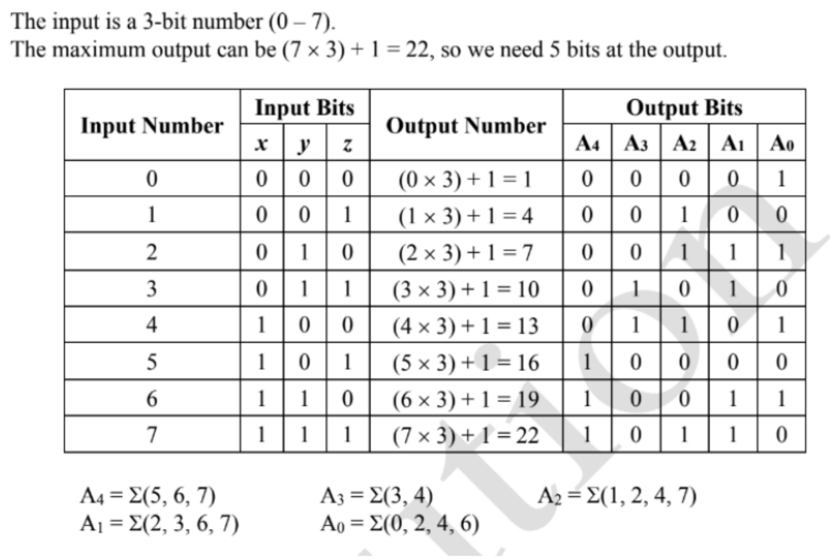
EE1005 – Digital Logic Design
Assignment – 2
Summer 2024

Instructor: Muhammad Adeel Tahir **Sections:** BSE-9A, BSE-9B

Question 1: Decoders

Outcomes: You must have knowledge of the following after this section for your exam point of view:
Use full-adders to implement a parallel binary adder
Explain the addition process in a parallel binary adder
Discuss the difference between a ripple carry adder and a look-ahead carry adder
State the advantage of look-ahead carry addition
Define carry generation and carry propagation and explain the difference
Develop look-ahead carry logic

a) Design a combinational circuit that takes 3-bit input and at the output it multiplies it by 3 and adds 1 to have the final output. Design this circuit using only 2 × 4 decoders and basic logic gates if necessary.
   i. Properly label and fill the truth table in neat and clean manner for this design.
   ii. Design the circuit diagram for this problem.



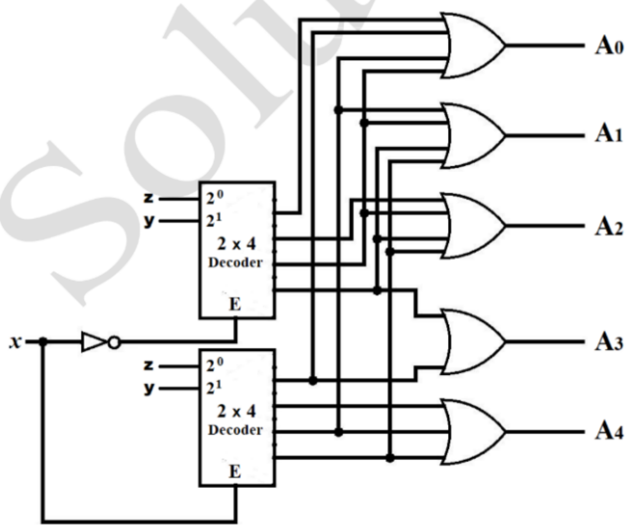
The input is a 3-bit number (0 – 7).
The maximum output can be (7 × 3) + 1 = 22, so we need 5 bits at the output.

Input Number	Input Bits			Output Number	Output Bits				
	x	y	z		A_4	A_3	A_2	A_1	A_0
0	0	0	0	$(0 \times 3) + 1 = 1$	0	0	0	0	1
1	0	0	1	$(1 \times 3) + 1 = 4$	0	0	1	0	0
2	0	1	0	$(2 \times 3) + 1 = 7$	0	0	1	1	1
3	0	1	1	$(3 \times 3) + 1 = 10$	0	1	0	1	0
4	1	0	0	$(4 \times 3) + 1 = 13$	0	1	1	0	1
5	1	0	1	$(5 \times 3) + 1 = 16$	1	0	0	0	0
6	1	1	0	$(6 \times 3) + 1 = 19$	1	0	0	1	1
7	1	1	1	$(7 \times 3) + 1 = 22$	1	0	1	1	0

$A_4 = \Sigma(5, 6, 7)$ $A_3 = \Sigma(3, 4)$ $A_2 = \Sigma(1, 2, 4, 7)$
$A_1 = \Sigma(2, 3, 6, 7)$ $A_0 = \Sigma(0, 2, 4, 6)$

$A_4 = \Sigma(5, 6, 7)$ $A_3 = \Sigma(3, 4)$ $A_2 = \Sigma(1, 2, 4, 7)$

$A_1 = \Sigma(2, 3, 6, 7)$ $A_0 = \Sigma(0, 2, 4, 6)$



b) Your task is to design an electronic circuit for a smart home security system called "Home Guardian." In this system, there are four surveillance cameras positioned around a house. Each camera can either detect motion (HIGH) or no motion (LOW) based on the activity in its field of view. The house is considered secure if at least three out of the four cameras do not detect motion.

Requirements:

Surveillance Cameras and Detection:

There are four surveillance cameras positioned around the house. Each camera can either detect motion (HIGH) or no motion (LOW) based on the activity.

Security Indicator: The system must include a "secure home indicator" that turns ON if the house is considered secure. For the purpose of this system, define "secure" as at least three out of the four cameras not detecting motion. If the house is not considered secure, the indicator should remain OFF, indicating that the house is at risk.

Circuit Design:

Use an 8 X 1 Multiplexer (MUX) to determine whether the house is secure based on the status of the cameras. You may use basic logic gates if necessary to assist in the design.

Determine how the outputs from the cameras will control the selection lines of the MUX to achieve the desired outcome.

Output Explanation:

Clearly explain how the MUX and any additional logic gates you use contribute to the final decision of turning the secure home indicator ON or OFF.

Implement the following:

A truth table that outlines how different combinations of camera statuses affect the secure home indicator.

A schematic diagram of the circuit, clearly labeling each input and output carefully.

A detailed explanation of how the circuit processes the camera statuses to control the secure home indicator. Note: Ensure your design is clear and well-documented, as you will need to explain how it works later.

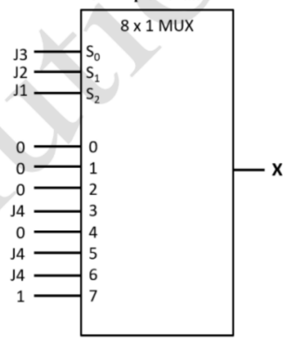
Solution:

Here we have 4 inputs and 1 output.
As we are using 8×1 Multiplexer, so there will be 3 selections lines and 8 data lines.
J1, J2, and J3 will act as selection lines.
We will divide the truth table in 8 equal parts and for each part we will represent S in terms of J4

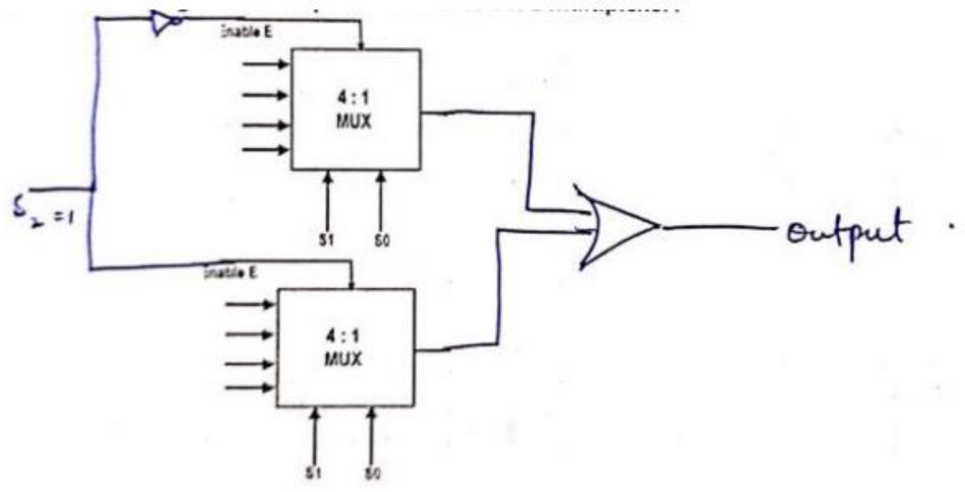
Truth Table

J1	J2	J3	J4	S	
0	0	0	0	0	S = 0
0	0	0	1	0	
0	0	1	0	0	S = 0
0	0	1	1	0	
0	1	0	0	0	S = 0
0	1	0	1	0	
0	1	1	0	0	S = J4
0	1	1	1	1	
1	0	0	0	0	S = 0
1	0	0	1	0	
1	0	1	0	0	S = J4
1	0	1	1	1	
1	1	0	0	0	S = J4
1	1	0	1	1	
1	1	1	0	1	S = 1
1	1	1	1	1	

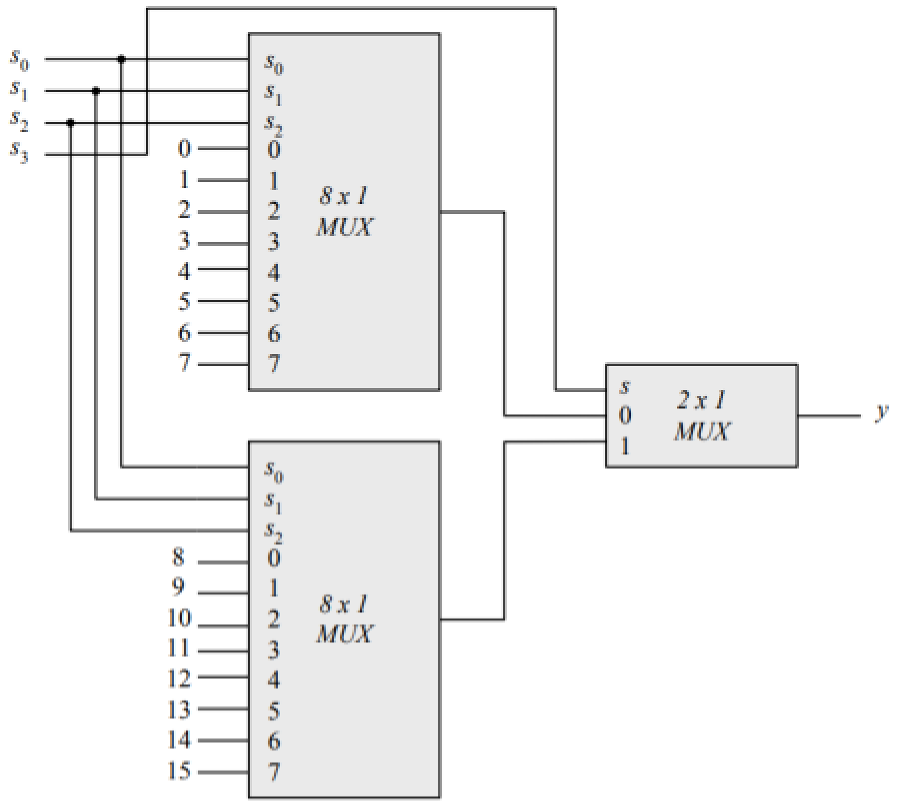
MUX Implementation



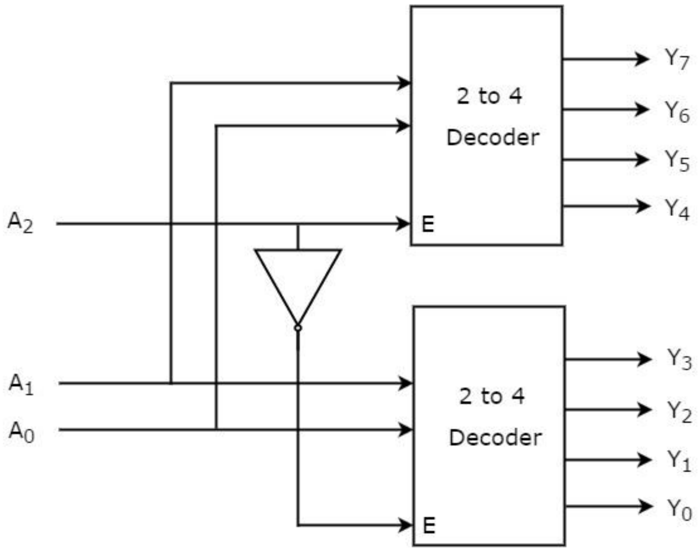
c) Use the following 4×1 MUX to create a 8×1 MUX. Label the diagram neatly on the paper.



d) Construct a 16 X I multiplexer with two 8 X I and one 2 X I multiplexers. Use block diagrams.



e) Construct 3:8 decoder using 2:4 decoders.



f) Implement a Full adder using **three** 2:4 decoders**. Your design must use 3 number of 2:4 decoders with enable input E, make sure you come up with a very feasible solution to**

implement it. To get yourself started first lay down the structure of first 2:4 decoder whose outputs are connected to the enable of 2:4 decoder of the other two decoders after implementing the truth table of the Full adder. (Hint: You will have to ground one of the inputs of decoder in order to make the selections work correctly). The outputs of the first decoder must be grounded which are not of use for us.

O	A	B	C	S	c
0	0	0	0	0	0
0	0	0	1	1	0
0	0	1	0	1	0
0	0	1	1	0	1
0	1	0	0	1	0
0	1	0	1	0	1
0	1	1	0	0	1
0	1	1	1	1	1

$$S = \Sigma(1,2,4,7)$$

$$C = \Sigma(3,5,6,7).$$

From the truth table of the Full adder, two observations can be made:

- The Sum (S) of a Full adder corresponds to the minterms: 1, 2, 4, 7.
- The Carry (C) of a Full adder corresponds to the minterms: 3, 5, 6, 7.

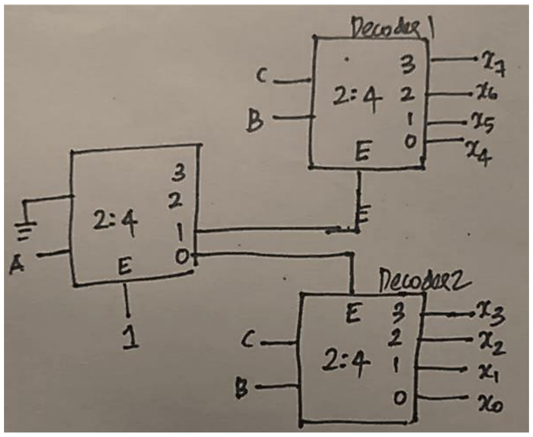
We will use the Sum of Products (SOP) form for simplicity.

Step 2: Implementation of a 3:8 decoder using 2:4 decoders

To implement a 3:8 decoder using 2:4 decoders, three 2:4 decoders are required. The implementation can be visualized in the diagram below:

1. Connect the most significant bit (MSB) A to the enable input (E) of the first 2:4 decoder.
2. When A is 0, the first 2:4 decoder enables the second 2:4 decoder.
3. When A is 1, the first 2:4 decoder enables the third 2:4 decoder.

This configuration allows us to effectively create a 3:8 decoder using three 2:4 decoders.



Step 3: Implementation of the Full adder using the 3:8 decoder

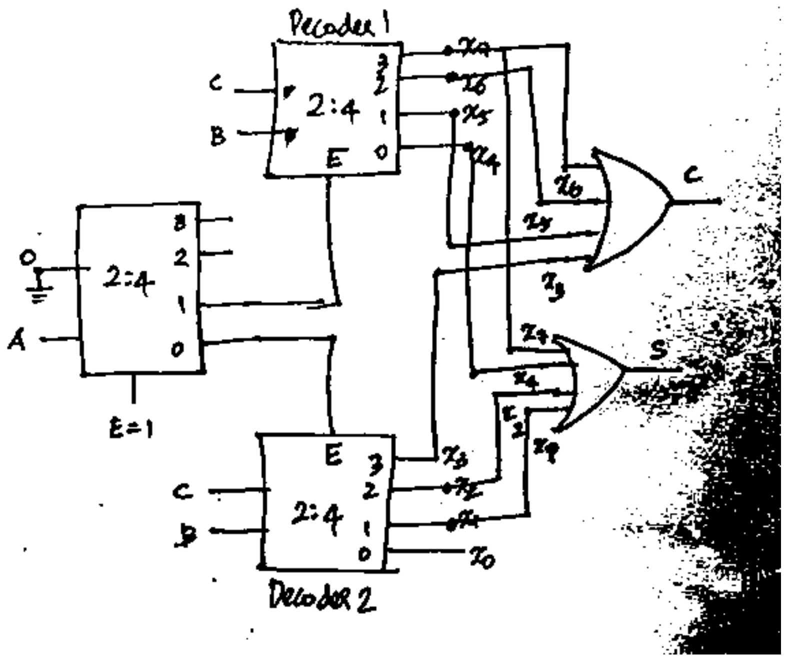
In this step, we will implement the SOP form for both the Sum and Carry outputs of the Full adder using the 3:8 decoder.

- For the Sum (S), connect the minterms 1, 2, 4, and 7 to the output.
- For the Carry (C), connect the minterms 3, 5, 6, and 7 to the output.

The diagram below shows the complete implementation of the Full adder using 2:4 decoders:

1. The first 2:4 decoder is used to determine which of the other two decoders will be enabled based on the MSB.
2. The second and third 2:4 decoders are used to generate the appropriate minterms for the Sum and Carry outputs.

This configuration provides a feasible and efficient solution to implement a Full adder using 2:4 decoders.



g) Implement the following using 4 × 1 MUX and external gates, connect A and B to the selection lines. The input requirements for the four data lines will be a function of variables C and D. These values are obtained by expressing F as a function of C and D for each of the four cases when AB= 00, 01,10,11. The functions may have to be implemented with external gates:

i. F (A,B,C,D) = ∑ (1,3,4,11,12,13,14,15)

ii. F (A,B,C,D) = ∑ (1,2,4,7,8,9,10,11)

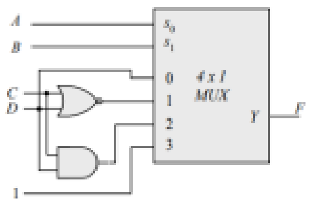
iii. Implement the part i using 8 × 1 MUX and externals gates if required, a truth table with neat and clean diagram is necessary

Solution:

Solution:

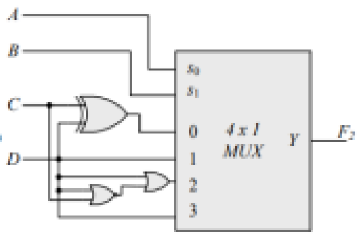
(a)

Inputs ABCD	F	
0000	0	
0001	1	$AB = 00$
0010	0	$F = D$
0011	1	
0100	1	$AB = 01$
0101	0	$F = C'D'$
0110	0	$= (C + D)'$
0111	0	
1000	0	
1001	0	$AB = 10$
1010	0	$F = CD$
1011	1	
1100	1	
1101	1	$AB = 11$
1110	1	$F = 1$
1111	1	



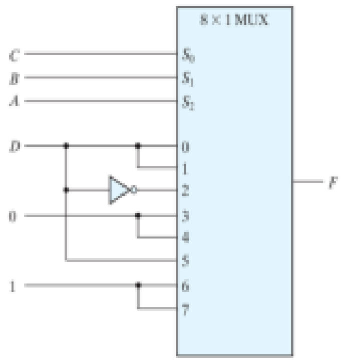
(b) F = S(1, 2, 5, 7, 8, 10, 11, 13, 15)

Inputs ABCD	$F_2 = \Sigma(1, 2, 5, 7, 8, 10, 11, 13, 15)$	
0000	0	
0001	1	$AB = 00$
0010	1	$F = C'D + CD'$
0011	0	
0100	0	$AB = 01$
0101	1	$F = C'D + CD = D$
0110	0	
0111	1	
1000	1	
1001	0	
1010	1	$AB = 10$
1011	1	$F = C'D' + C'D + CD = C'D' + D$
1100	0	$AB = 11$
1101	1	$F = D$
1110	0	
1111	1	



C)

A	B	C	D	F	
0	0	0	0	0	$F = D$
0	0	0	1	1	
0	0	1	0	0	$F = D$
0	0	1	1	1	
0	1	0	0	1	$F = D'$
0	1	0	1	0	
0	1	1	0	0	$F = 0$
0	1	1	1	0	
1	0	0	0	0	$F = 0$
1	0	0	1	0	
1	0	1	0	0	$F = D$
1	0	1	1	1	
1	1	0	0	1	$F = 1$
1	1	0	1	1	
1	1	1	0	1	$F = 1$
1	1	1	1	1	



h)

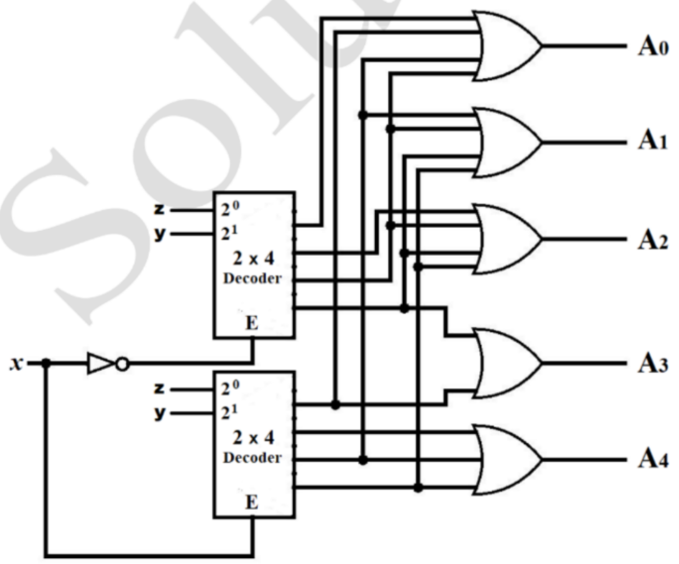
Part a:

The input is a 3-bit number (0 – 7).
The maximum output can be $(7 \times 3) + 1 = 22$, so we need 5 bits at the output.

Input Number	Input Bits			Output Number	Output Bits				
	x	y	z		A_4	A_3	A_2	A_1	A_0
0	0	0	0	$(0 \times 3) + 1 = 1$	0	0	0	0	1
1	0	0	1	$(1 \times 3) + 1 = 4$	0	0	1	0	0
2	0	1	0	$(2 \times 3) + 1 = 7$	0	0	1	1	1
3	0	1	1	$(3 \times 3) + 1 = 10$	0	1	0	1	0
4	1	0	0	$(4 \times 3) + 1 = 13$	0	1	1	0	1
5	1	0	1	$(5 \times 3) + 1 = 16$	1	0	0	0	0
6	1	1	0	$(6 \times 3) + 1 = 19$	1	0	0	1	1
7	1	1	1	$(7 \times 3) + 1 = 22$	1	0	1	1	0

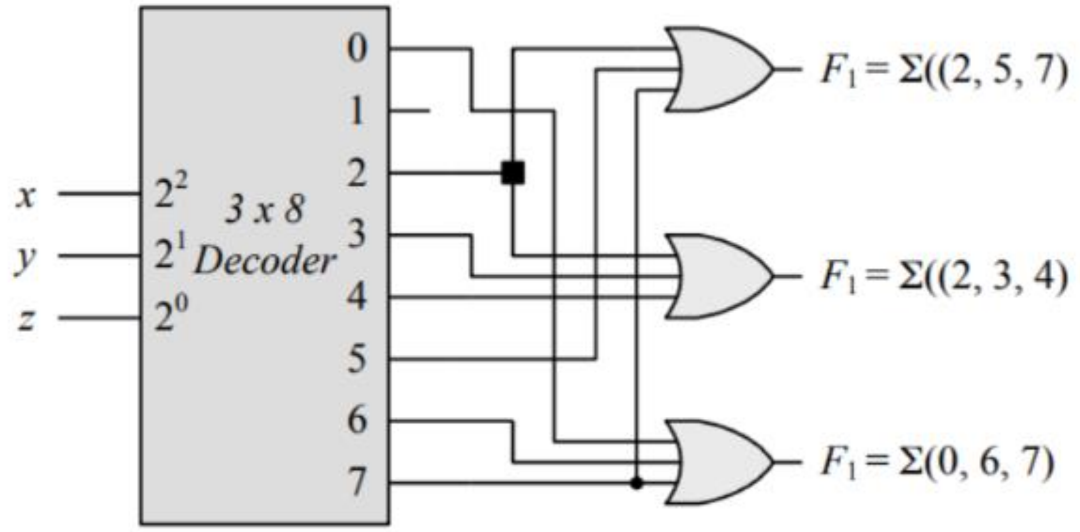
$A_4 = \Sigma(5, 6, 7)$ $A_3 = \Sigma(3, 4)$ $A_2 = \Sigma(1, 2, 4, 7)$
$A_1 = \Sigma(2, 3, 6, 7)$ $A_0 = \Sigma(0, 2, 4, 6)$

$A_4 = \Sigma(5, 6, 7)$ $A_3 = \Sigma(3, 4)$ $A_2 = \Sigma(1, 2, 4, 7)$
$A_1 = \Sigma(2, 3, 6, 7)$ $A_0 = \Sigma(0, 2, 4, 6)$

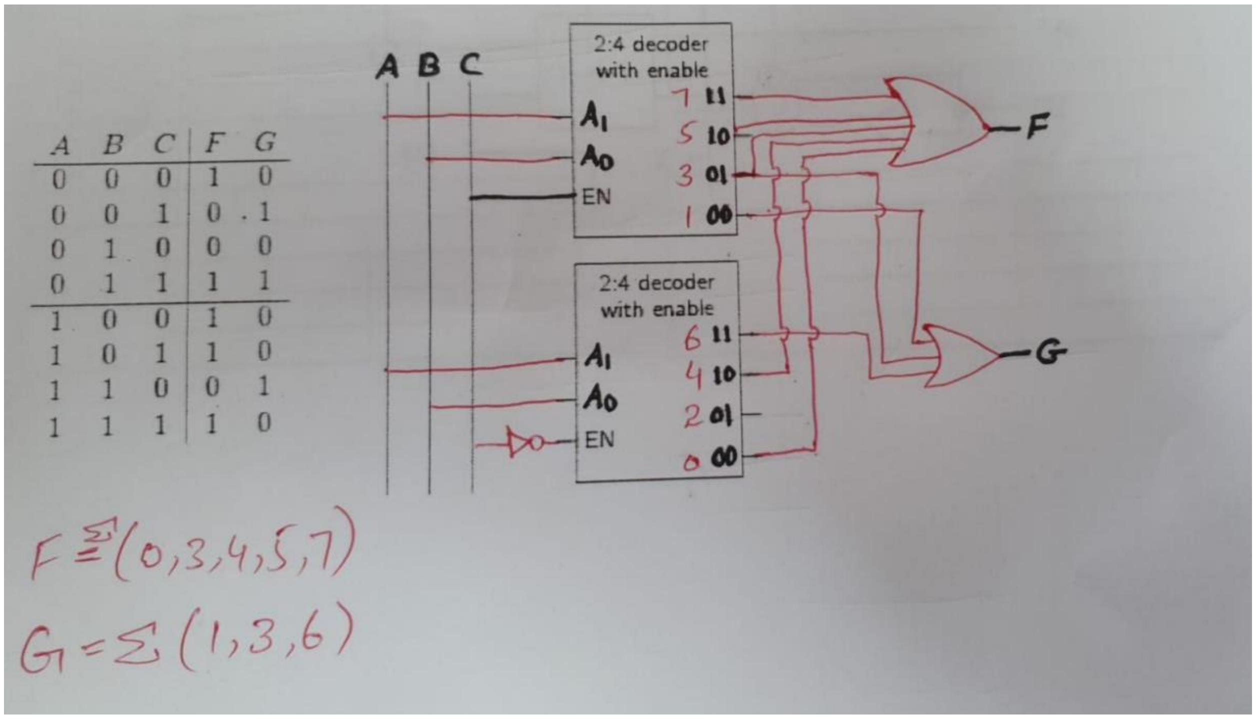


Part i:

$$F_1 = x(y + y')z + x'yz' = xyx + xy'z + x'yz' = \Sigma(2, 5, 7)$$
$$F_2 = xy'z' + x'y = xy'z' + x'yz + x'yz' = \Sigma(2, 3, 4)$$
$$F_3 = x'y'z' + xy(z + z') = x'y'z' + xyz + xyz' = \Sigma(0, 6, 7)$$



Part j:



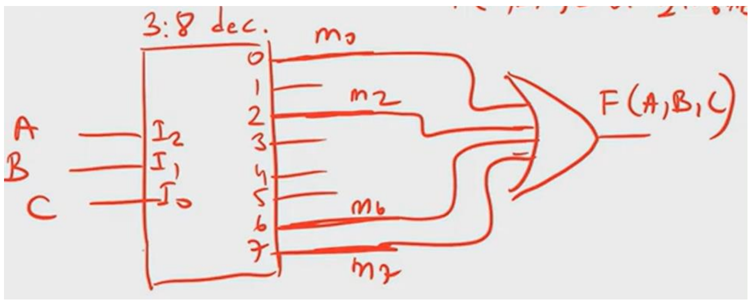
A	B	C	F	G
0	0	0	1	0
0	0	1	0	1
0	1	0	0	0
0	1	1	1	1
1	0	0	1	0
1	0	1	1	0
1	1	0	0	1
1	1	1	1	0

$F = \Sigma(0,3,4,5,7)$

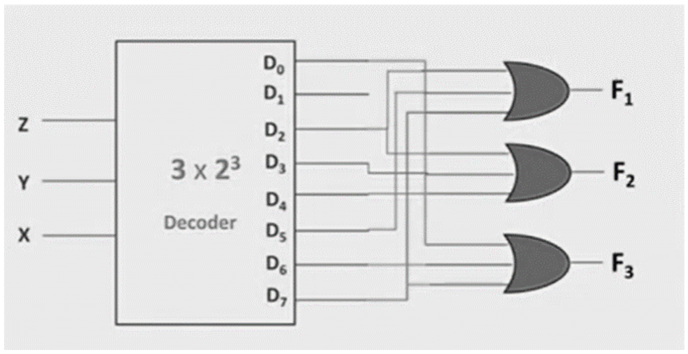
$G = \Sigma(1,3,6)$

Part k:

i)



ii)



Part l)

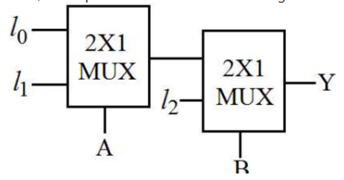
Given truth table is

A	B	Y
0	0	I_0
0	1	I_1
1	0	I_2
1	1	I_2

We know that the truth table for 2X1 MUX is

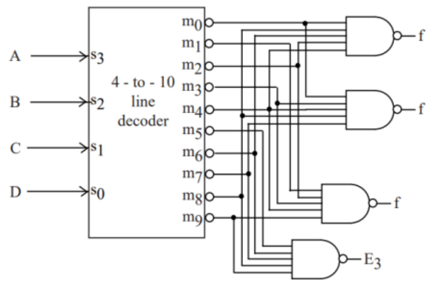
A	Y
0	S_0
1	S_1

Thus, the implementation of 3X1 MUX using 2X1 MUXs as



Part k)

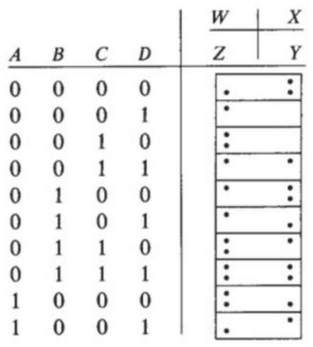
Since the decoder outputs are negative, NAND gates are required. The excess-3 outputs are Σ m(5,6,7,8,9), Σ m(1,2,3,4,9), Σ m(0,3,4,7,8), and Σ m(0,2,4,6,8) so four 5-input NAND gates are needed with inputs corresponding to the minterms of the excess-3 outputs.



Question 2: Number Systems, Operations and Codes

Outcomes: You must have knowledge of the following after this section for your exam point of view:
Use full-adders to implement a parallel binary adder
Explain the addition process in a parallel binary adder

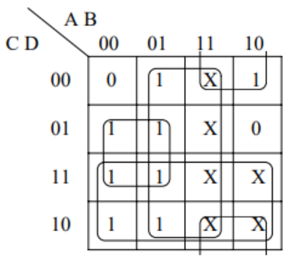
a) The following system named "Baads" has been designed to help a vision impaired person to read the letters by feeling the dots that are slightly raised. Design a circuit that converts BCD to this new system. The table shows the correspondence between BCD and Baads. Use a multiple-output NAND-gate circuit to design this problem. Truth table, k-map simplifications, equations and circuit diagram must be implemented in steps.



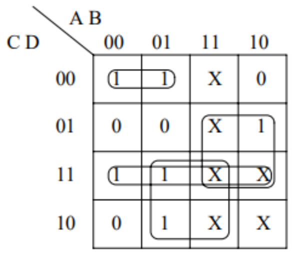
A	*B*	*C*	*D*	*W*	*X*	*Z*	*Y*
0	0	0	0		●	●	●
0	0	0	1	●			
0	0	1	0	●		●	
0	0	1	1	●	●		
0	1	0	0	●	●		●
0	1	0	1	●			●
0	1	1	0	●	●	●	
0	1	1	1	●	●	●	●
1	0	0	0	●		●	●
1	0	0	1		●	●	

Solution:

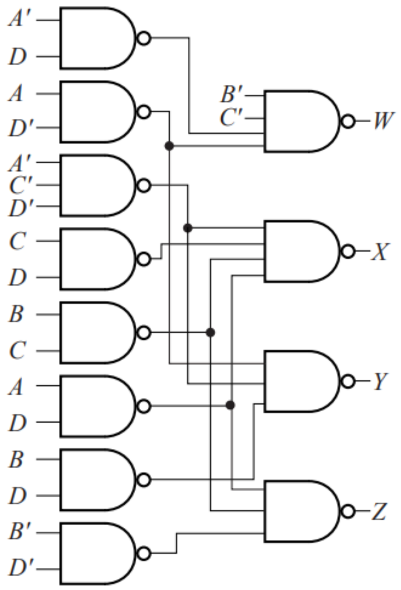
ABCD	WXYZ
0 0 0 0	0 1 1 1
0 0 0 1	1 0 0 0
0 0 1 0	1 0 0 1
0 0 1 1	1 1 0 0
0 1 0 0	1 1 1 0
0 1 0 1	1 0 1 0
0 1 1 0	1 1 0 1
0 1 1 1	1 1 1 1
1 0 0 0	1 0 1 1
1 0 0 1	0 1 0 1

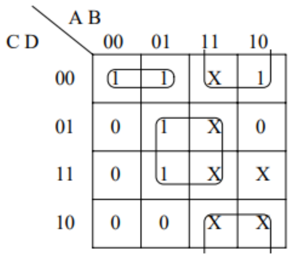


$W = A'D + C + B + AD'$

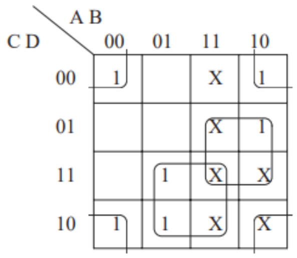


$X = A'C'D' + CD + AD + BC$





$Y = AD' + BD + A'C'D'$



$Z = AD + BC + B'D'$

Alt: $Z = A + BC + B'D'$

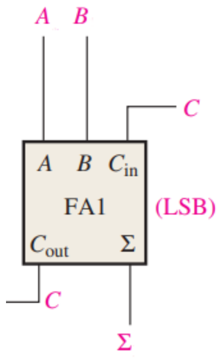
b) Solve each part carefully:

a. A diagram of 1-bit Full adder is given below, you are supposed to make the following parallel adders by utilizing as many 1-bit Full Adders necessary to complete the design. Label each adder's input and output carefully, deciding the LSB and MSB bits.

a) A 3-bit parallel adder

b) A 4-bit parallel adder

c) A 8-bit parallel adder



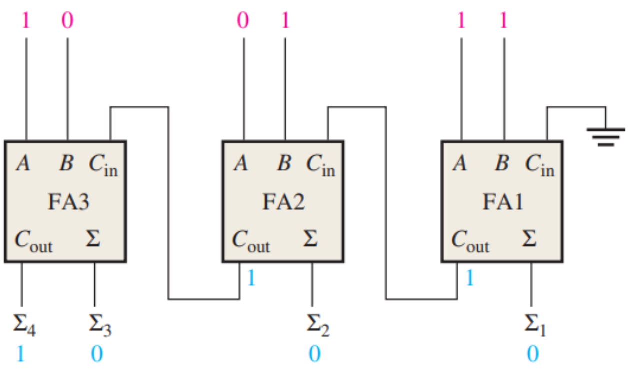
b. After designing the above adders, you are required to perform the following operations to confirm whether your adder is working correctly. For this purpose, we will assume an example: **5** = $(101)_2$ + **3** = $(011)_2$ = **8** (1000) is produced. When we apply the binary values as input to the **3-bit** parallel adder the output bits produced by it will indeed be 1000 where 1 will be C_{out} produced by the last adder at the MSB (FA_3). The following numbers are to be verified by the adders, you will choose a suitable adder that you have designed to perform the calculations at every step. **Note: You must draw the diagrams neatly labelling it carefully otherwise no marks shall be given. For every part, draw the adder again.**

i. **A**=$(127)_{10}$, **B**= $(23)_8$

ii. **A**= $(12)_{10}$, **B**= $(12)_{10}$

iii. **A** = $(13)_8$, **B** = $(12)_8$

Solution:



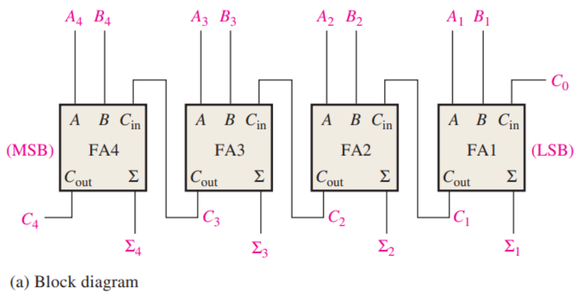
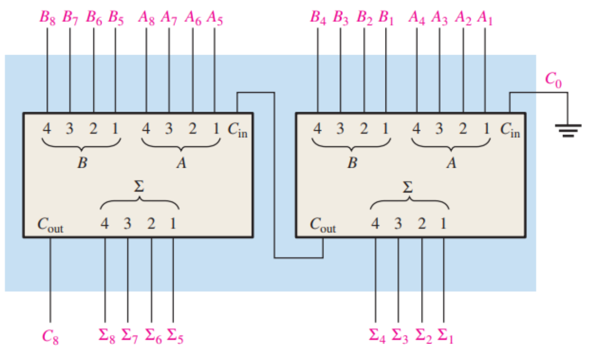
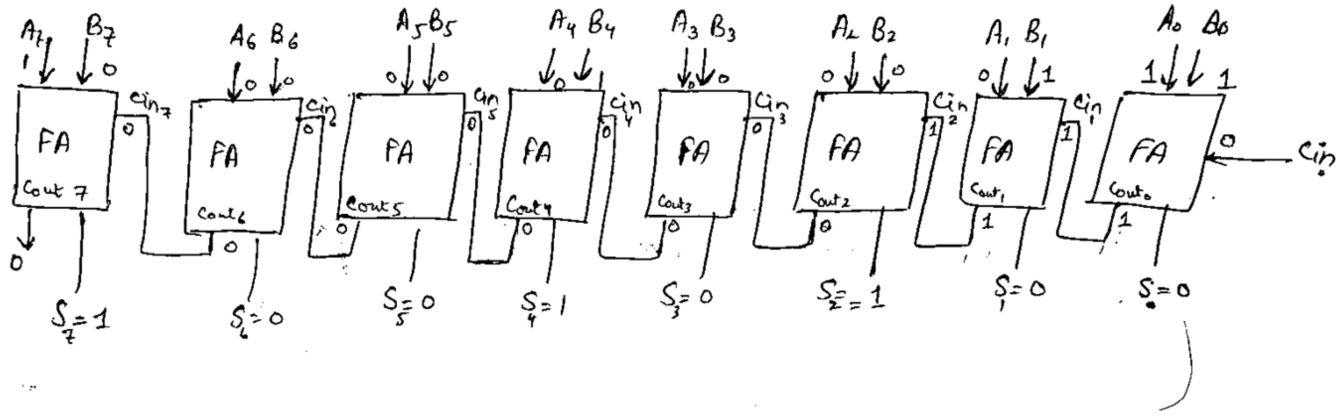


FIGURE 6–9 A 4-bit parallel adder.


B_8 B_7 B_6 B_5 A_8 A_7 A_6 A_5
B_4 B_3 B_2 B_1 A_4 A_3 A_2 A_1
C_0
4 3 2 1 4 3 2 1 C_{in}
B A
Σ
C_{out} 4 3 2 1
C_8
Σ_8 Σ_7 Σ_6 Σ_5
Σ_4 Σ_3 Σ_2 Σ_1

Cascading of two 4-bit adders to form an 8-bit adder.

One example is solved by me which was part a). All other verifications were to be done in same way for 4-bit and 3-bit adders.



$A = 127 = \overset{A_7A_6A_5A_4A_3A_2A_1A_0}{10000001}$ $B = 19 = \overset{B_7B_6B_5B_4B_3B_2B_1B_0}{00010011}$

0 0 0 0 0 0 0 0 2^8 $127 + 19 = 148 = 01001010\,00$

127 + 13 =

1. Apply bits one by one to FAdder from LSB
2. Then calculate some and send carry to next one.

c) Show how two 74HC283 adders can be connected to form an 8-bit parallel adder. Show output bits for the following 8-bit input numbers: A8A7A6A5A4A3A2A1 = 10111001 and B8B7B6B5B4B3B2B1 = 10011110. Draw the diagram neatly on your paper and label everything in a neat and clean manner.

Solution:

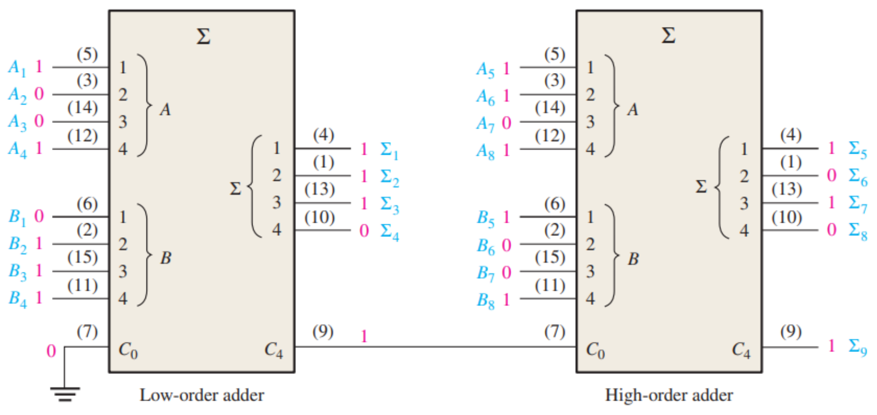
A8A7A6A5A4A3A2A1 = 10111001 and B8B7B6B5B4B3B2B1 = 10011110.

Two 74HC283 4-bit parallel adders are used to implement the 8-bit adder. The only connection between the two 74HC283s is the carry output (pin 9) of the low-order adder to the carry input (pin 7) of the high-order adder, as shown in Figure 6–12. Pin 7 of the low-order adder is grounded (no carry input). The sum of the two 8-bit numbers is = 101010111

Two 74HC283 4-bit parallel adders are used to implement the 8-bit adder. The only connection between the two 74HC283s is the carry output (pin 9) of the low-order adder to the carry input (pin 7) of the high-order adder, as shown in Figure 6–12. Pin 7 of the low-order adder is grounded (no carry input).

The sum of the two 8-bit numbers is

$$\Sigma_9\Sigma_8\Sigma_7\Sigma_6\Sigma_5\Sigma_4\Sigma_3\Sigma_2\Sigma_1 = 101010111$$

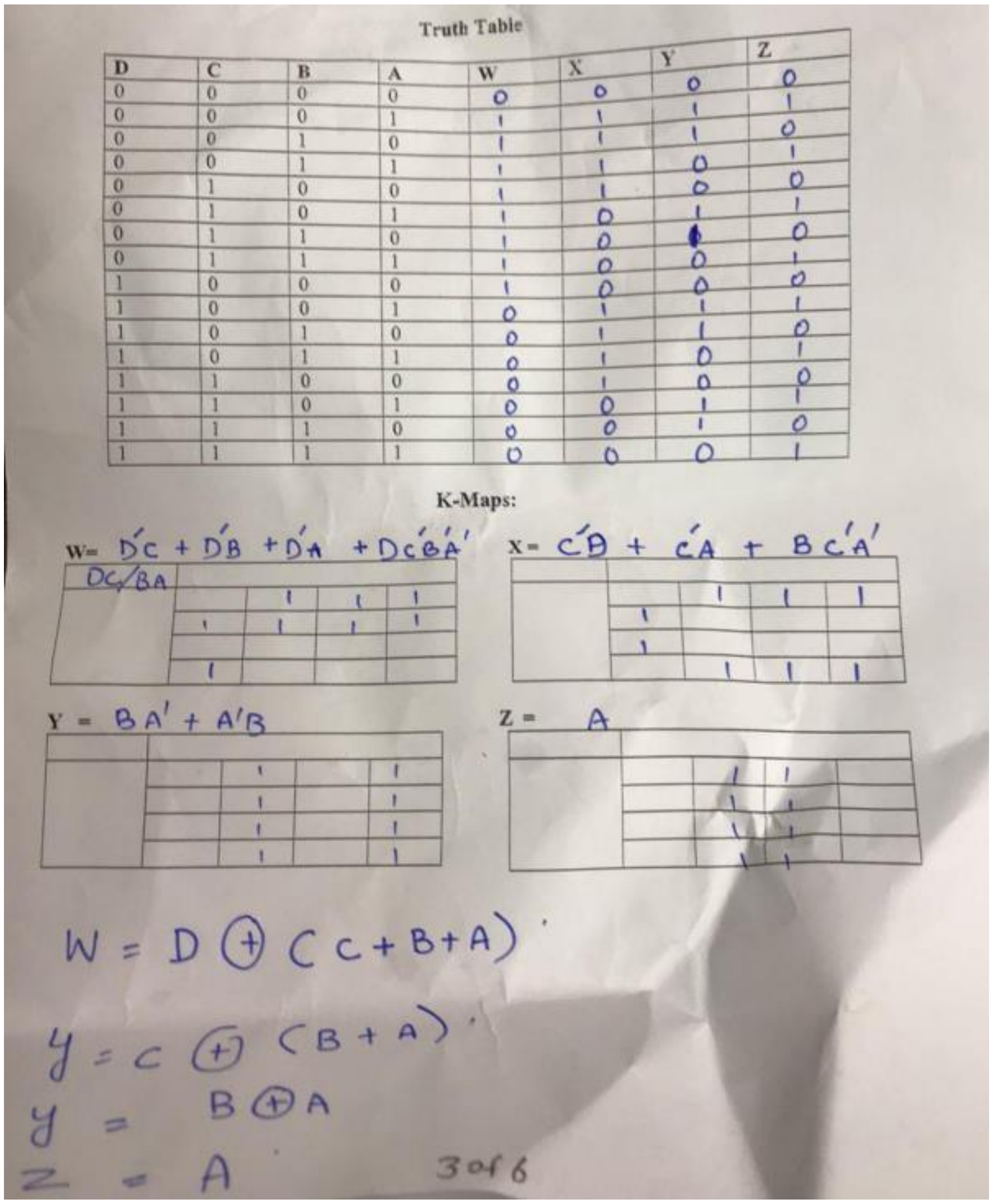


Two 74HC283 adders connected as an 8-bit parallel adder (pin numbers are in parentheses).

Design a 4 bit magnitude comparator. After this verify your design by comparing A= 12 and B=15 by applying bits to the circuit you implemented. The outputs must confirm that B>A.

Solution: you can see the design here: https://testbook.com/digital-electronics/comparators

Design a four-bit combinational circuit 2's complementor. (The output generates the 2's complement of the input binary number.) Show that the circuit can be constructed with exclusive-OR gates.



Truth Table

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

K-Maps:

W= $D'C + D'B + D'A + DC'B'A'$

DC/BA				
		1	1	1
	1	1	1	1
	1			

X= $C'B + C'A + BC'A'$

		1	1	1
	1			
	1			
		1	1	1

Y = $BA' + A'B$

		1		1
		1		1
		1		1
		1		1

Z = A

		1	1	
		1	1	
		1	1	
		1	1	

$W = D \oplus (C + B + A)$

$y = C \oplus (B + A)$

$y = B \oplus A$

$Z = A$

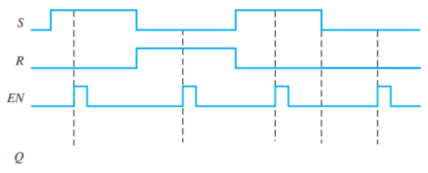
3 of 6

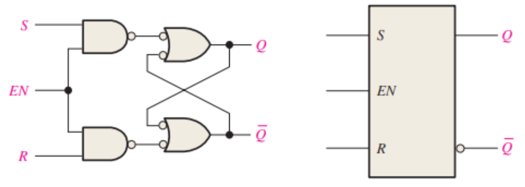
Question 3: Timing Diagrams | LATCHES

Outcomes: You must have knowledge of the following after this section for your exam point of view:
How timing diagrams for any circuit is made when given inputs are applied

a) **The Gated S-R Latch:** A gated latch requires an enable input, EN (G is also used to designate an enable input). The logic diagram and logic symbol for a gated S-R latch are shown in Figure.

a. Determine the Q output waveform if the inputs shown in Figure below are applied to a gated S-R latch that is initially RESET.



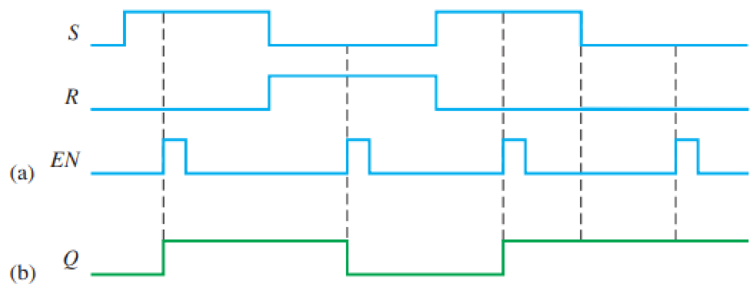


(a) Logic diagram (b) Logic symbol

A gated S-R latch.

b. Determine the Q output of a gated S-R latch if the S and R inputs in the timing diagram above are inverted.

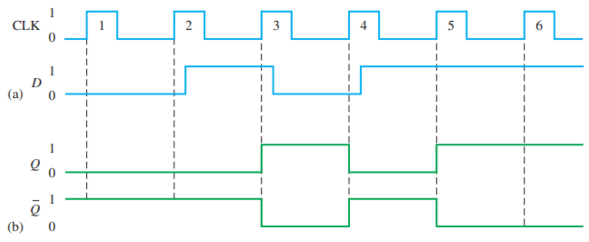
c. Draw the truth table of a gated S-R latch.

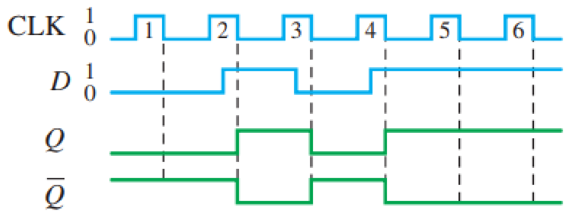


The Q waveform is shown. When **S is HIGH and R is LOW,** a HIGH on the EN input sets the latch. **When S is LOW and R is HIGH**, a HIGH on the EN input resets the latch. **When both S and R are LOW,** the Q output does not change from its present state. Same was for inverted.

b) Determine the Q and Q output waveforms of the flip-flop in **Figure 7–15** for the D and CLK inputs in **Figure 7–16(a).**

Fig 1 for +ve edge. Fig 2 for -ve edge





<u>**For Positive edge trigger:**</u>

Solution

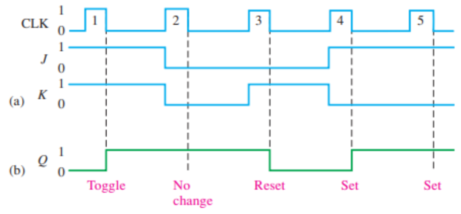
1. At clock pulse 1, D is LOW, so Q remains LOW (RESET).

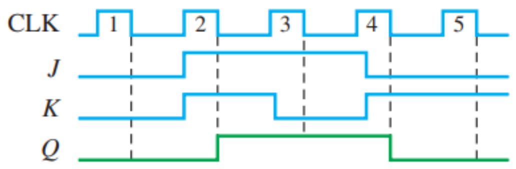
2. At clock pulse 2, D is LOW, so Q remains LOW (RESET).
3. At clock pulse 3, D is HIGH, so Q goes HIGH (SET).
4. At clock pulse 4, D is LOW, so Q goes LOW (RESET).
5. At clock pulse 5, D is HIGH, so Q goes HIGH (SET).
6. At clock pulse 6, D is HIGH, so Q remains HIGH (SET).

Once Q is determined, Q is easily found since it is simply the complement of Q. The resulting waveforms for Q and Q are shown in Figure 7–16(b) for the input waveforms in part (a).

The waveforms in Figure 7–18(a) are applied to the J, K, and clock inputs as indicated.

a. Determine the Q output, assuming that the flip-flop is initially **RESET**
b. Determine the Q output of the J-K flip-flop if the J and K inputs in Figure 7–18(a) are inverted.





Solution

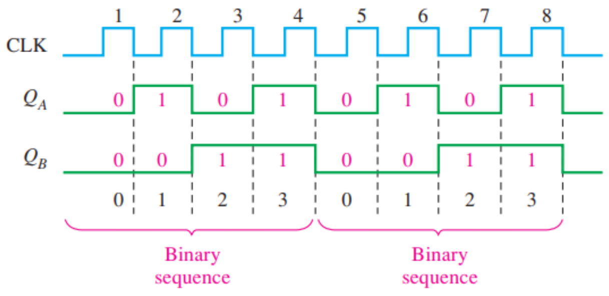
Since this is a negative edge-triggered flip-flop, as indicated by the "bubble" at the clock input, the Q output will change
only on the negative-going edge of the clock pulse.

1. At the first clock pulse, both J and K are HIGH; and because this is a toggle condition, Q goes HIGH.
2. At clock pulse 2, a no-change condition exists on the inputs, keeping Q at a HIGH level.
3. When clock pulse 3 occurs, J is LOW and K is HIGH, resulting in a RESET condition; Q goes LOW.
4. At clock pulse 4, J is HIGH and K is LOW, resulting in a SET condition; Q goes HIGH.
5. A SET condition still exists on J and K when clock pulse 5 occurs, so Q will remain HIGH.

The resulting Q waveform is indicated in Figure 7–18(b)

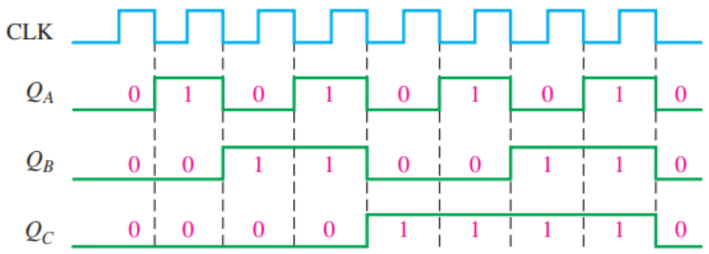
Part d:

Determine the output waveforms in relation to the clock for Q_A, Q_B, and Q_C in the circuit of F in figure (i) and (ii) and showthe binary sequence represented by these waveforms.



i)

ii) The output timing diagram is shown. Notice that the outputs change on the negative-going edge of the clock pulses. The outputs go through the binary sequence 000, 001, 010, 011, 100, 101, 110, and 111 as indicated.



Question 4: Counters

Outcomes: You must have knowledge of the following after this section for your exam point of view:
Make basic counters using flip flops.

For each of the following, draw state diagram, state table, and the corresponding steps required to complete your design.

a) Design a 3-bit binary counter using D-Flip Flop.
Solution: https://www.youtube.com/watch?v=xE-BOxZNJME

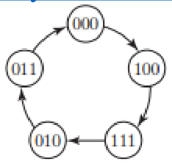


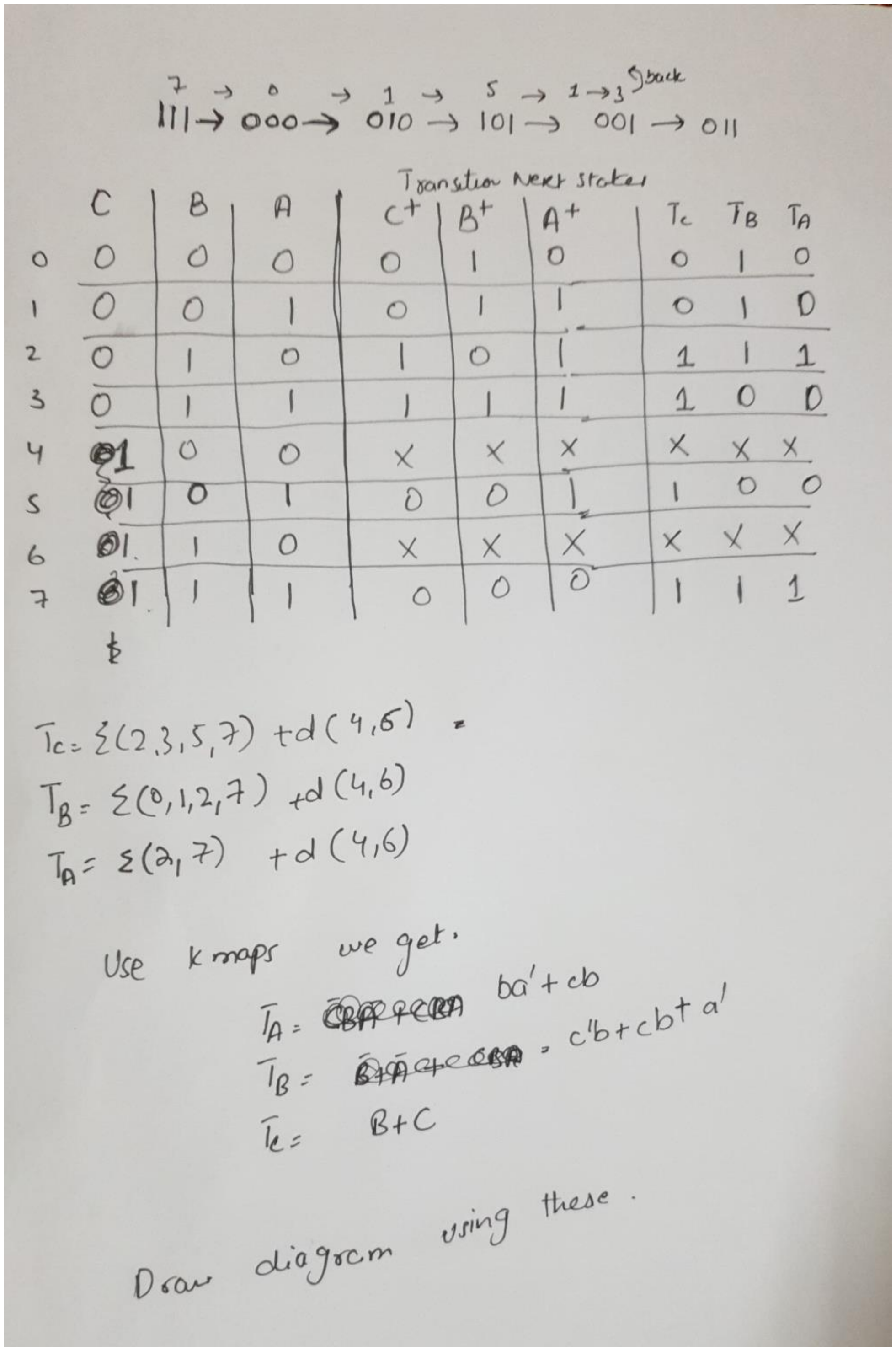
FIGURE 12-22
Transition Graph for Counter
© Cengage Learning 2014

TABLE 12-3
Transition Table for Figure 12-22
© Cengage Learning 2014

C	B	A	C^+	B^+	A^+
0	0	0	1	0	0
0	0	1	–	–	–
0	1	0	0	1	1
0	1	1	0	0	0
1	0	0	1	1	1
1	0	1	–	–	–
1	1	0	–	–	–
1	1	1	0	1	0

b) Rest steps are same as In part c.

c)



$$7 \rightarrow 0 \rightarrow 1 \rightarrow 5 \rightarrow 1 \rightarrow 3 \text{ back}$$

$$111 \rightarrow 000 \rightarrow 010 \rightarrow 101 \rightarrow 001 \rightarrow 011$$

Transition Next states

	C	B	A	C^+	B^+	A^+	T_c	T_B	T_A
0	0	0	0	0	1	0	0	1	0
1	0	0	1	0	1	1	0	1	0
2	0	1	0	1	0	1	1	1	1
3	0	1	1	1	1	1	1	0	0
4	1	0	0	X	X	X	X	X	X
5	1	0	1	0	0	1	1	0	0
6	1	1	0	X	X	X	X	X	X
7	1	1	1	0	0	0	1	1	1

$T_c = \Sigma(2,3,5,7) + d(4,6)$

$T_B = \Sigma(0,1,2,7) + d(4,6)$

$T_A = \Sigma(2,7) + d(4,6)$

Use k maps we get.

$T_A =$ ~~[illegible]~~ $ba' + cb$

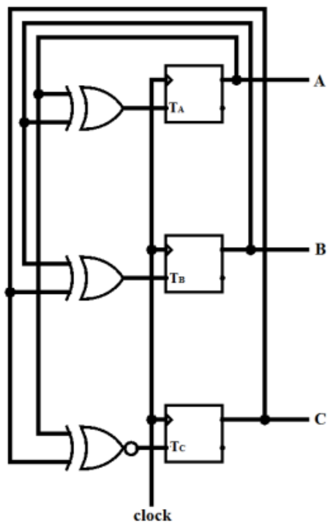
$T_B =$ ~~[illegible]~~ $= c'b + cb + a'$

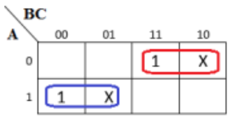
$T_c = B + C$

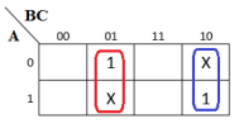
Draw diagram using these.

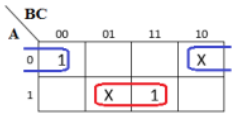
d)

Present State			Next State			Flip Flop Inputs		
A	**B**	**C**	**A**	**B**	**C**	**T_A**	**T_B**	**T_C**
0	0	0	0	0	1	0	0	1
0	0	1	0	1	1	0	1	0
0	1	0	X	X	X	X	X	X
0	1	1	1	1	1	1	0	0
1	0	0	0	0	0	1	0	0
1	0	1	X	X	X	X	X	X
1	1	0	1	0	0	0	1	0
1	1	1	1	1	0	0	0	1









$T_A = A'B + AB'$

$T_A = A \oplus B$

$T_B = B'C + BC'$

$T_B = B \oplus C$

$T_C = AC + A'C'$

$T_C = (A \oplus C)'$

In the above design the counter is not self correcting. In order to make the counter self-correcting,

$T_C = AC + A'B'C'$

With D Flip Flops

Present State				**Next State**				**Output**
A	**B**	**C**	**D**	**A**	**B**	**C**	**D**	**y**
0	0	0	0	0	0	0	1	0
0	0	0	1	0	0	1	0	0
0	0	1	0	0	0	1	1	0
0	0	1	1	0	1	0	0	0
0	1	0	0	0	1	0	1	0
0	1	0	1	0	1	1	0	0
0	1	1	0	0	1	1	1	0
0	1	1	1	1	0	0	0	0
1	0	0	0	1	0	0	1	0
1	0	0	1	0	0	0	0	1
1	0	1	0	X	X	X	X	X
1	0	1	1	X	X	X	X	X
1	1	0	0	X	X	X	X	X
1	1	0	1	X	X	X	X	X
1	1	1	0	X	X	X	X	X
1	1	1	1	X	X	X	X	X

By using k-map, we can get the equations of D_A, D_B, D_C, D_D and y.

$D_A = AD' + BCD$
$D_B = BC' + BD' + B'CD = B \oplus (CD)$
$D_C = A'C'D + CD$
$D_D = D'$

$y = AD$

e) By using above equations, we can easily construct the circuit diagram.

With JK Flip Flops

Present State				**Next State**				**Output**	**Flip Flop Inputs**							
A	**B**	**C**	**D**	**A**	**B**	**C**	**D**	**y**	**J_A**	**K_A**	**J_B**	**K_B**	**J_C**	**K_C**	**J_D**	**K_D**
0	0	0	0	0	0	0	1	0	0	X	0	X	0	X	1	X
0	0	0	1	0	0	1	0	0	0	X	0	X	1	X	X	1
0	0	1	0	0	0	1	1	0	0	X	0	X	X	0	1	X
0	0	1	1	0	1	0	0	0	0	X	1	X	X	1	X	1
0	1	0	0	0	1	0	1	0	0	X	X	0	0	X	1	X
0	1	0	1	0	1	1	0	0	0	X	X	0	1	X	X	1
0	1	1	0	0	1	1	1	0	0	X	X	0	X	0	1	X
0	1	1	1	1	0	0	0	0	1	X	X	1	X	1	X	1
1	0	0	0	1	0	0	1	0	X	0	0	X	0	X	1	X
1	0	0	1	0	0	0	0	1	X	1	0	X	0	X	X	1
1	0	1	0	X	X	X	X	X	X	X	X	X	X	X	X	X
1	0	1	1	X	X	X	X	X	X	X	X	X	X	X	X	X
1	1	0	0	X	X	X	X	X	X	X	X	X	X	X	X	X
1	1	0	1	X	X	X	X	X	X	X	X	X	X	X	X	X
1	1	1	0	X	X	X	X	X	X	X	X	X	X	X	X	X
1	1	1	1	X	X	X	X	X	X	X	X	X	X	X	X	X

By using k-maps the equations of J_A, K_A, J_B, K_B, J_C, K_C, J_D, K_D and y are found.

$J_A = BCD$
$K_A = D$

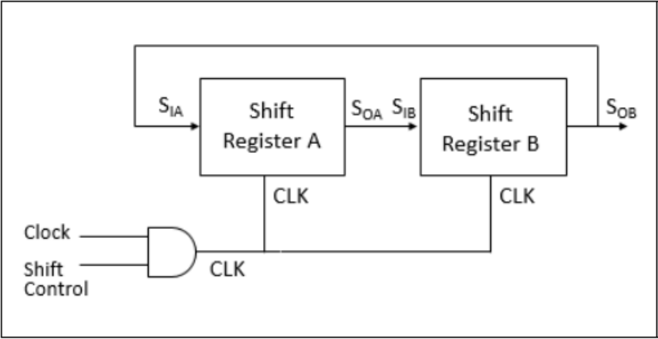
$J_B = K_B = CD$

$J_C = A'D$
$K_C = D$

$J_D = K_D = 1$

$y = AD$

By using above equations, we can easily construct the circuit diagram.



Pulse	Register A	Register B
Initial Value	1101	0110
T1	0110	1011
T2	1011	0101
T3	1101	1010
T4	0110	1101

f)

Question 5: State table reductions

Outcomes: You must have knowledge of the following after this section for your exam point of view:
Be able to reduce state tables and re-draw the state diagrams from it.

a) Draw the state diagram for the table below that describes a finite-state machine which has one input x and one output z.

Present State	Next State		Output (z)	
	$x = 0$	$x = 1$	$x = 0$	$x = 1$
A	A	E	1	0
B	C	F	0	0
C	B	H	0	1
D	E	F	0	0
E	D	A	0	1
F	B	F	1	1
G	D	H	0	1
H	H	G	1	0

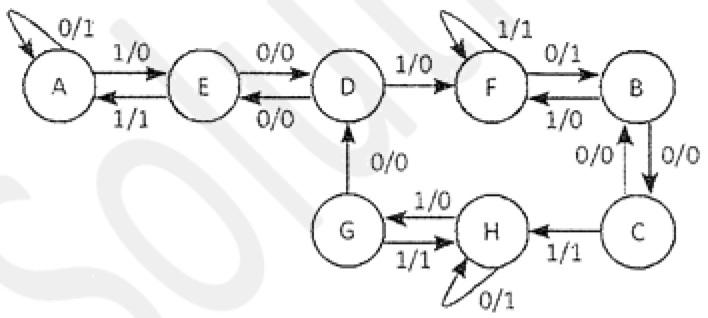
a) Determine whether it is a mealy machine or moor machine?

b) Assign the binary codes to all the states by using:

   a. One hot assignment

   b. Binary assignment

Part (i)



Part (ii)

As the output is a function of both input and present state, hence it is a mealy machine. The output is changing on the same state.

Part (iii)

State	Assignment	
	Binary	One Hot
A	000	00000001
B	001	00000010
C	010	00000100
D	011	00001000
E	100	00010000
F	101	00100000
G	110	01000000
H	111	10000000

b) Reduce the following state table to minimum states.

Table (2)
Reducing the State Table.

Present State	Next State		Output	
	x=0	x=1	x=0	x=1
a	a	b	0	0
b	c	d	0	0
c	a	d	0	0
d	e	f	0	1
e	a	f	0	1
f	e	f	0	1

Table (3)
Reduced State Table.

Present State	Next State		Output	
	x=0	x=1	x=0	x=1
a	a	b	0	0
b	c	d	0	0
c	a	d	0	0
d	e	d	0	1
e	a	d	0	1

Going through the state table, we look for two present states that go to the same next state and have the same output for both input combinations. **States (g) and (e) are two such states** The procedure of removing a state and replacing it by its equivalent is demonstrated in Table (2). **The row with present state (g) is removed and state (g) is replaced by state (e) each time it occurs in the next-state columns.**

States (f and d) are equivalent and state (f) can be removed and replaced by (d) as shown in Table (3)

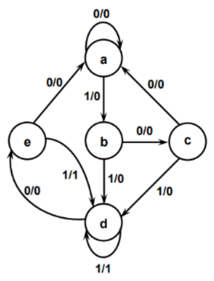


Fig. (2):
Reduced State Diagram.

3

c) Reduce the given state table to minimum possible number of states.

Present State	Next State		Output	
	$x = 0$	$x = 1$	$x = 0$	$x = 1$
A	A	E	1	0

B	C	F	0	1
C	B	H	1	0
D	B	F	1	0
E	D	F	0	1
F	H	G	1	1
G	D	H	0	1
H	H	G	1	1

i. Determine the number of flip flops required to design a sequential circuit described by the above-mentioned state table?
ii. Determine the number of flip flops required to design a sequential circuit described by the reduced state table?
iii. Draw the state diagram corresponding to the reduced state table.
iv. Design the circuit described by the reduced state table by using JK flip flop(s).

Part (i)

F and H are equivalent, remove H and replace H with F.

Present State	Next State		Output	
	$x = 0$	$x = 1$	$x = 0$	$x = 1$
A	A	E	1	0
B	C	F	0	1
C	B	F	1	0
D	B	F	1	0
E	D	F	0	1
F	F	G	1	1
G	D	F	0	1

E and G are equivalent, remove G and replace G with E.

C and D are equivalent, remove D and replace D with C.

Present State	Next State		Output	
	$x = 0$	$x = 1$	$x = 0$	$x = 1$
A	A	E	1	0
B	C	F	0	1
C	B	F	1	0
E	C	F	0	1
F	F	E	1	1

B and E are equivalent, remove E and replace E with B.

Present State	Next State		Output	
	$x = 0$	$x = 1$	$x = 0$	$x = 1$
A	A	B	1	0
B	C	F	0	1
C	B	F	1	0
F	F	B	1	1

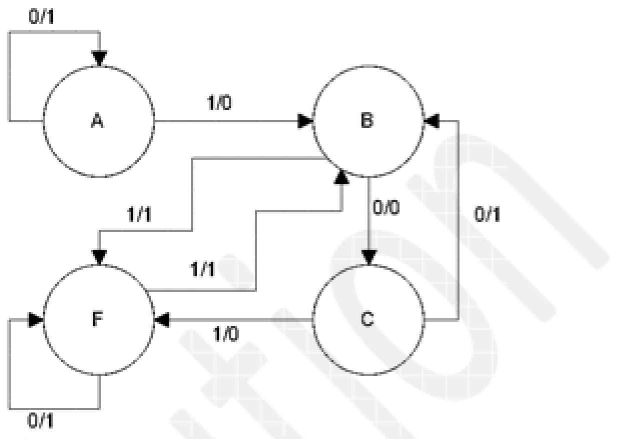
Part (ii)

As there are 8 states in the given table. So we need 3 flip flops to design the circuit.

Part (iii)

In the reduced state table there are 4 states, hence we need 2 flip flops to design the circuit given by reduced state table.

Part (iv)



Part (v)

Present State		Input	Next State		Output	Flip Flop Inputs			
A	B	x	A	B	y	J_A	K_A	J_B	K_B
0	0	0	0	0	1	0	X	0	X
0	0	1	0	1	0	0	X	1	X
0	1	0	1	0	0	1	X	X	1
0	1	1	1	1	1	1	X	X	0
1	0	0	0	1	1	X	1	1	X
1	0	1	1	1	0	X	0	1	X
1	1	0	1	1	1	X	0	X	0
1	1	1	0	1	1	X	1	X	0

By using k-maps, the equations are:

$J_A = B$

$K_A = B'x' + Bx = (B \oplus x)'$

$J_B = A + x$

$K_B = Ax'$

$y = B'x' + Bx + AB = (B \oplus x)' + AB$

with the help of above equations, we can draw the circuit diagram.

Question 6: Sequential Circuit – Design and analysis

Outcomes: You must have knowledge of the following after this section for your exam point of view:
Be able to design and analyze the sequential circuits.

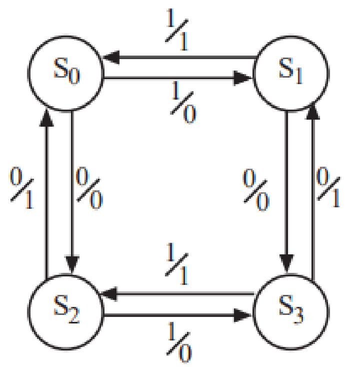
a) Design a sequential circuit having one input and one output that will produce an output of 1 for every second 0 it receives and for every second 1 it receives.

Example:

$$X(\text{input}) = 0\ 1\ 1\ 0\ 1\ 1\ 1\ 0\ 0\ 0\ 0\ 1\ 0\ 1\ 1\ 0\ 0\ 1\ 0\ 1\ 1\ 0\ 1\ 0$$
$$Z(\text{output}) = 0\ 0\ 1\ 1\ 0\ 1\ 0\ 0\ 1\ 0\ 1\ 1\ 0\ 0\ 1\ 1\ 0\ 0\ 1\ 1\ 0\ 0\ 1\ 1$$

Design a Mealy sequential circuit using D flip-flops, showing a reduced state graph, and equations for the output and D inputs. It should be a reasonably economical design.

ı) The output becomes 1 whenever an even #0's or an even #1's (greater than 0) occurs.



The state meanings are given in the following table:

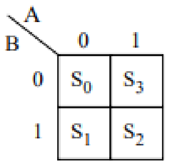
Name	Meaning
S_0	even #0's and even #1's received
S_1	even #0's and odd #1's received
S_2	odd #0's and even #1's received
S_3	odd #0's and odd #1's received

Present State	Next State $X=0$	Next State 1	Output $X=0$	Output $X=1$
S_0	S_2	S_1	0	0
S_1	S_3	S_0	0	1
S_2	S_0	S_3	1	0
S_3	S_1	S_2	1	1

Guidelines: I: --

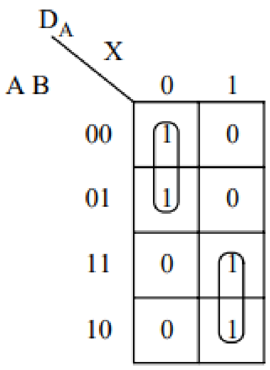
II: (1, 2)2x, (0, 3)2x

An assignment is



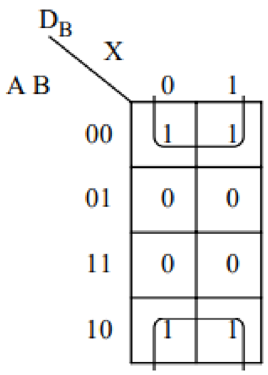
B \ A	0	1
0	S_0	S_3
1	S_1	S_2

A B	$A^+ B^+$ $X=0$	$A^+ B^+$ 1	Z $X=0$	Z $X=1$
0 0	1 1	0 1	0	0
0 1	1 0	0 0	0	1
1 1	0 0	1 0	1	0
1 0	0 1	1 1	1	1



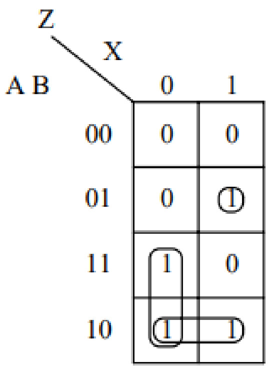
D_A: A B \ X	0	1
00	1	0
01	1	0
11	0	1
10	0	1

$D_A = X'A' + XA$



D_B: A B \ X	0	1
00	1	1
01	0	0
11	0	0
10	1	1

$D_B = B'$



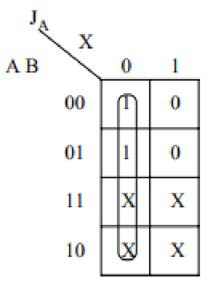
Z: A B \ X	0	1
00	0	0
01	0	1
11	1	0
10	1	1

$Z = X'A + XA'B + AB'$

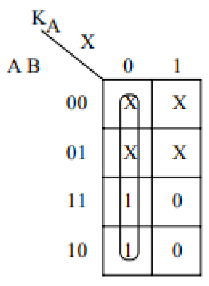
The above solution is for the question that was in assignment, but what if we want to make it using JK FF so then the design will be like this , just for your final exams pov:

A B	$J_A K_A$ X = 0	1
0 0	1 X	0 X
0 1	1 X	0 X
1 1	X 1	X 0
1 0	X 1	X 0

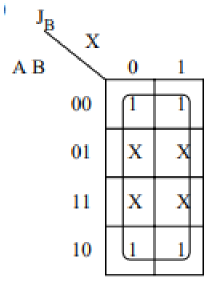
A B	$J_B K_B$ X = 0	1
0 0	1 X	1 X
0 1	X 1	X 1
1 1	X 1	X 1
1 0	1 X	1 X



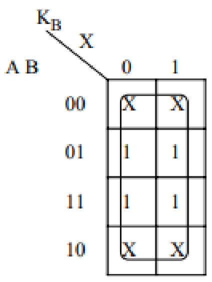
$J_A = X'$



$K_A = X'$



$J_B = 1$

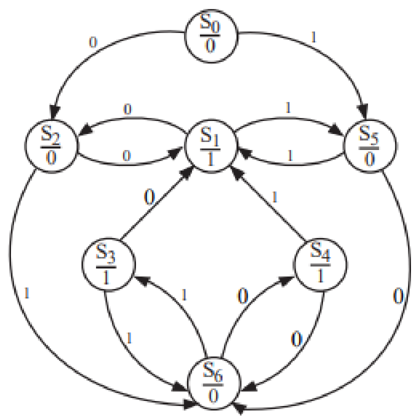


$K_B = 1$

What if we wanted to make it into MOORE Machine? Lets see that as well for your final exam pov:
Design a Moore sequential circuit using T flip-flops to do the same task, showing a state graph and input equations for a reasonably economical design.

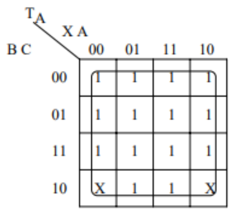
The state meanings are given in the following table:

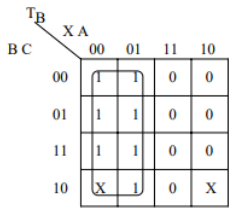
Name	Meaning
S_0	reset state
S_1	even #0's and even #1's received
S_2	odd #0's and even #1's received
S_3	odd #0's and even #1's received
S_4	even #0's and odd #1's received
S_5	even #0's and odd #1's received
S_6	odd #0's and odd #1's received

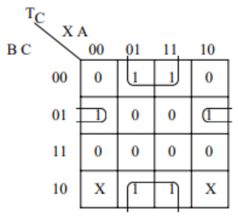


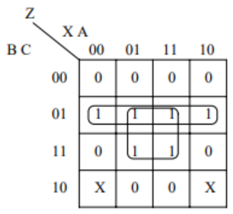
Present State	Next State X = 0	1	Z
S_0	S_2	S_5	0
S_1	S_2	S_5	1
S_2	S_1	S_6	0
S_3	S_1	S_6	1
S_4	S_6	S_1	1
S_5	S_6	S_1	0
S_6	S_4	S_3	0

	ABC	$A^+B^+C^+$ $X = 0$	1	Z
S_0	000	110	100	0
S_1	001	110	100	1
S_6	011	101	111	0
--	010	---	---	-
S_5	100	011	001	0
S_4	101	011	001	1
S_3	111	001	011	1
S_2	110	001	011	0

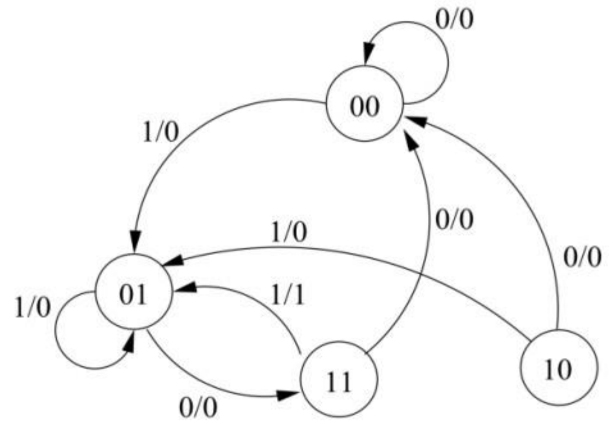

$T_A = 1$


$T_B = X'$


$T_C = A'B'C + AC'$


$Z = B'C + AC$

b) **Consider the following state diagram for a synchronous circuit with one input X and one output Z. Design the below machine by using T Flip Flop(s).**



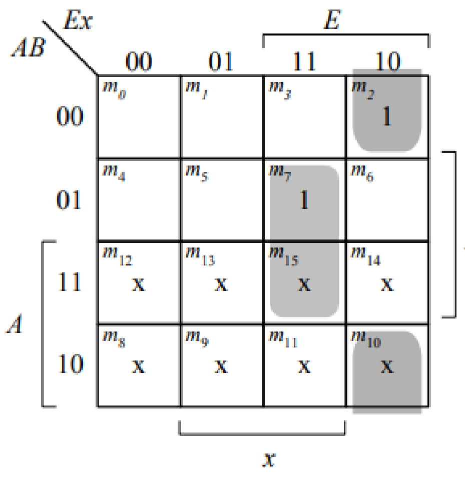
Present State		**Input**	**Next State**		**Output**	**T FF Inputs**	
A	**B**	*x*	**A+**	**B+**	*y*	**T_A**	**T_B**
0	0	0	0	0	0	0	0
0	0	1	0	1	0	0	1
0	1	0	1	1	0	1	0
0	1	1	0	1	0	0	0
1	0	0	0	0	0	1	0
1	0	1	0	1	0	1	1
1	1	0	0	0	0	1	1
1	1	1	0	1	1	1	0

By using k-map we can find the equations.

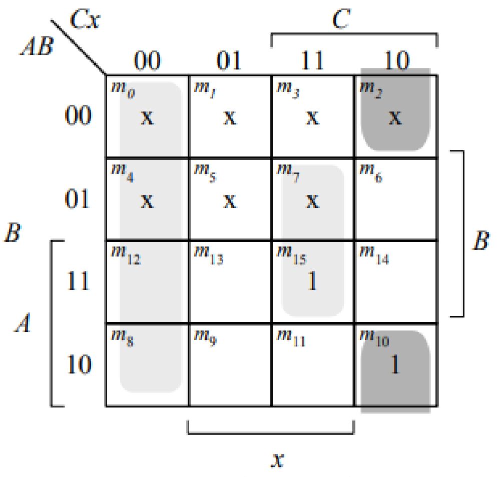
$$T_A = A + Bx'$$
$$T_B = B'x + ABx'$$
$$y = ABx$$

c) **Create a sequential circuit utilizing two JK flip-flops labeled A and B, alongside two inputs, E and F. When E is set to 0, the circuit should preserve its current state, disregarding any changes in F. When E equals 1 and F also equals 1, the circuit should follow a sequence of state transitions: starting from 00, moving to 01, then to 10, subsequently to 11, and finally returning to 00, repeating this cycle indefinitely. Conversely, when E is 1 and F is 0, the circuit should adhere to a different sequence of state transitions: beginning at 00, progressing to 11, then to 10, moving to 01, and finally circling back to 00, continuously repeating this pattern.**

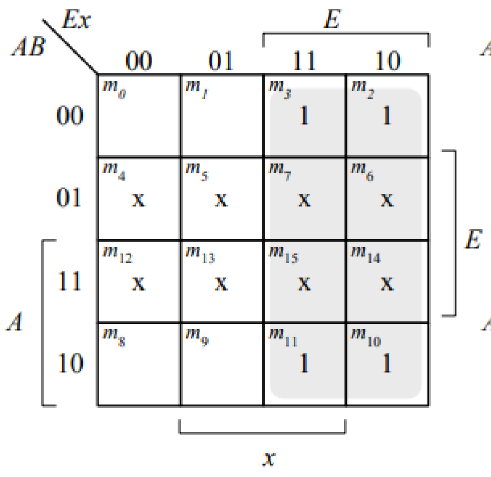
Present state	Input	Next state	Flip-flop inputs			
$A\ B$	$E\ F$	$A\ B$	J_A	K_A	J_B	K_B
0 0	0 1	0 0	0	x	0	x
0 0	0 1	0 0	0	x	0	x
0 0	1 0	1 1	1	x	1	x
0 0	1 1	0 1	0	x	1	x
0 1	0 0	0 1	0	x	x	0
0 1	0 1	0 1	0	x	x	0
0 1	1 0	0 1	0	x	x	1
0 1	1 1	1 0	1	x	x	1
1 0	0 0	1 0	x	0	1	0
1 0	0 1	1 0	x	0	1	0
1 0	1 0	0 1	x	1	x	1
1 0	1 1	1 1	x	0	x	1
1 1	0 0	1 1	x	0	x	0
1 1	0 1	1 1	x	0	x	0
1 1	1 0	1 1	1	0	x	1
1 1	1 1	1 1	x	1	x	1



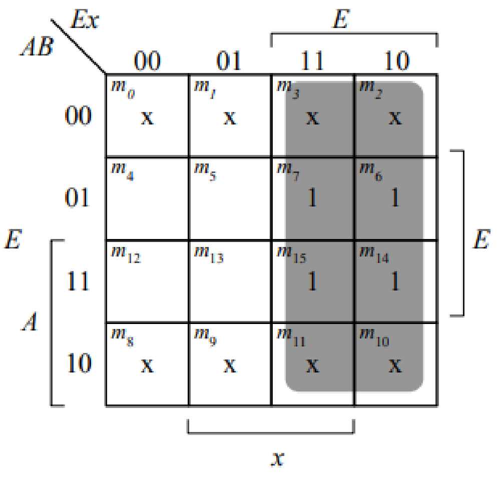
$J_A = (Bx + B'x')E$



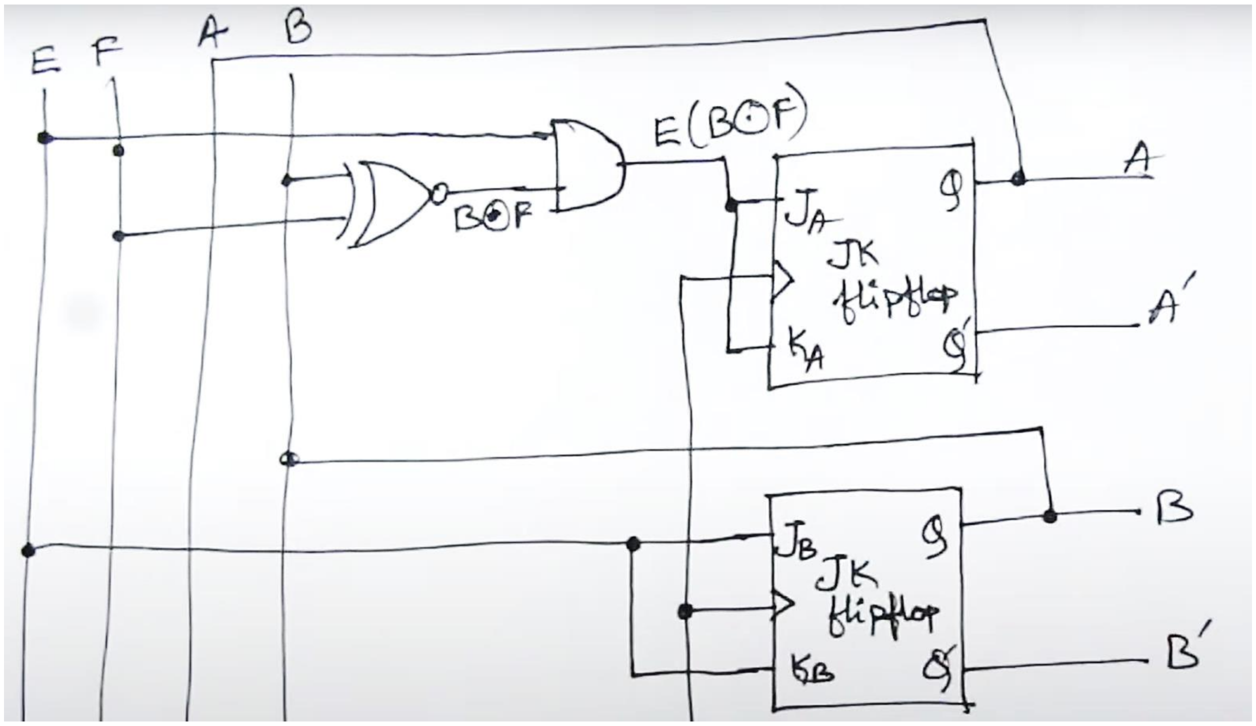
$K_A = (Bx + B'x')E$



$J_B = E$



$K_B = E$



For Reference: https://www.youtube.com/watch?v=t875Z-VCasQ

d) **Given below is the circuit diagram of a synchronous (same clock is applied to both flip flops) sequential circuit with two flip flops (JK), one input x, and no output.** Analyze the given circuit to find the:

Equations:

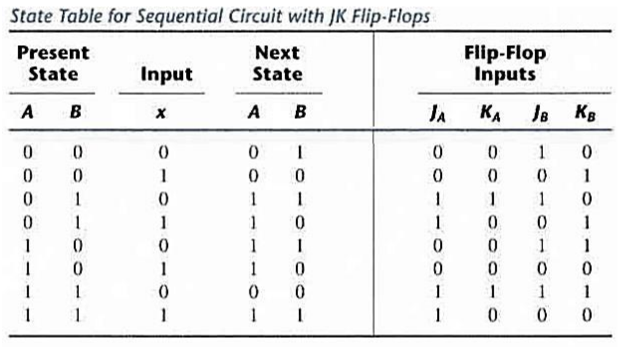
JA = B

KA = Bx'

JB = x'

KB = A'x + Ax' = A ⨁ x

State Table:



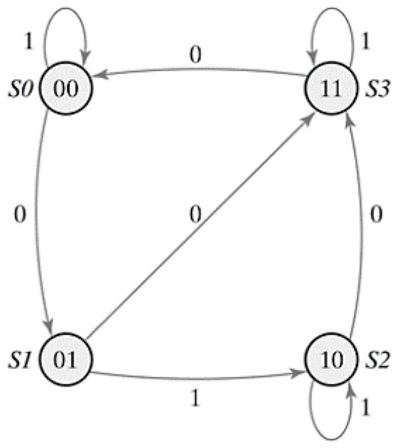
State Table for Sequential Circuit with JK Flip-Flops

Present State		Input	Next State		Flip-Flop Inputs			
A	B	x	A	B	J_A	K_A	J_B	K_B
0	0	0	0	1	0	0	1	0
0	0	1	0	0	0	0	0	1
0	1	0	1	1	1	1	1	0
0	1	1	1	0	1	0	0	1
1	0	0	1	1	0	0	1	1
1	0	1	1	0	0	0	0	0
1	1	0	0	0	1	1	1	1
1	1	1	1	1	1	0	0	0

Next State Equations:

The next-state values can also be obtained by evaluating the state equations from the characteristic equation. This is done by using the following procedure:
1. Determine the flip-flop input equations in terms of the present state and input variables.
2. Substitute the input equations into the flip-flop characteristic equation to obtain the state equations.
3. Use the corresponding state equations to determine the next state values in the state table
A (t + 1) = JA' + K'A
B (t + 1) = JB' + K'B
Substituting the values of JA and KA from the input equations, we obtain the state equation for A:
A (t + 1) = BA' + (Bx')'A
= A'B + AB' + Ax
Substituting the values of JB and KB from the input equations, we obtain the state equation for A:
B (t + 1) = x'B' + (A ⊕ x)'B
= B'x' + ABx + A'Bx'

State Diagram:

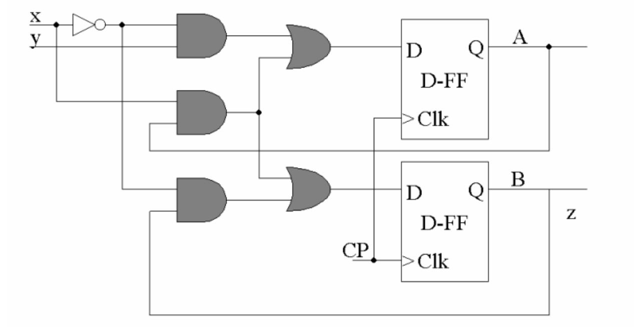


e) **A sequential circuit with two D Flip-Flops, A and B; two inputs, x and y; and one output, z, is specified by the following next-state and output equations:**

$$A(t+1) = x'y + xA$$
$$B(t+1) = x'B + xA$$
$$z = B$$

a) Draw the logic diagram of the circuit.
b) List the state table for the sequential circuit.
c) Draw the corresponding state diagram.

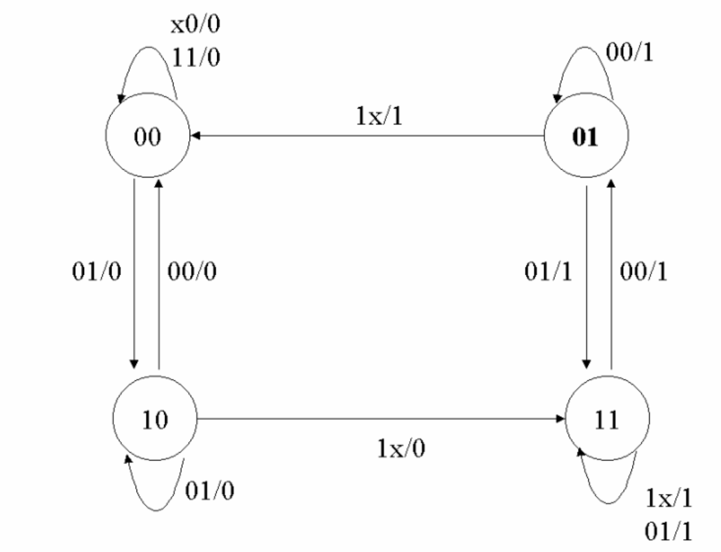
a)



b)

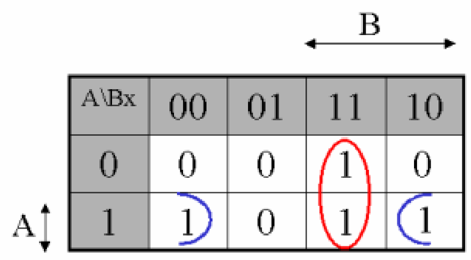
Present State		Inputs		Next State		Output
A	B	x	y	A	B	z
0	0	0	0	0	0	0
0	0	0	1	1	0	0
0	0	1	0	0	0	0
0	0	1	1	0	0	0
0	1	0	0	0	1	1
0	1	0	1	1	1	1
0	1	1	0	0	0	1
0	1	1	1	0	0	1
1	0	0	0	0	0	0
1	0	0	1	1	0	0
1	0	1	0	1	1	0
1	0	1	1	1	1	0
1	1	0	0	0	1	1
1	1	0	1	1	1	1
1	1	1	0	1	1	1
1	1	1	1	1	1	1

c)



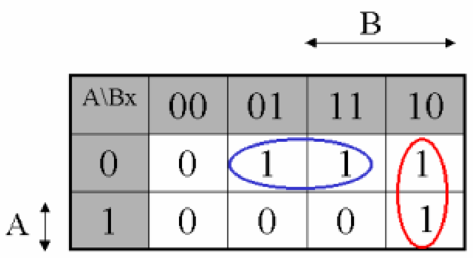
F) Construct a sequential circuit incorporating two D flip-flops, denoted as A and B, along with a single input, x. When x is set to 0, the circuit maintains its current state. Conversely, when x is set to 1, the circuit undergoes a sequence of state transitions as follows: starting from 00, advancing to 01, then to 11, subsequently to 10, and finally returning to 00, continuously repeating this cycle.

Present State AB	Input x	Nest State AB
00	0	00
00	1	01
01	0	01
01	1	11
10	0	10
10	1	00
11	0	11
11	1	10



A\Bx	00	01	11	10
0	0	0	1	0
1	1	0	1	1

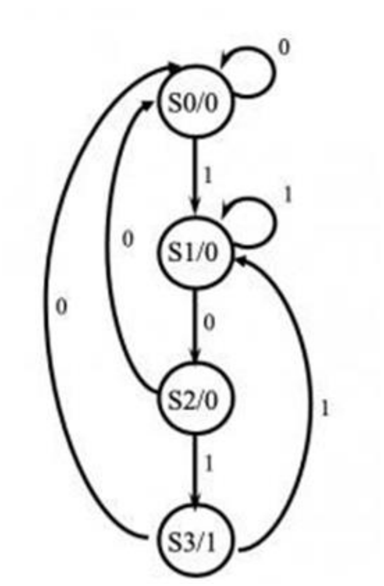
$D_A = AX' + BX$



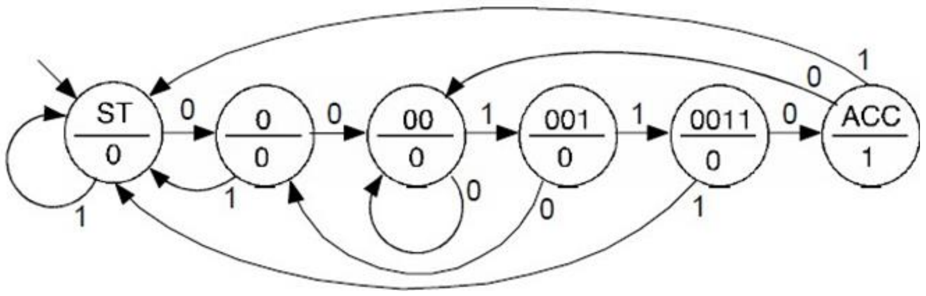
A\Bx	00	01	11	10
0	0	1	1	1
1	0	0	0	1

$D_B = A'X + BX'$

g) Draw the state diagram to detect the sequence 101.



h) Draw the state diagram to detect the sequence 00110.



i) https://www.youtube.com/watch?v=DSClog6vn3A