THE UNIVERSITY OF TEXAS AT DALLAS

CS4341 Digital Logic & Computer Design

Lecture Notes 7

Omar Hamdy
Assistant Professor
Department of Computer Science

Example: Multiple Output Circuits

- In many cases, there are more than one output required to carry out the desired function behavior.
- Example, 4-input priority circuit
  - The circuit has 4-outputs that signals which input should be given the priority when more than one input are requesting it.
  - The circuit allows only one output signal to be high at any time.
  - Assume inputs A_3 (highest priority), A_2, A_1, A_0(lowest priority)
  - Assume outputs Y_3 indicating A_3 gets the priority, Y_2 indicating A_2 gets the priority, and so on.

4-Input Priority Circuit Design Approach 1

- One approach is to build the truth table, determine the minterms for each output, express each output in SOM form, then draw the circuit.

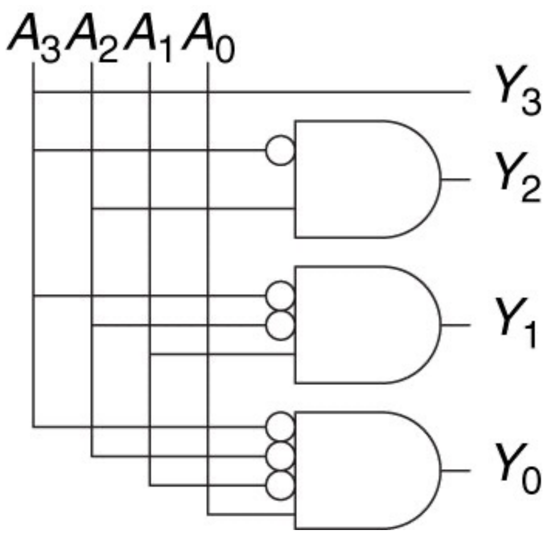
A_3	A_2	A_1	A_0	Y_3	Y_2	Y_1	Y_0
0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	1
0	0	1	0	0	0	1	0
0	0	1	1	0	0	1	0
0	1	0	0	0	1	0	0
0	1	0	1	0	1	0	0
0	1	1	0	0	1	0	0
0	1	1	1	0	1	0	0
1	0	0	0	1	0	0	0
1	0	0	1	1	0	0	0
1	0	1	0	1	0	0	0
1	0	1	1	1	0	0	0
1	1	0	0	1	0	0	0
1	1	0	1	1	0	0	0
1	1	1	0	1	0	0	0
1	1	1	1	1	0	0	0

- $Y_0 = m_1 = A'_3A'_2A'_1A_0$.
- $Y_1 = m_2 + m_3 = A'_3A'_2A_1A'_0 + A'_3A'_2A_1A_0$
  - Simplify $= A'_3A'_2A_1$ … why?
- $Y_2 = m_4 + m_5 + m_6 + m_7 = A'_3A_2A'_1A'_0 + A'_3A_2A'_1A_0 + A'_3A_2A_1A'_0 + A'_3A_2A_1A_0$
  - Simplify $= A'_3A_2$ … why?
- $Y_3 = m_8 + m_9 + m_{10} + m_{11} + m_{12} + m_{13} + m_{14} + m_{15} = A_3A'_2A'_1A'_0 + A_3A'_2A'_1A_0 + A_3A'_2A_1A'_0 + A'_3A'_2A_1A_0 + \ldots\ldots.$
  - Simplify $= A_3$ … why?
- Draw the circuit using the PLA approach

4-Input Priority Circuit Design Approach 2

- Careful review of the circuit function, we notice:
  - Y_3 output is 1 when A_3 is 1 and does not care about the values of any other input.
  - Therefore, $Y_3 = A_3$
  - Y_2 output is 1 when 1) A_3 is 0, 2) A_2 is 1 and does not care about the input values of A_1 or A_0
  - Y_1 output is 1 when 1) A_3 is 0, 2) A_2 is 0, 3) A_1 is 1 and does not care about the input value of A_0
  - Y_0 output is 1 when 1) A_3 is 0, 2) A_2 is 0, 3) A_1 is 0 and 4) A_0 is 1
- The concept of "Don't Care" can be represented with symbol X in the truth table, and can simplify the design process

A_3	A_2	A_1	A_0	Y_3	Y_2	Y_1	Y_0
0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	1
0	0	1	X	0	0	1	0
0	1	X	X	0	1	0	0
1	X	X	X	1	0	0	0

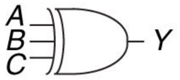


Multilevel Combinational Logic Design

- In many cases, multi-level circuit designs can reduce the number of true gates required.
- Example: Implementation of 3-input XOR function (remember, there is no actual 3-input XOR gate)
- Truth table out is 1 if the inputs have odd number of 1s

$A \oplus B \oplus C = m_1 + m_2 + m_4 + m_7 = A'B'C + AB'C' + ABC$ (8 gates)

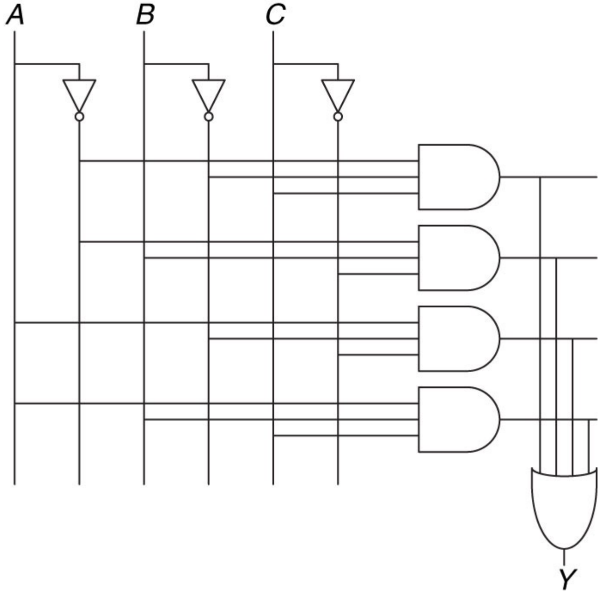
XOR3



$Y = A \oplus B \oplus C$

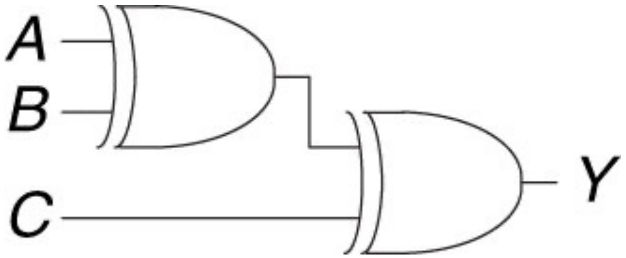
A	B	C	Y
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1

(a)



(b)

$A \oplus B \oplus C = (A \oplus B) \oplus C$ (2 gates)



Circuit Optimization

- The goal is to obtain the simplest implementation for a given function
- We need a methodical approach to simplify any design using a specific procedure or algorithm
- Circuit simplicity has to be quantified or measured against some criteria (cost, efficiency, power consumption, etc)
- Using algebraic rules can be challenging and not systematic
- K-Map is a simple straightforward procedure to simplify Boolean functions

Karnaugh Map (K-Map)

- A K-map is a diagram made of a collection of adjacent squares:
  - Each square represents a minterm
  - The collection of squares is a graphical representation of a Boolean function
  - Adjacent squares differ in the value of **<u>one</u>** variable only
  - Alternative algebraic expressions for the same function are derived by recognizing patterns of squares
- The K-map can be viewed as a reorganized version of the truth table

Importance of K-Map

- K-Maps provide means of:
  - Finding optimum or near optimum
    - SOP and POS standard forms
    - Two-level AND/OR and OR/AND circuits
  - Visualizing concepts related to manipulating Boolean expressions
  - Demonstrating concepts used by computer-aided design programs to simplify large circuits

2-Variable K-Map

- If we represent each minterm as a box, then we have:

$m_0 = x'y'$	$m_1 = x'y$	$m_2 = xy'$	$m_3 = xy$

- We need to assemble the boxes such that adjacent boxes differ in exactly one literal value

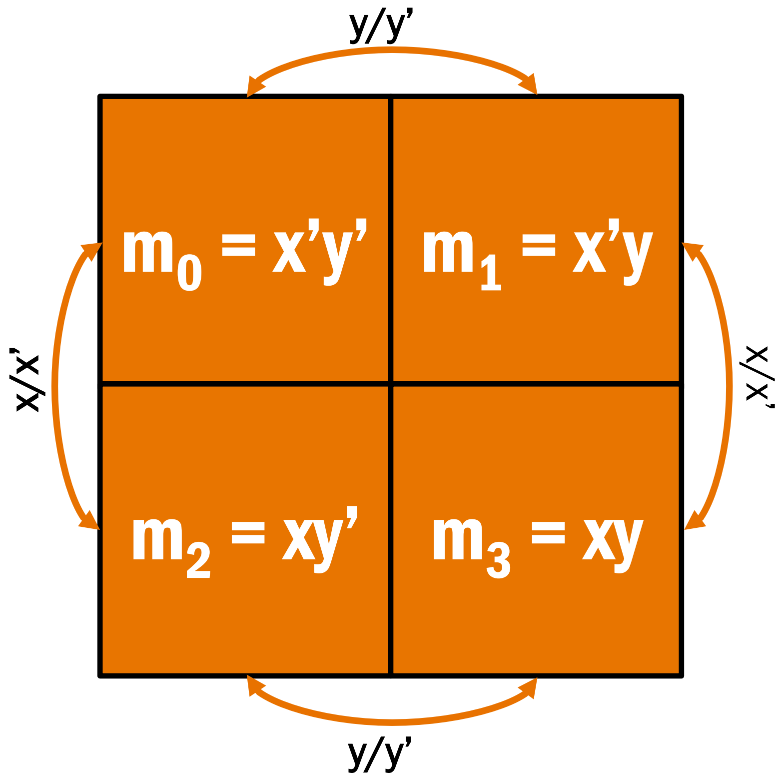
How to connect?

$m_0 = x'y'$	$m_1 = x'y$

$m_2 = xy'$	$m_3 = xy$

How to connect?

2-Variable K-Map



To Do List

- Review lecture notes, and try the examples yourself
- Work on assignment 1