COM 205 - Digital Logic Design
Boolean Algebra and Logic Gates -VI

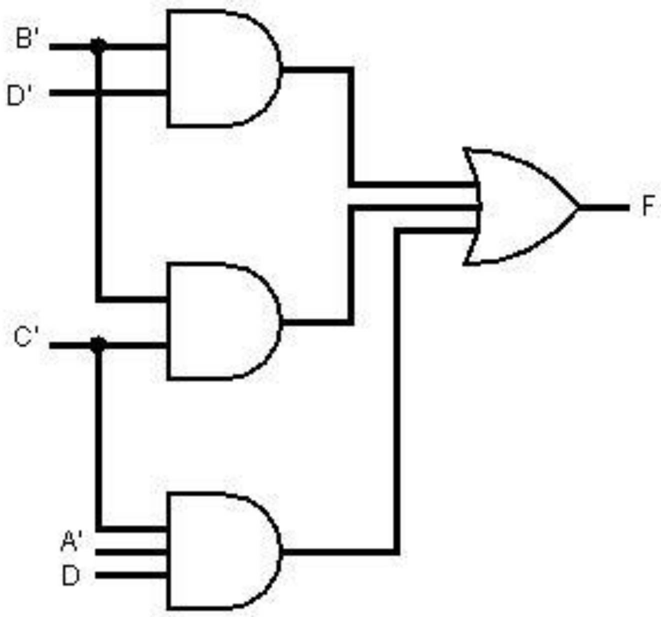
Assist. Prof. Özge ÖZTİMUR KARADAĞ

ALKÜ

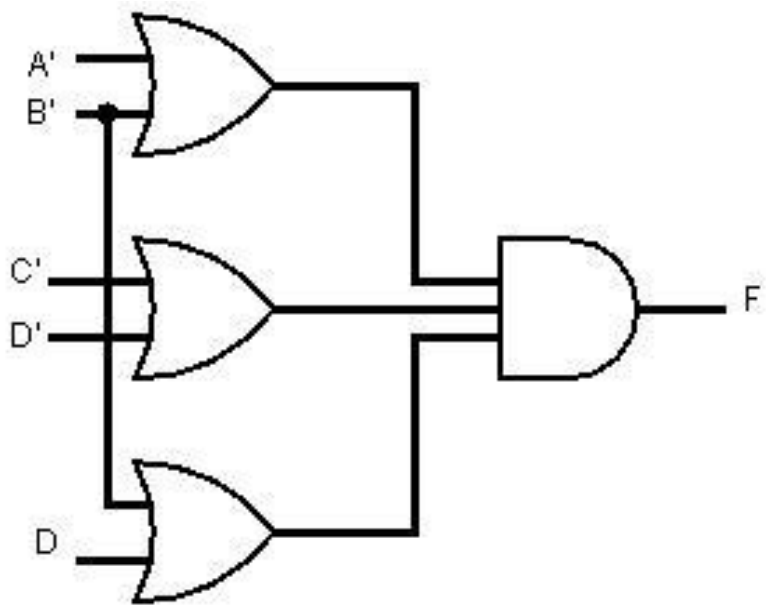
Last Week

- Implementation of the function in a standard form is said to be a two-level implementation

a. F=B’D’+B’C’+A’C’D

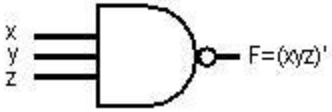


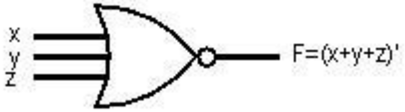
b. F=(A’+B’)(C’+D’)(B’+D)



Last Week..

- Two Level implementations:
  - NOR
  - NAND





Last Week..

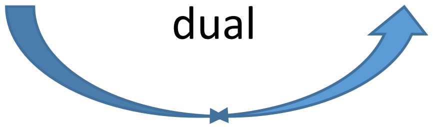
Rules for NAND and NOR Implementations

	Function to be simplified	Standard form to be used	How to obtain	Implementation	#levels for F
a	F	SOP	Grouping of 1s in map	NAND	2
b	F’	SOP	Grouping of 0s in map	NAND	3
c	F	POS	Complement of F’ in (b)	NOR	2
d	F’	POS	Complement of F in (a)	NOR	3

Other Two Level Implementations

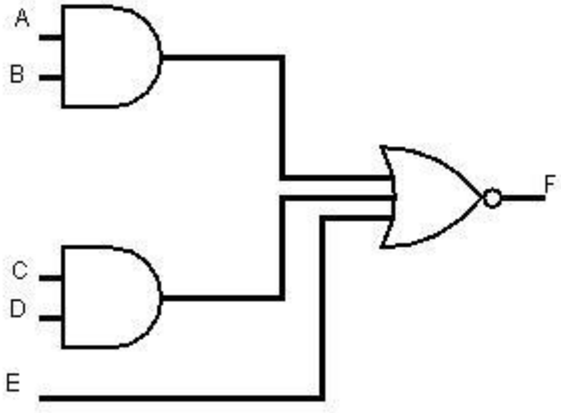
- Gate types: AND, OR, NOR, NAND
- Using one type of gate in the first level and one type of gate in the second level, it is possible to form 16 combinations.
- Among those combinations, ones which have the standard form are as follows:

• AND-OR	OR-AND
• NAND-NAND	NOR-NOR
• NOR-OR	NAND-AND
• OR-AND	AND-NOR

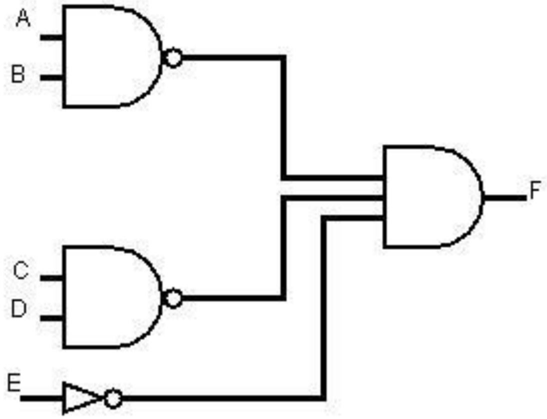


AND-OR-INVERT Implementation

- AND-NOR, NAND-AND
- F=(AB+CD+E)’



AND-NOR

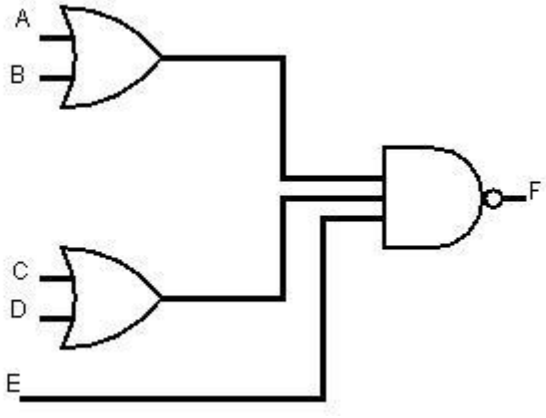


NAND-AND

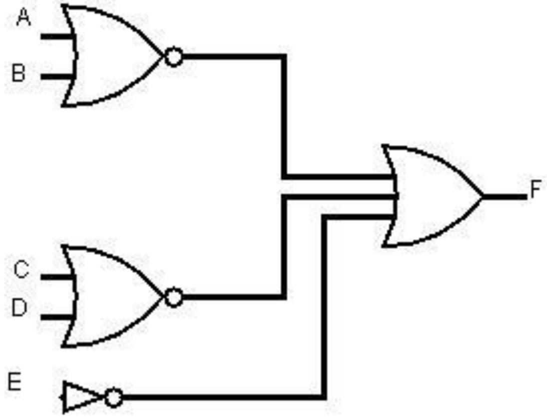
- Complement of the function can be represented in sum of products form by using 0s in the diagram.
- F’ is implemented in AND-OR form (Sum of products)
- F’ forms F by the invert operator in the end.

OR-AND-INVERT Implementation

- OR-NAND and NOR-OR forms both implement OR-AND-INVERT operation.
- F=[(A+B)(C+D)E]’



OR-NAND



NOR-OR

- OR-AND-INVERT requires a product of sums representation
- Simplify F’ in product of sums form
- Obtain F by inverting F’.

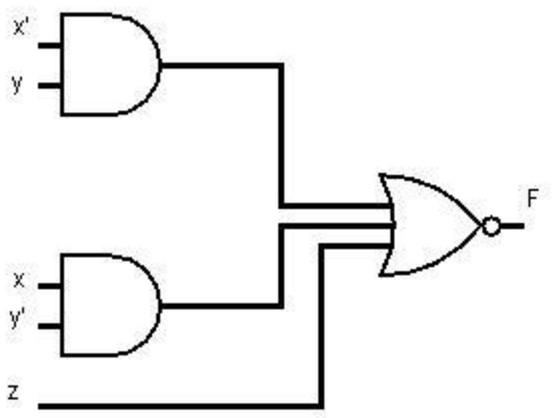
Equivalent Forms	Implements the function	Simplify F’ into	To get an output of
AND-NOR NAND-AND	AND-OR-INVERT	Sum of products form by combining 0s in the map	F
OR-NAND NOR-OR	OR-AND-INVERT	Product of sums form by combining 1s in the map and then complementing	F

- Ex: F(x,y,z)=$\sum(0,6)$ implement the function with the four 2 level forms.

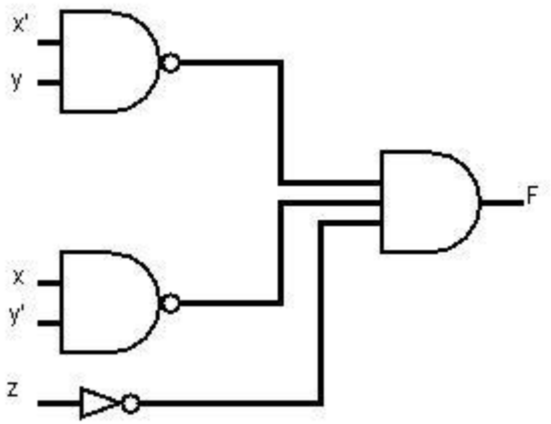
	00	01	11	10
0	1	0	0	0
1	0	0	0	1

AND-OR-INVERT ?

- F’=x’y+xy’+z
- F=(x’y+xy’+z)’



AND-NOR

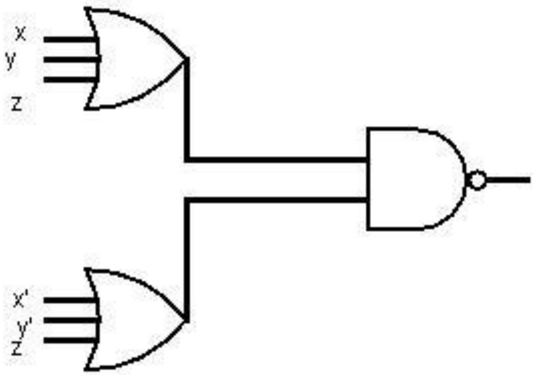


NAND-AND

- Ex: $F(x,y,z)=\sum(0,6)$

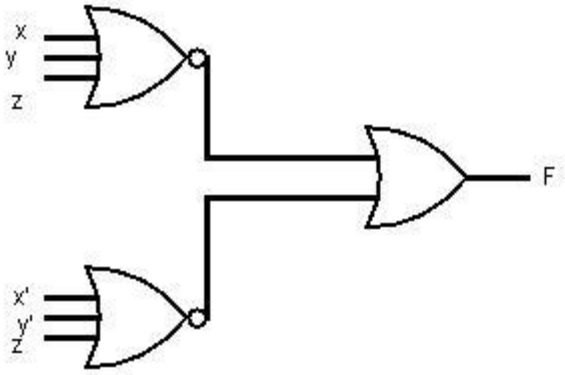
	00	01	11	10
0	1	0	0	0
1	0	0	0	1

OR-AND-INVERT ?



OR-NAND

- $F=x'y'z'+xyz'$ (using 1s in the diagram)
- $F'=[(x+y+z)(x'+y'+z)]'$



NOR-OR

DON’T CARE CONDITIONS

- Upto now, it is assumed that the minterms which are not equal to 1 are equal to 0. In practice, in some applications the function is not specified for certain combinations of variables.
- Ex: Four bit Binary code for the decimal digits has six combinations that are not used and are considered to be unspecified.
- Functions that have unspecified outputs for some input combinations are called incompletely specified functions.
- In most applications, we simply don’t care what value is assumed by the function for the unspecified minterms.
- For this reason, those minterms are called don’t care conditions.
- These conditions are represented with X’s in the diagram. X can take value either 0 or 1.

- Ex. Don’t care conditions d(w,x,y,z)=$\sum(0,2,5)$
- F(w,x,y,z)=$\sum(1,3,7,11,15)$ Simplify the function:

	00	01	11	10
00	X	1	1	X
01		X	1	
11			1	
10			1	

F=yz+w’x’

F=$\sum(0,1,2,3,7,11,15)$

	00	01	11	10
00	X	1	1	X
01		X	1	
11			1	
10			1	

F=yz+w’z

F=$\sum(1,3,5,7,11,15)$

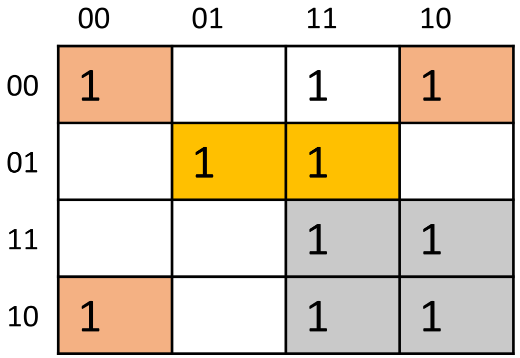
To represent the function in product of sums form:

F’=z’+wy’

F=z(w’+y)=$\sum(1,3,5,7,11,15)$

Exercise

- Find all prime implicants for the following Boolean function and determine which are essential?
- F(A,B,C,D)=$\sum(0,2,3,5,7,8,10,11,14,15)$



	00	01	11	10
00	1		1	1
01		1	1	
11			1	1
10	1		1	1

- Prime implicants: B’D’, B’C, CD, AC, A’BD
- Essential: B’D’, AC, A’BD
- Non-essential: CD, B’C
- F= B’D’+AC+A’BD +CD
- F= B’D’+AC+A’BD+ B’C

Exercise

- Find all prime implicants for the following Boolean function and determine which are essential?
- F(A,B,C,D)=$\sum(0,2,3,5,7,8,10,11,14,15)$

	00	01	11	10
00	1		1	1
01		1	1	
11			1	1
10	1		1	1

- Essential: B’D’, AC, A’BD
- Non-essential: CD, B’C
- F=B’D’+AC+A’BD+(CD or B’C)

- Simplify the following Boolean Function, Express the simplified function in sum of minterms form.
- F(A,B,C,D)=$\sum(0,6,8,13,14)$
- d(A,B,C,D)=$\sum(2,4,10)$

	00	01	11	10
00	1			X
01	X			1
11		1		1
10	1			X

- F=ABC’D+B’D’+CD’
- F=$\sum(0,2,6,8,10,13,14)$

- Simplify the following Boolean Function, Express the simplified function in sum of minterms form.
- F(A,B,C,D)=$\sum(0,6,8,13,14)$
- d(A,B,C,D)=$\sum(2,4,10)$

	00	01	11	10
00	1			X
01	X			1
11		1		1
10	1			X

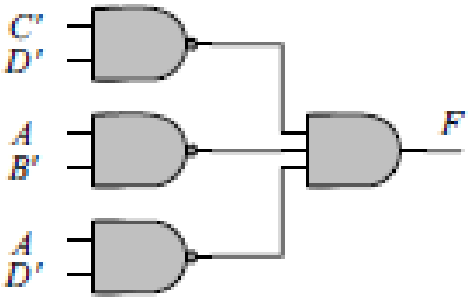
- F=B’D’+BCD’+ ABC’D
- F=$\sum(0,2,6,8,10, 13,14)$

- Implement the following Boolean function F, using two-level forms of logic NAND-AND, AND-NOR, OR-NAND, NOR-OR
- $F(A,B,C,D)=\prod(1,2,3,5,6,7,13,15)$

	00	01	11	10
00	1			
01	1			
11	1			1
10	1	1	1	1

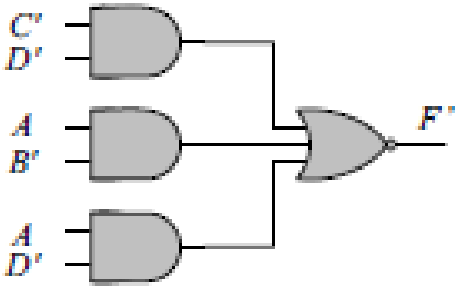
- F=C’D’+AB’+AD’

NAND-AND ?



F=C’D’+AB’+AD’
F’=(C’D’)’(AB’)’(AD’)’
NAND-AND

AND-NOR

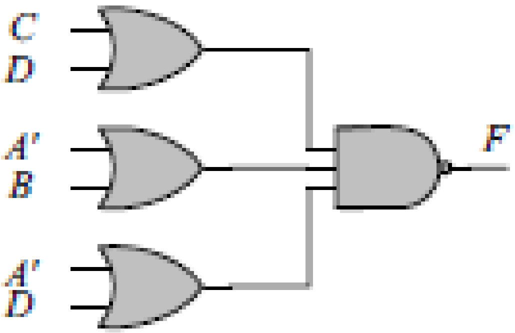


F’=[C’D’+AB’+AD’]’
AND-NOR

- $F=C'D'+AB'+AD'$

	00	01	11	10
00	1			
01	1			
11	1			1
10	1	1	1	1

OR-NAND ?

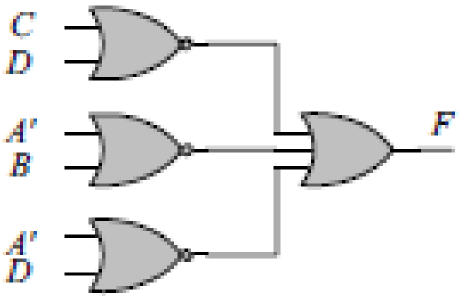


$F'=(C+D)(A'+B)(A'+D)$

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

OR-NAND

NOR-OR ?



$F=C'D'+AB'+AD'$

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

NOR-OR