Laboratory manual

Digital Electronics Laboratory

Faculty Laboratory Incharge : Er. Diljinder Singh

Technician Laboratory Incharge : Mr. Sukhminder Singh

Electrical and Instrumentation Engineering department

Sant Longowal Institute of Engineering and Technology, Longowal-148106, Punjab

Subject Code : PCIE-524
Title of the course : Digital Electronics Lab.
Lab Manual Created By : Dr. M. S. Manna

L	T	P	Credits	Weekly Load
0	0	2	1	2

Course Outcomes:
After successful completion of course, the students should be able to
CO 1: Exercise and verify truth tables of TTL gates, universal gates.
CO 2: Design and verify truth tables of Half and Full adder, subtractor circuits.
CO 3: Verify truth tables of Multiplexer 74150 and De-Multiplexer 74154.
CO 4: Design and verify truth tables of S-R (NOR/NAND gates based), J-K and D flip flops
CO 5: Operate counters and design4 bit shift register, modulo-4 counter using J K flip flop

CO/PO Mapping : (Strong(S) / Medium(M) / Weak(W) indicates strength of correlation):												
COs	Programme Outcomes (POs)											
	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12
CO1	S											M
CO2		S	M									
CO3	S			M								
CO4	S			M	M							
CO5	S	S			M							M

To understand the practicability of Digital Electronics, the list of experiments is given below to be performed (atleast 10) in the laboratory.

1. Verification of the truth tables of TTL gates.
2. Verify the NAND and NOR gates as universal logic gates.
3. Design and verification of the truth tables of Half and Full adder circuits.
4. Design and verification of the truth tables of Half and Full subtractor circuits.
5. Design and test of an S-R flip-flop using NOR/NAND gates.
6. Verify the truth table of a J-K flip-flop (7476)
7. Verify the truth table of a D flip-flop (7474)
8. Verification of the truth table of the Multiplexer 74150.
9. Verification of the truth table of the De-Multiplexer 74154.
10. Operate the counters 7490, 7493.
11. Design of 4 bit shift register (shift right).
12. Design of modulo-4 counter using J K flip flop.

Digital Electronics Lab.		
Sr. No.	**Do's**	**Don'ts**
1	Maintain strict discipline.	Do not touch or attempt to touch the mains power directly with bare hands.
2	Proper handling of apparatus must be done.	Do not manipulate the experiment results.
3	Before switching on the power supply get it checked by the technician.	Do not overcrowd the tables.
4	Switch off your mobile phone.	Do not tamper with equipments.
5	Be a keen observer while performing the experiment	Do not leave the lab without prior permission from the teacher.

Experiment No. 1

Verification and interpretation of truth table for AND, OR, NOT, NAND, NOR, Ex-OR, Ex-NOR gates.

OR

Verification of the truth tables of TTL gates.

Aim:

To verify and interpret the logic and truth table for AND, OR, NOT, NAND, NOR, Ex-OR, Ex-NOR gates using RTL (Resistor Transistor Logic), DTL (Diode Transistor Logic) and TTL (Transistor Transistor Logic) logics in simulator 1 and verify the truth table for AND, OR, NOT, NAND, NOR, Ex-OR, Ex-NOR gates in simulator 2.

Theory:

Introduction

Logic gates are the basic building blocks of any digital system. Logic gates are electronic circuits having one or more than one input and only one output. The relationship between the input and the output is based on a certain logic. Based on this, logic gates are named as

1) AND gate
2) OR gate
3) NOT gate
4) NAND gate
5) NOR gate
6) Ex-OR gate
7) Ex-NOR gate

1) AND gate:

The AND gate is an electronic circuit that gives a high output (1) only if all its inputs are high. A dot (.) is used to show the AND operation i.e. A.B or can be written as AB

$$Y = A.B$$

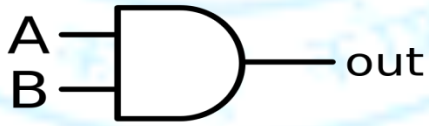


Figure-1:Logic Symbol of AND Gate

Input		Output
A	B	Y=A.B
0	0	0
0	1	0
1	0	0
1	1	1

Figure-2:Truth Table of AND Gate

A simple 2-input logic AND gate can be constructed using RTL (Resistor-Transistor-Logic) switches connected together as shown below with the inputs connected directly to the transistor bases. Both transistors must be saturated "ON" for an output at Q.

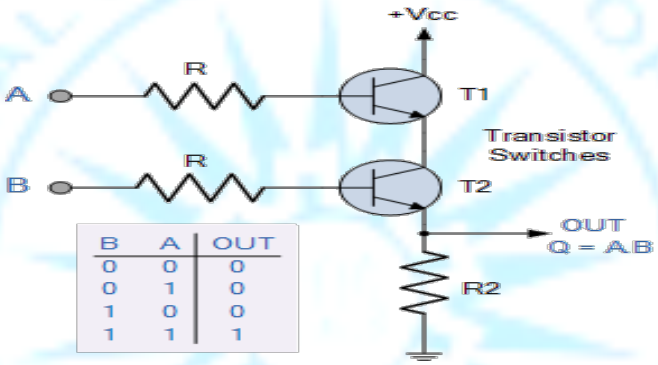


Figure-3:AND Gate through RTL logic

2) OR gate

The OR gate is an electronic circuit that gives a high output (1) if one or more of its inputs are high. A plus (+) is used to show the OR operation.

Y= A+B

Figure-4:Logic Symbol of OR Gate

Input		Output
A	B	Y=A+B
0	0	0
0	1	1
1	0	1
1	1	1

Figure-5:Truth Table of OR Gate

OR gate can be realized by DRL (Diode-Resistance-Logic) or by TTL (Transistor-Transistor-Logic). Presently, we will learn how to implement the OR gate using DRL (Diode-Resistance-Logic). To realise OR gate, we will use a diode at every input of the OR gate. The anode part of diode is connected with input while the cathode part is joined together and a resistor, connected with the cathode is grounded. In this case, we have taken two inputs which can be seen in the circuit below.

When both the inputs are at logic 0 or low state then the diodes D1 and D2 become reverse biased. Since the anode terminal of diode is at lower voltage level than the cathode terminal, so diode will act as open circuit so there is no voltage across resistor and hence output voltage is same as ground. When either of the diodes is at logic 1 or high state then the diode corresponding to that input is forward bias. Since this time anode is at high voltage than cathode therefore current will flow through forward biased diode and this current then appears on resistor causing high voltage at output terminal also. Hence at output we get high or logic 1 or +5V. So, if any or both inputs are high, the output will be high or "1".

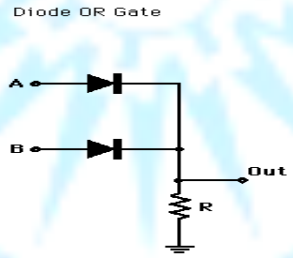


Figure-6: OR Gate through DRL logic

3) NOT gate

The NOT gate is an electronic circuit that produces an inverted version of the input at its output. It is also known as an inverter. If the input variable is A, he inverted output is known as NOT A. This is also shown as A' or A with a bar over the top, as shown at the outputs.

$$Y = A'$$

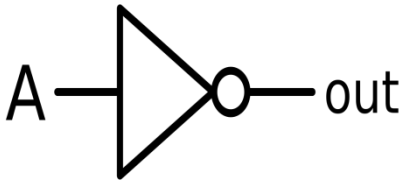


Figure-7: Logic Symbol of NOT Gate

Input	Output
A	Y
0	1
1	0

Figure-8:Truth Table of NOT Gate

NOT gate can be realized through transistor.The input is connected through resistor R2 to the transistor's base. When no voltage is present on the input, the transistor turns off. When the transistor is off, no current flows through the collector-emitter path. Thus, current from the supply voltage (Vcc) flows through resistor R1 to the output. In this way, the circuit's output is high when its input is low.

When voltage is present at the input, the transistor turns on, allowing current to flow through the collector-emitter circuit directly to ground. This ground path creates a shortcut that bypasses the output, which causes the output to go low.

In this way, the output is high when the input is low and low when the input is high.

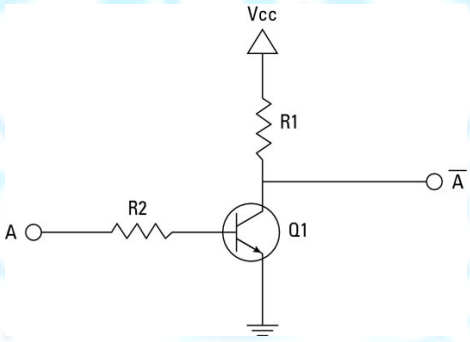


Figure-9:NOT Gate through Transistor

4) NAND gate:

This is a NOT-AND gate which is equal to an AND gate followed by a NOT gate. The outputs of all NAND gates are high if any of the inputs are low. The symbol is an AND gate with a small circle on the output. The small circle represents inversion.

$$Y = AB$$

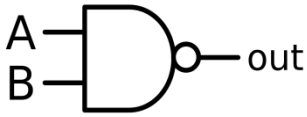


Figure-10:Logic Symbol of NAND Gate

Input	Input	Output
A	B	Y
0	0	1
0	1	1
1	0	1
1	1	0

Figure-11:Truth Table of NAND Gate

A simple 2-input logic NAND gate can be constructed using RTL (Resistor-transistor-logic) switches connected together as shown below with the inputs connected directly to the transistor bases. Either transistor must be cut-off or "OFF" for an output at Q.

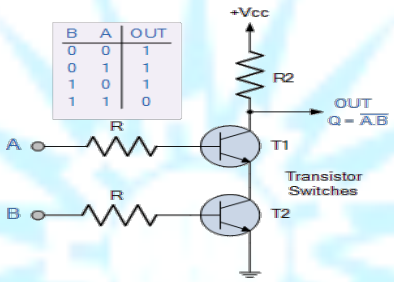


Figure-12:NAND gate through RTL Logic.

5) NOR gate:

This is a NOT-OR gate which is equal to an OR gate followed by a NOT gate. The outputs of all NOR gates are low if any of the inputs are high. The symbol is an OR gate with a small circle on the output. The small circle represents inversion.

$$Y = A+B$$

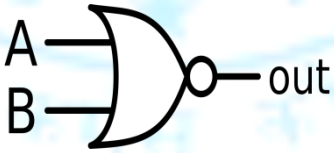


Figure-13:Logic Symbol of NOR gate

A	B	F
0	0	1
0	1	0
1	0	0
1	1	0

Figure-14:Truth Table of NOR gate

A simple 2-input logic NOR gate can be constructed using RTL (Resistor-transistor-logic) switches connected together as shown below with the inputs connected directly to the transistor bases. Both transistors must be cut-off or "OFF" for an output at Q.

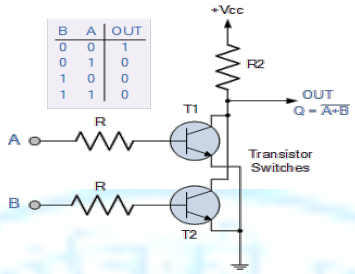


Figure-15:NOR gate through RTL Logic.

6) Ex-OR gate

The 'Exclusive-OR' gate is a circuit which will give a high output if either, but not both of its two inputs are high. An encircled plus sign (⊕) is used to show the Ex-OR operation.

$$Y = A \oplus B$$

Figure-16:Logic Symbol of Ex-OR gate

A	B	A **XOR** B
0	0	0
0	1	1
1	0	1
1	1	0

Figure-17:Truth Table of Ex-OR gate

Ex-OR gate is created from AND, NAND and OR gates.The output is high only when both the inputs are different.

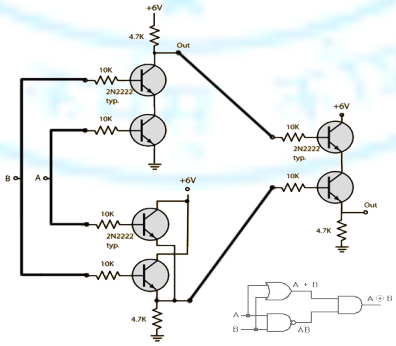


Figure-18:Ex-OR gate through RTL Logic.

7) Ex-NOR gate:

The 'Exclusive-NOR' gate circuit does the opposite to the EX-OR gate. It will give a low output if either, but not both of its two inputs are high. The symbol is an EX-OR gate with a small circle on the output. The small circle represents inversion.

$$Y = A \oplus B$$

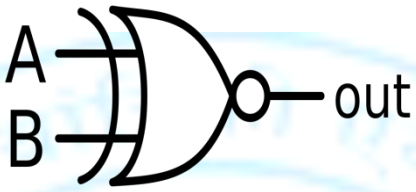


Figure-19:Logic Symbol of Ex-NOR gate

XNOR Truth Table		
A	B	Q
0	0	1
0	1	0
1	0	0
1	1	1

Figure-20:Truth Table of Ex-NOR gate

Ex-NOR gate is created from AND, NOT and OR gates.The output is high only when both the inputs are same.

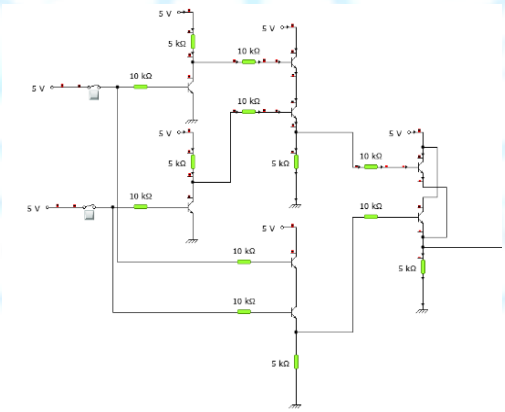


Figure-21:Ex-NOR gate through RTL Logic.

Experiment No. 2

Verify the NAND and NOR gates as universal logic gates.

Aim:

To implement the logic functions i.e. AND, OR, NOT, Ex-OR, Ex- NOR and a logical expression with the help of NAND and NOR universal gates respectively

Theory:

Introduction:

Logic gates are electronic circuits which perform logical functions on one or more inputs to produce one output. There are seven logic gates. When all the input combinations of a logic gate are written in a series and their corresponding outputs written along them, then this input/ output combination is called Truth Table.

1)Nand gate as Universal gate:

NAND gate is actually a combination of two logic gates i.e. AND gate followed by NOT gate. So its output is complement of the output of an AND gate. This gate can have minimum two inputs. By using only NAND gates, we can realize all logic functions: AND, OR, NOT, Ex-OR, Ex-NOR, NOR. So this gate is also called as universal gate.

1.1)NAND gates as OR gate

From DeMorgan's theorems:
(A.B)' = A' + B'
(A'.B')' = A'' + B'' = A + B
So, give the inverted inputs to a NAND gate, obtain OR operation at output.

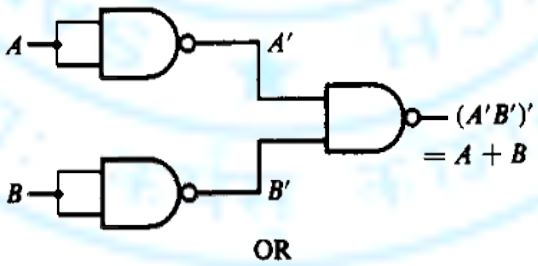


Figure-1:NAND gates as OR gate

A	B	Y
0	0	0
0	1	1
1	0	1
1	1	1

Figure-2:Truth table of OR

1.2)NAND gates as AND gate:

A NAND produces complement of AND gate. So, if the output of a NAND gate is inverted, overall output will be that of an AND gate.

$Y = ((A.B)')'$

$Y = (A.B)$

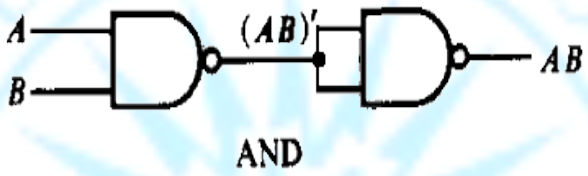


Figure-3:NAND gates as AND gate

Input		Output
A	B	Y = A.B
0	0	0
0	1	0
1	0	0
1	1	1

Figure-4:Truth table of AND

1.3)NAND gates as Ex-OR gate:

The output of a two input Ex-OR gate is shown by: Y = A'B + AB'. This can be achieved with the logic diagram shown in the left side.

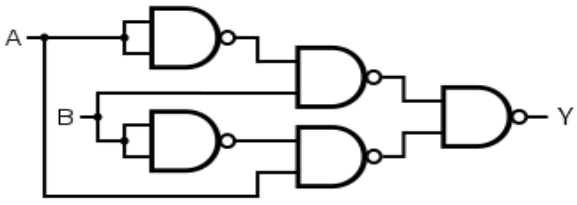


Figure-5:NAND gate as Ex-OR gate

A	B	A **XOR** B
0	0	0
0	1	1
1	0	1
1	1	0

Figure-6:Truth table of Ex-OR

1.4)NAND gates as Ex-NOR gate:

Ex-NOR gate is actually Ex-OR gate followed by NOT gate. So give the output of Ex-OR gate to a NOT gate, overall output is that of an Ex-NOR gate.

Y = AB+ A'B'

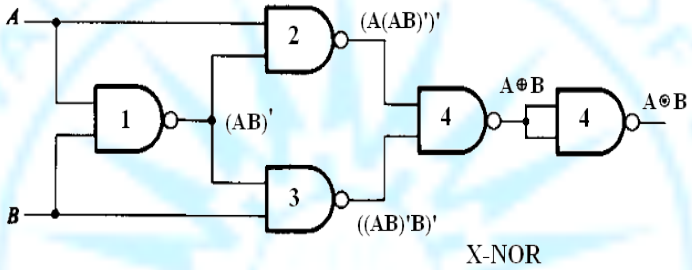


Figure-7:NAND gates as Ex-NOR gate

A	B	Y
0	0	1
0	1	0
1	0	0
1	1	1

Figure-8:Truth table of Ex-NOR

1.5) Implementing the simplified function with NAND gates only

We can now start constructing the circuit. First note that the entire expression is inverted and we have three terms ANDed. This means that we must use a 3-input NAND gate. Each of the three terms is, itself, a NAND expression. Finally, negated single terms can be generates with a 2-input NAND gate acting as an inverted. Figure 8 illustrates a circuit using NAND gates only.

F=((C'.B.A)'(D'.C.A)'(C.B'.A)')'

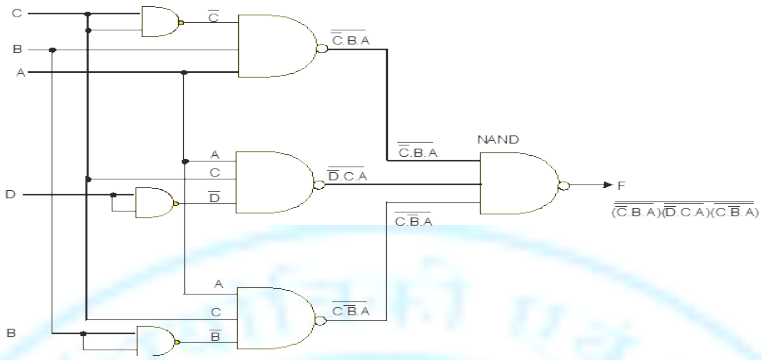


Figure-9:Implementing the simplified function with NAND gates only

2) Nor gate as Universal Gate

NOR gate is actually a combination of two logic gates: OR gate followed by NOT gate. So its output is complement of the output of an OR gate.This gate can have minimum two inputs, output is always one. By using only NOR gates, we can realize all logic functions: AND, OR, NOT, Ex-OR, Ex-NOR, NAND. So this gate is also called universal gate.

2.1) NOR gates as OR gate

A NOR produces complement of OR gate. So, if the output of a NOR gate is inverted, overall output will be that of an OR gate.

$$Y = ((A+B)')'$$
$$Y = (A+B)$$

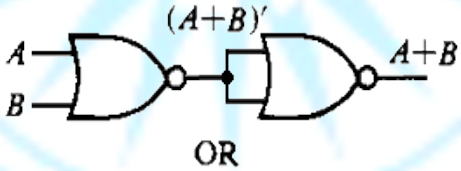


Figure-10:NOR gates as OR gate

A	B	Y
0	0	0
0	1	1
1	0	1
1	1	1

Figure-11:Truth table of OR

2.2)NOR gates as AND gate:

From DeMorgan's theorems:
(A+B)' = A'B'
(A'+B')' = A''B'' = AB
So, give the inverted inputs to a NOR gate, obtain AND operation at output.

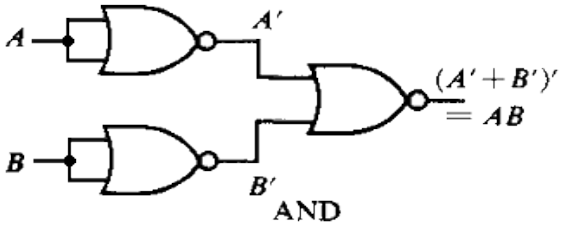


Figure-12:NOR gates as AND gate

Input		Output
A	B	Y = A.B
0	0	0
0	1	0
1	0	0
1	1	1

Figure-13:Truth table of AND

2.4)NOR gates as Ex-OR gate:

Ex-OR gate is actually Ex-NOR gate followed by NOT gate. So give the output of Ex-NOR gate to a NOT gate, overall output is that of an Ex-OR gate.
Y = A'B+ AB'

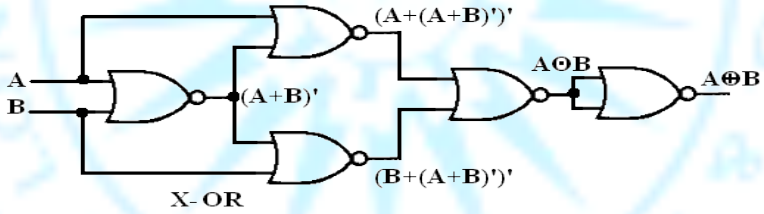


Figure-14:NOR gates as Ex-OR gate

A	B	A XOR B
0	0	0
0	1	1
1	0	1
1	1	0

Figure-15:Truth table of Ex-OR

2.3)NOR gates as Ex-NOR gate:

The output of a two input Ex-NOR gate is shown by: Y = AB + A'B'. This can be achieved with the logic diagram shown in the left side.

Gate No.	Inputs	Output
1	A, B	(A + B)'
2	A, (A + B)'	(A + (A+B)')'
3	(A + B)', B	(B + (A+B)')'
4	(A + (A + B)')', (B + (A+B)')'	AB + A'B'

Now the ouput from gate no. 4is the overall output of the configuration.

$$
\begin{aligned}
Y &= ((A + (A+B)')' (B +(A+B)')')' \\
&= (A+(A+B)')''.(B+(A+B)')'' \\
&= (A+(A+B)').(B+(A+B)') \\
&= (A+A'B').(B+A'B') \\
&= (A + A').(A + B').(B+A')(B+B') \\
&= 1.(A+B').(B+A').1 \\
&= (A+B').(B+A') \\
&= A.(B + A') +B'.(B+A') \\
&= AB + AA' +B'B+B'A' \\
&= AB + 0 + 0 + B'A' \\
&= AB + B'A' \\
\Rightarrow \quad Y &= AB + A'B'
\end{aligned}
$$

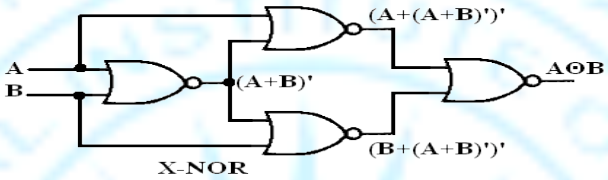


Figure-16:NOR gates as Ex-NOR gate

A	B	Y
0	0	1
0	1	0
1	0	0
1	1	1

Figure-17:Truth table of Ex-NOR

2.5)Constructing a circuit with NOR gates only:

Designing a circuit with NOR gates only uses the same basic techniques as designing a circuit with NAND gates; that is, the application of deMorgan's theorem. The only difference between NOR gate design and NAND gate design is that the former must eliminate product terms and the later must eliminate sum terms.

F=(((C.B'.A)+(D.C'.A)+(C.B'.A))')'

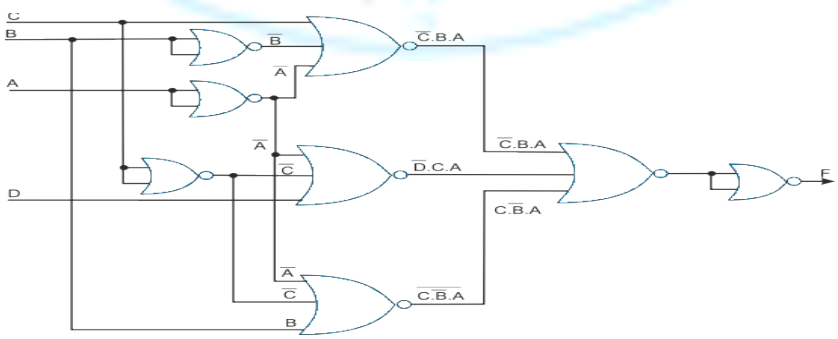


Figure-18:Implementing the simplified function with NOR gates only

Experiment No. 3

Design and verification of the truth tables of Half and Full adder circuits.

Aim

To verify the truth table of half adder and full adder by using XOR and NAND gates respectively and analyse the working of half adder and full adder circuit with the help of LEDs in simulator 1 and verify the truth table only of half adder and full adder in simulator 2.

Theory

Introduction

Adders are digital circuits that carry out addition of numbers. Adders are a key component of arithmetic logic unit. Adders can be constructed for most of the numerical representations like Binary Coded Decimal (BCD), Excess – 3, Gray code, Binary etc. out of these, binary addition is the most frequently performed task by most common adders. Apart from addition, adders are also used in certain digital applications like table index calculation, address decoding etc.

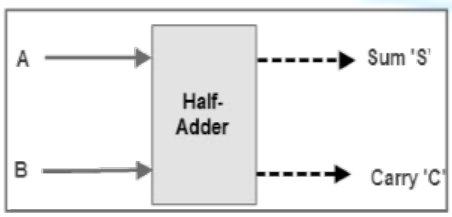
Binary addition is similar to that of decimal addition. Some basic binary additions are shown below.

0	0	1	1
+0	+1	+0	+1
0	1	1	(carry) 1 0

Figure 1. Schematic representation of half adder

1)Half Adder

Half adder is a combinational circuit that performs simple addition of two binary numbers. If we assume A and B as the two bits whose addition is to be performed,the block diagram and a truth table for half adder with A, B as inputs and Sum, Carry as outputs can be tabulated as follows.



Truth Table			
Input		Output	
A	B	Sum	Carry
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

Figure 2. Block diagram and truth table of half adder

The sum output of the binary addition carried out above is similar to that of an Ex-OR operation while the carry output is similar to that of an AND operation. The same can be verified with help of Karnaugh Map.
The truth table and K Map simplification and logic diagram for sum output is shown below.

Truth Table		
A	B	SUM
0	0	0
0	1	1
1	0	1
1	1	0

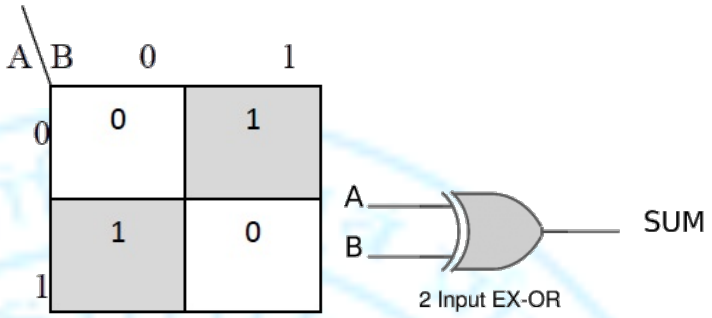


Figure 3. Truth table, K Map simplification and Logic diagram for sum output of half adder

$$Sum = A\,B' + A'\,B$$

The truth table and K Map simplification and logic diagram for carry is shown below.

Truth Table		
A	B	Carry
0	0	0
0	1	0
1	0	0
1	1	1

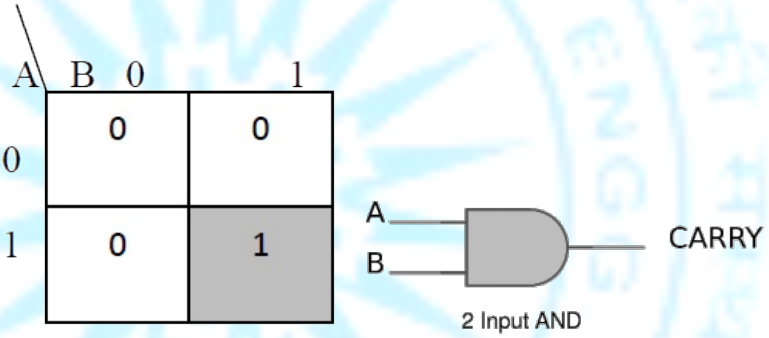


Figure 4. Truth table, K Map simplification and Logic diagram for sum output of half adder

$$Carry = AB$$

If A and B are binary inputs to the half adder, then the logic function to calculate sum S is Ex – OR of A and B and logic function to calculate carry C is AND of A and B. Combining these two, the logical circuit to implement the combinational circuit of half adder is shown below.

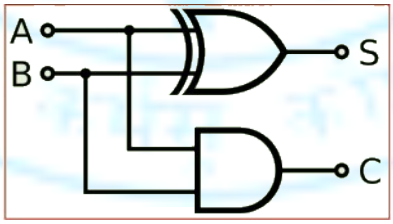


Figure 5. Half Adder Logic Diagram

As we know that NAND and NOR are called universal gates as any logic system can be implemented using these two, the half adder circuit can also be implemented using them. We know that a half adder circuit has one Ex – OR gate and one AND gate.

1.1)Half Adder using NAND gates:

Five NAND gates are required in order to design a half adder. The circuit to realize half adder using NAND gates is shown below.

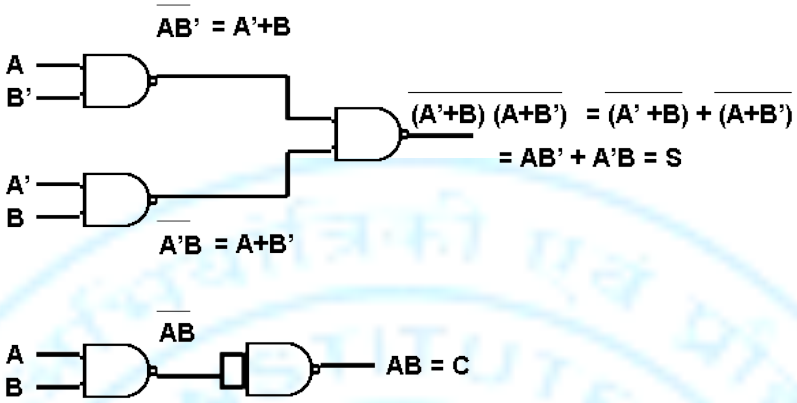


Figure 6. Realization of half adder using NAND gates

1.2)Half Adder using NOR gates

Five NOR gates are required in order to design a half adder. The circuit to realize half adder using NOR gates is shown below.

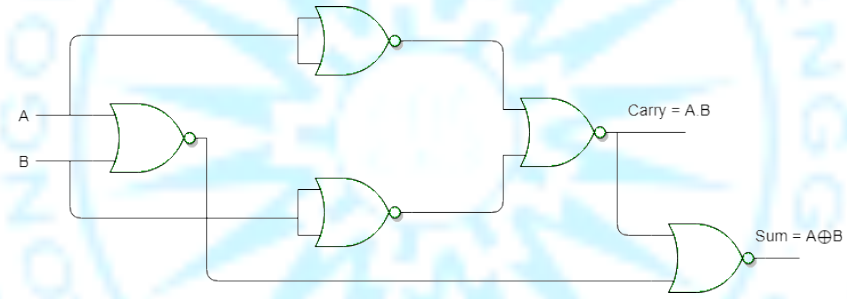
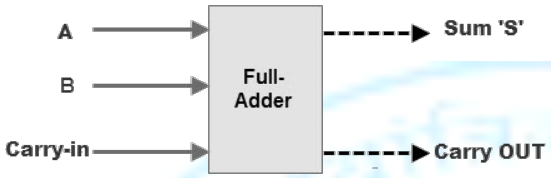


Figure 7. Realization of half adder using NOR Gates

2)Full Adder

Full adder is a digital circuit used to calculate the sum of three binary bits. Full adders are complex and difficult to implement when compared to half adders. Two of the three bits are same as before which are A, the augend bit and B, the addend bit. The additional third bit is carry bit from the previous stage and is called 'Carry' – in generally represented by CIN. It calculates the sum of three bits along with the carry. The output carry is called Carry – out and is represented by Carry OUT.

The block diagram of a full adder with A, B and CIN as inputs and S, Carry OUT as outputs is shown below.



Input			Output	
A	B	Cin	Sum	Carry
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

Figure 8. Full Adder Block Diagram and Truth Table

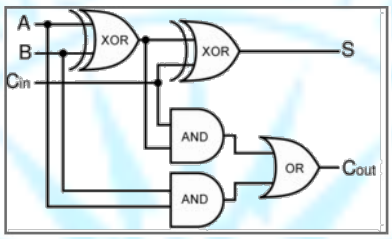


Figure 9. Full Adder Logic Diagram

Based on the truth table, the Boolean functions for Sum (S) and Carry – out (COUT) can be derived using K – Map.

A \ BC_{IN}	00	01	11	10
0	0	1	0	1
1	1	0	1	0

Figure 10. The K-Map simplified equation for sum is S = A'B'Cin + A'BCin' + ABCin

A \ BC_{IN}	00	01	11	10
0	0	0	1	0
1	0	1	1	1

Figure 11. The K-Map simplified equation for COUT is COUT = AB + ACIN + BCIN

In order to implement a combinational circuit for full adder, it is clear from the equations derived above, that we need four 3-input AND gates and one 4-input OR gates for Sum and three 2-input AND gates and one 3-input OR gate for Carry – out.

2.1)Full Adder using NAND gates:

As mentioned earlier, a NAND gate is one of the universal gates and can be used to implement any logic design. The circuit of full adder using only NAND gates is shown below.

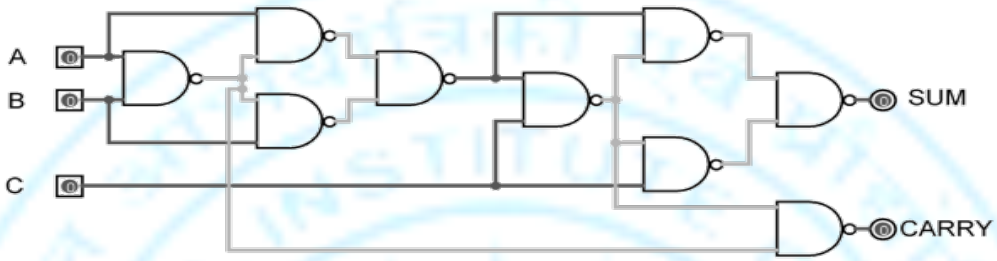


Figure 12. Full Adder using NAND gates

2.2)Full Adder using NOR gates:

As mentioned earlier, a NOR gate is one of the universal gates and can be used to implement any logic design. The circuit of full adder using only NOR gates is shown below.

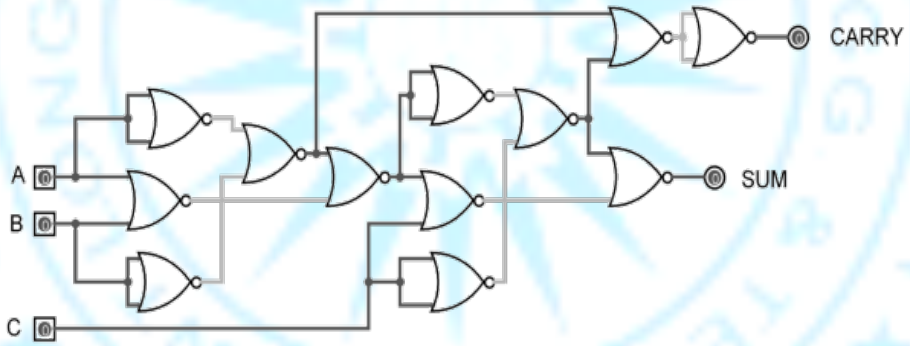


Figure 13. Full Adder using NOR gates

Experiment No. 4

Design and verification of the truth tables of Half and Full subtractor circuits.

Aim

To verify the truth table of half subtractor by using the ICs of XOR, NOT and AND gates and of full subtractor by using the ICs of XOR, AND, NOT and OR gates respectively and analyse the working of half subtractor and full subtractor circuit with the help of LEDs in simulator 1 and verify the truth table only of half subtractor and full subtractor in simulator 2.

Theory

Introduction

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.
1) Half Subtractor
2) Full Subtractor

1) Half Subtractor

The half-subtractor is a combinational circuit which is used to perform subtraction of two bits. It has two inputs, A (minuend) and B (subtrahend) and two outputs Difference and Borrow. The logic symbol and truth table are shown below.

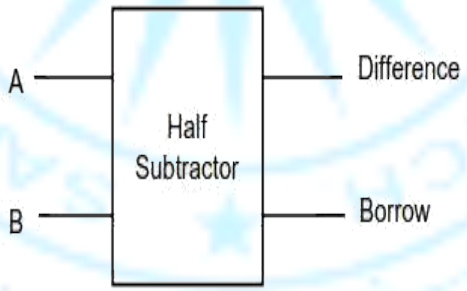


Figure-1:Logic Symbol of Half subtractor

Inputs		Outputs	
A	B	Difference	Borrow
0	0	0	0
0	1	1	1
1	0	1	0
1	1	0	0

Figure-2:Truth Table of Half subtractor

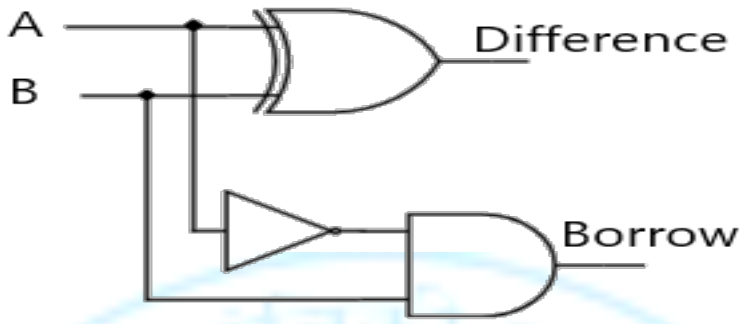


Figure-3:Circuit Diagram of Half subtractor

From the above truth table we can find the boolean expression.

Difference = A ⊕ B
Borrow = A' B

From the equation we can draw the half-subtractor circuit as shown in the figure 3.

2) Full Subtractor

A full subtractor is a combinational circuit that performs subtraction involving three bits, namely A (minuend), B (subtrahend), and Bin (borrow-in) . It accepts three inputs: A (minuend), B (subtrahend) and a Bin (borrow bit) and it produces two outputs: D (difference) and Bout (borrow out). The logic symbol and truth table are shown below.

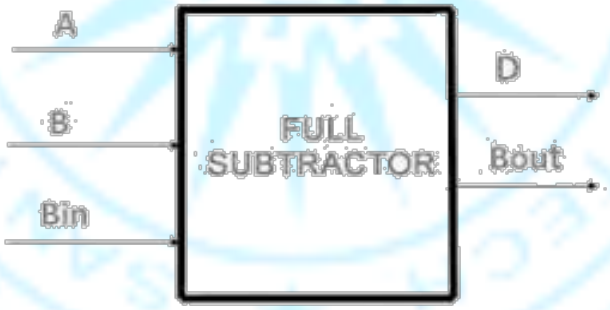


Figure-4:Logic Symbol of Full subtractor

A	B	B_{in}	D	B_{out}
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

Figure-5:Truth Table of Full subtractor

From the above truth table we can find the boolean expression.

$$D = A \oplus B \oplus Bin$$
$$Bout = A' Bin + A' B + B Bin$$

From the equation we can draw the Full-subtractor circuit as shown in the figure 6.

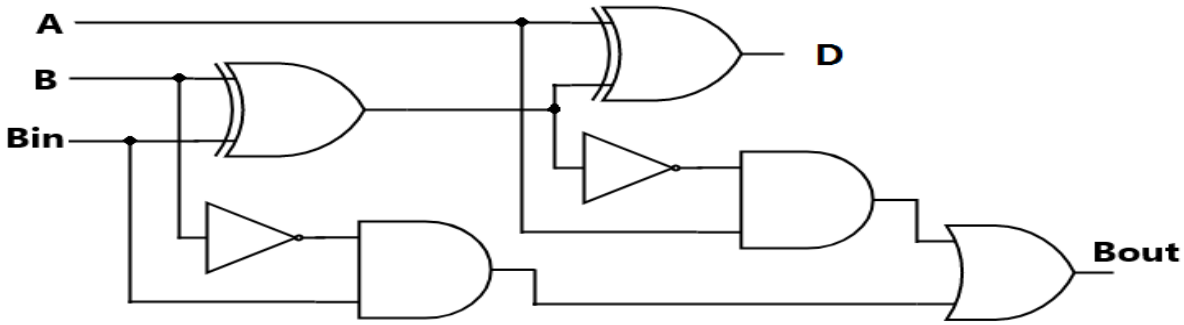


Figure-6:Circuit Diagram of Full subtractor

Experiment No. 5

Design and test of an S-R flip-flop using NOR/NAND gates.

Aim

To realize the functionality of sequential circuits using basic flip-flops.

Theory

Sequential Circuits: The logic circuits whose outputs at any instant of time depend not only on the present input but also on the past outputs are called sequential circuits.

The simplest kind of sequential circuit which is capable of storing one bit of information is called latch. The operation of basic latch can be modified, by providing an additional control input that determines when the state of the circuit is to be changed.

The latch with additional control input is called the Flip-Flop. The additional control input is either the clock or enable input.

S-R Flip-Flop

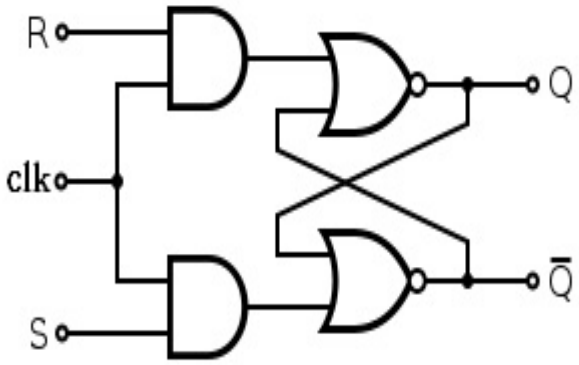


Figure 1: Clocked NOR-based S-R Flip-Flop

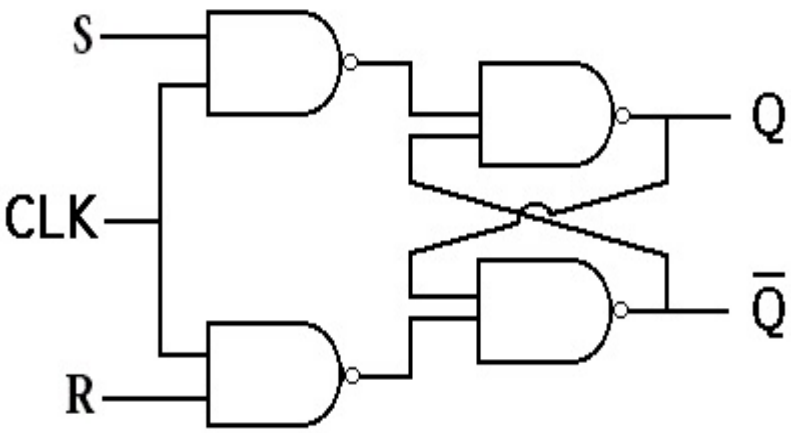


Figure 2: NAND-based S-R Flip-Flop

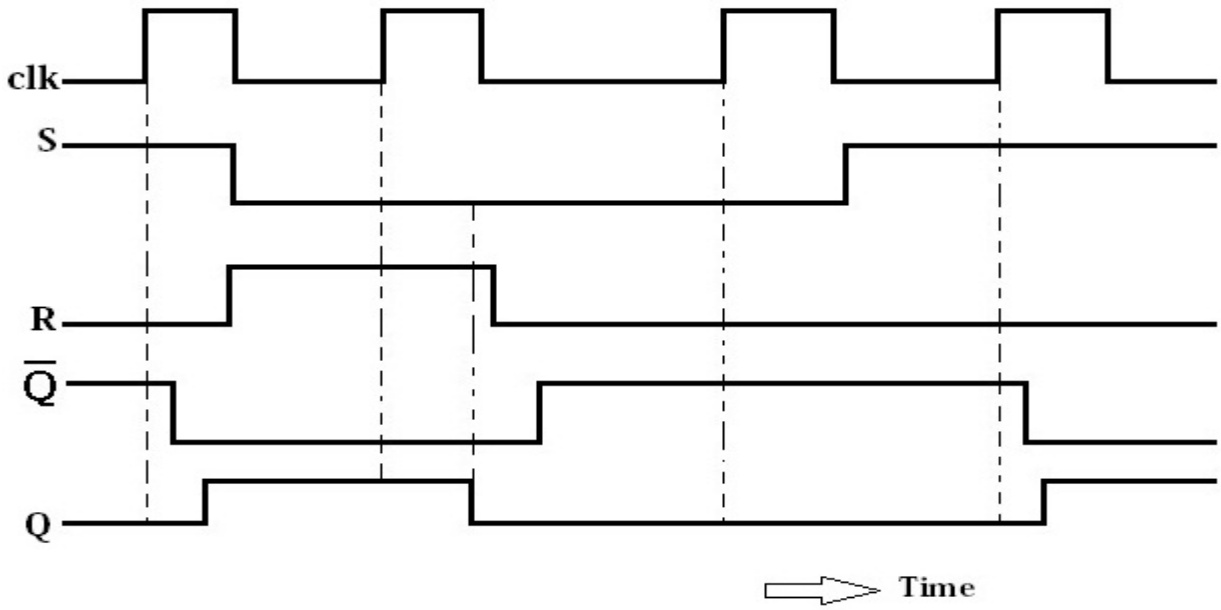


Figure 3: Typical wave-form in S-R Flip-Flop

S	R	Q(t+1)
0	0	Q(t)
0	1	0
1	0	1
1	1	Not Used

Figure 4: S-R Flip-Flop characteristic Table

NOTE :: clk, S and R signals are input signals
Qand Q : Output signals

Experiment No. 6

Verify the truth table of a J-K flip-flop.

Aim

To realize the functionality of sequential circuits using basic flip-flops.

Theory

Sequential Circuits: The logic circuits whose outputs at any instant of time depend not only on the present input but also on the past outputs are called sequential circuits.

The simplest kind of sequential circuit which is capable of storing one bit of information is called latch. The operation of basic latch can be modified, by providing an additional control input that determines when the state of the circuit is to be changed.

The latch with additional control input is called the Flip-Flop. The additional control input is either the clock or enable input.

J-K Flip-Flop

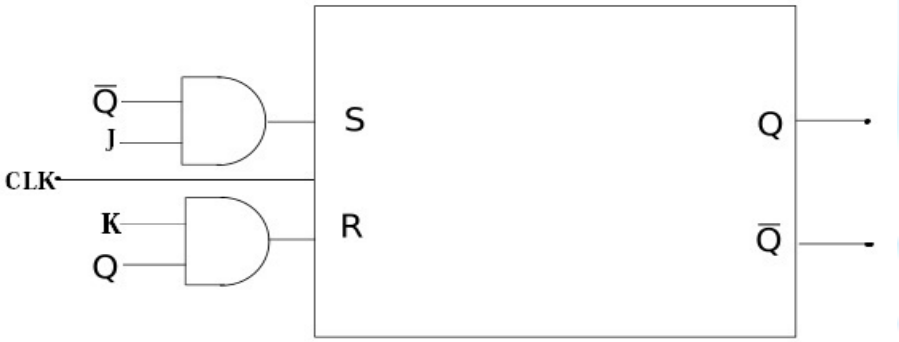


Figure 1: J-K Flip-Flop using S-R Flip-Flop

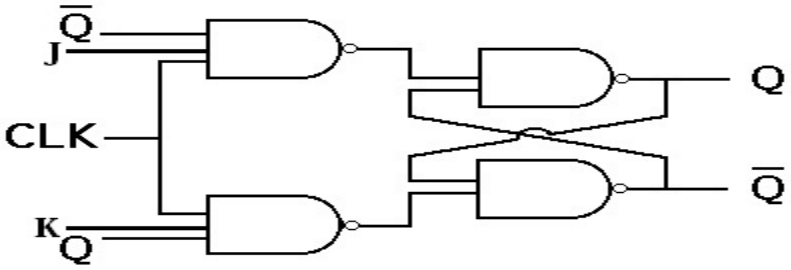


Figure 2: NAND based J-K Flip-Flop

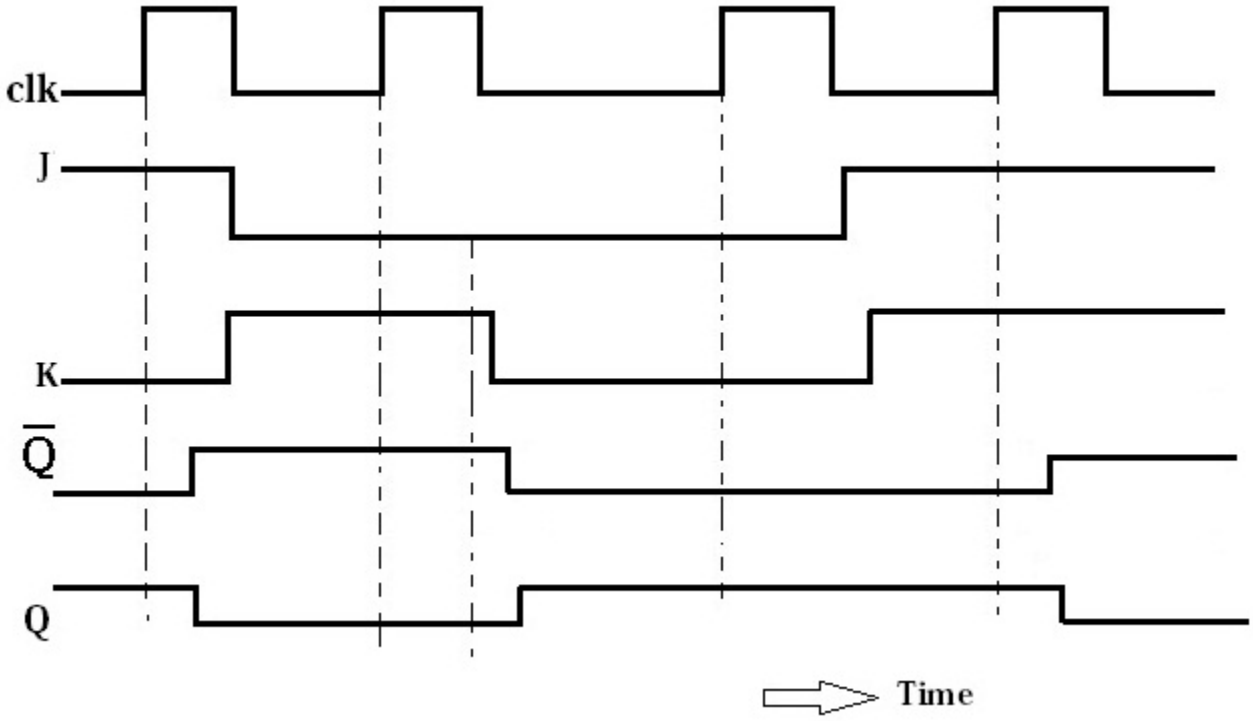


Figure 3: Typical wave-form in J-K Flip-Flop

J	K	Q(t+1)
0	0	Q(t)
0	1	0
1	0	1
1	1	$\overline{Q}$ (t)

Figure 4: J-K Flip-Flop characteristic Table

Experiment No. 7

Verify the truth table of a D flip-flop.

Aim

To realize the functionality of sequential circuits using basic flip-flops.

Theory

Sequential Circuits: The logic circuits whose outputs at any instant of time depend not only on the present input but also on the past outputs are called sequential circuits.

The simplest kind of sequential circuit which is capable of storing one bit of information is called latch. The operation of basic latch can be modified, by providing an additional control input that determines when the state of the circuit is to be changed.

The latch with additional control input is called the Flip-Flop. The additional control input is either the clock or enable input.

D Flip-Flop

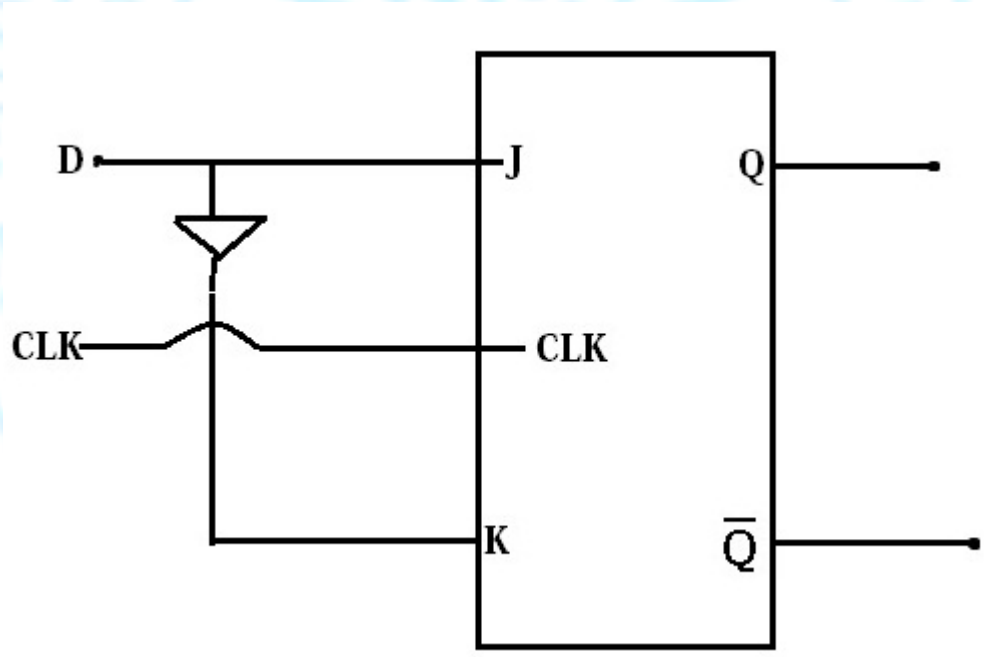


Figure 1: D Flip-Flop

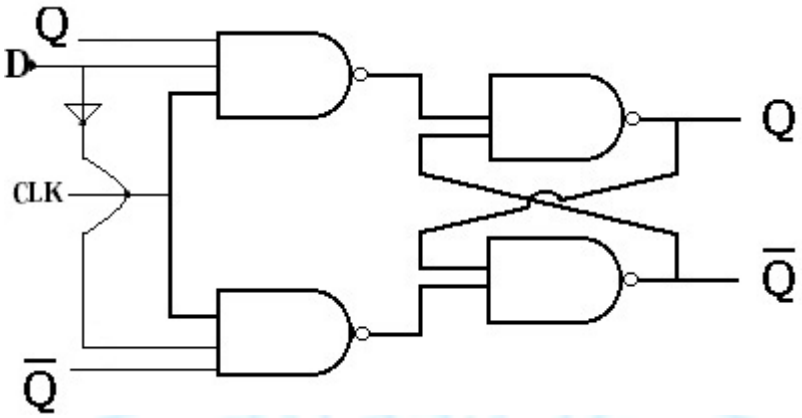


Figure 2: NAND-based D Flip-Flop

D	Q(t+1)
0	1
1	1

Figure 3: D Flip-Flop characteristic Table