BILKENT UNIVERSITY
Department of Electrical and Electronics Engineering
EEE102 Introduction to Digital Circuit Design
Midterm Exam-2
SOLUTION
20-12-2006

Duration 120 minutes

Surname: _____

Name: _____

ID-Number: _____

Signature: _____

There are 7 questions. Solve all.
Do not detach pages. Show all your work.

Q1	
Q2	
Q3	
Q4	
Q5	
Q6	
Q7	
Total	

Q1. (20 points)
Design a Finite State Machine with one input and two outputs that counts the number of 1's in the input according to mod 4. Note that initially, the number of 1's is zero.

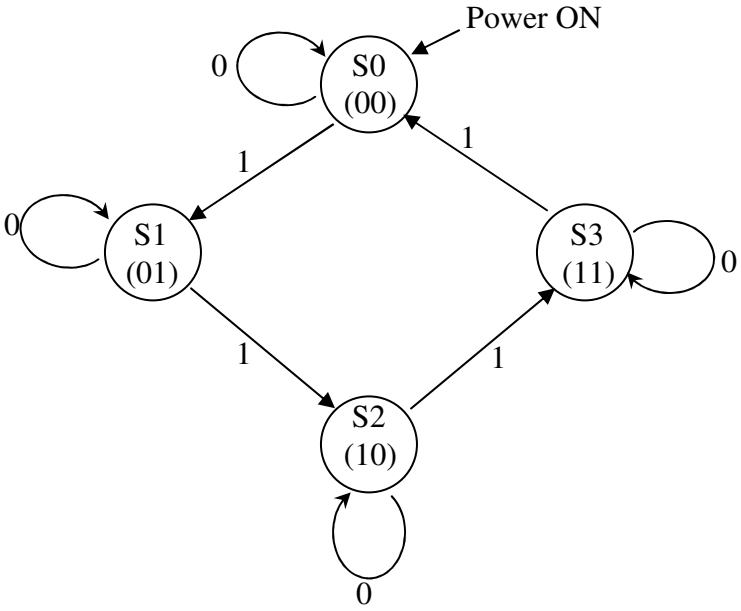
Show all your design steps including state definition, state diagram, state encoding, next state table, output table, excitation table and minimization of the combinatorial circuits. Draw your circuit.

Solution:

State definitions and state encoding:

states	definition	Q1	Q0
S0	**Initial state, number of 1s is 0**	**0**	**0**
S1	**Number of 1s is 1 (mod 4)**	**0**	**1**
S2	**Number of 1s is 2 (mod 4)**	**1**	**0**
S3	**Number of 1s is 3 (mod 4)**	**1**	**1**

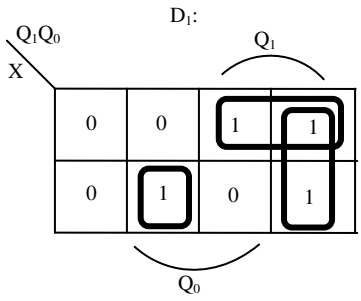
State/output diagram:



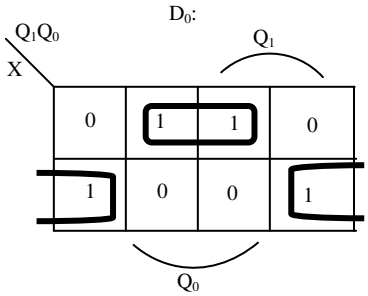
Next State and excitation table:

Q1	Q0	X	Q1*=D1	Q0*=D0
0	**0**	**0**	**0**	**0**
0	**0**	**1**	**0**	**1**
0	**1**	**0**	**0**	**1**
0	**1**	**1**	**1**	**0**
1	**0**	**0**	**1**	**0**
1	**0**	**1**	**1**	**1**
1	**1**	**0**	**1**	**1**
1	**1**	**1**	**0**	**0**

Minimization of next state circuit:



$D_1 = Q_1X' + Q_1 Q_0' + Q_1' Q_0X$

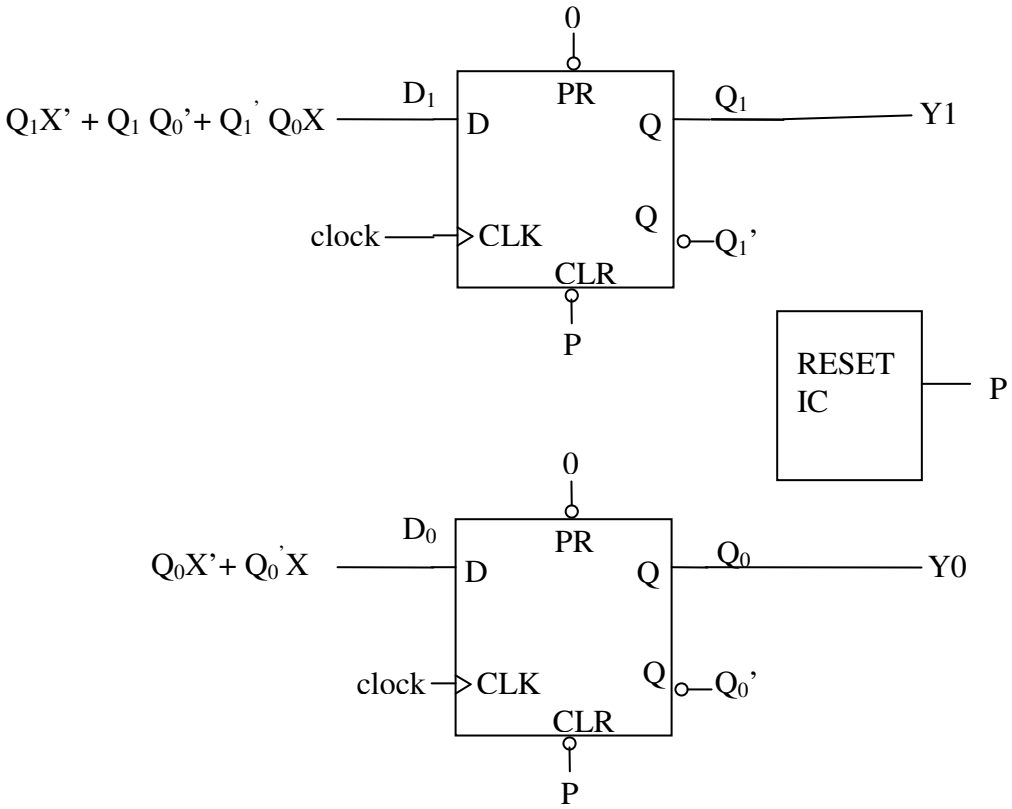


$D_0 = Q_0X' + Q_0'X$

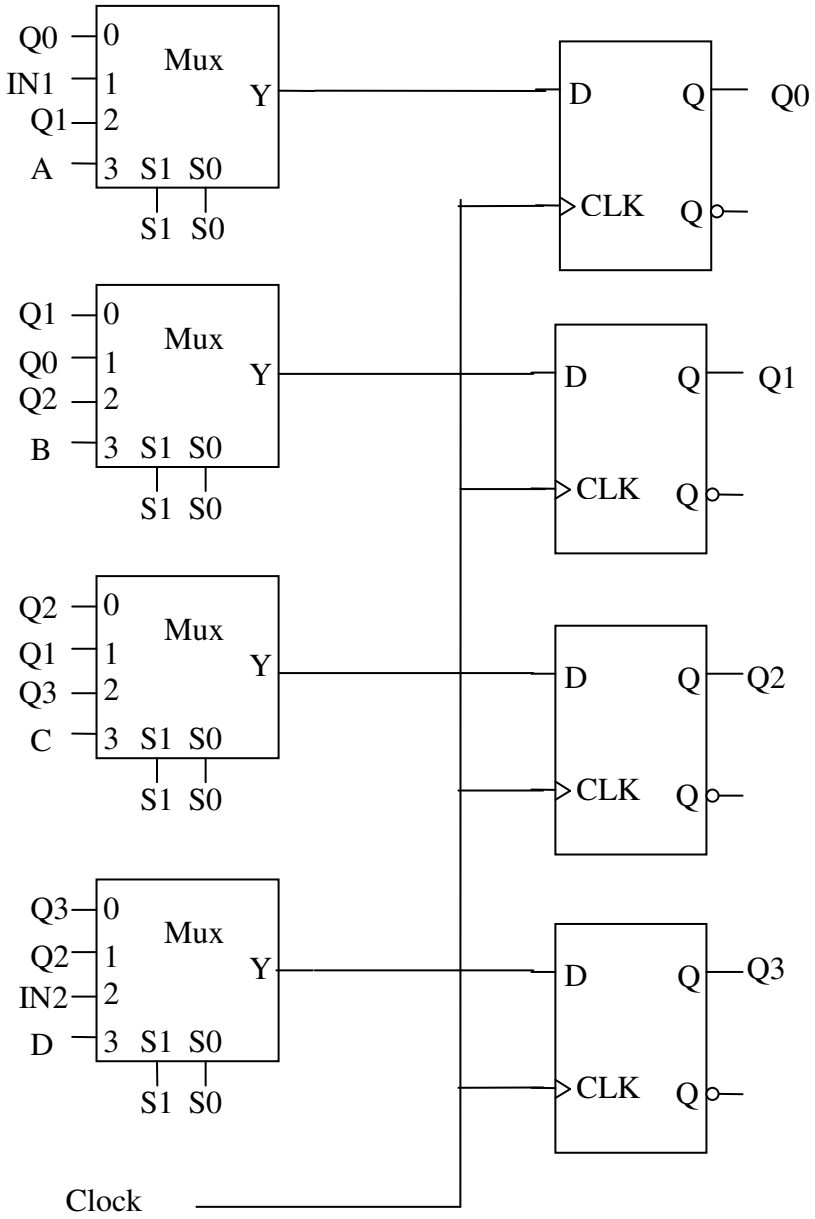
Output table and output circuit minimization:

Q1	Q0	Y1	Y0
0	0	0	0
0	1	0	1
1	0	1	0
1	1	1	1

Y1=Q1, Y0=Q0



Q2. (15 Points)
Write down the function table (compressed truth table) of the following circuit indicating what the circuit does for each case of the select inputs S1 and S0. What are IN1 and IN2?



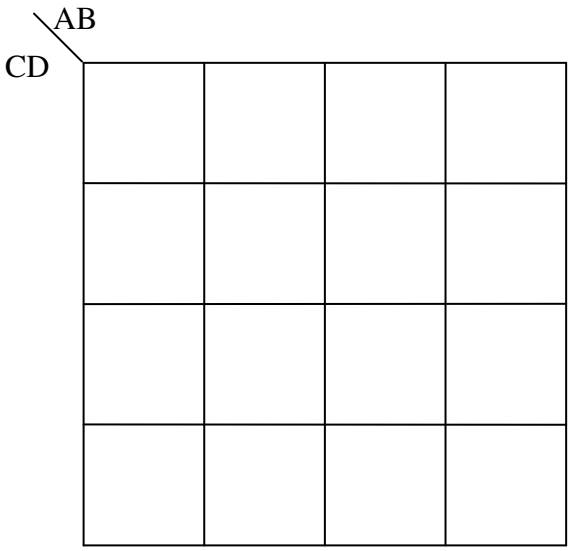
Solution:

S1	S0	Clock	Q3	Q2	Q1	Q0	
0	**0**	↑	**Q3**	**Q2**	**Q1**	**Q0**	**Hold**
0	**1**	↑	**Q2**	**Q1**	**Q0**	**IN1**	**Shift Left**
1	**0**	↑	**IN2**	**Q3**	**Q2**	**Q1**	**Shift Right**
1	**1**	↑	**D**	**C**	**B**	**A**	**Load**

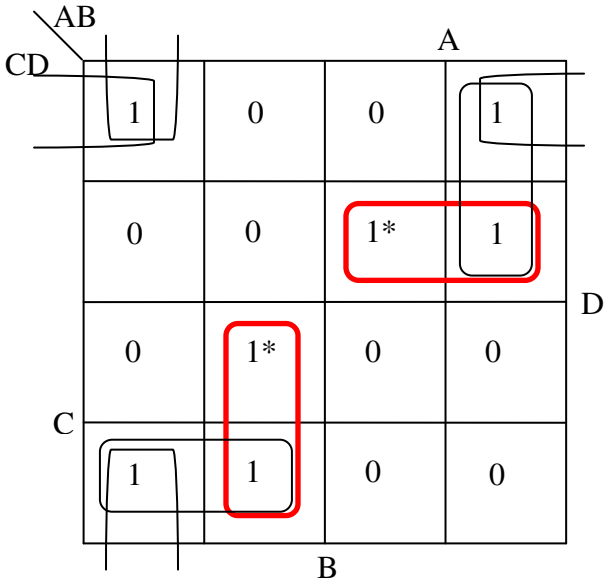
IN1 is shift left input. IN2 is shift right input.

Q3. (10 points)
Find three different minimal sum-of-products (SOP) for the following function by using the following K-map.

$\sum_{A,B,C,D}(0,2,6,7,8,9,13)$



Solution:



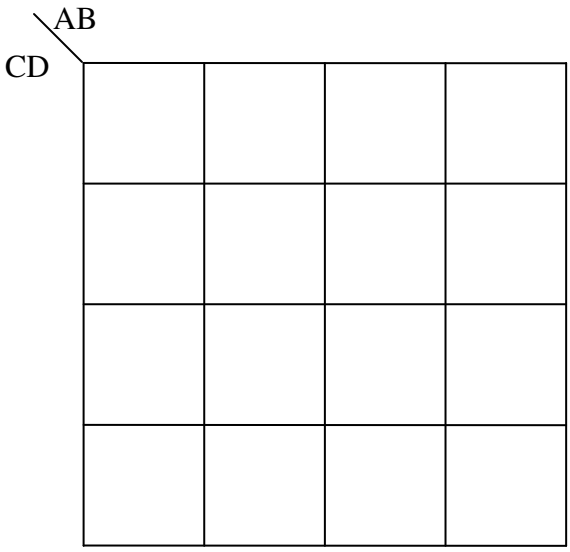
$F_1 = A'BC + AC'D + B'C'D' + A'B'D'$

$F_2 = A'BC + AC'D + B'C'D' + A'CD'$

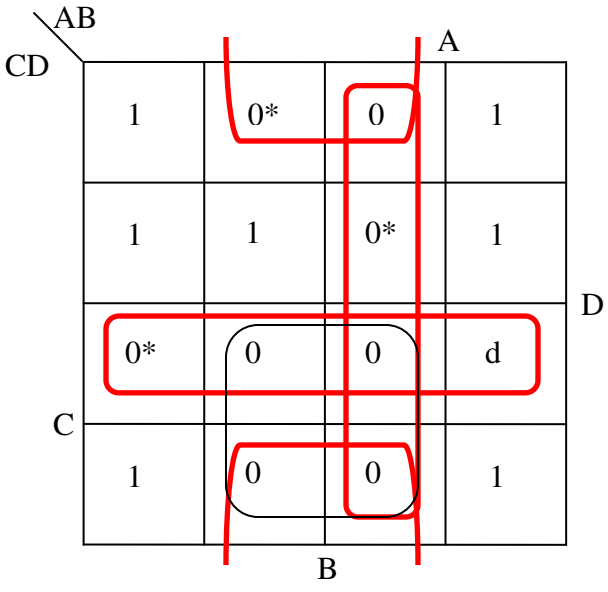
$F_3 = A'BC + AC'D + A'B'D' + AB'C'$

Q4. (10 points)

$F=\Sigma_{A,B,C,D}(0,1,2,5,8,9,10) + d(11)$
Find all minimal product-of-sums (POS) expressions for F by using the following K-map.



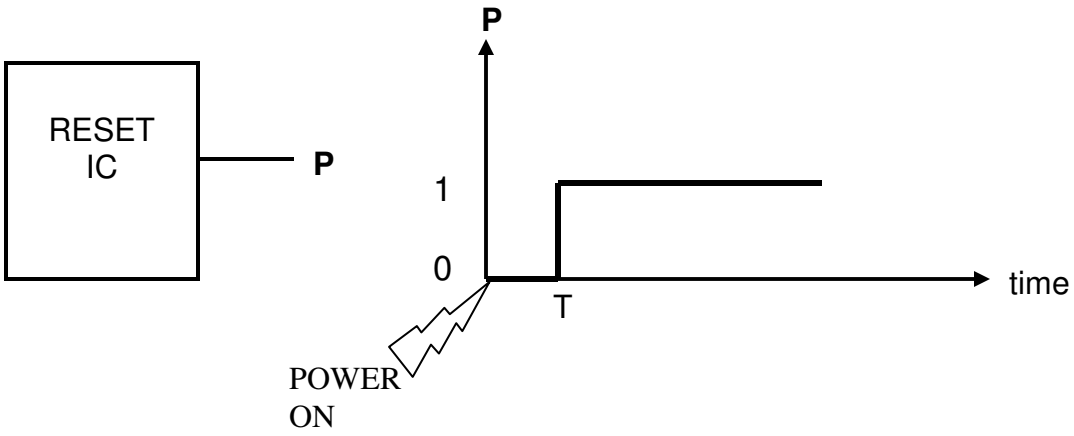
Solution:



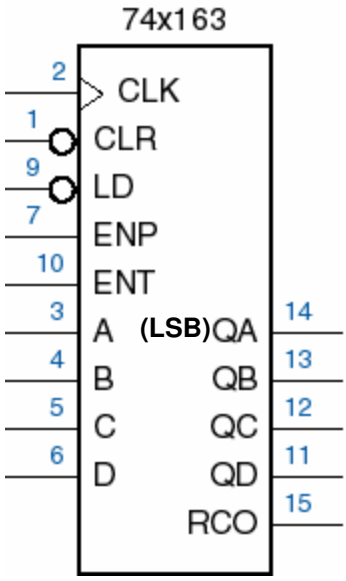
$$F = (B'+D)(A'+B')(C'+D')$$

Q5. (15 points)
Design a counter that counts in the following sequence by using the MSI package 74X163 given the pin diagram as below. (You may use additional simple logic gates.)

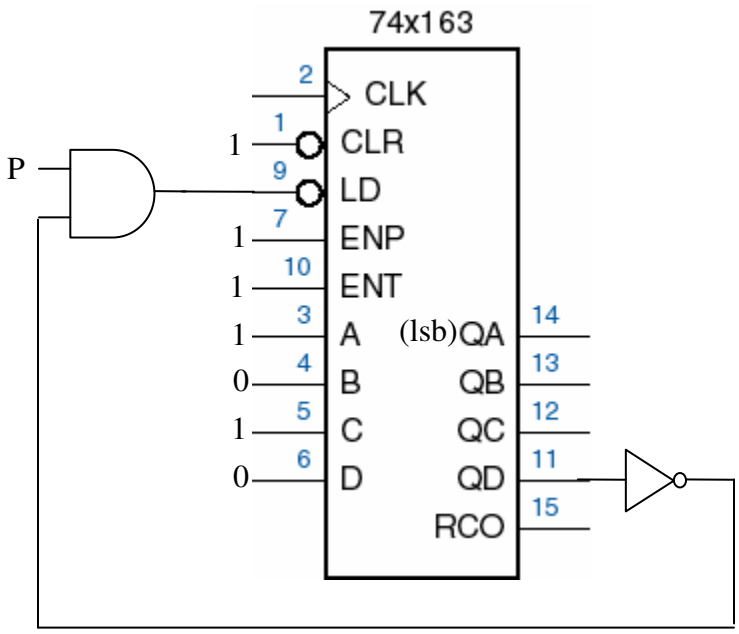
Count sequence: 5-6-7-8-5-6-7-8....
Use the below described reset IC to initially load 5 to the output during T.
Is your counter self correcting? Explain why or why not.



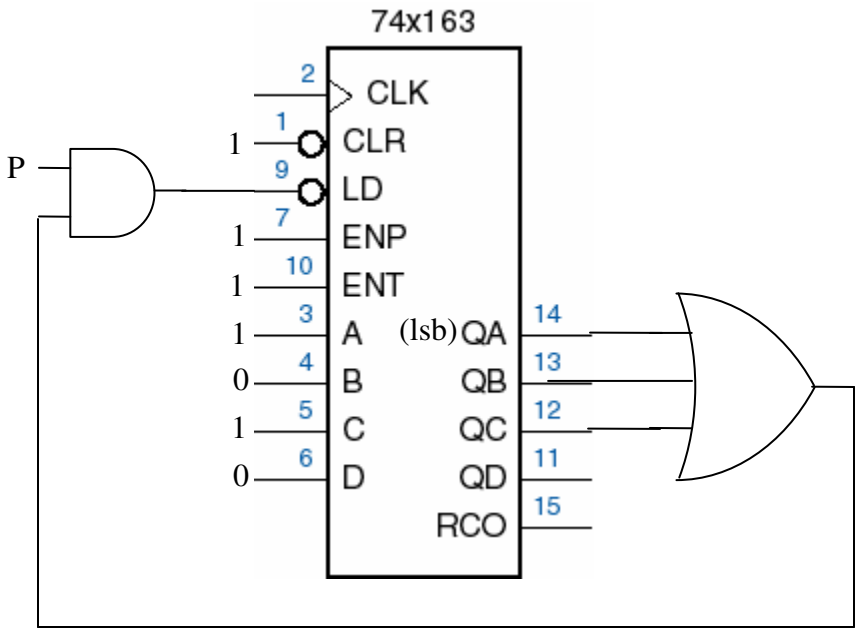
(T is larger than the clock period)



Solution:



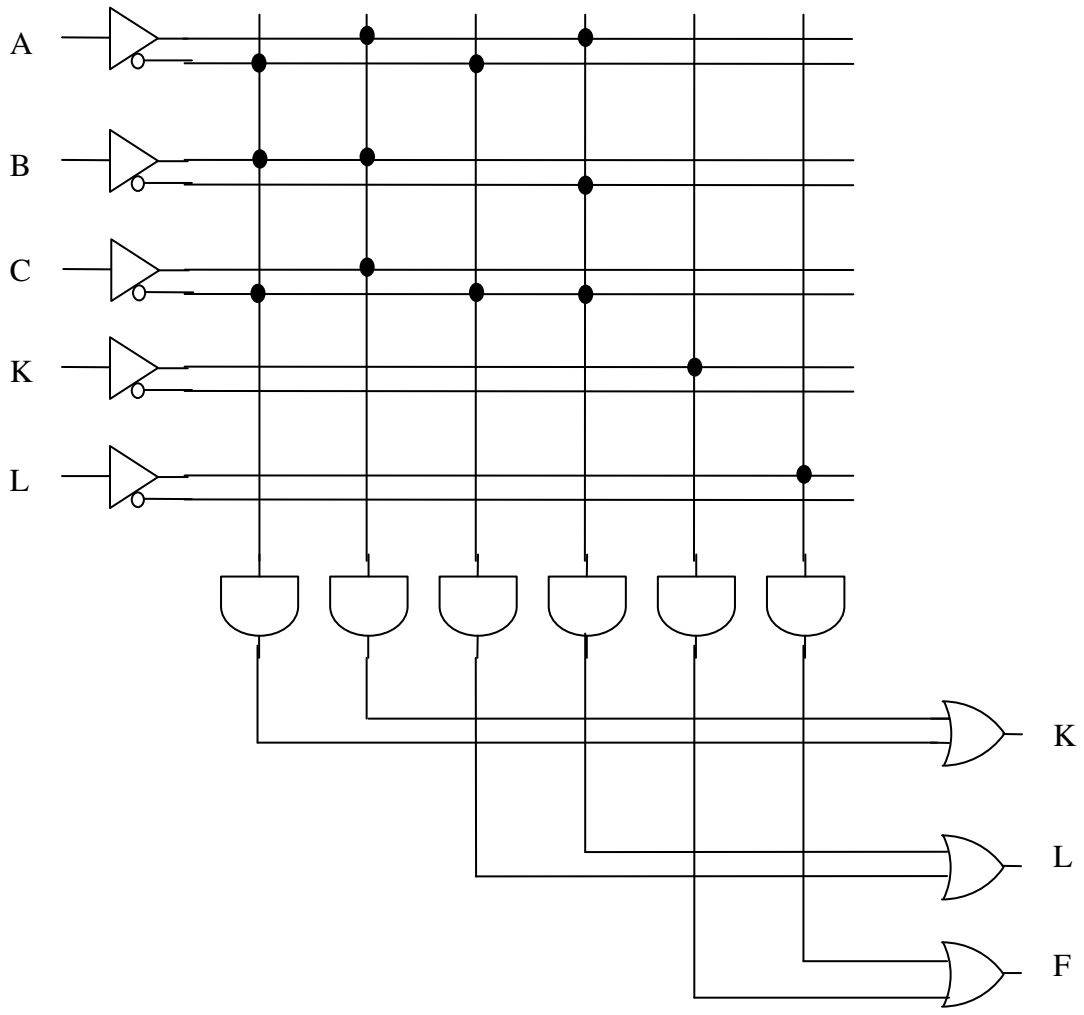
Above circuit is self correcting. If it goes to 0,1,2,3, or 4 then it counts up and eventually reaches 5. If it goes to 9,10,11,12,13,14, or 15 then it loads 5.
or;



Above circuit is self correcting. If it goes to 0 it will load 5. If it goes to 1,2,3, or 4 then it counts up and eventually reaches 5. If it goes to 9,10,11,12,13,14, or 15 then it counts up until it becomes 0 and then loads 5.

Q6. (15 points)
You are to implement the function F = A'BC' + ABC + A'C' + AB'C' using a simple PAL which has 5 inputs and 3 outputs with each OR gate having 2 inputs. The inputs of each OR gate are different from the inputs of the inputs of the other OR gates. Draw the internal structure of the PAL, show the fuses which are to be left intact, and also show all the outside connections. Multilevel implementation is acceptable. Do not attempt to simplify the function. No other gates are allowed.

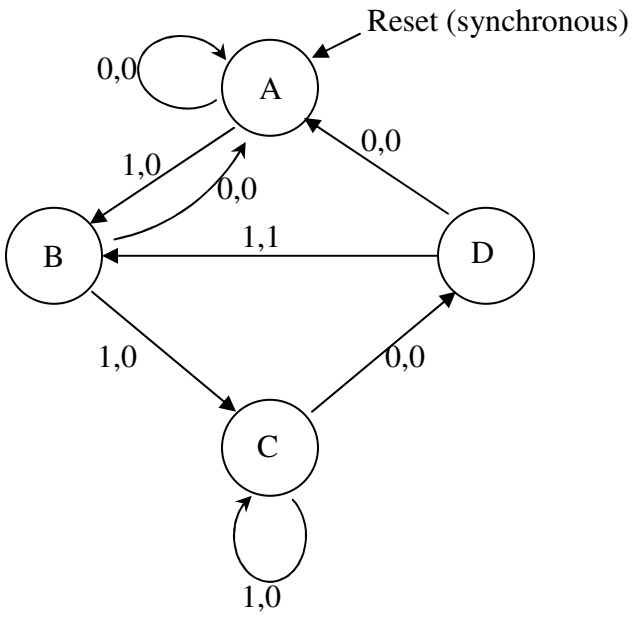
Solution:



Q7. (15 points)
Draw the state/output diagram of the FSM described by the following VHDL code. Is this a Moore or a Mealy FSM? Why? What does this FSM do? Hint: When does the output become '1'?

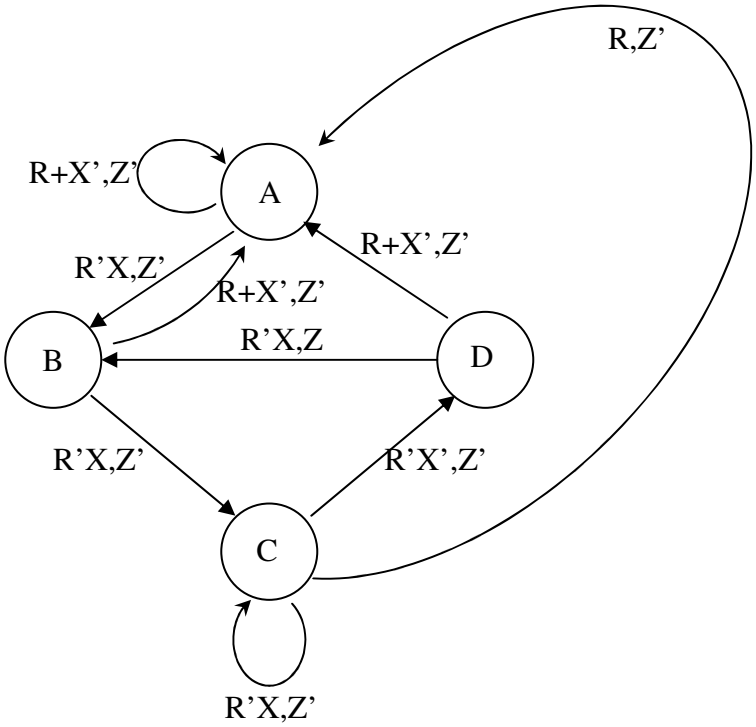
```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity seq_rec is
        port(CLK,RESET,X: in std_logic;
        Z:out std_logic);
end seq_rec;

architecture Behavioral of seq_rec is
        type state_type is(A,B,C,D);
        signal state, next_state :state_type;
begin
state_register: process(CLK,RESET)
begin
        if (RESET='1') then
                state<=A;
        elsif (CLK'event and CLK='1') then
                state<=next_state;
        end if;
end process;
next_state_fuction: process(X,state)
begin
        case state is
                when A=>
                if X='1' then    next_state<=B; else next_state<=A;   end if;
                when B=>
                if X='1' then    next_state<=C; else next_state<=A;   end if;
                when C=>
                if X='1' then    next_state<=C; else next_state<=D;   end if;
                when D=>
                if X='1' then    next_state<=B; else next_state<=A;   end if;
        end case;
end process;
output_function:process(X,state)
begin
        case state is
                when A=> Z<='0';
                when B=> Z<='0';
                when C=> Z<='0';
                when D=> if X='1' then Z<='1'; else Z<='0'; end if;
        end case;
end process;
end Behavioral;
```

Solution:

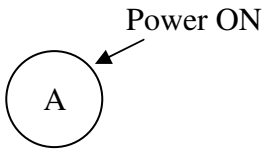


This is a Mealy FSM because output depends also on the present input.
The output is '1' if the present input is '1' and the last received three inputs are "110".

Actually since Reset is synchronous the proper way of drawing the state/output diagram is as follows: (In the following diagram since we have two inputs R and X, where R is Reset, in order to avoid to many arrows we have used expressions on the arrows. On each arrow there are two expressions, first is for the inputs, second is for the output).



Some students have drawn a Power ON signal like



This is not correct because if this were desired then in the VHDL code we should have had

```
signal state : state_type  :=A;
signal next_state : state_type;
```

meaning that the initial (power ON) value of the state is A.