
# “INSTRUCTION SET DE MICROPROCESADOR 8086”

**JOSE LUIS SANDOVAL PEREZ**

**Ingeniería en Computacion inteligente**

**7-A**

02/10/2024

## Substract with borrow (Resta con préstamo)

Resta el origen del destino, resta uno si la bandera de acarreo esta activa, y regresa el resultado al operador de destino. Ambos operadores deben de ser 'bytes' o palabras. Ambos operadores pueden ser números binarios con o sin signo. La resta con préstamo actualiza la *bandera de acarreo auxiliar, bandera de acarreo, bandera de desbordamiento, bandera de paridad, bandera de signo y bandera de cero.* Dado que incorpora un préstamo de una operación anterior, la resta con préstamo puede utilizarse para escribir instrucciones que resten números de más de 16 bits.

## Decimal adjust for substraction (Ajuste decimal para resta)

Corrige el resultado de una resta anterior de dos operandos decimales empaquetados (el operador de destino debido haber sido especificado como un registro AL). El ajuste decimal para la resta cambia el contenido del registro del acumulador a un par de paquetes de dígitos decimales válidos. El ajuste decimal para resta actualiza la bandera de acarreo auxiliar, la bandera de acarreo, bandera de paridad, bandera de signo y bandera de cero; el contenido de la bandera de desbordamiento es indefinido al completar esta operación.

## Add with Carry (Suma con acarreo)

Suma de operandos los cuales pueden ser bytes o palabras, agrega 1 si la bandera de acarreo esta activa y remplaza el operando de destino con el resultado. Ambos operandos pueden ser números binarios con o sin signo. La suma con acarreo actualiza las banderas auxiliares de aparidad, banderas de acarreo, banderas de desbordamiento, banderas de signo y banderas de cero. Dado que incorpora un acarreo de una operación anterior, la suma de acarreo puede utilizarse para escribir instrucciones que resten números de más de 16 bits.

## Compare (Comparación)

CMP (Comparar) resta la fuente del destino, que pueden ser bytes o palabras, pero no devuelve el resultado. Los operandos no cambian, pero las banderas se actualizan y se pueden probar mediante una instrucción de salto condicional posterior. CMP actualiza bandera auxiliar, bandera de acarreo, bandera de desbordamiento, bandera de paridad, bandera de signo y bandera de cero. La comparación reflejada en las banderas es la del destino a la fuente. Si una instrucción CMP es seguida por una instrucción JG (salto si es mayor), por ejemplo, el salto se toma si el operando de destino es mayor que el operando de origen.

## LODS String (Carga de cadena de texto)

Transfiere el byte o el elemento de cadena de palabras dirigido por SI para registrar AL o AX, y actualiza SI para apuntar al siguiente elemento de la cadena. Esta instrucción no se repite normalmente, ya que el acumulador se sobrescribiría con cada repetición, y solo se conservaría el último elemento. Sin embargo, LODS es muy útil en bucles de software como parte de una función de cadena más compleja construida a partir de cadenas primitivas y otras instrucciones.

## LEA Load Effective Address

LEA (dirección efectiva de carga) transfiere el desplazamiento del operando de origen (en lugar de su valor) al operando de destino. El operando de origen debe ser un operando de memoria, y el operando de destino debe ser un registro general de 16 bits. LEA no afecta a ninguna bandera. Las instrucciones XLAT y de cadena asumen que ciertos registros apuntan a operandos; LEA se puede usar para cargar estos registros (por ejemplo, cargando BX con la dirección de la tabla de traducción utilizada por la instrucción XLAT).

## IN

IN transfiere un byte o una palabra de un puerto de entrada al registro AL o al registro AX, respectivamente. El número de puerto se puede especificar con una constante de bytes inmediata, permitiendo el acceso a puertos numerados de O a 255, o con un número previamente colocado en el registro DX, permitiendo el acceso variable (cambiando el valor en DX) a puertos numerados de 0 a 65.535.

## OUT

OUT transfiere un byte o una palabra del registro AL o del registro AX, respectivamente, a un puerto de salida. El número de puerto se puede especificar con una constante de byte inmediata, permitiendo el acceso a puertos numerados del 0 al 255, o con un número previamente colocado en el registro DX, permitiendo el acceso variable (cambiando el valor en DX) a los puertos numerados del 0 al 65.535.

## AAM

AAM (Ajuste ASCII al multiplicar) corrige el resultado de una multiplicación previa de dos operandos decimales desempaquetados válidos. El número decimal desempaquetado de dos digitos se deriva del contenido de AH y AL y se devuelve a AH y AL. Los medios bytes de alto orden de los operandos multiplicados deben haber sido OH para que AAM produzca un resultado correcto. AAM actualiza PF, SF y ZF; el contenido de AF, CF y OF es indefinido después de la ejecución de AAM.

## XLAT

XLAT (traducir por tabla) reemplaza un byte en el registro AL con un byte de una tabla de traducción de 256 bytes codificada por el usuario. Se supone que el registro BX apunta al principio de la tabla. El byte en AL se utiliza como un índice en la tabla y se reemplaza por el byte en el desplazamiento en la tabla correspondiente al valor binario de AL. El primer byte de la tabla tiene un desplazamiento de 0. Por ejemplo, si AL contiene SH, y el sexto elemento de la tabla de traducción contiene 33H, entonces AL contendrá 33H siguiendo la instrucción. XLAT es útil para traducir caracteres de un código a otro, siendo el ejemplo clásico ASCII a EBCDIC o al revés.

## LDS

LDS (puntero de carga usando DS) transfiere una variable de puntero de 32 bits desde el operando de origen, que debe ser un operando de memoria, al operando de destino y al DS de registro. La palabra de desplazamiento del puntero se transfiere al operando

de destino, que puede ser cualquier registro general de 16 bits. La palabra de segmento del puntero se transfiere al registro DS. Especificar SI como operando de destino es una forma conveniente de prepararse para procesar una cadena de origen que no está en el segmento de datos actual (las instrucciones de cadena asumen que la cadena de origen se encuentra en el segmento de datos actual y que SI contiene el desplazamiento de la cadena).

## LES

LES (puntero de carga usando ES) transfiere una variable de puntero de 32 bits desde el operando de origen, que debe ser un operando de memoria, al operando de destino y al ES de registro. La palabra de desplazamiento del puntero se transfiere al operando de destino, que puede ser cualquier registro general de 16 bits. La palabra de segmento del puntero se transfiere al registro ES. Especificar DI como operando de destino es una forma conveniente de prepararse para procesar una cadena de destino que no está en el segmento adicional actual.

La cadena de destino debe estar ubicada en el segmento adicional, y DI debe contener el desplazamiento de la cadena)

## POPF

POPF transfiere bits específicos de la palabra en la parte superior actual de la pila (apuntada por el registro SP) a las banderas 8086/8088, reemplazando cualquier valor que las banderas contuvieran previamente (ver figura 2-32). Luego, el SP se incrementa en dos para apuntar a la nueva parte superior de la pila. PUSHF y POPF permiten un procedimiento para guardar y restaurar las banderas de un programa de llamada. También permiten que un programa cambie la configuración de TF (no hay instrucciones para actualizar esta bandera directamente). El cambio se logra empujando las banderas, alterando el bit 8 de la imagen de memoria y luego haciendo estallar las banderas.

## NEG

NEG (Negacion) resta el operando de destino, que puede ser un byte o una palabra, de 0 y devuelve el resultado al destino. Esto forma el complemento del número de los dos, invirtiendo efectivamente el signo de un número entero. Si el operando es cero, su signo no se cambia. Intentar negar un byte que contenga -128 o una palabra que contenga -32.768 no causa ningún cambio en el operando y los conjuntos OF. NEG actualiza AF, CF, OF, PF, SF y ZF. CF siempre se establece excepto cuando el operando es cero, en cuyo caso se borra.

## MUL

MUL (Multiplicar) realiza una multiplicación sin signo del operando de origen y el acumulador. Si la fuente es un byte, entonces se multiplica por el registro AL, y el resultado de doble longitud se devuelve en AH y AL. Si el operando de origen es una palabra, entonces se multiplica por el registro AX, y la retención de doble longitud se

devuelve en los registros DX y AX. Los operandos se tratan como números binarios sin signo (ver A, AM). Si la mitad superior del resultado (AH para la fuente de bytes, DX para la fuente de la palabra) no es cero. CF y OF están configurados: de lo contrario, están despejados. Cuando se establecen CE y OF, indican que AH o DX contiene dígitos significativos del resultado. El contenido de AF, PF, SE y ZF no está definido después de la ejecución de MUL.

## IMUL

IMUL (Multiplicación de Enteros / signo) realiza una multiplicación con signo del operando de origen y el acumulador. Si la fuente es un byte, entonces se multiplica por el registro AL, y el resultado de doble longitud se devuelve en AH y AL. Si la fuente es una palabra, entonces se multiplica por el registro AX, y el resultado de doble longitud se devuelve en los registros DX y AX. Si la mitad superior del resultado (AH para fuente de bytes, DX para fuente de palabra) no es la extensión de signo de la mitad inferior del resultado, CF y OF se establecen; de lo contrario, se borran. Cuando se establecen CF y OF, indican que AH o DX contiene dígitos significativos del resultado. El contenido de AF, PF, SF y ZF no está definido después de la ejecución de IMUL.

## SAR

SAR (Desplazamiento aritmetico a la derecha) mueve los bits en el operando de destino (byte o palabra) a la derecha por el número de bits especificados en el operando de conteo.

Los bits iguales al bit original de alto orden (signo) se desplazan a la izquierda, preservando el signo del valor original. Tenga en cuenta que SAR no produce el mismo resultado que el dividendo de una Instrucción IDIV *equivalente" si el operando de destino es negativo y se desplazan 1 bits. Por ejemplo, cambiar -5 a la derecha por un bit produce -3, mientras que la división entera de -5 por 2 produce -2. La diferencia en las instrucciones es que IDIV trunca todos los números hacia cero, mientras que SAR trunca los números positivos hacia cero y los números negativos hacia el infinito negativo.

## RCL

RCL (Rotar a través del acarreo hacia la izquierda) rota los bits en el byte o operando de destino de palabra hacia la izquierda por el número de bits especificados en el operando de conteo. La bandera de transporte (CF) se trata como "parte de" del operando de destino; es decir, su valor se gira en el bit de orden bajo del destino, y él mismo es reemplazado por el bit de orden alto del destino.

## SCAS

SCAS (Escanear cadena) resta el elemento de cadena de destino (byte o palabra) dirigido por DI del contenido de AL (cadena de bytes) o AX (cadena de palabras) y actualiza las banderas, pero no altera la cadena de destino o el acumulador. SCAS también actualiza DI para apuntar al siguiente elemento de cadena y AF, CF, OF, PF, SF y ZF para reflejar la relación del valor de escaneo en AL/AX con el elemento de cadena. Si SCAS está prefijado con REPE o REP, la operación se interpreta como "escanear mientras no sea el final de la cadena (CX no 0) y string-element =Valor de escaneo (ZF = 1)". Este formulario se puede utilizar para escanear la salida de un valor dado. Si SCAS está prefijado con REPNE o REPNZ, la operación se interpreta como "escanear mientras no sea el final de la cadena (CX no 0) y el elemento de cadena no es igual a valor de escaneo (ZF = 0)". Este formulario se puede utilizar para localizar un valor en una cadena.

## RET

RET (Retorno) transfiere el control de un procedimiento de vuelta a la instrucción después de la LLAMADA que activó el procedimiento. El ensamblador genera un RET intrasegmento si el programador ha definido el procedimiento NEAR, o un RET intersegmento si el procedimiento se ha definido como FAR. RET pone la palabra en la parte superior de la pila (apuntada por el registro SP) en el puntero de instrucciones e incrementa SP en dos. Si RET es intersegmento, la palabra en la nueva parte superior de la pila aparece en el registro CS, y SP se incrementa de nuevo en dos. Si se ha especificado un valor pop opcional, RET añade ese valor a SP. Esta función se puede utilizar para descartar parámetros empujados a la pila antes de la ejecución de la instrucción CALL.

## INTO

INTO (Interrupcion en desbordamiento) genera una interrupción de software si se establece la bandera de desbordamiento (OF); de lo contrario, el control procede a la siguiente instrucción sin activar un procedimiento de interrupción. INTO aborda el procedimiento de interrupción de destino (su tipo es 4) a través del puntero de interrupción en la ubicación 10H; borra las banderas TF e IF y, de lo contrario, funciona como INT. INTO puede escribirse después de una operación aritmética o lógica para activar un procedimiento de interrupción si se produce un desbordamiento.

## CLI

CLI (Limpiar bandera de interrupcion) zeroes IF.

Cuando se borra el indicador de interrupt-enable, el 8086 y el 8088 no reconocen una solicitud de interrupción externa que aparece en la línea INTR; en otras palabras, se deshabilitan las interrupciones enmascarables. Sin embargo, se respeta una interrupción

no enmascarable que aparece en la línea NMI, sin embargo, se respeta como una interrupción de software. CLI no afecta a ninguna otra bandera.

## HLT

HLT (Detener) hace que el 8086/8088 entre en el estado de parada. El procesador deja el estado de parada al activar la línea RESET, al recibir una solicitud de interrupción no enmascarable en NMI, o, si las interrupciones están habilitadas, al recibir una solicitud de interrupción enmascarable en INTR. HLT no afecta a ninguna bandera. Se puede usar como una alternativa a un bucle de software interminable en situaciones en las que un programa debe esperar una interrupción.

## ESC

ESC (Escape) proporciona un medio para que un procesador externo obtenga un código de operación y posiblemente un operando de memoria del 8086 o 8088. El código de operación externo es una constante inmediata de 6 bits que el ensamblador codifica en la instrucción de la máquina y que se construye (ver tabla 2-26). Un procesador externo puede monitorear el bus del sistema y capturar este código de operación cuando se busca el ESC. Si el operando de origen es un registro, el procesador no hace nada.

Si el operando de origen es una variable de memoria, el procesador obtiene el operando de la memoria y lo descarta. Un procesador externo puede capturar el operando de memoria cuando el procesador lo lee desde la memoria.

## IRET

IRET (Retorno de interrupcion) transfiere el control de vuelta al punto de interrupción haciendo estallar IP, CS y las banderas de la pila. Por lo tanto, IRET afecta a todas las banderas al restaurarlas a valores previamente guardados. IRET se utiliza para salir de cualquier procedimiento de interrupción, ya sea activado por hardware o software.