Renesas Synergy™ Platform

Advanced GUIX Application

Introduction

This application note will enable you to use Express Logic GUIX™ Studio and Renesas Synergy™ GUIX Module in your own design. Upon completion of the application project, you will be able to add Synergy GUIX Module to your own design, configure it correctly for the target application, and write code using the included application code as a reference and starting point. References to more detailed API descriptions and suggestions of other applications that describe basic and advanced uses of the module are available on the Renesas Synergy™ Knowledge Base as referenced in the References section of this document and should be a valuable resource for creating more complex designs.

This Application Note explains the step by step procedure for creating an advanced GUI for the Washing Machine application using advanced GUIX features such as:

- Widget creation
- Creating multiple screens inside the main screen
- Attaching and detaching the child screen when you switch screens
- Double-buffer toggling control for screen transition without tearing
- Radial slider, vertical and horizontal slider creation
- Running animation

It also briefly covers the following design guidelines and considerations while creating the Synergy GUIX application:

- How to leverage hardware engines (JPEG codec, 2D drawing engine) present on Synergy MCU.
- Minimum Flash and RAM requirement for the Synergy GUIX.
- RAM/SDRAM use for the frame buffer depending on the screen size.
- Storing the images inside the MCU flash and external flash.
- Leveraging the touch control framework along with the I²C framework to handle the touch interface.

Required Resources

To build and run the Renesas Synergy™ GUIX Application example, you need:

Development tools and software

- e² studio ISDE v7.5.1 or greater (www.renesas.com/synergy/tools/e2-studio)
- Synergy Software Package (SSP) 1.7.5 or greater (www.renesas.com/synergy/ssp)
- SEGGER J-Link® and its associated USB driver (www.renesas.com/synergy/jlink)
- GUIX Studio v5.6.0.0 or later (www.renesas.com/synergy/tools/guix-studio)

Hardware

- Renesas Synergy™ PE-HMI1 Kit version 2.0 or later (www.renesas.com/synergy/pe-hmi1)
- PC running Windows® 7 or 10
- 5 Volt power supply or PoE adapter with Ethernet cable.
- J-Link debugger.

Prerequisites and Intended Audience

This application note assumes that you have some experience with the Synergy e² studio ISDE and Synergy Software Package (SSP). Before you perform the procedure in this application note, follow the procedure in the *SSP User's Manual* (Section: 3.2 Tutorial: **Running Blinky**) to build and run the **Blinky** project. This ensures that you become familiar with e² studio and the SSP and helps debug the connection to your board functions properly.

You are required to know the basics of the general GUI interface, its components and design principles. Some knowledge of graphical image basics such as coloring schemes, different image formats, image processing, and of GUI structural elements (window, menu, icons, widgets and so on) are needed.

If you are new to the Renesas Synergy™ Platform and GUIX, you are required to go through the following documents to understand the GUIX and its integration with Synergy. See the Reference section for the links to these documents.

- *GUIX™ Synergy Port Framework Module Guide*
- *GUIX™ Hello World Application Project on PE-HMI1*

The following documents can be used as reference materials:

- *SSP User's Manual*
- *GUIX Studio User's Guide*
- *GUIX User's Guide*

Contents

1. Getting Started with GUIX and GUIX Studio Overview 4
1.1 GUIX Application Design Consideration for Renesas Synergy™ Platform 4
2. GUIX Module APIs Overview 6
3. GUIX Based Washing Machine Application Example 7
3.1 Application Overview 7
3.2 Sequence of Steps to Create the GUIX Application 8
3.3 Washing Machine Application GUI Overview 8
3.4 Washing Machine Example Project Creation Using GUIX Studio 10
3.4.1 Create New GUIX Project 10
3.4.1.1 Selecting the Directories for the Resource Files 12
3.4.1.2 Selecting the Target CPU and Toolchain 12
3.4.1.3 Display Configuration 12
3.4.1.4 Advanced Settings 13
3.4.1.5 GUIX Library Version 13
3.4.1.6 Configured Project 14
3.4.2 Adding GUIX Studio Resources 14
3.4.2.1 Adding Pixelmaps Resources 14
3.4.2.2 Adding Colors Resources 17
3.4.3 Creating the Main Screen under Project View 17
3.4.3.1 Changing the Window Properties using Properties View 18
3.4.3.2 Adding GUIX and Synergy Logo to the Main Screen 20
3.4.3.3 Adding Time and Date Information to the Main Screen 21
3.4.3.4 Adding Home Button to the Main Screen 23
3.4.3.5 Adding Buttons to the Main Screen 24
3.4.3.6 Adding Washer Window to the Main Screen 26
3.4.3.7 Adding Slider Bar to the Main Screen 28
3.4.3.8 Adding the Screen Name to the Main Screen 29

3.4.4 Creating Garments Window 30
3.4.5 Creating Temperature Window 32
3.5 Creating Water Level Window 33
3.5.1 Resource and Specification Files from the Created Project 35
3.6 Including and Configuring the GUIX Module in an Application 36
3.6.1 Screen Specific Functions 46

4. Running the Washing Machine Application 46
4.1 Importing, Building and Loading the Project 46
4.2 Loading and Debugging the Application 47
4.3 Verifying the Application 47

5. Next Steps 48

6. References 48

Revision History 50

1. Getting Started with GUIX and GUIX Studio Overview

GUIX Studio is an embedded GUI application design tool and an environment for creating and maintaining the graphical elements in the application's GUI. GUIX is in the Middleware software library from Express Logic and is used to run the GUI application on the target MCU. More details about the Express Logic GUIX Studio and GUIX Library used with the Renesas Synergy™ Platform can be found in the *GUIX Studio User's Guide* and the *GUIX User's Guide* as referenced in the Reference section of this document.

1.1 GUIX Application Design Consideration for Renesas Synergy™ Platform

Renesas Synergy™ S7 and S5 MCU Series provide Graphics LCD capability with:

- Graphic LCD Controller (GLCDC)
- 2D Drawing Engine (DRW)
- JPEG Code

1. Memory Requirement Considerations

A. RAM/ROM requirement: GUIX requires approximately 80 KB of Read only Memory (ROM) and 10 KB of RAM for the GUIX thread and other global data structures. This requirement is excluding the canvas memory or the frame buffer.
Internal Flash / External Flash: Most applications also utilize graphical resources, that are not included in the core GUIX library storage requirements. These resources include fonts, graphical icons (pixelmaps), and static strings. This data can be stored in the read only memory section, Flash.
The compiled application along with the GUIX resources image can be placed in the internal flash. The combined code image can also be placed in the external flash, when the size of the image is bigger and needs to be placed in the external flash.
Size of this memory area is dependent on many factors, including the number and size of unique fonts used, the number and size of the graphical icons used, the output color format, and if each resource is using compressed data, since GUIX supports RLE compression of both font and pixelmap data. The storage requirements for each resource are displayed within the GUIX Studio application, allowing you to track and monitor the amount of flash memory that will be consumed by the application resources.

B. **Internal /External RAM Considerations:** Memory size for the canvas or the frame buffer is dependent on the LCD display size, color depth, image format and the number of screens. For smaller LCD displays such as (320 pixel * 256 pixel), based on the image format and the number of screens, this can be fit into the SRAM of the Renesas Synergy S7/S5 MCU Series.
For the bigger LCD display (800 pixel * 480 pixel) with image format of (16 bits RGB 565), it cannot be placed in the SRAM of S7/S5 Synergy MCU boards. In this case, the frame buffer needs to be created in the external RAM (SDRAM) based on the hardware availability on the board.
Frame buffer memory requirements are a function of the frame size as well as the color depth, and are defined by the formula:
*Frame Buffer RAM (bytes) = (x * y * (bpp/8))*
Where "x" and "y" are the dimensions of the canvas (display), bpp: bits per pixel
For example, the Frame buffer size for display of size 800 *480 with 16 bits RGB 565 is calculated as (800 * 480) * 16/8 = 768000 B.

C. **JPEG Work Buffer Considerations:** The JPEG work buffer trades off the JPEG decode speed against the buffer size. When a widget on the screen is formatted in JPEG, the JPEG work buffer is used as a temporary storage memory to create the decoded image. If the buffer size is insufficient for decoding an entire image, JPEG decoding is performed in the output buffer streaming mode.
BitBLT operation by 2D Drawing engine decodes a piece of JPEG raster image in the buffer, then transfers it to the frame buffer. The JPEG work buffer minimum size is *{(The number of pixels in the horizontal line) x (bpp (bytes per pixel) of the display format) x 8 (lines)}.*
For instance, if the decoded image is 800 pixels in a horizontal line and RGB565 format, the number is 800 x 2 x 8 = 12800 (byte). If the buffer size was smaller, JPEG decoding is not processed.
To get better throughput, the parameter **Size of the JPEG Work Buffer** should be set larger because it improves the JPEG decode throughput. The JPEG output buffer streaming mode repeats partial JPEG decode operations and the repletion comes to be overhead.

D. **Software Rendering vs Hardware Rendering:** Hardware rendering has the edge on performance as compared to the Software rendering. The performance comes with the cost of additional buffer required by the hardware engines. On memory constraint systems, this needs to be considered.
E. **Screen Rotation:** GUIX module supports screen rotation with angles 90, 180, 270 degrees. To use the screen rotation feature, GUIX requires memory (Canvas buffer) in addition to the frame buffers.

2. SSP Framework Interface

SSP provides the following framework and HAL drivers as part of the ISDE configurations for the GUIX. While designing GUIX applications, you can take advantage of the following SSP components. When using the SSP Touch Panel framework, its dependency components such as Messaging framework, I^2C driver and Touch driver are included. PWM driver to drive the backlight of the LCD can used.

- Touch Panel Framework Interface
- Messaging Framework
- I2C Driver
- Touch Driver
- PWM Driver

3. Pin Configurations

SSP provides the Pin Configurator to configure the pins such as GLCD Peripheral pins, GPIO pins, and Timer related pins. While designing GUIX applications, use the pin configurator to configure the pins. The following pins configurations are required for the LCD panel with touch interface.

- GLCDC Peripheral pin configuration
- PWM pin configuration
- Interrupt pin configuration
- I^2C pin configuration
- LCD Control pins such as, Reset pins

4. Performance Considerations

- **External Flash / Internal Flash**: The performance of the application depends on where the application and GUI images are stored. Storing the application and GUI image in the internal flash leads to better performance, compared to placing them on external flash.
- **Internal /External RAM Considerations:** Using the internal RAM (SRAM) provides better performance, compared to the external RAM (SDRAM). Applications updating screens within the main screens and their corresponding memory layout can be selectively placed in the SRAM.
- **Color Depth:** Applications based on the lower color depth have good performance, compared to the higher color depth. For example, 8 bpp and 16 bpp have better performance when compared to 24 bpp and 32 bpp image formats.
- **Display Resolution:** For LCD displays with higher resolution, the performance can be achieved with hardware engines versus software rendering.
- **Software/Hardware Rendering:** GUIX supports the JPEG and 2D Drawing Engine. Performance of the GUI is improved while using hardware engines which support graphics rendering and accelerated displaying. These hardware modules take care of writing to the frame buffer without much intervention of the CPU. GUIX supports the software rendering wherein the hardware engines are not available. In this case, handling of the frame buffer is entirely controlled by the software. Software rendering is more suited for smaller displays requiring smaller frame buffer, and application where the MCU bandwidth is under-utilized. For bigger LCD displays, performance will be an issue while using the software rendering.

2. GUIX Module APIs Overview

This section describes the APIs available for the GUIX module. These can be grouped as Synergy GUIX Framework Module Generic APIs and GUIX Specific APIs.

GUIX Framework Module APIs

These are part of the Synergy GUIX module and used for configuring the low-level graphics drivers and initializing the canvas memories. See the *SSP User's Manual (section 5.1.27, GUIX Synergy Port Framework Module)* for more details.

Table 1. GUIX Framework Module APIs

`open`	Standard SSP complaint API to open the `SF_EL_GX` module and to configure the low-level graphics device drivers and frame buffers.
`close`	Closes the `SF_EL_GX` module, and the low-level drivers.
`versionGet`	Returns the version of the module.
`setup`	Interface to initialize low-level graphics device drivers and must be passed to GUIX through GUIX (Studio) service call `gx_studio_display_configure()` as the function pointer. GUIX then calls the API back and, at that moment, the API configures the SSP device drivers based on the configuration passed by open.
`canvasInit`	GUIX helper API to determine the memory address of GUIX canvas. The API has an argument with (`GX_WINDOW_ROOT *`) type and the API provides GUIX at the start address of canvas memory, and is needed for the low-level graphics device drivers to draw/display images.

GUIX APIs: For your applications, GUIX provides a rich set of APIs. There are more than 500 APIs for the user to develop the GUI application. The logical grouping of the GUIX APIs are listed below. For more detailed API descriptions, you can refer to the section 5 of the *GUIX User's Guide*.

Table 2. GUIX API Broad Categorization Based on Features

Animation	The animation component, along with its functions and services including fading in, fading out, movement or slide-type animation for any widget type.
Utility function	Common utility functions in GUIX. Examples include converting integer to ASCII, computing mathematical functions such as square root, sine functions, and more.
Window	Window specific components and its functions. Examples are window creation, window draw, process window event, get window scroll info and others.
Drawing	Drawing primitives that are required by GUIX to draw all the visual elements on the screen.
Widget related	Widget specific component and its functions, to create the widget, draw widget border and others.
Display	Display component and its functions, for creating the display, replace color in the display table and more.
Canvas	Canvas component related processing, these APIs cover processing the canvas creation, canvas hide, canvas show and others.
GUIX System	System specific functionalities such as system initialize, system language set, system timer start and others.

The reference to the APIs can also be found in the header file `gx_api.h`.

3. GUIX Based Washing Machine Application Example

3.1 Application Overview

To understand the application, you need to start with the architecture and how it is integrated with Renesas Synergy™ Platform, its interaction with the SSP Frameworks, Threads, and the steps involved in creating it.

Figure 1 shows the architectural overview of the Washing Machine Application integrated with the SSP Frameworks and HAL drivers. Here, the Application Human Machine Interface Thread (HMI Thread) performs the following:

- Initializes the GUIX and its drivers
- Configures the display
- Initializes the canvas
- Opens the PWM driver for backlight

Touch Panel Framework handles the touch controller on the LCD. Touch events from the touch panel of the LCD are received through the I²C interface. Touch Panel Framework processes the touch events and sends the touch event using the Messaging Framework to its subscribed threads. HMI thread that is subscribed to the touch event waits for the touch events from the Touch Panel Framework and gets the event from the Messaging Framework.

HMI thread also translates the touch event into the GUIX event for further processing by the GUIX thread (created as part of the GUIX Framework).

SF_EL_GX is the adaptation layer for GUIX to interact with the Synergy MCU board, to access the graphics engines GLCDC, DRW (2DG engine) or JPEG decode engine.

SF_EL_GX has the following key functions:

- Adapts GUIX to the SSP Framework.
- Attaches the SSP Display interface driver to the GUIX Display Driver Interface.
- Allows GUIX to draw widgets accelerated by the Synergy D2W (2DG) engine.
- Supports double-buffer toggling control for screen transition without tearing.

See the *SSP User's Manual* for more about GUIX Integration, SF_EL_GX and its HAL dependency.

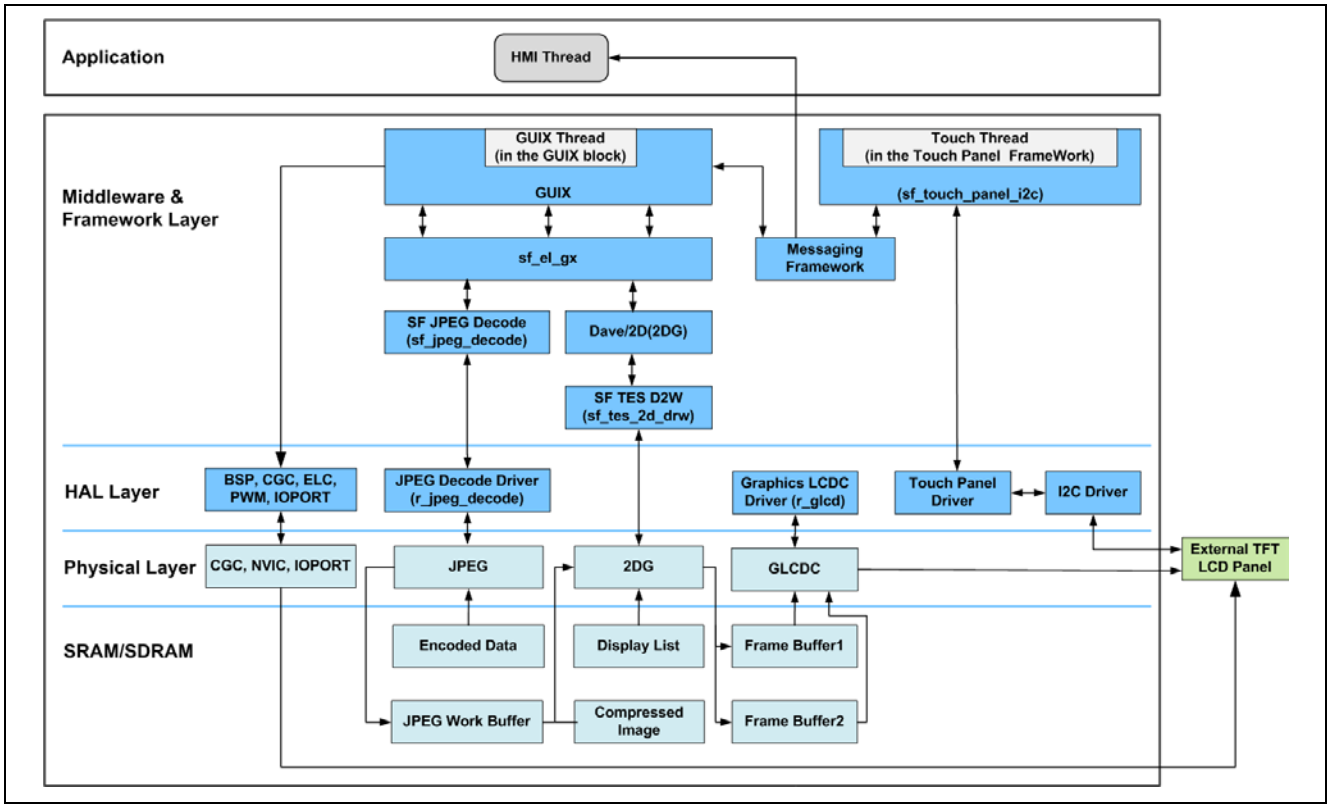


Figure 1. Architectural Overview

3.2 Sequence of Steps to Create the GUIX Application

Use the following sequence of steps while creating the GUIX Application. Steps 1 and 4 are not covered in the document, and the other steps are covered in the section 3.4 onwards, and section 4.

1. Create the Pixelmap images for the GUI screen.
   This is generally done by the Graphic Designer. These images are the inputs for the GUIX studio to create the required screens.
2. Create the GUI with GUIX Studio.
   Launch the GUIX Studio and create the different screens as per the project requirement.
3. Generate the resource files.
   After the screens are created with GUIX Studio, the widget and its handler functions need to be generated in the form (C Code and C data structures) for compilation with the application code.
4. Write code for Screen event and Screen Draw Handler functions.
   Each screen has an event handler and draw handler. User needs to code this to handle the events and to draw the screen as per the project requirement, and also the other initialization and utility functions.
5. Configure the GUIX using the ISDE Configurator.
   Create the Application Thread. GUIX is integrated with the SSP. You will need to add and configure the GUIX component using the ISDE configurator based on the LCD Type, Touch controller and Graphics engines being used in the project
6. Integrate the GUIX Application code with Synergy e^2 studio ISDE.
   In addition to the generated resource files, you need to initialize the GUIX system, configure the GUIX drivers, initialize Canvas, create screens using widget creation APIs, start the GUIX and handle the Touch Events from the Touch driver. All these are done from the Application Thread (HMI Thread in the Example Application).
7. Build the code. Generate the code and build it for the target MCU.

3.3 Washing Machine Application GUI Overview

The application demonstrated in the following section is the Washing Machine controller. Figure 2 shows the Graphic Designer's visual representation of the different screens to configure the Washing Machine controller. The four different screens are categorized as:

1. Main screen (Washer Settings screen)
2. Garments selection screen
3. Water level selection screen
4. Temperature selection screen

The application demonstrates the simulation of the Washing Machine controller from the GUI perspective.

Main Screen: Main screen has Washer Selection Control with radial slider along with the buttons to select the different screens and to power on/off the screen.

Along with the control configurations, there are:

- Status bar to indicate the Washing Cycles such as Soak, Wash, Rinse, and Spin.
- Status bar to indicate the remaining time of washing cycle.
- Time and date information

Garment Selection Screen: To control the different garments selection with radial slider along with the buttons to select the different screens and power on/off the screen.

In addition to the control configurations, this screen has:

- Status bar to indicate the Washing Cycles such as Soak, Wash, Rinse and Spin.
- Time and date information.

Water Level Selection Screen: This has the vertical slider to select the different water level settings as required by you, and also the buttons to select the different screens and power on/off the screen.

Additionally, there are:

- Status bar to indicate the Water Level in terms of percentage of the water to be filled into the tank.
- Status bar to indicate the Washing Cycles such as Soak, Wash, Rinse and Spin.
- Time and date information.

Temperature Selection Screen: This screen has the radial slider to control the water temperature for washing. Also included are the buttons to select the different screens and power on/off the screen.

Along with the control configurations, there are:

- Status bar to indicate the current water temperature.
- Status bar to indicate the Washing Cycles such as Soak, Wash, Rinse and Spin.
- Time and date information.

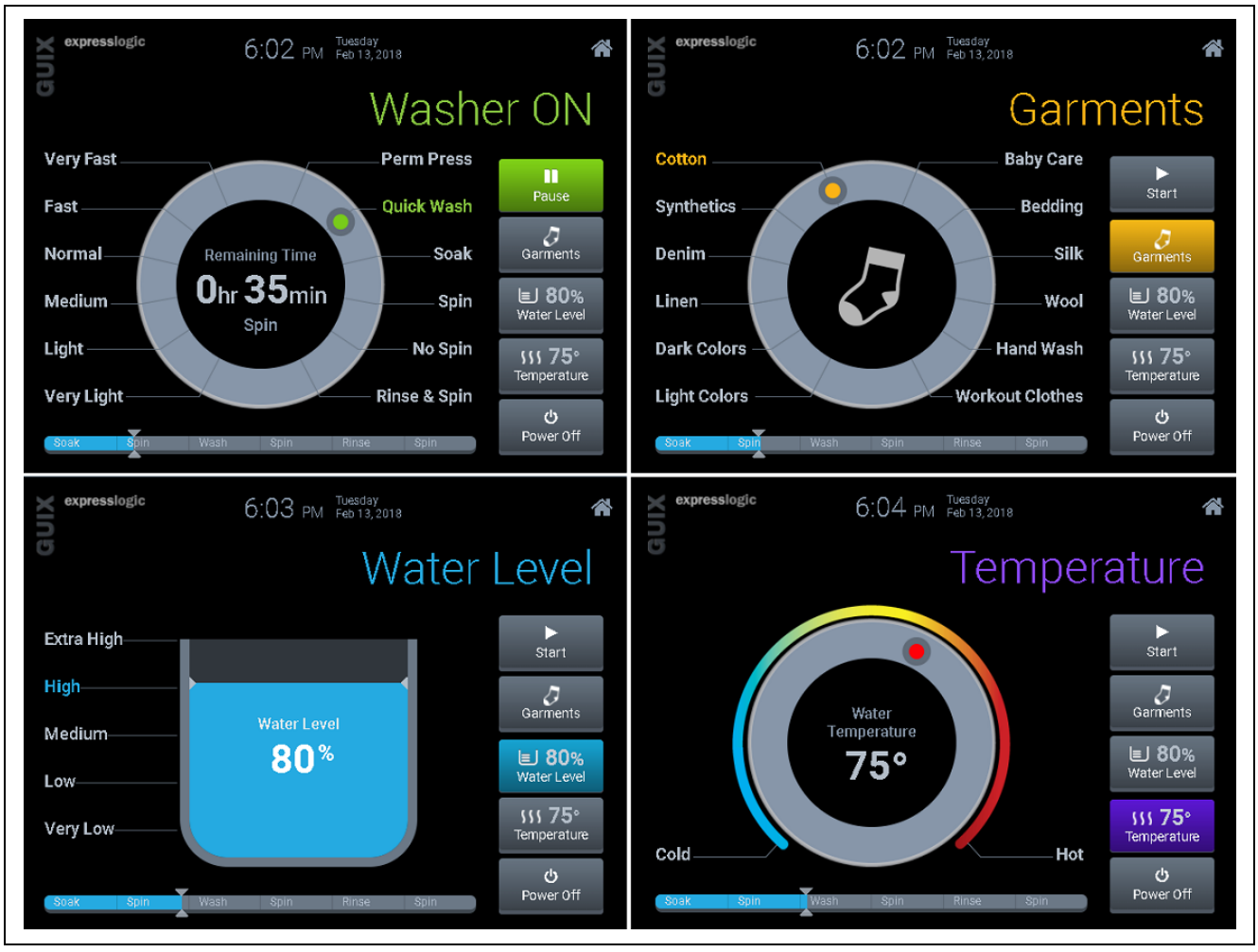


Figure 2. Washing Machine Controller GUI Overview

The following figure shows the logical representation and details of the main screen with the logical blocks numbered as follows:

1. Express Logic GUIX and Express Logic logo
2. Time, date and calendar information
3. Home button
4. Washer status
5. Control buttons to select the different screens
6. Status bar to represent the Washing cycle
7. Washer control window
8. Radial slider with wash status information

Figure 3. Washing Machine Controller GUI Overview

The numbered logical blocks are used in the same order when we create these widgets using the GUIX Studio in the following sections.

The next few sections assume that you have already gone through the basics of GUIX, GUIX Studio Views and the *GUIX Module Guides* listed in the Reference section at the end of this document. If not, it is advised to revisit those materials before continuing.

Before creating the GUI using GUIX Studio, the artistic visuals and its image files in the form of .png format (portable network graphics) should to be available in GUIX Studio. These image files are the input for the graphic resources. The creation of artistic visuals and its image files are beyond the scope of this Application Note. It is assumed that you are getting these resource files from the Graphic Designer.

3.4 Washing Machine Example Project Creation Using GUIX Studio

This section provides step-by-step procedures for Washing Machine GUI creation using the GUIX Studio. Before continuing to the detailed steps in creating the Example Project, you are advised to know how to launch GUIX Studio, and its **different Views**: Project View, Properties View, Target View, Resources View (Color, Fonts, Pixelmaps, strings and others). These operations are documented in chapters 3 and 4 of the *GUIX Studio User's Guide* (see link in the References section of this manual). To acquire a better understanding, go through these sections first before continuing through the following sections.

3.4.1 Create New GUIX Project

From the GUIX Studio, open **Project tab > New Project** and a **Create New Project** window pops up as shown in the following figure.

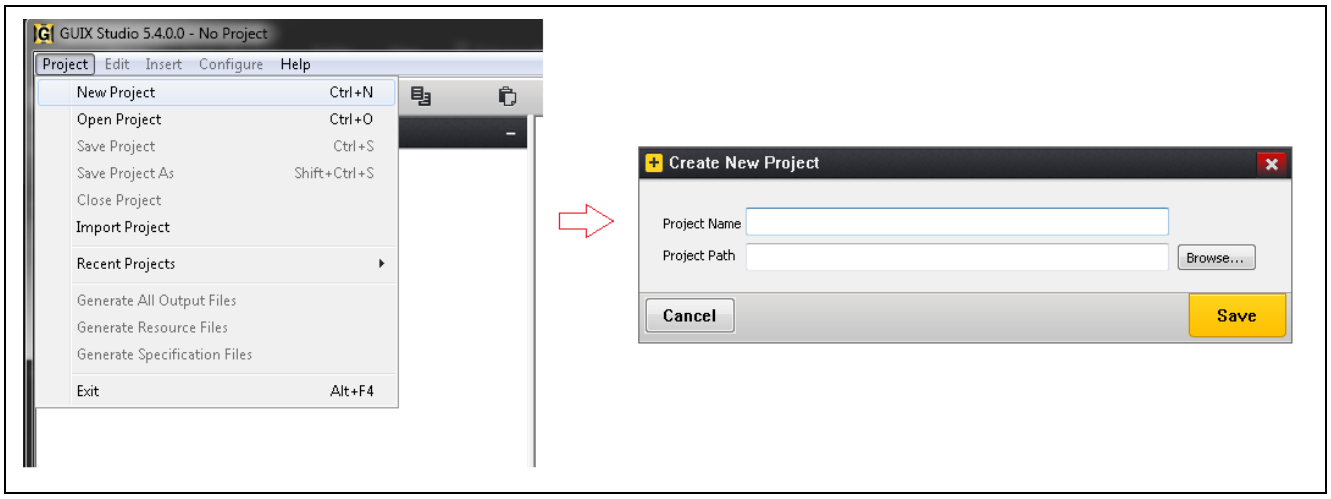


Figure 4. New Project creation

Enter the **Project Name** and **Project Path** where the directory project file needs to reside and press the **Save** button.

After the Save button is pressed, the **Configure Project** window appears. This Window has many configurations specific to the initial configuration of the project. In the next sections, the details of these configurations and how to modify them are explained.

Note: Keep the project path as the location where the GUIX Studio is installed as shown in the following figure.

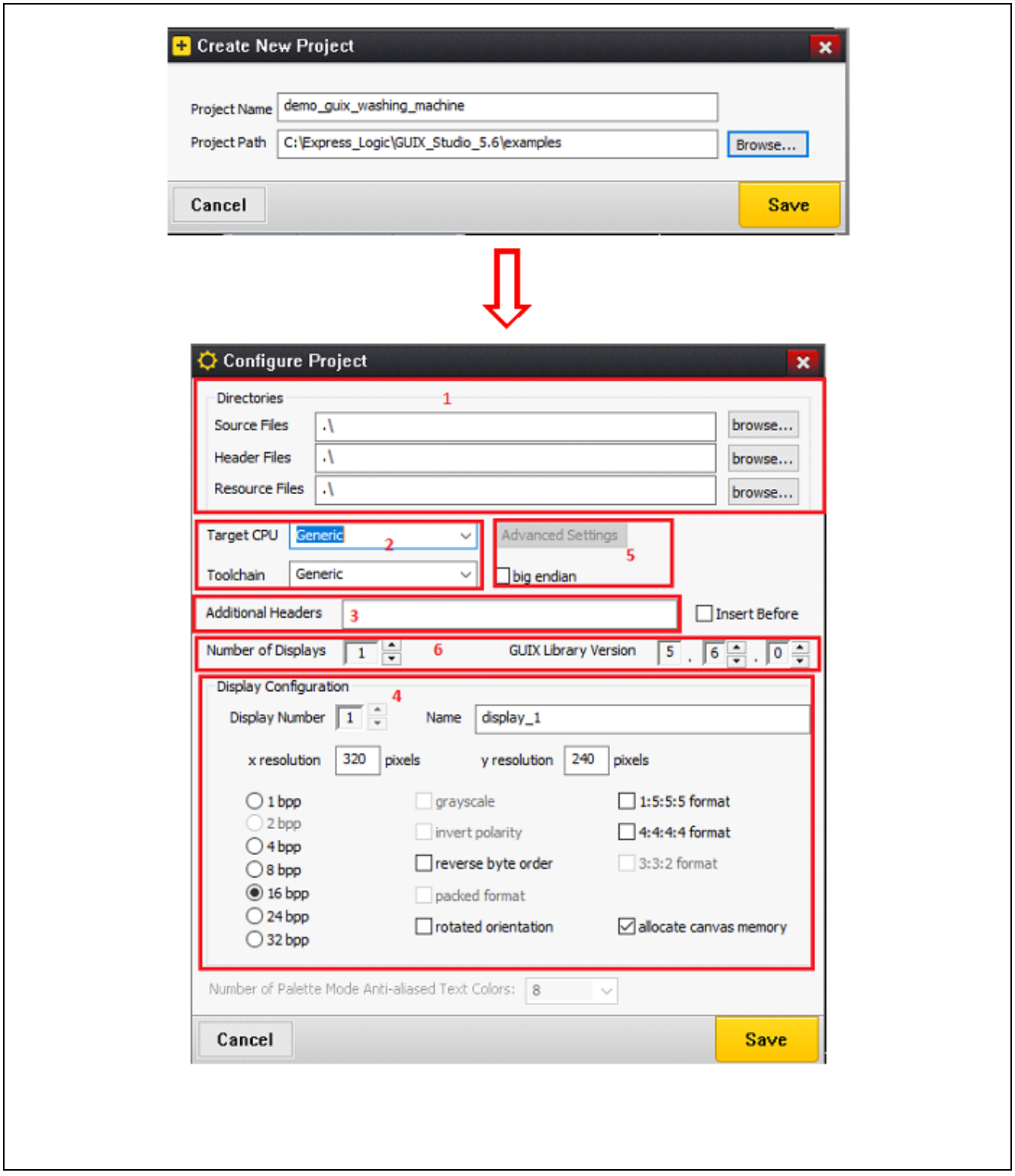


Figure 5. New Configure Project Window

3.4.1.1 Selecting the Directories for the Resource Files

From the **Configure Project** snapshot as shown in the Figure 5, directories section (1) shows the selectable directories where the GUIX source files and header files need to reside. The resource files directory is also your selectable directory where the GUIX Studio generated resource files (specification file will be located).

If the Project requires additional headers, it can be configured using the Additional Headers sections (3) as highlighted in the Project settings.

Note: It is recommended to point to the directory where application source code is located in your workspace. By doing so, when the modifications are done with Studio, the generated resource files are available to the project for the build. This avoids manual copying of resource files.

3.4.1.2 Selecting the Target CPU and Toolchain

GUIX Studio supports different target CPUs and Toolchains (2). By default, it supports the generic CPU and generic tool chain. In addition, it supports the Renesas Synergy™ CPU along with the GNU/IAR toolchain. To develop your applications on Renesas Synergy™ Platform, you are required to select the CPU as **Renesas Synergy** and the Toolchain as **GNU** or **ss**, depending on the project requirement. In this application, the GNU toolchain is selected.

3.4.1.3 Display Configuration

In the display configuration for the GUIX project (4), you can select the number of displays required for the project, which are configured using the Display Number. For the Washing Machine Application, the display number chosen is 1. You can select the name for the display, the name is chosen as the **main display**.

GUIX configuration gives you the ability to select the X and Y pixel resolution. The display resolution is selected as 800 * 480 (resolution of LCD on the PE-HMI1 board).

GUIX configuration provides the number of bits per pixel. In this example, it is selected as 16 bpp. (Note: Support for 24 bpp and 32 bpp is not there yet).

It provides you with the ability to configure the settings such as:

- grayscale
- invert polarity
- reverse byte order
- packed format
- rotated orientation

Additionally, it provides the different RGB format selection as required by the project.

The GUIX configuration provides the Allocation of Canvas memory, where this is more applicable in the windows environment. For the embedded systems, this option is not selected.

3.4.1.4 Advanced Settings

Provides the advanced settings to configure the features that are available as part of the Renesas Synergy™ MCU Group. You can select the settings for the Runtime Image Decoder and the option to choose the 2D Drawing Engine. In the Washing Machine application, the 2D Drawing Engine and Hardware JPEG Decoder is selected as shown in the following figure.

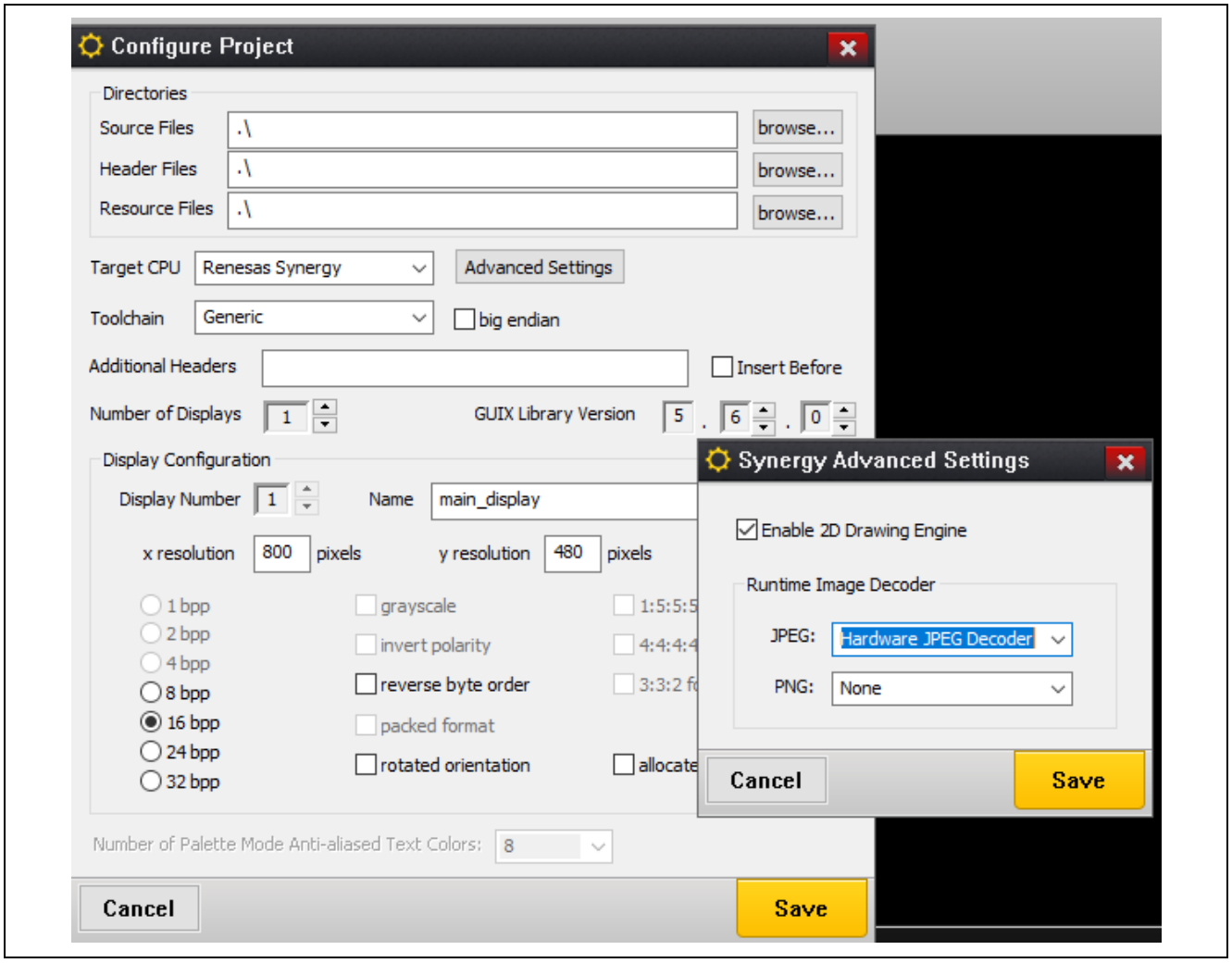


Figure 6. Synergy Advanced Settings

3.4.1.5 GUIX Library Version

This section provides details on how to configure the GUIX Library version. It is recommended to use the GUIX Library version used in the SSP release that works with GUIX Studio. The library version 5.6.0 is used along with GUIX Studio version 5.6.0.0.

3.4.1.6 Configured Project

The following figure shows the configured project with all the required configuration for the Washing Machine application.

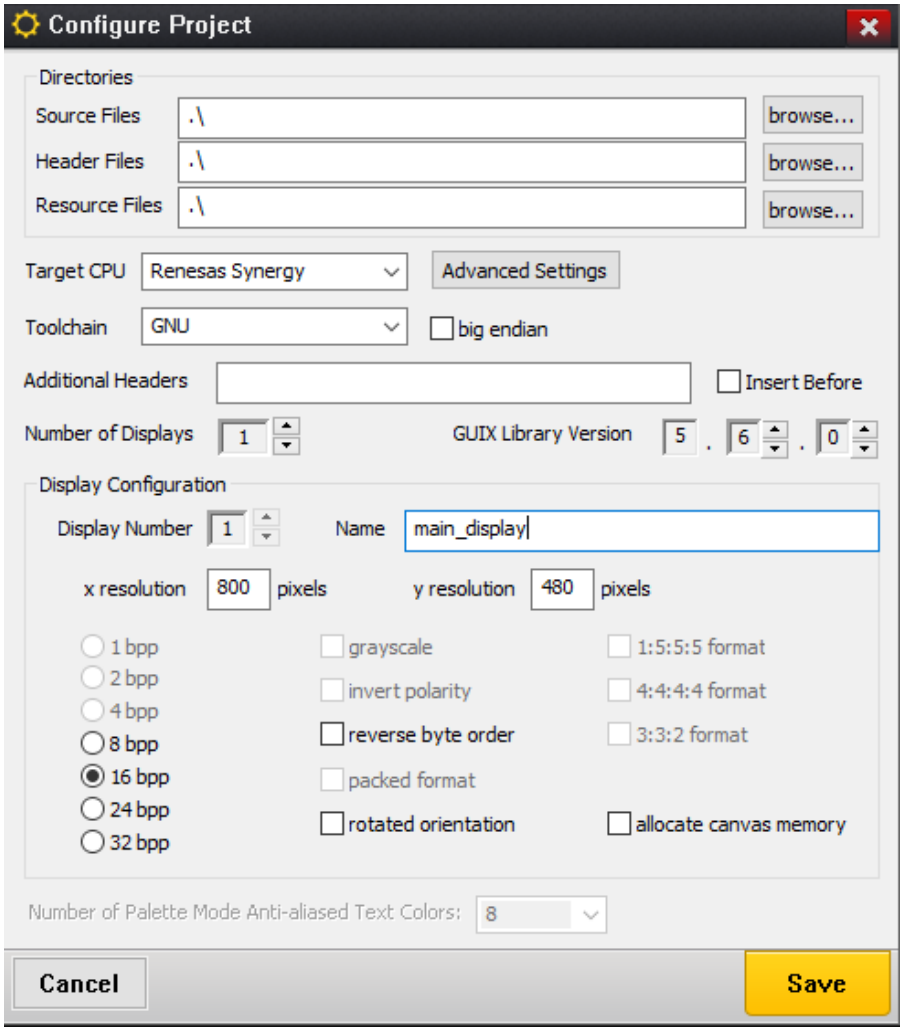


Figure 7. Washing Machine Project Configuration Settings

3.4.2 Adding GUIX Studio Resources

GUIX Studio provides you with an interface to the resources such as colors, fonts, pixel maps and strings. In this section, adding the pixelmaps and color resources are covered briefly. For additional coverage of GUIX resources, see *GUIX Studio User's Guide*, chapter 4 (GUIX Studio Resources), which is listed in the References section at the end of this document.

3.4.2.1 Adding Pixelmaps Resources

Adding pixelmaps adds graphic resources to the project using GUIX Studio. When the Graphical User Interface (GUI) design is obtained from the Graphics Designer, you will get the individual .png files for the images. These images are required to be added to the pixelmap resources to use it from GUIX Studio. Pixelmaps contain the System directory and Custom directory. The System directory has the standard images. Custom directory is the directory where the custom designed images will be placed. You are required to copy the images to this directory.

On the right side of the GUIX Studio IDE, you can notice the resources section. **Click** on the **Pixelmaps**, select the Custom folder, and click on the (**greyed + Add New Pixelmap**) **Pixelmaps->Custom + Add New Pixelmap** as shown in the following figure.

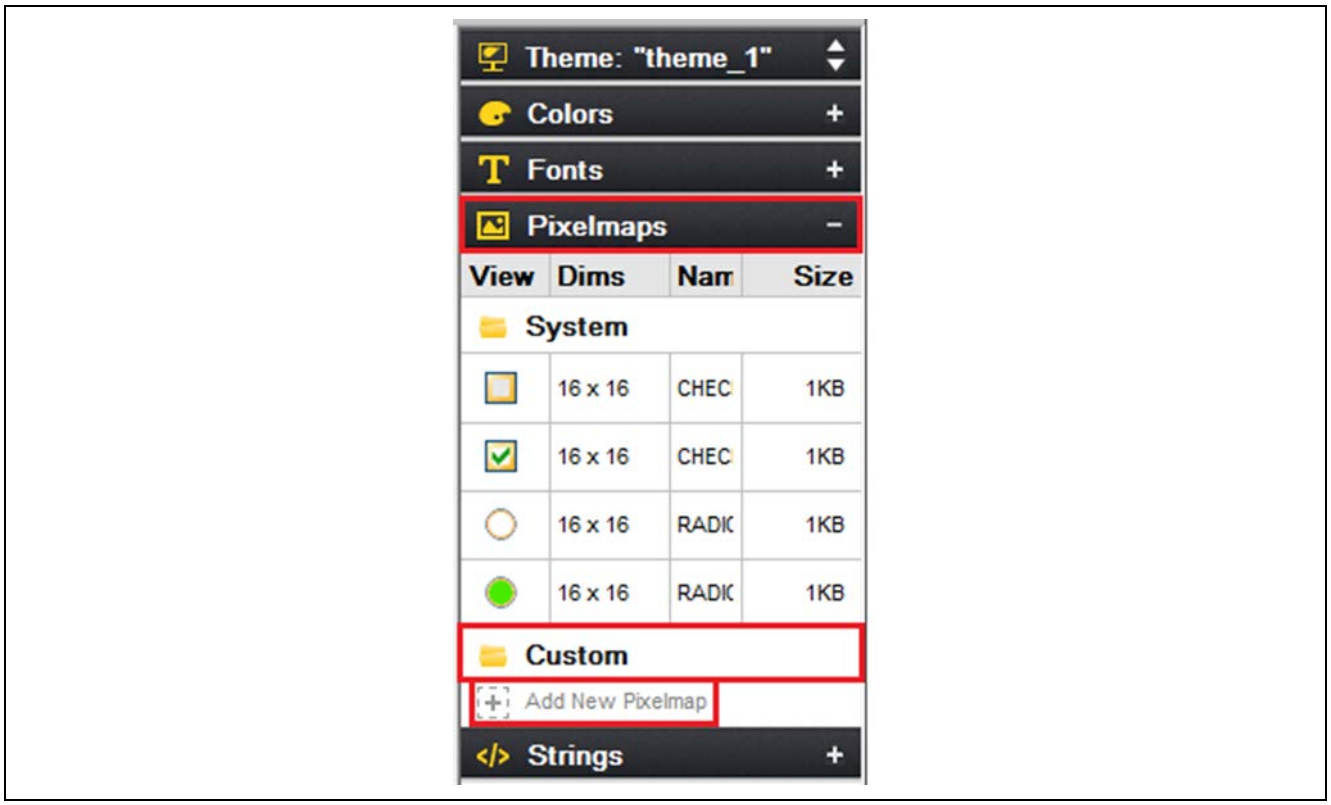


Figure 8. Adding Pixelmap Resources

This opens the window to select the image files. Point to the right directory to import image files into GUIX Studio. The following figure has a sample snapshot showing how it is done.

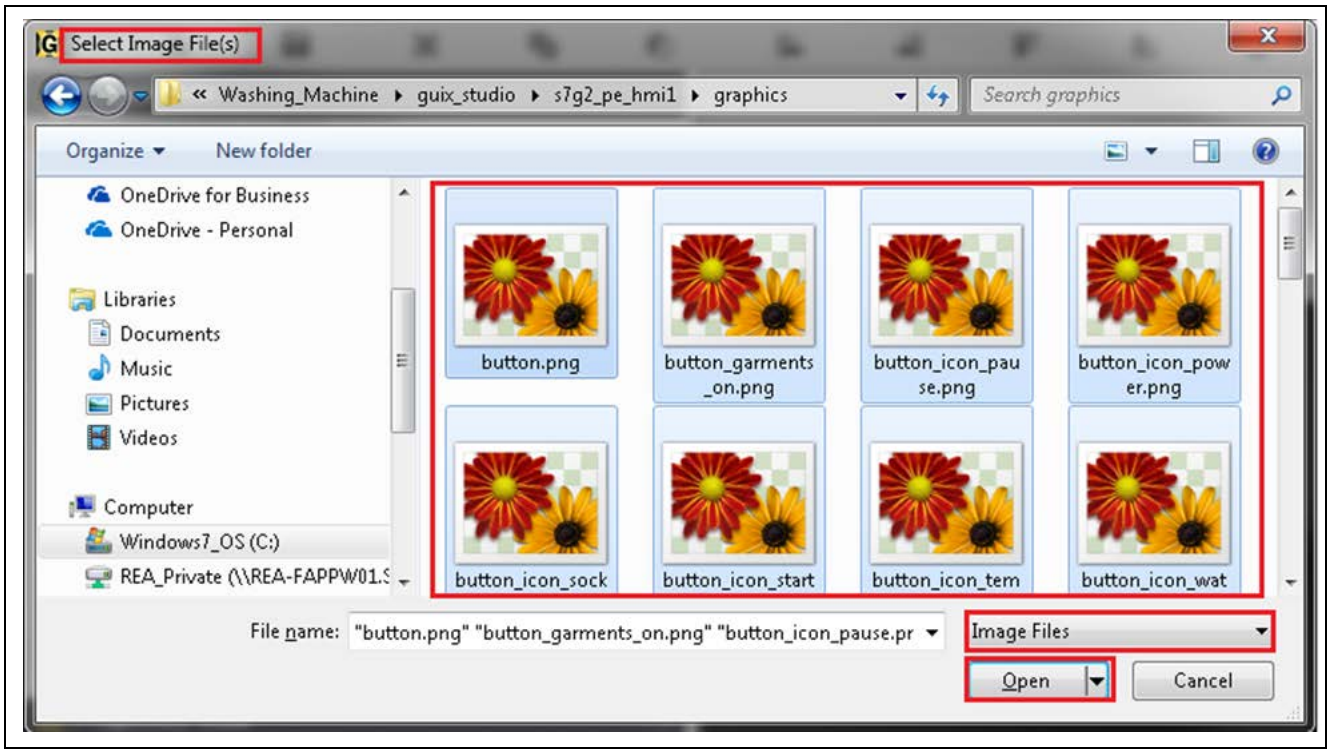


Figure 9. Adding Selected Image Files to Pixelmap Resources

The following figure shows that after the image files are imported to the GUIX Studio, you can see them on the GUIX Studio Resources under Custom Pixelmaps.

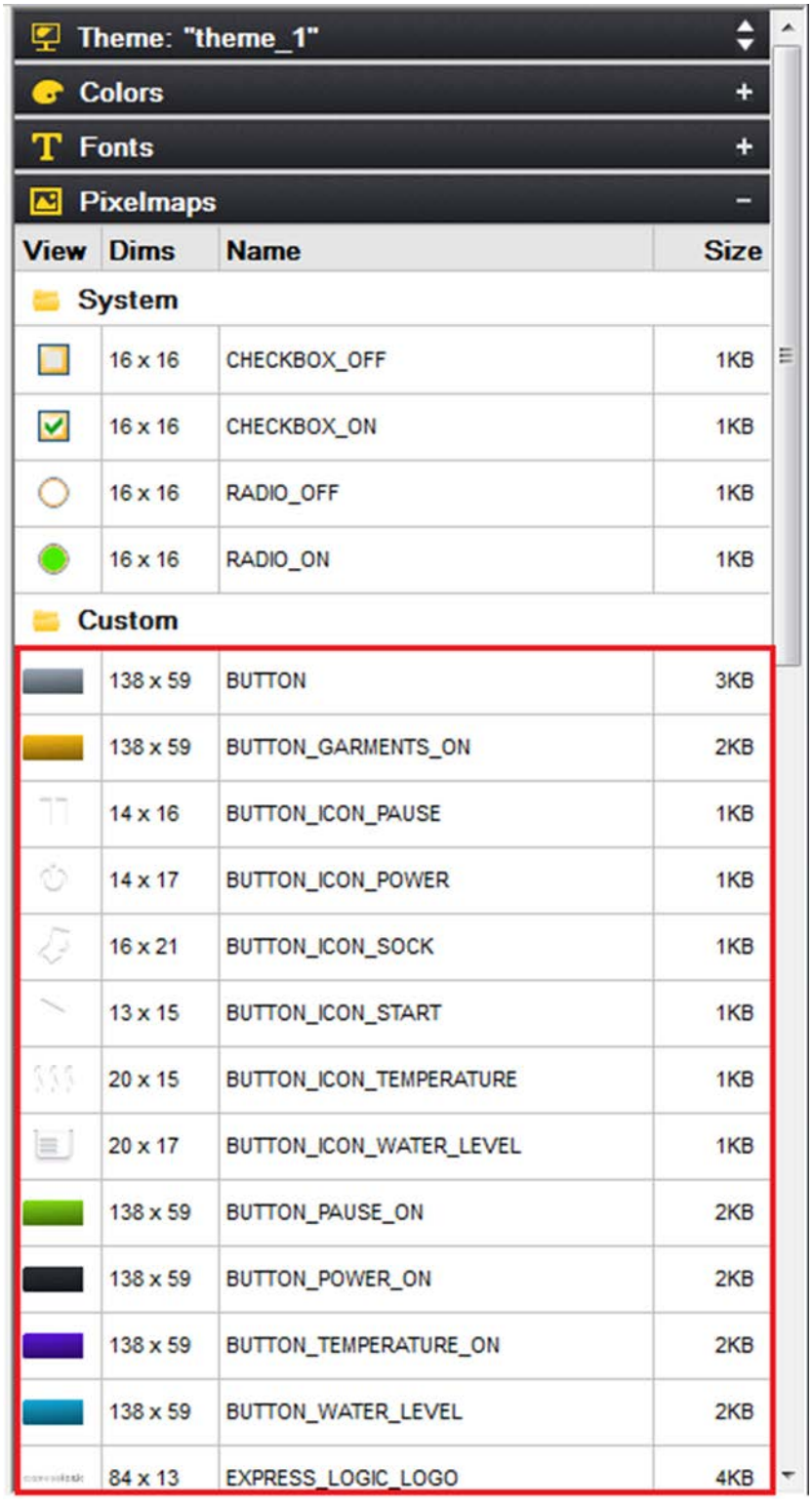


Figure 10. Added Pixelmap Resources

3.4.2.2 Adding Colors Resources

In addition to the existing default colors provided by GUIX Studio, you can add required colors to the project. The following figure shows how to add a new color to the project, **+** (**Add New Color**) by clicking the **+** sign. Clicking **+** opens a popup window to select the **Color Name**, as well as configures the RGB values to get the perfect shade of the color. Once the required shade is determined, the color can be saved. The following figure shows the Washing Machine application with a new color; in this example, adding the new color SILVERY. These colors, once added, are available in GUIX Studio for the new screen designs.

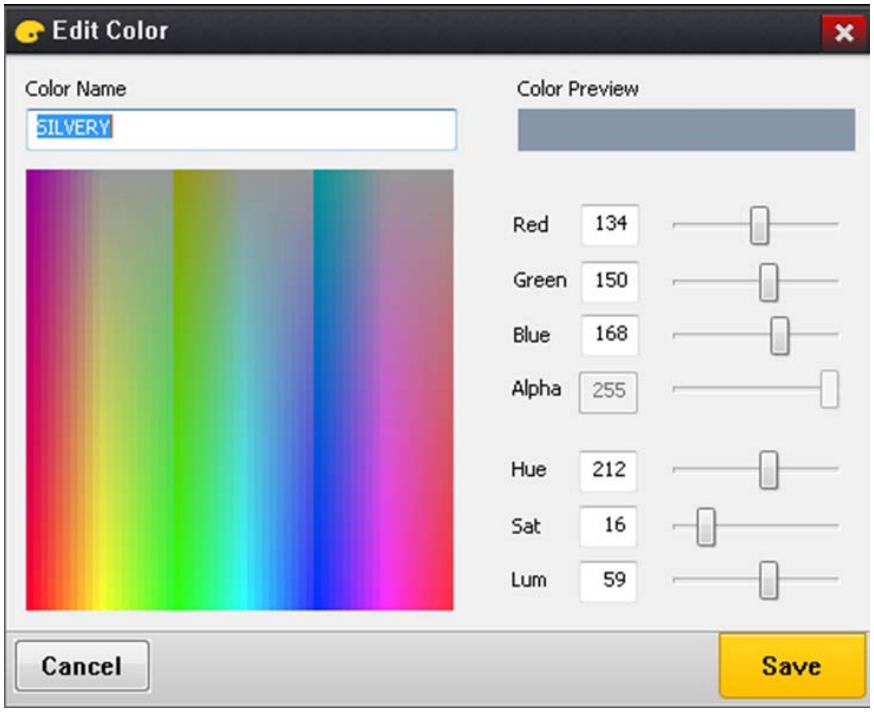


Figure 11. Adding New Color Resources

3.4.3 Creating the Main Screen under Project View

After the initial project configuration and adding the resource files to GUIX Studio, you can start creating the different windows (screens) as required for the project. The following figure shows a snapshot for adding a new window to the display. **Right-click** on the **main_display > Insert > Window > Window**, to add a new window to the display.

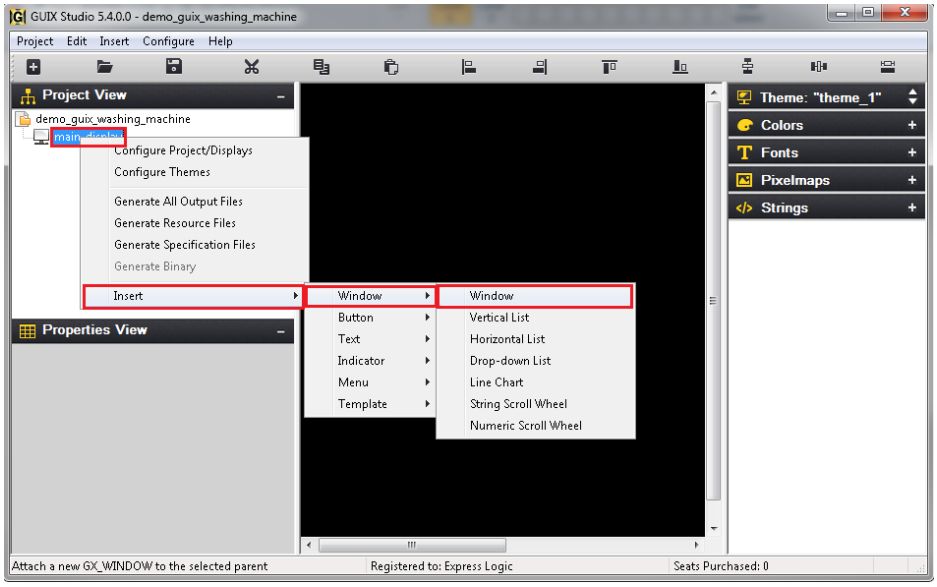


Figure 12. Adding New Window to the Display

3.4.3.1 Changing the Window Properties using Properties View

From the created window, you can change the properties displayed using the Properties View. The following figure shows that the default window created is half of the parent window.

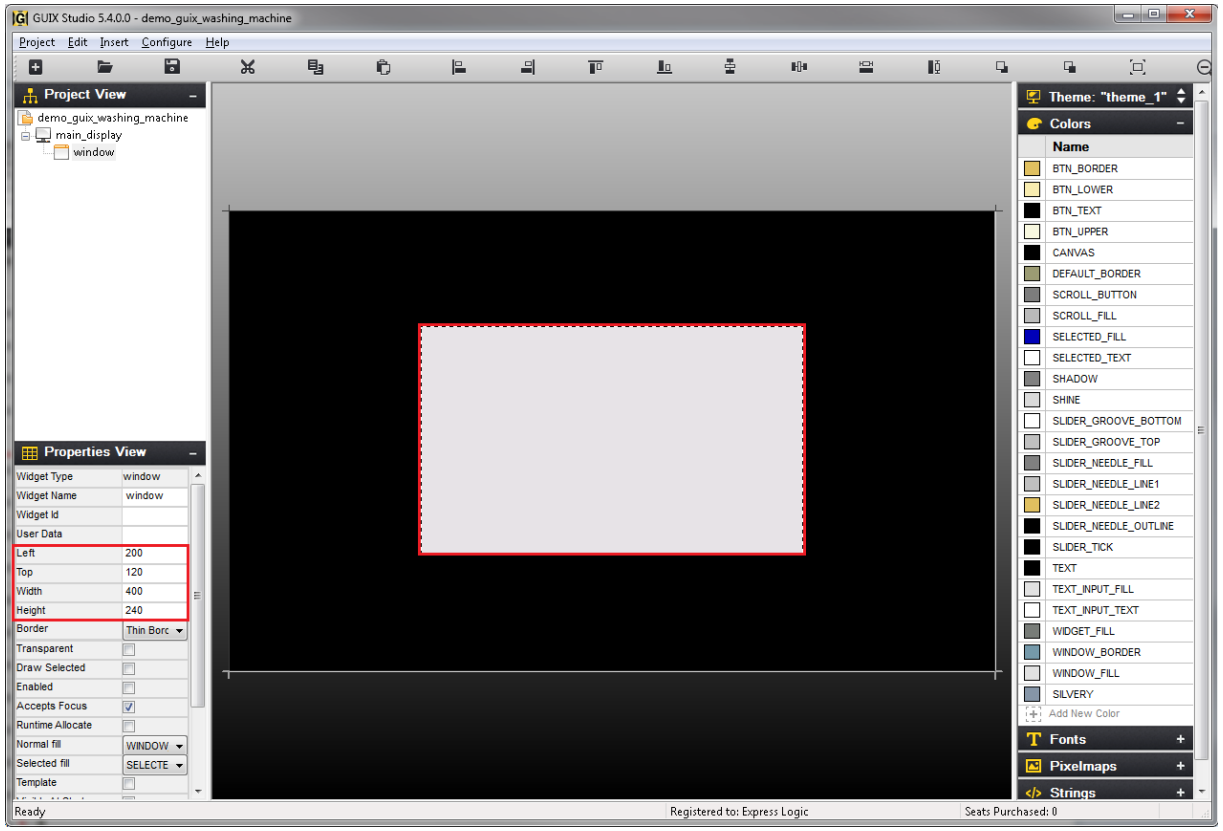


Figure 13. Changing the Properties of New Window

For the Washing Machine Application, to resize the window to occupy the whole screen, you can change the Height, Width, Left, and Top parameters to 480, 800, 0 and 0, respectively. In addition, the Window color needs to be changed. Here, the Washing Machine Application requires the window color as black (background) and it can be changed by modifying the `WINDOW_FILL` color, in the Color section of the GUIX Studio Resources. **Right-click** on **WINDOW_FILL->Edit Color** and set the **Red, Green,** and **Blue** to **0**. The following figure shows a sample snapshot of changing the `WINDOW_FILL`.

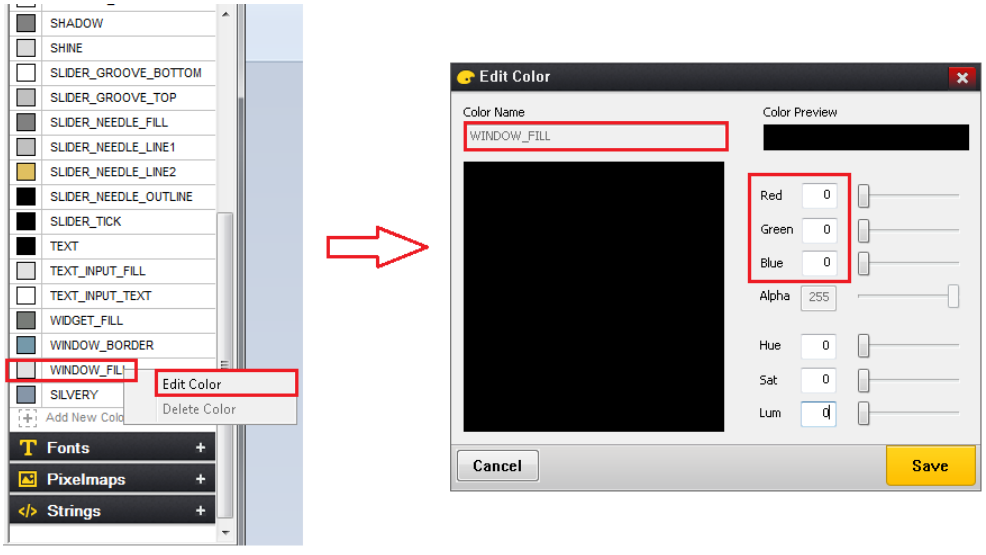


Figure 14. Edit Color of Window Fill

Once the Window properties (left, right, top and bottom) and the `WINDOW_FILL` colors are changed, the following figure shows that the final window looks as required by the application background.

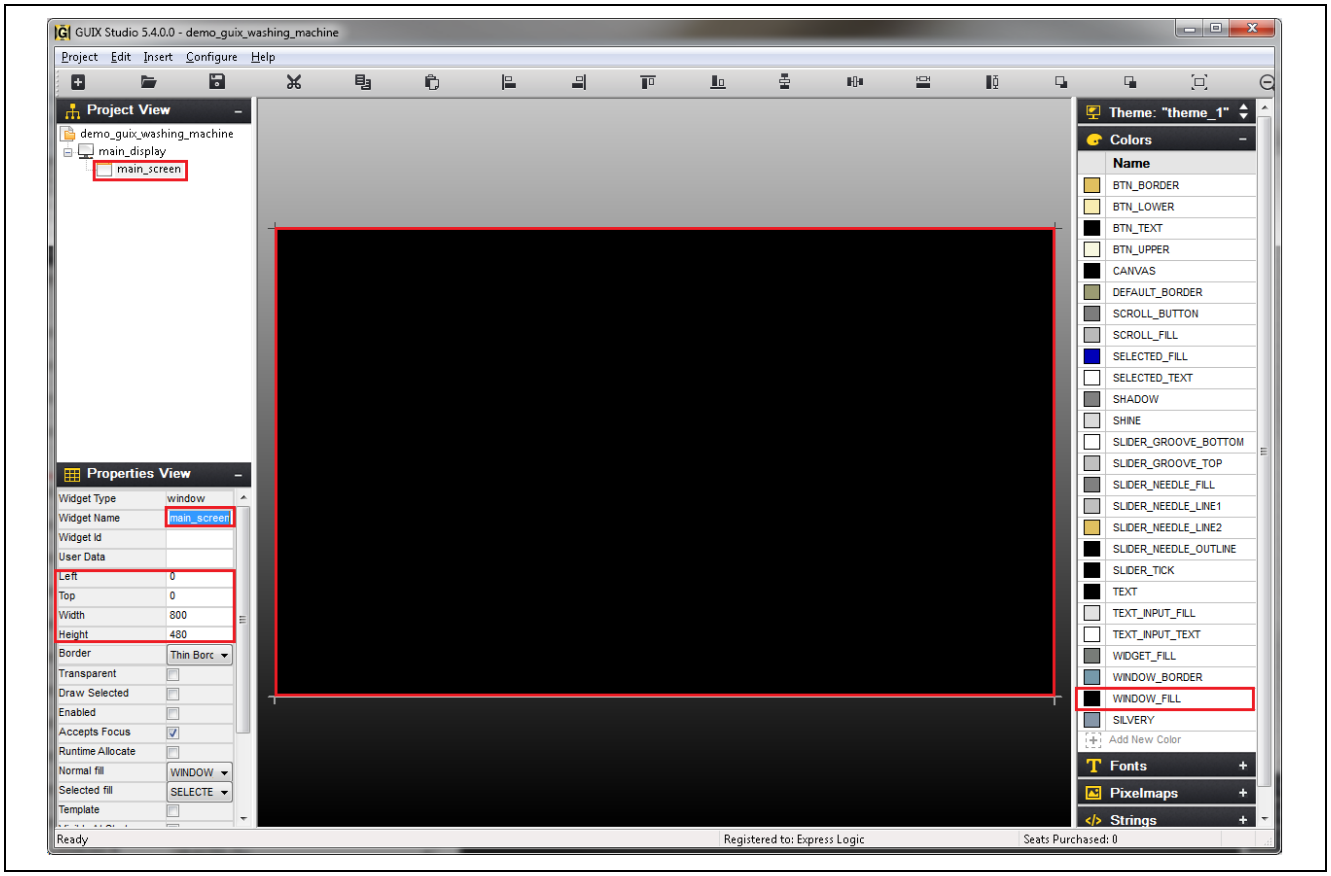


Figure 15. Main Screen Background Window

On top of the background window, you need to add the different widgets for the Washing Machine screens:

- GUIX logo
- Express Logic logo
- Time and date information
- Home button
- Control buttons

Different washer selection radial slider window and the status window, and so on are available.

3.4.3.2 Adding GUIX and Synergy Logo to the Main Screen

To add the GUIX logo and Express Logic logo to the screen, **right-click** on the target window, **Insert > Button > Icon**. The following figure shows how to add the icon widget to the screen.

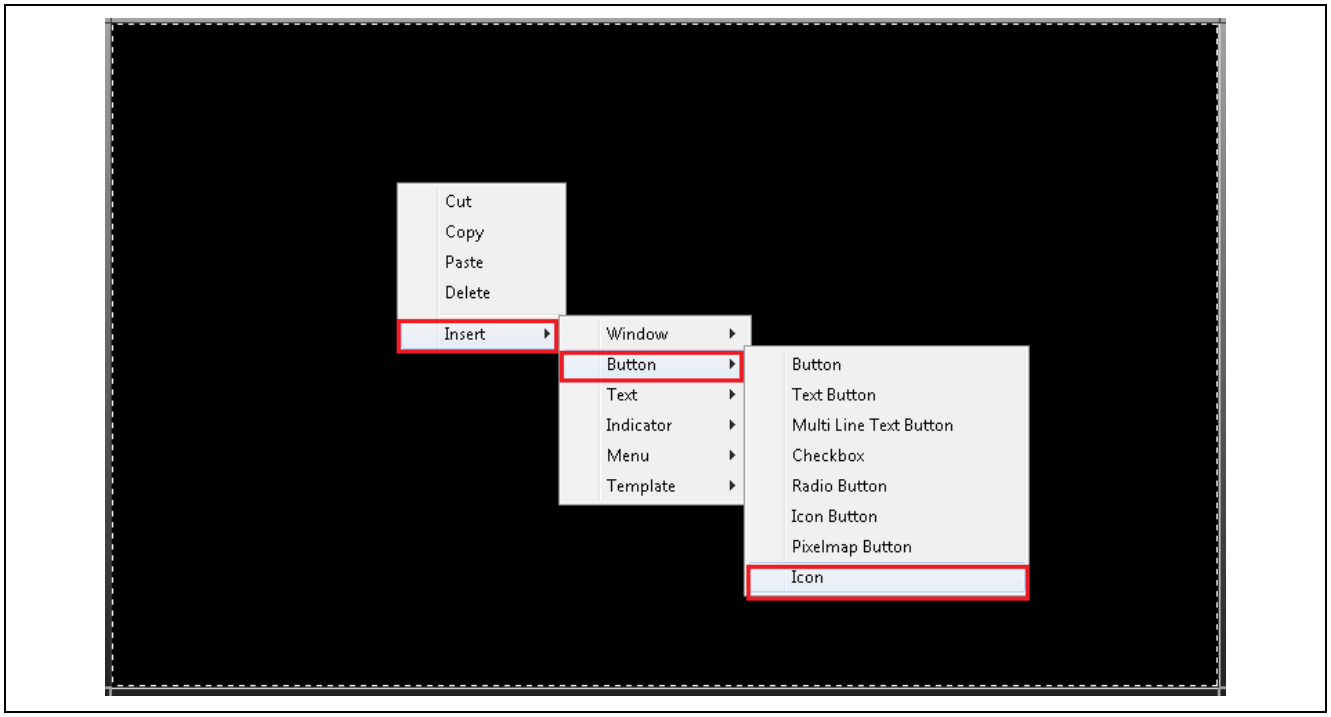


Figure 16. Inserting Button Icon to the window.

This icon is the placeholder for the GUIX logo. From the Pixelmap resources, drag the `GUIX_LOGO_VERTICAL` to the target window and place it on the newly created icon as shown in the following figure.

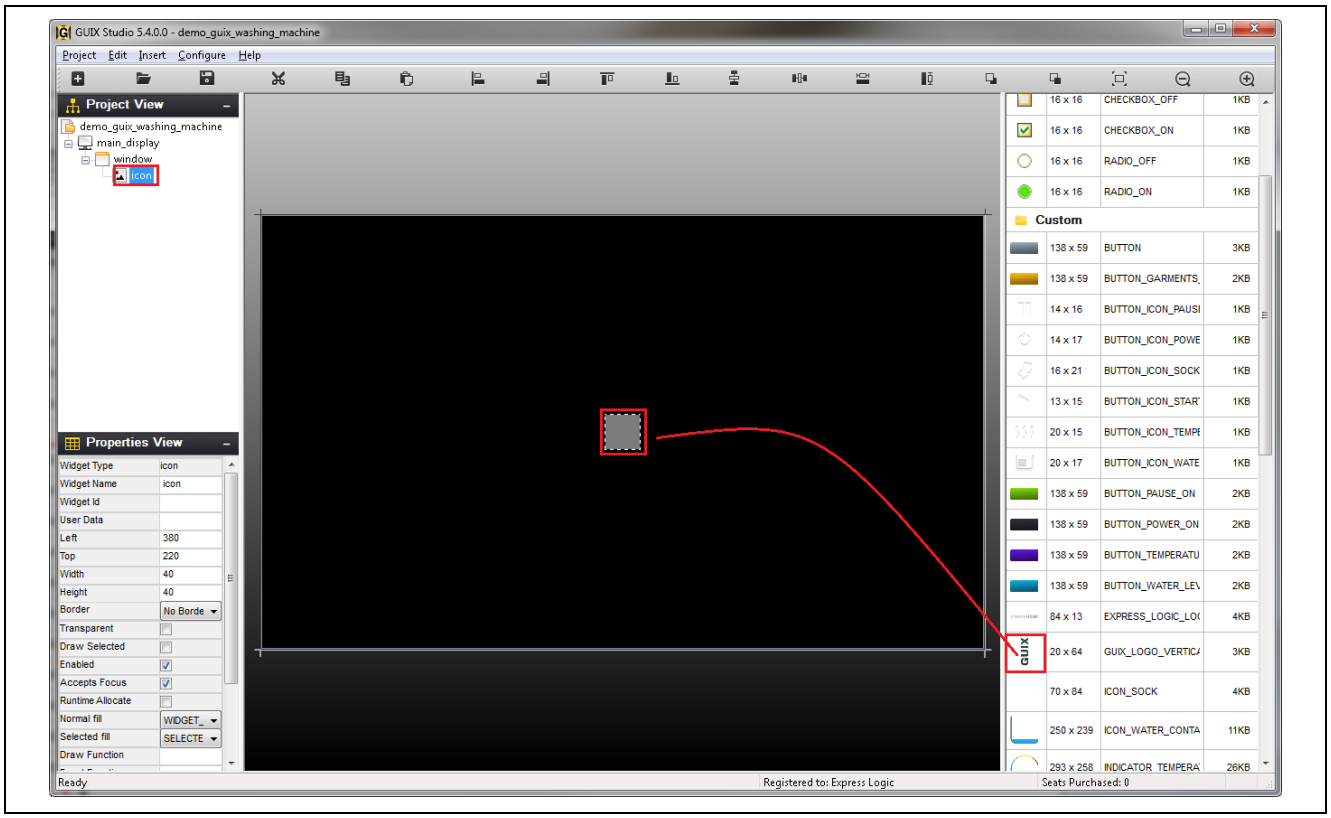


Figure 17. Adding GUIX Logo to the Button Icon.

The following figure shows the GUIX icon appear on the target window. This icon needs to be moved to the top right corner and the icon widget name needs to be changed to guix. Click the newly created icon on the target window and drag the GUIX icon to the top left corner. On the Properties View, change the widget name from icon to **guix**. The following figure shows how the final window appears.

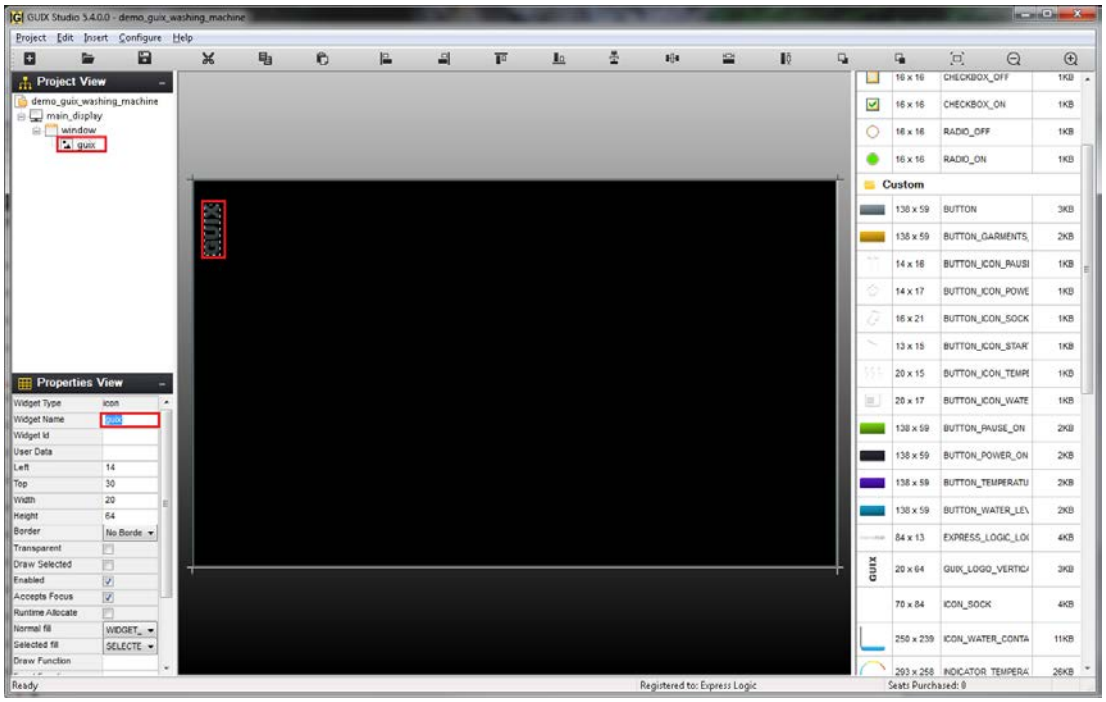


Figure 18. Added GUIX Logo to the Button Icon

Using a similar approach, the Express Logic logo can be added to the screen.

3.4.3.3 Adding Time and Date Information to the Main Screen

The time and date info are added as a text widget to the main screen. To add the text widget, right-click the target window and select **insert > text > prompt**. The following figure shows how this action adds the prompt to the screen.

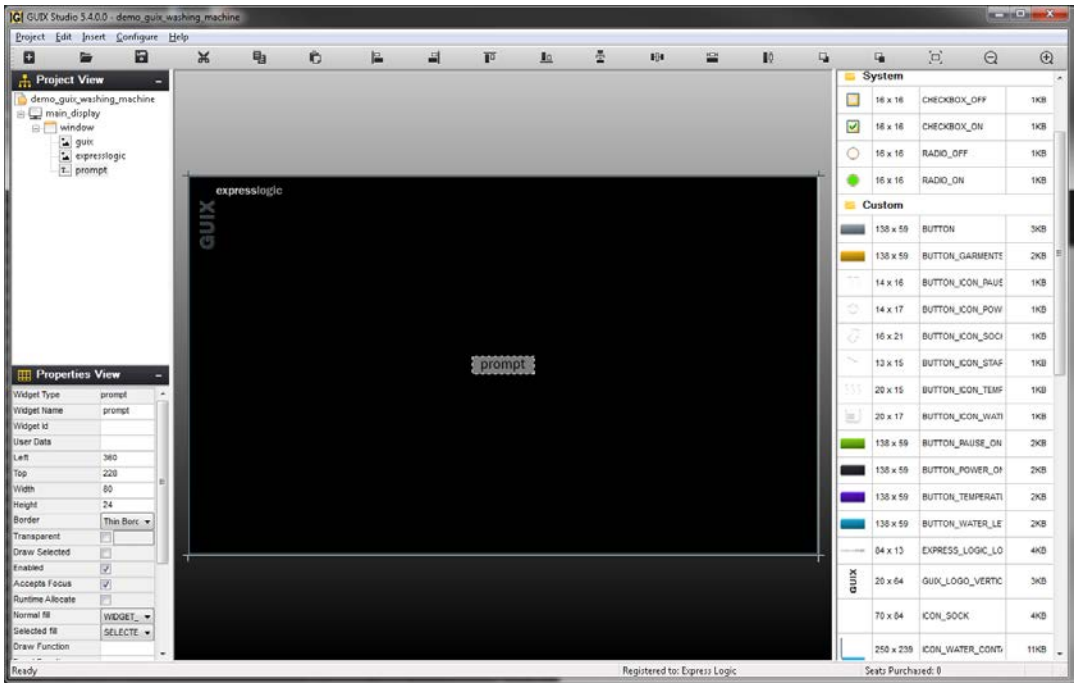


Figure 19. Adding Text Widget to the Main Screen

For the Washing Machine Application, the background of the text prompt (for Time and Date) is transparent without a border and the text color is shown as silver. The position of the time prompt is adjusted to top center. The following figure shows how this is done by changing the selections in the Properties View.

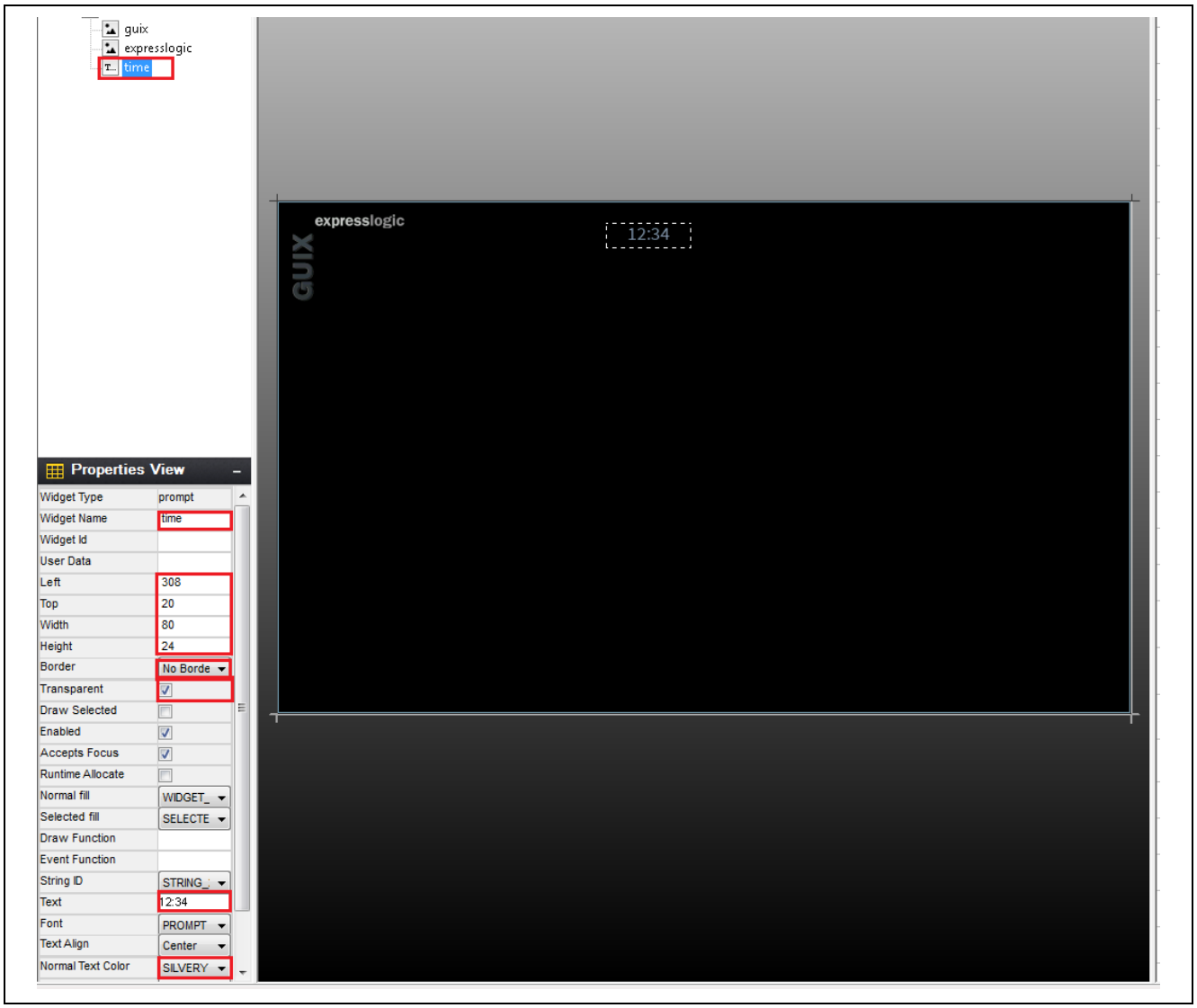


Figure 20. Modifying the Text Widget using Properties View

In a similar way, the AM/PM selection, day of the week, and date related text widgets can be added. Once these text widgets are added, the screen looks as shown in the above figure. For other details, see the attached GUIX Studio Project and its Properties View.

Note: Explaining the addition of the remaining text widget is a repetitive and mechanical step, so you can refer to the adding time widget as a reference.

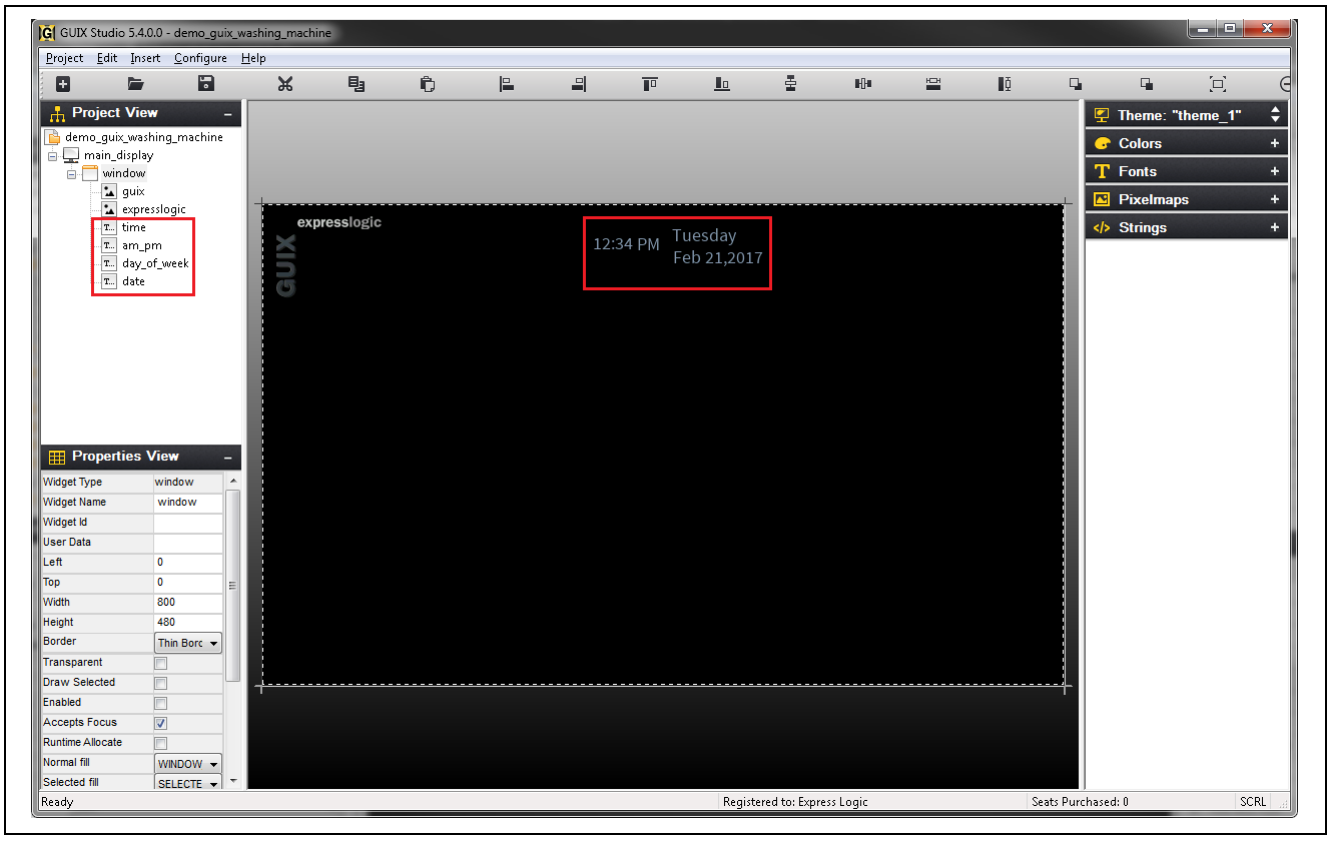


Figure 21. Added Time and Date Text Widget to the Main Screen

3.4.3.4 Adding Home Button to the Main Screen

The Home icon is a pixelmap button and that can be added to the target screen by right-clicking **Insert > Button > Pixelmap Button**.

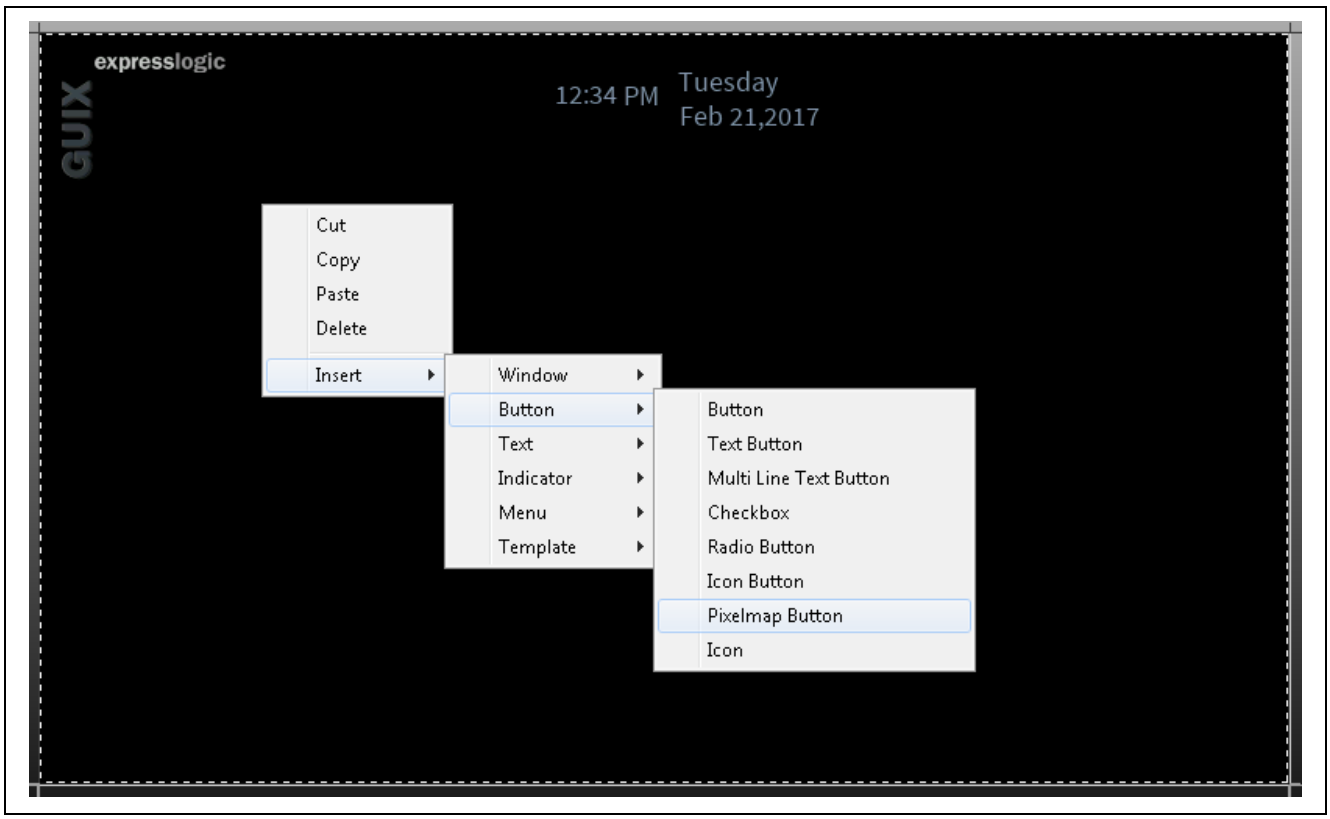


Figure 22. Adding Pixelmap Button to the Main Screen

For the created button, drag the MENU_ICON_HOME from the pixelmap resources and place it on the pixelmap button. Move the icon to the top right corner of the screen. The following figure shows how the resulting screen looks.

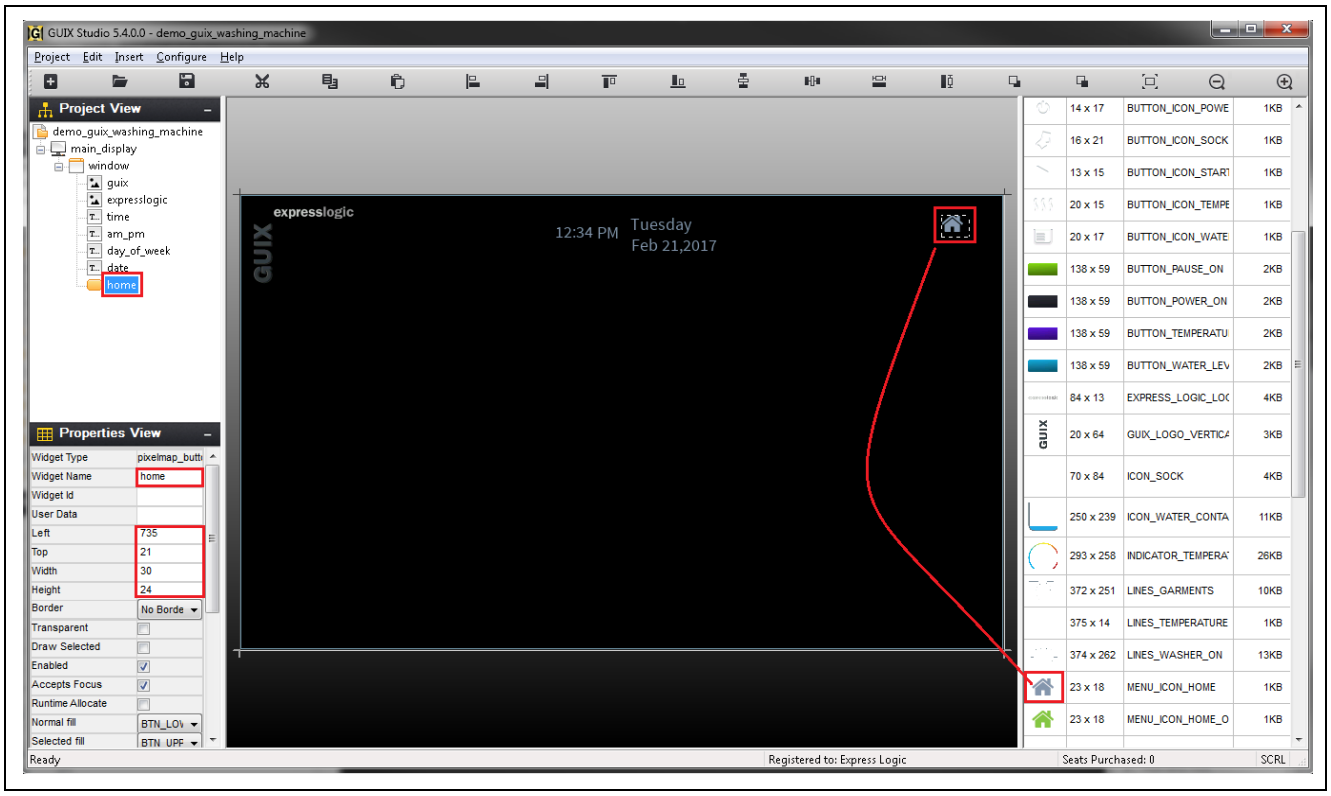


Figure 23. Adding Home Icon to the Pixelmap Button

3.4.3.5 Adding Buttons to the Main Screen

This section covers how to add buttons for selecting the washer, garments, water level, temperature and power on/off.

Washer Button is a Pixelmap Button with a pause icon inside and **pause** text. To add this button, **right-click** on the **target window > Insert > Button > Pixelmap button**. This adds the yellow looking button on the Target View. Go to the **Properties View** and change the **Normal Pixelmap** field to **BUTTON**. Change the **Selected Pixelmap** field to BUTTON_PAUSE_ON. Also, turn on the check boxes for **Pushed** and **Radio** fields. This button is going to come with initial push state by displaying the selected pixelmap instead of the normal pixelmap. Since this is selected as a radio button, when this button is selected, all other buttons are deselected. Go to the button bar of the Studio and click the size to content, and this will bring the button to the actual size. Move the button to the right side of the screen where it is intended to be placed. Click the **created button > Insert > Button > Icon**.

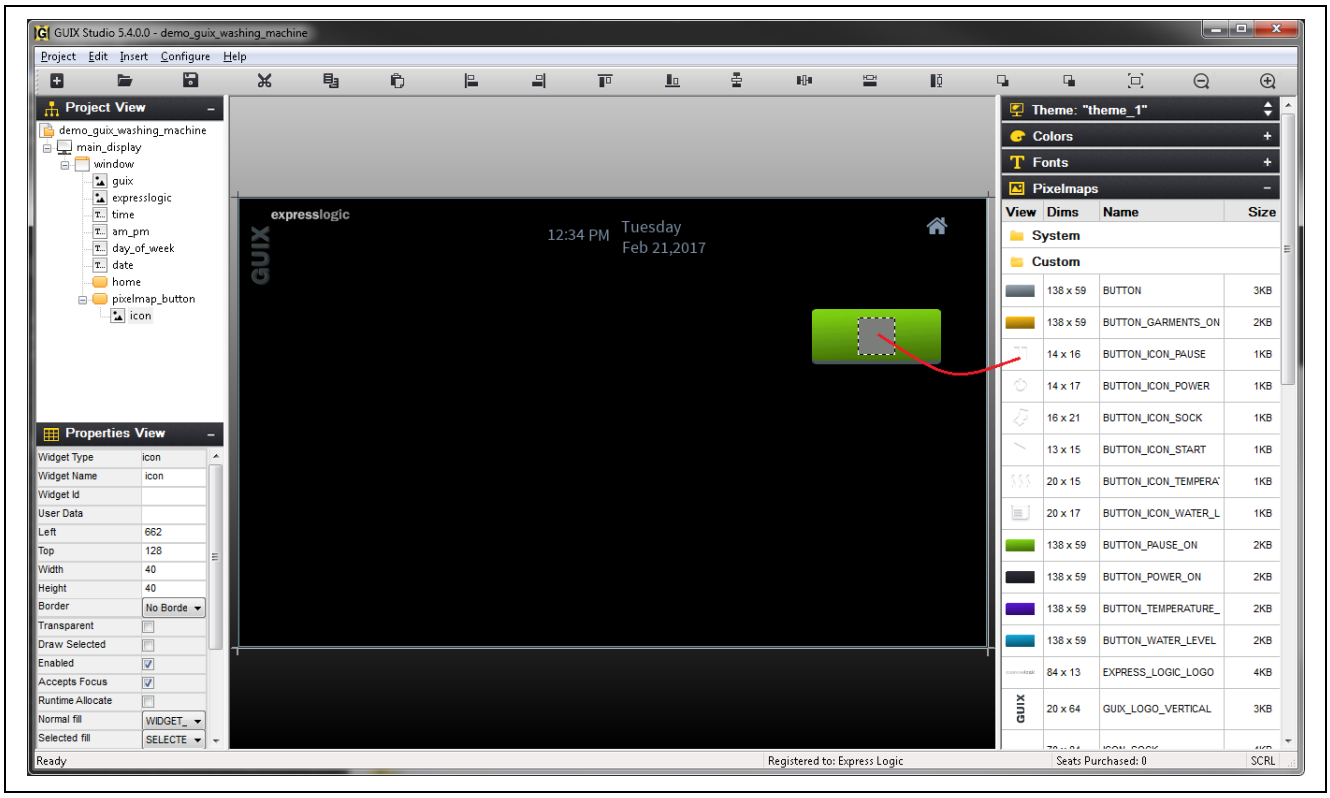


Figure 24. Adding Pause Icon to the Pixelmap Button

Drag the `BUTTON_ICON_PAUSE` and place on top of the icon created as shown in the above screen. Add the pause prompt to the button by **right-clicking** the top of the **button > Insert > text > prompt**.

Go to the **Properties View** of the prompt and change the text field to **pause**, **transparent field** selected, **border field** to **no border**. Select the **Normal text color** as **WHITE** (If the color is not found, create one).

The following figure shows the **Properties View** snapshot for **Washer Button, Pause Icon,** and the **Pause Prompt**.

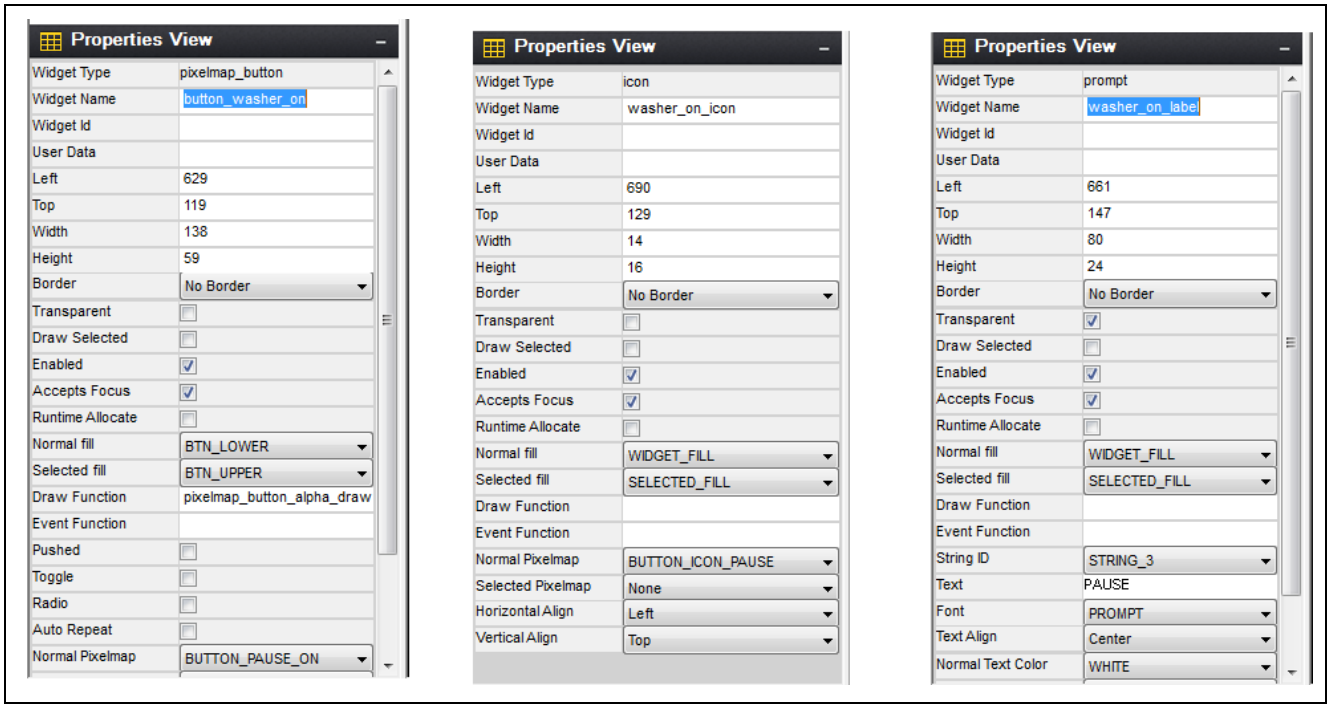


Figure 25. Properties View – Washer Button, Pause Icon, Washer Prompt

After modifying the Properties View for the Washer Button, the Washer Button on the target window looks as shown in the following figure.

Figure 26. Washer Button Added to the Main Screen

In a similar manner, the remaining buttons can be added to the main screen. Refer to the attached GUIX Studio Project for more details on the Properties View for the individual buttons.

3.4.3.6 Adding Washer Window to the Main Screen

Washer window is the child window of the main screen composed of circular wheel (radial slider) and the multiple lines for the prompt. These two icons are superimposed to create the washer window. On top of each prompt line, a text prompt is created as a label for the different washer settings.

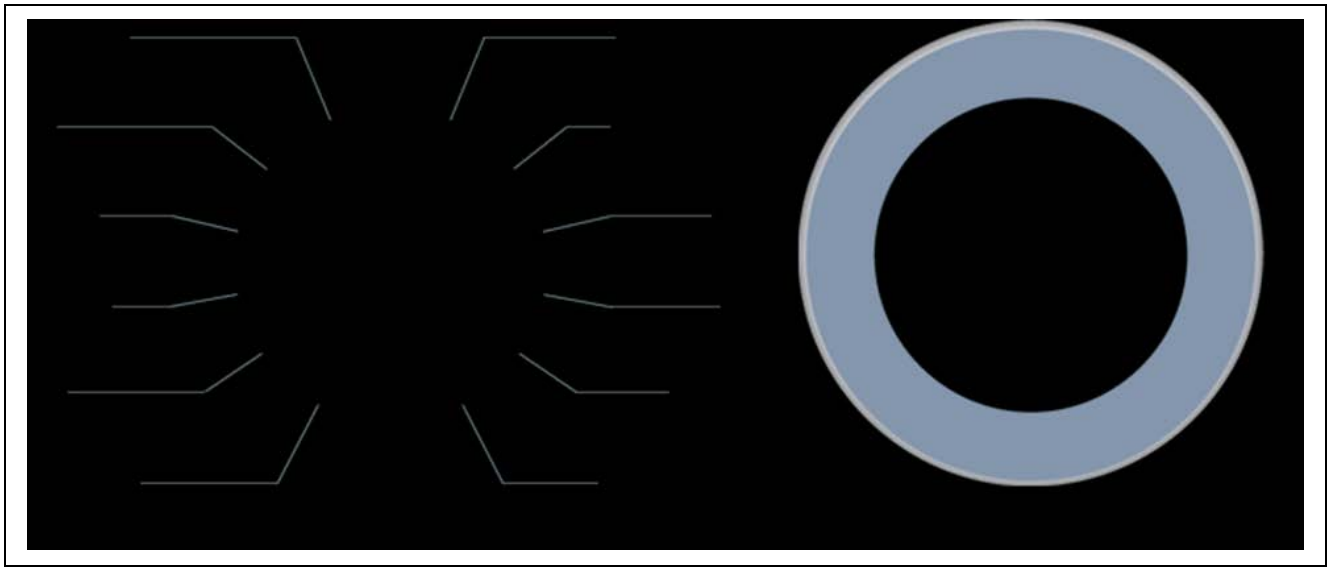


Figure 27. Radial Slider Components

Snapshot in the above screen shows the separate icons, and these are superimposed on the GUIX Studio, along with the text prompts to create the final washer window.

Note: This child window (Washer Window) is attached and detached based on the button selection for buttons such as garments, water level, and temperature. The Washing Machine Application is designed in this way, but this can be implemented in different ways using the GUIX Studio templates in the latest versions.

Creating the Washer Control Interface is similar to adding the two different icons under the Washer Window and adding the child widgets as text prompts. The Pixelmaps for these icons are available as part of the resource section under the Pixelmaps. The details of adding this interface to the Studio is not covered in detail. Refer to the previous section to add these widgets. For other details, see the GUIX Project attached with this application note.

Figure 28. Radial Slider Window after the Creation

The following figure shows the Properties View snapshot for the Washer window (radial slider icon widget (WHEEL) and Prompt line icon (`LINES_WASHER_ON`).

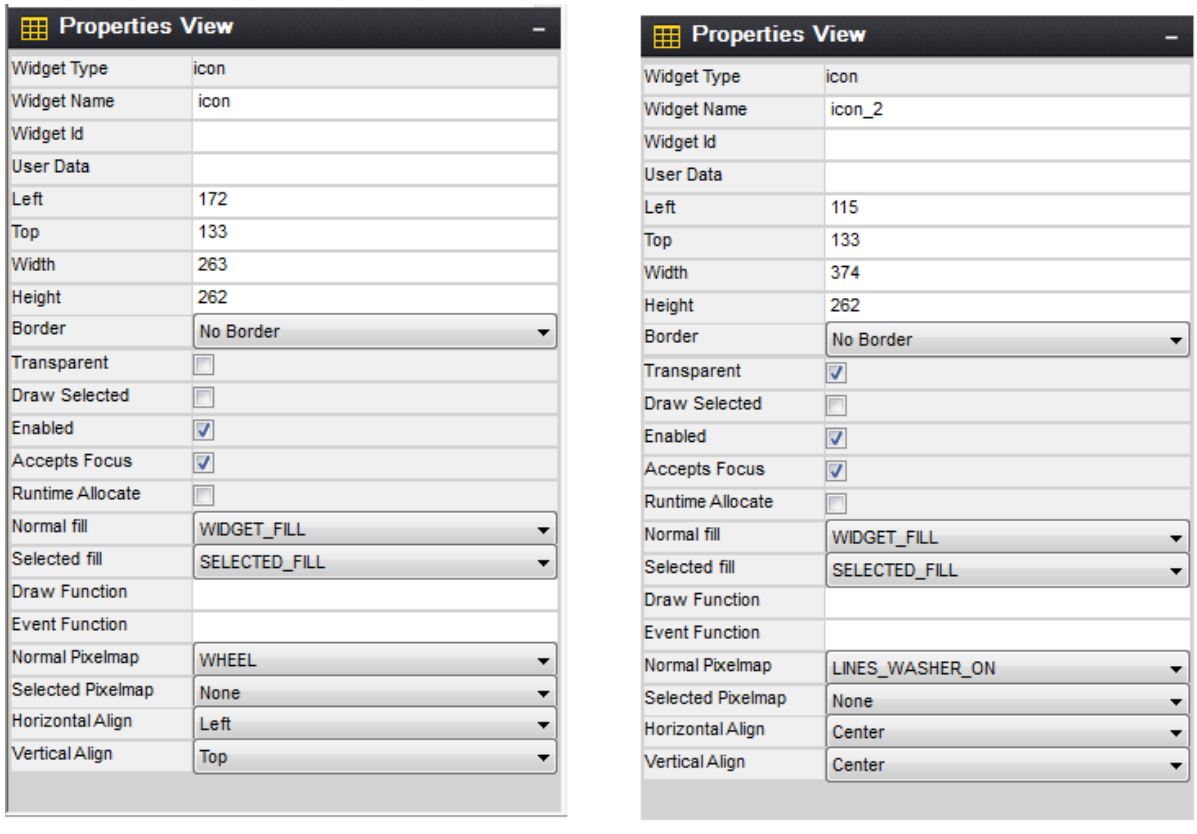


Figure 29. Radial Slider Properties View

3.4.3.7 Adding Slider Bar to the Main Screen

Slider bar is a status bar to show the different wash levels. To add the slider bar to the main screen, `Pixelmap_slider` widget is used with the progress bar icon, with different wash levels. The status update on the status bar is controlled through the draw function `pixelmap_slider_alpha_draw`. The following figure shows the individual components of the slider bar.

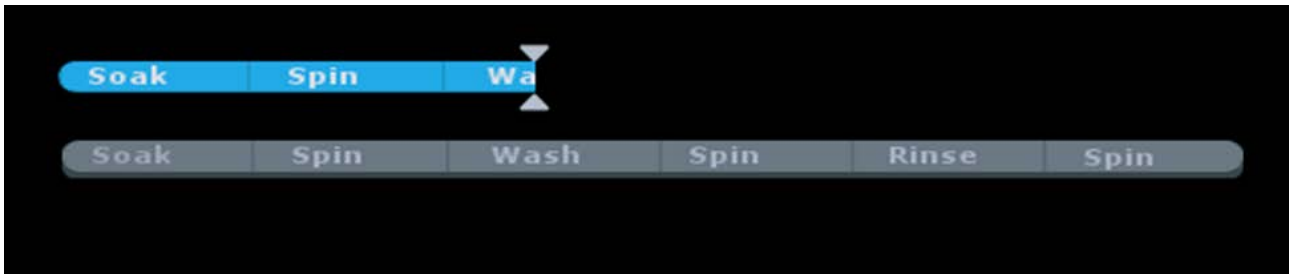


Figure 30. Slider Bar Components

The following figure shows the Properties View for the Pixelmap Slider and the Status Bar Icon. See the GUIX Studio Project for other details.

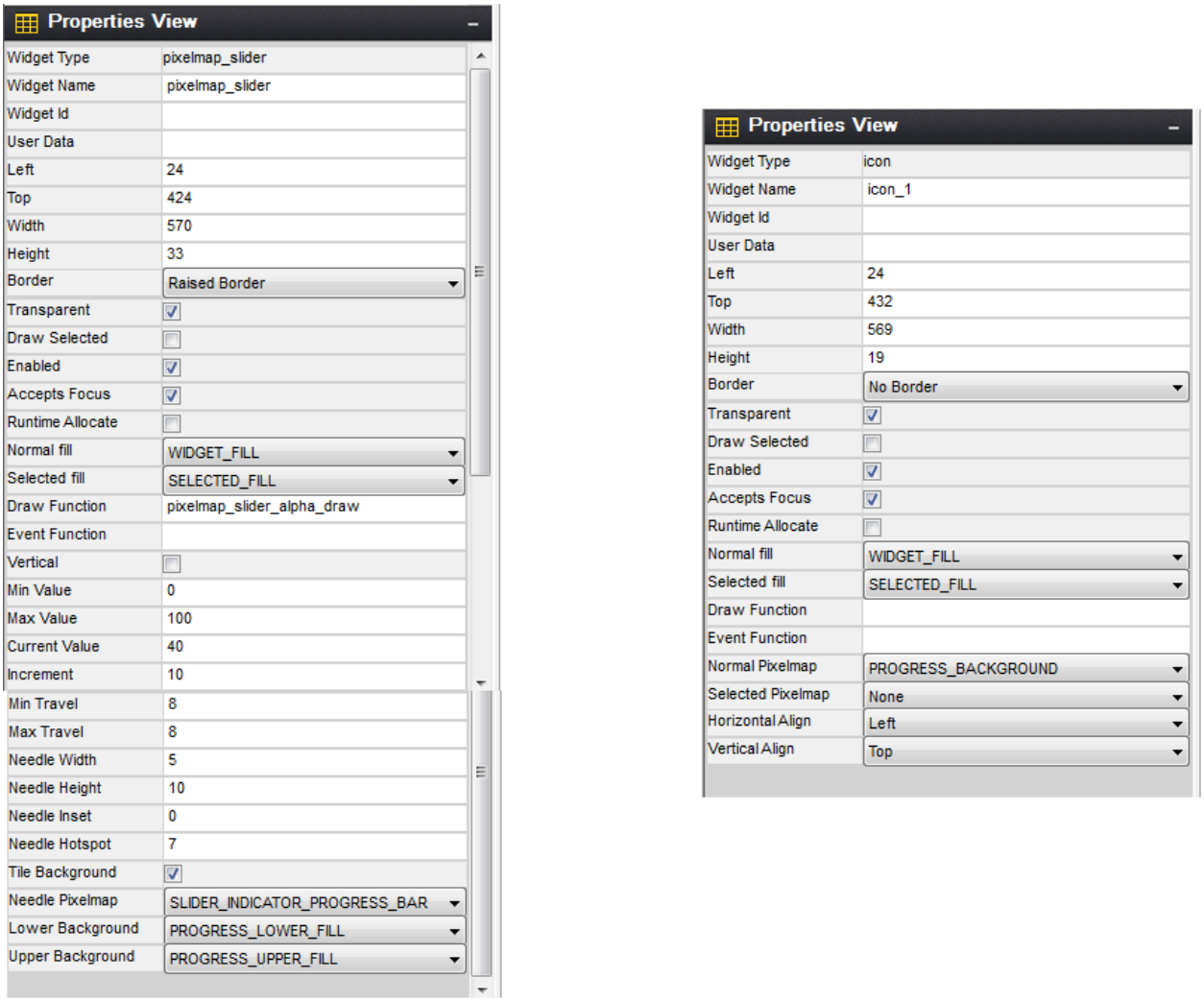


Figure 31. Slider Bar Properties View

3.4.3.8 Adding the Screen Name to the Main Screen

The page name or screen name is the widget type text prompt with the draw function `prompt_alpha_draw` to dynamically update the screen when different screens are selected. For the main screen, the default text displays as **Washer ON**.

The Properties View for the screen name is shown in the following figure.

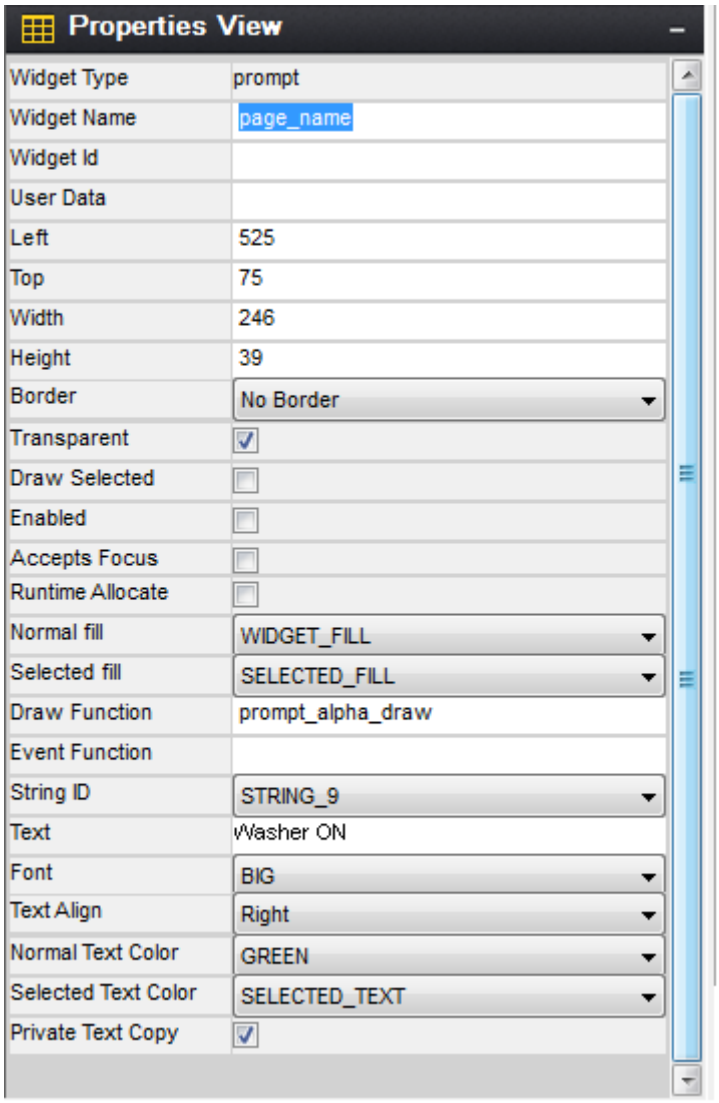


Figure 32. Slider Bar Properties View

With all the creation steps and guidelines explained in previous sections, you should have the main screen created as shown in the following figure. If not, revisit the earlier sections, reference documents, and the GUIX Project for more details.

Figure 33. Main Screen View With the Widgets Added

3.4.4 Creating Garments Window

Garments Window is a child's window of the main screen that is attached and detached based on the Garments Button selection or deselection. This window is like the Washer Window explained in the previous section. It has the Sock icon inside as status, that is controlled via the animation using the code. You can refer to the attached GUIX Studio Project for more details to create this Garments Window as this is like the Washer Window.

Garments window is a child of the Main screen composed of a circular wheel (radial slider) and the multiple lines (icon) for the prompt. These two icons are superimposed to create the Garments Window. On top of each line, the text prompt is created to prompt the Washer label on the Garments Window. The following figure shows the separate icons, these are superimposed on GUIX Studio, with the text prompts to create the final Garments Window.

Figure 34. Radial Slider Components

Once the Garments Window is created, the screen should look like the snapshot in the following figure.

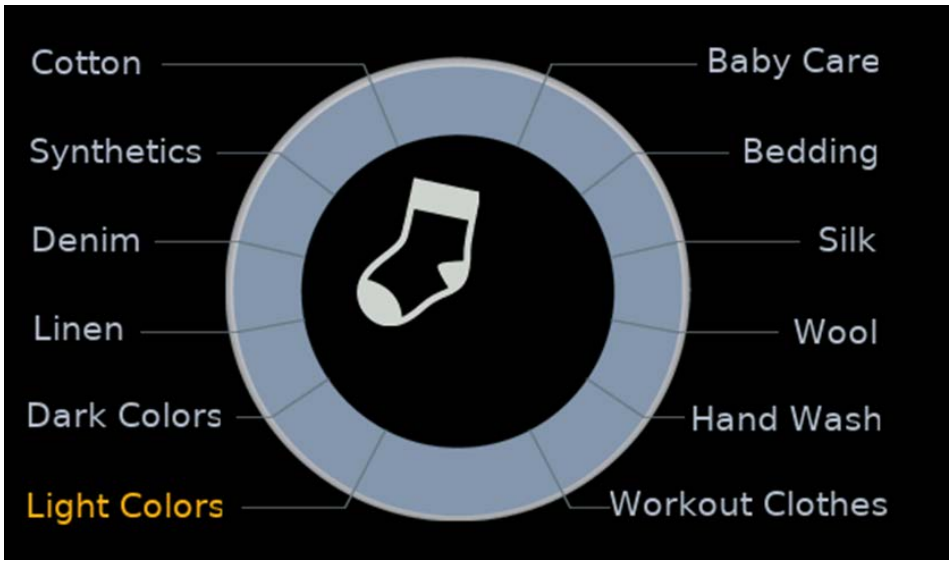


Figure 35. Garments Window with the Widgets Added

The Properties View of the Garments Window is shown in the following figure. The Draw function `window_alpha_draw` and the event function `garments_window_event_function` handles the drawing of the window and is even specific to the Garments Window. You can refer the `garments.c` in the `/src` folder to get a better understanding of the functionality.

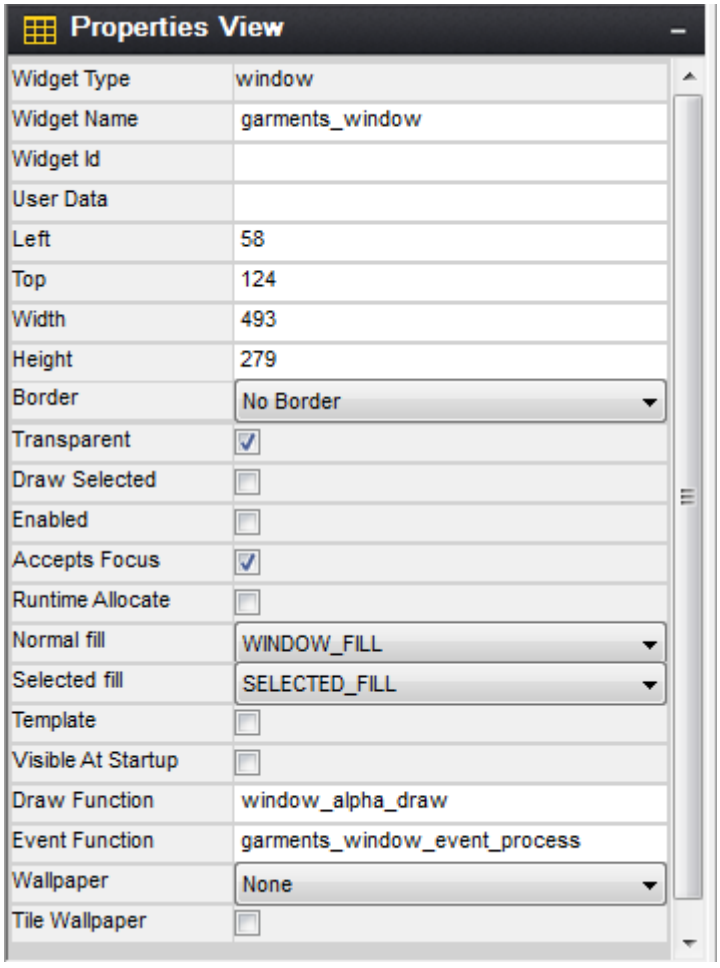


Figure 36. Properties View of Garments Window

3.4.5 Creating Temperature Window

Temperature Window is created by using the circular wheel (radial slider) with the arch (with blue to red variations) to showcase the cold and hot temperature of the water. Temperature Window has the text widget to show the water temperature from cold to hot and status widget inside the circular wheel to show the actual temperature. The individual icons for the Temperature Window are shown in the following figure.

This window is a child's window that can be attached or detached based on the selection or deselection of the Temperature Button. This window has the slider pen (round ball shaped) that is controlled by code as animation. This window is created using the superimposition of the two icons with the text widgets.

The Properties View for the Temperature Window is shown in the following figure. For more details, refer to the attached GUIX Studio Project, and the source code for understanding the draw function `temperature_window_draw` and event function `temperature_window_event_process`.

Figure 37. Components of Temperature Window

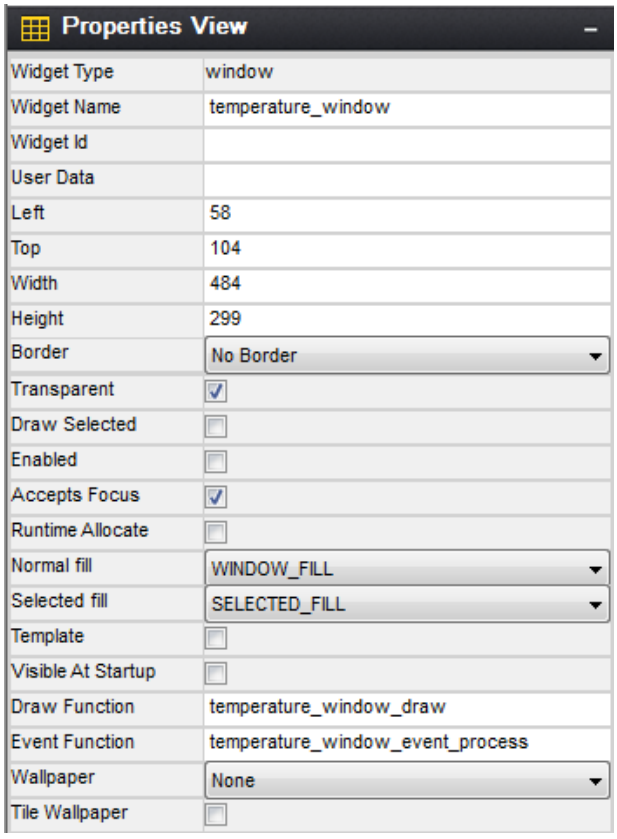


Figure 38. Properties View - Temperature Window

Once the Temperature Window is created, it should look like the following figure.

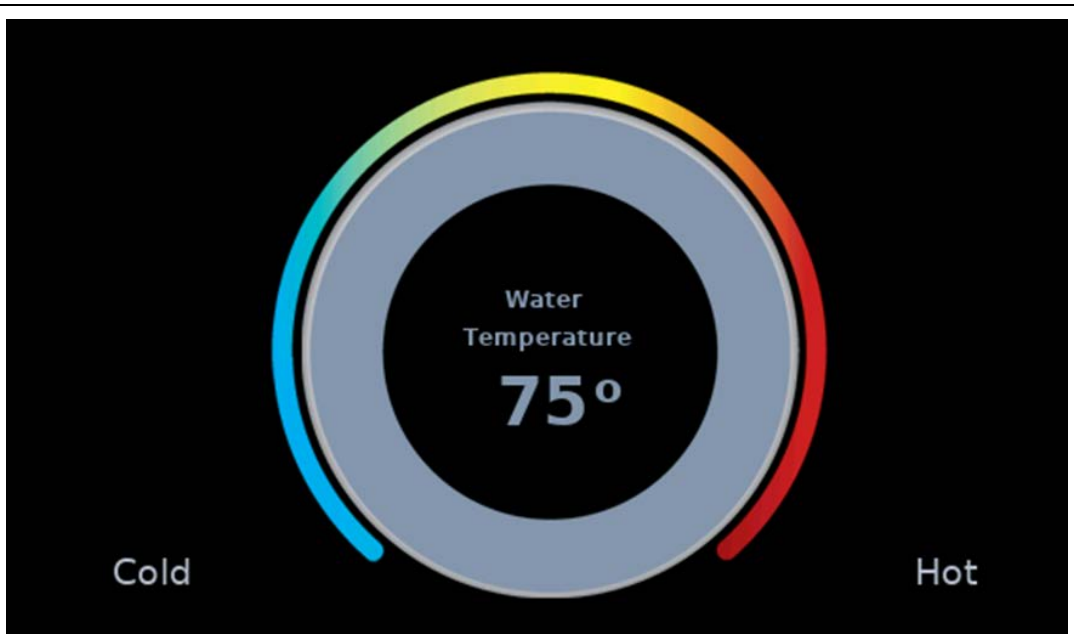


Figure 39. Temperature Window with Widgets Added

3.5 Creating Water Level Window

Water Level Window is the child window that can be attached or detached based on the Water Level Selection Button. This is different when compared to the Washer or Temperature window. This window is implemented using the superimposition of the Icon widget and the Pixelmap slider to create a complete icon. This window also has the text widgets to show the different water levels.

The individual icon and Pixel slider are shown in the following figure.

Figure 40. Components of Water Level Window

Water Level Window has the event function `water_level_window_event_process` to handle the Events and Draw function `water_level_window_draw` that handles the drawing of the different water levels when the slider is moved.

To create the Water Level Window, see the attached GUIX Studio Project for details on individual widgets. The Properties View for the Water Level Window is shown in the following figure.

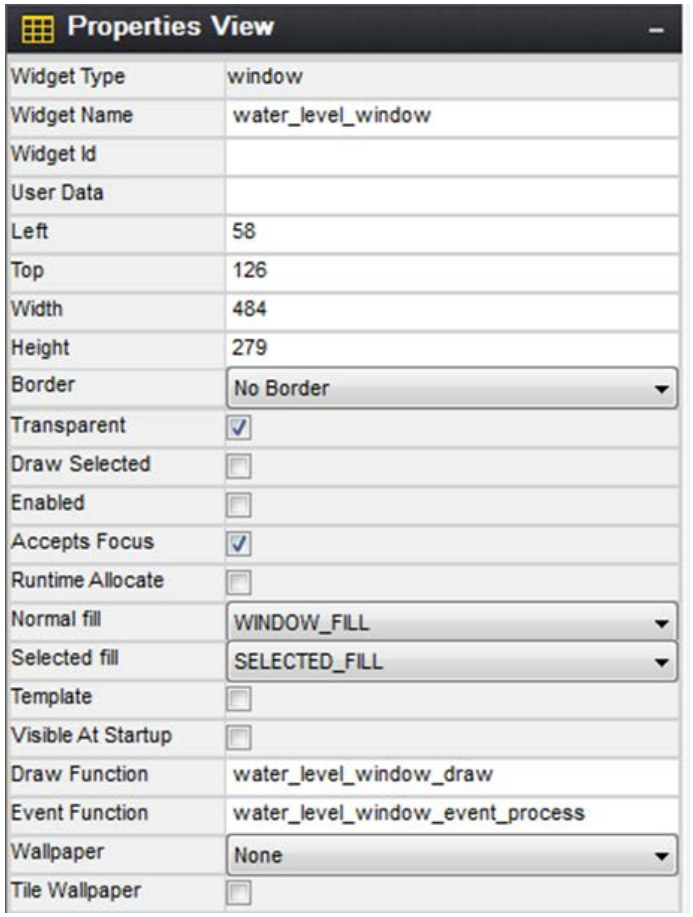


Figure 41. Properties View of Water Level Window

Once all widgets for the Water Level Window are created, the following figure shows what the final screen should look like.

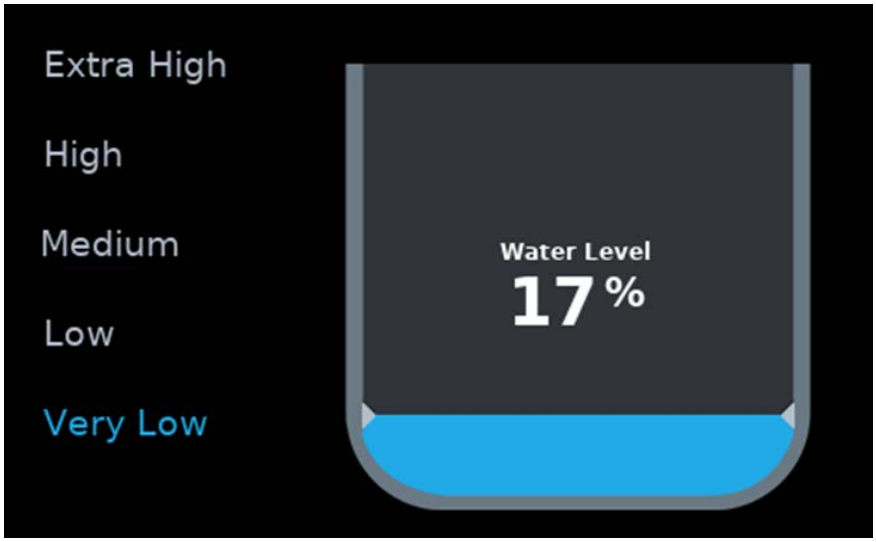


Figure 42. Water Level Window with Widgets Added

3.5.1 Resource and Specification Files from the Created Project

Once all the screens are created using the GUIX Studio as your project requires, you can use GUIX Studio options to generate resource and specification files. These resource files are for newly added screens, graphic images, data structures, as well as related draw and event handler functions in the form of ANSI C format code. These resource and specification files are input to the Application Project. The following figure shows how resource and specification files are generated using GUIX Studio.

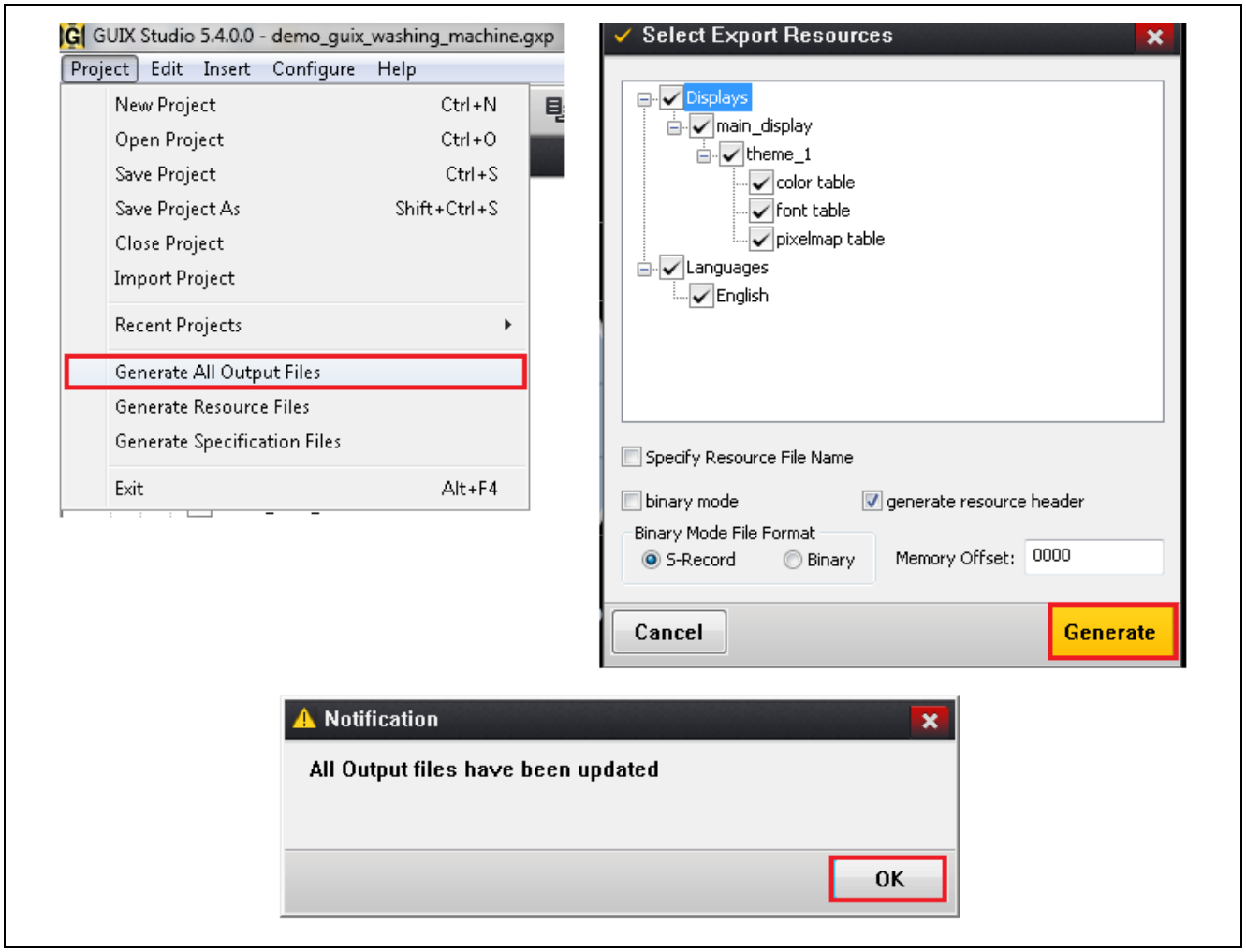


Figure 43. Resource File Generation

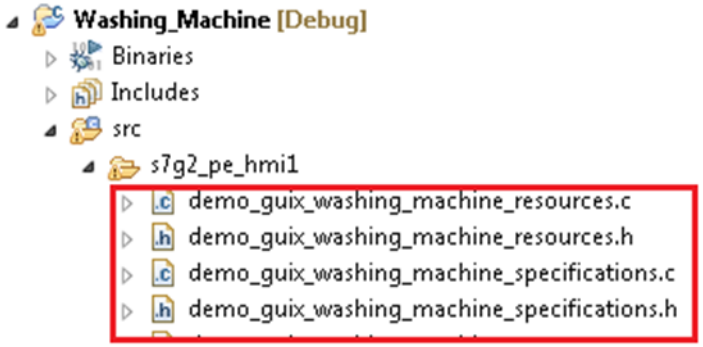


Figure 44. Resource File in the Project View

The generated codes are compiled with the application code to generate the final image for the target. The previous figure shows a sample snapshot of the resource and specification files used in the Washing Machine Application Project.

3.6 Including and Configuring the GUIX Module in an Application

- Create a new thread by clicking **New Thread** in the **Threads** area.

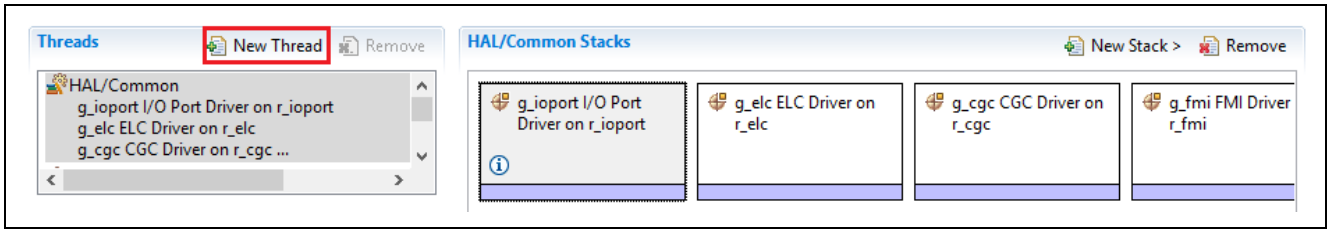


Figure 45. Create a New Thread

- Click on **New Thread** to show the properties.
- Edit the **Properties** to match the following figure.

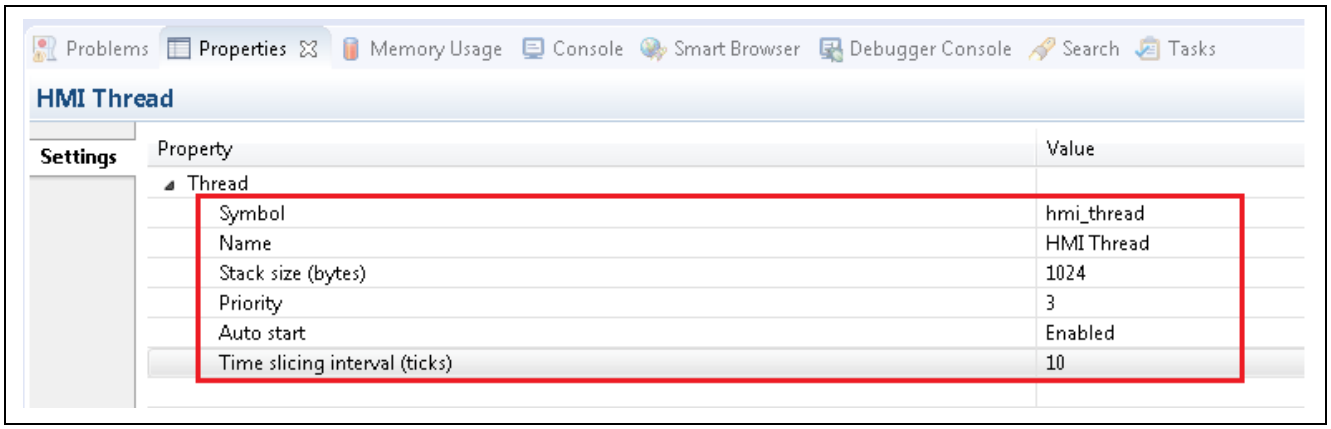


Figure 46. Configure HMI Thread Properties

- From the Synergy **Configuration Window > Threads tab > Main Thread Stacks** area and **click** on **New Stack**.

Note: Be sure that the **HMI Thread** is selected before adding new modules.

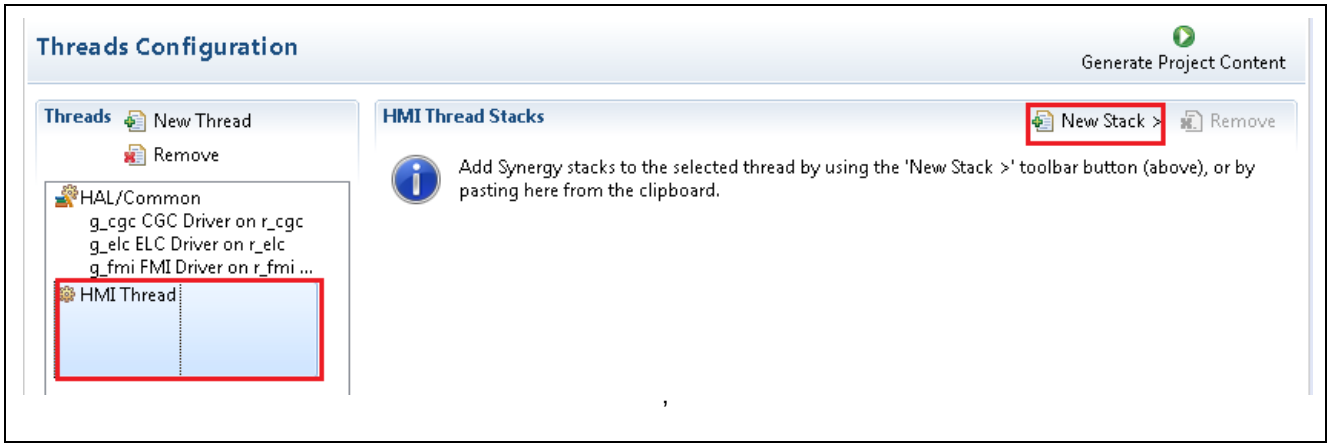


Figure 47. HMI Thread Stacks

- In the Synergy Configuration Window **> Threads tab > HMI Thread Stacks** area, add a framework for the Touch Panel by selecting **New Stack**, then **Framework > Input > Touch Panel Framework** on `sf_touch_panel_i2c`.

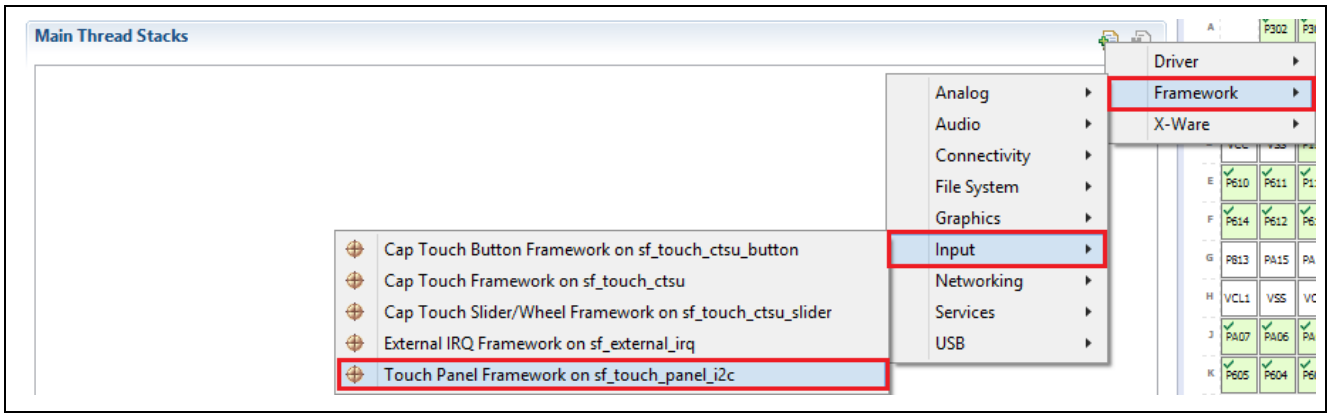


Figure 48. Adding Touch Panel Framework

- Configure the following Properties.

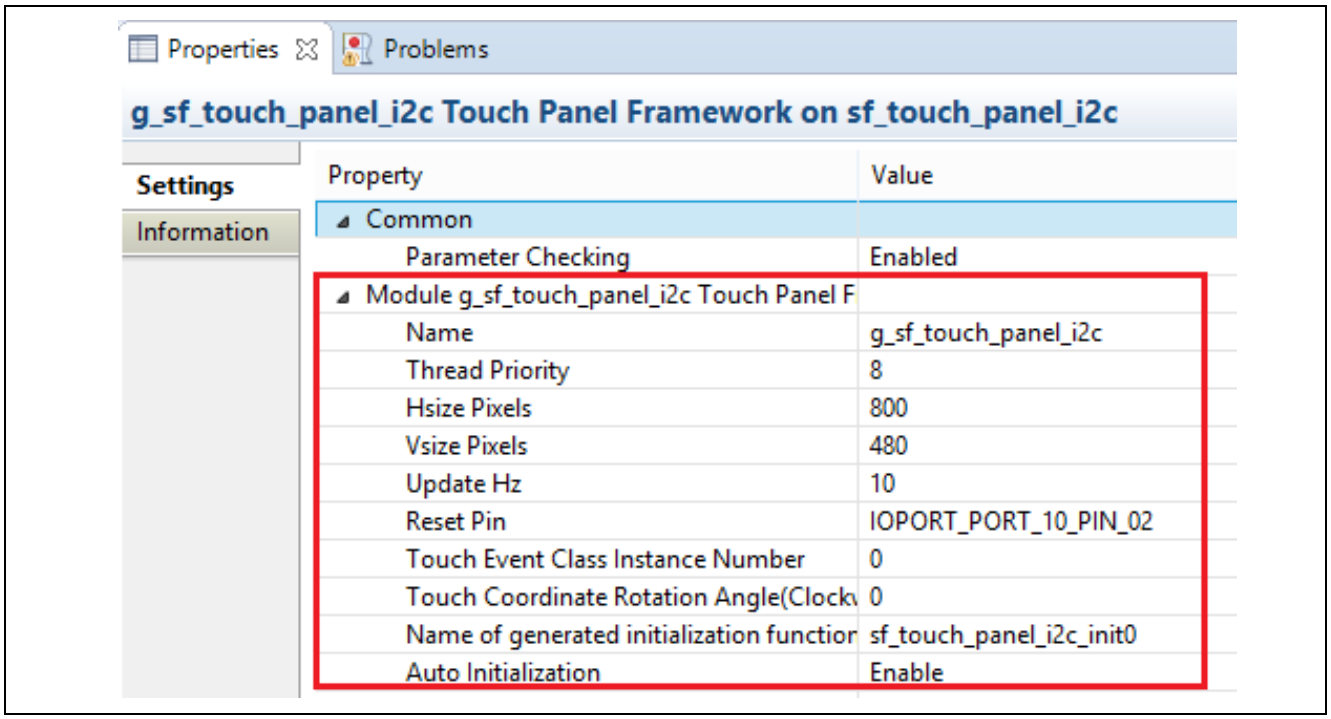


Figure 49. Configuring Touch Panel Properties

Notice that in the following figure, the Synergy Configurator has already created the message framework, external IRQ framework, and has a placeholder for the external IRQ and I²C driver stacks.

The messaging framework is used by other framework layers and tasks to pass messages around the system. This system passes data from the touch screen driver to the HMI Thread to handle touch inputs.

The following figure shows the SF External Interrupt, a framework layer used by the Touch Controller Driver.

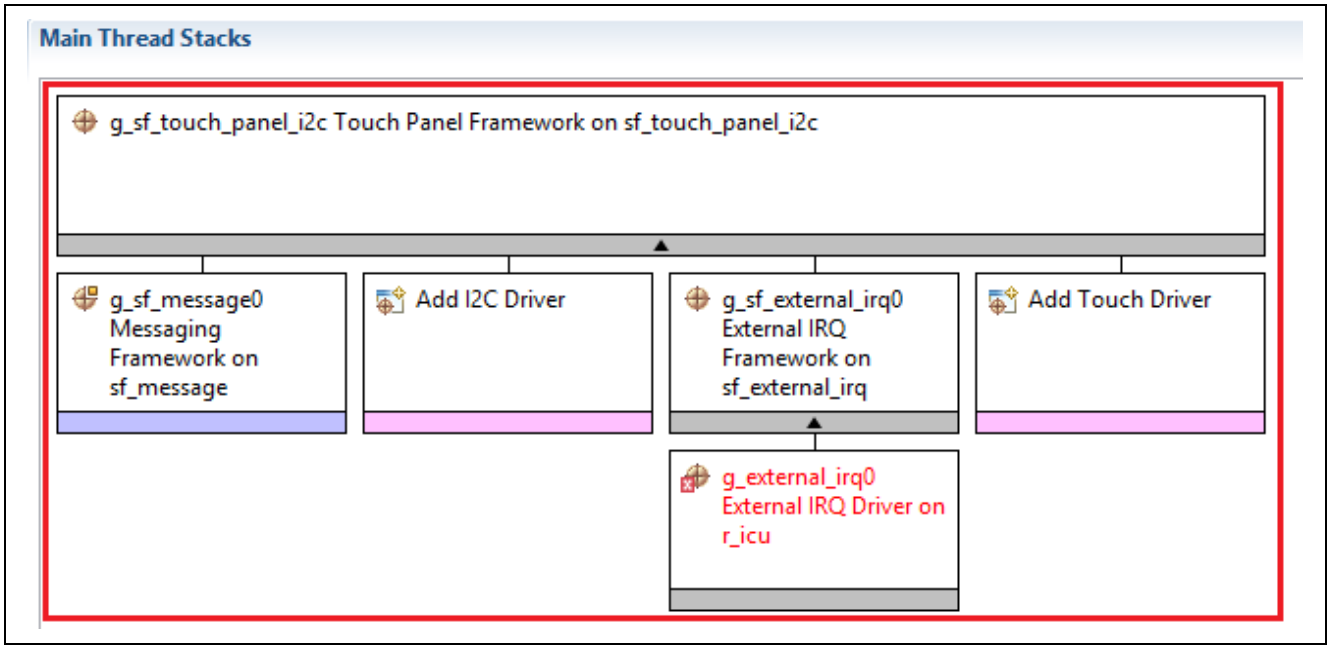
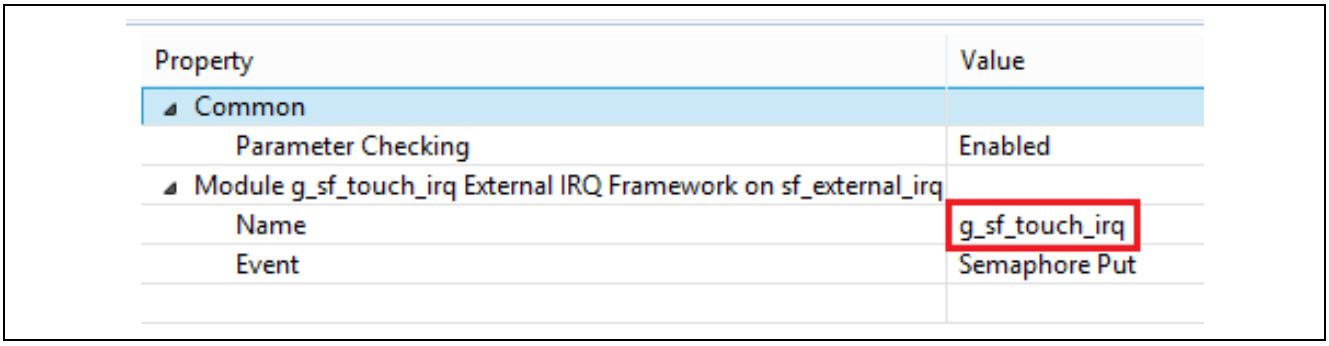


Figure 50. Touch Panel Framework Stack

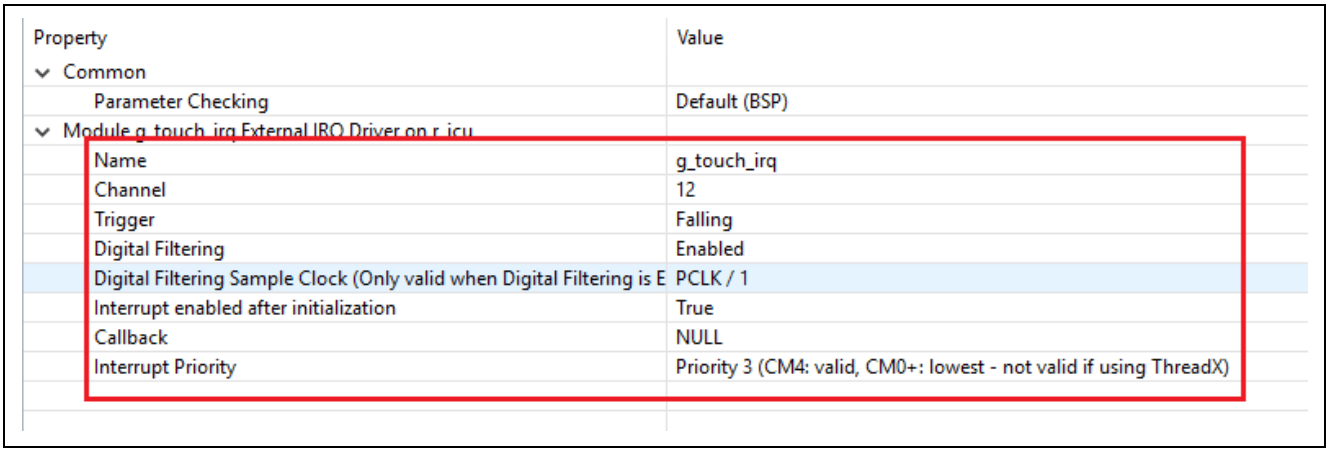
- Select the **External IRQ Framework on sf_external_irq** and configure the following properties.



Property	Value
Common	
Parameter Checking	Enabled
Module g_sf_touch_irq External IRQ Framework on sf_external_irq	
Name	g_sf_touch_irq
Event	Semaphore Put

Figure 51. Configuring External Interrupts Properties

- Select **External IRQ Driver on r_icu**. Configure the following properties for the new module.
  Note: Change the Channel first.



Property	Value
Common	
Parameter Checking	Default (BSP)
Module g_touch_irq External IRQ Driver on r_icu	
Name	g_touch_irq
Channel	12
Trigger	Falling
Digital Filtering	Enabled
Digital Filtering Sample Clock (Only valid when Digital Filtering is E	PCLK / 1
Interrupt enabled after initialization	True
Callback	NULL
Interrupt Priority	Priority 3 (CM4: valid, CM0+: lowest - not valid if using ThreadX)

Figure 52. Touch Screen IRQ Properties

- In the **Synergy Configuration Window > Threads tab > HMI Thread Stacks** area, add a driver for the I²C bus by **right-clicking Add I²C Driver**, and then selecting **New > I²C Master Driver** on `r_iic`.

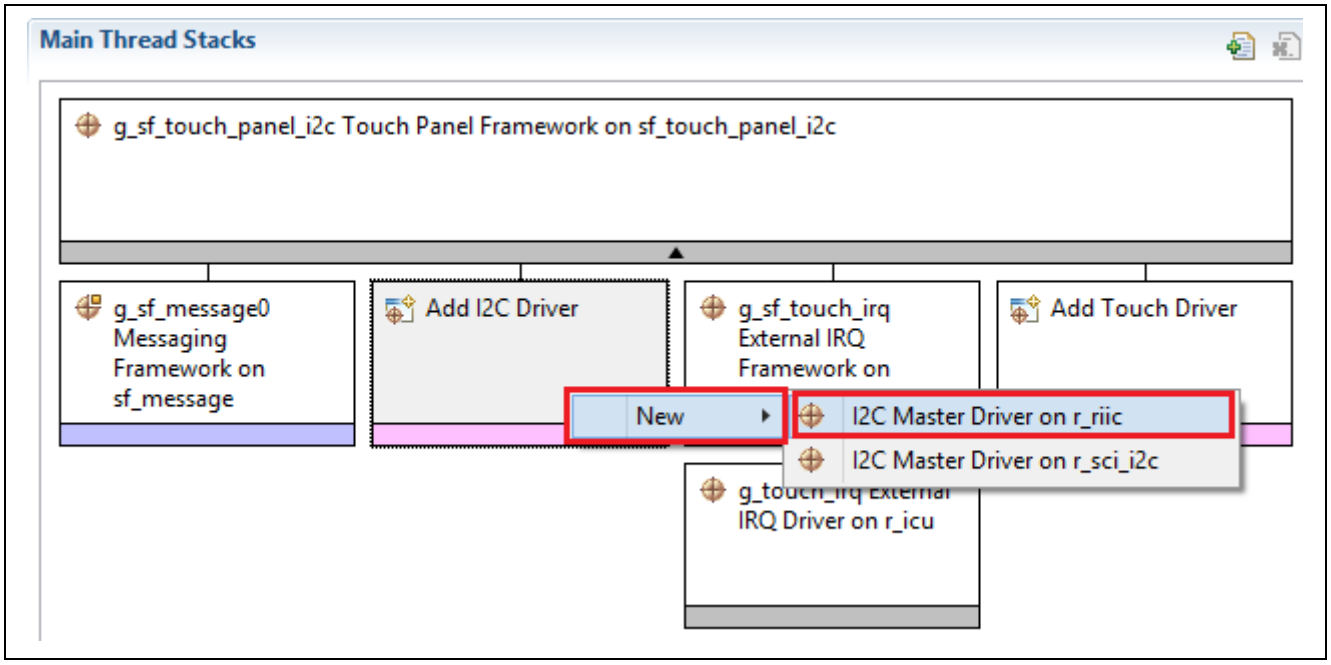


Figure 53. Adding I²C Driver

- Configure the following Properties for **I²C Master Driver** on `riic`.
  Note: Change the Channel option first.

Property	Value
Common	
Parameter Checking	Default (BSP)
Module g_i2c I2C Master Driver on r_riic	
Name	g_i2c
Channel	1
Rate	Fast-mode
Slave Address	0x38
Address Mode	7-Bit
Callback	NULL
Receive Interrupt Priority	Priority 3 (CM4: valid, CM0+: lowest - not valid if using ThreadX)
Transmit Interrupt Priority	Priority 3 (CM4: valid, CM0+: lowest - not valid if using ThreadX)
Transmit End Interrupt Priority	Priority 3 (CM4: valid, CM0+: lowest - not valid if using ThreadX)
Error Interrupt Priority	Priority 3 (CM4: valid, CM0+: lowest - not valid if using ThreadX)

Figure 54. Configuring I²C Driver

- Configure the following Properties of the **Touch Driver**.

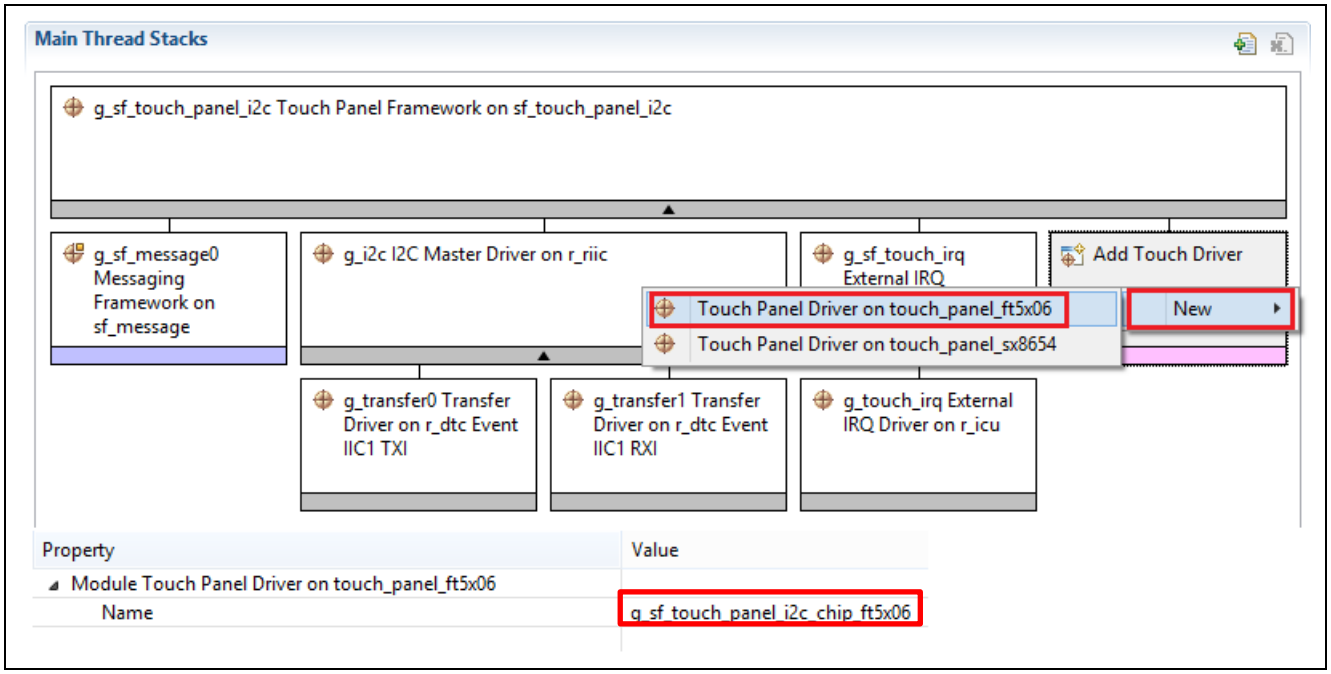


Figure 55. Configuring Touch Driver

- Under **HMI Thread Stacks**, select **New Stack > X-Ware > GUIX > GUIX on gx**.

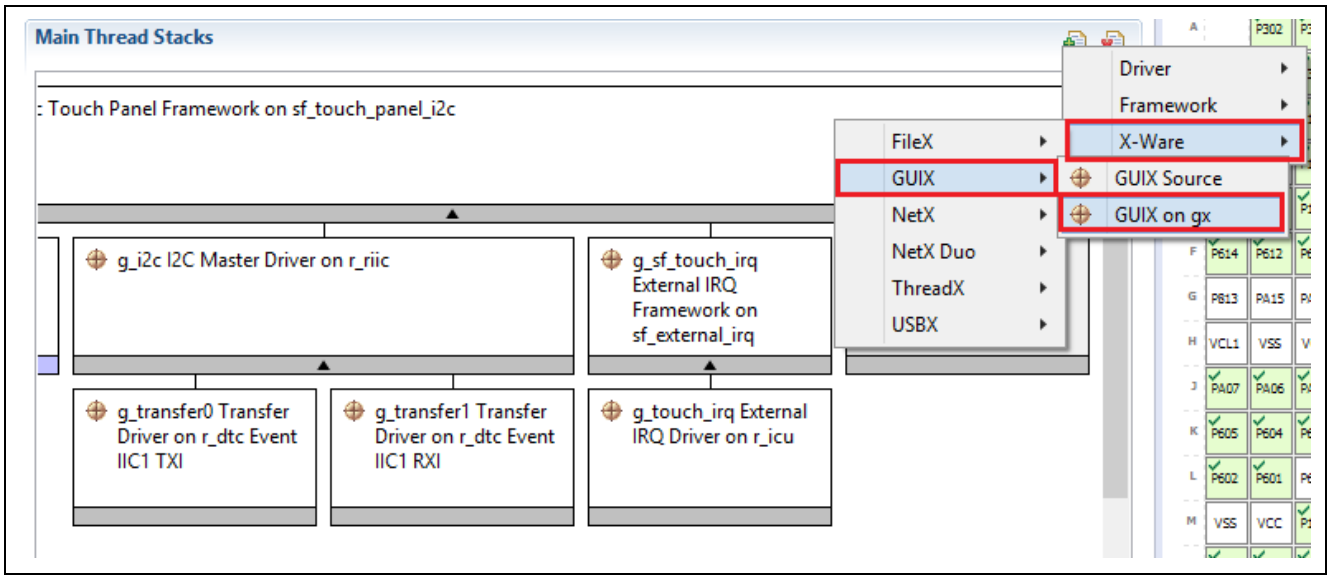


Figure 56. GUIX on gx

Notice that the Synergy Configurator has now already created the **GUIX Port on sf_el_gx framework**, **Display Driver**, and has a placeholder for the JPEG decode and D/AVE hardware accelerator stacks.

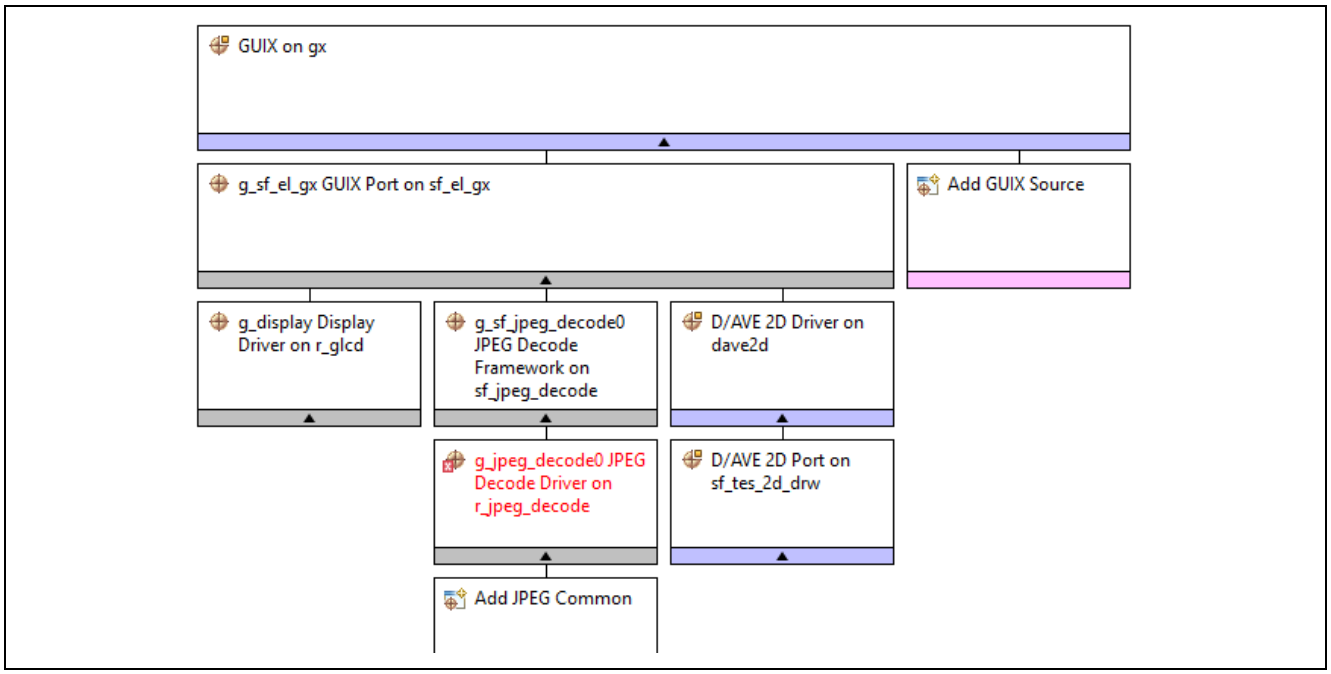


Figure 57. GUIX on gx

- Select **GUIX on gx** and configure the following **Properties**.

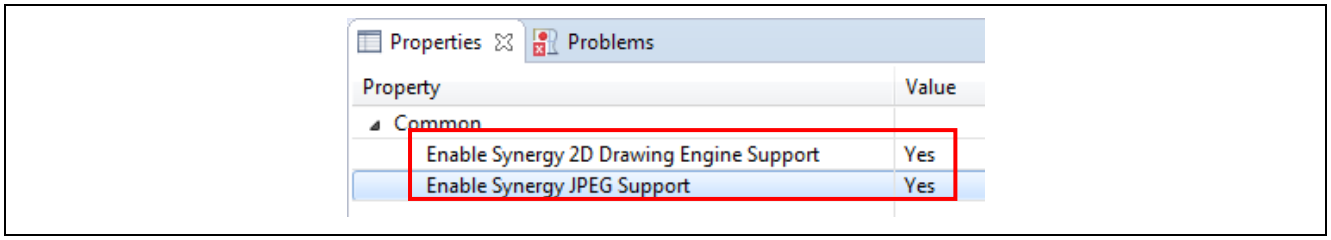


Figure 58. GUIX on gx Properties

- Add **JPEG Common** to the **Decode Driver** on `r_jpeg_decode`.

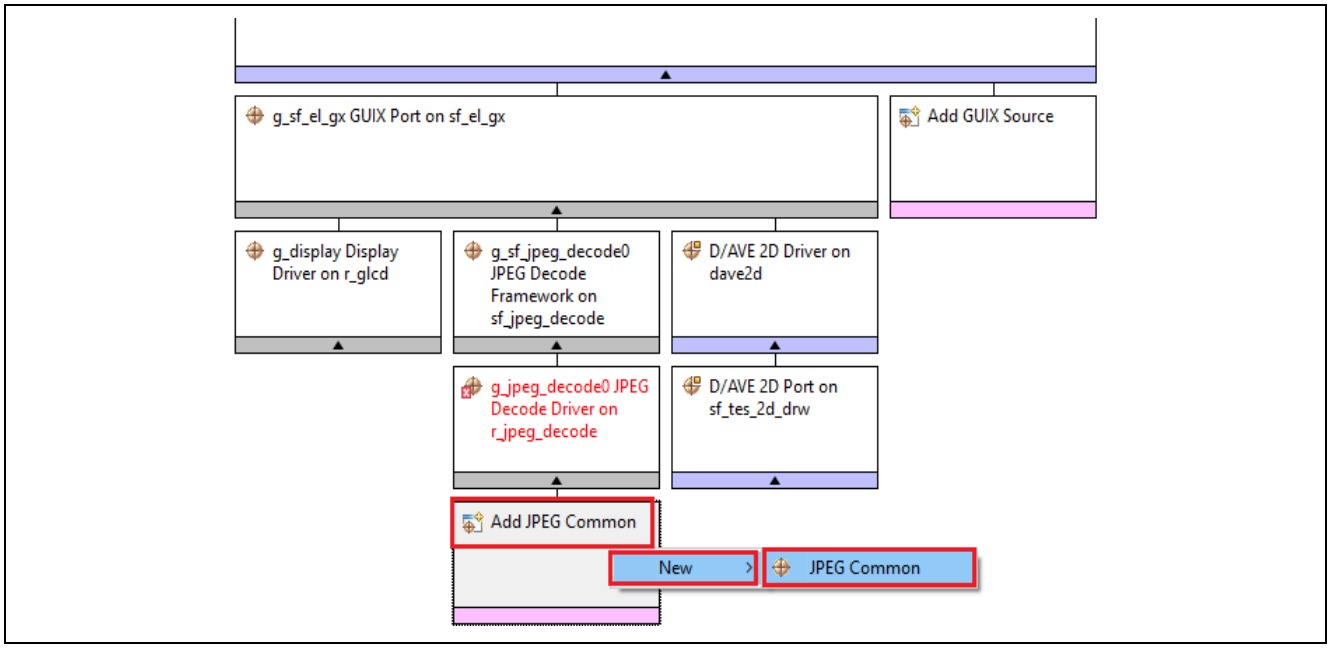
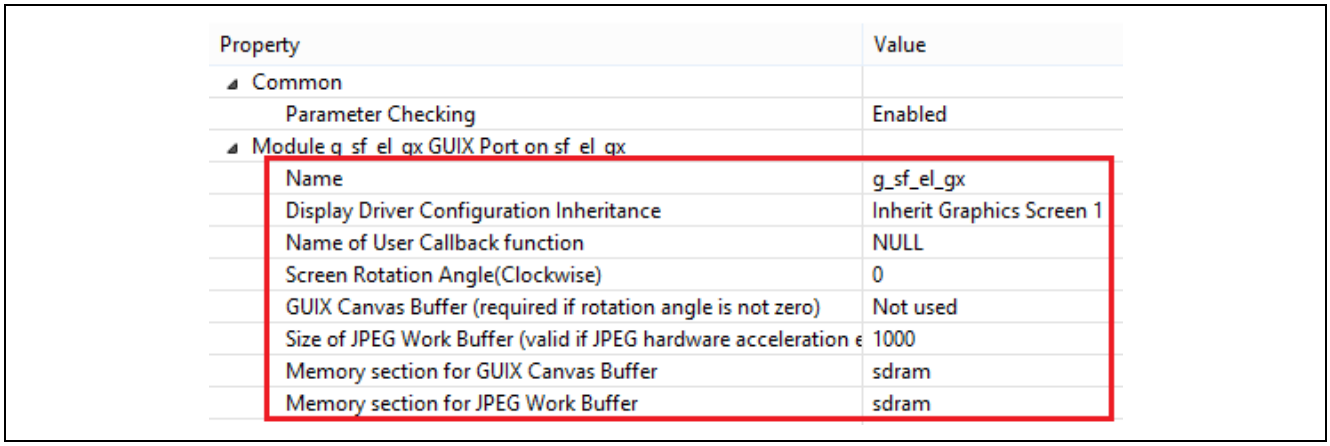


Figure 59. JPEG Common Module

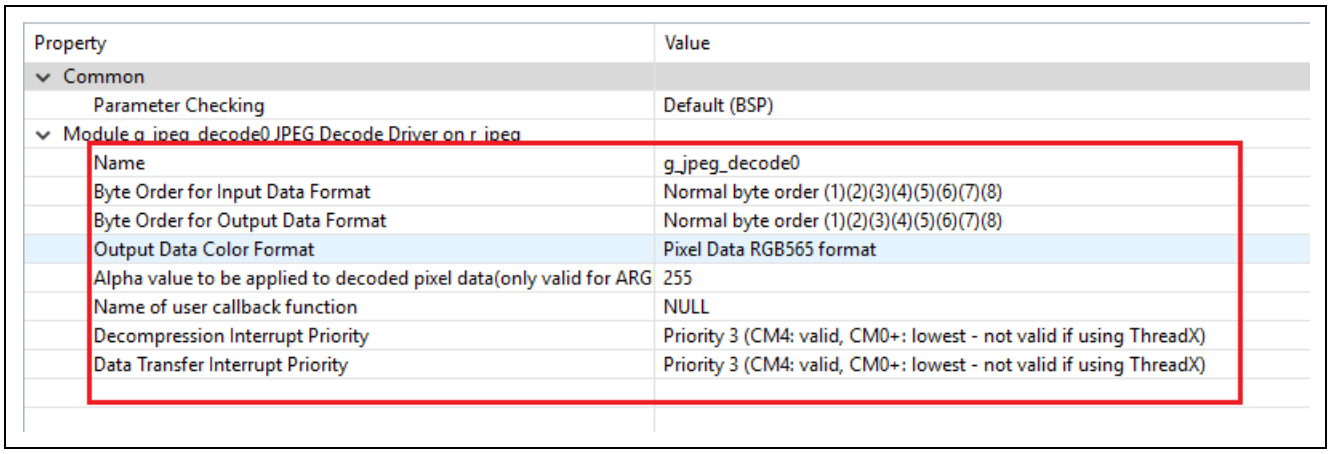
- Select **GUIX Port** on `sf_el_gx` and configure the following **Properties**.



Property	Value
Common	
Parameter Checking	Enabled
Module g_sf_el_gx GUIX Port on sf_el_gx	
Name	g_sf_el_gx
Display Driver Configuration Inheritance	Inherit Graphics Screen 1
Name of User Callback function	NULL
Screen Rotation Angle(Clockwise)	0
GUIX Canvas Buffer (required if rotation angle is not zero)	Not used
Size of JPEG Work Buffer (valid if JPEG hardware acceleration e	1000
Memory section for GUIX Canvas Buffer	sdram
Memory section for JPEG Work Buffer	sdram

Figure 60. GUIX Port on the sf_el_gx Properties

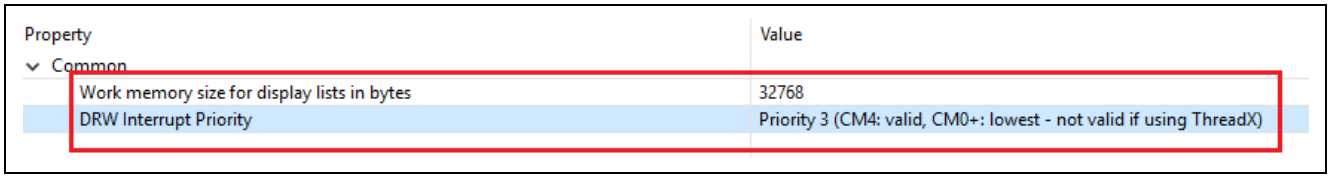
- Select **JPEG Decode Driver** `r_jpeg` and configure the following interrupt properties. Note that Priority 3 is just an arbitrary number.



Property	Value
Common	
Parameter Checking	Default (BSP)
Module g_jpeg_decode0 JPEG Decode Driver on r_jpeg	
Name	g_jpeg_decode0
Byte Order for Input Data Format	Normal byte order (1)(2)(3)(4)(5)(6)(7)(8)
Byte Order for Output Data Format	Normal byte order (1)(2)(3)(4)(5)(6)(7)(8)
Output Data Color Format	Pixel Data RGB565 format
Alpha value to be applied to decoded pixel data(only valid for ARG	255
Name of user callback function	NULL
Decompression Interrupt Priority	Priority 3 (CM4: valid, CM0+: lowest - not valid if using ThreadX)
Data Transfer Interrupt Priority	Priority 3 (CM4: valid, CM0+: lowest - not valid if using ThreadX)

Figure 61. JPEG Decode Driver on r_jpeg Properties

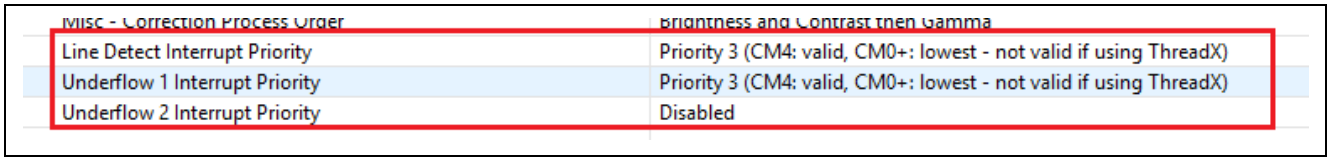
- Under **Main Thread Stacks**, select **D/AVE 2D Port** on `sf_tes_2d_drw` and configure the following properties.



Property	Value
Common	
Work memory size for display lists in bytes	32768
DRW Interrupt Priority	Priority 3 (CM4: valid, CM0+: lowest - not valid if using ThreadX)

Figure 62. D/AVE 2D Port Properties

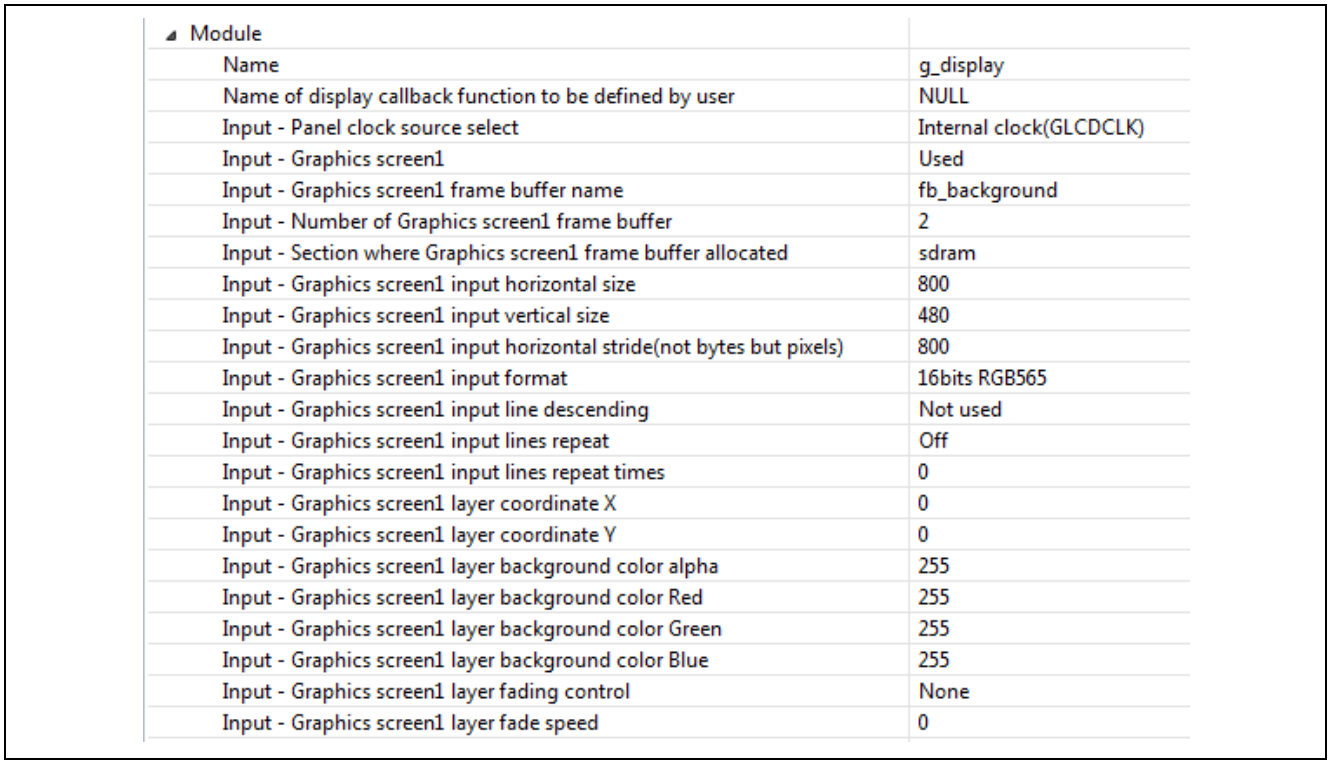
- Under **Main Thread Stacks**, select **Display Driver** on `r_glcd` and configure the following **Interrupt Properties**.



Misc - Correction Process Order	Brightness and Contrast then Gamma
Line Detect Interrupt Priority	Priority 3 (CM4: valid, CM0+: lowest - not valid if using ThreadX)
Underflow 1 Interrupt Priority	Priority 3 (CM4: valid, CM0+: lowest - not valid if using ThreadX)
Underflow 2 Interrupt Priority	Disabled

Figure 63. Interrupt Properties

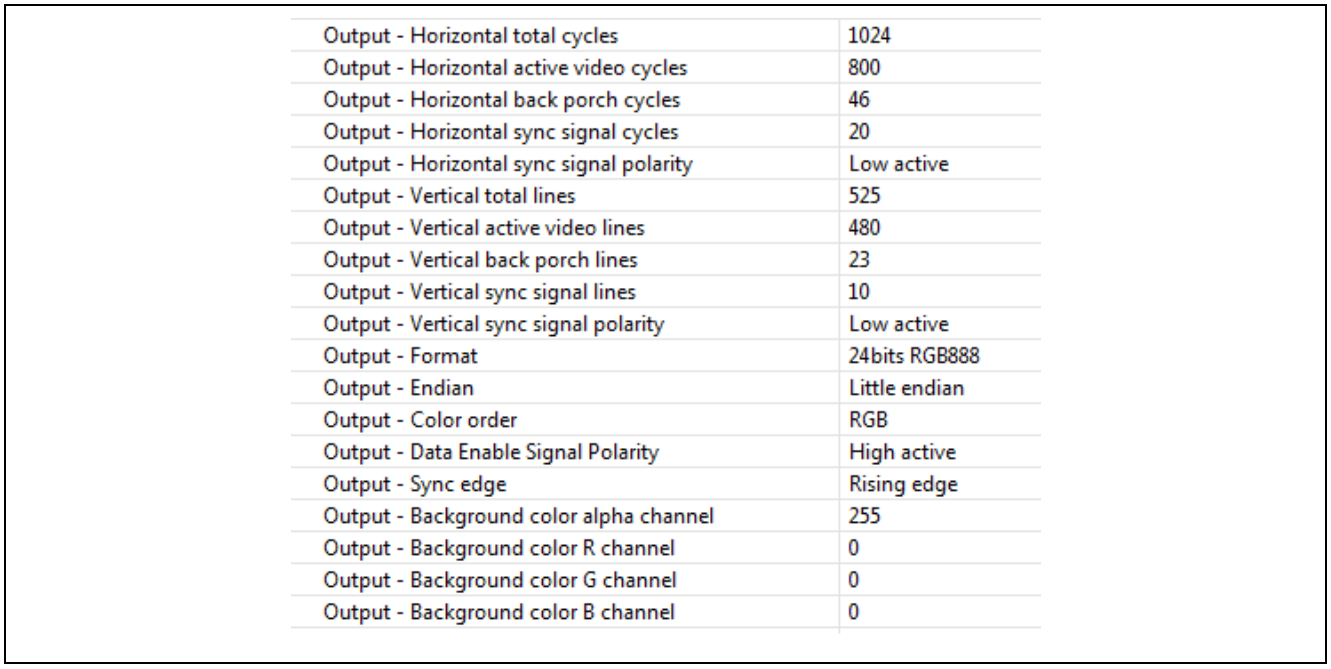
- Scroll down to show the following **Graphics Screen 1 Properties**.



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	800
Input - Graphics screen1 input vertical size	480
Input - Graphics screen1 input horizontal stride(not bytes but pixels)	800
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 64. Graphics Screen 1 Properties

- Configure the following **Output properties**.



Output - Horizontal total cycles	1024
Output - Horizontal active video cycles	800
Output - Horizontal back porch cycles	46
Output - Horizontal sync signal cycles	20
Output - Horizontal sync signal polarity	Low active
Output - Vertical total lines	525
Output - Vertical active video lines	480
Output - Vertical back porch lines	23
Output - Vertical sync signal lines	10
Output - Vertical sync signal polarity	Low active
Output - Format	24bits RGB888
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 65. Output Screen 2 Properties

- Change the following **TCON settings** to match.

TCON - Hsync pin select	LCD_TCON0
TCON - Vsync pin select	LCD_TCON1
TCON - DataEnable pin select	LCD_TCON2
TCON - Panel clock division ratio	1/8

Figure 66. TCON Settings

- Select the **Messaging** tab on the **Synergy Configuration** window. The following window is shown.

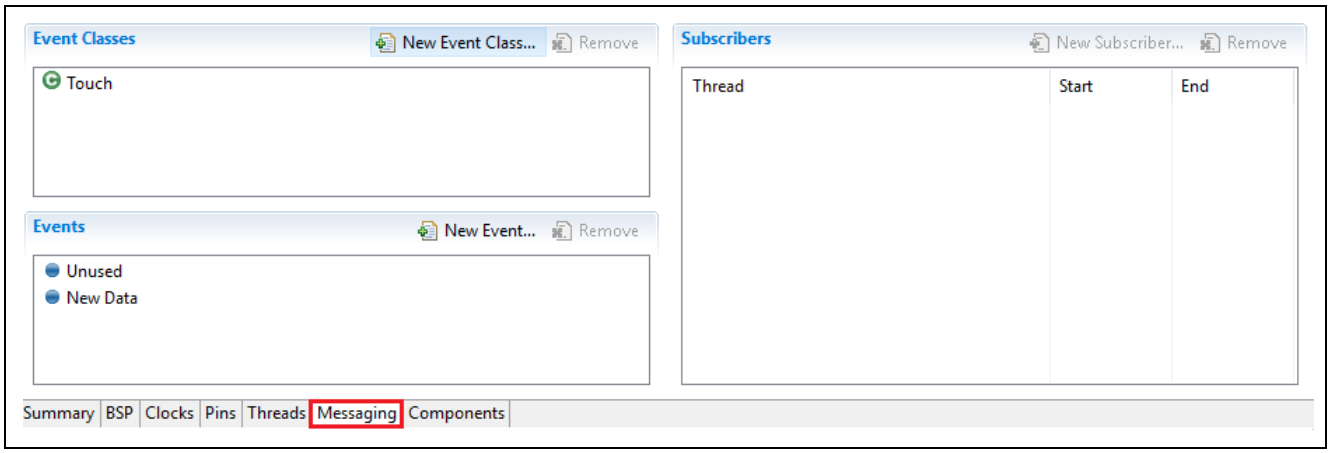


Figure 67. Messaging Tab

Note: This tab configures the event class definitions for the touchscreen events, along with the event queue initialization and linking variables. The touch event automatically generates when the **Touch Panel Framework** on `sf_touch_panel_i2c` is added in the **Threads** menu.

- Select the Touch Event class.
- On the **Touch Subscribers** menu, click the **New Subscriber button**.

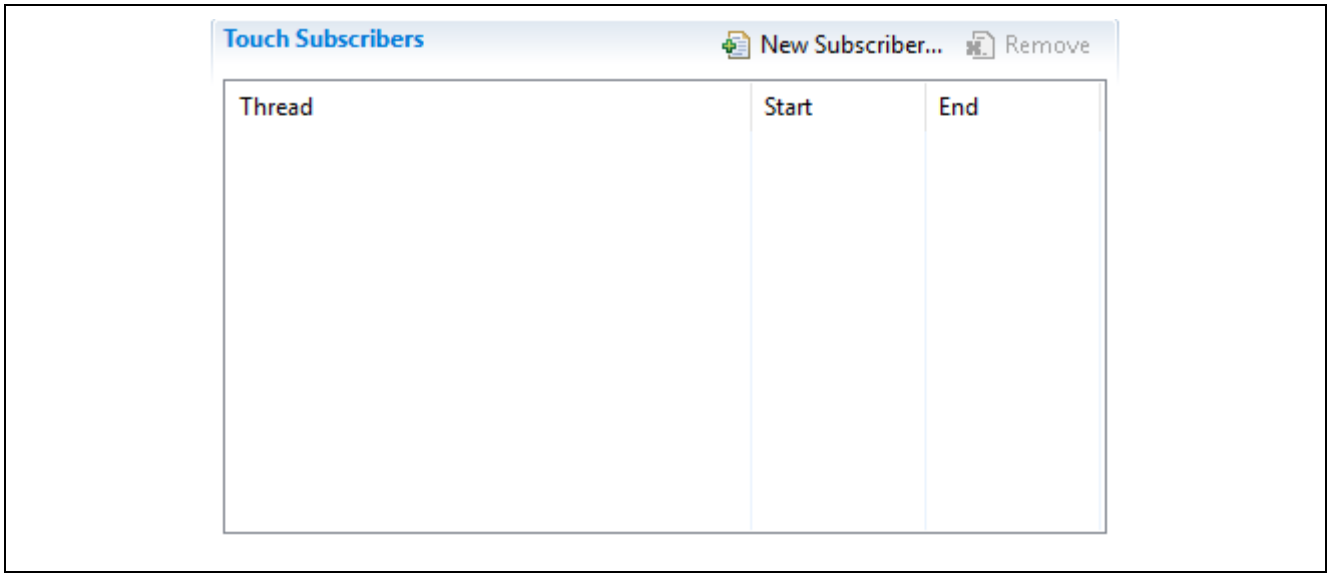


Figure 68. Messaging Tab

- In the **New Subscribers** Dialog, select the **HMI Thread**.

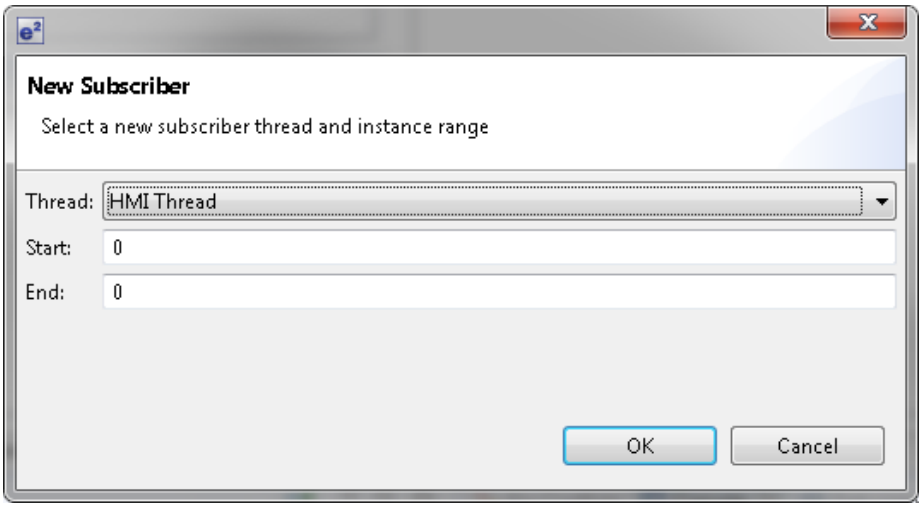


Figure 69. New Subscriber Dialog

- **Click** the **OK** button.
- **Save** the project by pressing **Ctrl + s** on the keyboard.
- Click the **Generate Project Content** button to update the project files.

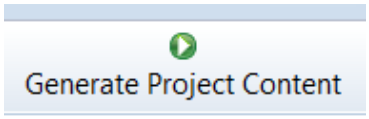


Figure 70. Generate Project Content

This will generate the `HMI_thread_entry.c` file under `/src` folder. You need to initialize the GUIX system from the application thread as described in the section 3.6.1.

Initializing GUIX from Application Thread

From your created Application Thread (HMI thread), the initialization is done as follows:

- Initialize GUIX with the `gx_system_initialize` function
- Initialize GUIX drivers using the open API, `g_sf_el_gx.p_api > open`
- Configure the GUIX system with the `gx_studio_display_configure` function using the setup API
- Initialize the memory address of the canvas with the `canvasInit API, g_sf_el_gx.p_api > canvasInit`
- Create the main screen with the `gx_studio_named_widget_create` function and attach it to the root window
- Create Washer Mode Radial Slider
- Create Garments Mode Radial Slider
- Create Water Level Window
- Create Temperature Window
- Create Temperature Radial Slider
- Show the root window using the `gx_widget_show` function and make the main screen visible.
- Start the GUIX system with the `gx_system_start` function
- Initialize the LCD GPIO control
- Initialize and open the PWM driver to control the TFT panel backlight
- GUI engine takes over and renders the image on the LCD screen

Note: You can refer the `HMI_thread_entry.c` file from the attached Washing Machine Application project for more details.

3.6.1 Screen Specific Functions

The specific code to the individual screens can found in the `src/s7g2_pe_hmi1` directory of the project as shown in the following figure. The code handles the following functions for the individual screens:

- Initialization of the page
- Creating the Radial Slider
- Update the Widget Label
- Handling the events for the window
- Drawing the Icon
- Rotating the Icon to specific angle
- Handle the animation
- Update the status for the window

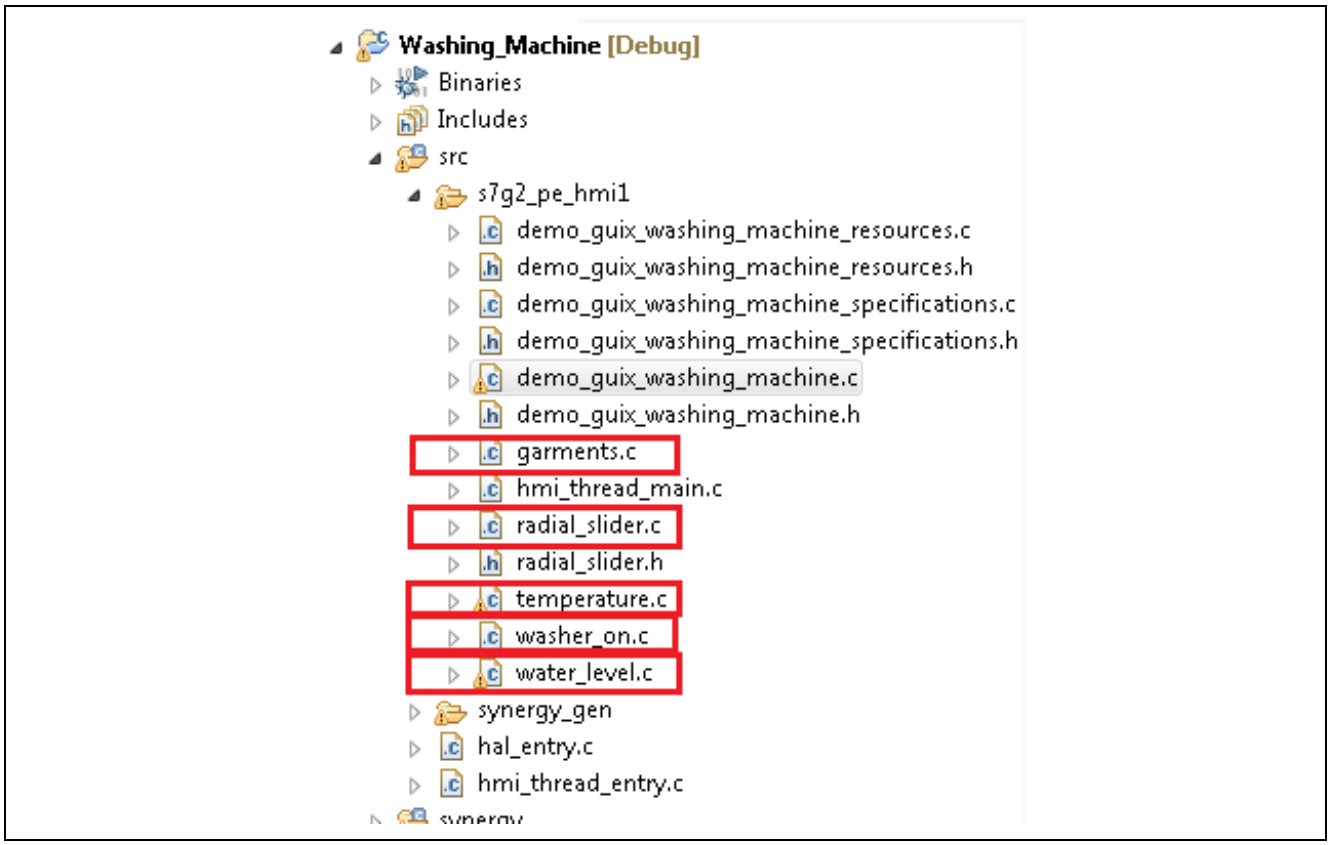


Figure 71. Screen Specific Files in the Project View

Note: These are not auto generated code. You need to write this to handle the Window Specific functions for the Washing Machine Application.
When all the code for the project is written, you can compile and test them on the PE-HMI board.
You can import the attached Washing Machine Application Project, build and test it on the PE-HMI board as described the following section.

4. Running the Washing Machine Application

4.1 Importing, Building and Loading the Project

Refer to the *Renesas Synergy Project Import Guide* (r11an0023eu0121-synergy-ssp-import-guide.pdf) included in this package. It contains instructions on importing the project into e^2 studio and building the project. The included Washing_Machine**.**zip file contains the completed project.

4.2 Loading and Debugging the Application

- Power the PE-HMI1 and connect the J-Link Lite Cortex M debugger to the PC and PE-HMI1.

Note: The application is not yet ready to be run on the target hardware. Use the following steps to run it.

- **Click** the **Debug drop-down menu** for the debug option.

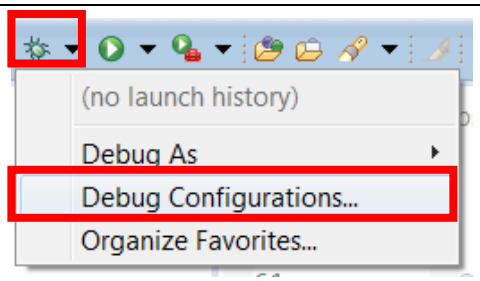


Figure 72. Debug Options

- Select the **Debug Configurations...** option.
- Under the **Renesas GDB Hardware Debugging** section, select **Washing_Machine Debug**.
- Click on the **Debug** button to start debugging.

4.3 Verifying the Application

After downloading and running the Washing_Machine Application on the PE-HMI board, you will see the main screen shown in the Figure 3. Select the **Garments, Water Level and Temperature Buttons**, as shown on the right side of the screen, to take you to the different GUI screens shown in the Figure 2.

The details of the screen are explained in the Section 3.3.

You can select, **Washer** screen and configure the different settings using:

- Small circular ball shaped icon within the Big Radial dialer wheel to move the Washer setting from anywhere between **Very Light** to **Rinse and Spin**.
- Touching the selection of the Wash, **Medium** or **Quick Wash**, brings the Small Icon to the right wash settings.

Select **Garments** screen and **configure** the different **Garment** selection using:

- Small circular ball shaped icon within the Big Radial dialer wheel to move the Garments selection setting from **Light Colors** to **Workout Clothes**.
- Touching the selection of desired garments will also take the small circular ball to the right settings. For example, touch the **Denim** or **Wool**, which brings the Small Ball Icon to the right garment settings.

Select **Water Level** screen and configure the different settings using:

- **Vertical Slider** within the **Water Level** icon to move the different temperature settings from **Very Low** to **Extra High**.
- Touch the selection of the **Water Temperature** to any desired settings, such as **Very Low** or **Extra High**, which brings the **Vertical Slider** to the right water temperature settings.

The status window shows the status for the remaining time wash cycle and temperature of the water, garments under wash and the remaining time for the simulated wash.

5. Next Steps

1. Visit www.renesas.com/synergy/kits for more information about the PE-HMII1 product example kits including its *Quick Start Guide*, design data, ordering information and other useful application projects.
2. Visit the **Website and Support** section for URL links to Synergy Software, Hardware, and Solutions, the Synergy Platform's key elements, where you can download components and documentation.
3. Visit the **Website and Support** section for URL links to Microcontrollers, Development tools, and Kits, as well as other support resources.

6. References

List of the GUIX reference documents and links.

- GUIX Studio User's Guide: https://rtos.com/solutions/guix-studio/embedded-ui-design-tool/
- GUIX User's Guide: (download zip: X-Ware™ Component Documents for Renesas Synergy™)
- GUIX Synergy™ Port Framework Module Guide: https://www.renesas.com/us/en/software/D6003967.html

List of the SSP and Application reference documents and links.

- GUIX Hello World Application Project on PE-HMI1 : https://www.renesas.com/us/en/software/D6003648.html
- SSP User Manual: https://www.renesas.com/en-us/products/synergy/software/ssp.html
- Renesas Synergy Knowledge Base: https://en-us.knowledgebase.renesas.com/English_Content/Renesas_Synergy%E2%84%A2_Platform/Renesas_Synergy_Knowledge_Base

Website and Support

Visit the following vanity URLs to learn about key elements of the Synergy Platform, download components and related documentation, and get support.

Synergy Software	www.renesas.com/synergy/software
Synergy Software Package	www.renesas.com/synergy/ssp
Software add-ons	www.renesas.com/synergy/addons
Software glossary	www.renesas.com/synergy/softwareglossary
Development tools	www.renesas.com/synergy/tools
Synergy Hardware	www.renesas.com/synergy/hardware
Microcontrollers	www.renesas.com/synergy/mcus
MCU glossary	www.renesas.com/synergy/mcuglossary
Parametric search	www.renesas.com/synergy/parametric
Kits	www.renesas.com/synergy/kits
Synergy Solutions Gallery	www.renesas.com/synergy/solutionsgallery
Partner projects	www.renesas.com/synergy/partnerprojects
Application projects	www.renesas.com/synergy/applicationprojects
Self-service support resources:	
Documentation	www.renesas.com/synergy/docs
Knowledgebase	www.renesas.com/synergy/knowledgebase
Forums	www.renesas.com/synergy/forum
Training	www.renesas.com/synergy/training
Videos	www.renesas.com/synergy/videos
Chat and web ticket	www.renesas.com/synergy/resourcelibrary

Revision History

Rev.	Date	Description	
		Page	Summary
1.00	Mar 27, 2018	—	Initial release
1.01	Apr.26.19	—	Updates for SSP v1.6.0
1.02	Jan.07.20	—	Updates for SSP v1.7.5

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 or any other use of the circuits, software, and information in the design of your product or system. Renesas Electronics disclaims any and all liability for any losses and damages incurred by you or third parties arising from the use of these circuits, software, or information.
2. Renesas Electronics hereby expressly disclaims any warranties against and liability for infringement or any other claims involving 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, including but not limited to, the product data, drawings, charts, programs, algorithms, and application examples.
3. 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 shall not alter, modify, copy, or reverse engineer any Renesas Electronics product, whether in whole or in part. Renesas Electronics disclaims any and all liability for any losses or damages incurred by you or third parties arising from such alteration, modification, copying or reverse engineering.
5. Renesas Electronics products are classified according to the following two quality grades: “Standard” and “High Quality”. The intended 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; industrial robots; etc.
   "High Quality": Transportation equipment (automobiles, trains, ships, etc.); traffic control (traffic lights); large-scale communication equipment; key financial terminal systems; safety control equipment; etc.

   Unless expressly designated as a high reliability product or a product for harsh environments in a Renesas Electronics data sheet or other Renesas Electronics document, Renesas Electronics products are not intended or 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 damage (space system; undersea repeaters; nuclear power control systems; aircraft control systems; key plant systems; military equipment; etc.). Renesas Electronics disclaims any and all liability for any damages or losses incurred by you or any third parties arising from the use of any Renesas Electronics product that is inconsistent with any Renesas Electronics data sheet, user’s manual or other Renesas Electronics document.
6. When using Renesas Electronics products, refer to the latest product information (data sheets, user’s manuals, application notes, “General Notes for Handling and Using Semiconductor Devices” in the reliability handbook, etc.), and ensure that usage conditions are within the ranges specified by Renesas Electronics with respect to maximum ratings, operating power supply voltage range, heat dissipation characteristics, installation, etc. Renesas Electronics disclaims any and all liability for any malfunctions, failure or accident arising out of the use of Renesas Electronics products outside of such specified ranges.
7. Although Renesas Electronics endeavors to improve the quality and reliability of Renesas Electronics products, semiconductor products have specific characteristics, such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Unless designated as a high reliability product or a product for harsh environments in a Renesas Electronics data sheet or other Renesas Electronics document, Renesas Electronics products are not subject to radiation resistance design. You are responsible for implementing safety measures to guard against the possibility of bodily injury, injury or damage caused by fire, and/or danger to the public in the event of a failure or malfunction of Renesas Electronics products, 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 and impractical, you are responsible for evaluating 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. You are responsible for carefully and sufficiently investigating applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive, and using Renesas Electronics products in compliance with all these applicable laws and regulations. Renesas Electronics disclaims any and all liability for damages or losses occurring as a result of your noncompliance with applicable laws and regulations.
9. Renesas Electronics products and technologies shall 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 shall comply with any applicable export control laws and regulations promulgated and administered by the governments of any countries asserting jurisdiction over the parties or transactions.
10. It is the responsibility of the buyer or distributor of Renesas Electronics products, or any other party who distributes, disposes of, or otherwise sells or transfers the product to a third party, to notify such third party in advance of the contents and conditions set forth in this document.
11. This document shall not be reprinted, 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.

(Note1) “Renesas Electronics” as used in this document means Renesas Electronics Corporation and also includes its directly or indirectly controlled subsidiaries.

(Note2) “Renesas Electronics product(s)” means any product developed or manufactured by or for Renesas Electronics.

(Rev.4.0-1 November 2017)

Corporate Headquarters

TOYOSU FORESIA, 3-2-24 Toyosu,
Koto-ku, Tokyo 135-0061, Japan
www.renesas.com

Trademarks

Renesas and the Renesas logo are trademarks of Renesas Electronics Corporation. All trademarks and registered trademarks are the property of their respective owners.

Contact information

For further information on a product, technology, the most up-to-date version of a document, or your nearest sales office, please visit:
www.renesas.com/contact/.