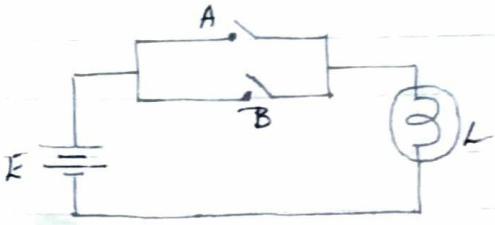
Logic Gates.

A gate is a digital circuit, with only one output and one or more inputs. The devices used in digital circuits generally operate in one of the two states, known as ON and OFF resulting in a very simple operation. The input and output signals of a gate have therefore 2 distinct values known as states. Any one state can be indicated by '0' and the other by '1'. Logic gates are of 2 types – Basic / Fundamental gates and Derived gates. The basic gates are OR, AND, NOT. These names indicate the function they perform.

Inclusive –OR gate :- This is a basic gate which gives a high output when any one or all inputs are high. The figure below shows a simple electrical circuit, to clear the concept of an OR gate.



OR gate using switches

A and B are two switches. 2 switches connected in parallel, placed in series with a source and lamp L is seen in the above circuit A switch can be either closed or open; Lamp L can be either dark or glowing.

When A and B are open, circuit is incomplete and lamp dark.

When one switch is open and other closed,

circuit is completed, and lamp glows.

With both switches closed also, ~~gto~~ lamp glows. Indicating open condition of switch and dark condition of lamp by binary '0' and closed condition of switch, glow condition of lamp by binary '1', we get the following table

A	B	L
0	0	0
0	1	1
1	0	1
1	1	1

Such a table is called truth table since it shows all the possible input combinations and the corresponding outputs.

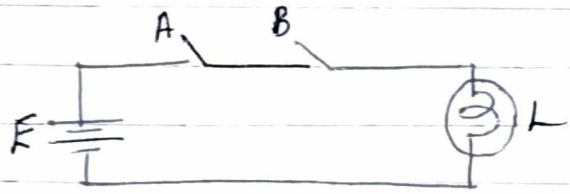
In Boolean Algebra notation, the OR gate is expressed as $Y = A + B$ where Y is the output and A, B are the inputs of the OR gate. This equation is called as Boolean equation of the OR gate and it is pronounced as "Y equals A OR B".

Symbol

For 3 input OR gate, $Y = A + B + C$

Symbol.

AND gate :- This is a basic gate which gives a high output when only all its inputs are high.



A and B are two switches connected in series.

With A and B open, circuit is incomplete and lamp dark. Even with one switch open and other closed, lamp is dark. The circuit is completed only when both the switches are closed and ∴ lamp glows.

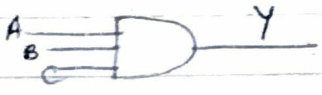
Indicating open condition of switch and dark condition of lamp by binary '0'; closed condition of switch and glow condition of lamp by binary '1', we have the following truth table.

A	B	L
0	0	0
0	1	0
1	0	0
1	1	1

The Boolean equation is $Y = A \cdot B$ and is pronounced as "Y equals A AND B"

Symbol

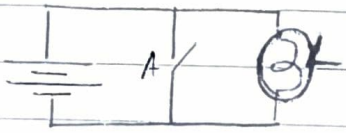
For a 3-input AND, $Y = A \cdot B \cdot C$



Symbol.

NOT gate :

This gate has only one input and one output terminal. It gives an output which is exactly opposite to the input. Hence also called inverter gate.



NOT gate using a switch.

With switch A open ~~is~~, current flows through lamp and it glows. When A is closed, battery is short-circuited and no current is supplied to the lamp. Hence lamp is dark.

If open condition of switch and ~~glo~~ dark condition of lamp is represented by '0' with closed condition of switch and glowing condition of lamp represented by '1', we get the following truth table of a NOT gate.

A	L
0	1
1	0

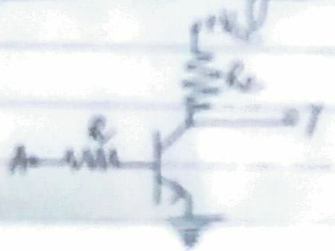
Looking at the above truth table, we can justify the name given to the gate. The output

is NOT equal to the input or the gate inverts the input given to it. In other words, it is said that, the input is complemented and then appears at the output.

Symbol

Boolean equation is $Y = \bar{A}$ and is read as "Y equals NOT A" or "Y equals complement of A".

Transistor NOT gate :



A is the input voltage and Y is the output voltage ie the collector voltage of transistor.

When A is low, transistor is cut off. No current through R_c and collector voltage is therefore high. ie output voltage Y is high

When A is high, transistor saturates. Current flows through R_c, voltage is dropped across it and output voltage Y is therefore low.

Low voltage = '0', High voltage = '1'

A	Y
0	1
1	0

Positive and Negative Logic :-

In a positive logic system, the more positive of the two voltage levels is represented by 1 and the other by 0. ie if the voltage levels are 0V and +5V, then according to positive logic, 5V will be represented by 1 and 0V by 0.

In a negative logic system, the more negative of the two voltage levels is represented by 1 and the other by 0. ie 1 will now represent 0V and 0 will represent 5V.

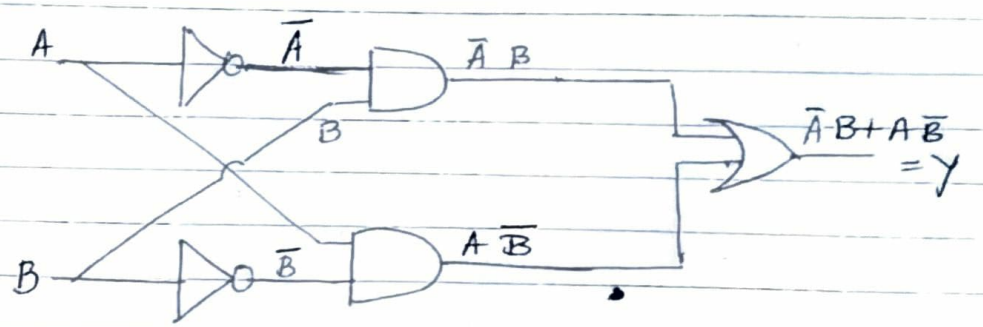
Derived gates

Exclusive-OR gate (EX-OR)

This is a derived gate which can be constructed using basic gates.

The Boolean equation of this gate is $Y = \bar{A}B + A\bar{B}$ where A, B are inputs and Y is the output.

The circuit diagram and truth table is as shown below



A	B	$\bar{A}$	$\bar{B}$	$\bar{A}B$	$A\bar{B}$	$Y=\bar{A}B+A\bar{B}$
0	0	1	1	0	0	0
0	1	1	0	1	0	1
1	0	0	1	0	1	1
1	1	0	0	0	0	0

From the above truth table, it can be seen that the output Y is high only when A and B are different from each other. When A and B are equal, either 0 or 1, the output is low. In other words, it can be said that the output is high only when inputs A and B are exclusive of each other. Since the first three combinations of A and B gives the same output as that obtained for an OR gate, the gate is called EXCLUSIVE-OR gate.

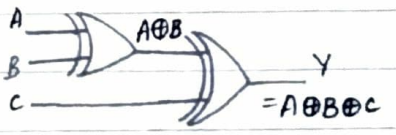
Symbol.

The Boolean equation for this gate can also be written in the form $Y = A \oplus B$ and is read as "Y equals A EX-OR B".

For a three-input EX-OR, boolean equation is $Y = A \oplus B \oplus C$ where A, B, C are the 3 inputs.

Truth table

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



III

Symbol

This table tells us that an EX-OR gate gives high output only when the input has an odd number of 1's.

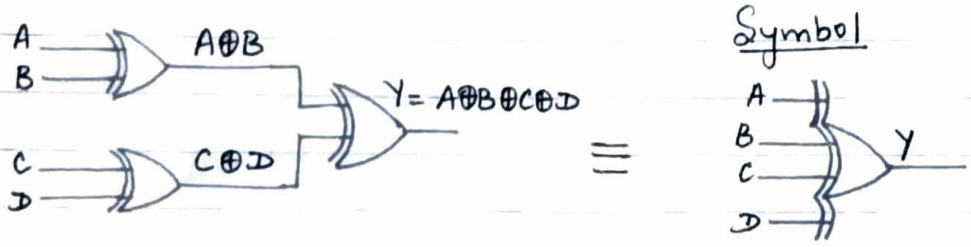
Even parity means an n-bit input has an even number of 1's while odd parity means input has odd number of 1's.

eg. 110001 — odd parity with three 1's in it.
111100 — even parity with four 1's in it.

PARITY CHECKER: EX-OR gates are ideal for checking the parity of a binary number, because they produce an output 1 when the input has an odd number of 1's.

Therefore, an even-parity input to an EX-OR gate produces a low output while an odd-parity input produces a high output.

For eg. consider the 4-input EX-OR gate shown below.



A 4-bit number drives the input, say 1101 The EX-OR gate produces an output 1 because the input has odd parity (an odd number of 1's). If the 4- bit number is changed to say 1001, then output of EX-OR gate goes low, indicating even parity.

EX-NOR gate :-

Boolean equation is $Y = \overline{A \oplus B}$. Circuit and truth table is as follows.

Truth table

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

An EX-NOR gate is constructed by

introducing a NOT gate at the output of an EX-OR gate. This gate gives a high output only when both the inputs are the same.

NAND gate

This is a derived gate which gives a high output when one input to it is low.

NAND gate is a combination of NOT and AND gates. The NOT gate is connected after the AND gate and then the output is noted. Naturally, the output is complement of an AND gate output. Thus output is NOT-AND or in short, the gate is called NAND.

Boolean equation is $Y = \overline{AB}$ where A, B are inputs and Y, the output.

Symbol

Truth table :

A	B	$Y = \overline{AB}$
0	0	1
0	1	1
1	0	1
1	1	0

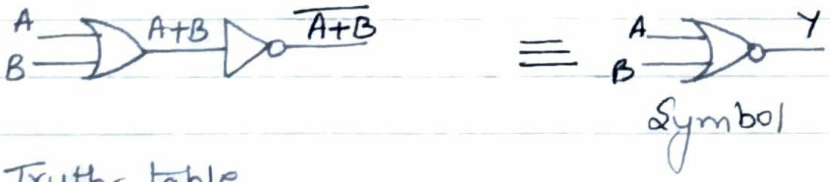
NOR gate

It is a derived gate which gives a

high output only when both inputs are low. With one input high, output goes low.

This gate is derived from an OR gate and its output is the complement of an OR gate output. ie the output is NOT-OR. In short, the gate is called NOR.

The boolean equation is $Y = \overline{A+B}$ where A, B are the inputs and Y, the output



Symbol

Truth table

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

Laws of Boolean Algebra

Boolean Algebra is a system of mathematics, dealing with binary numbers and notations. It has a few fundamental laws.

(a) Laws of complementation

(i) If $A = 0$, $\overline{A} = \overline{0} = 1$

(ii) If $A = 1$, $\overline{A} = \overline{1} = 0$

(iii) $\overline{\overline{A}} = A$.

(b) AND laws

(i) $A \cdot 0 = 0$

(ii) $A \cdot 1 = A$

(iii) $A \cdot A = A$

(iv) $A \cdot \bar{A} = 0$

(c) OR laws

(i) $A + 0 = A$

(ii) $A + 1 = 1$

(iii) $A + A = A$

(iv) $A + \bar{A} = 1$

(d) Commutative laws

(i) $A + B = B + A$

(ii) $AB = BA$

(e) Associative laws

(i) $A + (B + C) = (A + B) + C$

(ii) $A(BC) = (AB)C$

(f) Distributive law

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

In Boolean Algebra, multiplication indicates AND operation, addition indicates OR operation and bar on the top indicates inversion ie NOT operation.

De Morgan's Theorems : There are 2 theorems stated by De Morgan. These form a powerful tool for conversion of circuits into different form or to convert multiplication into sum and vice versa.

First theorem : It states that "The complement of a sum equals the product of complements"

In terms of Boolean expression the statement can be written as $\overline{A+B} = \bar{A} \cdot \bar{B}$

Proof : LHS : $\overline{A+B}$

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

RHS $\bar{A} \cdot \bar{B}$

A	B	$\bar{A}$	$\bar{B}$	$\bar{A} \cdot \bar{B}$
0	0	1	1	1
0	1	1	0	0
1	0	0	1	0
1	1	0	0	0

Observing the two truth tables, we conclude that $\overline{A+B}$ is equal to $\bar{A} \cdot \bar{B}$ since the two tables are identical. The circuits shown below are therefore logically equivalent and interchangeable.

Second Theorem :- It states that "The complement of a product is equal to the sum of the complements" ie $\overline{A \cdot B} = \bar{A} + \bar{B}$

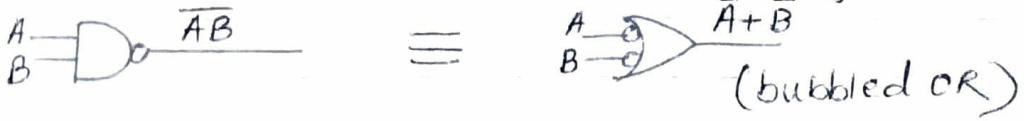
Proof : LHS $\overline{A \cdot B}$

A	B	$\overline{AB}$
0	0	1
0	1	1
1	0	1
1	1	0

RHS : $\bar{A} + \bar{B}$

A	B	$\bar{A}$	$\bar{B}$	$\bar{A} + \bar{B}$
0	0	1	1	1
0	1	1	0	1
1	0	0	1	1
1	1	0	0	0

Since the two truth tables are identical, the 2nd theorem is proved. The circuits below are therefore equivalent and interchangeable.



It therefore follows from the above stated theorems that $\overline{A+B+C} = \bar{A} \cdot \bar{B} \cdot \bar{C}$

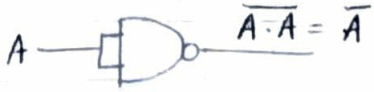
and $\overline{A \cdot B \cdot C} = \bar{A} + \bar{B} + \bar{C}$.

NAND as a Universal building block.

OR, AND, NOT are the basic building blocks of all logic circuits. NAND has an interesting property. It can be used to build an OR, AND and NOT gate. Hence NAND gates are called universal building blocks.

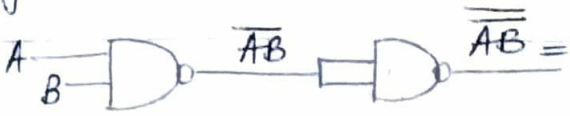
(1) NAND as a NOT gate :-

If both inputs to the NAND gate are connected to one another, with this short serving as the single input, then the output gives inversion.



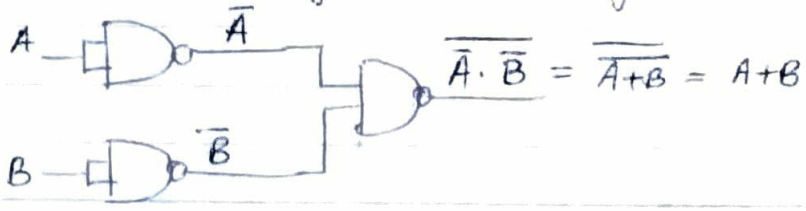
(2) NAND as AND gate :-

NAND is an inversion of an AND gate. Thus one more inversion will result in an AND gate.



(3) NAND as OR gate :-

Using De Morgan's theorem, NAND gates can be connected to give an OR gate.



NOR as a Universal building block

NOR gates can also be used to construct the three basic gates OR, AND, NOT. Hence NOR is also called an universal building block.

(1) NOR as a NOT gate :-

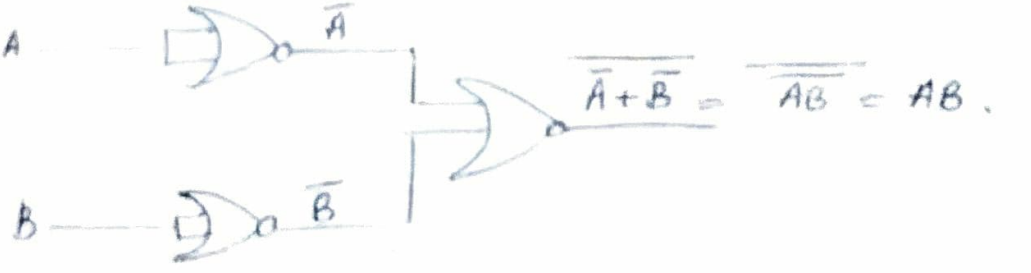
To construct a NOT gate, both the inputs of the NOR gate are connected to each other and this short serves as the single input. The combination then gives inversion

(2) NOR as a OR gate -

NOR is an inversion of an OR gate. One more inversion will result in an OR gate.

(3) NOR as a AND gate :-

This gate can be constructed using NOR in the following manner, making use of DeMorgans theorem.

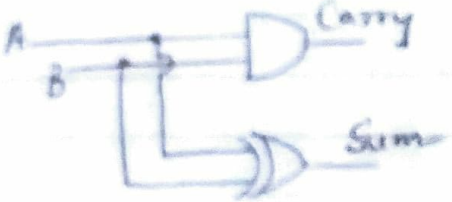


Half Adder

This is a logic circuit that adds two bits. It has two outputs, sum and carry. When two bits are added, we have the following four combinations. Suppose the two digits are A and B

A	B	Sum	Carry
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

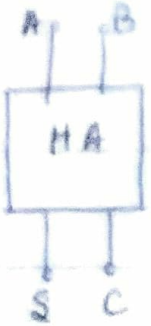
The rules used are those of binary addition. By looking at the above table, we note that, the output of carry corresponds to the output obtained in an AND gate and that of sum, corresponds to the output of an EX-OR gate. Thus for adding 2 binary digits, a circuit can be build using AND and EX-OR gates. Such a circuit is called as HALF ADDER and is shown below



The boolean equations are

Carry $= A \cdot B$, Sum $= A \oplus B$.

Symbol



Working :-

Suppose A = 1, B = 0. For AND gate, one input is low and hence its output is low. ie CARRY = 0. For EXOR, with inputs 1 and 0, output = 1 ie SUM = 1.

Suppose A = 1, B = 1. AND gate gives high output with both inputs high and EXOR gate gives low output with even number of 1's in input.

Full Adder :-

This is a logic circuit that can add 3 bits. When we perform addition of 3 bits, say P, Q, R, we have the following 8 combinations and 2 outputs, Sum and carry. The rules used are those of binary addition

P	Q	R	(S) SUM	(C) CARRY
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

Boolean equations are

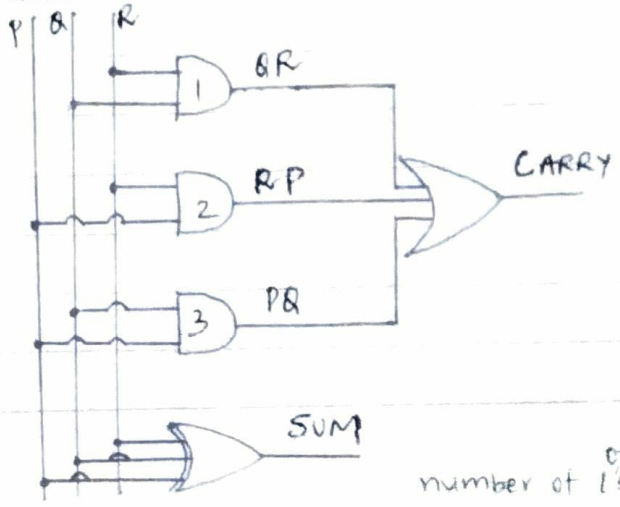
$S = \bar{P}\bar{Q}R + \bar{P}Q\bar{R} + P\bar{Q}\bar{R} + PQR$

or $S = P \oplus Q \oplus R$

$C = \bar{P}QR + P\bar{Q}R + PQ\bar{R} + PQR$

or $C = PQ + QR + RP$.

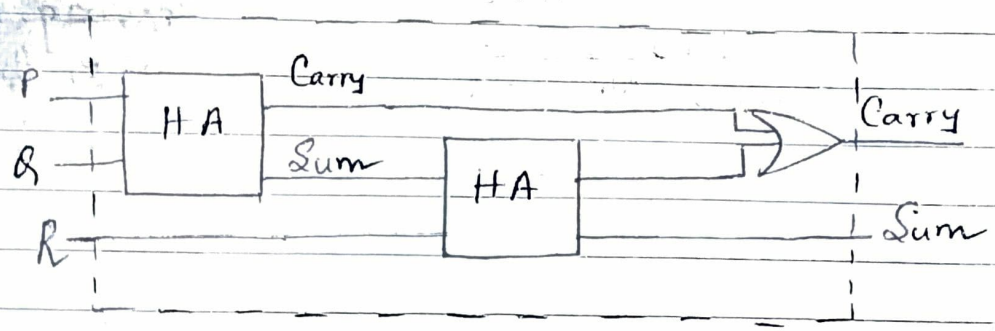
Thus, for adding 3 bits, the following circuit can be used. This is one of the methods of constructing a full adder.



Working :- Suppose P=Q=1, R=0 AND1 gives low output, along with AND2 AND3 gives high output. ∴ with one input high, OR gate gives high output ie CARRY = 1. With even number of 1's in input, EXOR gives low output. ie SUM = 0.

With P=Q=R=1, all AND gates give high output, so that output of OR gate ie CARRY = 1. for even number of 1's in input, EXOR gives high output

A full adder can be seen as a combination of 2 half adders and an OR gate as shown below.

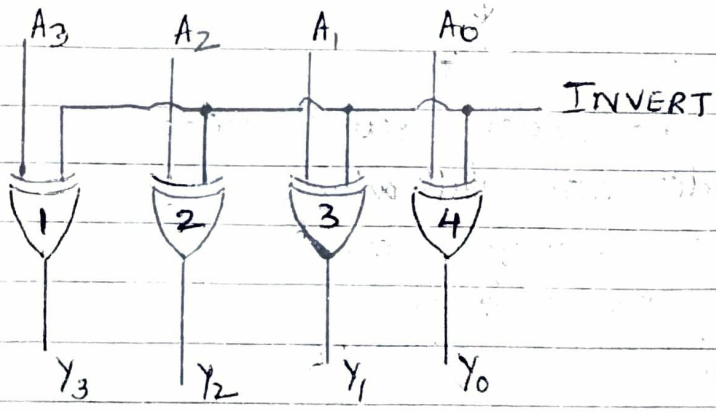


Hence, this is another method of constructing a full adder.

Symbol

Controlled Inverter

The figure below shows a controlled inverter.



A controlled inverter provides inversion when certain condition is satisfied.

If INVERT is low, no inversion occurs and the binary number is transmitted as it is to the output. ie $Y_3 Y_2 Y_1 Y_0 = A_3 A_2 A_1 A_0$

However if INVERT is high, the 1's complement of the binary number is transmitted to the output
ie $Y_3 Y_2 Y_1 Y_0 = \bar{A}_3 \bar{A}_2 \bar{A}_1 \bar{A}_0$

eg.

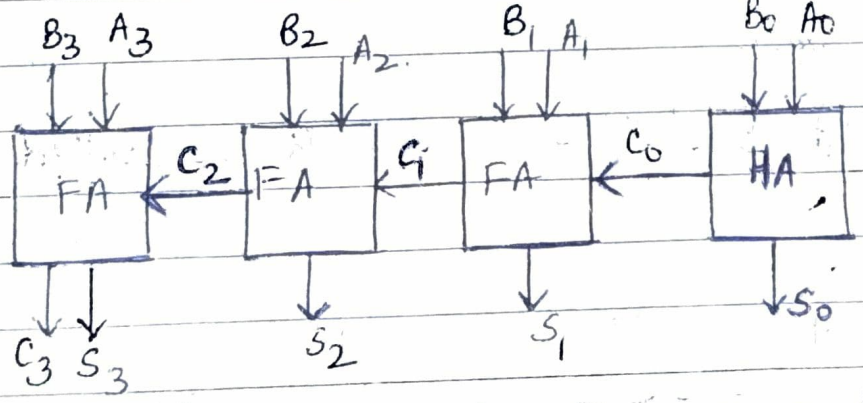
	Binary number				Output			
INVERT	A_3	A_2	A_1	A_0	Y_3	Y_2	Y_1	Y_0
0	0	1	1	0	0	1	1	0
1	0	1	1	0	1	0	0	1
0	0	0	0	1	0	0	0	1
1	0	0	0	1	1	1	1	0

This is because ~~an~~ EXOR gates ~~is~~ are used. EXOR gate gives high output when there are odd number of high inputs to it. So, if INVERT = 1 and $A_3 A_2 A_1 A_0$ = 0001, then EXOR 1, EXOR 2 EXOR 3 give high outputs

since they have odd number of high inputs. EXOR 4 has both inputs high and gives low output
$\therefore Y_3 Y_2 Y_1 Y_0 = 1110$

4-bit binary adder :

$$\begin{array}{r} A_3\ A_2\ A_1\ A_0 \\ +\ B_3\ B_2\ B_1\ B_0 \\ \hline C_3\ S_3\ S_2\ S_1\ S_0 \end{array}$$

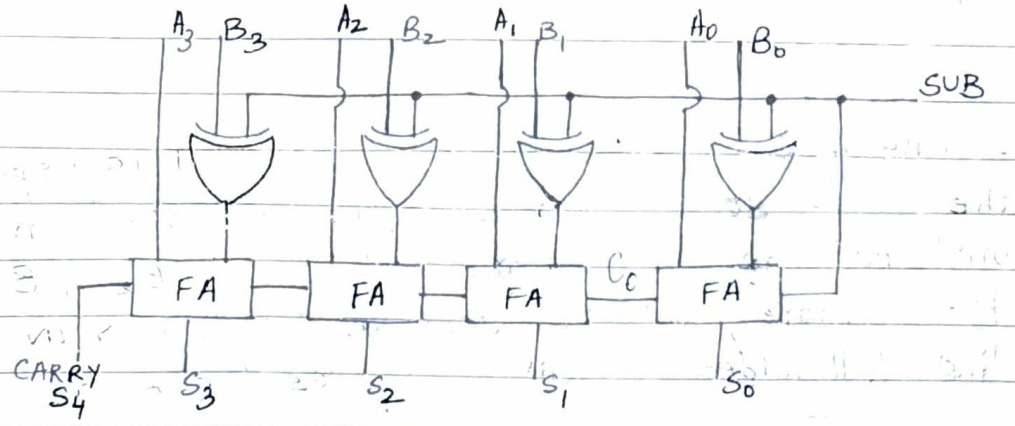


An adder circuit using a combination of half adder and full adders for addition of two 4-bit binary numbers is shown above. A and B are the two 4-bit inputs to be added. A half adder may be used to add the least-significant bits A_0 and B_0. The carry C_0 is given as input

to a full adder along with A_1 and B_1. The carry C_1 obtained is given to another full adder along with A_2 and B_2.

4 bit adder / subtractor

We can connect full adders as shown below to add or subtract binary numbers. The circuit is laid out from right to left, similar to the way we add binary numbers. Therefore, the LSB column is on the right and the MSB column is on the left.



The CARRY OUT from each full adder is the CARRY IN to the next-higher full adder. The numbers being processed are $A_3 A_2 A_1 A_0$ and $B_3 B_2 B_1 B_0$ while the answer is $S_3 S_2 S_1 S_0$.

ADDITION :- Here is how an addition appears :

$$\begin{array}{r} A_3\ A_2\ A_1\ A_0 \\ +\ B_3\ B_2\ B_1\ B_0 \\ \hline S_4\ S_3\ S_2\ S_1\ S_0 \end{array}$$

During an addition, the SUB signal is kept in the low state. Therefore the binary number $B_3 B_2 B_1 B_0$ passes through the controlled inverter with no change. The full adders then produce the correct output

sum. They do this by adding the bits in each column, passing carries to the next higher column and so on. Starting, for instance, at the LSB, the full adder adds A_0, B_0 and SUB. This produces a SUM of S_0 and a CARRY OUT to the next-higher full adder. The next higher full adder then adds A_1, B_1 and the CARRY IN to produce S_1 and a CARRY OUT. A similar addition occurs for each of the remaining full adders and the correct sum appears at the output lines.

SUBTRACTION : Here is how an subtraction appears:

$$\begin{array}{r} A_3\ A_2\ A_1\ A_0 \\ -\ B_3\ B_2\ B_1\ B_0 \\ \hline S_3\ S_2\ S_1\ S_0 \end{array}$$

During a subtraction, the SUB signal is kept in the high state. Therefore, the controlled inverter produces the 1's complement of $B_3\ B_2\ B_1\ B_0$. Furthermore, because SUB is the CARRY IN to the first full adder, circuit processes the data like this:

$$\begin{array}{r} \qquad\qquad\qquad\qquad 1 \leftarrow \text{SUB} \\ A_3\ A_2\ A_1\ A_0 \\ +\ \overline{B_3}\ \overline{B_2}\ \overline{B_1}\ \overline{B_0} \\ \hline S_3\ S_2\ S_1\ S_0 \end{array}$$

ie the effect is equivalent to adding $A_3 A_2 A_1 A_0$ and the 2's complement of $B_3 B_2 B_1 B_0$. Carry S_4 is ignored and $S_3 S_2 S_1 S_0$ gives the final answer.

— X —