

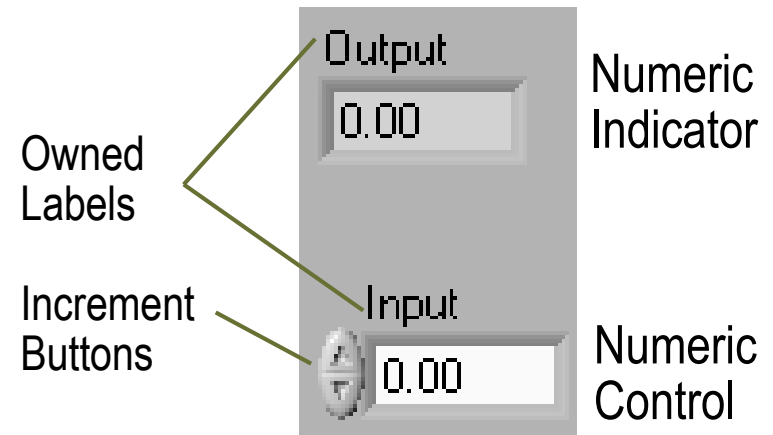
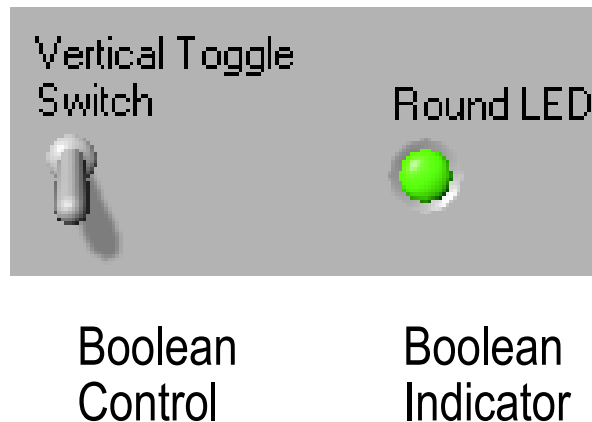
# **Lecture 2**

## **Creating, Editing and Debugging a VI SubVI and Hierarchical Structure**

吳文中

# Creating a VI Front Panel

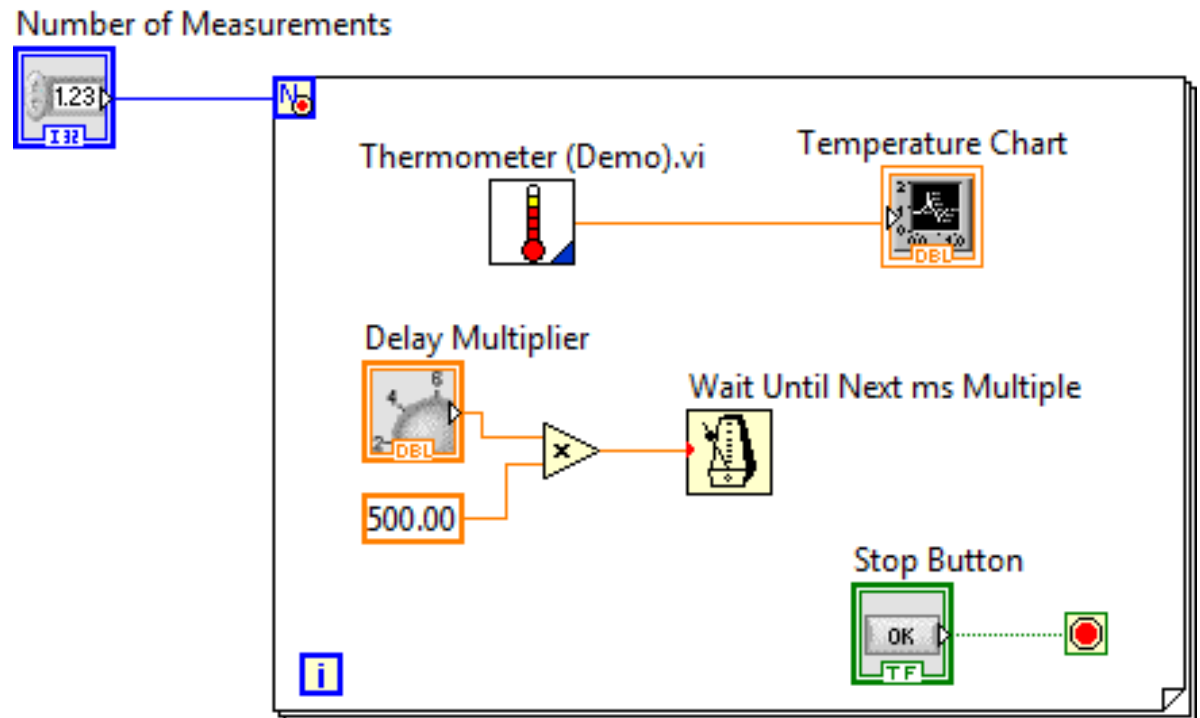
Build the front panel with controls (inputs) and indicators (outputs)



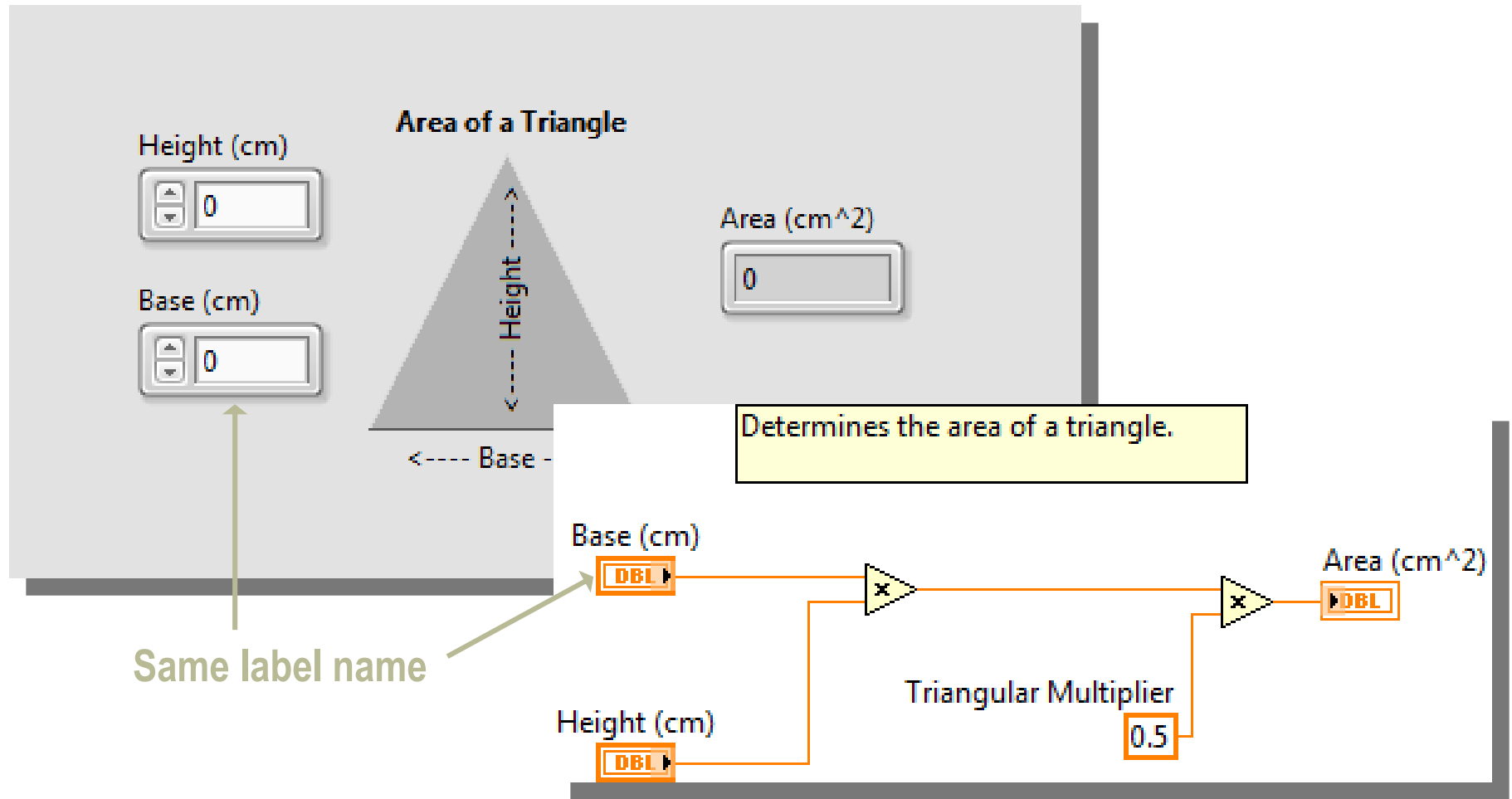
# Block Diagram

- Block diagram items:

- Terminals
- Constants
- Nodes
  - Functions
  - SubVIs
  - Structures
- Wires
- Free labels

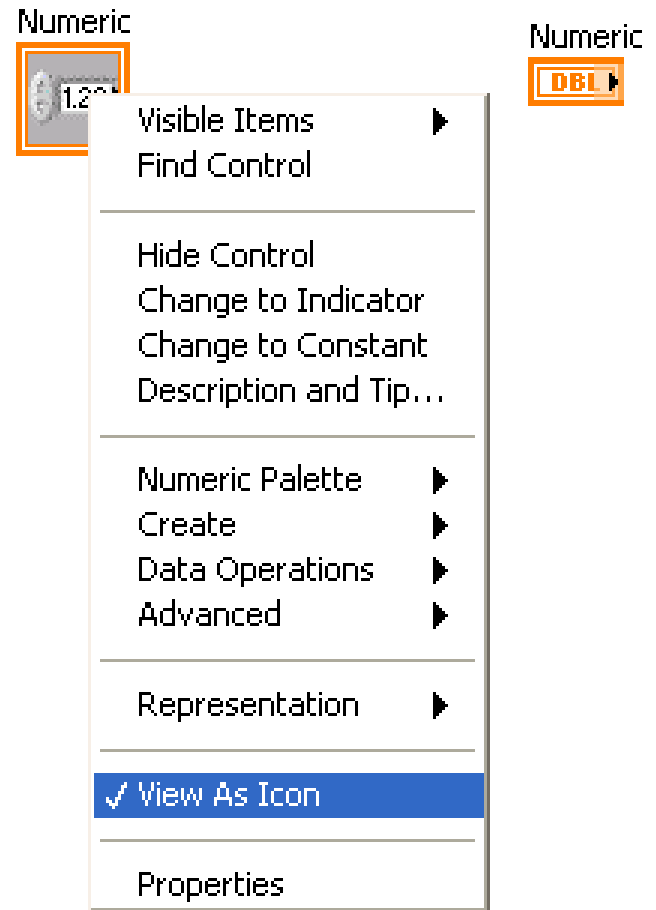


# Terminals



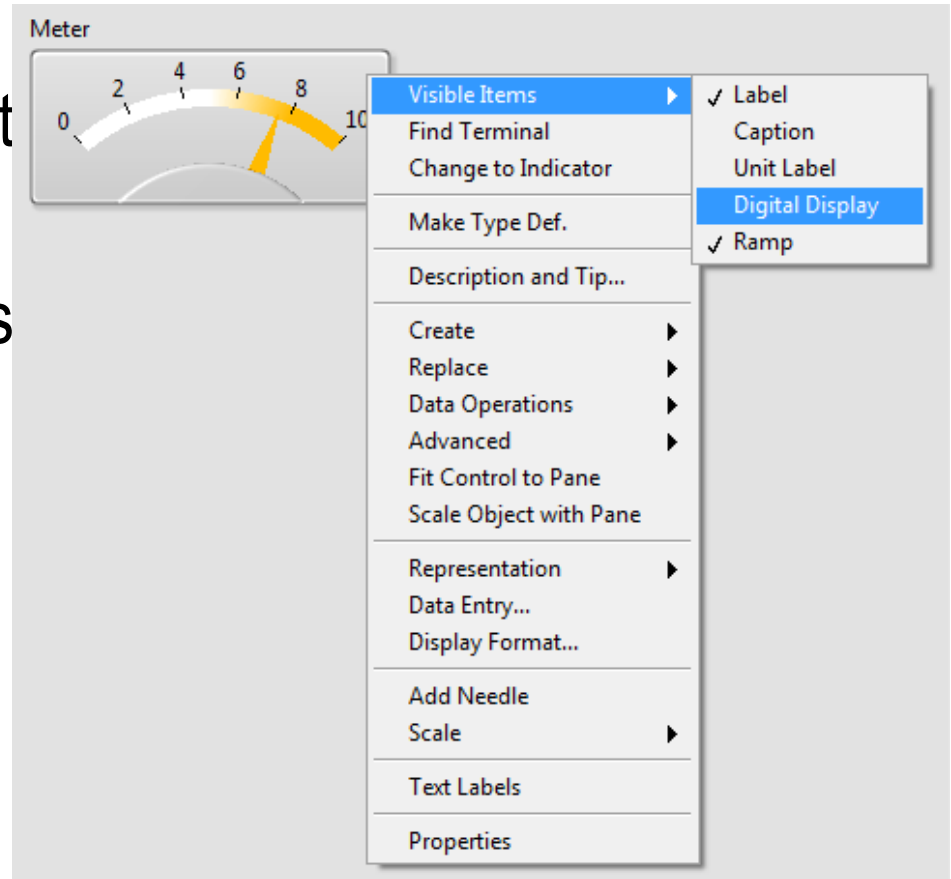
# Block Diagram Terminals

- Terminals are entry and exit ports that exchange information between the panel and diagram
- Terminals are analogous to parameters and constants in text-based programming languages
- Right-click and toggle View As Icon to change the icon view



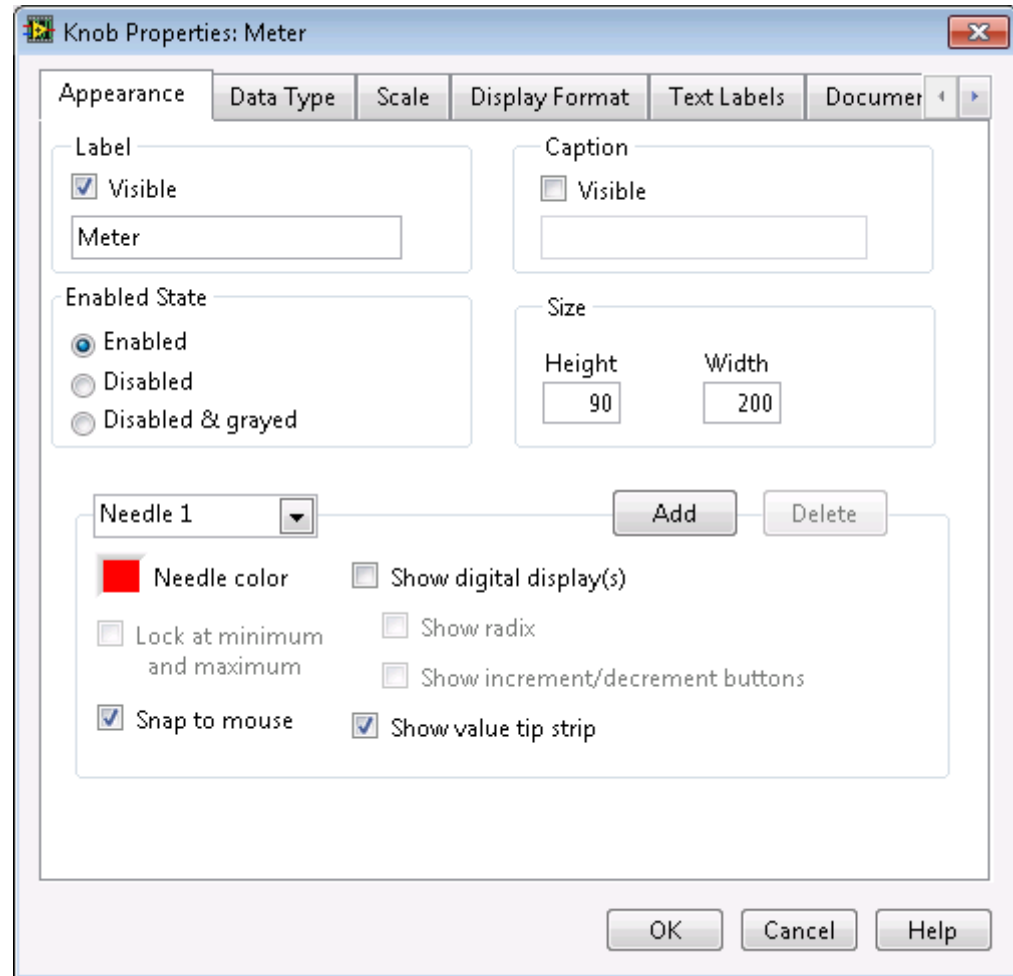
# Shortcut Menu

- All LabVIEW objects have associated shortcut menus.
- Use shortcut menu items to change the look or behavior of objects.
- To access the shortcut menu, right-click the object.



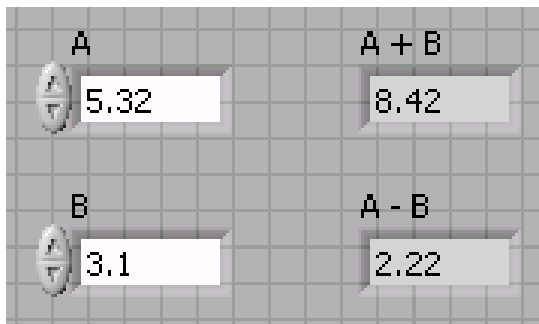
# Properties Dialog Box

- All LabVIEW objects have properties.
- To access properties, right-click the object and select **Properties**.
- Property options are similar to shortcut menu options.
- Select multiple objects to simultaneously configure shared properties.

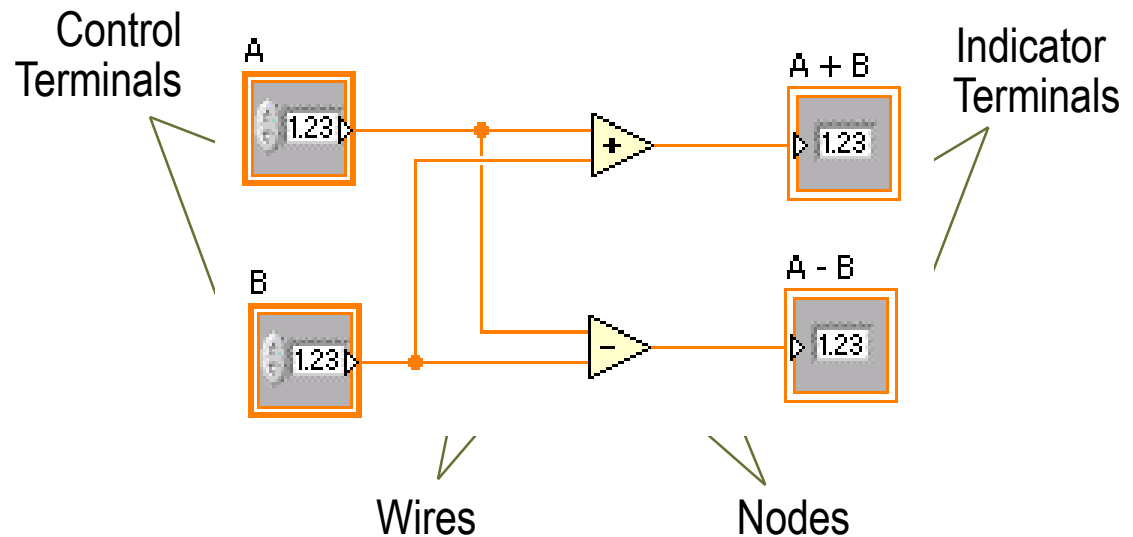


# Creating a VI Block Diagram

## Front Panel

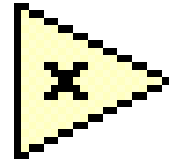


## Block Diagram





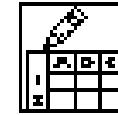
# Function Nodes



- Functions are:
  - Fundamental operating elements of LabVIEW.
  - Do not have front panels or block diagrams, but do have connector panes.
  - Has a pale yellow background on its icon.
- Double-clicking a function only selects the function.
- Functions do not open like VIs and subVIs.

# SubVI Nodes

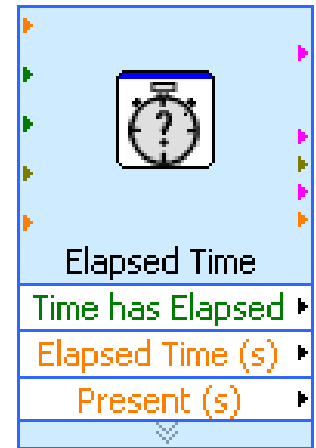
Write To Spreadsheet File.vi



- SubVIs :
  - Are VIs that you use on the block diagram of another VI.
  - Have front panels and block diagrams.
  - Use the icon from the upper-right corner of the front panel as the icon that appears when you place the subVI on a block diagram.
- When you double-click a subVI, the front panel and block diagram open.
- Any VI has the potential to be used as a subVI.

# Express VIs

- Express VIs:
  - Are a special type of subVI.
  - Require minimal wiring because you configure them with dialog boxes.
  - Save each configuration as a subVI.
- Icons for Express VIs appear on the block diagram as icons surrounded by a blue field.

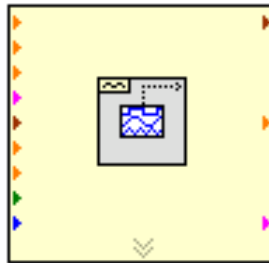


# Block Diagram Nodes

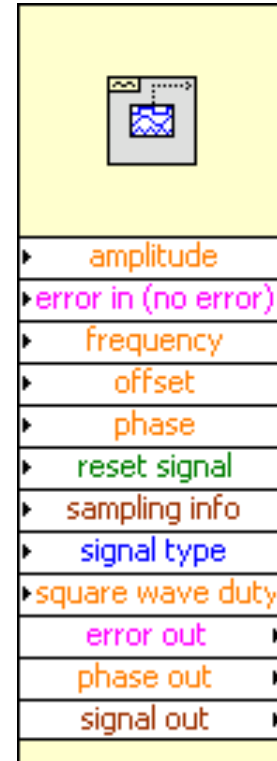
Icon



Expandable Node



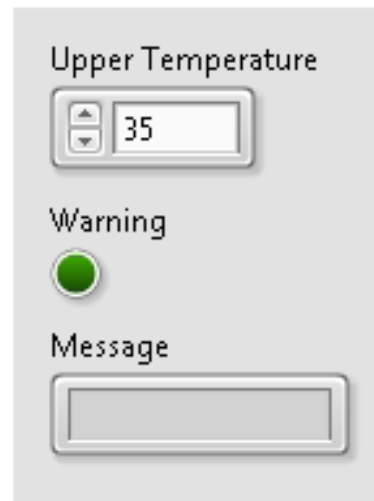
Expanded Node



- Function Generator VI
- Same VI, viewed three different ways
- Yellow field designates a standard VI
- Blue field designates an Express VI

# Front Panel Basics













Front panel controls and indicators create terminals on the block diagram.



# Wires

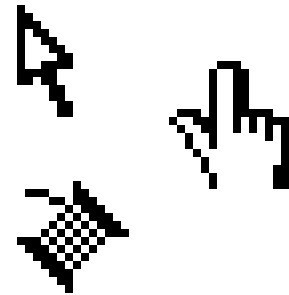
- Wires transfer data between block diagram objects.
- Wires are different colors, styles, and thicknesses, depending on their data types.
- A broken wire appears as a dashed black line with a red X in the middle.



	Floating-point	Integer	String	Boolean
Scalar				
1-D Array				
2-D Array				

# Selecting a Tool

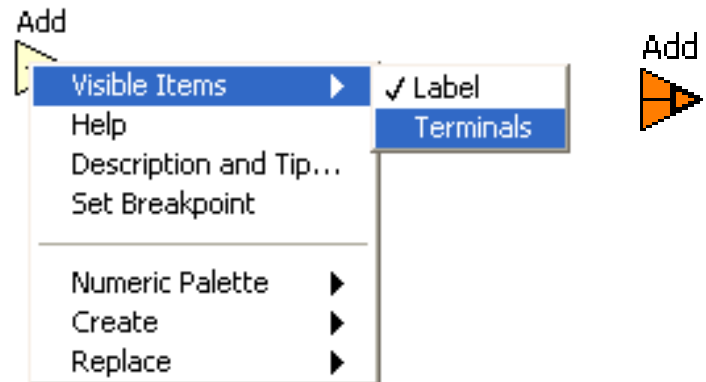
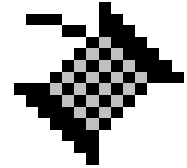
- A tool is a special operating mode of the mouse cursor.
- Create, modify, and debug VIs using the tools provided by LabVIEW.
- By default, LabVIEW automatically selects tools based on the context of the cursor.
- If you need more control, use the **Tools** palette to select a specific tool.
  - Select **View»Tools Palette** to open the **Tools** palette.



# Wiring Techniques

- Automatic Wiring
- Use Context Help Window when wiring
- Right-click wire and select **Clean Up Wire**
- Tip Strips
- Automatic wire routing
- Right-click terminals and select **Visible Items»Terminals**

Hot Spot



View the terminal connections to a function



# Cloning and Moving Items

– Clone an object in Windows using the following steps:

1. Select the Positioning tool.
2. Press the <Ctrl> key while clicking an object.
3. Drag the copy to new location.

– Move an object using the following steps:

1. Select the Positioning tool.
2. Click and drag the object to new location.

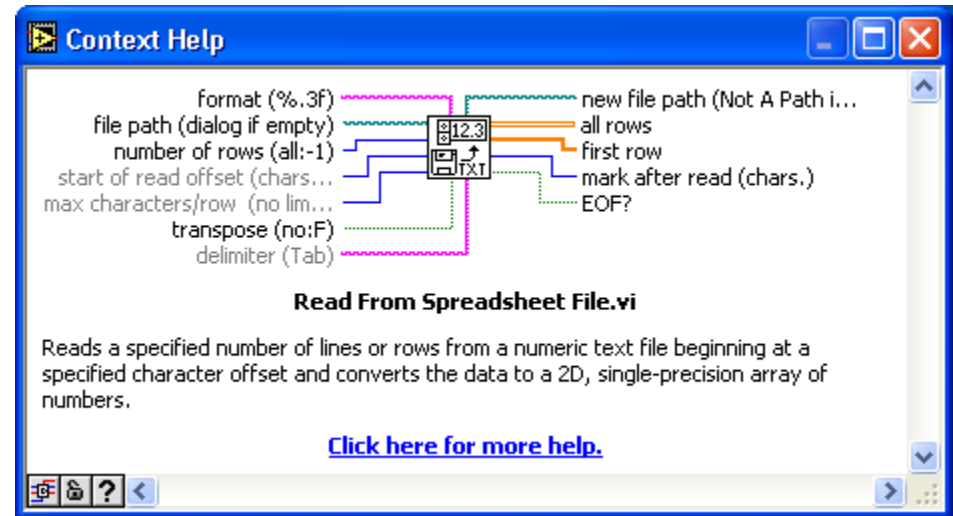
**Note:** Avoid cutting and pasting objects as this can impact related items. For example, cutting and pasting a block diagram terminal also moves the front panel object.

# Selecting, Editing, Resizing and Wiring

- Select item to move, copy, or delete
- Edit text
- Resize an object
- Wire terminals and nodes

# Context Help

- To display the Context Help window, select **Help»Show Context Help**, press the <**Ctrl-H**> keys, or press the **Show Context Help Window** button in the toolbar
- Move cursor over object to display help
- Connections:
  - Required – bold**
  - Recommended – normal
  - Optional - dimmed



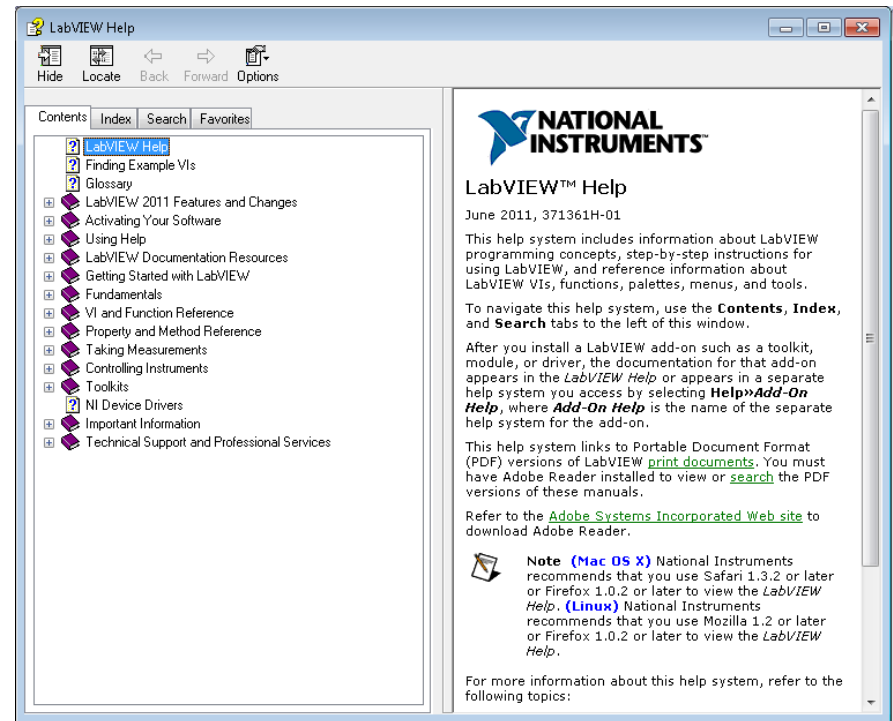
Simple/Detailed Context Help

Lock Help

More Help

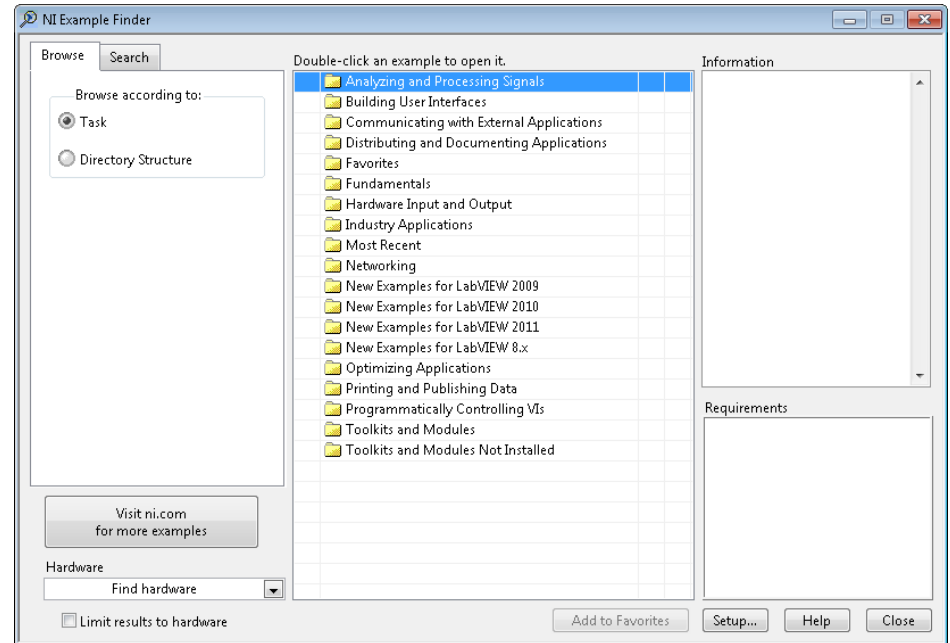
# LabVIEW Help

- Contains detailed descriptions and instructions for most palettes, menus, tools, VIs, and functions.
- Can be accessed by:
  - Selecting **Help» LabVIEW Help** from the menu.
  - Clicking the **Detailed help** link in the **Context Help** window.
  - Right-clicking an object and selecting **Help** from the shortcut menu.



# Examples

- LabVIEW includes hundreds of example VIs.
- Use NI Example Finder to browse and search installed examples.
  - Select **Help»Find Examples** in the menu.
- Click the example buttons in *LabVIEW Help* topics.



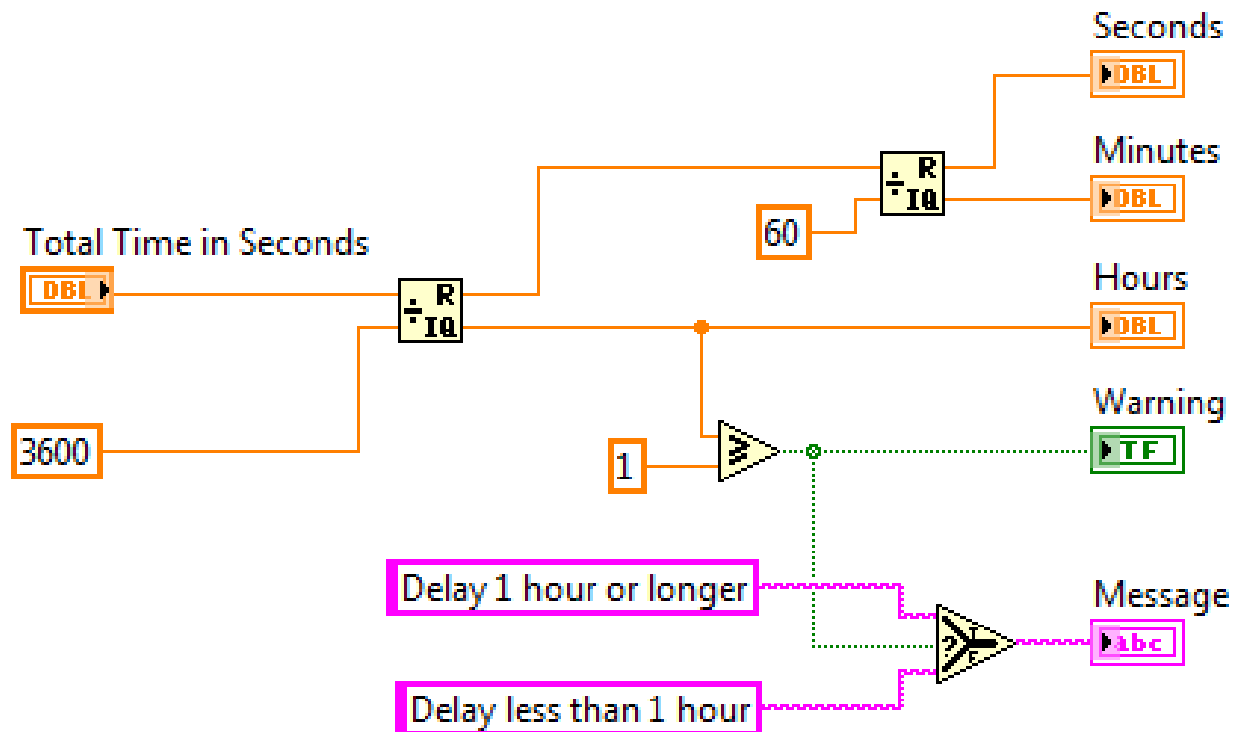
Open example



Find related examples

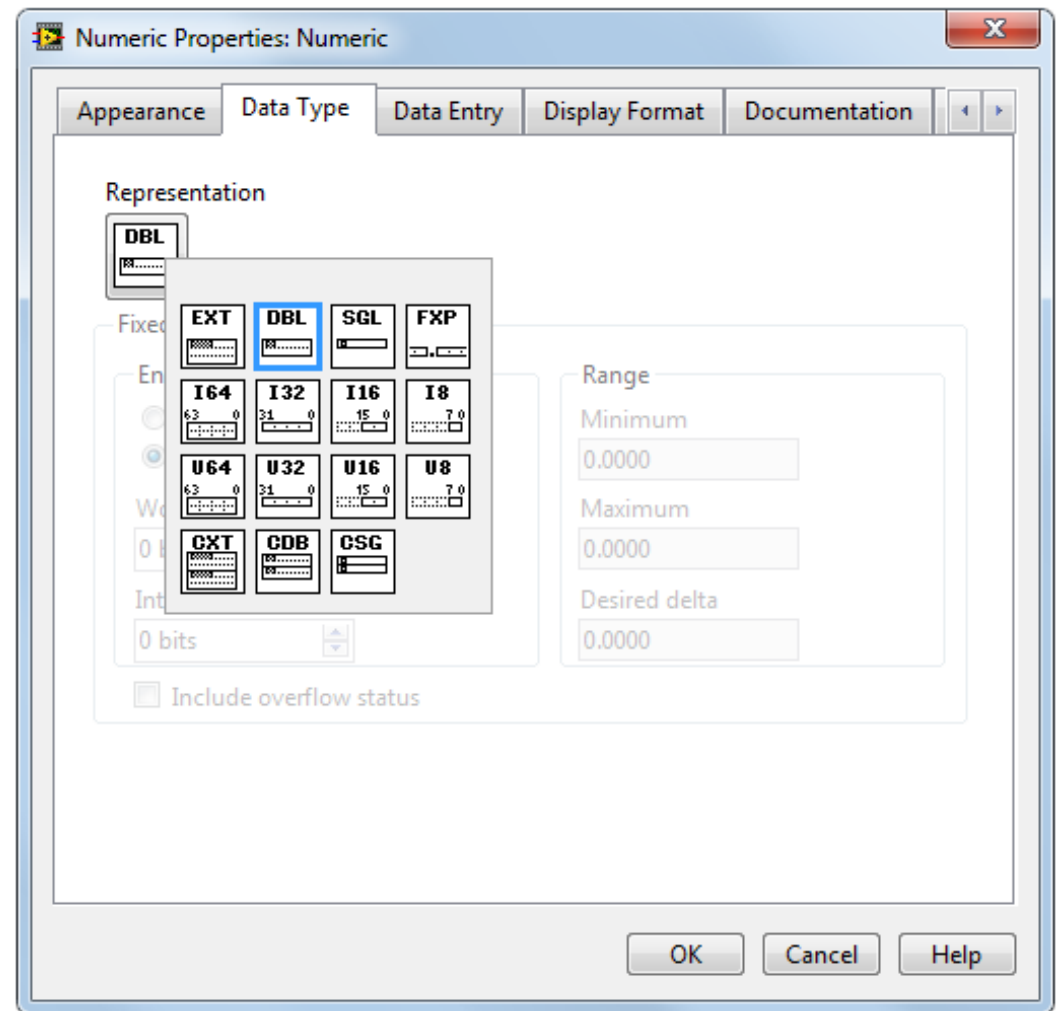
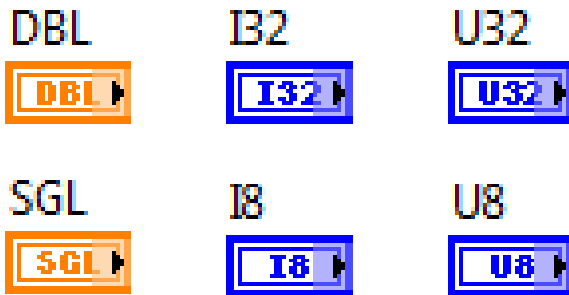
# LabVIEW Data Types

Terminals visually communicate information about the data type represented



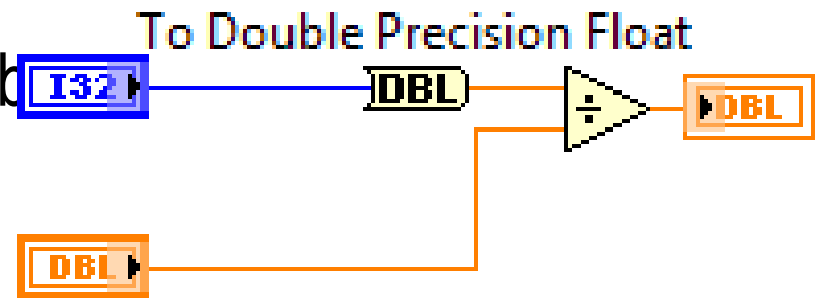
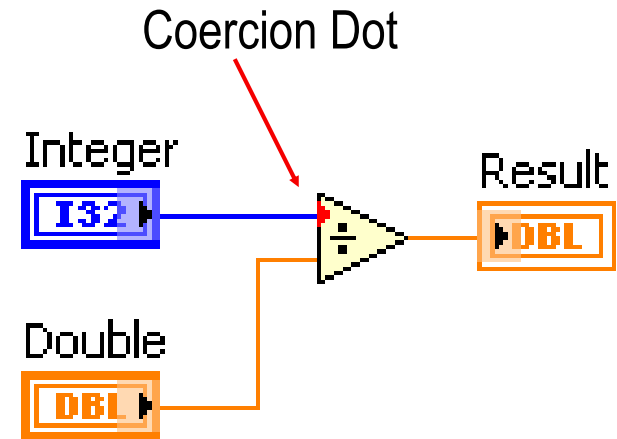
# Numerics

- Various data type representations:
  - Floating-point
  - Unsigned integers
  - Signed integers



# Numeric Conversion

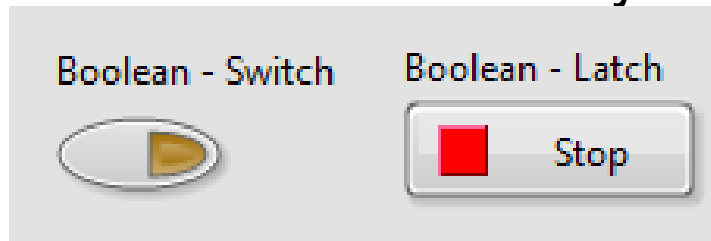
- Coercion dots indicate that LabVIEW converted the value passed into a node to a different representation.
  - Occurs when a node expects an input with a different representation.
- LabVIEW chooses the representation that uses more bits.
- Avoid coercion by programmatically converting to a matching data type.





# Booleans

- Behavior of Boolean controls is specified by the mechanical action.
- Boolean have only



Boolean - Switch



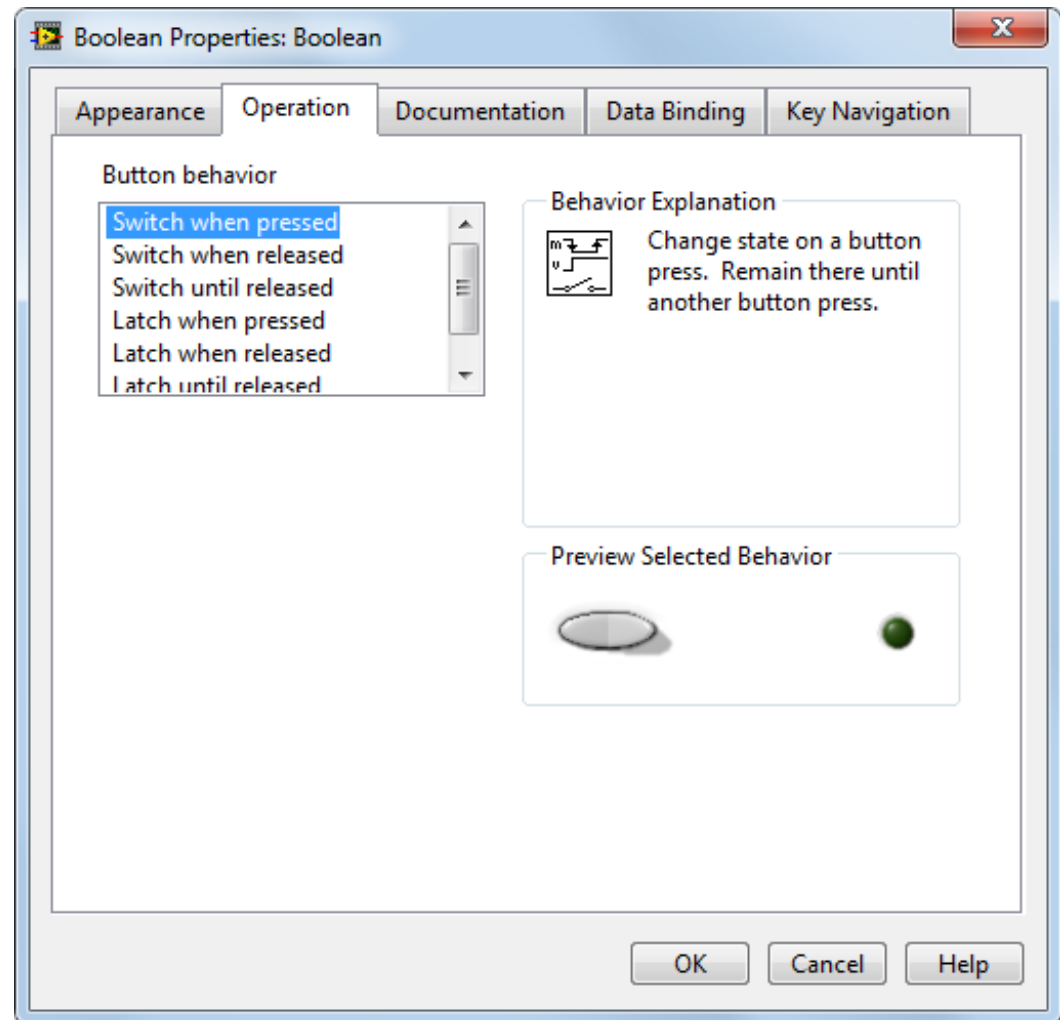
True Constant



Boolean - Latch



False Constant



# Mechanical Action of Booleans



•Use the **Properties»Operations** tab of a Boolean control to learn about the different switch and latch actions.

# Strings

- A string is a sequence of ASCII characters.
- Strings have various display styles.
  - Backslash codes
  - Password
  - Hex



String Control



Empty String Constant



String Indicator



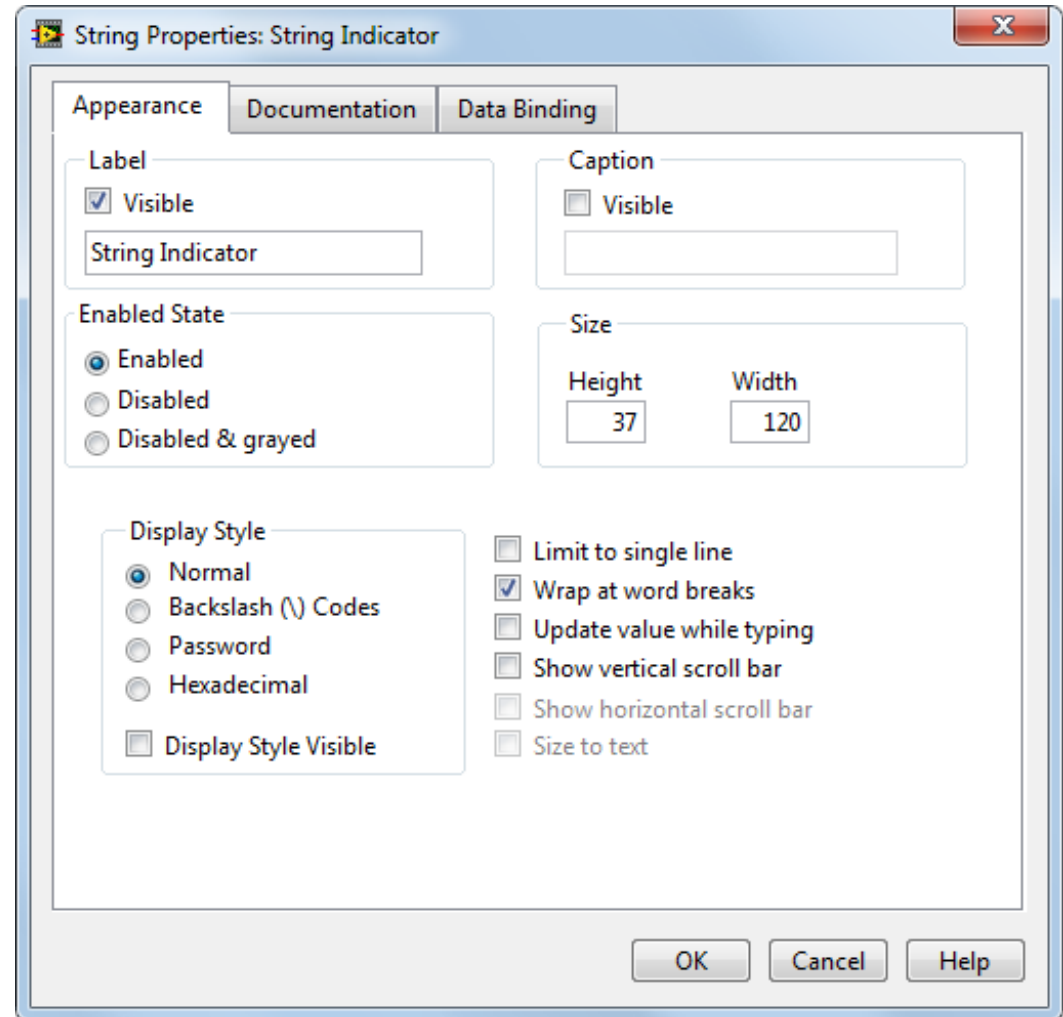
String Constant



Tab Constant

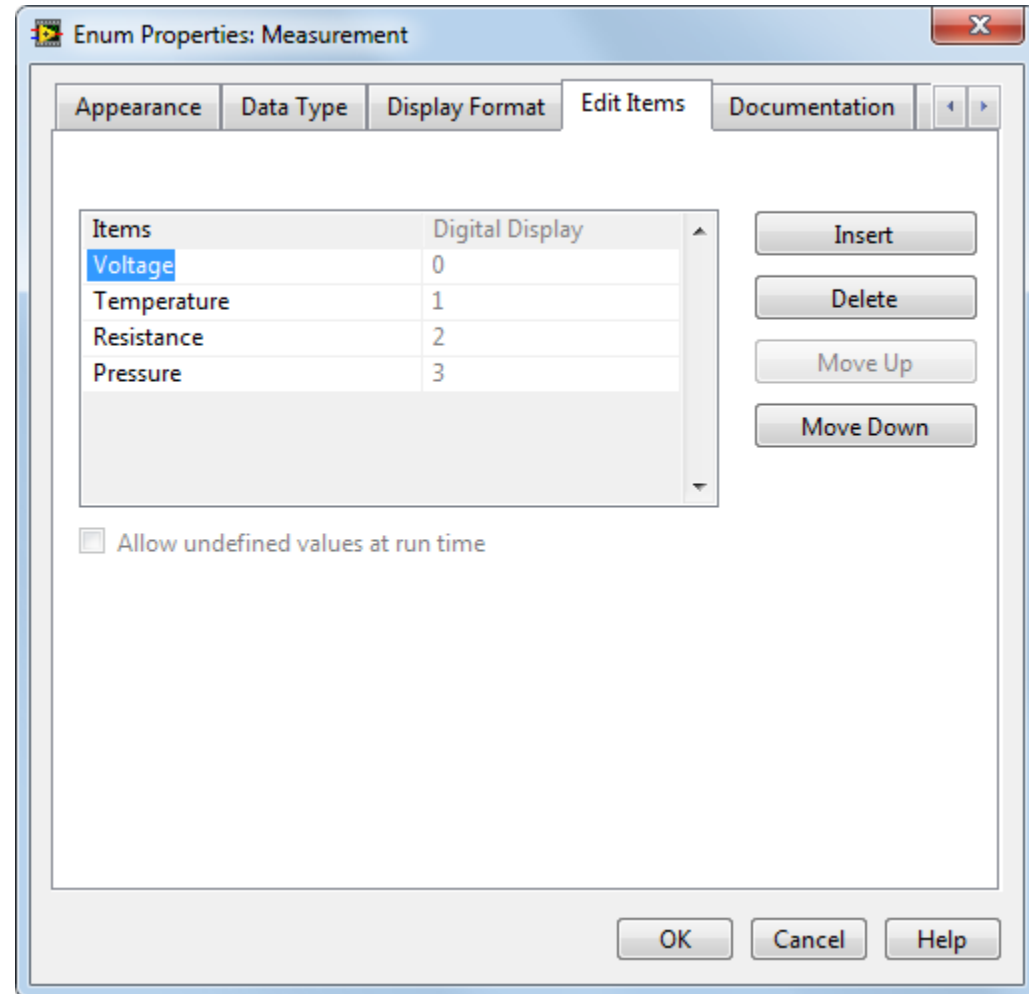
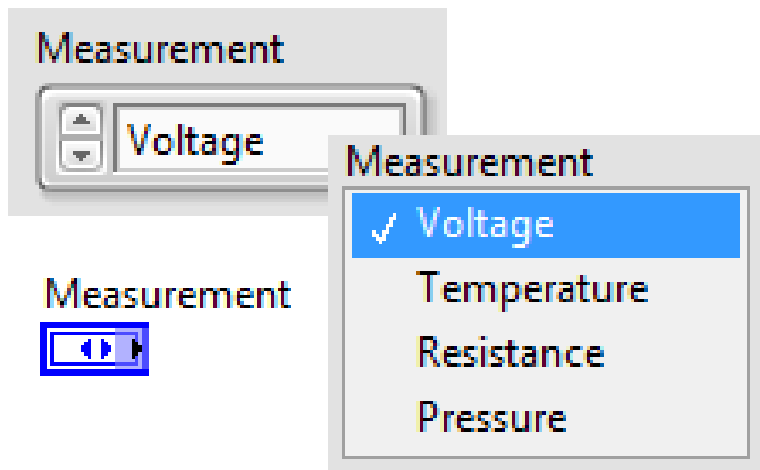


End of Line Constant



# Enums

- Enums give users a list of items from which to select.
- Each item represents a pair of values.
  - String
  - 16-bit Integer



# Other Data Types

Refer to *LabVIEW Help* for complete list of terminal symbols for different types of controls and indicators.

– Dynamic 

- Stores the information generated or acquired by an Express VI.

– Path 

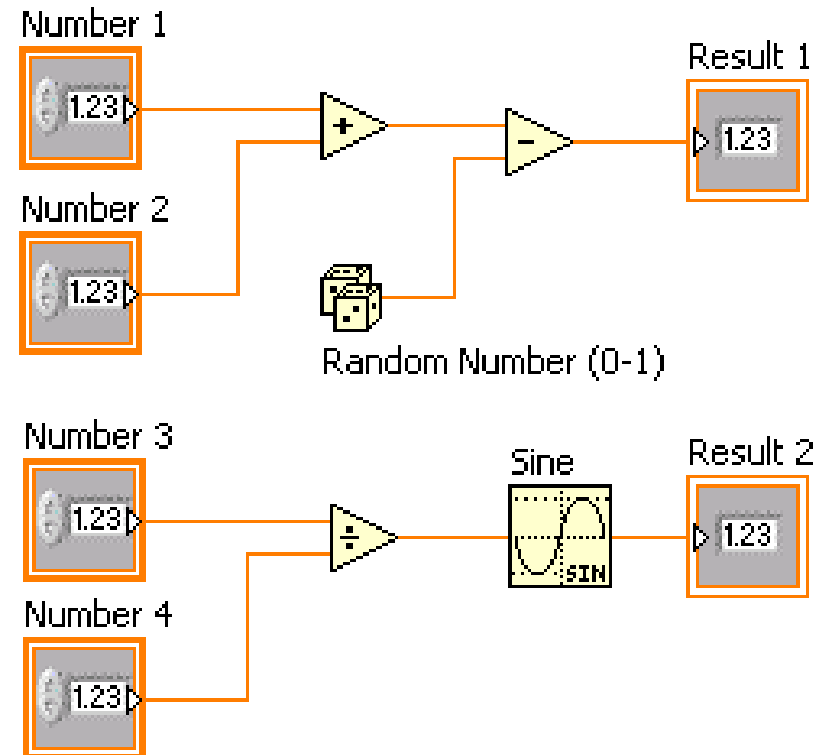
- Stores the location of a file or directory using the standard syntax for the platform you are using.

– Waveform 

- Carries the data, start time, and dt of a waveform.

# 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



# Debugging Techniques

## Finding Errors



Click on broken Run button. A window showing the error appears

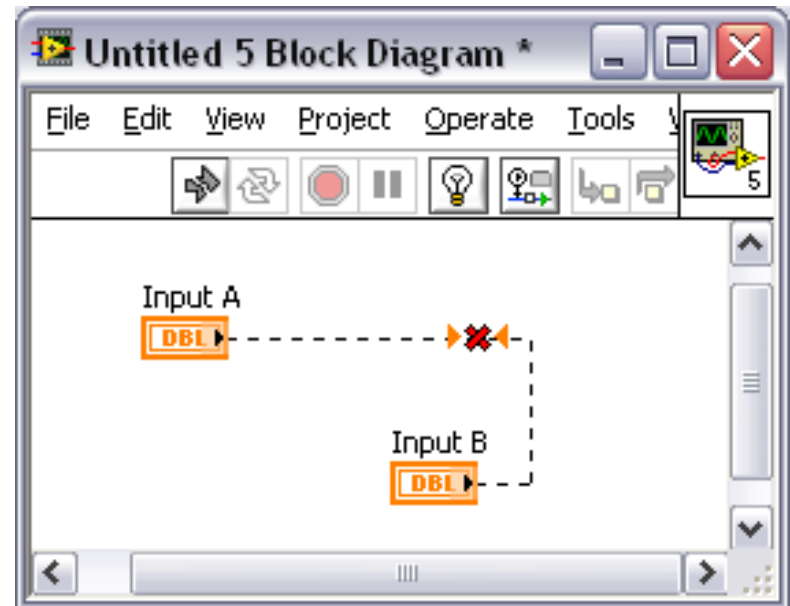
## Execution Highlighting



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

# Common Causes of Broken VIs

- Broken wires exist on the block diagram.
  - You wired a Boolean control to a String indicator.
  - You wired a numeric control to a numeric control.
- A required block diagram terminal is unwired.
- A subVI is broken.





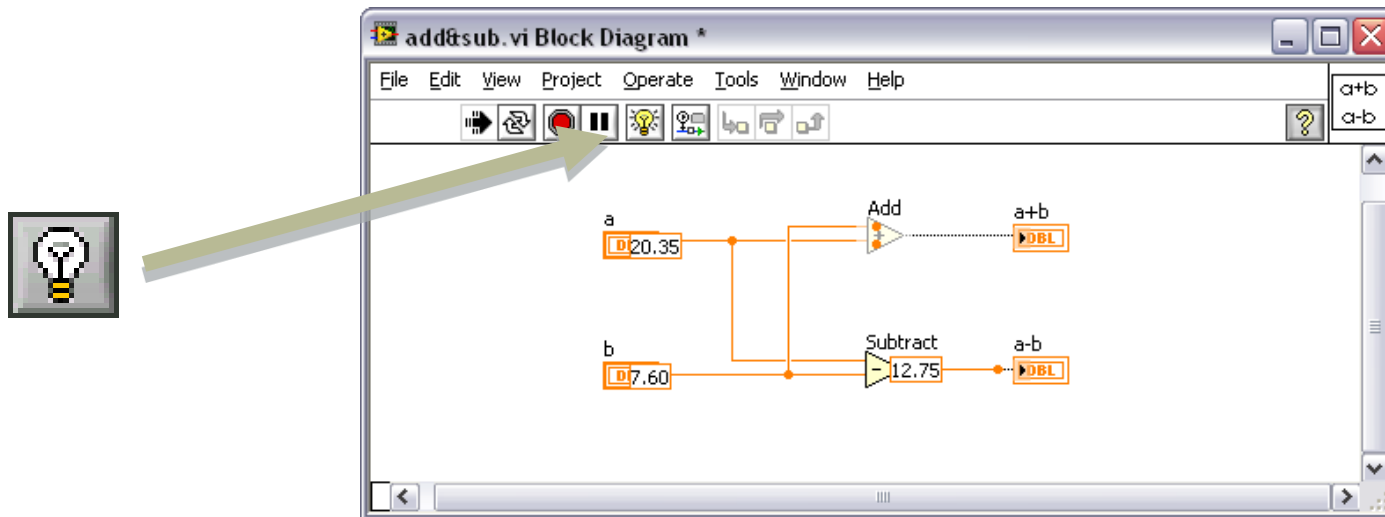
# Debugging Techniques

What to look for if a VI produces unexpected data or behavior:

- Are there any unwired or hidden subVIs?
- Is the default data correct?
- Does the VI pass undefined data?
- Are numeric representations correct?
- Are node executed in the correct order?

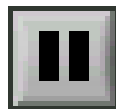
# Execution Highlighting

- Use execution highlighting to watch the data flow through the block diagram.
- If the VI runs more slowly than expected, confirm that you turned off execution highlighting in subVIs.



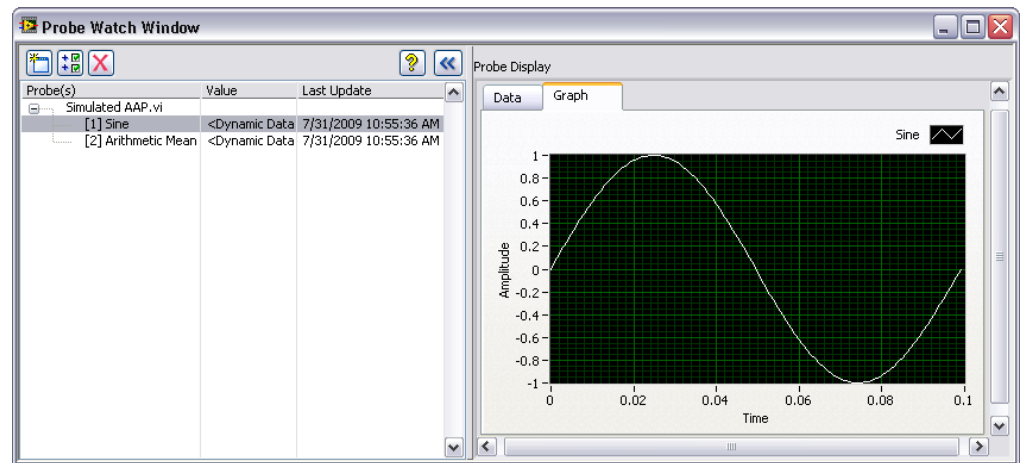
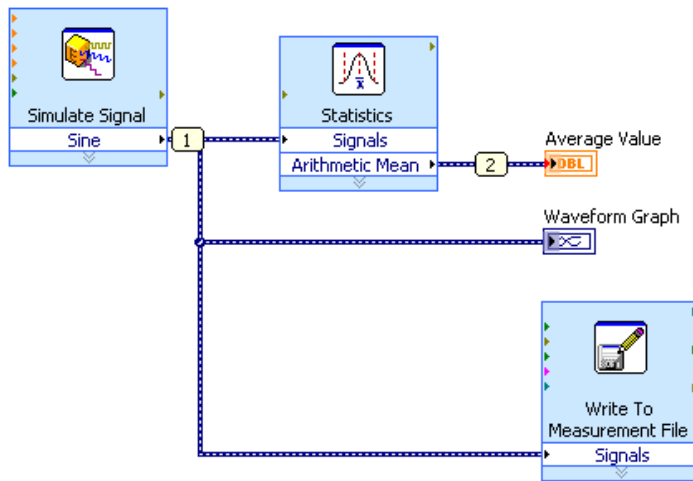
# Single-Stepping

- Single-step through the VI to view each action of the VI on the block diagram.
- Suspend the execution of a subVI to edit values of controls and indicators, to control the number of times it runs, or to go back to the beginning of the execution of the subVI.
  - Open subVI and select **Operate»Suspend When Called** from the shortcut menu.




# Probes

- Use the Probe tool to observe intermediate data values and check the error output of VIs and functions, especially those performing I/O.
- Specify to retain the values in the wires so that you can probe wires for data after execution.



# Breakpoints

- When you reach a breakpoint during execution, the VI pauses and the **Pause** button appears red. 
- You can take the following actions at a breakpoint:
  - Single-step through execution using the single-stepping buttons.
    - Probe wires to check intermediate values.
    - Change values of front panel controls.
  - Click the **Pause** button to continue running to the next breakpoint or until the VI finishes running.

# Undefined or Unexpected Data

Check for unexpected Inf values or NaN values in your mathematical operations:

- $\infty$  (Inf)
  - Infinity
  - Produced by dividing a number by zero.
- NaN
  - Not a number
  - Produced by invalid operations, such as taking the square root of a negative number.

# Documenting Code

## VI

- Name
- Description

## Front Panel

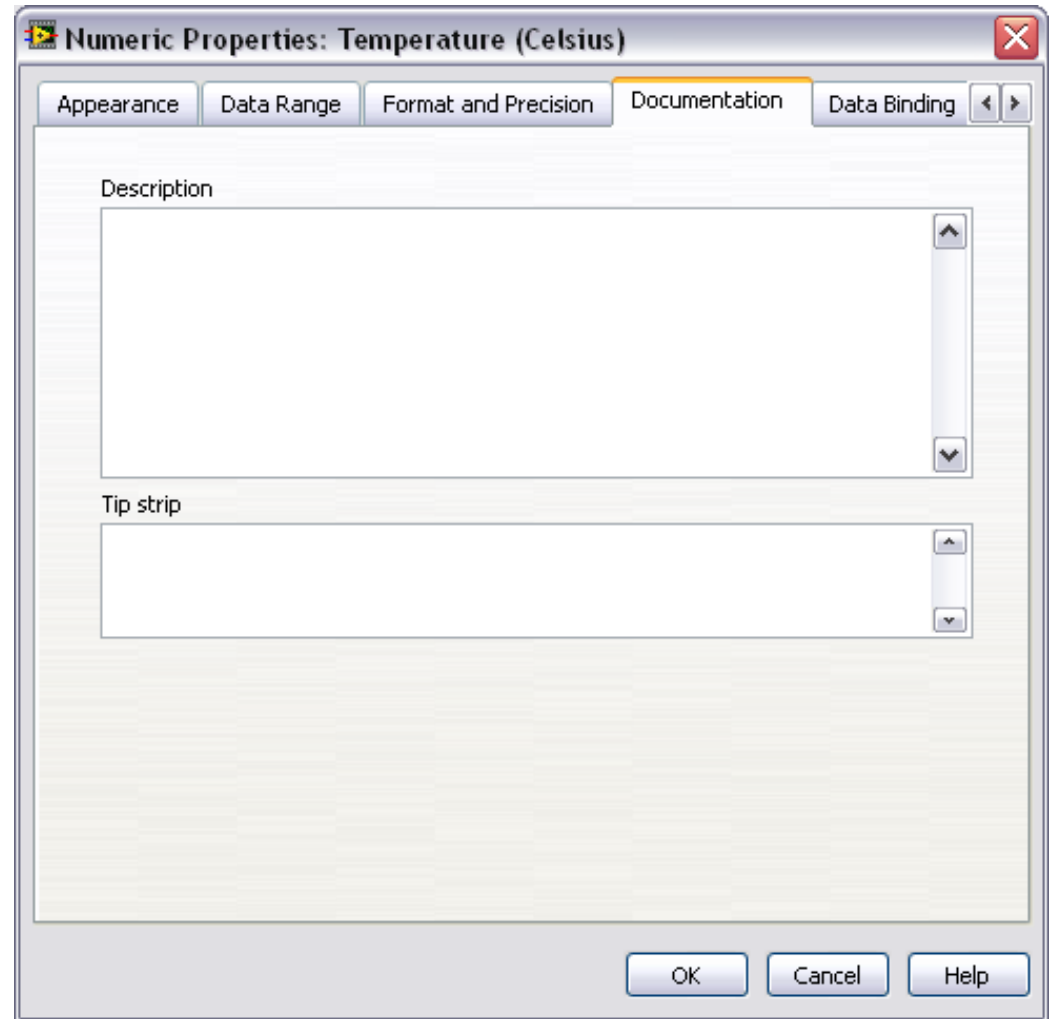
- Label Names
- Tip Strips
- Descriptions
- Free Labels

## Block Diagram

- Label Names
- Free Labels
- Owned Labels
- SubVI Descriptions

# Creating Descriptions and Tip Strips

- Use the **Properties** dialog box to create documentation for an object.

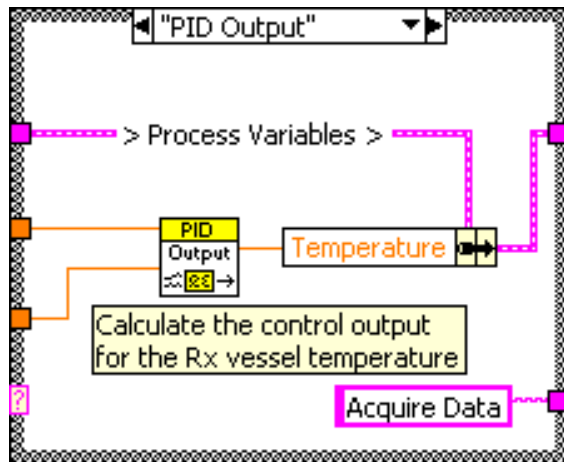




# Documenting Block Diagram Code

Free labels:

- Describe algorithms.
- Have pale yellow backgrounds.
- Double-click in any open space to create.



Owned labels:

- Explain data contents of wires and objects.
- Move with object.
- Have transparent backgrounds.
- Select **Visible Items»Label** from the shortcut menu to create.

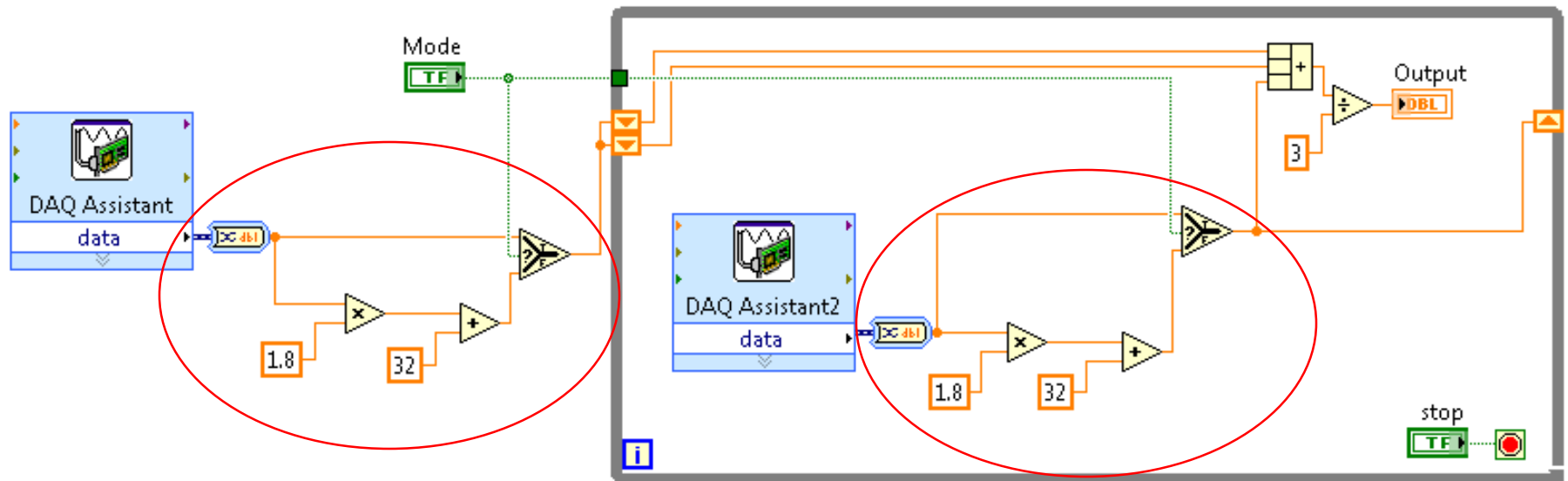
# Understanding Modularity – SubVIs



SubVI — A VI within another VI

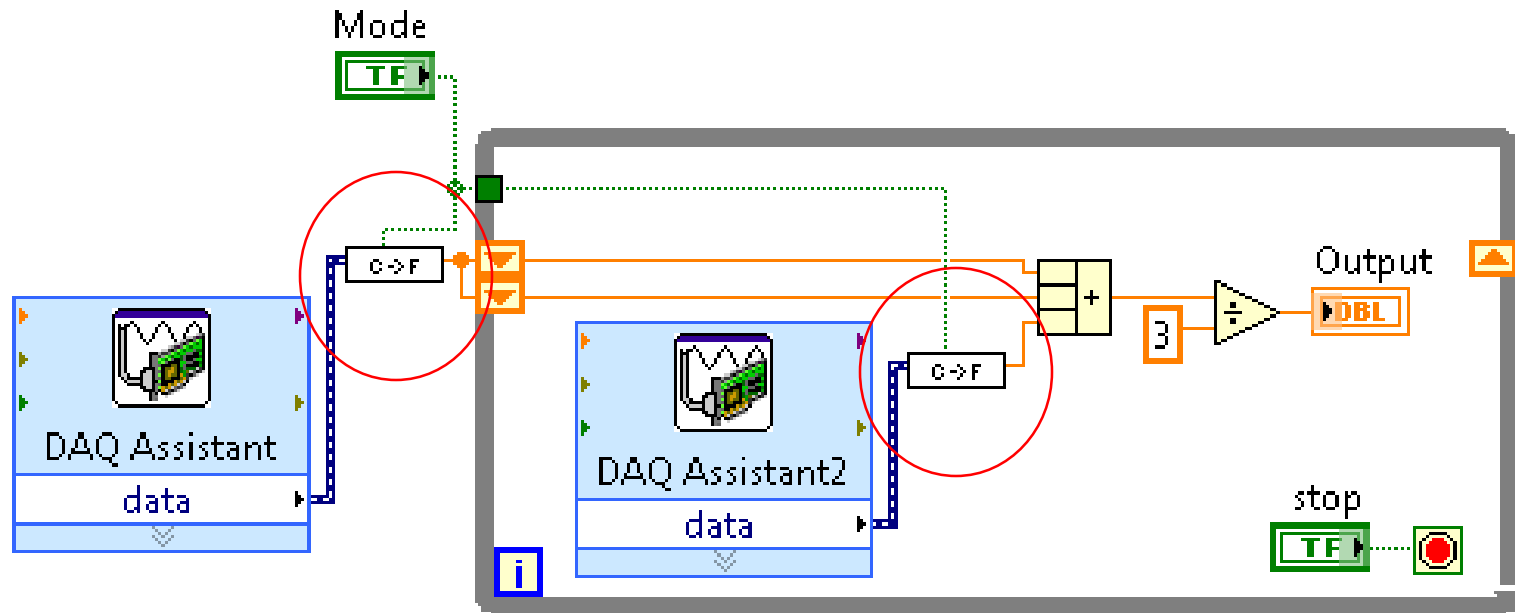
- SubVIs correspond to subroutines in text-based programming languages.
- The upper-right corner of the front panel and block diagram displays the icon for the VI.
- This icon identifies the VI when you place the VI on a block diagram.

# Understanding Modularity – SubVIs



Repeated code can become subVIs.

# Understanding Modularity – SubVIs



# SubVIs

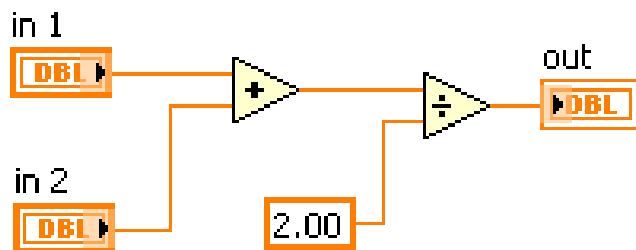
## Function Pseudo Code

```
function average (in1,  
    in2, out)  
{  
out = (in1 + in2)/2.0;  
}
```

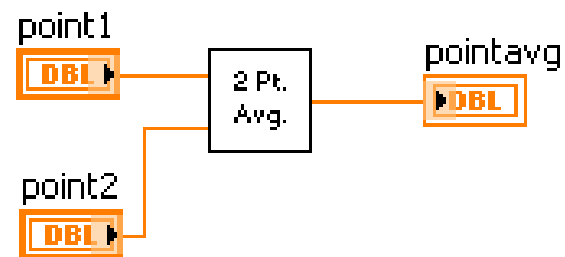
## Calling Program Pseudo Code

```
main  
{  
    average (point1, point2,  
        pointavg)  
}
```

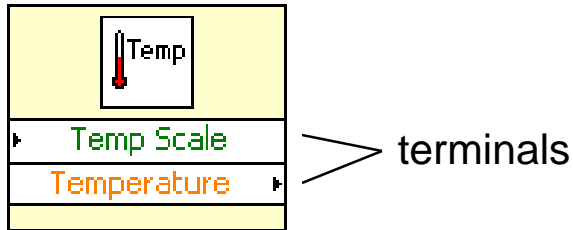
## SubVI Block Diagram



## Calling VI Block Diagram



# Icon/Connector



An icon represents a VI in other block diagrams

A connector passes data to and receives data from a subVI through terminals

**Icon**



**Connector**

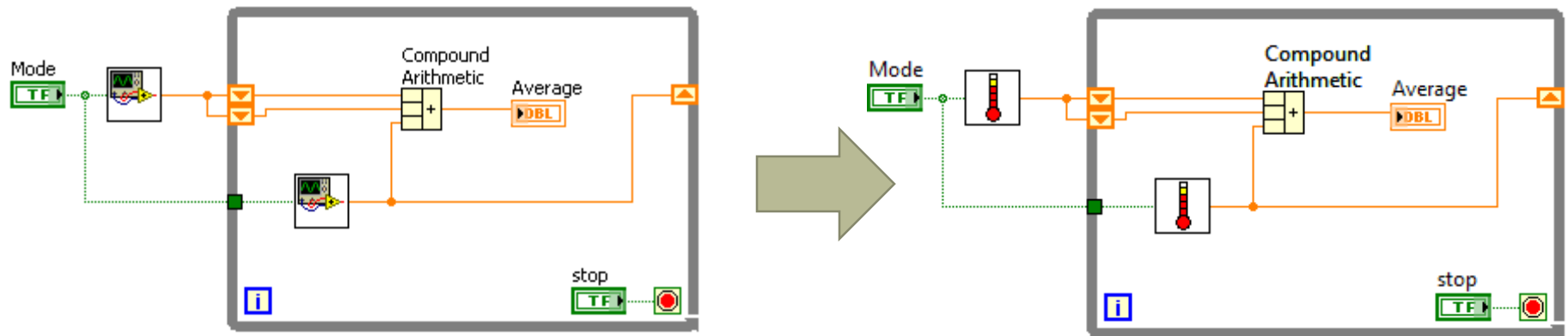


terminals

# Characteristics of a Good Icon

Good icons convey the functionality of the VI using:

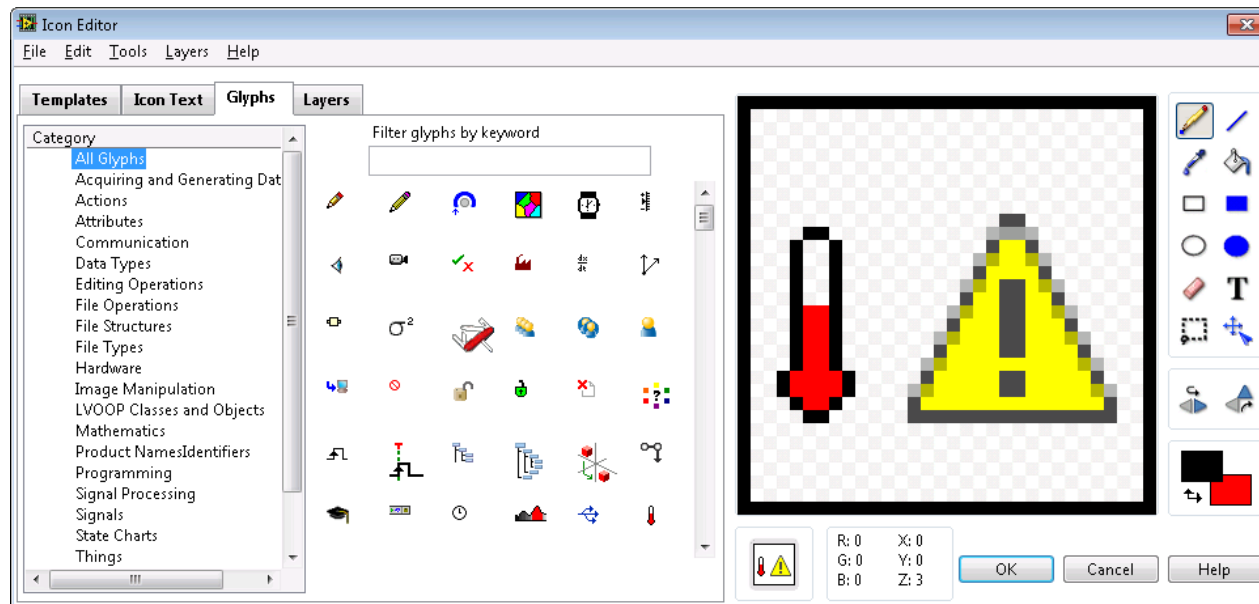
- Relevant graphics
- Descriptive text, if necessary



# Creating Icons - Icon Editor

Open the Icon Editor using one of these methods:

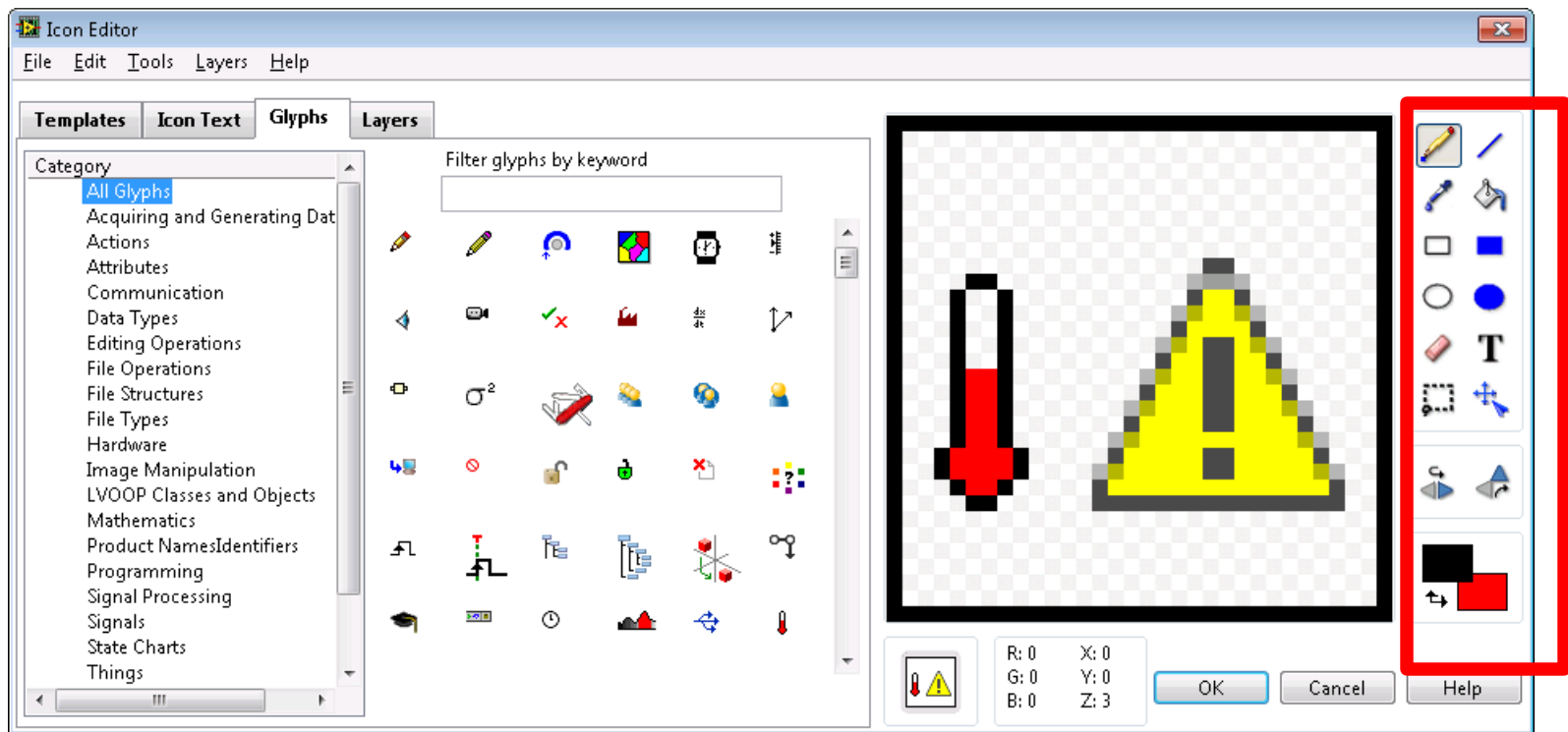
- Right-click the icon in the upper-right corner of the front panel or block diagram and select **Edit Icon**.
- Double-click the icon.





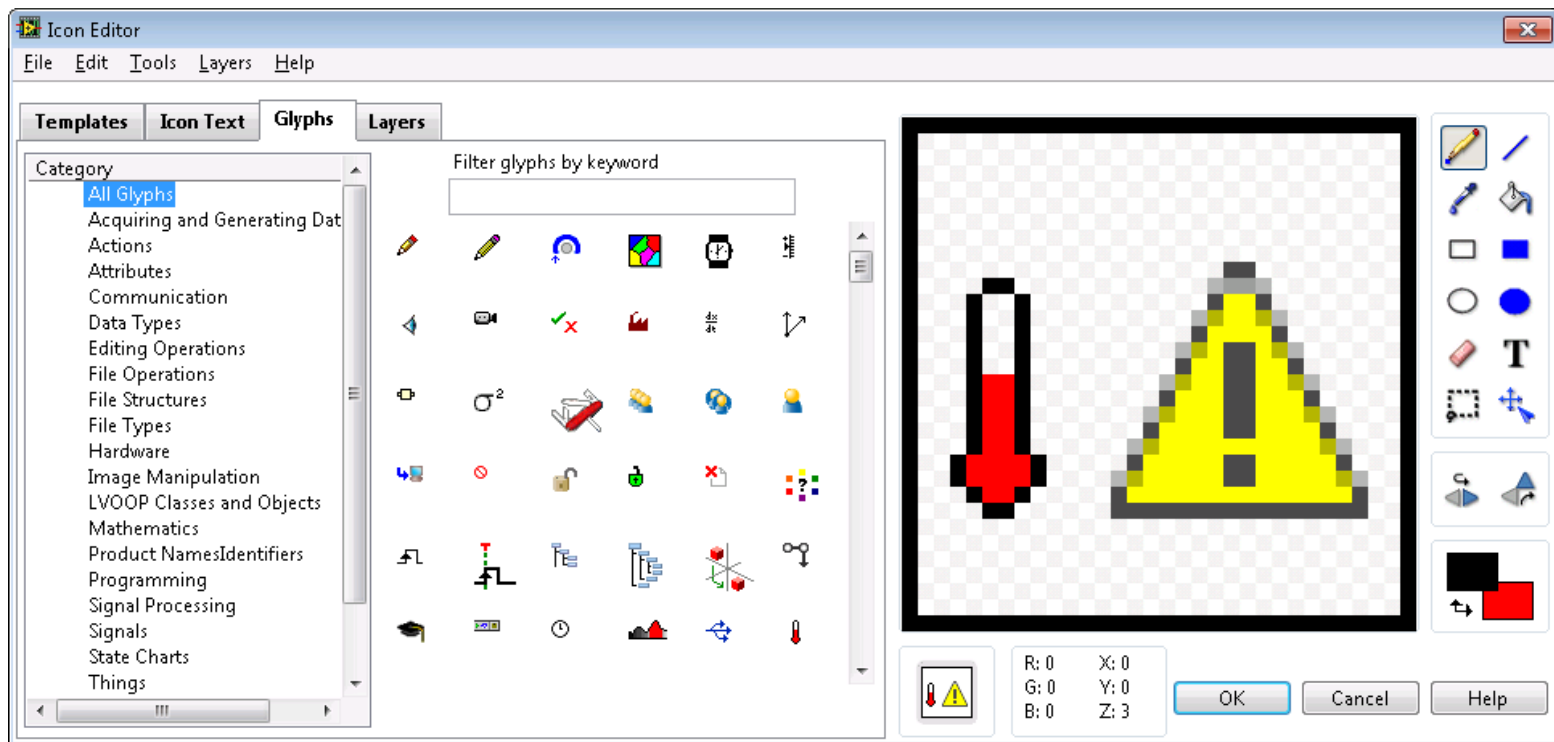
# Icon Editor

Use the editing tools to modify an icon manually.



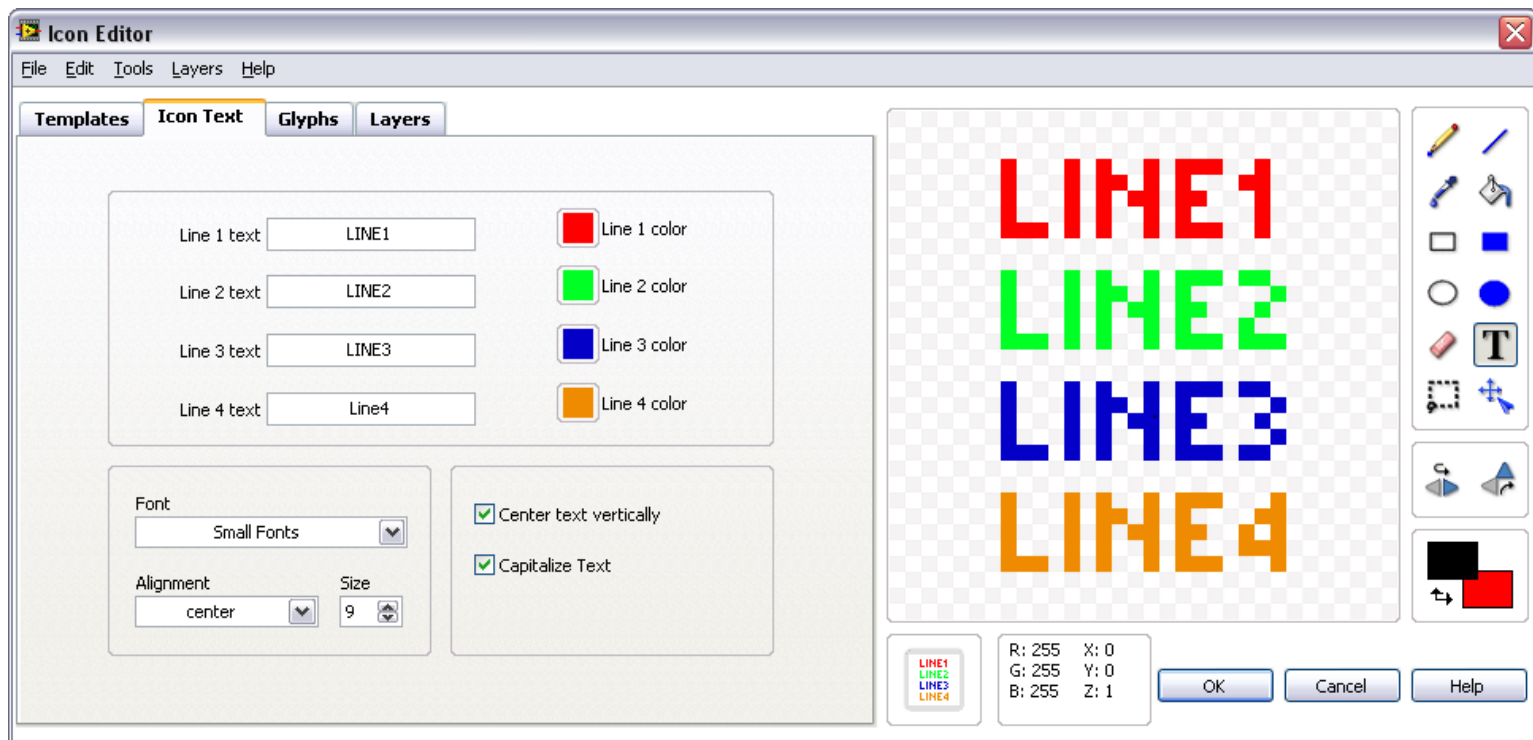
# Icon Editor

Use the **Glyphs** tab to display glyphs you can include in the icon.



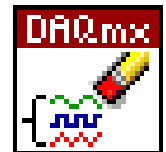
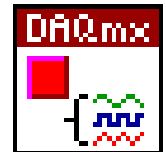
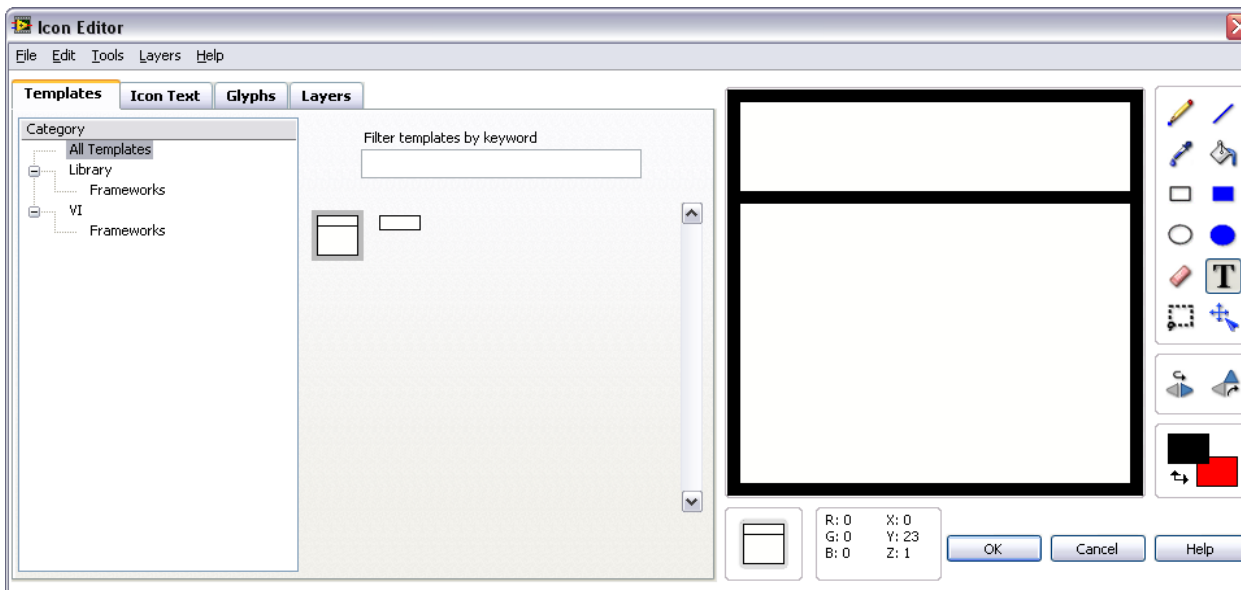
# Icon Editor

Use the **Icon Text** tab to specify the text to display in the icon.



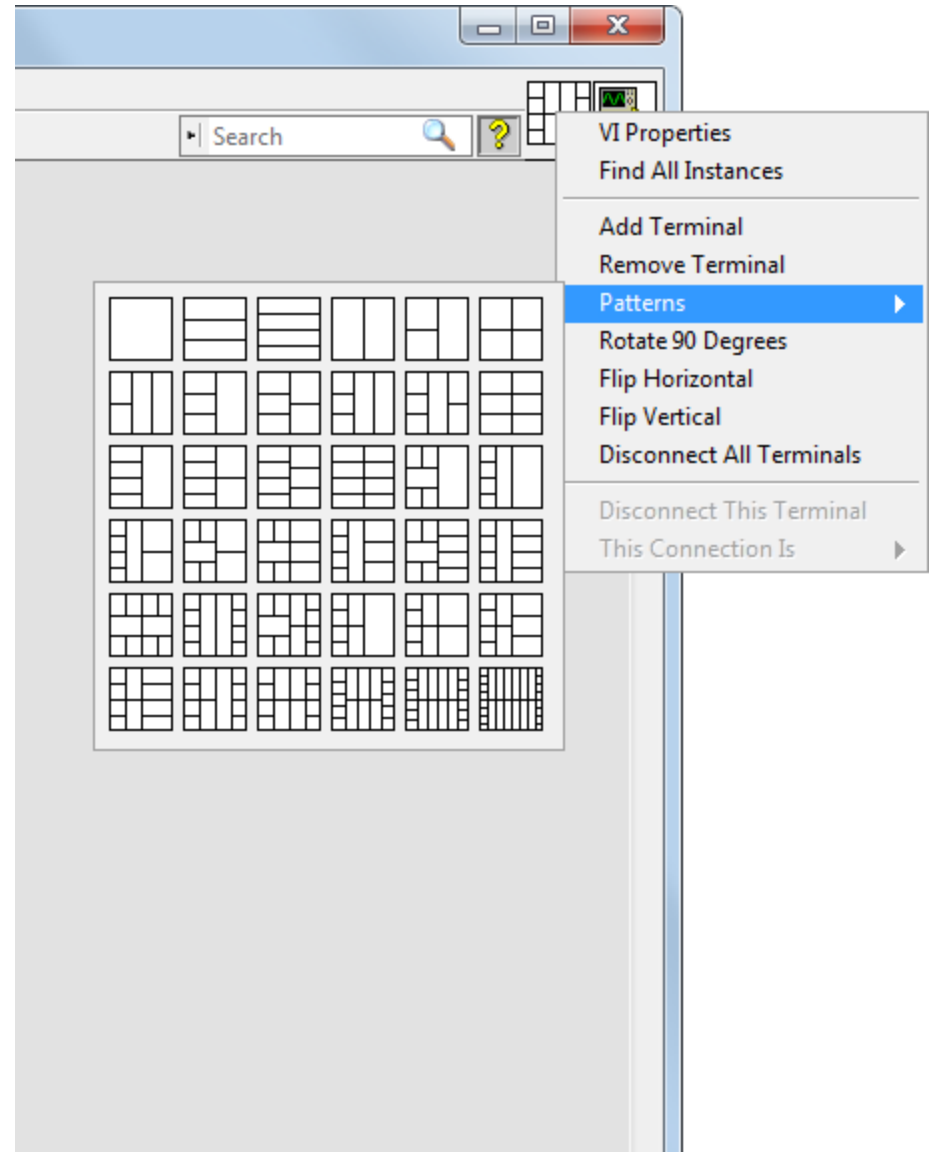
# Icon Editor

- Use the **Templates** tab to display icon templates you can use as a background for the icon.



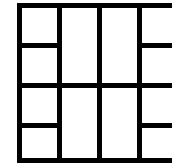
# Connector Pane

- The connector pane is displayed next to the icon in the upper right corner of the front panel.
  - Each rectangle on the connector pane represents a terminal.
  - Use the terminals to assign inputs and outputs.
- Select a different pattern by right-clicking the connector pane and selecting **Patterns** from

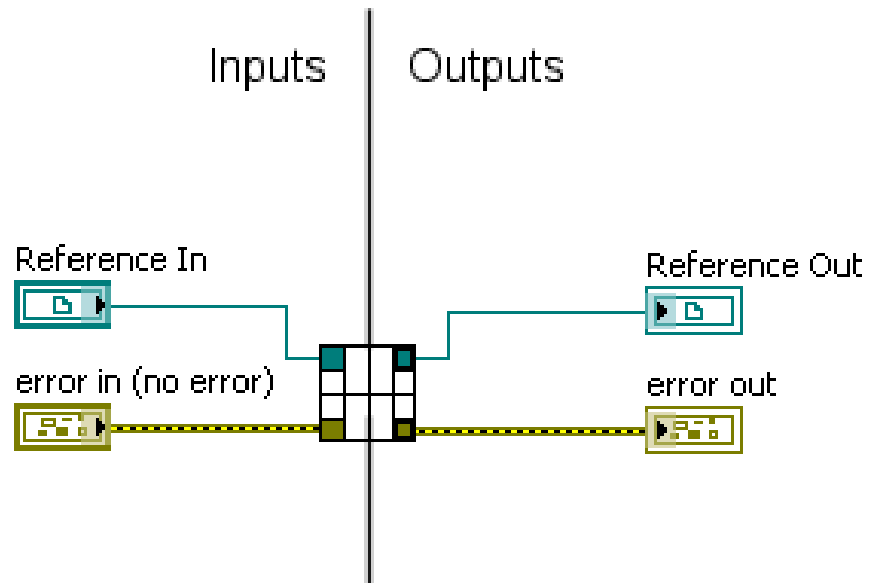


# Connector Pane – Standards

- Use this connector pane layout as a standard.

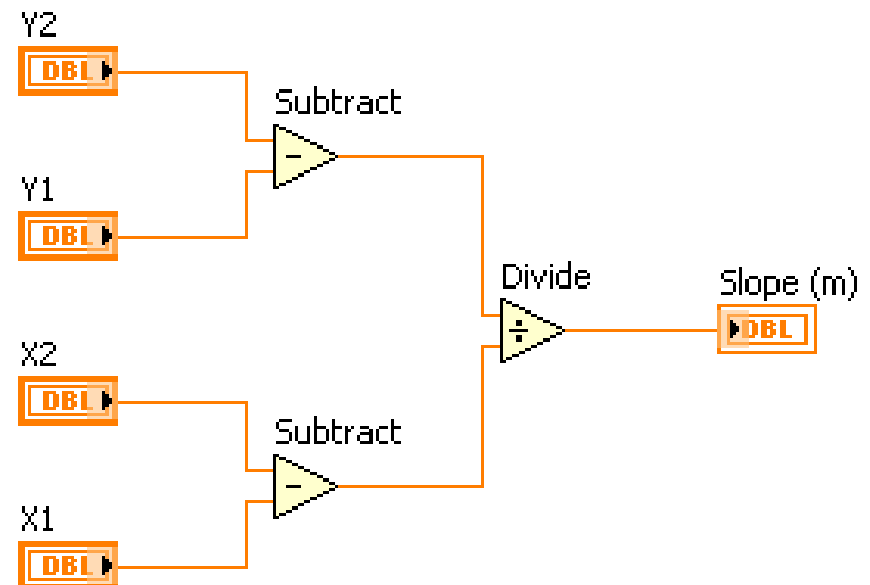


- Top terminals are usually reserved for references, such as a file reference.
- Bottom terminals are usually reserved for error clusters.



# SubVI Example – Calculating Slope

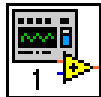
- A VI within another VI is called a subVI
- To use a VI as a subVI, create an icon and a connector pane after building the front panel and block diagram



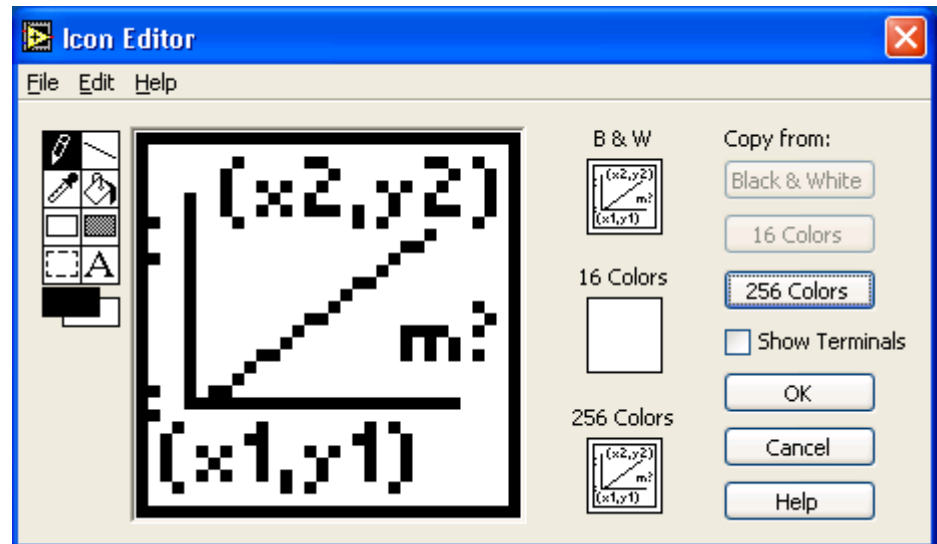
# Creating the Icon

- Icon: graphical representation of a VI
- Right-click in the icon pane (Panel or Diagram)
- Always create a black and white icon

Default Icon

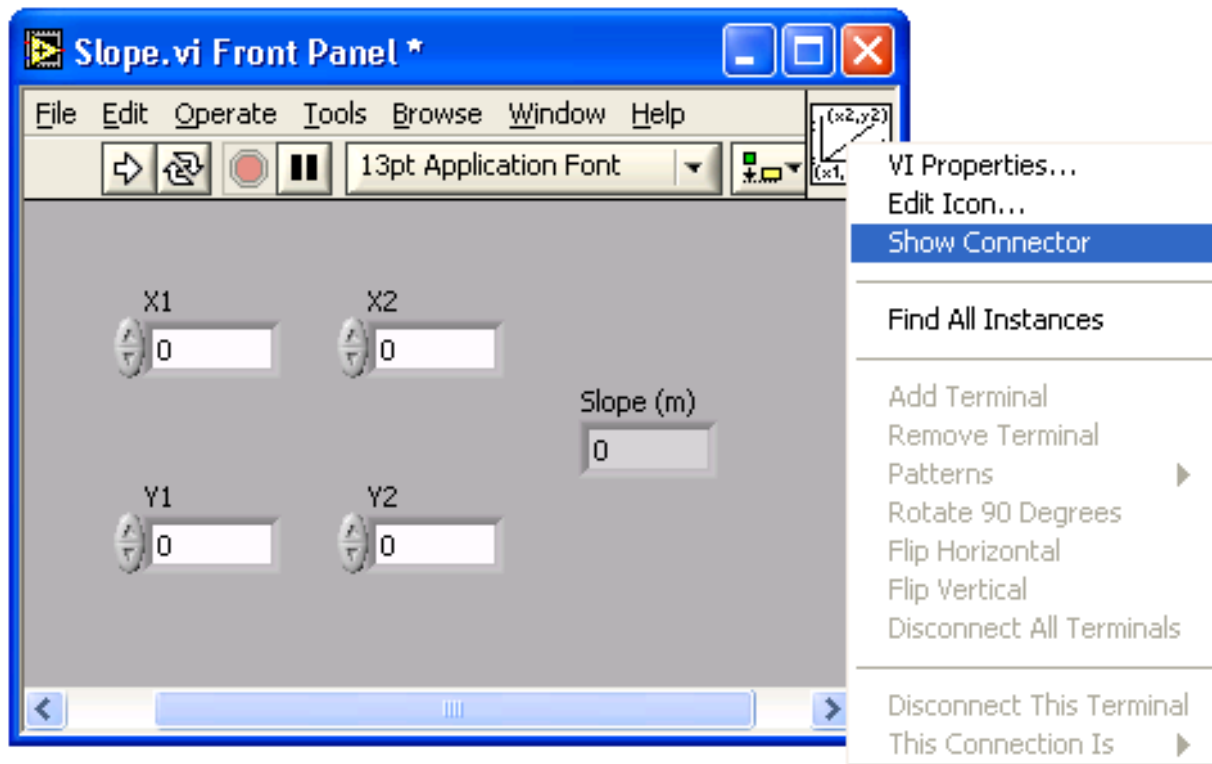


Create a custom icon





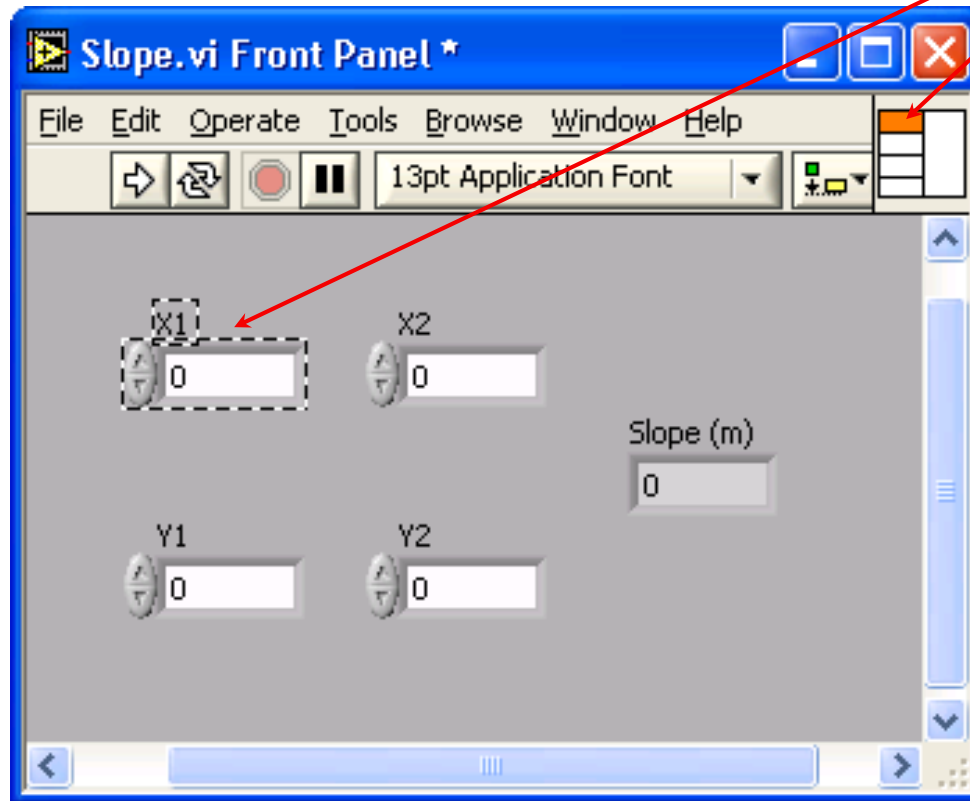
# Creating the Connector



Right click the icon  
(only)

# Creating the Connector - continued

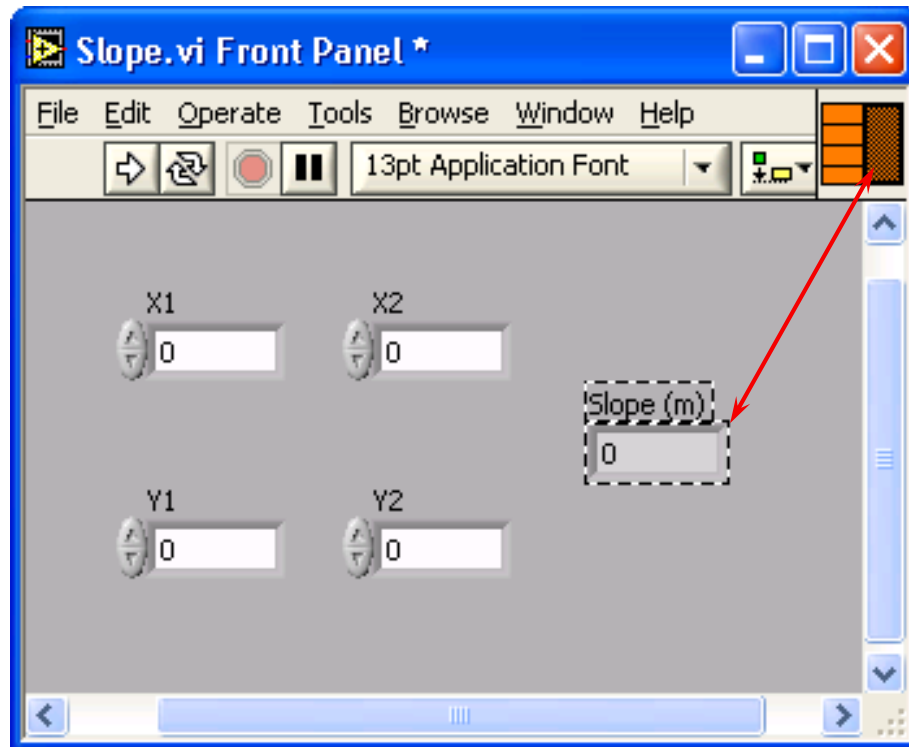
Click with  
wiring tool



# The Connector Pane

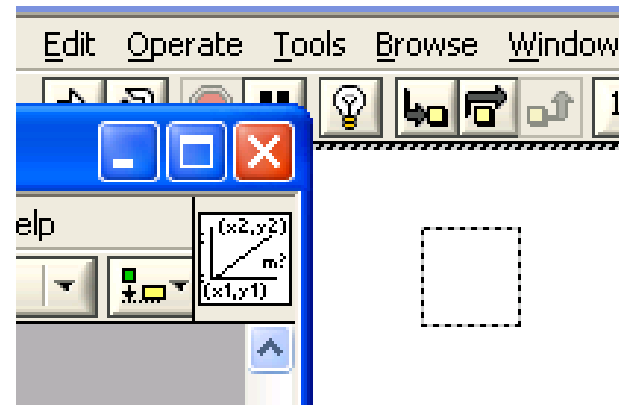
Terminal colors match the data types to which they are connected

Click the terminal to see its associated front panel object



# Using SubVIs

- Options to place a subVI on the block diagram:
  - Drag the VI from the Project Explorer to the block diagram.
  - Click **Select a VI** on the **Functions** palette and then navigate to the VI.
  - Drag the icon from an open VI to the block diagram of another VI.



# Terminal Settings

## –Bold

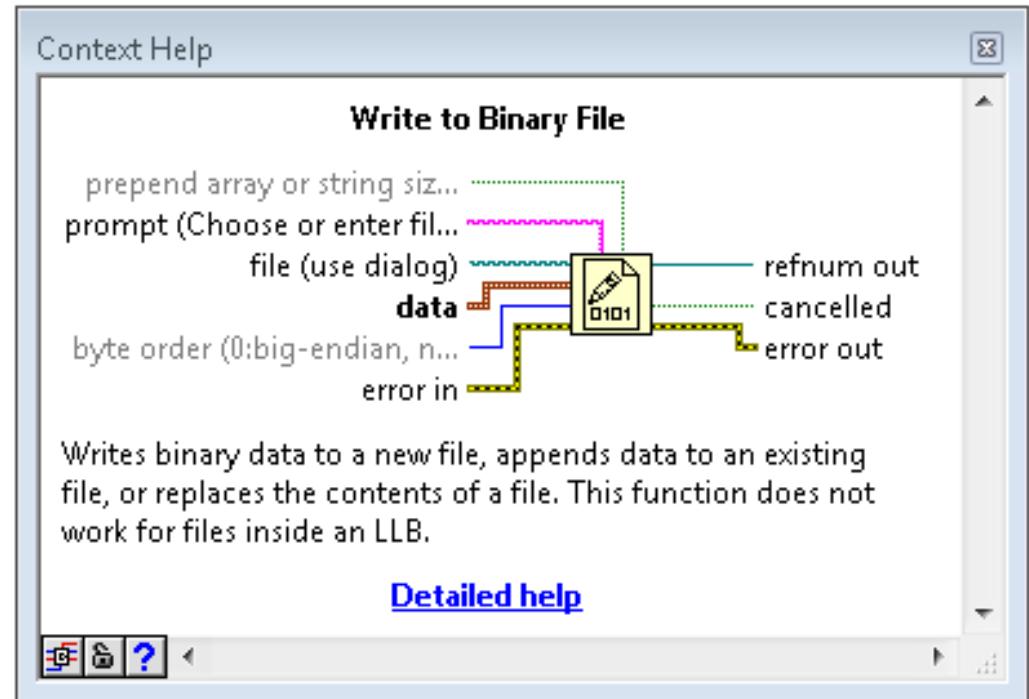
- Required terminal

## –Plain

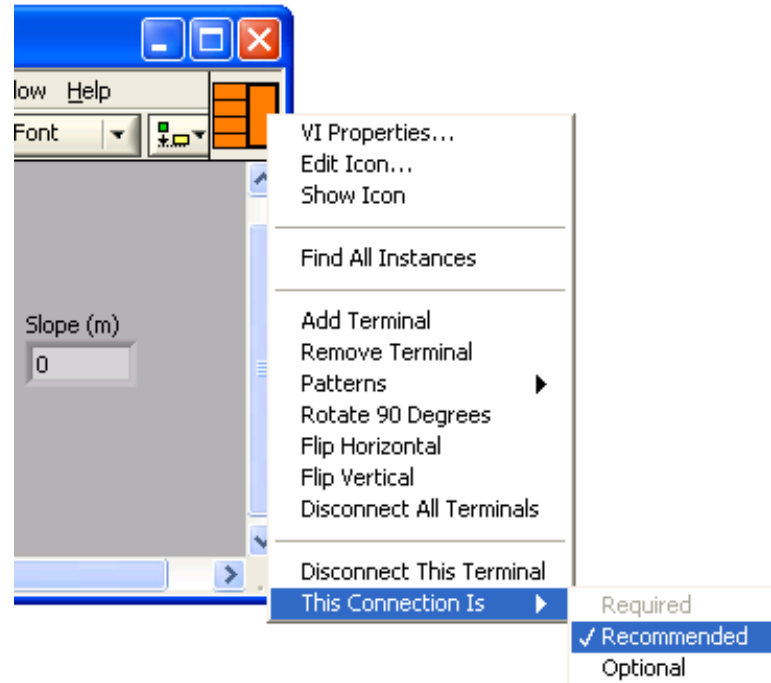
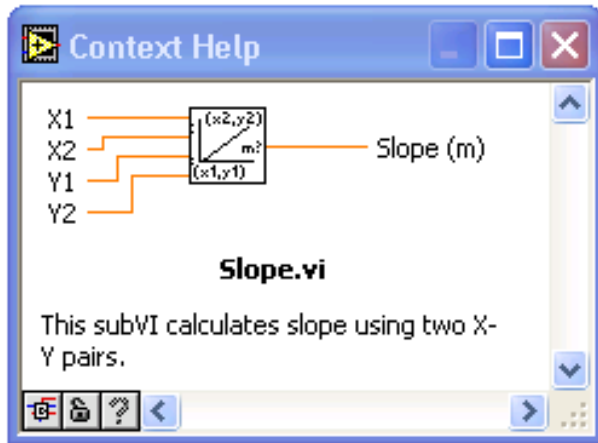
- Recommended terminal

## –Dimmed

- Optional terminal



# Help and Classifying Terminals

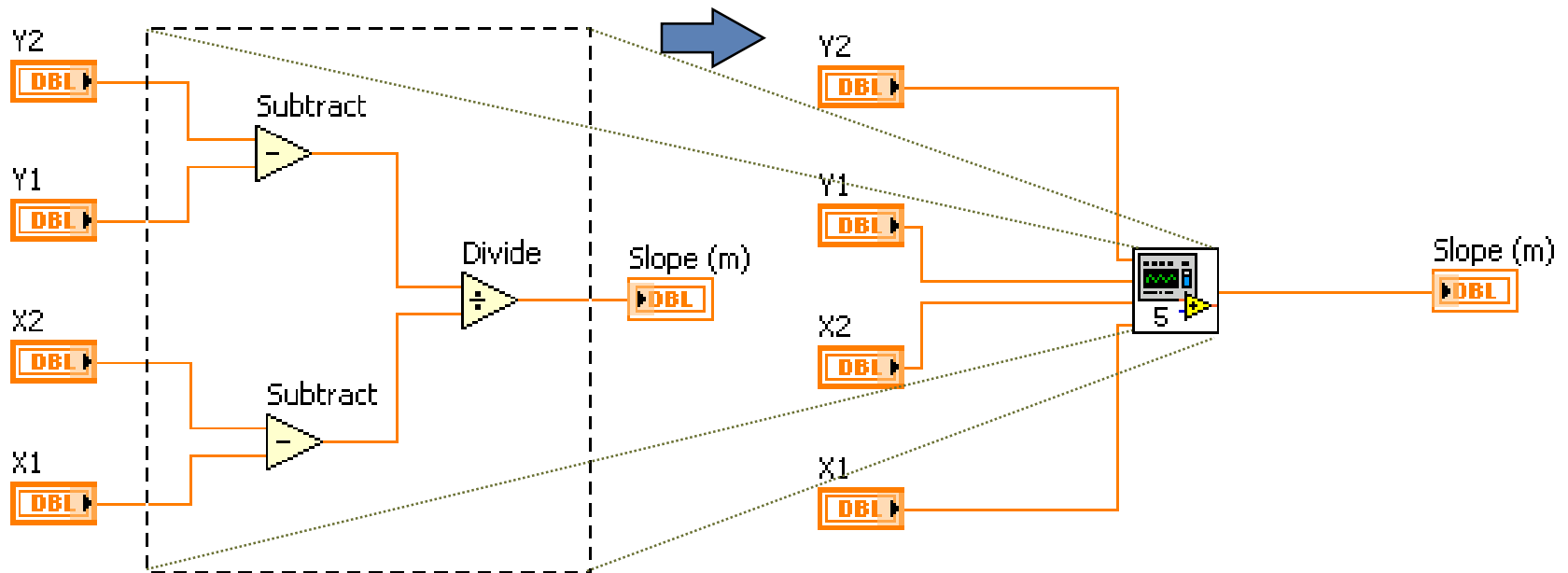


Classify inputs and outputs:

- **Bold** : Required — Error if no connection
- Plan: Recommended — Warning if no connection
- Dimmed: Optional — No effect if no connection

# Create SubVI Option

- Enclose area to be converted into a subVI
- Select Create SubVI from the Edit Menu



# Summary

- Place controls (inputs) and indicators (outputs) in the front panel window
- Use the Operating tool to manipulate panel objects. Use the Positioning tool to select, move, and resize panel objects. Use the Wiring tool to connect diagram objects
- Control terminals have thicker borders than indicator terminals
- All front panel objects have property pages and shortcut menus
- Wiring is the mechanism to control dataflow and produce LabVIEW programs
- Broken Run arrow means a nonexecutable VI
- Various debugging tools and options available such as setting probes and breakpoints, execution highlighting, and single stepping



# Summary

- Use Express VIs, standard VIs and functions on the block diagram to create your measurement code. For the most common requirements, use Express VIs with interactive configuration dialogs to define your application.
- VIs can be used as subVIs after you make the icon and connector
- Icon created using Icon Editor
- Connector defined by choosing number of terminals
- Load subVIs using the Select a VI option in the All Functions palette or dragging the icon onto a new diagram
- Online help for subVIs using the Show Context Help option
- Descriptions document functionality
- Use Create SubVI feature to easily modularize the block diagram

# Tips

- Common keyboard shortcuts

<Ctrl-R>	Run a VI
<Ctrl-F>	Find object
<Ctrl-H>	Activate Context Help window
<Ctrl-B>	Remove all broken wires
<Ctrl-W>	Close the active window
<Ctrl-E>	Toggle btwn Diagram/Panel Window

- Access Tools Palette with <shift>-right-click
- Increment/Decrement faster using <shift> key
- Tools»Options selection — set preferences in LabVIEW
- VI Properties (File menu)