**中国石油大学(北京)**

**课程设计报告**

**课程名: 硬件综合实践**

**姓 名 孙啸峰**

**学 号 2017011316**

**班 级 计算机17-1班**

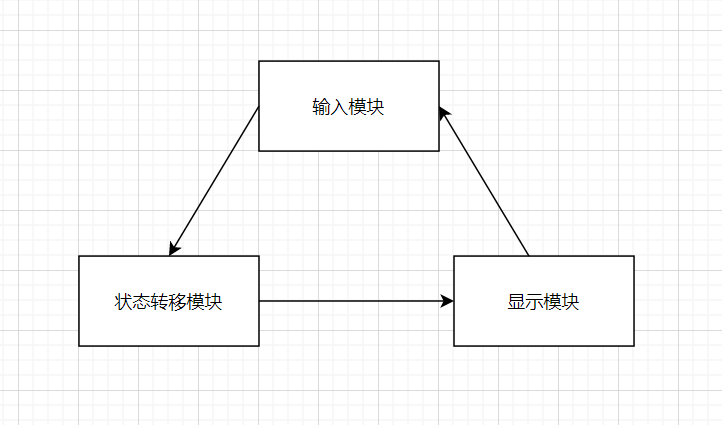
**设计时间 2021.1.4**

# 设计内容

实现10位以内的带括弧的四则混合运算，支持实数运算。

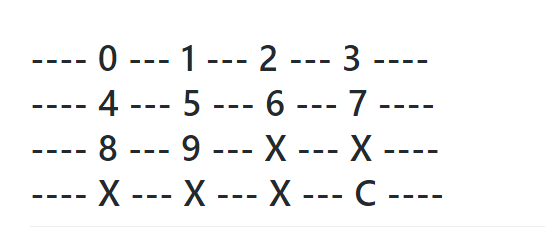
# 设计思路

将整个程序分为输入模块，状态转移模块，显示模块

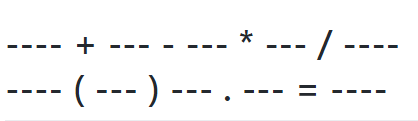


输入模块使用矩阵键盘和独立按键，布局如下：

矩阵键盘：



独立按键：



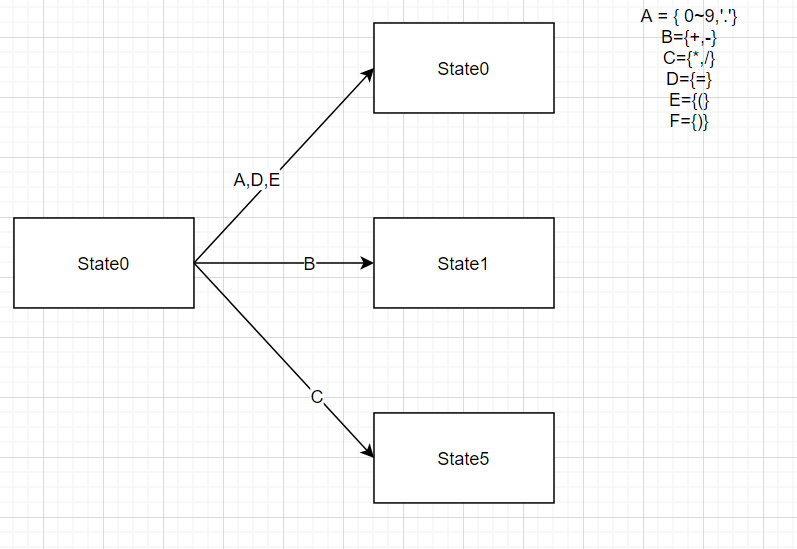
显示模块使用Lcd1602用来输出。

根据计算的状态图，通过构想最终划分为7种状态。

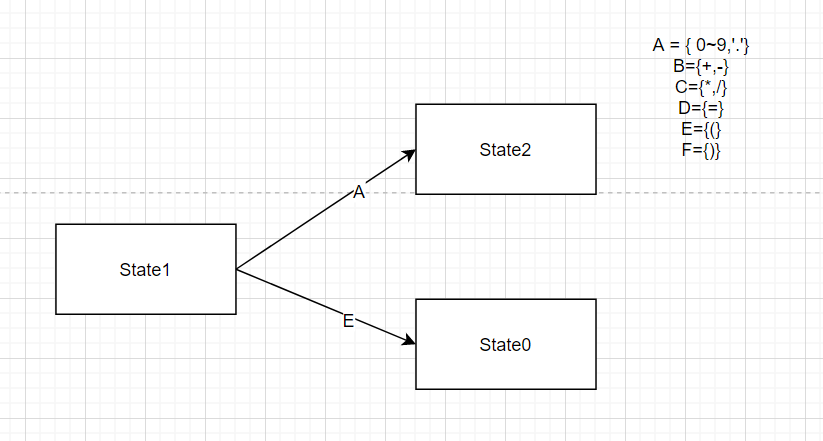
定义 A={0~9,.},B={+,-},C={\*,/},D={=},E={(},F={)}

状态图:

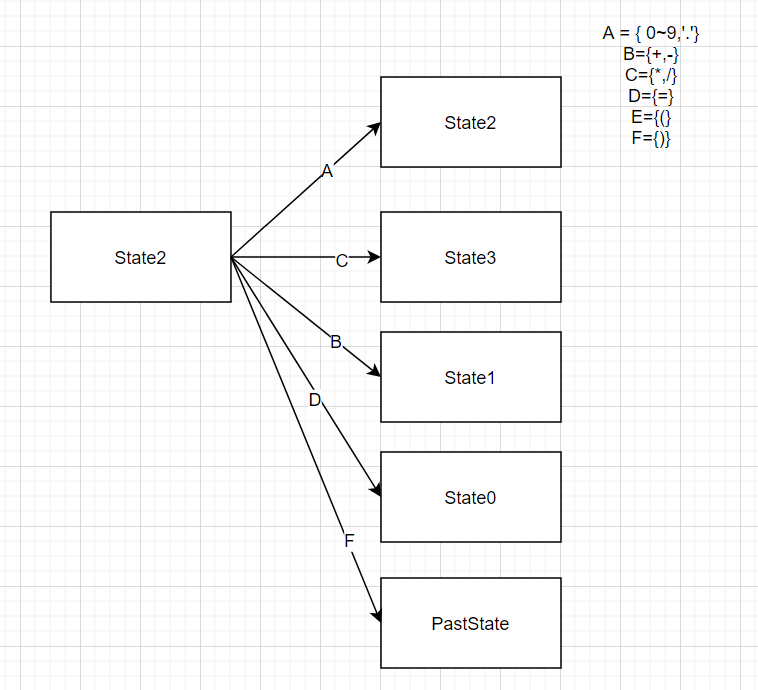
0：初始状态



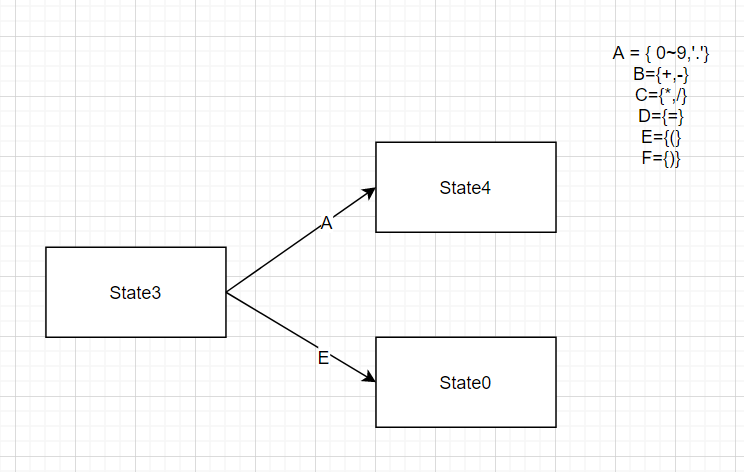
1： a +-状态



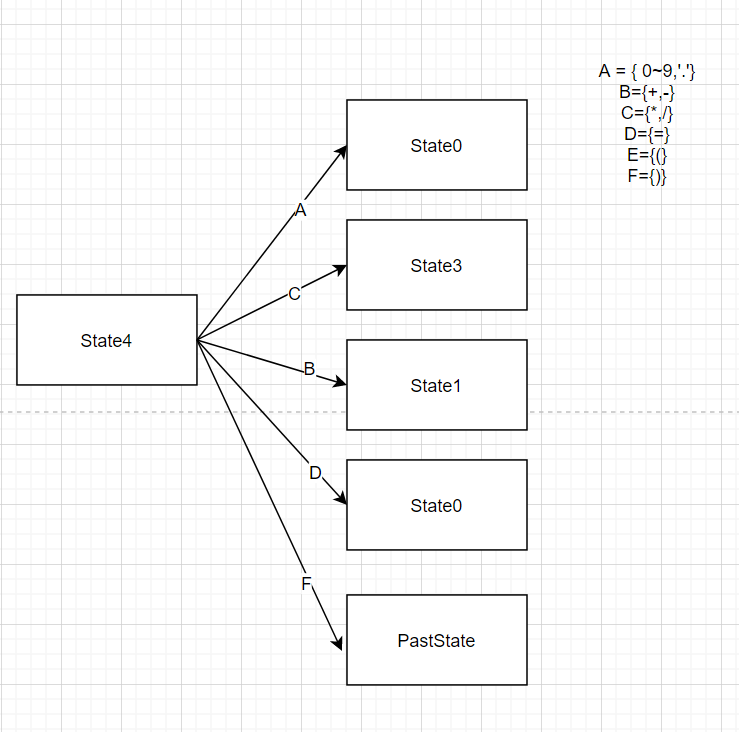
2：a+-b状态



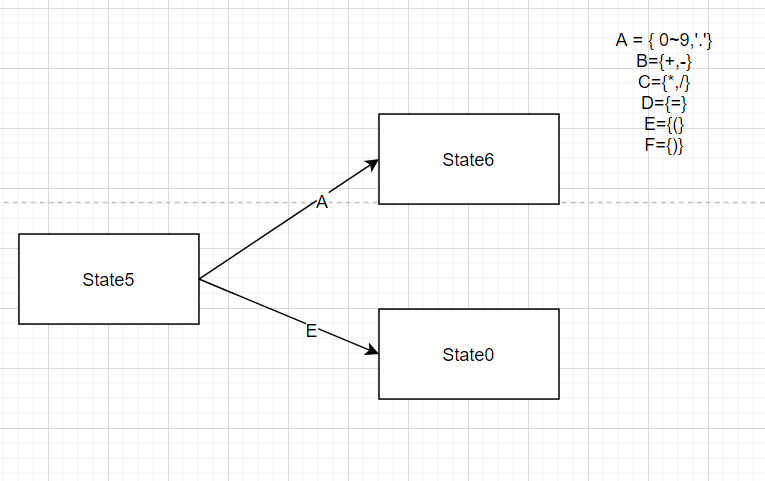
3：a+-b\*/状态



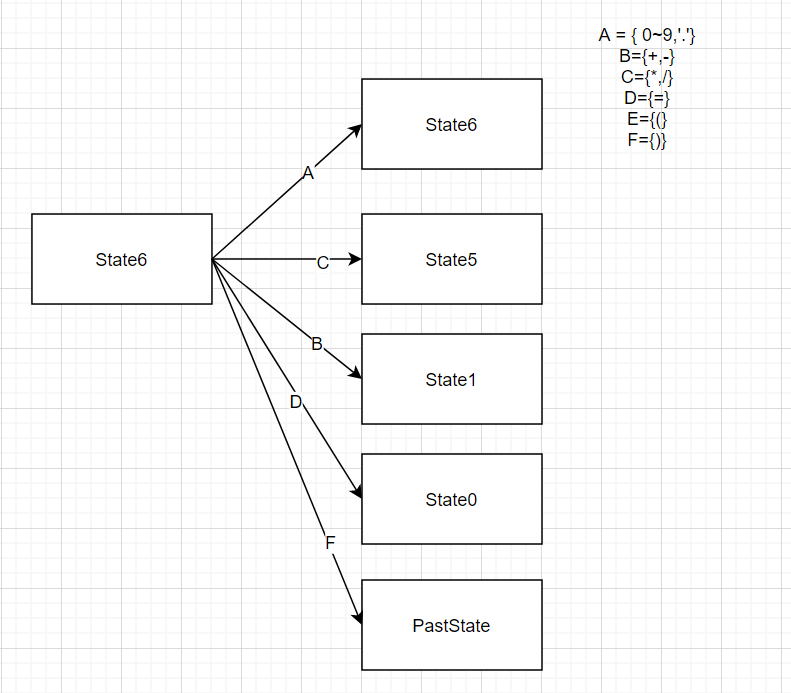
4：a+-b\*/c状态



5：a\*/状态



6：a\*/c状态



‘（’运算完成压栈操作，记录当前状态，同时回到状态0。

‘）’运算完成出栈，还原压栈前状态。

# 设计解释

## 3.1、设计环境介绍

使用Keil5软件C51软件。普中自动下载软件完成烧录。STC89C516芯片，LCD1602显示。

## 3．2、程序解释（可含硬件部分）

#include "lcd.h"

#include <reg51.h>

#include <stdio.h>

#include <math.h>

typedef unsigned int u16; //对数据类型进行声明定义

typedef unsigned char u8;

const char Outputchar[18]={48,49,50,51,52,53,54,55,56,57,43,45,42,47,40,41,46,61};

#define MAX\_LEN 4

float a = 0.0 , b = 0.0, c =0.0 ;

float pastdata[MAX\_LEN];

u8 Statestack[MAX\_LEN];

u8 stackpoint = -1, pastpoint = -1;

u8 flag\_a = 0 , flag\_b = 0 , flag\_c =0 ;

u8 State = 0; //最开始在初始状态

u8 add\_sub\_flag = 0 ; //1->'+' , 0->'-'

u8 multi\_div\_flag = 0 ; //1->'\*' , 0->'/'

u8 count = 0 ; //对应括号的匹配

// State : 0 0 0 0 0 0 0 0 -> 128 64 32 16 8 4 2 1

void printans(float a);

void initA(){

a = 0;

flag\_a = 0;

}

void initB(){

b = 0 ;

flag\_b =0 ;

}

void initC(){

c = 0 ;

flag\_c = 0;

}

void pushState(){

switch(State){

case 0: // (

stackpoint ++;

Statestack[stackpoint] = State;

break;

case 1: // a +/- (

if(add\_sub\_flag == 1){

State += 128;

}

Statestack[++stackpoint]=State;

pastdata[++pastpoint] = a;

break;

case 3: // a +/- b \*/ (

if(add\_sub\_flag == 1)

State+=128;

if(multi\_div\_flag == 1)

State+=64;

Statestack[++stackpoint]=State;

pastdata[++pastpoint] = a;

pastdata[++pastpoint] = b;

break;

case 5: // a \*/ (

if(multi\_div\_flag == 1){

State += 64;

}

Statestack[++stackpoint]=State;

pastdata[++pastpoint] = a;

break;

}

initA();

}

void PopStack(){

u8 past = Statestack[stackpoint--];

switch(past&0x3F){ // 0011 1111

case 0:

State = 0;

a = a;

break;

case 1:

State = 2;

if((past & 128) ==128){ // +

add\_sub\_flag = 1;

}

else

add\_sub\_flag = 0;

b = a;

a = pastdata[pastpoint--];

break;

case 3:

State = 4;

if((past & 128) == 128){

add\_sub\_flag= 1;

}

else

add\_sub\_flag = 0;

if((past & 64) == 64){

multi\_div\_flag = 1;

}

else

multi\_div\_flag = 0;

c = a;

b = pastdata[pastpoint--];

a = pastdata[pastpoint--];

break;

case 5:

State = 6;

if((past & 64) == 64){ // \*

multi\_div\_flag = 1;

}

else

multi\_div\_flag = 0;

c = a;

a = pastdata[pastpoint--];

break;

}

}

void printans(float a){

u8 i ;

char str[8];

sprintf(str,"%f",a);

LcdWriteCom(0xc0); //定位到第二行

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

LcdWriteData(str[i]);

}

}

void delay(u16 i){

while(i--);

}

/\*

P17->H1

P16->H2

P15->H3

P14->H4

P13->L1

P12->L2

P11->L3

P10->L4

\*/

#define GPIO\_KEY P1 // 0000 0000

#define GPIO\_BUTTON P3 //独立按键使用P3

sbit K1 = P3^0; sbit K2 = P3^1;

sbit K3 = P3^2; sbit K4 = P3^3;

sbit K5 = P3^4; sbit K6 = P3^5;

sbit K7 = P3^6; sbit K8 = P3^7;

u8 isNum(u8 num){

if((num>=0)&&(num<=9))

return 1;

return 0;

}

u8 KeyDown(void)

{

u8 KeyValue= 127;

char adelay=0;

GPIO\_KEY=0x0f;

if(GPIO\_KEY!=0x0f)//读取按键是否按下

{

delay(1000);//延时10ms进行消抖

if(GPIO\_KEY!=0x0f)//再次检测键盘是否按下

{

//测试列

GPIO\_KEY=0X0F;

switch(GPIO\_KEY)

{

case(0X07): KeyValue=0;break;

case(0X0b): KeyValue=1;break;

case(0X0d): KeyValue=2;break;

case(0X0e): KeyValue=3;break;

}

//测试行

GPIO\_KEY=0XF0;

switch(GPIO\_KEY)

{

case(0X70): KeyValue=KeyValue;break;

case(0Xb0): KeyValue=KeyValue+4;break;

case(0Xd0): KeyValue=KeyValue+8;break;

case(0Xe0): KeyValue=KeyValue+12;break;

}

}

}

//delay(1000);

while((adelay<50)&&(GPIO\_KEY!=0xf0)) //检测按键松手检测

{

delay(160);

adelay++;

}

if(KeyValue == 15)

return 18;

return KeyValue;

}

/\*

从独立按键输入字符

\*/

u8 keypros(){

GPIO\_BUTTON = 0xff;

delay(1000);

if(K1 == 0){

while(!K1);

return 10; //'+'

}

if(K2 == 0){

while(!K2);

return 11;

}

if(K3 == 0){

while(!K3);

return 12;

}

if(K4 == 0){

while(!K4);

return 13;

}

if(K5 == 0){

while(!K5);

return 14;

}

if(K6 == 0){

while(!K6);

return 15;

}

if(K7 == 0){

while(!K7);

return 16;

}

if(K8 == 0){

while(!K8);

return 17;

}

return 127;

}

u8 Getch(){

u8 op = 127;

while(op==127){

op = keypros();

if(op!=127)

return op ;

op = KeyDown();

if(op!=127){

return op;

}

}

return 127;

}

/\* if 不是小数

// 更新小数flag

// update

if 不是小数

\*10 + nun

是小数

根据小数计算

//

\*/

void clear(){

// LCD1602\_E = 0;

LcdWriteCom(0x01);

initA();

initB();

initC();

count = 0;

State = 0;

stackpoint = -1;

pastpoint = -1;

}

void function\_S0(){

u8 num = Getch();

if(num == 17){

LcdWriteData(Outputchar[num]);

printans(a);

}

else if(num == 18){

clear();

}

else if(num == 15){

if(count>0){

PopStack();

LcdWriteData(Outputchar[num]);

count --;

}

}

else if(num == 16){

if(flag\_a == 0){

flag\_a = 1;

LcdWriteData(Outputchar[num]);

}

}

else {

if(isNum(num)==1){

if(flag\_a == 0){

a = a\*10 +num;

}

else{

a += num\*(float)pow(0.1,flag\_a);

flag\_a ++;

}

}

else if(num == 14){

pushState();

State = 0;

count++;

}

else if(num == 10){ // + -

add\_sub\_flag = 1;

State = 1;

}

else if(num == 11){

add\_sub\_flag = 0 ;

State = 1;

}

else if(num ==12 ){

State = 5;

multi\_div\_flag = 1;

}// \* /

else if(num == 13){

State =5 ;

multi\_div\_flag = 0;

}

LcdWriteData(Outputchar[num]);

}

}

void function\_S1(){

u8 num = Getch();

initB();

if(isNum(num)==1){

b = num;

State = 2;

LcdWriteData(Outputchar[num]);

}

else if(num == 14){

pushState();

count++;

State = 0;

LcdWriteData(Outputchar[num]);

}

else if(num == 18){

clear();

}

}

void function\_S2(){

u8 num = Getch();

if(num==17){

State = 0;

if(add\_sub\_flag ==1){

a = a+b;

}

else

a = a-b;

LcdWriteData(Outputchar[num]);

printans(a);

}

else if(num == 18){

clear();

}

else if(num == 16){

if(flag\_b == 0){

flag\_b = 1;

LcdWriteData(Outputchar[num]);

}

}

else if(num == 15){

if(count>0){

if(add\_sub\_flag == 1)

a = a+b ;

else a = a-b;

PopStack();

LcdWriteData(Outputchar[num]);

count --;

}

}

else{

if(isNum(num) == 1){

if(flag\_b == 0){

b = b\*10 +num;

}

else{

b += num\*(float)pow(0.1,flag\_b);

flag\_b ++;

}

}

else if (num== 12){ //\*

State = 3;

multi\_div\_flag = 1;

}

else if (num == 13){ // /

State = 3;

multi\_div\_flag = 0;

}

else if (num == 10){

if(add\_sub\_flag ==1){

a = a+b;

}

else

a = a-b;

State = 1;

add\_sub\_flag = 1;

}

else if (num == 11){

if(add\_sub\_flag ==1){

a = a+b;

}

else

a = a-b;

State = 1;

add\_sub\_flag = 0;

}

LcdWriteData(Outputchar[num]);

}

}

void function\_S3(){

u8 num = Getch();

initC();

if(isNum(num)==1){

c = num;

State = 4;

LcdWriteData(Outputchar[num]);

}

else if(num == 14){

pushState();

State = 0;

count++;

LcdWriteData(Outputchar[num]);

}

else if(num == 18){

clear();

}

}

void function\_S4(){

u8 num = Getch();

if(isNum(num)==1){

if(flag\_c == 0){

c = c\*10 +num;

}

else{

c += num\*(float)pow(0.1,flag\_c);

flag\_c ++;

}

}

else if(num == 16){

if(flag\_c == 0){

flag\_c = 1;

LcdWriteData(Outputchar[num]);

}

}

else if(num == 15){

if(count > 0){

if(add\_sub\_flag == 1){

if(multi\_div\_flag ==1)

a= a+b\*c;

else

a= a+b/c;

}

else{

if(multi\_div\_flag ==1)

a= a-b\*c;

else

a= a-b/c;

}

PopStack();

LcdWriteData(Outputchar[num]);

count --;

}

}

else if(num == 10){ // +

if(add\_sub\_flag == 1){

if(multi\_div\_flag ==1)

a= a+b\*c;

else

a= a+b/c;

}

else{

if(multi\_div\_flag ==1)

a= a-b\*c;

else

a= a-b/c;

}

add\_sub\_flag = 1;

State = 1;

}

else if(num == 11){

if(add\_sub\_flag == 1){

if(multi\_div\_flag ==1)

a= a+b\*c;

else

a= a+b/c;

}

else{

if(multi\_div\_flag ==1)

a= a-b\*c;

else

a= a-b/c;

}

add\_sub\_flag = 0;

State = 1;

}

else if(num == 12){ //\*

if(multi\_div\_flag == 1){

b = b\*c ;

}

else

b = b/c;

State =3;

multi\_div\_flag = 1;

}

else if(num == 13){ // /

if(multi\_div\_flag == 1){

b = b\*c ;

}

else

b = b/c;

State =3;

multi\_div\_flag = 0;

}

else if(num == 17){

if(add\_sub\_flag == 1){

if(multi\_div\_flag ==1)

a= a+b\*c;

else

a= a+b/c;

}

else{

if(multi\_div\_flag ==1)

a= a-b\*c;

else

a= a-b/c;

}

add\_sub\_flag = 0;

State = 0;

}

if(num == 18){

clear();

}

else if(num==17){

LcdWriteData(Outputchar[num]);

printans(a);

}

else if((num !=15)&&(num!=16)){

LcdWriteData(Outputchar[num]);

}

}

void function\_S5(){

u8 num = Getch();

initC();

if(isNum(num)==1){

c = num;

State = 6;

LcdWriteData(Outputchar[num]);

}

else if(num == 14){

pushState();

State = 0;

LcdWriteData(Outputchar[num]);

count ++;

}

else if(num==18){

clear();

}

}

void function\_S6(){

u8 num = Getch();

if(isNum(num)==1){

if(flag\_c == 0){

c = c\*10 +num;

}

else{

c += num\*(float)pow(0.1,flag\_c);

flag\_c ++;

}

}

else if(num == 16){

if(flag\_c== 0){

flag\_c = 1;

LcdWriteData(Outputchar[num]);

}

}

else if(num == 10){ // +

if(multi\_div\_flag==1){

a = a\*c;

State = 1;

add\_sub\_flag = 1;

}

else{

a = a/c;

State = 1;

add\_sub\_flag = 1;

}

}

else if (num == 11){ // -

if(multi\_div\_flag==1){

a = a\*c;

State = 1;

add\_sub\_flag = 0;

}

else{

a = a/c;

State = 1;

add\_sub\_flag = 0;

}

}

else if(num ==12){ // \*

if(multi\_div\_flag==1){

a = a\*c;

multi\_div\_flag = 1;

}

else{

a = a/c;

multi\_div\_flag = 1;

}

State = 5;

}

else if (num == 13){ // /

if(multi\_div\_flag==1){

a = a\*c;

multi\_div\_flag = 0;

}

else{

a = a/c;

multi\_div\_flag = 0;

}

State = 5;

}

else if(num == 17){ // ==

State = 0;

if(multi\_div\_flag==1){

a = a\*c;

}

else{

a = a/c;

}

}

else if(num == 15){

if(count >0){

if(multi\_div\_flag==1){

a = a\*c;

}

else{

a = a/c;

}

PopStack();

LcdWriteData(Outputchar[num]);

count--;

}

}

if(num == 18){

clear();

}

else if(num==17){

LcdWriteData(Outputchar[num]);

printans(a);

}

else if((num !=15) && (num !=16)){ //不为) , .

LcdWriteData(Outputchar[num]);

}

}

//todo : a 的flag没有更新过

void main(){

LcdInit();

while(1){

switch(State){

case 0:

function\_S0();

break;

case 1:

function\_S1();

break;

case 2:

function\_S2();

break;

case 3:

function\_S3();

break;

case 4:

function\_S4();

break;

case 5:

function\_S5();

break;

case 6:

function\_S6();

break;

}

}

}

相关使用介绍可参照https://github.com/Sun2018421/Hardware-integrated-design

# 四、设计体会与建议

经过两周的硬件综合实践，完成了简易计算器的设计。在这个过程中学习到了之前很多忽略的知识，在有限的RAM和FLASH上写程序让我对每一个字节，每一个地址的理解更加深入。通过查看Lcd1602的数据手册让我对时序有了新的理解。在调试设计中有着各自各式各样的bug，比如按键抖动，运算符优先级等等让我不断的发现并解决一些隐蔽的问题。通过查阅了解状态图来完成整个计算器的设计拓展了我的新思路，对程序设计提供了新的思考角度。不仅感叹图灵机思维的巧妙性。在老师的讲解下对51单片机这一鼻祖级的单片机掌握的更加熟练。在疫情期间，课程圆满的结束感谢老师的教学付出。