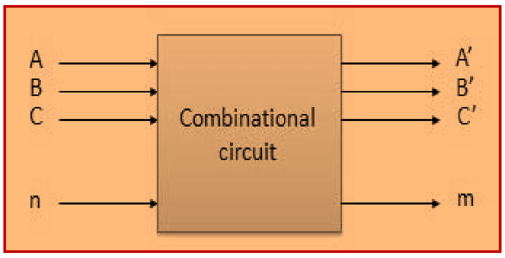
Section: 3.1
Features of combinational circuits and examples. Half adder (HA): Concept, truth-table, logical expression, gate-level implementation and application.

Combinational circuits

- A combinational circuit is the digital logic circuit in which the output depends on the combination of inputs at that point of time with total disregard to the past state of the inputs.
- The digital logic gate is the building block of combinational circuits.The function implemented by combinational circuit is depending upon the Boolean expressions.
- On the other hand, sequential logic circuits, consists of both logic gates and memory elements such as flip-flops.
- Figure below shows the combinational circuit having n inputs and and m outputs. The n number of inputs shows that there are 2^n possible combinations of bits at the input. Therefore, the output is expressed in terms m Boolean expressions.

Features of combinational circuits are

- The output of combinational circuit at any instant of time depends only on the levels present at input terminals.
- The combinational circuit does not use any memory. The previous state of input does not have any effect on the present state of the circuit.



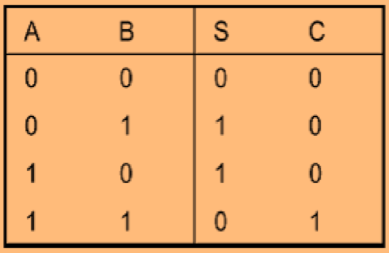
- A combinational circuit can have an n number of inputs and m number of outputs.

Examples :

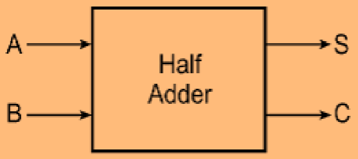
Adders, Subtractors, Encoders, Decoders & Multiplexers etc…

Half adder (HA):

- A half-adder is an arithmetic circuit block that can be used to add two bits. Such a circuit thus has two inputs that represent the two bits to be added and two outputs, with one producing the SUM output and the other producing the CARRY.
- The below figure shows the truth table of a half-adder, showing all possible input combinations and the corresponding outputs.



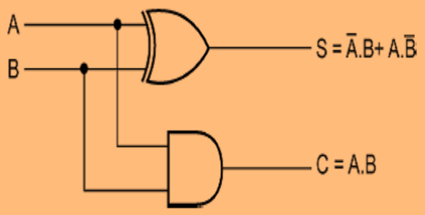
A	B	S	C
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1



- The Boolean expressions for the SUM and CARRY outputs are given by the equations

  SUM = S = A_B+A_B

  CARRY = C = A_B

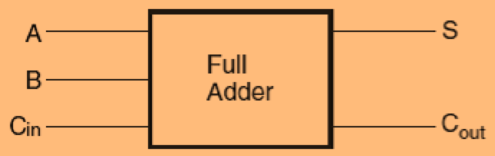


- An examination of the two expressions tells that there is no scope for further simplification. While the first one representing the SUM output is that of an EX-OR gate, the second one representing the CARRY output is that of an AND gate.

Section: 3.2
Full adder (FA): Concept, truth-table, logical expression, gate-level implementation and application. List of FA ICs.

Full adder (FA):

- The full-adder accepts two input bits and an input carry and generates a sum output and an output carry.
- The basic difference between a full-adder and a half-adder is that the full-adder accepts an input carry.
- The below figure shows the truth table of a full adder circuit showing all possible input combinations and corresponding outputs



A	B	C_{in}	SUM (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

The Boolean expressions for the two output variables for the SUM output (S) and for the CARRY output (Cout) are :

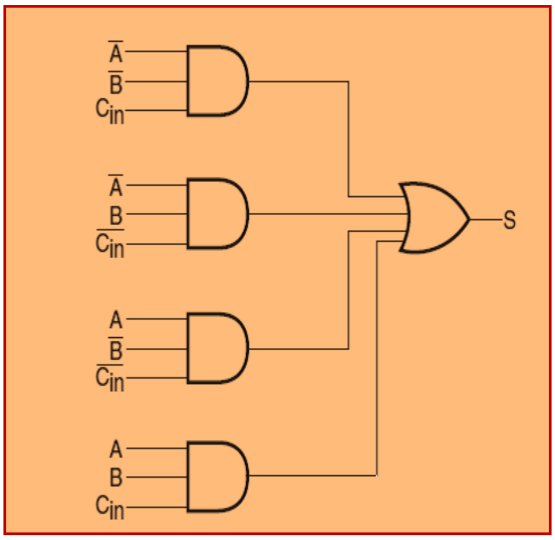
$$S = \overline{A}.\overline{B}.C_{in} + \overline{A}.B.\overline{C}_{in} + A.\overline{B}.\overline{C}_{in} + A.B.C_{in}$$

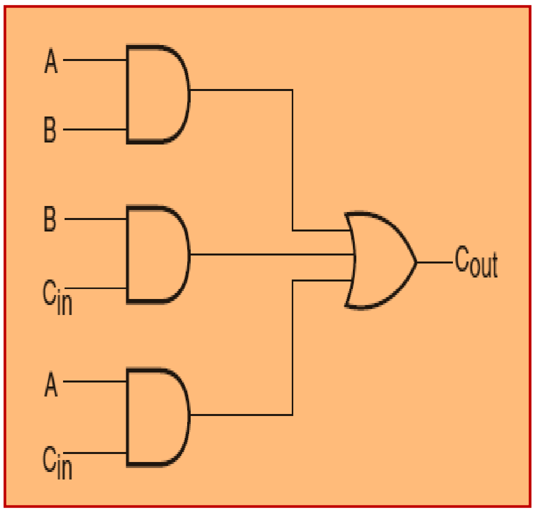
$$C_{out} = \overline{A}.B.C_{in} + A.\overline{B}.C_{in} + A.B.\overline{C}_{in} + A.B.C_{in}$$

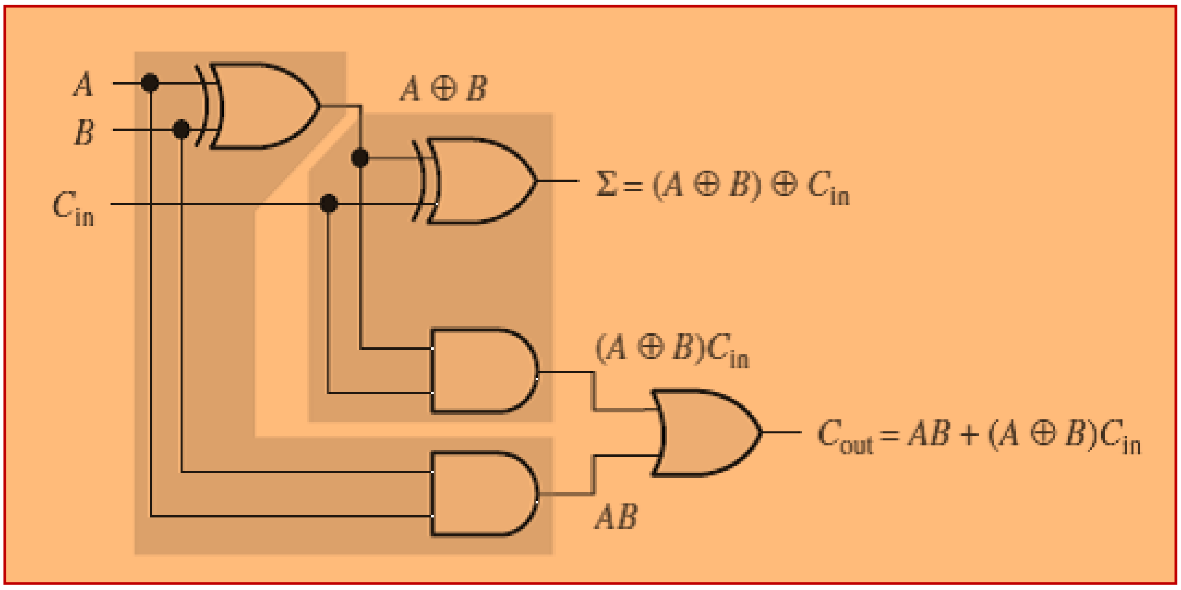
$$S = \overline{C}_{in}.(\overline{A}.B + A.\overline{B}) + C_{in}.(A.B + \overline{A}.\overline{B})$$

$$S = \overline{C}_{in}.(\overline{A}.B + A.\overline{B}) + C_{in}.(\overline{\overline{A}.B + A.\overline{B}})$$

$$C_{out} = A.B + C_{in}.(\overline{A}.B + A.\overline{B})$$

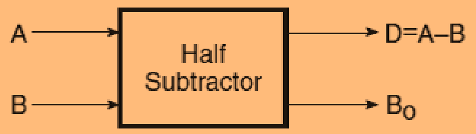

$\overline{A}$
$\overline{B}$
C_{in}
$\overline{A}$
B
$\overline{C_{in}}$
A
$\overline{B}$
$\overline{C_{in}}$
A
B
C_{in}
S


A
B
B
C_{in}
A
C_{in}
C_{out}


A
B
C_{in}
$A \oplus B$
$\Sigma = (A \oplus B) \oplus C_{in}$
$(A \oplus B)C_{in}$
$C_{out} = AB + (A \oplus B)C_{in}$
AB

Section: 3.3
Half Subtractor (HS): Concept, truth-table, logical expression, gate-level implementation and application.

- Half-subtractor is a combinational circuit that can be used to subtract one binary digit from another to produce a DIFFERENCE output and a BORROW output.
- The BORROW output here specifies whether a ‘1’ has been borrowed to perform the subtraction.
- The truth table of a half-subtractor, as shown below, explains this further.



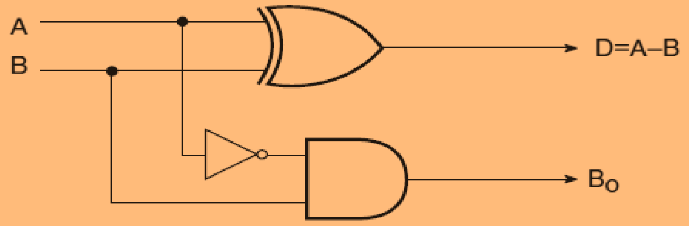
A	B	D	B_o
0	0	0	0
0	1	1	1
1	0	1	0
1	1	0	0

- The Boolean expressions for the two outputs are given by the equations

$$D = \overline{A}.B + A.\overline{B}$$

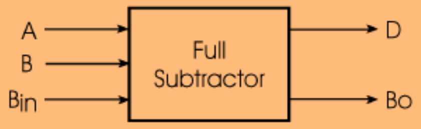
$$B_o = \overline{A}.B$$

- It is obvious that there is no further scope for any simplification of the Boolean expressions given.
- While the expression for the DIFFERENCE (D) output is that of an EX-OR gate, the expression for the BORROW output (Bo) is that of an AND gate with input a complemented before it is fed to the gate.
- The below figure shows the logic implementation of a half-subtractor.



<u>Section: 3.4</u>
<u>Full Subtractor (FS): Concept, truth-table, logical expression, gate-level implementation and application.</u>

- A full subtractor performs subtraction operation on two bits, a minuend and a subtrahend, and also takes into consideration whether a ‘1’ has already been borrowed by the previous adjacent lower minuend bit or not. As a result, there are three bits to be handled at the input of a full subtractor, namely the two bits to be subtracted and a borrow bit designated as Bin .
- There are two outputs, namely the DIFFERENCE output D and the BORROW output Bo.
- The BORROW output bit tells whether the minuend bit needs to borrow a ‘1’ from the next possible higher minuend bit.
- Figure 7.14 shows the truth table of a full subtractor.



Minuend (A)	Subtrahend (B)	Borrow In (B_{in})	Difference (D)	Borrow Out (B_o)
0	0	0	0	0
0	0	1	1	1
0	1	0	1	1
0	1	1	0	1
1	0	0	1	0
1	0	1	0	0
1	1	0	0	0
1	1	1	1	1

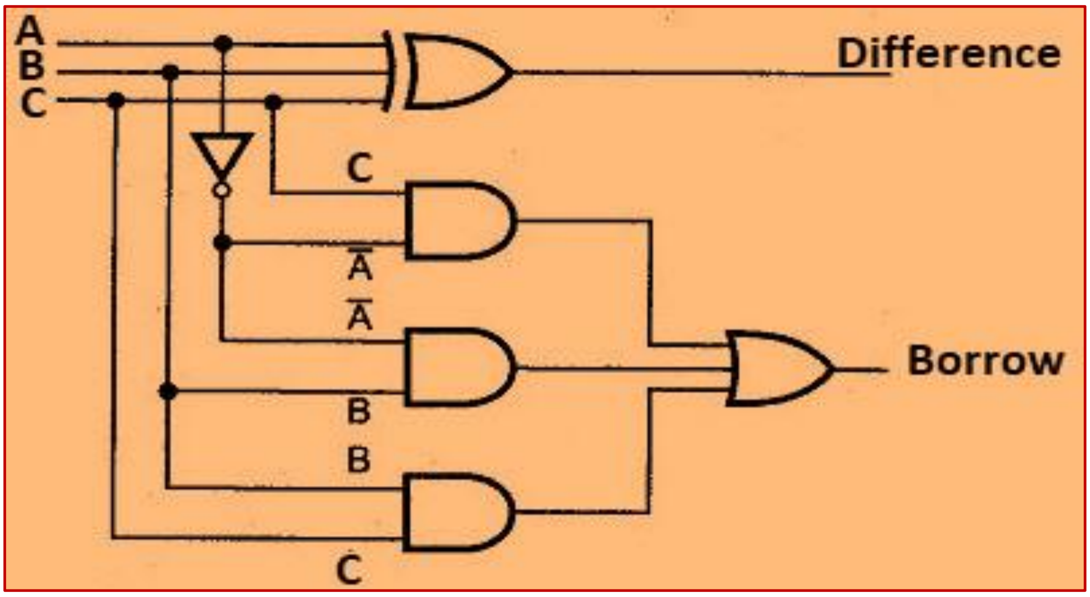
The Boolean expressions for the two output variables are given by the equations

$$D = \overline{A}.\overline{B}.B_{in} + \overline{A}.B.\overline{B}_{in} + A.\overline{B}.\overline{B}_{in} + A.B.B_{in}$$

$$B_o = \overline{A}.\overline{B}.B_{in} + \overline{A}.B.\overline{B}_{in} + \overline{A}.B.B_{in} + A.B.B_{in}$$

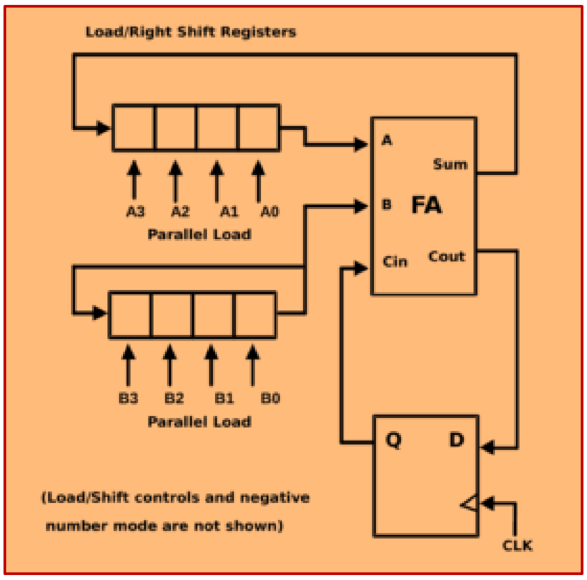
$$B_o = \overline{A}.B + \overline{A}.B_{in} + B.B_{in}$$

D = (A XOR B) XOR Bin



Subscribe RaviRnandi & download the documents by www.mathswithme.in

Section: 3.5
Serial & Parallel adders: Concept, comparison & applications.


Load/Right Shift Registers
A3 A2 A1 A0
Parallel Load
B3 B2 B1 B0
Parallel Load
A
B
Cin
Sum
Cout
FA
Q
D
CLK
(Load/Shift controls and negative number mode are not shown)

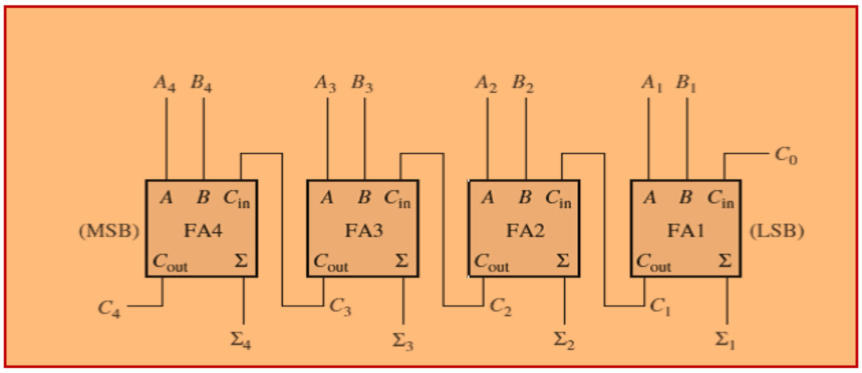
- Parallel adder performs the adding two bit operation very fast but the disadvantage of this adder is it's require large number of gate.
- One the other hand in serial adder the bit addition is bit-by-bit.
- Serial adder require simple circuitry as compare to parallel adder , so causes of simple circuitry this give low speed and perform bit-by-bit operation.
- In this diagram there is one D flip-flop one full adder and three register are given which operated by the clock pulse.
- Single full adder is use to add one pair of bits at a time along with carry.
- The D flip flop is carry flip-flop is used the carry.
- The contain of shift register shift from left to right and their output starting from A0 and B0 are fed in to a single full adder along with output of carry flip flop up on output application of each clock pulse.

Serial adder	**Parallel adder**
Serial adder is less fast.	Parallel adder is fast as compare to serial adder.
It requires fewer components for operation.	It require large component for operation.
Addition process is perform by bit-by-bit process	Addition process is performing by parallel order. Means all bits add simultaneously.
It requires one full adder circuit.	No. of full adder circuit is equal to no. of bits in binary adder.
Time required for addition depends on number of bits.	Time required does not depend on the number of bits'.

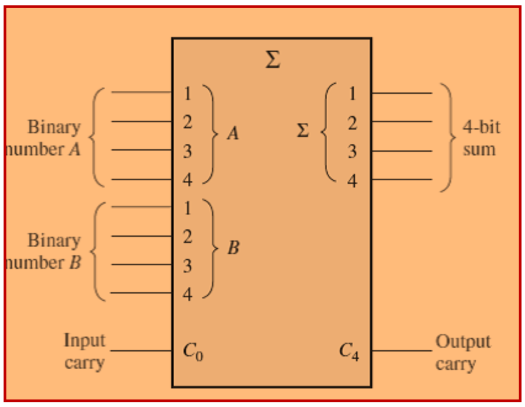
Section: 3.6
Three-bit parallel adder circuit: Given the circuit, analyze it's working.

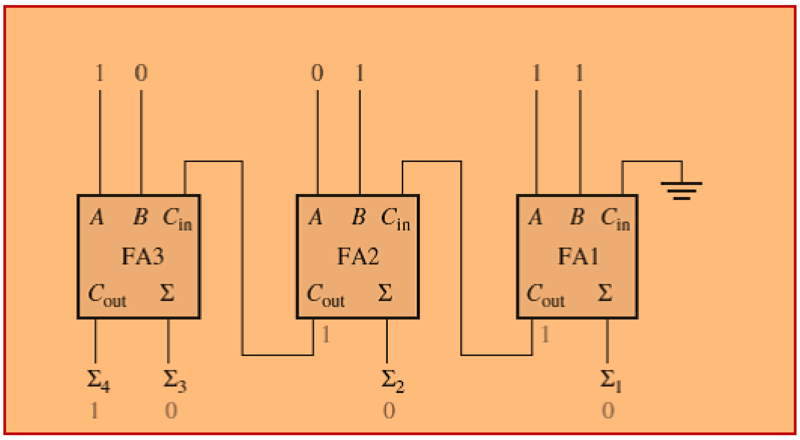
- In most logic circuit addition of more than 1- bit is carried out . for Example computer use 8 to 64 bit . The addition of multi bit can be accomplished using several adder.
- The 4 bit adder using full adder this is capable of adding two 4 bit number resulting in a 4 bit sum and a carry out put. Since all bit of the augends and addend are fed in to the adder circuit simultaneously and the addition in each position are taking place at the same time , this circuit is known as Parallel adder.
- Represented by A3 A2 A1 A0 =1111 and B3 B2 B1 B0 =0011.

Place	4 3 2 1
Carry	1 1 1 0
Augends word A	1 1 1 1
Addend word B	0 0 1 1



A_4 B_4 A_3 B_3 A_2 B_2 A_1 B_1
C_0
(MSB)
FA4
FA3
FA2
FA1
(LSB)
C_4 C_3 C_2 C_1
Σ_4 Σ_3 Σ_2 Σ_1





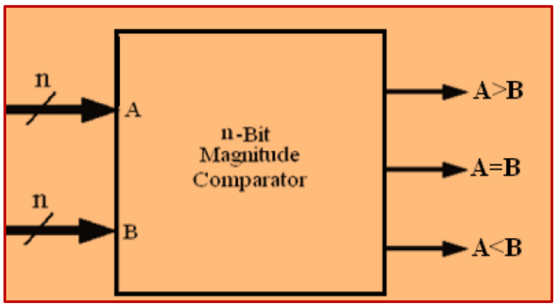
101 & 011

- A group of four bits is called a **nibble**.

- A basic 4-bit parallel adder is implemented with four full-adder stages as shown in the above figure.
- Again, the LSBs (A1 and B1) in each number being added go into the right-most full-adder; the higher-order bits are applied as shown to the successively higher-order adders, with the MSBs (A4 and B4) in each number being applied to the left-most full-adder.
- The carry output of each adder is connected to the carry input of the next higher-order adder as indicated. These are called internal carries.
- Though the parallel binary adder is said to generate its output very fast when input is applied its speed operation is limited by the carry propagation delay through all stage.
- In each full adder the carry input has to be generated from the previous full adder which has an inherent propogation delay.
- The propagation delay of full adder is the time different between the instant at which the input are applied and the instant at which is output are generated.

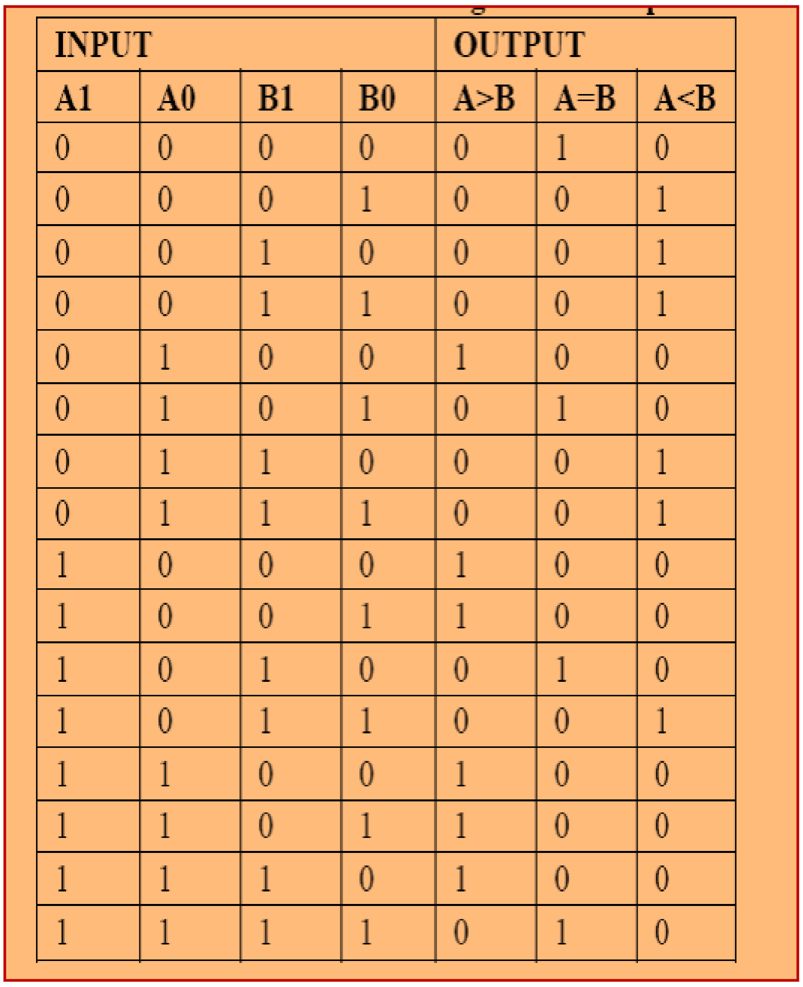
Section: 3.7
Two-bit magnitude comparator: Problem, truth-table, logical expression, gate-level implementation and application. Identify ICs

- In digital system, comparison of two numbers is an arithmetic operation that determines if one number is greater than, equal to, or less than the other number. So comparator is used for this purpose.



- Magnitude comparator is a combinational circuit that compares two numbers, A and B, and determines their relative magnitudes
- The outcome of comparison is specified by three binary variables that indicate whether **A>B, A=B, or A<B**

- 2-Bit Magnitude Comparator Compares two numbers each having two bits (A1, A0 & B1, B0).



INPUT				OUTPUT		
A1	A0	B1	B0	A>B	A=B	A<B
0	0	0	0	0	1	0
0	0	0	1	0	0	1
0	0	1	0	0	0	1
0	0	1	1	0	0	1
0	1	0	0	1	0	0
0	1	0	1	0	1	0
0	1	1	0	0	0	1
0	1	1	1	0	0	1
1	0	0	0	1	0	0
1	0	0	1	1	0	0
1	0	1	0	0	1	0
1	0	1	1	0	0	1
1	1	0	0	1	0	0
1	1	0	1	1	0	0
1	1	1	0	1	0	0
1	1	1	1	0	1	0

The Equation for the 2 bit Magnitude Comparator is :

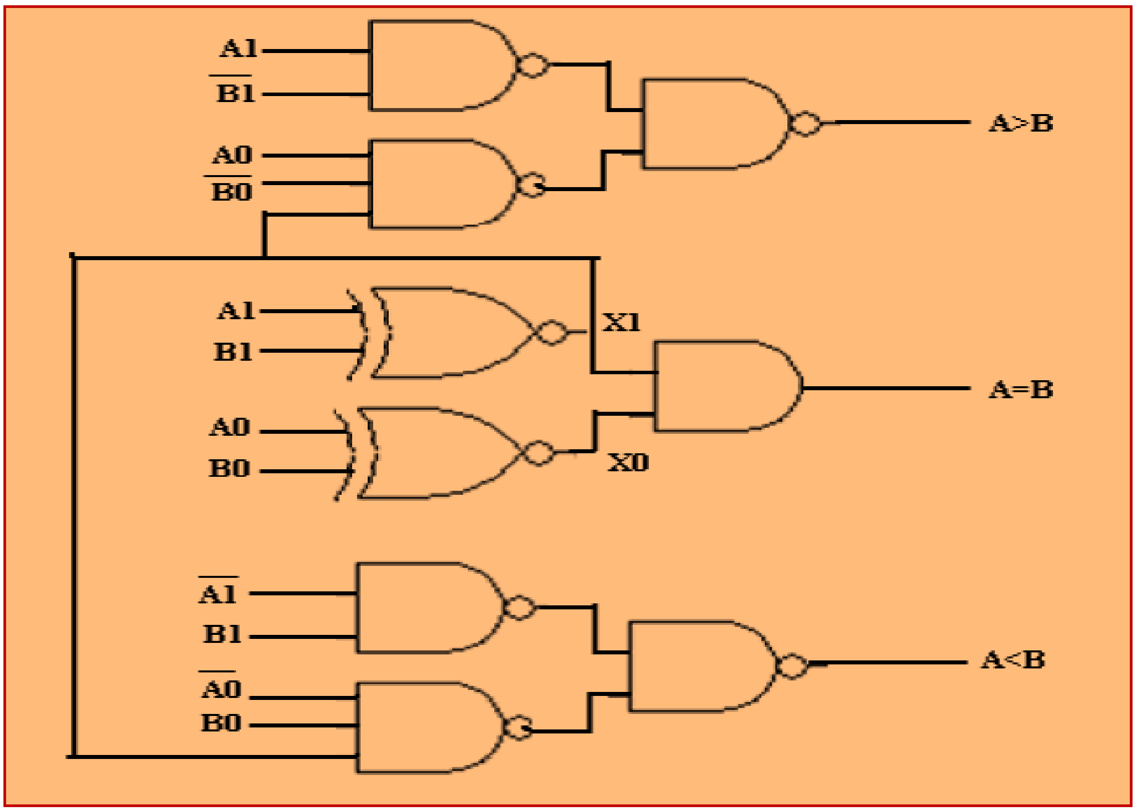
$$\begin{aligned} A>B: &= A1B1'+A0B0'A1'B1'+A0B0'A1B1 \\ &= A1B1'+A0B0'(A1'B1'+A1B1) \\ &= A1B1'+A0B0'\,X1 \end{aligned}$$

B0

$$\begin{aligned} A=B: &= A1'A0'B1'B0'+A1'A0B1'B0+A1A0'B1B0'+A1A0B1B0 \\ &= (A1'B1'+A1B1)(A0'B0'+A0B0) \\ &= X1X0 \end{aligned}$$

$$\begin{aligned} A<B: &= A1'B1+A0'B0A1'B1'+A0'B0A1B1 \\ &= A1'B1+A0'B0(A1'B1'+A1B1) \\ &= A1'B1+A0'B0\,X1 \end{aligned}$$

According to the logic function obtained from the truth table, logic diagram is drawn as follows:



A1
$\overline{B1}$
A0
$\overline{B0}$
A>B
A1
B1
X1
A0
B0
X0
A=B
$\overline{A1}$
B1
$\overline{A0}$
B0
A<B