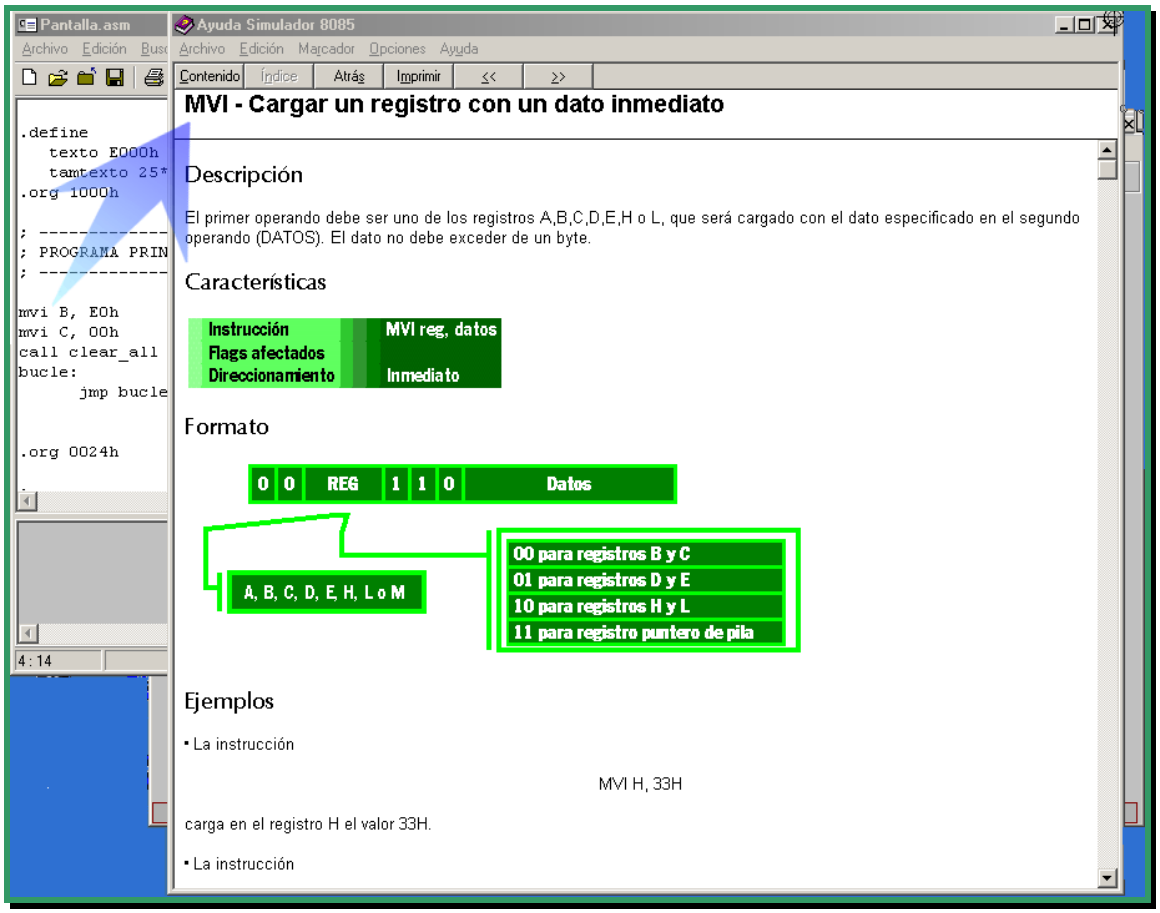
A1

Apéndice A1
Conjunto de instrucciones del 8085

A1.1. Introducción

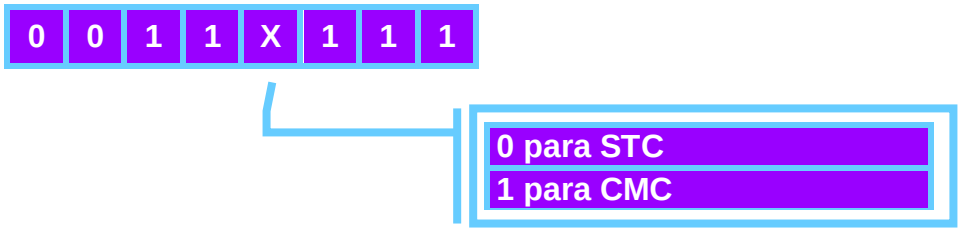
Aunque el conjunto de instrucciones puede encontrarse en cualquier libro que trate este tema específicamente, en este apartado se muestra un "diccionario" de las mismas. La descripción se lista funcionalmente para una localización más inmediata y todas están descritas de forma detallada.

Si desea hacer una búsqueda alfabética de las instrucciones puede usar la ayuda incorporada en el programa simulador.



A1.2. Instrucciones del bit de acarreo

A continuación se describen las instrucciones que operan directamente sobre el bit de acarreo. Estas instrucciones utilizan un byte en la forma siguiente:



A1.2.1. CMC Complementar acarreo

Instrucción	CMC
Bits afectados	CY
Direccionamiento	

Descripción

Si el bit de acarreo es 0, se pone a 1. Si es un 1, se pone a 0.

Formato

A1.2.2. STC Activar acarreo

Instrucción	STC
Bits afectados	CY
Direccionamiento	

Descripción

El bit de acarreo se pone a 1.

Formato

A1.3. Instrucciones de un registro

A continuación se describen las instrucciones que operan sobre un solo registro o posición de memoria. Si se especifica una referencia a memoria, la dirección de la misma viene dada por el contenido de los registros H y L, donde el registro H contiene los ocho bits más significativos de la dirección, y el registro L los restantes.

A1.3.1. INR Incrementar registro o memoria

Instrucción	INR reg
Bits afectados	Z, S, P, AC
Direccionamiento	Registro indirecto

Descripción

El contenido del registro o posición de memoria especificados se incrementa en una unidad.

Formato

Ejemplo

Si el registro A contiene 98H, la instrucción

INR A

hará que este registro contenga la cantidad 99H.

A1.3.2. DCR Decrementa registro o memoria

Instrucción	DCR reg
Bits afectados	Z, S, P, AC
Direccionamiento	Registro

Descripción

El contenido del registro o posición de memoria especificados se decrementa en una unidad.

Formato

Ejemplo

Si el registro A contiene 99H, la instrucción

DCR A

hará que este registro contenga la cantidad 98H.

A1.3.3. CMA Complementar acumulador

Instrucción	CMA
Bits afectados	
Direccionamiento	

Descripción

Cada uno de los bits del acumulador se complementa (operación denominada a uno).

Formato

0	0	1	0	1	1	1	1

A1.3.4. DAA Ajuste decimal del acumulador

Instrucción	DAA
Bits afectados	Z, S, P, CY, AC
Direccionamiento	Registro

Descripción

El número hexadecimal de 8 bits contenido en el acumulador se ajusta como dos dígitos de 4 bits codificados en binario decimal, según el proceso siguiente:

(1). Si los cuatro bits menos significativos del acumulador representan un número mayor que 9, o si el bit de acarreo auxiliar es igual a uno, el acumulador se incrementa en seis unidades. Si no se presentan tales condiciones, el contenido del acumulador no varía.

(2). Si los cuatro bits más significativos del acumulador representan ahora un número mayor que 9, o si el bit de acarreo es uno, los cuatro bits más significativos se incrementan en seis unidades. Asimismo, si no tienen lugar las circunstancias expuestas, el contenido del acumulador no se incrementa.

Si hay acarreo de los cuatro bits menos significativos durante el paso (1), el bit de acarreo auxiliar se pone a 1; si no lo hay, se pone a 0. Por otra parte, si hay acarreo de los cuatro bits más significativos durante el paso (2), se activará el bit de acarreo, poniéndose a cero si no se produce acarreo.

Formato

Nota

Esta instrucción se utiliza en las operaciones de suma de números decimales. Es la única instrucción cuya operación depende del bit de acarreo auxiliar.

Ejemplo

Supongamos que queremos realizar una suma decimal de dos números (40 + 80). Ambos bits de acarreo están a cero.

La secuencia de instrucciones podría ser:

```
(1). MVI   B,80H
(2). MVI   A,40H        ; Acumulador = 40H
(3). ADD   B            ; Acumulador = 40H + 80H = C0H
(4). DAA                ; Acumulador = 20H
                        ; Bit de acarreo = 1
```

La instrucción DAA opera de la siguiente forma:

1. Como el contenido de los bits [0 – 3] del acumulador es menor que 9 y el bit de acarreo auxiliar es cero, el contenido del acumulador no varía.

2. Como los 4 bits más significativos del acumulador representan un número mayor que 9, estos 4 bits se incrementan en 6 unidades, poniendo a uno el bit de

acarreo.

Acumulador	C0H	1	1	0	0	0	0	0	0	CY = 0
+ 6	60H	0	1	1	0	0	0	0	0	CY = 0
Nuevo acumulador	20H	0	0	1	0	0	0	0	0	CY = 1

A1.4. Instrucción NOP

Instrucción	NOP
Bits afectados	
Direccionamiento	

Descripción

No se realiza ninguna operación.

Formato

A1.5. Instrucciones de transferencia de datos

Esta serie de instrucciones transfieren datos entre los registros, la memoria y el acumulador. Ocupan un byte en la forma siguiente:

A1.5.1. MOV Movimiento

Instrucción	MOV reg, reg
Bits afectados	
Direccionamiento	Registro o registro indirecto

Descripción

Podemos distinguir 3 casos:

(A). Transferencia entre registros (direccionamiento registro).

(B). Transferencia desde la memoria (direccionamiento registro indirecto).

(C). Transferencia a la memoria (direccionamiento registro indirecto).

(A). MOV R_1, R_2

El contenido del registro R_2 es transferido al registro R_1. R_1 y R_2 pueden ser los registros B, C, D, E, H, L o el acumulador A.

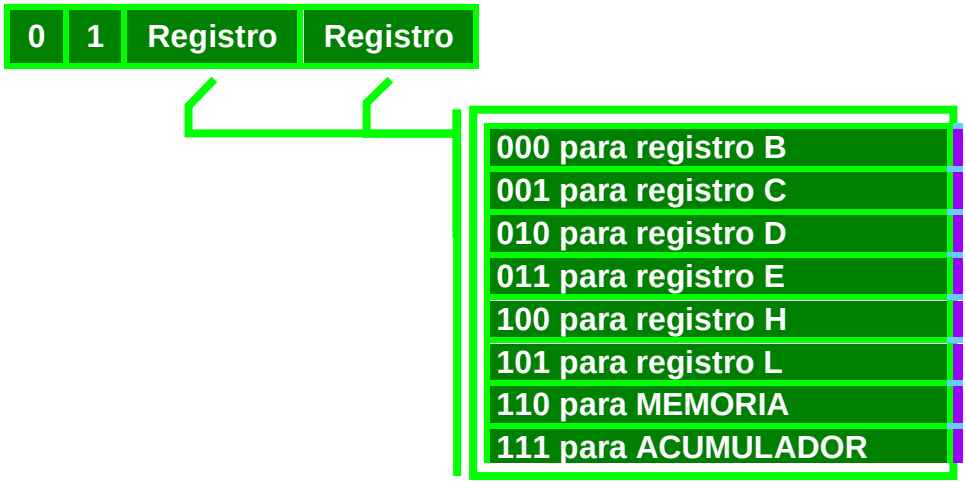
(B). MOV R, M

El contenido de la dirección de memoria, cuya dirección está en los registros H-L, es transferido al registro R. R puede ser cualquiera de los registros A, B, C, D, E, H o L.

(C). MOV M, R

El contenido del registro R es transferido a la dirección de memoria indicada por los registros H-L.

Formato



Ejemplos

1. Supongamos que el registro B contiene 00H y el registro C contiene 30H. La instrucción

   MOV B,C

   almacenará 30H en el registro B.

2. Supongamos que el registro H contiene 00H y el registro L contiene 30H. La instrucción

   MOV M, A

   almacenará el contenido del acumulador en la posición de memoria 0030H.

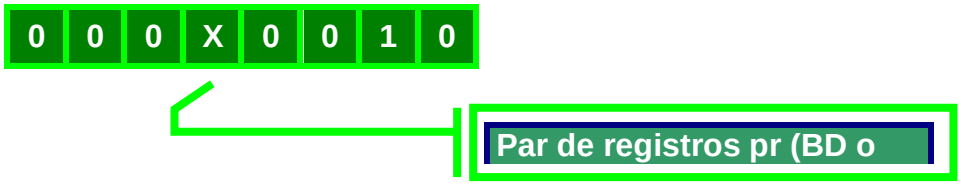
A1.5.2. STAX Almacenar el contenido del acumulador

Instrucción	STAX rp
Bits afectados	
Direccionamiento	Registro indirecto

Descripción

El contenido del acumulador se almacena en la posición de memoria especificada por los registros B y C, o los registros D y E.

Formato



Ejemplo

Si el registro B contiene 3FH y el registro C contiene 16H, la instrucción

STAX B

almacenará el contenido del acumulador en la posición de memoria 3F16H.

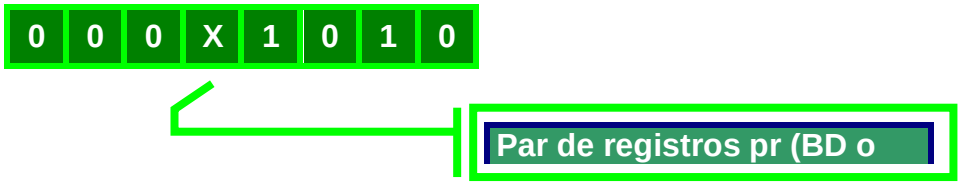
A1.5.3. LDAX Cargar el acumulador

Instrucción	LDAX rp
Bits afectados	
Direccionamiento	Registro indirecto

Descripción

El contenido de la posición de memoria especificada por los registros B y C, o los registros D y E, reemplaza el contenido del acumulador.

Formato



Ejemplo

Si el registro D contiene 3FH y el registro E contiene 16H, la instrucción

LDAX D

cargará en el acumulador el contenido de la posición de memoria 3F16H.

A1.6. Instrucciones de registro o memoria y acumulador

A continuación vamos a ver las instrucciones que operan con el contenido del acumulador y el de uno de los registros o posición de memoria. Estas instrucciones ocupan un byte en la forma siguiente:

La instrucción opera sobre el contenido del acumulador, con la cantidad definida por el registro especificado por REGISTRO. Si se especifica una referencia a memoria, la cantidad utilizada por la instrucción es la correspondiente a la posición de memoria determinada por los registros H y L, en los que el registro H guarda los 8 bits más significativos, y el registro L, los 8 restantes. Tanto el contenido del registro como de la posición de memoria no varían al finalizar la instrucción, guardándose el resultado en el acumulador.

A1.6.1. ADD Sumar registro o memoria al acumulador

Instrucción	ADD reg
Bits afectados	Z, S, P, CY, AC
Direccionamiento	Registro

Descripción

El contenido del registro o posición de memoria especificados se suma al contenido del

acumulador, usando aritmética de complemento a dos. El resultado se guarda en el acumulador.

Formato

1	0	0	0	0	REGISTRO

Ejemplos

1. Si el registro B contiene el valor 3AH y el acumulador contiene 6CH, la instrucción

   ADD B

   realiza la siguiente suma:

Registro B	3AH	0 0 1 1 1 0 1 0
Acumulador	6CH	0 1 1 0 1 1 0 0
Nuevo acumulador	A6H	1 0 1 0 0 1 1 0

2. La instrucción

   ADD A

   duplica el contenido del acumulador.

A1.6.2. ADC — Sumar registro o memoria al acumulador con acarreo

Instrucción	ADC reg
Bits afectados	Z, S, P, CY, AC
Direccionamiento	Registro indirecto

Descripción

El contenido del registro o posición de memoria especificados más el contenido del bit de acarreo, se suman al contenido del acumulador.

Formato

Ejemplo

Supongamos que el registro B contiene el valor 30H, el acumulador 76H, y el bit de acarreo está puesto a cero. La instrucción

ADC C

realizará la siguiente suma:

Registro B	30H	0 0 1 1 0 0 0 0
Acumulador	76H	0 1 1 1 0 1 1 0
Bit de acarreo		0
Nuevo acumulador	A6H	1 0 1 0 0 1 1 0

El nuevo contenido del acumulador será A6H, mientras que todos los bits de condición quedarán puestos a cero excepto los de signo y paridad.

Si el bit de acarreo hubiera sido 1 antes de realizar la operación, hubiera tenido lugar la siguiente suma:

Registro B	30H	0 0 1 1 0 0 0 0
Acumulador	76H	0 1 1 1 0 1 1 0
Bit de acarreo		1
Nuevo acumulador	A7H	1 0 1 0 0 1 1 1

El acumulador contendría ahora A7H y todos los bits de condición excepto el de signo, estarían puestos a cero.

A1.6.3. SUB Restar registro o memoria del acumulador

Instrucción	SUB reg
Bits afectados	Z, S, P, CY, AC
Direccionamiento	Registro

Descripción

El contenido del registro o posición de memoria especificados se resta al contenido del acumulador, usando aritmética de complemento a dos. El resultado se guarda en el acumulador.

Si no hay acarreo del bit de más peso, el bit de acarreo se pone a uno, y viceversa, al contrario de lo que ocurre con la operación de suma

Formato

1	0	0	1	0	REGISTRO

Ejemplos

Antes de entrar en los ejemplos recordamos que restar utilizando aritmética de complemento a dos equivale a complementar cada bit del segundo operando y sumar 1.

1. Si el acumulador contiene 60H y el registro E contiene 28H, la instrucción

SUB E

realizará la siguiente operación de resta:

Acumulador	60H	0	1	1	0	0	0	0	0
+(–Registro B)	+(–28H)	1	1	0	1	0	1	1	1
Bit de acarreo									1
Nuevo acumulador	38H	0	0	1	1	1	0	0	0

Al final de la operación el acumulador contendrá 38H y el bit de acarreo se pondrá a cero debido a que ha habido acarreo del bit más significativo.

2. La instrucción

SUB A

restará al acumulador a sí mismo, obteniéndose un resultado de cero. Se utiliza en muchas ocasiones para poner a cero el bit de acarreo y el acumulador.

A1.6.4. SBB Restar registro o memoria del acumulador con acarreo

Instrucción	SBB reg
Bits afectados	Z, S, P, CY, AC
Direccionamiento	Registro indirecto

Descripción

El valor del bit de acarreo se suma internamente al contenido del registro o posición de memoria especificados. Este valor se resta del acumulador usando aritmética de complemento a dos.

Esta instrucción es de gran utilidad en la realización de restas de varios bytes, pues tiene en cuenta el valor positivo o negativo de la sustracción anterior.

Formato

1	0	0	1	1	REGISTRO

Ejemplo

Si el registro C contiene 08H, el acumulador almacena 05H y el bit de acarreo está activado, la instrucción

SBB C

efectúa la siguiente operación:

1. 08H + bit de acarreo = 09H.
2. Complemento a dos de 09H = 11110111 (F7H)
3. Lo anterior se suma al acumulador:

Acumulador	05H	0	0	0	0	0	1	0	1
	F7H	1	1	1	1	0	1	1	1
Nuevo acumulador	FCH	1	1	1	1	1	1	0	0

4. No hay acarreo al final por lo que el bit de acarreo se queda a uno. Los bits de signo y paridad están puestos a uno mientras que es bit de cero está a cero.

A1.6.5. ANA — Función lógica AND entre registro o memoria con acumulador

Instrucción	ANA reg
Bits afectados	Z, S, P, CY, AC
Direccionamiento	Registro indirecto

Descripción

Se realiza la función lógica AND bit a bit entre el contenido del registro o posición de memoria especificados y el contenido del acumulador. El bit de acarreo se pone a cero.

Formato

1	0	1	0	0	REGISTRO

Nota

La tabla de verdad de la función lógica AND es:

A	B	R
0	0	0
0	1	0
1	0	0
1	1	1

Ejemplo

Si el registro B contiene 6CH y el acumulador almacena 3AH, la instrucción

ANA B

realiza la siguiente operación:

Acumulador	3AH	0 0 1 1 1 0 1 0
Registro B	6CH	0 1 1 0 1 1 0 0
Nuevo acumulador	28H	0 0 1 0 1 0 0 0

A1.6.6. XRA Función lógica O-EXCLUSIVO entre registro o memoria con acumulador

Instrucción	XRA reg
Bits afectados	Z, S, P, CY, AC
Direccionamiento	Registro

Descripción

Se realiza la función lógica O-EXCLUSIVO bit a bit entre el contenido del registro o posición de memoria especificados y el contenido del acumulador, guardándose el resultado en este último.

Formato

1	0	1	0	1	REGISTRO

Nota

La tabla de verdad de la función lógica O-EXCLUSIVO es:

A	B	R
0	0	0
0	1	1
1	0	1
1	1	0

Ejemplos

1. Si el registro B contiene 6CH y el acumulador almacena 3AH, la instrucción

   XRA B

   realiza la siguiente operación:

Acumulador	3AH	0	0	1	1	1	0	1	0
Registro B	6CH	0	1	1	0	1	1	0	0
Nuevo acumulador	56H	0	1	0	1	0	1	1	0

2. La función O-EXCLUSIVO de cualquier bit con uno da lugar al complemento del mismo. Así, si el acumulador contiene todo unos, la instrucción

   XRA B

   produce el complemento a uno del contenido del registro B, y lo guarda en el acumulador.

3. En algunas ocasiones, un byte se utiliza para reflejar los estados de ciertas condiciones dentro de un programa, donde cada uno de los ocho bits puede responder a una determinada condición de falso o verdadero, actuado o inhibido, etc.

   Mediante la función O-EXCLUSIVO podemos determinar cuántos bits de la palabra han cambiado de estado en un determinado lapsus de tiempo.

A1.6.7. ORA Función lógica OR entre registro o memoria con acumulador

Instrucción	ORA reg
Bits afectados	Z, S, P, CY, AC
Direccionamiento	Registro indirecto

Descripción

Se realiza la función lógica AND bit a bit entre el contenido del registro o posición de memoria especificados y el contenido del acumulador, quedando en este último el resultado. El bit de acarreo se pone a cero.

Formato

Nota

La tabla de verdad de la función lógica OR es:

A	B	R
0	0	0
0	1	1
1	0	1
1	1	1

Ejemplo

Como sea que la función OR de cualquier bit con un uno da como resultado uno, y de cualquier bit con cero, lo deja invariable, esta función se utiliza frecuentemente para poner a uno grupos de bits.

Si el registro B contiene OFH y el acumulador almacena 33H, la instrucción

ORA B

realiza la siguiente operación:

Acumulador	33H	0 0 1 1 0 0 1 1
Registro B	0FH	0 0 0 0 1 1 1 1
Acumulador	3FH	0 0 1 1 1 1 1 1

Este ejemplo concreto garantiza que los cuatro bits de menos peso del acumulador son unos, mientras que los demás permanecen invariables.

A1.6.8. CMP Comparar registro o memoria con acumulador

Instrucción	CMP reg
Bits afectados	Z, S, P, CY, AC
Direccionamiento	Registro indirecto

Descripción

El contenido del registro o posición de memoria especificados se compara con el contenido del acumulador. Esta comparación se realiza restando internamente el contenido del registro al del acumulador, permaneciendo éste invariable, y colocando los bits de condición en función del resultado. Concretamente, el bit de cero se pone a uno si las cantidades comparadas son iguales, y se pone a cero si son desiguales. Como sea que se realiza una operación de resta, el bit de acarreo se pondrá a uno si no hay acarreo del bit 7, indicando que el contenido del registro o posición de memoria es mayor que el contenido del acumulador, y se pondrá a cero si es mayor que el acumulador.

Si las dos cantidades difieren en signo, el acarreo adopta el valor contrario a lo anteriormente expuesto.

Formato

1	0	1	1	1	REGISTRO

Ejemplos

A continuación exponemos 3 ejemplos de esta operación.

1. Si el acumulador almacena 0AH y el registro B contiene 05H, cuando se realiza la instrucción

   CMP B

   Tiene lugar la siguiente resta interna:

Acumulador	0AH	0 0 0 0 1 0 1 0
+(– Registro B)	–5H	1 1 1 1 1 0 1 1
Acumulador	05H	0 0 0 0 0 1 0 1

Existe acarreo en el bit 7 por lo que el bit de acarreo se pone a cero. El acumulador sigue almacenando 0AH y el registro B, 05H. No obstante, el bit de acarreo se pone a cero, así como el bit de cero, indicando que el contenido del registro B es menor que el acumulador.

2. Ahora el acumulador tiene el valor 02H. Entonces:

Acumulador	02H	0 0 0 0 0 0 1 0
+(– Registro B)	–5H	1 1 1 1 1 0 1 1
Acumulador	FDH	1 1 1 1 1 1 0 1

En este el bit de acarreo se pone a uno (no existe acarreo del bit 7) y el bit de cero estará a cero, indicando que el contenido del registro B es mayor que el acumulador.

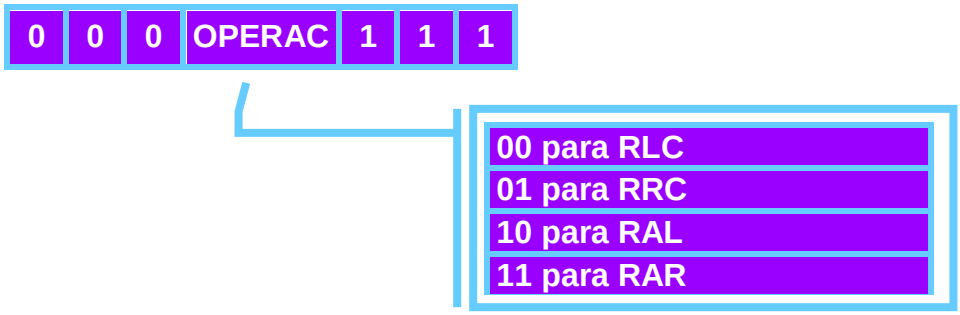
3. Por último supongamos un valor –1BH para el acumulador. En esta situación:

Acumulador	–1BH	1 1 1 0 0 1 0 1
+(– Registro B)	–5H	1 1 1 1 1 0 1 1
Acumulador	E0H	1 1 1 0 0 0 0 0

Aquí el bit de acarreo está a cero. Como los dos números difieren en signo, el hecho de que el bit de acarreo esté a cero indica que el contenido del registro B es mayor que el del acumulador, al contrario de cómo ocurría en el ejemplo anterior.

A1.7. Instrucciones de rotación del acumulador

A continuación se describen las instrucciones que provocan una rotación del contenido del acumulador. Esta operación únicamente puede realizarse con el acumulador, no con ningún registro o posición de memoria.



A1.7.1. RLC Desplazar el acumulador a la izquierda

Instrucción	RLC
Bits afectados	CY
Direccionamiento	

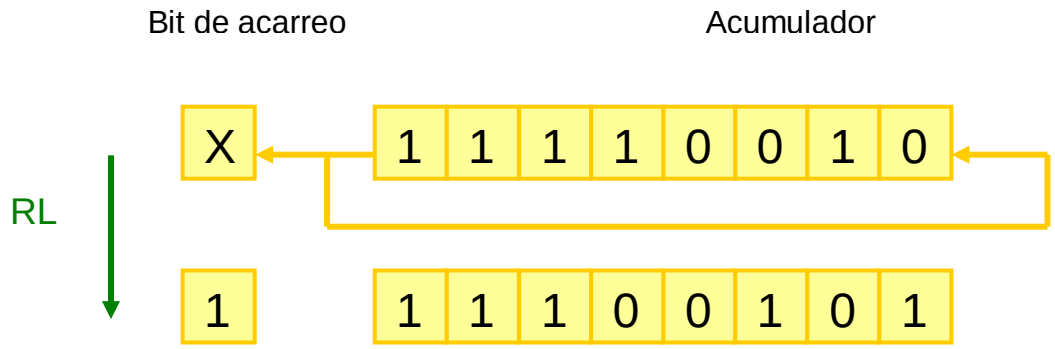
Descripción

RLC rota un bit hacia la izquierda todo el contenido del acumulador, transfiriendo el bit de más alto orden al bit de acarreo y al mismo tiempo a la posición de menor orden del acumulador.

Formato

Ejemplo

Supongamos que el acumulador contiene 82H.



A1.7.2. RRC Desplazar el acumulador a la derecha

Instrucción	RRC
Bits afectados	CY
Direccionamiento	

Descripción

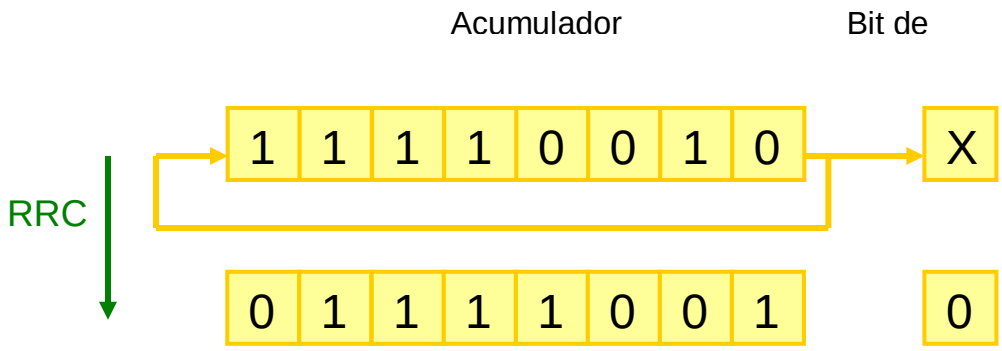
RRC rota el contenido del acumulador un bit a la derecha, transfiriendo el bit de más bajo orden a la posición de más alto orden del acumulador, además pone el bit de acarreo igual al bit de menor orden del acumulador.

Formato

0	0	0	1	0	1	1	1

Ejemplo

Supongamos que el acumulador contiene F2H. La instrucción RRC realizará la siguiente operación sobre el acumulador y el bit de acarreo:



A1.7.3. RAL Desplazar el acumulador hacia la izquierda a través del bit de acarreo

Instrucción	RAL
Bits afectados	CY
Direccionamiento	

Descripción

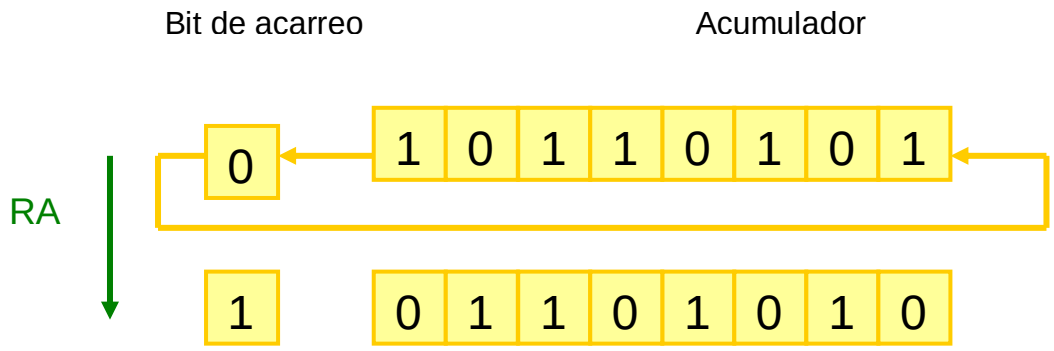
RAL hace girar el contenido del acumulador y el bit de acarreo un espacio de un bit hacia la salida (izquierda). El bit de acarreo que es tratado como si fuera del acumulador, se transfiere el bit de menor orden del acumulador. El bit de mayor orden del acumulador se transfiere al bit de acarreo. No tiene operandos.

Formato

0	0	0	1	0	1	1	1

Ejemplo

Supongamos que el acumulador contiene B5H. La instrucción RAL efectuará las siguientes modificaciones en el registro acumulador y en el bit de acarreo:



A1.7.4. RAR Desplazar el acumulador hacia la derecha a través del bit de acarreo

Instrucción	RAR
Bits afectados	CY
Direccionamiento	

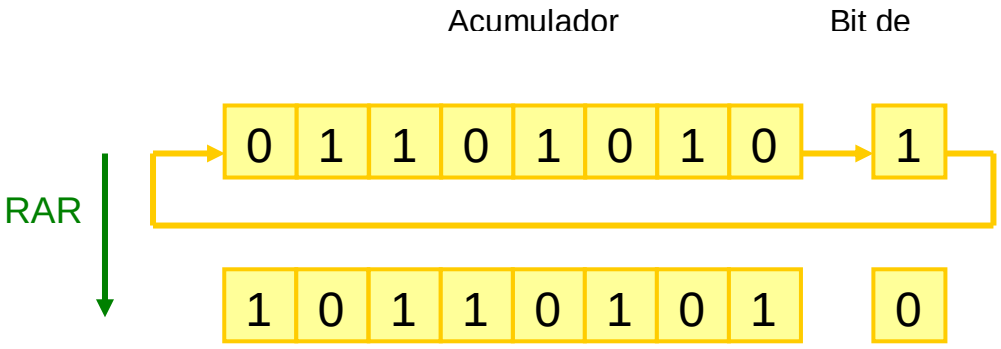
Descripción

RAR rota el contenido del acumulador y del bit de acarreo 1 bit de posición a la derecha. El bit de acarreo que es tratado como si fuera parte del acumulador se transfiere al bit de más alto orden del acumulador. El bit de menor peso del acumulador se carga en el bit de acarreo. No existen operandos en la instrucción RAR.

Formato

Ejemplo

En este caso el acumulador contendrá el valor 6AH. La instrucción RAL efectuará las siguientes modificaciones en el registro acumulador y en el bit de acarreo:


Acumulador
Bit de
0 1 1 0 1 0 1 0
1
RAR
1 0 1 1 0 1 0 1
0

A1.8. Instrucciones con pares de registros

A continuación se describen las instrucciones que dan lugar a operaciones con pares de registros.

A1.8.1. PUSH Colocar datos en stack

Instrucción	PUSH pr
Bits afectados	
Direccionamiento	Registro indirecto

Descripción

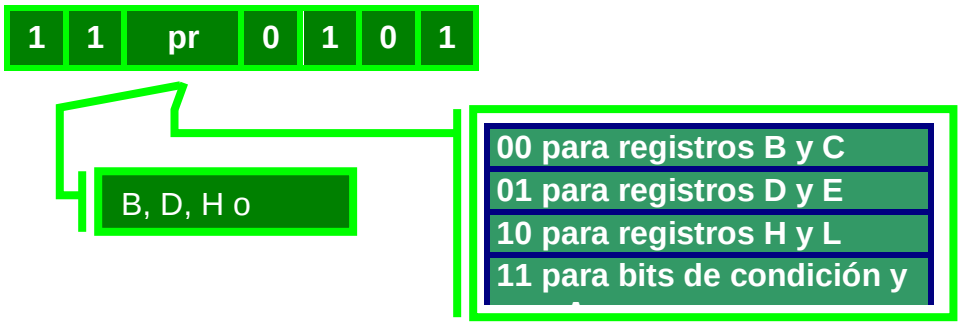
El contenido del par de registros especificado se guarda en dos bytes de memoria definidos por el puntero de stack.

El contenido del primer registro se guarda en la posición de memoria inmediatamente inferior a la del puntero de stack. El contenido del segundo registro del par se guarda en la posición de memoria dos unidades inferior al puntero de stack. Si se hace referencia al para de registros PSW, en el primer byte de información se guarda el estado de los cinco bits de condición. El formato de este byte es el siguiente:

signo	cero	0	arrrastre auxiliar	0	paridad	1	acarreo

Sea cual sea el par de registros especificado, una vez que los datos se han guardado, el puntero de pila se decrementa en dos unidades.

Formato



Ejemplos

1. Supongamos que el registro B contiene 3FH, el registro C contiene 16H y el puntero de pila vale 2030H. La instrucción

PUSH B

almacenará el contenido del registro B en posición de memoria 2029H, el contenido del registro C en la dirección de memoria 2028H, y decrementa dos unidades el puntero de stack, dejándolo en 2028H.

Gráficamente podemos ver el proceso anterior:

Antes de PUSH

Puntero stack 2030

Registro B 3F	Registro C 16

MEMORIA	DIRECCION
00	2027
00	2028
00	2029
00	2030

Después de PUSH

Puntero stack 2028

Registro B 3F	Registro C 16

MEMORIA	DIRECCION
00	2027
16	2028
3F	2029
00	2030

2. Supongamos ahora que el acumulador contiene 33H, el puntero de pila tiene 102AH, y los bits de condición de cero, acarreo y paridad están a uno, mientras que los de signo y acarreo auxiliar están a cero. La instrucción

PUSH PSW

Almacena el contenido del acumulador en la posición de memoria 1028H, y coloca el valor 47H, correspondiente a los citados estados de los bits de

condición, en la posición 1029H, mientras que en el puntero de pila queda el valor 1028H.

A1.8.2. POP Sacar datos del stack

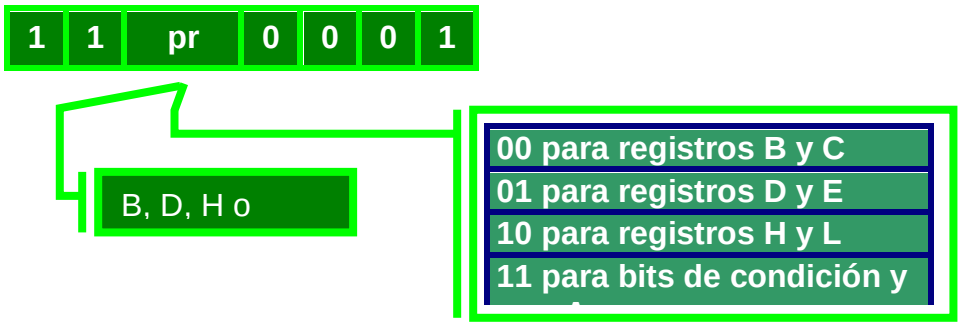
Instrucción	POP pr
Bits afectados	
Direccionamiento	Registro indirecto

Descripción

POP PR copia el contenido de la posición de memoria direccionada por el stack pointer en el registro de bajo orden del par de registros especificados. A continuación se incrementa el stack pointer en 1 y copia el contenido de la dirección resultante en el registro de más alto orden del par. Luego se incrementa el stack pointer otra vez de modo que se apunta al siguiente dato del stack. El operando debe especificar el par de registros BC, DE, HL o PSW.

POP PSW usa el contenido de la localización de memoria especificada por el stack pointer para restablecer los bits de condiciones.

Formato



Ejemplos

1. Supongamos que las posiciones de memoria 2028H y 2029H contienen respectivamente 16H y 3FH, mientras que el puntero de pila contiene 2028H. La instrucción

POP B

Carga el registro C con el valor de 16H de la posición de memoria 2028H, carga el registro B con el valor 3FH de la posición 2029H, e incrementa dos unidades el puntero de stack, dejándolo en 2030H. Gráficamente podemos ver este proceso:

Antes de POP		Después de POP	
Puntero stack 2028		Puntero stack 2030	
Registro B 00	Registro C 00	Registro B 3F	Registro C 16

MEMORIA	DIRECCION
00	2027
16	2028
3F	2029
00	2030

MEMORIA	DIRECCION
00	2027
16	2028
3F	2029
00	2030

2. Si las posiciones de memoria 1008H y 1009H poseen respectivamente 00H y 16H, y el puntero de pila vale 1008H, la instrucción

POP PSW

carga 00H en el acumulador y pone los bits de estado de la siguiente forma:

	S	Z		AC		P		CY
16h =	0	0	0	1	0	1	1	0

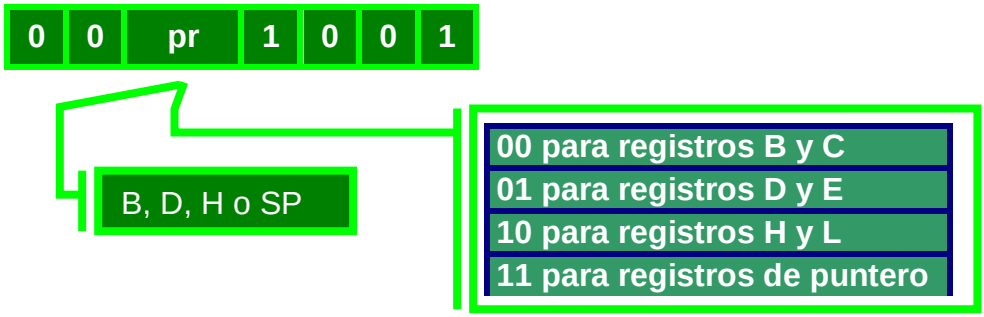
A1.8.3. DAD Suma doble

Instrucción	DAD pr
Bits afectados	CY
Direccionamiento	Registro

Descripción

DAD RP suma el valor de un dato de 16 bits contenido en un determinado par de registros (PR) al contenido del par de registros HL. El resultado es almacenado en el par HL. Los operandos (PR) pueden ser B = BC, D = DE, H = HL, SP. Téngase en cuenta que la letra H debe ser empleada para especificar que el par de registros HL debe ser doblado. DAD pone el bit de acarreo a 1 si hay una salida de acarreo de los registros HL. Y además no afecta a ningún otro bit.

Formato

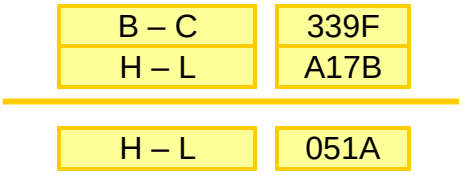


Ejemplos

1. Supuesto que los registros D, E, H y L contienen 33H, 9FH, A1H y 7BH respectivamente, la instrucción

   DAD D

   Realiza la siguiente suma:



2. Al ejecutar la instrucción

DAD H

se duplica el valor del número de 16 bits contenido en H – L (equivale a desplazar los 16 bits una posición hacia la izquierda).

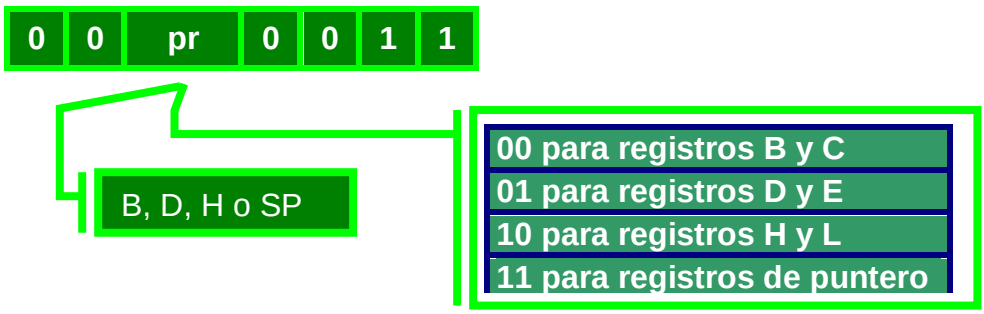
A1.8.4. INX Incrementar par de registros

Instrucción	INX pr
Bits afectados	
Direccionamiento	Registro

Descripción

El número de 16 bits contenido en el par de registros especificado se incrementa en una unidad.

Formato



Ejemplos

1. Suponiendo que los registros H y L contienen respectivamente 30H y 00H, la instrucción

INX H

hace que el registro H contenga 30H y el registro L el valor 01H.

2. Si el puntero de pila contiene FFFFH, la instrucción

INX SP

hace que éste contenga 0000H.

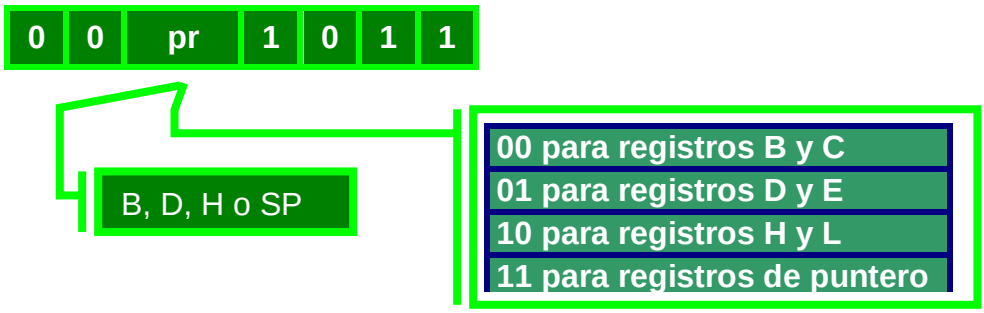
A1.8.5. DCX Decrementar par de registros

Instrucción	DCR pr
Bits afectados	
Direccionamiento	Registro

Descripción

El número de 16 bits contenido en el par de registros especificado se decrementa en una unidad.

Formato



Ejemplo

Suponiendo que los registros H y L contienen respectivamente 30H y 00H, la instrucción

DCX H

hace que el registro H contenga 2FH y el registro L el valor FFH.

A1.8.6. XCHG Intercambiar datos entre registros

Instrucción	XCHG
Bits afectados	
Direccionamiento	Registro

Descripción

XCHG cambia el contenido de los registros H y L con el contenido de los registros D y E.

Formato

Ejemplo

Si los registros H, L, D y E contienen respectivamente 01H, 02H, 03H y 04H, la instrucción XCHG realiza el siguiente intercambio:

Antes de ejecutar XCHG

D	E	H	L
03	04	01	02

Después de ejecutar XCHG

D	E	H	L
01	02	03	04

A1.8.7. XTHL Intercambiar datos con el stack

Instrucción	XTHL
Bits afectados	
Direccionamiento	Registro indirecto

Descripción

XTHL cambia los dos bytes de la posición más alta del stack con los dos bytes almacenados en los registros H y L. Así XTHL salva el contenido actual del par HL y

carga nuevos valores en HL.

XTHL cambia el contenido del L con la posición de memoria especificada por el stack pointer y el registro H es intercambiado con el contenido del SP+1.

Formato

Ejemplo

Si el puntero de pila contiene 40B4H, los registros H y L contienen AAH y BBH respectivamente, y las posiciones de memoria 40B4H y 40B5H contienen CCH y DDH respectivamente, la instrucción

XTHL

realizará la siguiente operación:

Antes de XTHL	
Puntero stack 40B4	
Registro H AA	Registro L BB

MEMORIA	DIRECCION
… …	… …
00	40B3
CC	40B4
DD	40B5
00	40B6
… …	… …

Después de XTHL	
Puntero stack 40B4	
Registro H DD	Registro L CC

MEMORIA	DIRECCION
… …	… …
00	40B3
BB	40B4
AA	40B5
00	40B6
… …	… …

A1.8.8. SPHL Cargar el puntero de stack desde los registros H y L

Instrucción	SPHL
Bits afectados	
Direccionamiento	

Descripción

Los 16 bits contenidos en los registros H y L reemplazan el contenido del puntero de stack. El contenido de los registros H y L permanece invariable.

Formato

Ejemplo

Si los registros H y L contienen respectivamente 50H y 6CH, la instrucción

SPHL

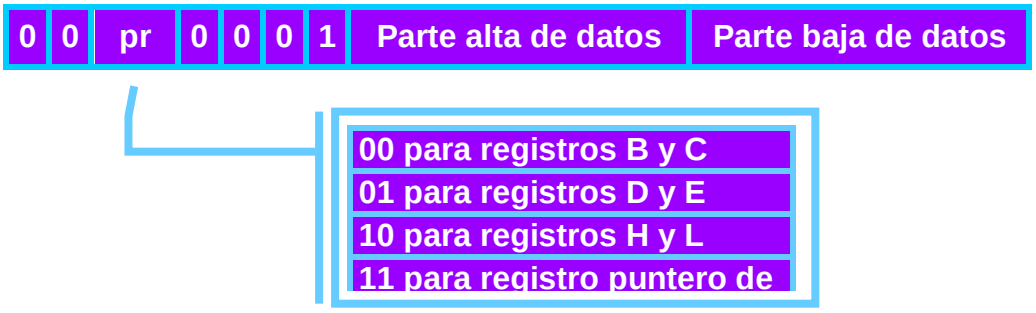
carga el puntero de stack con el valor 506CH.

A1.9. Instrucciones con datos inmediatos

A continuación se describen las instrucciones que realizan operaciones utilizando bytes de datos que forman parte de la propia instrucción.

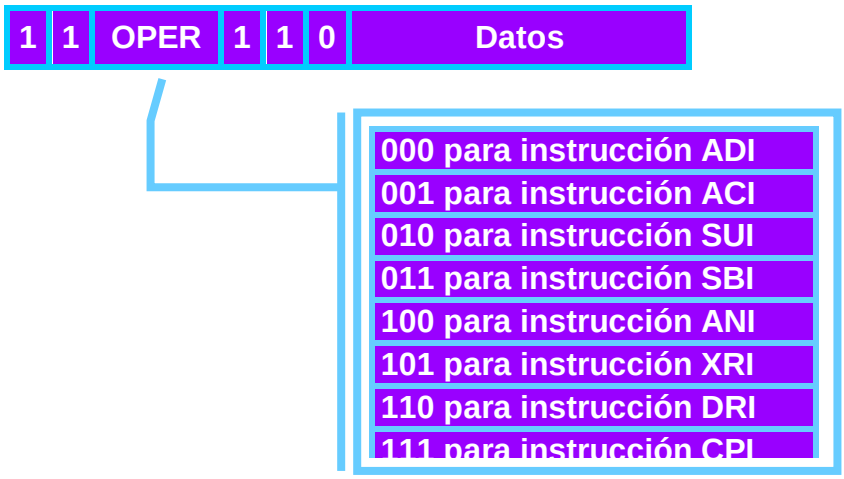
Estas instrucciones forman un grupo amplio en el que no todas tienen la misma longitud y formato. Las instrucciones ocupan dos o tres bytes del siguiente modo:

1. La instrucción LXI ocupa 3 bytes con el siguiente formato:



2. La instrucción MVI ocupa 2 bytes siguiente el formato siguiente:

3. Por último, para el resto de las instrucciones, que ocupan 2 bytes, se cuenta con este formato:



La instrucción LXI opera sobre el par de registros especificado por PR, usando dos bytes de datos inmediatos.

La instrucción MVI opera sobre el registro especificado por REG, usando un byte de datos inmediatos. Si se hace referencia a la memoria, la instrucción opera sobre la posición de memoria de la misma determinada por los registros H y L. El registro H contiene los 8 bits más significativos de la dirección, mientras que el registro L contiene los 8 bits menos significativos.

Las restantes instrucciones operan sobre el acumulador, usando un byte de datos inmediatos. El resultado sustituye al contenido del acumulador.

A1.9.1. LXI Cargar un par de registros con un dato Inmediato

Instrucción	LXI pr, datos
Bits afectados	
Direccionamiento	Inmediato

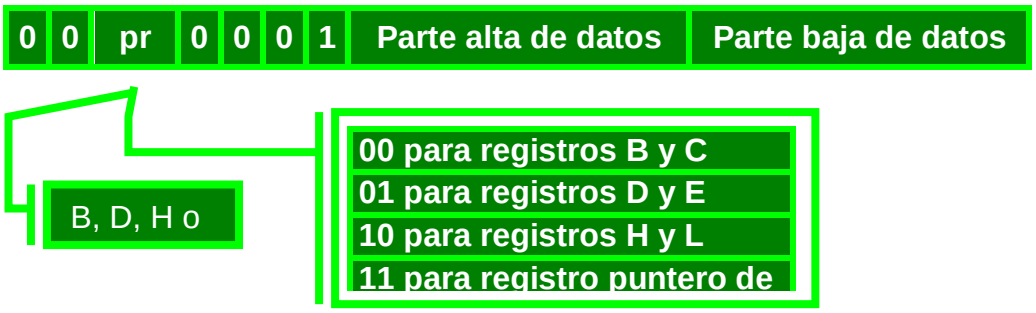
Descripción

LXI es una instrucción de 3 bytes; su segundo y tercer byte contienen el dato que ha de ser cargado en el par de registros (PR). El primer operando debe especificar el par de registros a ser cargados, pueden ser los pares BC, DE, HL, o el SP. El segundo operando especifica los dos bytes a ser cargados. LXI es la única instrucción inmediata

que acepta un valor de 16 bits. El resto trabajan con datos de 8 bits.

Si el par de registros especificados es SP, el segundo byte de la instrucción sustituye a los 8 bits menos significativos del puntero de pila, mientras que el tercer byte de la instrucción reemplaza a los 8 bits más significativos del puntero de pila.

Formato



Ejemplos

1. La instrucción

   LXI B, 00FFH

   carga en el registro B el valor 00H y en el registro C el valor FFH.

2. La siguiente instrucción carga en el puntero de pila el valor 1000H

   LXI SP, 1000H

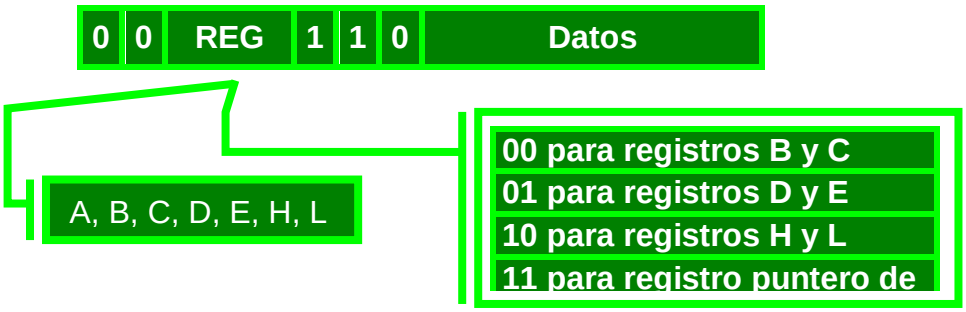
A1.9.2. MVI Cargar un registro con un dato Inmediato

Instrucción	MVI reg, datos
Bits afectados	
Direccionamiento	Inmediato

Descripción

El primer operando debe ser uno de los registros A, B, C, D, E, H o L, que será cargado con el dato especificado en el segundo operando (DATOS). El dato no debe exceder de un byte.

Formato



Ejemplos

(1). La instrucción

MVI H, 33H

carga en el registro H el valor 33H.

(2). La instrucción

MVI L, 44H

carga en el registro L el valor 44H.

(3). Supuestos los dos ejemplos anteriores, la instrucción

MVI M, 2AH

carga en la posición de memoria 3344H (dirección aportada por los registros H y L) el valor 2AH.

A1.9.3. ADI Sumar al acumulador un dato Inmediato

Instrucción	ADI datos
Bits afectados	Z, S, P, CY, AC
Direccionamiento	Inmediato

Descripción

Suma el valor del byte especificado en la instrucción (DATOS), al contenido del acumulador y deja el resultado en el acumulador. El dato debe ser expresado en forma de número, un ASCII constante, la etiqueta de un valor previamente definido o una expresión. El dato no debe exceder de un byte.

Se utiliza aritmética de complemento a dos.

Formato

1	1	0	0	0	1	1	0	Datos

Ejemplo

A continuación presentamos un ejemplo con 3 instrucciones:

(1). MVI A, 34
(2). ADI 20
(3). ADI -20

En todas las instrucciones se utilizan datos en base decimal. Así, por ejemplo, en la instrucción (2) el valor 20 es 14H.

La instrucción (1) carga en el acumulador el valor 22H.

La instrucción (2) realiza la siguiente suma:

Acumulador	22H	0	0	1	0	0	0	1	0
Dato inmediato	14H	0	0	0	1	0	1	0	0
Nuevo acumulador	33H	0	0	1	1	0	1	1	0

El bit de paridad se pone a uno y el resto se quedan a cero.

La instrucción (3) restaura el valor del acumulador realizando la siguiente suma:

Acumulador	33H	0	0	1	1	0	1	1	0
Dato inmediato	ECH	1	1	1	0	1	1	0	0
Nuevo acumulador	22H	0	0	1	0	0	0	1	0

Ahora los bits de paridad, acarreo y acarreo auxiliar se quedan a uno y el resto a cero.

A1.9.4. ACI Sumar al acumulador un dato Inmediato con arrrastre

Instrucción	ACI datos
Bits afectados	Z, S, P, CY, AC
Direccionamiento	Inmediato

Descripción

Suma el contenido del byte especificado (DATOS) en la instrucción, al contenido del acumulador, añadiendo además el bit del acarreo. El resultado se almacena en el acumulador (perdiéndose así el anterior contenido del Acumulador).

El dato (DATOS) debe estar especificado en forma de número, en ASCII constante, como etiqueta de un valor previamente definido o una expresión. El dato no debe exceder de un byte.

Formato

Ejemplo

Tenemos las siguientes instrucciones:

(1). MVI A, 34

(2). ACI 20

y suponemos el bit de acarreo puesto a uno.

La instrucción (1) carga en el acumulador el valor 22H.

La instrucción (2) realiza la siguiente suma:

Acumulador	22H	0 0 1 0 0 0 1 0
Dato inmediato	14H	0 0 0 1 0 1 0 0
Bit de acarreo		1
Nuevo acumulador	37H	0 0 1 1 0 1 1 1

Todos los bits se ponen a cero.

A1.9.5. SUI Restar del acumulador un dato Inmediato

Instrucción	SUI datos
Bits afectados	Z, S, P, CY, AC
Direccionamiento	Inmediato

Descripción

El byte de datos inmediato se resta del contenido del acumulador usando aritmética de complemento a dos. El resultado se deja en el acumulador.

Ya que se trata de una operación de resta, el bit de acarreo se pone a uno cuando no hay acarreo del bit de más peso, y se pone a cero si tiene dicho acarreo.

Formato

Ejemplo

A continuación presentamos un ejemplo con 2 instrucciones:

(1). MVI A, B3H
(2). SUI B3H

La instrucción (1) carga en el acumulador el valor B3H.

La instrucción (2) realiza la siguiente suma (usando el complemento a dos del dato inmediato):

Acumulador	B3H	1 0 1 1 0 0 1 1
Dato inmediato	6DH	0 1 0 0 1 1 0 1
Nuevo acumulador	00H	0 0 0 0 0 0 0 0

Como era de esperar el resultado final del acumulador es cero ya que le estamos restando su propio valor. El valor 6DH del dato inmediato corresponde al complemento a dos del valor B3H que estamos restando.

Debido a que existe desbordamiento del séptimo bit se produce acarreo y se pone el bit de acarreo a cero.

El bit de paridad se pone a uno mientras que los demás permanecen inactivos.

A1.9.6. SBI Restar del acumulador un dato Inmediato con arrrastre

Instrucción	SBI datos
Bits afectados	Z, S, P, CY, AC
Direccionamiento	Inmediato

Descripción

El bit de acarreo se suma internamente al byte de datos inmediato. El valor obtenido se resta del contenido del acumulador usando aritmética de complemento a dos. El resultado se deja en el acumulador.

Esta instrucción, al igual que SBB, se usa preferentemente para realizar restas multi-byte.

Al igual que en el apartado anterior, el bit de acarreo se pone a uno si no hay acarreo del bit de más peso, poniéndose a cero si lo hay.

Formato

Ejemplo

Tenemos las siguientes instrucciones:

(1). MVI A, 00H

(2). SBI 01H

y suponemos el bit de acarreo puesto a cero.

La instrucción (1) carga en el acumulador el valor 00H.

La instrucción (2) realiza la siguiente suma (usando el complemento a dos del dato inmediato):

Acumulador	00H	0	0	0	0	0	0	0	0
Dato inmediato	FFH	1	1	1	1	1	1	1	1
Bit de acarreo									0
Nuevo acumulador	- 1H	1	1	1	1	1	1	1	1

No hay acarreo, por lo que el bit de acarreo se pone a uno. Los bits de cero y acarreo auxiliar están a cero, mientras que los de signo y paridad se ponen a uno.

A1.9.7. ANI Función lógica AND entre el acumulador y un Dato Inmediato

Instrucción	ANI datos
Bits afectados	Z, S, P, CY, AC
Direccionamiento	Inmediato

Descripción

Realiza una operación Y lógica entre el dato (DATOS) especificado en la instrucción y el contenido del acumulador, el resultado queda en el acumulador. Se pone a cero el bit de acarreo. El dato, que no debe exceder de un byte, puede ser expresado en forma de número, un ASCII constante, la etiqueta de algún valor previamente definido o una expresión.

Formato

Ejemplo

Disponemos de las siguientes instrucciones:

(1). MVI A, A0H
(2). ANI 0FH

La instrucción (1) carga en el acumulador el valor A0H.

La instrucción (2) realiza la siguiente operación AND bit a bit entre el acumulador y el dato inmediato 0FH:

Acumulador	A0H	1 0 1 0 0 0 0 0
Dato inmediato	OFH	0 0 0 0 1 1 1 1
Nuevo acumulador	00H	0 0 0 0 0 0 0 0

La instrucción ANI del ejemplo pone a cero los cuatro bits de mayor peso, dejando invariables los cuatro menores. Ya que los cuatro bits de menor peso del acumulador eran cero, el resultado final es 00H con lo que el bit de cero se pondrá a cero.

A1.9.8. XRI Función lógica O-EXCLUSIVO entre el acumulador y un dato Inmediato

Instrucción	XRI datos
Bits afectados	Z, S, P, CY, AC
Direccionamiento	Inmediato

Descripción

Se realiza la función lógica O-EXCLUSIVO bit a bit entre un byte de datos inmediatos y el contenido del acumulador. El resultado se guarda en el acumulador. El bit de acarreo se pone a cero.

Formato

1	1	1	0	1	1	1	0	Datos

Ejemplo

Esta instrucción se suele utilizar para complementar bits específicos del acumulador dejando los restantes en su estado original. De este modo y suponiendo que el acumulador contiene ABH, la instrucción

XRI 80H

complementa el bit de más peso del acumulador, tal y como se muestra en la siguiente figura:

Acumulador	ABH	1	0	1	0	1	0	1	1
Dato inmediato	80H	1	0	0	0	0	0	0	0
Nuevo acumulador	2BH	0	0	1	0	1	0	1	1

A1.9.9. ORI Función lógica OR entre el acumulador y un dato Inmediato

Instrucción	ORI datos
Bits afectados	Z, S, P, CY, AC
Direccionamiento	Inmediato

Descripción

ORI desarrolla una operación lógica OR entre el contenido especificado por DATOS y el contenido del acumulador. El resultado se deja en el acumulador. Los bits de acarreo y acarreo auxiliar se ponen a cero.

Formato

Ejemplo

Si el acumulador inicialmente contiene 3CH, la instrucción

ORI F0H

realiza la siguiente operación OR bit a bit:

Acumulador	3CH	0 0 1 1 1 1 0 0
Dato inmediato	F0H	1 1 1 1 0 0 0 0
Nuevo acumulador	FCH	1 1 1 1 1 1 0 0

Como vemos la instrucción ORI de nuestro ejemplo activa los cuatro bits de menor peso, dejando invariables los restantes.

A1.9.10. CPI Comparar el contenido del acumulador con un dato Inmediato

Instrucción	CPI datos
Bits afectados	Z, S, P, CY, AC

Direccionamiento | Registro indirecto

Descripción

Compara el valor del byte especificado (DATOS) con el contenido del acumulador y posiciona los bits de cero y acarreo para indicar el resultado. El bit de cero indica igualdad. Un 0 en el acarreo indica que el contenido del acumulador es mayor que DATOS. Un 1 en el acarreo indica que el acumulador es menor que DATOS. Sin embargo, el significado del bit de acarreo es contrario cuando los valores tienen diferente signo o cuando uno de los valores está complementado. El valor de DATOS no debe exceder de un byte.

Formato

1	1	1	1	1	1	1	0	Datos

Ejemplo

Si tenemos la secuencia de instrucciones

(1). MVI A, 25H
(2). CPI 20H

La instrucción (1) carga en el acumulador el valor 25H.

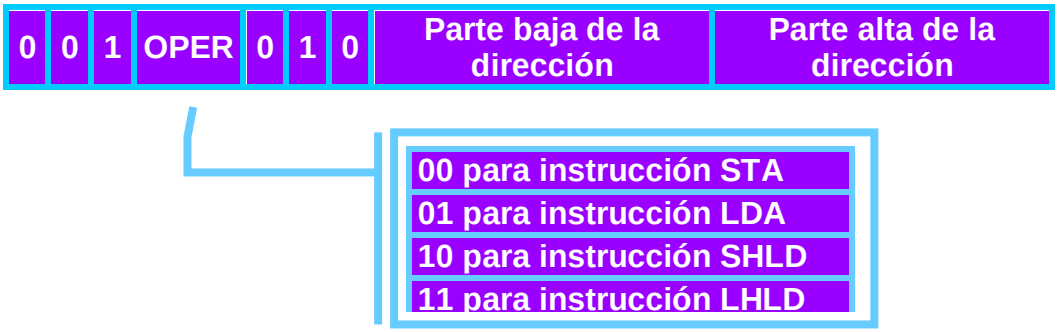
La instrucción (2) realiza la siguiente operación de suma (tomando el complemento a dos del dato inmediato, es decir, E1H):

Acumulador	25H	0	0	1	0	0	1	0	1
Dato inmediato	E1H	1	1	1	0	0	0	0	1
Nuevo acumulador	03H	0	0	0	0	0	1	1	0

Existe desbordamiento del último bit, por lo que el bit de acarreo se pone a cero. El acumulador continua con su valor inicial pero el bit de cero está a cero, indicando que las cantidades no son iguales. Al estar el bit de acarreo a cero, nos indica que los datos inmediatos son menores que el contenido del acumulador.

A1.10. Instrucciones de direccionamiento directo

A continuación se describen las instrucciones que hacen referencia a una posición de memoria específica, cuyos dos bytes de dirección forman parte de la propia instrucción. Las instrucciones de este tipo ocupan tres bytes en la forma siguiente:



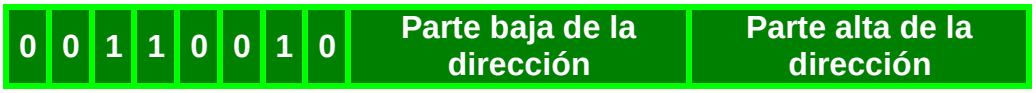
A1.10.1. STA Almacenamiento directo desde el Acumulador

Instrucción	STA dir
Bits afectados	
Direccionamiento	Directo

Descripción

STA DIR almacena una copia del contenido actual del acumulador en la posición de memoria especificada por DIR.

Formato



Ejemplo

Todas las instrucciones que se muestran a continuación introducen el contenido del acumulador en la posición de memoria 0080H:

- STA 0080H // Base hexadecimal
- STA 128 // Base decimal
- STA 0000000010000000B // Base binaria

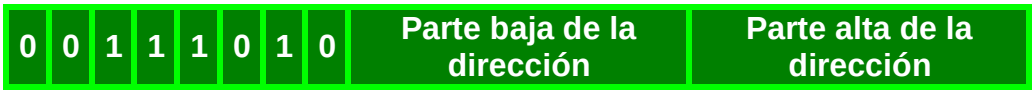
A1.10.2. LDA Carga directa en el acumulador

Instrucción	LDA dir
Bits afectados	
Direccionamiento	Directo

Descripción

LDA DIR carga el acumulador con el contenido de la memoria direccionada por DIR. La dirección puede ser puesta como un número, una etiqueta previamente definida o una expresión.

Formato



Ejemplo

Todas las instrucciones que se muestran a continuación introducen en el acumulador el contenido de la posición de memoria 300H:

- LDA 300H
- LDA 3 * (16 * 16)
- LDA 200H + 256

A1.10.3. SHLD Cargar directamente con H y L

Instrucción	SHLD dir
Bits afectados	
Direccionamiento	Directo

Descripción

Almacena una copia del registro L en la posición de memoria especificada por DIR, a continuación almacena una copia del registro H en la siguiente posición de memoria (DIR+1).

Formato

0	0	1	0	0	0	1	0	Parte baja de la dirección	Parte alta de la dirección

Ejemplo

Suponiendo que los registros H y L contienen respectivamente los valores 3CH y 54H, la instrucción

SHLD 34B3

efectuará las siguientes modificaciones en memoria:

Memoria antes de ejecutar SHLD		DIRECCIÓN		Memoria después de ejecutar SHLD
FF	←	34B2	→	FF
FF	←	34B3	→	54
FF	←	34B4	→	3C
FF	←	34B5	→	FF

A1.10.4. LHLD Cargar H y L directamente

Instrucción	LHLD dir
Bits afectados	
Direccionamiento	Directo

Descripción

LHLD DIR carga el registro L con una copia del byte almacenado en la posición de memoria especificada por DIR. Después carga el registro H con una copia del byte almacenado en la posición siguiente de memoria especificada por DIR. La instrucción LHLD esta prevista para cargar direcciones nuevas en los registros H y L.

Formato

0	0	1	0	1	0	1	0	Parte baja de la dirección	Parte alta de la dirección

Ejemplo

En el caso en el que las posiciones de memoria AB24H y AB25H contengan respectivamente 22H y 33H, la ejecución de la instrucción

LHLD AB24H

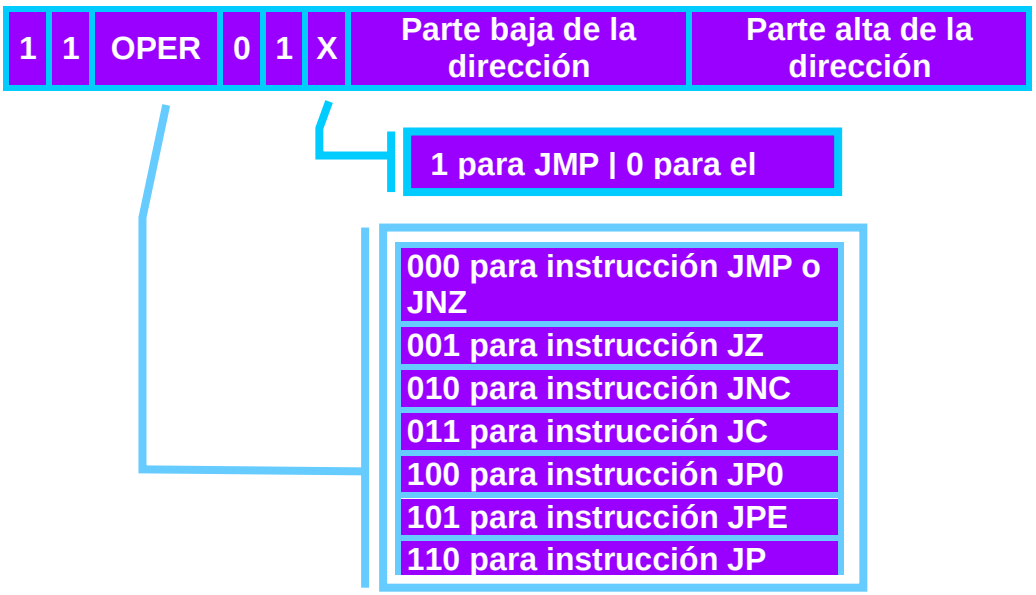
hará que el registro L contenga 22H y el registro H contenga 33H.

A1.11. Instrucciones de salto

A continuación se describen las instrucciones que modifican la secuencia normal de ejecución de las instrucciones de un programa. Estas instrucciones ocupan uno o tres bytes en la forma siguiente:

1. La instrucción PCHL ocupa un byte con el siguiente formato:

2. El resto de instrucciones de salto ocupan tres bytes con el formato siguiente:



Ciertas instrucciones de tres bytes de este tipo provocan un cambio en la secuencia normal de operaciones en la ejecución de un programa según unas determinadas condiciones.

Si la condición específica es verdadera, la ejecución del programa continúa en la dirección de memoria formada por la parte alta (tercer byte de la instrucción), y los ocho bits de la parte baja (segundo byte de la instrucción).

Si la condición específica es falsa, la ejecución del programa continúa en la próxima instrucción en secuencia.

A1.11.1. PCHL Cargar el contador de programa

Instrucción	PCHL
Bits afectados	
Direccionamiento	Registro

Descripción

PCHL carga el contenido de los registros H y L en el contador de programa. Como el procesador busca la siguiente instrucción en la siguiente dirección del contador de programa, PCHL tiene el efecto de una instrucción de salto. El contenido de H va a los 8 bits más altos de contador de programa y el contenido de L va a los 8 bits más bajos.

Formato

Ejemplo

Si el registro H contiene A5H y el registro L contiene 9DH, la instrucción

PCHL

hace que el programa en curso continúe ejecutándose en la dirección de memoria A59DH.

A1.11.2. JMP Salto incondicional

Instrucción	JMP dir
Bits afectados	
Direccionamiento	Inmediato

Descripción

La instrucción JMP DIR altera la ejecución del programa cargando el valor especificado por DIR en el contador de programa.

Formato

A1.11.3. JC Saltar si hay acarreo

Instrucción	JC dir
Bits afectados	
Direccionamiento	Inmediato

Descripción

La instrucción JC DIR comprueba el valor del bit de acarreo. Si es un 1 la ejecución del programa continúa en la dirección especificada por DIR. Si es un 0 el programa

continúa su ejecución normal de forma secuencial.

Formato

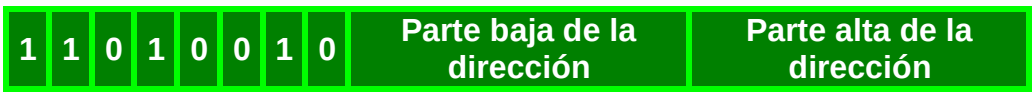
A1.11.4. JNC Saltar si no hay acarreo

Instrucción	JNC dir
Bits afectados	
Direccionamiento	Inmediato

Descripción

La instrucción JNC DIR comprueba el estado del bit acarreo. Si esta a 0 el programa cambia a la dirección especificada por DIR. Si esta a 1 la ejecución del programa continúa normalmente.

Formato



A1.11.5. JZ Saltar si hay cero

Instrucción	JZ dir
Bits afectados	
Direccionamiento	Inmediato

Descripción

La instrucción JZ DIR comprueba el bit de cero. Si está a 1 el programa continúa en la dirección expresada por DIR. Si está a 0 continúa con la ejecución secuencial normal.

Formato

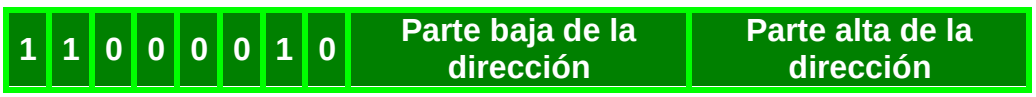
A1.11.6. JNZ Saltar si no hay cero

Instrucción	JNZ dir
Bits afectados	
Direccionamiento	Inmediato

Descripción

La instrucción JNZ DIR comprueba el valor del bit de cero. Si el contenido del acumulador no es cero (Bit de cero = 0) el programa continúa en la dirección especificada por DIR. Si el contenido del acumulador es cero (Bit de cero = 1) el programa continúa su ciclo normal.

Formato



A1.11.7. JM Saltar si hay signo negativo

Instrucción	JM dir
Bits afectados	
Direccionamiento	Inmediato

Descripción

La instrucción JM DIR comprueba el estado del bit de signo. Si el contenido del acumulador es negativo (bit de signo = 1) la ejecución del programa continúa en la dirección especificada por DIR. Si el contenido del acumulador es positivo (bit de signo = 0) continúa la ejecución de la secuencia normal.

Formato

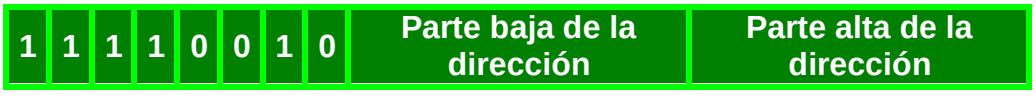
A1.11.8. JP Saltar si hay signo positivo

Instrucción	JP dir
Bits afectados	
Direccionamiento	Inmediato

Descripción

La instrucción JP DIR comprueba el estado de bit del signo. Si el contenido del acumulador es positivo (bit de signo = 0) la ejecución del programa continúa con la dirección especificada por DIR. Si el contenido del acumulador es negativo (bit de signo = 1) continúa el programa con su ejecución normal.

Formato



A1.11.9. JPE Saltar si la paridad es par

Instrucción	JPE dir
Bits afectados	
Direccionamiento	Inmediato

Descripción

La paridad existe si el byte que esta en el acumulador tiene un número par de bits. El bit de paridad se pone a 1 para indicar esta condición.

La instrucción JPE DIR comprueba la situación del bit de paridad. Si esta a 1, la ejecución del programa continúa en la dirección especificada por DIR. Si esta a 0, continúa con la siguiente instrucción de forma secuencial.

Las instrucciones JPE y JPO son especialmente usadas para comprobar la paridad de los datos de entrada. (Sin embargo con la instrucción IN los bits no actúan. Esto puede evitarse sumando 00H al acumulador para activarlos).

Formato

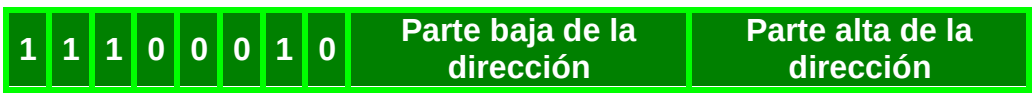
A1.11.10. JP0 Saltar si la paridad es impar

Instrucción	JPO dir
Bits afectados	
Direccionamiento	Inmediato

Descripción

La instrucción JPO DIR comprueba el estado del bit de paridad. Si esta a 0, el programa continúa en la dirección marcada por DIR. Si está a 1 continúa con la secuencia normal.

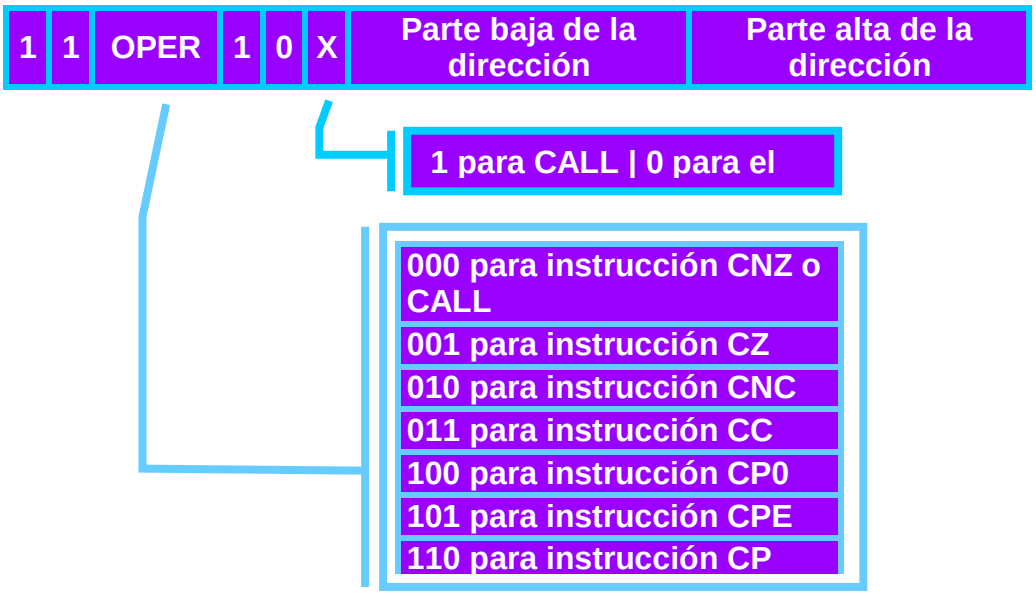
Formato



A1.12. Instrucciones de llamada a subrutina

A continuación se describen las instrucciones que llaman a subrutinas. Estas instrucciones operan en la misma forma que las instrucciones de salto, provocando una alteración en la secuencia de ejecución de las instrucciones, pero además, en el momento de su ejecución, se almacena en la pila una dirección de retorno, que es utilizada por las instrucciones de RETORNO (ver instrucciones de Retorno de Subrutinas en este mismo capítulo).

Las instrucciones de este tipo utilizan tres bytes en el formato siguiente:



En las instrucciones de llamada, las instrucciones se codifican como en las instrucciones de salto, es decir, almacenando en primer lugar el byte de dirección de memoria menos significativo.

A1.12.1. CALL Llamada incondicional

Instrucción	CALL dir
Bits afectados	
Direccionamiento	Inmediato / Registro indirecto

Descripción

CALL guarda el contenido del contador de programa (la dirección de la próxima instrucción secuencial) dentro del stack y a continuación salta a la dirección especificada por DIR.

Cada instrucción CALL o alguna de sus variantes implica una instrucción RET (retorno), de lo contrario el programa podría "perderse" con consecuencias impredecibles. La dirección debe ser especificada como un número, una etiqueta, o una expresión. La etiqueta es lo más normal (El ensamblador invierte los bytes alto y bajo de dirección durante el proceso de ensamblado). Las instrucciones CALL se emplean para llamadas a subrutinas y debemos tener presente que siempre emplean el stack.

Formato

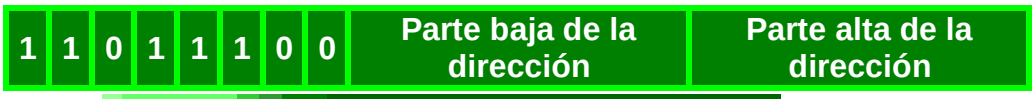
A1.12.2. CC Llamar si hay acarreo

Instrucción	CC dir
Bits afectados	
Direccionamiento	Inmediato / Registro indirecto

Descripción

CC comprueba el estado del bit de acarreo. Si el bit está a 1, CC carga el contenido del contador de programa en el stack y a continuación salta a la dirección especificada por DIR. Si el bit esta a 0, la ejecución del programa continúa con la próxima instrucción de su secuencia normal. Aunque el uso de una etiqueta es lo más normal, la dirección puede ser especificada también como un número o una expresión.

Formato



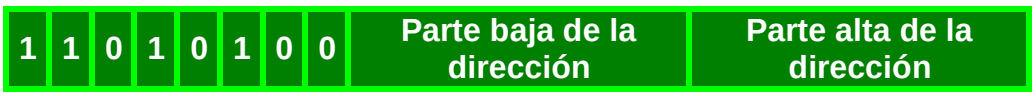
A1.12.3. CNC Llamar si no hay acarreo

Instrucción	CNC dir
Bits afectados	
Direccionamiento	Inmediato / Registro indirecto

Descripción

CNC chequea el valor del bit de acarreo. Si está en cero CNC carga el contenido de contador de programa en el stack y a continuación salta a la dirección especificada por la instrucción en DIR. Si el bit está a 1, el programa continúa con su secuencia normal. Aunque el uso de una etiqueta es lo más común, la dirección puede también estar indicada por un número o por una expresión.

Formato



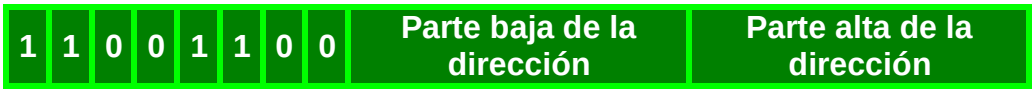
A1.12.4. CZ Llamar si hay cero

Instrucción	CZ dir
Bits afectados	
Direccionamiento	Inmediato / Registro indirecto

Descripción

CZ chequea el bit de cero. Si el bit esta a 1 (indicando que el contenido del acumulador es cero), CZ carga el contenido del contador de programa en el stack y salta a la dirección especificada en DIR. Si el bit está a 0 (indicando que el contenido del acumulador es distinto de cero) el programa continúa su desarrollo normal.

Formato



A1.12.5. CNZ Llamar si no hay cero

Instrucción	CNZ dir
Bits afectados	
Direccionamiento	Inmediato / Registro indirecto

Descripción

CNZ chequea el bit de Cero. Si está en 0 (indicando que el contenido del acumulador no es cero), CNZ manda el contenido del contador de programa al stack y salta a la dirección especificada por DIR. Si el bit está a 1 el programa continúa su desarrollo normal.

Formato

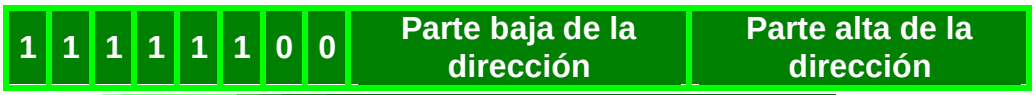
A1.12.6. CM Llamar si hay signo negativo

Instrucción	CM dir
Bits afectados	
Direccionamiento	Inmediato / Registro indirecto

Descripción

CM comprueba el estado del bit del signo. Si el bit esta a 1 (indicando que el contenido del acumulador es negativo) CM manda el contenido del contador de programa al stack y salta a la dirección especificada por DIR. Si el bit es 0 la ejecución del programa continúa con su secuencia normal. El uso de la etiqueta es lo más corriente, pero la dirección puede especificarse también por un número o una expresión.

Formato



A1.12.7. CP Llamar si hay signo positivo

Instrucción	CP dir
Bits afectados	
Direccionamiento	Inmediato / Registro indirecto

Descripción

CP chequea el valor del bit de signo. Si está a 0 (indicando que el contenido del acumulador es positivo), CP envía el contenido del contador de programa al stack y salta a la dirección especificada por DIR. Si el bit tiene un 1, continúa el programa normalmente con la instrucción siguiente.

Formato

A1.12.8. CPE Llamar si la paridad es par

Instrucción	CPE dir
Bits afectados	
Direccionamiento	Inmediato / Registro indirecto

Descripción

Existe paridad en un byte si el número de unos que tiene es par. El bit de paridad se pone a 1 para indicar esta condición. CPE chequea el valor del bit de paridad. Si tiene un 1, CPE envía el contenido del contador de programa al stack y salta a la dirección especificada por la instrucción en DIR. Si el bit tiene un cero, el programa continúa

normalmente.

Formato

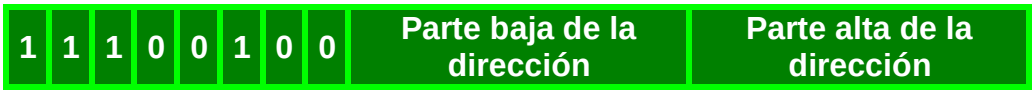
A1.12.9. CPO Llamar si la paridad es impar

Instrucción	CPO dir
Bits afectados	
Direccionamiento	Inmediato / Registro indirecto

Descripción

CPO chequea el bit de paridad. Si el bit esta a 0, CPO carga el contenido del contador de programa en el stack y salta a la dirección especificada en DIR. Si el bit está a 1 el programa continúa su desarrollo normal.

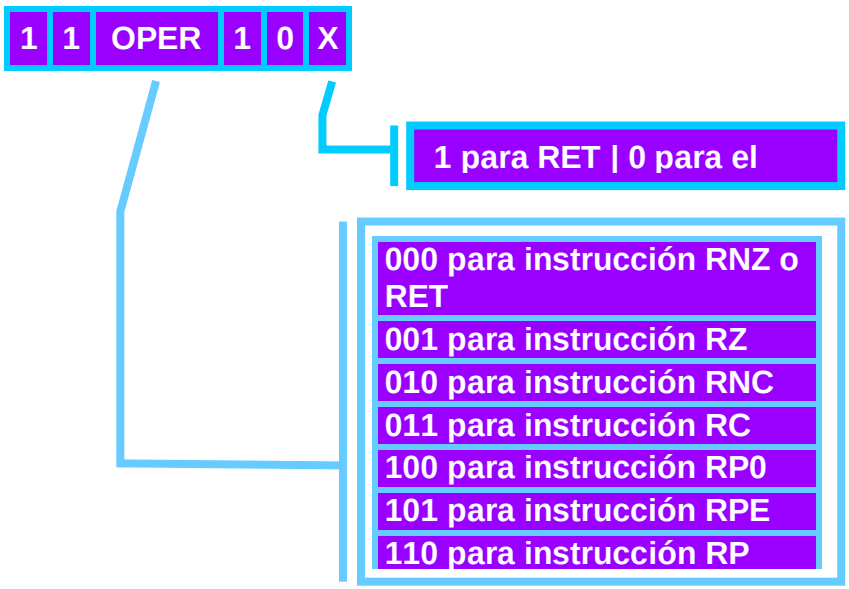
Formato



A1.13. Instrucciones de retorno desde subrutinas

A continuación se describen las instrucciones utilizadas para realizar un retorno desde las subrutinas. Estas instrucciones colocan en el contador de programa la última dirección puesta en la pila, haciendo que la ejecución del programa continúe en esta dirección.

Las instrucciones de este tipo utilizan un byte en el formato siguiente:



Ciertas instrucciones de este tipo realizan la operación de retorno bajo ciertas condiciones específicas. Si la condición especificada se cumple (es verdadera), tendrá lugar la operación de retorno. Por el contrario, si la condición no se cumple, la ejecución del programa continuará en la próxima instrucción en secuencia.

A1.13.1. RET Retorno incondicional

Instrucción	RET
Bits afectados	
Direccionamiento	Registro indirecto

Descripción

Se realiza una operación de retorno incondicional.

La instrucción RET echa fuera dos bytes de datos del stack y los mete en el registro contador de programa. El programa continúa entonces en la nueva dirección. Normalmente RET se emplea conjuntamente con CALL.

Formato

A1.13.2. RC Retorno si hay acarreo

Instrucción	RC
Bits afectados	
Direccionamiento	Registro indirecto

Descripción

La instrucción RC comprueba el estado del bit de acarreo. Si tiene un 1 (indicando que hay acarreo) la instrucción saca dos bytes del stack y los mete en el contador de programa. El programa continúa en la nueva dirección suministrada. Si el bit es 0, el programa continúa en la siguiente instrucción de la secuencia normal.

Formato

A1.13.3. RNC Retorno si no hay acarreo

Instrucción	RNC
Bits afectados	
Direccionamiento	Registro indirecto

Descripción

La instrucción RNC comprueba el bit de acarreo. Si está a 0 indicando que no hay acarreo, la instrucción echa fuera del stack dos bytes y los carga en el contador de programa. Si el bit está a 1 continúa el ciclo normal.

Formato

A1.13.4. RZ Retorno si hay cero

Instrucción	RZ
Bits afectados	
Direccionamiento	Registro indirecto

Descripción

La instrucción RZ comprueba el bit de cero. Si está a 1, indicando que el contenido del acumulador es cero, la instrucción echa fuera del stack dos bytes y los carga en el contador de programa. Si el bit está a 0, continúa el ciclo normal.

Formato

A1.13.5. RNZ Retorno si no hay cero

Instrucción	RNZ
Bits afectados	
Direccionamiento	Registro indirecto

Descripción

La instrucción RNZ comprueba el bit cero. Si está a 0, indicando que el contenido del acumulador no es cero, la instrucción echa fuera del stack dos bytes y los carga en el contador de programa. Si el bit está a 1, continúa el ciclo normal.

Formato

A1.13.6. RM Retorno si hay signo negativo

Instrucción	RM
Bits afectados	
Direccionamiento	Registro indirecto

Descripción

La instrucción RM comprueba el bit de signo. Si tiene un 1, indicando dato negativo en el acumulador, la instrucción echa dos bytes fuera del stack y los mete en el contador de programa. Si el bit tiene 0, continúa el programa normal con la siguiente instrucción.

Formato

A1.13.7. RP Retorno si hay signo positivo

Instrucción	RP
Bits afectados	
Direccionamiento	Registro indirecto

Descripción

La instrucción RP comprueba el bit signo. Si está a 0, indicando que el contenido del acumulador es positivo, la instrucción echa fuera del stack dos bytes y los carga en el contador de programa. Si el bit está a 1 continúa el ciclo normal.

Formato

A1.13.8. RPE Retorno si la paridad es par

Instrucción	RPE
Bits afectados	
Direccionamiento	Registro indirecto

Descripción

La instrucción RPE comprueba el bit de paridad. Si está a 1, indicando que existe paridad, la instrucción echa fuera del stack dos bytes y los carga en el contador de programa. Si el bit está a 0 continúa el ciclo normal. (Existe paridad si el byte que está en el acumulador tiene un número par de bits, colocándose el bit de paridad a 1 en este caso).

Formato

A1.13.9. RPO Retorno si la paridad es impar

Instrucción	RPO
Bits afectados	
Direccionamiento	Registro indirecto

Descripción

La instrucción RPO comprueba el bit de paridad. Si está a 0, indicando que no hay paridad, la instrucción echa fuera del stack dos bytes y los carga en el contador de programa. Si el bit está a 1, continúa el ciclo normal.

Formato

A1.14. Instrucción RST

Instrucción	RST exp
Bits afectados	
Direccionamiento	Registro indirecto

Descripción

Es una instrucción CALL para usar con interrupciones. RST carga el contenido del contador de programa en el stack, para proveerse de una dirección de retorno y salta a una de las "ocho" direcciones determinadas previamente.

Un código de tres bits incluido en el código de operación de la instrucción RST especifica la dirección de salto. Esta instrucción es empleada por los periféricos cuando intentan una interrupción.

Para volver a la instrucción en que ha tenido lugar la interrupción, se debe utilizar una instrucción de RETORNO.

Formato

donde EXP es un número binario entre 000B y 111B.

Por tanto, según la combinación de 0 y 1 que demos a EXP obtendremos los distintos formatos y las distintas direcciones de las interrupciones, que serán:

FORMATO		CONTADOR DE PROGRAMA	
1100 0111	→ C7H	0000 0000 0000 0000	→ 0000H

1100 1111	→	CFH	0000 0000 0000 1000	→	0008H
1101 0111	→	D7H	0000 0000 0001 0000	→	0010H
1101 1111	→	DFH	0000 0000 0001 1000	→	0018H
1110 0111	→	E7H	0000 0000 0010 0000	→	0020H
1110 1111	→	EFH	0000 0000 0010 1000	→	0028H
1111 0111	→	F7H	0000 0000 0011 0000	→	0030H
1111 1111	→	FFH	0000 0000 0011 1000	→	0038H

Ejemplos

1. La instrucción

   RST 3

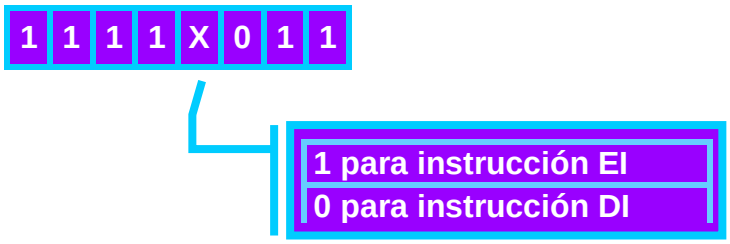
   llama a la subrutina de la posición 0018H.
2. La instrucción

   RST 10

   es una instrucción ilegal ya que el argumento de RST debe ser un número entre 0 y 7.

A1.15. Instrucciones del FLIP-FLOP de interrupción

En este apartado estudiamos las instrucciones que operan directamente sobre el flip-flop de actuación de interrupciones. Todas estas instrucciones ocupan un byte siguiendo el formato que se muestra:



A1.15.1. EI Activar interrupciones

Instrucción	EI
Bits afectados	
Direccionamiento	

Descripción

La instrucción EI pone en servicio el sistema de interrupciones a partir de la siguiente instrucción secuencial del programa. Esta instrucción activa el flip-flop INTE.

Se puede desconectar el sistema de interrupciones poniendo una instrucción DI al principio de una secuencia, puesto que no se puede predecir la llegada de una interrupción.

Al final de la secuencia se incluye la instrucción EI que vuelve a habilitar el

sistema de interrupciones. (RESET también pone fuera de servicio el sistema de interrupciones además de poner el contador de programa a cero).

Formato

A1.15.2. DI Desactivar interrupciones

Instrucción	DI
Bits afectados	
Direccionamiento	

Descripción

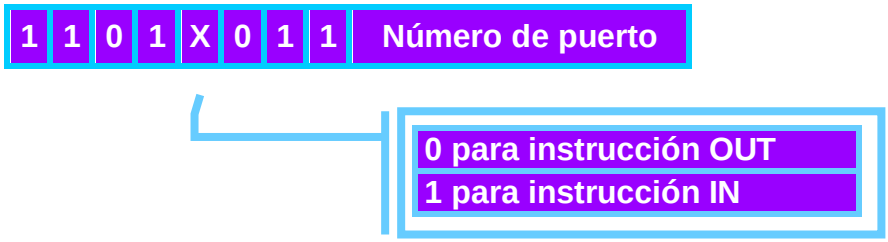
Esta instrucción desactiva el flip-flop INTE. Después de la ejecución de una instrucción DI, el sistema de "interrupciones" esta sin posibilidad de ponerse en marcha. En aplicaciones que empleen las interrupciones, la instrucción DI se emplea solamente cuando una determinada secuencia no debe ser interrumpida. Por ejemplo, se puede poner fuera de servicio el sistema de interrupciones incluyendo una instrucción DI el principio del código de secuencia.

La interrupción TRAP del 8085 no puede ser puesta fuera de servicio. Esta interrupción especial esta prevista para serios problemas que pueden presentarse independientemente del bit de interrupción (fallo de alimentación, etc.).

Formato

A1.16. Instrucciones de Entrada / Salida

Ahora nos ocuparemos de las instrucciones que hacen que los datos entren o salgan de la CPU. Estas instrucciones ocupan dos bytes en la forma siguiente:



El número de puerto viene definido por el hardware del sistema, no estando bajo el control del programador. Este, únicamente selecciona uno de ellos para realizar la correspondiente operación.

A1.16.1. IN Entrada

Instrucción	IN port
Bits afectados	
Direccionamiento	Directo

Descripción

La instrucción IN PORT lee los 8 bits de datos que hay en el "PORT" especificado y los carga en el acumulador. El operando debe ser un número o una expresión que produzca un valor comprendido entre 00H y FFH.

Formato

Ejemplos

1. La instrucción

   IN 2

   deposita en el acumulador los datos de entrada del puerto 2.

2. La instrucción

   IN 3*2

   deposita en el acumulador los datos de entrada del puerto 6.

A1.16.2. OUT Salida

Instrucción	OUT port
Bits afectados	
Direccionamiento	Directo

Descripción

OUT PORT pone el contenido del acumulador en el bus de datos de 8 bits del puerto seleccionado. El número de puertos oscila de 0 a 255 y es duplicado en el bus de direcciones. Es la lógica externa la encargada de seleccionar el puerto y aceptar el dato de salida. El operando (PORT) debe especificar el número del puerto de salida seleccionado.

Formato

Ejemplos

1. La instrucción

OUT 2

envía el contenido del acumulador al puerto de salida número 2.

2. La instrucción

OUT 3*2

envía el contenido del acumulador al puerto de salida número 6.

A1.17. Instrucción de alto HLT

Instrucción HLT
Bits afectados
Direccionamiento

Descripción

La instrucción HLT detiene el procesador.

El contador de programa contiene la dirección de la próxima instrucción secuencial. Por otro lado los bits y registros permanecen inactivos. Una vez en estado de parada el procesador puede volver a ser arrancado solamente por un acontecimiento externo, es decir una interrupción. Por tanto debemos asegurarnos que las interrupciones estén en disposición de ser activadas antes de ejecutar la instrucción HLT.

Si se ejecuta HLT estando las interrupciones fuera de servicio, la única manera de volver arrancar el procesador será mediante un RESET o a través de la interrupción TRAP.

El procesador puede salir temporalmente del estado de parada para servir un acceso directo a memoria, sin embargo terminado el acceso directo vuelve al estado de parada.

Un propósito básico de la instrucción HLT es permitir una pausa al procesador mientras espera por la interrupción de un periférico.

Formato

0	1	1	1	0	1	1	0

A1.18. Instrucción RIM para la lectura de la máscara de interrupciones

Instrucción	RIM
Bits afectados	
Direccionamiento	

Descripción

RIM carga los 8 bits de datos siguientes en el acumulador:

bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
SID	I7.5	I6.5	I5.5	IE	M7.5	M6.5	M5.5

donde:

- SID: Bit presente en la entrada serie.
- I7.5: Interrupción 7.5 pendiente si está a 1.
- I6.5: Interrupción 6.5 pendiente si está a 1.
- I5.5: Interrupción 5.5 pendiente si está a 1.
- IE: Las interrupciones son autorizadas si está a 1.
- M7.5: La interrupción 7.5 está prohibida si está a 1.
- M6.5: La interrupción 6.5 está prohibida si está a 1.

- M5.5: La interrupción 5.5 está prohibida si está a 1.

Formato

A1.19. Instrucción SIM para posicionar la máscara de interrupciones

Instrucción SIM
Bits afectados
Direccionamiento

Descripción

SIM es una instrucción de usos múltiples que utiliza el contenido del acumulador para posicionar el enmascaramiento de interrupciones para las RST 5.5, RST 6.5, RST 7.5; pone a cero el flanco sensitivo de la entrada RST 7.5 y saca el bit 7 del acumulador al latch de datos de salida serie. La estructura de la instrucción SIM es como sigue:

bit 7	bit 6	bit 5	Bit 4	bit 3	bit 2	bit 1	bit 0
SOD	SOE	X	R7.5	MSE	M7.5	M6.5	M5.5

donde:

- SOD: Bit a presentar sobre la salida serie.
- SOE: La salida serie está autorizada si está a 1.
- R7.5: Reset de RST 7.5. Si es 1 el flip-flop se pone a 0.
- MSE: Si es 1 los enmascarados están autorizados.
- M7.5: La interrupción 7.5 queda prohibida si está a 1.
- M6.5: La interrupción 6.5 queda prohibida si está a 1.

- M5.5: La interrupción 5.5 queda prohibida si está a 1.

Si el bit 3 se pone a 1, la función poner "mask" pasa a estar permitida.

Los bits 0 al 2 ponen en servicio la correspondiente interrupción RST colocando un 0 en la interrupción que deseamos habilitar. Si colocamos un 1 en alguno de los bits 0 al 2, la interrupción correspondiente no se cumplirá.

Si el bit 3 tiene un 0, los bits 0 al 2 no tienen efectos. Se debe usar esta peculiaridad para enviar un bit de salida serie sin afectar al enmascaramiento de las interrupciones. (La instrucción DI anula la SIM).

Si el bit 6 (SOE) se pone a 1 se habilita la función de salida serie.

El bit 7 se sitúa en la salida SOD donde puede ser tratado por los aparatos periféricos. Si el bit 6 se pone a cero, el bit 7 no tendrá efecto alguno, siendo ignorado.

Formato