

(micro-)instruction overview

hex	Instruction / stages:	0	1	2	3	4	5	6	7
NO Operation									
x00	NOP (No Operation)	PCO, MAI <i>point to PC value</i>	MO, IRI, CE <i>Load val into IR, count enable</i>						
LOAD instructions									
x01	LDAi (Load A immediate)	PCO, MAI	MO, IRI, CE	PCO, MAI <i>point to val</i>	MO, AI, CE <i>val to A</i>				
x02	LDAm (Load A from memory)	PCO, MAI	MO, IRI, CE	PCO, MAI <i>point to adr</i>	MO, MAI <i>point to val</i>	MO, AI, CE <i>val to A</i>			
x07	LDBi (Load B immediate)	PCO, MAI	MO, IRI, CE	PCO, MAI <i>point to val</i>	MO, BI, CE <i>val to B</i>				
x08	LDBm (Load B from memory)	PCO, MAI	MO, IRI, CE	PCO, MAI <i>point to adr</i>	MO, MAI <i>point to val</i>	MO, BI, CE <i>val to B</i>			
STORE instructions									
x10	STAm (STore A to memory)	PCO, MAI	MO, IRI, CE	PCO, MAI <i>point to adr</i>	MO, MAI <i>load adr</i>	AO, MI, CE <i>store A</i>			
x11	STBm (STore B to memory)	PCO, MAI	MO, IRI, CE	PCO, MAI <i>point to adr</i>	MO, MAI <i>load adr</i>	BO, MI, CE <i>store B</i>			
ARITHMETIC instructions									
x20	ADD (Add B to A)	PCO, MAI	MO, IRI, CE	AL3, AL0, EO, AI, CE <i>add B to A</i>					
x21	ADDi (Add immediate)	PCO, MAI	MO, IRI, CE	PCO, MAI <i>point to val</i>	MO, BI <i>load val in B</i>	AL3, AL0, EO, AI, CE <i>add to A</i>			
x22	ADDm (Add from memory)	PCO, MAI	MO, IRI, CE	PCO, MAI <i>point to adr</i>	MO, MAI <i>point to val</i>	MO, BI <i>load val in B</i>	AL3, AL0, EO, AI, CE <i>add to A</i>		
x24	SUB (Subtract B from A)	PCO, MAI	MO, IRI, CE	AL2, AL1, EO, AI, CE <i>sub B from A</i>					
x25	SUBi (Subtract immediate)	PCO, MAI	MO, IRI, CE	PCO, MAI <i>point to val</i>	MO, BI <i>load val in B</i>	AL2, AL1, EO, AI, CE <i>sub from A</i>			
x26	SUBm (Subtract from memory)	PCO, MAI	MO, IRI, CE	PCO, MAI <i>point to adr</i>	MO, MAI <i>point to val</i>	MO, BI <i>load val in B</i>	AL2, AL1, EO, AI, CE <i>sub from A</i>		
x28	INCA (Add 1 to register A)	PCO, MAI	MO, IRI, CE	AO, BI <i>A to B</i>	ALM, AL3, AL2, EO, AI <i>Make A = 1</i>	AL3, AL0, EO, AI <i>add B to A</i>			
x29	INCB (Add 1 to register B)	PCO, MAI	MO, IRI, CE	ALM, AL3, AL2, EO, AI	AL3, AL0, EO, AI	AO, BI	Destructive to A register contents!		

	INCB			Make A = 1	add B to A	A to B (so B = 1)
--	------	--	--	------------	------------	-------------------

LOGIC instructions

x30	XOR <i>A = A XOR B</i>	PCO, MAI	MO, IRI, CE	ALM, AL2, AL1, EO, AI, CE <i>xor B into A</i>			
x31	XORi (Xor immediate) <i>A = A XOR imm.value</i>	PCO, MAI	MO, IRI, CE	PCO, MAI <i>point to val</i>	MO, BI <i>load val in B</i>	ALM, AL2, AL1, EO, AI, CE <i>xor B into A</i>	
x32	XORm <i>XORm adr</i>	PCO, MAI	MO, IRI, CE	PCO, MAI <i>point to adr</i>	MO, MAI <i>point to val</i>	MO, BI <i>load val in B</i>	ALM, AL2, AL1, EO, AI, CE <i>xor B into A</i>
x34	AND <i>A = A AND B</i>	PCO, MAI	MO, IRI, CE	ALM, AL3, AL1, AL0, EO, AI, CE <i>and B into A</i>			
x35	ANDi (AND immediate) <i>A = A AND imm.value</i>	PCO, MAI	MO, IRI, CE	PCO, MAI <i>point to val</i>	MO, BI <i>load val in B</i>	ALM, AL3, AL1, AL0, EO, AI, CE <i>and B into A</i>	
x36	ANDm <i>ANDm adr</i>	PCO, MAI	MO, IRI, CE	PCO, MAI <i>point to adr</i>	MO, MAI <i>point to val</i>	MO, BI <i>load val in B</i>	ALM, AL3, AL1, AL0, EO, AI, CE <i>and B into A</i>
x38	OR <i>A = A OR B</i>	PCO, MAI	MO, IRI, CE	ALM, AL3, AL2, AL1, EO, AI, CE <i>or B into A</i>			
x39	ORi (OR immediate) <i>A = A OR imm.value</i>	PCO, MAI	MO, IRI, CE	PCO, MAI <i>point to val</i>	MO, BI <i>load val in B</i>	ALM, AL3, AL2, AL1, EO, AI, CE <i>or B into A</i>	
x3A	ORm <i>ORm adr</i>	PCO, MAI	MO, IRI, CE	PCO, MAI <i>point to adr</i>	MO, MAI <i>point to val</i>	MO, BI <i>load val in B</i>	ALM, AL3, AL2, AL1, EO, AI, CE <i>or B into A</i>
x3C	NOTA <i>A = NOT A</i>	PCO, MAI	MO, IRI, CE	ALM, EO, AI <i>NOT A to A</i>			
x3D	NOTB <i>B = NOT B</i>	PCO, MAI	MO, IRI, CE	ALM, AL2, AL0, EO, BI <i>NOT B to B</i>			
x3E	CLRA <i>A = 0</i>	PCO, MAI	MO, IRI, CE	ALM, AL1, AL0, EO, AI <i>A = 0</i>			
x3F	CLRB <i>B = 0</i>	PCO, MAI	MO, IRI, CE	ALM, AL1, AL0, EO, BI <i>B = 0</i>			

JUMP instructions

x40	JMPi (Jump immediate) <i>JMPi adr</i>	PCO, MAI	MO, IRI, CE	PCO, MAI <i>point to adr</i>	MO, PCI <i>adr to PC</i>		
x41	JMPm (Jump indirect) <i>JMPm adr</i>	PCO, MAI	MO, IRI, CE	PCO, MAI <i>point to adr</i>	MO, MAI <i>load adr</i>	MO, PCI <i>adr to PC</i>	
x44	JPCi (Jump when carry immediate) <i>JPCi adr</i>	PCO, MAI	MO, IRI, CE	PCO, MAI <i>point to adr</i>	CE <i>bump PC</i>	MO, PCiC <i>adr to PC when carry</i>	
x45	JPCm (Jump when carry indirect) <i>JPCm adr</i>	PCO, MAI	MO, IRI, CE	PCO, MAI <i>point to adr</i>	MO, MAI <i>load adr</i>	CE <i>bump PC</i>	MO, PCiC <i>adr to PC when carry</i>

OUTPUT instructions

x50	OUTA (Output A) <i>OUTA</i>	PCO, MAI	MO, IRI, CE	AO, OI <i>A to output</i>			
x51	OUTB (Output B) <i>OUTB</i>	PCO, MAI	MO, IRI, CE	BO, OI <i>B to output</i>			
x54	OUTi (Output immediate) <i>OUTi val</i>	PCO, MAI	MO, IRI, CE	PCO, MAI <i>point to val</i>	MO, OI, CE <i>val to output</i>		
x55	OUTm (Output from memory) <i>OUTi adr</i>	PCO, MAI	MO, IRI, CE	PCO, MAI <i>point to adr</i>	MO, MAI <i>load adr</i>	MO, OI, CE <i>val to output</i>	

MOVE instructions

x60	MOVi (Move immediate) <i>MOVi val, adr</i>	PCO, MAI	MO, IRI, CE	PCO, MAI <i>point to val</i>	MO, BI, CE <i>val in B</i>	PCO, MAI <i>point to adr</i>	BO, MI, CE <i>val to mem</i>	
x61	MOVm (Move mem to mem) <i>MOVm adr, adr</i>	PCO, MAI	MO, IRI, CE	PCO, MAI <i>point to adr1</i>	MO, MAI <i>point to val</i>	MO, BI, CE <i>val in B</i>	PCO, MAI <i>point to adr2</i>	BO, MI, CE <i>val to mem</i>

HALT

xFF	HLT (HaLT program)	PCO, MAI	MO, IRI, CE	HLT			
-----	--------------------	----------	-------------	-----	--	--	--