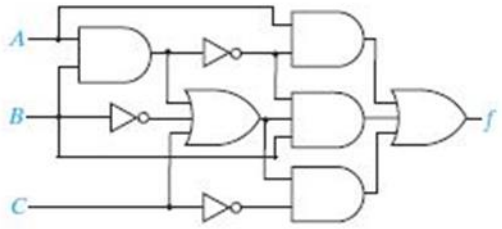
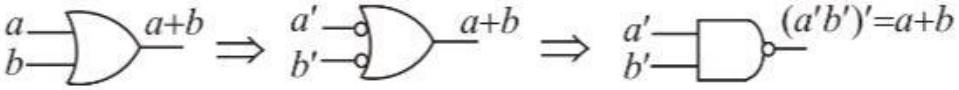
CDA Computer Logic Design Fall 2016
Homework 3 Solutions

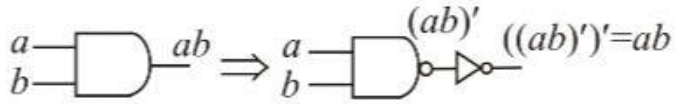
1) Implement the below logic circuit by using only NAND gates. (Do not simplify except to delete two inverters in series, if any) (15 points)



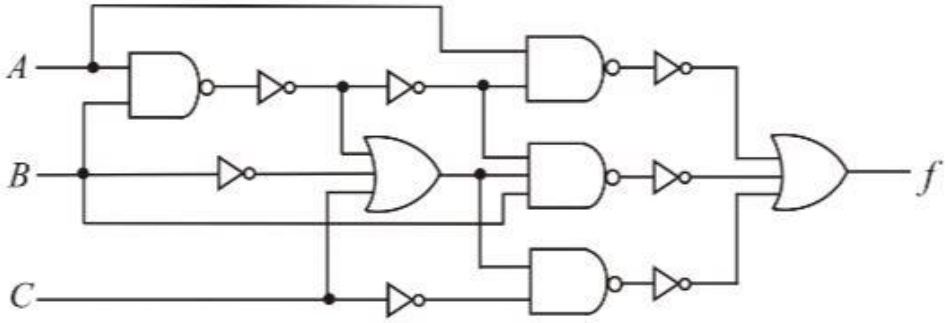
A)

Using the gate equivalent to convert the given circuit into a four-level circuit containing only NAND gates. Convert all AND gates into NAND gates by adding an inversion bubbles at the outputs. Convert all OR gates into NAND gates by adding inversion bubbles at the inputs.

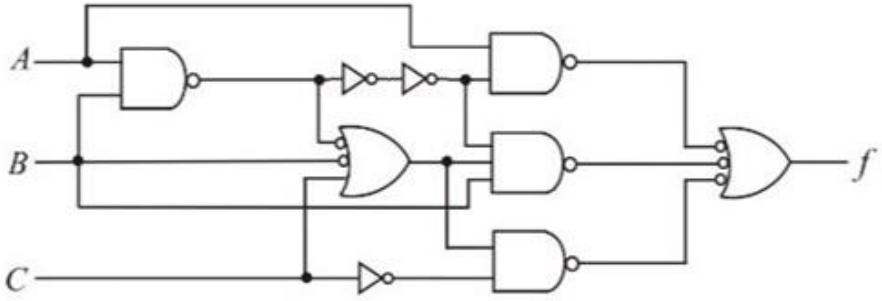




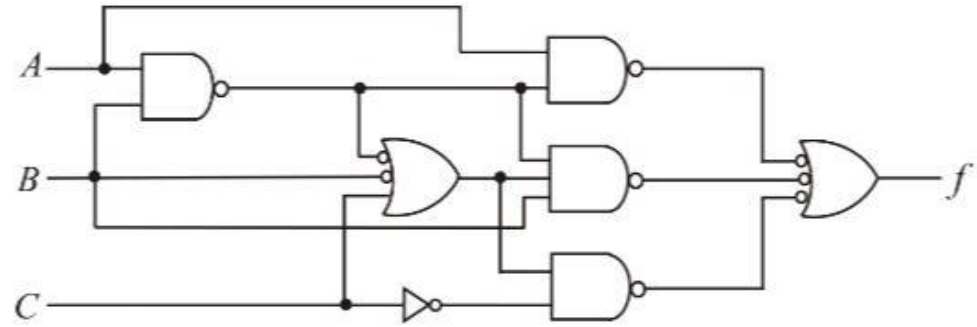
Thus, the AND gates are replaced by equivalent NAND gates in the given four-level circuit.
Then the circuit is redrawn as

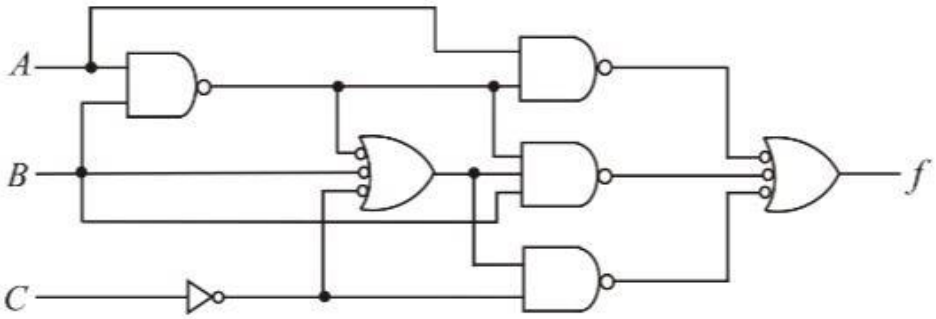


the above circuit is redrawn as

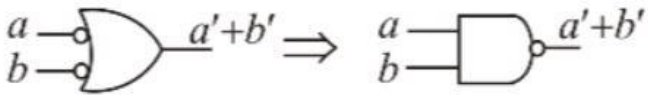


Modified circuit diagram is as follows:

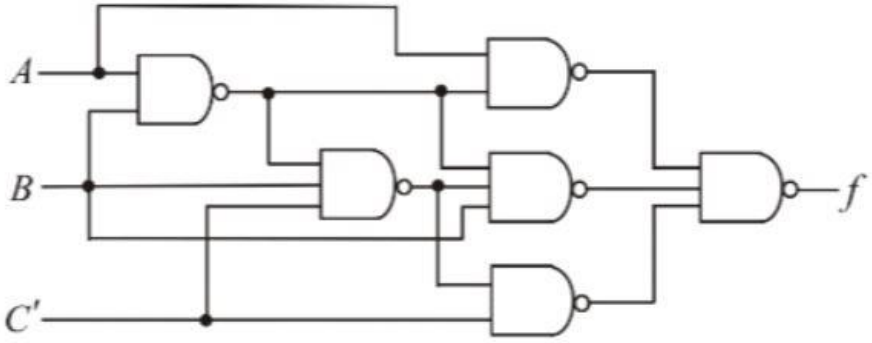




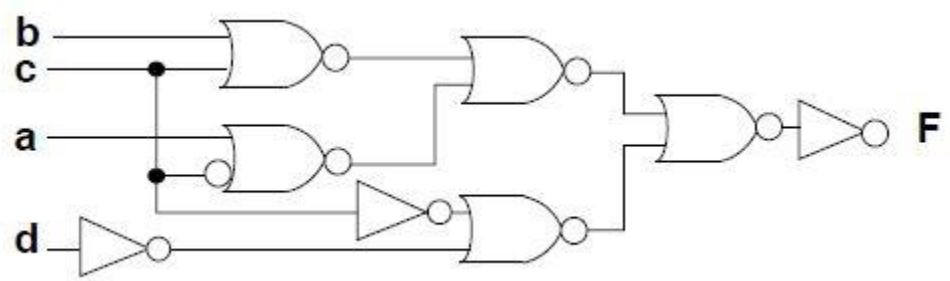
As we know that



Then, the bubbled OR gate is replaced with an NAND gate.
Simplified circuit diagram is as follows:



2) a) Find the hazards in the below circuit (5 points)

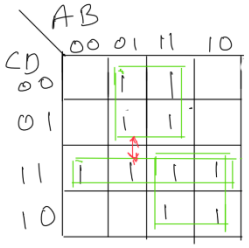


A) a) $F = \overline{\overline{b+c} + \overline{a+\overline{c}}} + \overline{\overline{c}+\overline{d}}$

$= (b + c)(a + \overline{c}) + cd$

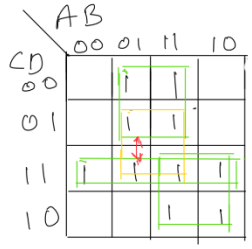
$= ab + ac + b\overline{c} + cd$

KMAP



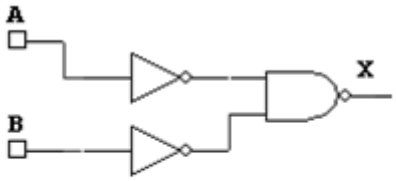
The above kmap displays the hazard

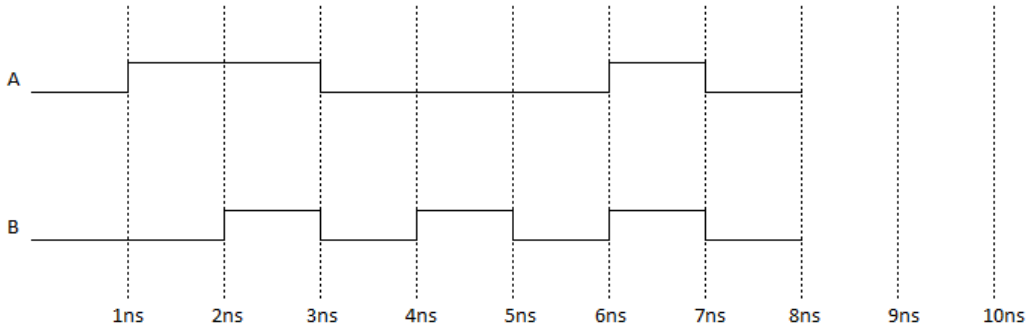
b) Modify the circuit so that it is hazard free (10 points)



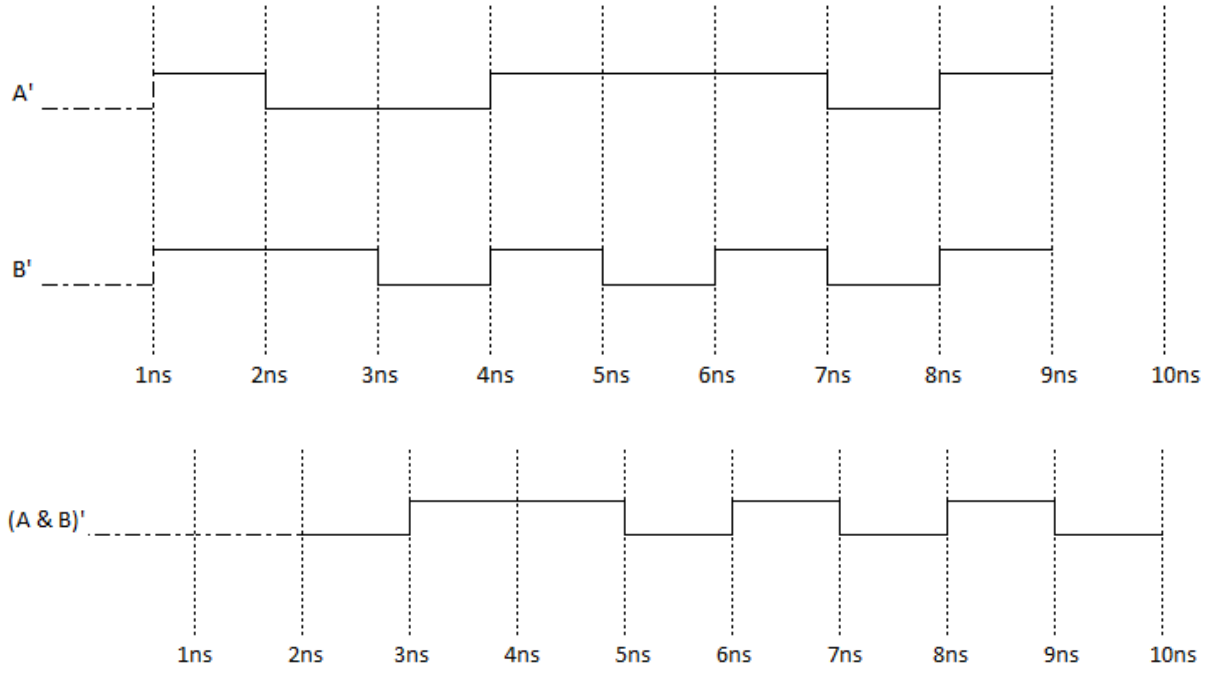
Adding the term ‘bd’ to the above circuit, removes hazards.

3) Complete the timing diagram for the given circuit. Assume that all gates have a propagation delay of 1 ns (10 points)





A)



Note:

The timing diagram for A' and B' starts with a delay of 1ns due to the inverter delay. All transitions are shifted right by 1ns. There is no signal from the inverters for 1 ns in the beginning represented by the dotted line. Similarly the final output starts with a delay of 2ns due to the combined delay of inverter and NAND gate. All transitions are shifted right by 2ns. There is no signal from the NAND gate for 2ns in the beginning represented by the dotted line.

4) Implement a full adder
(a) using two 8-to-1 MUXes. Connect X,Y, and Cin to the control inputs of the MUXes and connect 1 or 0 to each data input. (5 points)
(b) using two 4-to-1 MUXes and one inverter. Connect X and Y to the control inputs of the MUXes, and connect 1's, 0's, Cin, or C'in to each data input. (5 points)
(c) again using two 4-to-1 MUXes, but this time connect Cin and Y to the control inputs of the MUXes, and connect 1's, 0's, X, or X' to each data input. Note that in this fashion, any N-variable logic function may be implemented using a 2(N−1)-to-1 MUX. (5 points)

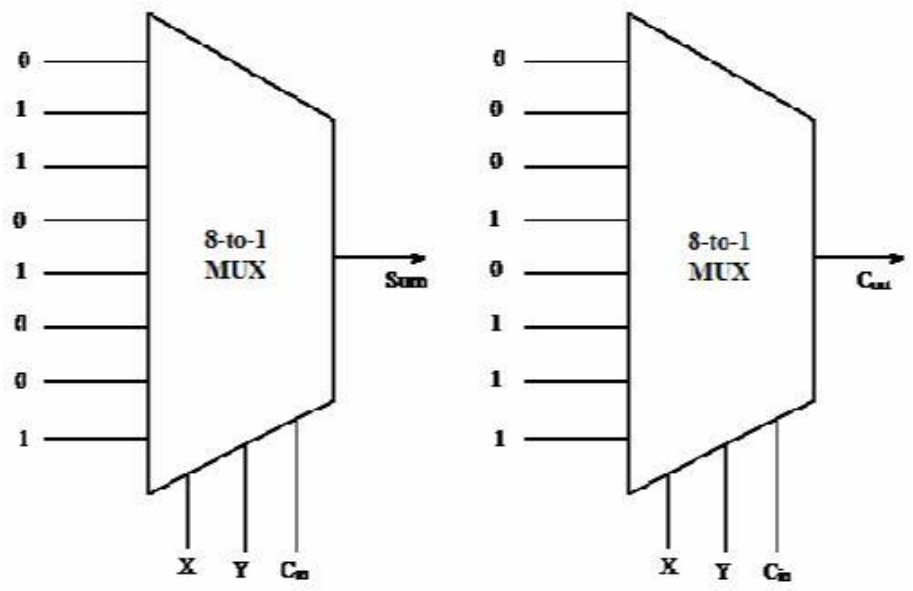
(a)

Let X, Y, C_{in} be the inputs to full adder and Sum, C_{out} outputs.

Truth-table:

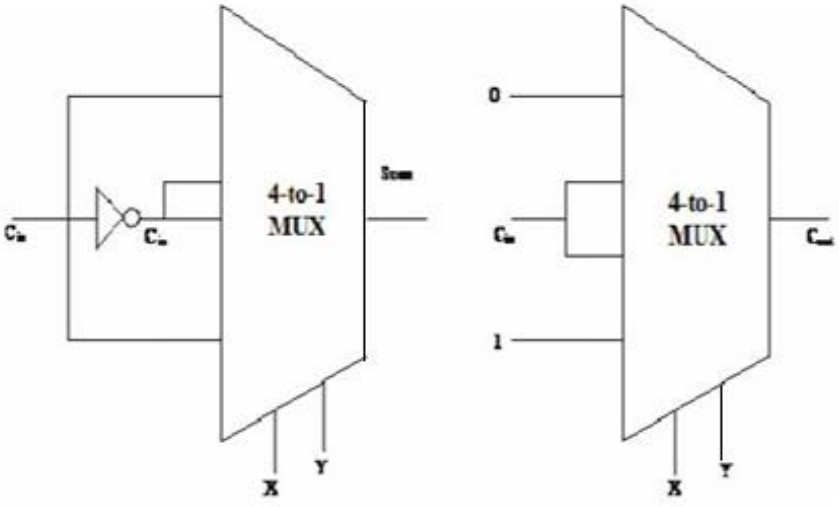
Inputs			Outputs	
X	Y	C_{in}	Sum	C_{out}
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

Implementing Sum, C_{out} using 8-to-1 MUX's:



(b)

Implementing using 4-to-1 MUX's:



When $X = 0, Y = 0$, then Sum depends on C_{in}.

When $X = 0, Y = 1$, then Sum depends on C_{in}'.

When $X = 1, Y = 0$, then Sum depends on C_{in}'.

When $X = 1, Y = 1$, then Sum depends on C_{in}.

When $X = 0, Y = 0$, no carry occurs whether C_{in} is 0 or 1.

When $X = 0, Y = 1$, then C_{out} depends on C_{in}.

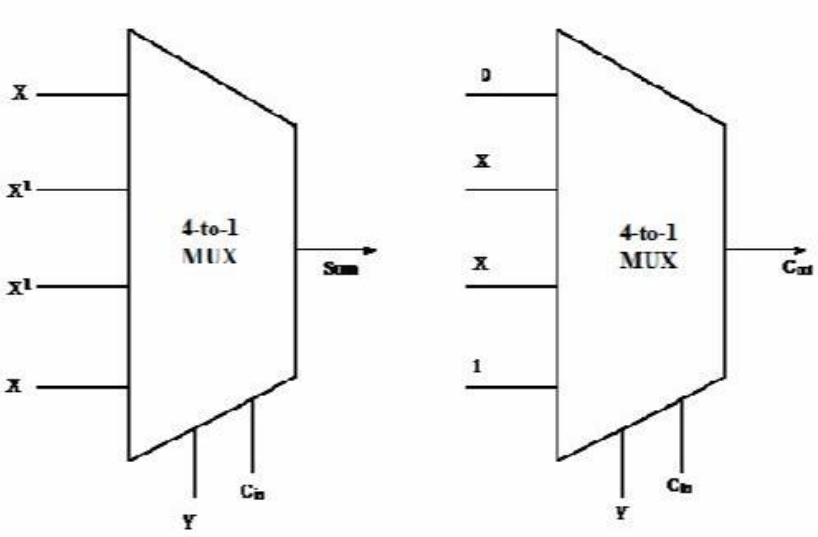
When $X = 1, Y = 0$, then C_{out} depends on C_{in}.

When $X = 1, Y = 1$, then C_{out} is 1 independent of C_{in} is 0 or 1.

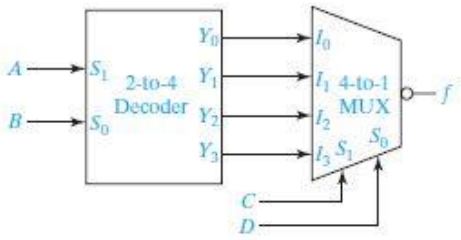
(c)

In this case C_{in} and Y are the control inputs.

Implementing using 4-to-1 MUX's:



5) The circuit below has a 2-to-4 decoder with active high outputs connected to a 4-to-1 MUX with an active low output. Derive a minimum SOP or a minimum POS expression for the output, f(A, B, C, D). (15 points)



Consider the following truth table for the 2-to-4 decoder with active high outputs:
Table 1: Truth table for 2-to-4 decoder

Input		Output			
A	*B*	Y_0	Y_1	Y_2	Y_3
0	0	1	0	0	0
0	1	0	1	0	0
1	0	0	0	1	0
1	1	0	0	0	1

Express each output values Y_0, Y_1, Y_2 and Y_3 from the truth table illustrated in Table 1 into their respective sum-of-products (SOP) logical expression.

$$\left.\begin{aligned} Y_0 &= A'B' \\ Y_1 &= A'B \\ Y_2 &= AB' \\ Y_3 &= AB \end{aligned}\right\} \quad \text{..... (1)}$$

Consider the following truth table for the 4-to-1 MUX circuit:

Table 2 Truth table for 4-to-1 MUX

Input		Output
C	*D*	*f*
0	0	$I_0' = Y_0'$
0	1	$I_1' = Y_1'$
1	0	$I_2' = Y_2'$
1	1	$I_3' = Y_3'$

Note :
The output *f* is active low as denoted by I_0', I_1', I_2' and I_3'

Express the output f from the truth table illustrated in Table 2 into its sum-of-products (SOP) logical expression.

$$f = C'D'I_0 + C'DI_1 + CD'I_2 + CDI_3$$

$$f = C'D'Y_0' + C'DY_1' + CD'Y_2' + CDY_3' \quad \text{..... (2)}$$

Substitute the expressions for Y_0, Y_1, Y_2 and Y_3 from equation (1) in equation (2).

$$\begin{aligned} f &= C'D'(A'B')' + C'D(A'B)' + CD'(AB')' + CD(AB)' \\ &= C'D'(A+B) + C'D(A+B') + CD'(A'+B) + CD(A'+B') \\ &= AC'D' + BC'D' + AC'D + B'C'D + A'CD' + BCD' + A'CD + B'CD \\ &= AC'D' + AC'D + B'C'D + B'CD + A'CD' + A'CD + BC'D' + BCD' \end{aligned}$$

Simplify further.

$$\begin{aligned} f &= AC'(D'+D) + B'D(C'+C) + A'C(D'+D) + BD'(C'+C) \\ &= AC' + A'C + BD' + B'D \end{aligned}$$

Therefore the minimum SOP expression for the output $f(A,B,C,D)$ is

$$\boxed{f(A,B,C,D) = AC' + A'C + BD' + B'D}.$$

6) Show how to make an 8-to-1 MUX using a PAL. Assume that PAL has 14 inputs and six outputs and assume that each output OR gate may have up to four AND terms as inputs, as in Figure 9-33. (Hint: Wire some outputs of the PAL around to the inputs, external to the PAL. Some PALs allow this inside the PAL to save inputs.) (15 points)

Implementing 8-to-1 MUX using PAL

8-to-1 MUX functional table

Inputs			Output
A	B	C	Z
0	0	0	I_0
0	0	1	I_1
0	1	0	I_2
0	1	1	I_3
1	0	0	I_4
1	0	1	I_5
1	1	0	I_6
1	1	1	I_7

$$Z = I_0A'B'C' + I_1A'B'C + I_2A'BC' + I_3A'BC + I_4AB'C' + I_5AB'C + I_6ABC' + I_7ABC$$

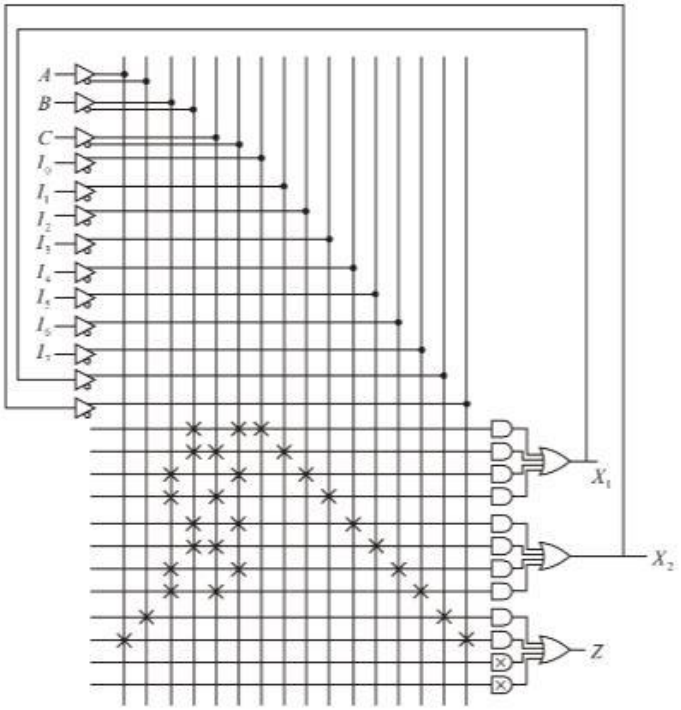
Directly implementing this in a PAL requres 11 inputs (*A, B, C, I_0, I_1, I_2, I_3, I_4, I_5, I_6, I_7*) and an output OR gate with 8 inputs. We only have OR gates with 4 inputs, hence rewrite as follows:

$$Z = X_1A' + X_2A$$

Where

$$X_1 = I_0B'C' + I_1B'C + I_2BC' + I_3BC$$

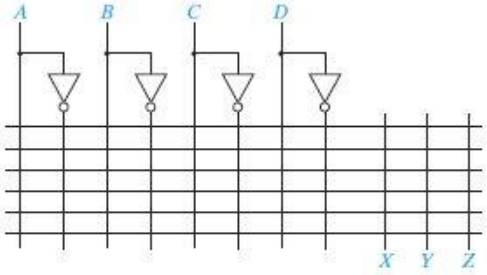
$$X_2 = I_4B'C' + I_5B'C + I_6BC' + I_7BC$$



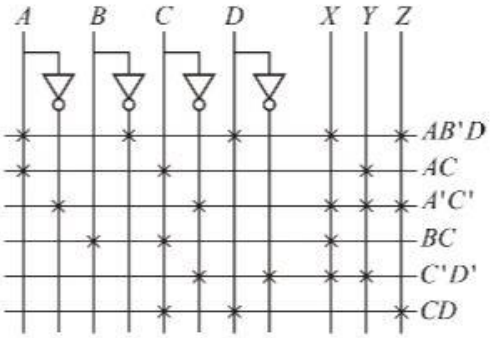
Note that this uses 13 inputs and 3 outputs. Which can be implemented within the available hardware

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

X = AB'D + A'C' + BC + C'D'
Y = A'C' + AC + C'D'
Z = CD + A'C' + AB'D
(a) Indicate the connections that will be made to program the PLA to implement these equations. (7 points)
(b) Specify the truth table for a ROM which realizes these same equations. (8 points)



(a)
Implementing PLA by the equations



(b)
Truth table for a ROM by using the equations.

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