Department of Electrical & Computer Engineering

COEN 212:
DIGITAL SYSTEMS DESIGN I
Lecture 6: Analysis and design of combinational circuits

Instructor: Dr. Reza Soleymani, Office: EV-5.125,
Telephone: 848-2424 ext.: 4103.

Lecture 6:
Objectives of this lecture

Concordia University
Department of Electrical & Computer Engineering

- In this lecture, we discuss:
  - Analysis of Digital Circuits.
  - Design of Circuits from a Definition.

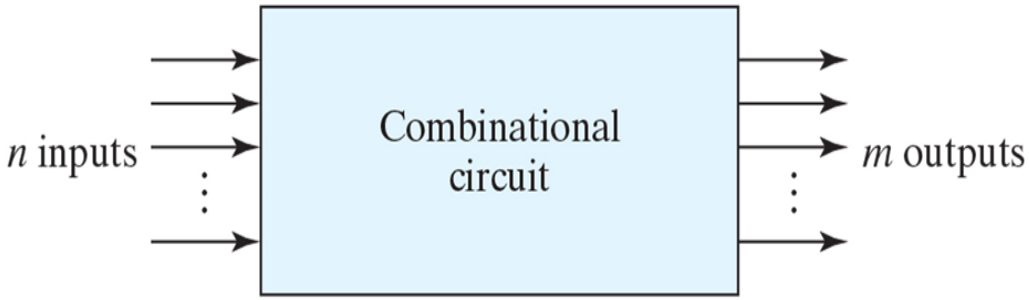
Lecture 6:
Reading for this lecture

- Digital Design by M. Morris R. Mano and Michael D. Ciletti, 6th Edition, Pearson, 2018:
  - Chapter 4 (4.1 and 4.4)

Lecture 6: Combinational Circuits

Concordia University
Department of Electrical & Computer Engineering

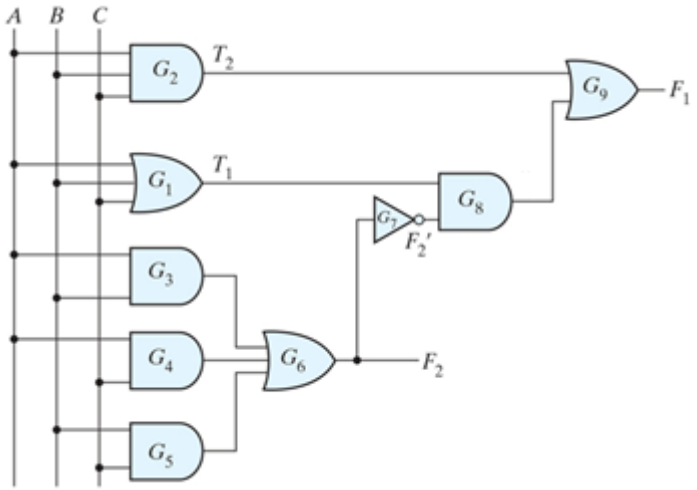
- A combinational circuit is a circuit whose output is a function of its inputs. The schematic for a combinational circuit with n inputs and m outputs is:



- A combinational circuit can be described using a truth table with 2^n rows. Each output will generate a column having a 1 or a 0 for each of the 2^n input choices.

Lecture 6:
Analysis

- The analysis is a procedure for finding the function that a given circuit performs.
- Starting from a circuit diagram:
  - Label the outputs of the gate with relevant and distinct names.
  - Find the output of each gate starting from the input.
  - Substituting the outputs of the gates in early stages in later stage gates, derive the expression for the outputs of the circuit.
- **Example:**



Lecture 6:
Analysis: Boolean Expression

- Label internal nodes with T_i , $i = 1,2, \ldots, 6$ and the final nodes with F_1 and F_2. Then:
  - $T_1 = ABC$
  - $T_2 = A + B + C$
  - $T_3 = AB$
  - $T_4 = AC$
  - $T_5 = BC$
- So: $F_2 = T_3 + T_4 + T_5 = AB + AC + BC$
- Then:
  - $T_6 = T_2 F_2' = T_2(AB + AC + BC)' = (A + B + C)(AB + AC + BC)'$
  - $F_1 = T_1 + T_6 = ABC + (A + B + C)(AB + AC + BC)'$
- $= ABC + (A + B + C)(A' + B')(A' + C')(B' + C')$
- $= ABC + (AB' + A'B + A'C + B'C)(A'B' + C')$
- $= ABC + AB'C' + A'BC' + A'B'C$

Lecture 6:
Analysis: Truth Table

- While Boolean expressions are compact way of representing a circuit, they do not give insight into the operation of the circuit. To get more insight, we can draw the truth table from the Boolean expression. Of course, it is also possible to derive the truth table from the circuit diagram directly.
- To derive the truth table following the procedure below:
  - Make a table with 2^n rows where n is the number of inputs. Write number 0 to $2^n - 1$ on the rows of the table.
  - Label the outputs of the gates.
  - Form the truth table for those gates whose inputs are the inputs of the circuit.
  - Continue making truth table for other gates until you get to the outputs.
- The truth table for the above example is shown below.

Analysis: Truth Table

Concordia University
Department of Electrical & Computer Engineering

- The truth table for the above example is shown below:

A	B	C	T_1	T_2	T_3	T_4	T_5	F_2	T_6	F_1
0	0	0	0	0	0	0	0	0	0	0
0	0	1	0	1	0	0	0	0	1	1
0	1	0	0	1	0	0	0	0	1	1
0	1	1	0	1	0	0	1	1	0	0
1	0	0	0	1	0	0	0	0	1	1
1	0	1	0	1	0	1	0	1	0	0
1	1	0	0	1	1	0	0	1	0	0
1	1	1	1	1	1	1	1	1	0	1

- Note that F_1 is equal to 1 if there is either one or three 1's at the input and F_2 is equal to 1 if two or more 1's are among A, B, and, C. This is a full adder, a circuit that adds two bits and a carry and gives the sum of the three bits and an output carry.

Lecture 6:
Design

- Design is the process of creating a combinational circuit from a functional specification.
- The design procedure consists of the following steps:
  - From the description of the circuit determine the number of the inputs and outputs. Assign a symbol to each input and each output.
  - Derive the truth table relating the inputs to each output.
  - Obtain a simplified Boolean function for each output.
  - Draw the logic diagram.
  - Verify the correctness of the design either manually or by simulation.

Lecture 6:
Design: Example

- **Example:** BCD to Excess-3 code converter.
- 4 input bits of a BCD (Binary Coded Decimal) value to 4 output bits of corresponding Excess-3 value.
- BCD represents digits 0 to 9 using natural 4-bit binary symbols 0000, 0001, …, 1001.
- In Excess-3 code each number is formed by adding three (3) to the corresponding BCD codeword.
- For example 0 is represented as 0011, one is represented as 0100,…, and 9 is 1100.
- The interesting thing about Excess-3 code is that if we invert each bit, we get the 9's complement of the codeword.
  - Take 0110 representing 3 and invert each bit to get 1001 that is the codeword for 6.

Lecture 6:
Design: Example

- **Example:** BCD to Excess-3 code converter.
- First we draw the Truth Table:

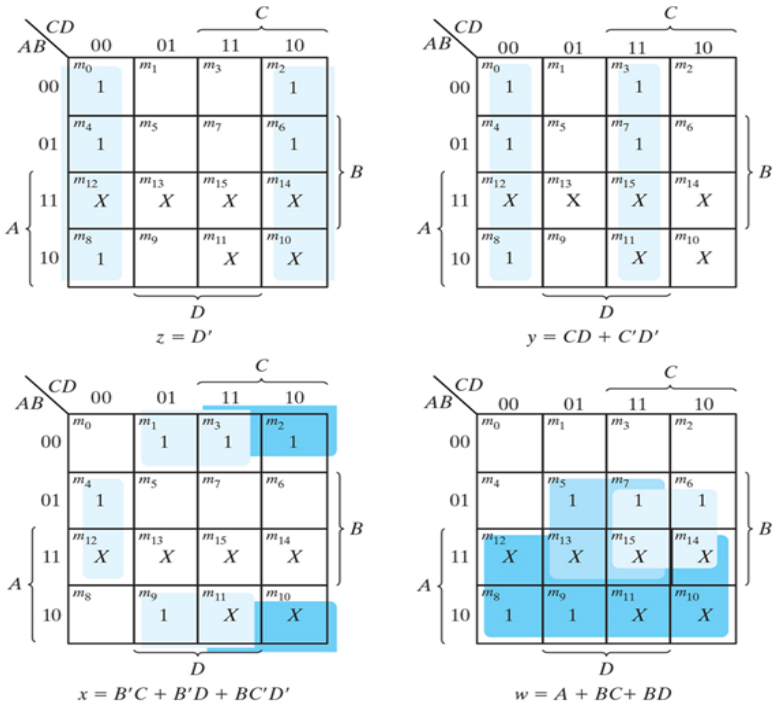
Input BCD				Output Excess-3 Code			
A	*B*	*C*	*D*	*w*	*x*	*y*	*z*
0	0	0	0	0	0	1	1
0	0	0	1	0	1	0	0
0	0	1	0	0	1	0	1
0	0	1	1	0	1	1	0
0	1	0	0	0	1	1	1
0	1	0	1	1	0	0	0
0	1	1	0	1	0	0	1
0	1	1	1	1	0	1	0
1	0	0	0	1	0	1	1
1	0	0	1	1	1	0	0

- Then we draw the K-map for the four outputs w, x, y, and z.

Lecture 6:
Design: Example

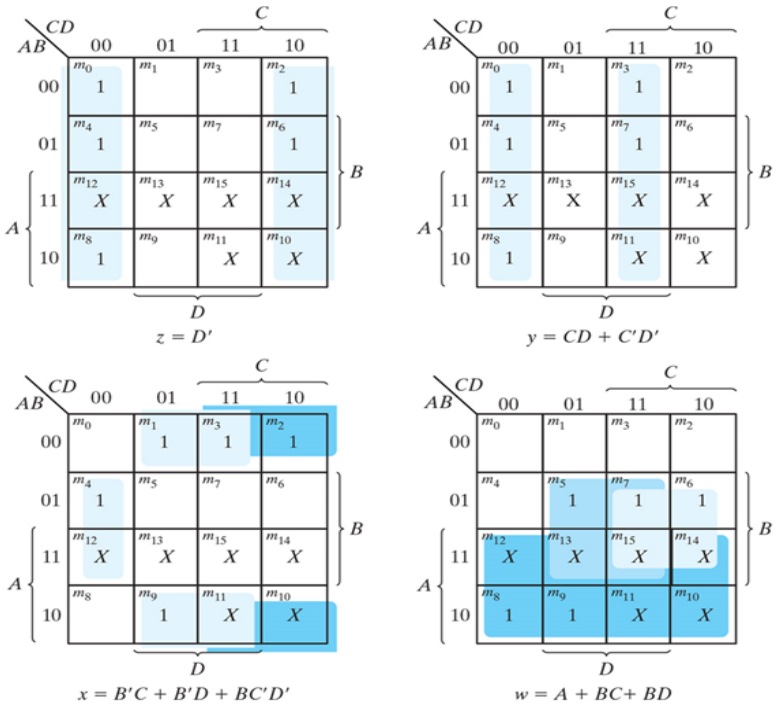
Department of Electrical & Computer Engineering

- We draw the K-map for the four outputs w, x, y, and z.



Lecture 6:
Design: Example

- We draw the K-map for the four outputs w, x, y, and z.



Université Concordia University
Department of Electrical & Computer Engineering

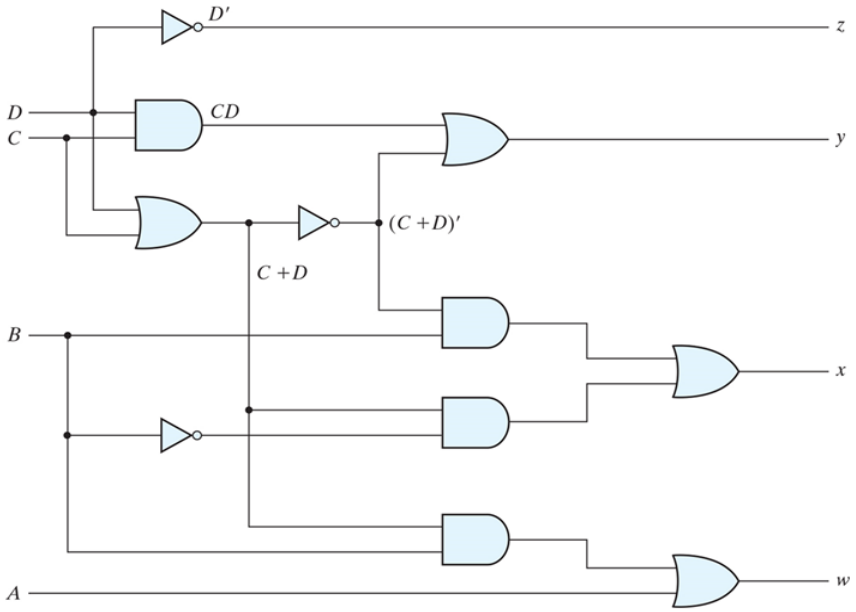
- So, we have:
  - $z = D'$
  - $y = CD + C'D' = CD + (C + D)'$
  - $x = B'C + B'D + BC'D' = B'(C + D) + B(C + D)'$
  - $w = A + BC + BD = A + B(C + D)$

- We have manipulated the expressions so that we can use similar two input gates. The implementation is shown below.

Lecture 6:
Design: Example

Concordia University
Department of Electrical & Computer Engineering

- Here is the implementation of BCD to Excess-3 converter



Lecture 6: Knowledge Check

- **Question 1:** Show that XOR can be implemented using four NAND gates.
- **Question 2:**
- **Question 3:**