

En este diagrama podemos ver que las banderas Desbordamiento, Signo, Cero y Paridad cambiarán a valores desconocidos. Las banderas Acarreo auxiliar y Acarreo se modificarán de acuerdo a las reglas asociadas con las banderas. Las banderas Dirección e Interrupción no se cambiarán.

B.1.2 Descripciones y formatos de las instrucciones

Al hacer una referencia a los operandos de origen y de destino, utilizamos el orden natural de los operandos en todas las instrucciones del Intel 80x86, en donde el primer operando es el destino y el segundo es el origen. Por ejemplo, en la instrucción MOV al destino se le asignará una copia de los datos en el operando de origen:

MOV destino, origen

Puede haber muchos formatos disponibles para una sola instrucción. La tabla B-1 contiene una lista de símbolos que se utilizan en los formatos de las instrucciones. En las descripciones de cada instrucción, utilizamos la notación "(IA-32)" para indicar que una instrucción, o una de sus variantes, sólo está disponible en los procesadores de la familia IA-32 (del Intel386 en adelante). De manera similar, la notación "(80286)" indica que debe usarse por lo menos un procesador 80286.

Las notaciones de los registros como (E)CX, (E)SI, (E)DI, (E)SP, (E)BP, y (E)IP hacen la diferencia entre los procesadores IA-32 que utilizan los registros de 32 bits y los primeros procesadores que utilizaban registros de 16 bits.

Tabla B-1 Símbolos que se utilizan en los formatos de las instrucciones.

Símbolo	Descripción
<i>reg</i>	Un registro general de 8, 16 o 32 bits de la siguiente lista: AH, AL, BH, BL, CH, CL, DH, DL, AX, BX, CX, DX, SI, DI, BP, SP, EAX, EBX, ECX, EDX, ESI, EDI, EBP y ESP
<i>reg8, reg16, reg32</i>	Un registro general, identificado por su número de bits
<i>regseg</i>	Un registro de segmento de 16 bits (CS, DS, ES, SS, FS, GS)
<i>acum</i>	AL, AX o EAX
<i>mem</i>	Un operando de memoria, usando cualquiera de los modos de direccionamiento de memoria estándar
<i>mem8, mem16, mem32</i>	Un operando de memoria, identificado por su número de bits
<i>etiquetacorta</i>	Una ubicación en el segmento de código dentro de un rango de -127 a +128 bytes de la ubicación actual
<i>etiquetacercana</i>	Una ubicación en el segmento de código actual, identificado por una etiqueta
<i>etiquetalejana</i>	Una ubicación en un segmento de código externo, identificado por una etiqueta
<i>Inm</i>	Un operando inmediato
<i>inm8, inm16, inm32</i>	Un operando inmediato, identificado por su número de bits
<i>instrucción</i>	Una instrucción en lenguaje ensamblador del 80x86

B.2 Detalles del conjunto de instrucciones (que no son de punto flotante)

AAA**Ajuste ASCII después de la suma**

O	D	I	S	Z	A	P	C
?			?	?	*	?	*

Ajusta el resultado en AL después de sumar dos dígitos ASCII. Si AL > 9, el dígito superior del resultado se coloca en AH y se activan las banderas Acarreo y Acarreo auxiliar.
Formato de la instrucción:

AAA

AAD**Ajuste ASCII antes de la división**

O	D	I	S	Z	A	P	C
?			*	*	?	*	?

Convierte los dígitos BCD desempaquetados en AH y AL a un solo valor binario, como preparación para la instrucción DIV.
Formato de la instrucción:

AAD

AAM**Ajuste ASCII después de la multiplicación**

O	D	I	S	Z	A	P	C
?			*	*	?	*	?

Ajusta el resultado en AX de dos dígitos BCD no empaquetados después de haberlos multiplicado.
Formato de la instrucción:

AAM

AAS**Ajuste ASCII después de la resta**

O	D	I	S	Z	A	P	C
?			?	?	*	?	*

Ajusta el resultado en AX después de una operación de resta. Si AL > 9, AAS decremente a AH y activa las banderas Acarreo y Acarreo auxiliar.
Formato de la instrucción:

AAS

ADC**Suma con acarreo**

O	D	I	S	Z	A	P	C
*			*	*	*	*	*

Suma el operando de origen y la bandera Acarreo con el operando de destino. Los operandos deben tener el mismo tamaño.

Formatos de la instrucción:

ADC reg, reg
ADC mem, reg
ADC reg, mem

ADC reg, inm
ADC mem, inm
ADC acum, inm

ADD**Suma**

O	D	I	S	Z	A	P	C
*			*	*	*	*	*

Un operando de origen se suma a un operando de destino, y la suma se almacena en el destino. Los operandos deben tener el mismo tamaño.

Formatos de la instrucción:

ADD *reg, reg*
ADD *mem, reg*
ADD *reg, mem*

ADD *reg, imm*
ADD *mem, imm*
ADD *acum, imm*

AND**AND lógico**

O	D	I	S	Z	A	P	C
0			*	*	?	*	0

A cada bit en el operando de destino se le aplica un AND con el bit correspondiente en el operando de origen.

Formatos de la instrucción:

AND *reg, reg*
AND *mem, reg*
AND *reg, mem*

AND *reg, imm*
AND *mem, imm*
AND *acum, imm*

BOUND**Comprobar límites de arreglo (80286)**

O	D	I	S	Z	A	P	C

Verifica que el valor de un índice con signo se encuentre dentro de los límites del arreglo. En el procesador 80286, el operando de destino puede ser cualquier registro de 16 bits que contenga el índice a comprobar. El operando de origen debe ser un operando de memoria de 32 bits, en el que las palabras superior e inferior contienen los límites superior e inferior del valor del índice. En la familia IA-32, el destino puede ser un registro de 32 bits y el origen puede ser un operando de memoria de 64 bits.

Formatos de la instrucción:

BOUND *reg16, mem32*

BOUND *reg32, mem64*

**BSF,
BSR****Exploración de bit (IA-32)**

O	D	I	S	Z	A	P	C
?			?	?	?	?	?

Explora un operando en busca del primer bit activado. Si se encuentra el bit, se borra la bandera Cero, y al operando de destino se le asigna el número de bit (índice) del primer bit activo que se haya encontrado. Si no se encontró un bit, ZF = 1. BSF explora desde el bit 0 hasta el bit más alto, y BSR empieza en el bit más alto y explora en orden descendente, hasta el bit 0.

Formatos de la instrucción (se aplican a BSF y BSR):

BSF *reg16/r/m16*

BOUND *reg32,r/m32*

B.2 DETALLES DEL CONJUNTO DE INSTRUCCIONES (QUE NO SON DE PUNTO FLOTANTE)

623

BSWAP

Intercambiar byte (IA-32)

O	D	I	S	Z	A	P	C

Invierte el orden de los bytes de un registro de destino de 32 bits.
Formato de la instrucción:

BSWAP reg32

**BT,
BTC,
BTR,
BTS**

Pruebas de bit (IA-32)

O	D	I	S	Z	A	P	C
?			?	?	?	?	*

Copia un bit especificado (*n*) en la bandera Acarreo. El operando de destino contiene el valor en el que se encuentra el bit, y el operando de origen indica la posición del bit dentro del destino. B copia el bit *n* a la bandera Acarreo. BTC copia el bit *n* a la bandera Acarreo y complementa el bit *n* en el operando de destino. BTR copia el bit *n* en la bandera Acarreo y borra el bit *n* en el destino. BTS copia el bit *n* en la bandera Acarreo y activa el bit *n* en el destino.

Formatos de la instrucción:

BT r/m16, inm8
BT r/m32, inm8

BT r/m16, r16
BT r/m32, r32

CALL

Llamar a un procedimiento

O	D	I	S	Z	A	P	C

Mete la ubicación de la siguiente instrucción en la pila y se transfiere a la ubicación de destino. Si el procedimiento es cercano (en el mismo segmento), sólo se mete el desplazamiento de la siguiente instrucción; en caso contrario, se meten el segmento y el desplazamiento.

Formatos de la instrucción:

CALL etiquetacercana
CALL etiquetalejana
CALL reg

CALL mem16
CALL mem32

CBW

Convertir byte a palabra

O	D	I	S	Z	A	P	C

Extiende el bit de signo en AL a lo largo del registro AH.
Formato de la instrucción:

CBW

CDQ	Convertir doble palabra a palabra cuádruple (IA-32)
	O D I S Z A P C <input type="checkbox"/> <input type="checkbox"/>
	Extiende el bit de signo en EAX a lo largo del registro EDX. Formato de la instrucción: CDQ
CLC	Borrar bandera Acarreo
	O D I S Z A P C <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> 0
	Borra la bandera Acarreo, asignándole un cero. Formato de la instrucción: CLC
CLD	Borrar bandera Dirección
	O D I S Z A P C <input type="checkbox"/> 0 <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/>
	Borra la bandera Dirección, asignándole un cero. Las instrucciones de primitivas de cadena incrementan (E)SI y (E)DI de manera automática. Formato de la instrucción: CLD
CLI	Borrar bandera Interrupción
	O D I S Z A P C <input type="checkbox"/> <input type="checkbox"/> 0 <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/>
	Borra la bandera Interrupción, asignándole un cero. Esto deshabilita las interrupciones de hardware enmascarables hasta que se ejecute una instrucción STI. Formato de la instrucción: CLI
CMC	Complementar la bandera Acarreo
	O D I S Z A P C <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> *
	Cambia el valor actual la bandera Acarreo, de 0 a 1 y viceversa. Formato de la instrucción: CMC

B.2 DETALLES DEL CONJUNTO DE INSTRUCCIONES (QUE NO SON DE PUNTO FLOTANTE)

625

CMP

Comparar

O	D	I	S	Z	A	P	C
*			*	*	*	*	*

Compara el destino con el origen, realizando una resta implícita entre el origen y el destino.
Formatos de la instrucción:

CMP *reg, reg*
CMP *mem, reg*
CMP *reg, mem*

CMP *reg, imm*
CMP *mem, imm*
CMP *acum, imm*

**CMPS,
CMPSB,
CMPSW,
CMPSD**

Comparar cadenas

O	D	I	S	Z	A	P	C
*			*	*	*	*	*

Compara las cadenas en la memoria direccionada por DS:(E)SI y ES:(E)DI. Realiza una resta implícita entre el destino y el origen. CMPSB compara bytes, CMPSW compara palabras y CMPSD compara dobles palabras (en procesadores IA-32). (E)SI y (E)DI se incrementan o decrementan de acuerdo con el tamaño del operando y el estado de la bandera Dirección. Si la bandera Dirección está activa, (E)SI y (E)DI se decrementan; en caso contrario (E)SI y (E)DI se incrementan.
Formatos de la instrucción (se omitieron de manera intencional los formatos que utilizan operandos explícitos):

CMPSB
CMPSD

CMPSW

CMPXCHG

Comparar e intercambiar

O	D	I	S	Z	A	P	C
*			*	*	*	*	*

Compara el destino con el acumulador (AL, AX o EAX). Si son iguales, el origen se copia al destino; en caso contrario, el destino se copia al acumulador.
Formatos de la instrucción:

CMPXCHG *reg, reg*

CMPXCHG *mem, reg*

CWD

Convierte palabra a doble palabra

O	D	I	S	Z	A	P	C

Extiende el bit de signo en AX hasta el registro DX.
Formato de la instrucción:

CWD

DAA	<p>Ajuste decimal después de la suma</p> <table border="1" style="margin-left: auto; margin-right: auto;"> <tr> <th>O</th><th>D</th><th>I</th><th>S</th><th>Z</th><th>A</th><th>P</th><th>C</th></tr> <tr> <td>?</td><td></td><td></td><td>*</td><td>*</td><td>*</td><td>*</td><td>*</td></tr> </table> <p>Ajusta la suma binaria en AL después de haber sumado dos valores BCD empaquetados. Convierte la suma a dos dígitos BCD en AL.</p> <p>Formato de la instrucción: DAA</p>	O	D	I	S	Z	A	P	C	?			*	*	*	*	*
O	D	I	S	Z	A	P	C										
?			*	*	*	*	*										
DAS	<p>Ajuste decimal después de la resta</p> <table border="1" style="margin-left: auto; margin-right: auto;"> <tr> <th>O</th><th>D</th><th>I</th><th>S</th><th>Z</th><th>A</th><th>P</th><th>C</th></tr> <tr> <td>?</td><td></td><td></td><td>*</td><td>*</td><td>*</td><td>*</td><td>*</td></tr> </table> <p>Convierte el resultado binario de una operación de resta a dos dígitos BCD empaquetados en AL.</p> <p>Formato de la instrucción: DAS</p>	O	D	I	S	Z	A	P	C	?			*	*	*	*	*
O	D	I	S	Z	A	P	C										
?			*	*	*	*	*										
DEC	<p>Decrementar</p> <table border="1" style="margin-left: auto; margin-right: auto;"> <tr> <th>O</th><th>D</th><th>I</th><th>S</th><th>Z</th><th>A</th><th>P</th><th>C</th></tr> <tr> <td>*</td><td></td><td></td><td>*</td><td>*</td><td>*</td><td>*</td><td></td></tr> </table> <p>Resta 1 de un operando. No afecta a la bandera Acarreo.</p> <p>Formatos de la instrucción: DEC reg DEC mem</p>	O	D	I	S	Z	A	P	C	*			*	*	*	*	
O	D	I	S	Z	A	P	C										
*			*	*	*	*											
DIV	<p>Dividir entero sin signo</p> <table border="1" style="margin-left: auto; margin-right: auto;"> <tr> <th>O</th><th>D</th><th>I</th><th>S</th><th>Z</th><th>A</th><th>P</th><th>C</th></tr> <tr> <td>?</td><td></td><td></td><td>?</td><td>?</td><td>?</td><td>?</td><td>?</td></tr> </table> <p>Realiza una división de enteros sin signo de 8, 16 o 32 bits. Si el divisor es de 8 bits, el dividendo es AX, el cociente es AL y el residuo es AH. Si el divisor es de 16 bits, el dividendo es DX:AX, el cociente es AX y el residuo es DX. Si el divisor es de 32 bits, el dividendo es EDX:EAX, el cociente es EAX y el residuo es EDX.</p> <p>Formatos de la instrucción: DIV reg DIV mem</p>	O	D	I	S	Z	A	P	C	?			?	?	?	?	?
O	D	I	S	Z	A	P	C										
?			?	?	?	?	?										
ENTER	<p>Crear marco de pila (80286)</p> <table border="1" style="margin-left: auto; margin-right: auto;"> <tr> <th>O</th><th>D</th><th>I</th><th>S</th><th>Z</th><th>A</th><th>P</th><th>C</th></tr> <tr> <td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> </table> <p>Crea un marco de pila para un procedimiento que recibe parámetros de pila y utiliza variables de pila locales. El primer operando indica el número de bytes a reservar para las variables de pila locales. El segundo operando indica el nivel de anidamiento del procedimiento (debe establecerse en 0 para C, Basic y FORTRAN).</p> <p>Formato de la instrucción: ENTER inm16, inm8</p>	O	D	I	S	Z	A	P	C								
O	D	I	S	Z	A	P	C										

8.2 Detalles del CONJUNTO DE INSTRUCCIONES (QUE NO SON DE PUNTO FLOTANTE)

HLT

Detener

627

O	D	I	S	Z	A	P	C

Detiene la CPU hasta que ocurra una interrupción de hardware. (Nota: la bandera Interrupción debe activarse con la instrucción STI para que puedan ocurrir interrupciones de hardware.)
Formato de la instrucción:

HLT

IDIV

Dividir entero con signo

O	D	I	S	Z	A	P	C
?			?	?	?	?	?

Realiza una operación de división de enteros con signo en EDX:EAX, DX:AX o AX. Si el divisor es de 8 bits, el dividendo es AX, el cociente es AL y el residuo es AH. Si el divisor es de 16 bits, el dividendo es DX:AX, el cociente es AX y el residuo es DX. Si el divisor es de 32 bits, el dividendo es EDX:EAX, el cociente es EAX y el residuo es EDX. Por lo general, se utiliza CBW o CWD antes de la operación IDIV para extender el signo del dividendo.

Formatos de la instrucción:

IDIV reg

IDIV mem

IMUL

Multiplicar entero con signo

O	D	I	S	Z	A	P	C
*			?	?	?	?	*

Realiza una multiplicación de enteros con signo en AL, AX o EAX. Si el multiplicador es de 8 bits, el multiplicando es AL y el producto es AX. Si el multiplicador es de 16 bits, el multiplicando es AX y el producto es DX:AX. Si el multiplicador es de 32 bits, el multiplicando es EAX y el producto es EDX: EAX. Las banderas Acarreo y Desbordamiento se activan si un producto de 16 bits se extiende hacia AH, si un producto de 32 bits se extiende hacia DX, o si un producto de 64 bits se extiende hacia EDX.

Formatos de la instrucción:

IMUL r/m8
IMUL r/,32

IMUL r/m16

Dos operandos:

IMUL r16/m16
IMUL r32/m32
IMUL r16/inm16

IMUL r16,inm8
IMUL r32,inm8
IMUL r32,inm32

Tres operandos:

IMUL r16,r/m16,inm8
IMUL r32,r/m32,inm8

IMUL r16,r/m16,inm16
IMUL r32,r/m32,inm32

IN	<p>Entrada desde un puerto</p> <table style="margin-left: auto; margin-right: auto;"> <tr> <td>O</td> <td>D</td> <td>I</td> <td>S</td> <td>Z</td> <td>A</td> <td>P</td> <td>C</td> </tr> <tr> <td>[]</td> </tr> </table> <p>Recibe un byte o palabra de un puerto y lo coloca en AL o AX. El operando de origen es la dirección de un puerto, que se expresa como una constante de 8 bits o como una dirección de 16 bits en DX. En el procesador IA-32, una doble palabra puede recibirse desde un puerto y colocarse en EAX.</p> <p>Formatos de la instrucción:</p> <table style="margin-left: auto; margin-right: auto;"> <tr> <td>IN <i>acum, inm</i></td> <td>IN <i>acum,DX</i></td> </tr> </table>	O	D	I	S	Z	A	P	C	[]	[]	[]	[]	[]	[]	[]	[]	IN <i>acum, inm</i>	IN <i>acum,DX</i>		
O	D	I	S	Z	A	P	C														
[]	[]	[]	[]	[]	[]	[]	[]														
IN <i>acum, inm</i>	IN <i>acum,DX</i>																				
INC	<p>Incrementar</p> <table style="margin-left: auto; margin-right: auto;"> <tr> <td>O</td> <td>D</td> <td>I</td> <td>S</td> <td>Z</td> <td>A</td> <td>P</td> <td>C</td> </tr> <tr> <td>*</td> <td>[]</td> <td>*</td> <td>*</td> <td>*</td> <td>*</td> <td>*</td> <td>[]</td> </tr> </table> <p>Suma 1 a un operando tipo registro o de memoria.</p> <p>Formatos de la instrucción:</p> <table style="margin-left: auto; margin-right: auto;"> <tr> <td>INC <i>reg</i></td> <td>INC <i>mem</i></td> </tr> </table>	O	D	I	S	Z	A	P	C	*	[]	*	*	*	*	*	[]	INC <i>reg</i>	INC <i>mem</i>		
O	D	I	S	Z	A	P	C														
*	[]	*	*	*	*	*	[]														
INC <i>reg</i>	INC <i>mem</i>																				
INS, INSB, INSW, INSD	<p>Entrada desde un puerto a una cadena (80286)</p> <table style="margin-left: auto; margin-right: auto;"> <tr> <td>O</td> <td>D</td> <td>I</td> <td>S</td> <td>Z</td> <td>A</td> <td>P</td> <td>C</td> </tr> <tr> <td>[]</td> </tr> </table> <p>Recibe una cadena a la que apunta ES:(E)DI desde un puerto. El número de puerto se especifica en DX. Para cada valor recibido, (E)DI se ajusta de la misma forma que en LODSB y las instrucciones de primitivas de cadena similares. Puede utilizarse el prefijo REP con esta instrucción.</p> <p>Formatos de la instrucción:</p> <table style="margin-left: auto; margin-right: auto;"> <tr> <td>INS <i>dest,DX</i></td> <td>REP INSB <i>dest,DX</i></td> </tr> <tr> <td>REP INSW <i>dest,DX</i></td> <td>REP INSD <i>dest,DX</i></td> </tr> </table>	O	D	I	S	Z	A	P	C	[]	[]	[]	[]	[]	[]	[]	[]	INS <i>dest,DX</i>	REP INSB <i>dest,DX</i>	REP INSW <i>dest,DX</i>	REP INSD <i>dest,DX</i>
O	D	I	S	Z	A	P	C														
[]	[]	[]	[]	[]	[]	[]	[]														
INS <i>dest,DX</i>	REP INSB <i>dest,DX</i>																				
REP INSW <i>dest,DX</i>	REP INSD <i>dest,DX</i>																				
INT	<p>Interrupción</p> <table style="margin-left: auto; margin-right: auto;"> <tr> <td>O</td> <td>D</td> <td>I</td> <td>S</td> <td>Z</td> <td>A</td> <td>P</td> <td>C</td> </tr> <tr> <td>[]</td> <td>[]</td> <td>0</td> <td>[]</td> <td>[]</td> <td>[]</td> <td>[]</td> <td>[]</td> </tr> </table> <p>Genera una interrupción de software, que a su vez llama a una subrutina del sistema operativo. Borra la bandera Interrupción y mete las banderas, CS e IP en la pila antes de bifurcar hacia la rutina de interrupción.</p> <p>Formatos de la instrucción:</p> <table style="margin-left: auto; margin-right: auto;"> <tr> <td>INT <i>inm</i></td> <td>INT 3</td> </tr> </table>	O	D	I	S	Z	A	P	C	[]	[]	0	[]	[]	[]	[]	[]	INT <i>inm</i>	INT 3		
O	D	I	S	Z	A	P	C														
[]	[]	0	[]	[]	[]	[]	[]														
INT <i>inm</i>	INT 3																				
INTO	<p>Interrupción por desbordamiento</p> <table style="margin-left: auto; margin-right: auto;"> <tr> <td>O</td> <td>D</td> <td>I</td> <td>S</td> <td>Z</td> <td>A</td> <td>P</td> <td>C</td> </tr> <tr> <td>[]</td> <td>[]</td> <td>*</td> <td>*</td> <td>[]</td> <td>[]</td> <td>[]</td> <td>[]</td> </tr> </table> <p>Genera la interrupción interna 4 de la CPU si se activa la bandera Desbordamiento. MS-DOS no realiza ninguna acción si se llama a INT 4, pero puede sustituirse por una rutina escrita por el usuario.</p> <p>Formato de la instrucción:</p> <table style="margin-left: auto; margin-right: auto;"> <tr> <td>INTO</td> </tr> </table>	O	D	I	S	Z	A	P	C	[]	[]	*	*	[]	[]	[]	[]	INTO			
O	D	I	S	Z	A	P	C														
[]	[]	*	*	[]	[]	[]	[]														
INTO																					

IRET**Retorno de interrupción**

O	D	I	S	Z	A	P	C
*	*	*	*	*	*	*	*

Regresa de una rutina de manejo de interrupciones. Saca el contenido de la pila y lo coloca en (E)IP, CS y las banderas.
 Formato de la instrucción:

IRET

Jcondición**Salto condicional**

O	D	I	S	Z	A	P	C

Salta a una etiqueta si una condición de bandera especificada es verdadera. Si se usa un procesador anterior al IA-32, la etiqueta debe estar en el rango de -128 a +127 bytes, a partir de la ubicación actual. En los procesadores IA-32, el desplazamiento de la etiqueta puede ser un valor positivo o negativo de 32 bits. En la tabla B-2 podrá ver una lista de nemáticos.

Formato de la instrucción:

Jcondición etiqueta

Tabla B-2 Nemáticos de salto condicional.

Nemónico	Comentario	Nemónico	Comentario
JA	Salta si es mayor	JE	Salta si es igual
JNA	Salta si no es mayor	JNE	Salta si no es igual
JAE	Salta si es mayor o igual	JZ	Salta si es cero
JNAE	Salta si no es mayor o igual	JNZ	Salta si no es cero
JB	Salta si es menor	JS	Salta si hay signo
JNB	Salta si no es menor	JNS	Salta si no hay signo
JBE	Salta si es menor o igual	JC	Salta si hay acarreo
JNBE	Salta si no es menor o igual	JNC	Salta si no hay acarreo
JG	Salta si es mayor	JO	Salta si hay desbordamiento
JNG	Salta si no es mayor	JNO	Salta si no hay desbordamiento
JGE	Salta si es mayor o igual	JP	Salta si hay paridad
JNGE	Salta si no es mayor o igual	JPE	Salta si la paridad es igual
JL	Salta si es menor	JNP	Salta si no hay paridad
JNL	Salta si no es menor	JPO	Salta si la paridad es impar
JLE	Salta si es menor o igual	JNLE	Salta si no es menor o igual

JCXZ, JECXZ	Salta si CX es cero	O D I S Z A P C <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/>
	Salta a una etiqueta corta si el registro CX es igual a cero. La etiqueta corta debe estar en el rango de -128 a +127 bytes, a partir de la siguiente instrucción. En el procesador IA-3, JECXZ salta si ECX es igual a cero. Formatos de la instrucción:	JCXZ etiquetacorta JECXZ etiquetacorta
JMP	Salta incondicionalmente a la etiqueta	O D I S Z A P C <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/>
	Salta a una etiqueta de código. Un salto corto se encuentra dentro de los -128 a +127 bytes a partir de la ubicación actual. Un salto cercano está dentro del mismo segmento de código, y un salto lejano se encuentra fuera del segmento actual. Formatos de la instrucción:	JMP etiquetacorta JMP reg16 JMP etiquetacercana JMP mem16 JMP etiquetalejana JMP mem32
LAHF	Cargar AH de las banderas	O D I S Z A P C <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/>
	Las siguientes banderas se copian a AH: Signo, Cero, Acarreo auxiliar, Paridad y Acarreo. Formato de la instrucción:	LAHF
LDS, LES, LFS, LGS, LSS	Cargar apuntador lejano	O D I S Z A P C <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/>
	Carga el contenido de un operando de memoria tipo doble palabra en un registro de segmento y en el registro de destino especificado. En los procesadores anteriores al IA-3, LDS carga en DS, LES carga en ES. En el procesador IA-32, LFS carga en FS, LGS carga en GS y LSS carga en SS. Formato de la instrucción (es igual para LDS, LES, LFS, LGS, LSS):	LDS reg,mem

B.2 DETALLES DEL CONJUNTO DE INSTRUCCIONES (QUE NO SON DE PUNTO FLOTANTE)

631

LEA

Cargar dirección efectiva

O	D	I	S	Z	A	P	C

Calcula y carga la dirección efectiva de 16 o 32 bits de un operando de memoria. Es similar a MOV, OFFSET, con la excepción de que sólo LEA puede obtener una dirección que se calcule en tiempo de ejecución.

Formato de la instrucción:

LEA reg,mem

LEAVE

Salir de procedimiento de alto nivel

O	D	I	S	Z	A	P	C

Termina el marco de pila de un procedimiento. Esto invierte la acción de la instrucción ENTER al inicio de un procedimiento, restaurando (E)SP y (E)BP a sus valores originales.

Formato de la instrucción:

LEAVE

LOCK

Bloquea el bus del sistema

O	D	I	S	Z	A	P	C

Evita que otros procesadores se ejecuten durante la siguiente instrucción. Esta instrucción se usa cuando otro procesador pueda modificar un operando de memoria accesado en ese momento por la CPU.

Formato de la instrucción:

LOCK instrucción

**LDS,
LODSB,
LODSW,
LOSD**

Carga el acumulador desde la cadena

O	D	I	S	Z	A	P	C

Carga un byte o palabra de memoria direccionada por DS:(E)SI en el acumulador (AL, AX o EAX). Si se utiliza LODS, debe especificarse el operando de memoria. LODSB carga un byte en AL, LODSW carga una palabra en AX y LODSB en el IA-32 carga una doble palabra en EAX. (E)SI se incrementa o decremente de acuerdo con el tamaño del operando y el estado de la bandera Dirección. Si la bandera Dirección (DF) = 1, (E)SI se decrementa; si DF = 0, (E)SI se incrementa.

Formatos de la instrucción:

**LODS mem
LODS regseg:mem
LODS**

**LODSB
LODSW**

LOOP, LOOPW	Ciclo	O D I S Z A P C	
	Decrementa (E)CX y salta a una etiqueta corta si (E)CX es mayor que cero. El destino debe estar entre -128 a +127 a partir de la ubicación actual. En los procesadores IA-32, ECX se utiliza como el contador de ciclo predeterminado. Formatos de la instrucción:	<input type="text"/>	LOOP <i>l</i> etiquetacorta
LOOPD	Ciclo (IA-32)	O D I S Z A P C	
	Decrementa ECX y salta a una etiqueta corta si ECX es mayor que Cero. El destino debe estar entre -128 y +127 bytes a partir de la ubicación actual. Formato de la instrucción:	<input type="text"/>	LOOPD etiquetacorta
LOOPE, LOOPZ	Salta si es igual (Cero)	O D I S Z A P C	
	Decrementa (E)CX y salta a una etiqueta corta si (E)CX > 0 y se activa la bandera Cero. Formatos de la instrucción:	<input type="text"/>	LOOPE etiquetacorta
LOOPNE, LOOPNZ	Salta si no es igual (Cero)	O D I S Z A P C	
	Decrementa (E)CX y salta a una etiqueta corta si (E)CX > 0 y se borra la bandera Cero. Formatos de la instrucción:	<input type="text"/>	LOOPNE etiquetacorta
MOV	Mover	O D I S Z A P C	
	Copia un byte o palabra de un operando de origen a un operando de destino. Formatos de la instrucción:	<input type="text"/>	MOV <i>reg, reg</i> MOV <i>mem, reg</i> MOV <i>reg, mem</i> MOV <i>reg16, regseg</i> MOV <i>regseg, reg16</i>

B.2 Detalles del conjunto de instrucciones (que no son de punto flotante)

**MOVS,
MOVSB,
MOVSW,
MOVSD**

Mover cadena

635

O	D	I	S	Z	A	P	C

Copia un byte o palabra de la memoria direccionada por DS:(E)SI a la memoria direccionada por ES:(E)DI. MOVS requiere que se especifiquen ambos operandos. MOVSB copia un byte, MOVSW copia una palabra, y en el IA-32, MOVSD copia una doble palabra. (E)SI y (E)DI se incrementan o decrementan de acuerdo con el tamaño del operando y el estado de la bandera Dirección. Si la bandera Dirección (DF) = 1, (E)SI y (E)DI se decrementan; si DF = 0, (E)SI y (E)DI se incrementan.

Formatos de la instrucción:

MOVSB
MOVSW
MOVSD
MOVS dest, origen
MOVS ES:dest, regseg:origen

MOVSX

Mover con extensión de signo

O	D	I	S	Z	A	P	C

Copia un byte o palabra de un operando de origen a un registro de destino y extiende el signo hacia la mitad superior del destino. Esta instrucción se utiliza para copiar un operando de 8 o 16 bits en un destino más grande.

Formatos de la instrucción:

MOVSX reg32,reg16	MOVSX reg32,mem16
MOVSX reg16,reg8	MOVSX reg16,m8

MOVZX

Mover con extensión de ceros

O	D	I	S	Z	A	P	C

Copia un byte o palabra de un operando de origen a un registro de destino y lo extiende con ceros hacia la mitad superior del destino. Esta instrucción se utiliza para copiar un operando de 8 o 16 bits en un destino más grande.

Formatos de la instrucción:

MOVZX reg32,reg16	MOVZX reg32,mem16
MOVZX reg16,reg8	MOVZX reg16,m8

MUL

Multiplicar entero sin signo

O	D	I	S	Z	A	P	C	E	D	X
*				?	?	?	?	*		

AX = AL * reg16
DX:AX = AX * 16 bits
EDX = EAX * 32 bits
EAX = EAX * 32 bits

Multiplica AL, AX o EAX por un operando de origen. Si el origen es de 8 bits, se multiplica por AL y el producto se almacena en AX. Si el origen es de 16 bits, se multiplica por AX y el producto se almacena en DX:AX. Si el origen es de 32 bits, se multiplica por EAX y el producto se almacena en EDX:EAX.

Formatos de la instrucción:

MUL mem

NEG	<p>Negar</p> <table style="margin-left: auto; margin-right: auto;"> <tr> <th>O</th><th>D</th><th>I</th><th>S</th><th>Z</th><th>A</th><th>P</th><th>C</th> </tr> <tr> <td>*</td><td></td><td></td><td>*</td><td>*</td><td>*</td><td>*</td><td>*</td> </tr> </table> <p>Calcula el complemento a dos del operando de destino y almacena el resultado en el destino.</p> <p>Formatos de la instrucción:</p> <table style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 50%;">NEG reg</td> <td style="width: 50%;">NEG mem</td> </tr> </table>	O	D	I	S	Z	A	P	C	*			*	*	*	*	*	NEG reg	NEG mem				
O	D	I	S	Z	A	P	C																
*			*	*	*	*	*																
NEG reg	NEG mem																						
NOP	<p>Ninguna operación</p> <table style="margin-left: auto; margin-right: auto;"> <tr> <th>O</th><th>D</th><th>I</th><th>S</th><th>Z</th><th>A</th><th>P</th><th>C</th> </tr> <tr> <td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td> </tr> </table> <p>Esta instrucción no hace nada, pero puede usarse dentro de un ciclo de sincronización o para alinear una instrucción subsiguiente en un límite de palabra.</p> <p>Formato de la instrucción:</p> <table style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 100%;">NOP</td> </tr> </table>	O	D	I	S	Z	A	P	C									NOP					
O	D	I	S	Z	A	P	C																
NOP																							
NOT	<p>Not</p> <table style="margin-left: auto; margin-right: auto;"> <tr> <th>O</th><th>D</th><th>I</th><th>S</th><th>Z</th><th>A</th><th>P</th><th>C</th> </tr> <tr> <td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td> </tr> </table> <p>Realiza una operación NOT lógica sobre un operando, invirtiendo cada uno de sus bits.</p> <p>Formatos de la instrucción:</p> <table style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 50%;">NOT reg</td> <td style="width: 50%;">NOT mem</td> </tr> </table>	O	D	I	S	Z	A	P	C									NOT reg	NOT mem				
O	D	I	S	Z	A	P	C																
NOT reg	NOT mem																						
OR	<p>OR inclusivo</p> <table style="margin-left: auto; margin-right: auto;"> <tr> <th>O</th><th>D</th><th>I</th><th>S</th><th>Z</th><th>A</th><th>P</th><th>C</th> </tr> <tr> <td>0</td><td></td><td></td><td>*</td><td>*</td><td>?</td><td>*</td><td>0</td> </tr> </table> <p>Realiza una operación OR booleana (a nivel de bits) entre cada bit coincidente en el operando de destino, y cada bit en el operando de origen.</p> <p>Formatos de la instrucción:</p> <table style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 50%;">OR reg, reg</td> <td style="width: 50%;">OR reg, imm</td> </tr> <tr> <td>OR mem, reg</td> <td>OR mem, imm</td> </tr> <tr> <td>OR reg, mem</td> <td>OR acum, imm</td> </tr> </table>	O	D	I	S	Z	A	P	C	0			*	*	?	*	0	OR reg, reg	OR reg, imm	OR mem, reg	OR mem, imm	OR reg, mem	OR acum, imm
O	D	I	S	Z	A	P	C																
0			*	*	?	*	0																
OR reg, reg	OR reg, imm																						
OR mem, reg	OR mem, imm																						
OR reg, mem	OR acum, imm																						
OUT	<p>Salida a un puerto</p> <table style="margin-left: auto; margin-right: auto;"> <tr> <th>O</th><th>D</th><th>I</th><th>S</th><th>Z</th><th>A</th><th>P</th><th>C</th> </tr> <tr> <td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td> </tr> </table> <p>En procesadores anteriores al IA-32, esta instrucción envía un byte o palabra del acumulador a un puerto. La dirección del puerto puede ser una constante si se encuentra en el rango 0-FFh, o DX puede contener una dirección de puerto entre 0 y FFFFh. En un procesador IA-32, puede enviarse una doble palabra a un puerto.</p> <p>Formatos de la instrucción:</p> <table style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 50%;">OUT imm8, acum</td> <td style="width: 50%;">OUT DX, acum</td> </tr> </table>	O	D	I	S	Z	A	P	C									OUT imm8, acum	OUT DX, acum				
O	D	I	S	Z	A	P	C																
OUT imm8, acum	OUT DX, acum																						

B.2 Detalles del CONJUNTO DE INSTRUCCIONES (QUE NO SON DE PUNTO FLOTANTE)

635

**OUTS,
OUTSB,
OUTSW,
OUTSD**

Enviar cadena a un puerto (80286)

O	D	I	S	Z	A	P	C
[]	[]	[]	[]	[]	[]	[]	[]

Envía una cadena a la que apunta ES:(E)DI a un puerto. El número de puerto se especifica en DX. Para cada valor que se envía, (E)DI se ajusta de la misma forma que LODSB y las instrucciones de primitivas de cadena similares. Puede usarse el prefijo REP con esta instrucción.

Formatos de la instrucción:

OUTS dest,DX
OUTSW dest,DX

REP OUTSB dest,DX
REP OUTSD dest,DX

POP

Sacar de la pila

O	D	I	S	Z	A	P	C
[]	[]	[]	[]	[]	[]	[]	[]

Copia una palabra o doble palabra en la ubicación actual del apuntador de pila al operando de destino, y suma 2 (o 4) a (E)SP.

Formatos de la instrucción:

POP reg16/r32
POP mem16/mem32

POP regseg

**POPA,
POPAD**

Sacar todos

O	D	I	S	Z	A	P	C
[]	[]	[]	[]	[]	[]	[]	[]

Saca 16 bytes de la parte superior de la pila y los coloca en los ocho registros de propósito general, en el siguiente orden: DI, SI, BP, SP, BX, DX, CX, AX. El valor para SP se descarta, por lo que SP no se reasigna. POPA saca hacia registros de 16 bits y POPAD en el IA-32 saca hacia registros de 32 bits.

Formatos de la instrucción:

POPA

POPAD

**POPF,
POPFD**

Sacar banderas de la pila

O	D	I	S	Z	A	P	C
*	*	*	*	*	*	*	*

POPF saca la parte superior de la pila y la coloca en el registro FLAGS de 16 bits. POPFD en el IA-32 saca la parte superior de la pila y la coloca en el registro EFLAGS de 32 bits.

Formatos de la instrucción:

POPF

POPFD

PUSH**Meter en la pila**

O	D	I	S	Z	A	P	C
<input type="text"/>							

Resta 2 de (E)SP y copia el operando de origen en la ubicación de la pila a la que apunta (E)SP. Del 80186 en adelante, puede meterse un valor inmediato en la pila.

Formatos de la instrucción:

PUSH reg16/reg32
PUSH mem16/mem32

PUSH regseg
PUSH inm16/inm32

**PUSHA,
PUSHAD****Meter todos (80286)**

O	D	I	S	Z	A	P	C
<input type="text"/>							

Mete los siguientes registros de 16 bits en la pila, en orden: AX, CX, DX, BX, SP, BP, SI y DI. La instrucción PUSHAD para el IA-32 mete EAX, ECX, EDX, EBX, ESP, EBP, ESI y EDI.

Formatos de la instrucción:

PUSHA

PUSHAD

**PUSHF,
PUSHFD****Meter banderas**

O	D	I	S	Z	A	P	C
<input type="text"/>							

PUSHF mete el registro FLAGS de 16 bits en la pila. PUSHFD mete el registro EFLAGS de 32 bits en la pila (IA-32).

Formatos de la instrucción:

PUSHF

PUSHFD

**PUSHW,
PUSHD****Meter en la pila**

O	D	I	S	Z	A	P	C
<input type="text"/>							

PUSHW mete una palabra de 16 bits en la pila, y en el IA-32, PUSHD mete una doble palabra de 32 bits en la pila.

Formatos de la instrucción:

PUSH reg16/reg32
PUSH mem16/mem32

PUSH regseg
PUSH inm16/inm32

RCL**Rotar acarreo a la izquierda**

O	D	I	S	Z	A	P	C
*							*

Rota el operando de destino a la izquierda, usando el operando de origen para determinar el número de rotaciones. La bandera Acarreo se copia al bit más inferior, y el bit más superior se copia a la bandera Acarreo. El operando *inm8* debe ser 1 al utilizar el procesador 8086/8088.

Formatos de la instrucción:

RCL *reg, inm8*
RCL *reg, CL*

RCL *mem, inm8*
RCL *mem, CL*

RCR**Rotar acarreo a la derecha**

O	D	I	S	Z	A	P	C
*							*

Rota el operando de destino a la derecha, usando el operando de origen para determinar el número de rotaciones. La bandera Acarreo se copia al bit más superior, y el bit más inferior se copia a la bandera Acarreo. El operando *inm8* debe ser 1 al utilizar el procesador 8086/8088.

Formatos de la instrucción:

RCR *reg, inm8*
RCR *reg, CL*

RCR *mem, inm8*
RCR *mem, CL*

REP**Repetir cadena**

O	D	I	S	Z	A	P	C

Repite una instrucción de primitiva de cadena, usando (E)CX como contador. (E)CX se decremente cada vez que se repite la instrucción, hasta que (E)CX = 0.

Formato (se muestra con MOVS):

REP MOVS *dest, origen*

REPcondición**Repetir cadena condicionalmente**

O	D	I	S	Z	A	P	C

Repite una instrucción de primitiva de cadena hasta que (E)CX = 0 y mientras sea verdadera una condición de bandera. REPZ (REPE) se repite mientras la bandera Cero esté activa, y REPZ (REPNE) se repite mientras la bandera Cero esté en cero. Sólo deben usarse SCAS y CMPS con REP condición, ya que son las únicas primitivas de cadena que modifican la bandera Cero.

Formatos utilizados con SCAS:

REP SCAS *dest*
REPZ SCASB
REPE SCASW

REPNE SCAS *dest*
REPNE SCASB
REPNZ SCASW

**RET,
RETN,
RETF****Retornar de un procedimiento**

O	D	I	S	Z	A	P	C

Saca una dirección de retorno de la pila. RETN (retorno cercano) saca sólo la parte superior de la pila y la coloca en (E)IP. En modo de direccionamiento real, RETF (retorno lejano) saca la pila y la coloca primero en (E)IP y después en CS. RET puede ser cercana o lejana, dependiendo del atributo especificado o implicado por la directiva PROC. Un operando inmediato opcional de 8 bits indica a la CPU que debe sumar un valor a (E)SP después de sacar la dirección de retorno.

Formatos de la instrucción:

RET

RETN

RETF

RET *inm8*RETN *inm8*RETF *inm8***ROL****Rotar a la izquierda**

O	D	I	S	Z	A	P	C
*							*

Rota el operando de destino a la izquierda, usando el operando de origen para determinar el número de rotaciones. El bit superior se copia en la bandera Acarreo y se mueve hacia la posición del bit más inferior. El operando *inm8* debe ser 1 al utilizar el procesador 8086/8088.

Formatos de la instrucción:

ROL *reg, inm8*ROL *reg, CL*ROL *mem, inm8*ROL *mem, CL***ROR****Rotar a la derecha**

O	D	I	S	Z	A	P	C
*							*

Rota el operando de destino a la derecha, usando el operando de origen para determinar el número de rotaciones. El bit inferior se copia en la bandera Acarreo y en la posición del bit superior. El operando *inm8* debe ser 1 al utilizar el procesador 8086/8088.

Formatos de la instrucción:

ROR *reg, inm8*ROR *reg, CL*ROR *mem, inm8*ROR *mem, CL***SAHF****Almacena AH en el registro Flags**

O	D	I	S	Z	A	P	C
			*	*	*	*	*

Copia AH en los bits del 0 al 7 del registro Flags.

Formato de la instrucción:

SAHF

B.2 DETALLES DEL CONJUNTO DE INSTRUCCIONES (QUE NO SON DE PUNTO FLOTANTE)

639

SAL

Desplazamiento aritmético a la izquierda

O	D	I	S	Z	A	P	C
*			*	*	?	*	*

Desplaza cada bit en el operando de destino a la izquierda, usando el operando de origen para determinar el número de desplazamientos. El bit superior se copia en la bandera Acarreo y el bit inferior se llena con un cero. El operando imm8 debe ser 1 al utilizar el procesador 8086/8088.

Formatos de la instrucción:

SAL reg, imm8
SAL reg, CL

SAL mem, imm8
SAL mem, CL

SAR

Desplazamiento aritmético a la derecha

O	D	I	S	Z	A	P	C
*			*	*	?	*	*

Desplaza cada bit en el operando de destino a la derecha, usando el operando de origen para determinar el número de desplazamientos. El bit inferior se copia en la bandera Acarreo y el bit superior retiene su valor anterior. Este desplazamiento se utiliza a menudo con los operandos con signo, ya que preserva el signo del número. El operando imm8 debe ser 1 al utilizar el procesador 8086/8088.

Formatos de la instrucción:

SAR reg, imm8
SAR reg, CL

SAR mem, imm8
SAR mem, CL

SBB

Resta con préstamo

O	D	I	S	Z	A	P	C
*			*	*	*	*	*

Resta el operando de origen al operando de destino y después resta la bandera Acarreo al destino.

Formatos de la instrucción:

SBB reg, reg
SBB mem, reg
SBB reg, mem

SBB reg, imm
SBB mem, imm

**SCAS,
SCASB,
SCASW,
SCASD**

Explorar cadena

O	D	I	S	Z	A	P	C
*			*	*	*	*	*

Explora una cadena en memoria a la que apunta ES:(E)DI, en busca de un valor que coincida con el acumulador. SCAS requiere que se especifiquen los operandos. SCASB explora en busca de un valor de 16 bits que coincida con AX, y SCASD explora en busca de un valor de 32 bits que coincide con EAX. (E)DI se incrementa o decremente de acuerdo con el tamaño del operando y el estado de la bandera Dirección. Si DF = 1, (E)DI se decrementa; si DF = 0, (E)DI se incrementa.

Formatos de la instrucción:

SCASB
SCASD
SCAS ES:dest

SCASW

SETcondición	Activar condicionalmente																							
	<table border="1" style="margin-left: auto; margin-right: auto;"> <tr> <td>O</td><td>D</td><td>I</td><td>S</td><td>Z</td><td>A</td><td>P</td><td>C</td></tr> <tr> <td>*</td><td>*</td><td>*</td><td>*</td><td>*</td><td>*</td><td>*</td><td>*</td></tr> </table> <p>Si la condición dada de la bandera es verdadera, se asigna el valor 1 al byte especificado por el operando de destino. Si la condición de la bandera es falsa, se asigna un valor de 0 al destino. En la tabla B-2 se listan los valores posibles para condición.</p> <p>Formatos de la instrucción:</p> <table style="margin-left: auto; margin-right: auto;"> <tr> <td><code>SETcond reg8</code></td> <td><code>SETcond mem8</code></td> </tr> </table>	O	D	I	S	Z	A	P	C	*	*	*	*	*	*	*	*	<code>SETcond reg8</code>	<code>SETcond mem8</code>					
O	D	I	S	Z	A	P	C																	
*	*	*	*	*	*	*	*																	
<code>SETcond reg8</code>	<code>SETcond mem8</code>																							
SHL	Desplazar a la izquierda																							
	<table border="1" style="margin-left: auto; margin-right: auto;"> <tr> <td>O</td><td>D</td><td>I</td><td>S</td><td>Z</td><td>A</td><td>P</td><td>C</td></tr> <tr> <td>*</td><td>*</td><td>*</td><td>*</td><td>*</td><td>?</td><td>*</td><td>*</td></tr> </table> <p>Desplaza cada bit en el operando de destino a la izquierda, usando el operando de origen para determinar el número de desplazamientos. El bit superior se copia en la bandera Acarreo, y el bit inferior se llena con un cero (en forma idéntica a SAL). El operando <i>inm8</i> debe ser 1 si se utiliza el procesador 8086/8088.</p> <p>Formatos de la instrucción:</p> <table style="margin-left: auto; margin-right: auto;"> <tr> <td><code>SHL reg, inm8</code></td> <td><code>SHL mem, inm8</code></td> </tr> <tr> <td><code>SHL reg, CL</code></td> <td><code>SHL mem, CL</code></td> </tr> </table>	O	D	I	S	Z	A	P	C	*	*	*	*	*	?	*	*	<code>SHL reg, inm8</code>	<code>SHL mem, inm8</code>	<code>SHL reg, CL</code>	<code>SHL mem, CL</code>			
O	D	I	S	Z	A	P	C																	
*	*	*	*	*	?	*	*																	
<code>SHL reg, inm8</code>	<code>SHL mem, inm8</code>																							
<code>SHL reg, CL</code>	<code>SHL mem, CL</code>																							
SHLD	Desplazar a la izquierda con doble precisión (IA-32)																							
	<table border="1" style="margin-left: auto; margin-right: auto;"> <tr> <td>O</td><td>D</td><td>I</td><td>S</td><td>Z</td><td>A</td><td>P</td><td>C</td></tr> <tr> <td>*</td><td>*</td><td>*</td><td>*</td><td>*</td><td>?</td><td>*</td><td>*</td></tr> </table> <p>Desplaza los bits del segundo operando hacia el primer operando. El tercer operando indica el número de bits a desplazar. Las posiciones abiertas por el desplazamiento se llenan con los bits más significativos del segundo operando. Éste siempre debe ser un registro, y el tercer operando puede ser un valor inmediato o el registro CL.</p> <p>Formatos de la instrucción:</p> <table style="margin-left: auto; margin-right: auto;"> <tr> <td><code>SHLD reg16, reg16, inm8</code></td> <td><code>SHLD mem16, reg16, inm8</code></td> </tr> <tr> <td><code>SHLD reg32, reg32, inm8</code></td> <td><code>SHLD mem32, reg32, inm8</code></td> </tr> <tr> <td><code>SHLD reg16, reg16, CL</code></td> <td><code>SHLD mem16, reg16, CL</code></td> </tr> <tr> <td><code>SHLD reg32, reg32, CL</code></td> <td><code>SHLD mem32, reg32, CL</code></td> </tr> </table>	O	D	I	S	Z	A	P	C	*	*	*	*	*	?	*	*	<code>SHLD reg16, reg16, inm8</code>	<code>SHLD mem16, reg16, inm8</code>	<code>SHLD reg32, reg32, inm8</code>	<code>SHLD mem32, reg32, inm8</code>	<code>SHLD reg16, reg16, CL</code>	<code>SHLD mem16, reg16, CL</code>	<code>SHLD reg32, reg32, CL</code>
O	D	I	S	Z	A	P	C																	
*	*	*	*	*	?	*	*																	
<code>SHLD reg16, reg16, inm8</code>	<code>SHLD mem16, reg16, inm8</code>																							
<code>SHLD reg32, reg32, inm8</code>	<code>SHLD mem32, reg32, inm8</code>																							
<code>SHLD reg16, reg16, CL</code>	<code>SHLD mem16, reg16, CL</code>																							
<code>SHLD reg32, reg32, CL</code>	<code>SHLD mem32, reg32, CL</code>																							
SHR	Desplazar a la derecha																							
	<table border="1" style="margin-left: auto; margin-right: auto;"> <tr> <td>O</td><td>D</td><td>I</td><td>S</td><td>Z</td><td>A</td><td>P</td><td>C</td></tr> <tr> <td>*</td><td>*</td><td>*</td><td>*</td><td>*</td><td>?</td><td>*</td><td>*</td></tr> </table> <p>Desplaza cada bit en el operando de destino a la derecha, usando el operando de origen para determinar el número de desplazamientos. El bit superior se llena con un cero, y el bit inferior se copia en la bandera Acarreo. El operando <i>inm8</i> debe ser 1 si se utiliza el procesador 8086/8088.</p> <p>Formatos de la instrucción:</p> <table style="margin-left: auto; margin-right: auto;"> <tr> <td><code>SHR reg, inm8</code></td> <td><code>SHR mem, inm8</code></td> </tr> <tr> <td><code>SHR reg, CL</code></td> <td><code>SHR mem, CL</code></td> </tr> </table>	O	D	I	S	Z	A	P	C	*	*	*	*	*	?	*	*	<code>SHR reg, inm8</code>	<code>SHR mem, inm8</code>	<code>SHR reg, CL</code>	<code>SHR mem, CL</code>			
O	D	I	S	Z	A	P	C																	
*	*	*	*	*	?	*	*																	
<code>SHR reg, inm8</code>	<code>SHR mem, inm8</code>																							
<code>SHR reg, CL</code>	<code>SHR mem, CL</code>																							

SHRD

Desplazar a la derecha con doble precisión (IA-32)

O	D	I	S	Z	A	P	C
*			*	*	?	*	*

Desplaza los bits del segundo operando hacia el primer operando. El tercer operando indica el número de bits a desplazar. Las posiciones abiertas por el desplazamiento se llenan con los bits menos significativos del segundo operando. Éste siempre debe ser un registro, y el tercer operando puede ser un valor inmediato o el registro CL.

Formatos de la instrucción:

SHRD reg16, reg16, imm8
 SHRD reg32, reg32, imm8
 SHRD reg16, reg16, CL
 SHRD reg32, reg32, CL

SHRD mem16, reg16, imm8
 SHRD mem32, reg32, imm8
 SHRD mem16, reg16, CL
 SHRD mem32, reg32, CL

STC

Activar bandera Acarreo

O	D	I	S	Z	A	P	C
							1

Establece la bandera Acarreo.

Formato de la instrucción:

STC

STD

Activar bandera Dirección

O	D	I	S	Z	A	P	C
		1					

Establece la bandera Dirección, haciendo que (E)SI y/o (E)DI se decrementen mediante instrucciones de primitivas de cadena. Por ende, el procesamiento de las cadenas será desde las direcciones superiores, hasta las direcciones inferiores.

Formato de la instrucción:

STD

STI

Activar bandera Interrupción

O	D	I	S	Z	A	P	C
		1					

Establece la bandera Interrupción, la cual habilita las interrupciones enmascarables. Las interrupciones se deshabilitan de manera automática cuando ocurre una interrupción, por lo que un procedimiento manejador de interrupciones las rehabilita de manera inmediata, usando STI.

Formato de la instrucción:

STI

SHRD

Desplazar a la derecha con doble precisión (IA-32)

O	D	I	S	Z	A	P	C
*			*	*	?	*	*

Desplaza los bits del segundo operando hacia el primer operando. El tercer operando indica el número de bits a desplazar. Las posiciones abiertas por el desplazamiento se llenan con los bits menos significativos del segundo operando. Éste siempre debe ser un registro, y el tercer operando puede ser un valor inmediato o el registro CL.

Formatos de la instrucción:

SHRD reg16, reg16, inm8
 SHRD reg32, reg32, inm8
 SHRD reg16, reg16, CL
 SHRD reg32, reg32, CL

SHRD mem16, reg16, inm8
 SHRD mem32, reg32, inm8
 SHRD mem16, reg16, CL
 SHRD mem32, reg32, CL

STC

Activar bandera Acarreo

O	D	I	S	Z	A	P	C
							1

Establece la bandera Acarreo.

Formato de la instrucción:

STC

STD

Activar bandera Dirección

O	D	I	S	Z	A	P	C
		1					

Establece la bandera Dirección, haciendo que (E)SI y/o (E)DI se decrementen mediante instrucciones de primitivas de cadena. Por ende, el procesamiento de las cadenas será desde las direcciones superiores, hasta las direcciones inferiores.

Formato de la instrucción:

STD

STI

Activar bandera Interrupción

O	D	I	S	Z	A	P	C
		1					

Establece la bandera Interrupción, la cual habilita las interrupciones enmascarables. Las interrupciones se deshabilitan de manera automática cuando ocurre una interrupción, por lo que un procedimiento manejador de interrupciones las rehabilita de manera inmediata, usando STI.

Formato de la instrucción:

STI

STOS, STOSB, STOSW, STOSD

Almacenar datos de cadena

O	D	I	S	Z	A	P	C

Almacena el acumulador en la ubicación de memoria direccionada por ES:(E)DI. Si se utiliza STOS, debe especificarse un operando de destino. STOSB copia AL a la memoria, STOSW copia AX a la memoria, y STOSD para el IA-32 copia EAX a la memoria. (E)DI se incrementa o decrementa de acuerdo con el tamaño del operando y el estado de la bandera Dirección. Si DF = 1, (E)DI se decremente; si DF = 0, (E)DI se incrementa.

Formatos de la instrucción:

STOSB
STOSD
STOS mem
STOS ES:dest

STOSW

SUB

Restar

O	D	I	S	Z	A	P	C
*			*	*	*	*	*

Resta el operando de destino del operando de origen.

Formatos de la instrucción:

SUB reg,reg
SUB mem,reg
SUB reg,mem

SUB reg,inm
SUB mem,inm
SUB acum,inm

TEST

Probar

O	D	I	S	Z	A	P	C
0			*	*	?	*	0

Prueba bits individuales en el operando de destino, comparándolos con los del operando de origen. Realiza una operación AND lógica que afecta a las banderas, pero no al operando de destino.

Formatos de la instrucción:

TEST reg,reg
TEST mem,reg
TEST reg,mem

TEST reg,inm
TEST mem,inm
TEST acum,inm

WAIT

Esperar al coprocesador

O	D	I	S	Z	A	P	C

Suspende la ejecución de la CPU hasta que el coprocesador termine la instrucción actual.

Formato de la instrucción:

WAIT

XADD**Intercambiar y sumar (Intel486)**

O	D	I	S	Z	A	P	C
*			*	*	*	*	*

Suma el operando de origen al operando de destino. Al mismo tiempo, el valor de destino original se mueve al operando de origen.

Formatos de la instrucción:

XADD reg, reg

XADD mem, reg

XCHG**Intercambiar**

O	D	I	S	Z	A	P	C

Intercambia el contenido de los operandos de origen y de destino.

Formatos de la instrucción:

XCH reg, reg

XCH mem, reg

XCH reg, mem

**XLAT,
XLATB****Traducir byte**

O	D	I	S	Z	A	P	C

Usa el valor en AL como índice en una tabla a la que apunta DS:BX. El byte al que apunta el índice se mueve hacia AL. Puede especificarse un operando para proporcionar una redefinición de segmento. Se puede usar XLATB en vez de XLAT.

Formatos de la instrucción:

XLAT

XLAT regseg:mem

XLAT mem

XLATB

XOR**OR exclusivo**

O	D	I	S	Z	A	P	C
0			*	*	?	*	0

Se aplica un OR exclusivo a cada bit en el operando de origen con su bit correspondiente en el destino. El bit de destino es 1 sólo cuando los bits de origen y de destino son distintos.

Formatos de la instrucción:

XOR reg, reg

XOR reg, imm

XOR mem, reg

XOR mem, imm

XOR reg, mem

XOR acum, imm