PROGRAMA QUE REALIZA LA MULTIPLICACION DE MATRIZ 4X4 POR VECTOR 4X1 EN LENGUAJE ENSAMBLADOR (Revisión de funcionamiento con EMU)

Alumno: Rubén Alvarez Martínez.

Puntos importantes:

De la dirección 0 a 100 estará el programa.

De la dirección 0600:0001-10 la matriz (4x4).

De la dirección 0600:0041-44 el vector (4x1).

De la dirección 0600:0061-70 el resultado de las multiplicaciones por cada elemento.

De la dirección 0600:0081-84 la matriz resultante o vector (4x1).

Los valores de cada elemento en los arreglos (matrices o vectores) son de 1 byte (8 bits).

MOV AX,600H

MOV DS,AX

MOV SS,AX

MOV AX,01H

MOV SI,AX PREPARA LOS DIRECCIONAMIENTOS

MOV AX,41H

MOV DI,AX

MOV AX,61H

MOV BP,AX

inicio

MOV AH,00H

MOV AL,[SI]

MOV BL,[DI] INICIA EL LLAMADO A CADA UNO DE LOS ELEMENTOS DE LAS MATRICES

MOV CL,BL

MOV BL,AL

CALL multi

JMP aqui

RET

multi

CMP CL,00H

JZ done

CMP BL,00H

JZ done

DEC CL LLAMA A FUNCION Y REALIZA LA MULTIPLICACION MEDIANTE SUMAS

cont

JZ mdone

ADC AX,BX

DEC CL

JMP cont

mdone

RET

done

CONTINUA LA FUNCION

MOV AL,00H

JMP mdone

aqui

MOV [BP],AX

CMP AL,BL

INC SI

INC DI EVALUA SI LA MULTIPLICACION ESTA COMPLETA Y TAMBIEN EVALUA SI

INC BP TODAS LAS MULTIPLICACIONES SE LLEVARON A CABO

INC BP

CMP DI,45H

JZ fm2

JMP inicio

fm2

MOV AH,00H

MOV AL,41H

MOV DI,AX

CMP AL,BL

CMP BP,81H

JZ salir

JMP inicio

salir

MOV AX,61H

MOV BP,AX

MOV DX,00H

MOV AL,03H

MOV CL,AL PREPARA LOS REGISTROS PARA DIRECCIONAR A LOS RESULTADOS DE

MOV AL,81H CADA MULTIPLICACION

MOV SI,AX

MOV AL,04H

MOV BL,AL

csuma2

MOV AX,[BP]

csuma INICIA LA SUMA DE LOS RESULTADOS DE LAS MULTIPLICACIONES

INC BP

INC BP

ADD AX,[BP]

JC carry CHECA SI EL RESULTADO DE LA SUMA ES MAYOR A 4 BYTES, SINO

JMP conti CONTINUA CON LA SUMA

carry

INC DX

JMP conti SIMULA EL RESULTADO DE UNA SUMA MAYOR DE 4 BYTES

conti

DEC CL

JZ fsuma CHECA SI LA SUMA ACTUAL HA SIDO COMPLETADA

JMP csuma

fsuma

MOV [SI],AX

CMP AL,01H

INC SI

INC SI

MOV [SI],DX

INC SI SE PREPARA PARA LA SIGUIENTE SUMA

MOV AL,03H

MOV CL,AL

INC BP

INC BP

MOV DL,00H

DEC BL

JZ final

JMP csuma2 CHECA SI TODAS LAS SUMAS SE REALIZARON

final

HLT

PROGRAMA QUE REALIZA LA MULTIPLICACION DE MATRIZ 4X4 POR VECTOR 4X1 EN LENGUAJE C (Revisión de funcionamiento con EMU)

Alumno: Rubén Alvarez Martínez.

Puntos importantes:

De la dirección 0 a 100 estará el programa.

De la dirección 0600:0001-10 la matriz (4x4).

De la dirección 0600:0041-44 el vector (4x1).

De la dirección 0600:0061-70 el resultado de las multiplicaciones por cada elemento.

De la dirección 0600:0081-84 la matriz resultante o vector (4x1).

Los valores de cada elemento en los arreglos (matrices o vectores) son de 1 byte (8 bits).

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include <math.h>

#include <ctype.h>

#define MOV 242

#define ADD 201

#define ADC 200

#define HLT 232

#define INC 218

#define DEC 204

#define CMP 224

#define JMP 231

#define JC 141

#define JZ 164

#define CALL 284 DEFINE CONSTANTES Y LIBRERIAS

#define RET 235

#define NUMEROYNUMERO 1

#define NUMEROYLETRA 2

#define NUMEROYCORCHETE 3

#define LETRAYNUMERO 4

#define LETRAYLETRA 5

#define LETRAYCORCHETE 6

#define CORCHETEYNUMERO 7

#define CORCHETEYLETRA 8

#define CORCHETEYCORCHETE 9

#define CEROYNUMERO 10

#define CEROYLETRA 11

#define CEROYCORCHETE 12

#define CEROYCERO 13

#define TOTALOP 5

typedef struct{

long int AH;

long int AL;

long int BH; ESTRUCTURA DE REGISTROS

long int BL;

long int CH;

long int CL;

long int DH;

long int DL;

}general\_registers8;

typedef struct{

long int AX;

long int BX;

long int CX;

long int DX;

}general\_registers16;

typedef struct{

long int SI;

long int DI;

}index\_registers;

typedef struct{

long int BP;

long int SP; ESTRUCTURAS DE REGISTROS Y DE MEMORIA

}pointer\_registers;

typedef struct{

long int IP;

\_Bool OF;

\_Bool DF;

\_Bool IF;

\_Bool TF;

\_Bool SF;

\_Bool ZF;

\_Bool AF;

\_Bool PF;

\_Bool CF;

}flag\_registers;

typedef struct{

int \*memoria[65535];

}memory;

typedef struct{

int DS;

int SS;

int CS;

int ES;

}segment\_registers;

typedef struct{

int IP;

}special\_registers;

void limpiezaderegistros();

int operando1(int a,int b);

int operando2(int a,int b); DECLARO LAS FUNCIONES

int operando3(int a,int b);

void pruebadeletras();

int pruebasaltodelinea();

void impresion();

int corchetesop1(int a, int b);

int corchetesop2(int a, int b);

int corchetesop3(int a, int b);

int sumaop1(int a,int b);

int sumaop2(int a,int b);

int sumaop3(int a,int b);

int detectaopcode(int a);

void direccionamiento();

void seleccionafuncion(int a);

void operandos(int a,int b);

void funcionmov(int a);

void funcionadd(int a);

void funcionadc(int a); DECLARO LAS FUNCIONES

void funcionhlt();

void funcioncmp();

void funciondec();

void funcioninc();

void funcionjc();

void funcionjz();

void funcionjmp();

void funcioncall();

void funcionret();

int hextodecop2(int a);

int hextodecop2c(int a);

void reg16to8();

void convertidorhextodec();

void pantallaprincipal();

int i,j;

int prueba; //para guardar caracter de lectura de archivo

int cont\_comas; //cuenta comas

int salto; // avisa cuando exista salto de linea

int c; //guarda valores de contadores en funciones

int bandera; //avisa cuando detener el programa

int cont\_espacios; //cuenta espacios

int sumaopcode; //detecta el codigo de operacion mediante suma de caracteres

int suma; //

int clave; //para ver modo de direccionamiento

int contador; // determina la linea que va ejecutándose DECLARO VARIABLES

int valor1; //indica el valor absoluto de op1 GLOBALES

int valor2; // indica el valor absoluto de op2.

int deci, coeficiente, ctr; // de decimal a hexadecimal

char hx[256];

int test;

char \*stop;

char p;

int compara; //para funcion compara

int funcion; //bandera de la funcion compara

int eti; // ayuda para encontrar etiqueta y destino

int eti\_compara; //ayuda para encontrar etiqueta y destino

int h; //indice de etiqueta y destino.

int etiqueta[200];

int destino[200]; //destino para funcion jmp o call

int opcode[200][8];

char op1[200][9];

char op2[200][9];

char op3[200][9];

char opj1[200][9];

DECLARO VARIABLES GLOBALES

general\_registers16 registros16;

general\_registers8 registros8;

memory memoriaext,memoriaext1;

segment\_registers segmentos;

flag\_registers banderas;

index\_registers indices;

pointer\_registers punteros;

special\_registers especiales;

int main()

{

limpiezaderegistros();

FILE \* archivo;

archivo = fopen ("matrices.txt","r");

if(!archivo)

printf("Error al abrir el archivo");

while(feof(archivo)==0){

prueba=getc(archivo);

aqui2:

if (prueba==10){ //salto de linea

salto=pruebasaltodelinea();

}

else if (prueba==44){ //coma

cont\_comas=cont\_comas+1;

c=0;

}

else if (prueba==32){ // espacios

cont\_espacios=cont\_espacios+1;

} LECTURA DEL ARCHIVO

else if(prueba>=65 && prueba<=90){ //letras mayúsculas EXTERNO Y CONDICIONES

if(cont\_espacios==0) PARA INTERPRETARLO

pruebadeletras();

else if (cont\_espacios>0){

if (cont\_comas==0){

c=operando1(salto,c); //operando1 letras

if (c==2) //modificaciones para etiquetas!!!!!!

c=0;

}

else if (cont\_comas==1){

c=operando2(salto,c); //operando2 letras

if (c==2)

c=0;

}

else{

c=operando3(salto,c); //operando3 letras

if (c==2)

c=0;

}

}

}

else if (prueba>=48 && prueba<=57 && (cont\_espacios==0)){ //etiquetas (numeros)

pruebadeletras();

destino[i]=destino[i]+prueba;

// printf("Destino %d\n",destino[i]);

//h=i;

}

else if (prueba>=97 && prueba<=122){ //etiquetas (letras minusculas)

if (cont\_espacios==0 && cont\_comas==0){ //modificaciones para etiquetas

pruebadeletras();

destino[i]=destino[i]+prueba;

//printf("Destino %d\n",destino[i]);

//h=h+1;

//eti=eti+1;

}

else if (cont\_espacios>0 && cont\_comas==0){

//etiqueta[h1]=prueba; // modificaciones para etiqutas

//h1=h1+1;

etiqueta[h]=etiqueta[h]+prueba;

c=operando1(salto,c); //

if (c==8) CONT.

c=0; COMPA-

//eti=eti+1; -RACION

}

else c=operando2(salto,c);

//if (eti\_compara==eti+1)

//printf("ETI %d\n",eti);

//printf("Eti\_compara %d\n",eti\_compara);

//printf("H %d\n",h);

}

else if (prueba>=97 && prueba<=102){ //letras minusculas (a-f)

if (cont\_comas==0){

c=operando1(salto,c); //operando1

}

else if (cont\_comas==1){

c=operando2(salto,c); //operando2

}

}

else if (prueba>=48 && prueba<=57){ // numeros 0-9

if (cont\_comas==0){

c=operando1(salto,c); //operando1

}

else if (cont\_comas==1){

c=operando2(salto,c); //operando2

/\*printf("Salto: %d C: %d\n" ,salto,c);

p=op2[salto][c];

if (isxdigit(p))

printf("es hexadecimal\n");\*/

/\*printf("Valor de P:%d\n",p);

char sh[100];

sh[100]=p[0];

test = strtol(sh,&stop,16);

//op2[salto][c]=test;

printf("Test:%d\n",test);\*/

}

else

c=operando3(salto,c); //operando3

}

else if (prueba==91){ //corchetes

bandera=1;

aqui: while (bandera){

prueba=getc(archivo);

if (prueba==44 || prueba==10){ CONTINUA COMPARACION Y

bandera=0; LECTURA

goto aqui;

}

if (cont\_comas==0){ //operando 1 y corchetes

c=corchetesop1(salto, c);

}

else if (cont\_comas==1){ //operando 2 y corchetes

c=corchetesop2(salto, c);

}

else { //operando 3 y corchetes

c=corchetesop3(salto,c);

}

}

c=0; //elimina el corrimiento generado en las funciones.

goto aqui2;

}

}

impresion();

printf("\n");

system("pause");

system("cls");

contador=0;

convertidorhextodec(); SE IMPRIME SELECCION Y

//impresion(); SEPARACION

//printf("\n");

//system("pause");

//system("cls");

punteros.SP=65534;

while(suma!=232){

suma=0;

c=detectaopcode(c);

direccionamiento(contador); SE LLAMA A LAS FUNCIONES

operandos(clave,contador); PARA ANALIZAR CODIGO

seleccionafuncion(clave); (PRGRAMA PRINCIPAL)

reg16to8();

pantallaprincipal();

contador=contador+1;

segmentos.CS=contador;

especiales.IP=contador+1; PROGRAMA PRINCIPAL

clave=0;

system("pause");

system("CLS");

}

return 0;

}

//////////////////////////////////////Funciones:

void limpiezaderegistros(){

int a,b;

eti=eti\_compara=0;

i=j=cont\_comas=0;

a=0;

b=0;

cont\_espacios=0;

for (a=0;a<200;a++){

for (b=0;b<2;b++){ SE LIMPIAN LOS REGISTROS

op1[a][b]='0';

op2[a][b]='0';

op3[a][b]='0';

}

}

for (a=0;a<150;a++){

memoriaext1.memoria[a]="00";

}

for (a=0;a<150;a++){

destino[a]=0;

}

}

void pruebadeletras(){

opcode[i][j]=prueba; SE GUARDAN LOS OPCODES

j=j+1;

}

int pruebasaltodelinea(){

cont\_comas=0;

j=0;

i=i+1; //siguiente renglon opcode

cont\_espacios=0; AL CREARSE UN SALTO DE LINEA SE EJECUTA

cont\_comas=0;

c=0;

//eti\_compara=eti;

//eti=0;

return i;

}

int operando1(int a,int b){

op1[a][b]=prueba;

opj1[a][b]=op1[a][b];

b=b+1;

/\*if (b==4){

b=0;

}\*/

return b;

}

int operando2(int a, int b){

op2[a][b]=prueba;

b=b+1;

/\* if (b==4){

b=0;

}\*/

return b;

}

int operando3(int a, int b){

op3[a][b]=prueba;

b=b+1;

/\*if (b==4){

b=0;

}\*/ FUNCIONES DONDE SE ANIDAN

return b; LOS VALORES DESPUES DE LA

} SELECCION Y ANALISIS DEL

ARCHIVO.

int corchetesop1(int a, int b){

if (b==0)

op1[a][b]=91;

op1[a][b+1]=prueba;

b=b+1;

return b;

}

int corchetesop2(int a, int b){

if (b==0)

op2[a][b]=91;

op2[a][b+1]=prueba;

b=b+1;

return b;

}

int corchetesop3(int a, int b){

if (b==0)

op3[a][b]=91;

op3[a][b+1]=prueba;

b=b+1;

return b;

}

void impresion(){

int a,b;

a=0;

b=0;

printf("Operation Code\t\tOP 1\t\t OP 2\n");

for (a=0;a<i+1;a++){

printf("\nLinea #:%d ",a);

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

printf("%c",opcode[a][b]);

}

printf("\t");

for (b=0;b<9;b++){ IMPRIME OPCODES Y CODIGOS DE OPERACION

printf("%c",op1[a][b]);

}

printf("\t ");

for (b=0;b<9;b++){

printf("%c",op2[a][b]);

}/\*

printf("\t ");

for (b=0;b<5;b++){

printf("%c",op3[a][b]);

}\*/

}

}

int detectaopcode(int a){

int i;

for(i=0;i<3;i++){

suma+=opcode[a][i];

}

i=3;

//printf("OPCODE %d\n",opcode[a][i]);

while(opcode[a][i]!=0){ REALIZA UNA SUMA DE LAS LETRAS DEL OPCODE

if(opcode[a][i]!=0)

suma+=opcode[a][i];

i++;

}

a=a+1;

return a;

}

void seleccionafuncion(int a){

int i;

//printf("%d \n",suma);

//printf("Valor1 %d\n",valor1);

switch(suma){ DEPENDIENDO DE LA SUMA, DETECTA EL OPCODE

case MOV:

funcionmov(a);

break;

case ADD:

funcionadd(a);

break;

case ADC:

funcionadc(a);

break;

case HLT:

funcionhlt();

break;

case INC:

funcioninc();

break;

case DEC:

funciondec();

break;

case CMP:

funcioncmp(); DETERMINA LA FUNCION DEPENDIENDO DE LA SUMA

break; Y MANDA A LLAMAR LA FUNCION.

case JMP:

funcionjmp();

break;

case JC:

funcionjc();

break;

case JZ:

funcionjz();

break;

case CALL:

funcioncall();

break;

case RET:

funcionret();

break;

default:

printf("Etiquieta (destino): ");

for (i=0;(opcode[contador][i]!=0);i++){

printf("%c",opcode[contador][i]); DETERMINA SI ES ETIQUETA

}

break;

}

}

void direccionamiento(int a){

int b;

b=0;

if (op2[a][b]==48 && op2[a][b+1]==48){ //checa si op2 existe (00)

if (op1[a][b]>=49 && op1[a][b]<=57)

clave=CEROYNUMERO;

else if (op1[a][b]>=65 && op1[a][b]<=90)

clave=CEROYLETRA;

else if (op1[a][b]==91)

clave=CEROYCORCHETE;

else

clave=CEROYCERO; // no existen operandos. A PRUEBA\*\*\*\*\*\*\*\*\*

//realiza accion.

}

//checa la posicion b=0

else if (op2[a][b]>=48 && op2[a][b]<=57){ //checa si numero

//checa si numero, letra o corchete para op1.

if (op1[a][b]>=48 && op1[a][b]<=57)

clave=NUMEROYNUMERO;

//asigna clave1

else if (op1[a][b]>=65 && op1[a][b]<=90)

clave=NUMEROYLETRA;

else

clave=NUMEROYCORCHETE;

//realiza accion.

}

FUNCION QUE DETERMINA

//checa la posicion b=0 EL TIPO DE

else if (op2[a][b]>=65 && op2[a][b]<=90){ //checa si letra DIRECCIONAMIENTO

//checa si numero, letra o corchete para op1.

if (op1[a][b]>=48 && op1[a][b]<=57)

clave=LETRAYNUMERO;

else if (op1[a][b]>=65 && op1[a][b]<=90)

clave=LETRAYLETRA;

else

clave=LETRAYCORCHETE;

//realiza accion.

}

//checa la posicion b=0

else if (op2[a][b]==91){ //corchetes

//checa si numero, letra para op1.

if (op1[a][b]>=48 && op1[a][b]<=57)

clave=CORCHETEYNUMERO;

else if (op1[a][b]>=65 && op1[a][b]<=90)

clave=CORCHETEYLETRA;

//else //no existe este modo de direccionamiento.

// clave=CORCHETEYCORCHETE;

//realiza accion.

}

//printf("Clave %d\n",clave);

}

void operandos(int a, int b){

int multiplicador,multiplicacion, cuenta\_sumas,cuenta,k;

valor1=0;

valor2=0;

cuenta=0;

//printf("A %d\n",a);

cuenta\_sumas=0;

switch (a){

case NUMEROYNUMERO:

//

while(k!=72){

cuenta=cuenta+1;

k=op2[b][cuenta];

}

multiplicador=pow(10,cuenta-1);

if(multiplicador==99){

multiplicador=multiplicador+1;

}

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

multiplicacion=(op2[b][i]-48)\*multiplicador;

valor2=valor2+multiplicacion;

multiplicador=(multiplicador)/10;

}

cuenta=0;

while(k!=72){

cuenta=cuenta+1; EXTRAE EL VALOR NUMERO Y REGISTRO

k=op1[b][cuenta]; DE LOS OPERANDOS 1 Y 2

} DEPENDIENDO DEL TIPO DE

multiplicador=pow(10,cuenta-1); DIRECCIONAMIENTO

if(multiplicador==99){

multiplicador=multiplicador+1;

}

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

multiplicacion=(op1[b][i]-48)\*multiplicador;

valor1=valor1+multiplicacion;

multiplicador=(multiplicador)/10;

}

break;

case NUMEROYLETRA:

//

while(k!=72){

cuenta=cuenta+1;

k=op2[b][cuenta];

}

multiplicador=pow(10,cuenta-1);

if(multiplicador==9999)

multiplicador=multiplicador+1;

if(multiplicador==99){

multiplicador=multiplicador+1;

}

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

multiplicacion=(op2[b][i]-48)\*multiplicador;

//printf("Multiplicador: %d",multiplicador);

valor2=valor2+multiplicacion;

multiplicador=(multiplicador)/10;

//printf("Valor 2: %d\n",valor2);

}

for (i=0;i<2;i++){

valor1=(op1[b][i])+valor1;

}

break;

case NUMEROYCORCHETE:

//

while(k!=72){

cuenta=cuenta+1;

k=op2[b][cuenta];

}

multiplicador=pow(10,cuenta-1);

if(multiplicador==9999)

multiplicador=multiplicador+1;

if(multiplicador==99){

multiplicador=multiplicador+1;

}

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

multiplicacion=(op2[b][i]-48)\*multiplicador;

valor2=valor2+multiplicacion;

multiplicador=(multiplicador)/10;

}

for (i=0;i<13;i++){ EXTRAE EL VALOR NUMERO Y REGISTRO

if (op1[b][i]=='+'){ DE LOS OPERANDOS 1 Y 2

cuenta\_sumas=cuenta\_sumas+1; DEPENDIENDO DEL TIPO DE

} DIRECCIONAMIENTO

}

if (cuenta\_sumas==2){

for (i=0;i<13;i++){

valor1=op1[b][i]+valor1;

}

valor1=valor1-((43\*cuenta\_sumas)+184);

}

else if (cuenta\_sumas==1){

for (i=0;i<7;i++){

valor1=op1[b][i]+valor1;

}

valor1=valor1-((43\*cuenta\_sumas)+184);

}

else if (cuenta\_sumas==0){

while(k!=72){

cuenta=cuenta+1;

k=op1[b][cuenta];

}

multiplicador=pow(10,cuenta-2);

if(multiplicador==99){

multiplicador=multiplicador+1;

}

for (i=1;i<cuenta;i++){

multiplicacion=(op1[b][i]-48)\*multiplicador;

valor1=valor1+multiplicacion;

multiplicador=(multiplicador)/10;

}

}

break;

case LETRAYNUMERO:

//

for (i=0;i<2;i++){

valor2=op2[b][i]+valor2;

}

while(k!=72){

cuenta=cuenta+1;

k=op1[b][cuenta];

}

multiplicador=pow(10,cuenta-1);

if(multiplicador==9999)

multiplicador=multiplicador+1;

if(multiplicador==99){

multiplicador=multiplicador+1;

}

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

multiplicacion=(op1[b][i]-48)\*multiplicador;

valor1=valor1+multiplicacion;

multiplicador=(multiplicador)/10;

}

break CONTINUA EXTRACCION

case LETRAYLETRA:

//

for (i=0;i<2;i++){

valor2=op2[b][i]+valor2;

}

for (i=0;i<2;i++){

valor1=op1[b][i]+valor1;

}

break;

case LETRAYCORCHETE:

//

for (i=0;i<2;i++){

valor2=op2[b][i]+valor2;

}

for (i=0;i<13;i++){

if (op1[b][i]=='+'){

cuenta\_sumas=cuenta\_sumas+1;

}

}

if (cuenta\_sumas==2){

for (i=0;i<13;i++){

valor1=op1[b][i]+valor1;

}

valor1=valor1-((43\*cuenta\_sumas)+184);

}

else if (cuenta\_sumas==1){

for (i=0;i<7;i++){

valor1=op1[b][i]+valor1;

}

valor1=valor1-((43\*cuenta\_sumas)+184);

}

else if (cuenta\_sumas==0){

while(k!=72 && k!=93){

cuenta=cuenta+1;

k=op1[b][cuenta];

// printf("Cuenta %d\n",cuenta);

}

multiplicador=pow(10,cuenta-2);

if(multiplicador==99){

multiplicador=multiplicador+1;

}

for (i=1;i<cuenta;i++){

multiplicacion=(op1[b][i]-48)\*multiplicador;

valor1=valor1+multiplicacion;

multiplicador=(multiplicador)/10;

// printf("valor1 %d",valor1);

}

}

break;

case CORCHETEYNUMERO:

//

for (i=0;i<13;i++){

if (op2[b][i]=='+'){

cuenta\_sumas=cuenta\_sumas+1; CONTINUA EXTRACCION

}

}

if (cuenta\_sumas==2){

for (i=0;i<13;i++){

valor2=op2[b][i]+valor2;

}

valor2=valor2-((43\*cuenta\_sumas)+184);

}

else if (cuenta\_sumas==1){

for (i=0;i<7;i++){

valor2=op2[b][i]+valor2;

}

valor2=valor2-((43\*cuenta\_sumas)+184);

}

else if (cuenta\_sumas==0){

while(k!=72){

cuenta=cuenta+1;

k=op2[b][cuenta];

}

multiplicador=pow(10,cuenta-2);

if(multiplicador==99){

multiplicador=multiplicador+1;

}

for (i=1;i<cuenta;i++){

multiplicacion=(op2[b][i]-48)\*multiplicador;

valor2=valor2+multiplicacion;

multiplicador=(multiplicador)/10;

}

}

while(k!=72){

cuenta=cuenta+1;

k=op1[b][cuenta];

}

multiplicador=pow(10,cuenta-1);

if(multiplicador==9999)

multiplicador=multiplicador+1;

if(multiplicador==99){

multiplicador=multiplicador+1;

}

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

multiplicacion=(op1[b][i]-48)\*multiplicador;

valor1=valor1+multiplicacion;

multiplicador=(multiplicador)/10;

}

break;

case CORCHETEYLETRA:

//

for (i=0;i<13;i++){

if (op2[b][i]=='+'){

cuenta\_sumas=cuenta\_sumas+1; CONTINUA EXTRACCION

}

}

if (cuenta\_sumas==2){

for (i=0;i<13;i++){

valor2=op2[b][i]+valor2;

}

valor2=valor2-((43\*cuenta\_sumas)+184);

}

else if (cuenta\_sumas==1){

for (i=0;i<7;i++){

valor2=op2[b][i]+valor2;

}

valor2=valor2-((43\*cuenta\_sumas)+184);

}

else if (cuenta\_sumas==0){

while(k!=72 && k!=93){

cuenta=cuenta+1;

k=op2[b][cuenta];

}

multiplicador=pow(10,cuenta-2);

if(multiplicador==99){

multiplicador=multiplicador+1;

}

for (i=1;i<cuenta;i++){

multiplicacion=(op2[b][i]-48)\*multiplicador;

valor2=valor2+multiplicacion;

multiplicador=(multiplicador)/10;

}

}

for (i=0;i<2;i++){

valor1=op1[b][i]+valor1;

}

break;

/\*case CORCHETEYCORCHETE:

//

break;\*/

case CEROYNUMERO:

//

while(k!=72){

cuenta=cuenta+1;

k=op1[b][cuenta];

}

multiplicador=pow(10,cuenta-1);

if(multiplicador==9999)

multiplicador=multiplicador+1;

if(multiplicador==99){

multiplicador=multiplicador+1;

}

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

multiplicacion=(op1[b][i]-48)\*multiplicador;

valor1=valor1+multiplicacion;

multiplicador=(multiplicador)/10;

}

break;

case CEROYLETRA:

// CONTINUA EXTRACCION

for (i=0;i<2;i++){

valor1=op1[b][i]+valor1;

}

valor2=0;

break;

case CEROYCORCHETE:

//

for (i=0;i<13;i++){

if (op1[b][i]=='+'){

cuenta\_sumas=cuenta\_sumas+1;

}

}

if (cuenta\_sumas==2){

for (i=0;i<13;i++){

valor1=op1[b][i]+valor1;

}

valor1=valor1-((43\*cuenta\_sumas)+184);

}

else if (cuenta\_sumas==1){

for (i=0;i<7;i++){

valor1=op1[b][i]+valor1;

}

valor1=valor1-((43\*cuenta\_sumas)+184);

}

else if (cuenta\_sumas==0){

while(k!=72){

cuenta=cuenta+1;

k=op1[b][cuenta];

}

multiplicador=pow(10,cuenta-2);

if(multiplicador==99){

multiplicador=multiplicador+1;

}

for (i=1;i<cuenta;i++){

multiplicacion=(op1[b][i]-48)\*multiplicador;

valor1=valor1+multiplicacion;

multiplicador=(multiplicador)/10; FINALIZA EXTRACCION

}

//valor1=valor1-(+184+72);

}

break;

case CEROYCERO:

//

valor1=0;

valor2=0;

break;

default:

printf("ERROR EN OPERANDOS\n");

break;

}

//printf("Valor 2 %d\n",valor2);

//printf("Valor 1 %d\n",valor1);

}

void funcionmov(int a){

// printf("valor1 %d\n",valor1);

switch(a){

case NUMEROYLETRA:

if (valor1==153)

registros16.AX=valor2;

else if (valor1==154)

registros16.BX=valor2;

else if (valor1==155)

registros16.CX=valor2;

else if (valor1==156)

registros16.DX=valor2; FUNCION MOV Y CADA

else if (valor1==151) UNO DE LOS CASOS

segmentos.DS=valor2;

else if (valor1==166)

segmentos.SS=valor2;

else if (valor1==141){

if (op1[contador][0]==65)

registros8.AL=valor2;

else if (op1[contador][0]==68)

indices.DI=valor2;

}

else if (valor1==156)

indices.SI=valor2;

else if (valor1==146)

punteros.BP=valor2;

else if (valor1==137)

registros8.AH=valor2;

else if (valor1==138)

registros8.BH=valor2;

else if (valor1==142)

registros8.BL=valor2;

else if (valor1==139)

registros8.CH=valor2;

else if (valor1==143)

registros8.CL=valor2;

else if (valor1==140)

registros8.DH=valor2;

else if (valor1==144)

registros8.DL=valor2;

break;

case CEROYLETRA:

if (valor1==153)

registros16.AX=valor2;

else if (valor1==154)

registros16.BX=valor2; CONTINUA FUNCION MOV

else if (valor1==155)

registros16.CX=valor2;

else if (valor1==156){

if(op1[contador][0]==68)

registros16.DX=valor2;

else if (op1[contador][0]==83)

indices.SI=valor2;

}

else if (valor1==151)

segmentos.DS=valor2;

else if (valor1==166)

segmentos.SS=valor2;

else if (valor1==141){

if (op1[contador][0]==65)

registros8.AL=valor2;

else if (op1[contador][0]==68)

indices.DI=valor2;

}

else if (valor1==146)

punteros.BP=valor2;

else if (valor1==137)

registros8.AH=valor2;

else if (valor1==138)

registros8.BH=valor2;

else if (valor1==142)

registros8.BL=valor2;

else if (valor1==139)

registros8.CH=valor2;

else if (valor1==143)

registros8.CL=valor2;

else if (valor1==140)

registros8.DH=valor2;

else if (valor1==144)

registros8.DL=valor2;

break;

// case NUMEROYCORCHETE:

// break;

case LETRAYLETRA:

if (valor1==153){

if (valor2==154)

registros16.AX=registros16.BX;

else if (valor2==155)

registros16.AX=registros16.CX;

else if (valor2==156)

registros16.AX=registros16.DX;

else if (valor2==151)

registros16.AX=segmentos.DS;

else if (valor2==166)

registros16.AX=segmentos.SS;

else if (valor2==141)

registros16.AX=indices.DI;

else if (valor2==156)

registros16.AX=indices.SI;

else if (valor2==146)

registros16.AX=punteros.BP; CONTINUA FUNCION MO

}

else if (valor1==154){

if (valor2==153)

registros16.BX=registros16.AX;

else if (valor2==155)

registros16.BX=registros16.CX;

else if (valor2==156)

registros16.BX=registros16.DX;

else if (valor2==151)

registros16.BX=segmentos.DS;

else if (valor2==166)

registros16.BX=segmentos.SS;

else if (valor2==141)

registros16.BX=indices.DI;

else if (valor2==156)

registros16.BX=indices.SI;

else if (valor2==146)

registros16.AX=punteros.BP;

}

else if (valor1==155){

if (valor2==154)

registros16.CX=registros16.BX;

else if (valor2==153)

registros16.CX=registros16.AX;

else if (valor2==156)

registros16.CX=registros16.DX;

else if (valor2==151)

registros16.CX=segmentos.DS;

else if (valor2==166)

registros16.CX=segmentos.SS;

else if (valor2==141)

registros16.CX=indices.DI;

else if (valor2==156)

registros16.CX=indices.SI;

else if (valor2==146)

registros16.CX=punteros.BP;

}

else if (valor1==156){

if (op1[contador][0]==68){

if (valor2==154)

registros16.DX=registros16.BX;

else if (valor2==155)

registros16.DX=registros16.CX;

else if (valor2==153)

registros16.DX=registros16.AX;

else if (valor2==151)

registros16.DX=segmentos.DS;

else if (valor2==166)

registros16.DX=segmentos.SS;

else if (valor2==141)

registros16.DX=indices.DI;

else if (valor2==156) CONTINUA FUNCION MOV

registros16.DX=indices.SI;

else if (valor2==146)

registros16.DX=punteros.BP;

}

else if (op1[contador][0]==83){

if (valor2==154)

indices.SI=registros16.BX;

else if (valor2==155)

indices.SI=registros16.CX;

else if (valor2==156)

indices.SI=registros16.DX;

else if (valor2==153)

indices.SI=registros16.AX;

}

}

if (valor1==137){ //AH hasta....

if (valor2==141)

registros8.AH=registros8.AL;

else if (valor2==138)

registros8.AH=registros8.BH;

else if (valor2==142)

registros8.AH=registros8.BL;

else if (valor2==139)

registros8.AH=registros8.CH;

else if (valor2==143)

registros8.AH=registros8.CL;

else if (valor2==140)

registros8.AH=registros8.DH;

else if (valor2==144)

registros8.AH=registros8.DL;

}

else if (valor1==141){

if(op1[contador][0]==65){

if (valor2==137)

registros8.AL=registros8.AH;

else if (valor2==138)

registros8.AL=registros8.BH;

else if (valor2==142)

registros8.AL=registros8.BL;

else if (valor2==139)

registros8.AL=registros8.CH;

else if (valor2==143)

registros8.AL=registros8.CL;

else if (valor2==140)

registros8.AL=registros8.DH;

else if (valor2==144)

registros8.AL=registros8.DL;

}

else if (op1[contador][0]==68){

if (valor2==154)

indices.DI=registros16.BX;

else if (valor2==155) CONTINUA FUNCION MOV

indices.DI=registros16.CX;

else if (valor2==156)

indices.DI=registros16.DX;

else if (valor2==153)

indices.DI=registros16.AX;

}

}

else if (valor1==138){

if (valor2==141)

registros8.BH=registros8.AL;

else if (valor2==137)

registros8.BH=registros8.AH;

else if (valor2==142)

registros8.BH=registros8.BL;

else if (valor2==139)

registros8.BH=registros8.CH;

else if (valor2==143)

registros8.BH=registros8.CL;

else if (valor2==140)

registros8.BH=registros8.DH;

else if (valor2==144)

registros8.BH=registros8.DL;

}

else if (valor1==142){

if (valor2==141)

registros8.BL=registros8.AL;

else if (valor2==138)

registros8.BL=registros8.BH;

else if (valor2==137)

registros8.BL=registros8.AH;

else if (valor2==139)

registros8.BL=registros8.CH;

else if (valor2==143)

registros8.BL=registros8.CL;

else if (valor2==140)

registros8.BL=registros8.DH;

else if (valor2==144)

registros8.BL=registros8.DL;

}

else if (valor1==139){

if (valor2==141)

registros8.CH=registros8.AL;

else if (valor2==138)

registros8.CH=registros8.BH;

else if (valor2==142)

registros8.CH=registros8.BL;

else if (valor2==127)

registros8.CH=registros8.AH;

else if (valor2==143)

registros8.CH=registros8.CL;

else if (valor2==140)

registros8.CH=registros8.DH; CONTINUA FUNCION MOV

else if (valor2==144)

registros8.CH=registros8.DL;

}

else if (valor1==143){

if (valor2==141)

registros8.CL=registros8.AL;

else if (valor2==138)

registros8.CL=registros8.BH;

else if (valor2==142)

registros8.CL=registros8.BL;

else if (valor2==139)

registros8.CL=registros8.CH;

else if (valor2==137)

registros8.CL=registros8.AH;

else if (valor2==140)

registros8.CL=registros8.DH;

else if (valor2==144)

registros8.CL=registros8.DL;

}

else if (valor1==140){

if (valor2==141)

registros8.DH=registros8.AL;

else if (valor2==138)

registros8.DH=registros8.BH;

else if (valor2==142)

registros8.DH=registros8.BL;

else if (valor2==139)

registros8.DH=registros8.CH;

else if (valor2==143)

registros8.DH=registros8.CL;

else if (valor2==137)

registros8.DH=registros8.AH;

else if (valor2==144)

registros8.DH=registros8.DL;

}

else if (valor1==144){ // ... DL

if (valor2==141)

registros8.DL=registros8.AL;

else if (valor2==138)

registros8.DL=registros8.BH;

else if (valor2==142)

registros8.DL=registros8.BL;

else if (valor2==139)

registros8.DL=registros8.CH;

else if (valor2==143)

registros8.DL=registros8.CL;

else if (valor2==140)

registros8.DL=registros8.DH;

else if (valor2==137)

registros8.DL=registros8.AH;

}

else if (valor1==151){

if (valor2==154)

segmentos.DS=registros16.BX; CONTINUA FUNCION MOV

else if (valor2==155)

segmentos.DS=registros16.CX;

else if (valor2==156)

segmentos.DS=registros16.DX;

else if (valor2==153)

segmentos.DS=registros16.AX;

}

else if (valor1==166){

if (valor2==154)

segmentos.SS=registros16.BX;

else if (valor2==155)

segmentos.SS=registros16.CX;

else if (valor2==156)

segmentos.SS=registros16.DX;

else if (valor2==153)

segmentos.SS=registros16.AX;

}

else if (valor1==146){

if (valor2==154)

punteros.BP=registros16.BX;

else if (valor2==155)

punteros.BP=registros16.CX;

else if (valor2==156)

punteros.BP=registros16.DX;

else if (valor2==153)

punteros.BP=registros16.AX;

}

break;

case LETRAYCORCHETE: //checar casos en que corchetes contengan registro y no numeros!!!!!!!!

//printf("valor1 = %d\n",valor1);

if (valor2==153){

if (valor1==375)

memoriaext.memoria[indices.SI+(16\*segmentos.DS)]=registros16.AX;

else if (valor1==225)

memoriaext.memoria[indices.DI+(16\*segmentos.DS)]=registros16.AX;

else if(valor1==212)

memoriaext.memoria[punteros.BP+(16\*segmentos.SS)]=registros16.AX;

else

memoriaext.memoria[valor1+(16\*segmentos.DS)]=registros16.AX;

}

else if (valor2==154){

if (valor1==375)

memoriaext.memoria[indices.SI+(16\*segmentos.DS)]=registros16.BX;

else if (valor1==225)

memoriaext.memoria[indices.DI+(16\*segmentos.DS)]=registros16.BX;

else if(valor1==212)

memoriaext.memoria[punteros.BP+(16\*segmentos.SS)]=registros16.BX;

else

memoriaext.memoria[valor1+(16\*segmentos.DS)]=registros16.BX;

}

else if (valor2==155){

if (valor1==375) CONT. MOV

memoriaext.memoria[indices.SI+(16\*segmentos.DS)]=registros16.CX;

else if (valor1==225)

memoriaext.memoria[indices.DI+(16\*segmentos.DS)]=registros16.CX;

else if(valor1==212)

memoriaext.memoria[punteros.BP+(16\*segmentos.SS)]=registros16.CX;

else

memoriaext.memoria[valor1+(16\*segmentos.DS)]=registros16.CX;

}

else if (valor2==156){

if (valor1==375)

memoriaext.memoria[indices.SI+(16\*segmentos.DS)]=registros16.DX;

else if (valor1==225)

memoriaext.memoria[indices.DI+(16\*segmentos.DS)]=registros16.DX;

else if(valor1==212)

memoriaext.memoria[punteros.BP+(16\*segmentos.SS)]=registros16.DX;

else

memoriaext.memoria[valor1+(16\*segmentos.DS)]=registros16.DX;

}

break;

case CORCHETEYLETRA: //checar casos en que corchetes contengan registro y no numeros!!!!!!!!

if (valor1==153){

if (valor2==375)

registros16.AX=memoriaext.memoria[indices.SI+(16\*segmentos.DS)];

else if (valor2==225)

registros16.AX=memoriaext.memoria[indices.DI+(16\*segmentos.DS)];

else if (valor2==212)

registros16.AX=memoriaext.memoria[punteros.BP+(16\*segmentos.SS)];

else

registros16.AX=memoriaext.memoria[valor2+(16\*segmentos.DS)];

}

else if (valor1==154){

if (valor2==375)

registros16.BX=memoriaext.memoria[indices.SI+(16\*segmentos.DS)];

else if (valor2==225)

registros16.BX=memoriaext.memoria[indices.DI+(16\*segmentos.DS)];

else if (valor2==212)

registros16.BX=memoriaext.memoria[punteros.BP+(16\*segmentos.SS)];

else

registros16.BX=memoriaext.memoria[valor2+(16\*segmentos.DS)];

}

else if (valor1==155){

if (valor2==375)

registros16.CX=memoriaext.memoria[indices.SI+(16\*segmentos.DS)];

else if (valor2==225)

registros16.CX=memoriaext.memoria[indices.DI+(16\*segmentos.DS)];

else if (valor2==212)

registros16.CX=memoriaext.memoria[punteros.BP+(16\*segmentos.SS)];

else

registros16.CX=memoriaext.memoria[valor2+(16\*segmentos.DS)]; CONT. MOV

}

else if (valor1==156){

if (valor2==375)

registros16.DX=memoriaext.memoria[indices.SI+(16\*segmentos.DS)];

else if (valor2==225)

registros16.DX=memoriaext.memoria[indices.DI+(16\*segmentos.DS)];

else if (valor2==212)

registros16.DX=memoriaext.memoria[punteros.BP+(16\*segmentos.SS)];

else

registros16.DX=memoriaext.memoria[valor2+(16\*segmentos.DS)];

}

else if (valor1==137){

if (valor2==375)

registros8.AH=memoriaext.memoria[indices.SI+(16\*segmentos.DS)];

else if (valor2==225)

registros8.AH=memoriaext.memoria[indices.DI+(16\*segmentos.DS)];

else if (valor2==212)

registros8.AH=memoriaext.memoria[punteros.BP+(16\*segmentos.SS)];

else

registros8.AH=memoriaext.memoria[valor2+(16\*segmentos.DS)];

}

else if (valor1==141){

if (valor2==375)

registros8.AL=memoriaext.memoria[indices.SI+(16\*segmentos.DS)];

else if (valor2==225)

registros8.AL=memoriaext.memoria[indices.DI+(16\*segmentos.DS)];

else if (valor2==212)

registros8.AL=memoriaext.memoria[punteros.BP+(16\*segmentos.SS)];

else

registros8.AL=memoriaext.memoria[valor2+(16\*segmentos.DS)];

}

else if (valor1==138){

if (valor2==375)

registros8.BH=memoriaext.memoria[indices.SI+(16\*segmentos.DS)];

else if (valor2==225)

registros8.BH=memoriaext.memoria[indices.DI+(16\*segmentos.DS)];

else if (valor2==212)

registros8.BH=memoriaext.memoria[punteros.BP+(16\*segmentos.SS)];

else

registros8.BH=memoriaext.memoria[valor2+(16\*segmentos.DS)];

}

else if (valor1==142){ FINALIZA MOV

if (valor2==375)

registros8.BL=memoriaext.memoria[indices.SI+(16\*segmentos.DS)];

else if (valor2==225)

registros8.BL=memoriaext.memoria[indices.DI+(16\*segmentos.DS)];

else if (valor2==212)

registros8.BL=memoriaext.memoria[punteros.BP+(16\*segmentos.SS)];

else

registros8.BL=memoriaext.memoria[valor2+(16\*segmentos.DS)];

}

break;

default:

printf("ERROR EN MOV\n");

break;

}

}

void funcionadd(int a){

int valormemoria;

valormemoria=0;

switch(a){

case NUMEROYLETRA:

break;

case LETRAYLETRA:

if (valor1==153){

if (valor2==154) INICIA FUNCION ADD

registros16.AX+=registros16.BX;

else if (valor2==155)

registros16.AX+=registros16.CX;

else if (valor2==156)

registros16.AX+=registros16.DX;

}

if (valor1==154){

if (valor2==153)

registros16.BX+=registros16.AX;

else if (valor2==155)

registros16.BX+=registros16.CX;

else if (valor2==156)

registros16.BX+=registros16.DX;

}

if (valor1==155){

if (valor2==154)

registros16.CX+=registros16.BX;

else if (valor2==155)

registros16.CX+=registros16.AX;

else if (valor2==156)

registros16.CX+=registros16.DX;

}

if (valor1==156){

if (valor2==154)

registros16.DX+=registros16.BX;

else if (valor2==155)

registros16.DX+=registros16.CX;

else if (valor2==156)

registros16.DX+=registros16.AX;

}

break;

case LETRAYCORCHETE:

break;

case CORCHETEYLETRA:

if (valor1==153){ CONT. ADD

if (valor2==225){

valormemoria=memoriaext.memoria[indices.DI+(16\*segmentos.DS)];

registros16.AX=(registros16.AX)+valormemoria;

}

else if (valor2==375){

valormemoria=memoriaext.memoria[indices.SI+(16\*segmentos.DS)];

registros16.AX=(registros16.AX)+valormemoria;

}

else if (valor2==212){

valormemoria=memoriaext.memoria[punteros.BP+(16\*segmentos.SS)];

registros16.AX=(registros16.AX)+valormemoria;

}

else

registros16.AX=registros16.AX+memoriaext.memoria[valor2+(16\*segmentos.DS)];

}

else if (valor1==154){

if (valor2==225){

valormemoria=memoriaext.memoria[indices.DI+(16\*segmentos.DS)];

registros16.BX=(registros16.BX)+valormemoria;

}

else if (valor2==375){

valormemoria=memoriaext.memoria[indices.SI+(16\*segmentos.DS)];

registros16.BX=(registros16.BX)+valormemoria;

}

else if (valor2==212){

valormemoria=memoriaext.memoria[punteros.BP+(16\*segmentos.SS)];

registros16.BX=(registros16.BX)+valormemoria;

}

else

registros16.BX=registros16.BX+memoriaext.memoria[valor2+(16\*segmentos.DS)];

}

else if (valor1==155){

if (valor2==225){

valormemoria=memoriaext.memoria[indices.DI+(16\*segmentos.DS)];

registros16.CX=(registros16.CX)+valormemoria;

}

else if (valor2==375){

valormemoria=memoriaext.memoria[indices.SI+(16\*segmentos.DS)];

registros16.CX=(registros16.CX)+valormemoria;

}

else if (valor2==212){

valormemoria=memoriaext.memoria[punteros.BP+(16\*segmentos.SS)];

registros16.CX=(registros16.CX)+valormemoria;

}

else

registros16.CX=registros16.CX+memoriaext.memoria[valor2+(16\*segmentos.DS)];

} CONT

else if (valor1==156){ ADD

if (valor2==225){

valormemoria=memoriaext.memoria[indices.DI+(16\*segmentos.DS)];

registros16.DX=(registros16.DX)+valormemoria;

}

else if (valor2==375){

valormemoria=memoriaext.memoria[indices.SI+(16\*segmentos.DS)];

registros16.DX=(registros16.DX)+valormemoria;

}

else if (valor2==212){

valormemoria=memoriaext.memoria[punteros.BP+(16\*segmentos.SS)];

registros16.DX=(registros16.DX)+valormemoria;

}

else

registros16.DX=registros16.DX+memoriaext.memoria[valor2+(16\*segmentos.DS)];

}

break;

default:

printf("ERRROR EN ADD\n");

break;

}

hextobin();

registros16.AX=registros16.AX&65535;

registros16.BX=registros16.BX&65535; TRANSFORMA A BINARIO PARA BANDERAS

registros16.CX=registros16.CX&65535; LIMITA RESULTADOS A FFFFh

registros16.DX=registros16.DX&65535;

}

void funcionadc(int a){

switch(a){

case LETRAYLETRA:

if (valor1==153){

if (valor2==154)

registros16.AX+=banderas.CF+registros16.BX;

else if(valor2==155)

registros16.AX+=banderas.CF+registros16.CX;

else if (valor2==156)

registros16.AX+=banderas.CF+registros16.DX;

}

if (valor1==154){

if (valor2==153)

registros16.BX+=banderas.CF+registros16.AX;

else if(valor2==155)

registros16.BX+=banderas.CF+registros16.CX;

else if (valor2==156)

registros16.BX+=banderas.CF+registros16.DX;

}

if (valor1==155){ FUNCION ADC

if (valor2==154)

registros16.CX+=banderas.CF+registros16.BX;

else if(valor2==153)

registros16.CX+=banderas.CF+registros16.AX;

else if (valor2==156)

registros16.CX+=banderas.CF+registros16.DX;

}

if (valor1==156){

if (valor2==154)

registros16.DX+=banderas.CF+registros16.BX;

else if(valor2==155)

registros16.DX+=banderas.CF+registros16.CX;

else if (valor2==156)

registros16.DX+=banderas.CF+registros16.AX;

}

break;

default:

printf("ERROR EN ADC\n");

break;

}

hextobin();

registros16.AX=registros16.AX&65535;

registros16.BX=registros16.BX&65535; CONVIERTE A BINARIO PARA BANDERAS

registros16.CX=registros16.CX&65535; LIMITA RESULTADOS A FFFFH

registros16.DX=registros16.DX&65535;

}

void funcionhlt(){

printf("El programa se ha detenido\n");

system("pause"); FUNCION HALT

system("cls");

exit(0);

}

void funcioninc(){

int carry;

carry=banderas.CF;

if (valor1==153)

registros16.AX++;

else if (valor1==154)

registros16.BX++;

else if (valor1==155)

registros16.CX++;

else if (valor1==156){

if (op1[contador][0]==68)

registros16.DX++;

else if (op1[contador][0]==83)

indices.SI++;

}

else if (valor1==137)

registros8.AH++;

else if (valor1==141){

if (op1[contador][0]==65)

registros8.AL++;

else if (op1[contador][0]==68)

indices.DI++;

}

else if (valor1==138) FUNCION INCREMENTA

registros8.BH++;

else if (valor1==142)

registros8.BL++;

else if (valor1==139)

registros8.CH++;

else if (valor1==143)

registros8.CL++;

else if (valor1==140)

registros8.DH++;

else if (valor1==144)

registros8.DL++;

else if (valor1==151)

segmentos.DS++;

else if (valor1==166)

segmentos.SS++;

else if (valor1==146)

punteros.BP++;

else

printf("ERROR EN INC\n");

//reg16to8();

hextobin(); SE CERCIORA DE QUE AH Y AL CAMBIEN A LA PAR QUE AX

banderas.CF=carry; SE CONVIERTE A BINARIO PARA BANDERAS, BANDERA CARRY

QUEDA SIN MODIFICAR

registros16.AX=registros16.AX&65535;

registros16.BX=registros16.BX&65535;

registros16.CX=registros16.CX&65535; SE LIMITA RESULTADO A FFFFH

registros16.DX=registros16.DX&65535;

}

void funciondec(){

int carry;

carry=banderas.CF;

if (valor1==153)

registros16.AX--;

else if (valor1==154)

registros16.BX--;

else if (valor1==155)

registros16.CX--;

else if (valor1==156){

if (op1[contador][0]==68)

registros16.DX--;

else if (op1[contador][0]==83)

indices.SI--;

}

else if (valor1==137)

registros8.AH--;

else if (valor1==141)

registros8.AL--;

else if (valor1==138) FUNCION DECREMENTA

registros8.BH--;

else if (valor1==142)

registros8.BL--;

else if (valor1==139)

registros8.CH--;

else if (valor1==143)

registros8.CL--;

else if (valor1==140)

registros8.DH--;

else if (valor1==144)

registros8.DL--;

else if (valor1==151)

segmentos.DS--;

else if (valor1==166)

segmentos.SS--;

else if (valor1==141)

indices.DI--;

else if (valor1==146)

punteros.BP--;

else

printf("ERROR EN DEC\n");

//reg16to8();

hextobin(); SE CERCIORA DE QUE AH Y AL CAMBIEN A LA PAR DE AX

banderas.CF=carry; CONVIERTE A BINARIO PARA BANDERAS, BANDERA CARRY

SE MANTIENE SIN MOFICACION

registros16.AX=registros16.AX&65535;

registros16.BX=registros16.BX&65535;

registros16.CX=registros16.CX&65535; SE LIMITAN LOS RESULTADOS A FFFFH

registros16.DX=registros16.DX&65535;

}

void funcioncmp(){

compara=0;

funcion=0;

//printf("Clave %d\n",clave);

//printf("Valor1 %d\n",valor1);

//printf("CL %d\n",registros8.CL);

switch (clave){

case LETRAYLETRA:

if (valor1==153){

if (valor2==154)

compara=registros16.AX-registros16.BX;

else if (valor2==155)

compara=registros16.AX-registros16.CX;

else if (valor2==156)

compara=registros16.AX-registros16.DX;

}

if (valor1==154){

if (valor2==153)

compara=registros16.BX-registros16.AX;

else if (valor2==155)

compara=registros16.BX-registros16.CX;

else if (valor2==156)

compara=registros16.BX-registros16.DX;

}

if (valor1==155){

if (valor2==154) FUNCION COMPARA

compara=registros16.CX-registros16.BX;

else if (valor2==155)

compara=registros16.CX-registros16.AX;

else if (valor2==156)

compara=registros16.CX-registros16.DX;

}

if (valor1==156){

if (valor2==154)

compara=registros16.DX-registros16.BX;

else if (valor2==155)

compara=registros16.DX-registros16.CX;

else if (valor2==156)

compara=registros16.DX-registros16.AX;

}

if (valor1==137){

if (valor2==142)

compara=registros8.AL-registros8.BL;

else if (valor2==143)

compara=registros8.AL-registros8.CL;

else if (valor2==144)

compara=registros8.AL-registros8.DL;

}

break;

case CEROYLETRA:

if (valor1==153)

compara=registros16.AX;

else if (valor1==154)

compara=registros16.BX;

else if (valor1==155)

compara=registros16.CX;

else if (valor1==156)

compara=registros16.DX;

else if (valor1==137)

compara=registros8.AH;

else if (valor1==141)

compara=registros8.AL;

else if (valor1==138)

compara=registros8.BH;

else if (valor1==142)

compara=registros8.BL;

else if (valor1==139) CONTINUA FUNCION COMPARA

compara=registros8.CH;

else if (valor1==143)

compara=registros8.CL;

else if (valor1==140)

compara=registros8.DH;

else if (valor1==144)

compara=registros8.DL;

break;

case NUMEROYLETRA:

if (valor1==146)

compara=punteros.BP-valor2;

else if (valor1==141){

compara=indices.DI-valor2;

//printf("Valor2 %d\n",valor2);

}

break;

}

//printf("Compara %d\n",compara);

funcion=1;

hextobin(); CONVIERTE A BINARIO PARA BANDERAS

funcion=0;

}

void funcionjmp(){

int f,g,sigue;

f=0;

valor1=0;

g=0;

//printf("Opj1 Posicion 3: %d\n",opj1[contador][3]);

while (sigue){

valor1=valor1+opj1[contador][g]; FUNCION JUMP

g++;

if (valor1==destino[f] && valor1!=0){

if(opj1[contador][g]!=0)

valor1=valor1+opj1[contador][g];

sigue=0;

}

else if (valor1>destino[f]){

g=0;

valor1=0; CONTINUA FUNCION JMP

f=f+1;

}

}

for(f=0;f<200;f++){

if (valor1==destino[f]){

contador=f-1;

c=f;

}

}

}

void funcionjc(){

int f,g,sigue;

f=0;

valor1=0;

g=0;

while (sigue){

valor1=valor1+opj1[contador][g];

g++;

if (valor1==destino[f] && valor1!=0){

if(opj1[contador][g]!=0)

valor1=valor1+opj1[contador][g];

sigue=0;

} FUNCION JUMP IF CARRY

else if (valor1>destino[f]){

g=0;

valor1=0;

f=f+1;

}

}

for(f=0;f<200;f++){

if (valor1==destino[f] && banderas.CF==1){

contador=f-1;

c=f;

}

}

}

void funcionjz(){

int f,g,sigue;

f=0;

valor1=0;

g=0;

/\*printf("entro\n"); FUNCION JUMP IF ZERO

printf("OPJ1 %d\n",opj1[contador][0]);

printf("OPJ1 %d\n",opj1[contador][1]);

printf("OPJ1 %d\n",opj1[contador][2]);

printf("OPJ1 %d\n",opj1[contador][3]);

printf("OPJ1 %d\n",opj1[contador][4]);

printf("OPJ1 %d\n",opj1[contador][5]);\*/

while (sigue){

valor1=valor1+opj1[contador][g];

g++;

/\* printf("G %d\n",g);

printf("OPJ1 %d\n",opj1[contador][0]);

printf("destino[f] %d\n",destino[f]);

printf("F %d\n",f);

printf("Valor 1 Fuera %d\n",valor1);

system("pause");

\*/

if (valor1==destino[f] && valor1!=0){

if(opj1[contador][g]!=0)

valor1=valor1+opj1[contador][g];

sigue=0;

//printf("Valor1 Dentro%d\n",valor1);

}

else if (valor1>destino[f]){ CONTINUA FUNCION JMP IF ZERO

g=0;

valor1=0;

f=f+1;

}

}

// printf("F %d\n",f);

//printf("Entre!!!!\n");

//printf("Destion %d\n",destino);

//printf("Valor 1 %d\n",valor1);

for(f=0;f<200;f++){

if (valor1==destino[f] && banderas.ZF==1){

contador=f-1;

c=f;

}

}

}

void funcioncall(){

int f,g,sigue;

f=0;

valor1=0;

g=0;

punteros.SP=contador;

while (sigue){

valor1=valor1+opj1[contador][g];

g++; FUNCION CALL

if (valor1==destino[f] && valor1!=0){

if(opj1[contador][g]!=0)

valor1=valor1+opj1[contador][g];

sigue=0;

}

else if (valor1>destino[f]){

g=0;

valor1=0;

f=f+1;

}

}

for(f=0;f<200;f++){

if (valor1==destino[f]){

contador=f-1; CONTINUA FUNCION CALL

c=f;

}

}

}

void funcionret(){

contador=punteros.SP; FUNCION RETURN

c=contador+1;

}

int hextodecop2(int a){

int b, cuenta,multiplicacion, multiplicador,resultado,k,r,f,longitud;

cuenta=0;

b=0;

multiplicacion=0;

multiplicador=0;

resultado=0;

r=0;

k=f=0;

longitud=0;

while(k!=72){

cuenta=cuenta+1;

k=op2[a][cuenta];

}

if (cuenta>9)

cuenta=1;

multiplicador=pow(16,cuenta-1);

//printf("Multiplicador: %d\n",multiplicador); FUNCION QUE TRANSFORMA DE

//printf("Cuenta: %d\n",cuenta); HEX A DECIMAL

r=cuenta;

for (r;r>=1;r--){

if(op2[a][b]>=48 && op2[a][b]<=57){

multiplicacion=multiplicador\*(op2[a][b]-48);

}

else if (op2[a][b]>=97 && op2[a][b]<=102){

if (op2[a][b]=='a')

multiplicacion=multiplicador\*10;

else if (op2[a][b]=='b')

multiplicacion=multiplicador\*11;

else if (op2[a][b]=='c')

multiplicacion=multiplicador\*12;

else if (op2[a][b]=='d')

multiplicacion=multiplicador\*13;

else if (op2[a][b]=='e')

multiplicacion=multiplicador\*14;

else if (op2[a][b]=='f')

multiplicacion=multiplicador\*15;

}

b=b+1;

multiplicador=multiplicador/16;

resultado=resultado+multiplicacion;

}

//printf("Transformacion de Hex: %d\n",resultado);

char buf[5];

for (f=0;f<5;f++){

buf[f]=0;

}

itoa(resultado,buf,10);

//printf("Transformacion a Cadena: %s\n",buf);

longitud=strlen(buf);

if (longitud==1){

buf[1]=buf[0];

buf[0]='0';

longitud=longitud+1;

}

if (longitud!=1){

for(f=0;f<5;f++){

op2[a][f]=buf[f];

//printf("OP2:%d ",op2[a][f]);

}

} CONTINIUA TRANSFORMACION

//printf("Longitud: %d\n",longitud); HEX TO DEC

op2[a][longitud]='H';

//printf("Caracter H: %d\n",op2[a][longitud+1]);

//printf("A: %d\n",a);

return a;

}

int hextodecop2c(int a){

int b, cuenta,multiplicacion, multiplicador,resultado,k,r,f, longitud;

cuenta=0;

b=0;

multiplicacion=0;

multiplicador=0;

resultado=0;

r=0;

k=f=0;

longitud=0;

if(op2[a][b]==48 && op2[a][b+1]==48)

goto brinca;

while(k!=72){

cuenta=cuenta+1;

k=op2[a][cuenta];

}

brinca:

multiplicador=pow(16,cuenta-2);

//printf("Multiplicador: %d\n",multiplicador);

//printf("Cuenta: %d\n",cuenta);

r=cuenta;

for (r;r>=2;r--){

if(op2[a][b+1]>=48 && op2[a][b+1]<=57){

multiplicacion=multiplicador\*(op2[a][b+1]-48);

}

else if (op2[a][b+1]>=97 && op2[a][b+1]<=102){

if (op2[a][b+1]=='a')

multiplicacion=multiplicador\*10;

else if (op2[a][b+1]=='b')

multiplicacion=multiplicador\*11;

else if (op2[a][b+1]=='c')

multiplicacion=multiplicador\*12;

else if (op2[a][b+1]=='d')

multiplicacion=multiplicador\*13;

else if (op2[a][b+1]=='e')

multiplicacion=multiplicador\*14;

else if (op2[a][b+1]=='f')

multiplicacion=multiplicador\*15;

}

b=b+1;

multiplicador=multiplicador/16;

resultado=resultado+multiplicacion;

} CONTINUA TRANSFORMACION

//printf("Transformacion de Hex: %d\n",resultado); HEXT TO DEC

char buf[5];

for (f=0;f<5;f++){

buf[f]=0;

}

itoa(resultado,buf,10);

//printf("Transformacion a Cadena: %s\n",buf);

longitud=strlen(buf);

if (longitud!=1){

for(f=0;f<5;f++){

op2[a][f+1]=buf[f];

//printf("OP2:%d ",op2[a][f]);

}

}

//printf("Longitud: %d\n",longitud);

op2[a][longitud+1]='H';

op2[a][longitud+2]=']';

//printf("Caracter H: %d\n",op2[a][longitud+1]);

//printf("Caracter ]: %d\n",op2[a][longitud+2]);

//printf("A: %d\n",a);

return a;

}

int hextodecop1(int a){

int b, cuenta,multiplicacion, multiplicador,resultado,k,r,f,longitud;

cuenta=0;

b=0;

multiplicacion=0;

multiplicador=0;

resultado=0;

r=0;

k=f=0;

longitud=0;

while(k!=72){

cuenta=cuenta+1;

k=op1[a][cuenta];

}

if (cuenta>9)

cuenta=1;

multiplicador=pow(16,cuenta-1);

//printf("Multiplicador: %d\n",multiplicador);

//printf("Cuenta: %d\n",cuenta);

r=cuenta;

for (r;r>=1;r--){

if(op1[a][b]>=48 && op1[a][b]<=57){

multiplicacion=multiplicador\*(op1[a][b]-48);

}

else if (op1[a][b]>=97 && op1[a][b]<=102){

if (op1[a][b]=='a') CONTINUA TRANSFORMACION

multiplicacion=multiplicador\*10; HEX TO DEC

else if (op1[a][b]=='b')

multiplicacion=multiplicador\*11;

else if (op1[a][b]=='c')

multiplicacion=multiplicador\*12;

else if (op1[a][b]=='d')

multiplicacion=multiplicador\*13;

else if (op1[a][b]=='e')

multiplicacion=multiplicador\*14;

else if (op1[a][b]=='f')

multiplicacion=multiplicador\*15;

}

b=b+1;

multiplicador=multiplicador/16;

resultado=resultado+multiplicacion;

}

//printf("Transformacion de Hex: %d\n",resultado);

char buf[7];

for (f=0;f<7;f++){

buf[f]=0;

}

itoa(resultado,buf,10);

//printf("Transformacion a Cadena: %s\n",buf);

longitud=strlen(buf);

if (longitud==1){

buf[1]=buf[0];

buf[0]='0';

longitud=longitud+1;

}

if (longitud!=1){

for(f=0;f<7;f++){

op1[a][f]=buf[f];

//printf("OP1:%d ",op1[a][f]);

}

}

//printf("Longitud: %d\n",longitud);

op1[a][longitud]='H';

//printf("Caracter H: %d\n",op1[a][longitud+1]);

//printf("A: %d\n",a);

return a;

}

int hextodecop1c(int a){

int b, cuenta,multiplicacion, multiplicador,resultado,k,r,f,longitud;

cuenta=0;

b=0;

multiplicacion=0;

multiplicador=0;

resultado=0;

r=0;

k=f=0;

longitud=0;

CONTINUA TRANSFORMACION

while(k!=72){ HEX TO DEC

cuenta=cuenta+1;

k=op1[a][cuenta];

}

multiplicador=pow(16,cuenta-2);

//printf("Multiplicador: %d\n",multiplicador);

//printf("Cuenta: %d\n",cuenta);

r=cuenta;

for (r;r>=2;r--){

if(op1[a][b+1]>=48 && op1[a][b+1]<=57){

multiplicacion=multiplicador\*(op1[a][b+1]-48);

}

else if (op1[a][b+1]>=97 && op1[a][b+1]<=102){

if (op1[a][b+1]=='a')

multiplicacion=multiplicador\*10;

else if (op1[a][b+1]=='b')

multiplicacion=multiplicador\*11;

else if (op1[a][b+1]=='c')

multiplicacion=multiplicador\*12;

else if (op1[a][b+1]=='d')

multiplicacion=multiplicador\*13;

else if (op1[a][b+1]=='e')

multiplicacion=multiplicador\*14;

else if (op1[a][b+1]=='f')

multiplicacion=multiplicador\*15;

}

b=b+1;

multiplicador=multiplicador/16;

resultado=resultado+multiplicacion;

}

//printf("Transformacion de Hex: %d\n",resultado);

char buf[7];

for (f=0;f<7;f++){

buf[f]=0;

}

itoa(resultado,buf,10);

//printf("Transformacion a Cadena: %s\n",buf);

longitud=strlen(buf);

if (longitud!=1){ FINALIZA TRANSFORMACION HEX TO DEC

for(f=0;f<7;f++){

op1[a][f+1]=buf[f];

//printf("OP1:%d ",op1[a][f]);

}

}

//printf("Longitud: %d\n",longitud);

op1[a][longitud+1]='H';

op1[a][longitud+2]=']';

//printf("Caracter H: %d\n",op1[a][longitud+1]);

//printf("Caracter ]: %d\n",op1[a][longitud+2]);

//printf("A: %d\n",a);

return a;

}

void reg16to8(){

if (valor1==137 || valor1==141 || valor1==138 || valor1==142 ||valor1==139 ||valor1==143 ||valor1==140 ||valor1==144){

//printf("Registro AH %d\n",registros8.AH);

registros16.AX=(registros8.AH\*pow(16,2))+registros8.AL;

registros16.BX=(registros8.BH\*pow(16,2))+registros8.BL;

registros16.CX=(registros8.CH\*pow(16,2))+registros8.CL;

registros16.DX=(registros8.DH\*pow(16,2))+registros8.DL;

}

else if (valor1==153 ||valor1==154 ||valor1==155 ||valor1==156){

registros8.AH=(registros16.AX&65280)/256; FUNCION QUE SE ENCARGA

registros8.AL=registros16.AX&255; DE MANTENER LOS CAMBIOS DE AX

registros8.BH=(registros16.BX&65280)/256; IGUAL CON AH Y AL

registros8.BL=registros16.BX&255;

registros8.CH=(registros16.CX&65280)/256;

registros8.CL=registros16.CX&255;

registros8.DH=(registros16.DX&65280)/256;

registros8.DL=registros16.DX&255;

}

}

void convertidorhextodec(){

int s;

for (s=0;s<200;s++){

if((op2[s][0]>=48 && op2[s][0]<=57) || (op2[s][0]>=97 && op2[s][0]<=102)) FUNCION QUE LLAMA A LA

s=hextodecop2(s); TRANSFORMACION

else if (op2[s][0]>=91 && ((op2[s][1]>=48 && op2[s][1]<=57) || (op2[s][1]>=97 && op2[s][1]<=102))){

s=hextodecop2c(s);

}

}

for (s=0;s<200;s++){ CONT. FUNCION

if((op1[s][0]>=48 && op1[s][0]<=57) || (op1[s][0]>=97 && op1[s][0]<=102)) TRANSFORMACION

s=hextodecop1(s);

else if (op1[s][0]>=91 && ((op1[s][1]>=48 && op1[s][1]<=57) || (op1[s][1]>=97 && op1[s][1]<=102))){

s=hextodecop1c(s);

}

}

}

void hextobin(){

int b[100], b2[100], b3[100], b4[100],b5[100],k,n,g,cuenta\_unos, cuenta\_ceros;

int b6[100],b7[100],b8[100],b9[100],b10[100],b11[100],b12[100],b13[100];

int b14[100],b15[100],b16[100],b17[100],b18[100];

cuenta\_ceros=0;

cuenta\_unos=0;

k=0;

n=registros16.AX;

while(n>0){

b[k]=n%2;

n=n/2;

k++;

}

k=0;

n=registros16.BX;

while(n>0){

b2[k]=n%2;

n=n/2;

k++;

}

k=0; FUNCION ENCARGADA DE CONVERTIR A BINARIO

n=registros16.CX; Y MODIFICAR LAS BANDERAS

while(n>0){

b3[k]=n%2;

n=n/2;

k++;

}

k=0;

n=registros16.DX;

while(n>0){

b4[k]=n%2;

n=n/2;

k++;

}

k=0;

n=compara;

while(n>0){

b5[k]=n%2;

n=n/2;

k++;

}

k=0;

n=registros8.AH;

while(n>0){

b6[k]=n%2;

n=n/2;

k++;

}

k=0;

n=registros8.AL;

while(n>0){

b7[k]=n%2;

n=n/2;

k++;

}

k=0;

n=registros8.BH;

while(n>0){

b8[k]=n%2;

n=n/2;

k++;

}

k=0;

n=registros8.BL;

while(n>0){

b9[k]=n%2; CONTINUA FUNCION PARA BINARIO Y BANDERAS

n=n/2;

k++;

}

k=0;

n=registros8.CH;

while(n>0){

b10[k]=n%2;

n=n/2;

k++;

}

k=0;

n=registros8.CL;

while(n>0){

b11[k]=n%2;

n=n/2;

k++;

}

k=0;

n=registros8.DH;

while(n>0){

b12[k]=n%2;

n=n/2;

k++;

}

k=0;

n=registros8.DL;

while(n>0){

b13[k]=n%2;

n=n/2;

k++;

}

k=0;

n=segmentos.DS;

while(n>0){

b14[k]=n%2;

n=n/2;

k++;

}

k=0;

n=segmentos.SS;

while(n>0){

b15[k]=n%2;

n=n/2;

k++;

}

k=0; CONTINUA FUNCION PARA BINARIO Y BANDERAS

n=indices.DI;

while(n>0){

b16[k]=n%2;

n=n/2;

k++;

}

k=0;

n=indices.SI;

while(n>0){

b17[k]=n%2;

n=n/2;

k++;

}

k=0;

n=punteros.BP;

while(n>0){

b18[k]=n%2;

n=n/2;

k++;

}

if (valor1==153){

if (registros16.AX>127 || registros16.AX<-128)

banderas.OF=1;

else

banderas.OF=0;

if (registros16.AX==0)

banderas.ZF=1; LAS BANDERAS SE EMPIEZAN A MODIFICAR

else { DEPENDIENDO DEL REGISTRO MODIFICADO

banderas.ZF=0;

banderas.CF=0;}

if (b[0]==1)

banderas.SF=1;

else

banderas.SF=0;

if (registros16.AX>255) //modificacion hecha (65535)

banderas.CF=1;

if (registros16.AX>127)

banderas.AF=1;

else

banderas.AF=0;

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

// printf("Unos: %d\n",cuenta\_unos);

// printf("Ceros: %d\n",cuenta\_ceros);

if (b[g]==1)

cuenta\_unos=cuenta\_unos+1;

else

cuenta\_ceros=cuenta\_ceros+1;

}

if (cuenta\_unos==cuenta\_ceros)

banderas.PF=1;

else

banderas.PF=0;

}

else if(valor1==154){

if (registros16.BX>127 || registros16.BX<-128)

banderas.OF=1;

else

banderas.OF=0;

if (registros16.BX==0)

banderas.ZF=1; CONTINUA MOD DE BANDERAS

else {

banderas.ZF=0;

banderas.CF=0;}

if (b2[0]==1)

banderas.SF=1;

else

banderas.SF=0;

if (registros16.BX>255) //modificacion (65535)

banderas.CF=1;

if (registros16.BX>127)

banderas.AF=1;

else

banderas.AF=0;

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

// printf("Unos: %d\n",cuenta\_unos);

//printf("Ceros: %d\n",cuenta\_ceros);

if (b2[g]==1)

cuenta\_unos=cuenta\_unos+1;

else

cuenta\_ceros=cuenta\_ceros+1;

}

if (cuenta\_unos==cuenta\_ceros)

banderas.PF=1;

else

banderas.PF=0;

}

if (valor1==155){

if (registros16.CX>127 || registros16.CX<-128)

banderas.OF=1;

else

banderas.OF=0;

if (registros16.CX==0)

banderas.ZF=1;

else {

banderas.ZF=0;

banderas.CF=0;}

if (b3[0]==1)

banderas.SF=1;

else

banderas.SF=0;

if (registros16.CX>255) //modificacion (65535)

banderas.CF=1;

if (registros16.CX>127)

banderas.AF=1;

else

banderas.AF=0;

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

//printf("Unos: %d\n",cuenta\_unos);

//printf("Ceros: %d\n",cuenta\_ceros);

if (b3[g]==1)

cuenta\_unos=cuenta\_unos+1;

else

cuenta\_ceros=cuenta\_ceros+1;

} CONT. MOD DE BANDERAS

if (cuenta\_unos==cuenta\_ceros)

banderas.PF=1;

else

banderas.PF=0;

}

if (valor1==156){

if (op1[contador][0]==68){

if (registros16.DX>127 || registros16.DX<-128)

banderas.OF=1;

else

banderas.OF=0;

if (registros16.DX==0)

banderas.ZF=1;

else {

banderas.ZF=0;

banderas.CF=0;}

if (b4[0]==1)

banderas.SF=1;

else

banderas.SF=0;

if (registros16.DX>255) //modificacion (65535)

banderas.CF=1;

if (registros16.DX>127)

banderas.AF=1;

else

banderas.AF=0;

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

//printf("Unos: %d\n",cuenta\_unos);

//printf("Ceros: %d\n",cuenta\_ceros);

if (b4[g]==1)

cuenta\_unos=cuenta\_unos+1;

else

cuenta\_ceros=cuenta\_ceros+1;

}

if (cuenta\_unos==cuenta\_ceros)

banderas.PF=1;

else

banderas.PF=0;

}

else if (op1[contador][0]==83){

if (indices.SI>4095 || indices.SI<-4096) //checar los limites para un registro de 16bits

banderas.OF=1;

else

banderas.OF=0;

if (indices.SI==0)

banderas.ZF=1;

else {

banderas.ZF=0;

banderas.CF=0;}

if (b17[0]==1)

banderas.SF=1;

else CONT. MOD. DE BANDERAS

banderas.SF=0;

if (indices.SI>65535)

banderas.CF=1;

if (indices.SI>127)

banderas.AF=1;

else

banderas.AF=0;

for (g=0;g<4;g++){

// printf("Unos: %d\n",cuenta\_unos);

// printf("Ceros: %d\n",cuenta\_ceros);

if (b17[g]==1)

cuenta\_unos=cuenta\_unos+1;

else

cuenta\_ceros=cuenta\_ceros+1;

}

if (cuenta\_unos==cuenta\_ceros)

banderas.PF=1;

else

banderas.PF=0;

}

}

if (valor1==137){

if (registros8.AH>127 || registros8.AH<-128)

banderas.OF=1;

else

banderas.OF=0;

if (registros8.AH==0)

banderas.ZF=1;

else {

banderas.ZF=0;

banderas.CF=0;}

if (b6[0]==1)

banderas.SF=1;

else

banderas.SF=0;

if (registros8.AH>255)

banderas.CF=1;

if (registros8.AH>127)

banderas.AF=1;

else

banderas.AF=0;

for (g=0;g<4;g++){

// printf("Unos: %d\n",cuenta\_unos);

// printf("Ceros: %d\n",cuenta\_ceros);

if (b6[g]==1)

cuenta\_unos=cuenta\_unos+1;

else

cuenta\_ceros=cuenta\_ceros+1;

}

if (cuenta\_unos==cuenta\_ceros)

banderas.PF=1;

else

banderas.PF=0; CONT. MOD. DE BANDERAS

}

if (valor1==141){

if(op1[contador][0]==65){

if (registros8.AL>127 || registros8.AL<-128)

banderas.OF=1;

else

banderas.OF=0;

if (registros8.AL==0)

banderas.ZF=1;

else {

banderas.ZF=0;

banderas.CF=0;}

if (b7[0]==1)

banderas.SF=1;

else

banderas.SF=0;

if (registros8.AL>255)

banderas.CF=1;

if (registros8.AL>127)

banderas.AF=1;

else

banderas.AF=0;

for (g=0;g<4;g++){

// printf("Unos: %d\n",cuenta\_unos);

// printf("Ceros: %d\n",cuenta\_ceros);

if (b7[g]==1)

cuenta\_unos=cuenta\_unos+1;

else

cuenta\_ceros=cuenta\_ceros+1;

}

if (cuenta\_unos==cuenta\_ceros)

banderas.PF=1;

else

banderas.PF=0;

}

else if (op1[contador][0]==68){

if (indices.DI>127 || indices.DI<-128)

banderas.OF=1;

else

banderas.OF=0;

if (indices.DI==0)

banderas.ZF=1;

else {

banderas.ZF=0;

banderas.CF=0;}

if (b16[0]==1)

banderas.SF=1;

else

banderas.SF=0;

if (indices.DI>255)

banderas.CF=1;

if (indices.DI>127)

banderas.AF=1; CONT. MOD. DE BANDERAS

else

banderas.AF=0;

for (g=0;g<4;g++){

// printf("Unos: %d\n",cuenta\_unos);

// printf("Ceros: %d\n",cuenta\_ceros);

if (b16[g]==1)

cuenta\_unos=cuenta\_unos+1;

else

cuenta\_ceros=cuenta\_ceros+1;

}

if (cuenta\_unos==cuenta\_ceros)

banderas.PF=1;

else

banderas.PF=0;

}

}

if (valor1==138){

if (registros8.BH>127 || registros8.BH<-128)

banderas.OF=1;

else

banderas.OF=0;

if (registros8.BH==0)

banderas.ZF=1;

else {

banderas.ZF=0;

banderas.CF=0;}

if (b8[0]==1)

banderas.SF=1;

else

banderas.SF=0;

if (registros8.BH>255)

banderas.CF=1;

if (registros8.BH>127)

banderas.AF=1;

else

banderas.AF=0;

for (g=0;g<4;g++){

// printf("Unos: %d\n",cuenta\_unos);

// printf("Ceros: %d\n",cuenta\_ceros);

if (b8[g]==1)

cuenta\_unos=cuenta\_unos+1;

else

cuenta\_ceros=cuenta\_ceros+1;

}

if (cuenta\_unos==cuenta\_ceros)

banderas.PF=1;

else

banderas.PF=0;

}

if (valor1==142){

if (registros8.BL>127 || registros8.BL<-128)

banderas.OF=1; CONT. MOD. DE BANDERAS

else

banderas.OF=0;

if (registros8.BL==0)

banderas.ZF=1;

else {

banderas.ZF=0;

banderas.CF=0;}

if (b9[0]==1)

banderas.SF=1;

else

banderas.SF=0;

if (registros8.BL>255)

banderas.CF=1;

if (registros8.BL>127)

banderas.AF=1;

else

banderas.AF=0;

for (g=0;g<4;g++){

// printf("Unos: %d\n",cuenta\_unos);

// printf("Ceros: %d\n",cuenta\_ceros);

if (b9[g]==1)

cuenta\_unos=cuenta\_unos+1;

else

cuenta\_ceros=cuenta\_ceros+1;

}

if (cuenta\_unos==cuenta\_ceros)

banderas.PF=1;

else

banderas.PF=0;

}

if (valor1==139){

if (registros8.CH>127 || registros8.CH<-128)

banderas.OF=1;

else

banderas.OF=0;

if (registros8.CH==0)

banderas.ZF=1;

else {

banderas.ZF=0;

banderas.CF=0;}

if (b10[0]==1)

banderas.SF=1;

else

banderas.SF=0;

if (registros8.CH>255)

banderas.CF=1;

if (registros8.CH>127)

banderas.AF=1;

else

banderas.AF=0;

for (g=0;g<4;g++){

// printf("Unos: %d\n",cuenta\_unos);

// printf("Ceros: %d\n",cuenta\_ceros);

if (b10[g]==1) CONT. MOD. DE BANDERAS

cuenta\_unos=cuenta\_unos+1;

else

cuenta\_ceros=cuenta\_ceros+1;

}

if (cuenta\_unos==cuenta\_ceros)

banderas.PF=1;

else

banderas.PF=0;

}

if (valor1==143){

if (registros8.CL>127 || registros8.CL<-128)

banderas.OF=1;

else

banderas.OF=0;

if (registros8.CL==0)

banderas.ZF=1;

else {

banderas.ZF=0;

banderas.CF=0;}

if (b11[0]==1)

banderas.SF=1;

else

banderas.SF=0;

if (registros8.CL>255)

banderas.CF=1;

if (registros8.CL>127)

banderas.AF=1;

else

banderas.AF=0;

for (g=0;g<4;g++){

// printf("Unos: %d\n",cuenta\_unos);

// printf("Ceros: %d\n",cuenta\_ceros);

if (b11[g]==1)

cuenta\_unos=cuenta\_unos+1;

else

cuenta\_ceros=cuenta\_ceros+1;

}

if (cuenta\_unos==cuenta\_ceros)

banderas.PF=1;

else

banderas.PF=0;

}

if (valor1==140){

if (registros8.DH>127 || registros8.DH<-128)

banderas.OF=1;

else

banderas.OF=0;

if (registros8.DH==0)

banderas.ZF=1;

else {

banderas.ZF=0;

banderas.CF=0;} CONT. MOD. DE BANDERAS

if (b12[0]==1)

banderas.SF=1;

else

banderas.SF=0;

if (registros8.DH>255)

banderas.CF=1;

if (registros8.DH>127)

banderas.AF=1;

else

banderas.AF=0;

for (g=0;g<4;g++){

// printf("Unos: %d\n",cuenta\_unos);

// printf("Ceros: %d\n",cuenta\_ceros);

if (b12[g]==1)

cuenta\_unos=cuenta\_unos+1;

else

cuenta\_ceros=cuenta\_ceros+1;

}

if (cuenta\_unos==cuenta\_ceros)

banderas.PF=1;

else

banderas.PF=0;

}

if (valor1==144){

if (registros8.DL>127 || registros8.DL<-128)

banderas.OF=1;

else

banderas.OF=0;

if (registros8.DL==0)

banderas.ZF=1;

else {

banderas.ZF=0;

banderas.CF=0;}

if (b13[0]==1)

banderas.SF=1;

else

banderas.SF=0;

if (registros8.DL>255)

banderas.CF=1;

if (registros8.DL>127)

banderas.AF=1;

else

banderas.AF=0;

for (g=0;g<4;g++){

// printf("Unos: %d\n",cuenta\_unos);

// printf("Ceros: %d\n",cuenta\_ceros);

if (b13[g]==1)

cuenta\_unos=cuenta\_unos+1;

else

cuenta\_ceros=cuenta\_ceros+1;

}

if (cuenta\_unos==cuenta\_ceros)

banderas.PF=1; CONT. MOD. DE BANDERAS

else

banderas.PF=0;

}

if (valor1==151){

if (segmentos.DS>4095 || segmentos.DS<-4096) //checar los limites para un registro de 16bits

banderas.OF=1;

else

banderas.OF=0;

if (segmentos.DS==0)

banderas.ZF=1;

else {

banderas.ZF=0;

banderas.CF=0;}

if (b14[0]==1)

banderas.SF=1;

else

banderas.SF=0;

if (segmentos.DS>65535)

banderas.CF=1;

if (segmentos.DS>127)

banderas.AF=1;

else

banderas.AF=0;

for (g=0;g<4;g++){

// printf("Unos: %d\n",cuenta\_unos);

// printf("Ceros: %d\n",cuenta\_ceros);

if (b14[g]==1)

cuenta\_unos=cuenta\_unos+1;

else

cuenta\_ceros=cuenta\_ceros+1;

}

if (cuenta\_unos==cuenta\_ceros)

banderas.PF=1;

else

banderas.PF=0;

}

if (valor1==166){

if (segmentos.SS>4095 || segmentos.SS<-4096) //checar los limites para un registro de 16bits

banderas.OF=1;

else

banderas.OF=0;

if (segmentos.SS==0)

banderas.ZF=1;

else {

banderas.ZF=0;

banderas.CF=0;}

if (b15[0]==1)

banderas.SF=1;

else

banderas.SF=0;

if (segmentos.SS>65535)

banderas.CF=1;

if (segmentos.SS>127) CONT. MOD. DE BANDERAS

banderas.AF=1;

else

banderas.AF=0;

for (g=0;g<4;g++){

// printf("Unos: %d\n",cuenta\_unos);

// printf("Ceros: %d\n",cuenta\_ceros);

if (b15[g]==1)

cuenta\_unos=cuenta\_unos+1;

else

cuenta\_ceros=cuenta\_ceros+1;

}

if (cuenta\_unos==cuenta\_ceros)

banderas.PF=1;

else

banderas.PF=0;

}

if (valor1==146){

if (punteros.BP>4095 || punteros.BP<-4096) //checar los limites para un registro de 16bits

banderas.OF=1;

else

banderas.OF=0;

if (punteros.BP==0)

banderas.ZF=1;

else {

banderas.ZF=0;

banderas.CF=0;}

if (b18[0]==1)

banderas.SF=1;

else

banderas.SF=0;

if (punteros.BP>65535)

banderas.CF=1;

if (punteros.BP>127)

banderas.AF=1;

else

banderas.AF=0;

for (g=0;g<4;g++){

// printf("Unos: %d\n",cuenta\_unos);

// printf("Ceros: %d\n",cuenta\_ceros);

if (b18[g]==1)

cuenta\_unos=cuenta\_unos+1;

else

cuenta\_ceros=cuenta\_ceros+1;

}

if (cuenta\_unos==cuenta\_ceros)

banderas.PF=1;

else

banderas.PF=0;

}

if (funcion==1){

//printf("compara: %d\n",compara);

if (compara>127 || compara<-128)

banderas.OF=1;

else

banderas.OF=0; CONT. MOD. DE BANDERAS

if (compara==0){

//printf("Bandera Zero %d\n",banderas.ZF);

banderas.ZF=1;

}

else {

banderas.ZF=0;

banderas.CF=0;}

if (b5[0]==1)

banderas.SF=1;

else

banderas.SF=0;

if (compara>255) //modificacion (65535)

banderas.CF=1;

if (compara>127)

banderas.AF=1;

else

banderas.AF=0;

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

// printf("Unos: %d\n",cuenta\_unos);

// printf("Ceros: %d\n",cuenta\_ceros);

if (b5[g]==1)

cuenta\_unos=cuenta\_unos+1;

else

cuenta\_ceros=cuenta\_ceros+1;

}

if (cuenta\_unos==cuenta\_ceros)

banderas.PF=1;

else

banderas.PF=0;

}

}

void pantallaprincipal(){

int f;

f=0;

int t;

printf("\n\n\n");

printf("Special Purpose Registers\n\nIP %x\n",especiales.IP);

printf("Segment Registers\n\nDS %x SS %x CS %x \n--------------------\n",segmentos.DS,segmentos.SS,segmentos.CS);

printf("Index Registers\n\nDI %x SI %x \n--------------------\n",indices.DI,indices.SI);

printf("Pointer Registers\n\nBP %x SP %x \n--------------------\n",punteros.BP,punteros.SP);

printf("General Purpose Registers\n\n AX %x \t BX %x \t CX %x \tDX %x \n\n",registros16.AX,registros16.BX, registros16.CX,registros16.DX);

printf("AH %x AL %x | BH %x BL %x | CH %x CL %x | DH %x DL %x\n------------------------------------------------------------\n", registros8.AH,registros8.AL,registros8.BH,registros8.BL,registros8.CH,registros8.CL,registros8.DH,registros8.DL);

printf("Flags Register\n\nCF %x PF %x AF %x ZF %x SF %x TF %x IF %x DF %x OF %x\n----------------\n",banderas.CF,banderas.PF,banderas.AF,banderas.ZF,banderas.SF,banderas.TF,banderas.IF,banderas.DF,banderas.OF);

printf("Program Memory %x:%x || %x:%x || %x:%x\n\n",segmentos.DS,indices.DI,segmentos.DS,indices.SI,segmentos.SS,punteros.BP);

f=0;

printf(" 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 \n"); IMPRESION

printf("------------------------------------------------------------------------------ \n"); DE RESULTADOS EN

for (t=24576;t<24784;t++){ PANTALLA

deci=memoriaext.memoria[t];

sprintf(hx,"%x",deci);

memoriaext1.memoria[t]=hx;

//if (i==0 || i==15 || i==30 || i==45 || i==60 || i==75 || i==90 || i==105 || i==120 || i==135 || i==150)

if (f==16){

printf("\n");

f=0;

}

else if (strlen(memoriaext1.memoria[t])>2)

printf("");

else if (strlen(memoriaext1.memoria[t])>1)

printf(" ");

else if (strlen(memoriaext1.memoria[t])>3)

printf("");

//else if (strlen(memoriaext1.memoria[i])==1)

// printf(" ");

f=f+1;

printf(" %s ",memoriaext1.memoria[t]);

}

/\*for (i=0;i<150;i++){

if (i==0 || i==15 || i==30 || i==45 || i==60 || i==75 || i==90 || i==105 || i==120 || i==135 || i==150)

printf(" \n");

else

printf(" ");

printf("%d",memoriaext.memoria[i]);

}\*/

}