Basic Logic Gates

Prepared By:
Karuna Patel

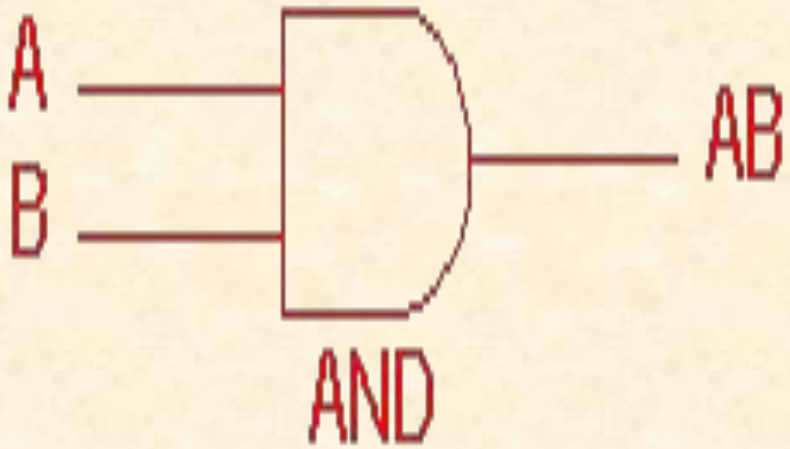
Basic Logic Gates and Basic Digital Design

All digital systems can be constructed by only three basic logic gates. These basic gates are called the AND gate, the OR gate, and the NOT gate. Some of the author also include the NAND gate, the NOR gate and the EOR gate as the members of the family of basic logic gates.

- **NOT, AND, and OR Gates**
- NAND and NOR Gates
- Exclusive-OR (EOR) Gate

AND gate

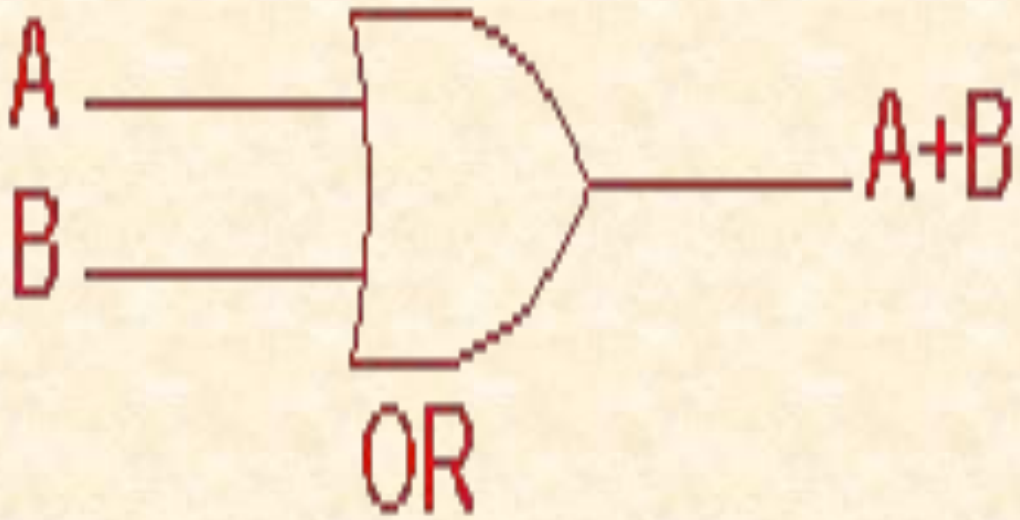
The AND gate is a circuit which gives a high output (logic 1) if all its inputs are high. A dot (.) is used to indicate the AND operation. In practice, however, the dot is usually omitted.



2 Input AND gate		
A	B	A.B
0	0	0
0	1	0
1	0	0
1	1	1

OR gate

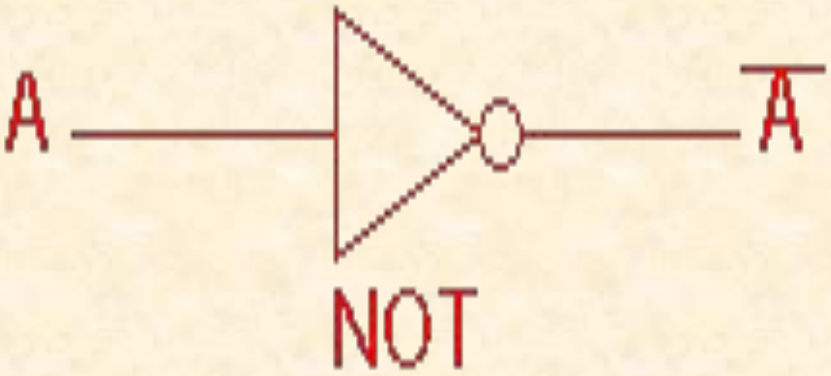
The OR gate is a circuit which gives a high output if one or more of its inputs are high. A plus sign (+) is used to indicate the OR operation.



2 Input OR gate		
A	B	A+B
0	0	0
0	1	1
1	0	1
1	1	1

NOT gate

The NOT gate is a circuit which produces at its output the negated (inverted) version of its input logic. The circuit is also known as an inverter. If the input variable is A, the inverted output is written as .

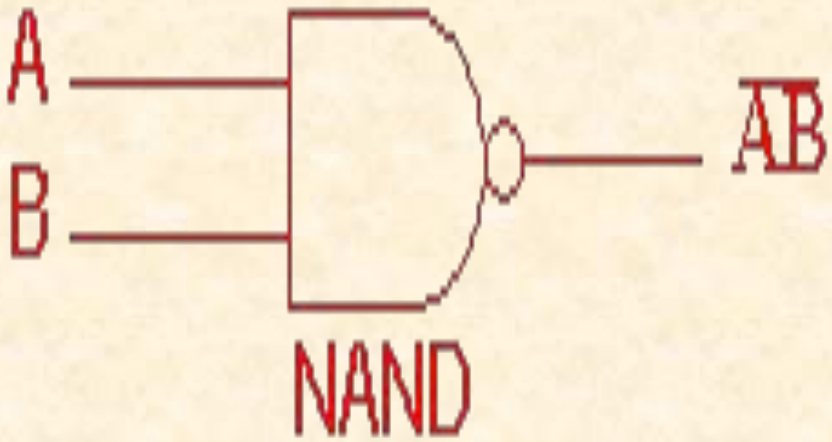


NOT gate	
A	$\overline{A}$
0	1
1	0

NAND gate

The NAND gate is a NOT-AND circuit which is equivalent to an AND circuit followed by a NOT circuit.

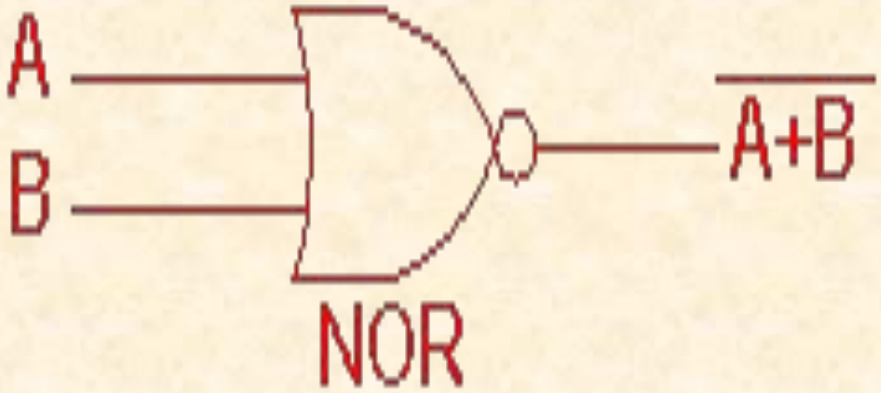
The output of the NAND gate is high if any of its inputs is low.



2 Input NAND gate		
A	B	$\overline{A.B}$
0	0	1
0	1	1
1	0	1
1	1	0

NOR Gate

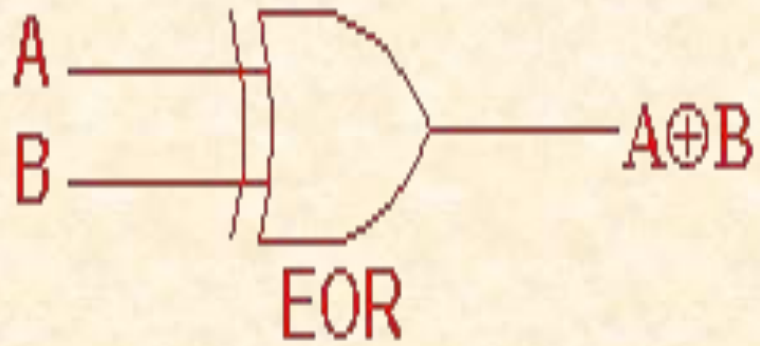
The NOR gate is a NOT-OR circuit which is equivalent to an OR circuit followed by a NOT circuit. The output of the NOR gate is low if any of its inputs is high.



2 Input NOR gate		
A	B	$\overline{A+B}$
0	0	1
0	1	0
1	0	0
1	1	0

EOR Gate

The Exclusive-OR gate is a circuit which gives a high output if either of its two inputs is high, but not both. A encircled plus sign () is used to indicate the EOR operation.



2 Input EXOR gate		
A	B	A⊕B
0	0	0
0	1	1
1	0	1
1	1	0

Combinational Circuits

Combinational circuit is a circuit in which we combine the different gates in the circuit, for example encoder, decoder, multiplexer and demultiplexer.

Some of the characteristics of combinational circuits are following −

- The output of combinational circuit at any instant of time, depends only on the levels present at input terminals.
- The combinational circuit do not use any memory. The previous state of input does not have any effect on the present state of the circuit.
- A combinational circuit can have an n number of inputs and m number of outputs.

Figure: Combinational Circuits

- In this output depends only upon present input.
- Speed is fast.
- It is designed easy.
- There is no feedback between input and output.
- This is time independent.
- Elementary building blocks: Logic gates
- Used for arithmetic as well as boolean operations.
- Combinational circuits don’t have capability to store any state.
- As combinational circuits don’t have clock, they don’t require triggering.
- These circuits do not have any memory element.
- It is easy to use and handle.

Sequential Circuits

Sequential circuit has memory so output can vary based on input. This type of circuits uses previous input, output, clock and a memory element.

When combinational circuit does not use any memory. Hence the previous state of input does not have any effect on the present state of the circuit.

Generally, there are two types of storage elements used: **Latches**, and **Flip-Flops**. Storage elements that operate with signal levels (rather than signal transitions) are referred to as latches ; those controlled by a clock transition are flip-flops.

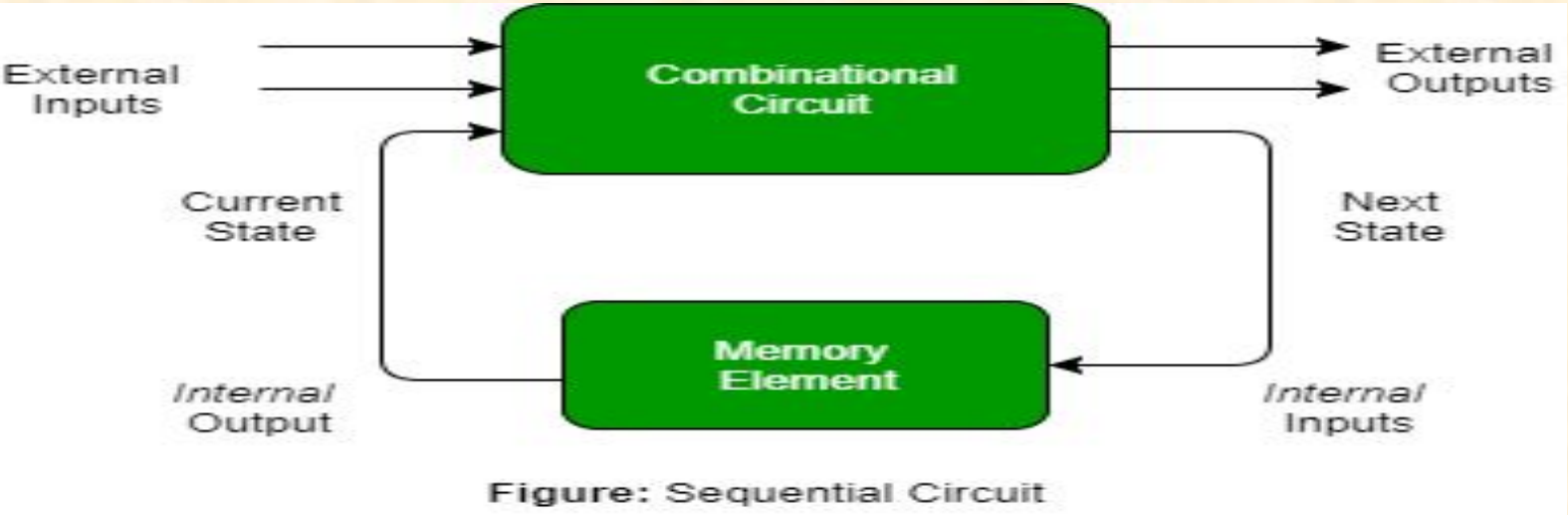


Figure: Sequential Circuit

- In this output depends upon present as well as past input.
- Speed is slow.
- It is designed tough as compared to combinational circuits.
- There exists a feedback path between input and output.
- This is time dependent.
- Elementary building blocks: Flip-flops
- Mainly used for storing data.
- Sequential circuits have capability to store any state or to retain earlier state.
- As sequential circuits are clock dependent they need triggering.
- These circuits have memory element.
- It is not easy to use and handle.

Sr. No.	Key	Combinational Circuit	Sequential Circuit
1	Definition	Combinational Circuit is the type of circuit in which output is independent of time and only relies on the input present at that particular instant.	Sequential circuit is the type of circuit where output not only relies on the current input but also depends on the previous output.
2	Feedback	In Combinational circuit as output does not depend on the time instant, no feedback is required for its next output generation.	Sequential circuit output relies on its previous feedback so output of previous input is being transferred as feedback used with input for next output generation.
3	Performance	As the input of current instant is only required in case of Combinational circuit, it is faster and better in performance as compared to that of Sequential circuit.	On other hand Sequential circuit are comparatively slower and has low performance as compared to that of Combinational circuit.
4	Complexity	No implementation of feedback makes the combinational circuit less complex as compared to sequential circuit.	However implementation of feedback makes sequential circuit more complex as compared to combinational circuit.
5	Elementary	Elementary building blocks for	Building blocks for sequential circuit

Adder

An **Adder** is a device that can add two binary digits. It is a type of digital circuit that performs the operation of additions of two number. It is mainly designed for the addition of binary number, but they can be used in various other applications like binary code decimal, address decoding, table index calculation, etc.

There are two types of Adder.

1. **Half Adder**
2. **Full Adder**

Half Adder

- There are two inputs and two outputs in a Half Adder.
- Inputs are named as A and B, and the outputs are named as Sum (S) and Carry (C).
- The Sum is X-OR of the input A and B.
- Carry is AND of the input A and B.

With the help of half adder, one can design a circuit that is capable of performing simple addition with the help of logic gates.

Let us first take a look at the addition of single bits.

$$0 + 0 = 0$$

$$0 + 1 = 1$$

$$1 + 0 = 1$$

$$1 + 1 = 10$$

These are the least possible single bit combinations. But the result for **1 + 1 =10.** This problem can be solved with the help of an **EX-OR** gate. The sum results can be re-written as a 2-bit output.

.

$0 + 0 = 00$

$0 + 1 = 01$

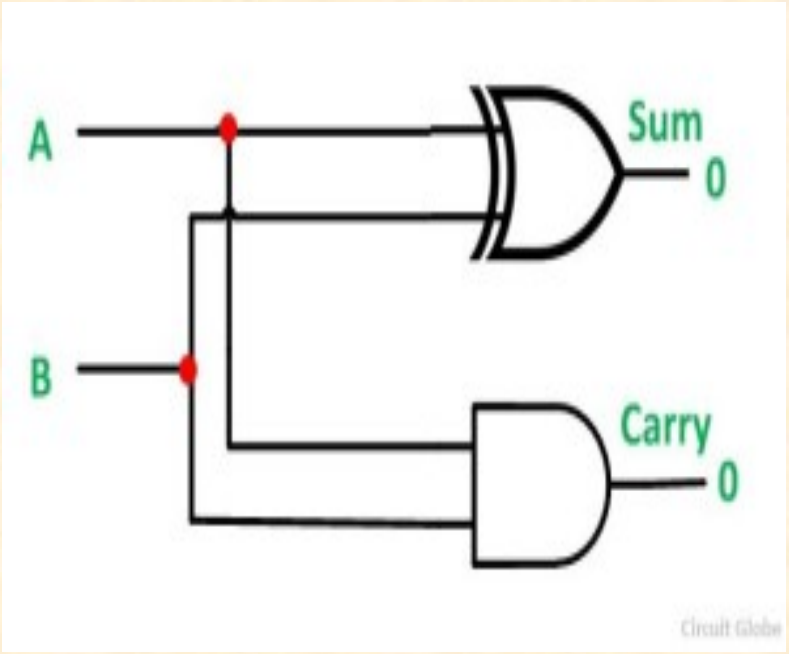
$1 + 0 = 01$

$1 + 1 = 10$

- Here the output “1” of “10” becomes the carry-out. **SUM** is the normal output and the **CARRY** is the carry-out.

Here is the truth table of half adder and circuit Diagram.

Inputs		**Outputs**	
A	B	Sum	Carry
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1



- The main disadvantage of this circuit is that it can only add two inputs and if there is any carry, it is neglected. Thus, the process is incomplete.
- To overcome this difficulty Full Adder is designed. While performing complex addition, there may be cases when you have to add two 8 bit byte together. This can be done with the help of Full Adder.

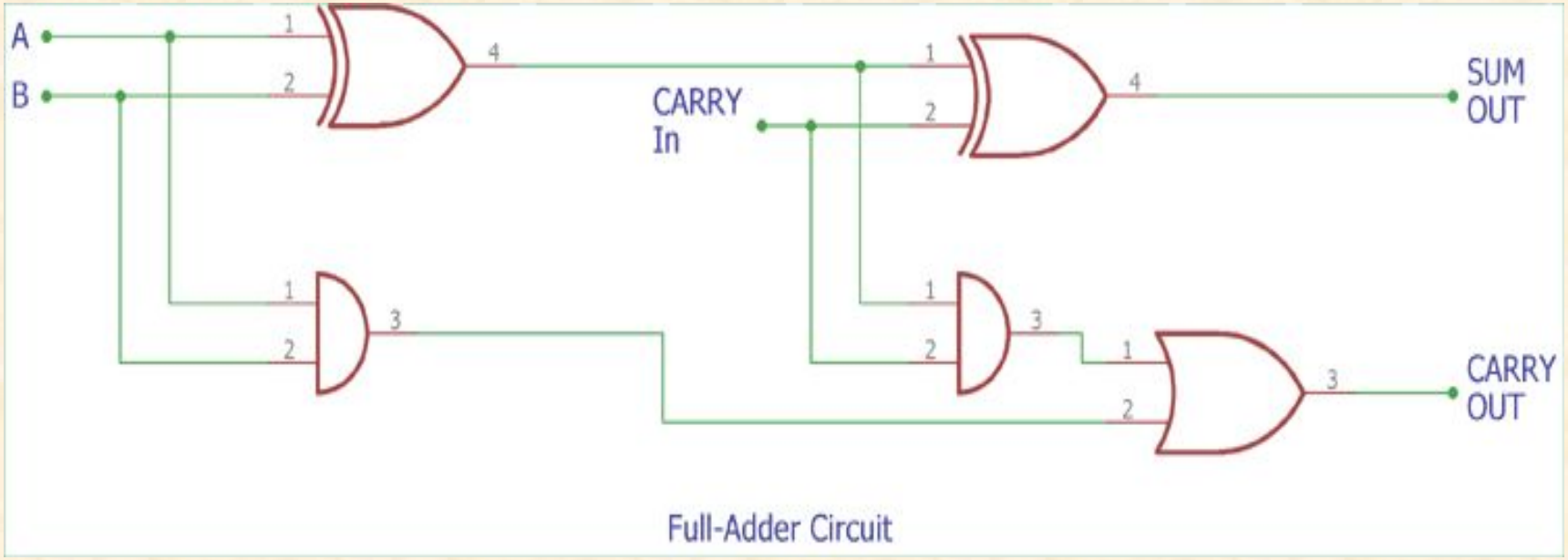
Full Adder

- The full adder is a little more difficult to implement than a half adder.
- The main difference between a half adder and a full adder is that the full-adder has three inputs and two outputs.
- The two inputs are A and B, and the third input is a carry input C_{IN}.
- The output carry is designated as C_{OUT}, and the normal output is designated as S.
- The **truth table** of the Full Adder Circuit is shown below.

Here is the truth table of Full Adder.

Inputs			**Outputs**	
A	B	CIN	COUT	S
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

Here is the truth table of circuit Diagram of Full Adder.



If we see the actual circuit inside the full adder, we will see two Half adders using **XOR gate and AND gate** with an additional **OR gate**.

Subtractor

Subtractor circuits take two binary numbers as input and subtract one binary number input from the other binary number input. Similar to adders, it gives out two outputs, difference and borrow (carry-in the case of Adder). There are two types of subtractors.

- Half Subtractor.
- Full Subtractor.

Half Subtractor

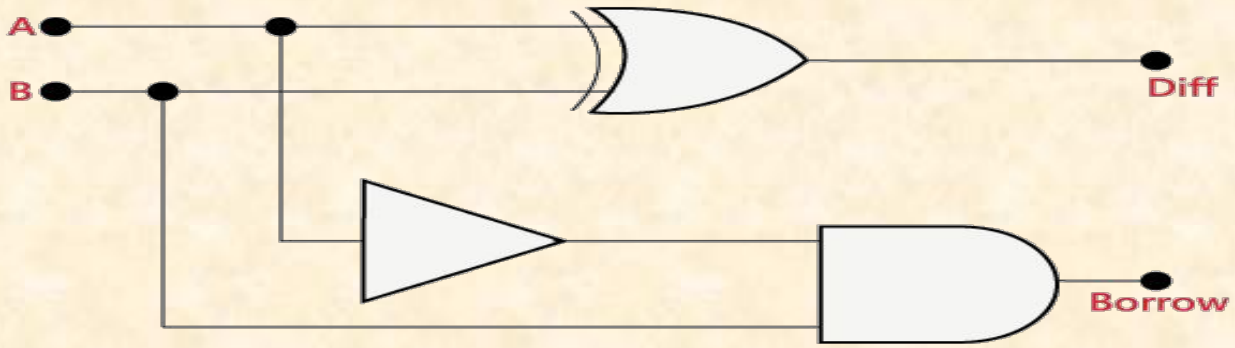
The half subtractor is also a building block for subtracting two binary numbers. It has two inputs and two outputs. This circuit is used to subtract two single bit binary numbers A and B. The **'diff'** and **'borrow'** are two output states of the half subtractor.

Truth Table

X	Y	D	B
0	0	0	0
0	1	1	1
1	0	1	0
1	1	0	0

From the above table we can draw the circuit for "difference" and "borrow".
In the above table,

1. 'A' and 'B' are the input variables whose values are going to be subtracted.
2. The 'Diff' and 'Borrow' are the variables whose values define the subtraction result, i.e., difference and borrow.
3. The first two rows and the last row, the difference is 1, but the 'Borrow' variable is 0.
4. The third row is different from the remaining one. When we subtract the bit 1 from the bit 0, the borrow bit is produced.



Half-Subtractor Circuit

Construction of Half Subtractor Circuit

In the block diagram, we have seen that it contains two inputs and two outputs. The **carry** and **sum** are the output states of the half subtractor. The half subtractor is designed with the help of the following logic gates:

1-input AND gate.

1-input Exclusive-OR Gate or Ex-OR Gate

1-NOT or inverter Gate

The **Boolean expression** of the **Half Adder circuit** is:

Diff= A XOR B (A⊕B)

Borrow= not-A AND B (A'.B)

Full Subtractor

The Half Subtractor is used to subtract only two numbers. To overcome this problem, a full subtractor was designed. The full subtractor is used to subtract three 1-bit numbers A, B, and C, which are minuend, subtrahend, and borrow, respectively. The full subtractor has three input states and two output states i.e., diff and borrow.

Truth Table

X	Y	Bin	D	Bout
0	0	0	0	0
0	0	1	1	1
0	1	0	1	1
0	1	1	0	1
1	0	0	1	0
1	0	1	0	0
1	1	0	0	0
1	1	1	1	1

In the above table,

1. 'A' and' B' are the input variables. These variables represent the two significant bits that are going to be subtracted.
2. '$Borrow_{in}$' is the third input which represents borrow.
3. The 'Diff' and 'Borrow' 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 variables.

From above table we can draw the circuit as shown below for "difference" and "borrow".

- The actual logic circuit of the full subtractor is shown in the above diagram. The full subtractor circuit construction can also be represented in a Boolean expression.

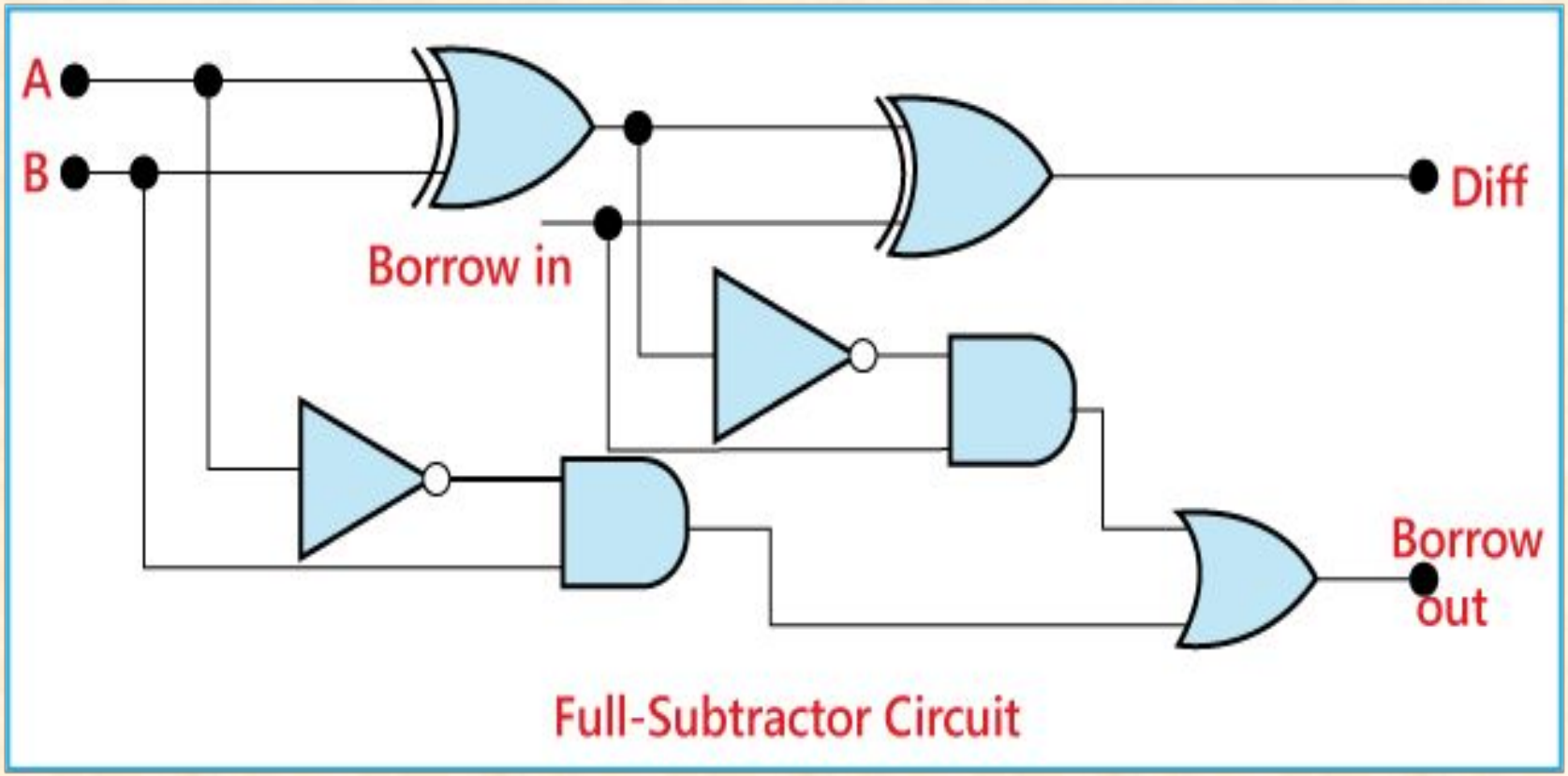
Diff:

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

Borrow:

- Perform the 'AND' operation of the inverted input A and B.
- Perform the 'XOR' operation of input A and B.
- Perform the 'OR' operations of both the outputs that come from the previous two steps. So the 'Borrow' can be represented as:

$$A'.B + (A \oplus B)'$$



Full-Subtractor Circuit

Basics of Flip Flop

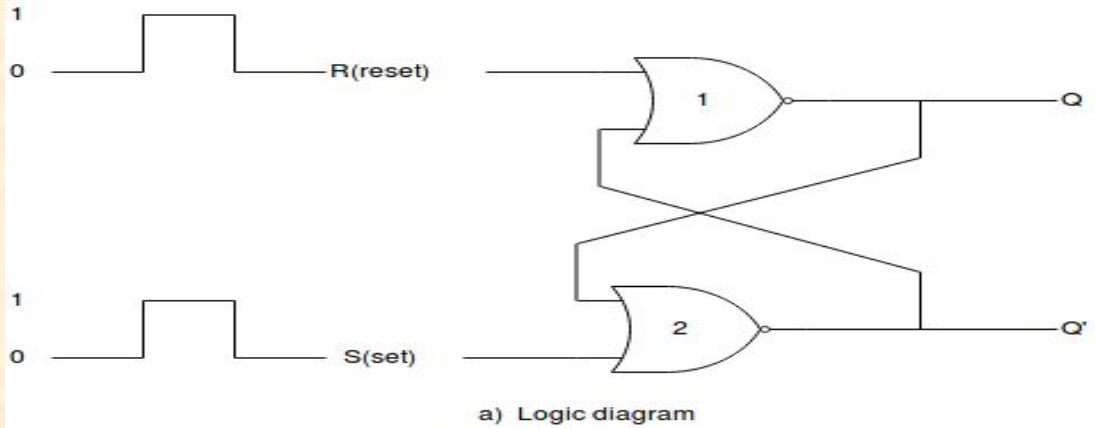
A circuit that has two stable states is treated as a **flip flop**. These stable states are used to store binary data that can be changed by applying varying inputs. The flip flops are the fundamental building blocks of the digital system. Flip flops and latches are examples of data storage elements. In the sequential logical circuit, the flip flop is the basic storage element. The latches and flip flops are the basic storage elements but different in working. There are the following types of flip flops:

- SR Flip-flops
- D Flip-flops
- JK Flip-flops
- JK Master Slave Flip-flops
- T Flip-flops

SR Flip-flop

The S-R flip flop is the most common flip flop used in the digital system. In SR flip flop, when the set input "S" is true, the output Y will be high, and Y' will be low. It is required that the wiring of the circuit is maintained when the outputs are established. We maintain the wiring until set or reset input goes high, or power is shutdown.

- The SET-RESET flip-flop consists of two NOR gates and also two NAND gates.
- The design of these flip flops also includes two inputs, called the SET [S] and RESET [R]. There are also two outputs, Q and Q'.

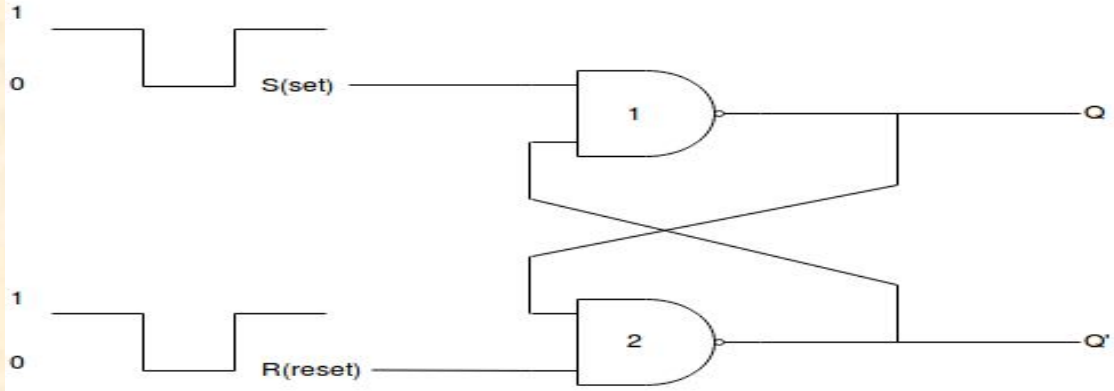


a) Logic diagram

S	R	Q	Q'
1	0	1	0
0	0	1	0
0	1	0	1
0	0	0	1
1	1	0	0

b) Truth table

fig: Basic flip-flop circuit with NOR gates



a) Logic diagram

S	R	Q	Q'
1	0	0	1
1	1	0	1
0	1	1	0
1	1	1	0
0	0	1	1

b) Truth table

fig: Basic flip-flop circuit with NAND gates

D Flip Flop

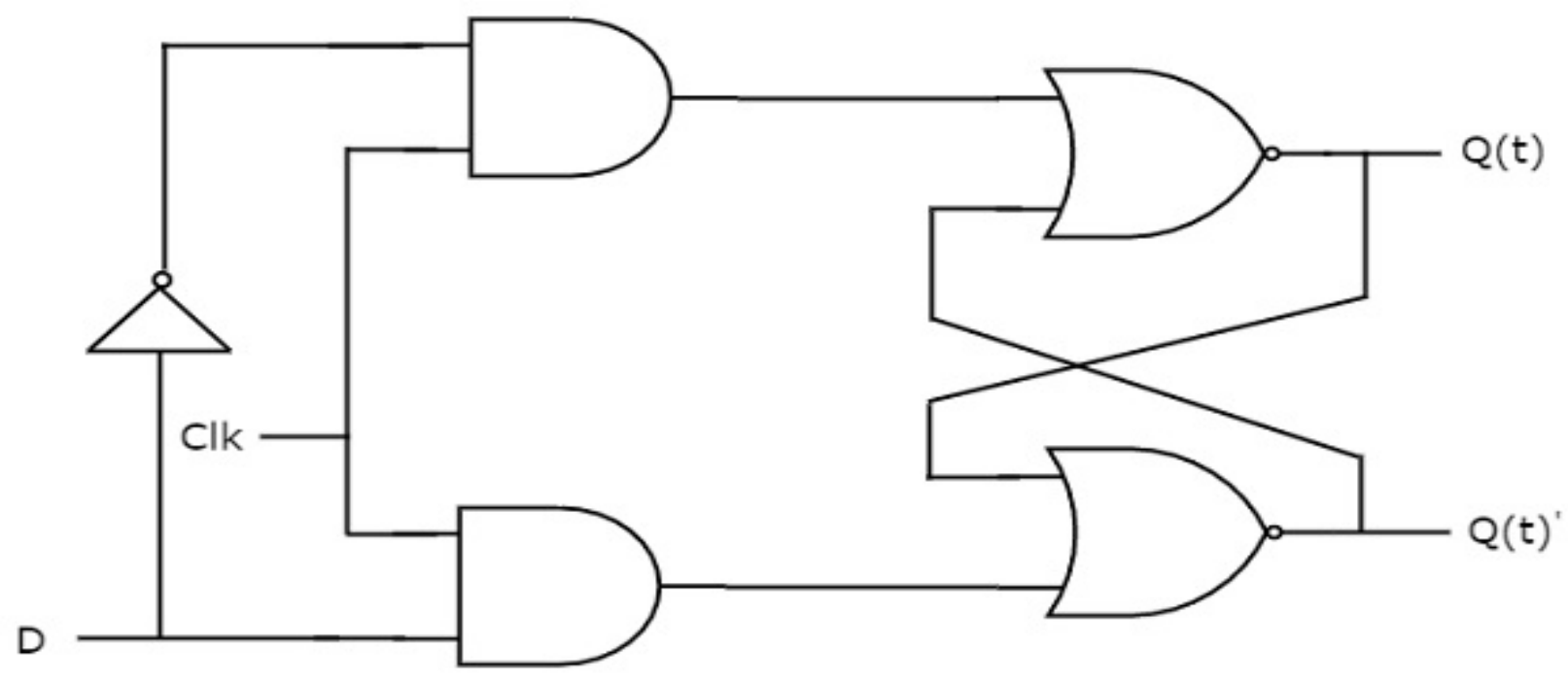
D flip flop is a widely used flip flop in digital systems. The D flip flop is mostly used in shift-registers, counters, and input synchronization.

D flip-flop operates with only positive clock transitions or negative clock transitions.

D flip-flop is insensitive to the changes in the input, D except for active transition of the clock signal.

D flip-flop always Hold the information, which is available on data input, D of earlier positive transition of clock signal.

D flip-flop is always equal to data input, D for every positive transition of the clock signal.

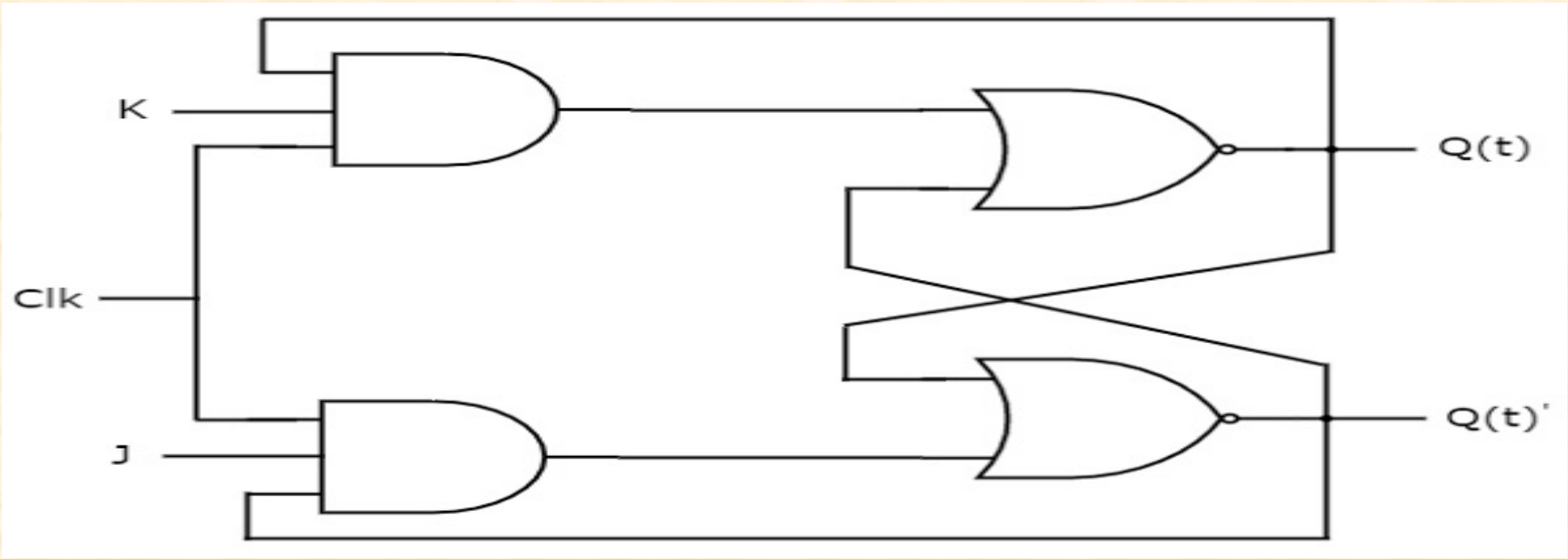


From the above figure, you can see that the D input is connected to the S input and the complement of the D input is connected to the R input.
When the value of CP is '1' (HIGH), the flip-flop moves to the SET state if it is '0' (LOW), the flip-flop switches to the CLEAR state.

J-K Flip-flop

The JK flip flop is used to remove the drawback of the S-R flip flop, i.e., undefined states. The JK flip flop is formed by doing modification in the SR flip flop. The S-R flip flop is improved in order to construct the J-K flip flop. When S and R input is set to true, the SR flip flop gives an inaccurate result. But in the case of JK flip flop, it gives the correct output.

When both the inputs J and K have a HIGH state, the flip-flop switches to the complement state, so, for a value of Q = 1, it switches to Q=0, and for a value of Q = 0, it switches to Q=1.



This circuit has two inputs J & K and two outputs Qt & Qt'. The operation of JK flip-flop is similar to SR flip-flop. Here, we considered the inputs of SR flip-flop as S = J Qt' and R = KQt in order to utilize the modified SR flip-flop for 4 combinations of inputs.

The following table shows the **state table** of JK flip-flop.

J	K	Qt+1t+1
0	0	Qt
0	1	0
1	0	1
1	1	Qt'

- Here, Qt & Qt+1 are present state & next state respectively. So, JK flip-flop can be used for one of these four functions such as **Hold, Reset, Set & Complement** of present state based on the input conditions, when positive transition of clock signal is applied. The following table shows the **characteristic table** of JK flip-flop.

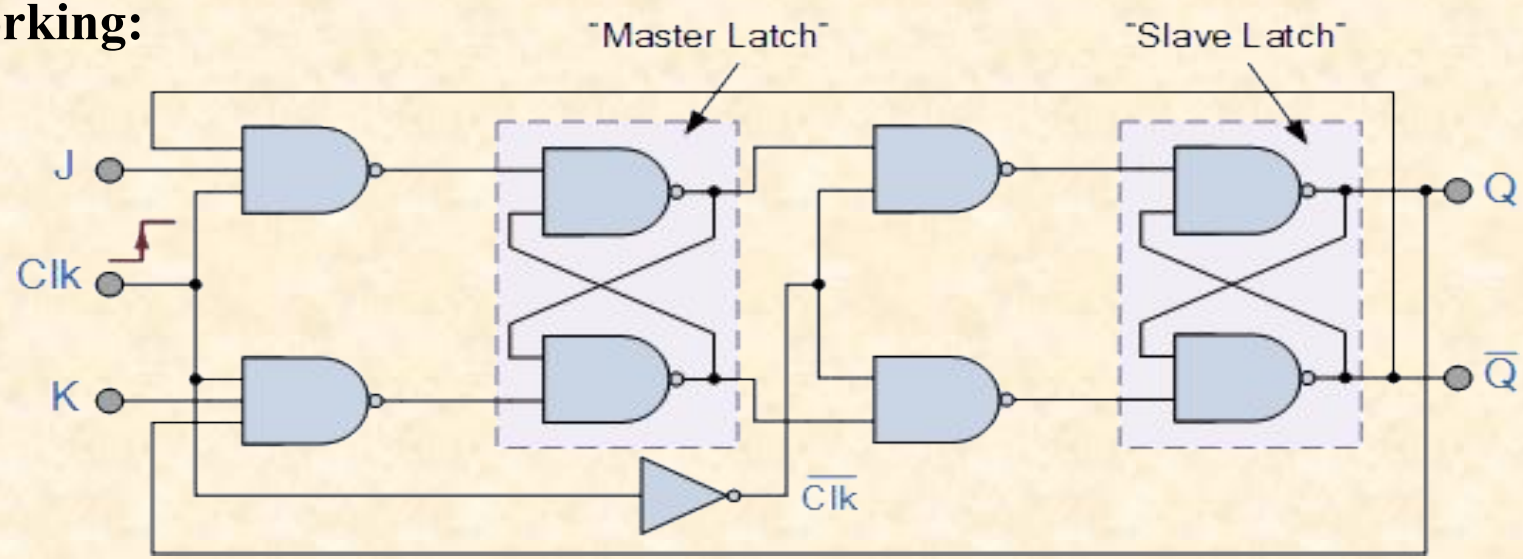
Present Inputs		Present State	Next State
J	**K**	**Qt**	**Q** t+1
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	0

Master Slave JK flip flop

The Master-Slave Flip-Flop is basically a combination of two JK flip-flops connected together in a series configuration. Out of these, one acts as the **“master”** and the other as a **“slave”**. The output from the master flip flop is connected to the two inputs of the slave flip flop whose output is fed back to inputs of the master flip flop.

In addition to these two flip-flops, the circuit also includes an **inverter**. The inverter is connected to clock pulse in such a way that the inverted clock pulse is given to the slave flip-flop. In other words if CP=0 for a master flip-flop, then CP=1 for a slave flip-flop and if CP=1 for master flip flop then it becomes 0 for slave flip flop.

Working:



The input signals J and K are connected to the gated "master" SR flip flop which "locks" the input condition while the clock (Clk) input is "HIGH" at logic level "1".

As the clock input of the "slave" flip flop is the inverse (complement) of the "master" clock input, the "slave" SR flip flop does not toggle.

The outputs from the "master" flip flop are only "seen" by the gated "slave" flip flop when the clock input goes "LOW" to logic level "0".

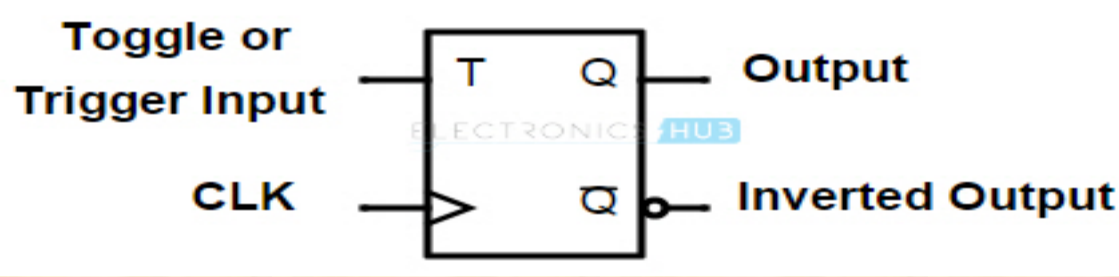
- When the clock is “LOW”, the outputs from the “master” flip flop are latched and any additional changes to its inputs are ignored. The gated “slave” flip flop now responds to the state of its inputs passed over by the “master” section.
- Then on the “Low-to-High” transition of the clock pulse the inputs of the “master” flip flop are fed through to the gated inputs of the “slave” flip flop and on the “High-to-Low” transition the same inputs are reflected on the output of the “slave” making this type of flip flop edge or pulse-triggered.
- Then, the circuit accepts input data when the clock signal is “HIGH”, and passes the data to the output on the falling-edge of the clock signal. In other words, the **Master-Slave JK Flip flop** is a “Synchronous” device as it only passes data with the timing of the clock signal.

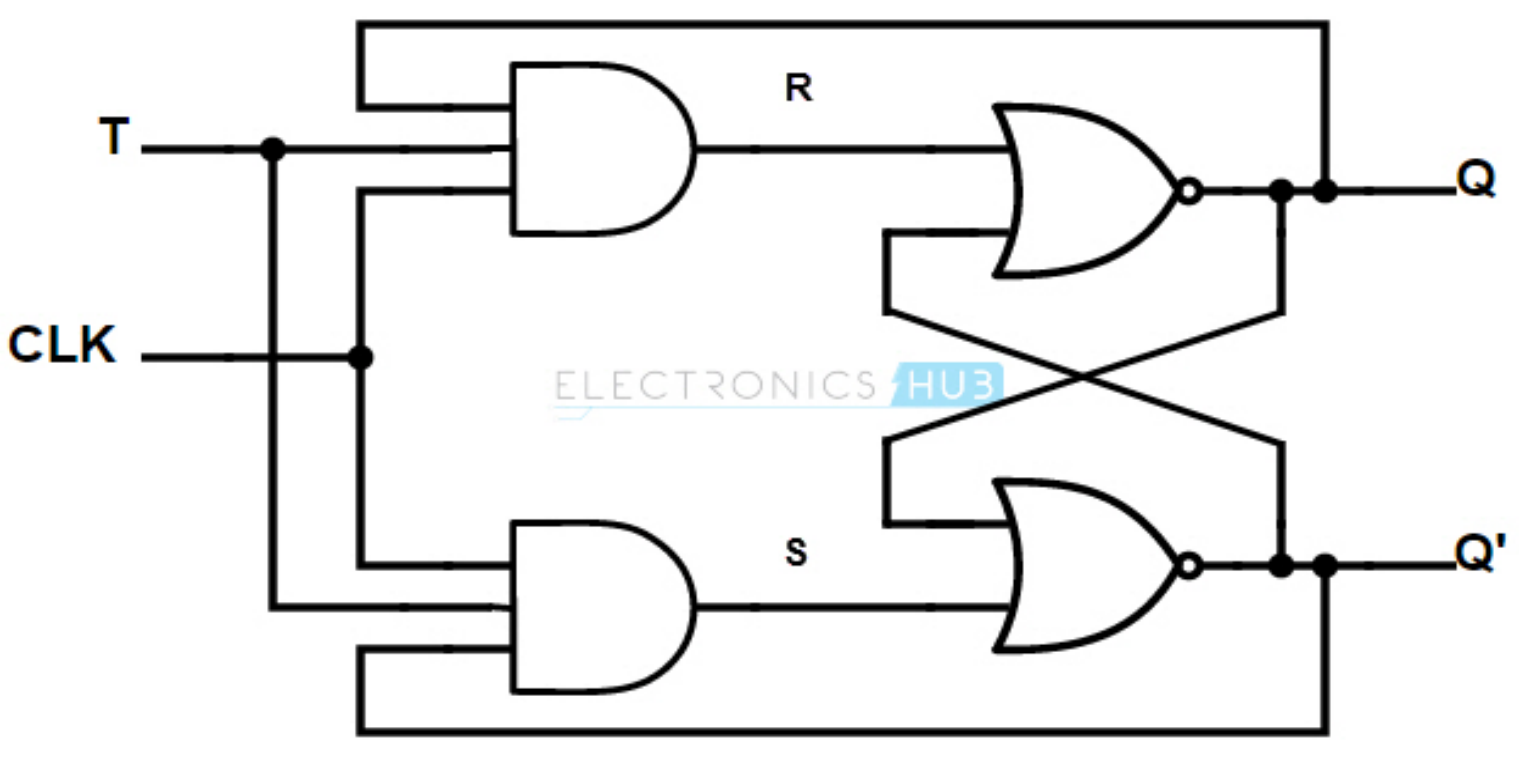
T Flip Flop

T flip – flop is also known as “Toggle Flip – flop”. To avoid the occurrence of intermediate state in SR flip – flop, we should provide only one input to the flip – flop called Trigger input or Toggle input (T). Then the flip – flop acts as a Toggle switch.

- **Construction**

- We can construct a T flip – flop by connecting AND gates as input to the NOR gate SR latch.
- And these AND gate inputs are fed back with the present state output Q and its complement Q’ to each AND gate.
- A toggle input (T) is connected in common to both the AND gates as an input.


Toggle or
Trigger Input
CLK
T
Q
Q
Output
Inverted Output


T
CLK
R
S
Q
Q'

In simple terms, the operation of the T flip – flop is

- When the T input is low, then the next sate of the T flip flop is same as the present state.
- T = 0 and present state = 0 then the next state = 0
- T = 1 and present state = 1 then the next state = 1

- When the T input is high and during the positive transition of the clock signal, the next state of the T flip – flop is the inverse of present state.
- T = 1 and present state = 0 then the next state = 1
- T = 1 and present state = 1 then the next state = 0

Encoder and Decoder

- The encoders and decoders play an essential role in digital electronics projects.

- Encoders & Decoders are used to convert data from one form to another form.

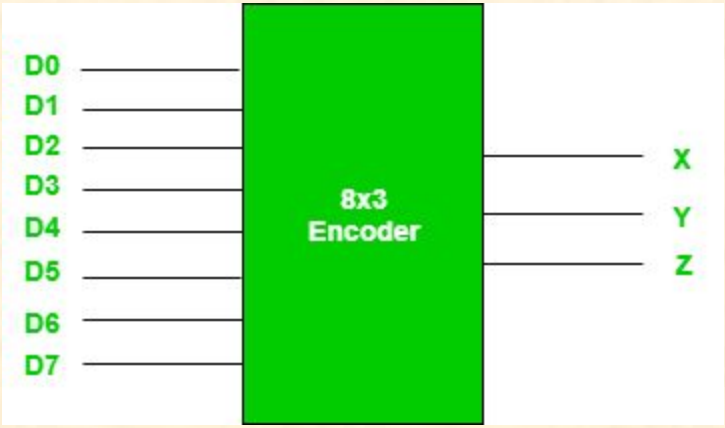
- These are frequently used in communication system such as telecommunication, networking, etc..to transfer data from one end to the other end.

- Similarly, in the digital domain, for easy transmission of data, it is often encrypted or placed within codes, and then transmitted.

Encoders

An encoder is a combinational circuit that converts binary information in the form of a 2^N input lines into 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.

Examplee: 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.



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.

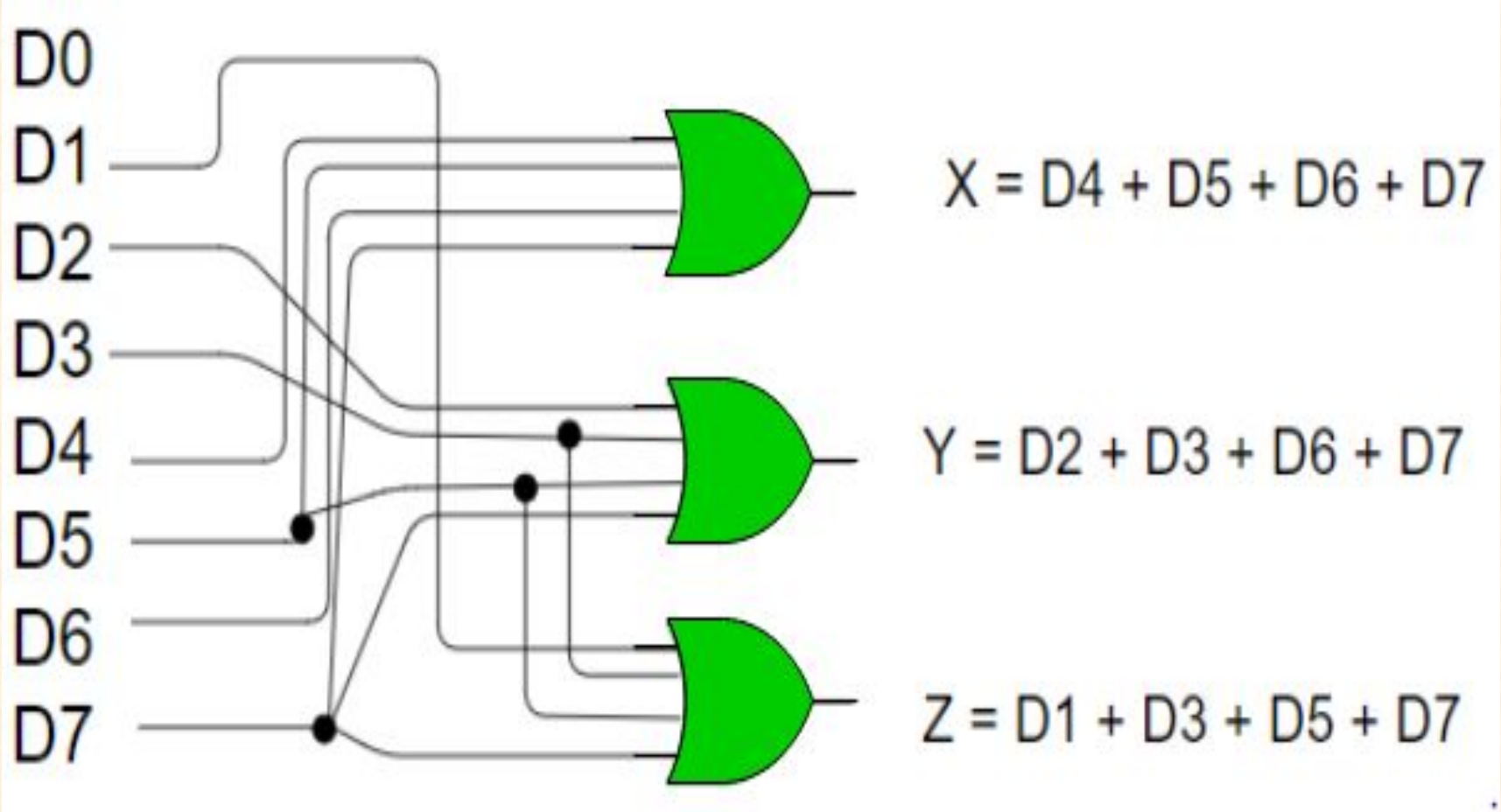
Implementation:

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


D0
D1
D2
D3
D4
D5
D6
D7
X = D4 + D5 + D6 + D7
Y = D2 + D3 + D6 + D7
Z = D1 + D3 + D5 + D7

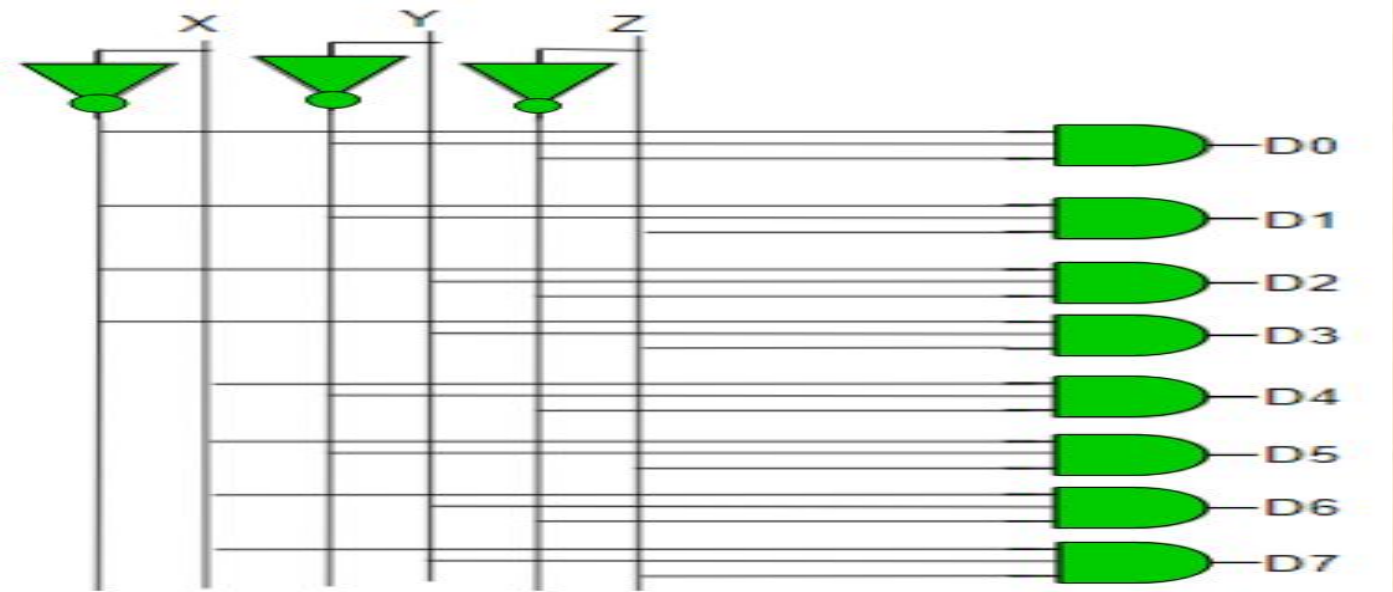
Decoders

A decoder does the opposite job of an encoder. It is a combinational circuit that converts n lines of input into 2^n lines of output.

Example of 3-to-8 line decoder.

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

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

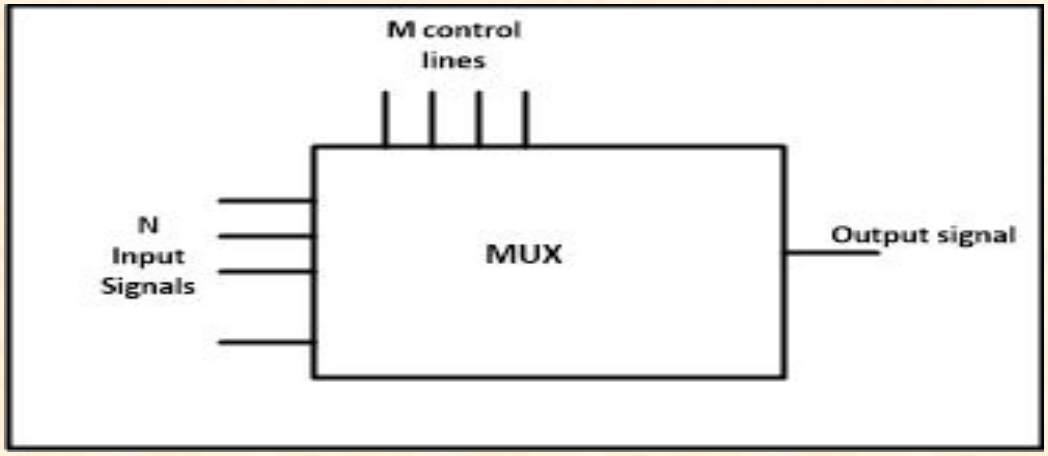


Multiplexer and Demultiplexer

- A **multiplexer** is a circuit that accept many input but give only one output.
- A **demultiplexer** function exactly in the reverse of a multiplexer, that is a demultiplexer accepts only one input and gives many outputs.
- Generally multiplexer and demultiplexer are used together, because of the communication systems are bi directional.

Mutliplexer

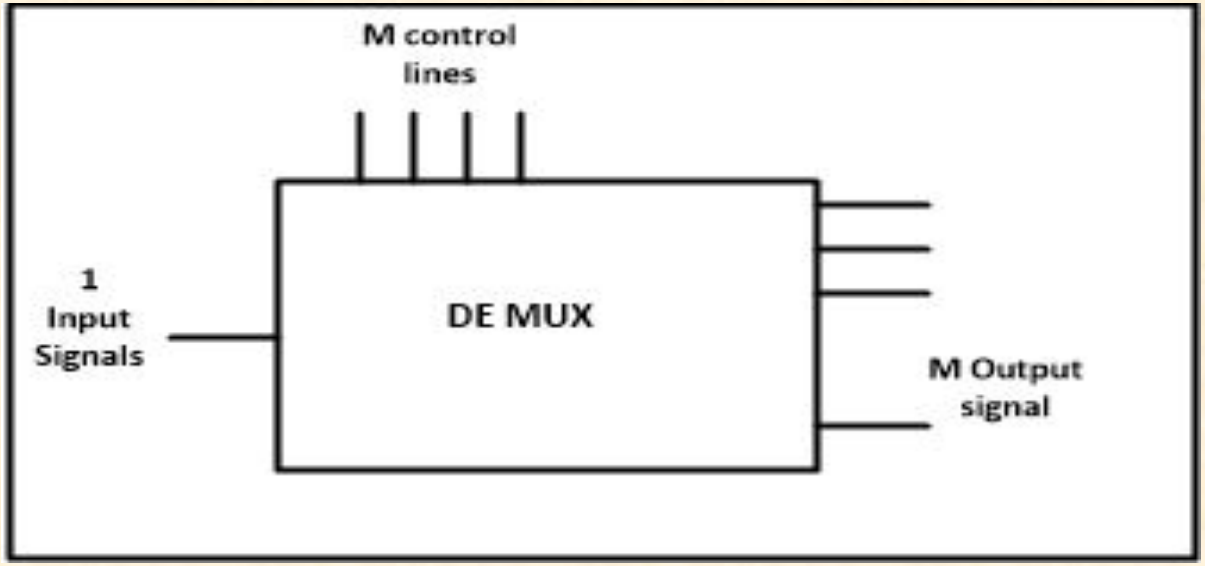
- Multiplexer means many into one. A multiplexer is a circuit used to select and route any one of the several input signals to a signal output.
- Multiplexer handle two type of data that is analog and digital. For analog application, multiplexer are built of relays and transistor switches. For digital application, they are built from standard logic gates.



- **Applications of Multiplexer:**
- **Communication system** – Communication system is a set of system that enable communication like transmission system, relay and tributary station, and communication network. The efficiency of communication system can be increased considerably using multiplexer. Multiplexer allow the process of transmitting different type of data such as audio, video at the same time using a single transmission line.
- **Telephone network –** In telephone network, multiple audio signals are integrated on a single line for transmission with the help of multiplexers. In this way, multiple audio signals can be isolated and eventually, the desire audio signals reach the intended recipients.
- **Computer memory** – Multiplexers are used to implement huge amount of memory into the computer, at the same time reduces the number of copper lines required to connect the memory to other parts of the computer circuit.
- **Transmission from the computer system of a satellite** – Multiplexer can be used for the transmission of data signals from the computer system of a satellite or spacecraft to the ground system using the GPS (Global Positioning System) satellites.

Demultiplexer

Demultiplexer means one to many. A demultiplexer is a circuit with one input and many output. By applying control signal.



- **Applications of Demultiplexer:**
- **Communication System** – Communication system use multiplexer to carry multiple data like audio, video and other form of data using a single line for transmission. This process make the transmission easier. The demultiplexer receive the output signals of the multiplexer and converts them back to the original form of the data at the receiving end. The multiplexer and demultiplexer work together to carry out the process of transmission and reception of data in communication system.
- **ALU (Arithmetic Logic Unit)** – In an ALU circuit, the output of ALU can be stored in multiple registers or storage units with the help of demultiplexer. The output of ALU is fed as the data input to the demultiplexer. Each output of demultiplexer is connected to multiple register which can be stored in the registers.
- **Serial to parallel converter** – The serial to parallel converter is used to reform parallel data. In this method, serial data are given as an input to the De-multiplexer at a regular interval, and a counter is attached to the demultiplexer at the control i/p to sense the data signal at the demultiplexer's o/p. When all data signals are stored, the output of the demultiplexer can be read out in parallel.