

Building a GUI using TI GUI Composer

Objective:

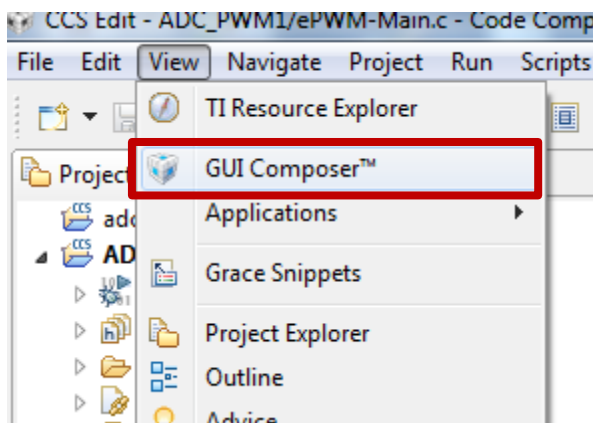
To build a basic GUI using TI GUI Composer to view a PWM waveform read from the ADC of the C2000 MCU using a Line Graph. To change the PWM duty cycle using a Dial and view the changes on the Graph.

Getting Started:

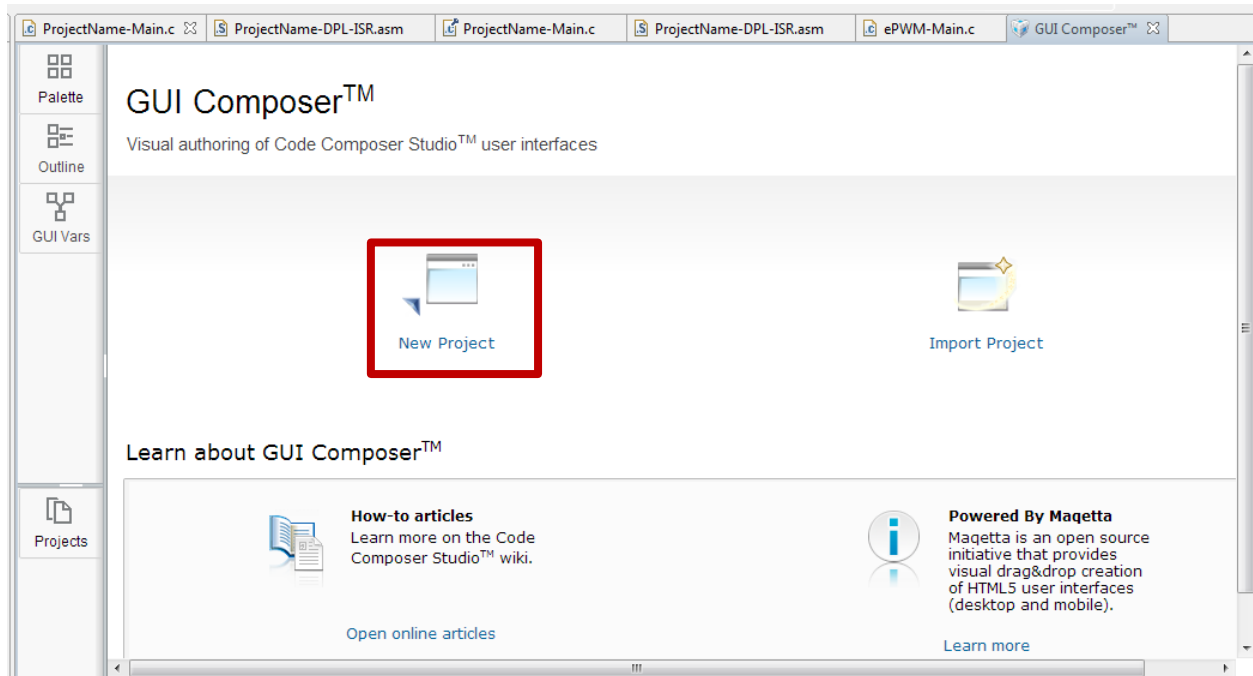
To build a GUI using GUI composer, you must first have a fully working code.

GUI Composer widgets “Bind” themselves to a global variable in your project, and monitor/modify these variables realtime.

To start GUI Composer, go to View>GUI Composer

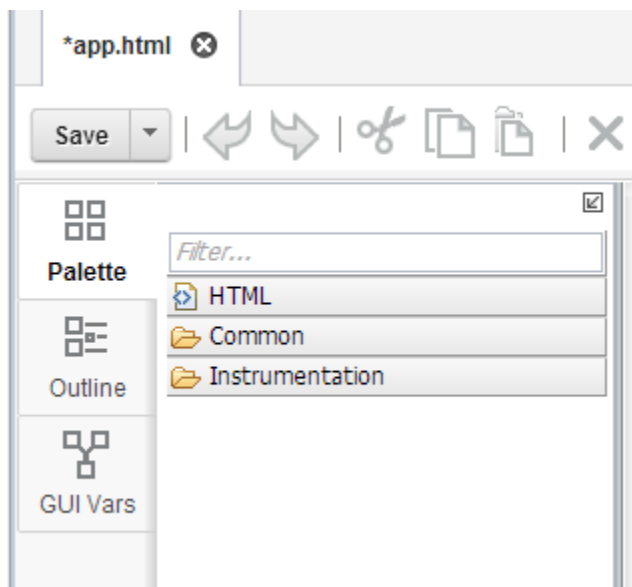


Click on “New Project” to start creating a new GUI



Enter the name of your GUI and click OK to continue

The widgets can be selected from the left hand pane “Palette”.

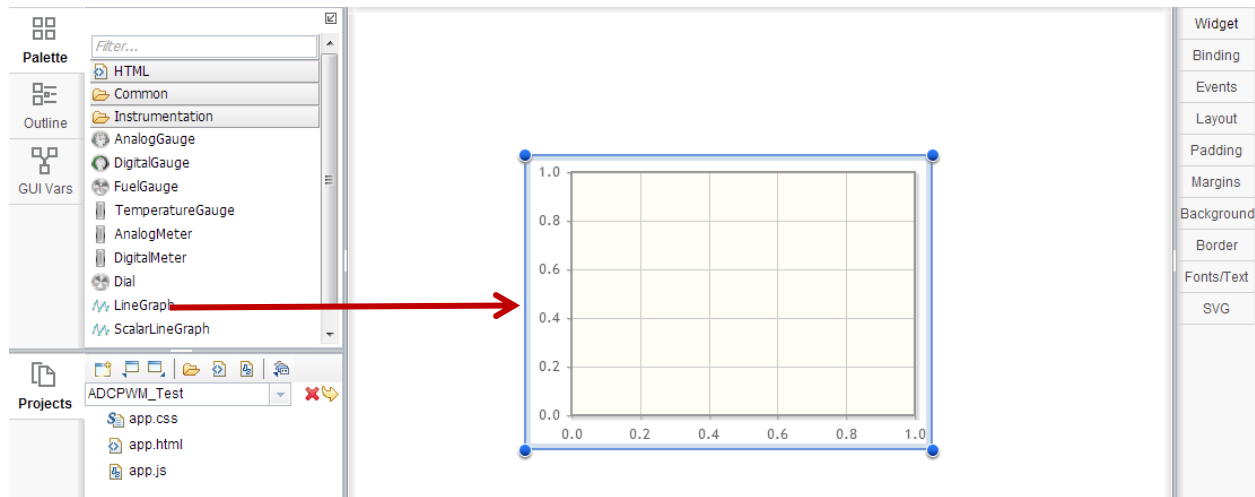


“Common” contains basic widgets like text boxes, labels, slider bars etc.

“Instrumentation” contains measurement and control widgets like dials, knobs, graphs, meters etc.

Using a Line Graph to view the ADC buffer contents

Under Instrumentation, Drag and drop the Line Graph on the project as shown below:












Resize and position the widget as per your requirements, and click on the “Widget” pane on the right hand side of the screen to edit widget properties:

Change the widget properties as per requirements:

Type in the Graph title, the x and y axis limits, and the axes labels. Note that the y axis range must be equal to the size of the buffer you are sampling.

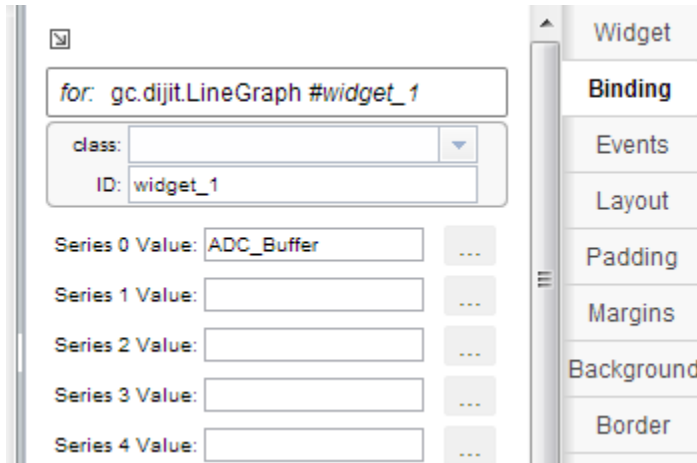
You can optionally change the background and line graph colours from the widgets menu as well.

Title:	ADC Sample	
Auto Scale:	on	▼
X-Axis Minimum Value:	0	
X-Axis Maximum Value:	50	
Show X-Axis Ticks:	<input checked="" type="checkbox"/>	
X-Axis Label:		
Y-Axis Minimum Value:	0	
Y-Axis Maximum Value:	5000	
Show Y-Axis Ticks:	<input checked="" type="checkbox"/>	
Y-Axis Label:		
Series 0 Label:		
Series 1 Label:		
Series 2 Label:		
Series 3 Label:		
Series 4 Label:		
Series 5 Label:		
Series 6 Label:		
Series 7 Label:		
Series 0 Color:	red ▼	
Series 1 Color:	▼	
Series 2 Color:	▼	
Series 3 Color:	▼	
Series 4 Color:	▼	
Series 5 Color:	▼	
Series 6 Color:	▼	
Series 7 Color:	▼	
Background Color:	black ▼	

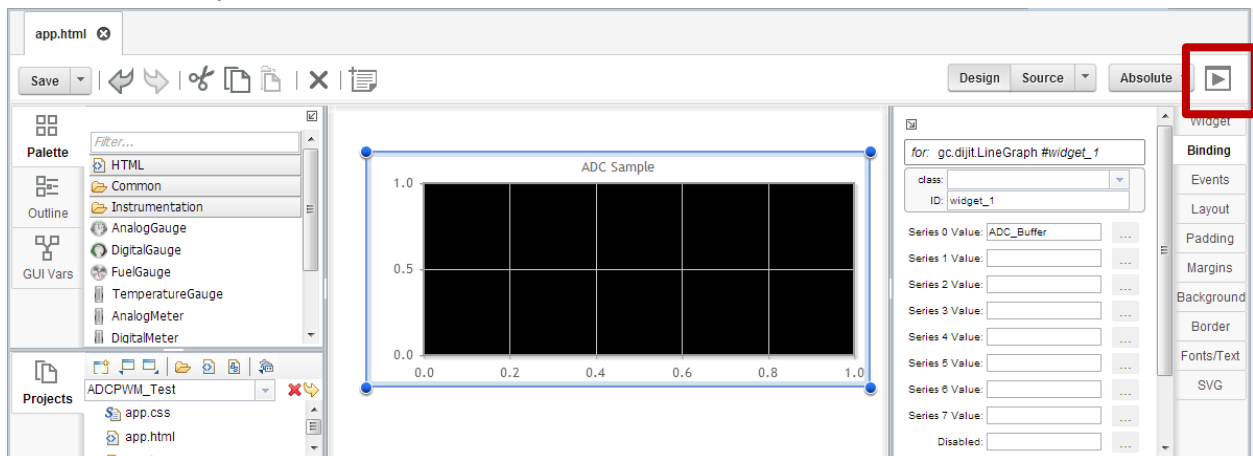
Next, click on the “Binding” Pane on the right hand side of the screen.

Enter the name of the global variable you intend to monitor under “Series 1 Value”

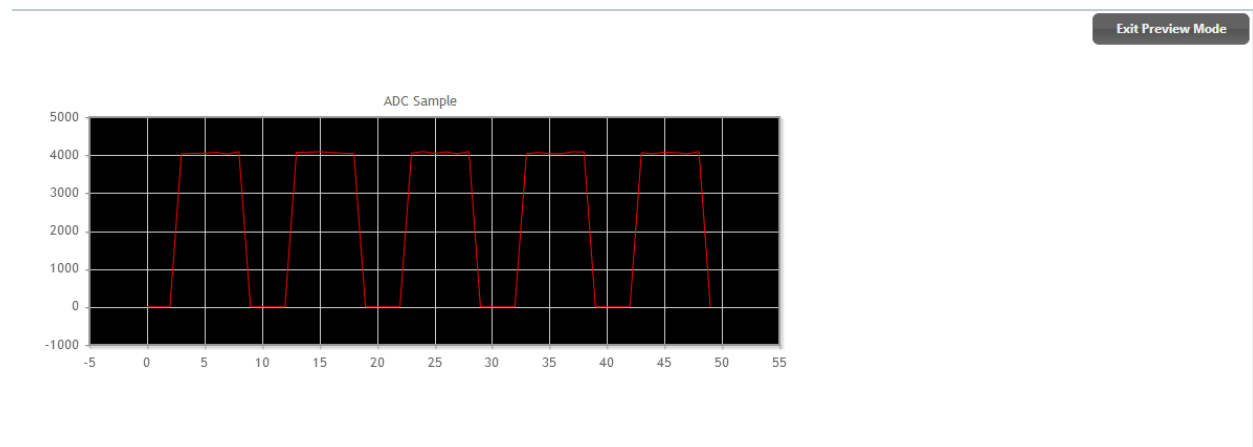
In this case, we will monitor ADC_Buffer on the line graph.



To preview the GUI, build the code that will run on the MCU and enter debug mode. Once the code is running on the MCU, click on the preview button to enter preview mode. Now you can preview the GUI and view it exactly as the end user would view it.



The preview mode will look like the following image:



Click on Exit Preview Mode when done.

Inserting a Dial to change the PWM Duty Cycle

Adding the Dial:

Drag and drop a Dial from the instrumentation pane.

In its widget properties, give it the title “PWM Duty”

Change the min and max. values to 0 and 1 respectively.

Widget

for: `gc.dijit.Dial #widget_10`

class:

ID: `widget_10`

Title:

Unit:

Minimum Value:

Maximum Value:

Set Value On Drag: ☐

Current Value:

Tick Labels:

Number Format:

Fractional Decimals:

Frame Design:

Dial Design:

Disabled: ☐

Visible: ☒

Read Only: ☐

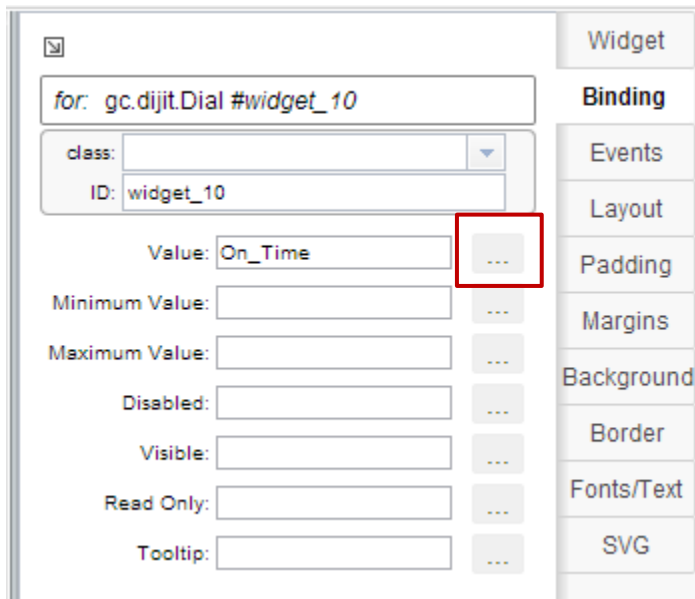
Tooltip:

Widget

- Binding
- Events
- Layout
- Padding
- Margins
- Background
- Border
- Fonts/Text
- SVG

You can change the frame and dial design as per your liking.

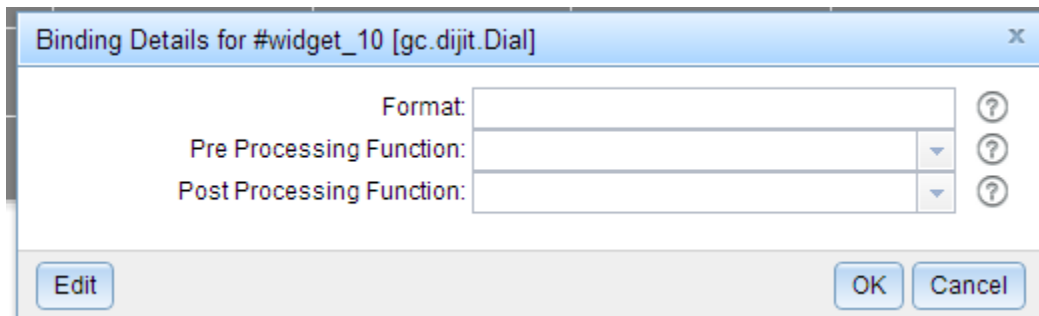
In the “Binding” menu, add `On_Time` as the bound variable as shown:



Now, `On_Time` must be scaled down to a 0-1 range when it is transferred from the MCU to the GUI and scaled up to a 0-10000 range when it is sent from the GUI to the MCU.

This scaling can be done by clicking on the “...” button next to the “Value” field.

The following screen will follow:

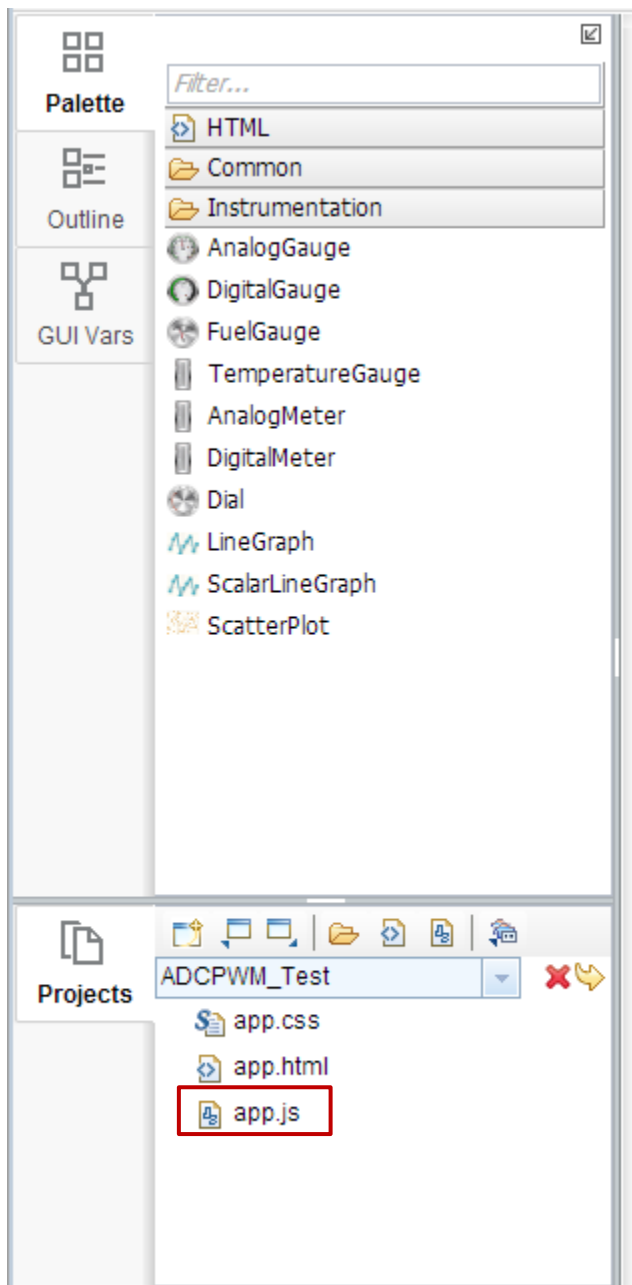


The preprocessing function is used to scale the variable being read from the MCU, and the post processing function is used to scale the variable being written to the MCU.

Type “ScaleDown” in the Pre processing function field, and “ScaleUp” in the post processing function field.

This will create the functions ScaleDown and ScaleUp in the app.js file.

Open the app.js file from the project files menu:



It will open in the Editor.

The following functions are created in the app.js file:

```
function ScaleDown( valueFromTarget) {  
    // return valueFromTarget/2;  
    return valueFromTarget;  
}  
  
function ScaleUp( valueToTarget) {  
    // return valueToTarget*2;  
    return valueToTarget;  
}
```

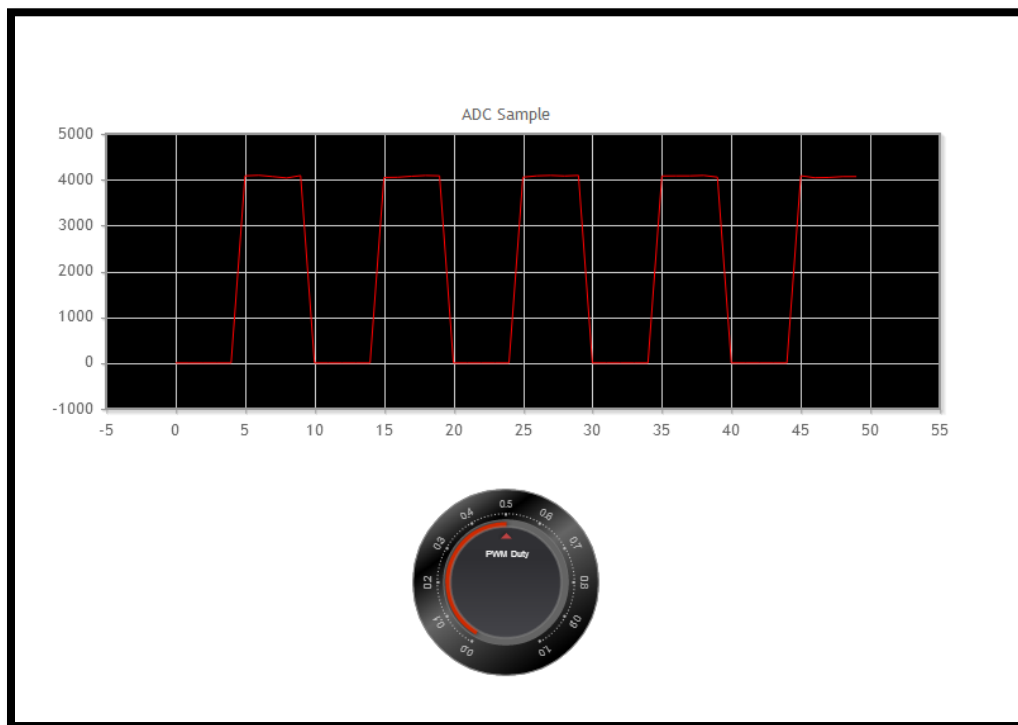
Edit the functions as follows:

```
function ScaleDown( valueFromTarget) {  
    // return valueFromTarget/2;  
    return valueFromTarget/10000;  
}  
  
function ScaleUp( valueToTarget) {  
    // return valueToTarget*2;  
    return valueToTarget*10000;  
}
```

Save the app.js file, enter debug mode and preview the GUI.

Short ADC A0 (J5 6th Pin on the Launchpad) to ePWM1 output (J6 1st Pin)

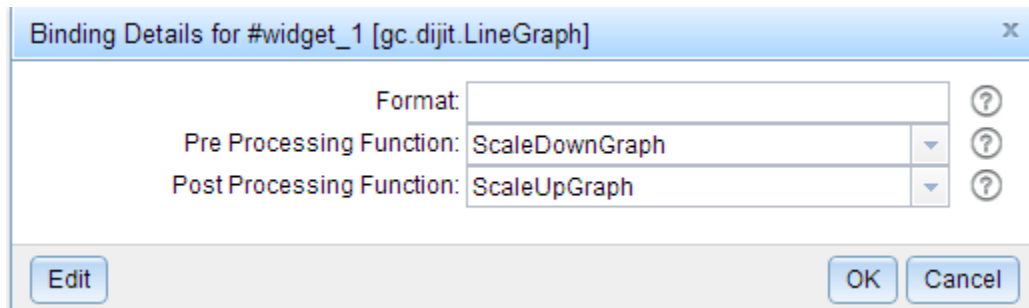
You will now be able to change the PWM duty cycle and view the PWM waveform from the Line Graph.



Scaling the Line Graph

For scaling the line graph, follow the same procedures. Add ScaleDownGraph as the preprocessor, and ScaleUpGraph as the post processor.

The line graph reads an array from the MCU. Therefore, the scaling must be applied to each element of the array. Open the app.js file:



In app.js, the functions will be created as follows:

```
function ScaleDown( valueFromTarget) {  
    // return valueFromTarget/2;  
    return valueFromTarget/10000;  
}  
  
function ScaleUp( valueToTarget) {  
    // return valueToTarget*2;  
    return valueToTarget*10000;  
}  
  
function ScaleDownGraph( valueFromTarget) {  
    // return valueFromTarget/2;  
    return valueFromTarget;  
}  
  
function ScaleUpGraph( valueToTarget) {  
    // return valueToTarget*2;  
    return valueToTarget;  
}
```

The graph scaling functions should be modified as follows.

```
function ScaleDownGraph( valueFromTarget) {  
    var result = [];  
    for (var i = 0; i < valueFromTarget.length; ++i) {  
        result.push((valueFromTarget[i]*3.3)/4095);  
    }  
    return result;  
}
```

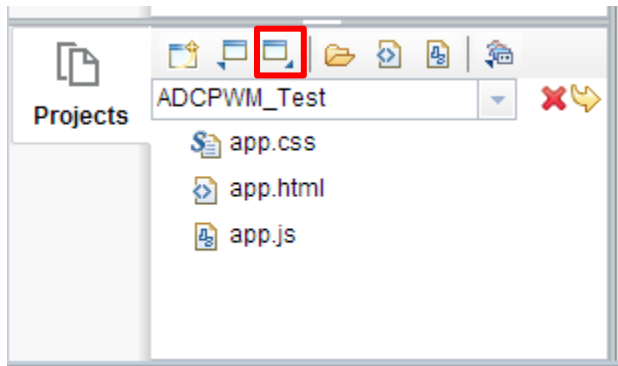
The ScaleUpGraph function should be modified as follows.

```
function ScaleUpGraph( valueToTarget) {  
    var result = [];  
    for (var i = 0; i < valueToTarget.length; ++i) {  
        result.push((valueToTarget[i]/3.3)*4095);  
    }  
    return result;  
}
```

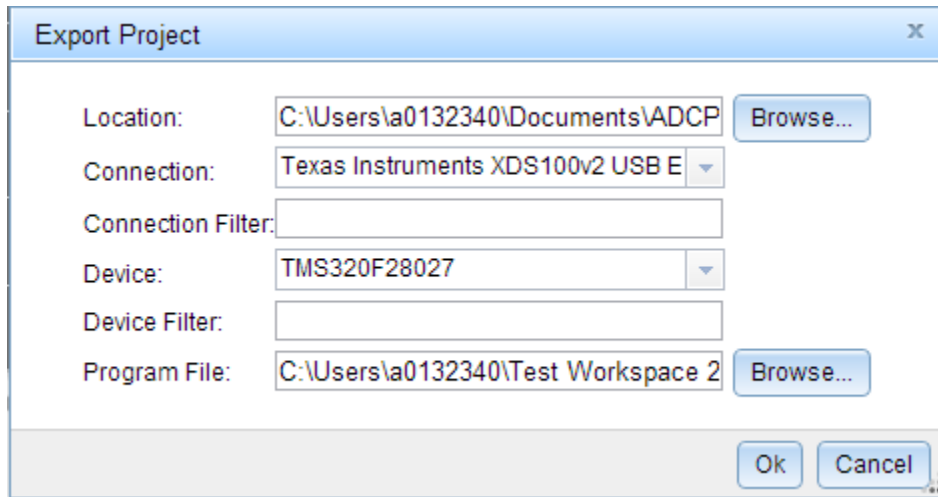
These functions will now make the graph display the ADC Values in a 0-3.3V range. (4095 being 3.3V and 0 being 0V)

Exporting the GUI to run from Windows Environment

When you are finished with the GUI, click the “Export” button as shown below:



You will now have to enter the export settings:

The image shows a Windows-style dialog box titled "Export Project" with a close button (X) in the top right corner. The dialog contains several input fields and buttons. The "Location:" field is a text box containing "C:\Users\la0132340\Documents\ADCP" with a "Browse..." button to its right. The "Connection:" field is a dropdown menu showing "Texas Instruments XDS100v2 USB E". The "Connection Filter:" field is an empty text box. The "Device:" field is a dropdown menu showing "TMS320F28027". The "Device Filter:" field is an empty text box. The "Program File:" field is a text box containing "C:\Users\la0132340\Test Workspace 2" with a "Browse..." button to its right. At the bottom right of the dialog are "Ok" and "Cancel" buttons.

Location: Enter the location where you want to export the application

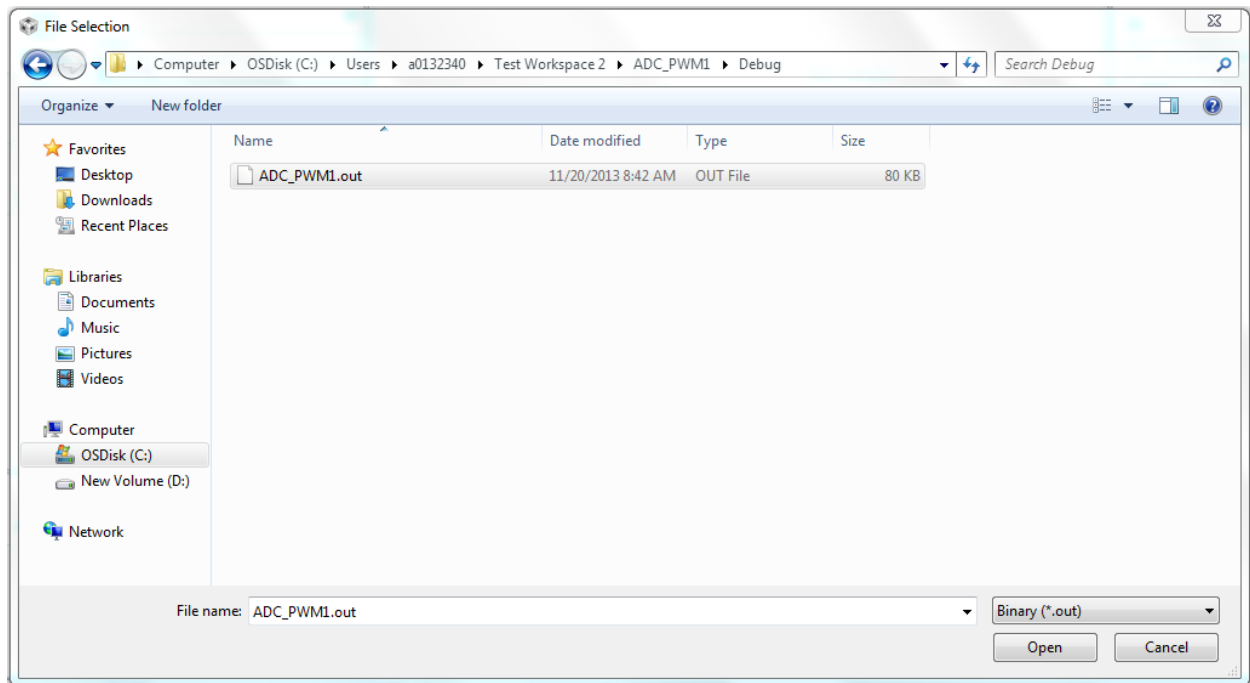
Connection: Type of JTAG connection. Select "Texas Instruments XDS100v2 USB Emulator" when operating from the Launchpad.

Device: Launchpad Device is TMS320F28027

Program File: This is the ".out" file that will be flashed to the MCU whenever the GUI is launched. Click Browse, Navigate to your present workspace. Enter the project folder of the project which you want to use for the GUI.

The ".out" file that you need is found in the build configurations folders (Debug or Release folders. If you have other names for the build configurations such as FLASH or RAM_Ink, navigate into those folders.

When you find the ".out" file, select the file and click "Open". The following screenshot is provided for your reference:



When you have entered all the Project Export fields, click OK.

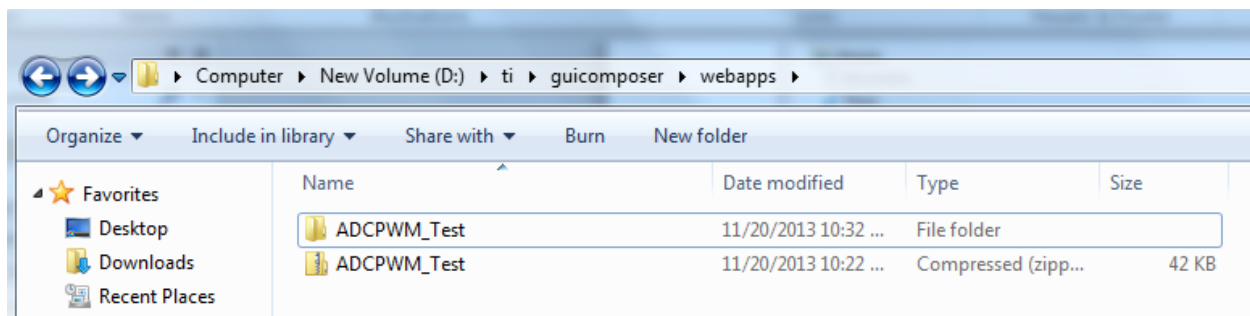
Next, navigate to the folder which you specified as the “Location”

Find the “.Zip” file in that folder.

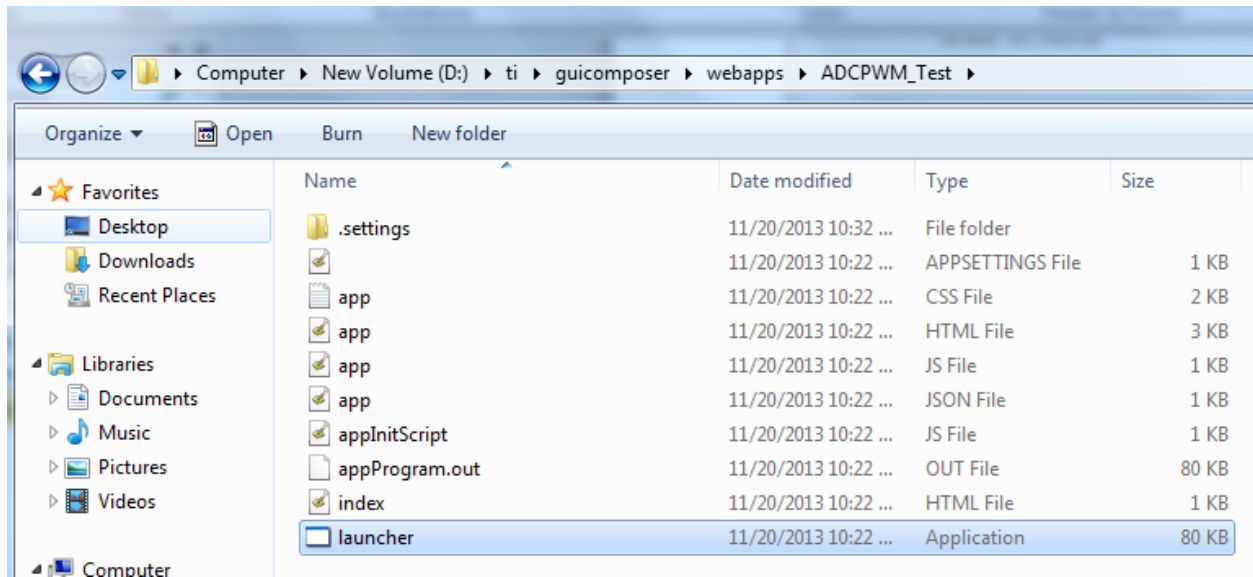
Copy the “.zip” file.

Navigate to C:\ti\guicomposer\webapps\

Paste the zip file in this folder and extract its contents as shown below:



Enter the ADCPWM_Test Folder, and click on the “launcher.exe” file:



This will launch the GUI:

GUI Composer runtime will automatically connect to your target board, and flash the right “.out” file into the MCU. The app will then automatically launch:

