MANIPAL INSTITUTE OF TECHNOLOGY
MANIPAL
(A constituent institution of MAHE, Manipal)

Department of Instrumentation and Control Engineering

VIRTUAL INSTRUMENTATION

LABORATORY MANUAL-ICE-2163

III SEM. B.Tech.

Name of the student:……

Registration Number:……...

February 2021

MANIPAL INSTITUTE OF TECHNOLOGY
MANIPAL
(A constituent institution of MAHE, Manipal)

CERTIFICATE

This is to certify that the Laboratory Manual/Journal for the lab titled **VIRTUAL INSTRUMENTATION (ICE-2163)** submitted by Mr./Ms._____ (Reg. No :_____) of third semester, Electronics and Instrumentation Engineering for the academic year _____ , as per laboratory course requirements, which has been evaluated and duly certified.

Place: Manipal
Date:

Lab In-Charge

CONTENTS

Evaluation Plan:

Continuous Evaluation : **60%** (Preparation, Lab performance, Journal, Assignments, and Regularity)

End Semester Lab Test: **40%**

EXPERIMENT NO.1	Date:

Introduction to LabVIEW

Aim: To understand the principles of Virtual Instrumentation.

Theory:

Virtual Instrumentation: Virtual instrumentation is the use of customizable software and modular measurement hardware to create user-defined measurement systems, called virtual instruments. Traditional hardware instrumentation systems are made up of pre-defined hardware components, such as digital multimeters and oscilloscopes that are completely specific to their stimulus, analysis, or measurement function. Because of their hard-coded function, these systems are more limited in their versatility than virtual instrumentation systems. The primary difference between hardware instrumentation and virtual instrumentation is that software is used to replace a large amount of hardware. The software enables complex and expensive hardware to be replaced by already purchased computer hardware; e. g. the analog-to-digital converter can act as a hardware complement of a virtual oscilloscope, a potentiostat enables frequency response acquisition and analysis in electrochemical impedance spectroscopy with virtual instrumentation.

LabVIEW: Laboratory Virtual Instrument Engineering Workbench (LabVIEW) is a graphical programming language that uses icons instead of lines of text to create applications. In contrast to text-based programming languages, where instructions determine program execution, LabVIEW uses dataflow programming, where the flow of data determines execution. In LabVIEW, a user interface can be built by using a set of tools and objects. The user interface is known as the front panel. Then code can be added using graphical representations of functions to control the front panel objects. The block diagram contains this code. In some ways, the block diagram resembles a flowchart.

LabVIEW programs are suitable for virtual instruments or VIs because their appearance and operation imitate physical instruments, such as oscilloscopes and multimeters. Every VI uses functions that manipulate input from the user interface or other sources and display that information or move it to other files or other computers.

A VI contains the following three components:

Front panel - Serves as the user interface.

Block diagram - Contains the graphical source code that defines the functionality of the VI.

Icon and connector panel - Identifies the VI so that the VI can be used in another VI. A VI within another VI is called a subVI. A subVI corresponds to a subroutine in text-based programming languages.

The front panel is the user interface of the VI. The front panel is built with controls and indicators, which are the interactive input and output terminals of the VI, respectively. Controls are knobs, pushbuttons, dials, and other input devices. Indicators are graphs, LEDs, and other displays. Controls simulate instrument input devices and supply data to the block diagram of the VI. Indicators simulate instrument output devices and display data the block diagram acquires or generates. After the front panel is built, add code using graphical representations of functions to control the front panel objects. The block diagram contains this graphical source code. Front panel objects appear as terminals on the block diagram.

Additionally, the block diagram contains functions and structures from built-in LabVIEW VI libraries. Wires connect each of the nodes on the block diagram, including control and indicator terminals, functions, and structures.

LabVIEW Palettes

LabVIEW palettes give provide the options needed to create and edit the front panel and block diagram. The Tools palette is available on the front panel and the block diagram. A tool is a special operating mode of the mouse cursor. By selecting a tool, the cursor icon changes to the tool icon. Use the tools to operate and modify the front panel and block diagram objects.

Select Window » Show Tools Palette to display the Tools palette.

The Tools palette can be placed anywhere on the screen. If automatic tool selection is enabled and as the cursor is moved over objects on the front panel or block diagram, LabVIEW automatically selects the corresponding tool from the Tools palette.

The Controls palette

The Controls palette is available only on the front panel. The Controls palette contains the controls and indicators used to create the front panel. to display the Controls palette, Select Window » Show Controls Palette or right-click the front panel workspace. The Controls palette can be placed anywhere on the screen.

The Functions palette

The Functions palette is available only on the block diagram. The Functions palette contains the VIs and functions used to build the block diagram. To display the Functions palette, select Window » Show Functions Palette or right-click the block diagram workspace. The Functions palette can be placed anywhere on the screen.

Dataflow Programming

LabVIEW follows a dataflow model for running VIs. A block diagram node executes when all its inputs are available. When a node completes execution, it supplies data to its output terminals and passes the output data to the next node in the dataflow path.

Procedure:

1. Open NI LabVIEW and press <Ctrl_N> to open a blank VI.
2. Press <Ctrl_T> to tile the front panel and block diagram windows.

EXPERIMENT NO.2	Date:

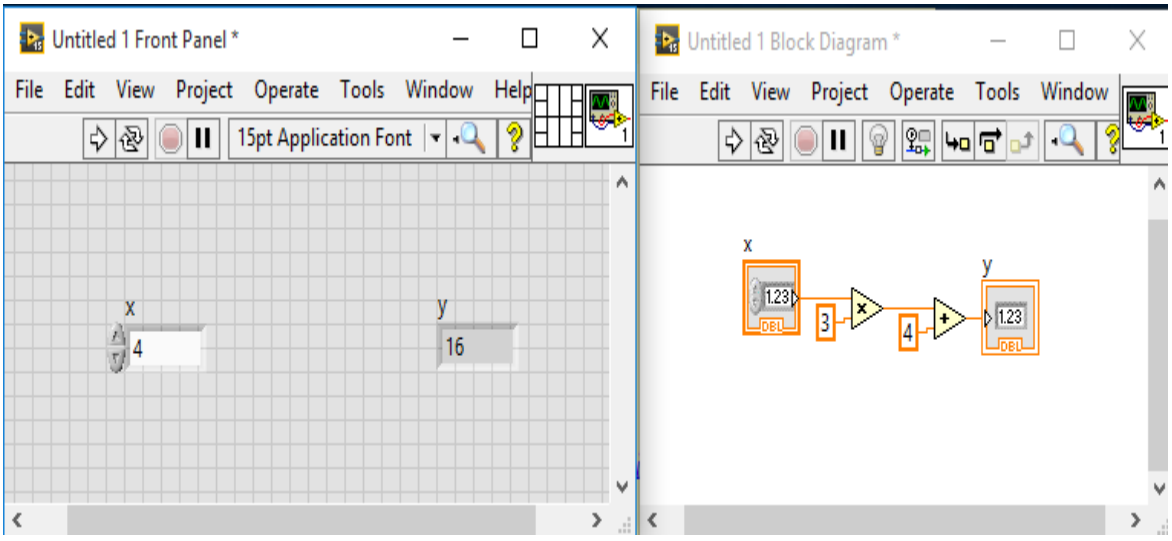
Arithmetic and logical operations

Aim: To perform basic arithmetic and logical operations using LabVIEW.

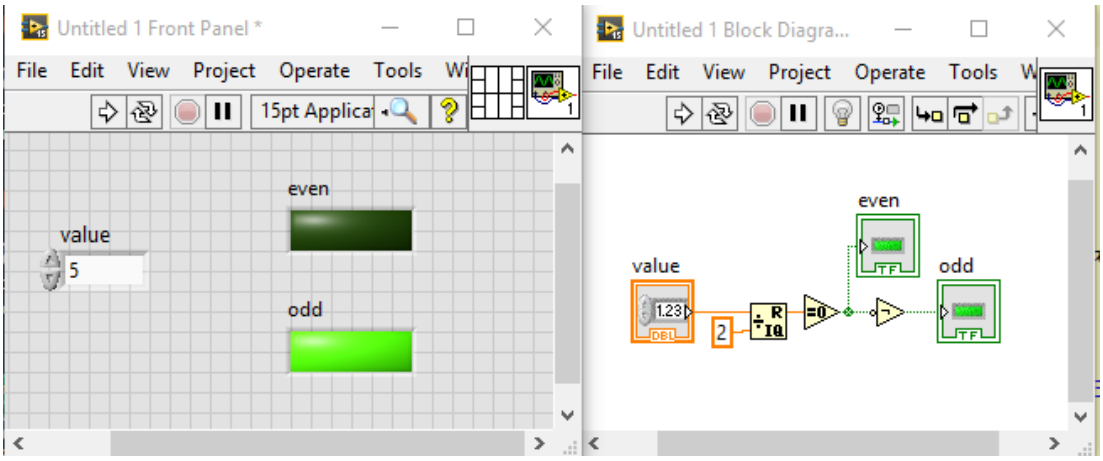
Instructions:

Step 1: Select controls by right-clicking on the front panel and choose the specific type of control similarly choose indicators.
Step 2: Different operators such as addition, subtraction, multiplication, less than, greater than, Boolean, etc. can be performed by right-clicking on the block diagram.
Step 3: Using wires inputs and outputs are connected to the respective operators in the block diagram panel.
Step 4: Enter the input values in the front panel.
Step 5: Observe the output by performing a single step and continuous mode execution.

Example 1: Execute a VI to perform the operation of y=3x+4.



Example 2: Execute a VI to perform the operation of checking a given number is odd or even



Exercise:

1. Write a program to convert inches to meters.
2. Write a program to perform y=4.5a*logb+$10^{(a+b)}$ using functions and formula node.
3. Write a program to perform a Boolean operation $Y = \overline{(A \oplus B)} + (C.D.A) + \overline{A}$
4. Compute the take-home salary of a person whose basic is 'X', DA is 20% of 'X', HRA is 15% of 'X', PI of Rs. 1200, and IT @5%.
5. Write a VI to convert a given complex number from rectangular to polar and vice versa.

EXPERIMENT NO.3	Date:

Modular programming

Aim:

To perform modular programming in LabVIEW.

Instructions:

A. Creating SubVI

Step 1: Build a VI with a front panel and block diagram
Step 2: Assign terminals to control and indicators using the control pane.
Step 3: Create an icon
Step 4: Save the file with extension .vi
Step 5: Call the file in the main program using the 'select vi' option in the block diagram pane.
Step 6: If multiple SubVI are to be used group can be saved as .llb (LabVIEW library)

B. Creating EXE from a VI

Step 1: Right-click 'build specifications' in the 'project explorer' window and select 'New'>> 'Application (EXE)' from the shortcut menu
Step 2: Modify the file name of the target and destination directory for the application in the 'Application Information category'.
Step 3: Select the 'Application Information category'.
Step 4: Change the target filename to your 'filename .exe'.

Exercise:

1. Create a vi to perform the operation a^3+b^3, using this vi as subvi perform $(a+b)^3$.
2. Write a program to find the average of two numbers. Using this subvi compute the average of the class.
3. Write the program to compute the area and using this subvi write a program to compute volume.
4. Write a program for NAND operation, using it as subvi write a program to find XOR.

EXPERIMENT NO.4	Date:

Loops and shift registers

Aim:

To perform the operations using loops in LabVIEW.

Instructions:

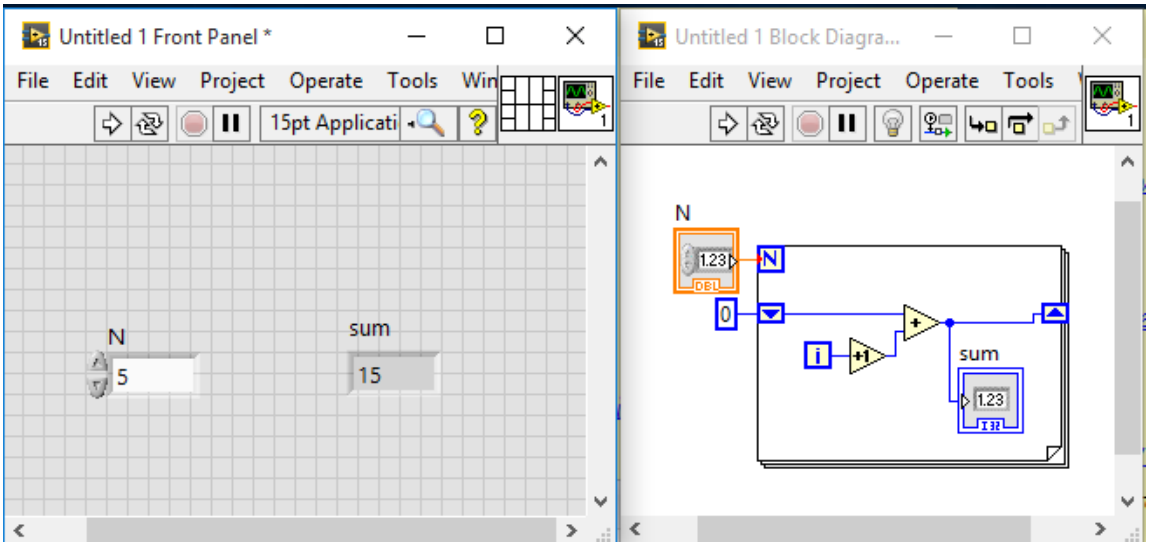
A. While loop: to be used when a condition needs to be checked after an iteration.

Step 1: In the block diagram panel, right-click and select 'Programming'>> 'structures' >> 'while loop'.
Step 2: choose the terminating condition.

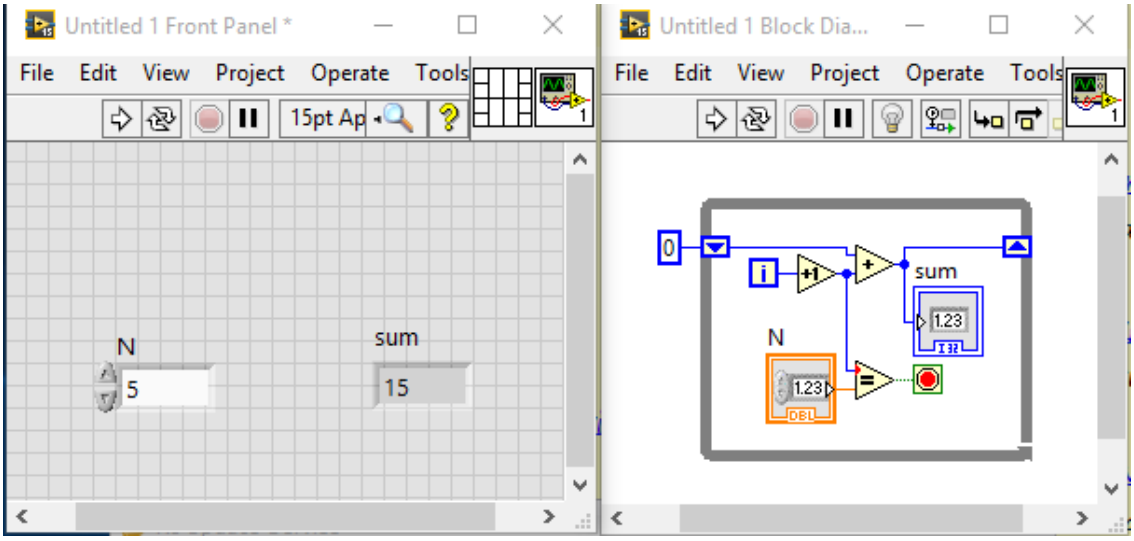
B. **For loop:** to be used when a condition needs to be checked before every iteration.

Step 1: In block diagram panel, right click and select 'Programming' >> 'Structures' >> 'for loop'.
Step 2: Assign the number 'N', which means the number of times loops should be executed.

Example 1: Compute the sum of the first N numbers.



Using 'For loop'



Using ‘while loop’

Exercise:

1. Find the sum of first ‘N’ odd numbers
2. Write a program to compute ${}^{N}P_{r}$ and ${}^{N}C_{r}$.
3. Find whether a given number is prime or not.
4. Find the sum of values lying between ‘P’ and ‘Q’. Where ‘P’ and ‘Q’ are positive integers and P>Q.
5. Create a VI to change the state of the Boolean indicator ‘n’ times between TRUE and FALSE using feedback node.

EXPERIMENT NO.5	Date:

Feedback node and communicating among loops

Aim:

To perform operations using the feedback node of LabVIEW and to communicate between multiple loops.

Instructions:

A: Feedback Node - The feedback node stores data when the loop completes an iteration, sends that value to the next iteration of the loop, and transfers any data type.

Step 1: Select 'Feedback Node' on the structures palette and place it in for or while loop.

Step 2: Initialize a feedback node by right-clicking the feedback node and select 'initializer terminal' from the short cut menu. Add a wire from outside the loop to the 'initializer terminal'.

B: Control timings- LabVIEW allows to add wait function in the loop, to wait an amount of time in the millisecond before the loop executes the next iteration.
To select the wait function right click on the block diagram, select 'Programming' >> 'timing'.

C: Communicating among multiple loops
Local variables transfer data within a single vi.
Right-click on an object's terminal and select 'create'>> 'local variable'.

Global variables are built-in LabVIEW objects, which are used to access and pass data among several vi's that run simultaneously.
Right-click on the block diagram, select 'structure' >> 'global variable', in the pop-up window select the variable.

Exercise:

1. Create a VI which converts a decimal number to a binary number.
2. Create a VI which will increment a counter every time the up button is pressed on decrements the counter value every time the down button is pressed. A LED glows if the counter is equal to zero.
3. Design a counter which displays first N even numbers with a time interval of 10 seconds.
4. Write a program in LabVIEW to solve a quadratic equation to check all possibilities of their coefficients ($ax^2 + bx + c = 0$).

EXPERIMENT NO.6	Date:

Arrays

Aim:

To perform the array operations in LabVIEW

Instructions:

Creating a one-dimensional array control/indicators/constant
Step 1: In front panel right click >>'Controls' >> 'Modern'>>'Array, Matrix & Clusters palette'.
Step 2: Insert an object in the array. The array data type will be assigned as that of the object inserted.
Step 3: To create a multidimensional array, Right-click on the array and select 'Add Dimension'.
Step 4: From the array particular set of elements can be made visible by varying the values of 'index' provided next to the array.

Exercise

1. Display 10 random integers between 2 and 50.
2. Perform AxB where 'A' and 'B' are 3x3 matrices.
3. Write a program to arrange a given set of numbers in ascending and descending order.
4. Create a VI to read a set of numbers and store it on the one-dimensional array and again read a number 'n', and check whether it is present in the array. If it is so, print out the position of 'n' in the array and also check whether it repeats in the array.
5. Display a square matrix of 5 rows and column elements, such that the addition of individual row elements should be the same and so the column elements.

EXPERIMENT NO.7	Date:

Clusters

Aim:

To perform the operations on clusters in LabVIEW.

Clusters are a group of data with elements of mixed type.

Instructions:

To create clusters, select cluster by right-clicking in the front panel and select 'controls'>>'All controls'>>'Arrays & cluster palette', place it on the front panel. Drag and drop objects or elements into the cluster.

Creating Cluster control and indicators- Select a cluster on the *Controls>>All Controls>>Arrays & Cluster* palette, place it on the front panel and drag a data object or element, which can be a numeric, Boolean, string, path, refnum, array, or cluster control or indicator, into the cluster shell.

Exercise:

1. Create a VI to compare clusters and Switch ON an LED in the output cluster if the *n*th element of cluster 1 is greater than the *n*th element of cluster 2.
2. Build a cluster control that consists of a seven-segment LED display, a switch, a string control, and numeric control. Split the cluster elements using the *Unbundle* function and alter the values of some of the cluster controls. Bundle them again and display them in a cluster indicator.
3. Create a database of the class consisting of Roll no, Reg No, Name, marks obtained. Process the data to display the database in terms of ascending order of marks for a particular batch. (Note: section can contain more than one batch of students; batch is indicated in first two digits of Reg no.).

EXPERIMENT NO.8	Date:

Graph & chart

Aim:

To plot the data on graphs and charts in LabVIEW.

Instructions:

Plotting Data:

1. Graph: display typical data
2. Chart: displays data at a continuous rate
    a. Strip chart: shows continuous data and continuously scroll from left to the right
    b. Scope chart: shows one item, and scroll partway across left to right.
    c. Sweep chart: shows old data on the right and new data on left.
3. XY graph: two-dimension plot
4. 3D plot
5. Intensity graph:

Right-click on the front panel, select 'graphs and charts'.

Exercise:

1. Build a VI that displays two plots, a random plot and a running average of the last four points, on a waveform chart in sweep update mode.
2. Build a VI which acts as a function generator.
3. Plot the equations $y1=3x^2+2x-1$ and $y2 = x+3$, in steps of 0.1 from range 0 to 5. Also check if the y1 and y2 meet, if they meet display their coordinates.
4. Build a VI to plot a circle in the *XY* graph using a For Loop

EXPERIMENT NO.9	Date:

Strings and File I/O

Aim:

To perform the operations on strings and file I/O in LabVIEW.

The string is a sequence of displayable or non-displayable ASCII characters.
String control and string indicator are located on the 'Controls'>> 'Text controls'>>'Text indicators palettes' in the front panel.

File I/O records or reads data in a file. Functions related to file I/O is located by right-clicking the block diagram 'Functions'>>'All functions'>> 'File I/O'.

Exercise:

1. Find the length of the given string A, if its length is less than 'N' concatenate string A with string B. Then find the final length.
2. Read the string from file xx.txt convert the text in the file to uppercase and store the resultant in yy.txt.
3. Read the string from file xx.txt and convert the paragraph in the file to the multiline text of 11 words each. Further, save the result in yy.txt.
4. Check whether the given text is a palindrome or not.
5. Create a VI to open and read a particular file. Display the file path and contents of the file. Display the numeric data and strings in separate files.

EXPERIMENT NO.10	Date:

Structures

Aim:

To perform the operations using structures in LabVIEW.

Structures are used in the block diagram to repeat blocks of code and to execute code conditionally or in a specific order. Section of block diagram inside the structure border is called 'subdiagram'. The terminals that feed data into and out of structures are called 'tunnels'.

Following structures can be used in LabVIEW

1. Loops
2. Case structure
3. Sequence structure
4. Event structure
5. Timed structure
6. Diagram disable structure
7. Conditional disable structure

To add structures right-click on the block diagram, select 'programming'>> 'structures'.

Exercise:

1. Create a program to perform the following, a. y= A+B+C if 0<N<5, b. $y=A^2+2B-C$ if 6<N<20, else y=0.
2. In a given set 'N' consisting of 20 numbers, create a set 'M' which contains only numbers greater than a given threshold.
3. Execute the given equation $y = \{(A^2-B^2)x(B+C)x(C-A)^2\}-2$ using stacked sequence and flat sequence.
4. Execute a program to compute y = Asinx+Bcosx(log[1/(p-1)]). If the resultant is greater than 5, the program should halt and display an error message.