OF THE
DEPARTMENT OF
INFORMATION ENGINEERING

UNIVERSITÀ
DEGLI STUDI
DI PADOVA

Digital Systems

Optimization of Logic Functions

Karnaugh Maps

Marta Bagatin, marta.bagatin@unipd.it

Degree Course in Information Engineering

Academic Year 2023-2024

Purpose of the Lesson

- Introduce definitions/terminology that will be used for the minimization of logic functions
- Learn how to express a Boolean function in **standard form**
- Study different techniques for the **optimization of logic functions**
  - Simplification through Boolean algebra
  - Minimization with Karnaugh maps (K-maps)

Definitions and Standard Forms

Standard Forms

- We saw that a logical function can be expressed in different ways, which are all equivalent from the algebraic point of view
- Our aim is to express a function in order to facilitate its implementation with digital circuits. The **standard form** is a representation of logic functions that makes it easier to simplify the functions
- The standard form of a function is derived from the truth table. It contains **product terms** (AND between 2 or more literals, ex: $X\ Y\ \bar{Z}$) **and sum terms** (OR between 2 or more literals, ex: $X + Y + \bar{Z}$)

Minterm

- A **product** term (**AND**) in which **all the variables of the function appear exactly once** (either in direct or complemented form) is defined **minterm**
- Each minterm represents a row of the truth table
  - A minterm is equal to '1' for the combination of the binary variables of that row, and it is equal to '0' for all the other combinations
  - A minterm consists of the product (AND) of the variables in each row, with the complemented variable if the bit corresponding to that variable is '0', uncomplemented if the bit is '1'
- There are **2^n distinct minterms for n variables**

Minterms for 3 Variables

Minterms for Three Variables

X	Y	Z	Product Term	Symbol	m_0	m_1	m_2	m_3	m_4	m_5	m_6	m_7
0	0	0	$\overline{X}\,\overline{Y}\,\overline{Z}$	m_0	1	0	0	0	0	0	0	0
0	0	1	$\overline{X}\,\overline{Y}Z$	m_1	0	1	0	0	0	0	0	0
0	1	0	$\overline{X}Y\overline{Z}$	m_2	0	0	1	0	0	0	0	0
0	1	1	$\overline{X}YZ$	m_3	0	0	0	1	0	0	0	0
1	0	0	$X\overline{Y}\,\overline{Z}$	m_4	0	0	0	0	1	0	0	0
1	0	1	$X\overline{Y}Z$	m_5	0	0	0	0	0	1	0	0
1	1	0	$XY\overline{Z}$	m_6	0	0	0	0	0	0	1	0
1	1	1	XYZ	m_7	0	0	0	0	0	0	0	1

A minterm is equal to '1' for the combination of variables in that row and is '0' for all the other combinations

Maxterm

- The **dual concept of the minterm** is the maxterm
- A **sum** term (**OR**) in which **all the variables of the function appear exactly once** (either in direct or complemented form) is defined **maxterm**
- Each maxterm represents a <u>row of the truth table</u>
  - A maxterm is equal to '0' for the combination of the binary variables of that row, and it is equal to '1' for all the other combinations
  - A maxterm consists of the sum (OR) of the variables in each row, with the complemented variable if the bit corresponding to that variable is '1', uncomplemented if the bit is '0'
- There are **2^n distinct maxterms for n variables**

Maxterms for 3 Variables

Maxterms for Three Variables

X	Y	Z	Sum Term	Symbol	M_0	M_1	M_2	M_3	M_4	M_5	M_6	M_7
0	0	0	$X + Y + Z$	M_0	0	1	1	1	1	1	1	1
0	0	1	$X + Y + \overline{Z}$	M_1	1	0	1	1	1	1	1	1
0	1	0	$X + \overline{Y} + Z$	M_2	1	1	0	1	1	1	1	1
0	1	1	$X + \overline{Y} + \overline{Z}$	M_3	1	1	1	0	1	1	1	1
1	0	0	$\overline{X} + Y + Z$	M_4	1	1	1	1	0	1	1	1
1	0	1	$\overline{X} + Y + \overline{Z}$	M_5	1	1	1	1	1	0	1	1
1	1	0	$\overline{X} + \overline{Y} + Z$	M_6	1	1	1	1	1	1	0	1
1	1	1	$\overline{X} + \overline{Y} + \overline{Z}$	M_7	1	1	1	1	1	1	1	0

A maxterm is equal to '0' for the combination of variables in that row and is '1' for all the other combinations

Minterms and Maxterms

Minterms for Three Variables

X	Y	Z	Product Term	Symbol
0	0	0	$\overline{X}\,\overline{Y}\,\overline{Z}$	m_0
0	0	1	$\overline{X}\,\overline{Y}Z$	m_1
0	1	0	$\overline{X}Y\overline{Z}$	m_2
0	1	1	$\overline{X}YZ$	m_3
1	0	0	$X\overline{Y}\,\overline{Z}$	m_4
1	0	1	$X\overline{Y}Z$	m_5
1	1	0	$XY\overline{Z}$	m_6
1	1	1	XYZ	m_7

Maxterms for Three Variables

X	Y	Z	Sum Term	Symbol
0	0	0	$X+Y+Z$	M_0
0	0	1	$X+Y+\overline{Z}$	M_1
0	1	0	$X+\overline{Y}+Z$	M_2
0	1	1	$X+\overline{Y}+\overline{Z}$	M_3
1	0	0	$\overline{X}+Y+Z$	M_4
1	0	1	$\overline{X}+Y+\overline{Z}$	M_5
1	1	0	$\overline{X}+\overline{Y}+Z$	M_6
1	1	1	$\overline{X}+\overline{Y}+\overline{Z}$	M_7

A minterm and a maxterm with the **same subscript** are the **complement** of each other

Example: $M_3 = X + \bar{Y} + \bar{Z} = \overline{\bar{X}\,Y\,Z} = \overline{m_3}$

Minterms and Maxterms of a Function

- The **minterms of a logic function** are the minterms for which the **function is equal to '1'**
- The **maxterms of a logic function** are the maxterms for which the **function is equal to ‘0’**

- Minterms and maxterms provide **two complementary descriptions** of a logic function

Standard Forms

- All Boolean functions can be represented in **standard form** starting from their truth table
- There are two types of standard forms

1) **Standard SOP form** (<u>S</u>um <u>O</u>f <u>P</u>roducts): **sum of all minterms** of the function
2) **Standard POS form** (<u>P</u>roduct <u>O</u>f <u>S</u>ums): **product of all maxterms** of the function

- Typically the standard forms are **redundant** and can be simplified. Standard forms are convenient as a starting point for the minimization of logic functions

Example: Boolean Function with 3 Variables

X	Y	Z	F	$\overline{F}$
0	0	0	1	0
0	0	1	0	1
0	1	0	1	0
0	1	1	0	1
1	0	0	0	1
1	0	1	1	0
1	1	0	0	1
1	1	1	1	0

- The function F can be expressed as a **sum of minterms of the function** (rows of the truth table with a ‘1’):

$$F = \bar{X}\bar{Y}\bar{Z} + \bar{X}Y\bar{Z} + X\bar{Y}Z + XYZ = m_0 + m_2 + m_5 + m_7$$

$$F = \sum m(0,2,5,7)$$

Example: Boolean Function with 3 Variables

X	Y	Z	F	$\overline{F}$
0	0	0	1	0
0	0	1	0	1
0	1	0	1	0
0	1	1	0	1
1	0	0	0	1
1	0	1	1	0
1	1	0	0	1
1	1	1	1	0

- Alternatively, F can be expressed as a **product of maxterms**:

$$F = (X + Y + \bar{Z}) \cdot (X + \bar{Y} + \bar{Z}) \cdot (\bar{X} + Y + Z) \cdot (\bar{X} + \bar{Y} + Z)$$

$$F = M_1 \cdot M_3 \cdot M_4 \cdot M_6$$

$$F = \prod M(1, 3, 4, 6)$$

In fact:

$$\bar{F} = \bar{X}\bar{Y}Z + \bar{X}YZ + X\bar{Y}\bar{Z} + XY\bar{Z}$$

$$= m_1 + m_3 + m_4 + m_6 = \overline{\overline{m_1 + m_3 + m_4 + m_6}}$$

$$= \overline{\overline{m_1} \cdot \overline{m_3} \cdot \overline{m_4} \cdot \overline{m_6}} = \overline{M_1 \cdot M_3 \cdot M_4 \cdot M_6} \qquad \text{(since } m_i = \overline{M_i}\text{)}$$

Example: Boolean Function with 3 Variables

X	Y	Z	F	$\overline{F}$
0	0	0	1	0
0	0	1	0	1
0	1	0	1	0
0	1	1	0	1
1	0	0	0	1
1	0	1	1	0
1	1	0	0	1
1	1	1	1	0

Note: the subscripts of the maxterms used in the standard POS form are always the same as the minterms used in the standard SOP form of the complemented function:

$$F(X,Y,Z) = \prod M(1,3,4,6) \qquad \bar{F}(X,Y,Z) = \sum m(1,3,4,6)$$

Example: Conversion to Standard Form

Convert the following logic function to standard form:

$$E(X,Y,Z) = \bar{Y} + \bar{X}\bar{Z}$$

The expression is not a SOP form (as each term does not contain all 3 variables X, Y, Z)

X	Y	Z	E
0	0	0	1
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	0
1	1	1	0

SOP standard form:

$$E(X,Y,Z) = \sum m(0,1,2,4,5)$$

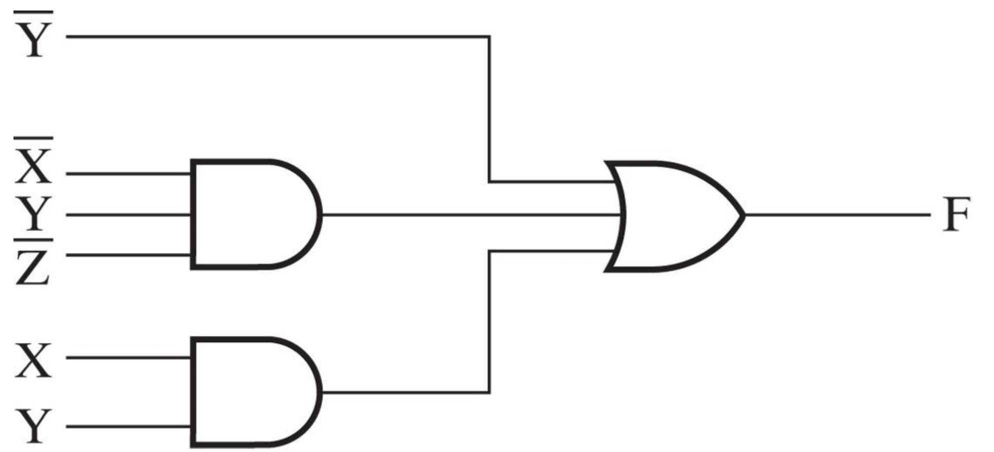
POS standard form:

$$E(X,Y,Z) = \prod M(3,6,7)$$

Simplified SOP Form

- A simplified SOP form is **more compact than the standard SOP form**: in a simplified form, the product terms do not necessarily contain all the literals!
- A simplified SOP form can be obtained from the standard SOP form, by simplifying it with the Boolean algebra rules (i.e. reducing the number of product terms and the number of literals in the terms)
  - Example of a simplified SOP form

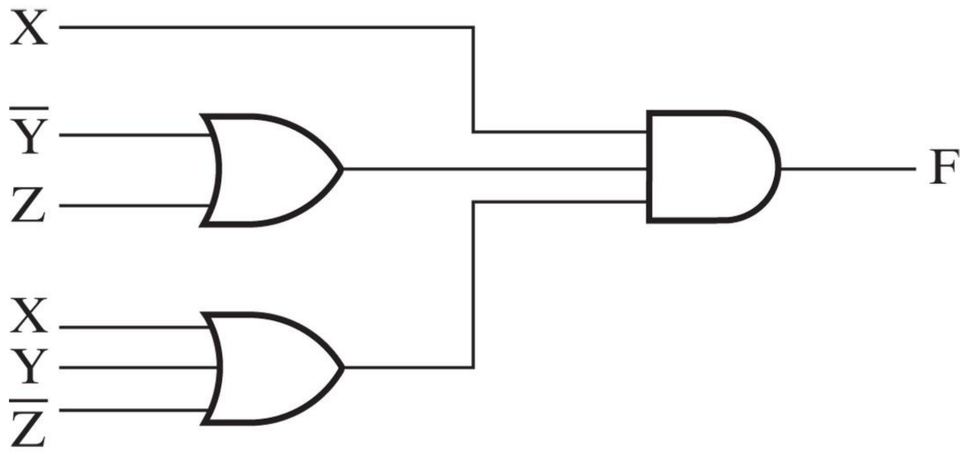
$$F = \bar{Y} + \bar{X}Y\bar{Z} + XY$$



Simplified POS Form

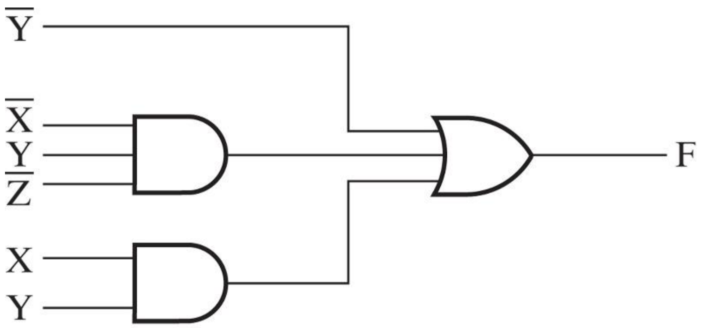
- A simplified POS **more compact than the standard SOP form**: the sum terms do not necessarily contain all the literals! Dual representation with respect to simplified SOP
- A simplified POS form can be obtained from the standard POS form, by simplifying it with the Boolean algebra rules (i.e. reducing the number of sum terms and the number of literals in the terms)
  - Example of a simplified POS form

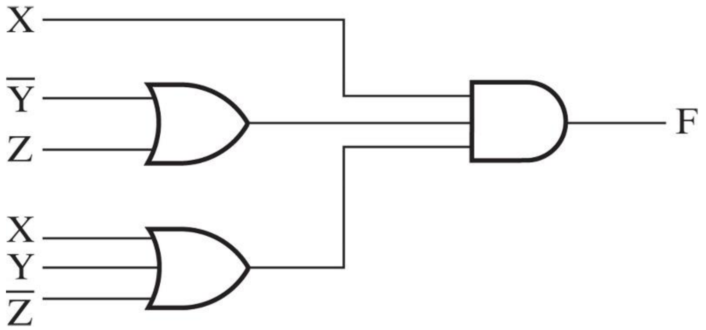
$$F = X(\bar{Y} + Z)(X + Y + \bar{Z})$$



Levels of Implementation

- Boolean expressions in **SOP or POS form** can be **implemented with two-levels circuits**





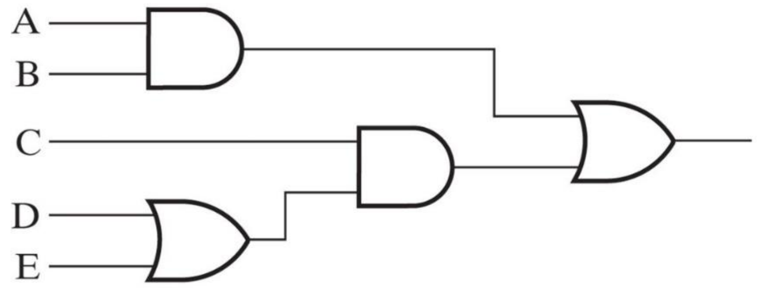
→ **Two-level logic gates implementations**

- On the other hand, expressions that are neither in the form of a product of sums, nor in the form of a sum of products must be implemented with multi-level circuits

Example: Multi-level Implementation

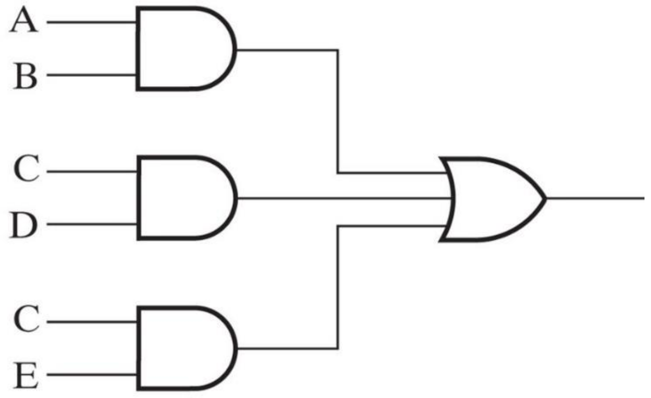
- Let's consider an expression that is not in sum-of-products-form: $F = AB + C(D + E)$

  This espression corresponds to a 3-level gate implementation:



- Applying the distributive property, we obtain a SOP form, corresponding to a 2-level implementation:

$$F = AB + CD + CE$$



Better to choose a two-level or multi-level implementation? It depends on the application! (number of gates, number of inputs, acceptable delay, technology platform)

Two-Level Circuit Optimization

Cost Criteria

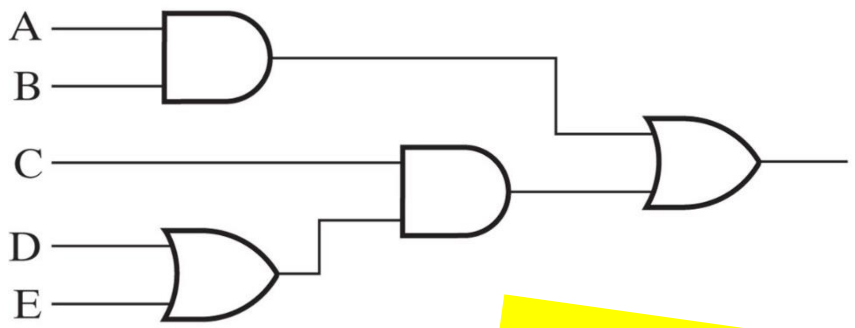
- Two alternative criteria can be used for the optimization of a circuit implementing a logic function
  - **Minimize the literal cost** (i.e. minimize the number of literals)
    - The literal cost can be found directly from the Boolean expression, counting the number of literal appearances
    - The literal cost does not always accurately represent the complexity of a circuit
  - **Minimize the gate input cost, GIC** (i.e. minimize the number of gate inputs)
    - The gate input cost can be found from the logic diagram, counting the number of inputs of all the gates. It can be also calculated from the Boolean expression
    - The GIC is more accurate than the literal cost in evaluating the cost of a modern circuit (with more than two levels), being proportional to the number of transistors and wires
- As we will see, regardless of the chosen criteria, the **minimum cost form is not necessarily unique!**

Example 1: Cost Criteria

Two expressions of function F are given. Find the expression with lower cost:

1) $\boldsymbol{F = AB + C(D + E)}$

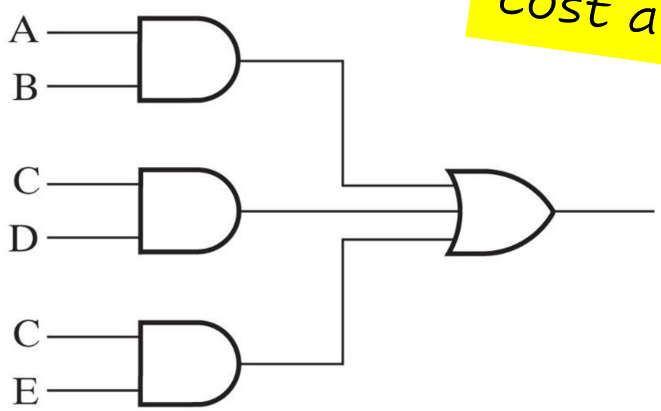
5 literals, 8 inputs ←



The first expression minimizes both the literal cost and the gate input cost!

2) $\boldsymbol{F = AB + CD + CE}$

6 literals, 9 inputs ←



Example 2: Cost Criteria

Two expressions of function G are given. Find the expression with lower cost:

1) $G = ABCD + \bar{A}\,\bar{B}\,\bar{C}\,\bar{D}$ → 8 literals, 10 inputs

2) $G = (\bar{A} + B)(\bar{B} + C)(\bar{C} + D)(\bar{D} + A)$ → 8 literals, 12 inputs

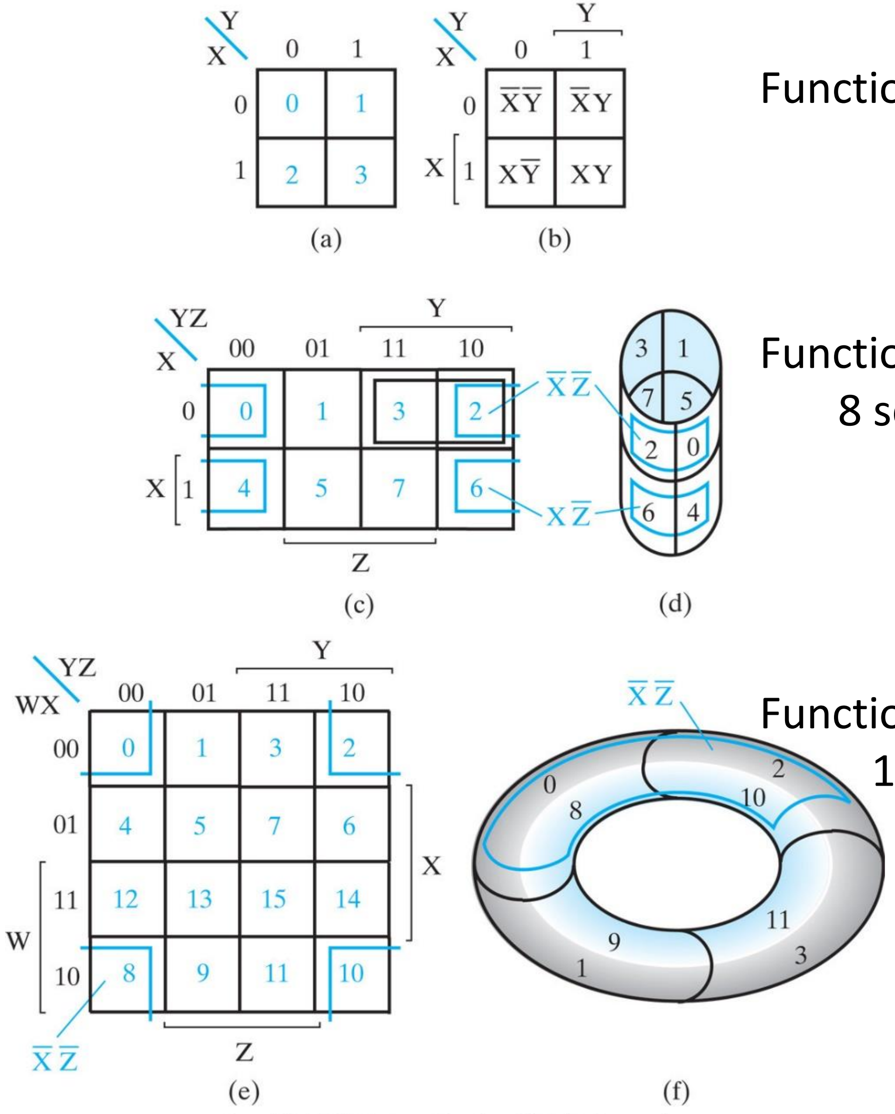
The two expressions have the same literal cost, but the first minimizes the gate input cost!

Karnaugh Maps

Simplification Methods

- The truth table of a logic function is unique, but the Boolean expression is not unique!
- Boolean expressions can be simplified by algebraic manipulation, but there is not a specific algorithm and it is difficult to understand if the simplest form has been reached
- **Karnaugh maps** (K-maps) provide a straightforward procedure **to optimize Boolean functions,** i.e. to find the optimum sum-of-products or product-of-sums standard form
- K-map is a graphic method that exploits the idea of Hamming distance
- K-maps are effective for Boolean functions with up to 4 variables (cumbersome with 5 or 6 variables)
- Methods based on K-maps are used in automatic tools for the synthesis of combinational logic circuits

Karnaugh Maps



Functions with 2 variables: 4 squares

Functions with 3 variables: 8 squares (cylinder)

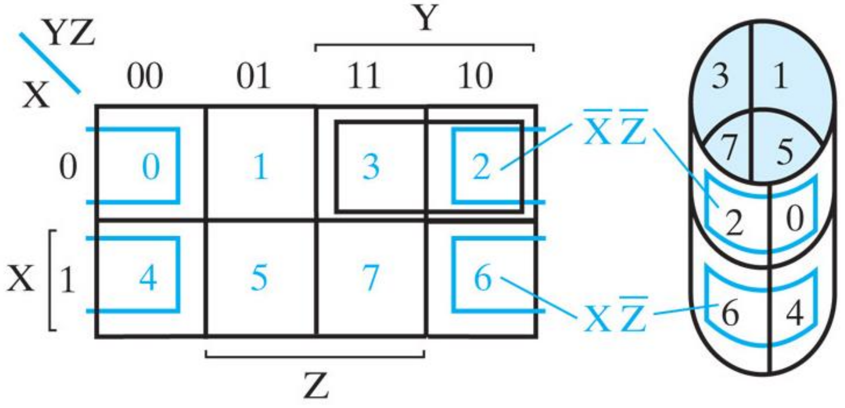
Functions with 4 variables: 16 squares (toroid)

- **Every square** represents a row of the truth table of the function, that is a **minterm**
- **Adjacent squares differ by only one literal** (Gray code), i.e. they have a Hamming distance equal to 1
- The map should be imagined as closed on itself (e.g. in the 4-variable map, square 3 on the upper edge is adjacent to square 11 on the lower edge, same for square 14 on the right and square 12 on the left)

Karnaugh Maps

- A function can be expressed as the sum of its minterms, so we can represent it graphically in the K-map putting a '1' in the squares for which the function is equal to 1 (i.e. squares corresponding to the minterms of the function)
- The K-map represents a **visual diagram of all the possible SOP forms of a function**
- In particular, we will use the map to find the optimized **SOP form**
- For the duality principle, if we consider the 0’s in the map, we will find the optimized **POS form**

Karnaugh Maps



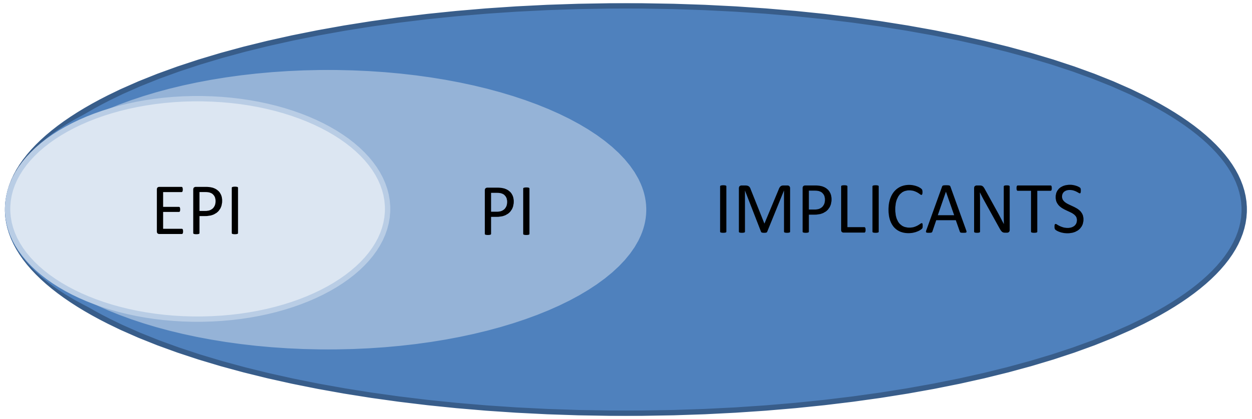
Combinations (X, Y, Z) = 011 (m_3) and (X, Y, Z) = 010 (m_2) are adjacent (bits X and Y have the same value, only Z is different) → the two squares corresponding to m_3 and m_2 minterms share an edge on the map

- In case both m_3 and m_2 minterms belong to the function, we can consider a single rectangle representing the two minterms, instead of the two separate squares
- This grouping corresponds to the following algebraic simplification of the product terms:

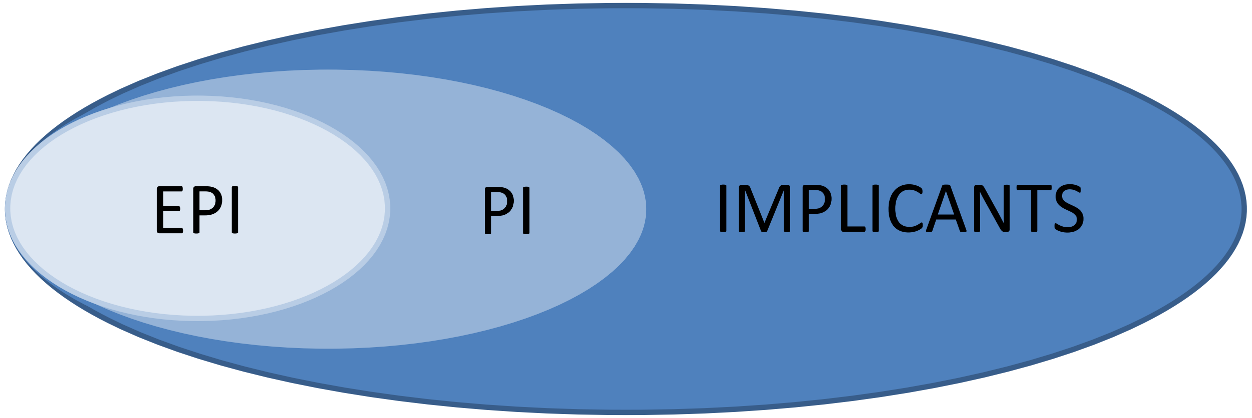
$$\bar{X}YZ + \bar{X}Y\bar{Z} = \bar{X}Y(Z + \bar{Z}) = \bar{X}Y$$

Implicants

- **Implicant**: product term for which the function is '1'
- **Prime implicant (PI)**: implicant that is **not contained** in any other implicant (cannot be made larger)
- **Essential Prime Implicant (EPI)**: implicant that contains at least one minterm of the function that is not covered by any other prime implicants (i.e. an EPI is the only prime implicant that contains one or more minterms)



Coverage of a Function

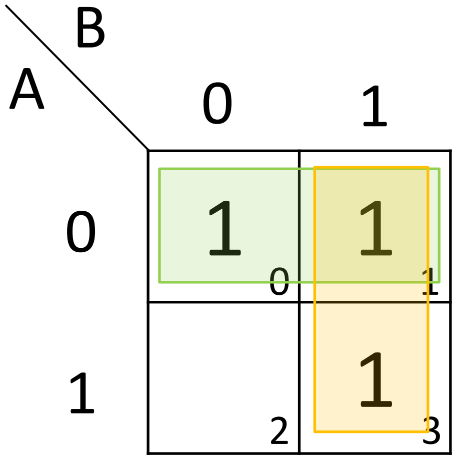


- **Coverage of a function**: contains at least all the EPIs (and possibly other PIs)
- **Our aim is to find the minimal coverage of the function**, which is obtained by taking the largest rectangles containing ‘1’ on the K-map
- We will analyze the procedure by examples

2-variable Functions

A	B	F
0	0	1
0	1	1
1	0	0
1	1	1

$$m_0 = \bar{A} \cdot \bar{B}$$
$$m_1 = \bar{A} \cdot B$$
$$m_2 = A \cdot \bar{B}$$
$$m_3 = A \cdot B$$

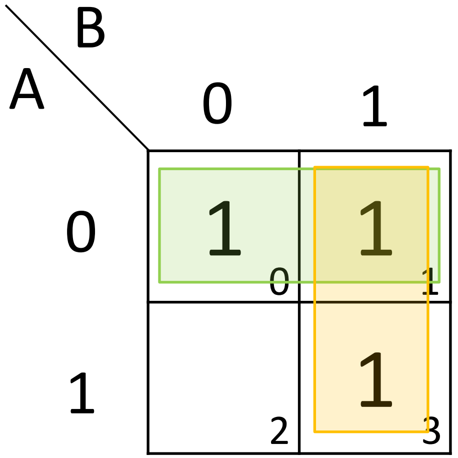


1) Fill the map with 1’s, following the truth table (or the definition of the function given by the minterms)
2) Find the largest 'rectangles' (groupings of 1-2-4 squares) consisting of adjacent 1’s (PI). The aim is to find the minimum number of rectangles that cover all the squares with logic '1'

2-variable Functions

A	B	F
0	0	1
0	1	1
1	0	0
1	1	1

$m_0 = \bar{A} \cdot \bar{B}$
$m_1 = \bar{A} \cdot B$
$m_2 = A \cdot \bar{B}$
$m_3 = A \cdot B$



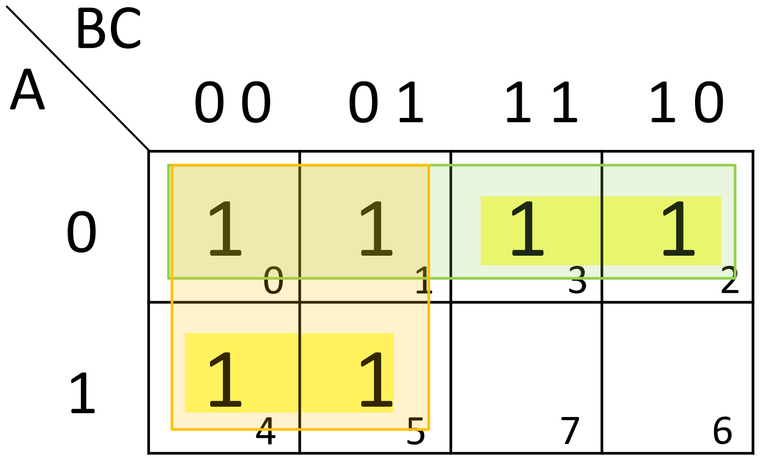
3) Find the EPIs (identifying the minterms covered by a single implicant)
4) Determine the minimum coverage of the function: in this case EPIs are enough, since all PIs are also EPIs

$$F = \bar{A} + B$$

3-variable Functions

Example 1

$$F(A,B,C) = \sum m(0,1,2,3,4,5)$$



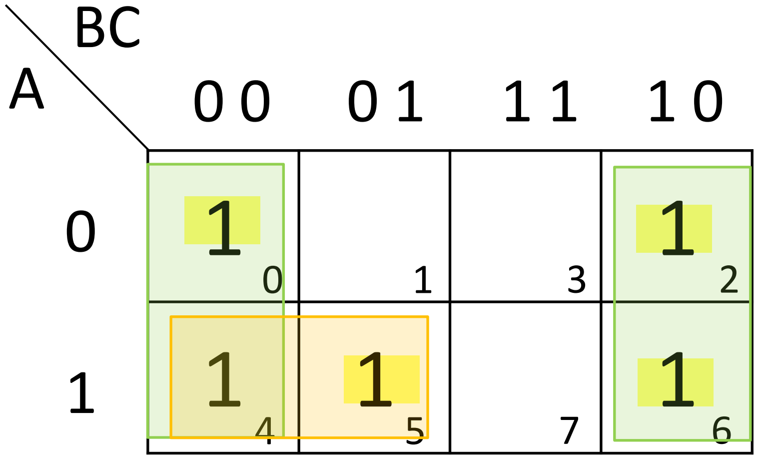
1) Fill the map with 1’s following the minterms definition
2) Find the largest rectangles with adjacent 1’s (PI), made of 1-2-4-8 squares
3) Find the EPIs (identifying the minterms covered by a single implicant)
4) Determine the minimum coverage of the function (EPIs are enough in this case)

$$F = \bar{A} + \bar{B}$$

3-variable Functions

<u>Example 2</u>

$$G(A, B, C) = \sum m(0, 2, 4, 5, 6)$$



- There are 2 prime implicants
- Both PIs are essential

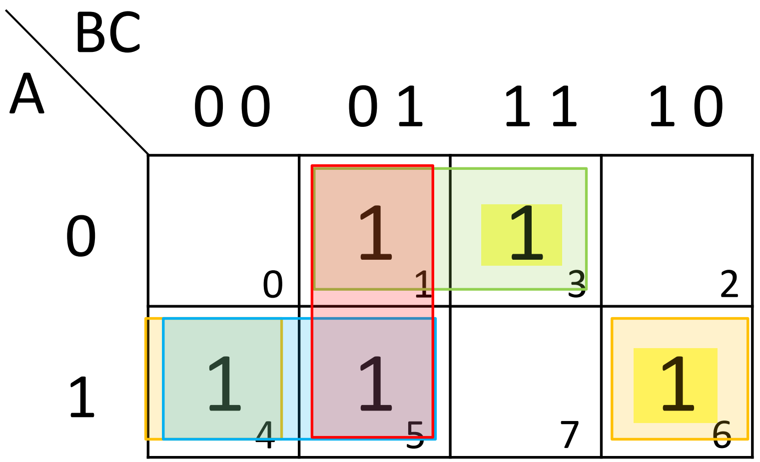
=> The optimized SOP form of the function is :

$$G(A, B, C) = \mathrm{A}\bar{B} + \bar{C}$$

3-variable Functions

Example 3

$$H(A,B,C) = \sum m(1,3,4,5,6)$$



- There are 4 prime implicants
- Of these 4 PIs, 2 are essential prime implicants
- Which other implicant(s) should we choose besides the 2 EPIs?

3-variable Functions

Example 3

$$H(A,B,C) = \sum m(1,3,4,5,6)$$

A \ BC	0 0	0 1	1 1	1 0
0	0	1	3	2
1	4	1 5	7	6

- m_5 (not covered by the EPIs) can be covered either by (4,5) or by (1,5)

=> Two possible (equivalent) minimized SOP forms:

$$H(A,B,C) = \bar{A}C + A\bar{C} + A\bar{B}$$ -> if we choose (4,5)

$$H(A,B,C) = \bar{A}C + A\bar{C} + \bar{B}C$$ -> if we choose (1,5)

4-variable Functions

<u>Example 1</u>: Simplify the following function via K-map

$$F(A, B, C, D) =$$
$$= \sum m(0, 1, 2, 4, 5, 6, 8, 9, 10, 12, 13)$$

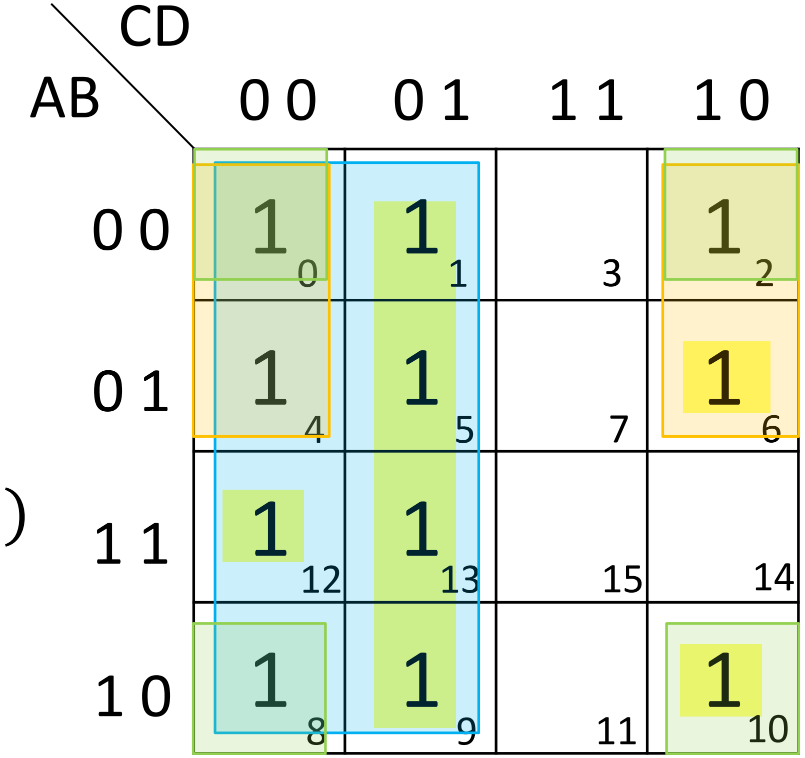
AB \ CD	0 0	0 1	1 1	1 0
0 0	1_{0}	1_{1}	$_{3}$	1_{2}
0 1	1_{4}	1_{5}	$_{7}$	1_{6}
1 1	1_{12}	1_{13}	$_{15}$	$_{14}$
1 0	1_{8}	1_{9}	$_{11}$	1_{10}

1) Fill the map with 1’s following the minterm definition
2) Locate the largest adjacent 1’s rectangles (PI), made of 1-2-4-8-16 squares

4-variable Functions

Example 1: Simplify the following function via K-map

$$F(A,B,C,D) =$$
$$= \sum m(0,1,2,4,5,6,8,9,10,12,13)$$



- There are 3 prime implicants
- All 3 IPs are also IPEs

=> The minimum SOP form is:

$$F(A,B,C,D) = \bar{C} + \bar{A}\bar{D} + \bar{B}\bar{D}$$

4-variable Functions

Example 2: Simplify the following function via K-map

$$G(A,B,C,D) = \bar{A}\bar{C}\bar{D} + \bar{A}D + \bar{B}C + CD + A\bar{B}\bar{D}$$

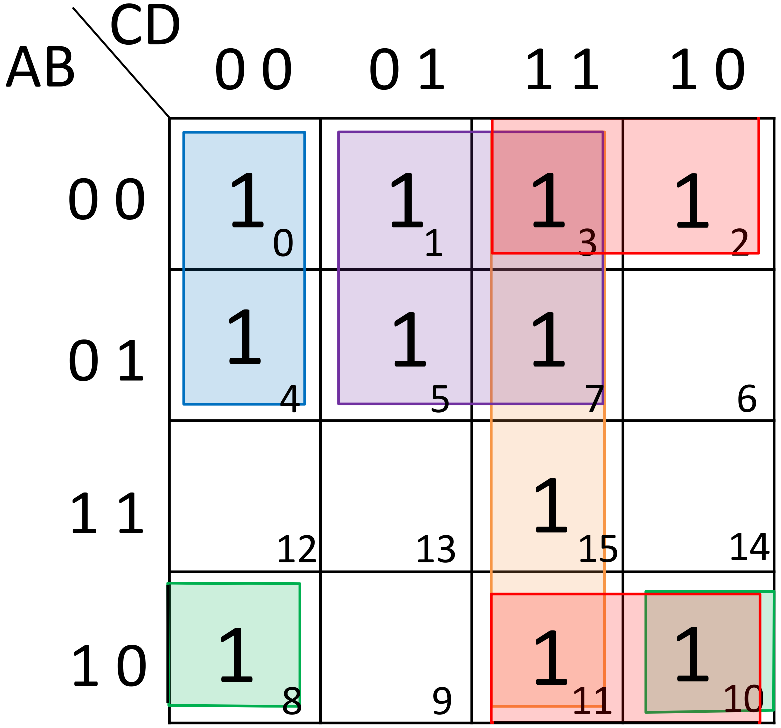
Find the implicants on the map and fill them with 1’s:

AB \ CD	0 0	0 1	1 1	1 0
0 0	0	1	3	2
0 1	4	5	7	6
1 1	12	13	15	14
1 0	8	9	11	10

4-variable Functions

Example 2: Simplify the following function via K-map

$$G(A,B,C,D) = \bar{A}\bar{C}\bar{D} + \bar{A}D + \bar{B}C + CD + A\bar{B}\bar{D}$$



4-variable Functions

Now we can proceed with the simplification from the K-map:

AB \ CD	0 0	0 1	1 1	1 0
0 0	1 [0]	1 [1]	1 [3]	1 [2]
0 1	1 [4]	1 [5]	1 [7]	[6]
1 1	[12]	[13]	1 [15]	[14]
1 0	1 [8]	[9]	1 [11]	1 [10]

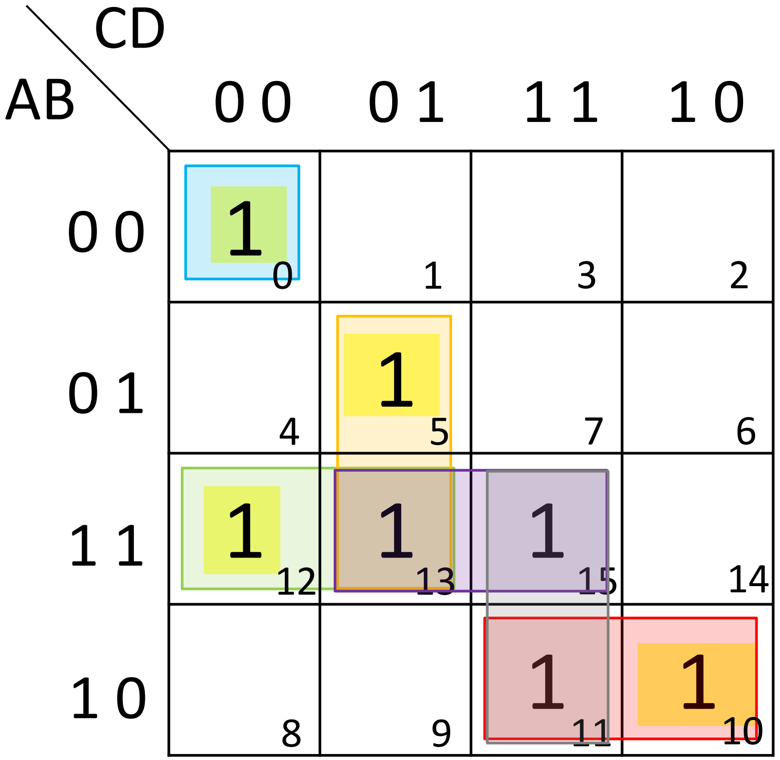
There are 3 PIs, which are also EPIs, so the simplified function is:

$$G(A,B,C,D) = \bar{A}\bar{C} + CD + \bar{B}\bar{D}$$

→ this expression is simpler than the original one!

4-variable Functions

Example 3: consider the function represented by the following K-map and find the optimized SOP form:



- There are 6 PIs
- 4 PIs are also EPIs

4-variable Functions

Example 3: consider the function represented by the following K-map and find the optimized SOP form:

AB \ CD	0 0	0 1	1 1	1 0
0 0	0	1	3	2
0 1	4	5	7	6
1 1	12	13	1 (15)	14
1 0	8	9	11	10

- The 4 EPIs cover all '1' except from m_{15}
- m_{15} can be covered by (13,15) or (15,11)

- We have 2 minimum equivalent forms:

$$F = \bar{A}\bar{B}\bar{C}\bar{D} + B\bar{C}D + AB\bar{C} + A\bar{B}C + \begin{matrix} ACD \\ \text{or} \\ ABD \end{matrix}$$

POS Optimization

- So far we have seen the optimization of functions in SOP (Sum of Products) form. The resulting logic circuit is a set of AND gates (level 1, to realize product terms, i.e. minterms) and a set of OR gates (level 2, to realize the sum)
- In a dual way, we can optimize the function in POS (Product of Sums) form

POS Optimization

<u>Example</u>: Find the minimal POS form for the following function:

$$F(A, B, C, D) = \sum m(0, 1, 2, 5, 8, 9, 10)$$

- Let's work on the complemented function and find the largest rectangles made of 0's
- There are 3 PIs, which are also EPIs
- The optimized SOP form of the complemented function is:

$$\bar{F} = AB + CD + B\bar{D}$$

$$\Rightarrow\ F = (\bar{A} + \bar{B})\,(\bar{C} + \bar{D})\,(\bar{B} + D)$$

AB \ CD	0 0	0 1	1 1	1 0
0 0	1_{0}	1_{1}	0_{3}	1_{2}
0 1	0_{4}	1_{5}	0_{7}	0_{6}
1 1	0_{12}	0_{13}	0_{15}	0_{14}
1 0	1_{8}	1_{9}	0_{11}	1_{10}

"Don’t care" Conditions

- Don’t care conditions occur when a **function is not completely specified**
- So far we have assumed that the function was equal to '0' when it was not equal to '1', but this is not always true
- For certain applications, the output of the function can be unspecified for some combinations of the input variables (either because those input combinations never occur or because we do not care what are the outputs produced by those combinations)
- Don't care conditions will be indicated with an ‘X’ (or ‘-’) in the K-map: these squares can take the value '0' or '1', depending on what is more convenient to simplify the function

"Don't care" Conditions: Example

Example: Find the optimized SOP form of the following incompletely specified function

$$F(A,B,C,D) = \sum m(1,3,7,11,15) \qquad d(A,B,C,D) = \sum m(0,2,5)$$

AB \ CD	0 0	0 1	1 1	1 0
0 0	X $_{0}$	1 $_{1}$	1 $_{3}$	X $_{2}$
0 1	0 $_{4}$	X $_{5}$	1 $_{7}$	0 $_{6}$
1 1	0 $_{12}$	0 $_{13}$	1 $_{15}$	0 $_{14}$
1 0	0 $_{8}$	0 $_{9}$	1 $_{11}$	0 $_{10}$

We can include or not the squares with 'X', choosing the implicants that lead to the simplest form of the function

$$F(A,B,C,D) = CD + \bar{A}\bar{B}$$

"Don’t care" Conditions: Example

Example: Find the optimized SOP form of the following incompletely specified function

$$F(A,B,C,D) = \sum m(1,3,7,11,15) \qquad d(A,B,C,D) = \sum m(0,2,5)$$

AB \ CD	0 0	0 1	1 1	1 0
0 0	X_{0}	1_{1}	1_{3}	X_{2}
0 1	0_{4}	X_{5}	1_{7}	0_{6}
1 1	0_{12}	0_{13}	1_{15}	0_{14}
1 0	0_{8}	0_{9}	1_{11}	0_{10}

An alternative (equivalent) choice can be:

$$F(A,B,C,D) = CD + \bar{A}D$$

The two expressions are **algebraically different**, **but the outputs are the same for the input combinations of interest, i.e. they both realize the desired function**

"Don’t care" Conditions: Exercize

- Realize a function indicating whether a number between 0 and 9 is prime

  Hints:

  – 4 binary digits are needed to represent the numbers from 0 to 9

  – The truth table of the function contains 16 rows, but only 10 of these rows are relevant to the function: the remaining rows are don't care conditions

XOR and XNOR Functions

- We defined the **Exclusive OR** (**XOR**) function:

$$\boldsymbol{X \oplus Y = X\overline{Y} + \overline{X}Y}$$

and its complement **Exclusive NOR** (**XNOR**):

$$\boldsymbol{\overline{X \oplus Y} = XY + \overline{X}\,\overline{Y}}$$

- They are commutative, associative, and have the following properties:

$$X \oplus 0 = X \qquad X \oplus 1 = \bar{X}$$

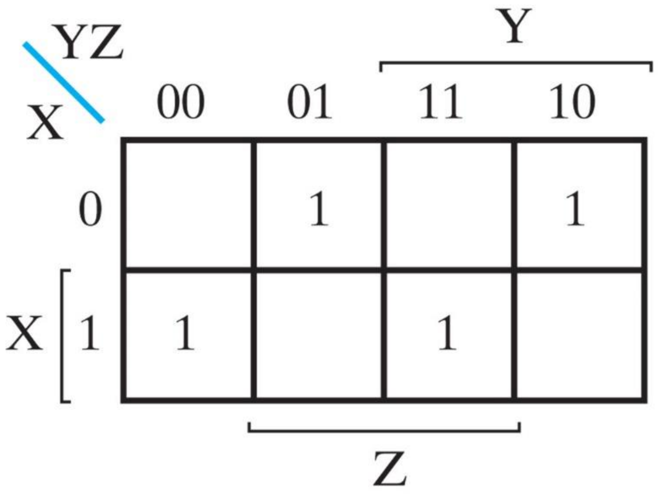
$$X \oplus X = 0 \qquad X \oplus \bar{X} = 1$$

$$X \oplus \bar{Y} = \overline{X \oplus Y} \qquad \bar{X} \oplus Y = \overline{X \oplus Y}$$

XOR: Odd Function

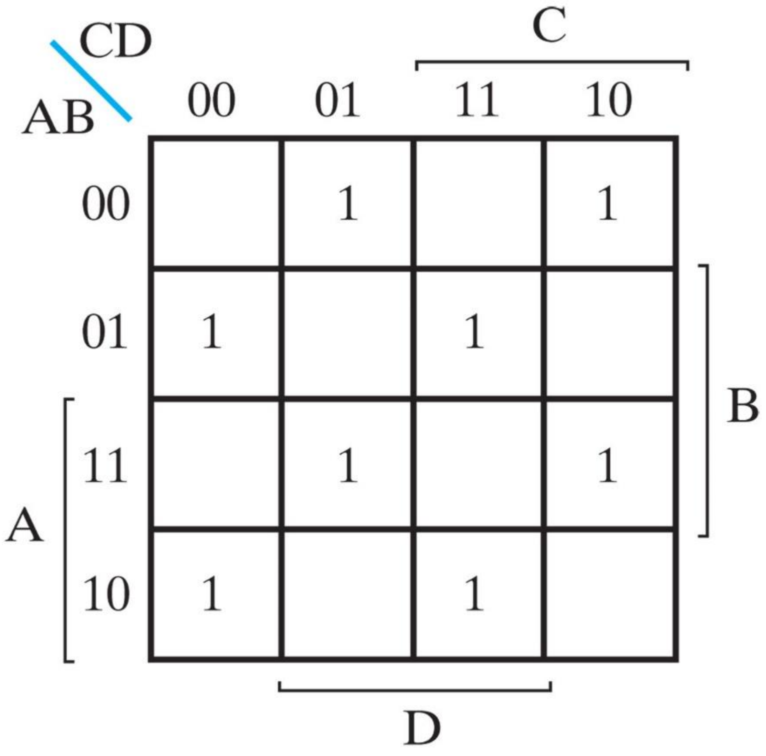
- The **XOR function can be extended to 3 or more variables**
- XOR with more than 2 inputs is known as **odd function**

3 variables:



(a) $X \oplus Y \oplus Z$

4 variables:

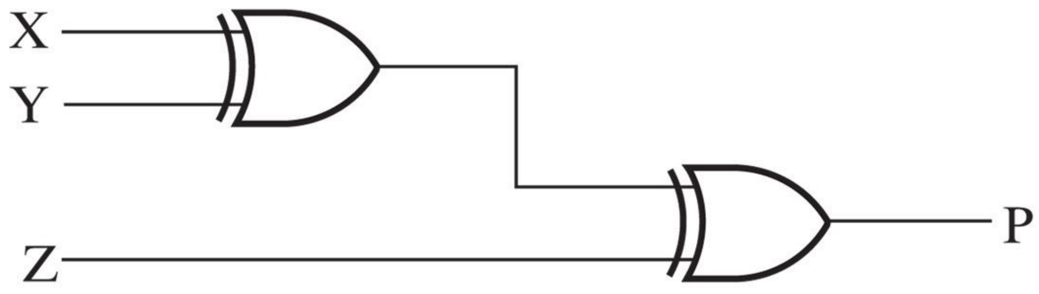


(b) $A \oplus B \oplus C \oplus D$

- Minterms have Hamming distance 2 from each other on the K-map
- The binary values of the minterms of the odd function have an odd number of 1's

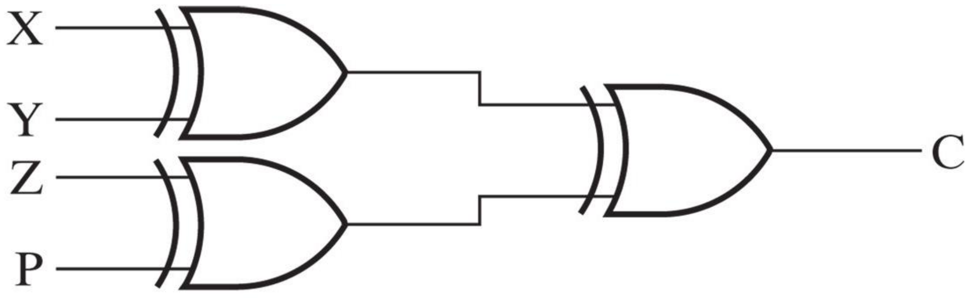
XOR: Odd Function

- The logic circuit for the **3-variable odd** function is:



$P = X \oplus Y \oplus Z$

- The logic circuit for the **4-variable odd** function is:



(b) $C = X \oplus Y \oplus Z \oplus P$

XNOR: Even Function

- The complement of the odd function is called **even function**
- The minterms of the even function have Hamming distance 2 on the K-map and their binary values have an even number of 1’s
- The logic circuit of the even function can be obtained from the odd function circuit, replacing the output gate with a XNOR gate

Summary

- **Minterm**: product of all variables (direct or complemented) of a function, with each variable appearing once
- **Maxterm**: sum of all variables (direct or complemented) of a function, with each variable appearing once
- **Minterm of a function**: minterm for which the function is equal to ‘1’
- **Maxterm of a function**: maxterm for which the function is equal to ‘0’
- **Standard SOP form**: sum of all the minterms of the function
- **Standard POS form**: product of all the maxterms of the function
- **Optimized SOP or POS form**: expression in SOP or POS form, simplified as much as possible with respect to the standard form (terms do not necessarily contain all literals)
- **Cost criteria**: number of literals (literal cost) or number of gate input (gate input cost)
- **Karnaugh Map (K-map)**: graphical method to optimize a function in SOP or POS form

Disclaimer

Figures from *Logic and Computer Design Fundamentals*, Fifth Edition, GE Mano | Kime| Martin