XMUT 202
Digital Electronics

A/Prof Pawel Dmochowski

School of Engineering and Computer Science
Victoria University of Wellington

Victoria
UNIVERSITY OF WELLINGTON
Te Whare Wānanga o te Ūpoko o te Ika a Māui
CAPITAL CITY UNIVERSITY

Week 11 Lecture 1

- Combinatorial Logic (cont'd)
  - Review: K-maps
  - Two new gates: XNOR and XOR
  - Parity

Complete K-Map simplification process

1. Construct the K map, place 1s as per the truth table.
2. Loop 1s that are not adjacent to any other 1s.
3. Loop 1s that are in pairs *and can't be looped into quads or octets.*
4. Loop 1s in octets (8) even if they have already been looped.
5. Loop quads (4) that have one or more 1s not already looped.
6. Loop any pairs (2) necessary to include 1s not already looped.
7. Form the OR sum of terms generated by each loop.

Example (a) K-Map simplification

Simplify the following Boolean expression:

$$\bar{A}\bar{B}C\bar{D} + \bar{A}B\bar{C}D + \bar{A}BCD + AB\bar{C}D + ABCD + A\bar{B}CD$$

	$\bar{C}\bar{D}$	$\bar{C}D$	CD	$C\bar{D}$
$\bar{A}\bar{B}$				
$\bar{A}B$				
AB				
$A\bar{B}$				

1. Construct the K map, place 1s as per the truth table.
2. Loop 1s that are not adjacent to any other 1s.
3. Loop 1s that are in pairs *and cannot be looped into quads or octets.*
4. Loop 1s in octets (8) *even if they have already been looped.*
5. Loop quads (4) that have one or more 1s not already looped.
6. Loop any pairs (2) necessary to *include 1s not already looped.*
7. Form the OR sum of terms generated by each loop.

Example (a) K-Map simplification

Simplify the following Boolean expression:

$$\bar{A}\bar{B}C\bar{D} + \bar{A}B\bar{C}D + \bar{A}BCD + AB\bar{C}D + ABCD + A\bar{B}CD$$

	$\bar{C}\bar{D}$	$\bar{C}D$	CD	$C\bar{D}$
$\bar{A}\bar{B}$				1
$\bar{A}B$		1	1	
AB		1	1	
$A\bar{B}$			1	

1. Construct the K map, place 1s as per the truth table.
2. Loop 1s that are not adjacent to any other 1s.
3. Loop 1s that are in pairs *and cannot be looped into quads or octets.*
4. Loop 1s in octets (8) *even if they have already been looped.*
5. Loop quads (4) that have one or more 1s not already looped.
6. Loop any pairs (2) necessary to *include 1s not already looped.*
7. Form the OR sum of terms generated by each loop.

Example (a) K-Map simplification

Simplify the following Boolean expression:

$$\bar{A}\bar{B}C\bar{D} + \bar{A}B\bar{C}D + \bar{A}BCD + AB\bar{C}D + ABCD + A\bar{B}CD$$

	$\bar{C}\bar{D}$	$\bar{C}D$	CD	$C\bar{D}$
$\bar{A}\bar{B}$				1
$\bar{A}B$		1	1	
AB		1	1	
$A\bar{B}$			1	

1. Construct the K map, place 1s as per the truth table.
2. Loop 1s that are not adjacent to any other 1s.
3. Loop 1s that are in pairs *and cannot be looped into quads or octets.*
4. Loop 1s in octets (8) *even if they have already been looped.*
5. Loop quads (4) that have one or more 1s not already looped.
6. Loop any pairs (2) necessary to *include 1s not already looped.*
7. Form the OR sum of terms generated by each loop.

Example (a) K-Map simplification

Simplify the following Boolean expression:

$$\bar{A}\bar{B}C\bar{D} + \bar{A}B\bar{C}D + \bar{A}BCD + AB\bar{C}D + ABCD + A\bar{B}CD$$

	$\bar{C}\bar{D}$	$\bar{C}D$	CD	$C\bar{D}$
$\bar{A}\bar{B}$				1
$\bar{A}B$		1	1	
AB		1	1	
$A\bar{B}$			1	

1. Construct the K map, place 1s as per the truth table.
2. Loop 1s that are not adjacent to any other 1s.
3. Loop 1s that are in pairs *and cannot be looped into quads or octets.*
4. Loop 1s in octets (8) *even if they have already been looped.*
5. Loop quads (4) that have one or more 1s not already looped.
6. Loop any pairs (2) necessary to *include 1s not already looped.*
7. Form the OR sum of terms generated by each loop.

Example (a) K-Map simplification

Simplify the following Boolean expression:

$$\bar{A}\bar{B}C\bar{D} + \bar{A}B\bar{C}D + \bar{A}BCD + AB\bar{C}D + ABCD + A\bar{B}CD$$

	$\bar{C}\bar{D}$	$\bar{C}D$	CD	$C\bar{D}$
$\bar{A}\bar{B}$				1
$\bar{A}B$		1	1	
AB		1	1	
$A\bar{B}$			1	

1. Construct the K map, place 1s as per the truth table.
2. Loop 1s that are not adjacent to any other 1s.
3. Loop 1s that are in pairs *and cannot be looped into quads or octets.*
4. Loop 1s in octets (8) *even if they have already been looped. (none here)*
5. Loop quads (4) that have one or more 1s not already looped.
6. Loop any pairs (2) necessary to *include 1s not already looped.*
7. Form the OR sum of terms generated by each loop.

Example (a) K-Map simplification

Simplify the following Boolean expression:

$$\bar{A}\bar{B}C\bar{D} + \bar{A}B\bar{C}D + \bar{A}BCD + AB\bar{C}D + ABCD + A\bar{B}CD$$

	$\bar{C}\bar{D}$	$\bar{C}D$	CD	$C\bar{D}$
$\bar{A}\bar{B}$				1
$\bar{A}B$		1	1	
AB		1	1	
$A\bar{B}$			1	

1. Construct the K map, place 1s as per the truth table.
2. Loop 1s that are not adjacent to any other 1s.
3. Loop 1s that are in pairs *and cannot be looped into quads or octets.*
4. Loop 1s in octets (8) *even if they have already been looped.*
5. Loop quads (4) that have one or more 1s not already looped.
6. Loop any pairs (2) necessary to *include 1s not already looped.*
7. Form the OR sum of terms generated by each loop.

Example (a) K-Map simplification

Simplify the following Boolean expression:

$\bar{A}\bar{B}C\bar{D} + \bar{A}B\bar{C}D + \bar{A}BCD + AB\bar{C}D + ABCD + A\bar{B}CD$

	$\bar{C}\bar{D}$	$\bar{C}D$	CD	$C\bar{D}$
$\bar{A}\bar{B}$				1
$\bar{A}B$		1	1	
AB		1	1	
$A\bar{B}$			1	

1. Construct the K map, place 1s as per the truth table.
2. Loop 1s that are not adjacent to any other 1s.
3. Loop 1s that are in pairs *and cannot be looped into quads or octets.*
4. Loop 1s in octets (8) *even if they have already been looped.*
5. Loop quads (4) that have one or more 1s not already looped.
6. Loop any pairs (2) necessary to *include 1s not already looped. (none here)*
7. Form the OR sum of terms generated by each loop.

Example (a) K-Map simplification

Simplify the following Boolean expression:

$$\bar{A}\bar{B}C\bar{D} + \bar{A}B\bar{C}D + \bar{A}BCD + AB\bar{C}D + ABCD + A\bar{B}CD$$

	$\bar{C}\bar{D}$	$\bar{C}D$	CD	$C\bar{D}$
$\bar{A}\bar{B}$				1
$\bar{A}B$		1	1	
AB		1	1	
$A\bar{B}$			1	

1. Construct the K map, place 1s as per the truth table.
2. Loop 1s that are not adjacent to any other 1s.
3. Loop 1s that are in pairs *and cannot be looped into quads or octets.*
4. Loop 1s in octets (8) *even if they have already been looped.*
5. Loop quads (4) that have one or more 1s not already looped.
6. Loop any pairs (2) necessary to *include 1s not already looped.*
7. Form the OR sum of terms generated by each loop.

$$BD + ACD + \bar{A}\bar{B}C\bar{D}$$

Don’t Care Output Conditions

Can be changed 0/1 so that the simplest expression can be obtained from the K-map. Typically occurs when we know certain input conditions are impossible.

Don’t care Output Conditions

Can be changed 0/1 so that the simplest expression can be obtained from the K-map. Typically occur when we know certain input conditions are impossible.

A	B	C	z	
0	0	0	0	
0	0	1	0	
0	1	0	0	
0	1	1	x	"don't
1	0	0	x	care"
1	0	1	1	
1	1	0	1	
1	1	1	1	

(a)

Don’t care Output Conditions

Can be changed 0/1 so that the simplest expression can be obtained from the K-map. Typically occur when we know certain input conditions are impossible.

A	B	C	z	
0	0	0	0	
0	0	1	0	
0	1	0	0	
0	1	1	x	"don't
1	0	0	x	care"
1	0	1	1	
1	1	0	1	
1	1	1	1	

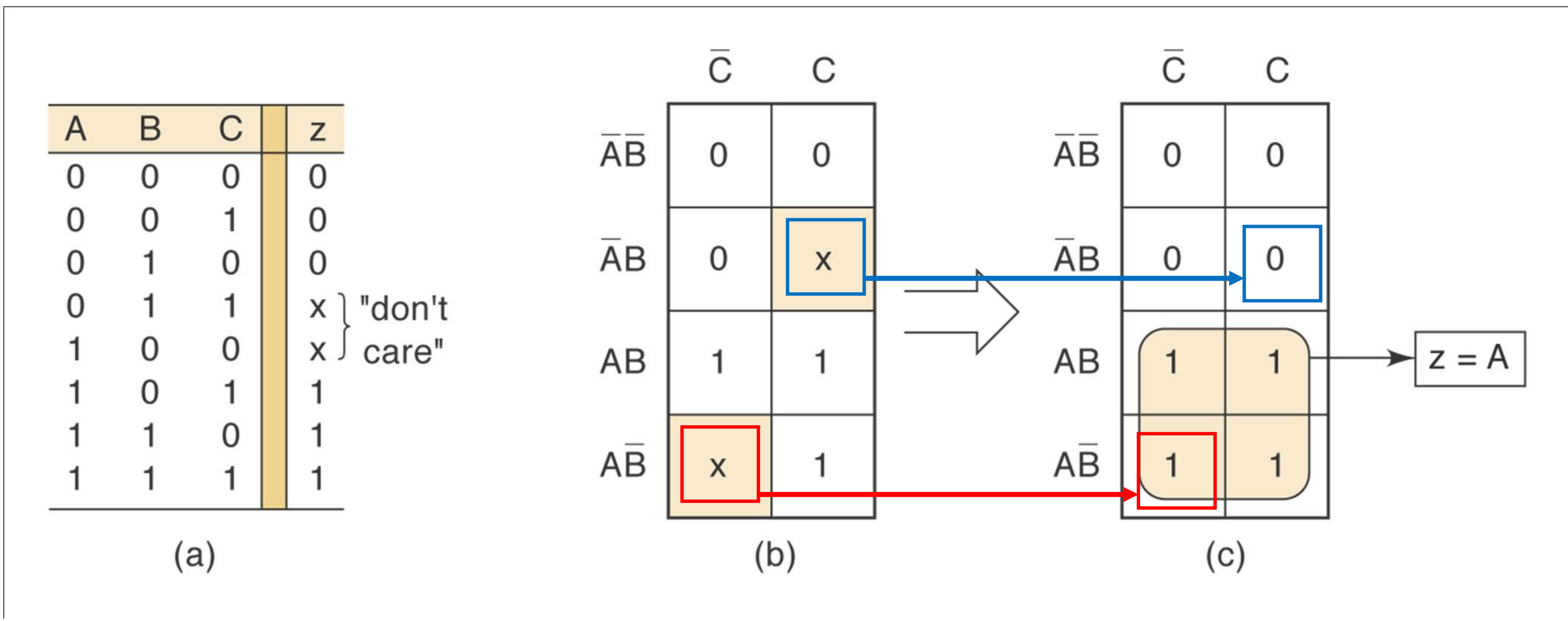
(a)

	$\bar{C}$	C
$\bar{A}\bar{B}$	0	0
$\bar{A}B$	0	x
AB	1	1
$A\bar{B}$	x	1

(b)

Don’t care Output Conditions

Can be changed 0/1 so that the simplest expression can be obtained from the K-map. Typically occur when we know certain input conditions are impossible.

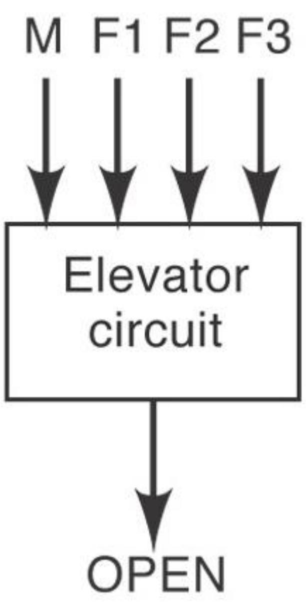


Example: Design a logic circuit for a three-storey elevator.

M = Logic signal indicating if the elevator is moving (M = 1) or stationary (M = 0)

F1, F2 and F3 are the floor level signals, normally LO but go HI when a particular floor is reached.

The circuit output (O/P) is the "Door Open" signal, should be normally LO but go HI when the door is to open



M	F1	F2	F3		OPEN
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		

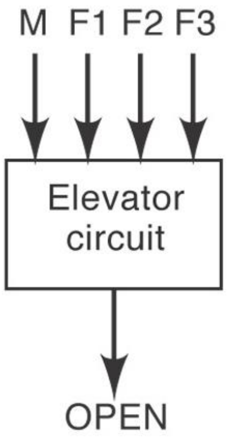
M = elevator moving

F1 = Floor 1

F2 – Floor 2

F3 – Floor 3

OPEN – elevator door opening



M	F1	F2	F3		OPEN
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		

- Can only be on one floor at a time (only one floor I/P can be HI) .
- The other floor I/P's are then don't care conditions.
- Use x to indicate the don't care conditions.
- Door can't open when moving!

M	F1	F2	F3	OPEN
0	0	0	0	0
0	0	0	1	1
0	0	1	0	1
0	0	1	1	X
0	1	0	0	1
0	1	0	1	X
0	1	1	0	X
0	1	1	1	X
1	0	0	0	0
1	0	0	1	0
1	0	1	0	0
1	0	1	1	X
1	1	0	0	0
1	1	0	1	X
1	1	1	0	X
1	1	1	1	X

- Can only be on one floor at a time (only one floor I/P can be HI) .
- The other floor I/P's are then don't care conditions.
- Use x to indicate the don't care conditions.
- Door can't open when moving!

M	F1	F2	F3	OPEN
0	0	0	0	0
0	0	0	1	1
0	0	1	0	1
0	0	1	1	X
0	1	0	0	1
0	1	0	1	X
0	1	1	0	X
0	1	1	1	X
1	0	0	0	0
1	0	0	1	0
1	0	1	0	0
1	0	1	1	X
1	1	0	0	0
1	1	0	1	X
1	1	1	0	X
1	1	1	1	X

- Can only be on one floor at a time (only one floor I/P can be HI) .
- The other floor I/P’s are then don’t care conditions.
- Use x to indicate the don’t care conditions.
- Door can’t open when moving!

$$\text{OPEN} = \overline{M}\,\overline{F1}\,\overline{F2}\,F3 + \overline{M}\,F1\,\overline{F2}\,\overline{F3} + \overline{M}\,\overline{F1}\,F2\,\overline{F3}$$

	$\overline{F2}\,\overline{F3}$	$\overline{F2}\,F3$	$F2\,F3$	$F2\,\overline{F3}$
$\overline{M}\ \overline{F1}$	0	1	X	1
$\overline{M}\ F1$	1	X	X	X
$M\ F1$	0	X	X	X
$M\ \overline{F1}$	0	0	X	0

	$\overline{F2}\,\overline{F3}$	$\overline{F2}\,F3$	$F2\,F3$	$F2\,\overline{F3}$
$\overline{M}\,\overline{F1}$	0	1	X	1
$\overline{M}\,F1$	1	X	X	X
$M\,F1$	0	X	X	X
$M\,\overline{F1}$	0	0	X	0

	$\overline{F2}\,\overline{F3}$	$\overline{F2}\,F3$	$F2\,F3$	$F2\,\overline{F3}$
$\overline{M}\,\overline{F1}$	0	1	1	1
$\overline{M}\,F1$	1	1	1	1
$M\,F1$	0	0	0	0
$M\,\overline{F1}$	0	0	0	0

$$\boxed{OPEN = \overline{M}\,(F1 + F2 + F3)}$$

Test yourself:

Use a K-map to simplify:

$$y = \overline{C}(\overline{A}\,\overline{B}\,\overline{D} + D) + A\overline{B}C + \overline{D}$$

$$y = \overline{A}\,\overline{B}\,\overline{C}\,\overline{D} + \overline{C}D + A\overline{B}C + \overline{D}$$

	$\overline{C}\,\overline{D}$	$\overline{C}D$	CD	$C\overline{D}$
$\overline{A}\,\overline{B}$	1	1	0	1
$\overline{A}B$	1	1	0	1
AB	1	1	0	1
$A\overline{B}$	1	1	1	1

$$y = \bar{A}\bar{B}\bar{C}\bar{D} + \bar{C}D + A\bar{B}C + \bar{D}$$

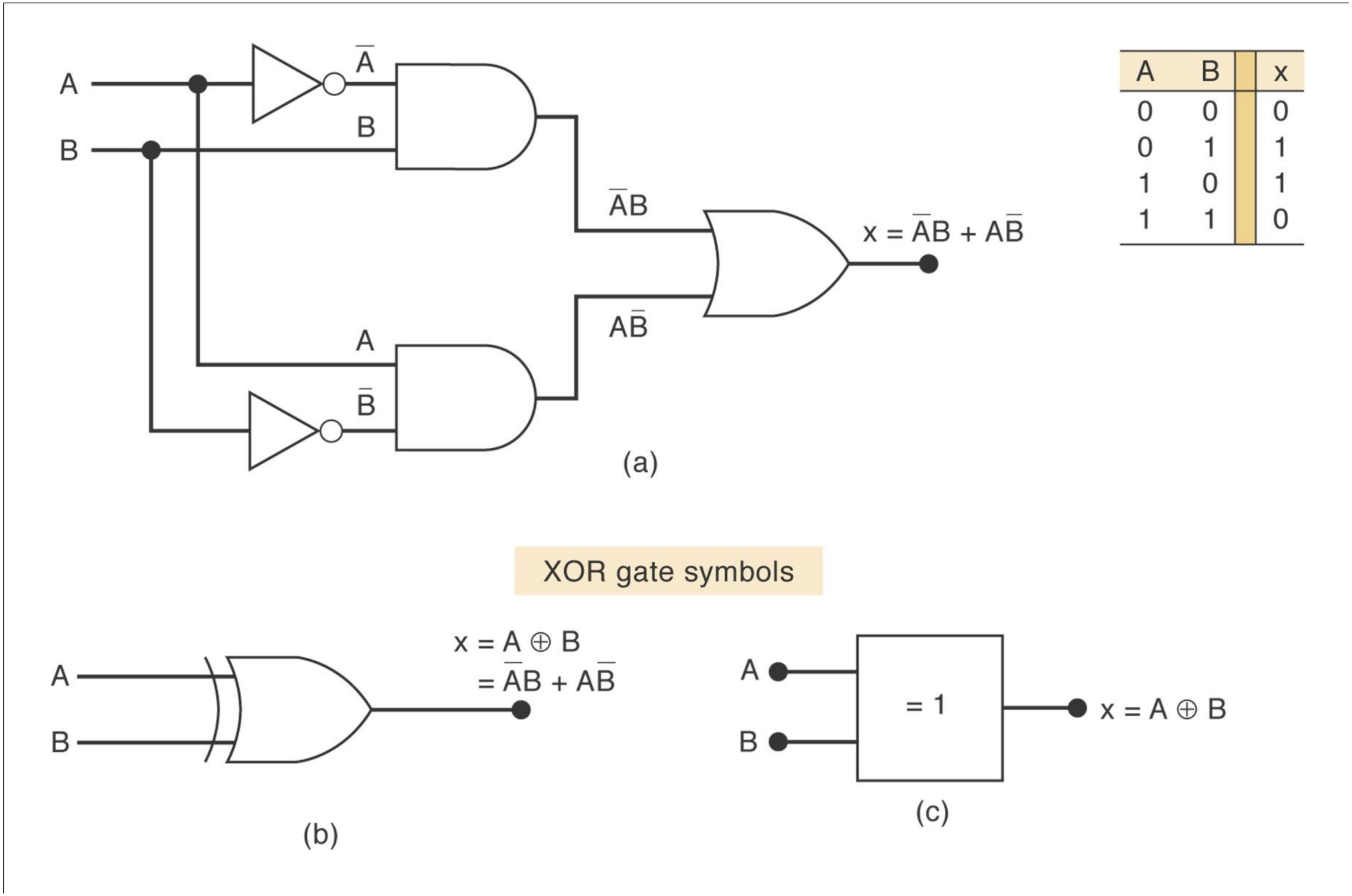
	$\bar{C}\bar{D}$	$\bar{C}D$	CD	$C\bar{D}$
$\bar{A}\bar{B}$	1	1	0	1
$\bar{A}B$	1	1	0	1
AB	1	1	0	1
$A\bar{B}$	1	1	1	1

$$y = A\bar{B} + \bar{C} + \bar{D}$$

Exclusive OR and Exclusive NOR Circuits

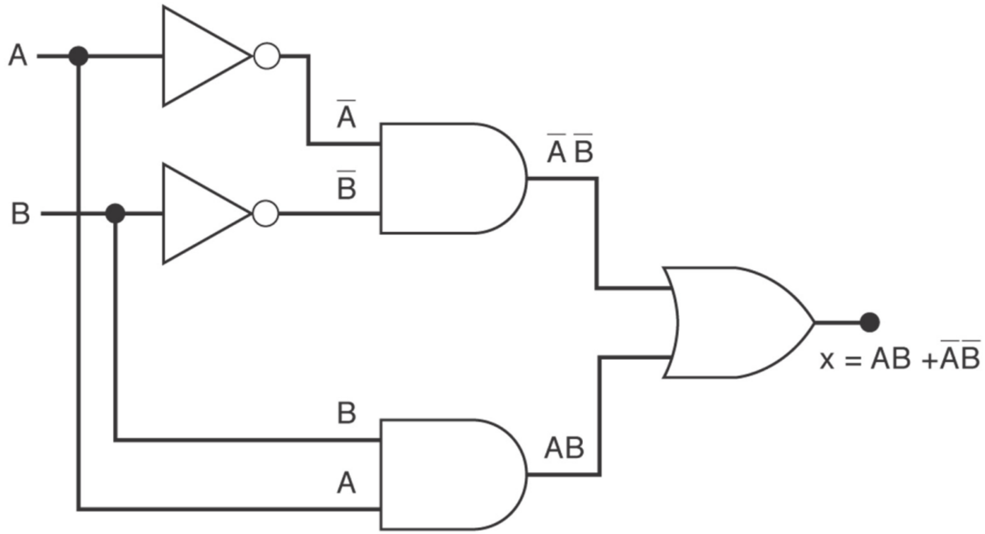
- The exclusive OR, abbreviated XOR produces a HIGH output whenever the two inputs are at opposite levels.
- The exclusive NOR, abbreviated XNOR produces a HIGH output whenever the two inputs are at the same level.
- XOR and XNOR outputs are opposite.

FIGURE 4-20 (a) Exclusive-OR circuit and truth table; (b) traditional XOR gate symbol; (c) IEEE/ANSI symbol for XOR gate.



A	B	x
0	0	0
0	1	1
1	0	1
1	1	0

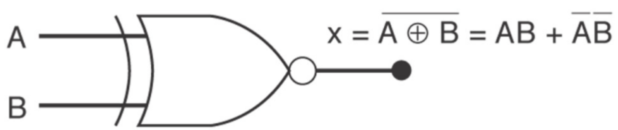
FIGURE 4-21 (a) Exclusive-NOR circuit; (b) traditional symbol for XNOR gate; (c) IEEE/ANSI symbol.



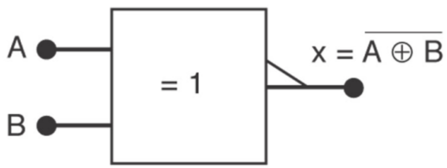
A	B	x
0	0	1
0	1	0
1	0	0
1	1	1

(a)

XNOR gate symbols

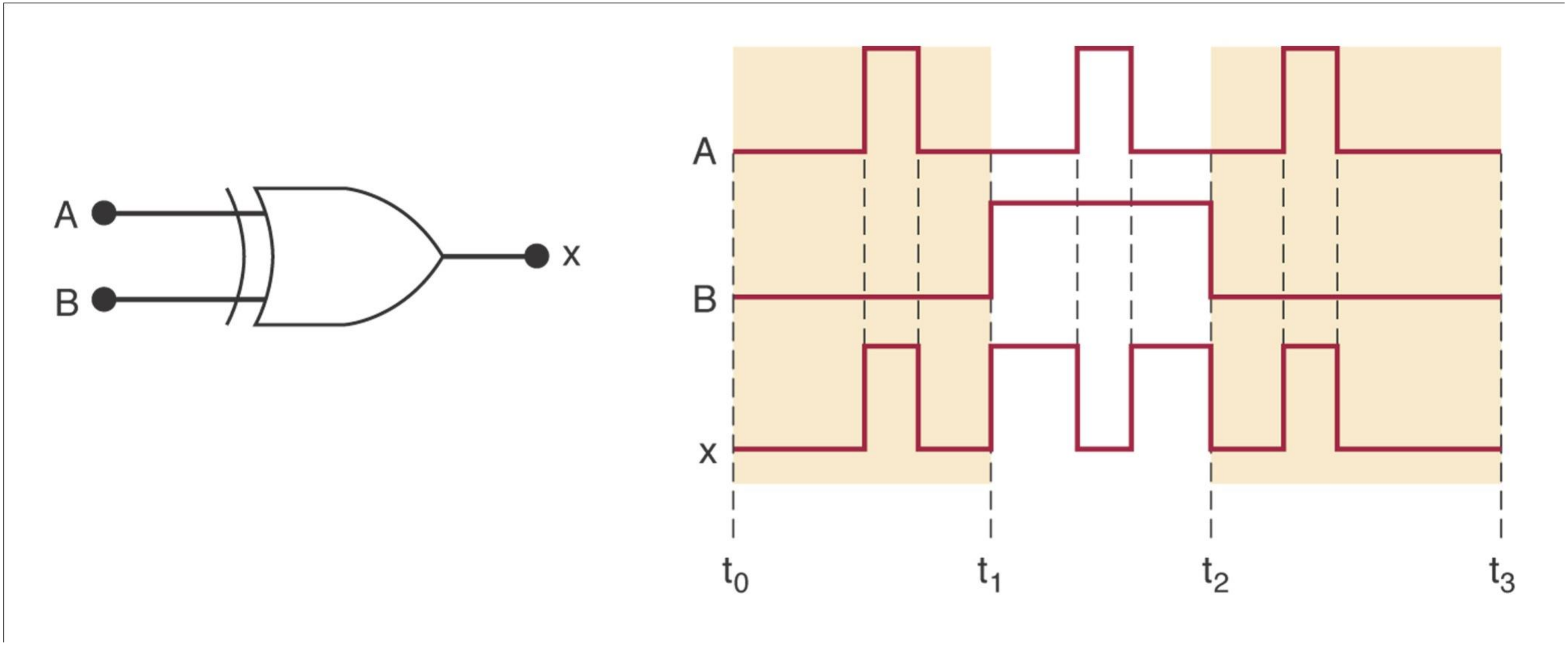


(b)



(c)

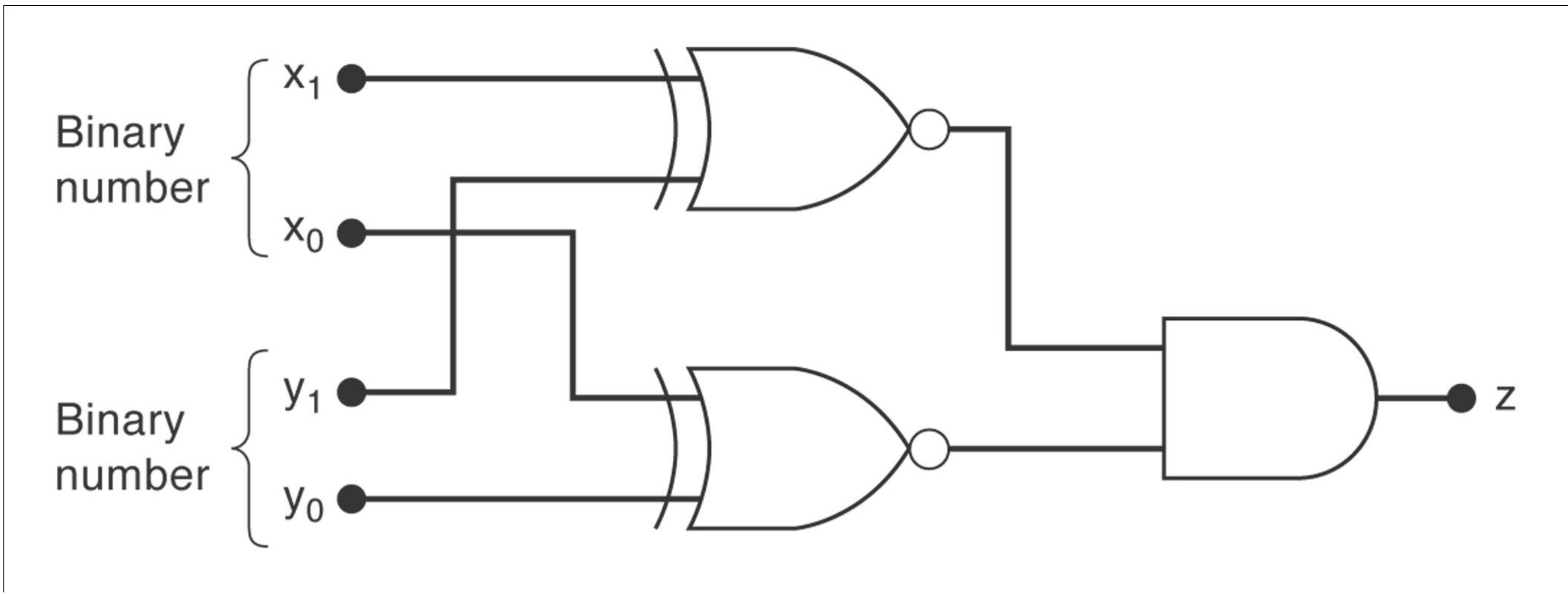
Determine the O/P waveform of the circuit below:



O/P Hi when I/P at different levels

Design a circuit so that the O/P will only be HI when the combination of two sets of two bit binary numbers are equal.

x_1	x_0	y_1	y_0	z (Output)
0	0	0	0	1
0	0	0	1	0
0	0	1	0	0
0	0	1	1	0
0	1	0	0	0
0	1	0	1	1
0	1	1	0	0
0	1	1	1	0
1	0	0	0	0
1	0	0	1	0
1	0	1	0	1
1	0	1	1	0
1	1	0	0	0
1	1	0	1	0
1	1	1	0	0
1	1	1	1	1


Binary
number
x_1
x_0
Binary
number
y_1
y_0
z

Parity Generator and Checker

- Parity bit: extra bit included with data to make the number of 1’s even (for even parity) or odd (for odd parity)
- It is used to detect error in transmission
- Example: if we use an even parity system:
    - Data: 1 1 1 0 – we add a parity bit 1
    - Data: 1 1 0 0 – we add parity bit 0
    - Data: 0 0 0 0 – we add a parity bit 0

- A Parity Checker will return a TRUE error bit if the number of 1’s is odd (for even parity) and if the number of 1’s is even (for odd parity)

Parity Generator

$$AB\oplus AB = \bar{A}B + A\bar{B} \qquad \overline{AB\oplus AB} = \bar{A}\bar{B} + AB$$

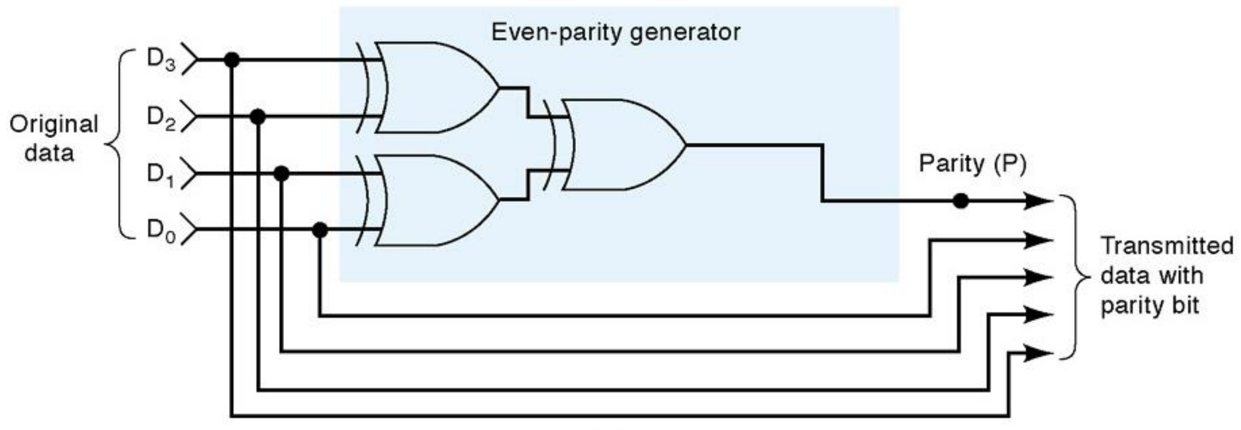
- How to construct an even parity generator (3 bits input)?
- Truth table:

A	B	C	P
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

$$\begin{aligned}
P &= \bar{A}\bar{B}C + \bar{A}B\bar{C} + A\bar{B}\bar{C} + ABC \\
&= \bar{A}(\bar{B}C + B\bar{C}) + A(\bar{B}\bar{C} + BC) \\
&= \bar{A}(BC\oplus BC) + A(\overline{BC\oplus BC}) \\
&= \bar{A}X + A\bar{X} \\
&= A\oplus B\oplus C
\end{aligned}$$

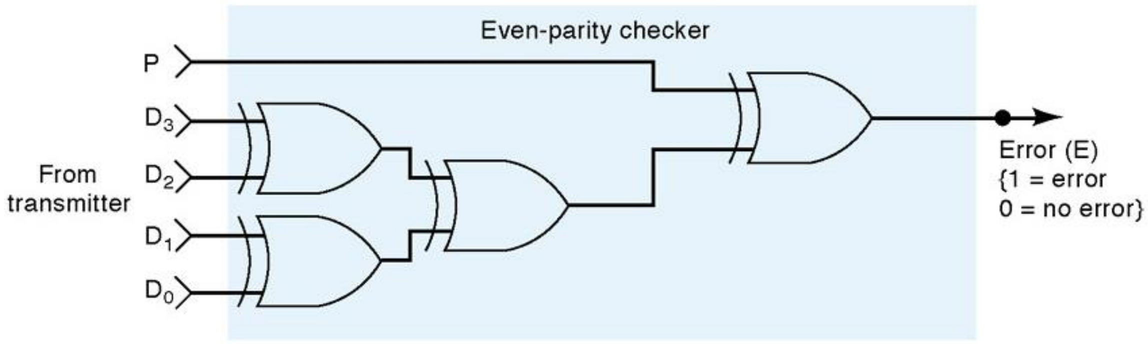
Parity Generator and Checker

Similarly for 4 bits:



(a)

Even Parity:



(b)

Parity Checker

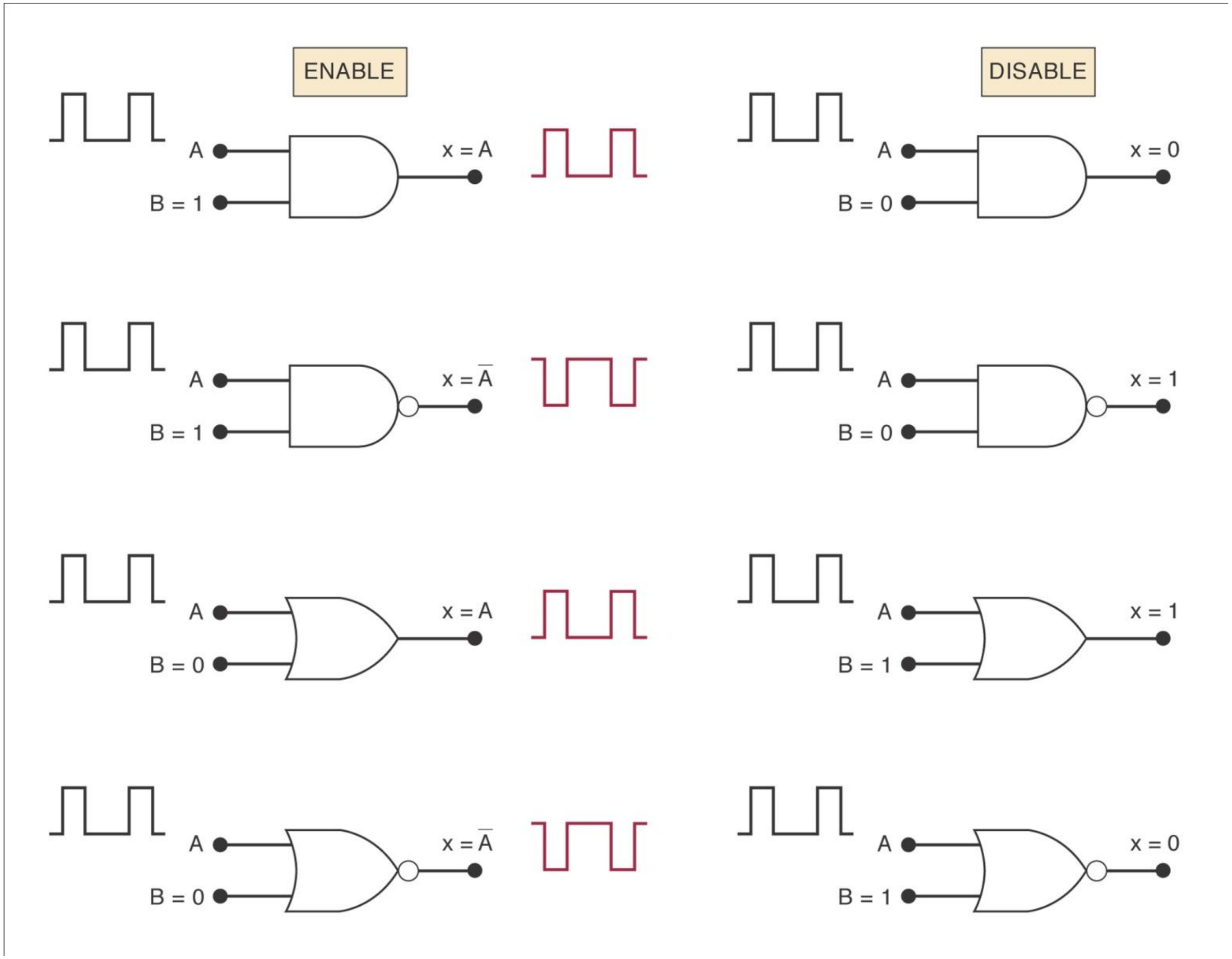
- Homework exercise:
  - Design an even parity checker (3 data bits) using a truth table
  - Express it using XOR or XNOR gates

A	B	C	P	E

Enable/Disable Circuits

- A circuit is enabled when it allows the passage of an input signal to the output.
- A circuit is disabled when it prevents the passage of an input signal to the output.
- Situations requiring enable/disable circuits occur frequently in digital circuit design.

FIGURE 4-26 Four basic gates can either enable or disable the passage of an input signal, *A*, under control of the logic level at control input *B*.



Week 11 Lecture 1

- K-Map method review
- XNOR and XOR gates
- Parity