UNIT 8
COMBINATIONAL CIRCUIT DESIGN AND SIMULATION

Spring 2011

Combinational Circuit Design & Simulation

- **Contents**
  - Review of combinational circuit design
  - Design of circuits with limited gate fan-in
  - Gate delay and timing diagrams
  - Hazards in combinational logic
  - Simulation and testing of logic circuits
- **Reading**
  - Unit 8

Review of Combinational Circuit Design

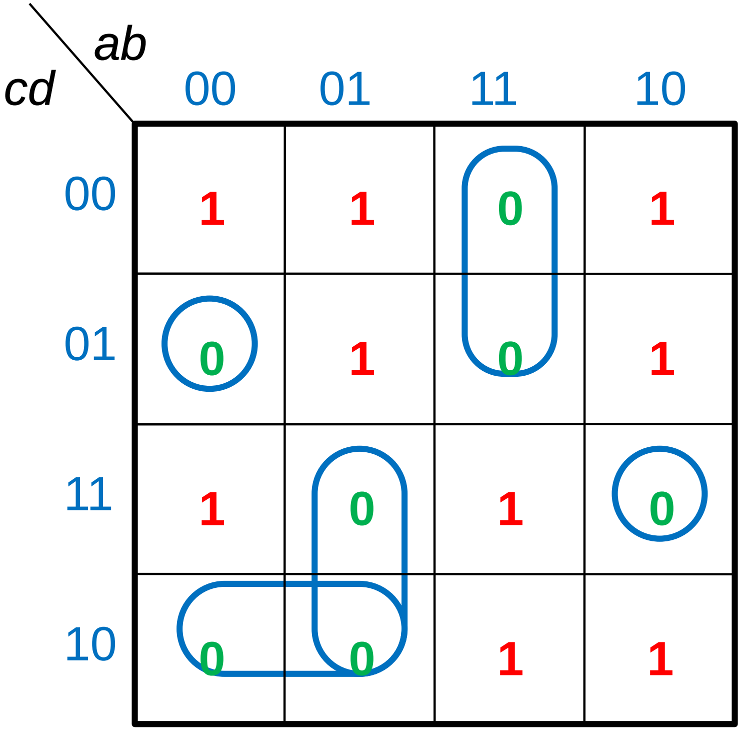
- **Generic combinational circuit design steps:**
  1. Specify the switching function(s)
     - A truth table
     - Algebraic equations
  2. Derive simplified algebraic expression(s)
     - Karnaugh maps
     - Algebraic simplification (large # of variables, small # of terms)
  3. Manipulate into the proper form
     - Apply algebraic simplification with considering the gate types used in circuit implementation
- **Minimum SOP (POS) ⇒ minimum two-level circuits**
  - Start from min SOP ≡ AND-OR ⇒NAND-NAND/OR-NAND/NOR-OR
  - Start from min POS ≡ OR-AND ⇒NOR-NOR/AND-NOR/NAND-AND
- **The # of level = the max # of cascaded gates between I/Os**
  - Change level by factoring or multiplying out
  - ⇒ Change # of gates or gate inputs
- **Design a multi-level multi-output ckt from a ckt of AND & OR gates**

Limited Gate Fan-in

For a gate, fan-in = # of inputs
For a circuit, fan-in = max # of inputs on each gate

Example 1

□ **Realize $f(a, b, c, d) = \Sigma\, m(0, 3, 4, 5, 8, 9, 10, 14, 15)$ using 3-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

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

⇒ two-level circuit:

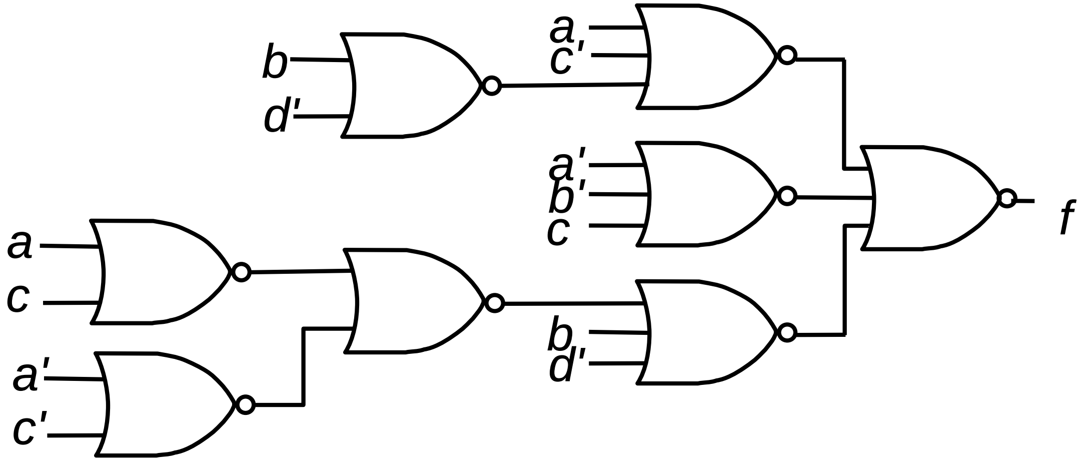
2 4-input AND (X) ⇒ factoring

3 3-input AND (O)

1 5-input OR (X) ⇒ sum of 3 items

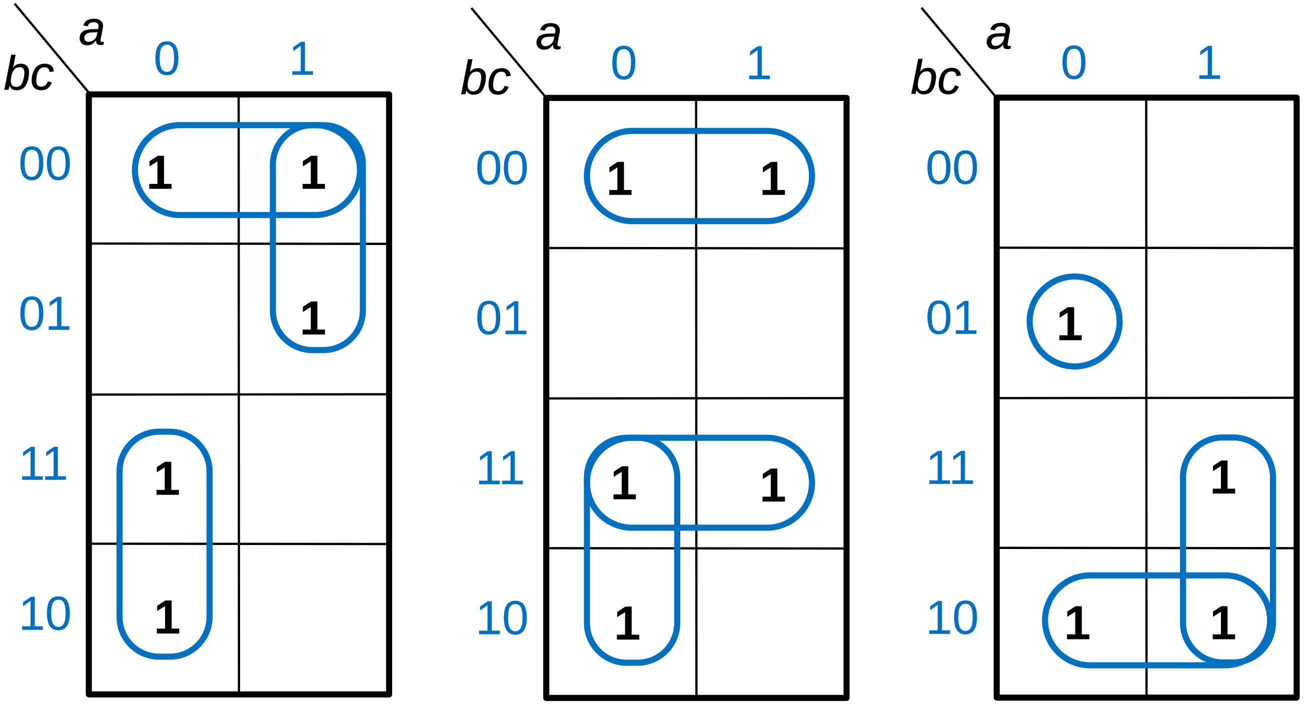
⇒ $f' = b'd(a'c' + ac) + a'c(b + d') + abc'$

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



Example 2 (1/2)

- Realize the functions using only 2-input NAND gates and inverters



bc \ a	0	1
00	1	1
01		1
11	1	
10	1	

bc \ a	0	1
00	1	1
01		
11	1	1
10	1	

bc \ a	0	1
00		
01	1	
11		1
10	1	1

$f_1 = \Sigma\, m(0,2,3,4,5)$

$f_1 = b'c' + ab' + a'b$

$f_2 = \Sigma\, m(0,2,3,4,7)$

$f_2 = b'c' + bc + a'b$

$f_3 = \Sigma\, m(1,2,6,7)$

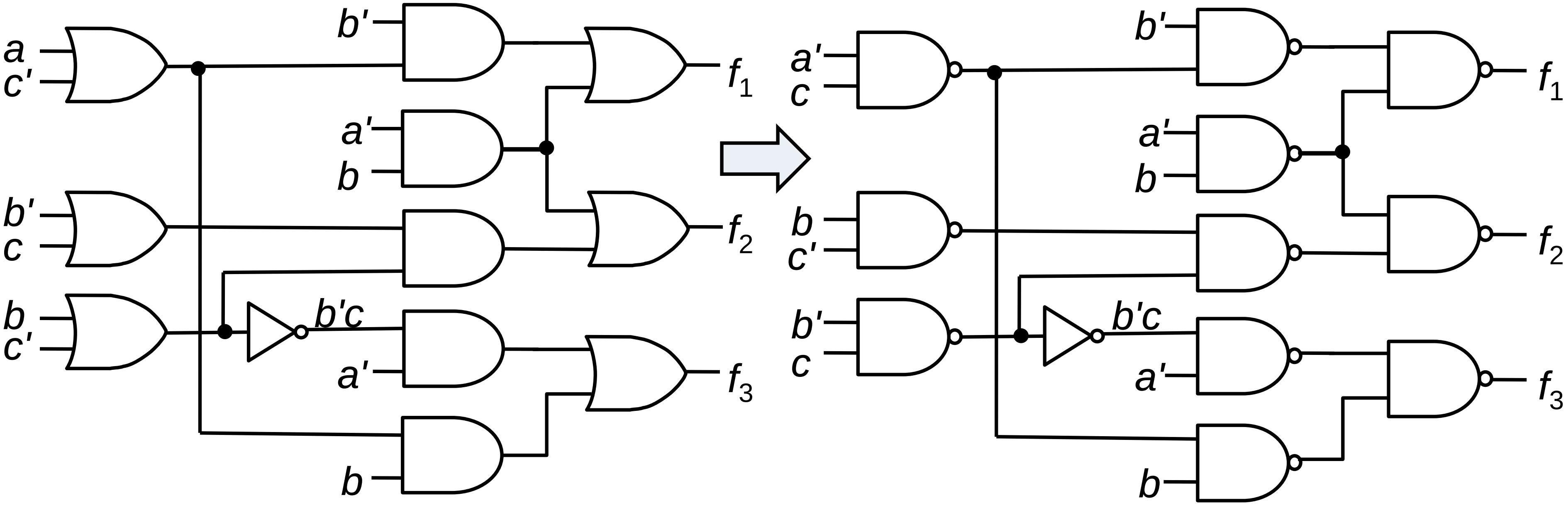
$f_3 = a'b'c + ab + bc'$

Example 2 (2/2)

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

$f_1 = b'(a + c') + a'b$
$f_2 = b'c' + b(a' + c)$ or $f_2 = (b' + c)(b + c') + a'b$
$f_3 = a'(b + c')' + b(a + c')$

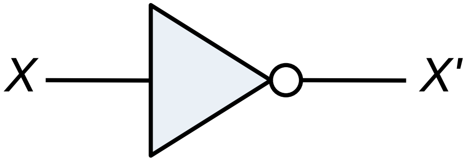
$a'b'c = a'(b'c) = a'(b + c')'$

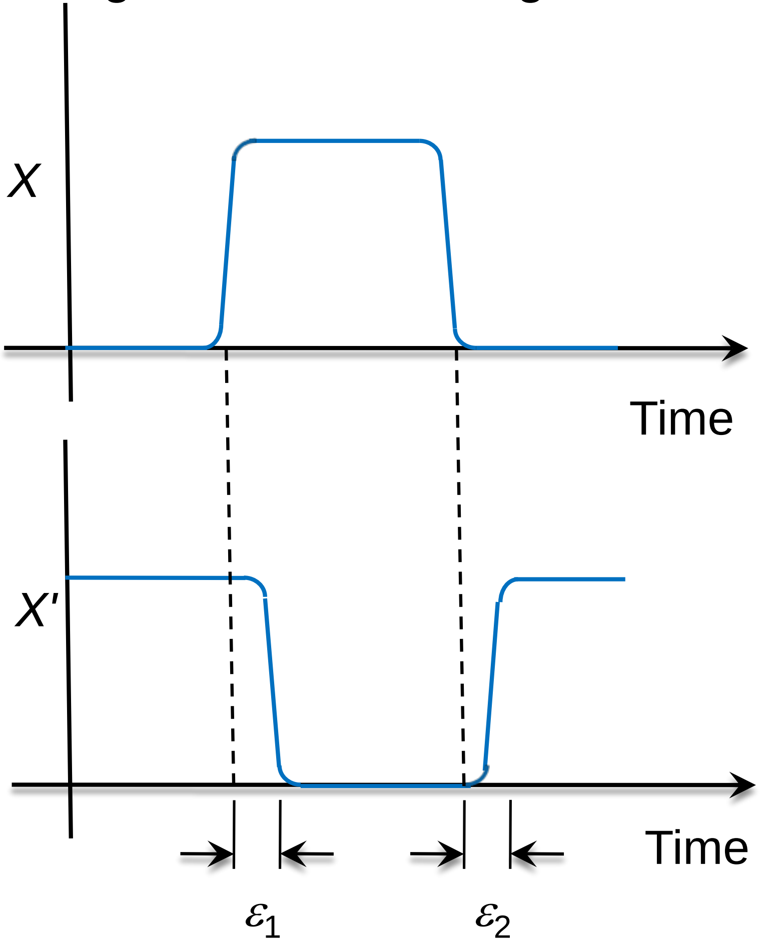


Gate Delays and Timing Diagrams

Gate Delay

- **Propagation delay: when the input to a logic gate is changed, the output takes some time to change value**
  - Usually different for input rising 0 → 1 and falling 1 → 0

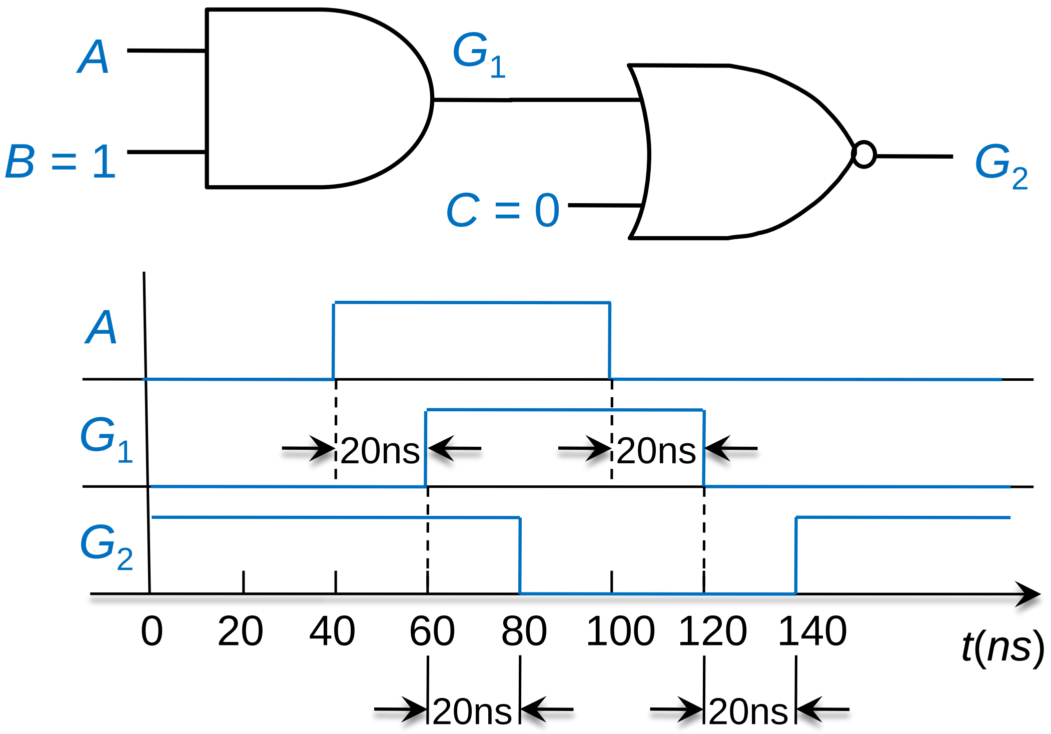




A few ns
(nanosecond = 10^{-9} sec)

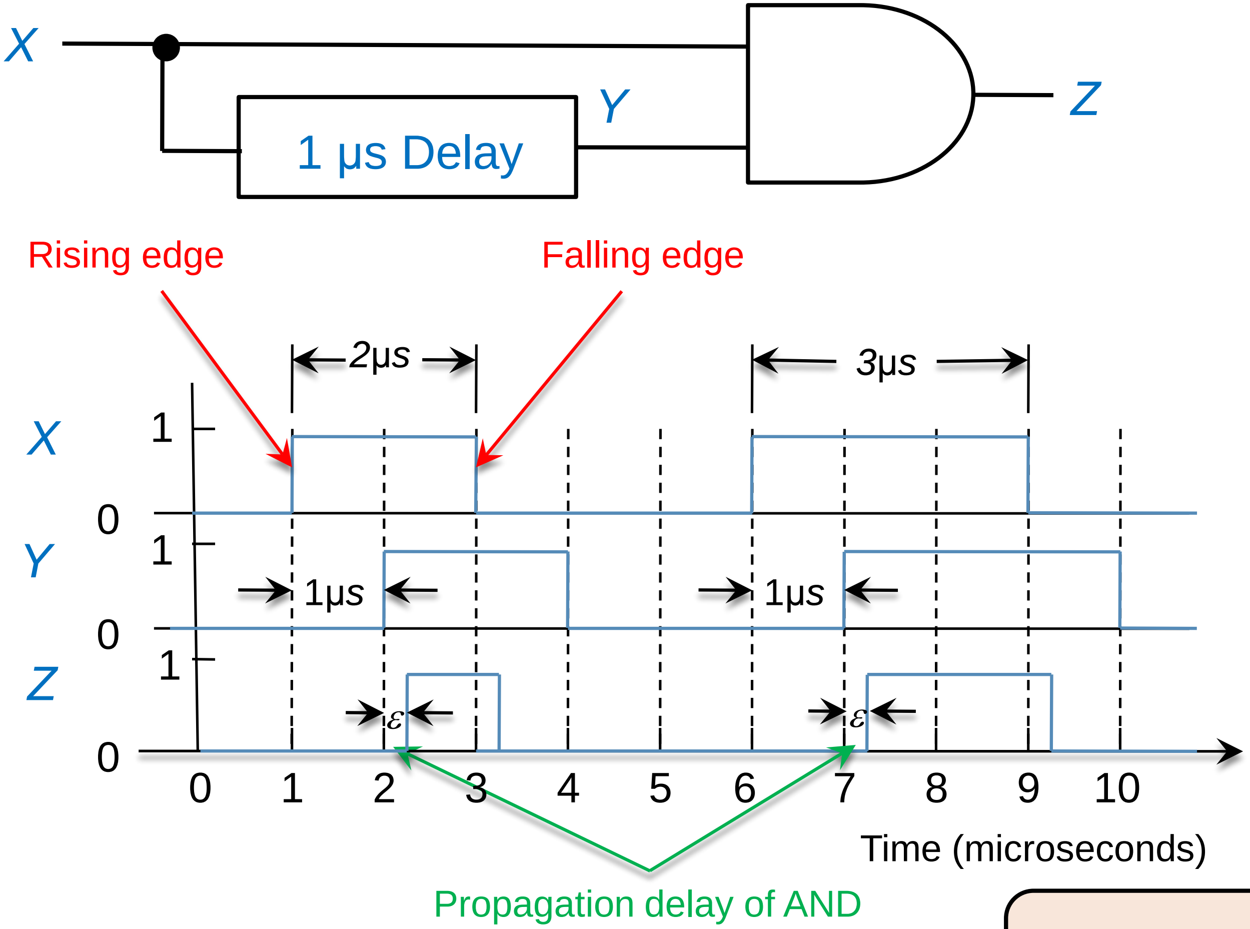
Sample Timing Diagram (1/2)

- **e.g., assume each gate has a propagation delay of 20 ns**



Sample Timing Diagram (2/2)

- **e.g., a circuit with a delay element**

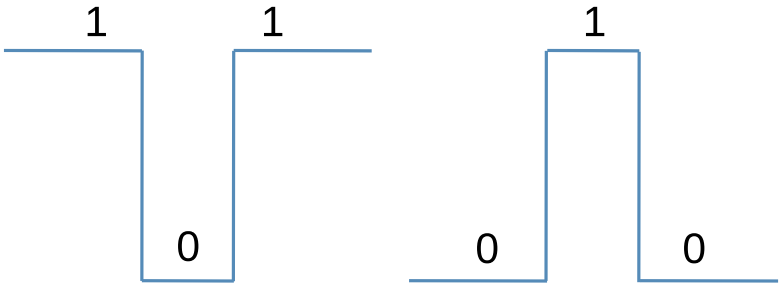


μs
(microsecond = 10^{-6} sec)

Hazards

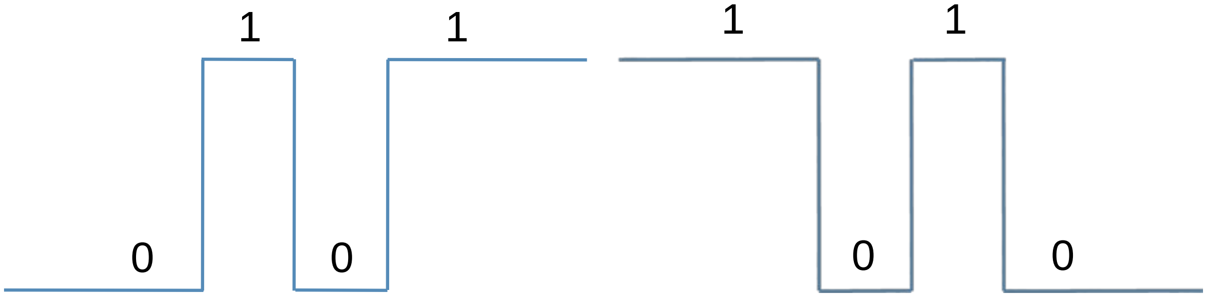
Hazards

- **Hazard: unwanted switching transients appearing at the output when the input to a combinational ckt changes**
  - Static 1-/0-hazard: output momentarily goes to 0/1 when it should remain a constant 1/0



(a) Static 1-hazard (b) Static 0-hazard

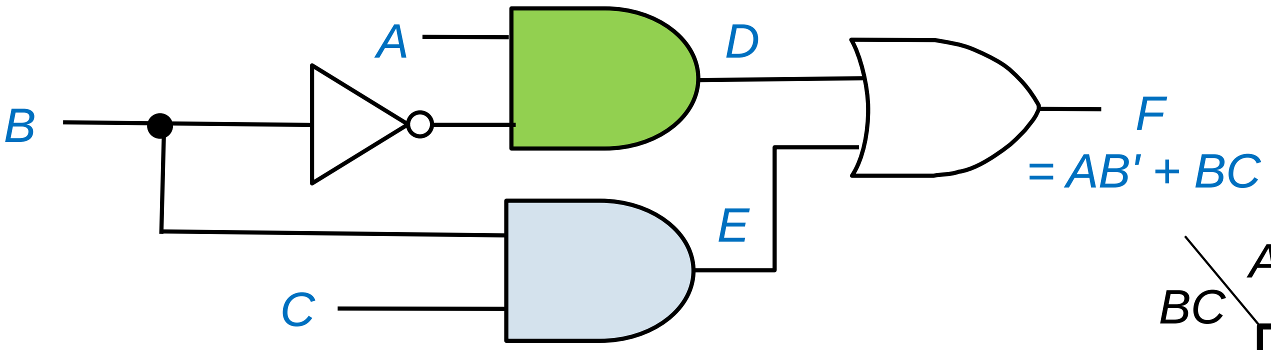
  - Dynamic hazard: output change 3 or more times when the output changes from 0 to 1 (1 to 0)

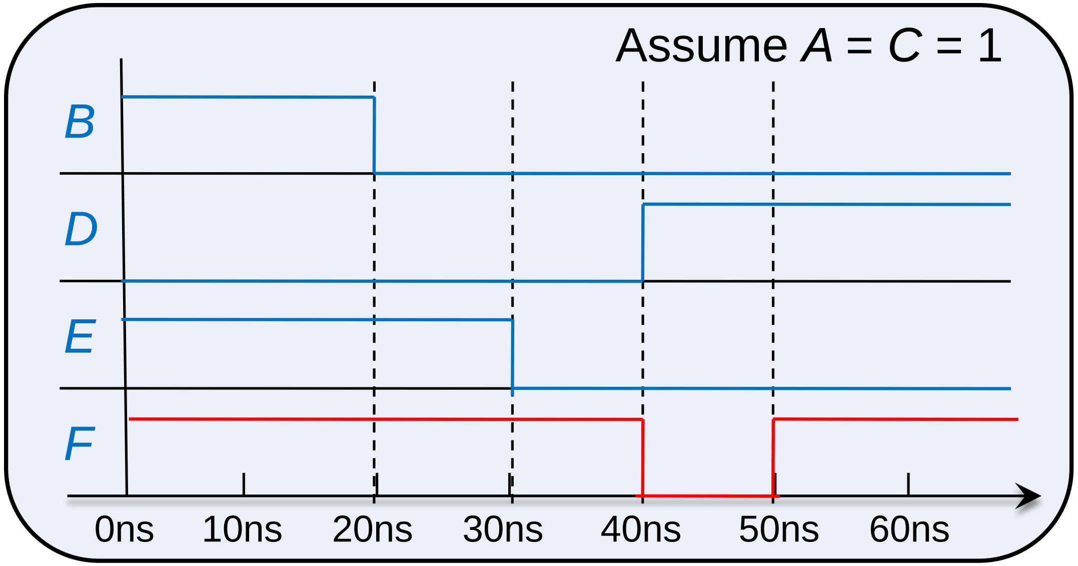


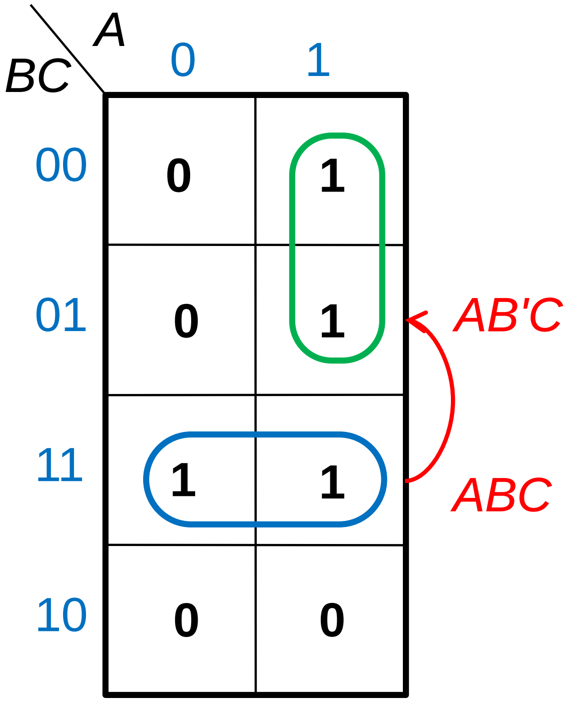
(c) Dynamic hazard

1-Hazards? When?

- **e.g., assume each gate has a propagation delay of 10 ns**
  If *A* = *C* = 1, static-1 hazard occurs when *B* changes from 1 to 0





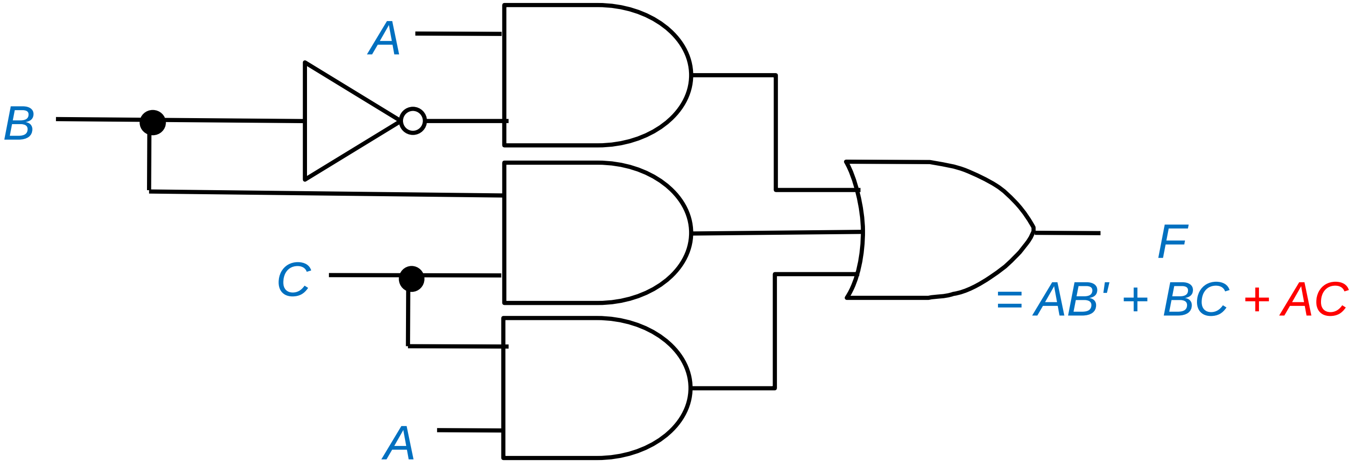


BC \ A	0	1
00	0	1
01	0	1
11	1	1
10	0	0

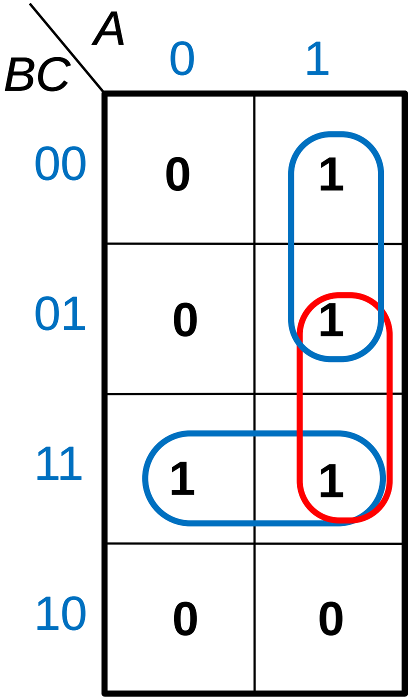
1-Hazard Removal for 2-Level AND/OR Circuit

- Q: When?
- A: In Karnaugh maps, 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
- Q: How to remove hazards?
- A: Add additional loops s.t. all adjacent 1's are covered by some loop

Consider only one-input change!



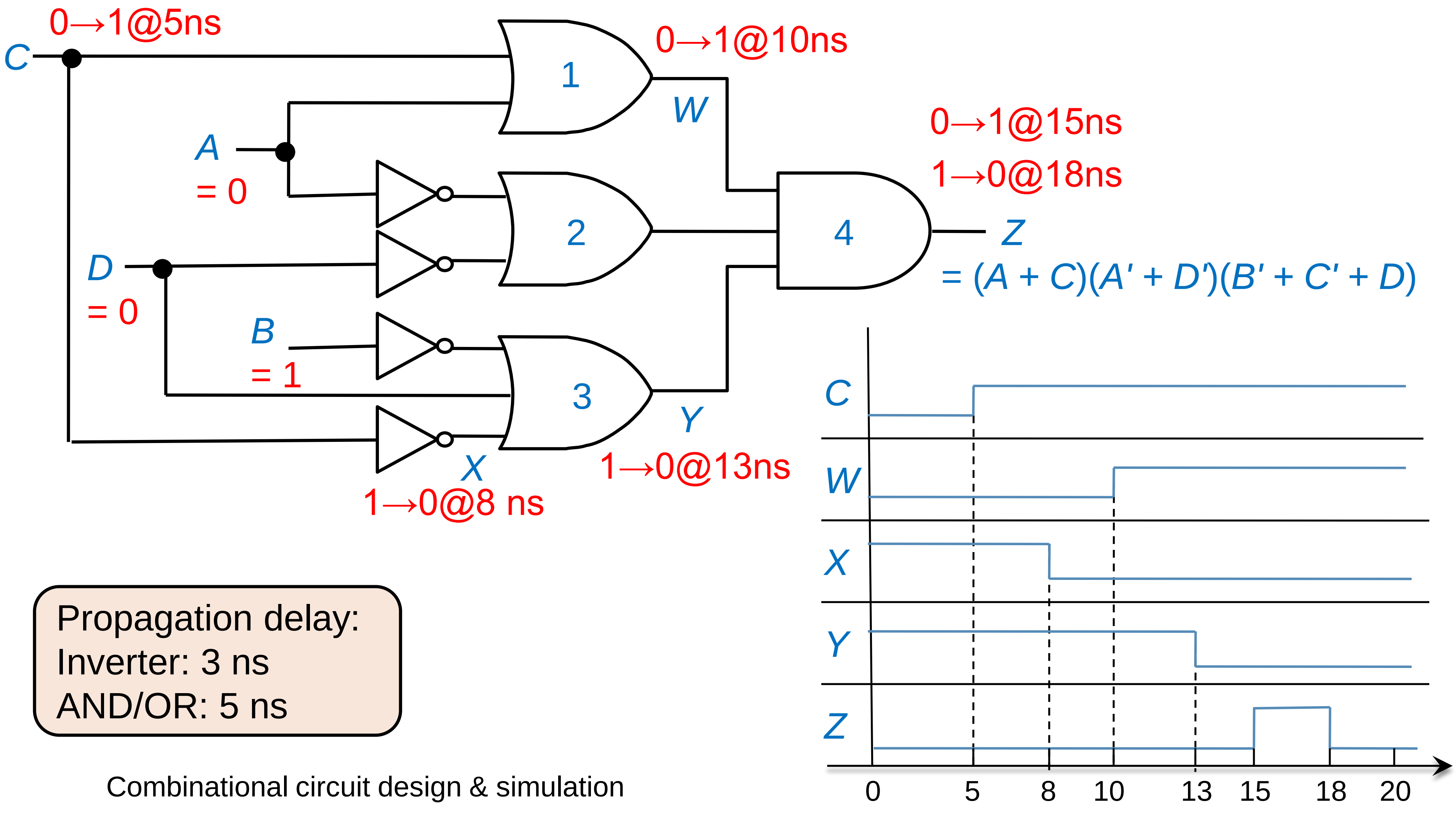
$= AB' + BC + AC$



BC \ A	0	1
00	0	1
01	0	1
11	1	1
10	0	0

0-Hazards? When?

□ **e.g., consider *A* = 0, *B* = 1, *D* = 0, *C* changes from 0 to 1**



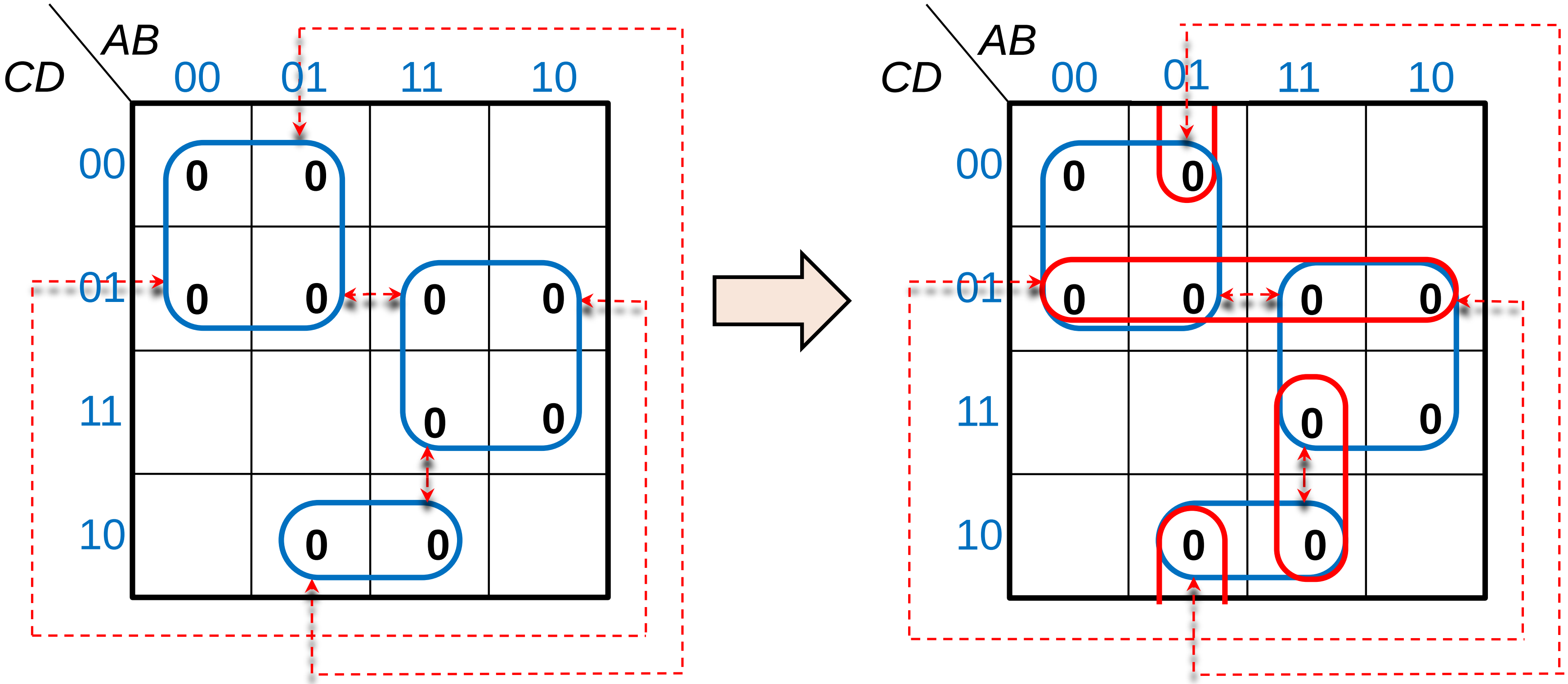
$Z = (A + C)(A' + D')(B' + C' + D)$

Propagation delay:
Inverter: 3 ns
AND/OR: 5 ns

0-Hazard Removal for 2-Level AND/OR Circuit

- Similarly, add additional loops for adjacent 0’s
- e.g,

Consider only one-input change!



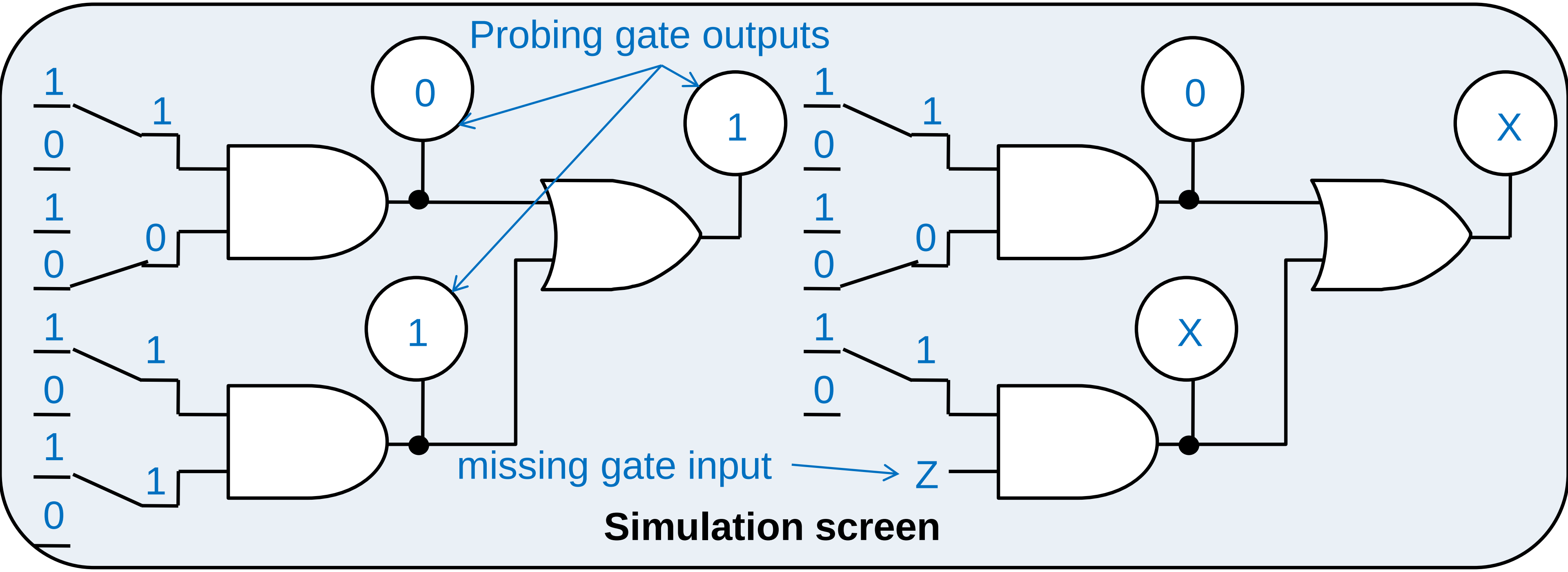
$Z = (A + C)(A' + D')(B' + C' + D)$

$Z = (A + C)(A' + D')(B' + C' + D)$
$(C + D')(A + B' + D)(A' + B' + C')$

Simulation

Simulation

- **Verify logic circuits by**
  - Actually building them
  - Simulating them on a computer
    - Computations are performed stage-by-stage from inputs
    - Evaluations are performed every time inputs change
- **4 logic values:** 0 (low), 1 (high), X (unknown), Z (high-Z/open ckt)



4-Valued Logic Simulator

- **AND and OR functions for 4-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

- **Possible errors in designs**
  - In simulation
    - Incorrect design
    - Gates connected wrong
    - Wrong input signals to the circuits
  - In real circuits
    - Defective gates
    - Defective connecting wires