

EE2000 Logic Circuit Design

Combinational Logic Circuit Design

Outline

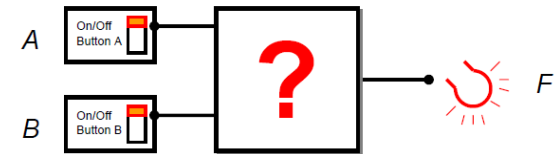
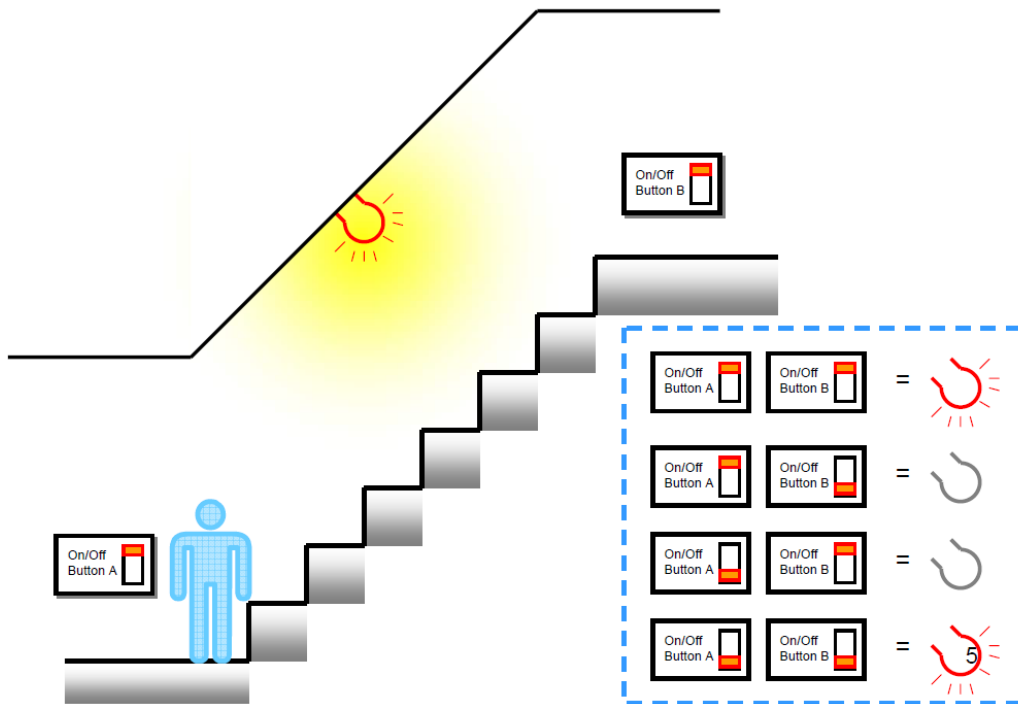
- 4.1 Design Procedure
- 4.2 Examples
 - Bi-switch lighting controller
 - Code converter
 - 7-segment display
- 4.3 Timing Hazard

4.1 Design Procedure

- 1. State the problem / specification for the design.
- 2. Determine the number of input variables and output variables.
- 3. Formulate truth tables / Boolean functions between inputs and outputs.
- 4. Simplification / minimization for the logic functions.
- 5. Design and draw the logic circuit diagram.

4.2 Examples

- Bi-switch lighting controller

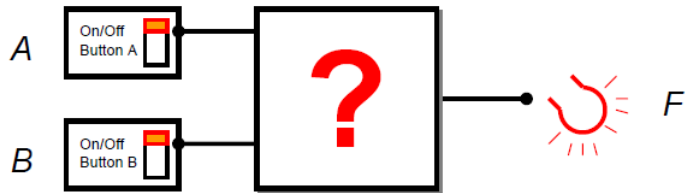


State the case

Design a circuit to control the bulb

- The light turns on when both buttons are turned UP / DOWN.
- The light turns off when both buttons swap in different positions.
- ON / OFF light is a binary decision output.
- Button positions are the inputs (variables)

Formulation



Assume that two variables **A** and **B** are the button positions. And **F** is binary decision output of **A** and **B**.

0 is the button turns UP.

1 is the button turns DOWN.

Inputs		Output
A	B	F
0	0	1
0	1	0
1	0	0
1	1	1



$F(A, B)$ is 1 if $(A = 0 \text{ AND } B = 0)$ OR $(A = 1 \text{ AND } B = 1)$

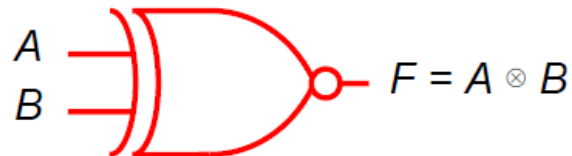
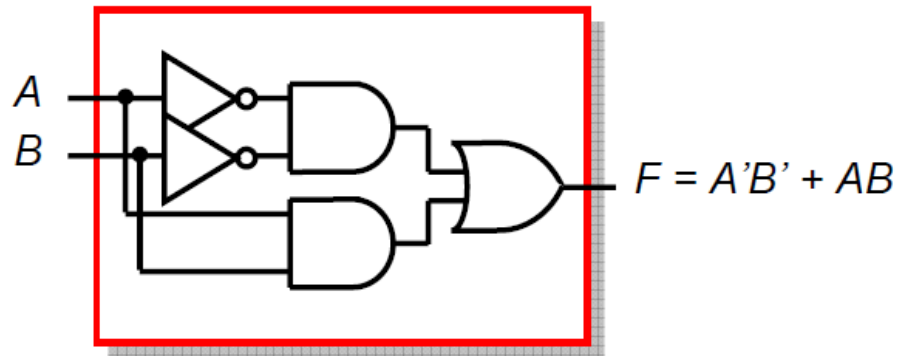
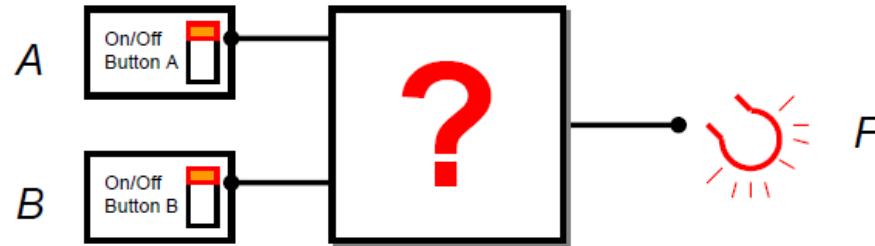
i.e. $F(A, B) = A'B' + AB = \sum m(0, 3)$

■ Optimization:

■ From the truth table,

■ $F(A, B) = AB + A'B' (= A \otimes B)$

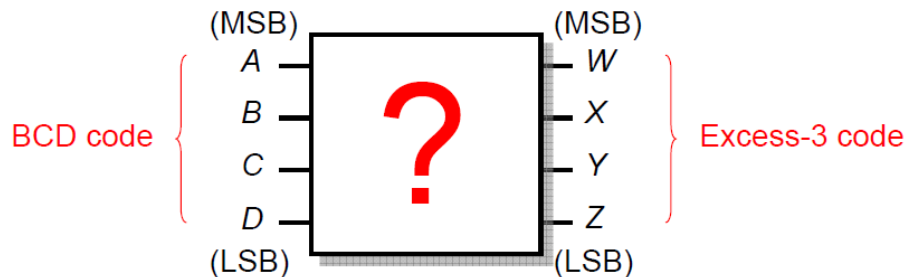
logic circuit diagram



4.2 Examples

- Code converter

■ Design a logic circuit that perform code conversion



- Input is BCD 8421 code
- Output is Excess-3 code

State the case

Design a circuit to convert the BCD to the Excess-3 code

- A, B, C, D are the input of BCD.
- W, X, Y, Z are the output of Excess-3 code.
- The output functions are:
 $W(A, B, C, D)$
 $X(A, B, C, D)$
 $Y(A, B, C, D)$
 $Z(A, B, C, D)$

Formulation

Decimal digit	Input (8421 code)				Output (Excess-3 code)			
	A	B	C	D	W	X	Y	Z
0	0	0	0	0	0	0	1	1
1	0	0	0	1	0	1	0	0
2	0	0	1	0	0	1	0	1
3	0	0	1	1	0	1	1	0
4	0	1	0	0	0	1	1	1
5	0	1	0	1	1	0	0	0
6	0	1	1	0	1	0	0	1
7	0	1	1	1	1	0	1	0
8	1	0	0	0	1	0	1	1
9	1	0	0	1	1	1	0	0
Unused	X	X	X	X	X	X	X	X
Unused	X	X	X	X	X	X	X	X
Unused	X	X	X	X	X	X	X	X
Unused	X	X	X	X	X	X	X	X
Unused	X	X	X	X	X	X	X	X
Unused	X	X	X	X	X	X	X	X

Unused outputs can consider as DON'T CARE.

Formulation

- There are four variables (A, B, C, D) in the functions
- Each output variable depends on 4 variables
- So we need 4 four-variable K-maps
 - $W(A, B, C, D) = \sum m(5, 6, 7, 8, 9) + \sum d(10, 11, 12, 13, 14, 15)$
 - $X(A, B, C, D) = \sum m(1, 2, 3, 4, 9) + \sum d(10, 11, 12, 13, 14, 15)$
 - $Y(A, B, C, D) = \sum m(0, 3, 4, 7, 8) + \sum d(10, 11, 12, 13, 14, 15)$
 - $Z(A, B, C, D) = \sum m(0, 2, 4, 6, 8) + \sum d(10, 11, 12, 13, 14, 15)$

Minimization

K-map for W:

CD \ AB	00	01	11	10
	00		x	1
	01		1	x
	11		1	x
	10		1	x

K-map for X:

CD \ AB	00	01	11	10
	00		1	x
	01	1		x
	11	1		x
	10	1		x

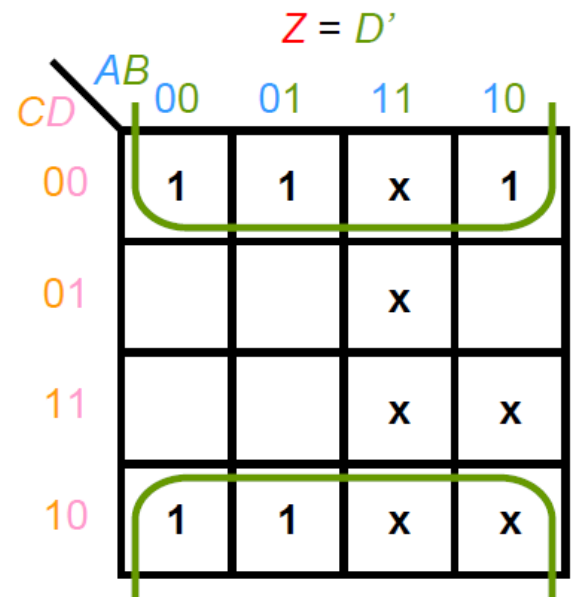
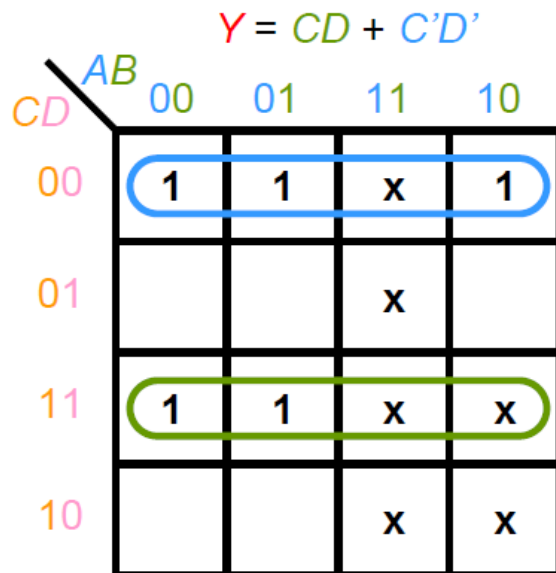
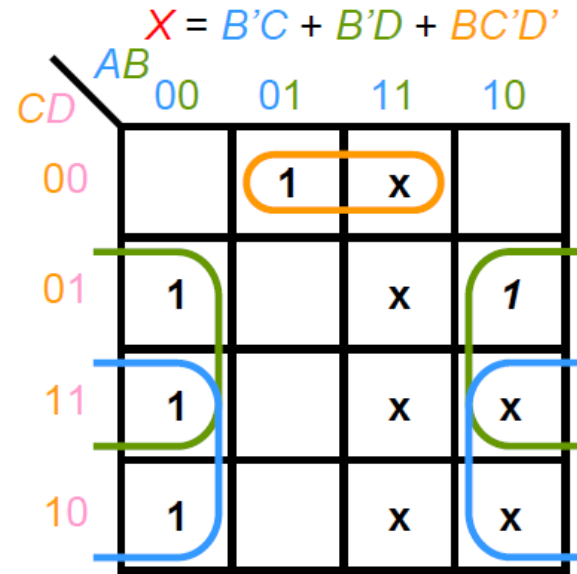
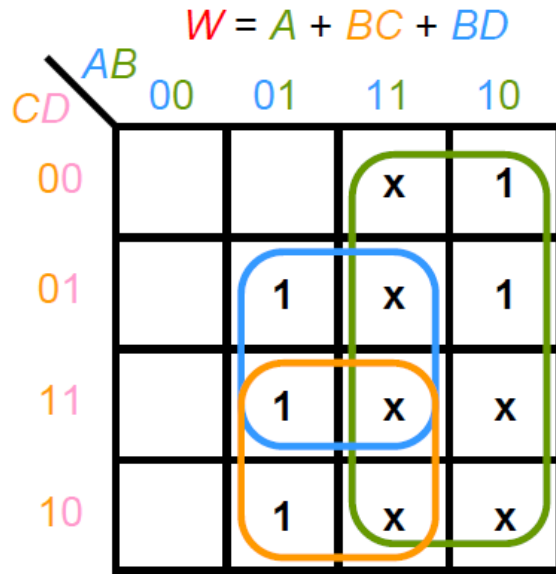
K-map for Y:

CD \ AB	00	01	11	10
	00	1	1	x
	01			x
	11	1	1	x
	10			x

K-map for Z:

CD \ AB	00	01	11	10
	00	1	1	x
	01			x
	11	1	1	x
	10			x

Minimization



Logic functions and circuit

■ From the pervious K-maps,

■ $W = A + BC + BD$

■ $X = B'C + B'D + BC'D'$

■ $Y = CD + C'D'$

■ $Z = D'$

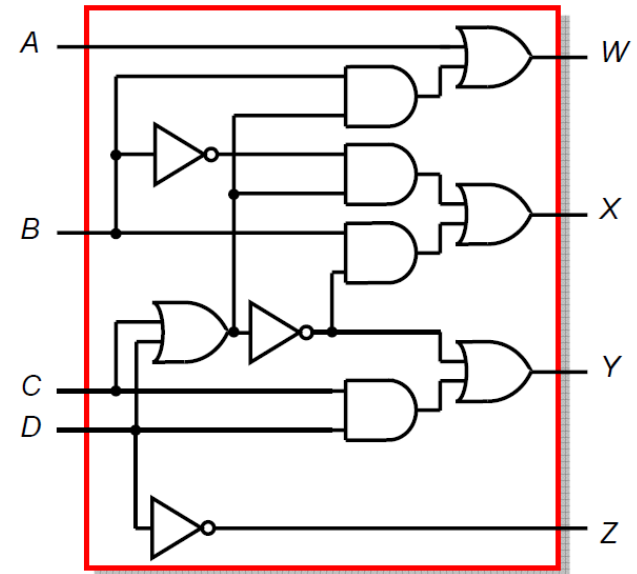
■ We further optimize them (optional)

■ $W = A + BC + BD = A + B(C+D)$

■ $X = B'C + B'D + BC'D' = B'(C+D) + B(C+D)'$

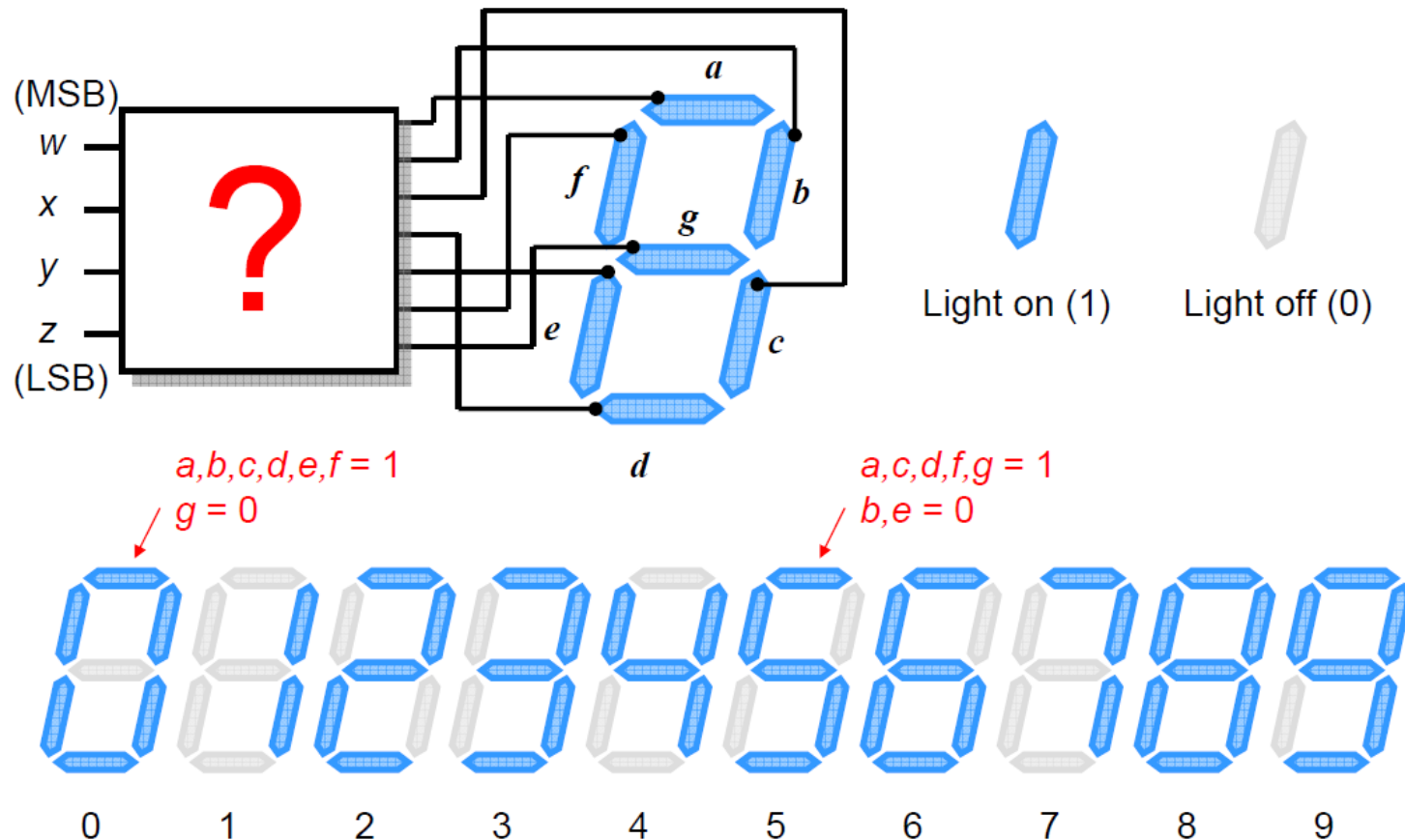
■ $Y = CD + C'D' = CD + (C+D)'$

DeMorgan's theorem



4.2 Examples

- 7-segment display



Formulation

Input BCD				Seven-Segment Display						
<i>w</i>	<i>x</i>	<i>y</i>	<i>z</i>	<i>a</i>	<i>b</i>	<i>c</i>	<i>d</i>	<i>e</i>	<i>f</i>	<i>g</i>
0	0	0	0	1	1	1	1	1	1	0
0	0	0	1	0	1	1	0	0	0	0
0	0	1	0	1	1	0	1	1	0	1
0	0	1	1	1	1	1	1	0	0	1
0	1	0	0	0	1	1	0	0	1	1
0	1	0	1	1	0	1	1	0	1	1
0	1	1	0	1	0	1	1	1	1	1
0	1	1	1	1	1	1	0	0	0	0
1	0	0	0	1	1	1	1	1	1	1
1	0	0	1	1	1	1	1	0	1	1
1	0	1	0	x	x	x	x	x	x	x
1	0	1	1	x	x	x	x	x	x	x
1	1	0	0	x	x	x	x	x	x	x
1	1	0	1	x	x	x	x	x	x	x
1	1	1	0	x	x	x	x	x	x	x
1	1	1	1	x	x	x	x	x	x	x

Note: The corresponding outputs of those unused BCD code should be either 0 or don't care (depends on the specification)

Formulation

- Number of variables (input)?
- Number of functions (output)?
- Number of K-map?
- Example of functions:
- $$a(w, x, y, z) = \sum m(0, 2, 3, 5, 6, 7, 8, 9) + \sum d(10, 11, 12, 13, 14, 15)$$

K-maps of segment-*a*

	wx	00	01	11	10
yz	00	1		x	1
01			1	x	1
11		1	1	x	x
10		1	1	x	x

	wx	00	01	11	10
yz	00	1		x	1
01			1	x	1
11		1	1	x	x
10		1	1	x	x

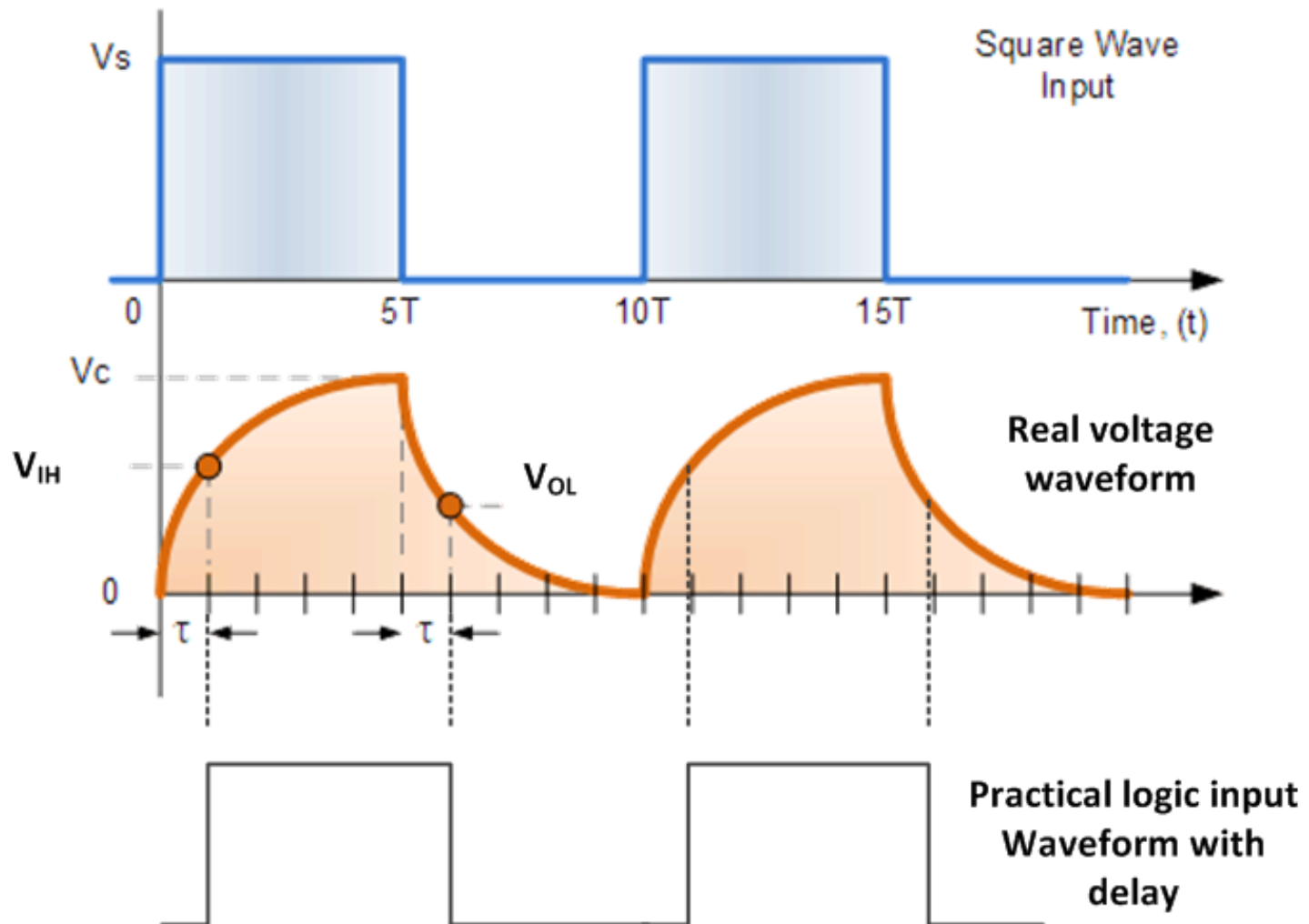
$$a(w, x, y, z) = w + y + xz + x'z'$$

Exercise: find the simplified expression of *b*, *c*, *d*, *e*, *f* and *g*

4.3 Timing Hazard

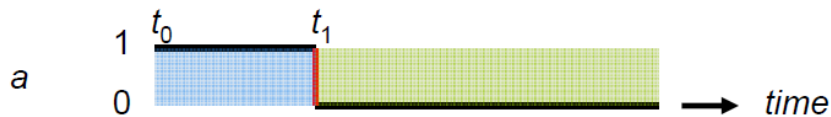
- Logic devices (gates or other more complex circuits) are essentially made from semiconductor
- IC input impedance and PCB copper track inductance cause charging and discharging effect (similar to RC circuit response)
- The “real” input and output voltages are not a perfect step function
- Practical logic input and output waveforms exhibit “delay” nature

I/P and O/P waveform

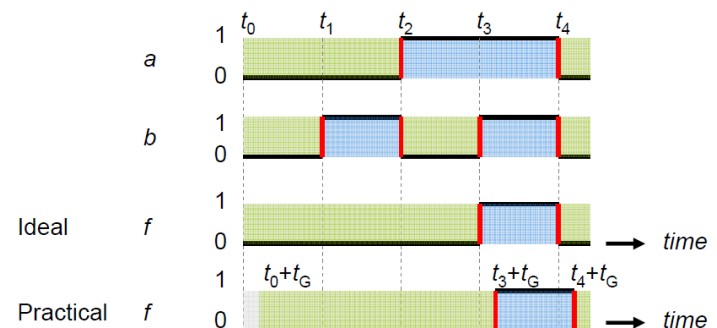
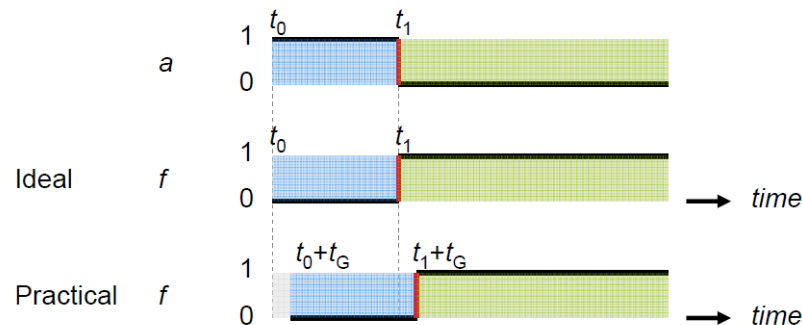
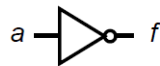


Timing Diagram

- The horizontal axis represents time
- The vertical axis shows a signal between 2 possible voltage levels



- When time is between t_0 to t_1 , a is 1
- At time t_1 , a is **changing** from 1 to 0
- When time is after t_1 , a becomes 0



Timing Hazards

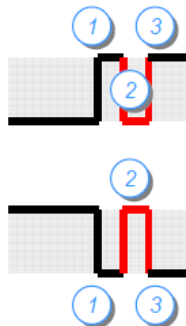
- The circuit output may produce a short pulse (which should not be happened) at the transient time caused by delay in propagation of signals
- Two kinds of hazards: **static** and **dynamic**



Static-0 hazard: the output may momentarily go to 1 when it should *remain 0*



Static-1 hazard: the output may momentarily go to 0 when it should *remain 1*

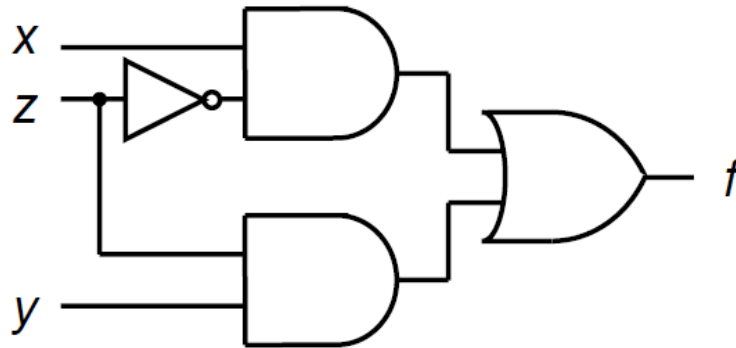


Dynamic hazard: The output changes three or more times when it should change from 1 to 0 or from 0 to 1 *only once*

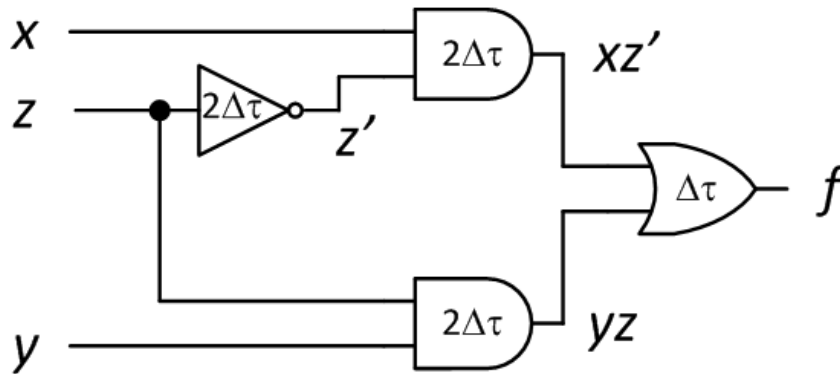
Timing Hazard

Gate Delay

- Output waveform is shifted t_G time units
- The length of time for an input change, to result in the corresponding output change



- The propagation delay could cause undesirable events: **timing hazards**



Assuming that the delay of the gates are different as shown above.

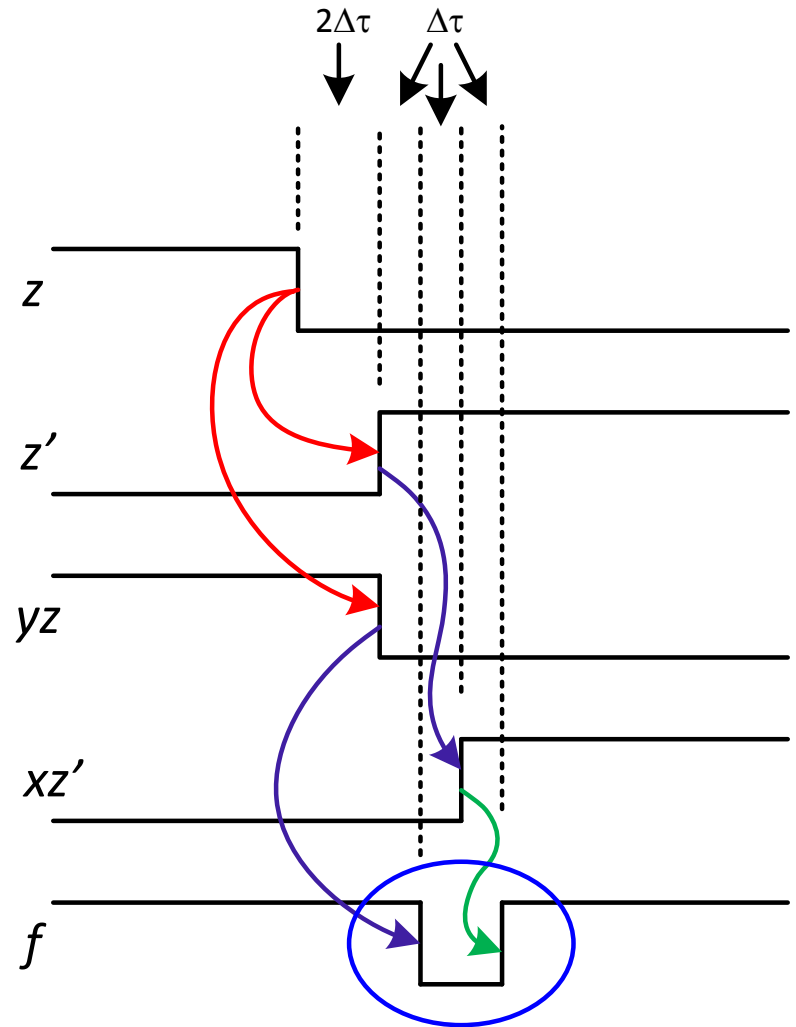
With the initial input condition:

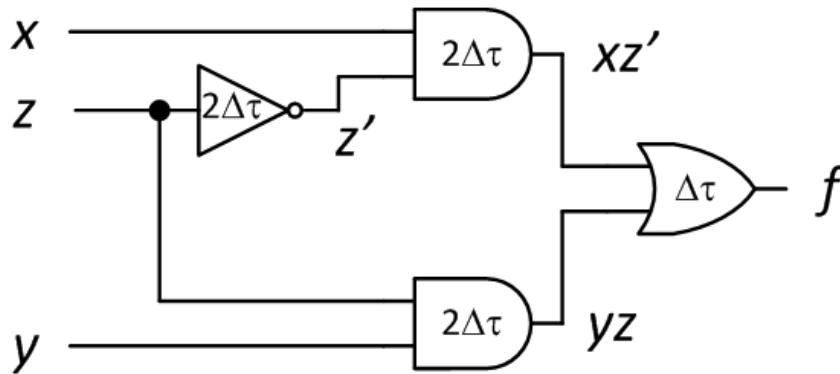
$$xy\textcolor{blue}{z} = 111$$

Now, change $\textcolor{blue}{z}$ from "1" to "0".

Then the associated gate outputs will be change after their corresponding delays.

An unwanted glitch will appear at the output – **static-1 hazard !!!**





Consider a different case

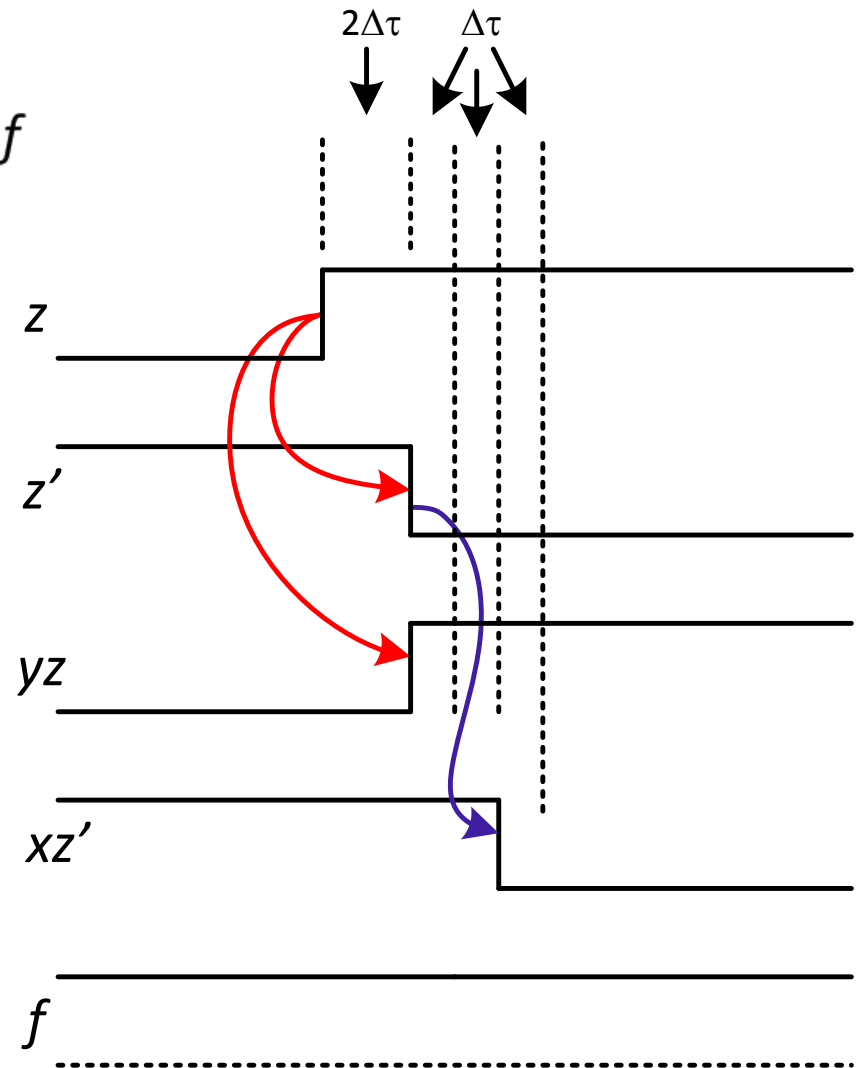
With the initial input condition:

$$xyz = 110$$

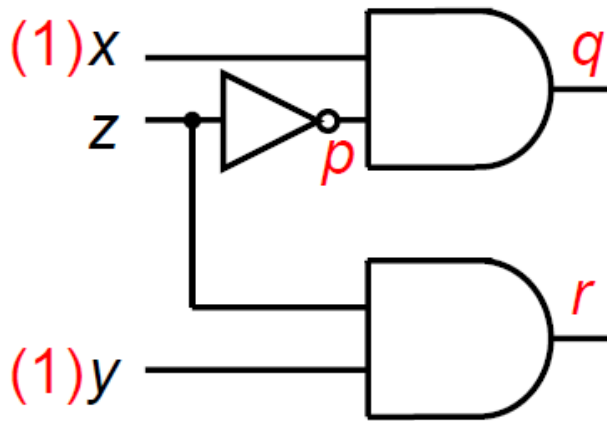
Now, change z from "0" to "1".

NO glitch !!!

Hazard only occurs when input of xyz change from 111 to 110, NOT for the case of from 110 to 111.



Example



First introduce 3 new variables p , q & r and assume the propagation delay for all logic gates are t_{PD} for easy analysis

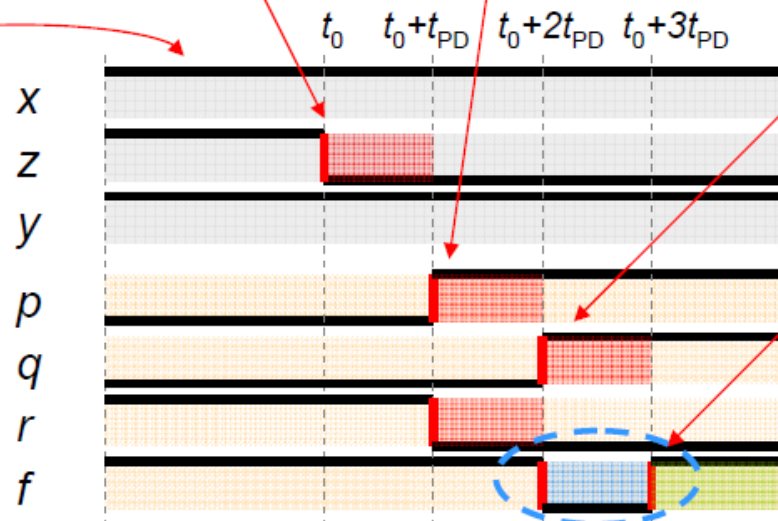
$$f(x, y, z) = xz' + yz$$

Now z changes from 1 to 0 at time t_0

But p changes from 0 to 1 after one t_{PD} time unit (r changes too)

Initial conditions:
(x, y, z) = (1, 1, 1)

Corresponding
output: (p, q, r, f)
= (0, 0, 1, 1)



Since p & r has changed, q & f change accordingly after another t_{PD} time unit

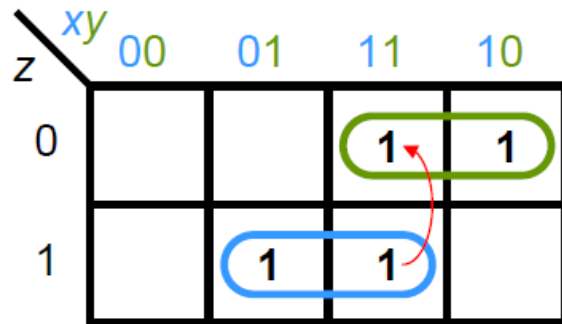
As q has changed, f changes again after another t_{PD} time unit. A static-1 hazard has formed

Timing hazard elimination

- The occurrence of the hazard can be detected by inspecting the K-map of the particular circuit.
- Eliminating a hazard is to enclose the two minterms in question with another product term that overlaps both groupings.
- The removal of hazards requires the addition of redundant gates to the circuit.

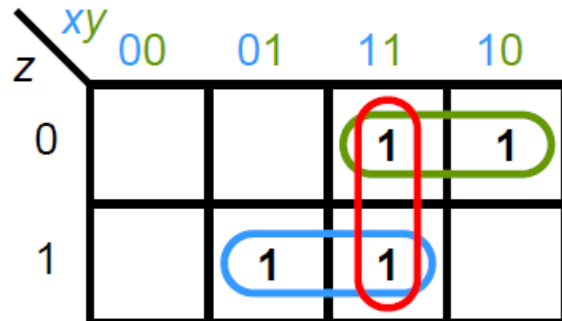
K-map for eliminating timing hazard

■ $f(x, y, z) = xz' + yz$



The optimal grouping

Static-1 hazard occurs when
change from 111 to 110



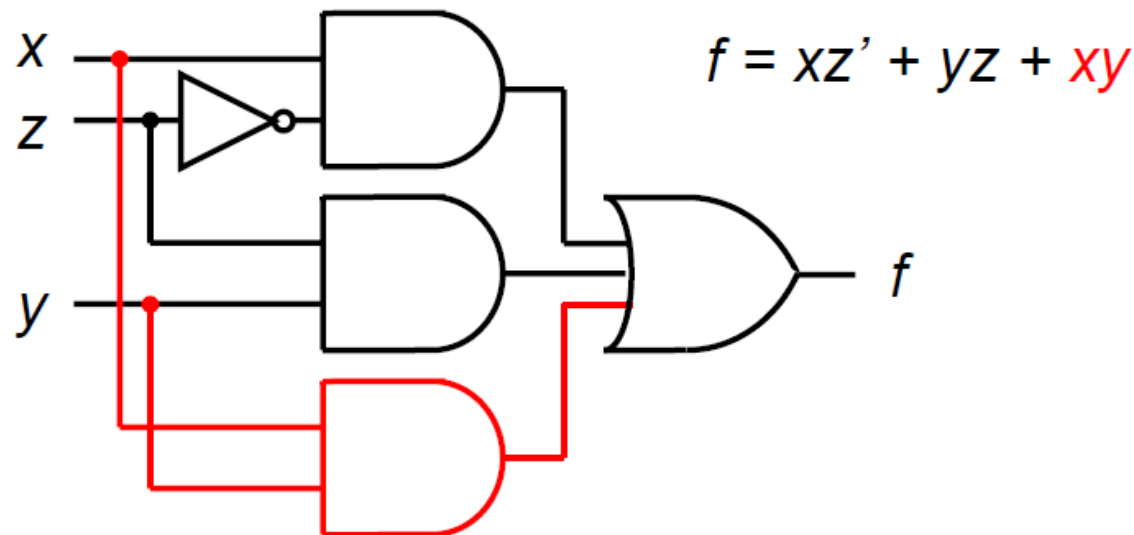
Include an redundant product term

$$f = xz' + yz + xy$$

Now the hazard is removed!

Use K-map to Eliminate Hazards

- To eliminating the hazard, group the two minterms that cause the hazards
- An additional redundant product term (gate) is introduced



Hazard-free realization

- General idea: to include all PIs
- Work for both K-map and QM-method

Please try to develop a hazard-free circuit for the function

$$f(a, b, c) = \sum m(0, 2, 4, 5)$$

Steps:

- a) Minimize the function f
- b) Realize f to a hazard-free circuit