法和步骤，理解数码管动态显示的实现方式和原理，掌握键盘的去抖动等相关问题的处理方法。熟悉掌握利用 Proteus 进行仿真的步骤和方法。

二、 实验要求

请将附件给出的 Proteus 图用 51 单片机完成一个计算器功能。

1、显示采用动态分时 6 位共阴数码管输出。

2、采用 4*4 矩阵键盘输入，键盘上已经标识对应键。

3、完成两位十进制数的加、减、乘、除运算。

4、开机显示最右边数码管显示‘0’，输入按键值后依次左移。″+—*/″数码管显示分别“A B C D”。

按要求上交实验报告（加、减、乘、除的对应的 Proteus 仿真截图一定要有）。

三、实验步骤

1、键盘处理程序

实现计算器功能首先需要通过键盘输入数据，所以首先需要书写键盘处理程序。其中键盘处理程序包括相关的去抖动问题、按钮识别、按键转化为数字。首先设计 kbscan()子函数实现键盘扫描功能，相应的去抖动问题采用延时函数 delayms()来实现，keypro()子程序将键盘扫描结果转化为数字。其相应的子程序设计如下所示：

1.1 delayms()子程序实现延时

```
void delayms(void)   //延时函数
{
  uchar i;
  for(i=200;i>0;i--){;}
}
```

1.2 kbscan()子程序实现键盘扫描

```
// 扫描键盘返回相应位置编码
uchar kbscan(void) //键盘扫描函数
{
    uchar sccode,recode;
    P1=0xf0; // P1.0~P1.3发全0，P1.4~P1.7输入

    if ( (P1 & 0xf0)!=0xf0)  // * 如P1口高四位不全为1有键按下* /
    {
        delayms(); // * 延时去抖动* /
        if ((P1&0xf0)!=0xf0) // * 在读输入值* /
        {
            sccode =0xfe; // * 最低位置0 * /
            while((sccode&0x10)!=0)   // 不到最后一行循环
            {
                P1 =sccode;     //*P1口输出扫描码
                if ( (P1 & 0xf0)!=0xf0) //  如P1.4~P1.7不全为1，该行有键按下
                {
                    recode = P1 & 0xf0; // * 保留P1口高四位值，低四位变为全1，作为列值* /
                    return((sccode&0x0f)|(recode));      //*行码+列值=键编码返回主程序
                }
                else
                {
                    sccode =(sccode<<1)|0x01; // * 如该行无键按下，查下一行，行扫描值左移一位
                }
            }
        }
    }
    return( 0 );      // * 无键按下，返回值为0
}
```

1.3 keypro()子程序实现数字转化

```
//根据键盘得到数字
unsigned char keypro(void)    //键值转化函数
{
    switch(kbscan())
    {
      case 0xee:return 7;break;
      case 0xde:return 8;break;
      case 0xbe:return 9;break;
      case 0x7e:return 13;break;// 除法
      case 0xed:return 4;break;
      case 0xdd:return 5;break;
      case 0xbd:return 6;break;
      case 0x7d:return 12;break;//乘法
      case 0xeb:return 1;break;
      case 0xdb:return 2;break;
      case 0xbb:return 3;break;
      case 0x7b:return 11;break;//减法
      case 0xe7:return 15;break;//清除
      case 0xd7:return 0;break;
      case 0xb7:return 14;break;//等于
      case 0x77:return 10;break;//加法
      default:return 0xff;break;
    }
}
```

2、数字处理程序

经过上述的键盘处理程序实现读取键盘的值，实现计算器需要设计相应的数字处理程序。该部分程序主要实现保存计算数、识别加减乘除运算、得到结果并将相应值存入

数码管显示数组中。

首先设置数码管段码数组得到每个数字的显示编码、位码决定那个 LED 灯亮、另外设置一个保存每个 LED 灯显示数字的数组，相应设置如下，因为要求最开始显示数字 0，所以 tempdata 中第一个数初始值为 0x3f。

```
uchar code duanma[]={0x3f,0x06,0x5b,0x4f,0x66,0x6d,0x7d,0x07,0x7f,0x6f,0x77,0x7c,0x39,0x5e};// 显示段码值0~F和+-*/ 号
uchar code weima[]={0x3e,0x3d,0x3b,0x37,0x2f,0x1f};//位码,决定显示哪个LED
uchar tempdata[6]={0x3f,0x00,0x00,0x00,0x00,0x00}; //存储相应LED应该显示的值，初始第一位设置为0
```

键盘读取值分为五种情况，分别为：输入第一个操作数、输入操作符、输入第二个操作数、输入等号、输入清零信号。

- 输入第一个操作数

  输入第一个操作数，需要记录第一个操作数的值 num1，将输入值存入 tempdata 以便后面显示，相关程序如下。

```
if(temp>=0&&temp<=9&&biaozhi==0)    //输入第一个操作数
{
  num1=num1*10+temp;
  for(i=count;i>0;i--)
  {
    tempdata[i] = tempdata[i-1];
  }
  tempdata[0]=duanma[temp];
  count++;

}
```

  其中 biaozhi 标志是第一个操作数还是第二个操作数。

- 输入操作符

  输入操作符需要记录操作符并将操作符转移进入 tempdata 数组，标志位变为 1。

```
else if(temp>=10&&temp<=13&&biaozhi==0)  //输入了操作符，需保存操作符
{
  sym=temp; //保留操作符

  for(i=count;i>0;i--)
  {
    tempdata[i] = tempdata[i-1];
  }
  tempdata[0]=duanma[temp];
  count++;

  num=count;    //记录第一个数加上操作符的位数
  biaozhi++;
}
```

- 输入第二个操作数

  需要记录第二个操作数并将输入数字转移进入 tempdata 数组，标志位不变。

```
else if(temp>=0&&temp<=9&&biaozhi==1)    //输入第二个操作数
{
  num2=num2*10+temp;

  for(i=count;i>0;i--)
  {
    tempdata[i] = tempdata[i-1];
  }
  tempdata[0]=duanma[temp];
  count++;

}
```

- 输入等号

  输入等号需要进行相关运算得到 value 结果同时将数组 tempdata 清零来清空输出，相关标志位、输入数字等清零。

```
else if(temp==14&&biaozhi==1)        //等号，需要输出结果
{

  //加减乘除
  if(sym==10)value=num1+num2;
  else if(sym==11)value=num1-num2;
  else if(sym==12)value=num1*num2;
  else if(sym==13)value=num1/num2;
  for(i=0;i<6;i++)tempdata[i]=0x00;    //前面显示的先变为0

  biaozhi=0;
  count=0;
  num=0;
  num1=0;
  num2=0;
}
```

- 输入清零信号

  数字清零并显示出 0，实现初始化。

```
else if(temp==15)                    //清除
{
  biaozhi=0;
  count=0;
  num=0;
  num1=0;
  num2=0;
  for(i=0;i<6;i++)tempdata[i]=0x00;
  tempdata[0]=0x3f;
}
```

3、数码管显示和结果转换

前面将需要转换的数据存入到 tempdata 数组中，将需要亮的数码管通过 weima 数组存储相应的显示程序如下。

```
void display(void)
{
  uchar i;
  for(i=0;i<6;i++)   //六个数码管
    {
      P2=0x00;
      P2=tempdata[i]; //数码管显示的值
      P3=weima[5-i];  //哪一个数码管显示
      delayms();
    }
}
```

得到结果 value 数值需要将其每个位数进行拆分，因此需要判断 value 的大小，根据大小进行判断，将各个位数拆开并通过数码管显示。

```
        if(value<10)tempdata[0]=duanma[value];
        else if(value<100){
          tempdata[1]=duanma[value/10];
          tempdata[0]=duanma[value%10];
        }

        else if(value<1000){
          tempdata[2]=duanma[value/100];
          tempdata[1]=duanma[(value%100)/10];
          tempdata[0]=duanma[value%10];
        }

        else if(value<10000){
          tempdata[3]=duanma[value/1000];
          tempdata[2]=duanma[(value%1000)/100];
          tempdata[1]=duanma[(value%100)/10];
          tempdata[0]=duanma[value%10];
        }

        else if(value<100000){
          tempdata[4]=duanma[value/10000];
          tempdata[3]=duanma[(value%10000)/1000];
          tempdata[2]=duanma[(value%1000)/100];
          tempdata[1]=duanma[(value%100)/10];
          tempdata[0]=duanma[value%10];
        }

        else if(value<1000000){
          tempdata[5]=duanma[value/100000];
          tempdata[4]=duanma[(value%100000)/10000];
          tempdata[3]=duanma[(value%10000)/1000];
          tempdata[2]=duanma[(value%1000)/100];
          tempdata[1]=duanma[(value%100)/10];
          tempdata[0]=duanma[value%10];
        }
```

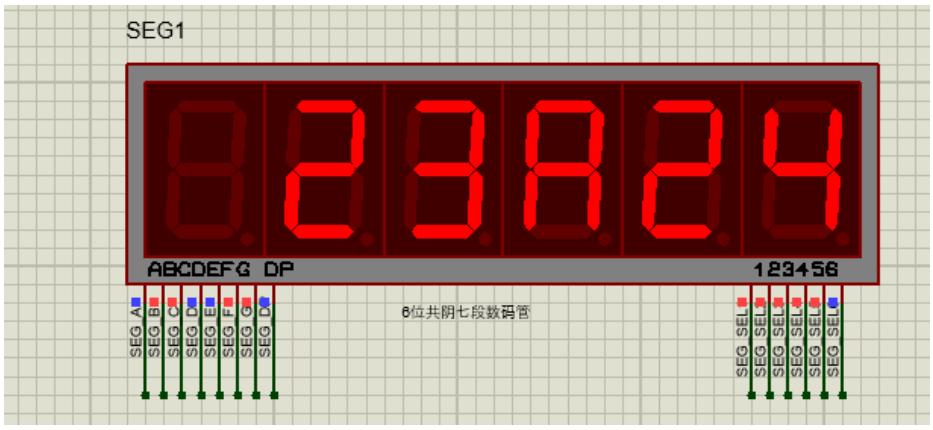
讲过上述步骤，将相应子程序组织起来得到计算器的 c 语言文件，在 keil 中编译成 hex 文件后，打开题目提供的文件，双击单片机添加编译生成的 hex 文件之后可以生成实验结果得到符号要求的计算器。

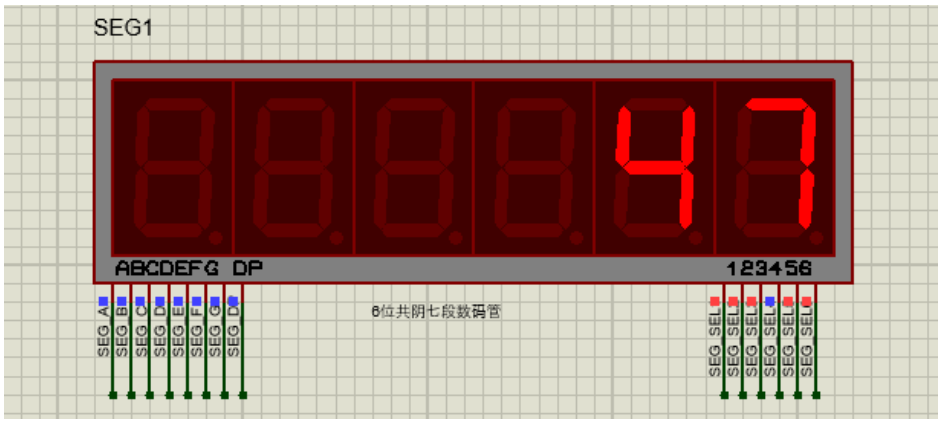
四、实验结果

由前述步骤得到的计算器得道如下结果。

1. 加法运算

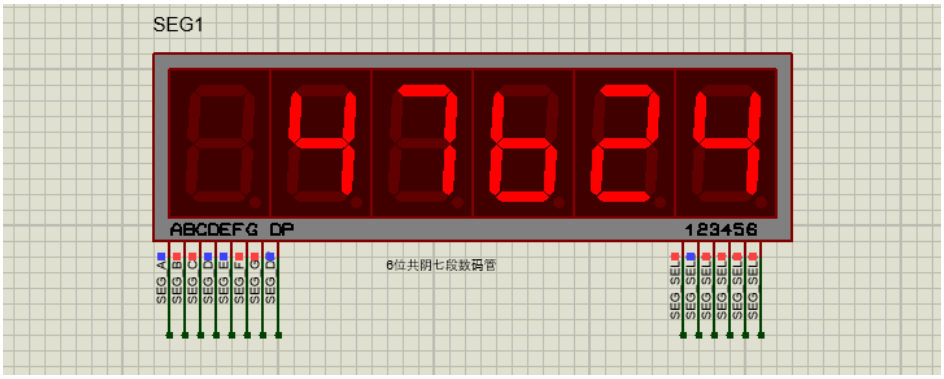
23+24=47

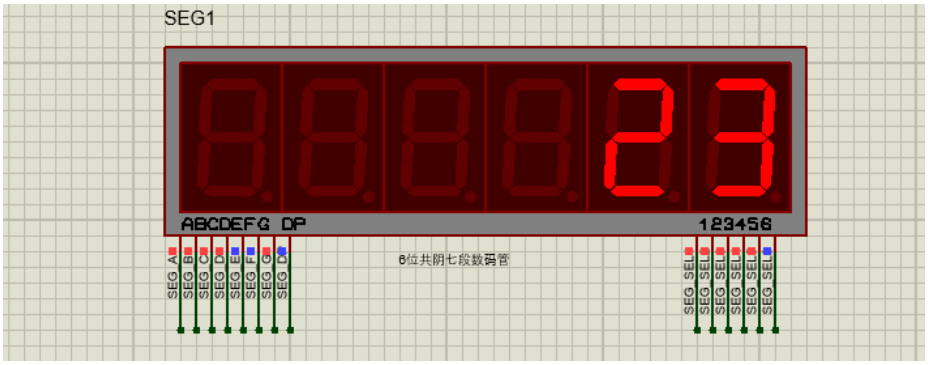




2. 减法运算

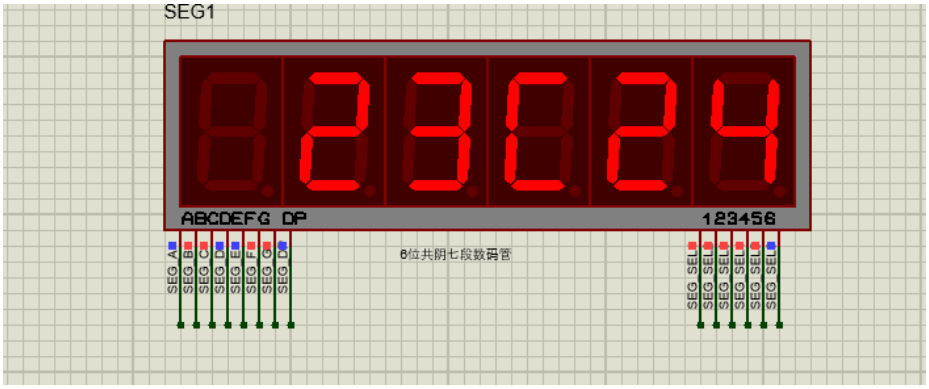
47-24=23

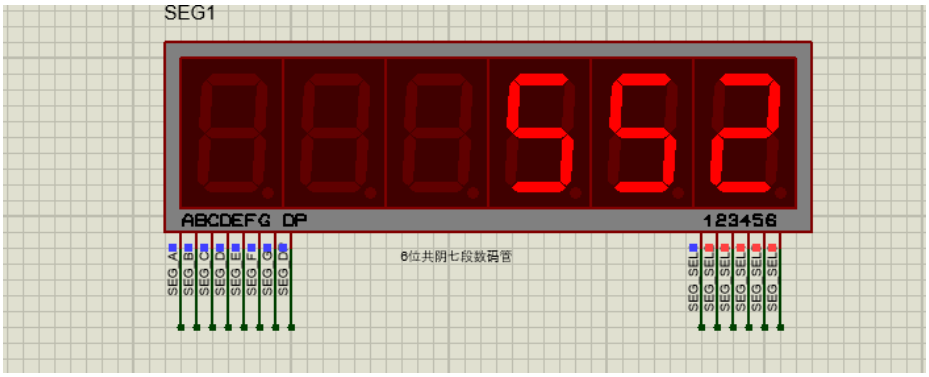




3. 乘法运算

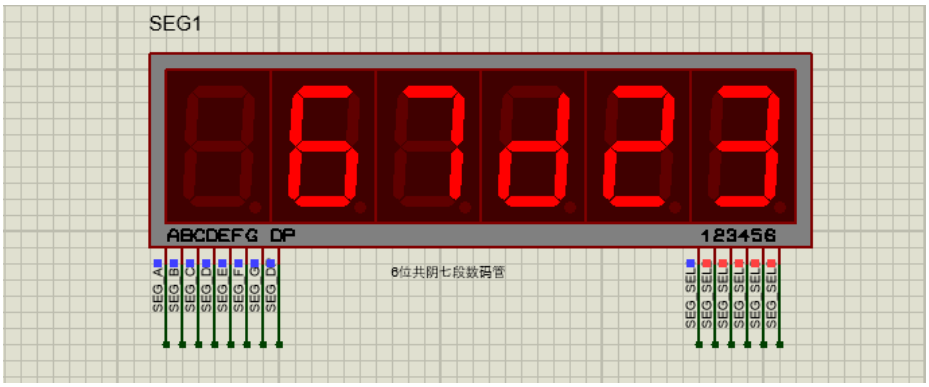
23*24=552

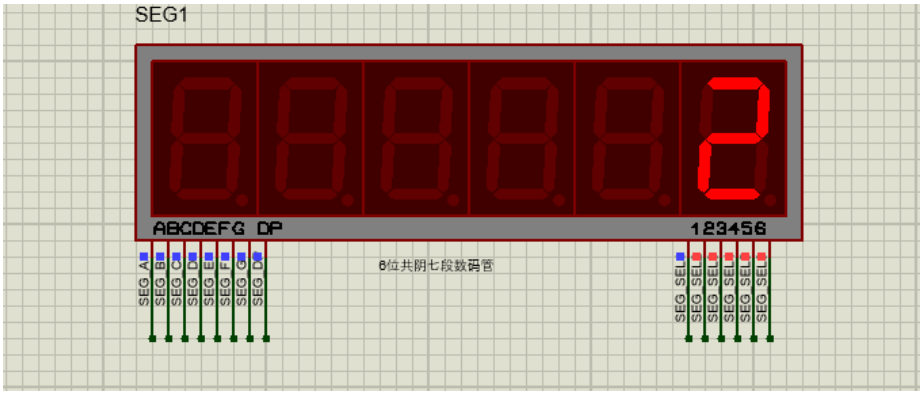




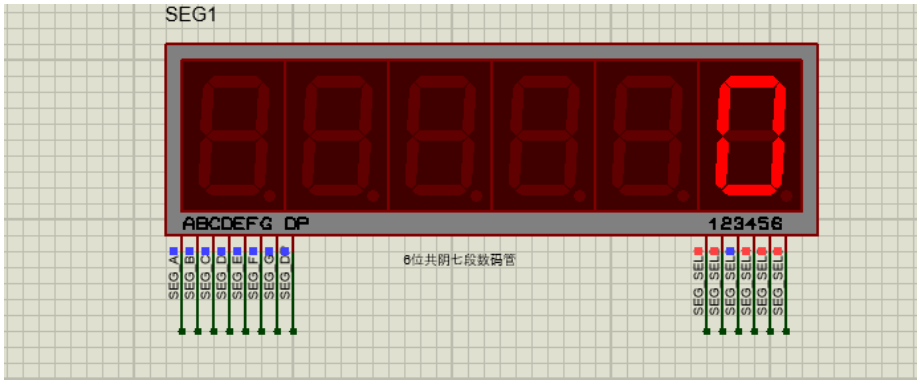
4. 除法运算

67/23=2





5. 清零及初始



五、实验总结

通过本次仿真实验了解了如何利用并行口设计 LED 数码显示和键盘电路。本次实验开始由于键盘去抖动问题处理不好难以得到解果，后来采用前后数字输出要尽量不一样的方法实现去抖问题。另外本次实验所得计算器理论上可以计算各种数值，一位数、二位数、三位数皆可。但是没有设置相应的溢出处理程序，若要求取得更大范围的计算可以考虑将数值设置为 long 类型，这样可以计算更大的数值。