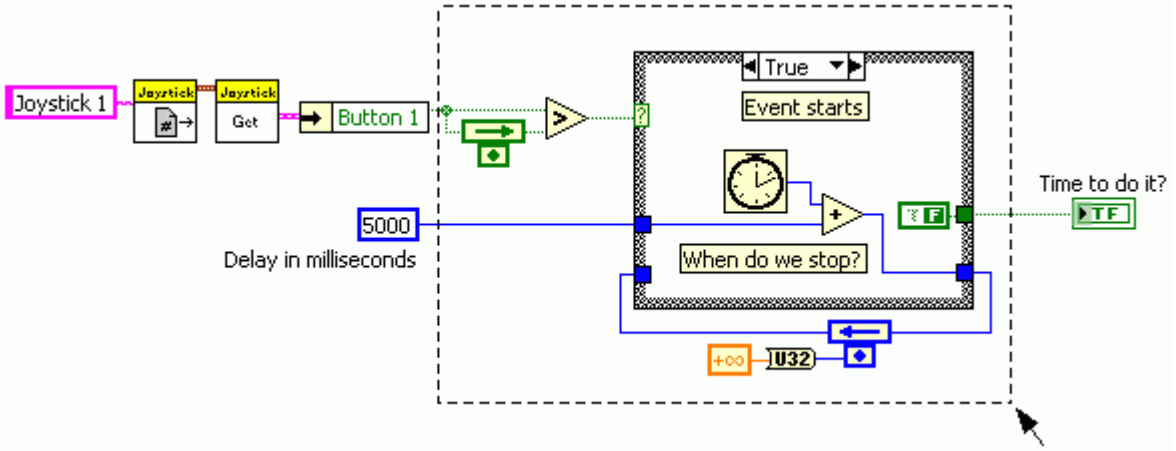
FRC LabVIEW Sub-vi Example

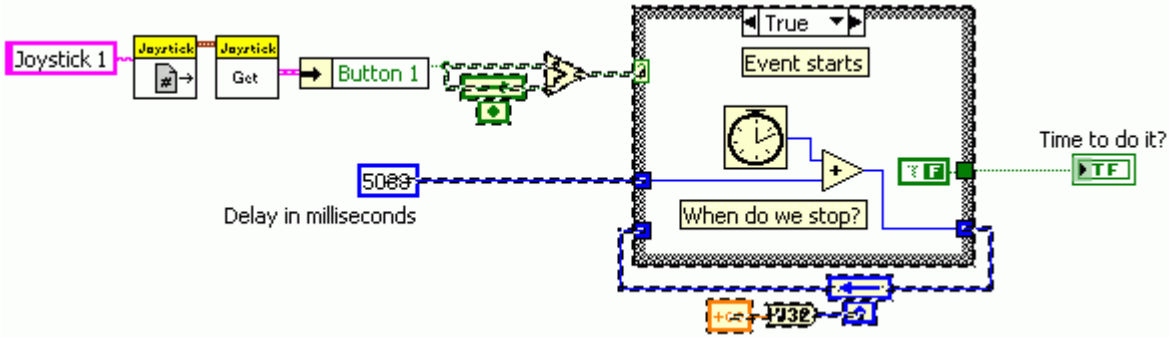
Realizing you have a clever piece of code that would be useful in lots of places, or wanting to un-clutter your program to make it more understandable, you decide to put some of it into a sub-vi…

Example A – The easiest way to create a sub-vi is from existing code

What makes this approach especially easy is that the vi inputs/outputs will automatically be setup for you without requiring you to create each one manually.

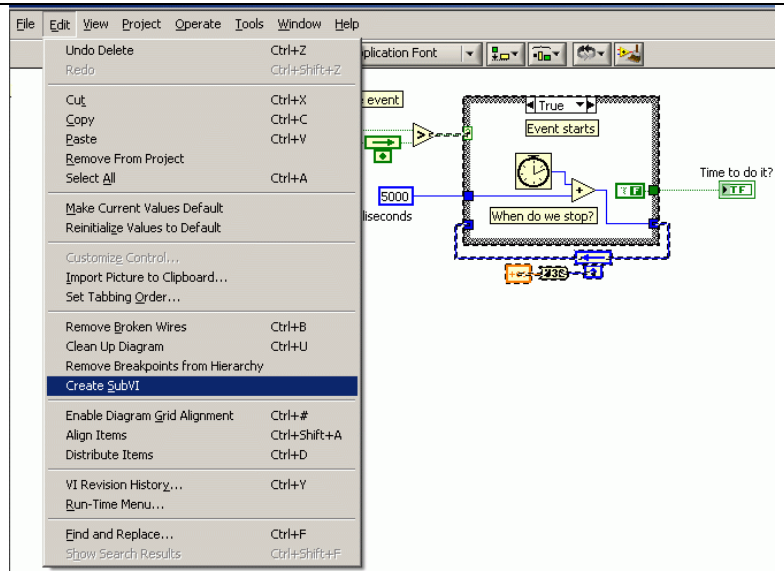
<table>
<tr>
<td> 
</td>
<td>Step 1:

Highlight the code to become the new vi, making sure the inputs and outputs you'll want remain outside the highlighted area. Drag things further out of the way if necessary.

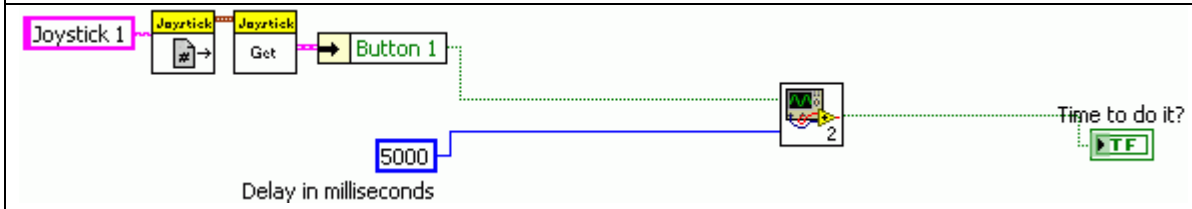
Tip:
If the code is crowded and difficult to isolate, then use Edit -> New VI to start with a blank slate, copy and paste a large chunk of your code and make room around the part to be turned into a subvi without risking your original code.</td>
</tr>
<tr>
<td> 
</td>
<td>Like so

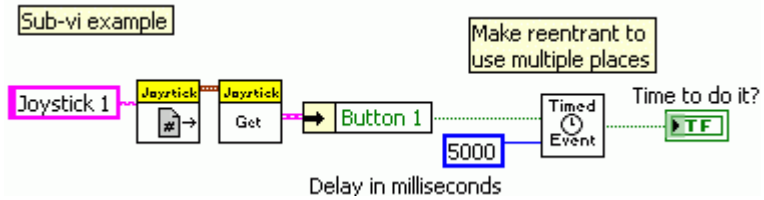
Inputs will be the un-highlighted:
• Start of event (Joystick 1, button 1)
• Time delay (ms)

Outputs will be the un-highlighted:
• Event trigger (t/f)</td>
</tr>
</table>

<table>
<tr>
<td> 
</td>
<td>Step 2:

Edit ->Create SubVI

to generate your new vi while replacing the code you've highlighted with the new vi icon.</td>
</tr>
<tr>
<td> 
</td>
<td>Your original code will automatically be replaced by the new vi with a generic numbered icon. This is what it will look like.

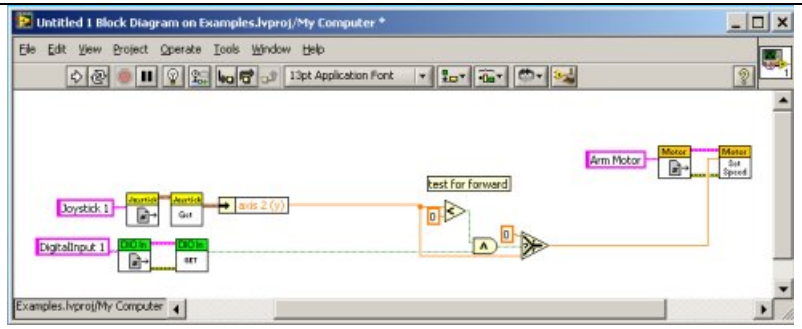
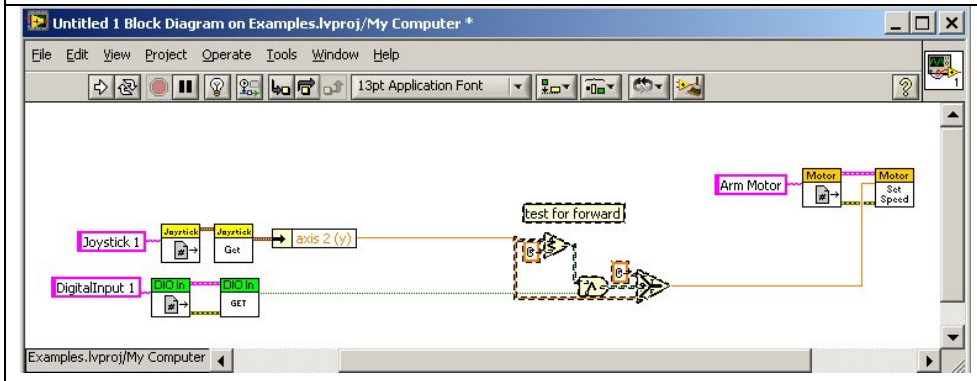
Double-click on the new icon to see your new sub-vi, make changes, redesign the icon picture, add inputs/outputs, etc. You must save (and name) the new vi before it can be used.</td>
</tr>
<tr>
<td> 
</td>
<td>Here's what it might look like after it's been dressed up with a personalized icon.

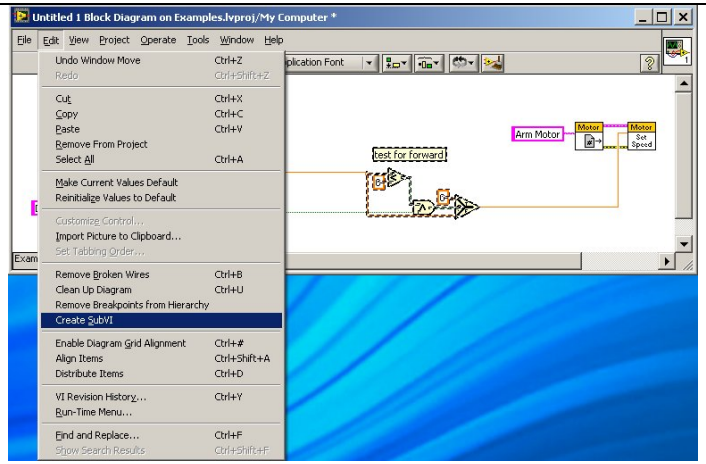
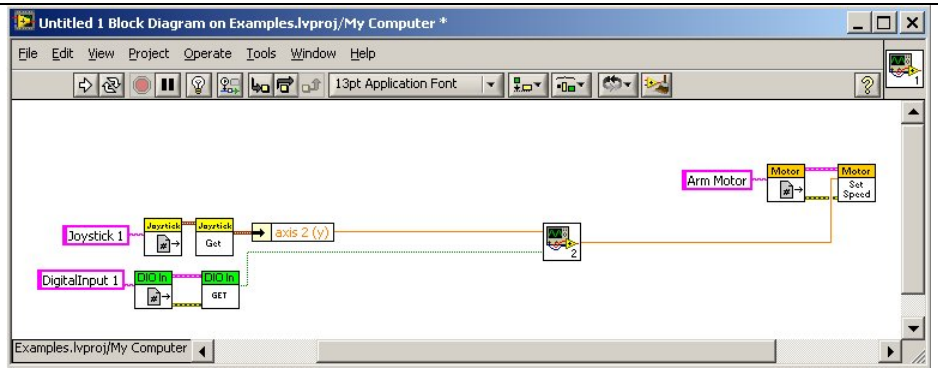
The next section will go into details about how to change the icon, add an input/output.</td>
</tr>
</table>

Example B – A similar example that goes into some common extras

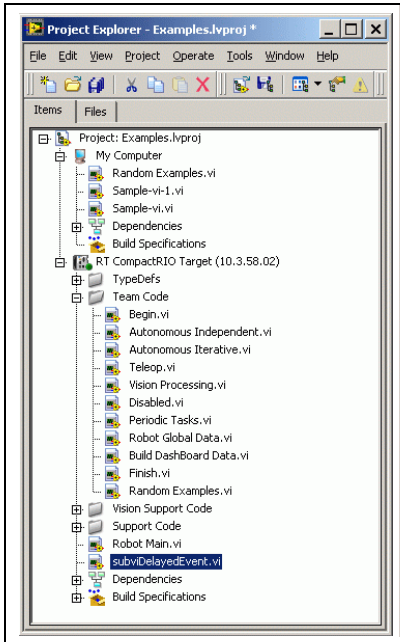
- Create subvi
- Save it
- Change the icon
- Add a new additional input
- Make reentrant

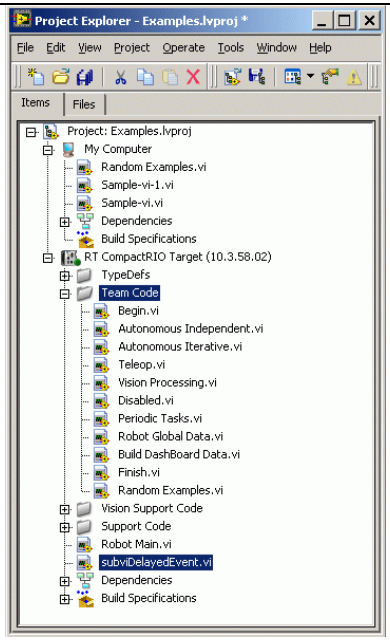
Create a subvi

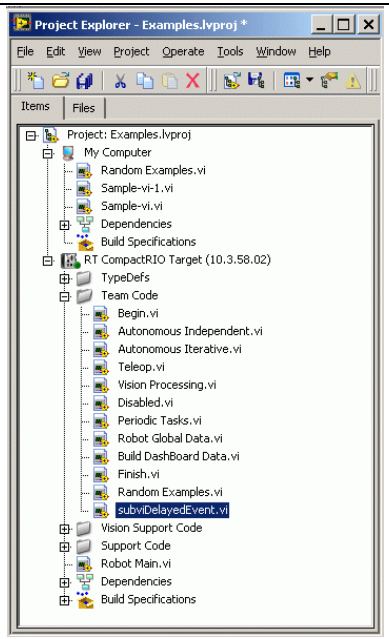
	Create a new vi from scratch, using *Edit -> New VI,* in a new blank window or create one from highlighted existing code. In existing code isolate the segment you'll make into a new vi.
	Highlight the new vi code

	Select *File -> Create SubVI*
	Your new vi appears in place of the code you highlighted. It's given a default generic numbered icon. It is ready to go, and will do the same job as the original code did, but does need to be saved before you can run it. If you forget to save LabVIEW will remind you when you build, deploy, or close the project.

Saving the new subvi:



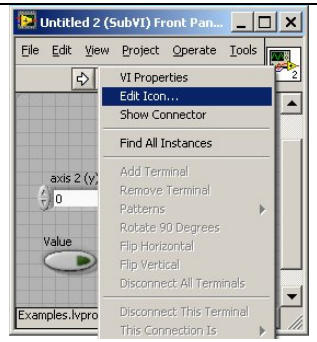




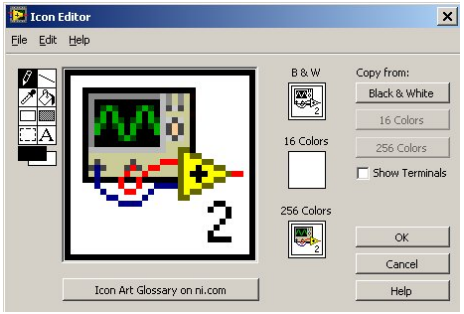
The new vi must be given a name and saved as part of your project.

1. Double click on the new icon to open the new subvi
2. Click *File -> Save As* to give it a name. Be sure to save it into your LabVIEW project folder.
3. After saving it the new vi will appear in your Project Explorer list.
4. It's recommended that you drag and drop the new vi into the Team Code sub-folder to keep all your modified code in one place

Change the Icon:



1. Open the new vi by double-clicking in the default icon it was given, or by double-clicking on the filename in Project Explorer.
2. Right-click on the picture of the icon in the upper right of the front panel window.
3. Choose *Edit Icon…* to begin

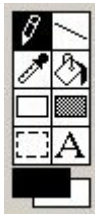


You'll get a popup editing window with the icon
A paint-style tools menu on the left gives you some basic drawing capabilities.

You can also copy and paste artwork from other places to add to your icon.

- There's a link to an NI webpage of some common artwork you can use.
- Or use *Edit->Import Picture to Clipboard…* & paste
- Or drag & drop any graphic/photo file to the upper right corner of the front panel or block diagram and it will be converted to a 32 × 32 pixel icon. Make sure the graphic is square, otherwise it'll end up squeezed or stretched looking.

Icons are 32x32 pixels



Pencil—Draws and erases pixel by pixel. To draw horizontal or vertical lines, press the <Shift> key while you use this tool to drag the cursor.

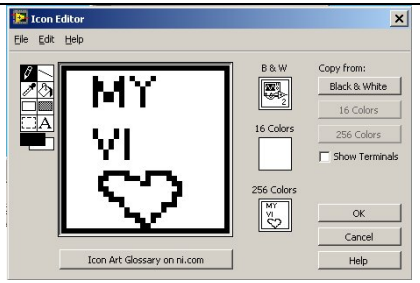
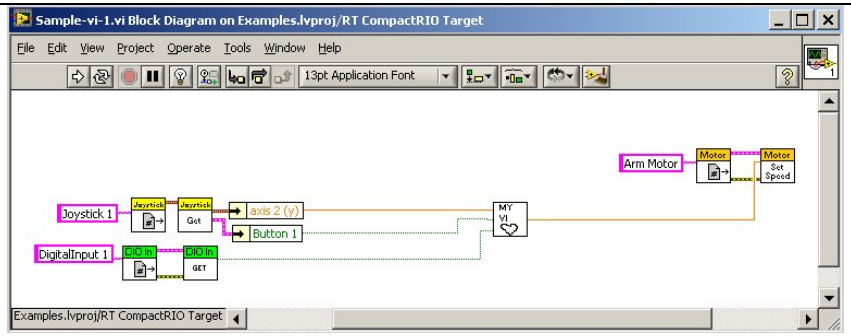
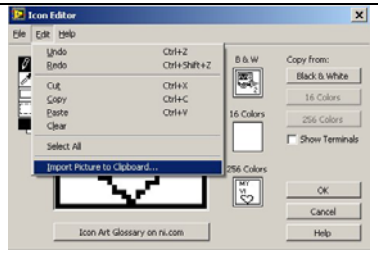
Line—Draws straight lines. To draw horizontal, vertical, and diagonal lines, press the <Shift> key while you use this tool to drag the cursor.

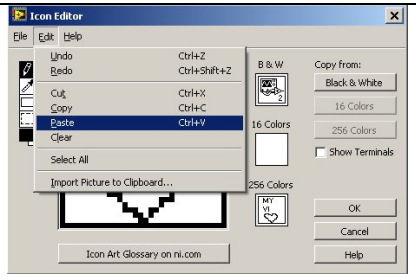
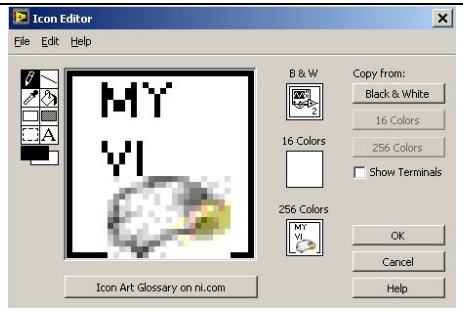
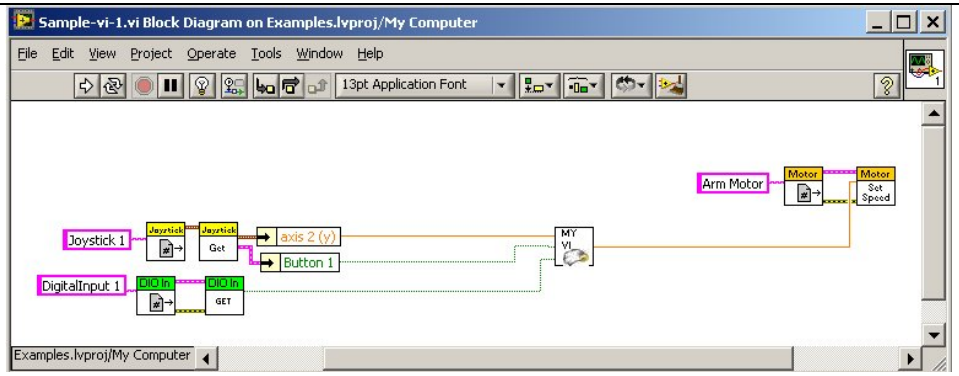
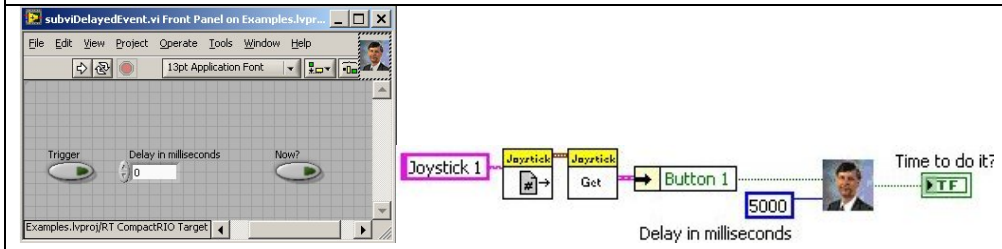
Color Copy—Copies the foreground color from an element in the icon.

Fill—Fills an outlined area with the foreground color.

Rectangle—Draws a rectangular border in the foreground color. Double-click this tool to frame the icon in the foreground color.

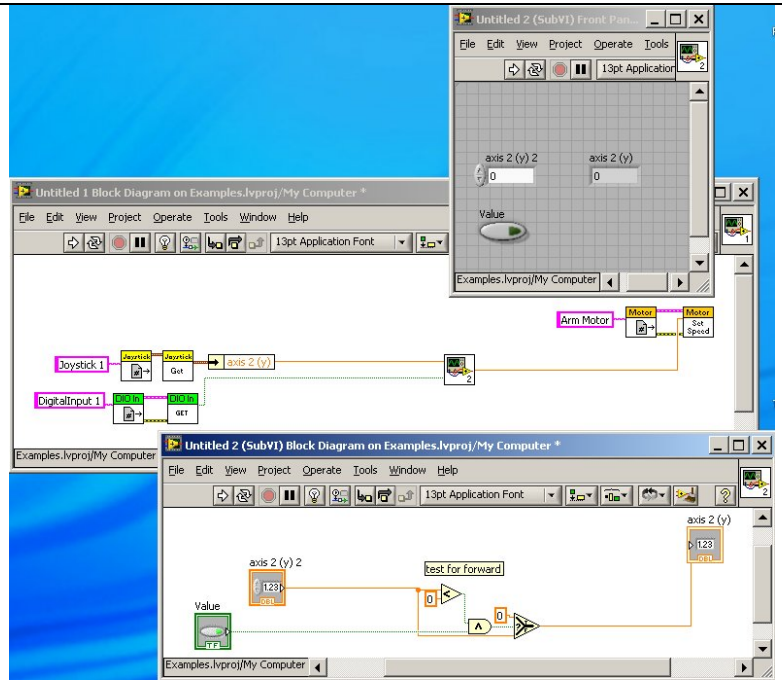
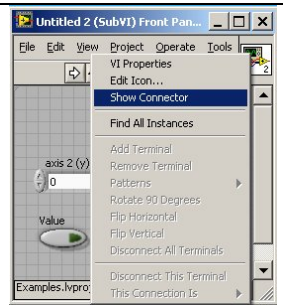
Filled Rectangle—Draws a rectangle with a foreground color frame and filled with the background color. Double-click this tool to frame the icon in the foreground color and fill it with the background color.

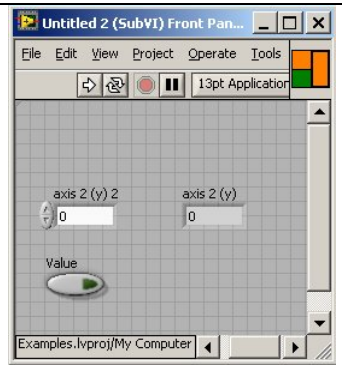
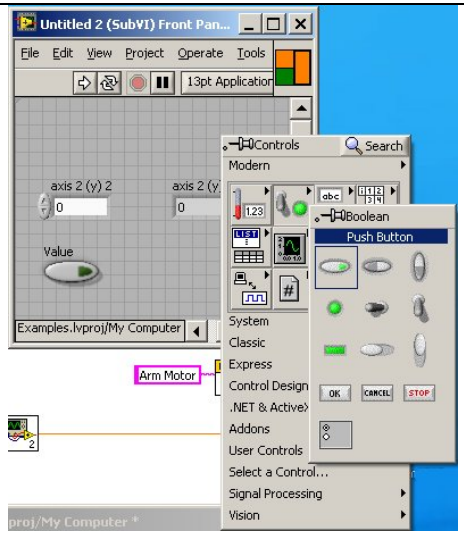
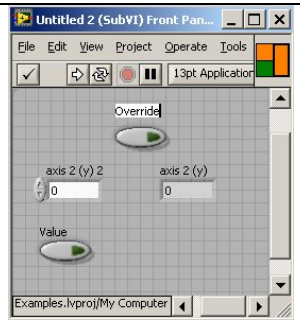
	Select—Selects an area of the icon to cut, copy, move, or make other changes. Double-click this tool and press the <Delete> key to delete the entire icon. **Text**—Enters text into the icon. Double-click this tool to select a different font. While text is active, you can move the text by pressing the arrow keys. **Foreground/Background**—Displays the current foreground and background colors. Click each rectangle to access a color picker.
	Here's a freehand example…
	Click *OK* and you'll see your new icon applied immediately in all your vi's.
	Here's a quick run through that adds pre-made artwork (limited to 32x32 pixels). These can be produced in any graphic editing utility, or found on the web. *Edit -> Import Picture to Clipboard…* and browse to a picture or graphic file of some sort . This just puts the picture in the clipboard. It won't appear anywhere yet.

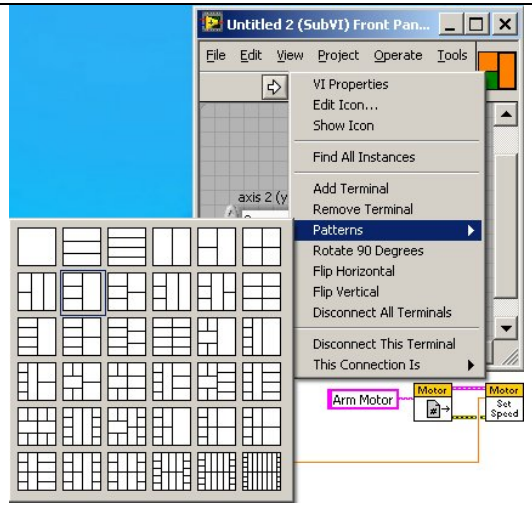
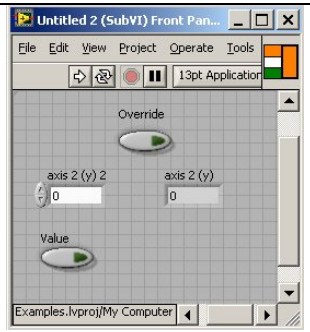
	Use *Edit -> Paste* to add your clipboard picture to the icon
	Use the arrow keys to reposition the new addition on the icon
	This is what it looks like after you click *OK*
	Example of drag & drop of any size photo to the icon in upper right of front panel. It gets automatically reduced to a 32x32 image, soif it's not square to begin with it'll appear foreshortened.

Add an Input:

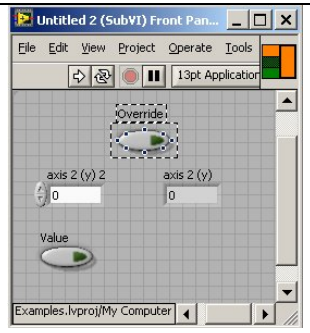
This same instruction holds for creating new outputs as well.

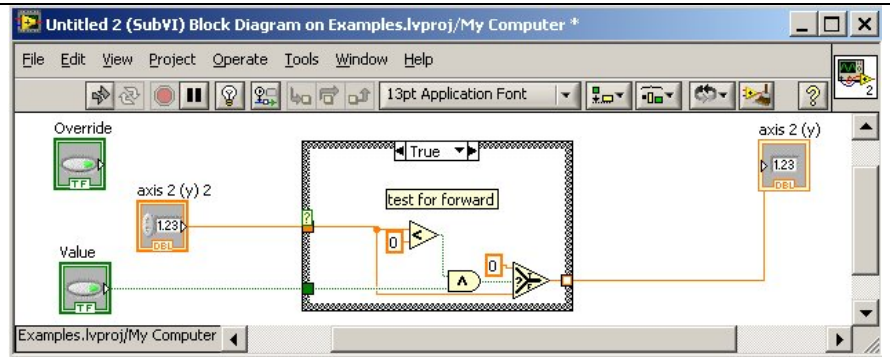
	Start by opening the front panel of your vi. <- This is the example we'll use The subvi restricts motor movement in one direction based on a limit switch input. Inputs are: • Joystick axis to drive a motor • Limit switch to stop the motor if it goes too far in just one direction Outputs are: • The motor speed
	1. First we need to add a place to connect the new input. Right-click on the icon in the upper right of the front panel and choose *Show Connector*

	The icon will change to a sub-divided box with smaller colored boxes for any pre-existing inputs and outputs. In this case there are two input boxes on the left colored orange (the joystick axis is a floating point number) and green (the digital input is a Boolean), and one larger orange box (motor speed is a floating point number) for our single output on the right. We'll need to add a new place to connect our new input.
	2. Add the new input you want. Here we're going to add a new Boolean that can be used to override the limit this subvi is enforcing. • Right-click on the front panel to get the controls palette, go to Boolean controls and choose one style. • Place the new control on the front panel
	This is what it now looks like with the new input control called "Override"

<table>
<tr>
<td> 
</td>
<td>3. Change the connector pattern to make room for the new input.
• Right-click on the connector in the upper right of the front panel
• Choose Patterns to see lots of variations for how I/O can connect to your subvi.
• Choose any one of the patterns that appeal to you. You just need to choose one that has as many or more open boxes than you have inputs and outputs.</td>
</tr>
<tr>
<td> 
</td>
<td>We picked a pattern that had just enough open boxes. Three on the left for our three inputs and only one on the right for our single output.

LabVIEW will keep the I/O connectors already defined, but depending on how radical you new pattern is, they may get put in different spots than you'd like them to be. You can reorder any of the existing I/O connections just as we'll show how to add your new input.

A white box means nothing is connected there.</td>
</tr>
<tr>
<td> 
</td>
<td>4. Make the connection
• Click on the white connector, then
• Click on the input that should be attached to that connector.
The white connector will take on the color of the input data type.</td>
</tr>
</table>

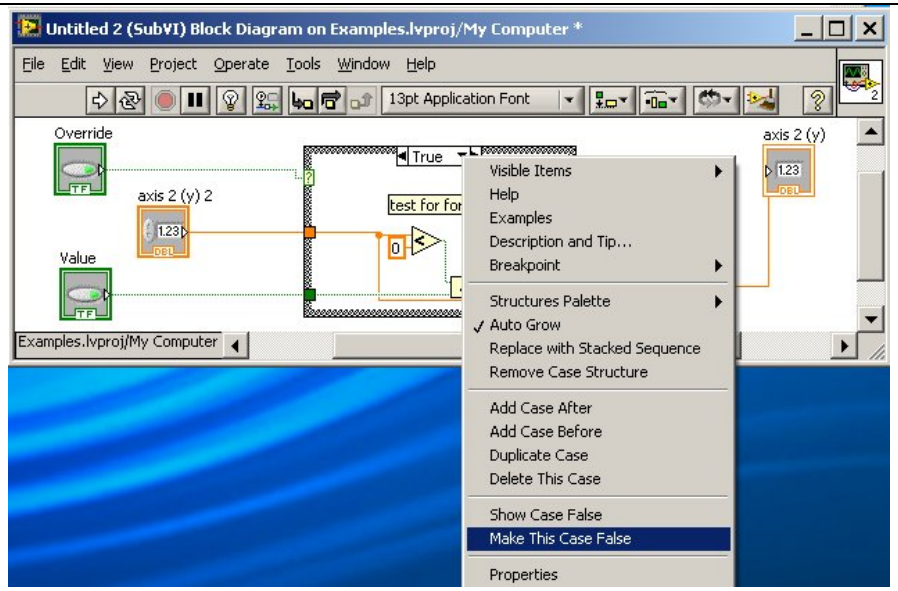


Open the subvi's block diagram and you'll see the new input you added just floating off by itself, not doing anything.

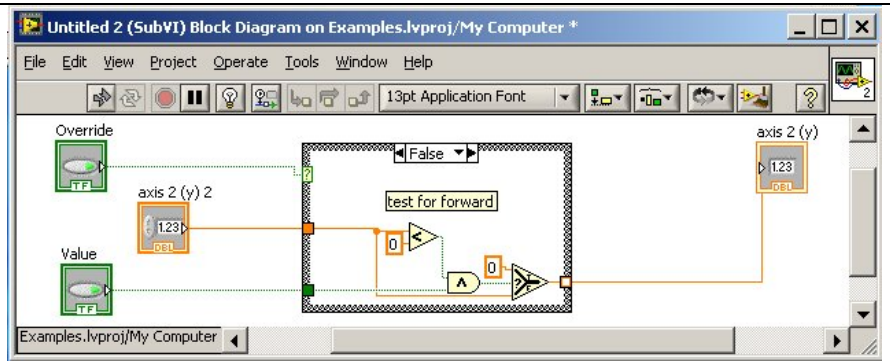
You need to wire that into place.

For this example this new override input will control a case statement that applies the limit switch or overrides it altogether.

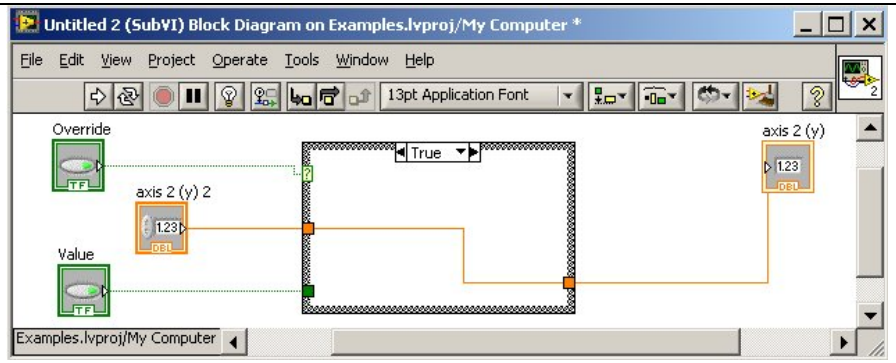
- Add a case around the code



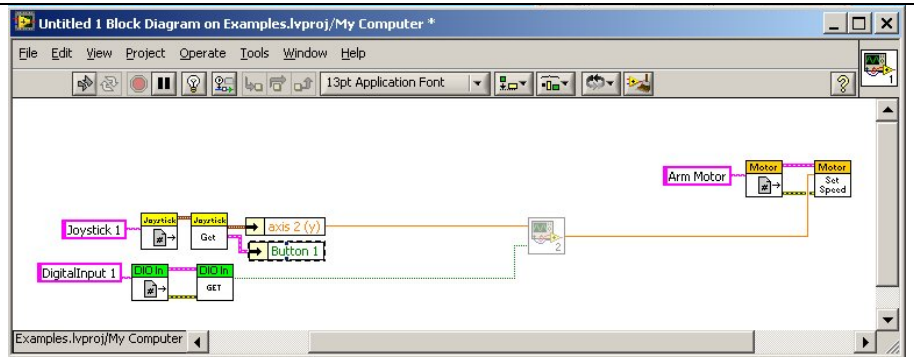
- Wire the new Override to control the case
- Right-click on the edge of the case structure and choose *Make This Case* False to switch the True/False cases so that the code is applied in the False case and bypassed in the True case.



This is what it should look like now.



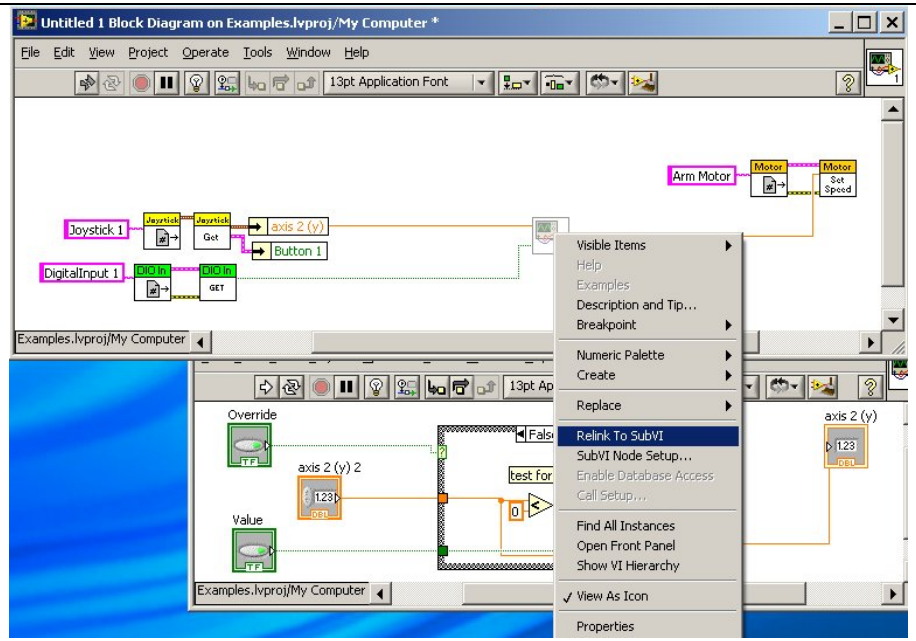
- Flip to the True case and just wire the input axis straight through to the output, so if Override is True then the input is left unaltered.



5. Begin adding your new input wherever your subvi is used.

In this example we'll add a joystick button to control our new override input.

- Copy the axis *unbundle by name*
- Wire the new unbundle to the Joystick Get button output

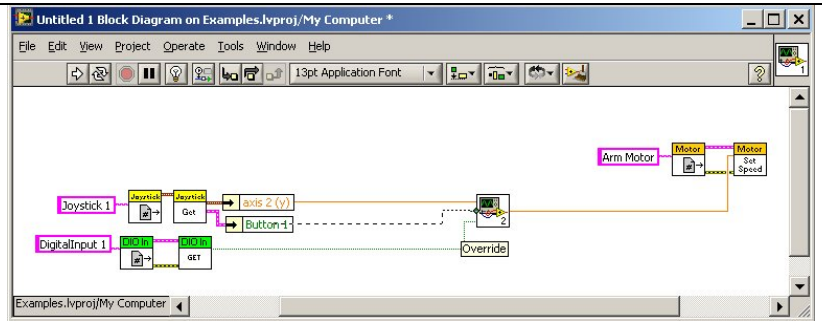


6. Relink the subvi wherever it's used.

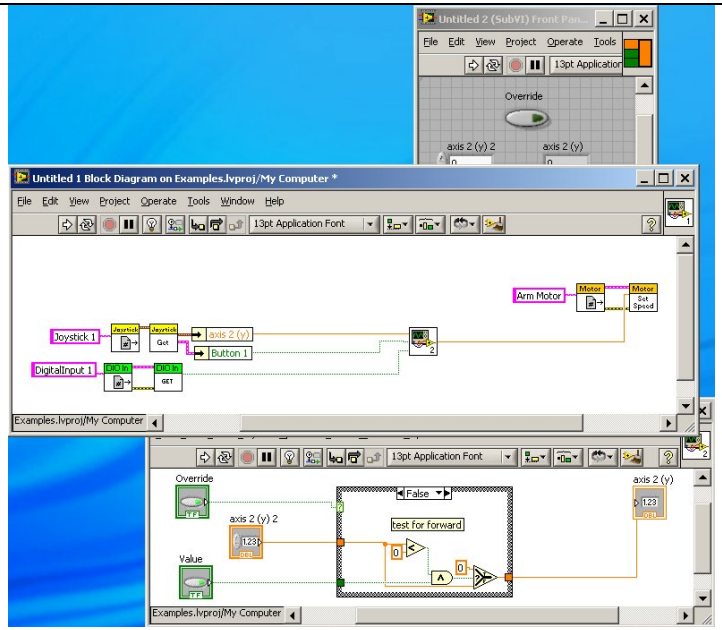
- Whenever you change the input/output connector LabVIEW wants you to inspect each use of the subvi to be sure inputs/outputs are still wired correctly. So a vi with a changed connector pattern will be grayed out wherever it's used in a block diagram.

- Right-click on the grayed out icon and choose *Relink To SubVI*

The subvi icon will no longer be grayed out.

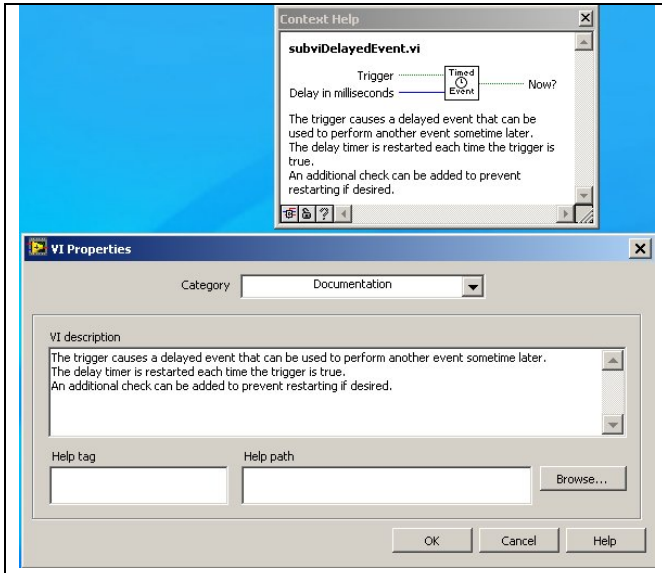


7. Attach the new override button to the new subvi input.



All done

Remember to Document it



Edit -> VI Properties

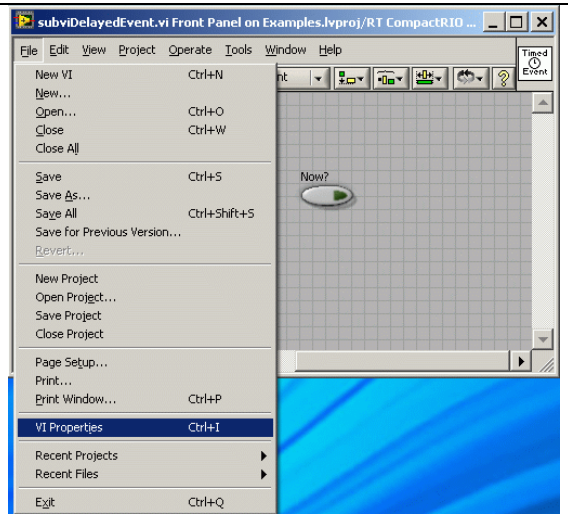
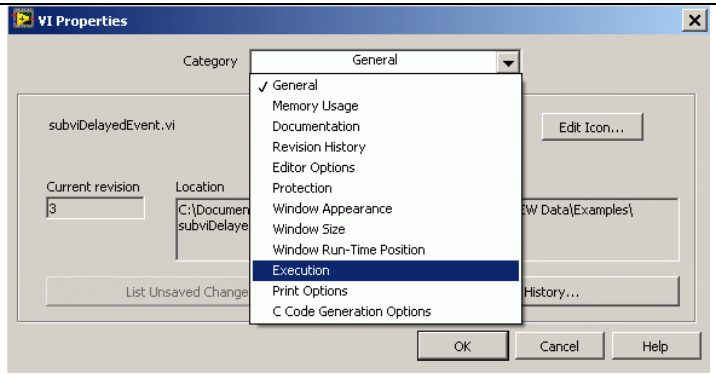
Go to Documentation on the Category pull-down and enter description and useful information.

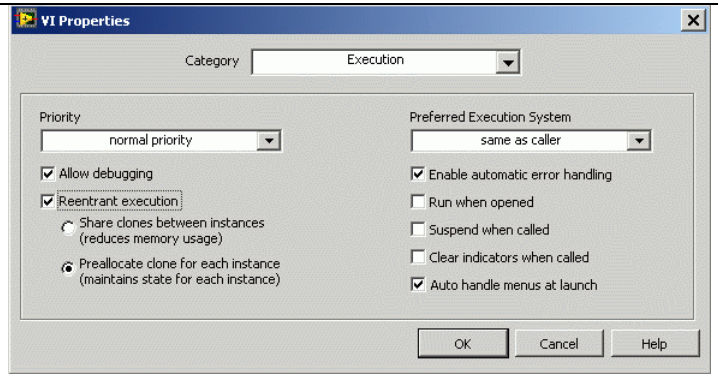
The inputs, outputs and icon will automatically be added at the top of the Context Help for your new vi with your descriptive text underneath.

Make reentrant for multiple uses:

This is only necessary if the subvi is preserving specific data about a particular call, such as a start or end time via feedback nodes AND you want to use it in more than one place. Normally a new subvi is not reentrant and this means if it's called from multiple places they all have to line up and wait their turn to use the subvi, and you should not try expect data to be untouched from call-to-call.

By default there is just one copy of a new subvi running in your program. This is normally how you want to run as long as the subvi is doing something straight forward like doing a calculation or measurement and doesn't have to remember or distinguish between calls. Making a subvi reentrant causes each call in your code to be it's own unique copy, so that it will preserve data that has to be remembered from call to call and not get mixed up with when it's called in other places which also have to remember states, time, counts, or whether buttons have already been pushed.

	Start by opening the vi properties *Edit -> VI Properties*
	In the pop-up properties window use the pull-down to go the *Execution.*



Click *Reentrant execution*

Choose *Preallocate clone for each instance*

This gives each instance it's own memory to preserve each unique state.