|  |  |
| --- | --- |
| Logo | LUKHDHIRJI ENGINEERING COLLEGE |
|  | *Information Technology Department* |

Digital Fundamentals [3130704]

***Laboratory Manual***

*For*

***Semester – III***

***List of Experiments***

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Sr.**  **No.** | **Experiment Title** | **Page**  **No.** | **Date** | **Grade** | **Signature** |
| 1. | Study number system and write a program to convert Decimal number to 8-bit Binary number. |  |  |  |  |
| 2. | Write a program to convert 8-bit binary to Decimal number. |  |  |  |  |
| 3. | Write a program to convert Decimal number to Octal number. |  |  |  |  |
| 4. | Write a program to convert Decimal number to Hexadecimal number. |  |  |  |  |
| 5. | Write a program to ADD two 4-bit binary numbers using binary number system |  |  |  |  |
| 6. | Write a program to SUBTRACT two 4- bit binary numbers using 2`s complement system |  |  |  |  |
| 7. | Perform Experiment of Adder circuit in Virtual Labs. |  |  |  |  |
| 8. | Perform Experiment of Multiplexer in Virtual Labs. |  |  |  |  |
| 9. | Perform Experiment of Counters in Virtual Labs. |  |  |  |  |
| 10. | Perform Experiment of Registers in Virtual Labs. |  |  |  |  |
| 11. | Perform Experiment of Latch and Flip-flop in Virtual Labs. |  |  |  |  |

TotalGradePoints: / 110

|  |  |  |
| --- | --- | --- |
| SUBJECT:  ***Digital Electronics*** | TITLE:  **Write a program to convert Decimal number to 8-bit Binary number** | |
| *Sem. III* | EXPERIMENTNO.: **01** | DATE : |

# AIM:

Write a program to convert Decimal number to 8-bit Binary number.

# THEORY:

# Number System

# Four types of Numbering System

## Decimal Number System

## The decimal system consists of 10 (0 to 9) unique symbols. Hence the base and radix is 10.

## It is positional weighted system. In this system, any number of any magnitude can be represented by the use of these ten symbols only

## The digits on the left side of the decimal point from the integer part of a decimal number while those on right side from the fractional part.

## Thedigitsontherightofthedecimalpointhaveweightswhich are negative powers of 10 and the digits to the left of the decimal point have weights which are positive powers of 10.

## Binary Number System

## Uses two digits, 0 and 1

## Also called as base 2 number system

## Each position in a binary number represents a 0 power of the base (2). Example 20

## Octal Number System

## Uses eight digits, 0,1,2,3,4,5,6,7

## Also called as base 8 number system

## Each position in an octal number represents a 0 power of the base (8). Example 80

## Hexadecimal Number System

## Uses 10 digits and 6 letters

## Letters represent the numbers starting from 10. A = 10. B = 11, C = 12, D = 13, E = 14, F = 15

## Also called as base 16 number system

## Each position in a hexadecimal number represents a 0 power of the base (16). Example, 160

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Sr.**  **No** | **Number System** | **Base** | **First digit** | **Last digit** | **All digits** |
| 1 | Binary | 2 | 0 | 1 | 0,1 |
| 2 | Octal | 8 | 0 | 7 | 0,1,2,3,4,5,6,7 |
| 3 | Decimal | 10 | 0 | 9 | 0,1,2,3,4,5,6,7,8,9 |
| 4 | Hexadecimal | 16 | 0 | F | 0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F |

# PROCEDURE:

# This program takes a decimal number as input and converts it into binary number.

# Steps for Decimal to Binary Conversion

# **Step 1**: Remainder when 10 is divided by 2 is zero. Therefore, arr[0] = 0.

# Step 2:Divide 10 by 2. New number is 10/2 = 5.

# Step 3:Remainder when 5 is divided by 2 is 1. Therefore, arr[1] = 1.

# Step 4:Divide 5 by 2. New number is 5/2 = 2.

# Step 5:Remainder when 2 is divided by 2 is zero. Therefore, arr[2] = 0.

# Step 6:Divide 2 by 2. New number is 2/2 = 1.

# Step 7:Remainder when 1 is divided by 2 is 1. Therefore, arr[3] = 1.

# Step 8:Divide 1 by 2. New number is 1/2 = 0.

# Step 9:Since number becomes = 0. Print the array in reverse order. Therefore the equivalent binary number is 1010.

# Below diagram shows an example of converting the decimal number 17 to equivalent binary number.

# 

# Program for Decimal to Binary Conversion

# #include <stdio.h>

# int main()

# {

# int a[10], number, i, j;

# printf("\n Please Enter the Number You want to Convert : ");

# scanf("%d", &number);

# for(i = 0; number > 0; i++)

# {

# a[i] = number % 2;

# number = number / 2;

# }

# printf("\n Binary Number of a Given Number = ");

# for(j = i - 1; j >= 0; j--) {

# printf(" %d ", a[j]);

# }

# printf("\n");

# return 0;

# }

# OUTPUT:

# Please Enter the Number You want to Convert : 17

# Binary Number of a Given Number = 10001

**Signature: Grade: /10**

|  |  |  |
| --- | --- | --- |
| SUBJECT:  ***Digital Electronics*** | TITLE:  **Write a program to convert 8-bit binary to Decimal number.** | |
| *Sem. III* | EXPERIMENTNO.: **02** | DATE : |

# AIM:

Write a program to convert Decimal number to 8-bit Binary number.

# THEORY:

# This program takes a binary number as input and converts it into decimal number.

# Steps forBinary to Decimal Conversion

# Step 1: Take a binary number as input.

# Step 2: Multiply each digits of the binary number starting from the last with the powers of 2 respectively.

# Step 3: Add all the multiplied digits.

# Step 4: The total sum gives the decimal number.

# Below diagram explains how to convert (1010) to equivalent decimal value:

# 

# PROCEDURE:

# Program for Binary to Decimal Conversion

#include<stdio.h>

#include<conio.h>

void main()

{

clrscr();

long int binnum, decnum=0, i=1, rem;

printf("Enter any binary number : ");

scanf("%ld",&binnum);

while(binnum!=0)

{

rem=binnum%10;

decnum=decnum+rem\*i;

i=i\*2;

binnum=binnum/10;

}

printf("Equivalent decimal value = %ld",decnum);

getch();

}

# OUTPUT:

# Enter the binary number:1010

# Equivalent decimal value= 10

**Signature: Grade: /10**

|  |  |  |
| --- | --- | --- |
| SUBJECT:  ***Digital Electronics*** | Write a program to convert Decimal number to Octal number. | |
| *Sem. III* | EXPERIMENTNO.: **03** | DATE : |

**AIM:**Write a program to convert Decimal number to Octal number.

**Theory:**

**Steps to Convert Decimal to Octal:**

**Step 1:**Accept the given decimal number

Step 2: If the number is less than 8 the octal number is the same

Step 3: If the num > 7 then Divide the number with 8

Step 4: Write down the remainder

Step 5: Do steps 3 and 4 with the quotient till that quotient is less than 8

Step 6: Write the remainders in reverse order (bottom to top)

Step 7:The resultant is the equivalent octal number to the given decimal number

# Below diagram shows an example of converting the decimal number 33 to equivalent octal number.

# 

# PROCEDURE:

# Program for Decimal to Octal Conversion

#include <stdio.h>

#include <math.h>

int decimalToOctal(int decimalnum)

{

int octalnum = 0, temp = 1;

while (decimalnum != 0)

{

octalnum = octalnum + (decimalnum % 8) \* temp;

decimalnum = decimalnum / 8;

temp = temp \* 10;

}

return octalnum;

}

int main()

{

int decimalnum;

printf("Enter a Decimal Number: ");

scanf("%d", &decimalnum);

printf("Equivalent Octal Number: %d", decimalToOctal(decimalnum));

return 0;

}

# OUTPUT:

# Enter a Decimal Number: 436

# Equivalent Octal Number: 664

**Signature: Grade: /10**

|  |  |  |
| --- | --- | --- |
| SUBJECT:  ***Digital Electronics*** | Write a program to convert Decimal number to Hexadecimal number. | |
| *Sem. III* | EXPERIMENTNO.: **04** | DATE : |

**AIM:**Write a program to convert Decimal number to Hexadecimal number.

**Theory:**

**Steps to Convert Decimal to Hexadecimal number:**

If the given decimal number is 2545.

**Step 1:** Calculate remainder when 2545 is divided by 16 is 1. Therefore, temp = 1. As temp is less than 10. So, arr[0] = 48 + 1 = 49 = ‘1’.

**Step 2**: Divide 2545 by 16. New number is 2545/16 = 159.

**Step 3**: Calculate remainder when 159 is divided by 16 is 15. Therefore, temp 15. As temp is greater than 10. So, arr[1] = 55 + 15 = 70 = ‘F’.

**Step 4**: Divide 159 by 16. New number is 159/16 = 9.

**Step 5:** Calculate remainder when 9 is divided by 16 is 9. Therefore, temp =

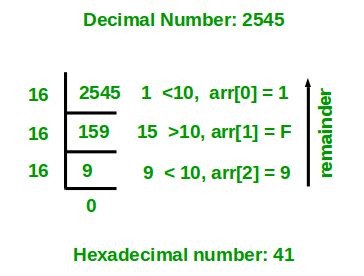
9. As temp is less than 10. So, arr[2] = 48 + 9 = 57 = ‘9’.

**Step 6**: Divide 9 by 16. New number is 9/16 = 0.

**Step 7**: Since number becomes = 0. Stop repeating steps and print the array

in reverse order. Therefore the equivalent hexadecimal number is 9F1.

* Below diagram shows an example of converting the decimal number 2545 to equivalent hexadecimal number.



# PROCEDURE:

# Program for Binary to Hexadecimal Conversion

#include<stdio.h>

#include<conio.h>

void main()

{

clrscr();

long int decnum, rem, quot;

char hexdecnum[100];

int i=1, j, temp;

printf("Enter any decimal number : ");

scanf("%ld",&decnum);

quot=decnum;

while(quot!=0)

{

temp=quot%16;

// to convert integer into character

if(temp<10)

{

temp=temp+48;

}

else

{

temp=temp+55;

}

hexdecnum[i++]=temp;

quot=quot/16;

}

printf("Equivalent hexadecimal value of %d is : \n",decnum);

for(j=i-1; j>0; j--)

{

printf("%c",hexdecnum[j]);

}

getch();

}

# OUTPUT:

# Enter any decimal number: 45

# Equivalent hexadecimal value is : 2D

**Signature: Grade: /10**

|  |  |  |
| --- | --- | --- |
| SUBJECT:  ***Digital Electronics*** | Write a program to ADD two 4-bit binary numbers using binary number system | |
| *Sem. III* | EXPERIMENTNO.: **05** | DATE : |

**AIM:**Write a program to ADD two 4-bit binary numbers using binary number system

**Theory:**

**Steps to ADD two 4-bit binary numbers:**

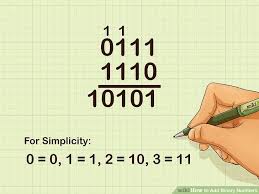
Step 1: Take two binary numbers as input.

Step 2:  Add each bits from the two binary numbers separately starting from LSB.

Step 3: The operations may be as follows.

1. (0+0) =0
2. (1+0) =1
3. (1+1)=0 and 1 reminder.

* Below diagram shows an example of to ADD two 4-bit binary numbers



# PROCEDURE:

* **Program for to ADD two 4-bit binary numbers**

#include <stdio.h>

int main()

{

long binary1, binary2;

int i = 0, remainder = 0, sum[20];

printf("Enter the first binary number: ");

scanf("%ld", &binary1);

printf("Enter the second binary number: ");

scanf("%ld", &binary2);

while (binary1 != 0 || binary2 != 0)

{

sum[i++] =(binary1 % 10 + binary2 % 10 + remainder) % 2;

remainder =(binary1 % 10 + binary2 % 10 + remainder) / 2;

binary1 = binary1 / 10;

binary2 = binary2 / 10;

}

if (remainder != 0)

sum[i++] = remainder;

--i;

printf("Sum of two binary numbers: ");

while (i>= 0)

printf("%d", sum[i--]);

return 0;

}

**OUTPUT:**

Enter the first binary number: 100000

Enter the second binary number: 101010

Sum of two binary numbers: 1001010

**Signature: Grade: /10**

|  |  |  |
| --- | --- | --- |
| SUBJECT:  ***Digital Electronics*** | Write a program to SUBTRACT two 4- bit binary numbers using 2`s complement system | |
| *Sem. III* | EXPERIMENTNO.: **06** | DATE : |

**AIM:**Write a program to SUBTRACT two 4- bit binary numbers using 2`s complement system.

**THEORY:**

Given two numbers a and b. The task is to subtract b from a by using **2’s Complement** method.

**Note**: Negative numbers represented as 2’s Complement of Positive Numbers.

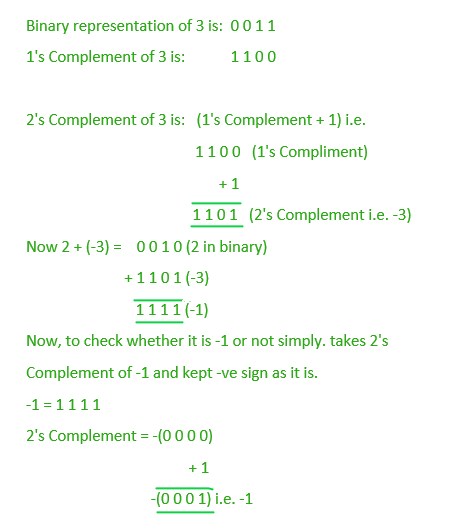
To subtract b from a Write the expression (a-b) as:

(a - b) = a + (-b)

Now (-b) can be written as (2’s complement of b). So the above expression can be now written as:

(a - b) = a + (2's complement of b)

So, the problem now reduces to “Add a to the 2’s complement of b “.

* Below image illustrates the above method of subtraction for the first example where a = 2 and b = 3.

# PROCEDURE:

* **Program for to SUBTRACT two 4- bit binary numbers using 2`s complement system.**

int Subtract(int a, int b)

{

    int c;

 // ~b is the 1's Complement of b

    // adding 1 to it make it 2's Complement

    c = a + (~b + 1);

    return c;

}

// Driver code

int main()

{

    int a = 2, b = 3;

    printf(“/n”Subtract(a, b));

   a = 9; b = 7;

    printf(Subtract(a, b));

    return 0;

}

**OUTPUT:**

-1

2

**Signature: Grade: /10**

|  |  |  |
| --- | --- | --- |
| SUBJECT:  ***Digital Electronics*** | Perform Experiment of Adder circuit in Virtual Labs. | |
| *Sem. III* | EXPERIMENTNO.: **07** | DATE : |

**AIM:**Perform Experiment of Adder circuit in Virtual Labs.

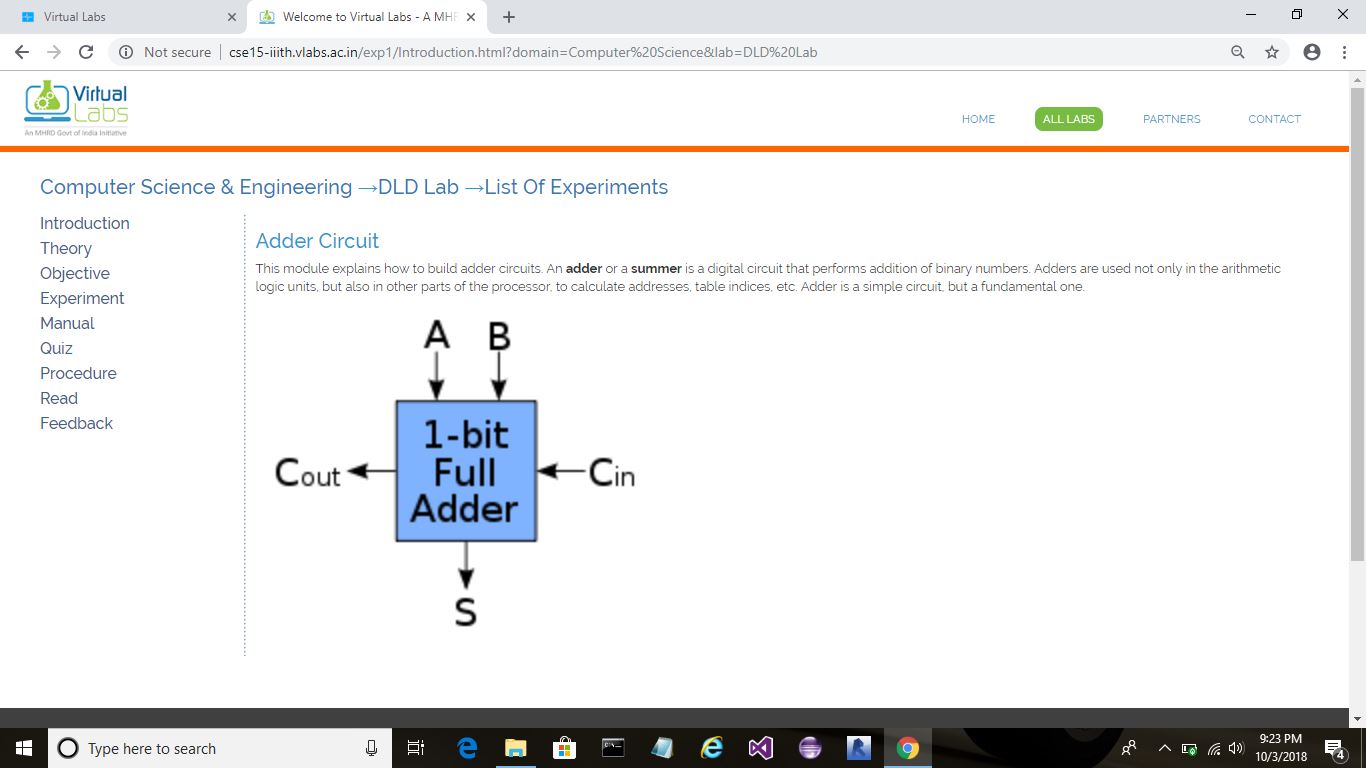
**THEORY:**

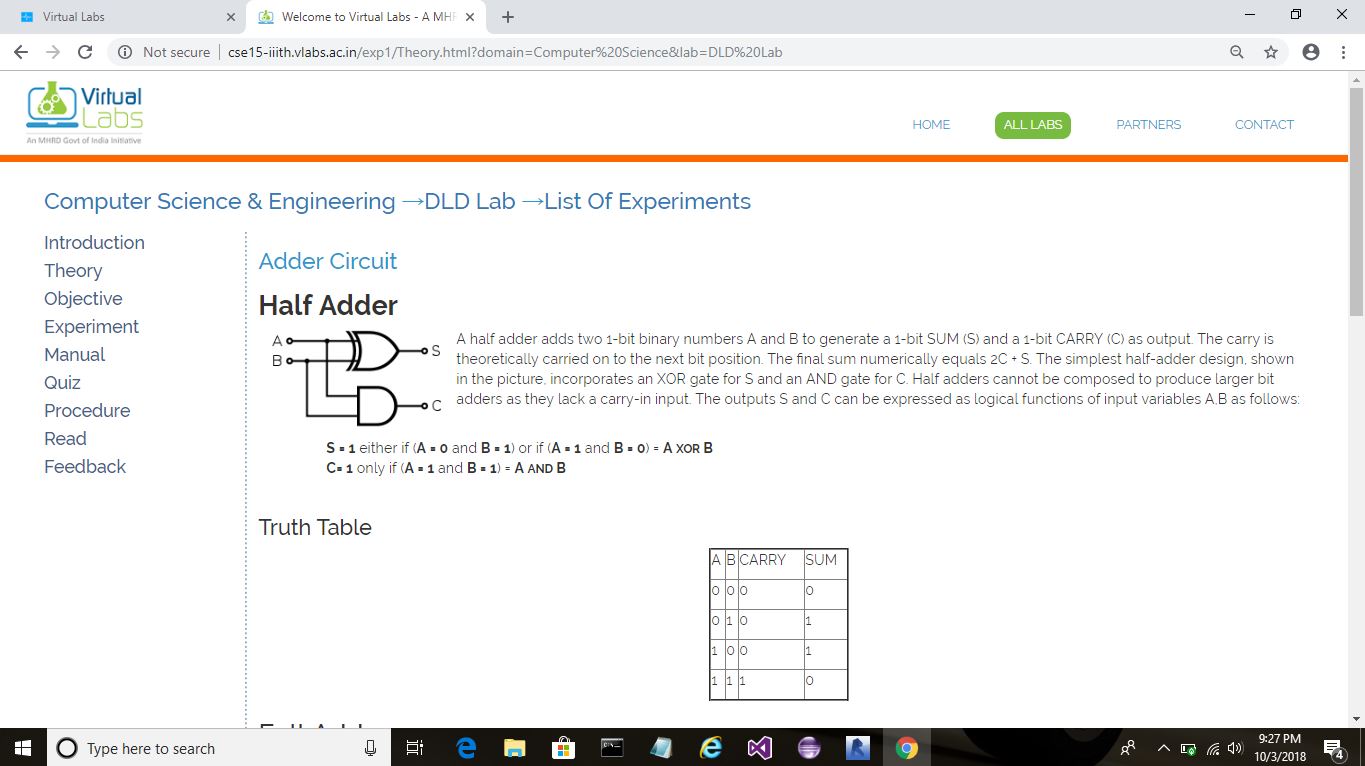
An adder is a digital circuit that performs addition of numbers.

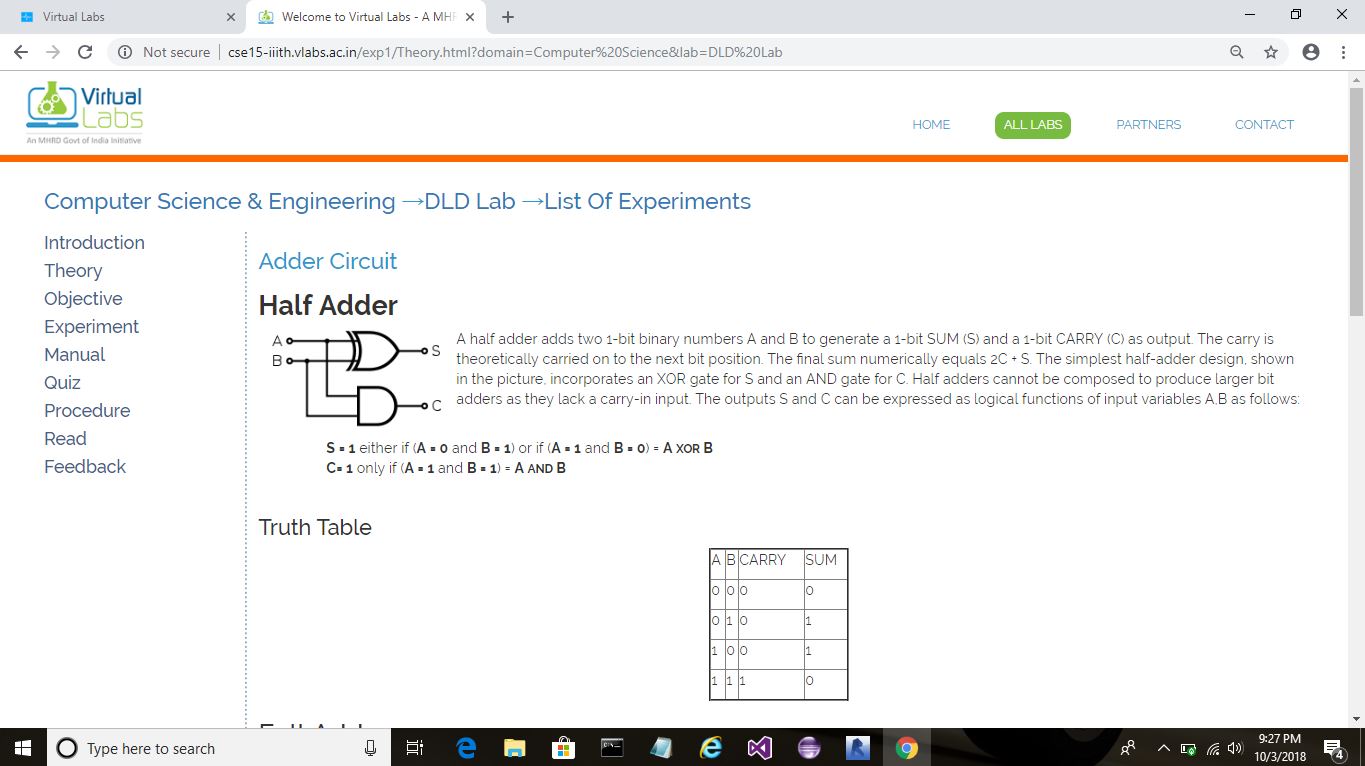
The half adder adds two binary digits called as augend and addend and produces two outputs as sum and carry; XOR is applied to both inputs to produce sum and AND gate is applied to both inputs to produce carry.

The full adder adds 3 one bit numbers, where two can be referred to as operands and one can be referred to as bit carried in. And produces 2-bit output, and these can be referred to as output carry and sum.

**PROCEDURE:**

****

****

****

**Signature: Grade: /10**

|  |  |  |
| --- | --- | --- |
| SUBJECT:  ***Digital Electronics*** | Perform Experiment of Multiplexer in Virtual Labs. | |
| *Sem. III* | EXPERIMENTNO.: **08** | DATE : |

**AIM:**Perform Experiment of Multiplexer in Virtual Labs.

**THEORY:**

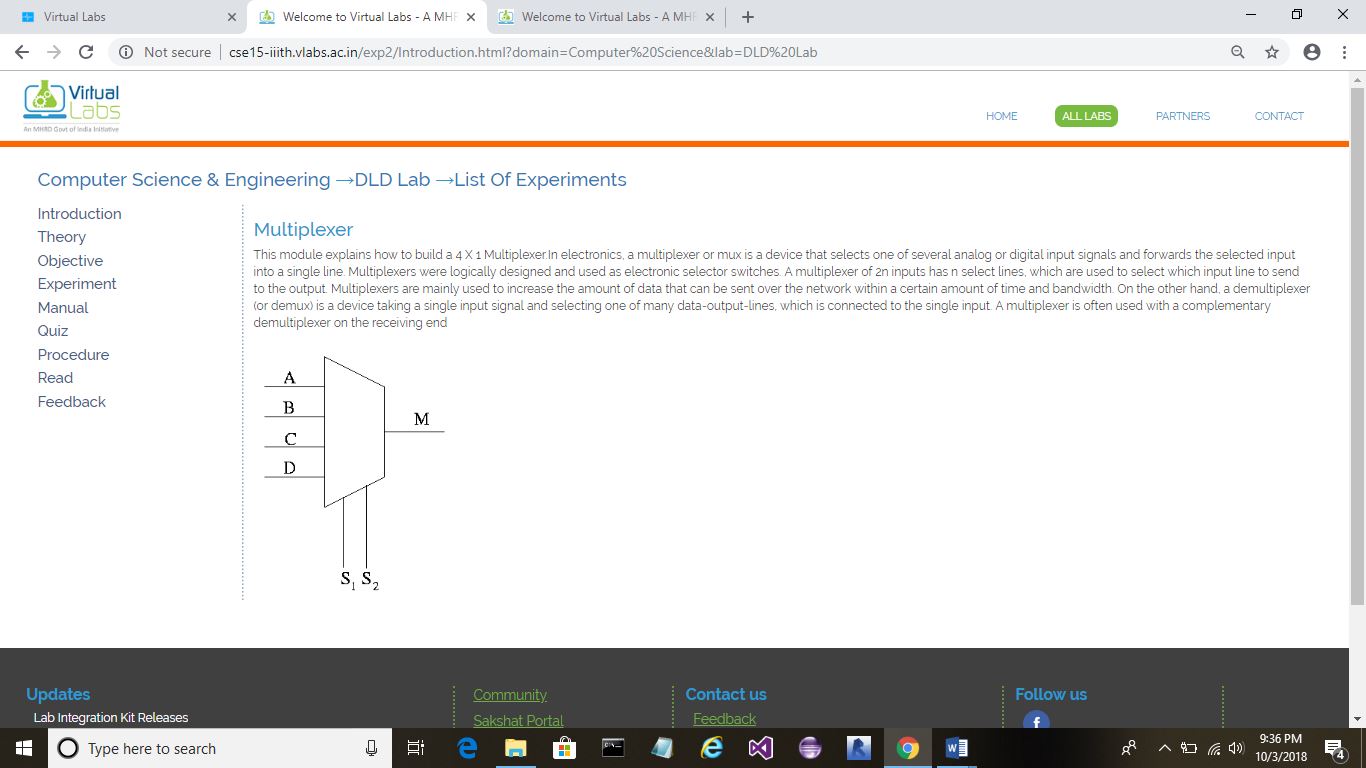
Multiplexer is a combinational circuit that has maximum of 2n data inputs, ‘n’ selection lines and single output line.

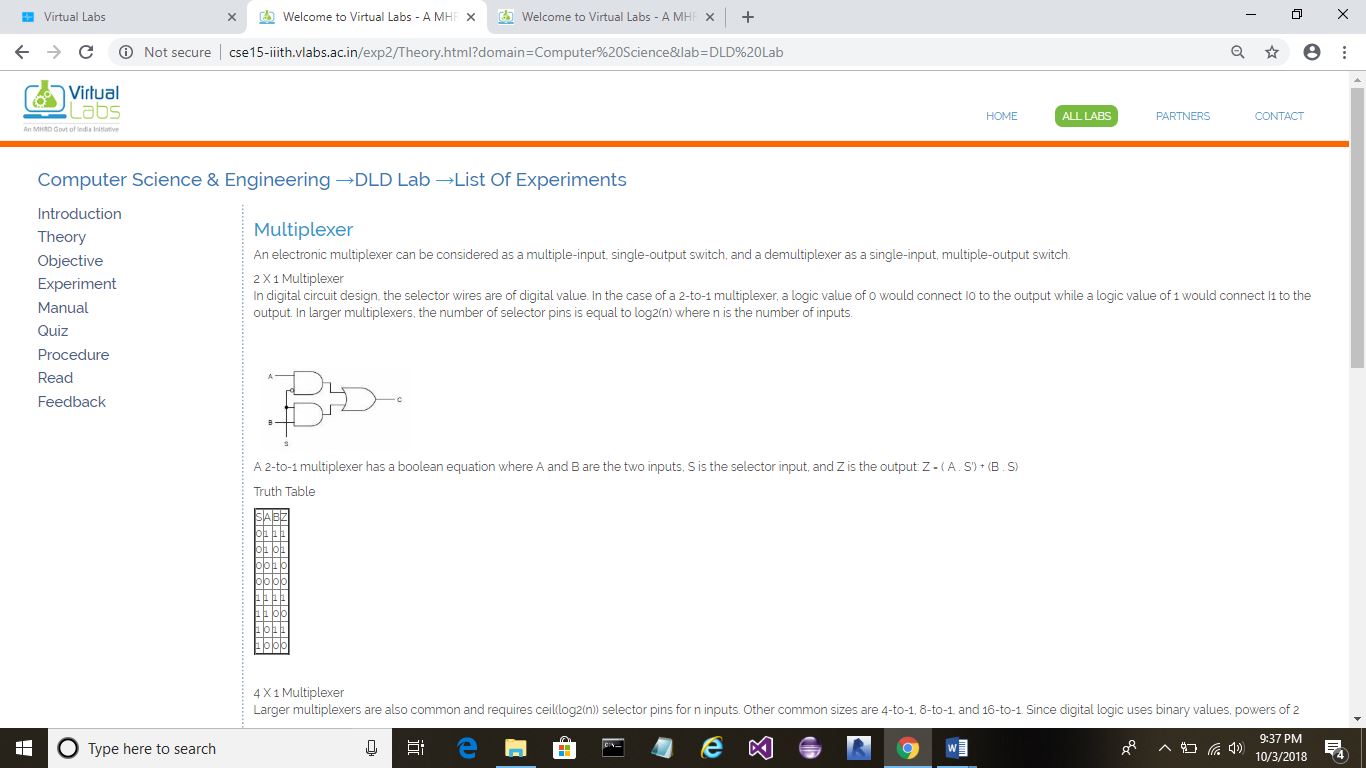
One of these data inputs will be connected to the output based on the values of selection lines.

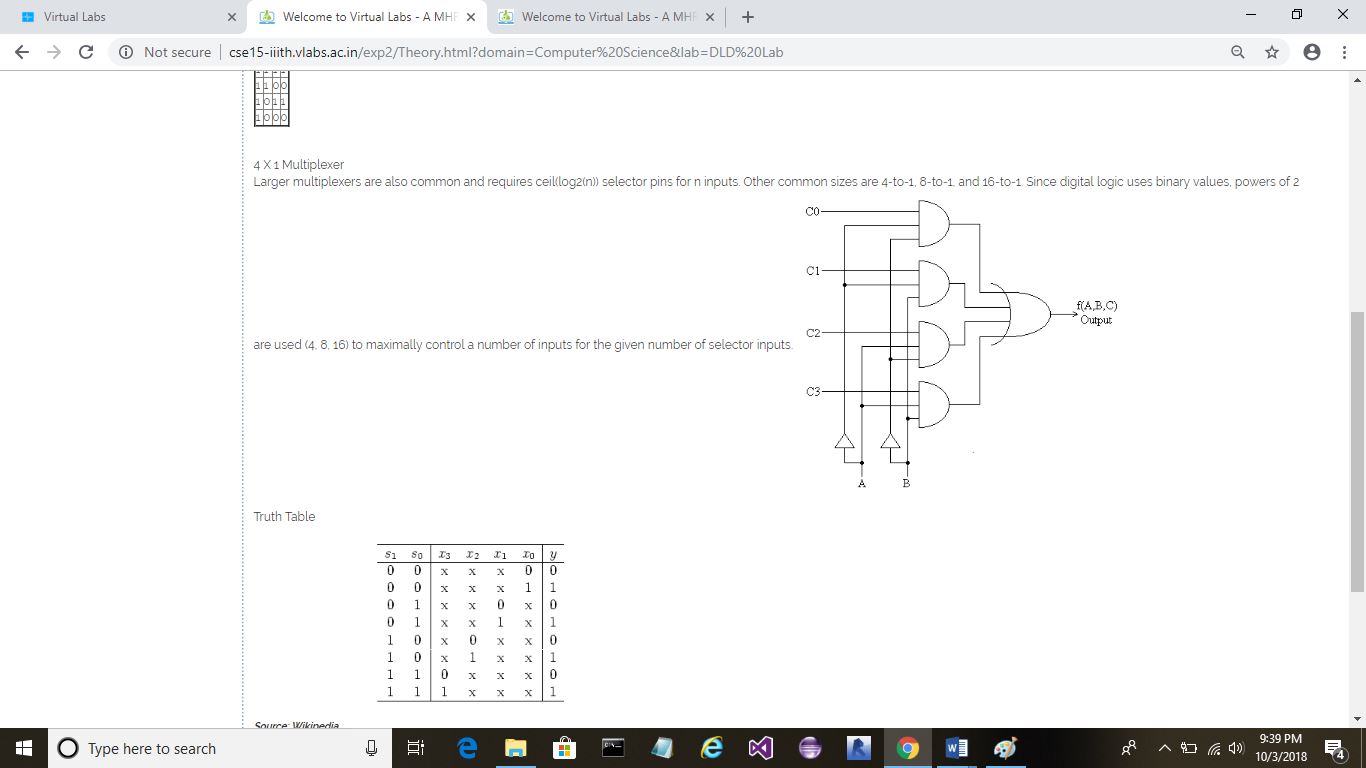
Since there are ‘n’ selection lines, there will be 2n possible combinations of zeros and ones.

So, each combination will select only one data input. Multiplexer is also called as **Mux**.

**PROCEDURE:**







**Signature: Grade: /10**

|  |  |  |
| --- | --- | --- |
| SUBJECT:  ***Digital Electronics*** | Perform Experiment of Counters in Virtual Labs. | |
| *Sem. III* | EXPERIMENTNO.: **09** | DATE : |

**AIM:**Perform Experiment of Counters in Virtual Labs.

**THEORY:**

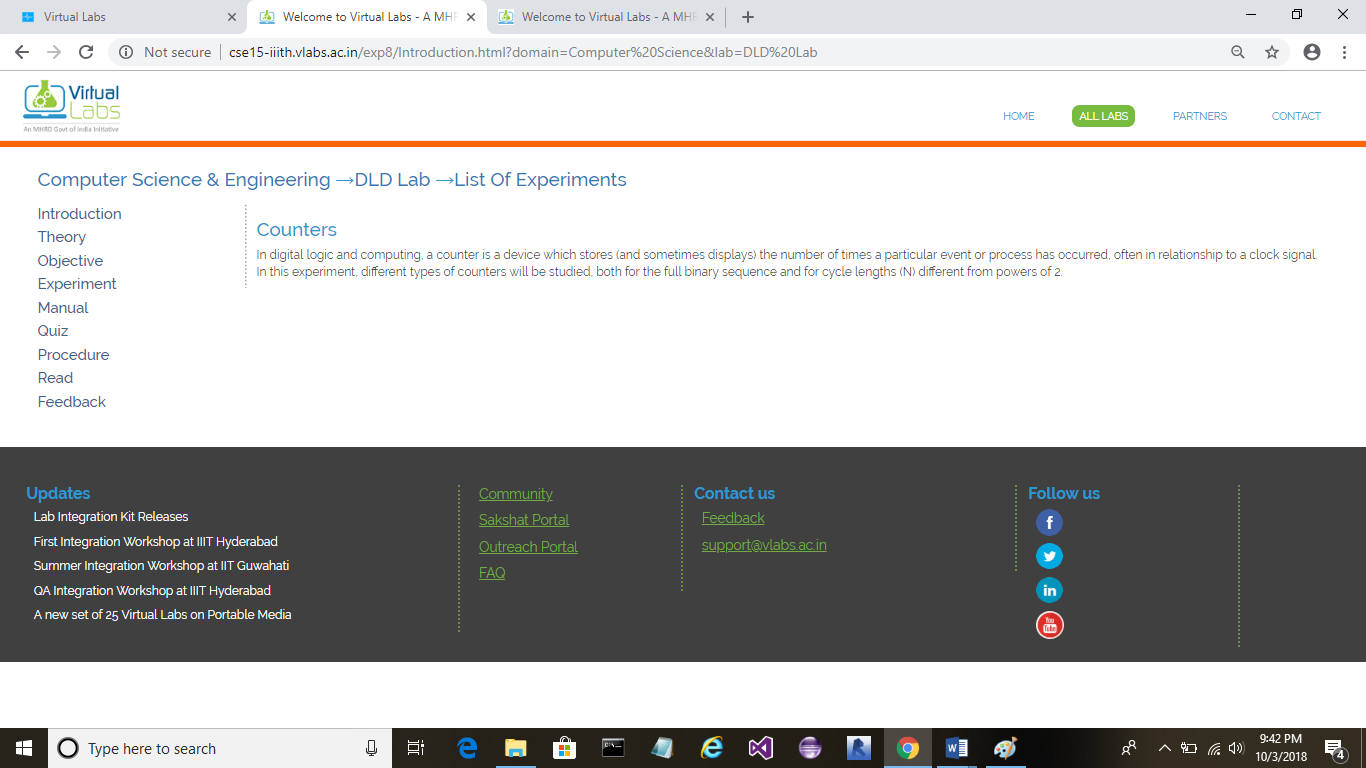
A counter is a device which stores the number of times a particular event or process has occurred, often in relationship to a clock signal.

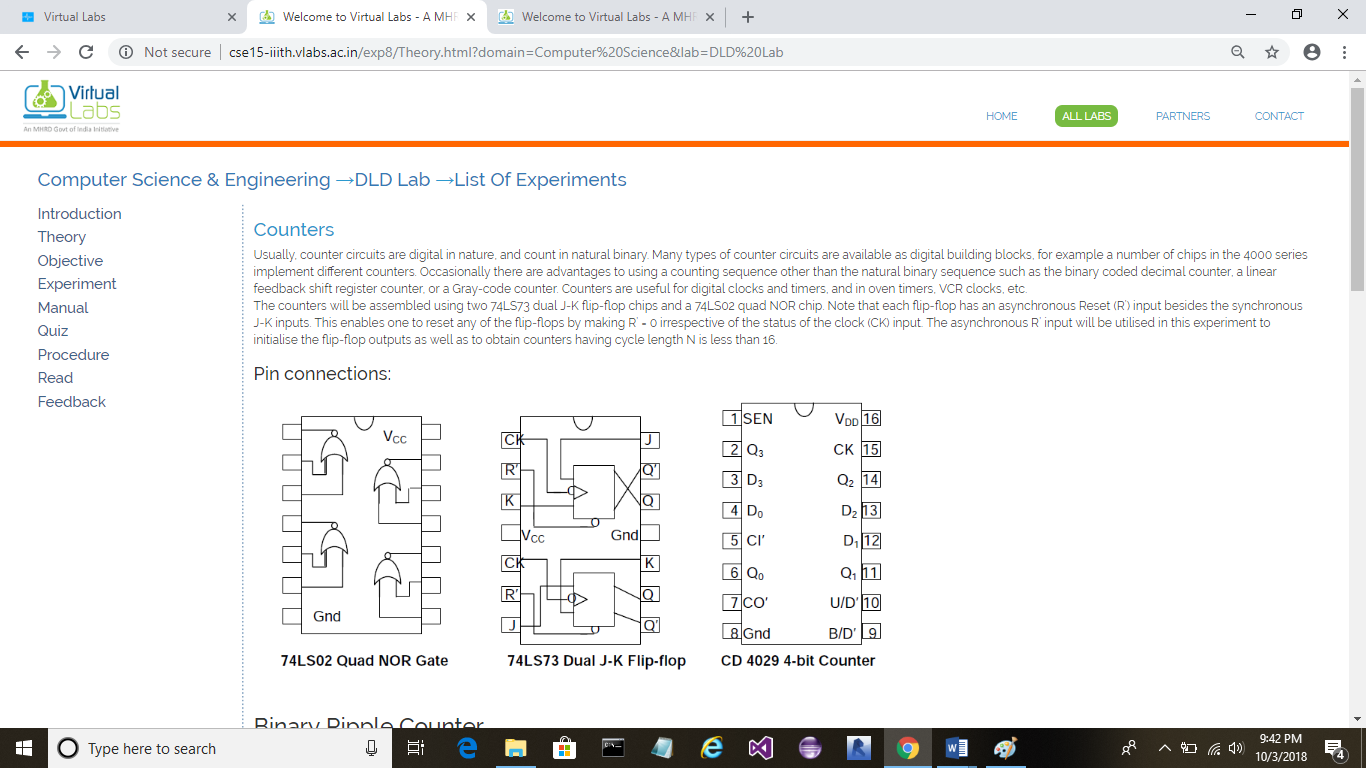
Counters are used in digital electronics for counting purpose, they can count specific event happening in the circuit.

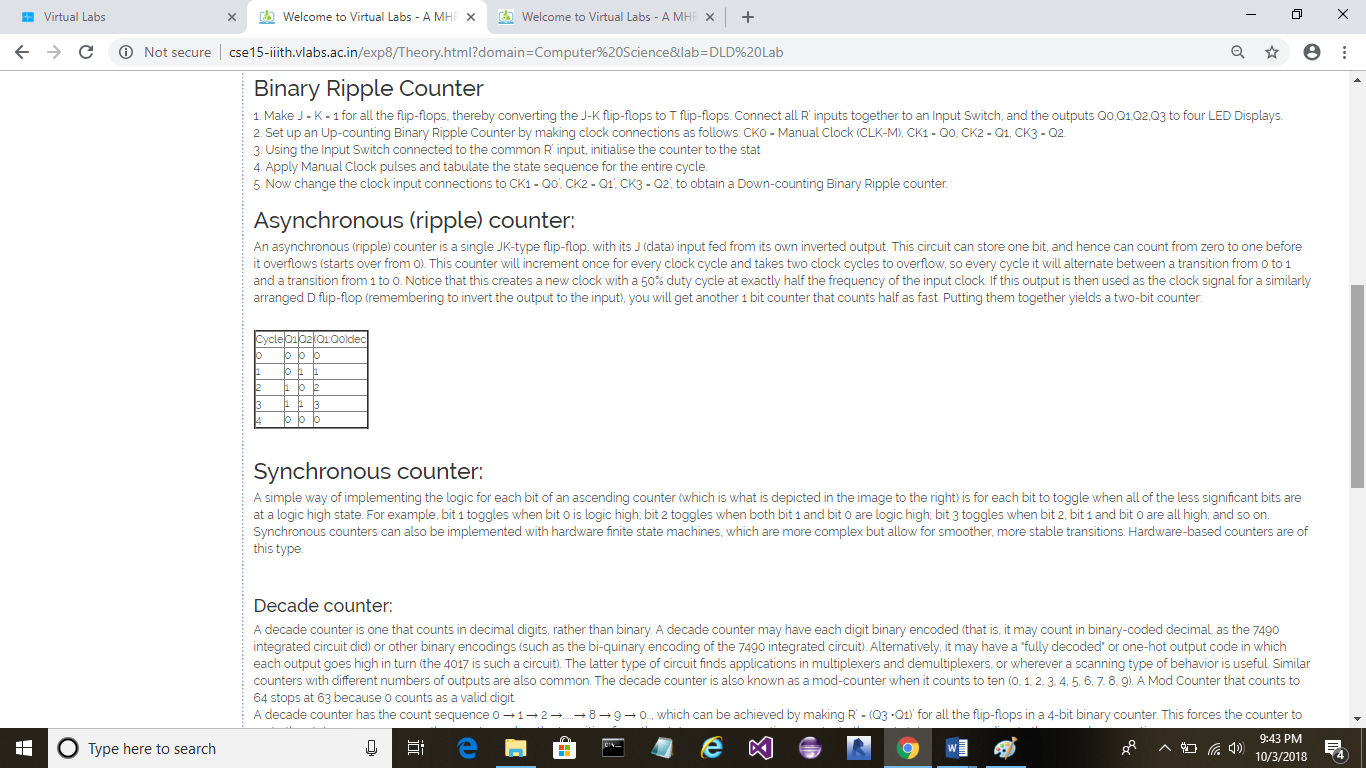
For example, in UP counter a counter increases count for every rising edge of clock.

Not only counting, a counter can follow the certain sequence based on our design like any random sequence 0,1,3,2…. They can also be designed with the help of flip flops.

**PROCEDURE:**







**Signature: Grade: /10**

|  |  |  |
| --- | --- | --- |
| SUBJECT:  ***Digital Electronics*** | **Perform Experiment of Registers in Virtual Labs.** | |
| *Sem. III* | EXPERIMENTNO.: **10** | DATE : |

**AIM:**Perform Experiment of Registers in Virtual Labs

**THEORY:**

Flip-flop is a 1bit memory cell which can be used for storing the digital data.

To increase the storage capacity in terms of number of bits, we have to use a group of flip-flop. Such a group of flip-flop is known as a Register.

The n-bitregister will consist of n number of flip-flop and it is capable of storing an **n-**bit word.

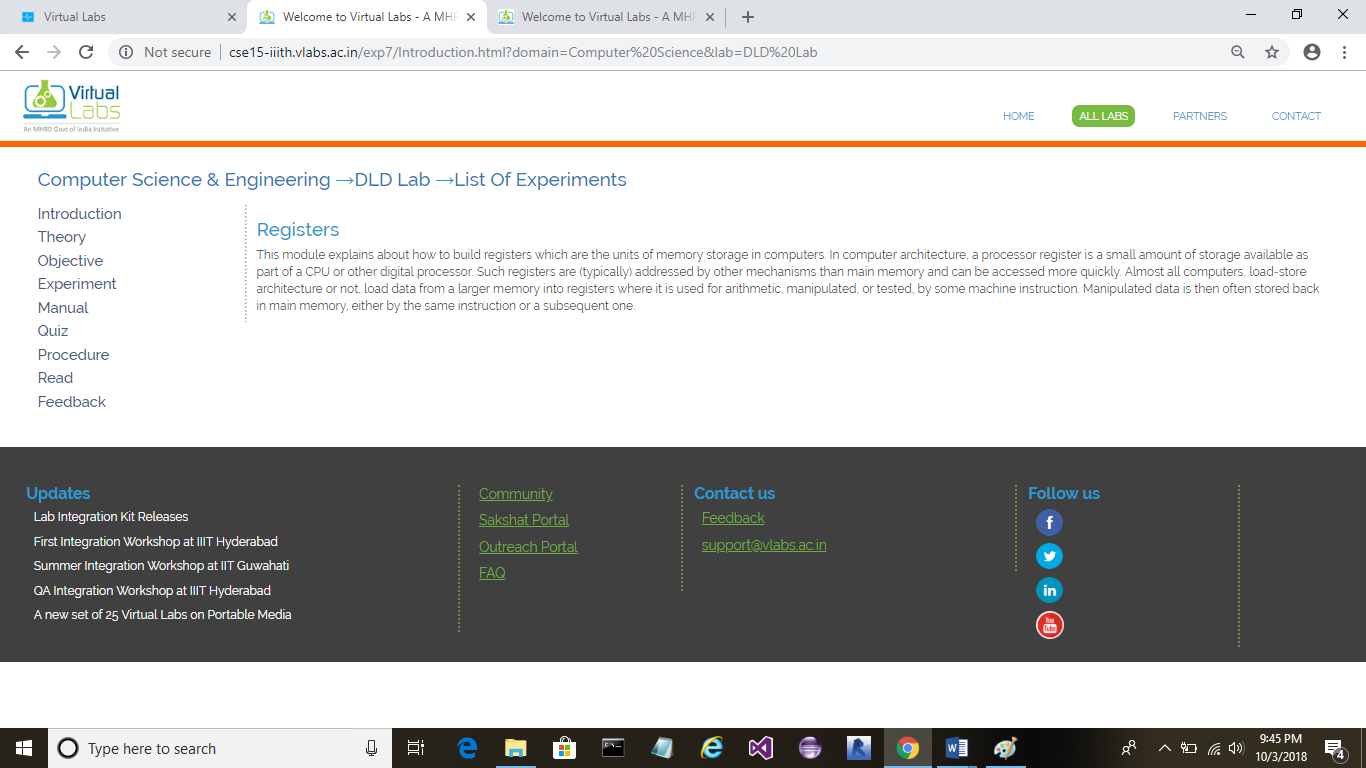
The binary data in a register can be moved within the register from one flip-flop to another.

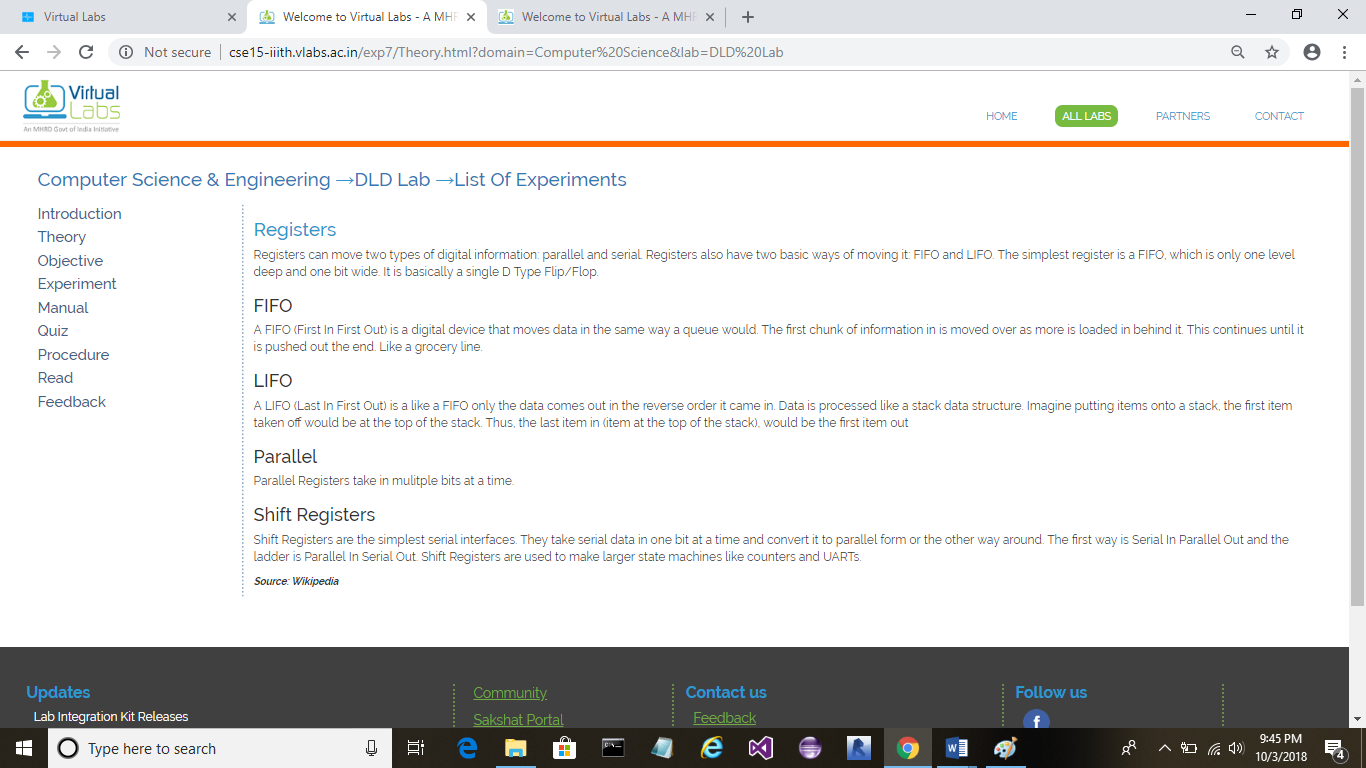
The registers that allow such data transfers are called as shiftregisters.

There are four mode of operations of a shift register.

* Serial Input Serial Output
* Serial Input Parallel Output
* Parallel Input Serial Output
* Parallel Input Parallel Output

**PROCEDURE:**





**Signature: Grade: /10**

|  |  |  |
| --- | --- | --- |
| SUBJECT:  ***Digital Electronics*** | **Perform Experiment of Latch and Flip-flop in Virtual Labs.** | |
| *Sem. III* | EXPERIMENTNO.: **11** | DATE : |

**AIM:**Perform Experiment of Latch and Flip-flop in Virtual Labs.

**THEORY:**

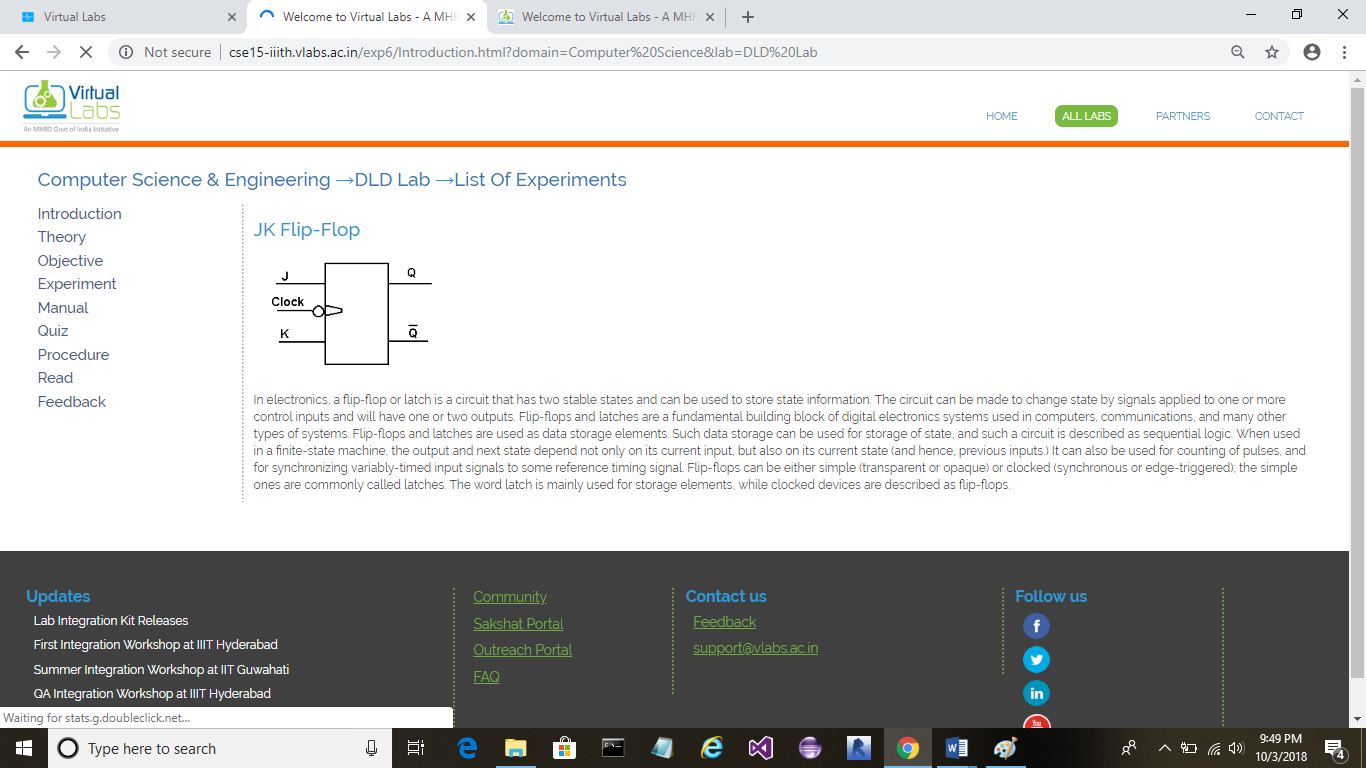
Latches and flip flops are the basic elements and these are used to store information.

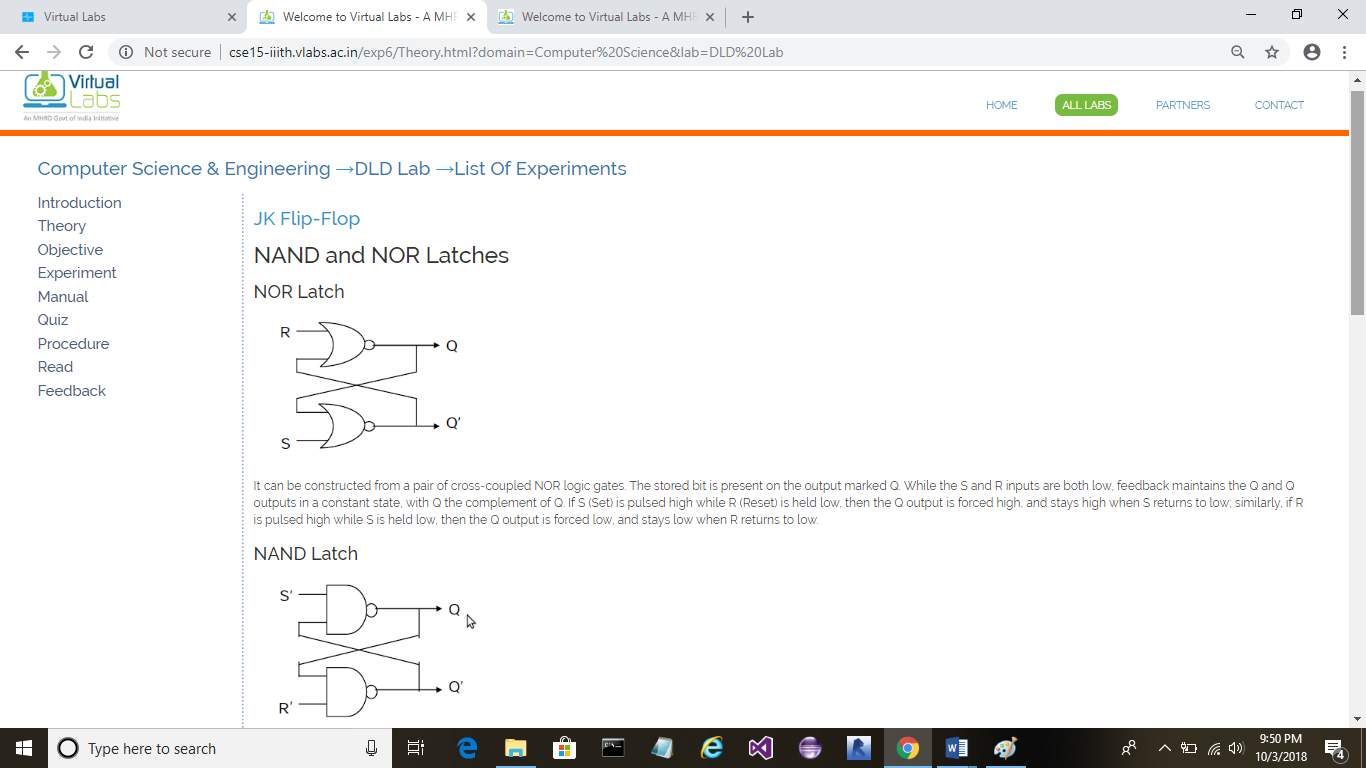
One flip flop and latch can store one bit of data.

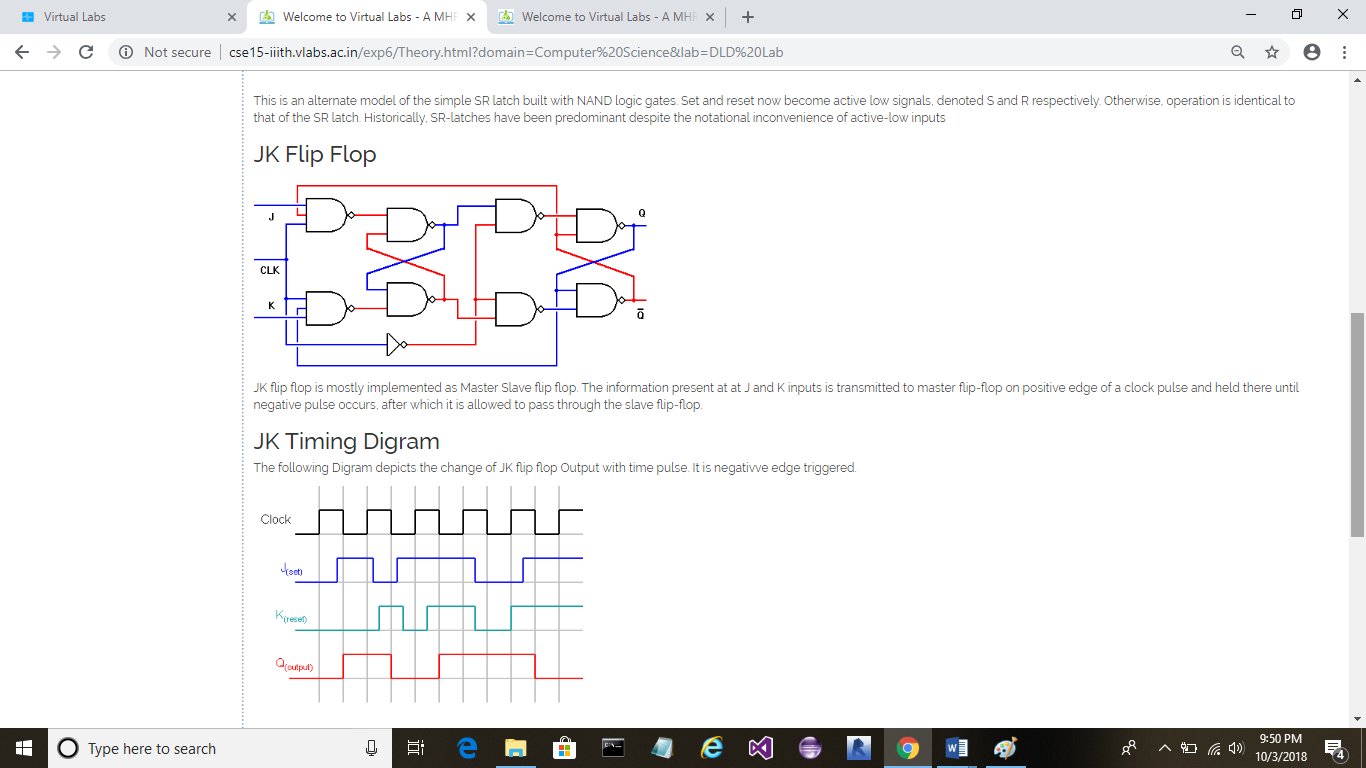
Both Latches and flip flops are circuit elements wherein the output not only depends on the current inputs, but also depends on the previous input and outputs.

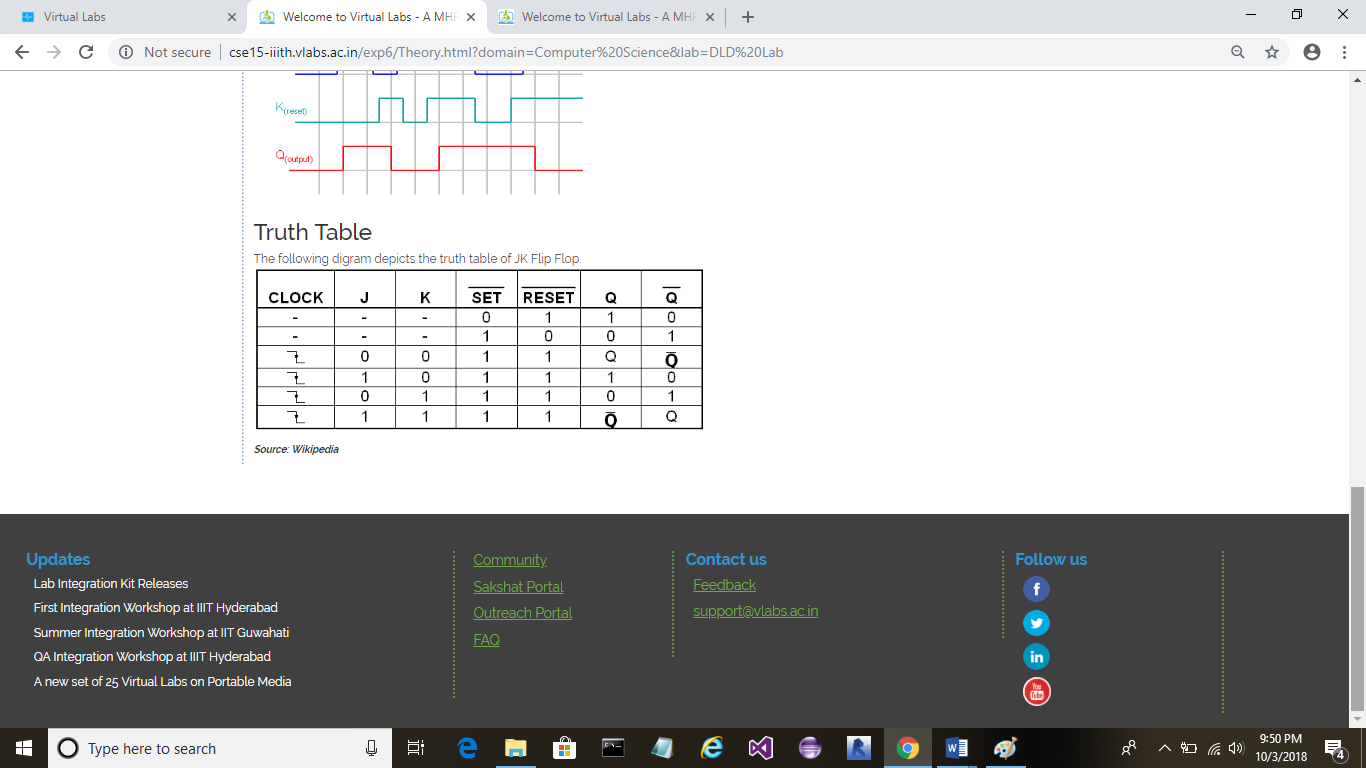
The main difference between the latches and flip flops is that, a latch checks input continuously and changes the output whenever there is a change in input. But, flip flop is a combination of latch and clock that continuously checks input and changes the output time adjusted by the clock.

**PROCEDURE:**









**Signature: Grade: /10**