

## OPERACIONES CON CADENAS DE CARACTERES

### INTRODUCCION:

- Sirven para mover o comparar campos de datos que excedan una palabra doble (4 bytes).
- Los elementos de datos se conocen como “datos de cadena”, que pueden ser de carácter o numérico.

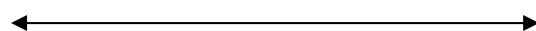
<b>MOVS</b>	<b>Mover un byte, palabra o palabra doble de una localidad de memoria a otra</b>
<b>LODS</b>	<b>Carga desde memoria un byte en el AL, palabra en el AX, o palabra doble en el EAX</b>
<b>STOS</b>	<b>Almacena el contenido de AL, AX o EAX en memoria</b>
<b>CMPS</b>	<b>Compara localidades de memoria de un byte, palabra o palabra doble</b>
<b>SCAS</b>	<b>Compara contenido de AL, AX, EAX con el contenido de una localidad de memoria</b>

### CARACTERÍSTICAS:

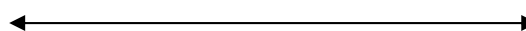
- Cada instrucción de cadena tiene una versión para byte, palabra o palabra doble.
- Asume que se utilizan los registros ES:DI o DS:SI. Por tanto estos registros deben contener direcciones de desplazamiento validas.

Codificación de las instrucciones:

Operación	Instrucción básica	Operandos	Operación con bytes	Operación con word	Operación con dword
Mover	MOVS	ES:DI, DS:SI	MOVSB	MOVSW	MOVSD
Cargar	LODS	AX, DS:SI	LODSB	LODSW	LODSD
Almacenar	STOS	ES:DI, AX	STOSB	STOSW	STOSD
Comparar	CMPS	DS:SI, ES:DI	CMPSB	CMPSW	CMPSD
Rastrear	SCAS	ES:DI, AX	SCASB	SCASW	SCASD



Primera forma: utiliza los operandos implicados  
Ej: MOVS byte1, byte2  
La definición de los operandos indica la longitud del movimiento



Segunda forma: se cargan las direcciones de los operandos  
En los registros DI, SI y se escribe la operación sin operandos

REP MOVSB byte1, byte2

### REP: PREFIJO DE REPETICIÓN DE CADENA

- Se utiliza antes de una instrucción de cadena
- Proporciona una ejecución repetida con base en un contador inicial que se inicia en el registro CX
- REP ejecuta la instrucción de cadena, disminuye el CX y repite la operación hasta que el CX sea cero
- La bandera de dirección determina la dirección de la operación que se repite:
  - o Para procesamiento de izq a der, utilice CLD (pone a cero la bandera de dirección) – modo normal
  - o Para procesamiento de a der a izq, utilice STD (pone a uno la bandera de dirección)

Ejemplo: Copia cadena1 a cadena2

```
CADENA1 DB 20 DUP('*')
CADENA2 DB 20 DUP(' ')

CLD                ; Bandera de dirección a cero
MOV CX,20          ; contador para 20 bytes
LEA DI, CADENA2    ; cadena destino
LEA SI, CADENA1    ; cadena fuente (source)
REP MOVSB          ; copia cadena1 a cadena2
```

Variaciones de REP:

- 
- REPE o REPZ: repite la operación mientras la bandera de cero sea Cero. Se detiene cuando ZF indica diferente de cero o CX es cero
- REPNE o REPNZ: repite la operación mientras la bandera de cero sea diferente de Cero. Se detiene cuando ZF indica igual de cero o CX es cero

### MOVS: MOVER UNA CADENA DE CARACTERES

[etiqueta:] REP MOVSn [ES:DI, DS:SI]

- ES:DI cadena destino
- ES:SI cadena fuente (source)
- **ES se inicializa con DS ¡OJO! Super importante. Sino lo hace, no pregunte porque no funciona su programa...**

Instrucciones equivalentes para REP MOVSB:

```
CICLO:    MOV CX, numbytes
          JCXZ SALIDA
          MOV AL, [SI]
          MOV [DI], AL
          INC DI
          INC SI
          LOOP CICLO

SALIDA:    .....
```

### **LODS: CARGA UNA CADENA DE CARACTERES**

- Carga el AL con 1 byte, AX con una palabra o EAX con una palabra doble desde la memoria
- La dirección de memoria esta indicada por DS:SI
- Dependiendo de la bandera de dirección, se incrementa o disminuye el SI en 1, 2 o 4 bytes
- No tiene sentido usar el REP
- Funciona igual que MOV, pero ahorra 2 bytes de código de maquina.

Ejemplo: Inversión de cadenas. En CAMPOB se copia ODALBMASNE

```
CAMPOA    DB    'ENSAMBLADO'
CAMPOB    DB    10 DUP(20H)
...
          CLD
          MOV CX, 10
          LEA SI, CAMPOA          ; carga dirección de campoa
          LEA DI, CAMPOB+9        ; carga dir de campob+9
CICLO:    LODSB                  ; obtiene el carácter en AL
          MOV [DI], AL
          DEC DI
          LOOP CICLO
```

Instrucciones equivalentes a LODSB:

```
MOV AL,[SI]
INC SI
```

## STOS: ALMACENAR UNA CADENA DE CARACTERES

- Almacena los contenidos de AL, AX o EAX en un byte, palabra o palabra doble en memoria
- La dirección en memoria esta indicada por ES:DI
- Dependiendo de la bandera de dirección, se incrementa o disminuye el SI en 1, 2 o 4 bytes
- Se utiliza con REP

```
NOMBRE    DB    '0123456789'
...
CLD                      ; izq a der
MOV  AX,2020H            ; mover 5 blancos a NOMBRE
MOV  CX,5
LEA  DI, NOMBRE
REP  STOSW
...
```

## CMPS: COMPARAR CADENAS

- Compara el contenido de una localidad de memoria, direccionada por DS:SI con el de otra localidad de memoria, direccionada por ES:DI
- Incrementa o disminuye SI y DI en 1, 2 o 4 bytes
- Al finalizar la ejecución, establece las banderas AF, CF, OF, PF, SF, ZF
- Puede combinarse con REP para comparar cadenas de cualquier longitud
- La comparación es alfanumérica, de acuerdo a los valores ASCII
- La comparación debe terminar cuando se encuentra un carácter diferente (REPE CMPSB)

Ejemplo: Compara NOM1 con NOM2 y NOM2 con NOM3

```
...  
NOM1      'ENSAMBLADO' ; elementos de datos  
NOM2      'ENSAMBLADO'  
NOM3      10 DUP ( ' )  
...  
          CLD           ; izq a der  
          MOV  CX,10     ; 10 bytes  
          LEA  DI,NOM1  
          LEA  SI,NOM2  
          REPE CMPSB     ; compare NOM1 y NOM2  
          JNE  CICLO     ; no es igual, terminar comparación  
          MOV  BH,1      ; si es igual, código indicador en BH  
CICLO:    MOV  CX,10  
          LEA  DI,NOM3  
          LEA  SI,NOM2  
          REPE CMPSB     ; compare NOM2 y NOM3  
          JE   SALIR     ; es igual, salir  
          MOV  BL,2      ; no es igual, código indicador en BL  
SALIR:    ...
```

- Nota importante: cuando se compara con palabras (CMPSW), se invierten los bytes

### SCAS: BÚSQUEDA EN CADENAS

- Busca una cadena por un valor de 1, 2 o 4 bytes
- Compara el contenido de la localidad de memoria, direccionada por ES:DI, con el contenido del registro AL, AX, EAX
- Incrementa o disminuye DI en 1, 2 o 4 bytes
- Al finalizar la ejecución, establece las banderas AF, CF, OF, PF, SF, ZF
- Puede combinarse con REP para buscar cadenas de cualquier longitud
- Es útil para la edición de texto, cuando necesita buscarse signos como coma(,), punto(.), blancos o algún carácter en particular

Ejemplo: Buscar '9' en la variable DIGITO

***	DIGITO	DB	'0123456789'	
		CLD		; izq a der
		MOV	AL, '9'	; busca '9' en DIGITO
		MOV	CX, 10	
		LEA	DI, DIGITO	
		REPNE	SCASB	; repite mientras no
		JNE	SALIR	; sea igual o CX es 0
		MOV	AL, 03	; código indicador
	SALIR:			

### BUSCAR Y REEMPLAZAR

- Utilizando SCASB

Ejemplo: Buscar el carácter '&' y reemplazarlo por un espacio en blanco

LONGITUD	EQU	13	
CADENA	DB	'Var1=1&Var2=2'	
..			
	CLD		; izq a der
	MOV	AL, '&'	; busca el &
	MOV	CX, LONGITUD	;
	LEA	DI, CADENA	
	REPNE	SCASB	; repite mientras no se
			; igual o CX es cero
	JNZ	SALIR	; se encontró carácter
	DEC	DI	; si, ajusta dirección
	MOV	BYTE PTR[DI], 20H	; reemplace con un blanco
	SALIR:	...	

### OBSERVACIONES:

- Inicializar el registro ES con @data
- Utilizar los sufijos B, W o D para manejo de byte, palabra o palabra doble
- Definir la dirección de procesamiento con CLD o STD
- Verificar la inicialización de SI y DI, de acuerdo a la instrucción. MOVS trabaja con DI, SI mientras CMPS trabaja con SI, DI
- Inicializar CX para usar REP



- 
- CMPSW y SCASW invierte los bytes
  - Para procesar cadenas de derecha a izquierda, inicializar registros SI, DI con el nombre de la variable + longitud