

Arquitectura 2023

Explicación 1


Práctica 1

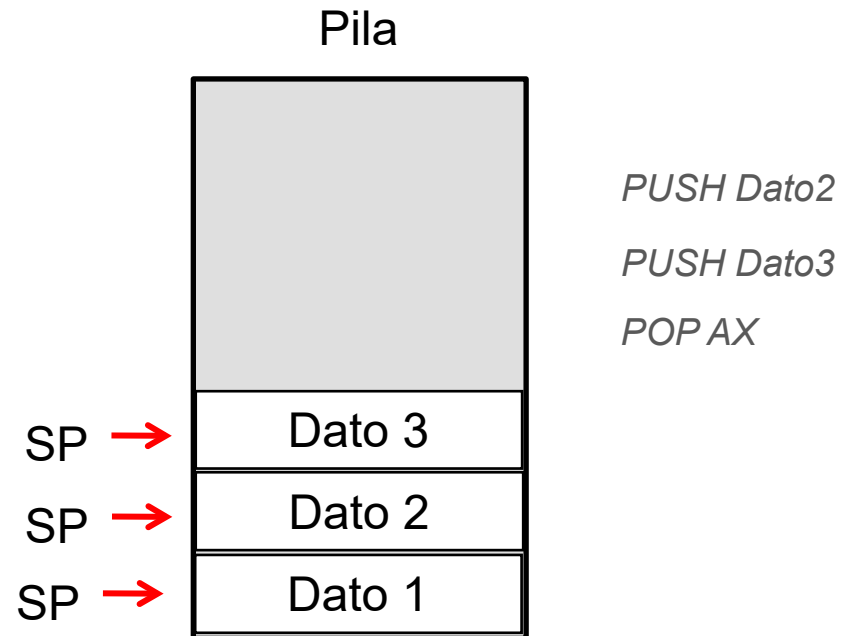
Subrutinas y pasaje de parámetros

Objetivos: *Comprender la utilidad de las **subrutinas** y la comunicación con el programa principal a través de una **pila**.*

*Escribir programas en el lenguaje assembly del simulador MSX88.
Ejecutarlos y verificar los resultados, analizando el flujo de información entre los distintos componentes del sistema.*

Pila

- Sector de la memoria con acceso **LIFO**.
 - Se requiere un registro Puntero de Pila (**SP**). Este se encuentra (*de forma implícita*) en la cabeza de la pila.
 - Operaciones sobre la pila
 - ❑ **PUSH** ; operación de Apilar
 - ❑ **POP** ; operación de Desapilar
- 



Ejemplo en Assembly

Recordar!
Siempre se apilan
registros de 16 bits)

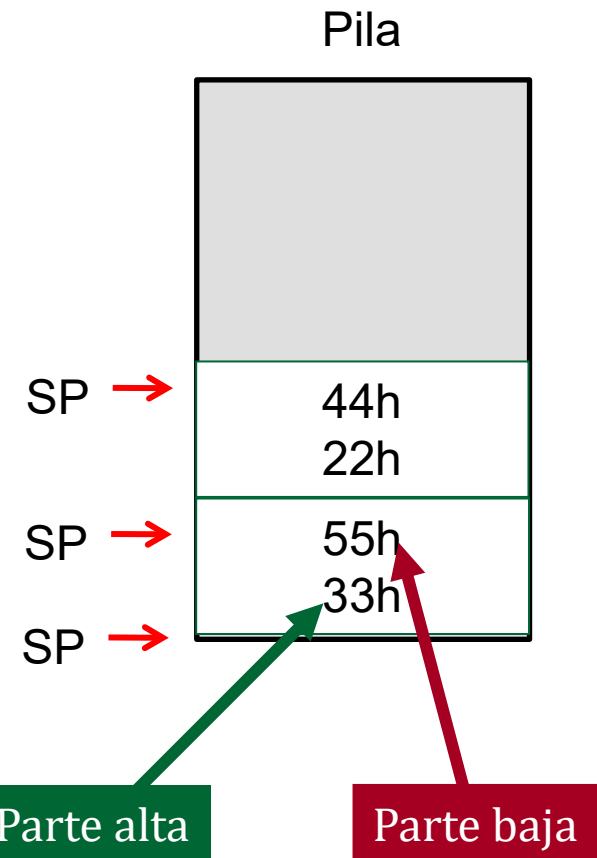
Ejemplo:

```
ORG 3000H
Datos DB 55h, 33h, 44h, 22h
```

```
ORG 2000H
MOV BX, 3000H
MOV AX, [BX]
PUSH AX
MOV BX, 3002H
MOV CX, [BX]
PUSH CX
POP AX
POP CX
HLT
END
```

3000h	55h
3001h	33h
3002h	44h
3003h	22h
3003h	

AX	22h 44h
BX	3002h
CX	33h 55h



Subrutinas

Módulos de programas que brindan economía (*código usado varias veces*) y modularidad (*subdivisión en unidades pequeñas*).

Puede invocarse desde cualquier punto de un programa mediante instrucción **CALL**. Para retornar de la subrutina se utiliza la instrucción **RET**

Pueden requerir pasaje de argumentos (*parámetros*)

- ❑ **por valor** (*copia de una variable*)
- ❑ **por referencia** (*dirección de la variable*)

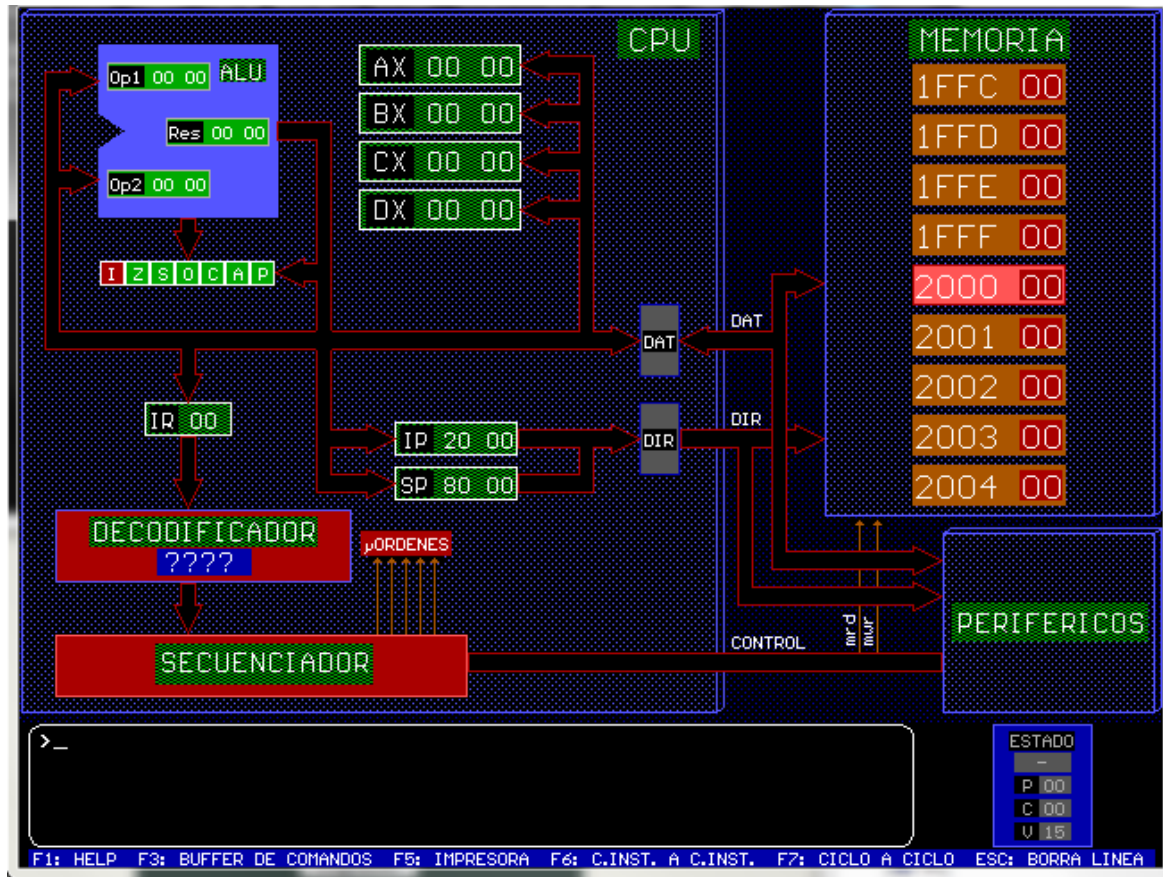
MSX88

MSX88 ofrece al usuario múltiples formas de analizar la evolución dinámica de los distintos elementos que conforman un sistema microprocesador simulado, durante la ejecución de un programa.

El kit de aprendizaje MSX88, está compuesto por las siguientes herramientas:

- **ASM88:** Ensamblador para la CPU SX88.
- **LINK88:** Programa montador para el MSX88
- **MSX88:** Emulador del sistema microcomputador, cuya CPU es SX88.

MSX88 - BLOQUES CONSTITUTIVOS



• CPU SX88:

Versión simplificada de la CPU 8088 de Intel.

• Memoria:

En total existen 64 Kbytes.

• Periferia:

Múltiples conexiones son posibles entre ésta y los dos bloques anteriores.

• Programa monitor:

Pequeño sistema operativo de que dispone MSX88.

MSX88 - Lenguaje de máquina

Instrucciones

Las instrucciones del SX88 están codificadas con cero, uno o dos operandos.

Los operandos pueden ser: **registro, memoria, dato inmediato.**

Las operaciones **nunca** puede realizarse entre **memoria y memoria.**

Modos de direccionamiento:

- **Direccionamiento inmediato.**

- **Direccionamiento directo.**

En la instrucción se indica la dirección real de memoria en la que está contenido el operando.
Si este ocupa varias posiciones de memoria, figurará la dirección más baja.

- **Direccionamiento indirecto a través del registro BX.**

La dirección de memoria donde está el operando viene determinada por BX.

MSX88 - TIPOS DE INSTRUCCIONES

Desde un punto de vista funcional se pueden distinguir los siguientes tipos de instrucciones:

- Instrucciones de transferencia de datos: **MOV**.
- Instrucciones aritmético-lógicas:
 - Instrucciones aritméticas: **ADD, ADC, SUB, SBB**.
 - Instrucciones lógicas: **AND, OR, XOR, NEG, NOT**.
- Instrucciones de comparación: **CMP**.
- Instrucciones de incremento/decremento: **INC, DEC**.
- Instrucciones de manejo de la Pila: **PUSH, POP**
- Instrucciones de cambio de flujo de programa:
 - Instrucciones de salto incondicional: **JMP**.
 - Instrucciones de salto condicional: **JZ, JNZ, JS, JNS, JC, JNC, JO, JNO**.
- Instrucciones asociadas a subrutinas: **CALL, RET**.
- Instrucciones de control: **NOP, HLT**.

Práctica 1

Ej 1)

Multiplicación de números sin signo.

Escribir un programa que calcule el producto entre dos números sin signo almacenados en la memoria del microprocesador:

Práctica 1

1.1) Sin hacer llamados a subrutinas, resolviendo el problema desde el programa principal;

; Memoria de Datos

ORG 1000H

NUM1 **DB** 05H

NUM2 **DB** 03H

; Memoria de Instrucciones

ORG 2000H

MOV AL, NUM1

CMP AL, 0

JZ FIN

MOV AH, 0

MOV DX, 0

MOV CL, NUM2

LOOP: CMP CL, 0

JZ FIN

ADD DX, AX

DEC CL

JMP LOOP

FIN: HLT

END

1000H	5H
1001H	3H
1002H	
1003H	

AX	0	5H
BX		
CX		0H
DX	0	FH

Práctica 1

1.2) Llamando a una subrutina **MUL** para efectuar la operación, pasando los parámetros por **valor** desde el programa principal a través de registros;

; Memoria de Datos

```
ORG 1000H
NUM1  DB  05H
NUM2  DB  03H
```

; Memoria de Instrucciones

ORG 3000H ; Subrutina MUL

```
MUL:  CMP AL, 0
      JZ  FIN
      CMP CL, 0
      JZ  FIN
      MOV AH, 0
      MOV DX, 0
LAZO: ADD DX, AX
      DEC CL
      JNZ LAZO
FIN:   RET
```

ORG 2000H ; Programa principal

```
MOV AL, NUM1
MOV CL, NUM2
CALL MUL
HLT
END
```

7FFDH		SP
7FFEH	IP L	
7FFFH	IP H	
8000H		SP

1000H	5H
1001H	3H
1002H	
1003H	

AX	0	5H
BX		
CX		0H
DX	0	FH

Práctica 1

1.3) Llamando a una subrutina MUL, pasando los parámetros por referencia desde el programa principal a través de registros.

; Memoria de datos

ORG 1000H

NUM1 DW 5H ; *NUM1 y NUM2 >0*

NUM2 DW 3H

; Memoria de Instrucciones

ORG 3000H ; Subrutina MUL

MUL: MOV DX, 0

LAZO: MOV BX, AX

ADD DX, [BX]

PUSH DX

MOV BX, CX

MOV DX, [BX]

DEC DX

MOV [BX], DX

POP DX

JNZ LAZO

RET

ORG 2000H ; Programa principal

MOV AX, OFFSET NUM1

MOV CX, OFFSET NUM2

CALL MUL

HLT

END

7FF9H		
7FFAH		
7FFBH		
7FFCH		
7FFDH	0FH	SP
	0	
7FFEH		
	IP L	SP
7FFFH	IP H	
8000H		SP

1000H	5H
1001H	
1002H	0H
1003H	

AX	10	00
BX	10	02
CX	10	02
DX	0	0FH

Práctica 1

2) Escribir un programa que calcule el producto entre dos números sin signo almacenados en la memoria del microprocesador llamando a una subrutina MUL, pero en este caso pasando los parámetros por valor y por referencia a través de la pila.

; Memoria de datos

ORG 1000H

NUM1 DW 5H

NUM2 DW 3H

RES DW ?

; Programa principal

ORG 2000H

MOV AX, NUM1

PUSH AX

MOV AX, NUM2

PUSH AX

MOV AX, OFFSET RES

PUSH AX

MOV DX, 0

CALL MUL

POP AX

POP AX

POP AX

HLT

END

MUL:

ORG 3000H ; Subrutina MUL

PUSH BX

MOV BX, SP

PUSH CX

PUSH AX

PUSH DX

ADD BX, 6

MOV CX, [BX]

ADD BX, 2

MOV AX, [BX]

SUMA:

ADD DX, AX

DEC CX

JNZ SUMA

SUB BX, 4

MOV AX, [BX]

MOV BX, AX

MOV [BX], DX

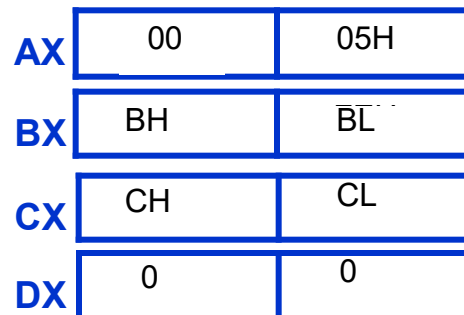
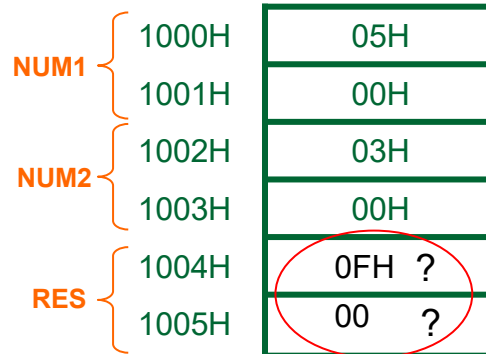
POP DX

POP AX

POP CX

POP BX

RET



7FF0H	0	SP
7FF1H	0	
7FF2H	04H	SP
7FF3H	10	
7FF4H	CL	SP
7FF5H	CH	
7FF6H	BL	SP
7FF7H	BH	
7FF8H	IP retorno L	SP
7FF9H	IP retorno H	
7FFAH	04H	SP
7FFBh	10H	
7FFCh	03H	SP
7FFDH	00H	
7FFEh	05H	SP
7FFFH	00H	
8000H		SP

¿Preguntas?