BE/EE189 Design and Construction of Biodevices
Lecture 1

LabVIEW Programming – Basics

- Virtual instrument and LabVIEW
- The LabVIEW development environment
- Basic programming with LabVIEW
- Navigation window
- Property nodes
- Cleaning up the block diagram

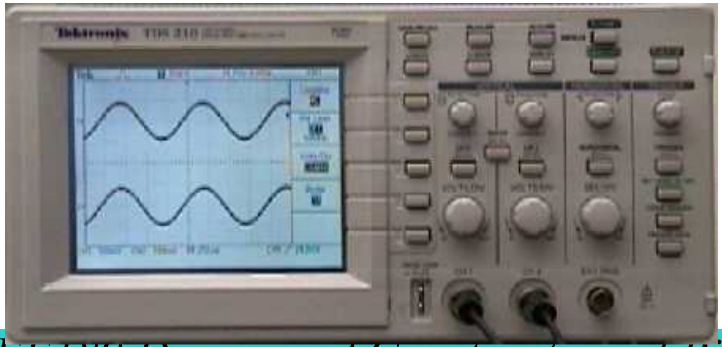
Virtual Instrument

- Virtual instrumentation can be defined as:
  - *A layer of software and/or hardware added to a general purpose computer in such a fashion that users can interact with the computer as though it were their own custom-designed traditional electronic instrument.*

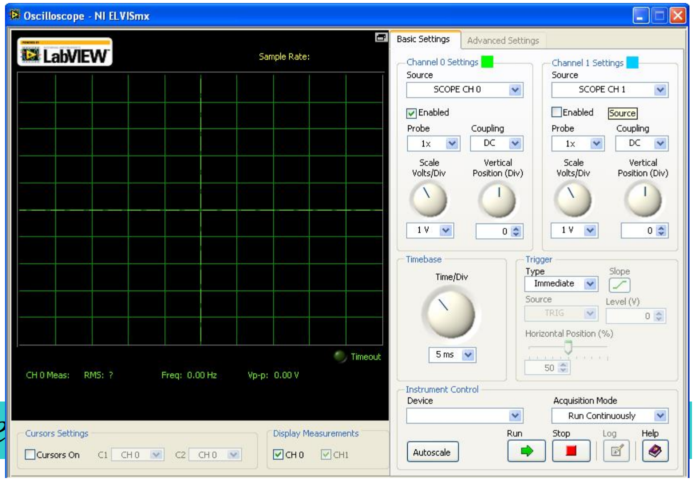
From Virtual Bio-instrumentation by Jon B. Olansen and Eric Rosow

- LabVIEW programs are called virtual instruments (VIs).

Real Oscilloscope



Virtual Oscilloscope



LabVIEW Graphical Development System

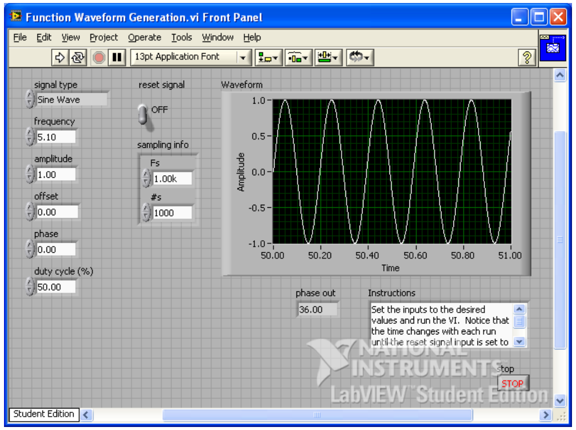
- Graphical programming environment – different from text-based programming language such as C or Fortran.
- Compile code for multiple OS and devices

Virtual Instrumentation Applications

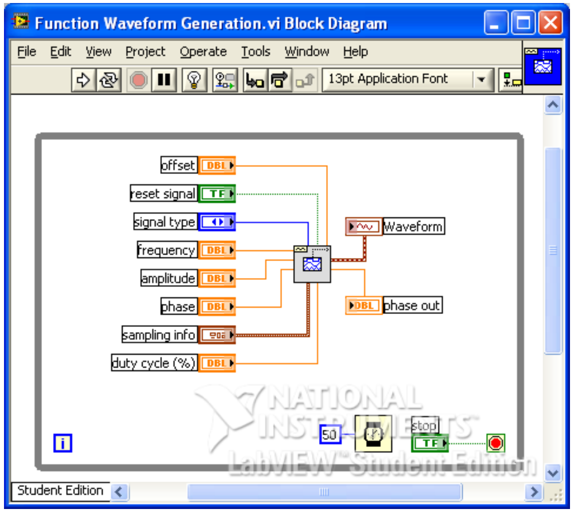
- Design
  - Signal and image processing
  - Embedded system programming (PC, DSP, FPGA, etc.)
  - Simulation and prototyping
- Control
  - Automatic controls and dynamic systems
  - Mechatronics and robotics
- Measurements
  - Circuits and electronics
  - Measurements and instrumentation

VI Programming Environment

- **Front panel**: interface to your VI program
  - Controls = inputs
  - Indicators = outputs



- **Block diagram**: program code in a graphical form
  - Terminals corresponding to front panel controls and indicators
  - Constants, functions, subVIs, structures
  - Wires connect components together



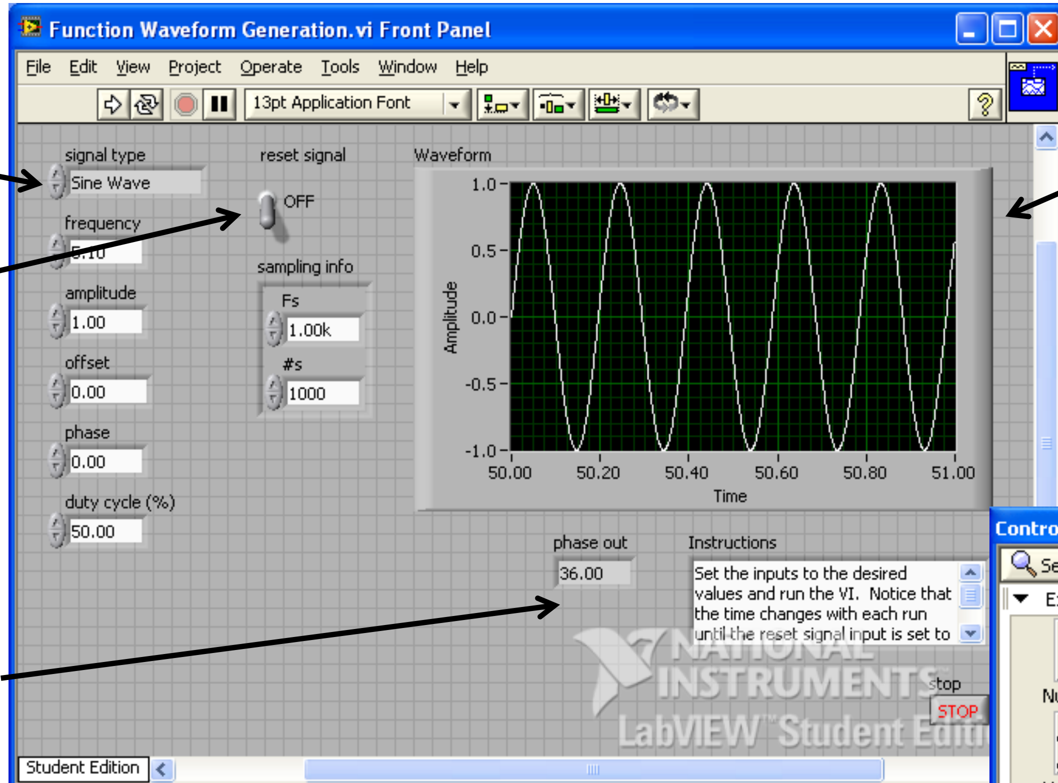
Front Panel and Controls Palette

Numeric control

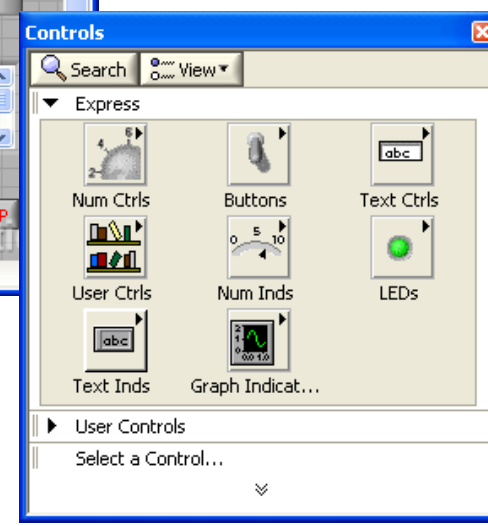
Boolean control

Numeric indicator

Waveform graph

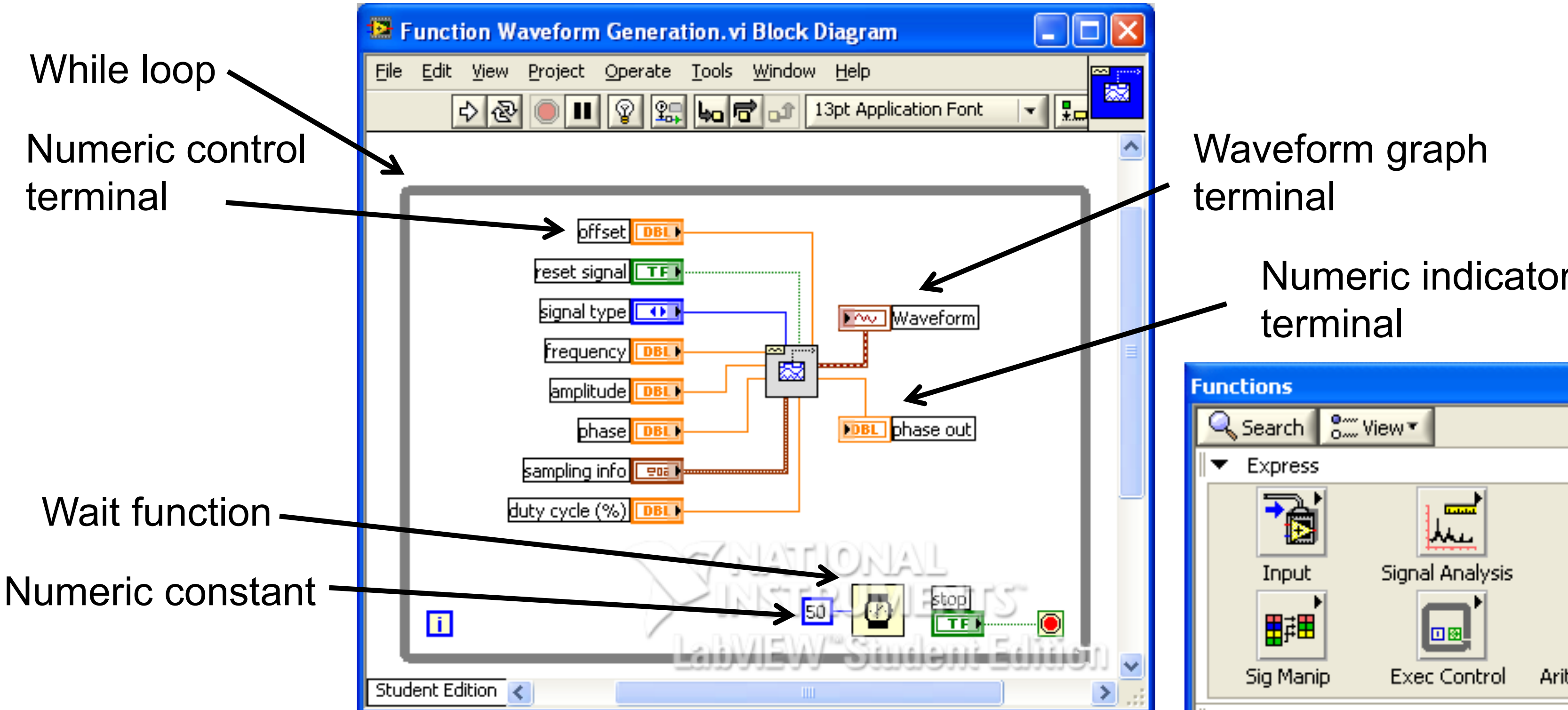


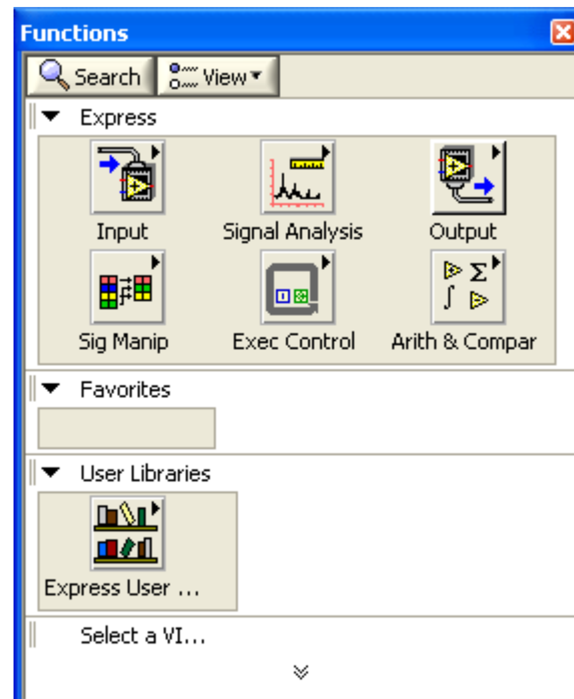
Controls Palette



- Controls and indicators can be created using the controls palette.

Block Diagram and Functions Palette

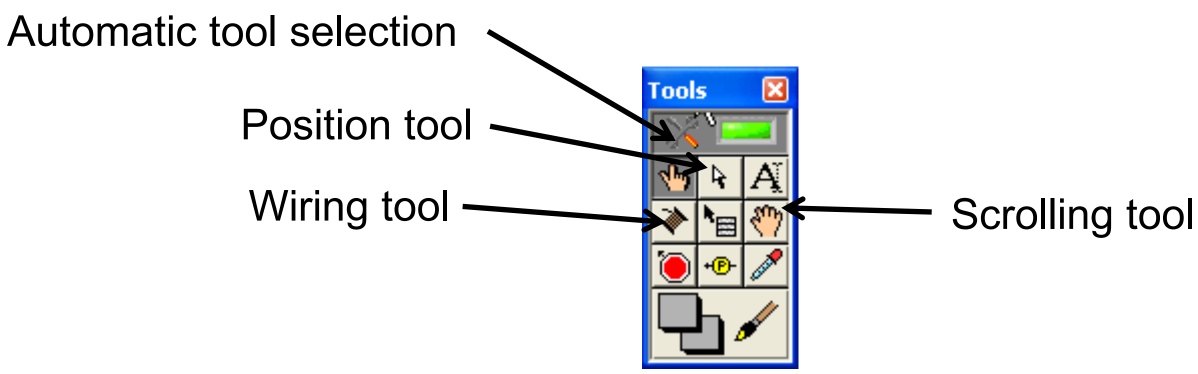




- Functions, execution controls, etc. can be created using the functions palette.

Tools Palette

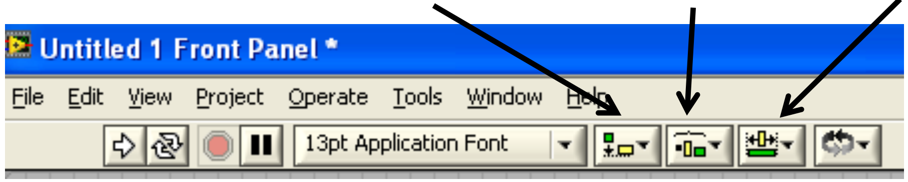
- A tool is a special operation mode of the mouse cursor. You use tools to perform specific editing functions.
- Automatic tool selection: labVIEW will automatically selects the corresponding tool as you move the cursor over objects on either the front panel or the block diagram.



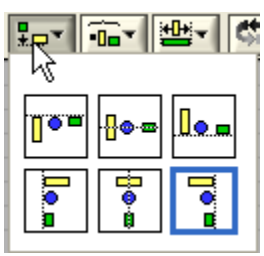
Aligning Objects

- After selecting the desired objects for alignment, you can align, distribute, or resize them – make things neat and pretty.

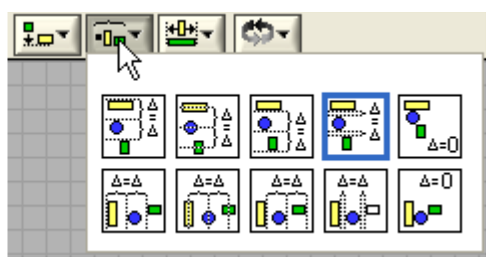
Align objects Distribute objects Resize objects



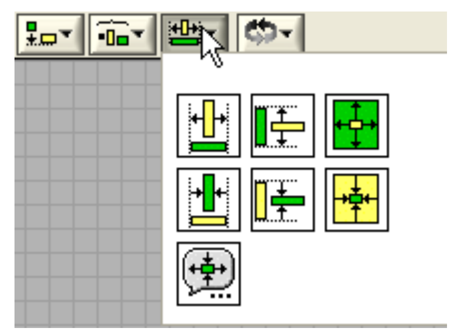
Align objects menu



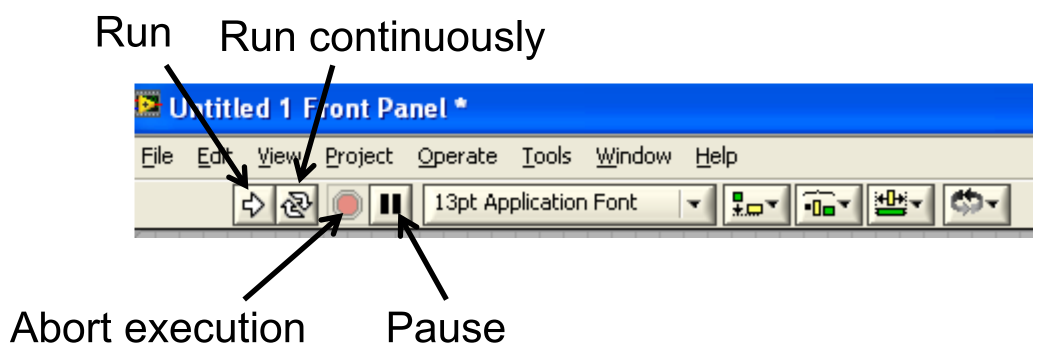
Distribute objects menu

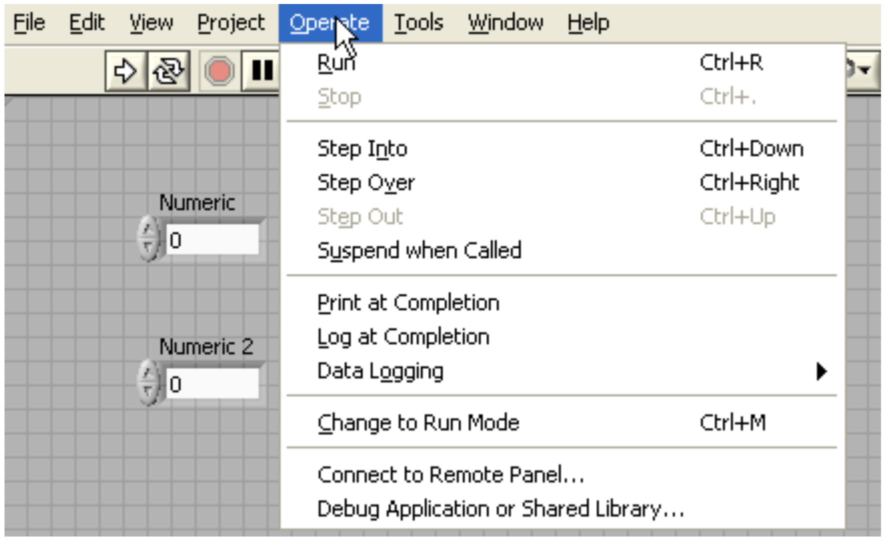


Resize objects menu



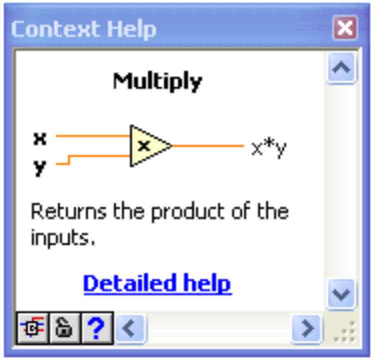
Execution Control





Context Help Window

- “Show Context Help” (Ctrl+H) – show context help associated with the selected objects.



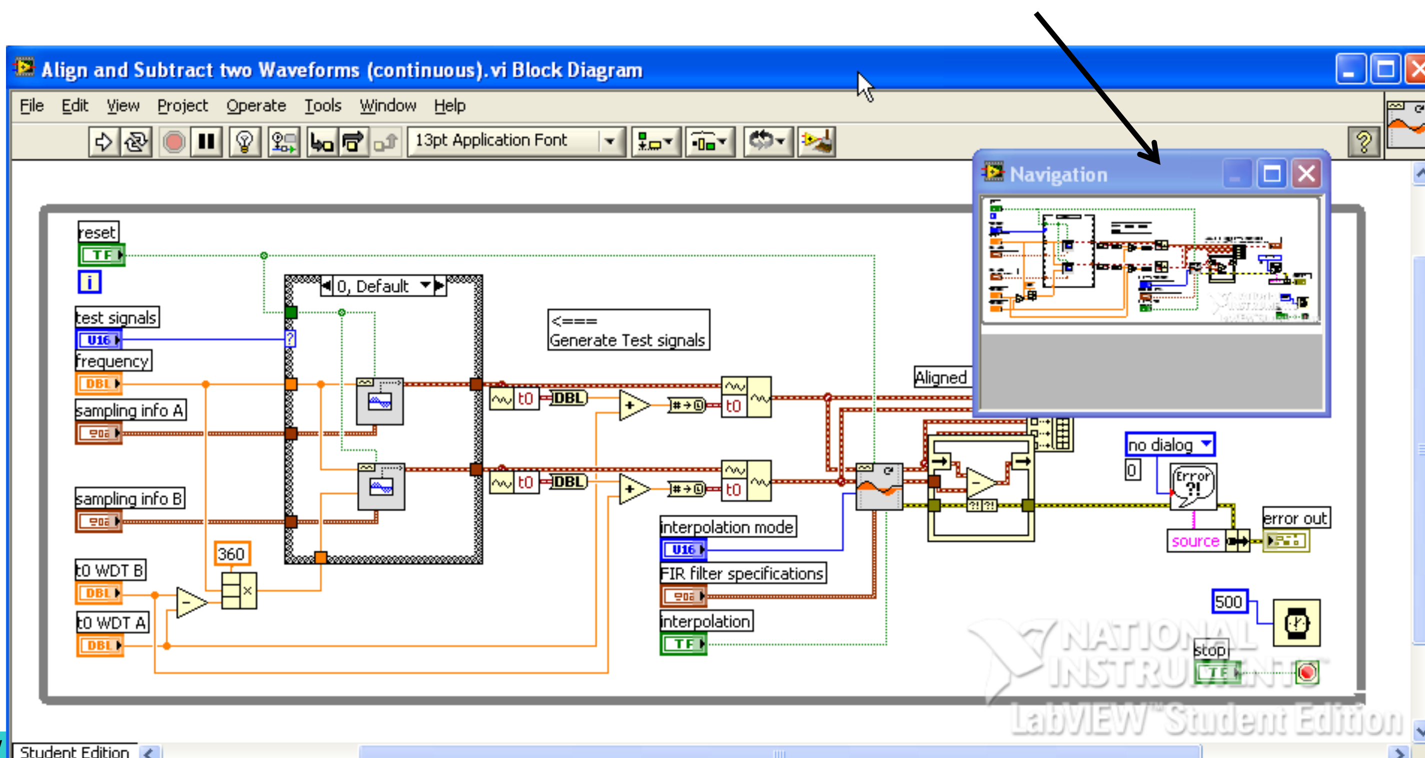
Basic Concept of LabVIEW Programming

- **Modular programming**: a given task is divided into a series of simpler subtasks which is implemented separately and then assembled.
- **Data flow programming**: the icons (subtasks) in the block diagram are wired together to allow data flow. The execution of a VI is governed by the data flow.

Navigation Window

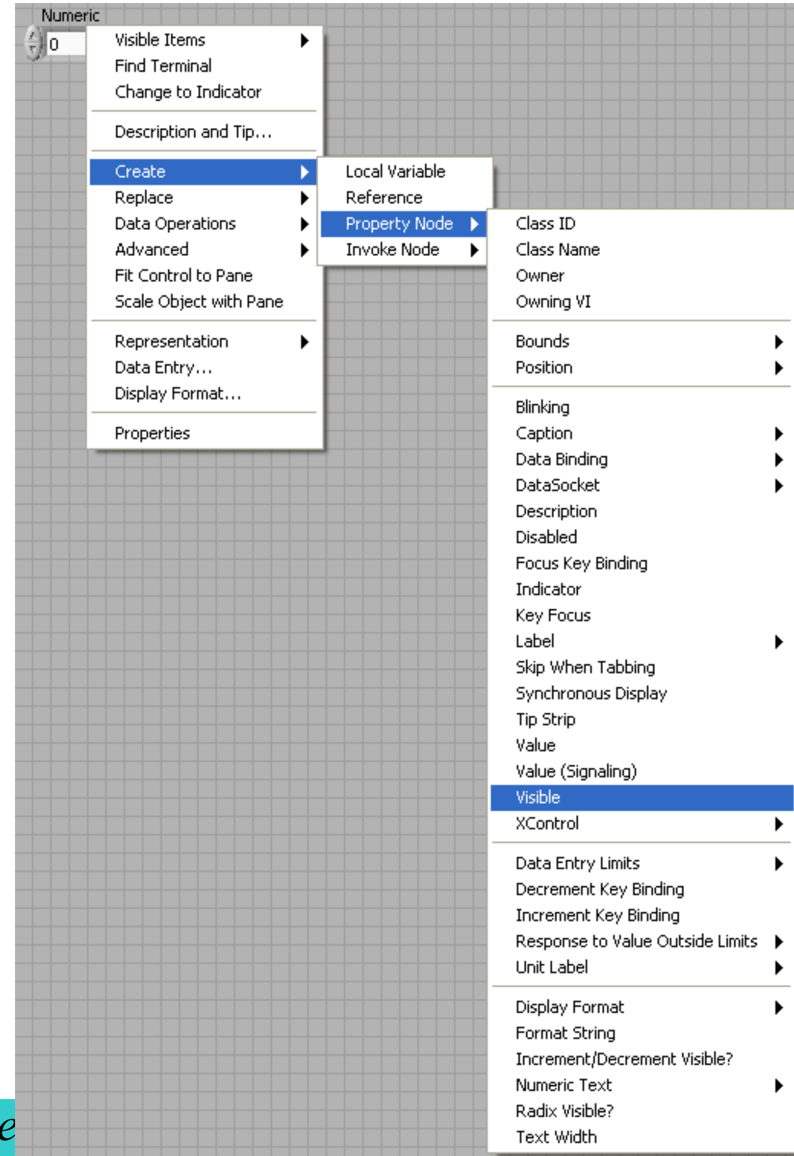
- For complicated VIs, the navigation window can display an overview of the active front panel in edit mode or the active block diagram.

Navigation window



Property Nodes

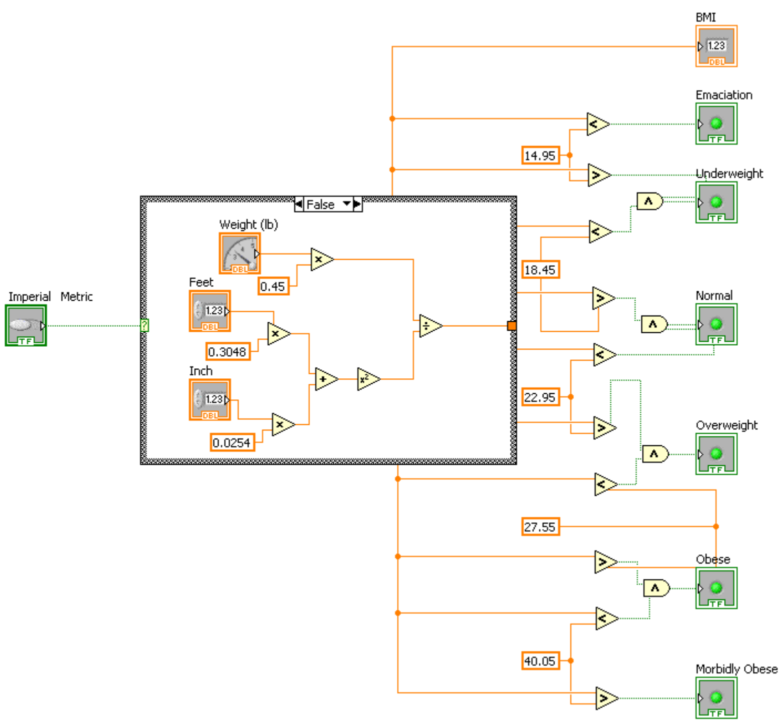
- Allow you to *set* or *get* the properties of objects. For example, in some applications, you might want to make a front panel object vanish while the VI is running if a Boolean input is True.



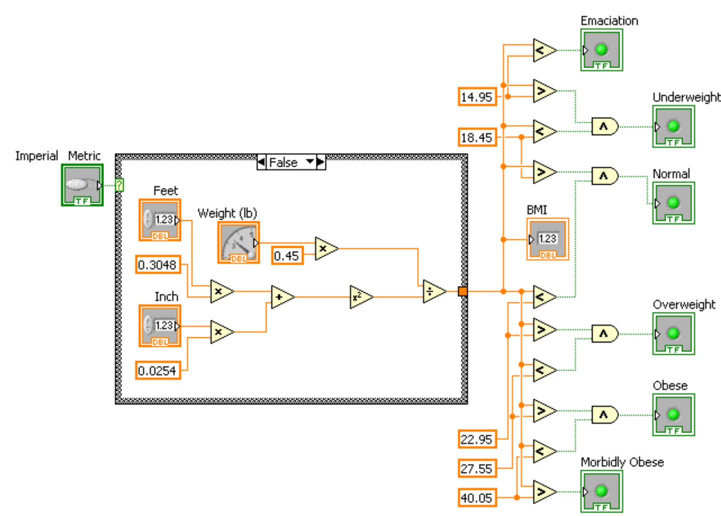
Clean Up the Block Diagram

- Cleaning up for easier-to-understand diagrams and for debug purposes

Before cleanup

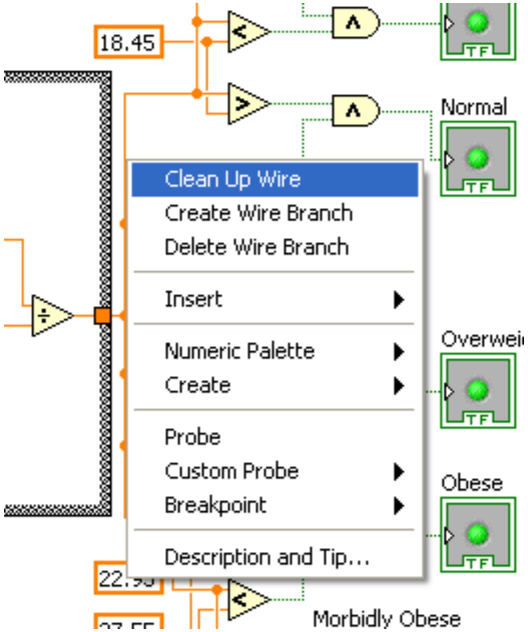


After cleanup



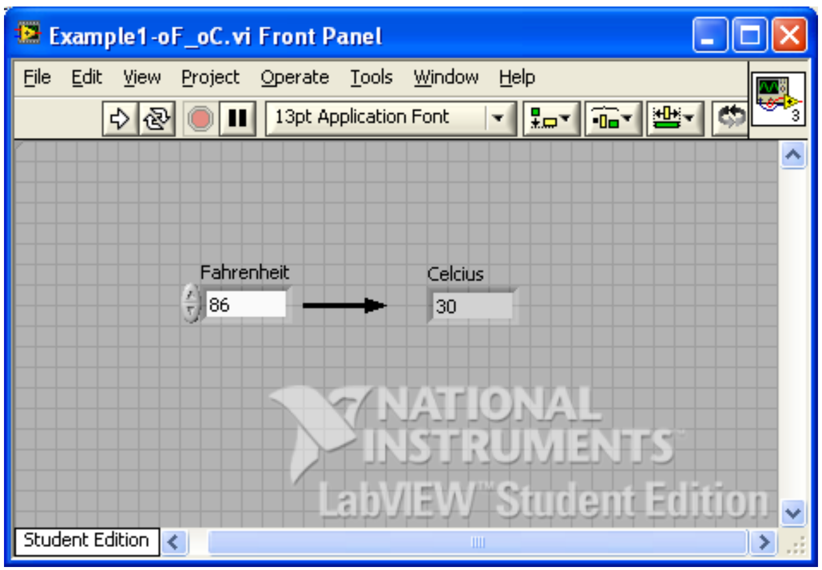
Clean Up Wire

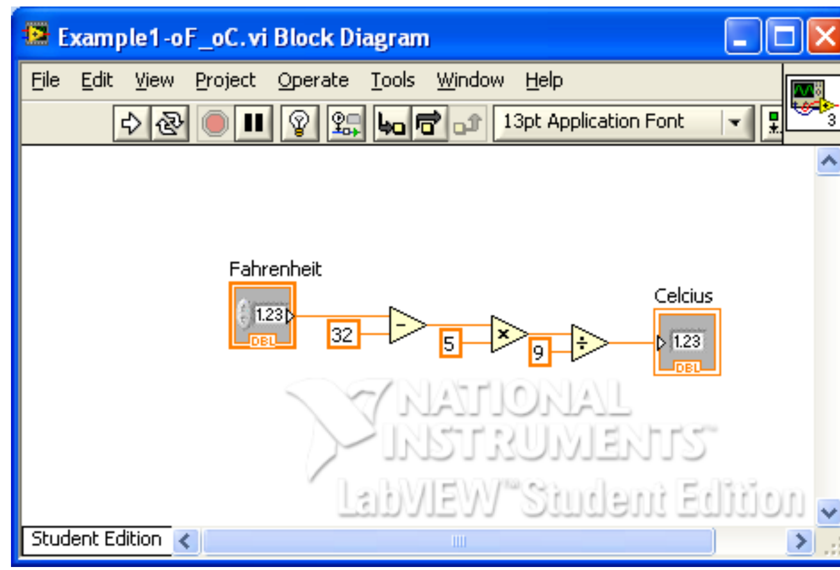
- You can clean up a wire by right clicking it, and select “Clean Up Wire” – Very useful.



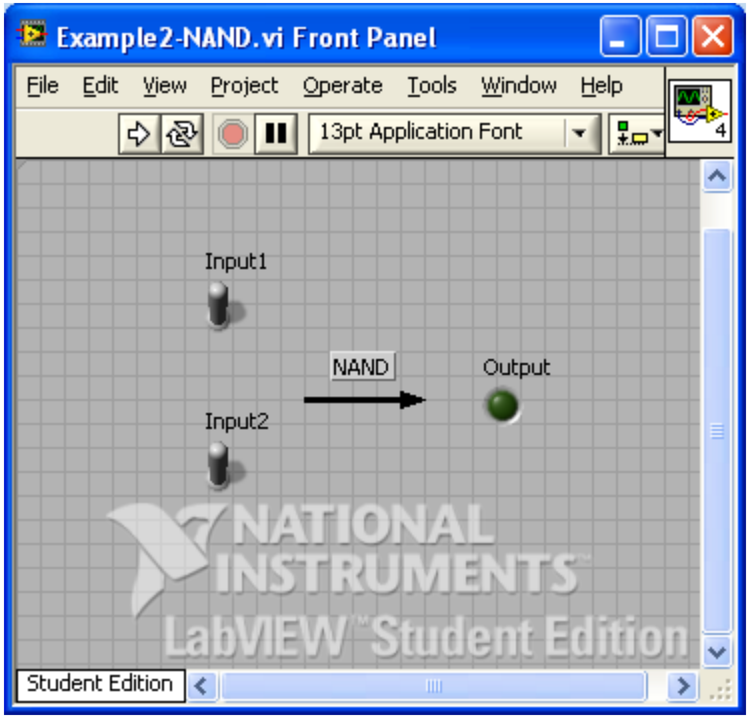
Example 1.1 – ºF to ºC Converter

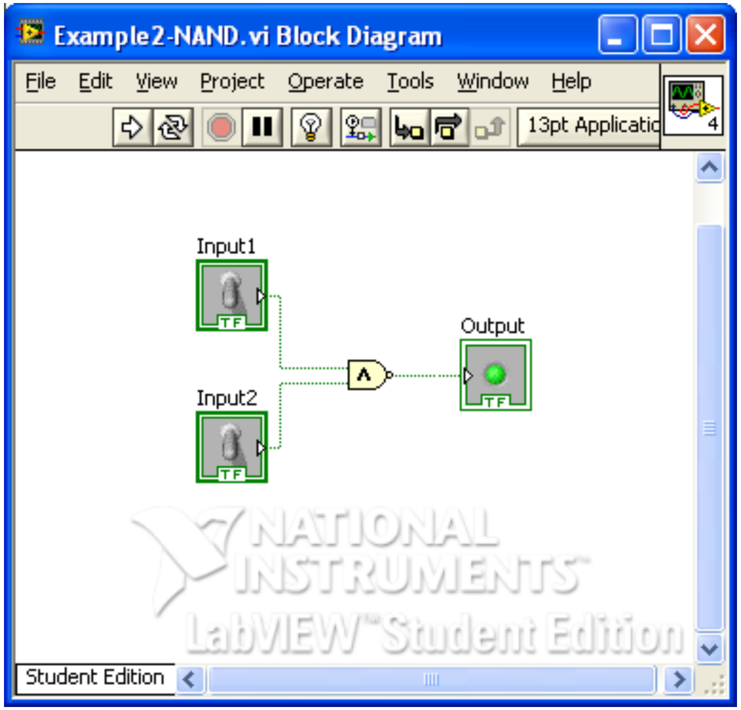
- [°C] = ([°F] - 32) × 5/9





Example 1.2 – NAND Gate





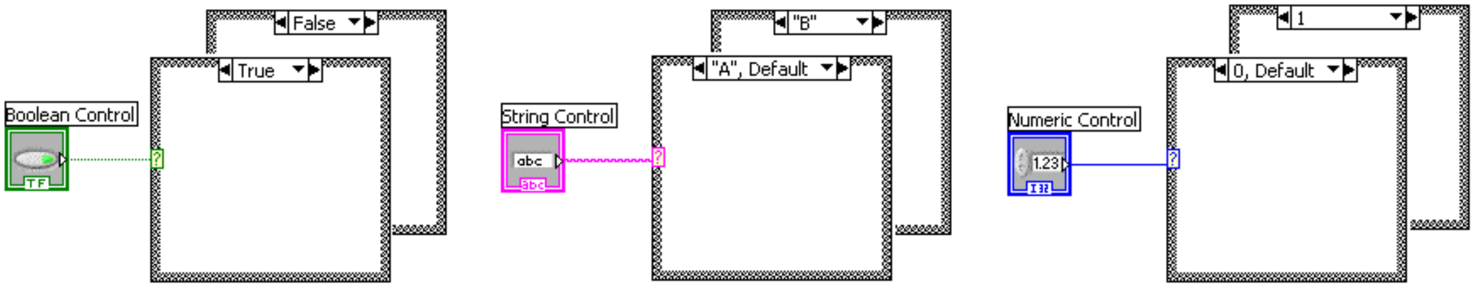
BE/EE189 Design and Construction of Biodevices
Lecture 2

LabVIEW Programming – More Basics, Structures, Data Types, VI

- Case structure
- Debugging techniques
- Useful shortcuts
- Data types in labVIEW
- Concept of subVI
- Creating a subVI
- Using a VI as a subVI
- Error checking and error handling
- The VI hierarchy window

Case Structure

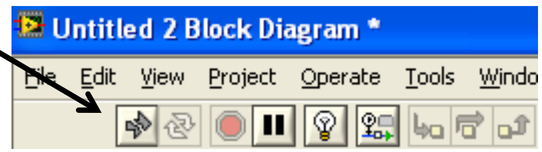
- CASE structure has two or more subdiagrams.
- Only one subdiagram to execute at a time based on the value of the selector.
- Each subdiagran must provide output value for the CASE structure.

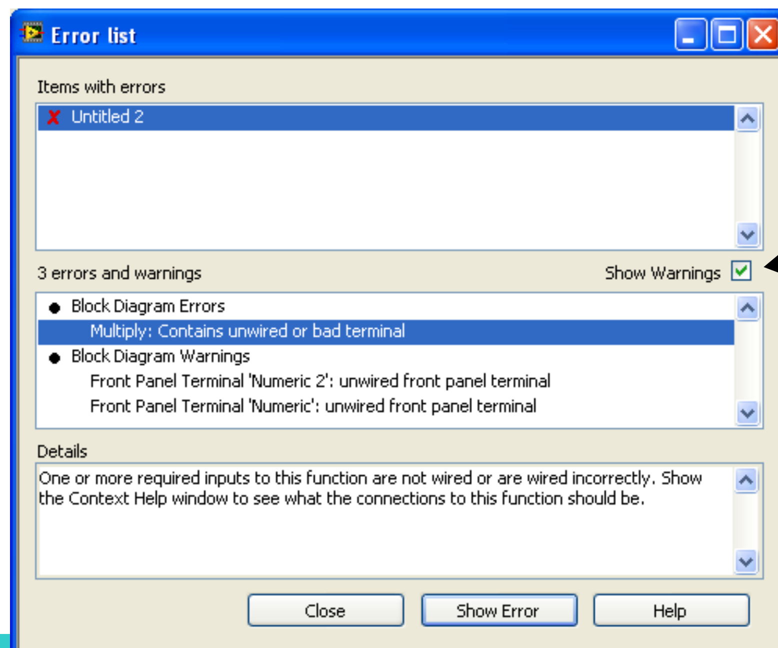


LabVIEW Debugging Techniques – Finding Errors

- Click on the broken run button to show the error list

Broken run button



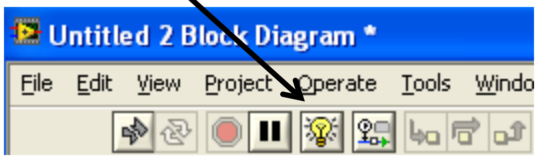


Show warnings

LabVIEW Debugging Techniques – Highlight Execution

- Highlight execution can be used to show the animation of the VI execution
- Will reduce performance

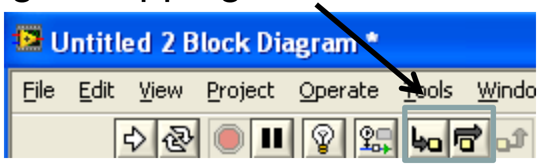
Highlight execution button



LabVIEW Debugging Techniques – Single-Stepping

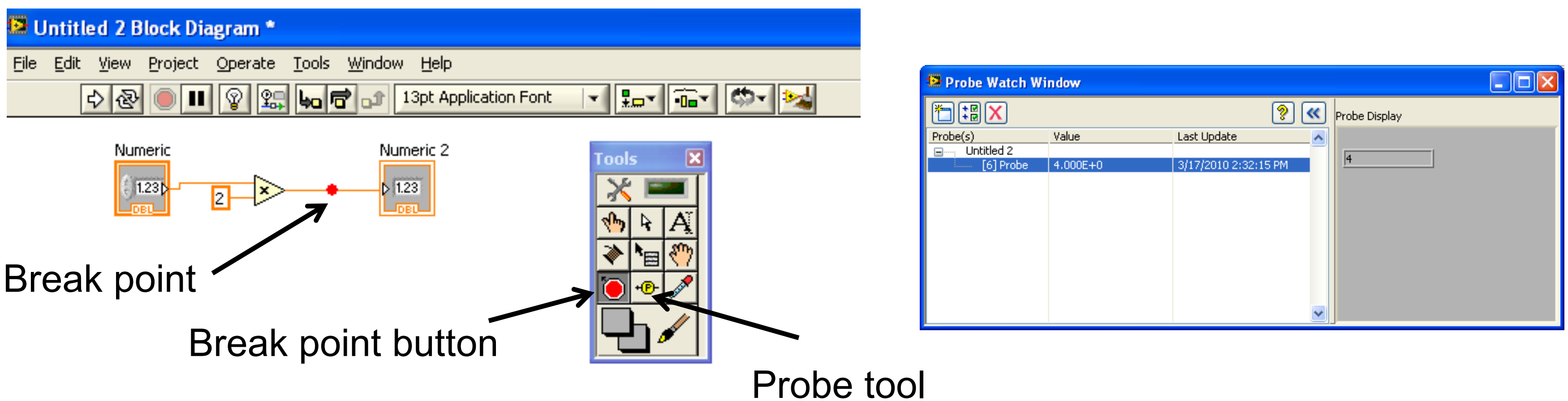
- In the single-step mode, you can either “step into” or “step over” the node in the block diagram.

Single-stepping buttons



LabVIEW Debugging Techniques – Breakpoints and Probes

- You can halt execution at certain locations by using the breakpoint.
- Use the probe tool to view data as it flows through a block diagram wire. A probe watch window will display the current data value.

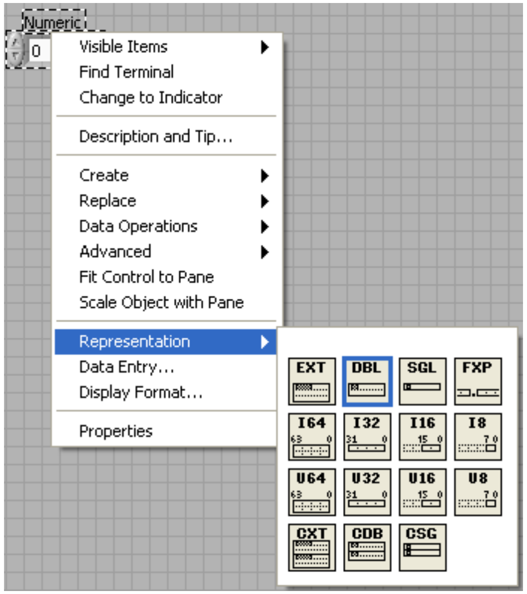


Useful Shortcuts

- Ctrl-S: Save a VI
- Ctrl-R: Run a VI
- Ctrl-E: Toggle between the front panel and the block diagram
- Ctrl-H: Toggle the **Context Help** window on and off
- Ctrl-B: Remove all bad wires
- Ctrl-W: Close the active window
- Ctrl-F: Find objects and VIs

Data Types

- Numeric
  - integer: signed, unsigned + precision → I64, I32, I16, I8, U64, U32, U16, U8
  - floating-point number: single, double, extended precision
- String: a sequence of characters
- Boolean: true or false

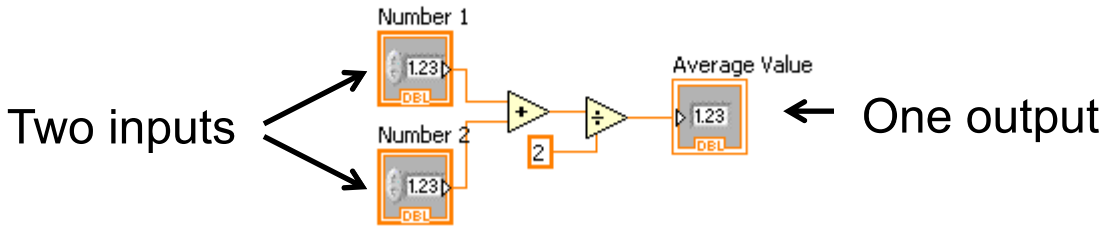


SubVI

- A subVI is a stand-alone VI that is called by other VIs, similar to a subroutine or function in text based programming languages.
- Remember **Modular programming**: a given task (top-level VI) is divided into a series of simpler subtasks (functions or subVIs) which is implemented separately and then assembled.

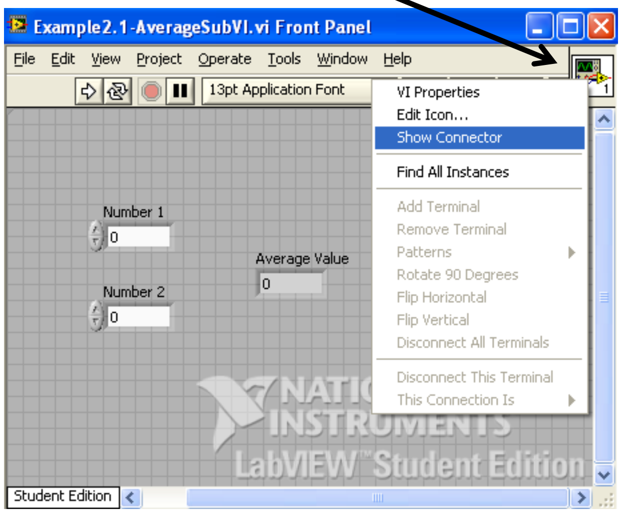
Creating subVI – 1: Develop an independent VI

- Define the input and output of your subVI, and develop the VI.

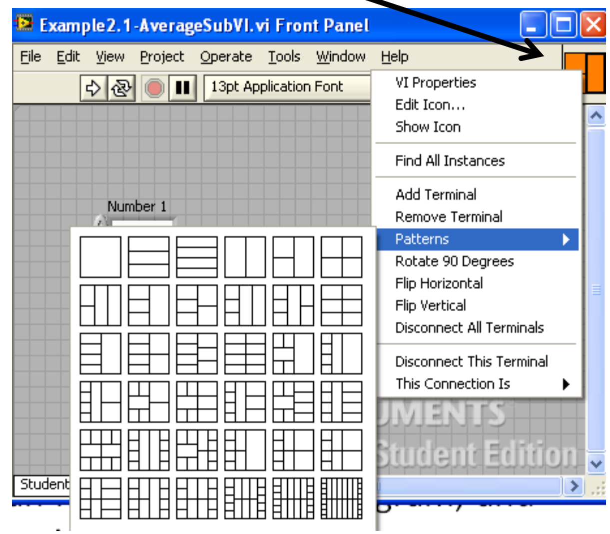


Creating subVI – 2: Assigning Input and Output

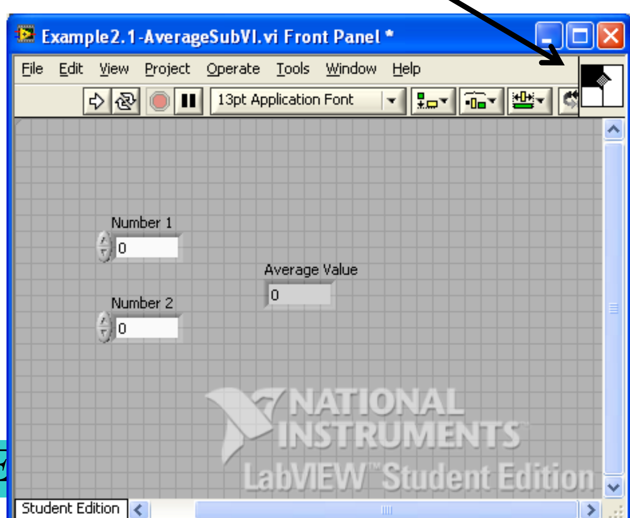
Right –click on the icon plane, select “Show Connector”



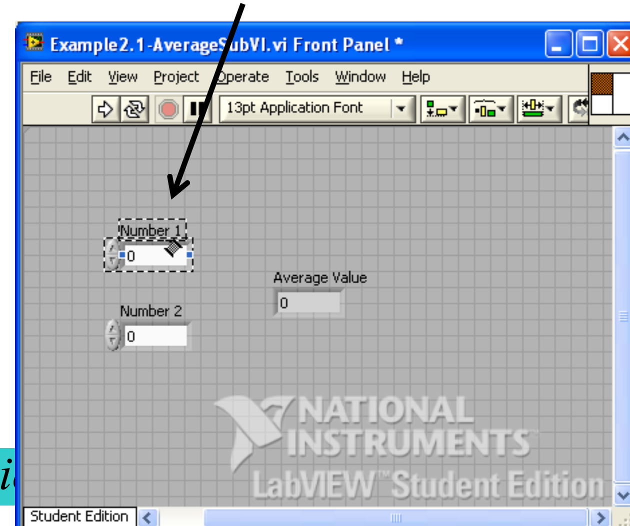
Right –click on the icon plane, select “Patterns” to define number of inputs and outputs



Click on the terminal with the wiring tool



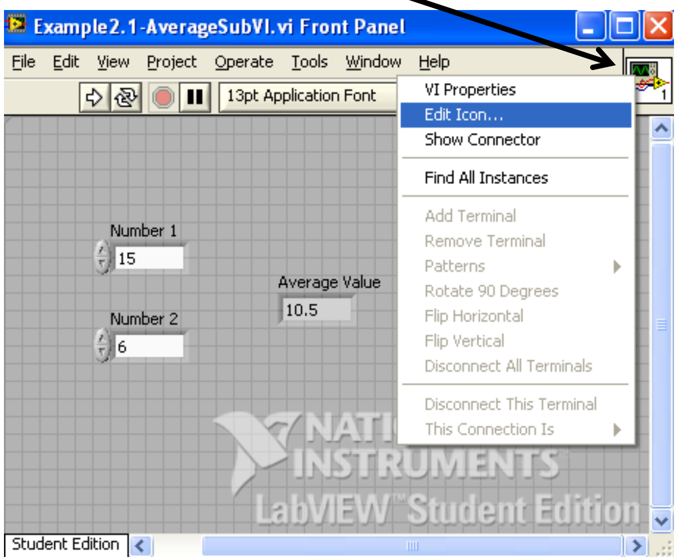
Then click on the control or indicator with the wiring tool



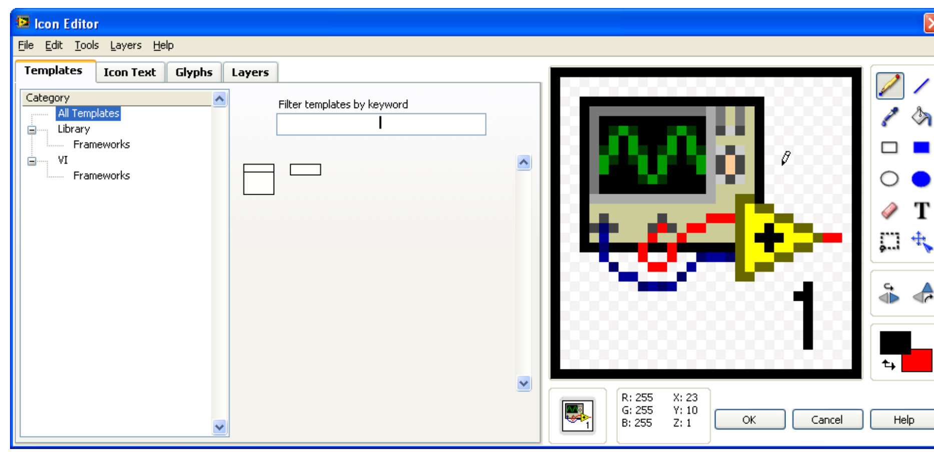
Creating subVI – 3: Edit Icon

- A subVI is represented by an icon in the block diagram, and you can customize the icon picture.

Right –click on the icon plane, select “Edit Icon…”

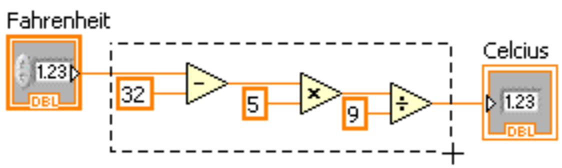


Edit the icon with the icon editor

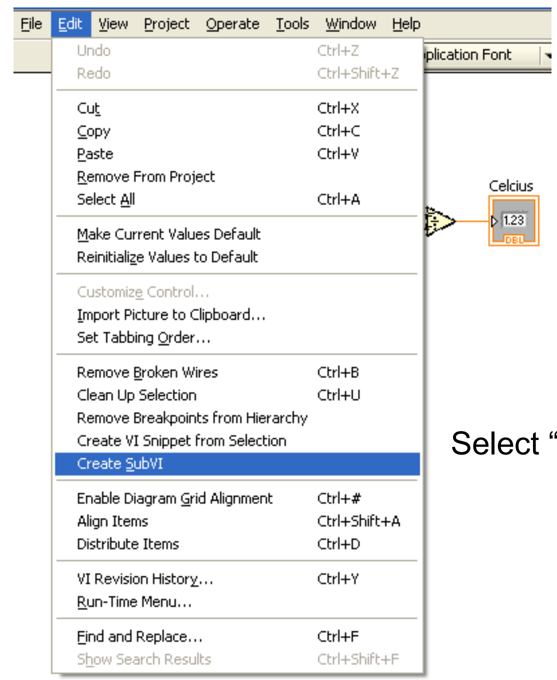


Creating a subVI from a selection

- You can select components of the main VI and group them into a subVI



Select components

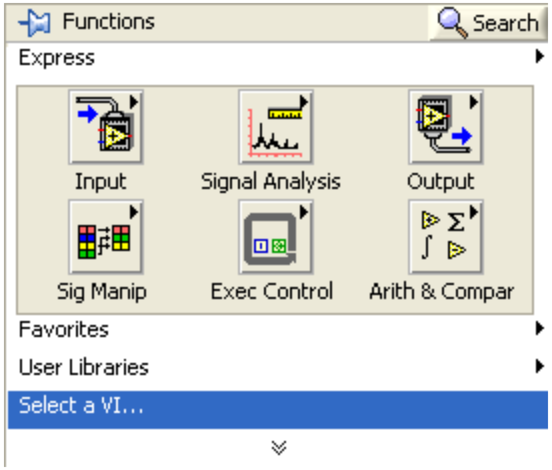


Select “Create SubVI”

SubVI created

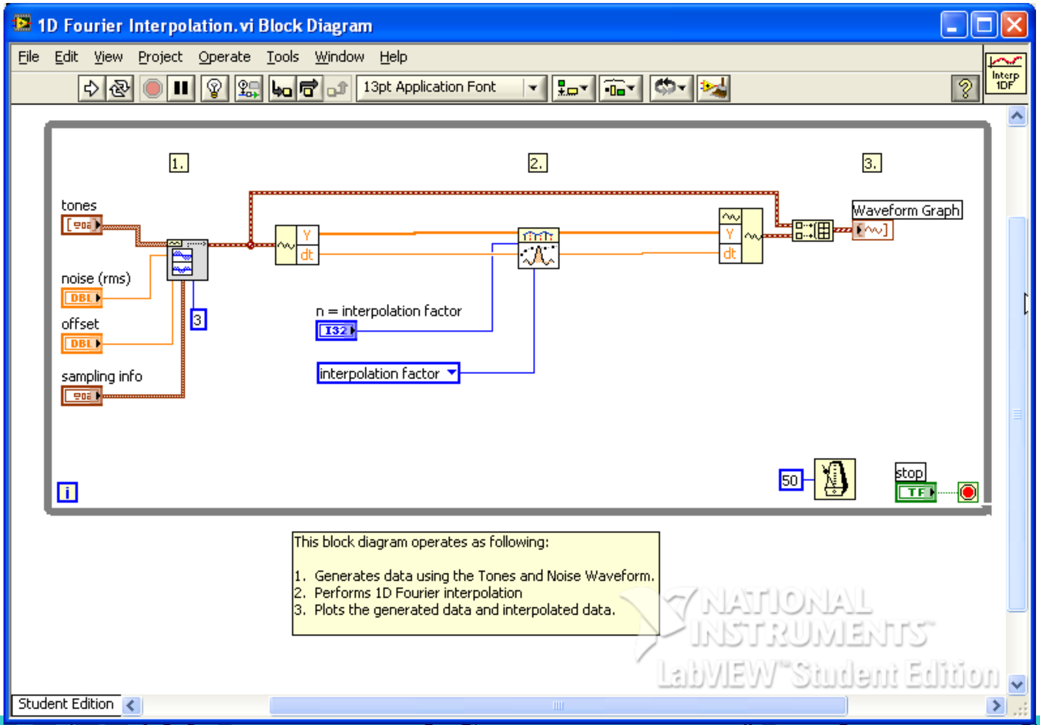
Calling SubVI

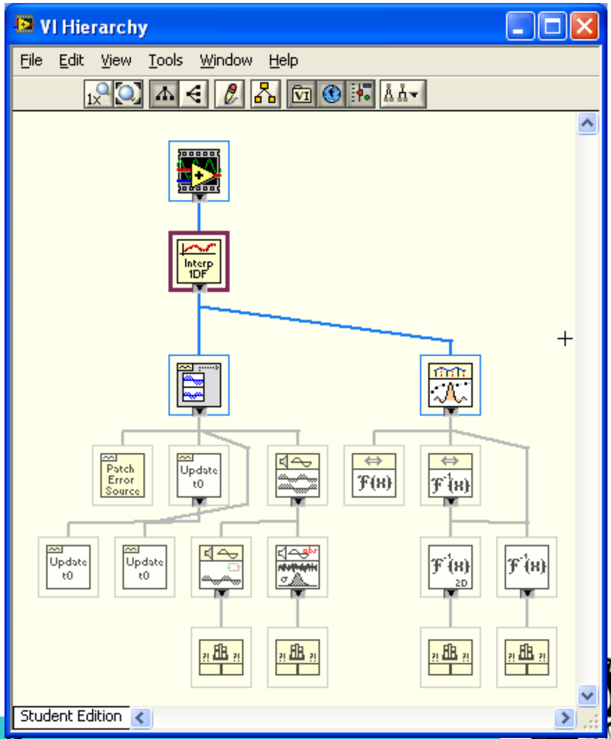
- In the functions palette, select “Select a VI…”, and choose the VI that you developed as a subVI. Just like you add a built-in function.



The VI Hierarchy Window

- Display a graphical representation of the hierarchical structure of all VIs in memory and shows the dependencies of top-level VIs and sub VIs (View >> VI Hierarchy).

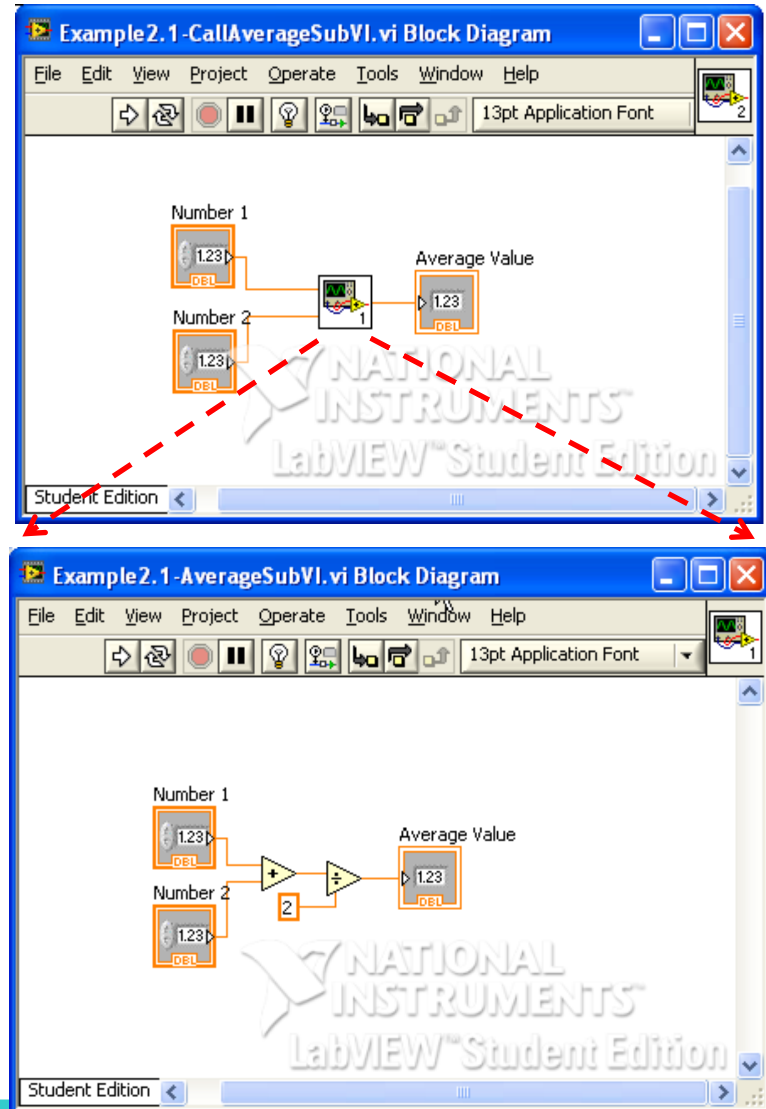




Example – SubVI to Calculate Average Value

Top-level VI

SubVI



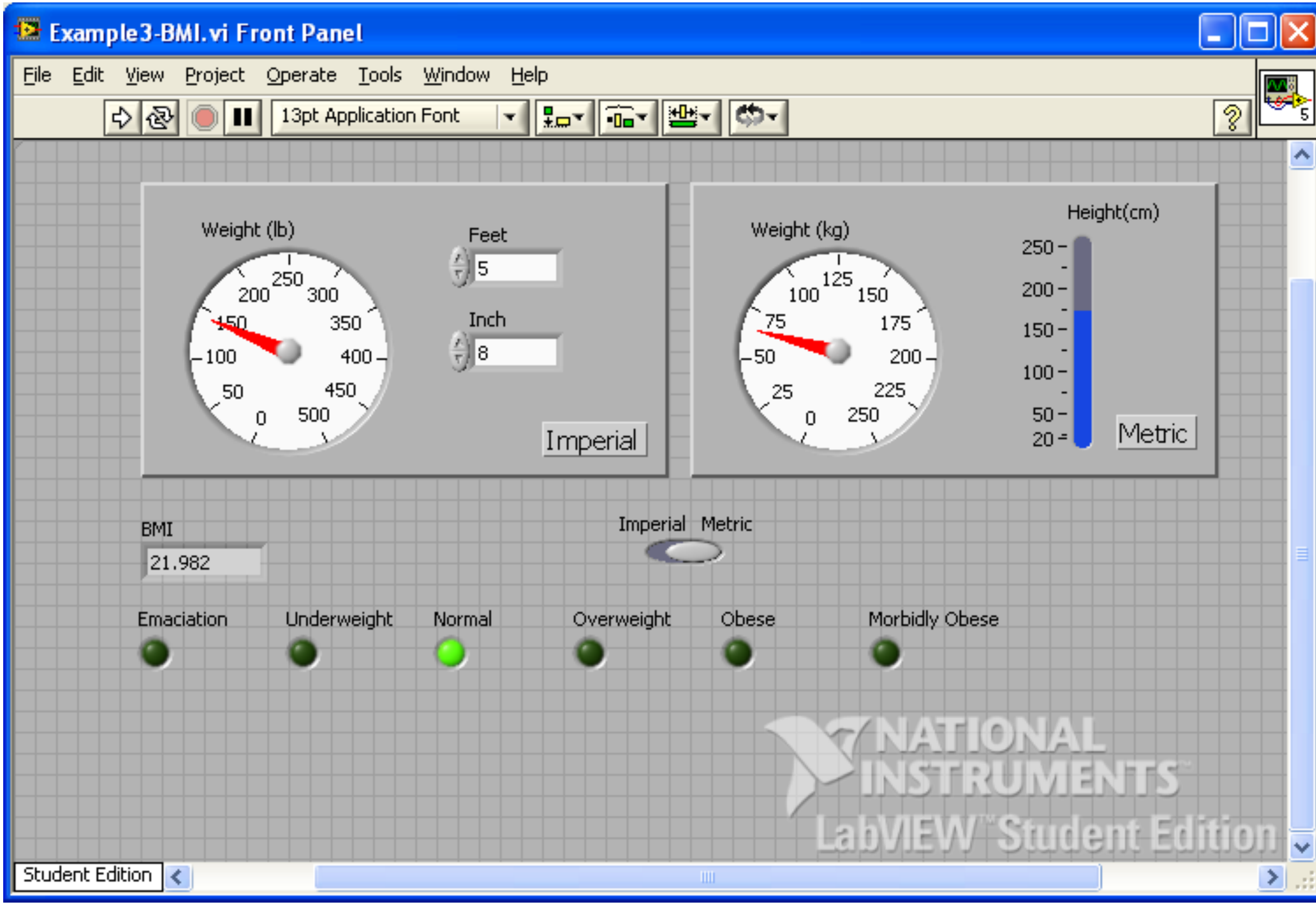
Work Example 2.1 – BMI Calculation and Display

- Body mass index (BMI) = body weight (kg)/(height(m))^2

Category	BMI range – kg/m^2
Emaciation	less than 14.9
Underweight	from 15 to 18.4
Normal	from 18.5 to 22.9
Overweight	from 23 to 27.5
Obese	from 27.6 to 40
Morbidly Obese	greater than 40

http://en.wikipedia.org/wiki/Body_mass_index

Work Example 2.1 – BMI Calculation and Display



Work Example 2.2 – SubVI for Evaluating Blood Pressure

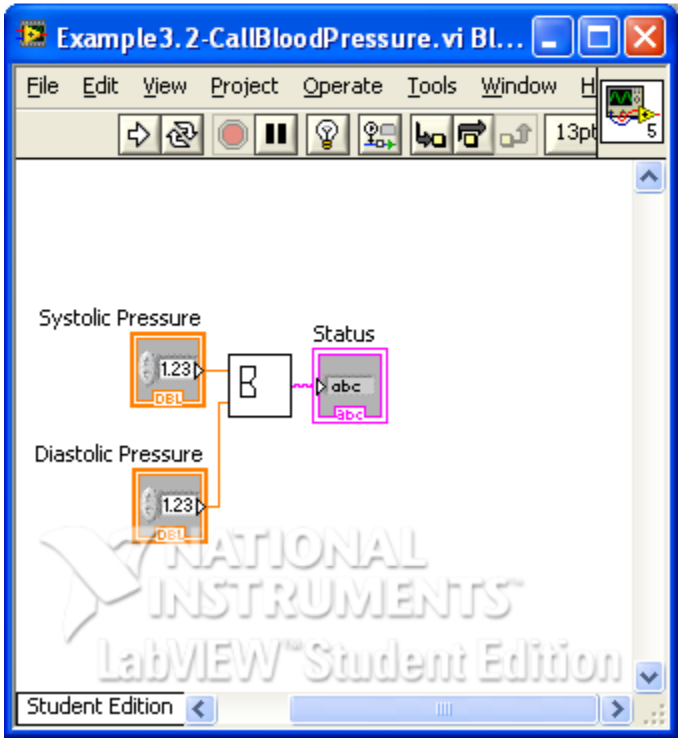
- For input systolic and diastolic pressure, output the status.
- Your systolic and diastolic numbers may not be in the same blood pressure category. In this case, the more severe category is the one you're in.

Category	Systolic (top number)		Diastolic (bottom number)
Normal	Less than 120	*And*	Less than 80
Prehypertension	120–139	*Or*	80–89
High blood pressure			
Stage 1	140–159	*Or*	90–99
Stage 2	160 or higher	*Or*	100 or higher

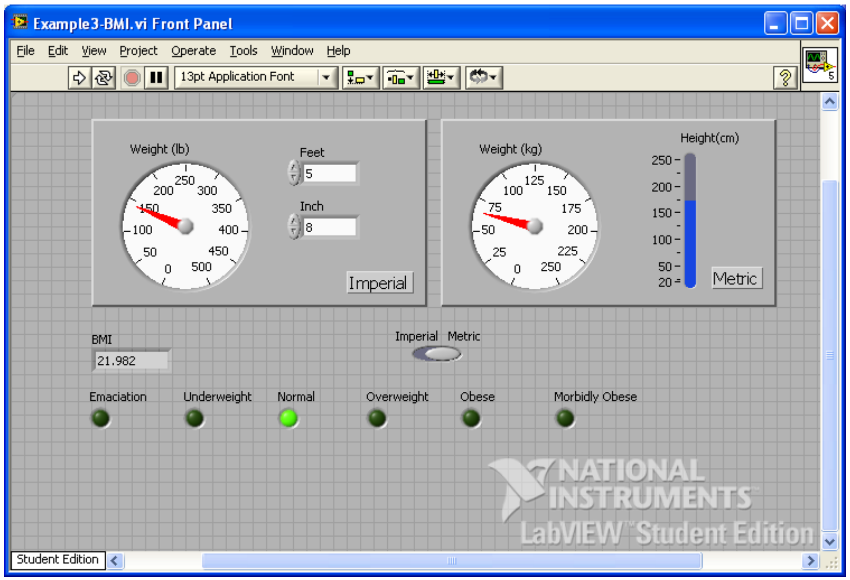
Work Example 2.2 – SubVI for Evaluating Blood Pressure

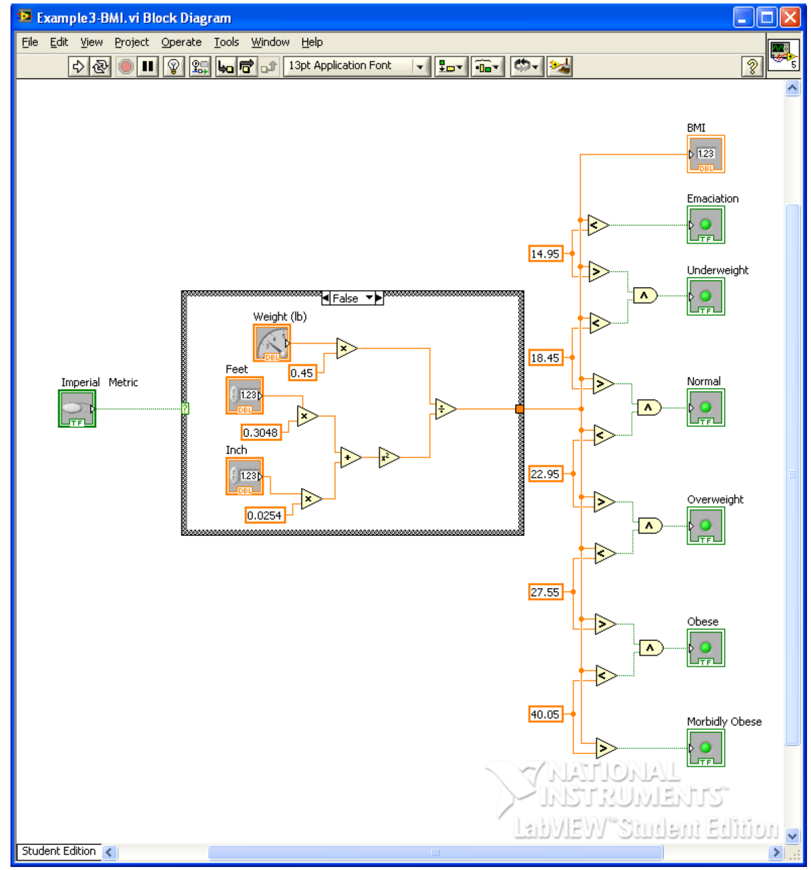
Top-level VI

What should the SubVI look like?



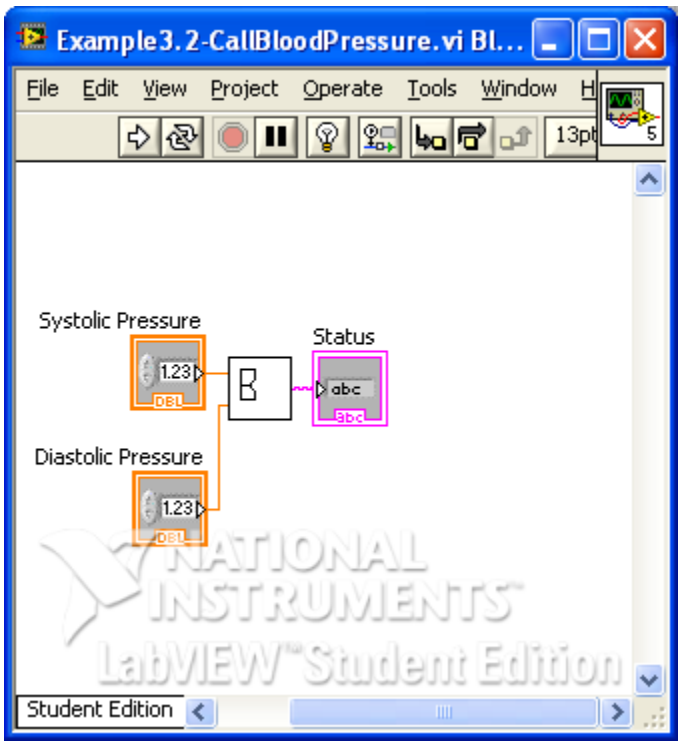
Answer


Example3-BMI.vi Front Panel
File Edit View Project Operate Tools Window Help
13pt Application Font
Weight (lb)
Feet
5
Inch
8
Imperial
Weight (kg)
Height(cm)
Metric
BMI
21.982
Imperial Metric
Emaciation
Underweight
Normal
Overweight
Obese
Morbidly Obese
NATIONAL INSTRUMENTS
LabVIEW Student Edition
Student Edition

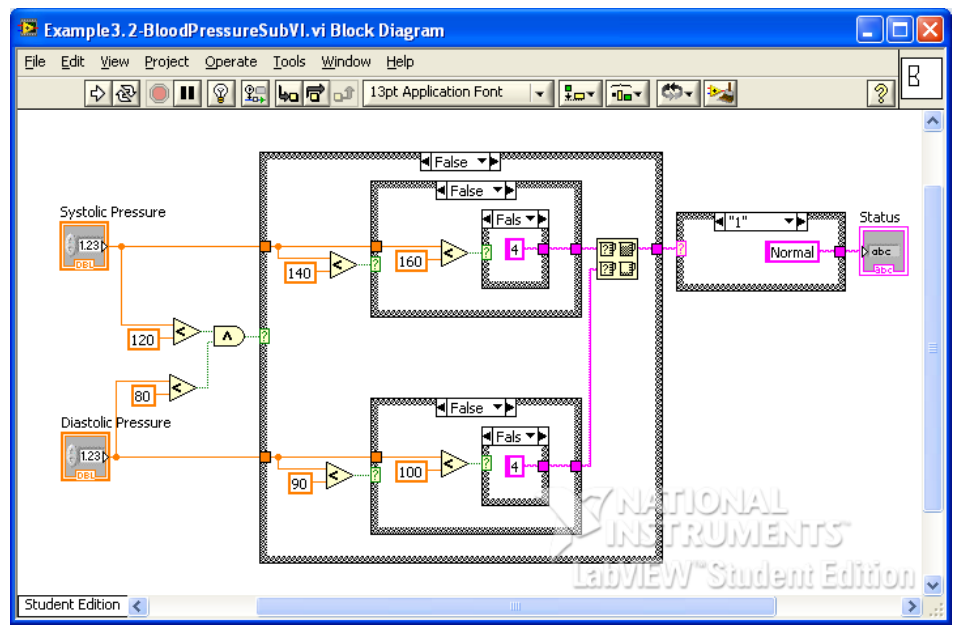

Example3-BMI.vi Block Diagram
File Edit View Project Operate Tools Window Help
13pt Application Font
BMI
Emaciation
Underweight
Normal
Overweight
Obese
Morbidly Obese
False
Weight (lb)
0.45
Feet
0.3048
Inch
0.0254
Imperial Metric
14.95
18.45
22.95
27.55
40.05
NATIONAL INSTRUMENTS
LabVIEW Student Edition
Student Edition

Answer

Top-level VI



SubVI



BE/EE189 Design and Construction of Biodevices
Lecture 3

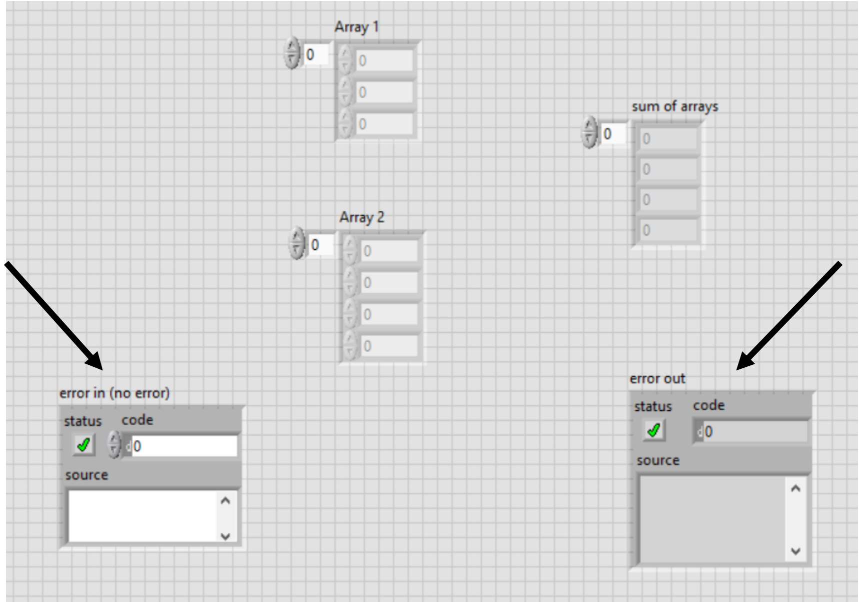
LabVIEW Programming – Error Handling & Structures

- Error Handling
- For loop, while loop
- Sequence structure
- Timing control
- Formula node
- Local variables

Error Handling

- LabVIEW allows you to define error codes and messages for your subVIs
- These errors can be passed along to and from multiple subVIs
- All NI-produced subVIs contain an error in and error out node

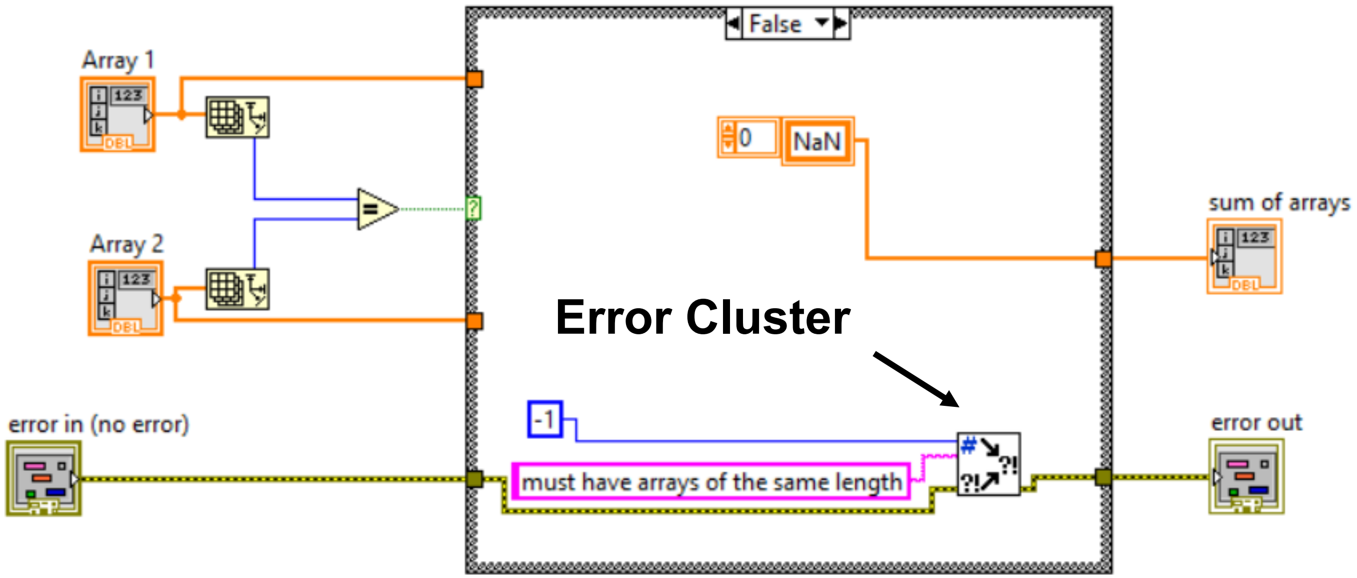
Error In:
The error in indicator reads errors passed to it by previous VIs. The default is "no error".



Error Out:
The error out indicator reads errors from previous VIs and the current VI.

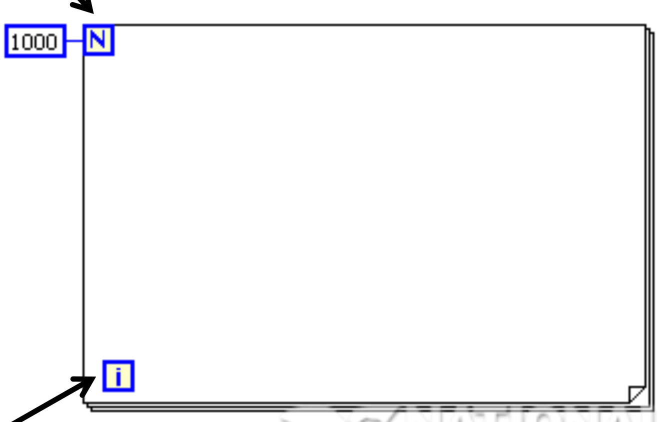
Error Handling

- Error codes (integer) and messages (string) are added in using the "error cluster" function in the "Dialog & User Interface".



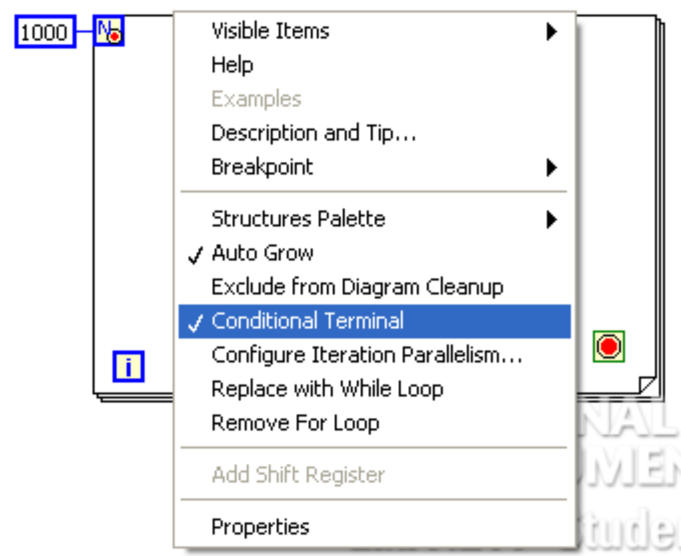
For Loop

Count terminal:
Number of times you want the loop to execute; Available for use inside the for loop.



Iteration terminal:
Number of times the loop has executed; Available for use inside the for loop.
Note: **starting from 0**

Conditional terminal:
A for loop with a conditional terminal executes until the condition occurs or until all iterations complete



While Loop

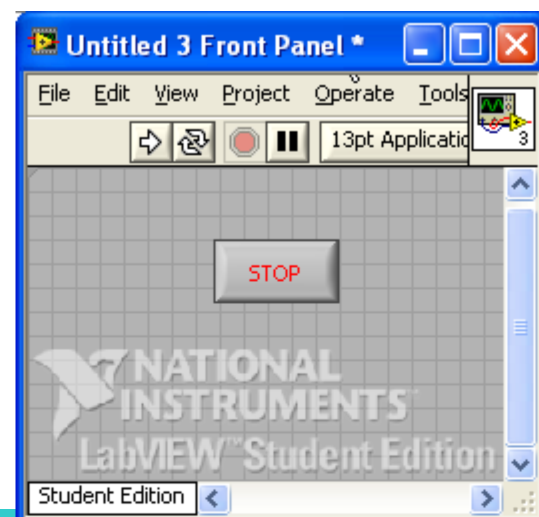
Iteration terminal:
Number of times the loop has executed; Available for use inside the for loop.
Note: **starting from 0**

Conditional terminal:
Loop executes until input is TRUE

Selected from Functions>>Programming>>Structures

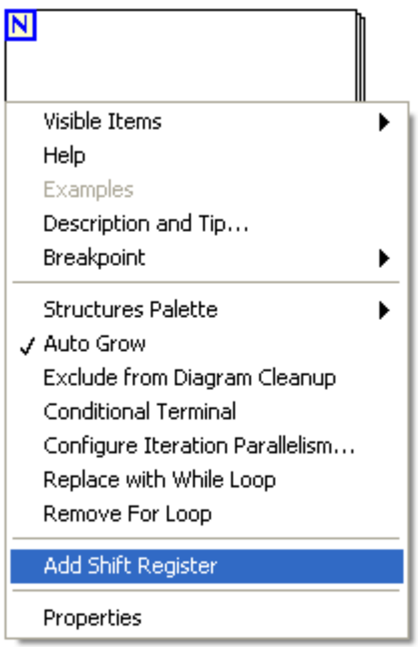
A while loop with a button to stop execution

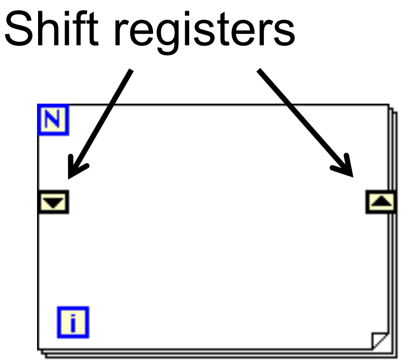
Selected from Functions>>Express>>Execution Control

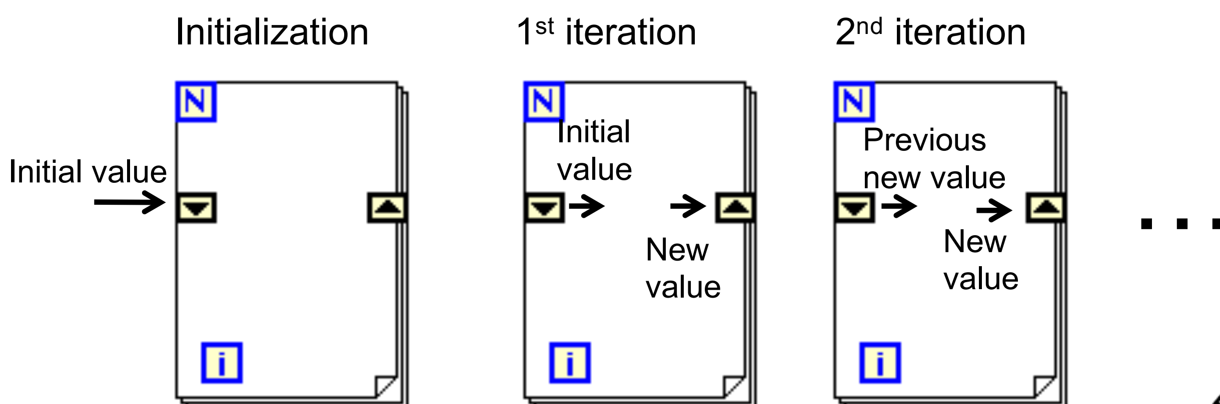


Shift Registers

- Shift registers transfer values from one iteration of a For Loop or While Loop to the next.

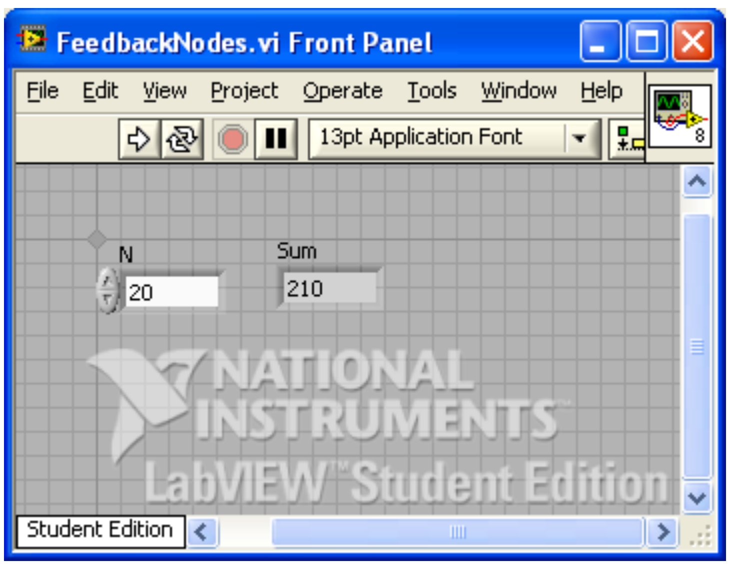


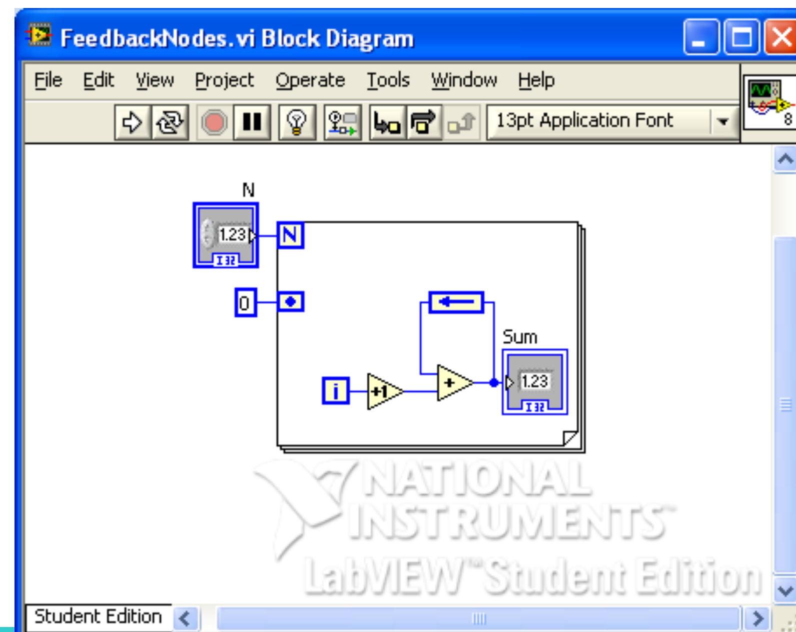




Feedback Nodes

- Similar to shift registers.
- Use the feedback node to avoid unnecessarily long wires.
- Can configure multiple delays.
- Example: calculate 1+2+…+N.

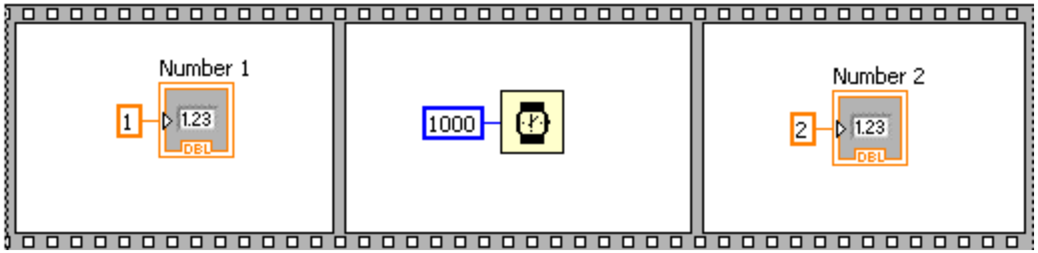




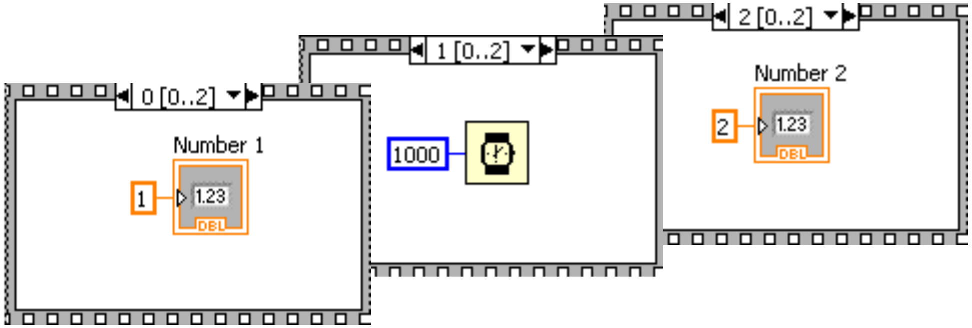
Sequence Structure

- The **sequence structure** executes subdiagrams sequentially.
- Two classes: **flat sequence** and **stacked sequence**.

Flat sequence

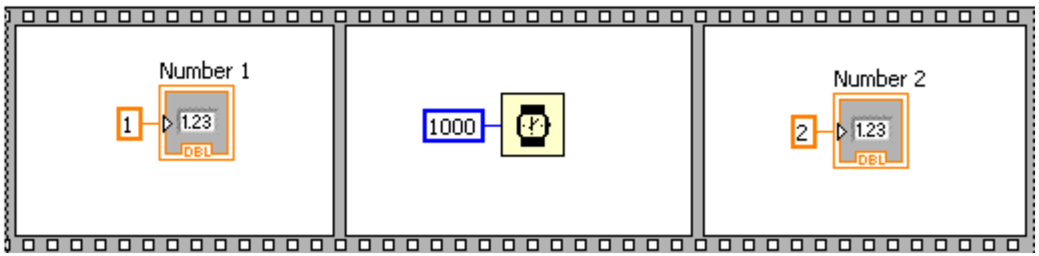


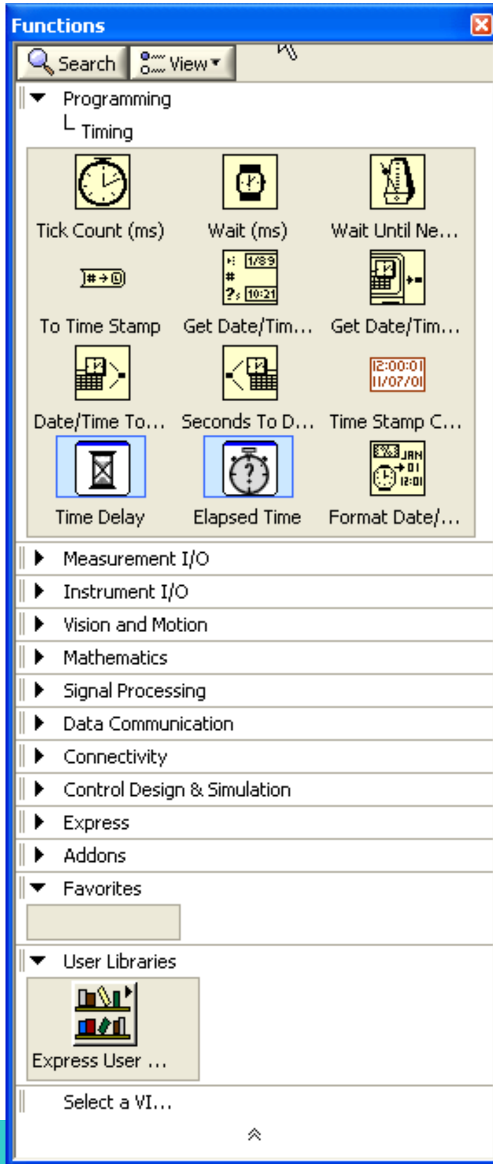
Stacked sequence



Timing Control

- **Functions>>Programming>>Timing**
- Example: wait 1 second before executing next step.

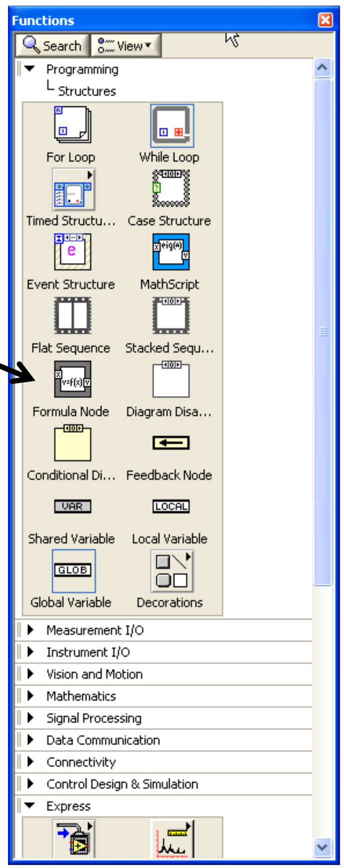


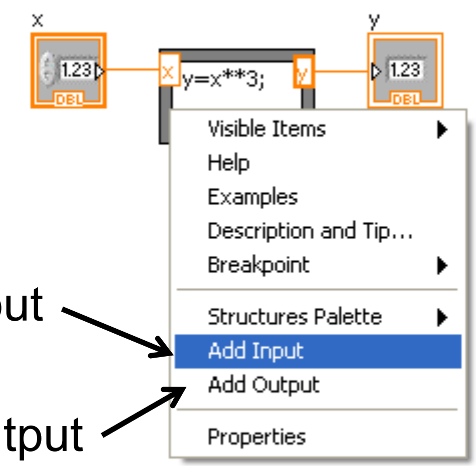


Formula Node

- Allows you to program one or more algebraic formulas.

Formula node





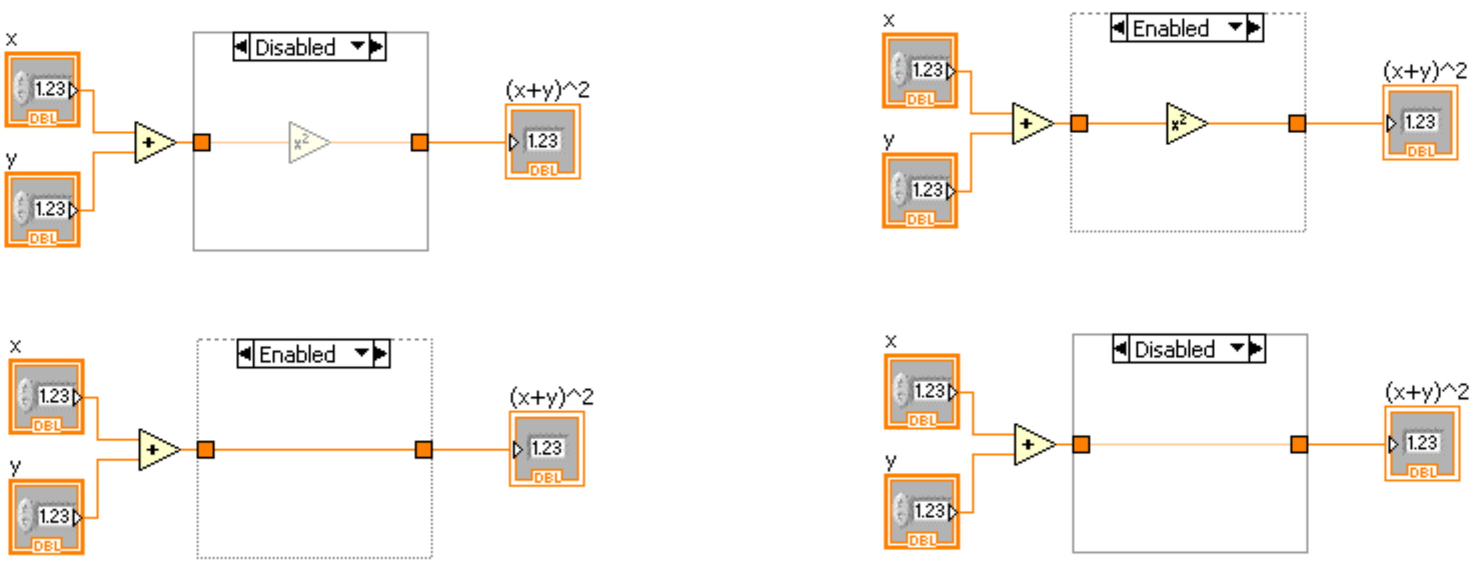
Add input

Add Output

Example: calculate $y=x^3$

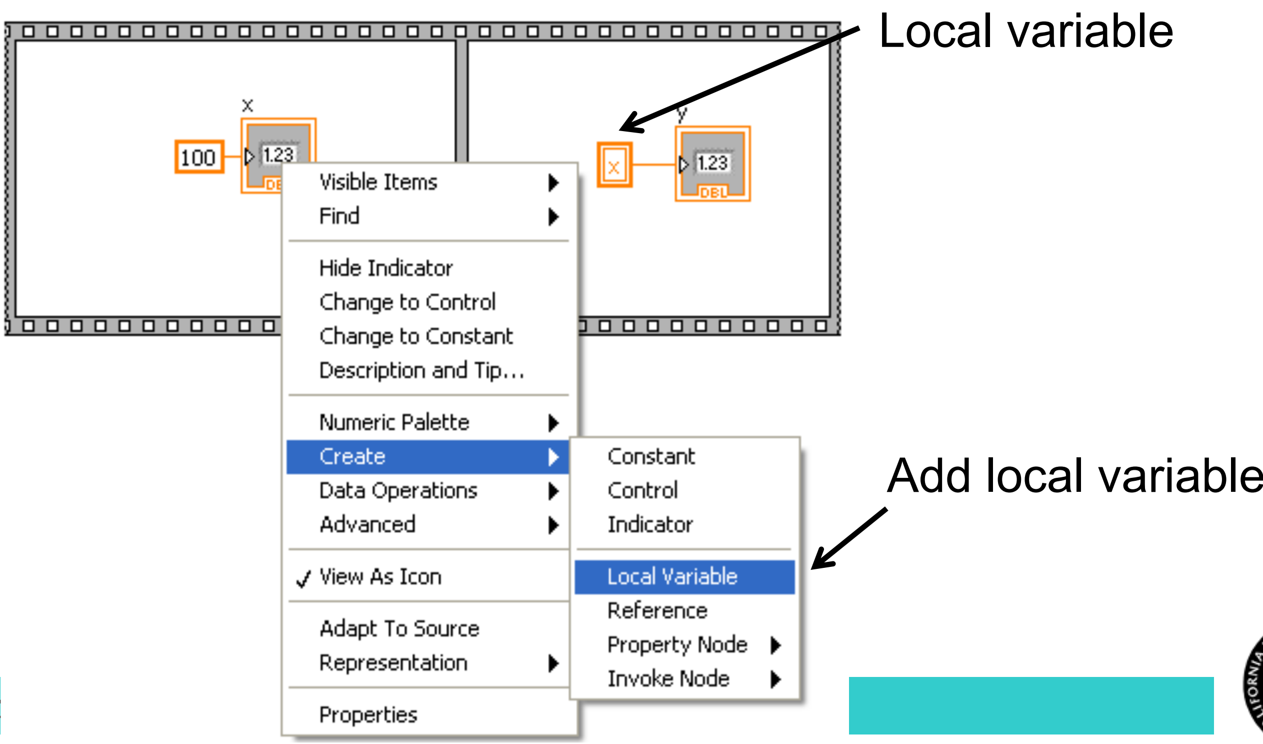
Diagram Disable Structure

- To disable specific sections of code, equivalent to commenting out code in a text-based Programming language.
- The disabled codes can be enabled by select “Enable This Subdiagram”.

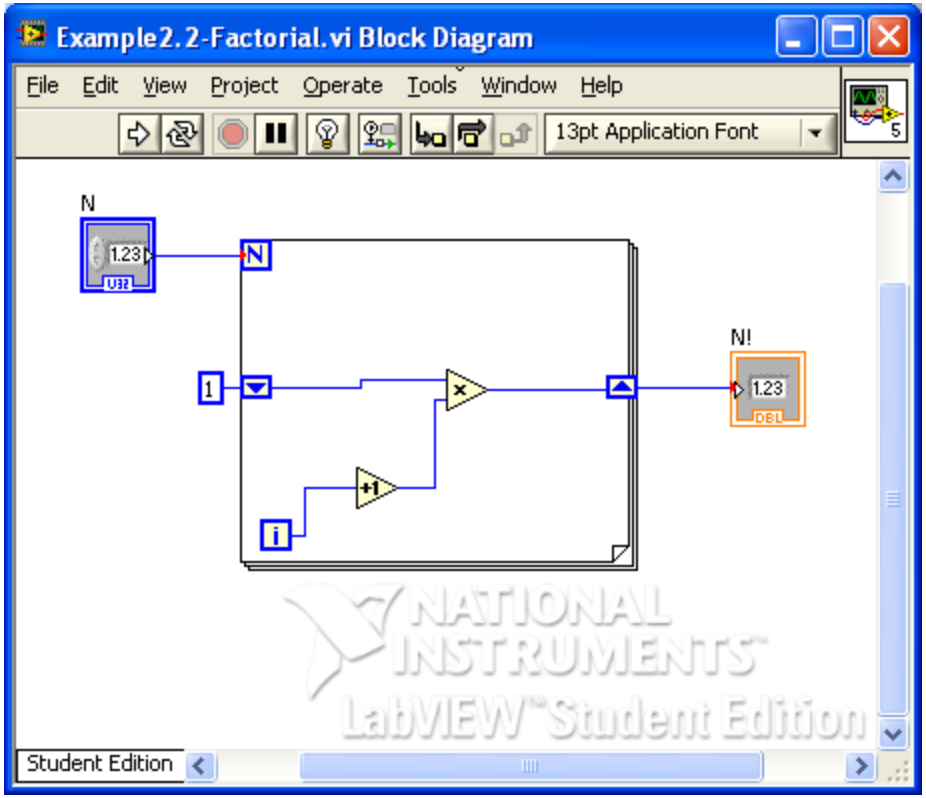


Local Variables

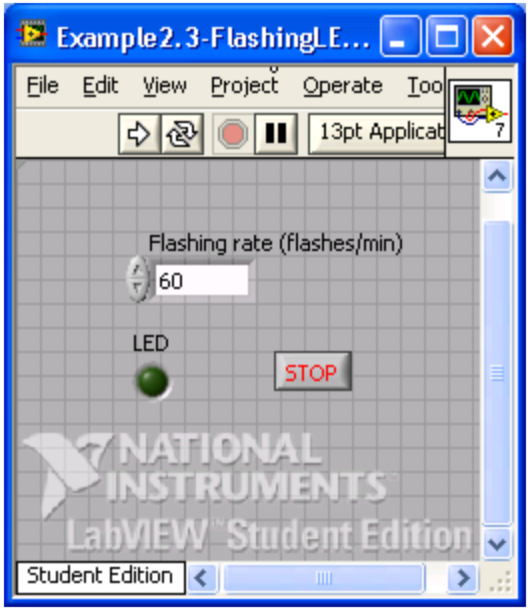
- Each front panel object has only one corresponding block diagram terminal. You can use **local variables** to access (read or write) front panel objects from more than one location in a single VI.



Work Example 3.1 – Calculate the factorial n!



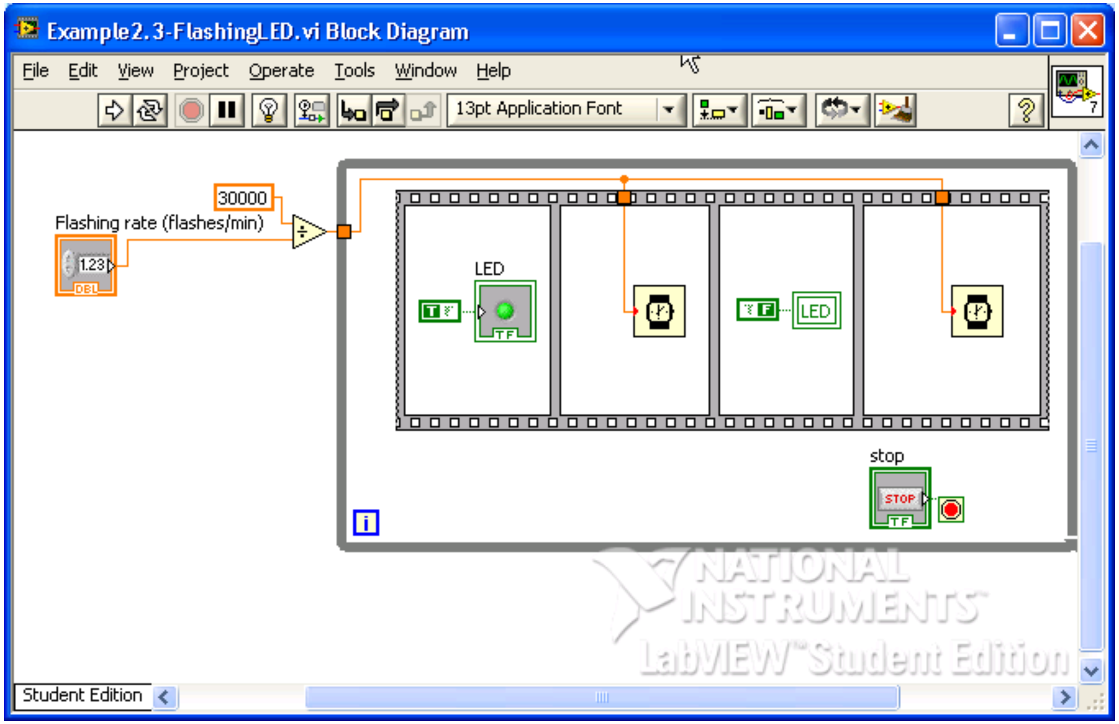
Work Example 3.2 – Flashing LED



Block diagram?

Answer


Example2.3-FlashingLE...
File Edit View Project Operate Too
13pt Applicat
Flashing rate (flashes/min)
60
LED
STOP
NATIONAL INSTRUMENTS
LabVIEW Student Edition
Student Edition


Example2.3-FlashingLED.vi Block Diagram
File Edit View Project Operate Tools Window Help
13pt Application Font
30000
Flashing rate (flashes/min)
1.23
DBL
LED
LED
stop
STOP
NATIONAL INSTRUMENTS
LabVIEW Student Edition
Student Edition

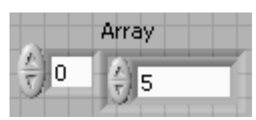
BE/EE189 Design and Construction of Biodevices
Lecture 4

LabVIEW Programming – Arrays, Clusters, Matrix, Chart and Graph

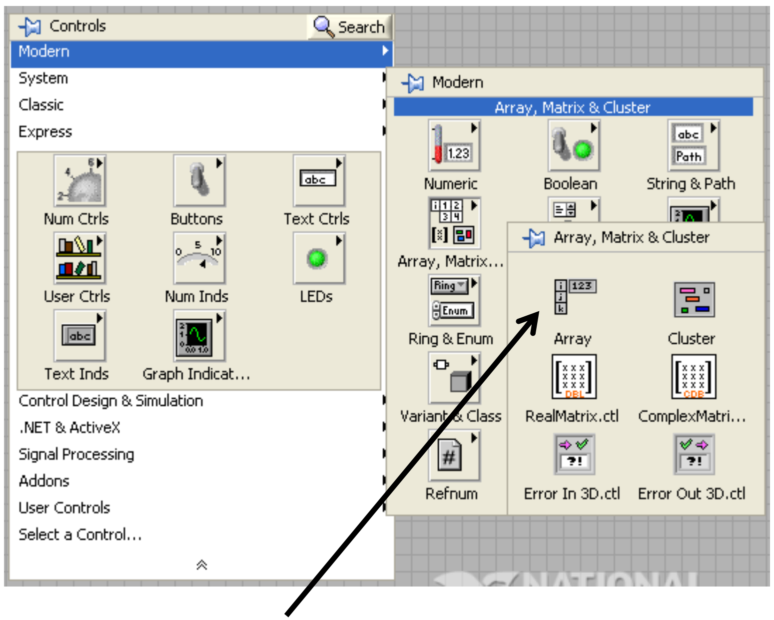
- Arrays
- Polymorphism
- Clusters
- Matrix
- Memory Usage
- Waveform charts
- Wavefrom graphs
- XY graphs
- Math plots

Array

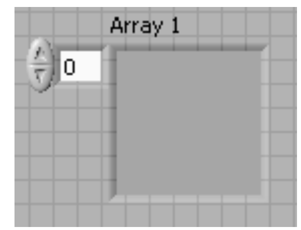
- Array – a *variable-sized* collection of data elements that are all the *same type*.
- Array can have one or more dimensions.
- If memory permitted, each dimension can have up to 2^{31}-1 elements.
- Cannot create an array of arrays, charts, or graphs.
- Can create an array of clusters which has one or more arrays.
- The index is zero-based.



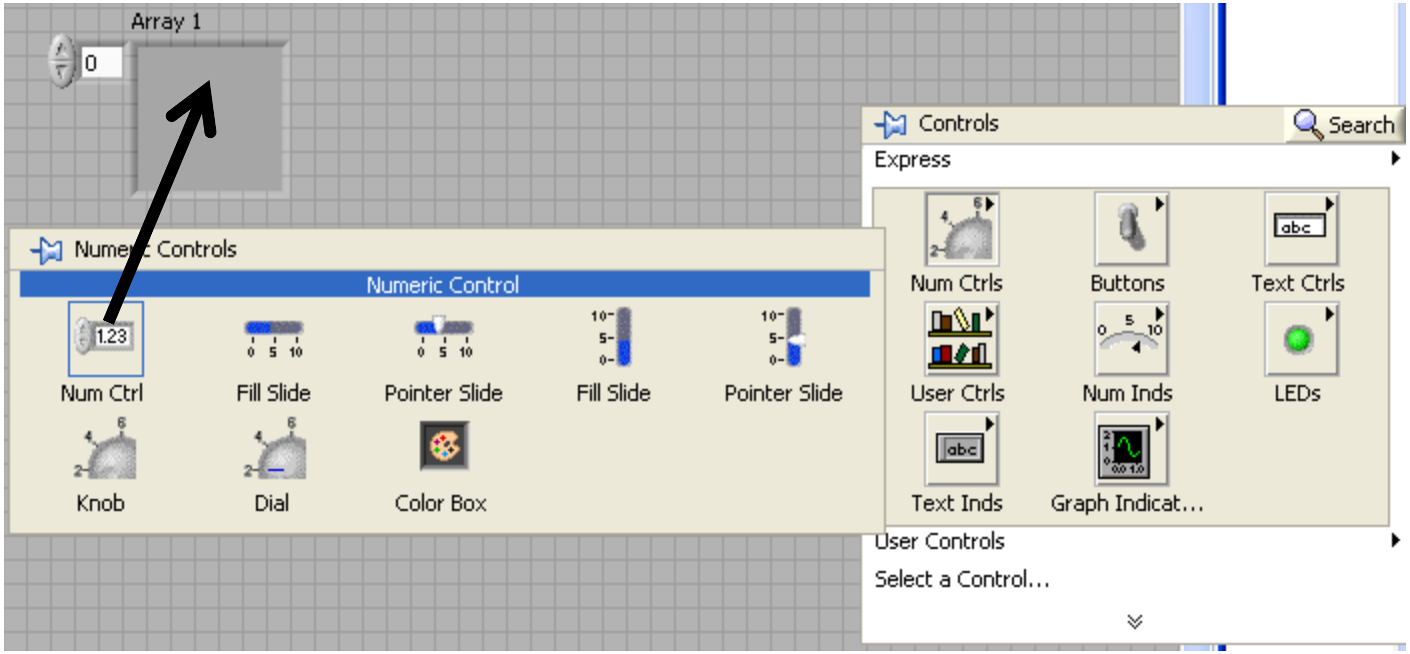
Creating an Array – 1: Add an Array Shell



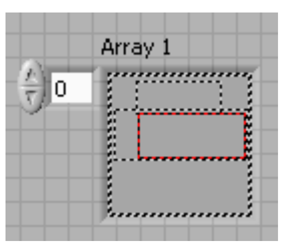
Array control



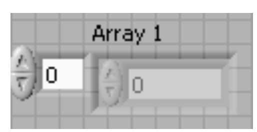
Creating an Array – 2: Place data object into shell



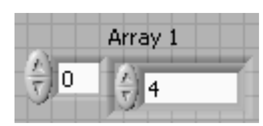
Add object



Becoming..



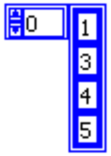
Set value



Creating an Array Constant

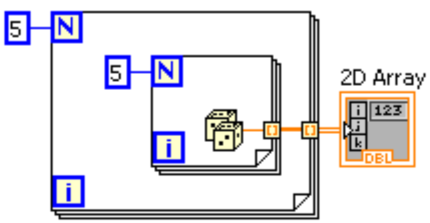
- Similar as previous procedures, except it happens in the block diagram.

An array constant

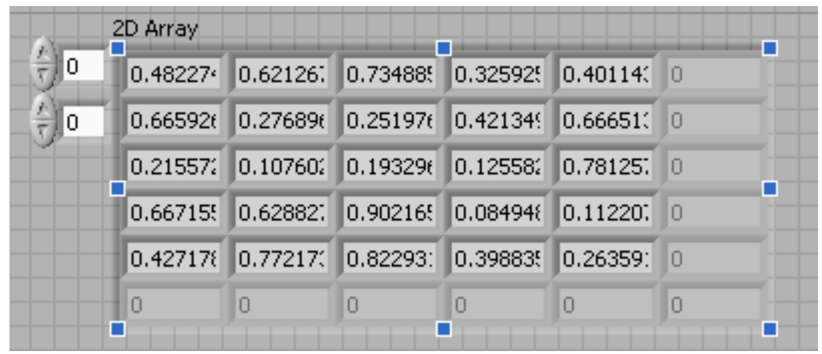


Creating Arrays with Loops

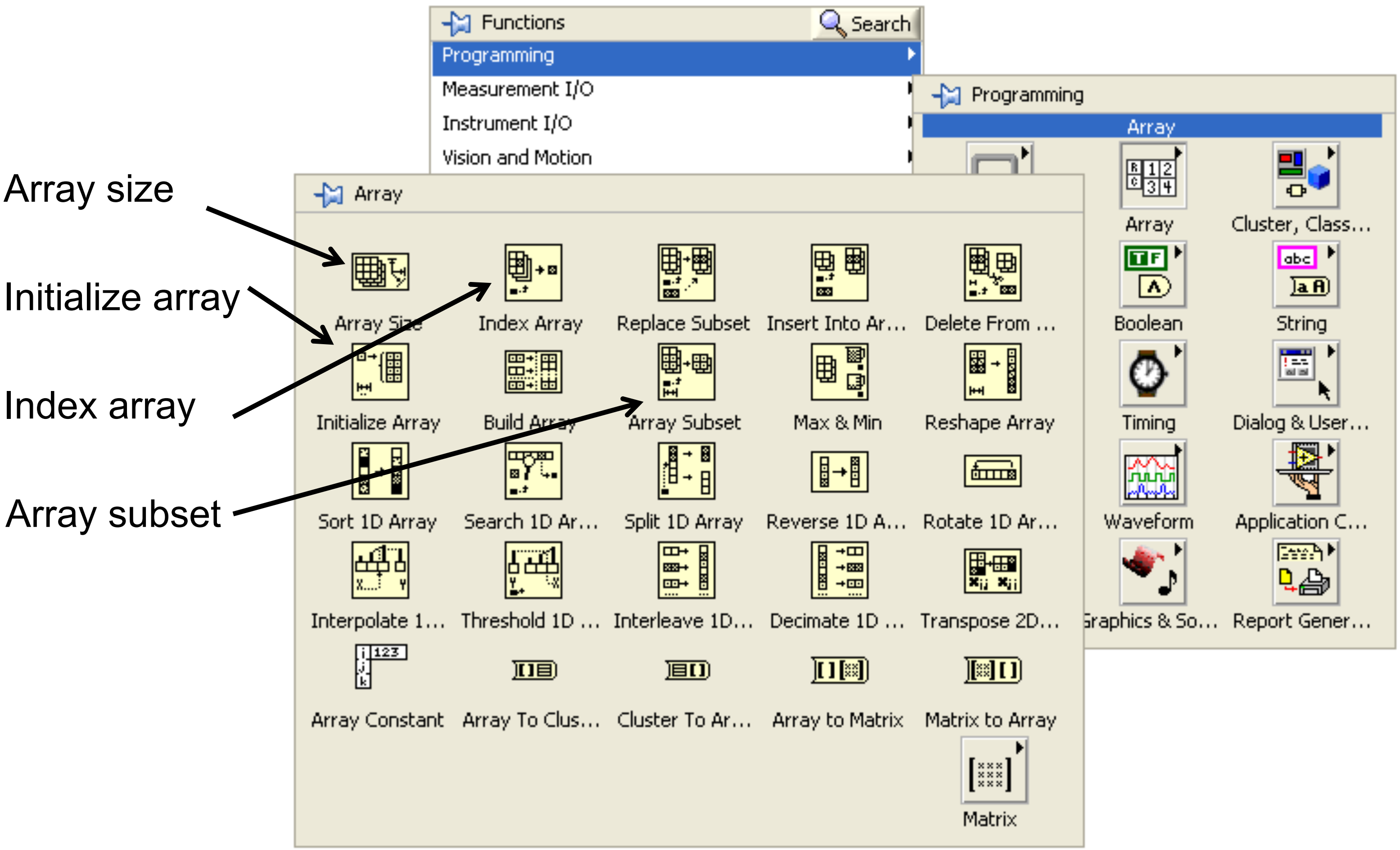
- Example: create a 2D Arrays of random numbers



5x5 Array of random numbers

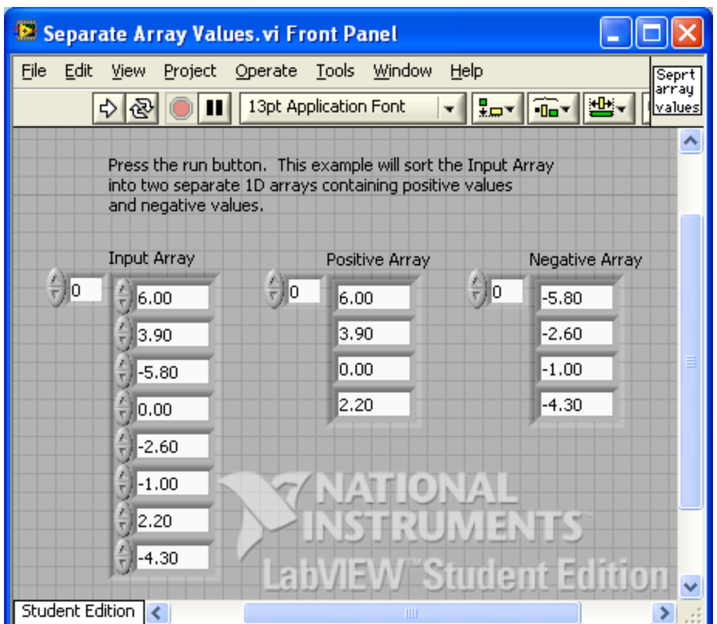


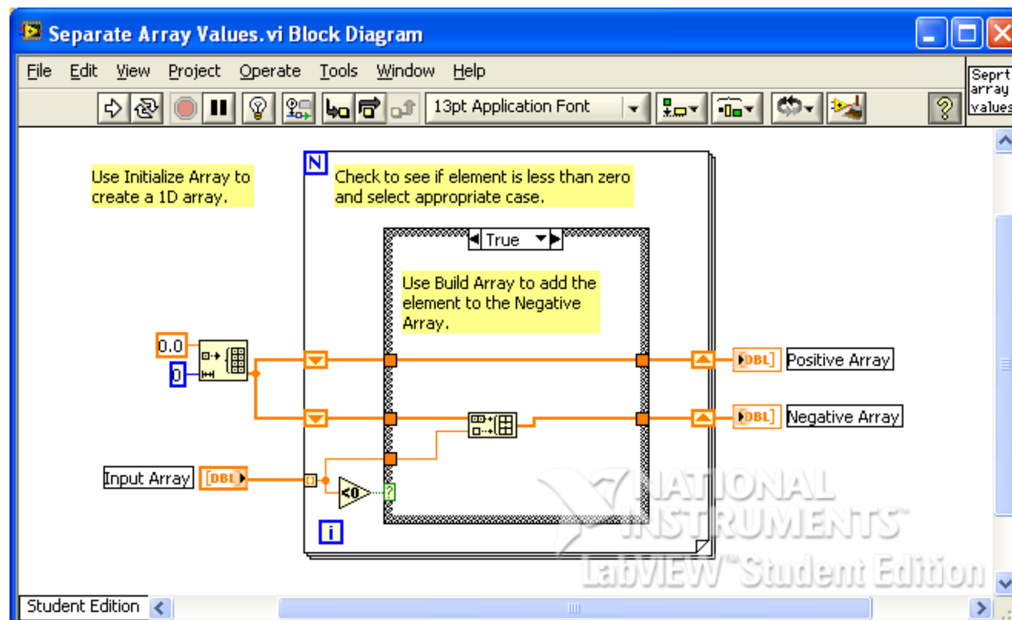
Array Functions



Example – Separate Array Values

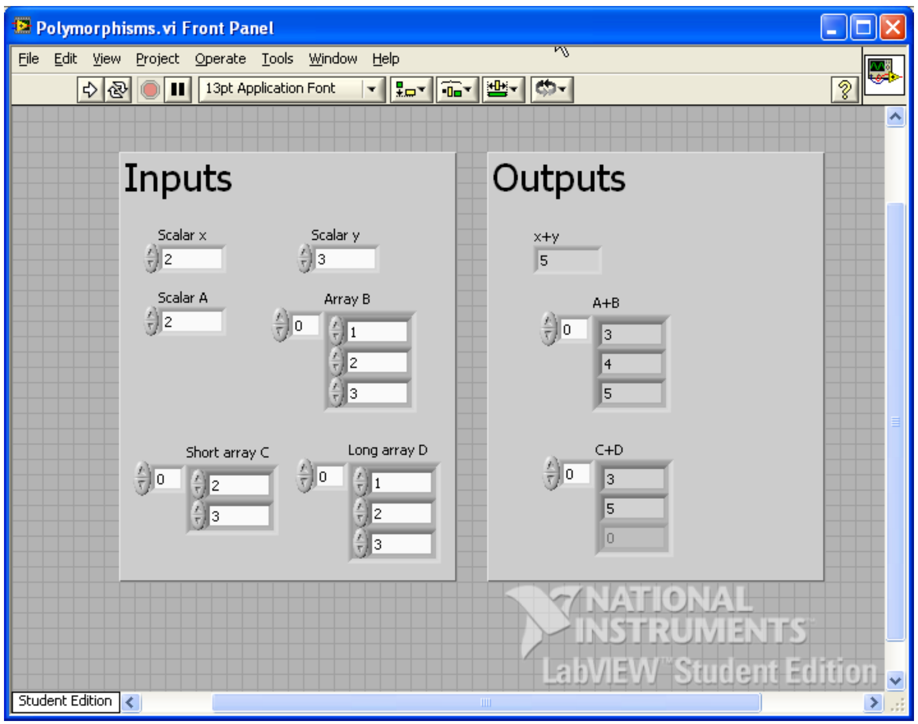
- Takes an input array that contains a mixture of positive and negative values and separates that array into two smaller arrays - one containing just the negative values and one containing just the positive values. (From LabVIEW examples)

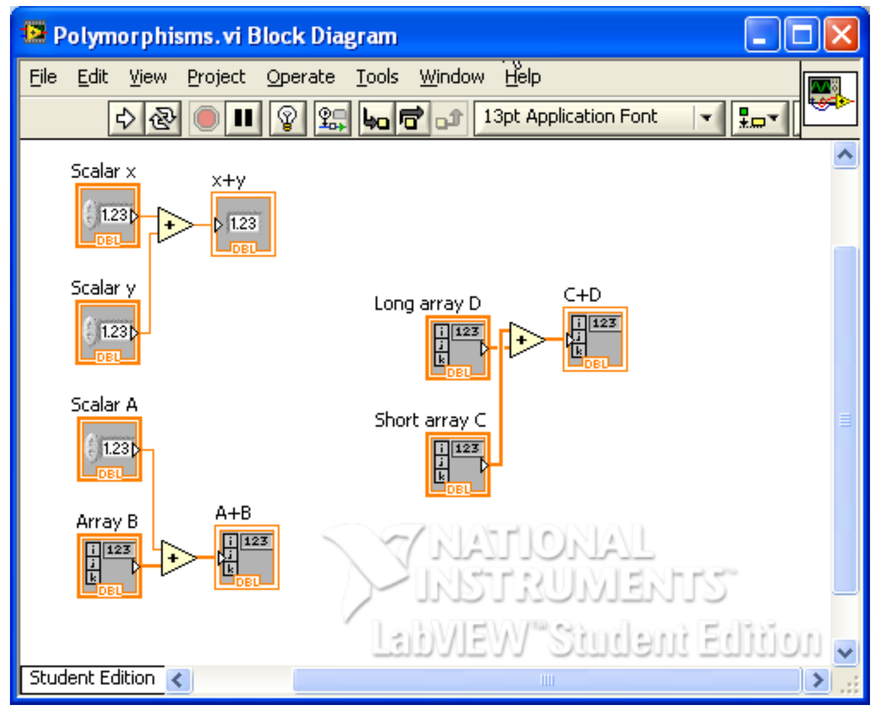




Polymorphism

- Polymorphism is the ability of certain LabVIEW functions to accept inputs of different dimensions and representations.





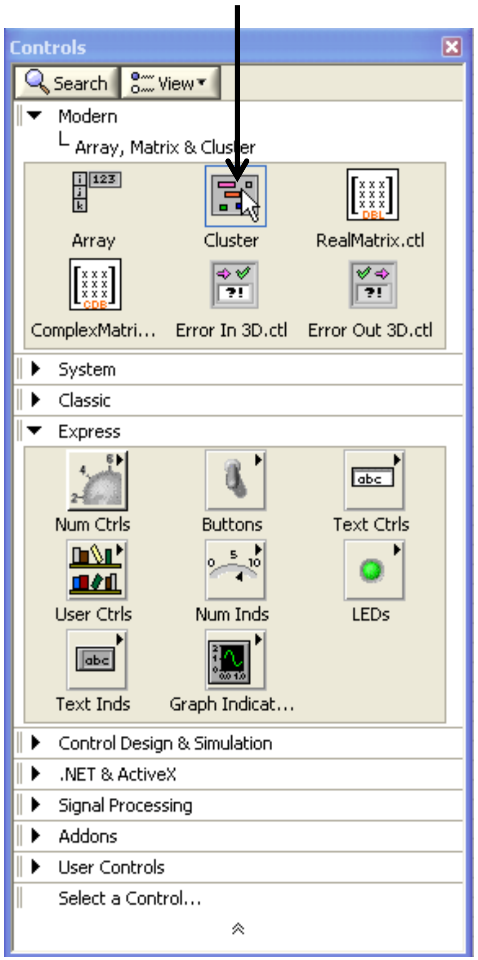
Notice that the sum of a short array and a long array is a short array.

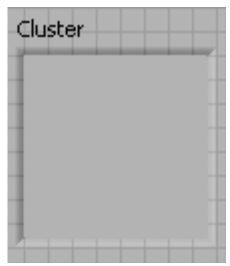
Cluster

- Cluster – a *fixed-sized* collection of data elements of *mixed types*.
- Similar to *struct* in C
- Elements must be either all controls or all indicators

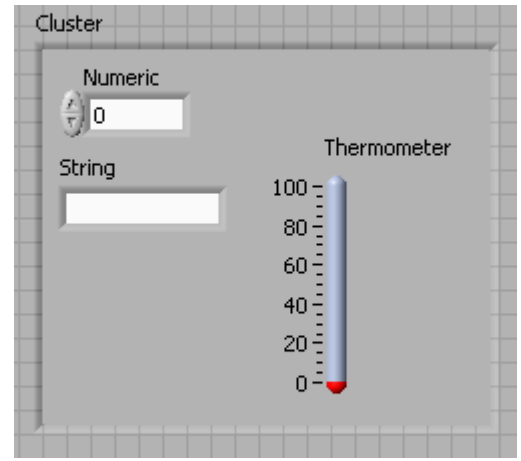
Creating a Cluster

1: Add a cluster shell





2: Placing objects in the cluster shell

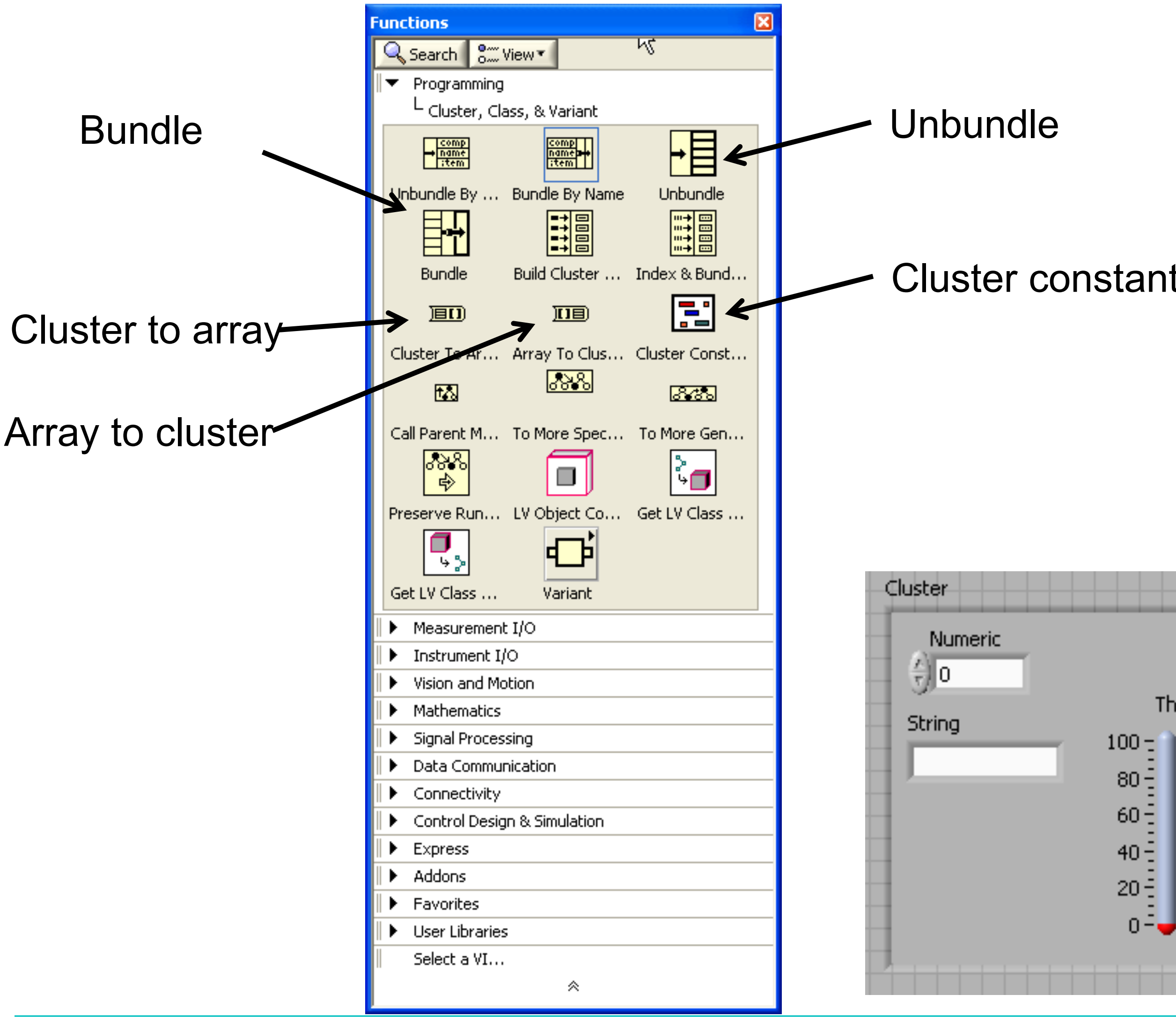


In block diagram

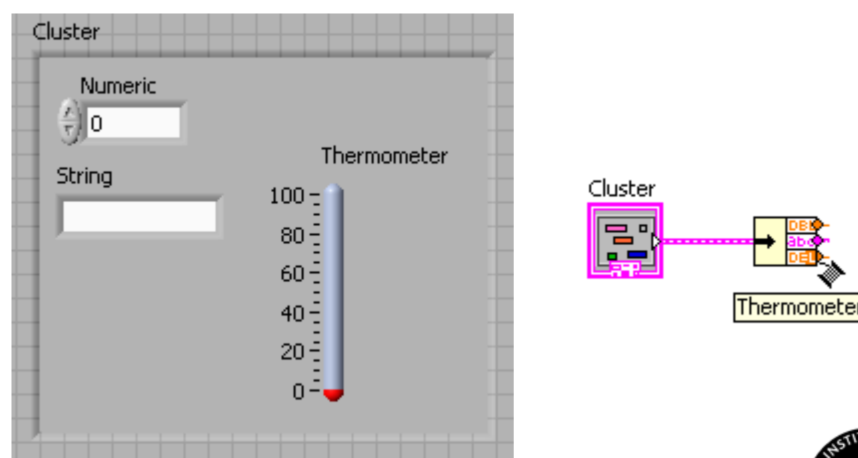
Cluster

- A cluster constant can be created in the block diagram in a similar way.

Cluster Functions



Unbundle example

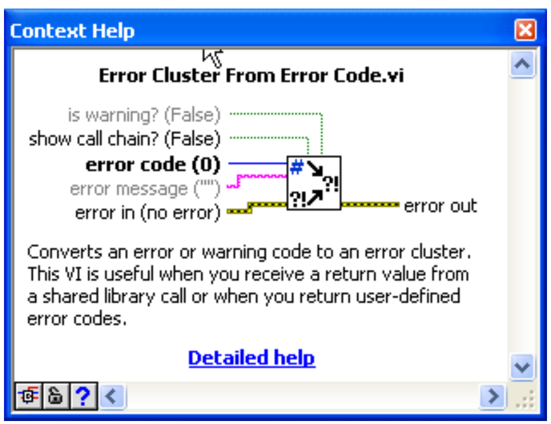


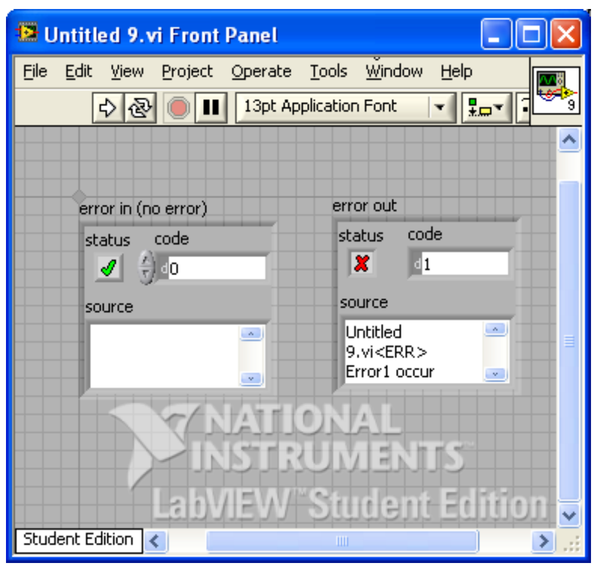
Error Handling

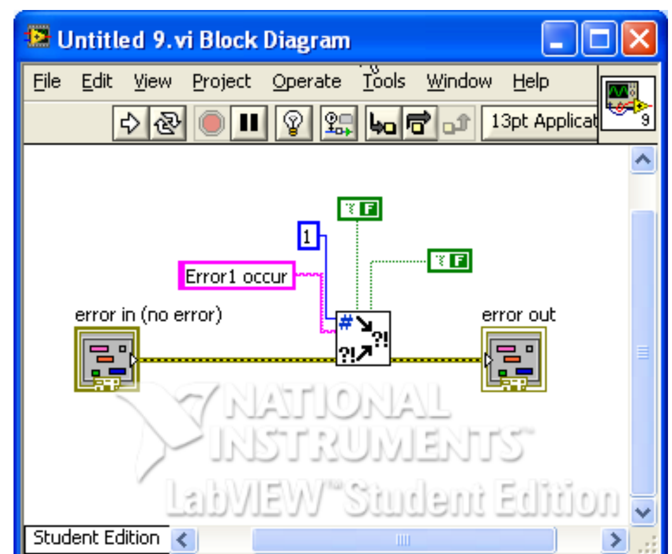
- Automatic error handling – displaying an error dialog box when error occurred. Each error has a numeric code and an error message. Can be disabled in *File >> VI Properties >> Execution*.
- Manual error handling: using error-handling VIs and functions on the **Dialog & User Interface** subpalette (found on the **Programming** palette).

Error Clusters

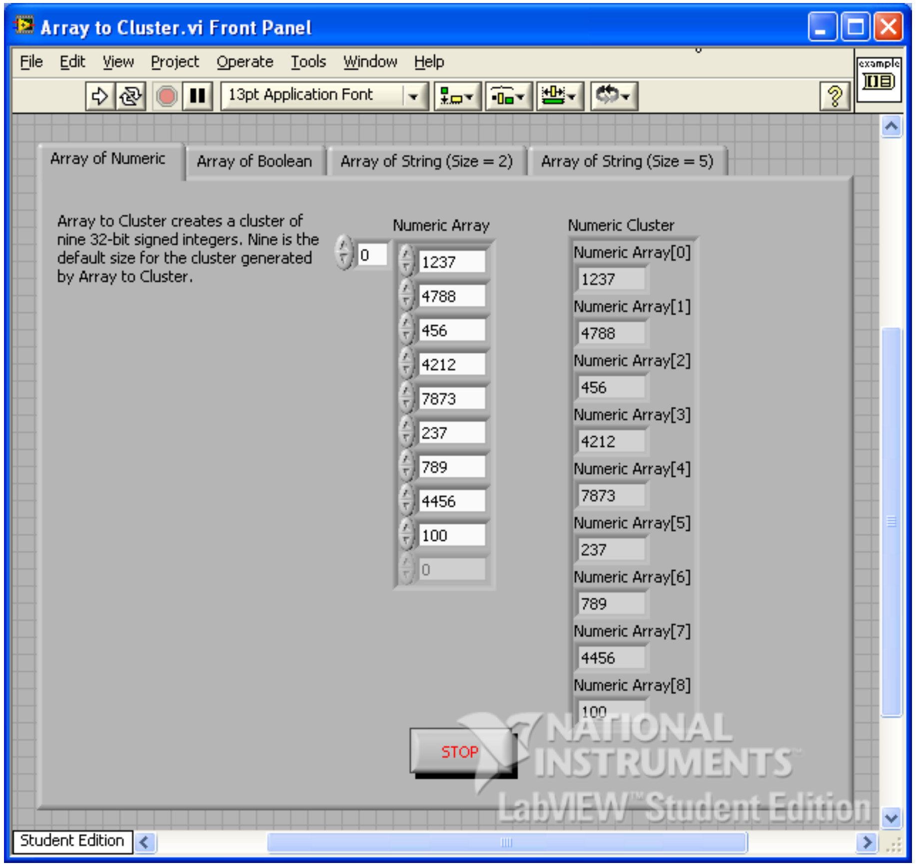
- Error-cluster controls and indicators are used to create error inputs and outputs.
- Programming >> Dialog & User Interface >> Error Cluster From Error Code.

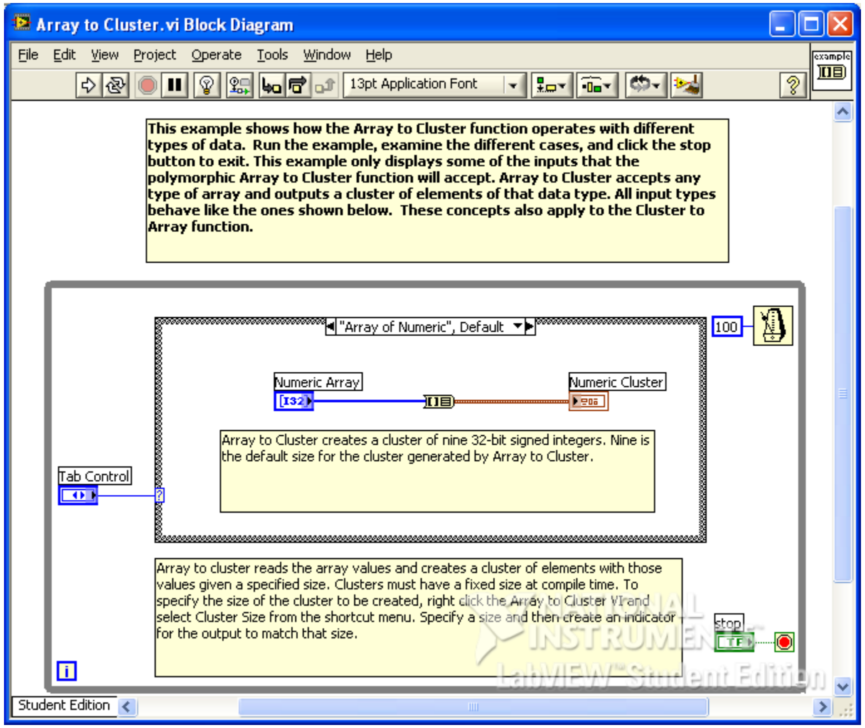






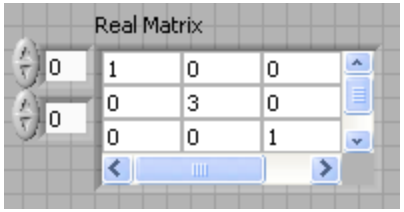
Work Example – Array to Cluster (From LabVIEW Examples)

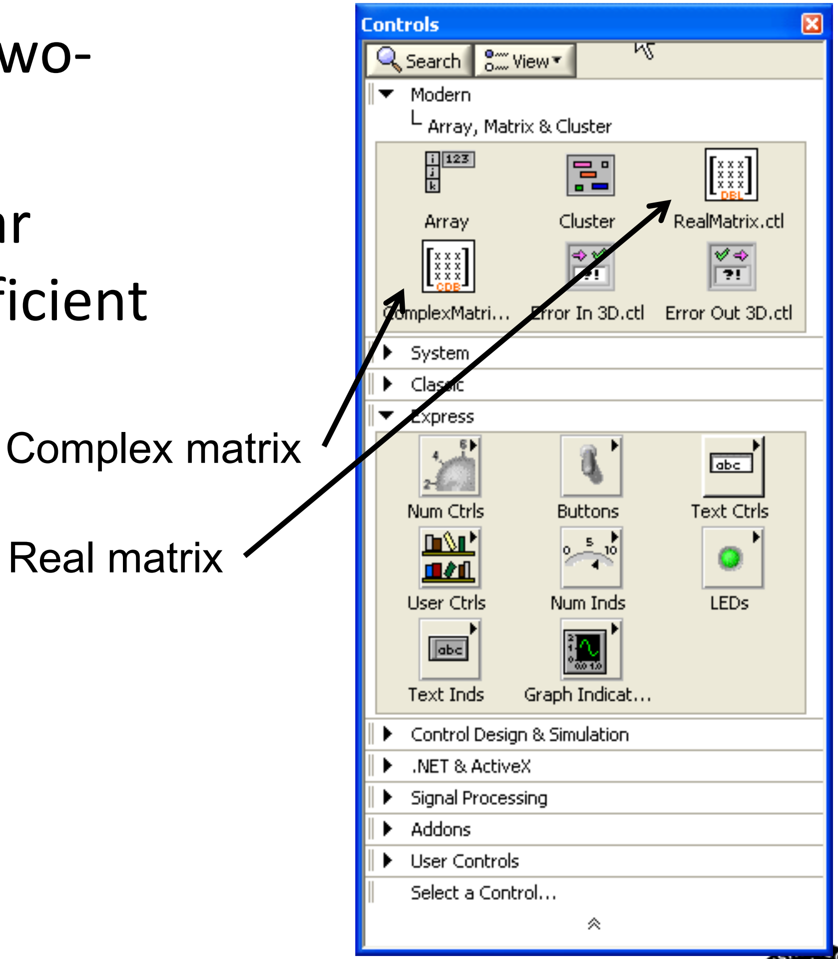




Matrix

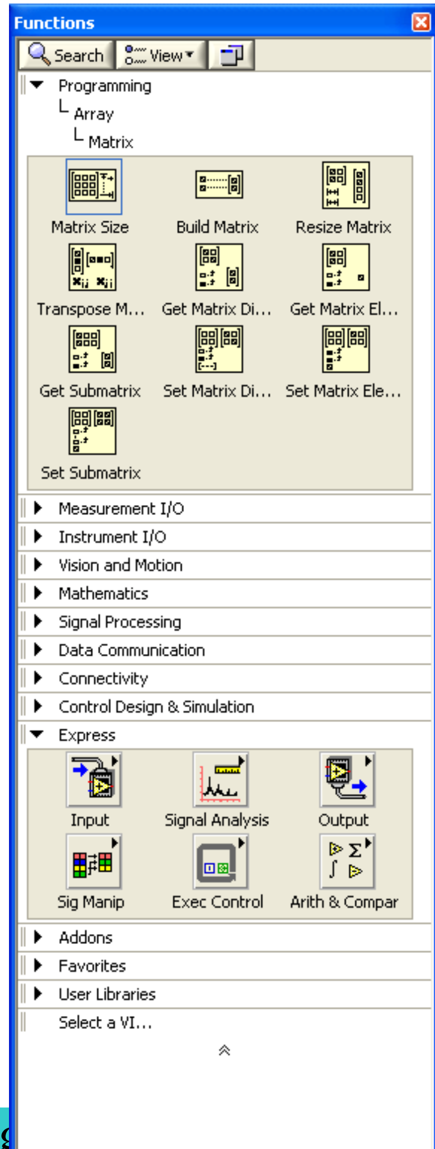
- Matrix can be used in place of two-dimensional arrays.
- Advantages: can use many linear algebra operations based on efficient matrix algorithms.



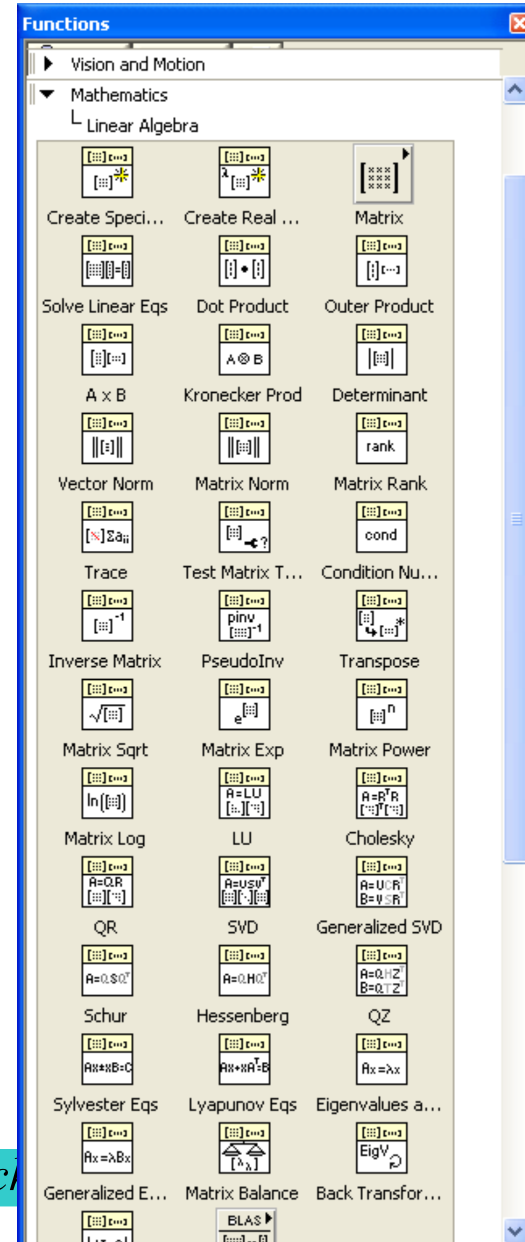


Matrix Functions

Programming
>>Array>>Matrix



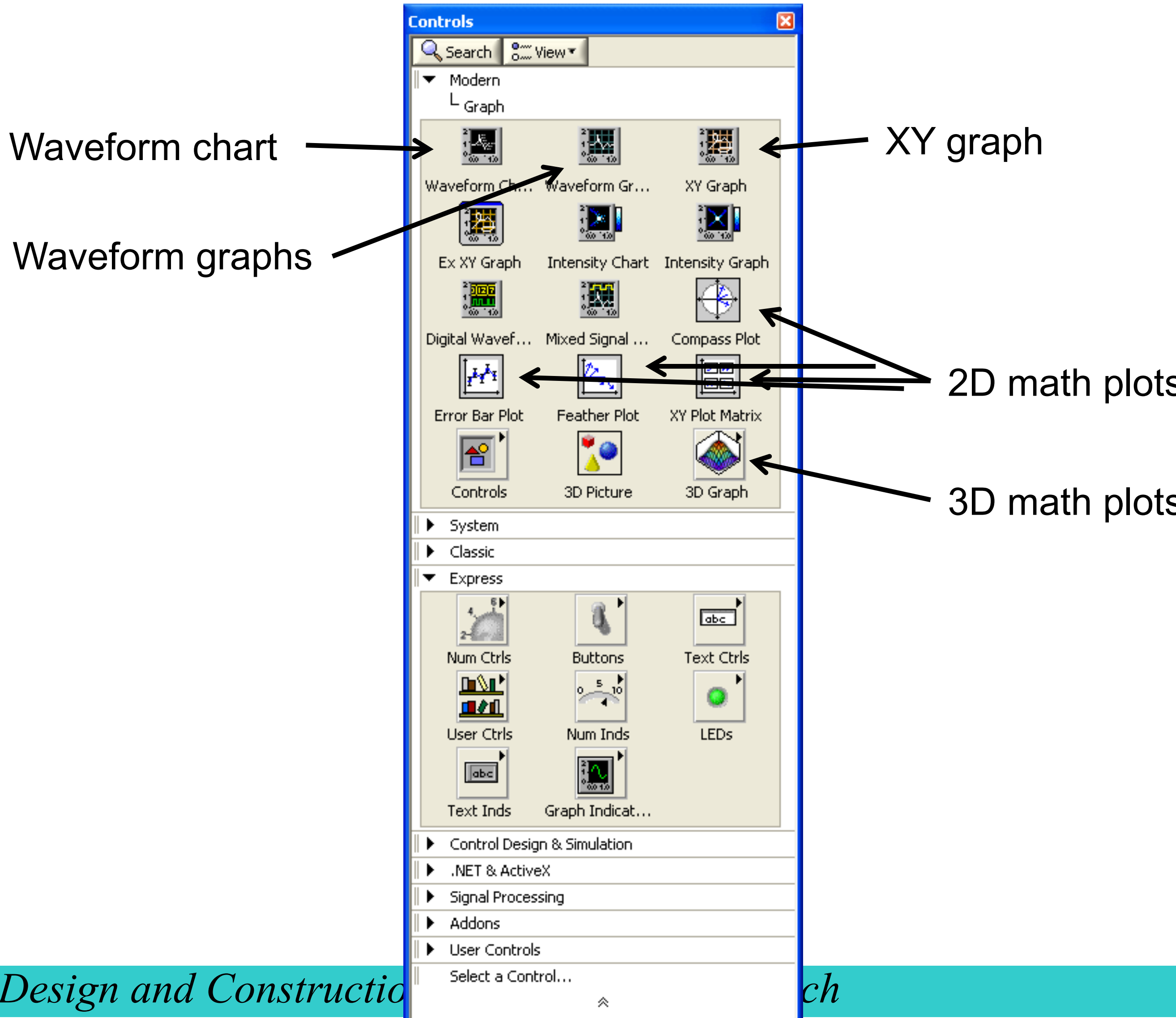
Mathematics>>
Linear Algebra



Memory Usage

- Automatic memory handling – no need to worry when working with small sets of data.
- Some hints for working with large data sets:
  - Initialize large data sets, other than dynamically creating them.
  - Breaking a VI into subVIs.
  - Do not overuse local variables.
  - Unless needed, do not display large arrays and strings on open front panel.
  - Use consistent data types for array.
  - Refrain from using complicated hierarchical data types, such as clusters or arrays of clusters containing large arrays or strings.
  - Unless needed, do not use transparent and overlapped front panel objects.

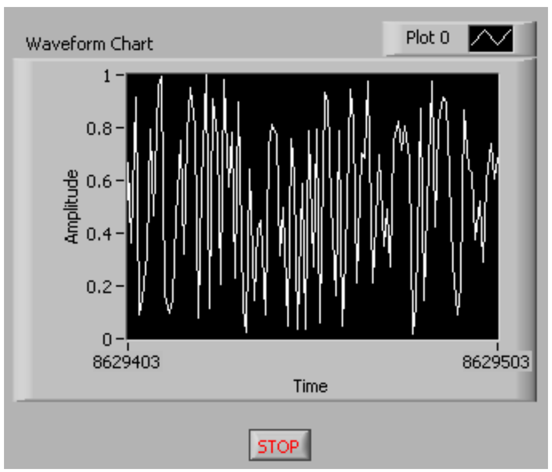
Charts and Graphs



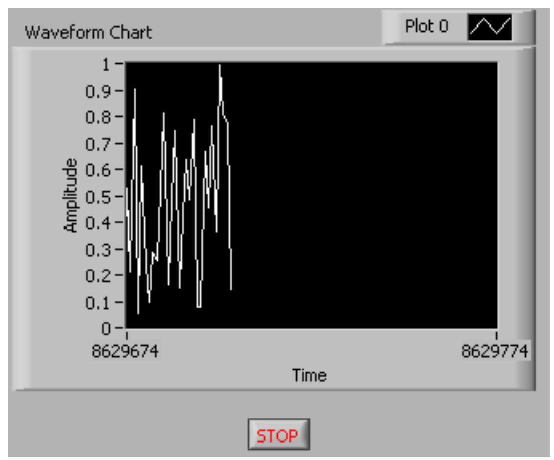
Waveform charts

- Plot new data as they become available.
- Three update modes: **strip chart**, **scope chart**, and **sweep chart** .

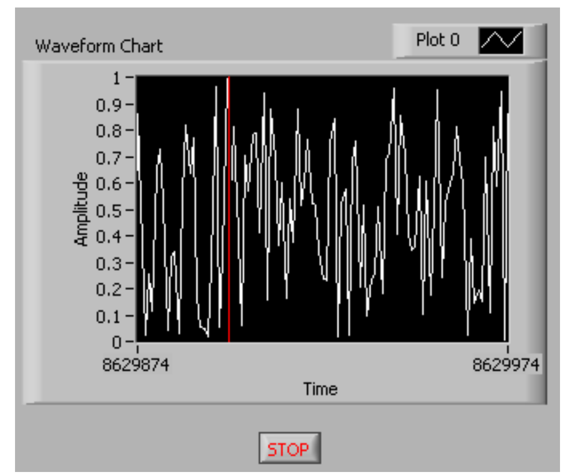
Strip chart

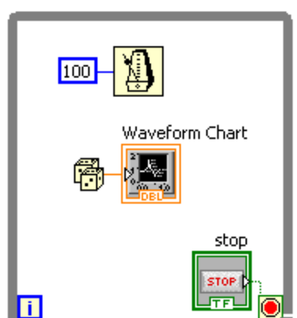


Scope chart

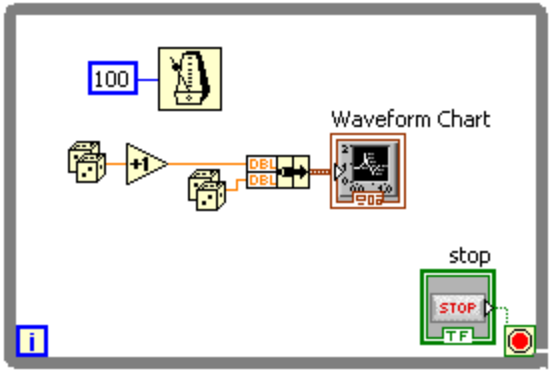


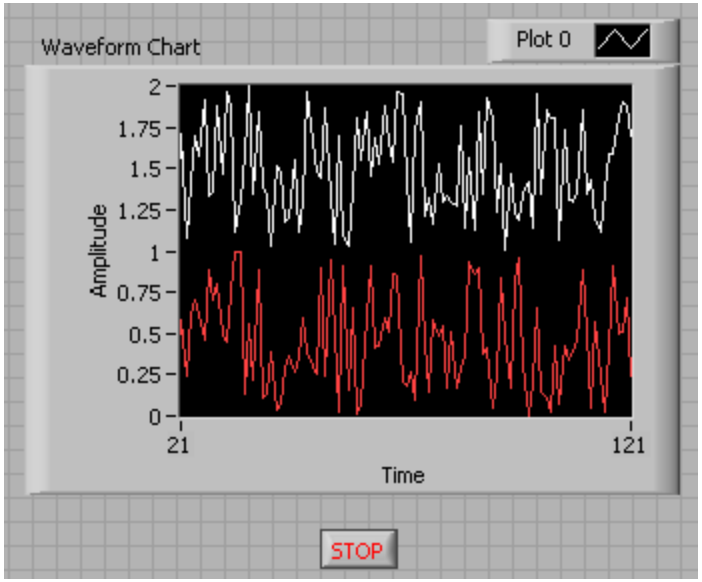
Sweep chart





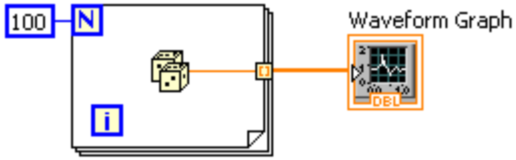
Multi-plot Waveform Charts

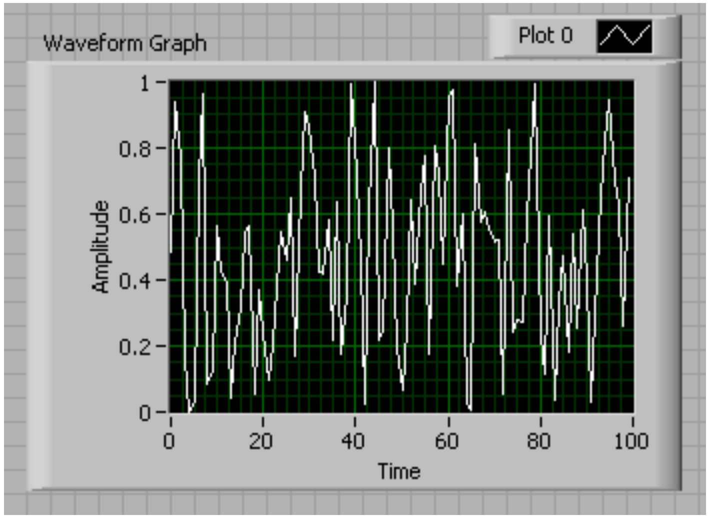




Waveform Graphs

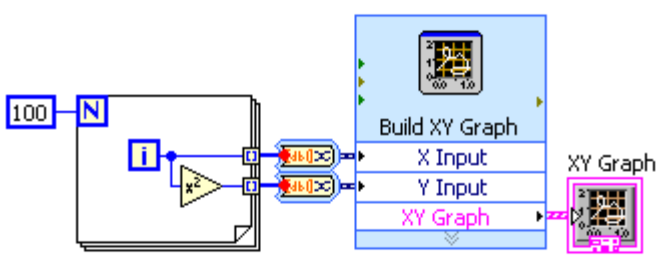
- Plot existing arrays of data all at once – different from waveform charts.

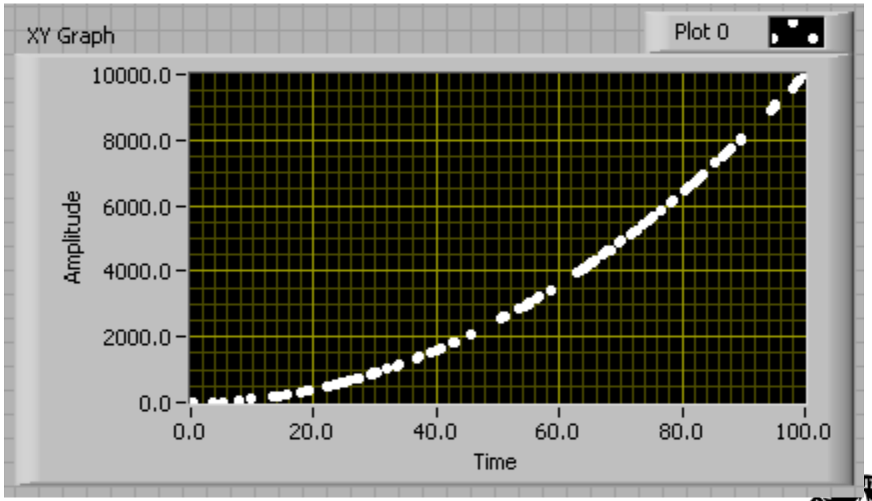




XY Graphs

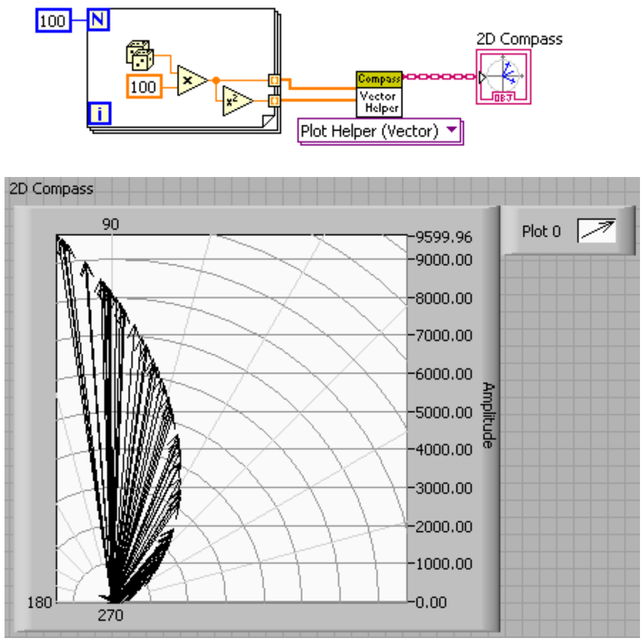
- Waveform graph is ideal for plotting evenly sampled waveforms.
- XY graph is more suitable for situations where you want to specify points using their (x, y) coordinates.
- Controls>>Express>>Graphs Indicators



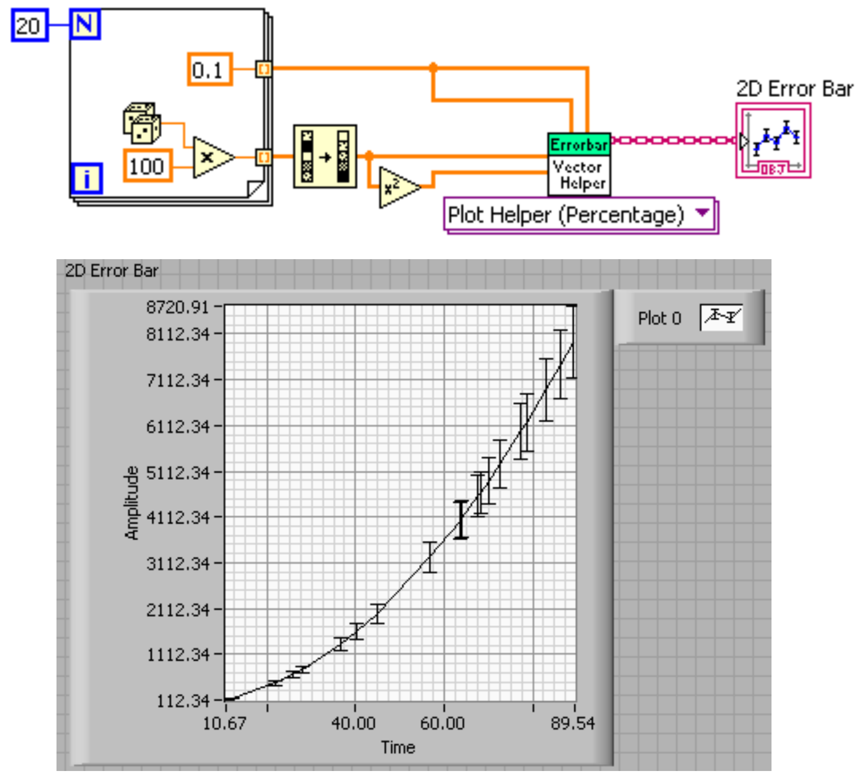


2D Math Plots

2D Compass



2D Error Bar

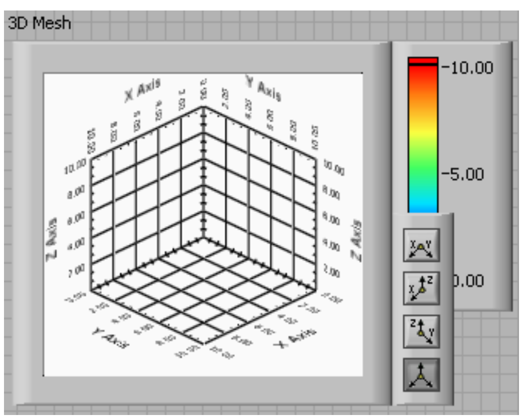


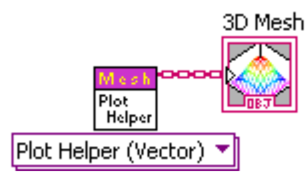
Others: 2D Feather and XY Plot Matrix

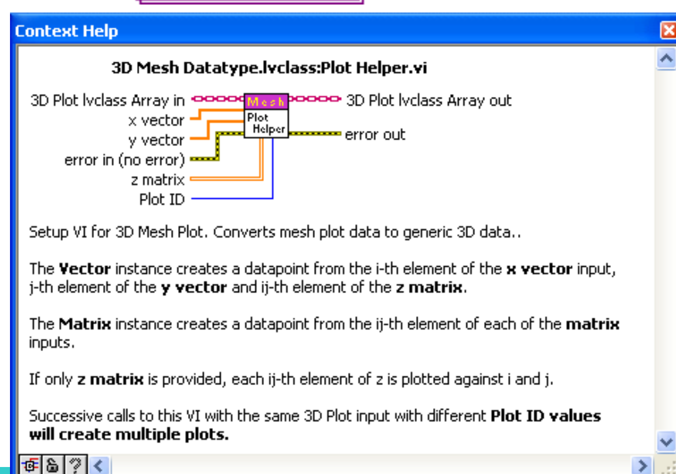
3D Math Plots

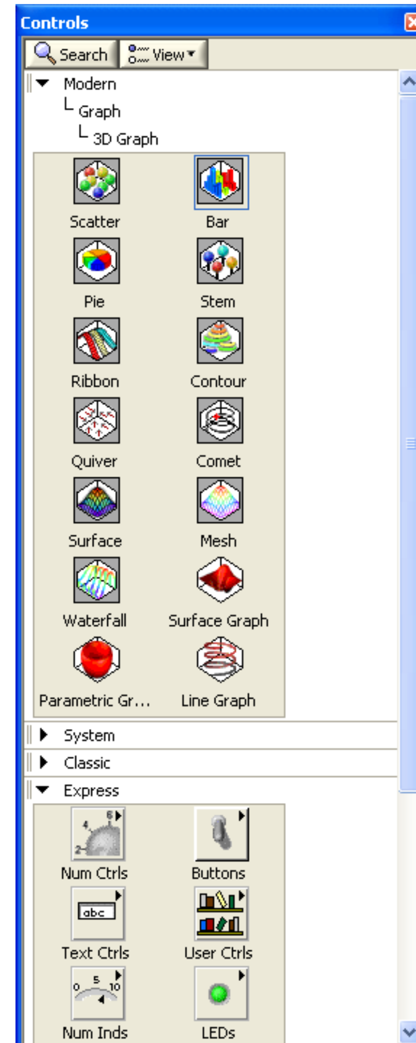
- To visualize data in three dimensions.
- Eleven types are available: Bar, Comet, Stem, Pie, Scatter, Surface, Contour, Mesh, Waterfall, Quiver, and Ribbon.

Example



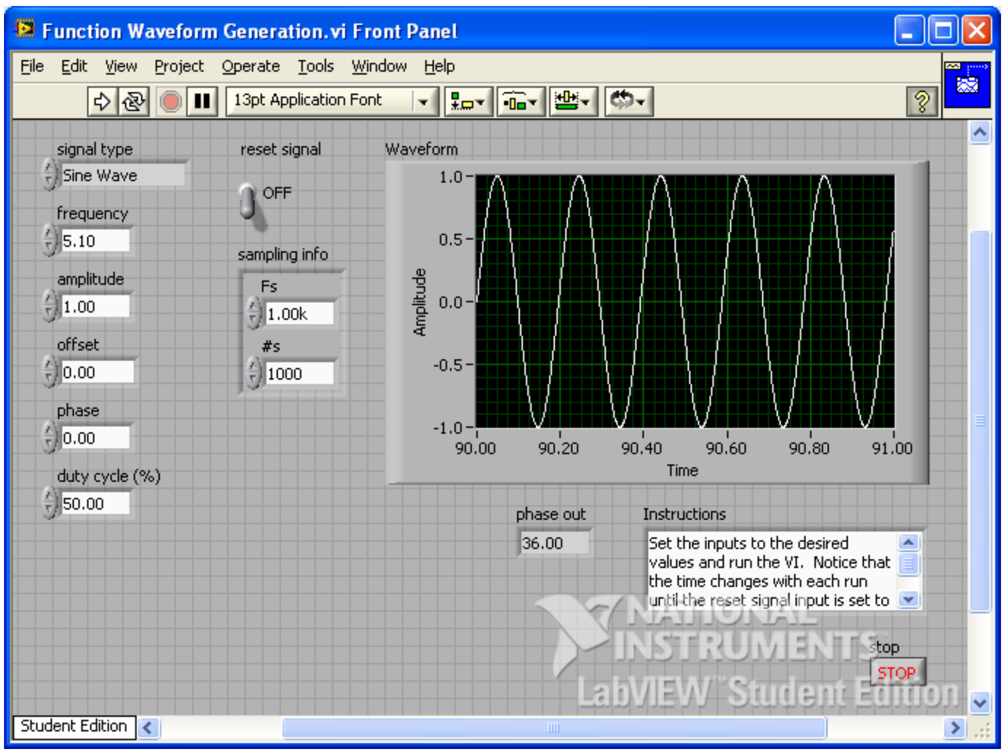


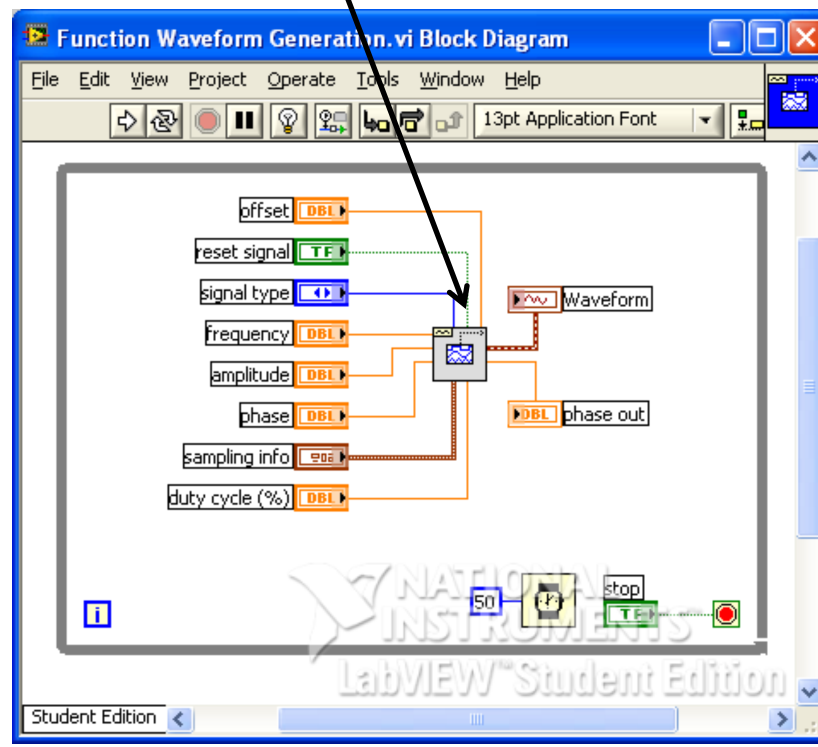




Example – Function Waveform Generation

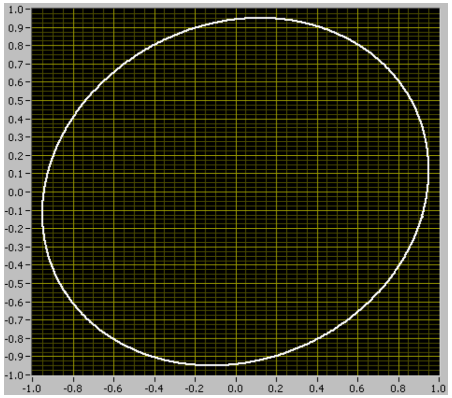
Functions palette>>Signal processing>>Waveform generation>>Basic function generator



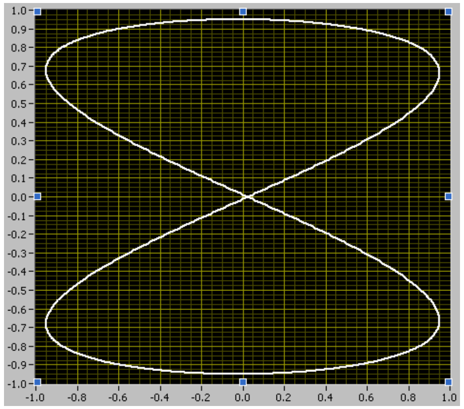


Example – Lissajous Curve

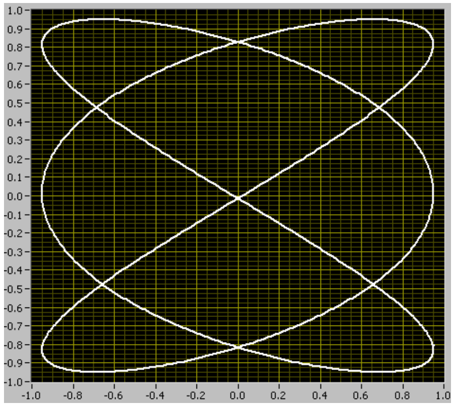
- Lissajous curve is an XY graph of $x = A\sin(at+\delta), \quad y = B\sin(bt),$
  The appearance of the figure is very sensitive to the ratio *a/b*
- Examples



a=b



a=2b



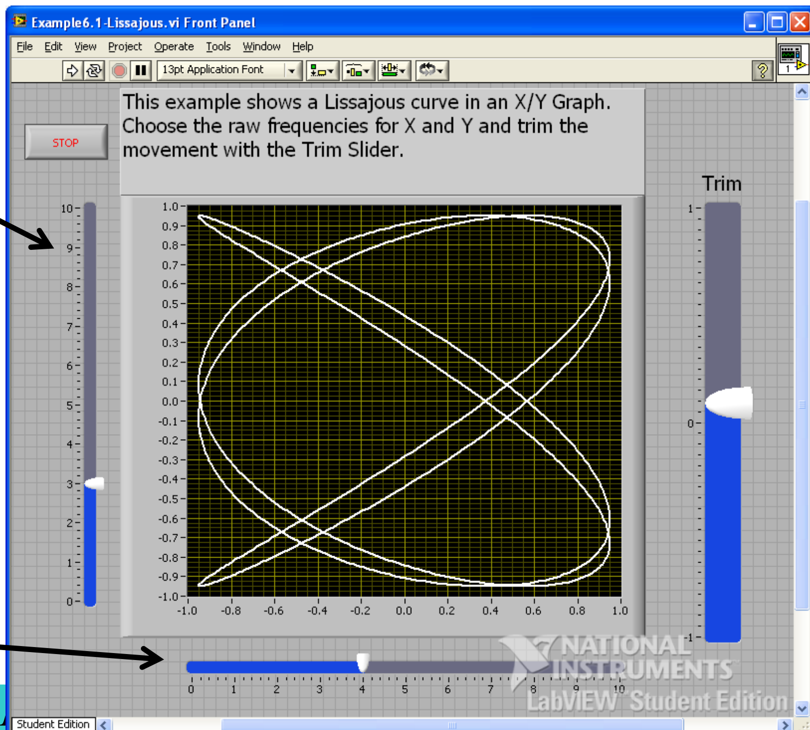
a/b = 3/2

Example – Lissajous Curve

- In this example, we change the frequency of X and Y signal and plot the Lissajous curve. The trim value is used to slightly adjust the frequency of X signal to see the movement of the curve.

Y slider

X slider

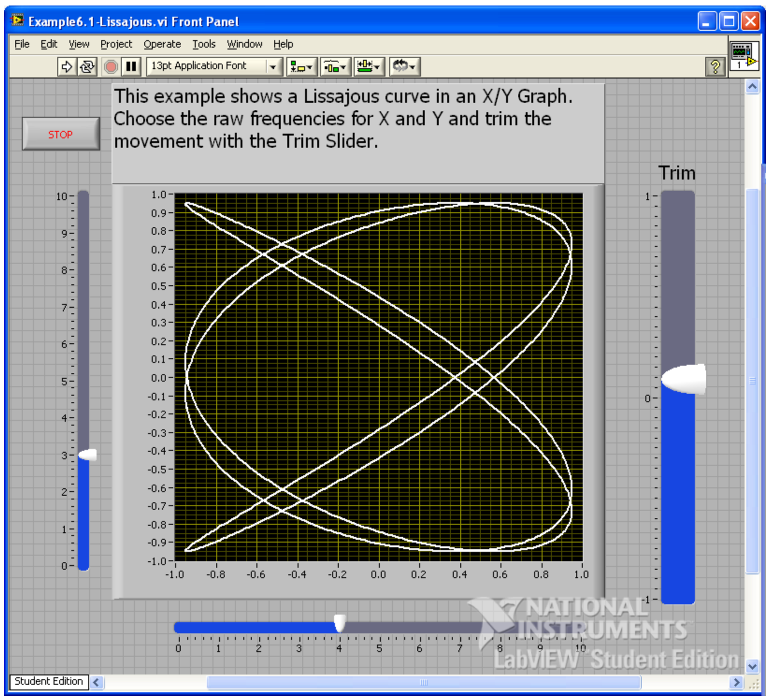


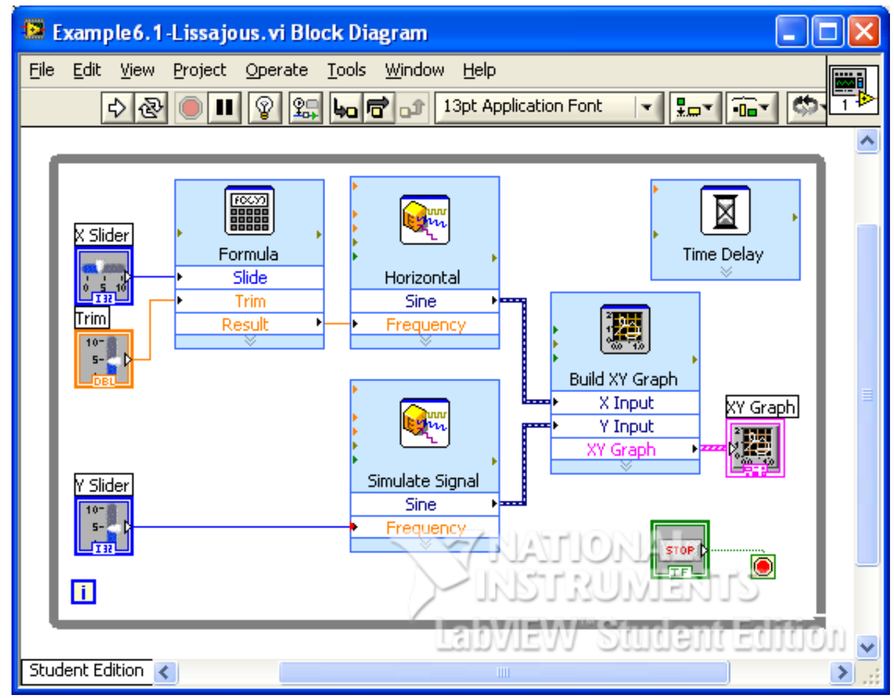
X frequency = X slider+Trim/200

Y frequency = Y slider

Functions palette>>Express>>Signal analysis>>Simulate signal

Example – Lissajous Curve (From LabVIEW Examples)





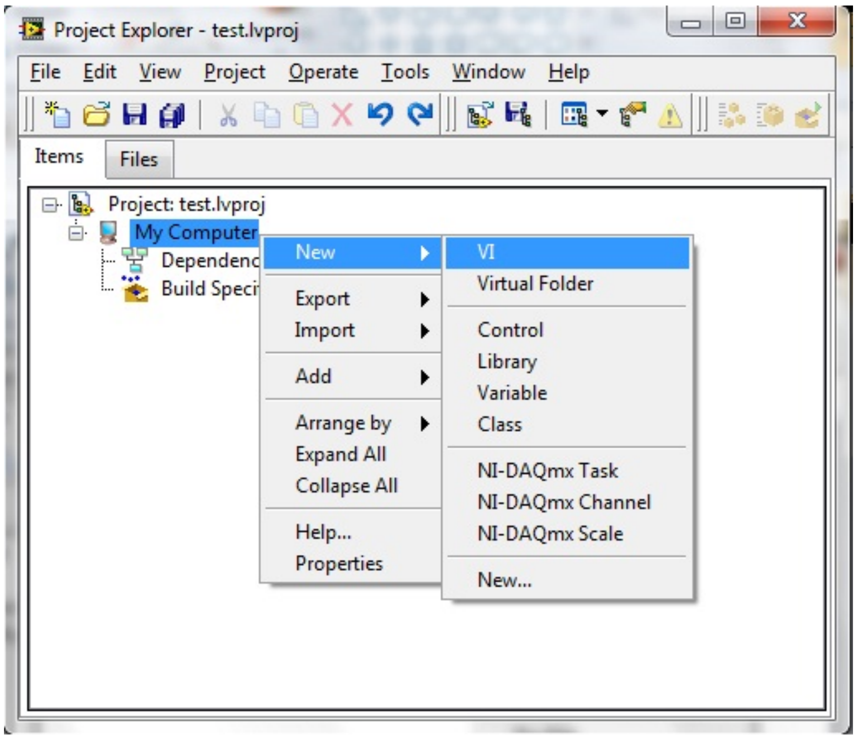
Stand-alone Application

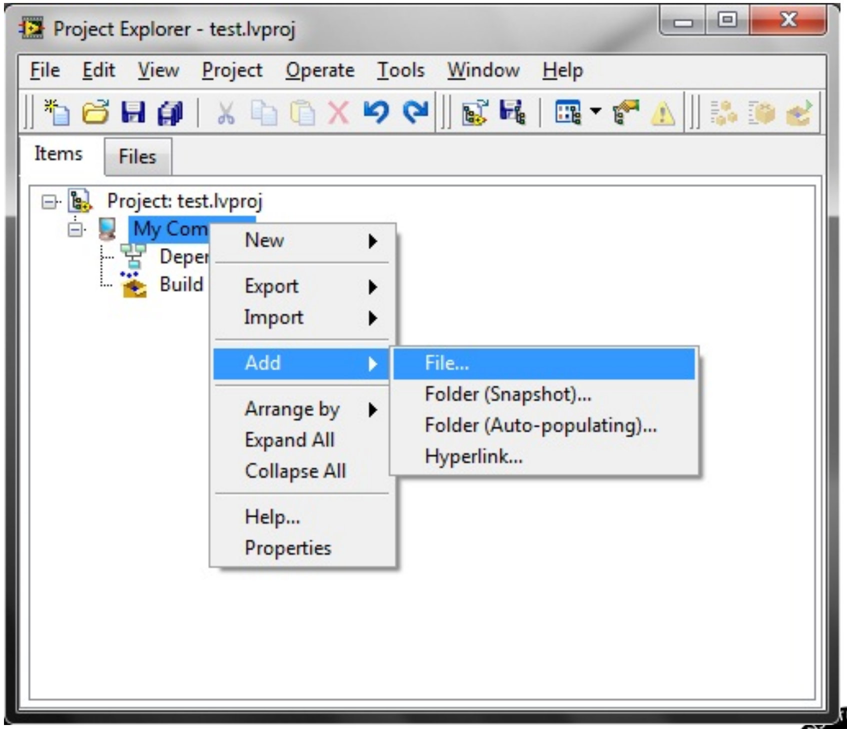
- You can use the Application Builder to build a stand-alone application.
- You need to use a LabVIEW project for the building.

New empty project

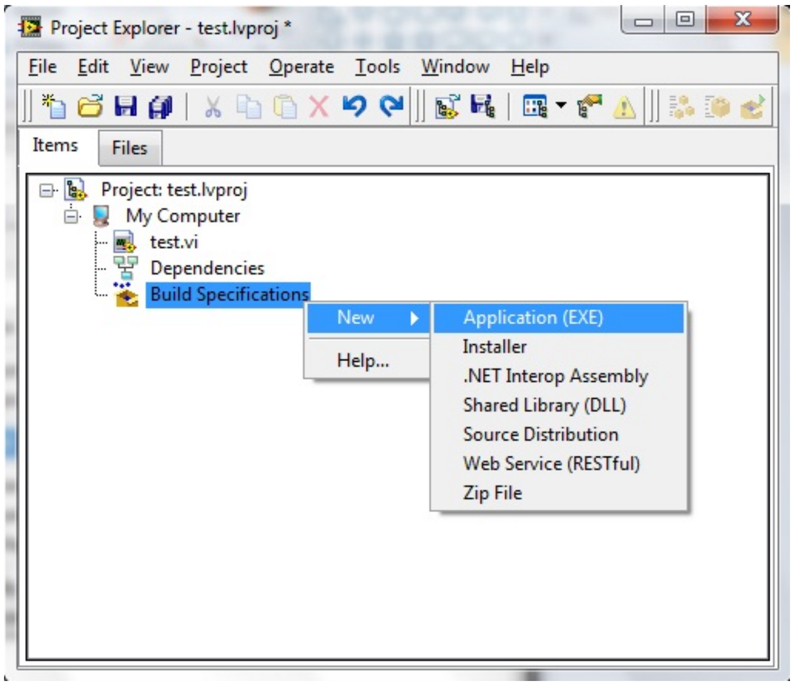
Stand-alone Application – Add VI to Project

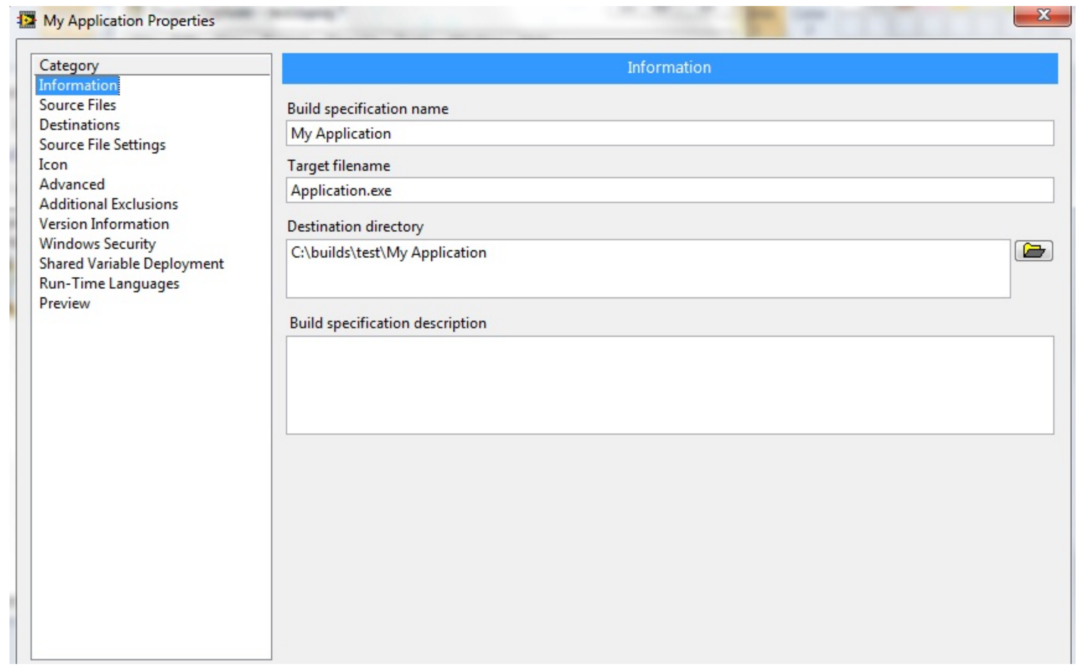
- You can either create a new VI or add existing VI to the project.





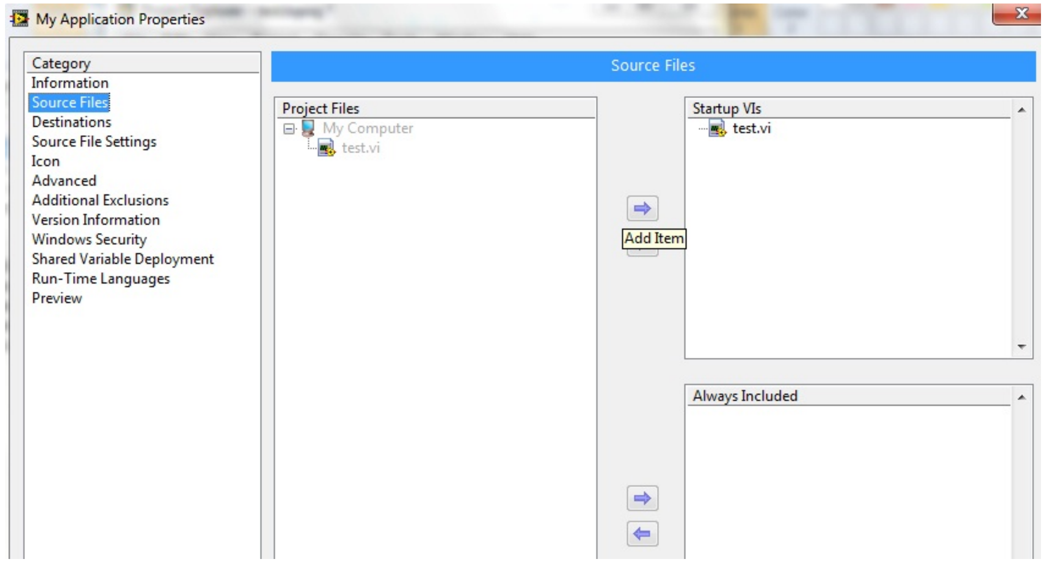
Stand-alone Application – Create an Application



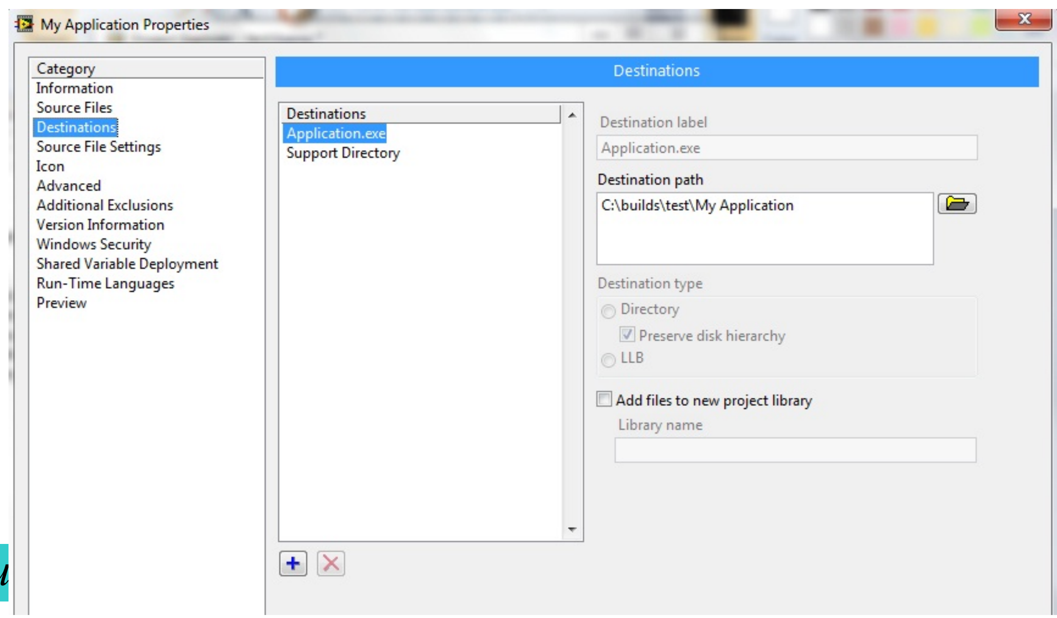


Stand-alone Application – Configure the Application

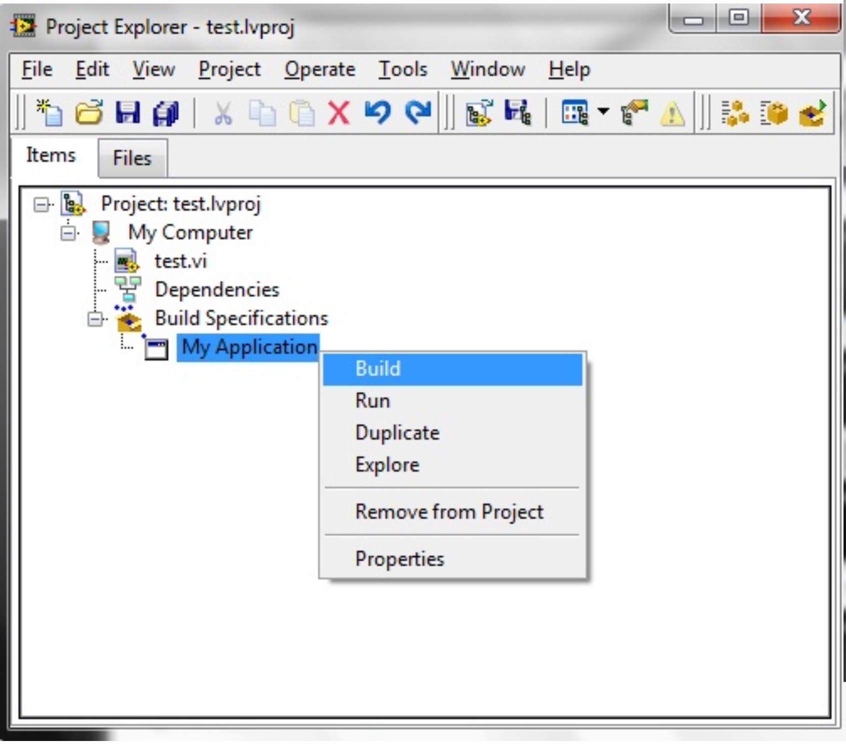
Specify startup VI

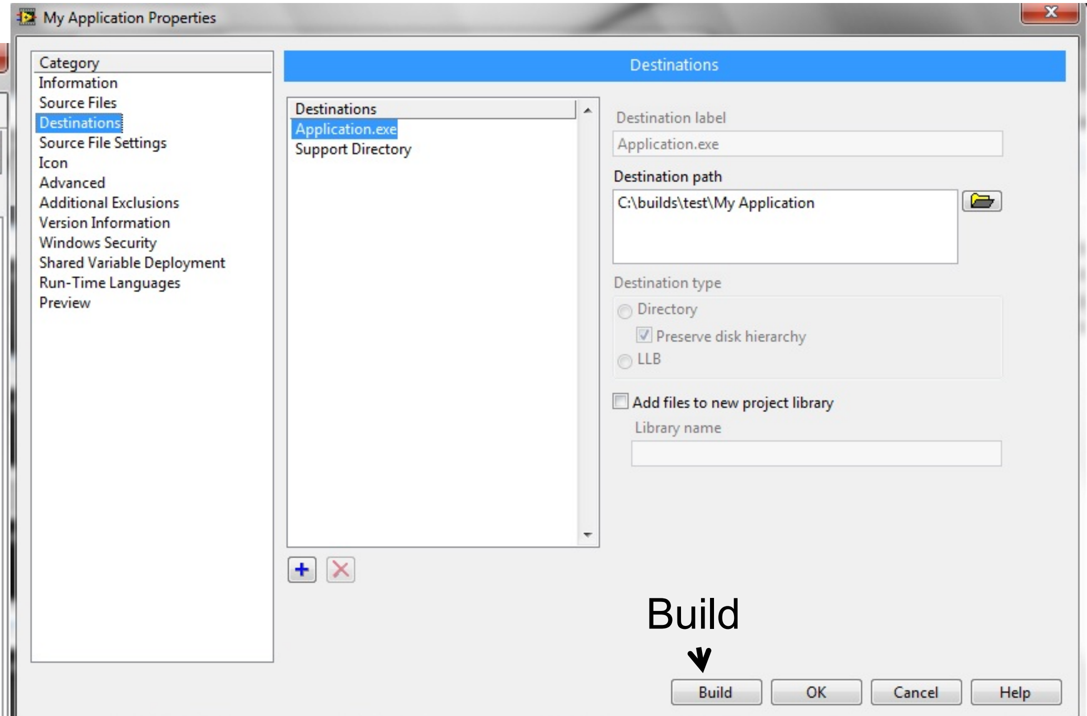


Choose destination



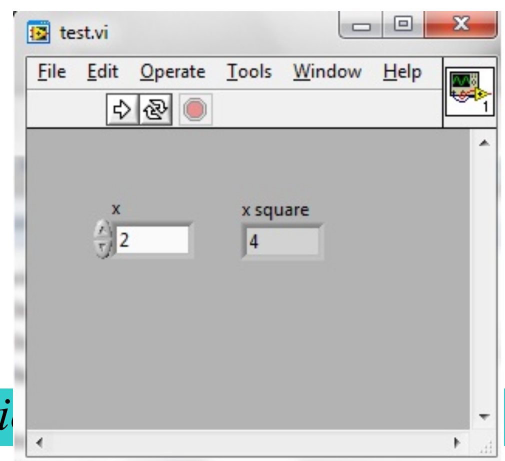
Stand-alone Application – Build





Build

Stand-alone application (*.exe)



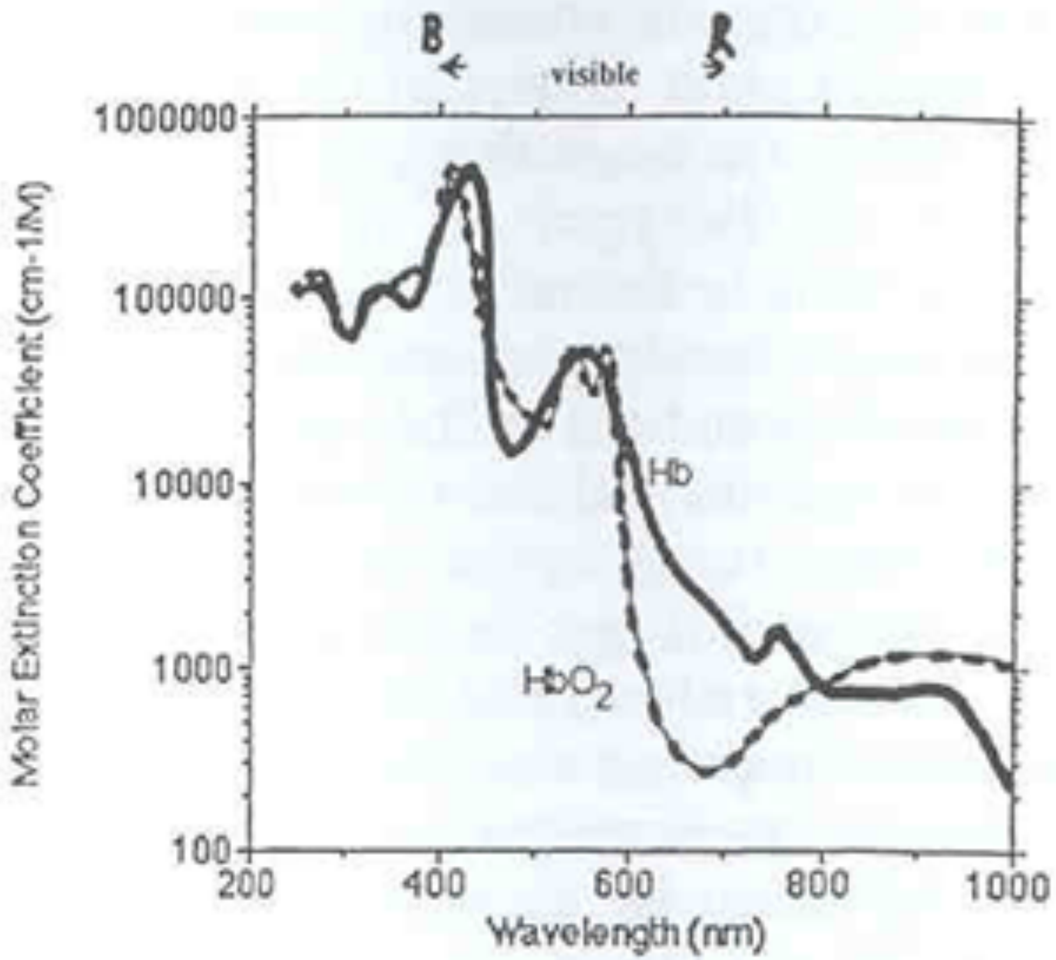


Figure 11 UV-visible Spectra of oxy- (HbO_2) and deoxyhaemoglobin (Hb)

BE/EE189 Design and Construction of Biodevices
Lecture 5

LabVIEW Programming – Data acquisition

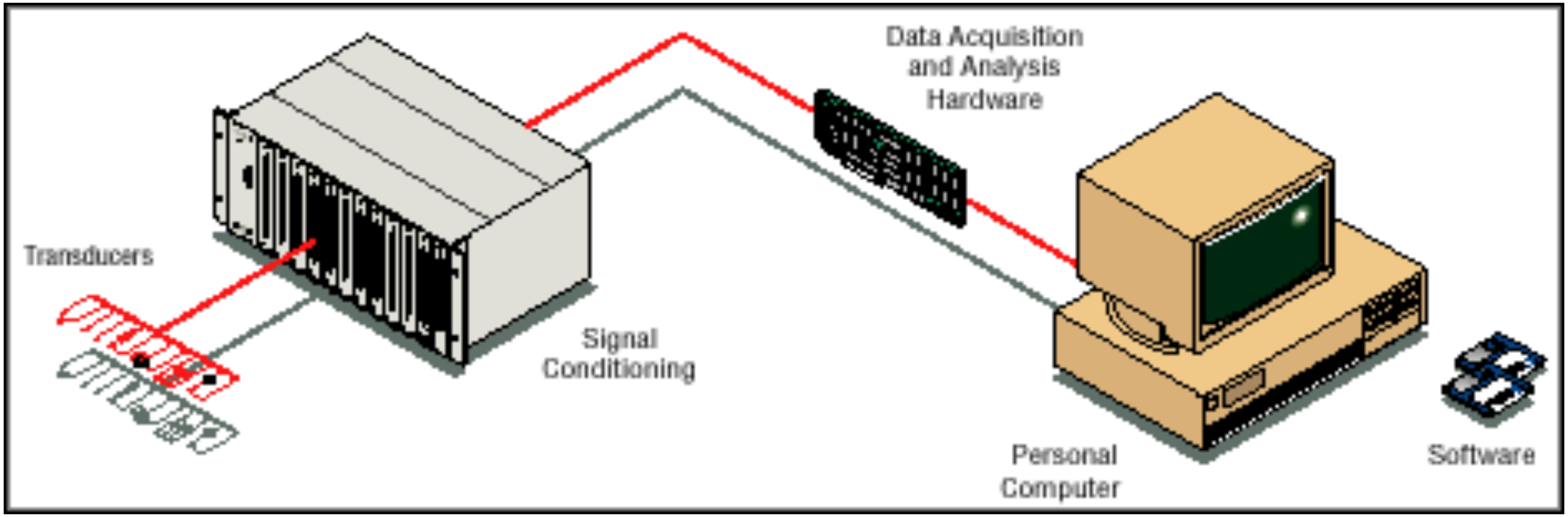
- DAQ system
- Signals and signal conditioning
- Nyquist frequency
- NI ELVIS II
- NI-DAQmx and DAQ assistant

LabVIEW Programming – NI-DAQmx, Strings and File I/O

- Using NI-DAQmx VIs
- Strings
- File I/O

DAQ System

- Data acquisition (DAQ) is the measurement or generation of electrical signals.



Types of Signals and Signal Conditioning

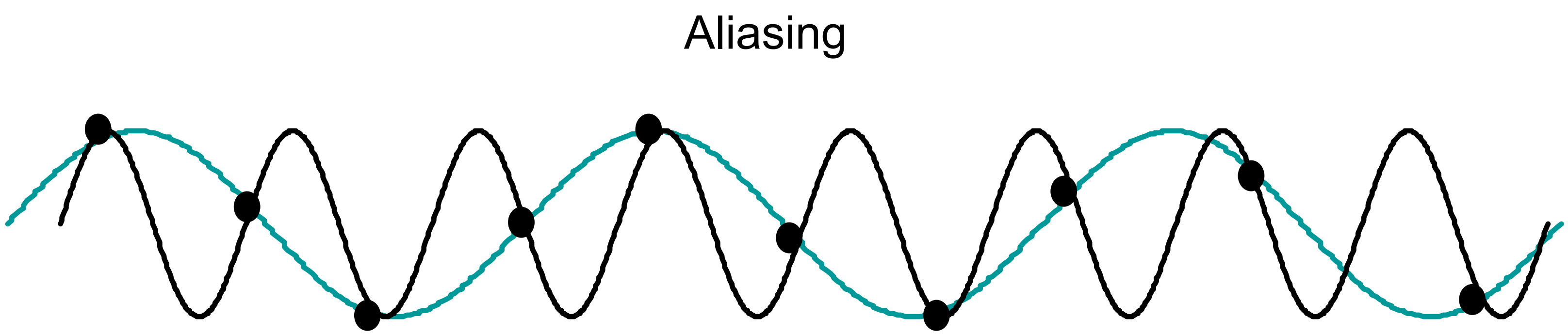
- Digital signals: On-Off, Pulse Train
- Analog signals
  - DC: static or slow changing signals
  - AC: fast changing signals
- Signal conditioning: manipulating the signal before further processing, e.g.,
  - Amplification
  - Filtering
  - Isolation

Common Transducers

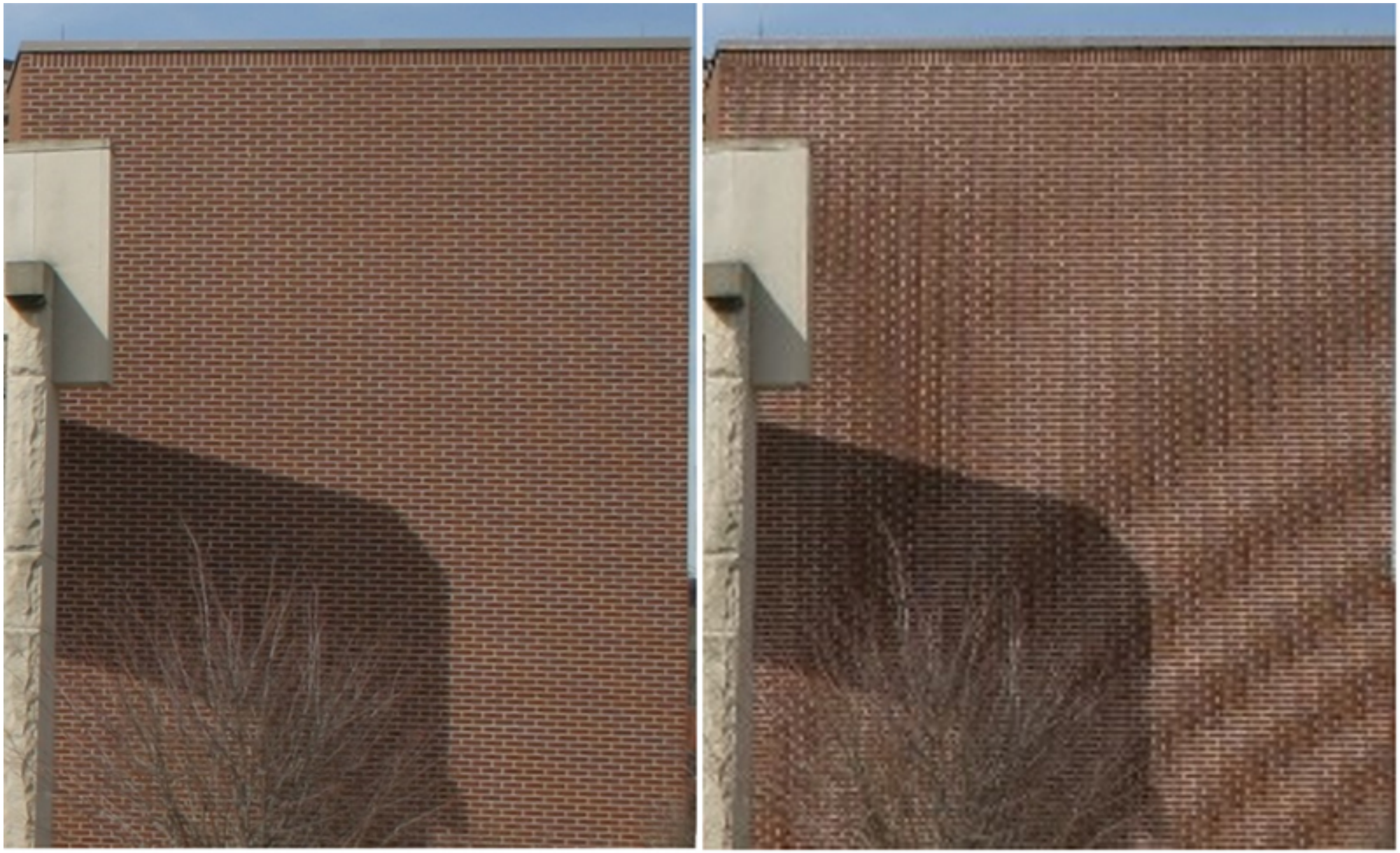
Phenomenon	Transducer
Temperature	Thermocouples Resistance temperature detectors (RTDs) Thermistors Integrated circuit sensor
Light	Vacuum tube photosensors Photoconductive cells
Sound	Microphone
Force and pressure	Strain gauges Piezoelectric transducers Load cells
Position (displacement)	Potentiometers Linear voltage differential transformer (LVDT) Optical encoder
Fluid flow	Head meters Rotational flowmeters Ultrasonic flowmeters
pH	pH electrodes

Sampling and Nyquist Frequency

- For a given sampling rate, we can only recover signals with maximum frequency less than the Nyquist frequency, which is half of the sampling rate.
- Aliasing will occur if the maximum signal frequency is larger than the Nyquist frequency.

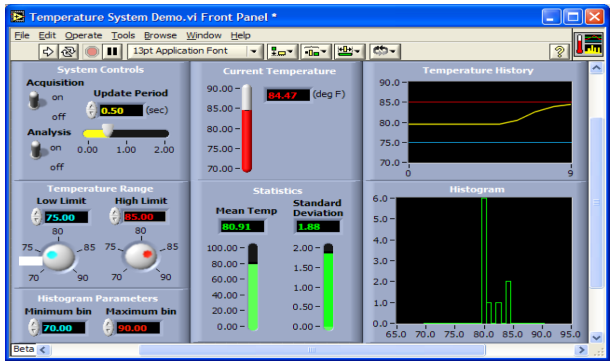


Aliasing

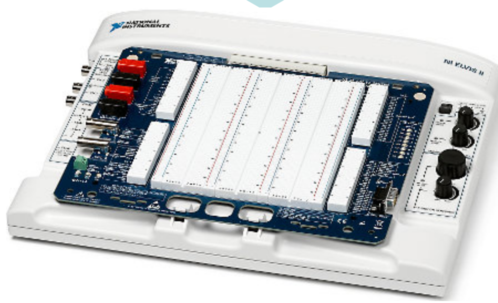


NI Data Acquisition Framework

LabVIEW



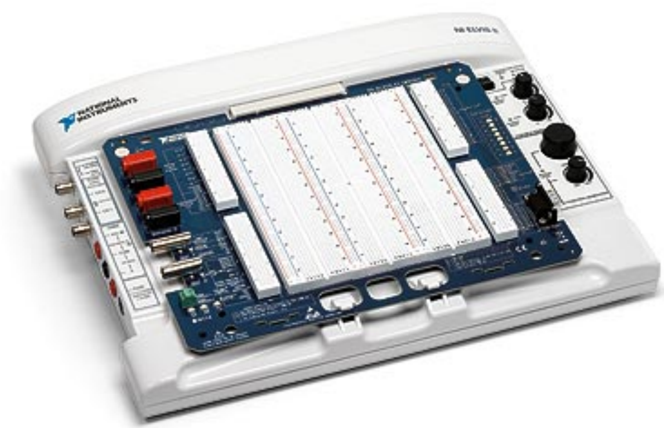
NI-DAQmx



We will use NI ELVIS II

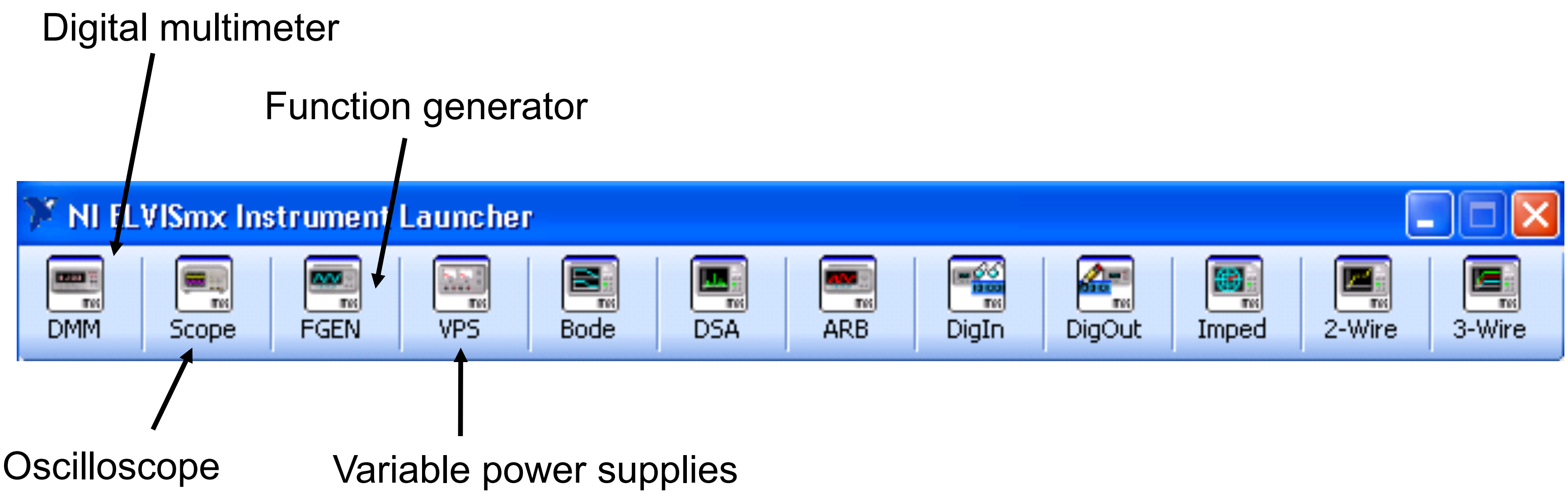
NI ELVIS II

- ELVIS - Educational Laboratory Virtual Instrumentation Suite
- High-speed USB plug-and-play connectivity
- 12 virtual instruments: oscilloscope, digital multimeter, function generator, variable power supply, etc.
- Bread board for circuit prototyping.



Virtual Instruments of NI ELVIS II

Instrument launcher



NI ELVIS II - Circuit Prototyping

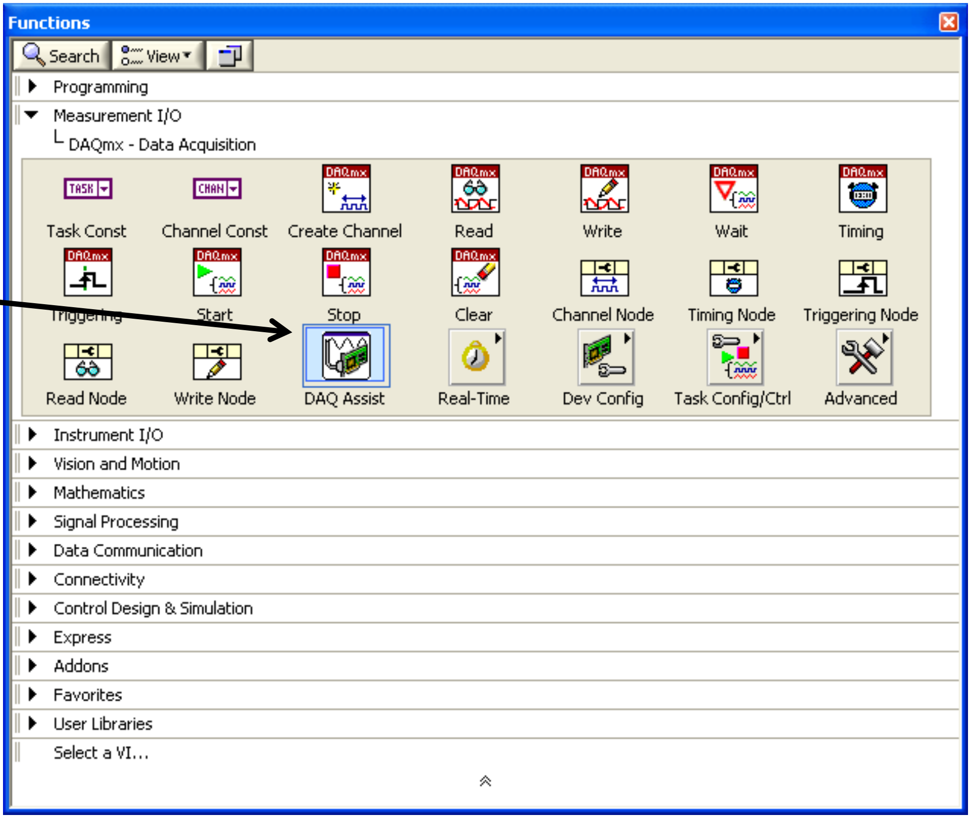
- Fixed power supply: +5V, +/-15V.
- Variable power supply: 0 to 12V, 0 to -12V.
- 16 single-ended, 16-bit analog input, maximum 1.25 MS/s sampling rate.
- Two 16-bit analog outputs (2.8 MS/s); 24 digital I/O.
- LEDs for indication.

NI-DAQmx

- NI-DAQmx: a DAQ driver architecture with significant improvement over previous NI-DAQ drivers.
- Physical channel: a terminal or pin at which an analog or digital signal is measured or generated
- Virtual channel: a collection of property settings that can include a name, a physical channel, input terminal connections, the type of measurement or generation, and scaling information.
- Task: a collection of one or more virtual channels with timing, triggering, and other properties.

NI-DAQmx-Data Acquisition Palette

DAQ Assistant

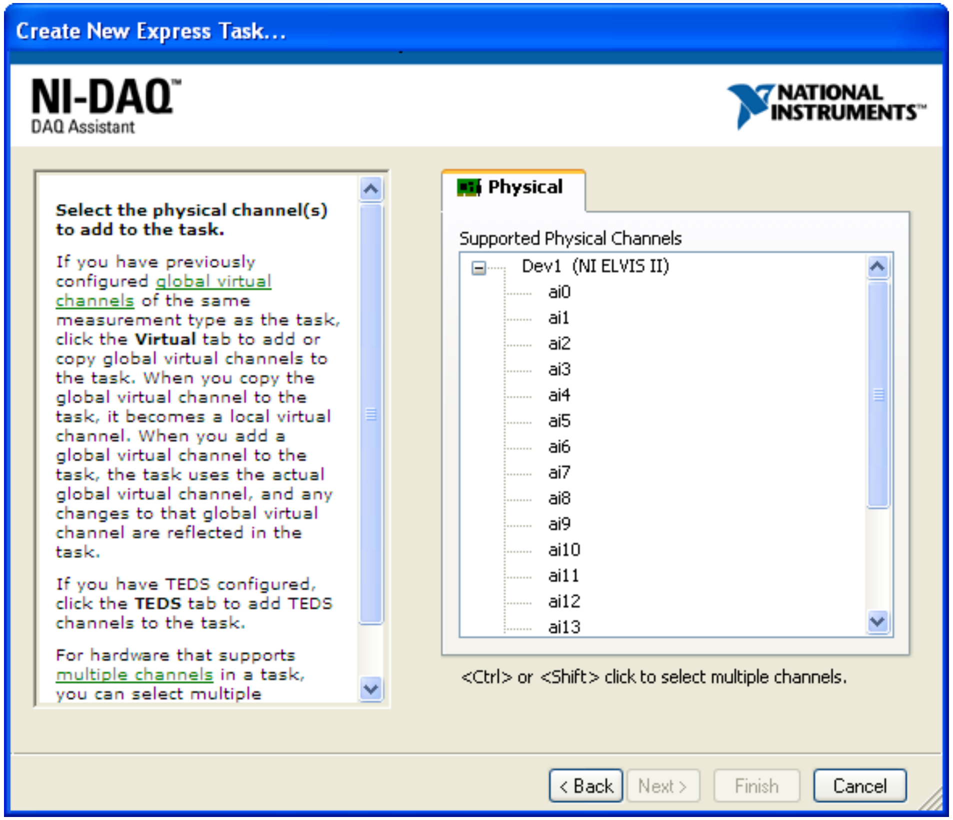


Using DAQ Assistant – 1

- You can acquire or generate signals
- Example: select "Acquire Signals >> Analog Input >> Voltage" for acquiring analog voltage signal.

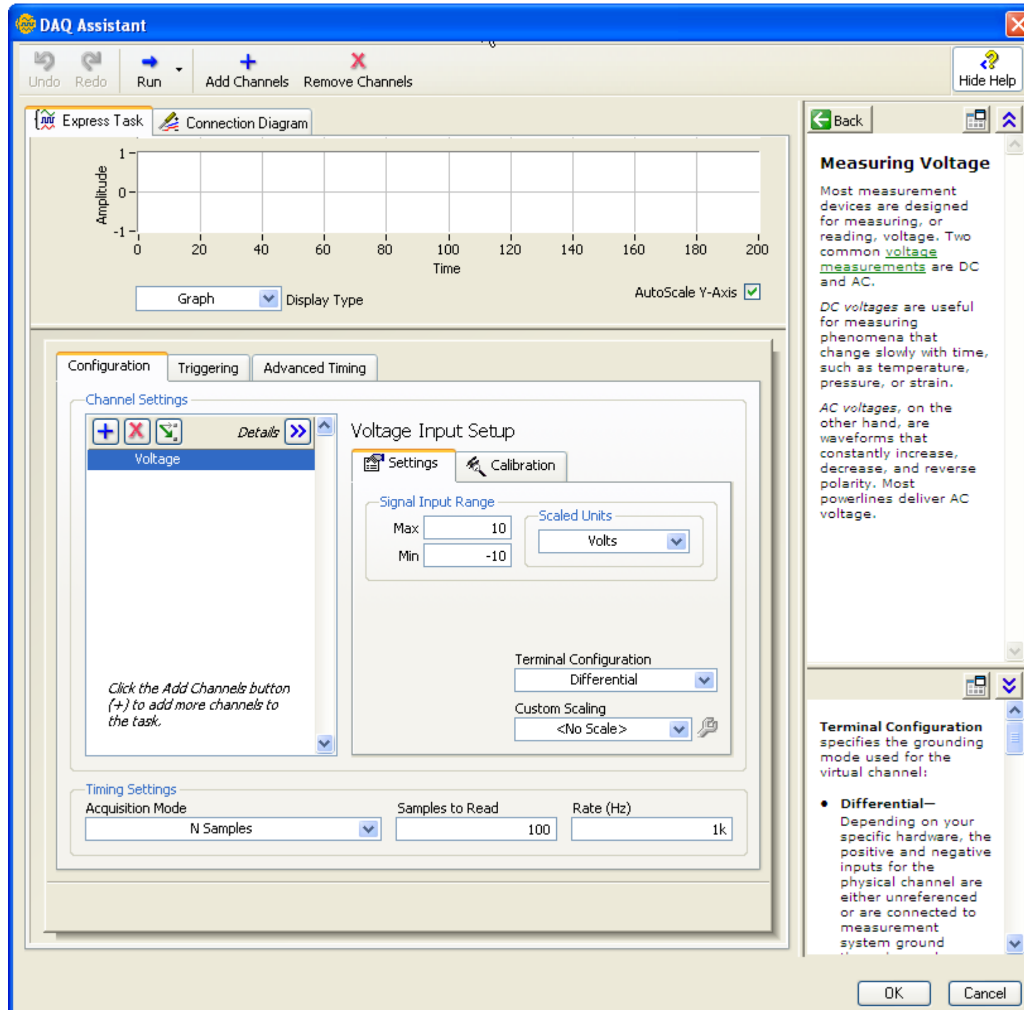
Using DAQ Assistant – 2

- Select the physical channel



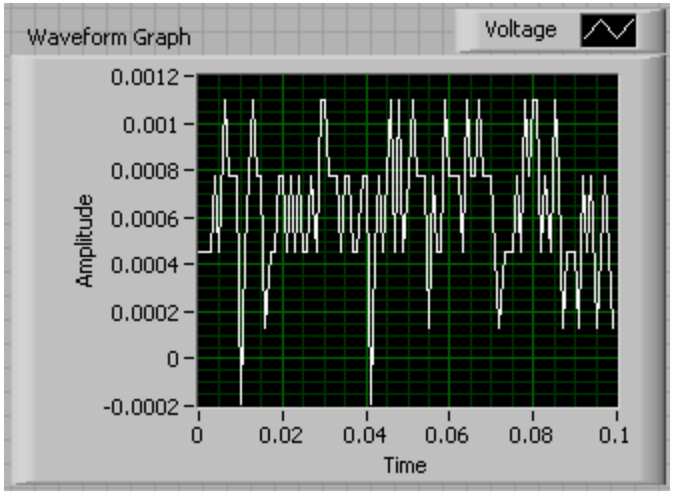
Using DAQ Assistant – 3

- Configuring the channel settings and testing the DAQmx task

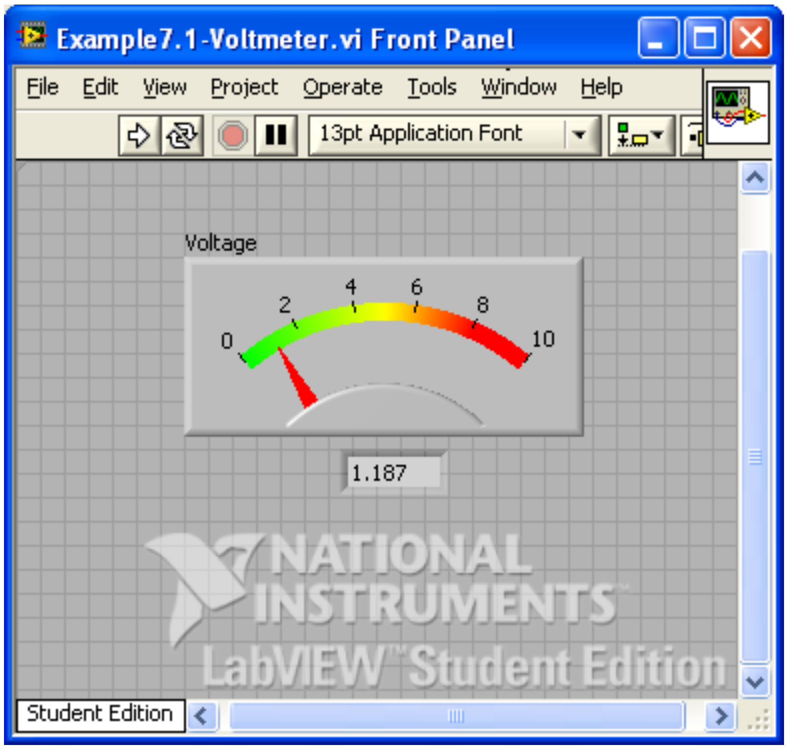


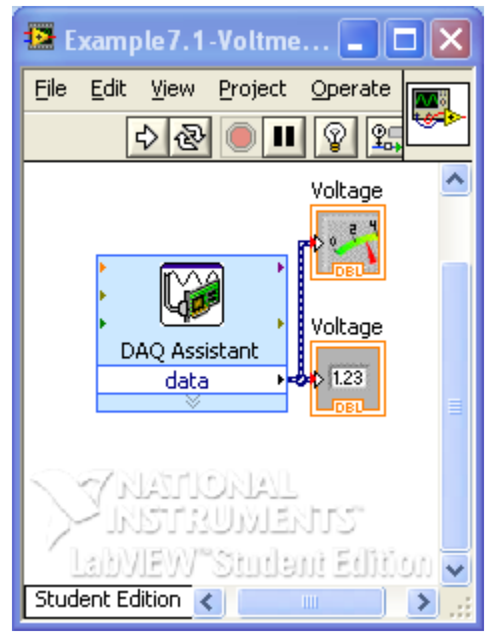
Using DAQ Assistant – 4

- The output can be displayed in a waveform graph



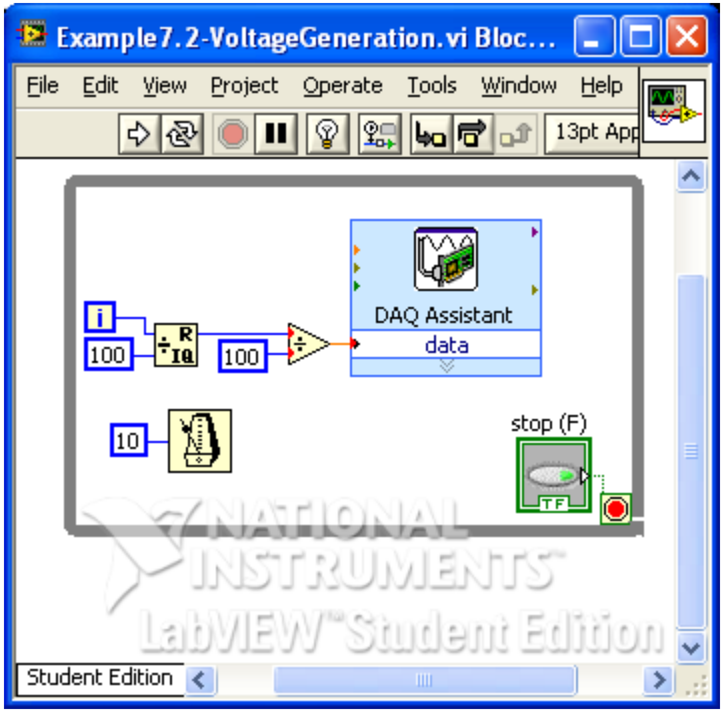
Work Example 5.1 – Voltmeter





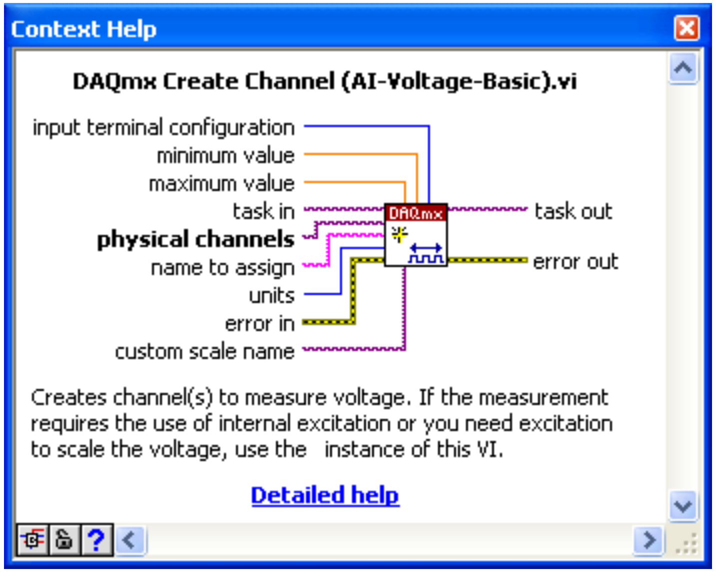
Work Example 5.2 – Voltage Generation

- Generate a sawtooth waveform



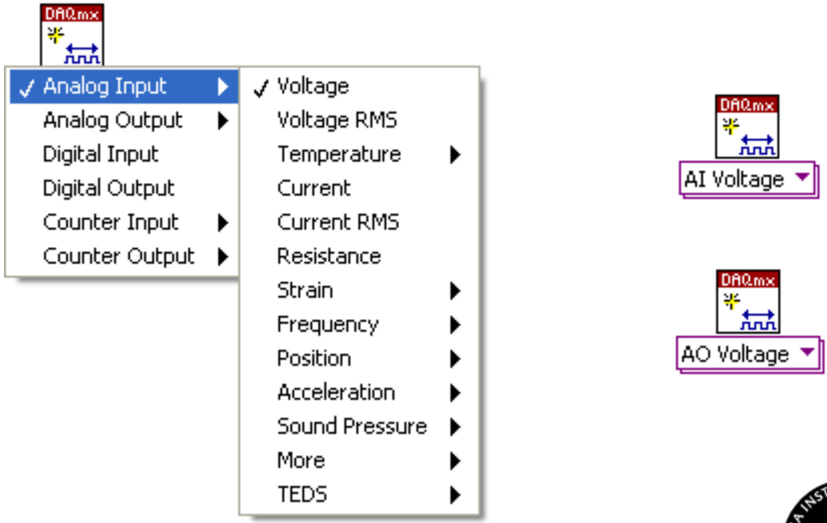
NI-DAQmx VIs – Create Virtual Channel

- DAQmx Create Channel: Creates a virtual channel or set of virtual channels and adds them to a task. If you do not specify a task, NI-DAQmx creates a task for you and adds the virtual channels this VI creates to that task



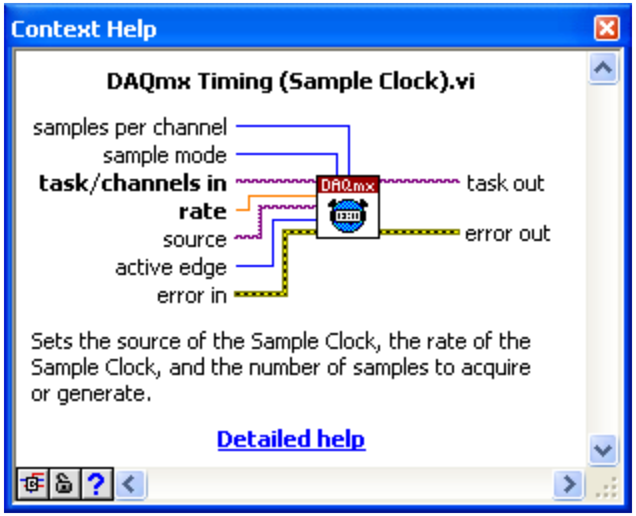
Select channel types

Examples



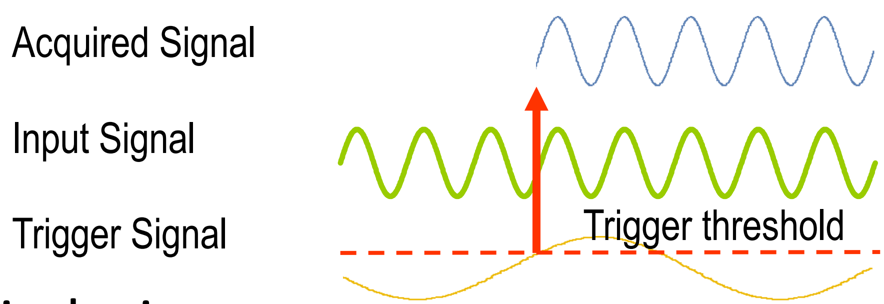
NI-DAQmx VIs – Timing

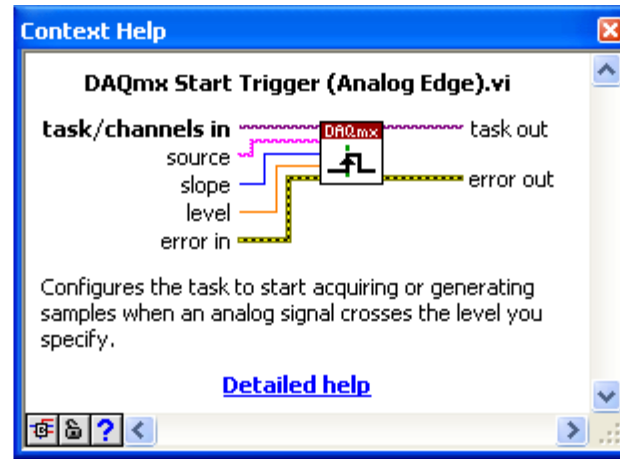
- Configures the number of samples to acquire or generate and creates a buffer when needed. Specify the sampling rate or use an external clock.



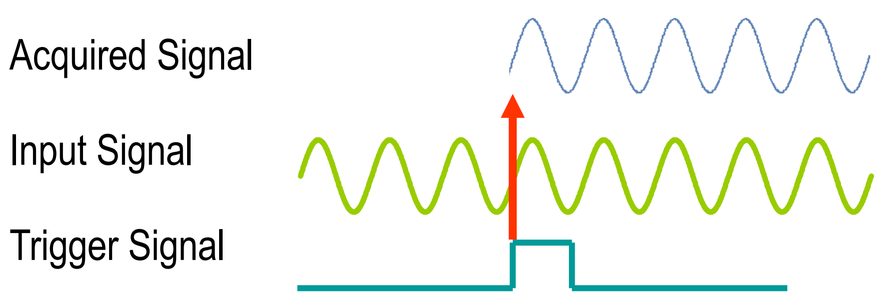
NI-DAQmx VIs – Trigger

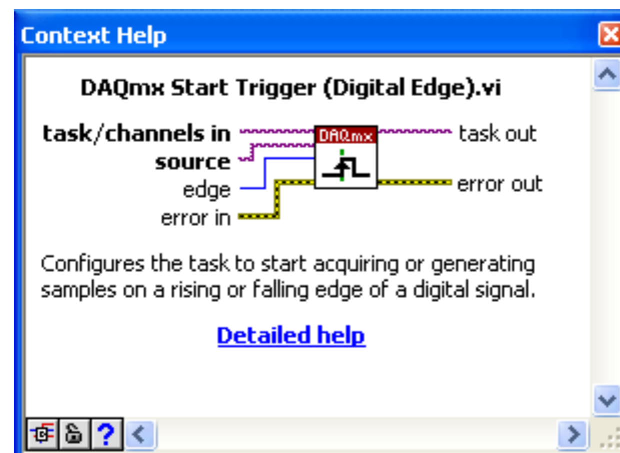
- Configures triggering for the task.
- Analog trigger





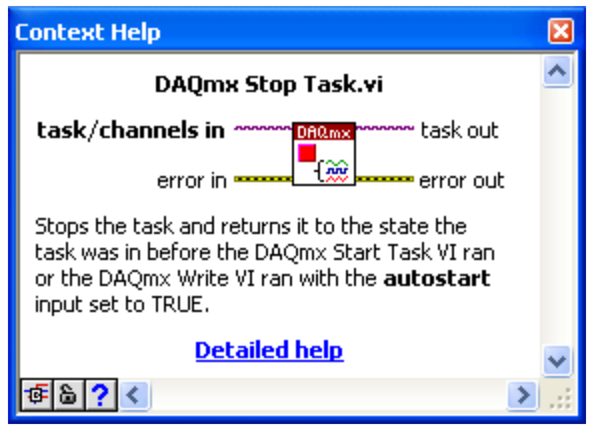
- Digital trigger

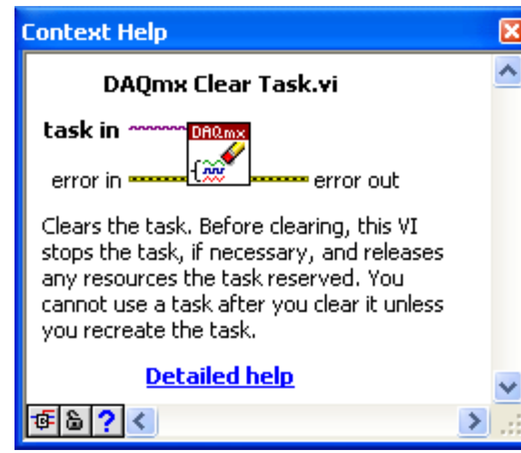




NI-DAQmx VIs – Start, Stop, and Clear

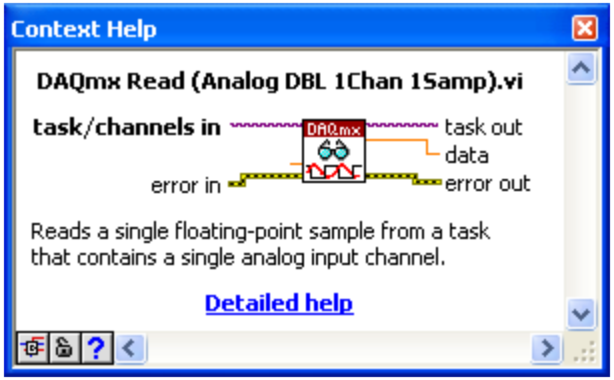
- To start, stop or clear the task

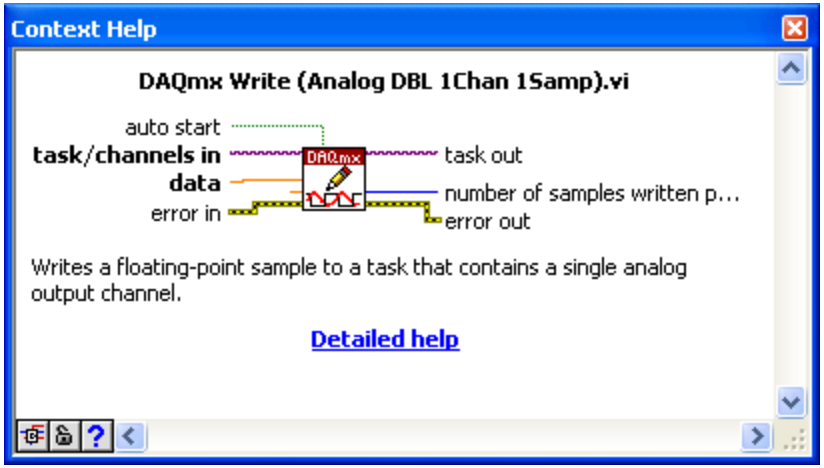




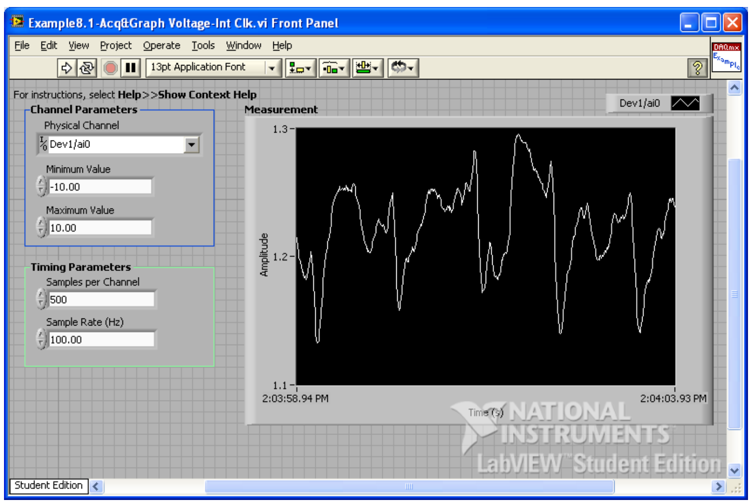
NI-DAQmx VIs – Read, Write

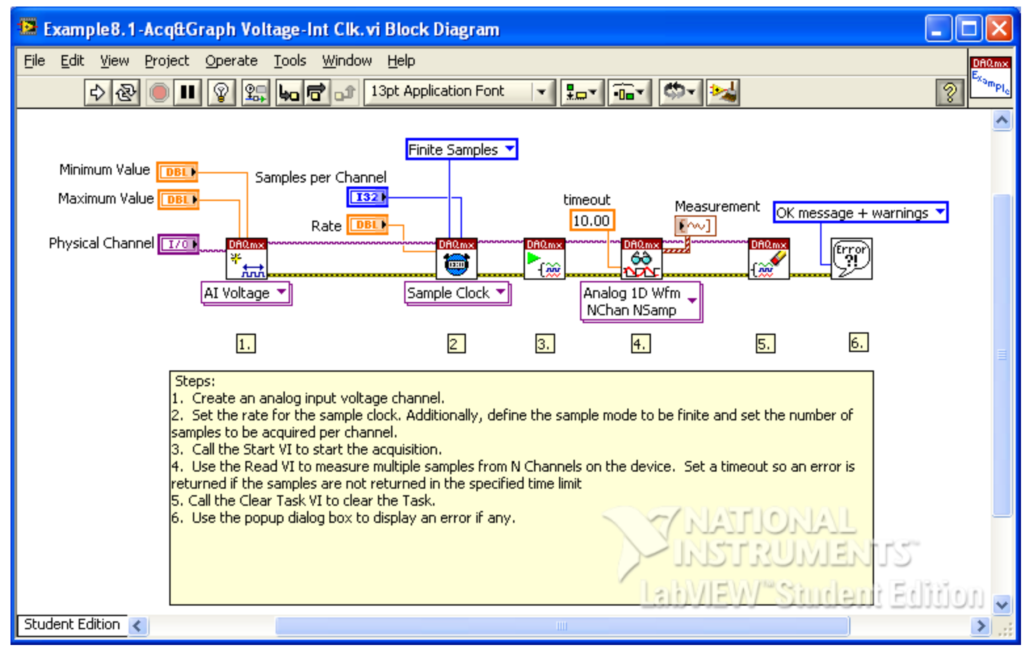
- To read from or write to the task/channels.

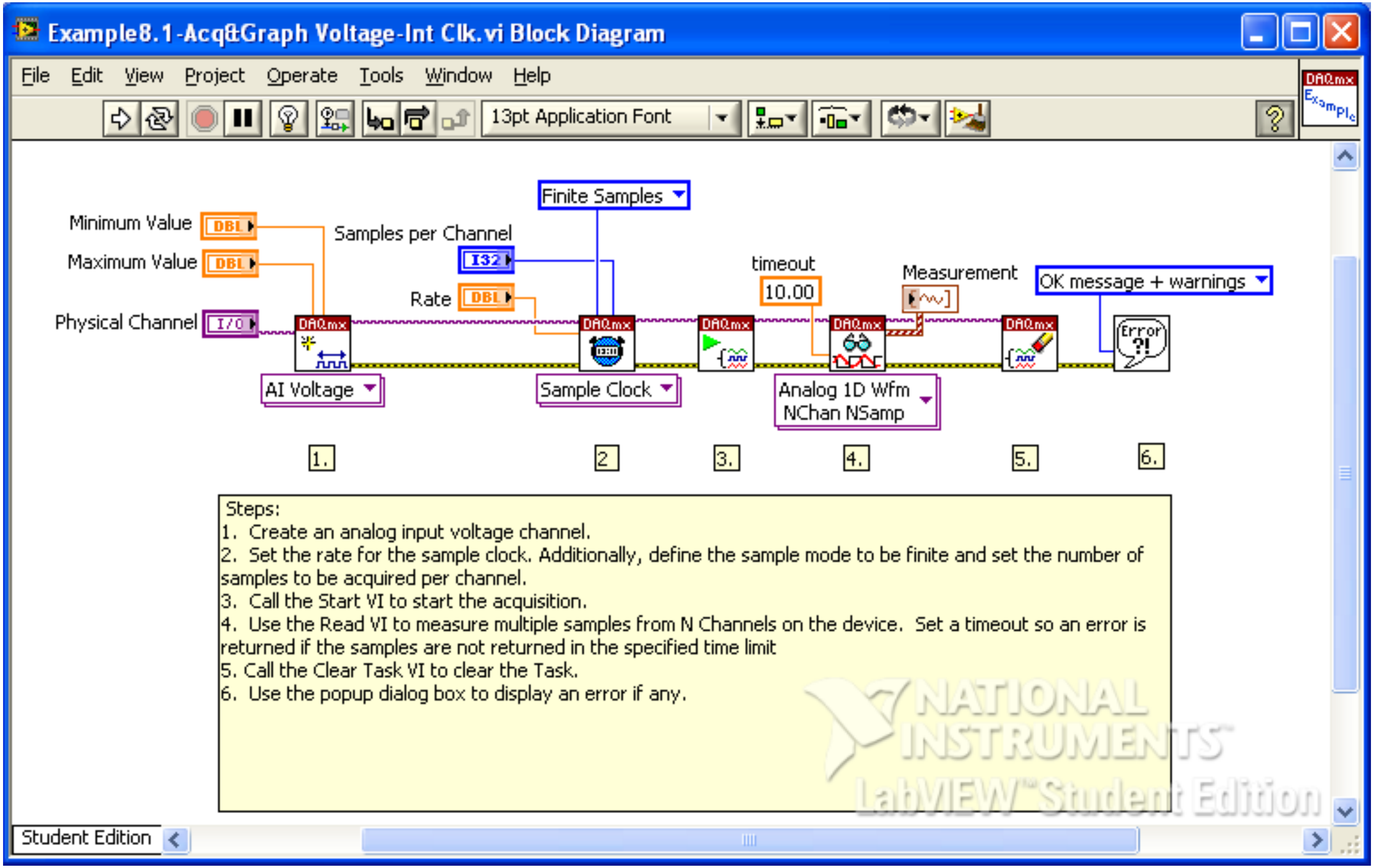




Example – Acq&Graph Voltage-Int Clk (From LabVIEW Examples)

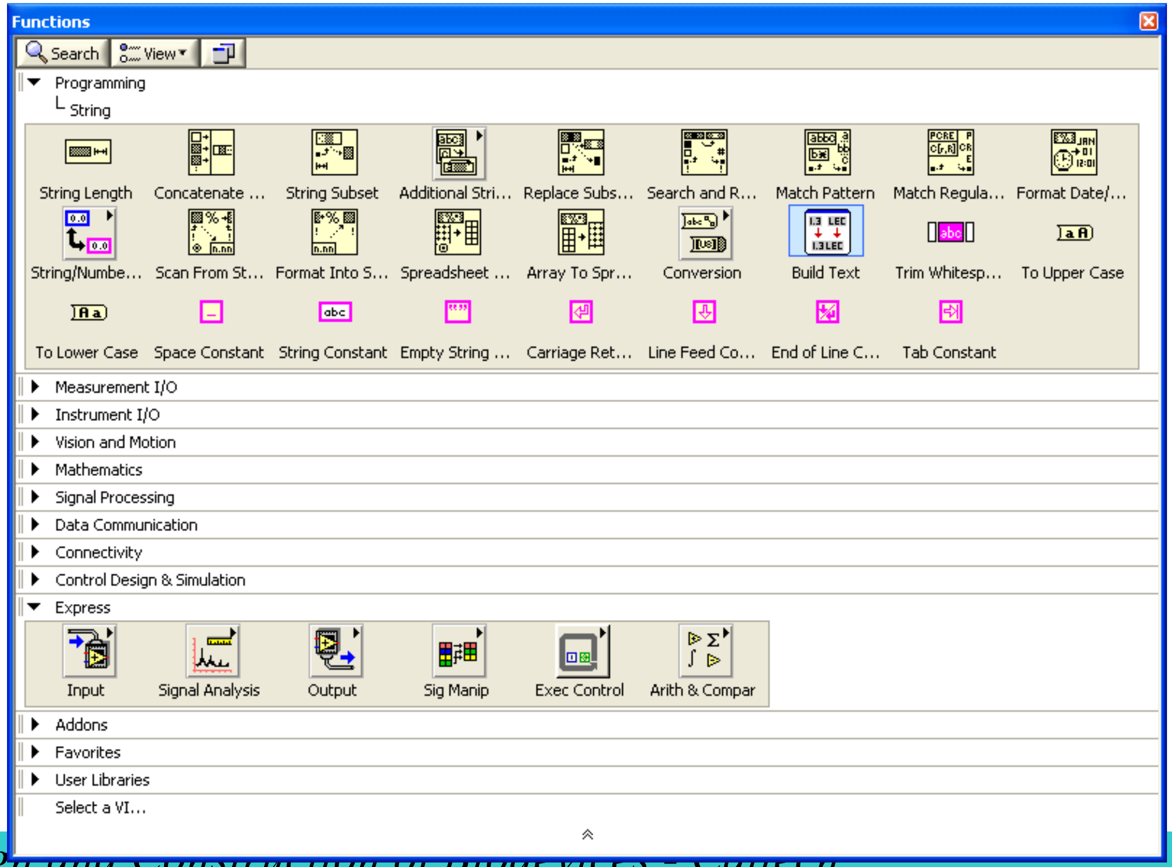





Example8.1-Acq&Graph Voltage-Int Clk.vi Block Diagram
File Edit View Project Operate Tools Window Help
13pt Application Font
Minimum Value
Maximum Value
Physical Channel
Samples per Channel
Rate
Finite Samples
timeout
10.00
Measurement
OK message + warnings
AI Voltage
Sample Clock
Analog 1D Wfm NChan NSamp
1.
2
3.
4.
5.
6.
Steps:
1. Create an analog input voltage channel.
2. Set the rate for the sample clock. Additionally, define the sample mode to be finite and set the number of samples to be acquired per channel.
3. Call the Start VI to start the acquisition.
4. Use the Read VI to measure multiple samples from N Channels on the device. Set a timeout so an error is returned if the samples are not returned in the specified time limit
5. Call the Clear Task VI to clear the Task.
6. Use the popup dialog box to display an error if any.
NATIONAL INSTRUMENTS
LabVIEW Student Edition
Student Edition

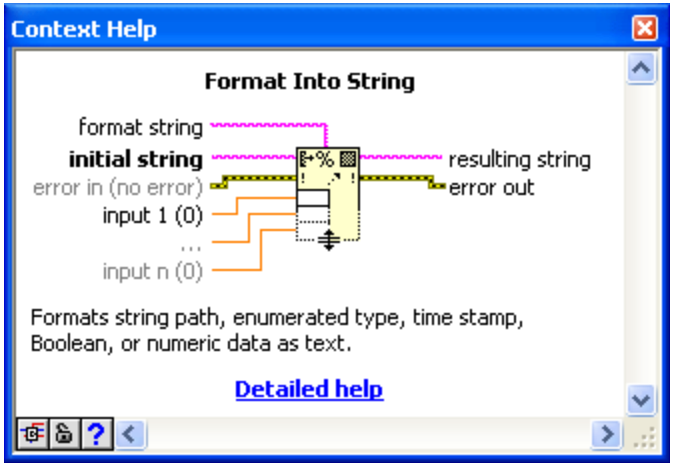
Strings

- String: a sequence of characters that can be displayable or nondisplayable.

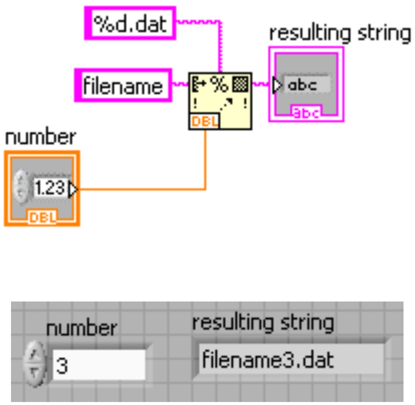


Format into String

- The format string is similar to those in C except some additional features.

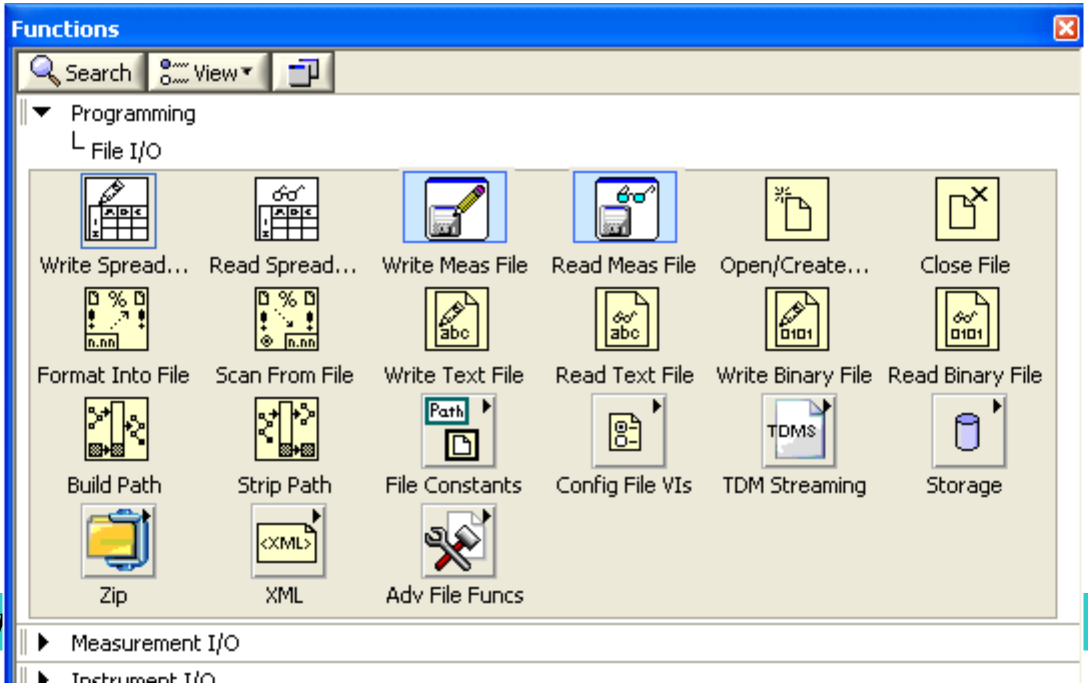


Example



File I/O

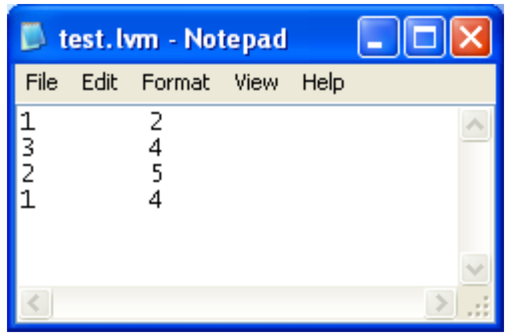
- File I/O operations pass data to and from files
  - Opening and closing data files.
  - Reading data from and writing data to files.
  - Reading from and writing to spreadsheet-formatted files.
  - Moving and renaming files and directories.
  - Changing file characteristics.

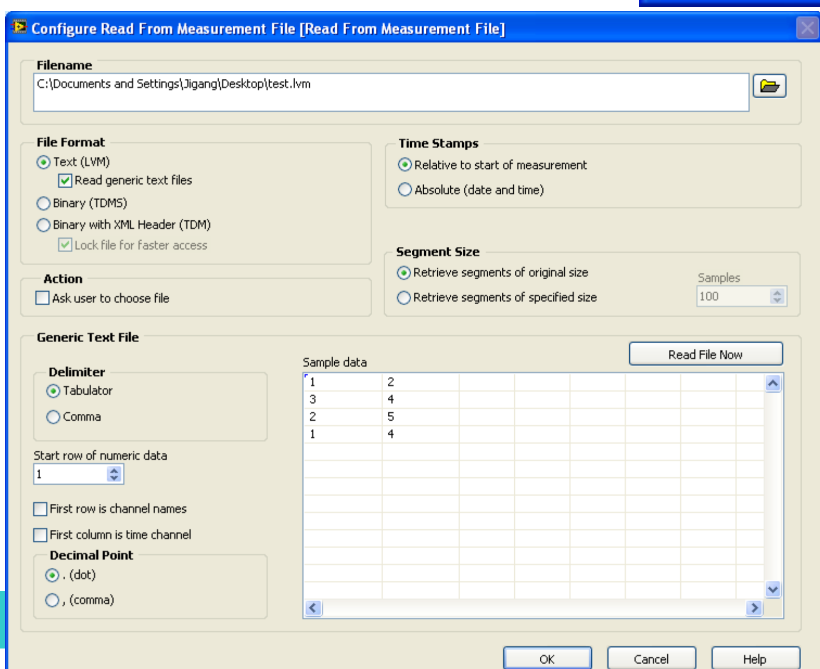


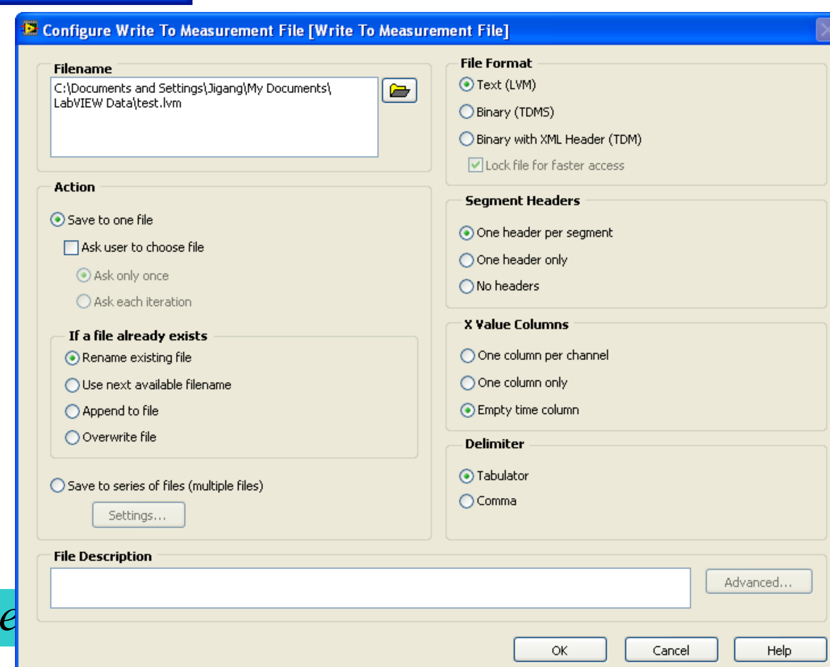
Read from and Write to Measurement File

- Use these express VIs for easy writing and reading

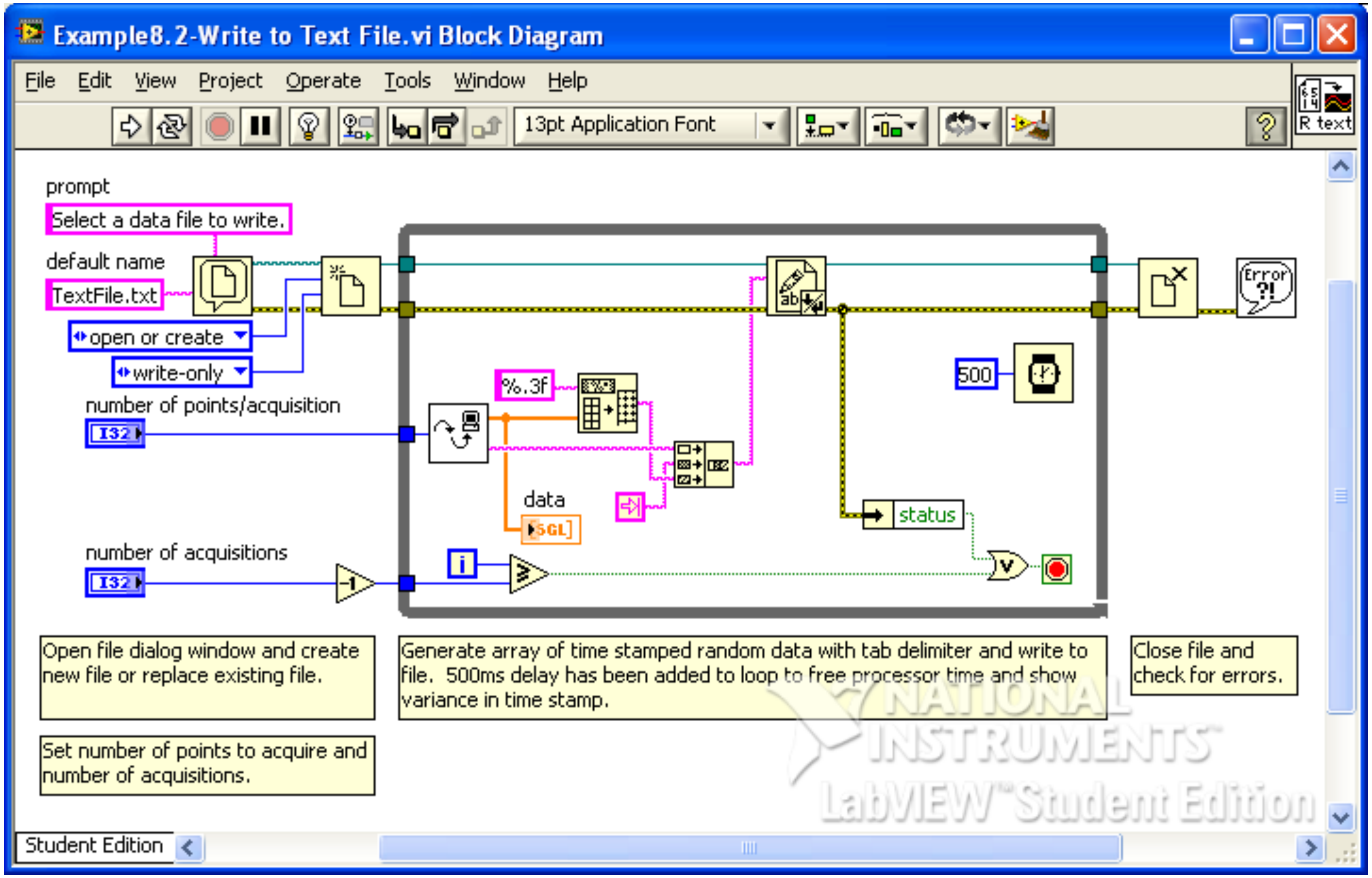
Example lvm file



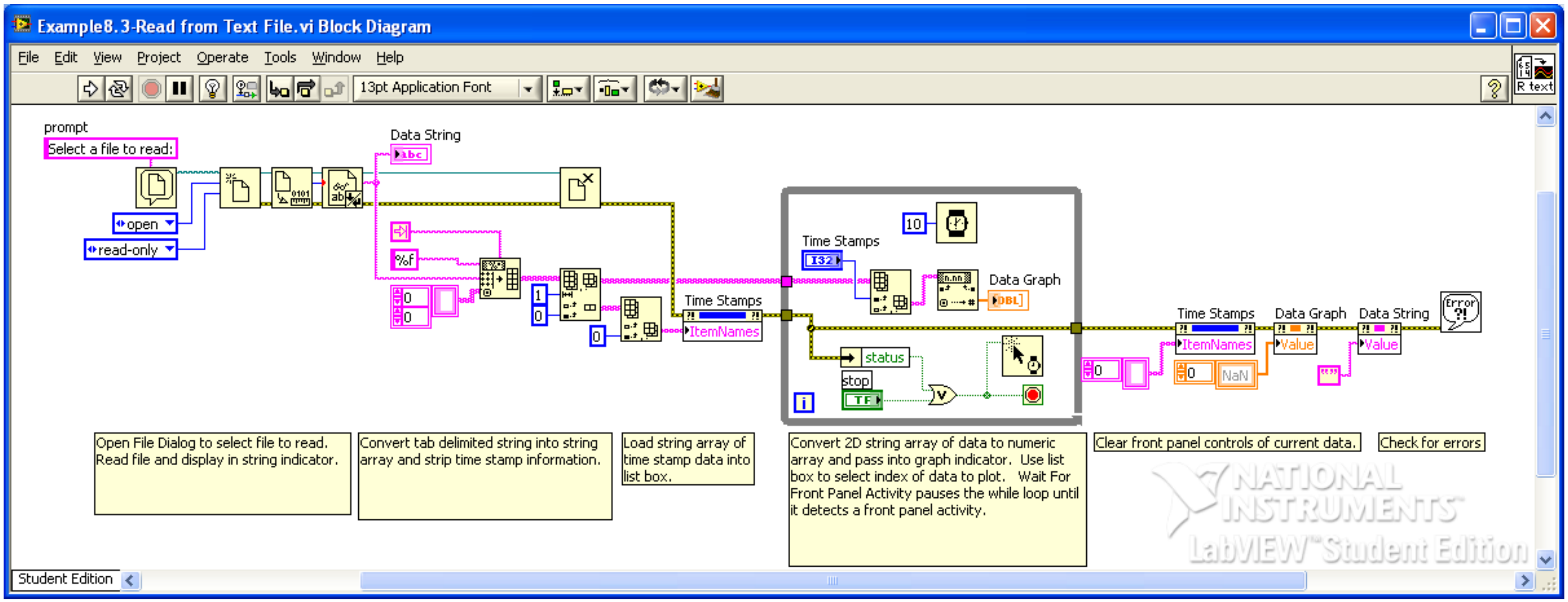




Example – Write to Text File (From LabVIEW Examples)



Example – Read from Text File (From LabVIEW Examples)



BE/EE189 Design and Construction of Biodevices
Lecture 6

LabVIEW Programming – MathScript, Matlab, Curve Fitting, and FFT

- MathScript RT module
- Matlab integration
- Curve fitting
- Signal processing – transforms

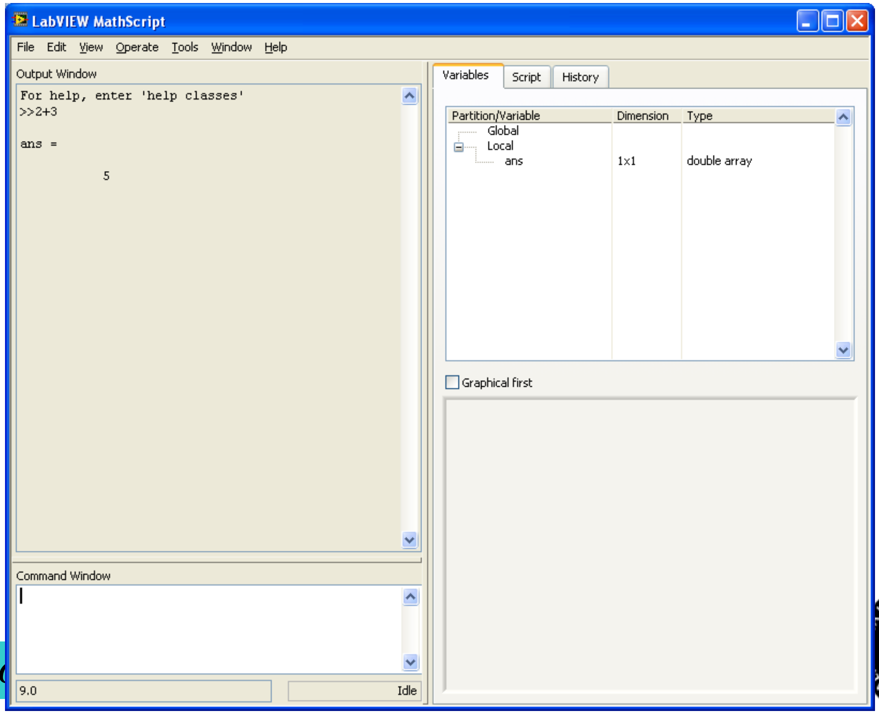
LabVIEW Programming – Analysis and Signal Processing

- Differential Equations
- Integration and Differentiation
- Signal generation and processing

MathScript RT Module

- Provides access to a text-based math-oriented language with a command-prompt from within the LabVIEW development environment.
- Similar to Matlab in syntax.

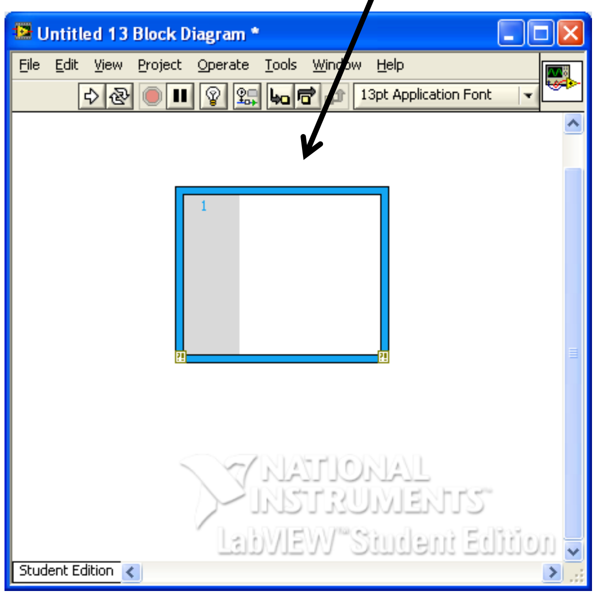
MathScript interactive window

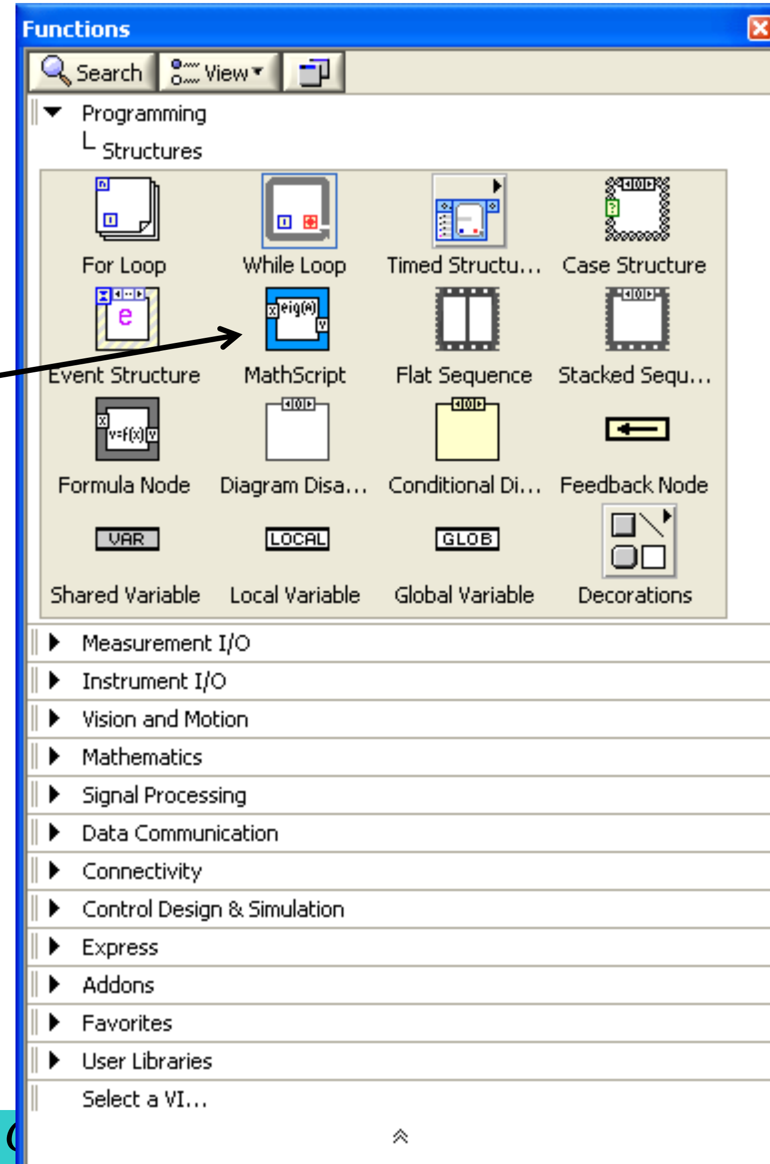


MathScript Nodes

- MathScript can be integrated into the VI using the MathScript Nodes.

MathScript Node

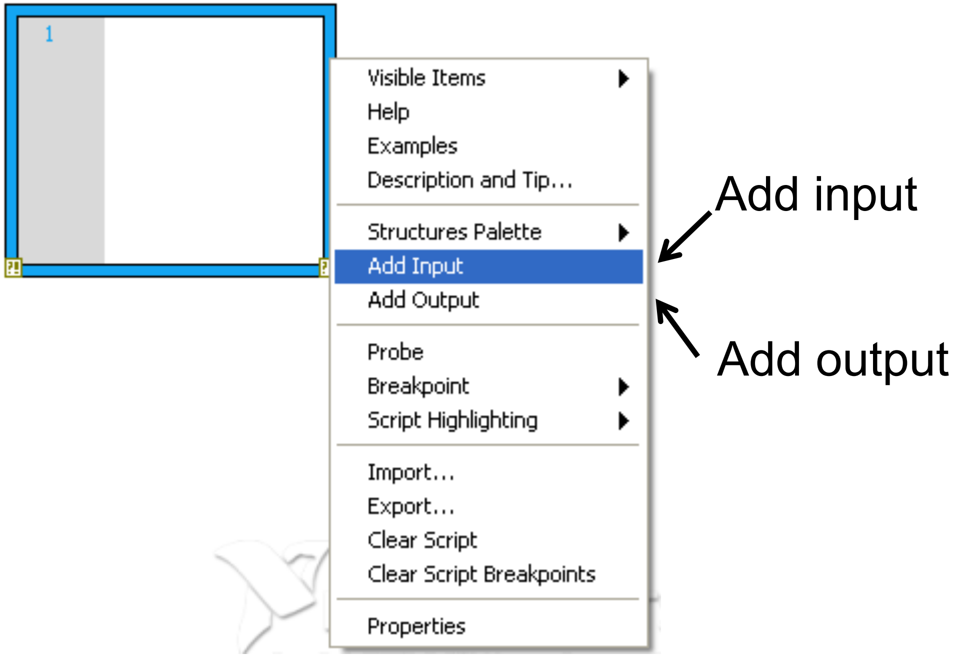




MathScript Node – Input and Output

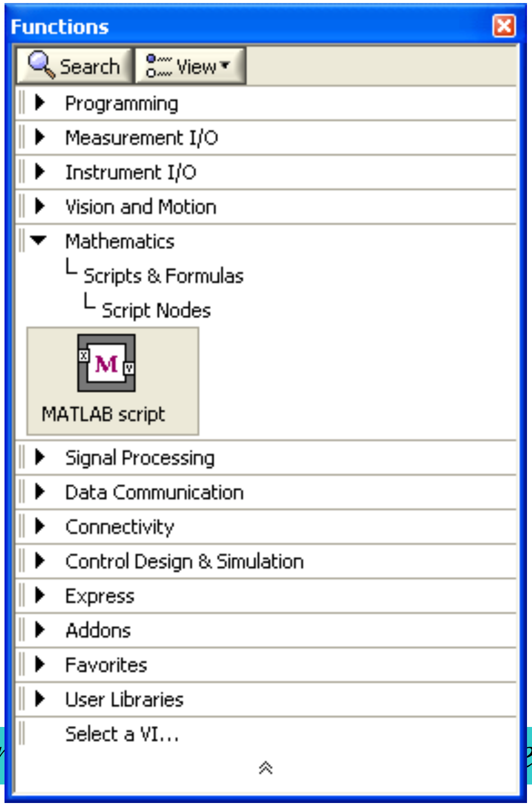
- To interact with the rest of VI, the MathScript Node usually need to specify input and output.

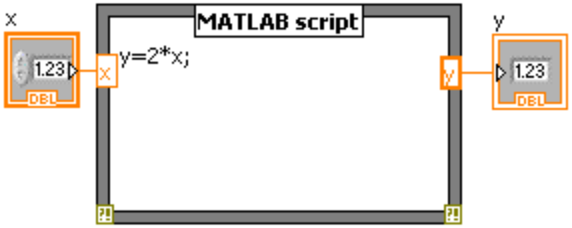
Right click the MathScript Node



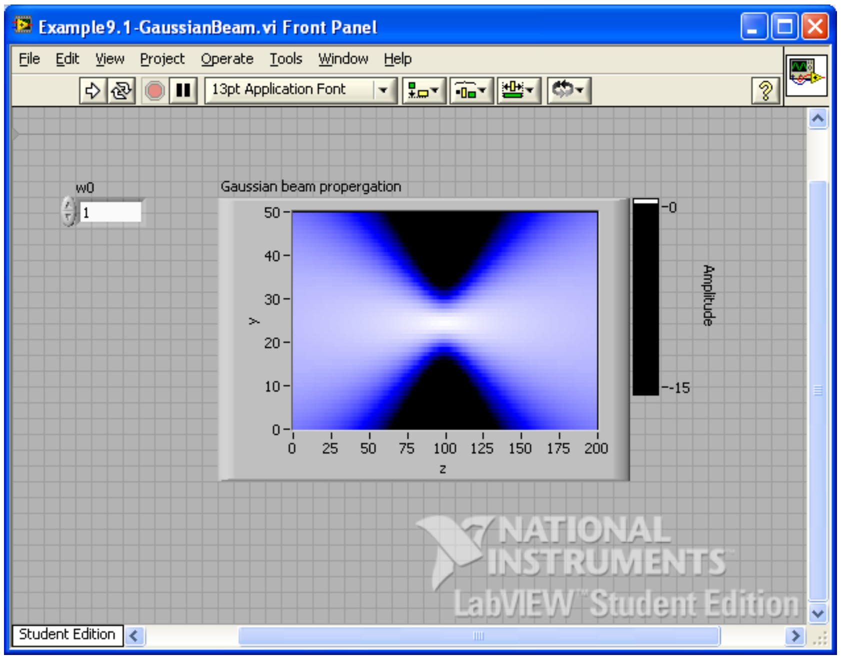
Matlab Integration

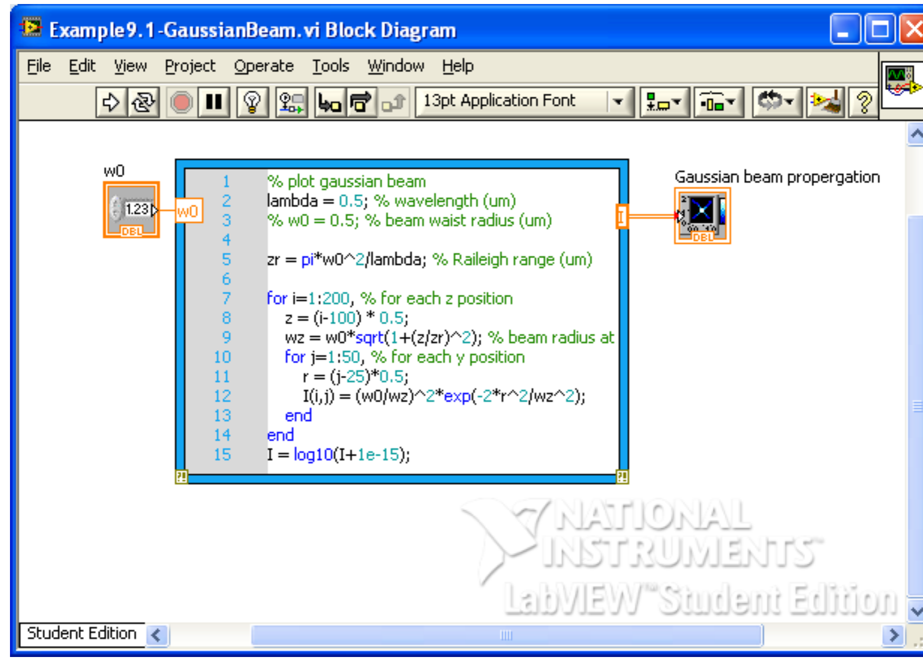
- Matlab script can be used directly in LabVIEW, similar to the MathScript Node.
- Matlab must be installed.



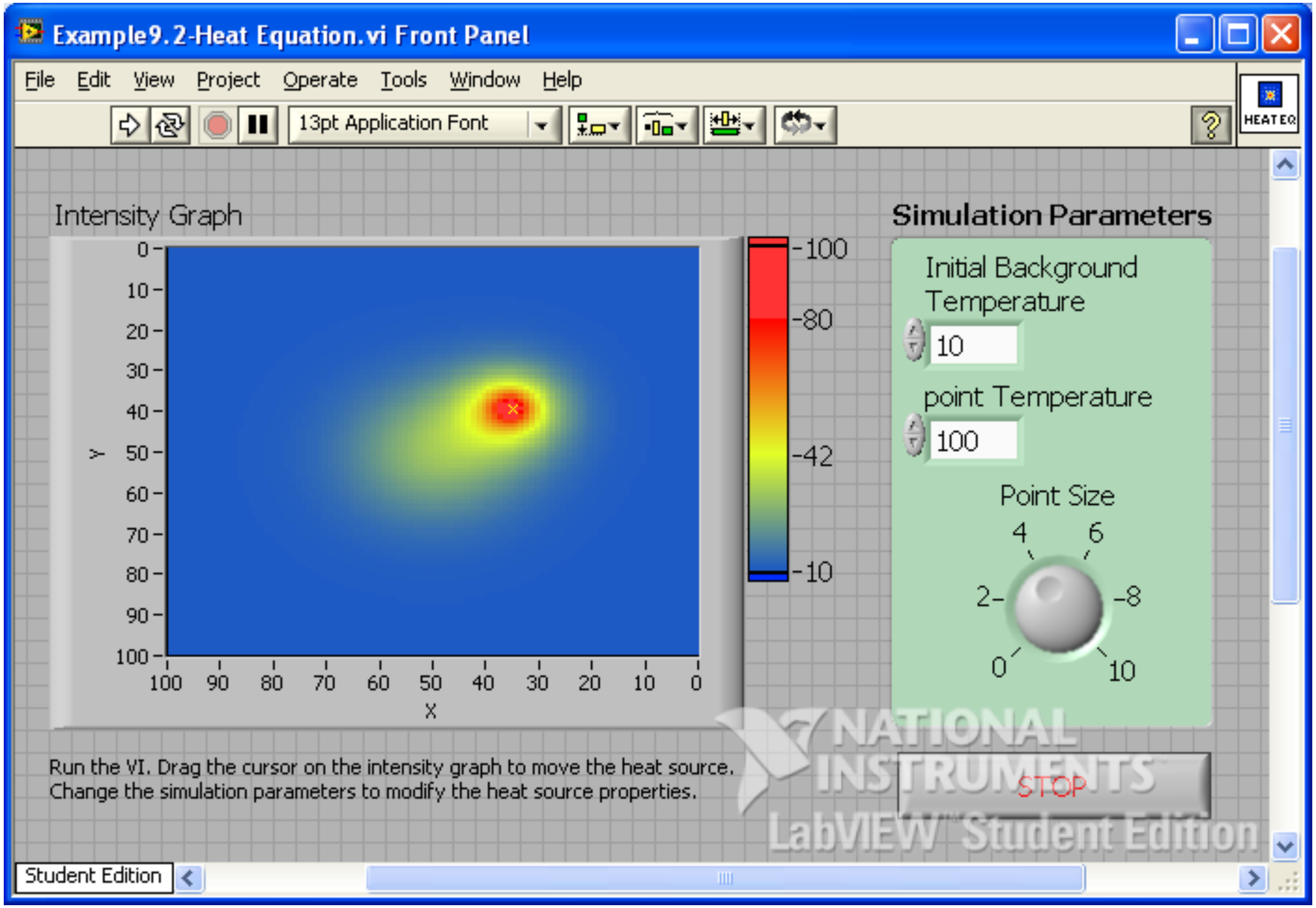


Example – Gaussian Beam Propagation





Example – Heat Transfer (From LabVIEW Example)



Curve Fitting

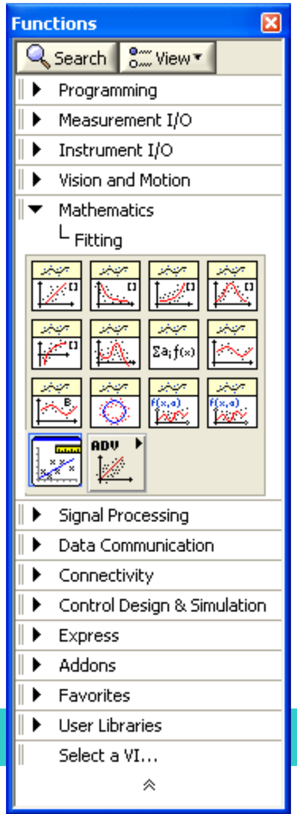
- Least squares method

Error → e, Function → f, Observed data → y; coefficients → a

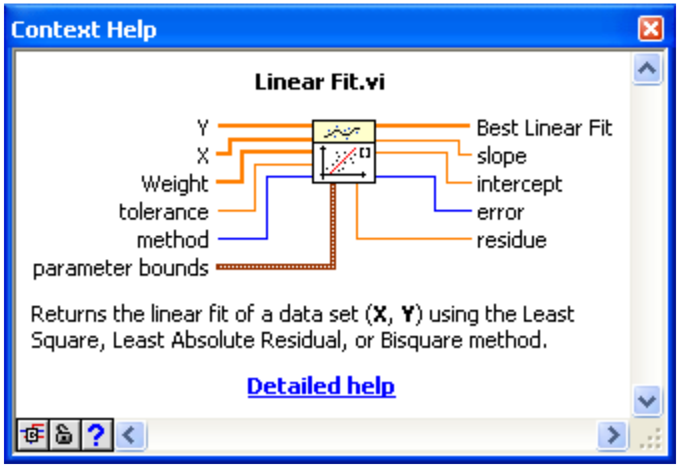
$$e(a) = [f(x,a) - y(x)]^2$$

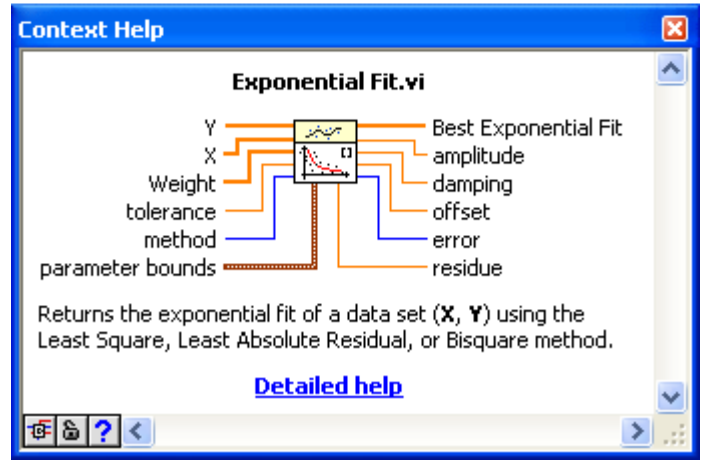
Jacobian equation: $$\frac{\partial e(a)}{\partial a} = 0$$

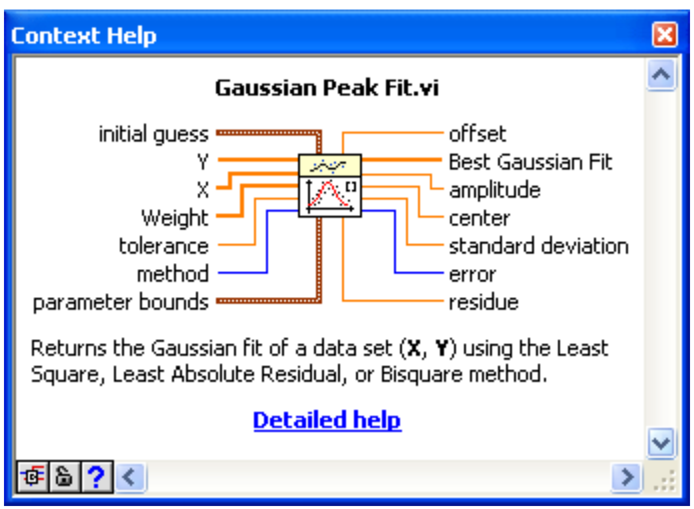
- Curve-fitting in LabVIEW:
  - Linear fit
  - Exponential fit
  - General polynomial fit
  - General linear fit
  - Nonlinear Levenberg-Marquardt fit
  - B-spline fit

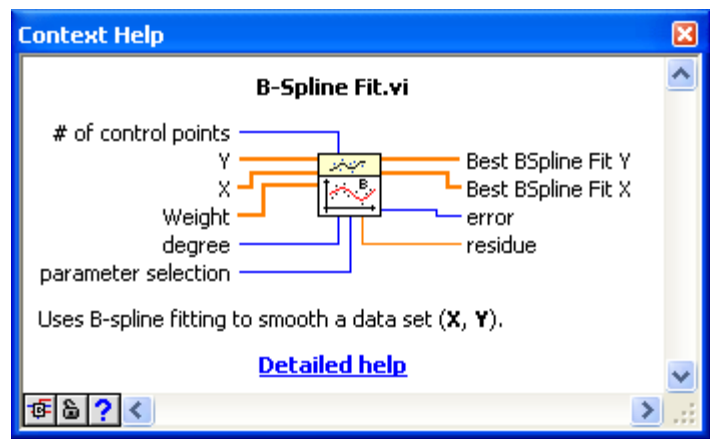


Fitting VIs

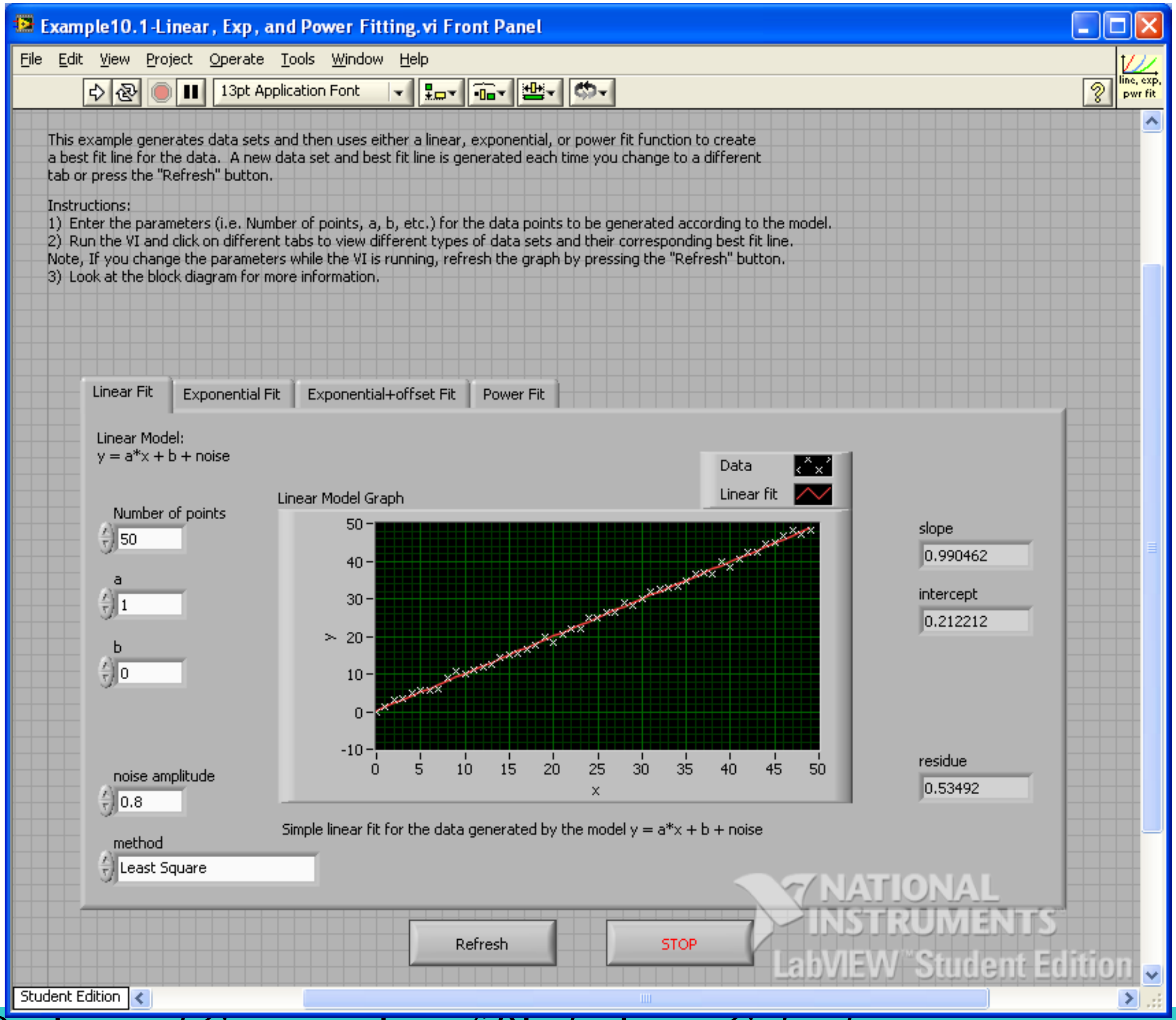

Context Help
Linear Fit.vi
Y
X
Weight
tolerance
method
parameter bounds
Best Linear Fit
slope
intercept
error
residue
Returns the linear fit of a data set (X, Y) using the Least Square, Least Absolute Residual, or Bisquare method.
Detailed help


Context Help
Exponential Fit.vi
Y
X
Weight
tolerance
method
parameter bounds
Best Exponential Fit
amplitude
damping
offset
error
residue
Returns the exponential fit of a data set (X, Y) using the Least Square, Least Absolute Residual, or Bisquare method.
Detailed help

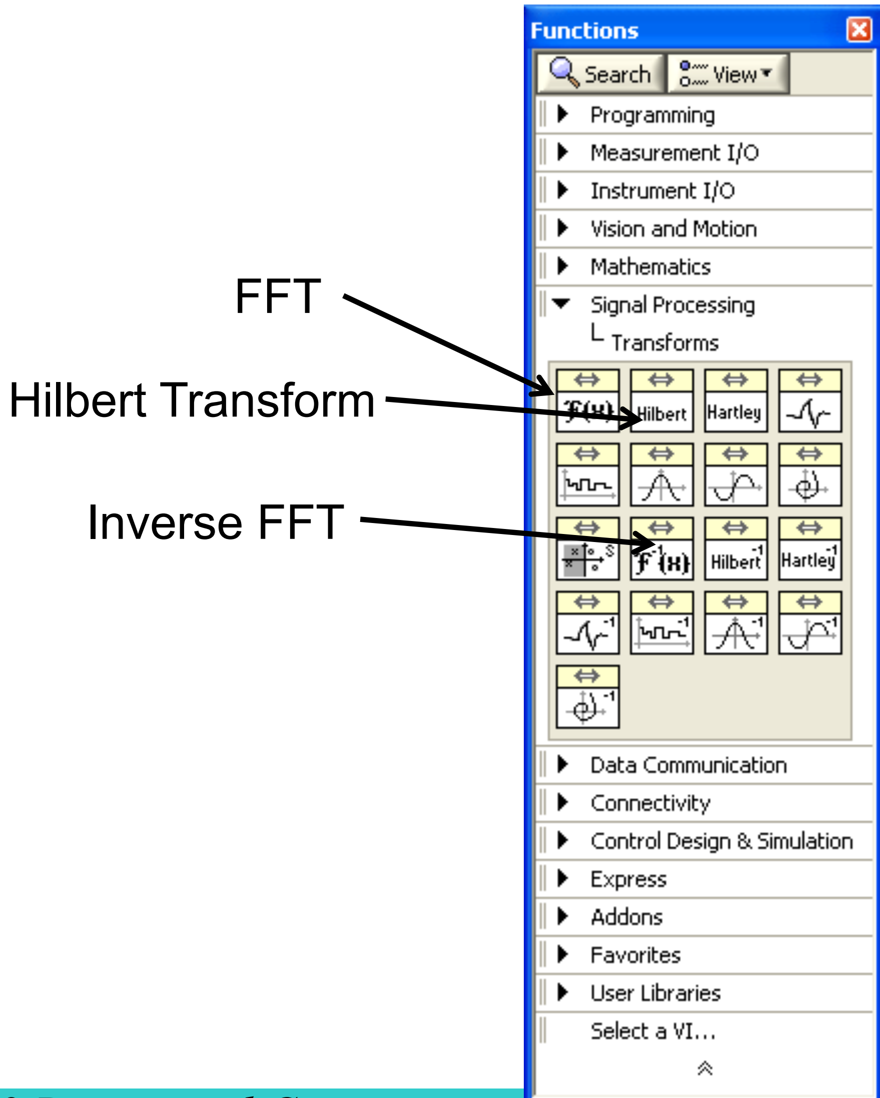

Context Help
Gaussian Peak Fit.vi
initial guess
Y
X
Weight
tolerance
method
parameter bounds
offset
Best Gaussian Fit
amplitude
center
standard deviation
error
residue
Returns the Gaussian fit of a data set (X, Y) using the Least Square, Least Absolute Residual, or Bisquare method.
Detailed help


Context Help
B-Spline Fit.vi
# of control points
Y
X
Weight
degree
parameter selection
Best BSpline Fit Y
Best BSpline Fit X
error
residue
Uses B-spline fitting to smooth a data set (X, Y).
Detailed help

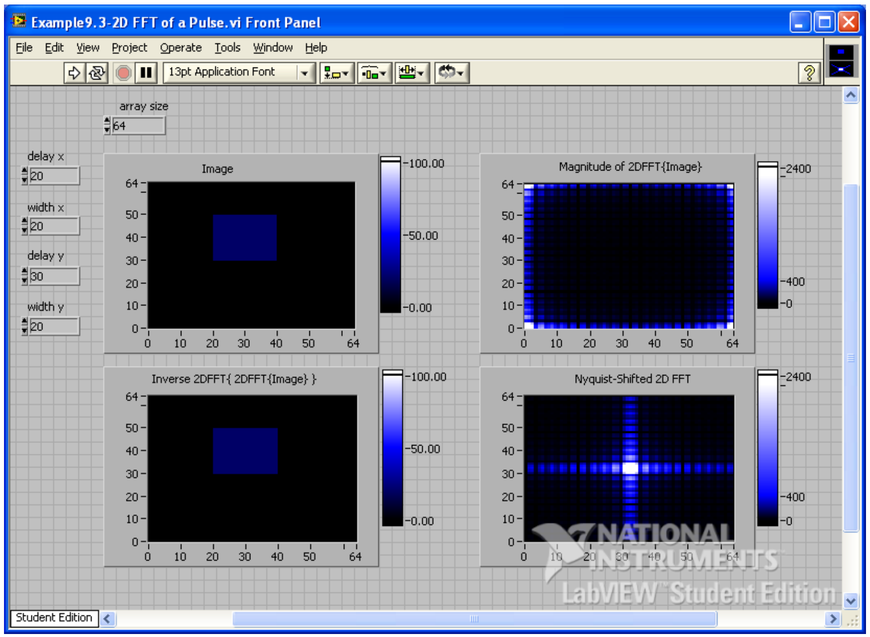
Example – Linear, Exp, and Power Fitting (From LabVIEW Example)

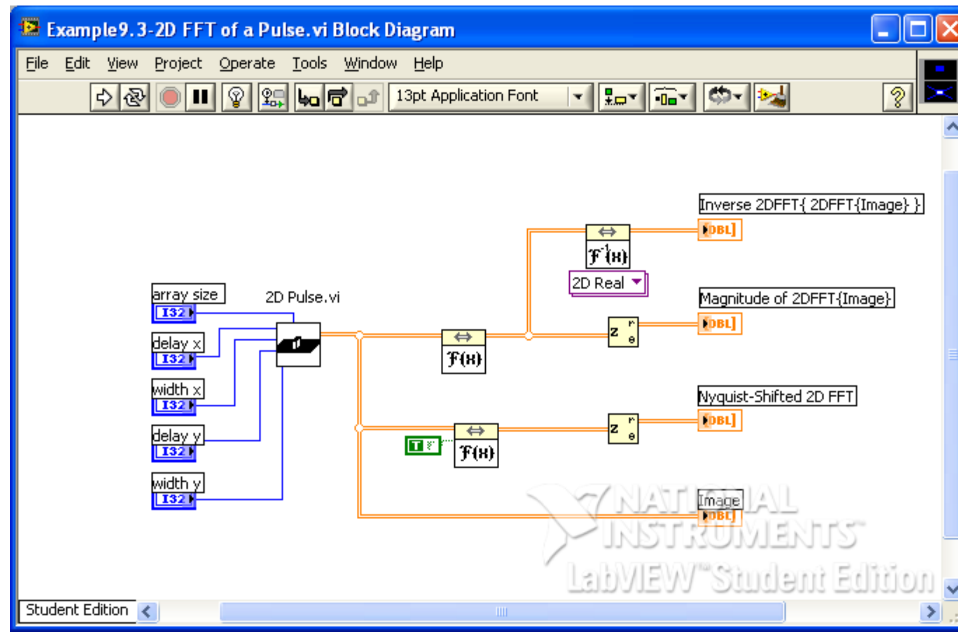


Signal Processing – Transforms



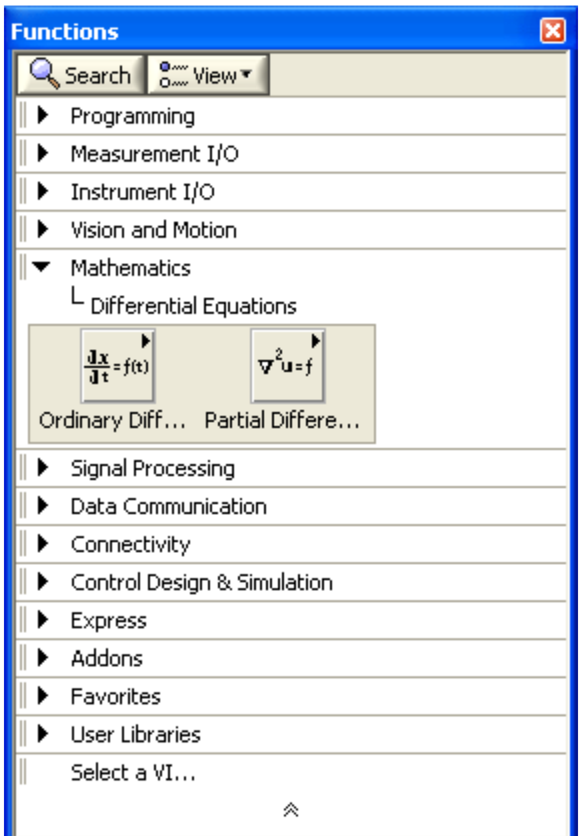
Example – 2D FFT of a Pulse (From LabVIEW Example)

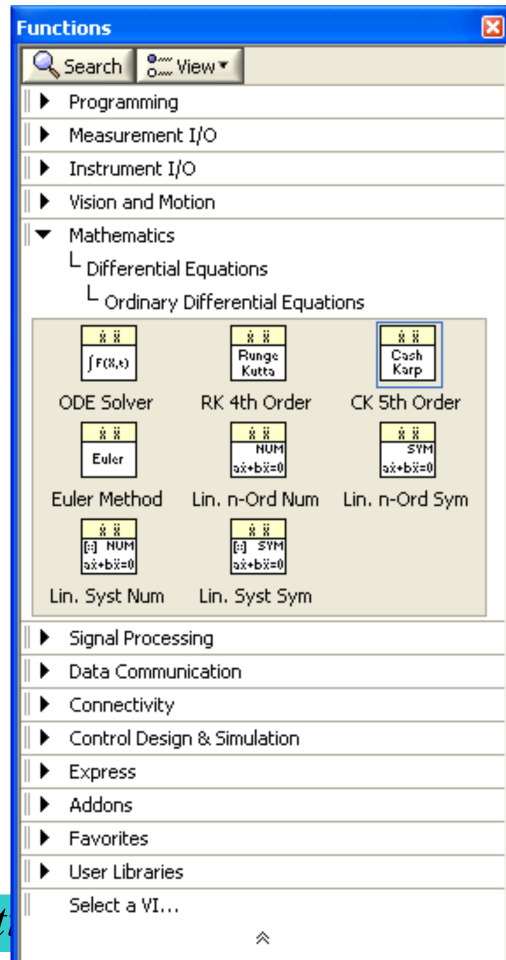


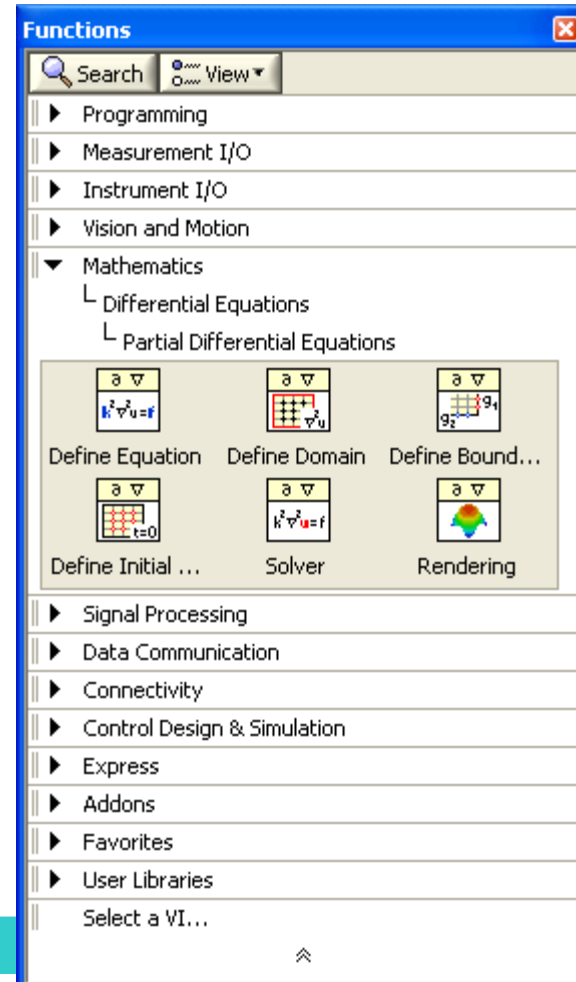


Differential Equations

- LabVIEW can solve ODEs and PDEs numerically.

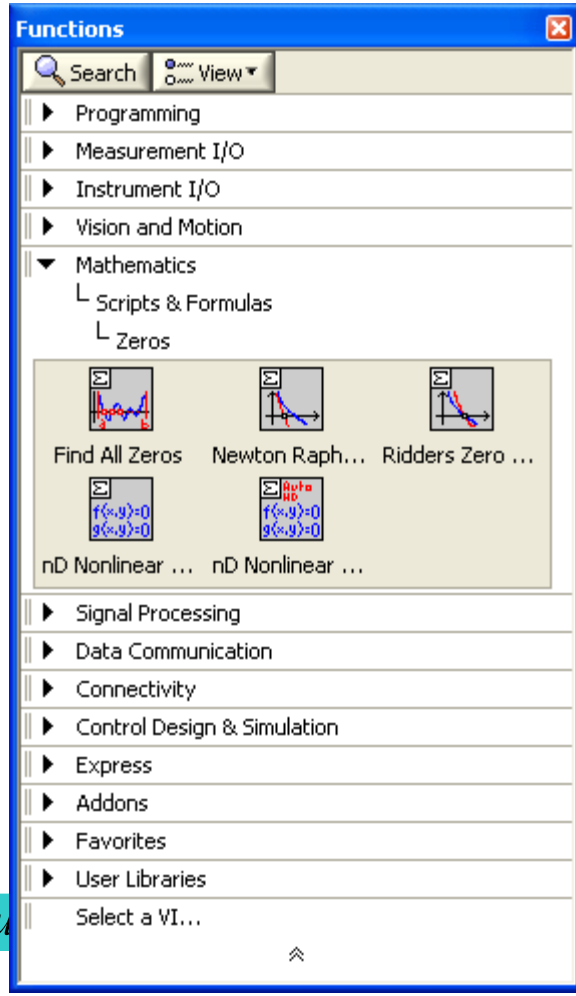






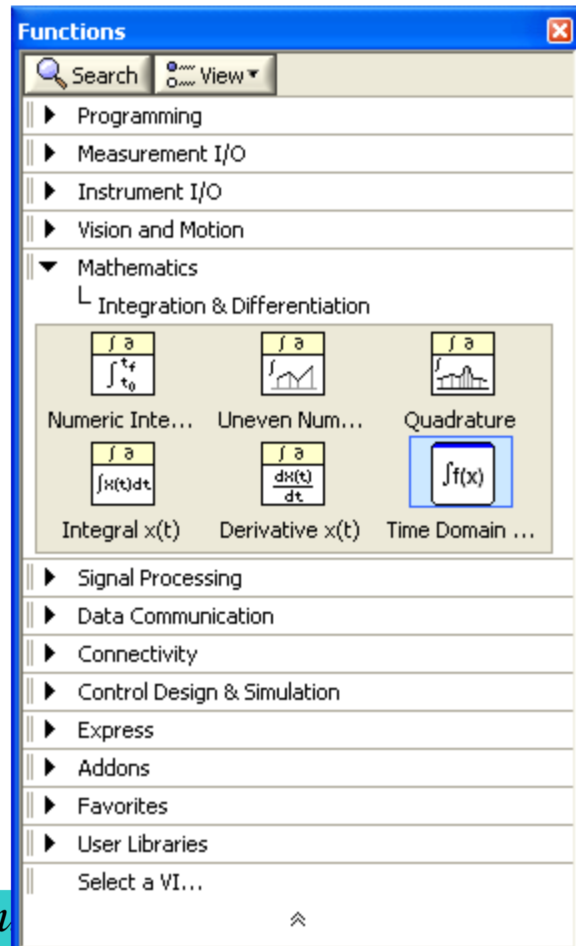
Finding Zeros of Functions

- LabVIEW provides VIs that can be used to compute zeros of functions.



Integration and Differentiation

- LabVIEW provides VIs for integration and differentiation.



Signal Generation

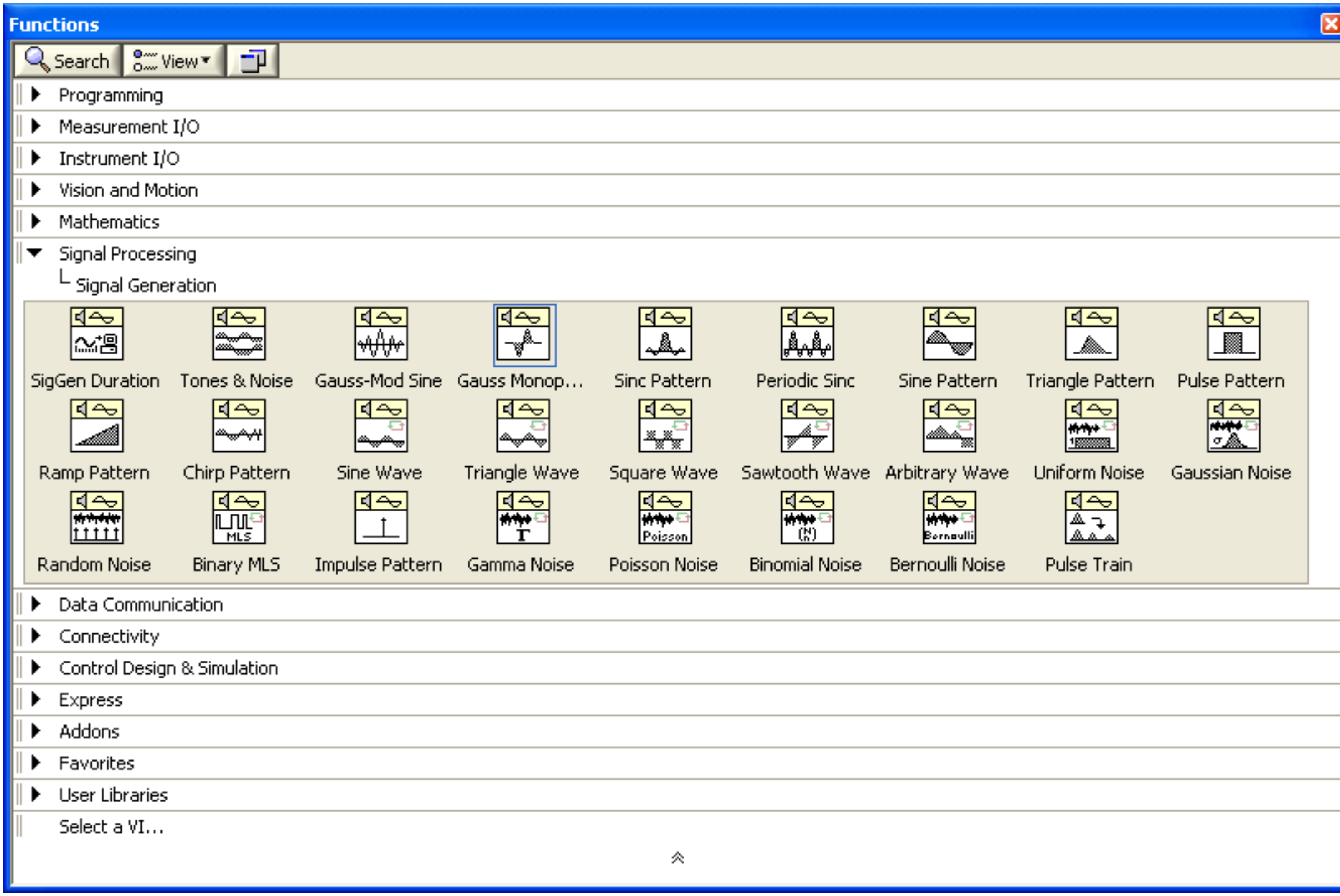
- For testing algorithms and other purposes when real-world signals are not available.
- Signal can be generated by
  - Mathematical equations
  - Arrays of data points
  - Signal generation Vis for common signals

Normalized Frequency

- Also called **digital frequency**.

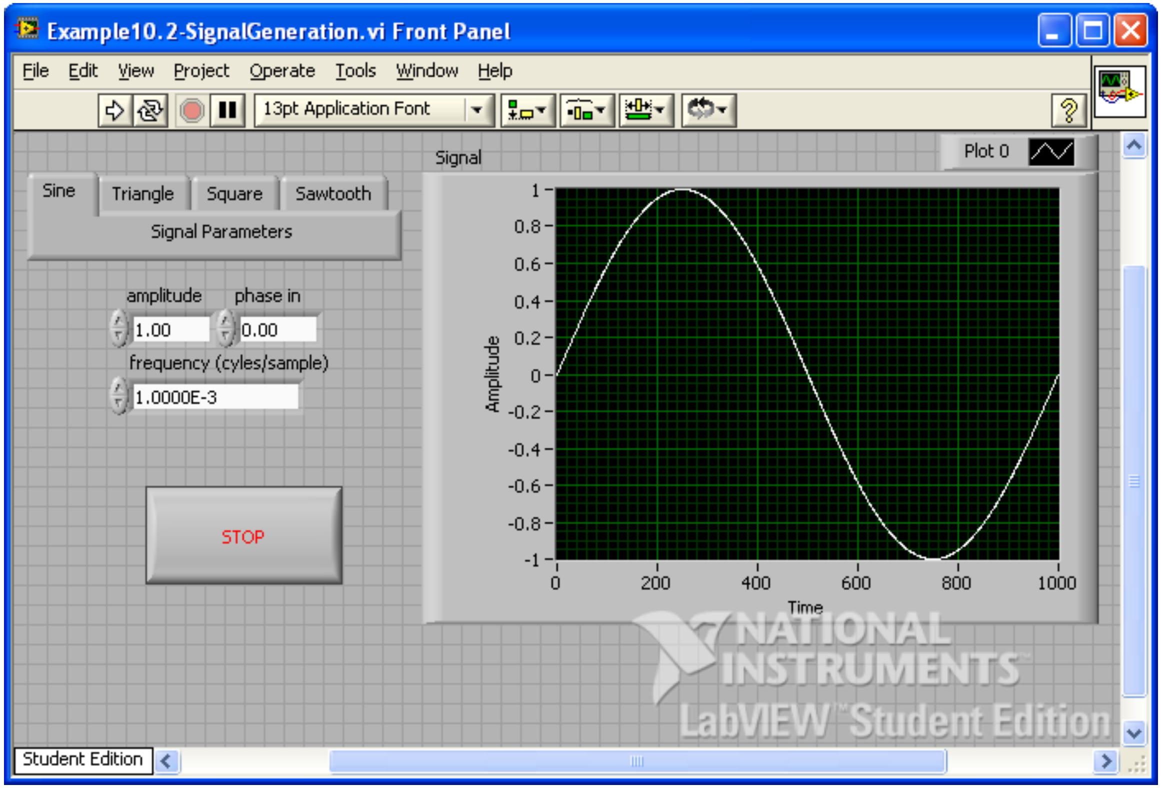
$$f = \text{Normalized frequency} = \frac{\text{Analog frequency}}{\text{Sampling frequency}}$$

Signal Generation VIs



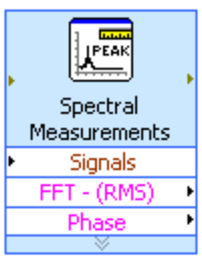
Example – Signal Generation

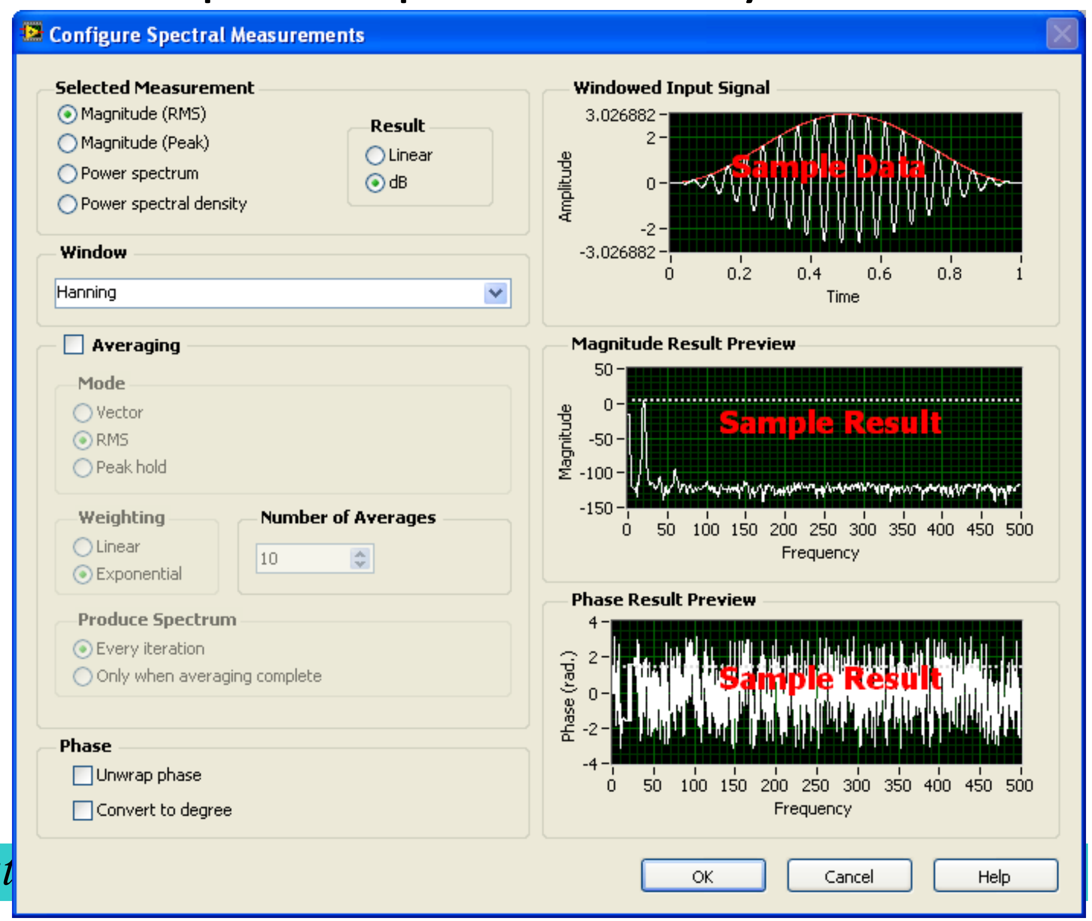
- Generate sine, triangle, square, and sawtooth signal.



Signal Processing – Spectral Measurements Express VI

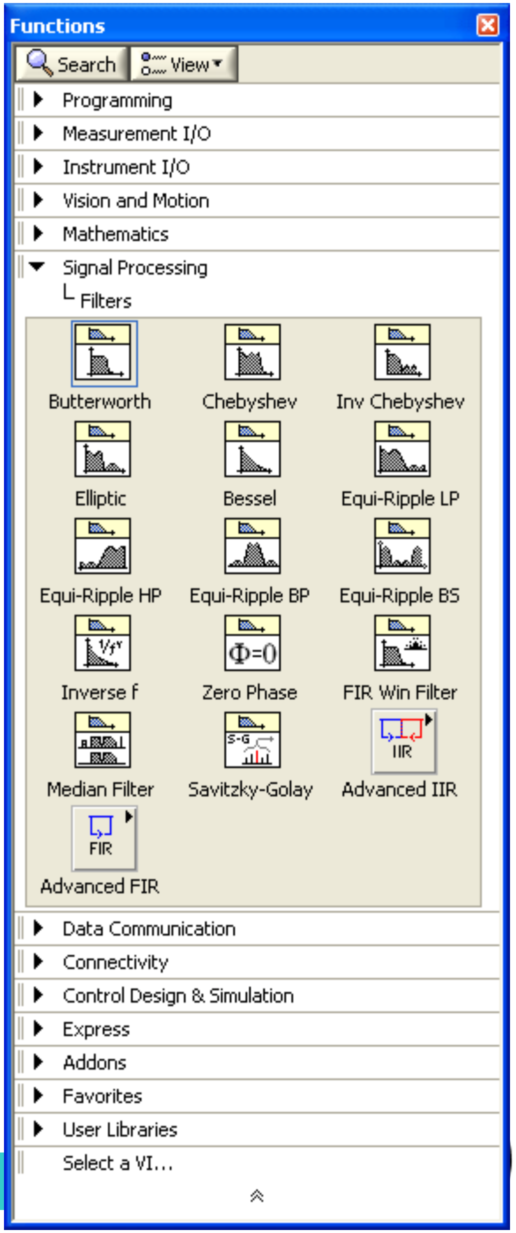
- The spectral measurements Express VI performs spectral measurements, such as spectral power density.



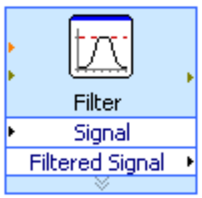


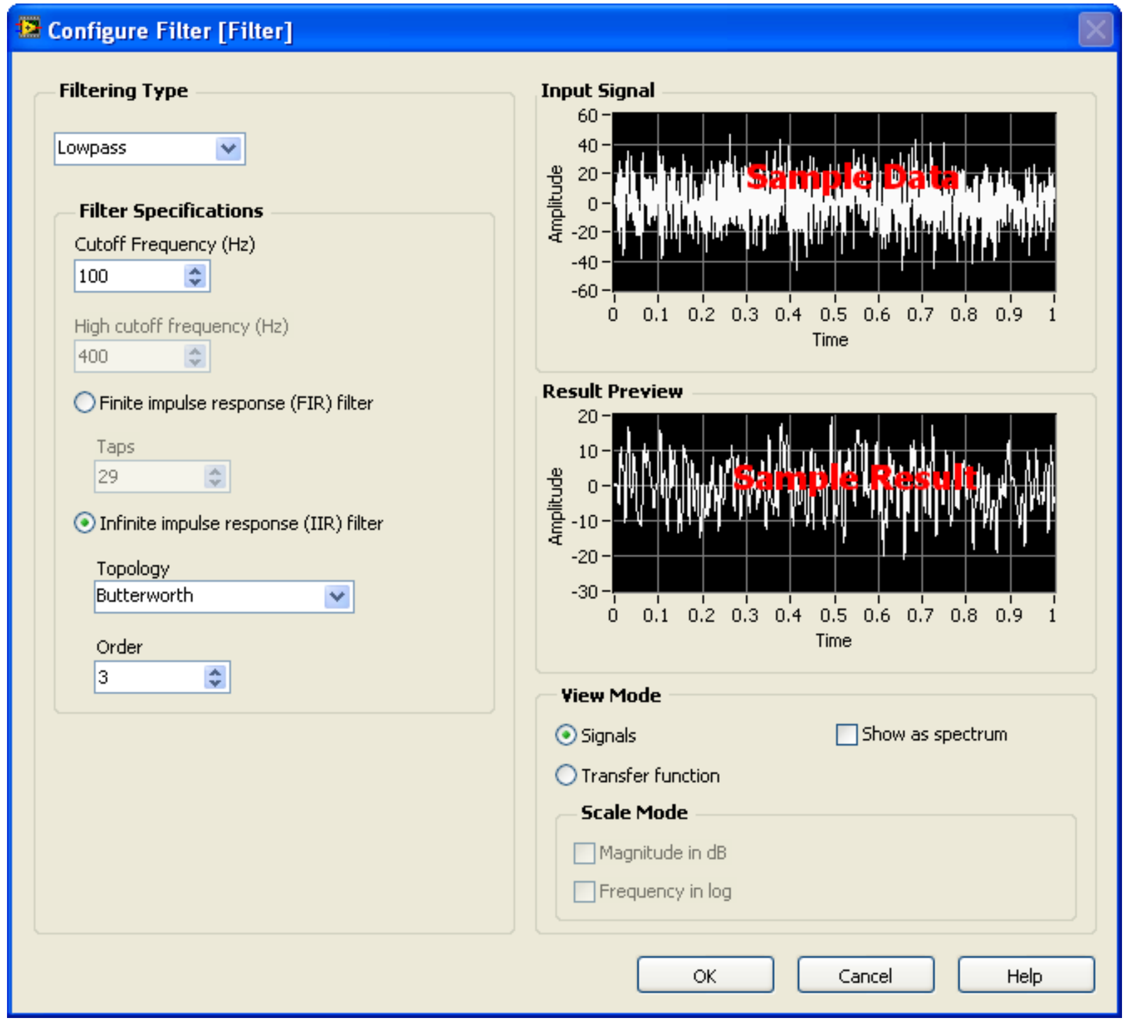
Signal Processing – Filtering

- LabVIEW can be used to implement digital filters
  - Finite impulse response (FIR) filters
  - Infinite impulse response (IIR) filters

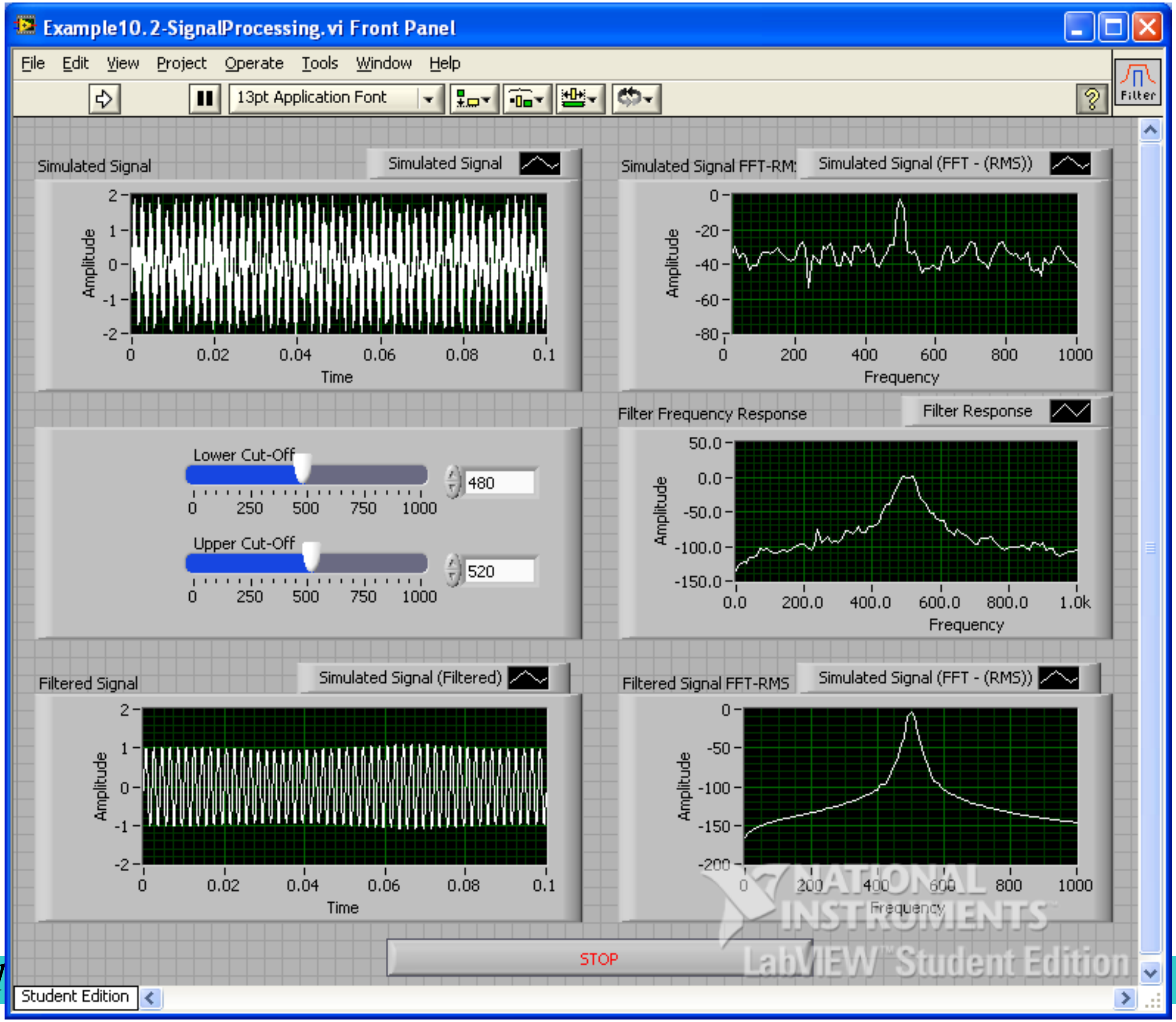


Signal Processing – Filter Express VI





Example – Signal Processing



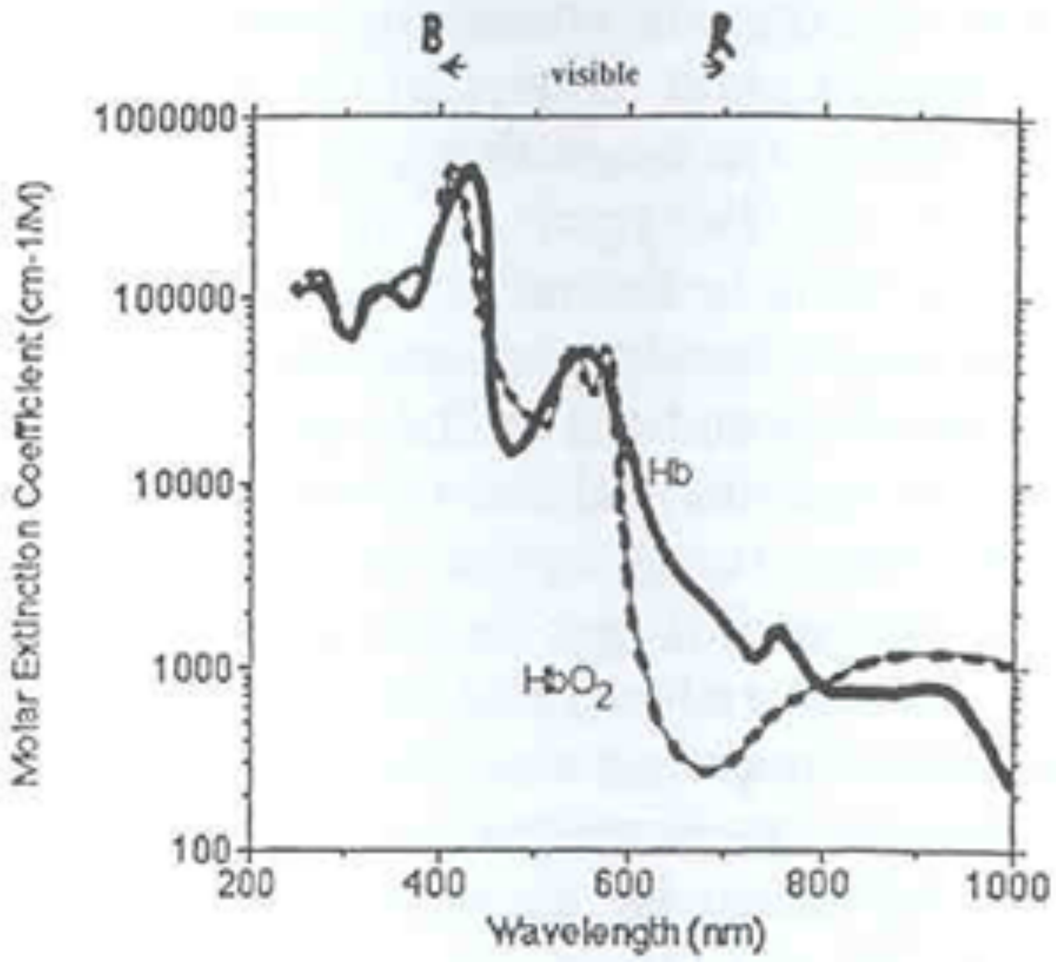
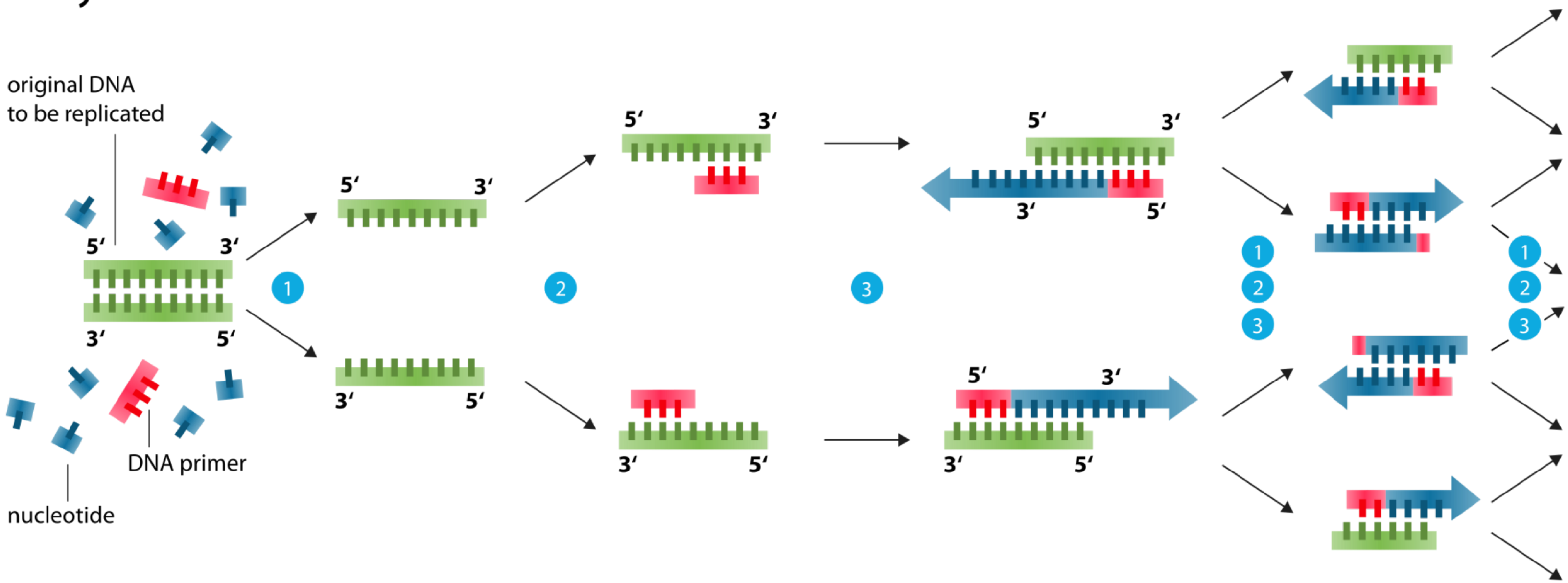


Figure 11 UV-visible Spectra of oxy- (HbO_2) and deoxyhaemoglobin (Hb)

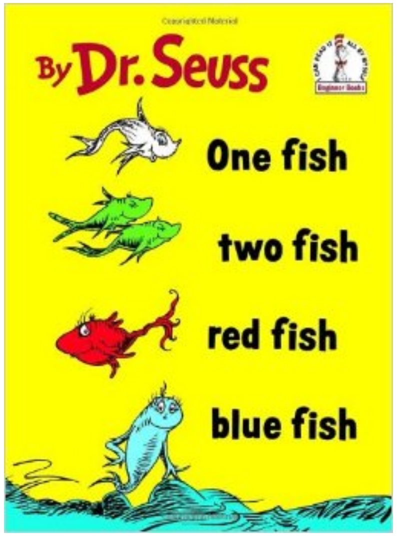
Polymerase chain reaction - PCR

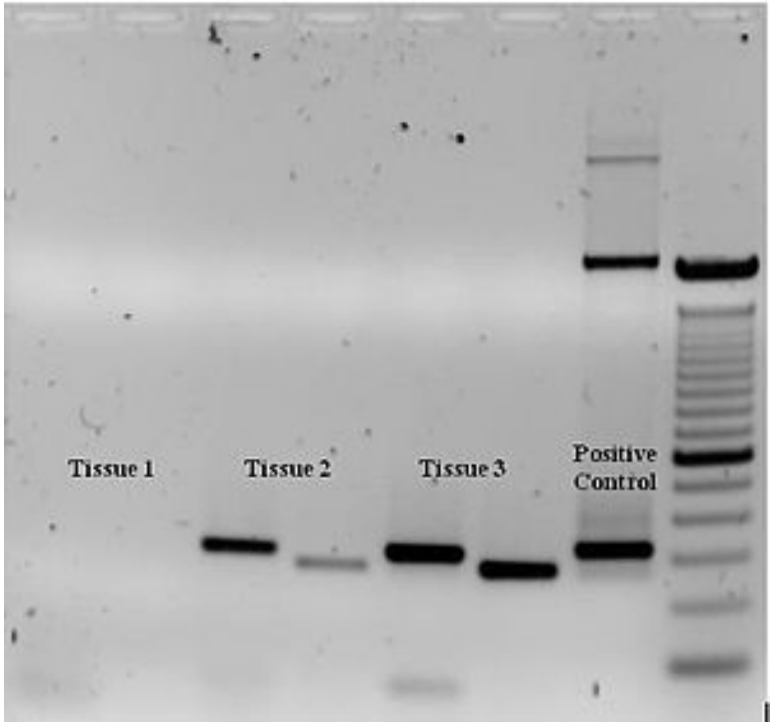


1 **Denaturation** at 94-96°C

2 **Annealing** at ~68°C

3 **Elongation** at ca. 72 °C





Ethidium bromide-stained PCR products after gel electrophoresis. Two sets of primers were used to amplify a target sequence from three different tissue samples. No amplification is present in sample #1; DNA bands in sample #2 and #3 indicate successful amplification of the target sequence. The gel also shows a positive control, and a DNA ladder containing DNA fragments of defined length for sizing the bands in the experimental PCRs.