King Fahd University of Petroleum and Minerals
College of Computer Science and Engineering
Computer Engineering Department

COE 202: Digital Logic Design (3-0-3)
Term 181 (Fall 2018)
Final Exam
Saturday, December 15th, 2018

Time: 150 minutes, Total Pages: 10

Name:_____ **ID:**_____ **Section:** _____

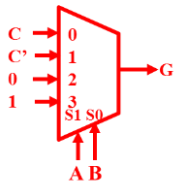
Notes:
- Do not open the exam book until instructed
- **Calculators are not allowed** (*basic, advanced, cell phones, etc*.)
- Answer all questions, show all steps, and clearly state any assumptions you make
- When instructed to use MSI components, only use standard components such as Adders, MUXs, Magnitude Comparators, DeMUXs, Decoders, Encoders (straight or priority). **Label each component and its inputs/outputs clearly.**

Question	Maximum Points	Your Points
1	14	
2	16	
3	16	
4	14	
Total	60	

Key

Question 1. **(14 points)**

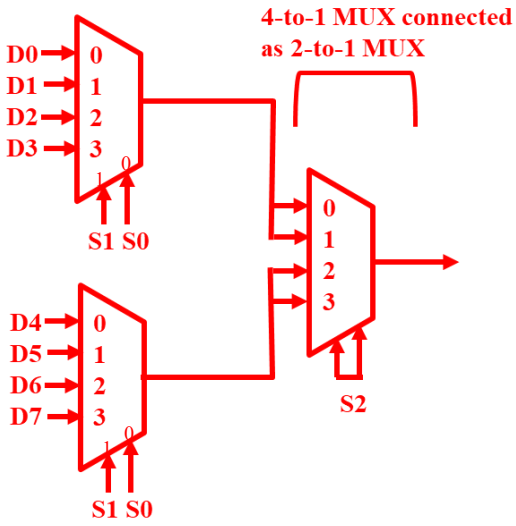
a) Implement the function G (A,B,C) with the following truth table using a single MUX with minimum size and inverters (as needed): **(3 points)**



A	*B*	*C*	*G*	**G**
0	0	0	0	**C**
0	0	1	1	
0	1	0	1	**C'**
0	1	1	0	
1	0	0	0	**0**
1	0	1	0	
1	1	0	1	**1**
1	1	1	1	

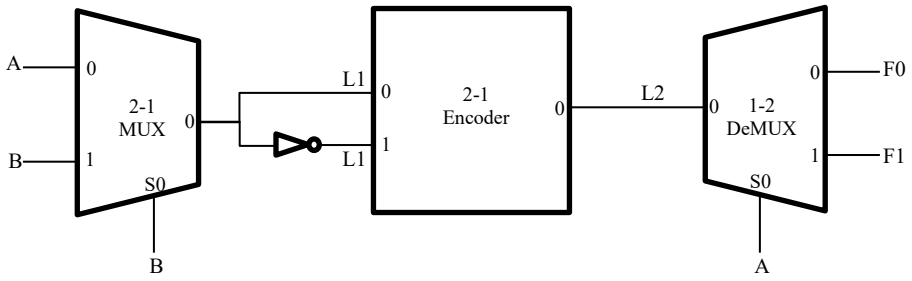
Grading: Every correct input of the MUX=0.5 point. Using larger MUX=−0.5. Missing data line labels −0.5. Missing selection line labels −0.5.

b) Suppose you have three 4-1 MUXs, but you need an 8-1 MUX. Show one possible implementation of the 8-1 MUX using the three 4-1 MUXs you have. **(3 points)**



Grading: Three 4-1 MUXs in the shape of a tree=1 point. Correct selection lines and data lines=1 point. Correct 4-1 as 2-1=1 point. Using incorrect MUX size=−0.5. Missing data line labels −0.5 for every MUX. Missing selection line labels −0.5 for every MUX, except when it does not matter, like the MUX in the right.

c) Consider the circuit shown below: **(2 points)**



Inputs are A and B, and outputs are F0 and F1. Derive the truth table for this circuit. **Hint:** It helps if you evaluate the internal signals, L1 and L2, before evaluating the outputs.

L1 = AB' + BB = A +B, the encoder is just an inverter → L2 = A'B', F0=A'L2=A'B', F1=AL2=0

A B	F0 F1
0 0	1 0
0 1	0 0
1 0	0 0
1 1	0 0

Grading: every row of the truth table is 0.5 point.

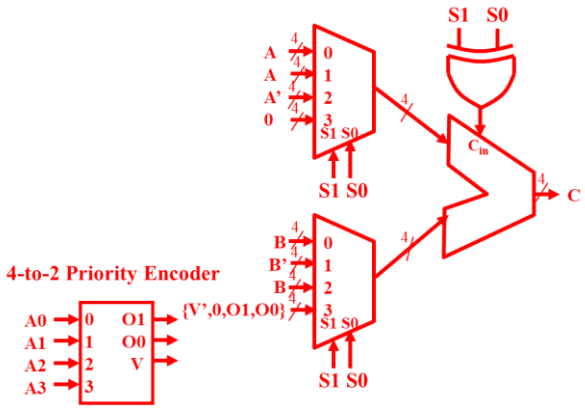
d) A circuit receives two **4-bit** signed numbers in 2's complement representation $\mathbf{A=A_3A_2A_1A_0}$, and $\mathbf{B=B_3B_2B_1B_0}$, two selection inputs S1 and S0, and produces a 4-bit output $\mathbf{C = C_3C_2C_1C_0}$ According to the following table: **(6 points)**

S1	S0	Function
0	0	C = A+B
0	1	C = A-B
1	0	C = B-A
1	1	$C = \begin{cases} 10XX_2 & \text{,if } A=0000_2 \\ \text{Position of most-significant 1 in A} & \text{,if } A\neq 0000_2 \end{cases}$

This table illustrates the last function (when ({S1,S0}=2'b11):

$A_3A_2A_1A_0$	$C_3C_2C_1C_0$
0 0 0 0	1 0 X X
1 X X X	0 0 1 1
0 1 X X	0 0 1 0
0 0 1 X	0 0 0 1
0 0 0 1	0 0 0 0

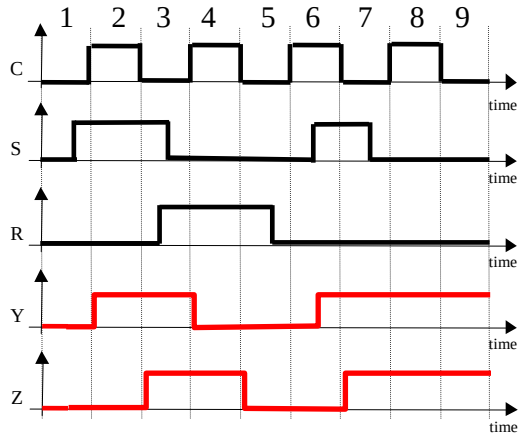
Design the above circuit using only one adder and any other MSI components and logic gates as needed. Note that you can freely use Verilog notation in your diagram, e.g. the concatenation operator, replacing 1000_2 with {1'b1,3'b0} is acceptable.

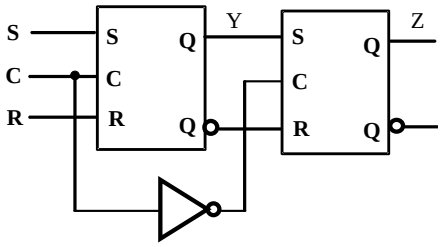


Grading: Every correct function=1.5 points. Every additional adder=−0.5.

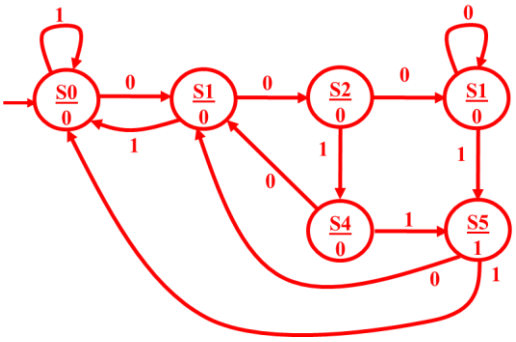
Question 2: **[16 points]**

a) Given the shown Master-Slave configuration of two clocked S-R latches, it is required to complete the Y and Z signals in the following timing diagram. Assume Y and Z are initially equal to zero. Also assume zero propagation delays. **(3 points)**

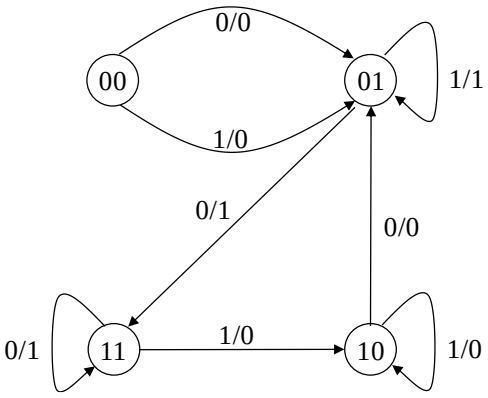




b) Derive the state diagram of a synchronous Moore sequential circuit that receives a serial input Y and produces a serial output Z that is set to 1 when the circuit detects the sequence 00X1, where X represents a 0 or 1. **(6 points)**



c) A sequential circuit has one input X and one output Y. Its state diagram is as shown below. Design the circuit using minimum number of FFs and minimum logic gates. Use the specified state assignment codes. **(6 points)**



e) PS input NS output

A	B	X	A+	B+	Y
0	0	0	0	1	0
0	0	1	0	1	0
0	1	0	1	1	1
0	1	1	0	1	1
1	0	0	0	1	0
1	0	1	1	0	0
1	1	0	1	1	1
1	1	1	1	0	0

BX / A	00	01	11	10
0	0	0	1	1
1	0	0	0	1

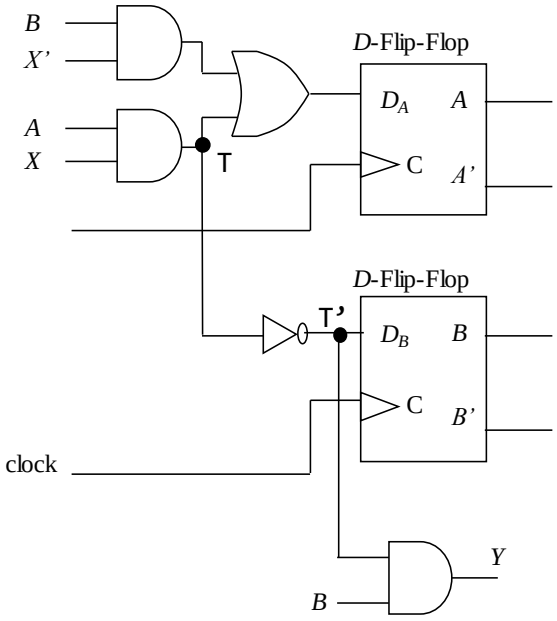
Y = BX'+A'B
= B(X'+A')
= B T'

BX / A	00	01	11	10
0	0	0	0	1
1	0	1	1	1

A+ = BX'+AX
= BX'+T

BX / A	00	01	11	10
0	1	1	1	1
1	1	0	0	1

B+' = AX = T
B+ = A'+X'=T'



Two FFs, 3 AND gates, 1 OR gate, and an inverter!

d) Specify whether the above circuit (in c above) has an unused state and explain why. **(1 point)**

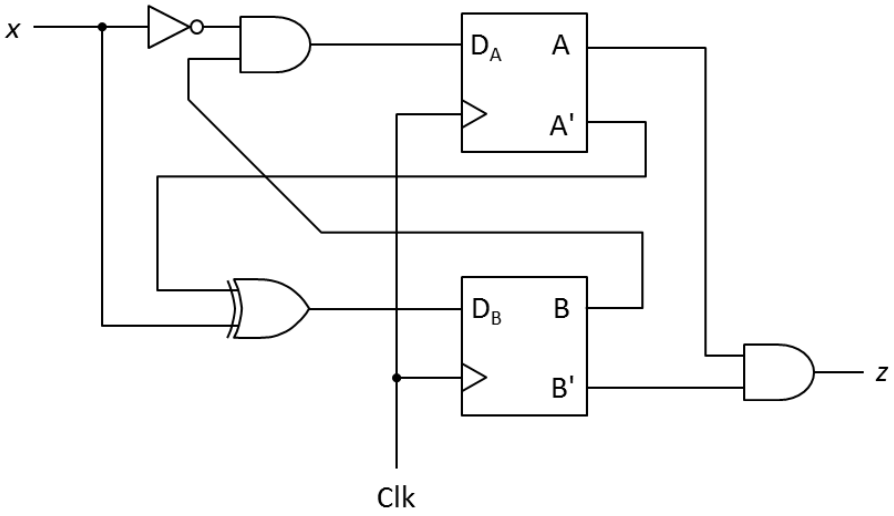
Yes, state 00 is an unused state, no input sequence can get the circuit to that state from any other state.

Question 3: **[16 points]**

Part I: Given the synchronous sequential circuit shown below:

a) Is it a Mealy or Moore sequential circuit and why? **(1 point)**

b) Obtain the state diagram of this circuit. **(6 points)**



Solution:

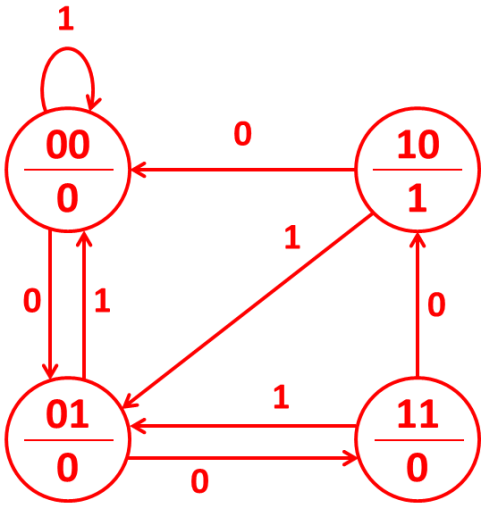
a) Moore sequential circuit because the output *z* does not depend on the input *x*.

b) $D_A = x'B \quad D_B = x \oplus A'$

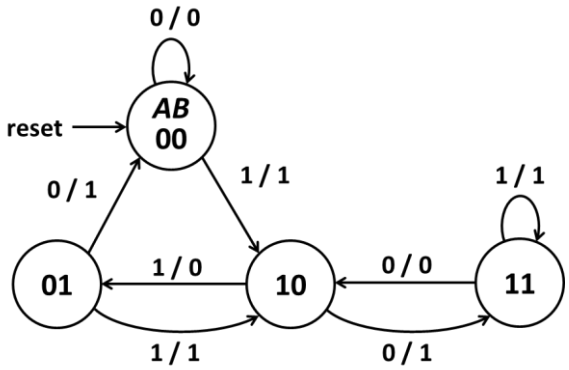
State table is not required (just for clarity)

Present State $A\ B$	Next State when x = 0 $D_A\ D_B$	Next State when x = 1 $D_A\ D_B$	Output z
0 0	0 1	0 0	0
0 1	1 1	0 0	0
1 0	0 0	0 1	1
1 1	1 0	0 1	0

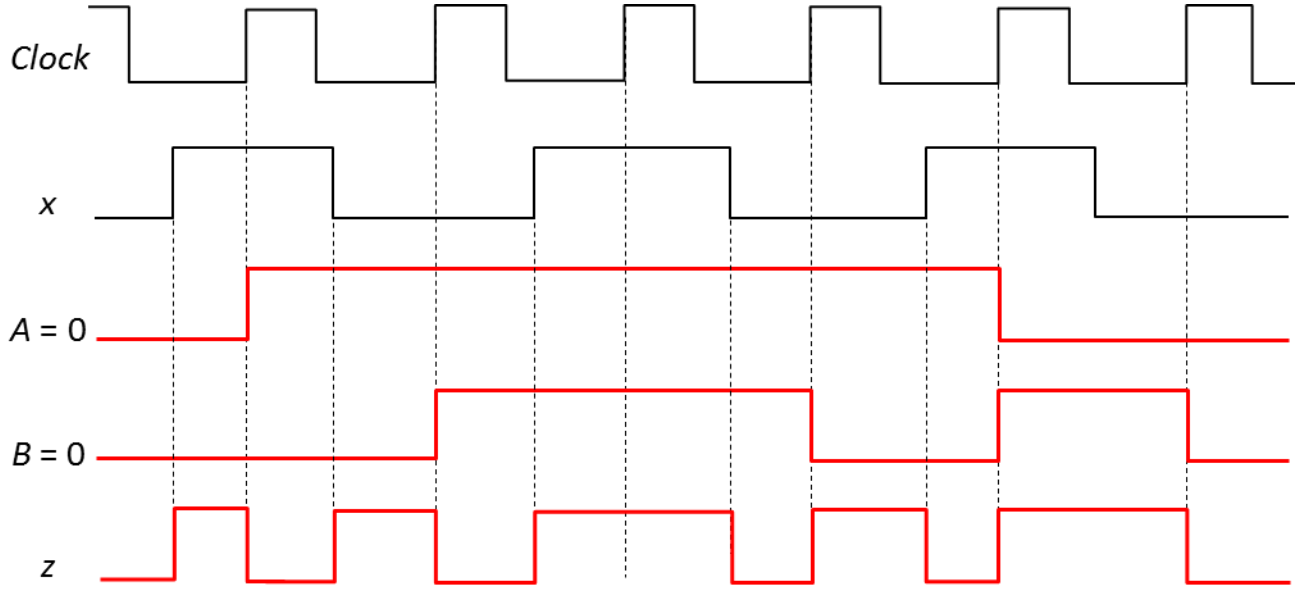
State Diagram:



Part II: The following state diagram is for a sequential circuit with input x, output z, two state variables A and B, **positive** edge-triggered flip-flops, and **asynchronous active HI** reset.



a) Complete the timing diagram below for the above state diagram, showing the waveforms of the state variables A and B, and the output z. Initially, the flip-flops are reset to $A = B = 0$. **(4 points)**



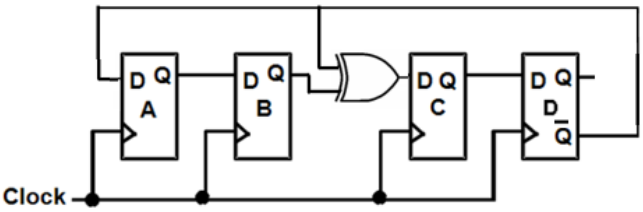
b) Write a behavioral Verilog description of the above state diagram shown in **part II. (5 points)**

```
module Q3_II (input x, clock, reset, output z);
reg [1:0] state;
assign z= ((state==0)||(state==3))&&(x) || (state==1) || (state==2)&&(!x)
;
always @(posedge clock, posedge reset)
  if (reset) state <= 2'b00;
  else case (state)
    0: if (x) state <= 2 ;
    1: if (x) state <= 2 ; else state <= 0 ;
    2: if (x) state <= 1 ; else state <= 3 ;
    3: if (!x) state <= 2 ;
  endcase
endmodule

// Solution 2:

module Q3_II (input x, clock, reset, output reg z);
reg [1:0] state, nxt_state;

always @(posedge clock, posedge reset)
  if (reset) state <= 2'b00;
  else state <= nxt_state

always @(*)     //combinational block for nxt_state and z
  case (state)
    0: if (x)  {nxt_state,z} = 3'b101 ; else {nxt_state,z} = 3'b000 ;
    1: if (!x) {nxt_state,z} = 3'b001 ; else {nxt_state,z} = 3'b101 ;
    2: if (!x) {nxt_state,z} = 3'b111 ; else {nxt_state,z} = 3'b010 ;
    3: if (x)  {nxt_state,z} = 3'b111 ; else {nxt_state,z} = 3'b010 ;
  endcase
endmodule
```

Question 4: **[14 points]**

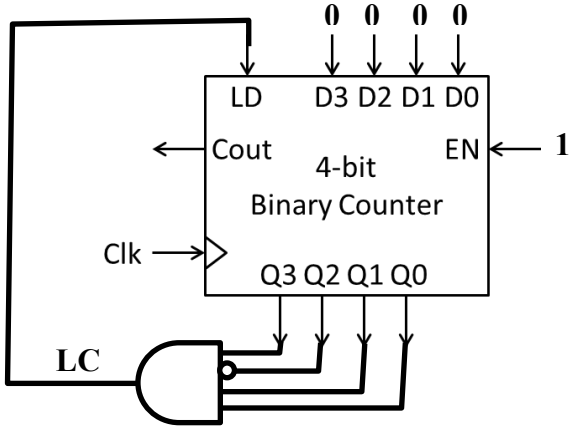
a) The 4-bit shift register shown was initially loaded with **ABCD = 0000**. List in the table below the contents of the register after receiving each of the clock pulses indicated. **(2 points).**

	Register Contents A B C D (Binary)
Initial	0 0 0 0
After Clock Pulse 1	**1 0 1 0**
After Clock Pulse 2	**1 1 1 1**



b) The counter circuit shown below is a modulo **_12__** counter. If the clock frequency is **240** MHz, then the frequency at the ***LC*** node (output of the 4-IP AND gate) is **___20_____** MHz and the frequency of the Q0 output is **___120_____** MHz. **(3 points)**

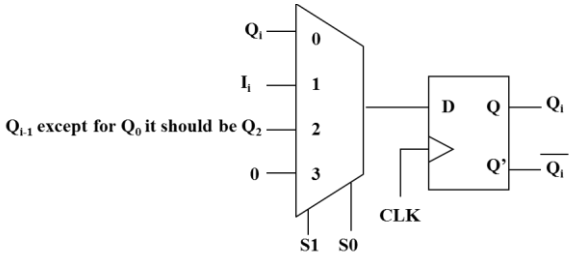
LD	EN	Operation of the counter
0	0	Hold count
0	1	Increment count
1	X	Parallel load



c) Using D flip-flop(s), MUXs, and logic gates only (i.e. **do not use any other MSI components**), design a **3-bit** register with two mode selection inputs **S1** and **S0**. The register should operate according to the following table: **Clearly Label all your components, inputs, and outputs**

(6 points)

S1	S0	Function
0	0	No change
0	1	Parallel load (load inputs $I_2I_1I_0$ into register in parallel)
1	0	Rotate Left (i.e., shift register contents to the lest feeding in the shifted bit out from the last bit location as a serial input to the first location)
1	1	Synchronous clear (i.e. load with zeros) – **the register does not have asynchronous reset**



d) Write a behavioral Verilog description of the register in (f) above. (**3 points)**

```
module rgst (input S1,S0,clk, input [2:0] I, output reg [2:0] Q);
always @(posedge clk)
case ({S1,S0})
1: Q<= I ;
2: Q<={Q[1:0],Q[2]};
3: Q<=0  ;
Default: Q<=Q ;
endcase
endmodule
```