EEE 3342C: Digital Systems

Chapter 7: Multi-Level Gate Circuits
NAND and NOR Gates

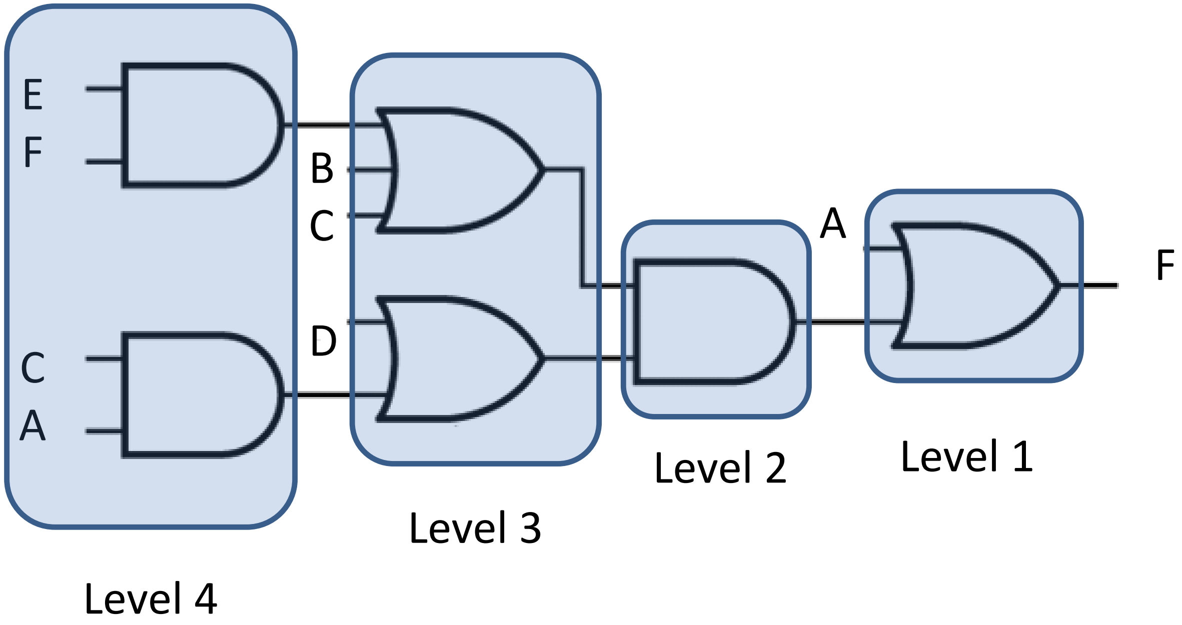
Instructor: Suboh A. Suboh

Department of Electrical Engineering and Computer Science
University of Central Florida

Slides adapted from Zakhia (Zak) Abichar Slides

Multi-Level Circuits

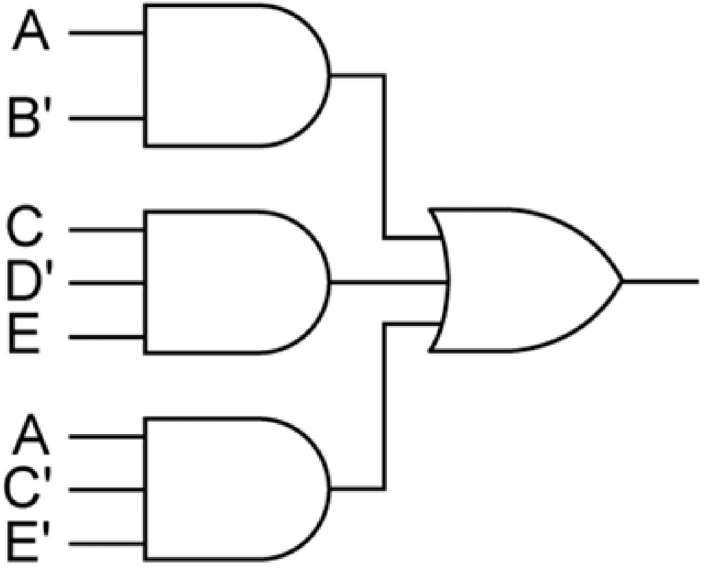
- This circuit has multiple levels



- Let’s say the delay of 1 gate is 10 ns (nanosecond)
- The result is available in 40 ns
- When there are more levels, there is more delay

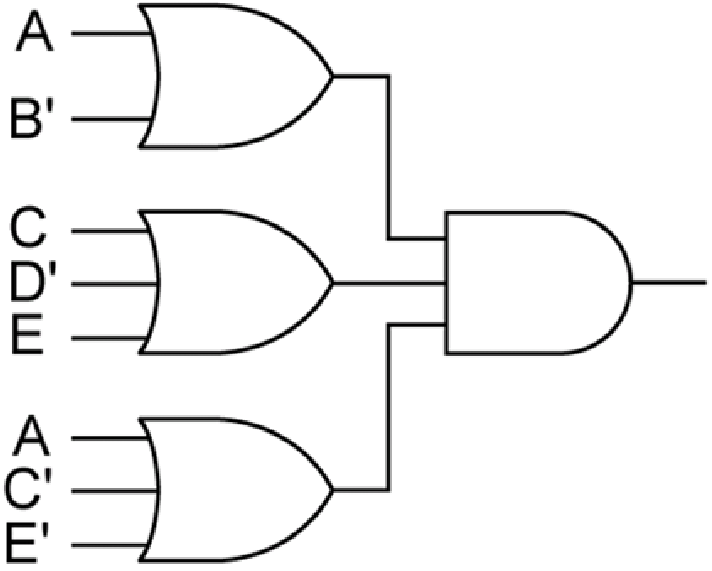
Sum-Of-Products and Product-Of-Sums

The sum-of-products form has 2 levels.



SOP form is also called an AND-OR circuit

The product-of-sums form has 2 levels.

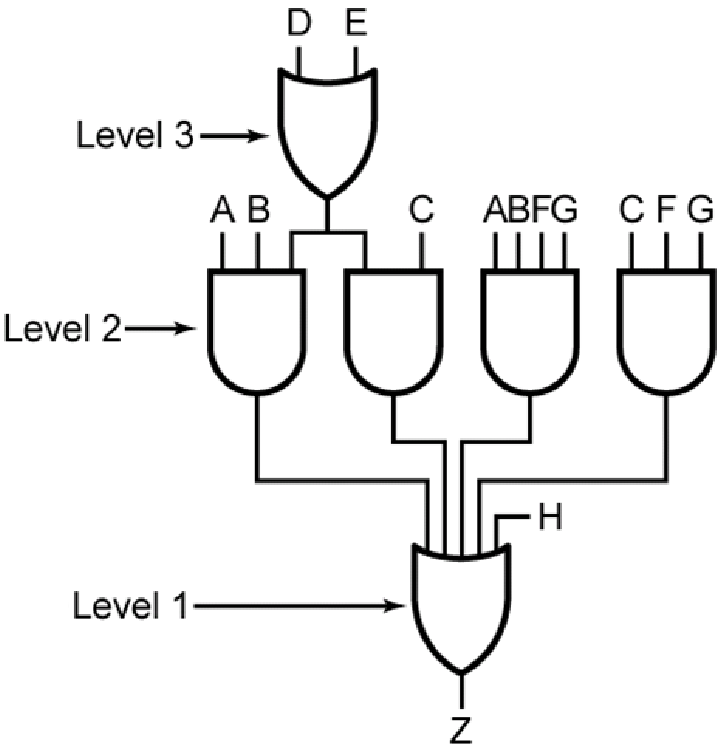


POS form is also called an OR-AND circuit

Factoring and Multiplying Could Change the Number of Levels

- We have this function
- We draw its diagram without changing the function
- We could reuse (D+E) term

$$Z = AB(D + E) + C(D + E) + ABFG + CFG + H$$



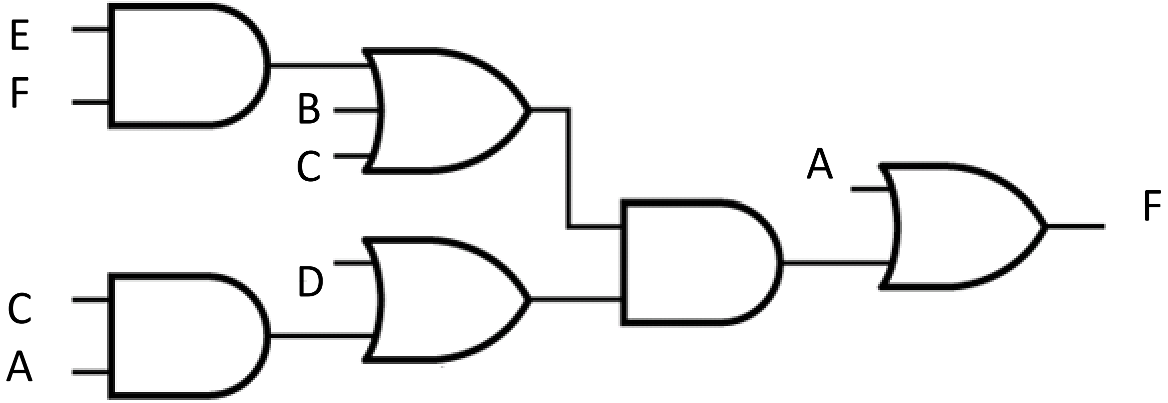
- If we multiply out, we get:

$z=ABD+ABE+CD+CE+ABFG+CFG+H$

- This is a SOP (sum-of-products), it is two level

Cost of a Circuit

- The hardware cost of a circuit is:
  - The number of gates and
  - The total number of inputs



- Number of gates: 6
- Total number of inputs: 13

Finding the Circuit with the Smallest Cost

- We have the function: f(a,b,c,d) = Σm(1, 5, 6, 10, 13, 14)
- We need to find the circuit of f that has the smallest cost
- Draw the K-map of f

❶ Find the minimized SOP expression; find its cost

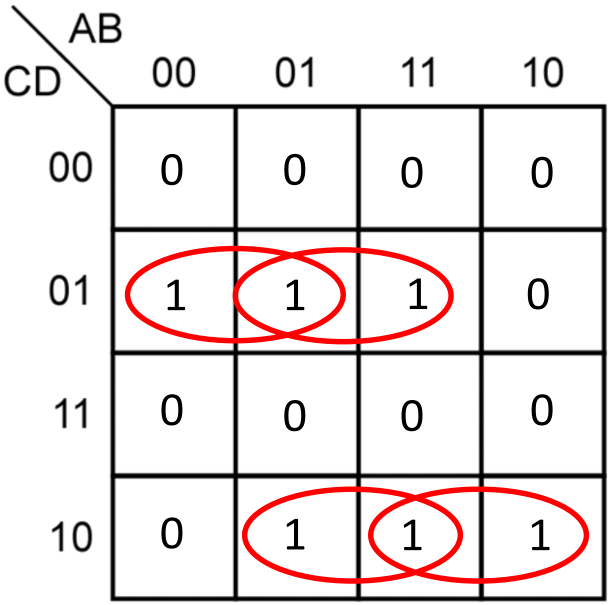
❷ Try to factor it; find its cost

❸ Find the minimized POS expression; find its cost

❹ Try to factor it; find its cost

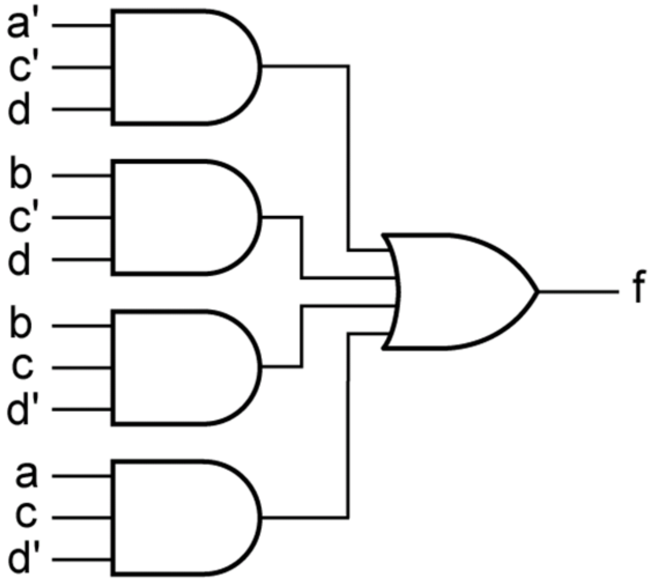
- Find the circuit with the smallest cost above the 4 options above

❶ Find the minimized SOP (sum-of-products) expression; find its cost



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

f = a’c’d + bc’d + bcd’ + acd’



Hardware cost

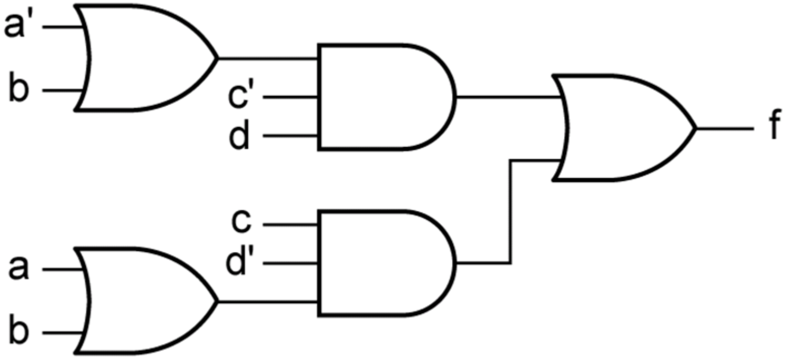
Number of gates: 5

Total number of inputs: 16

❷ Factor the expression

f = a’c’d + bc’d + bcd’ + acd’

f = c’d (a’ + b) + cd’ (a + b)

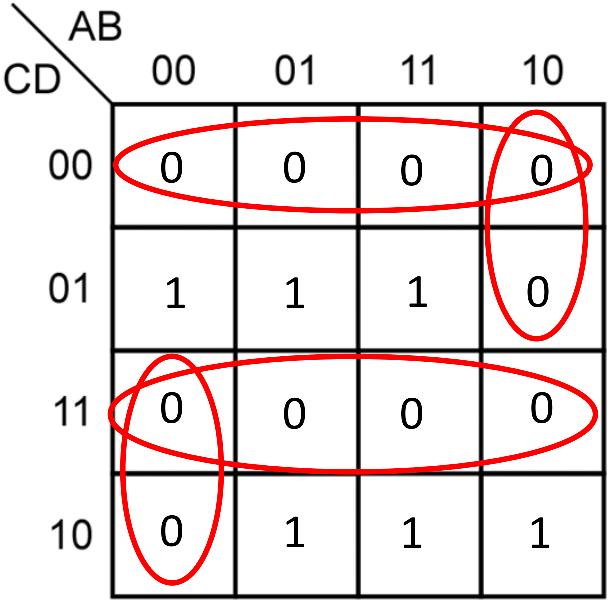


Hardware cost

Number of gates: 5

Total number of inputs: 12

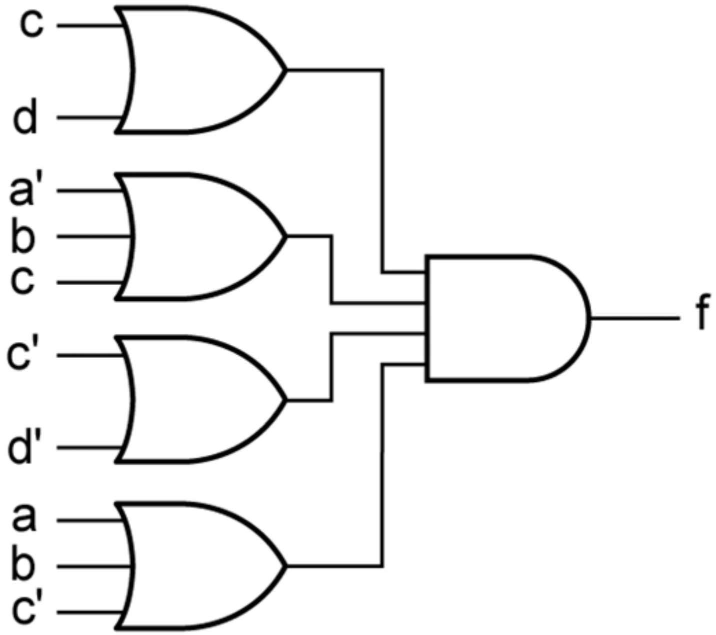
❸ Find the minimized POS (product-of-sums) expression; find its cost



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

$f' = c'd' + ab'c' + cd + a'b'c$

$f = (c+d)(a'+b+c)(c'+d')(a+b+c')$



Hardware cost

Number of gates: 5

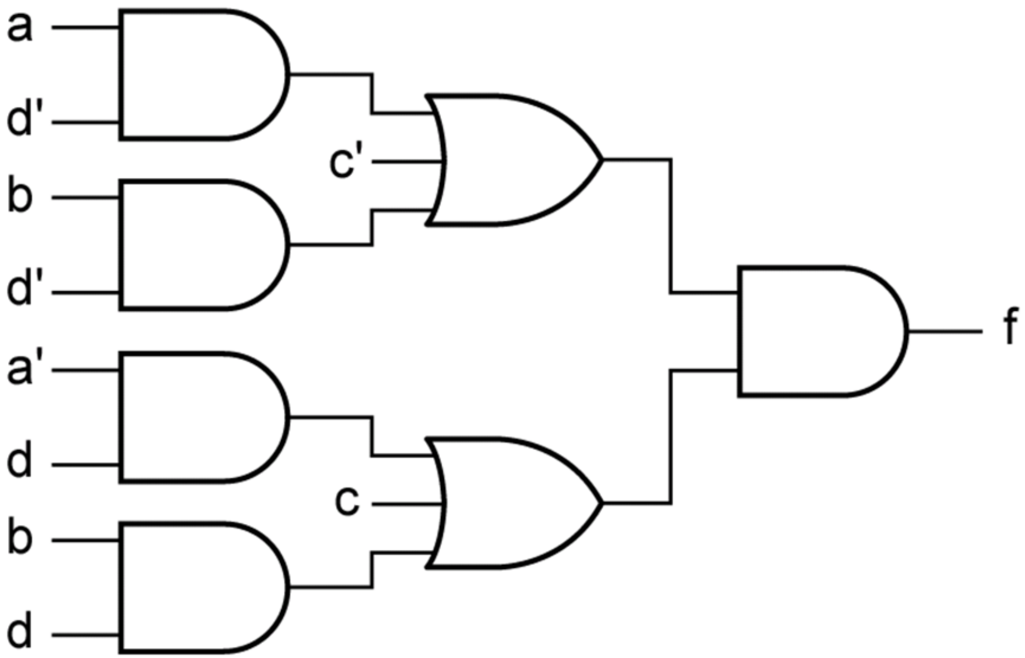
Total number of inputs: 14

❹ Factor the expression

f = (c+d)(a’+b+c)(c’+d’)(a+b+c’)

f = [c + d(a’+b)] [c’ + d’(a+b)]

f = (c + a’d + bd) (c’ + ad’ + bd’)



Hardware cost

Number of gates: 7

Total number of inputs: 16

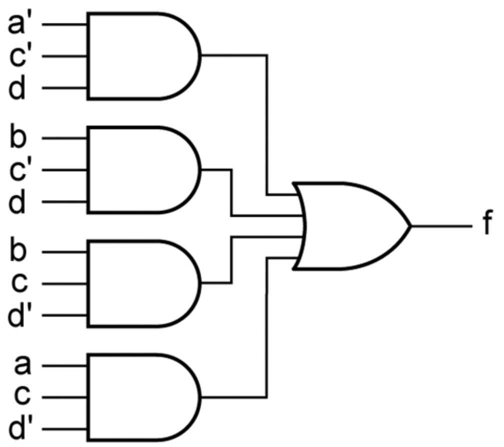
All the Results we Got

1

Hardware cost

Number of gates: 5

Total number of inputs: 16



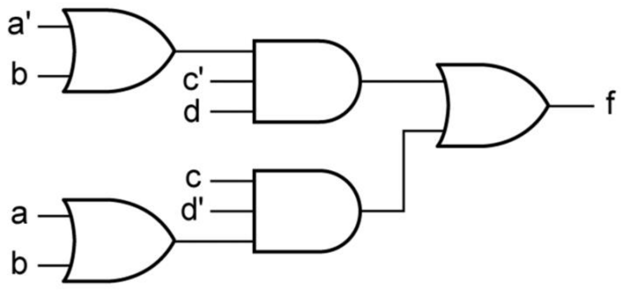
2

Lowest Hardware Cost

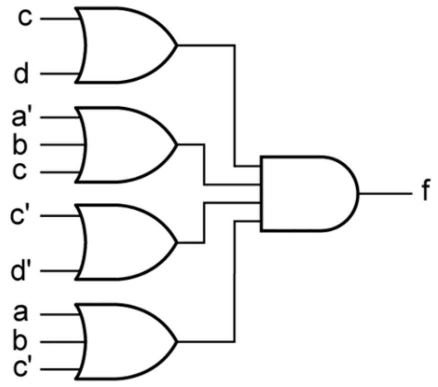
Hardware cost

Number of gates: 5

Total number of inputs: 12



3

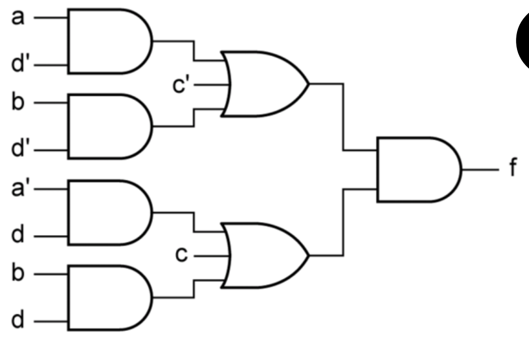


Hardware cost

Number of gates: 5

Total number of inputs: 14

4



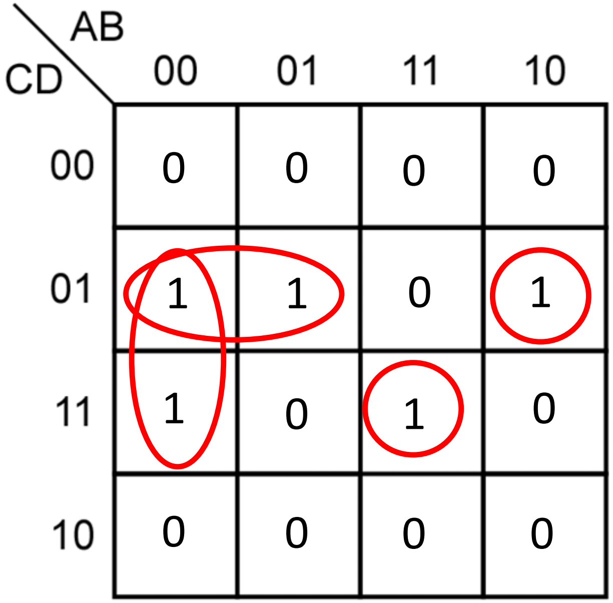
Hardware cost

Number of gates: 7

Total number of inputs: 16

Find the Circuit with the Smallest Cost

- We have the function: F(A,B,C,D) = Σm(1, 3, 5, 9, 15)



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

F = A’C’D + A’BC’D + A’B’D + ABCD

# gates: 5
# inputs: 18

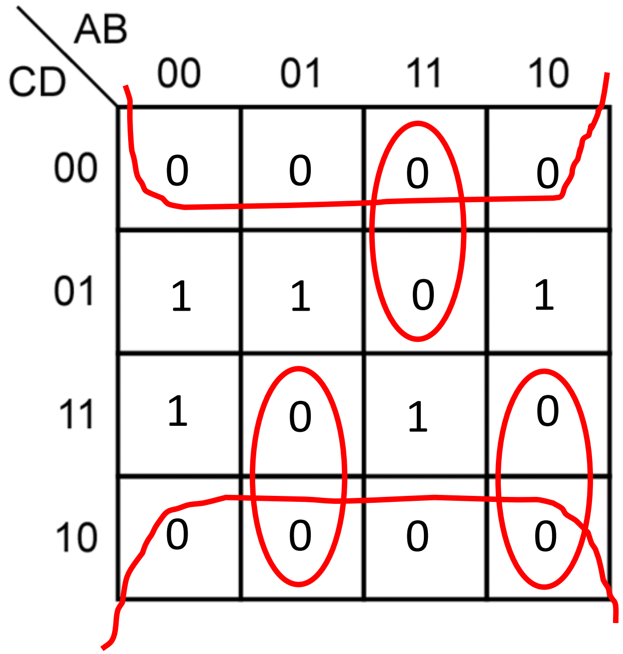
Factor:

F = C’D(A’ + A’B) + D(A’B’ + ABC)

#gates: 8
#inputs: 18

Find the Circuit with the Smallest Cost

- We have the function: F(A,B,C,D) = Σm(1, 3, 5, 9, 15)
- Find the expression of f’ and convert it to POS



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

F’ = D’ + ABC’ + A’BC + AB’C

F = D (A’+B’+C)(A+B’+C’)(A’+B+C’)

# gates: 4

# inputs: 13

Factor:

F = D[B’+(A’+C)(A+C’)](A’+B+C’)

= D(B’+A’C’+AC)(A’+B+C’)

\#gates: 5

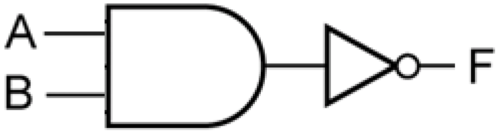
\#inputs: 13

Out of the 4 cases we found, this is the smallest cost.

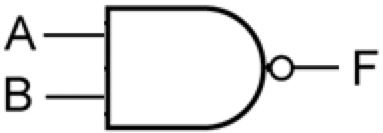
NAND Gate

- NAND gate is a “Not AND” gate
- Invert of AND

A	B	AND	NAND
0	0	0	1
0	1	0	1
1	0	0	1
1	1	1	0



NAND gate is an AND gate with
an inverter on the output

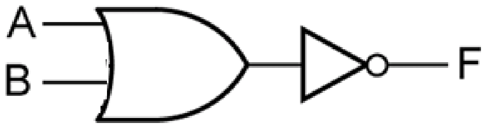


This is the symbol of the NAND gate
The bubble (round circle) on the output
signifies an inverter

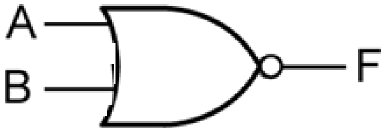
NOR Gate

- NOR gate is a “Not OR” gate
- Invert of OR

A B	OR	NOR
0 0	0	1
0 1	1	0
1 0	1	0
1 1	1	0

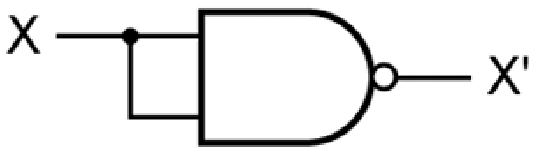


NOR gate is an OR gate with an inverter on the output

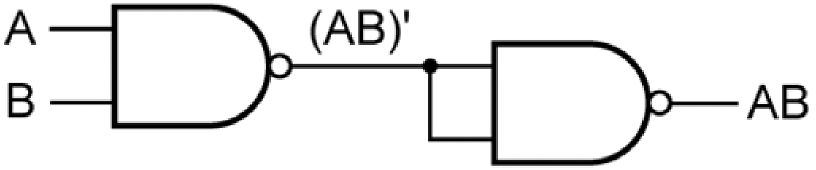


This is the symbol of the NOR gate
The bubble (round circle) on the output signifies an inverter

Using NAND gates to make an inverter, an AND gate and a NOR gate

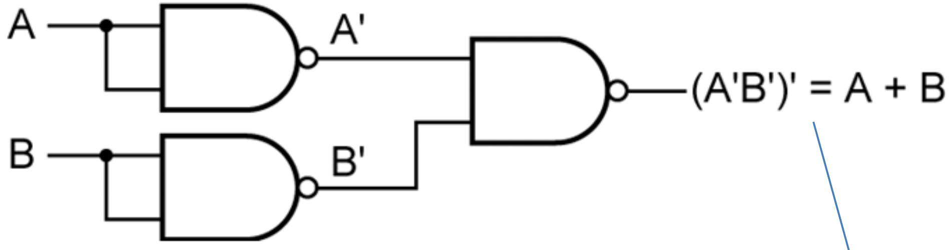


An inverter



An AND gate

A	B	NAND
0	0	1
0	1	1
1	0	1
1	1	0



An OR gate

By DeMorgan's

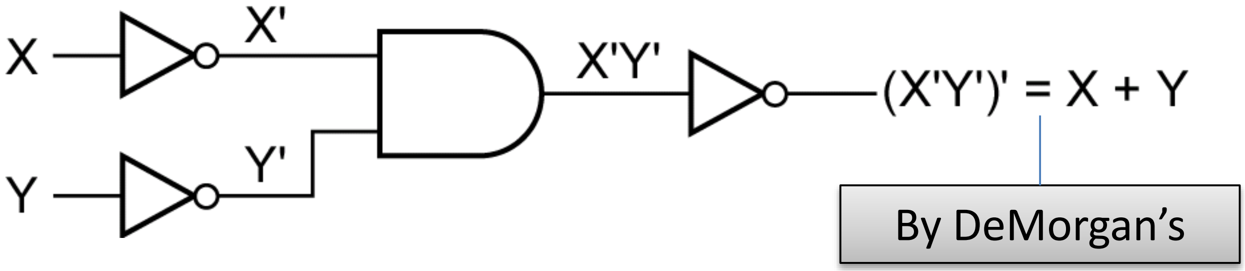
NAND is called a functionally complete set of gates.

Using NAND, we can implement any logic circuit.

Functionally Complete Set of Gates

- A set of gates is called functionally complete
  - We can implement any logic circuit using gates from this set
- This set {AND, OR, INVERTER} is functionally complete
- This set {NAND} is functionally complete
- This set {NOR} is functionally complete
- This set {AND, INVERTER} is functionally complete

Using AND and NOT, we can make an OR gate:



By DeMorgan's

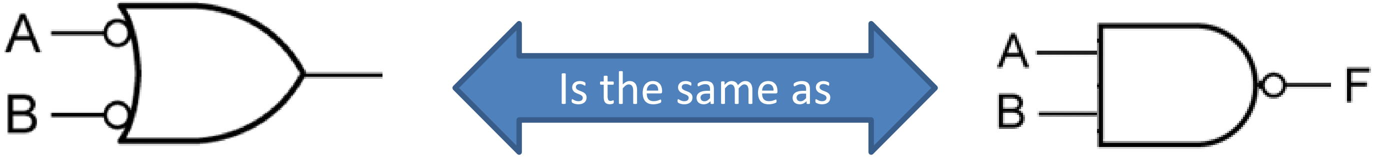
NAND-only Circuit & NOR-only Circuit

- NAND-only circuit
  - A circuit that has NAND gates only
- NOR-only circuit
  - A circuit that has NOR gates only

- Any logic function can be implemented using NAND gates only

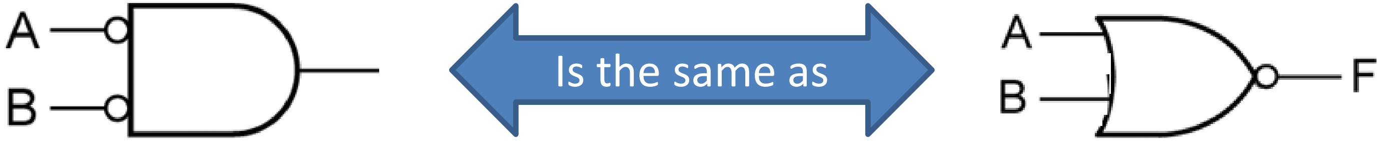
- Any logic function can be implemented using NOR gates only

Things to Remember



Is the same as

$$A' + B' \quad = \quad [(A' + B')']' \quad = \quad [A.B]'$$

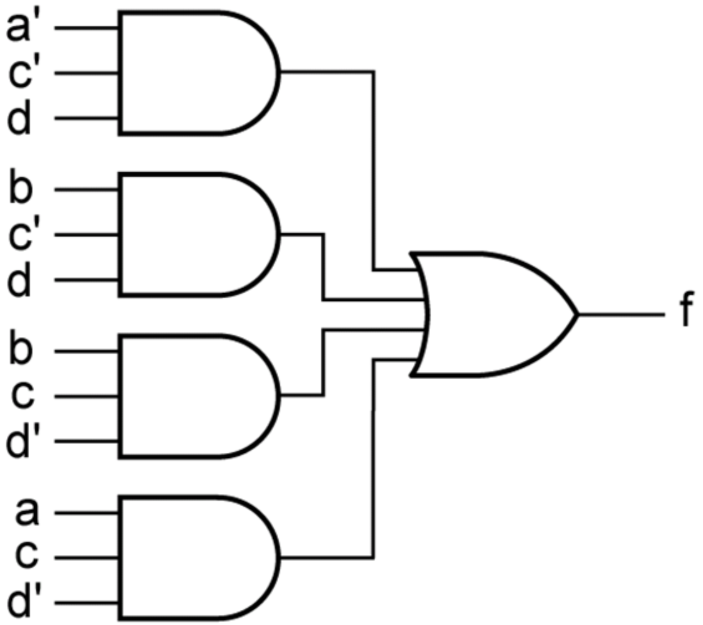


Is the same as

$$A'.B' \quad = \quad [(A'.B')']' \quad = \quad [A+B]'$$

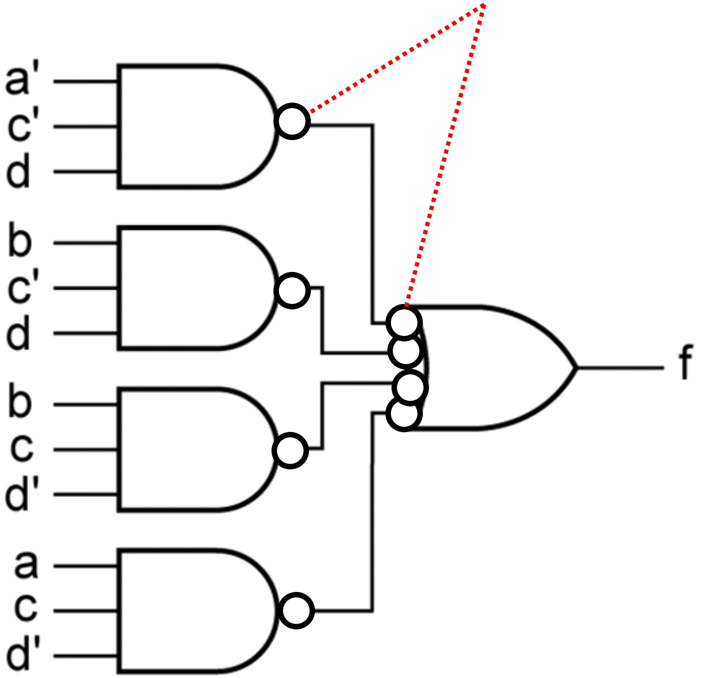
Transforming a Circuit to NAND-only Circuit

- Write the sum-of-product form of the function
- Draw the diagram
- Replace all the gates by NAND

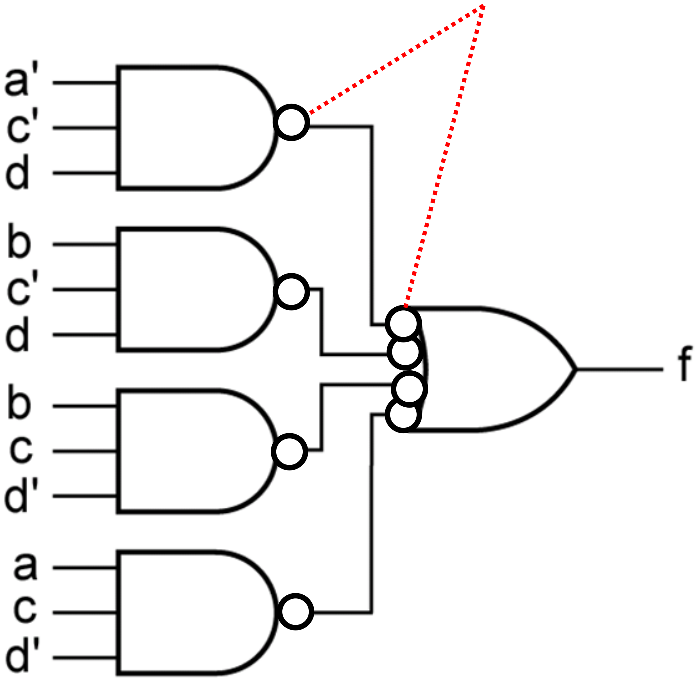


Sum-of-products

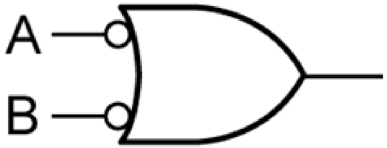
These bubbles will cancel out; these circuits are equivalent



These bubbles will cancel out; these circuits are equivalent

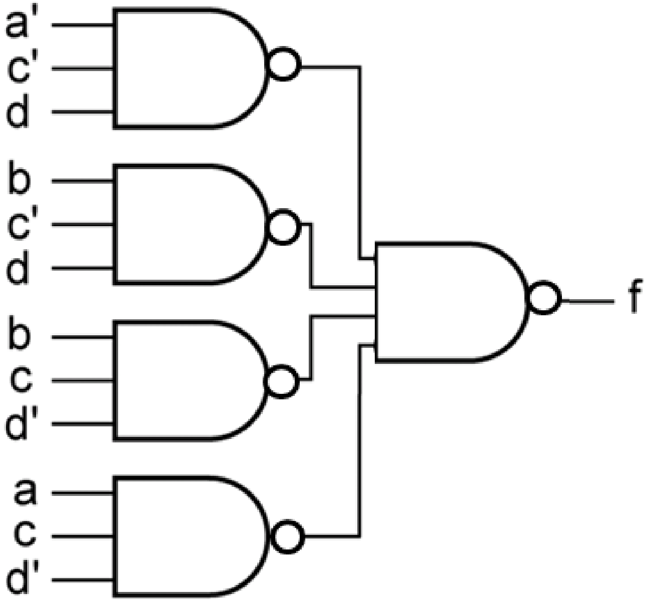


This is equivalent to a NAND gate



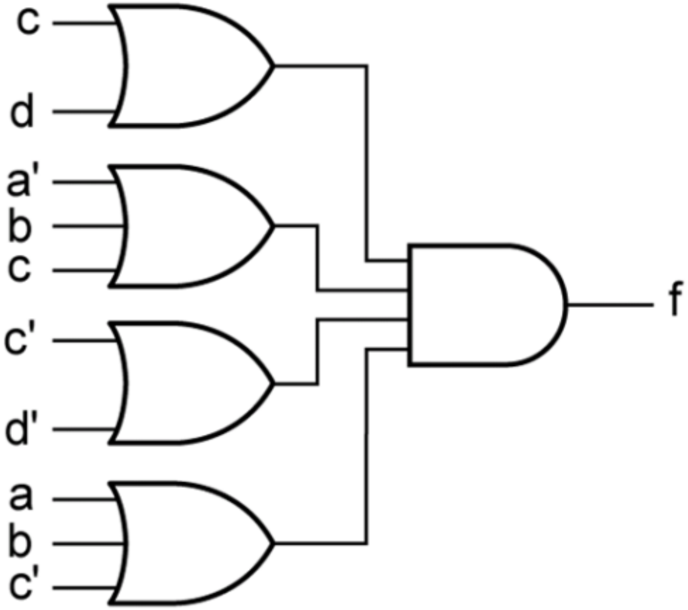
$$A' + B' = [(A'+B')']'$$

$$= (A.B)' = A \text{ nand } B$$

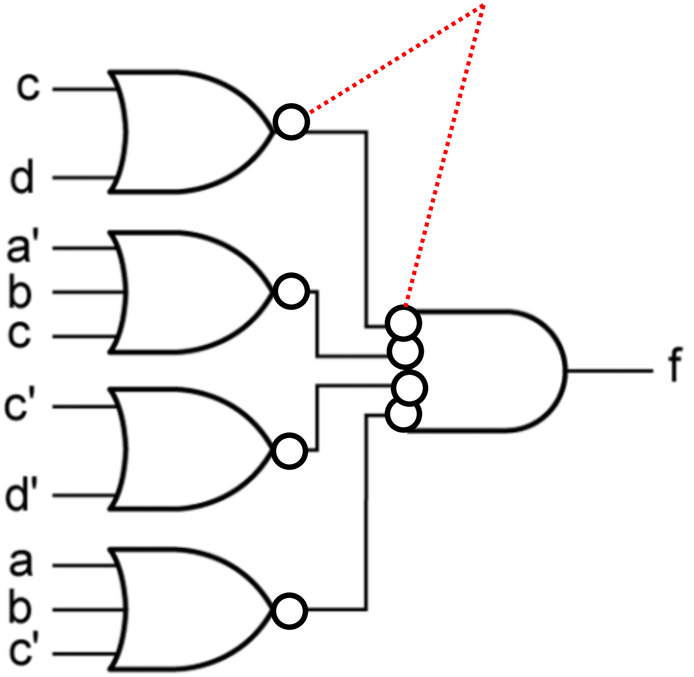


Transforming a Circuit to NOR-only Circuit

- Write the product-of-sums form of the function
- Draw the diagram
- Replace all the gates by NOR

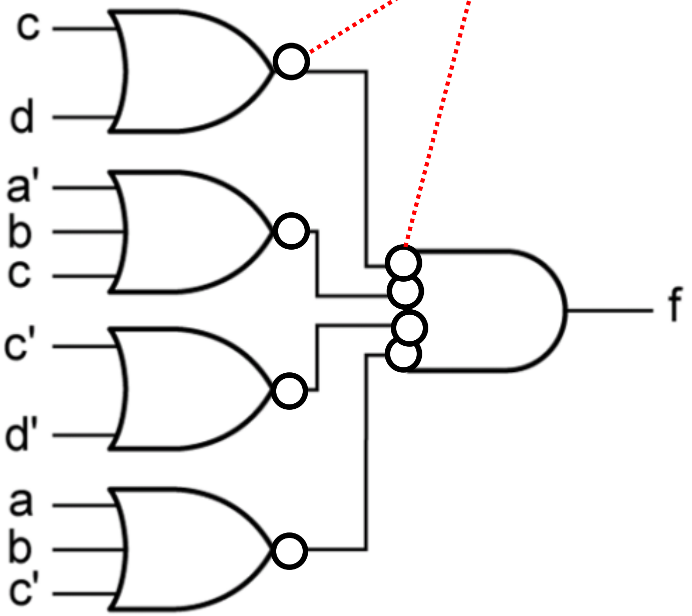


Product-of-sums

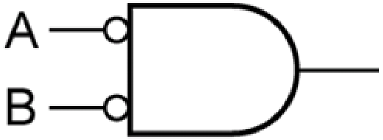


These bubbles will cancel out; these circuits are equivalent

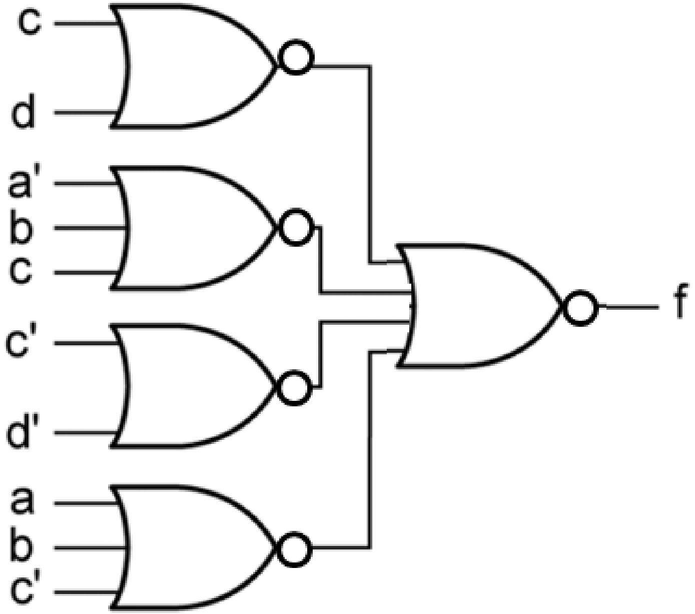
These bubbles will cancel out; these circuits are equivalent



This is equivalent to a NOR gate

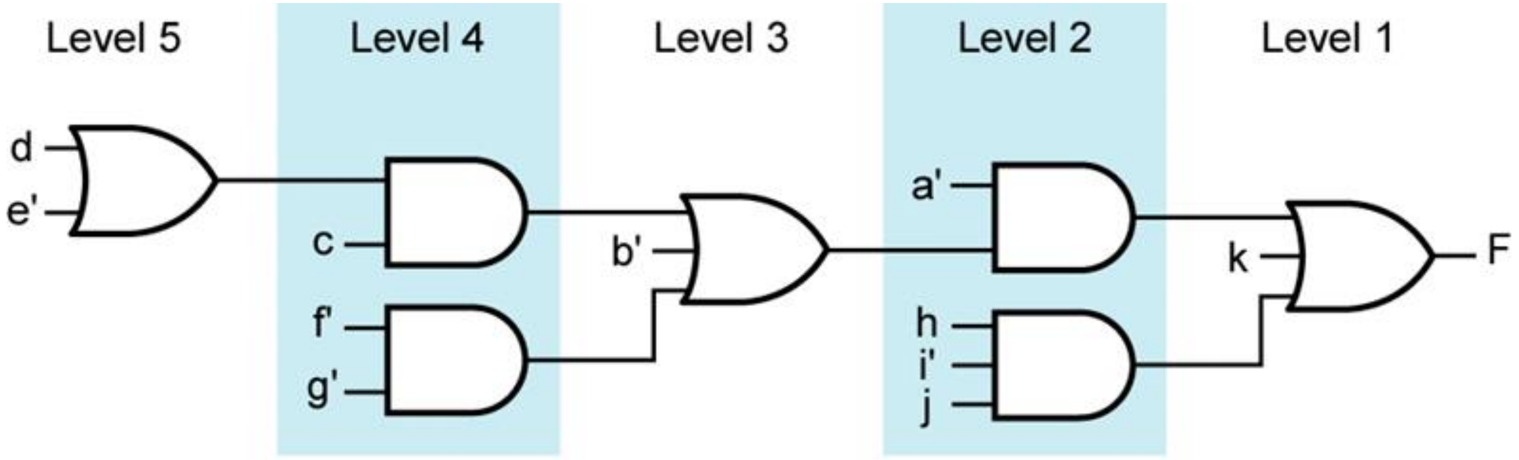


$$A'B' = [(A'B')'\,]' = (A+B)' = A \text{ nor } B$$

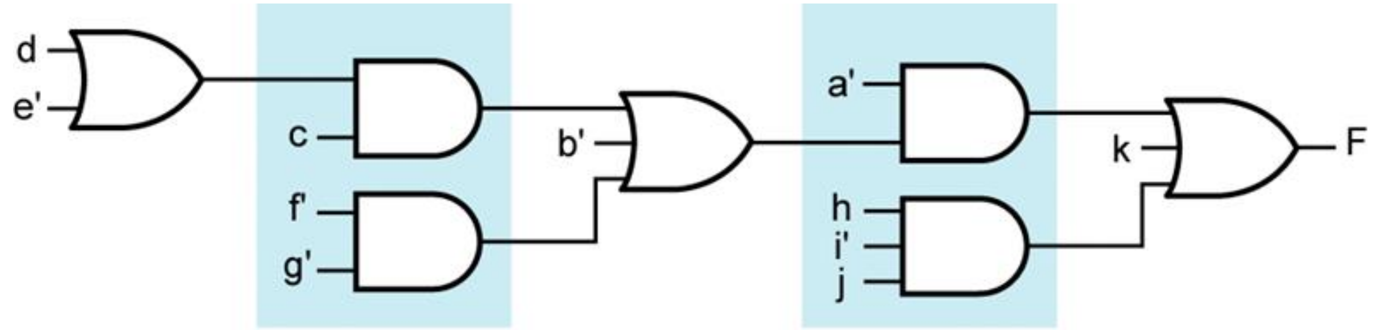


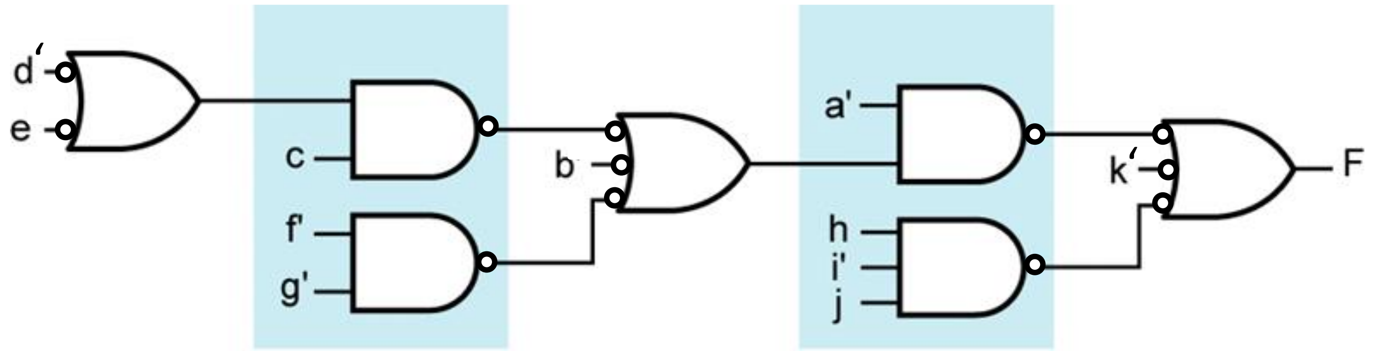
Transforming an Multi-Level Circuit to NAND-only Circuit

F = a’ [b’ + c(d+e’) + f’g’] + hi’j + k

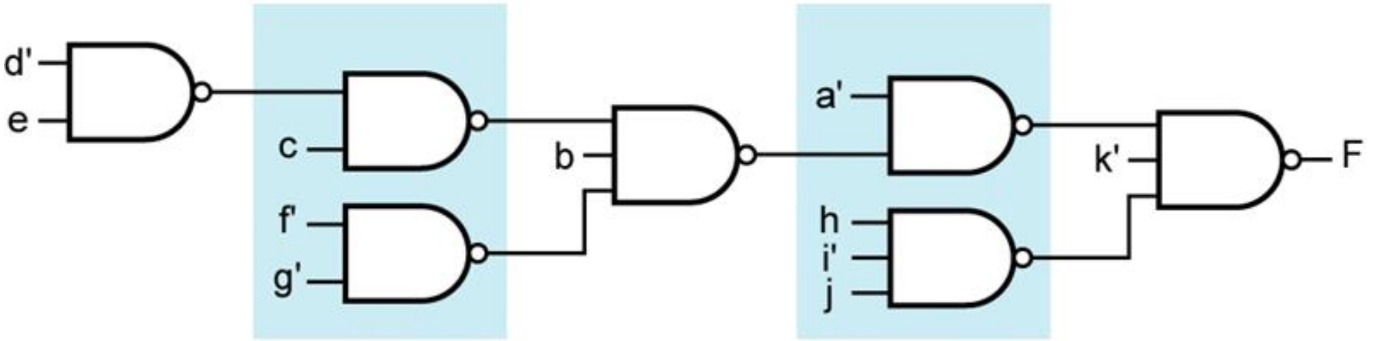


We need to transform this circuit to a NAND-only circuit

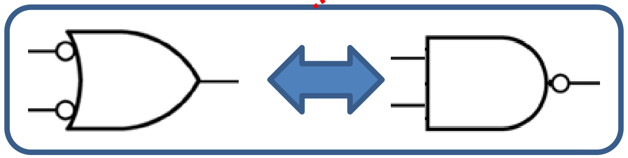




Add the bubble (inverters) on the output of AND and input of OR. Complement signals on inputs of OR.

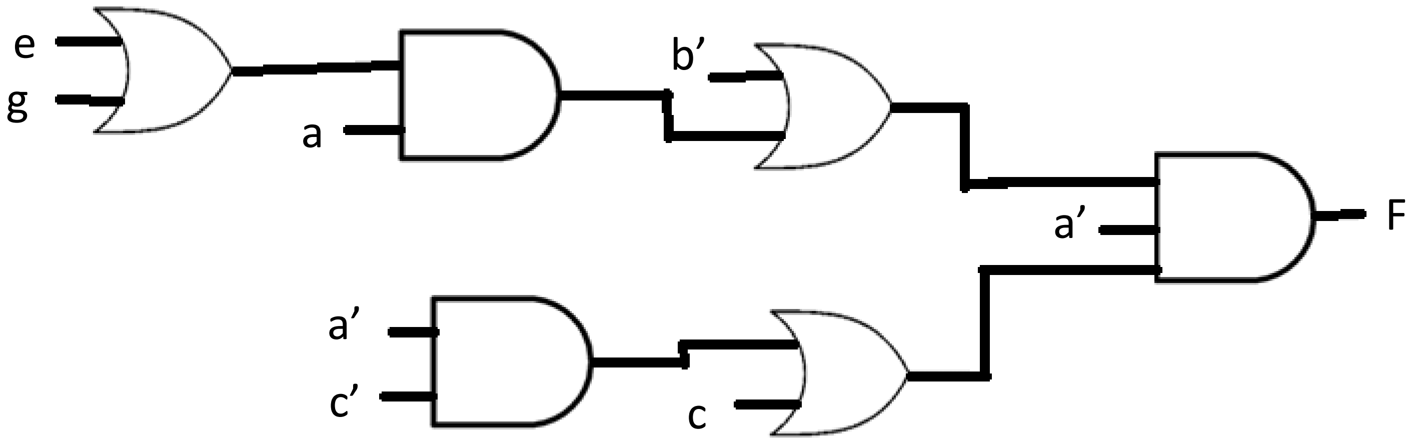


The AND already became NAND. The OR gates with bubbles on the input are equivalent to NAND

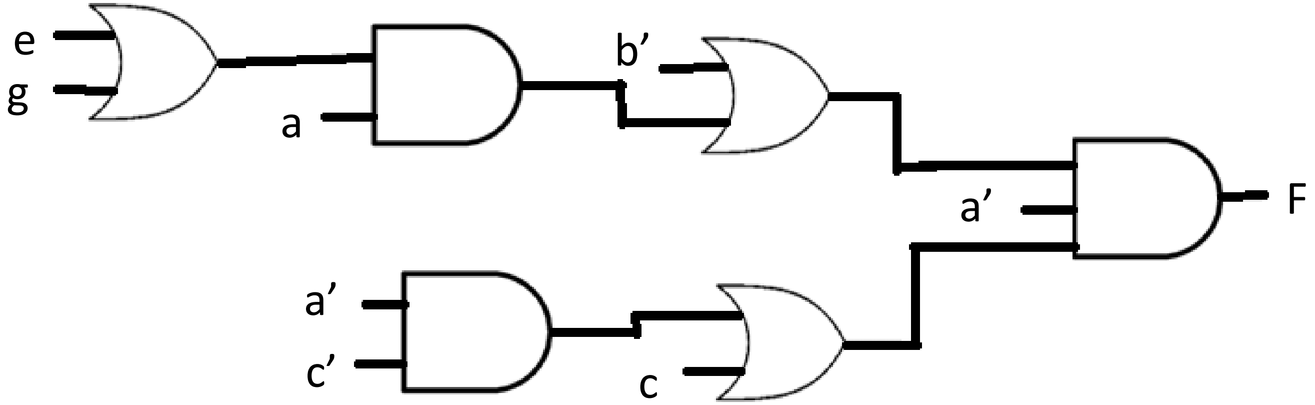


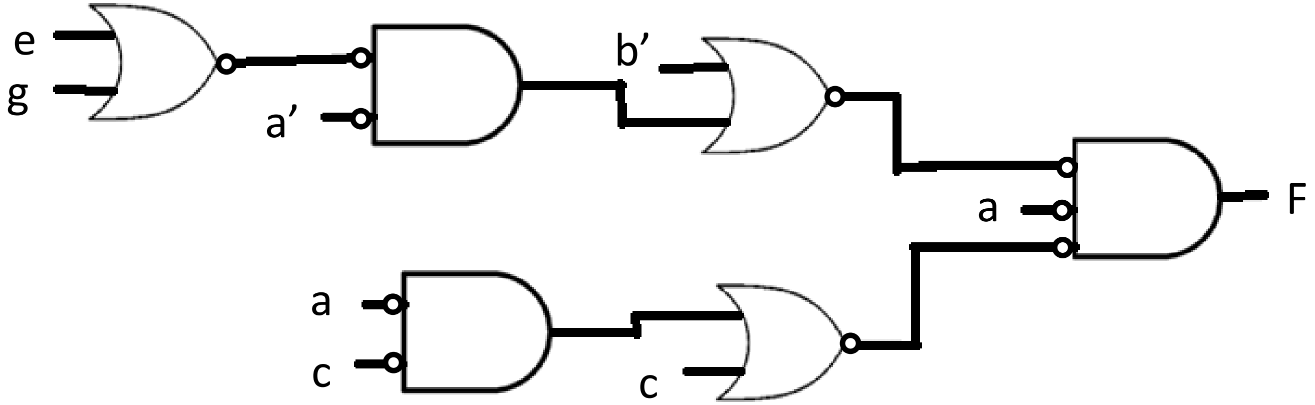
Transforming an Multi-Level Circuit to NOR-only Circuit

F = a’.[a(e+g) + b’].[c+(a’.c’)]

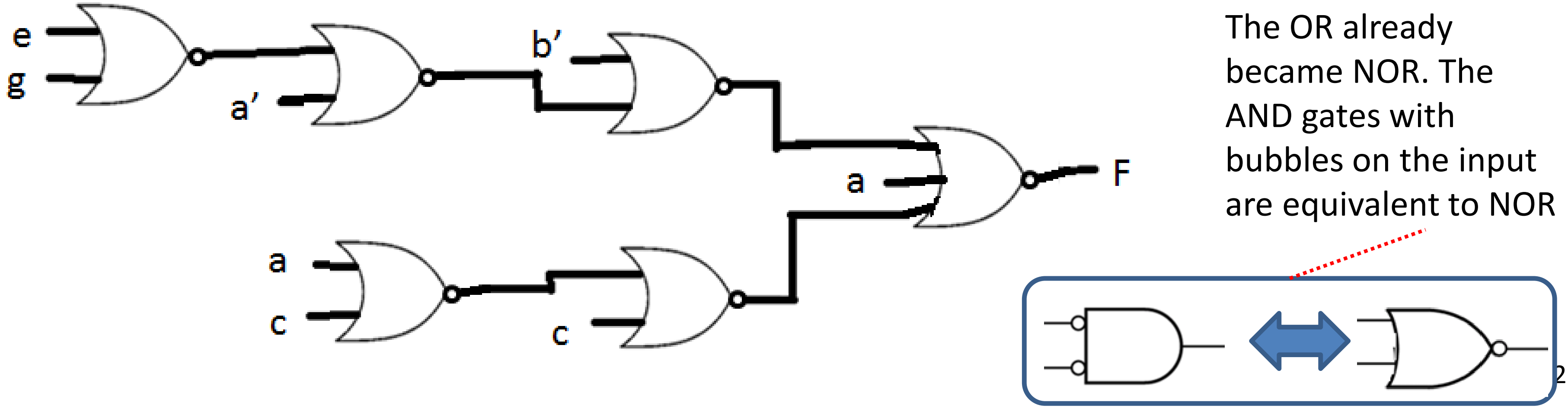


We need to transform this circuit to a NOR-only circuit





Add the bubble (inverters) on the output of OR and input of AND. Complement signals on inputs of AND.



The OR already became NOR. The AND gates with bubbles on the input are equivalent to NOR

Multiple-Output Circuits

- We have these three functions

  $F_1(A, B, C, D) = \Sigma\, m(11, 12, 13, 14, 15)$

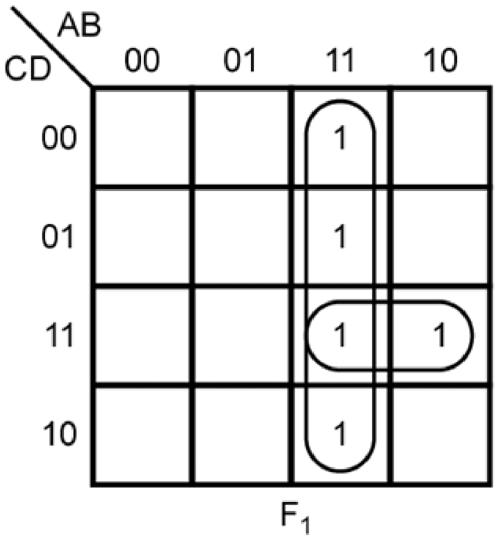
  $F_2(A, B, C, D) = \Sigma\, m(3, 7, 11, 12, 13, 15)$

  $F_3(A, B, C, D) = \Sigma\, m(3, 7, 12, 13, 14, 15)$

- We need to draw a circuit to implement them

Minimize every function separately

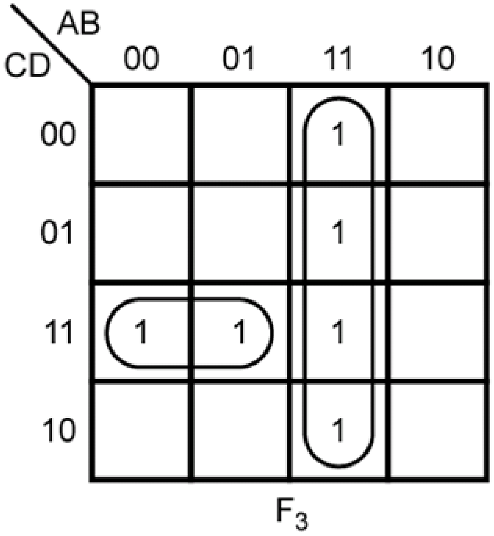
$F_1(A, B, C, D) =$
$\Sigma\, m(11, 12, 13, 14, 15)$



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

F_1

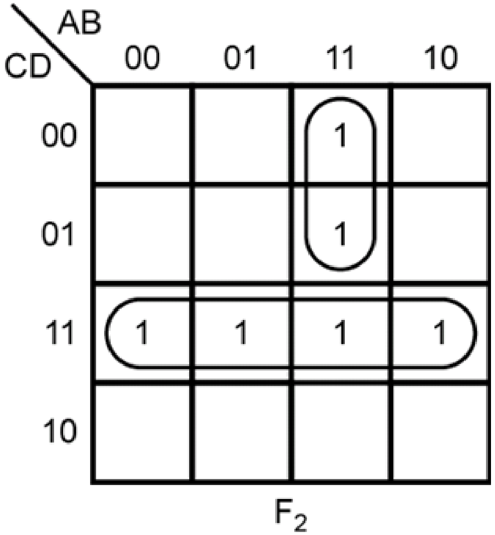
$F_3(A, B, C, D) =$
$\Sigma\, m(3, 7, 12, 13, 14, 15)$



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

F_3

$F_2(A, B, C, D) =$
$\Sigma\, m(3, 7, 11, 12, 13, 15)$

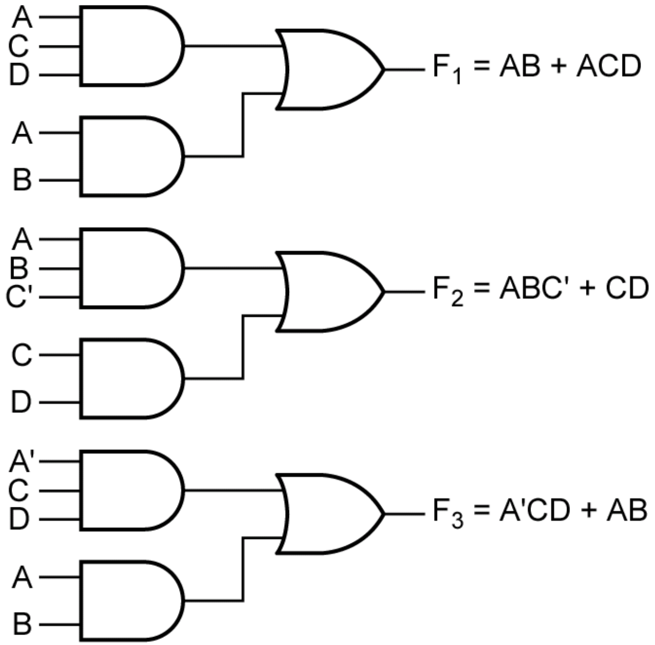


CD \ AB	00	01	11	10
00			1	
01			1	
11	1	1	1	1
10				

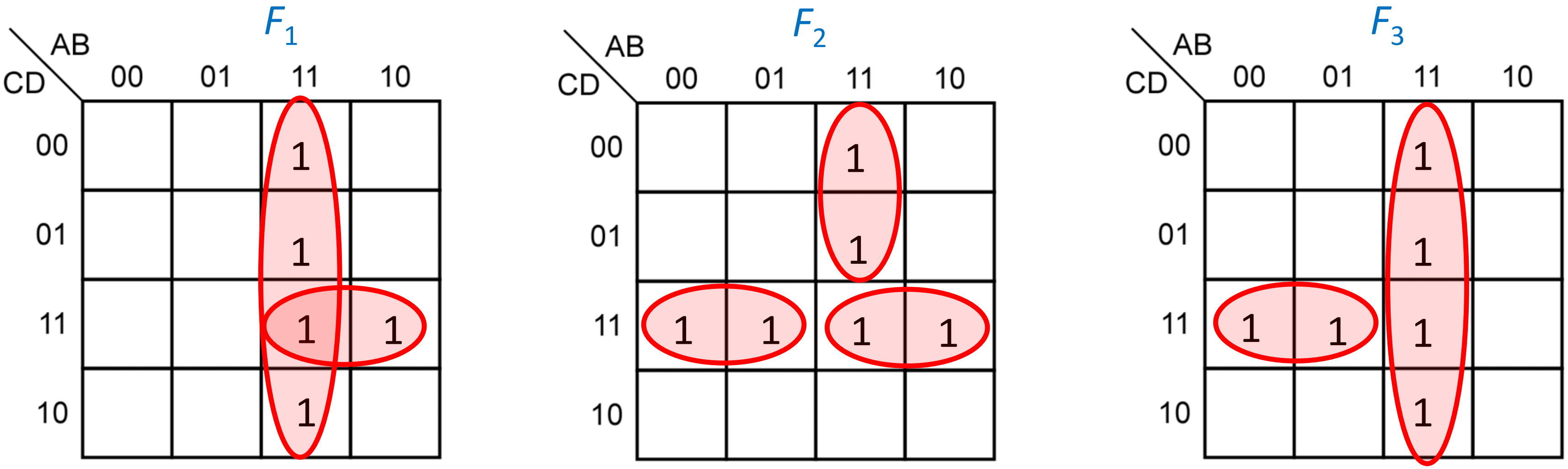
F_2

Number of gates: 9

Total # inputs: 21



Consider common terms



F_1

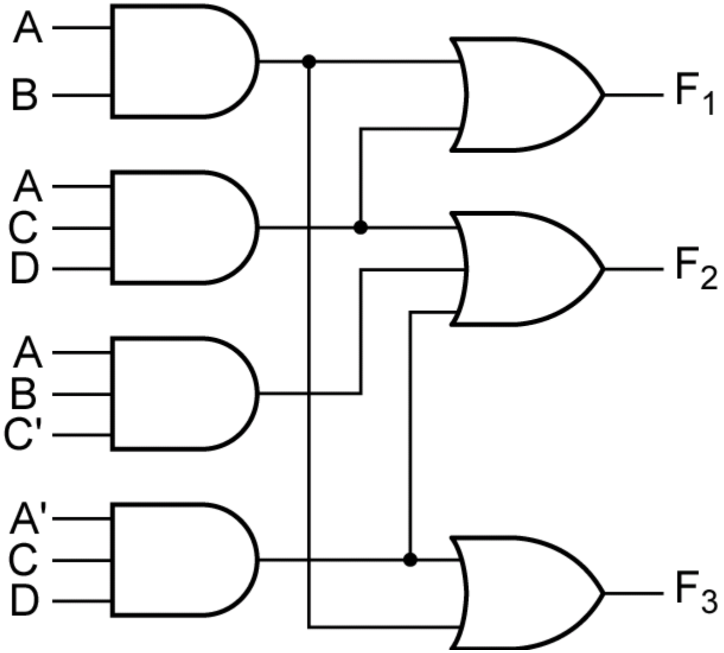
CD \ AB	00	01	11	10
00			1	
01			1	
11			1	1
10			1	

F_2

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

F_3

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



Number of gates: 7
Total # inputs: 18

Previously,
Number of gates: 9
Total # inputs: 21

Example 2

- We have these three functions

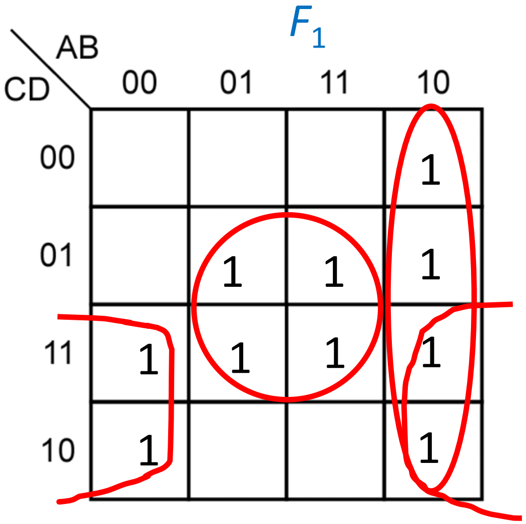
$F_1(A, B, C, D) = \Sigma\, m(2, 3, 5, 7, 8, 9, 10, 11, 13, 15)$

$F_2(A, B, C, D) = \Sigma\, m(2, 3, 5, 6, 7, 10, 11, 14, 15)$

$F_3(A, B, C, D) = \Sigma\, m(6, 7, 8, 9, 13, 14, 15)$

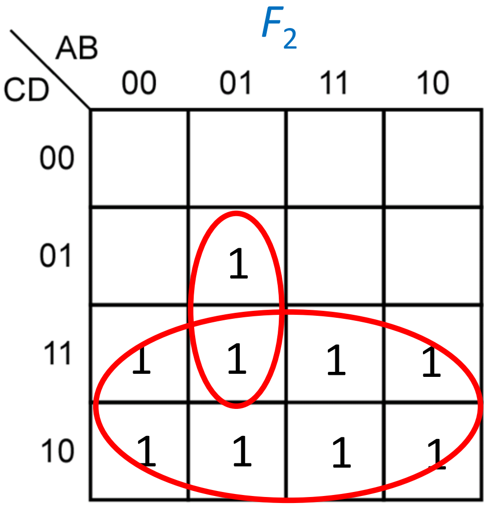
- We need to draw a circuit to implement them

Minimize every function separately



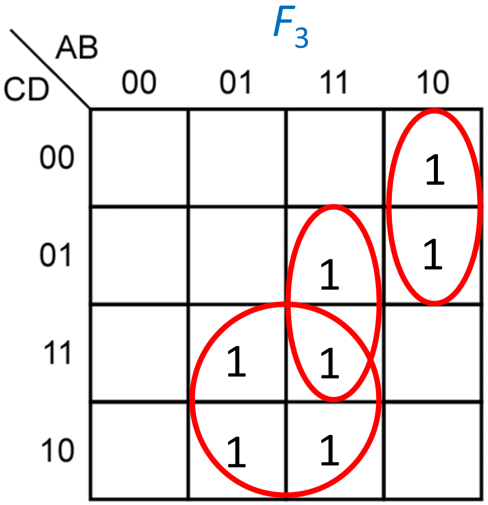
F_1

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



F_2

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



F_3

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

F_1 = AB' + BD + B'C

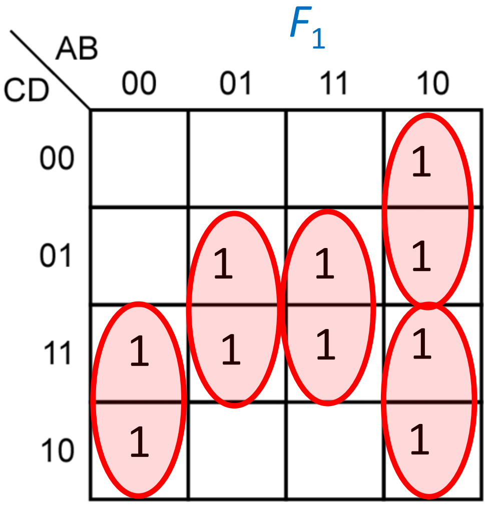
F_2 = A'BD + C

F_3 = AB'C' + ABD + BC

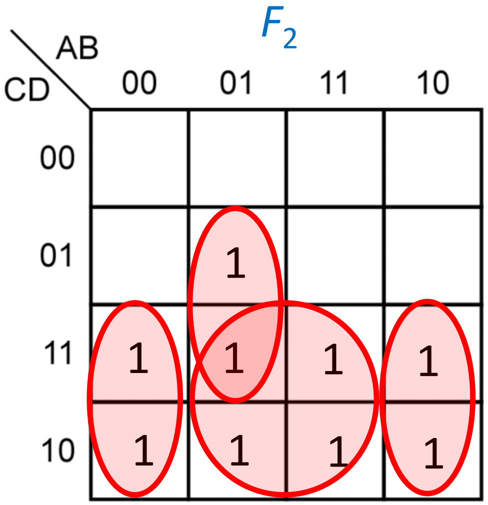
Number of gates: 10

total number of inputs: 25

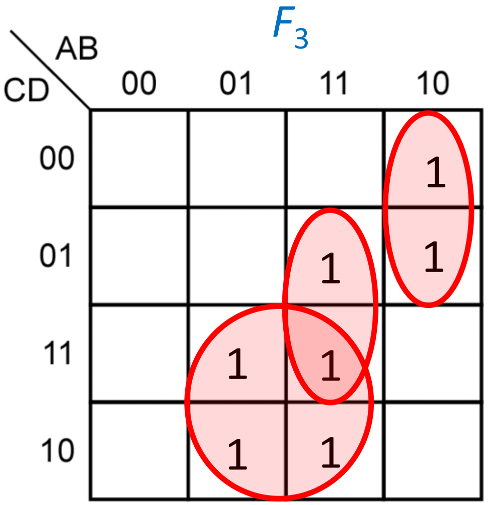
Consider common terms



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



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



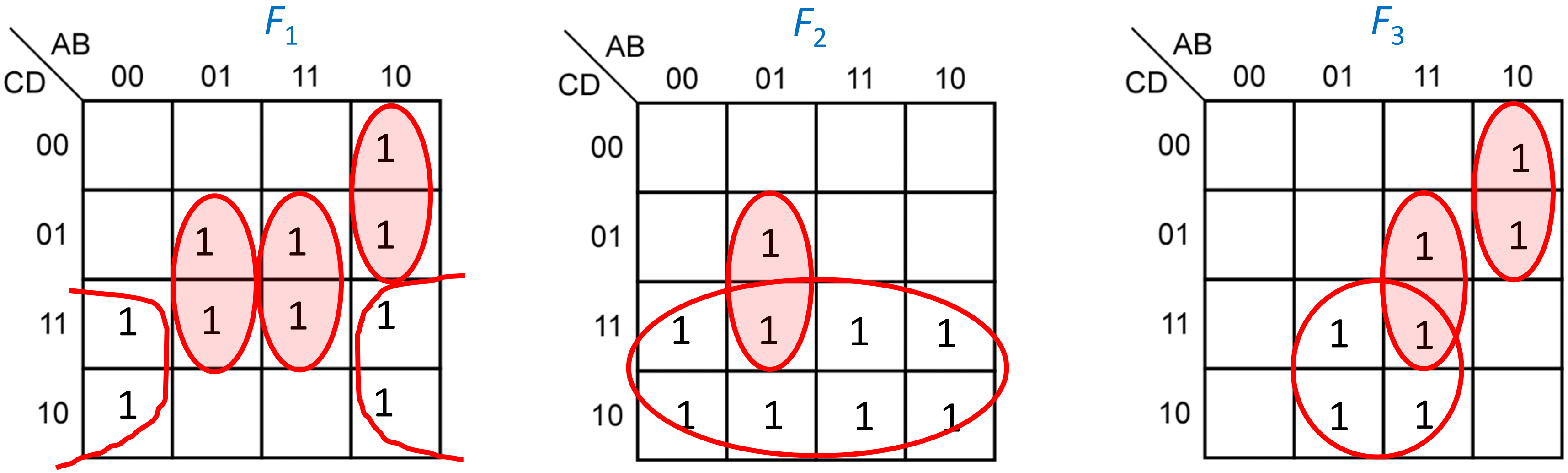
CD \ AB	00	01	11	10
00				1
01			1	1
11		1	1	
10		1	1	

F_1 = AB’C’ + A’BD + ABD + A’B’C + AB’C
F_2 = A’BD + A’B’C + BC + AB’C
F_3 = AB’C’ + ABD + BC

Number of gates: 9
total number of inputs: 29

It turns out there is a better solution.

Consider common terms



F_1

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

F_2

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

F_3

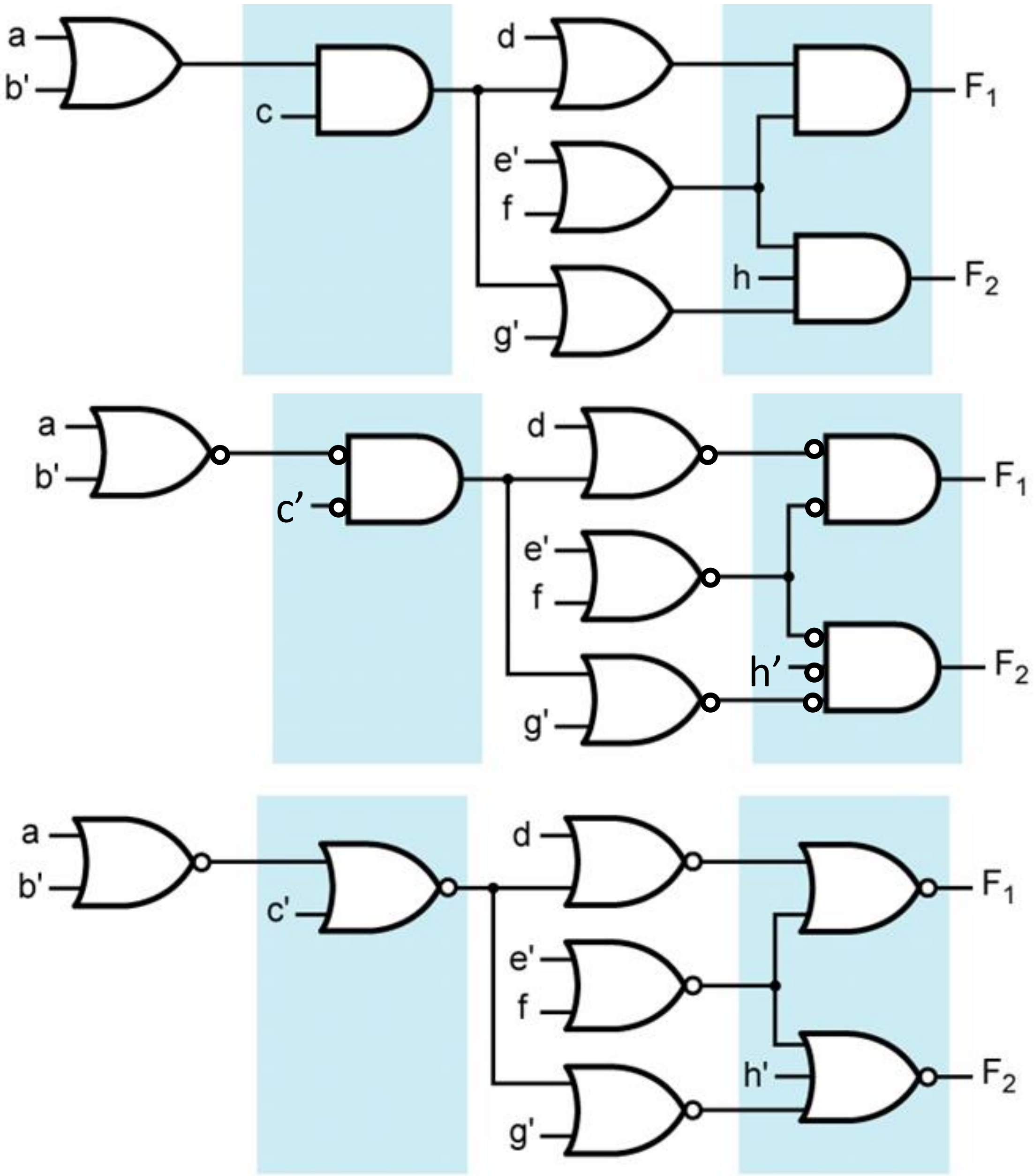
CD \ AB	00	01	11	10
00				1
01			1	1
11		1	1	
10		1	1	

#gates: 8, total # inputs: 22.

Taking common terms does not guarantee obtaining the least cost implementation, but sometimes it helps.

In this class, we do not cover the method to obtain the minimum hardware cost for the multiple-output function. Trial-and-error can be made to find a function with lesser hardware cost.

Convert Multiple-Output Function to NOR-only Circuit



Start by putting bubbles at the output of OR gates.

Put a bubble at the input of the AND gates. Complement the signals at the input of AND gates.

The AND gate with bubbles on the input can be replaced with a NOR gate.