

Lenguaje ensamblador

Manual de bolsillo

Allen L. Wyatt, Sr.

Versión en español de
Catalina Domínguez Reyes
México, D.F.

Con la colaboración de
Roberto Escalona
México, D.F.

ADDISON-WESLEY IBEROAMERICANA
Argentina • Brasil • Chile • Colombia • Ecuador • España

Editorial Addison Wesley-P. O. Box 936-Bethesda, Maryland 20034

Nota al lector

Obra basada en los conjuntos de instrucciones de los siguientes microprocesadores de Intel y procesadores numéricos:

8086/8088	8087
80286	80287
80386	80387

Créditos

Créditos de marcas registradas

Intel es una marca registrada de Intel Corporation.

– Índice general

Introducción	1
La familia de microprocesadores	
8086/8088 de Intel	1
Compendio del 8086/8088 de Intel	3
Conjunto de instrucciones del 8086/8088 de Intel	8
80286 de Intel	56
Conjunto de instrucciones del 80286 de Intel	58
80386 de Intel	69
Conjunto de instrucciones del 80386 de Intel	74
Coprocesadores numéricos de Intel	117
Compendio del 8087 de Intel	120
Conjunto de instrucciones del 8087 de Intel	124
Conjunto de instrucciones del 80287 de Intel	154
Compendio del 80387 de Intel	155
Conjunto de instrucciones del 80387 de Intel	157
Índice de materias	161

Version en español de la obra titulada *Assembly Language Quick Reference* de la serie *Que® Quick Reference Series*, publicada originalmente en inglés por Que® Corporation, Carmel, Indiana, E.U.A. © 1989 por Que Corporation.

Esta edición en español es la única autorizada.

©1990 por ADDISION-WESLEY IBEROAMERICANA, S.A.
Wilmington, Delaware, E.U.A.

Reservados todos los derechos. Ni todo el libro ni parte de él pueden ser reproducidos, archivados o transmitidos en forma alguna o mediante algún sistema electrónico, mecánico de fotoreproducción, memoria o cualquier otro, sin permiso por escrito del editor.

Impreso en Estados Unidos. *Printed in U.S.A.*

ISBN 0-201-51834-1
3 4 5 6 7 8 9 10-DM-96 95 94 93 92

Los Manuales de bolsillo de Addison-Wesley Iberoamericana

Al ponerse a trabajar con un computador —lo mismo para principiantes que para usuarios experimentados— es de gran ayuda tener a la mano la información básica del programa, lenguaje, sistema o equipo que se está usando, para dar solución inmediata a los problemas que se presenten. *Los Manuales de bolsillo de Addison-Wesley Iberoamericana cumplen sobradamente esta función.*

Los títulos de esta colección abarcan los temas de mayor interés y utilidad en los diversos ámbitos en que se emplean los computadores personales: procesamiento de textos, bases de datos, etcétera. En un solo texto usted encontrará, presentados en forma concisa y de fácil consulta, los mandatos y funciones básicas de programas como 1-2-3 y Word Perfect; los fundamentos de la programación en lenguajes como C y Quick BASIC 4, y hasta la respuesta a sus dudas respecto al manejo del disco duro.

Títulos disponibles: Títulos en preparación:

1-2-3	1-2-3 Versión 3
MS-DOS	1-2-3 Versión 2.2
Word Perfect	Disco duro
C	dBASE IV™
Quick BASIC	PC Tools
AutoCAD	
Turbo Pascal™	
Funciones de DOS y BIOS	
Microsoft® WORDS	

Introducción

Intel™ ha creado una familia de microprocesadores que constituyen el centro de la mayoría de los micro-computadores actualmente en uso. Debido al predominio de éstos en todo el mundo, los programadores necesitan un material de consulta rápida que les refresque la memoria respecto a instrucciones de máquina específicas.

Este libro pretende servir como guía de los conjuntos de instrucciones de los siguientes microprocesadores y coprocesadores numéricos de Intel:

- Microprocesador 8086/8088
- Microprocesador 80286
- Microprocesador 80386
- Coprocesador numérico 8087
- Coprocesador numérico 80287
- Coprocesador numérico 80387

Que Corporation publica otros libros que los programadores de lenguaje ensamblador pueden encontrar útiles en su trabajo. Entre éstos se encuentran

*Using Assembly Language
Dos Programmer's Reference*

La familia de microprocesadores 8086/8088 de Intel

La popular familia de microprocesadores 8086/8088 de Intel está constituida por varias generaciones diferentes de microprocesadores. Cada generación se basa en la anterior, ampliando el conjunto de instrucciones y las capacidades de la familia. El 8086 y el 8088, los

primeros miembros de esta familia, comparten un conjunto de instrucciones comunes que forman la base de todas las generaciones sucesivas.

Los microprocesadores posteriores, como el 80286 y el 80386, incluyen las mismas instrucciones empleadas en el 8086/8088. Dicho conjunto de instrucciones básico se explica con detalle en esta sección. Más adelante se describen las extensiones de este conjunto básico tanto para el 80286 como para el 80386.

La familia de instrucciones 8086/8088 Intel puede agruparse en seis clasificaciones generales, de acuerdo con el propósito de la instrucción:

- Transferencia de datos
- Aritmética
- Manipulación de bits
- Manipulación de cadenas
- Transferencia de control
- Control de banderas y del procesador

Las instrucciones individuales que se hallan dentro de cada clasificación general se enlistan en una sección de recapitulación titulada, por ejemplo, "Compendio del 8086/8088". Hay secciones similares para cada uno de los restantes microprocesadores.

A lo largo de este volumen, cada instrucción se describe con detalle. Las instrucciones están dispuestas en orden alfabético por el microprocesador (8086/8088, 80286 y 80386). Para cada instrucción se da la siguiente información:

Nombre de la instrucción. Este nombre se basa en el código nemotécnico diseñado por Intel. También se proporciona la clasificación general de la instrucción, junto con una descripción de ésta.

Banderas afectadas. La mayoría de las instrucciones cambia el estado de los bits en el registro de banderas. Si no hay banderas afectadas, entonces no se incluye esta sección. Las banderas individuales afectadas son las siguientes:

<u>Abreviatura</u>	<u>Significado</u>
OF	Sobreflujo
DF	Dirección
IF	Interrupción
TF	Desviación
SF	Signo
ZF	Cero
AF	Acarreo auxiliar
PF	Paridad
CF	Acarreo

Ejemplos de codificación. Se proporcionan ejemplos breves del uso de la instrucción. Se incluye un ejemplo únicamente si el nombre de la instrucción no se usa solo; es decir, se necesitan operandos o instrucciones adicionales para hacer que la instrucción trabaje.

Compendio del 8086/8088 de Intel

La siguiente lista contiene el conjunto de instrucciones completo para el 8086/8088 de Intel, dispuesto en orden alfabético:

AAA	CMPSB	INTO
AAD	CMPSW	IRET
AAM	CWD	JA
AAS	DAA	JAE
ADC	DAS	JB
ADD	DEC	JBE
AND	DIV	JC
CALL	ESC	JCXZ
CBW	HLT	JE
CLC	IDIV	JG
CLD	IMUL	JGE
CLI	IN	JL
CMC	INC	JLE
CMP	INT	JMP

JNA	LODSB	REPNE	IMUL	JGE	MOV	
JNAE	LODSW	REPNZ	INC	JL	OUT	
JNB	LOOP	REPZ	MUL	JLE	POP	
JNBE	LOOPE	RET	NEG	JMP	POPF	
JNC	LOOPNE	ROL	SBB	JNA	PUSH	
JNE	LOOPNZ	ROR	SUB	JNAE	PUSHF	
JNG	LOOPZ	SAHF	<i>Manipulación de bits</i>		SAHF	
JNL	MOV	SAL	AND	JNBE	XCHG	
JNLE	MOVS B	SAR	NOT	JNC	XLAT	
JNO	MOVSW	SBB	OR	JNE	<i>Control de banderas y procesador</i>	
JNP	MUL	SCASB	RCL	JNG	CLC	
JNS	NEG	SCASW	RCR	JNL	CLD	
JNZ	NOP	SHL	ROL	JNLE	CLI	
JO	NOT	SHR	ROR	JNO	CMC	
JP	OR	STC	SAL	JNP	ESC	
JPE	OUT	STD	SAR	JNS	HLT	
JPO	POP	STI	SHL	JNZ	LOCK	
JS	POPF	STOSB	SHR	JO	NOP	
JZ	PUSH	STOSW	TEST	JP	STC	
LAHF	PUSHF	SUB	XOR	JPE	STD	
LDS	RCL	TEST	<i>Transferencia de control</i>		STI	
LEA	RCR	WAIT	CALL	LOOP	WAIT	
LES	REP	XCHG	INT	LOOPE	<i>Manipulación de cadenas</i>	
LOCK	REPE	XLAT	INTO	LOOPNE	CMPSB	
		XOR	IRET	LOOPZ	CMPSW	
			JA	RET	IN	
			JAE	<i>Transferencia de datos</i>	LODSB	
			JB	LAHF	LODSW	
			JBE	LDS	MOVSB	
			JC	LEA	MOVSW	
			JCXZ	LES	REP	
			JE		REPE	
			JG			

La siguiente lista incluye el conjunto de instrucciones completo para el 8086/8088 de Intel, ordenado por la clasificación general de la instrucción. Dentro de cada clasificación, las instrucciones se listan en orden alfabético.

Aritmética	ADC	DAA
AAA	ADD	DAS
AAD	CBW	DEC
AAM	CMP	DIV
AAS	CWD	IDIV

REPNE	REPZ	STOSB
REPNZ	SCASB	STOSW
	SCASW	

En la siguiente tabla se detallan las instrucciones 8086/8088 que afectan banderas y la manera en que esas banderas se ven afectadas. La X en una columna indica que la bandera es modificada por la instrucción; el signo de interrogación indica que la bandera no está definida después de que se ejecuta la instrucción. En esta tabla se incluyen sólo aquellas instrucciones que afectan banderas.

<i>Inst</i>	<i>QE</i>	<i>DE</i>	<i>IE</i>	<i>TE</i>	<i>SE</i>	<i>ZE</i>	<i>AE</i>	<i>PE</i>	<i>CF</i>
A8A	?				?	?	X	?	X
AAD	?				X	X	?	X	?
AAM	?				X	X	?	X	?
AAS	?				?	?	X	?	X
ADC	X				X	X	X	X	X
ADD	X				X	X	X	X	X
AND	X				X	X	?	X	X
CLC							X		
CLD		X							
CLI			X						
CMC						X			
CMP	X				X	X	X	X	X
CMPSB	X				X	X	X	X	X
CMPSW	X				X	X	X	X	X
DAA	?				X	X	X	X	X
DAS	?				X	X	X	X	X
DEC	X				X	X	X	X	
DIV	?				?	?	?	?	?

<i>Inst</i>	<i>QE</i>	<i>DE</i>	<i>IE</i>	<i>TE</i>	<i>SE</i>	<i>ZE</i>	<i>AE</i>	<i>PE</i>	<i>CF</i>
IDIV	?					?	?	?	?
IMUL	X					?	?	?	X
INC	X					X	X	X	X
INT					X	X			
IRET	X	X	X	X	X	X	X	X	X*
MUL	X					?	?	?	X
NEG	X					X	X	X	X
OR	X					X	X	?	X
POPF	X	X	X	X	X	X	X	X	X*
RCL	X								X
RCR	X								X
ROL	X								X
SAHF						X	X	X	X
SAL	X					X	X	?	X
SAR	X					X	X	?	X
SBB	X					X	X	X	X
SCASB	X					X	X	X	X
SCASW	X					X	X	X	X
SHL	X					X	X	?	X
SHR	X					X	X	?	X
STC									X
STD			X						
STI				X					
SUB	X					X	X	X	X
TEST	X					X	X	?	X
XOR	X					X	X	?	X

* También afecta a NT y a IOPL para el 802866

Conjunto de instrucciones del 8086/8088 de Intel

Las instrucciones presentadas en esta sección funcionan como se describe en el 8086/8088, el 80286 y el el 80386.

- AAA

Aritmética

ASCII Adjust for Addition

(Ajuste ASCII para suma): AAA cambia el contenido de AL a un número decimal no empacado válido con el nibble de alto orden en ceros.

Banderas afectadas

AF, CF, OF (no definida), SF (no definida), ZF (no definida), PF (no definida)

- AAD

Aritmética

ASCII Adjust for Division

(Ajuste ASCII para división): AAD multiplica el contenido de AH por 10, añade el resultado al contenido del AL y ubica dicho resultado en AL. Luego, la instrucción pone AH en 0. Esta instrucción se usa antes de dividir números decimales no empacados.

Banderas afectadas

SF, ZF, PE, OF (no definida), AF (no definida), CF (no definida)

- AAM

Aritmética

ASCII Adjust for Multiplication

(Ajuste ASCII para multiplicación): Despues de multiplicar dos números decimales no empacados, se utiliza AAM a fin de corregir el resultado para un número decimal no empacado. Para que la instrucción trabaje adecuadamente, los nibbles de alto orden de cada número multiplicando deben ponerse en 0.

Banderas afectadas

SF, ZF, PF, OF (no definida), AF (no definida), CF (no definida)

- AAS

Aritmética

ASCII Adjust for Subtraction

(Ajuste ASCII para resta): AAS corrige el resultado de una resta decimal no empacada anterior, de tal forma que el valor en AL es un verdadero número decimal no empacado.

Banderas afectadas

AF, CF, OF (no definida), SF (no definida), ZF (no definida), PF (no definida)

- ADC

Aritmética

Add with carry

(Sumar con acarreo): ADC suma el contenido del operando fuente al operando de destino (y almacena el resultado en este último). Si la bandera de acarreo está activada, el resultado cambia en incrementos de 1. En

esta rutina se supone que los valores que se están añadiendo son binarios.

Banderas afectadas

OF, SF, ZF, AF, PF, CF

Ejemplos de codificación

ADC AX, BX	;AX=AX+BX+CF
ADC AX, TEMP	;AX=AX+TEMP+CF
ADC SUM, BX	;SUM=SUM+BX+CF
ADC CL,10	;CL=CL+10+CF
ADC AX, TEMP[BX]	;Dirección indirecta

ADD

Aritmética

Add:

(Sumar): ADD añade el contenido del operando fuente al operando de destino (y almacena el resultado en este último). En esta rutina se supone que los valores que se están añadiendo son binarios.

Banderas afectadas

OF, SF, ZF, AF, PF, CF

Ejemplos de codificación

ADD AX, BX	;AX=AX+BX
ADD AX, TEMP	;AX=AX+TEMP
ADD SUM, BX	;SUM=SUM+BX
ADD CL,10	;CL=CL+10
ADD AX, TEMP[BX]	;Dirección indirecta

AND

Manipulación de bits

Logical AND on Bits

(Y lógico sobre bits): Esta instrucción realiza un Y lógico de los operandos y almacena el resultado en el

operando de destino. Cada bit del byte o palabra resultante se pone en 1 sólo si el bit correspondiente de cada operando se pone en 1.

Banderas afectadas

OF, SF, ZF, PF, CF, AF (no definida)

Ejemplos de codificación

AND AX, BX	;
AND AX, TEMP	;TEMP debe ser una palabra
AND SUM, BX	;SUM debe ser una palabra
AND CL, 0000111b	;Nibble alto en cero
AND AX, TEMP[BX]	;Dirección indirecta

CALL

Transferencia de control

Perform Subroutine

(Ejecutar una subrutina): CALL hace lo siguiente: 1) mete la dirección de desplazamiento de la siguiente instrucción en la pila; 2) si el procedimiento llamado se declara como far (lejano), mete la dirección de segmento de la siguiente instrucción en la pila; 3) carga IP (apuntador a instrucción) con la dirección de desplazamiento del procedimiento llamado; y 4) si el procedimiento llamado se declara como far, carga CS (segmento de código) con la dirección de segmento del procedimiento llamado.

La ejecución continúa entonces en la recién cargada dirección CS:IP hasta que se encuentre RET.

Ejemplos de codificación

CALL WHIZ_BANG	;
CALL [BX]	;Subrutina en dirección en [BX]
CALL AX	;Dirección en AX

-CBW

Aritmética

Convert byte to Word

(Convertir el byte en palabra): CBW convierte el valor de byte en AL a un valor de palabra en AX extendiendo el valor de bit de alto orden de AL a través de todos los bits de AH.

-CLC

Control de banderas y procesador

Clear Carry Flag

(Limpiar la bandera de acarreo): CLC limpia la bandera de acarreo del registro de banderas poniendo la bandera en 0.

Bandera afectada

CF

-CLD

Control de banderas y procesador

Clear Direction Flag

(Limpiar la bandera de dirección): CLD limpia la bandera de dirección del registro de banderas poniendo la bandera en 0.

Bandera afectada

DF

-CLI

Control de banderas y procesador

Clear Interrupt Flag

(Limpiar la bandera de interrupción): CLI limpia la bandera de interrupción del registro de banderas poniendo la bandera en 0. Mientras la bandera de inte-

rrupción esté en cero, la UCP no reconocerá interrupciones enmascarables.

Bandera afectada

IF

-CMC

Control de banderas y procesador

Complement Carry Flag

(Complementar la bandera de acarreo): CMC conmuta la bandera de acarreo del registro de banderas a lo opuesto de la disposición actual de la bandera.

Bandera afectada

CF

-CMP

Aritmética

Compare

(Comparar): CMP se considera una instrucción aritmética debido a que el operando fuente se sustrae del operando de destino. Sin embargo, el resultado se emplea para activar las banderas; no se almacena en ningún lugar. Se puede hacer una prueba posterior de las banderas para el control del programa.

Banderas afectadas

OF, SF, ZF, AF, PF, CF

Ejemplos de codificación

CMP AX, BX ;	
CMP AX, TEMP ;	TEMP debe ser una palabra
CMP SUM, BX ;	SUM debe ser una palabra
CMP CL, 3 ;	Comparar con una constante
CMP AX, TEMP[BX] ;	Dirección indirecta

- CMPSB

Manipulación de cadenas

Compare Strings, Byte-for-Byte

(Comparar cadenas, byte por byte): CMPSB compara cadenas, byte por byte. DI (índice de destino) y SI (índice de fuente) cambian en incrementos o decrementos de 1, dependiendo de la disposición de la bandera de dirección. Normalmente, esta instrucción se usa con las instrucciones REPE, REPNE, REPNZ o REPZ a fin de repetir la comparación para un máximo de CX número bytes. Intel lista este mandato como CMPS, pero diversos ensambladores hacen las distinciones de byte (CMPSB) y palabra (CMPSW). Esta instrucción afecta sólo las banderas; no se hace ningún cambio a los operandos.

Banderas afectadas

OF, SF, ZF, AF, PF, CF

Ejemplos de codificación

CMPSB	:Comparar cadenas
REPE CMPSB	:Repetir

- CMPSW

Manipulación de cadenas

Compare Strings, Word-for-Word

(Comparar cadenas, palabra por palabra): CMPSW compara cadenas, palabra por palabra. DI y SI cambian en incrementos o decrementos de 2, dependiendo de la disposición de la bandera de dirección. Normalmente, esta instrucción se usa junto con las instrucciones REPE, REPNE, REPNZ o REPZ a fin de repetir la comparación para un máximo de CX palabras. Intel lista este mandato como CMPS, pero diversos ensambladores hacen las distinciones de byte (CMPSB) y palabra (CMPSW). Esta instrucción sólo afecta las banderas; no se hace ningún cambio a los operandos.

Banderas afectadas

OF, SF, ZF, AF, PF, CF

Ejemplos de codificación

CMPSW	:Comparar cadenas
REPE CMPSW	:Repetir el ciclo

- CWD

Aritmética

Convert Word to Doubleword

(Convertir una palabra en palabra doble): CWD convierte el valor de palabra en AX a un valor de palabra doble en DX:AX extendiendo el valor de bit de alto orden de AX a través de todos los bits de DX.

- DAA

Aritmética

Decimal Adjust for Addition

(Ajuste decimal para suma): DAA corrige el resultado (AL) de una operación anterior de suma decimal codificada en binario (BCD).

Banderas afectadas

SF, ZF, AF, PF, CF, OF (no definida)

- DAS

Aritmética

Decimal Adjust for Subtraction

(Ajuste decimal para resta): DAS corrige el resultado (AL) de una operación anterior de resta decimal codificada en binario (BCD).

Banderas afectadas

SF, ZF, AF, PF, CF, OF (no definida)

DEC

Aritmética

Decrement

(Disminuir): DEC cambia, en decrementos de 1, el contenido del operando. Se supone que el operando es un valor binario sin signo.

Banderas afectadas

OF, SF, ZF, AF, PF

Ejemplos de codificación

```
DEC AX
DEC SUM
DEC CL
DEC TEMP [SI]
```

DIV

Aritmética

Divide

(Dividir): Si el operando es un valor de byte, DIV divide el contenido de AX entre el contenido del operando y luego almacena el resultado en AL y el residuo en AH. Si el operando es un valor de palabra, DIV divide el contenido de DX:AX entre el contenido del operando y luego almacena el resultado en AX y el residuo en DX. Esta instrucción trata los números como valores binarios sin signo.

Banderas afectadas

OF (no definida), SF (no definida), ZF (no definida), AF (no definida), PF (no definida), CF (no definida)

Ejemplos de codificación

DIV BX	:AX=DX:AX/BX
DIV WRDTMP	:AX=DX:AX/WRDTMP
DIV BYTE_SUM	:AL=AX/BYTE_SUM
DIV WORD_TBL[BX]	:Dirección indirecta

ESC

Control de banderas y procesador

Escape

(Escapar): Esta instrucción proporciona un medio para que los coprocesadores (como el 8087 u otros coprocesadores numéricos) tengan acceso a datos en el flujo de datos del microprocesador. Cuando se encuentra, esta instrucción ocasiona que el microprocesador coloque el operando en el canal de datos y ejecute un NOP internamente.

Ejemplos de codificación

```
ESC 6, TEMP
ESC 15, CL
```

HLT

Control de banderas y procesador

Halt

(Parar): HLT ocasiona que el microprocesador detenga la ejecución y deje los registros CS:IP apuntando a la instrucción que sigue a HLT. Esta condición de parar se termina sólo después de que el sistema recibe una interrupción o se activa la línea de RESET.

IDIV

Aritmética

Integer Divide

(División de enteros): Si el operando es un valor de byte, IDIV divide el contenido de AX entre el contenido del operando; luego almacena el resultado en AL y el residuo en AH. Si el operando es un valor de palabra, IDIV divide el contenido de DX:AX entre el contenido del operando y luego almacena el resultado

en AX y el residuo en DX. Esta instrucción trata los números como valores binarios con signo.

Banderas afectadas

OF (no definida), SF (no definida), ZF (no definida), AF (no definida), PF (no definida), CF (no definida)

Ejemplos de codificación

IDIV BX	;AX=DX:AX/BX
IDIV WRDTMP	;AX=DX:AX/WRDTMP
IDIV BYTE_SUM	;AL=AX/BYTE_SUM
IDIV WORD_TBL[BX]	;Dirección indirecta

-IMUL

Aritmética

Integer Multiply

(Multiplicación de enteros): Si el operando es un valor de byte, IMUL multiplica AL por el contenido del operando y almacena el resultado en AX. Si el operando es un valor de palabra, IMUL multiplica el contenido de AX por el contenido del operando y almacena al resultado en DX:AX. Esta instrucción trata los números como valores binarios con signo.

Banderas afectadas

OF, CF, SF (no definida), ZF (no definida), AF (no definida), PF (no definida)

Ejemplos de codificación

IMUL BX	;DX:AX=AX*BX
IMUL WRDTMP	;DX:AX=AX*WRDTMP
IMUL BYTE_SUM	;AX=AL*BYTE_SUM
IMUL WORD_TBL[BX]	;Dirección indirecta

-IN

Transferencia de datos

Input from Port

(Entrada desde un puerto): IN carga un byte o una palabra de la dirección específica del puerto de E/S de hardware a AL o AX, respectivamente. Un número de puerto inferior a 256 puede especificarse como una constante o una variable en el registro DX, pero un número de puerto superior a 255 debe especificarse en el registro DX.

Ejemplos de codificación

IN AL, 64h
IN AX, DX

→INC

Aritmética

Increment

(Incrementar): INC cambia, mediante incrementos de 1, el contenido del operando. Se supone que el operando es un valor binario sin signo.

Banderas afectadas

OF, SF, ZF, AF, PF

Ejemplos de codificación

INC AX
INC SUM
INC CL
INC TEMP[SI]

-INT

Transferencia de control

Software Interrupt

(Interrupción por software): INT inicia una interrupción de software de la UCP y hace lo siguiente: 1) mete las banderas en la pila; 2) limpia las

banderas TF e IF; 3) mete el valor de CS en la pila; 4) carga CS con la dirección de segmento de la interrupción invocada (que se encontró en la dirección calculada en la tabla vectorial de interrupciones); 5) mete el valor de IP en la pila; y 6) carga IP con la dirección de desplazamiento de la interrupción invocada (que se encontró en la dirección calculada en la tabla vectorial de interrupciones).

Después, la ejecución continúa en la dirección CS:IP recién cargada hasta que se encuentre una instrucción IRET.

Banderas afectadas

IF, TF

Ejemplos de codificación

INT 10h

INT 13h

- INTO

Transferencia de control

Interrupt on Overflow

(Interrumpir en sobreflujo): Si la bandera de sobreflujo (OF) está activada, INTO ejecuta una interrupción 4 y el control procede como si se hubiera emitido una INT 4. Hay que tener en cuenta que, en este caso, el registro de banderas se ve afectado como se describe para la instrucción INT.

- IRET

Transferencia de control

Return from Interrupt

(Regresar de una interrupción): IRET ocasiona la terminación de un procedimiento de interrupción y —sacando los valores de IP, CS y el registro de banderas de la pila— regresa el control al punto en el que ocurrió la interrupción.

Banderas afectadas

OF, DF, IF, TF, SF, ZF, AF, PF, CF (NT e IOPL para el 80286)

- JA

Transferencia de control

Jump if Above

(Saltar si es arriba): JA ocasiona que la ejecución de un programa se ramifique hacia la dirección del operando si tanto la bandera de acarreo como la de cero están limpias. Esta instrucción es funcionalmente igual que JNBE.

Ejemplo de codificación

JA NEXT_STEP

(JA SIGUIENTE_PASO)

- JAE

Transferencia de control

Jump if Above or Equal

(Saltar si es arriba o igual): JAE ocasiona que la ejecución de un programa se ramifique hacia la dirección del operando si la bandera de acarreo está limpia. Esta instrucción es funcionalmente igual que JNB o JNC.

Ejemplo de codificación

JAE NEXT_STEP

(JAE SIGUIENTE_PASO)

-JB

Transferencia de control

Jump if Below

(Saltar si es abajo): JB ocasiona que la ejecución de un programa se ramifique hacia la dirección del operando si la bandera de acarreo está activada. Esta instrucción es funcionalmente igual que JNAE.

Ejemplo de codificación

JB _NEXT_STEP
(JB SIGUIENTE_PASO)

-JBE

Transferencia de control

Jump if Below or Equal

(Saltar si es abajo o igual): JBE ocasiona que la ejecución de un programa se ramifique hacia la dirección del operando si está activada la bandera de acarreo o la de cero. Esta instrucción es funcionalmente igual que JNA.

Ejemplo de codificación

JBE _NEXT_STEP
(JBE SIGUIENTE_PASO)

-JC

Transferencia de control

Jump on Carry

(Saltar en acarreo): JC ocasiona que la ejecución de un programa se ramifique hacia la dirección del operando si la bandera de acarreo está activada. Esta instrucción es funcionalmente igual que JB o JNAE.

Ejemplo de codificación

JC _NEXT_STEP
(JC SIGUIENTE_PASO)

-JCXZ

Transferencia de control

Jump if CX=0

(Saltar si CX=0): JCXZ ocasiona que la ejecución de un programa se ramifique hacia la dirección del operando si el valor de CX es 0.

Ejemplo de codificación

JCXZ _SKIP_LOOP
(JCXZ SALTAR_CICLO)

-JE

Transferencia de control

Jump if Equal

(Saltar si es igual): JE ocasiona que la ejecución de un programa se ramifique hacia la dirección del operando si la bandera de cero está activada. Esta instrucción es funcionalmente igual que JZ.

Ejemplo de codificación

JE _NEXT_STEP
(JE SIGUIENTE_PASO)

-JG

Transferencia de control

Jump if Greater

(Saltar si es mayor): JG ocasiona que la ejecución de un programa se ramifique hacia la dirección del operando si la bandera de signo es igual a la de sobreflujo o si la bandera de cero está limpia. Esta instrucción es funcionalmente igual que JNL.E.

Ejemplo de codificación

JG NEXT_STEP
(JG SIGUIENTE_PASO)

-JGE

Transferencia de control

Jump if Greater or Equal

(Saltar si es mayor o igual): JGE ocasiona que la ejecución de un programa se ramifique hacia la dirección del operando si la bandera de signo es igual a la de sobreflujo. Esta instrucción es funcionalmente igual que JNL.

Ejemplo de codificación

JGE NEXT_STEP
(JGE SIGUIENTE_PASO)

-JL

Transferencia de control

Jump if Less Than

(Saltar si es menor que): JL ocasiona que la ejecución de un programa se ramifique hacia la dirección del operando si la bandera de signo no es igual a la de sobreflujo. Esta instrucción es funcionalmente igual que JNGE.

Ejemplo de codificación

JL NEXT_STEP
(JL SIGUIENTE_PASO)

-JLE

Transferencia de control

Jump if Less Than or Equal

(Saltar si es menor que o igual a): JLE ocasiona que la ejecución de un programa se ramifique hacia la dirección del operando si la bandera de signo no es igual a la de sobreflujo o si la bandera de cero está activada. Esta instrucción es funcionalmente igual que JNG.

Ejemplo de codificación

JLE NEXT_STEP
(JLE SIGUIENTE_PASO)

-JMP

Transferencia de control

Jump

(Saltar): JMP ocasiona que la ejecución de un programa comience en la dirección del operando designado. JMP afecta los registros CS e IP según sea necesario para producir esta ramificación incondicional.

Ejemplos de codificación

JMP EXIT_CODE	
(JMP CODIGO_SALIDA)	
JMP [BX]	: Dirección en [BX]
JMP AX	: Dirección en AX

-JNA

Transferencia de control

Jump if Not Above

(Saltar si no es arriba): JNA ocasiona que la ejecución de un programa se ramifique hacia la dirección del operando si está activada la bandera de acarreo o la de

cero. Esta instrucción es funcionalmente igual que JBE.

Ejemplo de codificación

JNA (JNA SIGUIENTE_PASO)

- JNAE

Transferencia de control

Jump if Not Above or Equal

(Saltar si no es arriba o igual): JNAE ocasiona que la ejecución de un programa se ramifique hacia la dirección del operando si tanto la bandera de acarreo como la de cero están limpias. Esta instrucción es funcionalmente igual que JA o JC.

Ejemplo de codificación

JNAE (JNAE SIGUIENTE_PASO)

- JNB

Transferencia de control

Jump if not Below

(Saltar si no es abajo): JNB ocasiona que la ejecución de un programa se ramifique hacia la dirección del operando si la bandera de acarreo está en cero. Esta instrucción es funcionalmente igual que JAE o JNC.

Ejemplo de codificación

JNB (JNB SIGUIENTE_PASO)

- JNBE

Transferencia de control

Jump if Not Below or Equal

(Saltar si no es abajo o igual): JNBE ocasiona que la ejecución de un programa se ramifique hacia la dirección del operando si tanto la bandera de acarreo como la de cero están limpias. Esta instrucción es funcionalmente igual que JA.

Ejemplo de codificaciónb

JNBE (JNBE SIGUIENTE_PASO)

- JNC

Transferencia de control

Jump on No Carry

(Saltar en no acarreo): JNC ocasiona que la ejecución de un programa se ramifique hacia la dirección del operando si la bandera de carro está limpia. Esta instrucción es funcionalmente igual que JAE o JNB.

Ejemplo de codificación

JNC (JNC SIGUIENTE_PASO)

- JNE

Transferencia de control

Jump if Not Equal

(Saltar si no es igual): JNE ocasiona que la ejecución de un programa se ramifique hacia la dirección del operando si la bandera de cero está limpia. Esta instrucción es funcionalmente igual que JNZ.

Ejemplo de codificación

JNE NEXT_STEP

(JNE SIGUIENTE_PASO)

- JNG

Transferencia de control

Jump if Not Greater Than

(Saltar si no es mayor que): JNG ocasiona que la ejecución del programa se ramifique hacia la dirección del operando si la bandera de signo no es igual a la de sobreflujo o si la bandera de cero está activada. Esta instrucción es funcionalmente igual que JLE.

Ejemplo de codificación

JNG NEXT_STEP

(JNG SIGUIENTE_PASO)

- JNGE

Transferencia de control

Jump if Not Greater Than or Equal

(Saltar si no es mayor que o igual a): JNGE ocasiona que la ejecución del programa se ramifique hacia la dirección del operando si la bandera de signo no es igual a la de sobreflujo. Esta instrucción es funcionalmente igual que JL.

Ejemplo de codificación

JNGE NEXT_STEP

(JNGE SIGUIENTE_PASO)

- JNL

Transferencia de control

Jump if Not Less Than

(Saltar si no es menor que): JNL ocasiona que la ejecución de un programa se ramifique hacia la dirección del operando si la bandera de signo es igual a la de sobreflujo. Esta instrucción es funcionalmente igual que JGE.

Ejemplo de codificación

JNL NEXT_STEP

(JNL SIGUIENTE_PASO)

- JNLE

Transferencia de control

Jump if Not Less Than or Equal

(Saltar si no es menor que o igual a): JNLE ocasiona que la ejecución de un programa se ramifique hacia la dirección del operando si la bandera de signo es igual a la de sobreflujo o si la bandera de cero está limpia. Esta instrucción es funcionalmente igual que JG.

Ejemplo de codificación

JNLE NEXT_STEP

(JNLE SIGUIENTE_PASO)

- JNO

Transferencia de control

Jump on No Overflow

(Saltar en no sobreflujo): JNO ocasiona que la ejecución de un programa se ramifique hacia la dirección del operando si la bandera de sobreflujo está limpia.

Ejemplo de codificación

JNO NEXT_STEP

(JNO SIGUIENTE_PASO)

-JNP

Transferencia de control

Jump on No Parity

(Saltar en no paridad): JNP ocasiona que la ejecución de un programa se ramifique hacia la dirección del operando si la bandera de paridad está limpia. Esta instrucción es funcionalmente igual que JPO.

Ejemplo de codificación

JNP NEXT_STEP

(JNP SIGUIENTE_PASO)

-JNS

Transferencia de control

Jump on Not Sign

(Saltar en no signo): JNS ocasiona que la ejecución de un programa se ramifique hacia la dirección del operando si la bandera de signo está limpia.

Ejemplo de codificación

JNS NEXT_STEP

(JNS SIGUIENTE_PASO)

-JNZ

Transferencia de control

Jump on Not Zero

(Saltar en no cero): JNZ ocasiona que la ejecución de un programa se ramifique hacia la dirección del operando si la bandera de cero está limpia. Esta instrucción es funcionalmente igual que JNE.

Ejemplo de codificación

JNZ NEXT_STEP

(JNZ SIGUIENTE_PASO)

-JO

Transferencia de control

Jump on Overflow

(Saltar en sobreflujo): JO ocasiona que la ejecución de un programa se ramifique hacia la dirección del operando si la bandera de sobreflujo está activada.

Ejemplo de codificación

JO NEXT_STEP

(JO SIGUIENTE_PASO)

-JP

Transferencia de control

Jump on Parity

(Saltar en paridad): JP ocasiona que la ejecución de un programa se ramifique hacia la dirección del operando si la bandera de paridad está activada. Esta instrucción es funcionalmente igual que JPE.

Ejemplo de codificación

JP NEXT_STEP

(JP SIGUIENTE_PASO)

-JPE

Transferencia de control

Jump on Parity Even

(Saltar en paridad par): JPE ocasiona que la ejecución de un programa se ramifique hacia la dirección del operando si la bandera de paridad está activada. Esta instrucción es funcionalmente igual que JP.

Ejemplo de codificación

JPE NEXT STEP

(JPE SIGUIENTE PASO)

-JPO

Transferencia de control

Jump on Parity Odd

(Saltar en paridad impar): JPO ocasiona que la ejecución de un programa se ramifique hacia la dirección del operando si la bandera de paridad está limpia. Esta instrucción es funcionalmente igual que JNP.

Ejemplo de codificación

JPO NEXT STEP

(JPO SIGUIENTE PASO)

-JS

Transferencia de control

Jump on Sign

(Saltar en signo): JS ocasiona que la ejecución de un programa se ramifique hacia la dirección del operando si la bandera de signo está activada.

Ejemplo de codificación

JE NEXT STEP

(JS SIGUIENTE PASO)

-JZ

Transferencia de control

Jump on Zero

(Saltar en cero): JZ ocasiona que la ejecución de un programa se ramifique hacia la dirección del operando si la bandera de cero está activada. Esta instrucción es funcionalmente igual que JE.

Ejemplo de codificación

JZ NEXT STEP

(JZ SIGUIENTE PASO)

-LAHF

Transferencia de datos

Load AH Register with Flags

(Cargar el registro AH con banderas): LAHF copia el byte de orden inferior del registro de banderas en AH. Después de la ejecución de esta instrucción, los bits 7, 6, 4, 2 y 1 de AH son iguales a SF, ZF, AF, PF y CF, respectivamente.

-LDS

Transferencia de datos

Load DS Register

(Cargar el registro DS (segmento de datos)): LDS realiza dos operaciones distintas: carga DS con la dirección de segmento del operando fuente, y carga el operando de destino con la dirección de desplazamiento del operando fuente.

Ejemplo de codificación

LDS SI, SOURCE_BUFFER

(LDS SI, BUFFER_FUENTE)

-LEA

Transferencia de datos

Load Effective Address

(Cargar la dirección efectiva): LEA transfiere la dirección de desplazamiento del operando fuente al operando de destino. El operando de destino debe ser un registro general de palabras.

Ejemplo de codificación

```
LEA AX, MESSAGE_1
(LEA AX, MENSAJE_1)
```

-LES

Transferencia de datos

Load ES Register

(Cargar el registro ES (segmento extra)): LES realiza dos operaciones distintas: carga ES con la dirección de segmento del operando fuente, y carga el operando de destino con la dirección de desplazamiento del operando fuente.

Ejemplo de codificación

```
LES DI, DEST_BUFFER
(LES DI, BUFFER_DEST)
```

-LOCK

Control de banderas y procesador

Lock Bus

(Bloquear el canal): LOCK prohíbe la interferencia desde otros coprocesadores durante la ejecución de la siguiente instrucción emitida. LOCK es un prefijo que se debe usar con otras operaciones.

Ejemplo de codificación

```
LOCK XLAT
```

-LODSB

Manipulación de cadenas

Load a Byte from String into AL

(Cargar un byte de cadena en AL): Esta instrucción carga AL con el contenido de la dirección apuntada por

SI. Después SI cambia en incrementos o decrementos de 1, dependiendo de la activación de la bandera de dirección. Intel lista este mandato como LODS; sin embargo, varios ensambladores pueden hacer la distinción entre byte (LODSB) y palabra (LODSW).

-LODSW

Manipulación de cadenas

Load a Word from String into AX

(Cargar una palabra de cadena en AX): Esta instrucción carga AX con el contenido de la dirección apuntada por SI. Después SI cambia en incrementos o decrementos de 2, dependiendo de la activación de la bandera de dirección. Intel lista este mandato como LODS; sin embargo, varios ensambladores pueden hacer la distinción entre byte (LODSB) y palabra (LODSW).

-LOOP

Transferencia de control

Loop

(Ciclo): Basándose en el contenido de CX, la ejecución del programa se ramifica hacia la dirección del operando de destino. Si CX no es igual a 0, CX cambia en decrementos de 1 y ocurre la ramificación. Si CX es 0, no ocurre decremento o ramificación y la ejecución continúa en la siguiente instrucción.

Ejemplo de codificación

```
LOOP PRINT_LOOP
(LOOP IMPRIMIR_CICLO)
```

-LOOP

Transferencia de control

Loop While Equal

(Ciclo mientras es igual): Basándose en el contenido de CX y en la bandera de cero, la ejecución del programa se ramifica hacia la dirección del operando de destino. Si CX no es igual a 0 y la bandera de cero está activada, CX cambia en decrementos de 1 y ocurre la ramificación. Si CX es 0 o si la bandera de cero está limpia, no ocurre decremento o ramificación, y la ejecución continúa en la siguiente instrucción. Esta instrucción es funcionalmente equivalente a LOOPZ.

Ejemplo de codificación

```
LOOPE TEST_LOOP
(LOOPE CICLO_PRUEBA)
```

-LOOPNZ

Transferencia de control

Loop While Not Zero

(Ciclo mientras no es cero): Basándose en el contenido de CX y en la bandera de cero, la ejecución del programa se ramifica hacia la dirección del operando de destino. Si CX no es igual a 0 y la bandera de cero está limpia, CX cambia en decrementos de 1 y ocurre la ramificación. Si CX es 0 o si la bandera de cero está activada, no ocurre decremento o ramificación, y la ejecución continúa con la siguiente instrucción. Esta instrucción es funcionalmente equivalente a LOOPNE.

Ejemplo de codificación

```
LOOPNZ TEST_LOOP
(LOOPNZ CICLO_PRUEBA)
```

-LOOPNE

Transferencia de control

Loop While Not Equal

(Ciclo mientras no es igual): Basándose en el contenido de CX y en la bandera de cero, la ejecución del programa se ramifica hacia la dirección del operando de destino. Si CX no es igual a 0 y la bandera de cero está limpia, CX cambia en decrementos de 1 y ocurre la ramificación. Si CX es 0 o la bandera de cero está activada, no ocurre decremento o ramificación, y la ejecución continúa en la siguiente instrucción. Esta instrucción es funcionalmente equivalente a LOOPNZ.

Ejemplo de codificación

```
LOOPNE TEST_LOOP
(LOOPNE CICLO_PRUEBA)
```

-LOOPZ

Transferencia de control

Loop While Zero

(Ciclo mientras es cero): Basándose en el contenido de CX y en la bandera de cero, la ejecución del programa se ramifica hacia la dirección del operando de destino. Si CX no es igual a 0 y la bandera de 0 está activada, CX cambia en decrementos de 1 y ocurre la ramificación. Si CX es 0 o la bandera de cero está limpia, no ocurre decremento o ramificación, y la ejecución continúa con la siguiente instrucción. Esta instrucción es funcionalmente equivalente a LOOPE.

Ejemplo de codificación

```
LOOPZ TEST_LOOP
(LOOPZ CICLO_PRUEBA)
```

-MOV

Move

Transferencia de datos

(Transferir): MOV copia el contenido del operando fuente en el operando de destino. Ambos operandos deben tener la misma longitud.

Ejemplos de codificación

```
MOV AX, BX      ;AX=BX
MOV AX, WRDTMP ;AX=WRDTMP
MOV WRDSUM, BX ;WRDSUM=BX
MOV CL, 57      ;CL=57
MOV DEC, 10     ;DEC=10
MOV AX, TEMP [BX] ;Dirección indirecta
```

-MOVSB

Manipulación de cadenas

Move String, Byte-by-Byte

(Transferir una cadena, byte por byte): MOVSB transfiere cadenas, byte por byte. Los valores de SI y DI cambian en incrementos o decrementos de 1, dependiendo de la activación de la bandera de dirección. Normalmente, esta instrucción se usa con la instrucción REP a fin de repetir la transferencia para un máximo de bytes de CX. Intel lista este mandato como MOVS; sin embargo, varios ensambladores hacen la distinción entre byte y palabra.

Ejemplos de codificación

```
MOVSB
REP MOVSB ;Repetir un ciclo de transferencia
```

-MOVSW

Manipulación de cadenas

Move String, Word-by-Word

(Transferir una cadena, palabra por palabra): MOVSW transfiere cadenas, palabra por palabra. Los valores de SI y DI cambian en incrementos o decrementos de 2, dependiendo de la activación de la bandera de dirección. Normalmente, esta instrucción se usa con la instrucción REP a fin de repetir la transferencia para un máximo de palabras de CX. Intel lista este mandato como MOVS; sin embargo, varios ensambladores hacen la distinción entre byte y palabra.

Ejemplos de codificación

```
MOVSW
REP MOVSW ;Repetir un ciclo de transferencia
```

-MUL

Aritmética

Multiply

(Multiplicar): Si el operando es un valor de byte, MUL multiplica el contenido de AL por el contenido del operando y almacena al resultado en AX. Si el operando es un valor de palabra, MUL multiplica el contenido de AX por el contenido del operando y almacena el resultado en DX:AX. Esta instrucción trata los números como valores binarios.

Banderas afectadas

OF, CF, SF (no definida), ZF (no definida), AF (no definida), PF (no definida)

Ejemplos de codificación

```
MUL BX          ;DX:AX=AX*BX
MUL WORD_TEMP  ;DX:AX=
                AX*WORD_TEMP
MUL BYTE_SUM    ;AX=AL*BYTE_SUM
MUL WORD_TBL [BX] ;Dirección indirecta
```

-NEG

Aritmética

Negate

(Negar): NEG calcula el complemento a dos del operando de destino y almacena el resultado en ese operando. Este cálculo es efectivamente igual que restar el operando de destino de 0.

Banderas afectadas

OF, SF, ZF, AF, PF, CF

Ejemplos de codificación

NEG TEMP

NEG CL

-NOP

Control de banderas y procesador

No Operation

(Ninguna operación): NOP simplemente toma espacio y tiempo y ocasiona que la UCP no haga nada.

-NOT

Manipulación de bits

Logical NOT on Bits

(NO lógico en bits): NOT invierte los bits en el operando de destino (0 se convierte en 1, y 1 se convierte en 0) y almacena los bits invertidos en el operando de destino.

Ejemplos de codificación

NOT CL

NOT TEMP

NOT AX

-OR

Manipulación de bits

Logical OR on Bits

(O lógico en bits): Esta instrucción realiza un O lógico de los operandos y almacena el resultado en el operando de destino. Cada bit del byte o palabra resultante se pone en 1 si uno o ambos bits correspondientes a cada operando se ponen en 1.

Banderas afectadas

OF, SF, ZF, PF, CF, AF (no definida)

Ejemplos de codificación

OR AL, BL

OR AL, 10000000b

OR DX, TEMP

OR AX, CX

-OUT

Transferencia de datos

Output to Port

(Salir a puerto): OUT envía un byte (AL) o palabra (AX) a la dirección de puerto de E/S de hardware especificada. Un número de puerto menor que 256 puede especificarse como una constante o como una variable en el registro DX. Sin embargo, un número de puerto mayor que 255 debe especificarse en el registro DX.

Ejemplos de codificación

OUT AL, 64h

OUT AX, DX

-POP

Transferencia de datos

Remove Data from Stack

(Quitar datos de la pila): POP quita una palabra de la pila y la sitúa en el operando de destino deseado.

Ejemplos de codificación

POP AX
 POP DS
 POP HOLD_REG
 (POP GUARD_REG)

-POPF

Transferencia de datos

Remove Flags from Stack

(Quitar banderas de la pila): POPF quita una palabra de la pila y la sitúa en el registro de banderas.

Banderas afectadas

OF, DF, IF, TF, SF, ZF, AF, PF, CF (NT e IOPL para el 80286)

-PUSH

Transferencia de datos

Place Data on Stack

(Colocar datos en la pila): PUSH coloca en la pila una copia del valor del operando.

Ejemplos de codificación

PUSH AX
 PUSH DS
 PUSH HOLD_REG
 (PUSH GUARD_REG)

-PUSHF

Transferencia de datos

Place Flags on stack

(Colocar banderas en la pila): PUSHF coloca en la pila una copia del registro de banderas.

-RCL

Manipulación de bits

Rotate Left through Carry

(Rotar a la izquierda a lo largo del acarreo): RCL rota todos los bits del operando de destino a la izquierda tantos lugares como indique el operando fuente. La rotación se realiza a través de la bandera de acarreo en un orden que rota el bit más significativo del operando de destino a la bandera de acarreo, y ésta al bit menos significativo del operando de destino.

Banderas afectadas

OF, CF

Ejemplos de codificación

RCL AX, 1
 RCL BL, 3
 RCL TEMP, CL

-RCR

Manipulación de bits

Rotate Right through Carry

(Rotar a la derecha a lo largo del acarreo): RCR rota todos los bits del operando de destino a la derecha tantos lugares como indique el operando fuente. La rotación se realiza a través de la bandera de acarreo en un orden que rota el bit menos significativo del operando de destino a la bandera de acarreo, y ésta al bit más significativo del operando de destino.

Banderas afectadas
OF, CF

Ejemplos de codificación

RCR AX, 1
RCR BL, 3
RCR TEMP, CL

-REP

Manipulación de cadenas

Repeat

(Repetir): REP ocasiona que las instrucciones de manipulación de cadenas se repitan tantas veces como se indique en CX.

Ejemplo de codificación

REP MOVSB

-REPE

Manipulación de cadenas

Repeat if Equal

(Repetir si es igual): REPE ocasiona que las instrucciones de manipulación de cadenas se repitan tantas veces como se indique en CX. Cuando se usa con CMPSB, CMPSW, SCASB o SCASW, esta instrucción causa la repetición sólo mientras la bandera de cero está activada. Esta instrucción es funcionalmente equivalente a REPZ.

Ejemplo de codificación

REPE CMPSW

-REPNE

Manipulación de cadenas

Repeat if not Equal

(Repetir si no es igual): REPNE ocasiona que las instrucciones de manipulación de cadenas se repitan el número de iteraciones específicas en CX. Cuando se usa con CMPSB, CMPSW, SCASB o SCASW esta instrucción causa la repetición sólo mientras la bandera de cero vale cero. Esta instrucción es funcionalmente equivalente a REPNZ.

Ejemplo de codificación

REPNE CMPSW

-REPNZ

Manipulación de cadenas

Repeat if Not Zero

(Repetir si no es cero): REPNZ ocasiona que instrucciones de manipulación de cadenas se repitan el número de iteraciones especificadas en CX. Cuando se usa con CMPSB, CMPSW, SCASB o SCASW, esta instrucción causa la repetición sólo cuando la bandera de cero está limpia. Esta instrucción es funcionalmente equivalente a REPNE.

Ejemplo de codificación

REPNZ CMPSW

-REPZ

Manipulación de cadenas

Repeat if Zero

(Repetir si es cero): REPZ ocasiona que instrucciones de manipulación de cadenas se repitan tantas veces como se indique en CX. Cuando se usa con CMPSB, CMPSW, SCASB o SCASW, esta instrucción causa la repetición sólo cuando la bandera de cero está

activada. Esta instrucción es funcionalmente equivalente a REPE.

Ejemplo de codificación
REPZ CMPSW

- RET

Transferencia de control

Return from Subroutine

(Volver de la subrutina): Al sacar a IP de la pila, RET transfiere el control del programa de vuelta al punto en el que se emitió un CALL. Si CALL fuera una llamada a un procedimiento *far*, tanto CS como IP serían sacados de la pila. Si RET tiene un valor de retorno especificado (2, en el ejemplo de codificación), la pila se ajusta según ese número de bytes. En el ejemplo de codificación se muestra que una palabra se descarta de la pila después de que IP o CS:IP hayan sido sacados.

Ejemplos de codificación

RET
RET 2

- ROL

Manipulación de bits

Rotate Left

(Rotar a la izquierda): ROL rota todos los bits del operando de destino a la izquierda tantos lugares como indique el operando fuente.

Banderas afectadas
OF, CF

Ejemplos de codificación

ROL AX, 1
ROL BL, 3
ROL TEMP, CL

- ROR

Manipulación de bits

Rotate Right

(Rotar a la derecha): ROR rota todos los bits del operando de destino a la derecha tantos lugares como indique el operando fuente.

Banderas afectadas

OF, CF

Ejemplos de codificación

RR AX, 1
RR BL, 3
RR TEMP, CL

- SAHF

Transferencia de datos

Store AH into Flag Register

(Almacenar AH en el registro de banderas): SAHF copia el contenido de AH en el byte de orden inferior del registro de banderas. Después de la ejecución de esta instrucción, SF, ZF, AF, PF y CF son iguales a los bits 7, 6, 4, 2 y 1 de AH, respectivamente.

Banderas afectadas
SF, ZF, AF, PF, CF

- SAL

Manipulación de bits

Arithmetic Shift Left

(Desplazamiento aritmético a la izquierda): SAL desplaza todos los bits del operando de destino a la izquierda tantos lugares como indique el operando fuente. Los bits de alto orden se pierden, mientras que los de orden inferior se limpian.

Banderas afectadas

OF, SF, ZF, PF, CF, AF (no definida)

Ejemplos de codificación

SAL AX, 1

SAL BL, 3

SAL TEMP, CL

- SAR

Manipulación de bits

Arithmetic Shift Right

(Desplazamiento aritmético a la derecha): SAR desplaza todos los bits del operando de destino a la derecha tantos lugares como indique el operando fuente. Los bits de orden inferior se pierden mientras que los de alto orden adquieren el valor del bit de alto orden existente.

Banderas afectadas

OF, SF, ZF, PF, CF, AF (no definida)

Ejemplos de codificación

SAR AX, 1

SAR BL, 3

SAR TEMP, CL

- SBB

Aritmética

Subtract with Carry

(Restar con acarreo): SBB resta el contenido del operando fuente del operando de destino (y almacena el resultado en éste). Si la bandera de acarreo está activada, el resultado cambia en decrementos de 1. En esta instrucción, se supone que los valores añadidos son binarios.

Banderas afectadas

OF, SF, ZF, AF, PF, CF

Ejemplos de codificación

SBB AX, BX ;AX=AX-AX-CF

SBB AX, TEMP ;AX=AX-TEMP-CF

SBB SUM, BX ;SUM=SUM-BX-CF

SBB CL, 10 ;CL=CL-10-CF

SBB AX, TEMP[BX] ;Dirección indirecta

- SCASB

Manipulación de cadenas

Scan String for Byte

(Examinar una cadena por byte): SCASB resta el byte de cadena del operando de destino (apuntado por DI) del valor de AL. No se almacena el resultado pero se actualizan las banderas. Entonces el valor de DI cambia en incrementos o decrementos de 1, dependiendo de la activación de la bandera de dirección.

Normalmente, esta instrucción se usa con las instrucciones REPE, REPNE, REPNZ o REPZ para repetir el examen un máximo de CX bytes, o hasta que SCASB encuentre una concordancia o una diferencia. Intel lista este mandato como SCAS; sin embargo, varios ensambladores hacen las distinciones entre byte y palabra.

Banderas afectadas

OF, SF, ZF, AF, PF, CF

Ejemplos de codificación

SCASB

REPZ SCASB ;Repetir un ciclo de examen

-SCASW

Manipulación de cadenas

Scan String for Word

(Examinar una cadena por palabra): SCASW resta la palabra de cadena del operando de destino (apuntado por DI) del valor de AX. No se almacena el resultado pero se actualizan las banderas. Entonces el valor de DI cambia en incrementos o decrementos de 2, dependiendo de la activación de la bandera de dirección. Normalmente, esta instrucción se usa con las instrucciones REPE, REPNE, REPNZ o REPZ para repetir el examen un máximo de CX bytes, o hasta que SCASW encuentre una concordancia o una diferencia. Intel lista este mandato como SCAS; sin embargo, varios ensambladores hacen las distinciones entre byte y palabra.

Banderas afectadas

OF, SF, ZF, AF, PF, CF

Ejemplos de codificación

SCASW

REPZ SCASW ;Repetir un ciclo de examen

-SHR

Manipulación de bits

Shift Right

(Desplazar a la derecha): SHR desplaza todos los bits del operando de destino a la derecha tantos lugares como indique el operando fuente. Los bits de orden inferior se pierden y los de alto orden se limpian.

Banderas afectadas

OF, SF, ZF, PF, CF, AF (no definida)

Ejemplos de codificación

SHR AX, 1

SHR BL, 3

SHR TEMP, CL

-STC

Control de banderas y procesador

Set Carry Flag

(Activar la bandera de acarreo): STC activa la bandera de acarreo sin importar la condición presente de ésta.

Bandera afectada

CF

-STD

Control de banderas y procesador

Set Direction Flag

(Activar la bandera de dirección): STD activa la bandera de dirección sin importar la condición presente de ésta. Tal activación afecta las instrucciones de cadenas.

Bandera afectada

DF

-SHL

Manipulación de bits

Shift Left

(Desplazar a la izquierda): SHL desplaza todos los bits del operando de destino a la izquierda tantos lugares como indique el operando fuente. Los bits de alto orden se pierden y los de orden inferior se limpian.

Banderas afectadas

OF, SF, ZF, PF, CF, AF (no definida)

Ejemplos de codificación

SHL AX, 1

SHL BL, 3

SHL TEMP, CL

- STI

Control de banderas y procesador

Set Interrupt Flag

(Activar la bandera de interrupción): STI activa la bandera de interrupción sin importar la condición presente de ésta. Mientras se activa dicha bandera, la UCP responde a interrupciones enmascarables.

Bandera afectada

IF

- SUB

Aritmética

Subtract

(Restar): SUB resta el contenido del operando fuente del operando de destino (y almacena el resultado en éste). En esta instrucción, se supone que los valores añadidos son binarios.

Banderas afectadas

OF, SF, ZF, AF, PF, CF

Ejemplos de codificación

SUB AX, BX	;AX=AX-BX
SUB AX, TEMP	;AX=AX-TEMP
SUB SUM, BS	;SUM=SUM-BX
SUB CL, 10	;CL=CL-10
SUB AX, TEMP [BX]	;Dirección indirecta

- STOSB

Manipulación de cadenas

Store Byte in AL at String

(Almacenar el byte de AL en la cadena): Esta instrucción copia el contenido de AL en la dirección de byte apuntada por DI. Entonces DI cambia en incrementos o decrementos de 1, dependiendo de la activación de la bandera de dirección. Intel lista este mandato como STOS; sin embargo, varios ensambladores hacen las distinciones entre byte y palabra.

- STOSW

Manipulación de cadenas

Store Word in AX at String

(Almacenar palabra de AX en la cadena): Esta instrucción copia el contenido en AX en la dirección de palabras apuntada por DI. Entonces DI cambia en incrementos o decrementos de 2, dependiendo de la activación de la bandera de dirección. Intel lista este mandato como STOS; sin embargo, varios ensambladores hacen las distinciones entre byte y palabra.

- TEST

Manipulación de bits

Test Bits

(Bits de prueba): TEST realiza un Y lógico de los operandos, pero no se almacena el resultado. Sólo las banderas se ven afectadas. Cada bit del byte o palabra resultante se pone en 1 sólo si el bit correspondiente de cada operando es 1.

Banderas afectadas

OF, SF, ZF, PF, CF, AF (no definida)

Ejemplos de codificación

TEST AX, BX	;
TEST AX, TEMP	;TEMP debe ser una palabra
TEST SUM, BX	;SUM debe ser una palabra
TEST CL, 00001111b	;
TEST AX, TEMP [BX]	;Dirección indirecta

-WAIT

Control de banderas y procesador

Wait

(Esperar): WAIT ocasiona que la UCP espere una interrupción externa en la línea de TEST antes de continuar.

-XCHG

Transferencia de datos

Exchange

(Intercambiar): XCHG intercambia el contenido de los operandos fuente y de destino.

Ejemplos de codificación

XCHG AX,BX ; Intercambiar AX con BX
 XCHG CL, CH ; Intercambiar CL con CH
 XCHG AC, TEMP ; Intercambiar AX con TEMP

-XLAT

Transferencia de datos

Translate

(Traducir): Suponiendo que la dirección de desplazamiento de una tabla de traducción de 256 bytes está contenida en BX, esta instrucción usa el valor en AL como un desplazamiento basado en cero dentro de la tabla, y posteriormente carga AL con el valor de byte en el desplazamiento calculado. Esta instrucción es útil para tablas de traducción.

=XOR

Manipulación de bits

Logical Exclusive-Or on bits

(O exclusivo lógico sobre bits): Esta instrucción realiza un XOR lógico de los operandos y almacena el resultado en el operando de destino. Cada bit del byte o palabra resultante se pone en 1 sólo si el bit correspondiente de cada operando contiene valores opuestos.

Banderas afectadas

OF, SF, ZF, PF, CF, AF (no definida)

Ejemplos de codificación

XOR AX, BX	:	
XOR AX, TEMP	:	; TEMP debe ser una palabra
XOR SUM, BX	:	; SUM debe ser una palabra
XOR CL, 0001111b	:	
XOR AX, TEMP[BX]	:	Dirección indirecta