Name:_____

Homework #5 Computer Organization Due: March 15, 2019 (F) by 3 PM

1) Translate the given MARIE assembly language into machine language.

Address	**Label**	**Assembly Language**	**Machine Lang.** (in hex)
100_{16}	IF,	LOAD X	
101_{16}		SUBT Y	
102_{16}		SKIPCOND 800	
103_{16}		JUMP THEN	
104_{16}		JUMP ELSE	
105_{16}	THEN,	LOAD X	
106_{16}		OUTPUT	
107_{16}		JUMP END_IF	
108_{16}	ELSE,	CLEAR	
109_{16}		STORE Y	
$10A_{16}$	END_IF,	LOAD Y	
$10B_{16}$		OUTPUT	
$10C_{16}$		HALT	
$10D_{16}$	X,	DEC 0	
$10E_{16}$	Y,	DEC 50	

2) Translate the above MARIE assembly language into high-level language "pseudo" code. (answer on right above)

3) Explain why the hardwired control unit is faster than a microprogrammed control unit?

4) Suppose we want to add a new Assembly Language instruction to MARIE called TRIPLE_AC with opcode F_{16}. The TRIPLE_AC instruction causes the AC's current value to be tripled. For example if the AC value was 5 before the instruction, then after the TRIPLE_AC instruction the AC value would be 15. Complete the RTN steps to Fetch-Decode-Execute the new TRIPLE_AC instruction.

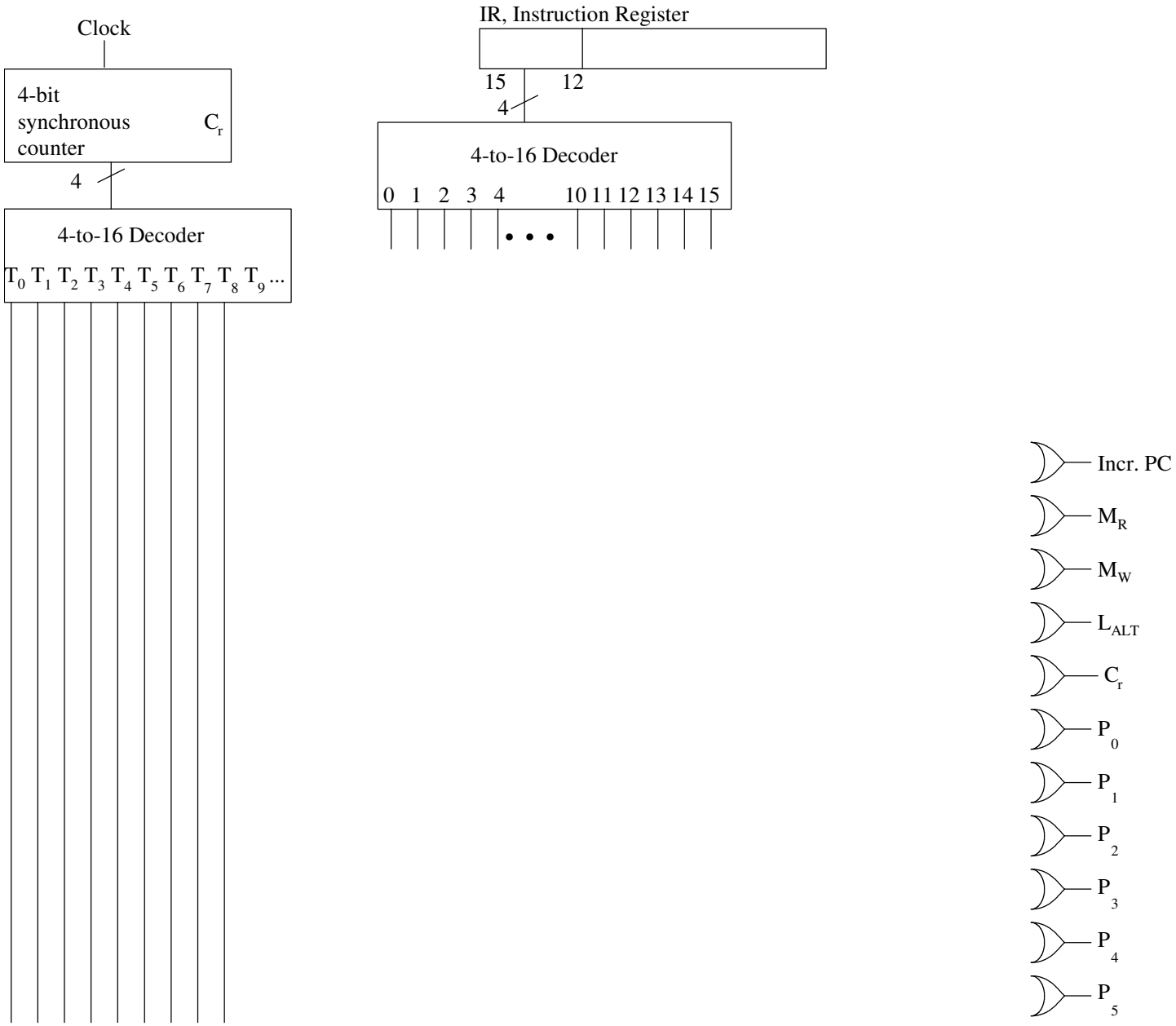
MAR ← PC
MBR ← M[MAR]
IR ← MBR
PC ← PC + 1

Name:_____

5) Which control signals should contain a "1" for each steps in the JUMPI instruction?

Step	RTN	Step #	P_5	P_4	P_3	P_2	P_1	P_0	C_r	Inc PC	M_R	L_{ALT}	M_W
Fetch	MAR ← PC	T_0											
	MBR ← M[MAR]	T_1											
	IR ← MBR	T_2											
Decode IR[15-12]	PC ← PC + 1	T_3											
Execute	MAR ← IR[11-0]	T_4											
	MBR ← M[MAR]	T_5											
	PC ← MBR	T_6											

6) Draw the partial combinational logic of the hardwired control unit to handle the JUMPI (opcode C_{16}) instruction.



Name:_____

7) Extend the partial microprogram below to include microoperations to decode and implement the execution of the instructions: CLEAR, LOADI, STORE. **(Fill in only the bolded boxes)**

Revised Figure 4.23 Partial Microprogram

Part of Cycle	RTN (of MicroOp1)	μAddr	MicroOp1	MicroOp2	Jump	Dest
Fetch	MAR ← PC	0	01010	00000	0	0
	MBR ← M[MAR]	1	01101	00000	0	0
	IR ← MBR	2	00110	00000	0	0
	PC ← PC + 1	3	10001	00000	0	0
Decode	If LOADI, Jump	4	11000	11010	1	
	If STOREI, Jump	5	11000	11100	1	
	If ADD, Jump	6	11000	00110	1	
("Jump Table")	If LOAD, Jump	7	11000	00010	1	
	If STORE, Jump	8	11000	00100	1	
	If SKIPCOND, Jump	9	11000	10000	1	
	If SUBT, Jump	10	11000	01000	1	
	If JUMP, Jump	11	11000	10010	1	
	If ADDI, Jump	12	11000	10110	1	
	If CLEAR, Jump	13	11000	10100	1	
	If JNS, Jump	14	11000	00000	1	
	If JUMPI, Jump	15	11000	11000	1	
	If INPUT, Jump	16	11000	01010	1	
	If OUTPUT, Jump	17	11000	01100	1	
	If HALT, Jump	18	11000	01110	1	0
Execute CLEAR	AC ← 0	19				
Execute LOADI	MAR ← IR[11-0]	20				
	MBR ← M[MAR]	21				
	MAR ← MBR	22				
	MBR ← M[MAR]	23				
	AC ← MBR	24				
Execute STORE	MAR ← IR[11-0]	25				
	MBR ← AC	26				
	M[MAR] ← MBR	27				
Execute ADD	. . .	28				