INTEL 8008 Instructions

CPU control group

binary	old	new	Description
0 0 0 0 0 0 x	HLT	HLT	
1 1 1 1 1 1 1	HLT	HLT	

Input and output group

binary	old	new	Description
0 1 0 0 M M M 1	INP	IN	port MMM
0 1 R R M M M 1	OUT	OUT	port RRMMM (RR <> 0)

Jump group

binary	old	new	Description
0 1 x x x 1 0 0	JMP	JMP	unconditionally jump
0 1 0 0 0 0 0 0	JFC	JNC	JMP if carry = 0
0 1 0 0 1 0 0 0	JFZ	JNZ	JMP if result <> 0
0 1 0 1 0 0 0 0	JFS	JP	JMP if sign = 0 (positive)
0 1 0 1 1 0 0 0	JFP	JPO	JMP if parity = odd
0 1 1 0 0 0 0 0	JC	JC	JMP if carry = 1
0 1 1 0 1 0 0 0	JZ	JZ	JMP if result = 0
0 1 1 1 0 0 0 0	JS	JM	JMP if sign = 1 (negative)
0 1 1 1 1 0 0 0	JP	JPE	JMP if parity = even

Call and return group

binary	old	new	Description
0 1 x x x 1 1 0	CAL	CALL	unconditionally call subroutine
0 1 0 0 0 0 1 0	CFC	CNC	CALL if carry = 0
0 1 0 0 1 0 1 0	CFZ	CNZ	CALL if result <> 0
0 1 0 1 0 0 1 0	CFS	CP	CALL if sign = 0 (positive)
0 1 0 1 1 0 1 0	CFP	CPO	CALL if parity = odd

0 1 1 0 0 0 1 0	CC	CC	CALL if carry = 1
0 1 1 0 1 0 1 0	CZ	CZ	CALL if result = 0
0 1 1 1 0 0 1 0	CS	CM	CALL if sign = 1 (negative)
0 1 1 1 1 0 1 0	CP	CPE	CALL if parity = even
0 0 x x x 1 1 1	RET	RET	unconditionally return
0 0 0 0 0 0 1 1	RFC	RNC	RET if carry = 0
0 0 0 0 1 0 1 1	RFZ	RNZ	RET if result <> 0
0 0 0 1 0 0 1 1	RFS	RP	RET if sign = 0 (positive)
0 0 0 1 1 0 1 1	RFP	RPO	RET if parity = odd
0 0 1 0 0 0 1 1	RC	RC	RET if carry = 1
0 0 1 0 1 0 1 1	RZ	RZ	RET if result = 0
0 0 1 1 0 0 1 1	RS	RM	RET if sign = 1 (negative)
0 0 1 1 1 0 1 1	RP	RPE	RET if parity = even
0 0 A A A 1 0 1	RST	RST	call subroutine at adrs AAA000

Load group

binary	old	new	Description
1 1 D D D S S S	Lds	MOV d,s	load d with content of s
1 1 D D D 1 1 1	LdM	MOV d,M	load d with content of Mem
1 1 1 1 1 s s s	LMs	MOV M,s	load M with content of s
0 0 d d d 1 1 0	LdI	MVI d	Load register d with data
0 0 1 1 1 1 1 0	LMI	MVI M	Load Memory M with data b

Arithmetic group

binary	old	new	Description
1 0 0 0 0 s s s	ADs	ADD s	add contents of s to A
1 0 0 0 0 1 1 1	ADM	ADD M	add contents of M to A
0 0 0 0 0 1 0 0	ADI	ADI b	add constant b to A

1 0 0 0 1 s s s	ACs	ADC s	add contents of s + CY to A
1 0 0 0 1 1 1 1	ACM	ADC M	add contents of M + CY to A
0 0 0 0 1 1 0 0	ACI	ACI b	add constant b + CY to A
1 0 0 1 0 s s s	SUs	SUB s	sub contents of s from A
1 0 0 1 0 1 1 1	SUM	SUB M	sub contents of M from A
0 0 0 1 0 1 0 0	SUI	SUI b	sub constant b from A
1 0 0 1 1 s s s	SBs	SBB s	sub contents of s + CY from A
1 0 0 1 1 1 1 1	SBM	SBB M	sub contents of M + CY from A
0 0 0 1 1 1 0 0	SBI	SBI b	sub constant b + CY from A
1 0 1 0 0 s s s	NDs	ANA s	logical AND of s and A to A
1 0 1 0 0 1 1 1	NDM	ANA M	logical AND of M and A to A
0 0 1 0 0 1 0 0	NDI	ANI b	logical AND of const b and A to A
1 0 1 0 1 s s s	XRs	XRA s	logical XOR of s and A to A
1 0 1 0 1 1 1 1	XRM	XRA M	logical XOR of M and A to A
0 0 1 0 1 1 0 0	XRI	XRI b	logical XOR of const b and A to A
1 0 1 1 0 s s s	ORs	ORA s	logical OR of s and A to A
1 0 1 1 0 1 1 1	ORM	ORA M	logical OR of M and A to A
0 0 1 1 0 1 0 0	ORI	ORI b	logical OR of const b and A to A
1 0 1 1 1 s s s	CPs	CMP s	compare s with A, set flags
1 0 1 1 1 1 1 1	CPM	CMP M	compare M with A, set flags
0 0 1 1 1 1 0 0	CPI	CPI b	compare const b with A, set flags
0 0 d d d 0 0 0	INd	INR d	increment register d (d<>A)
0 0 d d d 0 0 1	DCd	DCR r	decrement register d (d<>A)

Rotate group

binary	old	new	Description
0 0 0 0 0 0 1 0	RLC	RLC	rotate content of A left

0 0 0 0 1 0 1 0	RRC	RRC	rotate content of A right
0 0 0 1 0 0 1 0	RAL	RAL	rotate content of A left through CY
0 0 0 1 1 0 1 0	RAR	RAR	rotate content of A right through CY