University of Windsor

School of Computer Science
Faculty of Science

COMP-2650: Computer Architecture I: Digital Design
Winter 2021

Assignment#	Date	Title	Due Date	Grade Release Date
Lec 09	Week 09	**Combinational Logic**	March 09, 2021 Tuesday Midnight AoE Wednesday 7 AM EDT	March 15, 2021

The objectives of the lecture (weekly) assignments are to practice on topics covered in the lectures as well as improving the student's critical thinking and problem-solving skills in ad hoc topics that are closely related but not covered in the lectures. Lecture assignments also help students with research skills, including the ability to access, retrieve, and evaluate information (information literacy).

Lecture Assignments Deliverables

You should answer two of the below questions based on your preference using an editor like MS Word, Notepad, and the likes or pen in papers. You have to write and scan the papers clearly and merge them into a single file in the latter case. In the end, you have to submit all your answers in one single pdf file `Lec09_UWinID.pdf` containing the following items:

1. Your name, UWinID, and student number
2. The question Id for each answer. Preferably, the questions should be answered in order of increasing Ids. *Please note that if your answers cannot be read, you will lose marks.*
3. Including the questions in your submission pdf file is optional.

Please follow the naming convention as you lose marks otherwise. Instead of UWinID, use your own UWindsor account name, e.g., mine is hfani@uwindsor.ca, so, my submission would be: `Lec09_hfani.pdf`

Lecture Assignments
(Select Only 2 Questions based on your preference)

1. Design an Excess3-to-BCD decoder. *Hint: Since Excess-3 code starts at 3 and ends at 12, the binary input for 0,1,2 and 13,14,15 can be used as don't care conditions.*

2. Design an Excess3-to-Aiken decoder in the form of product of sums.

3. Design an Aiken-to-Excess3 decoder in the form of product of sums.

4. Design an Aiken-to-Gray decoder. *Hint: while Gray code can code 0 to 15, Aiken code is able to code 0 to 9. Hence, the Gray codes from 10 to 15 can be used as don't care conditions.*

5. Design a Gray-to-Aiken decoder. *Hint: while Gray code can code 0 to 15, Aiken code is able to code 0 to 9. Hence, the Aiken codes from 10 to 15 must be generated by concatenating the Aiken code for 1 and the Aiken codes for 0 to 5.*

6. Design a 7-segment decoder for Excess-3.

7. Design a 7-segment decoder for Aiken.

8. Design a half-subtractor circuit with inputs *x* and *y* and outputs D (Difference) and B_{out} (Output Borrow). The circuit subtracts the bits x – y and places the difference in D and the borrow in B_{out}.

9. Design a full-subtractor (1-bit subtractor) circuit with three inputs x, y, B_p in and two outputs D and B_{out}. The circuit subtracts x – y – B_p, where B_p in is the input borrow from previous subtraction, B_{out} is the output borrow, and D is the difference.

10. Assume that the exclusive-OR gate has a propagation delay of 10 ns and that the AND or OR gates have a propagation delay of 5 ns. What is the total propagation delay time in the 4-bit adder without carry lookahead circuit?

11. Design a combinational circuit that compares two 4-bit numbers to check if they are equal. The circuit output is equal to 1 if the two numbers are equal and 0 otherwise.

12. Draw the logic diagram of a 2-to-4-line decoder using (a) NOR gates only and (b) NAND gates only. Include an enable input.

13. Construct a 5-to-32-line decoder with four 3-to-8-line decoders with enable and a 2-to-4-line decoder. Use block diagrams for the components.

14. Construct a 4-to-16-line decoder with five 2-to-4-line decoders with enable.

15. A combinational circuit is specified by the following three Boolean functions using one decoder:

    $F_1(A, B, C) = \sum(11, 4, 6)$
    $F_2(A, B, C) = \sum(13, 5)$
    $F_3(A, B, C) = \sum(12, 4, 6, 7)$

16. Using a decoder and external gates, design the combinational circuit defined by the following three Boolean functions:
    a. $F_1 = x'yz' + xz$
       $F_2 = xy'z' + x'y$
       $F_3 = x'y'z' + xy$

    b. $F_1 = (y' + x)\, z$
       $F_2 = y'z' + x'y + yz'$
       $F_3 = (x + y)\, z$

17. Design a four-input priority encoder with input D_0 having the highest priority and input D_3 the lowest priority.