# Department of Electrical & Computer Engineering
# Dalhousie University

# ECED 4260
# IC Design and Fabrication
# Midterm Examination

Instructor: Y. Ma
Exam date: Oct. 22, 2015
Exam duration: 80 minutes
Aids permitted: This is a closed book exam. No books or notes are to be consulted. Two double-sided 8.5″x11″ formula sheets are allowed.

Instructions:
1. Provide your name printed, signature and I.D. number on this page.
2. Verify that this booklet contains 8 pages (including the cover page).
3. Neatly enter your answers in the spaces provided.
4. Use the reverse sides of the pages for rough work.

Student name: _____

Signature: _____

Student ID: _____

| Question | Time (min) | Worth | Mark | Subject |
|---|---|---|---|---|
| 1 | 20 | 25 | | Combinational logic design |
| 2 | 20 | 25 | | VHDL |
| 3 | 20 | 25 | | Logic Simulation |
| 4 | 20 | 25 | | Finite State Machine |
| Total | 80 min | 100 | | |

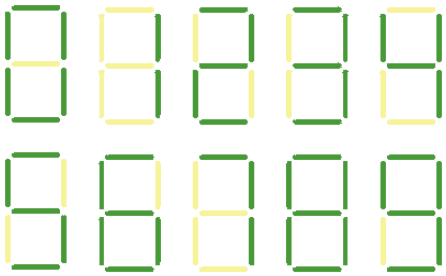
**Question 1. (25 marks)**
Use JK flip-flops to design a finite state machine (FSM) that works as a 2-bit binary synchronous up-counter when its input X=0 and as a 2-bit binary synchronous down-counter when its input X=1.

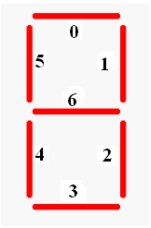
(a) (5 marks) Draw the state transition diagram.
(b) (5 marks) Derive the next state equations.
(c) (10 marks) Find minimized logic for the inputs of JK flip-flops.
(d) (5 marks) Draw the resulting circuit that implements this FSM.

Question 1 (cont'd)

**Question 2. (25 marks)**

Write a VHDL architecture for the entity below which implements a custom BCD to 7-segment decoder with the same output as assignment #1, also shown below. A darkened segment represents a '0'. Segment numbers are shown on the right.






| 7-segment display | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|---|
| Inputs: BCD | 0000 | 0001 | 0010 | 0011 | 0100 | 0101 | 0110 | 0111 | 1000 | 1001 |

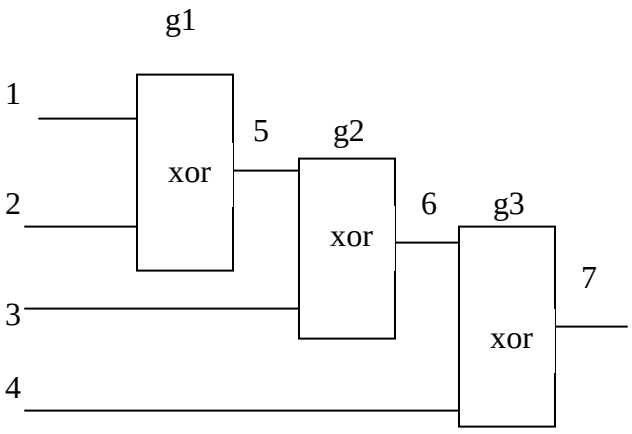
```
ENTITY BCD_to_7segment IS
        PORT ( BCD: in bit_vector (3 dwonto 0);
                LED_display: out bit_vector (6 downto 0));
END BCD_to_7segment;
```

.

Question 2 (cont’d)

**Question 3. (25 marks)**

The circuit given below is part of a larger system operating at a frequency of 200MHz. It was initialized by input x1x2x3x4=0000 at t=0, and all the circuit nodes are in known stable states. Perform the event-driven simulation of the circuit for two input vectors x1x2x3x4=0101 at the end of 1st clock cycle and x1x2x3x4=0000 at the end of second clock cycle. Complete the simulation table given below for t=0 to t=20, and derive the timing diagram from your simulation table. **Assume transport delay for all gates.**




Delay for g1: 5ns

Delay for g2: 3ns

Delay for g3: 4ns

(a) (18 marks) Complete the simulation table below

| | Line values | | | | | | | Gates affected | | | Scheduled events |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Time | 1 | 2 | 3 | 4 | 5 | 6 | 7 | g1 | g2 | g3 | |
| init | 0 | 0 | 0 | 0 | | | | | | | |
| | | | | | | | | | | | |
| | | | | | | | | | | | |
| | | | | | | | | | | | |
| | | | | | | | | | | | |
| | | | | | | | | | | | |
| | | | | | | | | | | | |
| | | | | | | | | | | | |
| | | | | | | | | | | | |
| | | | | | | | | | | | |
| | | | | | | | | | | | |
| | | | | | | | | | | | |
| | | | | | | | | | | | |
| | | | | | | | | | | | |

(b) (7 marks) Derive the wave-form timing diagram of all the circuits nodes.

1
---

2
---

3
---

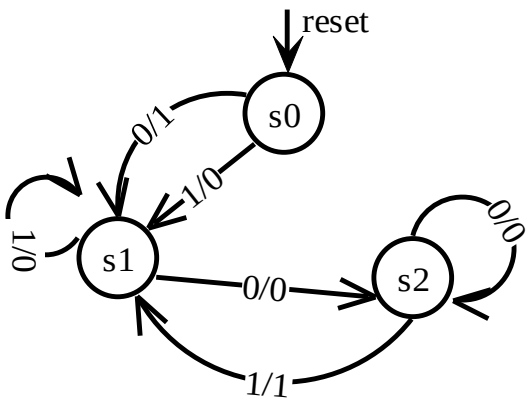
4
---

5
---

6
---

7
---

**Question 4. (25 marks)**
A finite state machine has one input x and one output z. The state transition diagram is shown below. The VHDL code given below describes part of the FSM architecture. By writing directly in the code, complete the architecture of this FSM with asynchronous reset.



```
library ieee;
use ieee.std_logic_1164.all;

entity example is
        port (clk, reset: in std_logic;
                x: in std_logic;
                z: out std_logic);

end entity example;

Architecture state_machine of example is
        Type stateType is (s0,s1,s2);
        Signal present_state, next_state: stateType;
Begin

Proc: process (present_state,x)

Begin
Case present_state is
        When s0 =>
        If (x='1') then Next_state <=s1; z <='0';
        Else Next_state <=s1; z <='1';
        End if;

        When s1 =>
        If (x='1') then Next_state <=s1; z<='0';
        Else Next_state <=s2; z<='0';
        End if;

        When s2 =>
        If (x='1') then Next_state <=s1; z <='1';
        Else Next_state <=s2; z <='0';
        End if;
End case;
End process Proc;




End architecture state_machine;
```