Combinational Circuits

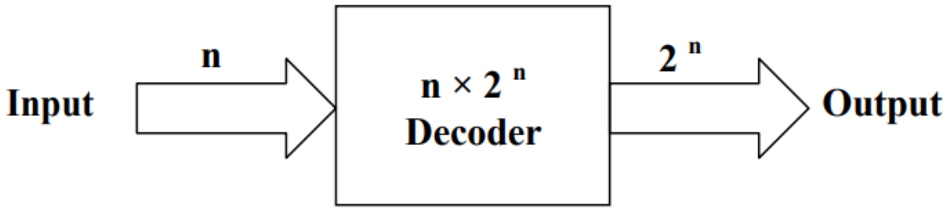
Data Transmission

1- Decoder and Encoder

The encoders and decoders play an essential role in digital electronics projects; encoders and decoders are combinational circuits that used to convert data from one form to another form. These are frequently used in communication system such as telecommunication, networking, etc.. to transfer data from one end to the other end. Similarly, in the digital domain, for easy transmission of data, it is often encrypted or placed within codes, and then transmitted. At the receiver, the coded data is decrypted or gathered from the code and is processed in order to be displayed or given to the load accordingly.

A. Decoder:

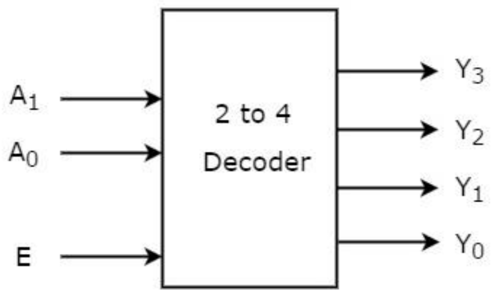
- The process of taking some type of code and determining what it represents in terms of a recognizable number or character is called decoding.
- A decoder is a combinational logic circuit that performs the decoding function, and produce an output that indicates the (meaning) of the input code.
- The decoder is an important part of the system which selects the cells to be read from and write into. This particular circuit is called a decoder matrix, or simply a decoder, and has a characteristic that for each of the possible 2^n binary input number which can be taken by the n input cells, the matrix will have a unique one of its 2n output lines selected.
- Decoders have a wide variety of applications in digital systems such as data demultiplexing, digital display, digital to analog converting, memory addressing, etc.



Block diagram of decoder

- Decoders have a wide variety of applications in digital systems such as data demultiplexing, digital display, digital to analog converting, memory addressing, etc.

- **2-to-4 Decoder:**
  Let 2 to 4 Decoder has two inputs A_1 & A_0 and four outputs Y_3, Y_2, Y_1 & Y_0. The **block diagram** of 2 to 4 decoder is shown in the following figure.



One of these four outputs will be ‘1’ for each combination of inputs when enable, E is ‘1’. The **Truth table** of 2 to 4 decoder is shown below.

Enable	**Inputs**		**Outputs**			
E	**A_1**	**A_0**	**Y_3**	**Y_2**	**Y_1**	**Y_0**
0	x	x	0	0	0	0
1	0	0	0	0	0	1
1	0	1	0	0	1	0
1	1	0	0	1	0	0
1	1	1	1	0	0	0

From Truth table, we can write the **Boolean functions** for each output as

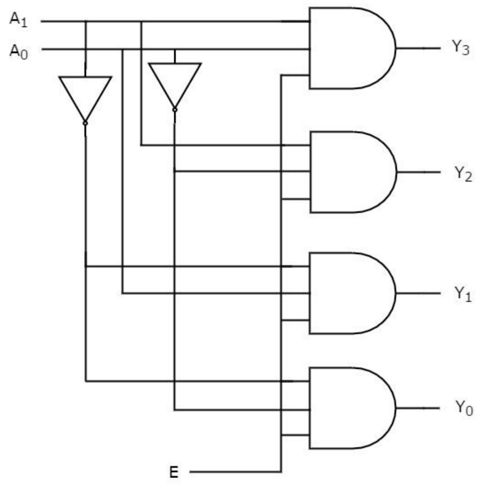
$Y_3=EA_1A_0$

$Y_2=EA_1A_0'$

$Y_1=EA_1'A0$

$Y_0=EA_1'A_0'$

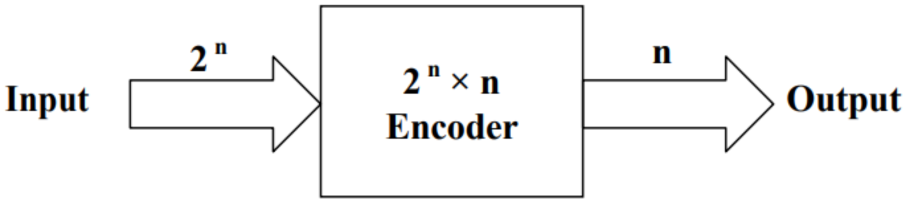
Each output is having one product term. So, there are four product terms in total. We can implement these four product terms by using four AND gates having three inputs each & two inverters. The **circuit diagram** of 2 to 4 decoder is shown in the following figure.



Therefore, the outputs of 2 to 4 decoder are nothing but the **min terms** of two input variables A_1 & A_0, when enable, E is equal to one. If enable, E is zero, then all the outputs of decoder will be equal to zero.

B. Encoder:

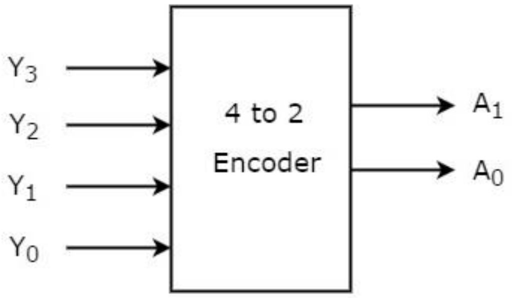
- An **Encoder** is a combinational circuit that performs the reverse operation of Decoder.
- It has maximum of 2^n input lines and 'n' output lines. It will produce a binary code equivalent to the input, which is active High. Therefore, the encoder encodes 2^n input lines with 'n' bits.
- It is optional to represent the enable signal in encoders.



Block diagram of an encoder

- **4-to-2 Encoder:**

Let 4 to 2 Encoder has four inputs Y_3, Y_2, Y_1 & Y_0 and two outputs A_1 & A_0. The **block diagram** of 4-to-2 Encoder is shown in the following figure.



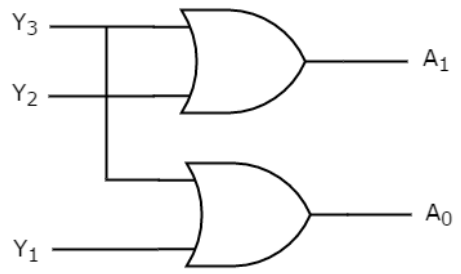
At any time, only one of these 4 inputs can be '1' in order to get the respective binary code at the output. The **Truth table** of 4 to 2 encoder is shown below.

Inputs				**Outputs**	
Y_3	**Y_2**	**Y_1**	**Y_0**	**A_1**	**A_0**
0	0	0	1	0	0
0	0	1	0	0	1
0	1	0	0	1	0
1	0	0	0	1	1

From Truth table, we can write the **Boolean functions** for each output as

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

We can implement the above two Boolean functions by using two input OR gates. The **circuit diagram** of 4 to 2 encoder is shown in the following figure.



Drawbacks of Encoder

Following are the drawbacks of normal encoder.

- There is an ambiguity, when all outputs of encoder are equal to zero. Because, it could be the code corresponding to the inputs, when only least significant input is one or when all inputs are zero.
- If more than one input is active High, then the encoder produces an output, which may not be the correct code. For **example**, if both Y_3 and Y_6 are '1', then the encoder produces 111 at the output. This is neither equivalent code corresponding to Y_3, when it is '1' nor the equivalent code corresponding to Y_6, when it is '1'.

So, to overcome these difficulties, we should assign priorities to each input of encoder. Then, the output of encoder will be the binary code corresponding to the active High inputs, which has higher priority. This encoder is called as **priority encoder**.

Priority Encoder

A 4 to 2 priority encoder has four inputs Y_3, Y_2, Y_1 & Y_0 and two outputs A_1 & A_0. Here, the input, Y_3 has the highest priority, whereas the input, Y_0 has the lowest priority. In this case, even if more than one input is '1' at the same time, the output will be the binary code corresponding to the input, which is having **higher priority**.

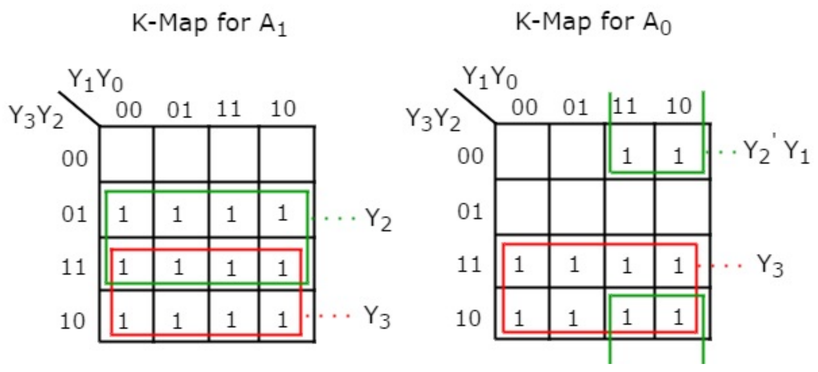
We considered one more **output, V** in order to know, whether the code available at outputs is valid or not.

- If at least one input of the encoder is '1', then the code available at outputs is a valid one. In this case, the output, V will be equal to 1.
- If all the inputs of encoder are '0', then the code available at outputs is not a valid one. In this case, the output, V will be equal to 0.

The **Truth table** of 4 to 2 priority encoder is shown below.

Inputs				**Outputs**		
Y_3	**Y_2**	**Y_1**	**Y_0**	**A_1**	**A_0**	**V**
0	0	0	0	0	0	0
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

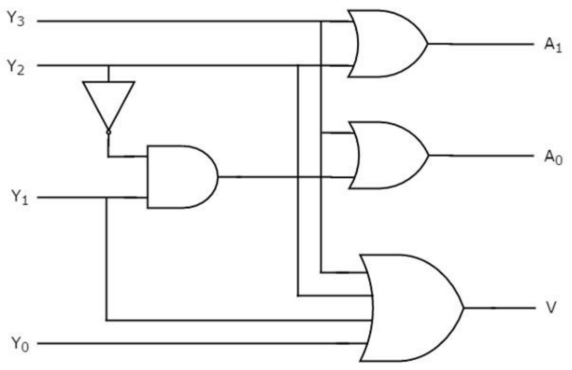
Use **4 variable K-maps** for getting simplified expressions for each output.



The simplified **Boolean functions** are

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

Similarly, we will get the Boolean function of output, V as
$V=Y_3+Y_2+Y_1+Y_0$
We can implement the above Boolean functions using logic gates. The **circuit diagram** of 4 to 2 priority encoder is shown in the following figure.



The above circuit diagram contains two 2-input OR gates, one 4-input OR gate, one 2input AND gate & an inverter. Here AND gate & inverter combination are used for producing a valid code at the outputs, even when multiple inputs are equal to ‘1’ at the same time. Hence, this circuit encodes the four inputs with two bits based on the **priority** assigned to each input.

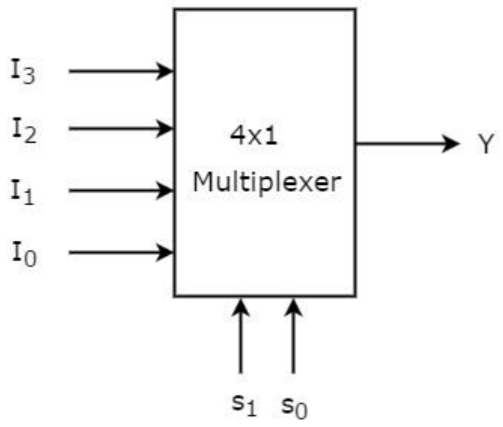
2- Multiplexer and Demultiplexer

The multiplexers and demultiplexers are digital electronic devices that are used to control applications.
A simple data transmission system can be implemented using a multiplexer and a demultiplexer in conjunction with an interconnecting single line link. Such a system used over a relatively short distance such as 500 meters can result in a significant reduction in the number of lines required to transmit the data.

C. Multiplexer:

- **Multiplexer** is a combinational circuit that has maximum of 2^n data inputs, 'n' selection lines and single output line. One of these data inputs will be connected to the output based on the values of selection lines.
- Since there are 'n' selection lines, there will be 2^n possible combinations of zeros and ones. So, each combination will select only one data input. Multiplexer is also called as **Mux**.

- **4x1 Multiplexer:**
  4x1 Multiplexer has four data inputs I_3, I_2, I_1 & I_0, two selection lines s_1 & s_0 and one output Y. The block diagram of 4x1 Multiplexer is shown in the following figure.



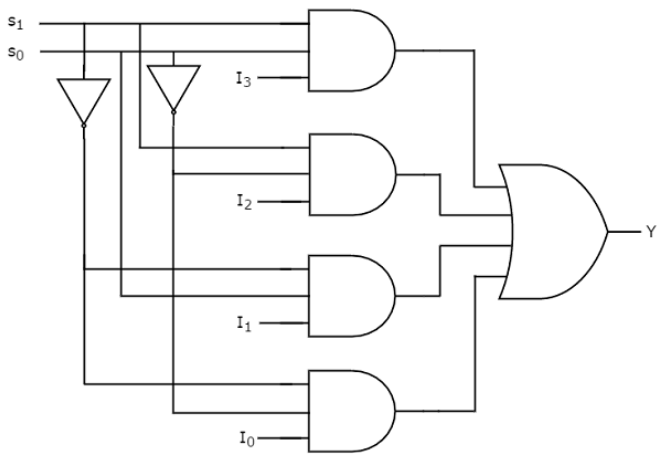
One of these 4 inputs will be connected to the output based on the combination of inputs present at these two selection lines. **Truth table** of 4x1 Multiplexer is shown below.

Selection Lines		**Output**
S_1	**S_0**	**Y**
0	0	I_0
0	1	I_1
1	0	I_2
1	1	I_3

From Truth table, we can directly write the **Boolean function** for output, Y as

$Y=S_1'S_0'I_0 + S_1'S_0I_1 + S_1S_0'I_2 + S_1S_0I_3$

We can implement this Boolean function using Inverters, AND gates & OR gate. The **circuit diagram** of 4x1 multiplexer is shown in the following figure.



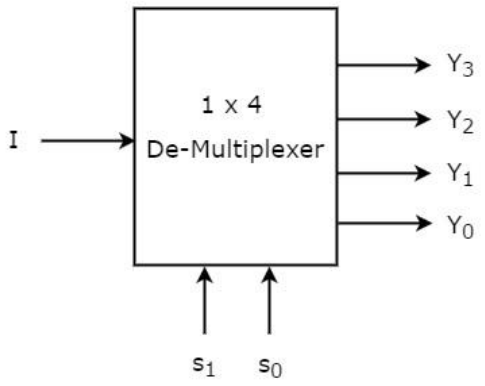
We can easily understand the operation of the above circuit. Similarly, you can implement 8x1 Multiplexer and 16x1 multiplexer by following the same procedure.

D. Multiplexer:

- **De-Multiplexer** is a combinational circuit that performs the reverse operation of Multiplexer. It has single input, 'n' selection lines and maximum of 2^n outputs. The input will be connected to one of these outputs based on the values of selection lines.
- Since there are 'n' selection lines, there will be 2^n possible combinations of zeros and ones. So, each combination can select only one output. De-Multiplexer is also called as **De-Mux**.

- **1x4 De-Multiplexer**

  1x4 De-Multiplexer has one input I, two selection lines, s_1 & s_0 and four outputs Y_3, Y_2, Y_1 &Y_0. The **block diagram** of 1x4 De-Multiplexer is shown in the following figure.



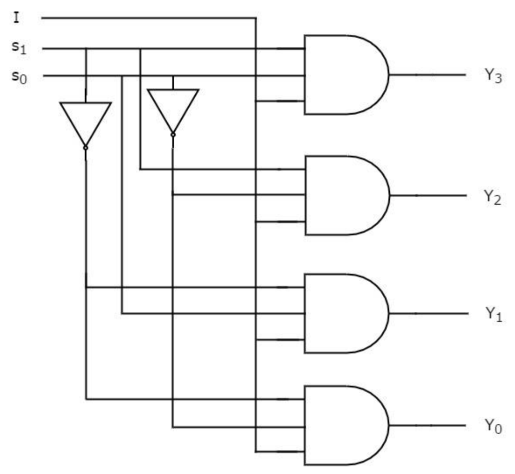
The single input 'I' will be connected to one of the four outputs, Y_3 to Y_0 based on the values of selection lines s_1 & s0. The **Truth table** of 1x4 De-Multiplexer is shown below.

Selection Inputs		**Outputs**			
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

From the above Truth table, we can directly write the **Boolean functions** for each output as

$Y_3=S_1S_0I$
$Y_2=S_1S_0'I$
$Y_1=s_1's_0I$
$Y_0=s_1's_0'I$

We can implement these Boolean functions using Inverters & 3-input AND gates. The **circuit diagram** of 1x4 De-Multiplexer is shown in the following figure.



We can easily understand the operation of the above circuit. Similarly, you can implement 1x8 De-Multiplexer and 1x16 De-Multiplexer by following the same procedure.