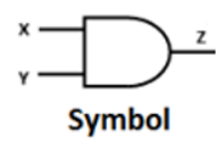
UNIT 04 SHORTNOTE

✦ Basic Logic Gates

○ AND Gate

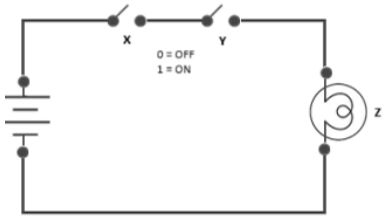


Symbol

Z = X.Y
Boolean Expression

X	Y	Z = X*Y
0	0	0
0	1	0
1	0	0
1	1	1

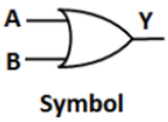
Truth Table



Electrical Circuit

○ OR Gate

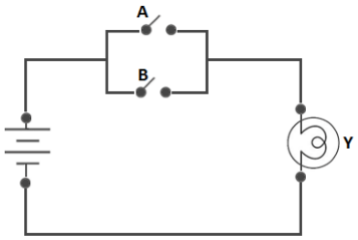
Y = A + B
Boolean Expression



Symbol

A	B	Y = A + B
0	0	0
0	1	1
1	0	1
1	1	1

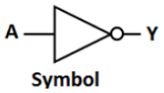
Truth Table



Electrical Circuit

○ NOT Gate

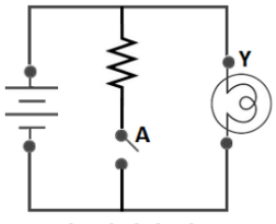
$A = \overline{A}$
Boolean Expression



Symbol

A	Y = Ā
0	1
1	0

Truth Table



Electrical Circuit

✦ Combinational Gates

○ XOR Gate

(a) Circuit symbol

S1	S2	L
0	0	0
0	1	1
1	0	1
1	1	0

(b) Truth table

$L = S1 \oplus S2$

(c) Boolean expression

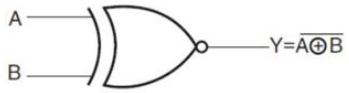
* Results a true if and only one of the inputs to the gate is true.

Truth table for three input XOR gate

INPUTS				Final Output
S1	S2	S3	S1 ⊕ S2	S1 ⊕ S2 ⊕ S3
0	0	0	0	0
0	0	1	0	1
0	1	0	1	1
0	1	1	1	0
1	0	0	1	1
1	0	1	1	0
1	1	0	0	0
1	1	1	0	1

* Returns True If odd number of '1' inputs to the gate.

○ XNOR Gate



$Y = (\overline{A \oplus B}) = (A.B + \overline{A}.\overline{B})$

A	B	Y
0	0	1
0	1	0
1	0	0
1	1	1

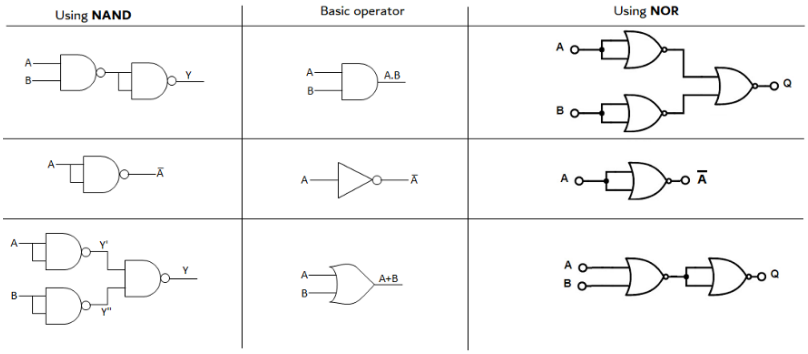
* The logical complement of the XOR gate.

✦ Universal Gates

* Can be used to construct all other logic gates.

** The advantage of universal gates:*

NAND and NOR gates are economical and easier to fabricate and are the basic gates used in all IC digital logic families.



✦ Boolean laws

Identity Name	AND Form	OR Form
Identity Law	$1x = x$	$0+x = x$
Null (or Dominance) Law	$0x = 0$	$1+x = 1$
Idempotent Law	$xx = x$	$x+x = x$
Inverse Law	$x\bar{x} = 0$	$x+\bar{x} = 1$
Commutative Law	$xy = yx$	$x+y = y+x$
Associative Law	$(xy)z = x(yz)$	$(x+y)+z = x+(y+z)$
Distributive Law	$x+yz = (x+y)(x+z)$	$x(y+z) = xy+xz$
Absorption Law	$x(x+y) = x$	$x+xy = x$
DeMorgan's Law	$\overline{(xy)} = \bar{x}+\bar{y}$	$\overline{(x+y)} = \bar{x}\bar{y}$
Double Complement Law	$\bar{\bar{x}} = x$	

✦ Minterms and Maxterms

A	B	C	Z
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

$A+B+C$

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

$A+\bar{B}+C$

$A+B+\bar{C}$

$\bar{A}+B+C$

$A\bar{B}C$

$AB\bar{C}$

ABC

Maxterm

Minterm

✦ Standard forms in Boolean expressions

- **Sum of Product**

AB + ABC ABC +CDE + $\bar{B}\bar{C}\bar{D}$

- × A single overbar cannot extend over more than one Variable.
- × A standard SOP expression is one in which all the variables in the domain appear in each product term in the expression. Eg: A $\bar{B}$CD + $\bar{A}\bar{B}$C$\bar{D}$ + AB$\bar{C}\bar{D}$

- **Product of Sum**

(A+B)(A+B+C)($\bar{A}$+C) ($\bar{A}$+ $\bar{B}$+$\bar{C}$)(C+$\bar{D}$+E)(+ $\bar{B}$+C+D)

- × A single overbar cannot extend over more than one Variable.
- × A standard POS expression is one in which all the variables in the domain appear in each product term in the expression. Eg: (A+ $\bar{B}$ +C)($\bar{A}$ + $\bar{B}$ +C)(A+B+$\bar{C}$)

✦ Transforming SOP into POS and vice versa

SOP TO POS

$F = A.\bar{B} + B.\bar{C} + \bar{A}.C$

$= \overline{A.\bar{B} + B.\bar{C} + \bar{A}C}$ — *Obtain the compliment of whole equation by adding an over bar to the whole*

$= \overline{A.\bar{B}} + \overline{B.\bar{C}} + \overline{\bar{A}C}$ — De Morgan's Law

$= (\bar{A}+\bar{\bar{B}}).(\bar{B}+\bar{\bar{C}})+(\bar{\bar{A}}+\bar{C})$ — De Morgan's Law

$= (\bar{A}+B).(\bar{B}+C).(A+\bar{C})$ — Double Inverse

POS TO SOP

$F = (\bar{A}+B).(\bar{B}+C).(A+\bar{C})$

$= \overline{(\bar{A}+B).(\bar{B}+C).(A+\bar{C})}$ — *Obtain the compliment of whole equation by adding an over bar to the whole*

$= \overline{(\bar{A}+B)}+\overline{(\bar{B}+C)}+\overline{(A+\bar{C})}$ — De Morgan's Law

$= (\bar{\bar{A}}.\bar{B})+(\bar{\bar{B}}.\bar{C})+(\bar{A}.\bar{\bar{C}})$ — De Morgan's Law

$= A.\bar{B} + B.\bar{C} + \bar{A}.C$ — Double Inverse

✦ Karnaugh Maps

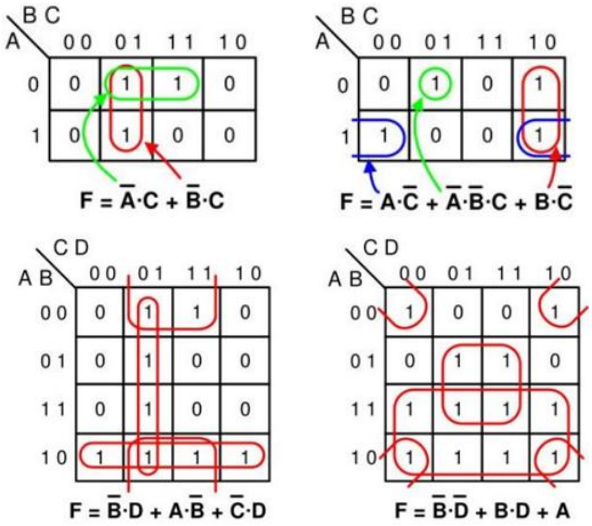
* An alternative way of simplifying logic circuits. Instead of using Boolean algebra simplification techniques

No. of cells = 2^n Where n is the number of input variables

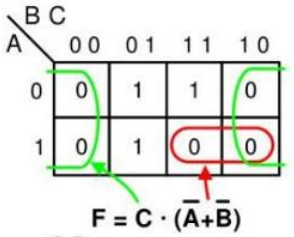
Rules of K- map simplification

1. No zeros allowed in a group.
2. No diagonals.
3. Only power of 2 number of cells in each group.
4. Groups should be as large as possible.
5. Every "one" must be in at least one group.
6. Overlapping allowed.
7. Wrap around allowed.
8. Fewest number of groups possible.

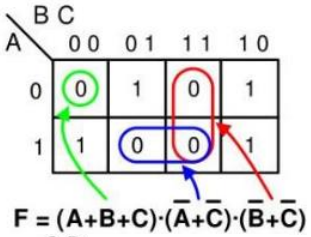
SOP Loops



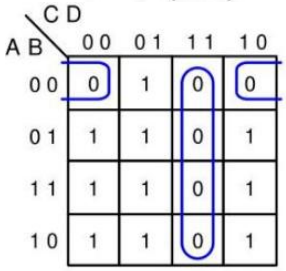
POS loops



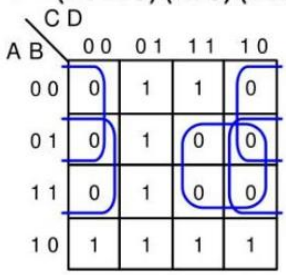
$F = C \cdot (\bar{A}+\bar{B})$



$F = (A+B+C)\cdot(\bar{A}+\bar{C})\cdot(\bar{B}+\bar{C})$



$F = (A+B+D)\cdot(\bar{C}+\bar{D})$

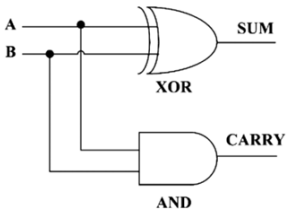


$F = (A+D)\cdot(\bar{B}+D)\cdot(\bar{B}+\bar{C})$

✦ **Half adder**

* used to add two single-bit binary numbers, producing two outputs: Sum (S) and Carry (C).
* sum is calculated using an XOR gate (S = A ⊕ B)
* carry is determined using an AND gate (C = A . B).

INPUTS		OUTPUTS	
A	**B**	**SUM**	**CARRY**
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1



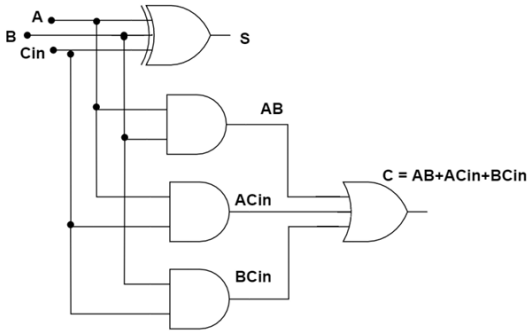
Combinational circuits Vs Sequential circuits

Feature	Combinational Circuits	Sequential Circuits
Dependence	Only on current inputs	On current inputs and previous states
Memory Elements	None	Present
Applications	Arithmetic operations, Data routing	Memory devices, Digital clocks, State machines
Examples	Adders, Multiplexers, Encoders, Decoders	Flip-flops, Counters, Registers

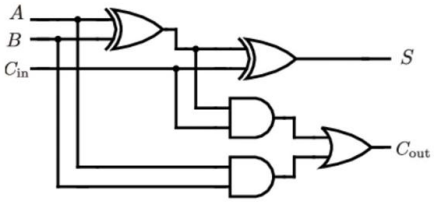
Full Adder

* full-adder has three inputs and two outputs.
* carry is designated as Carry Out (Cout = A . B . C)
* Sum (S = A⊕ B ⊕ C)

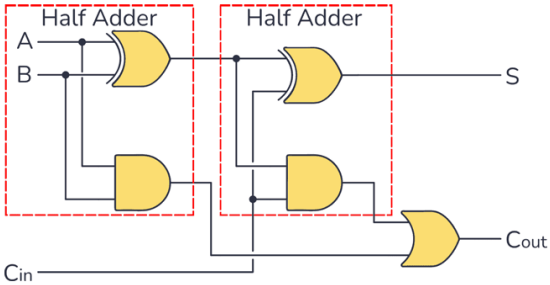
Inputs			Outputs	
A	B	C_{in}	S	C_{out}
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1



Method 2:



◆ **Full adder using two half adders**



✦ Sequential circuits

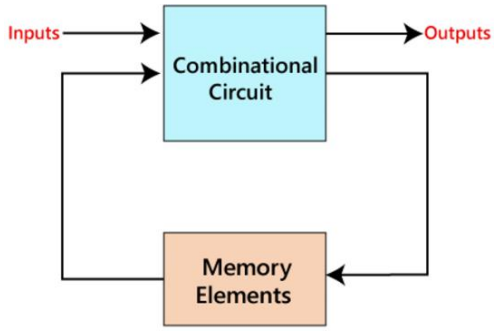


Figure – structure of a sequential circuit

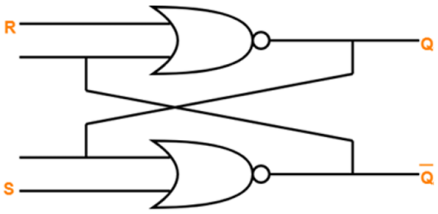
○ Flip Flops

* An application of logic gates
* Can create memory with them
* Can also be considered as the most basic idea of a Random-Access Memory.
* When a certain input value is given to them, they will be remembered.

○ Latch Flip-flop

* R-S (Reset Set) flip flop is the simplest flip flop of all and easiest to understand
* A device which has two outputs one output being the inverse or complement of the other, and two inputs.
* A pulse on one of the inputs to take on a particular logical state.
* The outputs will then remain in this state until a similar pulse is applied to the other input. The two inputs are called the Set and Reset input.

- **Latch constructed using NOR gates**

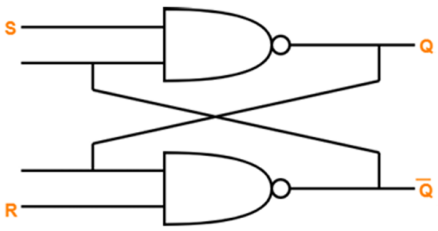


Truth Table

Set	Reset	Output
0	0	No change*
1	0	Q=1
0	1	Q=0
1	1	Invalid state

*can be used for data storage

- **Latch constricted using NAND gates**



Truth Table

Set	Reset	Output
1	1	No change*
0	1	Q=1
1	0	Q=0
0	0	Invalid state

*can be used for data storage