中国传媒大学电子竞赛设计报告

设计题目：简单电子计算器

设计人员：马小雨

学号：200910013344

所在院系：信息工程学院广播电视工程系

指导教师：卢起斌

目 录

一：设计任务与要求

1. 概要

简易计算器以 stc89c52 为核心，通过对 4*4 键盘和显示器件的控制以及单片机内部的运算程序，实现简易计算器的功能，由于 1602 液晶可以显示两行，可以同时看见输入和结果，更加人性化，因此显示器件选择 1602 液晶。

2. 设计任务与效果

功能 1：按运算符的优先级完成长整形数据的四则运算，而不是按输入运算符号的顺序进行运算。例如对于算式 8+9*2+1，如果每读到一个运算符便立即进行运算，那么上式的输出结果为：

Answer=(（8+9）*2)+1=72*2+1=145 ，明显不符合四则运算运算符的优先级顺序，正确顺序应该为：

Answer=8+（9*2）+1=27。

功能 2：保留前一次运算结果，具体功能如下例：

第一步：先输入 7+6*9+2，键入‘=’，则第二行显示输出结果 63；

第二步：再输入+3*2，键入‘=’，则第一行显示“ANS+3*2”，第二行显示 63+3*2 的结果 69；

第三步：不做数字和运算符的输入，直接再按一次‘=’，则第一行仍显示“ANS+3*2”，而第二行显示 69+3*2 的结果 75.

功能 3：按下清零键，显示清屏。如果发现输入失误或运算错误，可以按下清零键重新输入。

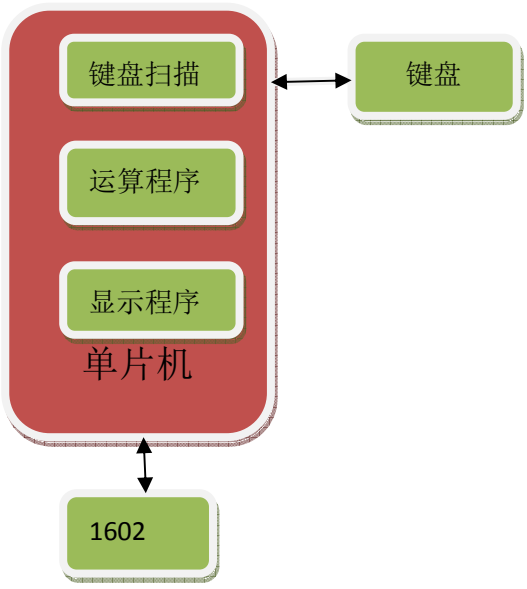
二：总体设计思路

1：通过键盘扫描函数将键入的值立即显示并且分别存入数值数组 num[]和符号数组 symb[]，例如键入算式 88+988*2+1 通过键盘扫描函数可以得到数值数组：num[]={88,988,2,1}和符号数组：symb[]={‘+’，‘*’，‘+’}。

2：如果通过键盘扫描函数发现‘=’键入，则进入运算函数，对 num[]和 symb[]进行解析，得到结果 anser（float 型）

3：把 anser 写入 result 数组（char 型），对 result 数组进行显示操作

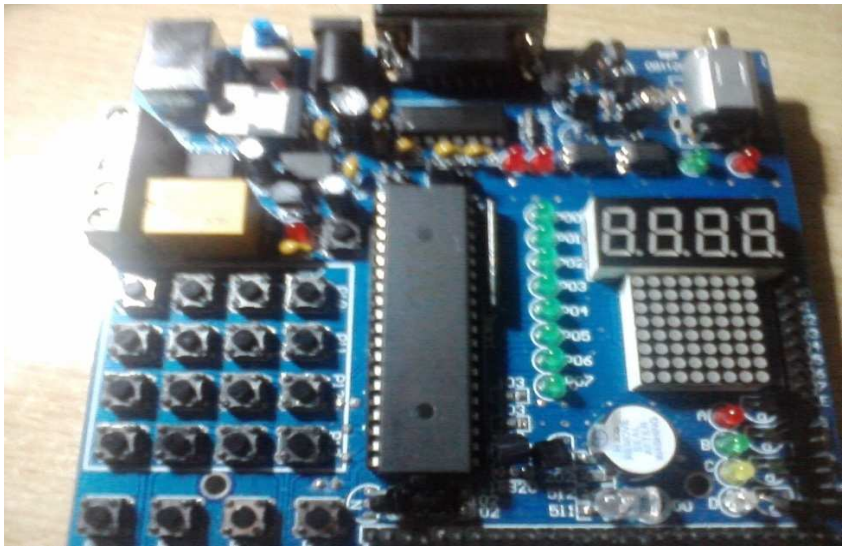
流程图：



三：硬件平台简介

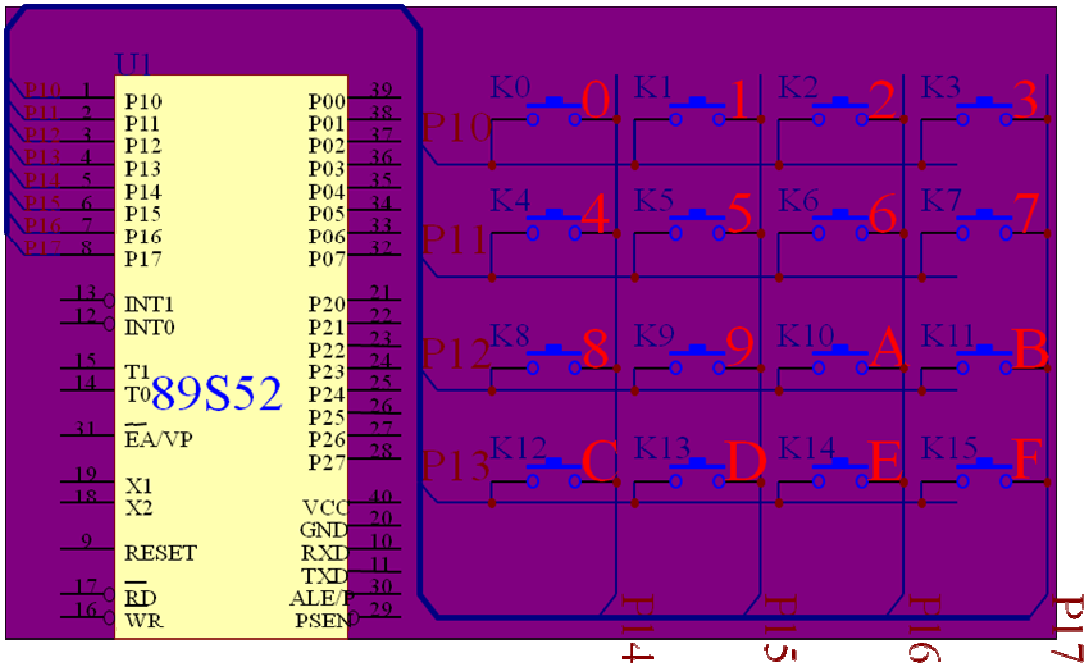
1. AM510 单片机开发板

此计算器的硬件设计基于 AM510 单片机开发板，开发板的具体器件有：4 位数码管（动态显示）；8 个发光二极管；四个独立按键；4*4 的矩阵键盘；一个无源蜂鸣器电路；RS232 串行接口；5V 继电器驱动电路；温度传感器 DS18B20；红外模拟发射接受电路；Rs485 通讯电路；直流电机电路等，如图所示：



2. 4*4 键盘

该单片机开发板上，4*4 键盘占用 P1 口, 4*4 键盘在开发板内部电路如图所示:



3. 1602 液晶

1602 液晶需要通过杜邦线与单片机连接，连线方法见下表：

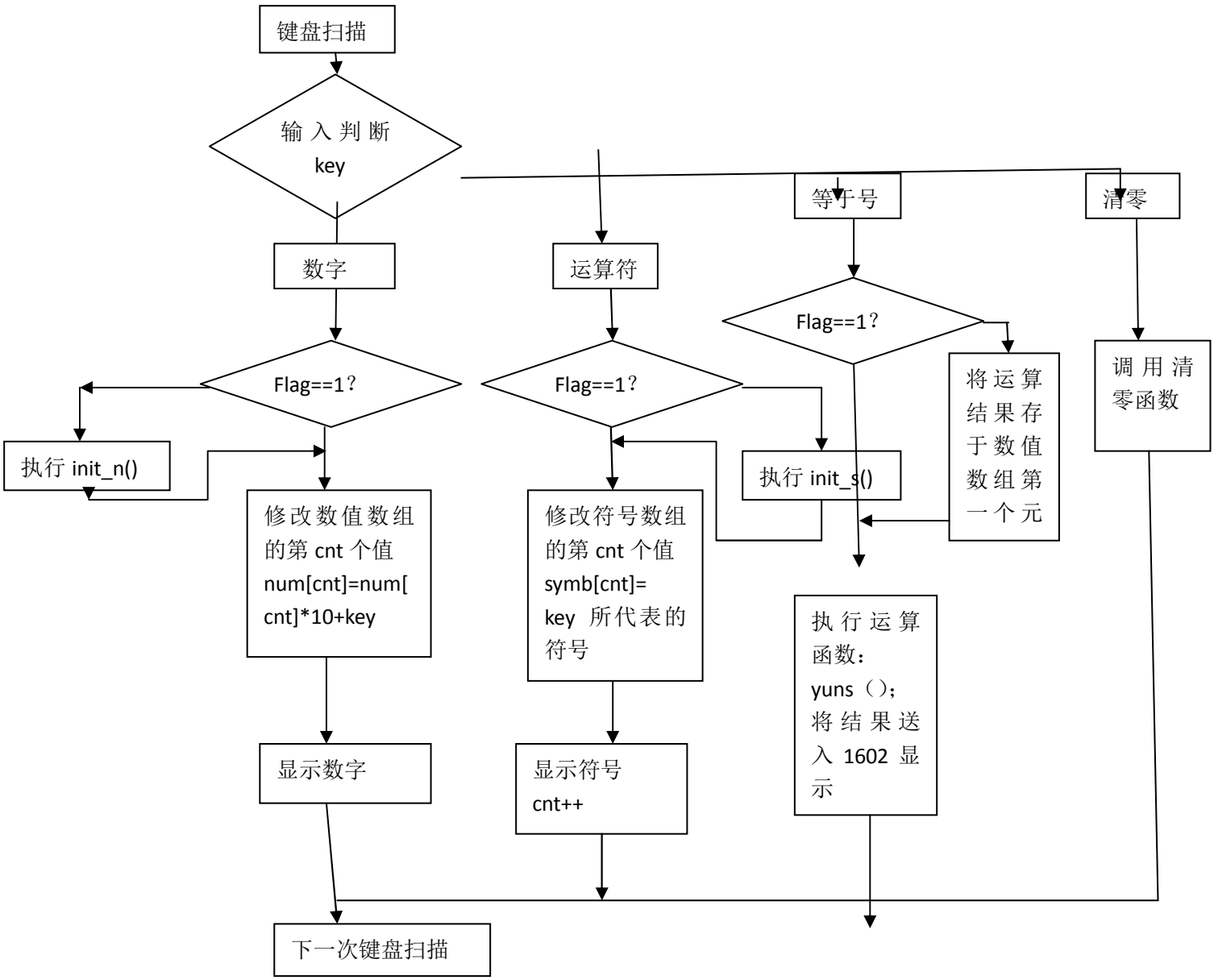
1602 脚	1	2	3	4	5	6	7	8
功能	Vss	Vdd	V0	Rs	Rw	En	D0	D1
单片机脚	Gnd	Vcc	Gnd	P2^1	P2^2	P2^3	P0^0	P0^1
1602 脚	9	10	11	12	13	14	15	16
功能	D2	D3	D4	D5	D6	D7	背光+	背光—
单片机脚	P0^2	P0^3	P0^4	P0^5	P0^6	P0^7	VCC	GND

其中 D0~D8 可以与单片机的 P0，P1，P2，P3，任意一端口相连，（键盘扫描占用的口不能用）

Rs，rw，en 也可以与任意三个 I/O 引脚相连，但在程序的 head.h 头文件中要声明清楚。

四：软件设计

1. 键盘扫描函数程序流程



键盘扫描函数的说明：通过键盘扫描得到不同 key 值，key=0~9 时代表数字，key=10，11，12，13 分别代表加减乘除四个运算符。然后分别存放进入 num 数组或符号数组或执行操作

例如键入算式 88+988*2+1 通过键盘扫描函数可以得到数值数组：num[]={88,988,2,1};

符号数组：symb[]={‘+’,‘*’,‘+’}。具体地说，一开始时，数组计数变量 cnt=0，键盘扫描得到 8，则 num[0]=8,第二次扫描仍为为数字 8，于是 num[0]=8*10+8=88。第三次扫描得符号‘+’，于是 symb[0]=’+’，cnt++,cnt 变为 1，

第四次扫描得到数字 9，于是 num[1]=9 第五次扫描得到数字 8……依次类推，得到上述的数字数组 num[]={88，988，2，1}和字符数组 symb[]={‘+’，‘*’，‘+’}。

之后只要将两个数组送入运算函数进行解析，即可得到结果。

另外，flag==1 表示前一次按下‘=’，则此时第一行需要重新写入，如果按下数字键，则直接显示数字，如果按下运算符如‘+’，则显示 ANS+，而如果此次键入值还是‘=’，则将上一次运算结果赋 num[0],以便进行迭代运算。

2. 运算函数

设计思想

如果仅仅需要按运算符号输入顺序而不是优先级进行运算，那么只需要简单的 for 语句和 switch 语句的嵌套即可。

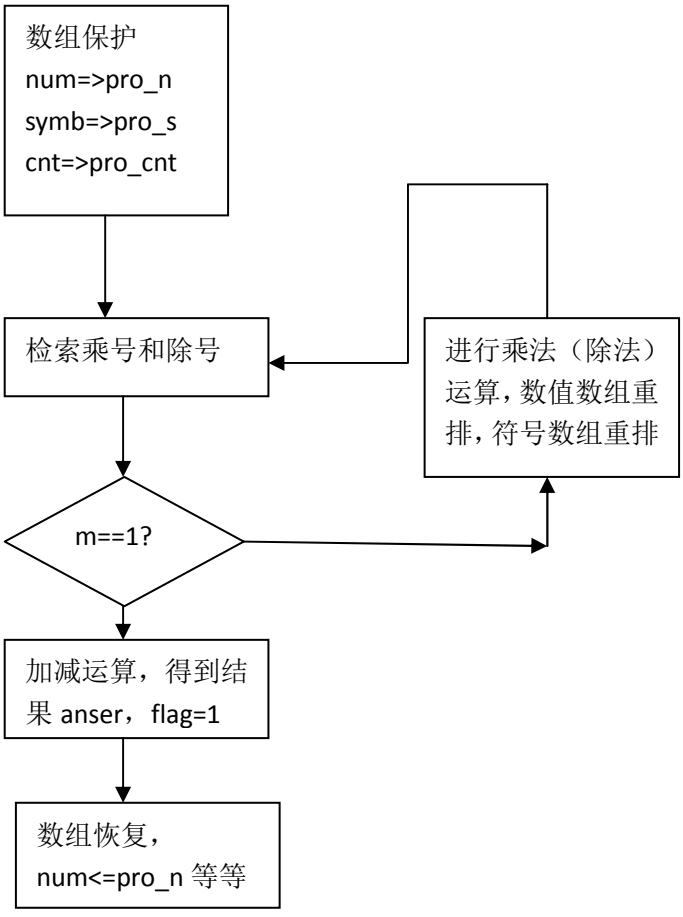
而若考虑运算优先级，则需要进行数组重排。

但是数组重排必然导致原数组的改变，这样连按两次等号就可能产生不同的结果，因此还需要在重排前进行数组保护

运算结束后还要将 flag 标志至 1

设计流程

（m 为标记位，m==1 表示找到‘/’或‘*’）



运算函数的具体程序

//num[]:数值数组；symb[]:符号数组；

```
//cnt:数组计数变量；
// pro_n[]数值数组保护数组；pro_s[]:字符数组保护数组；
//pro_cnt:计数变量保护变量
//m:乘除号查询成功标志位；
//tag:查询到的乘除号在数组中的序号
void yuns()
{
    pro_cnt=cnt;
    for(k=0;k<=cnt;k++)
    {
        pro_n[k]=num[k];
        pro_s[k]=symb[k];
    }//数组保护
    m=1;//表示是否找到乘除符号
    while(m)
    {
        m=0;
        for(k=0;k<cnt;k++)
        {
            if((symb[k]=='*')||(symb[k]=='/'))
            {
                m=1;tag=k;break;//tag 表示乘除号在符号数组中的位置
            }
        }
        if(m==1)
        {
            cnt--;
            switch(symb[tag])
            {
                case '*':num[tag]=num[tag]*num[tag+1];break;
                case '/':num[tag]=num[tag]/num[tag+1];break;
            }//每找到一个乘除号，立刻进行运算
            for(k=tag+1;k<=cnt;k++)
            { num[k]=num[k+1];}
            for(k=tag;k<cnt;k++)
            { symb[k]=symb[k+1];}//数组重排
        }
    }
    for(k=0;k<cnt;k++)
    {
        switch(symb[k])
        {
            case '+':num[k+1]=num[k]+num[k+1];break;
            case '-':num[k+1]=num[k]-num[k+1];break;
```

```
        }
    }//最后只剩下加减号，按顺序进行运算即可
    anser=num[cnt];
    cnt=pro_cnt;
    for(k=0;k<=cnt;k++)
    {
        num[k]=pro_n[k];
        symb[k]=pro_s[k];
    }//数组恢复
    flag=1;
}
```

3. 显示函数

显示函数较为简单，通过对 1602 的 rs，rw、en 口的操作实现读写状态的改变，通过对 D0~D7 写入数据来进行数据传递。对 1602 液晶内的存储器写入数据和写入命令的函数（write_data()和 write_com()）可以参考 AM510 自带的程序包。需要注意的是,通过 write_com()函数向 1602 的 DDRAM 写入显示地址（假设为 Address）时，write_com（）内的参数为应该为 0x80+Address，因为向 DDRAM 写入指令的格式为：

指令功能	指令编码										执行时间/us
	RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0	
设定DDRAM地址	0	0	1	CGRAM的地址(7位)							40

数据位的最高位总为一，所以要加上 0x80。另外每写入一次显示地址，下一次的显示位置会自动加一，所以写入一行数据时只要确定首字符的显示地址即可。

五：总结

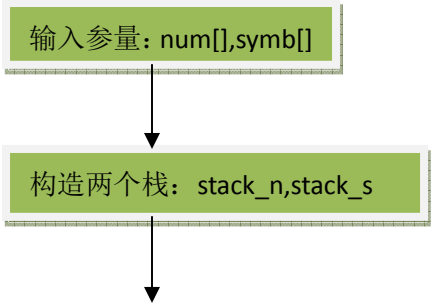
这款计算器还有很多地方可以改进，比如：

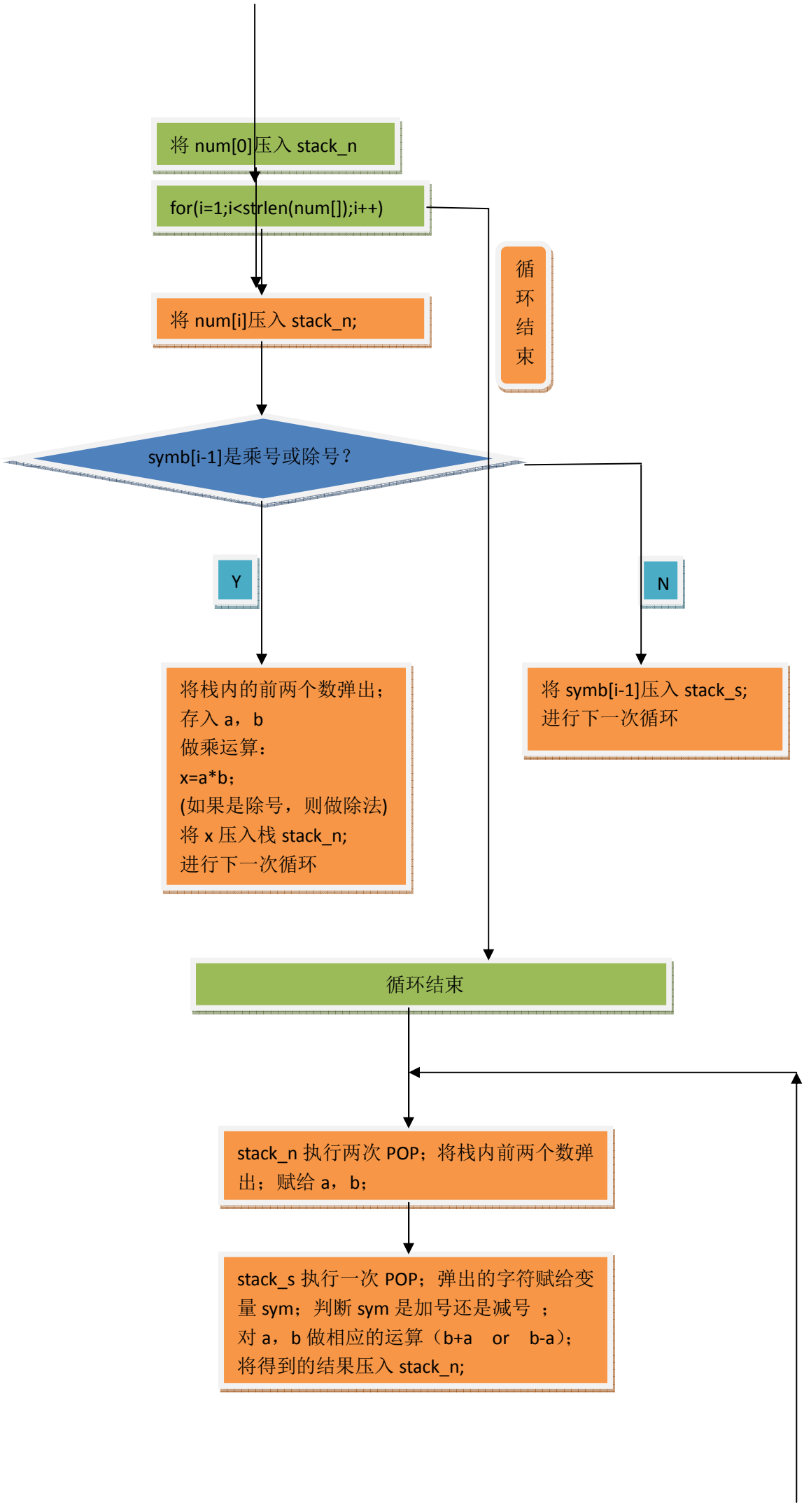
1：由于 4*4 键盘数目的限制，没有设置小数点键不能进行小数运算

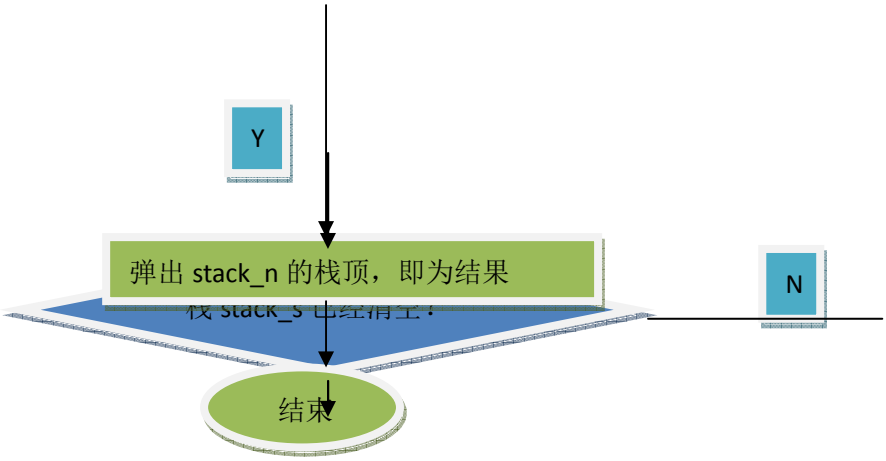
2：没有设置功能键进行按键复用

3：可以将算式以两个数组的形式保存，结合按键复用查询输入历史

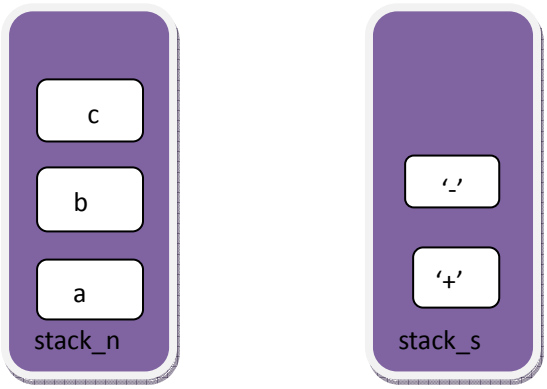
4: 此运算函数还可以通过堆栈结构实现,设计思路如下：



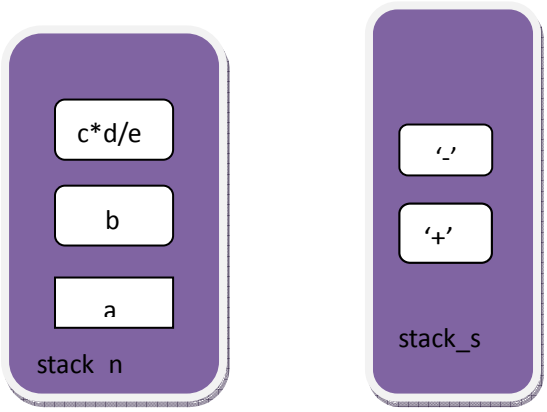

将 num[0]压入 stack_n
for(i=1;i<strlen(num[]);i++)
循环结束
将 num[i]压入 stack_n;
symb[i-1]是乘号或除号？
Y
N
将栈内的前两个数弹出；
存入 a，b
做乘运算：
x=a*b；
(如果是除号，则做除法)
将 x 压入栈 stack_n;
进行下一次循环
将 symb[i-1]压入 stack_s;
进行下一次循环
循环结束
stack_n 执行两次 POP；将栈内前两个数弹出；赋给 a，b；
stack_s 执行一次 POP；弹出的字符赋给变量 sym；判断 sym 是加号还是减号 ；
对 a，b 做相应的运算（b+a or b-a）；
将得到的结果压入 stack_n;



例如 num[]={a,b,c,d,e};symb[]={'+','-','*','/'};
第一步：将 a 压入 stack_n;
第二步：将 b 压入 stack_n;将 '+' 压入 stack_s;
第三步：将 c 压入 stack_n;将 '-' 压入 stack_s;
此时两个栈的情况如图：



第四步：将 d 压入 stack_n;此时 symb[2]为乘号，所以将 d 和 c 弹出，做乘法运算，得到 x（数值为 c*d）
第五步：将 e 压入 stack_n;此时 symb[3]为除号，所以将栈内前两个数弹出（即为 e 和 x），做除法运算：x=x/e;得到 x（数值为 c*d/e）
此时两个栈的情况：



第六步：将 x（即为 c*d/e）和 b 进行减运算得到 b-c*d/e；存入原 b 的位置
第七步：将 x（即为 b-c*d/e）和 a 进行加运算得到 a+b-c*d/e；即为结果
用堆栈的方法来实现运算函数可以使得计算器的速度更快，而且程序思路更加明确，省去了数组保护、数组重排时繁琐的循环语句。但是这种方法占用的内存很大，我将程序编好，调试无误并烧入单片机时，发现程序不能正确运行，很

可能是 128B 的内存不足以建立两个堆栈结构。我认为要解决这种问题需要再外接一个 RAM，由于目前资源有限，很遗憾没有将它实现。

5：由于用杜邦线连接，线路有时不稳定。

由于课余时间有限，只能将计算器做到这样，希望以后有时间可以完善。

最后，感谢卢老师，还有新媒体研究院师兄，让我的 C 语言编程的能力得到显著的提高，感谢你们。

六：参考文献

【1】李广第，朱月秀，冷祖祁，《单片机基础》，北京航空航天出版社，2010 年

【2】郭天翔，《51 单片机基础和 C 语言教程》 ，电子工业出版社，2009 年

【3】《AM510 单片机开发板简介》 ，http://www.docin.com/p-186583365.html