Basic Electronics Day 6

Decoders

Encoders

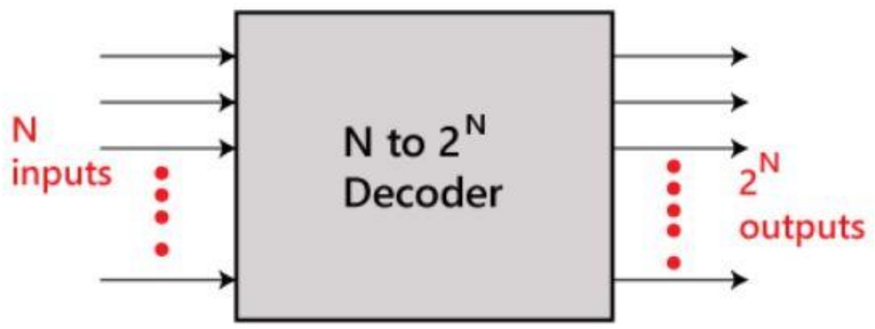
Comparators

Flip Flops

1st Year of 4 year B.Tech.

Decoders

- The combinational circuit that changes N 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
- At a time, only one input line is activated for simplicity. The produced 2^N-bit output code is equivalent to the binary information

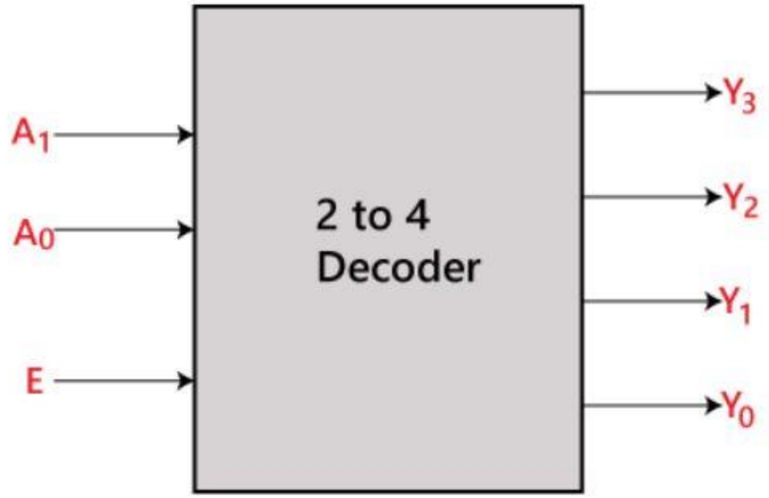


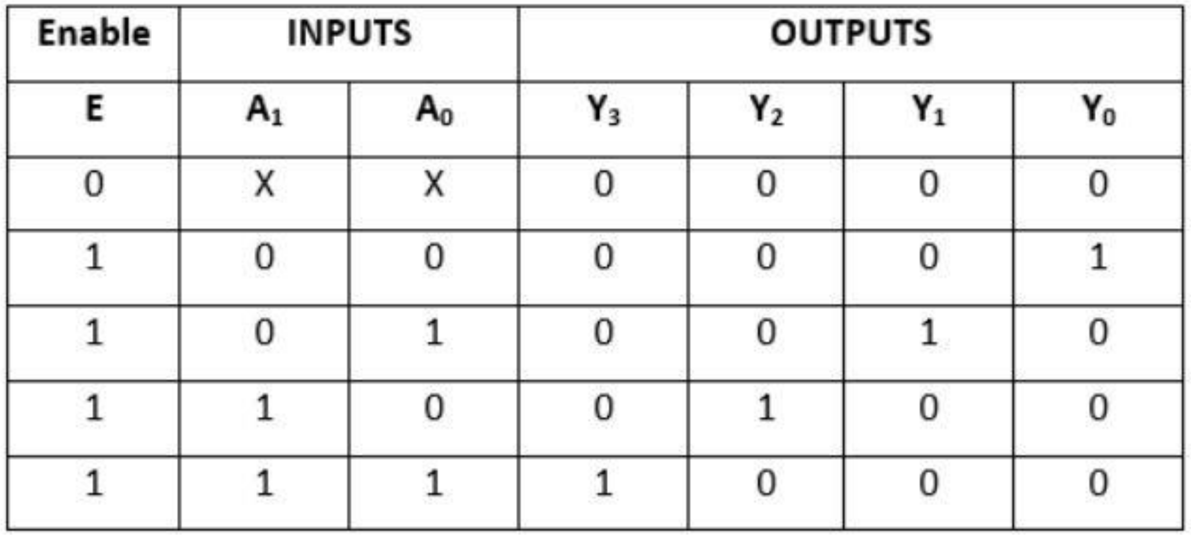
- Decoders are used in data multiplexing, digital display, digital to analog converters and memory addressing

2 to 4 line decoder:

- In the 2 to 4 line decoder, there is a total of two inputs, i.e., A_0, and A_1 and and enable input E
- 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.



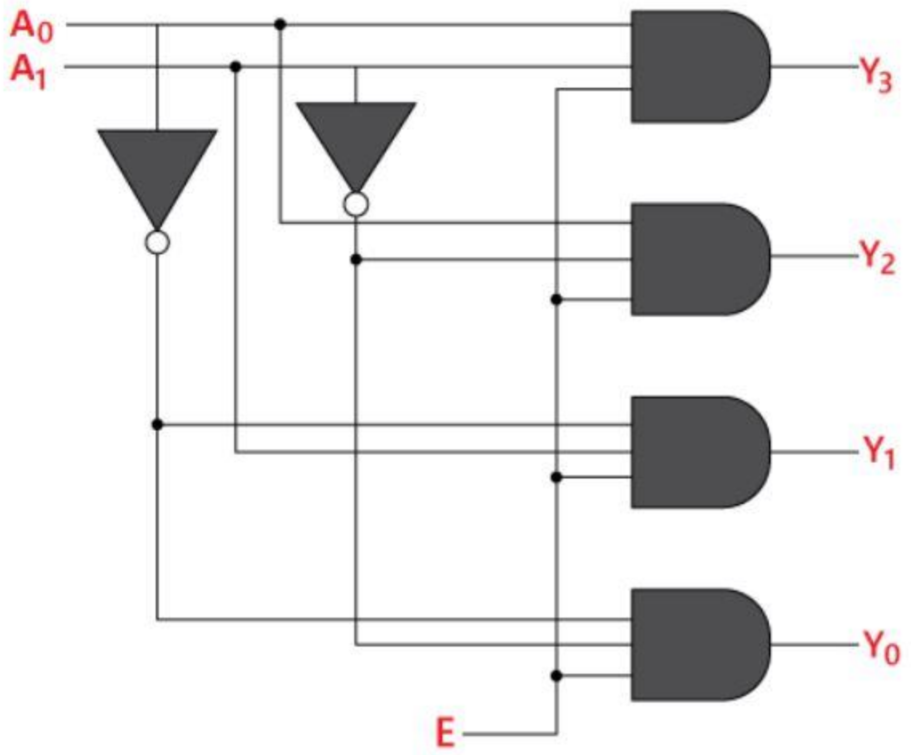


Enable	INPUTS		OUTPUTS			
E	A_1	A_0	Y_3	Y_2	Y_1	Y_0
0	X	X	0	0	0	0
1	0	0	0	0	0	1
1	0	1	0	0	1	0
1	1	0	0	1	0	0
1	1	1	1	0	0	0

Expressions for the outputs are as follows:

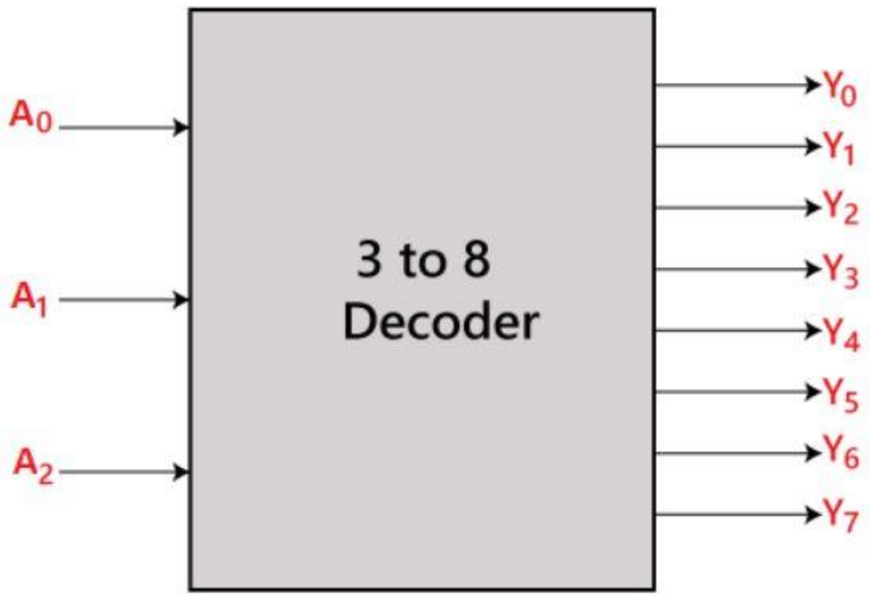
$Y_3 = E \cdot A_1 A_0$ $\quad Y_2 = E \cdot A_1 \overline{A}_0$ $\quad Y_1 = E \cdot \overline{A}_1 A_0$ $\quad Y_0 = E \cdot \overline{A}_1 \overline{A}_0$

Logical circuit of the above expressions is given below:



• 3 to 8 line decoder:

- The 3 to 8 line decoder is also known as **Binary to Octal 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 inputs, 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.



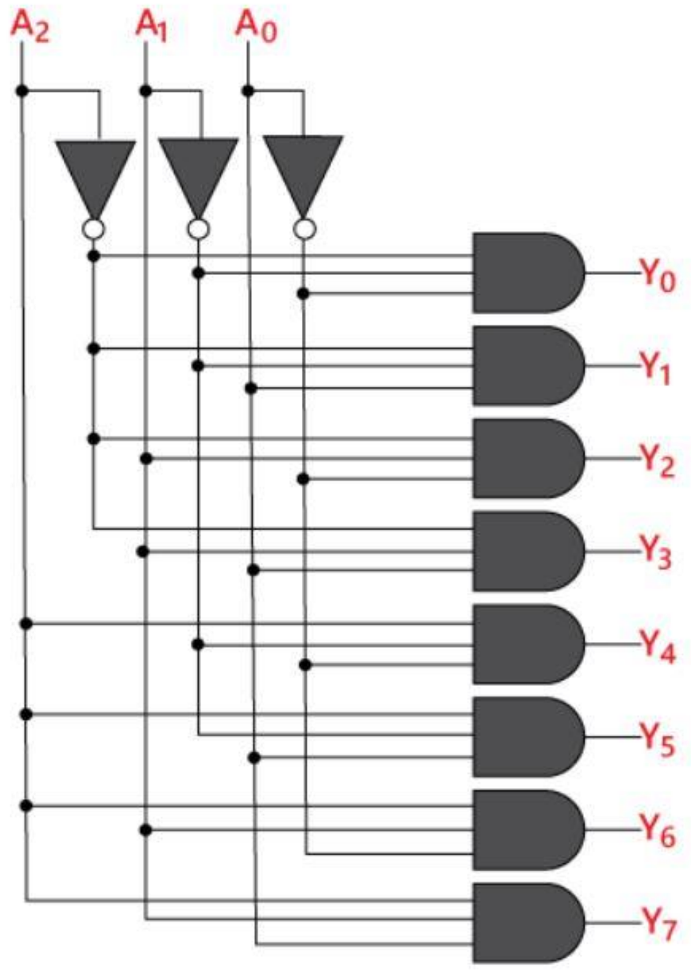
Enable	INPUTS			Outputs							
E	A_2	A_1	A_0	Y_7	Y_6	Y_5	Y_4	Y_3	Y_2	Y_1	Y_0
0	x	x	x	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	1
1	0	0	1	0	0	0	0	0	0	1	0
1	0	1	0	0	0	0	0	0	1	0	0
1	0	1	1	0	0	0	0	1	0	0	0
1	1	0	0	0	0	0	1	0	0	0	0
1	1	0	1	0	0	1	0	0	0	0	0
1	1	1	0	0	1	0	0	0	0	0	0
1	1	1	1	1	0	0	0	0	0	0	0

- The equations for the output lines are as below:

$Y_7 = E \cdot A_2 A_1 A_0$ $\quad$ $Y_6 = E \cdot A_2 A_1 \bar{A}_0$ $\quad$ $Y_5 = E . A_2 \overline{A_1} A_0$ $\quad$ $Y_4 = \mathrm{E}.\ A_2 \bar{A}_1 \bar{A}_0$

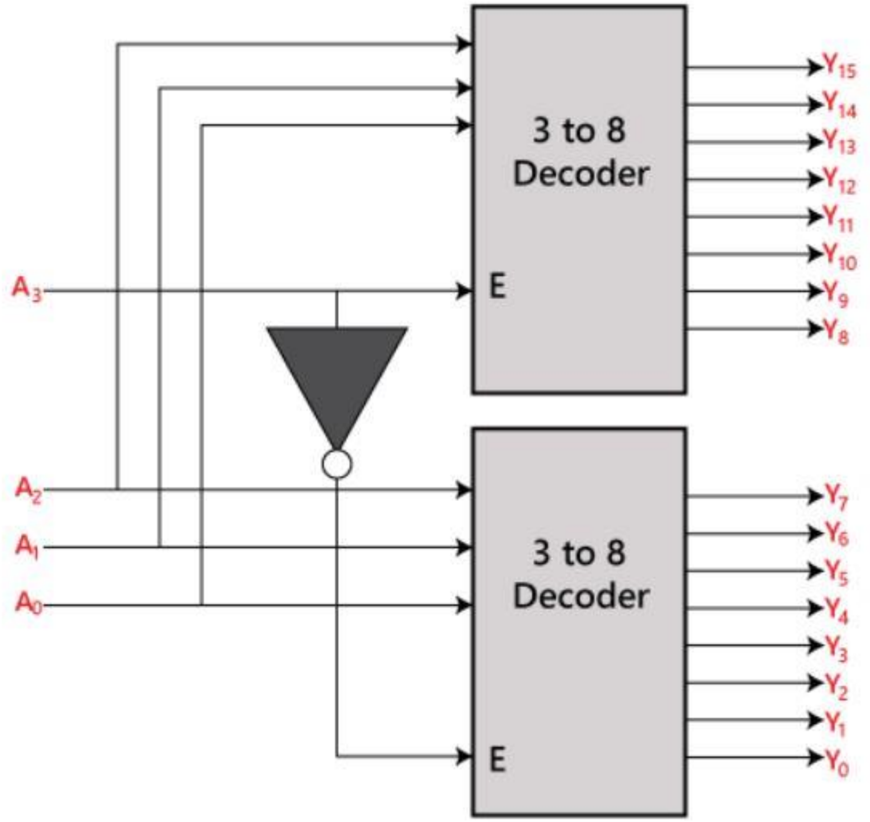
$Y_3 = \mathrm{E} . \bar{A}_2 A, A_0$ $\quad$ $Y_2 = \mathrm{E} . \bar{A}_2 A_1 \overline{A_0}$ $\quad$ $Y_1 = \mathrm{E}. \bar{A}_2 \bar{A}_1 A_0$ $\quad$ $Y_0 = \mathrm{E}. \bar{A}_2 \bar{A}_1 \bar{A}_0$

- Hence, the logical circuit diagram is:



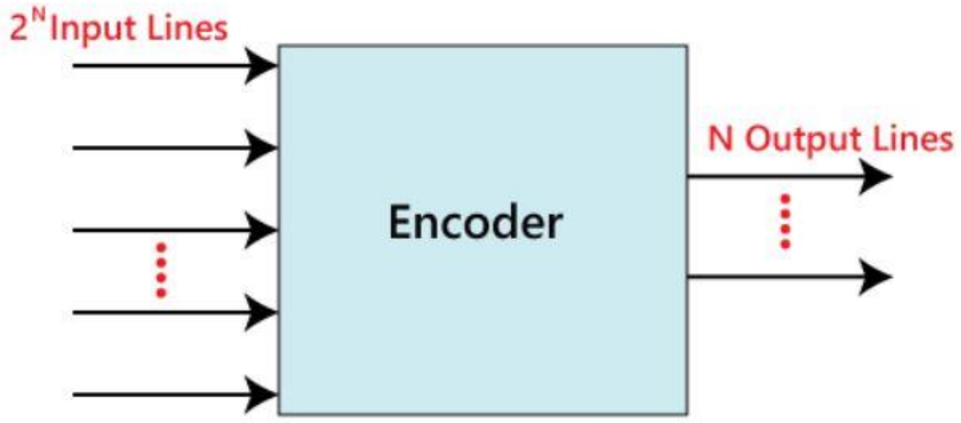
4 to 16 line Decoder

- In the 4 to 16 line decoder, there is a total of 16 outputs, i.e., Y_0, Y_1, Y_2,……, Y_{16} and four inputs, i.e., A_0, A1, A_2, and A_3.
- The 4 to 16 line decoder can be constructed using either 2 to 4 decoder or 3 to 8 decoder.
- Required number of lower order decoders=m_2/m_1
- $m_1 = 8$ $m_2 = 16$
- Required number of 3 to 8 decoders=16/8=2



Encoders:

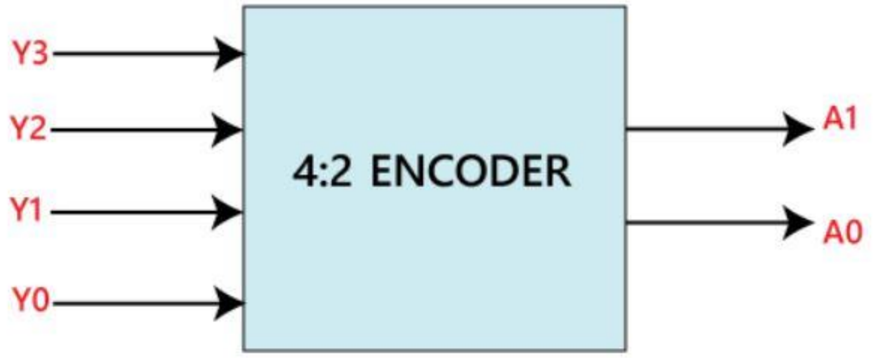
- The combinational circuits that change the 2^N binary information into N output lines are known as Encoders.
- The binary information is passed in the form of 2N input lines
- The output lines define the N-bit code for the binary information
- In simple words, the Encoder performs the reverse operation of the Decoder



4 to 2 line Encoder:

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

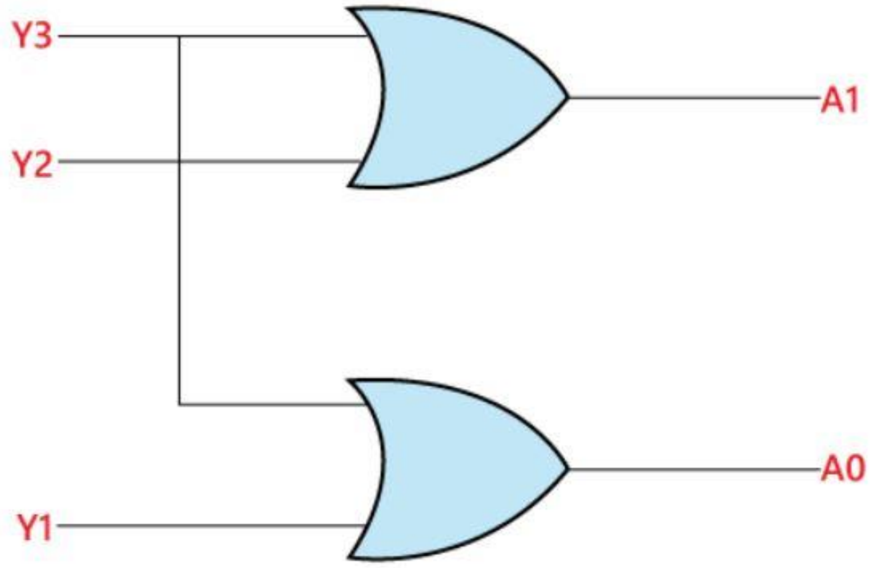


Inputs				Outputs	
Y_3	Y_2	Y_1	Y_0	A_1	A_0
0	0	0	1	0	0
0	0	1	0	0	1
0	1	0	0	1	0
1	0	0	0	1	1

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

$$A1=Y3+Y2 \qquad A0=Y3+Y1$$

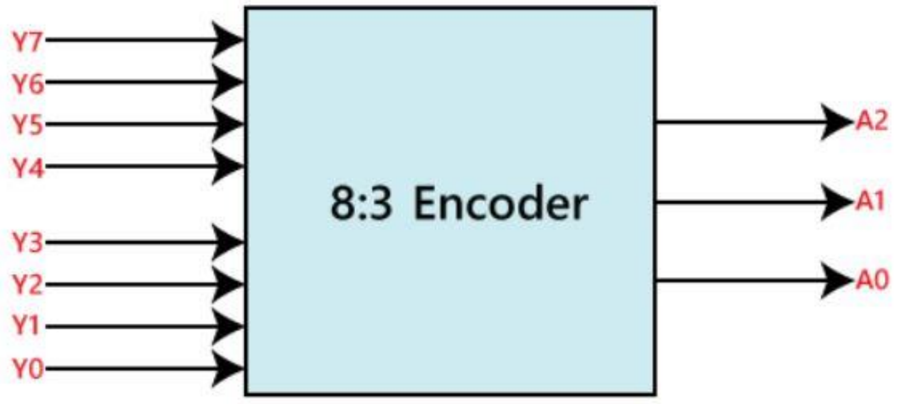
- Logical circuit of the above expressions is given below:



8 to 3 line Encoder:

- 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**.**



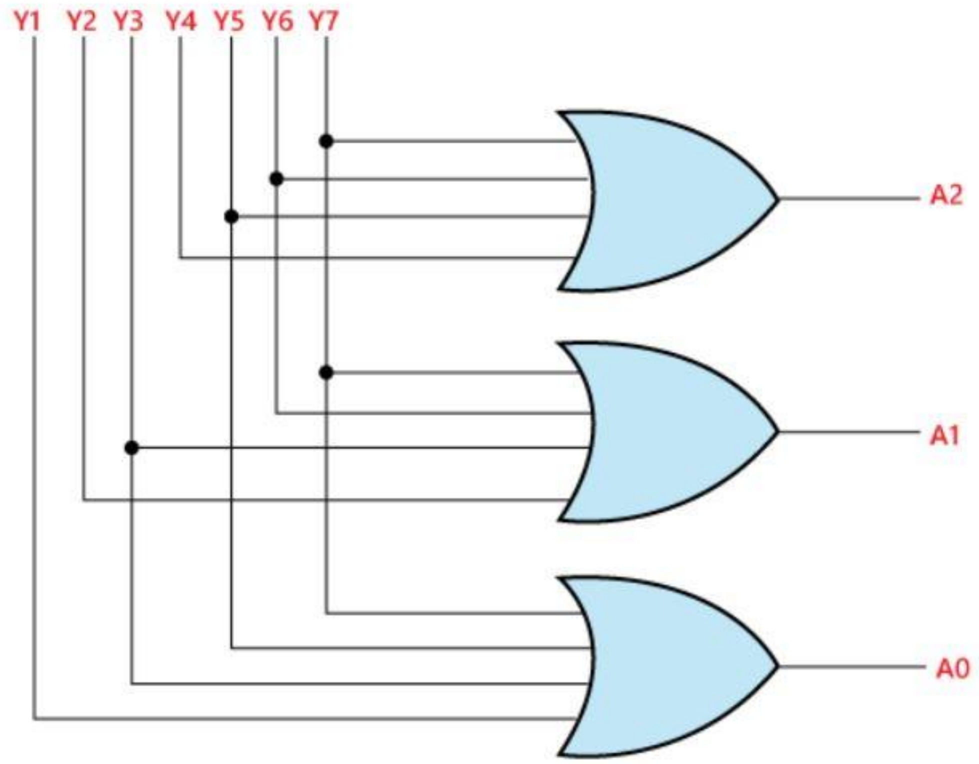
INPUTS								OUTPUTS		
Y_7	Y_6	Y_5	Y_4	Y_3	Y_2	Y_1	Y_0	A_2	A_1	A_0
0	0	0	0	0	0	0	1	0	0	0
0	0	0	0	0	0	1	0	0	0	1
0	0	0	0	0	1	0	0	0	1	0
0	0	0	0	1	0	0	0	0	1	1
0	0	0	1	0	0	0	0	1	0	0
0	0	1	0	0	0	0	0	1	0	1
0	1	0	0	0	0	0	0	1	1	0
1	0	0	0	0	0	0	0	1	1	1

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

$A_2=Y_4+Y_5+Y_6+Y_7$

$A_1=Y_2+Y_3+Y_6+Y_7$

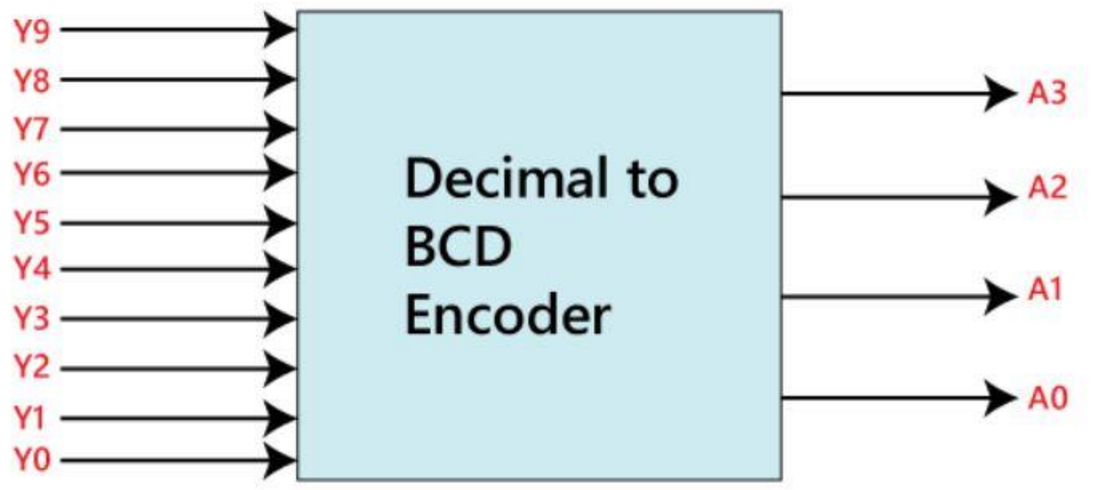
$A_0=Y_7+Y_5+Y_3+Y_1$



Decimal to BCD Encoder

- The Octal to Binary Encoder is also known as **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 corresponding to the 10 decimal digits (0-9) and four outputs, i.e., A_0, A1, A_2, and A_3 representing the BCD value
- 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.



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 for the circuit is

$$A3 = Y9 + Y8$$

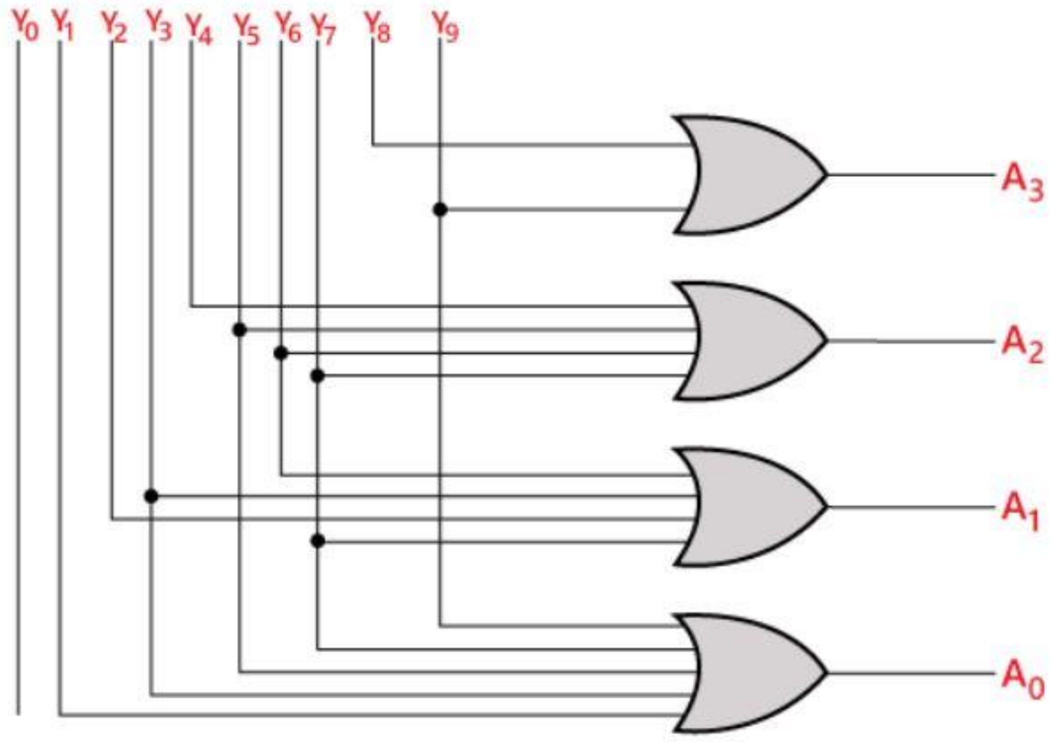
$$A2 = Y7 + Y6 + Y5 + Y4$$

$$A1 = Y7 + Y6 + Y3 + Y2$$

$$A0 = Y9 + Y7 + Y5 + Y3 + Y1$$

Hence, we can see that A3 is HIGH whenever 8 or 9 is HIGH

Similarly, A2 is HIGH if one of the inputs 7,6,5 or 4 is HIGH

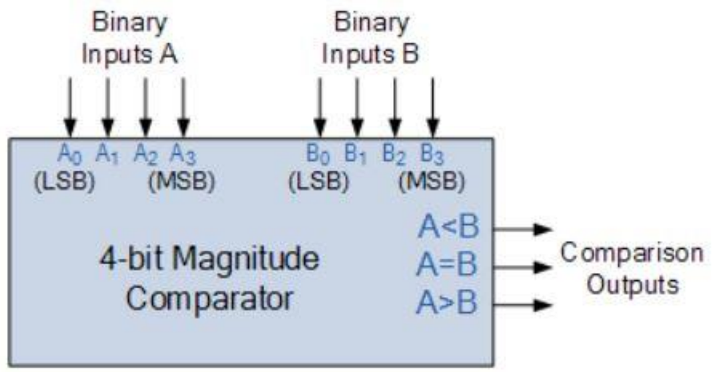


4 BIT MAGNITUDE COMPARATOR

- It compares two 4-bit numbers A and B and gives one of the following outputs
  - $A = B$,
  - $A < B$
  - $A > B$

Considering two 4 bit numbers $A_3A_2A_1A_0$ and $B_3B_2B_1B_0$

- If A3 = 1 and B3 = 0, then A is greater than B (A>B)
- If A3 and B3 are equal, and if A2 = 1 and B2 = 0, then A > B
- If A3 and B3 are equal & A2 and B2 are equal, and if A1 = 1, and B1 = 0, then A>B
- If A3 and B3 are equal, A2 and B2 are equal and A1 and B1 are equal, and if A0 = 1 and B0 = 0, then A > B



We will compare each bit of the two 4-bit numbers, and based on that comparison and the weight of their positions, we will draft a truth table.

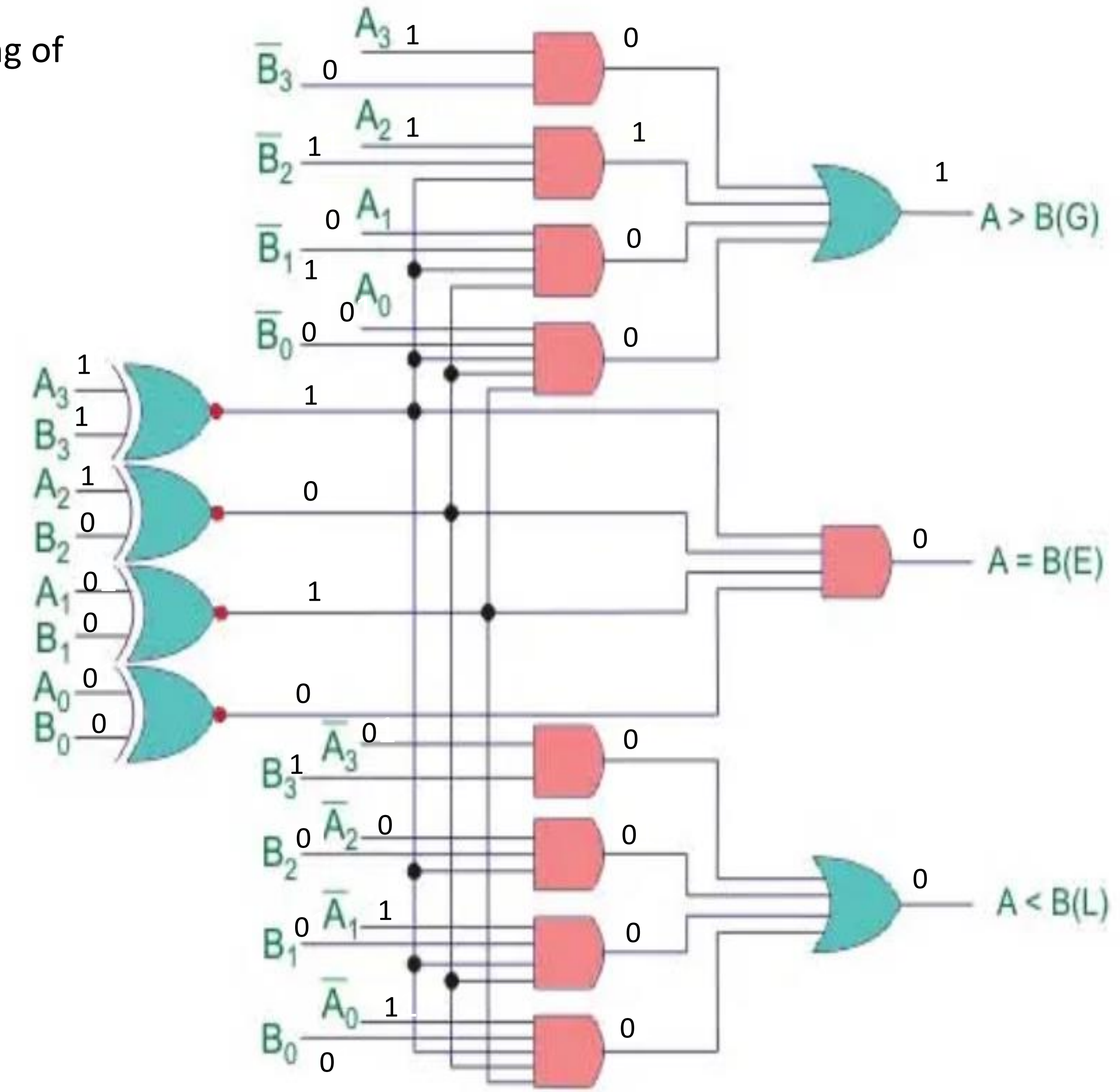
A3B3	**A2B2**	**A1B1**	**A0B0**	**A>B**	**A<B**	**A=B**
A3>B3	x	x	x	1	0	0
A3<B3	x	x	x	0	1	0
A3=B3	A2>B2	x	x	1	0	0
A3=B3	A2<B2	x	x	0	1	0
A3=B3	A2=B2	A1>B1	x	1	0	0
A3=B3	A2=B2	A1<B1	x	0	1	0
A3=B3	A2=B2	A1=B1	A0>B0	1	0	0
A3=B3	A2=B2	A1=B1	A0<B0	0	1	0
A3=B3	A2=B2	A1=B1	A0=B0	0	0	1

Let us compare two numbers to understand the working of this circuit

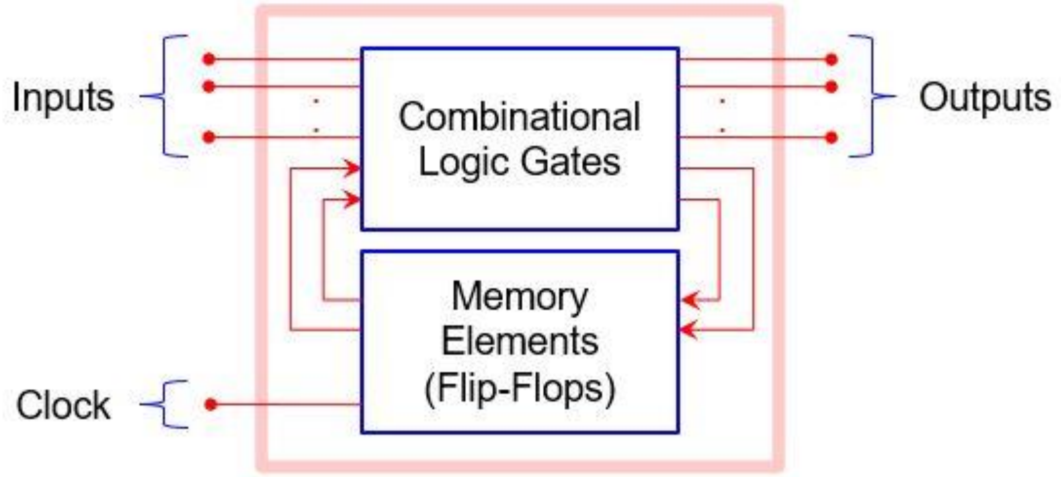
Let the two numbers A and B be

$A_3A_2A_1A_0$= 1 1 0 0

$B_3B_2B_1B_0$= 1 0 0 0



Sequential Logic & The Flip-Flop



Latches and Flip Flops

- Both **latches and flip-flops** are circuit elements whose output depends not only on the current inputs, but also on previous inputs and outputs.
- The **difference between** a **latch** and a **flip-flop** is that a **latch** does not have a clock signal, whereas a **flip-flop** always does
- Flip-flop is a basic memory element of a sequential circuit, which stores one bit of information
- Flip flops are the fundamental blocks of most sequential circuits
- The state of flip-flop changes at active state of clock pulses and remains unaffected when the clock pulse is not active
- Since a flip-flop stores a binary digit it must, by definition, have 2 states. Furthermore it is bistable, which means it is stable in each state: when is put in a specific state, it will stay in that state until something (clock pulse) causes it to change to the other state
- The term "Flip-flop" relates to the actual operation of the device, as it can be "flipped" into one logic Set state or "flopped" back into the opposing logic Reset state.

Types of Flip Flop:

- S-R Flip Flop

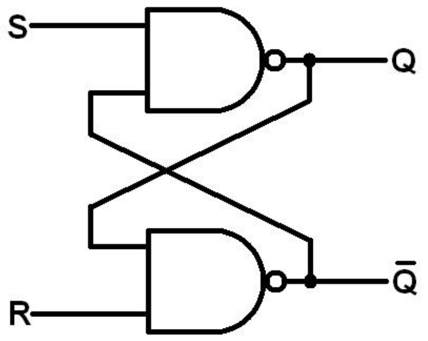
This Flip-Flop has two inputs that change its state: **R**eset and **S**et.

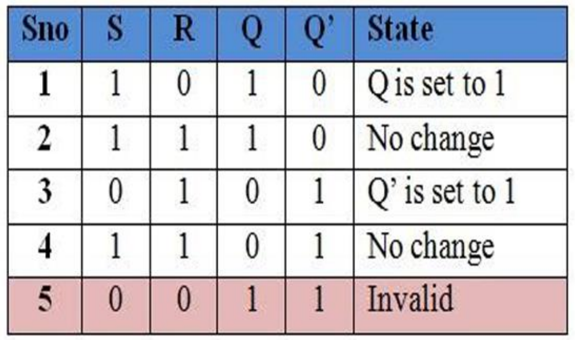
- The simplest way to make any basic single bit set-reset SR flip-flop is to connect together a pair of cross-coupled 2-input NAND gates as shown, to form a Set-Reset Bistable also known as an active LOW SR NAND Gate Latch, so that there is feedback from each output to one of the other NAND gate inputs.

- This device consists of two inputs, one called the Set, S and the other called the Reset, R with two corresponding outputs Q and its inverse or complement Q (not-Q)

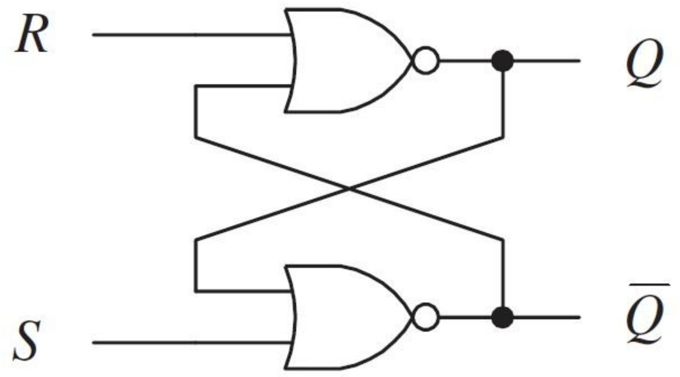
- When R goes low, Q goes low and Q' goes high
  When S goes low, Q goes high and Q' goes low

- When both R and S are high the Flip-Flop is stable and doesn't change

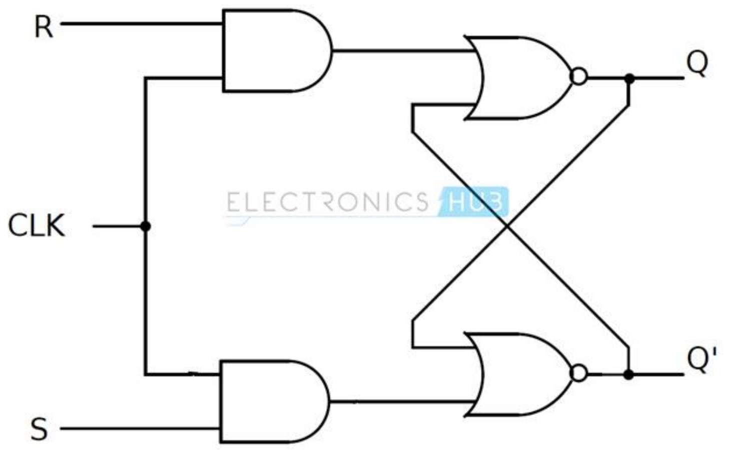




Sno	S	R	Q	Q'	State
1	1	0	1	0	Q is set to 1
2	1	1	1	0	No change
3	0	1	0	1	Q' is set to 1
4	1	1	0	1	No change
5	0	0	1	1	Invalid



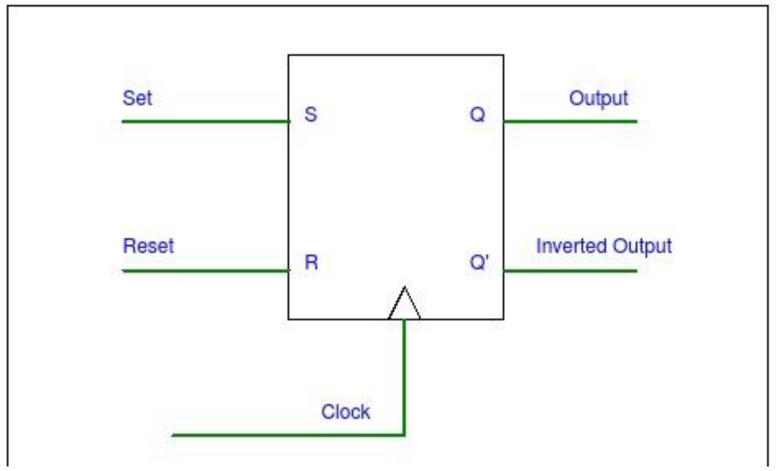
SR LATCH



AND based SR flip flop

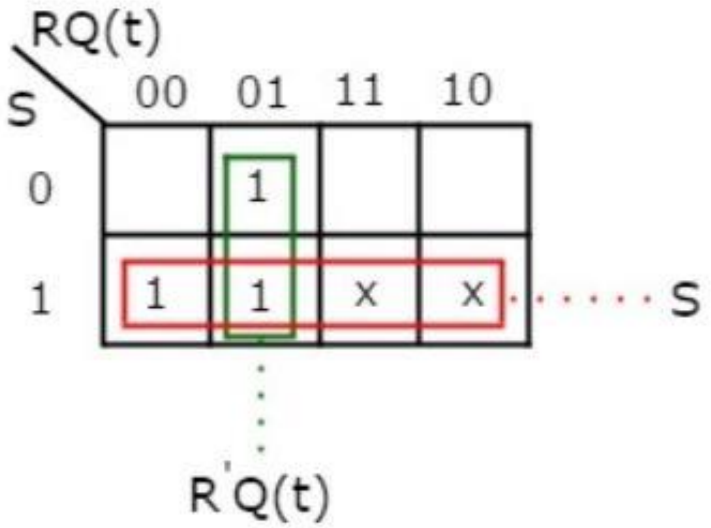
- In this circuit when the clock input is LOW, the output of both the AND gates are LOW and changes in S input or R input will not affect the output Q of the flip flop

- When clock becomes HIGH, value of S and R inputs will be passed on to output of the AND gates and output Q of the flip flop will change according to changes in S and R inputs



S	R	Q $t+1$
0	0	Q t
0	1	0
1	0	1
1	1	-

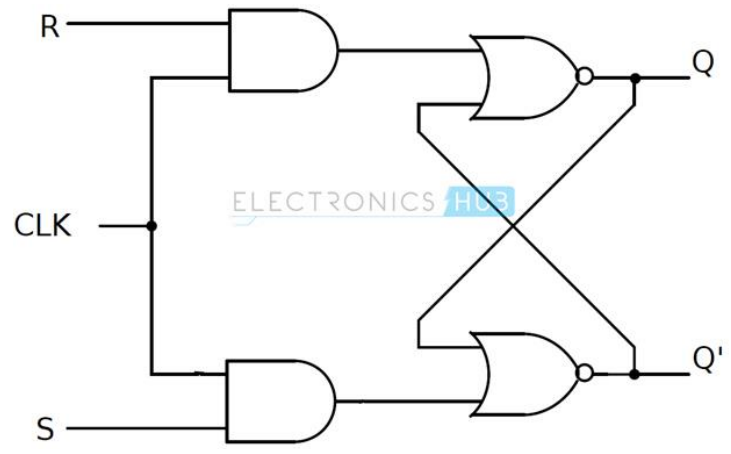
By using three variable K-Map, we can get the simplified expression for next state, Q $t+1$. The **three variable K-Map** for next state, Q $t+1$ is shown in the following figure.



The maximum possible groupings of adjacent ones are already shown in the figure. Therefore, the **simplified expression** for next state Q $t+1$ is

$$Q\left(t+1\right) = S + R'Q\left(t\right)$$

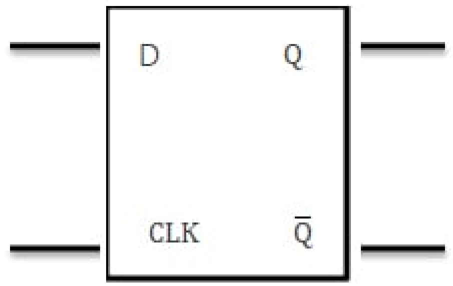
- For S=0 and R=0 and CLK=0, the flip flop simply remains in the previous state Q remains unchanged
- Whenever, CLK=0, the flip flop remains in its previous state
- For S=0, R=0 and CLK=1, the flip flop still remains in its previous state and both the inputs are zero
- For S=0 and R=1 and CLK=1, the output of And gate 1 is 1 and output of AND gate 2 is 0, since S=0, Q' is forced to become 1 and hence, output of NOR gate 1 is 0. So, Q = 0, Q' = 1
- For S=1, R=0 and CLK=1, the output of AND gate 2 is 1. R=0 forces the Q output to go to state 0 and hence output of Q' become 1. So, Q=1 and Q' = 0

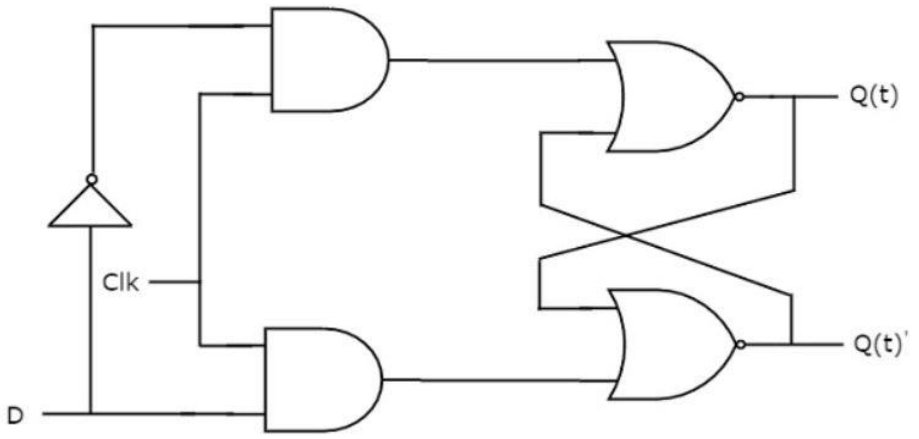


- An intermediate condition occurs when all the inputs S=1, R=1 and CLK=1. This condition results in 1's in both the outputs Q and Q' , which is not desired

D Flip Flop:

- The D (delay) flip flop has only one input called the Delay input and two outputs Q and $\overline{Q}$
- It can be constructed from SR flip flop by inserting an inverter between S and R
- When CLK input is LOW, the D input has no effect on the output
- When CLK goes HIGH, the Q output will take on the value of the D input.





Qn	Dn	Qn+1
0	0	0
0	1	1
1	0	0
1	1	1

State Table

Dn	Qn+1
0	0
1	1

Characteristic Table

Qn \ Dn	0	1
0	0	1
1	0	1

K-Map

$$\boxed{Q_{n+1} = D_n}$$

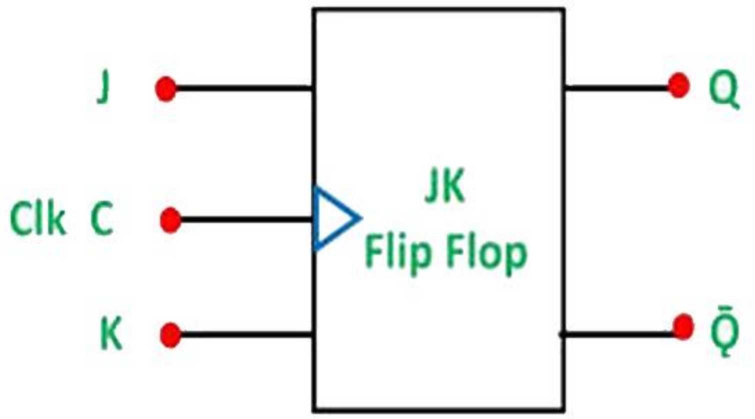
Characteristic Equation

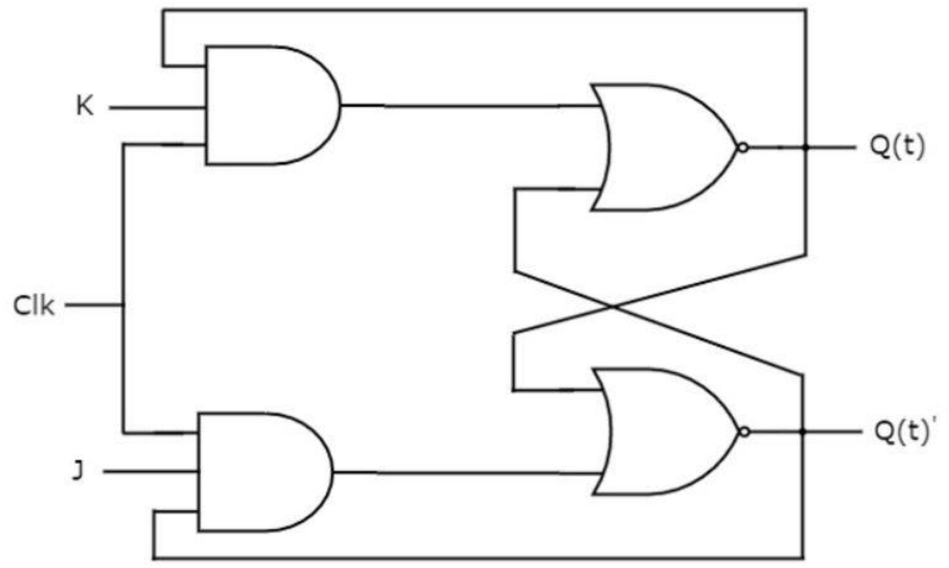
Qn	Qn+1	Dn
0	0	0
0	1	1
1	0	0
1	1	1

Excitation Table

J-K FLIP FLOP:

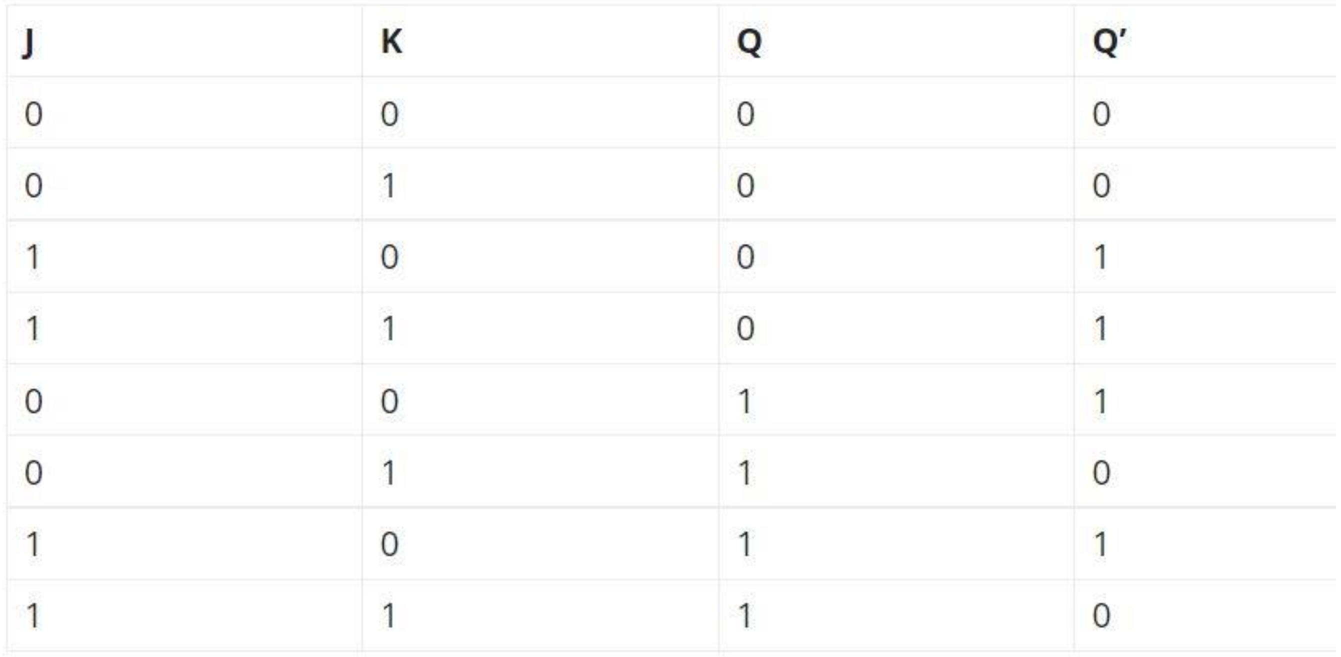
- JK flip-flop is the modified version of SR flip-flop. It operates with only positive clock transitions
- The intermediate condition that was not permitted in SR flip flop is permitted in JK flip flop
- The JK flip flop behaves similar to SR flip flop.
- When J=K=1, the flip flop output toggles, i.e., switches to its complement states. If Q=0, it switches to 1 and if Q=1 it switches to 0





The truth table of a JK flip flop is as given below-

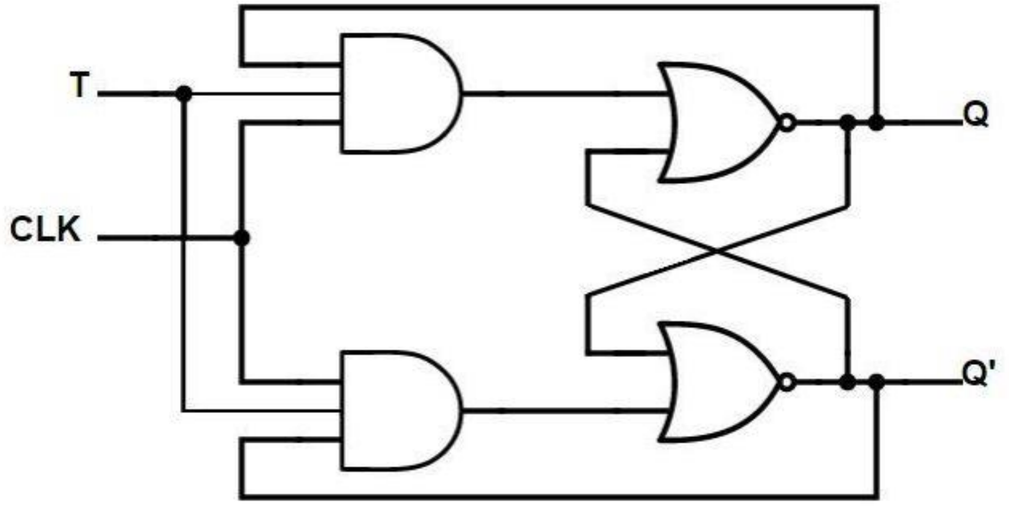
- If J and K data input are different (i.e. high and low) then the output Q takes the value of J at the next clock edge
- If J and K are both low then no change occurs
- If J and K are both high at the clock edge then the output will toggle from one state to the other



J	K	Q	Q’
0	0	0	0
0	1	0	0
1	0	0	1
1	1	0	1
0	0	1	1
0	1	1	0
1	0	1	1
1	1	1	0

T Flip Flop

- A T flip-flop is like a JK flip-flop. These are basically a single input version of JK flip-flops
- This modified form of JK flip-flop is obtained by connecting both inputs J and K together. This flip-flop has only one input along with the clock input
- These flip-flops are called T flip-flops because of their ability to complement its state (i.e.) Toggle, hence the name Toggle flip-flop.



T	Q	Q (t+1)
0	0	0
1	0	1 ⬅
0	1	1
1	1	0 ⬅

Toggling states with T