ENCS 2340 Summary Chapter 2

By : Malak Obaid

Chapter 2 : Boolean Algebra

* basic operations

* And denoted by ($\cdot$) $x \cdot y$ x and y
* or denoted by (+) $x+y$ x or y
* not denoted by ($'$ or $^{-}$) x' or $\bar{x}$ (the complement of x)

x	y	$x \cdot y$
0	0	0
0	1	0
1	0	0
1	1	1

x	y	$x+y$
0	0	0
0	1	1
1	0	1
1	1	1

x	$\bar{x}$
0	1
1	0

افتراضات

Postulates of Boolean Algebra

1) closure the result is in $B = \{0, 1\}$

2) Identity with respect to + is 0 $x + 0 = 0 + x = x$
" " " " . is 1 $x \cdot 1 = 1 \cdot x = x$

تبديل
3) Commutative with respect to + $x + y = y + x$
" " " " . $x \cdot y = y \cdot x$

توزيع
4) . is distributive over + $x \cdot (y + z) = (x \cdot y) + (x \cdot z)$
+ " " " . $x + (y \cdot z) = (x + y) \cdot (x + z)$

5) for every x in B there is $\overline{x}$ called complement of x
such that $x + \overline{x} = 1$ and $x \cdot \overline{x} = 0$

Operator precedence "الأولويات"

* Expressions btw parentheses "أول شي"
* Not (')
* And (.)
* Or (+)

De Morgan's Theorem

$(x+y)' = x'y'$

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

Complemanting boolean functions:

* نعكس ال OR لـ AND والعكس صحيح
* نعكس اشارة كل variable

Ex Find the complement of each function

1) $f = x'yz + xy'z'$

$f' = (x+y'+z')(x'+y+z)$

2) $f = (a'+bc)d' + e$

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

Duality Principle :- مبدأ الازدواجية

* we interchanging btw and (·) , or (+)
  1's and 0's

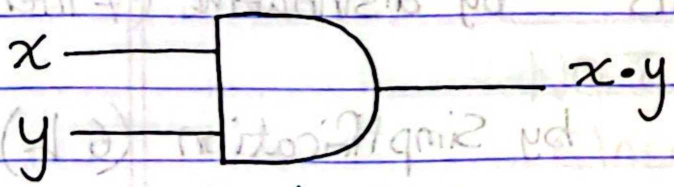
<u>Ex</u> The dual of $x(y+\bar{z}) \Rightarrow x + y\bar{z}$

Notes :-

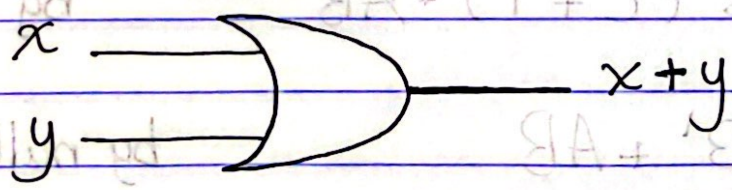
Logic 0 is a <u>low</u> voltage signals (around 0 V)

Logic 1 is a <u>high</u> voltage signal (5 or 3.3 V)

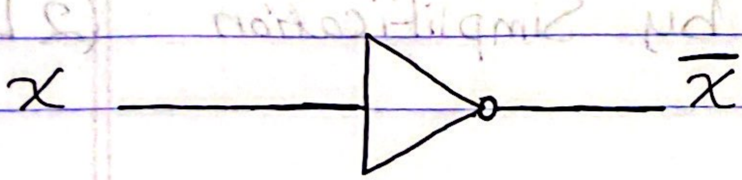
Logic Gates and symbols :-



and Gate



or Gate



Not Gate (Inverter)

Expression Simplification :-

((يعني أن نجعل الاقتران فيه أقل عدد ممكن literals))

<u>Ex</u> : Simplify the following boolean function to a minimum number of literals

1) $F(A,C) = (A+C)' + (A+C)(A'+C')$ (6 Literals)

$= (A+C)' + (A'+C')$ by simplification (4 literal)

$= A'C' + A' + C'$ by DeMorgan (4 literals)

$= A' + C'$ by absorption (2 literals)

2) $F(A,B,C) = A'B' + B'C + AB'C' + AB$ (9 Literals)

$= B'(A' + C + AC') + AB$ by distributive (7 literals)

$= B'(A' + C + A) + AB$ by Simplification (6 L)

$= B'(C+1) + AB$ by Complement (4 L)

$= B' + AB$ by null & identity (3 L)

$= B' + A$ by Simplification (2 L)

* اذا كان ال function لهذه الصورة

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

نعمل له dual ليصبح بهذا الشكل

$A'B'C' + AC' + BC' + B'C$

ثم نعمل باستخدام القوانين كالأمثلة السابقة
وعند الوصول لأقل عدد من الرموز نعمل له dual مرة أخرى.

* لايجاد ال complement of a function نستطيع
ذلك من خلال ايجاد ال dual له ثم عمل ال complement
لكل رمز

Ex

1) What is the complement of $f = x'yz' + xy'z'$

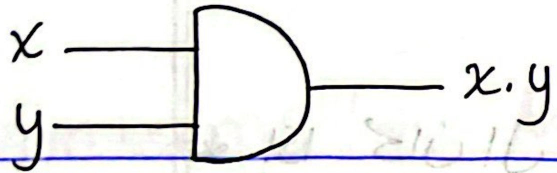
The dual $(x'+y+z')(x+y'+z')$

complement of each literal $(x+y'+z)(x'+y+z) = f'$

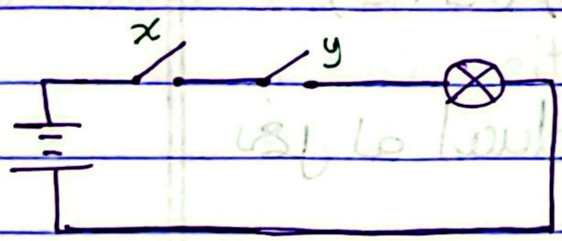
2) Find the complement of $f = (a'+bc)d'+e$

The dual $((a'.(b+c))+d').e$

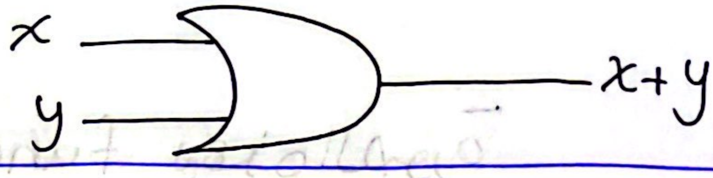
The complement $((a.(b'+c'))+d).e' = f'$



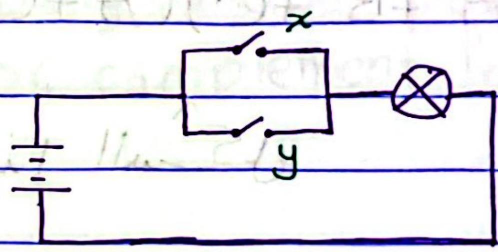
And Gate



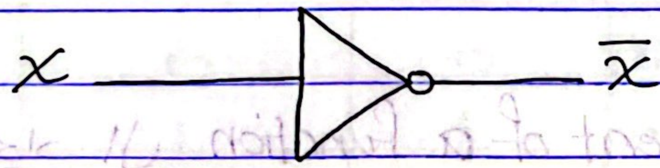
AND switches in series (0 is open)



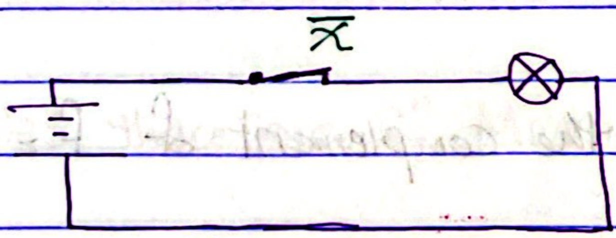
Or Gate



OR switches in Parallel (0 is open)

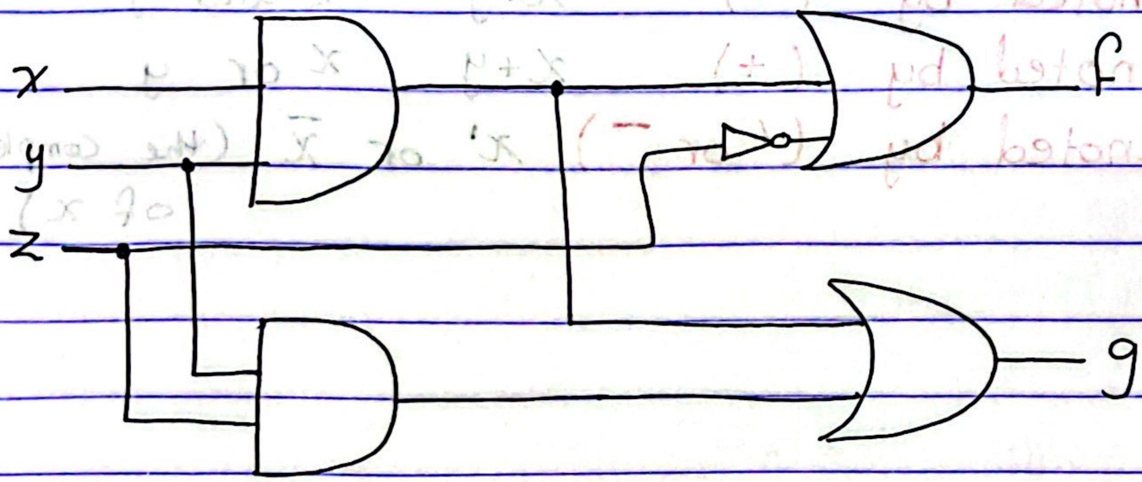


NOT Gate



NOT switch is normally Closed when x = 0

How to express the function from the circut



$f = xy + z'$

$g = xy + yz$

Minterms & Maxterms.

Maxterms : OR terms with every variable present in either true or complement form

Minterms : AND terms with every variable present in either true or complement form

x	y	indix	Min	Max
0	0	0	$m_0 = x'y'$	$M_0 = x+y$
0	1	1	$m_1 = x'y$	$M_1 = x+y'$
1	0	2	$m_2 = xy'$	$M_2 = x'+y$
1	1	3	$m_3 = xy$	$M_3 = x'+y'$

For n variables there are 2^n Minterms and Maxterms

☆ We note that Maxterm is the complement of the Minterm $M_i = m_i'$ and $m_i = M_i'$

for minterms : 1 means the variable is Not Complemented
0 " " " is Complemented

for maxterms : 0 means the variable is Not Complemented
1 " " " is Complemented

We Can represent the sum of Minterms with $\Sigma(\ ,\ )$

and represent the Sum of Maxterms with $\prod(\ \ )$

Sum of Minterms :

x	y	z	f	Minterms
0	0	0	0	
0	0	1	0	
0	1	0	1	$m_2 = x'yz'$
0	1	1	1	$m_3 = x'yz$
1	0	0	0	
1	0	1	1	$m_5 = xy'z$
1	1	0	0	
1	1	1	1	$m_7 = xyz$

$\Rightarrow$ we focus on the 1 entries

$f = m_2 + m_3 + m_5 + m_7 \Rightarrow \Sigma(2,3,5,7)$

$f = x'yz' + x'yz + xy'z + xyz$

* Express boolean function as sum of Minterm

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

Method 1 :- 1* بنشوف كل حد شو ناقصه رموز وبنزيدهن

* A is missing two variables

$A = A(B+B') = AB + AB'$ => also missing C so

$= AB(C+C') + AB'(C+C') = ABC + ABC' + AB'C + AB'C'$

* B'C is missing one variable (A)

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

2* نجمع كل الحدود معاً دون تكرار لأي حد .

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

$= m_7 + m_6 + m_5 + m_4 + m_1$

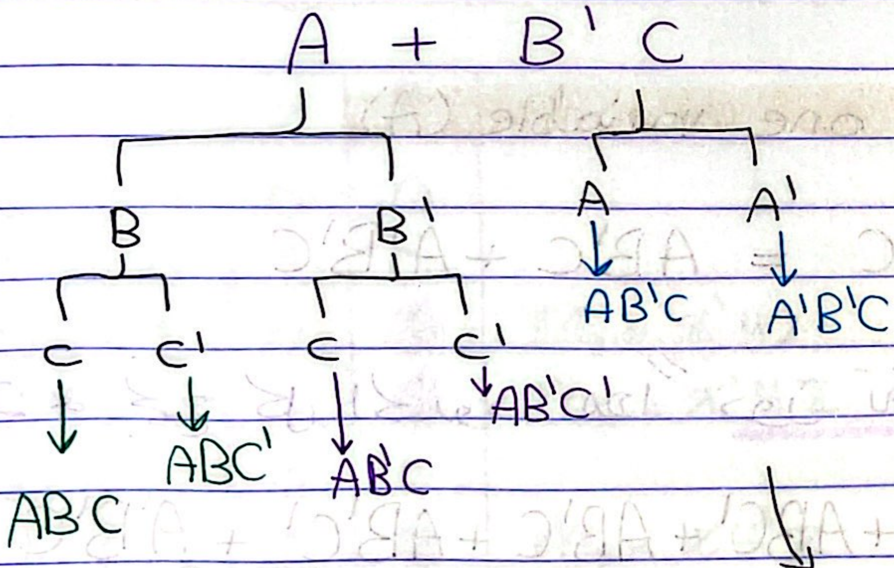
$F(A,B,C) = \sum(1,4,5,6,7)$

method 2 ⇒ truth table

method 3 (just for multiple choices)

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

*بنعملو زي شجرة النواقص



ثم نكتب النتائج دون تكرار ونكمل مثل method1

Another example :-

express $f(a,b,c,d) = \Sigma(2,3,6,10,11)$ in the sum of minterms form :-

$f(a,b,c,d) = m_2 + m_3 + m_6 + m_{10} + m_{11}$

$= a'b'cd' + a'b'cd + a'bcd' + ab'cd' + ab'cd$

"نكتب 8 4 2 1"

a b c d → وحسب الرقم عند m نضع ال complement

يعني m2 نضع اشارة ال complement فوق كلشي عدا ال 2

$a'b'cd'$

8 4 2 1

وهكذا نكمل في هذه الأمثلة ...

Product of Maxterms :-

x	y	z	f	Maxterm
0	0	0	0	$M_0 = x+y+z$
0	0	1	0	$M_1 = x+y+z'$
0	1	0	1	
0	1	1	1	
1	0	0	0	$M_4 = x'+y+z$
1	0	1	1	
1	1	0	0	$M_6 = x'+y'+z$
1	1	1	1	

We focus on the 0 entries

$$f = M_0 \cdot M_1 \cdot M_4 \cdot M_6$$

$$f = \prod(0,1,4,6)$$

$$f = (x+y+z)(x+y+z')(x'+y+z)(x'+y'+z)$$

Express the boolean functions as Product of Maxterms :-

$f(x,y,z) = xy + x'z$

distributive law باستخدام OR terms حول الحدود على شكل 1*

نعمل distribute كمان مرة

$f = xy + x'z = (xy+x')(xy+z) = \underbrace{(x'+x)}_{=1}(x'+y)(x+z)(y+z)$

$\therefore f = (x'+y)(x+z)(y+z)$

2* بنشوف كل حد شو ناقصه وبنضيفه ونعمل distribute

$f = (x'+y+zz')(x+yy'+z)(xx'+y+z)$

$f = (x'+y+z)(x'+y+z')(x+y+z)(x+y'+z)(x+y+z)(x'+y+z)$

3* نحذف الحدود المكررة ونكتب الشكل النهائي

$f = (x'+y+z)(x'+y+z')(x+y+z)(x+y'+z)$

$= M_4 \cdot M_5 \cdot M_0 \cdot M_2$

$f(x,y,z) = \Pi(0,2,4,5)$

*وكذلك عكسنا حلها بنفس الطرق السابقة لل minterms

Express $f(a,b,c,d) = \prod(1,3,11)$ in the product of Maxterms form:-

$f(a,b,c,d) = M_1 \cdot M_3 \cdot M_{11}$

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

*نفس مبدأ sum of minterms

8	4	2	1
a	b	c	d

لكن هنا المجموع للرقم عند M رح نضع ال complement

8	4	2	1	M_3	*
a	b	c	d	1+2	

اذا نضع ال Complement الهن

$(a+b+c'+d')$ تصبح

The same boolean function can be expressed in 2 ways :-

1) Sum of minterms $f = m_0 + m_2 + m_3 + m_5 + m_7 = \Sigma(0,2,3,5,7)$

2) Product of maxterms $f = M_1 \cdot M_4 \cdot M_6 = \Pi(1,4,6)$

نلاحظ انه الاثنين متممات لبعض

Function Complement

$f(x,y,z) = \Sigma(0,2,3,5,7) = \Pi(1,4,6)$

So the complement of the function f is f'

$f'(x,y,z) = \Pi(0,2,3,5,7) = \Sigma(1,4,6)$

* حتى نعمل ال complement لأي اقتران فقط نغير

$\Sigma \rightarrow \Pi$ or $\Pi \rightarrow \Sigma$

مع ابقاء نفس أرقام المؤشرات

Example :-

Write the complement of the following function as sum of minterms

$f(z,y,x) = \Sigma(0,2,3,4,6)$

بما أنه لدينا 3 متغيرات

اذا

the number of minterms and maxterms $= 2^3 = 8$

انتهينا بـ 7 لأننا نبدأ من index 0

$f(z,y,x) = \Sigma(0,2,3,4,6)$

$f'(z,y,x) = \Pi(0,2,3,4,6) = \Sigma(1,5,7)$

$= z'y'x + zy'x + zyx$

Operation on Functions :

* The And operation corresponds to the intersection of the 2 sets of minterms of the function

" التقاطع في العناصر (العناصر المشتركة) "

* The OR operation corresponds to the Union of the 2 sets of minterms of the function

" اتحاد لجميع العناصر للإقترانين " دون تكرار .

Example : $F(A,B,C) = \sum m(1,3,6,7)$ and
$G(A,B,C) = \sum m(0,1,2,4,6,7)$

$F \cdot G = \sum m(1,6,7)$

$F + G = \sum m(0,1,2,3,4,6,7)$

$F' \cdot G$ → to find it we have to find sum of minterms of F'

$F' = \sum m(0,2,4,5)$

$\therefore F' \cdot G = \sum m(0,2,4)$

Equal Functions:-

Two functions are equal if and only if they have the same Sum of minterms and same product of maxterms

Example 1:- if $F_1 = a'b' + ac + bc'$
$F_2 = a'c' + ab + b'c$

Are they equal to each other?

$F_1 = \sum m(0,1,2,5,6,7) = \prod(3,4)$

$F_2 = \sum m(0,1,2,5,6,7) = \prod(3,4)$

$\therefore$ equal

Ex 2:

Are $F_1(x,y,z) = \sum m(1,2,4,5,6,7)$ and $F_2(a,b) = \prod(0,3)$ equal?

* لازم ننتبه لعدد ال variables

$\therefore F_1 = \sum m(1,2,4,5,6,7) = \prod(0,3)$

$F_2 = \prod(0,3) = \sum m(1,2)$

$\therefore$ Not equal

Sum of Products and Products of Sums:

* Canonical forms contain a large number of literals
* We can express boolean functions in a standard form called Sum of Products and Product of sums

* Sum of Products (SOP)

*تكون على هذا الشكل

$f_1 = xy' + xz$ $\qquad$ $f_2 = y + xy'z$

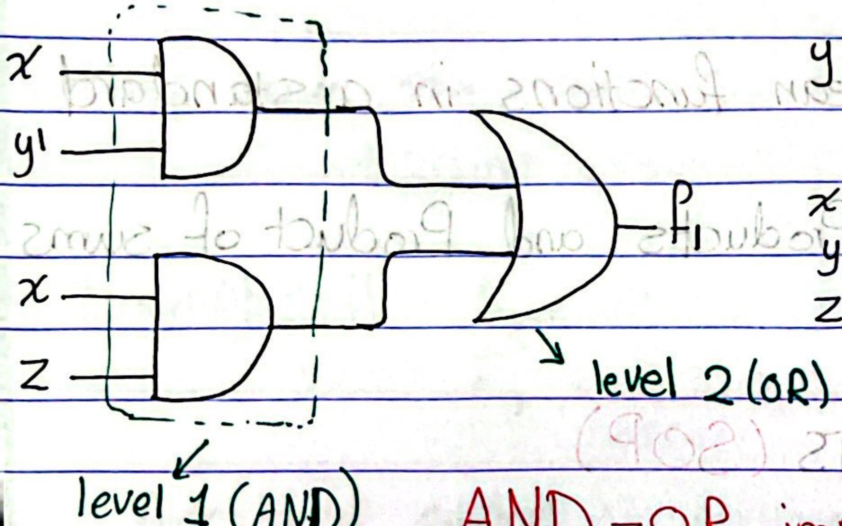
* Products of sums (POS)

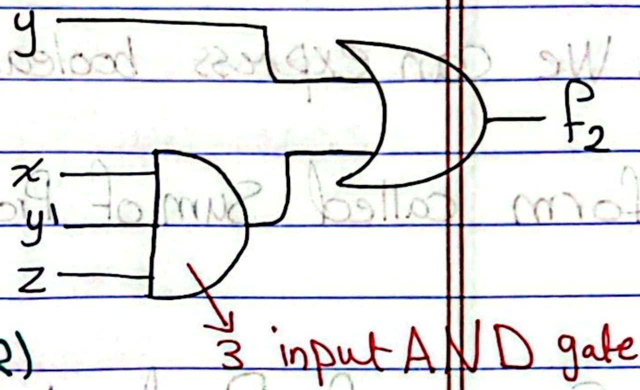
*تكون بهذا الشكل

$f_3 = (x+z).(x'+y')$ $\qquad$ $f_4 = x.(x'+y'+z)$

* Two-level Gate Implementation

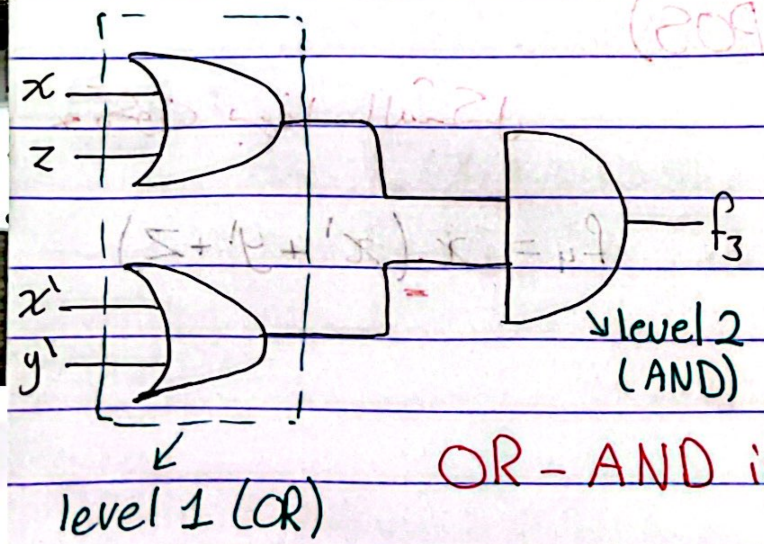
$f_1 = xy' + xz$ $\qquad$ $f_2 = y + xy'z$

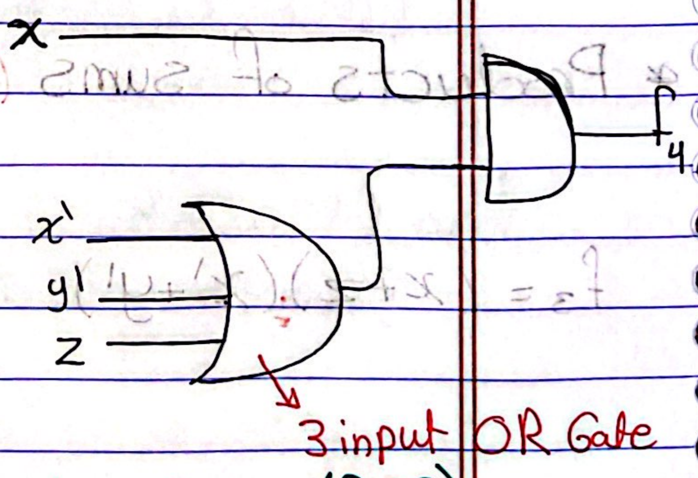




AND-OR implementation (SOP)

$f_3 = (x+z)(x'+y')$ $\qquad$ $f_4 = x(x'+y'+z)$





OR - AND implementation (POS)

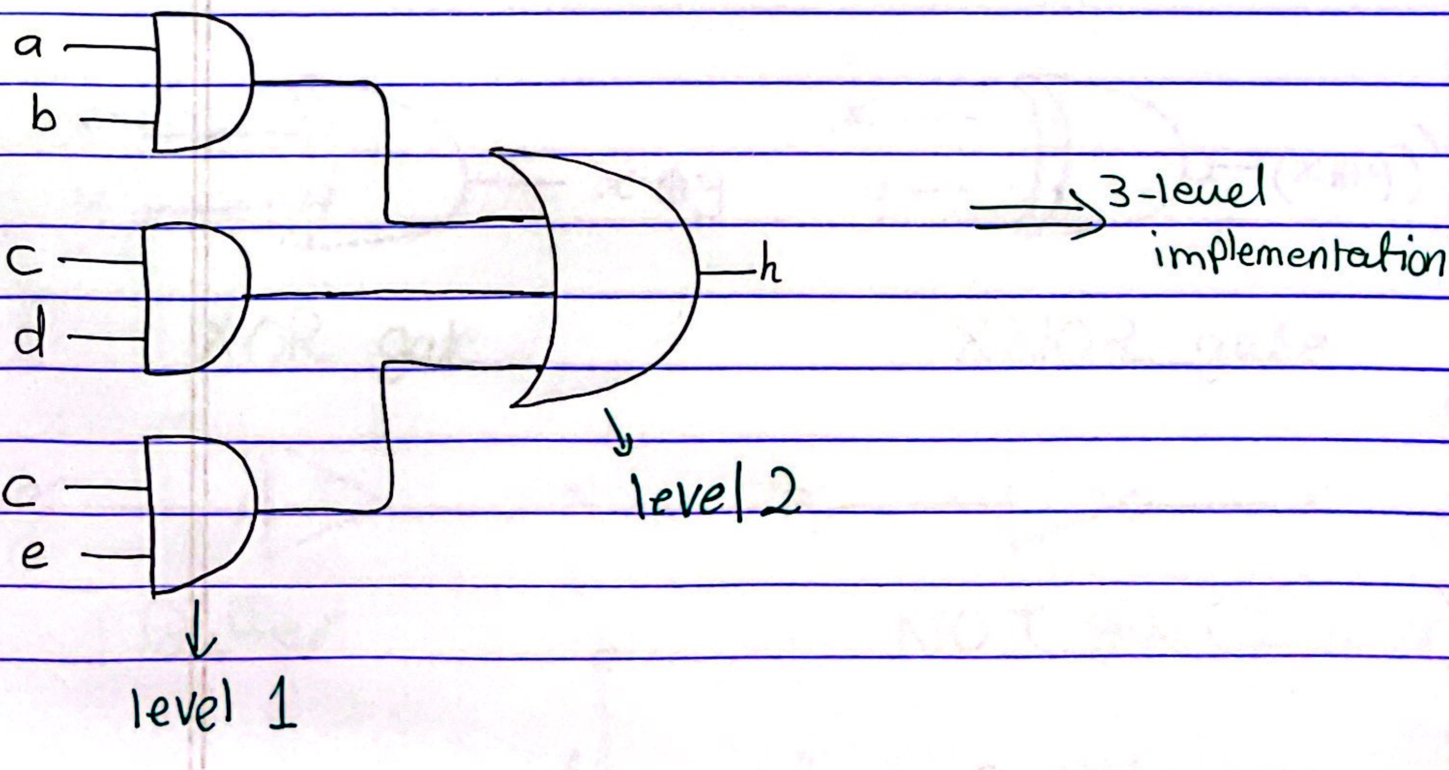
Two-Level VS Three-Level Implementation

If we have a function $h = ab + cd + ce$

* here it's a sum of products (6 literals) but we can write it like this $h = ab + c(d+e)$ but it's not in the standard form (not sum of products nor product of sum)

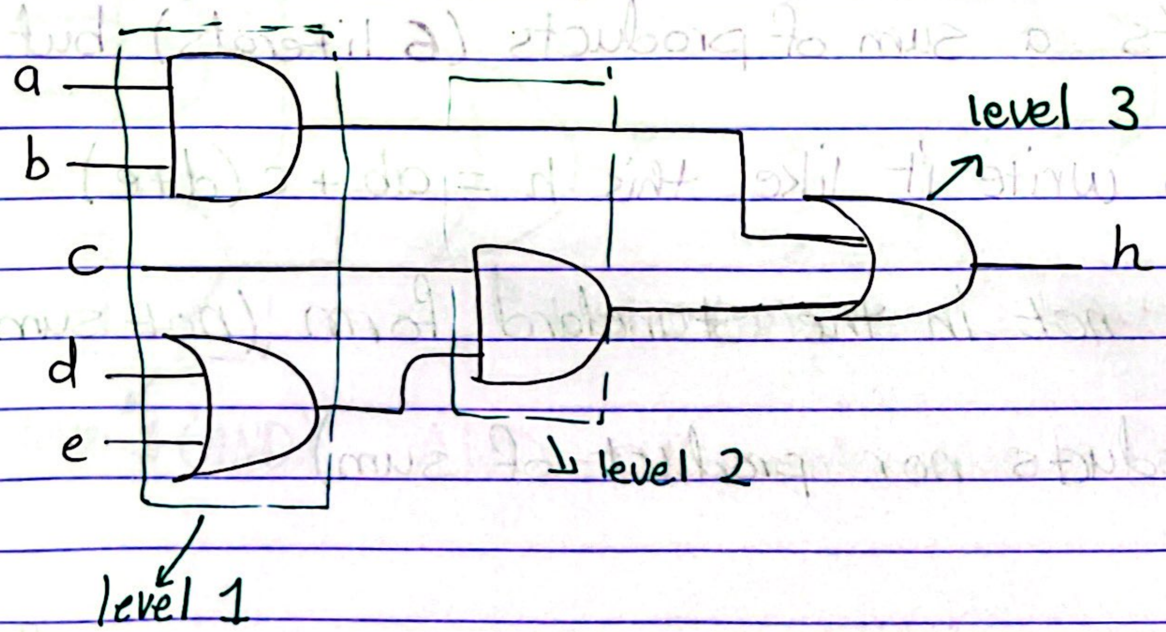
2-Level Implementation

$h = ab + cd + ce$



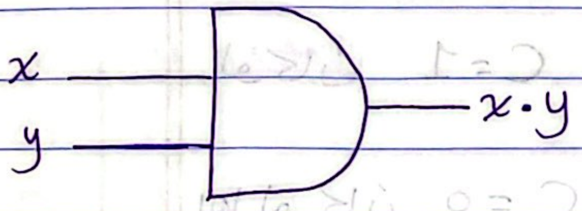
3-level implementation

$h = ab + c(d+e)$

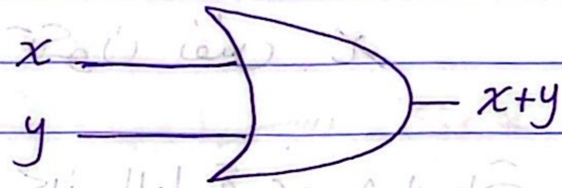


Additional Logic Gates and symbols :

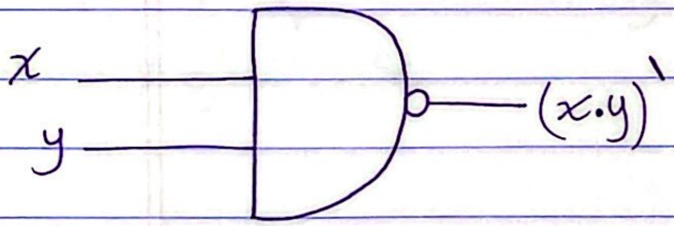
* لماذا كُنّا جبنه أو نستخدمهن ؟
* التكلفة أقل
* بمكنونا تسشي دائرات بعدد رموز أقل .



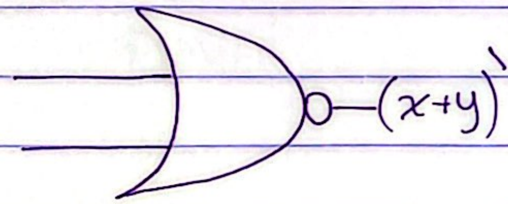
AND gate



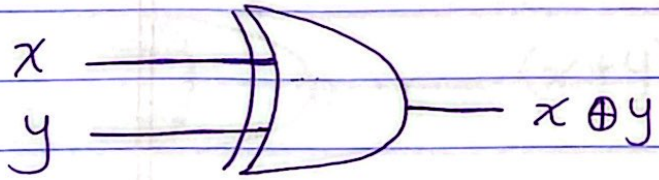
OR gate



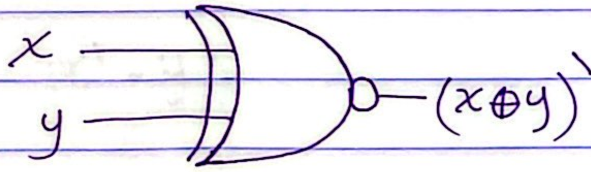
NAND gate



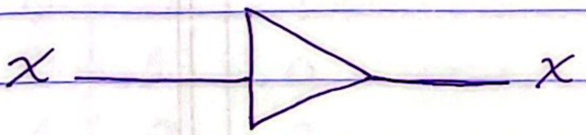
NOR gate



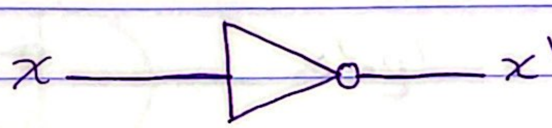
XOR gate



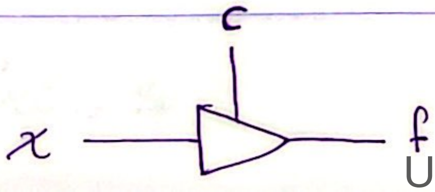
XNOR gate



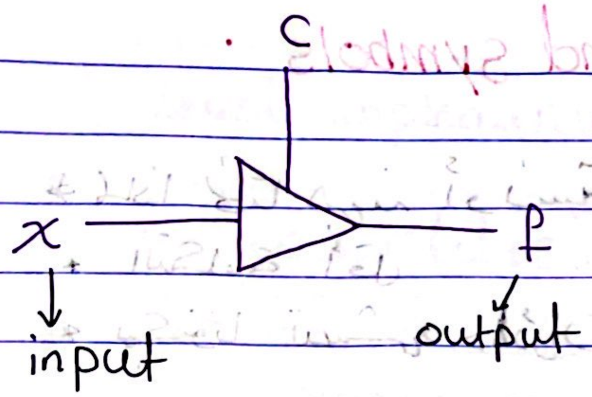
Buffer



NOT gate (Inverter)



$\Rightarrow$ 3-state gate



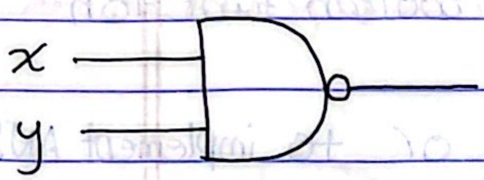
الذي يتحكم في هذه الدارة
هو c

لو كان C=1 ف f رح تكون نفس x

اما لو كان C=0 رح تصبح كان الدارة غير موصولة.

وال output غير موجود

NAND Gate (NOT AND)

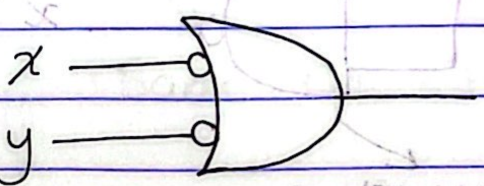


$(x.y)' = x'+y'$

x	y	NAND
0	0	1
0	1	1
1	0	1
1	1	0

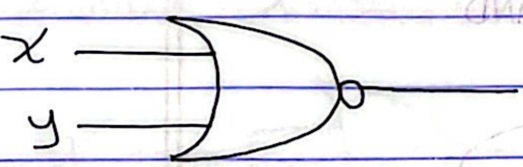
عكس نتائج ال AND

* يمكن رسمها بهذا الشكل أيضا .



$x'+y'$

NOR Gate (NOT OR)

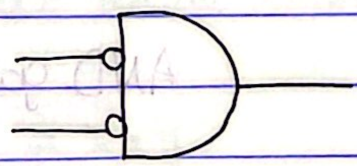


$(x+y)' = x'.y'$

x	y	NOR
0	0	1
0	1	0
1	0	0
1	1	0

عكس نتائج ال OR

* يمكن رسمها هكذا أيضا

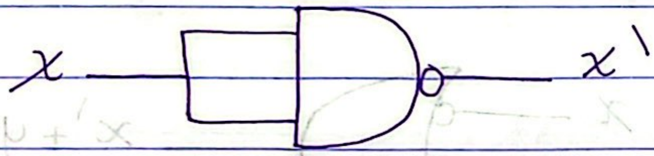


$x'.y'$

NAND Gate is Universal

* This gate can implement any boolean function
* also can be used as inverter or to implement AND/OR

NAND $x = (x \cdot x)' = x'$

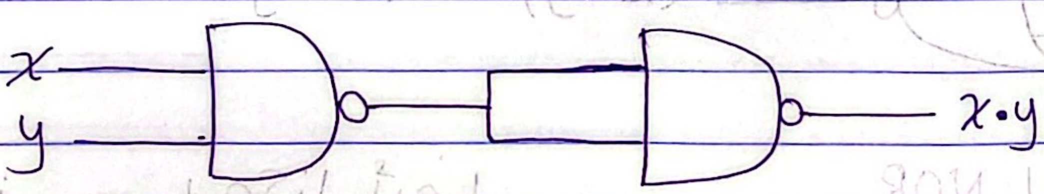


inverter

AND is equivalent to NAND with inverted output

$(x \text{ NAND } y)' = ((x \cdot y)')' = x \cdot y$

AND

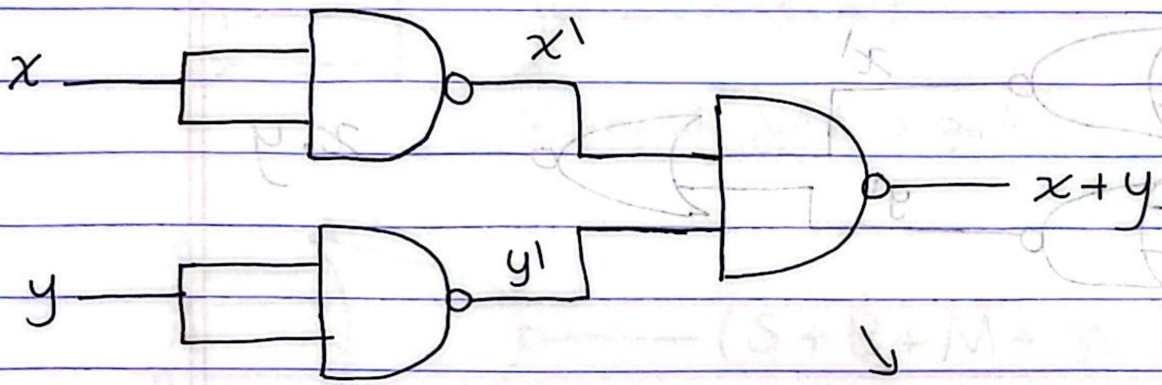


⇓

AND gate تكافئ

OR is equivalent to NAND with inverted inputs

$(x' \text{ NAND } y') = (x' \cdot y')' = \underbrace{x + y}_{\text{OR}}$

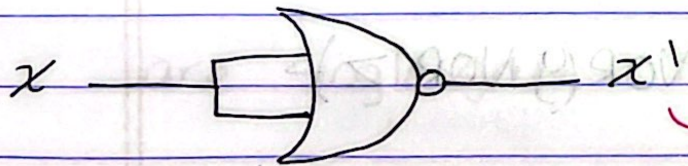


Same as OR Gate

NOR Gate is also Universal

→ has the same properties

$x \text{ NOR } x = (x + x)' = x'$

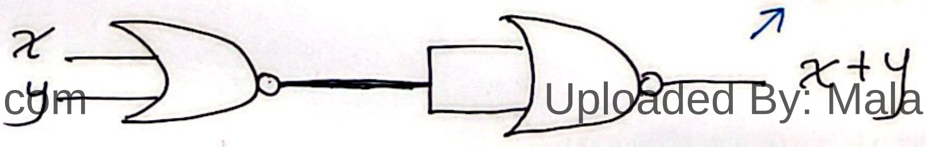


also inverter

OR is equivalent to NOR with inverted output

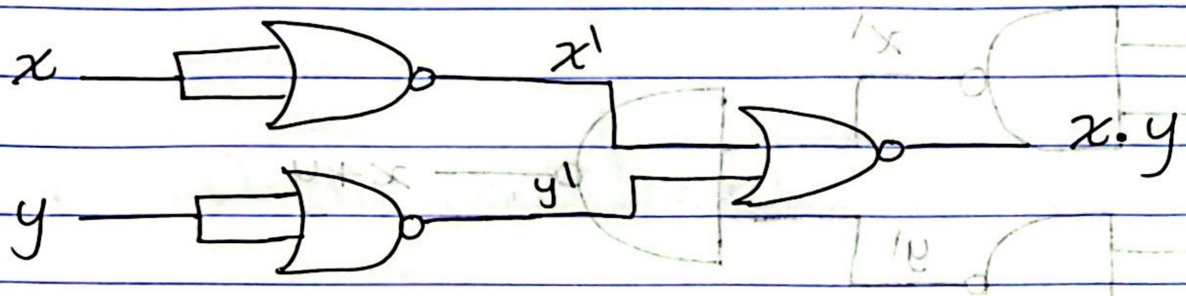
$(x \text{ NOR } y)' = ((x+y)')' = \underbrace{x + y}_{\text{OR}}$

same as OR gate



AND is equivalent to NOR with inverted inputs

$(x' \text{ NOR } y') = (x'+y')' = \underset{\text{AND}}{x \cdot y}$



NAND / NOR are Non-Associative

" ليست علاقة تجميعية "

$(x \text{ NAND } y) \text{ NAND } z \neq x \text{ NAND} (y \text{ NAND } z)$

because

$(x \text{ NAND } y) \text{ NAND } z = ((x \cdot y)' z)' = ((x'+y')z)' = xy + z'$

but

$x \text{ NAND} (y \text{ NAND } z) = (x(y \cdot z)')' = (x(y'+z'))' = x' + yz$

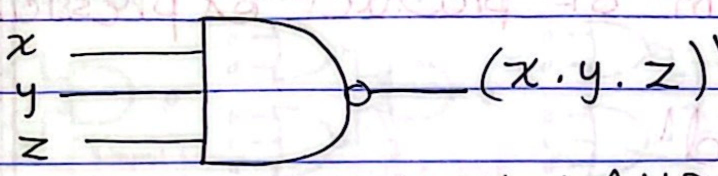
$(x \text{ NOR } y) \text{ NOR } z \neq x \text{ NOR} (y \text{ NOR } z)$

$(x \text{ NOR } y) \text{ NOR } z = ((x+y)' + z)' = ((x'y') + z)' = (x+y) z'$

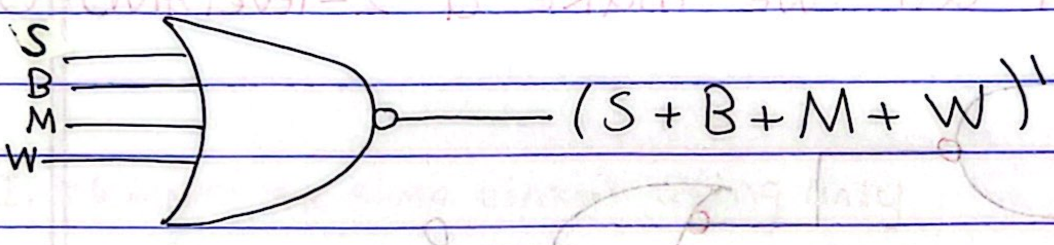
$x \text{ NOR} (y \text{ NOR } z) = (x + (y+z)')' = (x + (y'z'))' = x'(y+z)$

NAND / NOR Can have 2, 3, 4 - - - inputs

Ex



3-input NAND gate



4-inputs NOR gate

وممكن نعمل كذا عدد من ال inputs سواء NAND/NOR gates

Note that all mutiple-input NAND/NOR gates are single gates not a combination of 2-input gates.

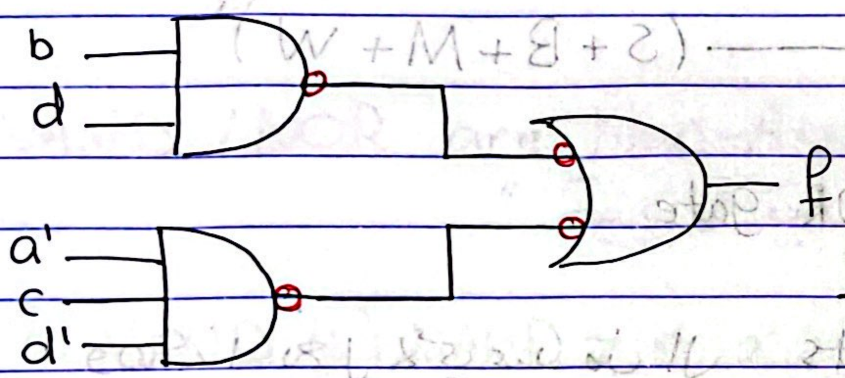
NAND - NAND Implementation

يعني هنا 2 level في كل وحده NAND

if we have sum of product expression like

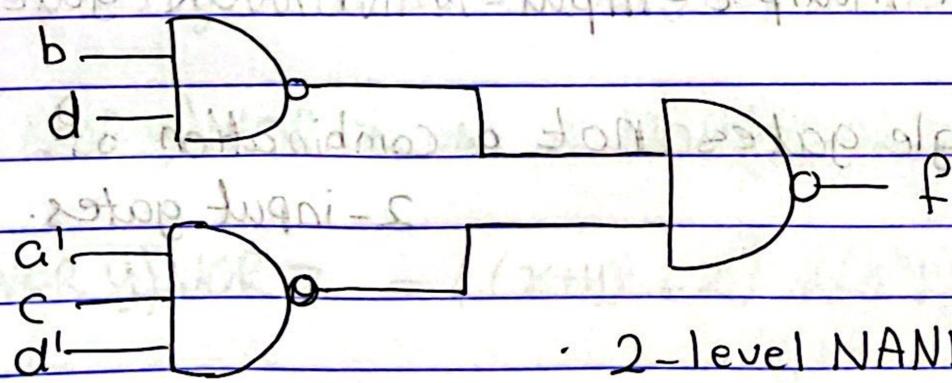
$f = bd + a'cd'$

* First of all we make a 2-level AND-OR circuit



2-level AND-OR

* then we add bubble on the same line (they cancel each other)



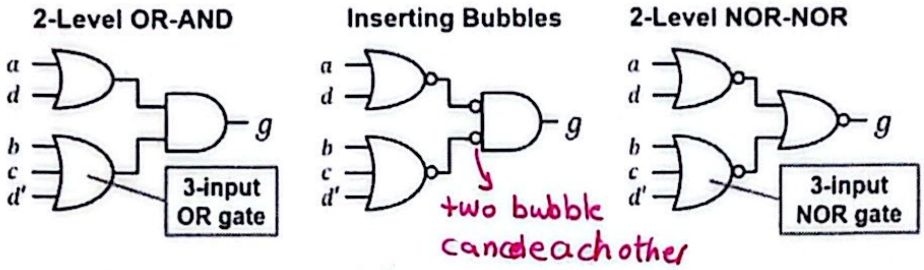
2-level NAND-NAND

NOR-NOR Implementation

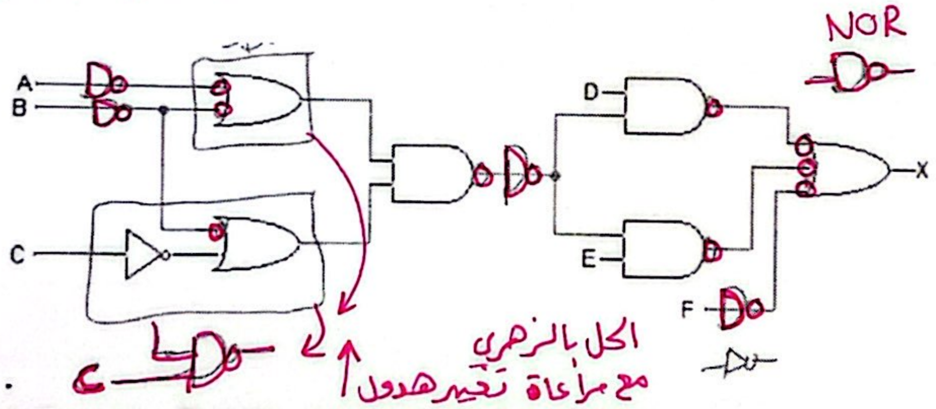
- Consider the following product-of-sums expression:

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

- A 2-level OR-AND circuit can be converted easily to a 2-level **NOR-NOR implementation**



Implement the given circuit using only NOR gates

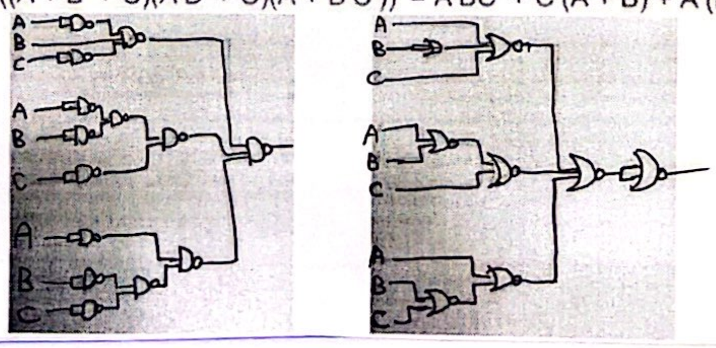


- Example: Find the complement of the following expression and implement it using (1) NAND gates, and (2) NOR gates:

$$G(A, B, C) = (A + B' + C)(A'B' + C)(A + B'C')$$

- Solution:

$$G' = ((A + B' + C)(A'B' + C)(A + B'C'))' = A'BC' + C'(A + B) + A'(B + C)$$



Exclusive OR / Eclusive NOR

x	y	xOR
0	0	0
0	1	1
1	0	1
1	1	0

القيم مختلفات

	x	y	xNOR
نفس بعض ←	0	0	1
	0	1	0
	1	0	0
نفس بعض ←	1	1	1

xOR function is :- $x'y + xy'$

xNOR function is :- $xy + x'y'$

xOR and xNOR don't exist for more than two inputs because they are complex

for exampe : for 3 inputs we use two gates of them not one

XOR and XNOR Properties :-

* $x \oplus 0 = x$ $\quad\quad$ $x \oplus 1 = x'$

* $x \oplus x = 0$ $\quad\quad$ $x \oplus x' = 1$

* $x \oplus y = y \oplus x$

* $x' \oplus y' = x \oplus y$

* $(x \oplus y)' = x' \oplus y = x \oplus y'$

also XOR and XNOR are associative (تجميعي)

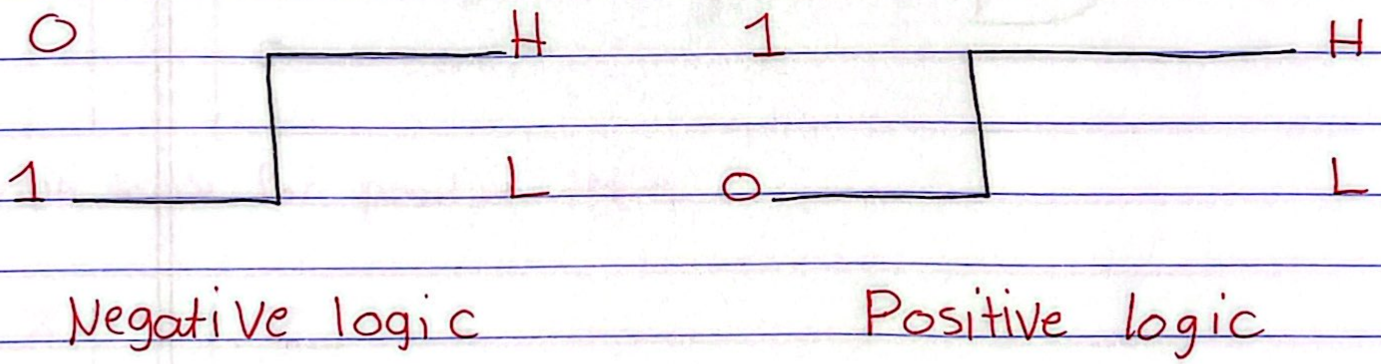
* $(x \oplus y) \oplus z = x \oplus (y \oplus z) = x \oplus y \oplus z$

* $((x \oplus y)' \oplus z)' = (x \oplus (y \oplus z)')' = x \oplus y \oplus z$

Positive and Negative Logic

* In Positive logic we choose high-level to represent logic 1 and low-level to represent zero

* In Negative logic we choose the high-level represent zero and low-level to represent 1

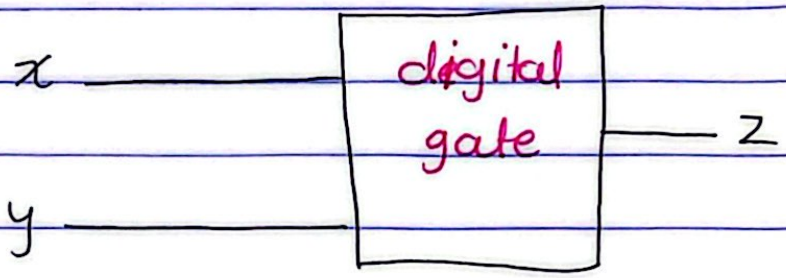


and it's up to the user to decide on a positive or negative logic polarity

Positive and negative logic

x	y	z
L	L	L
L	H	L
H	L	L
H	H	H

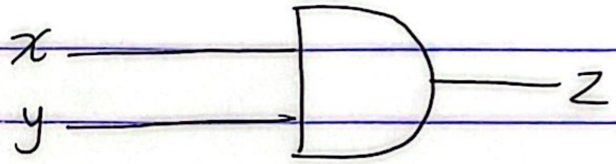
⇒



truth table with H and L

x	y	z
0	0	0
0	1	0
1	0	0
1	1	1

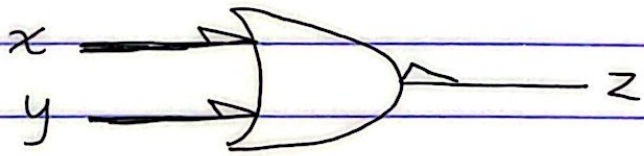
⇒



truth table for positive logic

x	y	z
1	1	1
1	0	1
0	1	1
0	0	0

⇒



truth table for negative logic