Chapter 5: The Design of Combinational Systems

The Design Process for Combinational Systems

The goal: We want to go from a problem statement (in English) to an implemented system (using logic gates).

Step 1. Decide on the binary inputs and outputs, and specify the logical 1 and logical 0 conditions.

If a light is on, is that logical 1 or logical 0?
If a switch is up, is that logical 1 or logical 0?
Can the problem be subdivided into smaller problems?

Step 2. Express the design problem in terms of a truth table and a Boolean expression.

How many lines in a truth table with n inputs?

How should we write the truth table in a systematic way?

Step 3. Simplify! How?

Step 4. Implement the system with gates.

Design Example 1 Design a system that has four inputs (A, B, C and D) and one output (Z_2). The output is 1 iff a majority of the inputs are 1.

Develop the truth table

A	B	C	D	Z_2
0	0	0	0	
0	0	0	1	
0	0	1	0	
0	0	1	1	
0	1	0	0	
0	1	0	1	
0	1	1	0	
0	1	1	1	
1	0	0	0	
1	0	0	1	
1	0	1	0	
1	0	1	1	
1	1	0	0	
1	1	0	1	
1	1	1	0	
1	1	1	1	

From Alan Marcovitz, *Introduction to Logic Design*, 3rd ed, McGraw Hill, 2010

Write the canonical SOP expression that governs this truth table.

Simplify:

Implement:

<u>Design Example 2</u> Design a system to add three bits together. The system has three inputs (representing the three bits to be added—let's call them *a*, *b* and c_{in}) and two outputs (a sum bit and a carry bit—let's call them *s* and c_{out}).

Develop the truth table

a	b	c_{in}	c_{out}	s
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

From Alan Marcovitz, *Introduction to Logic Design*, 3rd ed, McGraw Hill, 2010

Write the canonical SOP expression that governs this truth table.

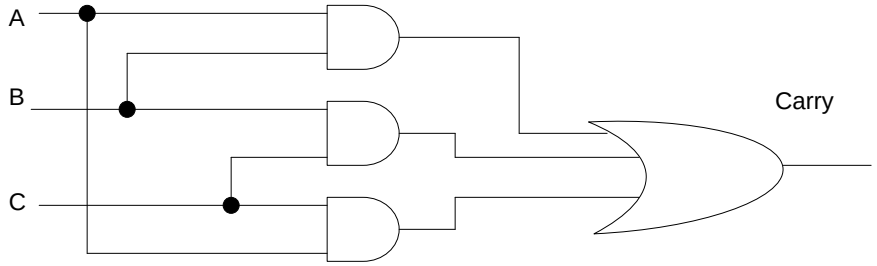
$s = \quad a'b'c + a'bc' + ab'c' + abc$

$c_{out} = \quad a'bc + ab'c + abc' + abc$

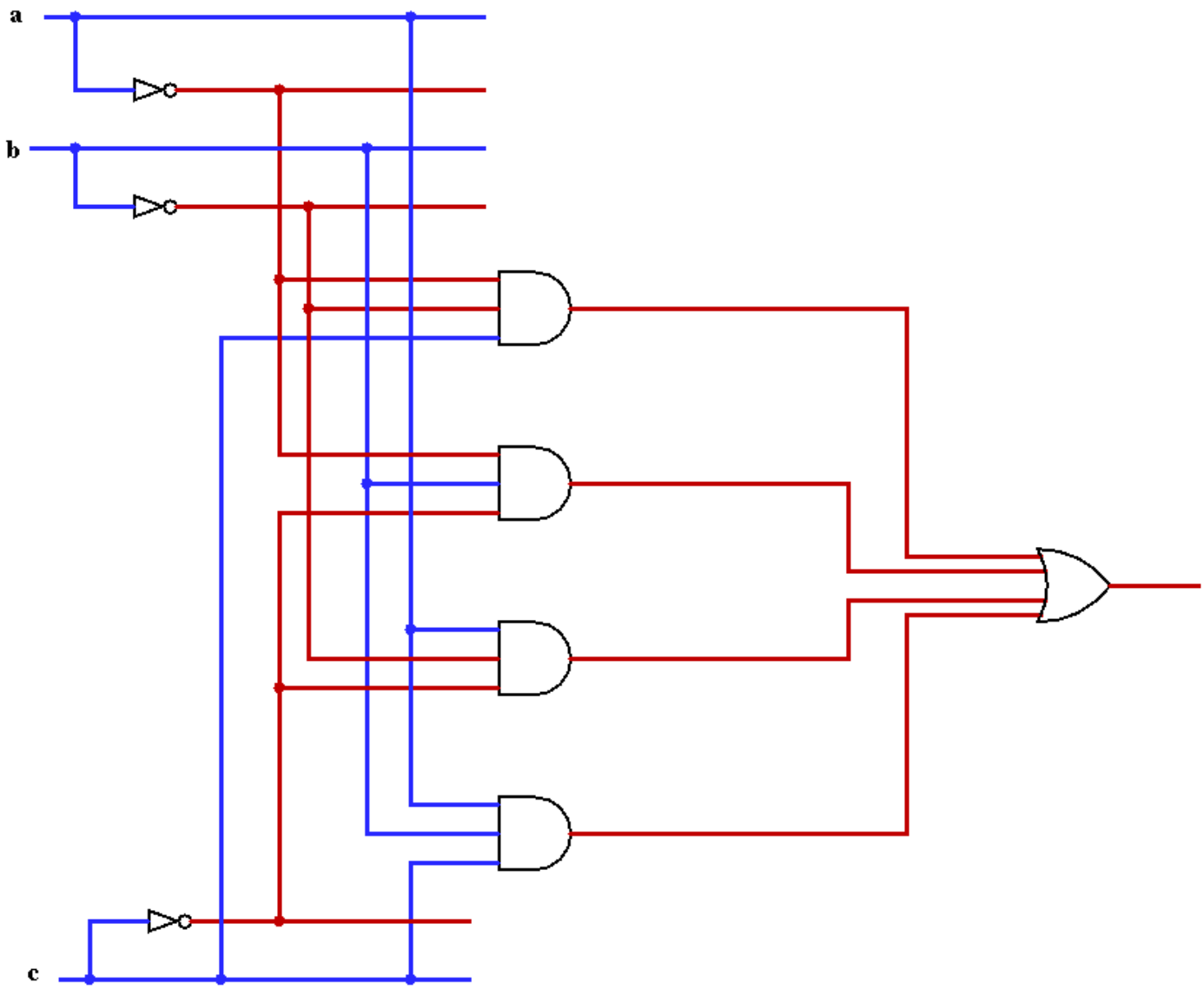
As it turns out, the expression for s cannot be further simplified. But we can simplify $c_{out.}$ Simplify!

Implement:

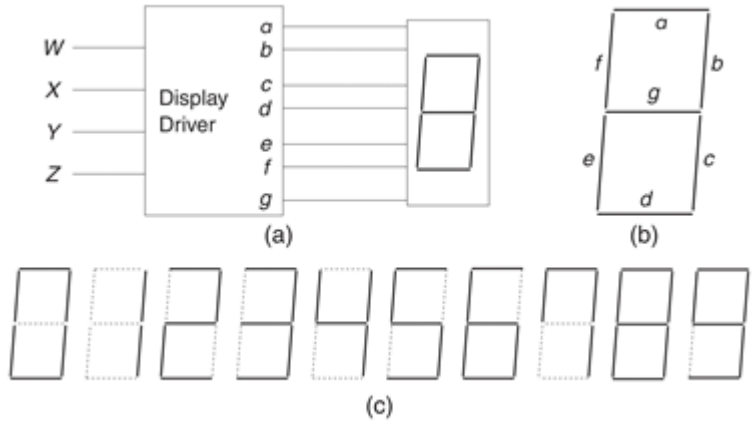
Carry bit c_{out}:



Sum bit s:



<u>Design Example 3</u> Design a display driver. The input is a four-bit BCD representation for a decimal digit (*W*, *X*, *Y*, *Z*, with Z being the LSB) and the output is a seven-segment display. A sketch of the desired operation is shown below.



From Alan Marcovitz, *Introduction to Logic Design*, 3rd ed, McGraw Hill, 2010

For this problem, we have to give some thought to Step 1. Assuming a logical 1 will light a display, what do you see as a problem?

Develop the truth table for segment e.

Digit	W	X	Y	Z	a	b	c	d	e	f	g
0	0	0	0	0							
1	0	0	0	1							
2	0	0	1	0							
3	0	0	1	1							
4	0	1	0	0							
5	0	1	0	1							
6	0	1	1	0							
7	0	1	1	1							
8	1	0	0	0							
9	1	0	0	1							
–	1	0	1	0							
–	1	0	1	1							
–	1	1	0	0							
–	1	1	0	1							
–	1	1	1	0							
–	1	1	1	1							

From Alan Marcovitz, *Introduction to Logic Design*, 3rd ed, McGraw Hill, 2010

Write the canonical SOP expression:

Simplify!

Implementation is straightforward and left for you to do.

Design Example 4

You are working on a boiler system and have been tasked with implementing a safety circuit. The boiler is equipped with the following sensors:

- an active low pressure sensor that provides indication of excessive steam pressure
- an active high temperature sensor that provides indication of excessive operating temperature
- an active high inlet water sensor that provides indication of normal water flow into the boiler
- an active low flame sensor that provides indication of normal flame operation of the boiler

The safety circuit is to provide an alarm signal (active high logic) under the following conditions:

- excessive steam pressure exists –or-
- loss of inlet water while the boiler flame is lit –or-
- excess temperature while the boiler flame is lit

Develop a truth table, minimum SOP (using the Table of Fun, K-Maps or engineering acumen) and implement your design. To make things consistent, label your sensors as:

P for the pressure sensor.
T for the temperature sensor
W for the water flow sensor
F for the flame sensor

P	*T*	*W*	*F*	Alarm
0	0	0	0	
0	0	0	1	
0	0	1	0	
0	0	1	1	
0	1	0	0	
0	1	0	1	
0	1	1	0	
0	1	1	1	
1	0	0	0	
1	0	0	1	
1	0	1	0	
1	0	1	1	
1	1	0	0	
1	1	0	1	
1	1	1	0	
1	1	1	1	