

# Digital Electronics

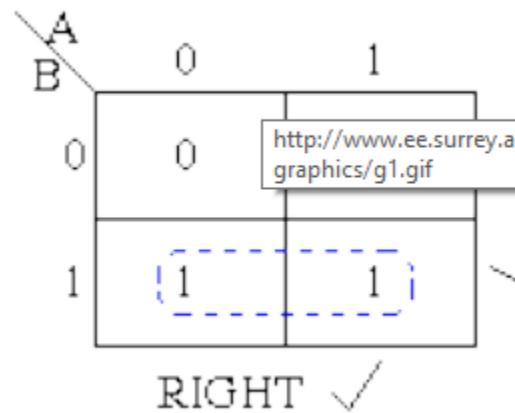
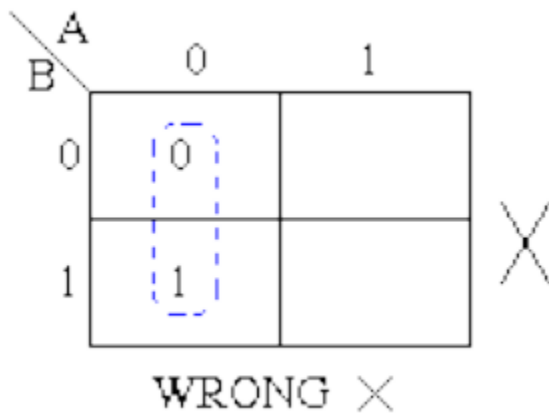
BY

DR FANUEL KEHEZE

# Karnaugh Map Rules

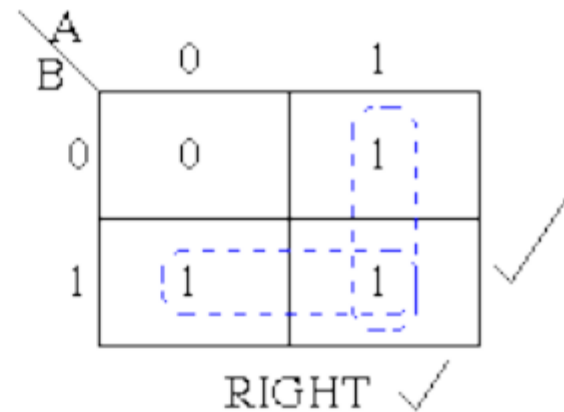
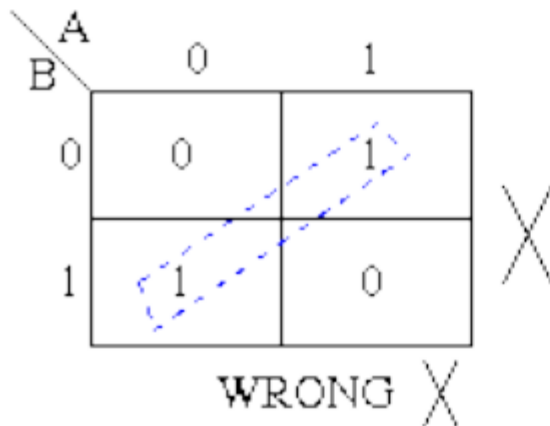
The Karnaugh map uses the following rules for the simplification of expressions by ***grouping together adjacent cells containing ones***

❖ Groups may not include any cell containing a zero



# Karnaugh Map Rules

❖ Groups may be horizontal or vertical, but not diagonal.



# Karnaugh Map Rules

- ❖ Groups **MUST** contain, 1,2,4,8 or  $2^n$  number of cells
- ❖ Each group should be as large as possible.

$\backslash AB$	00	01	11	10
C				
0	1	1	1	1
1	0	0	1	1

RIGHT ✓

$\backslash AB$	00	01	11	10
C				
0	1	1	1	1
1	0	0	1	1

WRONG ✗

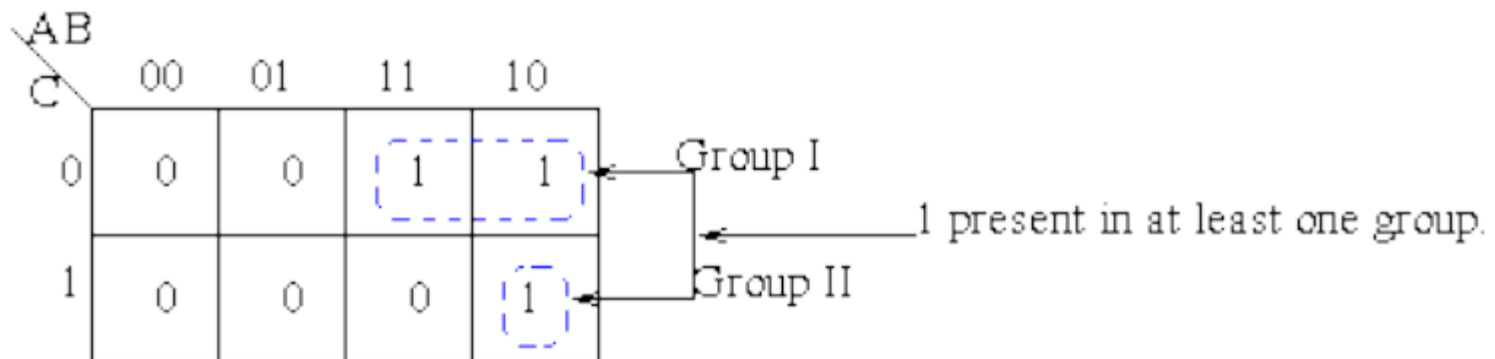
<http://www.ee.surrey.ac.uk/Projectgraphics/g4.gif>

(Note that no Boolean laws broken, but not sufficiently minimal)

- ❖ Each cell containing a *one* must be in at least one group.

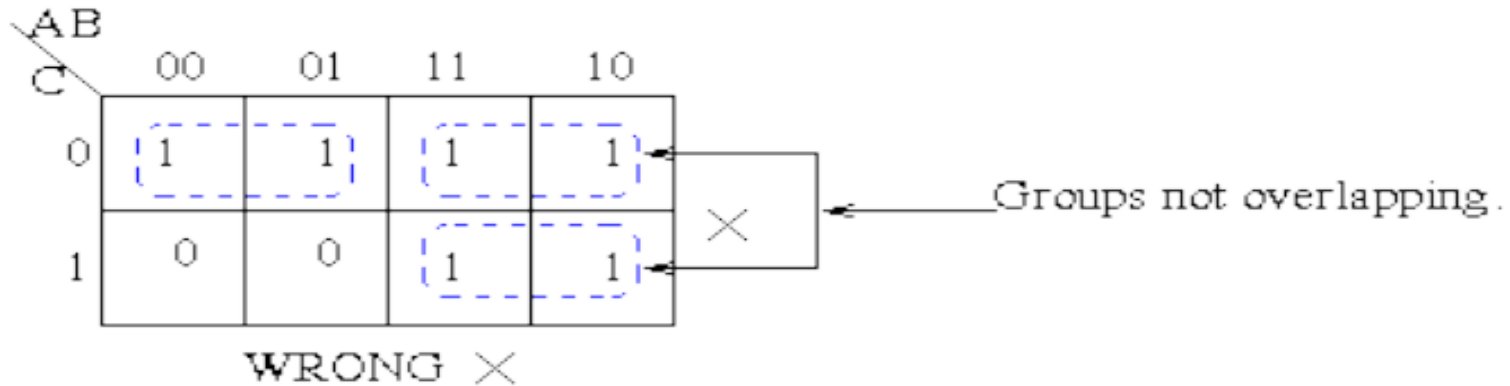
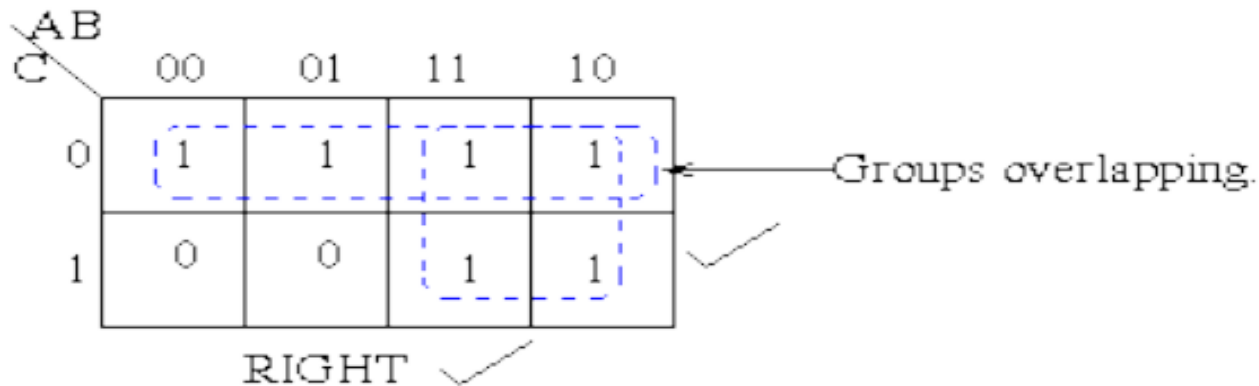
# Karnaugh Map Rules

- ❖ Each cell containing a *one* must be in at least one group.



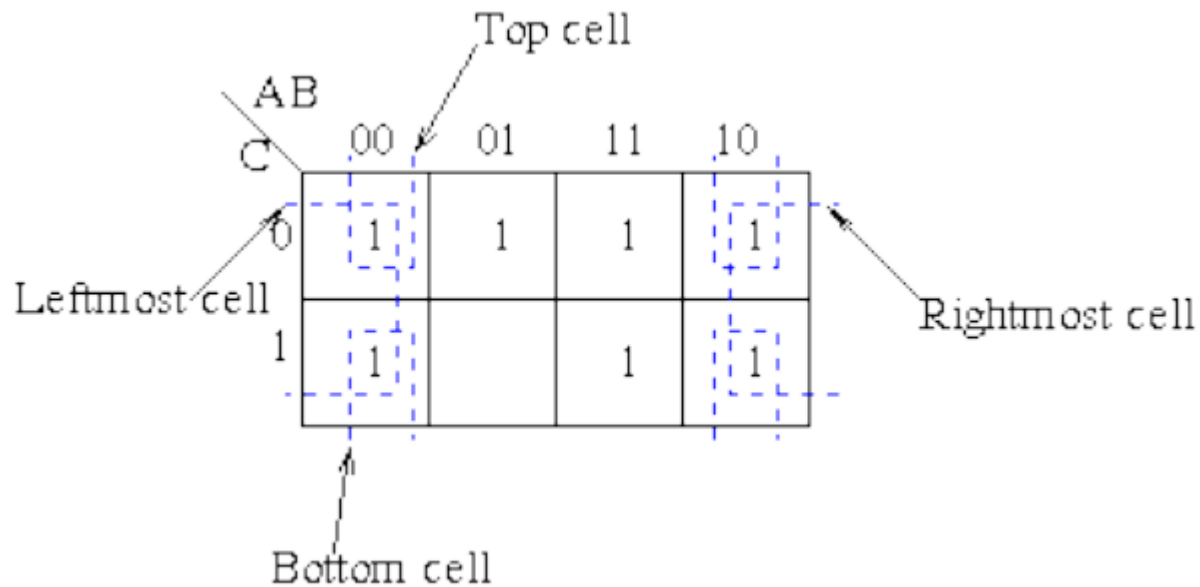
# Karnaugh Map Rules

❖ Groups may overlap.



# Karnaugh Map Rules

- ❖ Groups may wrap around the table. The leftmost cell in a row may be grouped with the rightmost cell and the top cell in a column may be grouped with the bottom cell.



# Karnaugh Map Rules

- ❖ There should be as few groups as possible, as long as this does not contradict any of the previous rules

$\diagdown$ AB	00	01	11	10
C				
0	1	1	1	1
1	0	0	1	1

RIGHT ✓

$\diagdown$ AB	00	01	11	10
C				
0	1	1	1	1
1	0	0	1	1

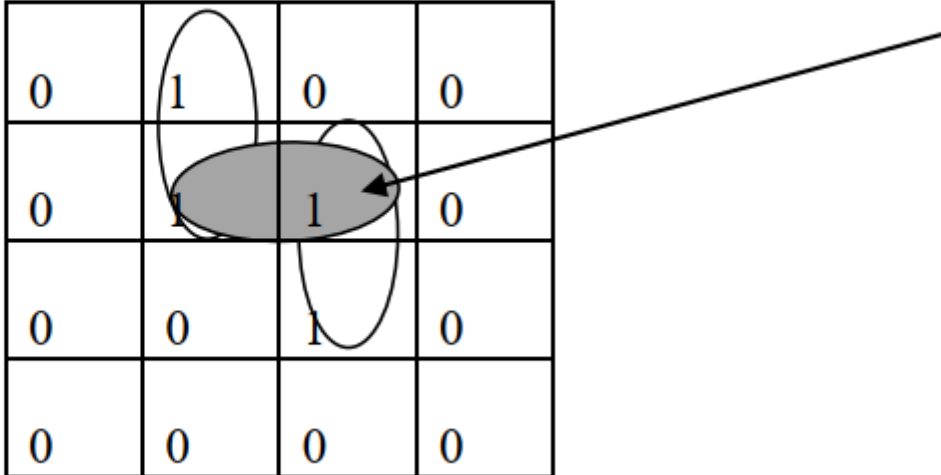
WRONG ✗



# Redundant group

- ❖ After encircling groups before coming up with the simplified Boolean expression eliminate any group whose 1's are completely overlapped by other groups. Such a group is known as redundant group.

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



Redundant group

# Don't care condition

- ❖ After encircling groups before coming up with the simplified Boolean expression eliminate any group whose 1's are completely overlapped by other groups. Such a group is known as redundant group. At times it does not matter what the output is for a given input word, to indicate this condition an x is used in the truth table instead of a zero or one. This x is known as don't care because they can either be zeros or ones. During simplification of the logic expression, the x's are circled with ones
- In certain cases some of the minterms may never occur or it may not matter what happens if they do. In such cases we fill in the Karnaugh map with and X meaning don't care
  - When minimizing an X is like a "joker"
  - X can be 0 or 1 - whatever helps best with the minimization
  - eg

# Don't care condition

Example 1

---

$A \backslash BC$	00	01	11	10
0	0	0	1	X
1	0	0	1	1

– simplifies to  $B$  if  $X$  is assumed 1

# Don't care condition

## Example 2

Truth table

$A$	$B$	$C$	$Z$
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	X
1	0	0	X
1	0	1	1
1	1	0	1
1	1	1	1

$A \setminus BC$	$\bar{A}\bar{B}$	$\bar{A}B$	$AB$	$A\bar{B}$
$\bar{C}$	0	1	X	
$C$	0	1	1	

↓

$A \setminus BC$	$\bar{A}\bar{B}$	$\bar{A}B$	$AB$	$A\bar{B}$
$\bar{C}$	0	0	1	1
$C$	0	0	1	1

$$Z=A$$

Note: a group involving the x must have a one.

# Don't care condition

## Review Questions

1. Simplify the following expressions using K-MAP
  - a.  $F = \bar{A}BC + A\bar{B}\bar{C} + ABC + AB\bar{C}$ .
  - b.  $F = \bar{A}\bar{B}C + \bar{A}BC + \bar{A}B\bar{C} + A\bar{B}C + ABC$ .
  - c.  $Z = f(A,B,C) = \bar{A}\bar{B}\bar{C} + \bar{A}B + AB\bar{C} + AC$
  - d.  $Z = f(A,B,C) = \bar{A}B + B\bar{C} + BC + A\bar{B}\bar{C}$
  - e.  $Z = f(A,B,C) = \bar{A}\bar{B}\bar{C} + \bar{A}B + AB\bar{C} + AC$
1. What are the don't-care conditions?
2. What are the advantages of the tabulation method?
3. Draw a Karnaugh map for a four-variable Ex-OR function and derive its expression.
4. How does a Karnaugh map differ from a truth table?
5. What kind of network is developed by sum of the products?
6. A combinational switching network has four inputs A, B, C, and D, and one output Z. The output is to be 0, if the input combination is a valid Excess-3 coded decimal digit. If any other combinations of inputs appear, the output is to be 1. Implement the network using basic gates

# Digital Circuits

- Digital circuits can be subdivided into two main categories;
  - ❑ Combination circuits
  - ❑ Sequential circuits
- **Combination circuits** are types of circuits where by the outputs depends on the inputs at that instance of time. This type of circuit has no memory devices.
- **Sequential circuits** have outputs which depend on inputs at that instance of time as whereas outputs from other circuits which serve as input to that circuit. These circuits therefore serve as inputs. This means that sequential circuits require memory elements.
- Both Sequential and Combination circuits make micro architecture of the standard micro processors as well as custom specific integrated circuits (ICs)

# Combinational Logic Circuit

- Combination Logic circuits perform data transforms for example logic operations and arithmetic operations (XOR, AND, complement operations etc). These circuits' devices include adders, multiplexers, de-multiplexors, parity encoders, comparators etc.
- **DESIGN PROCEDURE**

Any combinational circuit can be designed by the following steps of design procedure.

  1. The problem is stated.
  2. Identify the input variables and output functions.
  3. The input and output variables are assigned letter symbols.
  4. The truth table is prepared that completely defines the relationship between the input variables and output functions.
  5. The simplified Boolean expression is obtained by any method of minimization—algebraic method, Karnaugh map method, or tabulation method.
  6. A logic diagram is realized from the simplified expression using logic gates

# Combinational Logic Circuit

- A practical design approach should consider constraints like—
- i. minimum number of gates,
  - ii. minimum number of outputs,
  - iii. minimum propagation time of the signal through a circuit,
  - iv. minimum number of interconnections, and
  - v. limitations of the driving capabilities of each logic gate

## ADDERS

The simple addition consists of four possible elementary operations, which are  $0+0 = 0$ ,  $0+1 = 1$ ,  $1+0 = 1$ , and  $1+1 = 10$

The first three operations produce a sum of one digit, but the fourth operation produces a sum consisting of two digits

The higher significant bit of this result is called the carry. A combinational circuit that performs the addition of two bits as described above is called a **half-adder**.



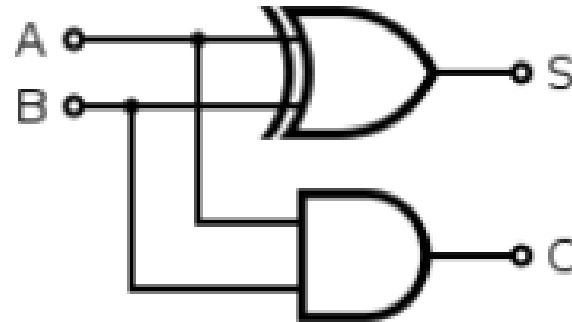
# Design of Half adder

- A **half adder** adds two one-bit binary numbers  $A$  and  $B$ . It has two outputs,  $S$  and  $C$  (the value theoretically carried on to the next addition); the final sum is  $2C + S$ . The simplest half-adder design, pictured on the right, incorporates an XOR gate for  $S$  and an AND gate for  $C$ . Half adders cannot be used compositely, given their incapacity for a carry-in bit.

## Truth Table

A	B	SUM(s)	CARRY(c)
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

$$S = A'B + AB'$$
$$C = AB$$



# Design of Full adder

## ➤ Full adder

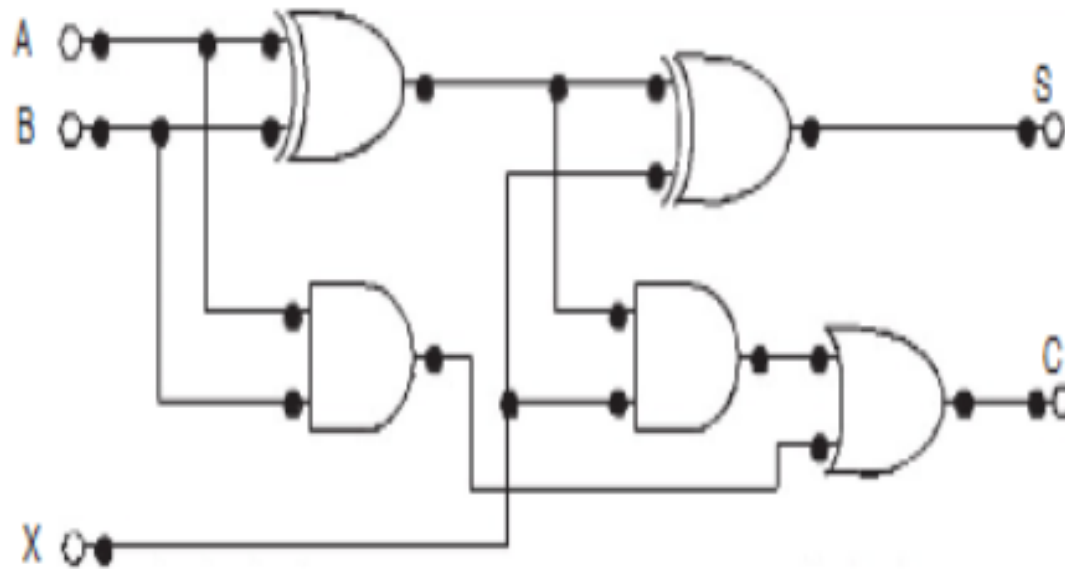
The half adder has two inputs but it has no provision to add a carry coming from the lower order bits i.e. when the multi-bit addition is being performed. A third input is required for the carry generated. This modified circuit is known as the full adder circuit. The full adder circuit can be designed using either the AND, OR and XOR or using NAND, and OR gates. A combinational circuit of full-adder performs the operation of addition of three bits—the augend, addend, and previous carry, and produces the outputs sum and carry

Inputs			Outputs	
<i>A</i>	<i>B</i>	<i>X</i>	<i>C</i>	<i>S</i>
0	0	0	0	0
1	0	0	0	1
0	1	0	0	1
1	1	0	1	0
0	0	1	0	1
1	0	1	1	0
0	1	1	1	0
1	1	1	1	1

The Boolean expressions of S and C are modified as follows

$$\begin{aligned}
 S &= X'A'B + X'AB' + XA'B' + XAB \\
 &= X' (A'B + AB') + X (A'B' + AB) \\
 &= X' (A \oplus B) + X (A \oplus B)' \\
 &= X \oplus A \oplus B \\
 C &= AB + BX + AX = AB + X (A + B) \\
 &= AB + X (AB + AB' + AB + A'B) \\
 &= AB + X (AB + AB' + A'B) \\
 &= AB + XAB + X (AB' + A'B) \\
 &= AB + X (A \oplus B)
 \end{aligned}$$

- Logic diagram according to the modified expression is shown below



Notice that the full-adder developed in Figure 5.7 consists of two 2-input AND gates, two 2-input XOR (Exclusive-OR) gates and one 2-input OR gate. This contains a reduced number of gates as well as type of gates

# Combinational Logic with MSI AND LSI

A large number of integrated circuit packages are commercially available nowadays. They can be widely categorized into three groups;

1. SSI or small scale integration where the number of logic gates is limited to ten in one IC package,
2. MSI or medium scale integration where the number of logic gates is eleven to one hundred in one IC package.
3. LSI or large-scale integration containing more than one hundred gates in one package.

Some of them are fabricated for specific functions. VLSI or very large scale integration IC packages are also introduced, which perform dedicated functions achieving high circuit space reduction and interconnection reduction

# Combinational Logic with MSI AND LSI

## 1. Magnitude Comparator

A magnitude comparator is one of the useful combinational logic networks and has wide applications. It compares two binary numbers and determines if one number is greater than, less than, or equal to the other number. It is a multiple output combinational logic circuit. If two binary numbers are considered as A and B, the magnitude comparator gives three outputs for

$A > B$ ,  $A < B$ , and  $A = B$

## **2. Multiplexers or Data Selectors**