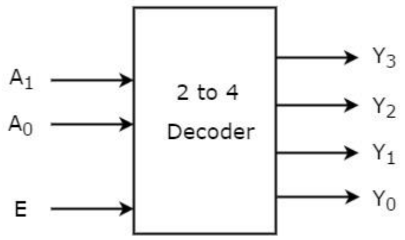
DLD – LAB # 08 (Encoders Vs Decoders)

Decoders:

Decoder is a combinational circuit that has 'n' input lines and maximum of 2^n output lines. One of these outputs will be active High based on the combination of inputs present, when the decoder is enabled. That means decoder detects a particular code. The outputs of the decoder are nothing but the **min terms** of 'n' input variables lines, when it is enabled.

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

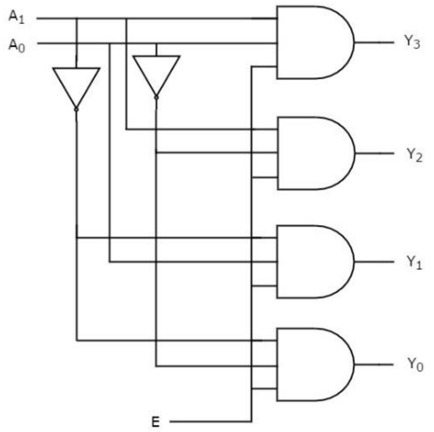
$$Y_3=E.A_1.A_0$$

$$Y_2=E.A_1.A_0'$$

$$Y_1=E.A_1'.A_0$$

$$Y_0=E.A_1'.A_0'$$

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.



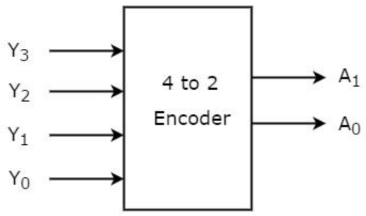
Similarly, 3 to 8 decoder produces eight min terms of three input variables A_2, A_1 & A_0 and 4 to 16 decoder produces sixteen min terms of four input variables A_3, A_2, A_1 & A_0.

Encoders:

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.

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.

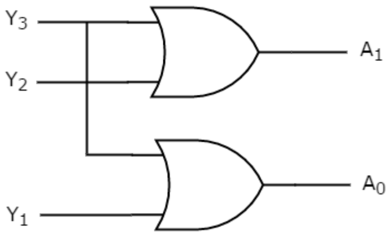
Inputs				Outputs	
Y_3	Y_2	Y_1	Y_0	A_1	A_0
0	0	0	1	0	0
0	0	1	0	0	1
0	1	0	0	1	0
1	0	0	0	1	1

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.



8-3 Binary Encoder

Octal to binary Encoder has eight inputs, Y_7 to Y_0 and three outputs A_2, A_1 & A_0. Octal to binary encoder is nothing but 8 to 3 encoder. The **block diagram** of octal to binary Encoder is shown in the following figure.

Drawbacks of Simple 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 Encoders:

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 binarybinary 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

Decimal	Y3	Y2	Y1	Y0		A1	A0	Enable
0	0	0	0	0		0	0	0
1	0	0	0	1		0	0	1
2	0	0	1	0		0	1	1
3	0	0	1	1		0	1	1
4	0	1	0	0		1	0	1
5	0	1	0	1		1	0	1
6	0	1	1	0		1	0	1
7	0	1	1	1		1	0	1
8	1	0	0	0		1	1	1
9	1	0	0	1		1	1	1
10	1	0	1	0		1	1	1
11	1	0	1	1		1	1	1
12	1	1	0	0		1	1	1
13	1	1	0	1		1	1	1
14	1	1	1	0		1	1	1
15	1	1	1	1		1	1	1

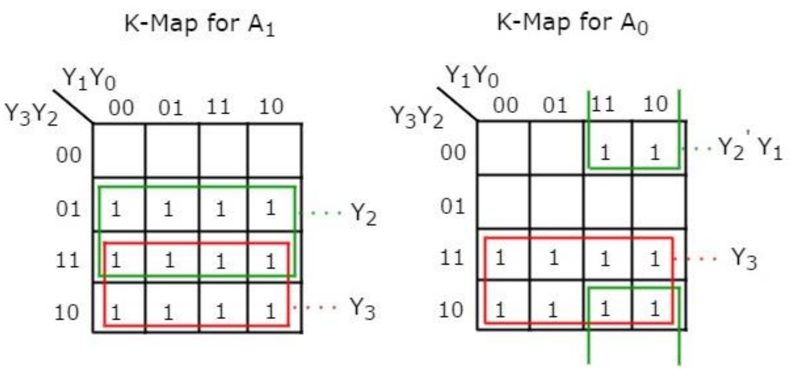
Red: All inputs zero and enable '0' - means: invalid input and no output

Orange: Y0 is 1, so output will be binary value of 0 = 0 0 (for 1)

Green: Y1 is 1 (high priority), so output will be binary value of 1 = 0 1 (2-3)

Light Blue: Y2 is 1 (high priority), so output will be binary value of 2 = 1 0 (4-7)

Dark Blue: Y3 is 1 (high priority), so output will be binary value of 3 = 1 1 (8-15)



The simplified **Boolean functions** are

$$A_1 = Y_3 + Y_2$$

$$A_0 = Y_3 + Y_2' Y_1$$

$$V = Y_3 + Y_2 + Y_1 + Y_0$$

LAB Tasks:

Design 3-to-8 **decoder** truth table, Boolean expression for each output column and its circuit diagram?

Design 3-to-8 **Priority Encoder** truth table, Boolean expression for each output column and its circuit diagram?