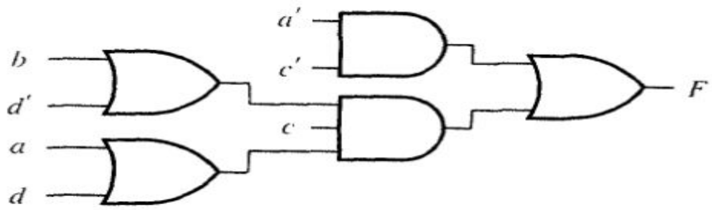
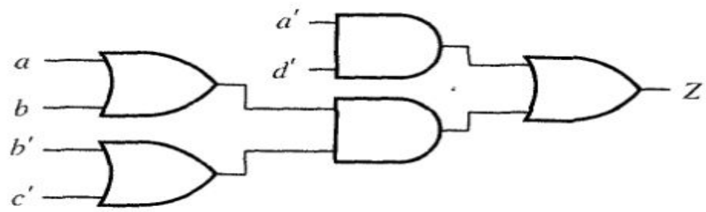
Tutorial 7

4 Find all of the static hazards in the following networks. For each hazard, specify the values of the variables which are constant and the variable which is changing. Indicate how all of these hazards could be eliminated by adding gates to the existing networks. (This means that you can add gates or gate inputs to a network as it stands, but you cannot change any of the connections in the given networks)



(a)



(b)

sol:

(a) 1-hazard: $a = 0$, $b = 1$, $d = 1$, c changing. To eliminate, add $a'bd$ term to the OR gate at the output.

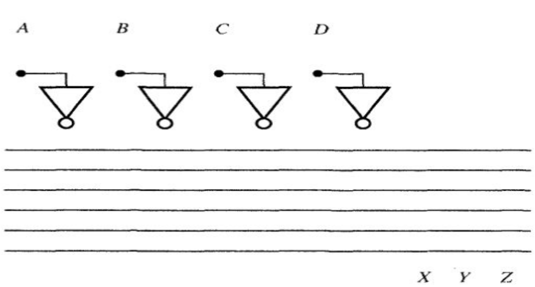
0-hazard: $a = 0$, $b = 0$, $c = 1$, d changing. To eliminate, add $(a + b + c')$ to AND gate which realizes $c(b + d')(a + d)$.

(a) 1-hazard: $a = 0$, $b = 1$, $d = 1$, c changing. Gate outputs—2, 4, 5, 3, 5 (for c 0 → 1) or 3, 5, 2, 4, 5 (for c 1 → 0)

6)The PLA below will be used to implement the following equations:

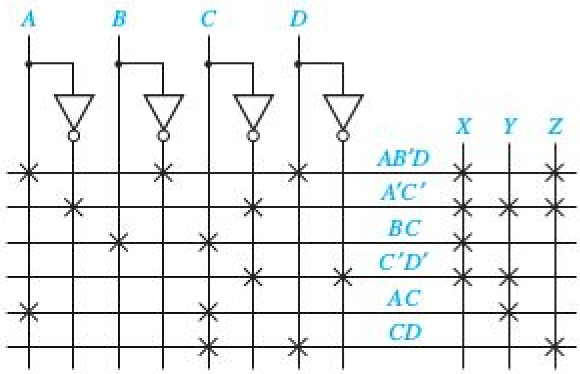
X = AB’D + A'C’ + BC + C’D' Z = CD + A'C' + AD + AB'D

Y = A'C' + AC + C'D''



(a)Indicate the connections that will be made to program the PLA to implement these equations.

sol:



(b) Specify the truth table for a ROM which realizes these same equations.

sol:

(b) Truth Table for the ROM

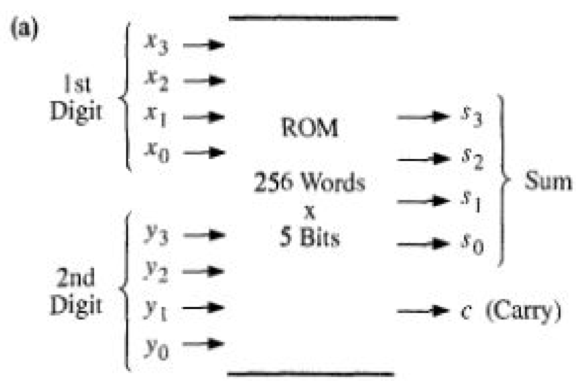
A	B	C	D	X	Y	Z
0	0	0	0	1	1	1
0	0	0	1	1	1	1
0	0	1	0	0	0	0
0	0	1	1	0	0	1
0	1	0	0	1	1	1
0	1	0	1	1	1	1
0	1	1	0	1	0	0
0	1	1	1	1	0	1
1	0	0	0	1	1	0
1	0	0	1	1	0	1
1	0	1	0	0	1	0
1	0	1	1	1	1	1
1	1	0	0	1	1	0
1	1	0	1	0	0	0
1	1	1	0	1	1	0
1	1	1	1	1	1	1

7)(a) An adder for Gray-coded decimal digits (see Table 1-1) is to be designed using a ROM. The adder should add two Gray-coded digits and give the Gray-coded sum and a carry. For example, 1011 + 1010 = 0010 with a carry of 1 (7 + 6 = 13). Draw a block diagram showing the required ROM inputs and outputs. What size ROM is required? Indicate how the truth table for the ROM would be specified by giving some typical rows.

(b) If the same adder were implemented using a PLA, what size PLA would be required? (Assume that only the 10 legal gray-coded digits can occur as inputs.)

sol:

(a)



$x_3x_2x_1x_0y_3y_2y_1y_0$	$s_3s_2s_1s_0c$
0 0 0 0 0 0 0 0	0 0 0 0 0
0 0 0 0 0 0 0 1	0 0 0 1 0
. . . .	. . . . .
0 1 0 1 0 0 1 1	1 0 0 0 0
0 1 0 1 0 1 0 0	1 0 0 1 0
. . . .	. . . . .
1 0 0 1 1 0 0 0	0 1 1 1 1
1 0 0 1 1 0 0 1	1 0 0 0 1
1 0 0 1 1 0 1 0	X X X X X
1 0 0 1 1 0 1 1	X X X X X
. . . .	. . . . .

(b) 8 inputs, 100 words, 5 outputs

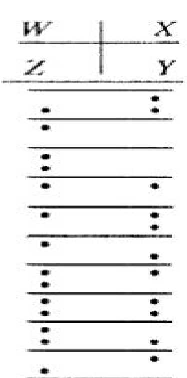
8) Braille is a system which allows a blind person to "read" alphanumerics by feeling a pattern of raised dots. Design a network that converts BCD to Braille. The table shows the correspondence between BCD and Braille.

(a) Use a multiple-output NAND-gate network.

(b) Use a PLA. Give the PLA table.

(c) Use a PAL. Specify the PAL type and fuse pattern.

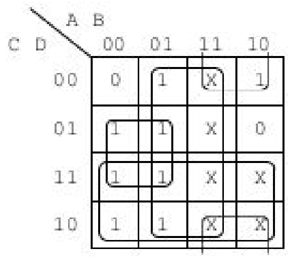
A	B	C	D
0	0	0	0
0	0	0	1
0	0	1	0
0	0	1	1
0	1	0	0
0	1	0	1
0	1	1	0
0	1	1	1
1	0	0	0
1	0	0	1



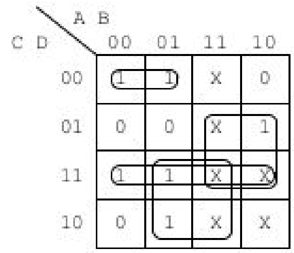
Sol:

(a)

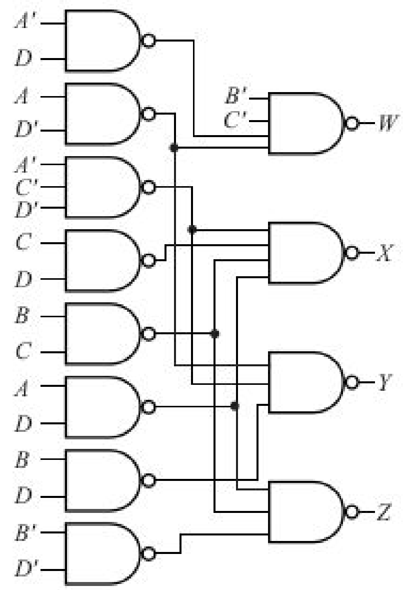
$A B C D$	$W X Y Z$
0 0 0 0	0 1 1 1
0 0 0 1	1 0 0 0
0 0 1 0	1 0 0 1
0 0 1 1	1 1 0 0
0 1 0 0	1 1 1 0
0 1 0 1	1 0 1 0
0 1 1 0	1 1 0 1
0 1 1 1	1 1 1 1
1 0 0 0	1 0 1 1
1 0 0 1	0 1 0 1

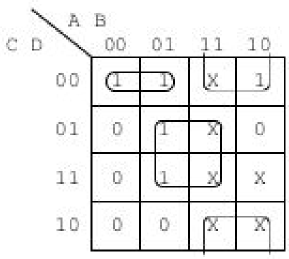


$W = A'D + C + B + \underline{A D'}$

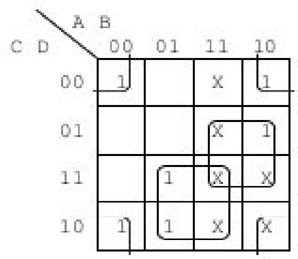


$X = \underline{A'C'D'} + C D + \underline{A D} + \underline{B C}$





$Y = \underline{A D'} + B D + \underline{A'C'D'}$



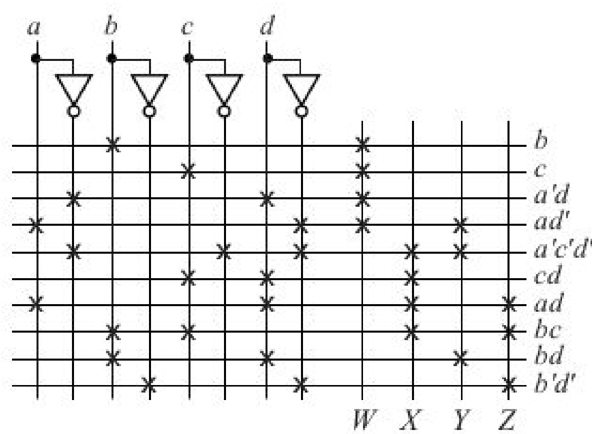
$Z = \underline{A D} + \underline{B C} + B'D'$

Alt: $Z = A + \underline{B C} + B'D'$

(b)

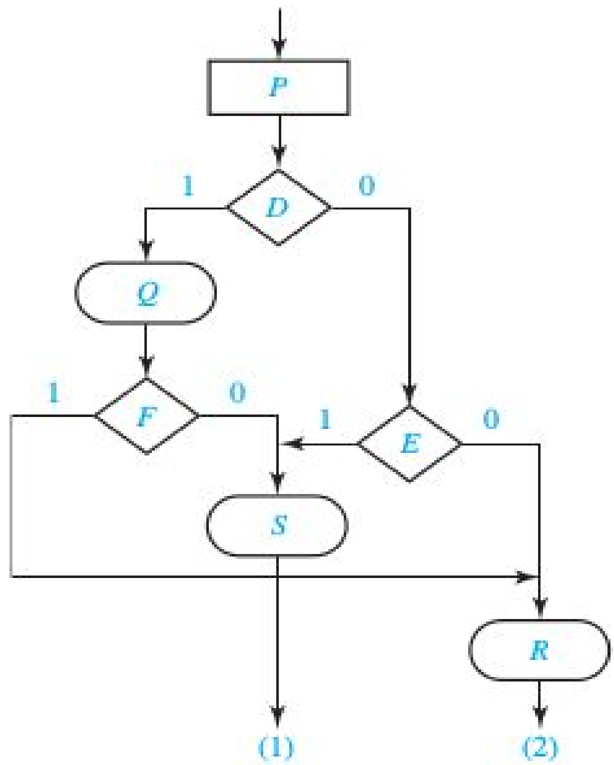
$a\ b\ c\ d$	$W X Y Z$
- 1 - -	1 0 0 0
- - 1 -	1 0 0 0
0 - - 1	1 0 0 0
1 - - 0	1 0 1 0
0 - 0 0	0 1 1 0
- - 1 1	0 1 0 0
1 - - 1	0 1 0 1
- 1 1 -	0 1 0 1
- 1 - 1	0 0 1 0
- 0 - 0	0 0 0 1

9.26 (c)

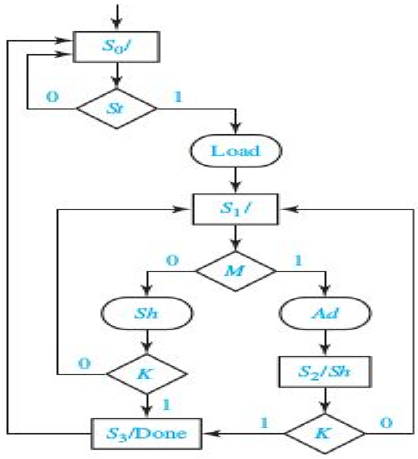


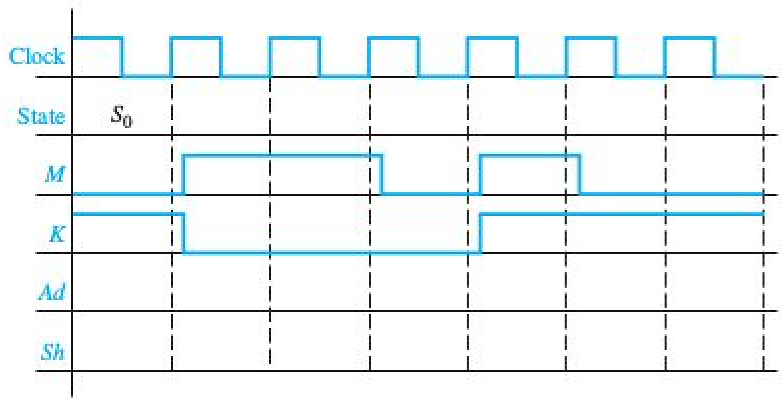
9) Construct an ASM block that has three input variables (D, E, F), four output variables (P, Q, R, S), and two exit paths. For this block, output P is always 1, and Q is 1 iff D 1. If D and F are 1 or if D and E are 0, R 1 and exit path 2 is taken. If (D 0 and E 1) or (D 1 and F 0), S 1 and exit path 1 is taken.

sol:



10) Complete the following timing diagram for the ASM chart given below. Assume St 1.





Sol: