* 1. Práctica 5
  2. Assembly, Instrucciones y Simulador MSX88

***Objetivos de la práctica: que el alumno domine***

* ***Las instrucciones básicas del lenguaje assembly del simulador MSX88.***
* ***Los diferentes modos de direccionamiento.***
* ***El diseño de programas utilizando instrucciones de salto condicional.***

***Bibliografía:***

* ***Apunte 4 de la cátedra, “Lenguaje Assembly”.***
* ***Manual del simulador MSX88.***
* ***Set de Instrucciones de MSX88.***

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**1)** Dada la siguiente definición de datos y el código: **F = [(A+B)/C]-D**

nombre tamaño valor

A: 1 byte 6

B: 1 byte 4

C: 1 byte 2

D: 1 byte 1

F: 1 byte ?

Suponiendo que se poseen las instrucciones necesarias en cada caso, escribir el programa que implemente el código anterior utilizando máquinas de 1, 2 ó 3 direcciones.

|  |  |  |
| --- | --- | --- |
| **Maq. de 1 dirección** | **Maq. de 2 direcciones** | **Maq. de 3 direcciones** |
| Load a  add b  div C  sub D  store F | Mov f,a  add f,b  dif f,c  sub f,d | Add f,a,b  div f,f,c  sub f,f,d |

**2) Suponga que cada código de operación ocupa 6 bits y las direcciones son de 10 bits. Analice las soluciones implementadas en el ejercicio anterior y complete la siguiente tabla:**

|  |  |  |  |
| --- | --- | --- | --- |
|  | **M. de 1 dirección** | **M. de 2 direcciones** | **M. de 3 direcciones** |
| **Tamaño del programa en memoria**  **(cod.operación + operandos)** | (5\*6)+(5\*10)=80bits  =10 bytes | 4\*6+ (4\*(10+10)=104 bits  =13 bytes | (3\*6)+(3\*(10+10+10))=108 bits  =13,5 bytes |
| **Cantidad de accesos a memoria (instrucciones + operandos)** | 5+5= 10 accsesos | 4+(2+3+3+3)=15 accesos | 3+(3+3+3)=12 accesos |

**3)** Dado el siguiente código: **F = ((A - B)\*C) + (D/E)**;

1. Implemente el código utilizando máquinas de 1, 2 y 3 direcciones.
2. Realice una tabla de comparación similar a la del ejercicio 2.
3. ¿Cuál máquina elegiría haciendo un balance de la cantidad de instrucciones, el espacio en memoria ocupado y el tiempo de ejecución (1 acceso a memoria = 1 ms)? ¿Es ésta una conclusión general?

Para cada programa propuesto en los siguientes ejercicios, deberá editar el archivo fuente con extensión asm (ej: ejer1.asm), luego ensamblarlo usando asm88.exe (comando: asm88 ejer1.asm) y enlazarlo con link88.exe (comando: link88 ejer1.o). Cada archivo obtenido con extensión eje (ej: ejer1.eje) deberá ser cargado y ejecutado en el simulador MSX88.

**4)** El siguiente programa utiliza una **instrucción de transferencia de datos** (instrucción MOV) con diferentes modos de direccionamiento para referenciar sus operandos. Ejecutar y analizar el funcionamiento de cada instrucción en el Simulador MSX88 observando el flujo de información a través del BUS DE DATOS, el BUS DE DIRECCIONES, el BUS DE CONTROL, el contenido de REGISTROS, de posiciones de MEMORIA, operaciones en la ALU, etc.

ORG 1000h

NUM0 DB 0CAh; declaracion de las variables, el primer dato siempre esta en 100h.

NUM1 DB 0

NUM2 DW ?

NUM3 DW 0ABCDh

NUM4 DW ?

Mover=copier lo que esta en un dato/direccion a otro lado

ORG 2000H

MOV BL, NUM0 muevo num0 a bl

MOV BH, 0FFh muevo 0ffh a bh

MOV CH, BL muevo bl a ch

MOV AX, BX muevo todo bx a ax

MOV NUM1, AL muevo al a num1

MOV NUM2, 1234h muevo 1234h a num2

MOV BX, OFFSET NUM3 muevo la direccion de num3 a bx

MOV DL, [BX] muevo a lo que esta apuntando la direccionde bx a dl

MOV AX, [BX] muevo a lo que esta apuntando la direccionde bx a ax

MOV BX, 1006h muevo 1006h a bx

MOV WORD PTR [BX], 0CDEFh PTR es el operador “puntero” que indica que [BX]

;apunta a un dato tipo WORD

HLT

END

Cuestionario:

1. Explicar detalladamente qué hace cada instrucción MOV del programa anterior, en función de sus operandos y su modo de direccionamiento.
2. Confeccionar una tabla que contenga todas las instrucciones MOV anteriores, el modo de direccionamiento y el contenido final del operando destino de cada una de ellas.

Instrucción método de direccionamiento destino

MOV BL,NUM0 DIRECTO BL:=CAH

MOV BH,0FFH INMEDIATO BH:=FFH

MOV CH,BL POR REGISTRO CH:=CAH

MOV AX,BX POR REGISTRO AX:=FFCAH

MOV NUM1,AL DIRECTO 1001H:=CAH

MOV NUM2,1234H INMEDIATO 1002H:=34H1003H:=12H

MOV BX,OFFSET NUM3 INMEDIATO BX:=1004H

MOV DL,[BX] INDIRECTO POR REG DL:=CDH

MOV AX,[BX] INDIRECTO POR REG AX:=ABCDH

MOV BX,1006H INMEDIATO BX:=1006H

MOV WORDPTR[BX],1006H INDIRECTO POR REG 1006H:=06H1007H:=10H

1. Notar que durante la ejecución de algunas instrucciones MOV aparece en la pantalla del simulador un registro temporal denominado “**ri**”, en ocasiones acompañado por otro registro temporal denominado “**id**”. Explicar con detalle que función cumplen estos registros.

RI=registro temporal de direcciones ( solo aparece cuando es necesario). Por ejemplo cuando se desea sumar a un registro el valor de una variable, en este caso el registro RI contendría temporalmente la posiocion de memoria de la variable..

**5)** El siguiente programa utiliza diferentes **instrucciones de procesamiento de datos** (instrucciones aritméticas y lógicas). Analice el comportamiento de ellas y ejecute el programa en el MSX88.

ORG 1000H

NUM0 DB 80h

NUM1 DB 200

NUM2 DB -1

BYTE0 DB 01111111B

BYTE1 DB 10101010B

ORG 2000H

MOV AL, NUM0

ADD AL, AL

INC NUM1

MOV BH, NUM1

MOV BL, BH

DEC BL

SUB BL, BH

MOV CH, BYTE1

AND CH, BYTE0

NOT BYTE0

OR CH, BYTE0

XOR CH, 11111111B

HLT

END

Cuestionario:

1. ¿Cuál es el estado de los FLAGS después de la ejecución de las instrucciones ADD y SUB del programa anterior? Justificar el estado (1 ó 0) de cada uno de ellos. ¿Dan alguna indicación acerca de la correctitud de los resultados?

Depues del add están en= 1101 estan activadas el overflor, el carry y el cero porque sumo un numero muy grande que no entra para guardarlo.

Después del sub están en=1010 porque el resultado es -1 en ca2 y al ser un numero muy grande queda el overflow (¿)

1. ¿Qué cadenas binarias representan a NUM1 y NUM2 en la memoria del simulador? ¿En qué sistemas binarios están expresados estos valores?

Num0:= 10000000

Num1:= 11001000

1. Confeccionar una tabla que indique para cada operación aritmética ó lógica del programa, el valor de sus operandos, en qué registro o dirección de memoria se almacenan y el resultado de cada operación.

ORG 2000H

MOV AL,NUM0 ; AL:= 80H

ADD AL,AL ; AL:=AL + AL = 80H + 80H = 00H ZNVC=1011

INC NUM1 ; NUM1:=NUM1 + 1 = C8 +1 = C9

MOV BH,NUM1; BH:= C9H

MOV BL,BH ; BL:= C9H

DEC BL; BL:=C9 – 1 = C8H

SUB BL,BH; BL:=BL – BH= C8 – C9 = FFH ( -1 ) ZNVC=0101

MOV CH,BYTE1; CH:= AAH

AND CH,BYTE0 ; CH:= AA AND 7F = 2AH

NOT BYTE0 ; Contenido de 1003H := 80H

OR CH,BYTE0; CH:= 2AH OR 80H = AAH

XOR CH,11111111B CH:= AAH XOR FFH = 55H

HLT

END

**6)** El siguiente programa implementa un contador utilizando una **instrucción de transferencia de control**. Analice el funcionamiento de cada instrucción y en particular las del lazo repetitivo que provoca la cuenta.

ORG 1000H

INI DB 0

FIN DB 15

ORG 2000H

MOV AL, INI

MOV AH, FIN

SUMA: INC AL

CMP AL, AH

JNZ SUM

HLT

END

Cuestionario:

1. ¿Cuántas veces se ejecuta el lazo? ¿De qué variables depende esto en el caso general?

Se hace 15 veces

1. Analice y ejecute el programa reemplazando la instrucción de salto condicional JNZ por las siguientes, indicando en cada caso el contenido final del registro AL:

1o) JS:=0f

2o) JZ:=01

3o) JMP:=infinito

**7)** Escribir un programa en lenguaje assembly del MSX88 que implemente la sentencia condicional de un lenguaje de alto nivel IF **A** < **B** THEN **C** = **A** ELSE **C** = **B**. Considerar que las variables de la sentencia están almacenadas en los registros internos de la CPU del siguiente modo **A** en AL, **B** en BL y **C** en CL.

Determine las modificaciones que debería hacer al programa si la condición de la sentencia IF fuera:

a) A  B

CMP AL, BL

JZ salto1 ; si es cero el resultado (Z=1) se cumple AL=BL

JS salto1 ; si no es cero y si hay signo (es negativo) se cumple AL<BL

MOV CL, BL ; instrucción del else

JMP salto2

salto1 : MOV CL, AL ; instrucción del then

salto2 : (sigue el programa)

b) A = B esta arriba

**8)** El siguiente programa suma todos los elementos de una tabla almacenada a partir de la dirección 1000H de la memoria del simulador. Analice el funcionamiento y determine el resultado de la suma. Comprobar resultado en el MSX88.

ORG 1000H

TABLA DB 2,4,6,8,10,12,14,16,18,20

FIN DB ?

TOTAL DB ?

MAX DB 13

ORG 2000H

MOV AL, 0

MOV CL, OFFSET FIN - OFFSET TABLA

MOV BX, OFFSET TABLA

SUMA: ADD AL, [BX]

INC BX

DEC CL

JNZ SUMA

HLT

END

¿Qué modificaciones deberá hacer en el programa para que el mismo almacene el resultado de la suma en la celda etiquetada TOTAL?

Tendrias que agregar un mov al,bx

**9)** Escribir un programa que, utilizando las mismas variables y datos que el programa del punto anterior (TABLA, FIN, TOTAL, MAX), determine cuántos de los elementos de TABLA son menores o iguales que MAX. Dicha cantidad debe almacenarse en la celda TOTAL.

ORG 1000H

TABLA DB 2,4,6,8,10,12,14,16,18,20

FIN DB ?

TOTAL DB 0

MAX DB 13

ORG 2000H

MOV AL, 0

MOV CL, OFFSET FIN - OFFSET TABLA

MOV BX, OFFSET TABLA

SUMA: ADD AL, [BX]

CMP AL, MAX

JS salto1

JMP salto2

salto1: inc TOTAL

salto2: DEC CL

JNZ SUMA

HLT

END

**10)** Analizar el funcionamiento del siguiente programa.

ORG 2000H

MOV AX, 1

MOV BX, 1000h

CARGA: MOV [BX], AX

ADD BX, 2

ADD AX, AX

CMP AX, 200

JS CARGA

HLT

END

Cuestionario:

1. El programa genera una tabla. ¿Cómo están relacionados sus elementos entre sí?
2. ¿A partir de qué dirección de memoria se crea la tabla? ¿Cuál es la longitud de cada uno de sus elementos (medida en bits)?
3. ¿Cuántos elementos tiene la tabla una vez finalizada la ejecución del programa? ¿De qué depende esta cantidad?

**11)** Escribir un programa que genere una tabla a partir de la dirección de memoria almacenada en la celda DIR con los múltiplos de 5 desde cero hasta MAX.

**12)** Escribir un programa que, dado un número X, genere un arreglo con todos los resultados que se obtienen hasta llegar a 0, aplicando la siguiente fórmula: si X es par, se le resta 7; si es impar, se le suma 5, y al resultado se le aplica nuevamente la misma fórmula. Ej: si X = 3 entonces el arreglo tendrá: 8, 1, 6, -1, 4, -3, 2, -5, 0.

**13)** Dada la frase "Organización y la Computación", almacenada en la memoria, escriba un programa que determine cuantas letras ‘a’ seguidas de ‘c’ hay en ella.