Computer Systems

Logic Expression Simplification

Part 02

Single Gate Type Circuits

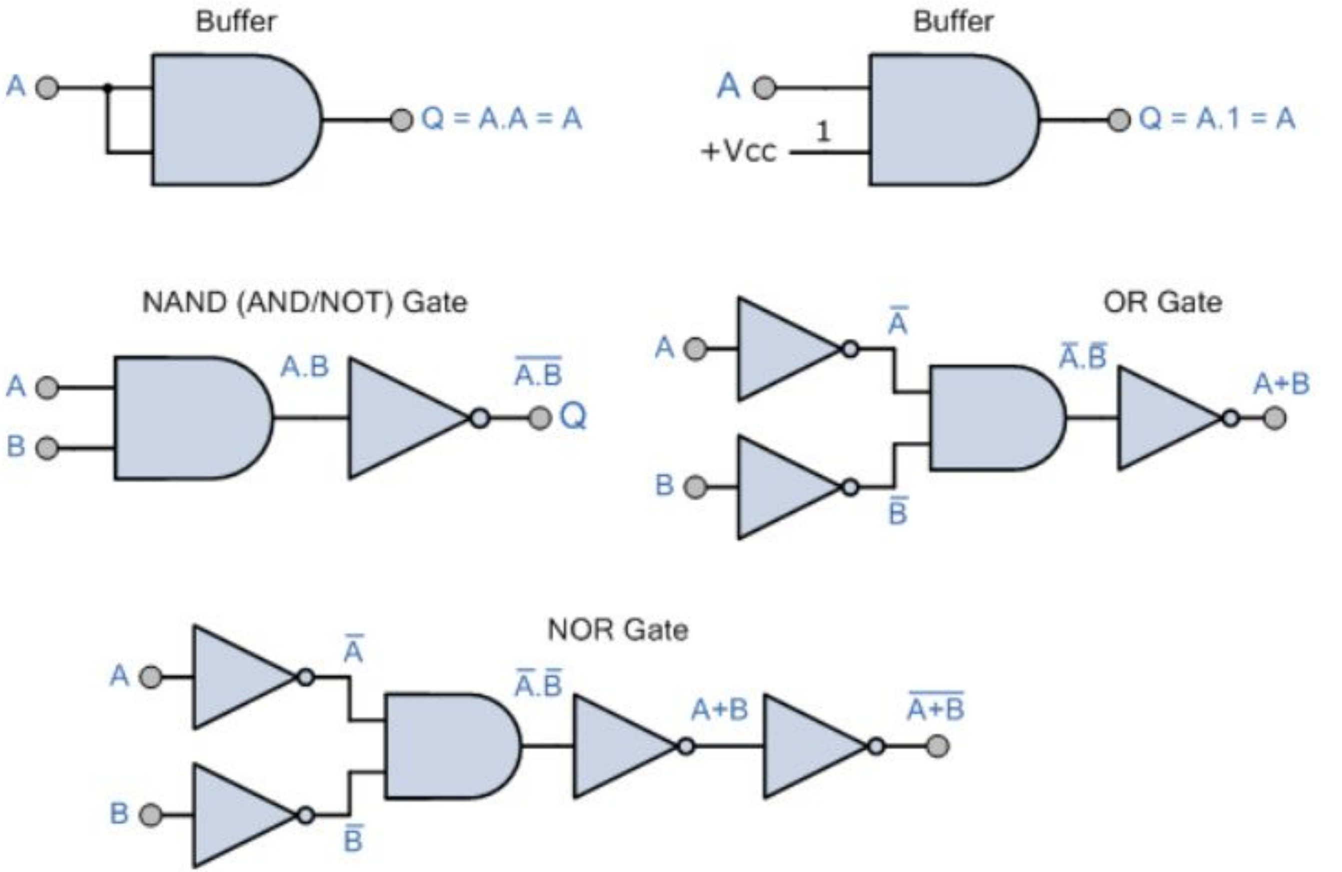
- Circuits are prefered to be built using a single gate type because it's cheaper.
- Manufacturing ICs with different types of gates are expensive.
- Therefore when we preparing the circuits we consider about Functional Completeness.

Functional Completeness

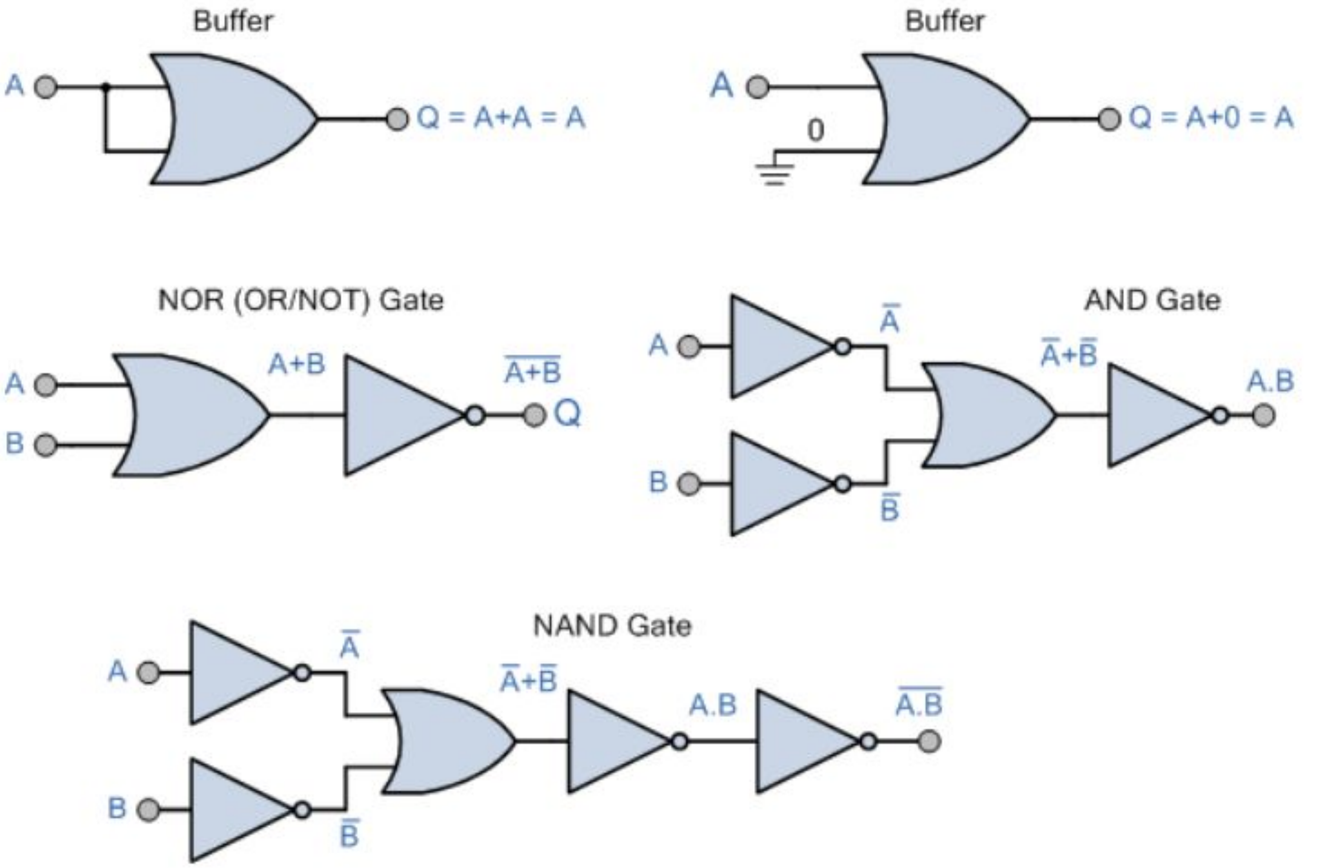
- AND, OR and NOT operations are the primary functionalities.
- Therefore, any set of gates that can demonstrate all three functionalities is called Functionally Complete Set.
- Functionally Complete Sets
  - AND , OR and NOT
  - AND and NOT
  - OR and NOT
  - NAND
  - NOR

> A set of Boolean operators, which can be used to express all possible truth tables by combining members of the set into a Boolean expression.

Functionally complete sets (AND and NOT gates)

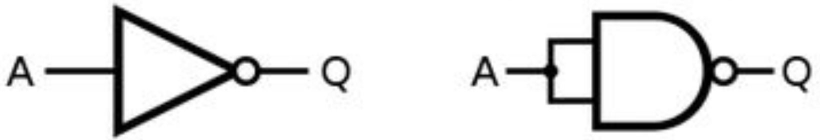


Functionally complete sets (OR and NOT gates)



Functionally complete sets (NAND gate)

Desired NOT Gate **NAND Construction**



Q = NOT(**A**) = **A** NAND **A**

Truth Table

Input A		Output Q
0		1
1		0

=

NOT from NAND

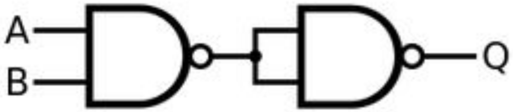
A	$Q = \overline{A.A} = \overline{A}$
0	1
1	0

Functionally complete sets (NAND gate)

Desired AND Gate

Q = **A** AND **B**

NAND Construction



= (**A** NAND **B**) NAND (**A** NAND **B**)

AND from NAND

Truth Table

Input A	Input B	Output Q
0	0	0
0	1	0
1	0	0
1	1	1

=

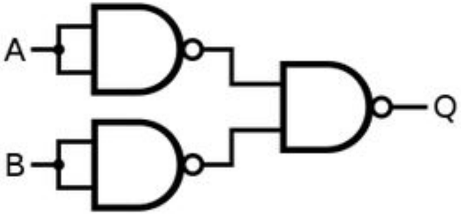
A	B	$\overline{A.B}$	Q = $\overline{\overline{A.B}}$ = $A.B$
0	0	1	0
0	1	1	0
1	0	1	0
1	1	0	1

Functionally complete sets (NAND gate)

Desired OR Gate

Q = **A** OR **B**

NAND Construction



= (**A** NAND **A**) NAND (**B** NAND **B**)

OR from NAND

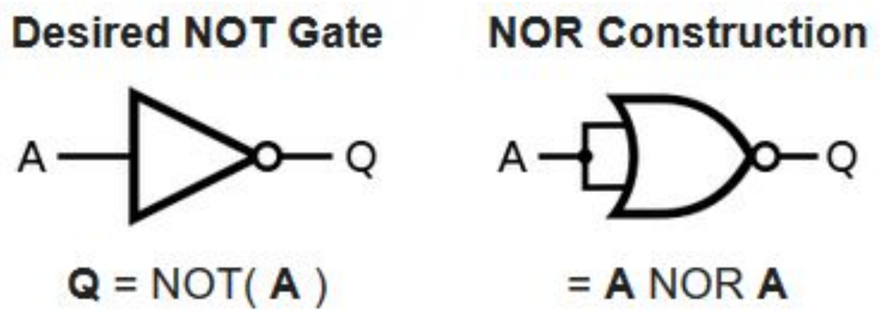
Truth Table

Input A	Input B	Output Q
0	0	0
0	1	1
1	0	1
1	1	1

=

A	B	$\overline{A}$	$\overline{B}$	$\overline{A}.\overline{B}$	Q
0	0	1	1	1	0
0	1	1	0	0	1
1	0	0	1	0	1
1	1	0	0	0	1

Functionally complete sets (NOR gate)



Truth Table

Input A	Output Q
0	1
1	0

=

NOT from NOR

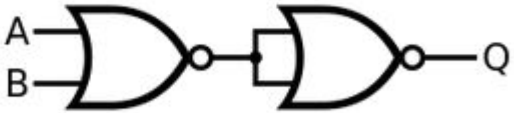
A	$Q = \overline{A + A} = \overline{A}$
0	1
1	0

Functionally complete sets (NOR gate)

Desired OR Gate

Q = **A** OR **B**

NOR Construction



= (**A** NOR **B**) NOR (**A** NOR **B**)

Truth Table

Input A	Input B		Output Q
0	0		0
0	1		1
1	0		1
1	1		1

=

OR from NOR

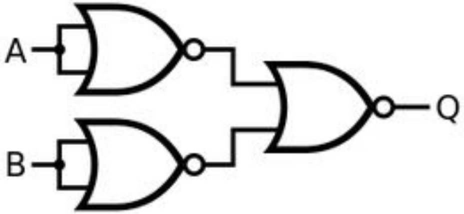
A	B	$\overline{A+B}$	Q = $\overline{\overline{A+B}}$ = A+B
0	0	1	0
0	1	0	1
1	0	0	1
1	1	0	1

Functionally complete sets (NOR gate)

Desired AND Gate

Q = A AND B

NOR Construction



= (**A** NOR **A**) NOR (**B** NOR **B**)

Truth Table

Input A	Input B	Output Q
0	0	0
0	1	0
1	0	0
1	1	1

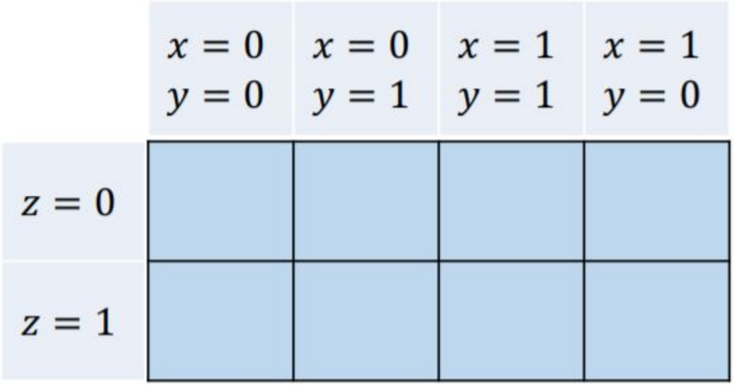
=

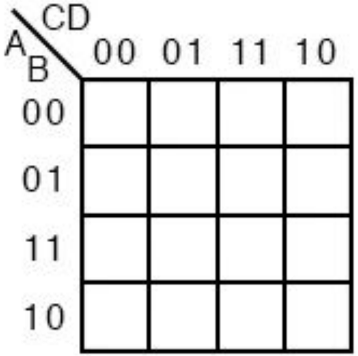
AND from NOR

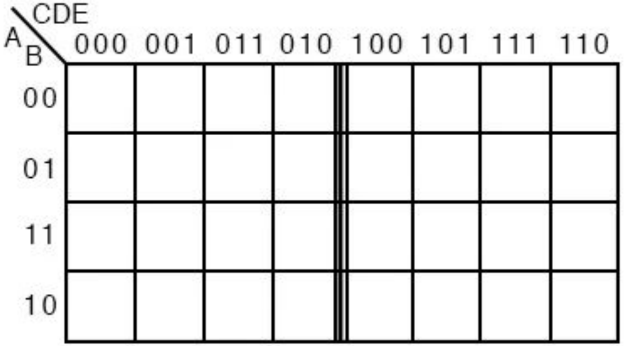
A	B	$\overline{A}$	$\overline{B}$	$\overline{A}+\overline{B}$	Q
0	0	1	1	1	0
0	1	1	0	1	0
1	0	0	1	1	0
1	1	0	0	0	1

Karnaugh Map (K-Map)

- A gate level minimization technique.
- For a boolean function with n variables, corresponding K Map have 2^n cells.
- Only one variable changed when moving to an adjacent column or row.







5-variable Karnaugh map (overlay)

Karnaugh Map (K-Map)

- Steps
    1. Mapping
    2. Grouping
    3. Deriving

K Map - (1) Mapping

- Draw the grid
  - Cells = 2^n
- Put 1 to the described terms ; others 0.

K Map - (1) Mapping

Sum of Minterms

- **Example :**

$$F = \bar{x}.\bar{y}.z + \bar{x}.y.z + x.\bar{y}.\bar{z} + x.y.\bar{z}$$

		$\bar{x}\bar{y}$	$\bar{x}y$	xy	$x\bar{y}$
	z \ xy	00	01	11	10
$\bar{z}$	0				
z	1				

K Map - (1) Mapping

- **Example :**

$$F = \bar{x}.\bar{y}.z + \bar{x}.y.z + x.\bar{y}.\bar{z} + x.y.\bar{z}$$

	z \ xy	$\bar{x}\bar{y}$ 00	$\bar{x}y$ 01	xy 11	$x\bar{y}$ 10
$\bar{z}$	0	0	0	1	1
z	1	1	1	0	0

K Map - (2) Grouping

- Group 1s (For Sum of Products) or 0s (For Product of Sums)
  - Grouping can be done vertically or horizontally (Not diagonal).
  - Number of cells in a group should be a power of 2 (1,2,4,8 etc.)
  - Select the largest group possible.
  - Groups can be overlapped to form the largest group.
  - Groups can be formed by wrapping around the grid.
  - There should be as few groups as possible while covering all the 1s (or 0s).

K Map - (2) Grouping

Example :

$$F = \bar{x}.\bar{y}.z + \bar{x}.y.z + x.\bar{y}.\bar{z} + x.y.\bar{z}$$

		$\bar{x}\bar{y}$	$\bar{x}y$	xy	$x\bar{y}$
	z \ xy	00	01	11	10
$\bar{z}$	0	0	0	1	1
z	1	1	1	0	0

K Map - (3) Deriving

- Write the unchanged terms in each group

K Map - (3) Deriving

- Group 1 (Red)
- Unchanged: x and $\bar{z}$

Example :

$$F = \bar{x}.\bar{y}.z + \bar{x}.y.z + x.\bar{y}.\bar{z} + x.y.\bar{z}$$

		$\bar{x}\,\bar{y}$	$\bar{x}\,y$	$x\,y$	$x\,\bar{y}$	
	z \ xy	00	01	11	10	
$\bar{z}$	0	0	0	1	1	$x.\bar{z}$
z	1	1	1	0	0	

K Map - (3) Deriving

- Group 2 (Blue)
- Unchanged: $\bar{x}$ and z

Example :

$$F = \bar{x}.\bar{y}.z + \bar{x}.y.z + x.\bar{y}.\bar{z} + x.y.\bar{z}$$

		$\bar{x}\bar{y}$	$\bar{x}y$	xy	$x\bar{y}$
	z \ xy	00	01	11	10
$\bar{z}$	0	0	0	1	1
z	1	1	1	0	0

$x.\bar{z}$

$\bar{x}.z$

K- Map Example

- $F = \overline{A}\ \overline{B}\ \overline{C} + \overline{A}\ \overline{B}\ C$

K- Map Example

- $F = \bar{A}\,\bar{B}\,\bar{C} + \bar{A}\,\bar{B}\,C$

		$\bar{A}\bar{B}$	$\bar{A}B$	AB	$A\bar{B}$
	AB \ C	00	01	11	10
$\bar{C}$	0	1	0	0	0
C	1	1	0	0	0

$\bar{A}\bar{B}$

$$A\,B\,\bar{C} + \bar{A}\,\bar{B}\,C$$
$$= \bar{A}\,\bar{B}$$

K- Map Examples

- $F = \overline{A}\ \overline{B}\ \overline{C} + \overline{A}\ \overline{B}\ C + \overline{A}\ B\ C + \overline{A}\ B\ \overline{C}$

K- Map Examples

- $F = \bar{A}\,\bar{B}\,\bar{C} + \bar{A}\,\bar{B}\,C + \bar{A}\,B\,C + \bar{A}\,B\,\bar{C}$

		$\bar{A}\bar{B}$	$\bar{A}B$	AB	$A\bar{B}$
	C \ AB	00	01	11	10
$\bar{C}$	0	1	1	0	0
C	1	1	1	0	0

$\bar{A}$

$\bar{A}\,\bar{B}\,\bar{C} + \bar{A}\,\bar{B}\,C + \bar{A}\,B\,C + \bar{A}\,B\,\bar{C}$

$= \bar{A}$

Exercise

1. **Simplify using K Maps and validate the result using boolean algebra**
   a. A’B + AB’ +AB
   b. A’BC’ + A’BC + A’B’C + ABC
   c. BC + BC’ + BA
   d. AB + A(B+C) + B(B+C)
   e. AB’ + A(B+C)’ + B(B+C)’