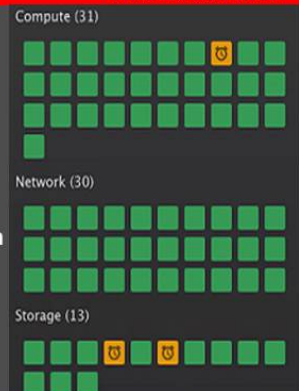


VIRTUAL INSTRUMENTATION & INTRODUCTION TO LABVIEW PROGRAM EXPERIMENTS

First Edition 2019

G.Ramachandran
 T.Sheela
 T.Muthumanickam
 S.Kannan
 A.Malarvizhi
 N.Manikanda Devarajan
 G.Sureshkumar
 G.Murali
 PM.Murali



Rs. 50

ISBN



9 788193 464458



Published By...
 Tamil Nadu Scientific Research Organisation
 # 39, Koodal Nagar, Rajagopalapuram Po, Pudukkottai - 622 003
 Tamil Nadu, India. Ph 04322 261088, 96528 86637.
 www.tnsroindia.org.in

VIRTUAL INSTRUMENTATION & INTRODUCTION TO LABVIEW PROGRAM EXPERIMENTS

**VIRTUAL INSTRUMENTATION &
INTRODUCTION TO LABVIEW PROGRAM EXPERIMENTS**

**G.Ramachandran
T.Sheela
T.Muthumanickam
S.Kannan
A.Malarvizhi
N.Manikanda Devarajan
G.Sureshkumar
G.Murali
PM.Murali**

**Department of Electronics and Communication Engineering
Vinayaka Mission's Research Foundation
Vinayaka Mission's Kirupananda Variyar Engineering College
Salem - 636 308, TamilNadu, India.**



TAMILNADU SCIENTIFIC RESEARCH ORGANISATION

Plot No.: 39, Mura Bhavan, Koodal Nagar, Rajagopalapuram Post,
Pudukkottai-622003, TamilNadu, India.Ph:04322-261088, Mobile:9952886637.
E-Mail:tnsroindia@gmail.com, web:www.tnsroindia.org.in

Science series 2019

TNSRO Publication No:026/2019

Copy Right : Authors



First Edition: March 2019

Explanation to Front Cover: Virtual experiments lab view.

ISBN : 978-81-934644-5-8

Typeset and Printed at Malligai Printers, Pudukkottai.

Rs.50/-

- No part of this publication may be reproduced, stored in a retrieval system or transmitted, in any form or by any means, mechanical, photocopying, recording or otherwise, without prior written permission of the publisher.

Published by: Publication Division, TamilNadu Scientific Research Organisation
#39,Koodal Nagar, Rajagopalapuram post,Pudukkottai-622003.
TamilNadu,INDIA.Off. No. 04322 261088. E.Mail:tnsroindia@gmail.com
Visit us: www.tnsroindia.org.in

CONTENTS

LIST OF EXPERIMENTS

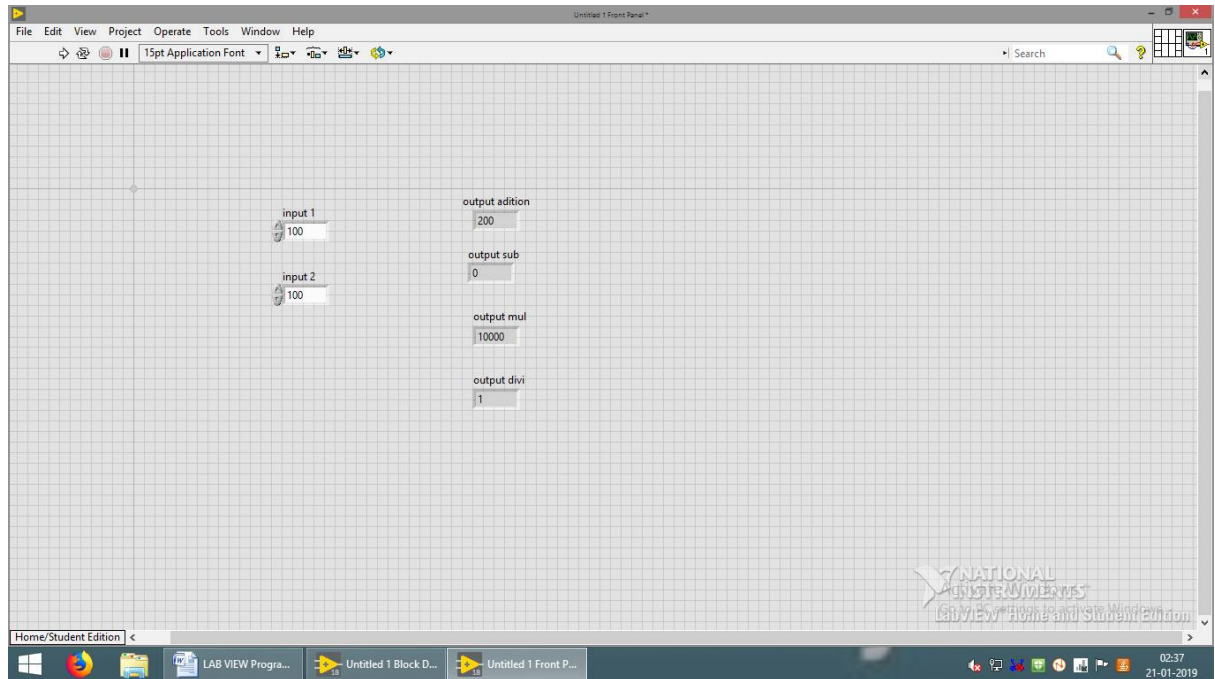
1. To perform basic arithmetic operations using Labview
2. Verification of Half Adder and Full adder.
3. Program to find Addition of First n natural numbers using for and while loop
4. Implementation of Array functions
 - 4.1 Even Numbers and Odd Number Using While Loop In An Array
5. Cluster.
6. Appendix A :-
7. Appendix B :-
8. Appendix C :-
9. Fundamentals of Lab View Questions And Answers

EXPERIMENT 1

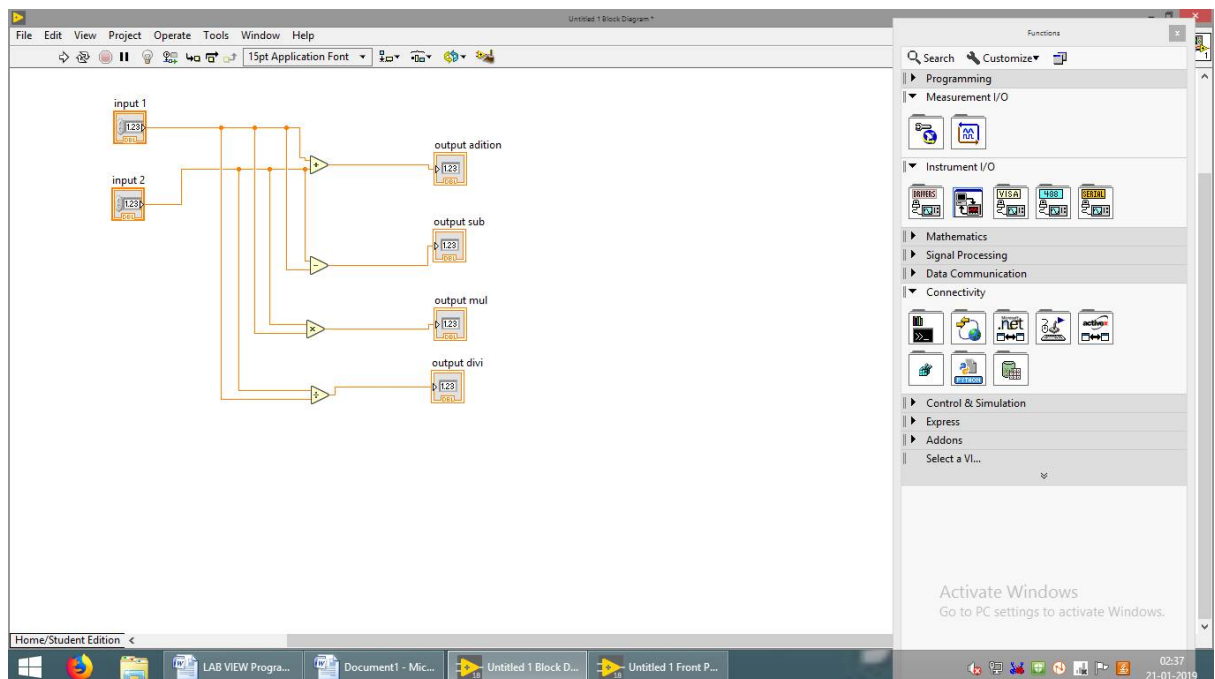
1.To perform basic arithmetic operations using Labview

The basic arithmetic operations are addition, subtraction, multiplication and division, although this subject also includes more advanced operations,

Front panel:-



Block diagram



Solution:-

Arithmetic operators:-

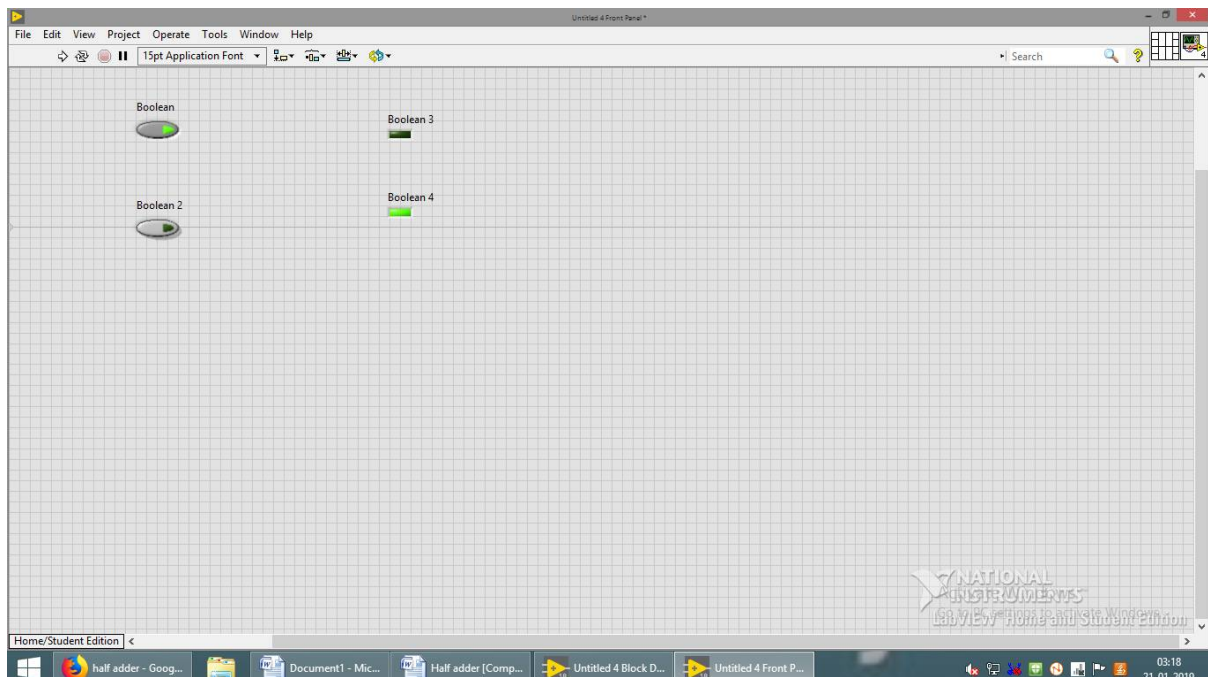
Operator	Meaning	Example
+	Addition	$4+7=11$
-	Subtraction	$12-5=7$
*	Multiplication	$6*6=36$
/	Division	$200/200=1$

EXPERIMENT 2

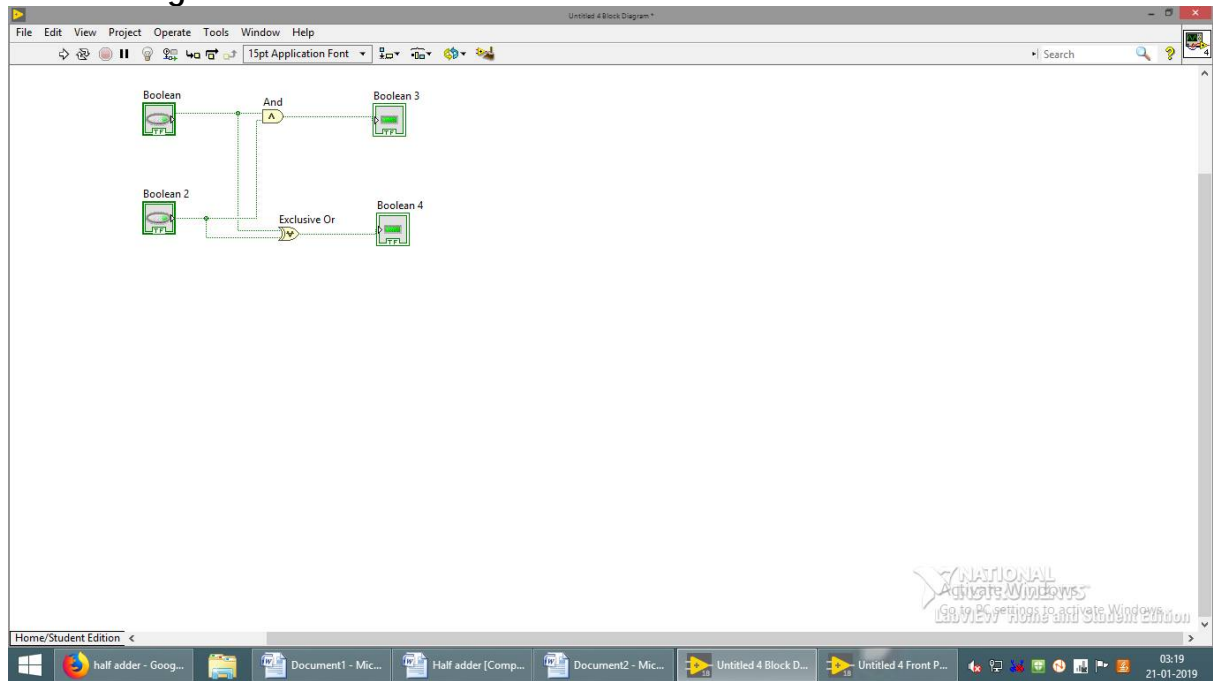
2. Verification of Half Adder and Full adder.

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. A full adder is a logical circuit that performs an addition operation on three one-bit binary numbers. The full adder produces a sum of the three inputs and carry value. It can be combined with other full adders

Front Panel



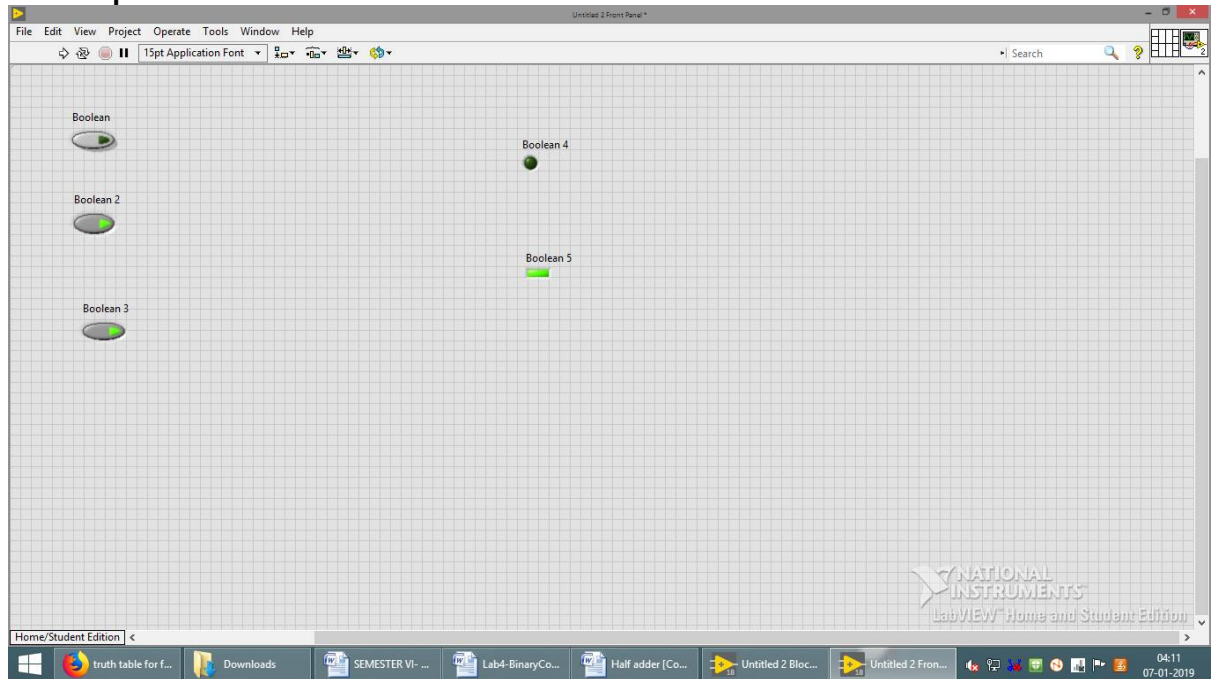
Block Diagram



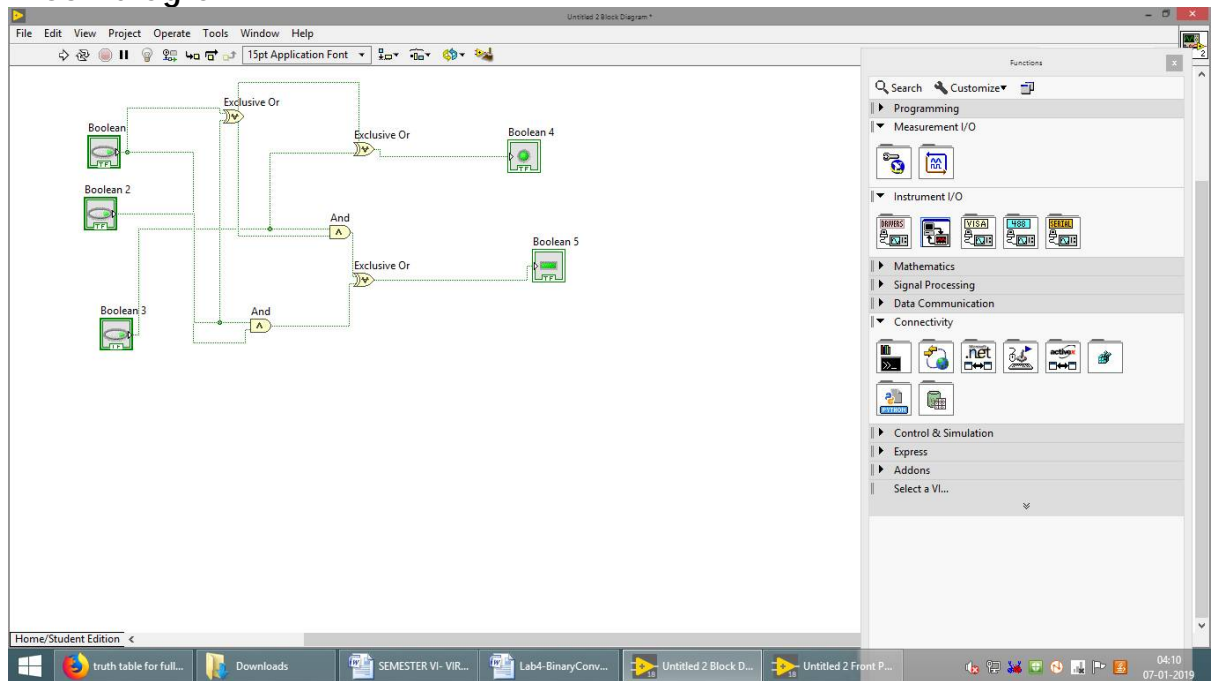
Truth Table			
A	B	Carry	Sum
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

Full adder

Front panel



Block diagram



Truth table:-

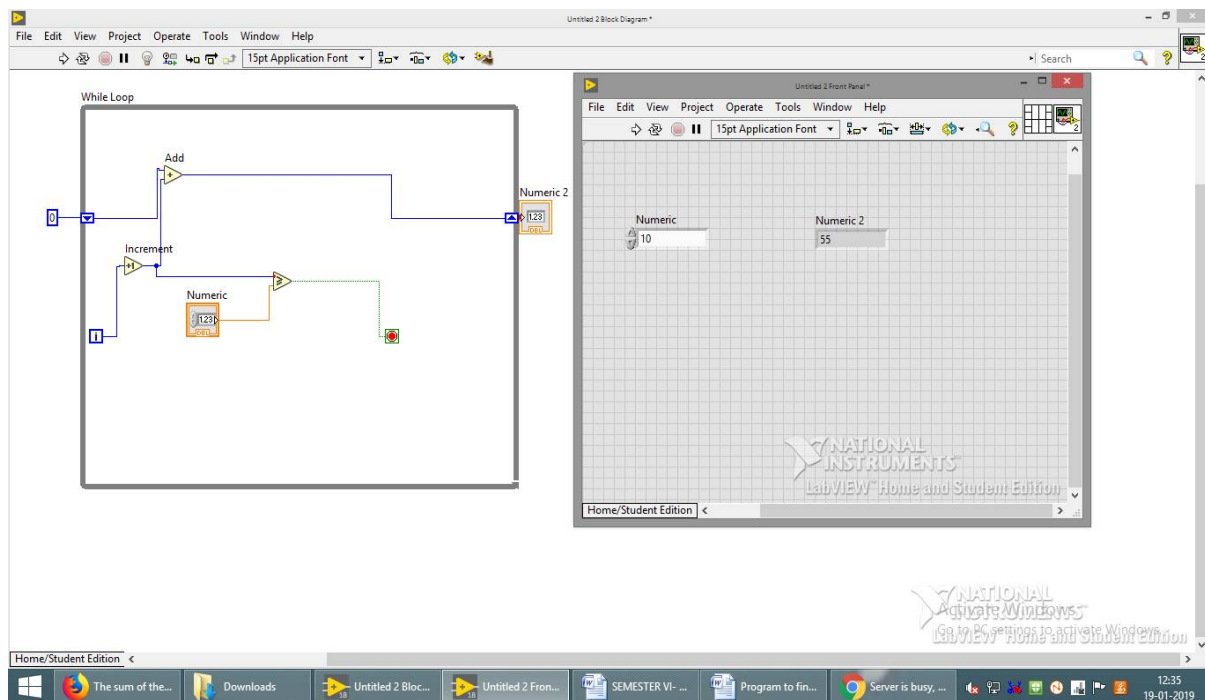
Full Adder Truth table

INPUTS			OUTPUTS	
A	B	C _{in}	SUM	CARRY _{OUT}
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

EXPERIMENT 3

3. Program to find Addition of First n natural numbers using for and while loop.

Front panel & Block diagram



Solution:-

Let's look at this problem for n=1, 2, 3, 4, and 5 and calculate the sum:

$$1=1$$

$$1+2=3$$

$$1+2+3=6$$

$$1+2+3+4=10$$

$$1+2+3+4+5=15$$

What is the formula to calculate this sum:

$$S_n = 1 + 2 + \dots + n = ?$$

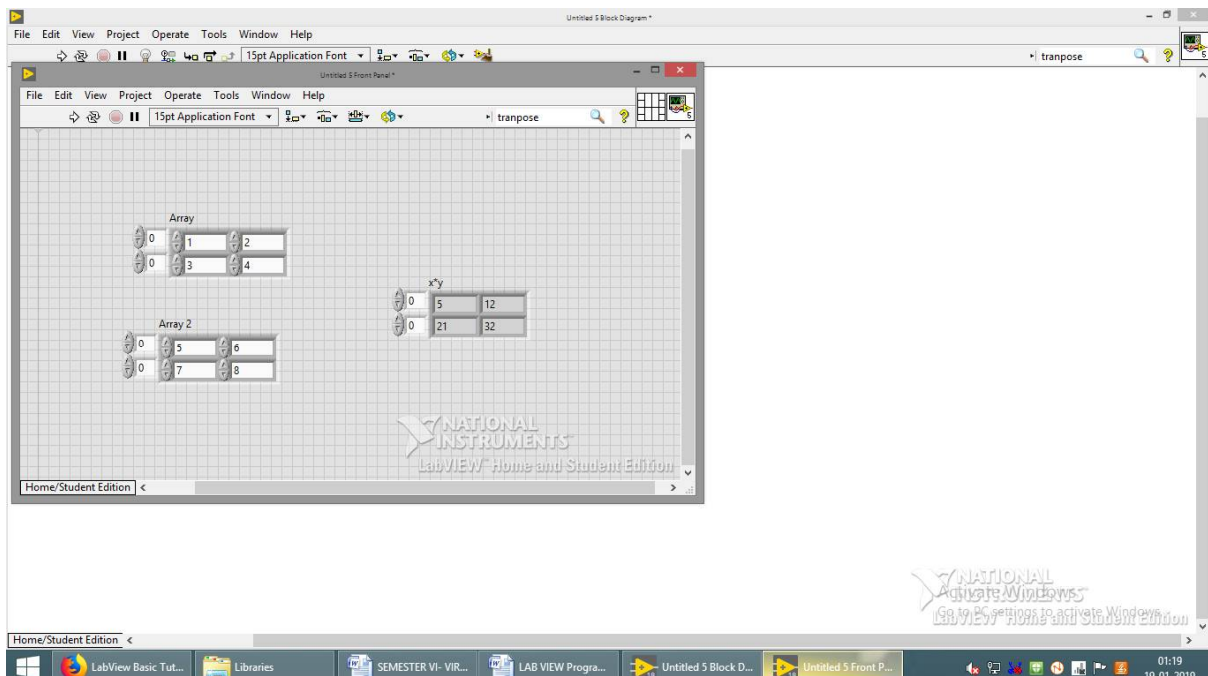
The answer is

$$S_n = \frac{n(n+1)}{2}$$

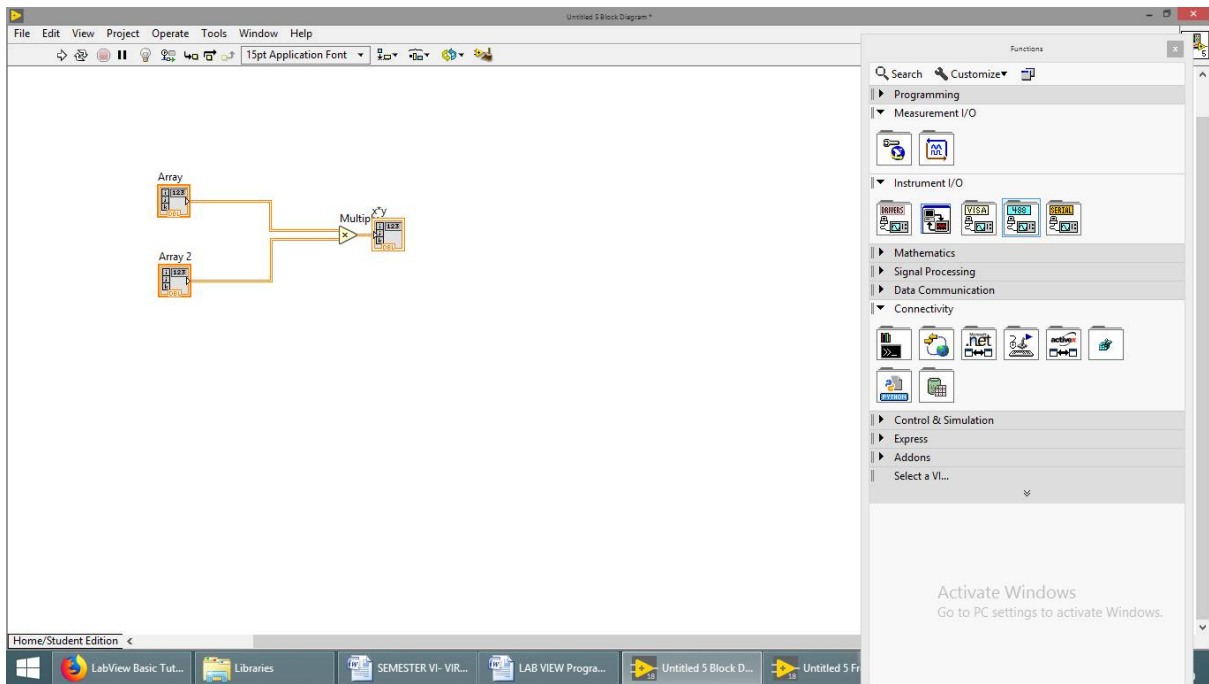
EXPERIMENT 4

4. Implementation of Array functions

Front panel

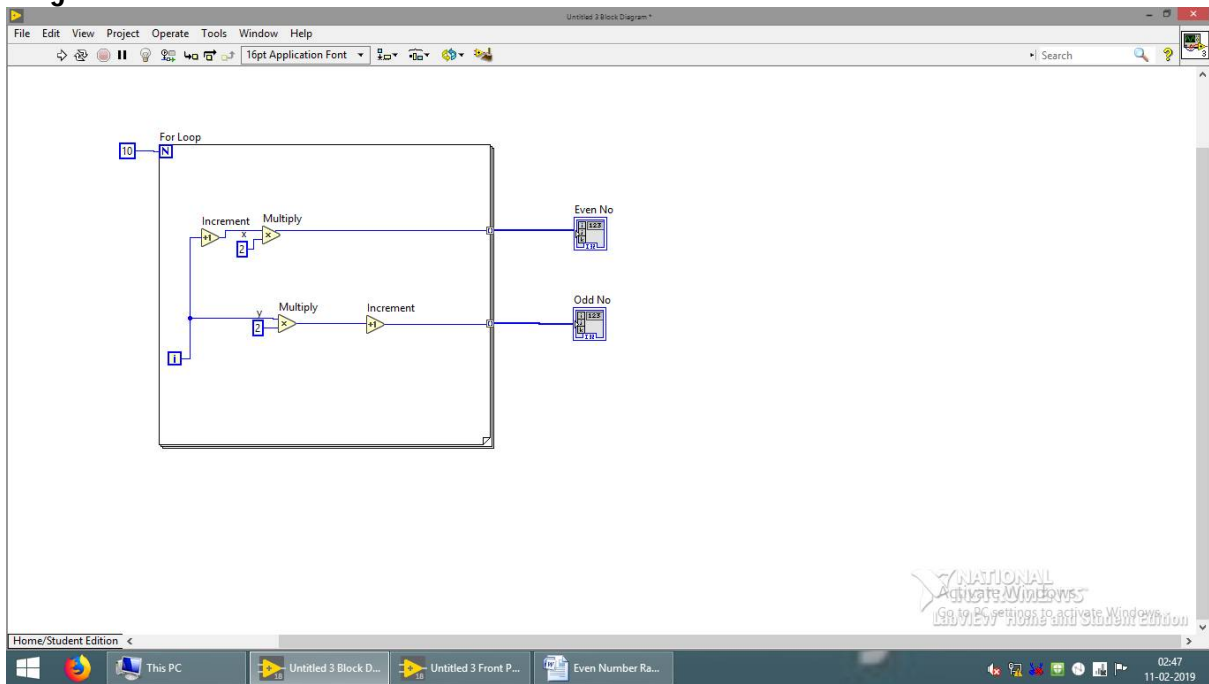


Block diagram

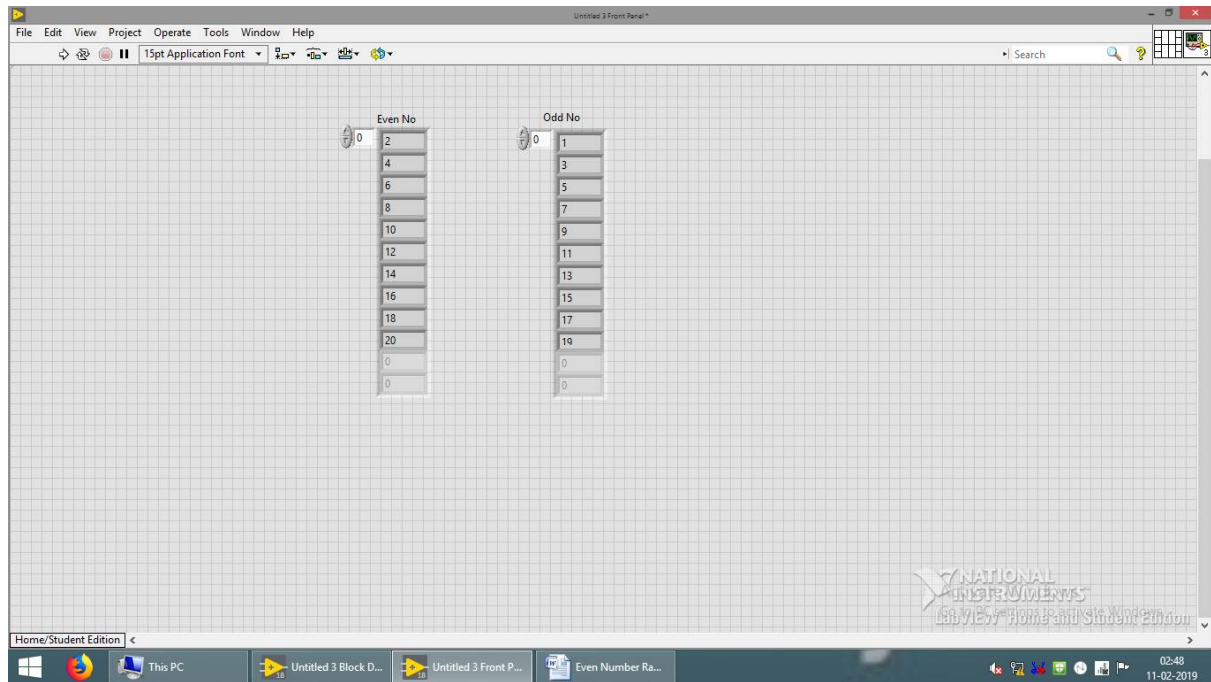


4.1 EVEN NUMBERS AND ODD NUMBER USING WHILE LOOP IN AN ARRAY

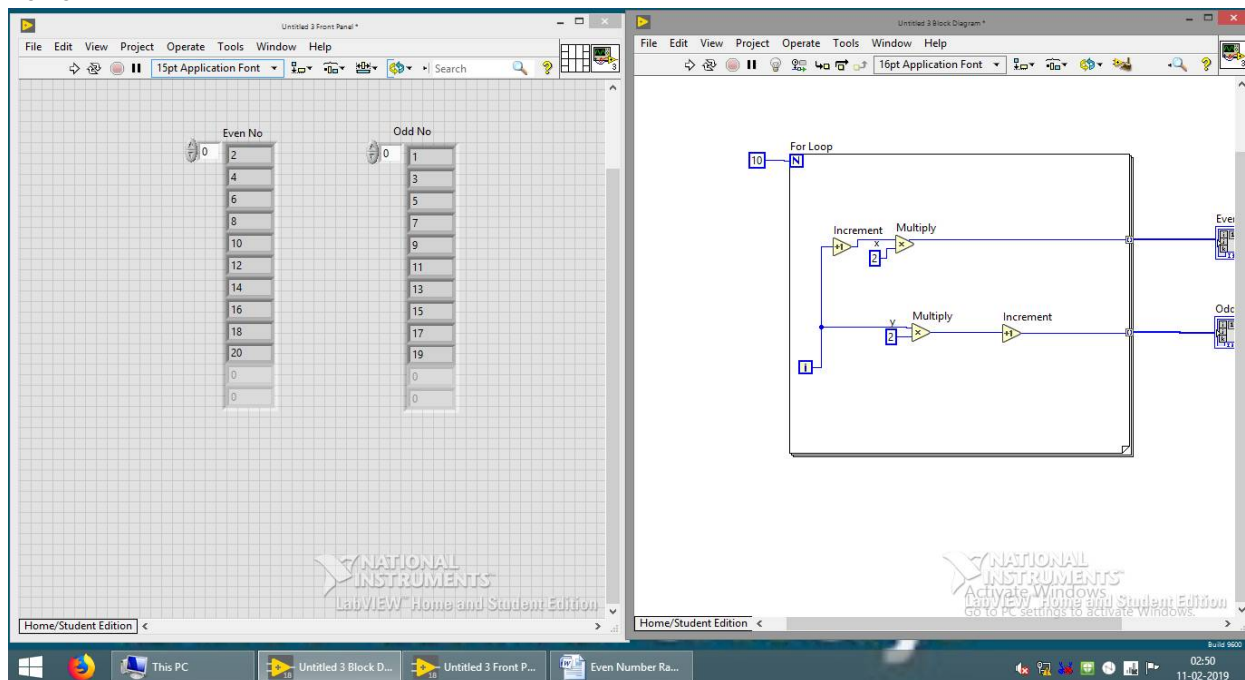
Block Diagram



Front Panel



Combined Block Diagram & Front Panel

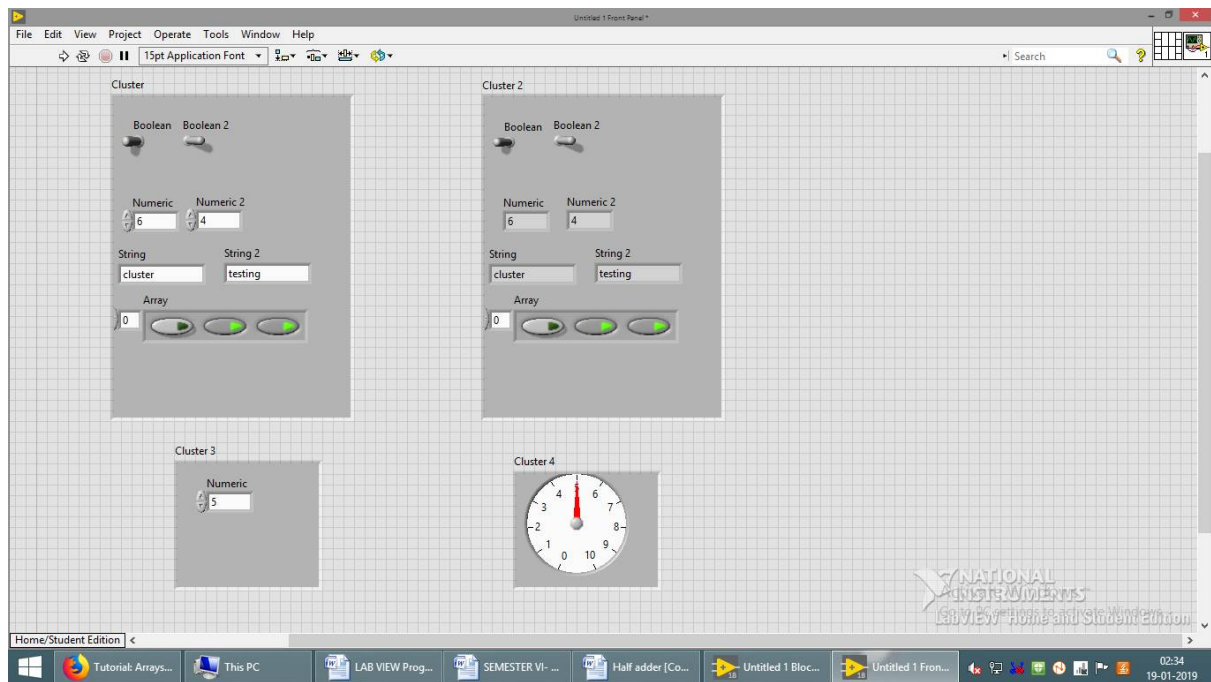


EXPERIMENT 5

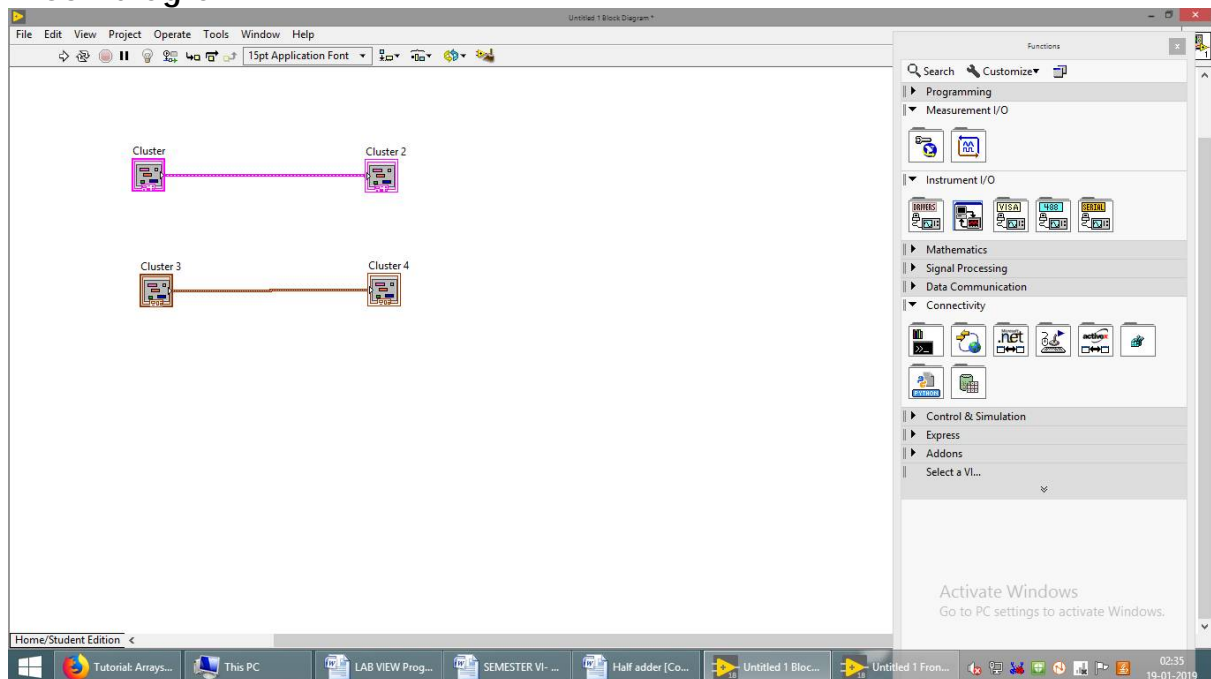
5. Cluster.

Clusters group data elements of mixed types. An example of a cluster is the LabVIEW error cluster, which combines a Boolean value, a numeric value, and a string. A cluster is similar to a record or a struct in text-based programming languages.

Front panel



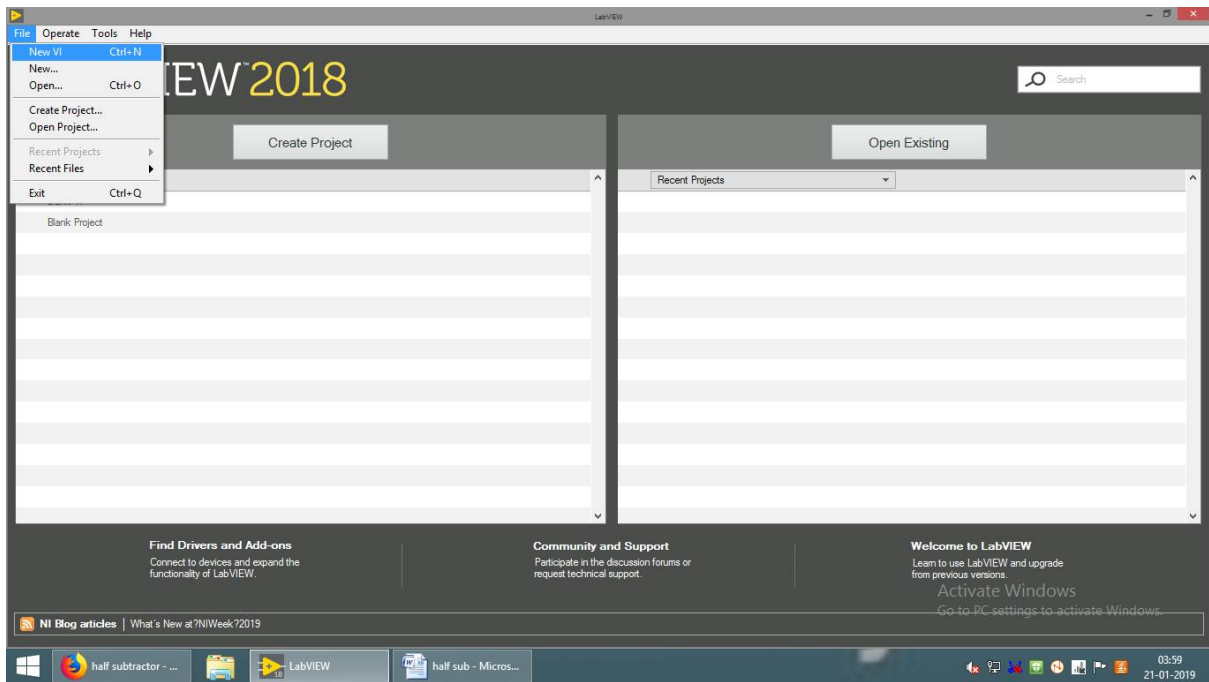
Block diagram



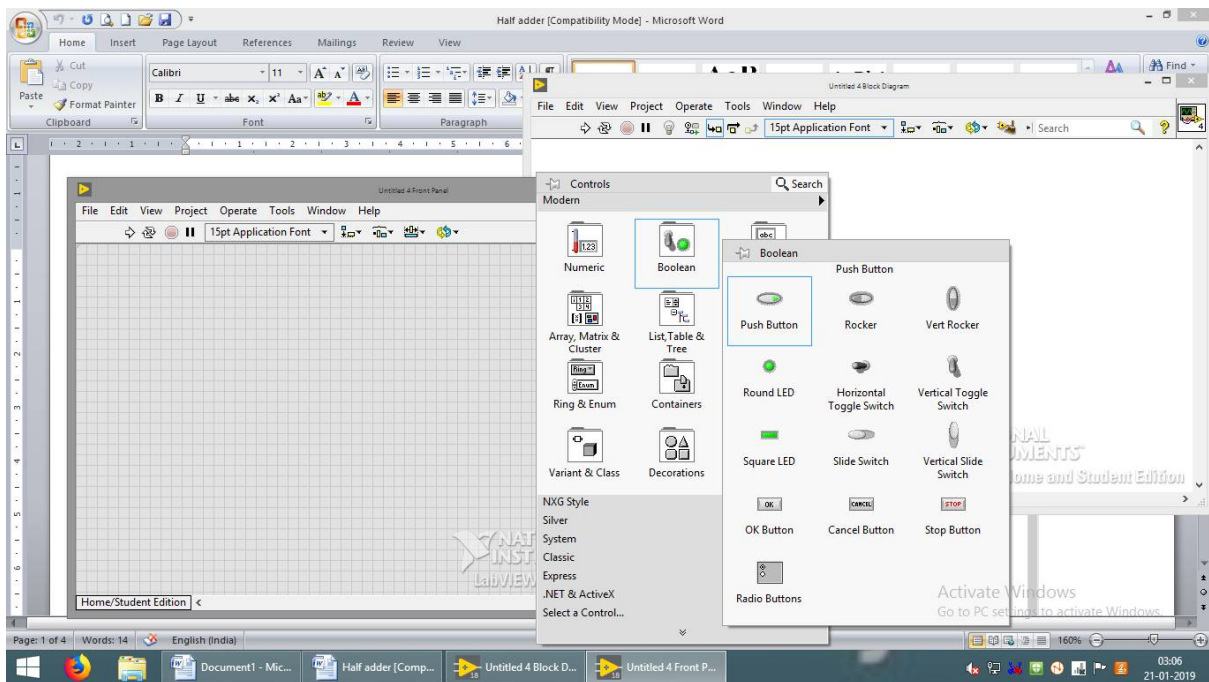
6. Appendix A :-

STEP 1:-

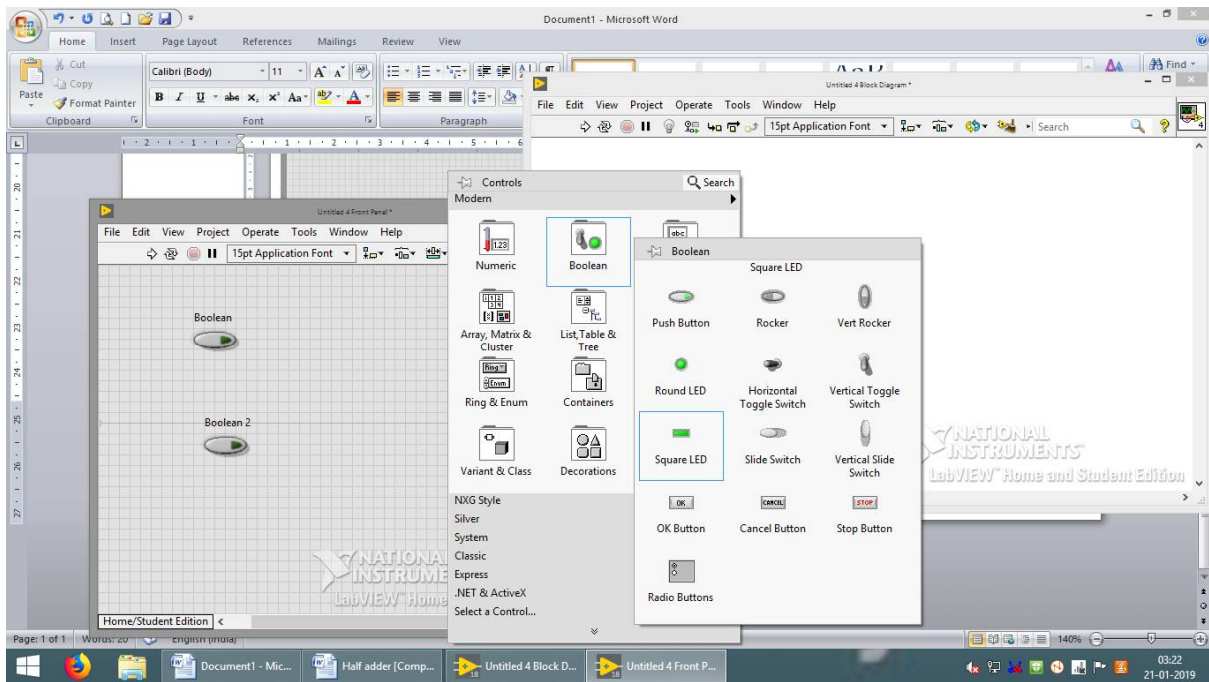
File New VI (Virtual Instrumentation)



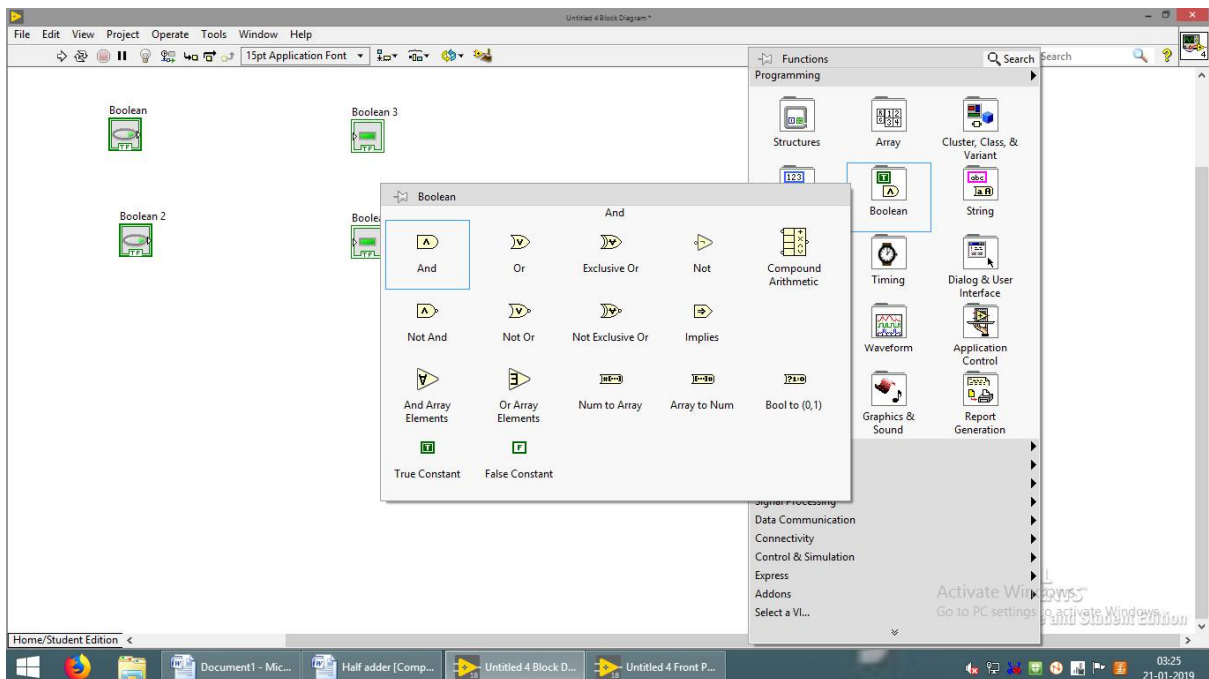
STEP 2:-
Right click on the front panel and select Boolean and select push button.



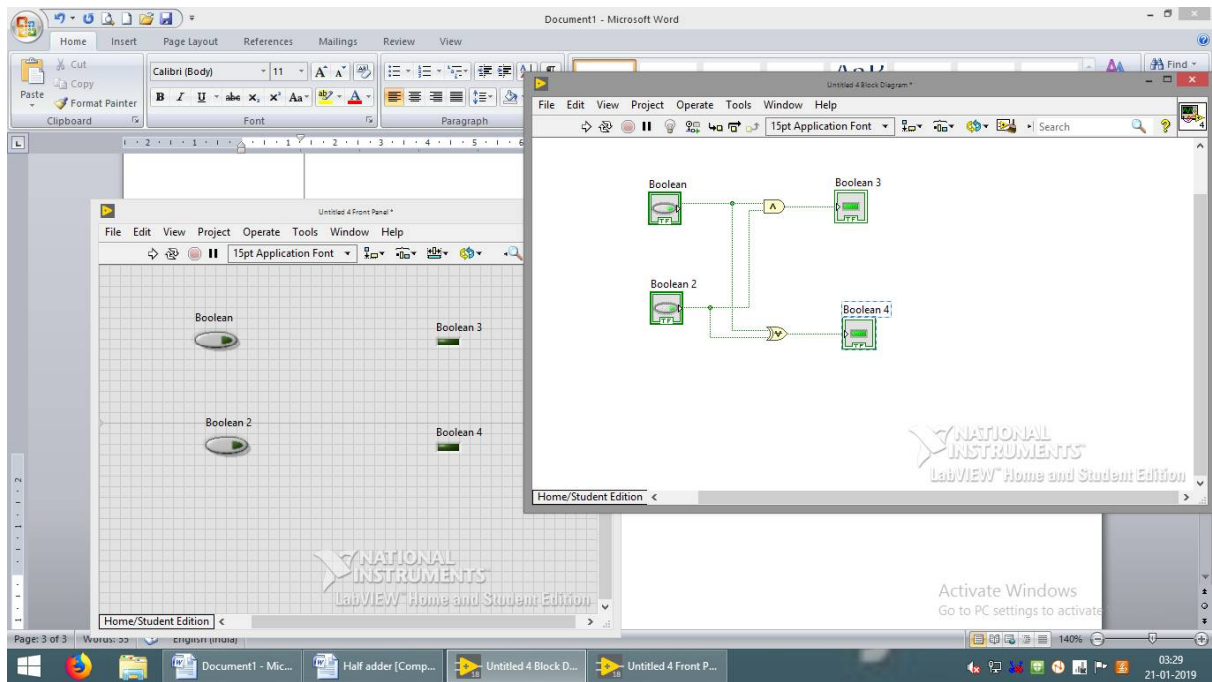
STEP 3:-
Right click on the front panel and select Boolean and select square LED.



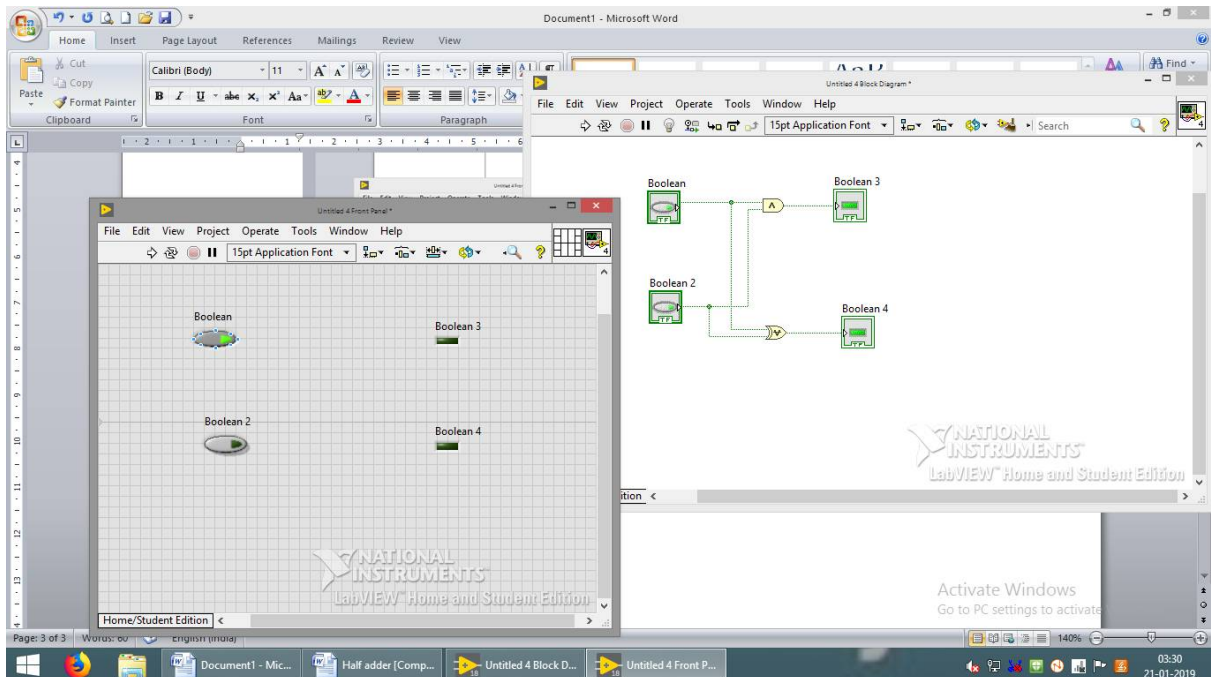
STEP 4:-
Right click on the block diagram and select Boolean and select AND gate and EX-OR gate.



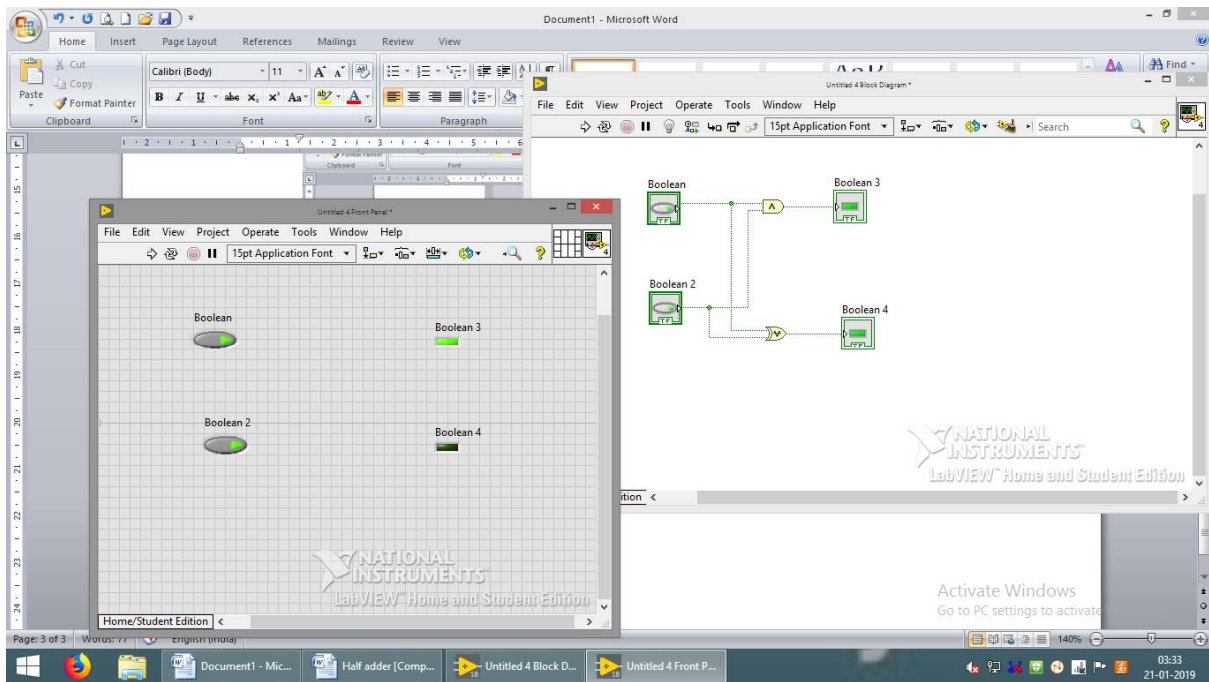
STEP 5:-
Connect using wire.



STEP 6:-
Click the Boolean button (Green colour appears), and click the arrow button for result.



STEP 7:-
Check the result by selecting the Boolean (on and off) and check the result in the square LED.



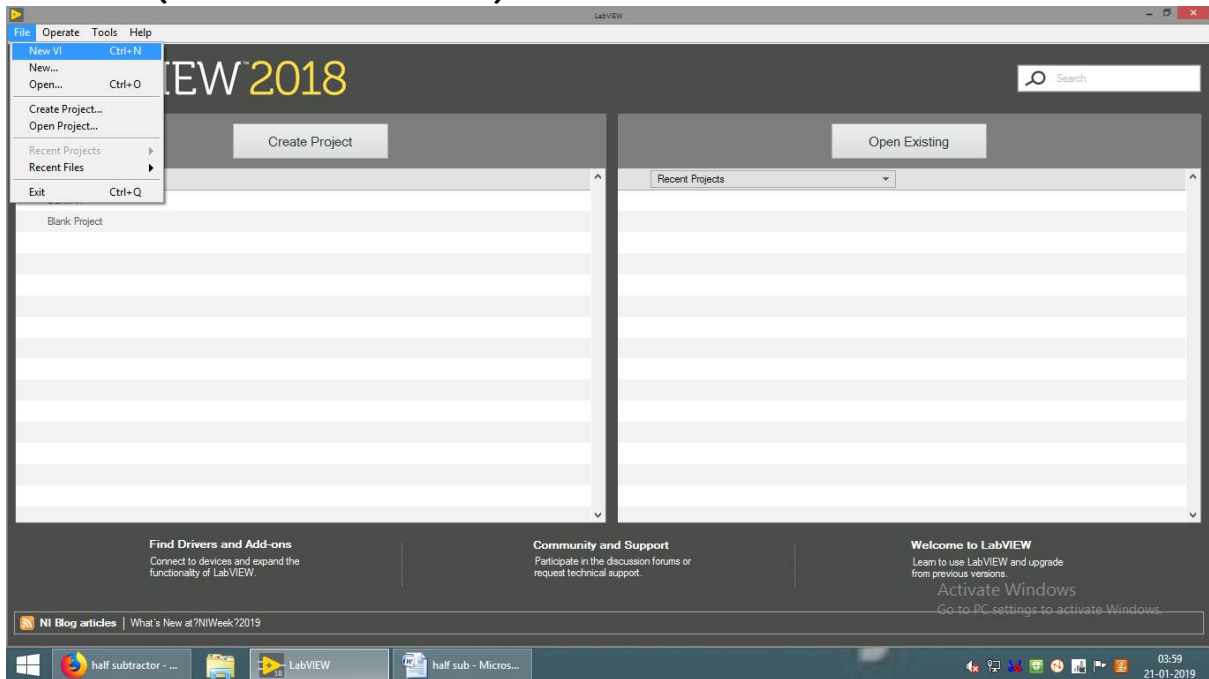
RESULT

The result verified.

7. Appendix B:-

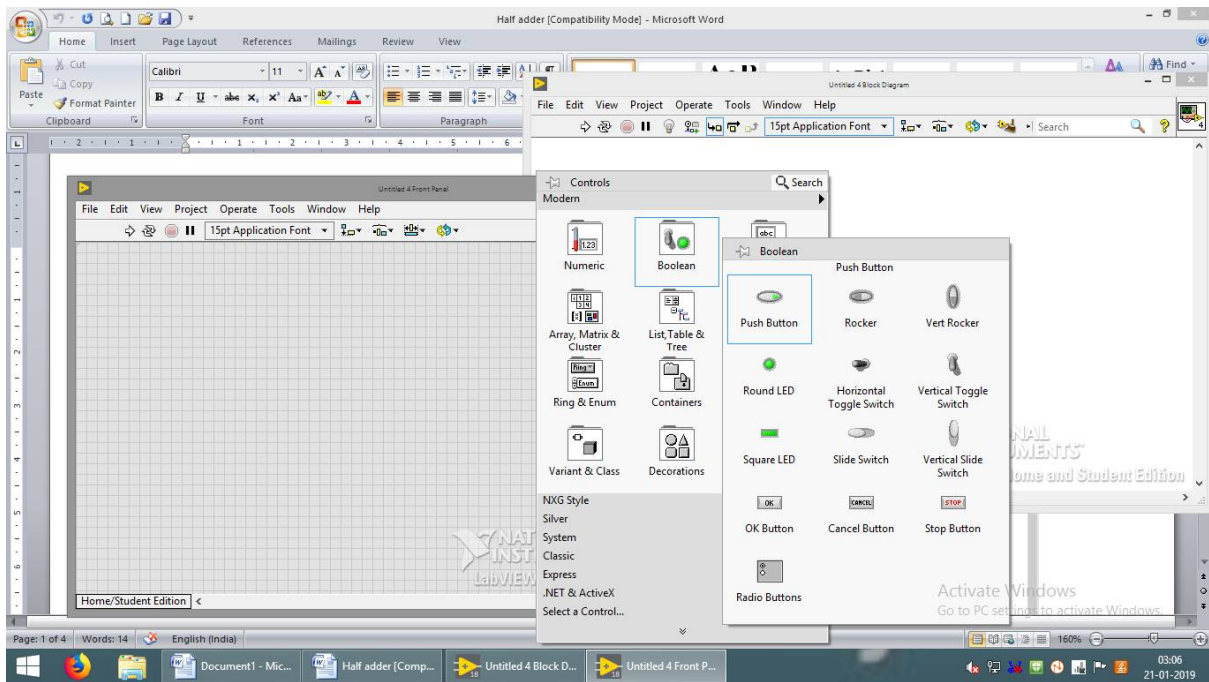
STEP 1:-

File New VI(Virtual Instrumentation)

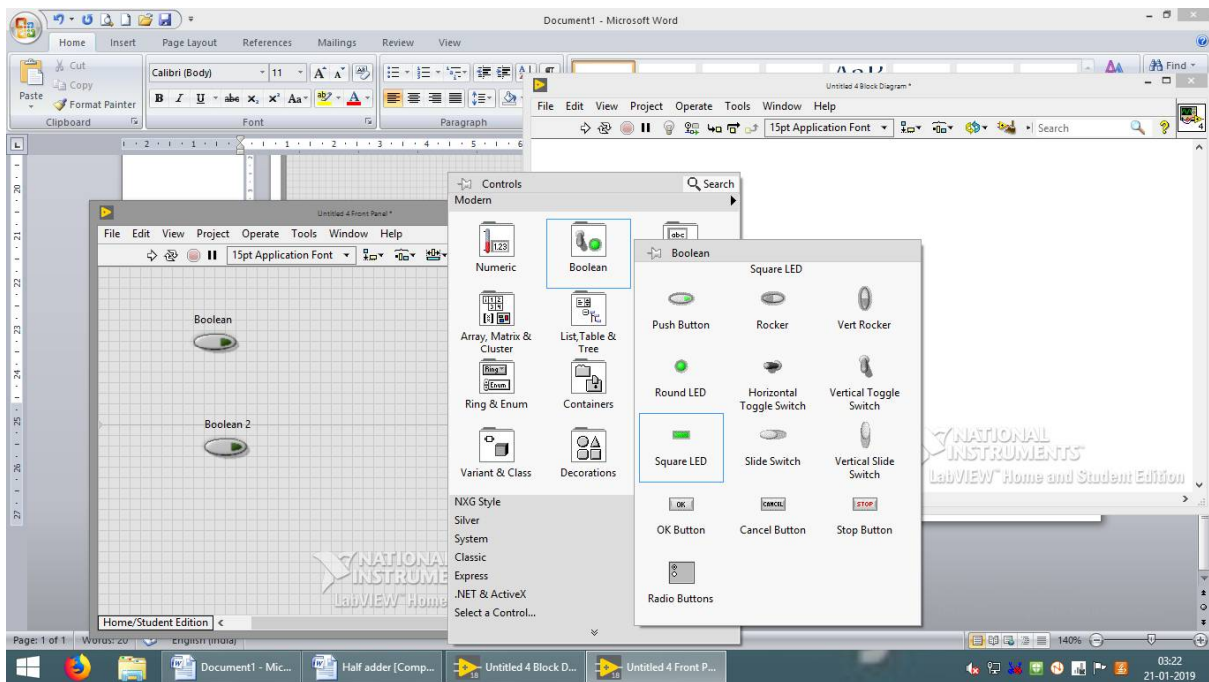


STEP 2:-

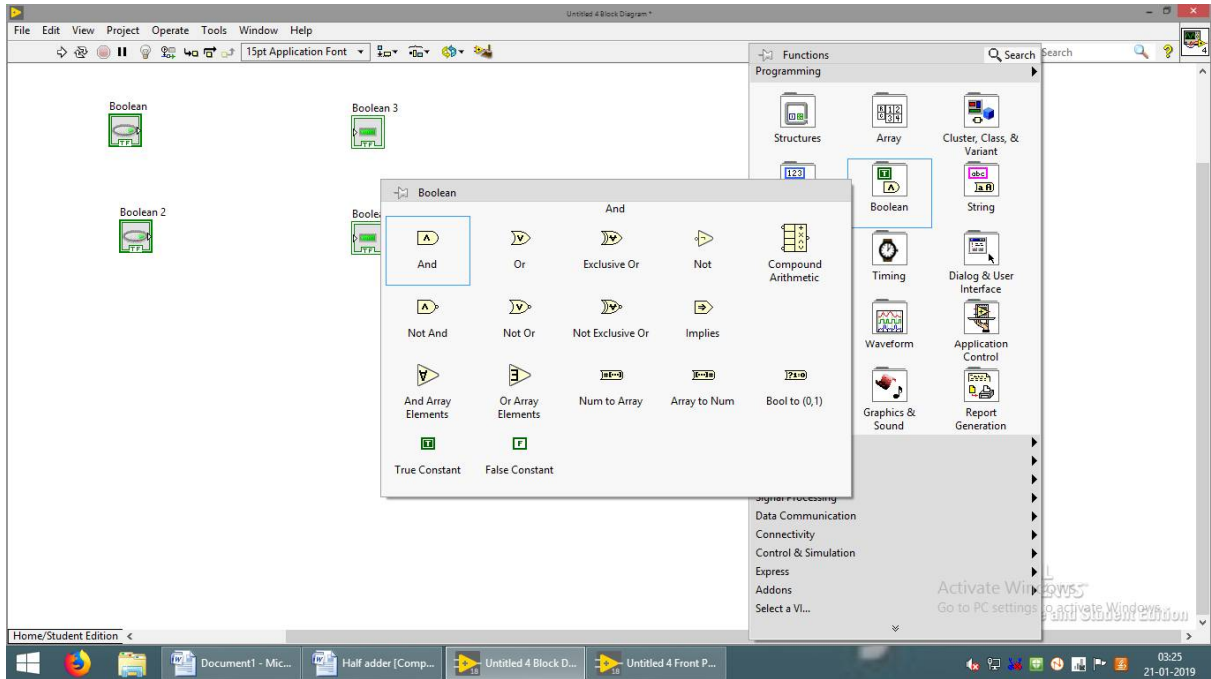
Right click on the front panel and select Boolean and select push button.



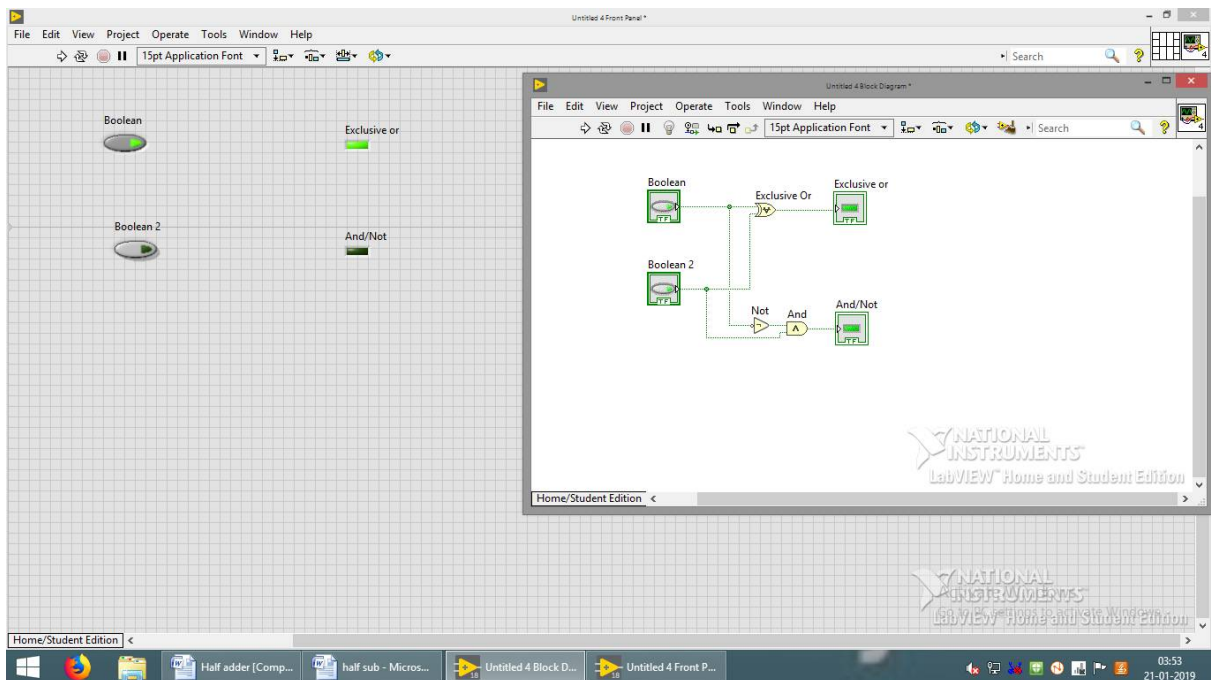
STEP 3:-
Right click on the front panel and select Boolean and select square LED.



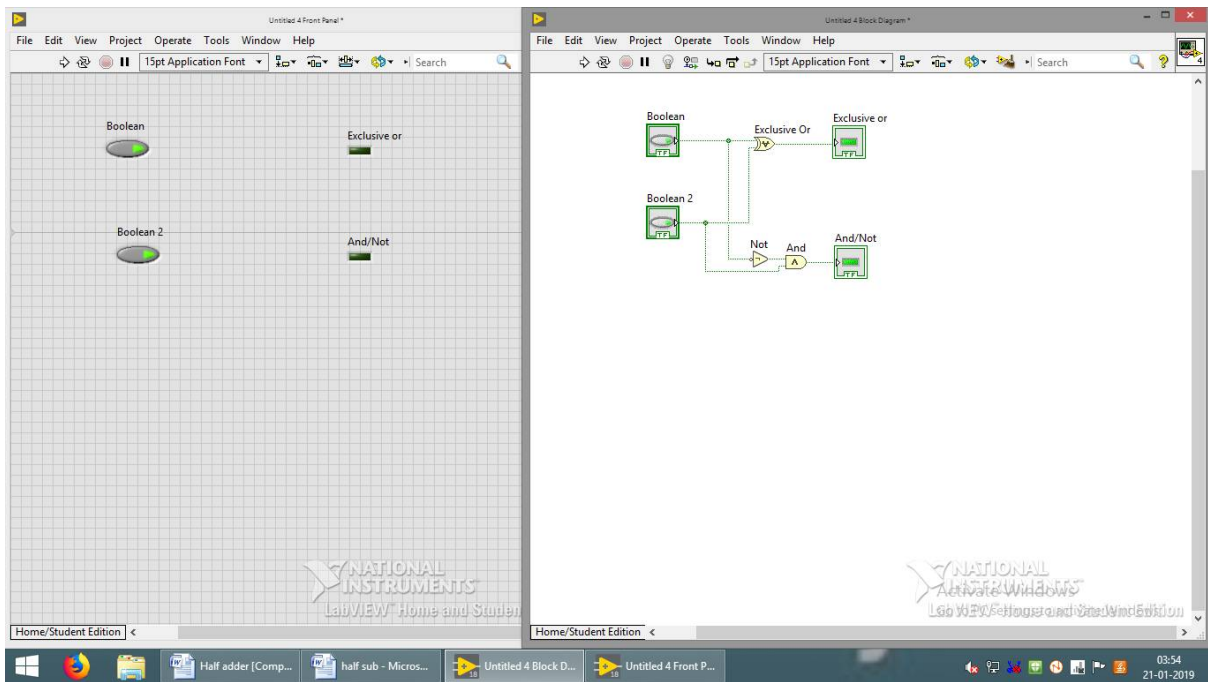
STEP 4:-
Right click on the block diagram and select Boolean and select AND gate, EX-OR gate and NOT gate.



**STEP 5:-
Connect using wire.**

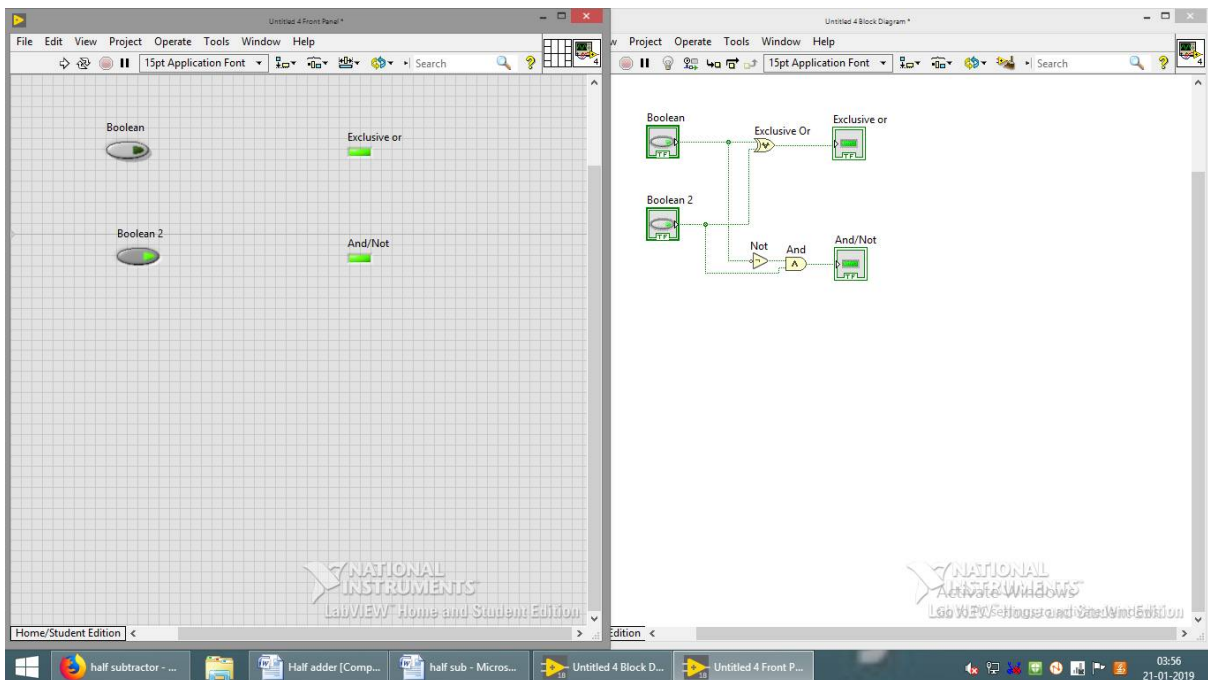


**STEP 6:-
Click the Boolean button (Green colour appears), and click the arrow button for result.**



STEP 7:-

Check the result by selecting the Boolean (on and off) and check the result in the square LED.



RESULT

The result verified.

8.Appendix C:-

The Name LabVIEW is a shortened form of its description: Laboratory Virtual Instrument Engineering Workbench.

LabVIEW is a visual programming language: it is a system-design platform and development environment that was aimed at enabling all forms of system to be developed.

LabVIEW was developed by National Instruments as a workbench for controlling test instrumentation. However its applications have spread well beyond just test instrumentation to the whole field of system design and operation.

History of LabVIEW

LabVIEW was first launched 1986 as a tool for scientists and engineers to facilitate automated measurements - the aim was that it would be a tool that would be as productive for scientists and engineers as spreadsheets were for financial analysts.

Says Jeff Kodowsky of National Instruments who came up with the initial idea and developed it: "We weren't seeking to create a language but that's what we ended up doing because we needed that level of flexibility and control in order to deal with the kinds of IO and processing required."

In addition to this, Kodowsky had been using an early Apple Mac which utilised graphics more than any other computing system. Kodowsky wanted to be able to utilise this capability to enable quicker programming of the control for instruments.

What is LabVIEW?

LabVIEW uses a graphic interface that enables different elements to be joined together to provide the required flow.

LabVIEW is essentially an environment that enables programming in G – this is a graphical programming language created by National Instruments that was initially developed to communicate via GPIB, but since then it has been considerably updated. Nowadays, G can be used for automated test applications, general data acquisition, programming FPGAs, etc

LabVIEW is essentially the user interface for G. However as the software has developed, the term LabVIEW is now synonymous with the G language. LabVIEW also provides a host of other facilities including debugging, automated multithreading, application user interface, hardware management and interface for system design.

In this way, LabVIEW acts as a portal for a variety of facilities, bring them together under a single element that is easy to manage.

LabVIEW runs on Windows, OS X (Apple) and Linux platforms, making it suitable for most computing systems.

LabVIEW key concepts

Within LabVIEW there are several elements and concepts that are key to the format and operation of the environment. These include:

- **LabVIEW environment:** The LabVIEW environment consists of LabVIEW VI manager (project explorer), the programming tools, debugging features, templates and ready built sample examples, and an easy interface to the hardware drivers. Read more about LabVIEW environment.
- **LabVIEW VIs:** The LabVIEW VI is a “Virtual Instrument” that enables a user interface to be built and it contains the programming code. Read more about LabVIEW Virtual Instruments, VIs.
- **LabVIEW G programming:** This is the graphical programming language where the functional algorithms are built using “drag and drop” techniques. Read more about LabVIEW programming.
- **LabVIEW dataflow :** This is the core concept that determines the running order for the programme.

LabVIEW applications

LabVIEW provides a powerful platform for undertaking a wide variety of different applications. It started as an environment for managing test programming, but since its inception, the applications for which it can be used have considerably expanded. It has expanded from being a graphical test management language to become a graphical system design environment.

This means that it can be used for an enormous variety of interesting and diverse applications. Not only can it be used for equipment control (including the control of the large Hadron Collider at CERN) and a variety of data acquisition applications (including car development simulation where Big Data monitoring is undertaken) to the system design arena where it has been used for development of projects from RF circuitry to biomedical equipment, green technology and much more.

LabVIEW advantages / disadvantages

Like any product or platform, LabVIEW has its advantages and disadvantages. These must be carefully considered before starting its use.

LabVIEW advantages

1. Graphical interface is flexible and simple to use. Most engineers and scientists can learn to use it quickly.
2. LabVIEW provides a universal platform for numerous applications in diverse fields.
3. LabVIEW can be used with 3rd party hardware: it can be interfaced with C/C++, VB, Fortran etc etc.
4. Easy to interface to many hardware items like data acquisition and test equipment products.
5. It has excellent customer support and a large active community forum.

LabVIEW disadvantages

1. LabVIEW is single sourced and some companies may not like to use a product that is single sourced and not standardised by the industry.
2. Cost of ownership – although in line with many other industry products of a similar nature, its cost should be considered before it is introduced.
3. For those more accustomed to text programming, graphical programming can take a little familiarisation time.

Like any product, LabVIEW has some disadvantages, but many who use it find it particularly effective. In view of this, LabVIEW should be carefully assessed before its use is introduced.

LabVIEW environment tools

Within the LabVIEW environment there is a good selection of tools to assist the development of the overall programmes.

These tools include:

- **Templates & sample projects:** This is a tool that features some of the common templates that can be used for starting projects along with some fully formed projects that can be used to see how projects may work, or to be customised for particular applications.
- **Example finder:** This has many hundreds of code snippets that illustrate the methodology behind a huge variety of different tasks. These can be used or customised as appropriate.
- **Debugging tools:** As with any project, debugging is an essential element of any development. The LabVIEW environment provides some comprehensive tools to assist with the speedy debug of programmes. The tools include elements like probes, breakpoints, single step, highlighting execution (the ability to slow execution to see how the system operates and thereby isolate issues in a dynamic manner).
- **Hardware manager (Measurement & Automation Explorer):** This is a complementary tool to LabVIEW. It enables drivers to be managed – updated and monitored. It also allows the monitoring of validity of the connection to hardware before running the programme.
- **Source control:** The LabVIEW environment has several tools to provide configuration management. This enables larger teams to work together on an applications whilst still retaining proper configuration management to be undertaken.
- **LabVIEW tools network:** This element of the LabVIEW environment enables the downloading of specialised toolkits for specific applications. Some are provided by National Instruments, whereas others may be provided by NI partners.

LabVIEW VIs

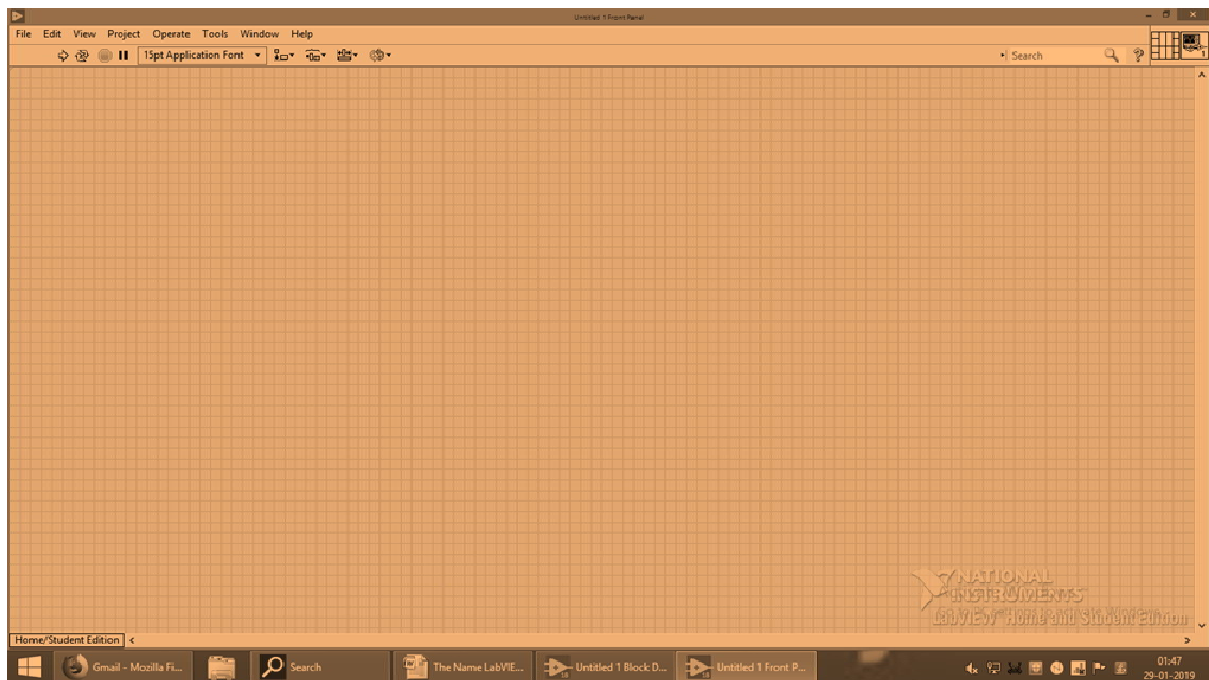
The Virtual Instrument or VI is an integral element of the LabVIEW environment. VIs are individual code modules that make up a complete application.

An application could be as simple as a single VI, but normally many more are included and some applications may utilise hundred of VIs or possibly more dependent upon the particular application.

LabVIEW VI basics

The LabVIEW VI consists of two main elements:

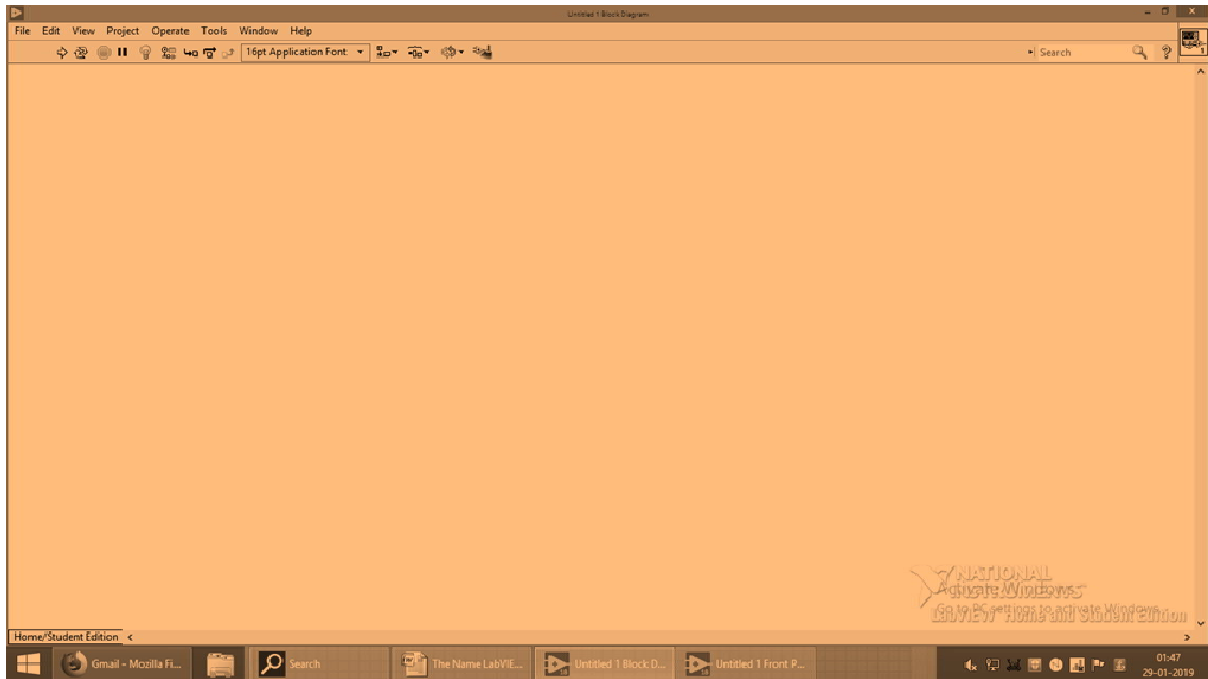
Front Panel: The LabVIEW front panel is what the user of the completed application will see. It enables them to interact with the VI, inputting controls and also seeing results. It can be likened to the front panel of a test instrument or other piece of equipment.



The LabVIEW VI front panel can be built up from scratch using the palette of different controls, indicators and data types.

LabVIEW VI front panels can be completely customized. By having a totally customized front panel, it is possible to simplify the operation to provide exactly what is needed without unnecessary controls that may not be required.

Block Diagram: The LabVIEW VI block diagram is where the functionality of the VI is programmed in G. The block diagram defines the functionality whilst also providing a visual representation of it. In this way the block diagram is similar to a flow diagram within a programme.



There is an associated functions palette within the block diagram space where all the elements needed to build the programme can be found. This enables swift accurate programming to be achieved.

9.FUNDAMENTALS OF LAB VIEW QUESTIONS AND ANSWERS

1. Differentiate between Data flow and control flow programming

Control Flow	Data Flow
In Control Flow tasks require completion (Success, fail or completion) before moving to the next task.	In Data Flow one component will not wait for other component to finish, all of them work together for processing data in the streaming way.
There is no proper structure for control flow task as we have for data flow task (Source->transformation->Destination).	It consists of Source, transformation and destination task items.
Precedence constraints are used to connect to the tasks together and manage the execution order.	There is no order for running the different data flow tasks. They can run simultaneously.
The data is not transferred between the tasks, if we have to use any data in many tasks; we have to define data in the variables.	Data is being transferred from one task to another. It defines its movement and transformation.
We can add the constraint in the link between the control flow tasks.	We can't add any constraint between the links like Control flow, we only can add the data viewer to watch out the intermediate Data.

2. Differentiate between Graphical and text based programming

Text-based Programming

- Syntax must to be known.
- Top to bottom program execution.
- Error checking → Program has to be compiled or executed.
- Front panel design → extra coding.
- Non – interactive.
- Conventional method.
- Logical error finding is easy.
- Program flow is not visible.
- Language (text) based programming.
- Passing parameter to sub routine is difficult.

Graphical Programming

- Syntax is knowledge; not required for programming.
- Left to right program execution.
- Errors are indicated as blocks are wired.
- Front panel design → part of programming.
- Highly interactive.
- Data flow programming.
- Logical error finding is complicated.
- Data flow is visible.
- Icon based programming and wiring.
- Passing parameter to sub VI is easy.

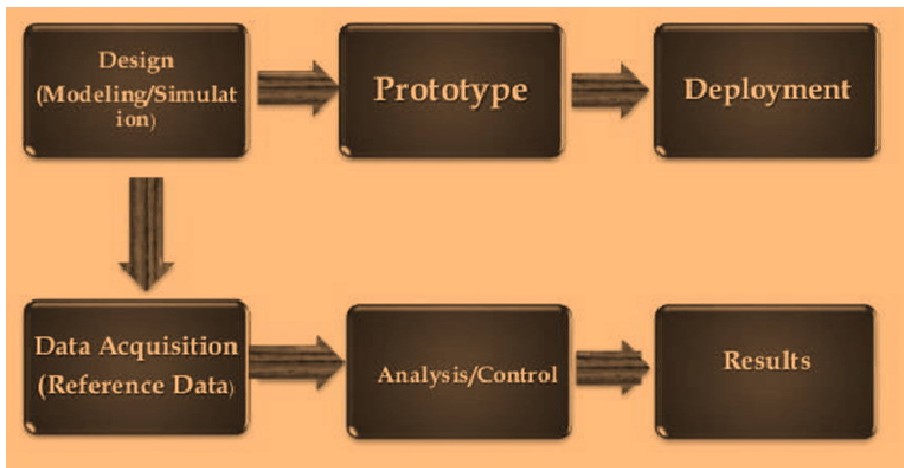
3. VI in system control

- PC – software flexibility and capability
- PLC – ruggedness and reliability
- More complex, dynamic, adaptive and algorithm based control
- Programmable Automation Controllers (PAC) – multi-domain functionality, multiple I/O types.

4. Advantages of LabVIEW

- Graphical user interface, Drag-and-drop built-in functions
- Modular design and hierarchical design
- Multiple high level development tools
- Professional development tools and Multi platforms
- Reduces cost and preserves investment, Flexibility and scalability
- Connectivity and instrument control, Open environment
- Distributed development, Visualization capabilities
- Rapid development with express technology
- Compiled language for fast execution, simple application distribution
- Target management, Object-oriented design, Algorithm design

5. Graphical System Design



- **Design**

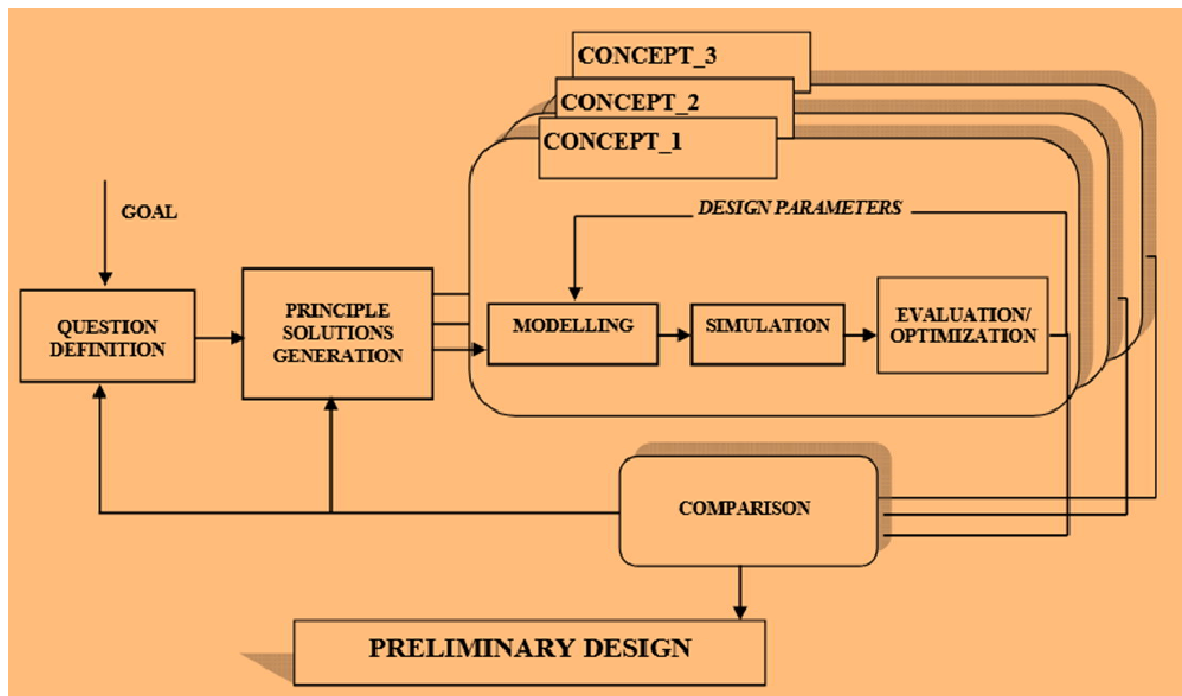
Mathematical model simulation, validating model performance and optimization. Hardware in Loop tests, high performance computing, use of GPU, multi-core processors and parallel programming

- **Prototype**

Test a prototype in lab, signal processing and analysis. Modify and optimize the original model based on experimental results. Use of RTOS, FPGA, DSP, GPU processors.

- **Deployment**

PC or dedicated embedded controller – standalone and real time (deterministic) mode. Use of LabVIEW real time module, symmetric multiprocessing techniques, RF vector signal analyzers, dynamic signal acquisition devices, etc. GUI or operator interfaces (OI) are needed. Data logging and supervisory control module Applications of Virtual Instrumentation in Modeling and Simulation



6. Historical perspective and development of VI

- Analog measurement devices.
- Data acquisition and processing devices.
- Digital processing based on general purpose computing platform.
Distributed virtual instrument

7. Applications of VI in Biomedical systems

- Basic and Clinical research and instrumentation
- Diagnostic and treatment equipment
- Imaging techniques
- Bio informatics
- Outcome analysis and patient monitoring
- Telemedicine
- Information retrieval and hospital information management
- Commercial medical devices testing and development
- Space medicine

8. LabVIEW Front Panel and Block Diagram

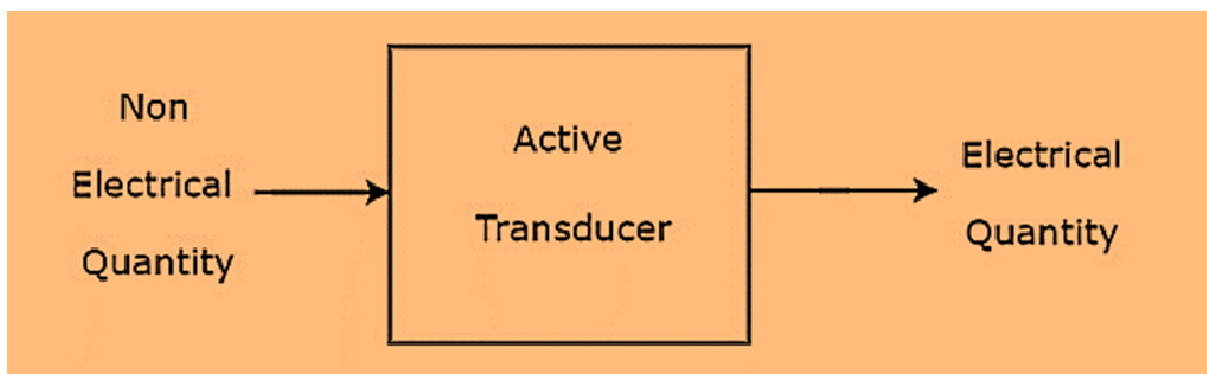
- Front Panel → to create GUI
- Block Diagram → to create code/logic that makes the program work
- Front Panel → Controls Palette. All necessary elements needed to create GUI.
- Block Diagram → Functions Palette. All functions needed to create the logic in the program.
- Explanation of Tool bar elements.

9. Classification of Transducer:

A sharp distinction among the types of transducers is difficult. The transducers may be classified according to their application, method of energy conversion, nature of the output signal and so on. All these classifications generally result in overlapping areas. In one way, the electrical transducers are classified as;

- (1) Active Transducers
- (2) Passive Transducers

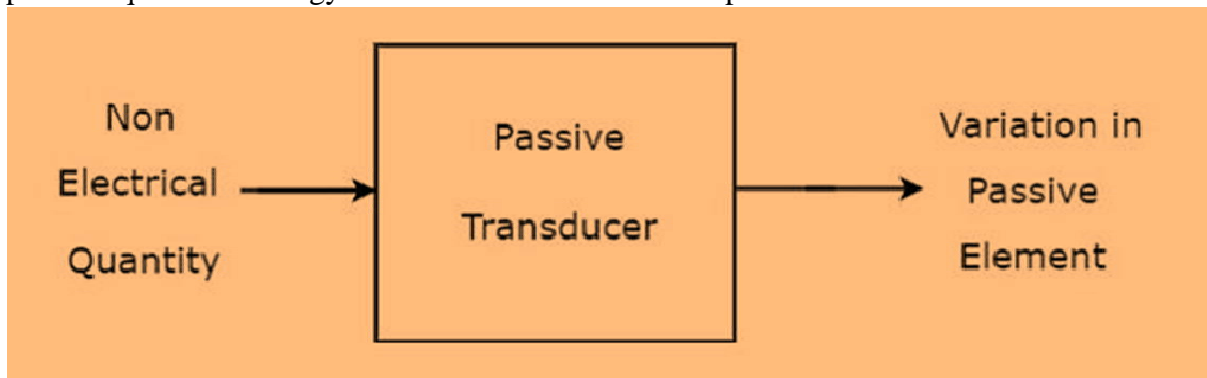
Active Transducers: It is also known as self-generating type transducers. They develop their own voltage or current as the output signal. The energy required for production for this output signal is obtained from the physical phenomenon being measured.



Examples of active transducers:

Thermocouple, Piezoelectric transducers, Photovoltaic cell, Moving coil generator, Photoelectric cell.

Passive Transducers: It is also called as externally powered transducers. They derive the power required for energy conversion from an external power source.

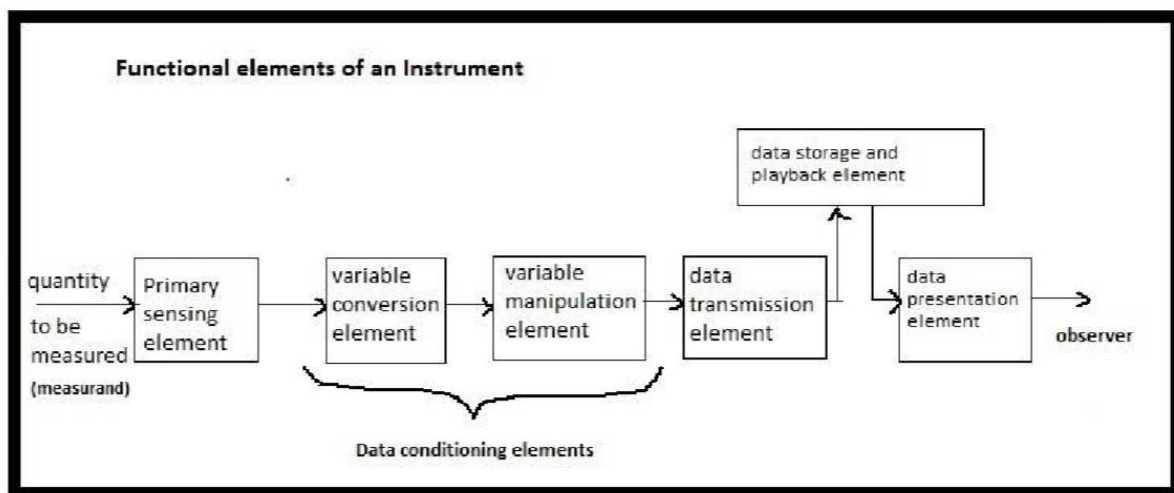


The passive transducers are further classified into Resistive type, Inductive type and capacitive type.

Phenomenon	Transducer
Temperature	Thermocouple, RTD, Thermistor
Light	Photo Sensor
Sound	Microphone
Force and Pressure	Strain Gage Piezoelectric Transducer
Position and Displacement	Potentiometer, LVDT, Optical Encoder
Acceleration	Accelerometer
pH	pH Electrode

GENERAL CONFIGURATION AND FUNCTIONAL DESCRIPTION OF MEASURING INSTRUMENTS

A systematic organization and analysis are more important for measurement systems. The whole operation system can be described in terms of functional elements. The functional elements of generalized measurement system



Most of the measurement system consists of following functional elements.

1. Primary sensing element
2. Variable conversion element
3. Variable manipulation element
4. Data transmission element

5. Data storage and playback element

6. Data presentation element

1. Primary Sensing Element : The quantity under measurement makes its first contact with primary sensing element of measurement system. The quantity is first sensed or detected by primary sensor. Then detected physical quantity signal is converted into an electrical signal by a transducer. Transducer is defined as a device which converts a physical quantity into an electrical quantity. Sensor is act as primary element of transducer. In many cases the physical quantity is directly converted into an electrical quantity by a transducer. So the first stage of a measurement system is known as a detector transducer stage. Example, Pressure transducer with pressure sensor, Temperature sensor ect.,

2. Variable Conversion Element : The output of primary sensing element is electrical signal of any form like a voltage, a frequency or some other electrical parameter. Sometime this output not suitable for next level of system. So it is necessary to convert the output some other suitable form while maintaining the original signal to perform the desired function the system. For example the output primary sensing element is in analog form of signal and next stage of system accepts only in digital form of signal. So, we have to convert analog signal into digital form using an A/D converter. Here A/D converter is act as variable conversion element.

3. Variable Manipulation Element : The function of variable manipulation element is to manipulate the signal offered but original nature of signal is maintained in same state. Here manipulation means only change in the numerical value of signal. Examples, 1. Voltage amplifier is act as variable manipulation element. Voltage amplifier accepts a small voltage signal as input and produces the voltage with greater magnitude .Here numerical value of voltage magnitude is increased. 2. Attenuator acts as variable manipulation element. It accepts a high voltage signal and produces the voltage or power with lower magnitude. Here numerical value of voltage magnitude is decreased.

Linear process manipulation elements: Amplification, attenuation, integration, differentiation, addition and subtraction ect.,

Nonlinear process manipulation elements: Modulation, detection, sampling, filtering, chopping and clipping ect.,

All these elements are performed on the signal to bring it to desired level to be accepted by the next stage of measurement system. This process of conversion is called signal conditioning. The combination of variable conversion and variable manipulation elements are called as Signal Conditioning Element.

4. Data Transmission Element : The elements of measurement system are actually physically separated; it becomes necessary to transmit the data from one to another. The element which is performs this function is called as data transmission element. Example, Control signals are transmitted from earth station to Space-crafts by a telemetry system using radio signals. Here telemetry system is act as data transmission element. The combination of Signal conditioning and transmission element is known as Intermediate Stage of measurement system.

5. Data storage and playback Element : Some applications requires a separate data storage and playback function for easily rebuild the stored data based on the command. The data

storage is made in the form of pen/ink and digital recording. Examples, magnetic tape recorder/ reproducer, X-Y recorder, X-t recorder, Optical Disc recording ect.,

6. Data presentation Element : The function of this element in the measurement system is to communicate the information about the measured physical quantity to human observer or to present it in an understandable form for monitoring, control and analysis purposes. Visual display devices are required for monitoring of measured data. These devices may be analog or digital instruments like ammeter, voltmeter, camera, CRT, printers, analog and digital computers. Computers are used for control and analysis of measured data of measurement system. This Final stage of measurement system is known as Terminating stage.

Resistance temperature detectors (RTDs)

RTDs work on the principle that the electric resistance of a metal changes due to change in its temperature. On heating up metals, their resistance increases and follows a linear relationship as shown in Figure 2.5.2. The Correlation is

$$R_t = R_0(1 + \alpha T) \dots\dots\dots (1)$$

Where R_t is the resistance at temperature $T(0^\circ\text{C.})$ and R_0 is the resistance at 0°C. and α is the constant for the metal termed as temperature coefficient of resistance.

The sensor is usually made to have a resistance of 100Ω at 0°C.

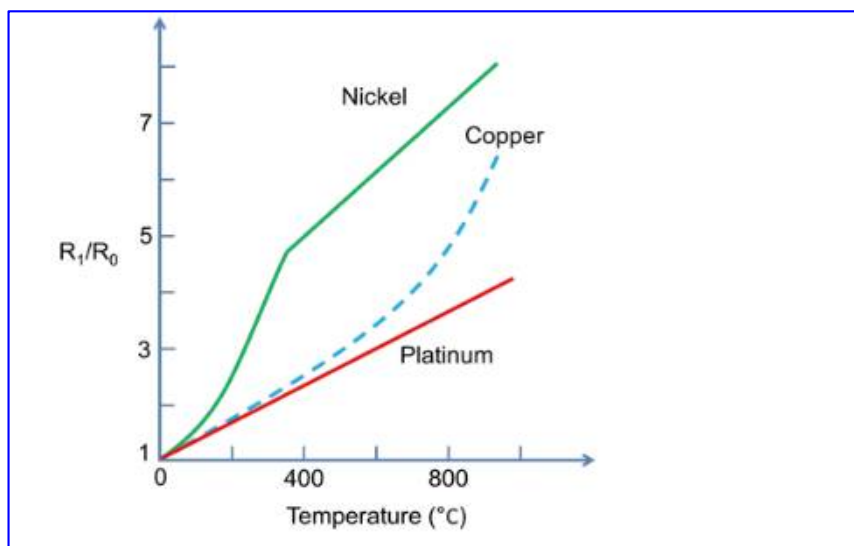


Figure 1 shows the construction of a RTD.

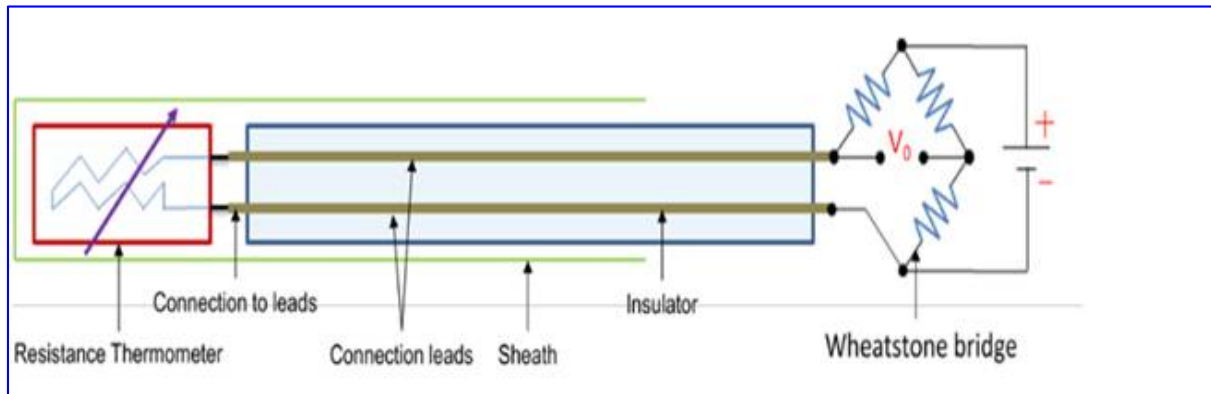


Fig 2 Construction of a Resistance Temperature detector (RTD)

It has a resistor element connected to a Wheatstone bridge. The element and the connection leads are insulated and protected by a sheath. A small amount of current is continuously passing through the coil. As the temperature changes the resistance of the coil changes which is detected at the Wheatstone bridge.

RTDs are used in the form of thin films, wire wound or coil. They are generally made of metals such as platinum, nickel or nickel-copper alloys. Platinum wire held by a high-temperature glass adhesive in a ceramic tube is used to measure the temperature in a metal furnace. Other applications are:

- Air conditioning and refrigeration servicing
- Food Processing
- Stoves and grills
- Textile production
- Plastics processing
- Petrochemical processing
- Micro electronics
- Air, gas and liquid temperature measurement in pipes and tanks
- Exhaust gas temperature measurement

STRAIN GAUGE

Strain gauge based sensors work on the principle of change in electrical resistance. When, a mechanical element subjects to a tension or a compression the electric resistance of the material changes. This is used to measure the force acted upon the element. The details regarding the construction of strain gauge transducer

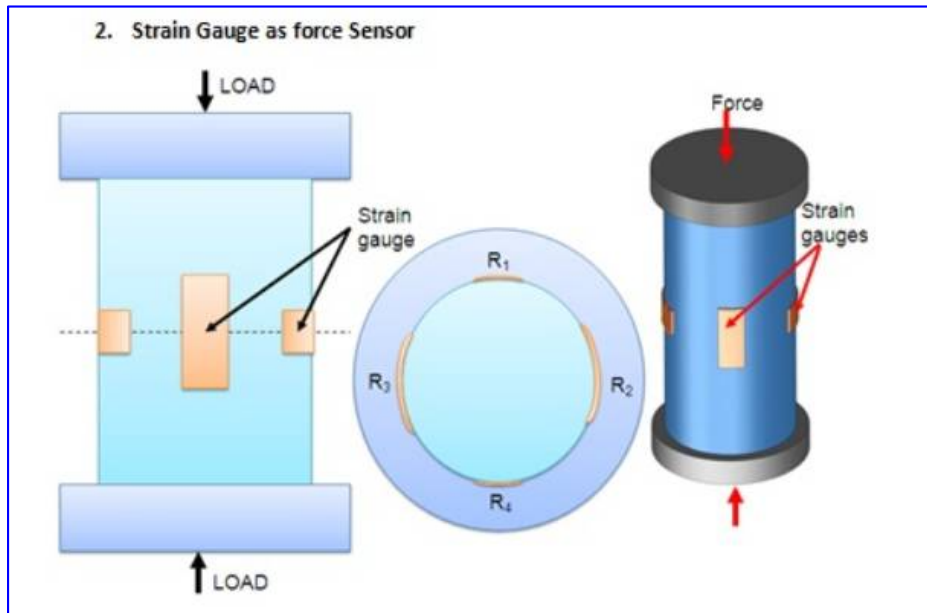


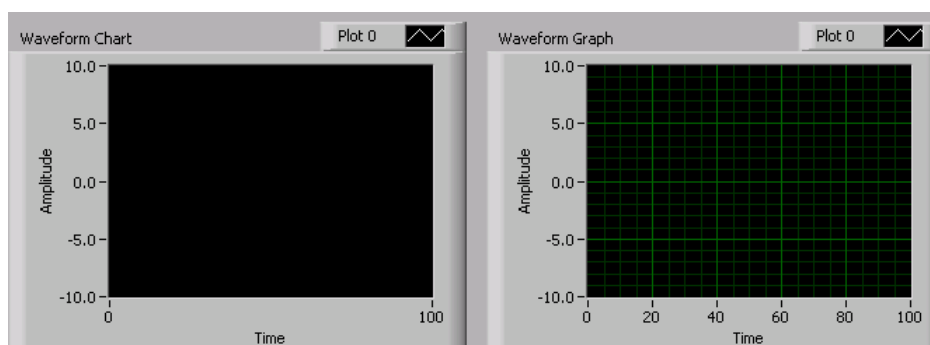
Figure 3. strain gauge load cell.

It comprises of cylindrical tube to which strain gauges are attached. A load applied on the top collar of the cylinder compress the strain gauge element which changes its electrical resistance. Generally strain gauges are used to measure forces up to 10 MN. The non-linearity and repeatability errors of this transducer are $\pm 0.03\%$ and $\pm 0.02\%$ respectively.

List of various modes of chart and waveform–LABVIEW

Graphs and Charts

Use graphs and charts to display plots of data in a graphical form.



Graphs and charts differ in the way they display and update data. VIs with graphs usually collect the data in an array and then plot the data to the graph, which is similar to a spreadsheet that first stores the data then generates a plot of it. In contrast, a chart appends new data points to those already in the display. On a chart, you can see the current reading or measurement in context with data previously acquired.

The graphs and charts located on the Controls»Graph palette include the following types:

- Waveform Chart and Graph—Displays data acquired at a constant rate.
- XY Graph—Displays data acquired at a non-constant rate, such as data acquired when a trigger occurs.
- Intensity Chart and Graph—Displays 3D data on a 2D plot by using color to display the values of the third dimension.
- Digital Waveform Graph—Displays data as pulses or groups of digital lines. Computers transfer digital data to other computers in pulses.
- (Windows) 3D Graphs—Displays 3D data on a 3D plot in an ActiveX object on the front panel.

You customize the appearance of graphs and charts by showing or hiding options. Right-click the graph or chart and select Visible Items from the shortcut menu to display or hide the following options:

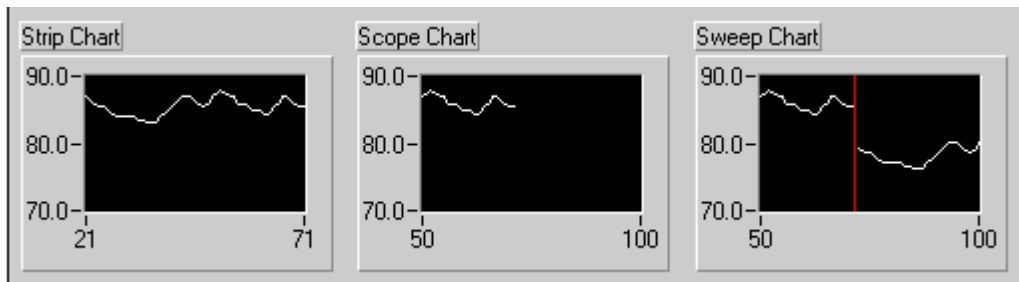
- Plot Legend—Defines the color and style of the plot(s). Resize the legend to display multiple plots.
- Scale Legend—Defines labels for scales and configures scale properties.
- Graph Palette—Changes scaling and formatting while a VI is running.
- X Scale and Y Scale—Formats the x- and y-scales.
- Cursor Legend (graph only)—Displays a marker at a defined point coordinate. You can display multiple cursors on a graph.
- Scrollbar (chart only)—Scrolls through the data in the chart. Use the scroll bar to view data that the buffer does not currently display.

National Instruments charts, in both LabVIEW and Measurement Studio, have three different modes to scroll data. These three update modes are Strip Chart, Scope Chart, and Sweep Chart. These update modes can be configured using chart properties. To do this, right-click on the chart indicator and select **Properties » Appearance**; select the **Update Mode** pull down menu and select **Strip Chart, Scope Chart, or Sweep Chart**.

Strip Chart: This mode has a scrolling display that is similar to a paper tape strip chart recorder. This mode first plots values from left to right. From here, it continues to plot new points at the rightmost point and shifts old values to the left.

Scope Chart: This mode has a retracing display similar to an oscilloscope. As it receives each new value, it plots the value to the right of the last value. When the plot reaches the right border of the plotting area, it erases the plot and begins plotting again from the left border.

Sweep Chart: This mode acts much like the scope chart, but the plot is not erased when the plot hits the right border. Instead, a moving vertical line marks the beginning of new data and moves across the display from left to right as it adds new data.



CLUSTERS

Clusters group data elements of mixed types.

- Elements of clusters must be all controls or all indicators or constants.
- The size of components in a cluster is fixed.
- Cluster elements are accessed through the cluster order.
- If a front panel contains more than 28 controls and indicators that you want to use programmatically, group some of them into a cluster and assign the cluster to a terminal on the connector pane to eliminate clutter on the block diagram.
 - To create a cluster control or indicator, select a cluster on the Functions»All Functions»Array & Cluster palette, place it on the front panel, and drag controls or indicators into the cluster shell.
 - Use the cluster functions located on the Functions»All Functions»Cluster palette to create and manipulate clusters.
 - Error checking tells you why and where errors occur.
 - The error cluster reports the status, code and source of the error.
 - Use the error cluster controls and indicators to create error inputs and outputs in subVIs.
 - Arrays and clusters are inter-convertible but only under certain conditions.

STRUCTURES

Structures are graphical representations of the loops and case statements of text-based programming languages. There are cases when a decision must be made in a program. In text-based programs, this can be accomplished with statements like if-else, case and so on. LabVIEW includes many different ways of making decisions. The simplest of these methods is the select function located in the functions palette. This function selects between two values dependent on a Boolean input. Use structures on the block diagram to repeat blocks of code and to execute code conditionally or in a specific order. Like other nodes, structures have terminals that connect them to other block diagram nodes, execute automatically when input data are available, and supply data to output wires when execution is complete. Each structure has a distinctive, resizable border to enclose the section of the block diagram that

executes according to the rules of the structure. The section of the block diagram inside the structure border is called a subdiagram. The terminals that feed data into and out of structures are called tunnels. A tunnel is a connection point on a structure border. Use the following structures located on the Structures palette to control how a block diagram executes processes:

- For Loop—Executes a sub diagram a set number of times.
- While Loop—Executes a sub diagram until a condition occurs.
- Case structure—Contains multiple sub diagrams, only one of which executes depending on the input value passed to the structure.
- Sequence structure—Contains one or more sub diagrams that execute in sequential order.
- Event structure—Contains one or more sub diagrams that execute depending on how the user interacts with the VI.
- Timed Structures—Execute one or more sub diagrams with time bounds and delays.
- Diagram Disable Structure—Has one or more sub diagrams, or cases, of which only the enabled subdiagram executes.
- Conditional Disable Structure—Has one or more sub diagrams, or cases, exactly one of which LabVIEW uses for the duration of execution, depending on the configuration.