UNIT-III

Instrumentation

The process in which the set of various interconnected electrical, measurement, and control instruments measure, analyse, and control electrical and non-electric physical quantities is called Instrumentation.

Introduction of virtual instrumentation (VI)

- The virtual instrumentation is a system that uses computer equipped with powerful application software and hardware.
- The system is capable of functioning as traditional instrumentation system and it can perform the task of testing and automation
- The traditional instrumentation system is oriented to hardware whereas virtual instrumentation is software oriented. Virtual instrumentation can be organized as per the requirement i.e., it is used defined.
- All the signal handling and control action is done through software.

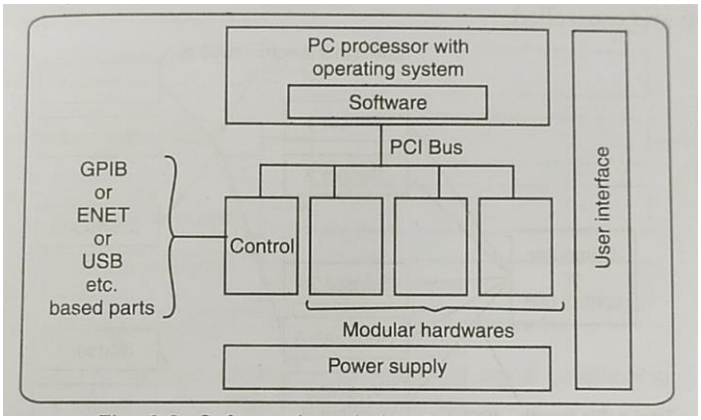

Fig.1 Software based virtual instrumentation

- Virtual instrumentation is flexible and suitable for innovations. It has two components software and hardware.
- Virtual instrumentation has power supply, PC processor with operating system, software PC interface bus, control, modular hardware, user interface and communication parts based on GPIB, ENLT or USB, etc.
- Virtual instruments are compatible with conventional instruments and provides libraries for interfacing with common ordinary instrumentation buses such is GPIB, Ethernet.
- Software runs on PC processor. The software achieves the functions of virtual instrumentation which allows plug in hardware and access through a network.
- All these lower cost, lower development and maintenance, cost, accuracy and precision.

Difference between Traditional and Virtual Instrumentation

S.No.	Traditional Instrumentation	Virtual Instrumentation
1.	It is hardware concentric.	It is software concentric
2.	Data handling is complicated	Data handling capacity is easier and better
3.	Data analysis is difficult	Due to computers data analysis is easy and quick.
4.	This lack is portability	Powerful software is available with new operating systems (Windows XP, Net etc.) So, this is portable.
5.	It cannot work for customized applications effectively.	It can work for customized application effectively.
6.	It makes the customer to pay for "What they use".	It lets the customer to pay for "What they need".
7.	It requires more power	It requires less power
8.	Not as economical as virtual instrumentation	It is economical
9.	More secured	Security is less since data is to be handled through computer.

Graphical Programming:

Graphical programming languages provide a different method of coding. Instead of the high-level statements in procedural languages, like C and Object-Oriented languages like C++ or Java, graphical languages are coded by selecting objects, connecting them, and adding functionality.

Graphical Programming Technique

(LAB VIEW- Software used for virtual instrumentation)

- Lab VIEW is a short name of Laboratory Virtual Instrument Engineering Workbench. It is a system-design platform and development environment for a visual programming language from National Instruments.
- It is a dataflow programming language.
- Execution is determined by the structure of a graphical block diagram on which the programmer connects different function-nodes by drawing wires.
- Lab VIEW is a graphical programming language that uses icons instead of lines of text to create applications.
- With LabVIEW, you can design custom virtual instruments by creating a graphical user interface on the computer screen through which you
  - Operate the instrumentation program
  - Control selected hardware
  - Analyse acquired data
  - Display results
- LabVIEW programs are called virtual instruments (VIs). Controls are inputs and indicators are outputs.
- Each VI contains three main parts,

Front Panel - How the user interacts with the VI.
Block Diagram-The code that controls the program.
Icon/Connector - Means of connecting a VI to another VIs.

- In LabVIEW, you build a user interface by using a set of tools and objects. The user interface is known as the front panel. You then add code 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.
- Users interact with the Front Panel when the program is running.
- Users can control the program, change inputs, and see data updated in real-time.
- Controls are used for inputs such as adjusting a slide control to set an alarm value, turning a switch on or off or to stop a program.
- Indicators are used as outputs. 'Thermometers, lights, and other indicators display output values from the program. These may include data, program states, and other information.
- Every front panel control or indicator has a corresponding terminal on the block diagram.
- When a VI is run, values from controls flow through the block diagram, where they are used in the functions on the diagram, and the results are passed into other functions or indicators through wires.

- The Functions Palette is a hierarchical list of VIs, functions and constants for programming on the Block Diagram. Elements are categorized to make navigation easier and help focus on specific topics.
- The functions **palette** is accessible either by right-clicking any empty location on the Block Diagram or clicking on the View-->Functions Palette menu item. Alternatively the functions palette can be pinned in place so that it stays open as a floating window by clicking the push pin icon on the top-left corner of the palette window.
- If accessed by right-click, the palette immediately dismisses itself after an element is selected. If pinned or accessed by menu item the palette will remain open after an element is selected.
- Different palette functions are programming, measure I/O, instrument I/O signal processing, etc.

Advantages of virtual instrumentation:

- **Performance-** VI interfaces often utilize a great deal of processing power which allow them to quickly perform complex tasks.

- **Platform-Independent Nature-** A benefit of the LabVIEW environment is the platform-independent nature of the G-code, which is portable between the different LabVIEW systems for different operating systems (Windows, Linux).

- **Flexibility-** VI can be modified and adapted to the particular needs. The functions would be performed by software running on the PC processor. We can extend the set of functions easily, limited only by the power of the software used.

- **Lower Cost-** By employing virtual instrumentation solutions, lower capital costs, system development costs, and system maintenance costs are reduced.

- **Minimising Set-Up and Configuration Time Costs**
  Once the users have specified and purchased measurement hardware, the real task of developing the application begins. However, the user must first install the hardware and software, configure any necessary setting and ensure that all pieces of the system function properly. This set-up and configuration stage also includes any necessary fixturing, such as sensor connection, wiring and any preparation of the unit under test. Therefore, saving time in set-up and configuration can result in a large reduction in the total application cost.

Data Types:

- Data types, as the name suggests, denotes the type of variables or data that can be used in LabView.
- Data types are important to remember as it helps us in determining what kind of data we have used in our program and what else we can use.
- In LabView, the data types cannot be interconnected i.e. if a variable is of integer type, then its value must also be the type of integer otherwise the LabView will give you an error.
- Data types basically indicate the type of objects, inputs, and outputs that can be wired together.
- There are some unique colours assigned to different data types, which is focused on ensuring that they are not interconnected to each other.
- Every data type has data stored according to its specific type i.e. integer can only store integer value and can only display integer type of data.

➢ **Boolean:**

Boolean data types consist of only two values i.e. true and false. It is a logical data type providing the output in the form of 0 or 1 specifying false and true respectively. The **Boolean data type is indicated by green data wires**. LabView stores the Boolean data as 8-bit values.

➢ **Numeric:**

Numeric data types in LabView are represented as floating point numbers, complex numbers, signed-unsigned integers, and fixed-point numbers. All the Integers either signed or Unsigned are indicated by blue data wires. Double and single precision and complex numbers are represented by orange data wires in LabView. The only difference in the numeric data is determined by the type of their values and the number of bits they store.

- **Strings:**

  Any sequence of displayable and non-displayable ASCII characters in the LabView data is known as Strings data type. It is indicated by pink data wire on the block diagram.

- **Array:**

  - Arrays can be distinguished as a group of specific data types. They are determined by thicker data wires. In array, we can store numeric, string, double or even Boolean data types.
  - An array control or indicator is created by first choosing the array shell and then depositing a data object into it.
  - Initially, the block diagram terminal of the array shell is black to indicate that the data type is not defined. Once a data type is assigned to the array shell, the block diagram takes the colour and lettering (in [ ] brackets) of the data type.

Array Operations:

LabVIEW provides many functions to manipulate arrays. Some of the commonly used functions are Array size, Initialize Array, Build Array.

Array Size: The Array Size function returns the number of elements in each dimension of array. If the input array is n-dimensional, the output will be a one-dimensional array with n-elements.

Initialize Array: The Initialize Array function creates an n-dimensional array with elements with values specified at the input. One important use of this function is to allocate memory for arrays. The data type of the array is determined by what is connected to the input.

Build Array: The Build Array function is used to concatenate multiple arrays or to append extra elements to an array.

- **Cluster:**

  - Clusters are the data type which consists different type of data in a single unit i.e., you can combine Boolean, numeric and string in a single value. This data type is indicated by thicker brown colour data wire.
  - Custers group data elements of mixed types, such as a bundle of wires, as in a telephone cable, where each wire in the cable represents a different element of the cluster.
  - A cluster is similar to a record or a structure and text-based programming languages.
  - T Clusters of numeric sometimes referred to as points, have a brown wire pattern and data type icon.
  - Numeric clusters can be wired to numeric functions, such as add or square root, to perform the same operation simultaneously on all elements of the cluster.

- **Waveform:**

The waveform data type that is used to store and display periodic signal measurements. The waveform creates a graph and charts of the particular data. It provides you with exact and precise information about your data in charts and graphs form.

- **Enum:**

  Enums are the combination of data types mainly consisting of a pair of data values i.e., string and a numeric value.

Concept of WHILE & FOR Loops:

- LabVIEW consists of FOR Loop and WHILE Loop. These loops are used to control repetitive operations.
- LabVIEW includes structures like the While Loop, For Loop, Case structure, Sequence structure and Formula Node.

While Loop in LabVIEW

A While Loop is a structure you use to execute a block of LabVIEW code repeatedly until a given condition is met. When the VI runs, the code inside the While Loop executes, and then the terminal condition is evaluated. While Loop execution does not depend on iteration count; thus, a While Loop executes indefinitely if the condition never occurs.

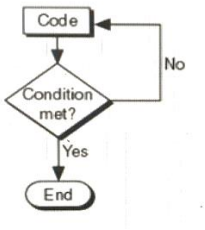

Fig. 2 While Loop Flowchart

While Loops should be used when:

- You want your code to run indefinitely.
- You want to run code until a condition is met

For Loops in LabVIEW

A For Loop is a structure you use to execute a block of code a set number of times. When the VI runs, the iteration count is evaluated, and then the code is executed. A For Loop can be configured to conditionally stop code execution in addition to its iteration-based exit. In these cases, the code will execute until the count terminal setting is reached *or* the condition is met – whichever happens first.

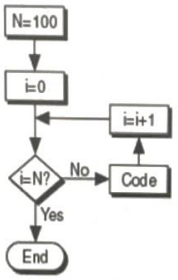

Fig. 3 LabVIEW For Loop flowchart

For Loops should be used when:

- You need to run code for a certain number of times.
- You want to run code until a condition is met OR a set number of iterations, whichever comes first.

Graph:

- Graphs display only the most recent array of values written to them and do not have a history of data.
- Graphs and charts differ in the way they display and update data.
- Virtual instrumentation with graphs usually collects the data in an array and then plot the data to the graph, which is like 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, the current reading or measurement in context with data previously acquired can be viewed.

Structures

It contains one or more sub diagrams or cases exactly one of which executes when the structures execute. The value wired to the case selector determines which case to execute.

Sequence Structure:

- A sequence of a structure is used especially when the program needs to be executed in a linear process (especially in sequential order).
- Within LabVIEW, it is quite hard to take control of the execution order. If a calculation is happening in a linear function, i.e., the next calculation is actually dependant on the current input, then the process will be executed in sequential order.
- But, if the calculations are happening in parallel, it gets complex and the process cannot be managed in a sequential structure. So, most of the calculations are actually forced to take up a sequential structure approach.

Case structure:

Case structures are widely used in the scenarios where the program or the users have to take a decision. The decision is categorized into two options, i.e. True or False. At any point of execution, only one condition (or case, i.e. True or False) will be executed.

Components of case structure

- **Selector label**- Displays the value(s) for which the associated case executes. You can specify a single value or a range of values. The selector label to specify the default case.
- **Sub-diagram(case)-** Contains the code that executes when the value wired to the case selector matches the value that appears in the selector label
- **Case selector-** Selects which case to execute based on the value of the input data. The input data can be a Boolean, string, integer, enumerated type, or error cluster

Formula Nodes:

It is a software that is a convenient, text-based node you can use to perform complicated mathematical operations on a block diagram using the C- syntax structure. It is most useful for equations that have many variables. The text-based code simplifies the block diagram and increases its readability.

Need of software-based instruments for industrial automation:

- Industrial Automation software such as Siemens- Simatic Manager is used to control and monitor the process (automatic) from a longer distance. An operator or Engineer does not need to go where the manual switches are to start the process. He/she just needs to start the process from the software. And another reason is, now a days we don't need to use timers and different kind of relays in the field. Everything is available in the software. So, its cost-effective.
- The foundation of Industry 4.0 is the Industrial Internet of Things (IIoT). The basis of this connectivity is the sensors that transmit data to a control computer. Without sensors collecting information, the IIoT doesn't exist.
- Industry 4.0 is a structured system, that uses the concept of a Smart Factory and uses the cybernetic solution and digital communication in industrial products and services. It is a transversal process irrespective of the business area, that makes industrial production entirely automated and interconnected.
- Smart manufacturing to establish remote and monitoring assistance system, a solution that actively contributes to improving the performance in industrial plant.
- Companies can supervise, check, and set the variables of the machine or the entire production process.

Industrial Instrumentation

The process of measuring and controlling various quantities in production processes using various instruments or industrial elements is defined as "industrial instrumentation".

Purpose of Industrial Instrumentation

- To control any quantity, one must primarily measure that particular quantity. After measuring the desired quantity, the measured values are transmitted for indication, calculation, or control purposes, either with manual or automatic operation.

- In an automatic control operation, the quantity can be controlled using the control signal sent by the CPU to the control devices.
- In practice, the sensors read the different field data such as flow, pressure, displacement, vibrations, etc., and transmit them to the control systems that will regulate these variable quantities.
- In industry, control is normally carried out by programmable logic controllers (PLC) or Distributed control systems (DCS).

A practical example of industrial instrumentation.

- The thermostat of the electric geyser is an example of a measurement and control system, in which the temperature of the water is the "Process" under control.
- In this example, the thermostat typically has two functions. One is sensing and the other is control, while the heater coil adds heat to the water to increase the temperature. Input water from the overhead tank extracts heat from the geyser to decrease the temperature.
- The job of this control system is to keep the water temperature at a comfortable level. Desired hot water temperature at desired value (called the set-point).

Main elements of industrial instrumentation

1. Sensors

There are various sensors used as input devices in real-time control and instrumentation applications, but the most commonly used sensors are pressure sensor, flow sensor, temperature sensor, level sensor, position and displacement sensor, etc.

2. Controllers

Most controllers or valves are generally implemented using mechanical or electronic systems. But recent industrial controllers in systems used in the industry rely on computers. Hence, it makes it easier to implement complex control algorithms. The most used control systems in industrial instrumentation are,

- PLC (Programmable Logic Controller)
- DCS (Distributed Control Systems)
- SCADA (Supervisory Control and Data Acquisition system)

3. Actuators

The actuator is used to control the mechanism or the system based on the signal given to it by converting the electrical signal into a large power action. The output signal from controllers or control systems like DCS, SCADA, or PLC used in industrial instrumentation is used to control the actuators. Actuators are generally controlled by electrical current, fluid pressure, pneumatic pressure, and this energy is converted into mechanical energy or power.

There are different types of actuators to know:

- Hydraulic actuator
- Pneumatic actuator
- Electrical actuator
- Mechanical actuator

These actuators are used to control output devices such as valves, motors, relays, contactors, emergency lights, etc.