





---

# Karnaugh Maps

# KARNAUGH MAPS

- Function Simplification
- Algebraic Simplification
- Half Adder
- Introduction to K-maps
- How to use K-maps
- Converting to Minterms Form
- Prime Implicants and Essential Prime Implicants
- Example on Finding Minimal SOP Expression
- Finding POS Expression
- Don't-care Conditions

# FUNCTION SIMPLIFICATION

- Why simplify?
  - Simpler expression uses fewer logic gates.
  - Thus cheaper, uses less power, (sometimes) faster.
- Techniques
  - Algebraic
    - Using theorems
    - Open-ended; requires skills
  - Karnaugh Maps
    - Easy to use
    - Limited to no more than 6 variables
  - Quine-McCluskey
    - Suitable for automation
    - Can handle many variables (but computationally intensive)

# ALGEBRAIC SIMPLIFICATION (1/4)

- Aims to minimise
  - Number of literals, and
  - Number of terms
- But sometimes conflicting, so let's aim at reducing the number of literals for the examples in the next few slides.
- Difficult – needs good algebraic manipulation skills.

# ALGEBRAIC SIMPLIFICATION (2/4)

## ■ Example 1: Simplify $(x+y) \cdot (x+y') \cdot (x'+z)$

$$\begin{aligned} & (x+y) \cdot (x+y') \cdot (x'+z) \\ &= (x \cdot x + x \cdot y' + x \cdot y + y \cdot y') \cdot (x'+z) && \text{(associativity)} \\ &= (x + x \cdot (y' + y) + 0) \cdot (x'+z) && \text{(idemp, assoc., complement)} \\ &= (x + x \cdot 1) \cdot (x'+z) && \text{(complement, identity)} \\ &= (x + x) \cdot (x'+z) && \text{(identity)} \\ &= x \cdot (x'+z) && \text{(idempotency)} \\ &= x \cdot x' + x \cdot z && \text{(associativity)} \\ &= 0 + x \cdot z && \text{(complement)} \\ &= x \cdot z && \text{(identity)} \end{aligned}$$

## ■ Number of literals reduced from 6 to 2.

# ALGEBRAIC SIMPLIFICATION (3/4)

- Example 2: Find minimal SOP and POS expressions of

$$F(x,y,z) = x' \cdot y \cdot (z + y' \cdot x) + y' \cdot z$$

$$x' \cdot y \cdot (z + y' \cdot x) + y' \cdot z$$
$$=$$

- Minimal SOP:

- Minimal POS:

# ALGEBRAIC SIMPLIFICATION (4/4)

- Example 3: Find minimal SOP expression of

$$F(a,b,c,d) = a \cdot b \cdot c + a \cdot b \cdot d + a' \cdot b \cdot c' + c \cdot d + b \cdot d'$$

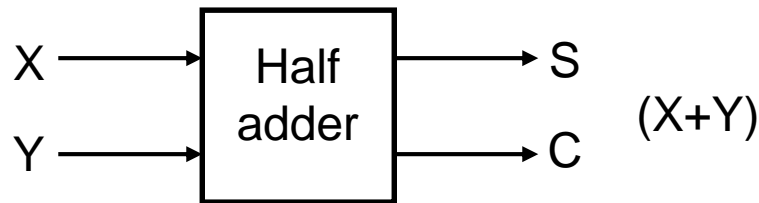
$$a \cdot b \cdot c + \mathbf{a \cdot b \cdot d} + a' \cdot b \cdot c' + c \cdot d + \mathbf{b \cdot d'}$$

=



# HALF ADDER (1/2)

- **Half adder** is a circuit that adds 2 single bits (X, Y) to produce a result of 2 bits (C, S).
- The **black-box representation** and **truth table** for half adder are shown below.



<i>Inputs</i>		<i>Outputs</i>	
X	Y	C	S
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

# HALF ADDER (2/2)

- In canonical form (sum-of-minterms):

- $C = X \cdot Y$

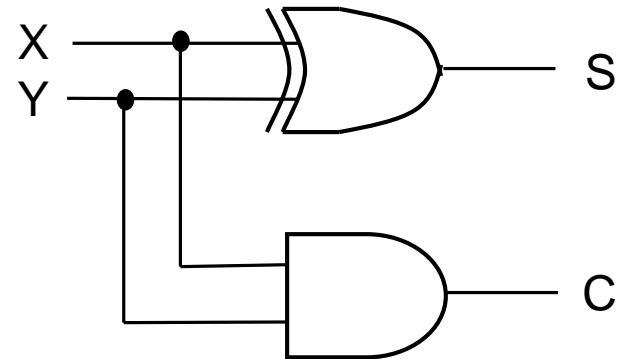
- $S = X' \cdot Y + X \cdot Y'$

- Output S can be simplified further (though no longer in SOP form):

- $S = X' \cdot Y + X \cdot Y' = X \oplus Y$

- Implementation of a half adder

X	Y	C	S
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0



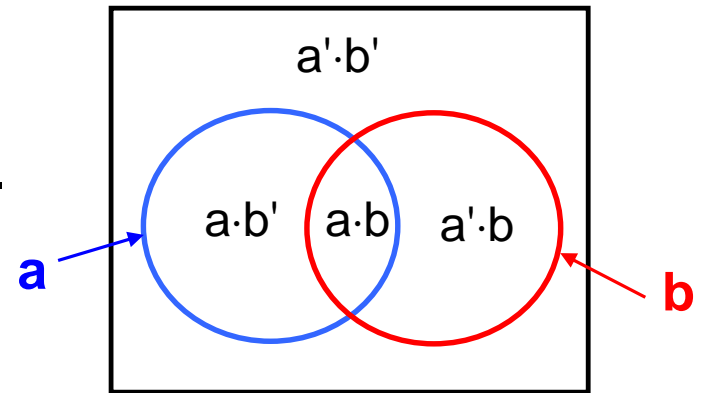
---

# INTRODUCTION TO K-MAPS

- Systematic method to obtain **simplified (minimal) sum-of-products (SOP) expressions**.
- Objective: *Fewest* possible product terms and literals.
- Diagrammatic technique based on a special form of *Venn diagram*.
- Advantage: Easy to use.
- Disadvantage: Limited to 5 or 6 variables.

# VENN DIAGRAM

- Example: 2 variables **a** and **b** represented by 2 circular regions. There are 4 minterms, each occupying their respective space.



- A set of minterms represents a certain Boolean function. Examples:

$$\{ a \cdot b, a \cdot b' \} \rightarrow a \cdot b + a \cdot b' = a \cdot (b + b') = a$$

$$\{ a' \cdot b, a \cdot b \} \rightarrow a' \cdot b + a \cdot b = (a' + a) \cdot b = b$$

$$\{ a \cdot b \} \rightarrow a \cdot b$$

$$\{ a \cdot b, a \cdot b', a' \cdot b \} \rightarrow a \cdot b + a \cdot b' + a' \cdot b = a + b$$

$$\{ \} \rightarrow 0$$

$$\{ a' \cdot b', a \cdot b, a \cdot b', a' \cdot b \} \rightarrow 1$$

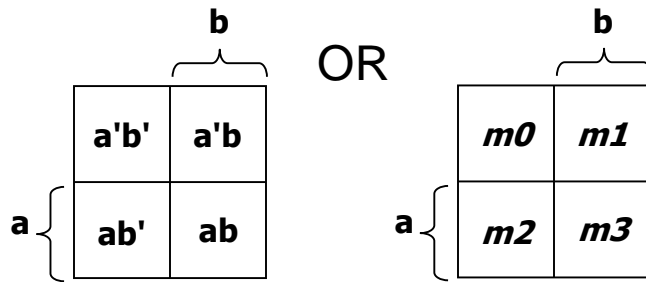
# 2-VARIABLE K-MAPS (1 / 4)

- Karnaugh-map (K-map) is an abstract form of Venn diagram, organised as **a matrix of squares**, where
  - Each square represents a **minterm**
  - Two adjacent squares represent minterms that **differ by exactly one literal**
- Let the 2 variables be a and b. The K-map can be drawn as...

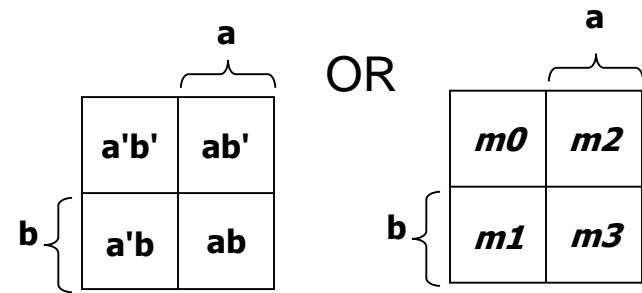
# 2-VARIABLE K-MAPS (2/4)

- Alternative **layouts** of a 2-variable (a, b) K-map:

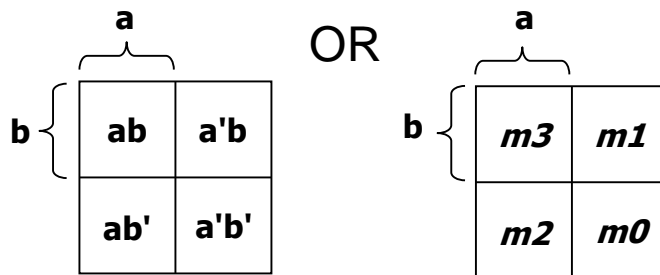
Alternative 1:



Alternative 2:



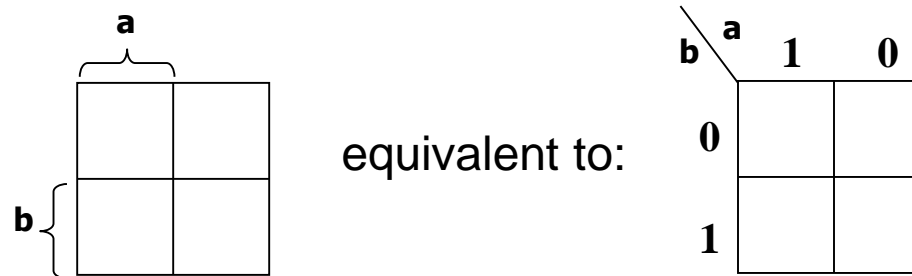
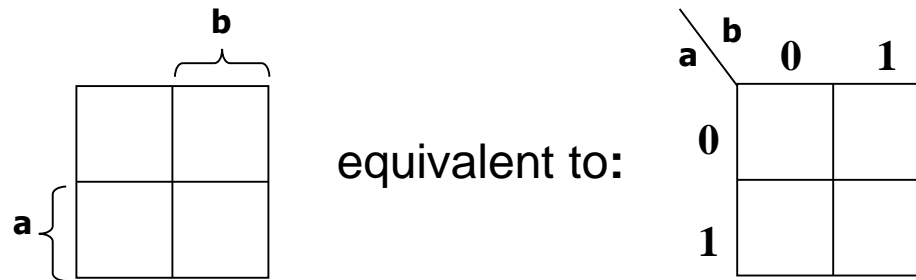
Alternative 3:



and others...

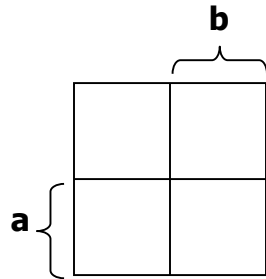
# 2-VARIABLE K-MAPS (3/4)

- Alternative **labelling** of a 2-variable (a, b) K-map:

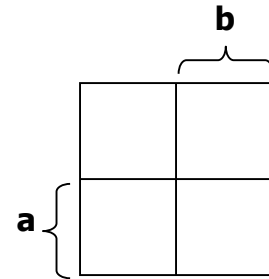


# 2-VARIABLE K-MAPS (4/4)

- The K-map for a function is filled by putting
  - A '1' in the square the corresponds to a minterm of the function
  - A '0' otherwise
- Example: Half adder.



$$C = a \cdot b$$



$$S = a \cdot b' + a' \cdot b$$



# 3-VARIABLE K-MAPS (1/2)

- As there are 8 minterms for 3 variables, so there are 8 squares in a 3-variable K-map.
- Example: Let the variables be a, b, c.

		b			
		bc			
a	0	$a'b'c'$	$a'b'c$	$a'bc$	$a'bc'$
	1	$ab'c'$	$ab'c$	$abc$	$abc'$

OR

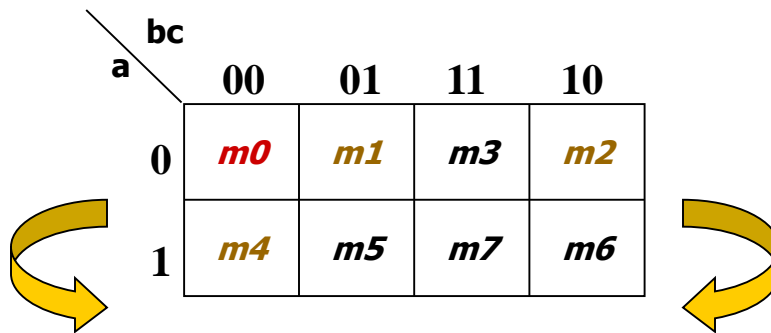
		b			
		bc			
a	0	$m0$	$m1$	$m3$	$m2$
	1	$m4$	$m5$	$m7$	$m6$

Above arrangement ensures that minterms of adjacent cells *differ by only ONE literal*.  
(Other arrangements which satisfy this criterion may also be used.)

*Note Gray code sequence*

# 3-VARIABLE K-MAPS (2/2)

- There is **wrap-around** in the K-map:
  - $a'b'c'$  ( $m0$ ) is adjacent to  $a'b'c'$  ( $m2$ )
  - $a'b'c'$  ( $m4$ ) is adjacent to  $a'b'c'$  ( $m6$ )



Each cell in a 3-variable K-map has 3 adjacent neighbours. In general, each cell in an  $n$ -variable K-map has  $n$  adjacent neighbours. For example,  $m0$  has 3 adjacent neighbours:  $m1$ ,  $m2$  and  $m4$ .

# QUICK REVIEW QUESTIONS (1)

5-1. The K-map of a 3-variable function  $F$  is shown below. What is the sum-of-minterms expression of  $F$ ?

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

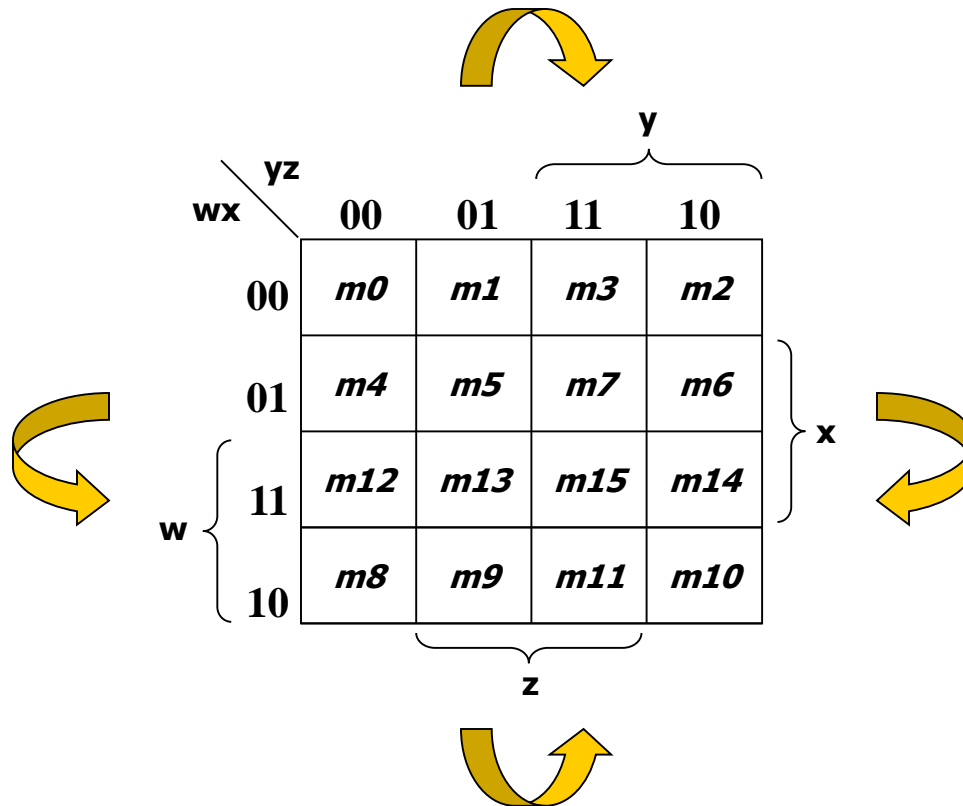
5-2. Draw the K-map for this function  $A$ :

$$A(x, y, z) = x \cdot y + y \cdot z' + x' \cdot y' \cdot z$$



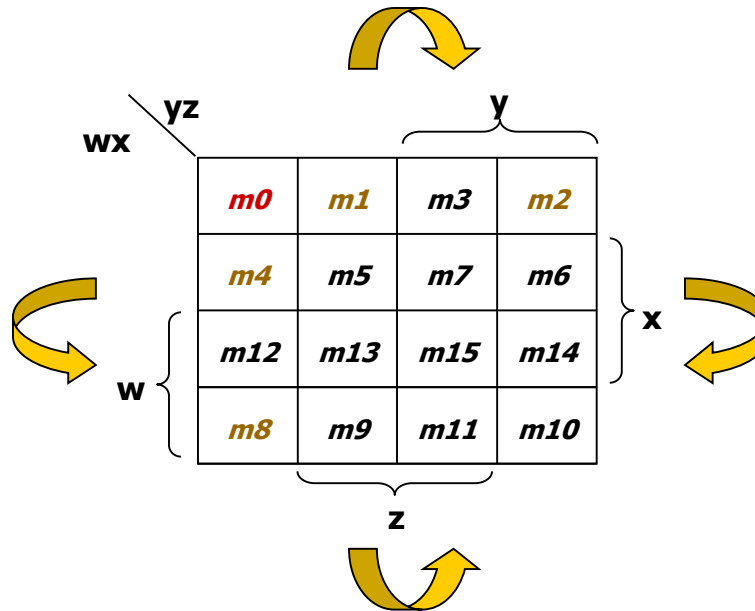
# 4-VARIABLE K-MAPS (1/2)

- There are 16 square cells in a 4-variable K-map.
- Example: Let the variables be  $w$ ,  $x$ ,  $y$ ,  $z$ .



# 4-VARIABLE K-MAPS (2/2)

- There are 2 wrap-arounds.
- Every cell has 4 neighbours.
  - Example: The cell corresponding to minterm  $m0$  has neighbours  $m1$ ,  $m2$ ,  $m4$  and  $m8$ .

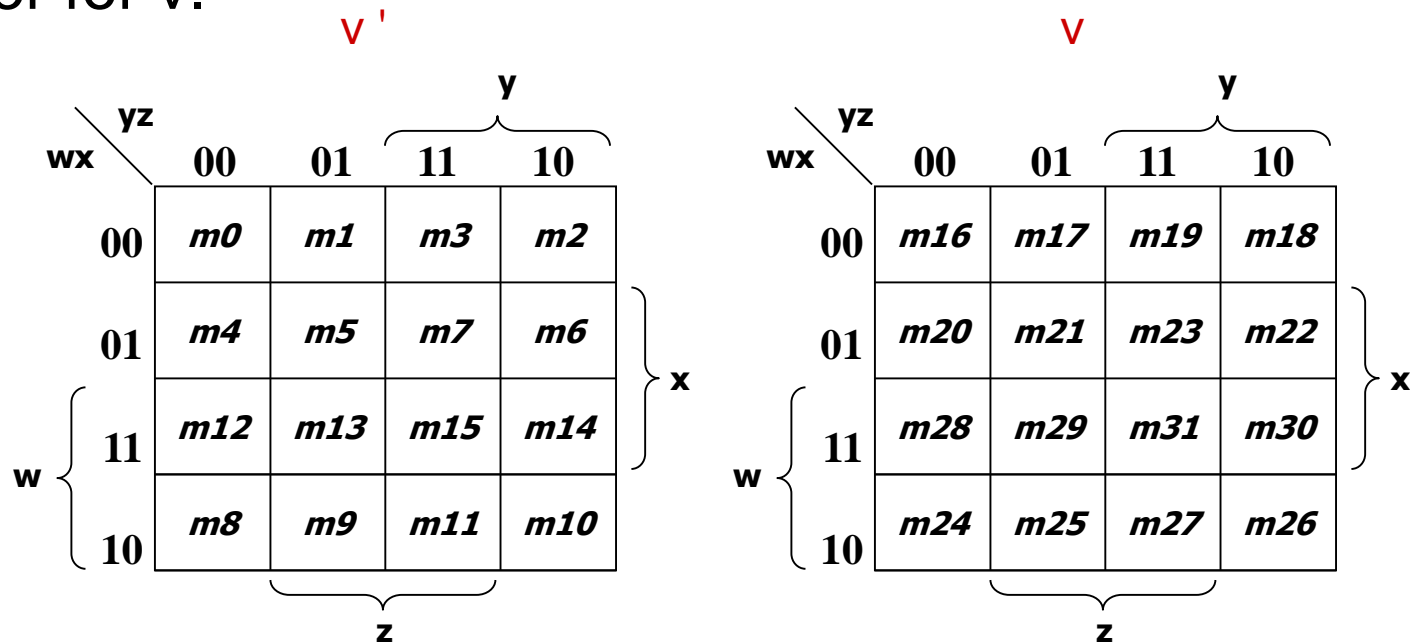


# 5-VARIABLE K-MAPS (1 / 2)

- K-maps of more than 4 variables are more difficult to use because the geometry (hypercube configurations) for combining adjacent squares becomes more involved.
- For 5 variables, e.g. v, w, x, y, z, we need  $2^5 = 32$  squares.
- Each square has 5 neighbours.

# 5-VARIABLE K-MAPS (2/2)

- Organised as two 4-variable K-maps. One for  $v'$  and the other for  $v$ .



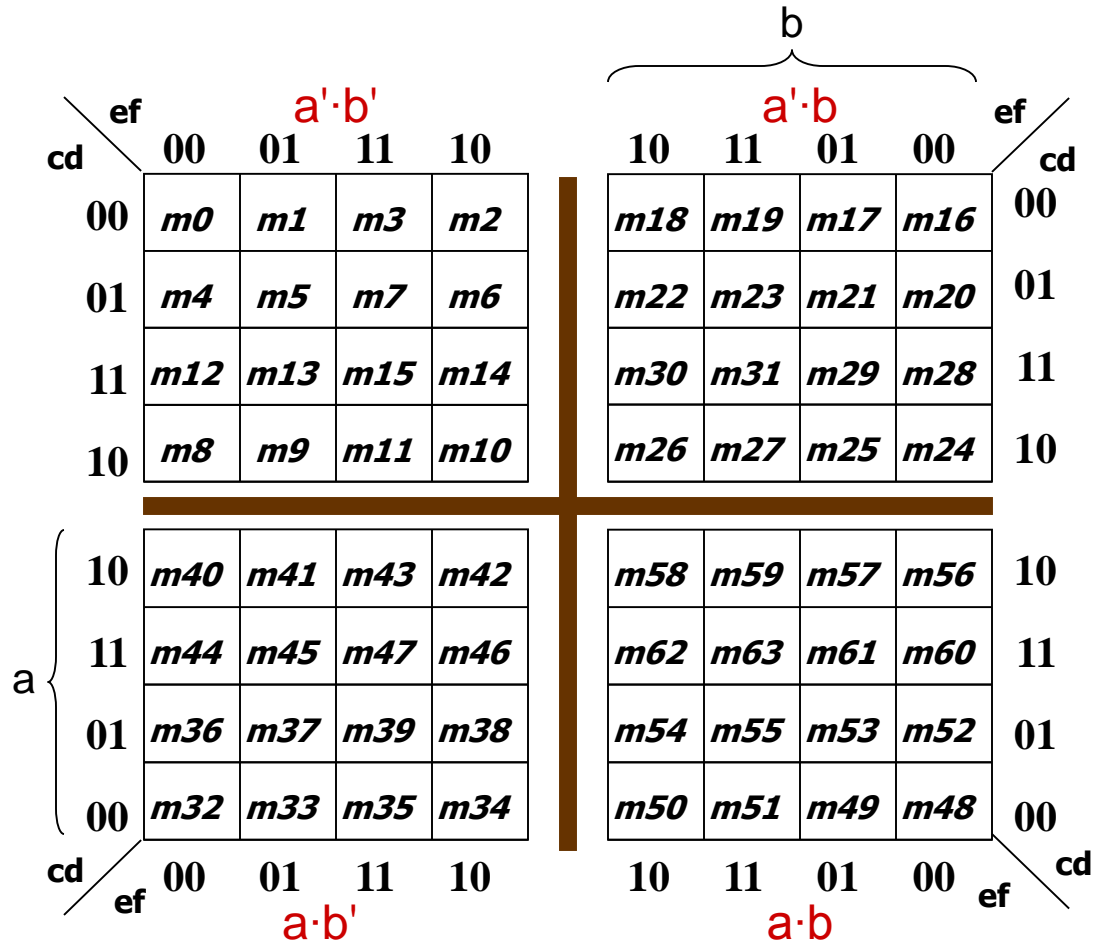
Corresponding squares of each map are adjacent.  
Can visualise this as *one 4-variable K-map* being on TOP of *the other 4-variable K-map*.

# LARGER K-MAPS (1 / 2)

- 6-variable K-map is pushing the limit of human's "pattern-recognition" capability.
- K-maps larger than 6 variables are practically unheard of!
- Normally, a 6-variable K-map is organised as four 4-variable K-maps, mirrored along two axes.



# LARGER K-MAPS (2/2)



Try stretch your recognition capability by finding simplest sum-of-products expression for  $\Sigma m(6,8,14,18,23,25,27,29,41,45,57,61)$ .

# HOW TO USE K-MAPS (1/7)

- Based on the **Unifying Theorem** (complement law):

$$\mathbf{A + A' = 1}$$

- In a K-map, each cell containing a '1' corresponds to a minterm of a given function  $F$ .
- Each valid grouping of adjacent cells containing '1' then corresponds to a **simpler product term** of  $F$ .
  - A group must have size in **powers of two**: 1, 2, 4, 8, ...
  - Grouping 2 adjacent cells eliminates 1 variable from the product term; grouping 4 cells eliminates 2 variables; grouping 8 cells eliminates 3 variables, and so on. In general, grouping  $2^n$  cells eliminates  $n$  variables.

# HOW TO USE K-MAPS (2/7)

- **Group as many cells as possible**
  - The larger the group, the fewer the number of literals in the resulting product term.
- **Select as few groups as possible to cover all the cells (minterms) of the function**
  - The fewer the groups, the fewer is the number of product terms in the simplified (minimal) SOP expression.

# HOW TO USE K-MAPS (3/7)

## ■ Example:

$$\begin{aligned} F(w,x,y,z) &= w' \cdot x \cdot y' \cdot z' + w' \cdot x \cdot y' \cdot z + w \cdot x' \cdot y \cdot z' \\ &\quad + w \cdot x' \cdot y \cdot z + w \cdot x \cdot y \cdot z' + w \cdot x \cdot y \cdot z \\ &= \Sigma m(4, 5, 10, 11, 14, 15) \end{aligned}$$

		y				
		00	01	11	10	
w	00					x
	01	1	1			
	11			1	1	
	10			1	1	

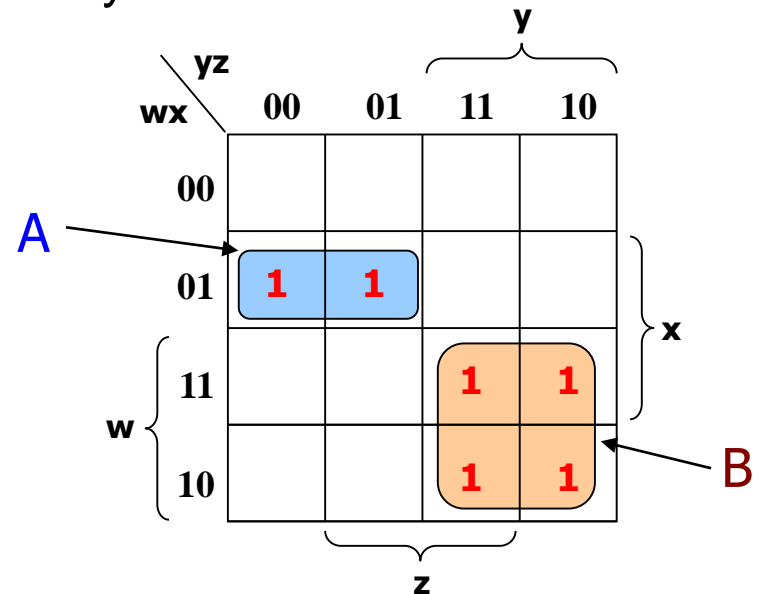
(cells with '0' are not shown for clarity)

# HOW TO USE K-MAPS (4/7)

- Each group of adjacent minterms corresponds to a possible product term of the given function.
- Here, there are 2 groups of minterms, A and B:

$$A = w'.x.y'.z' + w'.x.y'.z = w'.x.y'.(z' + z) = \mathbf{w'.x.y'}$$

$$\begin{aligned} B &= w.x'.y.z' + w.x'.y.z + w.x.y.z' + w.x.y.z \\ &= w.x'.y.(z' + z) + w.x.y.(z' + z) \\ &= w.x'.y + w.x.y \\ &= w.(x' + x).y \\ &= \mathbf{w.y} \end{aligned}$$



# HOW TO USE K-MAPS (5/7)

- Each product term that corresponds to a group,  $w' \cdot x \cdot y'$  and  $w \cdot y$ , represents the sum of minterms in that group.
- Boolean expression is therefore the sum of product terms (SOP) that represent all groups of the minterms of the function:

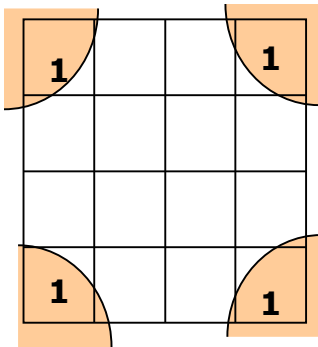
$$F(w,x,y,z) = \text{group A} + \text{group B} = w' \cdot x \cdot y' + w \cdot y$$

# HOW TO USE K-MAPS (6/7)

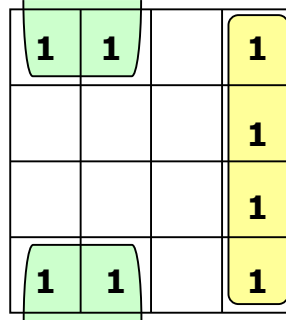
- The larger the group (the more minterms it contains), the fewer is the number of literals in the associated product term.
  - Recall that a group must have size in powers of two.
  - Example: For a 4-variable K-map with variables  $w, x, y, z$ 
    - 1 cell = 4 literals. Examples:  $w \cdot x \cdot y \cdot z$ ,  $w' \cdot x \cdot y' \cdot z$
    - 2 cells = 3 literals. Examples:  $w \cdot x \cdot y$ ,  $w \cdot y' \cdot z'$
    - 4 cells = 2 literals. Examples:  $w \cdot x$ ,  $x' \cdot y$
    - 8 cells = 1 literal. Examples:  $w$ ,  $y'$ ,  $z$
    - 16 cells = no literal (i.e. logical constant 1). Example: 1

# HOW TO USE K-MAPS (7/7)

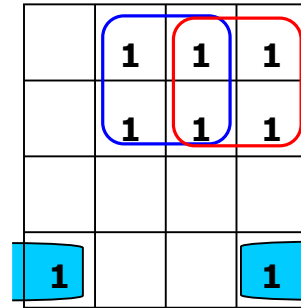
- Examples of valid and invalid groupings.



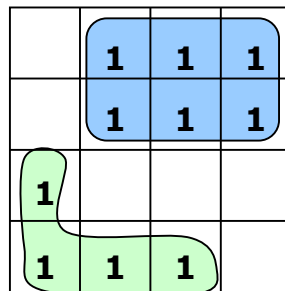
✓



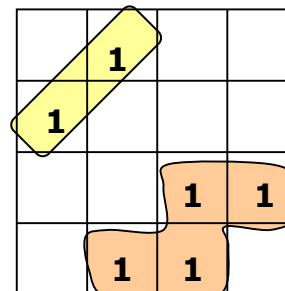
✓



✓



✗



✗



# CONVERTING TO MINTERMS FORM (1 / 2)

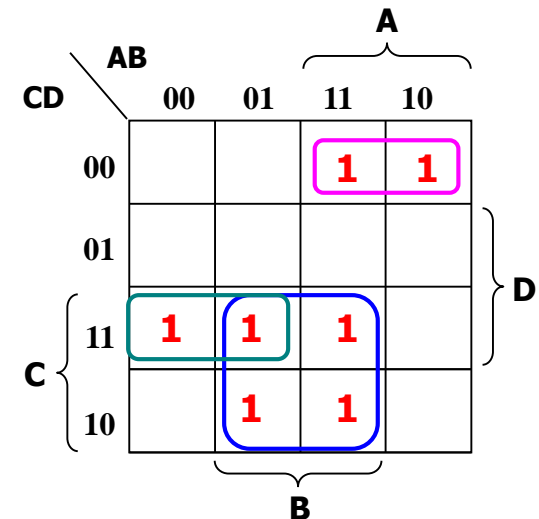
- The K-map of a function can be easily filled in when the function is given in sum-of-minterms form.
- What if it is not in sum-of-minterms form?
  - Convert it into sum-of-products (SOP) form
  - Expand the SOP expression into sum-of-minterms expression, or fill in the K-map directly based on the SOP expression.

# CONVERTING TO MINTERMS FORM (2/2)

## ■ Example:

$$\begin{aligned}
 F(A,B,C,D) &= A.(C+D)'.(B'+D') + C.(B+C'+A'.D) \\
 &= A.(C'.D').(B'+D') + B.C + C.C' + A'.C.D \\
 &= \underline{A.B'.C'.D'} + \underline{A.C'.D'} + \underline{B.C} + \underline{A'.C.D}
 \end{aligned}$$

$$\begin{aligned}
 &A.B'.C'.D' + \textcolor{red}{A.C'.D'} + B.C + A'.C.D \\
 &= A.B'.C'.D' + A.C'.D'.(B+B') + \textcolor{red}{B.C} + A'.C.D \\
 &= \underline{A.B'.C'.D'} + A.B.C'.D' + \underline{A.B'.C'.D'} + B.C.(A+A') \\
 &\quad + A'.C.D \\
 &= A.B'.C'.D' + A.B.C'.D' + A.B.C + A'.B.C + \\
 &\quad A'.C.D \\
 &= A.B'.C'.D' + A.B.C'.D' + A.B.C.(D+D') + \\
 &\quad A'.B.C.(D+D') + A'.C.D.(B+B') \\
 &= A.B'.C'.D' + A.B.C'.D' + A.B.C.D + A.B.C.D' + \\
 &\quad A'.B.C.D + A'.B.C.D' + A'.B'.C.D
 \end{aligned}$$

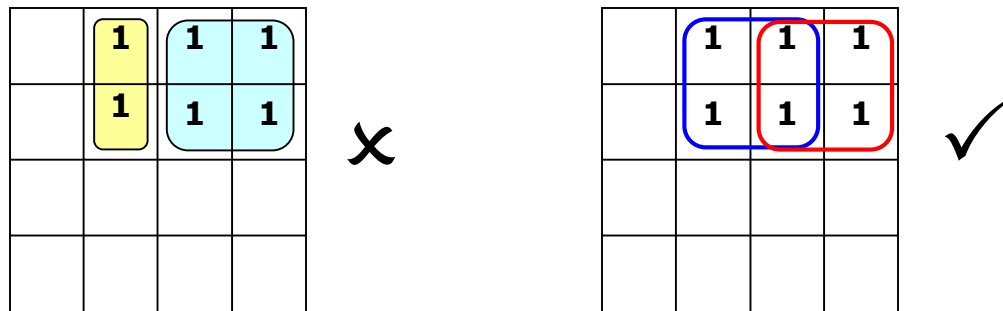


# PIs AND EPIs (1 / 3)

- To find the simplest (minimal) SOP expression from a K-map, you need to obtain:
  - Minimum number of literals per product term; and
  - Minimum number of product terms.
- Achieved through K-map using
  - *Bigger groupings* of minterms (**prime implicants**) where possible; and
  - *No redundant groupings* (look for **essential prime implicants**)
- **Implicant**: a product term that could be used to cover minterms of the function.

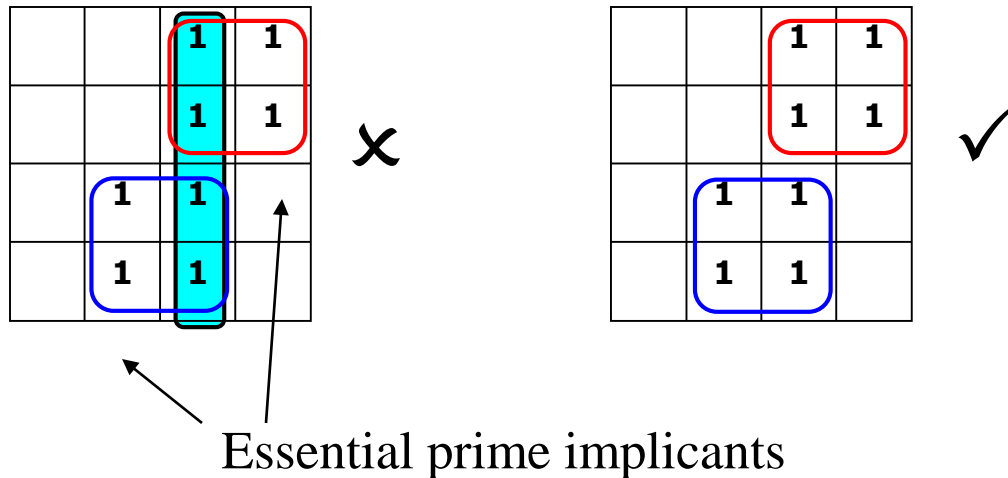
# PIs AND EPIs (2/3)

- **Prime implicant (PI)**: a product term obtained by combining the *maximum possible number of minterms* from *adjacent* squares in the map. (That is, it is the biggest grouping possible.)
- Always look for prime implicants in a K-map.



# PIs AND EPIs (3/3)

- No redundant groups:



- **Essential prime implicant (EPI):** a prime implicant that includes at least one minterm that is not covered by any other prime implicant.

# QUICK REVIEW QUESTIONS (2)

Identify the prime implicants and essential prime implicants of the two K-maps below.

		b			
		00	01	11	10
a	0	1	1	0	1
	1	0	1	0	0

Groupings:   
 - Group 'a' covers the first column (a=0 and a=1).   
 - Group 'b' covers the top row (b=00, 01, 11, 10).   
 - Group 'c' covers the bottom row (c=01, 11).

		A			
		00	01	11	10
CD	00	1	1		1
	01			1	1
	11	1	1		1
	10	1		1	1

Groupings:   
 - Group 'A' covers the top row (A=00, 01, 11, 10).   
 - Group 'B' covers the bottom row (B=00, 01, 11, 10).   
 - Group 'C' covers the first column (C=00, 01, 11, 10).   
 - Group 'D' covers the right column (D=11, 10).



# EXAMPLE ON FINDING MINIMAL SOP EXPRESSION (1/4)

## ■ Algorithm

1. Circle all prime implicants on the K-map.
2. Identify and select all essential prime implicants for the cover.
3. Select a minimum subset of the remaining prime implicants to complete the cover, that is, to cover those minterms not covered by the essential prime implicants.

# EXAMPLE ON FINDING MINIMAL SOP EXPRESSION (2/4)

## ■ Example #1:

$$F(A,B,C,D) = \Sigma m(2,3,4,5,7,8,10,13,15)$$

		A				
		AB				
CD		00	01	11	10	
	00		1		1	D
01		1	1			
11	1	1	1			
10	1			1		
		B				



# EXAMPLE ON FINDING MINIMAL SOP EXPRESSION (3/4)

AB		A			
		00	01	11	10
CD	00		1		1
	01		1	1	
	11	1	1	1	
	10	1			1

Groupings: **D** (C=01, 11), **B** (A=01, 11), **C** (C=11, 10)

AB		A			
		00	01	11	10
CD	00		1		1
	01		1	1	
	11	1	1	1	
	10	1			1

Groupings: **D** (C=01, 11), **B** (A=01, 11), **C** (C=11, 10)

← Essential prime implicants

AB		A			
		00	01	11	10
CD	00		1		1
	01		1	1	
	11	1	1	1	
	10	1			1

Groupings: **D** (C=01, 11), **B** (A=01, 11), **C** (C=11, 10)

← Answer

# EXAMPLE ON FINDING MINIMAL SOP EXPRESSION (4/4)

		A			
		AB		11	10
CD	00		1		1
	01		1	1	
C	11	1	1	1	
	10	1			1

**$F(A,B,C,D) =$**

# QUICK REVIEW QUESTIONS (3)

Find the minimal SOP expression for  $G(A,B,C,D)$ .

		A			
		AB		11	10
CD	00		1		
	01		1	1	1
C	11	1	1	1	
	10			1	

Brackets in the original image group the columns under 'A' (11, 10), the rows under 'D' (01, 11), and the columns under 'B' (01, 11).



# FINDING POS EXPRESSION (1/2)

- **Simplified POS expression** can be obtained by grouping the maxterms (i.e. 0s) of the given function.

- **Example:**

Given  $F = \Sigma m(0,1,2,3,5,7,8,9,10,11)$ , we first draw the K-map, then group the maxterms together:

A 4x4 Karnaugh map for variables A, B, C, and D. The columns are labeled AB (00, 01, 11, 10) and the rows are labeled CD (00, 01, 11, 10). The map contains 1s in all cells except for four cells containing 0s: (01, 00), (11, 00), (11, 01), and (11, 10). These four 0s are grouped by a blue rectangle labeled 'D'. Additionally, there are two green shaded regions: a horizontal group of two 0s in the top row (AB=01 and AB=11) labeled 'A', and a horizontal group of two 0s in the bottom row (AB=01 and AB=11) labeled 'B'. The labels A, B, and D are placed outside the map with brackets indicating their respective groupings.

		AB			
		00	01	11	10
CD	00	1	0	0	1
	01	1	1	0	1
	11	1	1	0	1
	10	1	0	0	1

# FINDING POS EXPRESSION (2/2)

K-map of  $F$

CD \ AB	A			
	00	01	11	10
00	1	0	0	1
01	1	1	0	1
11	1	1	0	1
10	1	0	0	1

Groupings:  $B$  (columns 01, 11),  $D$  (rows 01, 11),  $C$  (rows 11, 10)

K-map of  $F'$

CD \ AB	A			
	00	01	11	10
00	0	1	1	0
01	0	0	1	0
11	0	0	1	0
10	0	1	1	0

Groupings:  $B$  (columns 01, 11),  $D$  (rows 01, 11),  $C$  (rows 11, 10)

- This gives the SOP of  $F'$  to be  
$$F' = B \cdot D' + A \cdot B$$
- To get POS of  $F$ , we have

$F =$

# DON'T-CARE CONDITIONS (1/3)

- In certain problems, some outputs are not specified or are invalid.
- Hence, these outputs can be either '1' or '0'.
- They are called **don't-care conditions**, denoted by X (or sometimes, d).
- Example: An odd parity generator for BCD code which has 6 unused combinations.

A	B	C	D	P
0	0	0	0	1
0	0	0	1	0
0	0	1	0	0
0	0	1	1	1
0	1	0	0	0
0	1	0	1	1
0	1	1	0	1
0	1	1	1	0
1	0	0	0	0
1	0	0	1	1
1	0	1	0	X
1	0	1	1	X
1	1	0	0	X
1	1	0	1	X
1	1	1	0	X
1	1	1	1	X

# DON'T-CARE CONDITIONS (2/3)

- Don't-care conditions can be used to help simplify Boolean expression further in K-maps.
- They could be chosen to be either '1' or '0', depending on which choice results in a simpler expression.
- We usually use the notation  $\Sigma d$  to denote the set of don't-care minterms.
  - For example, the function  $P$  in the odd-parity generator for BCD code can be written as:

$$P = \Sigma m(0, 3, 5, 6, 9) + \Sigma d(10, 11, 12, 13, 14, 15)$$

# DON'T-CARE CONDITIONS (3/3)

## ■ Comparison

- Without don't-cares:

$$P = A' \cdot B' \cdot C' \cdot D' + A' \cdot B' \cdot C \cdot D + A' \cdot B \cdot C' \cdot D + A' \cdot B \cdot C \cdot D' + A \cdot B' \cdot C' \cdot D$$

A 4x4 Karnaugh map for variables A, B, C, and D. The vertical axis is labeled A with values 00, 01, 11, 10. The horizontal axis is labeled CD with values 00, 01, 11, 10. The map contains 1s at positions (A,B,C,D) = (0,0,0,0), (0,0,1,1), (0,1,0,1), (0,1,1,0), and (1,0,0,1). Brackets on the right group the 1s for B=0 and B=1. Brackets at the bottom group the 1s for D=0 and D=1.

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

- With don't-cares:

$$P = A' \cdot B' \cdot C' \cdot D' + B' \cdot C \cdot D + B \cdot C' \cdot D + B \cdot C \cdot D' + A \cdot D$$

A 4x4 Karnaugh map for variables A, B, C, and D, identical to the one above but with don't-care conditions (X) at (A,B,C,D) = (0,1,1,0), (1,0,0,1), (1,0,1,0), (1,1,0,1), and (1,1,1,0). The 1s are highlighted with colored boxes: a purple box for (0,0,0,0), a green box for (0,1,0,1), a yellow box for (0,0,1,1), an orange box for (0,1,1,0), and a blue box for (1,0,0,1). Brackets on the right group the 1s for B=0 and B=1. Brackets at the bottom group the 1s for D=0 and D=1.

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



# MORE EXAMPLES (1/6)

## ■ Example #2:

$$F(A,B,C,D) = A \cdot B \cdot C + B' \cdot C \cdot D' + A \cdot D + B' \cdot C' \cdot D'$$

		A			
		00	01	11	10
CD	00	1			1
	01			1	1
C	11			1	1
	10	1		1	1
		B			



Fill in the 1's.

# MORE EXAMPLES (2/6)

## ■ Example #2:

$$F(A,B,C,D) = A \cdot B \cdot C + B' \cdot C \cdot D' + A \cdot D + B' \cdot C' \cdot D'$$

		A			
		AB		11	10
CD	00	1			1
	01			1	1
C	11			1	1
	10	1		1	1
		B		D	



Find all PIs:

So the answer is:  $F(A,B,C,D) =$

# MORE EXAMPLES (3/6)

- Example #3 (with don't-cares):

$$F(A,B,C,D) = \Sigma m(2,8,10,15) + \Sigma d(0,1,3,7)$$

		A			
		AB			
CD		00	01	11	10
	00	X			1
	01	X			
C	11	X	X	1	
	10	1			1

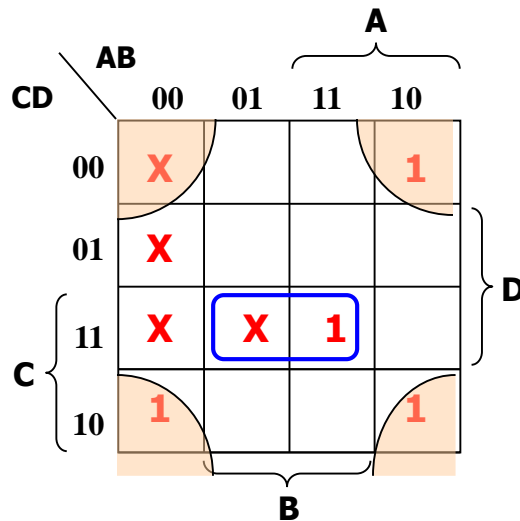


Fill in the 1's and X's.

# MORE EXAMPLES (4/6)

- Example #3 (with don't-cares):

$$F(A,B,C,D) = \Sigma m(2,8,10,15) + \Sigma d(0,1,3,7)$$



Do we need to have an additional term  $A' \cdot B'$  to cover the 2 remaining X's?

Answer:  $F(A,B,C,D) =$

# MORE EXAMPLES (5/6)

- Find the simplest POS expression for example #2:  
$$F(A,B,C,D) = A \cdot B \cdot C + B' \cdot C \cdot D' + A \cdot D + B' \cdot C' \cdot D'$$
- Draw the K-map of the complement of F, that is, F'.

		A			
		AB		11	10
CD	00		1	1	
	01	1	1		
C	11	1	1		
	10		1		

From K-map,

$F' =$

Using DeMorgan's theorem,

$F =$

# MORE EXAMPLES (6/6)

- Find the simplest POS expression for example #3:  
 $F(A,B,C,D) = \Sigma m(2,8,10,15) + \Sigma d(0,1,3,7)$
- Draw the K-map of the complement of F, that is, F'.  
 $F'(A,B,C,D) = \Sigma m(4,5,6,9,11,12,13,14) + \Sigma d(0,1,3,7)$

		A			
		00	01	11	10
CD	00	X	1	1	
	01	X	1	1	1
	11	X	X		1
	10		1	1	

Brackets in the original image indicate groupings: a vertical bracket on the right labeled 'D' groups rows 01 and 11; a horizontal bracket at the bottom labeled 'B' groups columns 01 and 11; a horizontal bracket at the top labeled 'A' groups columns 11 and 10; and a vertical bracket on the left labeled 'C' groups rows 11 and 10.

From K-map,

$F' =$

Using DeMorgan's theorem,

$F =$