Tutorial: For Loops and While Loops

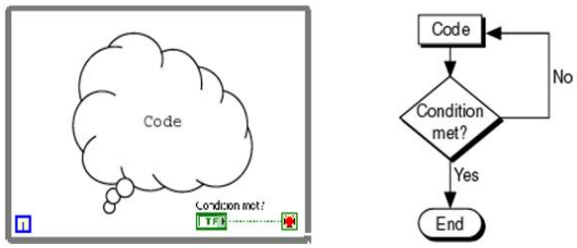
Publish Date: Jul 02, 2008 | 39 Ratings | **3.36** out of 5

Overview

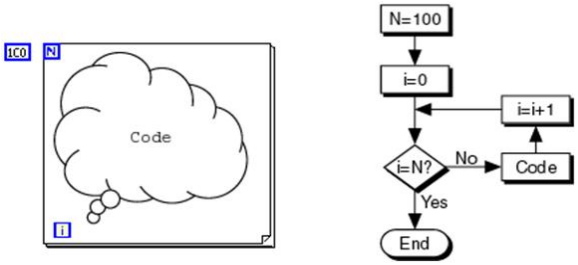
This tutorial explores some of the basic functions and uses of For Loops and While Loops. You will learn how to create For Loops and While Loops and when the appropriate time would be to use them in your program.

Table of Contents

A while loop is a control flow statement you use to execute a block of the subdiagram code repeatedly until a given Boolean condition is met. First, you execute the code within the subdiagram, and then the conditional terminal is evaluated. Unlike a for loop, a while loop does not have a set iteration count; thus, a while loop executes indefinitely if the condition never occurs.



A for loop is a control flow statement you use to execute a block of the subdiagram code a set number of times, but a while loop stops executing the subdiagram only if the value at the conditional terminal exists.

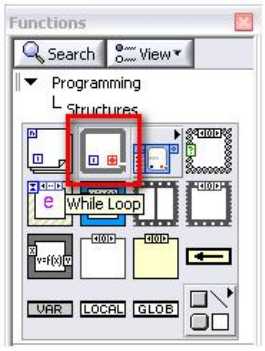


Building a While Loop

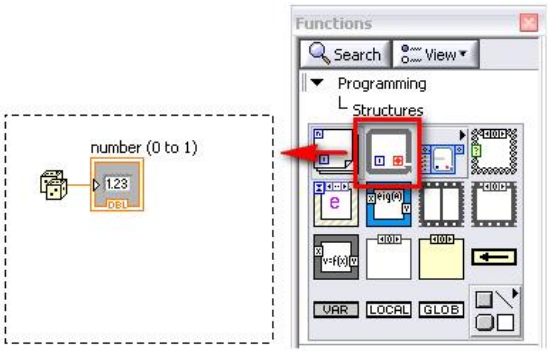
1. Open a new VI. You can open a blank VI by selecting File»New VI

2. If the Functions palette is not visible, right-click any blank space on the block diagram to display a temporary version of the palette. Click the thumbtack in the upper left corner of the Functions palette to pin the palette so it is no longer temporary.

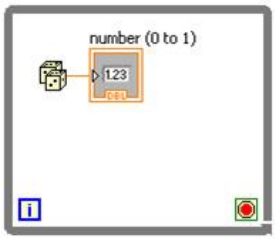
3. Select the while loop from the Structures palette under the Functions palette.



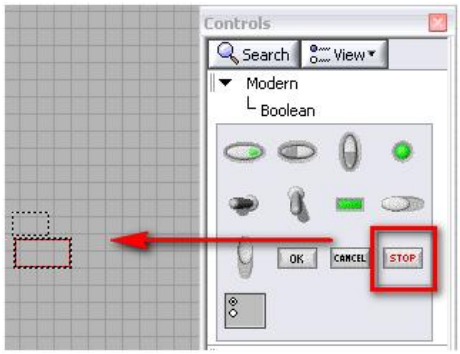
4. Use the cursor to drag a selection rectangle around the section of the block diagram you want to repeat.



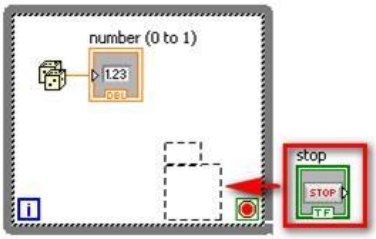
5. When you release the mouse button, a while loop boundary encloses the section you have selected.



6. Place a Stop button on the front panel. You can find this under Controls Palette»Boolean»Stop.



7. Add the Stop button from the block diagram to the while loop by dragging and dropping it inside the while loop.

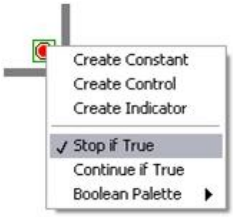


8. The conditional terminal, shown below, defines when the loop stops. There are two settings for the conditional terminal: Continue if True and Stop if True. When set to Continue if True, the while loop runs only if a Boolean value of true is sent to the terminal. If the conditional terminal is set to Stop if True, and a Boolean value of true is sent to the conditional terminal, the loop halts execution.

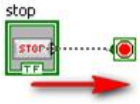
Stop if True

Continue if True

9. To switch the conditional terminal between Continue if True and Stop if True, right-click on the conditional terminal and check the corresponding setting.

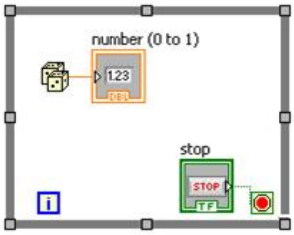


10. Wire the Stop button to the conditional terminal so that you can control the execution of the while loop. When the Stop button is pressed, a true value is passed to the conditional terminal causing the while loop to stop execution. You can wire any Boolean data to the conditional terminal to control the execution of a while loop.



11. The iteration terminal is an output terminal that contains the number of completed iterations. The iteration count always starts at zero. During the first iteration, the iteration terminal returns 0.

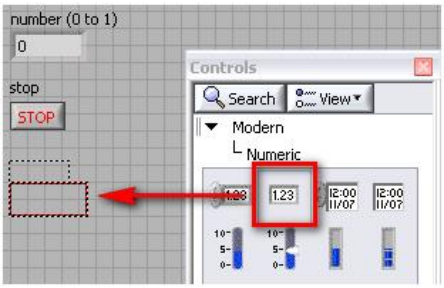
12. You have just created a simple while loop that generates random numbers and displays them until the Stop button is pressed.



Structure Tunnels

Use structure tunnels to feed data into and out of structures like the while loop. If you want to send data into your while loop, you need to create structure tunnels. Data sent into the While Loop is sent only on the first iteration, and data sent out of the while loop is sent only after the last iteration. Now create a structure tunnel to display the number of iterations executed.

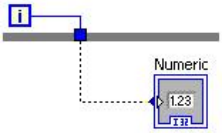
1. Create a numeric indicator on the front panel.



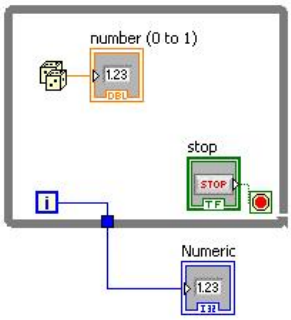
2. Drag a data wire from the iteration counter into the border of the while loop to create a structure tunnel.

3. The tunnel appears as a solid block on the border of the while loop. The block is the color of the data type wired to the tunnel. Data pass out of a loop after the loop terminates. When a tunnel passes data into a loop, the loop executes only after data arrive at the tunnel.

4. Drag a data wire from the structure tunnel into the numeric indicator to pass the number of iterations to your display.



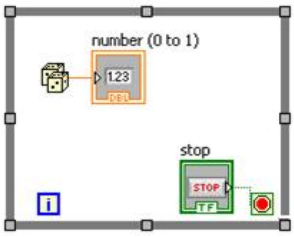
5. In the following block diagram, the iteration terminal is connected to a tunnel. The value in the tunnel does not get passed to the iteration indicator until the while loop finishes executing.



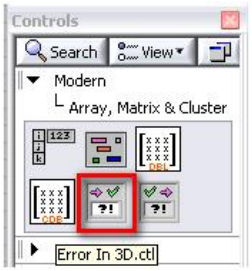
Manipulating the Conditional Terminal

By using Boolean functions, you can implement multiple conditions to affect your while loop conditional terminal. You can use an “or” function to compare an error wire status and a Stop button control so that if either is TRUE, the conditional terminal receives a TRUE signal, and the while loop stops.

1. Create a while loop as outlined previously.



2. Place an Error In 3D.ctl control on the front panel. You can find this under Controls Palette»Array, Matrix & Cluster»Error In 3D.ctl.

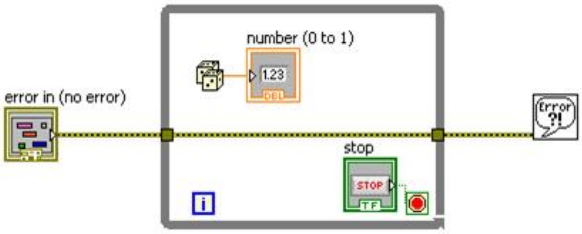


3. Remove the connection wire between the Stop button and the conditional terminal.

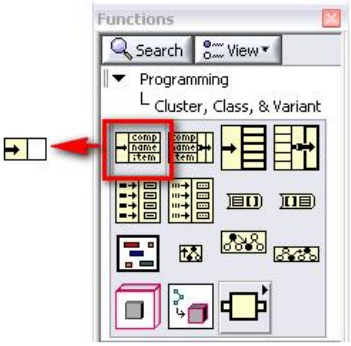
4. Make sure the Error In 3D.ctl is outside of the while loop.

5. Place a Simple Error Handler outside of the while loop. You can find this under Functions Palette»Dialog & User Interface»Simple Error Handler.vi. This function reports any errors encountered when called.

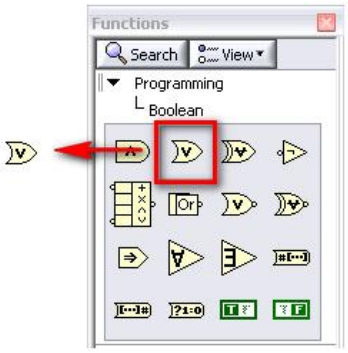
6. Link the error in control with the Simple Error Handler through the while loop via structure tunnels. Drag a wire from the output of the error in control to the border of the while loop, and then drag the error wire from the tunnel created to the opposite border of the while loop and into the input of the Simple Error Handler.



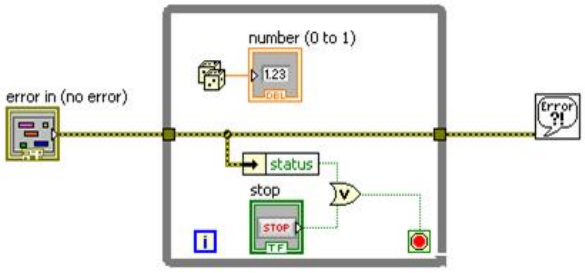
7. Place an Unbundle By Name function inside of the while loop. You can find this under Functions Palette»Programming»Cluster, Class, & Variant. This function separates the Boolean portion of the error data wire.



8. Place an “or” function inside of the while loop. You can find this under Functions Palette»Programming»Boolean.



9. Link the data wires as shown below. This simple VI stops execution of the while loop if either the Stop button is pressed or an error is detected inside of the while loop.

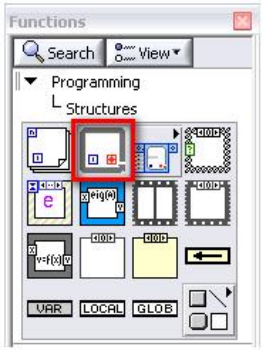


Building a For Loop

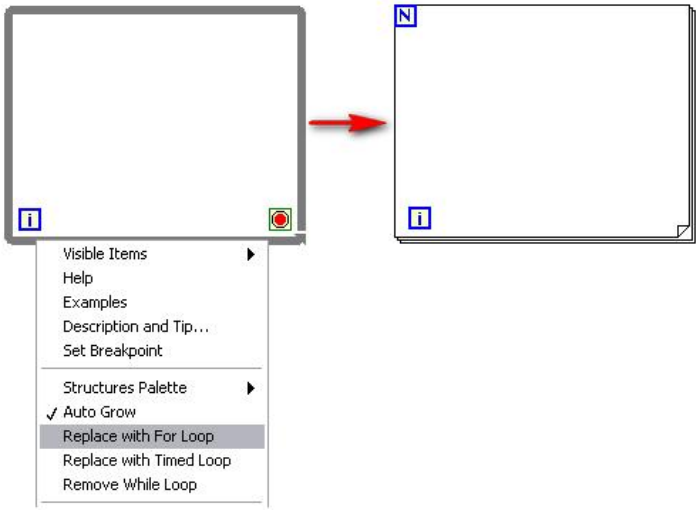
1. Open a new VI. You can open a blank VI by selecting File»New VI.

2. If the Functions palette is not visible, right-click any blank space on the block diagram to display a temporary version of the Functions palette. Click the thumbtack in the upper left corner of the Functions palette to pin the palette so it is no longer temporary.

3. Select the for loop from the Structures palette under the Functions palette.

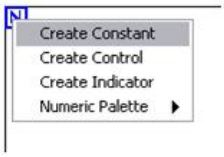


4. You also can place a while loop on the block diagram. Right-click the border of the while loop and select Replace with For Loop from the shortcut menu to change a while loop to a for loop.

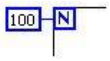


5. A for loop contains a count terminal. The count terminal dictates how many times the subdiagram is executed.

6. Right-click on the count terminal and Create Constant to link the count terminal with a numeric value.



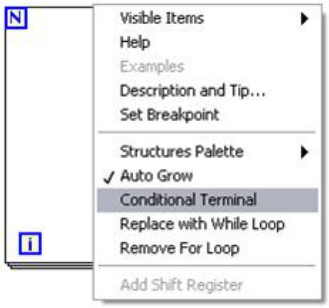
7. By inserting 100 into the numeric constant, the for loop executes 100 times before stopping.



8. The iteration terminal, shown as follows, is an output terminal that contains the number of completed iterations. The example below would update an indicator on the front panel with the current iteration number.

Adding a Conditional Terminal to a For Loop

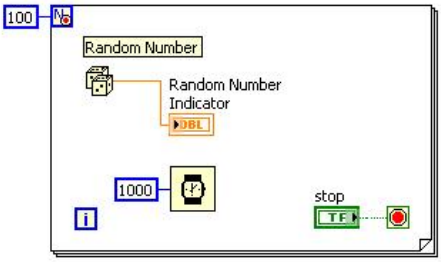
If necessary, you can add a conditional terminal to configure a for loop to stop when a Boolean condition or an error occurs.
1. Right-click on the edge of the for loop and select Conditional Terminal.



2. A for loop with a conditional terminal executes until the condition occurs or until all iterations are complete, whichever happens first.

3. For loops you configure for a conditional exit have a red glyph in the count terminal as well as a conditional terminal in the lower right corner.

4. After you configure the for loop to exit conditionally, the loop appears similar to the figure below. The following for loop generates a random number every second until 100 seconds has passed or the user clicks the Stop button.



For more information on the uses of for loops and while loops, refer to the Context Help. You can access Context Help from Help»Show Context Help or using the shortcut <Ctrl-H>.
