Universidad Estatal de Bolívar

CONJUNTO DE INSTRUCCIONES DE LA FAMILIA X86 DE INTEL®

ASSEMBLER / ENSAMBLADOR

Compilación de las Instrucciones Del Lenguaje Ensamblador Compatible con microprocesadores De la Familia x86 de Intel

Material Compilado para la esnseñanza del Lenguake Assembler

Compil. MSc. Henry Vallejo

CONJUNTO DE INSTRUCCIONES DE LA FAMILIA X86 DE INTEL

Las instrucciones presentadas en esta sección funcionan como se describe en el 80286 y el 80386.

ARPL

Manipulación de bits

Adjust RPL field of Selector

(Ajustar el campo RPL del selector): ARPL compara los bits RPL (bits 0 y 1) del primer operando con los del segundo. Si los bits RPL del primer operando son menos que los del segundo, los dos bits del primer operando se ponen igual que los dos bits del segundo, y se activa también la bandera de cero. De lo contrario, la bandera de cero se limpia. Esta instrucción se usa sólo en software de sistemas operativos (no se usa en software de aplicaciones).

Bandera – afectada

ZF
Ejemplos de codificación
ARPL SELECTOR,AX
ARPL AX,CX

BOUND

Control de banderas y procesador

Check Array Index against Bounds

(Verificar el índice de arreglos contra límites): BOUND determina si el valor con signo del primer operando cae entre los dos límites especificados por el segundo operando. Se supone que la palabra del segundo operando es el limite inferior, y la siguiente palabra el límite superior. Ocurre una interrupción 5 si el valor del primer operando es menor que el límite inferior o mayor que el límite superior.

Ejemplo de codificación

BOUND BX, LIMITS

(BOUND BX, LIMITES)

CLTS

Control de banderas y procesador

Clear Task Switched Flag

(Limpiar la bandera de tarea conmutada):CLTS limpia la bandera de tarea conmutada del registro de estado de la máquina. Esta instrucción se usa sólo en software del sistema operativo (no en software de aplicaciones).

Banderas afectadas

Ninguna del registro de banderas; afecta la palabra de estado de la máquina

ENTER

Control de banderas y procesador

Make Stackfor Procedure Parameters

(Crear estructura de pila para parámetros de procedimiento): ENTER modifica la pila adecuadamente para entrar a un procedimiento de lenguaje de alto nivel. El primer operando especifica el número de bytes de almacenamiento que debe asignarse a la pila; el segundo especifica el nivel de anidamiento de la rutina. Los efectos de esta instrucción se deshacen mediante la instrucción LEAVE.

Ejemplos de codificación
ENTER PPTR, 3
ENTER DS:BX,O

INS

Transferencia de datos

Input String from Port

(Cadena de entrada desde el puerto): INS carga un byte o una palabra desde la dirección especificada de puerto de E/S de hardware hasta el operando de destino. El tamaño del operando de destino determina si se transfiere un

byte o una palabra. El número de puerto puede variar entre 0 y 65, 535.

Ejemplos de codificación

```
INS CX, DX          Cargar una palabra;
INS BL, DX          Cargar un byte
```

INSB

Transferencia de datos

Input String Byte from Port

(Byte de cadena de entrada desde el puerto): INSB carga un byte desde la dirección de puerto de E/S de hardware especificada en DX hasta la dirección especificada por ES: [DI]. El número de puerto puede variar de 0 a 65,535. Después de la transferencia, DI cambia en un incremento o decremento de 1, dependiendo de la activación de la bandera de dirección.

INSW

Transferencia de datos

Input String Wordfrom Port

(Palabra de cadena de entrada desde el puerto): INSW carga una palabra desde la dirección de puerto de E/S de hardware especificada en DX hasta la dirección especificada por ES: [DI]. El número de puerto puede varia de 0 a 65 535 después de la transferencia, DI cambia incrementos o decrementos de 2, dependiendo de la activación de la bandera de dirección.

LAR

Transferencia de datos.

Load Access-Rights Byte

(Cargar el byte de derechos de acceso): Basándose en la selección del segundo operando, el byte alto del registro de destino se sobreescribe con el valor del byte de derechos de acceso, mientras que el byte inferior se pone en cero. La carga se hace sólo si el descriptor es visible en el nivel actual de privilegio y en el selector RPL. La bandera de cero se activa si la operación de carga resulta exitosa.

Bandera afectada
ZF
Ejemplo de codificación
LAR AX, SELECT

LEAVE

Control de banderas y procesador

High-Level Procedure Exit

(Salida de procedimiento de alto nivel): LEAVE deshace los cambios realizados por la instrucción ENTER. Esta instrucción se usa para salir de subrutinas de lenguajes de alto nivel.

LGDT

Transferencia de datos

Load Global Descriptor Table Register

(Cargar el registro de tabla de descriptor global): LGDT carga los seis bytes asociados con la tabla de descriptor global de la dirección de memoria especificada en el operando. Esta instrucción se debe usar en software de sistemas operativos de modo protegido; no se usa en software de aplicaciones.

Ejemplo de codificación

LGDT TEMP[BX]

LIDT

Transferencia de datos

Load Interrupt Descriptor Table Register

(Cargar el registro de tabla de descriptor de interrupción): LIDT carga los seis bytes asociados con la tabla de descriptor de interrupción de la dirección de memoria especificada en el operando. Esta instrucción debe usarse en software de sistemas operativos de modo protegido; no se usa en software de aplicaciones.

Ejemplo de codificación

LIDT TEMP[BX]

LLDT

Transferencia de datos

Load Local Descriptor Table Register

(Cargar el registro de tabla de descriptor local):

Basándose en el selector especificado en el operando, LIDT transfiere la entrada de tabla de descriptor global válido a la tabla de descriptor local. Esta instrucción se debe usar en software de sistemas operativos de modo protegido (no en software de aplicaciones).

Ejemplo de codificación

LLDT AX

LMSW

Transferencia de datos

Load Machine Status Word

(Cargar la palabra de estado de máquina): LMSW copia el valor del operando en la palabra de estado de la máquina. Esta instrucción se debe usar sólo en software de sistemas operativos (no en software de aplicaciones).

Ejemplo de codificación

LMSW AX

LSL

Transferencia de datos

Load Segment Limit

(Cargar el límite de segmento): Basándose en el selector especificado en el operando fuente, LSL carga el campo de límite del descriptor en el operando (registro) objetivo. El descriptor denotado por el selector debe ser visible. Si la carga resulta exitosa, se activa la bandera de cero; de otra manera, se limpia.

Bandera afectada
ZF

Ejemplo de codificación
LSL AX, SELECTOR

LTR

Transferencia de datos

Load Task Register

(Cargar el registro de tareas): LTR carga el registro de tareas a partir del valor del operando fuente. Esta instrucción debe usarse sólo en software de sistemas operativos y no en software de aplicaciones.

Manual de bolsillo Lenguaje ensamblador

Ejemplos de codificación

LTR DX

LTR TEMP [BX]

OUTS

Transferencia de datos

Output String to Port

(Cadena de salida a puerto): OUTS envía un byte o una palabra (la longitud se especifica mediante la el tamaño del operando fuente) a la dirección del puerto de E/S de hardware especificada en DX. El número de puerto puede variar entre O y 65,535.

Ejemplos codificación

```
OUTS DX,CX     ;Palabra de salida

OUTS DX,BL     ;Byte de salida
```

OUTSB

Transferencia de datos

Output String Byte to Port

(Byte de cadena de salida a puerto): OUTSB envía un byte de la dirección especificada por DS:[SI] a la dirección de puerto de E/S de hardware especificada en DX. El número de puerto puede variar entre O y 65,535. Después

de la transferencia, DI cambia en un incremento o decremento de 1, dependiendo de la activación de la bandera de dirección.

OUTSW

Transferencia de datos

Output String Word to Port

(Palabra de cadena de salida a puerto): OUTSW envía una palabra de la dirección especificada por DS: [SI] a la dirección de puerto de E/S de hardware especificada en DX. El número de puerto puede variar de O a 65,535. Después de la transferencia, DI cambia en incrementos o decrementos de 2, dependiendo de la activación de la bandera de dirección.

POPA

Transferencia de datos

Pop All General Registers

(Sacar todos los registros generales): POPA quita los registros de propósito general y los carga de la pila en este orden: DI, SI, BP (apuntador de base), SP (apuntador de pila), BX, DX, CX, AX. El registro SP se descarta cuando se saca.

PUSHA

Transferencia de datos

Push All General Registers

(Meter todos los registros generales): Los registros de propósito general se meten en la pila en este orden: AX, CX, DX, BX, SP, BP, SI, DI. El valor SP que se mete es aquel existente antes de ejecutar esta instrucción.

SGDT

Transferencia de datos

Store Global Descriptor Table Register

(Almacenar el registro de tabla de descriptor global): SGDT transfiere los seis bytes de la tabla de descriptor

global a la dirección de memoria especificada en el operando. Esta instrucción debe usarse en software de sistemas operativos en modo protegido y no en software de aplicaciones.

Ejemplo de codificación

SGDT TEMP[BX]

SIDT

Transferencia de datos

Store Interrupt Descriptor Table Register

(Almacenar el registro de tabla de descriptor de interrupción): SIDT transfiere los seis bytes de la tabla de descriptor de interrupción a la dirección de memoria especificada en el operando. Esta instrucción debe usarse en software de sistemas operativos en modo protegido y no en software de aplicaciones.

Ejemplo de codificación

SITD TEM [BX]

SLDT

Transferencia de datos

Store Local Descriptor Table Register

(Almacenar el registro de tabla de descriptor local): SLDT copia el contenido de la tabla de descriptor local en los dos bytes del operando. Esta instrucción debe usarse es software de sistemas operativos en modo protegido y no en software de aplicaciones.

Ejemplos de codificación

SLDT AX

SLDT LDT_TEMP

SMSW

Transferencia de datos

Store Mach Status Word

(Almacenar la palabra de estado de la máquina): SMSW copia el valor de la palabra de estado de la máquina en el operando. Esta instrucción debe usarse sólo en software de sistemas operativos y no en software de aplicaciones.

Ejemplos de codificación

SMSW AX

SMSW MSW_TEMP

STR

Transferencia de datos

Store Task Register

(Almacenar el registro de tareas): STR copia el valor del registro de tareas en el operando. Esta instrucción sólo debe usarse en software de sistemas operativos y no en software de aplicaciones.

Ejemplos de codificación

STR AX

STR MSW_TEMP

VERR

Control de banderas y procesador

Verify a Segment for Reading

(Verificar un segmento para leer): VERR determina si el selector especificado en el operando es visible en el actual nivel de privilegio y si es legible. La bandera de cero se activa si el selector es accesible.

Bandera afectada

ZF

Ejemplo de codificación

VERR TEMP

VERR AX

VERW

Control de banderas y procesador

Verify a Segment for Writing

(Verificar un segmento para escribir): VERW determina si el selector especificado en el operando es visible en el actual nivel de privilegio y si puede escribirse. La bandera de cero se activa si el selector está accesible.

Bandera afectada

ZF

Ejemplo codificación

```
VERW TEMP

VERM AX
```

COMPENDIO DEL 80386 DE INTEL

Existen muchas más diferencias entre el 80386 y las primeras generaciones de microprocesadores que entre el 80286 y el 8086/8088. Las diferencias (desde el punto de vista de un programador) tienen que ver con los registros, las banderas y el conjunto de instrucciones.

Registro del 80386

A diferencia de sus predecesores de 8 y 16 bits, el 80386 en un microprocesador de 32 bits. Los registros en el 80386 reflejan su estructura agrandada. El 80386 sigue usando los mismos registros de propósito general que el 8086/8088 y el 80286 (AX, BX, CX y DX), pero las contrapartes completas de 32 bits de los registros se dirigen mediante el uso del prefijo E (ampliado). EAX, EBX, ECX y EDX son registros de propósito general de 32 bits. Sin la E, sólo se tiene acceso a los 16 bits inferiores de cada registro. El uso de los tradicionales AL, AH, BL, BH, CL, CH, DL o DH permite el acceso a trozos de 8 bits de los 16 bits inferiores de los registros.

Este uso del prefijo E para denotar el tamaño de registros de 32 bits también se aplica a otros registros de microprocesadores, como BP, SI, DI y SP, que se convierten en EBP, ESI, EDI y ESP, respectivamente.

Los otros registros de segmento CS, DS, SS y ES permanecen intactos, tal y como se establecieron en los anteriores microprocesadores Intel. Estos registros siguen siendo de 16 bits de anchura. Sin embargo, se han incluido dos nuevos registros de segmento (también de 16

bits de anchura): los registros FS y GS, que operan igual que el registro ES.

Banderas del 80386

Como los microprocesadores anteriores, el 80386 utiliza un registro de banderas, pero éste de 32 bits de ancho. Sin embargo, los 14 bits superiores de este registro están reservados y no se encuentran disponibles para los programadores. Los restantes 18 bits son aquellos con los cuales los programadores se deben relacionar. La descripción detallada de cada instrucción indica cómo estos 18 bits de bandera se ven afectados por la ejecución de la instrucción.

Además de las banderas estándar listadas para el 8086/ 8088, el 80386 también contiene las siguientes:

Abreviatura	Significado
VM	Modo virtual
R	Reanudar
NT	Tarea anidada
IOPL	Nivel de privilegio de E/S

Conjunto de instrucciones del 80386

El conjunto de instrucciones del 80386 de Intel es una extensión del conjunto de instrucciones para el 80286 y, posteriormente, para el 8086/8088 de Intel. El conjunto de instrucciones del 80386 tiene otras 57 instrucciones que no están disponibles para el 80286. Algunas de estas nuevas instrucciones se usan para el desarrollo del software de sistemas operativos; ninguna está diseñada para no usarse en software de aplicaciones.

Esta sección no contiene instrucciones listadas anteriormente en esta guía. Sólo incluye las extensiones disponibles específicamente con el 80386 y aquellas instrucciones que cambian operacionalmente cuando se usan los registros de 32 bits del 80386.

Las 57 instrucciones añadidas con el 80386 se incluyen en la siguiente lista:

BSF	CDQ	LGS
BSR	CMPSD	LODSD
BT	CWDE	LSS
BTC	INSD	MOVSD
BTR	JECXZ	MOVSX
BTS	LFS	MOVZX
OUTSD	SETGE	SETNO
POPAD	SETL	SETNP
POPFD	SETLE	SETNS
PUSHAD	SETNA	SETNZ
PUSHFD	SETNAE	SETO
SCASD	SETNB	SETP
SETA	SETNBE	SETPE
SETAE	SETNC	SETPO
SETB	SEINE	SETS
SETBE	SETNO	SETZ
SETC	SETNGE	SHLD
SETE	SETNL	SHRD
SETG	SETNLE	STOSD

En la siguiente lista se proporcionan aquellas instrucciones modificadas en cuanto a su establecimiento respectivo en el 8086/8088 o el 80286:

ADC	MOV	REPNZ
ADD	MUL	REPZ
AND	NEO	ROL
CMP	NOT	ROR
DEC	OR	SAL
DIV	OUT	SAR
IDIV	POP	SBB
IMUL	PUSH	SHL
IN	RCL	SHR
INC	RCR	SUB
INS	REP	TEST
LAR	REPE	XCHG
LEA	REPNE	XOR

La siguiente lista contiene las 57 instrucciones adicionales para el 80386 de Intel, ordenada según una clasificación general de instrucciones. Dentro de cada clasificación, las instrucciones se hallan en orden alfabético.

Aritmética	LGS	SETNC
CBQ	LSS	SETNE
CWDE	MOVSX	SETNG
Manipulación de	MOVZX	SEINGE
bits	OUTSD	SETNL
BSF	POPAD	SETNLE
BSR	POPFD	SETNO
BT	PUSHAD	SETNP

BTC	PUSHFD	SETNS
BTR	SETA	SETNZ
BTS	SETAE	SETO
SHLD	SETB	SETP
SHRD	SETBE	SETPE
Transferencia	SETC	SETPO
de control	SETE	SETS
JECXZ	SETG	SETZ
Transferencia	SETGE	Manipulación
de datos	SETL	de cadenas
INSD	SETLE	CMPSD
LFS	SETNA	LODSD
	SETNAE	MOVSD
	SETNB	SCASD
	SETNBE	STOSD

La siguiente tabla muestra la forma en que las instrucciones adicionales del 80386 afectan las banderas. 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 ejecutarse la instrucción. En esta tabla se incluyen sólo aquellas instrucciones que afectan banderas.

Inst	OF	DF	IF	TF	SF	ZF	AF	PF	CF
BSF						X			
BSR						X			
BT									X
BTC									X
BTR									X
BTS									X

CMPSD	X		X	X	X	X	X
LAR				X			
SCASD	X		X	X	X	X	X
SHLD	?		X	X	?	X	X
SHRD	?		X	X	?	X	X

CONJUNTO DE INSTRUCCIONES DEL 80386 DE INTEL

Las instrucciones presentadas en esta sección funcionan como se describe en el 80386.

ADC

Aritmética

Add with Carry

(Sumar con acarreo): ADC suma el contenido del operando fuente al operando de destino (y almacena el resultado en éste). Si la bandera de acarreo está activada, el resultado cambia en un incremento de 1. En esta rutina, se supone que los valores sumados son binarios.

Banderas afectadas

OF, SF, ZF, AF, PF, CF

Ejemplos de codificación

```
ADC AX,BX           ;AX=AX+BX+CF

ADC EAX, TEMP       ;EAX=EAX+TEMP+CF

ADC SUM, EBX        ;SUM=SUM+EBX+CF

ADC CL,10           ;CL=CL+10+CF

ADC AX, TEMP [BX]   ;Dirección indirecta
```

ADD

Aritmética

Add

(Sumar): ADD suma el contenido del operando fuente al operando de destino (y almacena el resultado en éste). En esta rutina, se supone que los valores sumados son binarios.

Banderas afectadas

OF, SF, ZF, AF, PF, CF

Ejemplos de codificación

```
ADD AX,BX                ;AX=AX+BX

ADD EAX, TEMP            ;EAX=EAX+TEMP

ADD SUM, EBX             ;SUM=SUM+EBX

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 activa en 1 sólo si el bit correspondiente de cada operando también se activa en 1. Las banderas de acarreo y de sobreflujo se limpian mediante esta operación.

Banderas afectadas

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

Ejemplos de codificación

```
AND AX; BX;

AND EAX, TEMP       ;TEMP debe ser una palabra doble

AND SUM, EBX        ;SUM debe ser una palabra doble

AND CL, 00001111b   ;Nibble alto en cero

AND AX, TEMP [BX]   ;Dirección indirecta
```

BSF

Manipulación de bits

Bit Scan Forward

(Examen de bits hacia adelante): BSF examina los bits del segundo operando (comenzando con el bit 0) para ver si alguno está activado. Si todos los bits están limpios (el segundo operando es 0), el primer operando no se modifica, y se activa la bandera de cero. Si algún bit está activado, la bandera de cero se limpia y el primer operando se activa igual que el número del bit activado.

Bandera afectada

ZF

Ejemplos de codificación

BSF EAX, TEMP

BSF CX, BX

BSR

Manipulación de bits

Bit Scan Reverse

(Examen de bits hacia atrás): BSR examina los bits del segundo operando (comenzando con el bit de alto orden) para ver si alguno está activado. Si todos los bits están limpios (el segundo operando es 0), el primer operando no se modifica, y se activa la bandera de cero. Si algún bit está activado, la bandera de cero se limpia y el primer operando se activa igual que el número del bit que está activado.

Bandera afectada

ZF

Ejemplos de codificación

BSR EAX, TEMP

BSR CX,FX

BT

Manipulación de bits

Bit Test

(Prueba de bits): Usa el valor del segundo operando como índice de bit en el valor del primer operando. El bit que está en la posición indizada del primer operando se copia en la bandera de acarreo.

Bandera afectada

CF

Ejemplos de codificación

```
BT TEMP   ;EAX

BT BX,CX

BT TEMP,3 ;Probar el tercer bit
```

BTC

Manipulación de bits

Bit Test and Complement

(Prueba de bit y complemento): BTC usa el valor del segundo operando como índice de bits en el valor del primer operando. El valor opuesto del bit en la posición indizada del primer operando se copia en la bandera de acarreo.

Bandera afectada

CF

Ejemplos de codificación

```
BTC TEMP, EAX

BTC BX, CX

BTC TEMP,3     ;Opuesto al tercer bit
```

BTR

Manipulación de bits

Bit Testand Reset

(Prueba de bits y reinicio): BTR usa el valor del segundo operando como índice de bits en el valor del primer operando. El bit de la posición indizada del primer

operando se copia en la bandera de acarreo, y después se limpia el valor del bit original.

Bandera afectada

CF

Ejemplos de codificación

```
BTR TEMP, FAX

BTR BX, CX

BTR TFMP, 3    ;Valor del tercer bit
```

BTS

Manipulación de bits

Bit Test and Set

(Prueba de bits y activación): BTS usa el valor del segundo operando como índice de bits en el valor del primer operando. El bit de la posición indizada del primer operando se copia en la bandera de acarreo, y después se activa el valor del bit original.

Bandera afectada

CF

Ejemplos de codificación

```
BTS TEMP, EAX

BTS BX, CX

BIS TEMP, 3    ;Valor del tercer bit
```

CDQ

Aritmética

Convert Double word to Quadword

(Convertir una palabra doble en palabra cuádruple): CDQ convierte el valor de palabra doble de EAX en un valor de palabra cuádruple de EDX: EAX ampliando el valor de bit de alto orden de EAX a través de todos los bits de EDX.

CMP

Aritmética

Compare

(Comparar): CMP se considera una instrucción aritmética debido a que el operando fuente se resta del operando de destino. Sin embargo, el resultado se usa para activar las banderas; no se almacena en ningún lado. Puede usarse 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, FBX   ;SUM debe ser una palabra doble

CMP CL,3 ;Comparar con una constante

CMP AX, TEMP [BX] ;Dirección indirecta
```

CMPSD

Manipulación de cadenas

Compare Strings, Doubleword-for-Doubleword

(Comparar cadenas, palabra doble por palabra doble): Esta instrucción compara cadenas, palabra doble por palabra doble. EDI y ESI cambian en incrementos o decrementos de 4, dependiendo de la activación 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 ECX palabras. 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

```
CMPSD          ;Comparar cadenas

REPE CMPSD     ;Repetir un ciclo
```

CWDE

Aritmética

Convert Word to Doubleword

(Convertir una palabra en palabra doble): CWDE convierte el valor de palabra en AX al valor de palabra doble en EAX extendiendo el valor del bit de alto orden de AX a través de los bits restantes de EAX.

DEC

Aritmética

Decrement

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

Banderas afectadas

OF, SF; ZF; AF; PF

Ejemplos de codificación

```
DEC AX

DEC FCX

DEC SUM

DEC BL

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 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 almacena el resultado en AX y el residuo en DX. Si el operando es una palabra doble, DIV divide el contenido de EDX:EAX entre el contenido del operando y almacena el resultado en EAX y el residuo en EDX. Esta instrucción trata los números como valores binarios sin signo.

Banderas Afectadas

CF (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 WORD_TEMP  ;AX=DX:AX/WORD_TFMP

DIV BYTE_SUM   ;AL=AX/BYTE_SUM

DIV WORDSUM    ;EAX=EDX:EAX/DWORDSUM

DIV WORD_TBL [BX];Dirección indirecta
```

IDIV

Aritmética

Integer Divide

(Dividir enteros): Si el operando es un valor de byte, IDIV divide el contenido de AX entre el contenido del operando y 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 almacena el resultado en AX y el residuo en DX. Si el operando es una palabra doble, IDIV divide el contenido de EDX:EAX entre el contenido del operando y almacena el resultado en EAX y el residuo en EDX. 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 WORD_TEMP ;AX=DX:AX/WORD_TEMP

IDIV BYTE_SUM  ;AL=AX/BYTE_SUM

IDIV DWD_SUM   ;EAX=EDX:EAX/DWD_SUM

IDIV WORD_TBL [BX]  ;Dirección indirecta
```

IMUL

Aritmética

Integer Multiply

(Multiplicar enteros): Los resultados de esta operación dependen del número de operandos especificado. Si sólo se da un operando, éste se multiplica por AL, AX o EAX. Si el operando es un valor de byte, IMUL multiplica el contenido de 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 el resultado en DX:AX.

Si se dan dos operandos, IMUL multiplica el primer operando por el segundo y almacena el resultado en el primer operando. Ambos operandos deben coincidir el tamaño.

Si se dan tres operandos y el tercero es un valor inmediato, IMUL multiplica el segundo operando por el tercero y almacena el resultado en el primer operando.

Esta instrucción trata 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 WORD_TEM ;DX:AX=AX*WORD_TEMP

IMUL BYTE_SUM  ;AX=AL*BYTE_SUM

IMUL WORD_TBL [BX] ;Dirección indirecta

IMUL ECX, DWRD, 10 ;ECX=DWRD*10
```

IN

Transferencia de datos

Jnput from Port

(Entrada desde un puerto): IN carga un byte, una palabra o una palabra doble en AL, AX o EAX, respectivamente, desde la dirección especificada de puerto de E/S de hardware. Un número de puerto menor que 256 puede especificarse como constante o como variable en registro DX. Sin embargo, un número de puerto mayor que 255 debe especificarse en el registro DX.

Ejemplos de codificación

```
IN AL,64h

IN AX,DX

IN EAX, DX
```

INC

Aritmética

Increment

(Incrementar): INC cambia el contenido del operando en incrementos de 1. 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 EDI

INC TEMP [SI]
```

INS

Transferencia de datos

Jnput Stringfrom Port

(Cadena de entrada desde un puerto): INS carga un byte, una palabra o una palabra doble desde la dirección especificada de puerto de E/S de hardware (indicada por el valor en DX) en el operando de destino. El tamaño del operando de destino determina si se transfiere un byte, una palabra o una palabra doble. Si el operando de destino es una dirección de desplazamiento, esa dirección es relativa al registro ES. No es posible sobrepasar ningún segmento.

Ejemplos de codificación

```
INS CX,DX ;Cargar una palabra

INS BL,DX ;Cargar un byte

INS EAX,DX ;Cargar una palabra doble
```

INSD

Transferencia de datos

Input String Double word from Port

(Palabra doble de cadena de entrada desde un puerto): INSD carga una palabra de la dirección de puerto de E/S especificada en DX en la dirección especificada por ES:[EDI]. Después de la transferencia, EDI cambia en incrementos o decrementos de 4, dependiendo de la activación de la bandera de dirección.

JECXZ

Transferencia de control

Jump if ECX=O

(Saltar si ECX=0): JECXZ ocasiona que la ejecución del programa se ramifique a la dirección del operando si el valor de ECX es cero.

Ejemplo de codificación

```
JECXZ SKIP_LOOP
```

LAR

Transferencia de datos

Load Access-Rights Byte

(Cargar el byte de derechos de acceso): Basándose en la selección del segundo operando, el byte alto del registro de destino se sobreescribe con el valor del byte de derechos de acceso, y el byte inferior se pone en ceros. La carga se hace sólo si el descriptor es visible en el nivel de privilegio actual en ese momento y en el selector RPL. La bandera de cero se activa si la operación de carga resulta exitosa.

Bandera afectada

ZF

Ejemplo de codificación

```
LAR AX,SELECT
```

LEA

Transferencia de datos

Load Effective Address

(Cargar la dirección efectiva): LEA transfiere la dirección de desplazamiento del operando fuente al operando de destino. Este último operando debe ser una palabra general o un registro de palabra doble.

Ejemplos de codificación

```
LEA AX, MESSAGE_1

LEA EBX, SOURCE BLOCK
```

LFS

Transferencia de datos

Load FS Register

(Cargar el registro FS): LFS realiza dos operaciones distintas: carga FS con la dirección de segmento del operando fuente y luego carga el operando de destino con la dirección de desplazamiento del operando fuente.

Ejemplo de codificación

```
LFS DI, DEST_BUEFER
```

LGS

Transferencia de datos

Load GS Register

(Cargar al registro GS): LGS realiza dos operaciones distintas: carga GS con la dirección de segmento del operando fuente y luego carga el operando de destino con la dirección de desplazamiento del operando fuente.

Ejemplo de codificación

```
LGS DI, DEST_BUEFER
```

LODSD

Manipulación de cadenas

Load a Doubleword from String into EAX

(Cargar una palabra doble de cadena en EAX): LODSD carga EAX con el contenido de la dirección apuntada por ESI; después ESI cambia en incrementos o decrementos de 4, dependiendo de la activación de la bandera de dirección.

LSS

Transferencia de datos

Load SS Register

(Cargare registro SS(segmento de pila)): LSS realiza dos operaciones distintas: carga SS con la dirección de segmento del operando fuente y luego carga el operando de destino con la dirección de desplazamiento del operando fuente.

Ejemplo de codificación

```
LSS DI, DEST_BUFFER
```

MOV

Transferencia de datos

Move

(Trasladar): MOV copia el contenido del operando fuente en el operando de destino. Cuando el operando fuente no es un valor inmediato, ambos operandos deben concordar en la longitud. Si el operando fuente o el de destino es un registro de palabra doble, el otro registro puede ser un

registro especial, como CR0, CR2, CR3, DR0, DR1, DR2, DR3, DR6, DR7, TR6 TR7.

Ejemplos de codificación

```
MOV AX,BX; AX=BX

MOV FAX, TEMP; EAX=TEMP

SUM, BX; SUM=BX

MOV CL, 57; CL = 57

MOV DECIMAL, 10; DECIMAL 10

MOV AX, TEMP [BX]; dirección indirecta
```

MOVSD

Manipulación de cadenas

Move String, Doubleword-by-Doubleword

(Trasladar una cadena, palabra doble por palabra doble): MOVSD traslada cadena, palabra doble por palabra doble. Los valores de ESI y EDI cambian en incrementos o decrementos de 4, dependiendo de la activación de la bandera de dirección. Normalmente, esta instrucción se usa con la instrucción REP para repetir el traslado por un máximo de ECX palabras.

Ejemplos de codificación

```
MOVSD

REP MOVSD; Repetir un ciclo de traslado
```

MOVSX

Transferencia de datos

Move with Sign Extended

(Trasladar con signo extendido): MOVSX traslada el operando fuente al operando de destino y extiende el bit

de alto orden al resto de los bits del operando de destino. El operando fuente debe ser más pequeño que el de destino.

Ejemplos de codificación

```
MOVSX EAX,BX;  EAX=BX
MOVSX EAX, TEMP; EAX=TEMP
MOVSX CX,AL; CX=AL
```

MOVZX

Transferencia de datos

Move with Zero Extended

(Trasladar con cero extendido): MOVZX traslada el operando fuente al operando de destino y limpia los bits restantes del operando de destino. El operando fuente debe ser más pequeño que el de destino.

Ejemplos de codificación

```
MOVZX FAX, BX; EAX=BX
MOVZX FAX, TEMP; EAX=TEMP
MOVZX CX,AL; CX=AL
```

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 el 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 de DX:AX. Si el operando es un valor de palabra doble, MUL multiplica el contenido de EAX por el contenido del operando y almacena el resultado en EDX:EAX. Esta instrucción trata los números como valores binarios sin signo.

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 ECX; EDX:EAX=EAX*ECX
MUL WORD_TEMP; DX:AX=AX* WORD_TEMP
MUL BYTE_SUM ;AX=AL*BYTE_SUM
MUL WORD_TBL [BXJ; 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 la resta del operando de destino de 0.

Banderas afectadas
OF, SF, ZF, AF, PF, CF

Ejemplos de codificación

```
NEG TFMP
NEG CL
NEG EAX
```

NOT

Manipulación de bits

Logical NOT on Bits

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

Ejemplos de codificación

```
    NOT CL
    NOT BYTE_SUM   ;Usar un valor de byte
    NOT WORD_SUM   ;Usar 1111 valor de palabra
    NOT DWORD_SUM ;Usar un valor de palabra doble

    NOT AX
```

```
NOT EBX
```

OR

Manipulación de bits

Logical OR on Bits

(O lógico sobre bits): O 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 alguno o ambos bits correspondientes de cada operando se pone en 1.

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

Ejemplos de codificación

```
OR AL,BL
OR EAX, OFFFFh
OR DX, TEMP
OR AX, CX
```

OUT

Transferencia de datos

Output to Port

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

Ejemplos de codificación

```
OUT 64h, AL
OUT DX,AX
OUT DX,EAX
```

OUTSD

Output String Doubleword to Port

(Salida de una palabra doble de cadena a puerto):
OUTSD envía una palabra de la dirección especificada por DS:[ESI] a la dirección de puerto de E/S de hardware especificada en DX. Después de la transferencia, DI cambia en incrementos o decrementos de 4, dependiendo de la activación de la bandera de dirección.

POP

Remove Datafrom Stack

(Sacar datos de la pila): POP traslada una palabra o una palabra doble (dependiendo del tamaño del operando) de la pila y la coloca en el operando de destino deseado.

Ejemplos de codificación

```
POP AX
POP DS
POS GS
POP HOLD_REG
```

POPAD

Transferencia de datos

Pop All General Doubleword Registers

(Sacar todos los registros generales de palabra doble): POPAD traslada y carga los registros de propósito general de la pila en este orden: EDI, ESI, EBP, ESP, EBX, EDX, ECX, EAX. El registro ESP se descarta cuando se saca.

POPFD

Transferencia de datos

Remove Extended Flags from Stack

(Trasladar banderas extendidas de la pila): POPFD traslada una palabra doble de la pila y la coloca en el registro de banderas extendidas.

Banderas afectadas
VM, R, NT, IOPL, OF, DF, IF, TF, SF, ZF, AF, PF, CF

PUSH

Transferencia de datos

Place Data on Stack

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

Ejemplos de codificación

```
PUSH AX
PUSH EBX
PUSH DS
PUSH HOLD REG
```

PUSHAD

Transferencia de datos

Push All General Doubleword Registers

(Meter todos los registros generales de palabra doble): PUSHAD mete en la pila los registros de palabra doble de propósito general, en este orden: EAX, ECX, EDX, EBX, ESP EBP, ESI, EDI. El valor ESP metido es el valor existente antes de ejecutar esta instrucción.

PUSHFD

Transferencia de datos

Place Extended Flags on Stack

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

RCL

Manipulación de bits

Rotate Left through Carry

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

Banderas Afectadas

OF, CF

Ejemplos de codificación

```
RCL AX,1
RCL BL,3
RCL EDX,16
RCL TEMP;CL
```

RCR

Manipulación de bits

Rotate Right Through Carry

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

Banderas Afectadas

OF, CF

Ejemplos de codificación

```
RCR AX,1
RCR BL,3
RCR EDX,16
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 (si esta instrucción se usa con operandos de byte o de palabra) o en ECX (si esta instrucción se usa con operandos de palabra doble).

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 (si esta instrucción se usa con operandos de byte o de palabra) o en ECX (si esta instrucción se usa con operandos de palabra doble). Cuando se usa con CMPSB, CMPSW, SCA&B o SCASW, esta instrucción sólo se repite cuando 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 tantas veces como se indique en CX (si esta instrucción se usa con operandos de byte o de palabra) o en ECX (si esta instrucción se usa con operandos de palabra doble. Cuando se usa con CMPSB, CMPSW, SCASB o SCASW, REPNE causa la repetición sólo si la bandera de cero está activada. 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 las instrucciones de manipulación de cadenas se repitan tantas veces como se indique en CX (si esta instrucción se usa con operandos de byte o de palabra) o en ECX (si esta instrucción se usa con operandos de palabra doble). 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 REPNE.

Ejemplo de codificación

```
REPNZ CMPSW
```

REPTZ

Manipulación de cadenas

Repeat if Zero

(Repetir si es cero): REPZ ocasiona que las instrucciones de manipulación de cadenas se repitan tantas veces como se indique en CX (si esta instrucción se usa con operandos de byte o de palabra) o en ECX (si esta instrucción se usa con operandos de palabra doble). 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
```

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 EAX,1
ROL BL,3
ROL DX, 16
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

```
ROR EAX, 1
ROR BL,3
ROR DX,16
ROR TEMP,CL
```

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 EAX, 1
SAL BL,3
SAL DX, 16
SAL TEMP,CL
```

SAR

Manipulación de bits

Arithmetic Shif right

Desplazamiento aritmético a la derecha): SAR el operando de destino a la derecha tantos lugares como indique el operando fuente.)r se pierden, mientras que los igual que el bit de alto orden existente.

Banderas afectadas

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

Ejemplos de codificación

```
SAR EAX,1
SAR BL,3
SAR DX,16
SAR TEMP,CL
```

SBB

Aritmética

Substrac With Carry

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

Bandera efectuadas

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, EDX   ;SIJM=SUM-EBX-CF
SBB CL, 10; CL ;CL - 10 - CF
SBB AX, TEMP[BX]; Dirección Indirecta
```

SCASD

Manipulación de cadenas

Scan String for Doubleword

(Examinar una cadena por palabra doble): SCASD resta la palabra de cadena del operando de destino (apuntada por EDI) del valor de EAX. No se almacena el resultado, sino que se actualizan las banderas. Después, el valor de EDI cambia en incrementos o decrementos de 4, dependiendo de la activación de la bandera de dirección. Normalmente, esta instrucción se usa con REPE, REPNE, REPNZ o REPZ para repetir el examen un máximo de CX bytes o hasta que se encuentre una concordancia o una diferencia.

Banderas afectadas

OF, SF, ZF, AF, Pfl CF

Ejemplos de codificación

```
SCASD
REPNZ SCASD
```

SETA

Transferencia de datos

Set Byte if Above

(Activar un byte si es superior): SETA almacena un 1 en el operando si las banderas de acarreo y de cero están limpias. Si esta condición no se cumple, se almacena un 0 en el operando. Este debe ser un registro o una localidad de memoria de un byte de longitud. Esta instrucción es funcionalmente igual que SETNBE.

Ejemplo de codificación

```
SETA CL
```

SETAE

Transferencia de datos

Set Byte if Above or Equal

(Activar un byte si es superior o igual): SETAE almacena un 1 en el operando si la bandera de acarreo está limpia. Si esta condición no se cumple, se almacena un 0 en el operando. Este debe ser un registro o una localidad de memoria de un byte de longitud. Esta instrucción es funcionalmente igual que SETNB o SETNC.

Ejemplo de codificación

```
SETAE CL
```

SETB

Transferencia de datos

Set Byte if Below

(Activar un byte si es inferior): SETB almacena un 1 en el operando si la bandera de acarreo está activada. Si esta condición no se cumple, se almacena un 0 en el operando. Este debe ser un registro o una localidad de memoria de un byte de longitud. Esta instrucción es funcionalmente igual que SETC o SETNAE.

Ejemplo de codificación

SETB CL

SETBE

Transferencia de Datos

Set Byte if Below or Equal

(Activar un byte si es inferior o igual): SETBE almacena un 1 en el operando si está activada la bandera de acarreo o la de cero. Si esta condición no se cumple, se almacena un 0 en el operando. Este debe ser un registro o una localidad de memoria de un byte de longitud.

Esta instrucción es funcionalmente igual que SETNA.

Ejemplo de codificación

SETBE CL

SETC

Transferencia de datos.

Set Byte on Carry

(Activar un byte en acarreo): SETC almacena un 1 en el operando si la bandera de acarreo está activada. Si esta condición no se cumple, se almacena un O en el operando. Este debe ser un registro o una localidad de memoria de un byte de longitud. Esta instrucción es funcionalmente igual que SETB o SETNAE.

Ejemplo de codificación

SETC CL

SETE

Transferencia de datos

Set Byte if Equal

(Activar un byte si es igual): SETE almacena un 1 en el operando si la bandera de cero está activada. Si esta condición no se cumple, se almacena uno en el operando. Este debe ser un registro o una localidad de memoria de un byte de longitud. Esta instrucción es funcionalmente igual que SETZ.

Ejemplo de codificación

SETE CL

SETG

Transferencia de datos

Set Byte if Greater

(Activar un byte si es mayor): SETG almacena un 1 en el operando si la bandera de signo es igual a la bandera de sobre flujo o si la bandera de cero está limpia. Si no se cumple ninguna condición, se almacena un 0 en el operando. Este debe ser un registro o una localidad de memoria de un byte de longitud. Esta instrucción es funcionalmente la misma que SETNLE.

Ejemplo de codificación

SETG CL

SETGE

Transferencia de Datos.

Set Byte if Greater or Equal

(Activar un byte si es mayor o igual): SETGE almacena un 1 en el operando si la bandera de signo es igual a la bandera de sobreflujo. Si esta condición no se cumple, se almacena un 0 en el operando. Este debe ser un registro o una localidad de memoria de longitud. Esta instrucción es funcionalmente igual que SETNL.

Ejemplo de codificación

SETGE CL

SETL

Transferencia de datos

Set Byte if Less Than

(Activar un byte si es menor que): SETL almacena un 1 en el operando si la bandera de signo no es igual a la bandera de sobreflujo. Si esta condición no se cumple, se almacena un 0 en el operando. Este debe ser un registro o una localidad de memoria de un byte de longitud. Esta instrucción es funcionalmente igual que SETNGE.

Ejemplo de codificación

SETL CL

SETLE

Transferencia de datos

Set Byte if Less Than or Equal

(Activar un byte sí es menor que o igual a): SETLE almacena un 1 en el operando si la bandera de signo no es igual a la bandera de sobreflujo o si la bandera de cero está activada. Si no se cumple ninguna condición, se almacena un 0 en el operando. Este debe ser un registro o una localidad de memoria de un byte de longitud. Esta instrucción es funcionalmente igual que SETNG.

Ejemplo de codificación

```
SETLE CL
```

SETNA

Transferencia de datos

Set Byte if Not Above

(Activar un byte si no es superior): SETNA almacena un 1 en el operando si está activada la bandera de acarreo o la de cero. Si ninguna condición se cumple, se almacena un 0 en el operando. Este debe ser un registro o una localidad de memoria de un byte de longitud. Esta instrucción es funcionalmente equivalente a SETBE.

Ejemplo de codificación

```
SETNA CL
```

SETNAE

Transferencia de datos

Set Byte if Not Above or Equal

(Activar un byte si no es superior o igual): SETNAE almacena un 1 en el operando si la bandera de acarreo está activada. Si esta condición no se cumple, se almacena un 0 en el operando. El operando debe ser un registro o una localidad de memoria de un byte de

longitud. Esta instrucción es funcionalmente igual que SETB SETC.

Ejemplo de codificación
SETNAE CL

SETNB

Transferencia de datos

Set Byte If Not Below.

(Activar un byte si no es inferior): SETNB almacena un 1 en el operando si la bandera de acarreo está limpia. Si esta condición no se cumple, se almacena un 0 en el operando. Este debe ser un registro o una localidad de memoria de un byte de longitud. Esta instrucción es funcionalmente igual que SETAE O SETNC.

Ejemplo de codificación
SETNB CL

SETNBE

Transferencia de datos

Set Byte if Not Below or Equal

(Activar un byte si no es inferior o igual): SETNBE almacena un 1 en el operando si la bandera de acarreo como la de cero están limpias. Si esta condición no se cumple, se almacena un e en el operando. El operando debe ser un registro o una localidad de memoria de un byte de longitud. Esta instrucción es funcionalmente igual que SETA.

Ejemplo de codificación
SETNBE CL

SETNC

Transferencia de datos

Set Byte on No Carry

(Activar un byte en no acarreo): SETNC almacena un 1 en el operando si la bandera de acarreo está limpia. Si esta condición no se cumple, se almacena un 0 en el operando. Este debe ser un registro o una localidad de memoria de un byte de longitud. Esta instrucción es funcionalmente igual que SETAE o SETNB.

Ejemplo de codificación

SETNC CL

SETNE

Transferencia de datos

Set Byte if Not Equal

(Activar un byte si no es igual): SETNE almacena un 1 en el operando si la bandera de cero está limpia. Si esta condición no se cumple, se almacena un 0 en el operando. Este debe ser un registro o una localidad de memoria de un byte de longitud. Esta instrucción es funcionalmente igual que SETNZ.

Ejemplo de codificación

SETNE CL

SETNG

Transferencia de datos

Set Byte if Not Greater Than or Equal
(Activar un byte sino es mayor que): SETNG alma-cena un 1 en el operando si la bandera de signo no es igual a la bandera de sobre flujo o si la bandera de cero está activada. Si ninguna condición se cumple, se almacena un O en el operando. Este debe ser un registro o una localidad de memoria de un byte de longitud. Esta instrucción es funcionalmente igual que SETLE.

Ejemplo de codificación

SETNG CL

SETNGE

Transferencia de datos

Set Byte if Not Greater Than or Equal

(Activar un byte si no es mayor que o igual a): SETNGE almacena un 1 en el operando si la bandera de signo no es igual a la bandera de sobreflujo. Si esta condición no se cumple, se almacena un 0 en el operando. Este debe ser un registro o una localidad de memoria de un byte de longitud. Esta instrucción es funcionalmente igual que SETL.

Ejemplo de **codificación**

SETNGE CL

SETNL

Transferencia de datos

Set Byte if Not Less Than

(Activar un byte si no es menor que): SETNL almacena un 1 en el operando si la bandera de signo es igual a la bandera de sobreflujo. Si esta condición no se cumple, se almacena un 0 en el operando. Este debe ser un byt registro o una localidad de memoria de un byte de longitud. Esta instrucción es funcionalmente igual que SETGE.

Ejemplo de codificación
SETNL CL

SETNLE

Transferencia de datos

Set Byte if Not Less Than or Equal

(Activar un byte si no es menor que o igual a): SETNLE almacena un 1 en el operando si la bandera de signo es igual a la bandera de sobreflujo o si la bandera de cero está limpia. Si ninguna condición se cumple, se almacena un 0 en el operando. Este debe ser un registro o una localidad de memoria de un byte de longitud. Esta instrucción es funcionalmente igual que SETO.

Ejemplo de codificación

SETNLE CL

SETNO

Transferencia de datos

Set Byte on No Overflow

(Activar un byte en no sobreflujo): SETNO almacena un 1 en el operando si la bandera de sobreflujo está limpia. Si esta condición no se cumple, se almacena un 0 en el operando. Este debe ser un registro o una localidad de memoria de un byte de longitud.

Ejemplo de codificación

SETNO CL

SETNP

Transferencia de datos

Set Byte on No Parity

(Activar un byte en no paridad): SETNP almacena un 1 en el operando si la bandera de paridad está limpia. Si esta condición no se cumple, se almacena un 0 en el operando. Este debe ser un registro o una localidad de memoria de un byte de longitud. Esta instrucción es funcionalmente igual que SETPO.

Ejémplo de codificación

SETNP CL

SETNS

Transferencia de datos

Set Byte on Not Sign

(Activar un byte en no signo): SETNS almacena un 1 en el operando si la bandera de signo está limpia. Si esta condición no se cumple, se almacena un 0 en el operando. Este debe ser un registro o una localidad de memoria de un byte de longitud.

Ejemplo de codificación

SETNS CL

SETNZ

Transferencia de datos

Set Byte if Not zero

(Activar un byte si no es cero): SETNZ almacena un 1 en el operando si la bandera de cero está limpia. Si esta condición no se cumple, se almacena un 0 en el operando. Este debe ser un registro o una localidad de memoria de un byte de longitud. Esta instrucción es funcionalmente igual que SETNE.

Ejemplo de codificación

SETNZ CL

SETO

Transferencia de datos

Set Byte on Oveflow

(Activar un byte en sobreflujo): SETO almacena un 1 en el operando si la bandera de sobreflujo está activada. Si esta condición no se cumple, se almacena un 0 en el operando. Este debe ser un registro o una localidad de memoria de un byte de longitud.

Ejemplo de codifican
SETO CL

SETP

Transferencia d e datos

Set Byte on Parity

(Activar un byte en paridad): SETP almacena un 1 en el operando si la bandera de paridad está activada. Si esta condición no se cumple, se almacena un 0 en el operando. El operando debe ser un registro o una localidad de memoria de un byte de longitud. Esta instrucción es funcionalmente igual que SETPE.

Ejemplo de codificación
SETP CL

SETPE

Transferencia de datos

Set Byte on Parity Even

(Activar un byte en paridad par): SETPE almacena un 1 en el operando si la bandera de paridad está activada.
Si esta condición no se cumple, se almacena un 0 en el operando. Este debe ser un registro o una localidad de memoria de un byte de longitud. Esta instrucción es **1** funcionalmente igual que SETP.

Ejemplo de codificación

SETPE CL

SETPO

Transferencia de datos

Set Byte on Parity Odd

(Activar un byte en paridad impar): SETPO almacena un 1 en el operando si la bandera de paridad está limpia. Si esta condición no se cumple, se almacena un 0 en el operando. Este debe ser un registro o una localidad de memoria de un byte de longitud. Esta instrucción es funcionalmente igual que SETNP

Ejemplo de codificación
SETPO CL

SETS

Transferencia de datos

Set Byte on Sign

(Activar un byte en signo): SETS almacena un 1 en el operando si la bandera de signo está activada. Si esta condición no se cumple, se almacena un 0 en el operando. Este debe ser un registro o una localidad de memoria de un byte de longitud.

Ejemplo de codificación
SETS CL

SETZ

Transferencia de datos

Set Byte if Zero

(Activar un byte si es cero): SETZ almacena un 1 en el operando si la bandera de cero está activada. Si esta condición no se cumple, se almacena un 0 en el operrando. Este debe ser un registro o una localidad de memoria de un byte de longitud. Esta instrucción es funcionalmente igual que SETE.

Ejemplo de codificación

SETZ CL

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, mientras que los de orden inferior se limpian.

Banderas afectadas

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

Ejemplos de codificación

```
SHL EAX,1
SHL BL,3
SHL DX, 16
SHL TEMP;CL
```

SHLD

Manipulación de bits

Shift Left, Doirbie Precision

(Desplazar a la izquierda, doble precisión): SHLD desplaza todos los bits del primer operando a la izquierda tantos lugares como indique el tercer operando. Los bits de alto orden se pierden, mientras que los de orden inferior se copian del segundo operando, comenzando con el bit de orden inferior. El resultado se almacena en el primer operando.

Banderas afectadas

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

Ejemplos de codificación

```
SHLD AX,BX,4
SHLD DWORD_TEMP, EAX, 16
```

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, mientras que los de alto orden se limpian.

Banderas afectada

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

Ejemplos de codificación

```
SHR EAX,1
SHR BL,3
SHR DX,16
SHR TEMP,CL
```

SHRD

Manipulación de bits

Shift Right, Double Precision

(Desplazar a la derecha, doble precisión): SHRD desplaza todos los bits del primer operando a la derecha tantos lugares como indique el tercer operando. Los bits de orden inferior se pierden y los de alto orden se copian del segundo operando, comenzando con el bit de alto orden. El resultado se almacena en el primer operando.

Banderas afectadas

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

Ejemplos de codificación

```
SHRD AX,BX,4
SHRD DWORD_TEMP, EAX, 16
```

STOSD

Manipulación de cadenas

Store Doubleword in EAX at String

(Almacenar palabra doble en EAX en cadena): Esta instrucción copia el contenido de EAX en la dirección de palabra apuntada por EDI. Luego DI cambia en incrementos o decrementos de 4, dependiendo de la activación de la bandera de dirección.

SUB

Aritmética

Subtract

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

Banderas afectadas

OF, SF, ZF, AF, PF, CF

Ejemplos de codificación

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

TEST

Manipulación de bits

Test Bits

(Probar bits): TEST realiza un Y lógico de los operandos, pero el resultado no se almacena. Sólo las banderas se afectan. Cada bit del byte o palabra resultante se activa

en 1 sólo si el bit correspondiente de cada operando es 1.

Banderas afectadas

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

Ejemplo de codificación

```
TEST AX, BX          ;
TEST AX, TEMP        ;TEMP DEBE SER UNA PALABRA
TEST SUM, EBX        ;SUM DEBE SER UNA PALABRA DOBLE
TEST CL, 00001111b   ;
TEST AX, TEMP [BX],Dirección indirecta
```

XCHG

Transferencia de datos

Exchange

(Intercambiar): Intercambia el contenido de los operandos fuente y de destino. La longitud de ambos operandos debe concordar.

Ejemplo de codificación

```
XCHG AX, BX     ;Intercambia AX con BX
XCHG EAX, DWRD ;Intercambia EAX con DWRD
XCHG CL,CH      ;Intercambia CL con CH
XCHG AX, TEMP   ;Intercambia AX con TEMP
```

XOR

Manipulación de bits

Logical Exciusive-Or on Bits

(O lógico exclusivo sobre bits): XOR realiza un XOR lógico de los operandos y almacena el resultado en el operando de destino. Cada bit del byte o palabra resultante se activa en 1 sólo silos bits correspondientes de cada operando contienen valores opuestos.

Banderas afectadas

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

Ejemplo de codificación

```
XOR AX, BX          ;
XOR FAX, TFMP       ;TEMP debe ser una palabra doble
XOR SUM, BX         ;SUM debe ser una palabra
XOR CL, 00001111b   ;
XOR AX, TEMP [BX]   ;Dirección indirecta
```

COPROCESADORES NUMÉRICOS DE INTEL

En las series Intel hay tres diferentes coprocesadores numéricos. Estos son el 8087 (que trabaja con el 8086/ 8088), el 80287 (que trabaja con el 80286 o el 80386) y el 80387 (que trabaja con el 80386). Intel llama a estos circuitos integrados "extensiones de procesador numérico" debido a que, para todo intento y propósito, se muestran transparentes a los programadores. Los conjuntos de instrucciones y de registros de los microprocesadores principales simplemente parecen haberse ampliado.

Como sucede con la relación entre el 8086/8088 y las generaciones sucesivas de microprocesadores, el conjunto de instrucciones básico para los coprocesadores numéricos Intel es capturado dentro del 8087. Más adelante se explorarán las extensiones de este conjunto básico (como se incorporan en coprocesadores numéricos posteriores).

Antes de que comience a buscar en los conjuntos de registros y de instrucciones de cualquiera de los coprocesadores numéricos, debe darse cuenta de que el 8087 trabaja sobre el principio de una pila flotante (similar a la del 8086/8088), en la que virtualmente se realizan todas las operaciones.

LOS REGISTROS DE PILA

El 8087 y el 80287 usan ocho registros de pila internos, cada uno de los cuales tiene 80 bits de ancho. Estos registros de pila están numerados del O al 7, y la mayor parte de las operaciones son capaces de dirigir estos registros directamente como ST, ST(l), ST(2), ST(3), etc., hasta ST(7).

LA PALABRA DE ESTADO DEL 8087

El 8087 usa una palabra de estado rara describir la condición actual del 8087. La siguiente lista incluye los códigos para las banderas de palabra de estado y el significado de cada código:

CODIGO	USO
IE	Excepción de operación no válida
DE	Excepción de operando desnormalizado
ZE	Excepción de división entre cero
OE	Excepción de sobreflujo
UE	Excepción de subflujo
PE	Excepción de precisión
IR	Solicitud de interrupción
CO	Código de condición O
C1	Código de condición 1
C2	Código de condición 2
ST	Apuntador de tope de pila
C3	Código de condición 3
B	Señal ocupada

Esta palabra de estado no puede examinarse directamente. Debe transferirse mediante instrucciones específicas del 8087 a memoria, donde puede analizarse por medio de instrucciones del 8086/8088.

LA PALABRA DE CONTROL DEL 8087

El 8087 usa una palabra de control para el control de programas de operaciones del 8087. Las banderas (o códigos de estado) de esta palabra de 16 bits se detallan de la siguiente manera:

CODIGO	USO
IM	Máscara de excepción de operación inválida
DM	Máscara de excepción de operando desnormalizado

ZM Máscara de excepción de división entre cero
OM Máscara de excepción de sobreflujo
UM Máscara de excepción de subflujo
PM Máscara de excepción de precisión

CODIGO **USO**

IEBM Máscara de habilitación de interrupción
0 = Interrupciones habilitadas
1 = Interrupciones no habilitadas
PC Control de precisión
00 = 24 bits
01 = (Reservado)
10 = 53 bits
11 = 64 bits
RC Control de redondeo
00 = Redondeo al más cercano o par
01 = Redondear hacia abajo
10 = Redondear hacia arriba
11 = Truncar
IC Control de infinito
0 = Proyectivo
1 Afín

Esta palabra puede construirse en la memoria principal y luego dirigirse al 8087 mediante instrucciones específicas del 8087.
Algunos coprocesadores numéricos posteriores han alterado ligeramente esta palabras de control; las diferenz se cubren más adelante.

AGRUPAMIENTOS DE CONJUNTOS DE INSTRUCCIONES

El 8087 extiende, con 77 instrucciones, el conjunto de instrucciones del 8086/8088. En esta sección se detallan sólo esas 77 instrucciones, agrupadas de acuerdo con el propósito de la instrucción. Las seis clasificaciones generales de instrucciones son:

Transferencia de datos	Trascendental
Aritmética	Constante
Comparaciones	Control de procesador

COMPENDIO DEL 8087 DE INTEL

La siguiente lista contiene las instrucciones disponibles con el 8087:

F2XM1
FABS
FADD
FADDP
FBLD
FBSTP
FCHS
FCLEX
FCOM
FCOMP
FCOMPP
FDECSTP
FDISI
FDIV
FDIVP
FDIVR
FDIVRP
FENI
FFREE
FIADD
FICOM
FICOMP
FIDIV
FIDIVR
FILD
FIMUL
FINCSTP
FINIT
FIST
FISTP
FISUB
FISUBR
FLD
FLDl
FLDCW
FLDENV
FLDL2E
FLDL2T
FLDL(32
FLDLN2
FLDPI
FLDZ
FMUL
FMULP
FNCLEX
FNDISI
FNENI
FNINIT
FNOP
FNSAVE
FNSTCW
FNSTENV
FNSTSW
FPATAN
FPREM
FPTAN
FRNDINT
FRSTOR
FSAVE
FSCALE
FSQRT
FST
FSTCW
FSTENV
FSTP
FSTSW
FSUB
FSUBP
FSUBR
FSUBRP
FTST
FWAIT
FXAM
FXCR
FXTRACT
FYL2X
FYL2XP1

En la siguiente tabla se muestra la forma en que las instrucciones del 8087 afectan las banderas de estado del coprocesador numérico. La X en una columna indica que la bandera de estado es modificada por la instrucción; el signo de interrogación indica que la bandera de estado no se encuentra definida después de ejecutar la instrucción. En esta tabla sólo se incluyen aquellas instrucciones que afectan el estado.

CONJUNTO DE INSTRUCCIONES DEL 8087 DE INTEL

Lo que resta de esta sección se diseñó como una referencia al conjunto de instrucciones del 8087. Cada instrucción se describe en orden numérico ascendente. Se proporciona la siguiente información para cada instrucción:

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

Estado afectado. La mayoría de las instrucciones modifica la palabra de estado de una u otra forma. Si la palabra de estado no se ve afectada, no se incluye esta categoría para la instrucción.

Ejemplos de codificación. Se dan ejemplos breves del uso de la instrucción. El ejemplo se incluye únicamente si el nombre de la instrucción no se usa solo, es decir, si se requieren operandos o instrucciones adicionales para que la instrucción trabaje.

F2XM1

Trascendental

2x-1: Calcula $y=2^X-1$, donde X es el elemento tope de la pila (ST). El resultado (Y) reemplaza a X como el elemento tope de la pila (ST). Esta instrucción no verifica la validación del valor de entrada. El Programa asegura que 0<=X<=0.5.

Estados afectados

PE;UE

FABS

Aritmética

Absolute Value

(Valor absoluto): Cambia el elemento tope de la pila (ST) a su valor absoluto.

Estado afectado
IE

FADD

Aritmética

Add Real

(Suma real): Suma dos números y los almacena en el operando de destino Si no se proporciona operando de destino (sólo se especifica un operando), se supone que ST es el destino.

Estados afectados
PE, UF, OF, DE, LE

Ejemplos de codificación
```
FADD TEMP       ;ST=ST+TEMP
FADD TEM, ST (3);TEMP=TEMP+ST(3)
```

FADDP

Aritmética

Add Real and pop
(Suma real y extracción): suma dos números, los almacena en el operando de destino y hace una extracción de la pila. Si no se da ningún operando de destino (sólo se especifica un operando), se supone que ST es el destino.

Estados afectados

PE, UE, OF, DE, LE

Ejemplos de codificación
```
FADDP TEMP         ;ST=ST+TENP
FADDP TEMP, ST (3);TEMP=TEMP+ST(3)
```

FBLD

Transferencia de datos

BCD Load

(Cargar BCD): Convierte el número BCD de la dirección del operando en un real temporal y lo mete en la pila.

Estado afectado
IE

Ejemplo de codificación
```
FBLD TEMP
```

FBSTP

Transferencia de datos

BCD Store and Pop

(Almacenar y sacar BCD): Convierte el elemento tope de la pila (ST) en un entero BCD, lo almacena en la dirección del operando y lo saca de la pila.

Estado afectado
IE
Ejemplo de codificación
```
FBSTP TEMP
```

FCHS

Aritmética

Change Sign

(Cambiar signo): Cambia el signo del elemento tope de la pila (ST).

Estado Afectado
IE

FCLEX

Control de procesador

Clear Exceptions with WAlT

(Limpiar excepciones con WAlT): Limpia las banderas de excepción, la de petición de interrupción y la de ocupado de la palabra de estado del 8087. Esta instrucción es precedida por un prefijo de espera de la UCP. Véase también FNCLEX.

Estados afectados
B, IR, PE, UE, OE, ZE, DE, IE

FCOM

Comparación

Compare Real

(Comparación real): El elemento tope de la pila (ST) se compara con el segundo elemento de la pila (ST(1)) o con otro operando especificado. Los códigos de condición se afectan en consecuencia.

Estados Afectados
C3, C2, CO, DE, IE
Ejemplos de codificación

```
FCOM                ;Compara ST con ST(1)
FCOM ST (4)         ;Compara ST con ST(4)
FCOM TEMP          ;Compara ST con la memoria
```

FCOMP

Compare Real and Pop

Comparación

(Comparación real y extracción): El elemento tope de la pila (ST) se compara con el segundo elemento de la pila (ST(1)) o con otro operando especificado; se hace una extracción de la pila. Los códigos de condición se afectan en consecuencia.

ESTADOS AFECTADOS
C3, C2, CO, DE, IE

Ejemplos de codificación

```
FCOMP       ;Comparar 5 con ST (1)
FCOMP ST(4);Comparar ST con ST (4)
FCOMP TEMP ;Comparar ST con la memoria
```

FCOMPP

Comparación

Compare Real and Pop Twice

(Comparación real y extracción dos veces): El elemento tope de la pila (ST) se compara con el segundo elemento de ésta (ST(1)) y se hacen dos extracciones de la pila. Los códigos de condición se afectan en consecuencia.

Estados afectados

C3, C2, CO, DE, IE

FDECSTP

Control de procesador

Decrement Stack Pointer

(Disminuir el apuntador de la pila): Disminuye el apuntador de la pila de la palabra de estado del 8087.

Estado afectado

ST

FDISI

Control de procesador

Disable Jnterrupts with WAlT

(No habilitar interrupciones con WAlT): Activa la máscara de habilitación de interrupciones de la palabra de control del 8087, impidiendo así que el 8087 inicie una interrupción. Esta instrucción es precedida por un prefijo de espera de la UCP. Véase también FNDISI.

MFDIV

Aritmética

Divide Real

(División real): Divide el operando de destino entre el operando fuente, y almacena el resultado en el operando de destino. Si no se proporciona operando de destino (sólo se especifica un operando), se supone que ST es el destino.

Estados afectados
PE,UE,OE,ZE,DE,IE

Ejemplos de codificación
```
FDIV TEMP ;ST=TEMP/ST
FDIV TEMP, ST (3)   ;TEMP=ST (3)/TEMP
```

FDIVP

Aritmética

Divide Real and Pop

(División real y extracción): Divide el operando de destino entre el operando fuente, almacena el resultado en el operando de destino y hace una extracción de la pila. Si no se proporciona operando de destino (sólo se especifica un operando), se supone que ST es el destino.

Estados afectados
PE, UF, OE, ZE, DE, IE

Ejemplos de codificación
```
DIV TEMP          ; ST=TEMP/ST
FDIV TEMP, ST (3); TEMP=ST(3)/TEMP
```

FDIVR

Aritmética

Divide Real Reversed

(División real invertida): Divide el operando fuente entre el operando de destino y almacena el resultado en este último. Si no se proporciona operando de destino (sólo se especifica un operando), se supone que ST es el destino.

Estados afectados
PE, UF, OF, ZF, DE, IE

Ejemplos de codificación
```
FDIVR TEMP     ;ST=ST/TEMP
FDIVR TFMP, ST (3) ;TEMP=TEMP IST (3)
```

FDIRVRP

Aritmética

Divide Real Reversed and Pop

(División real invertida y extracción): Divide el operando fuente entre el operando de destino, almacena el resultado en este último y hace una extracción de la pila. Si no se proporciona operando de destino (sólo se especifica un operando), se supone que ST es el destino.

Estados afectados
PE, UF, OF, ZF, DF, IE

Ejemplos de codificación

```
FDIVRP TEMP              ;ST=ST/TEMP
FDIYRP TEMP, ST (3)      ;TEMP=TEMP/ST (3)
```

FENI

Control de procesador

Enable interrupts with WAIT

(Habilitar interrupciones con WAlT): Limpia la máscara de habilitación de interrupción de la palabra de control del 8087, permitiendo así que el 8087 inicie interrupciones. Esta instrucción es precedida por un prefijo de espera de la UCP. Véase también FNENI.

FFREE

Control de proesador

Free Register

(Liberar el registro): Cambia la etiqueta del registro de pila especificado para indicar que el registro de la pila está vacío.

Ejemplo de codificación

```
FFREE ST (3)
```

FIADD

Aritmética

Integer Add

(Sumar enteros): Sumados números como enteros y los almacena en él operando de destino. Si no se proporciona operando de destino (sólo se especifica un operando), se supone que ST es el destino.

Estados afectados
PE, OF, DE, IE

Ejemplos de codificación

```
FIADD TEMP              ;ST=ST+TEMP
FIADD TÉMP, ST (3)      ;TEMP=TEMP+ST (3)
```

FICOM

Comparación

Integer Compare

(Comparar enteros): Convierte el operando (que se supone es un entero) en un real temporal y lo compara con el elemento tope de la pila (ST). Los códigos de condición se activan como corresponde.

Estados afectados
C3, C2, C0, DE, IE

Ejemplo de codificación

```
FICOM TEMP_INT ;Comparar la memoria con ST
```

FICOMP

Comparación

Integer Compare and Pop

(Comparación de enteros y extracción): Convierte el operando (que se supone un entero) en un real temporal, lo compara con el elemento tope de la pila (ST), y luego hace una extracción de la pila. Los códigos de condición se activan como corresponde.

Estados afectados
C3, C2, C0, DE, IE

EJEMPLO DE CODIFICAION
FICOMP TEMP_INT ;Comparar la memoria con ST

FIDIV

Aritmética

Integer Divide

(Dividir enteros): Divide el destino entre el operando fuente, como enteros, y almacena el resultado en el operando de destino. Si no se proporciona operando de destino (sólo se especifica un operando), se supone que ST es el destino.

Estados afectados
PE, UF, OF, ZE, DF, IE

Ejemplo de codificación
```
FIDIV TEMP          ;ST=TEMP/ST
FIDIV TEMP ST (3) ;TEMP=ST (3) ITEMP
```

FIDIVR

Aritmética

Integer Divide Reversed

(División de enteros en forma invertida): Divide la fuente entre el operando de destino, como enteros, y almacena el resultado en el operando de destino. Si no se proporciona operando de destino (sólo se especifica un operando), se supone que ST es el destino.

Estados afectados
PE, UE, OE, ZE, DE, IE

Ejemplos de codificación
```
FIDIVR TEMP    ;ST=ST/TEMP
FIDIVR TEMP;ST (3) ;TEMP=TEMP/ST (3)
```

FILD

Transferencia de datos

Integer Load

(Cargar un entero): Convierte el número entero binario de la dirección del operando en un real temporal y lo mete en la pila.

Estado afectado
IE

Ejemplo de codificación
FILD TEMP

FIMUL

Aritmética

Integer Multiply

(Multiplicación de enteros): Multiplica la fuente por el operando de destino, como enteros, y almacena el resultado en el operando de destino. Si no se proporciona operando de destino (sólo se especifica un operando), se supone que ST es el destino.

Estados afectados
PE, OE, DE, IE

Ejemplos de codificación
```
FIMUL TEMP     ;ST=ST*TEMP
FIMUL TEMP, ST (3) ;TEMP=TEMP*ST (3)
```

FINCSTP

Control de procesador

Increment Stack Pointer

(Incrementar el apuntador de la pila): Incrementa el apuntador de la pila de la palabra de estado del 8087.

Estado afectado
ST

FINIT

Control de procesador

Initialize Processor with WAIT

(Iniciar el procesador con WAIT): Inicia el 8087. Esta acción es funcionalmente equivalente a realizar un RESET de hardware. Esta instrucción es precedida por un prefijo de espera (WAIT) de la UCP. Véase también FNINIT.

FIST

Transferencia de datos

Integer Store

(Almacenar un entero) Redondea el elemento tope de la pila (ST) a un nmero entero binario y lo almacena en la dirección del operando

Estados afectados
PE, IE

Ejemplo de codificación
FIST TEMP

FLD

Transferencia de datos

Load Real

(Cargar real): Mete el valor del operando fuente en la pila.

Estados afectados
DE, IE

Ejemplos de codificación
FLD ST (3) FLD TEMP

FLD1

Constante

Load 1.0

(Cargar 1.0): Mete el valor +1 en la pila. este valor se convierte en ST.

Estado afectado
IE

FLDCW

Control de procesador

Load Control Word

(Cargar la palabra de control): Carga la palabra de control del 8087 con el valor de palabra apuntado por el operando fuente.

Ejemplo de codificación

```
FLDCW MEM_CW   ;Transferir la palabra de control
```

FLDENV

Control de procesador

Load Environment

(Cargar el ambiente): Restablece todas las variables de ambiente del 8087 a partir de la localidad de memoria de 14 palabras especificada por el operando.

Estados afectados

B, C3, ST, C2, C1, CO, IR, PE, UE, OE, ZE, DE, IE

Ejemplo de codificación

```
FLDENV SAVE_AREA
```

FLDL2E

Constante

Load $\log_2 e$

(Cargar $\log_2 e$): Mete el valor de $LOG_2 e$ en la pila. Este valor se convierte en ST.

Estado afectado

IE

FLDL2T

Constante

Load $\log_2 10$

(Cargar $\log_2 10$): Mete el valor de LOG_2 10 en la pila. Este valor se convierte en ST.

Estado afectado

IE

FLDLG2

Constante

Load $\log_{10} 2$

(Cargar $\log_{10} 2$): Mete el valor del $LOG_{10} 2$ en la pila. Este valor se convierte en ST.

Estado afectado

IE

FLDLN2

Constante

Load $\log_e 2$

(Cargar log_e2): Mete el valor del LOG_e2 en la pila. Este valor se convierte en ST.

Estado Afectado

IE

FLDPI

Constante

Load Pi

(Cargar pi): Mete el valor de pi en la pila. Este valor se convierte en ST.

Estado afectado

IE

FLDZ

Constante

Load 0.0

(Cargar 0.0): Mete el valor 0.0 en la pila. Este valor se convierte en ST.

Estado Afectado

IE

FMUL

Aritmética

Multiply Real

(Multiplicación real): Multiplica el operando fuente por el operando de destino y almacena el resultado en este último. Si no se proporciona operando de destino (sólo se especifica un operando), se supone que ST es el destino.

Estados afectados

PE, UE, OE, DE, IE

Ejemplos de Codificación

```
FMUL TEMP           ;ST=ST*TEMP

FMUL TEMP, ST (3)   ;TEMP=TEMP*ST (3)
```

FMULP

Aritmética

Multiply Real and Pop

(Multiplicación real y extracción): Multiplica el operando fuente por el operando de destino, almacena el resultado en éste último y hace una extracción de la pila. Si no se proporciona operando de destino (sólo se especifica un operando), se supone que ST es el destino.

Estados afectados

PE,UE,OE,DE,IE

Ejemplos de codificación

```
FMULP TEMP          ;ST=ST*TEMP

FMULP TEMP, ST (3) ;TEMP=TEMP*ST (3)
```

FNCLEX

Control de procesador

Clear Exceptions

(Limpiar excepciones): Limpia las banderas de excepción, la de petición de interrupción y la de ocupado de la palabra del estado del 8087. Esta instrucción no es precedida por un prefijo de espera de la UCP. Véase también FCLEX.

Estados afectados

B, IR, PE, UE, OE, ZE, DE, IE

FNDISI

Control de Procesador

Disable Interrupts

(Impedir interrupciones): Activa la máséara de habilitación de interrupción de la palabra de control del 8087, impidiendo así que el 8087 inicie una interrupción. Esta instrucción no es precedida por un prefijo de espera de la UCP. Véase también FDISI.

FNENI

Control de procesador

Enable Interrupts

(Habilitar interrupciones): Limpia la máscara de habilitación de interrupción de la palabra de control del 8087, permitiendo así que el 8087 inicie interrupciones. Esta instrucción no es precedida por un prefijo de espera de la UCP. Véase también FENI.

FNINIT

Control de procesador

Initialize Processor

(Iniciar el procesador): Inicia el 8087. Esta acción es funcionalmente equivalente a realizar un RESET de hardware. Esta instrucción no es precedida por un prefijo de espera de la UCP. Véase también FINIT.

FNOP

Control de procesador

No Operation

(Ninguna operación): No hace nada más que tomar tiempo y espacio: el 8087 no realiza ninguna operación.

FNSAVE

Control de procesado,

Save State

(Grabar estado): Graba, en la localidad de memoria especificada por el operando, todos los registros y variables de ambiente del 8087. Esta acción requiere 94 palabras de memoria. Después de ella, el 8087 se inicia como si se hubieran expedido las instrucciones FINITo FNINIT. Esta instrucción (FNSAVE) no es precedida por un prefijo de espera de la UCP. Véase también FSAVE.

Ejemplo de codificación

FNSAVE SAVE_AREA

FNSTCW

Control de procesador

Store Control Word

(Almacenar la palabra de control): Copia la palabra de control del 8087 en el valor apuntado por el operando fuente. Esta instrucción no es precedida por un prefijo de espera de la UCP. Véase también FSTCW.

Ejemplo de codificación.

FNSTCW MEM_CW ;Transferir la palabra de control

FNSTENV

Control de procesador

Store Environment

(Almacenar el ambiente): Guarda, en la localidad de memoria especificada por el operando, todas las variables de ambiente del 8087. Ello requiere 14 palabras de memoria. Después de dicha acción, esta instrucción activa las máscaras de excepción de la palabra de control del 8087. Esta instrucción no es precedida por un prefijo de espera de la UCP. Véase también FSTENV.

Ejemplo de codificación

FNSTENV SAVE_AREA

FNSTSW

Control de procesador

Store Status Word

(Almacenar la palabra de control): Copia la palabra de estado del 8087 en el valor de palabra apuntado por él operando fuente. Esta instrucción no es precedida por un prefijo de espera de la UCP. Véase tambén FSTSW.

Ejmplo de Codficación

FNSTSW MEM_SW ;Transferir la palabra de control

FPATAN:

Trascendental

Partial Arctangent (Arcotangente parcial): Calcula q=ARCTAN(Y/X), donde X es el elemento tope de la pila (ST) y Y es el segundo elemento de ésta (ST(l)). Ambos elementos se sacan y el resultado (0) se mete en la pila y se convierte en ST. Esta instrucción no verifica la validación del valor de entrada. El programa asegura que $0<Y<X<\infty$

Estados afectados

PE, UE

FPRENM

Aritmética

Partial Remainder

(Residuo parcial): Calcula el módulo de los dos elementos tope de la pila. Restando sucesivamente ST(1 de ST, se calcula un residuo exacto que perrnanece en ST.

Estados afectados

C3, Cl, C0, UE, DE, IE

FPTAN

Trascendental

Partial Tangent

(Tangente parcial):Calcula Y/X=TAN(Z), donde Z es el elemento tope de la pila (ST). Este elemento es reemplazado por el Y calculado, mientras que el X calculado se mete en la pila. Por lo tanto, al final de esta operación, ST (1)=Y y ST=X. Esta instrucción no verifica la validación del valor de entrada.

Estados afectados

PE, IE

FRNDINT

Aritmética

Round to Integer

(Redondear a entero): Redondea el número del elemento tope de la pila (ST) a un entero.

Estados afectados

PE, IE