Digital Circuits - Logic Gates

Digital electronic circuits operate with voltages of two logic levels namely Logic Low and Logic High. The range of voltages corresponding to Logic Low is represented with '0'. Similarly, the range of voltages corresponding to Logic High is represented with '1'.

The basic digital electronic circuit that has one or more inputs and single output is known as Logic gate. Hence, the Logic gates are the building blocks of any digital system. We can classify these Logic gates into the following three categories.

1. Basic gates
2. Universal gates
3. Special gates

Now, let us discuss about the Logic gates come under each category one by one.

Basic Gates

We learnt that the Boolean functions can be represented either in sum of products form or in product of sums form based on the requirement. So, we can implement these Boolean functions by using basic gates. The basic gates are AND, OR & NOT gates.

➢ AND gate

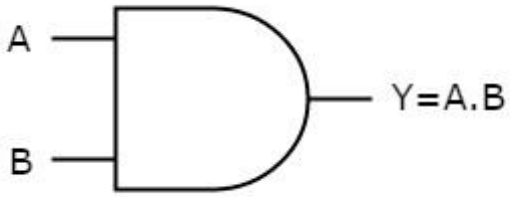
An AND gate is a digital circuit that has two or more inputs and produces an output, which is the logical AND of all those inputs. It is optional to represent the Logical AND with the symbol '.'.

The following table shows the truth table of 2-input AND gate.

A	B	Y = A.B
0	0	0
0	1	0
1	0	0
1	1	1

Here A, B are the inputs and Y is the output of two input AND gate. If both inputs are '1', then only the output, Y is '1'. For remaining combinations of inputs, the output, Y is '0'.

The following figure shows the symbol of an AND gate, which is having two inputs A, B and one output, Y.



This AND gate produces an output (Y), which is the logical AND of two inputs A, B. Similarly, if there are 'n' inputs, then the AND gate produces an output, which is the logical AND of all those inputs. That means, the output of AND gate will be '1', when all the inputs are '1'.

OR gate

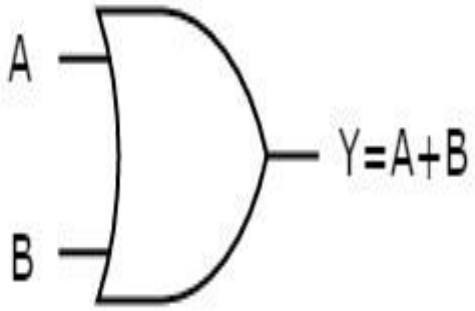
An OR gate is a digital circuit that has two or more inputs and produces an output, which is the logical OR of all those inputs. This logical OR is represented with the symbol '+'.

The following table shows the truth table of 2-input OR gate.

A	B	Y = A + B
0	0	0
0	1	1
1	0	1
1	1	1

Here A, B are the inputs and Y is the output of two input OR gate. If both inputs are '0', then only the output, Y is '0'. For remaining combinations of inputs, the output, Y is '1'.

The following figure shows the symbol of an OR gate, which is having two inputs A, B and one output, Y.



This OR gate produces an output (Y), which is the logical OR of two inputs A, B. Similarly, if there are 'n' inputs, then the OR gate produces an output, which is the logical OR of all those inputs. That means, the output of an OR gate will be '1', when at least one of those inputs is '1'.

NOT gate

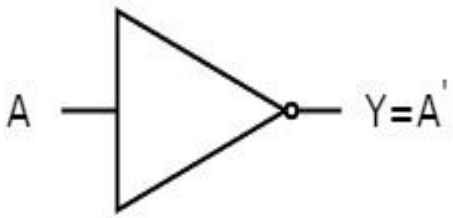
A NOT gate is a digital circuit that has single input and single output. The output of NOT gate is the logical inversion of input. Hence, the NOT gate is also called as inverter.

The following table shows the truth table of NOT gate.

A	Y = A’
0	1
1	0

Here A and Y are the input and output of NOT gate respectively. If the input, A is ‘0’, then the output, Y is ‘1’. Similarly, if the input, A is ‘1’, then the output, Y is ‘0’.

The following figure shows the symbol of NOT gate, which is having one input, A and one output, Y.



This NOT gate produces an output (Y), which is the complement of input, A.

Universal gates

NAND & NOR gates are called as universal gates. Because we can implement any Boolean function, which is in sum of products form by using NAND gates

alone. Similarly, we can implement any Boolean function, which is in product of sums form by using NOR gates alone.

NAND gate (AND+NOT)

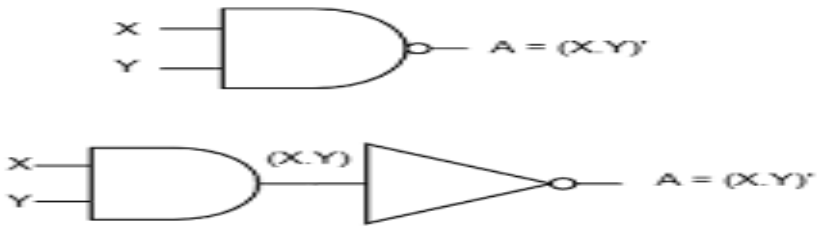
NAND gate is a digital circuit that has two or more inputs and produces an output, which is the inversion of logical AND of all those inputs.

The following table shows the truth table of 2-input NAND gate.

A	B	Y = (A.B)'
0	0	1
0	1	1
1	0	1
1	1	0

Here A, B are the inputs and Y is the output of two input NAND gate. When both inputs are '1', the output, Y is '0'. If at least one of the input is zero, then the output, Y is '1'. This is just opposite to that of two input AND gate operation.

The following image shows the symbol of NAND gate, which is having two inputs A, B and one output, Y.



NAND gate operation is same as that of AND gate followed by an inverter. That's why the NAND gate symbol is represented like that.

NOR gate

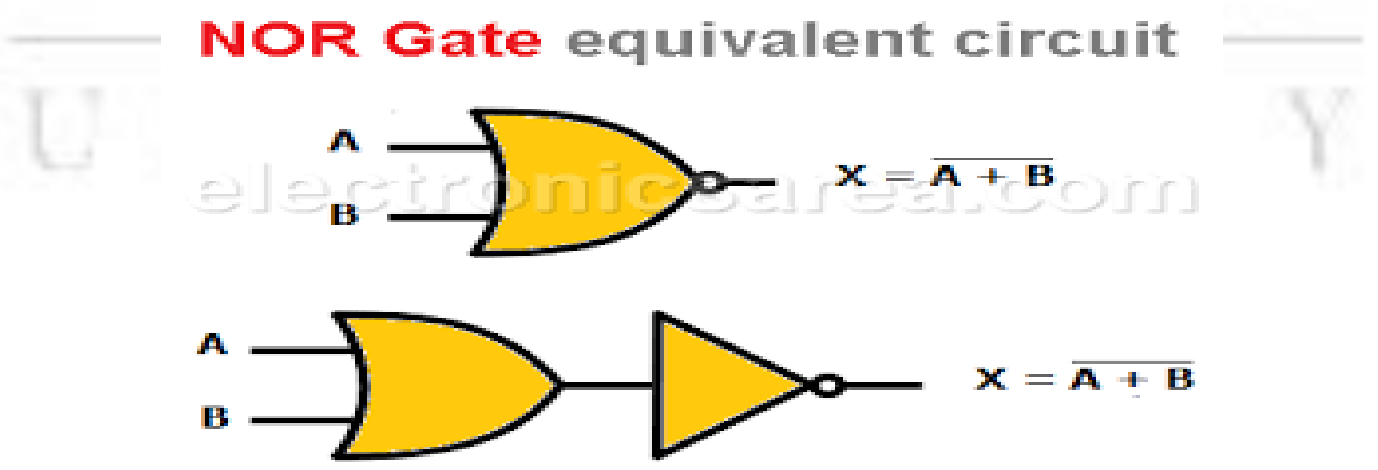
NOR gate is a digital circuit that has two or more inputs and produces an output, which is the inversion of logical OR of all those inputs.

The following table shows the truth table of 2-input NOR gate

A	B	Y = (A+B)'
0	0	1
0	1	0
1	0	0
1	1	0

Here A, B are the inputs and Y is the output. If both inputs are '0', then the output, Y is '1'. If at least one of the input is '1', then the output, Y is '0'. This is just opposite to that of two input OR gate operation.

The following figure shows the symbol of NOR gate, which is having two inputs A, B and one output, Y.



NOR gate operation is same as that of OR gate followed by an inverter. That's why the NOR gate symbol is represented like that.

Special Gates

Ex-OR & Ex-NOR gates are called as special gates. Because, these two gates are special cases of OR & NOR gates.

Ex-OR gate

The full form of Ex-OR gate is Exclusive-OR gate. Its function is same as that of OR gate except for some cases, when the inputs having even number of ones.

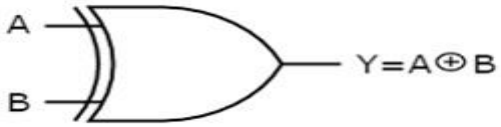
The following table shows the truth table of 2-input Ex-OR gate.

A	B	Y = A⊕B
0	0	0
0	1	1
1	0	1
1	1	0

Here A, B are the inputs and Y is the output of two input Ex-OR gate. The truth table of Ex-OR gate is same as that of OR gate for first three rows. The only modification is in the fourth row. That means, the output (Y) is zero instead of one, when both the inputs are one, since the inputs having even number of ones.

Therefore, the output of Ex-OR gate is '1', when only one of the two inputs is '1'. And it is zero, when both inputs are same.

Below figure shows the symbol of Ex-OR gate, which is having two inputs A, B and one output, Y.



Ex-OR gate operation is similar to that of OR gate, except for few combination(s) of inputs. That's why the Ex-OR gate symbol is represented like that. The output of Ex-OR gate is '1', when odd number of ones present at the inputs. Hence, the output of Ex-OR gate is also called as an odd function.

Ex-NOR gate

The full form of Ex-NOR gate is Exclusive-NOR gate. Its function is same as that of NOR gate except for some cases, when the inputs having even number of ones.

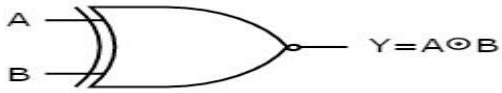
The following table shows the truth table of 2-input Ex-NOR gate.

A	B	Y = A⊙B
0	0	1
0	1	0
1	0	0
1	1	1

Here A, B are the inputs and Y is the output. The truth table of Ex-NOR gate is same as that of NOR gate for first three rows. The only modification is in the

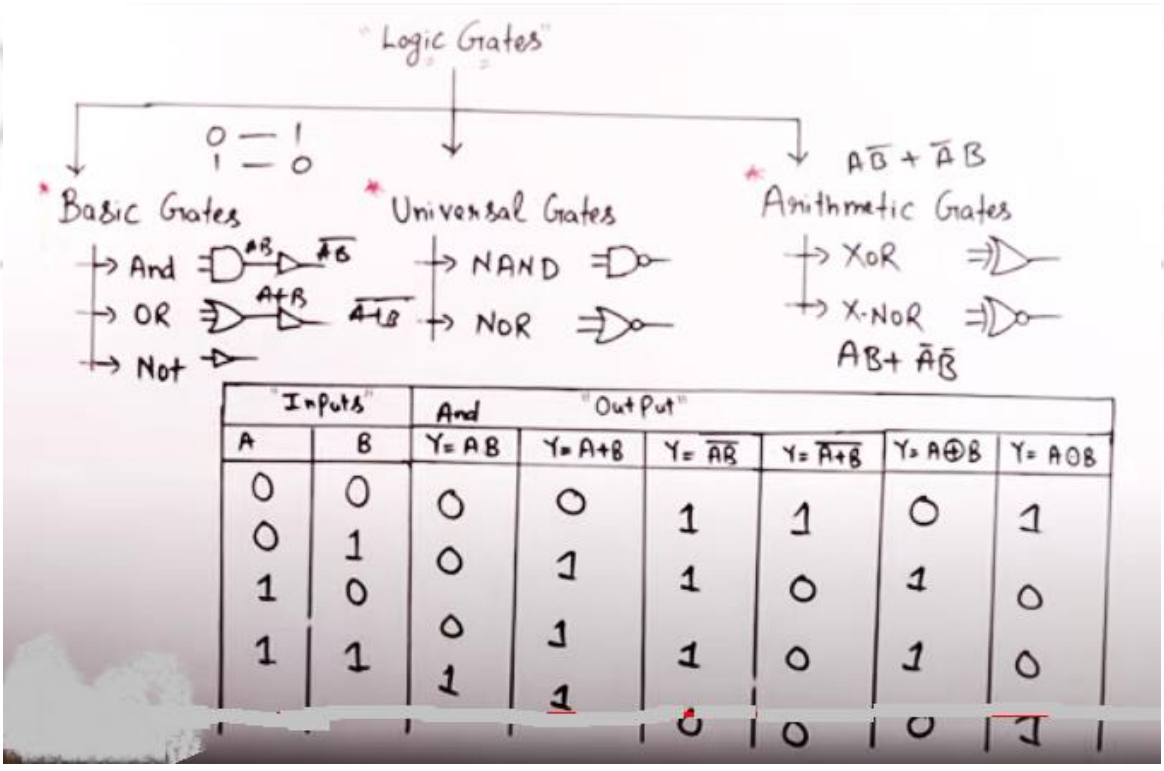
fourth row. That means, the output is one instead of zero, when both the inputs are one. Therefore, the output of Ex-NOR gate is '1', when both inputs are same. And it is zero, when both the inputs are different.

The following figure shows the symbol of Ex-NOR gate, which is having two inputs A, B and one output, Y.



Ex-NOR gate operation is similar to that of NOR gate, except for few combination(s) of inputs. That's why the Ex-NOR gate symbol is represented like that. The output of Ex-NOR gate is '1', when even number of ones present at the inputs. Hence, the output of Ex-NOR gate is also called as an even function.

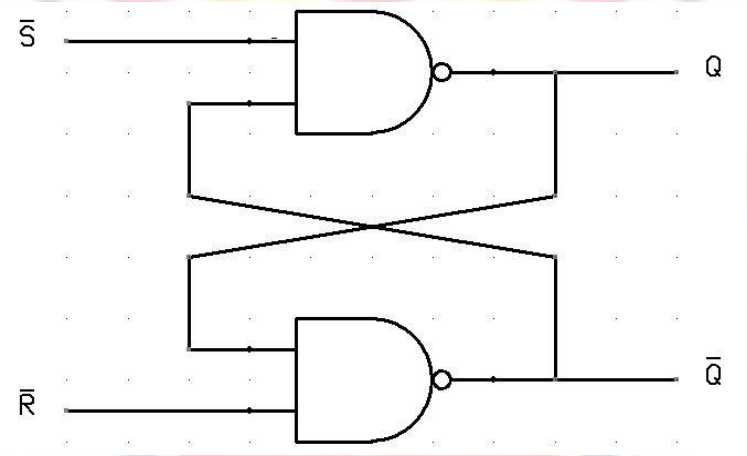
From the above truth tables of Ex-OR & Ex-NOR logic gates, we can easily notice that the Ex-NOR operation is just the logical inversion of Ex-OR operation.



Inputs		And		Output			
A	B	Y = AB	Y = A+B	Y = $\overline{AB}$	Y = $\overline{A+B}$	Y = A⊕B	Y = A⊙B
0	0	0	0	1	1	0	1
0	1	0	1	1	0	1	0
1	0	0	1	1	0	1	0
1	1	1	1	0	0	0	1

Digital Circuits - Flip-Flops

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.



Flip-flops are formed from pairs of logic gates where the gate outputs are fed into one, of the inputs of the other gate in the pair. This results in a regenerative circuit 'having two stable output states (binary one and zero). Frequently additional gates are added for control of the circuit. While some flip-flops are operated " as yet chrohously (without timing pulses), most are ,operated 'Linier clock control in a synchronous system. , Individual fli~fiol's " can"be ~ combiried " 'to fond memory registers, ' couhters' ' and shlli' registers

There are the following types of flip flops:

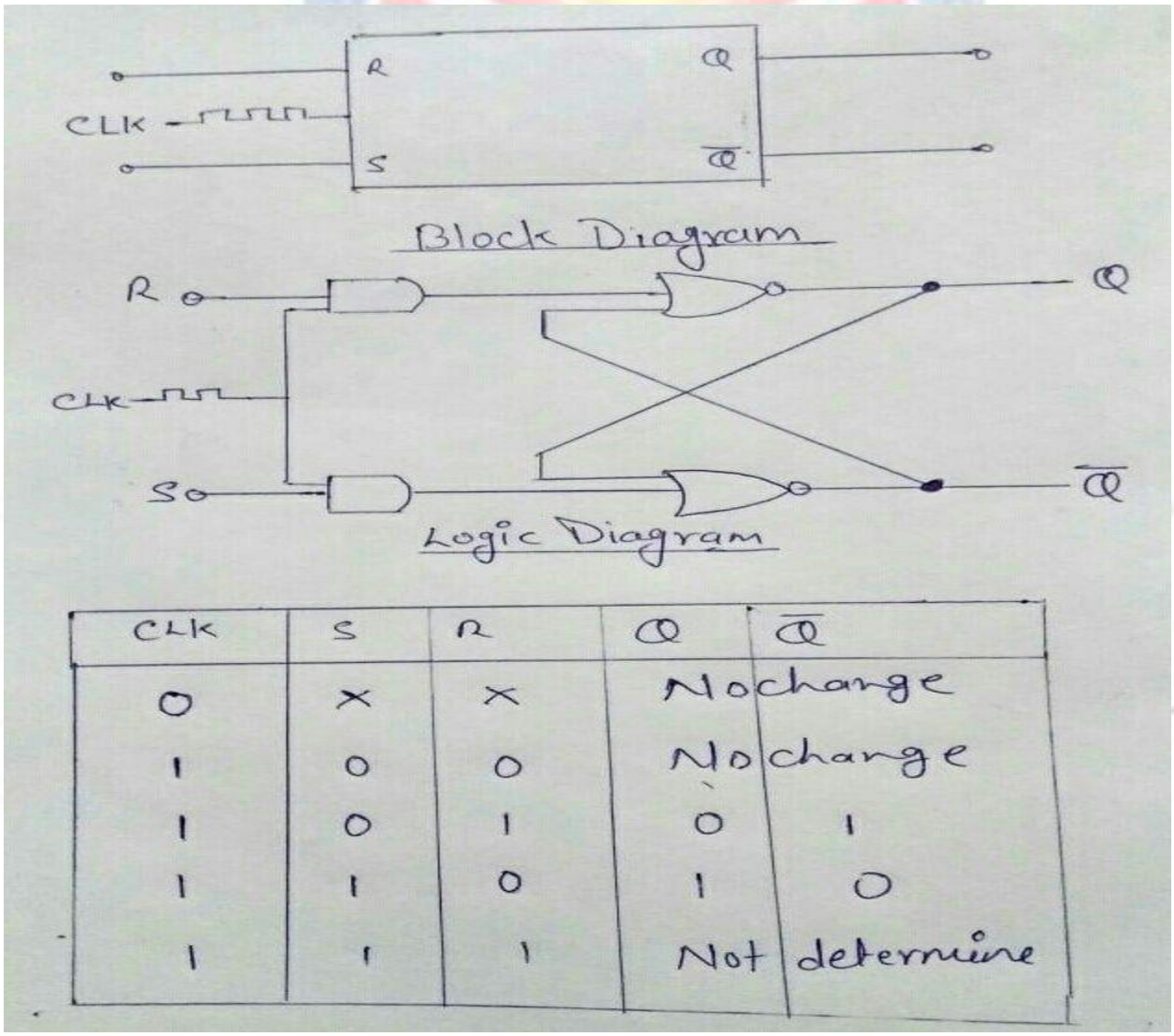
- SR Flip-Flop
- D Flip-Flop
- JK Flip-Flop
- T Flip-Flop

SR Flip-Flop

SR flip-flop operates with only positive clock transitions or negative clock transitions. Whereas, SR latch operates with enable signal. The **circuit diagram** of SR flip-flop is shown in the following figure.

This circuit has two inputs S & R and two outputs Q & Q'. The operation of SR flip-flop is similar to SR Latch. But, this flip-flop affects the outputs only when positive transition of the clock signal is applied instead of active enable.

The following table shows the state table of SR flip-flop.



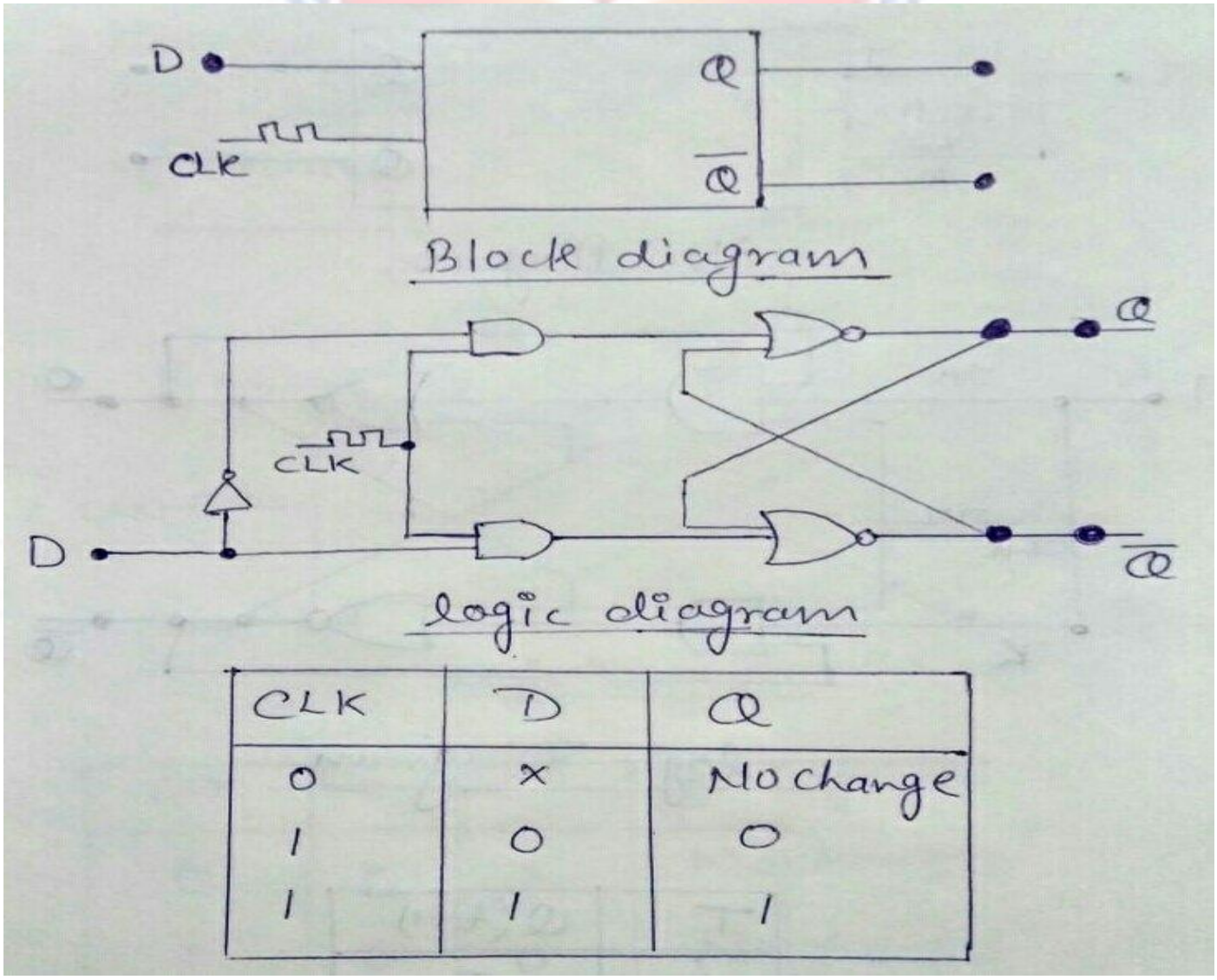
CLK	S	R	Q	$\overline{Q}$
0	x	x	No change	
1	0	0	No change	
1	0	1	0	1
1	1	0	1	0
1	1	1	Not determine	

R active (R=1) so output reset Q=0 and S active (S=1) so output set Q=1 .

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

D Flip-Flop

D(data) flip-flop operates with only positive clock transitions or negative clock transitions. Whereas, D latch operates with enable signal. That means, the output of D flip-flop is insensitive to the changes in the input, D except for active transition of the clock signal. The circuit diagram of D flip-flop is shown in the following figure.



CLK	D	Q
0	x	No change
1	0	0
1	1	1

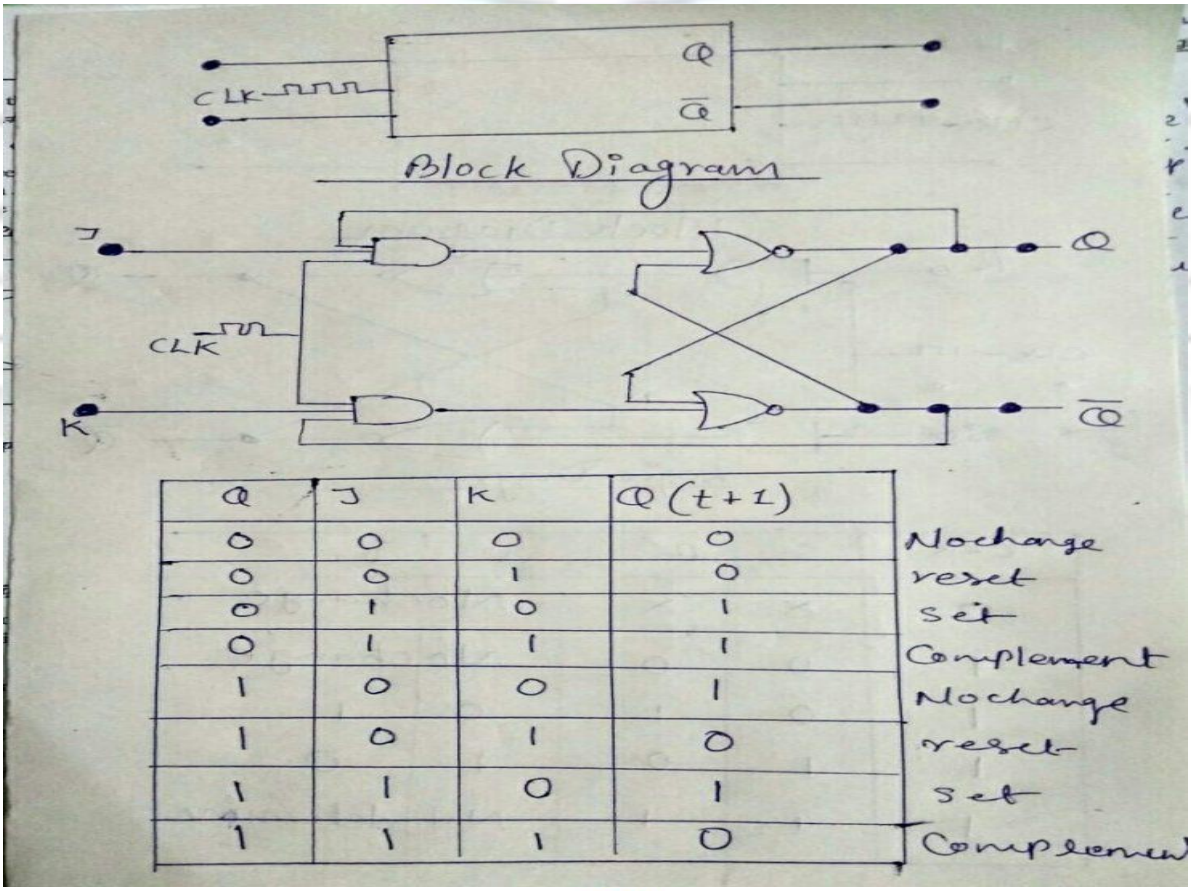
This circuit has single input D. The operation of D flip-flop is similar to D Latch. But, this flip-flop affects the outputs only when positive transition of the clock signal is applied instead of active enable.

Therefore, D flip-flop always Hold the information, which is available on data input, D of earlier positive transition of clock signal. From the above state table, we can directly write the next state equation as

Next state of D flip-flop is always equal to data input, D for every positive transition of the clock signal. Hence, D flip-flops can be used in registers, shift registers and some of the counters.

JK Flip-Flop

JK flip-flop is the modified version of SR flip-flop. It operates with only positive clock transitions or negative clock transitions. The circuit diagram of JK flip-flop is shown in the following figure.

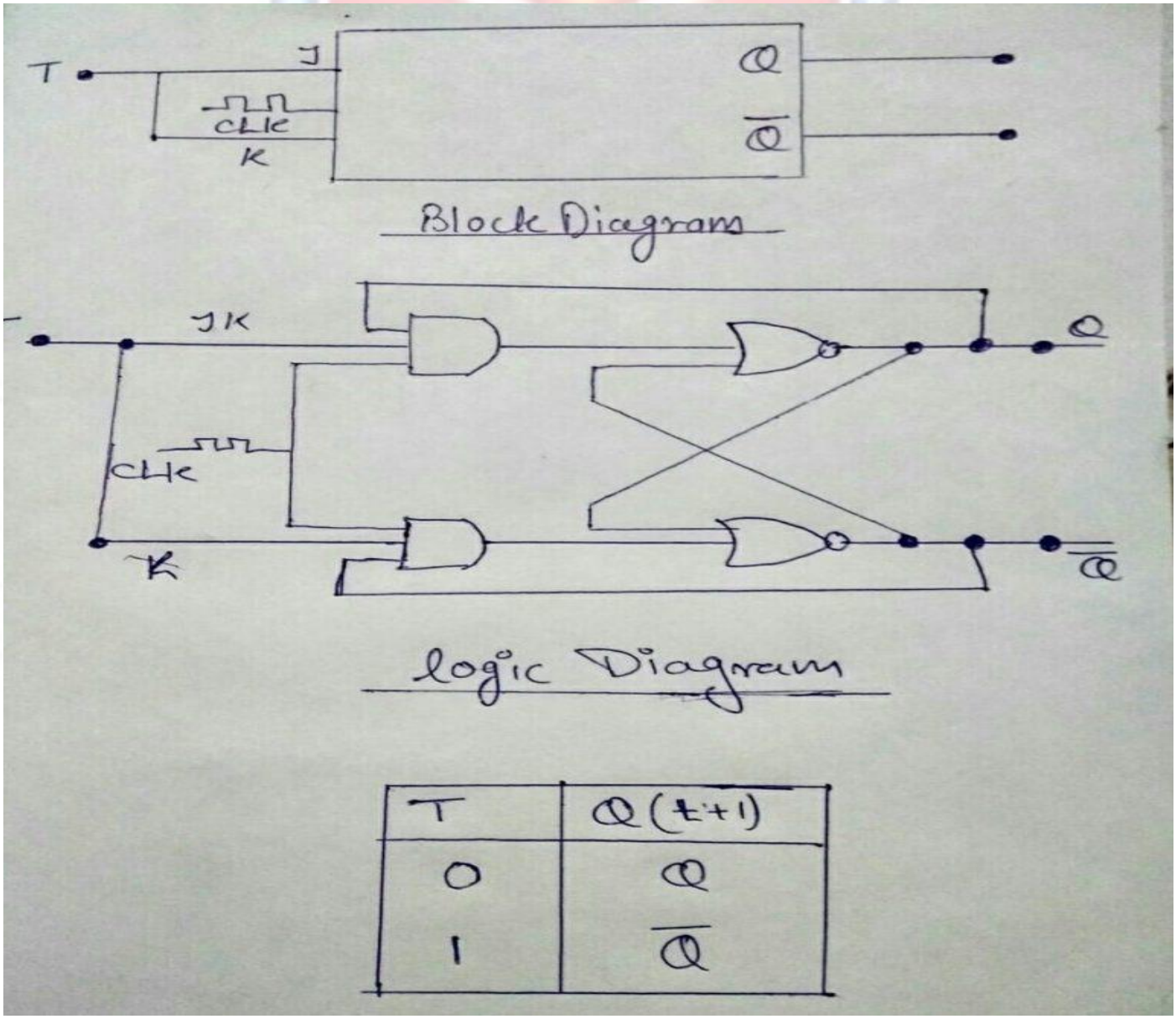


Q	J	K	Q(t+1)	
0	0	0	0	No change
0	0	1	0	reset
0	1	0	1	set
0	1	1	1	Complement
1	0	0	1	No change
1	0	1	0	reset
1	1	0	1	set
1	1	1	0	Complement

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.

T Flip-Flop

T (Toggle)flip-flop is the simplified version of JK flip-flop. It is obtained by connecting the same input 'T' to both inputs of JK flip-flop. It operates with only positive clock transitions or negative clock transitions. The circuit diagram of T flip-flop is shown in the following figure.



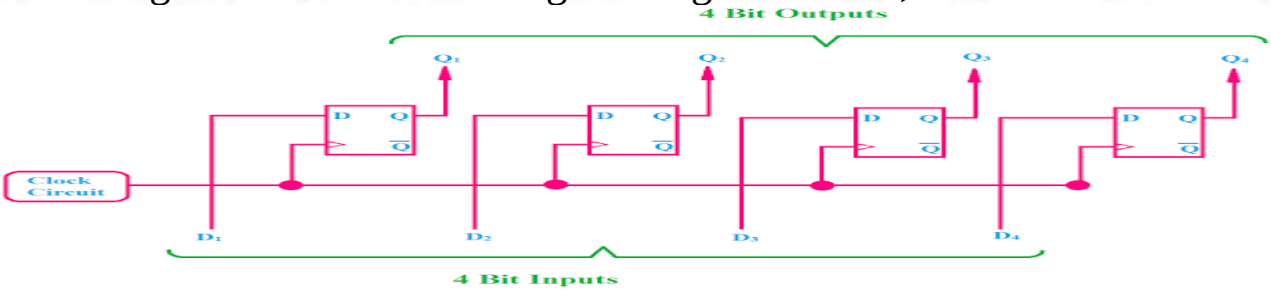
Register:-

A Register is a circuit consisting of Flip-Flops which can store more than one-bit data. The register is nothing but a sequential logic circuit in digital electronics. We know that there are two types of circuits in digital electronics one is Combinational logic circuit and other is Sequential logic circuit. We know that Flip-Flop is a Combinational Logic Circuit. To store the data in digital form, the concepts of Flip-Flop came. We also know that a Flip-Flop can only store one-bit data. So if we want to store more than one-bit data then what to do? This problem is overcome by making Register. So Register is nothing but a group of Flip-flops which can store more than one-bit data. In digital electronics Logic Gates, flip-flops, registers are very important and interesting topics because they are the basic components of Microprocessor, CPU and Memory etc.

Register perform below work:

- Fetch
- Decode
- Execute

In this blog, we are always trying to give clear basic knowledge of the topics of electrical and electronics engineering.
To better understand the Register first read the below topics, As I told you the Flip-flop is the basic element of the Register so if we want to store four-bit data we need four-bit register. The four-bit register can be made by four flip-flops. A circuit diagram of a basic 4-bit register is given below,



Circuit diagram of Register

As you see in the above figure the four-bit register consists of four flip-flops. Each flip-flop can store one-bit data. Here D Flip-flops are used. You can also see that the clock terminal of each flip-flop is connected together because we give the clock pulse to all flip-flops together. Always remember that in the case of the register or any memory circuit using flip-flops, the clock pulse is always given together to all the flip-flops.
The concept of Register is very simple and you can understand easily.
***NOTE*: -** Now if I ask you that what is the physical component of a Register? You can give the answer to this question if you have clear basic knowledge about the Register, Flip-Flop, and Logic Gate.
But I would say. As we already know that the Registers are made by the combination of Flip-Flops, the Flip-flops are made by the combination of Logic Gates and the Logic gates are made by the combination of Transistors(Most of the cases transistors are used but Diode also may be used). So the physical components of registers are Transistors or we can say Registers are made by Transistors.

Types of Register and division : There are mainly two types of the register.

1. Memory address register (MAR)
2. Instruction register (IR)
3. Address register
4. Accumulator
5. Program counter (PC)
6. Memory buffer register (MBR)
7. Index register
8. Data register
9. Input output register

A. General Purpose Register

B. Special Purpose Register

- The General purpose registers are mainly stored data. The general purpose registers are,

1) Accumulator
2) Data Register
3) Counter Register

- The Special purpose registers mainly to hold the instructions or lines or states of a program before execution. The special purpose registers are,

1) Instruction Register
2) Program Counter

- **Memory address registers (MAR):**

Memory address register (MAR) - holds the address of the current instruction that is to be fetched from memory, or the address in memory to which data is to be transferred.

- **Instruction register (IR):**

In computing, the instruction registers (IR) or current instruction register (CIR) is the part of a CPU's control unit that holds the instruction currently being executed or decoded.

- **Address register:**

A high-speed circuit in a computing device that holds the addresses of data to be processed or of the next instruction to be executed.

- **Accumulator:**

An accumulator is a type of register for short-term, intermediate storage of arithmetic and logic data in a computer's central processing unit (CPU).

- **Program counter (PC):**

The program counter (PC) is a register that manages the memory address of the instruction to be executed next.

- **Memory buffer register (MBR):**

A Memory Buffer Register is the register in a computer's processor, or central processing unit, CPU, that stores the data being transferred to and from the immediate access store.

- **Index register:**

An index register in a computer's CPU is a processor register used for pointing to operand addresses during the run of a program.

- **Data register:**

The Memory Data Register (MDR) or Memory Buffer Register (MBR) is the register of a computer's control unit that contains the data to be stored in the computer storage (e.g. RAM), or the data after a fetch from the computer storage

- **Input output register:**

The Input Registers (IR) holds the input characters given by the user. The Output Registers (OR) holds the output after processing the input data.

Application of Register:

I. The main application of register is storing data in digital form.
II. They also can hold data and address
III. The registers are also used to make digital memory chips like ROM Chips, Flash Memory etc.
IV. Cache memory in CPU is also made by registers.

Shift Register:-

- The Shift Register is another type of sequential logic circuit that can be used for the storage or the transfer of binary data.
- A shift register basically consists of several single bit "D-Type Data Latches", one for each data bit, either a logic "0" or a "1", connected together in a serial arrangement so that the output from one data latch becomes the input of the next latch and so on.
- Data bits may be fed in or out of a shift register serially, that is one after the other from either the left or the right direction, or all together at the same time in a parallel configuration.

- The individual data latches that make up a single shift register are all driven by a common clock (Clk) signal making them synchronous devices.
- Generally, shift registers operate in one of four different modes with the basic movement of data through a shift register being:

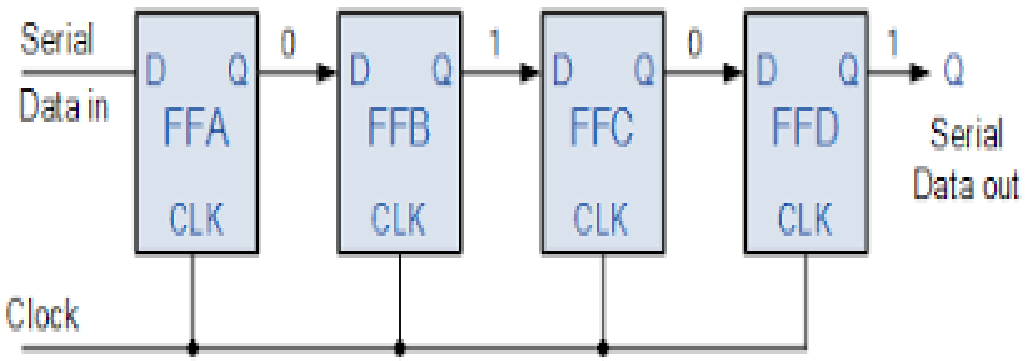
Mode	Clocks needed for n-bit shift register		
	Loading	Reading	Total
SISO	n	n-1	2n-1
SIPO	n	0	n
PISO	1	n-1	n
PIPO	1	0	1

1. Serial-in to Serial-out (SISO) - The data is shifted serially "IN"⁄ and "OUT" of the register, one bit at a time in either a left or right direction under clock control. Loading with n and reading with n-1.
2. .Serial-in to Parallel-out (SIPO) - The register is loaded with⁄ serial data, one bit at a time, with the stored data being available at the output in parallel form. Loading with n and reading with 0.
3. Parallel-in to Serial-out (PISO) - The parallel data is loaded⁄ into the register simultaneously and is shifted out of the register serially one bit at a time under clock control. Loading with 1 and reading with n-1.
4. Parallel-in to parallel-out (PIPO) - The parallel data is loaded⁄ simultaneously into the register, and transferred together to their respective outputs by the same clock pulse. Loading with 1 and reading with 0.

➢ Serial-In Serial-Out Shift Register (SISO) –slow

The shift register, which allows serial input (one bit after the other through a single data line) and produces a serial output is known as Serial-In Serial-Out shift register. Since there is only one output, the data leaves the shift register one bit at a time in a serial pattern, thus the name Serial-In Serial-Out Shift Register.

The logic circuit given below shows a serial-in serial-out shift register. The circuit consists of four D flip-flops which are connected in a serial manner. All these flip-flops are synchronous with each other since the same clock signal is applied to each flip flop.



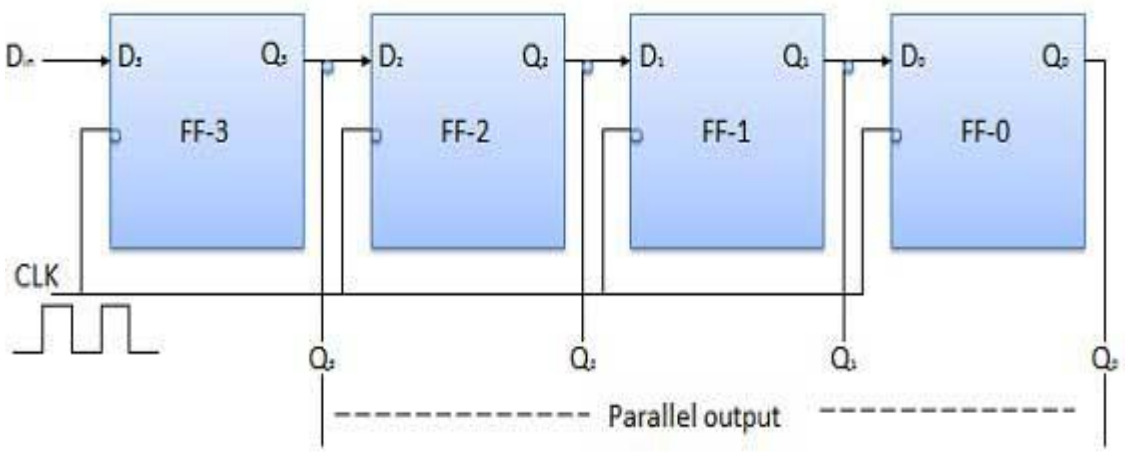
The above circuit is an example of shift right register, taking the serial data input from the left side of the flip flop. The main use of a SISO is to act as a delay element.

➢ Serial-In Parallel-Out shift Register (SIPO) –

The shift register, which allows serial input (one bit after the other through a single data line) and produces a parallel output, is known as Serial-In Parallel-Out shift register.

The logic circuit given below shows a serial-in-parallel-out shift register. The circuit consists of four D flip-flops which are connected. The clear (CLR)

signal is connected in addition to the clock signal to all the 4 flip flops in order to RESET them. The output of the first flip flop is connected to the input of the next flip flop and so on. All these flip-flops are synchronous with each other since the same clock signal is applied to each flip flop.



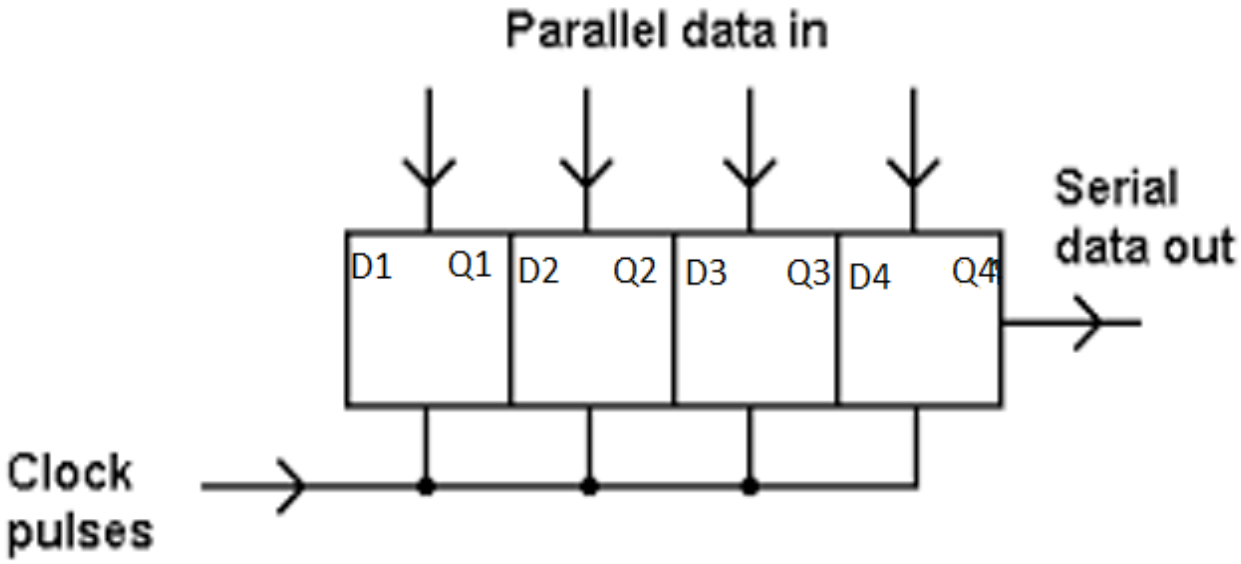
The above circuit is an example of shift right register, taking the serial data input from the left side of the flip flop and producing a parallel output. They are used in communication lines where demultiplexing of a data line into several parallel lines is required because the main use of the SIPO register is to convert serial data into parallel data.

- **Parallel-In Serial-Out Shift Register (PISO) –**

The shift register, which allows parallel input (data is given separately to each flip flop and in a simultaneous manner) and produces a serial output is known as Parallel-In Serial-Out shift register.

The logic circuit given below shows a parallel-in-serial-out shift register. The circuit consists of four D flip-flops which are connected. The clock input is directly connected to all the flip flops but the input data is connected individually to each flip flop through a multiplexer at the input of every flip flop. The output of the previous flip flop and parallel data input are connected to the input of the MUX and the output of MUX is connected to the next flip flop. All these flip-

flops are synchronous with each other since the same clock signal is applied to each flip flop.

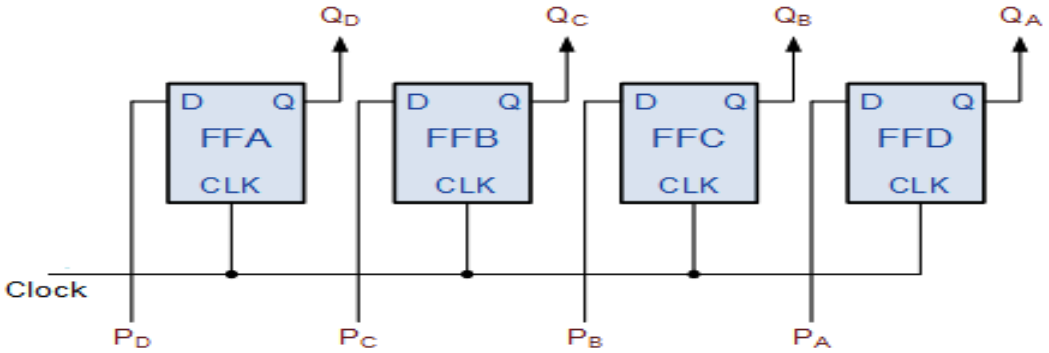


A Parallel in Serial out (PISO) shift register us used to convert parallel data to serial data.

- **Parallel-In Parallel-Out Shift Register (PIPO) –fast**

The shift register, which allows parallel input (data is given separately to each flip flop and in a simultaneous manner) and also produces a parallel output is known as Parallel-In parallel-Out shift register.

The logic circuit given below shows a parallel-in-parallel-out shift register. The circuit consists of four D flip-flops which are connected. The clear (CLR) signal and clock signals are connected to all the 4 flip flops. In this type of register, there are no interconnections between the individual flip-flops since no serial shifting of the data is required. Data is given as input separately for each flip flop and in the same way, output also collected individually from each flip flop.



A Parallel in Parallel out (PIPO) shift register is used as a temporary storage device and like SISO Shift register it acts as a delay element.

Applications of shift Registers –

The shift registers are used for temporary data storage.

The shift registers are also used for data transfer and data manipulation.

The serial-in serial-out and parallel-in parallel-out shift registers are used to produce time delay to digital circuits.

The serial-in parallel-out shift register is used to convert serial data into parallel data thus they are used in communication lines where demultiplexing of a data line into several parallel line is required.

A Parallel in Serial out shift register us used to convert parallel data to serial data.

Register Transfer Language (RTL):-

In symbolic notation, it is used to describe the micro-operations transfer among registers. It is a kind of intermediate representation (IR) that is very close to assembly language, such as that which is used in a compiler. The term "Register Transfer" can perform micro-operations and transfer the result of operation to the same or other register.

Micro-operations:

The operation executed on the data store in registers is called micro-operations.

They are detailed low-level instructions used in some designs to implement complex machine instructions.

Register Transfer:

The information transformed from one register to another register is represented in symbolic form by replacement operator is called Register Transfer.

Replacement Operator:

In the statement, R2 <- R1, <- acts as a replacement operator. This statement defines the transfer of content of register R1 into register R2.

- There are various methods of RTL –

General way of representing a register is by the name of the register enclosed in a rectangular box as shown in (a).
Register is numbered in a sequence of 0 to (n-1) as shown in (b).
The numbering of bits in a register can be marked on the top of the box as shown in (c).
A 16-bit register PC is divided into 2 parts- Bits (0 to 7) are assigned with lower byte of 16-bit address and bits (8 to 15) are assigned with higher bytes of 16-bit address as shown in (d).

Register Transfer Operations:

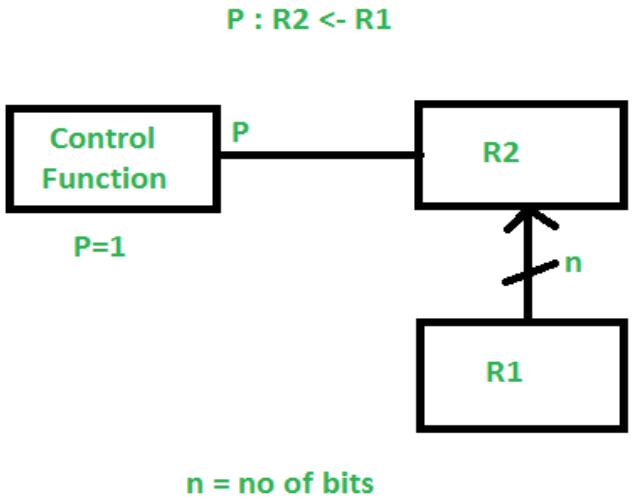
The operation performed on the data stored in the registers is referred to as register transfer operations.

There are different types of register transfer operations:

1. Simple Transfer – R2⇦R1

The content of R1 are copied into R2 without affecting the content of R1. It is an unconditional type of transfer operation.

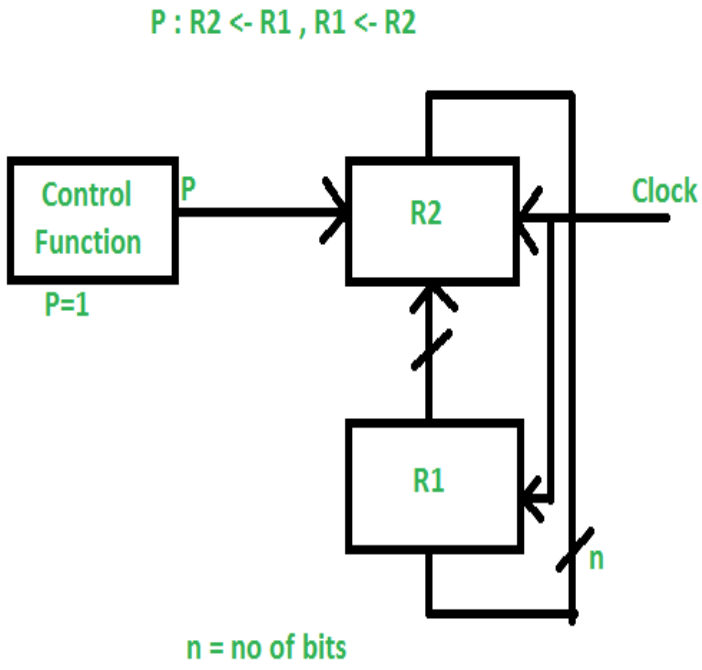
2. Conditional Transfer –



It indicates that if P=1, then the content of R1 is transferred to R2. It is a unidirectional operation.

3. Simultaneous Operations –

If 2 or more operations are to occur simultaneously then they are separated with comma (,).



If the control function P=1, then load the content of R1 into R2 and at the same clock load the content of R2 into R1.

Register Transfer:-

The term Register Transfer refers to the availability of hardware logic circuits that can perform a given micro-operation and transfer the result of the operation to the same or another register.

Most of the standard notations used for specifying operations on various registers are stated below.

- The memory address register is designated by MAR.
- Program Counter PC holds the next instruction's address.
- Instruction Register IR holds the instruction being executed.

R1 (Processor Register).

We can also indicate individual bits by placing them in parenthesis. For instance, PC (8-15), R2 (5), etc.

Data Transfer from one register to another register is represented in symbolic form by means of replacement operator. For instance, the following statement denotes a transfer of the data of register R1 into register R2.

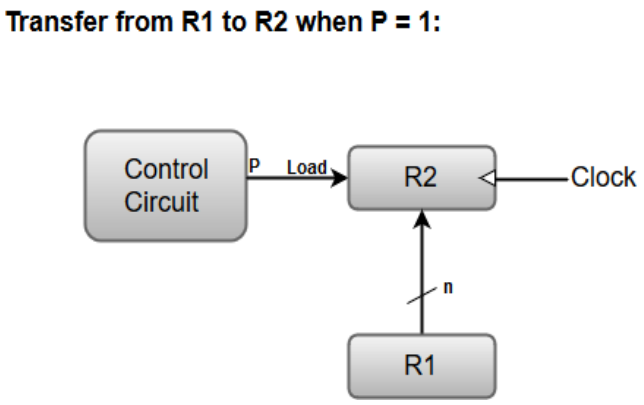
$$R2 \leftarrow R1$$

Typically, most of the users want the transfer to occur only in a predetermined control condition. This can be shown by following if-then statement: If (P=1) then (R2 ← R1); Here P is a control signal generated in the control section.

It is more convenient to specify a control function (P) by separating the control variables from the register transfer operation. For instance, the following statement defines the data transfer operation under a specific control function (P).

$$P: \ R2 \leftarrow R1$$

The following image shows the block diagram that depicts the transfer of data from R1 to R2.



Here, the letter 'n' indicates the number of bits for the register. The 'n' outputs of the register R1 are connected to the 'n' inputs of register R2.

A load input is activated by the control variable 'P' which is transferred to the register R2.

Bus and Memory Transfers:-

A digital system composed of many registers, and paths must be provided to transfer information from one register to another. The number of wires connecting all of the registers will be excessive if separate lines are used between each register and all other registers in the system.

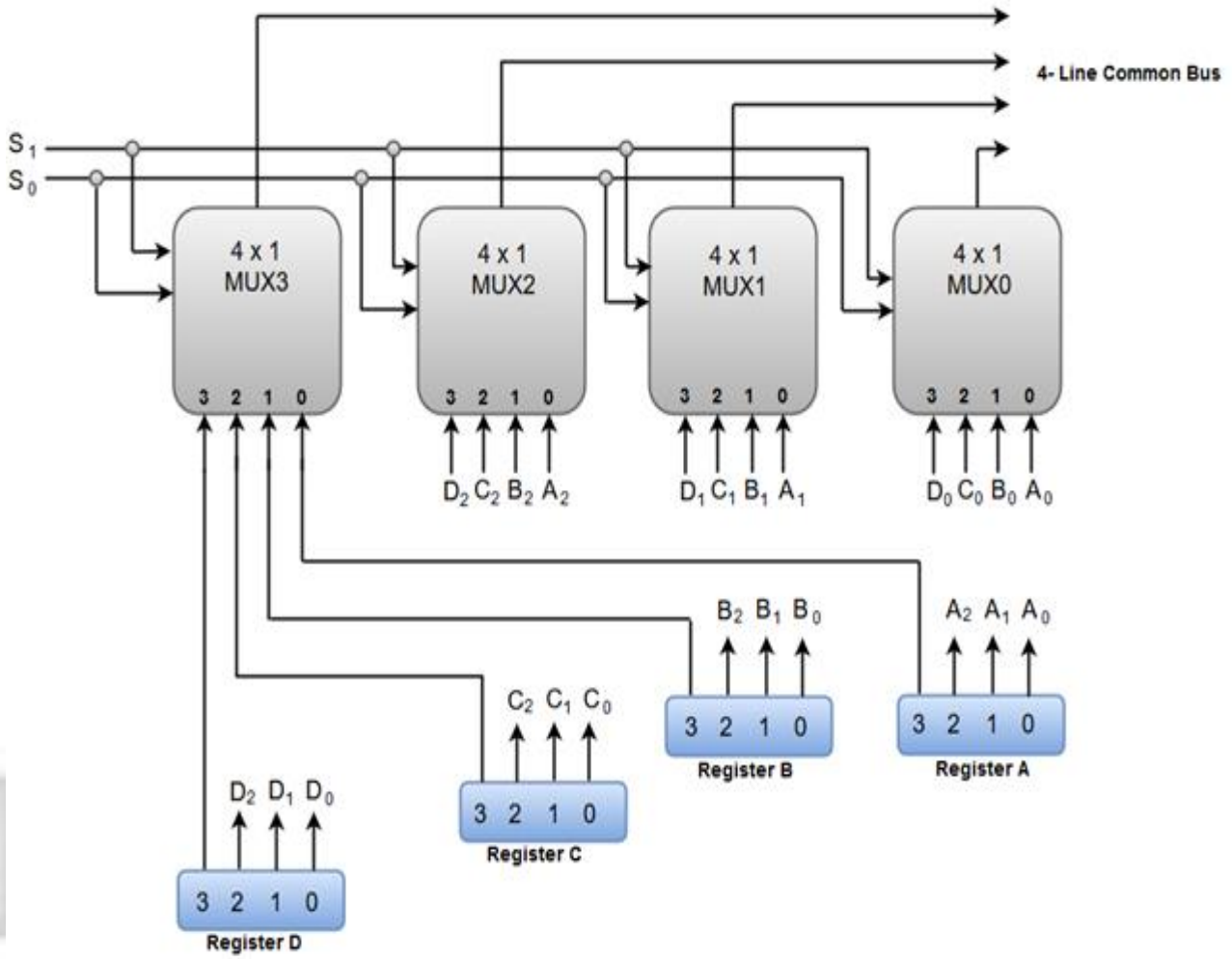
A bus structure, on the other hand, is more efficient for transferring information between registers in a multi-register configuration system.

A bus consists of a set of common lines, one for each bit of register, through which binary information is transferred one at a time. Control signals determine which register is selected by the bus during a particular register transfer.

The following block diagram shows a Bus system for four registers. It is constructed with the help of four 4 * 1 Multiplexers each having four data inputs (0 through 3) and two selection inputs (S1 and S2).

We have used labels to make it more convenient for you to understand the input-output configuration of a Bus system for four registers. For instance, output 1 of register A is connected to input 0 of MUX1.

Bus System for 4 Registers:



The two selection lines S1 and S2 are connected to the selection inputs of all four multiplexers. The selection lines choose the four bits of one register and transfer them into the four-line common bus.

When both of the select lines are at low logic, i.e. S1S0 = 00, the 0 data inputs of all four multiplexers are selected and applied to the outputs that forms the bus. This, in turn, causes the bus lines to receive the content of register A since the outputs of this register are connected to the 0 data inputs of the multiplexers.

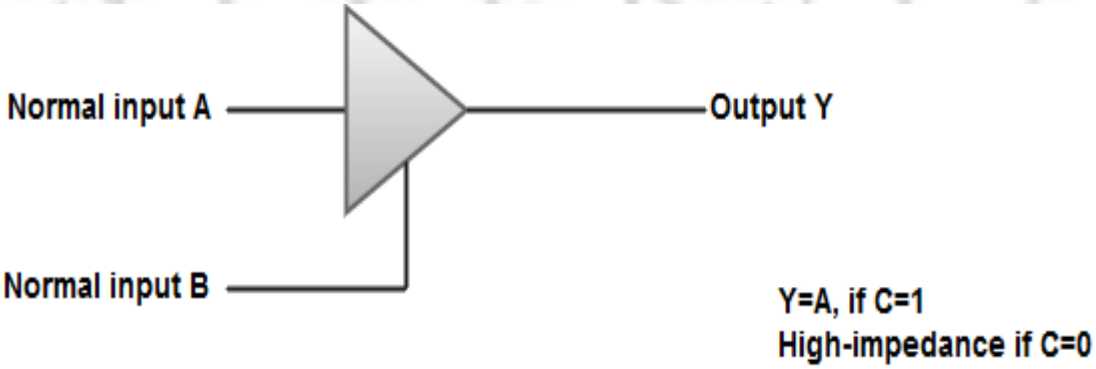
Similarly, when S1S0 = 01, register B is selected, and the bus lines will receive the content provided by register B.

The following function table shows the register that is selected by the bus for each of the four possible binary values of the Selection lines.

S1	S0	Register Selected
0	0	A
0	1	B
1	0	C
1	1	D

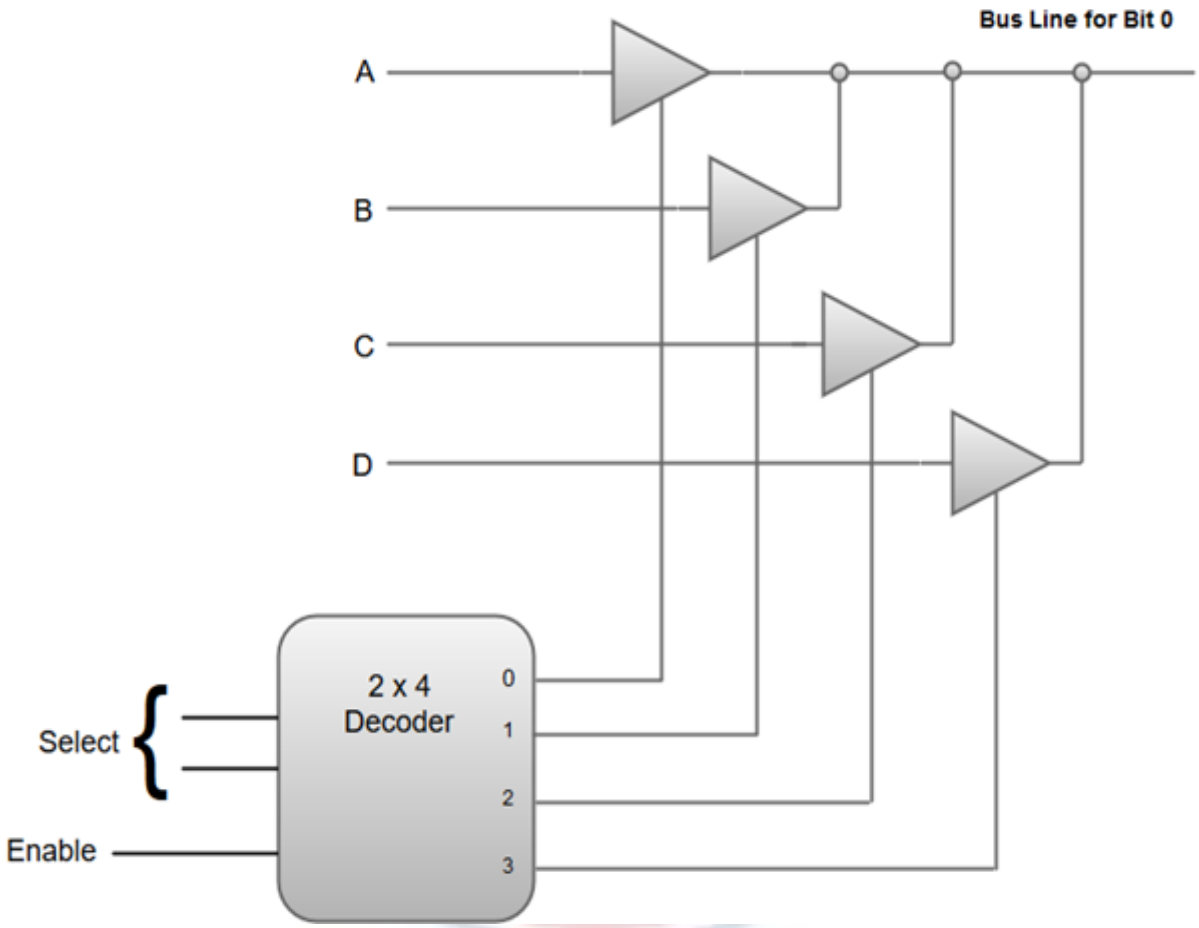
A bus system can also be constructed using three-state gates instead of multiplexers. The three state gates can be considered as a digital circuit that has three gates, two of which are signals equivalent to logic 1 and 0 as in a conventional gate. However, the third gate exhibits a high-impedance state.

The most commonly used three state gates in case of the bus system is a buffer gate. The graphical symbol of a three-state buffer gate can be represented as:



The following diagram demonstrates the construction of a bus system with three-state buffers.

Bus line with three state buffer:



- The outputs generated by the four buffers are connected to form a single bus line.
- Only one buffer can be in active state at a given point of time.
- The control inputs to the buffers determine which of the four normal inputs will communicate with the bus line.
- A 2 * 4 decoder ensures that no more than one control input is active at any given point of time.

Memory Transfer

Most of the standard notations used for specifying operations on memory transfer are stated below.

The transfer of information from a memory unit to the user end is called a **Read operation**.

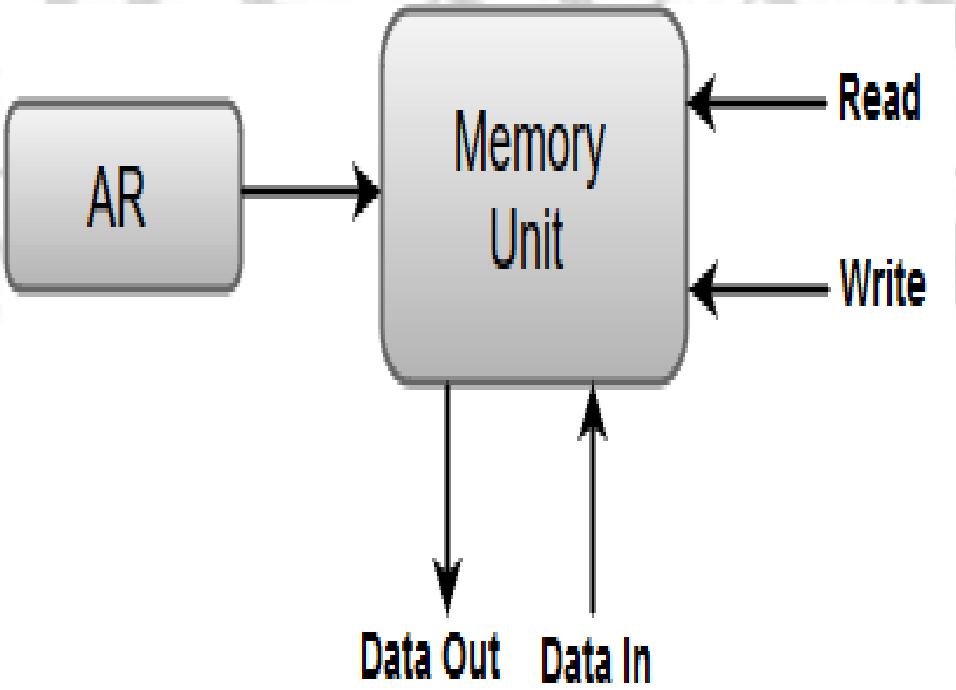
- The transfer of new information to be stored in the memory is called a **Write operation.**
- A memory word is designated by the letter **M**.
- We must specify the address of memory word while writing the memory transfer operations.
- The **address register** is designated by **AR** and the **data register** by **DR.**
- Thus, a read operation can be stated as:

  Read: DR ← M [AR]

- The **Read statement** causes a transfer of information into the data register (DR) from the memory word (M) selected by the address register (AR).
- And the corresponding write operation can be stated as:

  Write: M [AR] ← R1

- The Write statement causes a transfer of information from register R1 into the memory word (M) selected by address register (AR).



Arithmetic Micro-operations

In general, the Arithmetic Micro-operations deals with the operations performed on numeric data stored in the registers.

The basic Arithmetic Micro-operations are classified in the following categories:

1. Addition
2. Subtraction
3. Increment
4. Decrement
5. Shift

The following table shows the symbolic representation of various Arithmetic Micro-operations.

Symbolic Representation	Description
R3 ← R1 + R2	The contents of R1 plus R2 are transferred to R3.
R3 ← R1 - R2	The contents of R1 minus R2 are transferred to R3.
R2 ← R2'	Complement the contents of R2 (1's complement)
R2 ← R2' + 1	2's complement the contents of R2 (negate)
R3 ← R1 + R2' + 1	R1 plus the 2's complement of R2 (subtraction)
R1 ← R1 + 1	Increment the contents of R1 by one
R1 ← R1 - 1	Decrement the contents of R1 by one

✍ **Note:** The increment and decrement micro-operations are symbolized by '+ 1' and '? 1' respectively. Arithmetic operations like multiply and divide are not included in the basic set of micro-operations.

Binary Adder

The Add micro-operation requires registers that can hold the data and the digital components that can perform the arithmetic addition.

A Binary Adder is a digital circuit that performs the arithmetic sum of two binary numbers provided with any length.

A Binary Adder is constructed using full-adder circuits connected in series, with the output carry from one full-adder connected to the input carry of the next full-adder.

The following block diagram shows the interconnections of four full-adder circuits to provide a 4-bit binary adder.

In addition micro-operation, the value in register R1 is added to the value in the register R2 and then the sum is transferred into register R3.

$$R3 \leftarrow R1 + R2$$

The augend bits (A) and the addend bits (B) are designated by subscript numbers from right to left, with subscript '0' denoting the low-order bit.

- The carry inputs starts from C0 to C3 connected in a chain through the full-adders. C4 is the resultant output carry generated by the last full-adder circuit.
- The output carry from each full-adder is connected to the input carry of the next-high-order full-adder.
- The sum outputs (S0 to S3) generates the required arithmetic sum of augend and addend bits.
- The n data bits for the A and B inputs come from different source registers. For instance, data bits for A input comes from source register R1 and data bits for B input comes from source register R2.

- The arithmetic sum of the data inputs of A and B can be transferred to a third register or to one of the source registers (R1 or R2).

Binary Adder-Subtractor

The Subtraction micro-operation can be done easily by taking the 2's compliment of addend bits and adding it to the augend bits.

In subtraction micro-operation, the contents of register R2 are subtracted from contents of the register R1, and then the result is transferred into R3.

$$R3 \longleftarrow R1 - R2$$

There is another way of doing the subtraction. In this, 2's complement of R2 is added to R1, which is equivalent to R1 – R2, and then the result is transferred into register R3.

$$R3 \longleftarrow R1 + \overline{R2} + 1$$

- **Note:** The 2's compliment can be obtained by taking the 1's compliment and adding one to the least significant pair of bits. The 1's compliment can be implemented with inverters, and one can be added to the sum through the input carry.

The Arithmetic micro-operations like addition and subtraction can be combined into one common circuit by including an exclusive-OR gate with each full adder.

The block diagram for a 4-bit adder-sub tractor circuit can be represented as:

- When the mode input (M) is at a low logic, i.e. '0', the circuit act as an adder and when the mode input is at a high logic, i.e. '1', the circuit act as a subtractor.

- The exclusive-OR gate connected in series receives input M and one of the inputs B.
- When M is at a low logic, we have B⊕ 0 = B. The full-adders receive the value of B, the input carry is 0, and the circuit performs A plus B.
- When M is at a high logic, we have B⊕ 1 = B' and C0 = 1. The B inputs are complemented, and a 1 is added through the input carry. The circuit performs the operation A plus the 2's complement of B.

Binary Incrementer

The increment micro-operation adds one binary value to the value of binary variables stored in a register. For instance, a 4-bit register has a binary value 0110, when incremented by one the value becomes 0111.

The increment micro-operation is best implemented by a 4-bit combinational circuit incrementer. A 4-bit combinational circuit incrementer can be represented by the following block diagram.

In Increment micro-operation, the value inside the R1 register is increased by 1.

$$R1 \leftarrow R1 + 1$$

- A logic-1 is applied to one of the inputs of least significant half-adder, and the other input is connected to the least significant bit of the number to be incremented.
- The output carry from one half-adder is connected to one of the inputs of the next-higher-order half-adder.
- The binary incrementer circuit receives the four bits from A0 through A3, adds one to it, and generates the incremented output in S0 through S3.
- The output carry C4 will be 1 only after incrementing binary 1111.

✍ **Note:** The 4-bit binary incrementer circuit can be extended to an n-bit binary incrementer by extending the circuit to include n half-adders. The

least significant bit must have one input connected to logic-1. The other inputs receive the number to be incremented or the carry from the previous stage.

Decrement–

In Decrement micro-operation, the value inside the R1 register is decreased by 1.

1's Complement –

In this micro-operation, the complement of the value inside the register R1 is taken.

2'sComplement–

In this micro-operation, the complement of the value inside the register R2 is taken and then 1 is added to the value and then the final result is transferred into the register R2. This process is also called Negation. It is equivalent to -R2.

Shift micro-operations

Shift micro-operations are those micro-operations that are used for serial transfer of information. These are also used in conjunction with arithmetic micro-operation, logic micro-operation, and other data-processing operations.

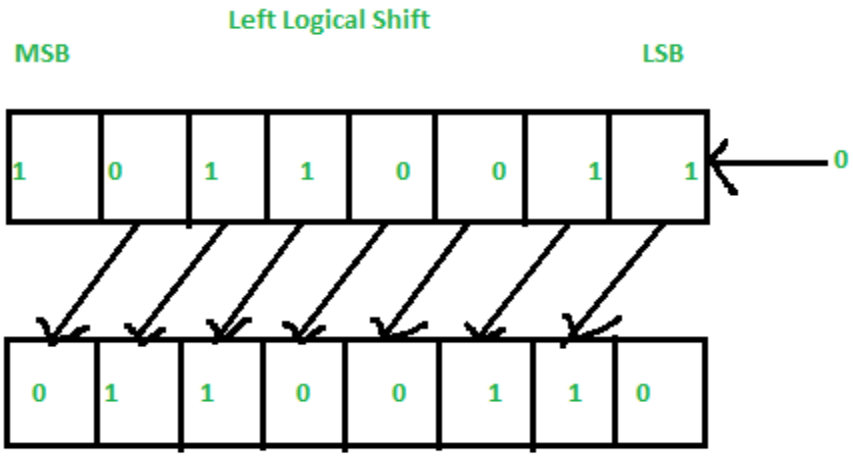
There are three types of shifts micro-operations:

1.Logical:

It transfers the 0 zero through the serial input. We use the symbols shl for logical shift-left and shr for shift-right.

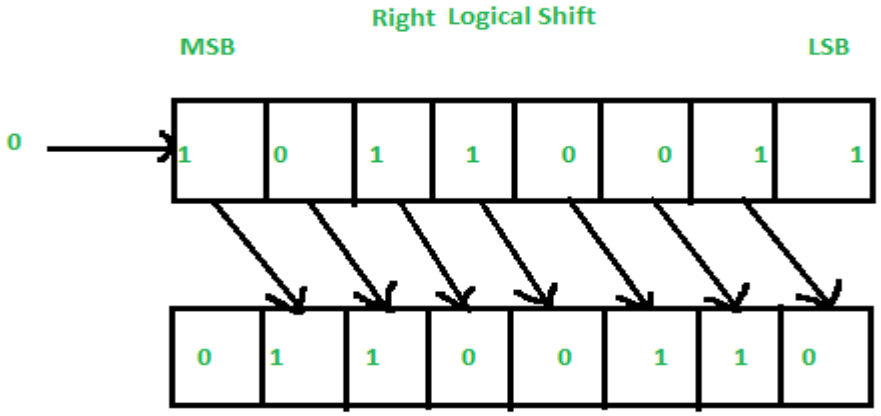
1. Logical Shift Left –

In this shift one position moves each bit to the left one by one. The Empty least significant bit (LSB) is filled with zero (i.e, the serial input), and the most significant bit (MSB) is rejected.



2. **Right Logical Shift –**

   In this one position moves each bit to the right one by one and the least significant bit(LSB) is rejected and the empty MSB is filled with zero.

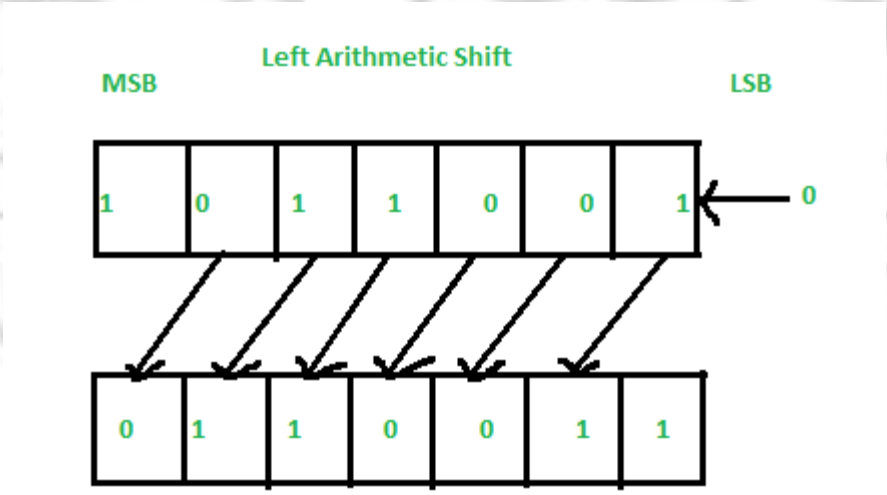


2.Arithmetic:

This micro-operation shifts a signed binary number to the left or to the right position. In arithmetic shift-left, it multiplies a signed binary number by 2 and In an arithmetic shift-right, it divides the number by 2.

1. Left Arithmetic Shift –

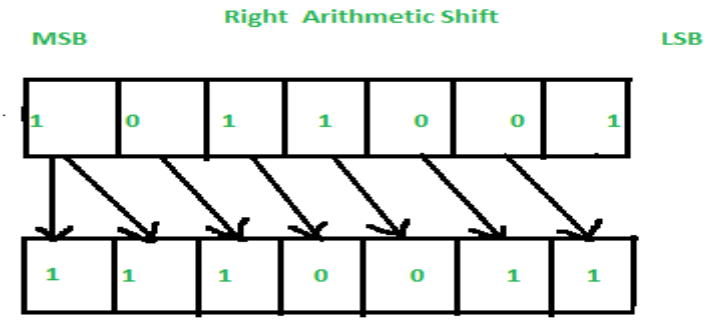
In this one position moves each bit to the left one by one. The empty least significant bit (LSB) is filled with zero and the most significant bit (MSB) is rejected. Same as the Left Logical Shift.



RightArithmeticShift.–

In this one position moves each bit to the right one by one and the least

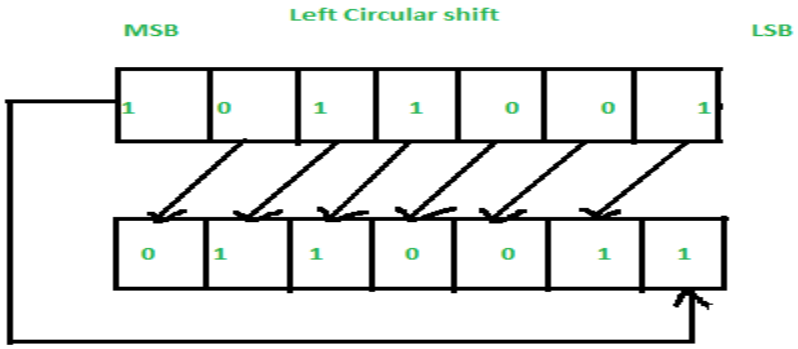
significant bit is rejected and the empty MSB is filled with the value of the previous MSB.



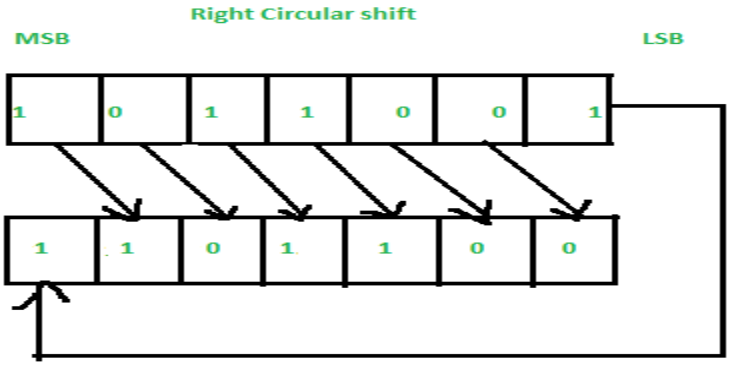
3. Circular :.-

The circular shift circulates the bits in the sequence of the register around the both ends without any loss of information.

Left Circular Shift –



Right Circular Shift –



Thank You