GUIX "Hello World" for DK-S7G2

GUI Application

R12AN0001EU0101
Rev.1.01
Jul 6, 2016

Introduction

This application note describes the process of creating a simple two-screen GUI using GUIX Studio for the DK-S7G2. This application demonstrates how easily a user can create and configure a new application using the Renesas Synergy™ Software Package (SSP).

The Synergy Software Package includes Express Logic's ThreadX® real-time operating system (RTOS), the X-Ware suite of stacks (NetX™, USBX™, GUIX™, FileX®) and hardware drivers unified under a single robust framework. This powerful suite of tools provides a comprehensive integrated framework for rapid development of complex embedded applications.

The "Hello World" application was developed under e^2 studio using the Synergy Framework.

Target Device

DK-S7G2 board version 3.0

Minimum PC Recommendation

- Microsoft® Windows® 7
- Intel® Core™ family processor running at 2.0 GHz or higher (or equivalent processor)
- 8-GB memory
- 250-GB hard disk or SSD
- USB 2.0
- Connection to the Internet

Installed Software

- Synergy e^2 studio 5.0.0.043 or later
- Synergy Software Platform (SSP) v1.1.0
- GUIX Studio v5.3.0.1

NOTE:

If you do not have one of these software applications you should install it before continuing.

Provided Software Files

- guiapp_event_handlers.c
- main_thread_entry.c
- R7FS7G27H2A01CBD.pincfg

Purpose

This document will guide you through the setup of a GUIX touch screen interface "Hello World" application in e^2 studio. This document will show how to configure the drivers and framework included with the SSP. These will allow you to set up the LCD Controller, touch screen drivers, and messaging framework to communicate with application tasks. It also shows the steps necessary to create a simple GUI interface using the GUIX Studio editor. In addition, this app note will also cover project setup in e^2 studio along with basic debugging operations. When it is running, the application will respond to touchscreen actions, presenting a basic graphical user interface (GUI).

Intended Audience

The intended audience is users who want to design GUI applications.

Contents

1. Overview 2
2. Import the Project into e² studio 2
3. Creating the Project in e² studio ISDE (Integrated Solution Development Environment) 2
4. Configure the Project in the ISDE 8
5. Create the GUIX Interface Using GUIX Studio 27
6. Adding Code for Custom Interface Controls 39
7. Running the Application 40

1. Overview

This document explains the steps to setup a project and develop a simple GUI based application using GUIX Studio.

2. Import the Project into e² studio

NOTE:

This step is included to allow you to skip the development steps and just jump to the point of verifying a working project on the DK-S7G2. Most people SKIP THIS STEP and proceed to step 3 to create a project in e2 atudio. If you do import the project, then skip to section 7, Running the Application.

To skip the development walkthrough in this document and open a completed project in e² studio, refer to the *Synergy Project Import Guide*, r11an0023eu0110_synergy_ssp.pdf (included). It contains instructions on importing the project into e² studio and building the project. The included "GUIX_Hello_World_DK-S7G2.zip" file contains the completed project.

3. Creating the Project in e² studio ISDE (Integrated Solution Development Environment)

Start by creating a new project in e² studio.

1. Open e² studio by clicking on the "e² studio" icon in the Windows Start Menu -> "All Programs -> Renesas Electronics e² studio" folder.

2. If the workspace launcher dialog box appears, click <OK> to use the default workspace.

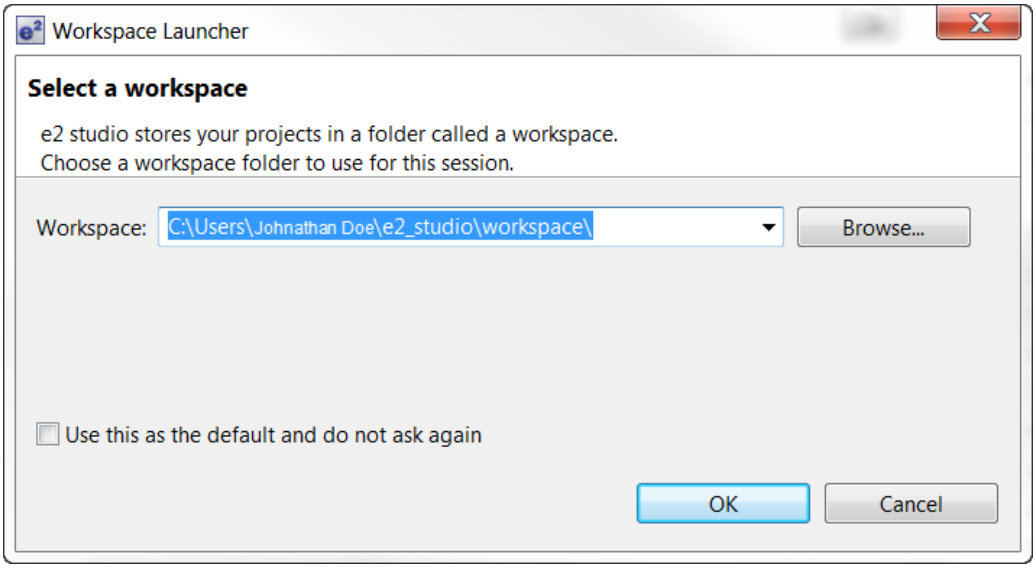

Figure 1 Workspace Launcher Dialog

3. Create a new workspace:
   A. From the “File” pulldown menu, select “Switch Workspace -> Other…”
4. Append a workspace name:
   A. In the Workspace Launcher window, add text to the end of the workspace name to make it unique. We suggest “GUI_APP”. If you installed to the default location, the new workspace name will be “C:\Users\[your name]\e2_studio\workspace\GUI_APP”.
5. Click <OK> to create the new workspace.
6. Proceed past the Welcome Screen by clicking in the Workbench area.

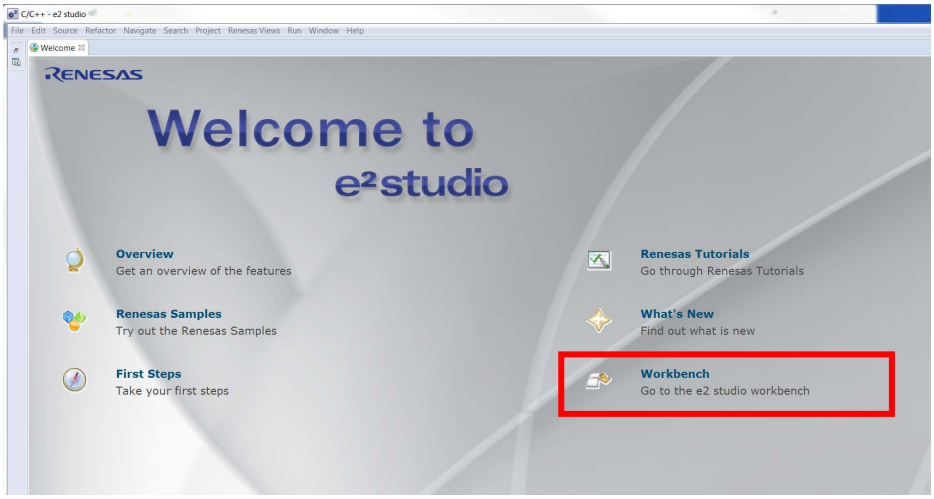

Figure 2 Close the Welcome Window by Clicking in the Workbench Area

7. Start a new project by clicking the black triangle, , next to the New icon in the Tool Bar.

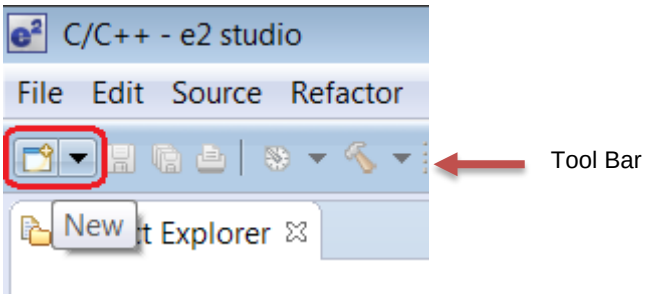

Figure 3 Start a New Project

8. Select “Synergy Project” from the menu.

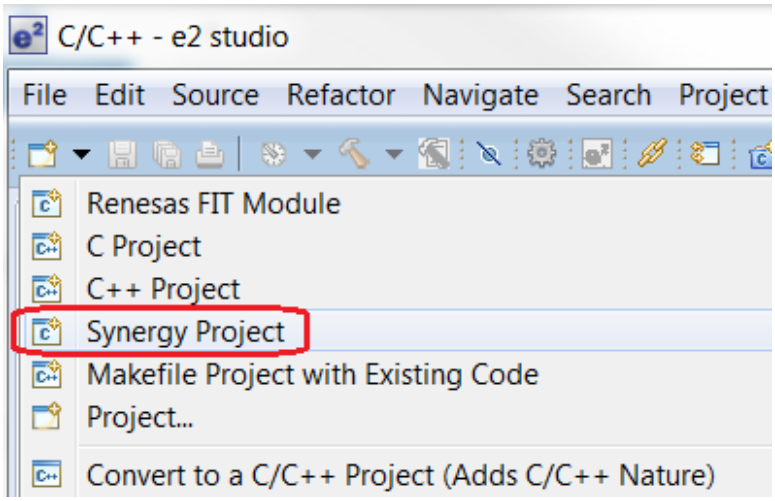

Figure 4 Select Synergy Project in the Drop Down Menu

9. If the License file is configured, the License area of the form will look like Figure 5. If it does, skip to step 10. If it is empty, continue with the below steps (A to G).

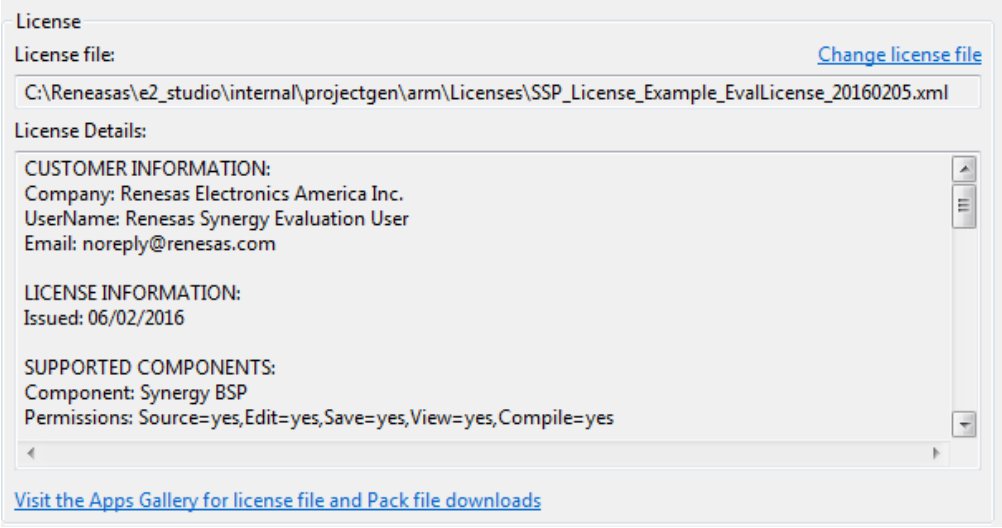

Figure 5 Configured License File

A. Click the <Change license file> button. e² studio will display the “Preferences” dialog box.

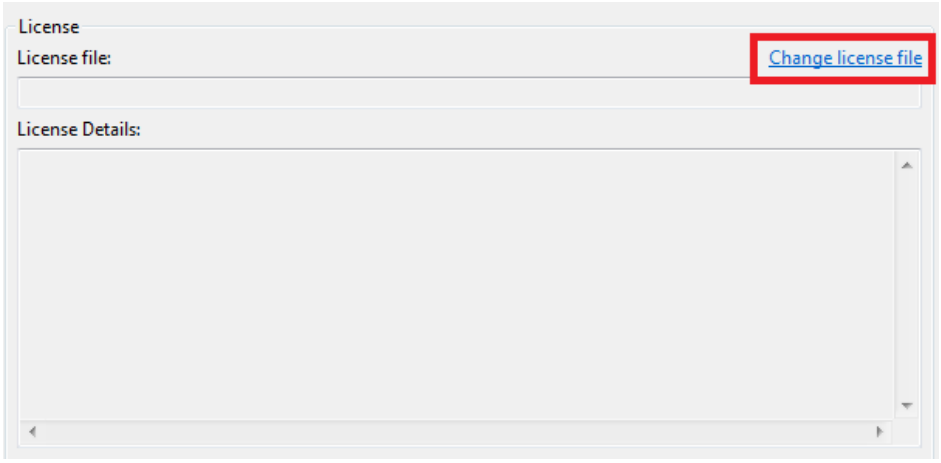

Figure 6 Unconfigured License File

B. Click the browse <…> button to open the “Specify Synergy License” dialog box.

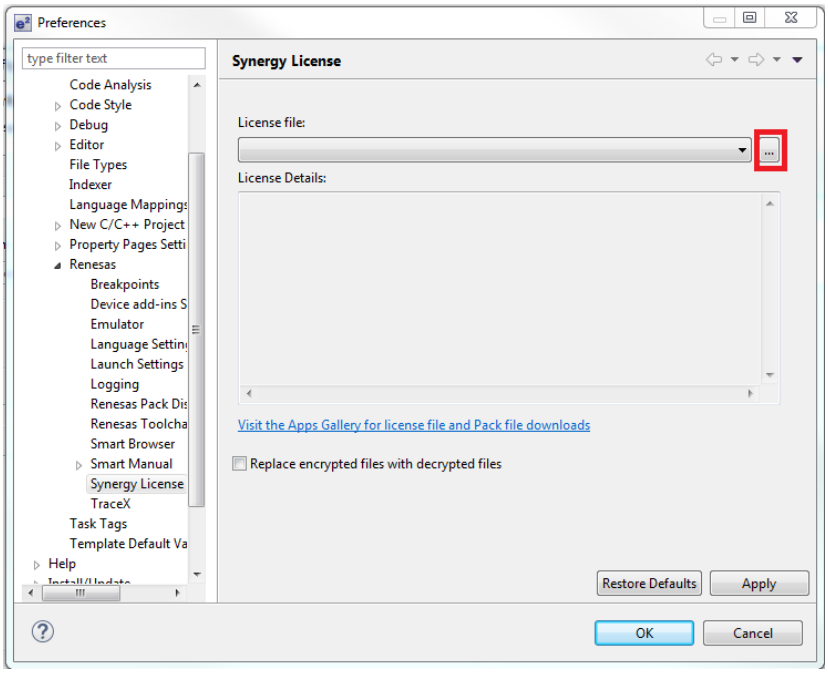

Figure 7 Preferences Dialog Box with Synergy License Configuration

C. Click the <Browse…> button. e² studio will display the Open Dialog box and should be displaying the Licenses directory.

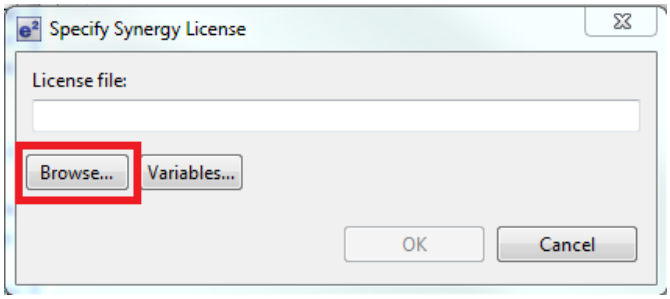

Figure 8 Synergy License Dialog box

NOTE:
If you installed e² studio into the default location, the license file is located in the "C:\Renesas\e2_studio\internal\projectgen\arm\Licenses" directory.

D. Select "SSP_License_Example_EvalLicense_*.xml" located in the directory.

E. Click <Open> to select the License file.

F. Click <OK> to set the license and close the dialog.

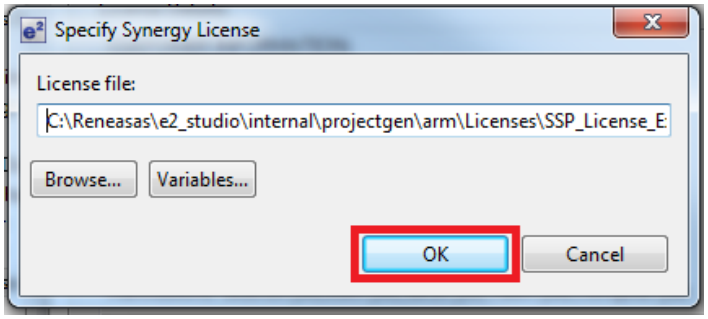

Figure 9 Confirm License File

G. Click <Apply> and then <OK> in the Preferences Dialog box.

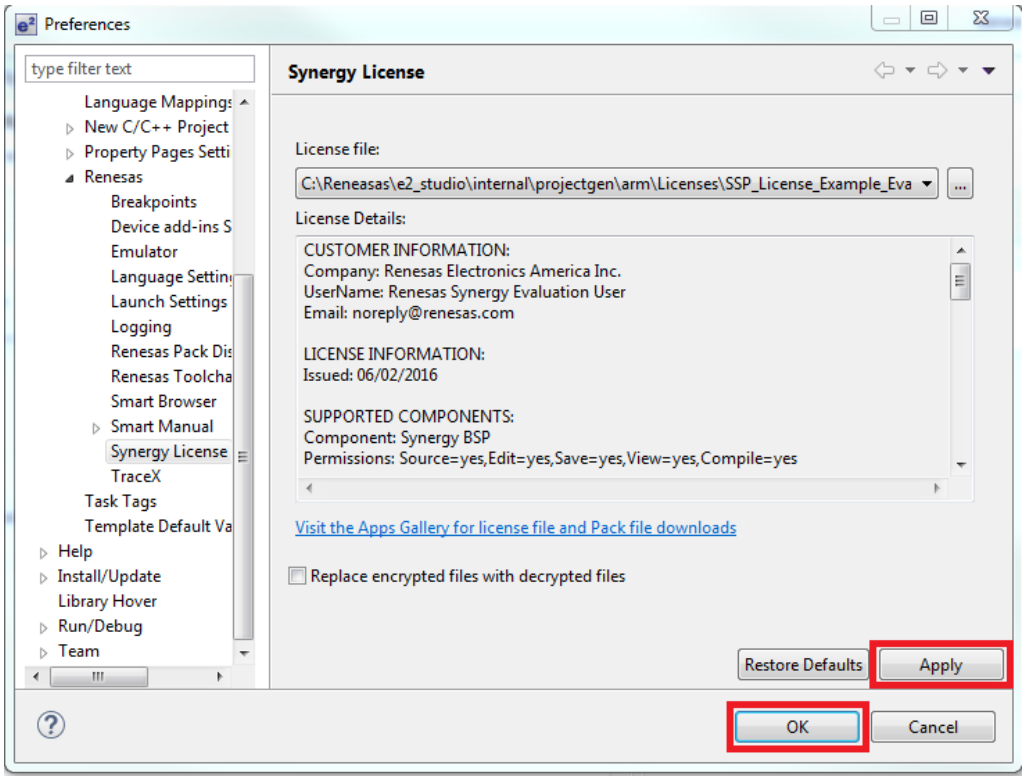

Figure 10 Apply and Confirm Synergy License File Selection

10. Enter a name for the project in the Project name text field, for example "GUIApp".

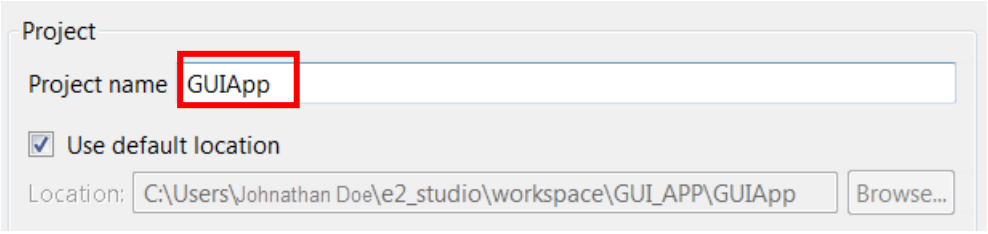

Figure 11 Enter a Project Name

11. On the top right of this page, verify that the Toolchain option is set to “GCC ARM Embedded”.

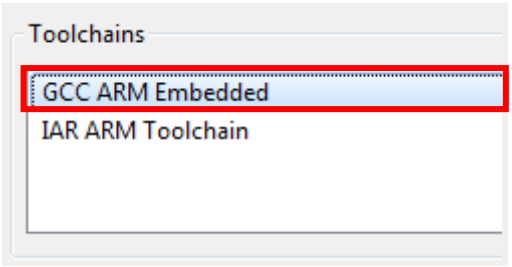

Figure 12 Verify GCC ARM Embedded Toolchain

12. Click the <Next> button to continue.
13. Under Device Selection (top left), select SSP version 1.1.0 (or later).
14. For Board, select “S7G2 DK”. The Device will be updated automatically.

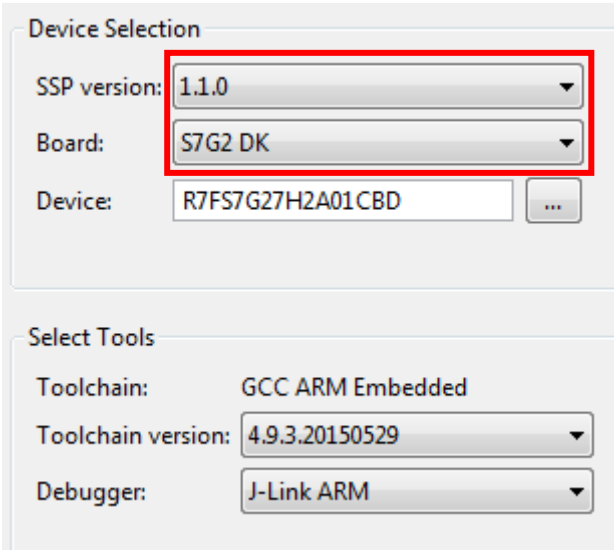

Figure 13 Device Selection

15. Click the <Next> button to continue.
16. In the Project Configuration Dialog select the option “S7G2-DK BSP”.

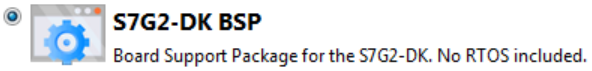

Figure 14 Select the S7G2-DK BSP

17. Click the <Finish> button.
18. If you have not directed e^2 studio to remember your perspectives, e^2 studio will display the “Open Associated Perspective?” dialog box. If opened, click <Yes> to acknowledge and close.

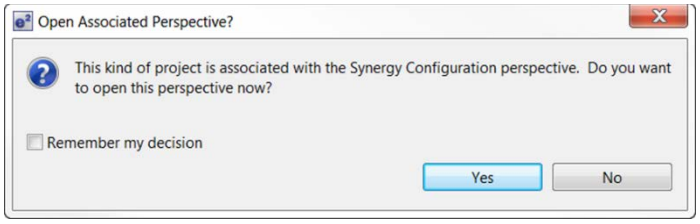

Figure 15 Open Perspective Dialog Box

When e² studio finished creating the project, you will see the screen in Figure 16.

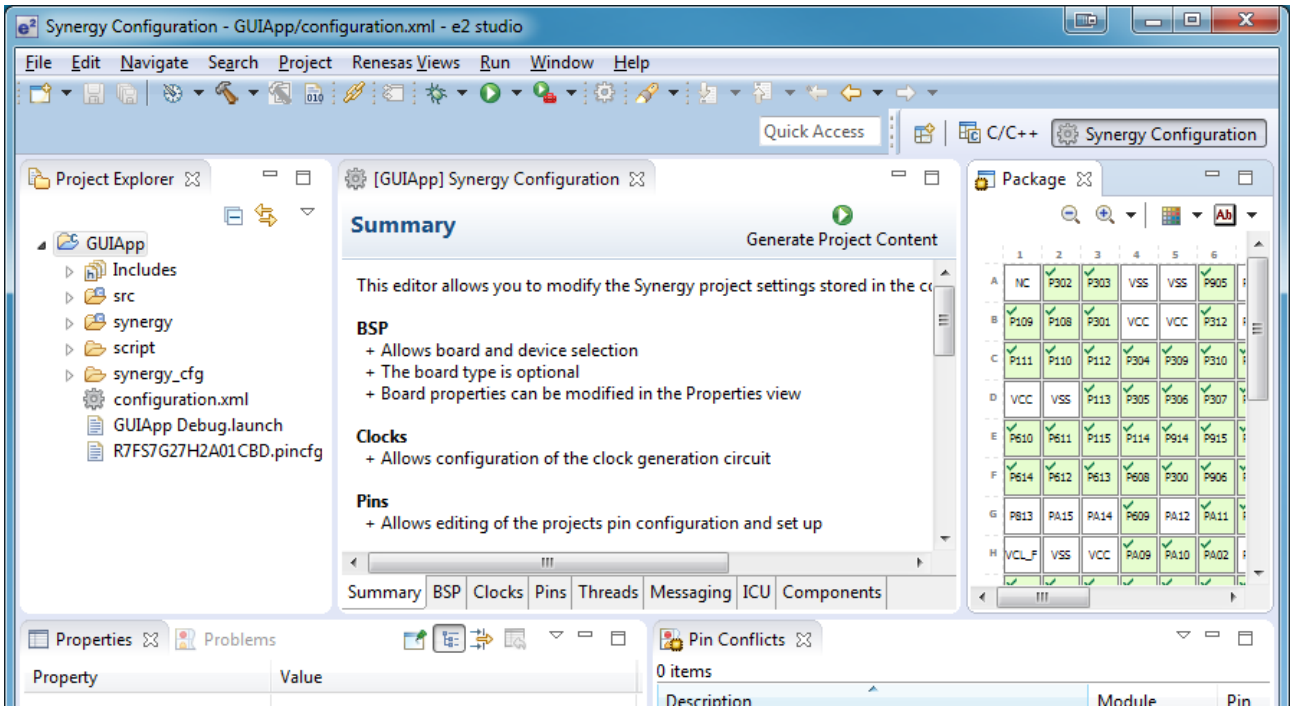

Figure 16 GUIApp Project

4. Configure the Project in the ISDE

Now that the project has been successfully created in section the Project in e² studio ISDE (Integrated Solution Development Environment), we can start configuring the project for our GUI application.e can start configuring the project for our GUI application.

1. Open the Synergy Configuration, if not already open, by double clicking the “configuration.xml” file in the Project Explorer Window.

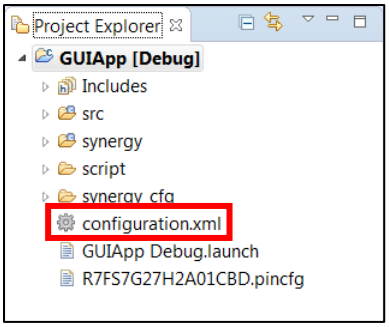

Figure 17 Selecting the configuration.xml file in Project Explorer

2. Click the <BSP> tab of the Synergy Configuration Window.

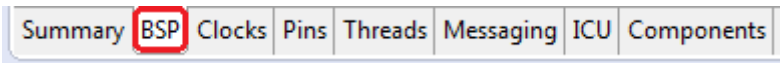

Figure 18 Synergy Configuration BSP Tab

3. Now look in the Properties Window (bottom left window) and set the “RTOS being used” to “ThreadX”. (Helpful hint: After you set ThreadX, click on another area of the properties and verify that ThreadX is still set.)

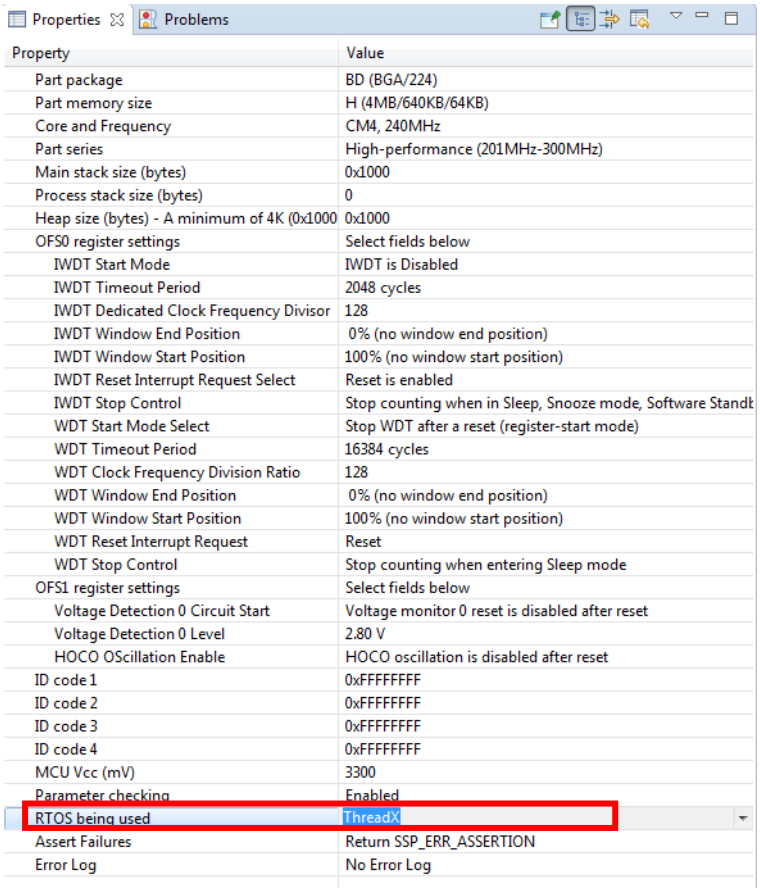

Figure 19 Setting the BSP to use the ThreadX RTOS

4. In the Synergy Configuration Window, click the <Threads> tab.

Figure 20 Synergy Configuration Threads Tab

5. Select the HAL/Common thread (on the top left).

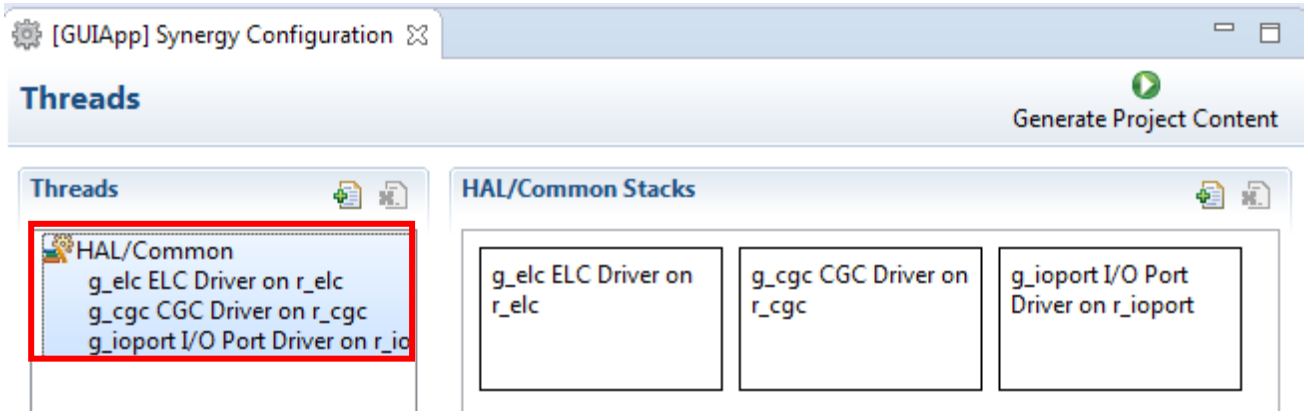

Figure 21 Threads

6. In the HAL/Common Stacks area, click the <New> Button.

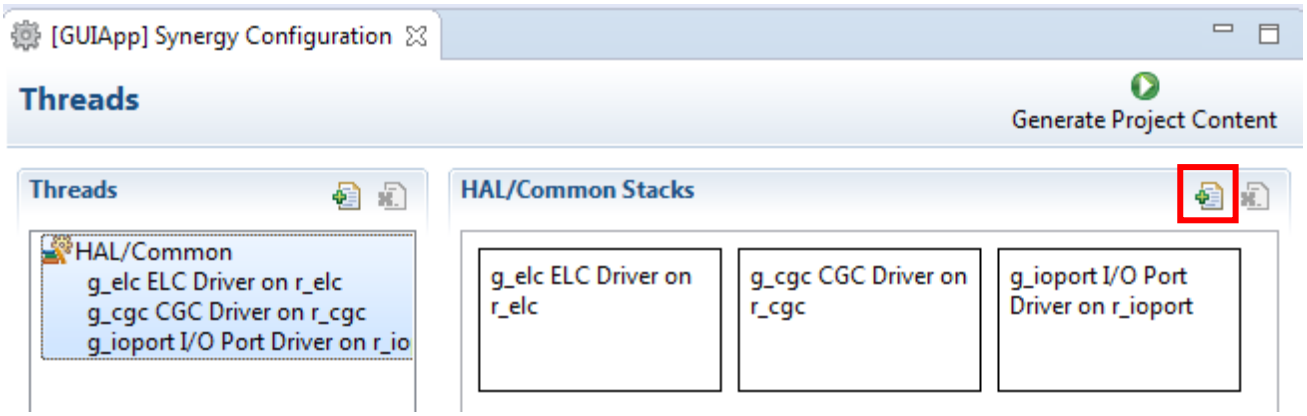

Figure 22 Add a Timer Driver Module to the HAL/Common Thread Part 1

7. In the Menu select “Driver -> Timers -> Timer Driver on r_gpt”.

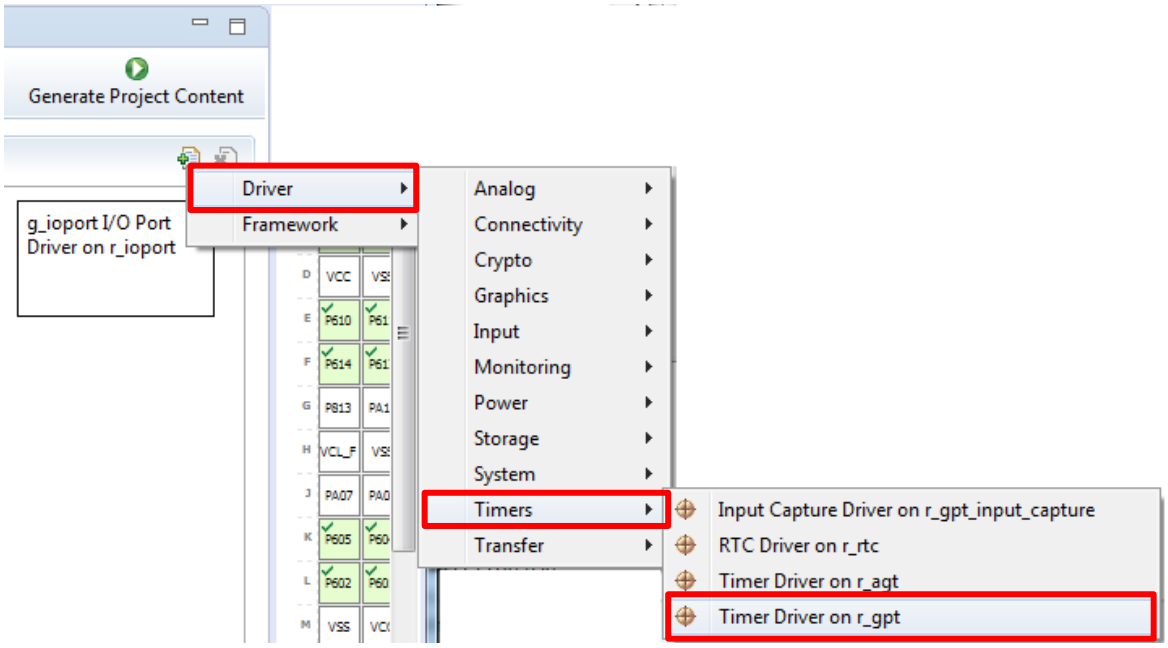

Figure 23 Add a Timer Driver Module to the HAL/Common Thread Part 2

8. In the HAL/Common Modules area, select the newly created module “g_timer0 Timer Driver on r_gpt”.

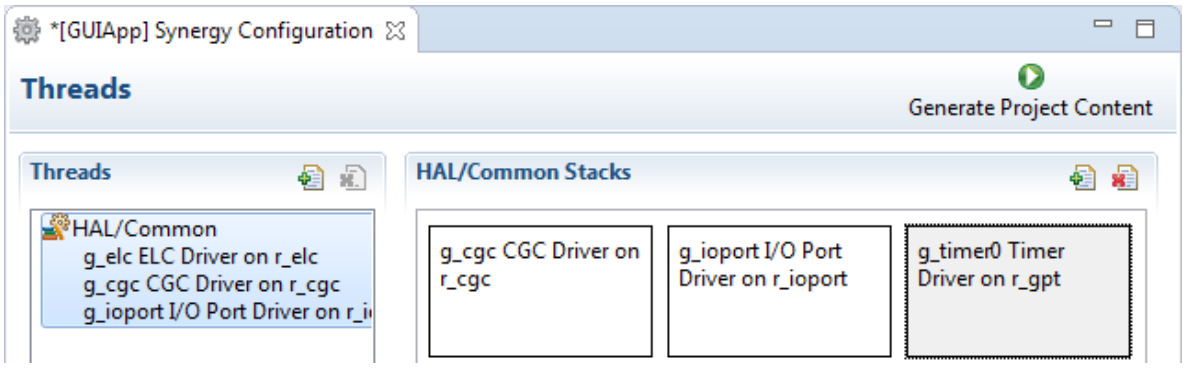

Figure 24 Select the Newly Created Timer Driver Module

9. In the Properties Window change the properties to match those below. Hint: Change the channel to 2 first!

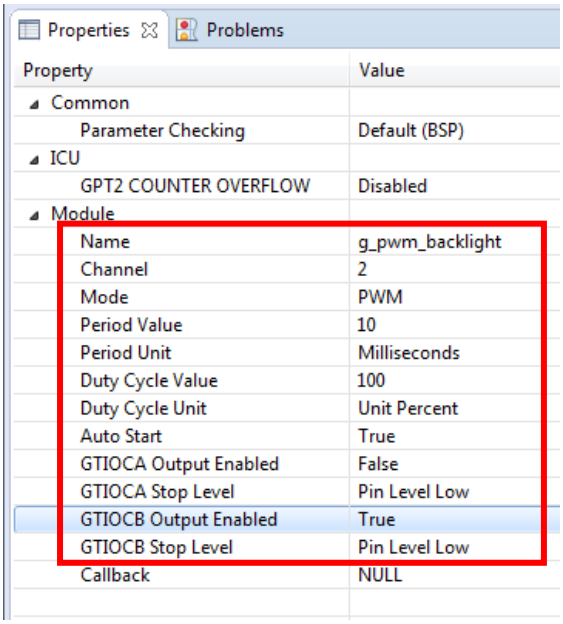

Figure 25 Configure the PWM Module

In the next steps, we will add the required software to enable the touch screen and configure the LCD driver.

The touch screen requires several frameworks and drivers to be used. We need external interrupts to know when to read the data, an I2C driver to handle the reads, and a framework to translate the register data from the peripheral to touch coordinates the software can use.

10. Create a new thread by clicking <New> in the Threads area.

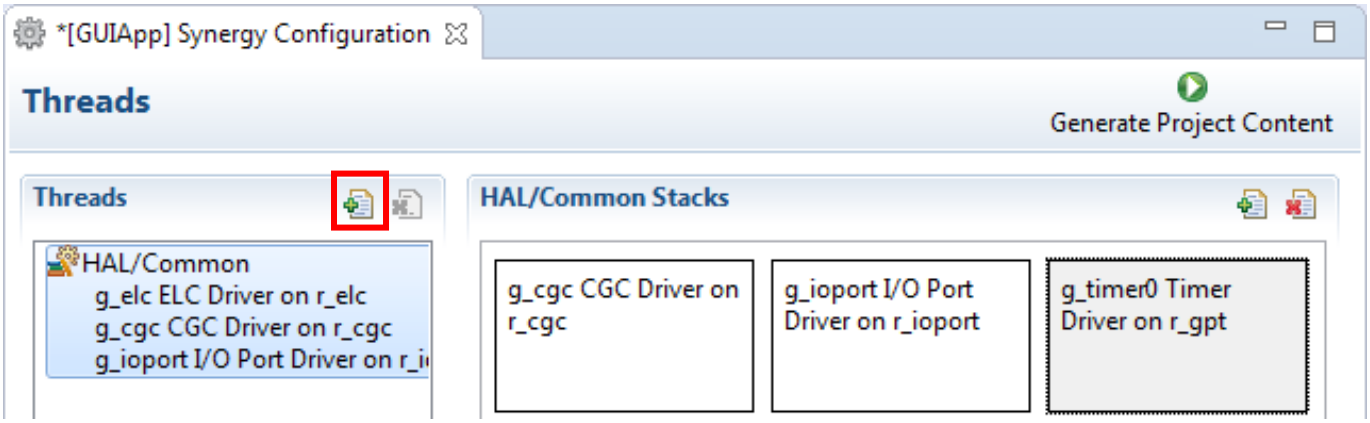

Figure 26 Create a New Thread

11. Click on “New Thread” to pull up the properties.
12. Edit the Properties to match Figure 27.

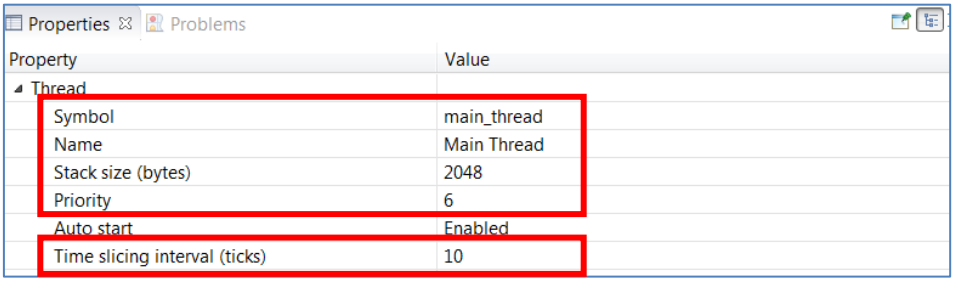

Figure 27 Configure Main Thread Properties

13. Back in the Synergy Configuration Window, Threads tab, Main Thread Stacks area, click on <New >.

Note:
Be sure Main Thread is selected before adding new modules.

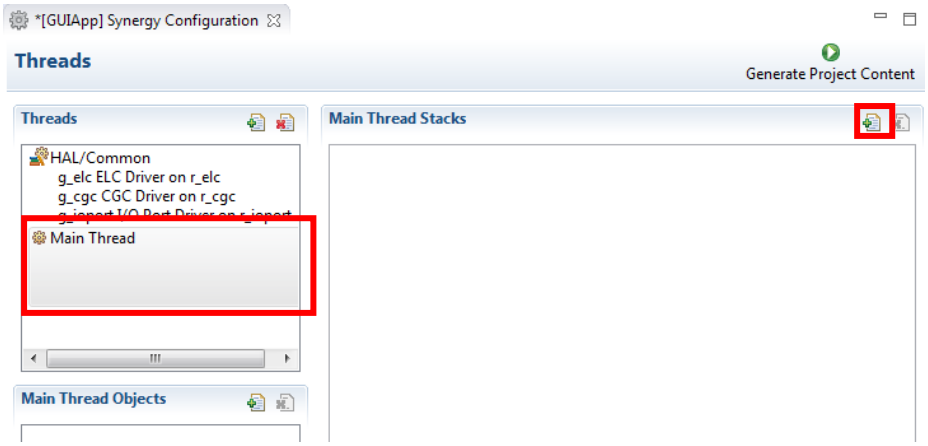

Figure 28 Main Thread Stacks

14. Add a framework for the touch panel by selecting <New> “Framework -> Input -> Touch Panel Framework on sf_touch_panel_i2c”.

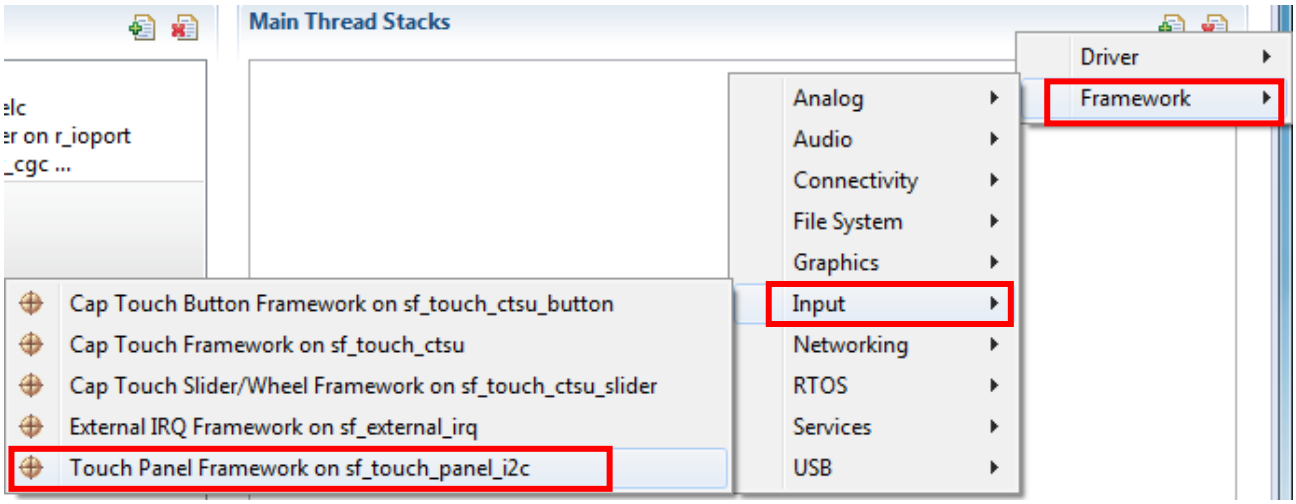

Figure 29 Adding Touch Panel Framework

15. Configure the properties as in Figure 30.

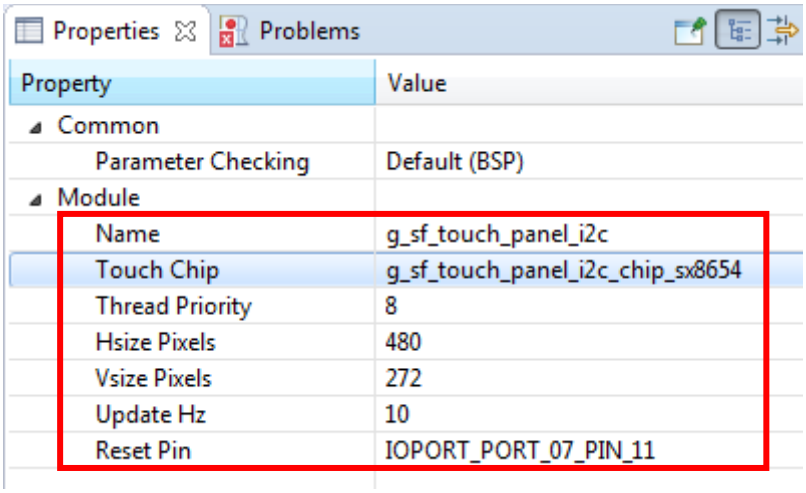

Figure 30 Configuring Touch Panel Properties

16. Notice that the Synergy Configurator has now already created the message framework, external IRQ framework and driver, and has a placeholder for the I2C driver as shown below.

    The messaging framework is used by other framework layers and tasks to pass messages around the system. This system will be used to pass data from the touch screen driver to the Main Task to handle touch inputs.

    The SF External Interrupt is a framework layer used by the touch controller driver as in Figure 31.

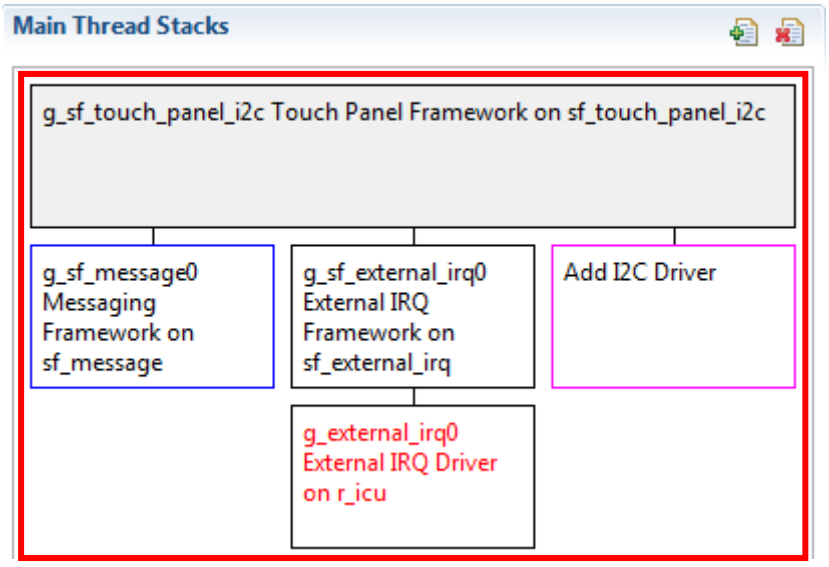

Figure 31 Touch Panel Framework Stack

17. Select the “External IRQ Framework on sf_external_irq” and configure the properties as in Figure 32.

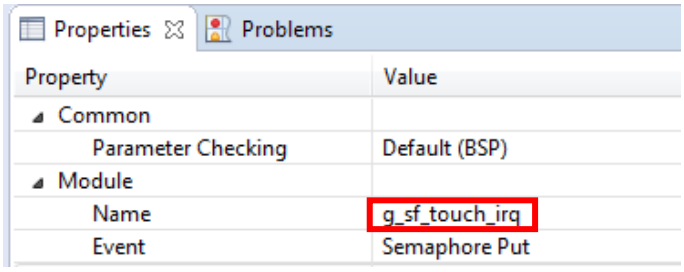

Figure 32 Configuring External Interrupts Properties

18. Select “External IRQ Driver on r_icu”. Configure the properties for the new module as in Figure 33. Hint: Change the “Channel” first!

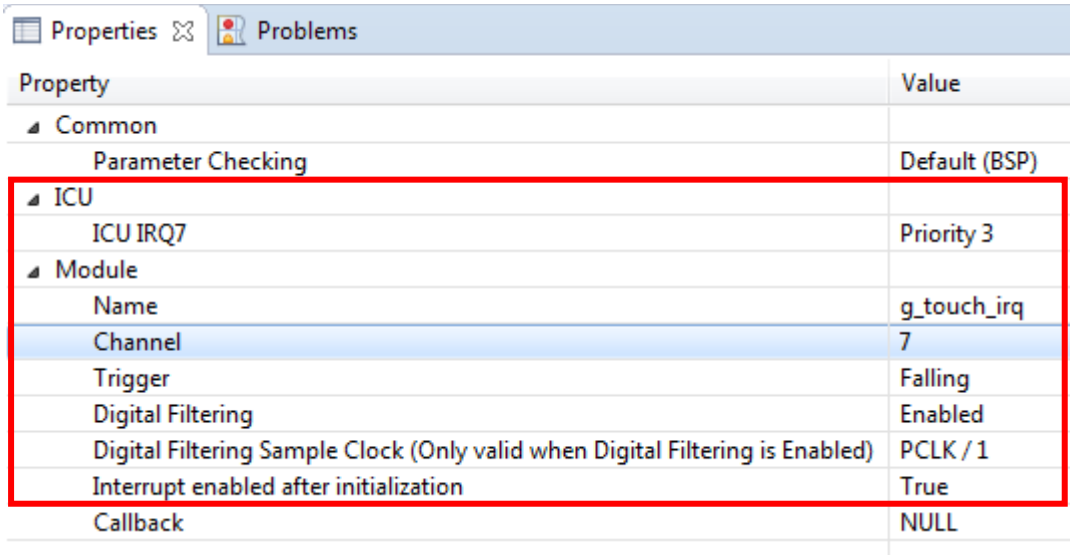

Figure 33 Touch Screen IRQ Properties

19. In the Synergy Configuration Window, Threads tab, Main Thread Stacks area, add a driver for the I2C bus by rightclicking- on the “Add I2C Driver” then selecting ”New -> I2C Driver on r_sci_i2c”.

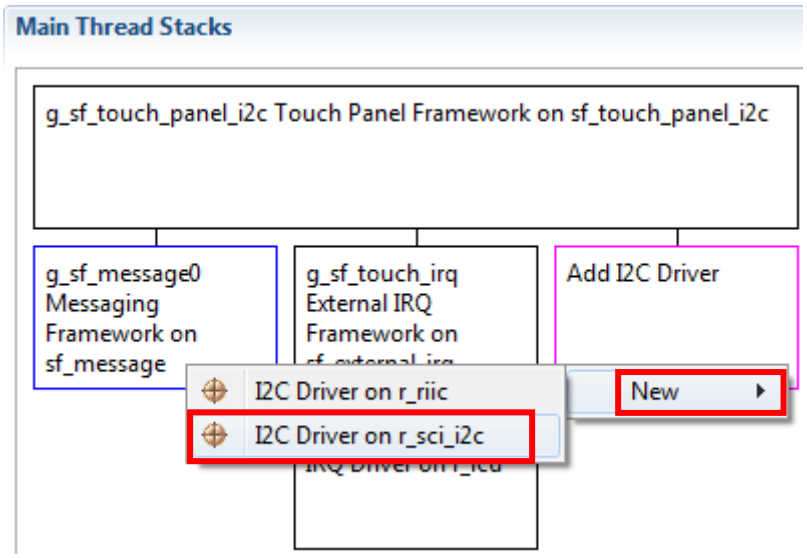

Figure 34 I2C Driver on SCI

20. After the previous selection, “SCI Common” will be added to the stack. Configure the properties for SCI Common as in Figure 35 by enabling only the “Simple I2C Mode”.

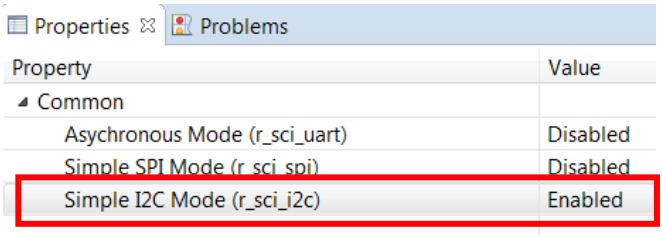

Figure 35 SCI Common Properties

21. Configure the properties for I2C Driver on r_sci_i2c as in Figure 36. **Hint: Change the "Channel" first!**

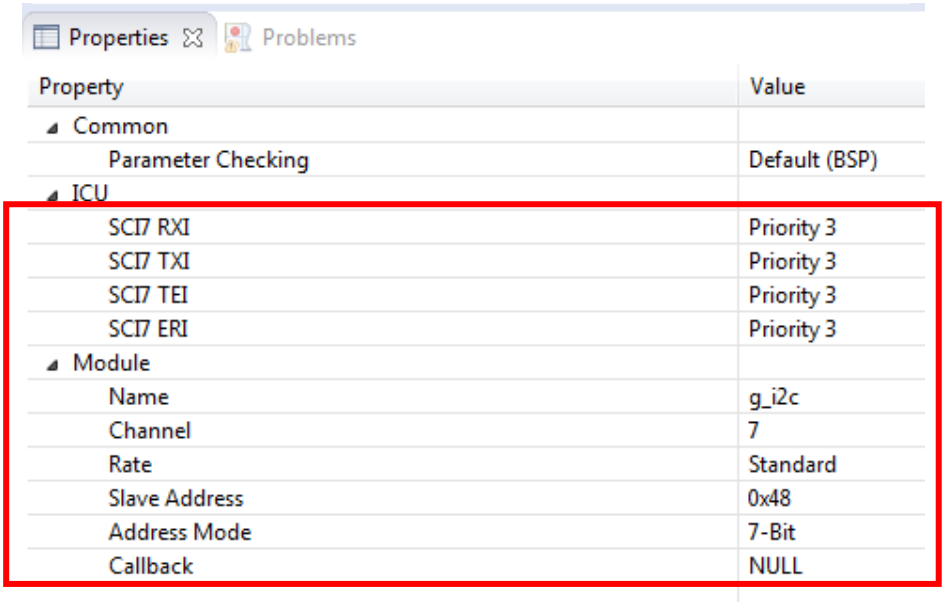

Figure 36 I2C on SCI Properties

22. Under Main Thread Stacks, select <New> "Framework->Graphics->GUIX on gx".

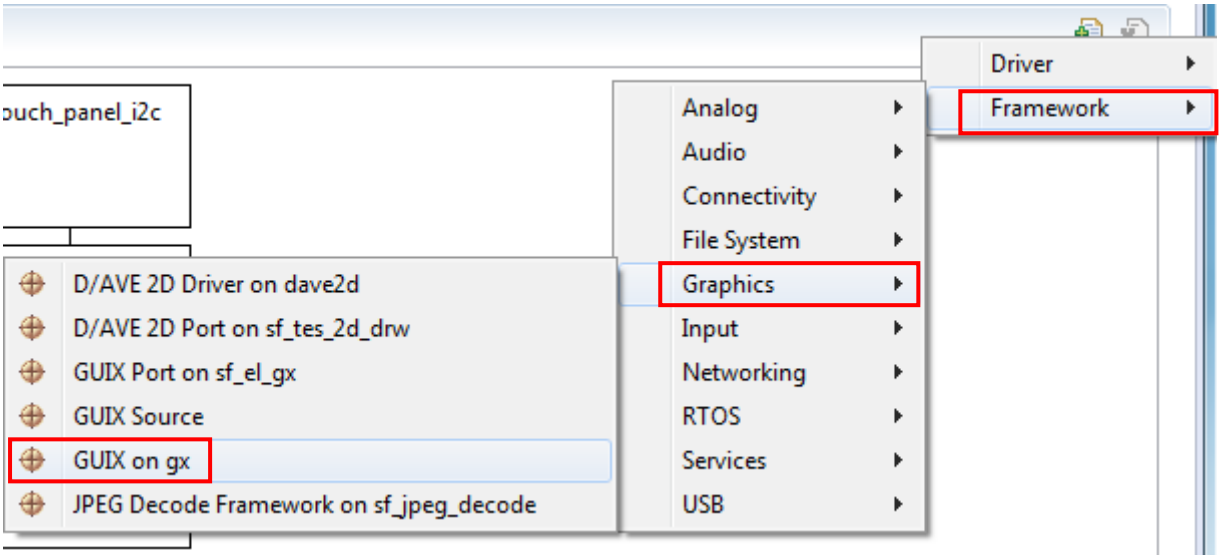

Figure 37 GUIX on gx

Notice that the Synergy Configurator has now already created the “GUIX Port on sf_el_gx” framework, Display Driver, JPEG decoder, and D/AVE hardware accelerator drivers as in Figure 38.

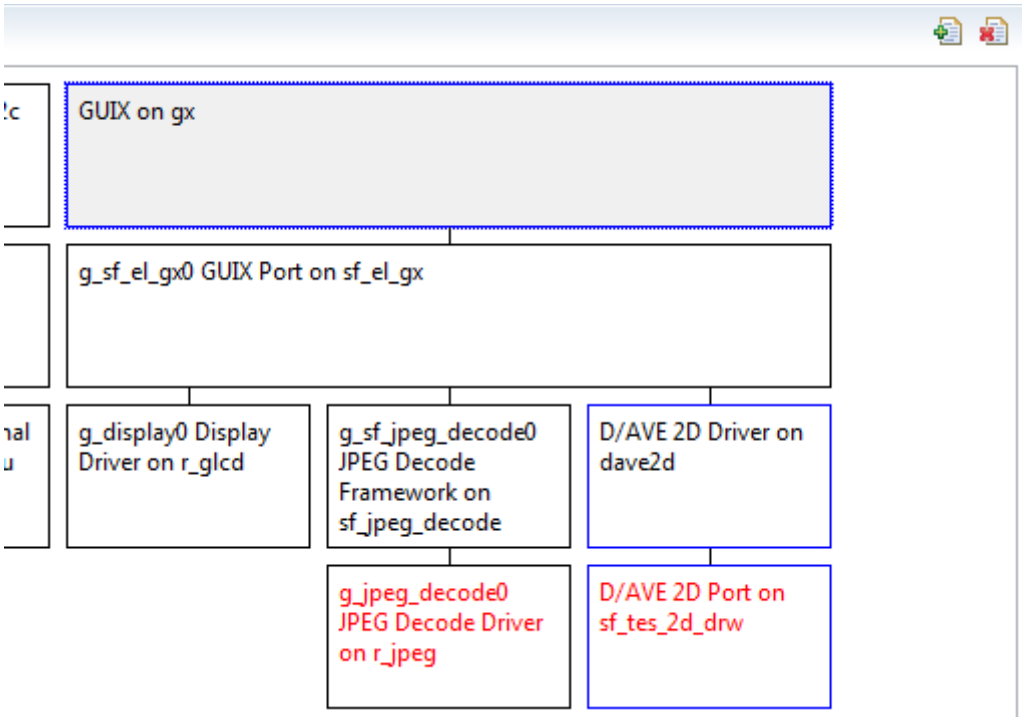

Figure 38 GUIX on gx

23. Select “GUIX on gx” and configure the Properties as in Figure 39.

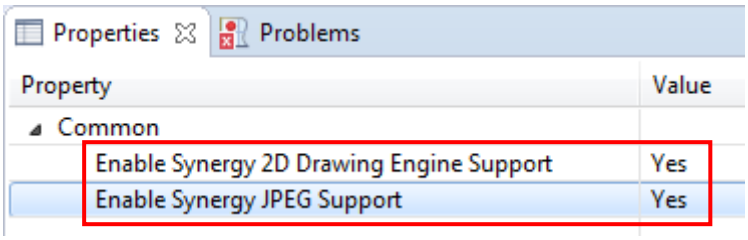

Figure 39 GUIX on gx Properties

24. Select “GUIX Port on sf_el_gx” and configure the Properties as in Figure 40.

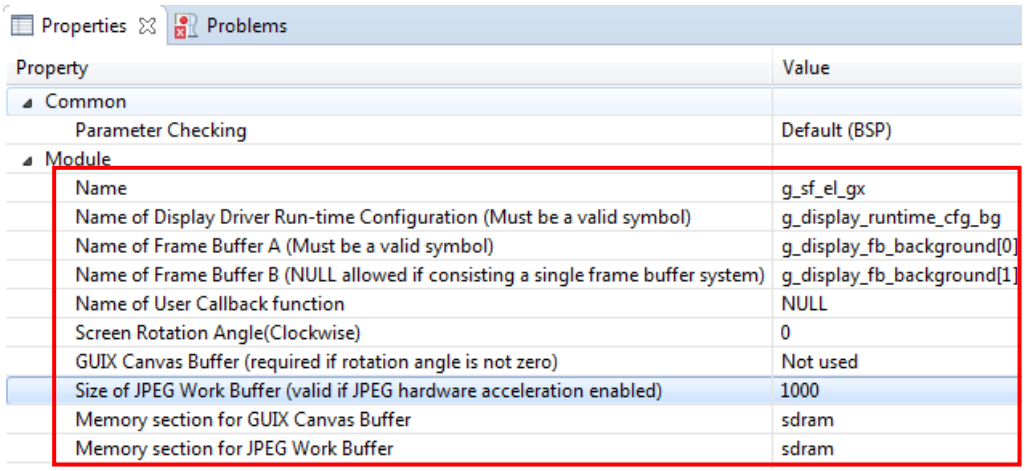

Figure 40 GUIX Port on sf_el_gx Properties

25. Select the “JPEG Decode Driver on r_jpeg” and configure the interrupt properties as in Figure 41. Note that Priority 3 is just an arbitrary number.

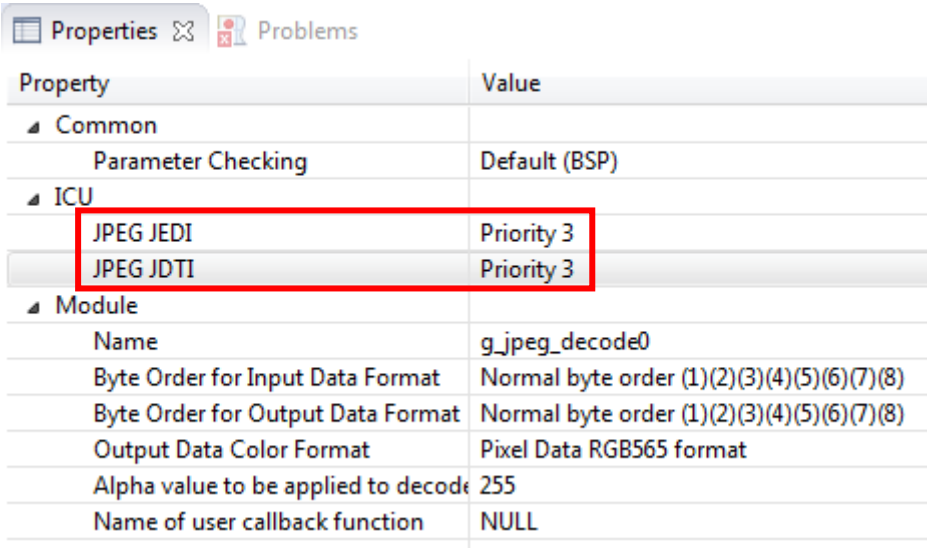

Figure 41 JPEG Decode Driver on r_jpeg Properties

26. Under Main Thread Stacks, select “D/AVE 2D Port on sf_tes_2d_drw” and configure the properties as in Figure 42.

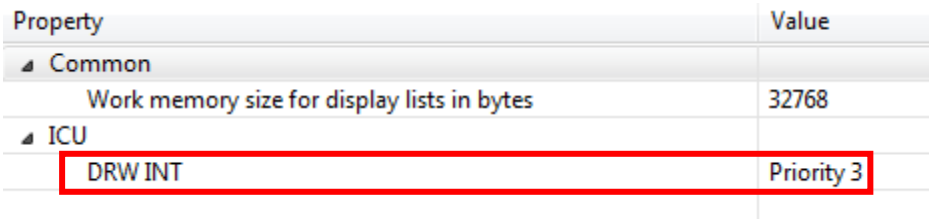

Figure 42 D/AVE 2D Port Properties

27. Under Main Thread Stacks, Select “Display Driver on r_glcd” and configure the Interrupt Properties as in Figure 43.

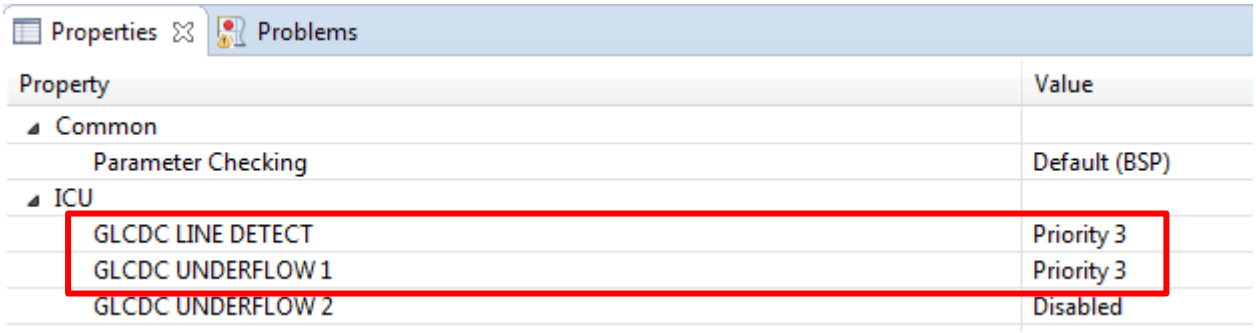

Figure 43 Interrupt Properties

28. Configure the Graphics Screen 1 Properties as in Figure 44.

Module	
Name	g_display
Name of display callback function to be defined by user	NULL
Input - Panel clock source select	Internal clock(GLCDCLK)
Input - Graphics screen1	Used
Input - Graphics screen1 frame buffer name	fb_background
Input - Number of Graphics screen1 frame buffer	2
Input - Section where Graphics screen1 frame buffer allocated	sdram
Input - Graphics screen1 input horizontal size	480
Input - Graphics screen1 input vertical size	272
Input - Graphics screen1 input horizontal stride(not bytes but pixels)	480
Input - Graphics screen1 input format	16bits RGB565
Input - Graphics screen1 input line descending	Not used
Input - Graphics screen1 input lines repeat	Off
Input - Graphics screen1 input lines repeat times	0
Input - Graphics screen1 layer coordinate X	0
Input - Graphics screen1 layer coordinate Y	0
Input - Graphics screen1 layer background color alpha	255
Input - Graphics screen1 layer background color Red	255
Input - Graphics screen1 layer background color Green	255
Input - Graphics screen1 layer background color Blue	255
Input - Graphics screen1 layer fading control	None
Input - Graphics screen1 layer fade speed	0

Figure 44 Graphics Screen 1 Properties

29. Configure the Output properties as in Figure 45.

Output - Horizontal total cycles	582
Output - Horizontal active video cycles	480
Output - Horizontal back porch cycles	43
Output - Horizontal sync signal cycles	41
Output - Horizontal sync signal polarity	Low active
Output - Vertical total lines	286
Output - Vertical active video lines	272
Output - Vertical back porch lines	12
Output - Vertical sync signal lines	10
Output - Vertical sync signal polarity	Low active
Output - Format	16bits RGB565
Output - Endian	Little endian
Output - Color order	RGB
Output - Data Enable Signal Polarity	High active
Output - Sync edge	Rising edge
Output - Background color alpha channel	255
Output - Background color R channel	0
Output - Background color G channel	0
Output - Background color B channel	0

Figure 45 Output Screen 2 Properties

30. Configure the TCON pins and clock as in Figure 46.

TCON - Hsync pin select	LCD_TCON1
TCON - Vsync pin select	LCD_TCON2
TCON - DataEnable pin select	LCD_TCON0
TCON - Panel clock division ratio	1/16

Figure 46 TCON Settings

31. Save the project by pressing “Ctrl + s” on the keyboard.
32. Click the <Generate Project Content> button to update the project files.

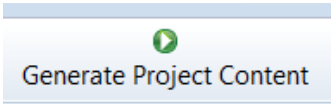

Figure 47 Generate Project Contnet

33. Close the Synergy Configuration window.
34. Open Windows Explorer and go to the directory where you put the files included with this application note. Locate the file “Source Files\ R7FS7G27H2A01CBD.pincfg”. Now drag the file from the Windows Explorer Window into the “GUIApp” root directory inside the e^2 studio Project Explorer window.
    A. When asked how to import the selected files, click <OK> to copy the files.
    B. When asked if you want to overwrite, click <Yes>.

NOTE:

This file contains the pin configuration for the DK-S7G2.

35. Reopen the Synergy Configuration by double clicking the “configuration.xml” file in Project Explorer.

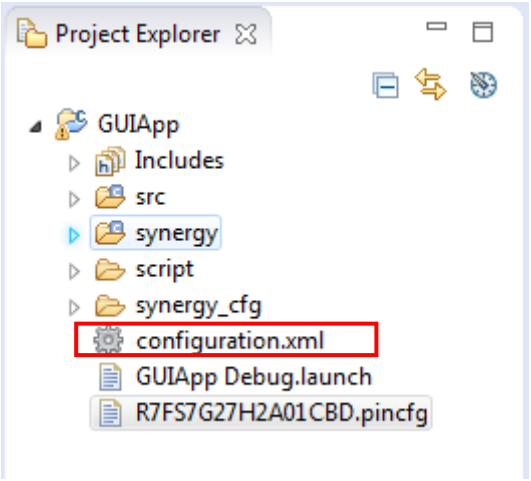

Figure 48 Synergy Configuration

In the next steps, we will show how to configure pins of the S7G2 to control LCD panel and touch screen on DK-S7G2. Proceed to Step 33 to skip this optional informational section.

The “Timer Driver on r_gpt” will be used to configure the peripheral as a PWM to control the backlight level using a hardware pin on the S7G2. For the DK-S7G2, the pin that controls the backlight for the LCD is located on P7_12, as we can see from the snippet of the DK-S7G2 breakout board schematic in Figure 49.

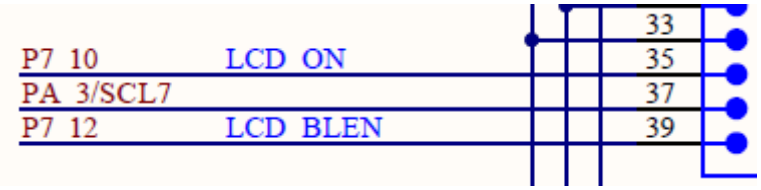

Figure 49 LCD Backlight Pin

Since we are using an existing pin configuration, it is not necessary to set this pin up using the pin configurator. However, if you are interested you can follow the steps below to see how it was configured.

1. Select the <Pins> tab on the Synergy Configuration Window.

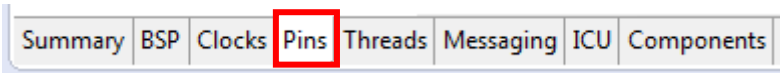

Figure 50 Configuration Pins

2. Expand “Ports” and “P7” to show the port 7 pins.

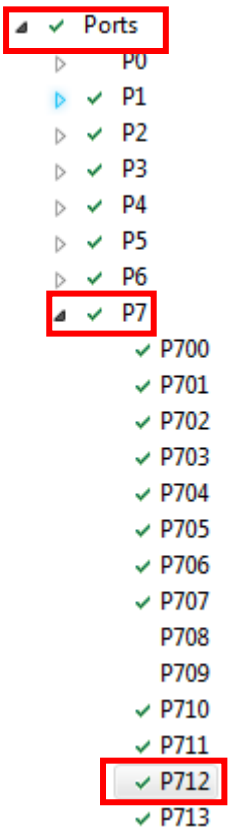

Figure 51 Port 7 Pins

3. Select “P712” to show the options for this pin.

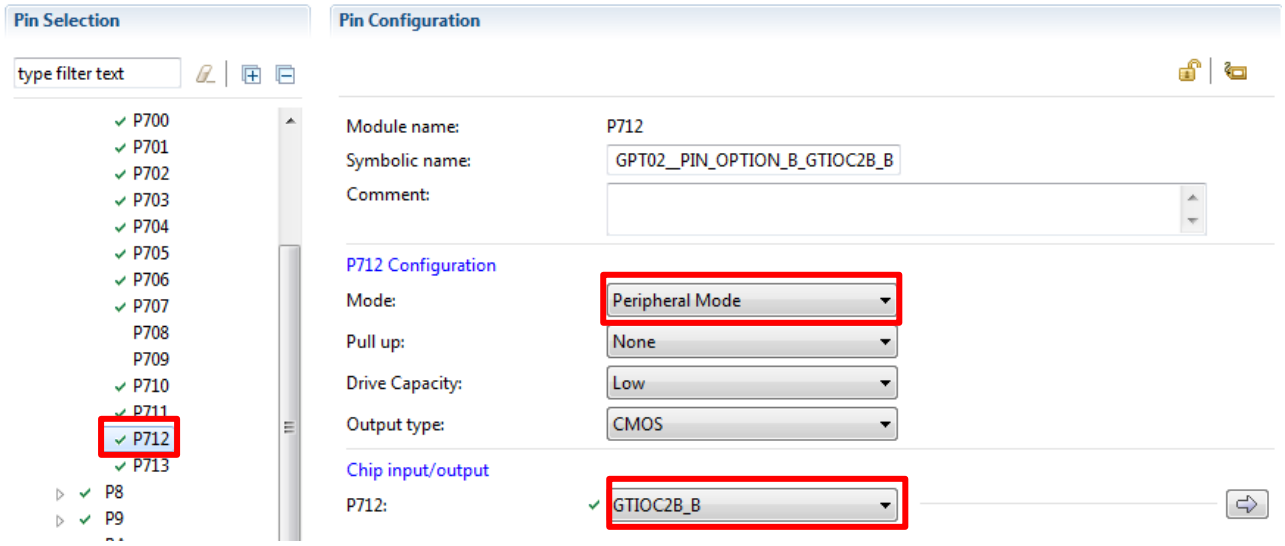

Figure 52 Pin Configuration for P7_12

- Module name: Synergy defined port and pin name
- Symbolic name: Optional symbolic name for code reference
- Comments: Optional description for the pin
- Mode: Pin’s function; Input Mode, Output Mode, Peripheral Mode
- Pull up: Internal resistor pullup; None, input pull-up
- Drive Capacity: Output drive capability; Low, Medium, High
- Output type: CMOS, n-ch open drain
- P712: This option changes based on the mode setting. In this case, GTIOC2B_B is selected to use as the Timer 2 B output.

Pins can also be configured using the peripheral as a starting point.

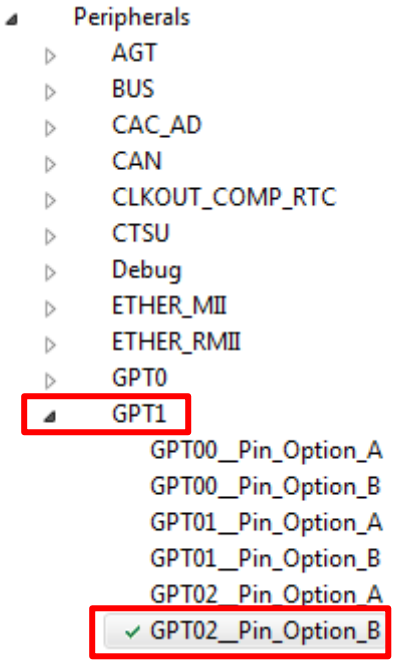

Figure 53 GPT02 Pin Option B

This view will show which pins are available for particular functions.

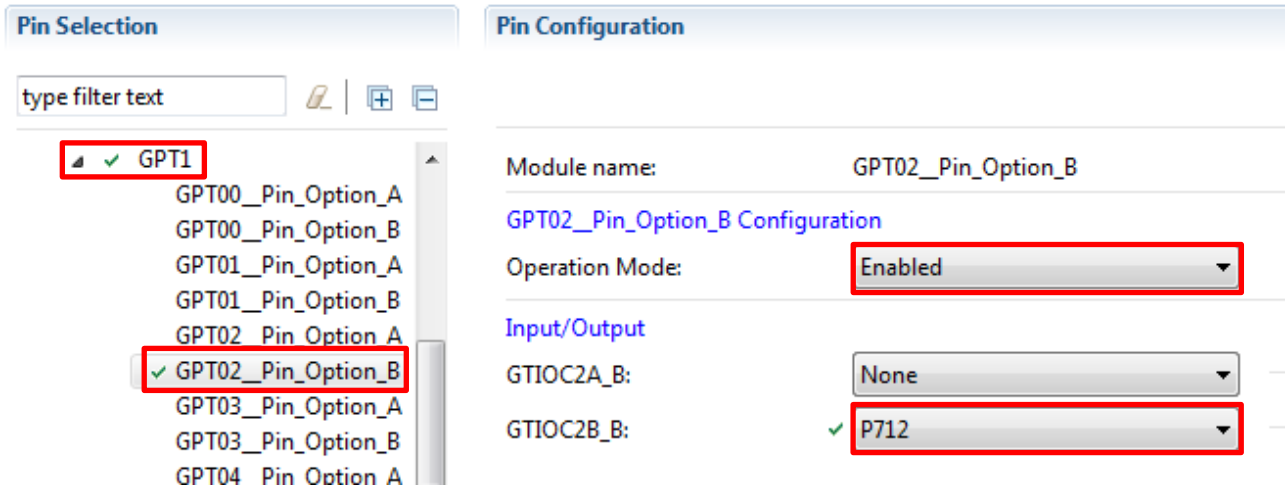

Figure 54 GPT02 Pin Configuration

In most cases, after you enable and select a pin, it will be automatically configured. This can be configured by pressing the , which will show a screen similar to Figure 54.

The interrupt for the touch controller is located on pin “P0_1” as seen in the breakout board schematic.

Figure 55 Touch IRQ

The touch panel pin is configured as an IRQ in the Pin Configuration window.

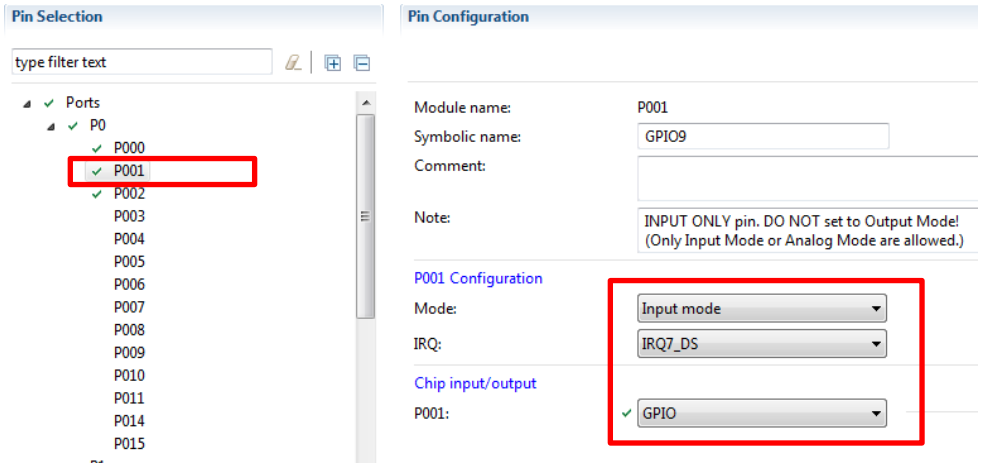

Figure 56 P0_01 Pin Configuration

For the LCD board schematic we can see that the touch controller is the SX8656.

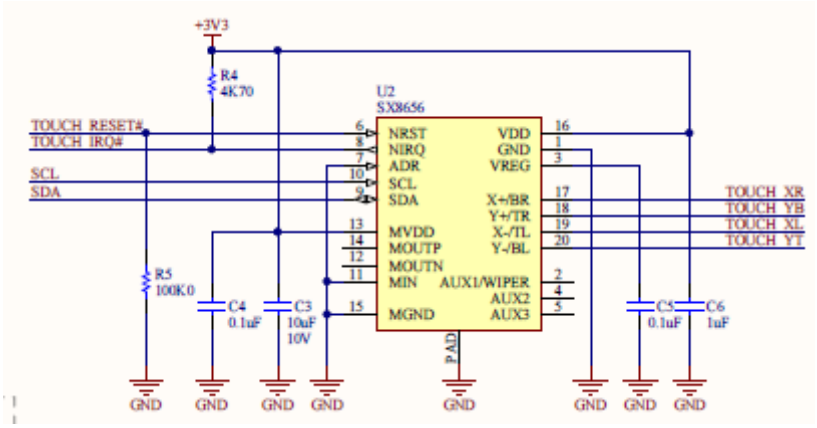

Figure 57 Touch Controller

As you can see from Figure 57, the touch controller's reset pin is located on P07_11. To make use of this function, the pin is setup as a GPIO output.

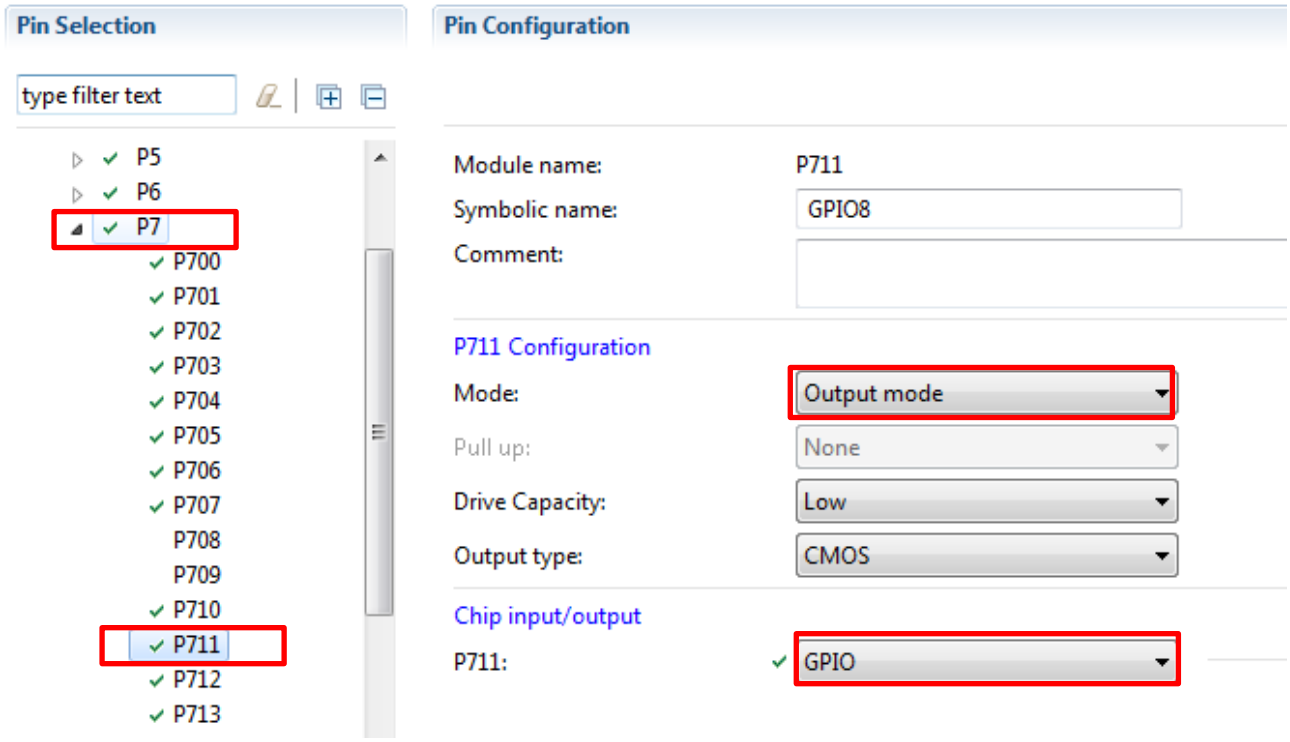

Figure 58 P7_11 Pin Configuration

The SCI driver can be configured for different serial communication protocols for the DK-S7G2 pin PA_3 and PA_2 are used to handle the I2C functionality.

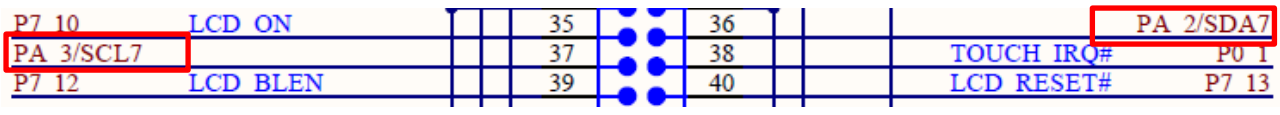

Figure 59 I2C Pins

The pins are configured in the Pin Configurator under the peripheral section.

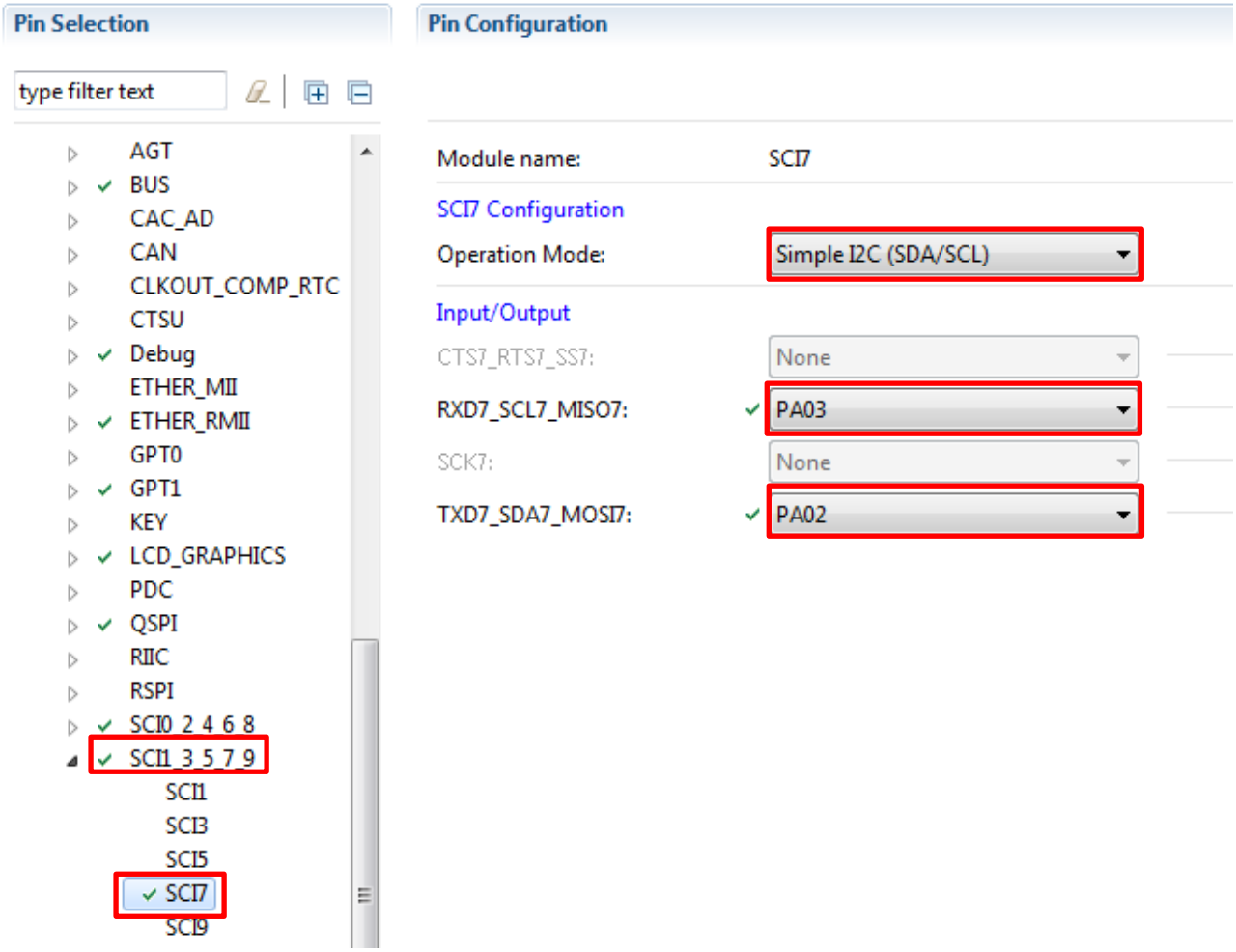

Figure 60 SCI7 Configuration

The LCD pin configuration is based on the B option for the g_lcd controller as seen in the pin configurator.

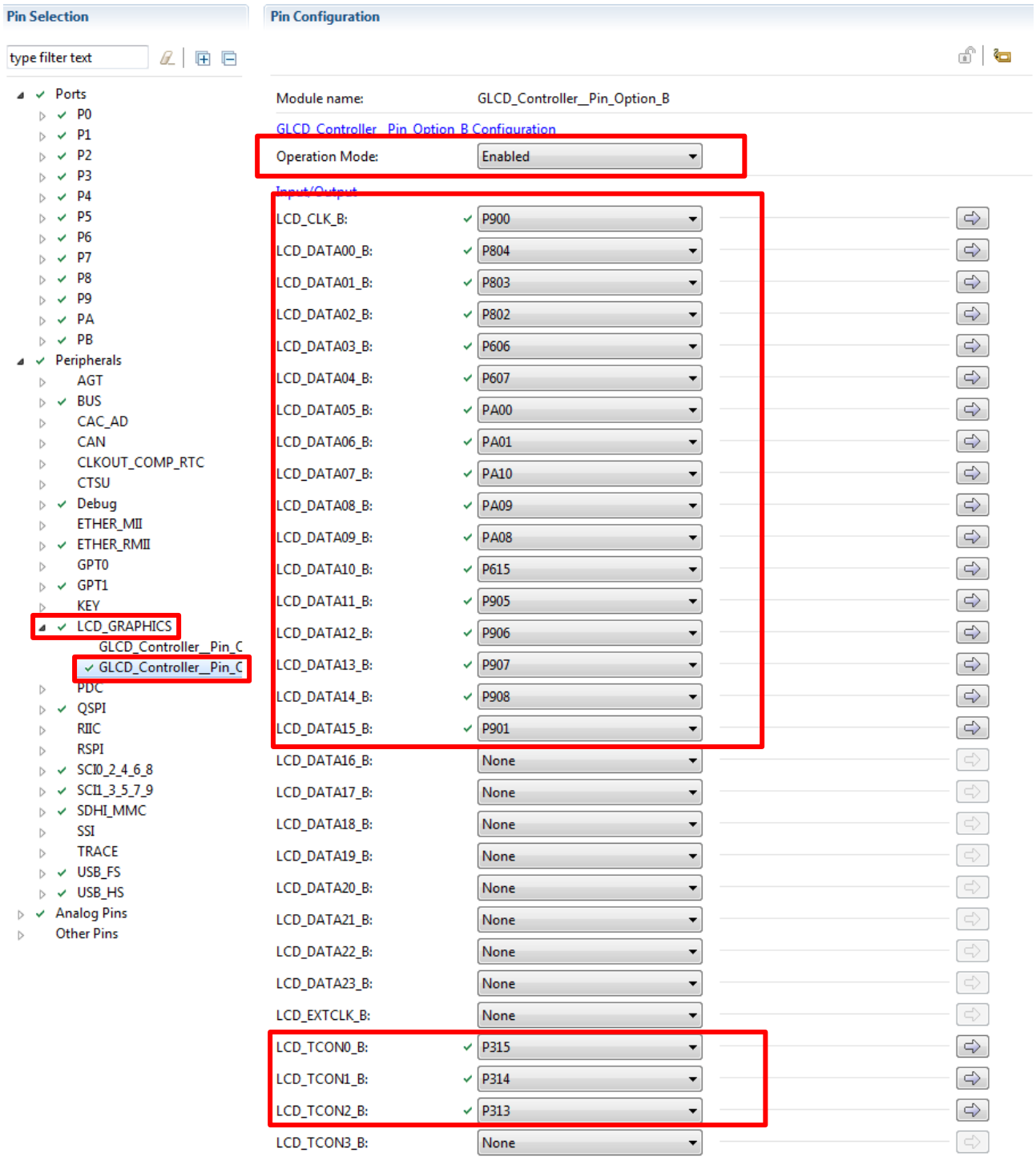

Figure 61 GLCD Pin Option B

The breakout board schematic shows the full list of pins. The DK-D7G2 uses a 16-bit LCD interface.

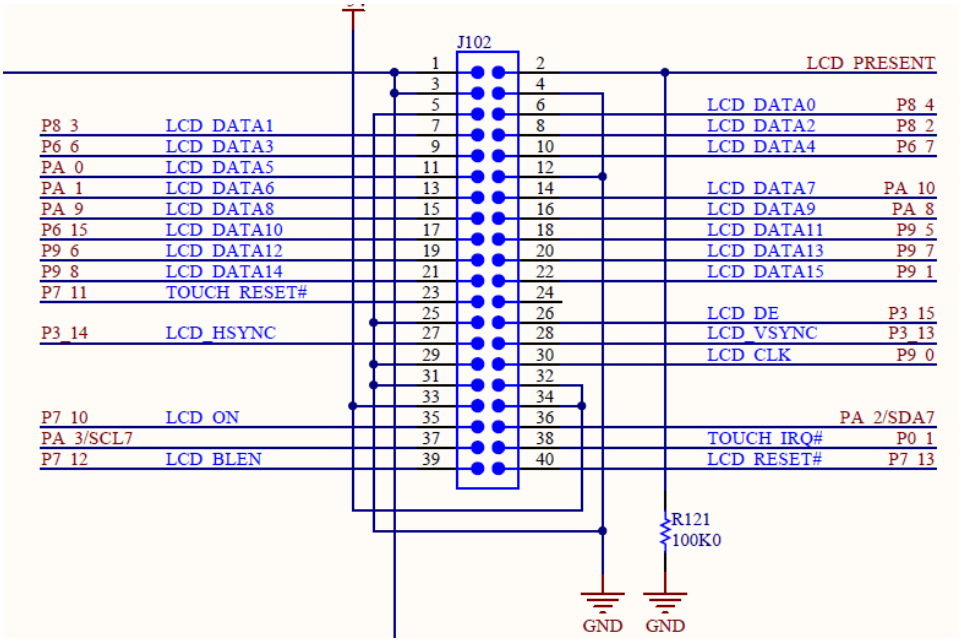

Figure 62 LCD Connector Pin Out

36. Select the “Messaging” tab on the Synergy Configuration Window. The window in Figure 63 will be shown.

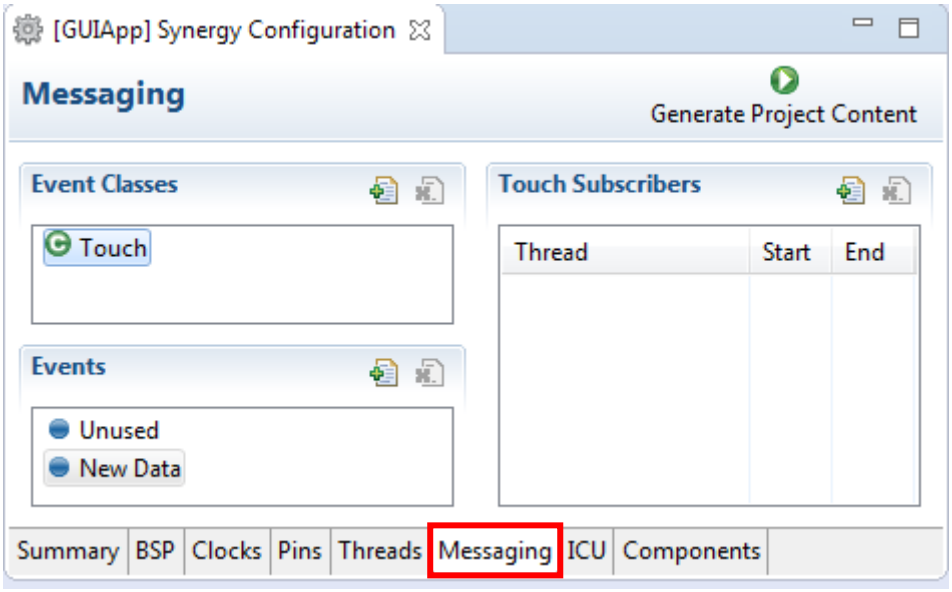

Figure 63 Messaging Tab

NOTE:
This tab configures the event class definitions for the touchscreen events along with the event queue initialization and linking variables. The touch event was automatically generated when “Touch Panel Framework on sf_touch_panel_i2c” was added in the threads menu.

37. Select the Touch Event class.
38. On the touch subscribers menu, click the new button.

39. Inside of the “New Subscriber” Dialog, select the main thread.

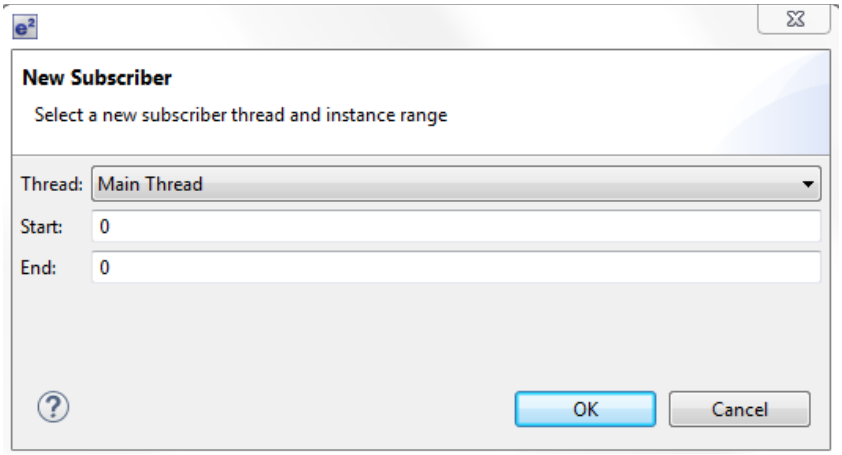

Figure 64 New Subscriber dialog

40. Click the <OK> button.
41. Save the project by pressing “Ctrl + s” on the keyboard.
42. Click the <Generate Project Content> button to update the project files.

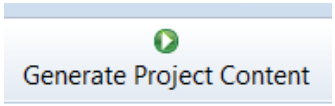

Figure 65 Generate Project Content

43. Open Windows Explorer and surf to where you put the files included with this application note. Locate the file “Source Files**main_thread_entry.c**”. Now drag the file from the Windows Explorer Window into the “src” folder inside the e^2 studio Project Explorer window.
    A. When asked how to import the selected files, click <OK> to copy the files.
    B. When asked if you want to overwrite, click <Yes>.

NOTE:

This file contains the Main Thread event handling code. It reads low level touchscreen events from the queue and transforms them to graphical user interface actions.

5. Create the GUIX Interface Using GUIX Studio

Now that the base project is set up, we can start adding the GUIX components.

1. Create a new folder named “gui” inside the “src” by right clicking on the “src” folder and selecting “New -> Folder”.

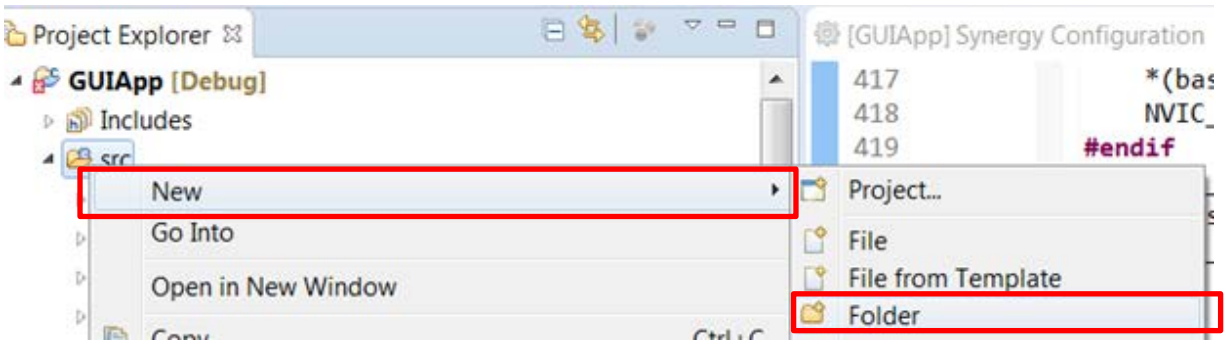

Figure 66 Creating a New Folder

2. Create another new folder named “guix_studio” in the root folder of the project by right clicking on “GUIApp” and selecting “New -> Folder”. The final folder layout should now look like Figure 67.

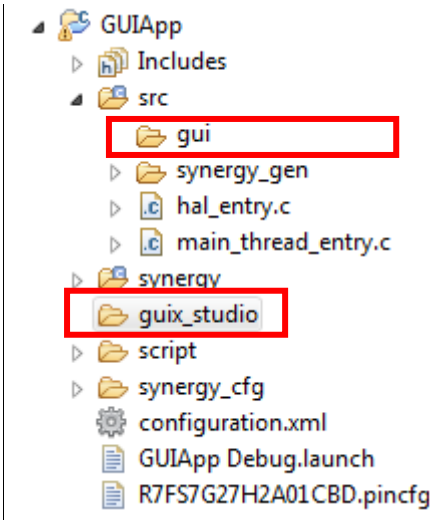

Figure 67 Final Folder List

3. Open GUIX Studio by clicking the desktop icon or by clicking on the GUIX icon in the Windows Start Menu, “All Programs -> Express Logic -> GUIX Studio 5.3” folder.

Figure 68 Start GUIX Studio

4. On the Recent Projects dialog click the button <Create New Project…>

Figure 69 Create New Project

5. Name the project “guiapp”.

WARNING:

Filenames will be generated by appending names to the project name. You must be careful to be case sensitive when you define your project name. Later, we will add files to the project that will assume you have called this GUIX project “guiapp”.

6. For the Project Path, browse to the location of the folder we created earlier called “guix_studio”.

NOTE:

If you installed the tools into the default directories, the folder will be located at “C:\Users\[User]\e2_studio\workspace\GUI_APP\GUIApp\guix_studio”

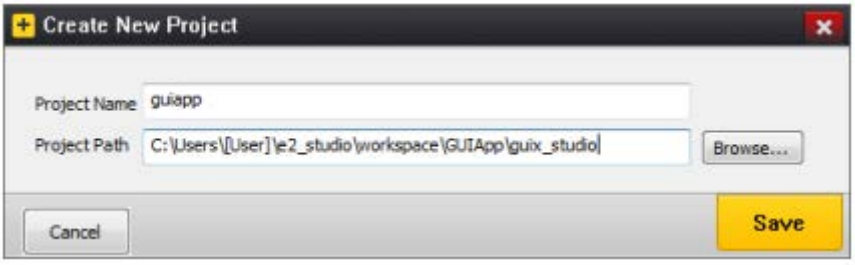

Figure 70 Create a New GUIX Project

7. Click <Save>.
8. Change the Directories for all three options to be ..\src\gui.

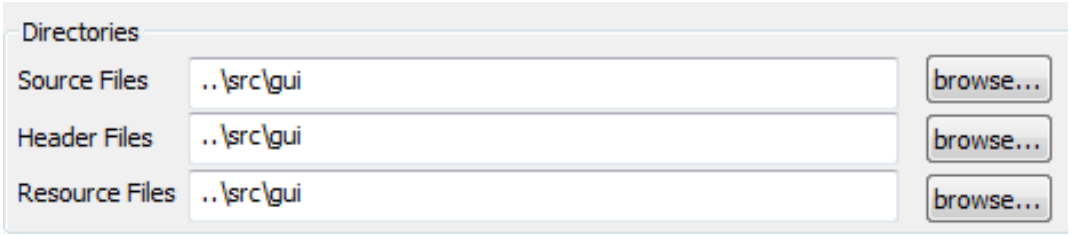

Figure 71 Correct the file locations

CAUTION:

Make sure you put in two dots ".." in the directories above.

9. Change the Target CPU Setting to "Renesas Synergy".
10. Change the Toolchain setting to "GNU".

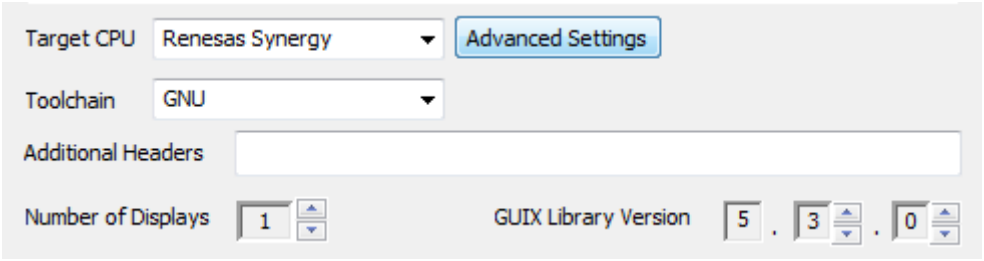

Figure 72 Target and GUIX version settings

11. Click the <Advanced Settings> button. A dialog will appear.
12. Enable the graphics accelerator and hardware JPEG decoder as shown below.

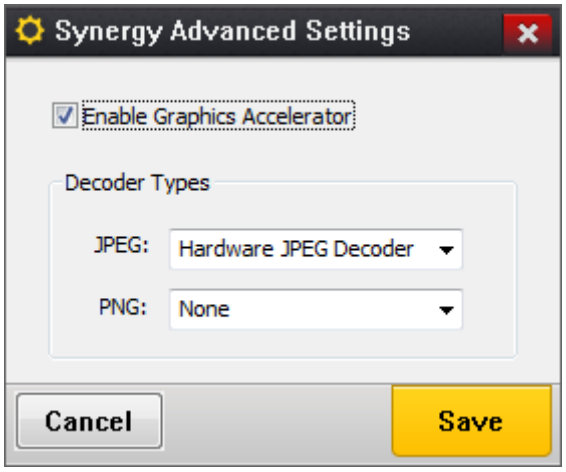

Figure 73 Synergy Advanced Settings

13. Click <Save>.

14. Setup the Display Configuration as in Figure 74.

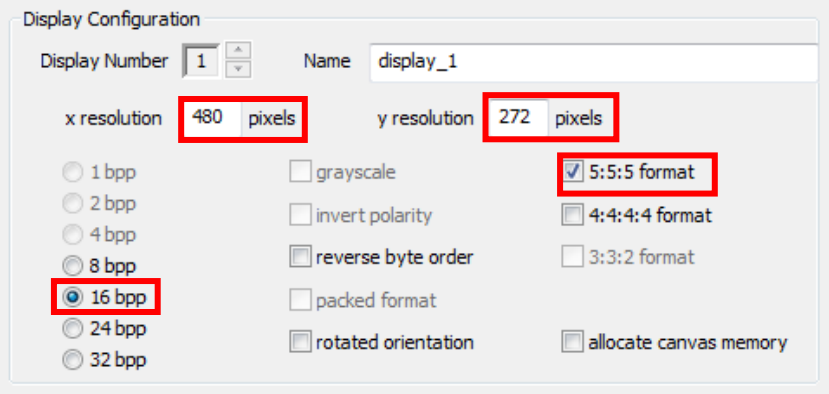

Figure 74 Configure the Display

15. Click <SAVE> to generate the project.
16. Right-click on display_1 in the project view.
17. Select “Insert -> Window -> Window”.

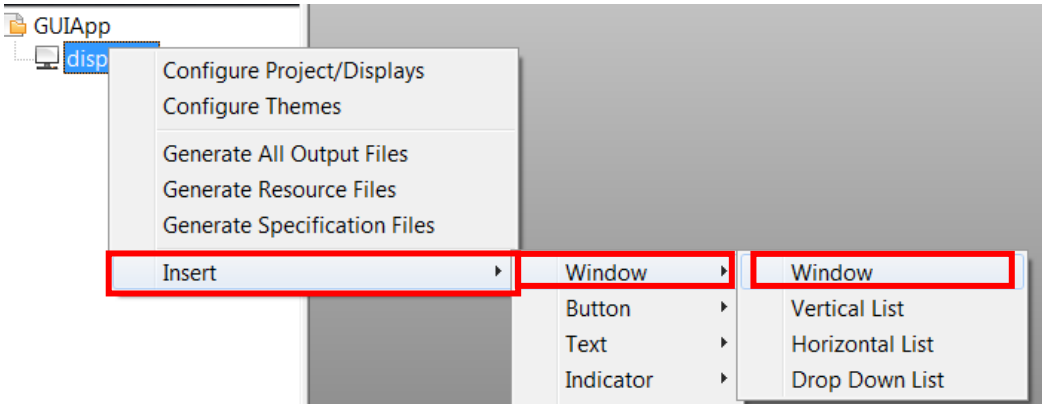

Figure 75 New Window

18. Modify the properties by selecting the new window and editing the Properties View. Update the current settings to match Figure 76.

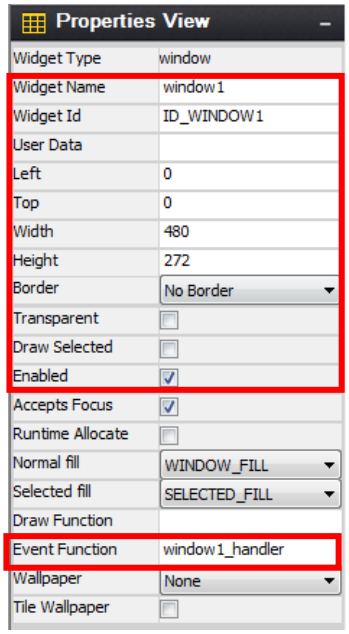

Figure 76 Configure Window1 Properties

19. In the Project View Window, right click on display_1 and create another window by selecting “Insert -> Window -> Window”.

20. Modify the properties to match Figure 77.

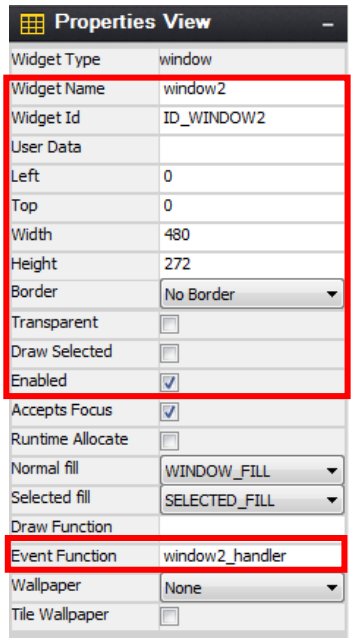

Figure 77 Configure Window2 Properties

21. In the Project View, right-click on window1 and insert a Button (Text Button) by selecting “Insert -> Button ->Text Button”.

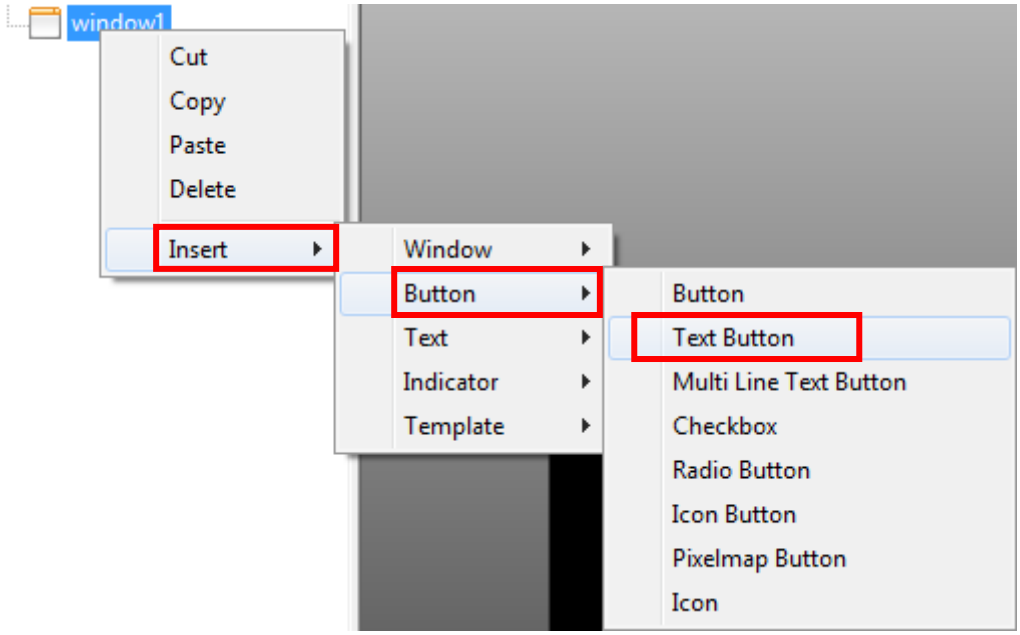

Figure 78 Add a New Text Button

22. In the Project View, right-click on window1 and insert a Button Checkbox by selecting “Insert -> Button -> Checkbox”.

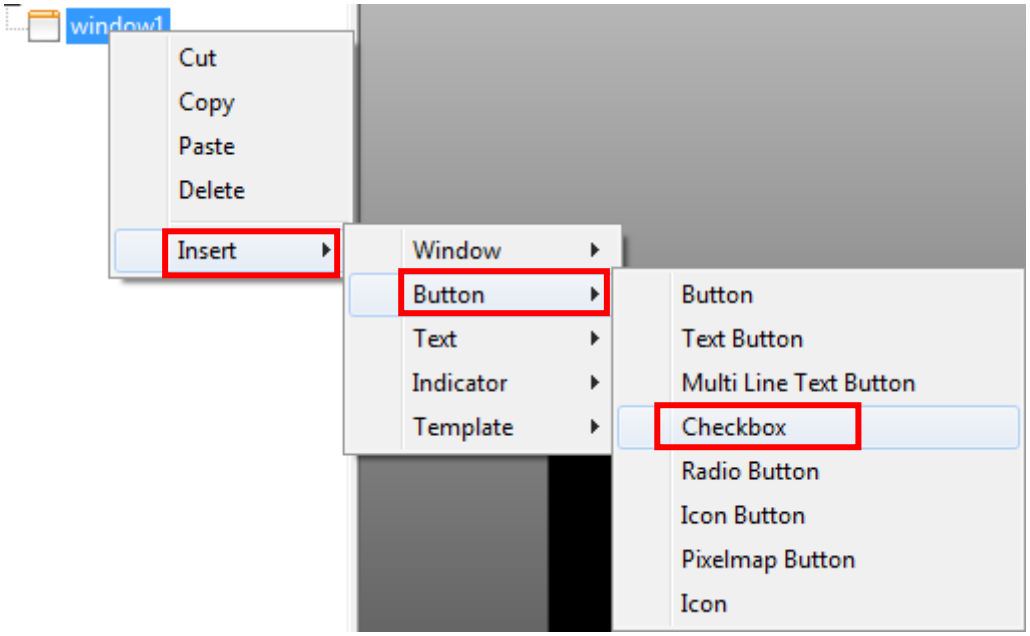

Figure 79 Add a New Checkbox

23. In the Project View, right-click on window1 and insert a Text Prompt by selecting “Insert -> Text -> Prompt”.

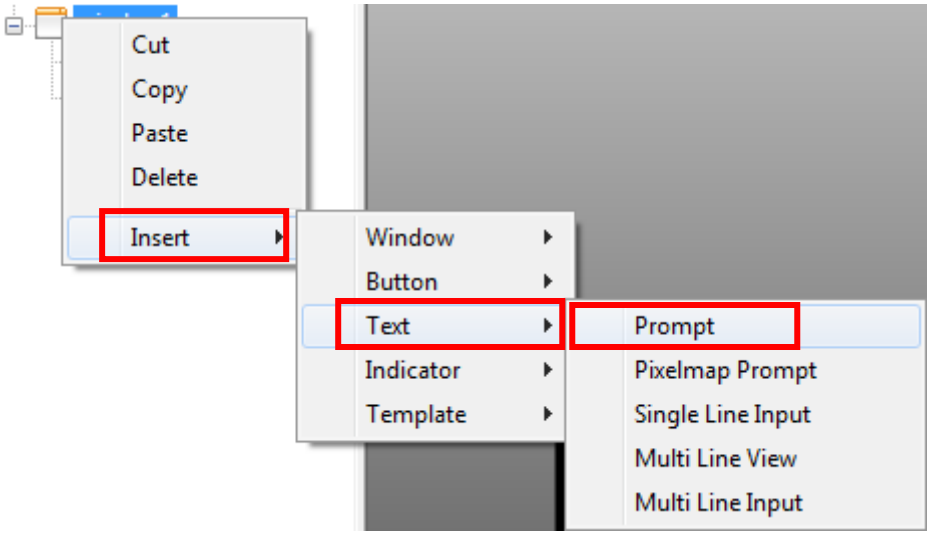

Figure 80 Adding New Prompt

24. In the Project View, right-click on window1 and insert another Text Prompt.
25. In the Project View, right-click on window2 and insert a Text Prompt.
26. In the Project View, right-click on window2 and insert another Text Prompt.

    If you have followed these directions correctly, your Project View should look like Figure 81.

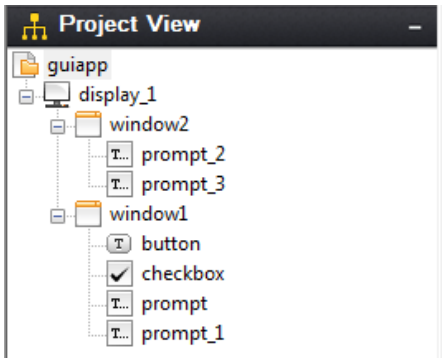

Figure 81 GUIX Project View

27. Press the ‘+’ character on right of “</> Strings” to expand the Strings menu.

Figure 82 Strings Button

28. Double click on any of the strings to open the String Table Editor.
29. Delete the existing strings by selecting them then clicking on the <Delete String> button in the String Table Editor.

30. Add the strings using the <Add String> button in Figure 83.

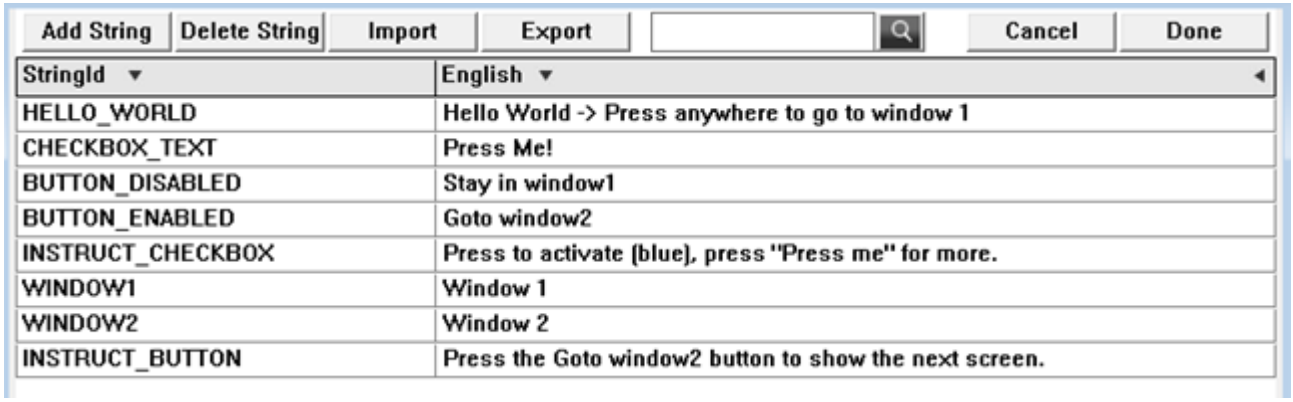

Figure 83 New Strings

31. When correct, click the <Save> button.
32. In the Project View under window1, click on button then modify the properties (Properties View) to match Figure 84.

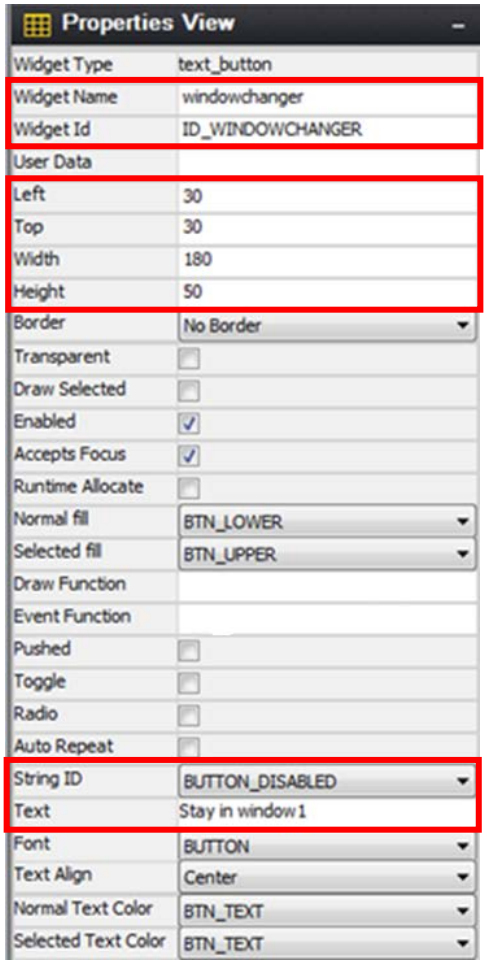

Figure 84 Configure Windowchanger Button Properties

33. In the Project View under window1, click on checkbox then modify the properties (Properties View) to match Figure 85.

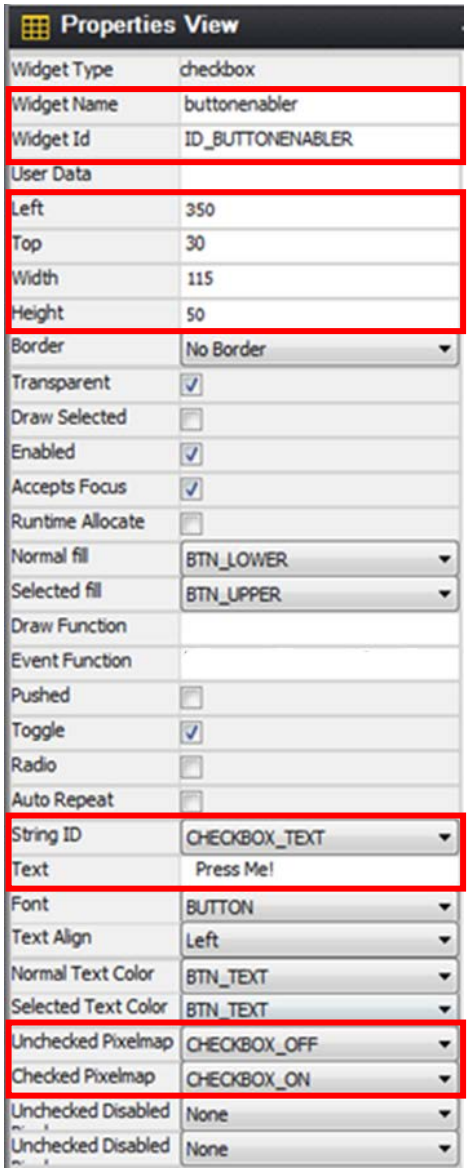

Figure 85 Configure Buttonenabler Checkbox Properties

34. In the Project View under window1, click on prompt then modify the properties to match Figure 86.

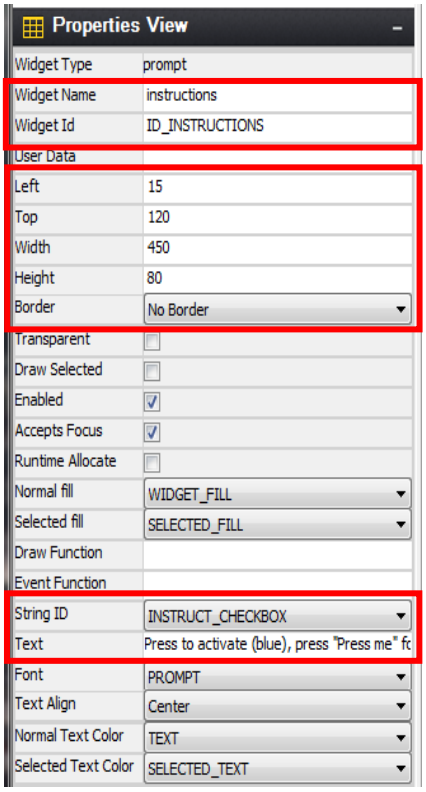

Figure 86 Configure Prompt Properties

35. In the Project View under window1, click on prompt_1 then modify the properties to match Figure 87.

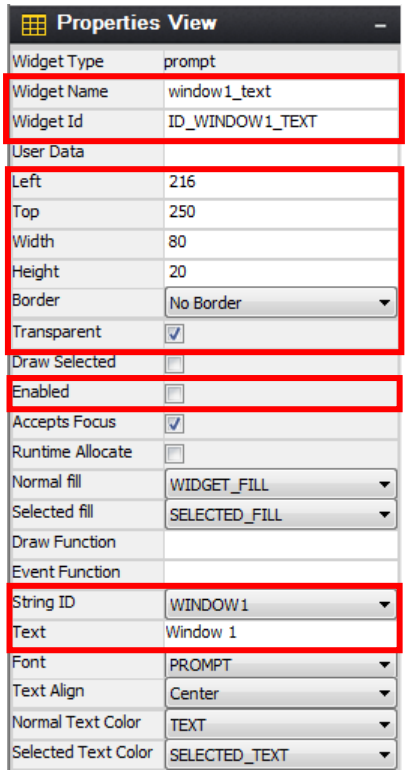

Figure 87 Configure Window Text Properties

36. In the Project View under window2, click on prompt_2 then modify the properties to match Figure 88.

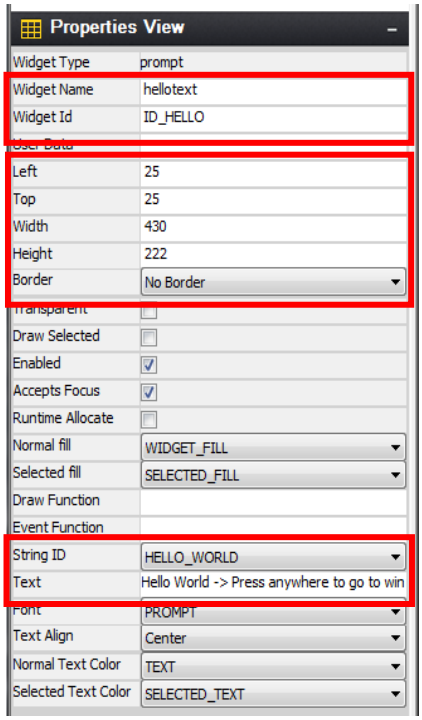

Figure 88 Configure Hello Text Prompt Properties

37. In the Project View under window2, click on prompt_3 then modify the properties to match Figure 89.

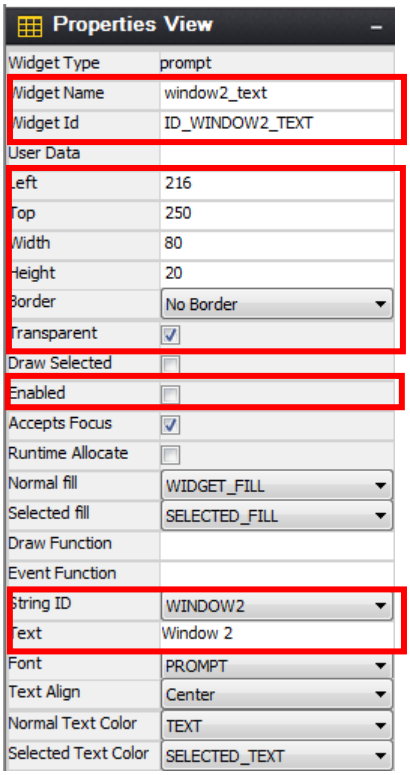

Figure 89 Configure Window Text Properties

After these configuration steps, the two windows should now look similar to Figure 90 and Figure 91.

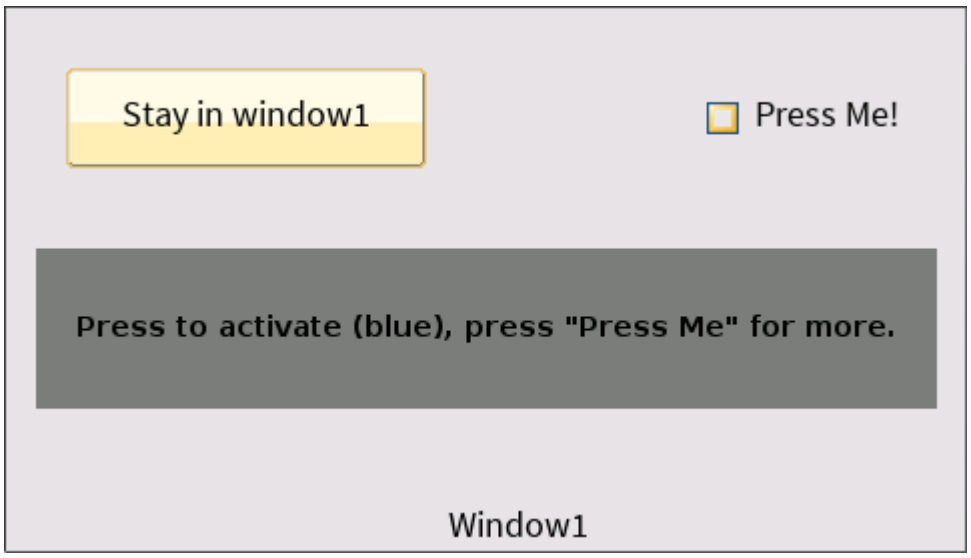

Figure 90 Configured Window1

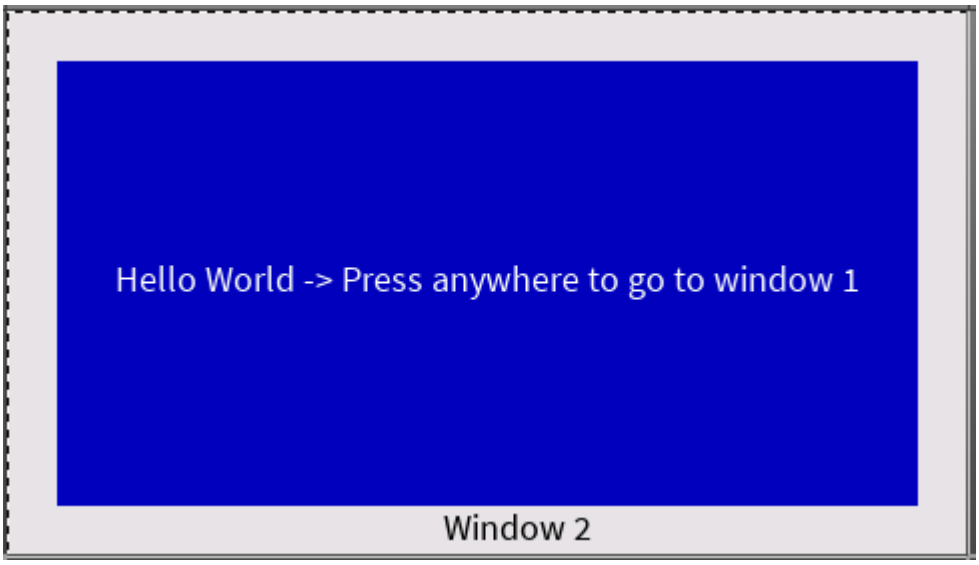

Figure 91 Configured Window2

38. Expand the “Pixelmaps” section on the right by clicking on the “+”.
39. Click “System”.

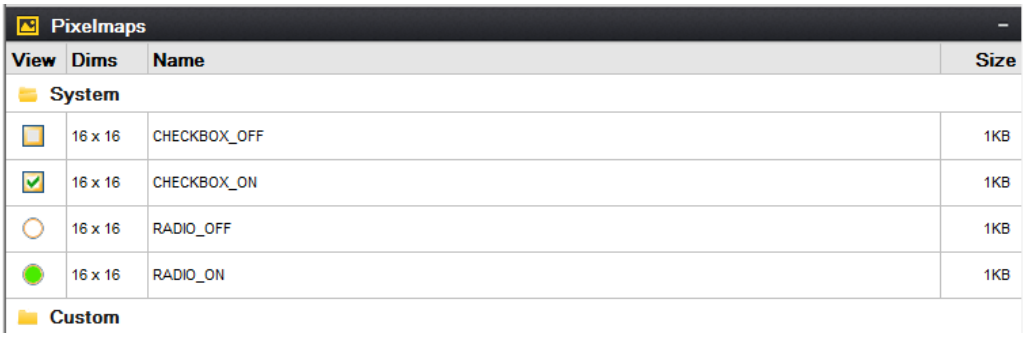

Figure 92 Configuration of Pixelmaps

40. Double-click “CHECKBOX_OFF” to edit the Pixelmap.

41. Deselect “Compress Output” and click <Save>.
42. Double-click “CHECKBOX_ON” to edit the Pixelmap.
43. Deselect “Compress Output” and click <Save>.
44. Save the project.

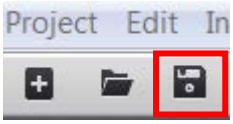

Figure 93 Save Project

45. From the pull down menu select “Project -> Generate all Output Files”.

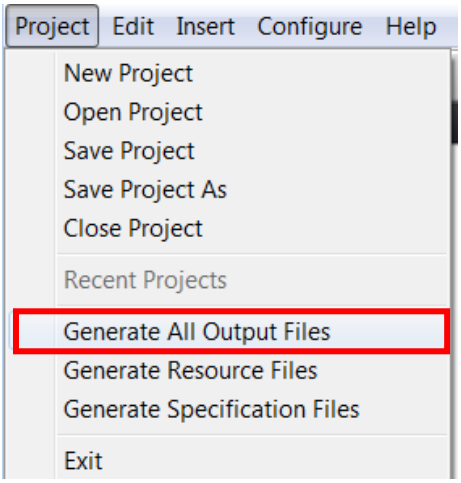

Figure 94 Generate All Output files

46. Return to e^2 studio.

6. Adding Code for Custom Interface Controls

1. Open Windows Explorer and surf to where you put the files included with this application note. Locate the file “Source Files\ guiapp_event_handlers.c”. Now drag the file from the Windows Explorer Window into the “src” folder inside the e^2 studio Project Explorer window.

    A. When asked how to import the selected files, click <OK> to copy the files.

NOTE:

This file contains the event management functions for the different graphical elements created in GUIX Studio (button, checkbox, prompt).

Build the project by clicking the Hammer icon,  , below the Menu Bar. If all steps were followed correctly, there should be no errors reported in the build output.

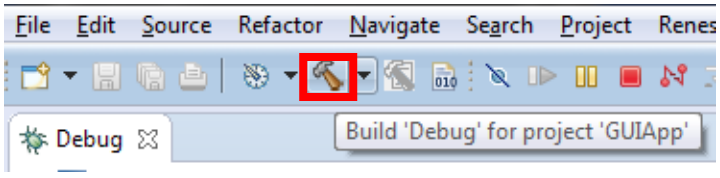

Figure 95 Build Button

GUIX handles the events that are required at a system level. To handle custom commands like screen transitions and button actions event handler need to be defined. Shown below is the event handler for window1.

```
UINT window1_handler(GX_WINDOW *widget, GX_EVENT *event_ptr)
{
    UINT result = gx_window_event_process(widget, event_ptr);

    switch (event_ptr->gx_event_type)
    {
    case GX_SIGNAL(ID_BUTTONENABLER, GX_EVENT_TOGGLE_ON):
        button_enabled = true;
        update_text_id(widget->gx_widget_parent, ID_WINDOWCHANGER, GX_STRING_ID_BUTTON_ENABLED);
        update_text_id(widget->gx_widget_parent, ID_INSTRUCTIONS, GX_STRING_ID_INSTRUCT_BUTTON);
        break;
    case GX_SIGNAL(ID_BUTTONENABLER, GX_EVENT_TOGGLE_OFF):
        button_enabled = false;
        update_text_id(widget->gx_widget_parent, ID_WINDOWCHANGER, GX_STRING_ID_BUTTON_DISABLED);
        update_text_id(widget->gx_widget_parent, ID_INSTRUCTIONS, GX_STRING_ID_INSTRUCT_CHECKBOX);
        break;
    case GX_SIGNAL(ID_WINDOWCHANGER, GX_EVENT_CLICKED):
        if(button_enabled){
            show_window((GX_WINDOW*)&window2, (GX_WIDGET*)widget, true);
        }
        break;
    default:
        gx_window_event_process(widget, event_ptr);
        break;
    }

    return result;
}
```

Events can be routed based on the ID of the widget and the signal from GUIX. For example, the checkbox ID_BUTTONENABLER can have two states; GX_EVENT_TOGGLE_ON and GX_EVENTS_TOGGLE_OFF. When the box is unchecked and then pressed, the event GX_EVENT_TOGGLE_ON is sent to the handler. After that, the box will be checked.

7. Running the Application

1. On the DK-S7G2,
   A. Set DIPSW S5 “DRAM” switch to on
   B. Set DIPSW S5 “EXP” to off
   C. Connect 5VDC Power to J1 (power plug)
   D. Connect the JLink-OB on J17 of the DK-S7G2 main board to the PC using a micro USB cable.

NOTE:

The application is not yet ready to be run on the target hardware. The following steps are necessary in order to run it.

2. On the PC, click the dropdown menu for the debug icon.

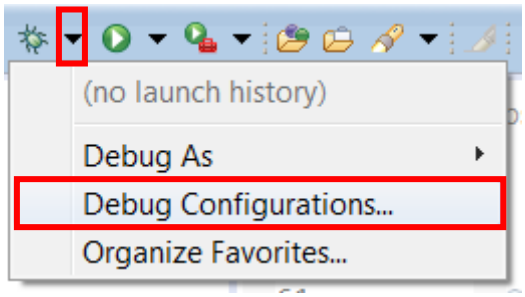

Figure 96 Debug Options

3. Select the “Debug Configurations...” option
4. Under the “Renesas GDB Hardware Debugging” section, select “GUIApp Debug”.

5. Click on the <Debug> button to start debugging.

NOTE:

If the debug button is greyed out then there is likely an issue with the build. Check all steps from the document again for mismatched options.

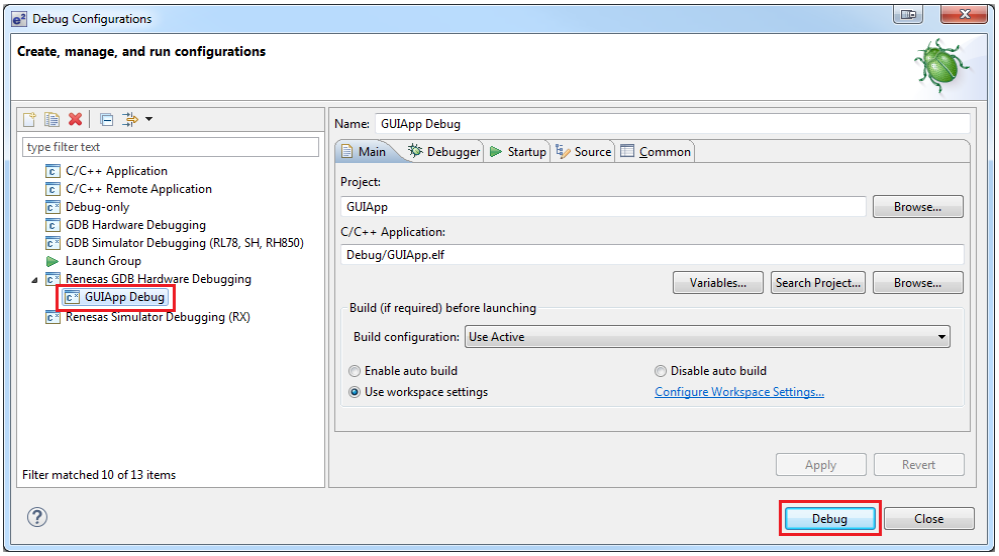

Figure 97 Debug Configurations

6. If asked to confirm a Perspective Switch, click <Yes>. (If you have previously instructed e^2 studio to remember your decision, this dialog box will not be displayed.)

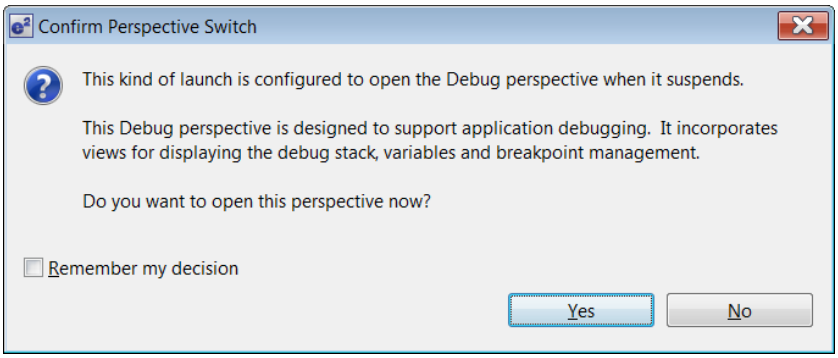

Figure 98 Perspective Switch Dialog

7. Press “F8” or the resume button to start the application. It will now stop at main.

Figure 99 Resume Button

8. Press “F8” or the resume button to run the code.

NOTE:

The GUI created earlier should now be on the screen.

9. Overview of the Demo:

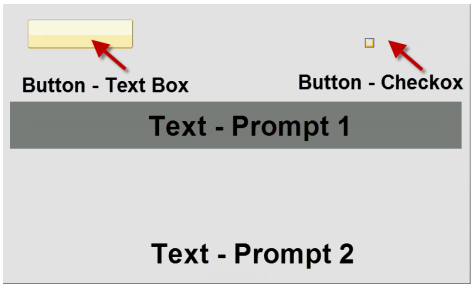

Figure 100 Window1

A. Figure 100 shows Window1. In this window are four elements:

i. **Button – Checkbox**: We use this button to enable going to Window2. Text is set to "Press Me!" and it is unchecked. When the user presses within the Checkbox active area we activate the event "window1_event_handler". This event is picked up inside "guiapp_event_handlers.c" where the code toggles the checkbox then sets the text in "Text –Prompt 1" and "Button – Text Box" to the appropriate message.

ii. **Button – Text Box**: This box simply shows what window you will go to if you press outside the "Text –Prompt 1" area. (Refer to "Button – Checkbox" to see how it is changed.) Press in this area to activate the "window1 _handler" event which is picked up by "guiapp_event_handlers.c" where the code changes the window to window2.

iii. **Text – Prompt 1**: This area instructs the user how to control the demo. (Refer to "Button – Checkbox" to see how it is changed.)

iv. **Text – Prompt 2:** This Prompt is used to show the user which window they are in. It never changes (always shows window1).

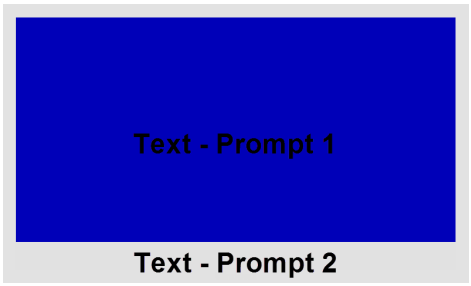

Figure 101 Window2

B. Figure 101 shows Window2. In this window are two elements:

i. **Text – Prompt 1**: This area presents "Hello World" and instructs the user how to return to window1. Pressing in this area initiates the "window2_handler" event which is picked up guiapp_event_handlers.c and changes the active window to window1.

ii. **Text – Prompt 2:** This Prompt is used to show the user which window they are in. It never changes (always shows window2).

10. Press "Ctrl + F2" or the stop button to end the debug session.

Figure 102 Stop Button

11. This concludes the GUIX "Hello World" for DK-S7G2.

Website and Support

Renesas Electronics Website: http://www.renesas.com/
Renesas Electronics Support: https://synergygallery.renesas.com/support

Technical Contact Details:

America: https://renesas.zendesk.com/anonymous_requests/new
Europe: http://www.renesas.eu/support/index.jsp
Japan: http://japan.renesas.com/contact/index.jsp

Revision History

		Description	
Rev.	**Date**	**Page**	**Summary**
1.00	1/22/2015	All	Created Initial Document
1.01	7/6/2016	All	Update to SSP 1.1.0

General Precautions in the Handling of Microprocessing Unit and Microcontroller Unit Products

The following usage notes are applicable to all Microprocessing unit and Microcontroller unit products from Renesas. For detailed usage notes on the products covered by this document, refer to the relevant sections of the document as well as any technical updates that have been issued for the products.

1. Handling of Unused Pins

   Handle unused pins in accordance with the directions given under Handling of Unused Pins in the manual.

   — The input pins of CMOS products are generally in the high-impedance state. In operation with an unused pin in the open-circuit state, extra electromagnetic noise is induced in the vicinity of LSI, an associated shoot-through current flows internally, and malfunctions occur due to the false recognition of the pin state as an input signal become possible. Unused pins should be handled as described under Handling of Unused Pins in the manual.

2. Processing at Power-on

   The state of the product is undefined at the moment when power is supplied.

   — The states of internal circuits in the LSI are indeterminate and the states of register settings and pins are undefined at the moment when power is supplied.
   In a finished product where the reset signal is applied to the external reset pin, the states of pins are not guaranteed from the moment when power is supplied until the reset process is completed.
   In a similar way, the states of pins in a product that is reset by an on-chip power-on reset function are not guaranteed from the moment when power is supplied until the power reaches the level at which resetting has been specified.

3. Prohibition of Access to Reserved Addresses

   Access to reserved addresses is prohibited.

   — The reserved addresses are provided for the possible future expansion of functions. Do not access these addresses; the correct operation of LSI is not guaranteed if they are accessed.

4. Clock Signals

   After applying a reset, only release the reset line after the operating clock signal has become stable. When switching the clock signal during program execution, wait until the target clock signal has stabilized.

   — When the clock signal is generated with an external resonator (or from an external oscillator) during a reset, ensure that the reset line is only released after full stabilization of the clock signal. Moreover, when switching to a clock signal produced with an external resonator (or by an external oscillator) while program execution is in progress, wait until the target clock signal is stable.

5. Differences between Products

   Before changing from one product to another, i.e. to a product with a different part number, confirm that the change will not lead to problems.

   — The characteristics of Microprocessing unit or Microcontroller unit products in the same group but having a different part number may differ in terms of the internal memory capacity, layout pattern, and other factors, which can affect the ranges of electrical characteristics, such as characteristic values, operating margins, immunity to noise, and amount of radiated noise. When changing to a product with a different part number, implement a system-evaluation test for the given product.

Notice

1. Descriptions of circuits, software and other related information in this document are provided only to illustrate the operation of semiconductor products and application examples. You are fully responsible for the incorporation of these circuits, software, and information in the design of your equipment. Renesas Electronics assumes no responsibility for any losses incurred by you or third parties arising from the use of these circuits, software, or information.
2. Renesas Electronics has used reasonable care in preparing the information included in this document, but Renesas Electronics does not warrant that such information is error free. Renesas Electronics assumes no liability whatsoever for any damages incurred by you resulting from errors in or omissions from the information included herein.
3. Renesas Electronics does not assume any liability for infringement of patents, copyrights, or other intellectual property rights of third parties by or arising from the use of Renesas Electronics products or technical information described in this document. No license, express, implied or otherwise, is granted hereby under any patents, copyrights or other intellectual property rights of Renesas Electronics or others.
4. You should not alter, modify, copy, or otherwise misappropriate any Renesas Electronics product, whether in whole or in part. Renesas Electronics assumes no responsibility for any losses incurred by you or third parties arising from such alteration, modification, copy or otherwise misappropriation of Renesas Electronics product.
5. Renesas Electronics products are classified according to the following two quality grades: "Standard" and "High Quality". The recommended applications for each Renesas Electronics product depends on the product's quality grade, as indicated below.
   "Standard": Computers; office equipment; communications equipment; test and measurement equipment; audio and visual equipment; home electronic appliances; machine tools; personal electronic equipment; and industrial robots etc.
   "High Quality": Transportation equipment (automobiles, trains, ships, etc.); traffic control systems; anti-disaster systems; anti-crime systems; and safety equipment etc.
   Renesas Electronics products are neither intended nor authorized for use in products or systems that may pose a direct threat to human life or bodily injury (artificial life support devices or systems, surgical implantations etc.), or may cause serious property damages (nuclear reactor control systems, military equipment etc.). You must check the quality grade of each Renesas Electronics product before using it in a particular application. You may not use any Renesas Electronics product for any application for which it is not intended. Renesas Electronics shall not be in any way liable for any damages or losses incurred by you or third parties arising from the use of any Renesas Electronics product for which the product is not intended by Renesas Electronics.
6. You should use the Renesas Electronics products described in this document within the range specified by Renesas Electronics, especially with respect to the maximum rating, operating supply voltage range, movement power voltage range, heat radiation characteristics, installation and other product characteristics. Renesas Electronics shall have no liability for malfunctions or damages arising out of the use of Renesas Electronics products beyond such specified ranges.
7. Although Renesas Electronics endeavors to improve the quality and reliability of its products, semiconductor products have specific characteristics such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Further, Renesas Electronics products are not subject to radiation resistance design. Please be sure to implement safety measures to guard them against the possibility of physical injury, and injury or damage caused by fire in the event of the failure of a Renesas Electronics product, such as safety design for hardware and software including but not limited to redundancy, fire control and malfunction prevention, appropriate treatment for aging degradation or any other appropriate measures. Because the evaluation of microcomputer software alone is very difficult, please evaluate the safety of the final products or systems manufactured by you.
8. Please contact a Renesas Electronics sales office for details as to environmental matters such as the environmental compatibility of each Renesas Electronics product. Please use Renesas Electronics products in compliance with all applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive. Renesas Electronics assumes no liability for damages or losses occurring as a result of your noncompliance with applicable laws and regulations.
9. Renesas Electronics products and technology may not be used for or incorporated into any products or systems whose manufacture, use, or sale is prohibited under any applicable domestic or foreign laws or regulations. You should not use Renesas Electronics products or technology described in this document for any purpose relating to military applications or use by the military, including but not limited to the development of weapons of mass destruction. When exporting the Renesas Electronics products or technology described in this document, you should comply with the applicable export control laws and regulations and follow the procedures required by such laws and regulations.
10. It is the responsibility of the buyer or distributor of Renesas Electronics products, who distributes, disposes of, or otherwise places the product with a third party, to notify such third party in advance of the contents and conditions set forth in this document, Renesas Electronics assumes no responsibility for any losses incurred by you or third parties as a result of unauthorized use of Renesas Electronics products.
11. This document may not be reproduced or duplicated in any form, in whole or in part, without prior written consent of Renesas Electronics.
12. Please contact a Renesas Electronics sales office if you have any questions regarding the information contained in this document or Renesas Electronics products, or if you have any other inquiries.

(Note 1) "Renesas Electronics" as used in this document means Renesas Electronics Corporation and also includes its majority-owned subsidiaries.

(Note 2) "Renesas Electronics product(s)" means any product developed or manufactured by or for Renesas Electronics.

RENESAS

SALES OFFICES Renesas Electronics Corporation http://www.renesas.com

Refer to "http://www.renesas.com/" for the latest and detailed information.

Renesas Electronics America Inc.
2801 Scott Boulevard Santa Clara, CA 95050-2549, U.S.A.
Tel: +1-408-588-6000, Fax: +1-408-588-6130

Renesas Electronics Canada Limited
9251 Yonge Street, Suite 8309 Richmond Hill, Ontario Canada L4C 9T3
Tel: +1-905-237-2004

Renesas Electronics Europe Limited
Dukes Meadow, Millboard Road, Bourne End, Buckinghamshire, SL8 5FH, U.K
Tel: +44-1628-585-100, Fax: +44-1628-585-900

Renesas Electronics Europe GmbH
Arcadiastrasse 10, 40472 Düsseldorf, Germany
Tel: +49-211-6503-0, Fax: +49-211-6503-1327

Renesas Electronics (China) Co., Ltd.
Room 1709, Quantum Plaza, No.27 ZhiChunLu Haidian District, Beijing 100191, P.R.China
Tel: +86-10-8235-1155, Fax: +86-10-8235-7679

Renesas Electronics (Shanghai) Co., Ltd.
Unit 301, Tower A, Central Towers, 555 Langao Road, Putuo District, Shanghai, P. R. China 200333
Tel: +86-21-2226-0888, Fax: +86-21-2226-0999

Renesas Electronics Hong Kong Limited
Unit 1601-1611, 16/F., Tower 2, Grand Century Place, 193 Prince Edward Road West, Mongkok, Kowloon, Hong Kong
Tel: +852-2265-6688, Fax: +852 2886-9022

Renesas Electronics Taiwan Co., Ltd.
13F, No. 363, Fu Shing North Road, Taipei 10543, Taiwan
Tel: +886-2-8175-9600, Fax: +886 2-8175-9670

Renesas Electronics Singapore Pte. Ltd.
80 Bendemeer Road, Unit #06-02 Hyflux Innovation Centre, Singapore 339949
Tel: +65-6213-0200, Fax: +65-6213-0300

Renesas Electronics Malaysia Sdn.Bhd.
Unit 1207, Block B, Menara Amcorp, Amcorp Trade Centre, No. 18, Jln Persiaran Barat, 46050 Petaling Jaya, Selangor Darul Ehsan, Malaysia
Tel: +60-3-7955-9390, Fax: +60-3-7955-9510

Renesas Electronics India Pvt. Ltd.
No.777C, 100 Feet Road, HALII Stage, Indiranagar, Bangalore, India
Tel: +91-80-67208700, Fax: +91-80-67208777

Renesas Electronics Korea Co., Ltd.
12F., 234 Teheran-ro, Gangnam-Gu, Seoul, 135-080, Korea
Tel: +82-2-558-3737, Fax: +82-2-558-5141

Colophon 5.0