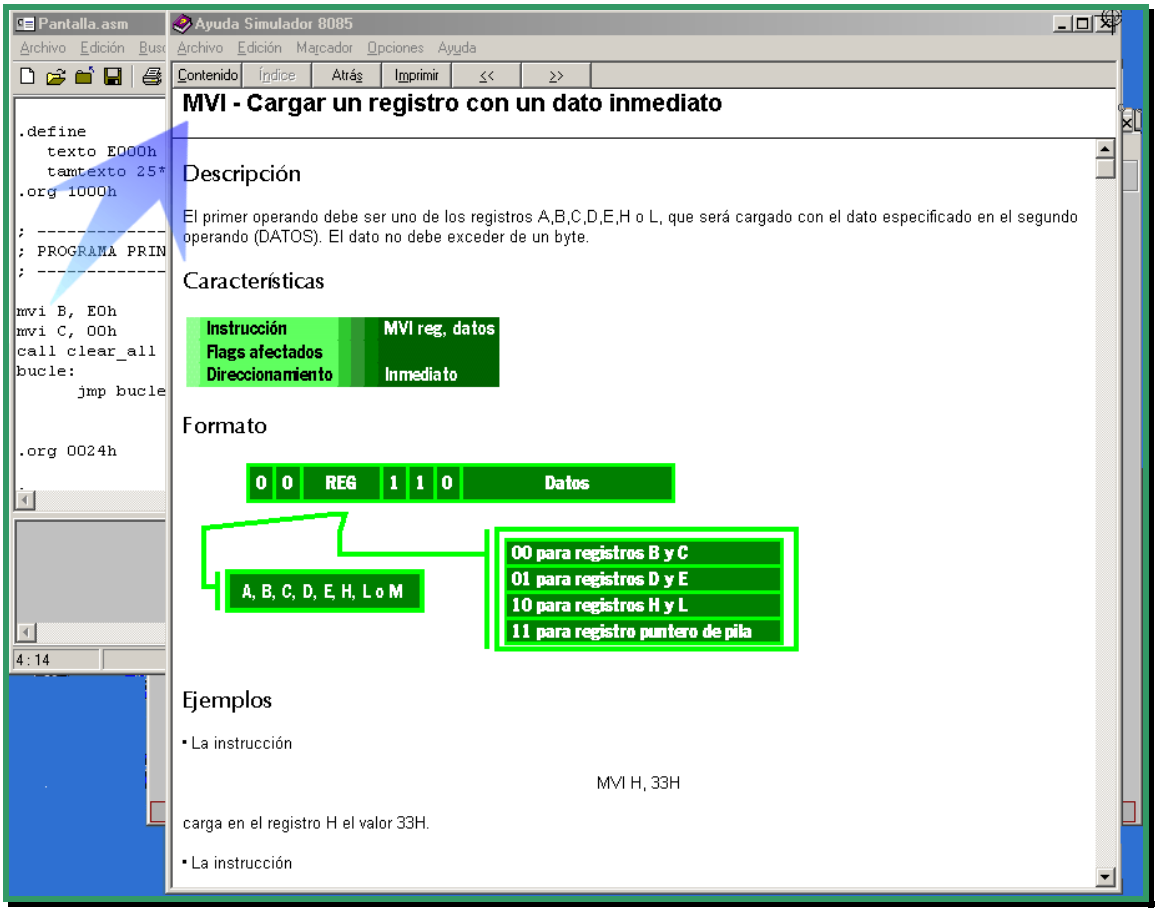
Conjunto de instruções do 8085

Conteúdo

A.1. Introdução

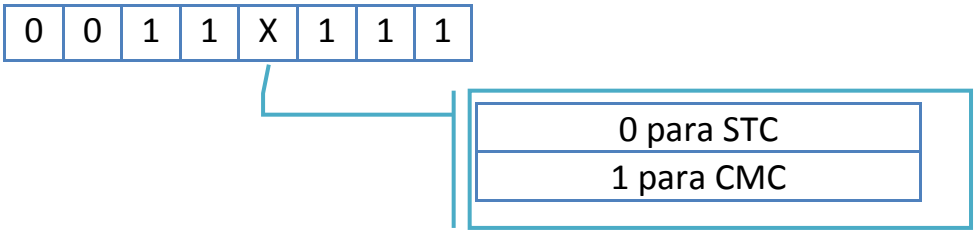
Ainda que o conjunto de instruções pode ser encontrado em qualquer livro que trate deste tema especificamente, neste documento se mostra um “dicionário” das mesmas. A descrição se propõe funcionalmente para uma localização imediata e todas estão descritas de forma detalhada.

Se desejar fazer uma busca alfabética das instruções pode-se usar a ajuda incorporada no programa simulador.



A.2. Instruções do bit de carry

Na continuação serão descritas as instruções que operam diretamente sobre o bit de carry. Estas instruções utilizam um byte e são expressas da seguinte forma:



A2. 1. CMC - Complementar carry

Instrução	CMC
Bits afetados	CY
Endereçamento	

Descrição

Se o bit de carry for 0, ficará em 1. Se for um 1, ficará em 0.

Formato

0	0	1	1	1	1	1	1

A2. 2. STC - Ativar carry

Instrução	STC
Bits afetados	CY
Endereçamento	

Descrição

O bit de carry é colocado em 1.

Formato

0	0	1	1	0	1	1	1

A.3. Instruções de um registrador

Na continuação serão descritas as instruções que operam sobre um só registrador ou posição de memória. Se for especificada uma referencia a memória, o endereço da mesma será dado pelo conteúdo dos registradores H y L, onde o registrador H contém os oito bits mais significativos do endereço r o registrador L os restantes.

A3. 1. INR - Incrementar registrador ou memória

Instrução	INR reg
Bits afetados	Z, S, P, AC
Endereçamento	Registrador indireto

Descrição

O conteúdo do registrador ou posição de memória especificados é incrementado em uma unidade.

Formato

0	0	REGISTRADOR	1	0	0

000 para Registrador B
001 para Registrador C
010 para Registrador D
011 para Registrador E
100 para Registrador H
101 para Registrador L
110 para MEMORIA
111 para ACUMULADOR

Exemplo

Se o registrador A contém 98H, a instrução:

INR A

Fará com que este registrador contenha a quantidade 99H.

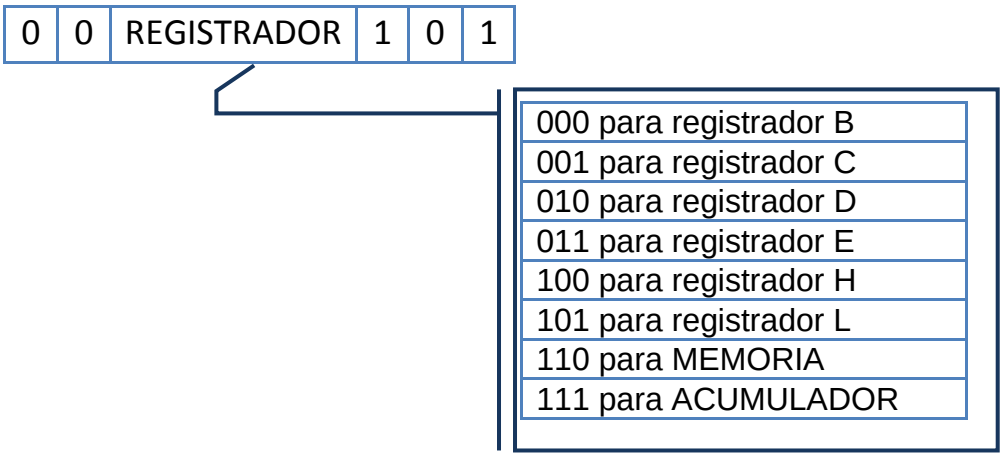
A.3.2. DCR - Decrementa registrador ou memória

Instrução	DCR reg
Bits afetados	Z, S, P, AC
Endereçamento	Registrador

Descrição

O conteúdo do registrador ou posição de memória especificados é decrementado em uma unidade.

Formato



Exemplo

Se o registrador A contém 99H, a instrução:

DCR A

Esta fará com que este registrador contenha a quantidade 98H.

A.3.3. CMA - Complementar acumulador

Instrução	CMA
Bits afetados	
Endereçamento	

Descrição

Cada um dos bits do acumulador é complementado (operação denominada de um).

Formato

0	0	1	0	1	1	1	1

A.3.4. DAA - Ajuste decimal do acumulador

Instrução	DAA
Bits afetados	Z, S, P, CY, AC
Endereçamento	Registrador

Descrição

O número hexadecimal de 8 bits contido no acumulador é ajustado como dois dígitos de 4 bits codificados em BCD, segundo o seguinte processo:

(1). Se os quatro bits menos significativos do acumulador representam um número maior que 9, ou se o bit de carry auxiliar é igual a um, o acumulador é incrementado em seis unidades. Caso contrário se não se apresentarem tais condições, o conteúdo do acumulador não irá variar.

(2). Se os quatro bits mais significativos do acumulador representam agora um número maior que 9, ou se o bit de carry for um, os quatro bits mais significativos são incrementados em seis unidades. Assim mesmo, se não tiverem lugar as circunstancias expostas, o conteúdo do acumulador não será incrementado.

Se ocorrer carry dos quatro bits menos significativos durante o passo (1), o bit de carry auxiliar é colocado em 1; se não ocorrer, será colocado em 0. Por outro lado, se ocorrer carry dos quatro bits mais significativos durante o passo (2), será ativado o bit de carry, colocando-se em zero se não for produzido carry.

Formato

0	**0**	**1**	**0**	**0**	**1**	**1**	**1**

Nota

Esta instrução é utilizada nas operações de soma de números decimais. É a única instrução cuja operação depende do bit de carry auxiliar.

Exemplo

Suponhamos que queremos realizar uma soma decimal de dois números (40 + 80). Ambos os bits de carry estão em zero.

A seqüência de instruções poderia ser:

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

A instrução DAA opera da seguinte forma:

1. Como o conteúdo dos bits [0 – 3] do acumulador é menor que 9 e o bit de carry auxiliar é zero, o conteúdo do acumulador não varia.

2. Como os 4 bits mais significativos do acumulador representam um número maior que 9 estes 4 bits são incrementados em 6 unidades, colocando em um o bit de carry.

Acumulador		C0H		1	1	0	0	0	0	0	0	CY = 0
+ 6		60H		0	1	1	0	0	0	0	0	CY = 0

Novo acumulador		20H		0	0	1	0	0	0	0	0	CY = 1

A4. Instrução NOP

Instrução	NOP
Bits afetados	
Endereçamento	

Descrição

Não é realizada nenhuma operação.

Formato

0	**0**	**0**	**0**	**0**	**0**	**0**	**0**

A5. Instruções de transferência de dados

Esta série de instruções transfere dados entre os registradores, a memória e o acumulador. Ocupam um byte da forma seguinte:

A5. 1. MOV - Movimento

Instrução	MOV reg, reg
Bits afetados	
Endereçamento	Registrador o registrador indireto

Descrição

Podemos distinguir 3 casos:

(A). Transferência entre registradores (Endereçamento registrador).

(B). Transferência desde a memória (Endereçamento registrador indireto).

(C). Transferência para a memória (Endereçamento registrador indireto).

(A). MOV R_1, R_2

O conteúdo do registrador R_2 é transferido para os registradores R_1. R_1 e R_2 podem ser os registradores B, C, D, E, H, L ou o acumulador A.

(B). MOV R, M

O conteúdo do endereço de memória, cujo endereço está nos registradores H-L, é transferido para o registrador R. R pode ser qualquer dos registradores A, B, C, D, E, H ou L.

(C). MOV M, R

O conteúdo do registrador R é transferido para o endereço de memória indicada pelos registradores H-L.

Formato

0	1	Registrador	Registrador

000 para registrador B
001 para registrador C
010 para registrador D
011 para registrador E
100 para registrador H
101 para registrador L
110 para MEMORIA
111 para ACUMULADOR

Exemplos

1. Suponhamos que o registrador B contém 00H e o registrador C contém 30H. A instrução:

MOV B,C

Armazenará 30H no registrador B.

2. Suponhamos que o registrador H contém 00H e o registrador L contém 30H. A instrução:

MOV M, A

Armazenará o conteúdo do acumulador na posição de memória 0030H.

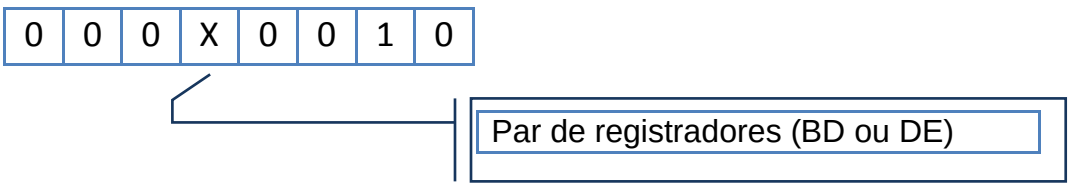
A5.2. STAX - Armazenar o conteúdo do acumulador

Instrução	STAX rp
Bits afetados	
Endereçamento	Registrador indireto

Descrição

O conteúdo do acumulador é armazenado na posição de memória especificada pelos registradores B e C, ou os registradores D e E.

Formato



Exemplo

Se o registrador B contém 3FH e o registrador C contém 16H, a instrução

STAX B

Armazenará o conteúdo do acumulador na posição de memória 3F16H.

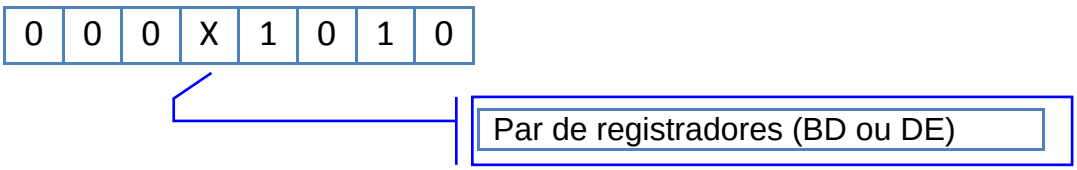
A5.3. LDAX - Carregar o acumulador

Instrução	LDAX rp
Bits afetados	
Endereçamento	Registrador indireto

Descrição

O conteúdo da posição de memória especificada pelos registradores B e C, ou os registradores D e E, substituem o conteúdo do acumulador.

Formato



Exemplo

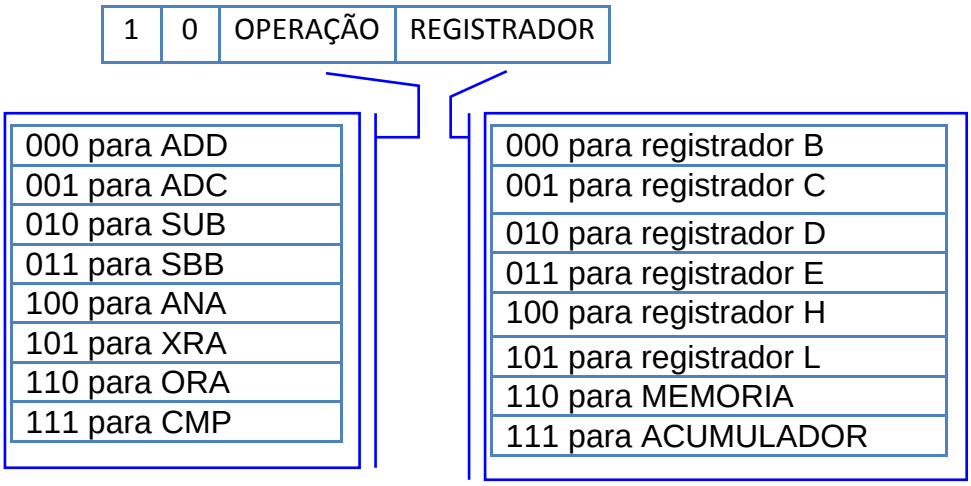
Se o registrador D contém 3FH e o registrador E contém 16H, a instrução:

LDAX D

Carregará no acumulador o conteúdo da posição de memória 3F16H.

A6. Instruções de registrador ou memória e acumulador

Na continuação vamos a ver as instruções que operam com o conteúdo do acumulador e de um dos registradores ou posição de memória. Estas instruções ocupam um byte da seguinte forma:



A instrução opera sobre o conteúdo do acumulador, com a quantidade definida pelo registrador especificado por REGISTRADOR. Se for especificada uma referencia a memória, a quantidade utilizada pela instrução é a correspondente a posição de memória determinada pelos registradores H e L, nos quais o registrador H guarda os 8 bits mais significativos, e o registrador L, os 8 restantes. Tanto o conteúdo do registrador como da posição de memória não variam ao finalizar a instrução, guardando-se o resultado no acumulador.

A6.1. ADD - Somar registrador ou memória ao acumulador

Instrução	ADD reg
Bits afetados	Z, S, P, CY, AC
Endereçamento	Registrador

Descrição

O conteúdo do registrador ou posição de memória especificados são somados ao conteúdo do acumulador, usando aritmética de complemento de dois. O resultado é guardado no acumulador.

Formato

1	0	0	0	0	REGISTRADOR

Exemplos

1. Se o registrador B contém o valor 3AH e o acumulador contém 6CH, a instrução:

   ADD B

   Realiza a seguinte soma:

Registrador B	3AH	0	0	1	1	1	0	1	0
Acumulador	6CH	0	1	1	0	1	1	0	0
Novo acumulador	A6H	1	0	1	0	0	1	1	0

2. A instrução:

   ADD A

   Duplica o conteúdo do acumulador.

A6.2. ADC - Somar registrador ou memória ao acumulador com carry

Instrução	ADC reg

Bits afetados	Z, S, P, CY, AC
Endereçamento	Registrador indireto

Descrição

O conteúdo do registrador ou posição de memória especificados mais o conteúdo do bit de carry são somados ao conteúdo do acumulador.

Formato

1	0	0	0	1	REGISTRADOR

Exemplo

Suponhamos que o registrador B contém o valor 30H, o acumulador 76H, e o bit de carry está posto em zero. A instrução:

ADC C

Realizará a seguinte soma:

Registrador B	30H	0	0	1	1	0	0	0	0
Acumulador	76H	0	1	1	1	0	1	1	0
Bit de carry									0
Novo acumulador	A6H	1	0	1	0	0	1	1	0

O novo conteúdo do acumulador será A6H, mesmo que todos os bits de condição fiquem postos em zero exceto os de sinal e paridade.

Se o bit de carry tivesse valor 1 antes de realizar a operação, a operação iria ocorrer da seguinte forma, uma soma:

Registrador B	30H	0	0	1	1	0	0	0	0
Acumulador	76H	0	1	1	1	0	1	1	0

Bit de carry								1
Novo acumulador	A7H	1	0	1	0	0	1	1	1

O acumulador conteria agora A7H e todos os bits de condição exceto o de sinal, estariam postos em zero.

A6.3. SUB - Subtrair registrador ou memória do acumulador

Instrução	SUB reg
Bits afetados	Z, S, P, CY, AC
Endereçamento	Registrador

Descrição

O conteúdo do registrador ou posição de memória especificados serão subtraídos do conteúdo do acumulador, usando aritmética de complemento de dois. O resultado será guardado no acumulador.

Se não ocorrer carry no bit de maior peso, o bit de carry será colocado em 1, e vice-versa, ao contrario do que ocorre com a operação de soma

Formato

1	0	0	1	0	REGISTRADOR

Exemplos

Antes de entrar nos exemplos devemos recordar que subtrair utilizando aritmética de complemento de dois equivale a complementar cada bit do segundo operando e somar 1.

1. Se o acumulador contém 60H e o registrador E contém 28H, a instrução:

SUB E

Realizará a seguinte operação de subtração:

Acumulador	60H	0	1	1	0	0	0	0	0
+(–Registrador B)	+(–28H)	1	1	0	1	0	1	1	1
Bit de carry									1
Novo acumulador	38H	0	0	1	1	1	0	0	0

Ao final da operação o acumulador conterá 38H e o bit de carry será colocado em zero devido ao fato que tenha ocorrido carry no bit mais significativo.

2. A instrução:

SUB A

Fará a subtração do acumulador dele mesmo, obtendo-se um resultado de zero. É utilizada em muitas ocasiões para colocar em zero o bit de carry e o acumulador.

A6.4. SBB - Subtrair registrador ou memória do acumulador com carry

Instrução	SBB reg
Bits afetados	Z, S, P, CY, AC
Endereçamento	Registrador indireto

Descrição

O valor do bit de carry é somado internamente ao conteúdo do registrador ou posição de memória especificados. Este valor se subtrai do acumulador usando aritmética de complemento de dois.

Esta instrução é de grande utilidade na realização de subtrações de vários bytes, pois leva em conta o valor positivo ou negativo da subtração anterior.

Formato

1	0	0	1	1	REGISTRADOR

Exemplo

Se o registrador C contém 08H, o acumulador armazena 05H e o bit de carry está ativado, a instrução:

SBB C

Efetua a seguinte operação:

1. 08H + bit de carry = 09H.
2. Complemento de dois de 09H = 11110111 (F7H)
3. O anterior se soma ao acumulador:

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

4. Não há carry no final pelo que o bit de carry ficará em um. Os bits de sinal e paridade estão postos em um mesmo que o bit de zero está em zero.

A6.5. ANA - Função lógica AND entre registrador ou memória com acumulador

	Instrução				ANA reg
	Bits afetados				Z, S, P, CY, AC
	Endereçamento				Registrador indireto

Descrição

Realiza a função lógica AND bit a bit entre o conteúdo do registrador ou posição de memória especificados e o conteúdo do acumulador. O bit de carry é colocado em zero.

Formato

1	0	1	0	0	REGISTRADOR

Nota

A tabela verdade da função lógica AND é:

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

Exemplo

Se o registrador B contém 6CH e o acumulador armazena 3AH, a instrução

ANA B

Realiza a seguinte operação:

Acumulador	3AH	0	0	1	1	1	0	1	0
Registrador B	6CH	0	1	1	0	1	1	0	0
Novo acumulador	28H	0	0	1	0	1	0	0	0

A6.6. XRA - Função lógica OR-EXCLUSIVO entre registrador ou memória com acumulador

Instrução	XRA reg
Bits afetados	Z, S, P, CY, AC
Endereçamento	Registrador

Descrição

É realizada a função lógica OU-EXCLUSIVO bit a bit entre o conteúdo do registrador ou posição de memória especificados e o conteúdo do acumulador, guardando-se o resultado neste último.

Formato

1	**0**	**1**	**0**	**1**	**REGISTRADOR**

Nota

A tabela verdade da função lógica OU-EXCLUSIVO é:

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

Exemplos

1. Se o registrador B contém 6CH e o acumulador armazena 3AH, a instrução

XRA B

Realiza a seguinte operação:

Acumulador	3AH	0	0	1	1	1	0	1	0
Registrador B	6CH	0	1	1	0	1	1	0	0
Novo acumulador	56H	0	1	0	1	0	1	1	0

2. A função OU-EXCLUSIVO de qualquer bit com um dá lugar ao complemento dele mesmo. Assim, se o acumulador contém todos em um, a instrução

XRA B

Produz o complemento de um do conteúdo do registrador B, e o guarda no acumulador.

3. Em algumas ocasiões, um byte é utilizado para refletir os estados de certas condições dentro de um programa, onde cada um dos oito bits pode responder a uma determinada condição de falso ou verdadeiro, atuando ou inibindo, etc. Mediante a função OU-EXCLUSIVO podemos determinar quantos bits da palavra irão mudar de estado em um determinado lapso de tempo.

A6.7. ORA - Função lógica OR entre registrador ou memória com o acumulador

Instrução	ORA reg
Bits afetados	Z, S, P, CY, AC
Endereçamento	Registrador indireto

Descrição

É realizada a função lógica AND bit a bit entre o conteúdo do registrador ou posição de memória especificados e o conteúdo do acumulador, ficando neste último o resultado. O bit de carry é colocado em zero.

Formato

1	0	1	1	0	REGISTRADOR

Nota

A tabela verdade da função lógica OR é:

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

Exemplo

Qualquer que seja que a função OR de qualquer bit com um dá como resultado um, e de qualquer bit com zero, o deixa invariável, esta função é utilizada freqüentemente para por em um grupos de bits.

Se o registrador B contém OFH e o acumulador armazena 33H, a instrução

ORA B

Realiza a seguinte operação:

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

Neste exemplo se garante que os quatro bits de menor peso do acumulador são “um”, mesmo que os demais permaneçam invariáveis.

A6.8. CMP - Comparar registrador ou memória com acumulador

Instrução	CMP reg
Bits afetados	Z, S, P, CY, AC
Endereçamento	Registrador indireto

Descrição

O conteúdo do registrador ou posição de memória especificados é comparado com o conteúdo do acumulador. Esta comparação é realizada subtraindo internamente o conteúdo do registrador ao do acumulador, permanecendo este invariável, e colocando os bits de condição em função do resultado. Concretamente, o bit de zero é colocado em um se as quantidades comparadas forem iguais, e colocados em zero se forem diferentes. Quando se realiza uma operação de subtração, o bit de carry será colocado em um se não ocorrer carry no bit 7, indicando que o conteúdo do registrador ou posição de memória é maior que o conteúdo do acumulador, e será colocado em zero se for maior que o acumulador.

Se as duas quantidades tiverem sinal diferentes, o carry adota o valor contrário ao anteriormente exposto.

Formato

1	0	1	1	1	REGISTRADOR

Exemplos

Na continuação vamos abordar 3 exemplos desta operação.

1. Si o acumulador armazena 0AH e o registrador B contém 05H, quando é realizada a instrução

   CMP B

   Tem lugar a seguinte subtração interna:

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

Existe carry nol bit 7 pelo que o bit de carry é colocado em zero. O acumulador segue armazenando 0AH e o registrador B, 05H. Não obstante, o bit de carry é colocado em zero, assim como o bit de zero, indicando que o conteúdo do registrador B é menor que o acumulador.

2. Agora o acumulador tem o valor 02H. Então:

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

Neste o bit de carry é colocado em um (não existe carry do bit 7) e o bit de zero estará em zero, indicando que o conteúdo do registrador B é maior que o acumulador.

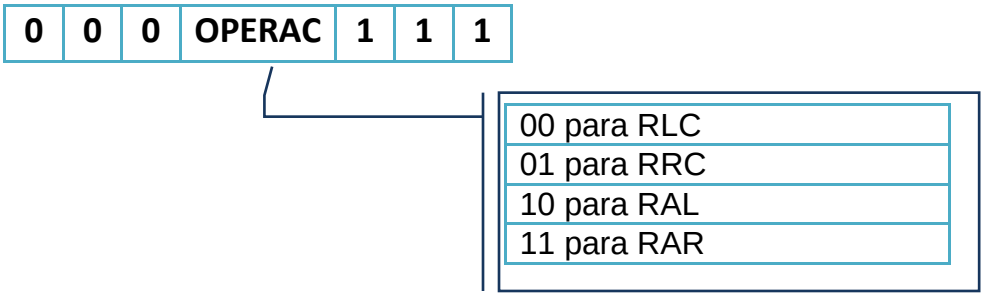
3. Por último suponhamos um valor –1BH para o acumulador. Nesta situação:

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

Aqui o bit de carry está em zero. Como os dois números diferem em sinal, o dato de que o bit de carry esteja em zero indica que o conteúdo do registrador B é maior que o do acumulador, ao contrario de como ocorria no exemplo anterior.

A7. Instruções de rotação do acumulador

Na continuação serão descritas as instruções que provocam uma rotação do conteúdo do acumulador. Esta operação unicamente pode realizar-se com o acumulador, não com nenhum registrador ou posição de memória.



A7.1. RLC - Deslocar o acumulador para a esquerda

Instrução	RLC
Bits afetados	CY
Endereçamento	

Descrição

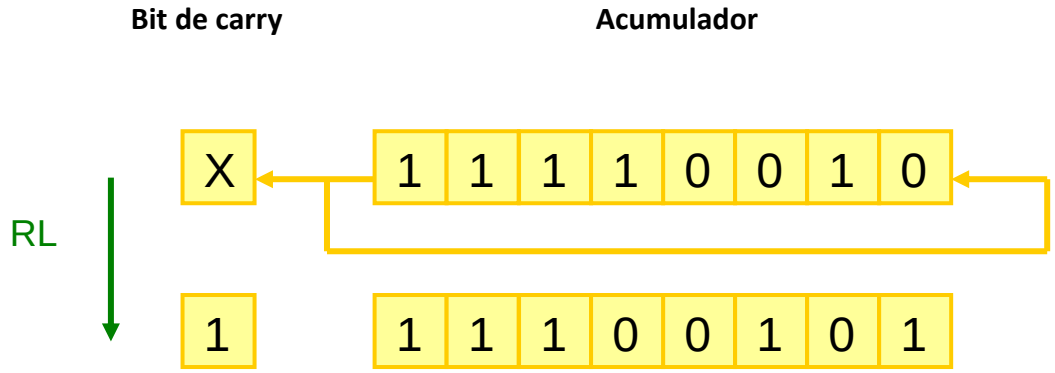
RLC rota um bit para a esquerda todo o conteúdo do acumulador, transferindo o bit de mais alta ordem para o bit de carry e ao mesmo tempo para a posição de menor ordem do acumulador.

Formato

0	0	0	0	0	1	1	1

Exemplo

Suponhamos que o acumulador contém 82H.



A7.2. RRC - Deslocar o acumulador para a direita

Instrução	RRC
Bits afetados	CY
Endereçamento	

Descrição

RRC rota o conteúdo do acumulador um bit para a direita, transferindo o bit de mais baixa ordem para a posição de mais alta ordem do acumulador, além disso põe o bit de carry igual ao bit de menor ordem do acumulador.

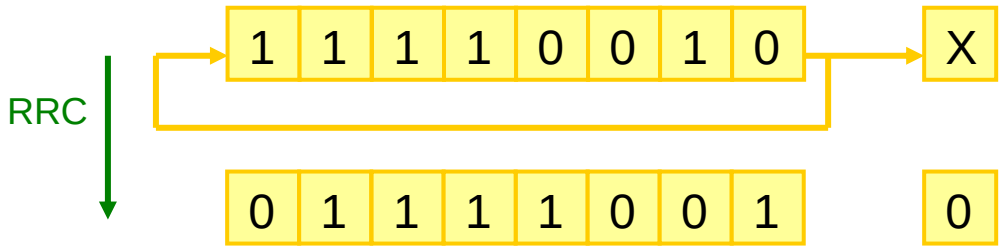
Formato

0	**0**	**0**	**1**	**0**	**1**	**1**	**1**

Exemplo

Suponhamos que o acumulador contém F2H. A instrução RRC realizará a seguinte operação sobre o acumulador e o bit de carry:

Acumulador **Bit de carry**



A7.3. RAL - Deslocar o acumulador para a esquerda através do bit de carry

Instrução	RAL
Bits afetados	CY
Endereçamento	

Descrição

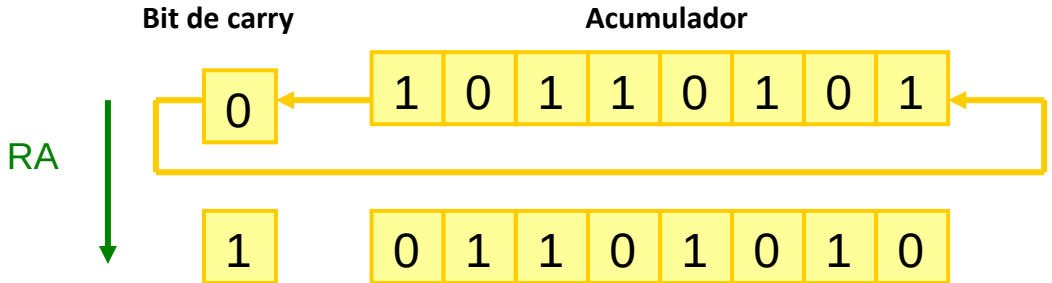
RAL faz girar o conteúdo do acumulador e o bit de carry um espaço de um bit para a saída (esquerda). O bit de carry que é tratado como se fosse do acumulador, se transfere o bit de menor ordem do acumulador. O bit de maior ordem do acumulador se transfere para o bit de carry. Não tem operandos.

Formato

0	**0**	**0**	**1**	**0**	**1**	**1**	**1**

Exemplo

Suponhamos que o acumulador contém B5H. A instrução RAL efetuará as seguintes modificações no registrador acumulador e no bit de carry:



A7.4. RAR - Deslocar o acumulador para a direita através do bit de carry

Instrução	RAR
Bits afetados	CY
Endereçamento	

Descrição

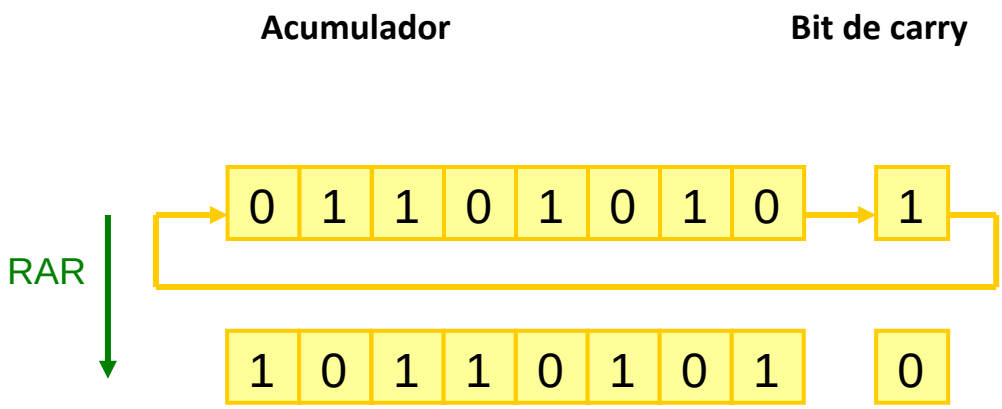
RAR rota o conteúdo do acumulador e do bit de carry 1 bit de posição para a direita. O bit de carry que é tratado como se fosse parte do acumulador se transfere para o bit de mais alta ordem do acumulador. O bit de menor peso do acumulador é carregado no bit de carry. Não existem operandos na instrução RAR.

Formato

0	**0**	**0**	**1**	**1**	**1**	**1**	**1**

Exemplo

Neste caso o acumulador conterá o valor 6AH. A instrução RAL efetuará as seguintes modificações no registrador acumulador e no bit de carry:



A8. Instruções com pares de registradores

Na continuação são descritas as instruções que dão lugar a operações com pares de registradores.

A8.1. PUSH - Colocar dados na stack

Instrução	PUSH pr
Bits afetados	
Endereçamento	Registrador indireto

Descrição

O conteúdo do par de registradores especificado é guardado nos dois bytes de memória definidos pelo ponteiro da stack.

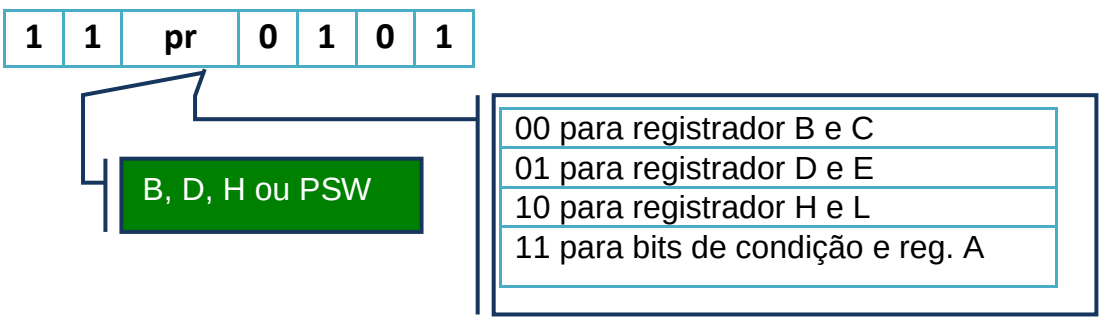
O conteúdo do primeiro registrador é guardado na posição de memória imediatamente inferior a do ponteiro da stack. O conteúdo do segundo registrador do par é guardado na posição de memória duas unidades inferiores ao ponteiro da stack.

Se faz referencia para o par de registradores PSW, no primeiro byte de informação se guarda o estado dos cinco bits de condição. O formato deste byte é o seguinte:

sinal	**zero**	**0**	**arraste auxiliar**	**0**	**paridade**	**1**	**carry**

Seja qual seja o par de registradores especificado, uma vez que os dados foram guardados, no ponteiro de pilha é decrementado em duas unidades.

Formato



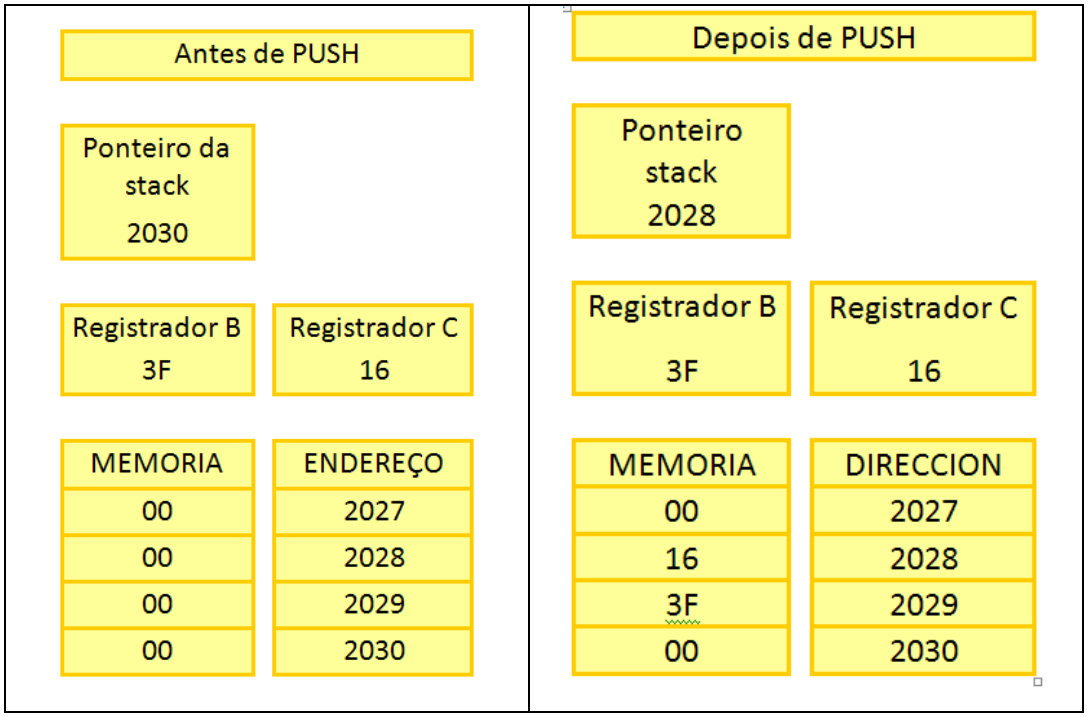
Exemplos

1. Suponhamos que o registrador B contém 3FH, o registrador C contém 16H e o ponteiro da pilha vale 2030H. A instrução:

PUSH B

Armazenará o conteúdo do registrador B na posição de memória 2029H, o conteúdo do registrador C no endereço de memória 2028H, e decrementa duas unidades no ponteiro da stack, deixando-o em 2028H.

Graficamente podemos ver o processo anterior:



2. Suponhamos agora que o acumulador contém 33H, o ponteiro de pilha tem 102AH e os bits de condição de zero, carry e paridade estão em um, mesmo que os de sinal e carry auxiliar estão em zero. A instrução:

PUSH PSW

Armazena o conteúdo do acumulador na posição de memória 1028H e coloca o valor 47H, correspondente aos citados estados dos bits de condição na posição 1029H, mesmo que no ponteiro da pilha fique o valor 1028H.

A8.2. POP - Retirar dados da stack

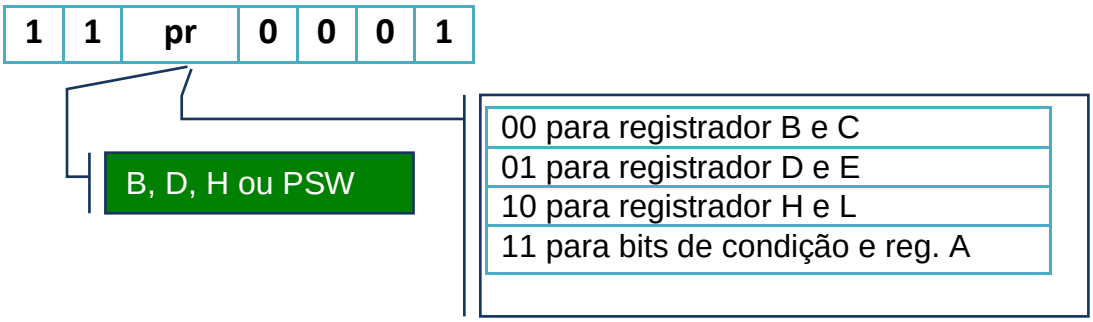
Instrução	POP pr
Bits afetados	
Endereçamento	Registrador indireto

Descrição

POP PR copia o conteúdo da posição de memória endereçada pelo stack pointer no registrador de baixa ordem do par de registradores especificados. Na continuação se incrementa o stack pointer em 1 e copia o conteúdo doa endereço resultante no registrador de mais alta ordem do par. Logo se incrementa o stack pointer outra vez de modo que se aponta para o seguinte dado do stack. O operando deve especificar o par de registradores BC, DE, HL ou PSW.

POP PSW usa o conteúdo da localização de memória especificada pelo stack pointer para restabelecer os bits de condições.

Formato



Exemplos

1. Suponhamos que as posições de memória 2028H e 2029H contenham respectivamente 16H e 3FH, mesmo que o ponteiro de pilha contenha 2028H. A instrução:

POP B

Carrega o registrador C com o valor de 16H da posição de memória 2028H, carrega o registrador B com o valor 3FH da posição 2029H, e incrementa duas unidades no ponteiro da stack, deixando-o em 2030H. Graficamente podemos ver este processo:

Antes de POP

Ponteiro stack 2028

Registrador B 00	Registrador C 00

MEMORIA	ENDEREÇO
00	2027
16	2028
3F	2029
00	2030

Depois de POP

Ponteiro stack 2030

Registrador B 3F	Registrador C 16

MEMORIA	ENDEREÇO
00	2027
16	2028
3F	2029
00	2030

2. SE as posições de memória 1008H E 1009H possuem respectivamente 00H e 16H, e o ponteiro da pilha vale 1008H, a instrução:

POP PSW

Carrega 00H no acumulador e coloca os bits de estado da seguinte forma:

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

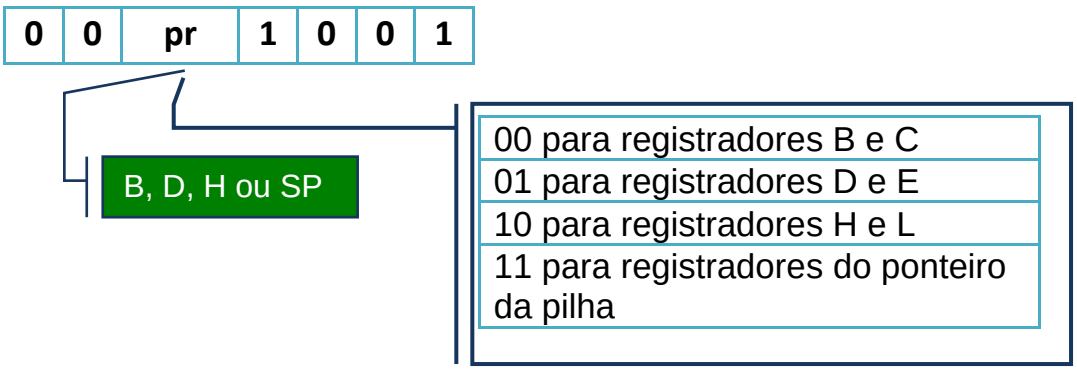
A8.3. DAD - Soma dupla

Instrução	DAD pr
Bits afetados	CY
Endereçamento	Registrador

Descrição

DAD RP soma o valor de um dado de 16 bits contido em um determinado par de registradores (PR) ao conteúdo do par de registradores HL. O resultado é armazenado no par HL. Os operandos (PR) podem ser B = BC, D = DE, H = HL, SP. Deve-se levar em conta que a letra H deve ser empregada para especificar que o par de registradores HL deve ser dobrado. DAD põe o bit de carry em 1 se houver uma saída de carry dos registradores HL. E além disso não afeta a nenhum outro bit.

Formato



Exemplos

1. Supondo que os registradores D, E, H e L contenham 33H, 9FH, A1H e 7BH respectivamente, a instrução:

DAD D

Realiza a seguinte soma:

B – C	339F
H – L	A17B
H – L	051A

2. Ao executar a instrução:

DAD H

É duplicado o valor do número de 16 bits contido em H – L (equivale a deslocar os 16 bits uma posição para a esquerda).

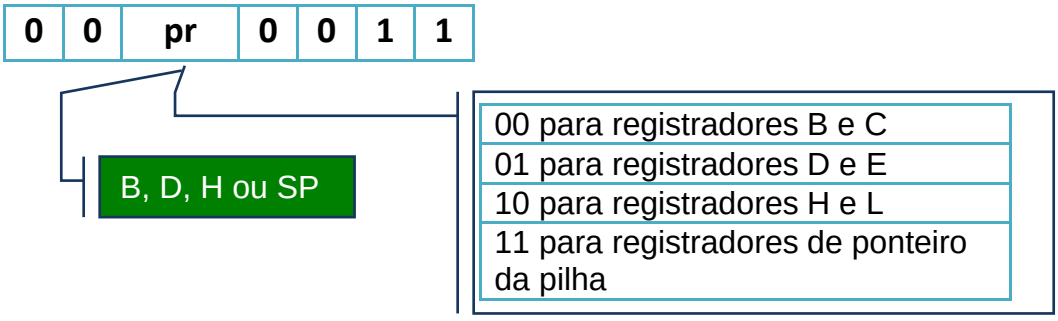
A8.4. INX - Incrementar par de registradores

Instrução	INX pr
Bits afetados	
Endereçamento	Registrador

Descrição

O número de 16 bits contido no par de registradores especificado é incrementado em uma unidade.

Formato



Exemplos

1. Supondo que os registradores H e L contenham respectivamente 30H e 00H, a instrução:

   INX H

   Faz com que o registrador H contenha 30H e o registrador L o valor 01H.

2. Se o ponteiro da pilha contém FFFFH, a instrução:

   INX SP

   Faz com que este contenha 0000H.

A8.5. DCX - Decrementar par de registradores

Instrução	DCR pr
Bits afetados	
Endereçamento	Registrador

Descrição

O número de 16 bits contido no par de registradores especificado é decrementado em uma unidade.

Formato

0	0	pr	1	0	1	1

B, D, H ou SP

00 para registradores B e C
01 para registradores D e E
10 para registradores H e L
11 para registradores de ponteiro da pilha

Exemplo

Supondo que os registradores H e L contenham respectivamente 30H e 00H, a instrução:

DCX H

Faz com que o registrador H contenha 2FH e o registrador L o valor FFH.

A8.6. XCHG - Intercambiar dados entre registradores

Instrução	XCHG
Bits afetados	
Endereçamento	Registrador

Descrição

XCHG troca o conteúdo dos registradores H e L com o conteúdo dos registradores D e E.

Formato

1	**1**	**1**	**0**	**1**	**0**	**1**	**1**

Exemplo

Se os registradores H, L, D e E contém respectivamente 01H, 02H, 03H e 04H, a instrução XCHG realiza o seguinte intercambio:

Antes de executar XCHG			
D	E	H	L
03	04	01	02

Depois de executar XCHG			
D	E	H	L
01	02	03	04

A 8.7. XTHL - Intercambiar dados com a stack

Instrução	XTHL
Bits afetados	
Endereçamento	Registrador indireto

Descrição

XTHL troca os dos bytes da posição mais alta da stack com os dois bytes armazenados nos registradores H e L. Assim XTHL salva o conteúdo atual do par HL e carrega novos valores em HL.

XTHL troca o conteúdo de L com a posição de memória especificada pelo stack pointer e o registrador H é intercambiado com o conteúdo de SP+1.

Formato

1	**1**	**1**	**0**	**0**	**0**	**1**	**1**

Exemplo

Se o ponteiro da pilha contém 40B4H, os registradores H e L contém AAH e BBH respectivamente, e as posições de memória 40B4H e 40B5H contém CCH e DDH respectivamente, e a instrução:

XTHL

Realizará a seguinte operação:

Antes de XTHL

Ponteiro stack 40B4

Registrador H AA	Registrador L BB

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

Depois de XTHL

Ponteiro stack 40B4

Registrador H DD	Registrador L CC

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

A8.8. SPHL - Carregar o ponteiro da stack desde os registradores H e L

Instrução	SPHL
Bits afetados	
Endereçamento	

Descrição

Os 16 bits contidos nos registradores H e L substituem o conteúdo do ponteiro da stack. O conteúdo dos registradores H e L permanece invariável.

Formato

1	1	1	1	1	0	0	1

Exemplo

Se os registradores H e L contém respectivamente 50H e 6CH, a instrução:

SPHL

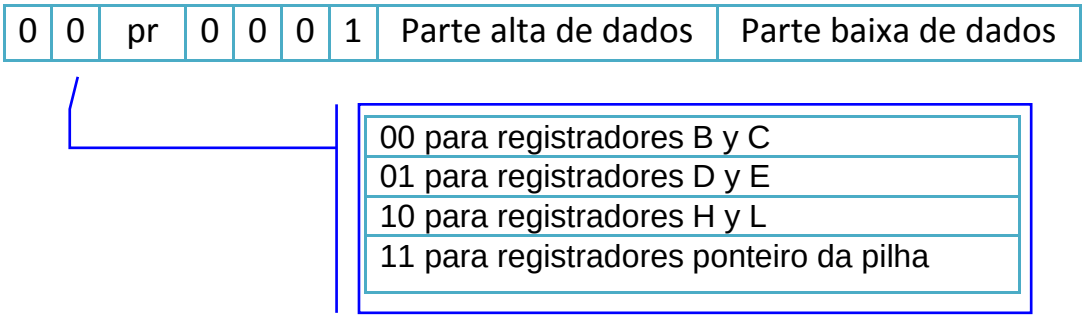
Carrega o ponteiro da stack com o valor 506CH.

A9. Instruções com dados imediatos

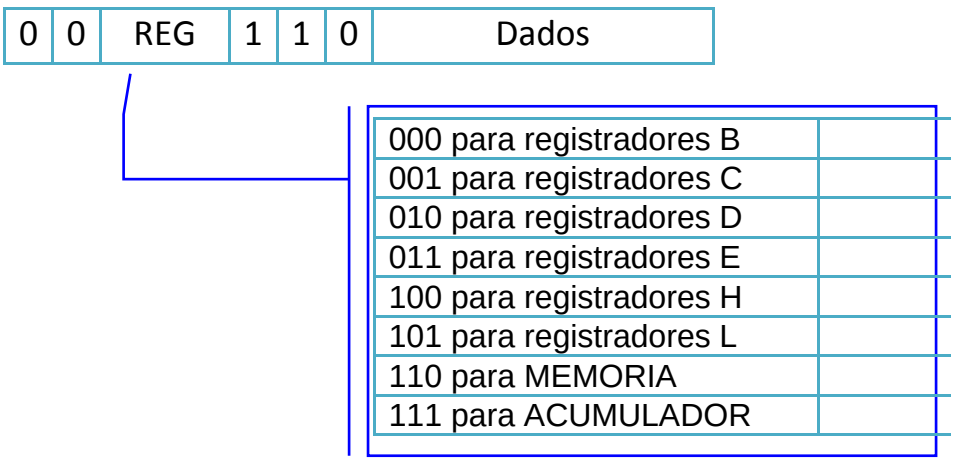
Na continuação serão descritas as instruções que realizam operações utilizando bytes de dados que formam parte da própria instrução.

Estas instruções formam um grupo amplo em que nem todas têm o mesmo tamanho e formato. As instruções ocupam dois ou três bytes do seguinte modo:

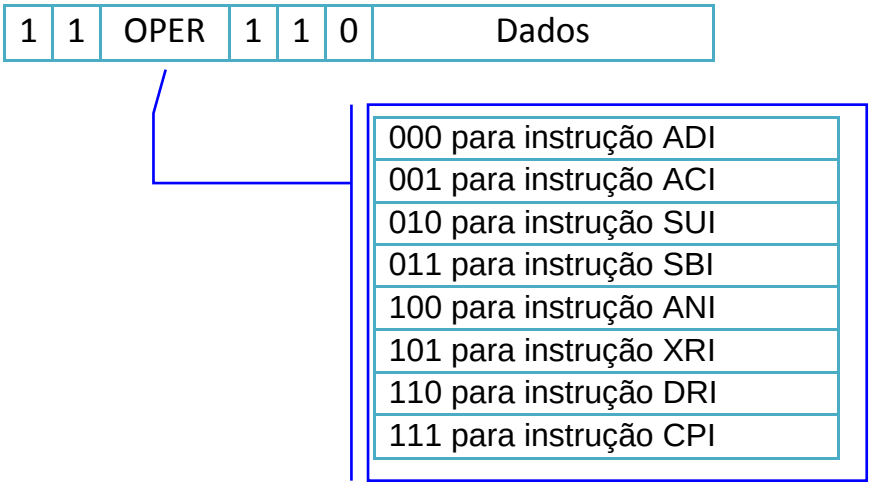
1. A instrução LXI ocupa 3 bytes com o seguinte formato:



2. A instrução MVI ocupa 2 bytes segundo o formato a seguir:



3. Por último, para o restante das instruções, que ocupam 2 bytes, se conta com o formato a seguir:



A instrução LXI opera sobre o par de registradores especificado por PR, usando dois bytes de dados imediatos.

A instrução MVI opera sobre o registrador especificado por REG, usando um byte de dados imediatos. Se fizer referencia a memória, a instrução opera sobre a posição de memória da mesma determinada pelos registradores H e L. O registrador H contém os 8 bits mais significativos doa endereço, mesmo que o registrador L contenha os 8 bits menos significativos.

As instruções restantes operam sobre o acumulador, usando um byte de dados imediatos. O resultado substitui o conteúdo do acumulador.

A9.1. LXI - Carregar um par de registradores com um dado Imediato

Instrução	LXI pr, dados
Bits afetados	
Endereçamento	Imediato

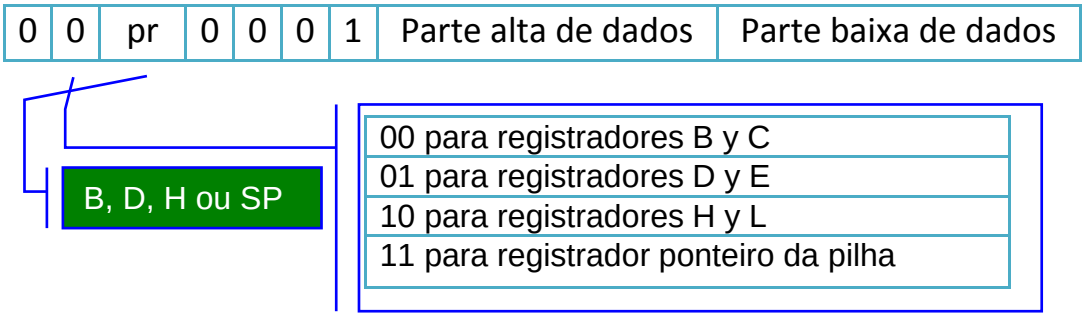
Descrição

LXI é uma instrução de 3 bytes; seu segundo e terceiros byte contém o dado que devera ser carregado no par de registradores (PR). O primeiro operando deve especificar o par de registradores a ser carregado, pode ser os pares BC, DE, HL, ou o

SP. O segundo operando especifica os dois bytes a serem carregados. LXI é a única instrução imediata que aceita um valor de 16 bits. O restante trabalha com dados de 8 bits.

Se o par de registradores especificados for SP, o segundo byte da instrução substitui aos 8 bits menos significativos do ponteiro de pilha, mesmo que o terceiro byte da instrução substitua aos 8 bits mais significativos do ponteiro de pilha.

Formato



Exemplos

1. A instrução:

LXI B, 00FFH

Carrega no registrador B o valor 00H e no registrador C o valor FFH.

2. A instrução seguinte carrega no ponteiro da pilha o valor 1000H:

LXI SP, 1000H

A9.2. MVI - Carregar um registrador com um dado Imediato

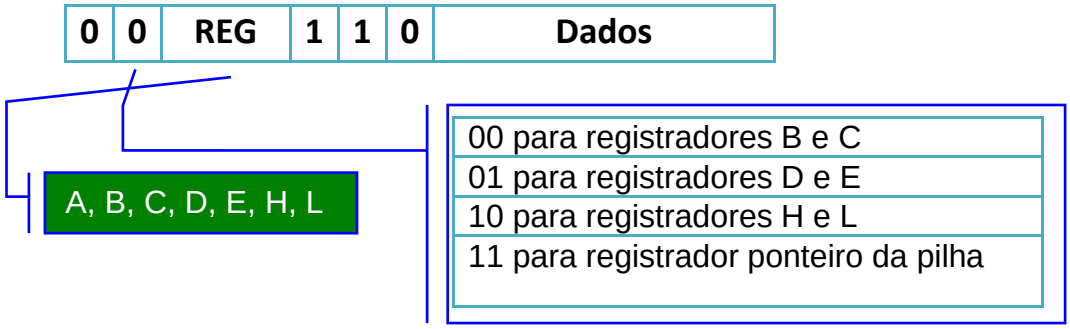
Instrução	MVI reg, dados
Bits afetados	

Endereçamento	Imediato

Descrição

No primeiro operando deve ser um dos registradores A, B, C, D, E, H ou L, que será carregado com o dado especificado no segundo operando (DADOS). O dado não deve exceder de um byte.

Formato



Exemplos

(1). A instrução:

MVI H, 33H

Carrega no registrador H o valor 33H.

(2). A instrução:

MVI L, 44H

Carrega no registrador L o valor 44H.

(3). Na mesma linha dos exemplos anteriores, a instrução:

MVI M, 2AH

Carrega na posição de memória 3344H (endereço aportado pelos registradores H e L) o valor 2AH.

A9.3. ADI - Somar ao acumulador um dado Imediato

Instrução	ADI dados
Bits afetados	Z, S, P, CY, AC
Endereçamento	Imediato

Descrição

Soma o valor do byte especificado na instrução (DADOS), ao conteúdo do acumulador e deixa o resultado no acumulador. O dado deve ser expresso em forma de número, uma constante ASCII, um label de um valor previamente definido ou uma expressão. O dado não deve exceder a um byte.

Utiliza-se aritmética de complemento de dois.

Formato

1	**1**	**0**	**0**	**0**	**1**	**1**	**0**	**Dados**

Exemplo

Na continuação apresentamos um exemplo com 3 instruções:

(1). MVI A, 34

(2). ADI 20

(3). ADI -20

Em todas as instruções são utilizam dados em base decimal. Assim, por exemplo, na instrução (2) o valor 20 é 14H.

A instrução (1) carrega no acumulador o valor 22H.

A instrução (2) realiza a seguinte soma:

Acumulador	22H	0	0	1	0	0	0	1	0
Dado imediato	14H	0	0	0	1	0	1	0	0
Novo acumulador	33H	0	0	1	1	0	1	1	0

O bit de paridade é colocado em um e o resto ficará em zero.

A instrução (3) restaura o valor do acumulador realizando a seguinte soma:

Acumulador	33H	0	0	1	1	0	1	1	0
Dado imediato	ECH	1	1	1	0	1	1	0	0
Novo acumulador	22H	0	0	1	0	0	0	1	0

Agora os bits de paridade, carry e carry auxiliar ficam em um e os demais em zero.

A9.4. ACI - Somar ao acumulador um dado Imediato com arraste

Instrução	ACI dados
Bits afetados	Z, S, P, CY, AC
Endereçamento	Imediato

Descrição

Soma o conteúdo do byte especificado (DADOS) na instrução, ao conteúdo do acumulador, adicionando, além disso, o bit de carry. O resultado é armazenado no acumulador (perdendo-se assim o conteúdo anterior do Acumulador).

O dado (DADOS) deve estar especificado em forma de número, em constante ASCII, como label de um valor previamente definido ou uma expressão. O dado não deve exceder a um byte.

Formato

1	1	0	0	1	1	1	0	Dados

Exemplo

Teremos as seguintes instruções:

(1). MVI A, 34

(2). ACI 20

e vamos supor que o bit de carry é colocado em um.

A instrução (1) carrega no acumulador o valor 22H.

A instrução (2) realiza a seguinte soma:

Acumulador	22H	0	0	1	0	0	0	1	0
Dado imediato	14H	0	0	0	1	0	1	0	0
Bit de carry									1
Novo acumulador	37H	0	0	1	1	0	1	1	1

Todos os bits são colocados em zero.

A9.5. SUI - Subtrair do acumulador um dado Imediato

Instrução	SUI dados

Bits afetados	Z, S, P, CY, AC
Endereçamento	Imediato

Descrição

O byte de dados imediato é subtraído do conteúdo do acumulador usando aritmética de complemento de dois. O resultado é deixado no acumulador.
Já que se trata de uma operação de subtração, o bit de carry é colocado em um quando não ocorrer carry do bit de maior peso, e colocado em zero se ocorrer carry.

Formato

1	**1**	**0**	**1**	**0**	**1**	**1**	**0**	**Dados**

Exemplo

Na continuação apresentamos um exemplo com 2 instruções:

(1). MVI A, B3H

(2). SUI B3H

A instrução (1) carrega no acumulador o valor B3H.

A instrução (2) realiza a seguinte soma (usando complemento de dois do dado imediato):

Acumulador	B3H	1 0 1 1 0 0 1 1
Dado imediato	6DH	0 1 0 0 1 1 0 1
Novo acumulador	00H	0 0 0 0 0 0 0 0

Como era de se esperar o resultado final do acumulador é zero já que lhe estamos subtraindo seu próprio valor. O valor 6DH do dado imediato corresponde ao complemento de dois do valor B3H que estamos subtraindo.
Devido ao fato que existe transbordamento (overflow) do sétimo bit é produzido carry e assim o bit de carry é colocado em zero.

O bit de paridade é colocado em um mesmo que os demais permaneçam inativos.

A9.6. SBI - Subtrair do acumulador um dado Imediato com arraste

Instrução	SBI dados
Bits afetados	Z, S, P, CY, AC
Endereçamento	Imediato

Descrição

O bit de carry é somado internamente ao byte de dados imediato. O valor obtido é subtraído do conteúdo do acumulador usando aritmética de complemento de dois. O resultado é deixado no acumulador.
Esta instrução é similar a SBB, e se usa preferentemente para realizar subtrações multi-byte.

Similar a seção anterior, o bit de carry é colocado em um se não ocorrer carry do bit de maior peso, colocando-se em zero se não ocorrer.

Formato

1	**1**	**0**	**1**	**1**	**1**	**1**	**0**	**Dados**

Exemplo

Temos as seguintes instruções:

(1). MVI A, 00H

(2). SBI 01H

E supomos que o bit de carry é colocado em a zero.

A instrução (1) carrega no acumulador o valor 00H.

A instrução (2) realiza a seguinte soma (usando complemento de dois do dado imediato):

Acumulador	00H	0	0	0	0	0	0	0	0
Dado imediato	FFH	1	1	1	1	1	1	1	1
Bit de carry									0
Novo acumulador	- 1H	1	1	1	1	1	1	1	1

Não há carry, pelo que o bit de carry é colocado em um. Os bits de zero e carry auxiliar estão em zero, mesmo que os de sinal e paridade sejam colocados em um.

A9.7. ANI - Função lógica AND entre o acumulador e um Dado Imediato

Instrução	ANI dados
Bits afetados	Z, S, P, CY, AC
Endereçamento	Imediato

Descrição

Realiza uma operação E (AND) lógica entre o dado (DADOS) especificado na instrução e o conteúdo do acumulador, o resultado fica no acumulador. É colocado em zero o bit de carry. O dado, que não deve exceder a um byte, pode ser expresso na forma de número, uma constante ASCII, um label de algum valor previamente definido ou uma expressão.

Formato

1	1	1	0	0	1	1	0	Dados

Exemplo

Dispomos das seguintes instruções:

(1). MVI A, A0H

(2). ANI 0FH

A instrução (1) carrega no acumulador o valor A0H.

A instrução (2) realiza a seguinte operação AND bit a bit entre o acumulador e o dado imediato 0FH:

Acumulador	A0H	1	0	1	0	0	0	0	0
Dado imediato	OFH	0	0	0	0	1	1	1	1
Novo acumulador	00H	0	0	0	0	0	0	0	0

A instrução ANI de exemplo põe em zero os quatro bits de maior peso, deixando invariáveis os quatro menores. Já que os quatro bits de menor peso do acumulador eram zero, o resultado final é 00H com o que o bit de zero será colocado em zero.

A9.8. XRI - Função lógica OU-EXCLUSIVO entre o Acumulador e um dado Imediato

Instrução	XRI dados
Bits afetados	Z, S, P, CY, AC
Endereçamento	Imediato

Descrição

É realizada a função lógica OU-EXCLUSIVO bit a bit entre um byte de dados imediatos e o conteúdo do acumulador. O resultado é guardado no acumulador. O bit de carry é colocado em zero.

Formato

1	**1**	**1**	**0**	**1**	**1**	**1**	**0**	**Dados**

Exemplo

Esta instrução pode ser utilizada para complementar bits específicos do acumulador deixando os restantes em seu estado original. Deste modo e supondo que o acumulador contenha ABH, a instrução:

XRI 80H

Complementa o bit de maior peso do acumulador, tal e como se mostra na figura a seguir:

Acumulador	ABH	1	0	1	0	1	0	1	1
Dado imediato	80H	1	0	0	0	0	0	0	0
Novo acumulador	2BH	0	0	1	0	1	0	1	1

A9.9. ORI - Função lógica OR entre o acumulador e um dado Imediato

Instrução	ORI dados
Bits afetados	Z, S, P, CY, AC
Endereçamento	Imediato

Descrição

ORI desenvolve uma operação lógica OR entre o conteúdo especificado por DADOS e o conteúdo do acumulador. O resultado será colocado no acumulador. Os bits de carry e

carry auxiliar são colocados em zero.

Formato

1	**1**	**1**	**1**	**0**	**1**	**1**	**0**	**Dados**

Exemplo

Se o acumulador inicialmente contiver 3CH, a instrução:

ORI F0H

Realiza a seguinte operação OR bit a bit:

Acumulador	3CH	0	0	1	1	1	1	0	0
Dado imediato	F0H	1	1	1	1	0	0	0	0
Novo acumulador	FCH	1	1	1	1	1	1	0	0

Como vemos a instrução ORI de nosso exemplo ativa os quatro bits de menor peso, não alterando os demais.

A9.10. CPI - Comparar o conteúdo do acumulador com um dado Imediato

Instrução	CPI dados
Bits afetados	Z, S, P, CY, AC
Endereçamento	Registrador indireto

Descrição

Compara o valor do byte especificado (DADOS) com o conteúdo do acumulador e

posiciona os bits de zero e carry para indicar o resultado. O bit de zero indica igualdade. Um “0” no carry indica que o conteúdo do acumulador é maior que DADOS. Um “1” no carry indica que o acumulador é menor que DADOS. Sem dúvida, o significado do bit de carry é contrário quando os valores têm sinal diferente ou quando um dos valores está complementado. O valor de DADOS não deve exceder a um byte.

Formato

1	**1**	**1**	**1**	**1**	**1**	**1**	**0**	**Dados**

Exemplo

Se tivermos a seqüência de instruções

(1). MVI A, 25H

(2). CPI 20H

A instrução (1) carrega no acumulador o valor 25H.

A instrução (2) realiza a seguinte operação de soma (tomando o complemento de dois do dado imediato, ou seja, E1H):

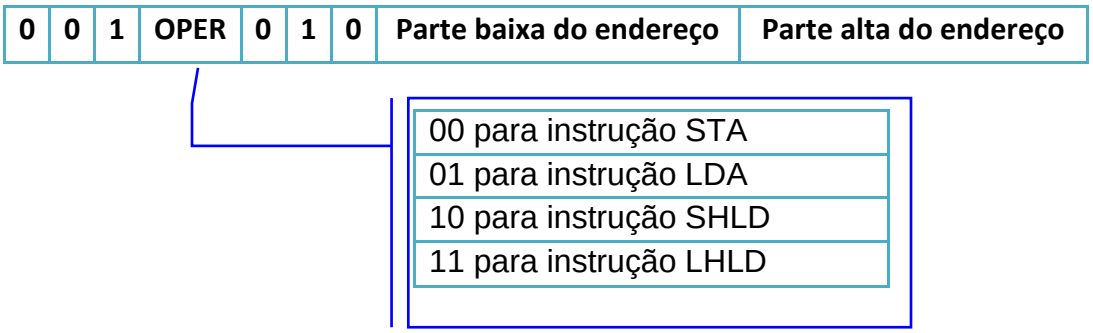
Acumulador	25H	0	0	1	0	0	1	0	1
Dado imediato	E1H	1	1	1	0	0	0	0	1
Novo acumulador	03H	0	0	0	0	0	1	1	0

Existe transbordamento (overflow) do último bit, pelo que o bit de carry é colocado em zero.

O acumulador continua com seu valor inicial, porém o bit de zero está a zero, indicando que as quantidades não são iguais. Ao estar o bit de carry em zero, nos indica que os dados imediatos são menores que o conteúdo do acumulador.

A10. Instruções de Endereçamento direto

Na continuação serão descritas as instruções que fazem referencia a uma posição de memória específica, cujos dois bytes de endereço formam parte da própria instrução. As instruções deste tipo ocupam três bytes da seguinte forma:



A10.1. STA - Armazenamento direto desde o Acumulador

Instrução	STA dir
Bits afetados	
Endereçamento	Direto

Descrição

STA DIR armazena uma cópia do conteúdo atual do acumulador na posição de memória especificada por DIR.

Formato

0	0	1	1	0	0	1	0	Parte baixa do	Parte alta do

								endereço	endereço

Exemplo

Todas as instruções que são mostradas na continuação carregam o conteúdo do acumulador na posição de memória 0080H:

- **STA 0080H** **// Base hexadecimal**
- **STA 128** **// Base decimal**
- **STA 0000000010000000B** **// Base binária**

A10.2. LDA - Carga direta no acumulador

Instrução	LDA dir
Bits afetados	
Endereçamento	Direto

Descrição

LDA DIR carrega o acumulador com o conteúdo da memória endereçada por DIR. O endereço pode ser posto como um número, um label previamente definido ou uma expressão.

Formato

0	**0**	**1**	**1**	**1**	**0**	**1**	**0**	**Parte baixa do endereço**	**Parte alta do endereço**

Exemplo

Todas as instruções que são mostradas a seguir carregam o acumulador com o conteúdo da posição de memória 300H:

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

A10.3. SHLD - Carregar diretamente com H e L

Instrução	SHLD dir
Bits afetados	
Endereçamento	Direto

Descrição

Armazena uma cópia do registrador L na posição de memória especificada por DIR, na continuação armazena uma cópia do registrador H na posição seguinte de memória (DIR+1).

Formato

0	**0**	**1**	**0**	**0**	**0**	**1**	**0**	**Parte baixa do endereço**	**Parte alta do endereço**

Exemplo

Supondo que os registradores H e L contém respectivamente os valores 3CH e 54H, a instrução

SHLD 34B3

Efetuará as seguintes modificações na memória:

Memória antes de executar SHLD		Memória depois de executar SHLD

		ENDEREÇO		
FF	←	34B2	→	FF
FF	←	34B3	→	54
FF	←	34B4	→	3C
FF	←	34B5	→	FF

A10.4. LHLD - Carregar H e L diretamente

Instrução	LHLD dir
Bits afetados	
Endereçamento	Direto

Descrição

LHLD DIR carrega o registrador L com uma cópia do byte armazenado na posição de memória especificada por DIR. Depois carrega o registrador H com uma cópia do byte armazenado na posição seguinte de memória especificada por DIR. A instrução LHLD esta prevista para carregar endereços novos nos registradores H e L.

Formato

0	**0**	**1**	**0**	**1**	**0**	**1**	**0**	**Parte baixa do endereço**	**Parte alta do endereço**

Exemplo

No caso em que as posições de memória AB24H e AB25H contenham respectivamente 22H e 33H, a execução da instrução:

LHLD AB24H

Fará com que o registrador L contenha 22H e o registrador H contenha 33H.

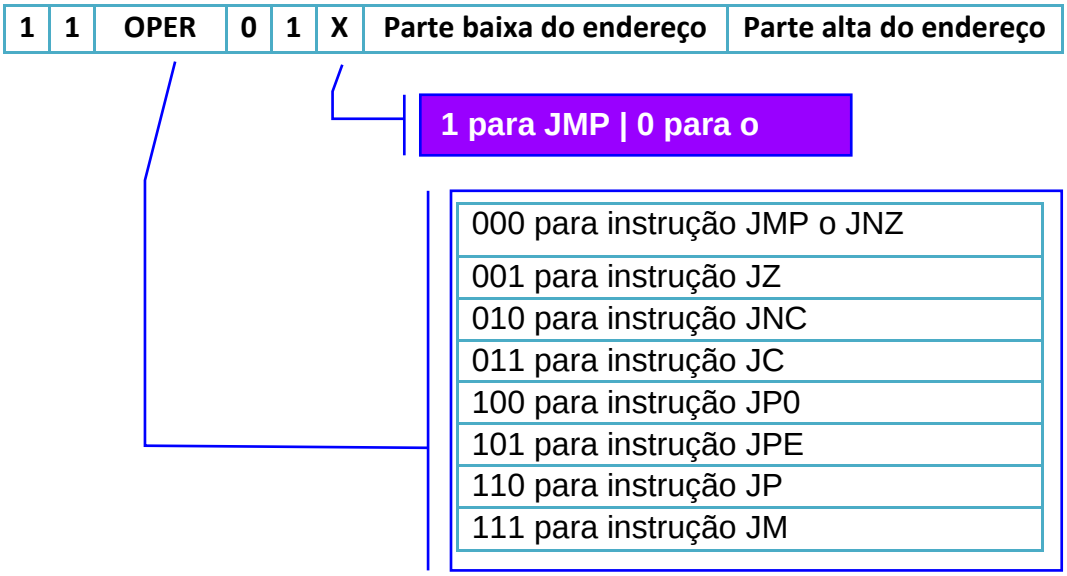
A11. Instruções de salto

Na continuação serão descritas as instruções que modificam a seqüência normal de execução das instruções de um programa. Estas instruções ocupam um ou três bytes da forma seguinte:

1. A instrução PCHL ocupa um byte com o seguinte formato:

1	1	1	0	1	0	0	1

2. O resto das instruções de salto ocupam três bytes com o seguinte formato:



Certas instruções de três bytes deste tipo provocam uma mudança na seqüência normal de operações na execução de um programa segundo algumas determinadas condições.

Se a condição específica for verdadeira, a execução do programa continua no endereço de memória formado pela parte alta (terceiro byte da instrução), e os oito bits da parte baixa (segundo byte da instrução).

Se a condição específica for falsa, a execução do programa continua na próxima instrução em seqüência.

A11.1. PCHL - Carregar o contador de programa

Instrução	PCHL
Bits afetados	
Endereçamento	Registrador

Descrição

PCHL carrega o conteúdo dos registradores H e L no contador de programa. Como o processador busca a instrução seguinte no próximo endereço do contador de programa, PCHL tem o efeito de uma instrução de salto. O conteúdo de H vai para os 8 bits mais altos do contador de programa e o conteúdo de L vai para os 8 bits mais baixos.

Formato

1	**1**	**1**	**0**	**1**	**0**	**0**	**1**

Exemplo

Se o registrador H contiver A5H e o registrador L o valor 9DH, a instrução:

PCHL

Faz com que o programa em curso continue sendo executado no endereço de memória A59DH.

A11.2. JMP - Salto incondicional

Instrução	JMP dir
Bits afetados	
Endereçamento	Imediato

Descrição

A instrução JMP DIR altera a execução do programa carregando o valor especificado por DIR no contador de programa.

Formato

1	**1**	**0**	**0**	**0**	**0**	**1**	**1**	**Parte baixa do endereço**	**Parte alta do endereço**

A11.3. JC - Saltar se houver carry

Instrução	JC dir
Bits afetados	
Endereçamento	Imediato

Descrição

A instrução JC DIR testa o valor do bit de carry. Se for “1” a execução do programa continua no endereço especificado por DIR. Se for “0” o programa continua sua execução normal de forma seqüencial.

Formato

1	**1**	**0**	**1**	**1**	**0**	**1**	**0**	**Parte baixa do endereço**	**Parte alta do endereço**

A11.4. JNC - Saltar se não ocorrer carry

Instrução	JNC dir
Bits afetados	
Endereçamento	Imediato

Descrição

A instrução JNC DIR testa o estado do bit carry. Se estiver em “0” o programa muda o endereço especificado por DIR. Se estiver em “1” a execução do programa continua normalmente.

Formato

1	**1**	**0**	**1**	**0**	**0**	**1**	**0**	**Parte baixa do endereço**	**Parte alta do endereço**

A11.5. JZ - Saltar se for zero

Instrução	JZ dir
Bits afetados	
Endereçamento	Imediato

Descrição

A instrução JZ DIR testa o bit de zero. Se estiver em “1” o programa continua no endereço expresso por DIR. Si for “0” continua com a execução seqüencial normal.

Formato

1	**1**	**0**	**0**	**1**	**0**	**1**	**0**	**Parte baixa do endereço**	**Parte alta do endereço**

A11.6. JNZ - Saltar se não for zero

Instrução	JNZ dir
Bits afetados	
Endereçamento	Imediato

Descrição

A instrução JNZ DIR testa o valor do bit de zero. Se o conteúdo do acumulador não for zero (Bit de zero = 0) o programa continua no endereço especificado por DIR. Se o conteúdo do acumulador for zero (Bit de zero = 1) o programa continua seu ciclo normal.

Formato

1	**1**	**0**	**0**	**0**	**0**	**1**	**0**	**Parte baixa do endereço**	**Parte alta do endereço**

A11.7. JM - Saltar se ocorrer sinal negativo

Instrução	JM dir
Bits afetados	
Endereçamento	Imediato

Descrição

A instrução JM DIR testa o estado do bit de sinal. Se o conteúdo do acumulador for negativo (bit de sinal = 1) a execução do programa continua no endereço especificado por DIR. Se o conteúdo do acumulador for positivo (bit de sinal = 0) continua a execução da seqüência normal.

Formato

1	**1**	**1**	**1**	**1**	**0**	**1**	**0**	**Parte baixa do endereço**	**Parte alta do endereço**

A11.8. JP - Saltar se ocorrer sinal positivo

Instrução	JP dir
Bits afetados	
Endereçamento	Imediato

Descrição

A instrução JP DIR testa o estado do bit de sinal. Se o conteúdo do acumulador for positivo (bit de sinal = 0) a execução do programa continua com o endereço especificado por DIR. Se o conteúdo do acumulador for negativo (bit de sinal = 1) continua o programa com sua execução normal.

Formato

1	**1**	**1**	**1**	**0**	**0**	**1**	**0**	**Parte baixa do endereço**	**Parte alta do endereço**

A11.9. JPE - Saltar se a paridade for par

Instrução	JPE dir
Bits afetados	
Endereçamento	Imediato

Descrição

A paridade existe se o byte que está no acumulador tem um número par de bits. O bit de paridade é colocado em 1 para indicar esta condição.

A instrução JPE DIR testa a situação de bit de paridade. Se estiver em “1”, a execução do programa continua no endereço especificado por DIR. Se estiver em “0”, continua com a instrução seguinte de forma seqüencial.

As instruções JPE e JPO são especialmente usadas para comprovar a paridade dos dados de entrada. (Sem dúvida com a instrução IN os bits não atuam. Isto pode ser evitada somando 00H ao acumulador para ativá-los).

Formato

1	**1**	**1**	**0**	**1**	**0**	**1**	**0**	**Parte baixa do endereço**	**Parte alta do endereço**

A11.10. JP0 - Saltar se a paridade for impar

Instrução	JPO dir
Bits afetados	
Endereçamento	Imediato

Descrição

A instrução JPO DIR testa o estado do bit de paridade. Se estiver em “0”, o programa continua no endereço marcado por DIR. Se estiver em “1” continua com a seqüência normal.

Formato

1	**1**	**1**	**0**	**0**	**0**	**1**	**0**	**Parte baixa do endereço**	**Parte alta do endereço**

A12. Instruções de chamada a sub-rotina

Na continuação são descritas as instruções que chamam a sub-rotinas. Estas instruções operam da mesma forma que as instruções de salto, provocando uma alteração na seqüência de execução das instruções, porém, além disso, no momento de sua execução, se armazena na pilha um endereço de retorno, que é utilizado pelas instruções de RETORNO (ver instruções de Retorno de sub-rotinas neste mesmo capítulo).

As instruções deste tipo utilizam três bytes no seguinte formato:

1	**1**	**OPER**	**1**	**0**	**X**	**Parte baixa do endereço**	**Parte alta do endereço**

1 para CALL | 0 para o restante

000 para instrução CNZ ou CALL
001 para instrução CZ
010 para instrução CNC
011 para instrução CC
100 para instrução CP0
101 para instrução CPE
110 para instrução CP
111 para instrução CM

Nas instruções de chamada, as instruções são codificadas como nas instruções de salto, ou seja, armazenando em primeiro lugar o byte de endereço de memória menos significativo.

A12.1. CALL - Chamada incondicional

Instrução	CALL dir
Bits afetados	
Endereçamento	Imediato / Registrador indireto

Descrição

CALL guarda o conteúdo do contador de programa (o endereço da próxima instrução seqüencial) dentro da stack e na continuação salta para o endereço especificada por DIR.

Cada instrução CALL ou alguma de suas variantes implica em um instrução RET (retorno), caso contrário o programa poderia "perder-se" com conseqüências imprevisíveis. O endereço deve ser especificado como um número, um label, ou uma expressão. O label é a forma mais usual (O programa montador inverte o byte alto e baixo do endereço durante o processo de montagem). As instruções CALL são empregadas para chamadas de sub-rotinas e devemos ter presente que sempre utilizam a stack.

Formato

1	**1**	**0**	**0**	**0**	**1**	**0**	**1**	**Parte baixa do endereço**	**Parte alta do endereço**

A12.2. CC - Chamar se ocorrer carry

Instrução	CC dir
Bits afetados	
Endereçamento	Imediato / Registrador indireto

Descrição

CC testa o estado do bit de carry. Se o bit estiver em “1”, CC carrega o conteúdo do contador de programa na stack e na continuação salta para o endereço especificado por DIR. Se o bit estiver em “0”, a execução do programa continua com a próxima instrução de sua seqüência normal. Ainda que o uso de um label seja o usual, o endereço pode ser especificado também como um número ou uma expressão.

Formato

1	**1**	**0**	**1**	**1**	**1**	**0**	**0**	**Parte baixa do endereço**	**Parte alta do endereço**

A12.3. CNC - Chamar se não ocorrer carry

Instrução	CNC dir
Bits afetados	
Endereçamento	Imediato / Registrador indireto

Descrição

CNC verifica o valor do bit de carry. Se estiver em “zero” CNC carrega o conteúdo do contador de programa na stack e na continuação salta para o endereço especificado pela instrução em DIR. Se o bit estiver em “1”, o programa continua com sua seqüência normal. Mesmo que o uso de um label seja o mais comum, o endereço pode também estar indicado por um número ou por uma expressão.

Formato

1	1	0	1	0	1	0	0	Parte baixa do endereço	Parte alta do endereço

A12.4. CZ - Chamar se ocorrer zero

Instrução	CZ dir
Bits afetados	
Endereçamento	Imediato / Registrador indireto

Descrição

CZ verifica o bit de zero. Se o bit estiver em "1" (indicando que o conteúdo do acumulador é "zero"), CZ carrega o conteúdo do contador de programa na stack e salta para o endereço especificado em DIR. Se o bit estiver em "0" (indicando que o conteúdo do acumulador é diferente de zero) o programa continua seu desenvolvimento normal.

Formato

1	1	0	0	1	1	0	0	Parte baixa do endereço	Parte alta do endereço

A12.5. CNZ - Chamar se não for zero

Instrução	CNZ dir
Bits afetados	
Endereçamento	Imediato / Registrador indireto

Descrição

CNZ verifica o bit de Zero. Se estiver em "0" (indicando que o conteúdo do acumulador não é zero), CNZ manda o conteúdo do contador de programa para a stack e salta para o endereço especificado por DIR. Se o bit estiver em "1" o programa continua seu desenvolvimento normal.

Formato

1	**1**	**0**	**0**	**0**	**1**	**0**	**0**	**Parte baixa do endereço**	**Parte alta do endereço**

A12.6. CM - Chamar se ocorrer sinal negativo

Instrução	CM dir
Bits afetados	
Endereçamento	Imediato / Registrador indireto

Descrição

CM testa o estado do bit de sinal. Se o bit estiver em "1" (indicando que o conteúdo do acumulador é negativo) CM manda o conteúdo do contador de programa para a stack e salta para o endereço especificado por DIR. Se o bit for "0" a execução do programa continua com sua seqüência normal. O uso de label é mais corrente, porém o endereço pode ser especificado também por um número ou uma expressão.

Formato

1	**1**	**1**	**1**	**1**	**1**	**0**	**0**	**Parte baixa do endereço**	**Parte alta do endereço**

A12.7. CP - Chamar se ocorrer sinal positivo

Instrução	CP dir
Bits afetados	
Endereçamento	Imediato / Registrador indireto

Descrição

CP verifica o valor do bit de sinal. Se estiver em “0” (indicando que o conteúdo do acumulador é positivo), CP envia o conteúdo do contador de programa para a stack e salta para o endereço especificado por DIR. Se o bit estiver em “1”, continua o programa normalmente com a instrução seguinte.

Formato

1	**1**	**1**	**1**	**0**	**1**	**0**	**0**	**Parte baixa do endereço**	**Parte alta do endereço**

A12.8. CPE - Chamar se a paridade for par

Instrução	CPE dir
Bits afetados	
Endereçamento	Imediato / Registrador indireto

Descrição

Existe paridade em um byte se o número de “1s” que tem é par. O bit de paridade é colocado em 1 para indicar esta condição. CPE verifica o valor do bit de paridade. Se tiver “1”, CPE envia o conteúdo do contador de programa para a stack e salta para o endereço especificado pela instrução em DIR. Se o bit estiver em “zero”, o programa continua normalmente.

Formato

1	**1**	**1**	**0**	**1**	**1**	**0**	**0**	**Parte baixa do endereço**	**Parte alta do endereço**

A12.9. CPO - Chamar se a paridade for impar

Instrução	CPO dir
Bits afetados	
Endereçamento	Imediato / Registrador indireto

Descrição

CPO verifica o bit de paridade. Se o bit estiver em “0”, CPO carrega o conteúdo do contador de programa na stack e salta para o endereço especificado em DIR. Se o bit estiver em “1” o programa continua seu desenvolvimento normal.

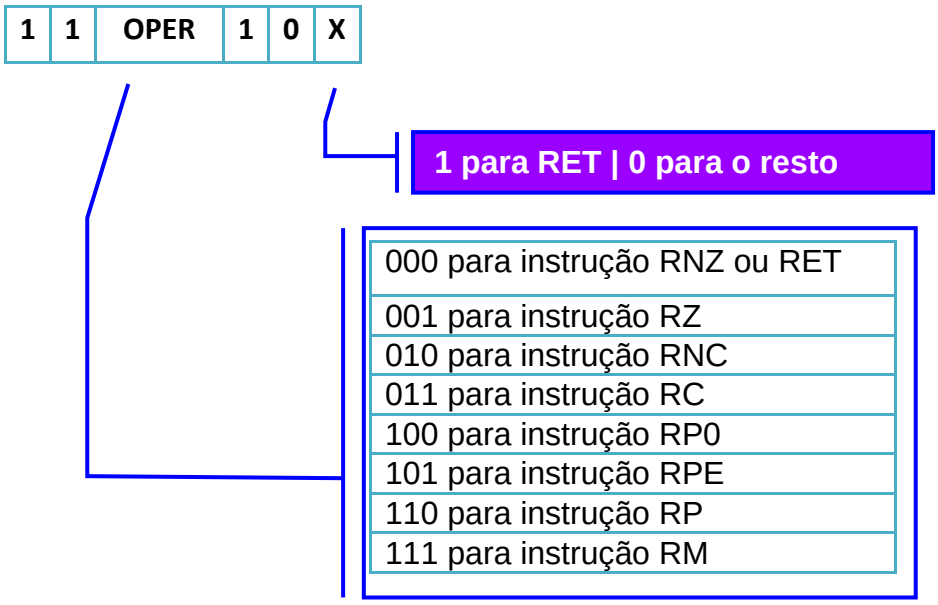
Formato

1	1	1	0	0	1	0	0	Parte baixa do endereço	Parte alta do endereço

A13. Instruções de retorno de sub-rotinas

Na continuação serão descritas as instruções utilizadas para realizar um retorno desde as sub-rotinas. Estas instruções colocam no contador de programa o último endereço posto na pilha, fazendo com que a execução do programa continue neste endereço.

As instruções deste tipo utilizam um byte no seguinte formato:



Certas instruções deste tipo realizam a operação de retorno sob certas condições específicas. Se a condição especificada for cumprida (verdadeira), ocorrerá a operação de retorno. Caso contrario, se a condição não for cumprida, a execução do programa continuará na próxima instrução em seqüência.

A13.1. RET - Retorno incondicional

Instrução	RET
Bits afetados	
Endereçamento	Registrador indireto

Descrição

Realiza-se uma operação de retorno incondicional.

A instrução RET feita fora dos bytes de dados da stack e os coloca no registrador contador de programa. O programa continua então no novo endereço. Normalmente RET é empregado conjuntamente com CALL.

Formato

1	**1**	**0**	**0**	**1**	**0**	**0**	**1**

A13.2. RC - Retorno se ocorrer carry

Instrução	RC
Bits afetados	
Endereçamento	Registrador indireto

Descrição

A instrução RC testa o estado do bit de carry. Se tiver “1” (indicando que ocorreu carry) a instrução retira dois bytes da stack e os coloca no contador de programa. O

programa continua no novo endereço apontado. Se o bit for “0”, o programa continua na seguinte instrução da seqüência normal.

Formato

1	1	0	1	1	0	0	0

A13.3. RNC - Retorno se não ocorrer carry

Instrução	RNC
Bits afetados	
Endereçamento	Registrador indireto

Descrição

A instrução RNC testa o bit de carry. Se estiver em “0” indicando que não ocorreu carry, a instrução verifica fora do stack dois bytes e os carrega no contador de programa. Se o bit estiver em “1” continua o ciclo normal.

Formato

1	1	0	1	0	0	0	0

A13.4. RZ - Retorno se ocorrer zero

Instrução	RZ
Bits afetados	
Endereçamento	Registrador indireto

Descrição

A instrução RZ testa o bit de zero. Se estiver em “1”, indicando que o conteúdo do acumulador é zero, a instrução verifica fora da stack dois bytes e os carrega no

contador de programa. Se o bit estiver em “0”, continua o ciclo normal.

Formato

1	1	0	0	1	0	0	0

A13.5. RNZ - Retorno se não ocorrer zero

Instrução	RNZ
Bits afetados	
Endereçamento	Registrador indireto

Descrição

A instrução RNZ testa o bit zero. Se estiver em “0”, indicando que o conteúdo do acumulador não for zero, a instrução verifica fora da stack dois bytes e os carrega no contador de programa. Se o bit estiver em “1”, continua o ciclo normal.

Formato

1	1	0	0	0	0	0	0

A13.6. RM - Retorno se ocorrer sinal negativo

Instrução	RM
Bits afetados	
Endereçamento	Registrador indireto

Descrição

A instrução RM testa o bit de sinal. Se estiver em “1”, indicando dado negativo no

acumulador, a instrução verifica dois bytes fora da stack e os coloca no contador de programa. Se o bit estiver em “0”, continua o programa normal com a instrução seguinte.

Formato

1	1	1	1	1	0	0	0

A13.7. RP - Retorno se ocorrer sinal positivo

Instrução	RP
Bits afetados	
Endereçamento	Registrador indireto

Descrição

A instrução RP testa o bit sinal. Se estiver em “0”, indicando que o conteúdo do acumulador é positivo, a instrução verifica fora da stack dois bytes r os carrega no contador de programa. Se o bit estiver em “1” continua o ciclo normal.

Formato

1	1	1	1	0	0	0	0

A13.8. RPE - Retorno se a paridade for par

Instrução	RPE
Bits afetados	
Endereçamento	Registrador indireto

Descrição

A instrução RPE testa o bit de paridade. Se estiver em “1”, indicando que existe paridade, a instrução verifica fora da stack dois bytes e os carrega no contador de programa. Se o bit estiver em “0” continua o ciclo normal. (Existirá paridade se o byte que estiver no acumulador tiver um número par de bits, colocando o bit de paridade em “1” neste caso).

Formato

1	**1**	**1**	**0**	**1**	**0**	**0**	**0**

A13.9. POR - Retorno se a paridade for impar

Instrução	RPO
Bits afetados	
Endereçamento	Registrador indireto

Descrição

A instrução RPO testa o bit de paridade. Se estiver em “0”, indicando que não há paridade, a instrução verifica fora da stack dois bytes e os carrega no contador de programa. Se o bit estiver em “1”, continua o ciclo normal.

Formato

1	**1**	**1**	**0**	**0**	**0**	**0**	**0**

A14. Instrução RST

Instrução	RST exp
Bits afetados	
Endereçamento	Registrador indireto

Descrição

É uma instrução CALL para usar com interrupções. RST carrega o conteúdo do contador de programa na stack, para prover um endereço de retorno e salta para uma dos "oito" endereços determinados previamente.

Um código de três bits incluído no código de operação da instrução RST especifica o endereço de salto. Esta instrução é empregada pelos periféricos quando tentam uma interrupção.

Para voltar para a instrução em que tenha ocorrido a chamada de interrupção, se deve utilizar uma instrução de RETORNO.

Formato

1	1	EXP	1	1	1

Onde EXP é um número binário entre 000B e 111B.

Portanto, segundo a combinação de 0 e 1 que dermos a EXP obteremos os diferentes formatos e os distintas endereços das interrupções, que serão:

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

Exemplos

1. A instrução:

   RST 3

Chama a a sub-rotina da posição 0018H.

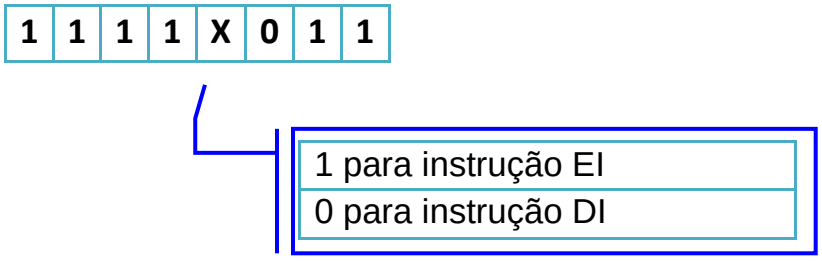
2. A instrução:

RST 10

É uma instrução ilegal já que o argumento de RST deve ser um número entre 0 e 7.

A15. Instruções do FLIP-FLOP de interrupção

Nesta seção estudamos as instruções que operam diretamente sobre o flip-flop de atuação de interrupções. Todas estas instruções ocupam um byte seguindo o formato que se mostra a seguir:



A15.1. EI - Ativar interrupções

Instrução	EI
Bits afetados	
Endereçamento	

Descrição

A instrução EI faz trabalhar o sistema de interrupções a partir da instrução seguinte seqüencial do programa. Esta instrução ativa o flip-flop INTE.

Pode-se desconectar o sistema de interrupções colocando uma instrução DI no início de uma seqüência, posto que não se pode predizer a chegada de uma interrupção.

Ao final da seqüência se inclui a instrução EI que volta a habilitar o sistema de interrupções. (RESET também põe fora de serviço o sistema de interrupções além de colocar o contador de programa em zero).

Formato

1	**1**	**1**	**1**	**1**	**0**	**1**	**1**

A15.2. DI - Desativar interrupções

Instrução	DI
Bits afetados	
Endereçamento	

Descrição

Esta instrução desativa o flip-flop INTE. Depois da execução de uma instrução DI, o sistema de “interrupções” ficará sem a possibilidade de trabalhar. Em aplicações que empreguem as interrupções, a instrução DI é utilizada somente quando uma determinada seqüência não deve ser interrompida. Por exemplo, se pode deixar inoperante o sistema de interrupções incluindo uma instrução DI no início de uma seqüência de códigos.

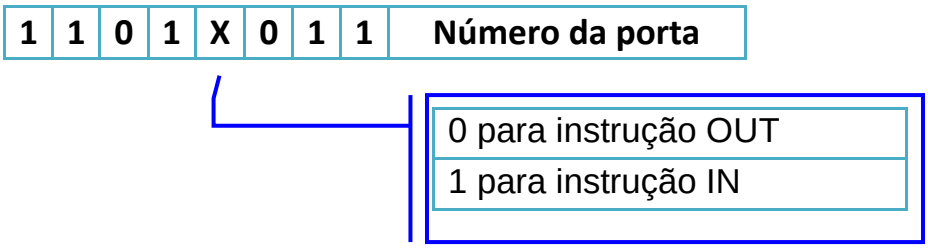
A interrupção TRAP do 8085 não pode ser colocada fora de serviço (desabilitada). Esta interrupção especial é prevista para sérios problemas que podem ocorrer independentemente do bit de interrupção (por exemplo: falha da alimentação, etc.).

Formato

1	**1**	**1**	**1**	**0**	**0**	**1**	**1**

A16. Instruções de Entrada / Saída

Agora nos ocuparemos das instruções que fazem com que os dados entrem ou saiam da CPU. Estas instruções ocupam dois bytes no seguinte formato:



O número da porta é definido pelo hardware do sistema, não estando sob o controle do programador. Este, unicamente seleciona um deles para realizar a operação correspondente.

A16.1. IN - Entrada

Instrução	IN port
Bits afetados	
Endereçamento	Direto

Descrição

A instrução IN PORT le os 8 bits de dados que estão na "PORT" especificada e os carrega no acumulador. O operando deve ser um número ou uma expressão que produza um valor compreendido entre 00H e FFH.

Formato

1	1	0	1	1	0	1	1	Número da porta

Exemplos

1. A instrução:

   IN 2

   Deposita no acumulador os dados de entrada da porta 2.

2. A instrução:

   IN 3*2

   Deposita no acumulador os dados de entrada da porta 6.

A16.2. OUT - Saída

Instrução	OUT port
Bits afetados	
Endereçamento	Direto

Descrição

OUT PORT põe o conteúdo do acumulador no bus de dados de 8 bits da porta selecionada. O número de portas oscila de 0 a 255 e duplicado no bus de endereços. É a lógica externa a encarregada de selecionar a porta e aceitar o dado de saída. O operando (PORT) deve especificar o número da porta de saída selecionada.

Formato

1	1	0	1	0	0	1	1	Número da porta

Exemplos

1. A instrução:

   OUT 2

   Envia o conteúdo do acumulador para a porta de saída número 2.

2. A instrução:

   OUT 3*2

   Envia o conteúdo do acumulador para a porta de saída número 6.

A17. Instrução de parada HLT

Instrução	HLT
Bits afetados	
Endereçamento	

Descrição

A instrução HLT para o processador.

O contador de programa contém o endereço da próxima instrução seqüencial. Por outro lado os bits e registradores permanecem inativos. Uma vez em estado de parada o processador pode voltar a ser arrancado somente por um acontecimento externo, ou seja, uma interrupção. Portanto devemos assegurar-nos que as interrupções estejam a disposição para serem ativadas antes de executar a instrução HLT.

Se for executado HLT estando as interrupções fora de serviço, a única maneira de voltar a arrancar o processador será mediante um RESET ou através da interrupção TRAP.

O processador pode sair temporariamente do estado de parada para proporcionar um acesso direto a memória, sem dúvida terminado o acesso direto volta ao estado de parada.

Um propósito básico da instrução HLT é permitir uma pausa ao processador enquanto espera pea interrupção de um periférico.

Formato

0	**1**	**1**	**1**	**0**	**1**	**1**	**0**

A18. Instrução RIM para a leitura da máscara de Interrupções

Instrução	RIM
Bits afetados	
Endereçamento	

Descrição

RIM carrega os 8 bits de dados seguintes no 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**

Onde:

- SID: Bit presente na entrada serial.
- I7.5: Interrupção 7.5 pendente se estiver em 1.
- I6.5: Interrupção 6.5 pendente se estiver em 1.
- I5.5: Interrupção 5.5 pendente se estiver em 1.
- IE: As interrupções são autorizadas se estiver em 1.
- M7.5: A interrupção 7.5 está proibida se estiver em 1.
- M6.5: A interrupção 6.5 está proibida se estiver em 1.
- M5.5: A interrupção 5.5 está proibida se estiver em 1.

Formato

0	**0**	**1**	**0**	**0**	**0**	**0**	**0**

A19. Instrução SIM para posicionar a máscara de interrupções

Instrução	SIM
Bits afetados	
Endereçamento	

Descrição

SIM é uma instrução de usos múltiplos que utiliza o conteúdo do acumulador para posicionar o mascaramento de interrupções para as RST 5.5, RST 6.5, RST 7.5; põe em zero a borda sensitiva da entrada RST 7.5 e retira o bit 7 do acumulador do latch de dados da saída serial. A estrutura da instrução SIM é como segue:

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**

onde:

- SOD: Bit a apresentar sobre a saída serial.
- SOE: A saída serie está autorizada se estiver em 1.
- R7.5: Reset de RST 7.5. Se o 1 do flip-flop for colocado em 0.
- MSE: Se for “1” os mascaramentos estão autorizados.
- M7.5: A interrupção 7.5 fica proibida se estiver em 1.
- M6.5: A interrupção 6.5 fica proibida se estiver em 1.
- M5.5: A interrupção 5.5 fica proibida se estiver em “1”.

Se o bit 3 for colocado em “1”, a função por “mask” passa a estar permitida.

Os bits de 0 ao 2 colocam em serviço a correspondente interrupção RST colocando um 0 na interrupção que desejamos habilitar. Se colocamos “1” em algum dos bits 0 ao 2, a interrupção correspondente não será cumprida.

Se o bit 3 tiver “0”, os bits 0 ao 2 não terão efeito. Deve-se usar esta peculiaridade para enviar um bit da saída serial sem afetar o mascaramento das interrupções. (A instrução DI anula a SIM).

Se o bit 6 (SOE) é colocado em “1” se habilita a função de saída serial.

O bit 7 é situado na saída SOD onde pode ser tratado pelos dispositivos periféricos. Se o bit 6 for colocado em “zero”, o bit 7 não terá efeito algum, sendo ignorado.

Formato

0	0	1	1	0	0	0	0