circuits

The combinational circuit is time-independent. The output it generates does not depend on any of its previous inputs. On the other hand, sequential circuits are the ones that depend on clock cycles. They depend entirely on the past as well as the present inputs for generating output.

What is a Combinational Circuit?

- The output of a Combinational Circuit depends entirely on the present input.
- It exhibits a faster speed.
- It is comparatively easier to design.
- No feedback is present between the input and output.
- The combinational circuit depends on time.
- Logic gates form the building blocks of such circuits.
- One can make use of it for both Boolean and arithmetic operations.
- They don’t hold the capacity of storing any state.
- These circuits do not have a clock- thus, they don’t require triggering.
- They do not possess any memory element.
- Users can feasibly use as well as handle them.
- Example –Decoder, Encoder, Multiplexer, DE multiplexer, half-adder, full-adder etc.

What is a Sequential Circuit?

- The output of a Sequential Circuit depends on both- past as well as present inputs.
- It works at a comparatively slower speed.
- The design of these circuits is comparatively much tougher than the Combinational Circuit.

- A feedback path exists between the output and the input.
- The circuit is time-dependent.
- Flip-flops constitute the building blocks of such a circuit.
- People mainly use them for storing data and information.
- They possess the capability of storing any data state or retaining an earlier state at any given point.
- Because a Sequential circuit depends on a clock, it usually requires triggering.
- They always possess a memory element.
- A user may not be able to handle and use these circuits easily.

For Example:

Flip-flops like SR flip-flop, JK flip-flop, T flip-flop, registers, counters, etc.

Difference Between Combinational and Sequential Circuit

Parameters	**Combinational Circuit**	**Sequential Circuit**
Meaning and Definition	It is a type of circuit that generates an output by relying on the input it receives at that instant, and it stays independent of time.	It is a type of circuit in which the output does not only rely on the current input. It also relies on the previous ones.
Feedback	A Combinational Circuit requires no feedback for generating the next output. It is because its output has no dependency on the time instance.	The output of a Sequential Circuit, on the other hand, relies on both- the previous feedback and the current input. So, the output generated from the previous inputs gets transferred in the form of feedback. The

		circuit uses it (along with inputs) for generating the next output.
Performance	We require the input of only the current state for a Combinational Circuit. Thus, it performs much faster and better in comparison with the Sequential Circuit.	In the case of a Sequential Circuit, the performance is very slow and also comparatively lower. Its dependency on the previous inputs makes the process much more complex.
Complexity	It is very less complex in comparison. It is because it basically lacks implementation of feedback.	This type of circuit is always more complex in its nature and functionality. It is because it implements the feedback, depends on previous inputs and also on clocks.
Elementary Blocks	Logic gates form the building/ elementary blocks of a Combinational Circuit.	Flip-flops form the building/ elementary blocks of a Sequential Circuit.
Operation	One can use these types of circuits for both- Boolean as well as Arithmetic operations.	You can mainly make use of these types of circuits for storing data.

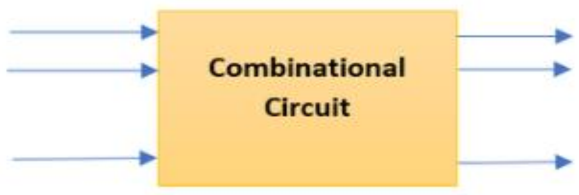
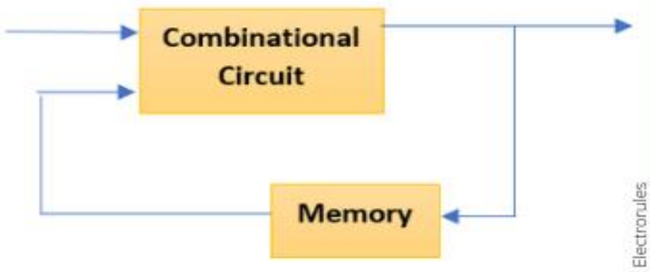
Combinational Circuit	Sequential Circuit
Output only depends on the present input	Output depends on present input and past output
Memory element is absent	Memory element is present
No clock signal is applied	Clock signal is required
	
Example - Half Adder, Full Adder, Multiplexer	Examples - Flipflop, Counters, Registers

Image Credits: Electrorules

What is an Adder?

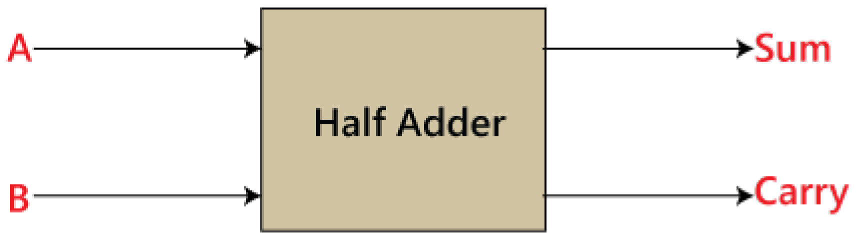
An adder is a digital logic circuit in electronics that is extensively used for the addition of numbers. In many computers and other types of processors, adders are even used to calculate addresses and related activities and calculate table indices in the ALU and even utilized in other parts of the processors. These can be built for many numerical representations like excess-3 or binary coded decimal. Adders are basically classified into two types: Half Adder and Full Adder.

Half Adder (HA):

Half adder is the simplest of all adder circuits. Half adder is a combinational arithmetic circuit that adds two numbers and produces a sum bit (s) and carry bit (c) both as output. The addition of 2 bits is done using a combination circuit called a Half adder. The input variables are augend and addend bits and output variables are sum & carry bits. A and B are the two input bits.

let us consider two input bits A and B, then sum bit (s) is the X-OR of A and B. it is evident from the function of a half adder that it requires one X-OR gate and one AND gate for its construction.

Block diagram

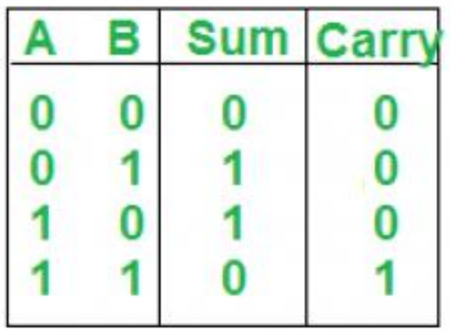


In the above table,

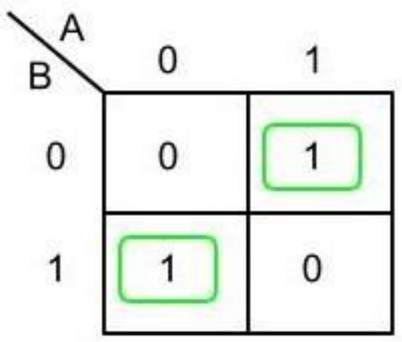
1. 'A' and' B' are the input states, and 'sum' and 'carry' are the output states.
2. The carry output is 0 in case where both the inputs are not 1.
3. The least significant bit of the sum is defined by the 'sum' bit.

The SOP form of the sum and carry are as follows:

Sum = x'y+xy'
Carry = xy

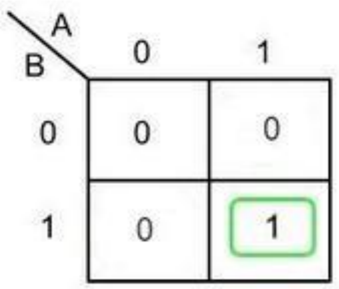


A	B	Sum	Carry
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1



B \ A	0	1
0	0	1
1	1	0

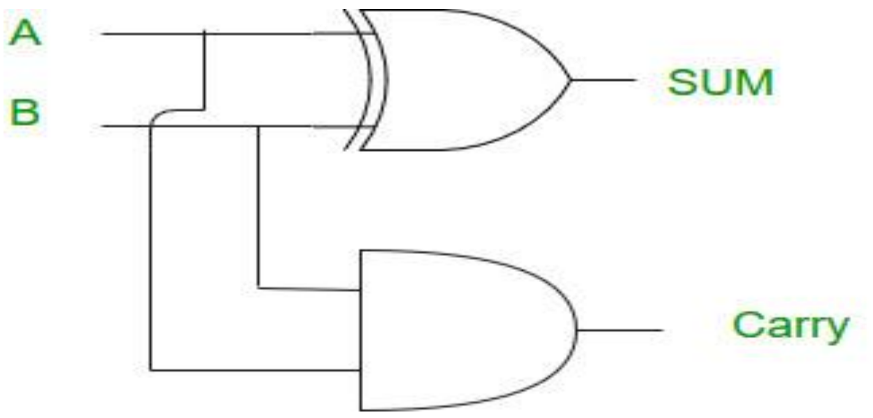
Sum = A XOR B



B \ A	0	1
0	0	0
1	0	1

Carry = A AND B

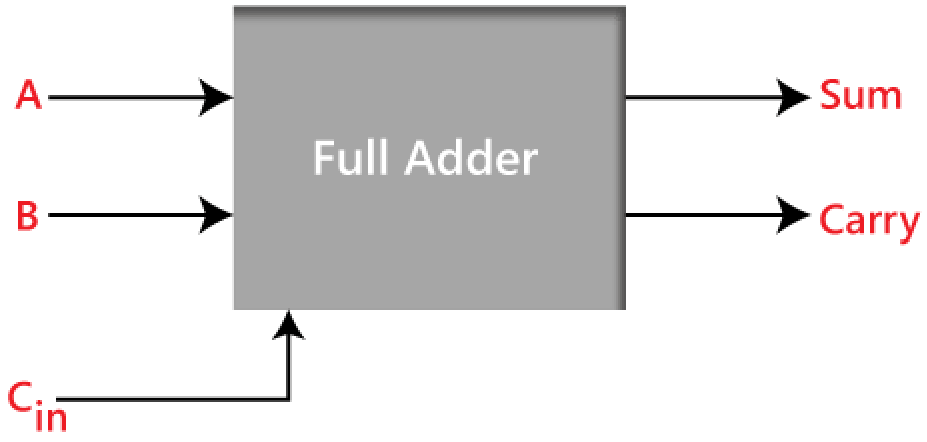
Implementation:



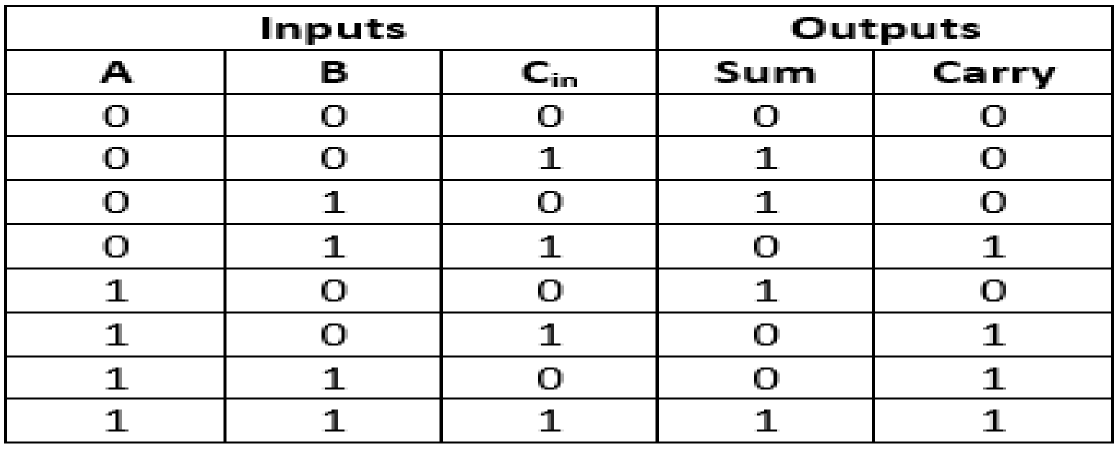
Full Adder

The half adder is used to add only two numbers. To overcome this problem, the full adder was developed. The full adder is used to add three 1-bit binary numbers A, B, and carry C. The full adder has three input states and two output states i.e., sum and carry.

Block diagram



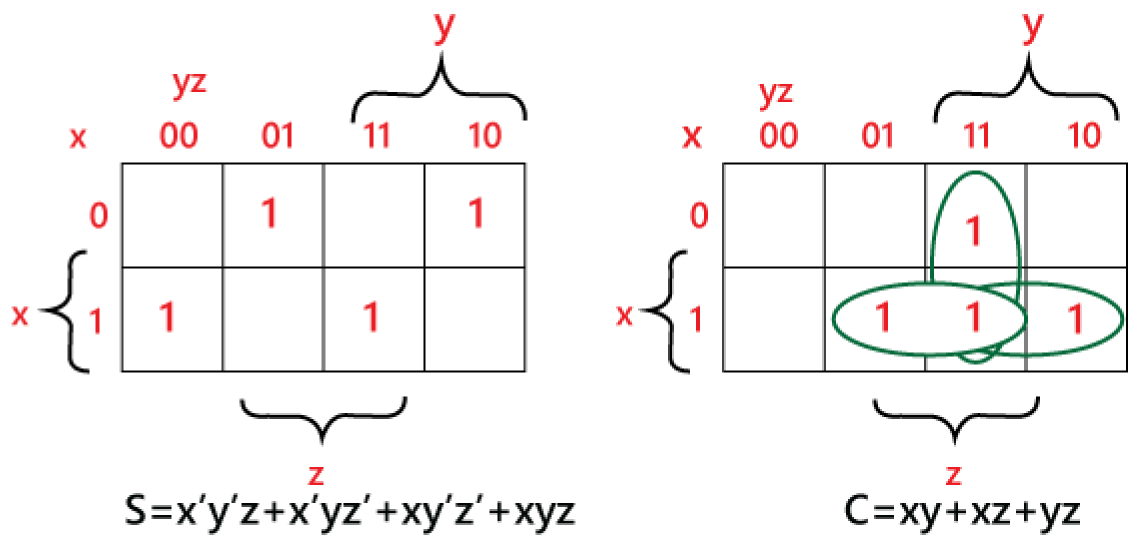
Truth Table



Inputs			Outputs	
A	B	C_{in}	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

In the above table,

1. 'A' and' B' are the input variables. These variables represent the two significant bits which are going to be added
2. 'C_{in}' is the third input which represents the carry. From the previous lower significant position, the carry bit is fetched.
3. The 'Sum' and 'Carry' are the output variables that define the output values.
4. The eight rows under the input variable designate all possible combinations of 0 and 1 that can occur in these variable
5. The SOP form can be obtained with the help of K-map as:



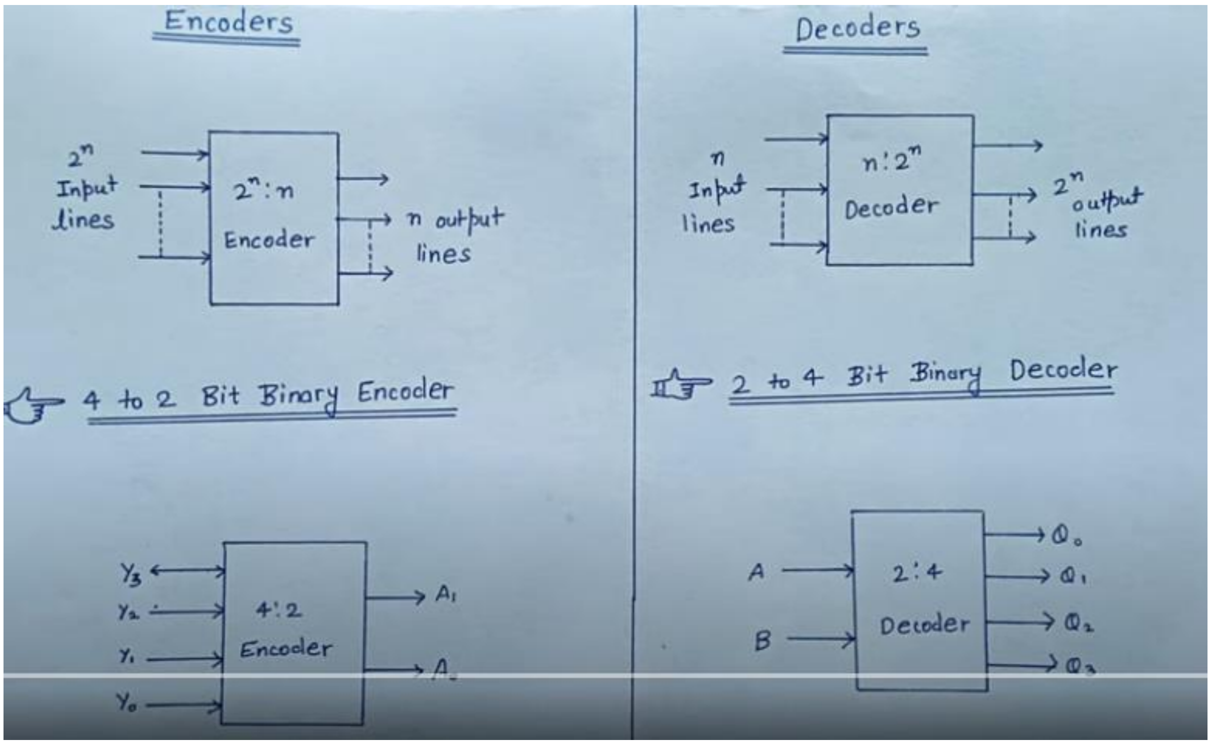
6.
7. Sum = x' y' z+x' yz+xy' z'+xyz
   Carry = xy+xz+yz

Sum:

- Perform the XOR operation of input A and B.
- Perform the XOR operation of the outcome with carry. So, the sum is (A XOR B) XOR C_{in} which is also represented as:
  $(A \oplus B) \oplus C_{in}$

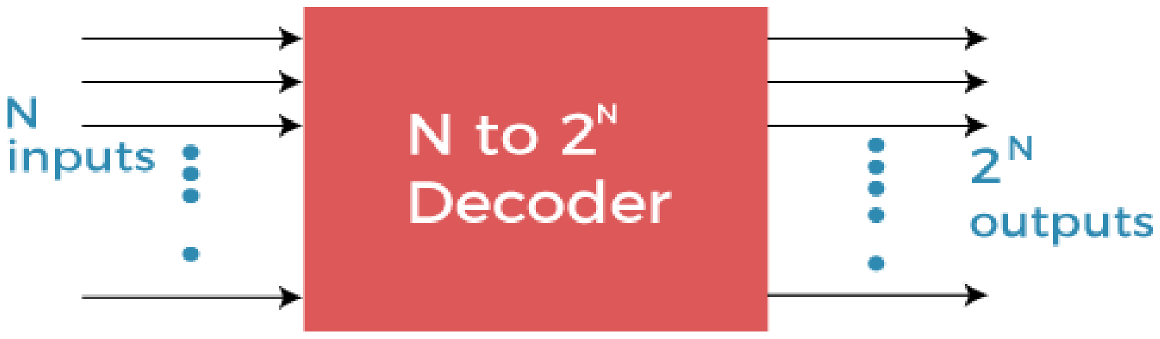
Carry:

1. Perform the 'AND' operation of input A and B.
2. Perform the 'XOR' operation of input A and B.
3. Perform the 'OR' operations of both the outputs that come from the previous two steps. So the 'Carry' can be represented as:
   $A.B + (A \oplus B)$



Decoders

The combinational circuit that change the binary information into 2^N output lines is known as **Decoders.** The binary information is passed in the form of N input lines. The output lines define the 2^N-bit code for the binary information. In simple words, the **Decoder** performs the reverse operation of the **Encoder**. At a time, only one input line is activated for simplicity. The produced 2^N-bit output code is equivalent to the binary information.

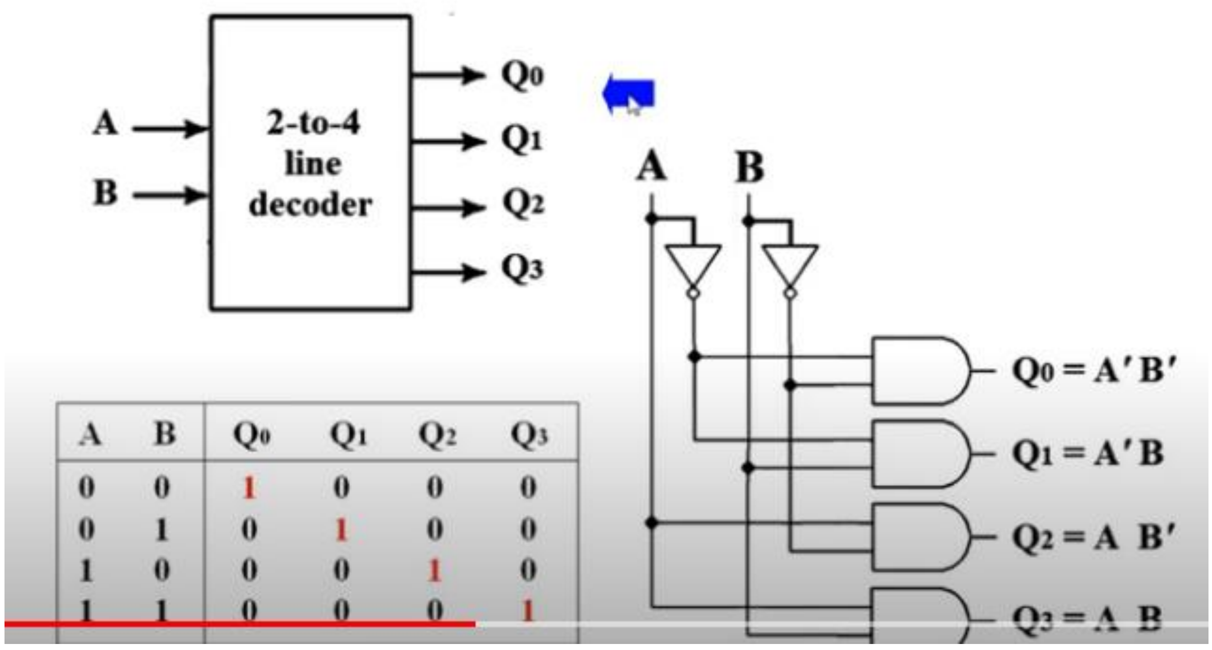


2 to 4-line decoder:

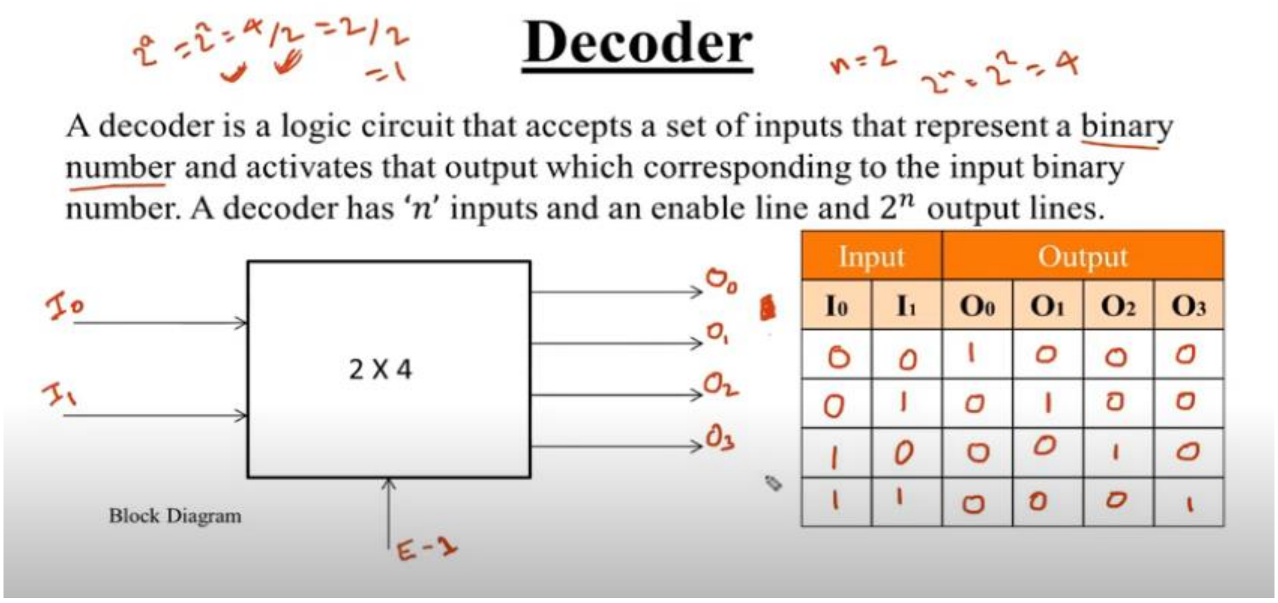
In the **2 to 4-line decoder**, there are a total of three inputs, i.e., **A0, A1,** and **E** and four outputs, i.e., **Y0, Y1, Y2,** and **Y3**.

In the 2 to 4-line decoder, there is a total of three inputs, i.e., A_0, and A_1 and E and four outputs, i.e., Y_0, Y_1, Y_2, and Y_3. For each combination of inputs, when the enable 'E' is set to 1, one of these four outputs will be 1. The block diagram and the truth table of the 2 to 4-line decoder are given below.

Design the **decoder** shown below:



A	B	Q_0	Q_1	Q_2	Q_3
0	0	1	0	0	0
0	1	0	1	0	0
1	0	0	0	1	0
1	1	0	0	0	1



$2^{a} = 2^{n} = 4/2 = 2/2$ $= 1$

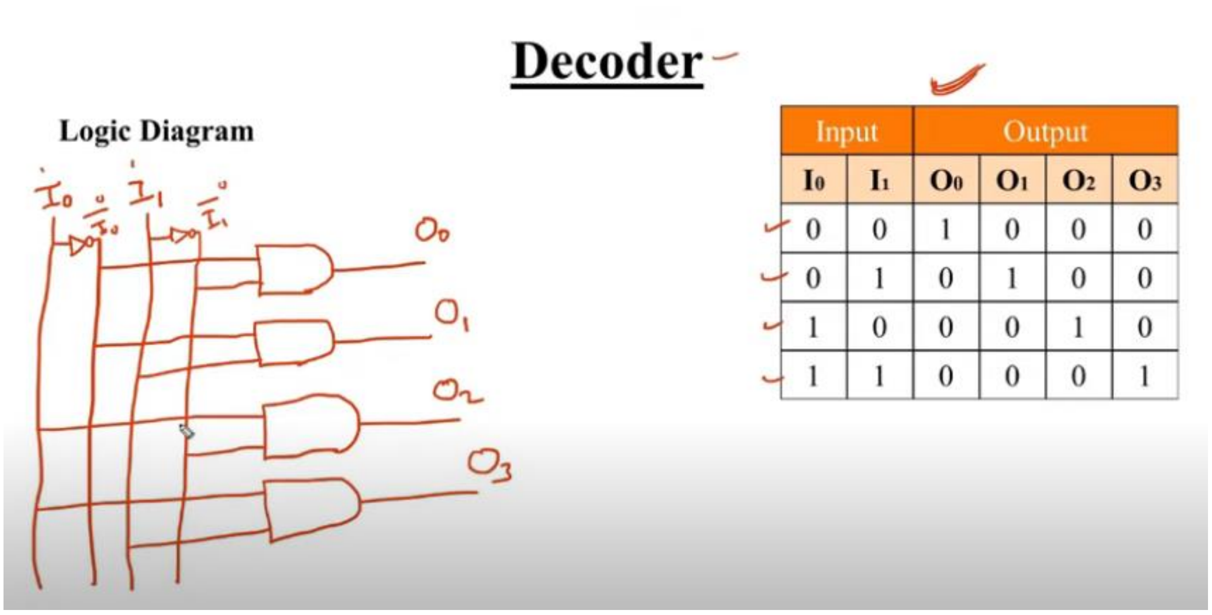
Decoder

$n=2$ $2^n = 2^2 = 4$

A decoder is a logic circuit that accepts a set of inputs that represent a binary number and activates that output which corresponding to the input binary number. A decoder has 'n' inputs and an enable line and 2^n output lines.

Block Diagram

Input		Output			
I_0	I_1	O_0	O_1	O_2	O_3
0	0	1	0	0	0
0	1	0	1	0	0
1	0	0	0	1	0
1	1	0	0	0	1

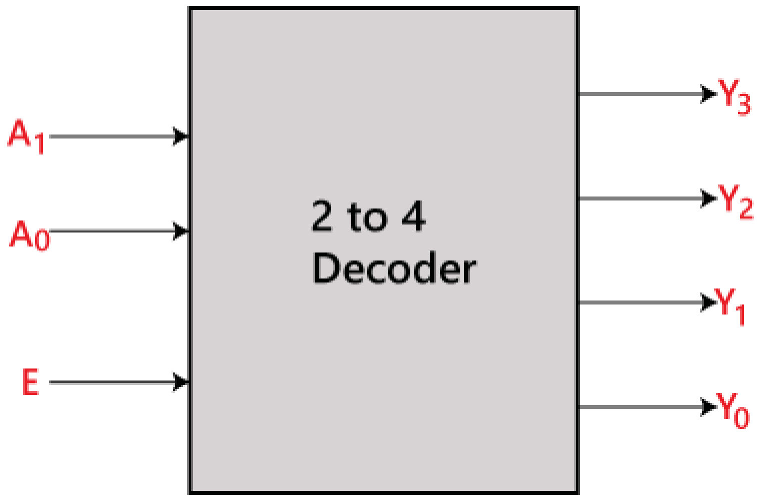


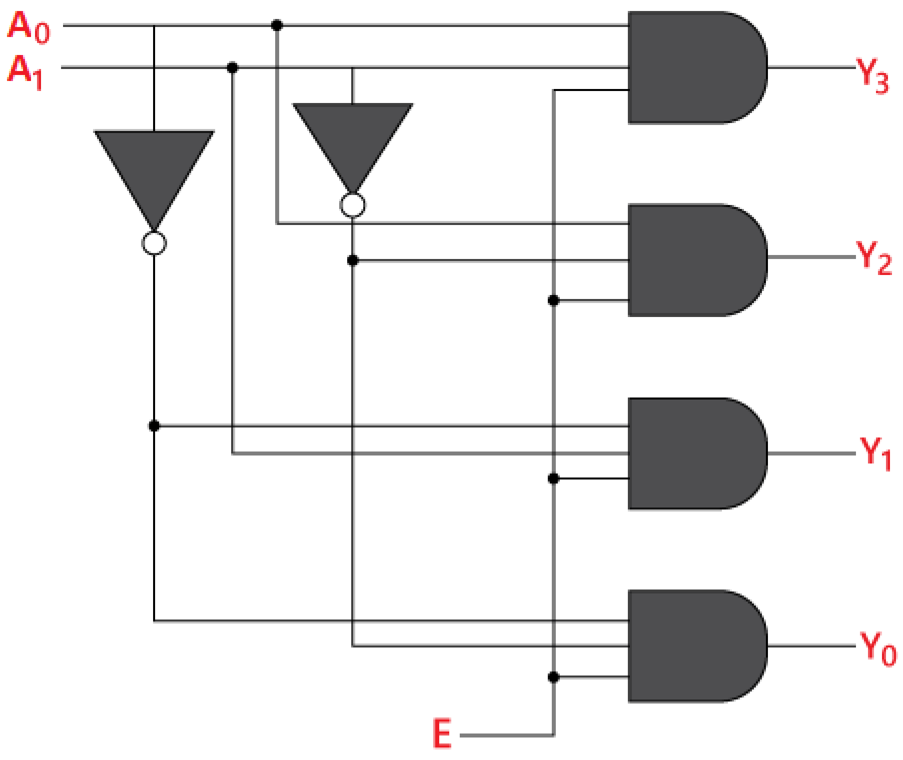
Decoder

Logic Diagram

Input		Output			
I_0	I_1	O_0	O_1	O_2	O_3
0	0	1	0	0	0
0	1	0	1	0	0
1	0	0	0	1	0
1	1	0	0	0	1

Block Diagram:





3 to 8-line decoder:

In **3 to 8-line decoder**, there are a total of eight outputs, i.e., **Y0, Y1, Y2, Y3, Y4, Y5, Y6,** and **Y7** and three inputs, i.e., **A0, A1,** and **A2**. It is also called as Binary to octal decoder.

The 3 to 8-line decoder is also known as **Binary to Octal Decoder**. In a 3 to 8-line decoder, there is a total of eight outputs, i.e., Y_0, Y_1, Y_2, Y_3, Y_4, Y_5, Y_6, and Y_7 and three outputs, i.e., A_0, A1, and A_2. This circuit has an enable input 'E'. Just like 2 to 4-line decoder, when enable 'E' is set to 1, one of these four outputs will be 1. The block diagram and the truth table of the 3 to 8-line encoder are given below

The commonly used or most preferred decoders are **n to m decoders,** where m<= 2^n. It is also referred to as **n * m decoder** that has n inputs and m outputs.

Truth Table –

X	Y	Z	D0	D1	D2	D3	D4	D5	D6	D7
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

Implementation –
D0 is high when X = 0, Y = 0 and Z = 0. Hence,
D0 = X’ Y’ Z’

Similarly,

D1 = X’ Y’ Z

D2 = X’ Y Z’

D3 = X’ Y Z

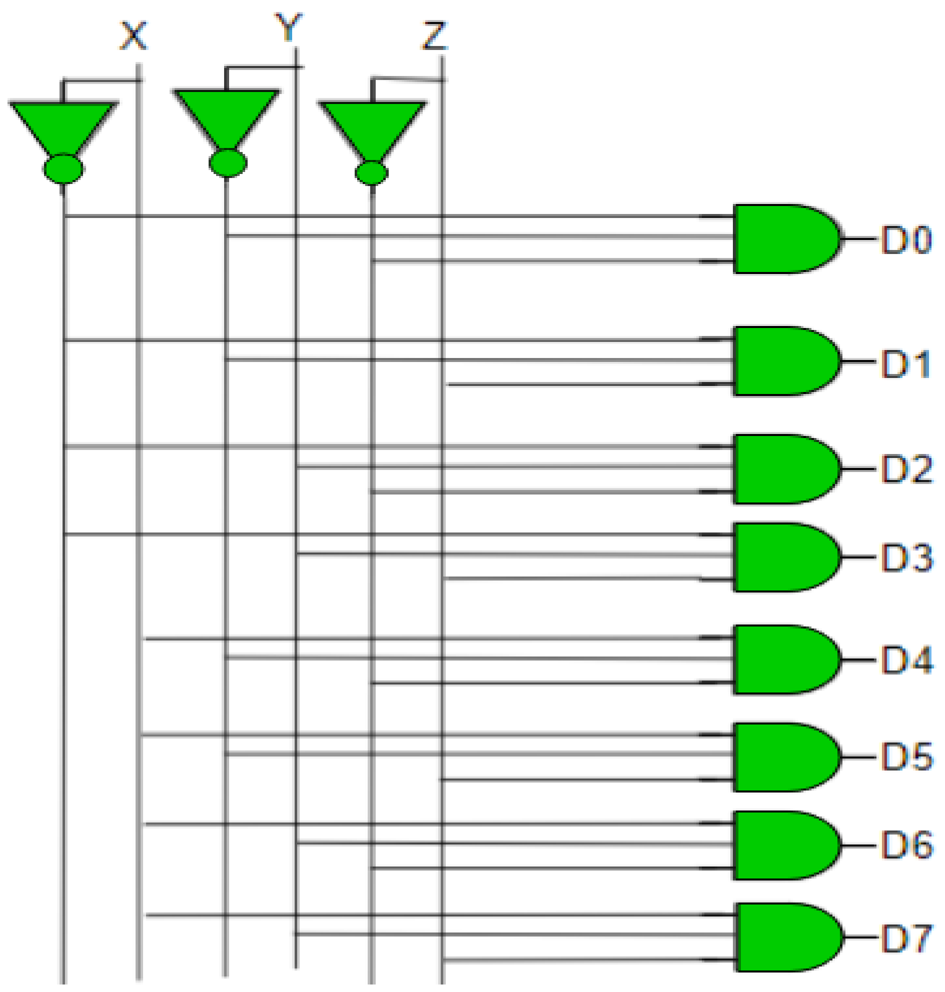
D4 = X Y’ Z’

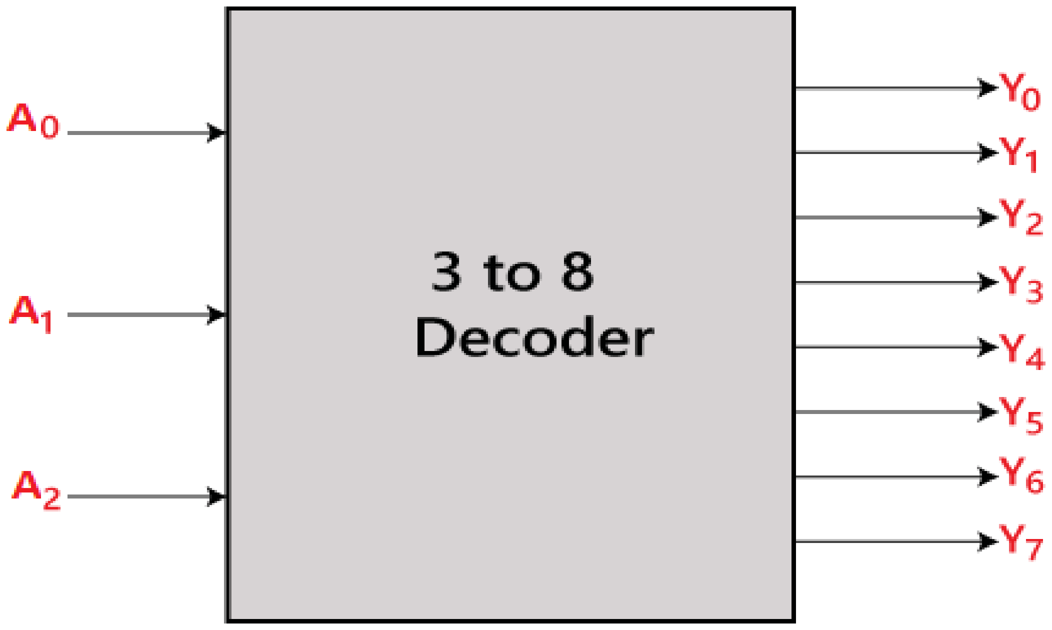
D5 = X Y’ Z

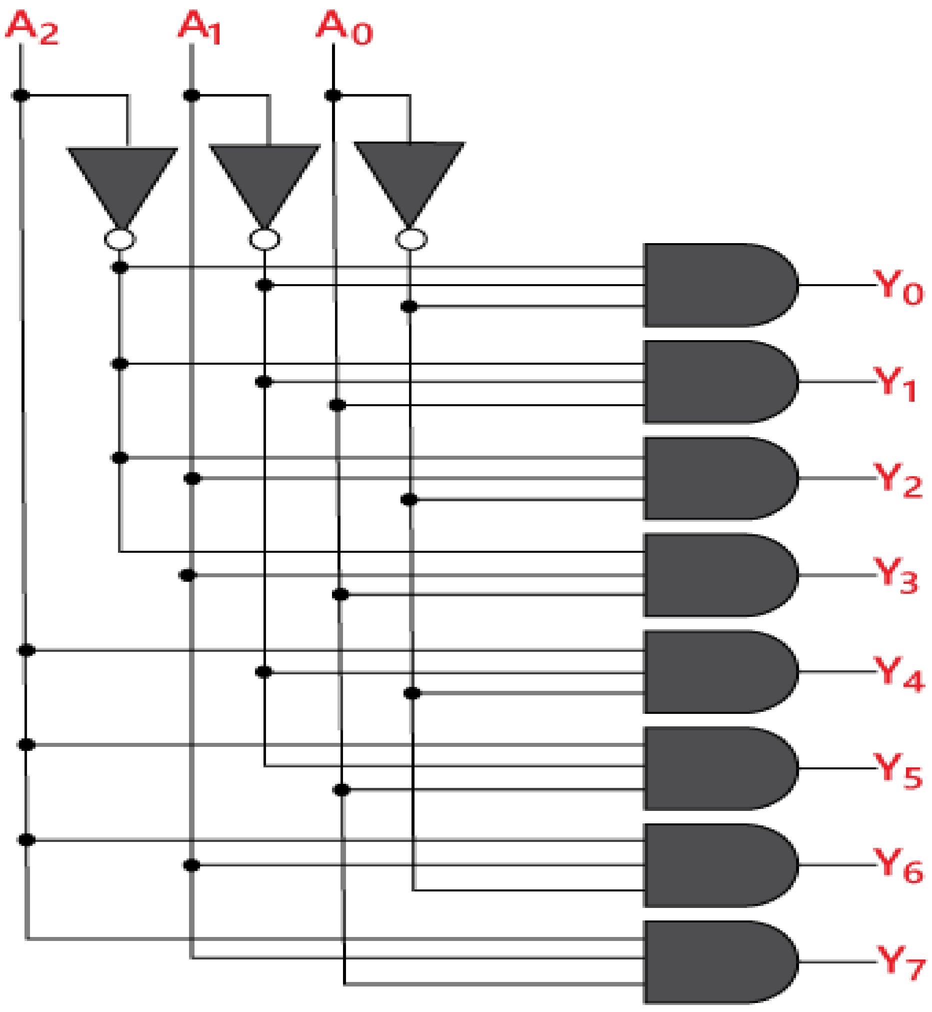
D6 = X Y Z’

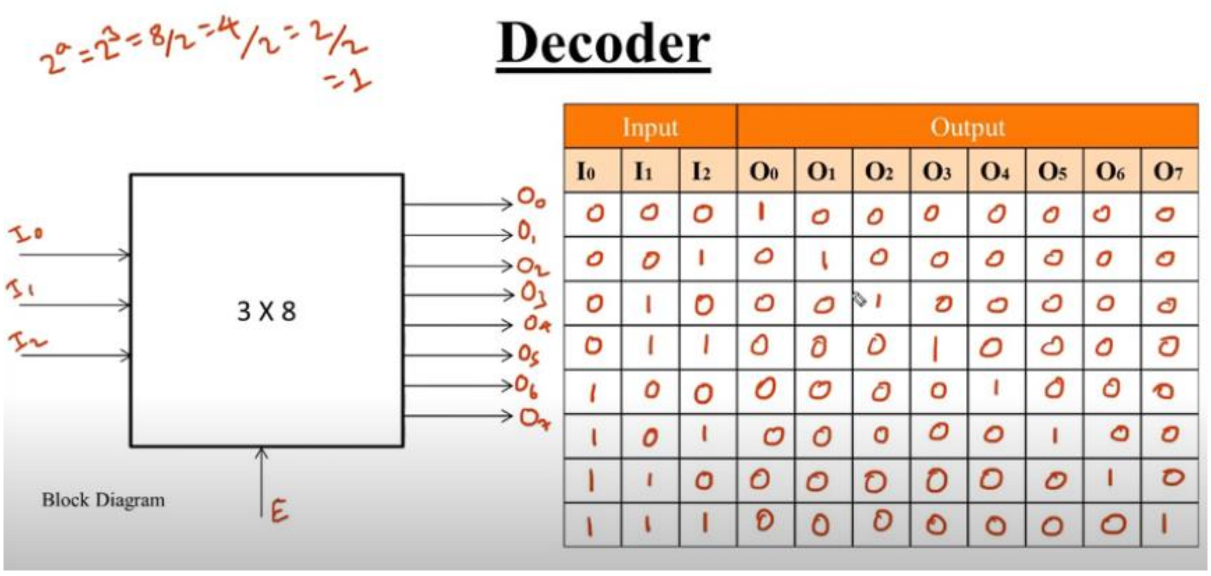
D7 = X Y Z

Hence,


X
Y
Z
D0
D1
D2
D3
D4
D5
D6
D7


A_0
A_1
A_2
3 to 8 Decoder
Y_0
Y_1
Y_2
Y_3
Y_4
Y_5
Y_6
Y_7





$2^n = 2^3 = 8/2 = 4/2 = 2/2 = 1$

Decoder

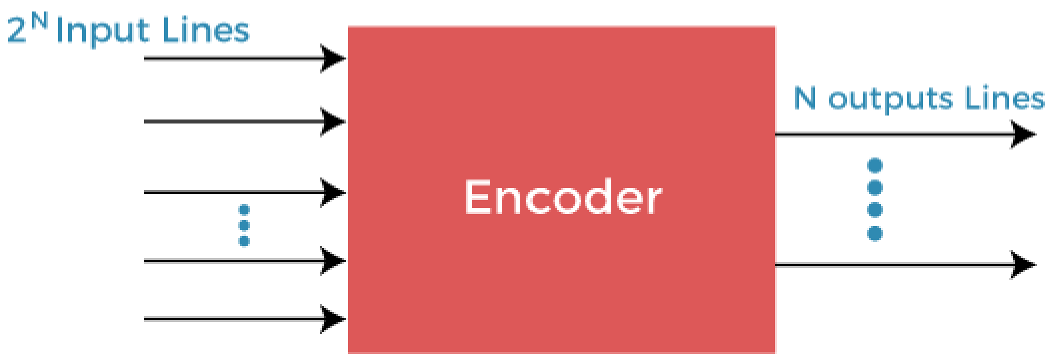
Input			Output							
I_0	I_1	I_2	O_0	O_1	O_2	O_3	O_4	O_5	O_6	O_7
0	0	0	1	0	0	0	0	0	0	0
0	0	1	0	1	0	0	0	0	0	0
0	1	0	0	0	1	0	0	0	0	0
0	1	1	0	0	0	1	0	0	0	0
1	0	0	0	0	0	0	1	0	0	0
1	0	1	0	0	0	0	0	1	0	0
1	1	0	0	0	0	0	0	0	1	0
1	1	1	0	0	0	0	0	0	0	1

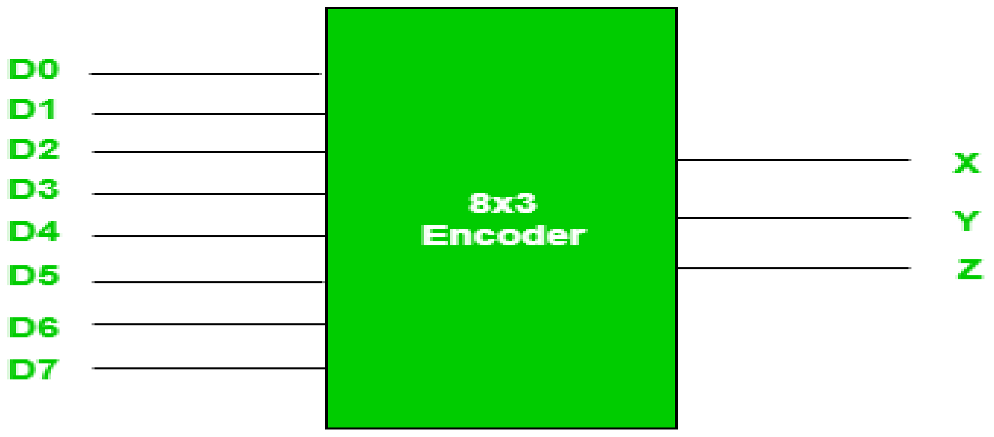
Encoder

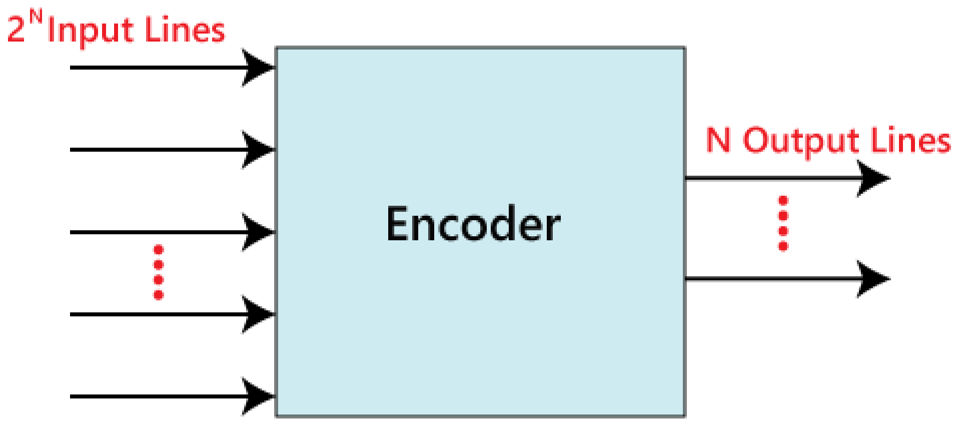
The combinational circuits that change the binary information into N output lines are known as **Encoders**. The binary information is passed in the form of 2^N input lines. The output lines define the N-bit code for the binary information. In simple words, the **Encoder** performs the reverse operation of the **Decoder**. At a time, only one input line is activated for simplicity. The produced N-bit output code is equivalent to the binary information.

An encoder can also be described as a combinational circuit that has a maximum of 2^n (or less) input lines and n output lines.

Encoders are used to convert a decimal number into a binary number. The objective is to perform a binary operation such as addition, subtraction, multiplication, etc. The basic logic element that is used in encoders is **OR gate.** The operation of the encoder is fairly simple. Encoders are commonly used in Videos, E-mail, etc.





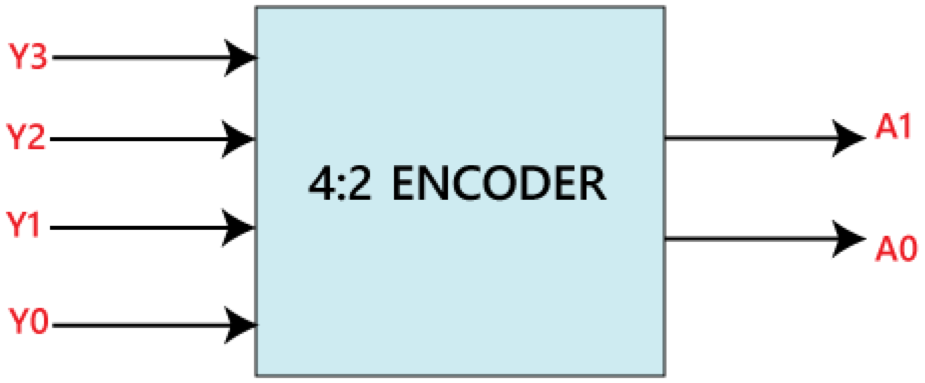


In **4 to 2 line encoder**, there are a total of four inputs, i.e., **Y0, Y1, Y2,** and **Y3**, and two outputs, i.e., **A0** and **A1**.

4 to 2-line Encoder:

In 4 to 2-line encoder, there are total of four inputs, i.e., Y_0, Y_1, Y_2, and Y_3, and two outputs, i.e., A_0 and A_1. In 4-input lines, one input-line is set to true at a time to get the respective binary code in the output side. Below are the block diagram and the truth table of the 4 to 2-line encoder.

Block Diagram:



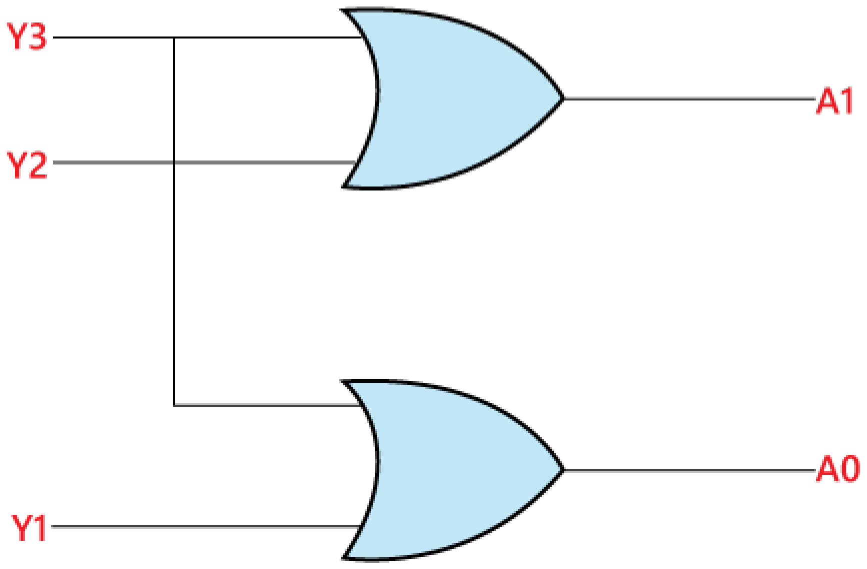
Truth Table:

INPUTS				OUTPUTS	
Y_3	Y_2	Y_1	Y_0	A_1	A_0
1	0	0	0	0	0
0	1	0	0	0	1
0	0	1	0	1	0
0	0	0	1	1	1

The logical expression of the term A0 and A1 is as follows:

$A_1=Y_3+Y_2$
$A_0=Y_3+Y_1$

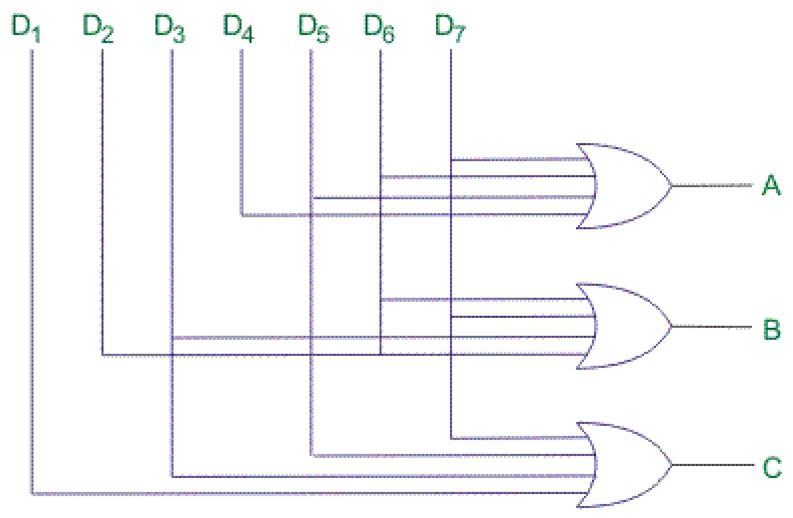
Logical circuit of the above expressions is given below:



known as Octal to Binary Encoder.

Octal Digit	Binary Equivalent		
	A	B	C
$D_0 \rightarrow 0$	0	0	0
$D_1 \rightarrow 1$	0	0	1
$D_2 \rightarrow 2$	0	1	0
$D_3 \rightarrow 3$	0	1	1
$D_4 \rightarrow 4$	1	0	0
$D_5 \rightarrow 5$	1	0	1
$D_6 \rightarrow 6$	1	1	0
$D_7 \rightarrow 7$	1	1	1

$$A = D_4 + D_5 + D_6 + D_7$$
$$B = D_2 + D_3 + D_6 + D_7$$
$$C = D_1 + D_3 + D_5 + D_7$$

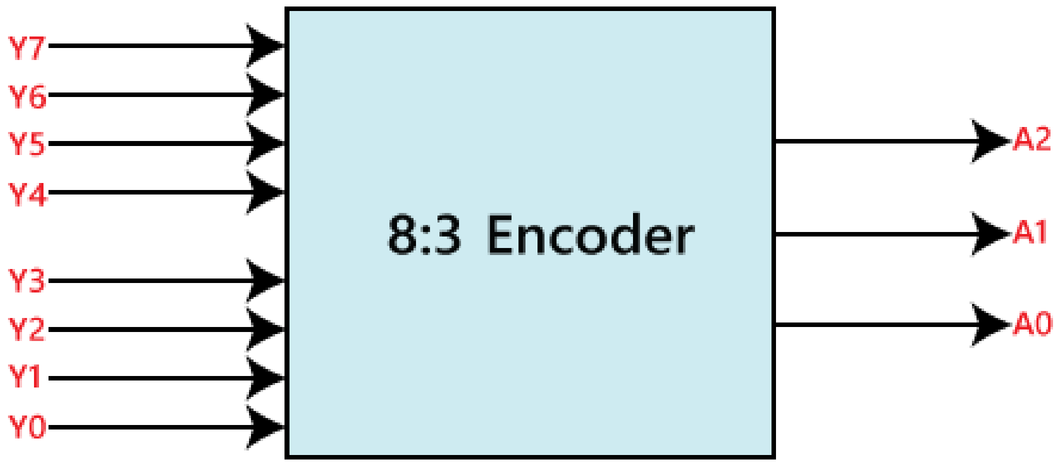


8 to 3-line Encoder:

In **8 to 3-line encoder**, there are a total of eight inputs, i.e., **Y0, Y1, Y2, Y3, Y4, Y5, Y6,** and **Y7** and three outputs, i.e., **A0, A1,** and **A2**. It is also

The 8 to 3-line Encoder is also known as **Octal to Binary Encoder**. In 8 to 3-line encoder, there is a total of eight inputs, i.e., Y_0, Y_1, Y_2, Y_3, Y_4, Y_5, Y_6, and Y_7 and three outputs, i.e., A_0, A1, and A_2. In 8-input lines, one input-line is set to true at a time to get the respective binary code in the output side. Below are the block diagram and the truth table of the 8 to 3-line encoder.

Block Diagram:



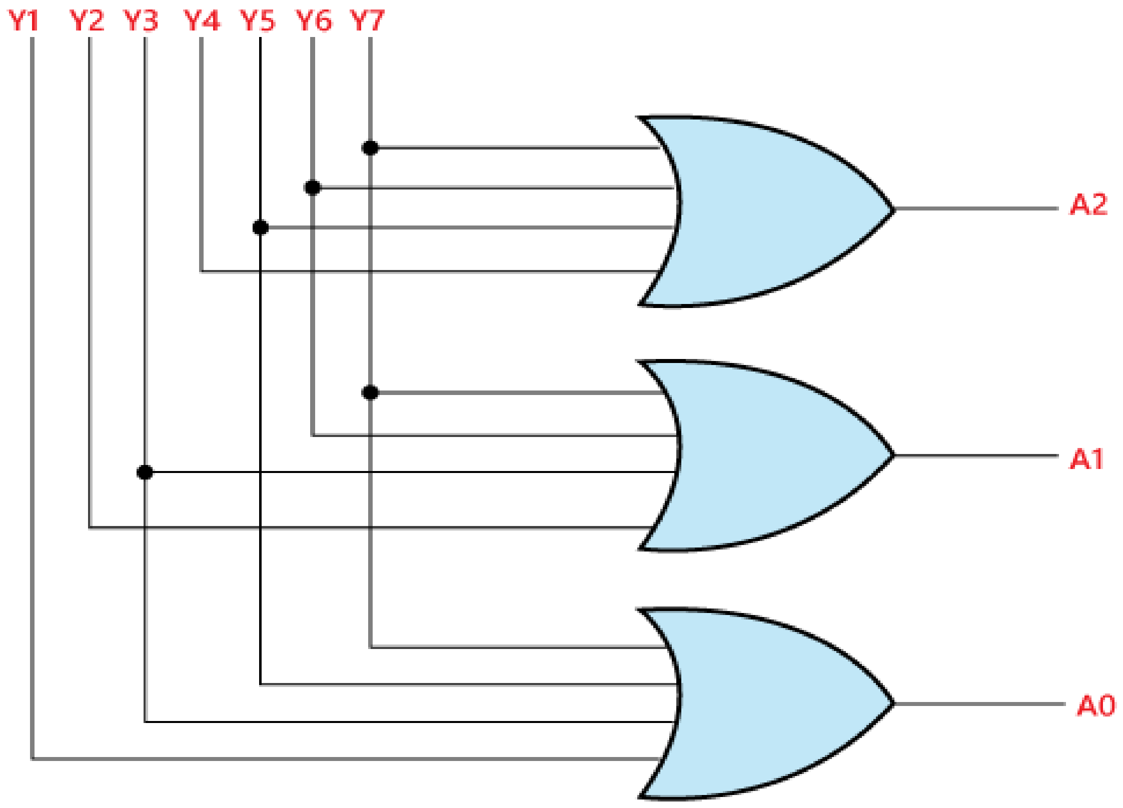
Truth Table:

INPUTS								OUTPUTS		
Y_7	Y_6	Y_5	Y_4	Y_3	Y_2	Y_1	Y_0	A_2	A_1	A_0
0	0	0	0	0	0	0	1	0	0	0
0	0	0	0	0	0	1	0	0	0	1
0	0	0	0	0	1	0	0	0	1	0
0	0	0	0	1	0	0	0	0	1	1
0	0	0	1	0	0	0	0	1	0	0
0	0	1	0	0	0	0	0	1	0	1
0	1	0	0	0	0	0	0	1	1	0
1	0	0	0	0	0	0	0	1	1	1

The logical expression of the term A0, A1, and A2 are as follows:

$A_2=Y_4+Y_5+Y_6+Y_7$
$A_1=Y_2+Y_3+Y_6+Y_7$
$A_0=Y_7+Y_5+Y_3+Y_1$

Logical circuit of the above expressions is given below:



Truth Table –

D7	D6	D5	D4	D3	D2	D1	D0	X	Y	Z
0	0	0	0	0	0	0	1	0	0	0
0	0	0	0	0	0	1	0	0	0	1
0	0	0	0	0	1	0	0	0	1	0
0	0	0	0	1	0	0	0	0	1	1
0	0	0	1	0	0	0	0	1	0	0
0	0	1	0	0	0	0	0	1	0	1
0	1	0	0	0	0	0	0	1	1	0

D7	D6	D5	D4	D3	D2	D1	D0	X	Y	Z
1	0	0	0	0	0	0	0	1	1	1

As seen from the truth table, the output is 000 when D0 is active; 001 when D1 is active; 010 when D2 is active and so on.

Implementation –

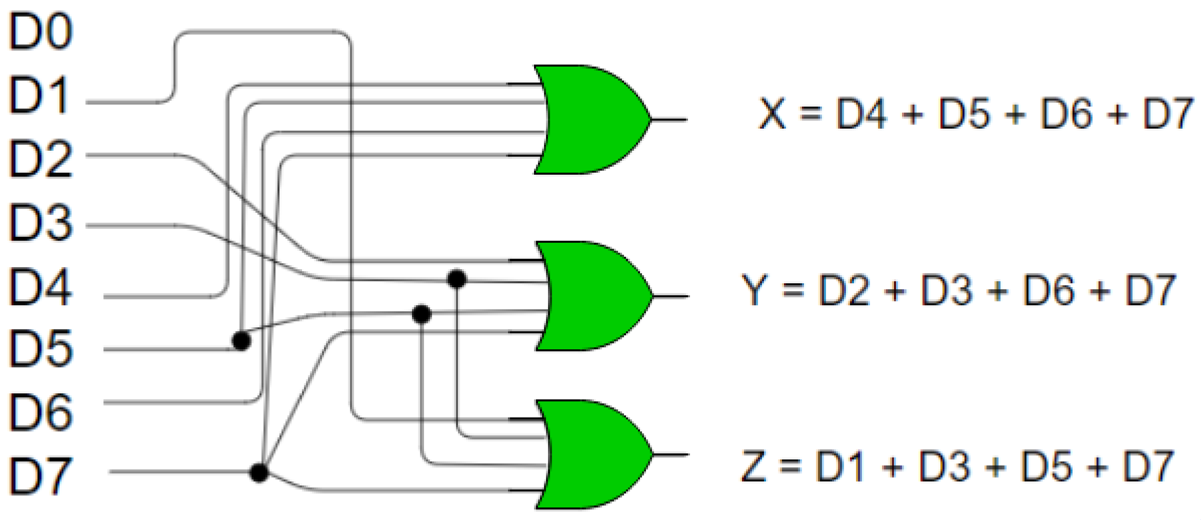
From the truth table, the output line Z is active when the input octal digit is 1, 3, 5 or 7. Similarly, Y is 1 when input octal digit is 2, 3, 6 or 7 and X is 1 for input octal digits 4, 5, 6 or 7. Hence, the Boolean functions would be:

X = D4 + D5 + D6 + D7

Y = D2 +D3 + D6 + D7

Z = D1 + D3 + D5 + D7

Hence, the encoder can be realised with OR gates as follows:



Truth Table –

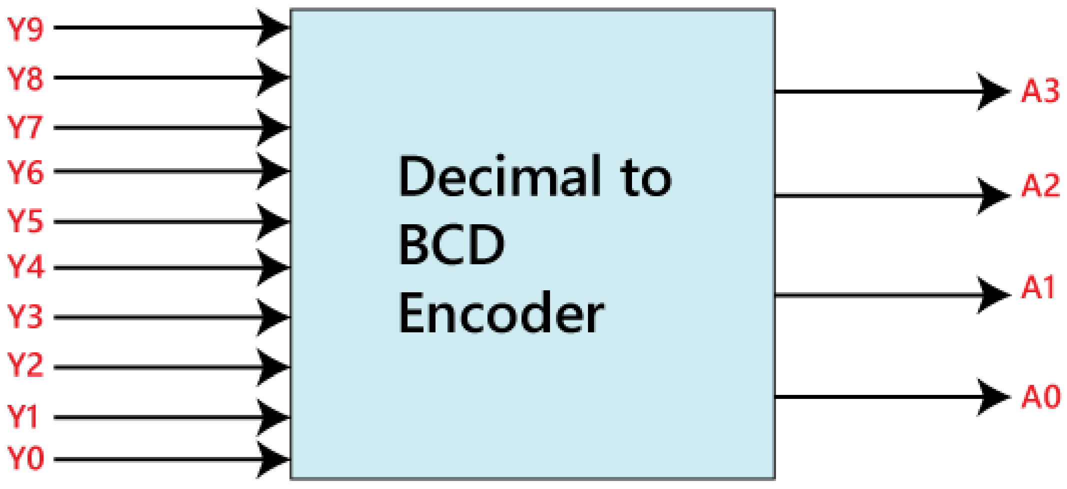
D3	D2	D1	D0	X	Y	V
0	0	0	0	x	x	0

D3	D2	D1	D0	X	Y	V
0	0	0	1	0	0	1
0	0	1	x	0	1	1
0	1	x	x	1	0	1
1	x	x	x	1	1	1

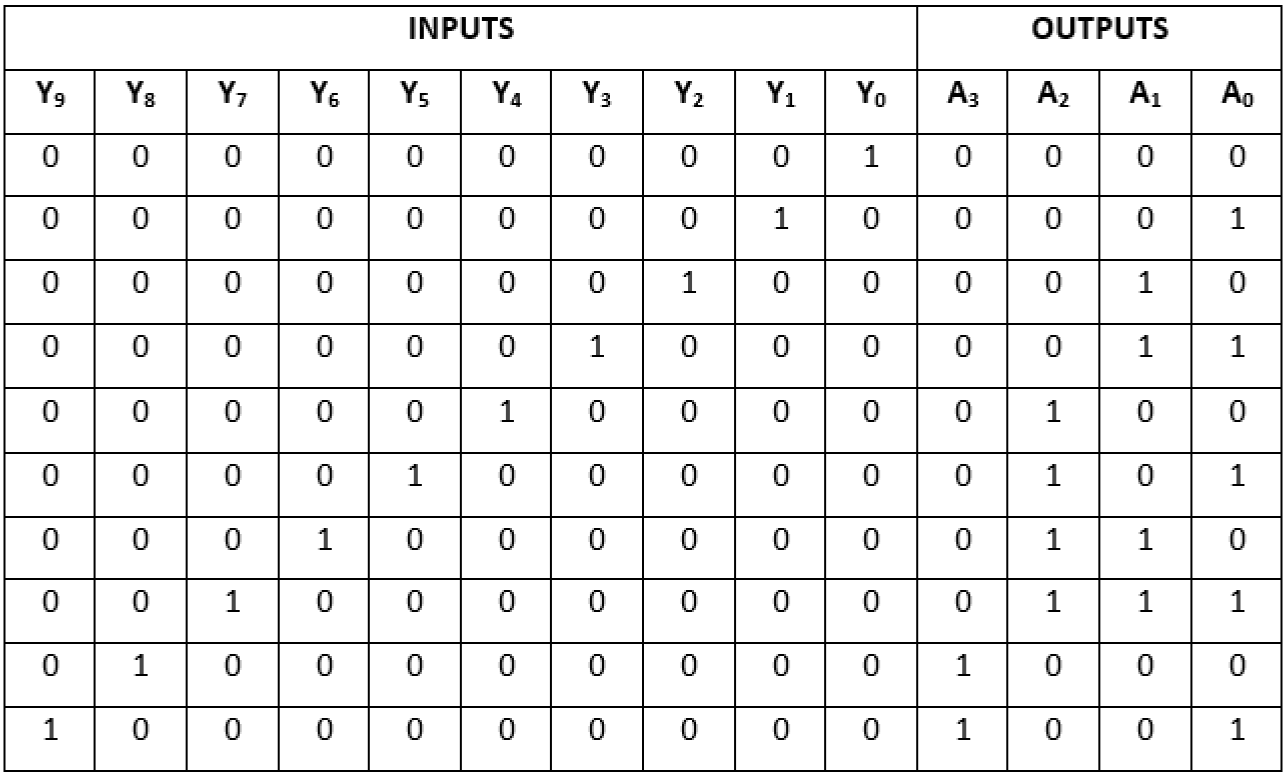
Decimal to BCD Encoder

The Octal to Binary Encoder is also known as **10 to 4 line Encoder**. In 10 to 4 line encoder, there are total of ten inputs, i.e., Y_0, Y_1, Y_2, Y_3, Y_4, Y_5, Y_6, Y_7, Y_8, and Y_9 and four outputs, i.e., A_0, A1, A_2, and A_3. In 10-input lines, one input-line is set to true at a time to get the respective **BCD code** in the output side. The block diagram and the truth table of the decimal to BCD encoder are given below.

Block Diagram:



Truth Table:

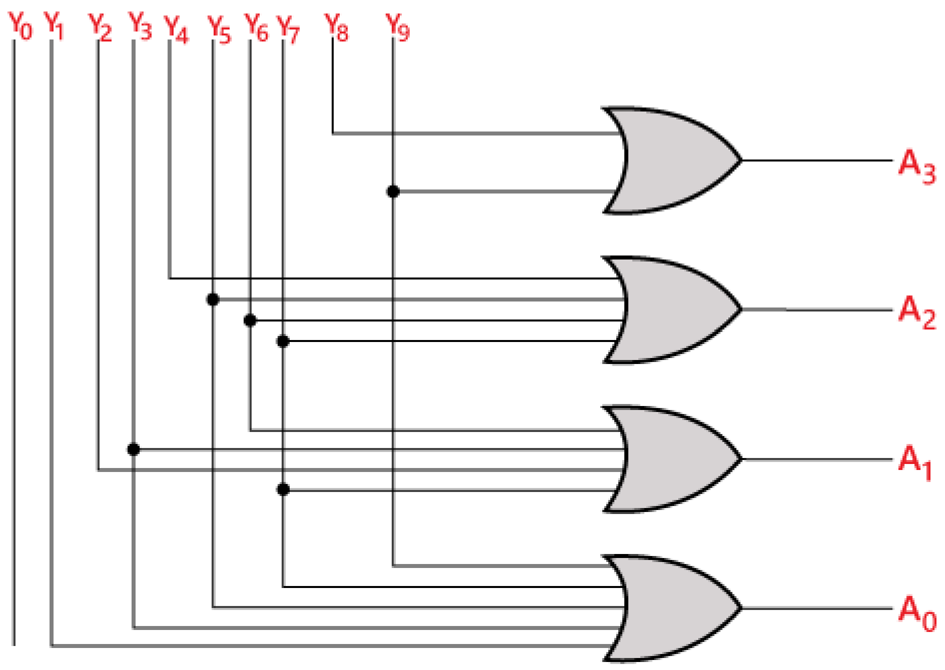


INPUTS										OUTPUTS			
Y_9	Y_8	Y_7	Y_6	Y_5	Y_4	Y_3	Y_2	Y_1	Y_0	A_3	A_2	A_1	A_0
0	0	0	0	0	0	0	0	0	1	0	0	0	0
0	0	0	0	0	0	0	0	1	0	0	0	0	1
0	0	0	0	0	0	0	1	0	0	0	0	1	0
0	0	0	0	0	0	1	0	0	0	0	0	1	1
0	0	0	0	0	1	0	0	0	0	0	1	0	0
0	0	0	0	1	0	0	0	0	0	0	1	0	1
0	0	0	1	0	0	0	0	0	0	0	1	1	0
0	0	1	0	0	0	0	0	0	0	0	1	1	1
0	1	0	0	0	0	0	0	0	0	1	0	0	0
1	0	0	0	0	0	0	0	0	0	1	0	0	1

The logical expression of the term A_0, A_1, A_2, and A_3 is as follows:

$A3 = Y9 + Y8$
$A2 = Y7 + Y6 + Y5 + Y4$
$A1 = Y7 + Y6 + Y3 + Y2$
$A0 = Y9 + Y7 + Y5 + Y3 + Y1$

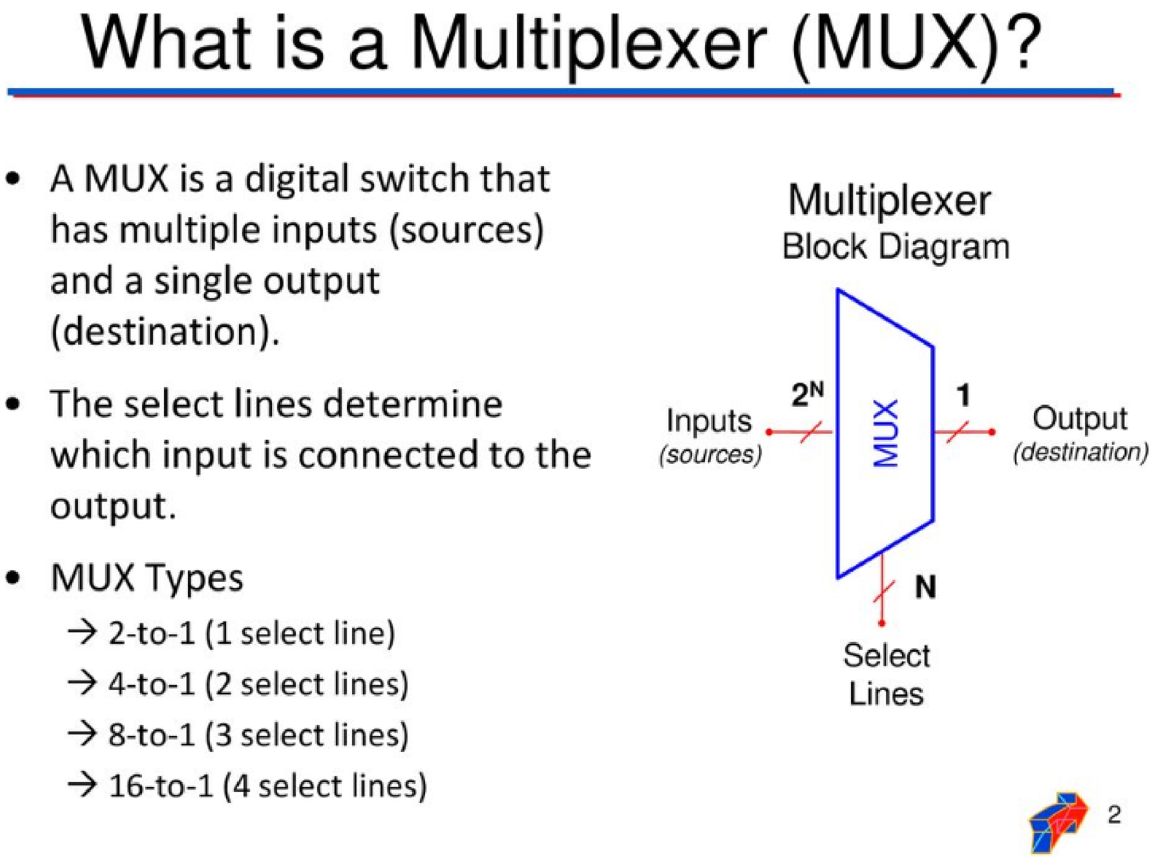
Logical circuit of the above expressions is given below:



S.no.	On the basis of	Encoder	Decoder
1.	**Basic**	It is a combinational circuit that basically converts the information signal to the coded digital bit stream.	It is also a combinational circuit that converts the coded bits to the original information. It performs the reverse operation of the encoder.
2.	**Input lines**	There are **2^n** input lines in the encoder.	There are **n** input lines in the decoder.
3.	**Output lines**	There are **n** output lines in the encoder.	There are **2^n** output lines in the decoder.
4.	**Operation**	The operation of the encoder is fairly simple.	Unlike encoder, the operation of decoder is complex.

5.	**Basic logic element**	The basic logic element that is used in encoders is **OR gate.**	In decoder, the basic logic element is **AND gate** along with the **NOT gate.**
6.	**Installation**	It is installed at the transmitting end.	It is installed at the receiving side.
7.	**Application**	Encoders are commonly used in Videos, E-mail, etc.	Decoders are commonly used in memory chips, microprocessors, etc.

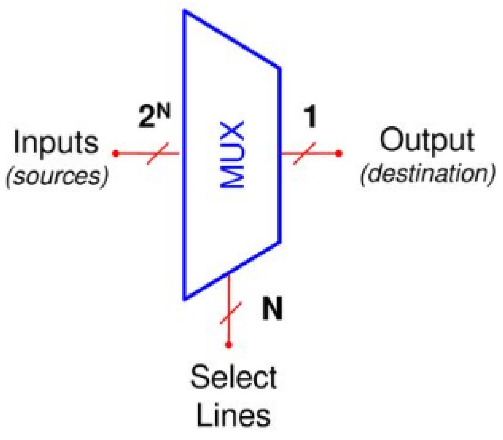
Multiplexer



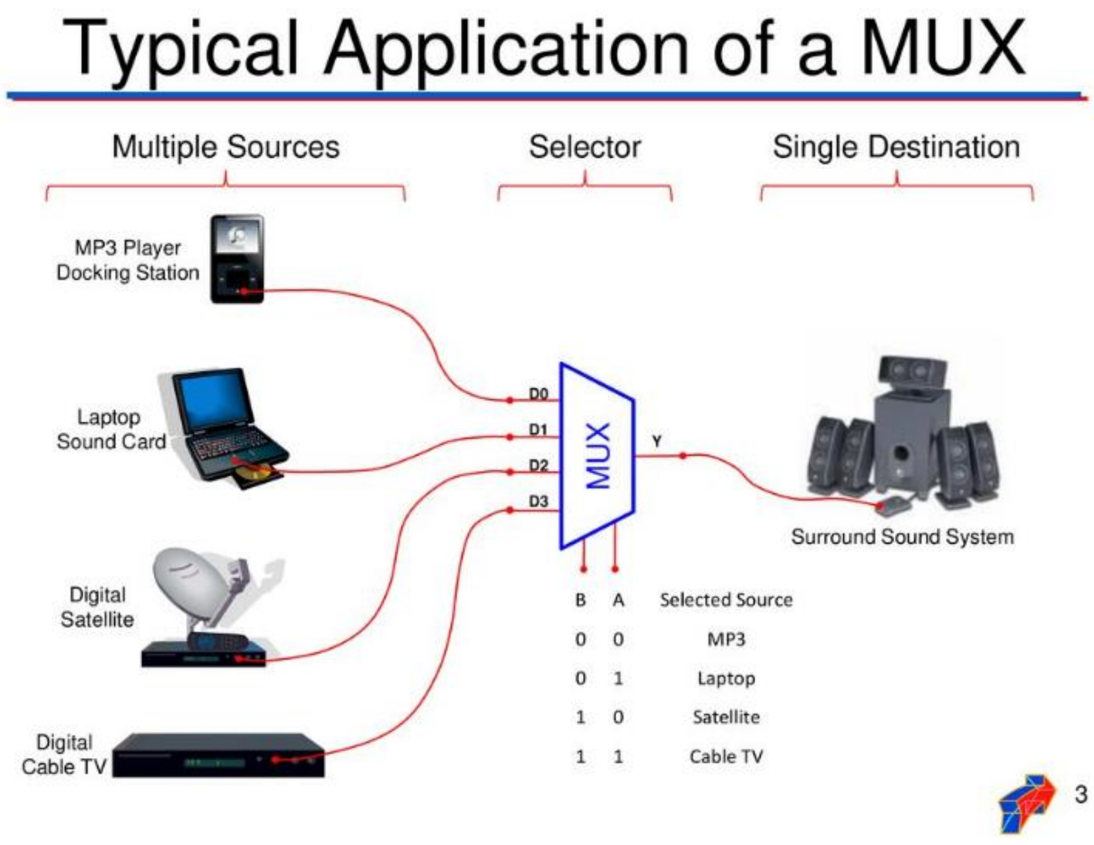
What is a Multiplexer (MUX)?

- A MUX is a digital switch that has multiple inputs (sources) and a single output (destination).
- The select lines determine which input is connected to the output.
- MUX Types
  - → 2-to-1 (1 select line)
  - → 4-to-1 (2 select lines)
  - → 8-to-1 (3 select lines)
  - → 16-to-1 (4 select lines)

Multiplexer Block Diagram



2



B	A	Selected Source
0	0	MP3
0	1	Laptop
1	0	Satellite
1	1	Cable TV

A multiplexer is a combinational circuit that has 2^n input lines and a single output line. Simply, the multiplexer is a multi-input and single-output combinational circuit. The binary information is received from the input lines and directed to the output line. On the basis of the values of the selection lines, one of these data inputs will be connected to the output.

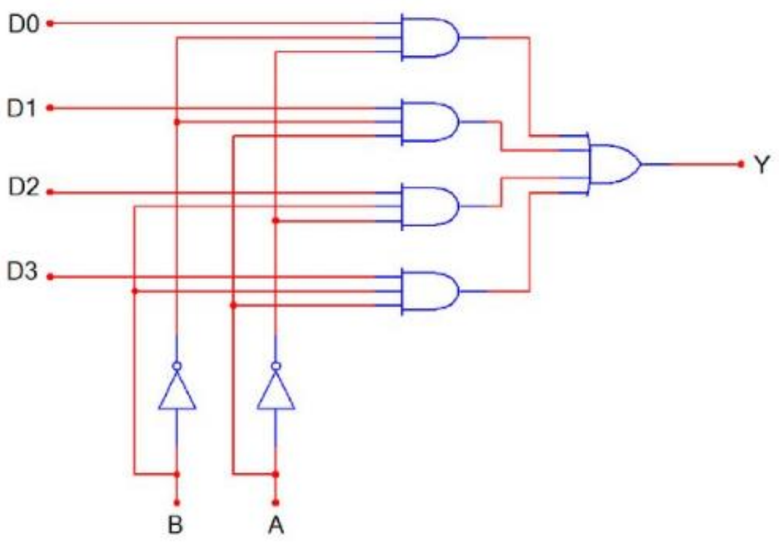
Unlike encoder and decoder, there are n selection lines and 2^n input lines. So, there is a total of 2^N possible combinations of inputs. A multiplexer is also treated as **Mux**.

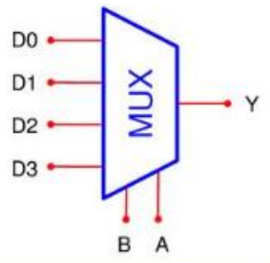
There are various types of the multiplexer which are as follows:

2×1 Multiplexer:

In 2×1 multiplexer, there are only two inputs, i.e., A_0 and A_1, 1 selection line, i.e., S_0 and single outputs, i.e., Y. On the basis of the combination of inputs which are present at the selection line S^0, one of these 2 inputs will be connected to the output. The block diagram and the truth table of the 2×1 multiplexer are given below.

4-to-1 Multiplexer (MUX)





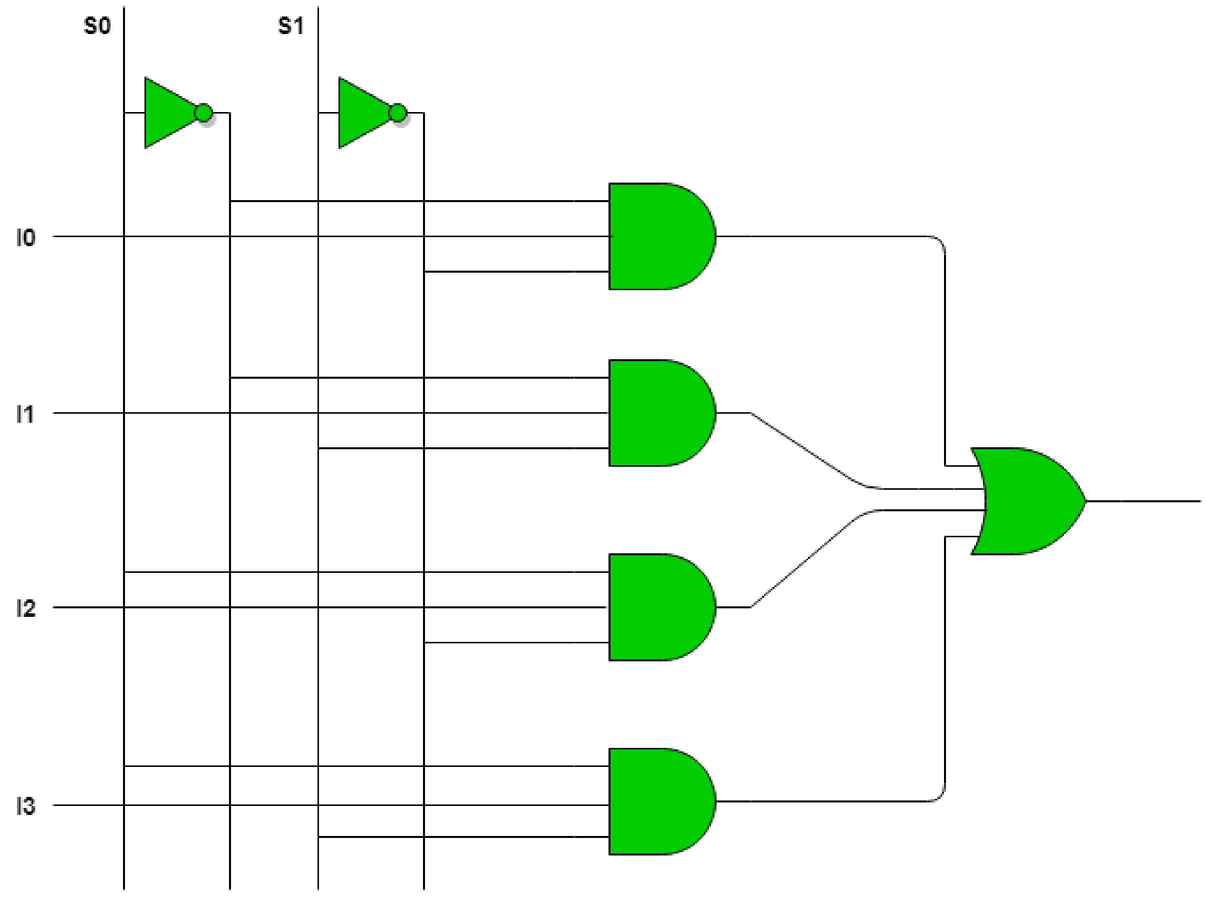
B	A	Y
0	0	D0
0	1	D1
1	0	D2
1	1	D3

Truth Table

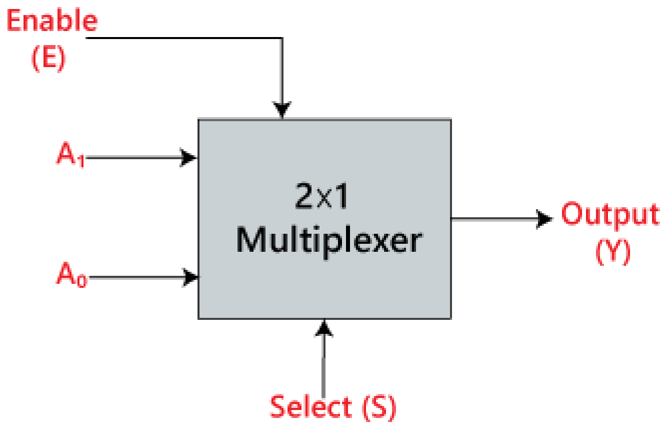
S0	S1	Y
0	0	I0
0	1	I1
1	0	I2
1	1	I3

So, final equation,

Y = S0'.S1'.I0 + S0'.S1.I1 + S0.S1'.I2 + S0.S1.I3



Block Diagram:



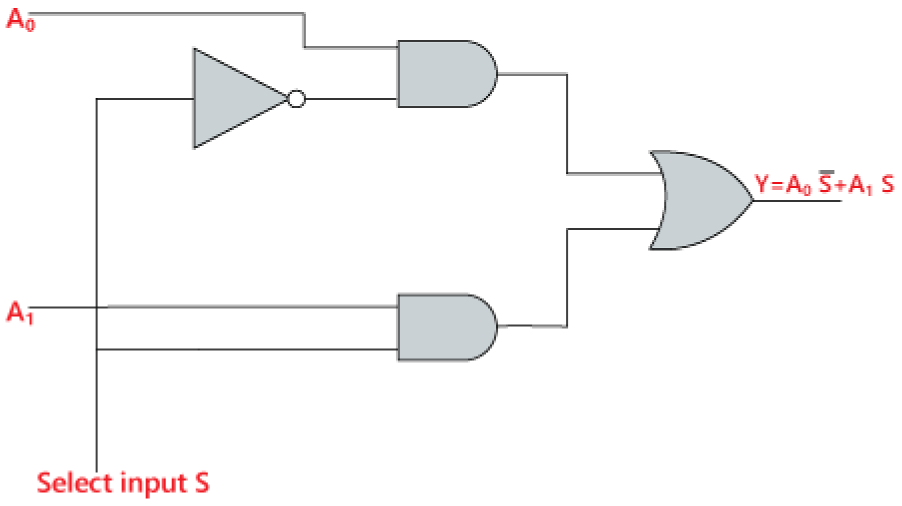
Truth Table:

INPUTS	Output
S_0	Y
0	A_0
1	A_1

The logical expression of the term Y is as follows:

$Y=S_0'.A_0+S_0.A_1$

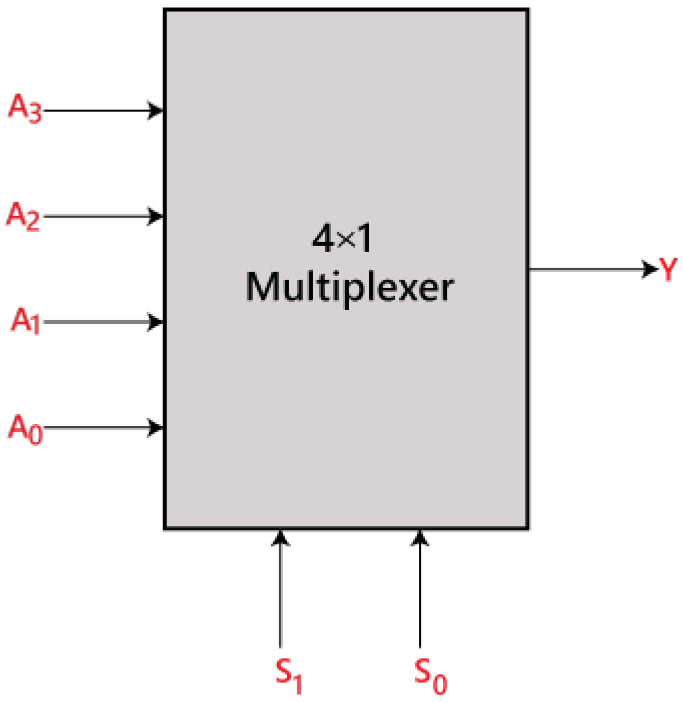
Logical circuit of the above expression is given below:



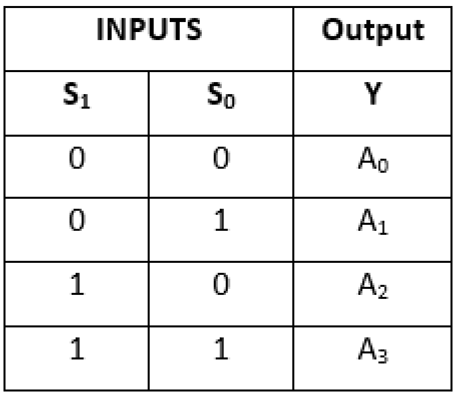
4×1 Multiplexer:

In the 4×1 multiplexer, there is a total of four inputs, i.e., A_0, A_1, A_2, and A_3, 2 selection lines, i.e., S_0 and S_1 and single output, i.e., Y. On the basis of the combination of inputs that are present at the selection lines S^0 and S_1, one of these 4 inputs are connected to the output. The block diagram and the truth table of the 4×1 multiplexer are given below.

Block Diagram:



Truth Table:

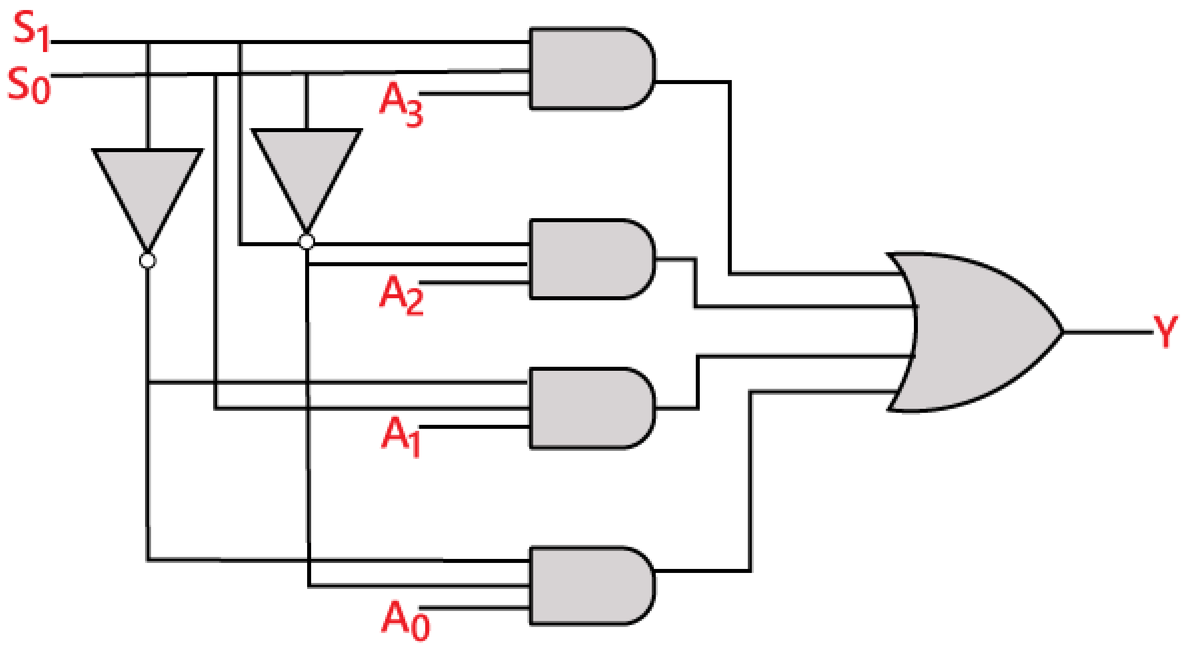


INPUTS		Output
S_1	S_0	Y
0	0	A_0
0	1	A_1
1	0	A_2
1	1	A_3

The logical expression of the term Y is as follows:

$Y=S_1' \, S_0' \, A_0+S_1' \, S_0 \, A_1+S_1 \, S_0' \, A_2+S_1 \, S_0 \, A_3$

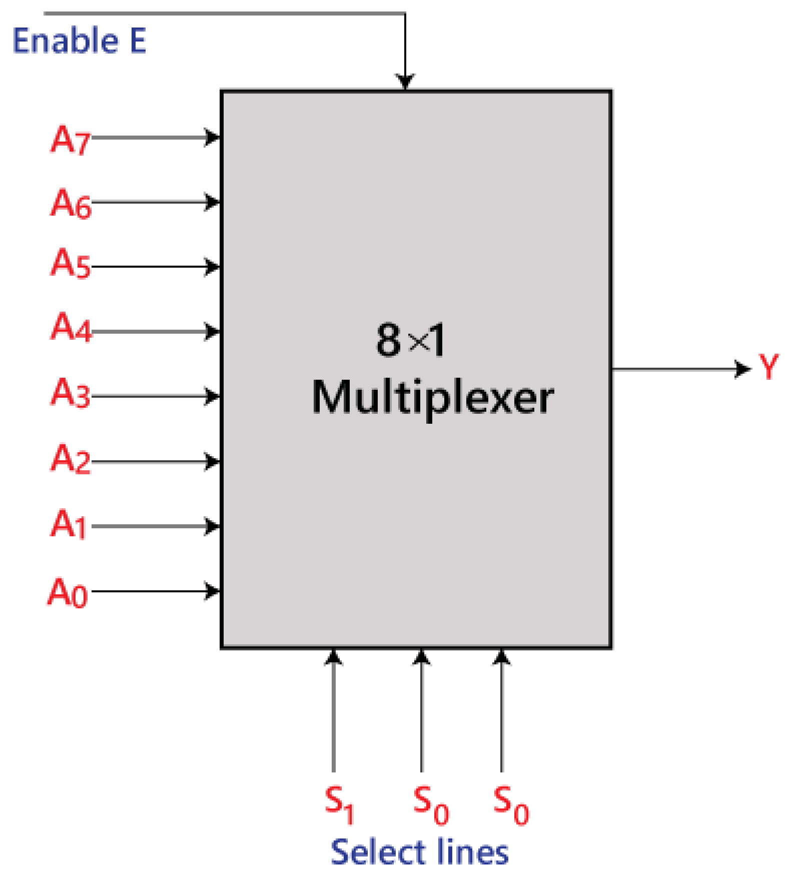
Logical circuit of the above expression is given below:



8 to 1 Multiplexer

In the 8 to 1 multiplexer, there are total eight inputs, i.e., A_0, A_1, A_2, A_3, A_4, A_5, A_6, and A_7, 3 selection lines, i.e., S_0, $S_1$and S_2 and single output, i.e., Y. On the basis of the combination of inputs that are present at the selection lines S^0, $S^{1,}$ and S_2, one of these 8 inputs are connected to the output. The block diagram and the truth table of the 8×1 multiplexer are given below.

Block Diagram:



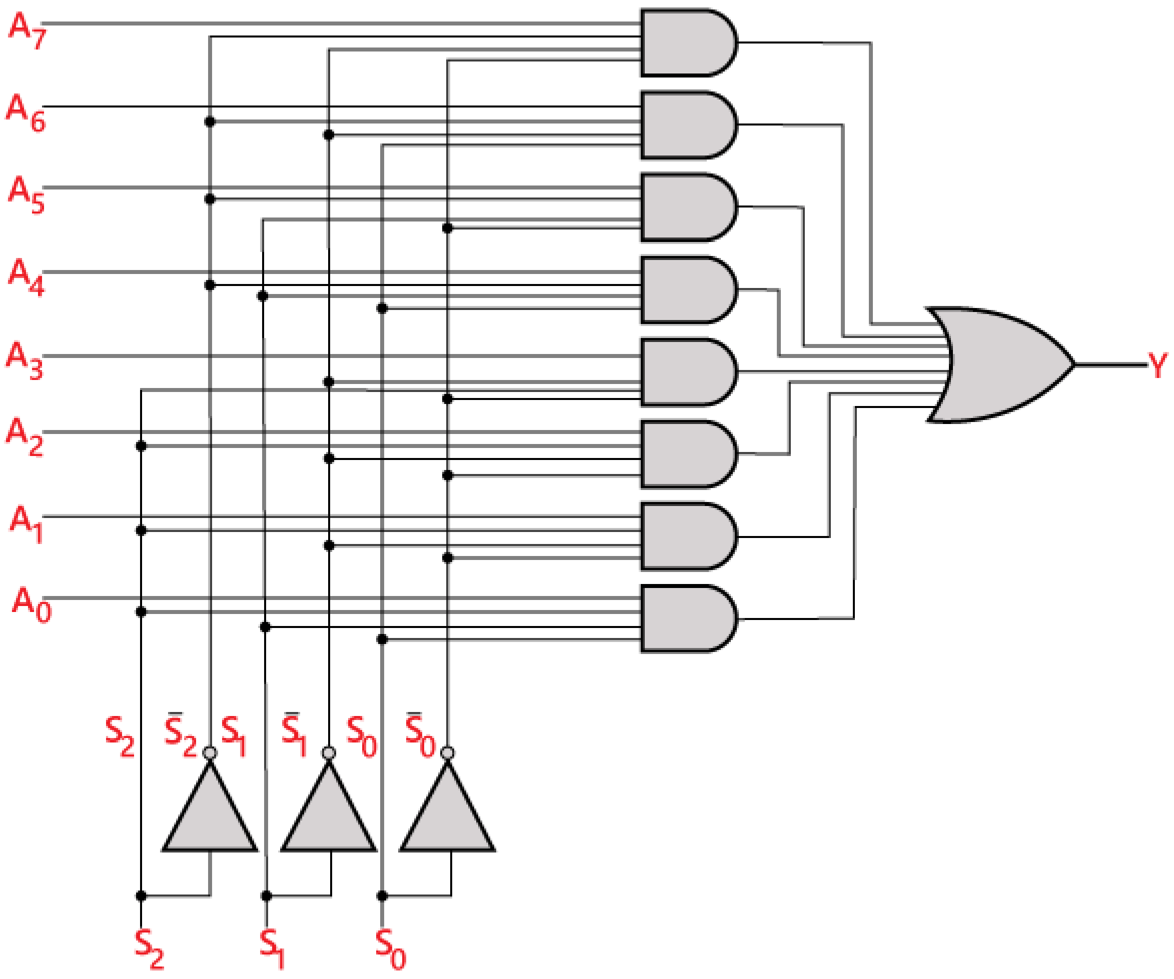
Truth Table:

INPUTS			Output
S_2	**S_1**	**S_0**	**Y**
0	0	0	A_0
0	0	1	A_1
0	1	0	A_2
0	1	1	A_3
1	0	0	A_4
1	0	1	A_5
1	1	0	A_6
1	1	1	A_7

The logical expression of the term Y is as follows:

$Y = S_0'.S_1'.S_2'.A_0 + S_0.S_1'.S_2'.A_1 + S_0'.S_1.S_2'.A_2 + S_0.S_1.S_2'.A_3 + S_0'.S_1'.S_2\ A_4 + S_0.S_1'.S_2\ A_5 + S_0'.S_1.S_2\ .A_6 + S_0.S_1.S_3.A_7$

Logical circuit of the above expression is given below:


A_7
A_6
A_5
A_4
A_3
A_2
A_1
A_0
S_2
$\bar{S}_2$
S_1
$\bar{S}_1$
S_0
$\bar{S}_0$
S_2
S_1
S_0
Y