***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 161 (Fall 2016)**

**Final Exam**

**Wednesday, December 11, 2017**

**Time: 120 minutes, Total Pages: 14**

**Name:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ ID:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Section: \_\_\_\_\_\_\_**

**Notes:**

Do not open the exam book until instructed

**Calculators are not 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** | **16** |  |
| **2** | **7** |  |
| **3** | **6** |  |
| **4** | **15** |  |
| **5** | **9** |  |
| **6** | **6** |  |
| **7** | **6** |  |
| **8** | **15** |  |
| **9** | **10** |  |

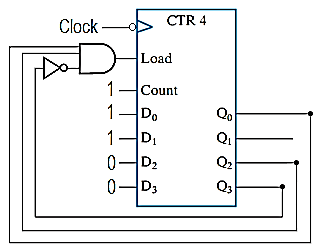
|  |  |  |
| --- | --- | --- |
| **Total** | **90** |  |

**Question 1: Fill in the Spaces: (Show all work needed to obtain your answer) (16 points)**

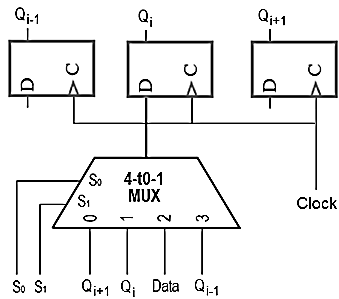
1. A high value to the two R and S inputs of the NAND-gate latch will **\_\_No Change\_\_\_** (Change/No Change) the state/output of the latch. **(1 point)**
2. Asynchronous reset to a flip flop doesn’t depend on the clock input **\_\_ True \_\_** (True/False).

**(1 point)**

1. Given a synchronous sequential circuit with 9 states, the minimum number of flip flops required to implement the circuit is **\_\_4\_\_** flip flops and the number of unused states is **\_\_\_7\_\_\_** states. **(2 points)**



1. The following circuit shows a parallel load binary counter, the range of the counter is **\_\_3\_\_** to **\_\_5\_\_**. **(2 points)**
2. The below figure shows connections to the ***D*** input of stage ***i*** in a multi-function register of D-type flip flops. Study the circuit and fill in the missing information in the table below (empty slots) **only for supported register functions**. **(2 points)**



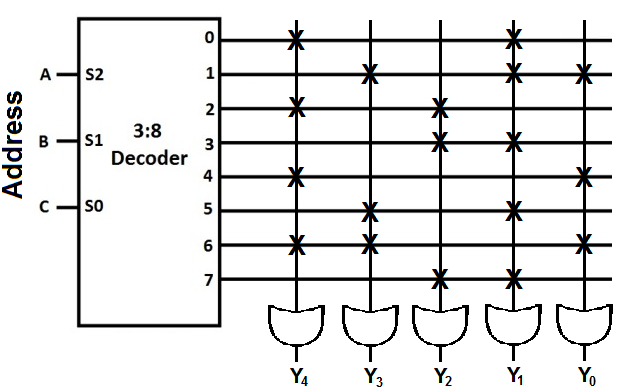
Qi+1

Qi-1

Qi

S1 S0

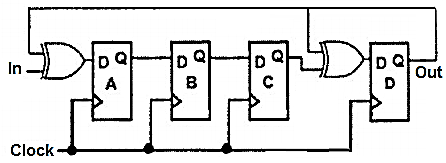
|  |  |
| --- | --- |
| **S1 S0** | **Register Function**  **(where applicable)** |
| **00** | Shift right |
| **--** | Clear register |
| **01** | No change in output |
| **10** | Load Data input |



1. In the ROM circuit shown, **X** indicates a connection. At address ***ABC = 110***, the ROM stores the data ***Y4Y3Y2Y1Y0 =*** **\_\_\_\_11001\_\_\_\_**.

**(1 point)**

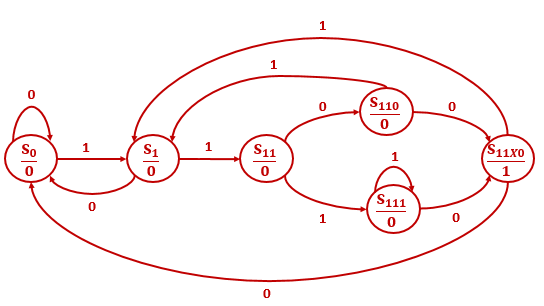
1. For a 4-bit synchronous binary counter (outputs Q3, Q2, Q1 and Q0), with input clock frequency of 48 MHZ, the frequency of Q1 is **\_\_\_12\_\_\_** MHZ and the frequency of Q3 is **\_\_\_\_3\_\_\_\_** MHZ. **(2 points)**
2. Two RS level sensitive latches and one inverter are used to make an edge triggered flip flop which can be used to store up to two bits **\_\_ False \_\_** (True/False). **(1 point)**
3. Consider the below 4-bit register. If the initial register contents (Q outputs) ***ABCD*** are ***0111*** and the serial input is kept at 1, show the contents ***ABCD*** ***=*** **\_\_\_1001\_\_\_** of the register after two clock pulses. **(2 points)**



1. A state machine with two inputs and three outputs will have **\_\_\_4\_\_\_** (how many) arcs going out from each state to any other states. **(1 point)**
2. For any given problem/requirement, the number of states required by a Mealy machine or a Moore machine is always the same **\_\_\_ False \_\_\_** (True/False). **(1 point)**

**Question 2:**   **(7 points)**

Derive the state diagram of a synchronous ***Moore*** sequential circuit that receives a serial input ***Y*** and produces a serial output ***F*** that is set to **1** when the circuit detects the sequence **11X0**, where **X** represents don’t care.



**Question 3:** **(6 points)**

Design a combinational circuit using a ROM. The circuit accepts a 3-bit number ***X = X2X1X0*** and generates an output binary number ***Y*** equal to ***3X + 4***. The ROM should contain a minimum number of columns. Fill the truth table and ROM table below and draw the block diagram.

|  |  |
| --- | --- |
| **Truth Table** | **ROM Table** |
| |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | | ***Inputs*** | | | ***Outputs*** | | | | | | **X2** | **X1** | **X0** | **Y4** | **Y3** | **Y2** | **Y1** | **Y0** | | 0 | 0 | 0 | **0** | **0** | **1** | **0** | **0** | | 0 | 0 | 1 | **0** | **0** | **1** | **1** | **1** | | 0 | 1 | 0 | **0** | **1** | **0** | **1** | **0** | | 0 | 1 | 1 | **0** | **1** | **1** | **0** | **1** | | 1 | 0 | 0 | **1** | **0** | **0** | **0** | **0** | | 1 | 0 | 1 | **1** | **0** | **0** | **1** | **1** | | 1 | 1 | 0 | **1** | **0** | **1** | **1** | **0** | | 1 | 1 | 1 | **1** | **1** | **0** | **0** | **1** | | |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | | ***Inputs*** | | | ***Outputs*** | | | | **X2** | **X1** | **X0** | **F2** | **F1** | **F0** | | 0 | 0 | 0 | **0** | **1** | **0** | | 0 | 0 | 1 | **0** | **1** | **1** | | 0 | 1 | 0 | **1** | **0** | **1** | | 0 | 1 | 1 | **1** | **1** | **0** | | 1 | 0 | 0 | **0** | **0** | **0** | | 1 | 0 | 1 | **0** | **0** | **1** | | 1 | 1 | 0 | **0** | **1** | **1** | | 1 | 1 | 1 | **1** | **0** | **0** | |
| **Block Diagram** | |
|  | |

**Question 4: (15 points)**

Given the following state table:

**a)** Obtain minimal sum-of-products equations for the next state and output. **(12 points)**

**b)** Draw the circuit diagram. **(3 points)**

|  |  |  |  |
| --- | --- | --- | --- |
| **Present State** | **Next State** | | **Output** |
| **A B C** | **x = 0** | **x = 1** | **z** |
| **0 0 0** | **0 0 1** | **0 0 0** | **0** |
| **0 0 1** | **0 1 0** | **0 0 0** | **1** |
| **0 1 0** | **0 1 1** | **0 0 0** | **1** |
| **0 1 1** | **1 0 0** | **0 0 0** | **1** |
| **1 0 0** | **1 0 1** | **0 0 1** | **0** |
| **1 0 1** | **1 0 1** | **0 1 1** | **1** |

**Part a)**

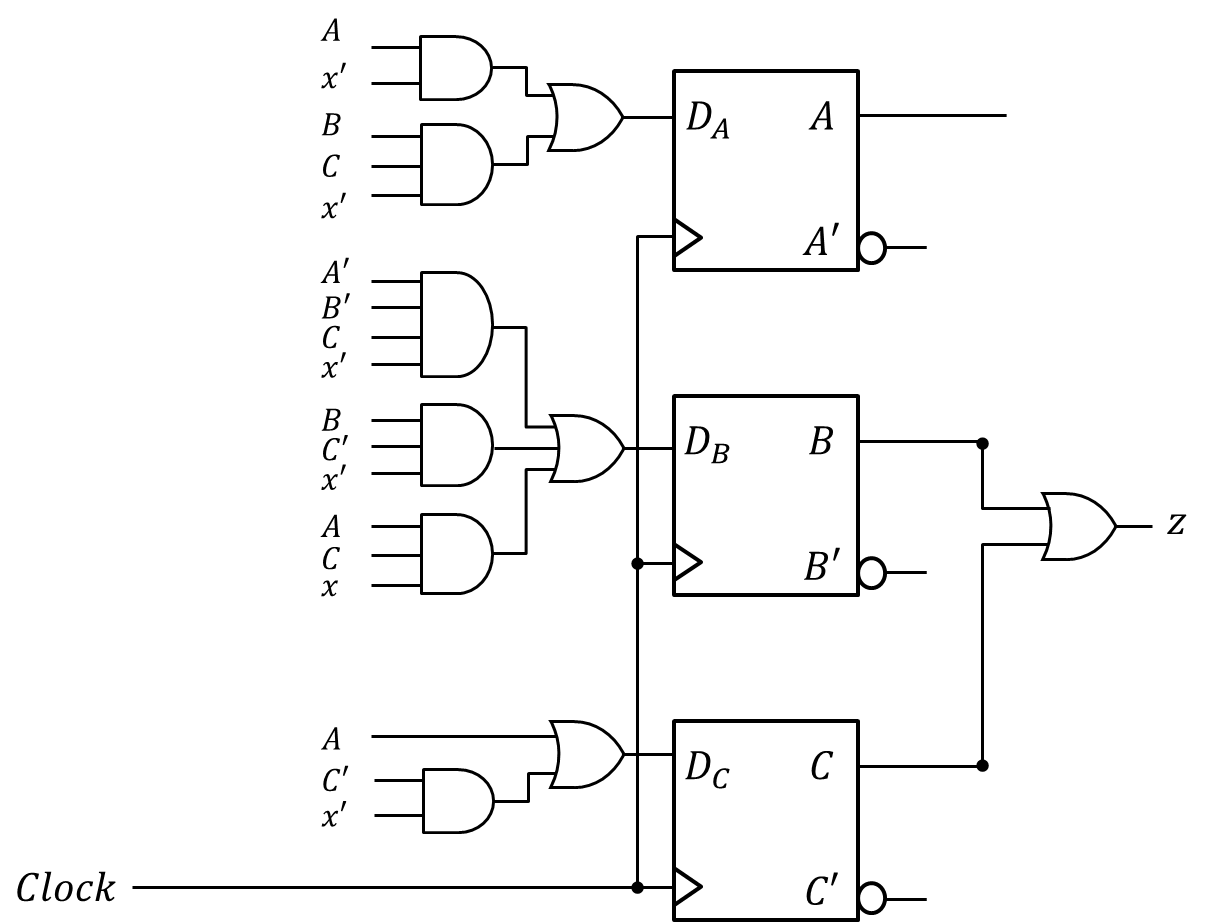
**Solution1: Using Don't Care for the unused states**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **K-Map for DA** | | | | **K-Map for DB** | | | | **K-Map for DC** | | | | **K-Map for z** | |
|  | **C x** | | | | **C x** | | | | **C x** | | | | **C** | |
| **A B** | **0 0** | **0 1** | **1 1** | **1 0** | **0 0** | **0 1** | **1 1** | **1 0** | **0 0** | **0 1** | **1 1** | **1 0** | **C=0** | **C=1** |
| **0 0** |  |  |  |  |  |  |  | **1** | **1** |  |  |  |  | **1** |
| **0 1** |  |  |  | **1** | **1** |  |  |  | **1** |  |  |  | **1** | **1** |
| **1 1** | **X** | **X** | **X** | **X** | **X** | **X** | **X** | **X** | **X** | **X** | **X** | **X** | **X** | **X** |
| **1 0** | **1** |  |  | **1** |  |  | **1** |  | **1** | **1** | **1** | **1** |  | **1** |

**Solution2: Forcing transition from the unused states to state 000**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **K-Map for DA** | | | | **K-Map for DB** | | | | **K-Map for DC** | | | | **K-Map for z** | |
|  | **C x** | | | | **C x** | | | | **C x** | | | | **C** | |
| **A B** | **0 0** | **0 1** | **1 1** | **1 0** | **0 0** | **0 1** | **1 1** | **1 0** | **0 0** | **0 1** | **1 1** | **1 0** | **C=0** | **C=1** |
| **0 0** |  |  |  |  |  |  |  | **1** | **1** |  |  |  |  | **1** |
| **0 1** |  |  |  | **1** | **1** |  |  |  | **1** |  |  |  | **1** | **1** |
| **1 1** |  |  |  |  |  |  |  |  |  |  |  |  | **X** | **X** |
| **1 0** | **1** |  |  | **1** |  |  | **1** |  | **1** | **1** | **1** | **1** |  | **1** |

**Part b) Circuit Diagram for Solution 1**



**Question 5:** Consider the following sequential circuit:  **(9 points)**



**a)** Provide a state table for the given circuit showing the Present State, the input **X**, the Next State, and the output **Z**. **(8 points)**



|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| *QA* | *QB* | *X* | *QA+* | *QB+* | *Z* |
| 0 | 0 | 0 | 0 | 1 | 0 |
| 0 | 0 | 1 | 1 | 1 | 0 |
| 0 | 1 | 0 | 1 | 0 | 1 |
| 0 | 1 | 1 | 0 | 0 | 1 |
| 1 | 0 | 0 | 1 | 0 | 1 |
| 1 | 0 | 1 | 0 | 0 | 1 |
| 1 | 1 | 0 | 0 | 0 | 1 |
| 1 | 1 | 1 | 1 | 0 | 1 |

**b)** Is the circuit type *Mealy* or *Moore*? Justify your answer. **(1 point)**

Moore since Z = Π*M*(0,1) = (*QA* + *QB* +*X*)(*QA* + *QB* + *X* **’**) = *QA* + *QB* which is a function of the present states only. This is also clear from the state table obtained in part (a).

**Question 6: (6 points)**

Consider the following state table. Assume that the initial state of the circuit implementation of the given state table is (*QAQB* = 10). Draw the waveforms of *QA*, *QB*, and *Z* for the given 2 clock cycles in response to the shown applied input *X*. ***Ignore propagation delays, setup times, and hold times. Assume that the circuit uses rising edge-triggered D-FF(s)***.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Present State | | *X* | Next State | | *Z* |
| *QA* | *QB* | *DA* | *DB* |
| 0 | 0 | 0 | 0 | 1 | 1 |
| 0 | 0 | 1 | 1 | 1 | 0 |
| 0 | 1 | 0 | 1 | 0 | 0 |
| 0 | 1 | 1 | 0 | 0 | 0 |
| 1 | 0 | 0 | 1 | 0 | 0 |
| 1 | 0 | 1 | 0 | 0 | 1 |
| 1 | 1 | 0 | 0 | 0 | 1 |
| 1 | 1 | 1 | 1 | 0 | 1 |



**Question 7: (6 points)**

Use minimal external gates and as many of the following counter as necessary to design a counter that counts from **1** to **52** and then repeats. Note that the operation of the provided counter is according to the table to the right of the counter. Note also that CO stands for Carry-output which gets set to 1 when the maximum count of 15 is reached. Assume that the counter is initially loaded with the value 1. Properly label (Q0 – Q7) and (D0 – D7) of the designed counter.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | |  |  |  | | --- | --- | --- | | ***LOAD*** | ***EN*** | ***Operation*** | | 0 | 0 | Hold count | | 0 | 1 | Increment count | | 1 | X | Parallel load | |



**Question 8: (15 points)**

Using D-FFs and any other components, design a 4-bit rotator register with direct asynchronous

reset that has two control inputs; Load and Rot with the functionality shown in the table below:

Reset

CLK

Load

Rot

**Q[3:0]**

**D[3:0]**

**4**

**4**

|  |  |  |  |
| --- | --- | --- | --- |
| Reset | Load | Rot | Function |
| 0 | 0 | 0 | No Change (Q stay as is) |
| 1 | X | X | asynchronous reset: Q = 0 |
| 0 | 1 | X | Load: Q = D |
| 0 | 0 | 1 | Rotate Left: Q3=Q2, Q2=Q1,Q1=Q0, Q0=Q3 |

**a)** Draw the circuit diagram **(5 points)**

**D Q**

R

**0 1 2 3**

**S1 S0**

**Load Rot**

**Reset**

**D1**

**D Q**

R

**0 1 2 3**

**S1 S0**

**Load Rot**

**Reset**

**D3**

**Q2**

**D Q**

R

**0 1 2 3**

**S1 S0**

**Load Rot**

**Reset**

**D2**

**Q1**

**Q0**

**Q1**

**Q2**

**Q3**

**Q1**

**Q0**

**D Q**

R

**0 1 2 3**

**S1 S0**

**Load Rot**

**Reset**

**D0**

**Q3**

**Q0**

**b)** Write a **behavioral** Verilog description of the above rotator register **(6 points)**

module rotator4 (input [3:0] D, reset, clk, load, rot, output reg [3:0] Q);

always @ (posedge clk, posedge reset)

if (reset) Q<= 0 ;

elseif (load) Q<=D ;

else if (rot) Q <= {Q[2:0],Q[3]} ;

endmodule

**c)** Write a test bench to test the above rotator register. Let the clock cycle = 20 time units. First, reset the register, then load the register with **1010**, then do nothing for two clock cycles, then rotate the register **3** times. **(4 points)**

module tb\_rot ();

wire [3:0] Q ;

reg [3:0] D ;

reg clk, reset, load, rot ;

rotator m1 (D, reset, clk, load, rot, Q) ;

initial begin /reset and clock sequence

reset = 1; clk=0;

#5 reset =0 ; forever #10 clk=~clk;

end

initial begin

load=0 ; rot=0 ; D = 4’b1010 ; //or D=10 .. initialize inputs

@ (negedge clk) load= 1 ; //load the register with 1010

@ (negedge clk) load= 1 ; //deactivate the load signal – one clock cycle after load – do nothing

@ (negedge clk) ; //another clock cycle after load– do nothing

@ (negedge clk) rot=1 ; //1st rotation

@ (negedge clk) ; //2nd rotation, rot signal remains 1

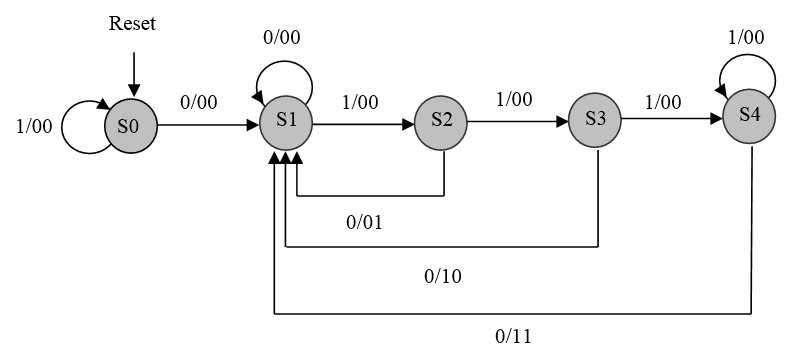
@ (negedge clk) ; //3rd rotation,rot signal remains 1

end

endmodule

**Question 9: (10 points)**

**a)** Write a **behavioral** Verilog description of a sequential circuit with the state diagram below. The circuit has an asynchronous **Reset** input, one input **X**, and two outputs: **Y** and **Z**. Use the following state encodings: **S0=000, S1=001, S2=010, S3=100, S4=111**. If the circuit ever gets into any of the unused states, it will go to state **S0** no matter what the input value is. **(6 points)**



module Mealy\_fsm (output wire y,z, input x, clk, reset );

localparam S0 = 2'b000, S1=2'b001, S2=2'b010, S3=2'b011, S4=100 ; //symbolic names for state values

reg [2:0] state ; //state register

assign y = ((state== s3) | (state== s4)) & ~x ; //y=1 only if we are in state S3 or state S4, and x is 0

assign z = ((state== s2) | (state== s4)) & ~x ; //y=1 only if we are in state S2 or state S4, and x is 0

always @(posedge clk, posedge reset) //This is only for the state transition

if (reset) state <= S0;

else case (state)

S0: if (!x) state <= S1 ; //if x=1, state remain at S0

S1: if (x) state <= S2 ; //if x=0, state remain at S0

S2: if (x) state <= S3 ; else state <= S1 ;

S3: if (x) state <= S4 ; else state <= S1 ;

S4: if (!x) state <= S1 ; else state <= S1 ;

default: state <=S0 ; //all non-used states go to S0

endcase

endmodule

**b)** Write a test bench to test the circuit. Let the clock cycle = 20 time units. First, reset the circuit, then apply the following input sequence to **X**: **0, 1, 0, 1, 1, 1, 0**. **(4 points)**

module tb\_fsm ();

wire y,z ; reg clk, reset,x ;

Mealy\_fsm m1 (y, z, x, clk, reset );

initial begin /reset and clock sequence

reset = 1; clk=0;

#5 reset =0 ; forever #10 clk=~clk;

end

initial begin

x=0 ; //1st input bit x=0

@ (negedge clk) x=1 ; //2nd input bit x=1

@ (negedge clk) x=0 ; //3rd input bit x=0

@ (negedge clk) x=1 ; //4th input bit x=1

@ (negedge clk) ; //5th input bit, x remain 1

@ (negedge clk) ; //6th input bit, x remain 1

@ (negedge clk) x=0 ; //7th input bit x=0

end

endmodule