Unit-4
Combinational Logic

Combinational circuit is a circuit which consist of logic gates whose outputs at any instant of time are determined directly from the present combination of inputs without regard to previous input. The combinational circuit do not use any memory.

- There will be 2^n combination of input variable for n inputs.
- A combinational circuit can have n number of inputs and m number of outputs.
- For e.g. adders, subtractors, decoders, encoders etc.

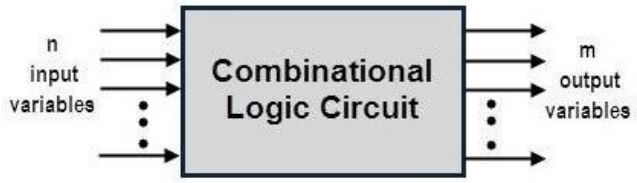


Fig: Block diagram of combinational circuit

Combinational logic circuit design procedure:

1. The problem is stated.
2. The number of available input variables and required output variables is determined.
3. The input and output variables are assigned letter symbols.
4. The truth table that defines the required relationships between inputs and outputs is derived.
5. The simplified Boolean function for each output is obtained.
6. The logic diagram is drawn.

Adders

Adders are the combinational circuits which is used to add two or more than two bits at a time.

Types of adders:

- Half Adder
- Full Adder

1. **Half Adder:**
   A combinational circuit that performs the addition of bits is called half adder. This circuit needs two binary inputs and two binary outputs. The input variables designate the augend(A) and addend(B) bits; the output variables produce the sum(S) and carry(C).

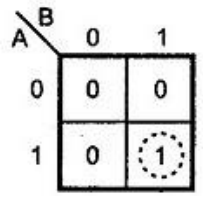
Fig: Block diagram

Truth table to identify the function of half adder:

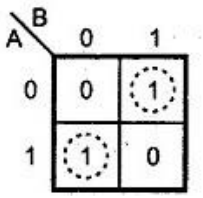
Input		Output	
A	B	Sum	Carry
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

K-map:

For Carry



For Sum

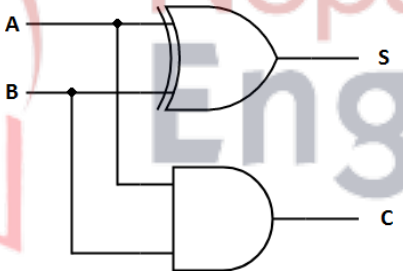


From k-map the logical expression for sum and carry is:

$C = AB$

$S = \bar{A}B + A\bar{B} = A \oplus B$

Logic diagram:



Q. Design a half adder using only NAND gates.

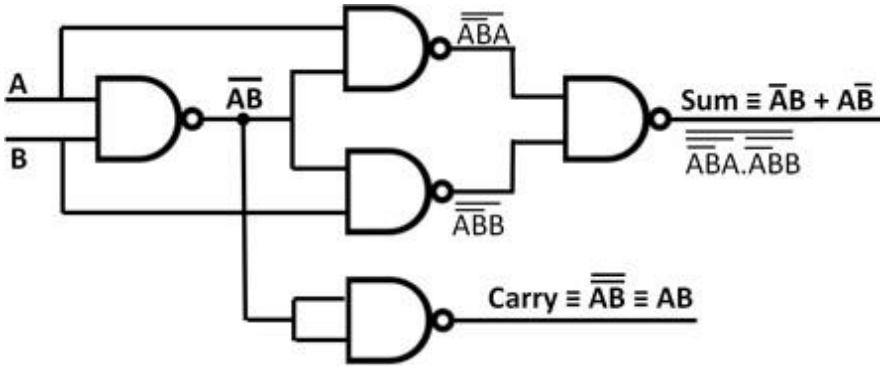
Sol^n:

Input variables: A & B, Output variables: sum(S) and carry(C)

$S = \bar{A}B + A\bar{B}$

$C = AB$

Logic diagram of half adder using NAND gates only:



Q. Design a half adder logic circuit using NOR gates only.

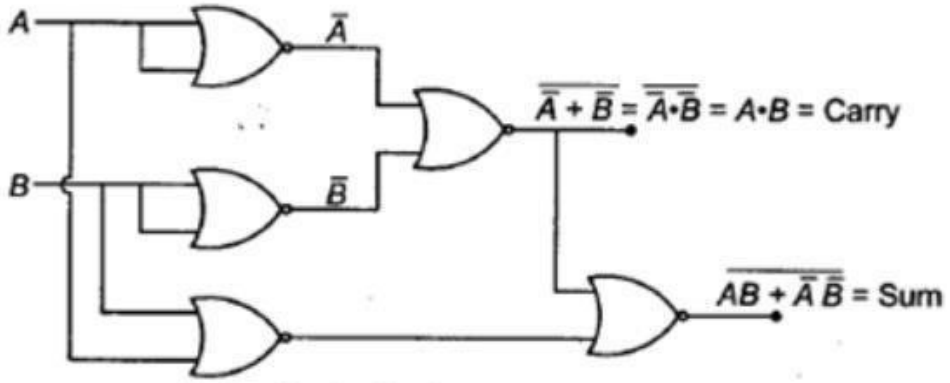
Sol^n:

Input variables: A & B, Output variables: sum(S) and carry(C)

$S = \bar{A}B + A\bar{B}$

$C = AB$

Logic diagram of half adder using NOR gates only:



Half adder logic diagram using NOR gate

2. <u>Full Adder</u>:

A combinational circuit that performs the addition of three bits at a time is called full adder. It consists of three inputs and two outputs, two inputs are the bits to be added, the third input represents the carry from the previous position.

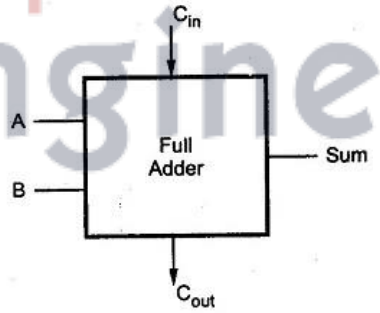


Fig: Block diagram

Truth table for full adder:

Input			Output	
A	B	Cin	Sum	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

- The sum(S) output is equal to 1 when only one input is equal to 1 or when all three inputs are equal to 1.
- The carry output (C_{out}) has a carry 1 if two or three inputs are equal to 1.

Simplified expression using k-map in SOP can be obtained as;

For Carry (C_{out})

A \ BC_{in}	00	01	11	10
0	0	0	1	0
1	0	1	1	1

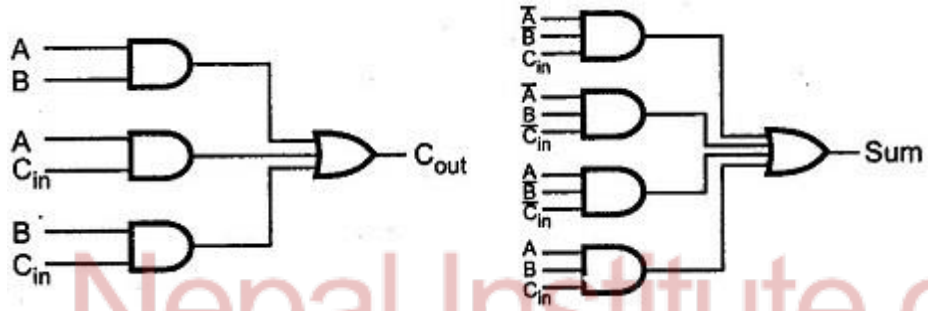
For Sum

A \ BC_{in}	00	01	11	10
0	0	1	0	1
1	1	0	1	0

$Sum(S) = \bar{A}\bar{B}C_{in} + \bar{A}B\bar{C}_{in} + A\bar{B}\bar{C}_{in} + ABC_{in}$

$C_{out} = AB + AC_{in} + BC_{in}$

Logic Diagram:

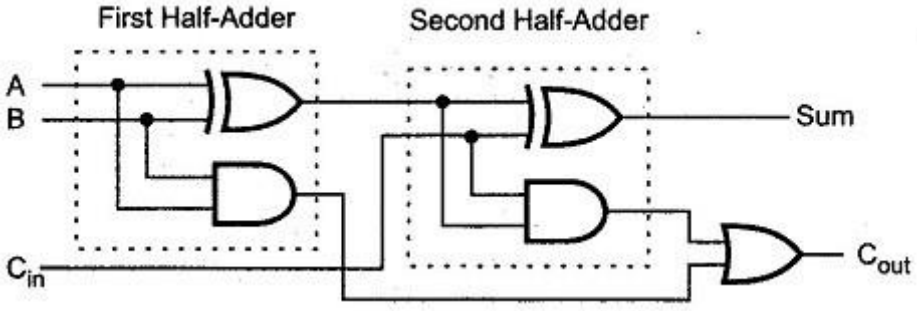


For carry For sum

Fig: SOP implementation of full-adder

Note: *It can also be implemented in POS form. (Try yourself)*

Implementation of a full-adder with two half-adders and an OR gate:



(See the derivation part in your class note)

Subtractors

Subtractor is a combinational logic circuit which is used to subtract two or more than two bits at a time, and provides difference and borrow as an output.

Types of Subtractors:

- Half subtractor
- Full subtractor

1. Half Subtractor:

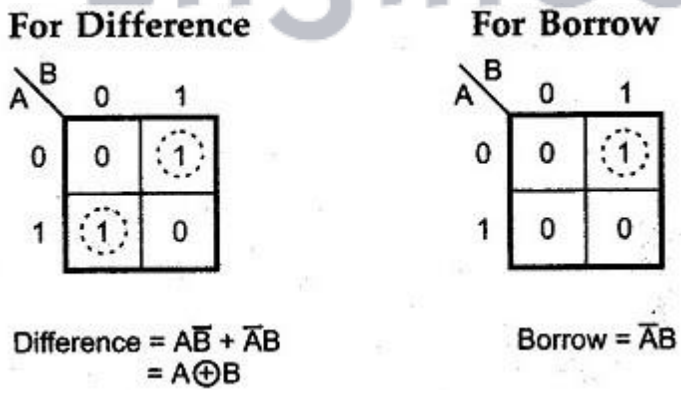
A half-subtractor is a combinational logic circuit that subtract two bits at a time and produces their difference.

It has two inputs minuend (A) & subtrahend (B) and two outputs difference and borrow. The difference is a result of subtraction and borrow is used to indicate borrow from next most significant bit. The borrow bit is present only when $A < B$.

Truth table:

Inputs		Output	
A	B	Difference	Borrow
0	0	0	0
0	1	1	1
1	0	1	0
1	1	0	0

K-map:



Logic Diagram:

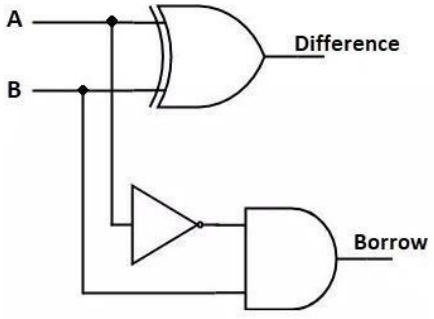


Fig: Implementation of half-subtractor

2. <u>Full Subtractor</u>:

A combinational logic circuit used to subtract three binary digits at a time is called full subtractor.

This circuit has three input and two outputs. The three inputs are A, B *and* B_{in}, denote the minuend, subtrahend and previous borrow respectively. The two outputs, D *and* B_{out} represent the difference and output borrow, respectively.

Truth table for full subtractor:

Inputs			Outputs	
A	B	B_{in}	D	B_{out}
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

Simplified expression of D and B_{out} using k-map in SOP can be obtained as;

For D

A \ BB_{in}	00	01	11	10
0	0	1	0	1
1	1	0	1	0

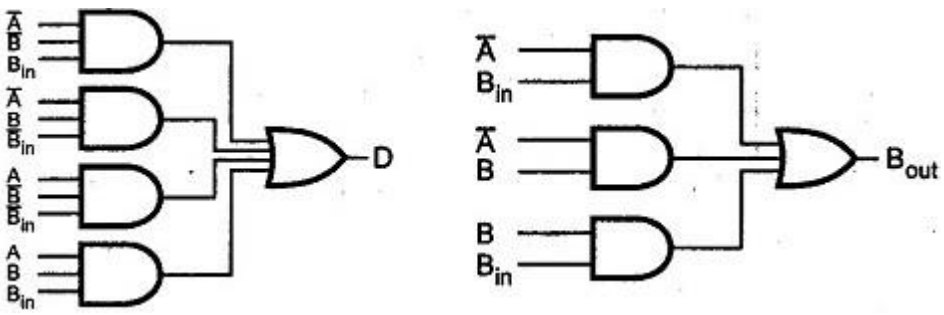
$D = \bar{A}\bar{B}B_{in} + \bar{A}B\bar{B}_{in} + A\bar{B}\bar{B}_{in} + ABB_{in}$

For B_{out}

A \ BB_{in}	00	01	11	10
0	0	1	1	1
1	0	0	1	0

$B_{out} = \bar{A}B_{in} + \bar{A}B + BB_{in}$

Logic circuit for full subtractor:

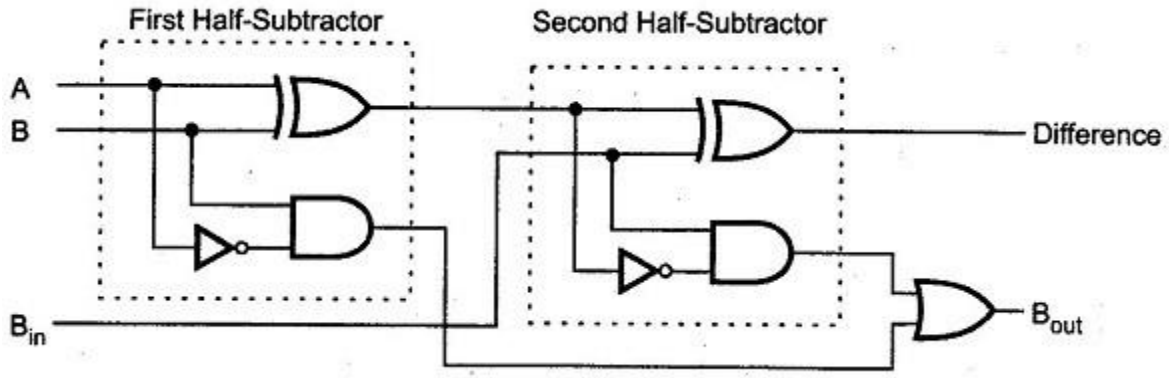


<u>For D</u> <u>For B_{out}</u>

Fig: SOP implementation of full-subtractor

Note: *It can also be implemented in POS form. (Try yourself)*

Implementation of full subtractor using two half subtractor and one OR gate:



(Similar to that of full adder)

Parity Generator and Checker

Parity Generator:

A parity generator is a combinational logic circuit that generates the parity bit in the transmitter.

- A parity bit is used for the purpose of detecting errors during transmission of binary information. It is an extra bit included with a binary message to make the number of 1's either odd or even.
- Types of parity: Even parity & Odd parity.
- In Even parity, added parity bit will make the total number of 1's an even amount.
- In Odd parity, added parity bit will make the total number of 1's an odd amount.

3-bit even parity generator truth table:

3-bit message			Even parity bit generator (P)
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

Solving the truth table for all the cases where *P* is 1 using SOP method:

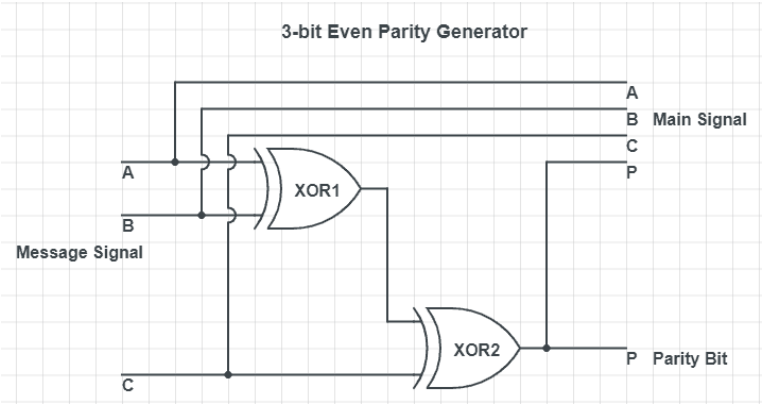
$$P = \bar{A}\,\bar{B}\,C + \bar{A}\,B\,\bar{C} + A\,\bar{B}\,\bar{C} + A\,B\,C$$

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

$$= \bar{A}\,(B \oplus C) + A\,(\overline{B \oplus C})$$

$$P = A \oplus B \oplus C$$

3-bit even parity generator circuit:



Parity checker:

A circuit that checks the parity in the receiver is called parity checker. The parity checker circuit checks for possible errors in the transmission.

- Since the information transmitted with even parity, the received must have an even number of 1's. If it has odd number of 1's, it indicates that there is an error occurred during transmission.

3-bit even parity checker truth table;

4-bit received message				Parity error check C_p
A	B	C	P	
0	0	0	0	0
0	0	0	1	1
0	0	1	0	1
0	0	1	1	0
0	1	0	0	1
0	1	0	1	0
0	1	1	0	0
0	1	1	1	1
1	0	0	0	1
1	0	0	1	0
1	0	1	0	0
1	0	1	1	1
1	1	0	0	0
1	1	0	1	1
1	1	1	0	1
1	1	1	1	0

The output of the parity checker is denoted by PEC (Parity Error Checker). If there is error, that is, if it has odd number of 1's, it will indicate 1. If no then PEC will indicate 0.

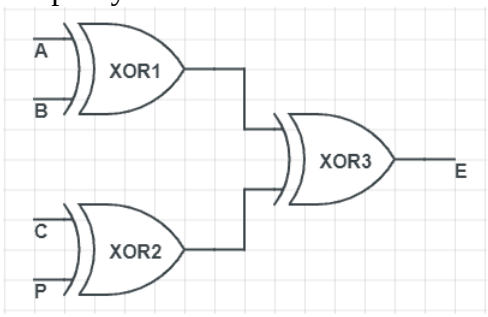
$$PEC = \bar{A}\,\bar{B}\,(\bar{C}\,D + C\,\bar{D}) + \bar{A}\,B\,(\bar{C}\,\bar{D} + C\,D) + A\,B\,(\bar{C}\,D + C\,\bar{D}) + A\,\bar{B}\,(\bar{C}\,\bar{D} + C\,D)$$

$$= \bar{A}\,\bar{B}\,(C \oplus D) + \bar{A}\,B\,(\overline{C \oplus D}) + A\,B\,(C \oplus D) + A\,\bar{B}\,(\overline{C \oplus D})$$

$$= (\bar{A}\,\bar{B} + A\,B)\,(C \oplus D) + (\bar{A}\,B + A\,\bar{B})\,(\overline{C \oplus D})$$

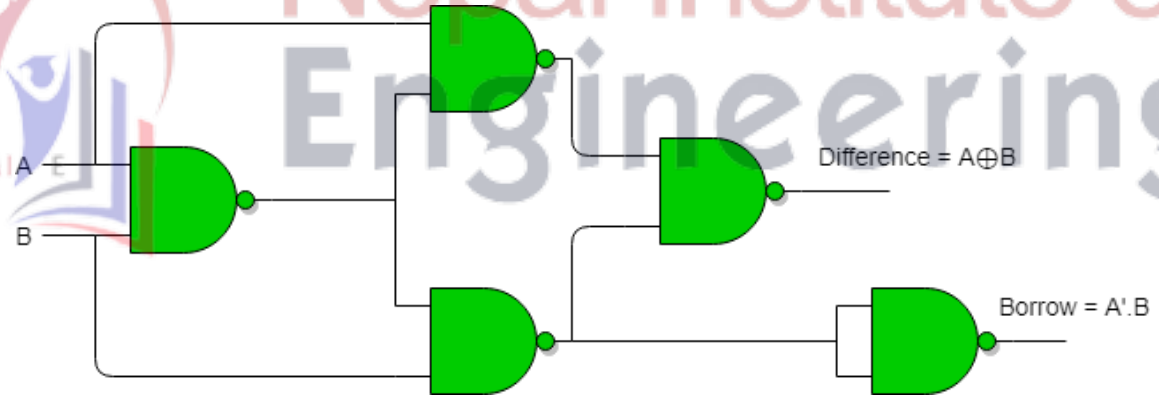
$$= (A \oplus B) \oplus (C \oplus D)$$

3-bit even parity checker circuit:

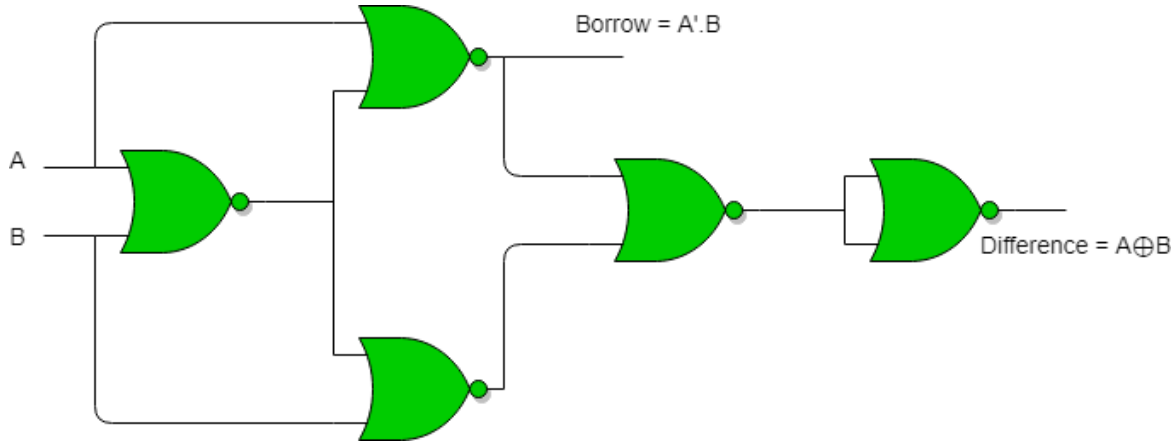


 # Implementation of half subtractor using universal gate:

Using NAND gate only:



Using NOR gate only:



Binary Adder

This circuit sums up two binary numbers A *and* B of n-bits using full-adders to add each bit pair and carry from previous bit position.

Binary Parallel Adder:

A binary parallel adder is a digital circuit that produces the arithmetic sum of two binary numbers in parallel. It consists of full adders connected in cascade, with the output carry from one full adder connected to the input carry of the next full adder. An n bit parallel adder requires n full adders.

4-bit binary parallel adder:

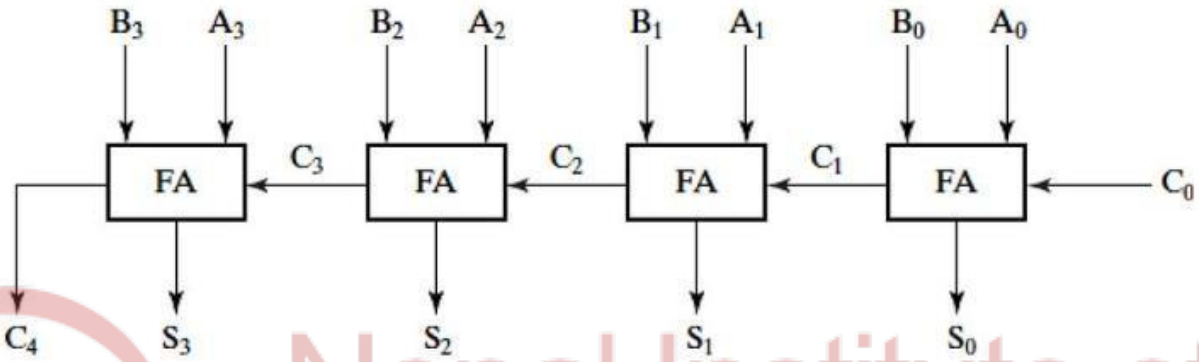
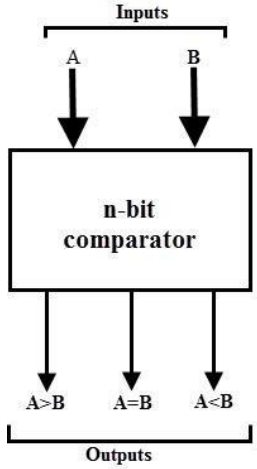


Fig: 4-bit binary parallel adder

A 4-bit binary parallel adder consists of 4-full adder. The augend bits are A_4, A_3, A_2, A_1 and addend bits are B_1, B_2, B_3, B_4. This parallel adder produces their sum as $C_4S_3S_2S_1S_0$ where C_4 is the final carry. The carries are connected in chain through the full-adders. The input carry to the first full adder is C_1 and the output carry from MSB position of full adder is C_4.

Magnitude Comparator

A magnitude comparator is a combinational circuit that compares two numbers A & B and determines their relative magnitudes. The outcome of the comparison is specified by three binary variables that indicate whether $A > B, A = B, or\ A < B$.



Note: Out of these three outputs only one output will be 1 and other two outputs will be 0 at a time.

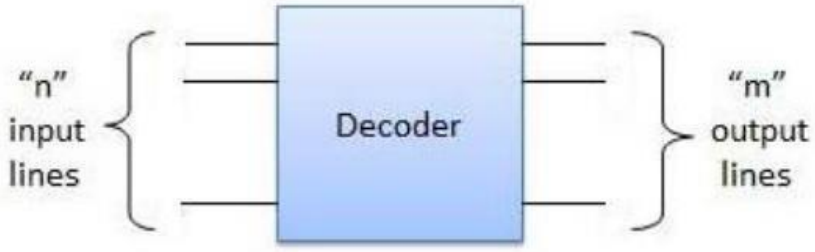
1-bit magnitude comparator: (Refer to class note)

2-bit magnitude comparator: (Refer to class note)

Decoders

A decoder is a combinational circuit that converts binary information from n input lines to a maximum of 2^n unique output lines.

- If n-bit decoded information has unused or don't care combinations, the decoder output will have less than 2^n outputs.
- The decoders presented here are called $n - to - m$ line decoders where $m \leq 2^n$. Their purpose is to generate the 2^n (or less) minterms of n input variables.



3-to-8 line decoder:

The three inputs are decoded into eight outputs, each output representing one of the minterms of the 3-input variables.
A particular application of this decoder would be a binary-to-octal conversion. The input variable may represent a binary number and the outputs will then represent the eight digits in the octal number system

Three inputs: X, Y & Z
Eight outputs: $D_0 - D_7$

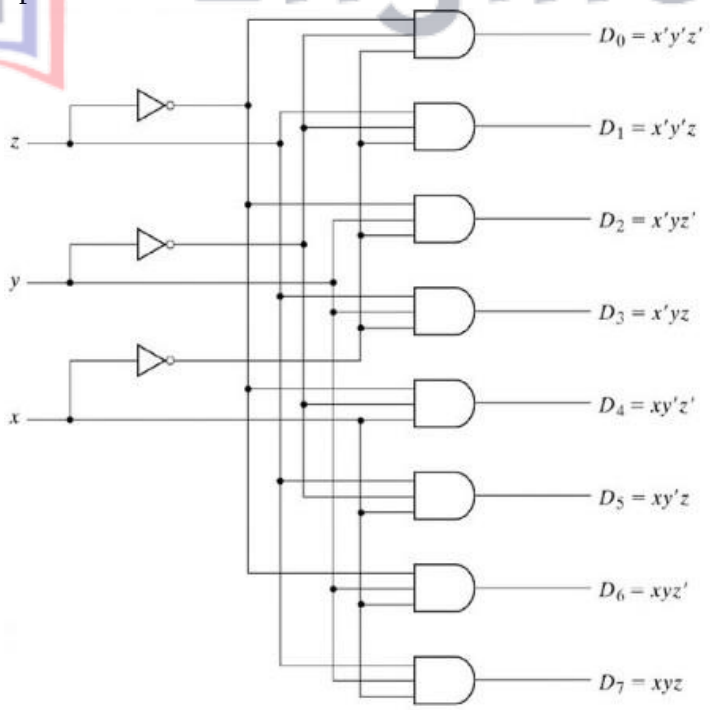


Fig: 3-to-8 line decoder

Truth table:

Inputs			Outputs							
X	**Y**	**Z**	**D_0**	**D_1**	**D_2**	**D_3**	**D_4**	**D_5**	**D_6**	**D_7**
0	0	0	**1**	0	0	0	0	0	0	0
0	0	1	0	**1**	0	0	0	0	0	0
0	1	0	0	0	**1**	0	0	0	0	0
0	1	1	0	0	0	**1**	0	0	0	0
1	0	0	0	0	0	0	**1**	0	0	0
1	0	1	0	0	0	0	0	**1**	0	0
1	1	0	0	0	0	0	0	0	**1**	0
1	1	1	0	0	0	0	0	0	0	**1**

From the truth table it is observed that the output variables are mutually exclusive because only one output can be equal to 1 at any one time. The output line whose value is equal to 1 represents the minterm equivalent of the binary number presently available in the input lines.

Encoder

An encoder is a combinational circuit that performs the inverse operation from that of decoder. It has 2^n input lines and n output lines.
The output lines generate the binary code corresponding to the input value.

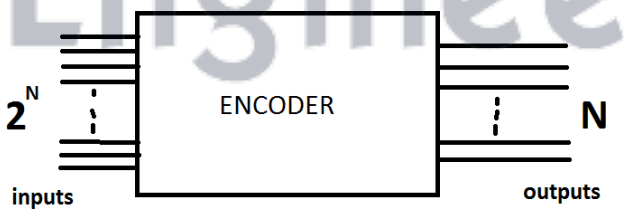


Fig: Block diagram of encoder

E.g. **Octal to binary encoder** which has 8 inputs and 3 outputs.

Truth table for octal to binary encoder:

INPUT								OUTPUT		
D_0	D_1	D_2	D_3	D_4	D_5	D_6	D_7	X	Y	Z
1	0	0	0	0	0	0	0	0	0	0
0	1	0	0	0	0	0	0	0	0	1
0	0	1	0	0	0	0	0	0	1	0
0	0	0	1	0	0	0	0	0	1	1
0	0	0	0	1	0	0	0	1	0	0
0	0	0	0	0	1	0	0	1	0	1
0	0	0	0	0	0	1	0	1	1	0
0	0	0	0	0	0	0	1	1	1	1

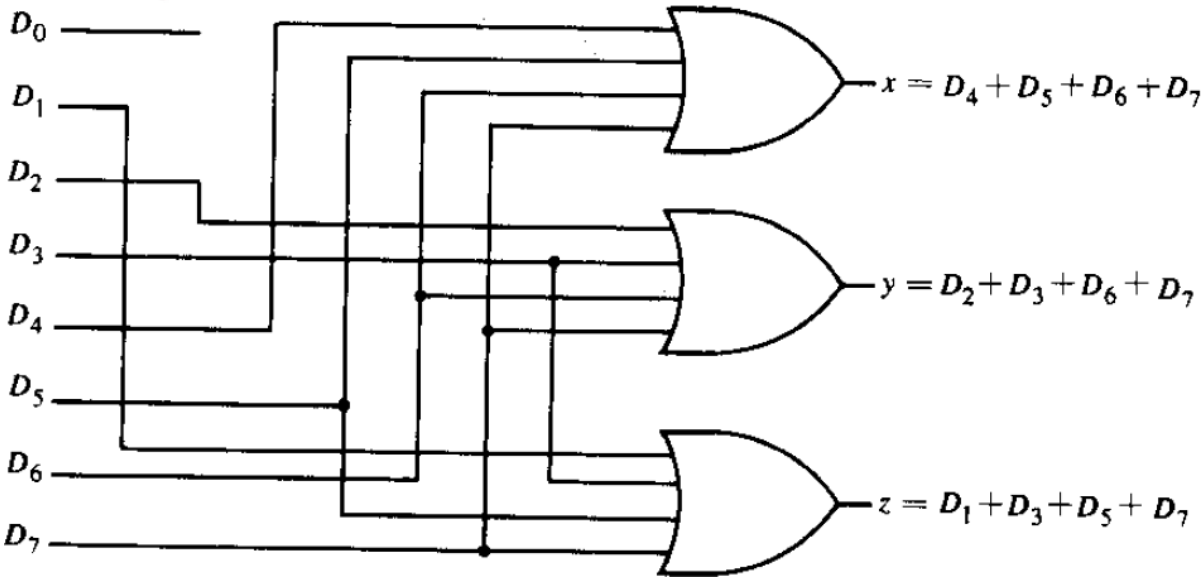
Boolean function of output variables:

$X = D_4 + D_5 + D_6 + D_7$

$X = D_2 + D_3 + D_6 + D_7$

$X = D_1 + D_3 + D_5 + D_7$

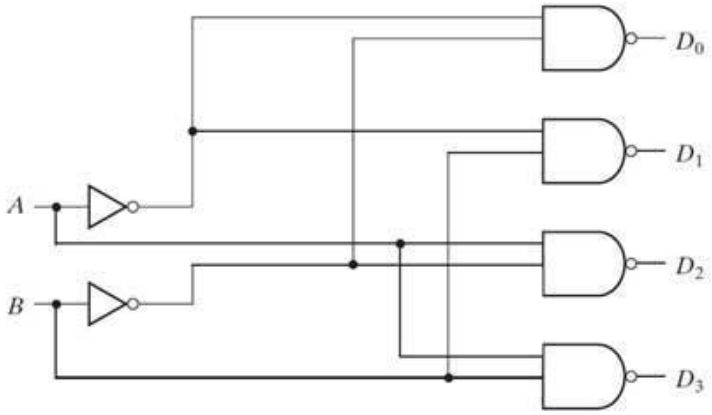
Logic circuit:



Limitation: Only one input can be enabled at a time. If two inputs are enabled at the same time, then output is undefined.

Q. Design a 2-to-4 line decoder using NAND gates.

Sol^{n}:



Truth table:

Inputs		Outputs			
A	B	D_0	D_1	D_2	D_3
0	0	0	1	1	1
0	1	1	0	1	1
1	0	1	1	0	1
1	1	1	1	1	0

For the NAND decoder only one output can be LOW and equal to logic '0' at any given time with all other outputs being HIGH at logic '1'.

Note: *Similar method for 3-to-8 line decoder in which 3-lines of input are present and 8 output lines.*

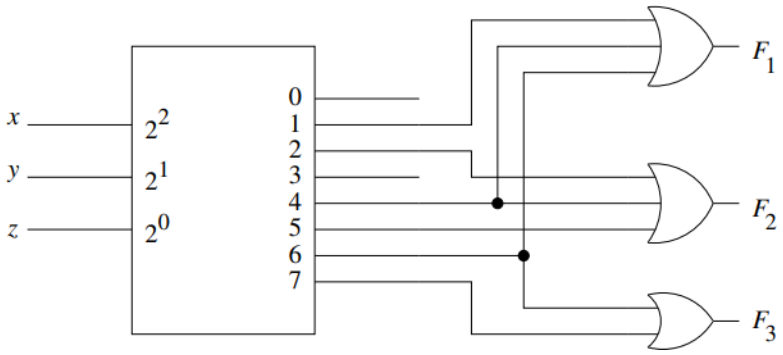
Q. Using a decoder and external gates, design the combinational circuit defined by the following three Boolean functions:

$$F_1 = x'y'z + xz'$$
$$F_2 = x'yz' + xy'$$
$$F_3 = xyz' + xy$$

Sol^n:

Truth table:

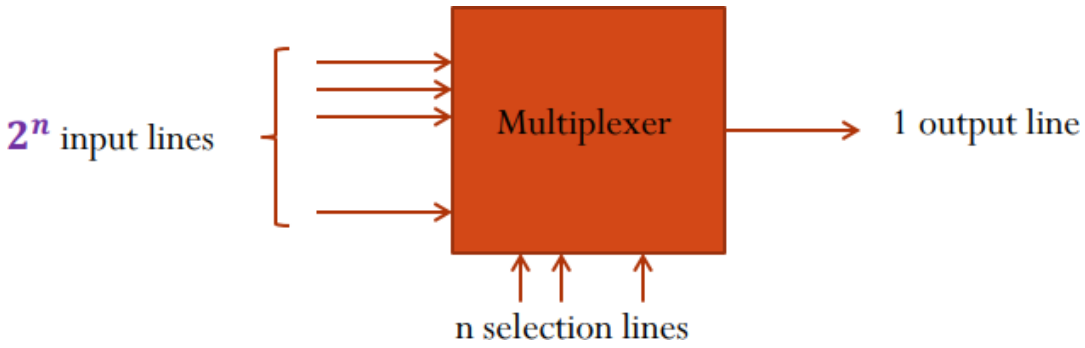
x	y	z	F_1	F_2	F_3
0	0	0	0	0	0
0	0	1	1	0	0
0	1	0	0	1	0
0	1	1	0	0	0
1	0	0	1	1	0
1	0	1	0	1	0
1	1	0	1	0	1
1	1	1	0	0	1



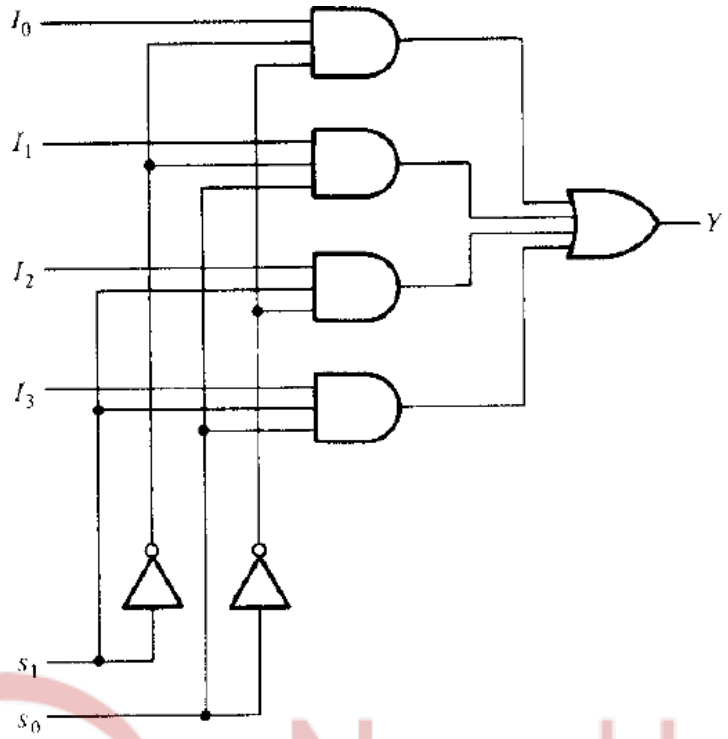
Logic Diagram

Multiplexer (MUX)

- A multiplexer is a combinational circuit that selects binary information from one of many input lines and directs it to a single output line.
- Multiplexing is the process of transmitting a large number of information over a single line.
- The selection of a particular input lines is controlled by a set of selection lines. Normally there are 2^n input lines and n selection lines whose bit combinations determine which input is selected.
- A multiplexer is also called a data selector, since it selects one of many inputs and steers the binary information to the output line.



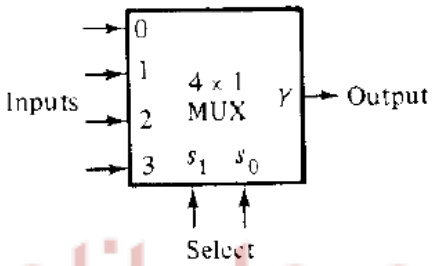
4-to-1 line Multiplexer:



(a) Logic diagram

s_1	s_0	Y
0	0	I_0
0	1	I_1
1	0	I_2
1	1	I_3

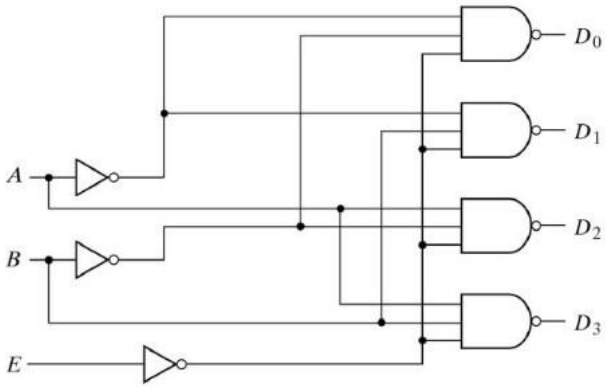
(b) Function table



(c) Block diagram

Demultiplexer (DEMUX)

- A decoder with an enable input can function as a de-multiplexer.
- A de-multiplexer is a circuit that **receives information on a single line** and transmit this information on one of 2^n **possible output lines**. The selection for particular output line is controlled by the bit values of n selection lines.



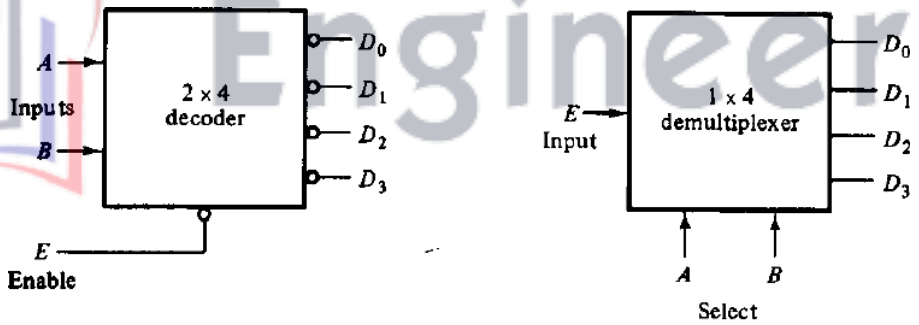
(a) Logic diagram

E	A	B	D_0	D_1	D_2	D_3
1	X	X	1	1	1	1
0	0	0	0	1	1	1
0	0	1	1	0	1	1
0	1	0	1	1	0	1
0	1	1	1	1	1	0

(b) Truth table

Fig: A 2-to-4 line decoder with enable (E) input

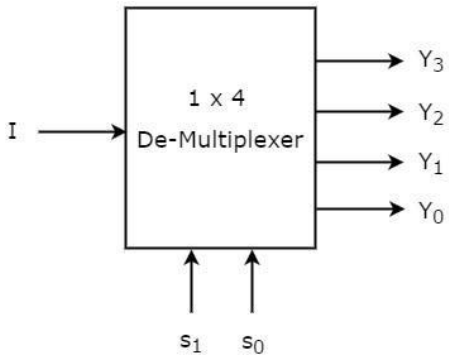
The decoder of fig can function as a de-multiplexer if the *E* line is taken as a data input line and lines A and B are taken as the selection lines.

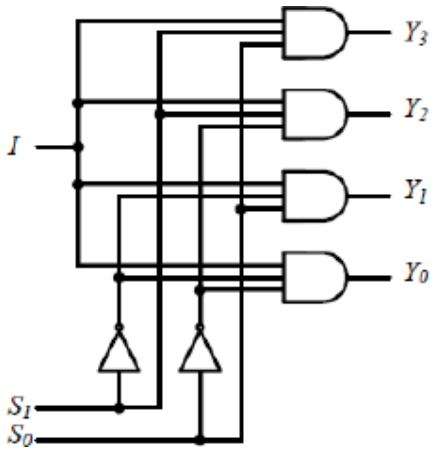


(a) Decoder with enable **(b) Demultiplexer**

1 to 4 DEMUX:

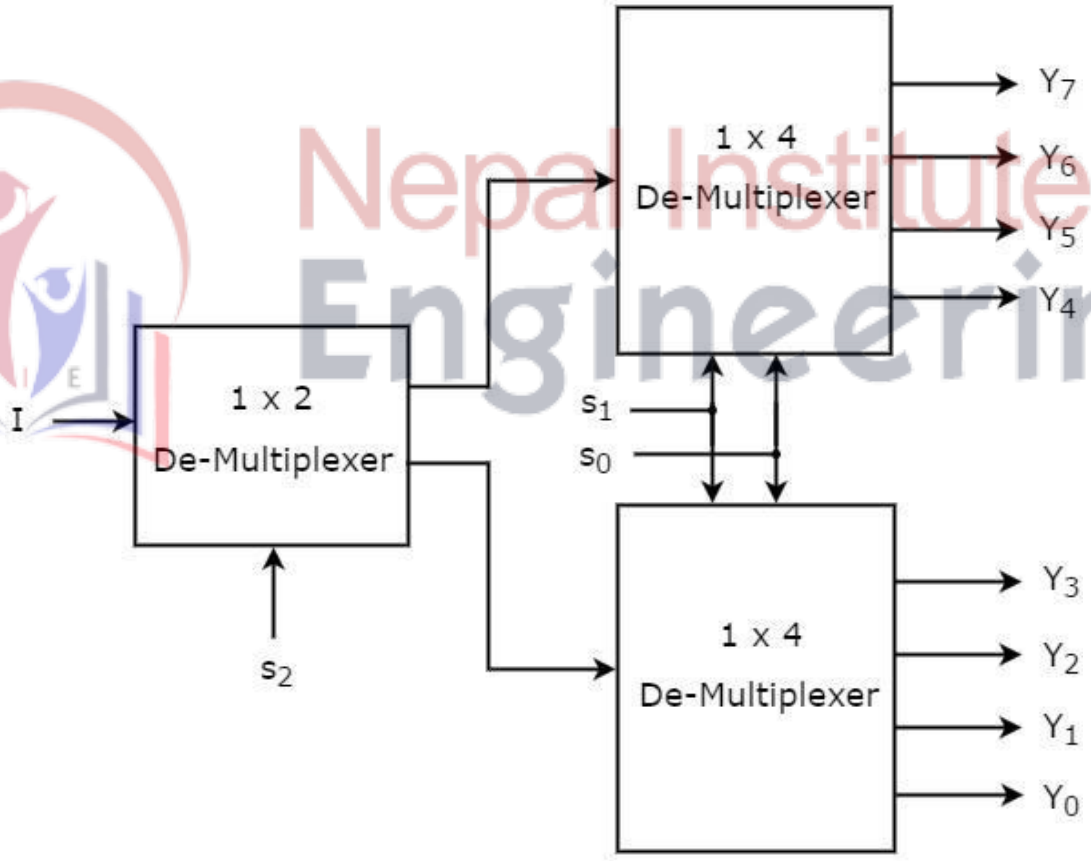
The 1:4 Demux consists of 1 data input bit, 2 control bits and 4 output bits. I is the input bit, Y_0, Y_1, Y_2, Y_3 are the four output bits and S_0 and S_1 are the control bits.





S_1	S_0	Y_3	Y_2	Y_1	Y_0
0	0	0	0	0	I
0	1	0	0	I	0
1	0	0	I	0	0
1	1	I	0	0	0

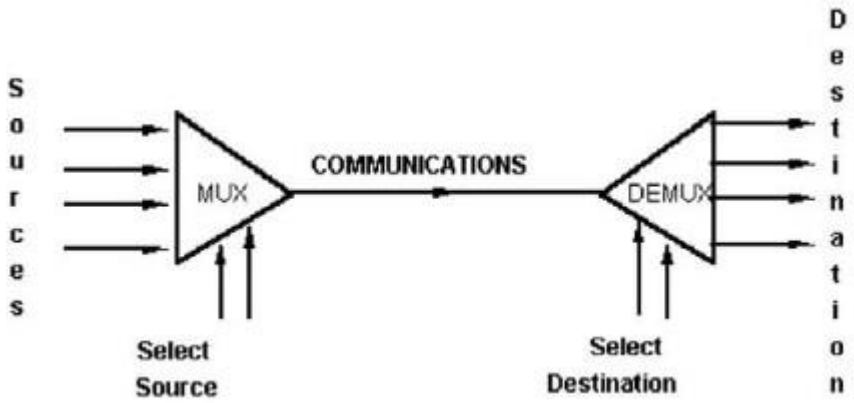
1 to 8 De-Multiplexer using 1x4 De-Multiplexers and 1x2 De-Multiplexer:



The common **selection lines, s_1 & s_0** are applied to both 1x4 De-Multiplexers. The outputs of upper 1x4 De-Multiplexer are Y_7 to Y_4 and the outputs of lower 1x4 De-Multiplexer are Y_3 to Y_0.

The other **selection line, s_2** is applied to 1x2 De-Multiplexer. If s_2 is zero, then one of the four outputs of lower 1x4 De-Multiplexer will be equal to input, I based on the values of selection lines s_1 & s_0. Similarly, if s_2 is one, then one of the four outputs of upper 1x4 De-Multiplexer will be equal to input, I based on the values of selection lines s_1 & s_0.

MUX-DEMUX Application Example



- This enables sharing a single communication line among a number of devices.
- At any time, only one source and one destination can use the communication line.