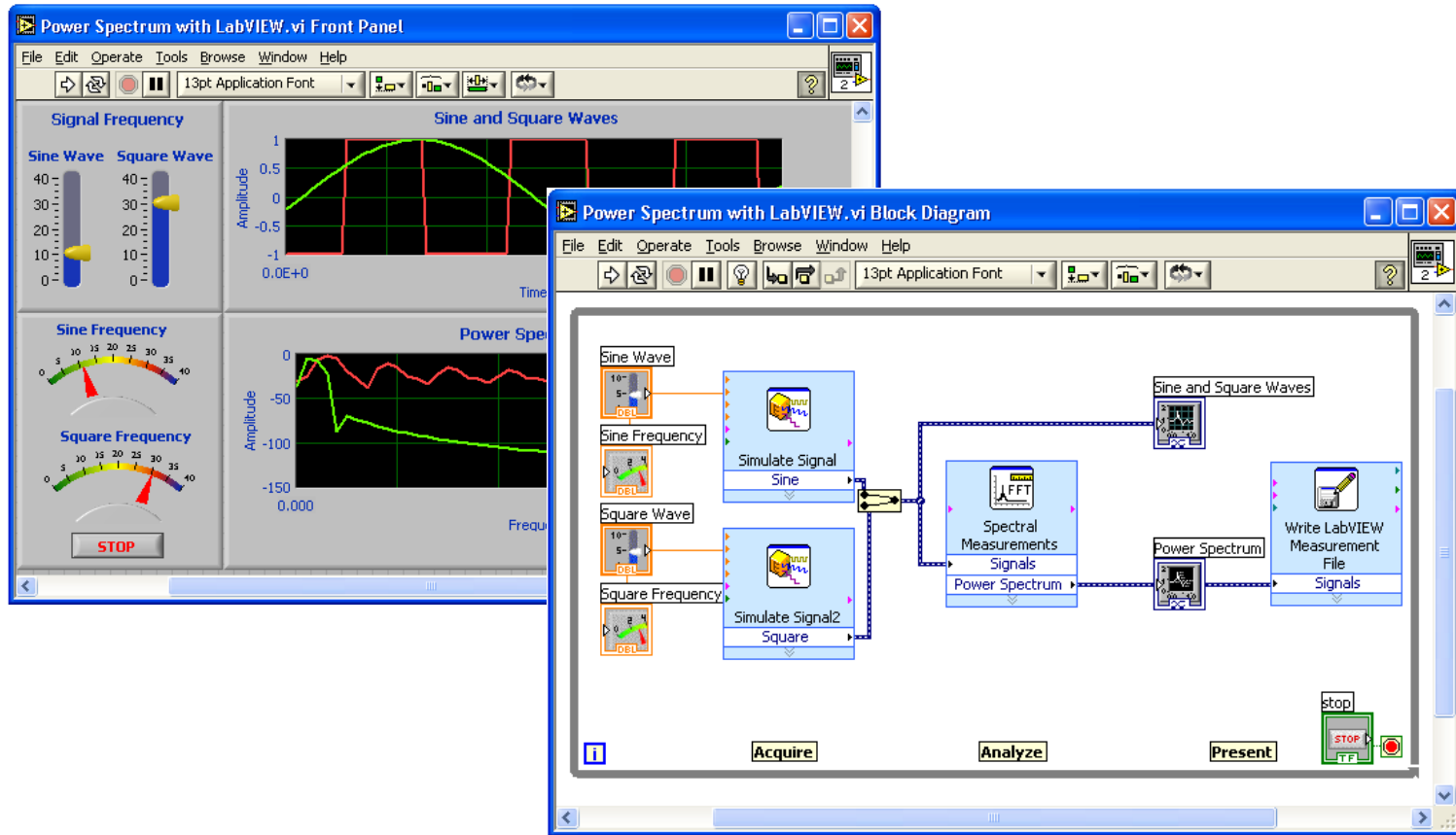


Virtual Instrumentation With LabVIEW



Course Goals

- Understand the components of a Virtual Instrument
- Introduce LabVIEW and common LabVIEW functions
- Create a subroutine in LabVIEW
- Work with Arrays, Clusters, and Structures
- Develop in Basic Programming Architectures

Section I

- LabVIEW terms
- Components of a LabVIEW application
- LabVIEW programming tools
- Creating an application in LabVIEW

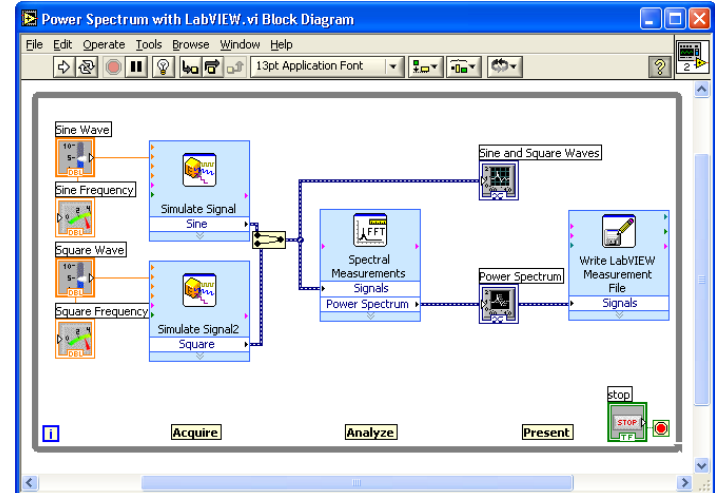
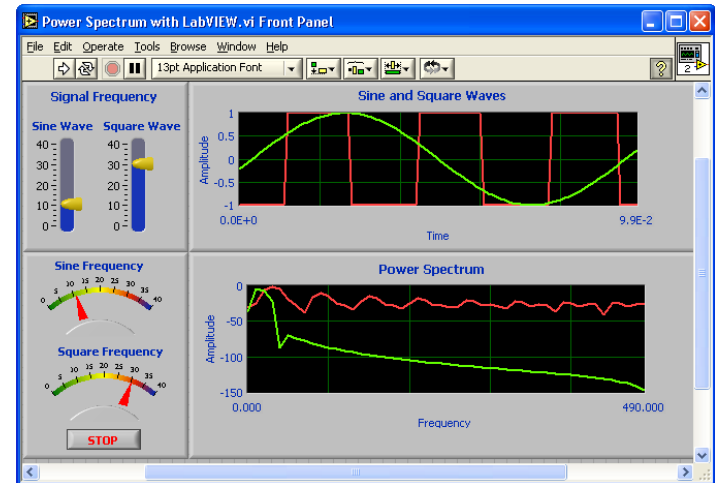
LabVIEW Programs Are Called Virtual Instruments (VIs)

Front Panel

- Controls = Inputs
- Indicators = Outputs

Block Diagram

- Accompanying “program” for front panel
- Components “wired” together



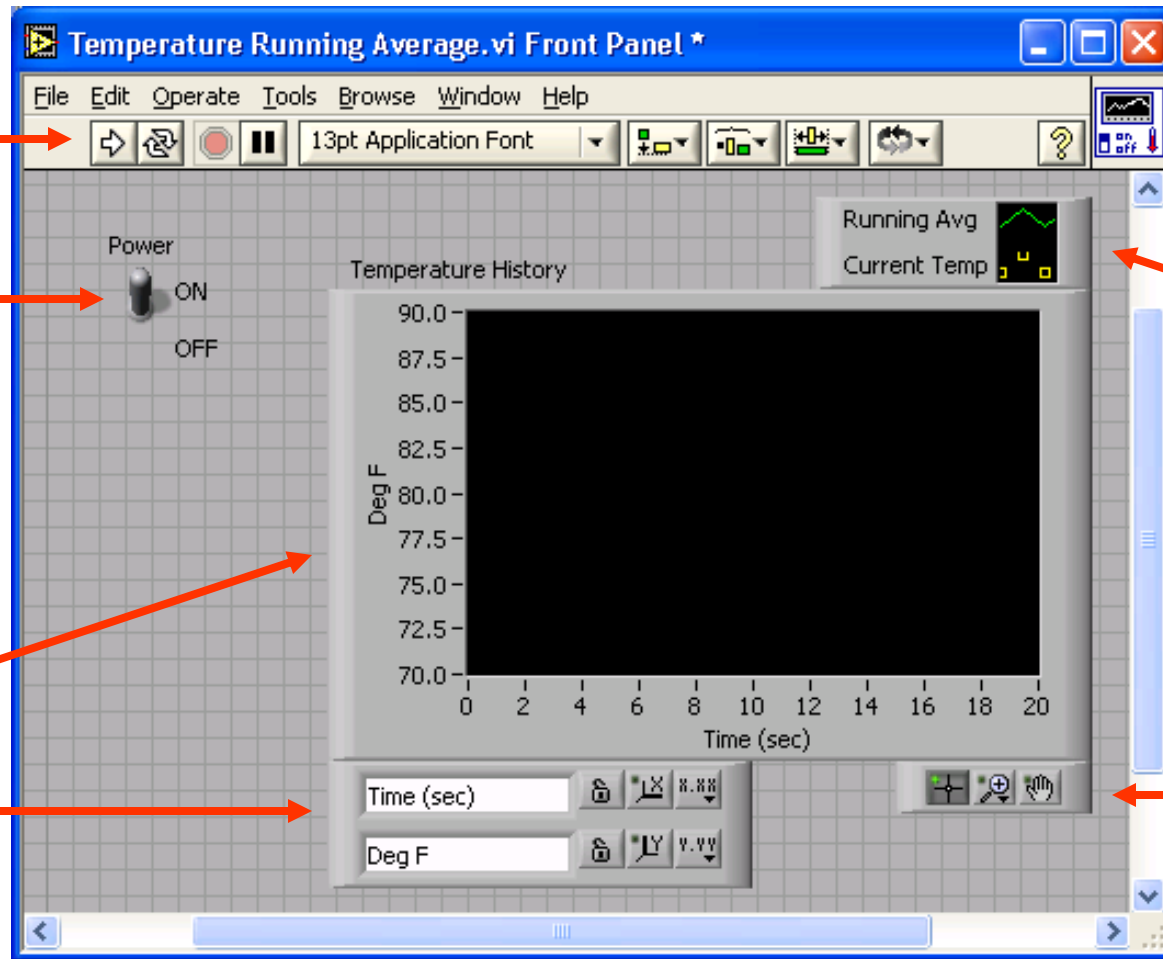
VI Front Panel

Front Panel
Toolbar

Boolean
Control

Waveform
Graph

Plot
Legend

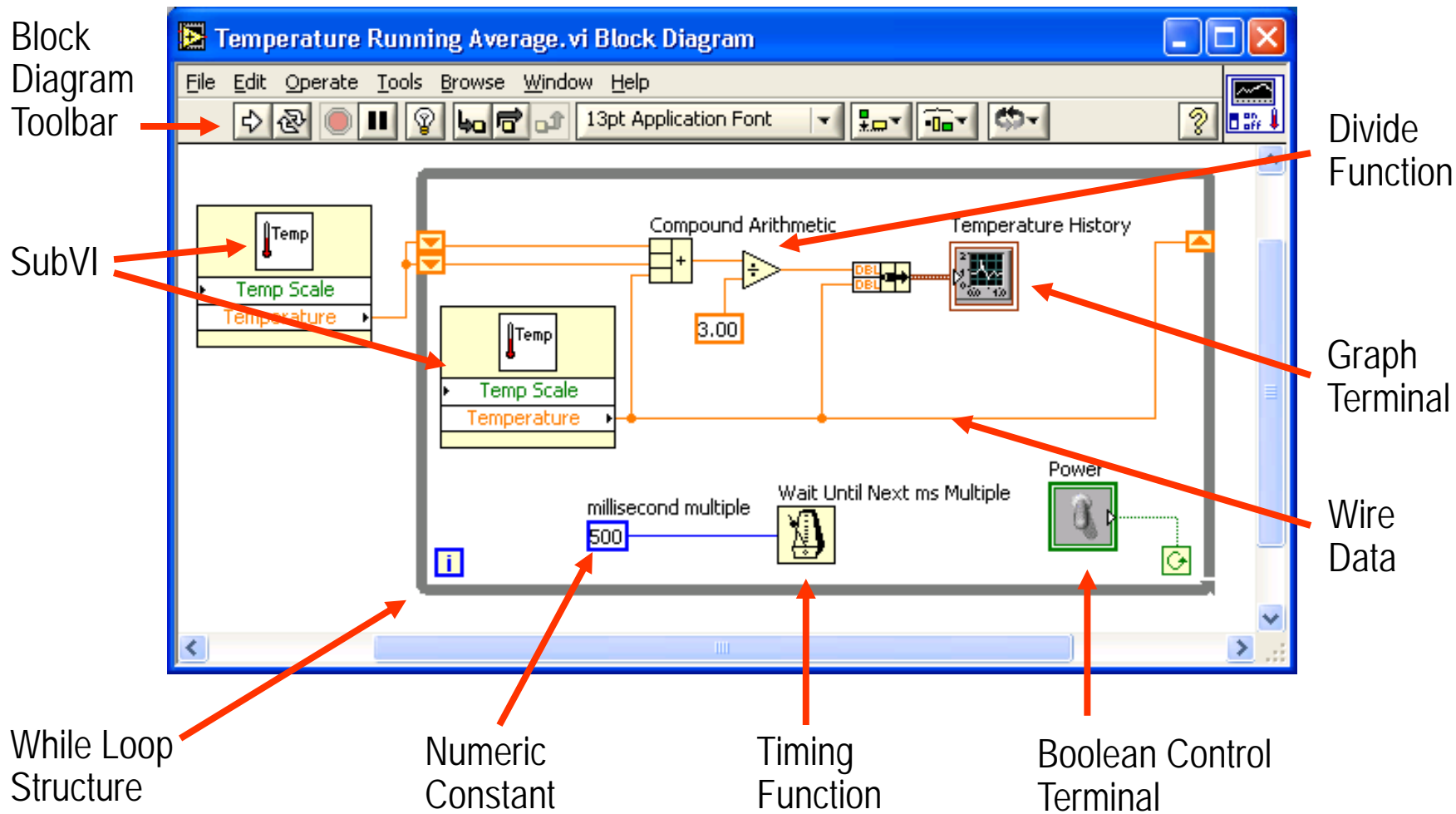


Icon

Graph
Legend

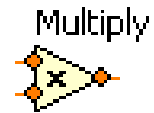
Scale
Legend

VI Block Diagram

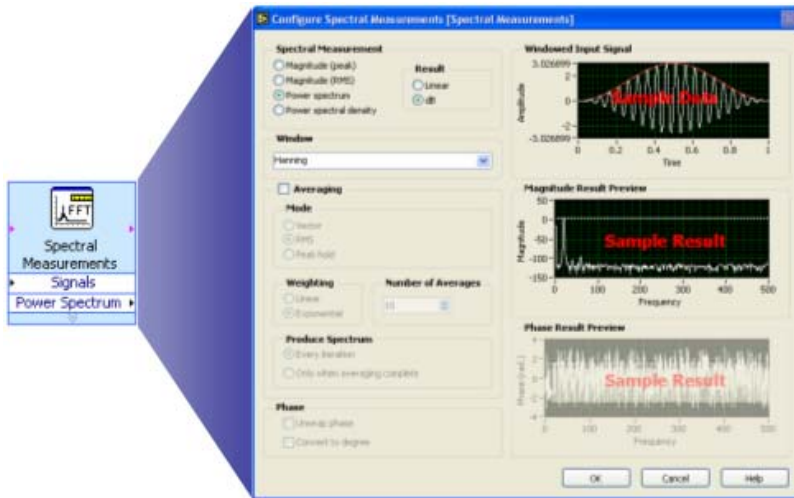


Express VIs, VIs and Functions

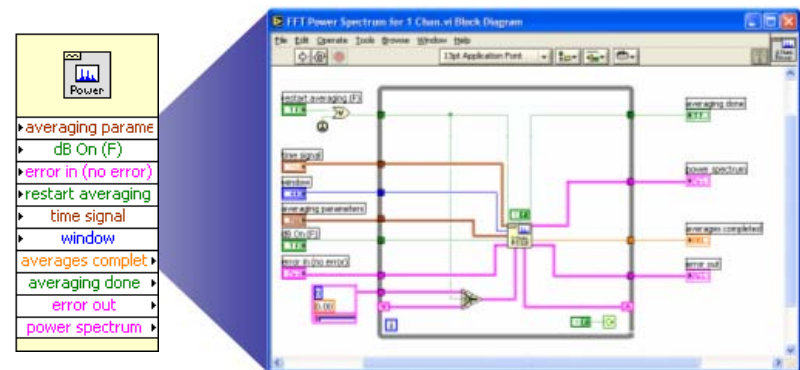
- Express VIs: interactive VIs with configurable dialog page
- Standard VIs: modularized VIs customized by wiring
- Functions: fundamental operating elements of LabVIEW; no front panel or block diagram



Function



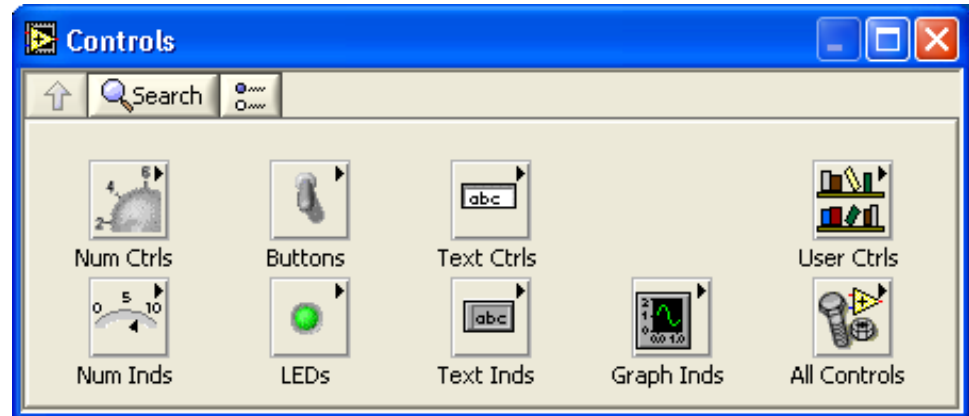
Express VI



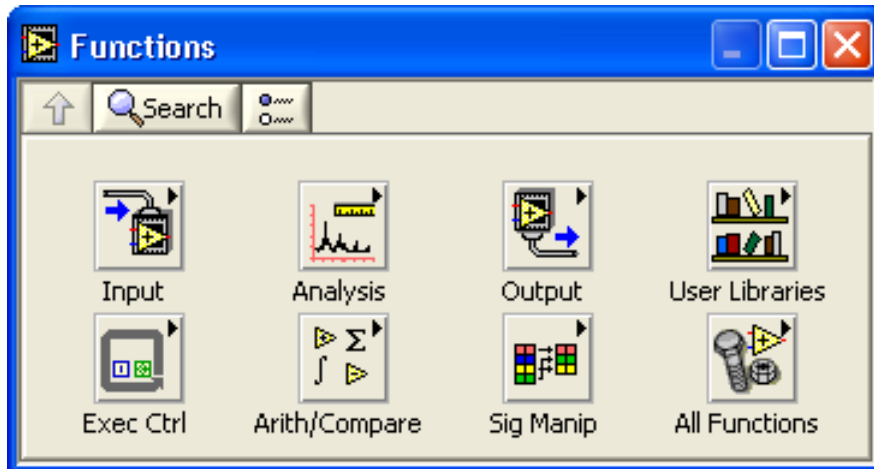
Standard VI

Controls and Functions Palettes

Controls Palette (Front Panel Window)



Functions Palette (Block Diagram Window)



Tools Palette



- **Floating Palette**
- **Used to operate and modify front panel and block diagram objects.**



Automatic Selection Tool



Operating Tool



Scrolling Tool



Positioning/Resizing Tool



Breakpoint Tool



Labeling Tool



Probe Tool



Wiring Tool



Color Copy Tool



Shortcut Menu Tool



Coloring Tool

Status Toolbar



Run Button



Continuous Run Button



Abort Execution



Pause/Continue Button

13pt Application Font

Text Settings



Align Objects



Distribute Objects



Reorder



Resize front panel objects



Execution Highlighting Button



Step Into Button



Step Over Button

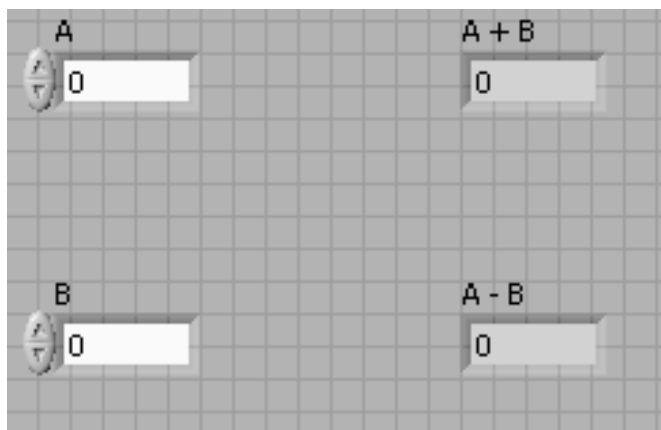


Step Out Button

Additional Buttons on the Diagram Toolbar

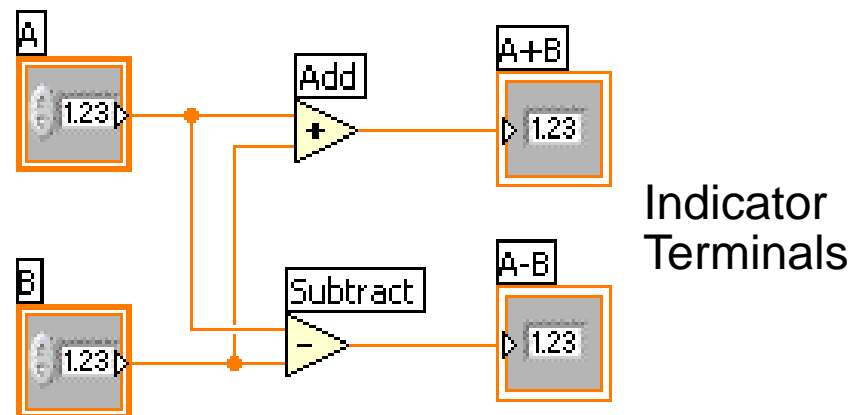
Creating a VI

Front Panel Window



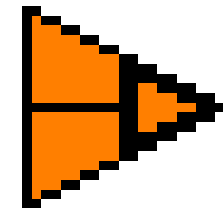
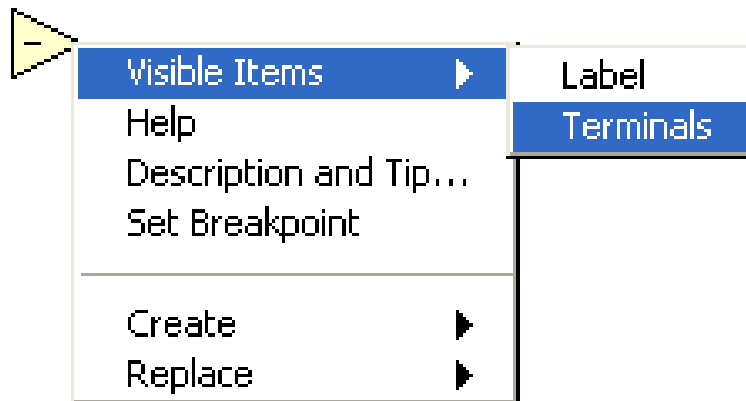
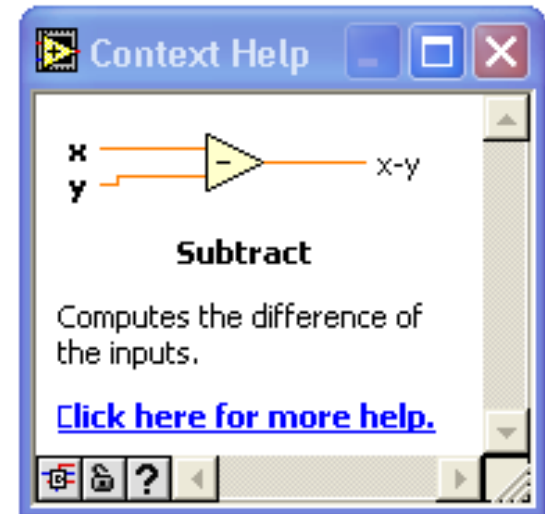
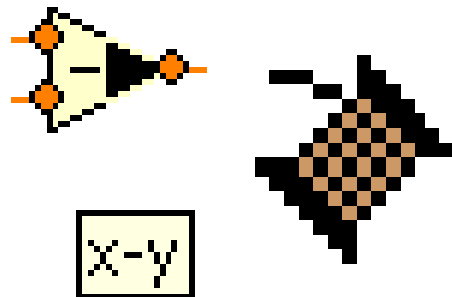
Control
Terminals

Block Diagram Window



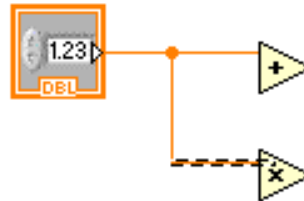
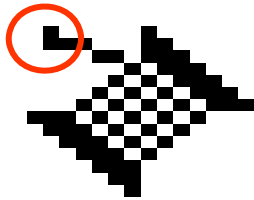
Indicator
Terminals

Creating a VI – Block Diagram

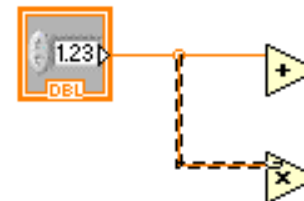


Wiring Tips – Block Diagram

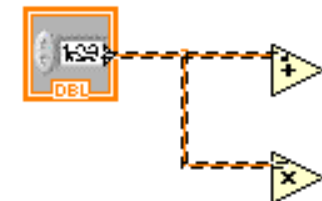
Wiring “Hot Spot”



single-click



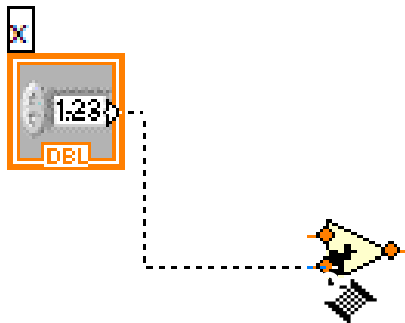
double-click



triple-click

Click To Select Wires

Use Automatic Wire Routing

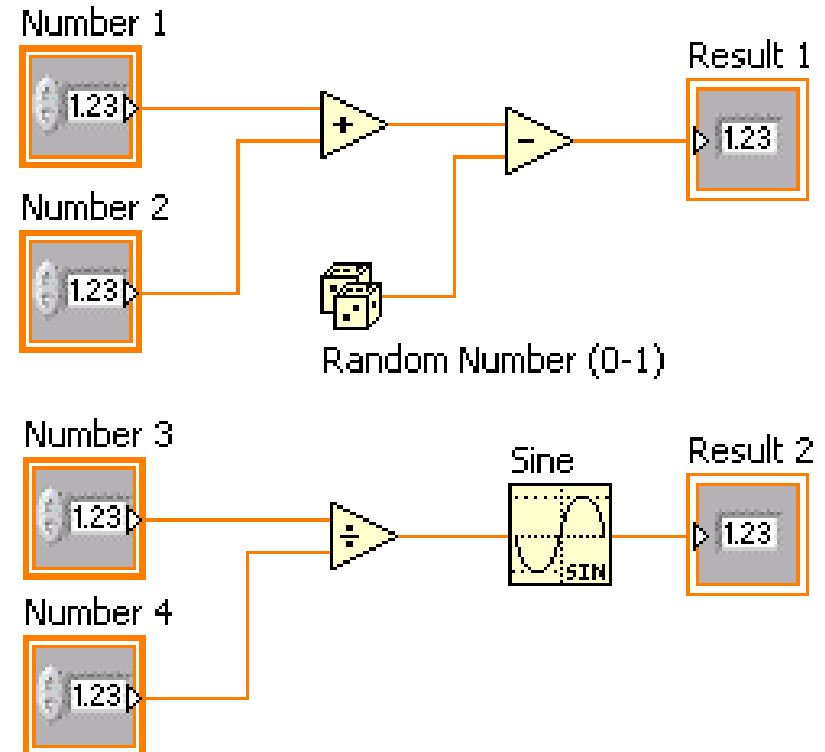


Clean Up Wiring



Dataflow Programming

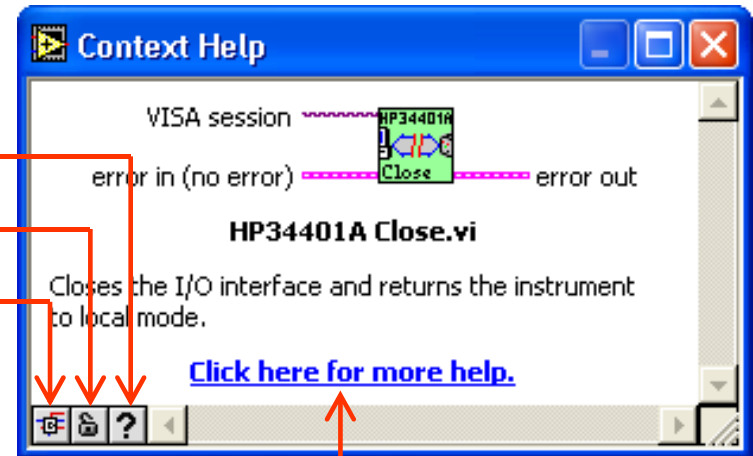
- Block diagram executes dependent on the flow of data; block diagram does NOT execute left to right
- Node executes when data is available to ALL input terminals
- Nodes supply data to all output terminals when done



Help Options

Context Help

- Online help
- Lock help
- Simple/Complex Diagram help
- Ctrl + H



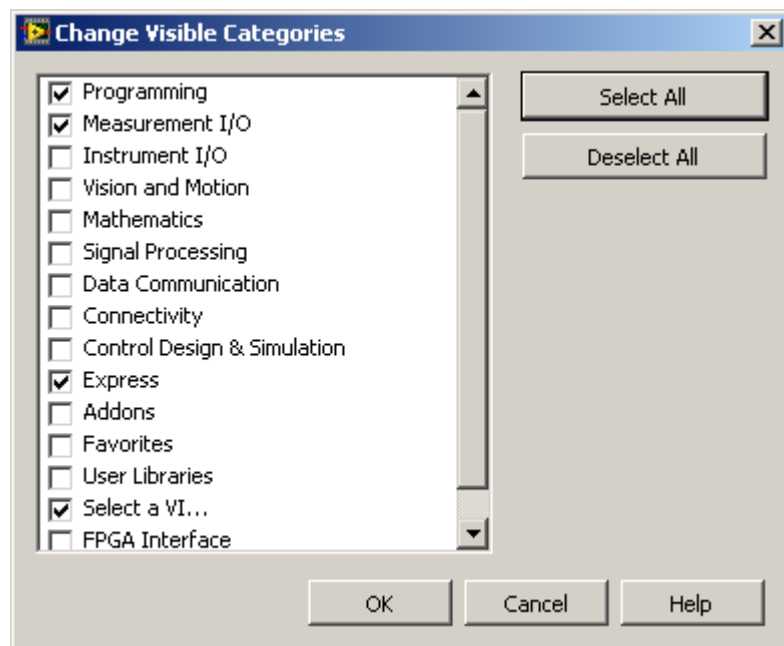
Online reference

- All menus online
- Pop up on functions in diagram to access online info directly

Customize LabVIEW

- Launch LabVIEW and create a Blank VI.
- Set Up Programming Palette
 - Click on **Window -> Show Block Diagram**
 - Right Click on the blank white screen to bring up the functions palette.
 - Click **Search** - this takes a minute the first time
 - Click **View -> Change Visible Categories**

Customize LabVIEW (cont.)

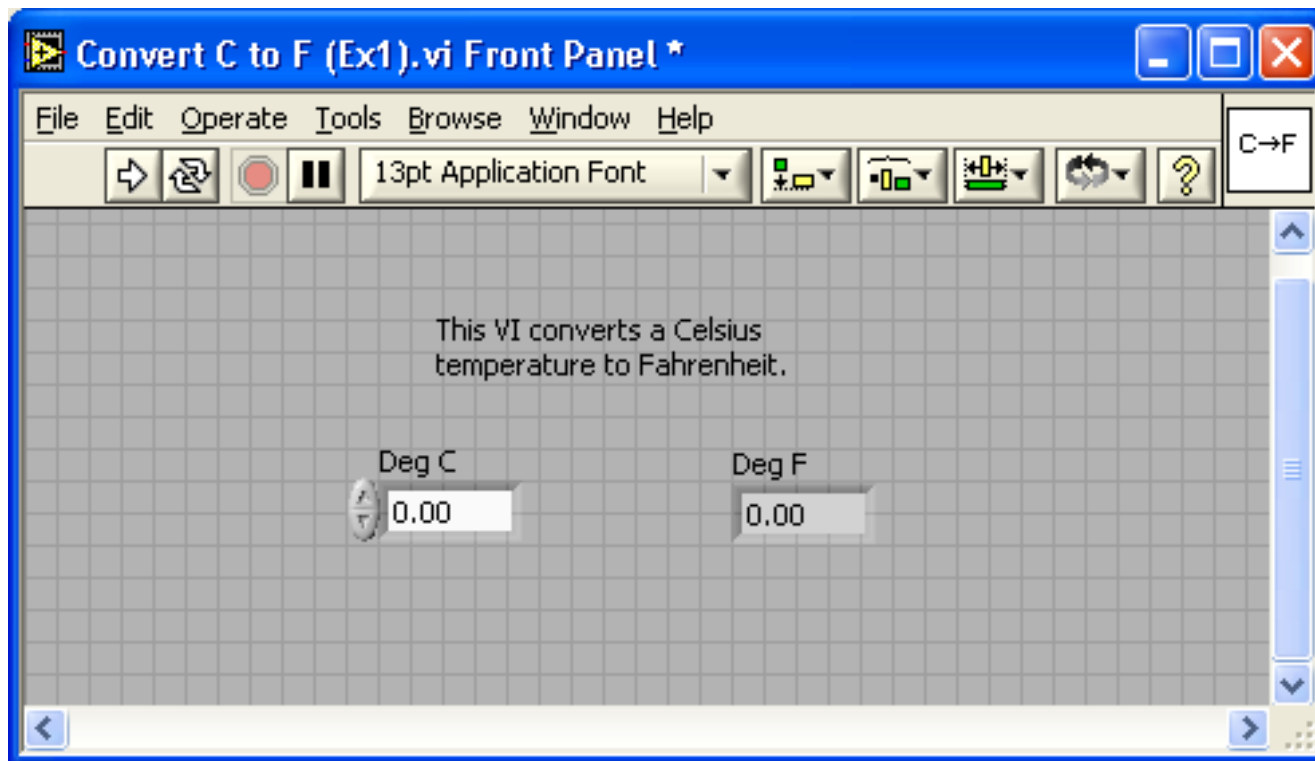


- Check Programming, Measurement I/O, Express, and Select a VI.... Click OK

Customize LabVIEW (cont.)

- Set Options
 - Click on **Tools** -> **Options...**
 - Click on Block Diagram
 - Uncheck **Enable automatic wire routing**
 - Uncheck **Place front panel terminals as icons**
 - Click on Environment
 - Uncheck **Maximum undo steps per VI** -> **Use default**
 - Set **Maximum undo steps per VI** to 99
 - Click **OK**

Exercise 1 - Convert °C to °F



Debugging Techniques

- **Finding Errors**



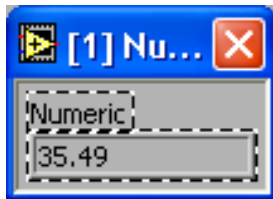
Click on broken Run button
Window showing error appears

- **Execution Highlighting**



Click on Execution Highlighting button; data flow is animated using bubbles. Values are displayed on wires.

- **Probe**

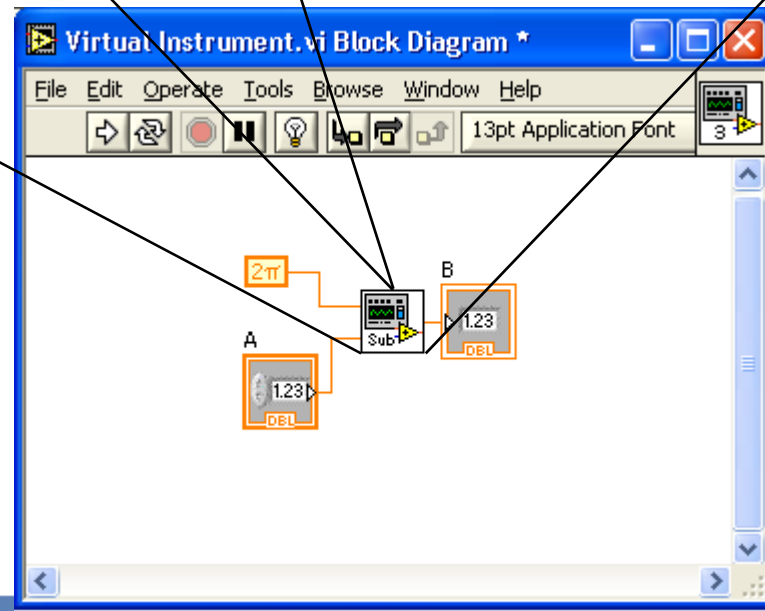
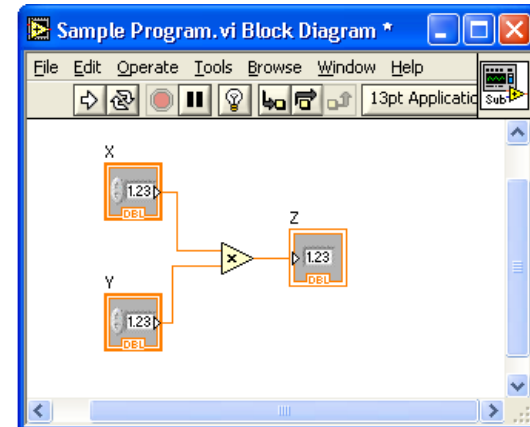
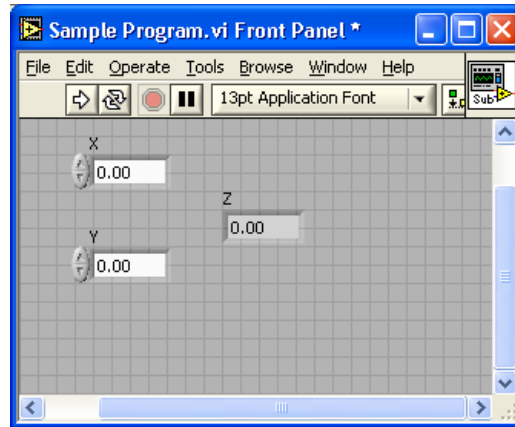


Right-click on wire to display probe and it shows data as it flows through wire segment



You can also select Probe tool from Tools palette and click on wire

Section II – SubVIs

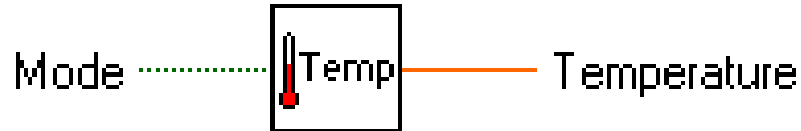
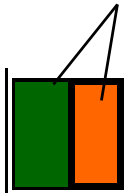
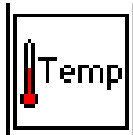


- What is a subVI?
- Making an icon and connector for a subVI
- Using a VI as a subVI

SubVIs

- A SubVI is a VI that can be used within another VI
- Similar to a subroutine
- Advantages
 - Modular
 - Easier to debug
 - Don't have to recreate code
 - Require less memory

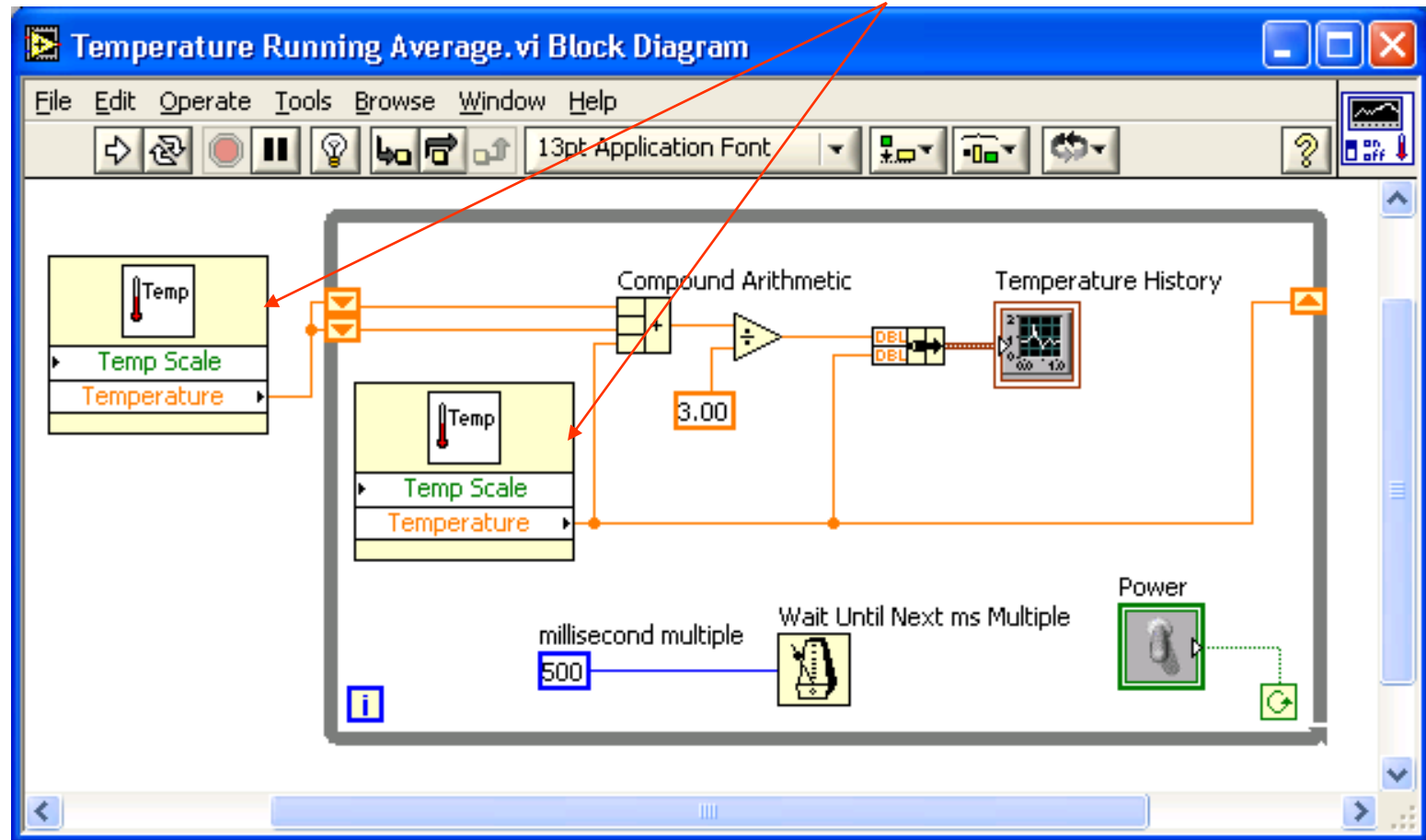
Icon and Connector



- An icon represents a VI in other block diagrams
- A connector shows available terminals for data transfer

SubVIs

Sub VIs

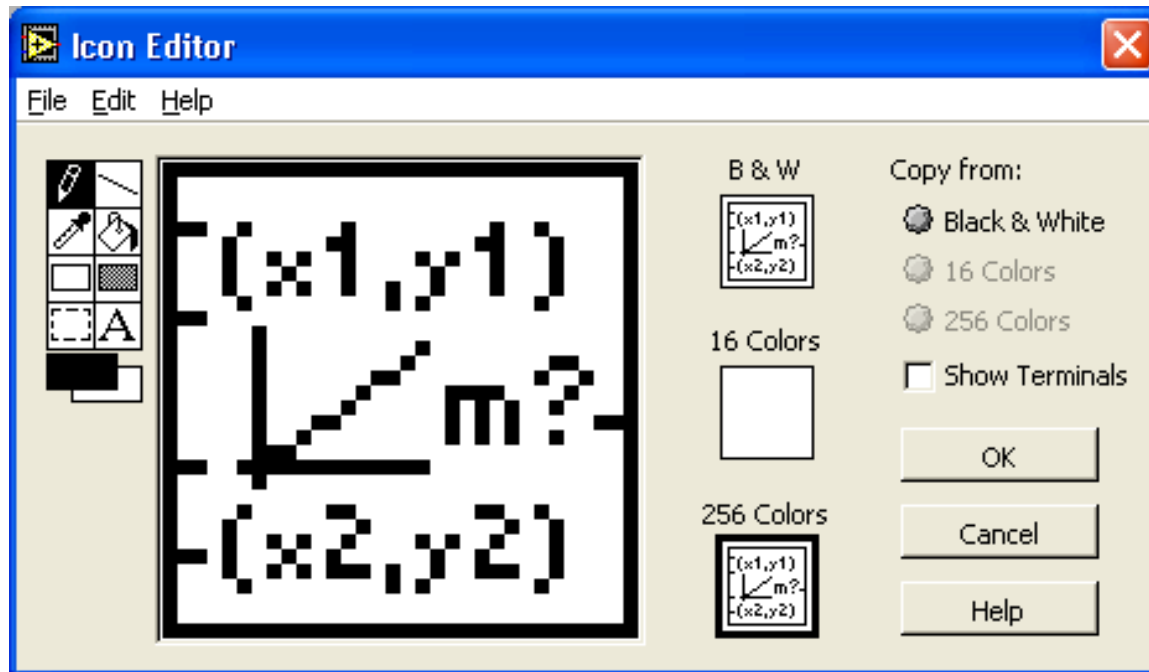


Steps to Create a SubVI

- Create the Icon
- Create the Connector
- Assign Terminals
- Save the VI
- Insert the VI into a Top Level VI

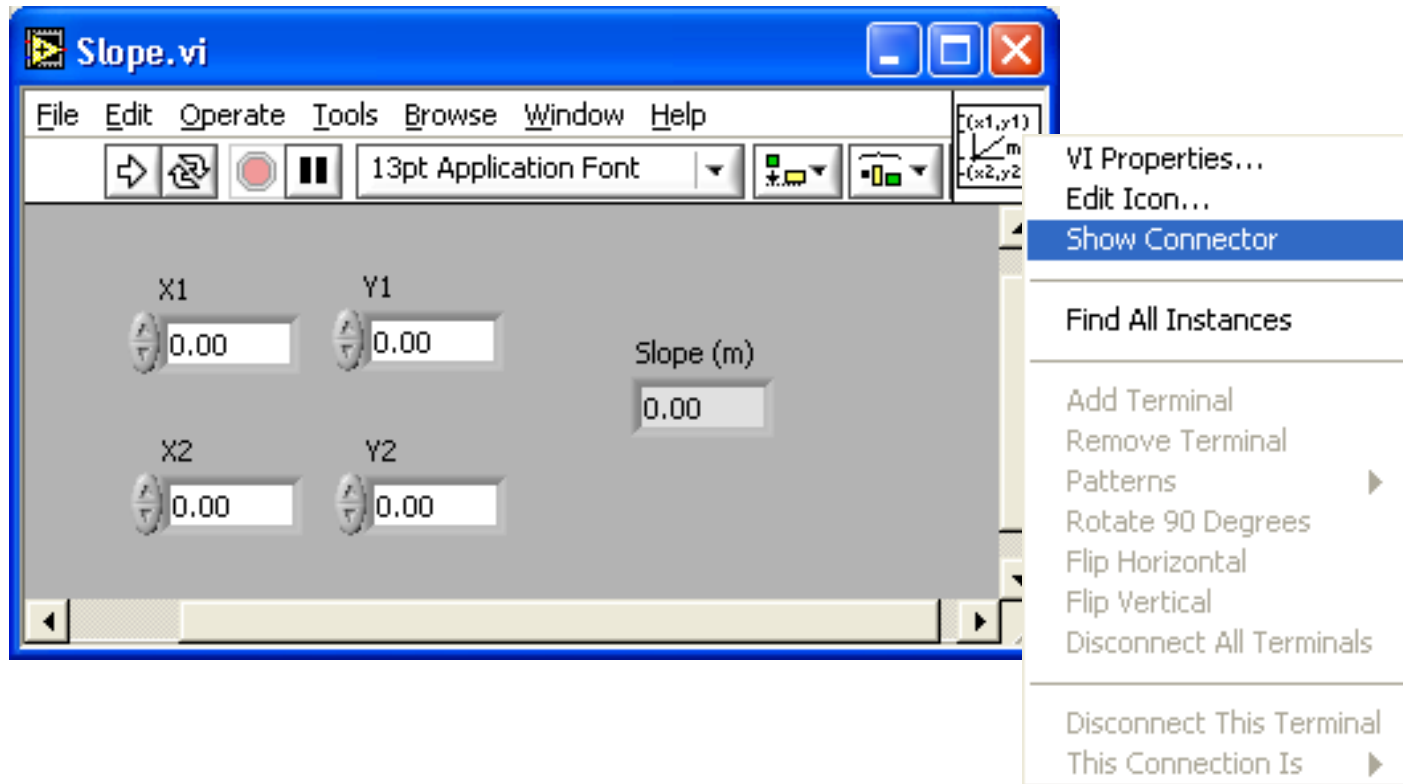
Create the Icon

- Right-click on the icon in the block diagram or front panel

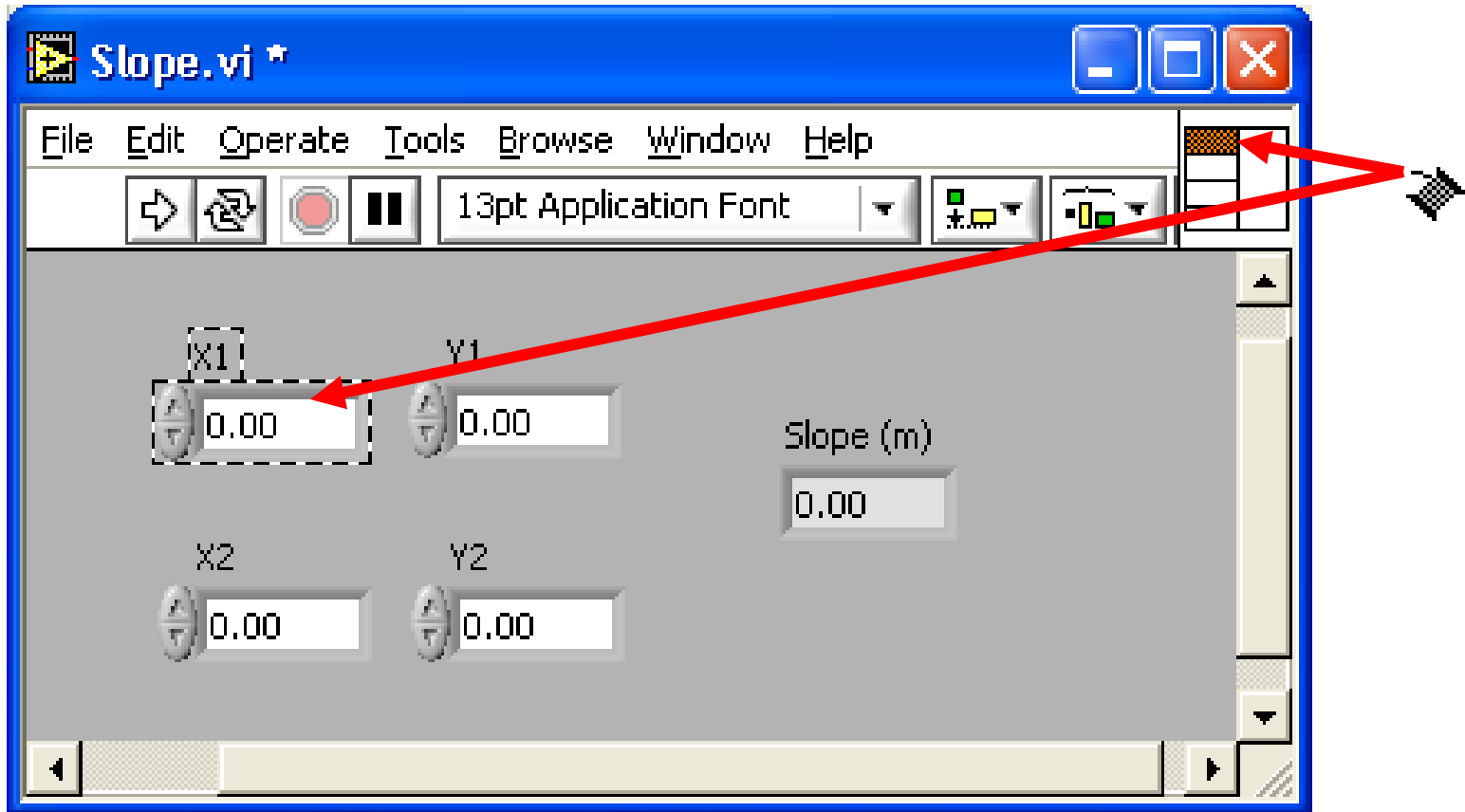


Create the Connector

Right click on the icon pane (front panel only)



Assign Terminals



Save The VI

- Choose an Easy to Remember Location
- Organize by Functionality
 - Save Similar VIs into one directory (e.g. Math Utilities)
- Organize by Application
 - Save all VIs Used for a Specific Application into one directory or library file (e.g. Lab 1 – Frequency Response)
 - Library Files (.lcls) combine many VI's into a single file, ideal for transferring entire applications across computers

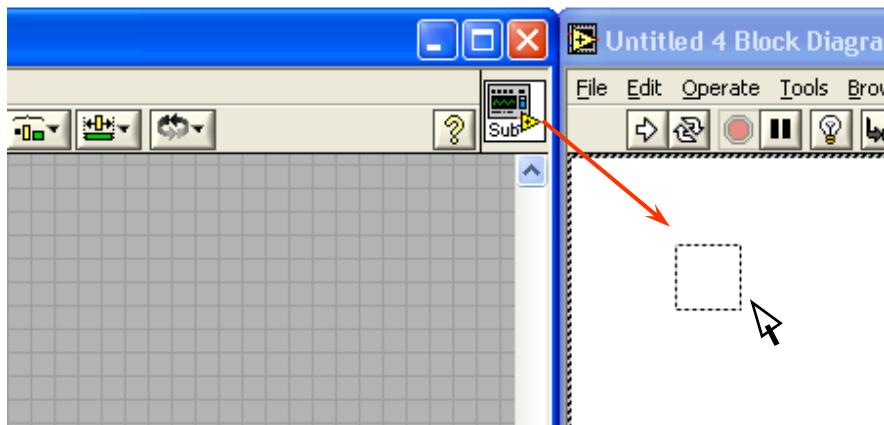
Insert the SubVI into a Top Level VI

Accessing user-made subVIs

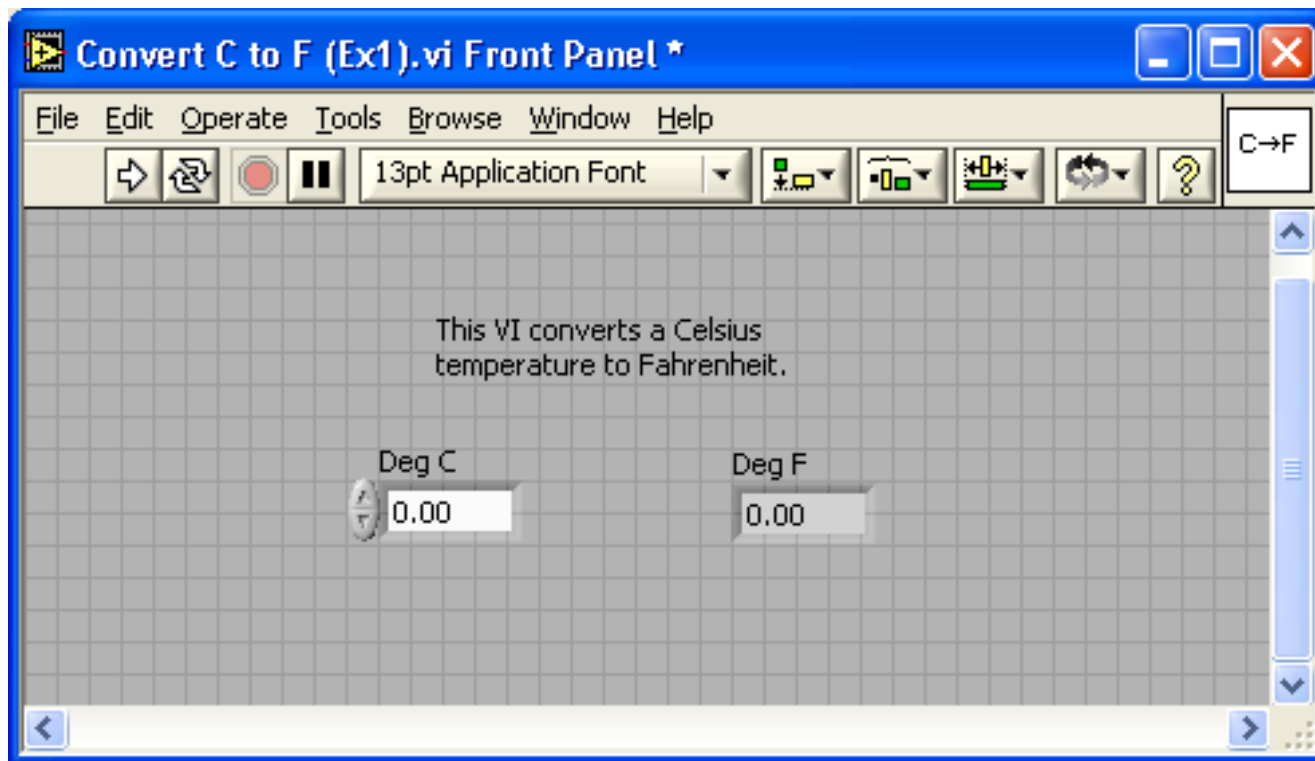
Functions >> All Functions >> Select a VI

Or

Drag icon onto target diagram



Exercise 2 – Make C2F.vi a SubVI



Section III – Loops and Charts

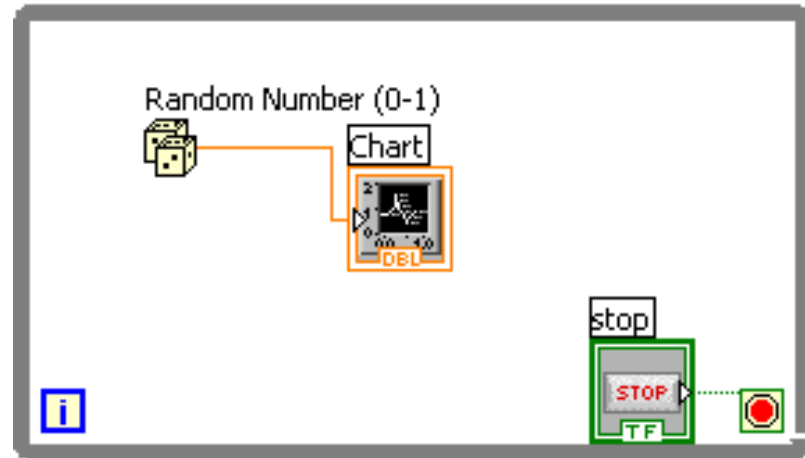
- For Loop
- While Loop
- Charts
- Multiplots

Loops

- While Loops

- Have Iteration Terminal
- Always Run at least Once
- Run According to Conditional Terminal

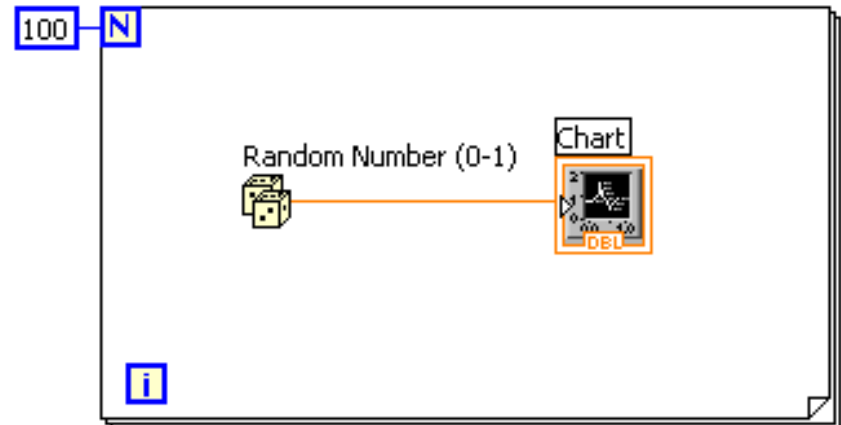
While Loop



- For Loops

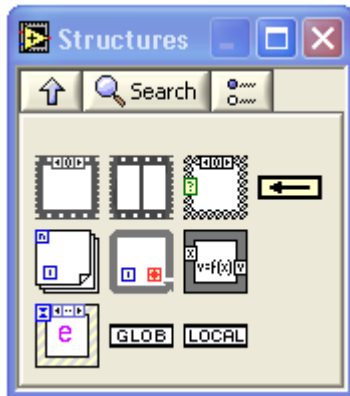
- Have Iteration Terminal
- Run According to input **N** of Count Terminal

For Loop

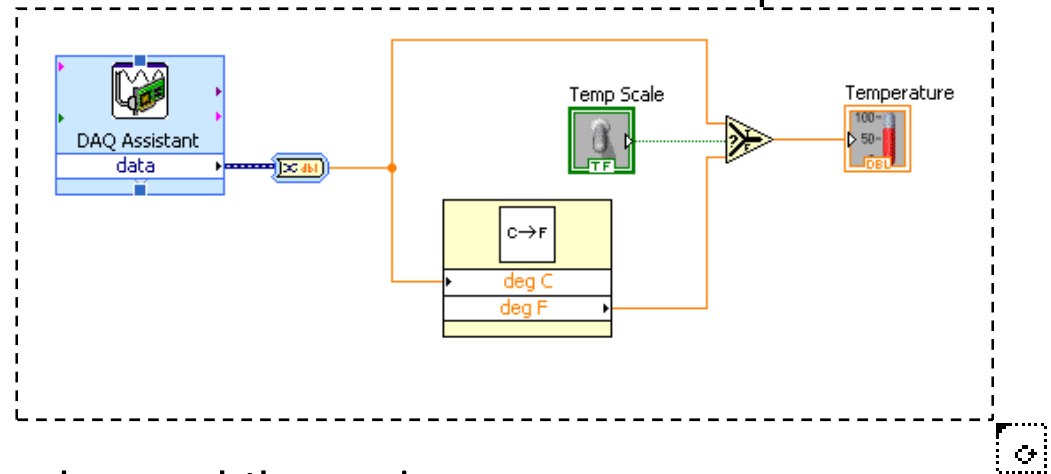


Loops (cont.)

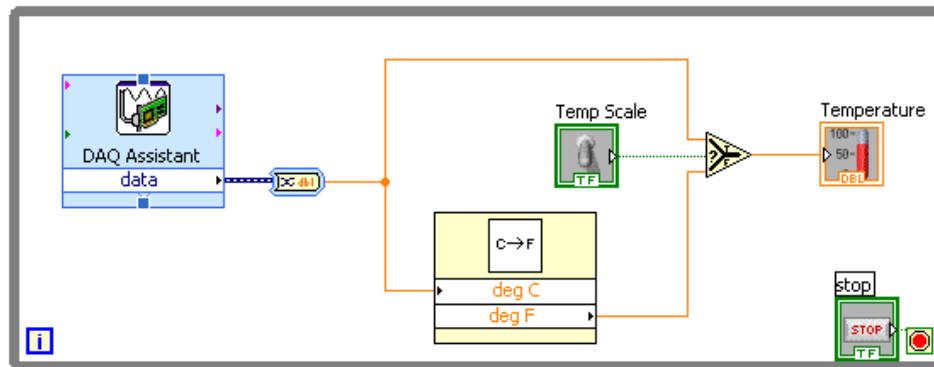
1. Select the loop



2. Enclose code to be repeated



3. Drop or drag additional nodes and then wire

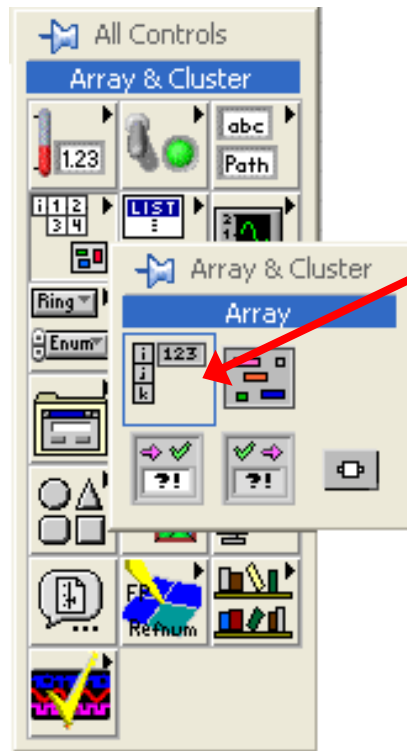


Section IV – Arrays

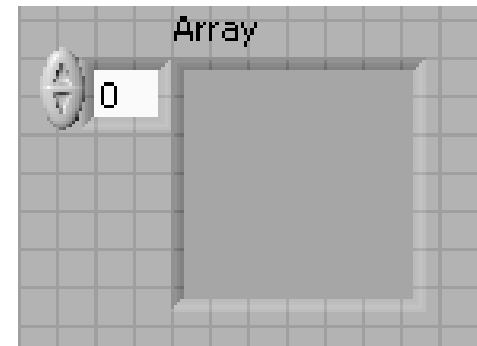
- Build arrays manually
- Have LabVIEW build arrays automatically

Adding an Array to the Front Panel

From the **Controls >> All Controls >> Array and Cluster** subpalette, select the **Array Shell**

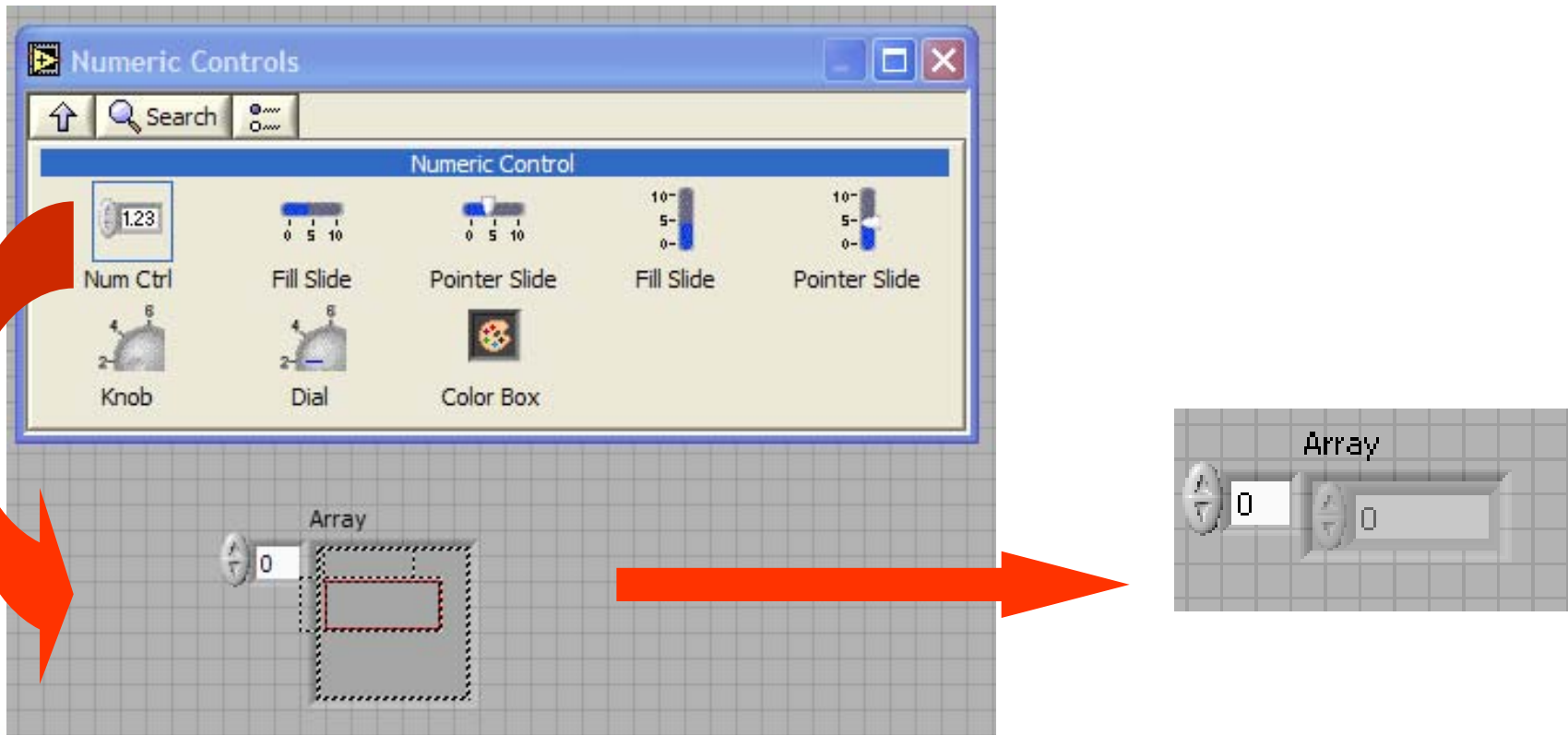


Drop it on the screen.



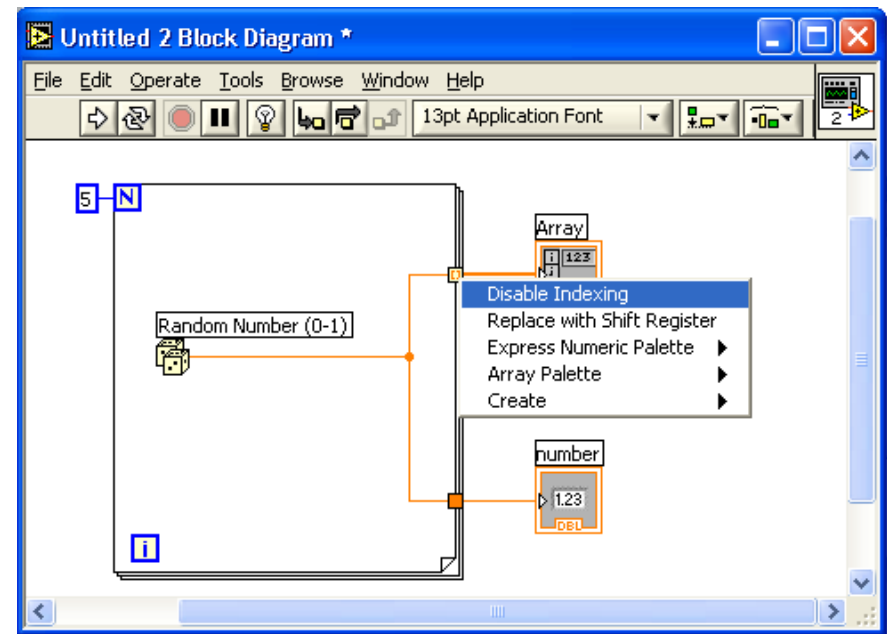
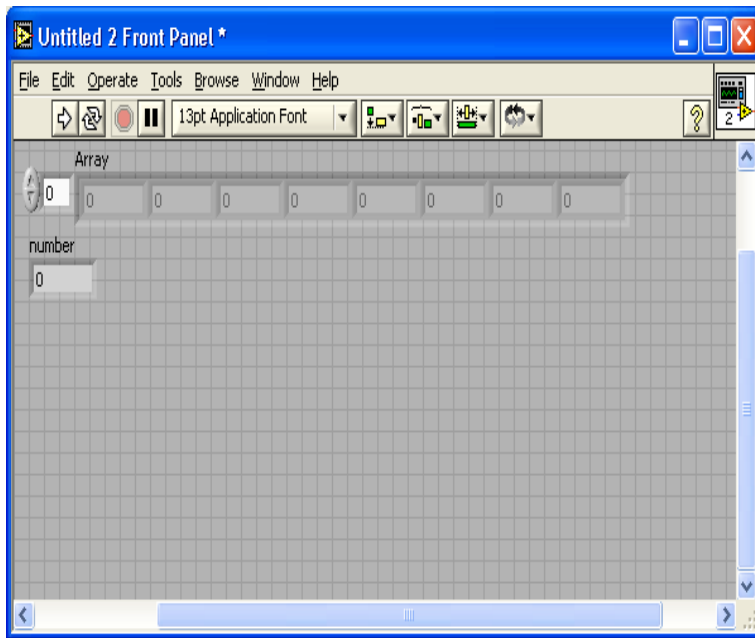
Adding an Array (cont.)

Place data object into shell (i.e. Numeric Control)

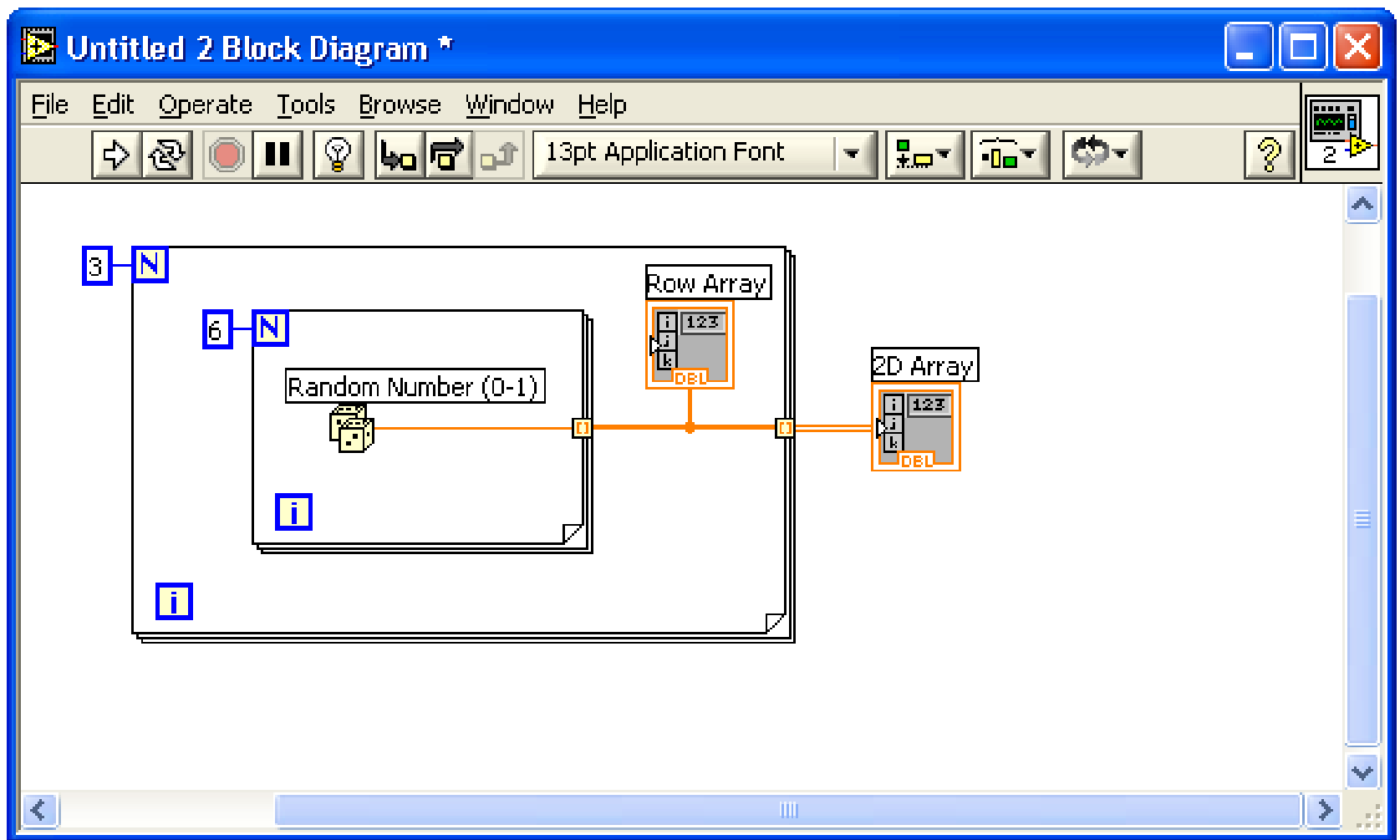


Creating an Array with a Loop

- Loops accumulate arrays at their boundaries



Creating 2D Arrays

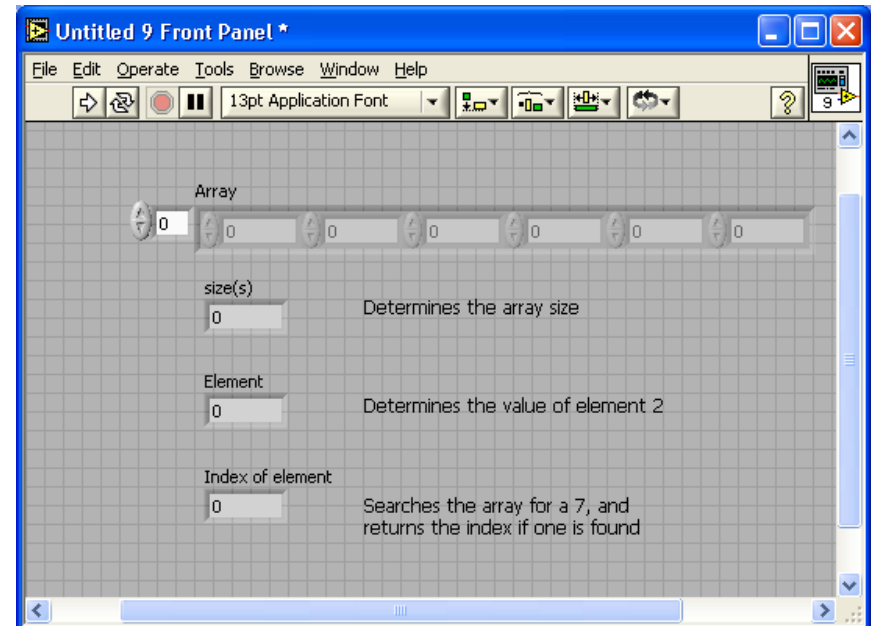
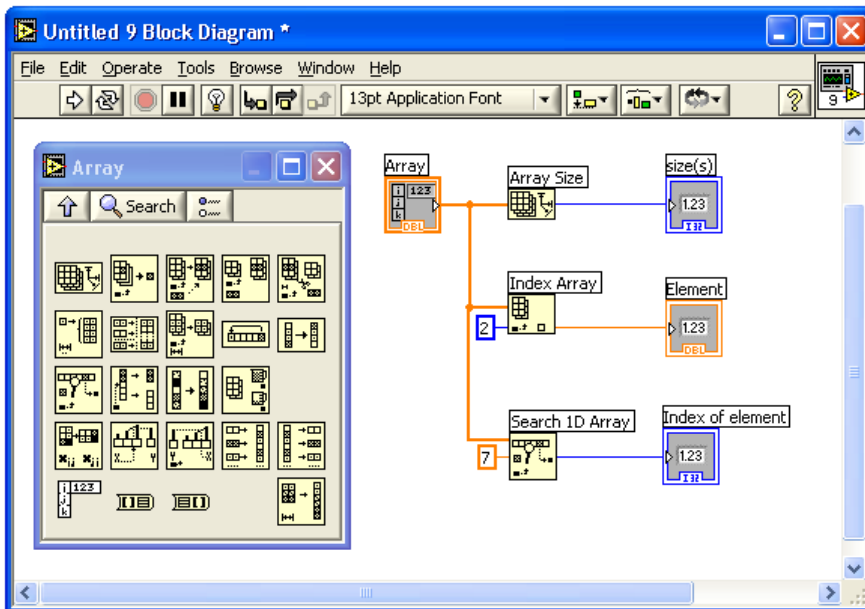


Section V – Array Functions & Graphs

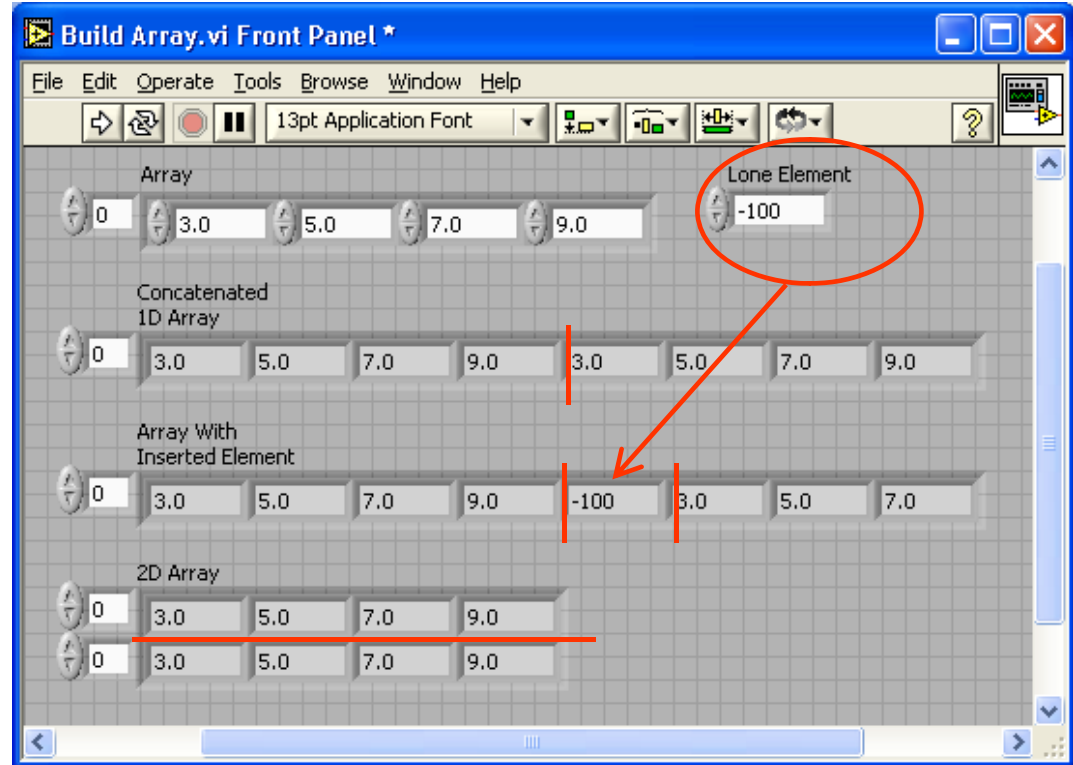
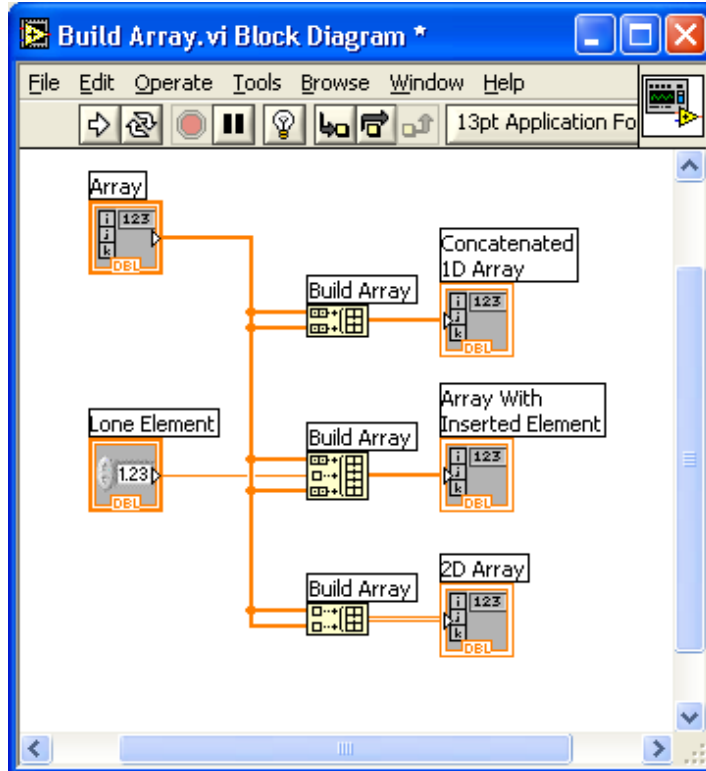
- Basic Array Functions
- Use graphs
- Create multiplots with graphs

Array Functions – Basics

Functions >> All functions>> Array



Array Functions – Build Array



Graphs

- Selected from the Graph palette of Controls menu
Controls>>All Controls>>Graphs

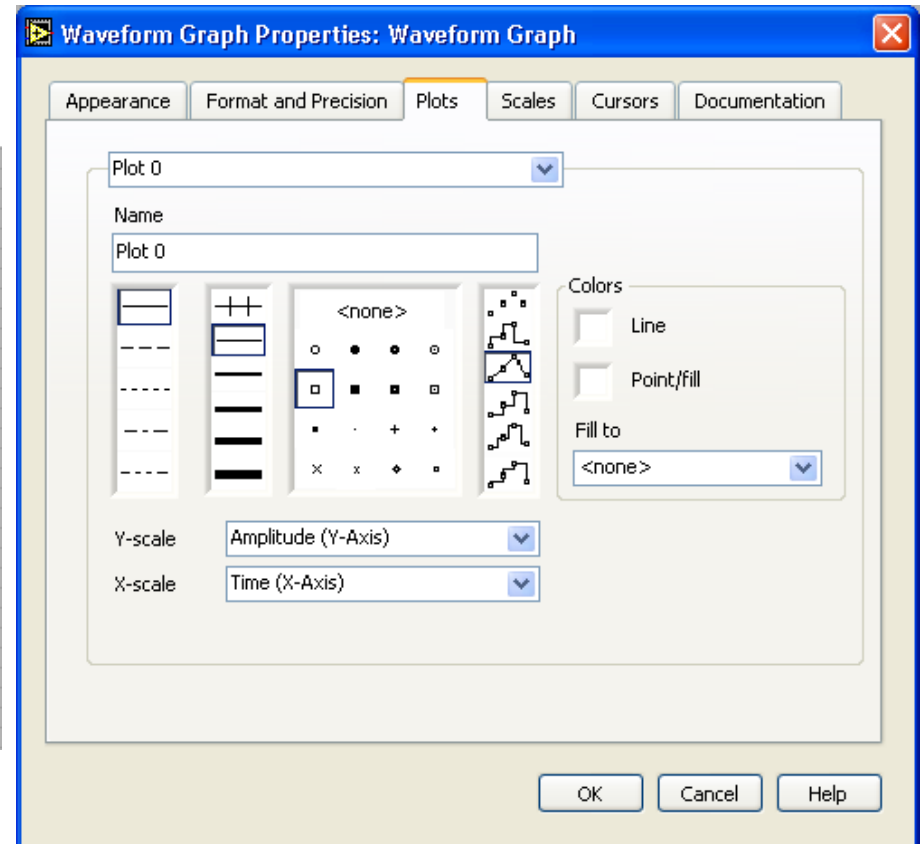
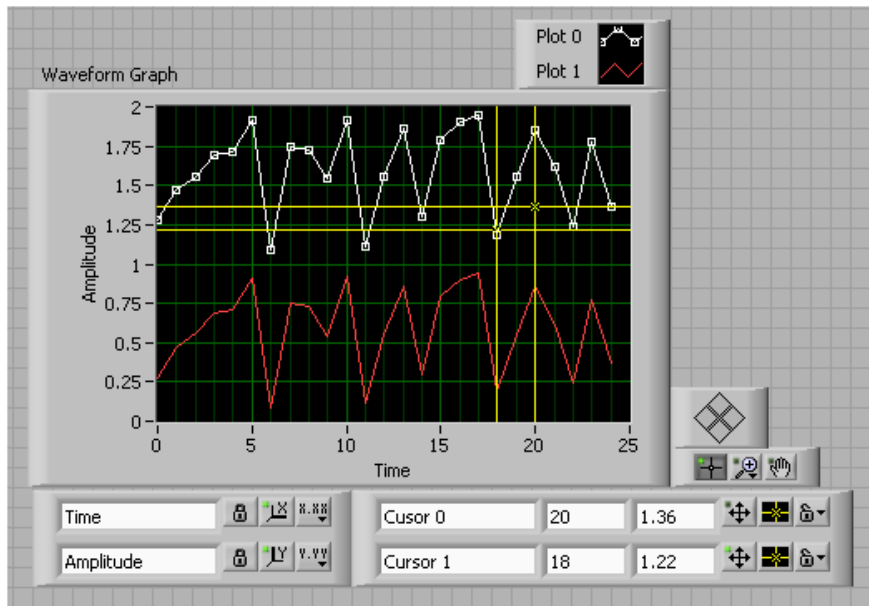
Waveform Graph – Plot an array of numbers against their indices

Express XY Graph – Plot one array against another

Digital Waveform Graph – Plot bits from binary data



Graphs



Right-Click on the Graph and choose Properties to Interactively Customize

Exercise 3 – Instantiate C2F.vi in a Top Level VI

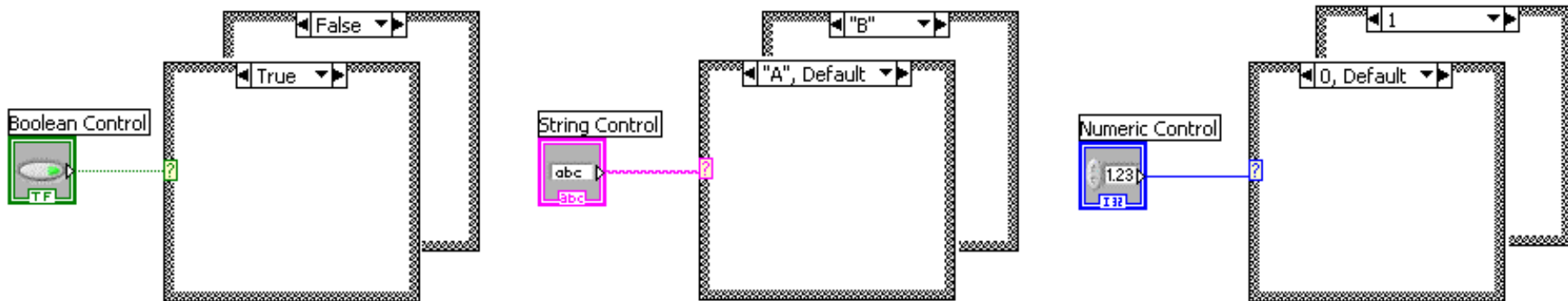
- Create a Top Level VI and insert C2F.vi
- Put C2F.vi in a For Loop and call it 100 times
 - Use the index i as the Celsius input to C2F.vi
 - Wire the output to the edge of the For Loop to create an array and plot the output

Section VI - Case & Sequence Structures, Formula Nodes

Case Structures

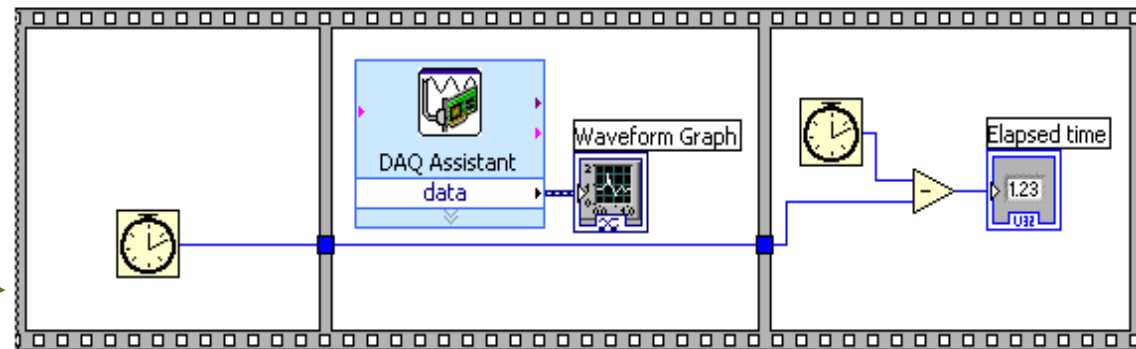
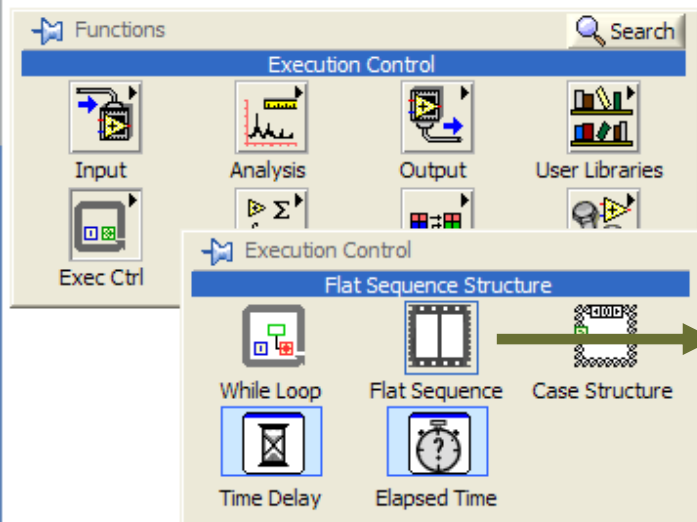
- In the Structures subpalette of Functions palette
- Enclose nodes or drag them inside the structure
- Stacked like a deck of cards, only one case visible

Functions >> Execution control



Sequence Structures

- In the **Execution Control** subpalette of Functions palette
- Executes diagrams sequentially
- Right-click to add new frame



Formula Nodes

- In the Structures subpalette
- Implement complicated equations
- Variables created at border
- Variable names are case sensitive
- Each statement must terminate with a semicolon (;)
- Context Help Window shows available functions

