Syllabus of the course SM2:

Chapter 1: Combinational circuits

- Introduction
- Definition of combinational circuits
- Steps of Construction of Combinational Circuits
- Half Adder
- Full Adder
- Subtractor
- Encoder / decoder /Priority encoder
- Multiplexer / demultiplexer
- Converters/ Comparator

Chapter 2: Sequential circuits

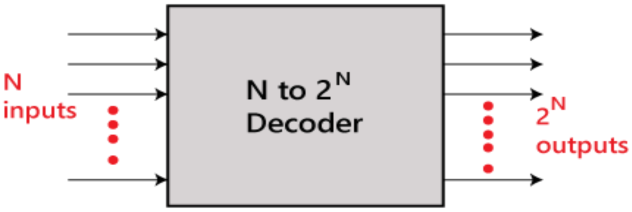
- Introduction
- Definition of sequential circuits
- Flip-flops (RS, JK, D, T)
- The registers
- The Counters
- Automata of Moore and Mearly

Chapter 1: Combinational circuits

6. Decoder

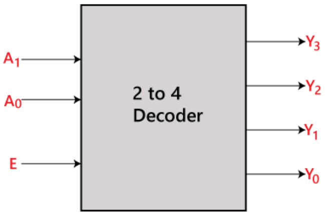
A decoder is a combinational circuit having n inputs and to a maximum of m = 2^n outputs, when the validation signal is active, only the output whose number corresponds to the binary value displayed on the input is activated; and all other outputs are inactive. That is to say for each input combination only one output line is activated at a time.

A device with n binary inputs and 2^n binary outputs. Each bit pattern at the input causes exactly one of the 2^n outputs to equal 1. So, a decoder can be thought of as a device for converting an n-bit input to one of 2^n outputs.



6.1. 2-to-4-line Decoder (2-4)

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.



Truth Table:

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

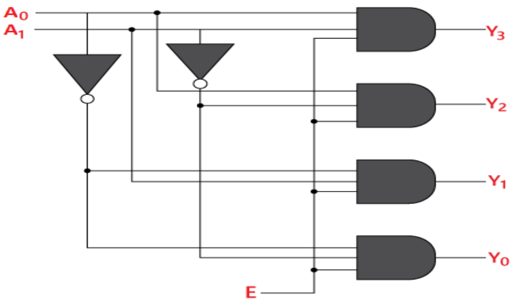
The logical expression of the term Y0, Y0, Y2, and Y3 is as follows:

Y_3=E.A_1.A_0

Y_2=E.A_1.A_0'

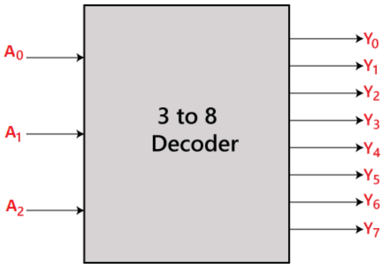
Y_1=E.A_1'.A_0

Y_0=E.A_1'.A_0'

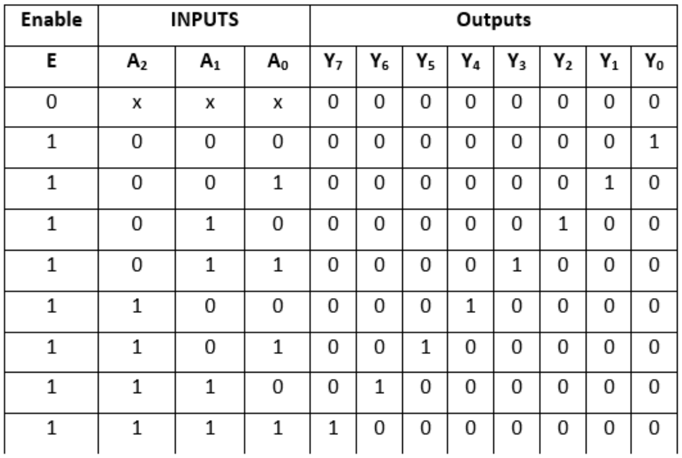


6.1. 3-to-8-line decoder:

Block Diagram:



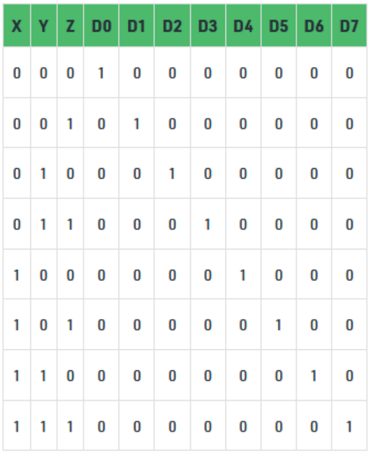
Truth Table:



Enable	INPUTS			Outputs							
E	A_2	A_1	A_0	Y_7	Y_6	Y_5	Y_4	Y_3	Y_2	Y_1	Y_0
0	x	x	x	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	1
1	0	0	1	0	0	0	0	0	0	1	0
1	0	1	0	0	0	0	0	0	1	0	0
1	0	1	1	0	0	0	0	1	0	0	0
1	1	0	0	0	0	0	1	0	0	0	0
1	1	0	1	0	0	1	0	0	0	0	0
1	1	1	0	0	1	0	0	0	0	0	0
1	1	1	1	1	0	0	0	0	0	0	0

The logical expression of the term Y_0, Y_1, Y_2, Y_3, Y_4, Y_5, Y_6, and Y_7 is as follows:

$Y_0=A_0'.A_1'.A_2'$
$Y_1=A_0.A_1'.A_2'$
$Y_2=A_0'.A_1.A_2'$
$Y_3=A_0.A_1.A_2'$
$Y_4=A_0'.A_1'.A_2$
$Y_5=A_0.A_1'.A_2$
$Y_6=A_0'.A_1.A_2$
$Y_7=A_0.A_1.A_2$



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

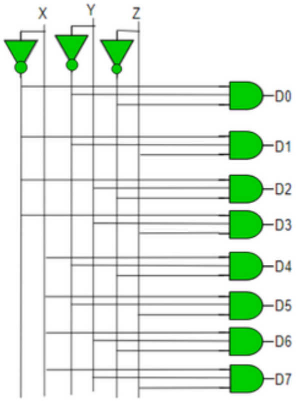
Implementation –
D0 is high when X = 0, Y = 0 and Z = 0. Hence,

```
D0 = X′ Y′ Z′
```

Similarly,

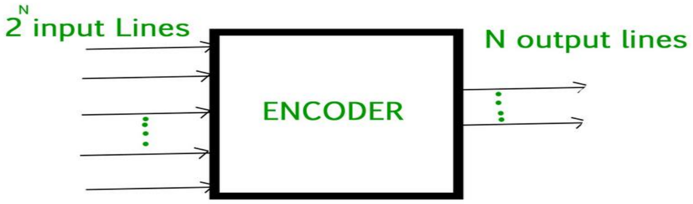
```
D1 = X' Y' Z
D2 = X' Y Z'
D3 = X' Y Z
D4 = X Y' Z'
D5 = X Y' Z
D6 = X Y Z'
D7 = X Y Z
```

Hence,

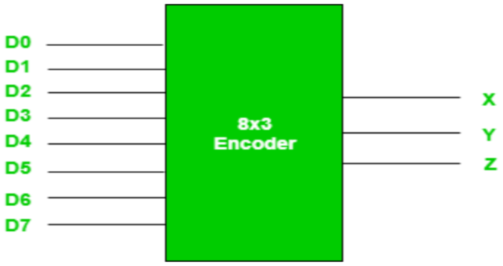


7. Encoder

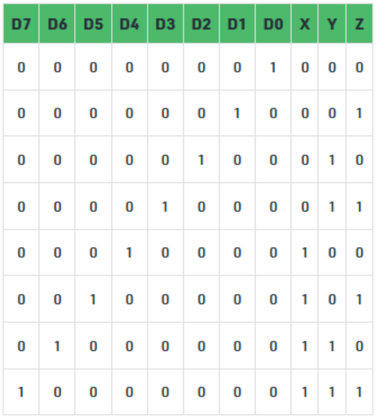
The encoder is a logic circuit that has 2n input lines of which only one is activated and n output lines which represent N bit code for the input. For simple encoders, it is assumed that only one input line is active at a time. It performs the opposite function of the decoder, i.e. at an active input (state 1), among the 2 inputs, a code on n lines must correspond at the output.



As an example, let’s consider Octal to Binary encoder. As shown in the following figure, an octal-to-binary encoder takes 8 input lines and generates 3 output lines.



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
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.

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

One limitation of this encoder is that only one input can be active at any given time. If more than one inputs are active, then the output is undefined. For example, if D6 and D3 are both active, then, our output would be 111 which is the output for D7. To overcome this, we use Priority Encoders.

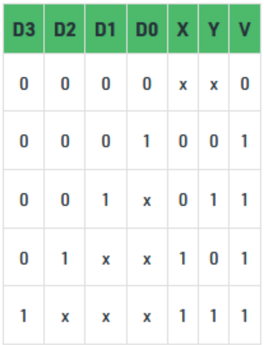
Another ambiguity arises when all inputs are 0. In this case, encoder outputs 000 which actually is the output for D0 active. In order to avoid this, an extra bit can be added to the output, called the valid bit which is 0 when all inputs are 0 and 1 otherwise.

7.1. Priority Encoder

A priority encoder is an encoder circuit in which inputs are given priorities. When more than one inputs are active at the same time, the input with higher priority takes precedence and the output corresponding to that is generated.

Let us consider the 4 to 2 priority encoder as an example. From the truth table, we see that when all inputs are 0, our V bit or the valid bit is zero and outputs are not used. The x's in the table show the don't care condition, i.e, it may either be 0 or 1. Here, D3 has highest priority, therefore, whatever be the other inputs, when D3 is high, output has to be 11. And D0 has the lowest priority, therefore the output would be 00 only when D0 is high and the other input lines are low. Similarly, D2 has higher priority over D1 and D0 but lower than D3 therefore the output would be 010 only when D2 is high and D3 are low (D0 & D1 are don't care).

Truth Table



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

V = D0 + D1 + D2 + D3

X = D2 + D3

Y = D1 D2' + D3

8. Converters (transcoders)

The transcoder is a combinatory logic circuit which converts a given binary code (input of n Bits) into another binary code (output of m Bits).

- Encoder is a converter with 2^n inputs and n outputs
- decoder is a converter with n inputs and 2^n outputs
- The transcoder is a combinational logic circuit of converting n inputs into m outputs

8.1. Decoder DCB

The BCD code (for binary coded decimal) is an encoding system used in electronics and computing to store integer numbers (decimal) by encoding their digits over 4-bit.

It is a decoder whose inputs are the four bits of the DCB code and it has ten outputs (only one is activated =1).

A	B	C	D		O_9	O_8	O_7	O_6	O_5	O_4	O_3	O_2	O_1	O_0
0	0	0	0											1
0	0	0	1										1	
0	0	1	0									1		
0	0	1	1								1			
0	1	0	0							1				
0	1	0	1						1					
0	1	1	0					1						
0	1	1	1				1							
1	0	0	0			1								
1	0	0	1		1									
1	0	1	0											
1	0	1	1											
1	1	0	0											
1	1	0	1											
1	1	1	0											
1	1	1	1											

$O_0 = \bar{A}\bar{B}\bar{C}\bar{D}$

$O_1 = \bar{A}\bar{B}\bar{C}D$

$O_2 = \bar{A}\bar{B}C\bar{D}$

$O_3 = \bar{A}\bar{B}CD$

$O_4 = \bar{A}B\bar{C}\bar{D}$

$O_5 = \bar{A}B\bar{C}D$

$O_6 = \bar{A}BC\bar{D}$

$O_7 = \bar{A}BCD$

$O_8 = A\bar{B}\bar{C}\bar{D}$

$O_9 = A\bar{B}\bar{C}D$

8.2. Encoder DCB

It is an encoder having 10 inputs representing the 10 digits of the decimal system of numeration, and 4 outputs intended to produce DCB codes corresponding to each input when it is activated.

O_9	O_8	O_7	O_6	O_5	O_4	O_3	O_2	O_1	O_0		A	B	C	D
									1		0	0	0	0
								1			0	0	0	1
							1				0	0	1	0
						1					0	0	1	1
					1						0	1	0	0
				1							0	1	0	1
			1								0	1	1	0
		1									0	1	1	1
	1										1	0	0	0
1											1	0	0	1

8.3. Converter Binary-Gray

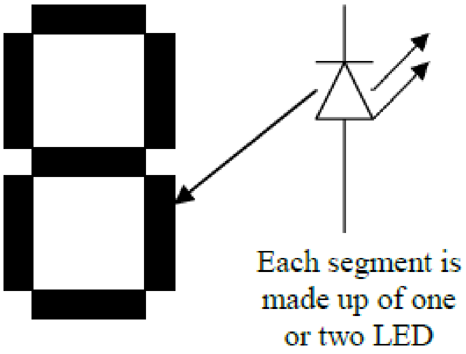
For instance, let us design binary to Gray converter. It is a circuit which receives at its inputs binary numbers and outputs corresponding Gray codes.

Number	Binary Code			Gray Code		
	B_2	B_1	B_0	G_2	G_1	G_0
0	0	0	0	0	0	0
1	0	0	1	0	0	1
2	0	1	0	0	1	1
3	0	1	1	0	1	0
4	1	0	0	1	1	0
5	1	0	1	1	1	1
6	1	1	0	1	0	1
7	1	1	1	1	0	0

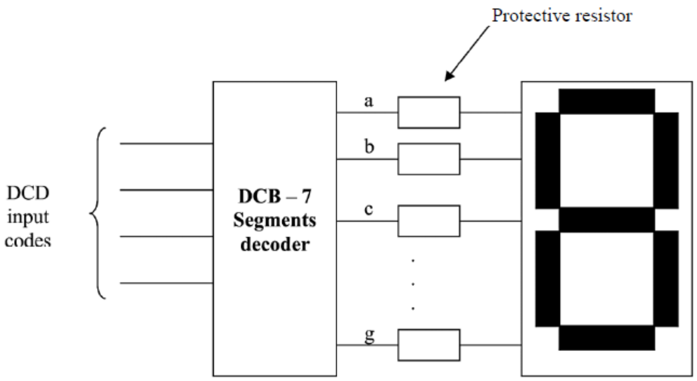
$G_0 = B_0 \oplus B_1$
$G_1 = B_1 \oplus B_2$
$G_2 = B_2$
$G_n = B_n \oplus B_{n+1}$

8.4. DCB – seven segments decoder

In many systems, seven segments displays are used to represent numbers from 0 to 9 and sometimes alphabetical characters.



The DCB - seven segments decoder accepts at its inputs a DCB code of four bits, and activate its outputs which will permit to enlighten the LEDs representing the corresponding cipher (corresponding to the DCB code).



Let us design a DCB – Seven segment segments decoder which will permit us to represent digital numbers from 0 to 9 on a 7 segments display.

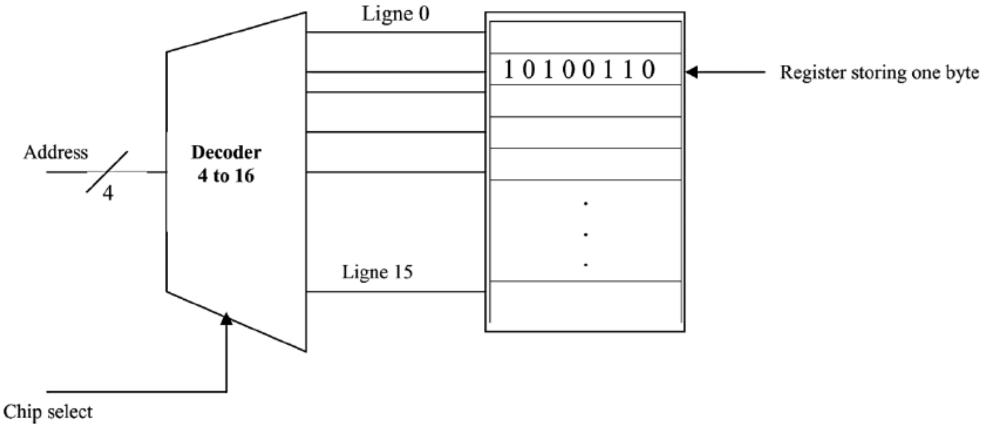
- Truth table:

Input				Output						
A	**B**	**C**	**D**	**a**	**b**	**c**	**d**	**e**	**f**	**g**
0	0	0	0	1	1	1	1	1	1	0
0	0	0	1	0	1	1	0	0	0	0
0	0	1	0	1	1	0	1	1	0	1
0	0	1	1	1	1	1	1	0	0	1
0	1	0	0	0	1	1	0	0	1	1

0	1	0	1	1	0	1	1	0	1	1
0	1	1	0	0	0	1	1	1	1	1
0	1	1	1	1	1	1	0	0	0	0
1	0	0	0	1	1	1	1	1	1	1
1	0	0	1	1	1	1	0	0	1	1

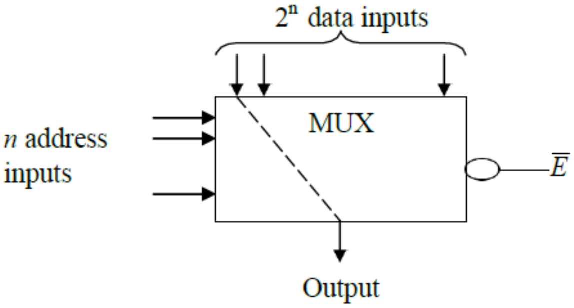
8.5. Applications of decoders and encoders

Addressing of a memory:



9. Multiplexer

A multiplexer also called data selector is a combinatory logic circuit which permits to direct towards single output information coming from many inputs. According to the address received by the multiplexer, only one among the information available at its inputs is selected and directed toward the output. A multiplexer can be considered as a commutator having multiple poles which are switched according to the address sent to the multiplexer.

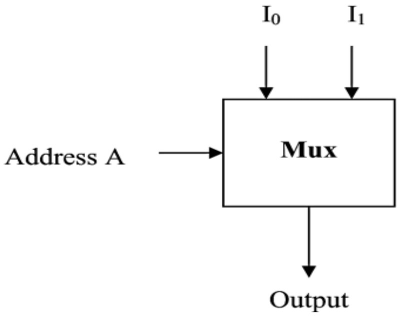


E is the enable input. It enables the multiplexer to function when the right logic signal level is sent to it (For the principle diagram above, the right signal level is slow). For an address bus having n lines, up to 2^n data can be addressed such that for each of these addresses, one among the 2^n data is selected and directed towards the output.

Decimal	Inputs						Output
	C_{n-1}		C_3	C_2	C_1	C_0	S
0	0		0	0	0	0	I_0
1	0		0	0	0	1	I_1
2	0		0	0	1	0	I_2
3	0		0	0	1	1	I_3
2^{n-1}	1		1	1	1	1	$I_{2^{n-1}}$

9.1. MUX 2 to 1

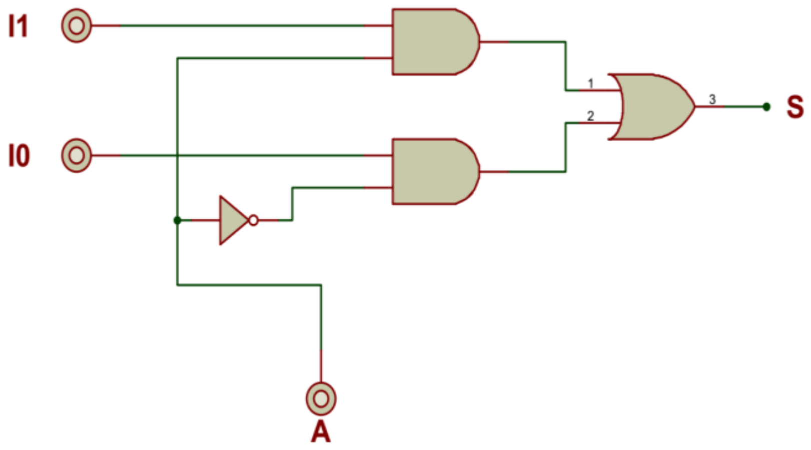
Let us design an elementary multiplexer. It is a multiplexer having two data inputs I_0 and I_1 and one address input A.



The address can be either 0 or 1. When the address is 0, the datum I_0 is selected and sent to the output. When the address is 1, I_1 is selected and directed towards the output.

A	S
0	I_0
1	I_1

$$S = I_0\bar{A} + I_1A$$

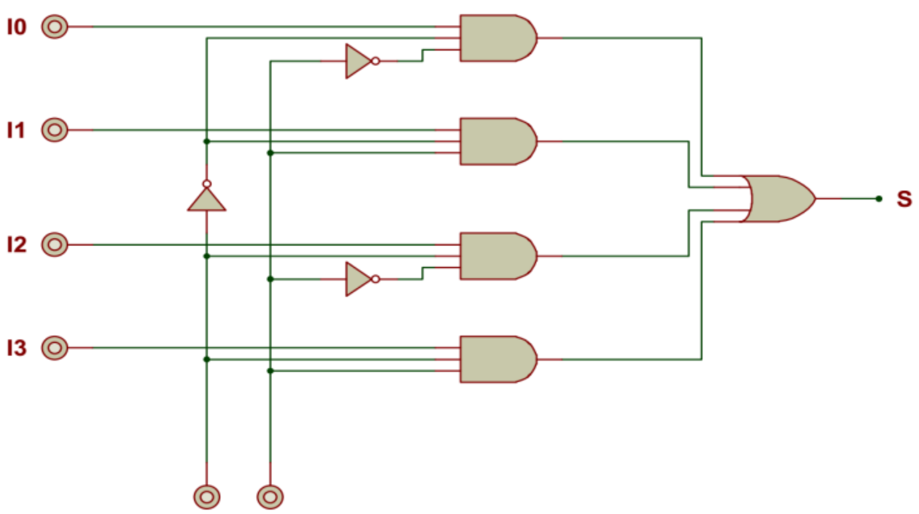


9.2. MUX 4 to 1

Let us design now a more complex multiplexer; it is a multiplexer having four data inputs. For 4 data, we need 2 address lines to address all the information.

C_1	C_0	S
0	0	I_0
0	1	I_1
1	0	I_2
1	1	I_3

$$\boldsymbol{S = I_0\overline{C_0}\,\overline{C_1} + I_1C_0\overline{C_1} + I_2\overline{C_0}C_1 + I_3C_0C_1}$$

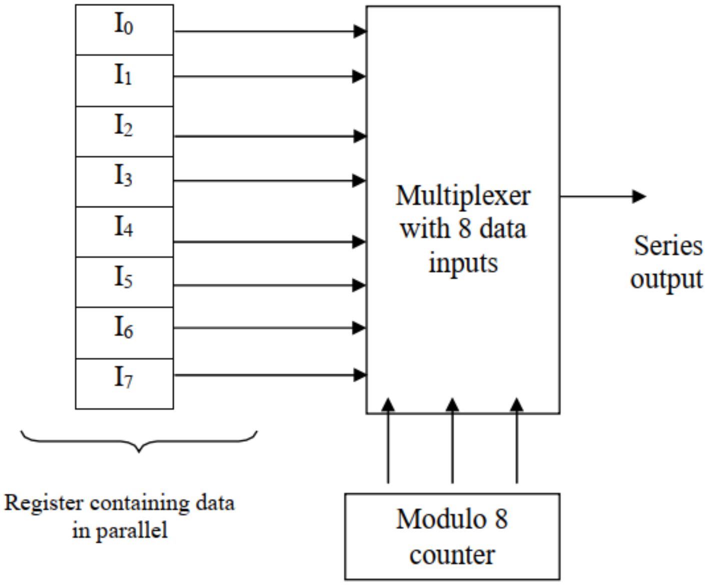


9.3. Applications of Multiplexer

The applications of multiplexers are found in many digital systems. These are some of those applications:

1- **Parallel to series conversion:**

In many digital systems, the treatment of information is done in parallel; when those information are to be transferred on a long distance, this cannot be done in parallel. In fact, parallel transfer of information is not effective because it requires a large number of lines through which data will flow; on the other hand, its causes a lot of errors in data transfer. The parallel to series conversion permits to make in such a way that the information treated in parallel can be transferred in series trough a single line. A parallel to series converter can be realized using a multiplexer as shown by the following figure:

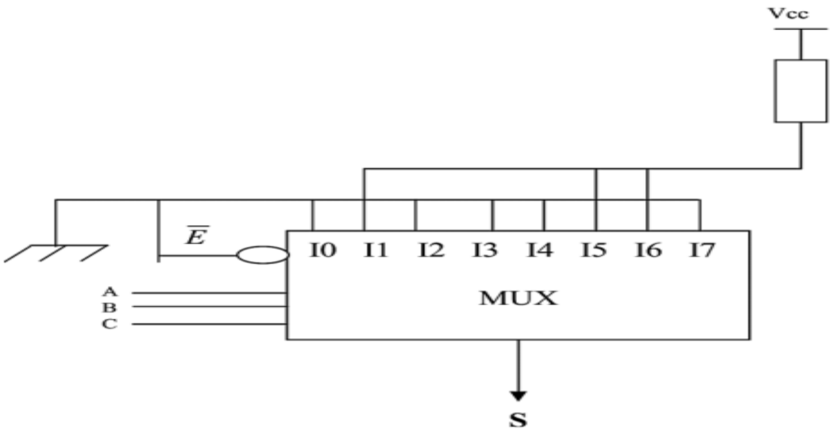


2- Realization of logic functions:

Logic functions can be realized using multiplexers. Let us consider a logic function described by the following truth table:

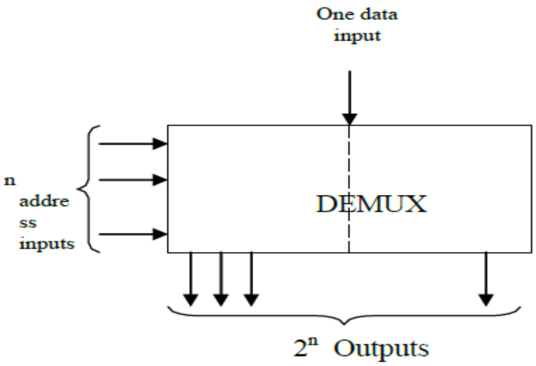
A	B	C	S	
0	0	0	0	I_0
0	0	1	1	I_1
0	1	0	0	I_2
0	1	1	0	I_3
1	0	0	0	I_4
1	0	1	1	I_5
1	1	0	1	I_6
1	1	1	0	I_7

This function can be realized without using logic gates. We can use a single multiplexer integrated circuit having 8 data inputs. The principle consists in connecting the outputs having low logic level to the earth and those having high logic level to the positive probe of the supply Vcc, as shown by the following figure:



10. The demultiplexer:

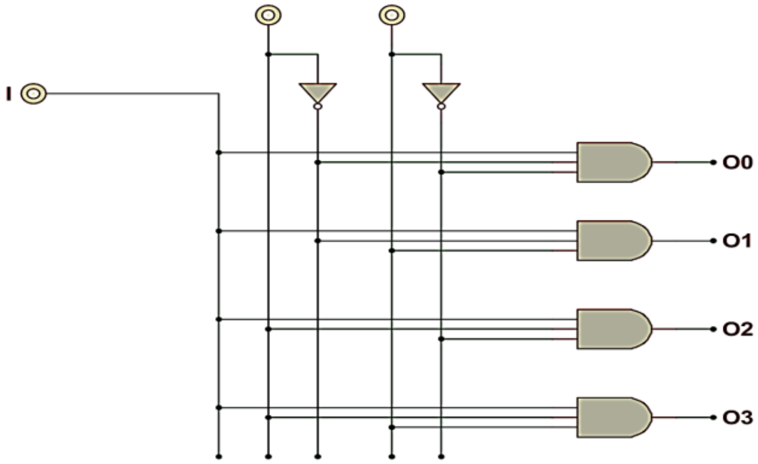
The demultiplexer is a combinatory logic circuit which permits to direct towards many outputs information coming from a single input. The demultiplexer has n address inputs and 2^n outputs such that, when an address is sent to it, the information is directed towards the corresponding output (the information is sent to only one among the 2^n available outputs).



10.1. DEMUX 1 to 4

Let us realize a multiplexer having four outputs (and therefore 2 address inputs).

C_1	C_0	O_3	O_2	O_1	O_0
0	0	0	0	0	I
0	1	0	0	I	00
1	0	0	I	0	0
1	1	I	0	0	


I
O0
O1
O2
O3