SR NO	INSTRUCTION	TYPE	DESCRIPTION	EXAMPLE
1	mov b/w x(data),reg	Data transfer	Transfers data into reg	mov b x15,r1 mov w x150,r1
2	mov b/w reg1,reg2	Data transfer	Transfers data of reg1 into reg2	mov b r1,r2 mov w r1,r2
3	store b/w x(memloc),reg	Data transfer	Data of reg is going to be stored at specified memory location (Direct Addressing)	store b x200,r1 store w x197,r1
4	store b/w [reg1],reg2	Data transfer	Data of reg2 is going to be stored at memory location pointed by reg1 (Indirect Addressing)	store b [r1],r2 store w [r1],r2
5	store b/w x(mem-loc)[reg1],reg2	Data transfer	Data of reg2 is going to be stored at memory location reg1+memloc (Indexed Addressing)	store b x200[r1],r2 store w x200[r1],r2
6	load b/w x(memloc),reg	Data transfer	Data of specified memory location is going to be stored at reg (Direct Addressing)	load b x200,r1 load w x197,r1
7	load b/w [reg1],reg2	Data transfer	Data of memory location pointed by reg1 is going to be stored at reg2 (Indirect Addressing)	load b [r1],r2 load w [r1],r2
8	load b/w x(mem-loc)[reg1],reg2	Data transfer	Data of memory location reg1+memloc is going to be stored at reg2 (Indexed Addressing)	load b x200[r1],r2 load w x200[r1],r2
9	out (port-num),reg	Data transfer	Data of reg is going to be transferred at specified port	out x08,r1
10	in(port-num),reg	Data transfer	Here the simulator will ask the user to enter 8 bit binary data, that will be input through specified port and stored in reg	in x08,r1
11	and b/w x(data),reg	Logical	Logical 'AND' between data and reg. The result is stored in reg	and b x15,r1 and w x150,r1
12	and b/w reg1,reg2	Logical	Logical 'AND' between reg1 and reg2. The result is stored in reg2	and b r1,r2 and w r1,r2
13	or b/w x(data),reg	Logical	Logical 'OR' between data and reg. The result is stored in reg	or b x15,r1 or w x150,r1

14	or b/w reg1,reg2	Logical	Logical 'OR' between reg1 and reg2. The result is stored in reg2	or b r1,r2 or w r1,r2
15	xor b/w x(data),reg	Logical	Logical 'XOR' between data and reg. The result is stored in reg	xor b x15,r1 xor w x150,r1
16	xor b/w reg1,reg2	Logical	Logical 'XOR' between reg1 and reg2. The result is stored in reg2	xor b r1,r2 xor w r1,r2
17	not b reg	Logical	Logical 'NOT' of specified reg	not b r1
18	comp b/w x(data),reg	Logical	Comparison between data and reg.	comp b x15,r1 comp w x150,r1
19	comp b/w reg1,reg2	Logical	Comparison between reg1 and reg2.	comp b r1,r2 comp w r1,r2
20	jge 'label'	Branching	After executing 'comp' instruction, the execution will jump to 'label' if (data) >= reg (OR) reg1 >= reg2	jge L1
21	jle 'label'	Branching	After executing 'comp' instruction, the execution will jump to 'label' if (data) <= reg (OR) reg1 <= reg2	jle L1
22	jeq 'label'	Branching	After executing 'comp' instruction, the execution will jump to 'label' if (data) = reg (OR) reg1 = reg2	jeq L1
23	jmp 'label'	Branching	Jump to 'Label'	jmp L1
24	jgt 'label'	Branching	After executing 'comp' instruction, the execution will jump to 'label' if (data) > reg (OR) reg1 > reg2	jgt L1
25	jlt 'label'	Branching	After executing 'comp' instruction, the execution will jump to 'label' if (data) < reg (OR) reg1 < reg2	jlt L1
26	jne 'label'	Branching	After executing 'comp' instruction, the execution will jump to 'label' if (data) != reg (OR) reg1 != reg2	jne L1
27	add b/w x(data),reg	Arithmetic	Addition of data and reg. The result is stored in reg	add b x15,r1 add w x150,r1
28	add b/w reg1,reg2	Arithmetic	Addition of reg1 and reg2. The result is stored in reg2	add b r1,r2 add w r1,r2

29	sub b/w x(data),reg	Arithmetic	Subtraction. reg minus data. The result is stored in reg	sub b x15,r1 sub w x150,r1
30	sub b/w reg1,reg2	Arithmetic	Subtraction. reg2 minus reg1. The result is stored in reg2.	sub b r1,r2 sub w r1,r2
31	j(flag) 'label' jn(flag)'label'	Branching	After the arithmetic operation, if the specified (flag) is set or n(flag) not set then the execution will jump to 'label'	jc L1 jnc L1 jn L1 jnn L1 jp L1 jnp L1 jz L1 jnz L1
32	call 'label'	Branching	To execute subroutine, stored at specified 'label'.	call L1
33	ret	Branching	To terminate the subroutine. To go back to the main program	L1: mov b x03,r2 mov b x15,r3 ret
34	halt	Control	Halts the execution	halt
35	shift b/w r/l c/nc reg,count	Shifting	Shifts the bits within register; byte or word, in left or right direction, through carry or without carry. Process is repeated for 'count' times.	shift b l c r1,x01