MÓDULO ASSEMBLER

Pasaje de parámetros por pila

Autores:
Alejandro Héctor Gonzalez
Silvana Lis Gallo
Junio 2021

RESUMEN

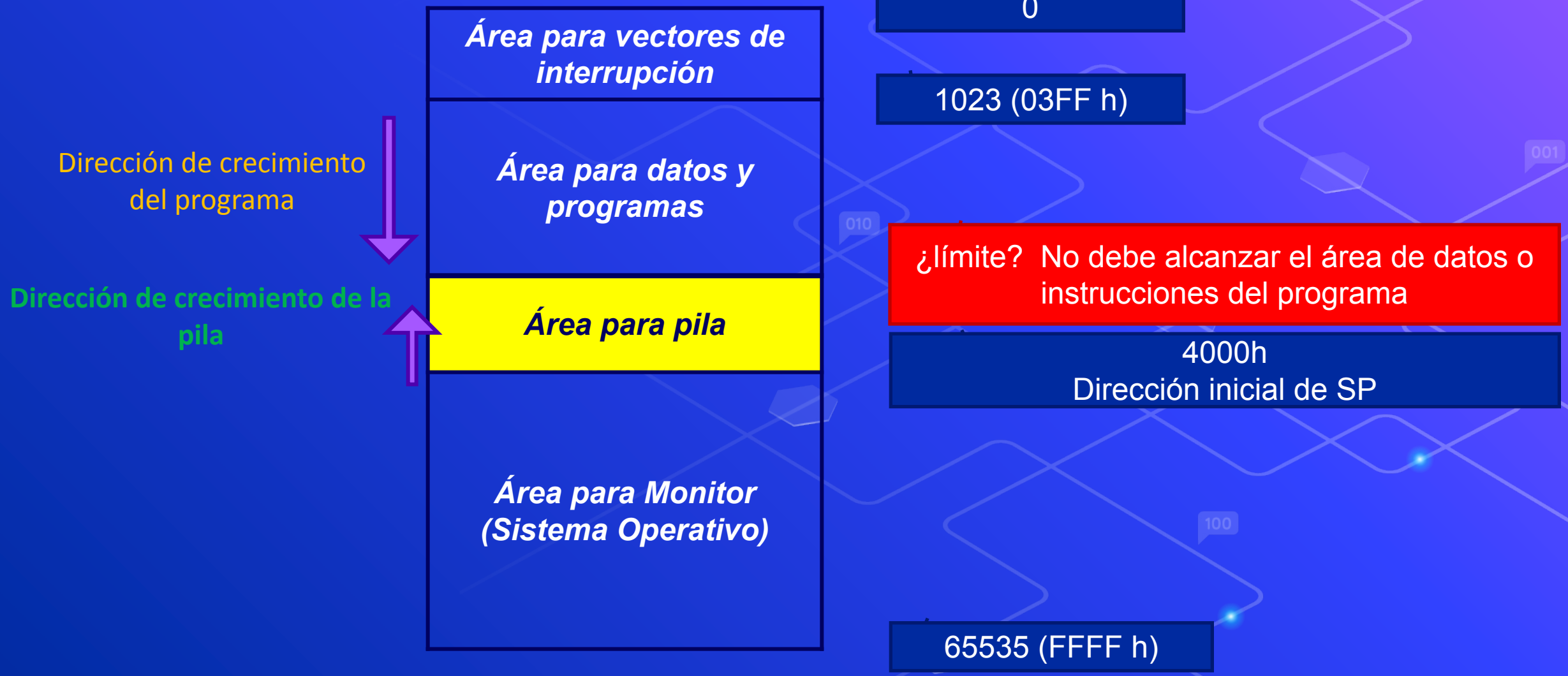
En esta clase se trabajan ejemplos de pasaje de parámetros por valor y por referencia vía pila.
Se explica el concepto de rotación y se trabaja con análisis de caracteres a bajo nivel

Palabras clave

pila, parámetros, rotación, caracteres

Pasaje de parámetros por PILA

Pila - Memoria



Subrutinas – Tipos de pasaje de Parámetros

- Vía Registro
  - Por valor
  - Por referencia
- Vía PILA
  - Por valor
  - Por referencia

Parámetros vía PILA por valor

Por Pila y Valor

Tamaño: siempre 16 bits

Pasaje: cargar registros, apilar, llamar a subrutina, desapilar registros.

Uso: recuperar los valores de la pila con BX, para acceder a los valores.

Importante: hay que tener cuidado al manipular la pila desde la subrutina. Si no se pone atención en el orden en que se apilan los parámetros, la dirección de retorno de la subrutina podría perderse.

Parámetros vía PILA por valor

```
MOV AX, MiVar
PUSH AX          ; solo 1 parámetro
CALL SubSuma2
POP AX
…
                    ORG 3000H
SubSuma2:    MOV BX, SP  ; recupera dirección de pila (3FFCH)
             ADD BX, 2   ; para acceder al parámetro (3FFEH)
             ADD AX, [BX]; accede al valor de MiVar
             …
             RET
```

Parámetros vía PILA por referencia

Por Pila y Referencia

Tamaño: siempre 16 bits

Pasaje: cargar registros usando OFFSET, apilar, llamar a subrutina, desapilar registros.

Uso: recuperar los valores de la pila con BX, y luego volver a usar BX indirectamente para acceder al dato.

Recordar que lo que hay en la pila es una referencia al valor y no el valor del parámetro. Hay 2 un niveles de indirección: uno para acceder al valor de la pila y otro adicional para acceder al valor del parámetro.

Parámetros vía PILA por referencia

Ejemplo de parámetro por Pila y Referencia

```
MOV AX, OFFSET MiVar
PUSH AX          ; solo 1 parámetro
CALL Subrutina
POP AX
…
ORG 3000H
Subrutina:       MOV BX, SP  ; recupera dirección de pila (3FFCH)
                 ADD BX, 2   ; para acceder al parámetro (3FFEH)
                 MOV BX, [BX]; recupera dir de MiVar
                 ADD AX, [BX];  utiliza valor de MiVar
                 …
                 RET
```

3FFCH	dir. Retorno
3FFEH	Dir. de MiVar
4000H	--

Parámetros – Ejemplo combinado

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.

```
1.  ORG 1000H; Mem. de datos
2.  NUM1 DW 5H          ; comienza en 1000H y termina en 1001H
3.  NUM2 DW 3H          ; comienza en 1002H y termina en 1003H
4.  RES DW ?            ; comienza en 1004H y termina en 1005H

...
27. ORG 2000H ; Programa
28. MOV AX, NUM1        ; asigna valor de NUM1 (5H) a AX
29. PUSH AX             ; apila en dir. 3FFEH
30. MOV AX, NUM2        ; asigna valor de NUM2 (3H) a AX
31. PUSH AX             ; apila en dir. 3FFCH
32. MOV AX, OFFSET RES  ; recupera dirección de RES (1004H)
33. PUSH AX             ; apila en dir. 3FFAH
34. MOV DX, 0           ; resultado inicial de multiplicación en DX
35. CALL MUL            ; llama rutina, apila dir. retorno en 3FF8H
36. POP AX              ; desapila la misma cantidad que apilo
37. POP AX
38. POP AX
39. HLT
40. END
```

Programa

3FFAH	Dir RES (1004H)
3FFCH	NUM2 (0003H)
3FFEH	NUM1 (0005H)
4000H	--

Subrutina

```
5.          ORG 3000H ; Subrutina
6.   MUL:  PUSH BX      ; apila BX en 3FF6H, preserva valor anterior
7.         MOV BX, SP   ; asigna SP (3FF6H) en BX (BX apunta a pila)
8.         PUSH CX      ; apila CX, AX, DX para preservar valores de quien llama a la
9.         PUSH AX      subrutina. Es una buena practica de programación. SP queda en
10.        PUSH DX      3FF0H
11.        ADD BX, 6    ; desplaza 3 lugares en pila BX=3FFCH (NUM2)
12.        MOV CX, [BX] ; CX queda con el valor de NUM2
13.        ADD BX, 2    ; desplaza 1 lugar en pila BX=3FFEH (NUM1)
14.        MOV AX, [BX] ; AX queda con el valor de NUM1
15.  SUMA:  ADD DX, AX   ; acumula valor de NUM1 en el resultado
16.        DEC CX       ; decrementa veces restantes a sumar (NUM2)
17.        JNZ SUMA     ; salta si CX (valor de NUM2) no llego a 0
18.        SUB BX, 4    ; desplaza 2 lugares en pila BX=3FFA (dir. RES)
19.        MOV AX, [BX] ; asigna dir. de RES: [BX]⇔[3FFAH]⇔1004H
20.        MOV BX, AX   ; asigna dir. de RES a BX (1004H)
21.        MOV [BX], DX ; guarda resultado. [BX]⇔[1004H]⇔DX⇔15
22.        POP DX
23.        POP AX
24.        POP CX
25.        POP BX
26.        RET
```

Estado de pila luego de línea 10

3FF0H	Valor de DX
3FF2H	Valor de AX
3FF4H	Valor de CX
3FF6H	Valor de BX
3FF8H	Dir Retorno (L36)
3FFAH	Dir RES (1004H)
3FFCH	NUM2 (0003H)
3FFEH	NUM1 (0005H)
4000H	--

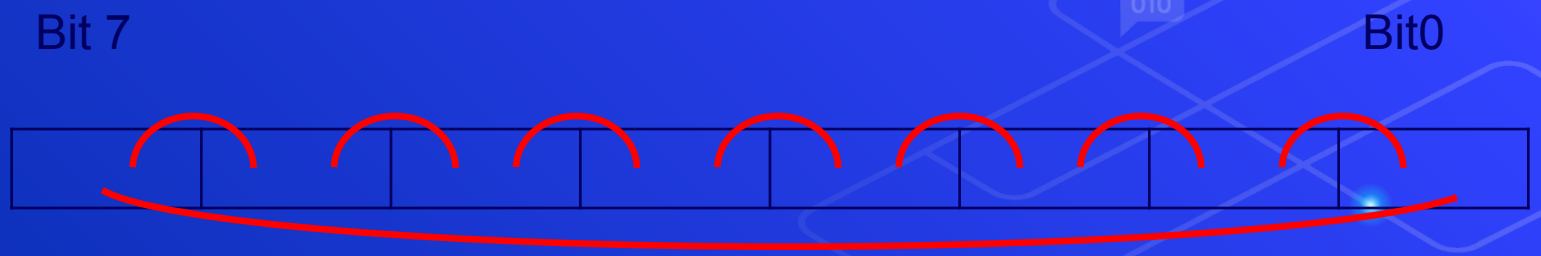
Estado de pila al ingresar a la subrutina

Ejercicios de rotación

Ejercicios de Rotación

Rotar a izquierda un byte:

Rotar a la izquierda significa que todos los bits se desplazan a izquierda y que el más significativo se copie en el bit menos significativo:



Ejemplos:

10001000 → 00010001

00011100 → 00111000

Ejercicios de Rotación

Rotar a izquierda un byte:

El 8088 no tiene instrucciones para rotar y ni desplazar

¿Cómo hacemos? Analicemos que significa desplazar a izquierda varias veces en binario y decimal:

00000011 □ 00000110 □ 00001100 □ 00011000 □ 00110000

3 □ 6 □ 12 □ 24 □ 48

En cada desplazamiento el número se duplica, es decir multiplica por 2 (pero el 8088 no tiene instrucciones de multiplicación):

2*A puede escribirse como una suma: A + A

Se utiliza la instrucción ADD. Ej: ADD AL, AL

Ejercicios de Rotación

Rotar a izquierda un byte:

Es importante no perder el bit más significativo

Al desplazar a izquierda el bit mas significativo queda en el acarreo

¿Como se ubica el bit más significativo en el menos significativo? Hay 2 maneras:

Opción 1

Puede usarse la ADC con 0: sumar al valor desplazado el valor cero y el del carry.

Opción 2

Puede utilizarse instrucción de salto verificando C=1 y ajustarlo solo en este caso. El ajuste se puede hacer con una instrucción OR con 1 o ADD con 1

Ejercicios de Rotación

Rotar a derecha un byte:

Para rotar a derecha se puede utilizar la rotación a izquierda.

Ejemplo para 4 bits:

Rotar a izquierda: 0001□0010□0100□1000

Rotar a derecha : 0001□1000□0100□0010

Dados **n** bits (4, 8, 16, 32, etc.) de un valor a rotar:

rotar 1 lugar a derecha es equivalente a rotar n-1 lugares a izquierda

Ejercicios con cadenas de caracteres

Ejercicios de Cadenas

Se pueden declarar strings o cadenas de caracteres. Recuerda que los caracteres en verdad se almacenan como códigos; los mismos se obtienen del estándar ASCII.

> Por ejemplo, la letra *A* se codifica con el número **41h**, y la letra *a* con el número **61h**.

Entonces, como Assembler es un lenguaje de bajo nivel, en realidad lo que declararemos es un vector de números, donde cada número es el código ASCII de un carácter.

El compilador de assembler nos permite ingresar un texto entre comillas y el convierte el mismo en códigos.

Ejercicios de Cadenas

Escribir subrutina que cuente los caracteres de una cadena terminada en "%"

Terminada en "%" significa que el último carácter legible es "%". Significa **código ASCII del signo %,** que es 21H

Cuando hablamos de **cadena**, el **pasaje de parámetros** se realiza **por referencia**, pasando el puntero al primer carácter.

El recorrido implica avanzar hasta llegar final de la cadena.

Ejercicios de Cadenas

La declaración:

NUM DB “3210%” es una secuencia de 5 bytes con los valores hexadecimales 33h, 32h, 31h, 30h, 21h

Recordar que los caracteres número son:

Para el número 0 en ASCII es 30h, el 1 es 31h, el 2 32h, etc.

Ejercicios de Cadenas

Escribir subrutina que determine si un carácter es vocal.

Hay dos alternativas de solución:

Solución 1 Por ejemplo en AL tengo el valor 45H

La simple, fácil, poco escalable y poco digna de un buen programador: comparar con instrucciones vocal por vocal con el carácter a probar:

```
CMP AL, 41h ; valor de “A”
JZ DEVOLVER_VERDADERO
CMP AL, 61h ; valor de “a”
JZ DEVOLVER_VERDADERO
CMP AL, 45h ; valor de “E”
JZ DEVOLVER_VERDADERO
…
```

Ejercicios de Cadena

Solución 2 Por ejemplo en AL tengo el valor 45H

La no tan simple, no tan fácil, escalable y digna de un buen programador: usar un “arreglo” de vocales para comparar con el carácter a probar

```
        VOCALES DB “AaEeIiOoUu”
        MOV BX, OFFSET VOCALES
        MOV CL, 10 ; cantidad en el arreglo
VOLVER: CMP AL, [BX]
        JZ DEVOLVER_VERDADERO
        INC BX ; apunta a próxima vocal
        DEC CL; actualiza cantidad vocales restantes
        JNZ VOLVER ; quedan vocales para testear?
        …
```