Application Note

PowerStar 5/6 - LabView VI Integration

INTRODUCTION

This application note describes how to integrate LabView VI's into PowerStar 5 and 6. Parameters may be passed from PowerStar to a LabView VI as either a constant or as an expression, LabView indicators may be returned to PowerStar and logged as a result or exported to a variable.

GENERAL DESCRIPTION

In this application note we will integrate LabView as a Dynamic Link Library into a test program and into a Virtual Application.

It must be noted that LabView Shared Libraries and Applications are NOT standalone applications; they require the appropriate LabView runtime engine to be installed. The runtime engine is as per the version of LabView you are using, i.e. if building applications using LabView 7.1 then the applications will require the runtime engine 7.1

NOTE:

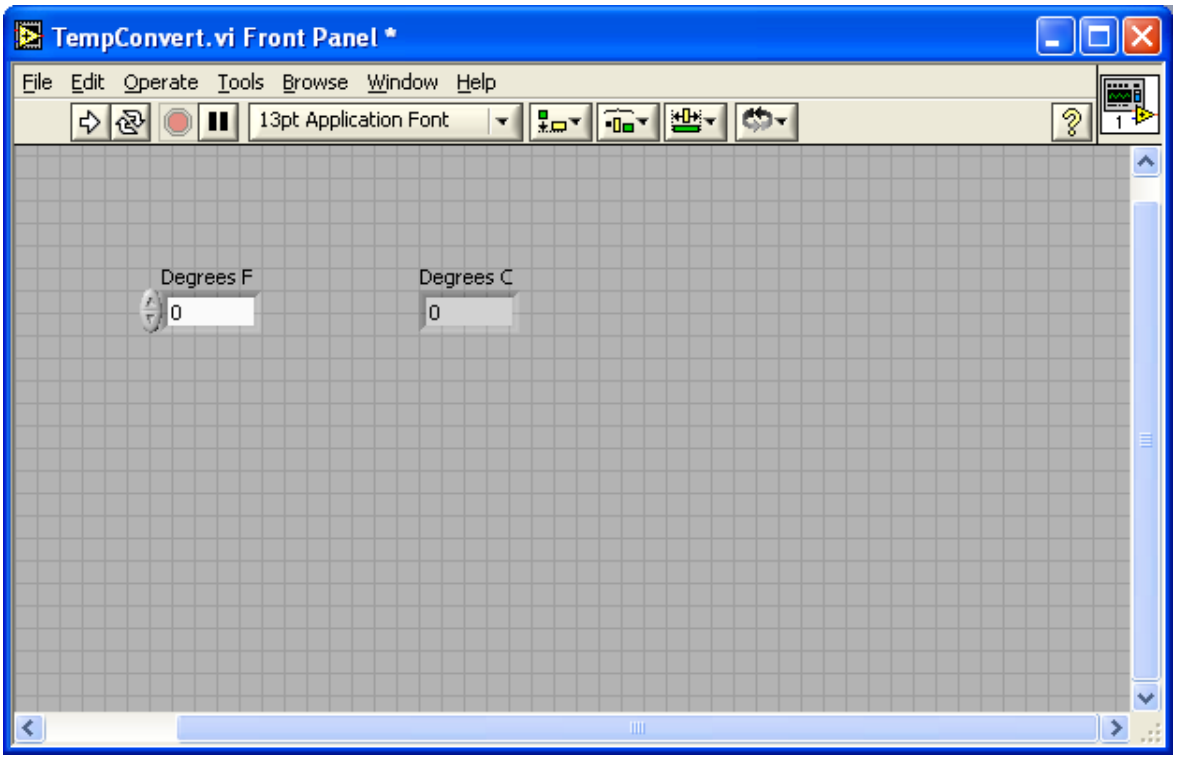
In order to build a LabView VI into a dynamic link library the LabVIEW Professional Development System is required which includes an Application Builder. It is this application builder that is required to build the VI into a dynamic link library (DLL).

Rev B, 7/22/15

CREATING A LABVIEW DYNAMIC LINK LIBRARY

The following example creates a Dynamic Link Library (DLL) that converts the temperature from Celsius to Fahrenheit.

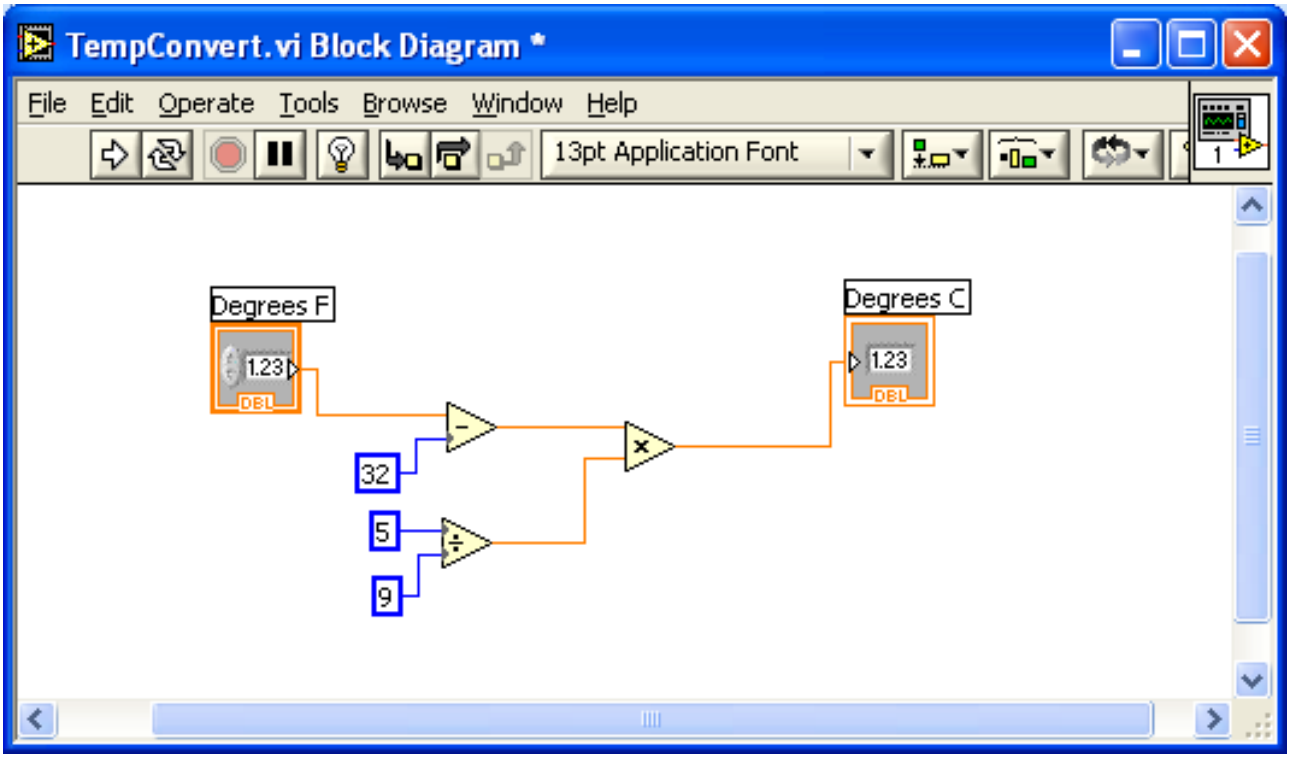
This example will use the formula C=5/9 (F-32)



Step 1 – Create a VI with a Control & Indicator

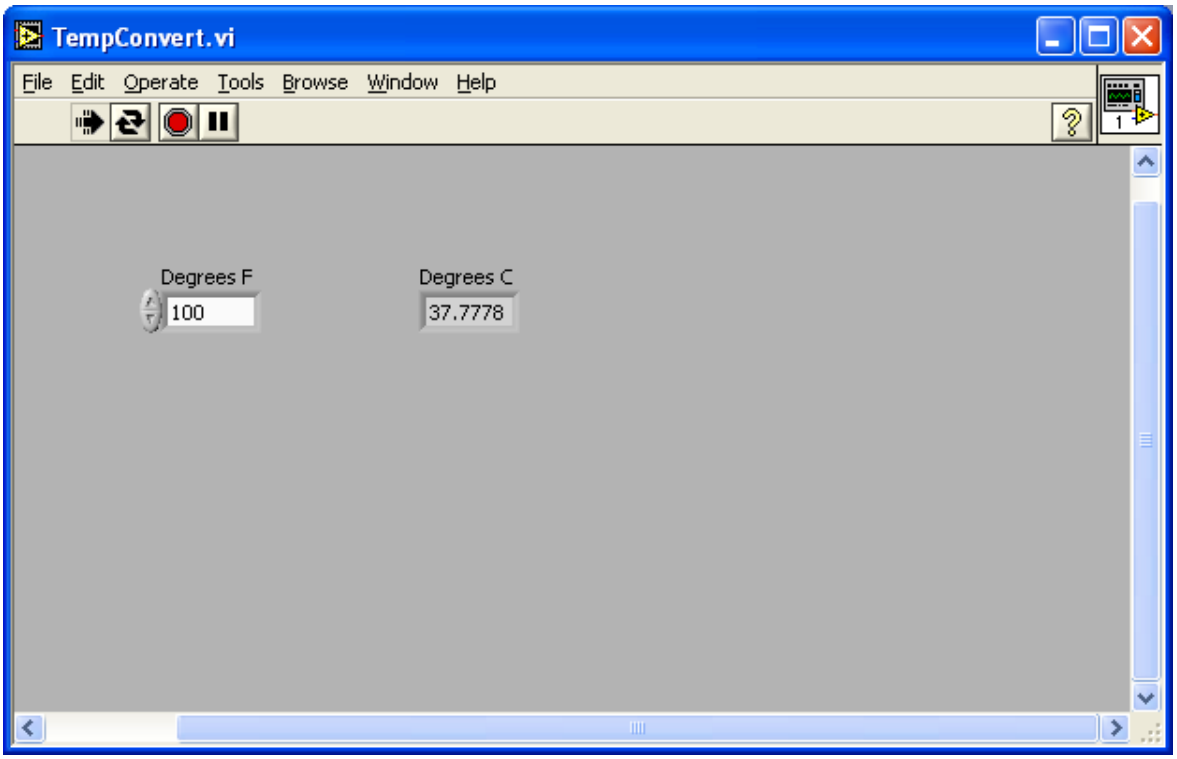
Above Degrees F is a control and Degrees C is an indicator.

The block diagram is as follows.



Step 2 – Create the block diagram.

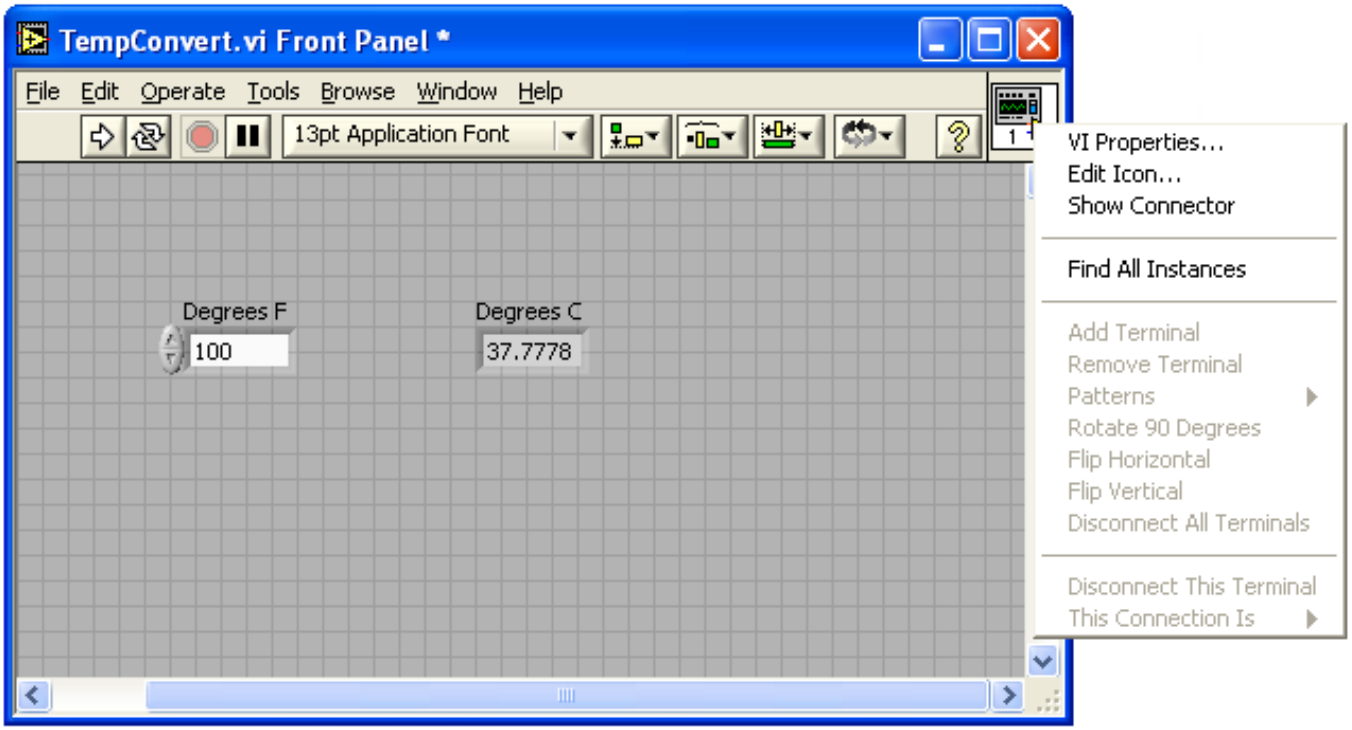
To verify the operation of the temperature converter VI, run the VI and enter various temperatures in Fahrenheit and verify that the correct temperature is displayed in Celsius.



Step 3 – The temperature converter running

The next step is to create VI connectors from the VI, these connectors will enable the selected VI controls & indicators visible outside of the VI.

To add VI connectors right click on the icon at the top right of the front panel and select Show Connectors.



Step 4 – Select Show Connector option

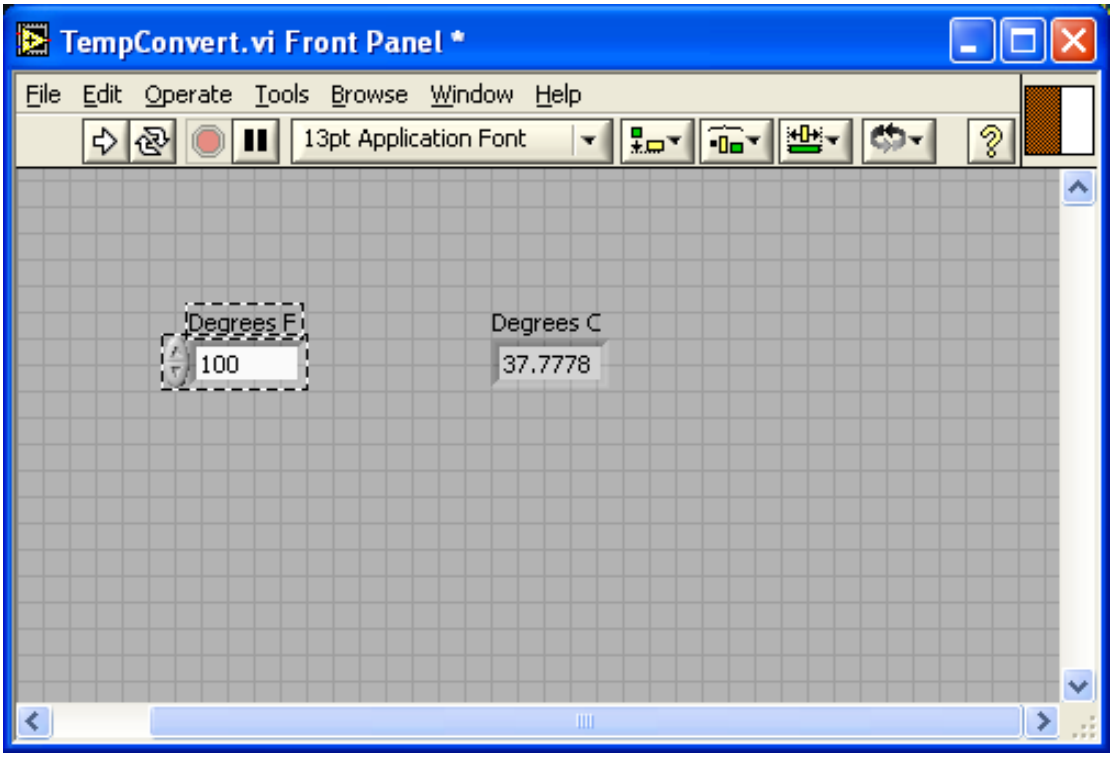
The VI connector icon will by default display a connector pattern with 2 nodes, additional nodes may be added by right clucking on the connector icon and selecting the patterns option.

To assign a connector node to a control or indicator on the VI - click on the node to be assigned then click on the required control or indicator.

Rev B, 7/22/15

Controls and indicators are only visible outside the VI if a connector node has been assigned to them.

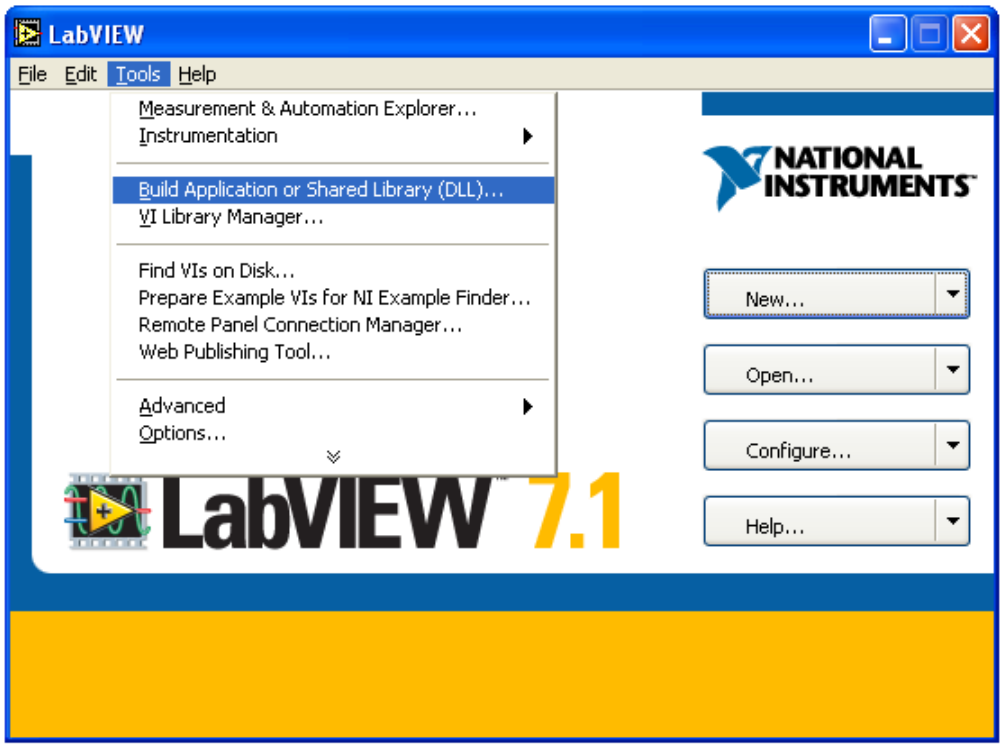
This will be reflected in the dynamic link library (DLL) exported function(s) prototypes when we build the DLL.



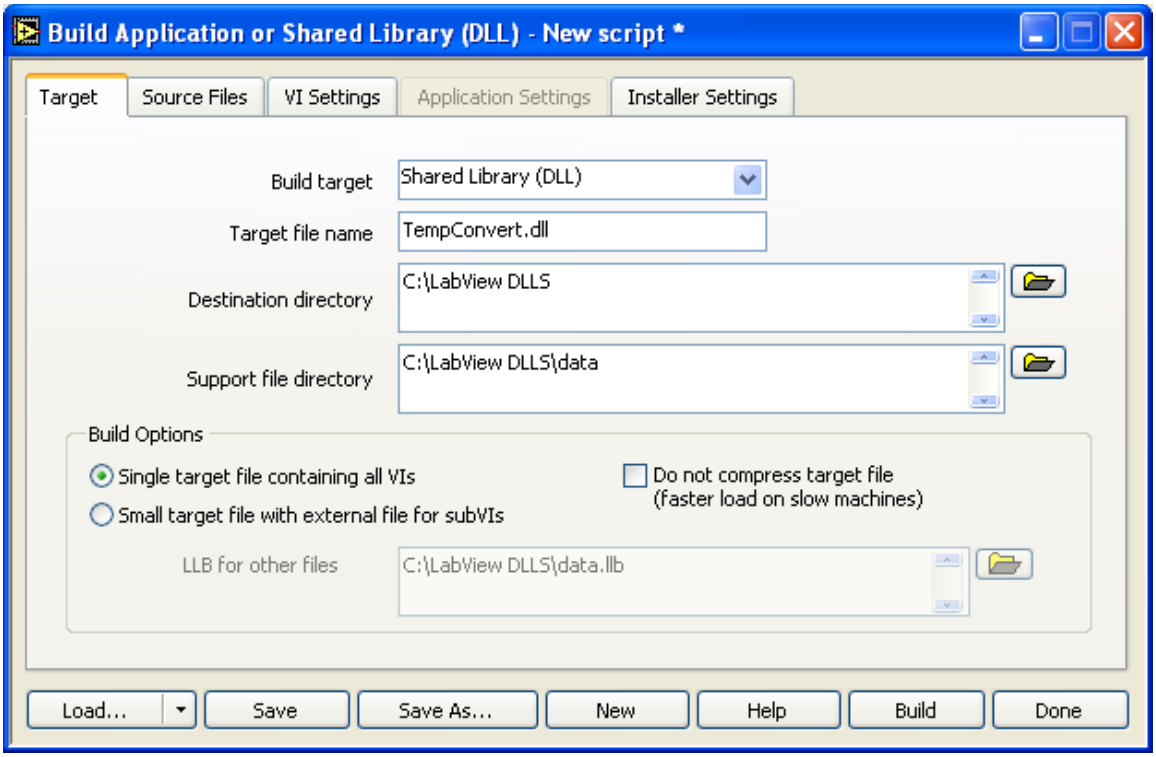
Step 5 – Assign each node of the connector to controls / indicators

In order to build a LabView VI into a dynamic link library you will need the LabVIEW Professional Development System which includes an Application Builder. It is this application builder that is required to build the VI into a dynamic link library (DLL).

To build the VI into a dynamic link library select the “Build Application or Shared Library (DLL)” option from the tools menu from the opening screen of LabView.



Step 6 – Open the “Build Application or Shared Library (DLL)” utility



Step 7 – The Build Application or Shared Library (DLL) utility [Target Tab]

The application builder can build either a Shared library (DLL) or an Application (EXE) the fields available for the Shared Library (DL) are as follows:

Target Tab:

Build Target:

Specifies if the VI shall be build into either a Shared library (DLL) or Application (EXE), select Shared Library (DLL).

Target file name:

Specify the name of the Shared Library.

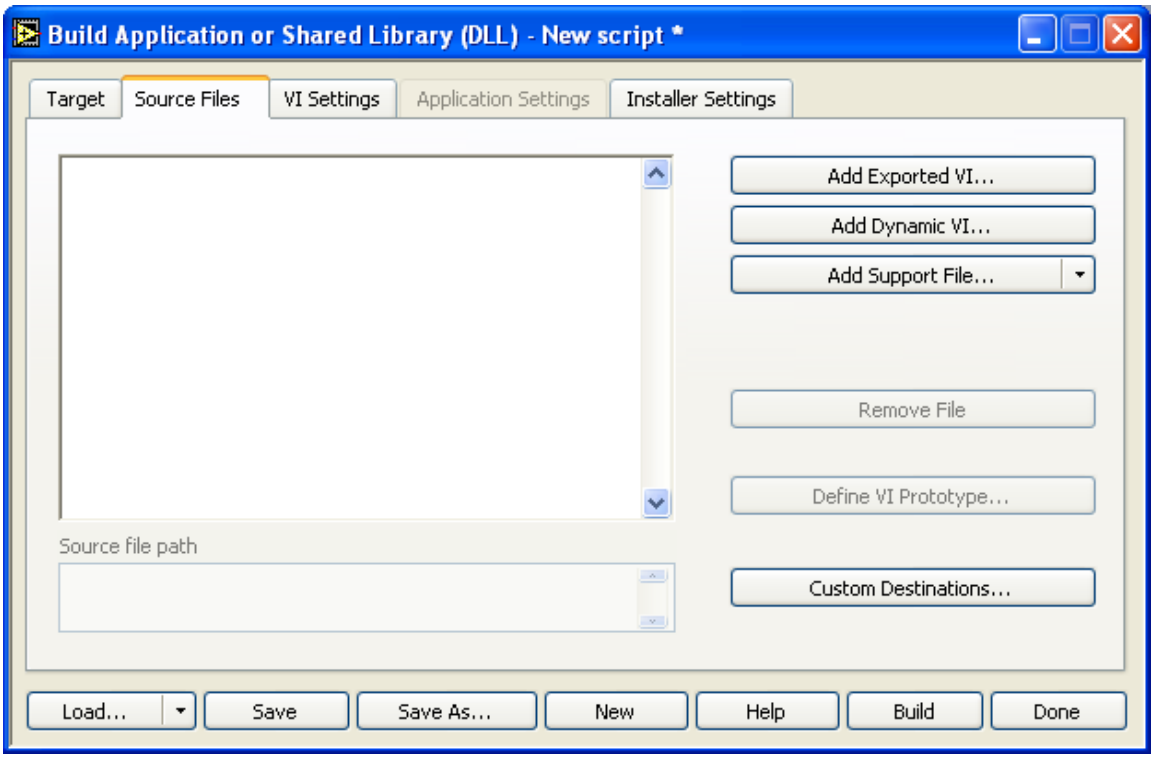
Destination directory:

Specifies the destination where the Shared Library will be created.

Support file directory:

N/A

In Source Files tab select add VI(s) to the Shared Library (DLL).



Step 8 - Build Application or Shared Library (DLL) utility [Source Files Tab]

Add Exported VI:

Adds an exported VI to the shared library. You must have at least one exported VI. When you add an exported VI, LabVIEW automatically includes all its subVIs and related files, such as menu files.

Add Dynamic VI:

N/A

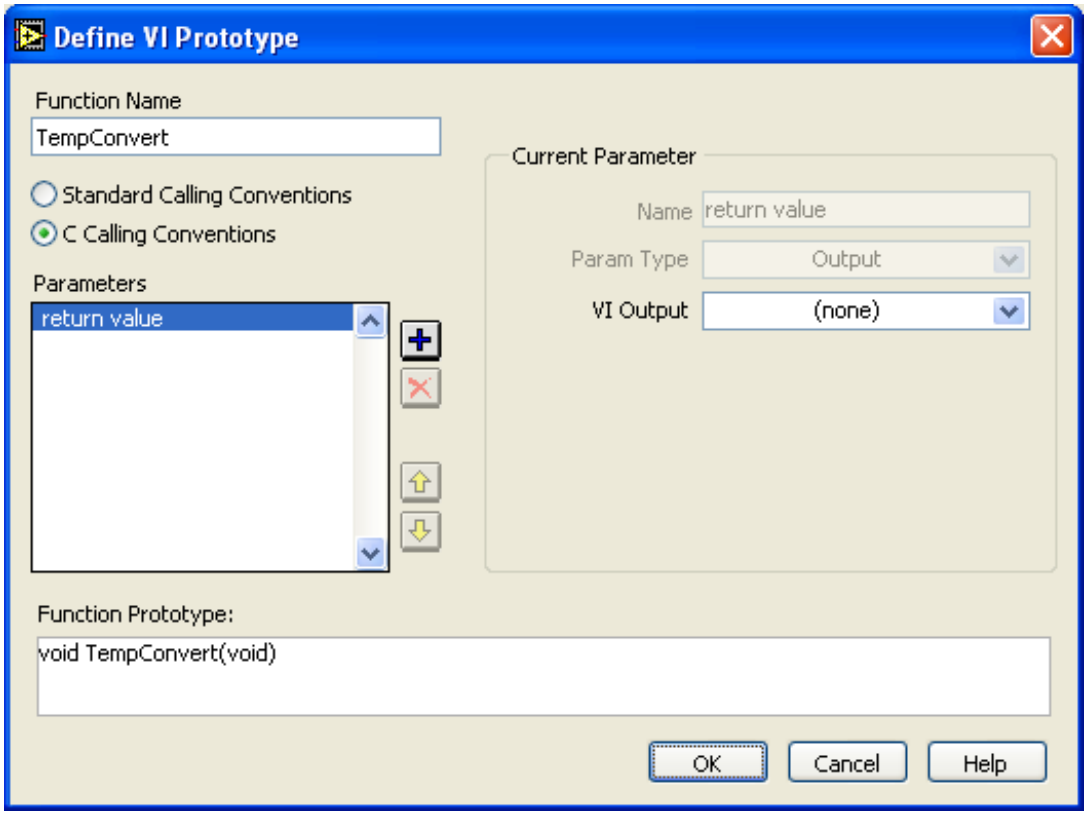
Add Support File:

N/A

Define VI Prototype:

Defines the prototype for the current exported VI that you select in the listbox. The Define VI Prototype dialog box appears automatically when you add an exported VI to the listbox.

Click on Add Exported VI option and select our VI TempConvert.vi, a new dialog Define VI Protype will be automatically displayed.

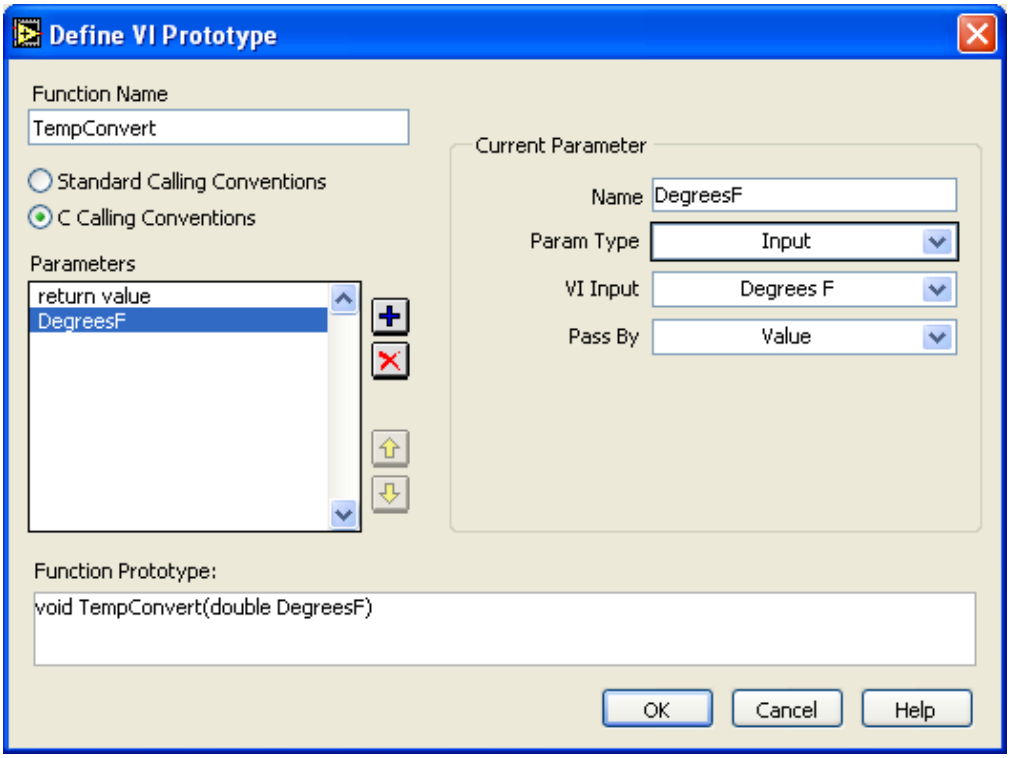


Step 9 – Define VI Prototype

The Function Name will default to the name of the added VI, this will be the exported function name within our Shared Library (DLL). By default the VI will have a prototype of void <Function Name> (void).

In out example we are going to pass a parameter to the VI which is the temperature in Fahrenheit and the function will then return the temperature in Celsius.

To add the temperature to be converted click on the + button to the right of the Parameters list box.

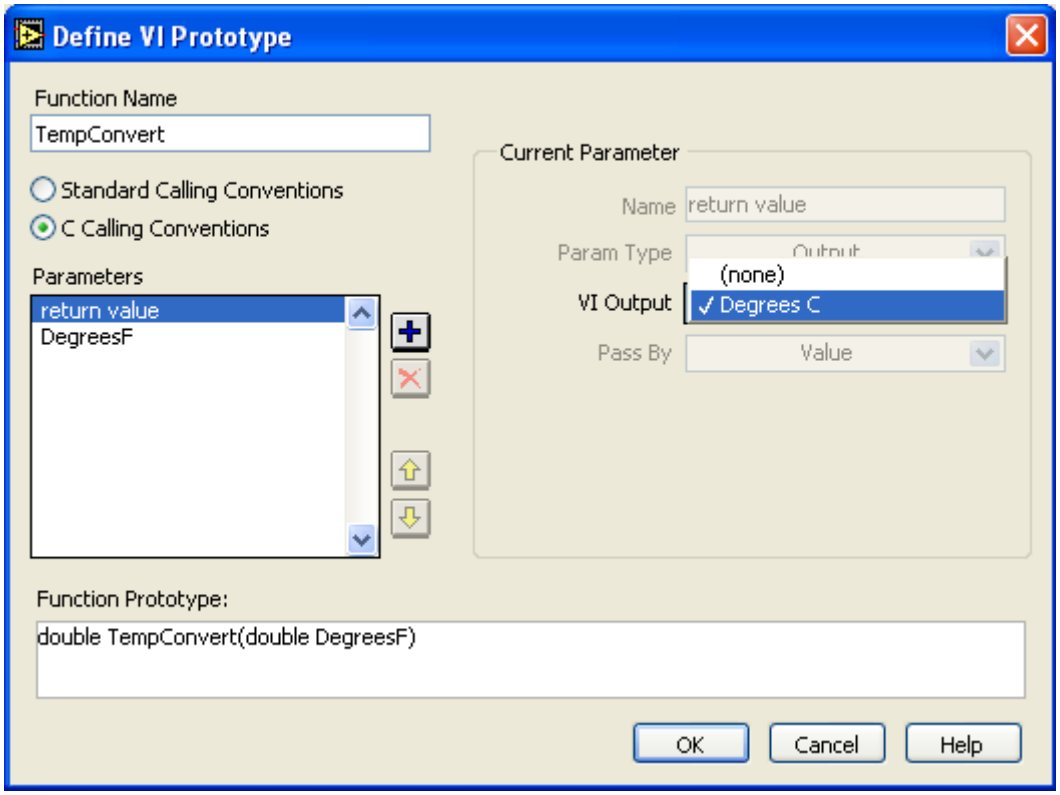


Step 10 – Add function parameter

In the parameters list box we now have a function return value and a parameter DegreesF, the Function Prototype is now displaying void TempConvert(double DegreesF). By selecting the parameters in the list box the Current Parameter group displays the relevant information on the parameter. Parameter details may be changed via the Current Parameter group.

If a VI has multiple exported controls you may selected to add parameters for all or none of the controls, just because a control is exported via the connector pane doesn't mean that it must be included in the function prototype.

To specify the return value from our VI, select the “return value” parameter from the Parameters list box – The Current parameter group will display all possible return indicators on the TempConvert VI. In this case we have only one indicator “Degrees C”.



Step 11 – Assign a return value

Powerstar now has a prototype of:

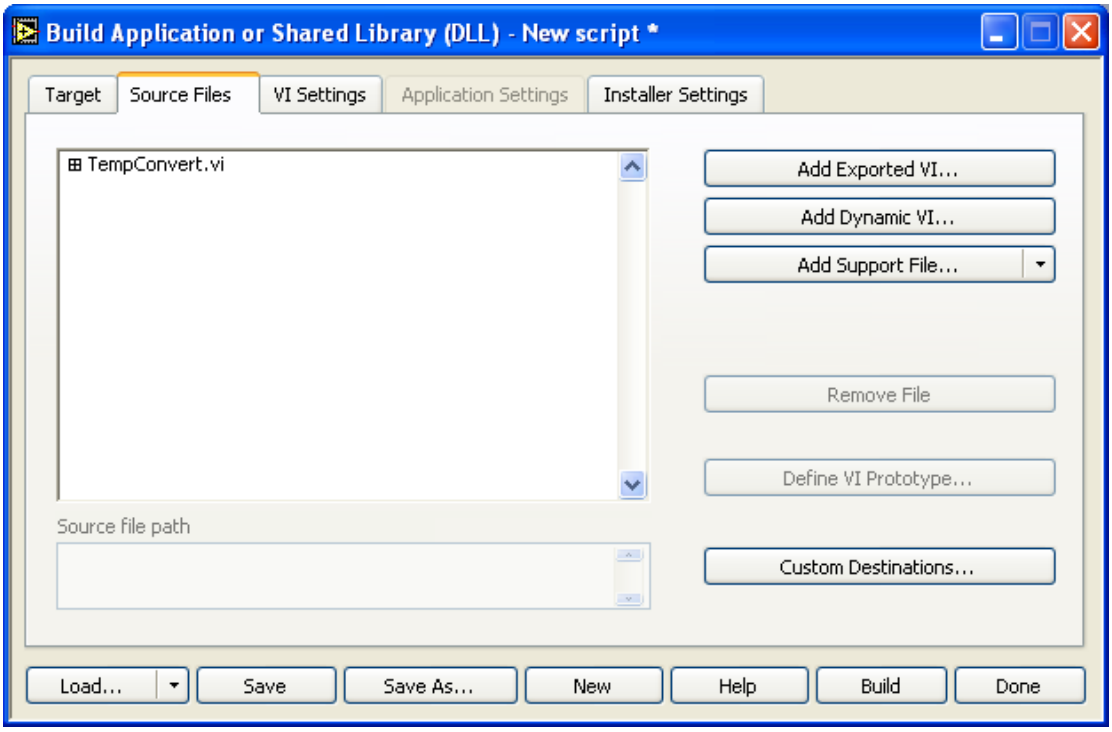
Double TempConvert (double DegreesF)

The Define VI Protype can now be closed.

The final step is now to build TempConvert.DLL

Rev B, 7/22/15

To build the Shared Library (DLL) TempConvert.dll click on the Build button of the application builder utility.



Step 12 – Build TempConvert.dll

When inspecting the Destination Folder (C:\LabView DLLS) you will see that the application builder has created 3 files.

TempConvert.dll	18 KB	Application Extension
TempConvert.h	1 KB	C Header File
TempConvert.lib	3 KB	LIB File

TempConvert.dll — This is the Dynamic Link Library (DLL) which contains the exported function TempConvert.

TempConvert.h — This is a header file which contains the prototype for the function TempConvert. This is only required if you want to call the TempConvert function directly from C / C++.

TempConvert.lib — This is a library file which is required if you want to call the function TempConvert directly from C / C++.

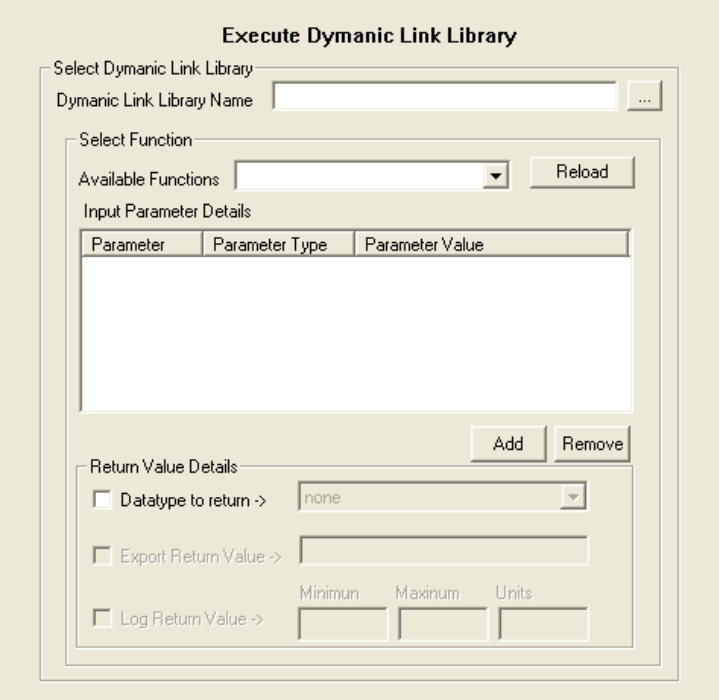
Calling the TempConvert Function from PowerStar

PowerStar 5 has several ways to execute a Dynamic Link Library (DLL) we will look at the following:

1) Execute a Dynamic Link Library function using the test “Execute DLL”.
2) Execute a Dynamic Link Library function from a Virtual Application (VA).
3) Execute a Dynamic Link Library function from a user VBA test.

Executing a Dynamic Link Library function using the test “Execute DLL”

Insert the test “Execute DLL” from the test index under Intepro 9000 > I9000 Control



Execute DLL Test Screen

Dynamic Link Library Name:

Selects the Dynamic Link Library to be executed.

Available Functions:

PowerStar 5 will search the specified Dynamic Link Library and display all exported function within the Dynamic Link Library.

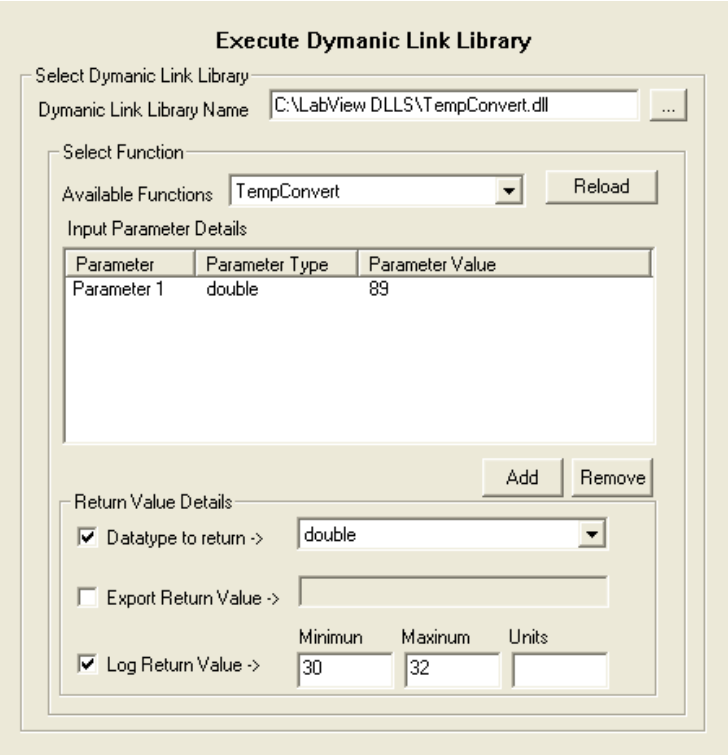
Input Parameter Details:

Contains the list of parameter(s) to be passed as arguments to the selected function.

Rev B, 7/22/15

Datatype to return:

Specifies the function return type, if specified this may be treated as the test result and logged accordingly.



Call the TempConvert Function of TempConvert.dll

The function prototype for TempConvert is:

double TempConvert(double DegreesF)

So in order to execute the TempConvert we must pass the required parameters and also read back the required datatype from the TempConvert function.

The Input Parameters are specified as 1 parameter of type double with a explicit value 89 being passed to the TempConvert Function.

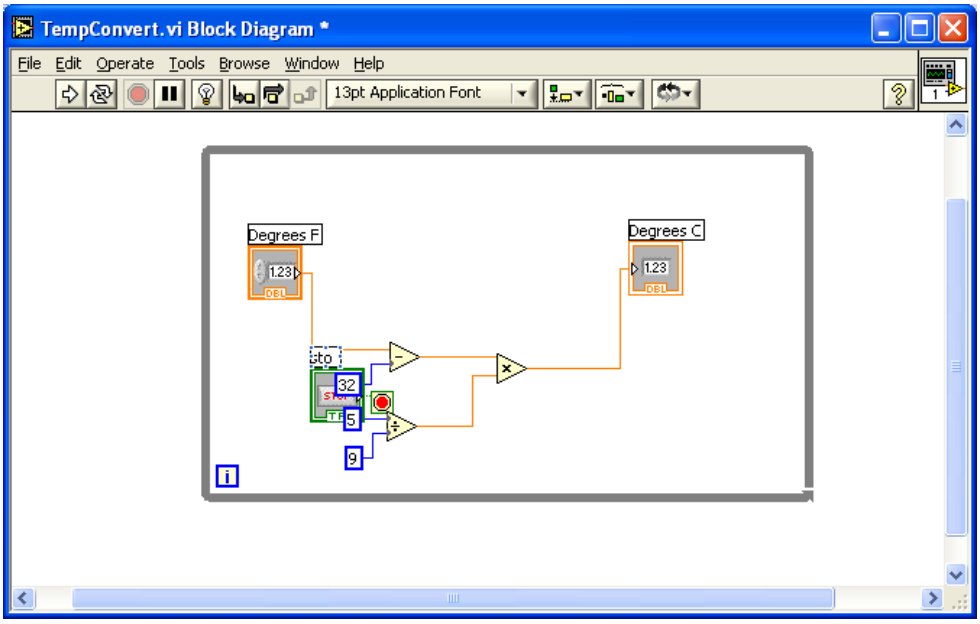
The Return Value Details specifies the return data type as being a double value.

Log Return Value specifies that the value returned from the TempConvert function be logged as the test result.

LabView Front Panels

LabView VI's when built as Shared Libraries (DLL) will not display any front panels by default.

In order to make the TempConvert VI interactive the VI has been modified to add a while loop, the loop will continue until the while break button is pressed.

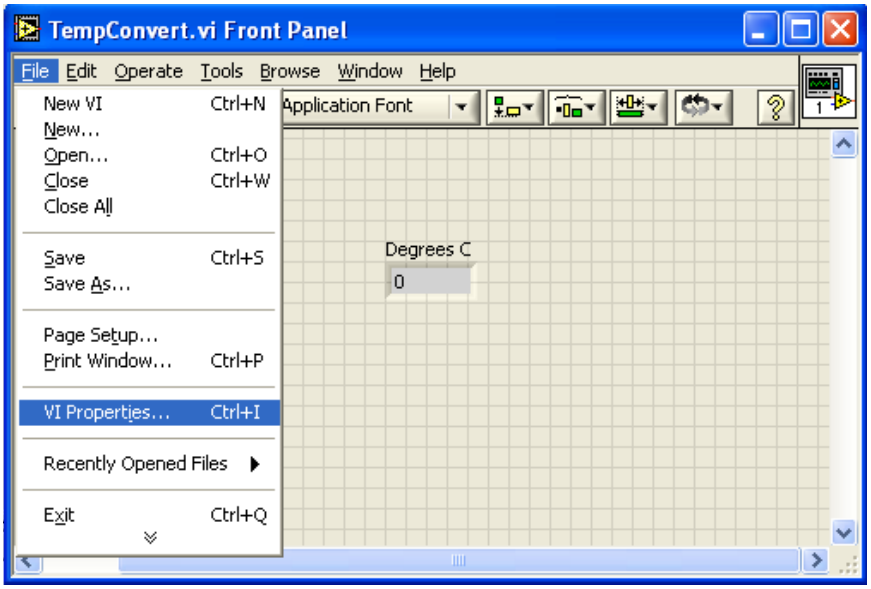


TempConvert VI with a while loop added

The only way for the VI to terminate now is for the user to press the stop button.

Rev B, 7/22/15

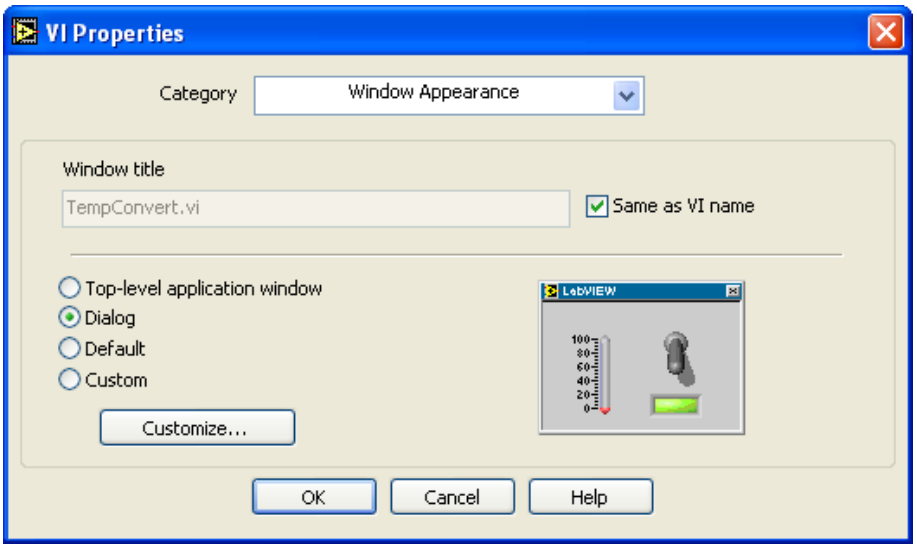
LabView by default will NOT display the Front Panel for a Shared Library (DLL). In order to display the Front Panel you must modify the properties of the VI.



VI Properties

Rev B, 7/22/15

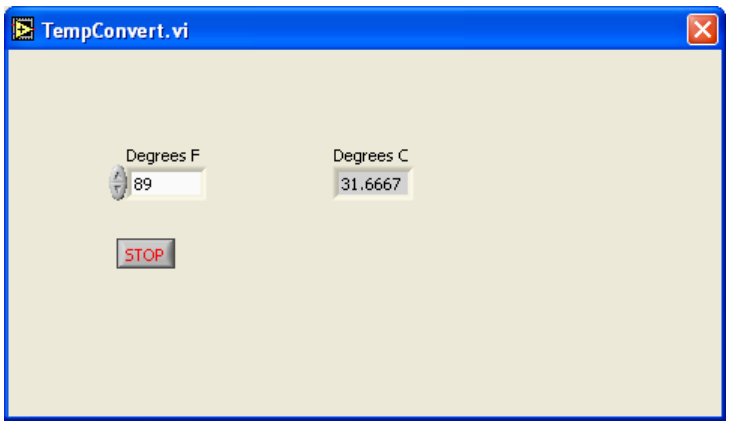
In the Window Appearance Category of the VI Properties specify the Window Appearance as Dialog. Save the VI and rebuild the Shared Library TempConvert.



VI Properties – Window Appearance

By specifying that the VI is a dialog the front panel will now be shown.

Running the Execute DLL test now shows the following



TempConvert function running as a dialog

Rev B, 7/22/15

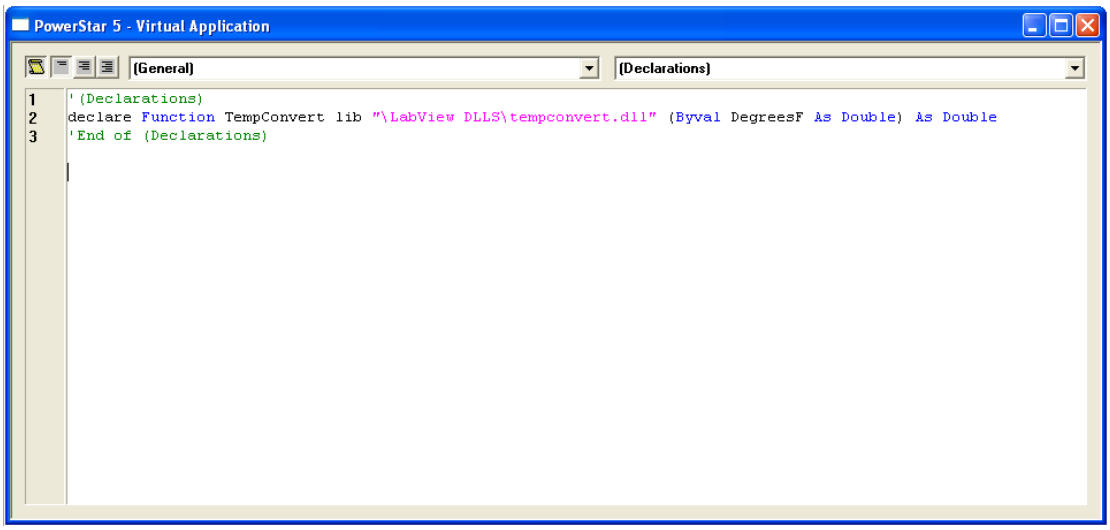
The VI will now show its front panel the initial temperature will be set to 89 but the user may now change the value, PowerStar 5 will wait until the user has pressed the STOP button before reading the return value.

Calling the ConvertTemp function from a Virtual Application (VI)

LabView Dynamic Link Libraries may be called from a PowerStar 5 Virtual Application in the same manner as any Dynamic Link Library.

The function(s) within the Dynamic Link Library must be declared in the General > Declarations section of the Virtual Application. In the case of TempConvert the declaration will be

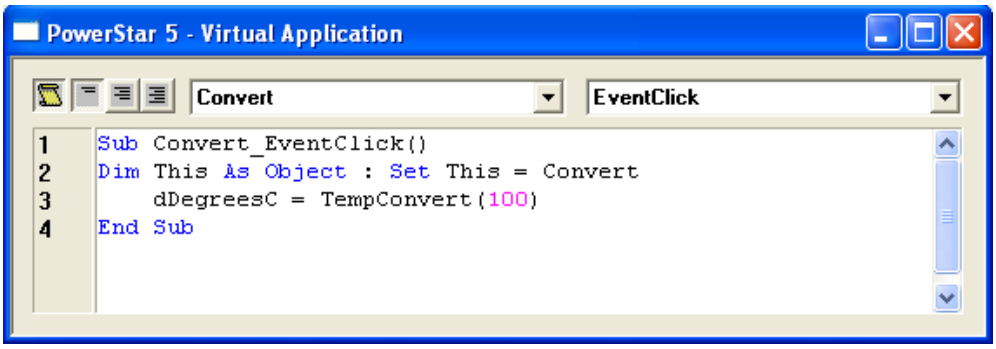
Declare Function TempConvert lib "\LabView DLLS\tempconvert.dll" (byval DegreesF as double) as double



TempConvert function declaration

Rev B, 7/22/15

To execute the function just call the declared function



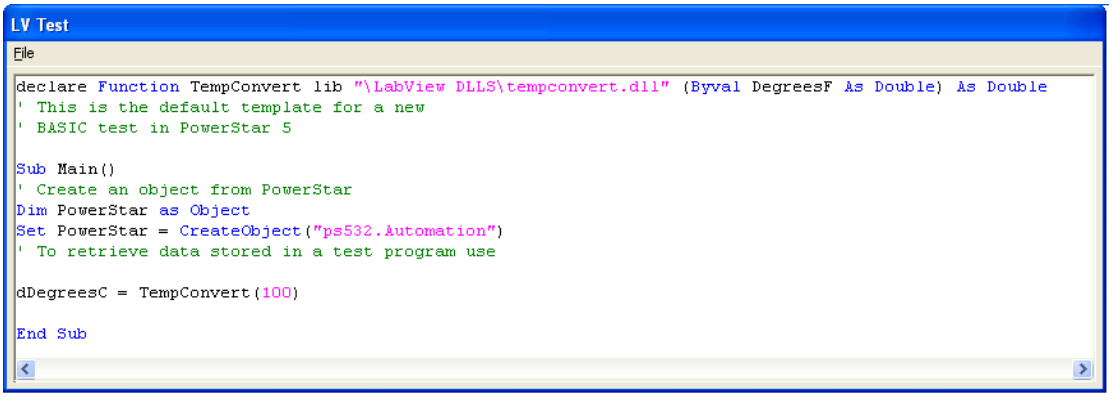
Execute the TempConvert function

Calling the ConvertTemp function from a VBA Test

LabView Dynamic Link Libraries may be called from a PowerStar VBA test in the same manner as any Dynamic Link Library.

The function(s) within the Dynamic Link Library must be declared globally within the test code of the VBA test. In the case of TempConvert the declaration will be

Declare Function TempConvert lib “\LabView DLLS\tempconvert.dll” (byval DegreesF as double) as double



Rev B, 7/22/15

LabView Shared Libraries and Multithreaded Applications

PowerStar 5/6 test programs run in a separate thread to the PowerStar 5 UI and Virtual Applications. LabView applications are not complete standalone and require the appropriate runtime engine. Due to threading issues with the initialization and release of the runtime engine PowerStar 5/6 requires that the supplies LabView VI (LVRTEInit.vi) be built as a dynamic link library using your version of LabView and placed in the PowerStar 5\6 Bin folder.

This will enable PowerStar to initialize your runtime engine in the main PowerStar thread.

Rev B, 7/22/15