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 172 (Spring 2018)
Final Exam
Sunday May 13, 2018

Time: 120 minutes, Total Pages: 14

Name:_KEY________________________________ ID:_________________ Section: _______

Notes:

- Do not open the exam book until instructed
- **No Calculators are allowed** (*basic, advanced, cell phones, etc.*)
- Answer all questions
- All steps must be shown
- Any assumptions made must be clearly stated

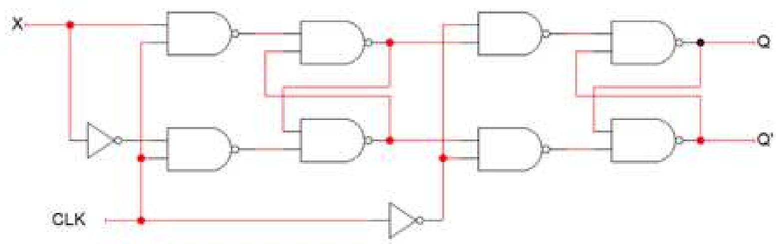
Question	Maximum Points	Your Points
1	13	
2	15	
3	15	
4	10	
5	10	
6	12	
Total	75	

Question 1. **(13 points)**

Answer the following questions by **filling** the spaces with the correct answers:

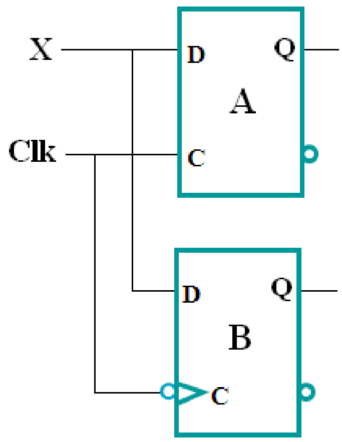
a) (2 points) The following circuit implements an edge-triggered D flip flop. Is it a positive or negative edge-triggered and why?

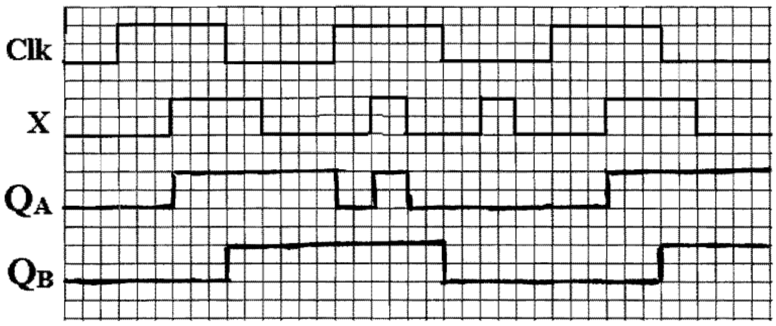
It is a negative edge-triggered D-FF because when CLK=1 the master latch is active capturing input changes and the slave lave is inactive while when CLK=0, the master is turned off and the slave is turned on copying the last value captured by the master.



b) (3 points) In the circuit shown, A is a D-type latch and B is a D-type flip flop. For the input waveforms given for the clock signal (Clk) and the input X, accurately draw the resulting waveforms at outputs **Q_A** and **Q_B**.

Assume that both Q_A and Q_B are initially at 0.

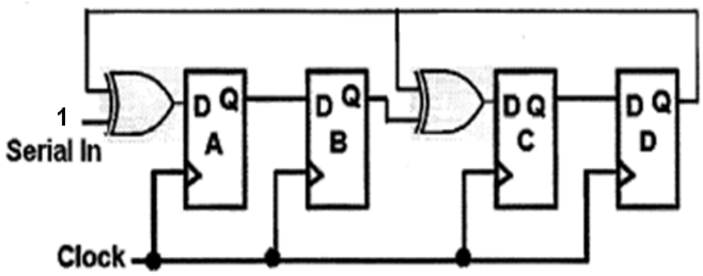




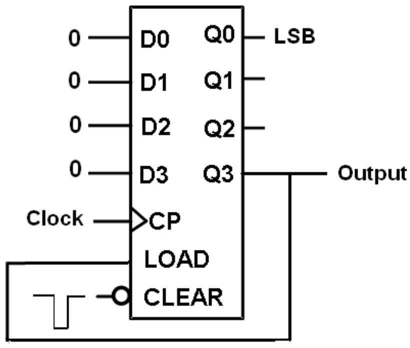
c) (2 points) Given a synchronous sequential circuit with 5 states, the minimum number of flip-flops required to implement the circuit is 3 flip flops and the number of unused states is 3 states.

d) (2 points) For a 3-bit synchronous binary counter (outputs Q_2, Q_1 and Q_0), with input clock frequency of 64 MHZ, the frequency of Q_0 is 32 MHZ and the frequency of Q_2 is 8 MHZ.

e) (2 points) Consider the sequential circuit given below with the 4-bit register ABCD . Initially the register has the contents ABCD = 1001 and the serial input line is kept at 1. After the first rising-edge of the clock, the register contents become ABCD = 0110. After the second rising-edge, the register contents become ABCD = 1011.

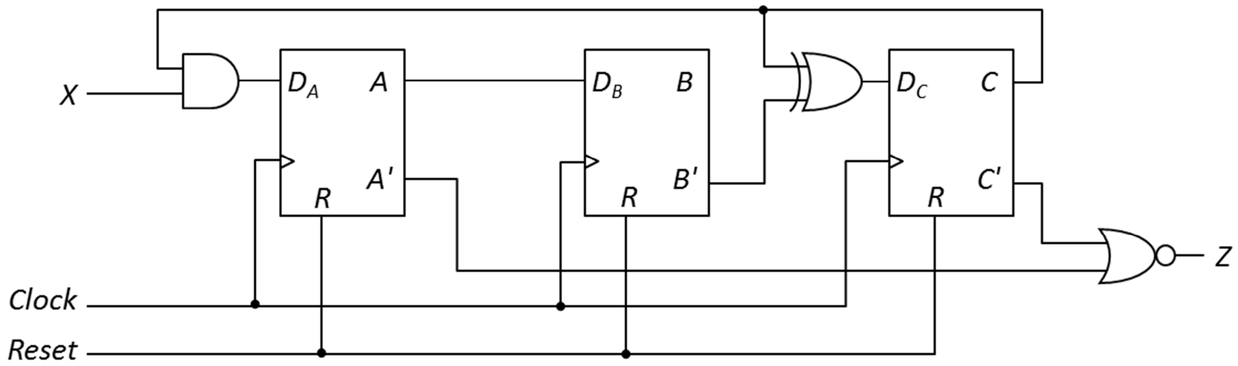


f) (2 points) Consider the given synchronous counter circuit. The LOAD input is synchronous and the Clear (reset to 0) input is asynchronous. Initially one short 0 pulse is applied at the clear input. The circuit implements a modulo-9 counter. With a clock input signal having a frequency of 720 cycles/sec, the frequency at the output signal is 80 cycles/sec.



Question 2. **(15 points)**

Given the following synchronous sequential circuit with input *X* and output *Z* and asynchronous *Reset*:



a) (1 point) Is it a Mealy or Moore sequential circuit and why?
b) (2 points) Write the equations at the inputs of all flip-flops.
c) (1 point) Write the output equation.
d) (5 points) Draw the state table showing the present state, next state, and output *Z*.
e) (1 point) Are there any unused states? Which ones?

Solution:

a) **Moore sequential circuit, because output *Z* does not depend on input *X*.**

b) $D_A = X \cdot C$ $D_B = A$ $D_C = B' \oplus C$

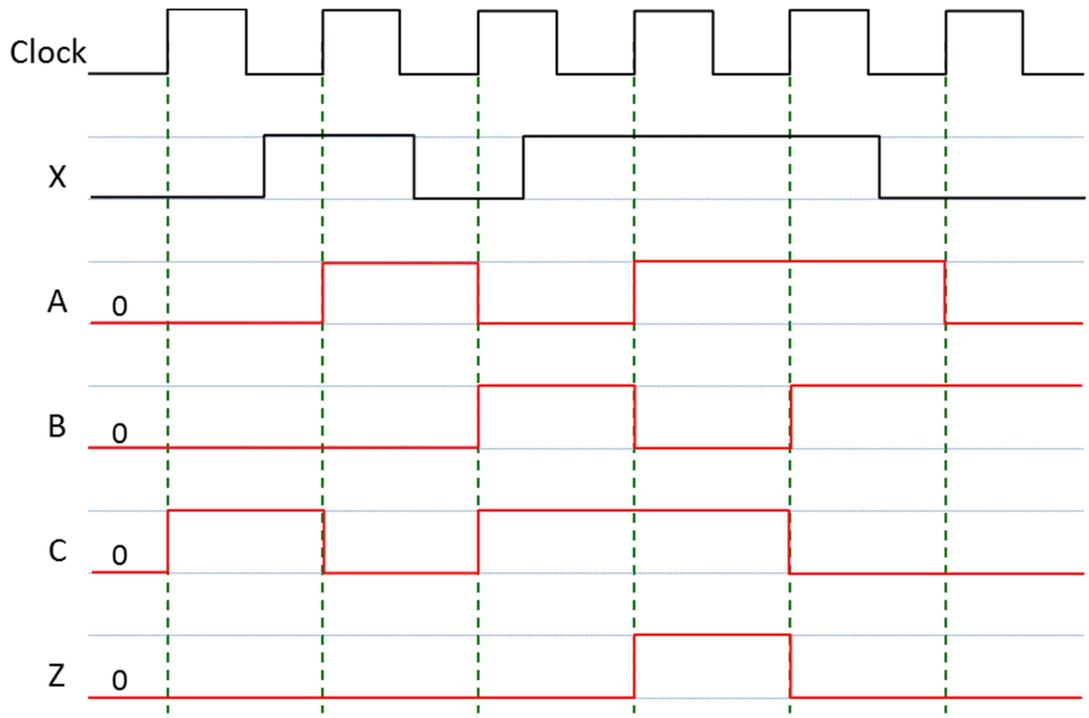
c) $Z = (A' + C')' = A \cdot C$

d)

Present State *A, B, C*	Next State when *X* = 0	Next State when *X* = 1	Output *Z*
0 0 0	0 0 1	0 0 1	0
0 0 1	0 0 0	1 0 0	0
0 1 0	0 0 0	0 0 0	0
0 1 1	0 0 1	1 0 1	0
1 0 0	0 1 1	0 1 1	0
1 0 1	0 1 0	1 1 0	1
1 1 0	0 1 0	0 1 0	0
1 1 1	0 1 1	1 1 1	1

e) **State 111 is unused. It is unreachable from the other states.**

f) (5 points) Complete the timing diagram, starting at the initial state (*ABC* = 000). Note that all the flip-flops are positive edge-triggered.



Question 3. **(15 points)**

The state transition table below is for a sequential circuit with one input **X** and one output **Y**. The circuit has two state variables **A** and **B,** and **synchronous** input Reset that resets the circuit to state AB=**01** when Reset=1:

	Present State	Next State		Output	
		X=0	X=1	X=0	X=1
	A B	A B	A B	Y	Y
	0 0	0 0	0 1	0	1
Reset State →	0 1	0 0	1 0	1	0
	1 0	0 1	1 0	0	1

a) (9 points) Implement the sequential circuit using minimum number of logic gates and rising-edge triggered D-FFs and draw the logic diagram of the implemented circuit. The circuit should have **synchronous** reset that resets it to state **01**.

	00	01	11	10
0			?	
1		1	?	1

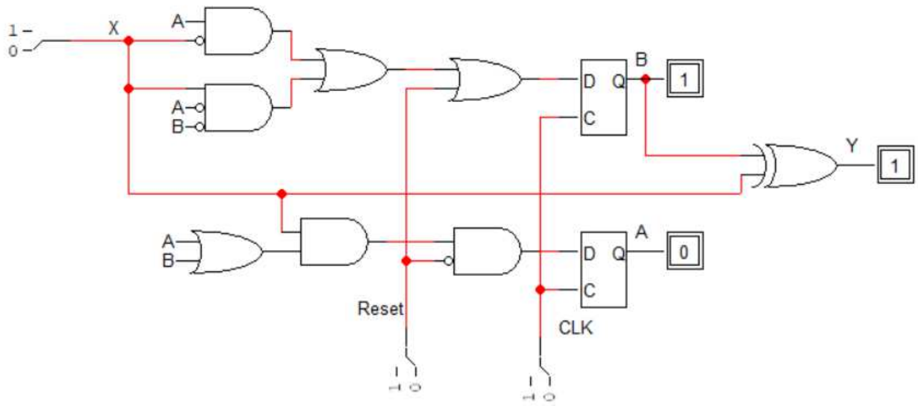
$D_A = X A + X B = X (A + B)$

	00	01	11	10
0			?	1
1	1		?	

$D_B = X' A + X A'B'$

	00	01	11	10
0		1	?	
1	1		?	1

$Z = X' B + X B' = X \oplus B$



b) (6 points) It is required to write a **behavioral** Verilog module to model the given sequential circuit. Complete the given Verilog code to achieve that.

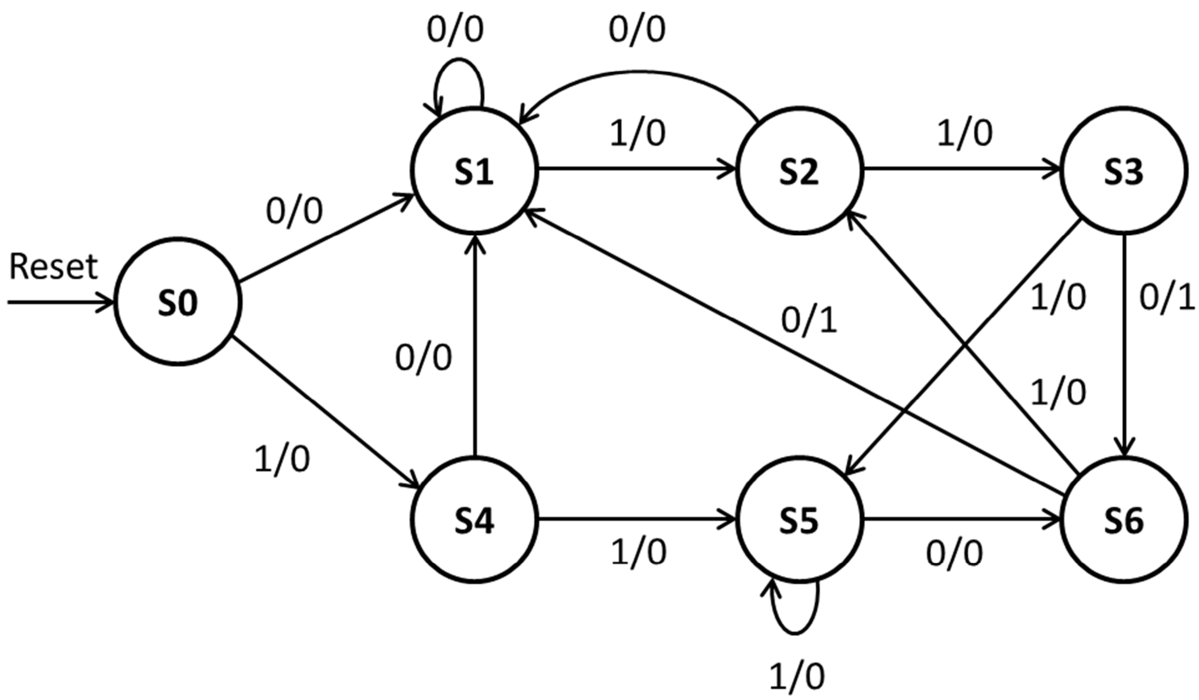
```
module FinalQ3 (output reg Y, input X, Reset, CLK);

reg [1:0] state, next_state;

always @(posedge CLK)  //synch reset
  if (Reset) state <= 2'b01;
  else state <= next_state;  // the state transition

always @(state, X) begin //comb. Logic
                 Y = 0;        next_state=2'b00; //Default value of Y and next_state
   case (state)
      2'b00:  if (X)  begin next_state=2'b01;  Y=1; end
      2'b01:  if (X)  next_state=2'b10; else Y=1;
      2'b10:  if (X)  begin next_state=2'b10;  Y=1; end else next_state=2'b01;
   endcase
end

endmodule
```

Question 4. **(10 points)**

Draw the state diagram of a Mealy sequence detector with input *X*, output *Z*, and a *reset* input, that detects the input sequences **0110** or **1100**. The sequence detector should detect overlapping sequences.
The following is an example of an input/output stream, starting at the initial state:

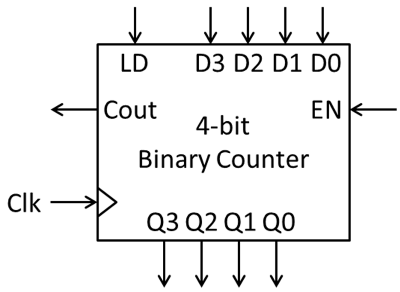
Clock Cycle	0	1	2	3	4	5	6	7	8	9	10	11	12	13
Input *X*	0	1	1	0	0	1	1	0	1	1	0	1	1	1
Output *Z*	0	0	0	1	1	0	0	1	0	0	1	0	0	0

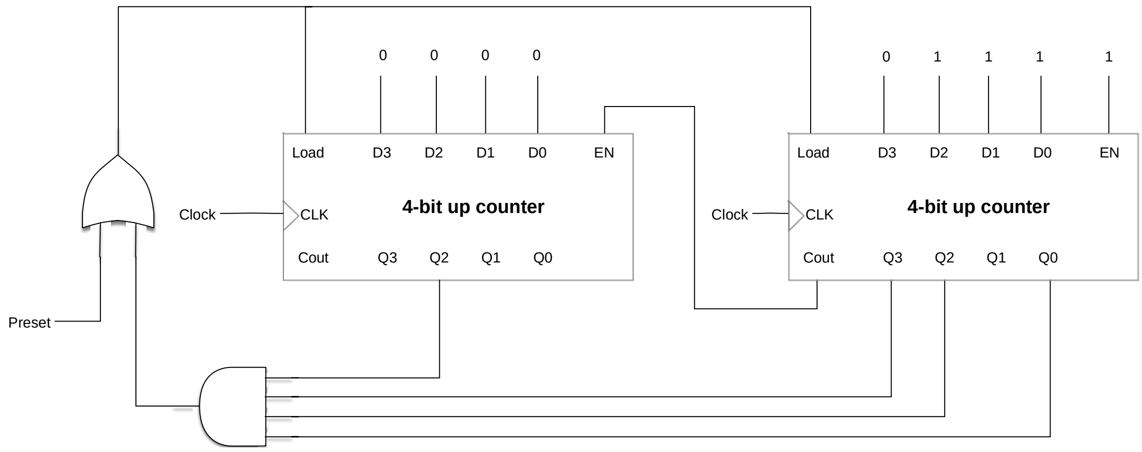
Solution:



Question 5. **(10 points)**

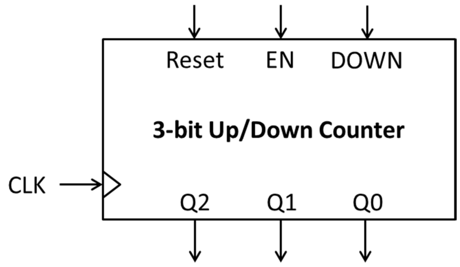
a) (5 points) Using 4-bit synchronous binary counters, with synchronous load input (LD), enable input (EN), and Cout output, as shown in the graphic symbol below, show how to implement a counter that counts up from 7 to 77 and then back to 7, repeatedly. You may use multiple instances of the 4-bit counter and additional gates as needed to implement the 7 to 77 counter. Include an external preset input that initializes the counter to 7.

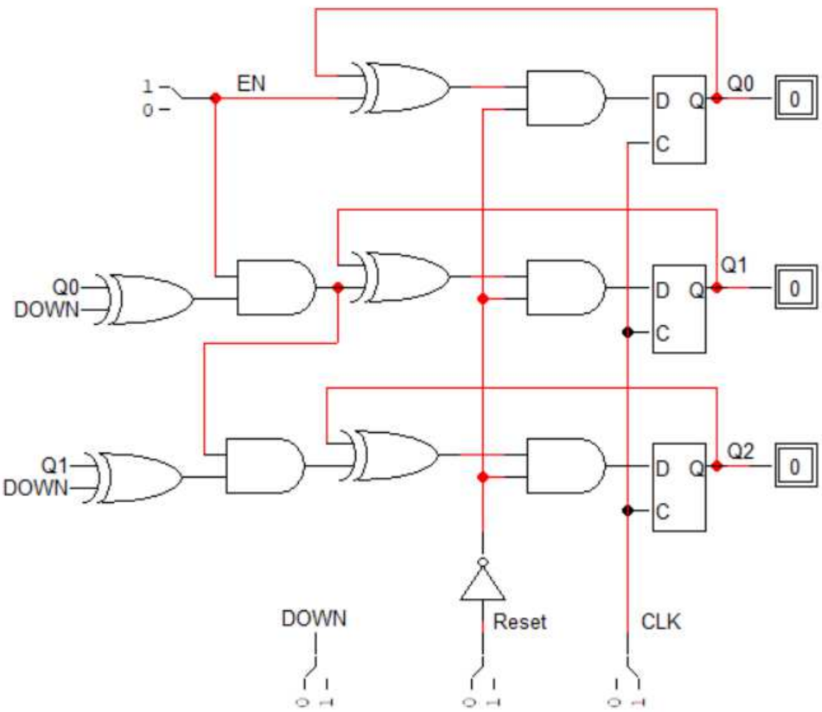




b) (5 points) Show how to implement a 3-bit synchronous up/down counter using D-FFs and **minimal** additional combinational logic elements. The counter should have three **synchronous** control inputs (Reset, EN, and DOWN) and should function according to the following table (graphic symbol also shown below). Please note that you are not allowed to design the counter based on the use of state diagram.

Reset	**EN**	**DOWN**	**Function**
1	X	X	Reset (Q=0)
0	0	X	No change
0	1	0	Count up (Q=Q+1)
0	1	1	Count down (Q=Q-1)

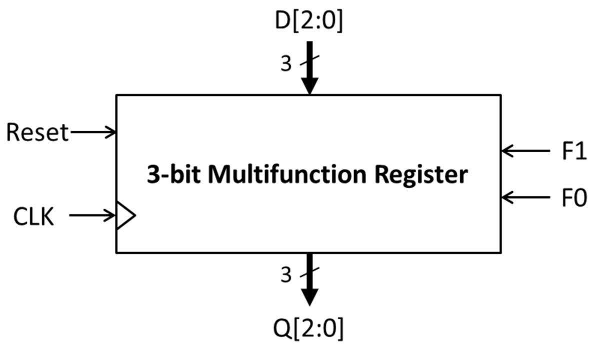




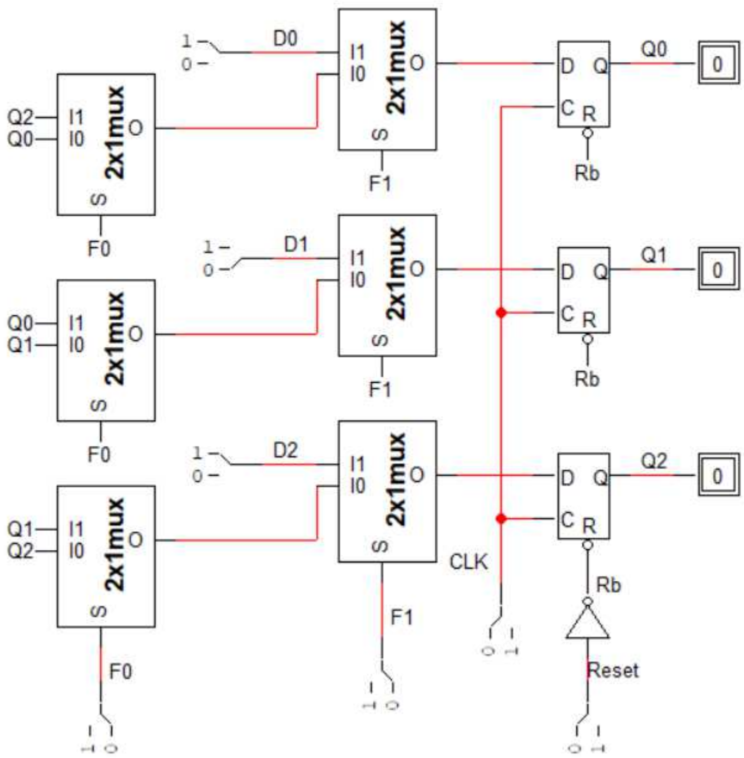
Question 6: **(12 marks)**

Using positive edge-triggered D-FFs and any other components, you are required to design a 3-bit multi-function register with direct **asynchronous** reset. The register operates according to the following function table:

Reset	F1	F0	Function
1	X	X	Asynchronous reset: Q = 0
0	0	0	No change
0	0	1	Rotate left: Q2<=Q1, Q1<=Q0, Q0<=Q2
0	1	X	Load: Q = D



a) (4 points) Draw the circuit diagram of this register.



b) (4 points) The following Verilog module is supposed to model the register described above. Complete the module by filling the blanks in the following code:

```
// mfr: multi-function register
module mfr (input [2:0] D, input reset, clk, F1, F0,
            output reg [2:0] Q);
always @(  posedge clk, posedge reset      )
begin

 if (reset) Q <= 0;
 else if (F1) Q <= D;
 else if (F0) Q <= {Q[1:0],Q[2]};


end

endmodule
```

c) (4 points) Write a test bench module to verify the functionality of Module **`mfr`**. Use the following test patterns for verification:

Clock Cycle	**1**	**2**	**3**	**4**
Reset	1	0	0	0
F1	X	0	1	0
F0	X	0	0	1
D[2:0]	X	X	3	X

Complete the missing parts of the following test bench module.

```
module tb;
reg clk, rst, F1, F0;
reg  [2:0] D;
wire [2:0] Q;
// instantiate Module mfr below this line
mfr mfr1(D, rst, clk, F1, F0, Q);

// In the space provided below, setup the clock
// and apply the test patterns shown above.

always #5 clk = ~clk;

initial begin

 rst=1; clk=0;
 #10 rst=0;
 @(negedge clk) F1=1'b0; F0=1'b0;
 @(negedge clk) F1=1'b1; F0=0; D=3'b011;
 @(negedge clk) F1=1'b0; F0=1'b1;

end

endmodule
```