CSE 260
Logic Gates &
Boolean Algebra

Binary Logic

- Binary logic consists of binary variables and logical operations.
- Variables are designated by letters such as A, B, C, x, y, z etc. with only 2 possible values: 1 and 0.
- Logic operations: and, or, not etc.

Logic Gates

- The most basic digital devices are called gates.
- A gate has one or more inputs and produces an output that is a function of the current input values.
- The relationship between the input and the output is based on a **certain logic**.

Truth Table

- Provides a listing of every possible combination of inputs and its corresponding outputs.

INPUTS	OUTPUTS
...	...
...	...

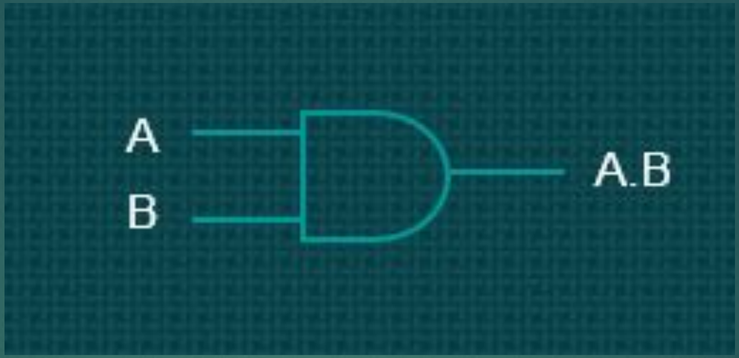
All Logic Gates

- NOT
- AND
- OR
- XOR
- XNOR
- NAND
- NOR

Most Important logic gates

- AND
- OR
- NOT

2-input AND gate

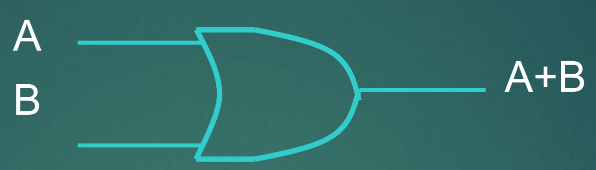


A	B	A.B
0	0	0
0	1	0
1	0	0
1	1	1

1 = on
0 = off

Output will be 1 only when both inputs are 1

2-input OR gate



A	B	A + B
0	0	0
0	1	1
1	0	1
1	1	1

1 = on
0 = off

Output will be 1 when at least one input is 1

NOT gate (Inverter)

A	A'
0	1
1	0

1 = on
0 = off

Output will be the inverse of the input

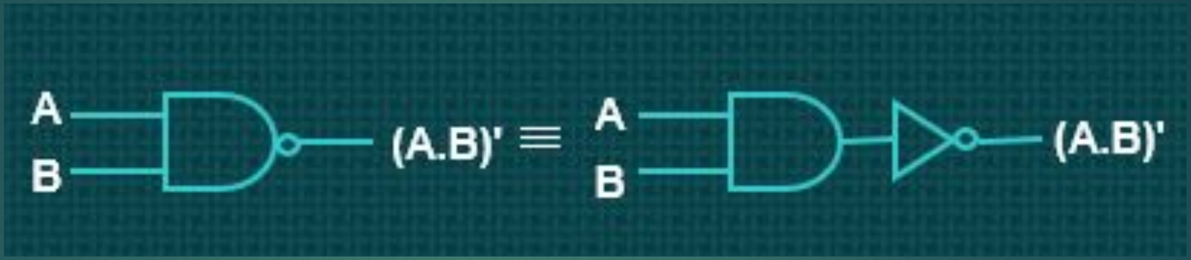
Some Other Gates

- NAND
- NOR
- XOR
- XNOR

- NAND and NOR are also known as universal gates

- A **universal gate** is a **gate** which can implement any other **gate**

2-input NAND gate

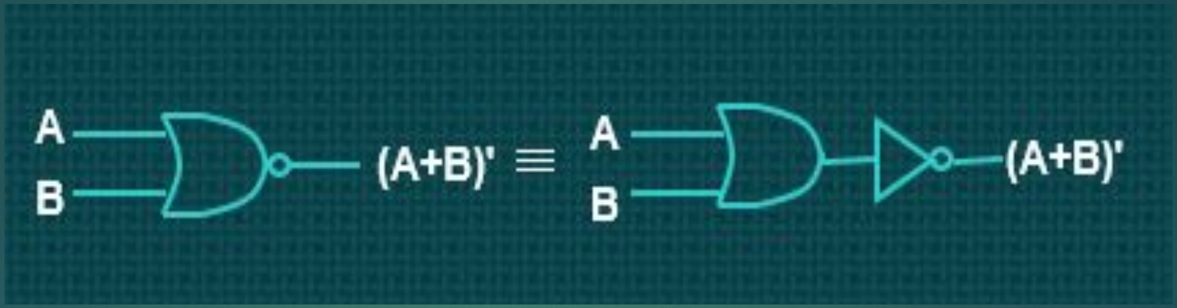


A	B	(A.B)'
0	0	1
0	1	1
1	0	1
1	1	0

1 = on
0 = off

Inverse of AND

2-input NOR gate



A	B	(A+B)'
0	0	1
0	1	0
1	0	0
1	1	0

1 = on
0 = off

Inverse of OR

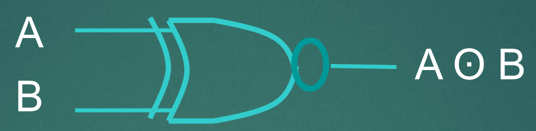
2-input XOR gate

A	B	$A \oplus B$
0	0	0
0	1	1
1	0	1
1	1	0

1 = on
0 = off

Output will be 1 for odd number of 1s in input

2-input XNOR gate



A	B	A⊙B
0	0	1
0	1	0
1	0	0
1	1	1

1 = on
0 = off

- Inverse of XOR

Proof using Truth Table

- Prove that: $x \cdot (y + z) = (x \cdot y) + (x \cdot z)$

(i) Construct truth table for LHS & RHS of above equality.

Note: if there are 3 variable, truth table should have 2^n combination of input

x	y	z	y + z	x.(y + z)	x.y	x.z	(x.y)+(x.z)
0	0	0	0	0	0	0	0
0	0	1	1	0	0	0	0
0	1	0	1	0	0	0	0
0	1	1	1	0	0	0	0
1	0	0	0	0	0	0	0
1	0	1	1	1	0	1	1
1	1	0	1	1	1	0	1
1	1	1	1	1	1	1	1

(ii) Check that LHS = RHS

Postulate is SATISFIED because output column 5 & 8 (for LHS & RHS expressions) are equal for all cases.

BOOLEAN ALGEBRA

Boolean Algebra

- Like any other deductive mathematical system, defined with a set of elements, a set of operators and a number of axioms or postulates.

- In Boolean algebra, set consists at least 2 variables say x & y, with 2 binary operations {+} and {.} and 1 unary operation {'}

Basic Theorems of Boolean Algebra

- Theorems can be proved using the truth table method. (Exercise: Prove De-Morgan's theorem using the truth table.)
- They can also be proved by algebraic manipulation using axioms/postulates or other basic theorems.

Basic Theorems of Boolean Algebra

- Postulate 5 (a) $x+0=x$ (b) $x.1=x$ identity
- Postulate 3 (a) $x+x'=1$ (b) $x.x'=0$ complement
- Th 1 (a) $x+x=x$ (b) $x.x=x$
- Th 2 (a) $x+1=1$ (b) $x.0=0$
- Th 3, involution $(x')'=x$
- Pos 2 (a) $x+y=y+x$ (b) $xy=yx$ commutative
- Th 4 (a) $x(yz)=(xy)z$ (b) $x+(y+z)=(x+y)+z$
- Pos 6 (a) $x(y+z)=xy+xz$ (b) $x+yz=(x+y)(x+z)$ Distributi- -ve
- Th 5, DeMorgan (a) $(x+y)'=x'y'$ (b) $(xy)'=x'+y'$
- Th 6, Absorption (a) $x+xy=x$ (b) $x(x+y)=x$

All are very very important!

Basic Theorems of Boolean Algebra

- Theorem 2a can be proved by:

$$\begin{aligned} x + 1 &= x+(x+x') \quad \text{(complement)} \\ &= (x+x)+x' \quad \text{(Th. 4)} \\ &= x+x' \quad \text{(complement)} \\ &= 1 \end{aligned}$$

- By duality, theorem 2b:

$$x.(0)= 0$$

- *Note: There can be other ways of making this proof. See Morris Mano*

Basic Theorems of Boolean Algebra

- Theorem 6a (absorption) can be proved by:

$$
\begin{aligned}
x + x.y &= x.1 + x.y && \text{(identity)} \\
&= x.(1 + y) && \text{(distributivity)} \\
&= x.(y + 1) && \text{(commutativity)} \\
&= x.1 && \text{(Theorem 2a)} \\
&= x && \text{(identity)}
\end{aligned}
$$

- By duality, theorem 6b:

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

- Try prove this by algebraic manipulation.

All Together

TABLE 2-1
Postulates and Theorems of Boolean Algebra

Postulate 2	(a) $x + 0 = x$	(b) $x \cdot 1 = x$
Postulate 5	(a) $x + x' = 1$	(b) $x \cdot x' = 0$
Theorem 1	(a) $x + x = x$	(b) $x \cdot x = x$
Theorem 2	(a) $x + 1 = 1$	(b) $x \cdot 0 = 0$
Theorem 3, involution	$(x')' = x$	
Postulate 3, commutative	(a) $x + y = y + x$	(b) $xy = yx$
Theorem 4, associative	(a) $x + (y + z) = (x + y) + z$	(b) $x(yz) = (xy)z$
Postulate 4, distributive	(a) $x(y + z) = xy + xz$	(b) $x + yz = (x + y)(x + z)$
Theorem 5, DeMorgan	(a) $(x + y)' = x'y'$	(b) $(xy)' = x' + y'$
Theorem 6, absorption	(a) $x + xy = x$	(b) $x(x + y) = x$

Duality

- **Duality Principle** – every valid Boolean expression (equality) remains valid if the operators and identity elements are interchanged, as follows:

  $+ \leftrightarrow .$

  $1 \leftrightarrow 0$

- Example: Given the expression

  $$a + (b.c) = (a+b).(a+c)$$

  then its dual expression is

  $$a\ .\ (b+c) = (a.b) + (a.c)$$

Duality

- Duality gives free theorems – “two for the price of one”. You prove one theorem and the other comes for free!
- If $(x+y+z)' = x'.y.'z'$ is valid, then its dual is also valid:

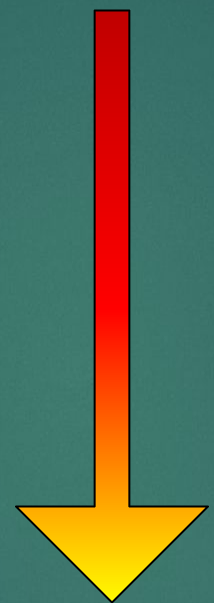
  $(x.y.z)' = x'+y'+z'$
- If $x + 1 = 1$ is valid, then its dual is also valid:

  $x . 0 = 0$

Operator Precedence

- Parenthesis
- NOT
- AND
- OR

Highest



Lowest

Boolean Functions (Solve ?)

- Examples:

  F1= xyz'

  F2= x + y'z

  F3=(x'y'z)+(x'yz)+(xy')

  F4=xy'+x'z

x	y	z	F1	F2	F3	F4
0	0	0	0	0	0	0
0	0	1	0	1	1	1
0	1	0	0	0	0	0
0	1	1	0	0	1	1
1	0	0	0	1	1	1
1	0	1	0	1	1	1
1	1	0	1	1	0	0
1	1	1	0	1	0	0

From the truth table, F3=F4.
Can you also prove by algebraic manipulation that F3=F4?

- F3=(x'y'z)+(x'yz)+(xy')

  = x'y'z+x'yz+xy'

  =x'z(y'+y)+xy'

  =x'z(1)+xy'

  =x'z+xy'

  =F4

TABLE 2-1
Postulates and Theorems of Boolean Algebra

Postulate 2	(a) $x + 0 = x$	(b) $x \cdot 1 = x$
Postulate 5	(a) $x + x' = 1$	(b) $x \cdot x' = 0$
Theorem 1	(a) $x + x = x$	(b) $x \cdot x = x$
Theorem 2	(a) $x + 1 = 1$	(b) $x \cdot 0 = 0$
Theorem 3, involution	$(x')' = x$	
Postulate 3, commutative	(a) $x + y = y + x$	(b) $xy = yx$
Theorem 4, associative	(a) $x + (y + z) = (x + y) + z$	(b) $x(yz) = (xy)z$
Postulate 4, distributive	(a) $x(y + z) = xy + xz$	(b) $x + yz = (x + y)(x + z)$
Theorem 5, DeMorgan	(a) $(x + y)' = x'y'$	(b) $(xy)' = x' + y'$
Theorem 6, absorption	(a) $x + xy = x$	(b) $x(x + y) = x$

Try it yourself

a)Simplify to minimum literals: xy+xy’

b)Reduce to 4 literals(variables) : BC+AC’+AB+BCD

Solution

- A) xy+xy’=x(y+y’)=x(1)=x
- B)BC+AC’+AB+BCD

  =BC(1+D)+AC’+AB

  =BC(1)+AC’+AB

  =BC+AB+AC’

  =B(C+A)+AC’

Try it yourself: simplify the following equations

1. $x+x'y$
2. $x(x'+y)$
3. $x'y'z+x'yz+xy'$

TABLE 2-1
Postulates and Theorems of Boolean Algebra

Postulate 2	(a) $x + 0 = x$	(b) $x \cdot 1 = x$
Postulate 5	(a) $x + x' = 1$	(b) $x \cdot x' = 0$
Theorem 1	(a) $x + x = x$	(b) $x \cdot x = x$
Theorem 2	(a) $x + 1 = 1$	(b) $x \cdot 0 = 0$
Theorem 3, involution	$(x')' = x$	
Postulate 3, commutative	(a) $x + y = y + x$	(b) $xy = yx$
Theorem 4, associative	(a) $x + (y + z) = (x + y) + z$	(b) $x(yz) = (xy)z$
Postulate 4, distributive	(a) $x(y + z) = xy + xz$	(b) $x + yz = (x + y)(x + z)$
Theorem 5, DeMorgan	(a) $(x + y)' = x'y'$	(b) $(xy)' = x' + y'$
Theorem 6, absorption	(a) $x + xy = x$	(b) $x(x + y) = x$

Solution

1. $x+x'y=(x+x').(x+y)=1.(x+y)=x+y$
2. $x(x'+y)=xx'+xy=0+xy=xy$
3. $x'y'z+x'yz+xy' =$
$x'z(y'+y)+xy'=x'z+xy'$

Now Try Proving Using Truth Table!!!

Complementing a function

1. Take dual of the function
2. Complement each literals

<u>Example:</u> $F1= x'yz'+x'y'z$

1. Dual of the function F1 is $(x'+y+z')(x'+y'+z)$
2. Complement each literal= $(x+y'+z)(x+y+z')$

Therefore, $F1'= (x+y'+z)(x+y+z')$

Same as applying De-Morgan's law on the function

Try it yourself

- What is the complement of $F2=x(y'z'+yz)$

Solution

- Duality: $x+(y'+z')(y+z)$
- Complement= $x'+(y'+z')(y+z)$

More Practice:

Simplify the following Boolean expression to a minimum number literals:

- a) $xy + xy'$
- b) $(x + y)(x + y')$
- c) $xyz + x'y + xyz'$
- d) $(A+B)'(A'+B')'$
- e) $(AB)'(A' + B)(B' + B)$
- f) $(A + C)(AD + AD') + AC + C$

Solution

a) $xy + xy' = x(y+y') = x.1 = x$

b) $(x+y)(x+y') = xx + xy' + yx+yy' = x + xy' + xy + 0 = x(1+ y' + y) = x.1= x$

Also $(x+y)(x+y') = x+ yy' = x + 0 = x$

c) $xyz + x'y + xyz' = xy(z+z') + x'y = xy + x'y = y(x+x')=y$

d) $(A+B)'(A'+B')'= (A'B').(AB) = 0$

e) A' [Find the process yourself]

f) $A + C$ [Find the process yourself]

Practice! Practice! Practice!

Find the complement of the following expressions:

- a) $xy' + x'y$
- b) $(AB' + C)D' + E$
- c) $(x + y' + z)(x' + z')(x + y)$

Solution

a) $[xy'+x'y]' = (xy')' . (x'y)' = (x'+y).(x+y') = xx' + yy' + xy+x'y'=xy+x'y'$

b) $[(AB'+C)D'+E]' = [(AB'+C)D']'.E' = [(AB'+C)'+D] . E'= [(A'+B).C' +D].E' =$

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

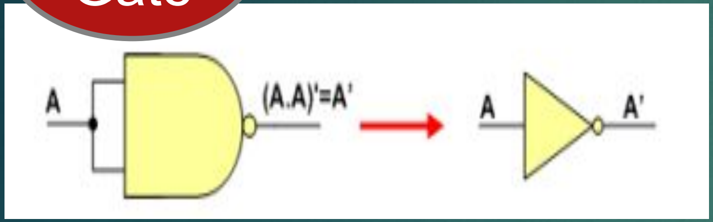
c) $[(x+y'+z)(x'+z')(x+y)]' = (x+y'+z)'+(x'+z')'+(x+y)'= x'yz' + xz + x'y'$

If the question doesn't ask you to simplify, then you don't need to simplify after complement.

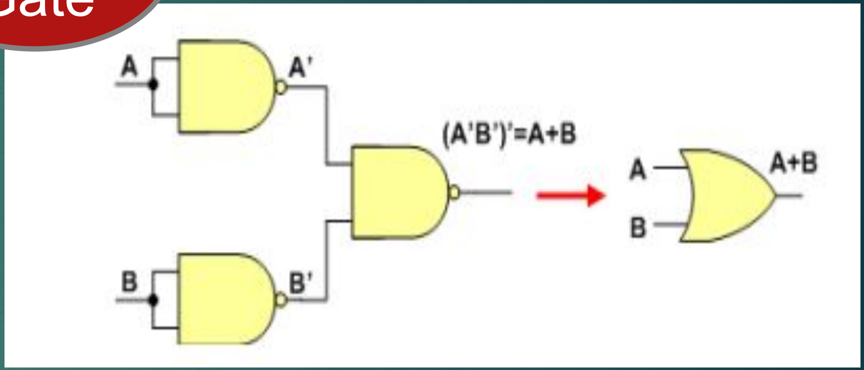
Using NAND and NOR to Build Other Gates and Functions

Using NAND

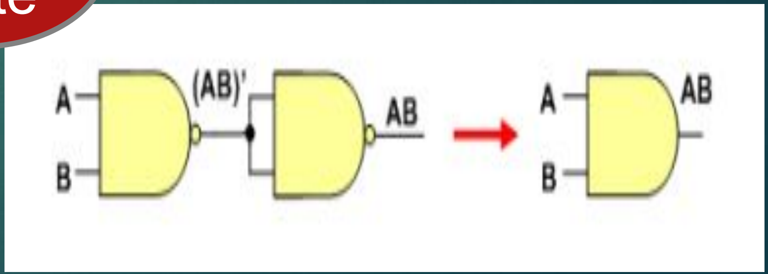
NOT Gate



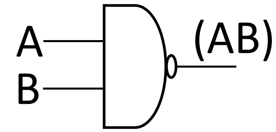
OR Gate

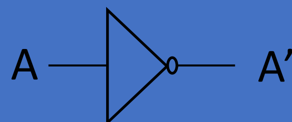
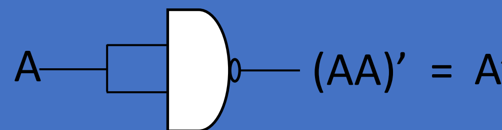
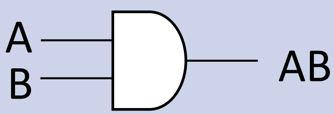
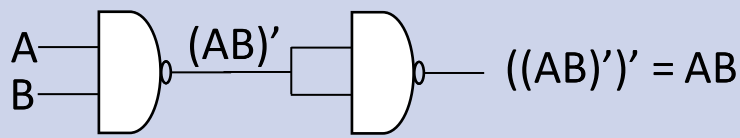
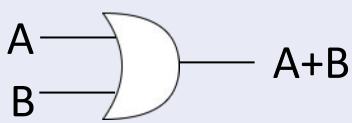
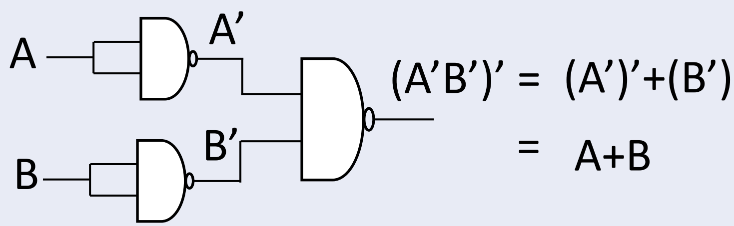


AND Gate



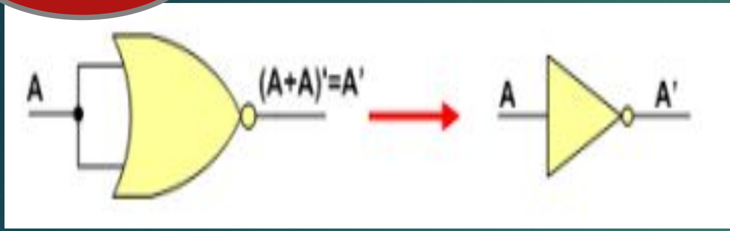
Basic gates using NAND gate



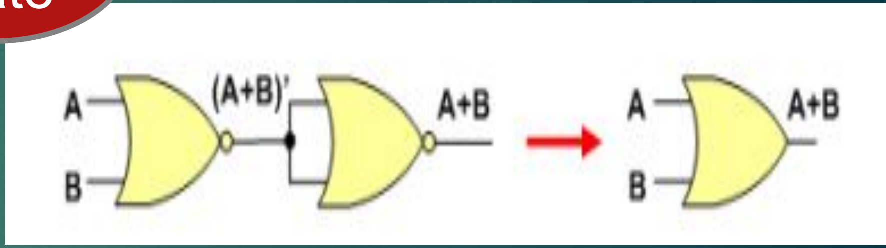
Not		
And		
Or		

Using NOR

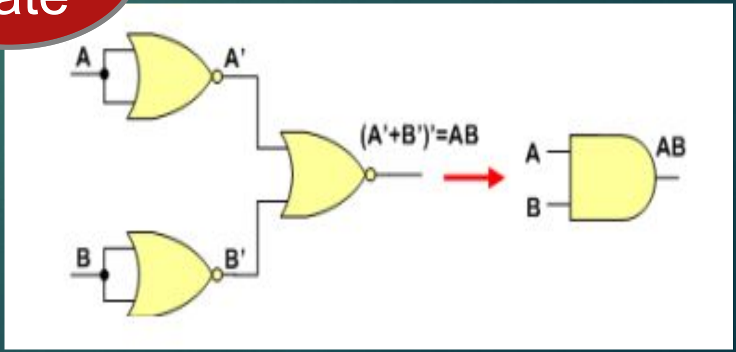
NOT Gate



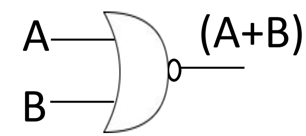
OR Gate

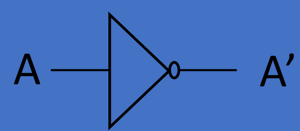
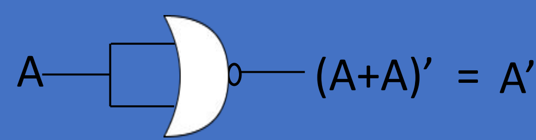
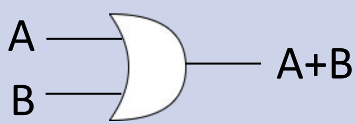
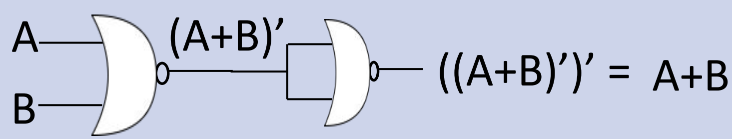
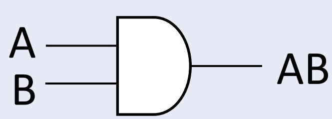
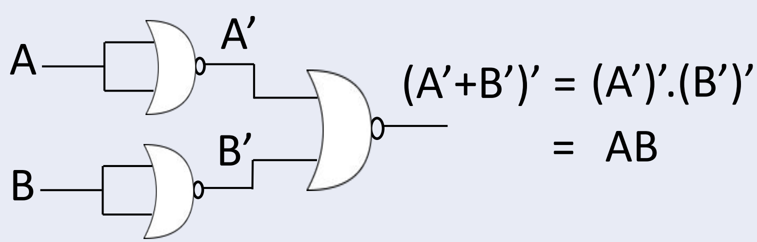


AND Gate

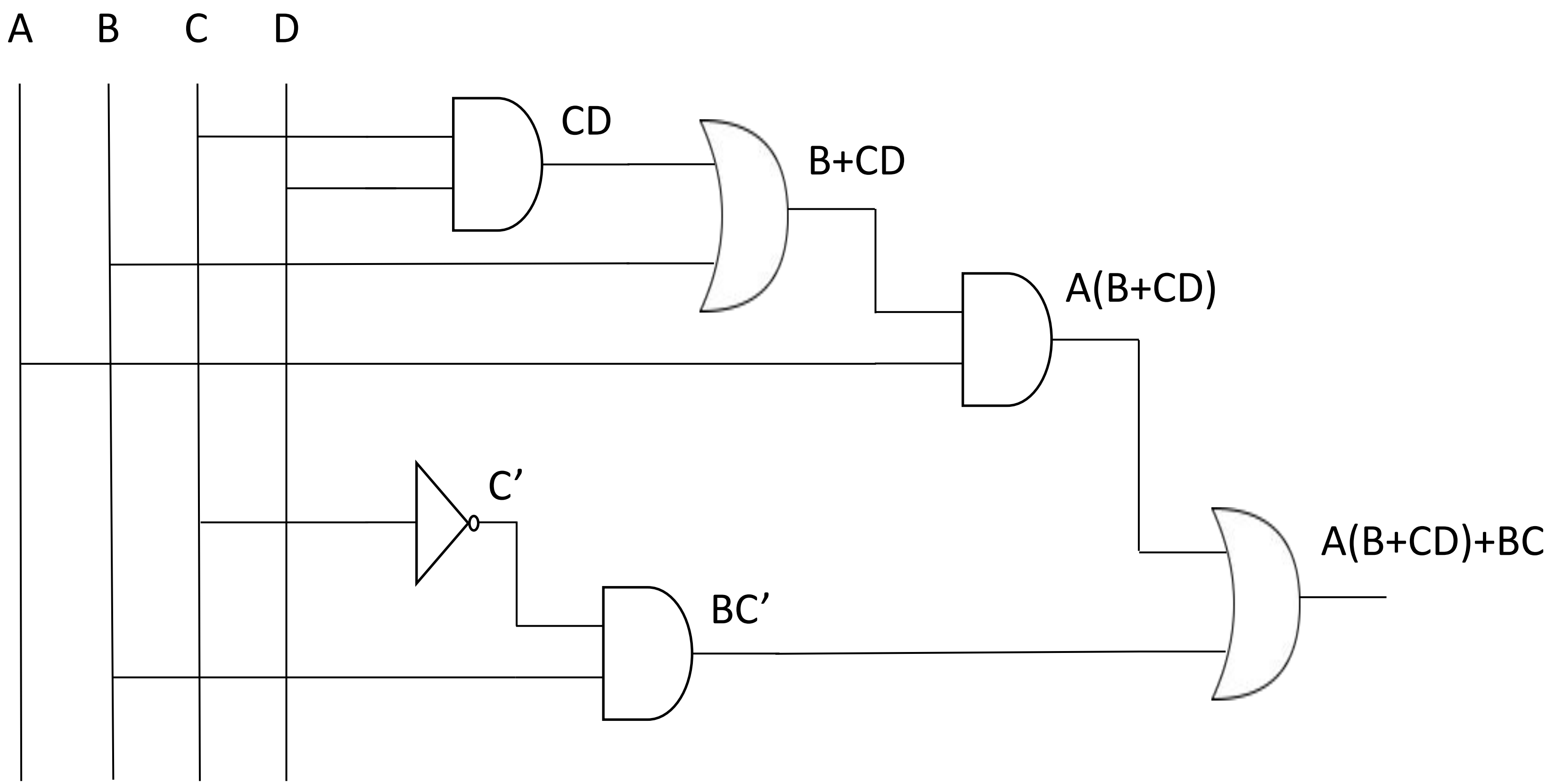


Basic gates using NOR gate



<table>
<tr><td>Not</td><td>
</td><td>
</td></tr>
<tr><td>Or</td><td>
</td><td>
</td></tr>
<tr><td>And</td><td>
</td><td>
</td></tr>
</table>

Use AND, OR, NOT to represent F=A(B+CD)+BC’

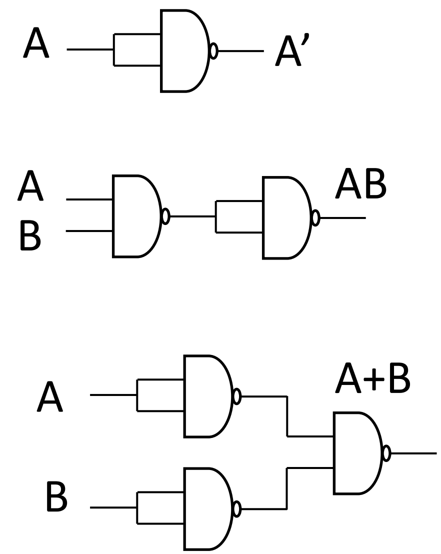


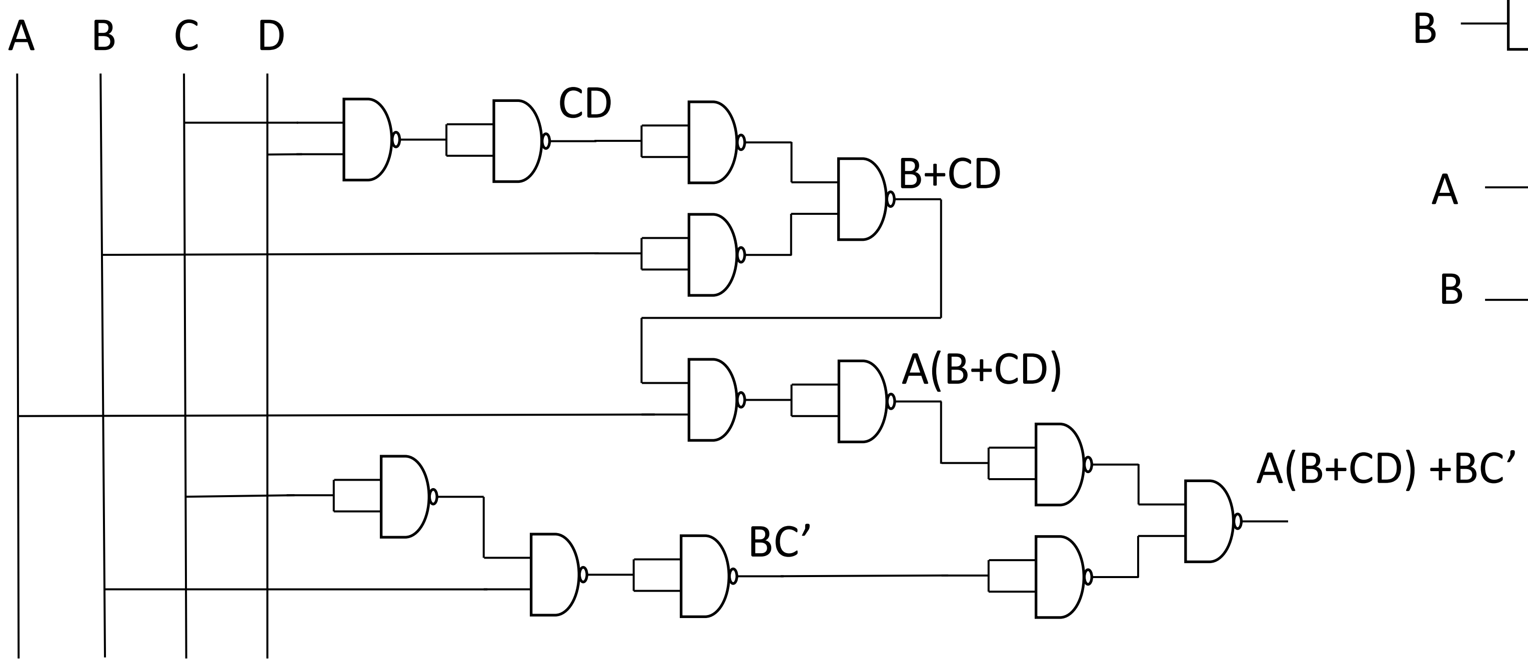
Use NAND to represent F=A(B+CD)+BC’

1. If an expression is given, represent it using AND,OR and NOT gates
2. Draw each gate with equivalent NAND representation
3. Remove any 2 cascading inverters
4. Remove inverters from single input connection and replace input with its complement.

Tutorial:
https://www.youtube.com/watch?v=-EjGrPhol70&list=PLTIXQu_162Qg8-oRqv_iGYHSz2XrfUc51&index=7

Use NAND to represent F=A(B+CD)+BC’





Use NOR to represent F=A(B+CD)+BC’

1. If an expression is given, represent it using AND,OR and NOT gates
2. Draw each gate with equivalent NOR representation
3. Remove any 2 cascading inverters
4. Remove inverters from single input connection and replace input with its complement.

Tutorial: https://www.youtube.com/watch?v=eAuOqgT5lqM&list=PLTIXQu_162Qg8-oRqv_iGYHSz2XrfUc51&index=8

Use NOR to represent F=A(B+CD)+BC’