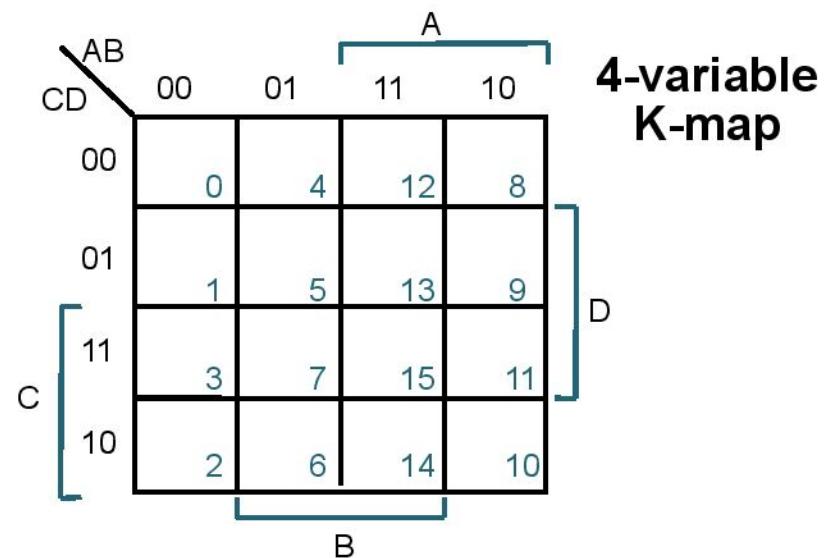
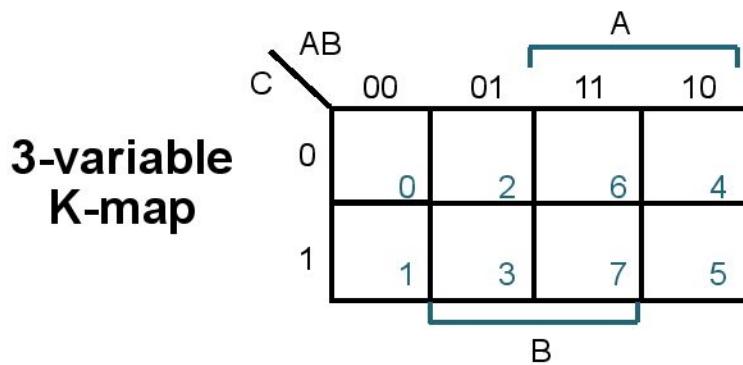
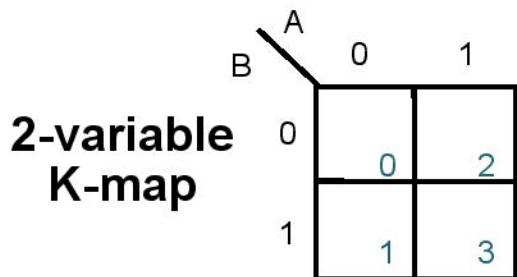


بهینه سازی با نقشه کارنو

Karnaugh Map

Karnaugh Map

- Method of graphically representing the truth table that helps visualize adjacencies



Karnaugh Map

- One cell (in K-map) = a row (in truth table)
- one cell = a minterm (or a maxterm)
- Multiple-cell areas = product terms (sum terms)

	A	0	1
B	0		
1	0	2	

	AB	00	01	11	10
C	0				
1	0	2	6	4	

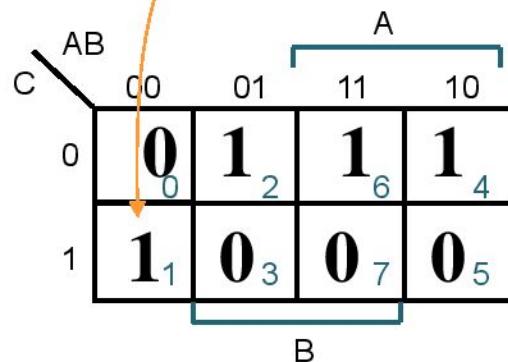
	B	0	1	2	3	7	5

	AB	00	01	11	10		
CD	00						
01	0	4	12	8			
11	1	5	13	9			
10	3	7	15	11			
	C	10	2	6	14	10	
	D						
	B						

Karnaugh Map

$$f_1(A,B,C) = m_1 + m_2 + m_4 + m_6$$

$$= A'B'C + A'BC' + AB'C' + ABC'$$



A	B	C		f_1
0	0	0	m_0	0
0	0	1	m_1	1
0	1	0	m_2	1
0	1	1	m_3	0
1	0	0	m_4	1
1	0	1	m_5	0
1	1	0	m_6	1
1	1	1	m_7	0

Karnaugh Map

- Numbering Scheme: 00, 01, 11, 10

Gray Code - only a single bit changes from code word to next code word.

	A	0	1
0		0	2
1		1	3

	AB	00	01	11	10
0		0	2	6	4
1	B	1	3	7	5

	AB	00	01	11	10
CD		0	4	12	8
00		1	5	13	9
01		3	7	15	11
11	C	2	6	14	10
10	D				

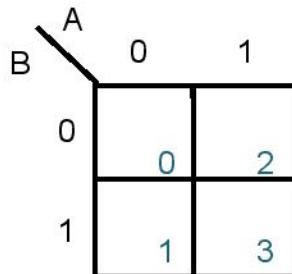
Two-Variable Map

- Any two adjacent cells: differ by ONLY one variable

- complemented in one cell and uncomplemented in the other.

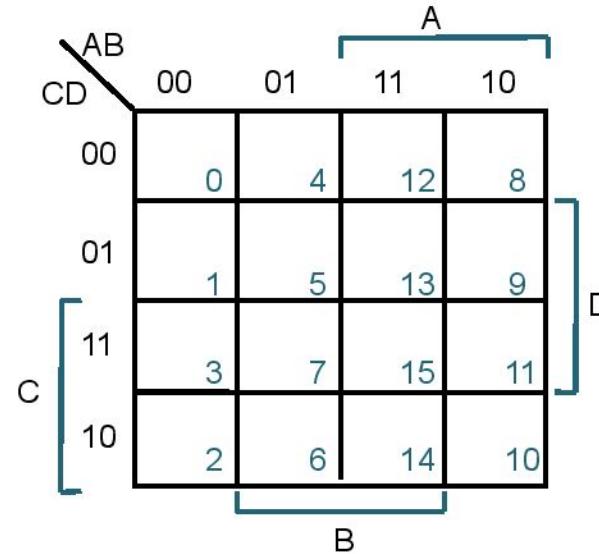
- **Example:**

- $m_0 (=A'B')$ is adjacent to $m_1 (=A'B)$ and $m_2 (=AB')$
- but NOT $m_3 (=AB)$



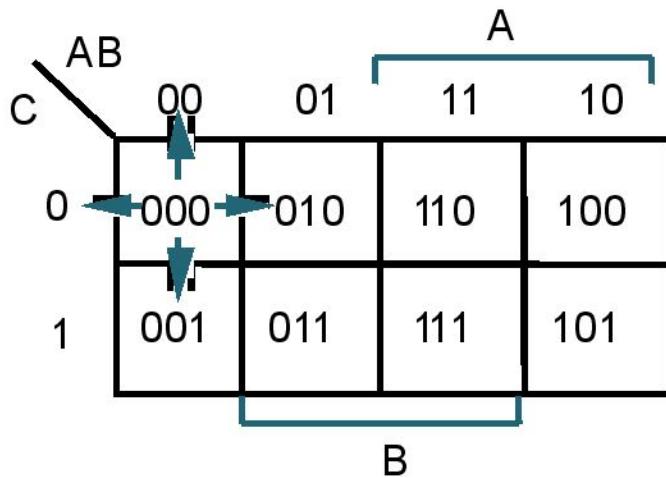
Two-Variable Map

- Any two adjacent cells: differ by ONLY one variable
 - complemented in one cell and uncomplemented in the other.
- Example:
 - m_5 is adjacent to m_7



Karnaugh Map

- Adjacencies in the K-Map



- Wrap from
 - first to last column
 - top row to bottom row

2-Variable Map -- Example

- $f(x_1, x_2) = x_1'x_2' + x_1'x_2 + x_1x_2'$
 $= m_0 + m_1 + m_2$
- Grouping (ORing) of 1s allows simplification

	x_1	
	x_2	
0	0	2
1	1	1

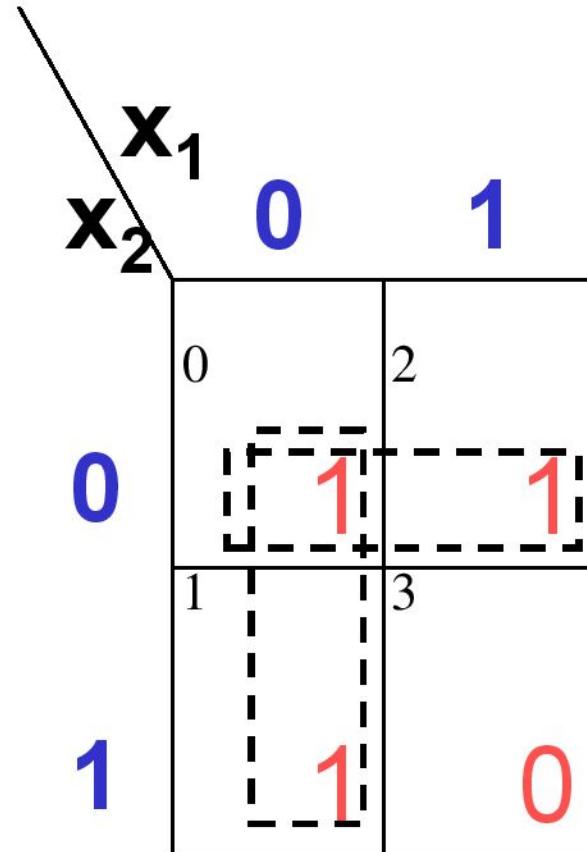
	x_1	
	x_2	
0	0	1
1	1	0

2-Variable Map -- Example

- $f(x_1, x_2) = x_1'x_2' + x_1'x_2 + x_1x_2'$
 $= m_0 + m_1 + m_2$

- Grouping (ORing) of 1s allows simplification
- What (simpler) function is represented by each dashed rectangle?
 - $m_0 + m_1 = x_1'x_2' + x_1'x_2 = x_1'(x_2' + x_2) = x_1'$
 - $m_0 + m_2 = x_1'x_2' + x_1x_2' = x_2'(x_1' + x_1) = x_2'$

- Note: m_0 covered twice



Minimization as SOP using K-map

- Enter 1s in the K-map for each product term in the function
- Group *adjacent* K-map cells containing 1s to obtain a product with fewer variables.
 - Groups must be in power of 2 (2, 4, 8, ...)
- Handle “boundary wrap”
- Minimum number of groups
- Maximal groups
- Answer may not be unique

Minimization as SOP

	A 0	1
B 0	0	1
1	0	1

A asserted, unchanged
B varies

$$F = ?$$

Minimization as SOP

	A 0	1
B 0	0	1
1	0	1

A asserted, unchanged
B varies

F = ?

B complemented, unchanged
A varies

	A 0	1
B 0	1	1
1	0	0

G = ?

Minimization as SOP

A	0	1
0	0	1
1	0	1

A asserted, unchanged
B varies

$$F = ?$$

A	0	1
0	1	1
1	0	0

B complemented, unchanged
A varies

$$G = ?$$

Cin	A	B	00	01	11	10
0	0	0	0	0	1	0
1	0	1	0	1	1	1

$$C_{out} = ?$$

Minimization as SOP

	A	0	1
	B	0	1
	1	0	1

A asserted, unchanged
B varies

$$F = ?$$

	A	0	1
	B	0	1
	1	0	0

B complemented, unchanged
A varies

$$G = ?$$

		A			
		00	01	11	
		0	0	1	0
		1	0	1	1
			B		

$$C_{out} = ?$$

		A			
		00	01	11	
		0	0	1	1
		1	0	0	1
			B		

$$F(A,B,C) = ?$$

Minimization as SOP

	A	0	1
	B	0	1
	1	0	1

A asserted, unchanged
B varies

$$F = A$$

	A	0	1
	B	0	1
	1	0	0

B complemented, unchanged
A varies

$$G = B'$$

		A			
		00	01	11	
		0	0	1	0
		1	0	1	1
Cin	AB	00	01	11	10
		0	0	1	0
		1	0	1	1

$$C_{out} = AB + BC_{in} + AC_{in}$$

		A			
		00	01	11	
		0	0	1	1
		1	0	0	1
C	AB	00	01	11	10
		0	0	1	1
		1	0	0	1

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

More Examples

AB		A	
C	00	01	11
0	1	0	0
1	0	0	1

B

$$F(A,B,C) = \Sigma m(0,4,5,7)$$

$$F =$$

AB		A	
C	00	01	11
0	0	1	1
1	1	1	0

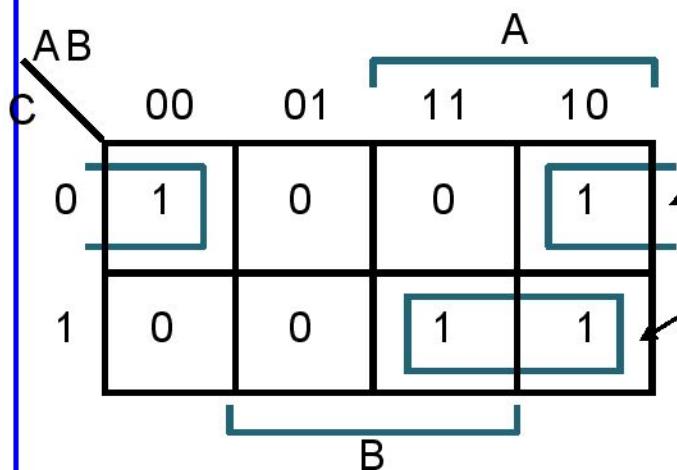
B

F' : simply replace 1's with 0's and vice versa

$$F'(A,B,C) = \Sigma m(1,2,3,6)$$

$$F' =$$

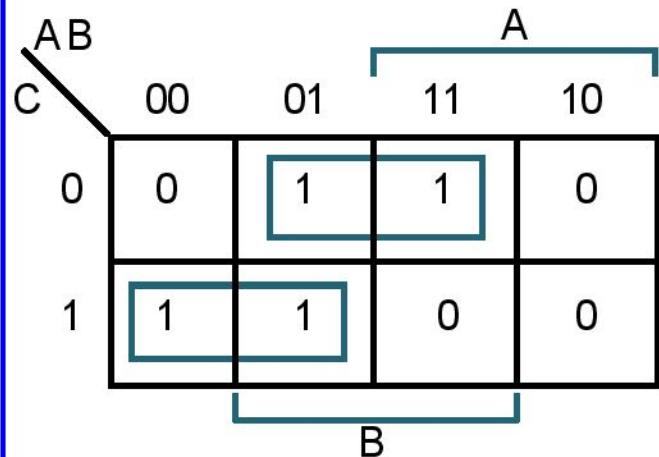
More Examples



Why not group m₄ and m₅?

$$F(A,B,C) = \Sigma m(0,4,5,7)$$

$$F = B' C' + A C$$



F' simply replaces 1's with 0's and vice versa

$$F'(A,B,C) = \Sigma m(1,2,3,6)$$

$$F' = B C' + A' C$$

Simplification

- To simplify a Boolean function:
 - Enter minterms of the function into the map
 - Group terms
- Example:
 - $f(a,b,c) = bc' + abc + ab'$

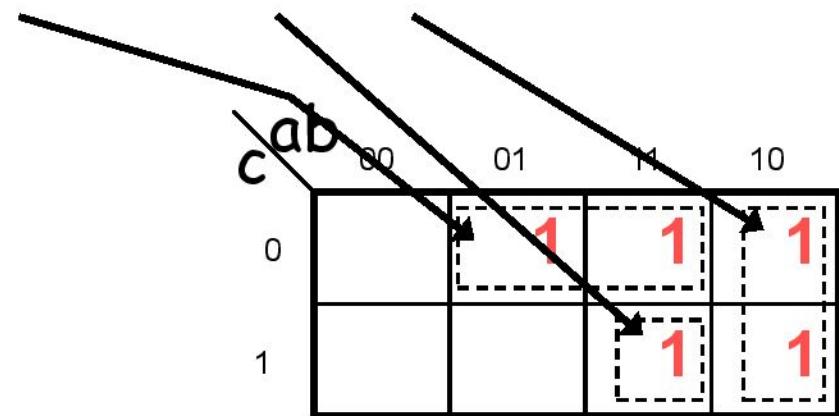
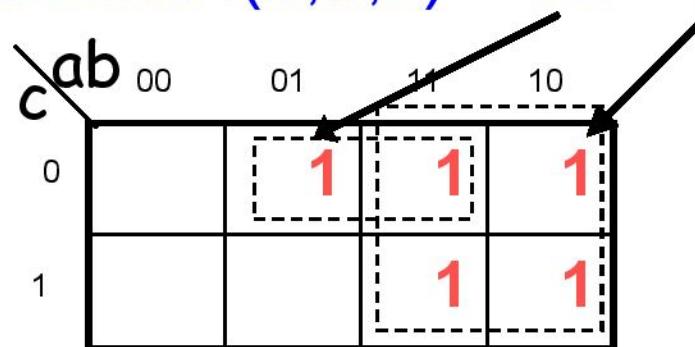
Simplification

- Example:

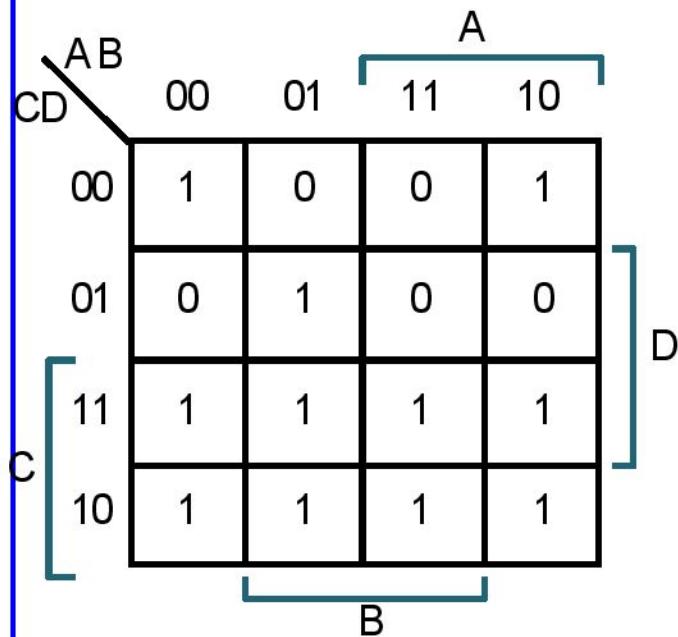


$$f(a,b,c) = bc' + abc + ab'$$

➤ Result: $f(a,b,c) = bc' + a$



4-Variable Map



$$F(A,B,C,D) = \Sigma m(0,2,3,5,6,7,8,10,11,14,15)$$

$F =$

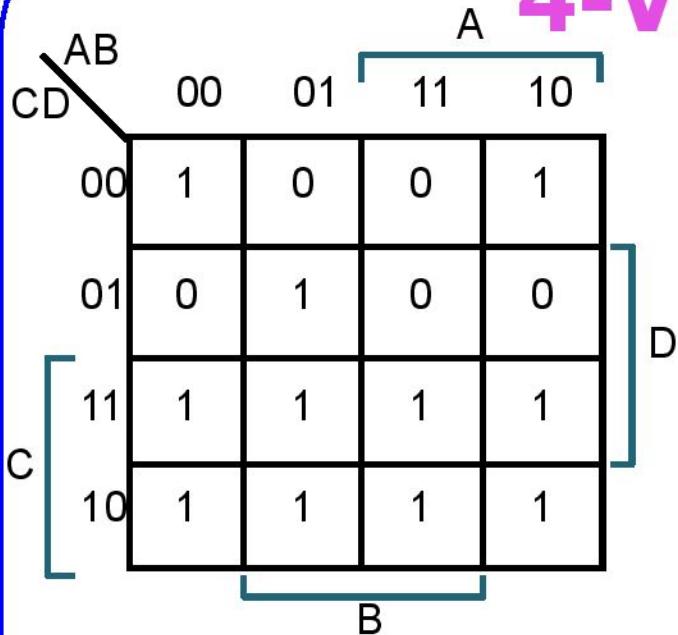
A minterm map for 4 variables (AB, CD). The columns are labeled AB (00, 01, 11, 10) and the rows are labeled CD (00, 01, 11, 10). The map shows the following minterms:

AB\CD	00	01	11	10
00	m_0	m_4	m_{12}	m_8
01	m_1	m_5	m_{13}	m_9
11	m_3	m_7	m_{15}	m_{11}
10	m_2	m_6	m_{14}	m_{10}

Four-variable Map Simplification

- One square represents a minterm of 4 literals.
- A rectangle of 2 adjacent squares represents a product term of 3 literals.
- A rectangle of 4 squares represents a product term of 2 literals.
- A rectangle of 8 squares represents a product term of 1 literal.
- A rectangle of 16 squares produces a function that is equal to logic 1.

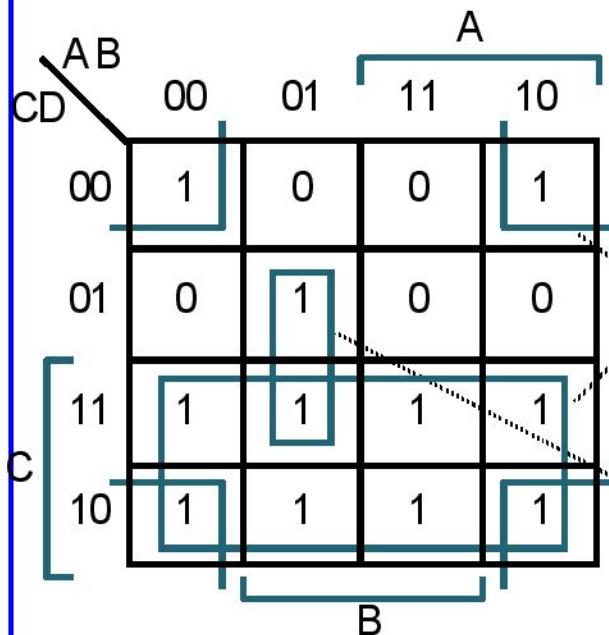
4-Variable Map



$$F(A,B,C,D) = \Sigma m(0,2,3,5,6,7,8,10,11,14,15)$$

- Find the **smallest number** of the **largest possible subcubes** that cover the ON-set

4-Variable Map



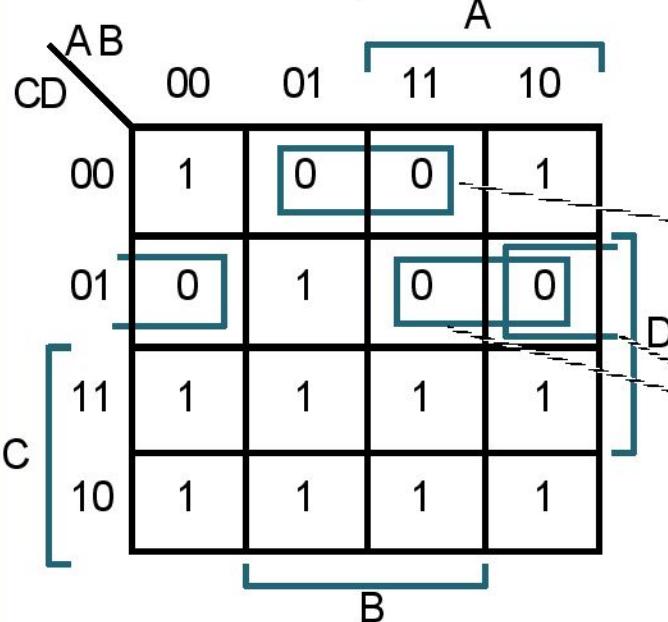
$$F(A,B,C,D) = \sum m(0,2,3,5,6,7,8,10,11,14,15)$$

$$F = C + A' B D + B' D'$$

➤ Find the **smallest number** of the **largest possible subcubes** that cover the ON-set

Simplify for POS

K-map Method: Circling Zeros to get product of sums form



$$F = (B' + C + D)(A' + C + D')(B + C + D')$$

Replace F by F' , 0's become 1's and vice versa

$$F' = B C' D' + A C' D + B' C' D$$

$$(F')' = (B C' D' + A C' D + B' C' D)'$$

$$F = (B' + C + D)(A' + C + D')(B + C + D')$$

Don't Cares

Don't Cares can be treated as 1's or 0's if it is advantageous to do so

AB		00	01	11	10
CD		0	0	X	0
C		1	1	X	1
10	B	1	1	0	0
	D	0	X	0	0

$$F(A,B,C,D) = \sum m(1,3,5,7,9) + \sum d(6,12,13)$$

$$F = A'D + B'C'D \quad \text{w/o don't cares}$$

$$F = C'D + A'D \quad \text{w/ don't cares}$$

AB		00	01	11	10
CD		0	0	X	0
C		1	1	X	1
10	B	1	1	0	0
	D	0	X	0	0

In Product of Sums form : $F = D(A' + C')$

Same answer as above,
but fewer literals

Don't Cares

Don't Cares can be treated as 1's or 0's if it is advantageous to do so

AB		00	01	11	10	
CD		0	0	X	0	
C		01	1	1	X	1
	D	11	1	1	0	0
	B	10	0	X	0	0

$$F(A,B,C,D) = \sum m(1,3,5,7,9) + \sum d(6,12,13)$$

$$F = A'D + B'C'D \quad \text{w/o don't cares}$$

$$F = C'D + A'D \quad \text{w/ don't cares}$$

AB		00	01	11	10	
CD		0	0	X	0	
C		01	1	1	X	1
	D	11	1	1	0	0
	B	10	0	X	0	0

In Product of Sums form : $F = D(A' + C')$

Same answer as above,
but fewer literals

Order Dependency

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

Diagram illustrating a 4x4 truth table with inputs AB (row) and CD (column). The output is labeled A. The table shows the following values:

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

The output A is 1 for rows 01, 11, and 10, and 0 for rows 00 and 11.

Order Dependency

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

Diagram illustrating a dependency matrix with inputs AB (row) and CD (column), and outputs A and B. The matrix shows values 0 or 1 at each intersection. Red boxes highlight specific dependencies: (00, 00) is 1, (01, 01) is 1, (11, 01) is 1, (11, 11) is 1, and (10, 01) is 1. Brackets indicate grouping: A = {11, 10}, B = {01, 10}, C = {00, 01, 11}, D = {00, 01, 11}.

Definition of Terms

- **Implicant:**
 - Single element of the ON-set or any group of elements that can be combined together
- **Prime Implicant (PI) (maximal implicant):**
 - Implicant that if gets larger (covering twice), covers 0s.
- **Distinguished 1-cell:**
 - An input combination that is covered by only one PI.
- **Essential Prime Implicant (EPI):**
 - A PI that covers one or more distinguished 1-cells.

Implicant, PI and EPI

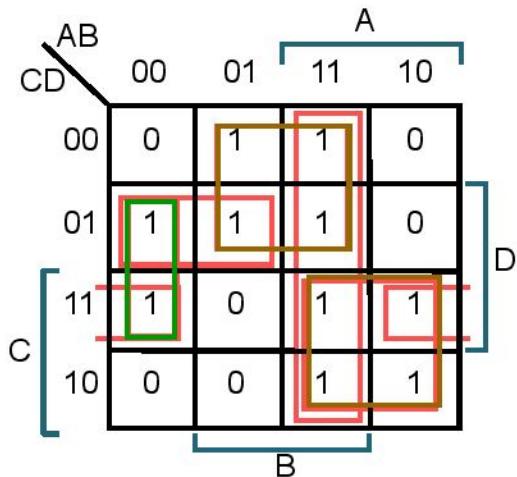
AB		A			
CD		00	01	11	10
C	00	0	1	1	0
	01	1	1	1	0
	11	1	0	1	1
	10	0	0	1	1
B		D			

6 Prime Implicants:

$A' B' D$, $B C'$, $A C$, $A' C' D$, $A B$, $B' C D$

Essential

Implicant, PI and EPI



6 Prime Implicants:

$A' B' D, B' C', A C, A' C' D, A B, B' C D$

Essential

Implicant, PI and EPI

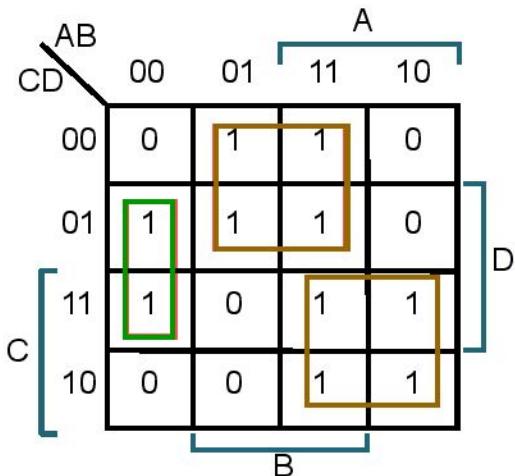
AB				A
		00	01	
CD	00	0	1	1
	01	1	1	0
	11	1	0	1
	10	0	0	1
B		D		

6 Prime Implicants:

$A' B' D$, $B C'$, $A C$, $A' C' D$, $A B$, $B' C D$

Essential

Implicant, PI and EPI



6 Prime Implicants:

$A' B' D$, $B C'$, $A C$, $A' C' D$, $A B$, $B' C D$

Essential

Minimum cover =
First: cover EPIs
Then: minimum number of PIs

$$= B C' + A C + A' B' D$$

Implicant, PI and EPI

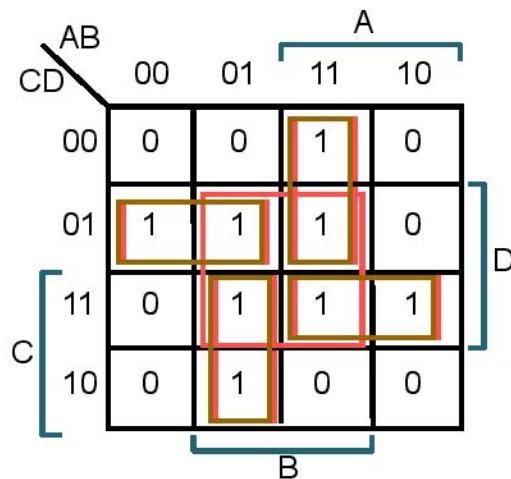
		AB		A	
		00	01	11	10
CD	00	0	0	1	0
	01	1	1	1	0
	11	0	1	1	1
	10	0	1	0	0

5 Prime Implicants:

$B D, A B C', A C D, A' B C, A' C' D$

essential

Implicant, PI and EPI



5 Prime Implicants:

B D, A B C', A C D, A' B C, A' C' D

essential

Minimum cover =

First: cover EPIs

Then: minimum number of PIs

$$= A B C' + A C D + A' B C + A' C' D$$

More Examples

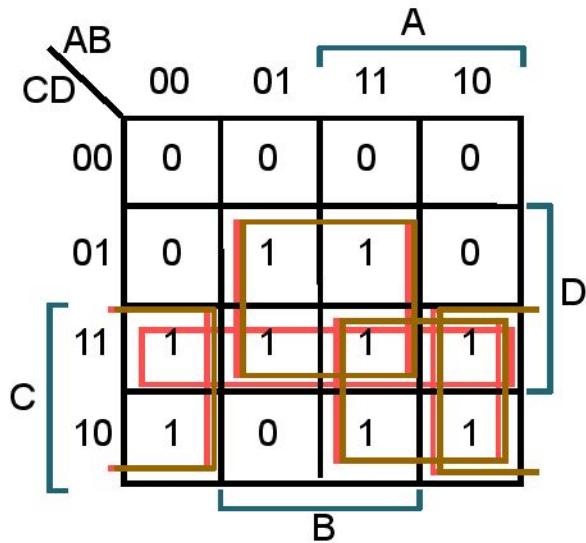
		AB		A	
		00	01		
CD	00	0	0	0	0
	01	0	1	1	0
	11	1	1	1	1
	10	1	0	1	1
C		D		B	

Prime Implicants:

B D, C D, A C, B' C

essential

More Examples



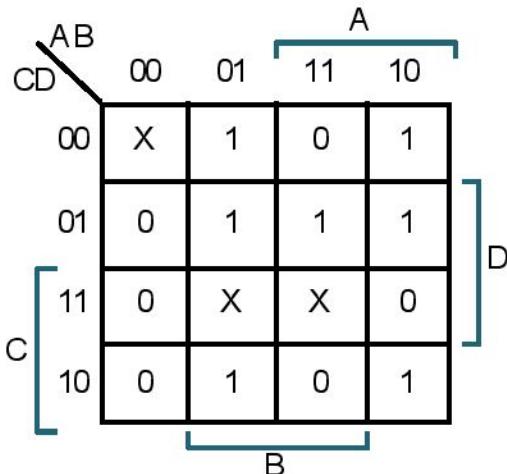
Prime Implicants:

B D, C D, A C, B' C
essential

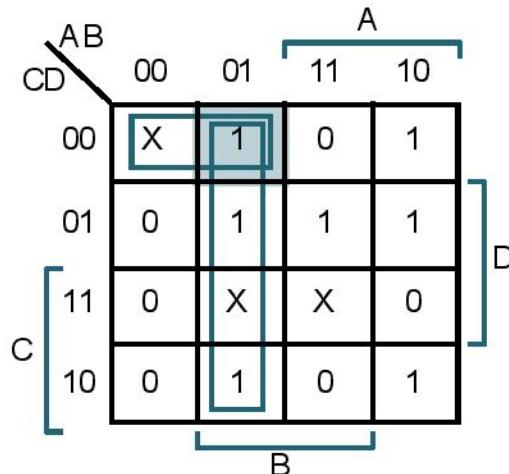
$$= BD + AC + B' C$$

Example

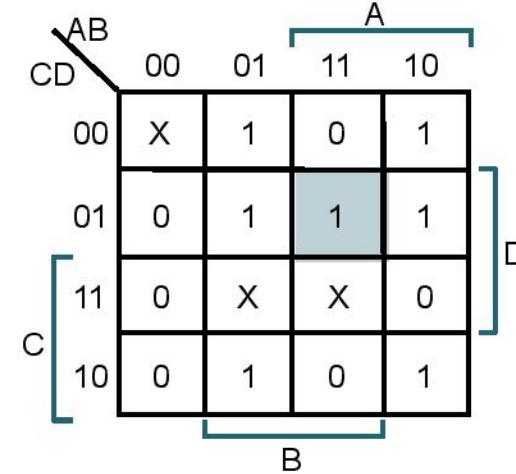
$$\text{Example: } f(A,B,C,D) = m(4,5,6,8,9,10,13) + d(0,7,15)$$



Initial K-map



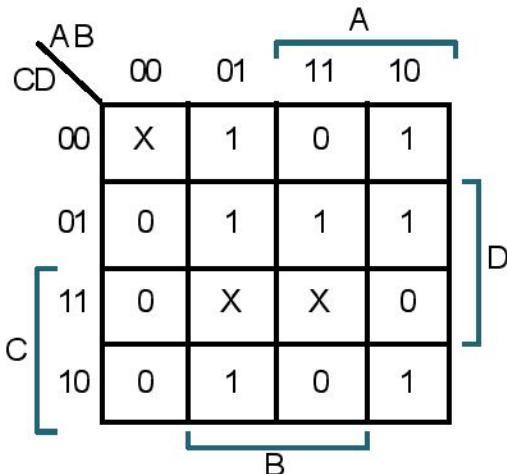
Primes around
 $A' B C' D'$



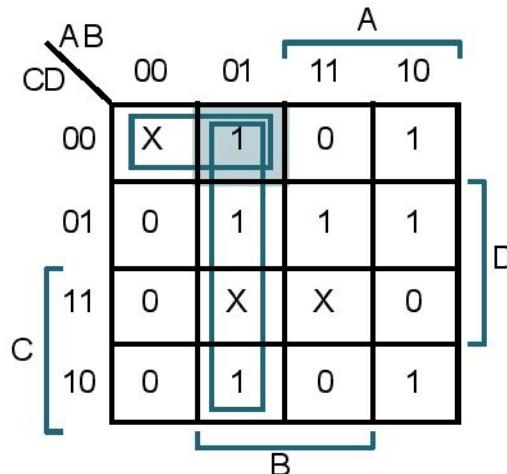
Primes around
 $A B C' D'$

Example

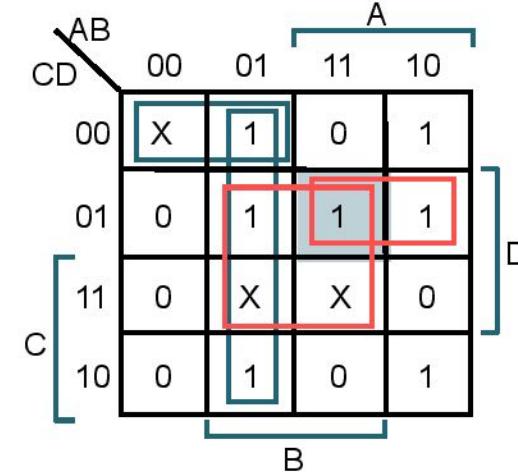
$$\text{Example: } f(A,B,C,D) = m(4,5,6,8,9,10,13) + d(0,7,15)$$



Initial K-map



Primes around
 $A' B C' D'$



Primes around
 $A B C' D$

Example: Continued

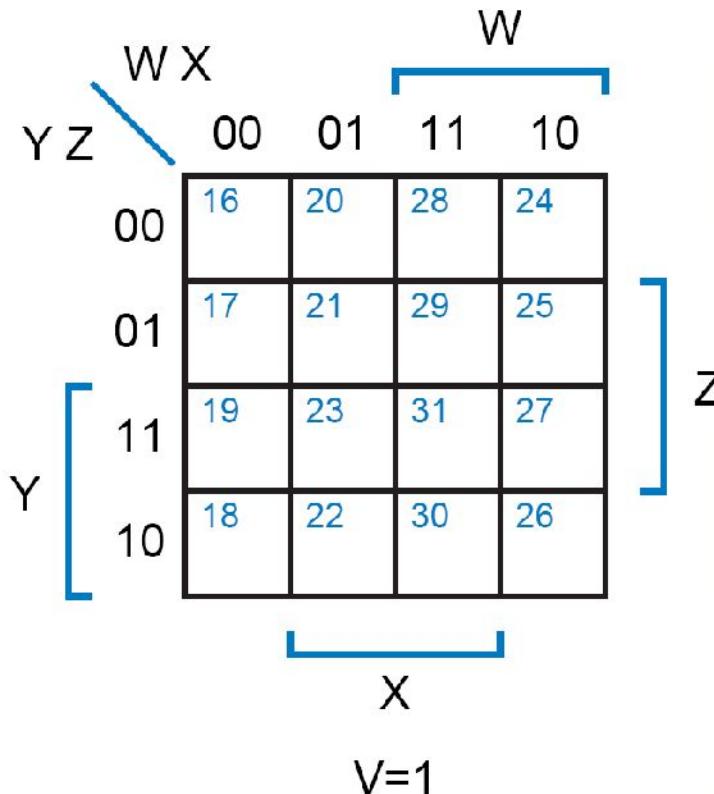
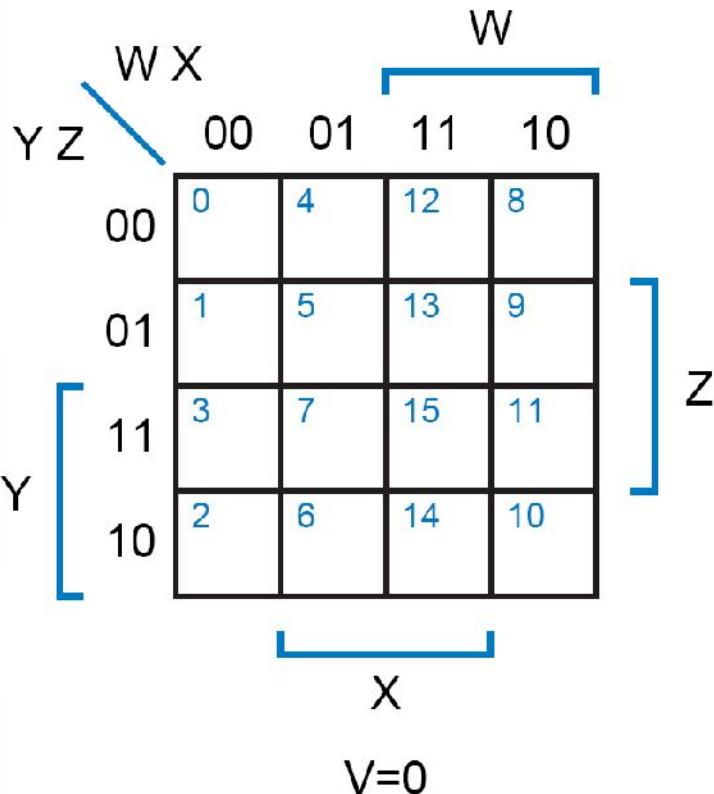
		AB		A	
		00	01	11	10
CD		00	X	1	0
C	01	0	1	1	1
	11	0	X	X	0
	10	0	1	0	1
	B				
D					

Primes around
A B' C' D'

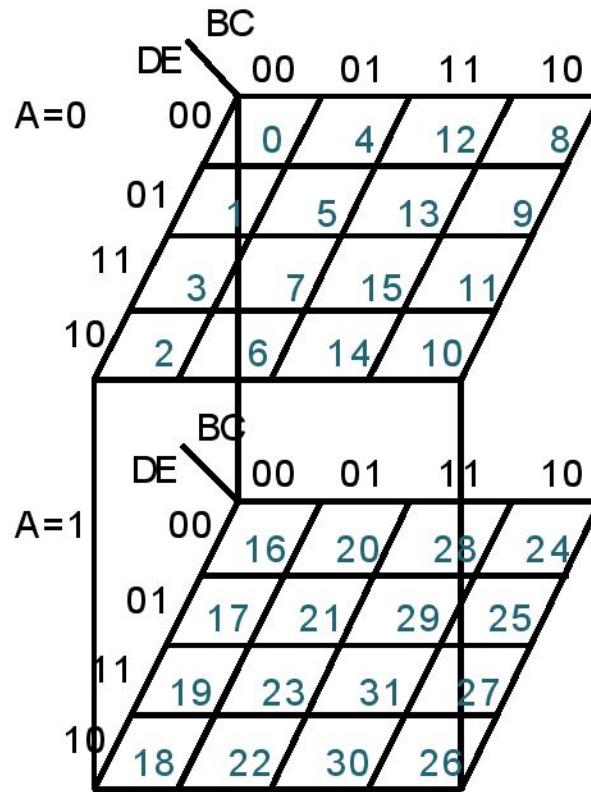
		AB		A	
		00	01	11	10
CD		00	X	1	0
C	01	0	1	1	1
	11	0	X	X	0
	10	0	1	0	1
	B				
D					

Essential Primes
with Min Cover
(each element covered once)

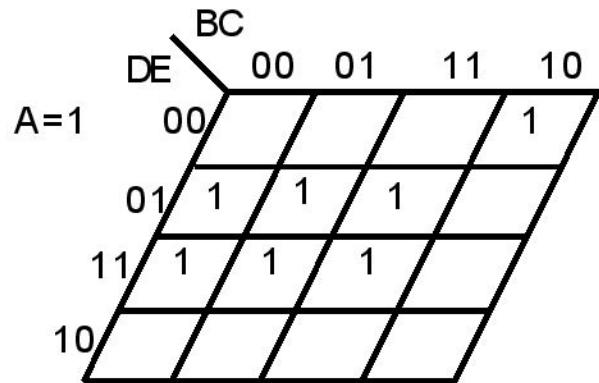
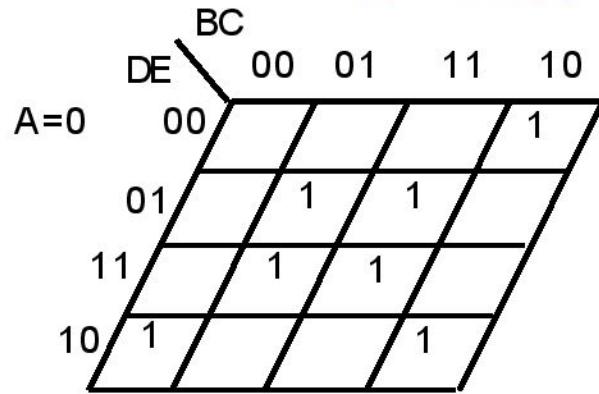
5-Variable K-Map



5-Variable K-Map

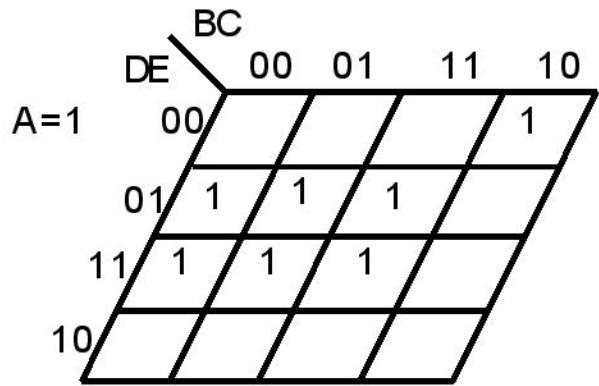
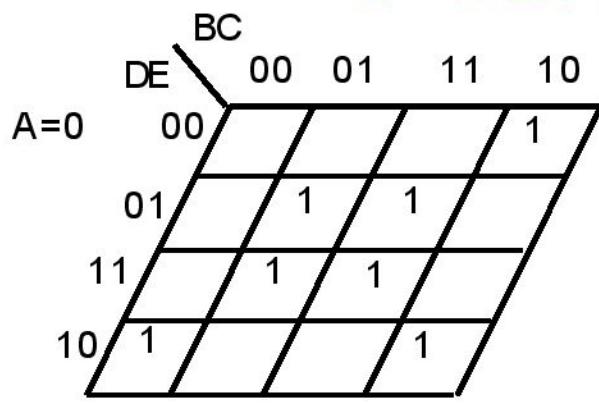


5-Variable K-Map

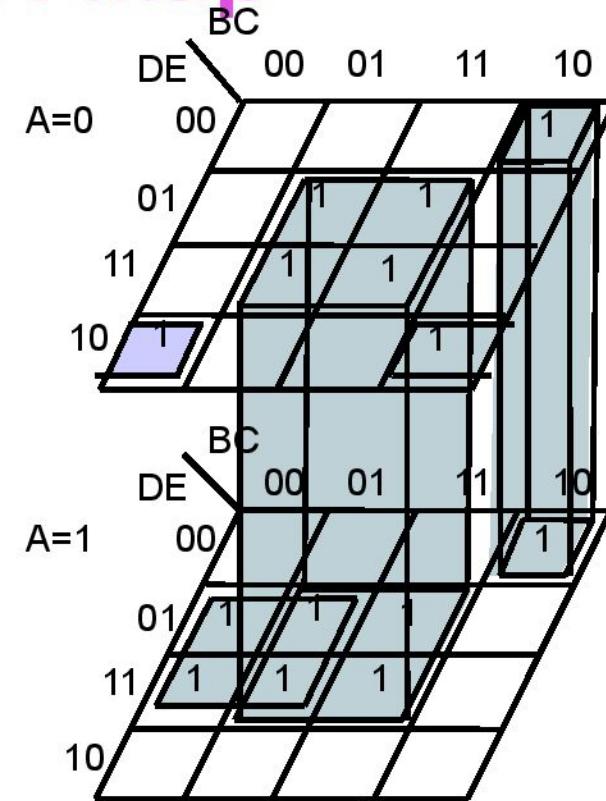


$$f(A,B,C,D,E) = \Sigma m(2,5,7,8,10, 13,15,17,19,21,23,24,29, 31)$$

5-Variable K-Map

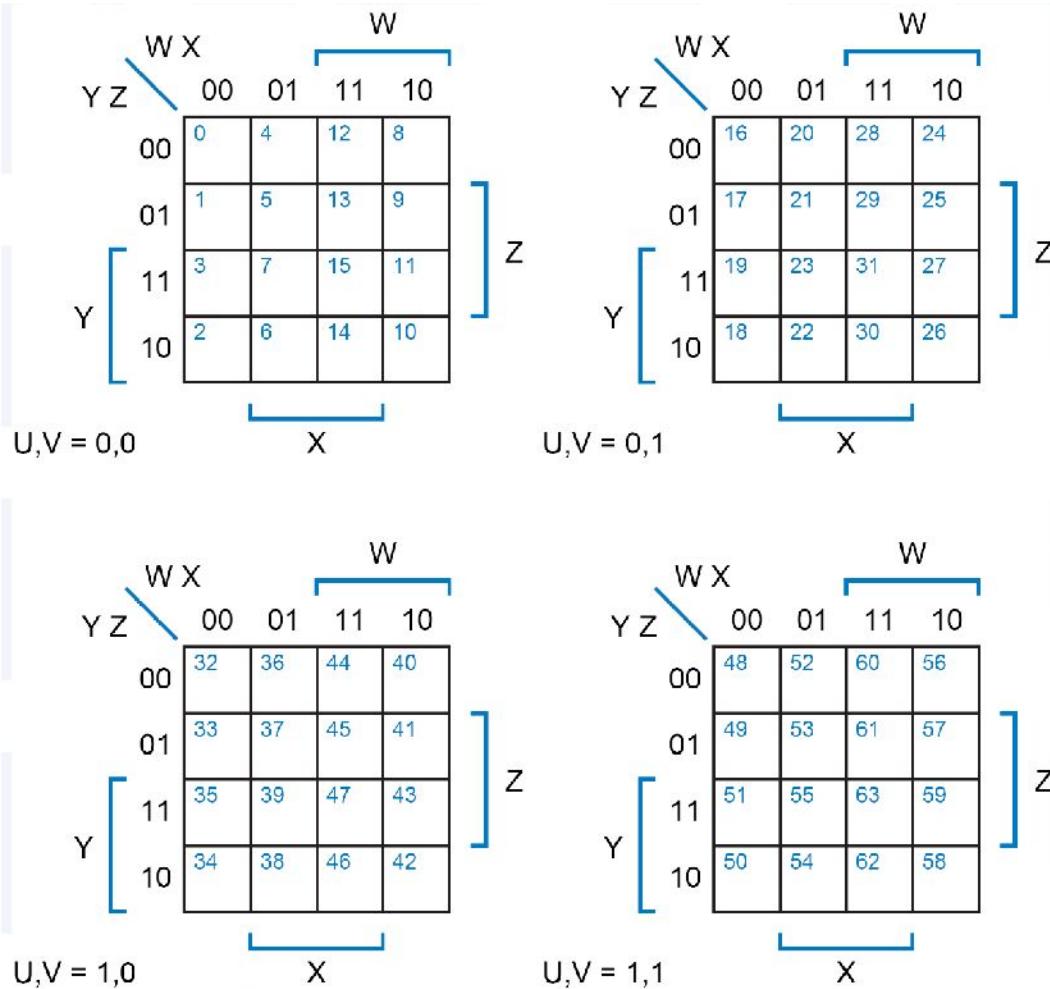


$$f(A,B,C,D,E) = \Sigma m(2,5,7,8,10, 13, 15, 17, 19, 21, 23, 24, 29, 31)$$



$$= C'E + AB'E + BC'D'E' + A'C'DE'$$

6-Variable K-Map



7-Variable K-Map



8-Variable K-Map



Implementing by Nands only

- **Nand:**

- Universal gate
- → can replace gates by equivalent Nand circuit.
 - Large circuit (many gates)
- But

New Symbols for AND/OR

- **DeMorgan's Law:**

➤ $(a + b)' = a' b'$ $(a b)' = a' + b'$

➤ $a + b = (a' b')'$ $(a b) = (a' + b')'$

New Symbols for AND/OR

- **DeMorgan's Law:**

➤ $(a + b)' = a' b'$ $(a b)' = a' + b'$

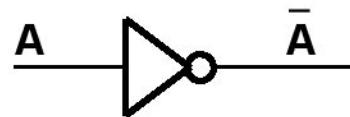


➤ $a + b = (a' b')'$ $(a b) = (a' + b')'$



New Symbols for NOT

➤ $(a \cdot a)' = a'$

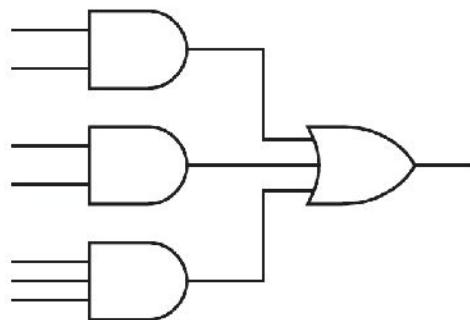


➤ $(a + a)' = a'$



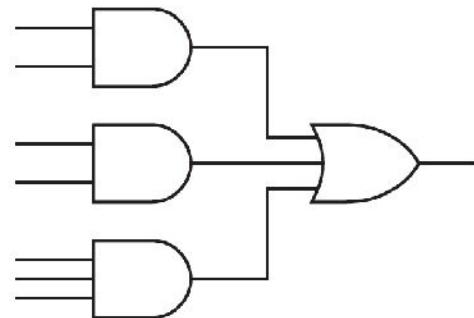
NAND-Only Implementation

- Find Sum-of-Products form.

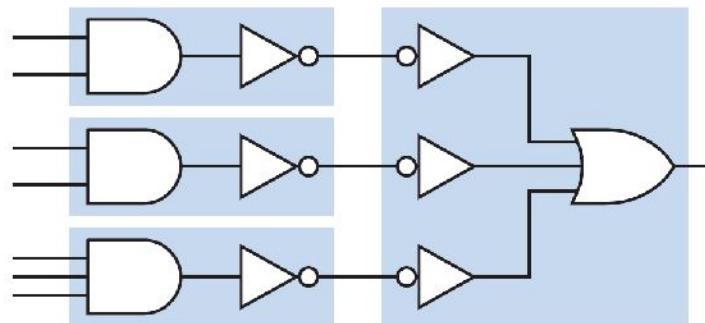


NAND-Only Implementation

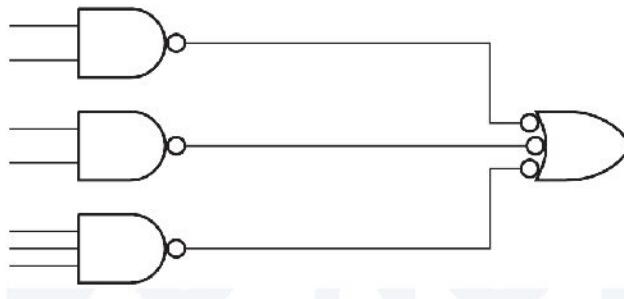
- Find Sum-of-Products form.



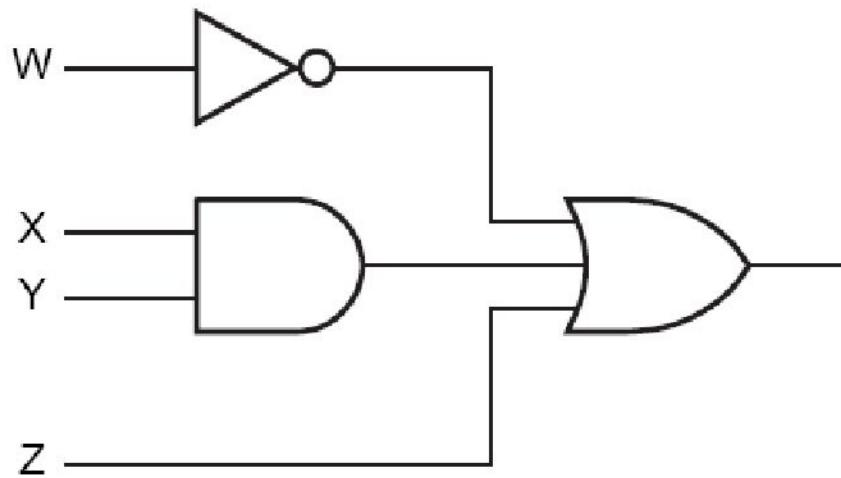
- Inventers can be added



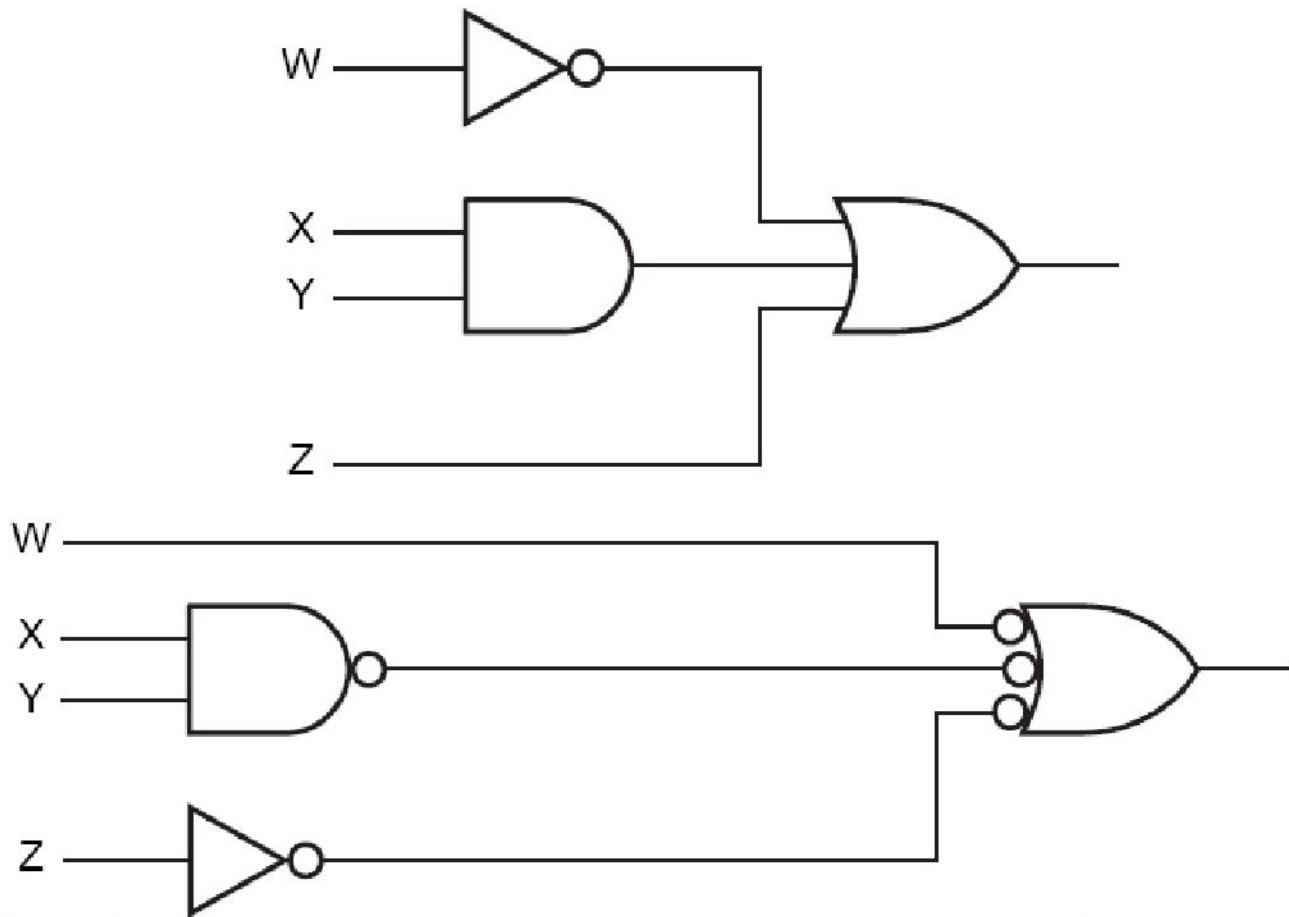
- Equivalent NAND-only



Another Example

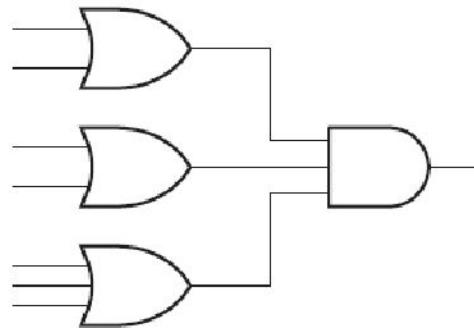


Another Example

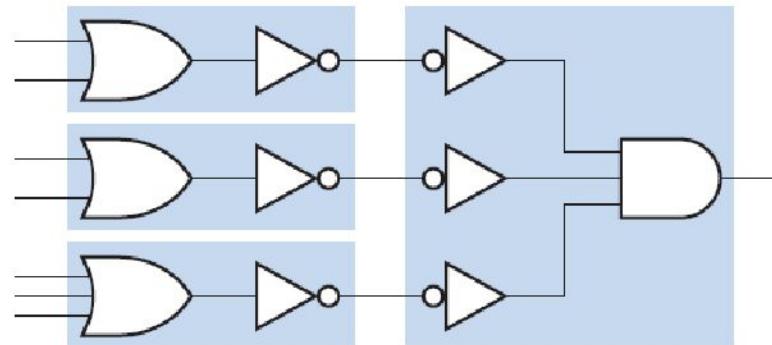


NOR-Only Implementation

- Find Product-of-Sums form.



- Inverters can be added



- Equivalent NOR-only



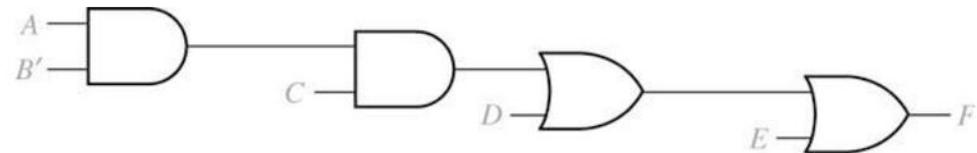
NAND-Only Implementation

- **NAND-only:**
 - Another method:
 - Group 0s in K-Map
 - Find F' in SOP form
 - Add an inverter at the end.

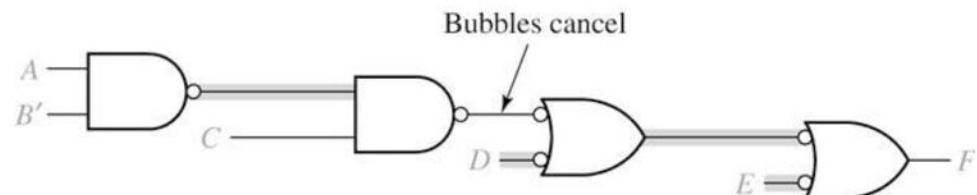
NAND-Only Implementation

- Multi-Level Circuits

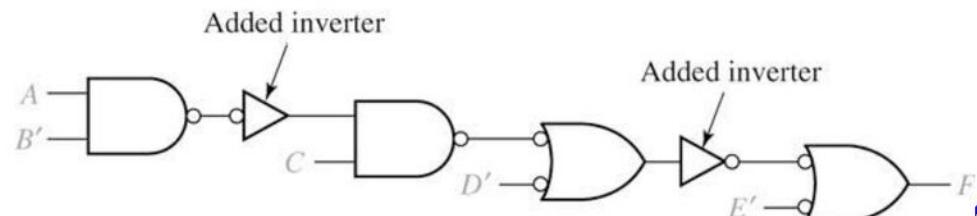
- Convert AND/OR gates to proper NAND gates
 - AND → AND-NOT symbol
 - OR → NOT-OR symbol
- Bubbles must cancel each other;
- otherwise, insert a NAND inverter.
- Take care of appropriate input literals.



(a) AND-OR network



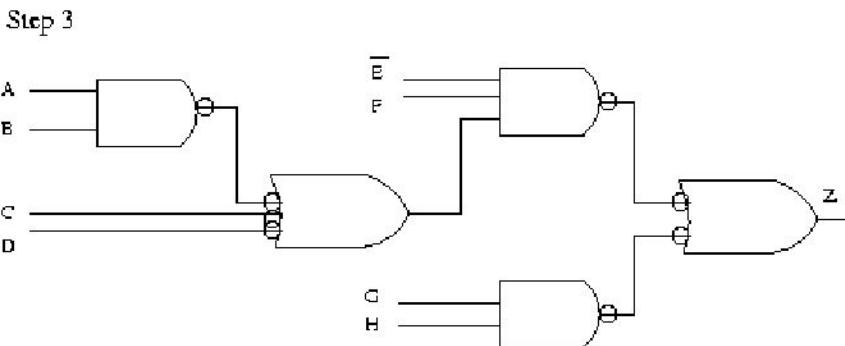
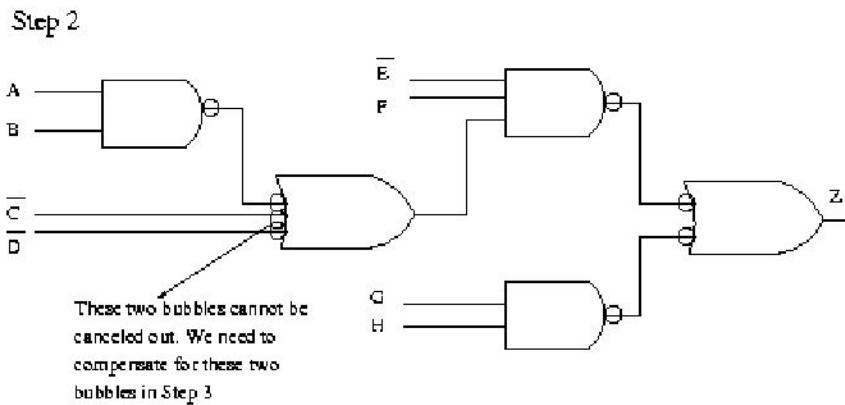
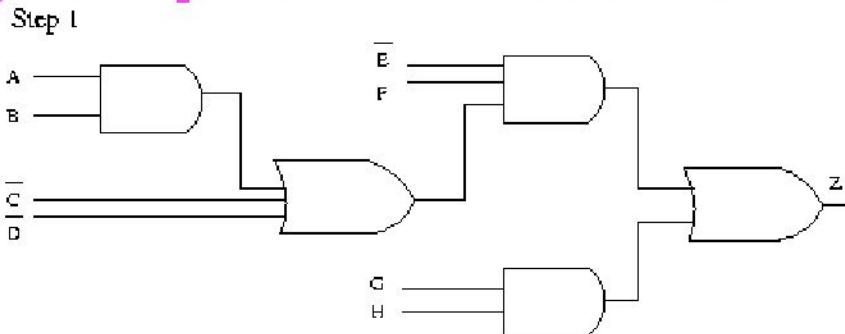
(b) First step in NAND conversion



(c) Completed conversion

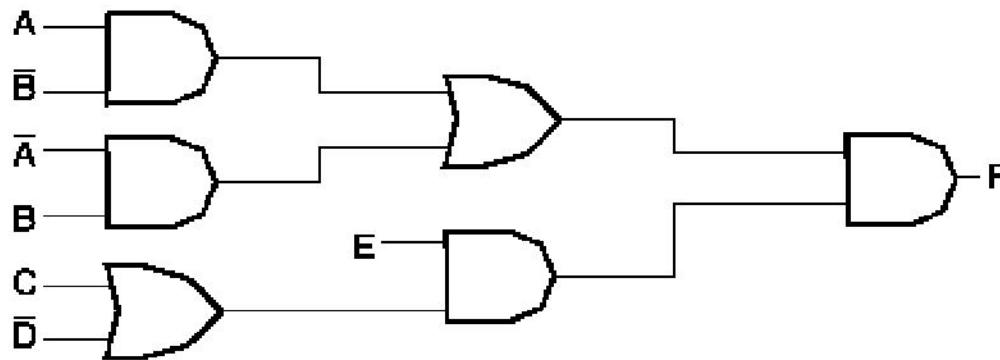
NAND-Only Implementation

- Example:



NAND-Only Implementation

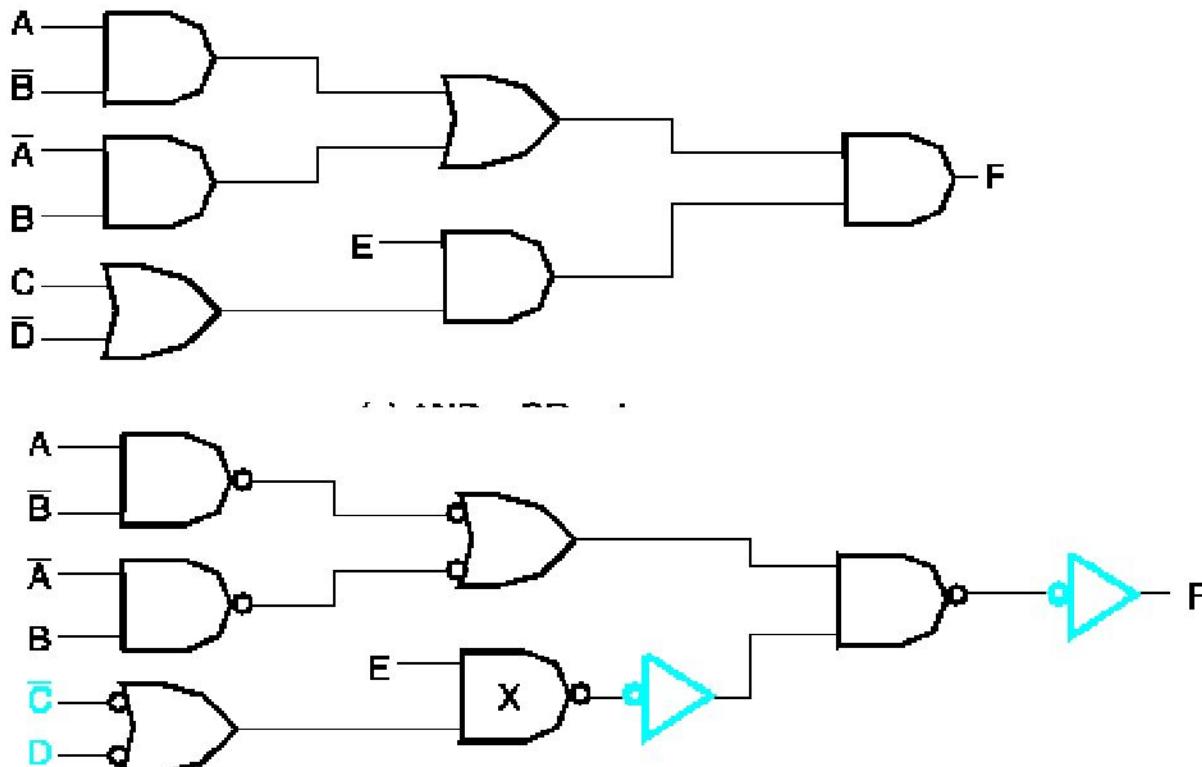
- Another Example:



(a) AND – OR gates

NAND-Only Implementation

- Another Example:

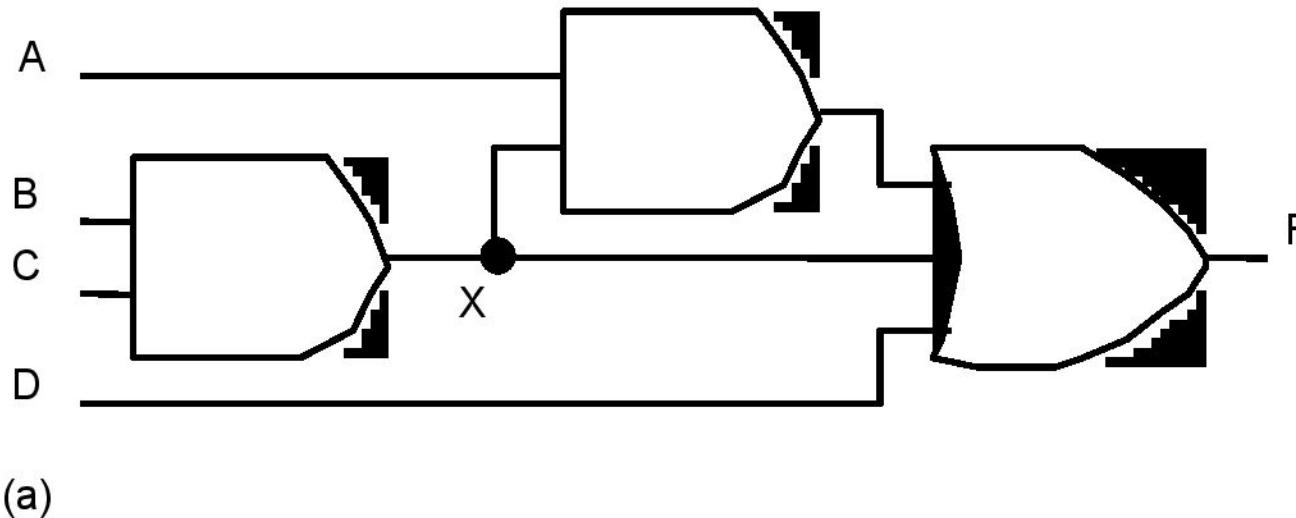


(b) NAND gates

Fig. 2-32 Implementing $F = (A\bar{B} + \bar{A}B)E(\bar{C} + \bar{D})$

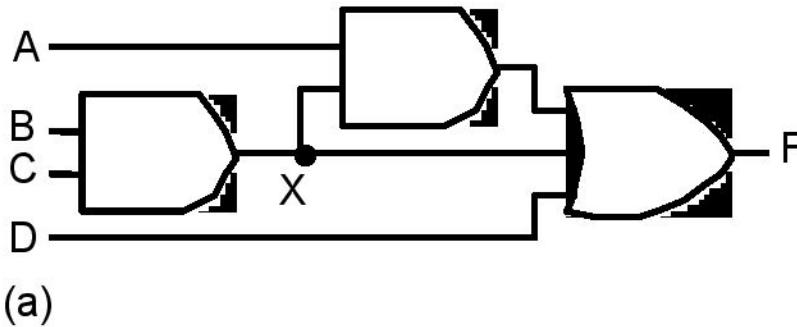
NAND-Only Implementation

- Be careful about branches:
 - Gates with multi-fanouts

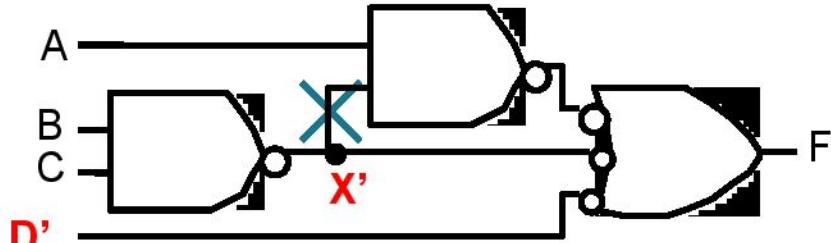


NAND-Only Implementation

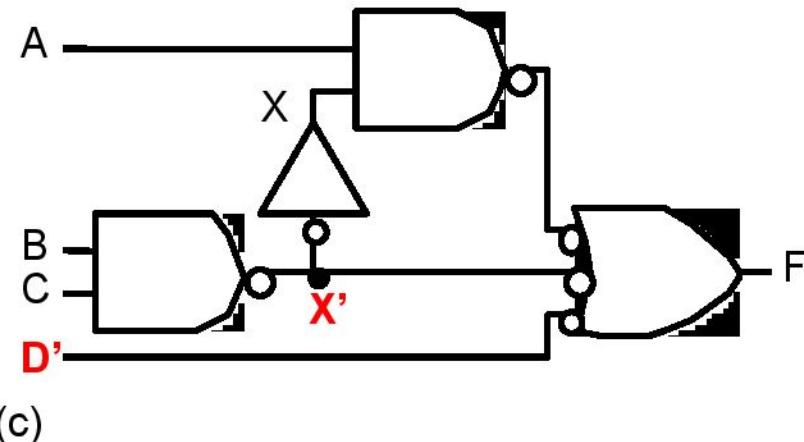
- Be careful about branches:
 - Gates with multi-fanouts



(a)



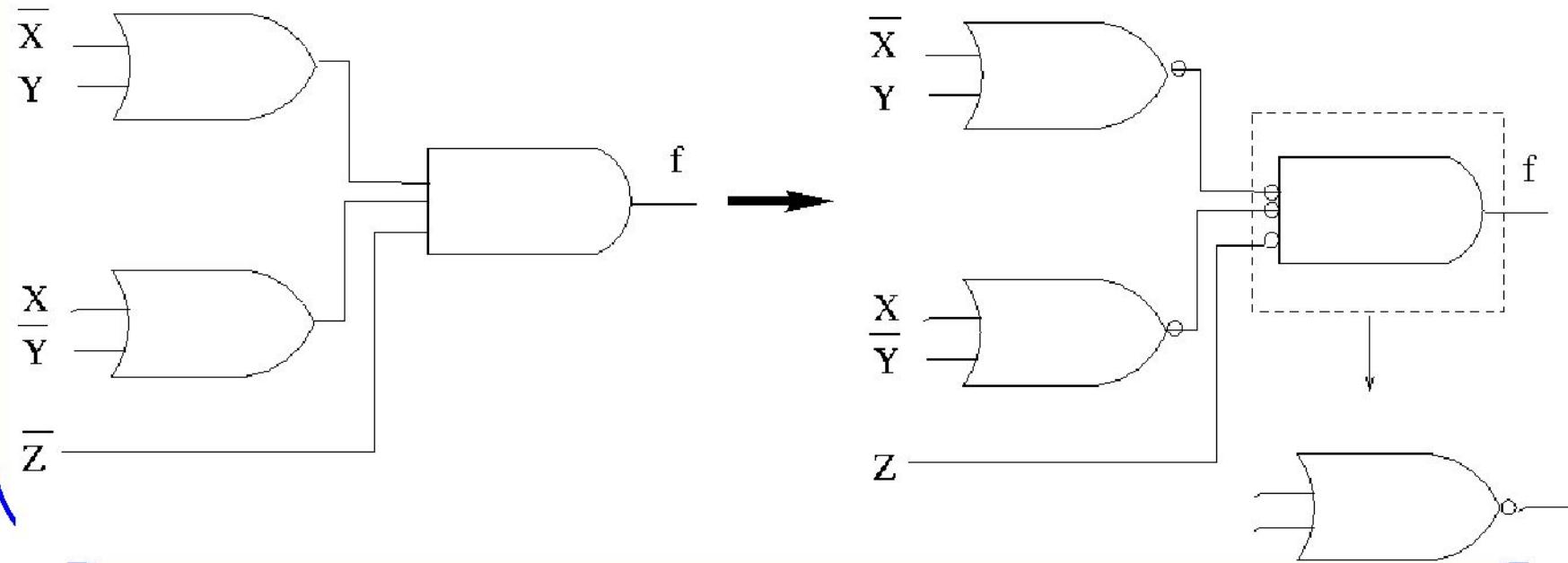
(b)



(c)

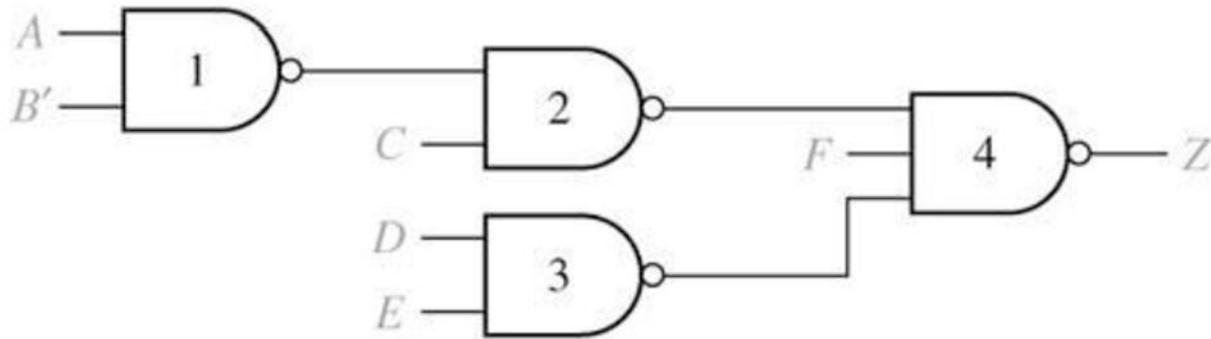
NOR-Only Implementation

- **NOR-Only:**
 - Use “Duality” for the last several slides.



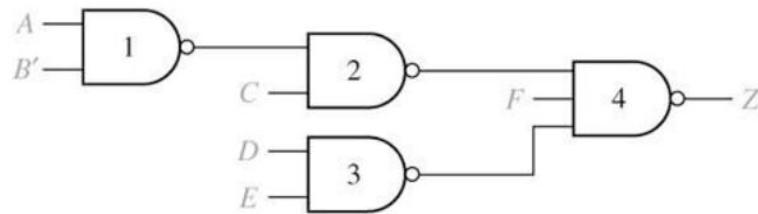
Analysis of NAND Circuits

- The functionality of a NAND-only circuit is not clear.
- Must be converted to AND-OR circuit.

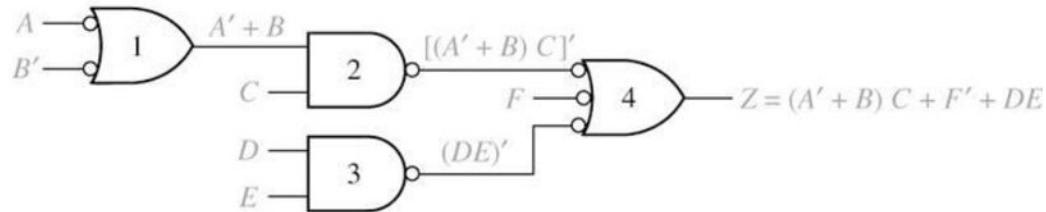


Analysis of NAND Circuits

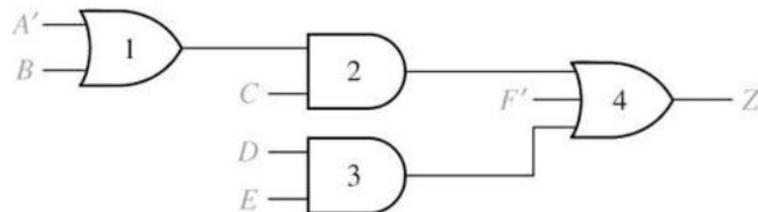
- The functionality of a NAND-only circuit is not clear.
- Must be converted to AND-OR circuit.



(a) NAND gate network



(b) Alternate form for NAND gate network



(c) Equivalent AND-OR network

Analysis of NAND Circuits

- Example

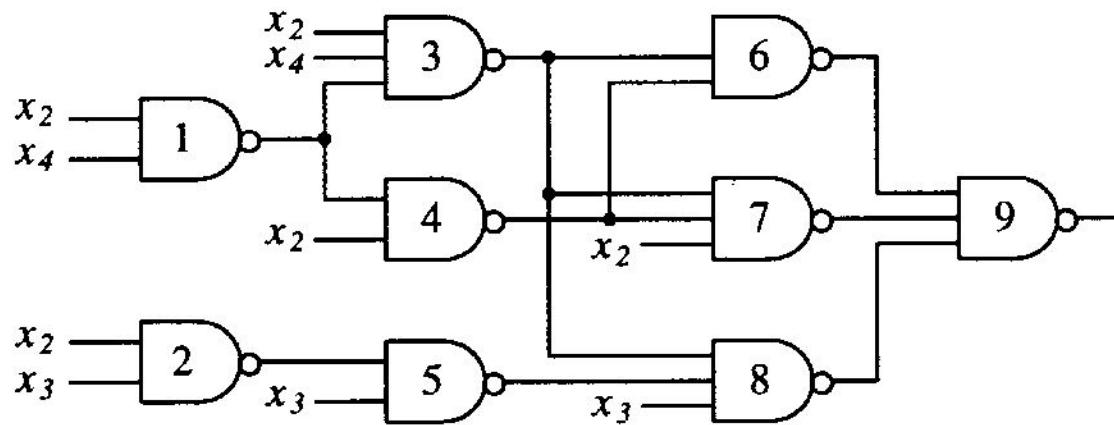


Figure 3.22 When $x_1 = 1$.

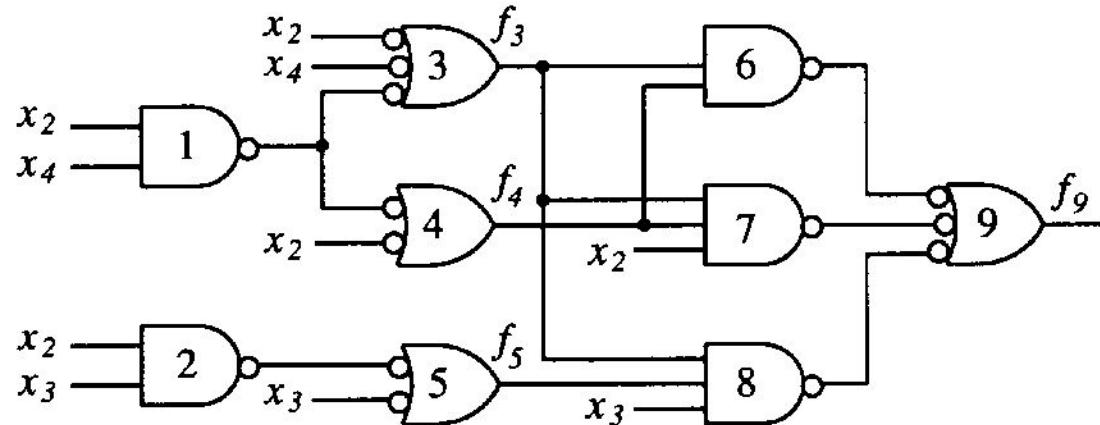
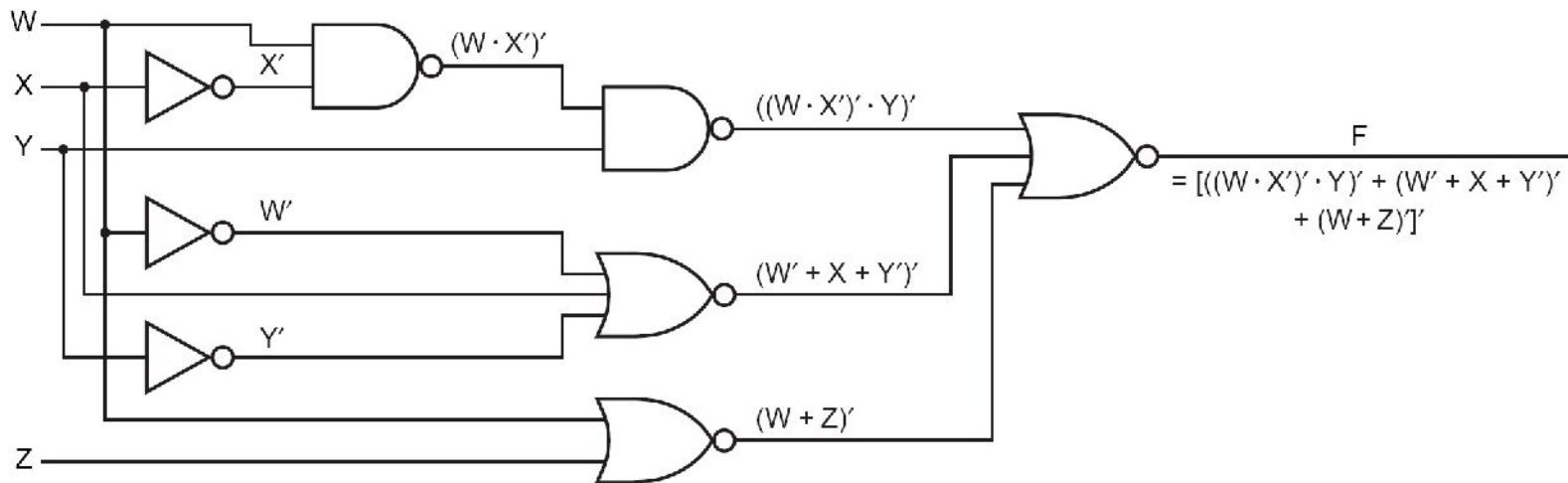
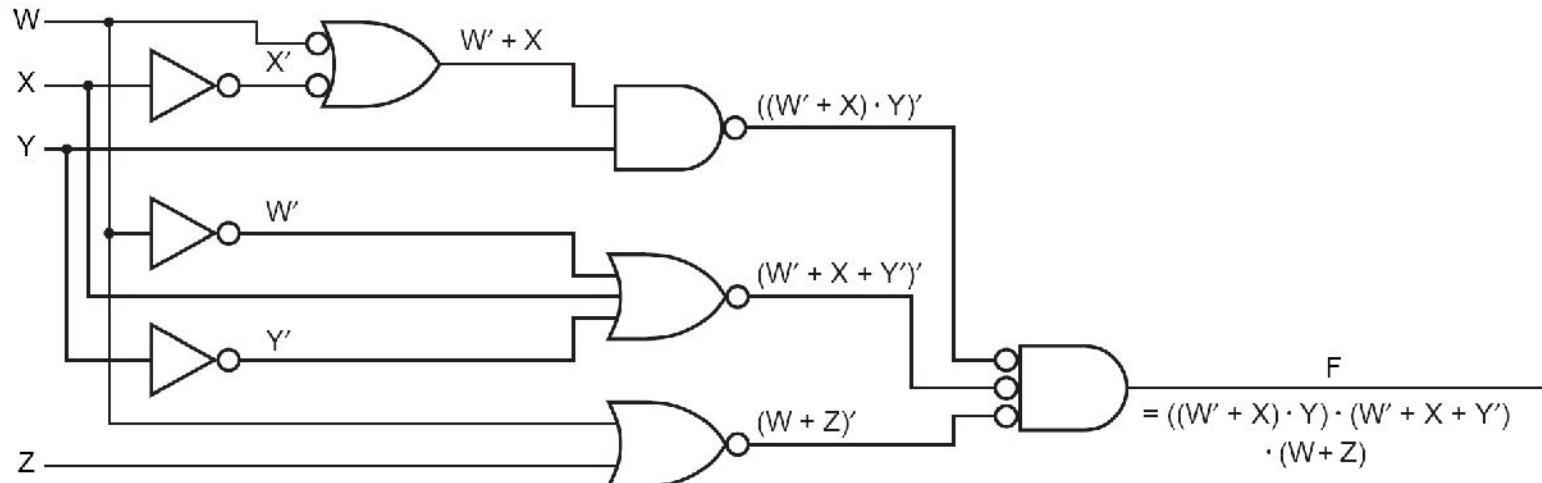
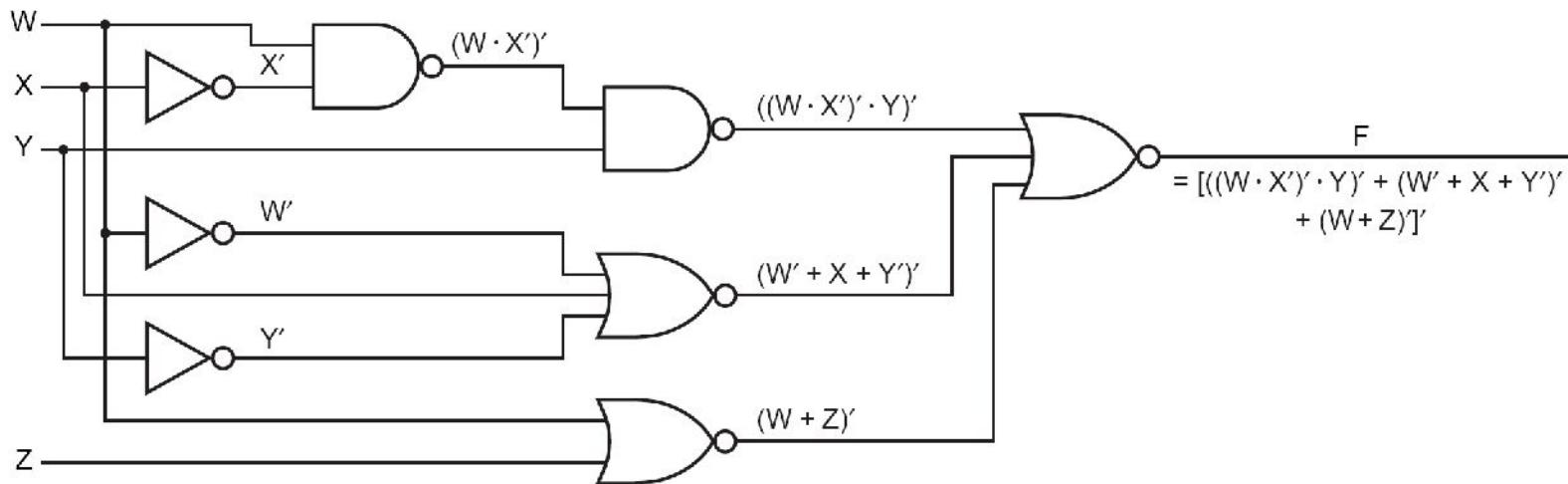


Figure 3.23 Replacement of NAND symbols.

Analysis of NAND/NOR Circuits



Analysis of NAND/NOR Circuits



Analysis of NAND/NOR Circuits

