

**SIEMENS**

DANIEL Q.

# Conjunto de Instruções

**Microprocessador  
SAB 8080/8085**

Mnemonico	Binário	Bits de Flag influenciados	Número de Bytes-Estados	Instrução em Inglês	Função da Instrução
-----------	---------	----------------------------	-------------------------	---------------------	---------------------

## USO DE SUB-ROTINAS

### a) Instruções de chamada

Para todas instruções de chamada o endereço de retorno é armazenado no STACK

CALL	adr	11001101	- - - - -	3 17	Call unconditional	O programa prossegue no endereço adr
CC	adr	11011100	- - - - -	3 11/17	Call on carry	Se Carry-Bit = 1 o programa prossegue no endereço adr
CNC	adr	110101100	- - - - -	3 11/17	Call on no carry	Se Carry-Bit = $\phi$ o programa prossegue no endereço adr
CZ	adr	11001100	- - - - -	3 11/17	Call on zero	Se Zero-Bit = 1 o programa prossegue no endereço adr
CNZ	adr	11000100	- - - - -	3 11/17	Call on no zero	Se Zero-Bit = $\phi$ o programa prossegue no endereço adr
CM	adr	111111100	- - - - -	3 11/17	Call on minus	Se Sign-Bit = 1 o programa prossegue no endereço adr
CP	adr	11110100	- - - - -	3 11/17	Call on positiv	Se Sign-Bit = $\phi$ o programa prossegue no endereço adr
CPE	adr	11101100	- - - - -	3 11/17	Call on parity even	Se Parit-Bit = 1 o programa prossegue no endereço adr
CPO	adr	11100100	- - - - -	3 11/17	Call on parity odd	Se Parit-Bit = $\phi$ o programa prossegue no endereço adr
RST	konst	11nnnn111	- - - - -	1 11	Restart	$\phi \leqslant$ constante $\leqslant 7$ o programa prossegue no endereço 8 x constante

### b) Instruções de retorno

RET		11001001	- - - - -	1 10	Return	O programa prossegue no endereço preservado pela palavra armazenada, indicada pelo stack pointer
RC		11011000	- - - - -	1 5/11	Return on carry	Se Carry-Bit = 1 o programa prossegue no endereço preservado pela palavra armazenada, indicada pelo stack pointer
RNC		11010000	- - - - -	1 5/11	Return on no carry	Se Carry-Bit = $\phi$ o programa prossegue no endereço preservado pela palavra armazenada, indicada pelo stack pointer
RZ		11001000	- - - - -	1 5/11	Return on zero	Se Zero-Bit = 1 o programa prossegue no endereço preservado pela palavra armazenada, indicada pelo stack pointer
RNZ		11000000	- - - - -	1 5/11	Return on no zero	Se Zero-Bit = $\phi$ o programa prossegue no endereço preservado pela palavra armazenada, indicada pelo stack pointer
RM		11111000	- - - - -	1 5/11	Return on minus	Se Sign-Bit = 1 o programa prossegue no endereço preservado pela palavra armazenada, indicada pelo stack pointer
RP		11110000	- - - - -	1 5/11	Return on positiv	Se Sign-Bit = $\phi$ o programa prossegue no endereço preservado pela palavra armazenada, indicada pelo stack pointer
RPE		11101000	- - - - -	1 5/11	Return on parity even	Se Parit-Bit = 1 o programa prossegue no endereço preservado pela palavra armazenada, indicada pelo stack pointer
RPO		11100000	- - - - -	1 5/11	Return on parity odd	Se Parit-Bit = $\phi$ o programa prossegue no endereço preservado pela palavra armazenada, indicada pelo stack pointer

## INTERRUPÇÕES DO PROGRAMA

EI		11111011	- - - - -	1 4	Enable interrupts	O Flip-flop INTE é "set"; o microprocessador está apto a reconhecer e responder a interrupções
DI		11110011	- - - - -	1 4	Disable interrupts	O Flip-flop INTE é "reset"; o microprocessador ignora os pedidos de interrupção

## OUTRAS INSTRUÇÕES

HLT		01110110	- - - - -	1 7	Halt	O programa para até ocorrer um pedido de interrupção
NOP		00000000	- - - - -	1 4	No operation	Nenhuma execução, Instrução "vazia"

## RIM - SIM 8085 Instruções

RIM		00100000	- - - - -	1 4	Read Interrupt mask	Leitura da máscara de interrupção e entrada serial no bit 7 do acumulador.
SIM		00110000	- - - - -	1 4	Set interrupt mask	Setar máscara de interrupção e saída serial, através do bit 7 do acumulador.

## PSEUDO-INSTRUÇÕES

ORG	adr	-	-	-	Origin	O contador de instruções do programa Assembler é preenchido com o valor do operando
name	EQU	exp	-	-	Equate	O valor da exp é designada pelo símbolo "name". O "name" pode ocorrer apenas uma vez no campo da instrução EQU

END	-	-	-	-	-	-	-	-	-	-	-	-	END significa para o programa Assembler que o fim do programa foi atingido
IF	exp	-	-	-	-	-	-	-	-	-	-	-	Assembly condicional, se o valor da exp é igual a zero são ignoradas as expressões entre IF e ENDIF pelo programa Assembler
ENDIF	-	-	-	-	-	-	-	-	-	-	-	-	Definição do macro: Expressões entre MACRO e ENDM são aceitas como macro denominadas "name"
name MACRO	-	-	-	-	-	-	-	-	-	-	-	-	"text" é inserido como título de cada lista (comprimento máx. 68 caracteres)
ENDM	-	-	-	-	-	-	-	-	-	-	-	-	
TITLE "text"	-	-	-	-	-	-	-	-	-	-	-	-	

## Conjunto das instruções em Hexadecimal

Hex	Mnemonico	Hex	Mnemonico	Hex	Mnemonico	Hex	Mnemonico	Hex	Mnemonico	Hex	Mnemonico	Hex	Mnemonico
00	NOP	2B	DCX H	56	MOV D,M	81	ADD C	AC	XRA H	D7	RST 2		
01	LXI B,D16	2C	INR L	57	MOV D,A	82	ADD D	AD	XRA L	D8	RC		
02	STAX B	2D	DCR L	58	MOV E,B	83	ADD E	AE	XRA M	D9	- - -		
03	INX B	2E	MVI L,D8	59	MOV E,C	84	ADD H	AF	XRA A	DA	JC Adr		
04	INR B	2F	CMA	5A	MOV E,D	85	ADD L	B0	ORA B	DB	IN D8		
05	DCR B	30	- - -	5B	MOV E,E	86	ADD M	B1	ORA C	DC	CC Adr		
06	MVI B,D8	31	LXI SP,D16	5C	MOV E,H	87	ADD A	B2	ORA D	DD	- - -		
07	RLC	32	STA Adr	5D	MOV E,L	88	ADC B	B3	ORA E	DE	SBI D8		
08	- - -	33	INX SP	5E	MOV E,M	89	ADC C	B4	ORA H	DF	RST 3		
09	DAD B	34	INR M	5F	MOV E,A	8A	ADC D	B5	ORA L	E0	RPO		
0A	LDAX B	35	DCR M	60	MOV H,B	8B	ADC E	B6	ORA M	E1	POP H		
0B	DCX B	36	MVI M,D8	61	MOV H,C	8C	ADC H	B7	ORA A	E2	JPO Adr		
0C	INR C	37	STC	62	MOV H,D	8D	ADC L	B8	CMP B	E3	XTHL		
0D	DCR C	38	- - -	63	MOV H,E	8E	ADC M	B9	CMP C	E4	CPO Adr		
0E	MVI C,D8	39	DAD SP	64	MOV H,N	8F	ADC A	BA	CMP D	E5	PUSH H		
0F	RRC	3A	LDA Adr	65	MOV H,I	90	SUB B	BB	CMP E	E6	ANI D8		
10	- - -	3B	DCX SP	66	MOV H,M	91	SUB C	BC	CMP H	E7	RST 4		
11	LXI D,D16	3C	INR A	67	MOV H,A	92	SUB D	BD	CMP L	E8	RPE		
12	STAX D	3D	DCR A	68	MOV L,B	93	SUB E	BE	CMP M	E9	PCHL		
13	INX D	3E	MVI A,D8	69	MOV L,C	94	SUB H	BF	CMP A	EA	JPE Adr		
14	INR D	3F	CMC	6A	MOV L,D	95	SUB L	C0	RNZ	EB	XCHG		
15	DCR D	40	MOV B,B	6B	MOV L,E	96	SUB M	C1	POP B	EC	CPE Adr		
16	MVI D,D8	41	MOV B,C	6C	MOV L,H	97	SUB A	C2	JNZ Adr	ED	- - -		
17	RAL	42	MOV B,D	6D	MOV L,L	98	SSB B	C3	JMP Adr	EE	XRI D8		
18	- - -	43	MOV B,E	6E	MOV L,M	99	SSB C	C4	CNZ Adr	EF	RST 5		
19	DAD D	44	MOV B,H	6F	MOV L,A	9A	SSB D	C5	PUSH B	F0	RP		
1A	LDAX D	45	MOV B,L	70	MOV M,B	9B	SSB E	C6	ADI D8	F1	POP PSW		
1B	DCX D	46	MOV B,M	71	MOV M,C	9C	SSB H	C7	RST 0	F2	JP Adr		
1C	INR E	47	MOV B,A	72	MOV M,D	9D	SSB L	C8	RZ	F3	DI		
1D	DCR E	48	MOV C,B	73	MOV M,E	9E	SSB M	C9	RET	F4	CP Adr		
1E	MVI E,D8	49	MOV C,C	74	MOV M,H	9F	SSB A	CA	JZ Adr	F5	PUSH PSW		
1F	RAR	4A	MOV C,D	75	MOV M,L	A0	ANA B	CB	- - -	F6	ORI D8		
20	- - -	4B	MOV C,E	76	HLT	A1	ANA C	CC	CZ Adr	F7	RST 6		
21	LXI H,D16	4C	MOV C,H	77	MOV M,A	A2	ANA D	CD	CALL Adr	F8	RM		
22	SHLD Adr	4D	MOV C,L	78	MOV A,B	A3	ANA E	CE	ACI D8	F9	SPHL		
23	INX H	4E	MOV C,M	79	MOV A,C	A4	ANA H	CF	RST 1	FA	JM Adr		
24	INR H	4F	MOV C,A	7A	MOV A,D	A5	ANA L	D0	RNC	FB	EI		
25	DCR H	50	MOV D,B	7B	MOV A,E	A6	ANA M	D1	POP D	FC	CM Adr		
26	MVI H,D8	51	MOV D,C	7C	MOV A,H	A7	ANA A	D2	JNC Adr	FD	- - -		
27	DAA	52	MOV D,D	7D	MOV A,L	A8	XRA B	D3	OUT D8	FE	CPI D8		
28	- - -	53	MOV D,E	7E	MOV A,M	A9	XRA C	D4	CNC Adr	FF	RST 7		
29	DAD H	54	MOV D,H	7F	MOV A,A	AA	XRA D	D5	PUSH D				
2A	LHLD Adr	55	MOV D,L	80	ADD B	AB	XRA E	D6	SUI D8				

D8 = Constante ou expressão lógica/aritmética com 8 bits de comprimento.

D16 = Constante ou expressão lógica/aritmética com 16 bits de comprimento.

Adr = Endereço de 16 bits

## Tabela de Conversão

### Código Hexadecimal → Código ASCII

Hex	ASCII	Hex	ASCII	Hex	ASCII	Hex	ASCII	Hex	ASCII	Hex	ASCII	Hex	ASCII
00	NUL	10	DLE	20	SP	30	0	40	@	50	P	60	À
01	SOH	11	DC1 (X-ON)	21	!	31	1	41	A	51	Q	61	a
02	STX	12	DC2 (TAPE)	22	"	32	2	42	B	52	R	62	b
03	ETX	13	DC3 (X-OFF)	23	#	33	3	43	C	53	S	63	c
04	EOT	14	DC4 (TAPE)	24	\$	34	4	44	D	54	T	64	d
05	ENQ	15	NAK	25	%	35	5	45	E	55	U	65	e
06	ACK	16	SYN	26	&	36	6	46	F	56	V	66	f
07	BEL	17	ETB	27	'	37	7	47	G	57	W	67	g
08	BS	18	CAN	28	(	38	8	48	H	58	X	68	h
09	HT	19	EM	29	)	39	9	49	I	59	Y	69	i
0A	LF	1A	SUB	2A	*	3A	:	4A	J	5A	Z	6A	j
08	VT	1B	ESC †	2B	+	3B	:	4B	K	5B	[	6B	k
0C	FF	1C	FS	2C	.	3C	<	4C	L	5C	\	6C	l
0D	CR	1D	GS	2D	-	3D	=	4D	M	5D	]	6D	m
0E	SO	1E	RS	2E	.	3E	>	4E	N	5E	^ (↑)	6E	n
0F	SI	1F	US	2F	/	3F	?	4F	O	5F	= (←)	6F	o
													7F DEL (RUB OUT)

† (ESC) = 0x1B

# Conjunto de Instruções

Mnemonico	Binário	Bits de Flag influenciados	Número de Bytes-Estados	Instrução em Inglês	Função da Instrução
-----------	---------	----------------------------	-------------------------	---------------------	---------------------

## INSTRUÇÕES DE TRANSFERÊNCIA

### a) Registrador → Registrador

MOV	r <sub>1</sub> , r <sub>2</sub>	01 d d d sss	- - - - -	1    5	Move register to register	r <sub>1</sub> , r <sub>2</sub> = A, B, C, D, E, H ou L. Carrega o registrador r <sub>1</sub> com o conteúdo do registrador r <sub>2</sub>
XCHG		11101011	- - - - -	1    4	Exchange D&E, H&L	Troca os conteúdos dos pares de registradores (D, E) e (H, L)
XTHL		11100011	- - - - -	1    18	Exchange top of stack H&L	Troca os conteúdos do par de registradores (H, L) e o da palavra endereçada pelo stack-pointer
SPHL		11111001	- - - - -	1    5	H&L to stackpointer	Carrega o stack-pointer com os conteúdos do par de registradores H, L

### b) Memória, Periférico → Registrador

MOV	r <sub>1</sub> , M	01 d d d 110	- - - - -	1    7	Move memory to register	r <sub>1</sub> = A, B, C, D, E, H ou L. Carrega o registro r <sub>1</sub> com o conteúdo do byte de memória endereçada pelo conteúdo dos registradores (H, L)
LDA	adr	00111010	- - - - -	3    13	Load accu direct	Carrega o acumulador com o conteúdo do endereço adr
LDAX	rp	00rr1010	- - - - -	1    7	Load accu indirect	rp = B ou D. Carrega o acumulador com o conteúdo da memória endereçado pelo conteúdo do par de registradores (H, L)
LHLD	adr	00101010	- - - - -	3    16	Load H&L direct	Carrega o par de registradores (H, L) com o conteúdo do endereço adr e (adr + 1)
POP	rp PSW	11rr0001	Z, S, P, CY, AC	1    10	Pop register pair off stack	rp = B, D, H, PSW: O par de registradores rp é carregado com a palavra endereçada pelo stack-pointer
IN	nr	11011011	- - - - -	2    10	Input	O acumulador é carregado com o conteúdo da porta de entrada nr (nr ≤ 255)

### c) Constante → Par de Registradores

LXI	rp, adr	00rr0001	- - - - -	3    10	Load register pair immediate	rp = B, D, H, SP: Carrega o par de registradores rp com o valor adr
-----	---------	----------	-----------	---------	------------------------------	---

### d) Registrador → Memória - Periférico

MOV	M, r <sub>1</sub>	01110sss	- - - - -	1    7	Move register to memory	r <sub>1</sub> = A, B, C, D, E, H ou L: Armazena o conteúdo do registrador r <sub>1</sub> na posição de memória endereçada pelo par de registradores (H, L)
STA	adr	00110010	- - - - -	3    13	Store accu direct	Armazena o conteúdo do acumulador no endereço adr
STAX	rp	00rr0010	- - - - -	1    7	Store accu indirect	rp = B, D: Armazena o conteúdo do acumulador no byte endereçado pelo conteúdo do par de registradores rp
SHLD	adr	00100010	- - - - -	3    16	Store H&L direct	Armazena o conteúdo dos registradores (H, L) no endereço adr e (adr + 1)
PUSH	rp	11rr0101	- - - - -	1    11	Push register pair rp on stack	rp = B, D, H, PSW: O conteúdo do par de registradores rp é transferido para a palavra endereçada pelo stack-pointer
OUT	nr	11010011	- - - - -	2    10	Output	O conteúdo do acumulador é carregado na porta de saída nr (nr ≤ 255)

### e) Constante → Registrador-Memória

MVI	M, konst	00110110	- - - - -	2    10	Move to memory immediate	Move o valor de konst (konst ≤ 255) para a posição de memória endereçada pelo conteúdo do par de registradores (H, L)
MVI	r <sub>1</sub> , konst	00 d d d 110	- - - - -	2    7	Move immediate Register	r <sub>1</sub> = A, B, C, D, E, H, L: Carrega o registrador r <sub>1</sub> com o valor da constante konst (konst ≤ 255)

## OPERAÇÕES ARITMÉTICAS

INR	r <sub>1</sub>	00 d d d 100	Z, S, P, - AC	1    5	Increment register	r <sub>1</sub> = A, B, C, D, E, H ou L: Adiciona 1 ao conteúdo do registrador r <sub>1</sub>
INR	M	00110100	Z, S, P, - AC	1    10	Increment memory	Adiciona 1 ao byte endereçado pelos conteúdos do par de registradores (H, L)
DCR	r <sub>1</sub>	00 d d d 101	Z, S, P, - AC	1    5	Decrement register	r <sub>1</sub> = A, B, C, D, E, H ou L: Subtrai 1 do conteúdo do registrador r <sub>1</sub> uma unidade
DCR	M	00110101	Z, S, P, - AC	1    10	Decrement memory	Subtrai 1 do byte endereçado pelo par de registradores (H, L) uma unidade
INX	rp	00rr0011	- - - - -	1    5	Increment register pair	rp = B, D, H, SP: Os conteúdos do par de registradores rp são incrementados de uma unidade
DCX	rp	00rr1011	- - - - -	1    5	Decrement register pair	rp = B, D, H, SP: Os conteúdos do par de registradores rp são decrementados de uma unidade
ADD	r <sub>1</sub>	10000sss	Z, S, P, CY, AC	1    4	Add register to accu	r <sub>1</sub> = A, B, C, D, E, H ou L: Adiciona o conteúdo do registrador r <sub>1</sub> ao conteúdo do acumulador
ADD	M	10000110	Z, S, P, CY, AC	1    7	Add memory to accu	Adiciona o conteúdo da posição de memória, endereçada pelo par de registradores (H, L), ao conteúdo do acumulador
ADC	r <sub>1</sub>	10001sss	Z, S, P, CY, AC	1    4	Add register to accu with carry	r <sub>1</sub> = A, B, C, D, E, H ou L: Adiciona o conteúdo do registrador r <sub>1</sub> e o conteúdo do Carry-Bit ao conteúdo do acumulador
ADC	M	10001110	Z, S, P, CY, AC	1    7	Add memory to accu with carry	Adiciona o conteúdo da posição de memória, endereçada pelo par de registradores (H, L), e o conteúdo do Carry-Bit ao conteúdo do acumulador

DAD	rp	00rr1001	- - CY -	1	10	Add register pair to H and L	acumulador
SUB	r <sub>1</sub>	10010sss	Z,S,P,CY,AC	1	4	Subtract register from accu	rp = B, D, H, SP: Adiciona o conteúdo do par de registradores rp ao conteúdo do par de registradores (H, L) e o resultado fica em (H, L) r <sub>1</sub> = A, B, C, D, E, H ou L: Subtrai do conteúdo do acumulador o conteúdo do registrador r <sub>1</sub>
SUB	M	10010110	Z,S,P,CY,AC	1	7	Subtract memory from accu	Subtrai do acumulador o conteúdo da posição de memória endereçada pelo par de registradores (H, L)
SBB	r <sub>1</sub>	10011sss	Z,S,P,CY,AC	1	4	Subtract register from accu with borrow	r <sub>1</sub> = A, B, C, D, E, H ou L: Subtrai do acumulador o conteúdo do Carry-Bit e o conteúdo do registrador r <sub>1</sub>
SBB	M	10011110	Z,S,P,CY,AC	1	7	Subtract memory from accu with borrow	Subtrai do acumulador o conteúdo do Carry-Bit e a posição de memória endereçada pelo par de registradores (H, L)
ADI	konst	11000110	Z,S,P,CY,AC	2	7	Add immediate to accu	Adiciona uma constante (konst $\leq 255$ ) ao conteúdo do acumulador
ACI	konst	11001110	Z,S,P,CY,AC	2	7	Add immediate to accu with carry	Adiciona uma constante (konst $\leq 255$ ) e o Carry-Bit ao conteúdo do acumulador
SUI	konst	11010110	Z,S,P,CY,AC	2	7	Subtract immediate from accu	Subtrai do acumulador uma constante (konst $\leq 255$ )
SBI	konst	11011110	Z,S,P,CY,AC	2	7	Subtract immediate from accu with borrow	Subtrai do acumulador uma constante (konst $\leq 255$ ) e o Carry-Bit
DAA		00100111	Z,S,P,CY,AC	1	4	Decimal adjust accu	Ajusta o acumulador em um número de dois dígitos decimais

## OPERAÇÕES LÓGICAS

CMA		00101111		1	4	Complement accu	Complementa o conteúdo do acumulador
ANA	r <sub>1</sub>	10100sss	Z,S,P,CY,AC	1	4	And register with accu	r <sub>1</sub> = A, B, C, D, E, H ou L: O conteúdo do acumulador e o conteúdo do registrador r <sub>1</sub> são combinados logicamente (AND) entre si
ANA	M	10100110	Z,S,P,CY,AC	1	7	And memory with accu	O conteúdo da posição de memória endereçada pelo par de registradores (H, L) e o conteúdo do acumulador são combinados logicamente (AND) entre si
ANI	konst	11100110	Z,S,P,CY,AC	2	7	And immediate with accu	O conteúdo do acumulador é combinado logicamente (AND) com a constante konst, (konst $\leq 255$ )
ORA	r <sub>1</sub>	10110sss	Z,S,P,CY,AC	1	4	Or register with accu	r <sub>1</sub> = A, B, C, D, E, H ou L: O conteúdo do acumulador e o conteúdo do registrador r <sub>1</sub> são combinados logicamente (OR) entre si
ORA	M	10110110	Z,S,P,CY,AC	1	7	Or memory with accu	O conteúdo da posição de memória endereçada pelo conteúdo do par de registradores (H, L) e o conteúdo do acumulador são combinados logicamente (OR) entre si
ORI	konst	11110110	Z,S,P,CY,AC	2	7	Or immediate with accu	O conteúdo do acumulador é combinado logicamente (OR) com uma constante (konst $\leq 255$ )
XRA	r <sub>1</sub>	10101sss	Z,S,P,CY,AC	1	4	Exclusive Or register with accu	r <sub>1</sub> = A, B, C, D, E, H ou L: Combina logicamente (Exclusive-Or) o conteúdo do acumulador com o conteúdo do registrador r <sub>1</sub>
XRA	M	10101110	Z,S,P,CY,AC	1	7	Exclusive Or memory with accu	Combina logicamente (Exclusive-Or) o conteúdo da posição de memória endereçada pelo par de registradores (H, L) com o conteúdo do acumulador
XRI	konst	11101110	Z,S,P,CY,AC	2	7	Exclusive Or immediate with accu	Combina logicamente (Exclusive-Or) o conteúdo do acumulador com uma constante (konst $\leq 255$ )
CMP	r <sub>1</sub>	10111sss	Z,S,P,CY,AC	1	4	Compare register with accu	r <sub>1</sub> = A, B, C, D, E, H ou L: O conteúdo do acumulador é comparado com o conteúdo do registrador r <sub>1</sub>
CMP	M	10111110	Z,S,P,CY,AC	1	7	Compare memory with accu	O conteúdo da posição de memória endereçada pelo conteúdo do par de registradores (H, L) é comparado com o conteúdo do acumulador
CPI	konst	11111110	Z,S,P,CY,AC	2	7	Compare immediate with accu	Compara o conteúdo do acumulador com uma constante (konst $\leq 255$ )

## INSTRUÇÕES DE REGISTRADOR

### a) Rotação do acumulador

RLC		00000111	- - CY -	1	4	Rotate accu left	O conteúdo do acumulador é girado de 1 bit para a esquerda. O bit 2 <sup>7</sup> é escrito no Carry-Bit
RRC		00001111	- - CY -	1	4	Rotate accu right	O conteúdo do acumulador é girado de 1 bit para a direita. O bit 2 <sup>0</sup> é escrito no Carry-Bit
RAL		00010111	- - CY -	1	4	Rotate accu left through carry	O conteúdo do acumulador é girado de 1 bit para a esquerda. O bit 2 <sup>7</sup> é escrito no Carry-Bit e o Carry-Bit é escrito no bit 2 <sup>0</sup>
RAR		00011111	- - CY -	1	4	Rotate accu right through carry	O conteúdo do acumulador é girado de 1 bit para a direita. O bit 2 <sup>0</sup> é escrito no Carry-Bit e o Carry-Bit é escrito no bit 2 <sup>7</sup>

### b) Instruções do Carry-Bit

CMC		00111111	- - CY -	1	4	Complement carry	Complementa o Carry-Bit
STC		00110111	- - CY -	1	4	Set carry	Posiciona o Carry-Bit

## INSTRUÇÕES DE JUMP

### a) Jump incondicional

PCHL		11101001	- - - -	1	5	H&L to program counter	O programa prossegue no endereço contido no par de registradores (H, L)
JMP	adr	11000011	- - - -	3	10	Jump unconditional	O programa prossegue no endereço adr

### b) Jump condicional

JC	adr	11011010	- - - -	3	10	Jump on carry	Se Carry-Bit = 1 o programa prossegue no endereço adr
JNC	adr	11010010	- - - -	3	10	Jump on no carry	Se Carry-Bit = 0 o programa prossegue no endereço adr
JZ	adr	11001010	- - - -	3	10	Jump on zero	Se Zero-Bit = 1 o programa prossegue no endereço adr
JNZ	adr	11000010	- - - -	3	10	Jump on no zero	Se Zero-Bit = 0 o programa prossegue no endereço adr
JM	adr	11111010	- - - -	3	10	Jump on minus	Se Sign-Bit = 1 o programa prossegue no endereço adr
JP	adr	11110010	- - - -	3	10	Jump on positiv	Se Sign-Bit = 0 o programa prossegue no endereço adr
JPE	adr	11101010	- - - -	3	10	Jump on parity even	Se Parity-Bit = 1 o programa prossegue no endereço adr
JPO	adr	11100010	- - - -	3	10	Jump on parity odd	Se Parity-Bit = 0 o programa prossegue no endereço adr