NATIONAL INSTRUMENTS

Developing Modular Applications

TOPICS

A. Understanding Modularity

B. Icon

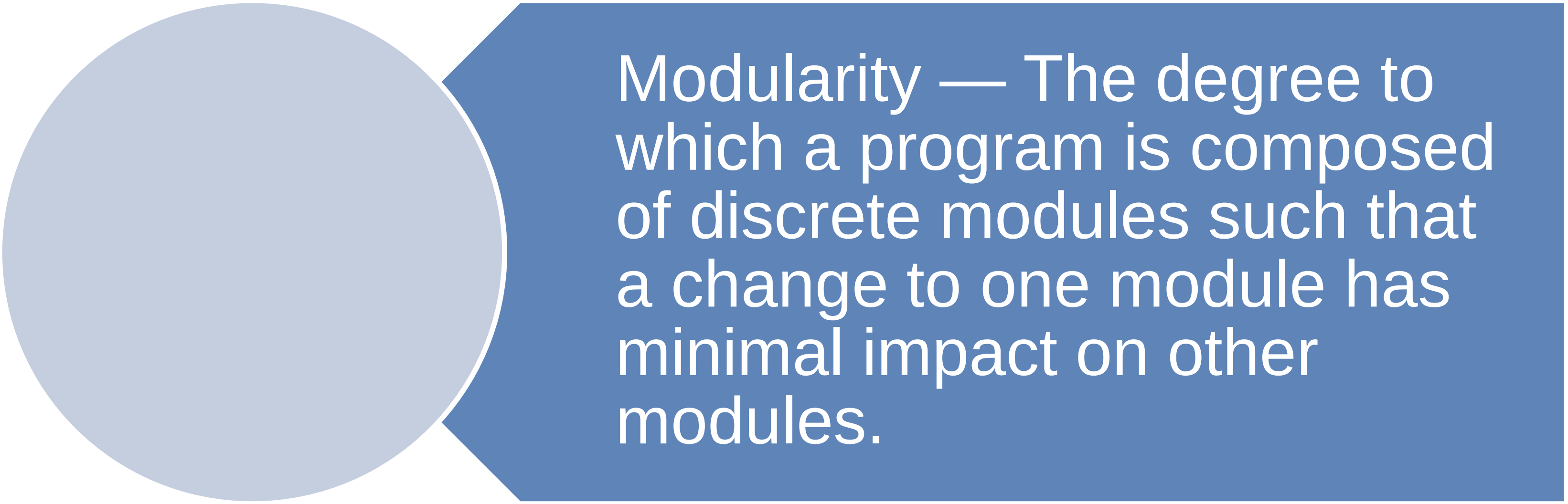
C. Connector Pane

D. Using SubVIs

E. Documenting Code

A. Understanding Modularity

Understanding Modularity



Modularity — The degree to which a program is composed of discrete modules such that a change to one module has minimal impact on other modules.

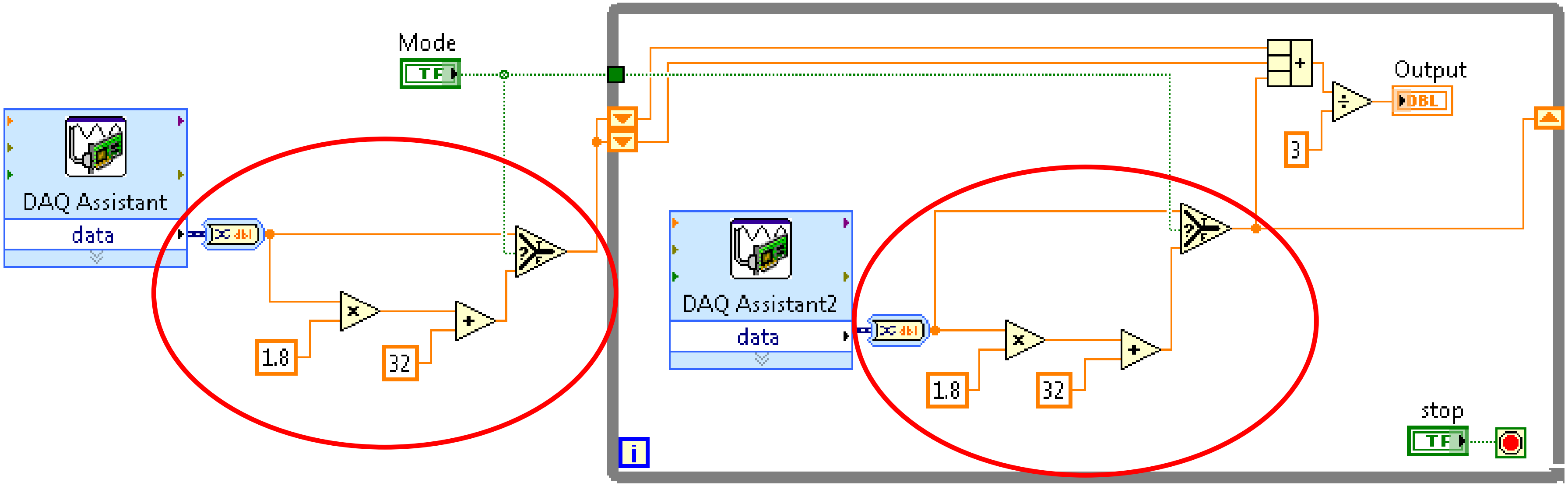
LabVIEW uses subVIs to create modularity.

Understanding Modularity – SubVIs

SubVI — a VI within another VI

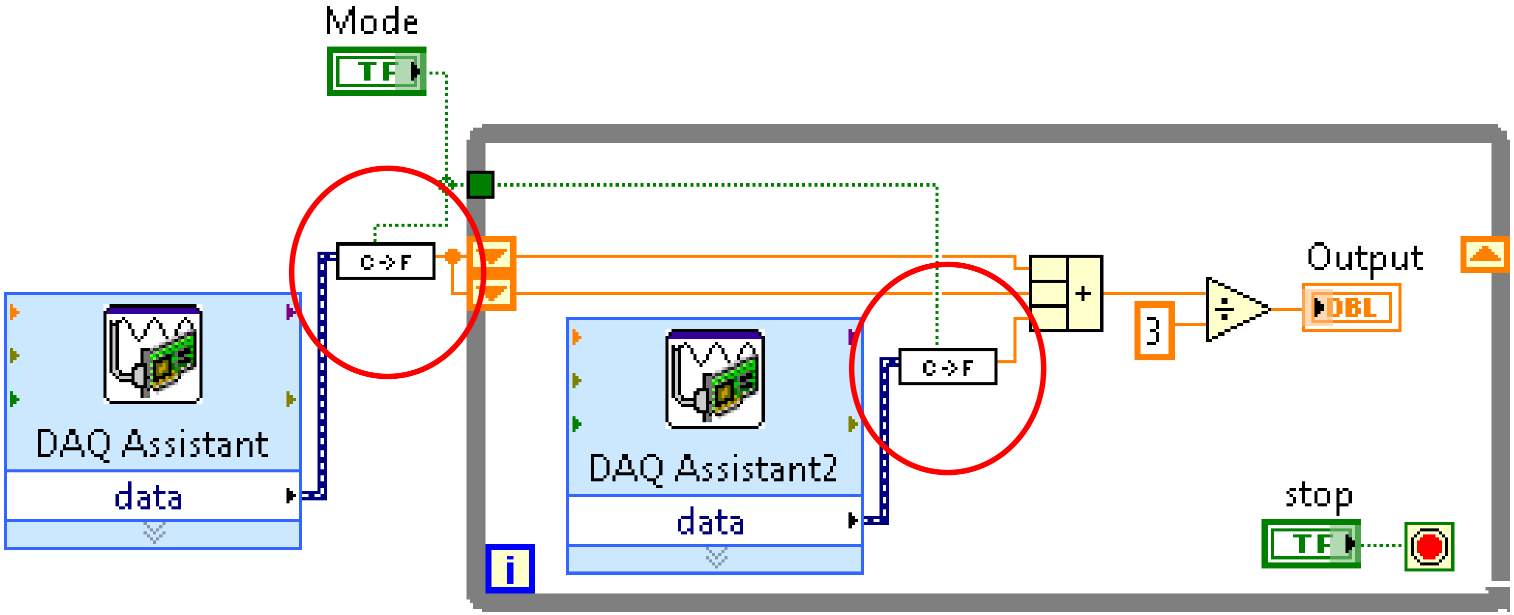
- SubVIs correspond to subroutines in text-based programming languages.
- The upper-right corner of the front panel and block diagram displays the icon for the VI.
- This icon identifies the VI when you place the VI on a block diagram.

Understanding Modularity – SubVIs



Repeated code can become subVIs.

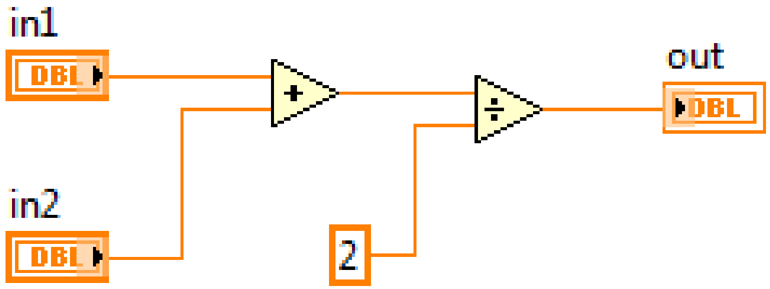
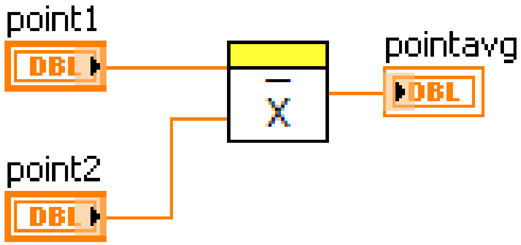
Understanding Modularity – SubVIs



Understanding Modularity – SubVIs

Function Code	Calling Program Code

`function average (in1, in2, out)`
`{`
`out = (in1 + in2)/2.0;`
`}`
	`main`
	`{`
	`average (point1, point2, pointavg)`
	`}`

SubVI Block Diagram	Calling VI Block Diagram
	

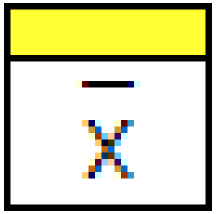
B. Icon

Characteristics of a Good Icon

Using the Icon Editor

Icon

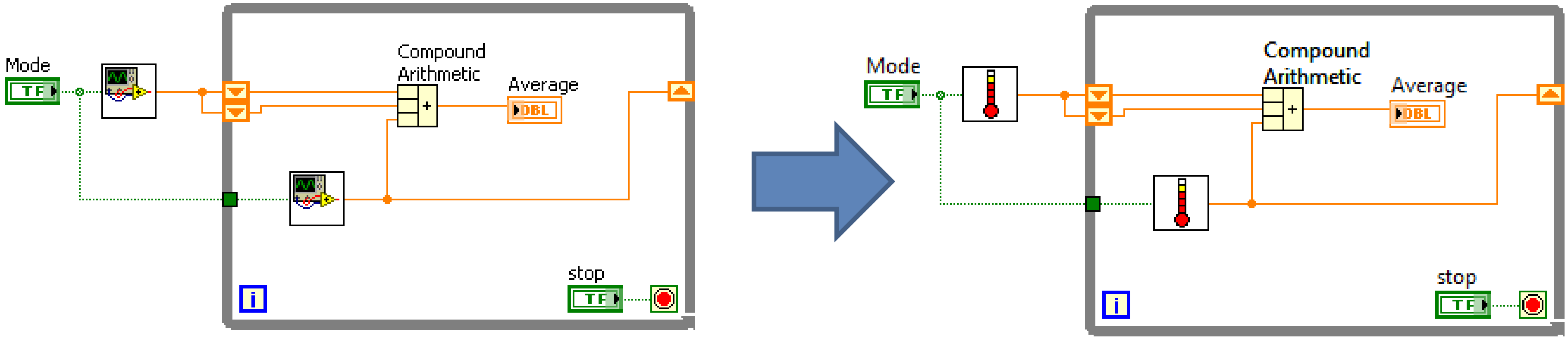
- An icon is a graphical representation of a VI.
- If you use a VI as a subVI, the icon identifies the subVI on the block diagram of the VI.



Characteristics of a Good Icon

Good icons convey the functionality of the VI using:

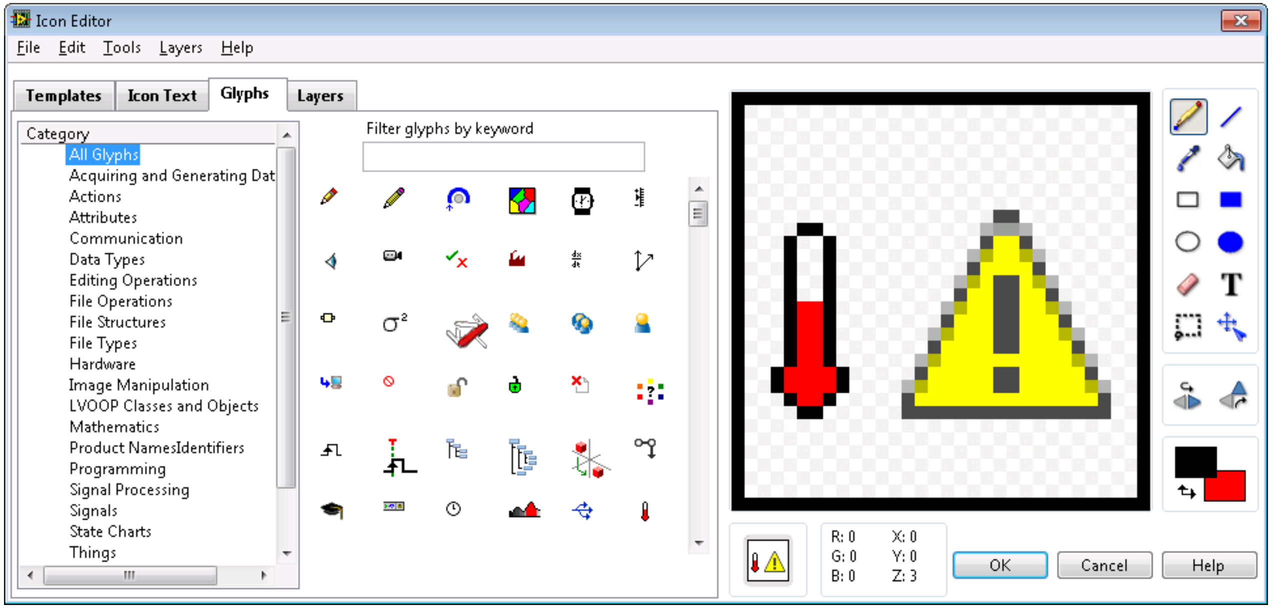
- Relevant graphics
- Descriptive text, if necessary



Creating Icons - Icon Editor

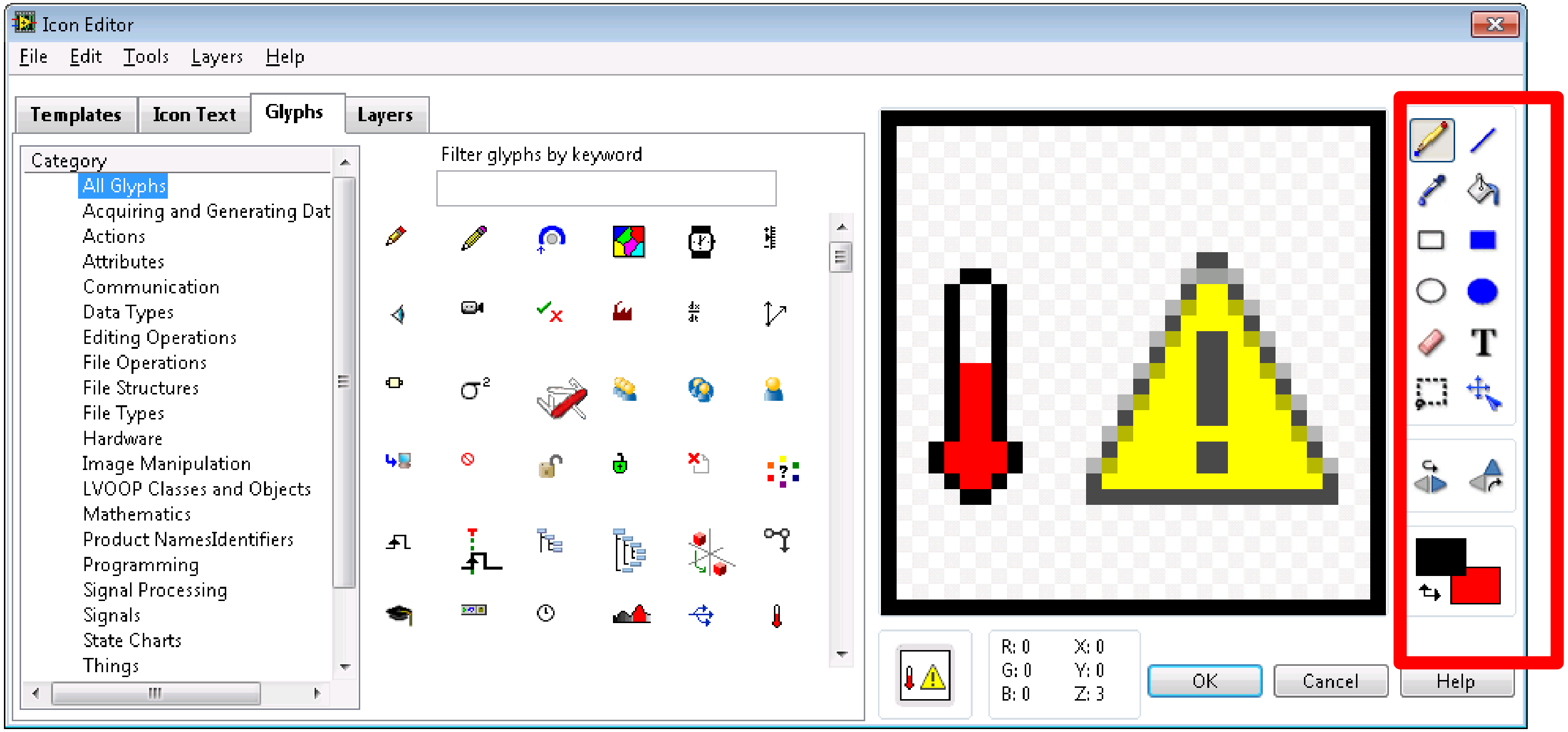
Open the Icon Editor using one of these methods:

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



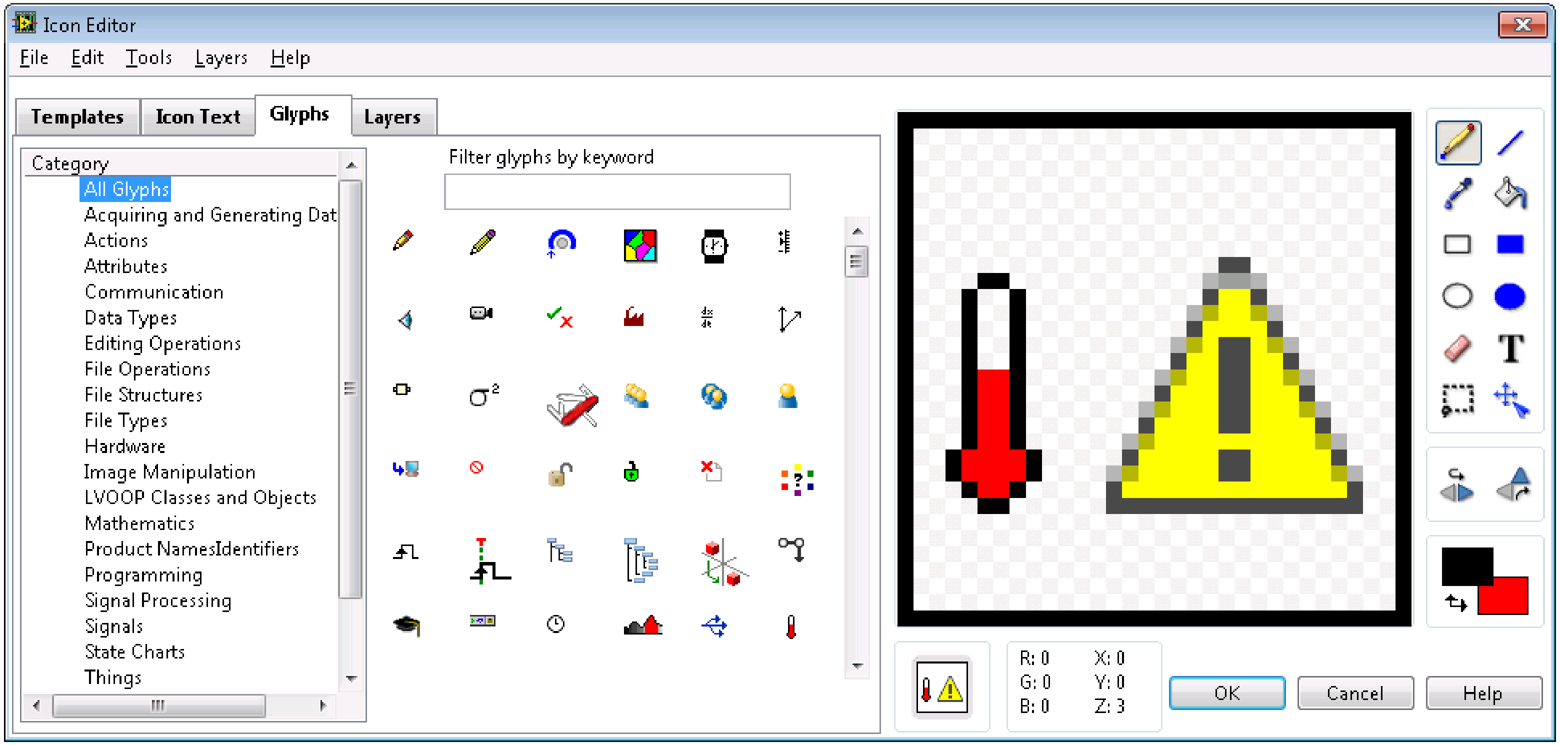
Icon Editor

Use the editing tools to modify an icon manually.



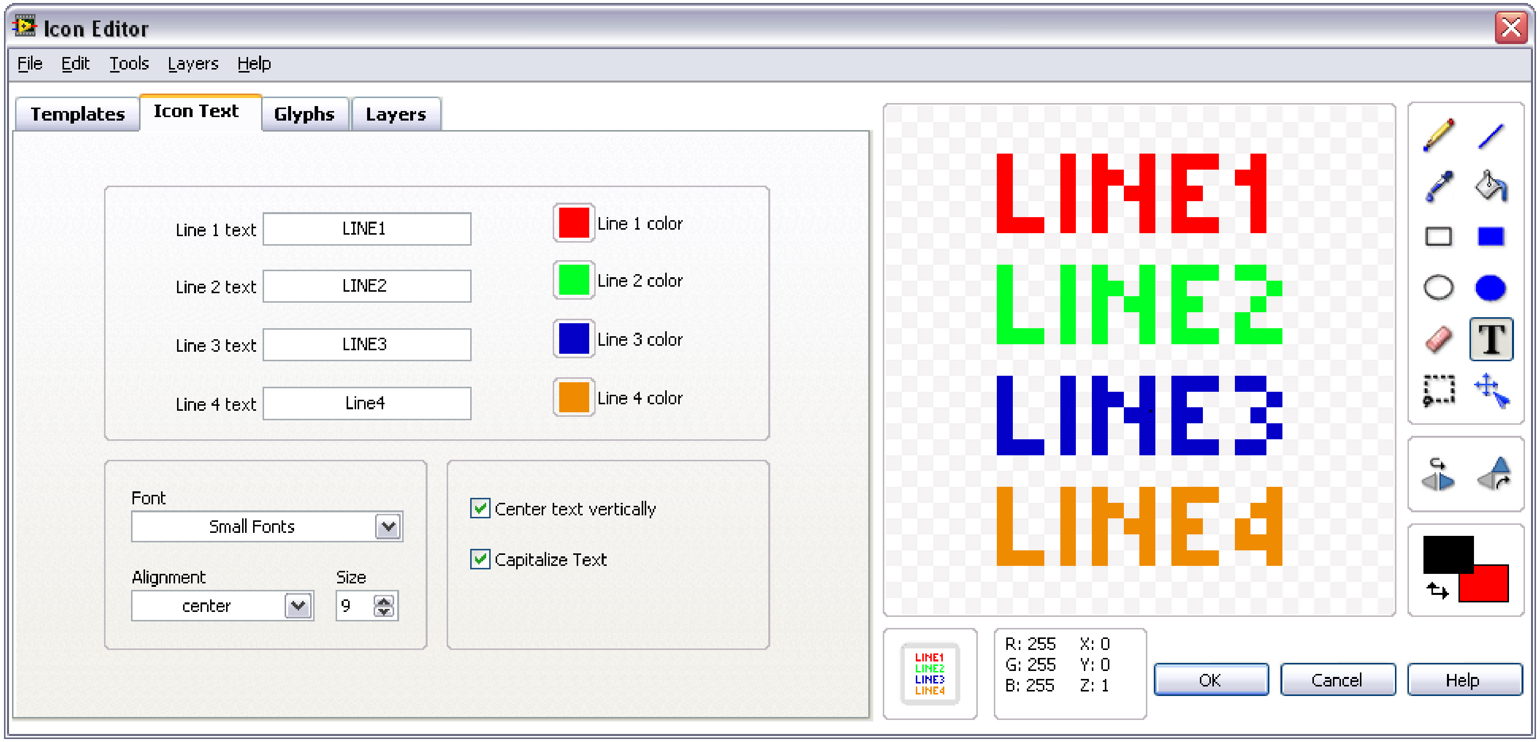
Icon Editor

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.



Line 1 text: LINE1 — Line 1 color 🟥

Line 2 text: LINE2 — Line 2 color 🟩

Line 3 text: LINE3 — Line 3 color 🟦

Line 4 text: Line4 — Line 4 color 🟧

Font: Small Fonts

Alignment: center

Size: 9

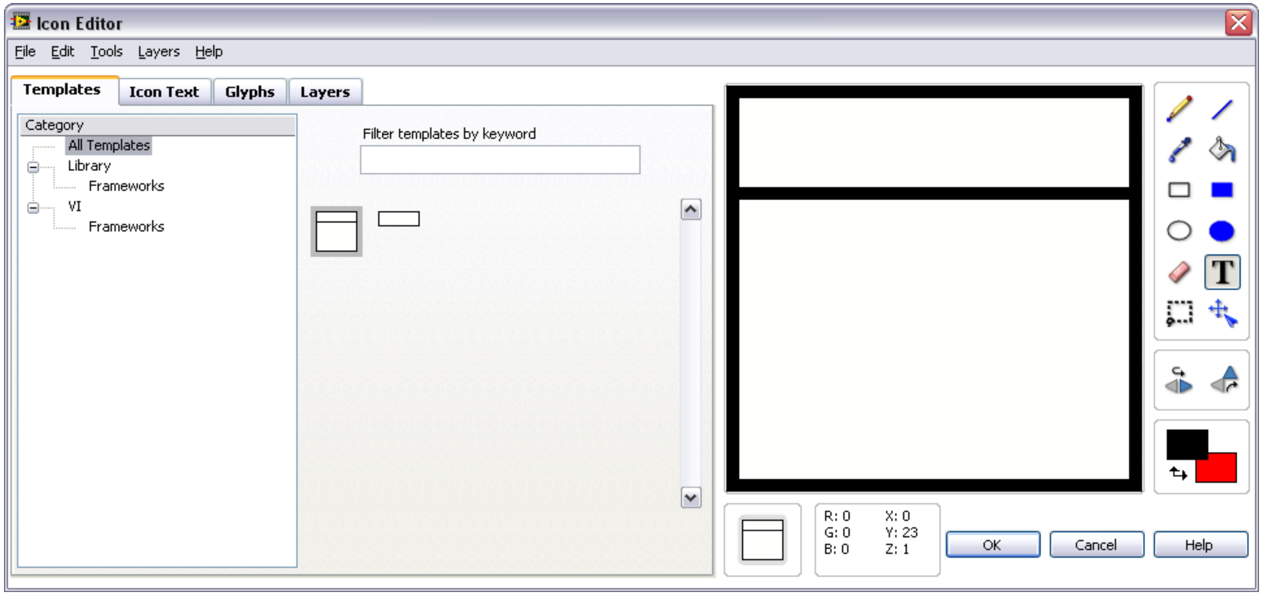
☑ Center text vertically

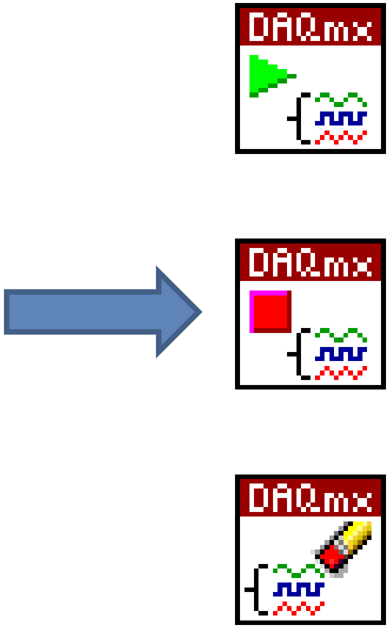
☑ Capitalize Text

R: 255 X: 0
G: 255 Y: 0
B: 255 Z: 1

Icon Editor

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





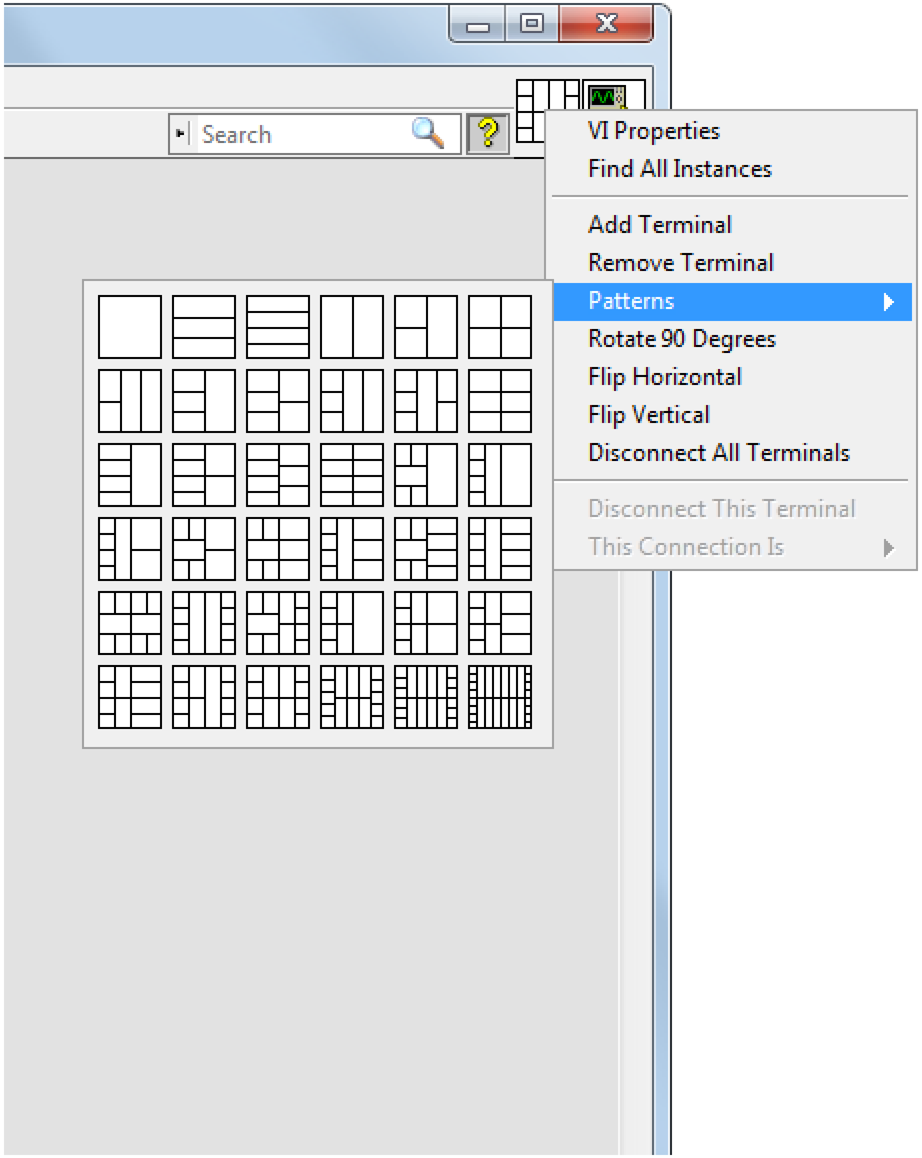
C. Connector Pane

Patterns

Standards

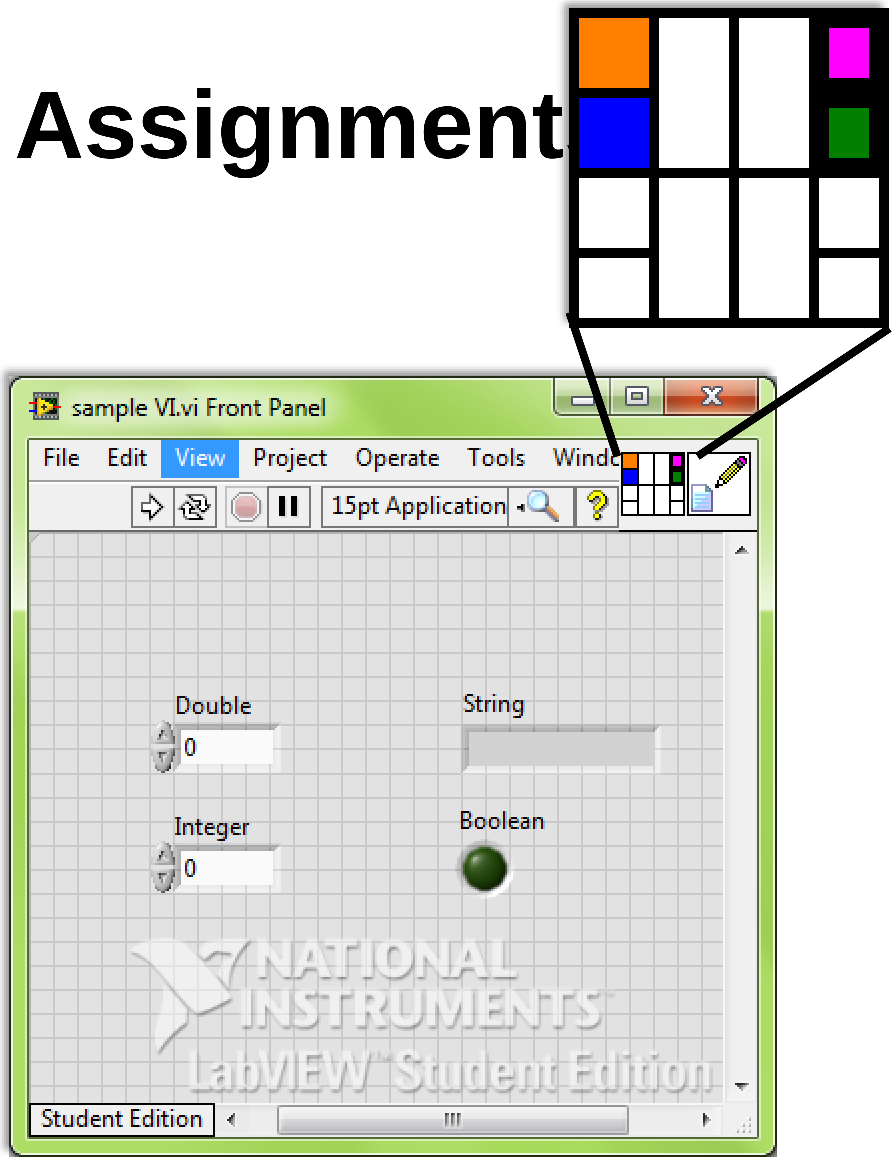
Connector Pane

- 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 the shortcut menu.



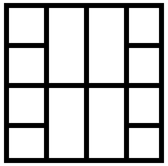
Connector Pane - Assignment

- Assign a front panel control or indicator to a connector pane terminal by clicking the terminal first and then clicking the control/indicator.
- The color of the terminal matches the data type of the connected control/indicator.



Connector Pane – Standards

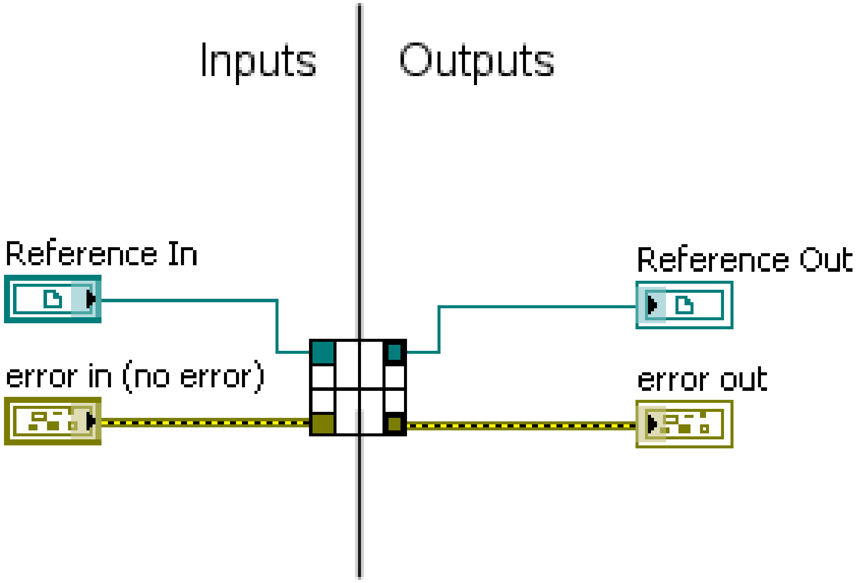
- Use this connector pane layout as a standard.



- Wire inputs (controls) to the left and outputs (indicators) to the right.

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

- Bottom terminals are usually reserved for error clusters.



D. Using SubVIs

Using on Block Diagram

Terminal Settings

Handling Errors

Creating from a Section of Code

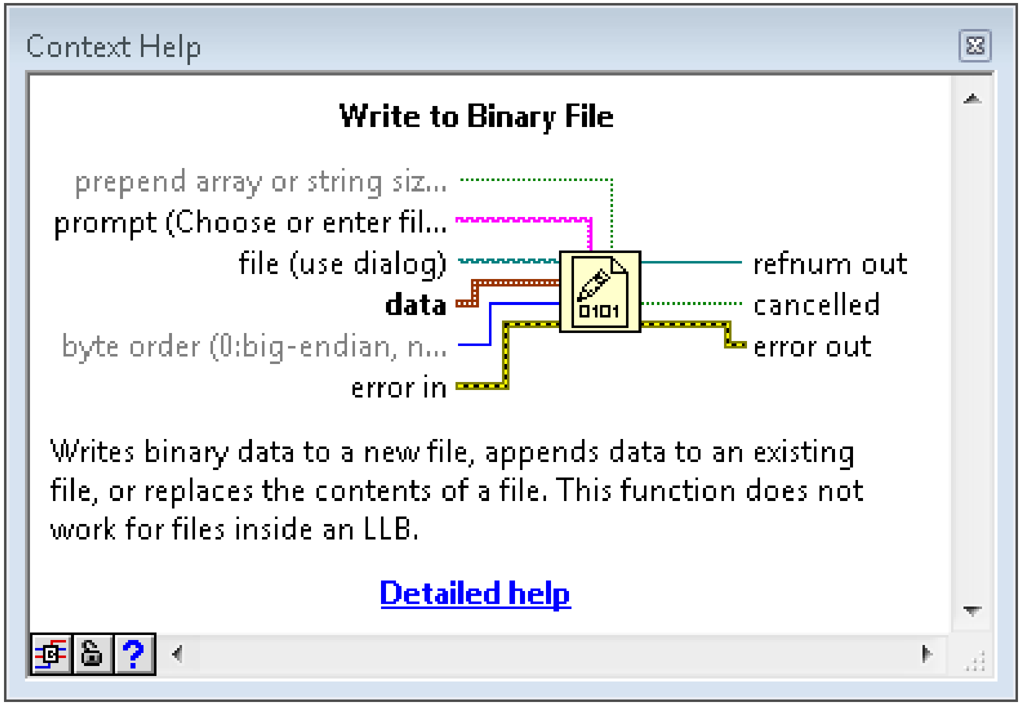
Using SubVIs

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



Context Help

Write to Binary File

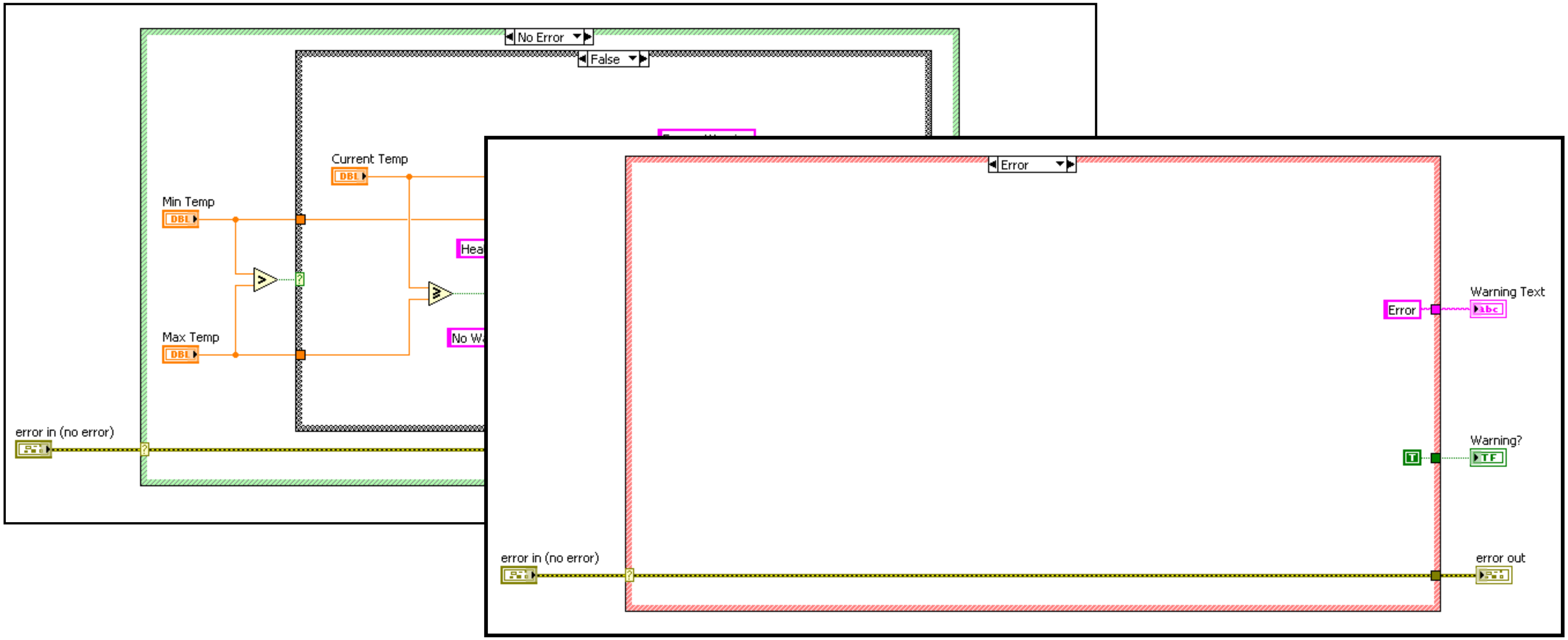
prepend array or string siz...
prompt (Choose or enter fil...
file (use dialog)
data
byte order (0:big-endian, n...
error in
refnum out
cancelled
error out
0101

Writes binary data to a new file, appends data to an existing file, or replaces the contents of a file. This function does not work for files inside an LLB.

Detailed help

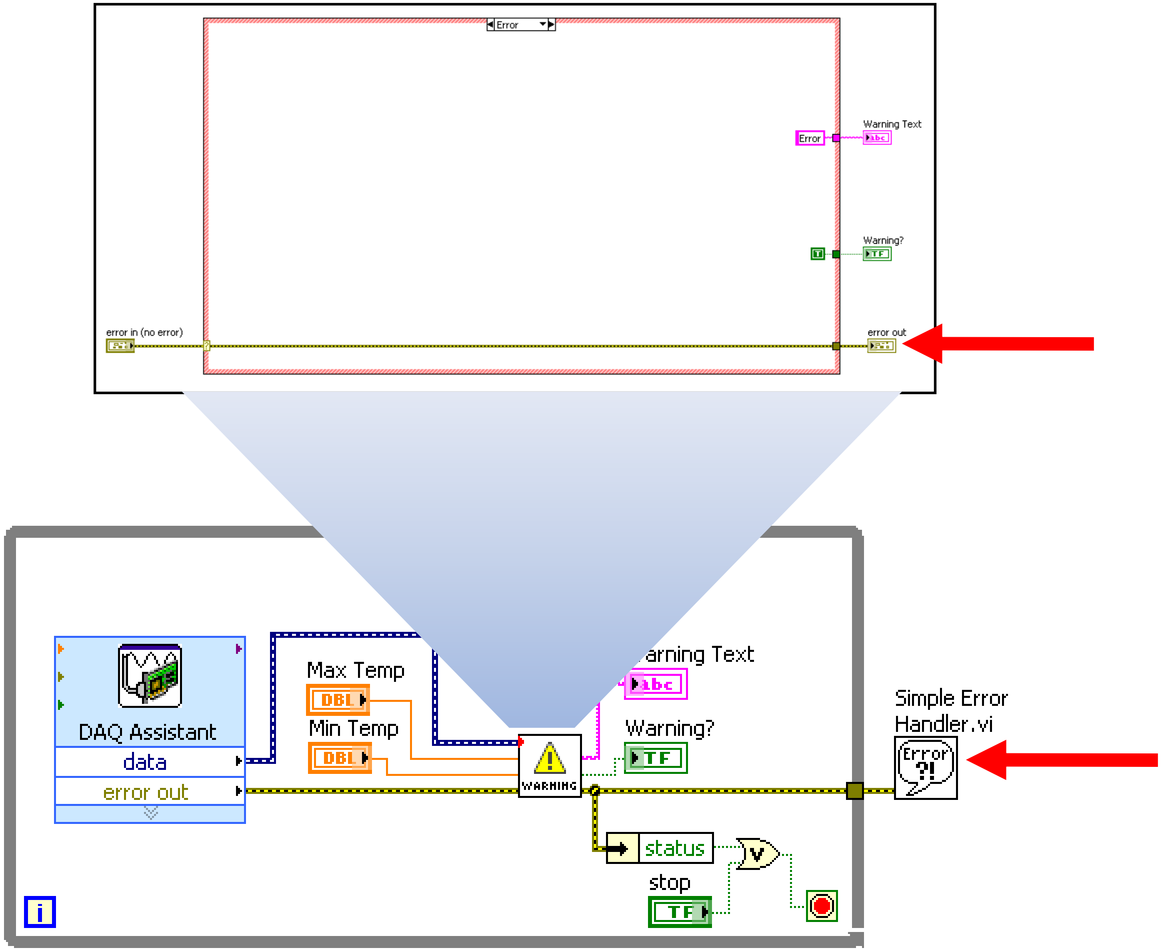
Handling Errors

Use a Case structure to handle errors passed into the subVI.



Handling Errors

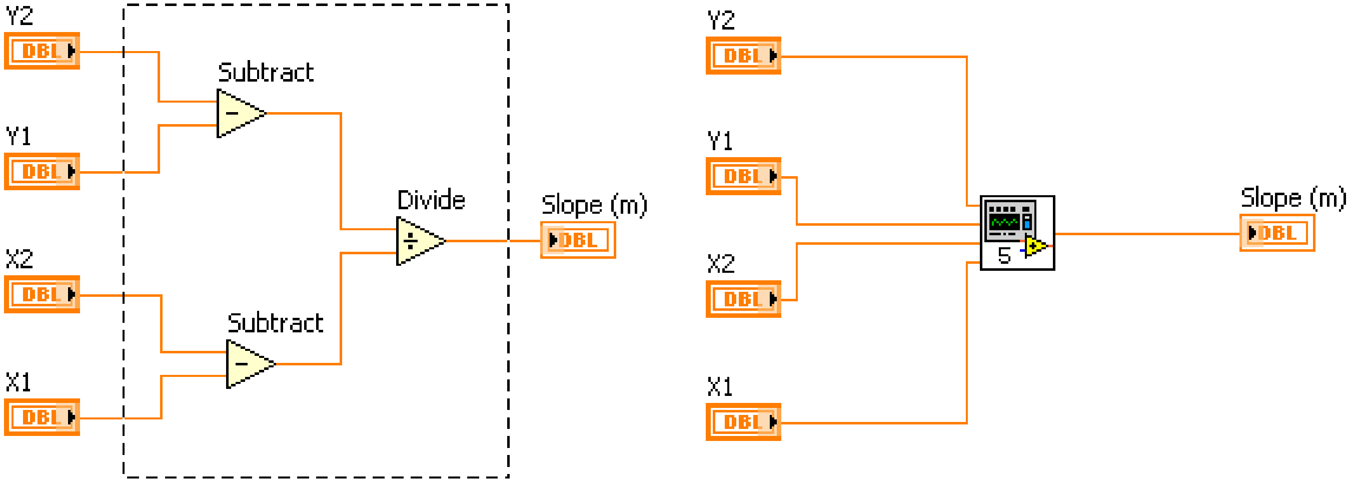
Avoid using LabVIEW error handler VIs inside subVIs.



Convert a Section of a VI to SubVI

To convert a section of a VI into a subVI:

1. Use the Positioning tool to select the section of the block diagram you want to reuse.
2. Select **Edit»Create SubVI**.



E. Documenting Code

Descriptions and Tip Strips

Documenting Block Diagram Code

Documenting Code

VI

- Name
- Description
- Icon

Front Panel

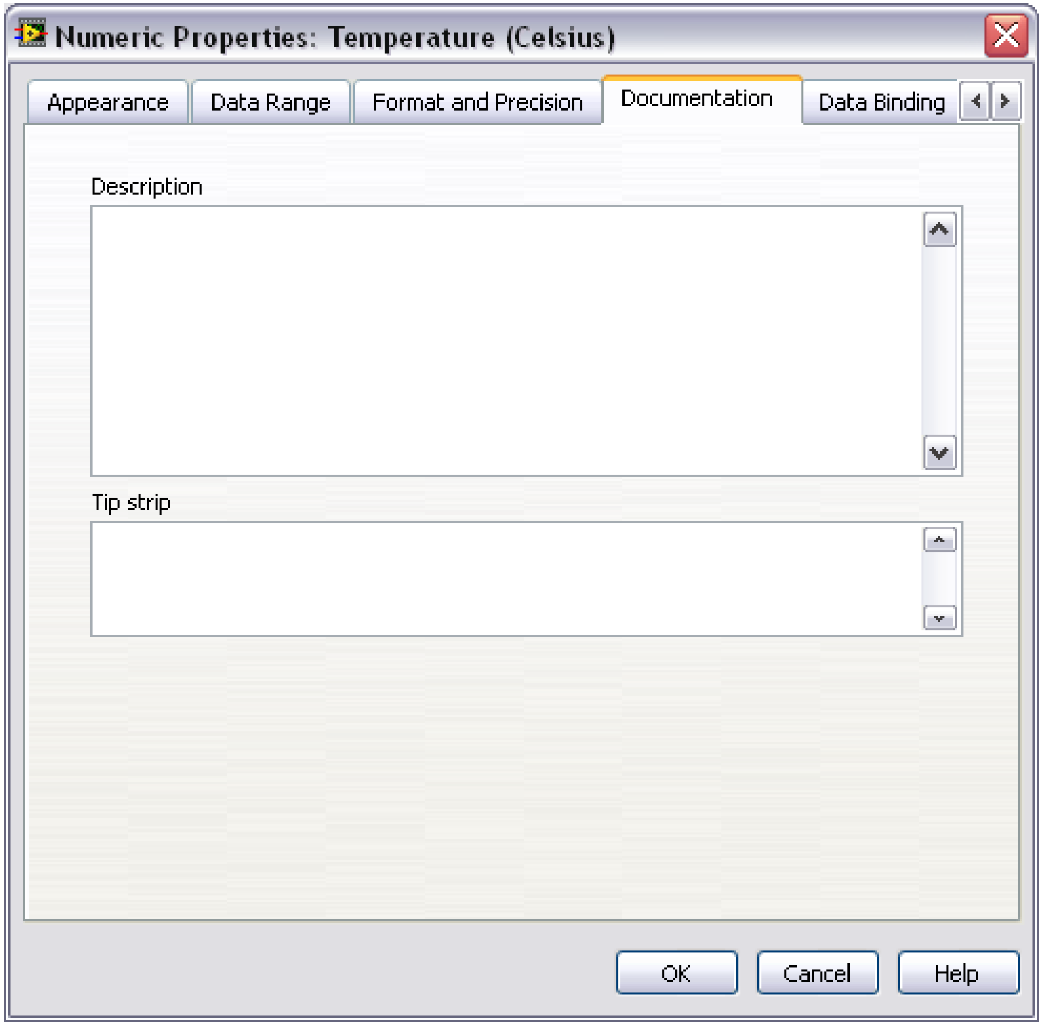
- Label Names
- Tip Strips
- Descriptions
- Free Labels

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.

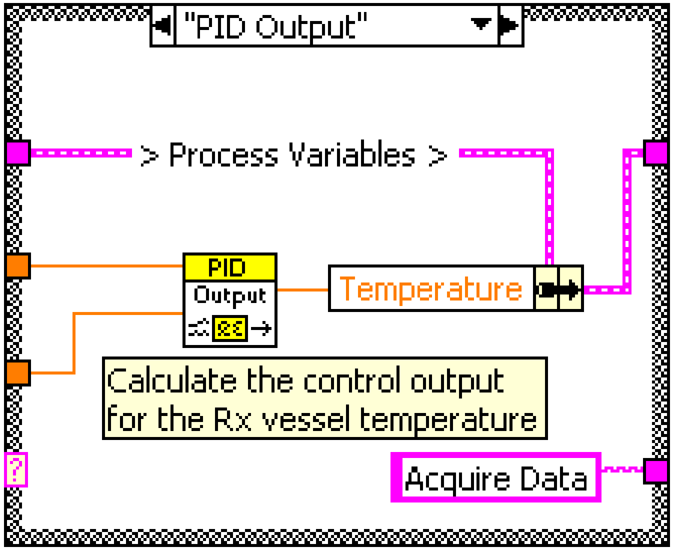


Numeric Properties: Temperature (Celsius)

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.

Homework:
Creating and Using SubVIs

Create a subVI.

- Do not forget to wire the connector pane and create an icon.
- Use error Case Structure to surround the code of the subVI.

Call the subVI in another VI and verify if it works correctly.

Homework:
Documenting Code

Create and document a VI.