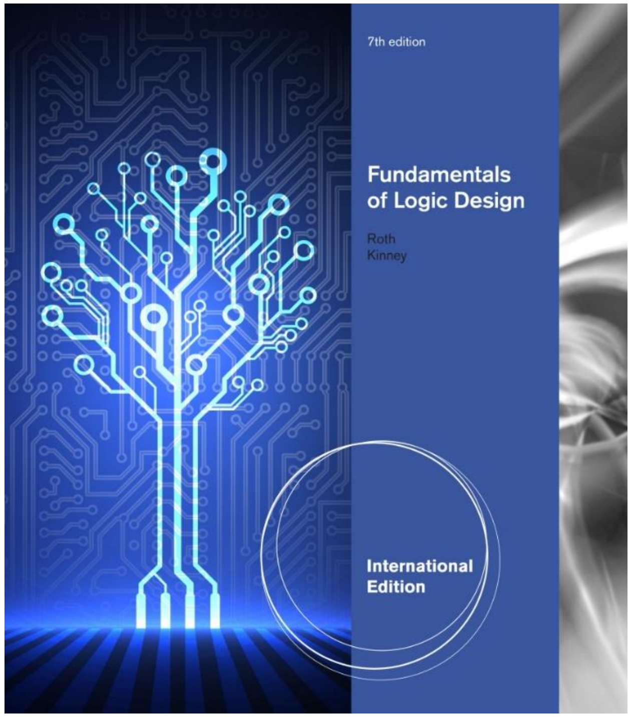


7th edition

Fundamentals of Logic Design

Roth
Kinney

International Edition

Combinational Circuit Design and Simulation Using Gates

성균관대학교
SUNG KYUN KWAN UNIVERSITY

Contents

Objectives

- Draw a timing diagram for a combinational circuit with gate delays.
- Define static 0- and 1-hazards and dynamic hazards. Given a combinational circuit, find all of the static 0- and 1-hazards. For each hazard, specify the order in which the gate outputs must switch in order for the hazard to actually produce a false output.
- Given a switching function, realize it using a two-level circuit which is free of static and dynamic hazards (for single input variable changes).
- Design a multiple-output NAND or NOR circuit using gates with limited fan-in.
- Explain the operation of a logic simulator that uses four-valued logic.
- Test and debug a logic circuit design using a simulator.

Review of Combinational Circuit Design

Review of Combinational Circuit Design

- The first step in combinational switching circuit design is to set up a truth table which specifies the output(s) as a function of the input variables.
- The next step is to derive simplified algebraic expressions for the output functions using Karnaugh maps, the Quine-McCluskey method, or a similar procedure. The resulting equations can then be simplified algebraically.
- The simplified algebraic expressions are then manipulated into the proper form, depending on the type of gates to be used in realizing the circuit.

Two-level and Multi-level Circuit Design

- Minimum two-level AND-OR, NAND-NAND, OR-NAND, and NOR-OR circuits can be realized using <u>the minimum sum of products</u> as a starting point.
- Minimum two-level OR-AND, NOR-NOR, AND-NOR, and NAND-AND circuits can be realized using <u>the minimum product of sums</u> as a starting point.
- Design of multi-level, multiple-output NAND-gate circuits is most easily accomplished by first designing a circuit of AND and OR gates. The minimum SOP expression is found and factored in various ways until an economical circuit of the desired form can be found.

Review of Combinational Circuit Design

- Design of multi-level, multiple-output NOR-gate circuits is most easily accomplished by first designing a circuit of AND and OR gates.
- In this case, the minimum sum-of-products expressions for the *complements* of the output functions is found.
- *After* factoring these expressions to the desired form, they are then complemented to get expressions for the output functions, and the corresponding circuit of AND and OR gates is drawn.

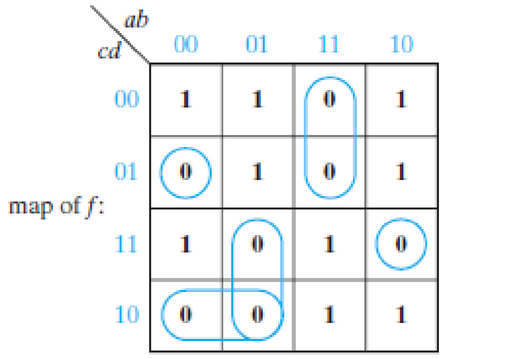
Design of Circuits with Limited Gate Fan-In

Introduction:

- In practical logic design problems, the maximum number of inputs on each gate (or the **fan-in**) is limited.
- If a two-level realization of a circuit requires more gate inputs than allowed, factoring the logic expression to obtain a multi-level realization is necessary.

Example 1: NOR-gate circuit design

Example Realize $f(a, b, c, d) = \Sigma m(0, 3, 4, 5, 8, 9, 10, 14, 15)$ using three-input NOR gates.



cd \ ab	00	01	11	10
00	1	1	0	1
01	0	1	0	1
11	1	0	1	0
10	0	0	1	1

$$f' = a'b'c'd + ab'cd + abc' + a'bc + a'cd'$$

Design of Circuits with Limited Gate Fan-In

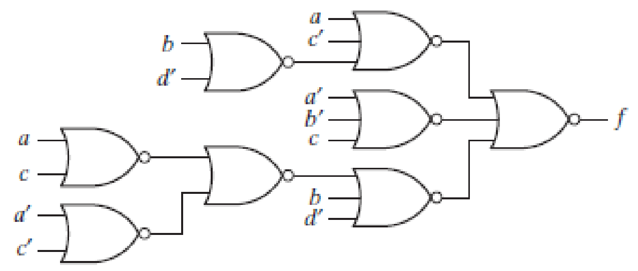
Example 1: NOR-gate circuit design

As can be seen from the preceding expression, a two-level realization requires two four-input gates and one five-input gate. The expression for f' is factored to reduce the maximum number of gate inputs to three and, then, it is complemented:

$$f' = b'd(a'c' + ac) + a'c(b + d') + abc'$$
$$f = [b + d' + (a + c)(a' + c')][a + c' + b'd][a' + b' + c]$$

The resulting NOR-gate circuit is shown in Figure 8-1.

FIGURE 8-1



Design of Circuits with Limited Gate Fan-In

Multi-level Circuit Design

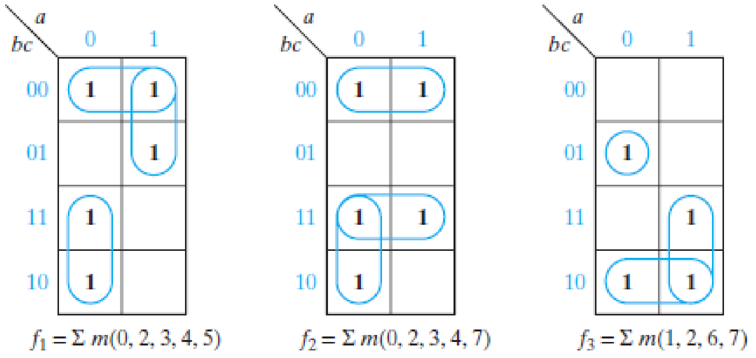
- When designing multiple-output circuits with more than two levels, it is usually best to minimize each function separately.
- The resulting two-level expressions must then be factored to increase the number of levels.
- This factoring should be done in such a way as to introduce common terms wherever possible.

Example 2: NAND-gate circuit design

Example

Realize the functions given in Figure 8-2, using only two-input NAND gates and inverters. If we minimize each function separately, the result is

$$f_1 = b'c' + ab' + a'b$$
$$f_2 = b'c' + bc + a'b$$
$$f_3 = a'b'c + ab + bc'$$



a
bc
0
1
00
01
11
10
$f_1 = \Sigma\, m(0, 2, 3, 4, 5)$
$f_2 = \Sigma\, m(0, 2, 3, 4, 7)$
$f_3 = \Sigma\, m(1, 2, 6, 7)$

FIGURE 8-2

© Cengage Learning 2014

Design of Circuits with Limited Gate Fan-In

Example 2: NAND gate circuit design

Each function requires a three-input OR gate, so we will factor to reduce the number of gate inputs:

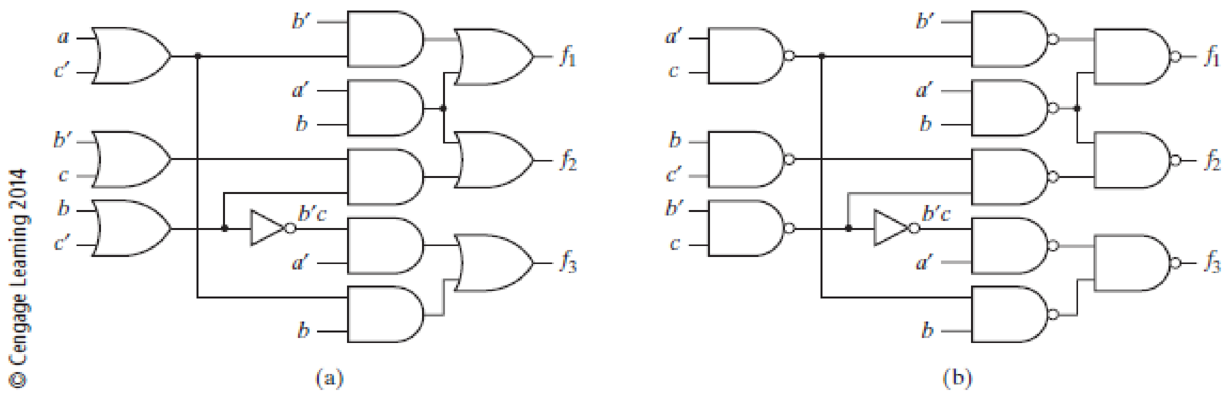
$$f_1 = b'\underline{(a + c')} + \underline{a'b}$$
$$f_2 = b(a' + c) + b'c' \qquad \text{or} \qquad f_2 = (b' + c)(b + c') + \underline{a'b}$$
$$f_3 = a'b'c + b\underline{(a + c')}$$

The second expression for f_2 has a term common to f_1, so we will choose the second expression. We can eliminate the remaining three-input gate from f_3 by noting that

$$a'b'c = a'(b'c) = a'(b + c')'$$

Figure 8-3(a) shows the resulting circuit, using common terms $a'b$ and $a + c'$. Because each output gate is an OR, the conversion to NAND gates, as shown in Figure 8-3(b), is straightforward.

FIGURE 8-3 Realization of Figure 8-2



Gate Delays and Timing Diagrams

Gate Delays and Timing Diagrams

- When the input to a logic gate is changed, the output will not change instantaneously. <u>The transistors or other switching elements within the gate take a finite time to react to a change in input</u>, so that the change in the gate output is delayed with respect to the input change.
  - ✓ Propagation delays for a integrated circuit gates may be as short as a few nanoseconds (1 nanosecond = 10^{-9} second).
  - ✓ In the analysis of some types of sequential circuits, even short delays may be important.
- **Timing diagrams** are frequently used in the analysis of sequential circuits. These diagrams show various signals in the circuit as a function of time.

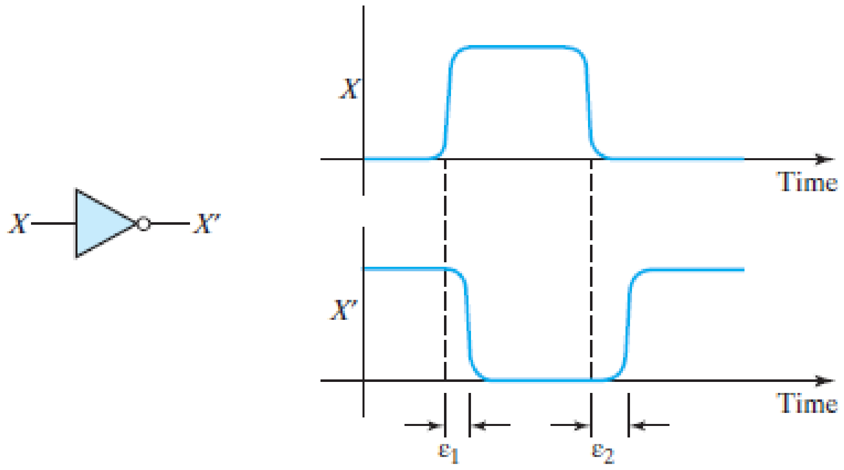
Gate Delays and Timing Diagrams

Propagation Delay for An Inverter:

- Figure 8-4 shows possible input and output waveforms for an inverter.
- If the change in output is delayed by time, ε , with respect to the input, we say that this gate has a propagation delay of ε.
- In practice, the propagation delay for a 0 to 1 output change may be different than the delay for a 1 to 0 change.

FIGURE 8-4
Propagation Delay in an Inverter

© Cengage Learning 2014

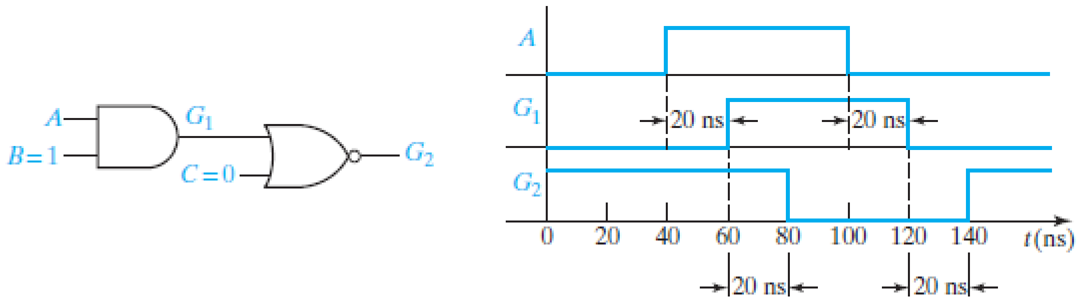


Gate Delays and Timing Diagrams

Timing Diagram for AND-NOR Circuit:

- Figure 8-5 shows the timing diagram for a circuit with two gates, assuming propagation delay of 20 ns.
- This timing diagram indicates what happens when gate inputs *B* and *C* are held at constant values 1 and 0 respectively, and input *A* is changed to 1 at *t* = 40 ns and then changed back to 0 at *t* = 100 ns.

FIGURE 8-5
Timing Diagram for AND-NOR Circuit

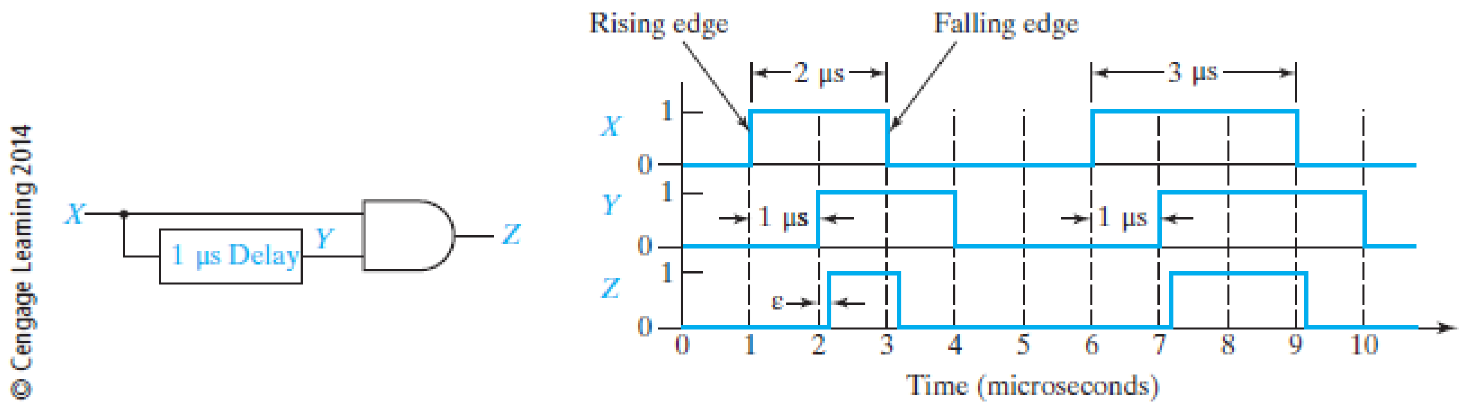


Gate Delays and Timing Diagrams

Timing Diagram for Circuit with Delay:

- Figure 8-6 shows a timing diagram for a circuit with an added delay element.
  - ✓ The input *X* consists of two pulses, the first of which is 2 microseconds wide and the second is 3 microseconds wide.
  - ✓ The delay element has an output *Y* which is the same as the input except that it is delayed by 1 microsecond.
  - ✓ The output *Z* of the AND gate should be 1 during the time interval in which both *X* and *Y* are 1. We assume a small propagation delay ε in the AND gates.

FIGURE 8-6 Timing Diagram for Circuit with Delay

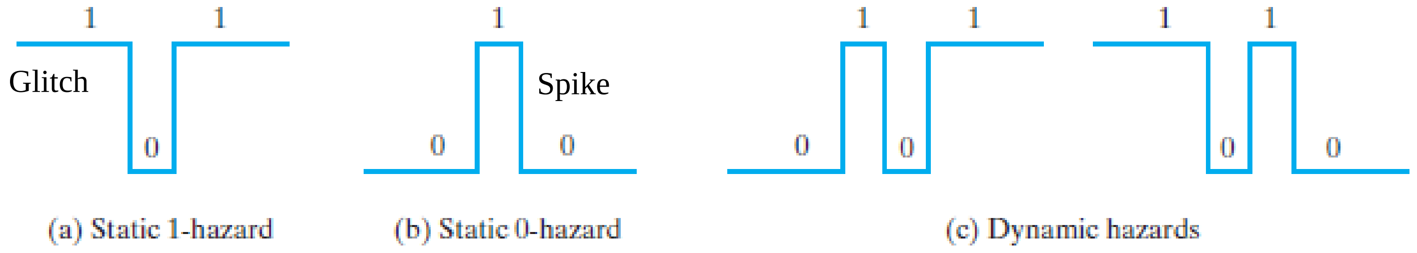


Hazards in Combinational Logic

Hazards:

- Switching transients occur when different paths from input to output have different propagation delays.
  - ✓ A circuit output may momentarily go to 0 when it should remain a constant 1, we say that the circuit has a **static 1-hazard.**
  - ✓ If the output may momentarily go to 1 when it should remain a 0, we say that the circuit has a **static 0-hazard**.
  - ✓ If, when the output is supposed to change from 0 to 1 (or 1 to 0), the output may change three or more times, we say that the circuit has a **dynamic hazard**.

FIGURE 8-7 Types of Hazards

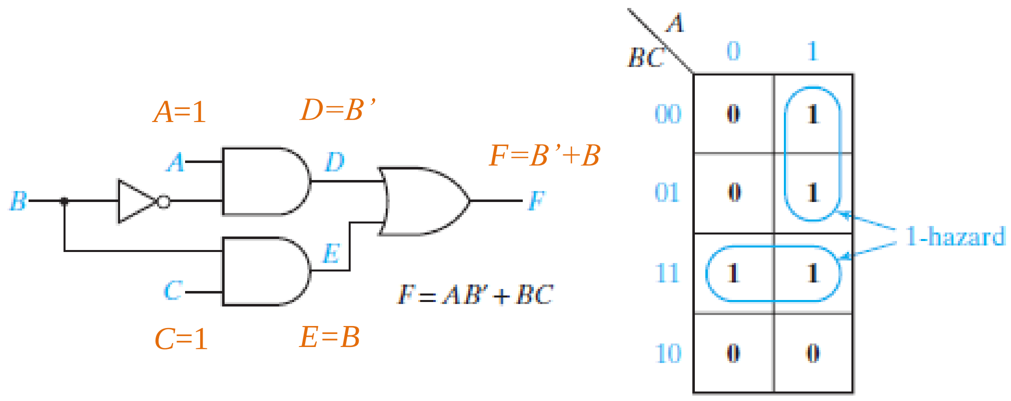


Hazards in Combinational Logic

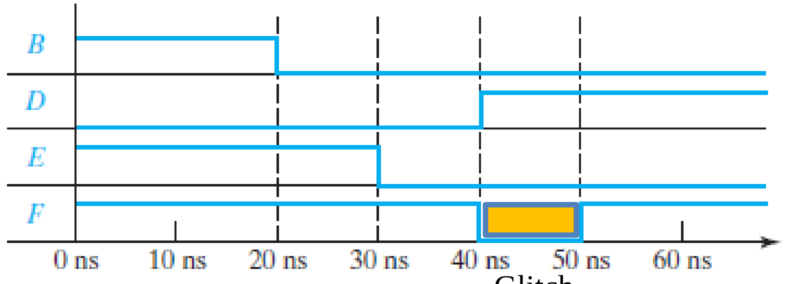
Detection of a 1-Hazard:

- $A = C = 1$, $F = B + B' = 1$
- If each gate has a propagation delay of 10ns, *E* will go to 0 before *D* goes to 1.
- *F* momentarily goes to 0 (a glitch cased by the 1-hazard)

FIGURE 8-8 Detection of a 1-Hazard



(a) Circuit with a static 1-hazard



(b) Timing chart

Hazards in Combinational Logic

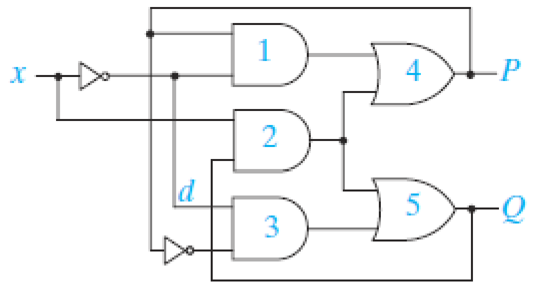
Static 1-Hazard and Asynchronous Sequential Circuit

- The next-state equations are

$$P^+ = x'P + xQ$$
$$Q^+ = x'P' + xQ$$

- Note that both equations contain static 1-hazards, and <u>P^+ contains a static 1-hazard</u> for $xPQ = 111 \leftrightarrow 011$.
- This circuit has two feedback loops.
- If the x inverter delay is large compared to the gate delays, <u>the circuit may generate incorrect state transition</u> due to the static1-hazard in P^+.

FIGURE 11-32 Asynchronous Circuit



(a)

PQ	x = 0	1
00	01	00
01	01	11
11	10	11
10	10	00

(b)

Hazards in Combinational Logic

Procedure to Detect Hazards in a Two-Level AND-OR Circuit:

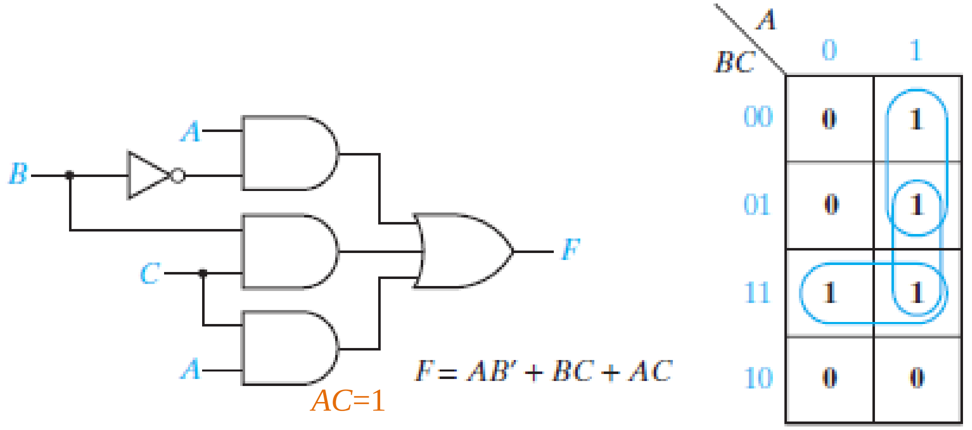
1. Write down the sum-of-products expression for the circuit.
2. Plot each term on the map and loop it.
3. If any two adjacent 1's are not covered by the same loop, a 1-hazard exists for the transition between the two 1's. For an *n*-variable map, this transition occurs when one variable changes and the other *n*−1 variables are held constant.

Circuit with Hazard Removed:

- The term *AC* remains 1 while *B* is changing, so no glitch can appear in the output.
- Note that *F* is no longer a minimum sum of products.

FIGURE 8-9
Circuit with Hazard Removed

© Cengage Learning 2014

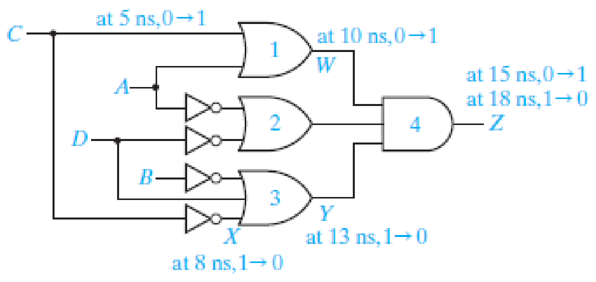


Hazards in Combinational Logic

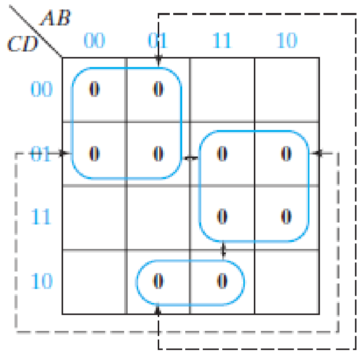
Detection of a Static-0 Hazard:

- The product-of-sums representation: $F = (A+C)(A'+D')(B'+C'+D)$
- Assuming gate delays of 3ns for each inverters and of 5ns for each AND gate and each OR gate.
- When $A = 0$, $B = 1$, $D = 0$, and C changes from 0 to 1, a spike appears at the Z output.

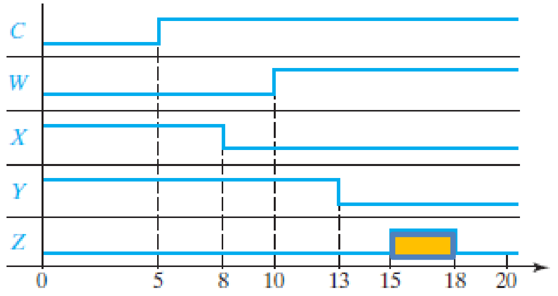
FIGURE 8-10 Detection of a Static 0-Hazard



(a) Circuit with a static 0-hazard



(b) Karnaugh map for circuit of (a)



Spike

(c) Timing diagram illustrating 0-hazard of (a)

Hazards in Combinational Logic

Eliminate the 0-Hazards:

- Looping additional prime implicants that cover the adjacent 0’s that are not already covered by a common loop.
- The resulting equation is
  $F = (A+C)(A'+D')(B'+C'+D)\boxed{(C+D')(A+B'+D)(A'+B'+C')}$

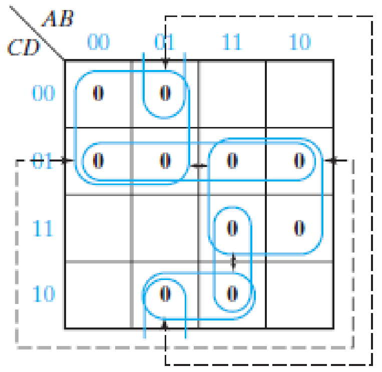


FIGURE 8-11
Karnaugh Map Removing Hazards of Figure 8-10

© Cengage Learning 2014

Hazards in Combinational Logic

Analysis on the Static and Dynamic Hazards:

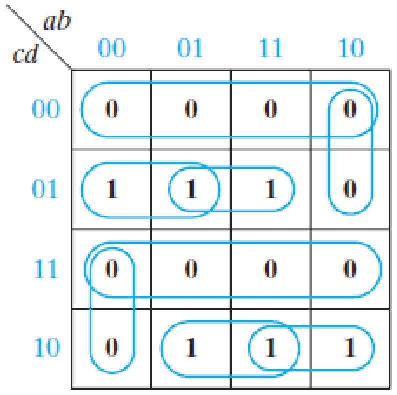
- Consider the function $f(a, b, c, d) = \sum m(1, 5, 6, 10, 13, 14)$.
- The output expression for the function f can be written as (complementation laws are not used here)

$$
\begin{aligned}
f &= (c'+ad'+bd')(c+a'd+bd) \\
&= cc' + acd' + bcd' + a'c'd + aa'dd' + a'bdd' + bc'd + abdd' + bdd' \\
&= \underline{cc'} + acd' + bcd' + a'c'd + aa'dd' + bc'd + \underline{bdd'}
\end{aligned}
$$

- Detection of static and dynamic hazards.
  - ✓ Static 1-hazard: The circuit does not contain any static 1-hazards because each pair of adjacent 1's are covered by one of the product terms.
  - ✓ Static 0-hazard: The cc' term can cause the output to temporarily become 1, this transition is a static 0-hazard.
  - ✓ Dynamic hazard: The term bdd' can cause a static 0- or dynamic hazard only if $b = 1$.

FIGURE 7-3

© Cengage Learning 2014



$f = a'c'd + bc'd + bcd' + acd'$

Hazards in Combinational Logic

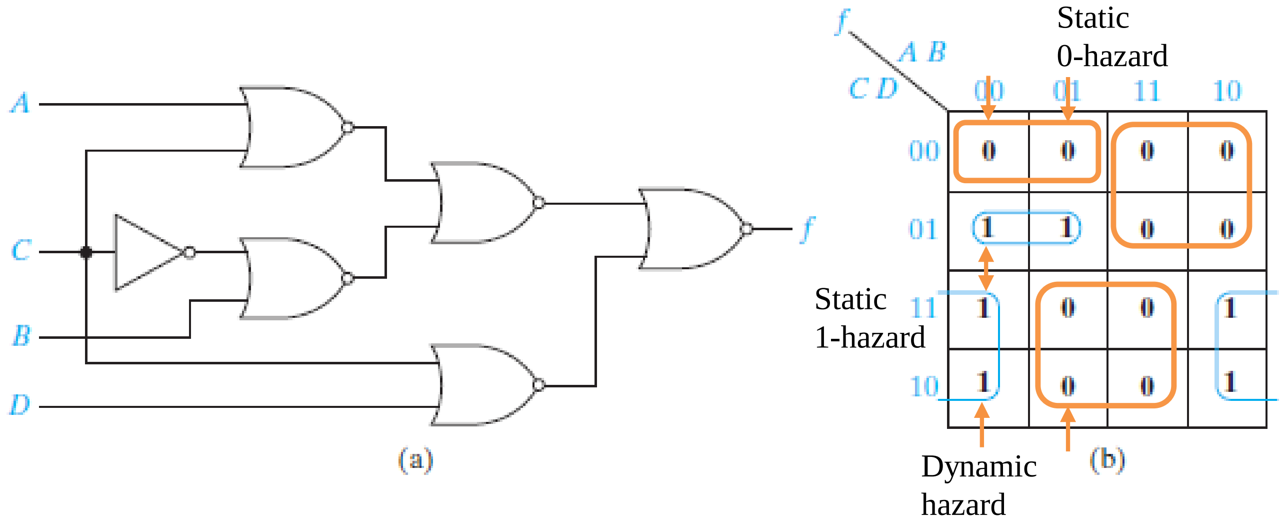
Finding static and dynamic hazards from a SOP expression

- The SOP expression for *f* in Figure 8-12(a) can be expressed as

$$f = (A'C' + B'C)(C + D) = A'CC' + A'C'D + B'C$$

- Static 1-hazard: Two input combination between (0,0,0,1) and (0,0,1,1) imply a static 1-hazard for C changing.
- The product *A'CC'* in *f* indicate the possibility of a static 0-hazard and a dynamic hazard for *A* = 0 and *C* Changing.
  - ✓ Static 0-hazard: Two input combination between (0,1,0,0) and (0,1,1,0)
  - ✓ Dynamic hazard: only occurs for (0,0,0,0) and (0,0,1,0)

FIGURE 8-12
Hazard Example



Hazards in Combinational Logic

Procedure to Design a Circuit without Static and Dynamic Hazards:

- Find a sum-of-products expression (F^t) for the output in which every pair of adjacent 1's is covered by a 1-term. (The sum of all prime implicants will always satisfy this condition.) A two-level AND-OR circuit based on this F^t will be free of 1-, 0-, and dynamic hazards.
- If a different form of the circuit is desired, manipulate F^t to the desired form by simple factoring, DeMorgan's laws, etc. Treat each x_i and x_i' as independent variables to prevent introduction of hazards.

Simulation and Testing of Logic Circuits

Simulation and Verification of Logic Circuits:

- An important part of the logic design process is verifying that the final design is correct and debugging the design if necessary.
- Simulation on a computer is generally an easy, fast, and more economical way to verify a circuit's output.
- To use a computer program for simulating logic circuits, you must first specify the circuit components and connections; then, specify the circuit inputs; and, finally, observe the circuit outputs
- A typical simulator which runs on a personal computer uses switches or input boxes to specify the inputs and probes to read the logic outputs. Alternatively, the inputs and outputs may be specified as sequences of 0's and 1's or in the form of timing diagrams.

Simple Simulator for Combinational Logic:

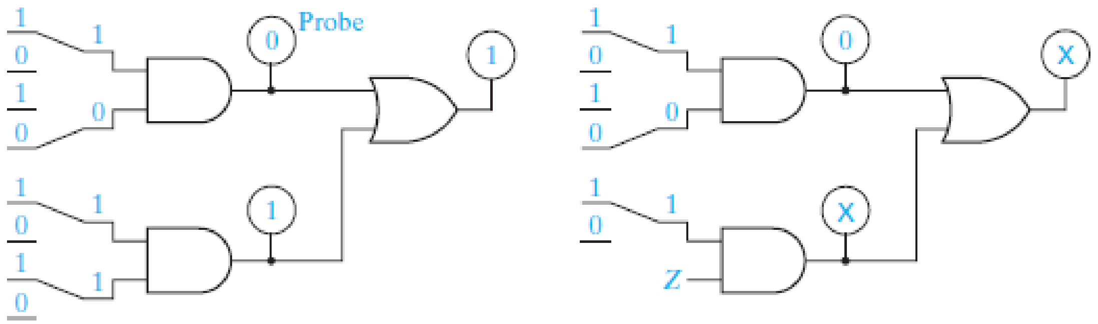
1. The circuit inputs are applied to the first set of gates in the circuit, and the outputs of those gates are calculated.
2. The outputs of the gates that changed are fed into the next level of gate inputs. If the input to any gate has changed, then the output of that gate is calculated.
3. Step 2 is repeated until no more changes in gate inputs occur. The circuit is then in a steady-state condition, and the outputs may be read.
4. Steps 1 through 3 are repeated every time a circuit input changes.

Simulation and Testing of Logic Circuits

Typical Simulation Screens:

- Use switches or input boxes to specify the inputs
- Use probes to read the logic outputs

FIGURE 8-13



(a) Simulation screen showing switches (b) Simulation screen with missing gate input

AND and OR Functions for Four-Valued Simulation:

- For an AND gate, if one of the input is 0, the output is always 0.
- For an OR gate, if one of the input is 1, the output is 1 regardless of the other input.
- *X* represents unknown value.
- *Z* represents an open circuit, or *high impedance* (hi-Z)

TABLE 8-1 AND and OR Functions for Four-Valued Simulation

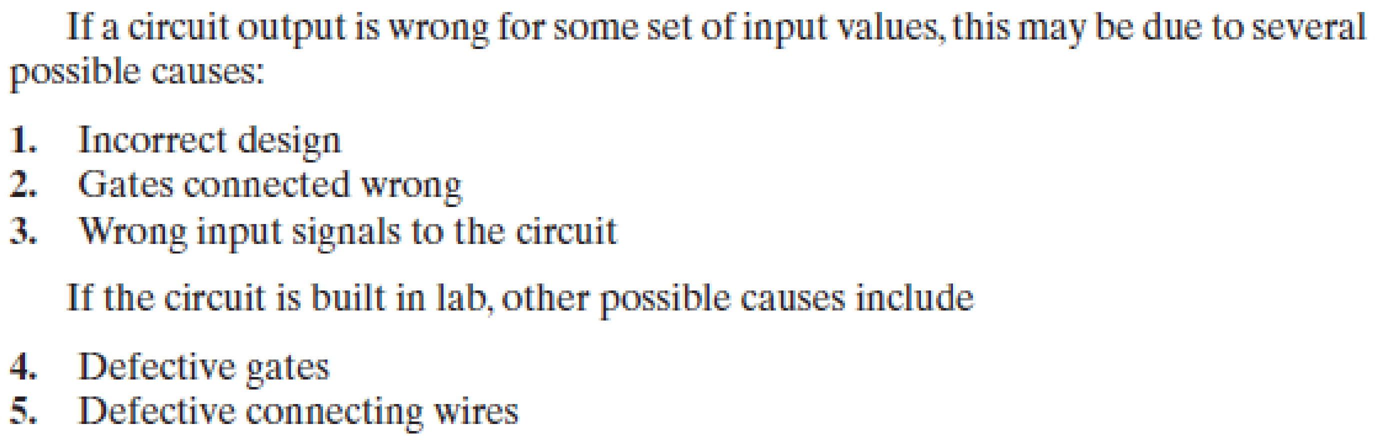
·	0	1	X	Z
0	0	0	0	0
1	~~0~~	~~1~~	~~X~~	~~X~~
X	0	X	X	X
Z	0	X	X	X

+	0	1	X	Z
0	~~0~~	1	X	X
1	~~1~~	1	1	1
X	~~X~~	1	X	X
Z	X	1	X	X

성균관대학교 SUNG KYUN KWAN UNIVERSITY

Simulation and Testing of Logic Circuits

Causes for Error in Simulation and Physical Building of Circuits:



If a circuit output is wrong for some set of input values, this may be due to several possible causes:

1. Incorrect design
2. Gates connected wrong
3. Wrong input signals to the circuit

If the circuit is built in lab, other possible causes include

4. Defective gates
5. Defective connecting wires

성균관대학교
SUNG KYUN KWAN UNIVERSITY

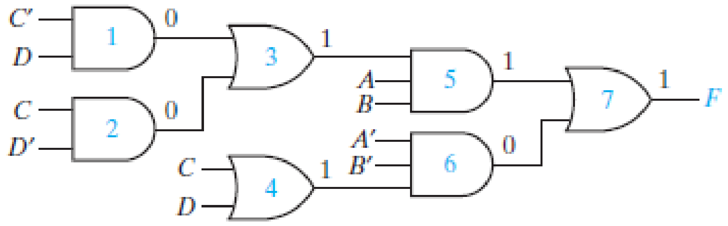
Simulation and Testing of Logic Circuits

Example 3: Troubleshoot a logic circuit

Example

The function $F = AB(C'D + CD') + A'B'(C + D)$ is realized by the circuit of Figure 8-14:

FIGURE 8-14 Logic Circuit with Incorrect Output



A student builds the circuit in a lab and finds that when $A = B = C = D = 1$, the output F has the wrong value, and that the gate outputs are as shown in Figure 8-14. The reason for the incorrect value of F can be determined as follows:

1. The output of gate 7 (F) is wrong, but this wrong output is consistent with the inputs to gate 7, that is, $1 + 0 = 1$. Therefore, one of the inputs to gate 7 must be wrong.
2. In order for gate 7 to have the correct output ($F = 0$), both inputs must be 0. Therefore, the output of gate 5 is wrong. However, the output of gate 5 is consistent with its inputs because $1 \cdot 1 \cdot 1 = 1$. Therefore, one of the inputs to gate 5 must be wrong.
3. Either the output of gate 3 is wrong, or the A or B input to gate 5 is wrong. Because $C'D + CD' = 0$, the output of gate 3 is wrong.
4. The output of gate 3 is not consistent with the outputs of gates 1 and 2 because $0 + 0 \neq 1$. Therefore, either one of the inputs to gate 3 is connected wrong, gate 3 is defective, or one of the input connections to gate 3 is defective.

This example illustrates how to troubleshoot a logic circuit by starting at the output gate and working back until the wrong connection or defective gate is located.