Renesas RA Family

GUIX "Hello World" for EK-RA8D1

Introduction

This application note describes the process of creating a simple two-screen GUI using Azure RTOS GUIX Studio for the EK-RA8D1 kit. This application demonstrates how easily a user can create and configure a new application using the Renesas Flexible Software Package (FSP).
The Renesas Flexible Software Package includes Azure RTOS ThreadX® real-time operating system, the Azure RTOS GUIX library and hardware drivers unified under a single robust software package. This powerful suite of tools provides a comprehensive integrated framework for rapid development of complex embedded applications.

The Hello World application was developed under e² studio using the Renesas Flexible Software Package.

Required Resources

Development tools and software

- e² studio IDE Version: 2023-10 (23.10.0)
- Renesas Flexible Software Package (FSP) v5.1.0
- Azure RTOS GUIX Studio V6.2.1.0

Hardware

- Renesas EK-RA8D1 kit (RA8D1 MCU Group)
  - ER-TFT043-3 with Capacitive Touch Panel 40 pins connection.
  - Recommended user should use RA6M3G kit's LCD.
- Renesas EK-RA8D1's "SW1" switches setting.
  - Switch #6 for GLCDC set "ON" and switch #7 for SDRAM set "ON".
- Renesas-app-lcd-conv_v1_b_mfg order from the link: https://oshpark.com/shared_projects/pzfp0mCD
  - User needs to click to "Actions" button to order LCD converter board.
  - Refer to section "3". Step 11 on Figure: 29 for pins connection.

Reference Manuals

- RA Flexible Software Package Documentation Release v5.1.0
- Azure RTOS GUIX and GUIX Studio v6.2.1.0
- Renesas RA8D1 Group User's Manual Rev.1.1.0
- EK-RA8D1-v1.0 Schematics

Provided Software Files

- A `Source.zip` folder that has a `touch_ft5x06` folder and 4 `*.c` files inside.
- `hal_entry.c`, `system_thread_entry.c`, `touch_thread_entry.c`, `windows_handler.c`

Purpose

This document will guide you through the setup of an Azure RTOS GUIX touch screen interface Hello World application in e² studio. This document will show how to configure the drivers and library included with the FSP. These will allow you to set up the GLCD Controller, the touch screen driver, and semaphores to communicate with application tasks. It also shows the steps necessary to create a simple GUI interface using the Azure RTOS GUIX Studio editor. In addition, this app note will also cover project setup 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.
Note: If the user wants to skip fully design and running the fully functional project see Chapter or Section 6.

Contents

1. Downloading and Installing Tools

1.1 Overview

In this section you will copy materials to your PC and install v2023-10.0 /FSP v5.1.0 and Azure RTOS GUIX studio v6.2.1.0.

1.2 Procedural Steps

1. If you already have e^2 studio v2023-10 with FSP v5.1.0 or later installed, you can skip this step. Otherwise, you can download it from https://www.renesas.com/us/en/software-tool/flexible-software-package-fsp

   Detailed installation instructions for the e^2 studio and the FSP are available on the Renesas website https://www.renesas.com/fsp. Review the release notes for e^2 studio to ensure that the e^2 studio version supports the selected FSP version. The starting version of the installer includes all features of the RA MCUs.

2. You can get Azure RTOS GUIX Studio v6.2.1.0 or greater from this link.

   You should see the window in the next step on the web browser.
   Note: Microsoft Store must be installed and working on your PC to install Azure RTOS GUIX studio.

3. Click **Install** button and a new window pops up. Click **Open Microsoft Store**.

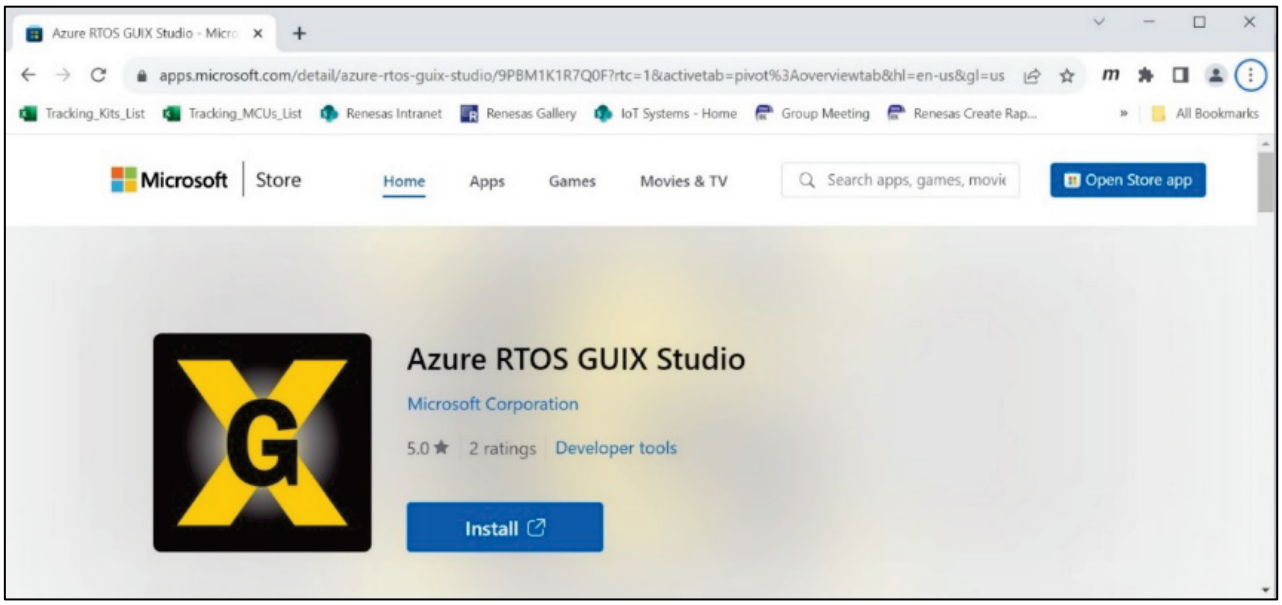


Figure 1. Get Azure RTOS GUIX Studio

4. Click **Open Microsoft store** to continue installing Azure RTOS GUIX studio.

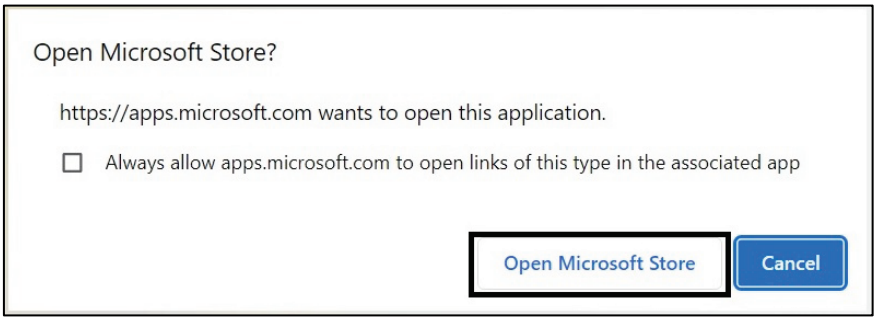
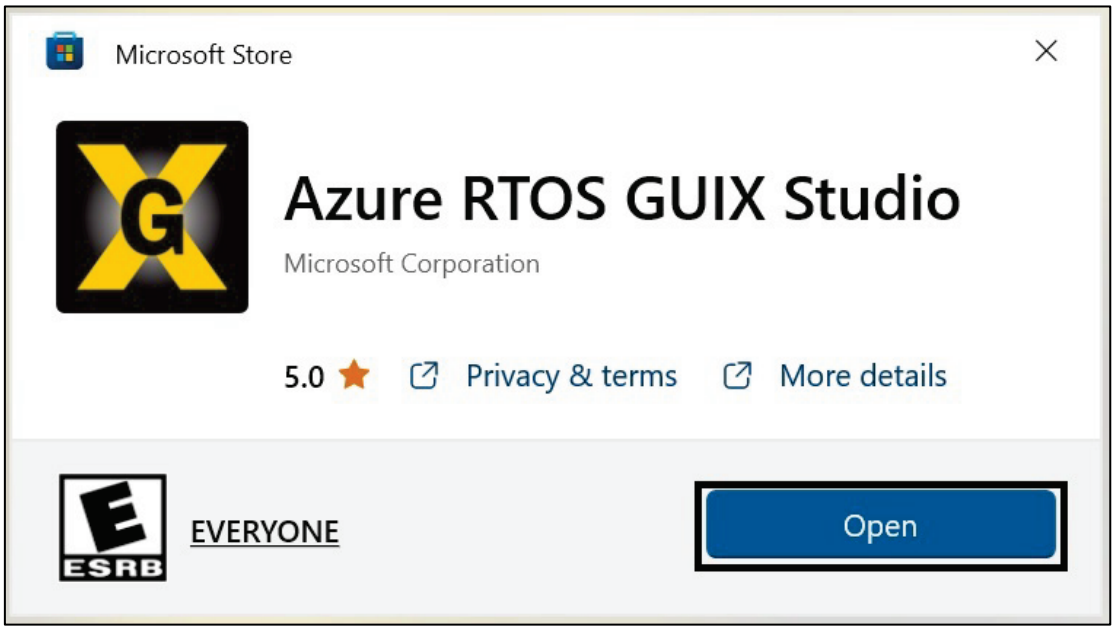


Figure 2. Open Microsoft Store

5. Click “Open” to open “Azure RTOS GUIX Studio” App.



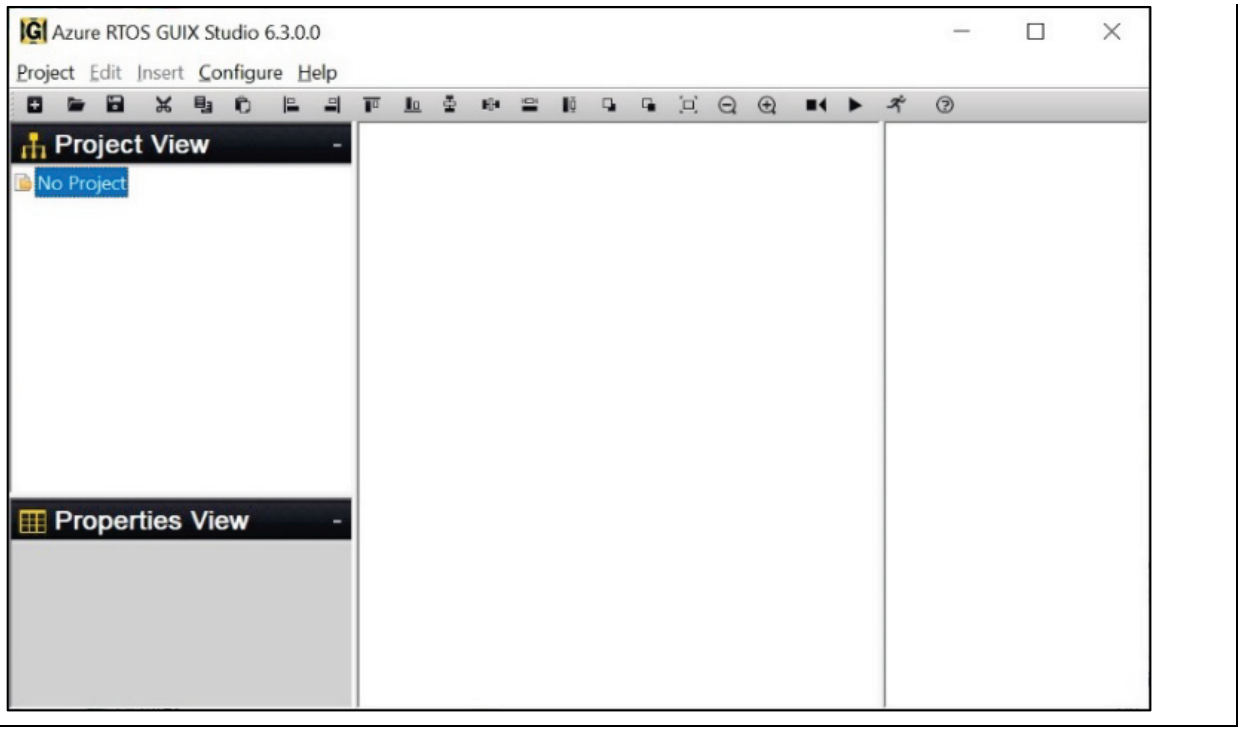


Figure 3. Click Open to start "Azure RTOS GUIX Studio"

2. Creating Application Project and Enabled Backlight

2.1 Overview

In this section, you will create a project to which you will add pre-written source code and integrate it with a pre-created Azure RTOS GUIX studio project.

2.2 Procedural Steps

1. Create a new Renesas RA C/C++ project. Name it ra8d1_guix_hello_world.

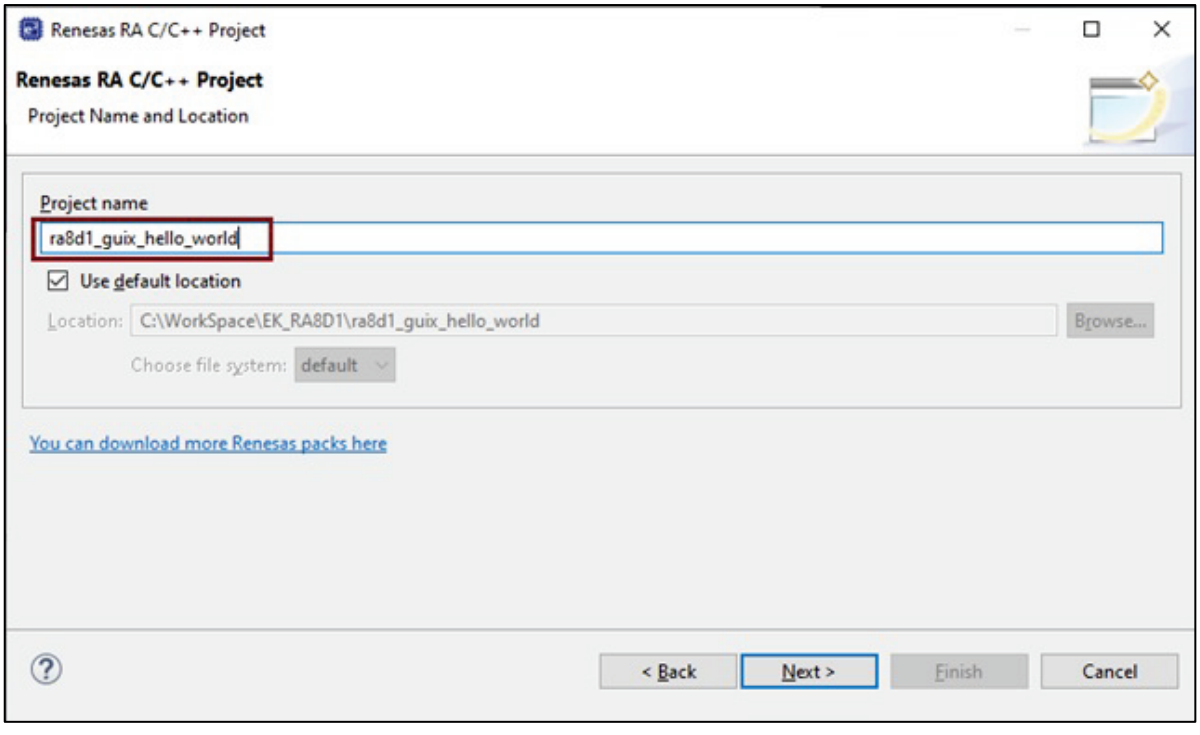


Figure 4. Create New Project

2. Select and set board to EK-RA8D1.

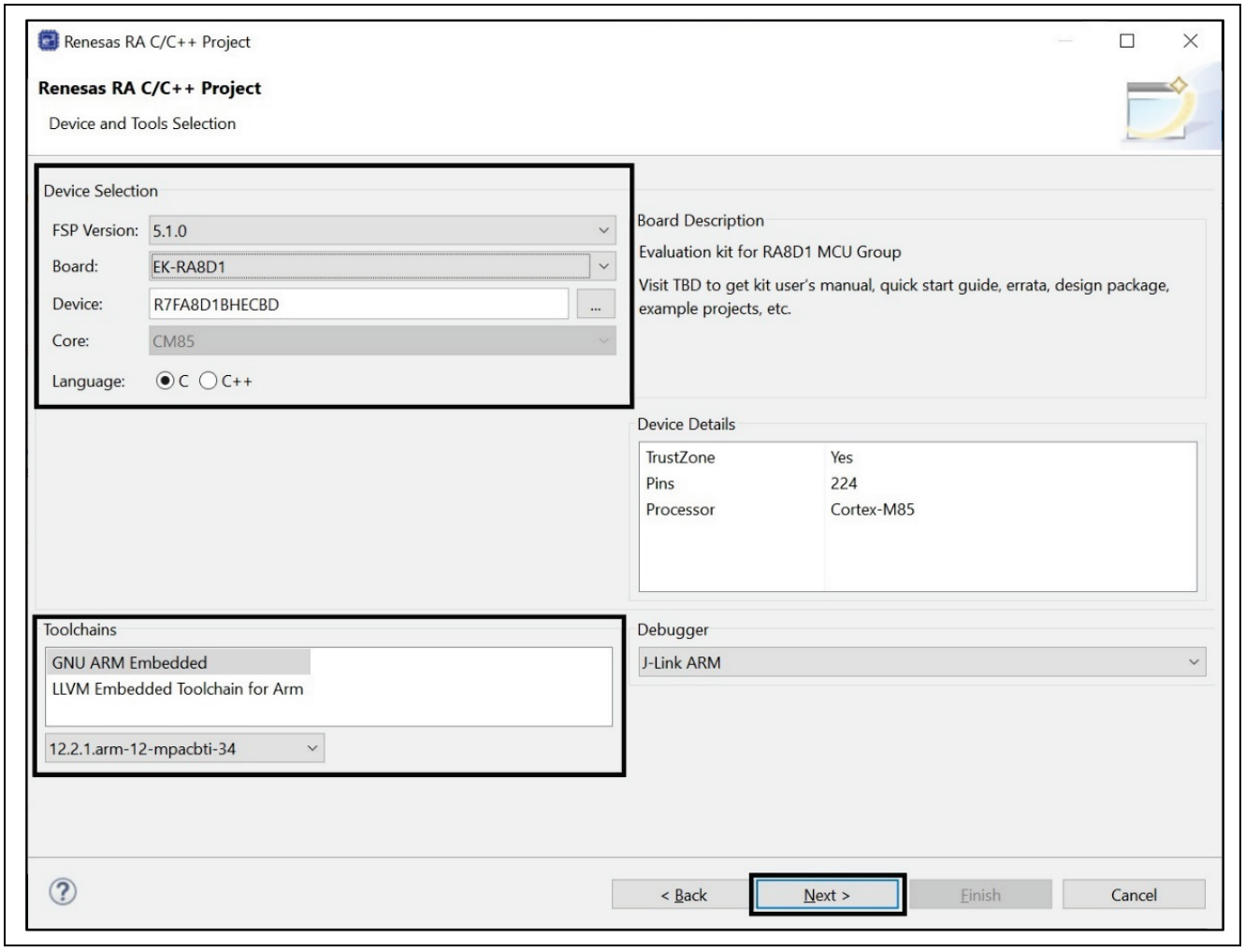


Figure 5. Select and Set Board to EK-RA8D1

3. Select Flat (Non-TrustZone) and Executable with Azure RTOS ThreadX (v6.2.1+FSP.5.1.0).

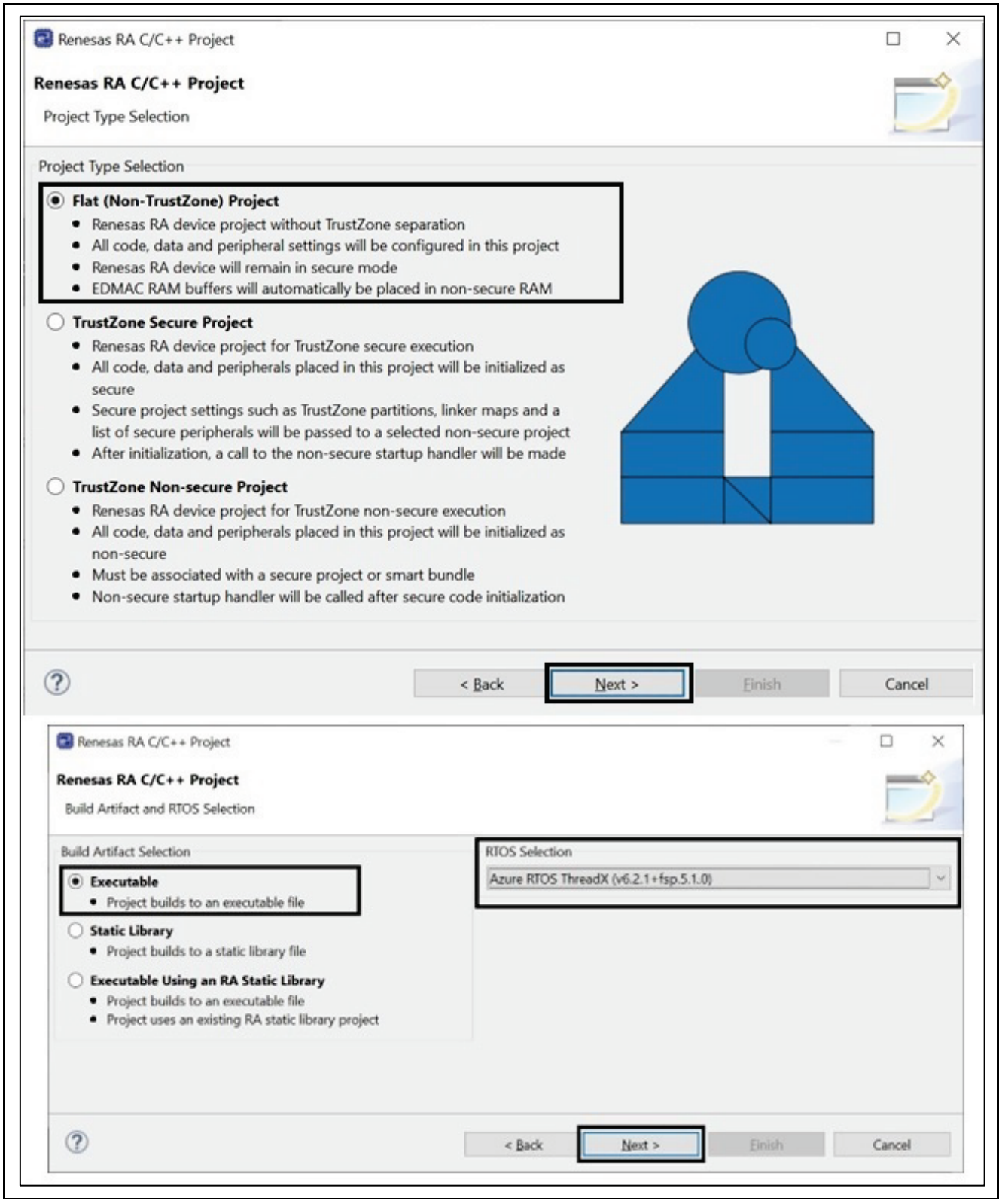


Figure 6. Select Azure RTOS ThreadX

4. Use **FreeRTOS - Minimal** template.

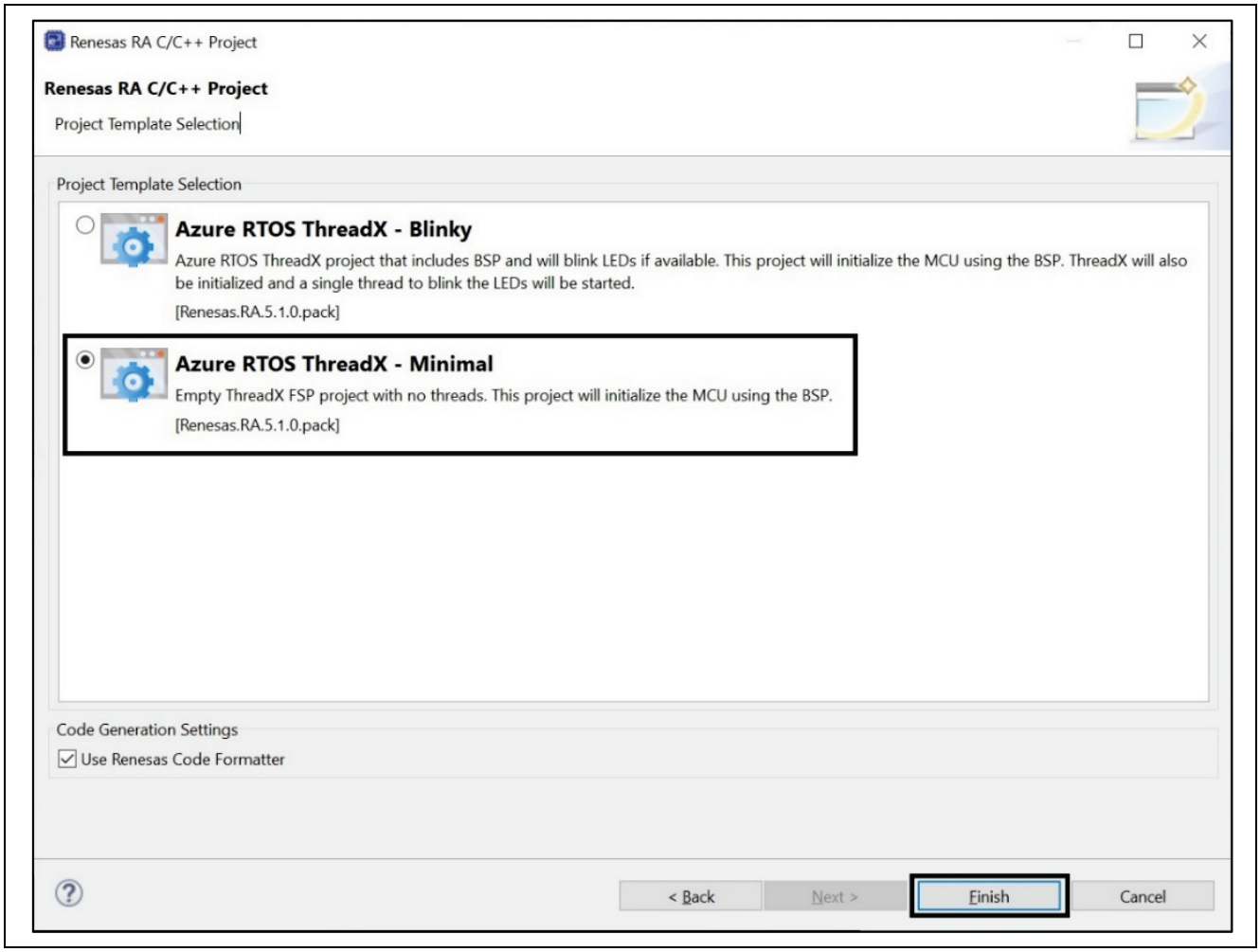


Figure 7. Select Azure RTOS ThreadX Minimal Finish

5. Open the project configuration and go to the **BSP** tab. Change **Heap size (bytes)** to **0x2000**.

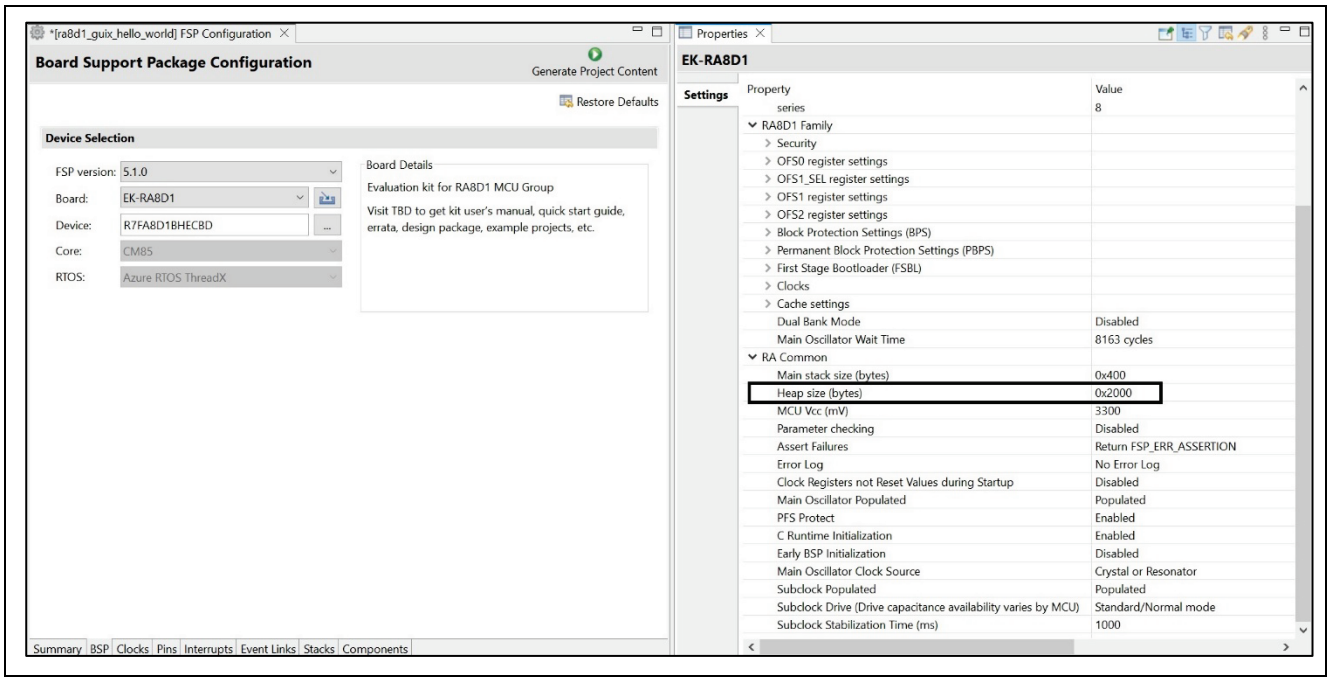


Figure 8. Change Heap Size

6. Click tab “**Clocks**” and set “Clocks” for the LCD.

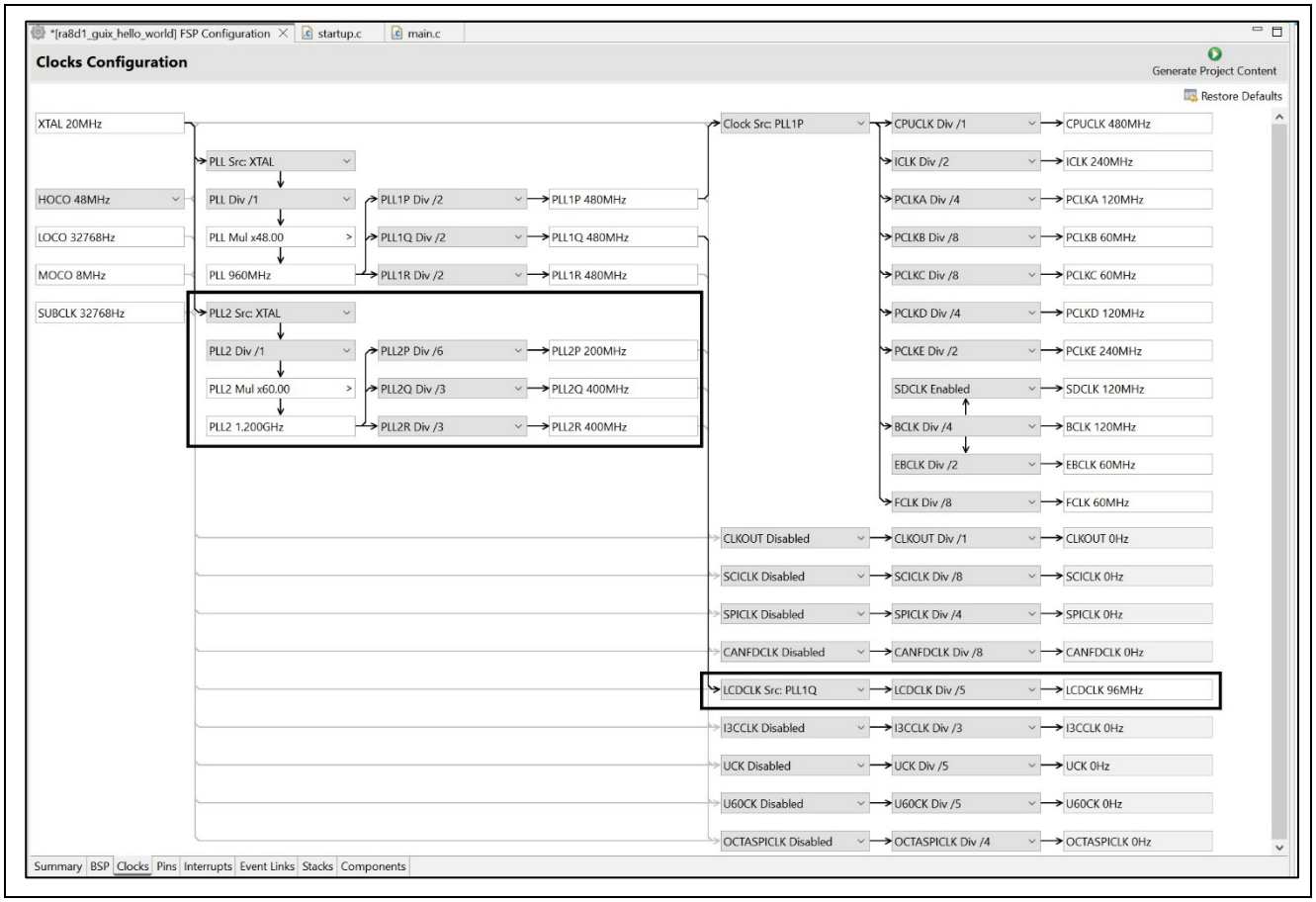


Figure 9. Setting All Clocks

7. Add a **new thread** and name it as **System Thread** with the settings below.

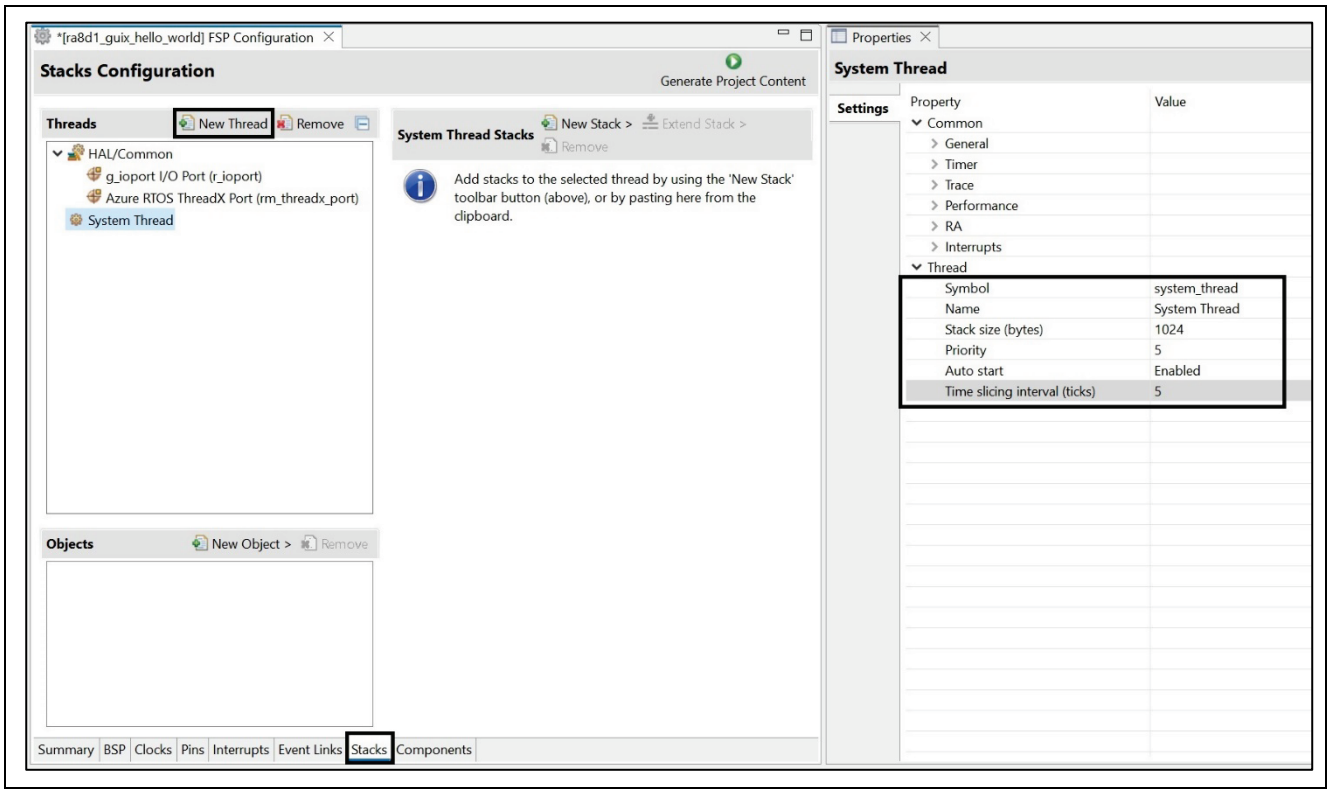


Figure 10. Add System Thread

8. Click “**New Stack**” and add **Azure RTOS GUIX** to System thread.

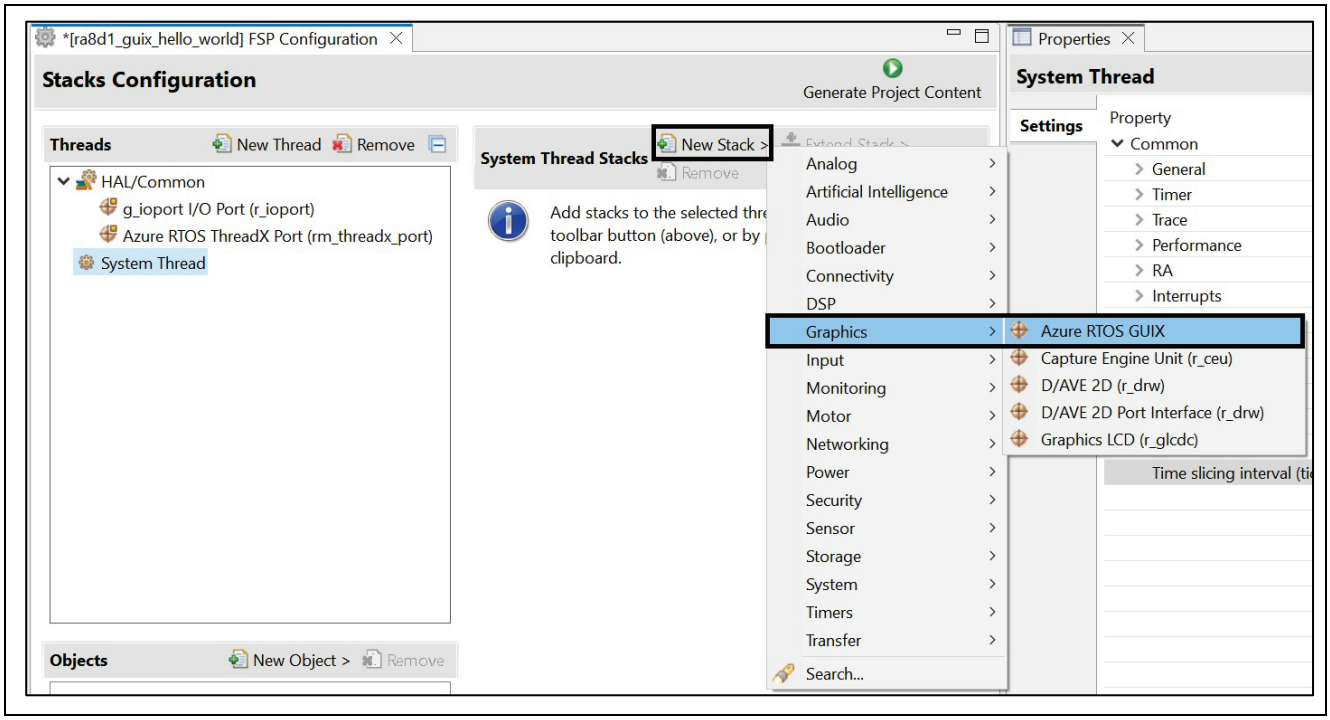


Figure 11. Add Azure RTOS GUIX

9. Settings properties for **Azure RTOS GUIX**

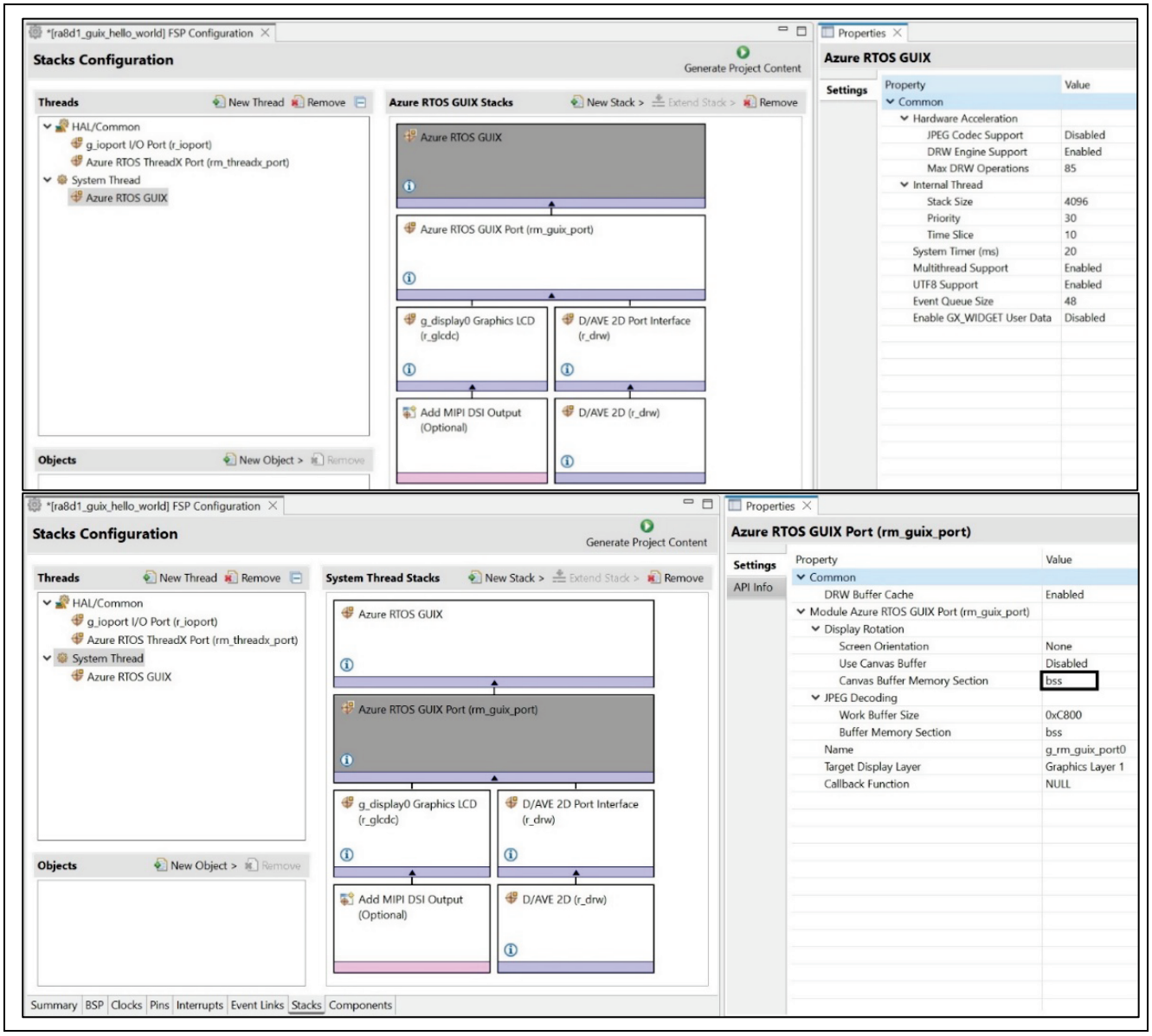


Figure 12. Setting and Checking Properties for Hardware

10. Settings property for **Graphics LCD**
    Note: Setting properties for the LCD.

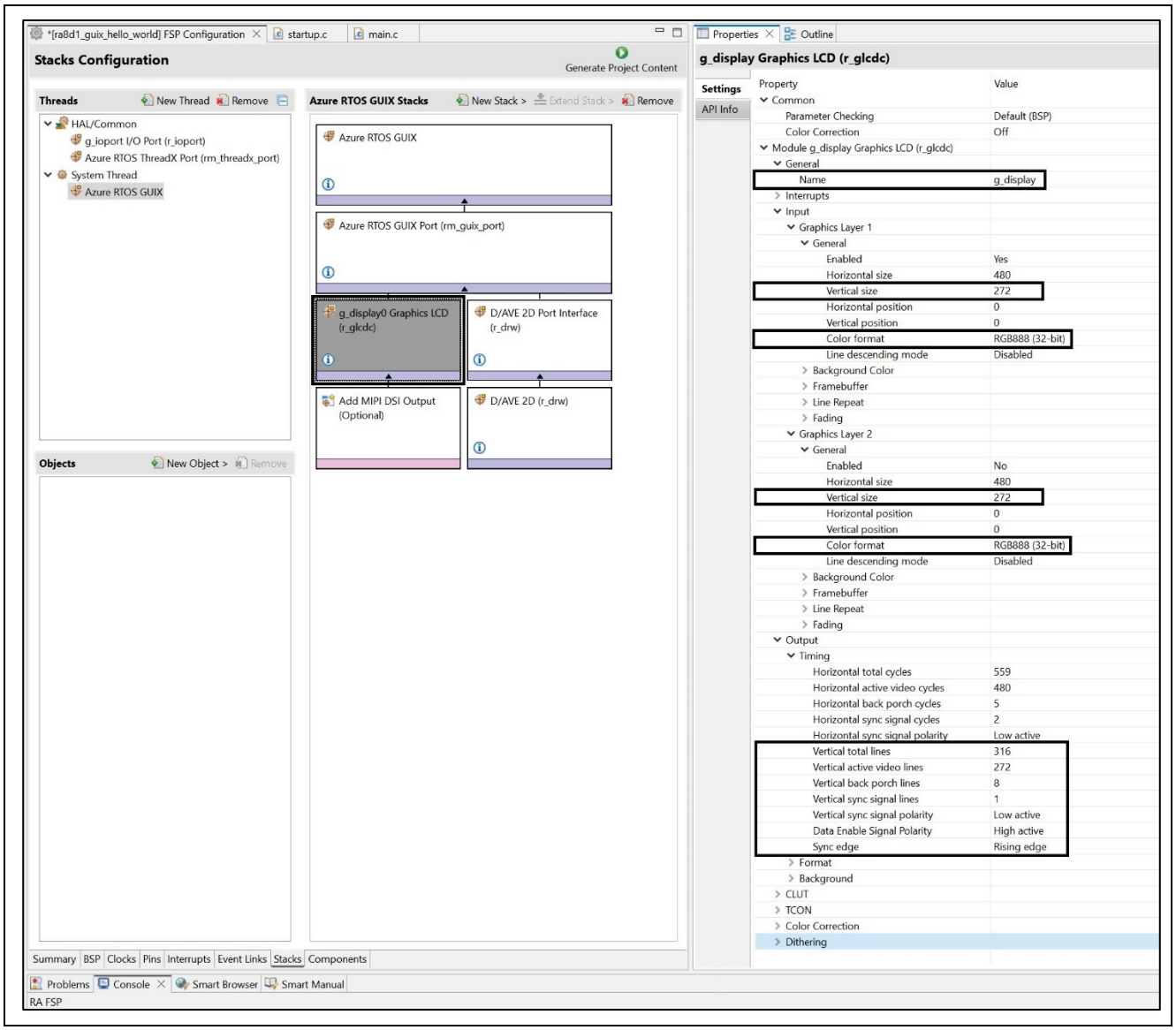


Figure 13. Setting Properties for Graphic LCD

11. Click “**New Stack**” and add **PWM Timer**.

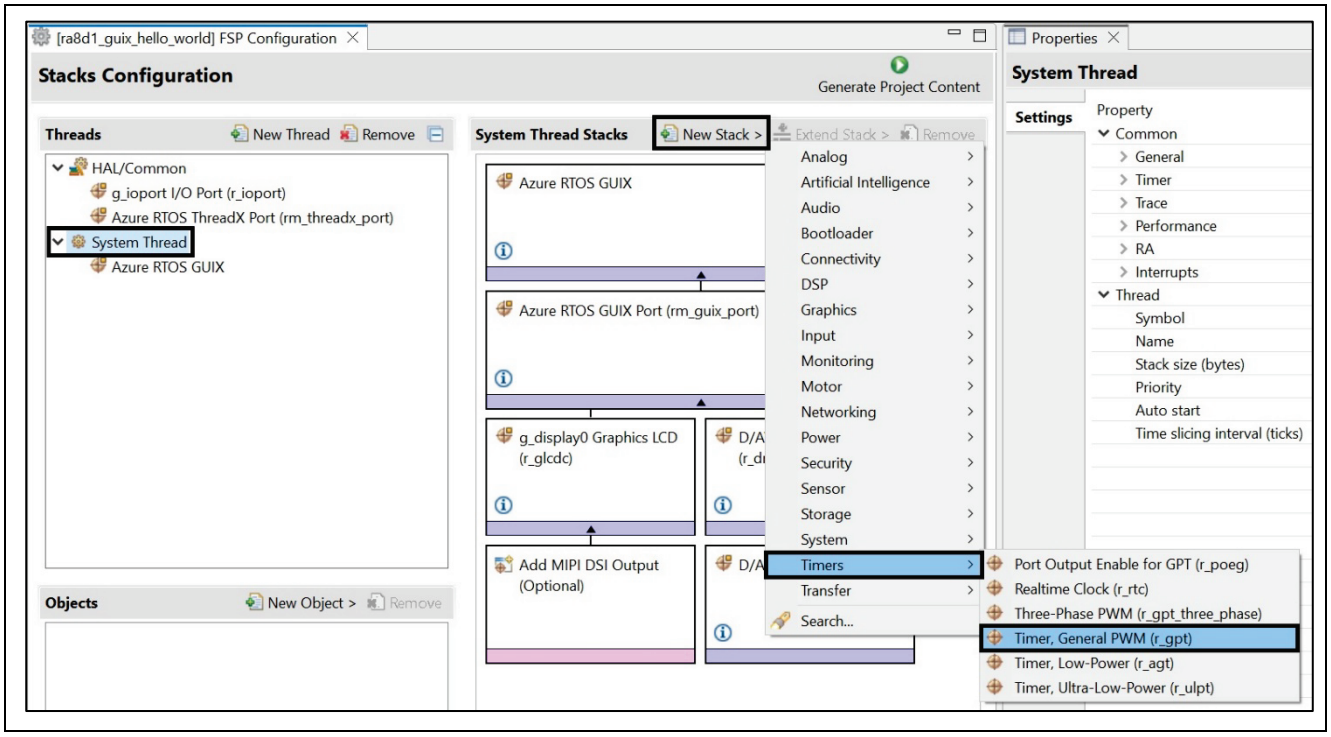


Figure 14. Add the Timer

12. Setting “**Timer**” module properties. Click “arrow”  to set pin P404 for “DISP_BLEN”.

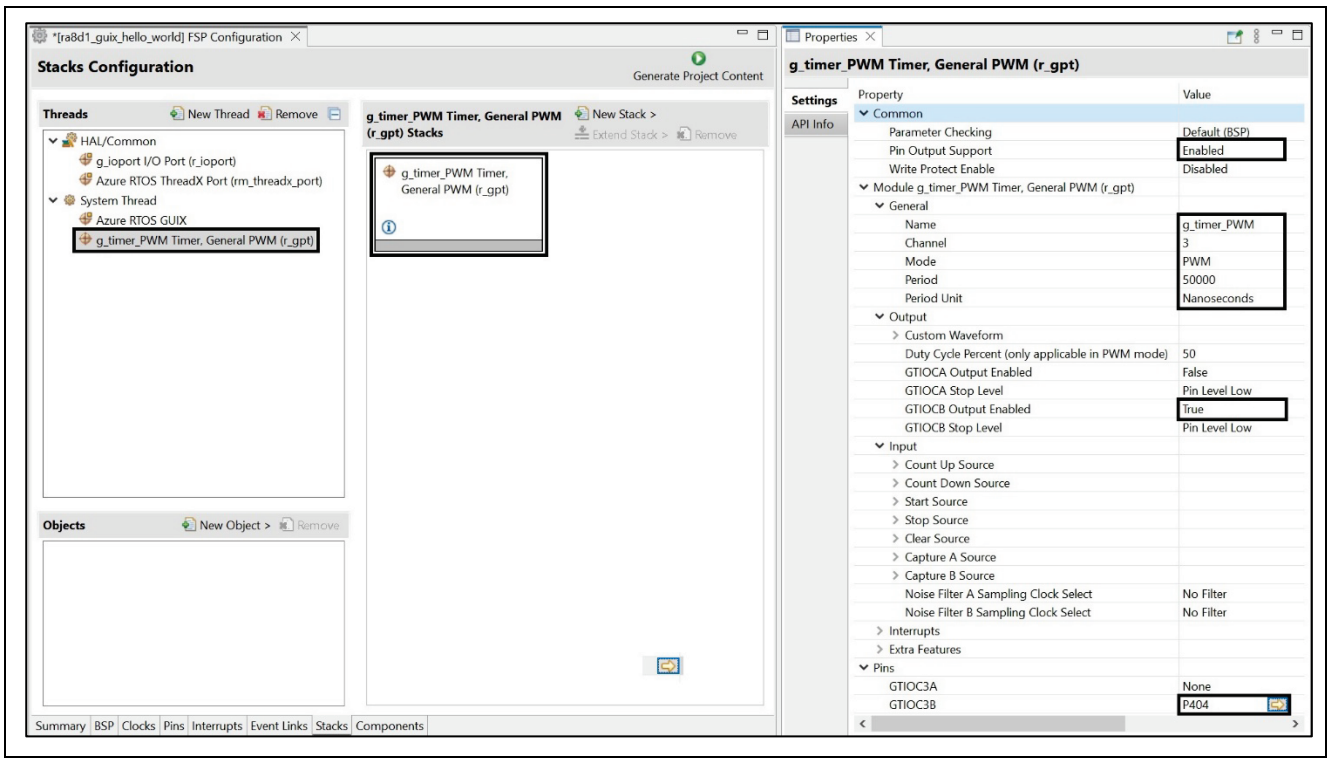


Figure 15. Settings Timer Property and DISP_BLEN pin P404

13. Configuration pin **P404** for the **DISP_BLEN** signal of the LCD panel.
    Note: If the GTIOC3B Pin showed as <unavailable> then user needs to perform step 13 first then step 12.

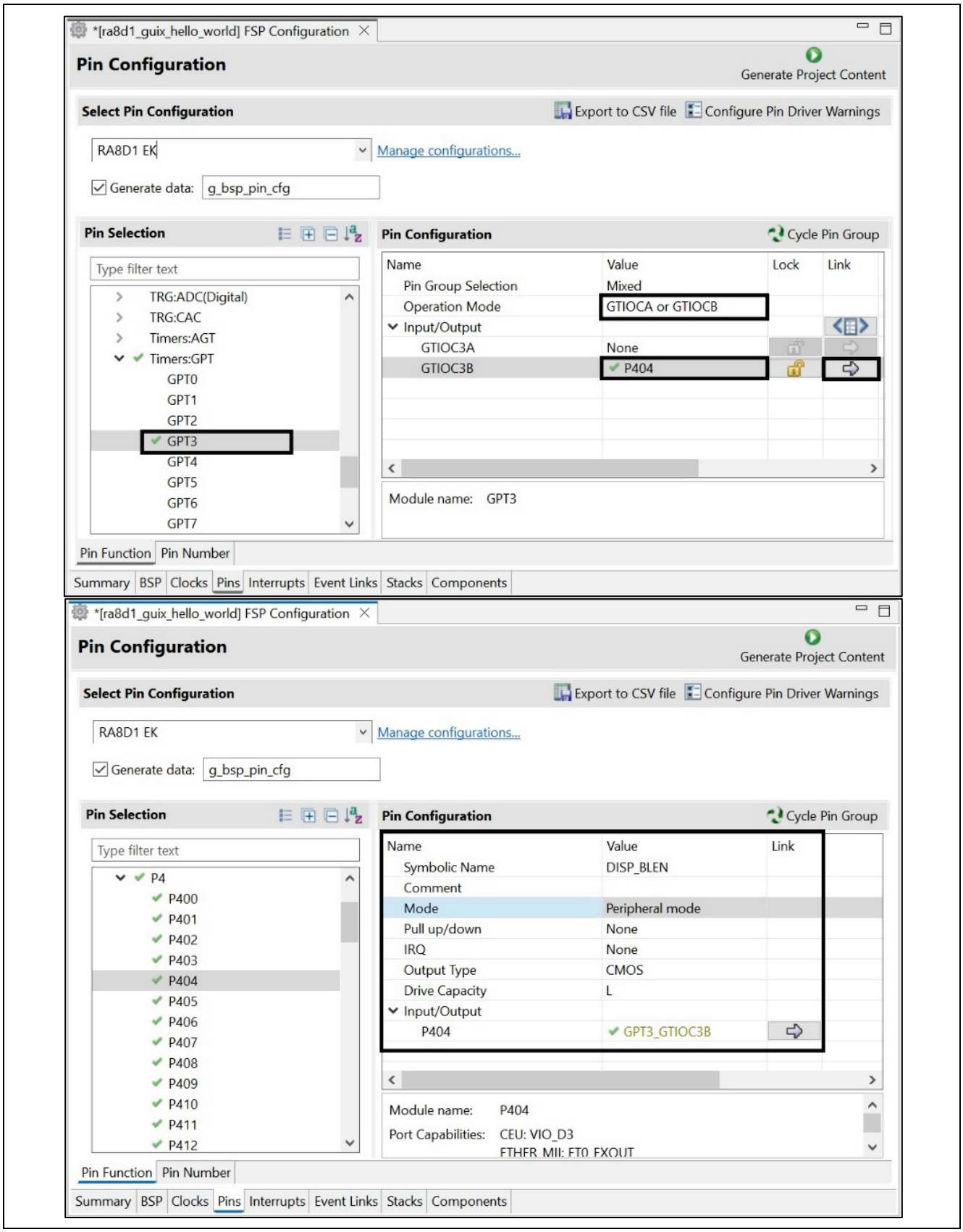


Figure 16. Settings for Pin P404 DISP_BLEN

14. LCD_BLEN pin from the LCD panel will connect to DISP_BLEN pin P404 of the board RA8D1 **Figure 17**.

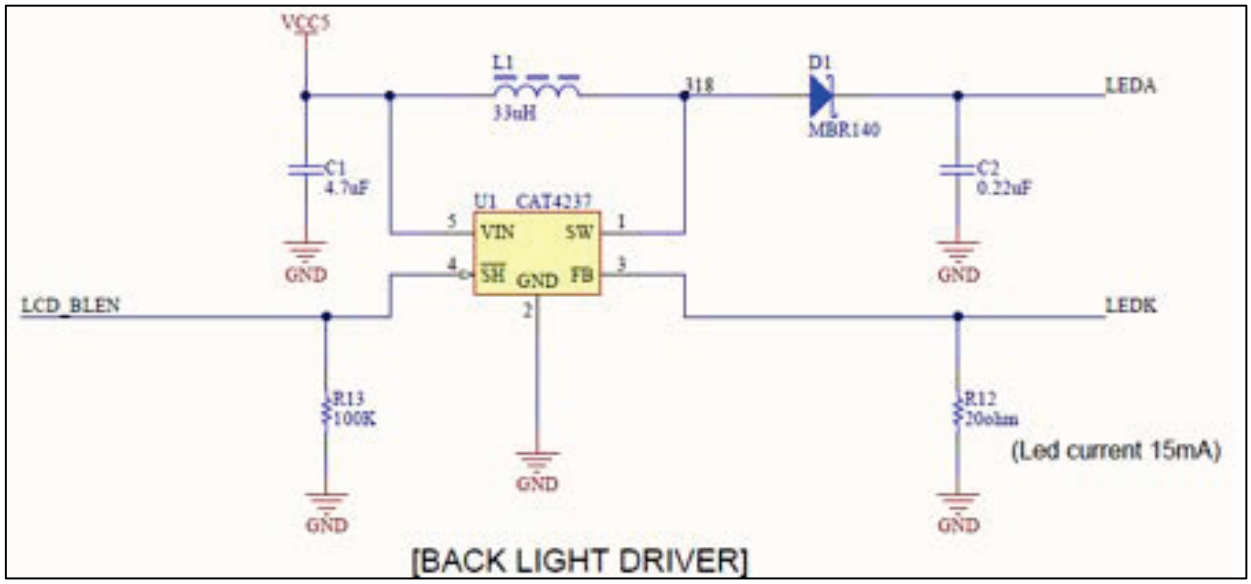


Figure 17. LCD_BLEN will connect to DISP_BLEN pin P404

15. Refer to “system_thread_entry.c” file in “**Source.zip**” for more information. This function below controls the PWM output.

```
 * @brief This function is setting up GPT/PWM timer
static fsp_err_t gpt_timer_PWM_setup(void)
{
    fsp_err_t err = FSP_SUCCESS;
    /* Open GPT */
    err = R_GPT_Open(&g_timer_PWM_ctrl, &g_timer_PWM_cfg);
    if(FSP_SUCCESS != err)
    {
        return err;
    }
    /* Enable GPT Timer */
    err = R_GPT_Enable(&g_timer_PWM_ctrl);
    /* Handle error */
    if (FSP_SUCCESS != err)
    {
        return err;
    }
    /* Start GPT timer */
    err = R_GPT_Start(&g_timer_PWM_ctrl);;
    if(FSP_SUCCESS != err)
    {
        return err;
    }

    return err;
}
```

Figure 18. gpt_timer_PWM

3. Adding and Configuring “Touch Function Driver”

1. Click “**New Thread**” and name “**Touch Thread**” with setting below.

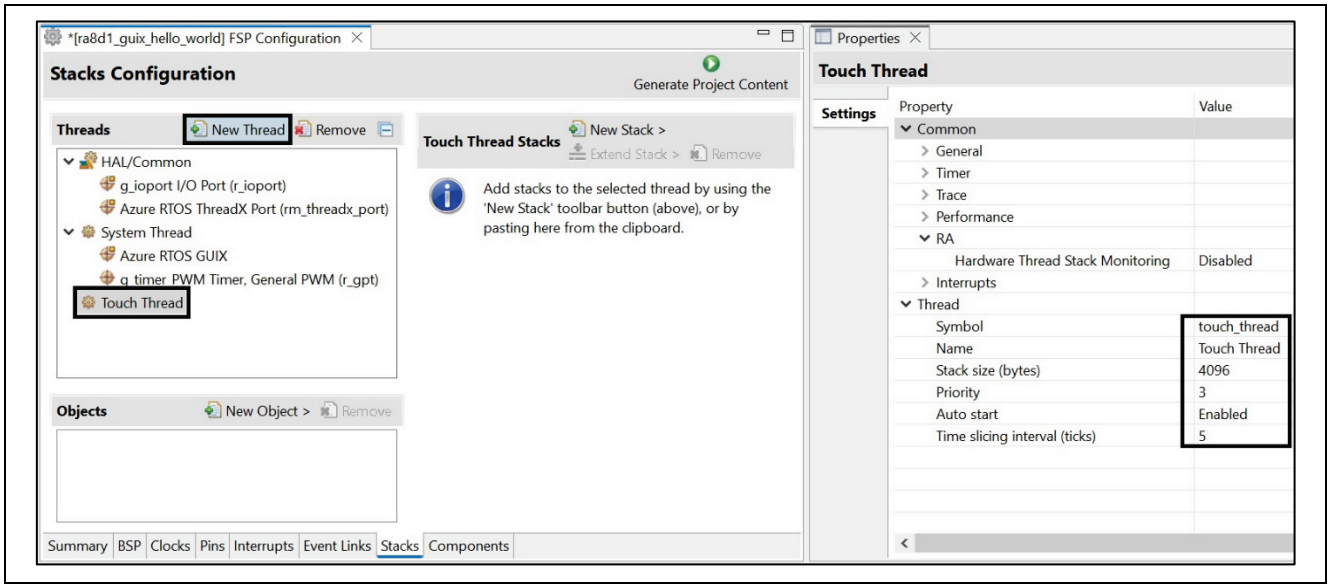


Figure 19. Add and Set TOUCH THREAD Properties

2. Click “**New Stack**” and add “**External IRQ**” module.

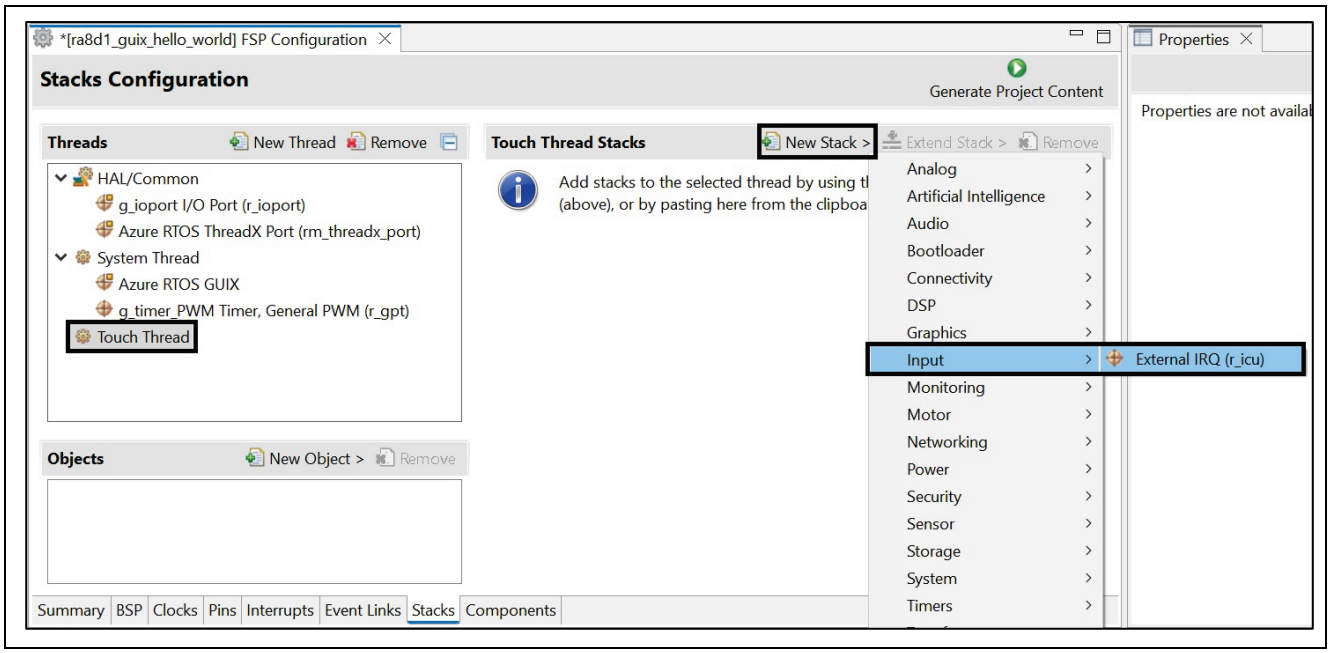


Figure 20. Add External IRQ

3. Name “**g_touch_irq**” g_touch_irq and setting **External IRQ** property.

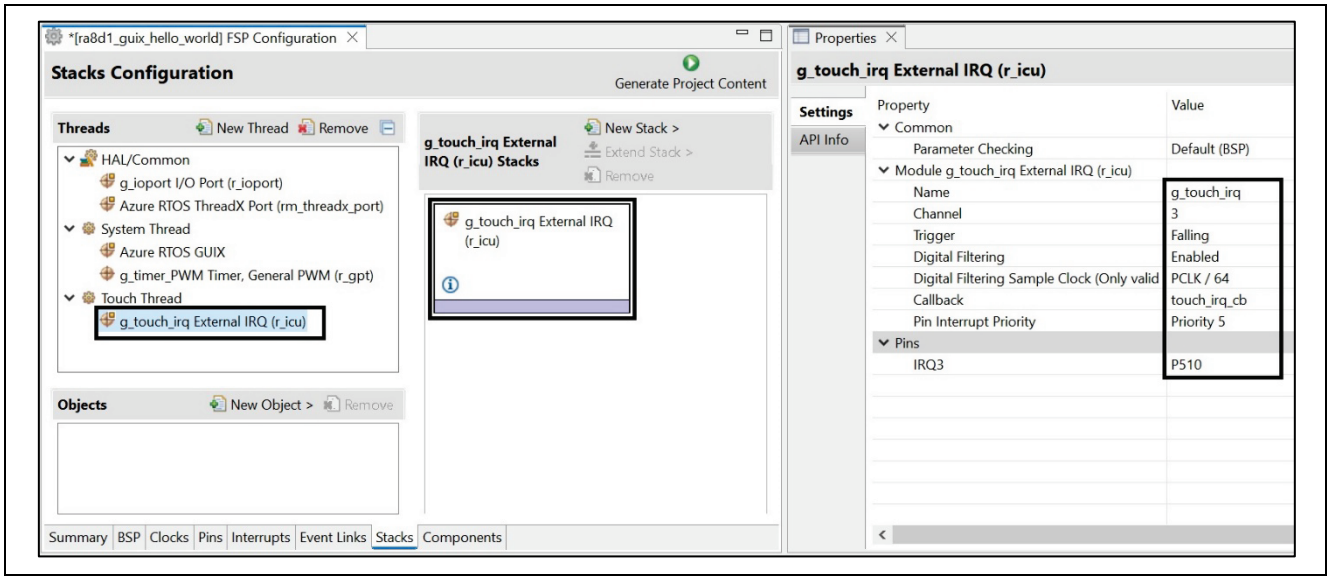


Figure 21. Setting External IRQ Properties

4. Configuration pin **P510** for **DISP_INT** signal pin.

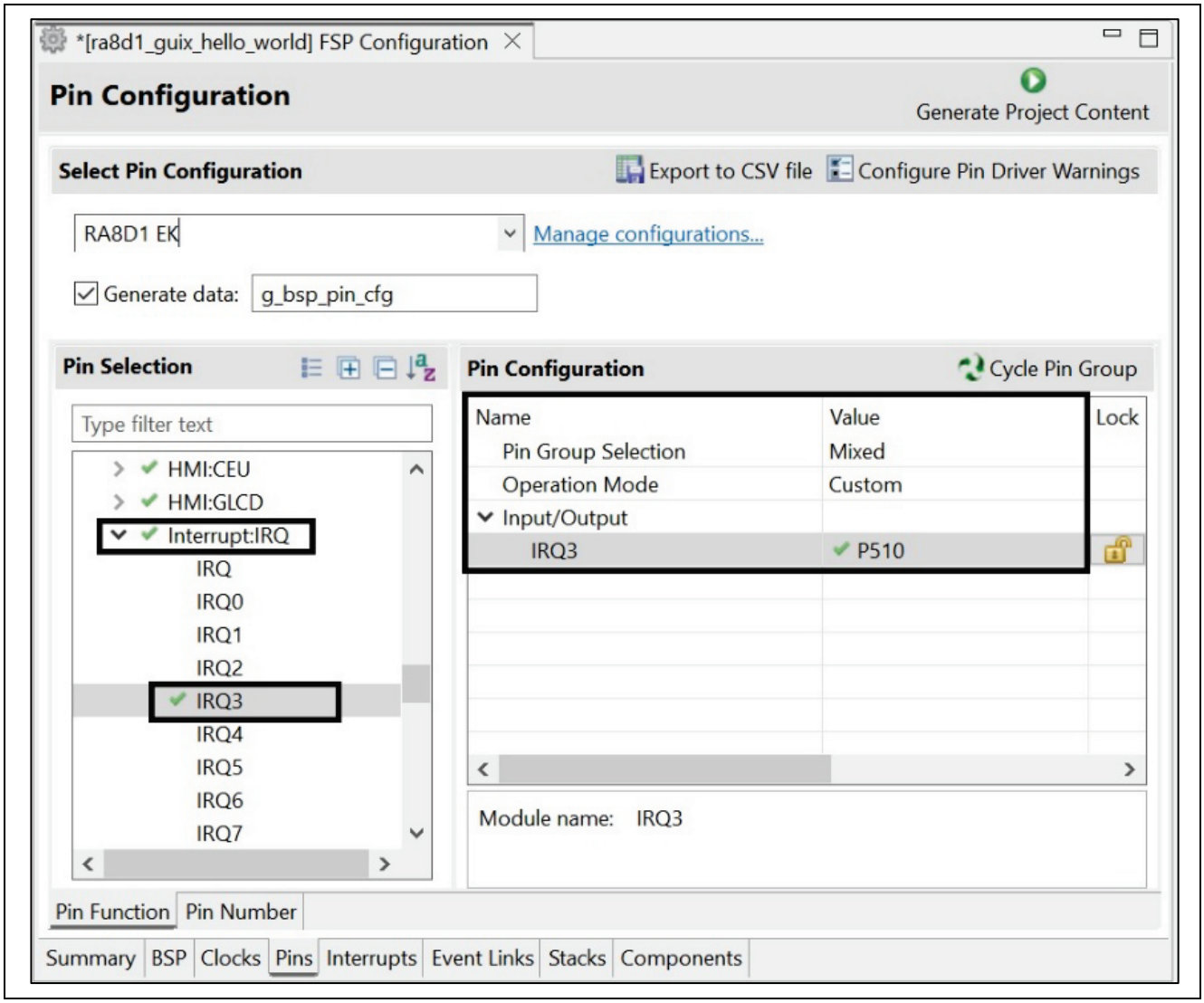


Figure 22. Configuration Pin P510 “IRQ3”

5. Click “**New Stack**” and add “**I2C Master**”.

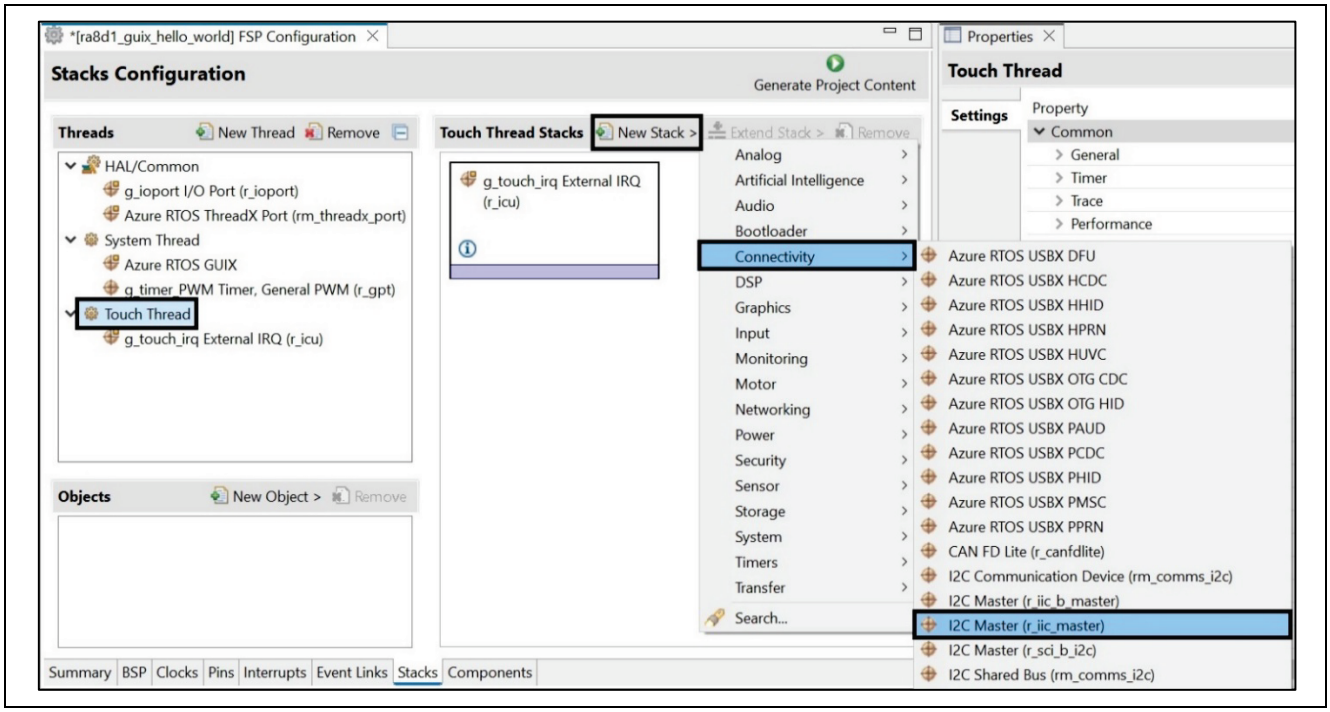


Figure 23. Add Touch I2C Master

6. Name “**g_i2c_touch**” and setting property.

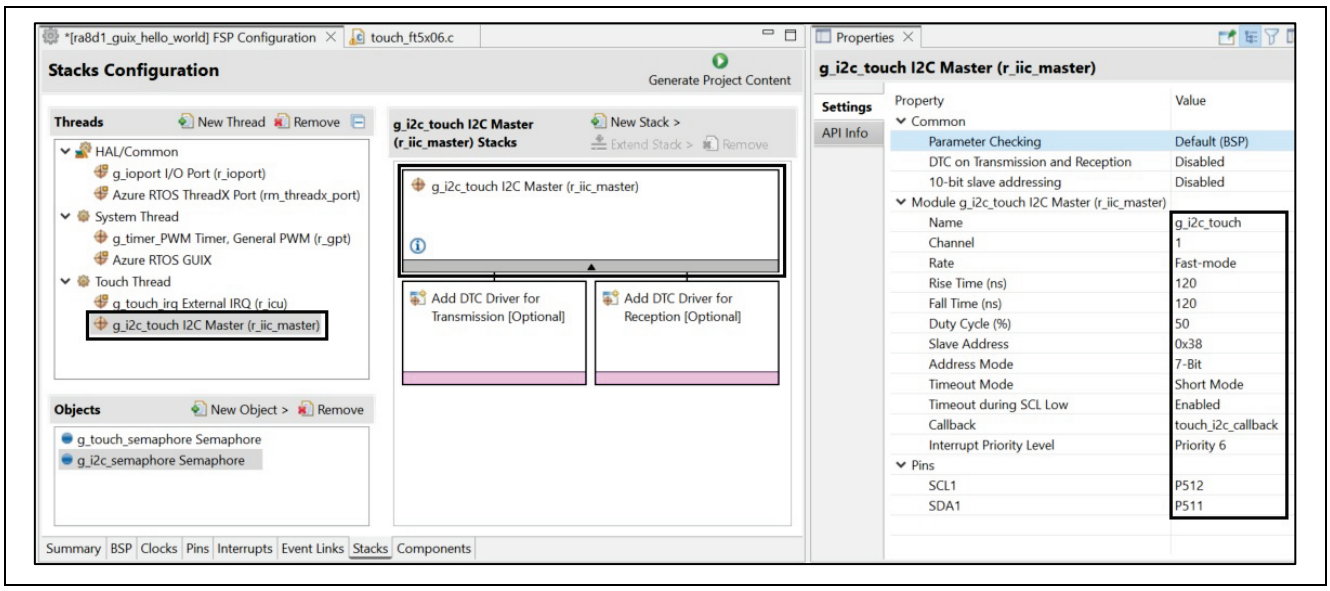


Figure 24. Name and Settings Property

7. Click “New Object” and add Semaphore.

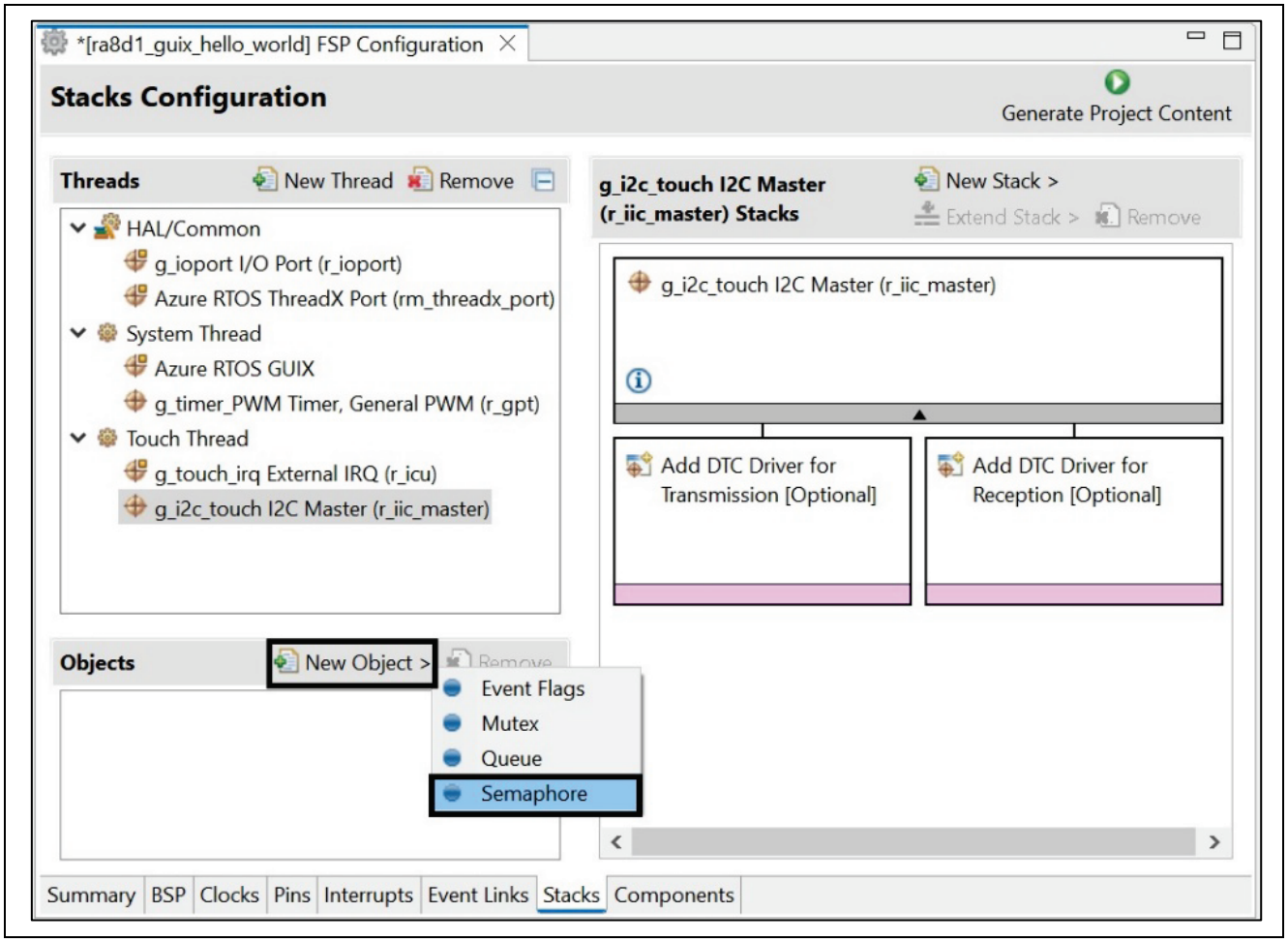


Figure 25. Add New Semaphore

8. Name “Touch Semaphore” and setting Property.

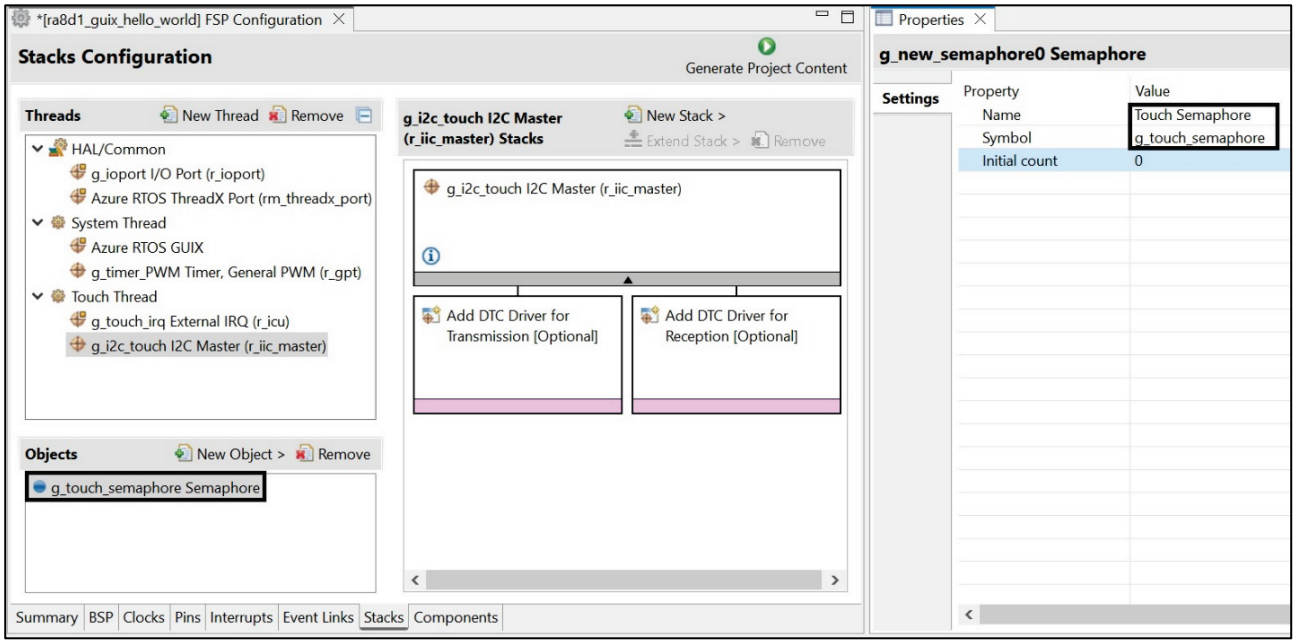


Figure 26. Add and Name Touch Semaphore

9. Click “New Object” and add another Semaphore

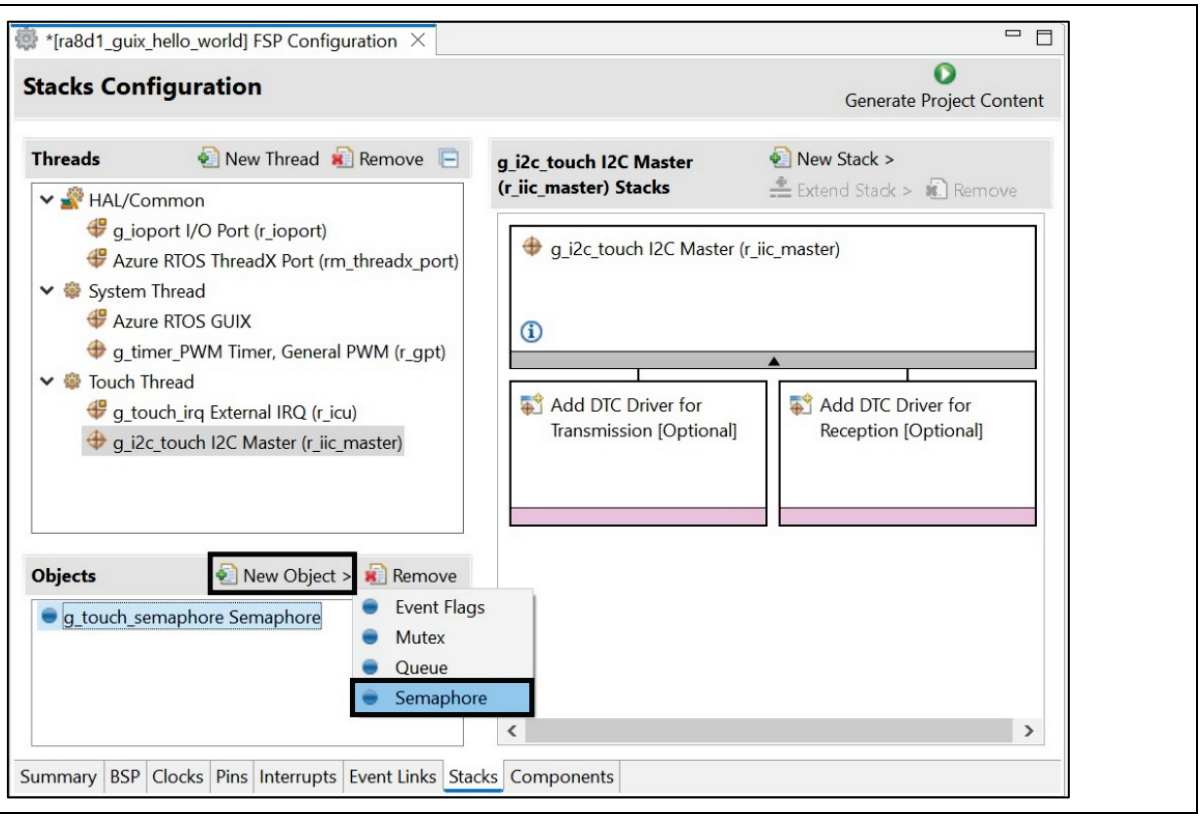


Figure 27. Add Another New Semaphore

10. Name “**I2C Semaphore**” and setting Property.

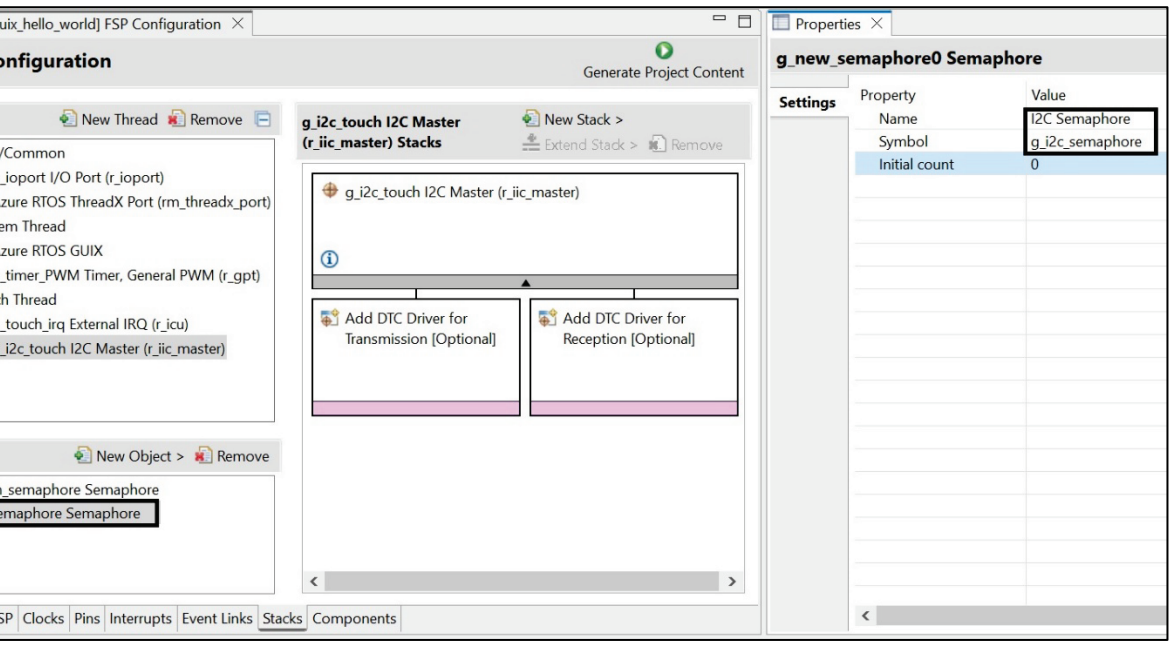


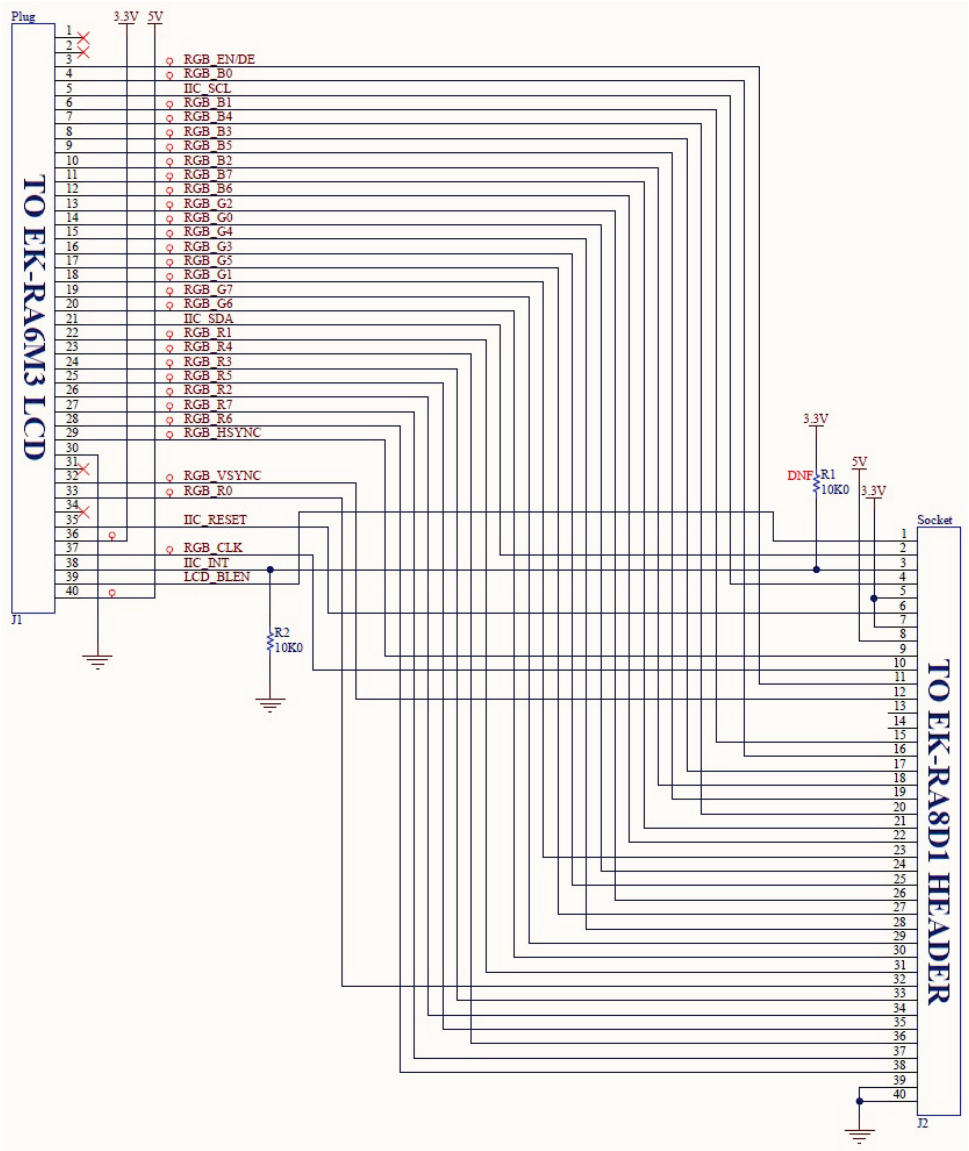
Figure 28. Add and Name I2C Semaphore

11. Wiring RA6M3G’s LCD pins connector “NC1” to the pins header “J57” of EK-RA8D1 as shown below.

RA6M3 LCD Board needs to modify			**Pins Connector ON RA8D1 Board**		
Pins# on LCD board	**LCD Pin Name**			**Pins # on RA8D1 board**	**RA8D1 Pin Name**
Connector		Connector	Connect to	Connector J57	

RA6M3 LCD Board needs to modify			Pins Connector ON RA8D1 Board		
Pins# on LCD board	**LCD Pin Name**			**Pins # on RA8D1 board**	**RA8D1 Pin Name**
CN1 needs to modify		NC1			
1	OPEN	1			
2	OPEN	2			
3	DE = TCON2	3	→	11	LCDC_TCON2
4	RGB_B0	4	→	16	LCDC_DATA00
5	SCL	5	→	4	IIC_SCL
6	RGB_B1	6	→	15	LCDC_DATA01
7	RGB_B4	7	→	20	LCDC_DATA04
8	RGB_B3	8	→	17	LCDC_DATA03
9	RGB_B5	9	→	19	LCDC_DATA05
10	RGB_B2	10	→	18	LCDC_DATA02
11	RGB_B7	11	→	21	LCDC_DATA07
12	RGB_B6	12	→	22	LCDC_DATA06
13	RGB_G2	13	→	26	LCDC_DATA10
14	RGB_G0	14	→	24	LCDC_DATA08
15	RGB_G4	15	→	28	LCDC_DATA12
16	RGB_G3	16	→	25	LCDC_DATA11
17	RGB_G5	17	→	27	LCDC_DATA13
18	RGB_G1	18	→	23	LCDC_DATA09
19	RGB_G7	19	→	29	LCDC_DATA15
20	RGB_G6	20	→	30	LCDC_DATA14
21	IIC_SDA	21	→	2	IIC_SDA
22	RGB_R1	22	→	31	LCDC_DATA17
23	RGB_R4	23	→	36	LCDC_DATA20
24	RGB_R3	24	→	33	LCDC_DATA19
25	RGB_R5	25	→	35	LCDC_DATA21
26	RGB_R2	26	→	34	LCDC_DATA18
27	RGB_R7	27	→	37	LCDC_DATA23
28	RGB_R6	28	→	38	LCDC_DATA22
29	TCON0 = HSYNC	29	→	9	LCDC_TCON0
30	GND	30	→	39, 40	GND
31	OPEN	31			
32	TCON4 = VSYNC	32	→	12	LCDC_TCON1
33	RGB_R0	33	→	32	LCDC_DATA16
34	OPEN	34			
35	IIC_RST	35	→	6	DISP_RST
36	=+V3.3	36	→	5, 7	=+V3.3
37	RGB_CLK	37	→	10	CLK
38	IIC_INT	38	→	3	DISP_INT
39	LCD_BLEN	39	→	1	DISP_BLEN
40	=+V5.0	40	→	8	=+V5.0

Important note for touch function: User needs to add 10K Ohm resistor to R1 on connection board (40 pins board connect between LCD and J57 pins connector)



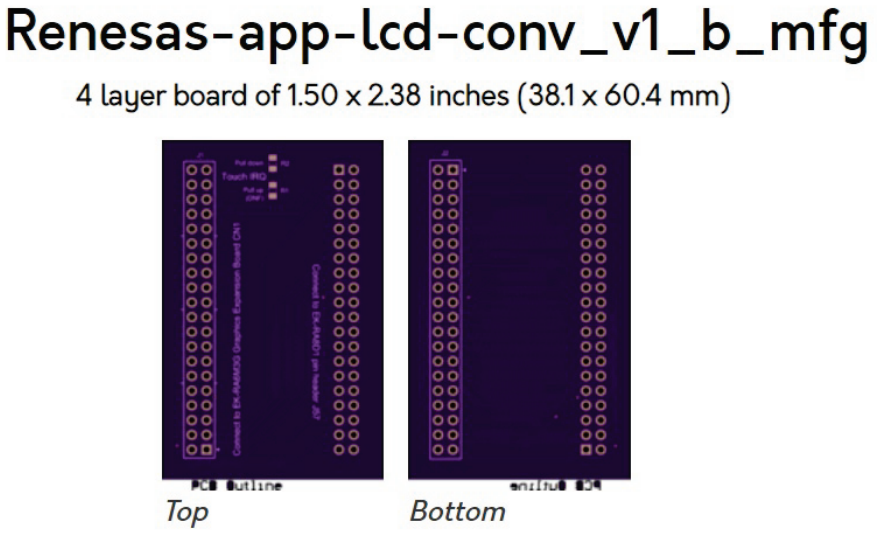


Figure 29. Schematic and Board Connection between LCD to J57 Connector on EK-RA8D1 Board

12. LCD Module pins connection

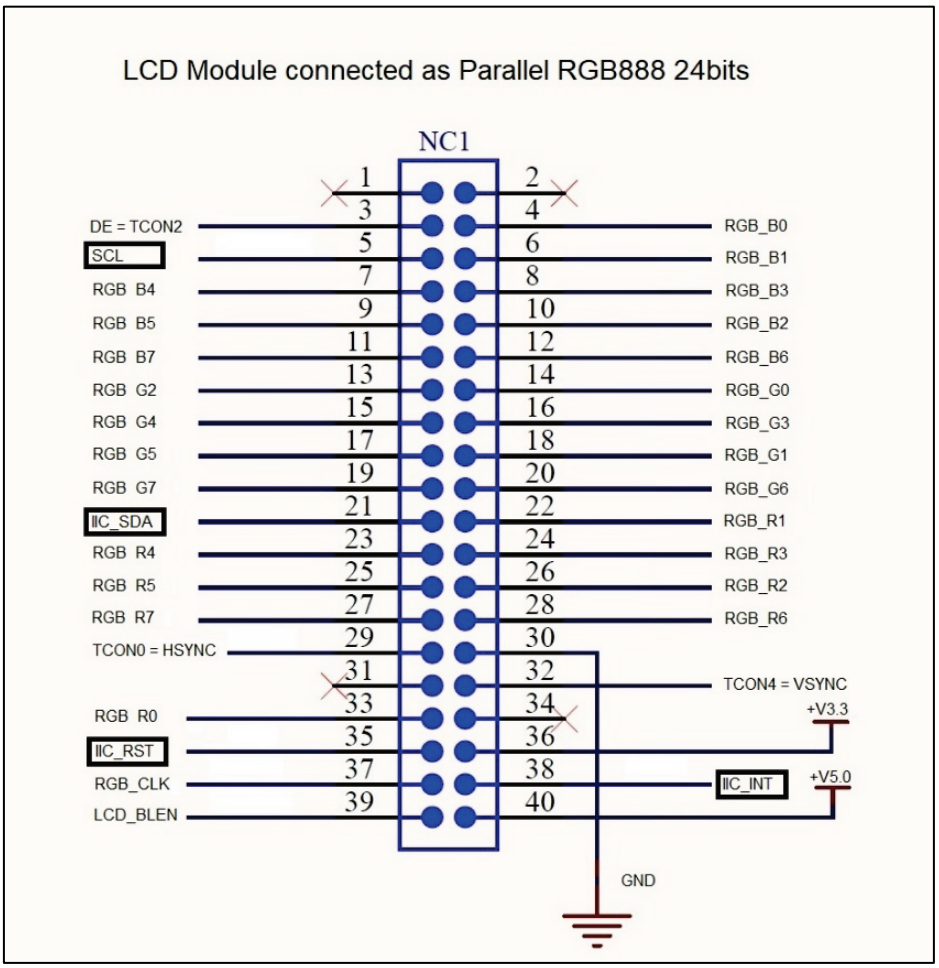


Figure 30. Set g_i2c_semaphore Semaphore Properties

The pins marked in a black square above are used for the touch panel controller on the LCD board:

- DISP_INT interrupt (P510) is used to trigger touch events.
- I2C channel 1 (P512, P511) is used to read and write data to the touch controller.
- Touch driver folder `touch_ft5x06` for touch function is inside the provided `Source` folder.
- PA01 is used to reset the LCD's touch controller.

13. Note: Refer to the `touch_thread_entry.c` file in `Source.zip` for more information. The following code initializes the touch controller and process touch events.

```
/* Touch Thread entry function */
void touch_thread_entry(void)
{
    GX_EVENT gxe = {0};
    UINT status  = 0;
    /*
     * Initialize Touch controller:
     *  FT5X06 driver
     *  IRQ interrupt
     */
    ft5x06_init(&g_i2c_touch, &g_i2c_semaphore, BSP_IO_PORT_10_PIN_01);
    /* Enable touch IRQ */
    R_ICU_ExternalIrqOpen(g_touch_irq.p_ctrl, g_touch_irq.p_cfg);
    R_ICU_ExternalIrqEnable(g_touch_irq.p_ctrl);

    while (1)
    {
        /* Wait for IRQ from touch controller. */
        status = tx_semaphore_get(&g_touch_semaphore, TX_WAIT_FOREVER);
        if(TX_SUCCESS != status)
        {
            APP_ERR_TRAP(FSP_ERR_ASSERTION);
        }

        /* Get touch data from the FT5X06 */
        ft5x06_payload_get(&touch_data);

        if(0U < touch_data.num_points)
        {
            /* Send only the TOUCH event of the first finger to the GUIX Model View Controller. */
            if( TOUCH_EVENT_DOWN == touch_data.point[0].event)
            {
                gxe.gx_event_payload.gx_event_pointdata.gx_point_x       = touch_data.point[0].x;
                gxe.gx_event_payload.gx_event_pointdata.gx_point_y       = touch_data.point[0].y;
                gxe.gx_event_type = GX_EVENT_PEN_DOWN;
                gx_system_event_send(&gxe);

            }
            else if (TOUCH_EVENT_UP == touch_data.point[0].event && GX_EVENT_PEN_DOWN == gxe.gx_event_type)
            {
                gxe.gx_event_payload.gx_event_pointdata.gx_point_x       = touch_data.point[0].x;
                gxe.gx_event_payload.gx_event_pointdata.gx_point_y       = touch_data.point[0].y;
                gxe.gx_event_type = GX_EVENT_PEN_UP;
                gx_system_event_send(&gxe);
            }
```

Figure 31. Initializes the Touch Controller and Process Touch Events

14. From Stacks Configuration, click "**Generate Project Content**" to generate project content.

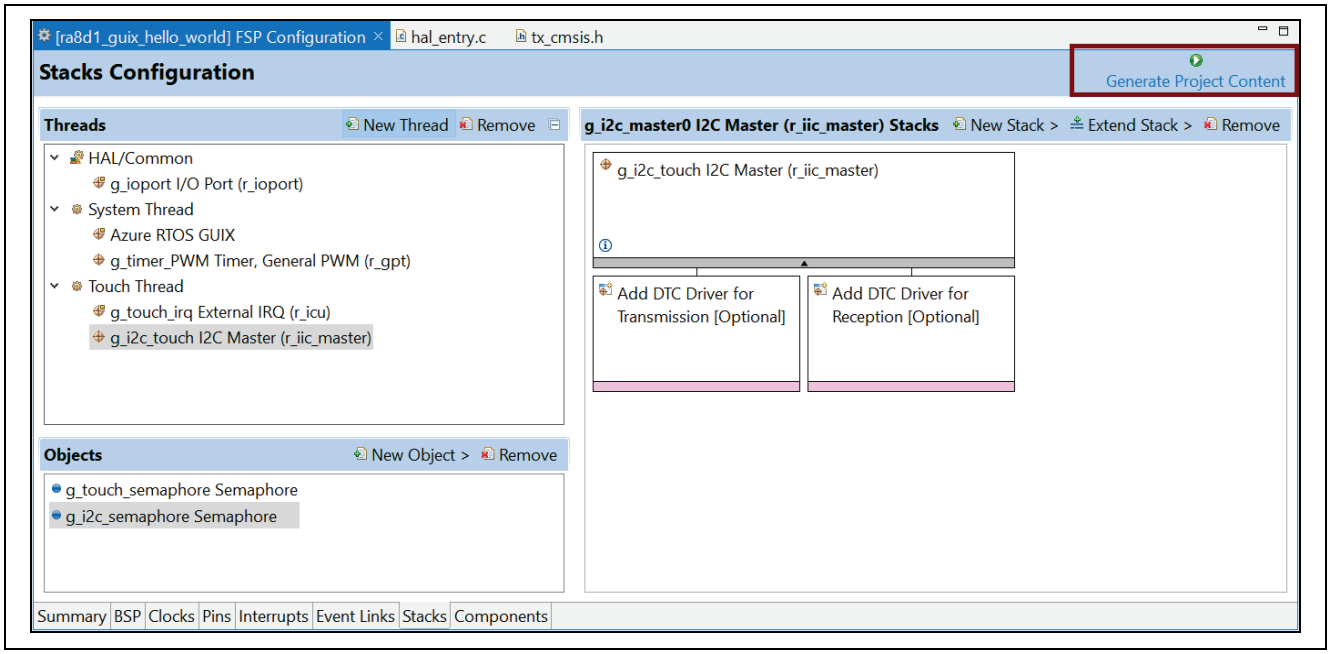


Figure 32. "Generate Project Content"

15. Unzip and open the provided folder `Source.zip`. Copy the four `*.c` files and one folder `touch_ft5x06` and paste into the folder `src` of your project "**ra8d1_guix_hello_world**".
    - Touch_ft5x06 folder
    - hal_entry.c
    - system_thread_entry.c
    - touch_thread_entry.c
    - windows_handler.c

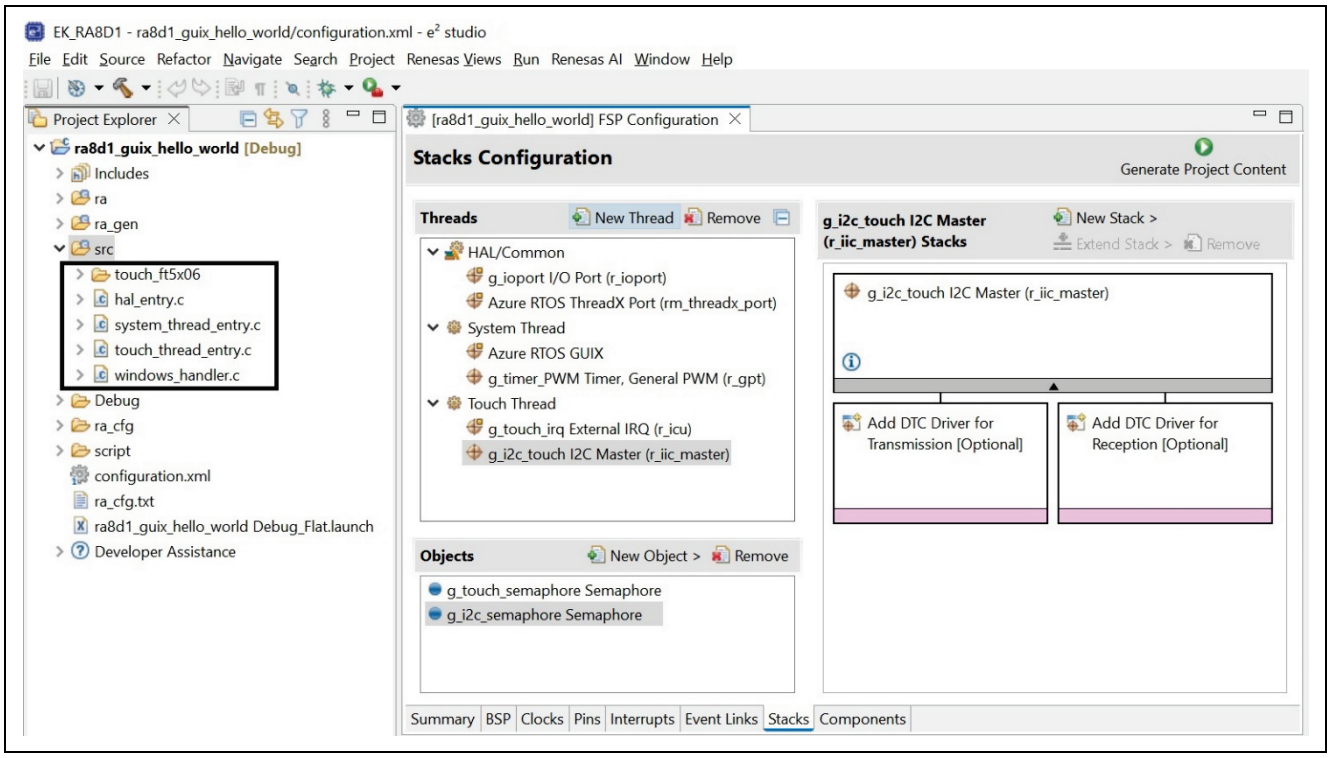


Figure 33. Copied Folder and Files

4. Creating Folders in the Hello_World GUIX_EK_RA8D1 Project for Azure RTOS GUIX Studio Project

1. Under folder `src` create a new folder and name it `guix_gen`. Follow the image below, then click **Finish**.

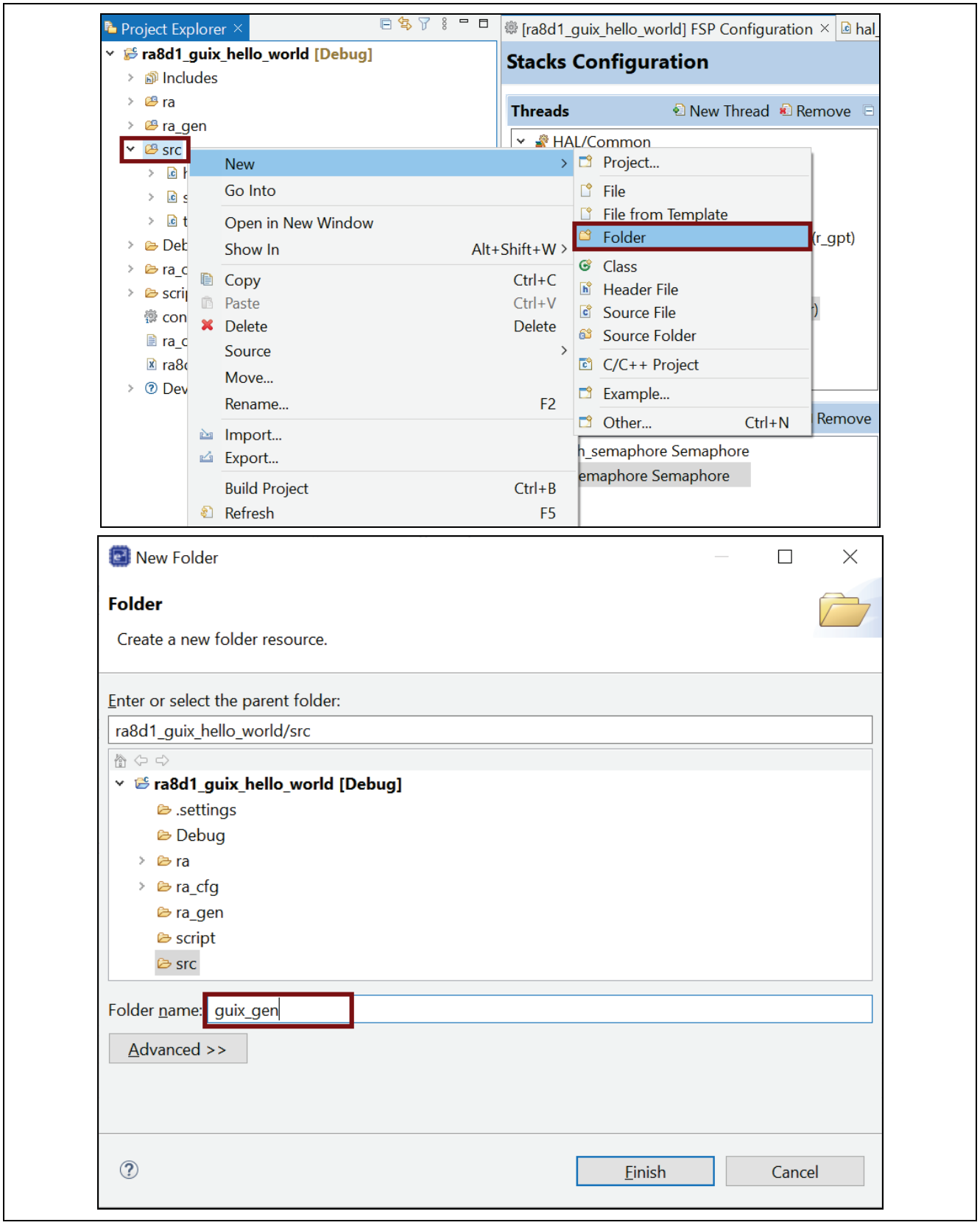


Figure 34. Create and Name “guix_gen” Folder

2. Under folder `src` create a new folder and name it `guix_studio`. Then follow the image below and click **Finish**.

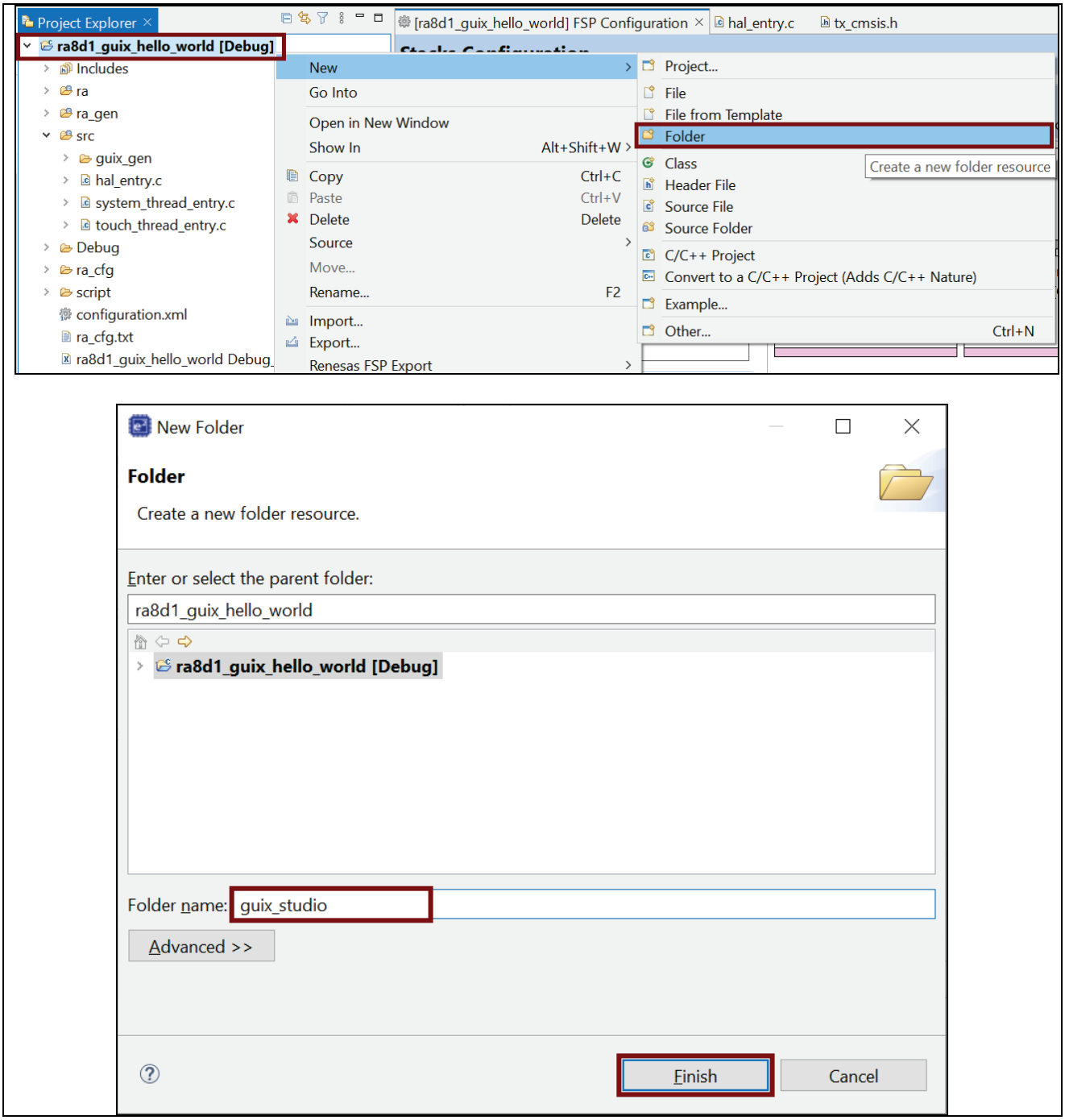


Figure 35. Create New Folder and Name “guix_studio”

3. Under folder `guix_studio` that was created earlier, create a new folder and name `GNU`. Follow the image below, then click **Finish.**

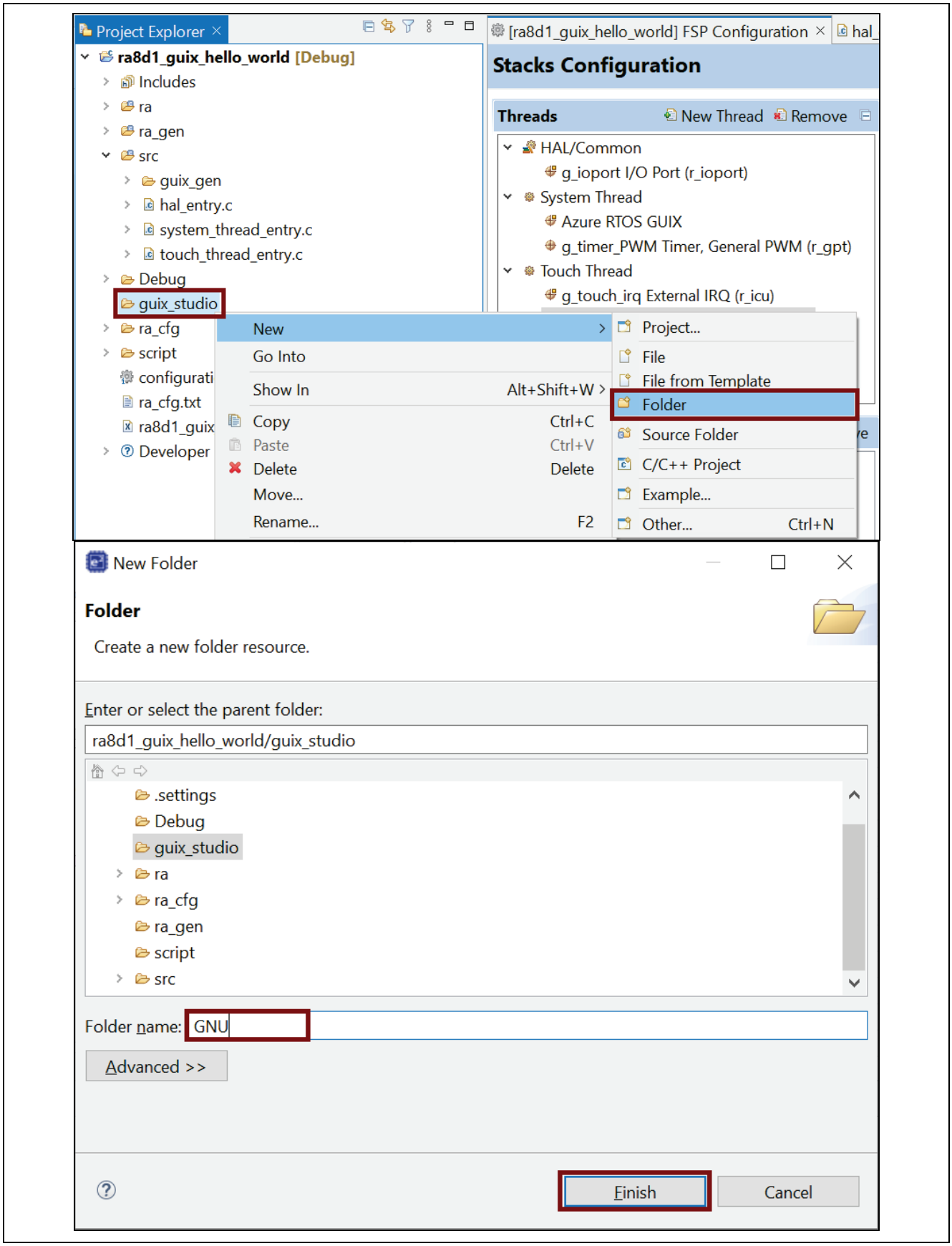


Figure 36. Create New Folder and Name “GNU”

4. After the sub-folder `GNU` is created, the folder structure should look like the image below.

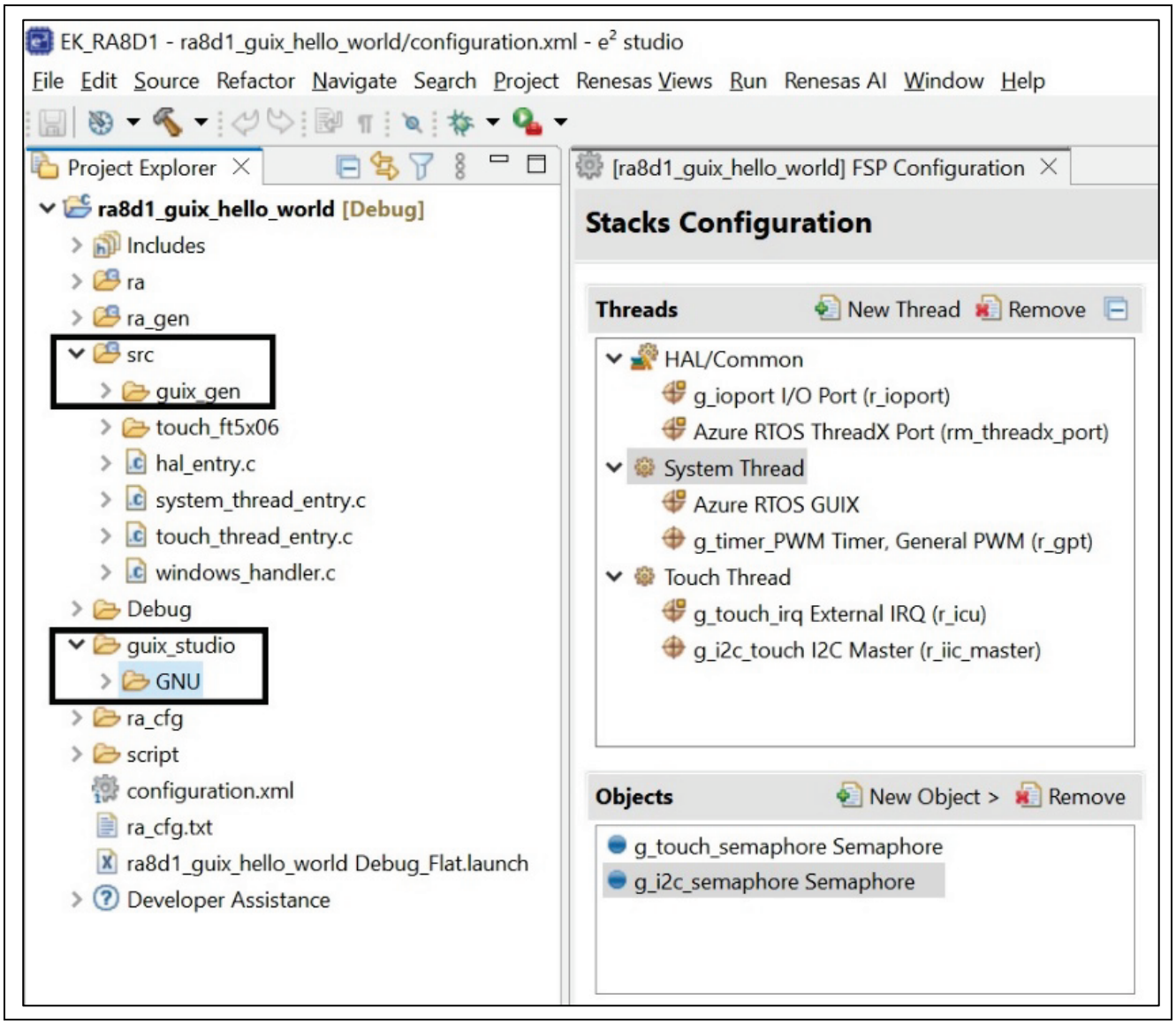


Figure 37. Project with New Folder

5. Using Azure RTOS GUIX Studio create GUI Windows

1. Open Azure RTOS GUIX Studio v6.2.1.0 or greater.

Figure 38. GUIX Studio Icon

2. Create a **New Project** and name it **Hello World**.
   A. Select **Project**. From the drop-down list, select **New Project**.
   B. Project Name: **Hello_World**.
   C. Project Path: Browse to the location of the folder you created in the **ra8d1_guix_hello_world>**
   D. **guix_studio\GNU** as shown in the image below.
   E. Hit **OK** button and then the **Save** button to confirm your selections.

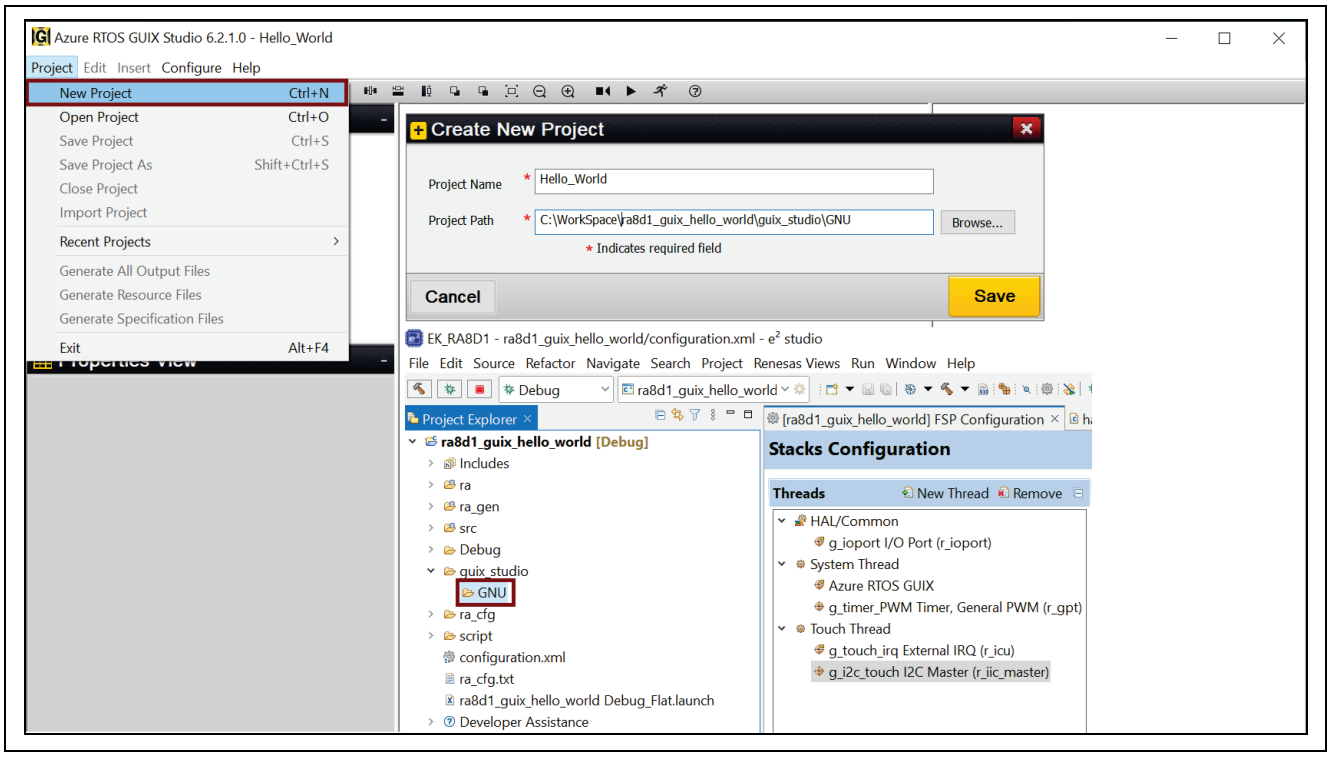


Figure 39. GUIX Studio Creates Hello World Project

3. A new **Configure Project** window will pop up and user needs to set all the options as shown in **Figure 40**. included Advanced Settings. Finally, click **Save**.

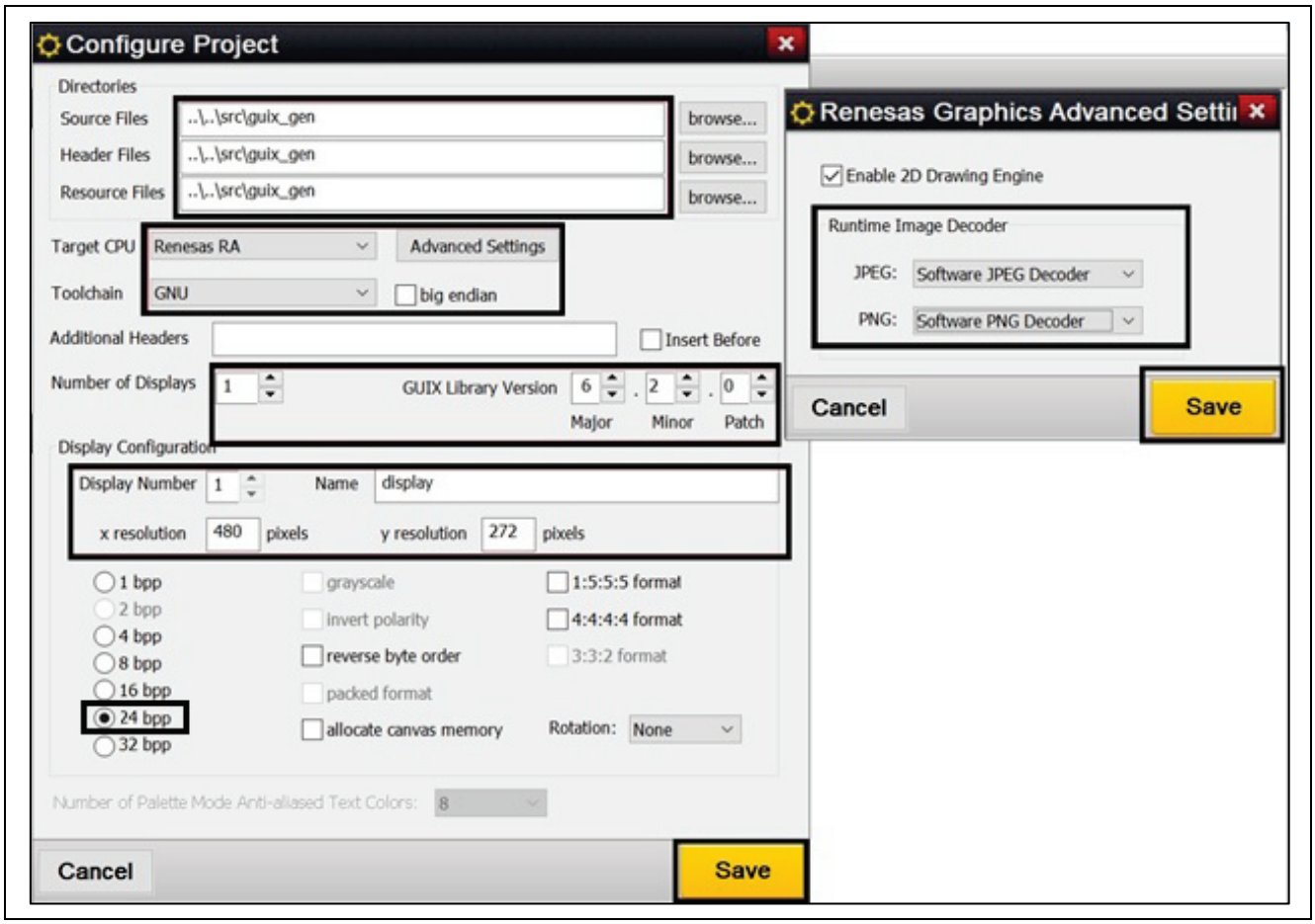


Figure 40. Configuration New Project "Hello World" with Advanced Settings

4. Star New Project “Hello World” should look similar Figure 41.

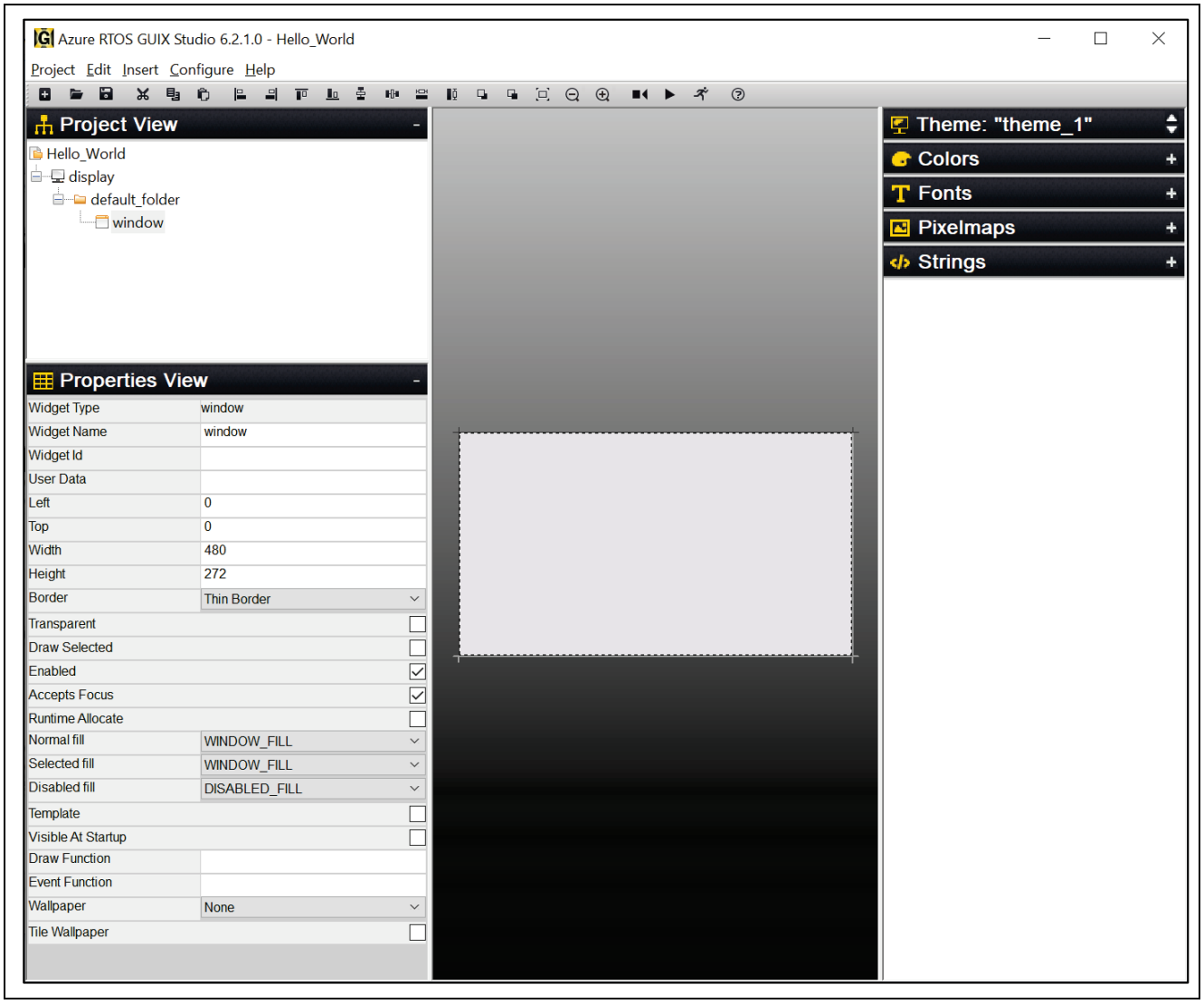


Figure 41. After Settings New Project “Hello World”

5. Setting Properties View of Window1.

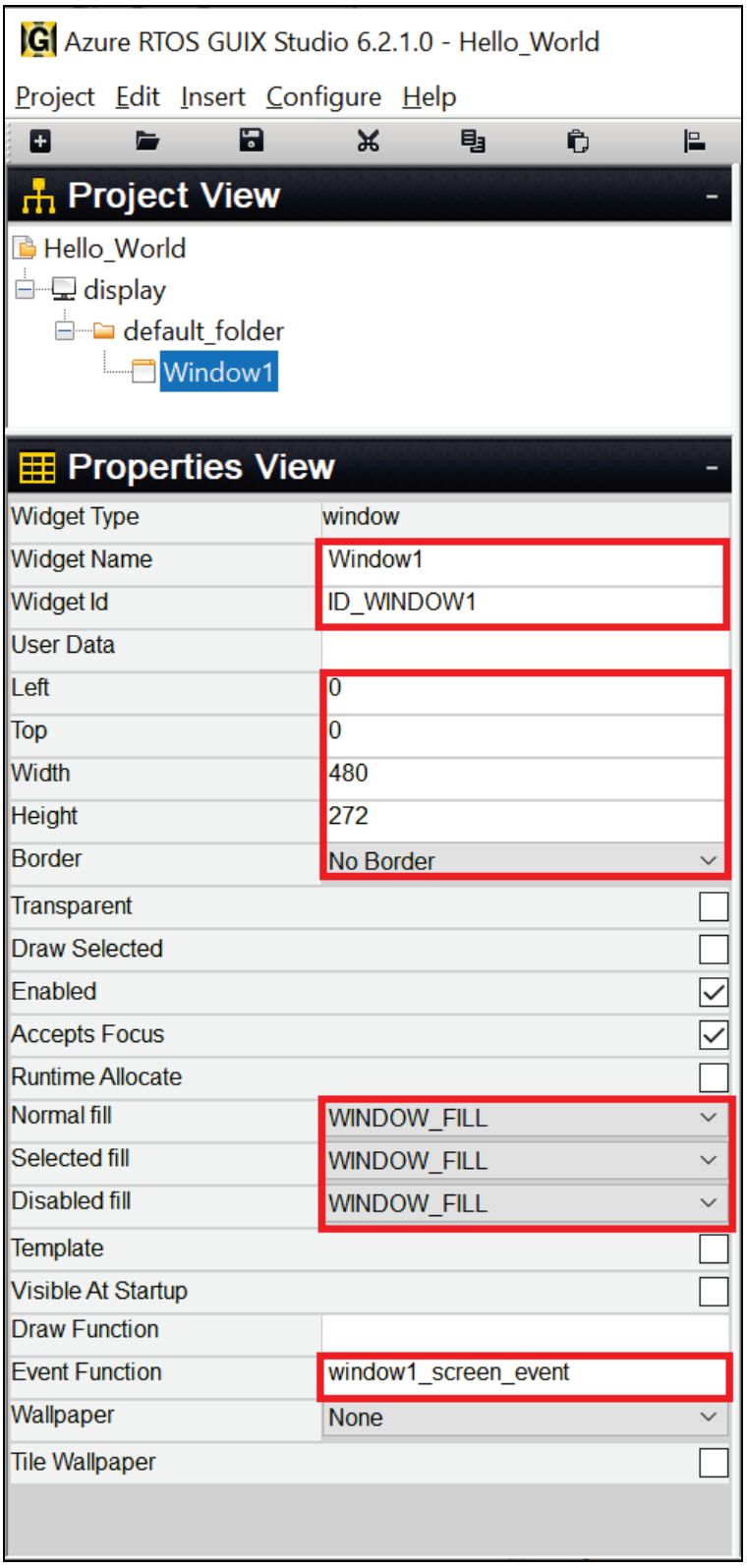


Figure 42. Property View of Window1

6. To add String ID, click on **Strings**. Follow the images below.

Figure 43. Strings

7. From the **Strings** dropdown, click **+ Add New String**.

Figure 44. Add New String

8. New **String Table Editor** window will pop up. Click the **Add String** button.

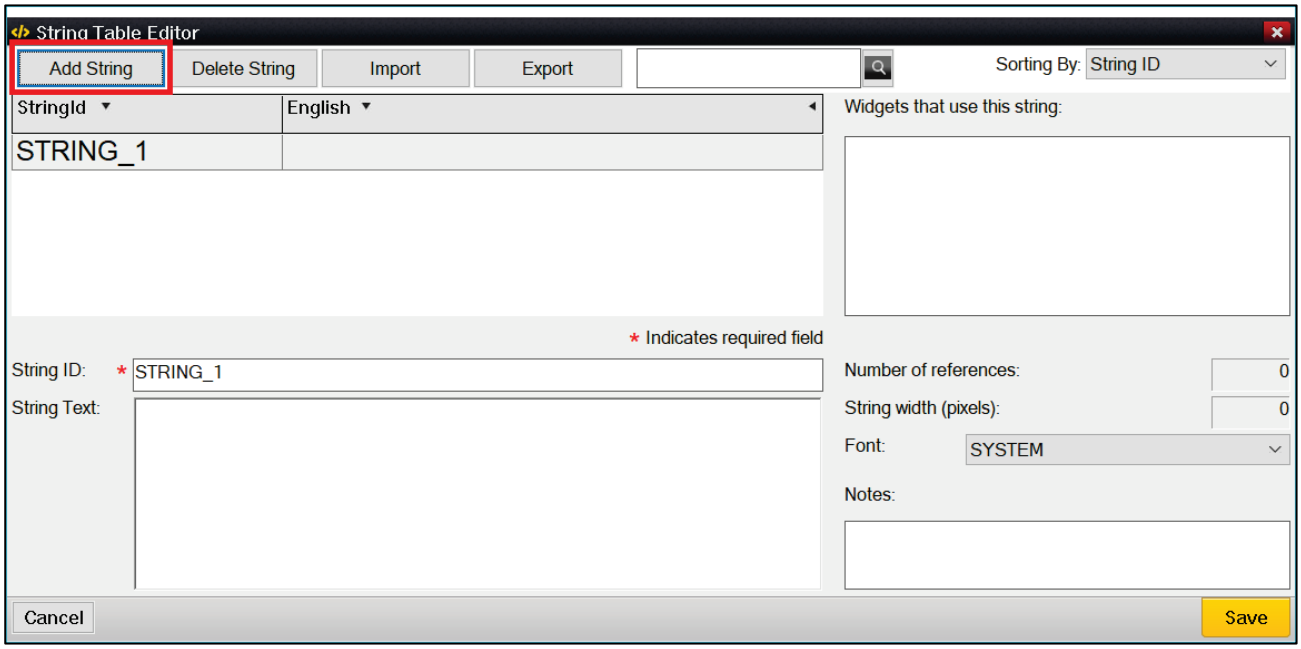


Figure 45. String Table Editor

9. Edit String ID and String Text.

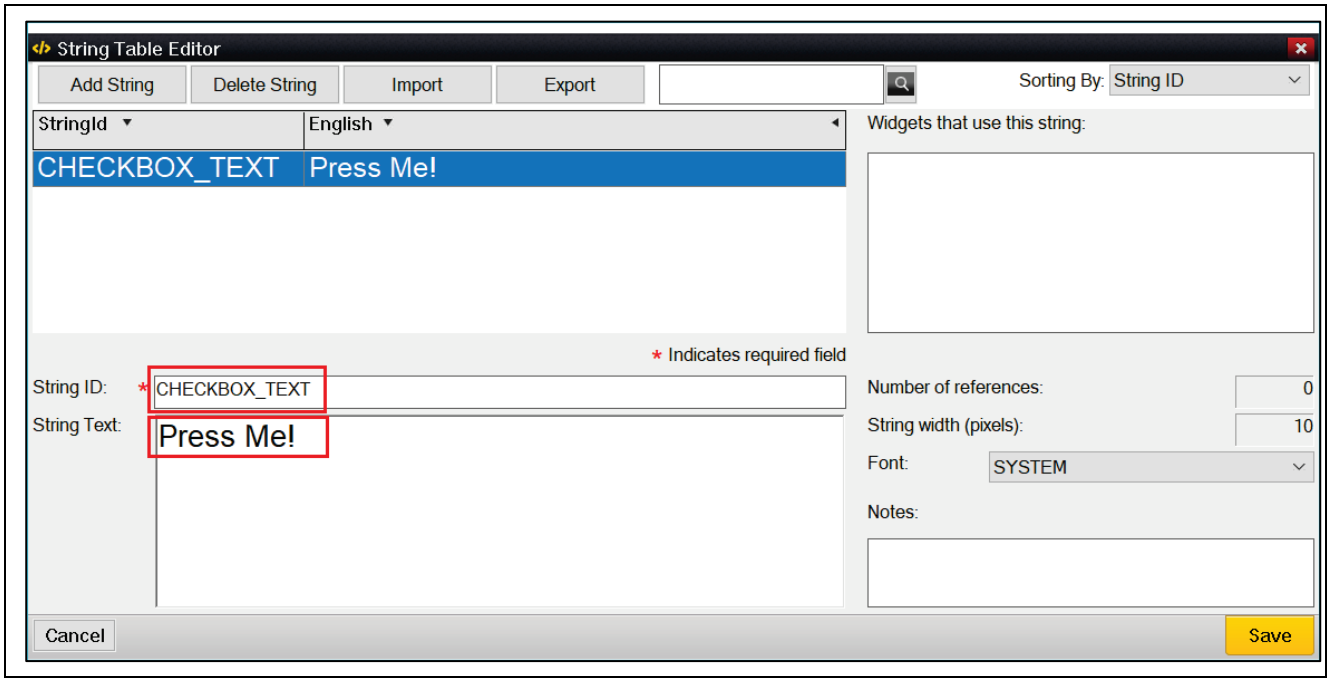


Figure 46. Edit String ID and String Text

10. Continue to click **Adding String**, then edit **String ID** and **String Text** until the table appears like **Figure 47** Then click the **Save** button.

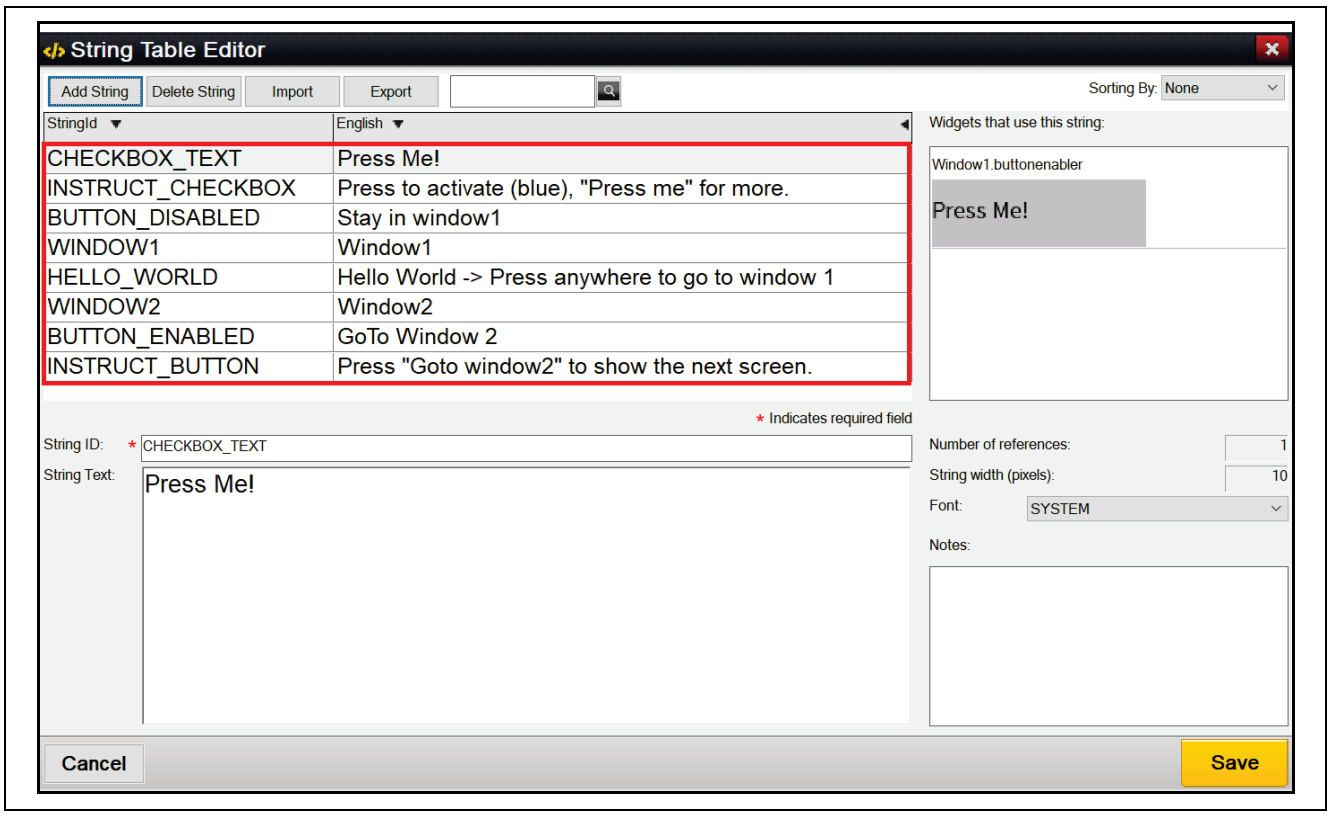


Figure 47. All Strings

11. Right-click on **Window1** to insert a **Text Button** and follow Figure 48.

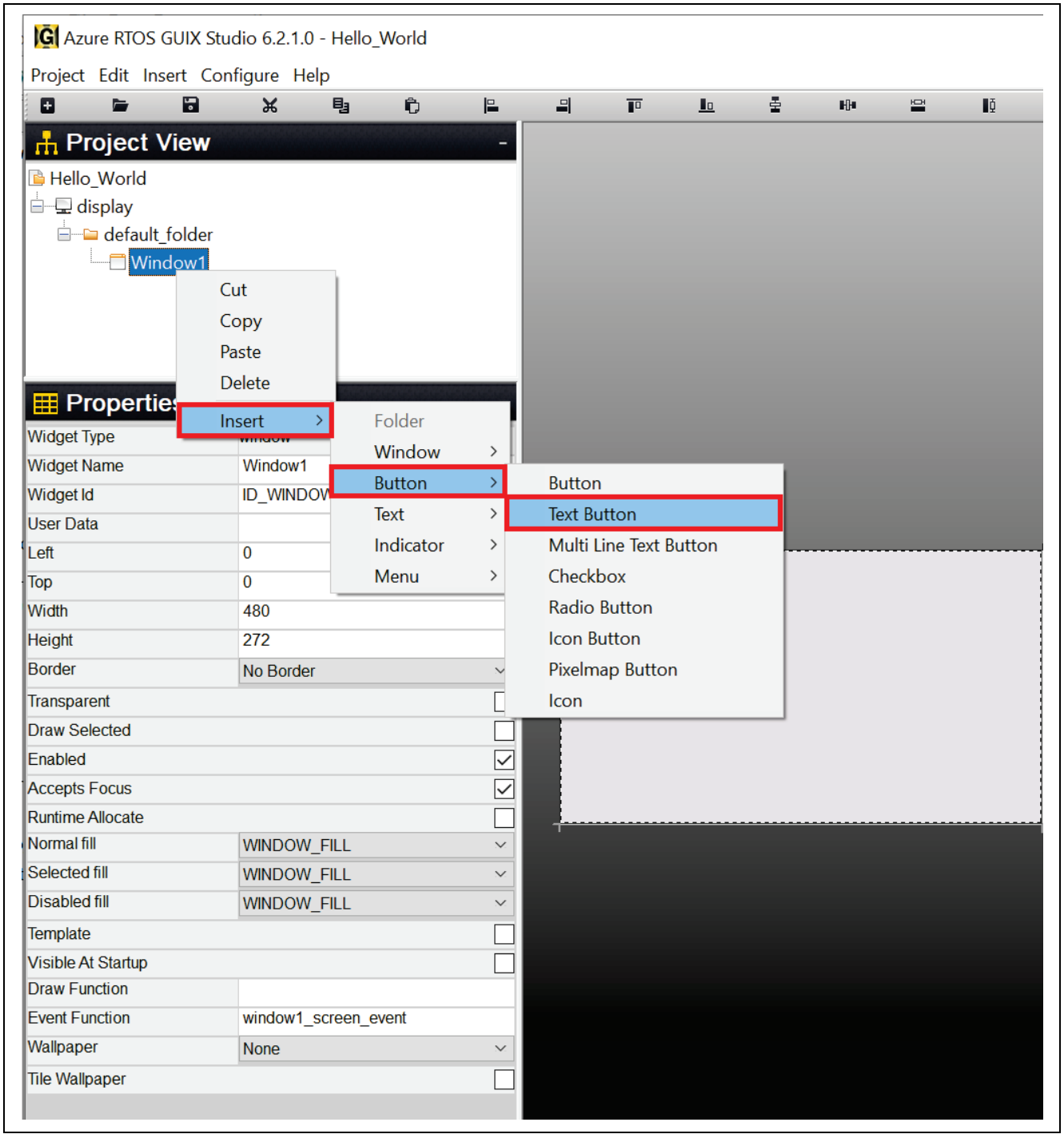


Figure 48. Insert Text Button

12. Set Properties View of “text_button”.

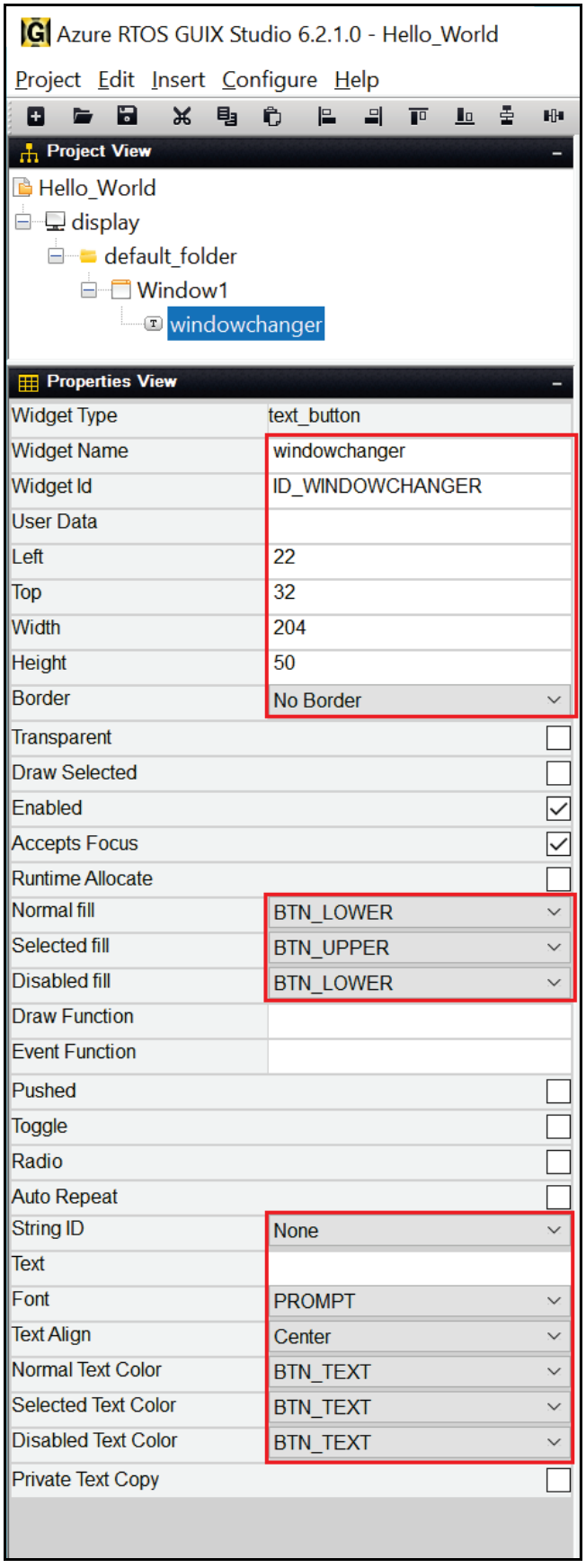


Figure 49. Properties View of text_button

13. Right-click **windowchanger** to insert a **Prompt** and follow **Figure 50**.

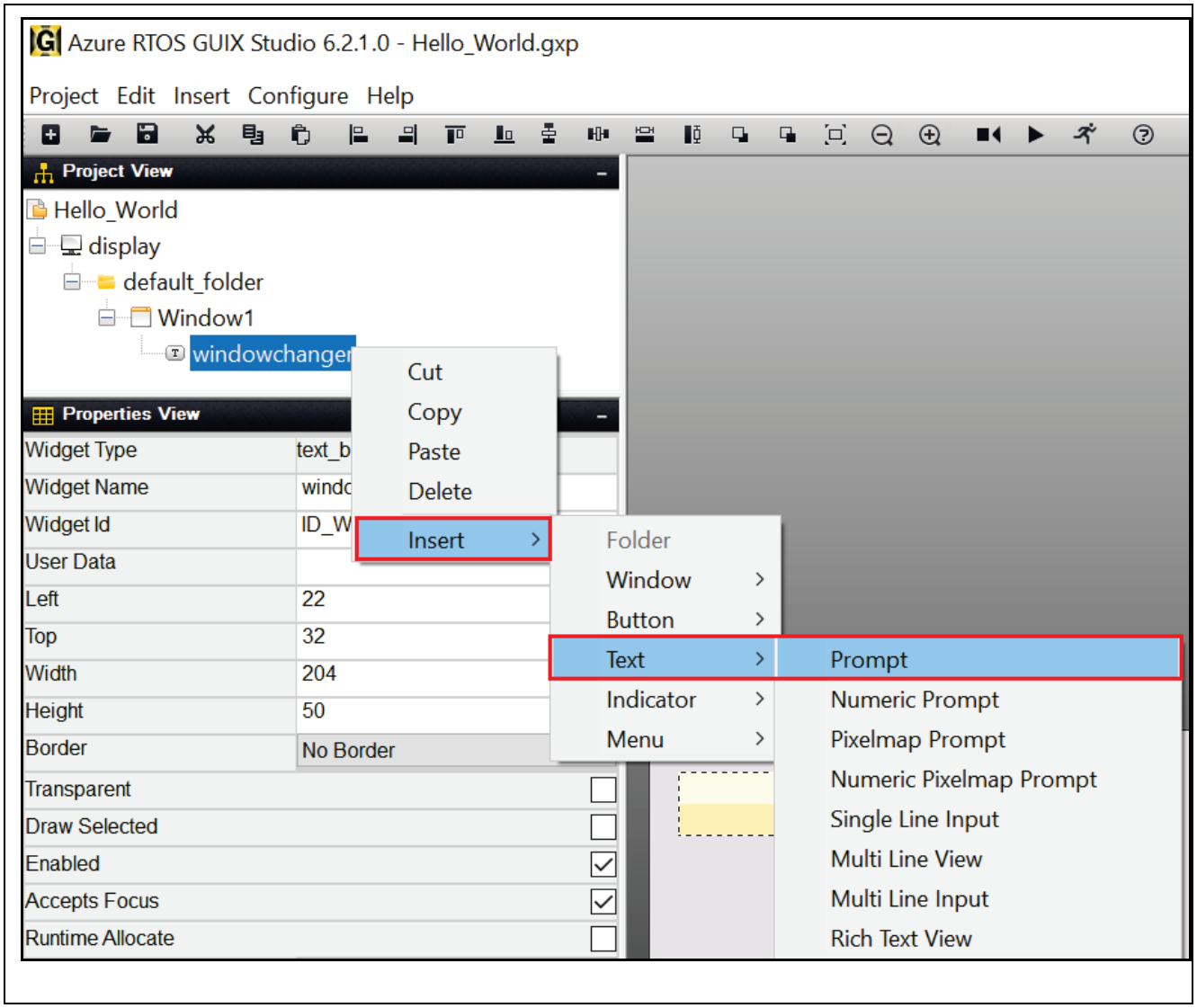


Figure 50. Insert Prompt

14. Set Properties View of prompt.

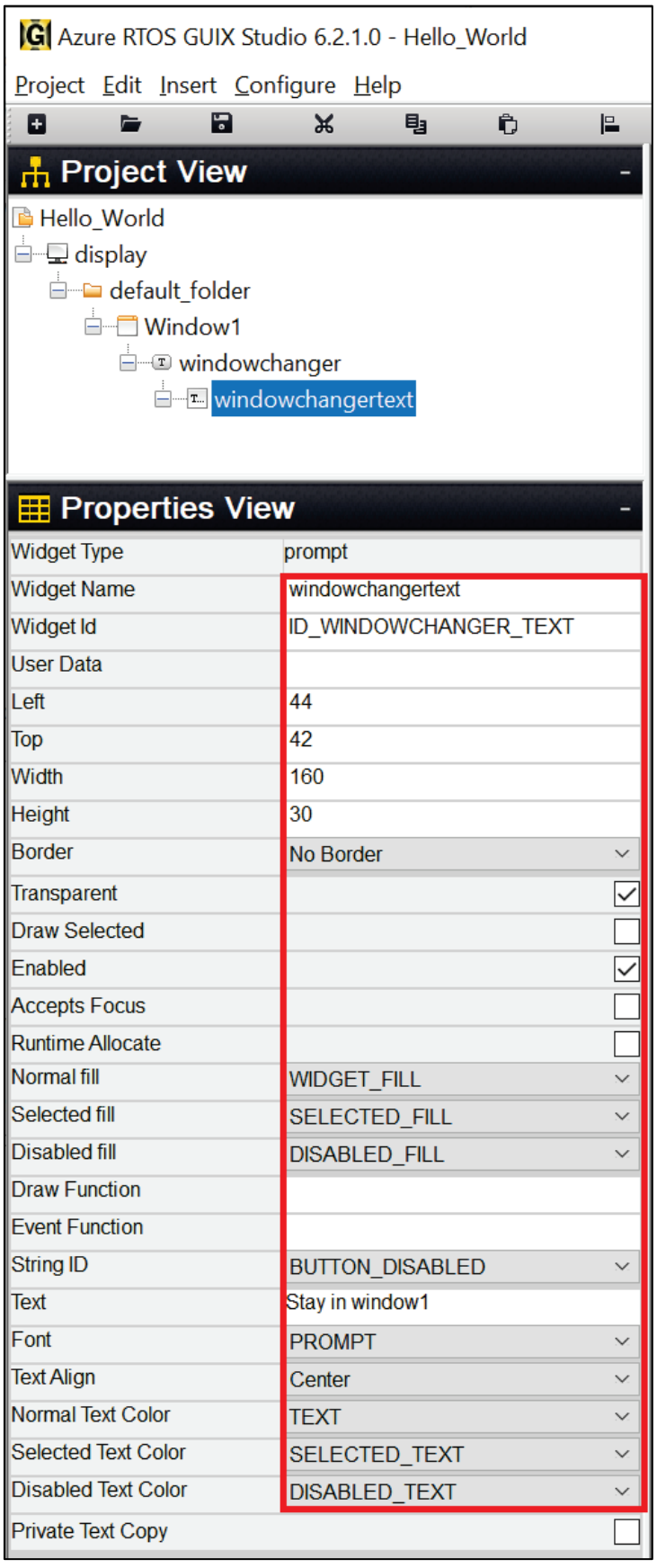


Figure 51. Properties View of Prompt

15. Insert new **Button**. Right-click on **windowchangertext** and follow **Figure 50**.

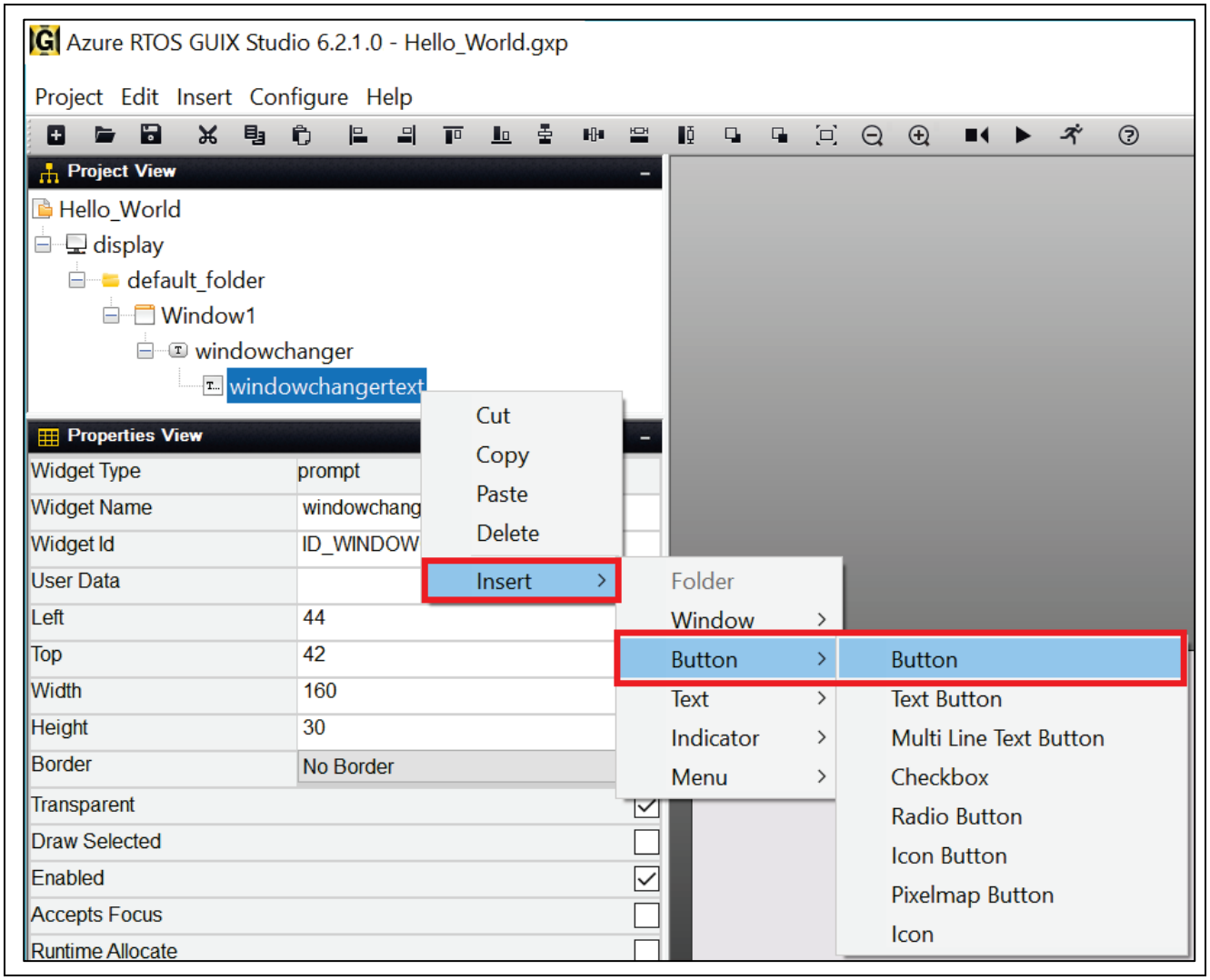


Figure 52. Insert Button

16. Set the Properties view of button.

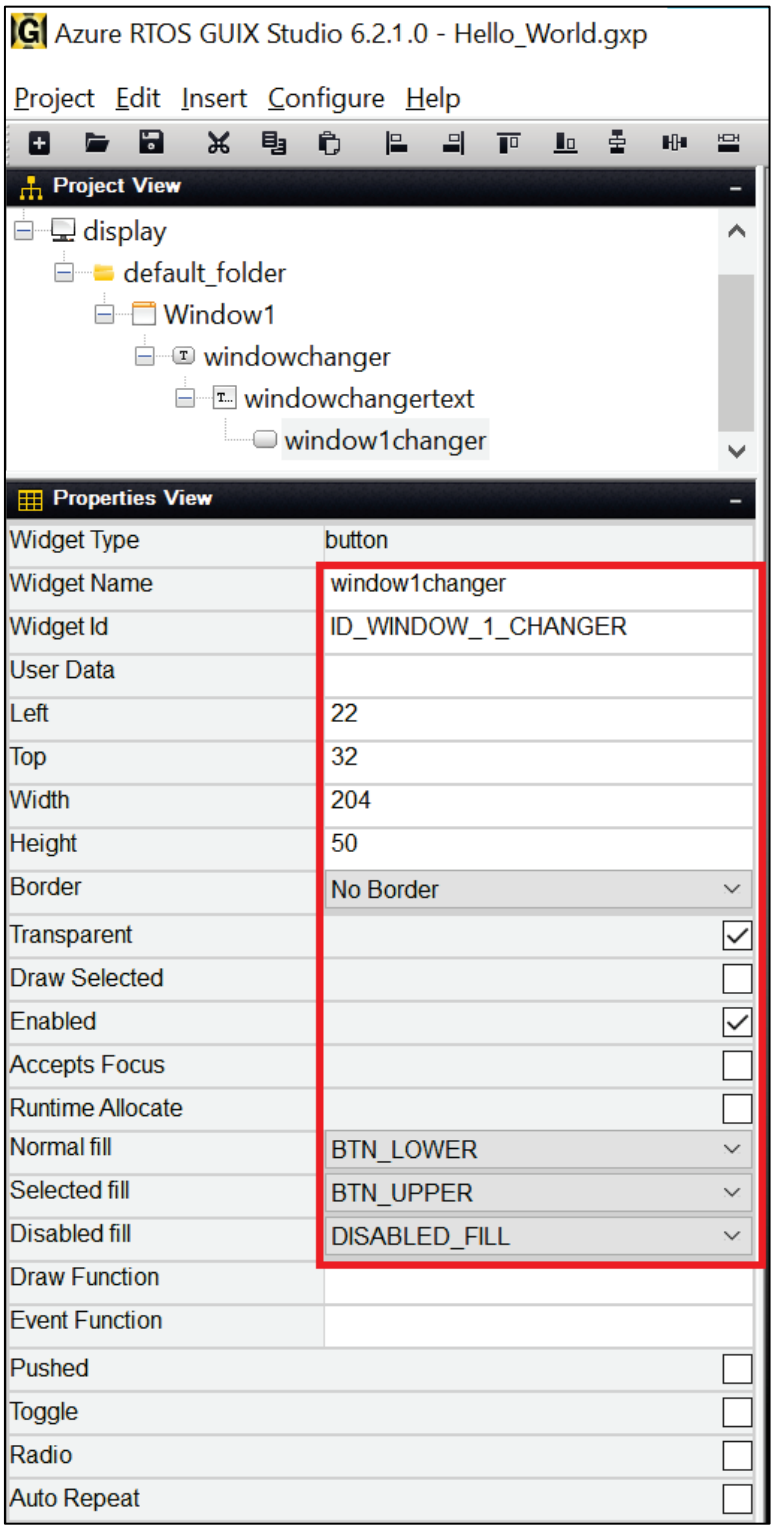


Figure 53. Properties View of Button

17. Insert **Prompt**. Right click on **Window1** and follow **Figure 54**. Insert two times to get two prompts. **Prompt** and **Prompt1** will show up once you finish.

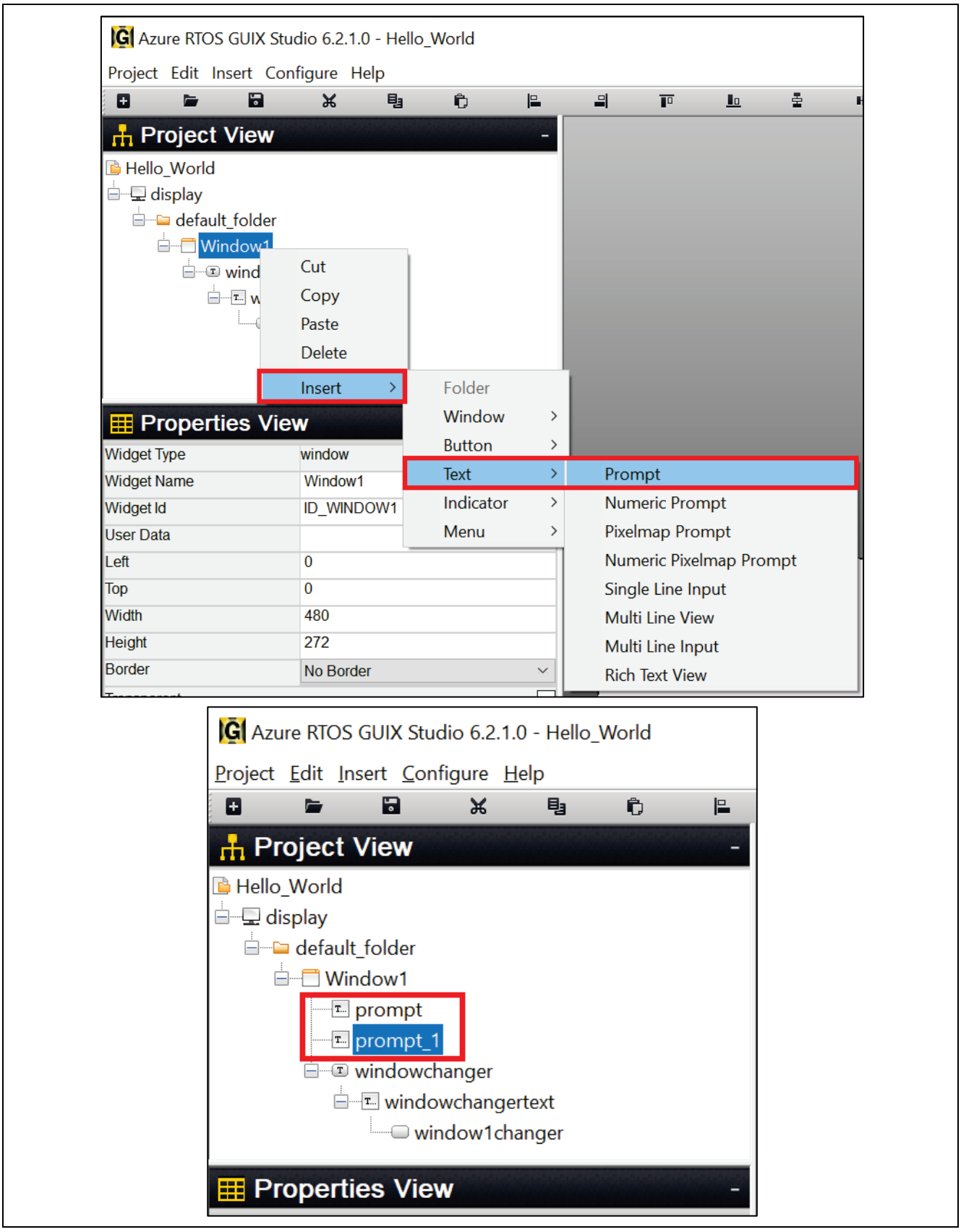


Figure 55. Insert Prompts

18. Set the Properties View of Prompt.

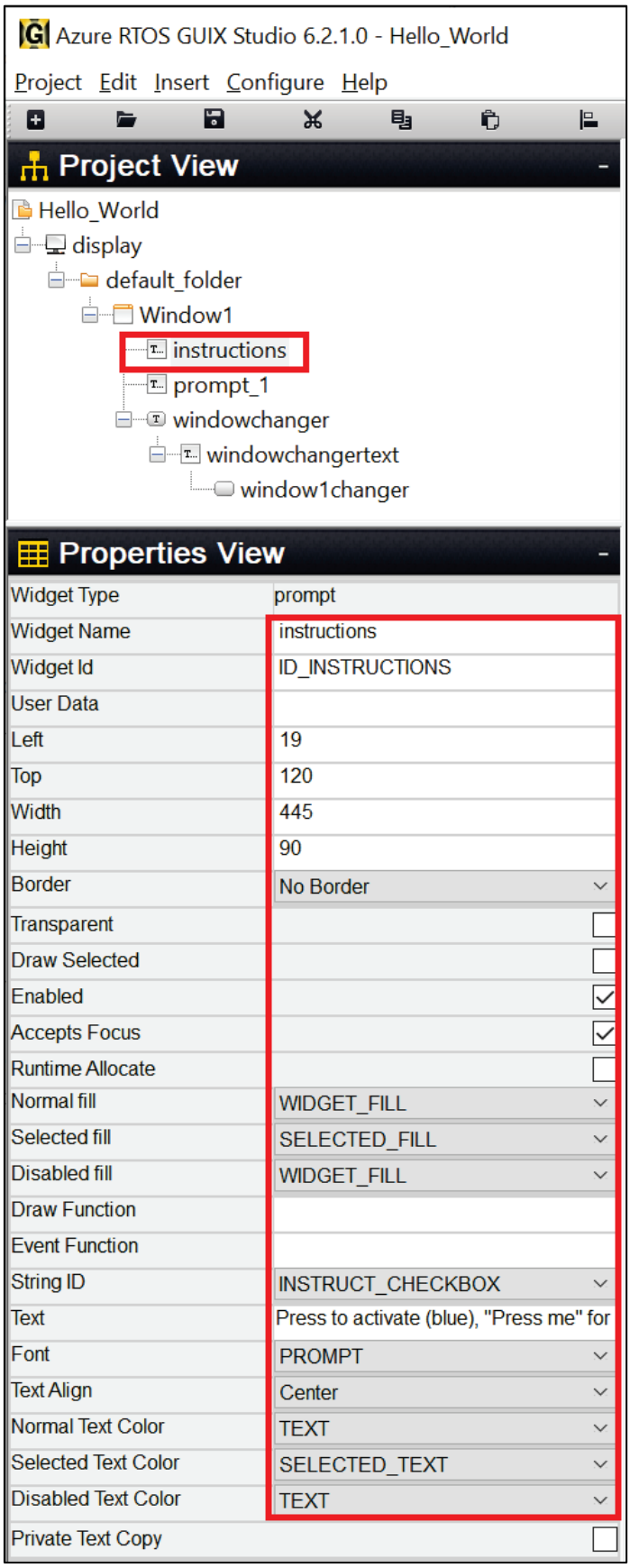


Figure 56. Properties View of Prompt

19. Set the Properties View of Prompt1.

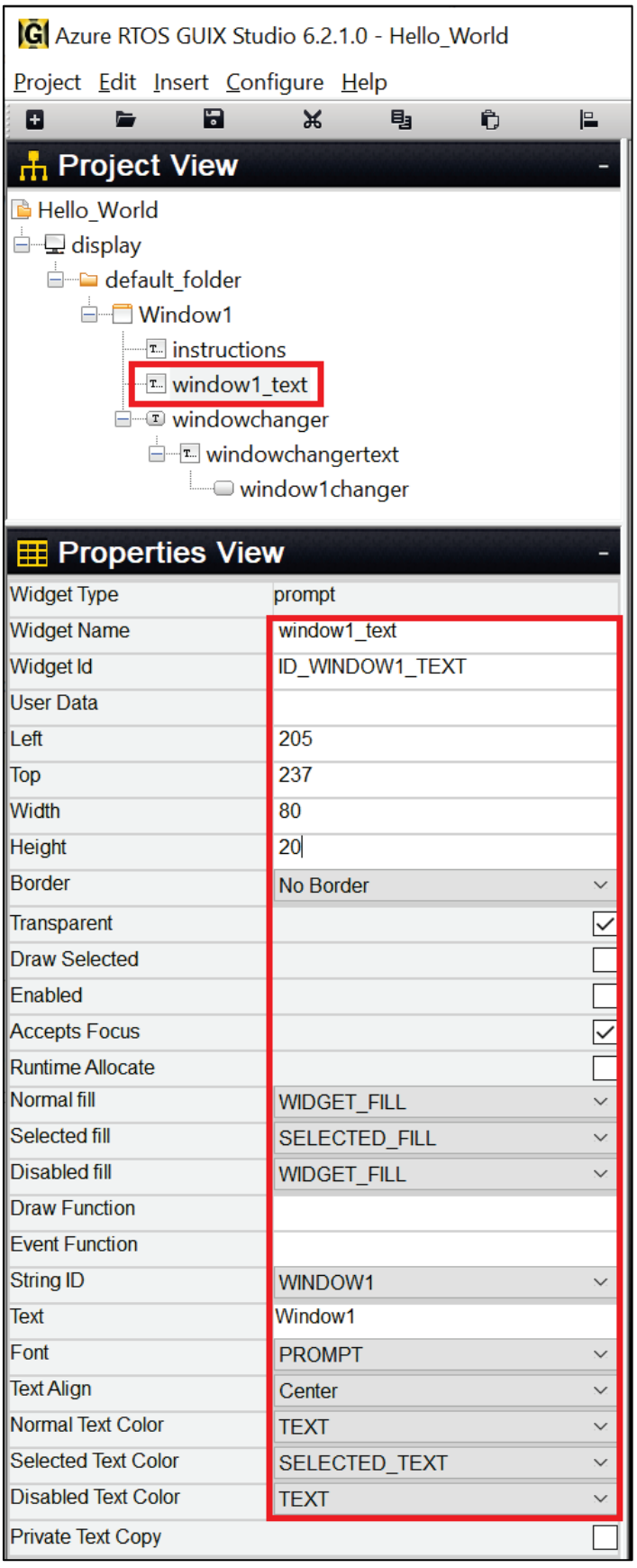


Figure 57. Properties View for Prompt1

20. Insert **button Checkbox**. Right click on **Window1** and follow Figure 58.

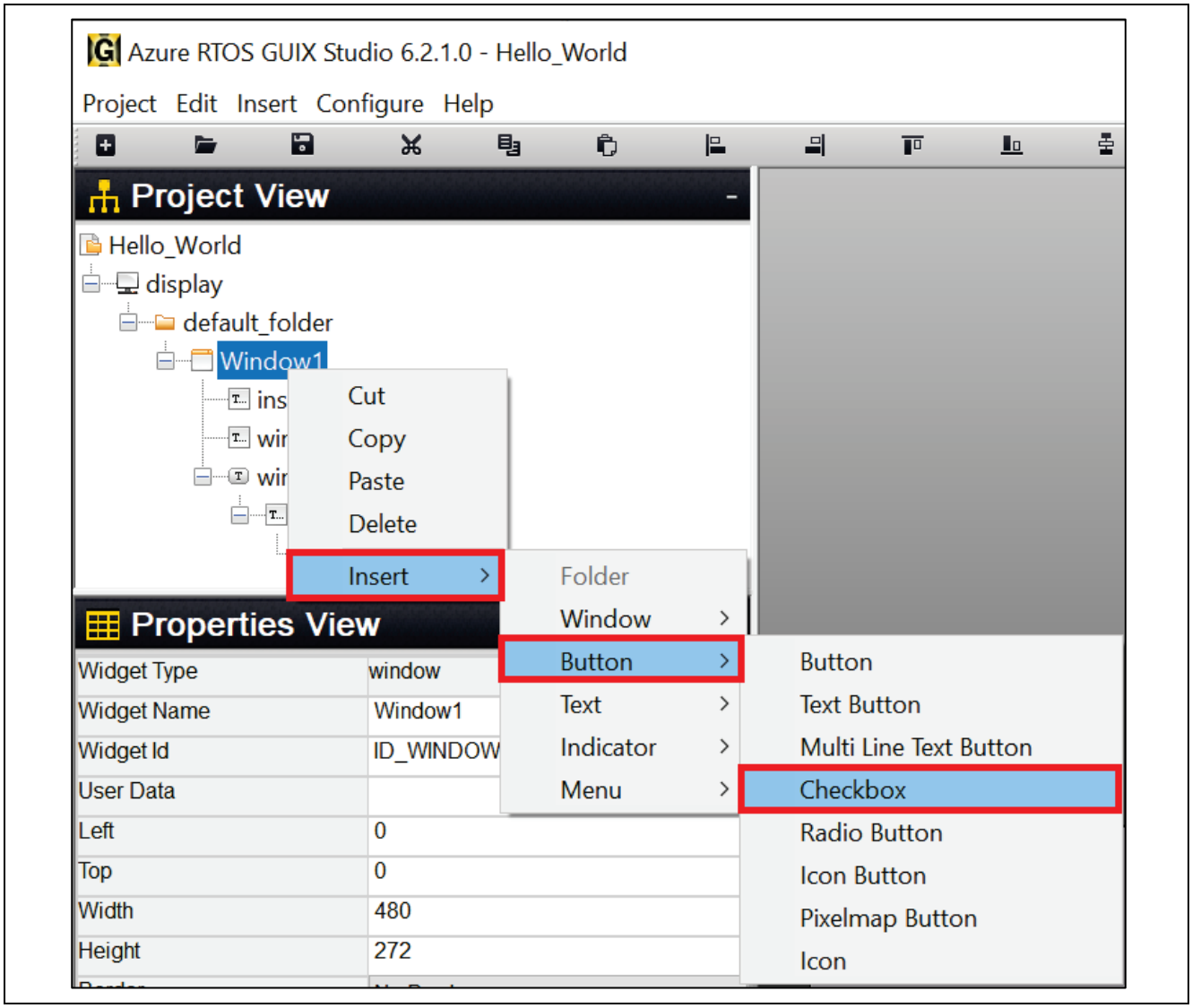


Figure 58. Insert Button Checkbox

21. Setting Properties View of Button Checkbox.

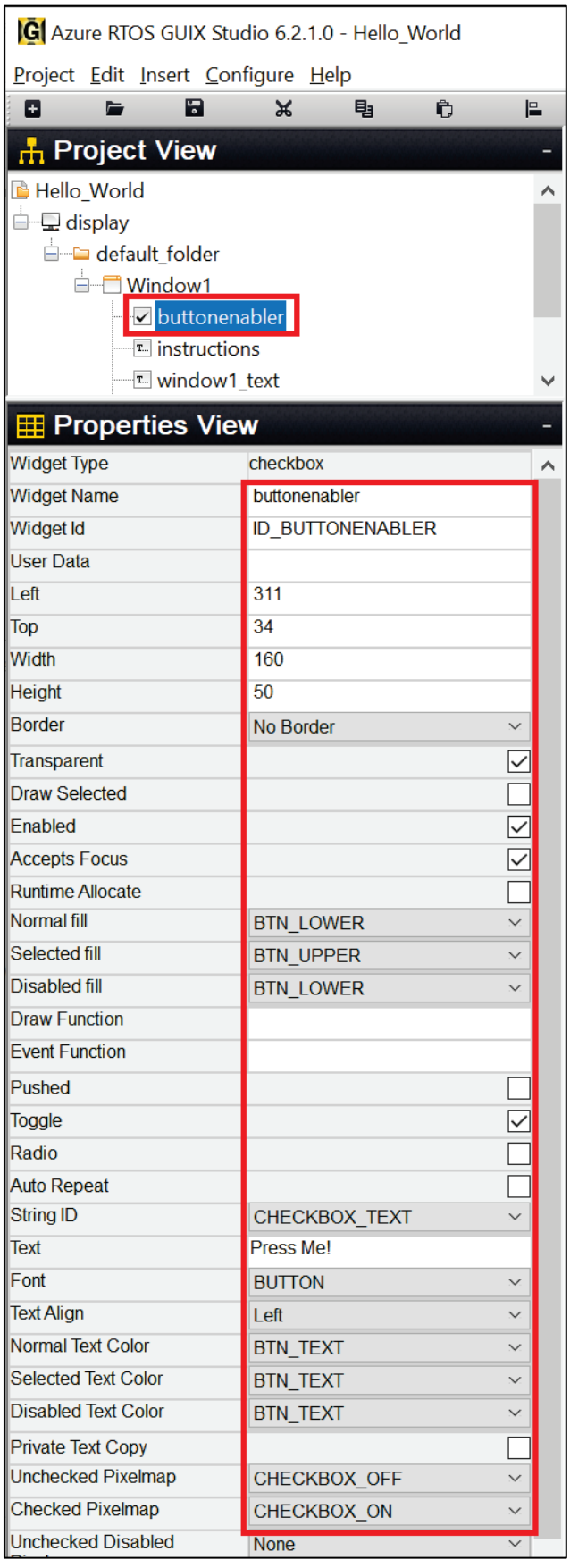


Figure 59. Setting Button Checkbox Properties

22. After you have finished creating **Window1**, it should be like the image below.

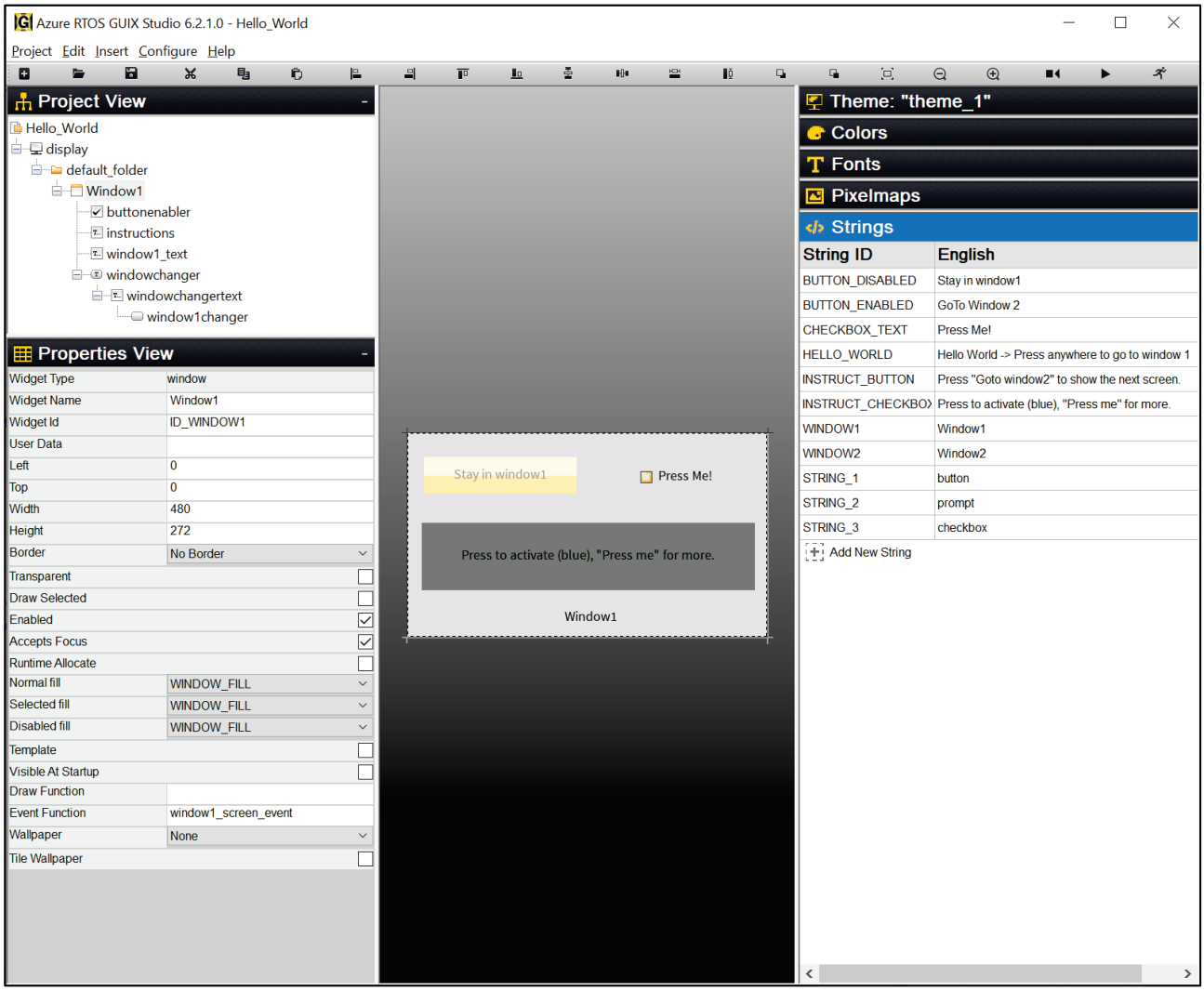


Figure 60. Window1 Created

23. **Insert Window2.** Right click on **default_folder** and **Insert** > **window** > **window** follow Figure 61.

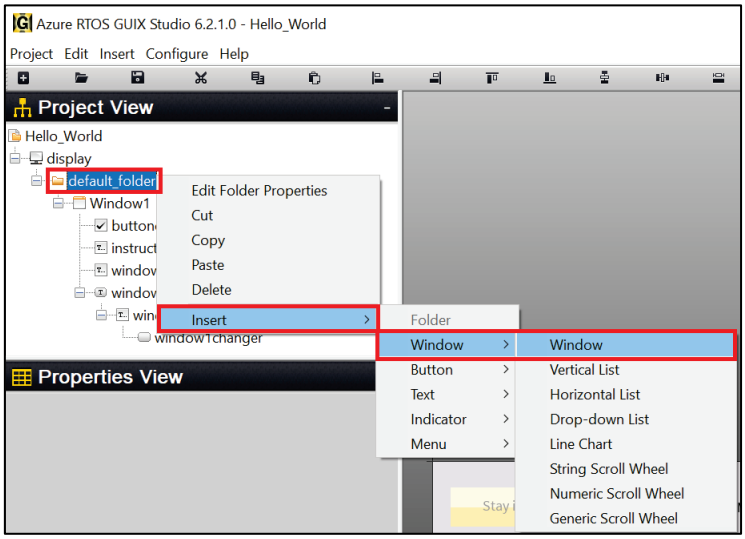


Figure 61. Insert Window2

24. Setting Properties View of Window2.

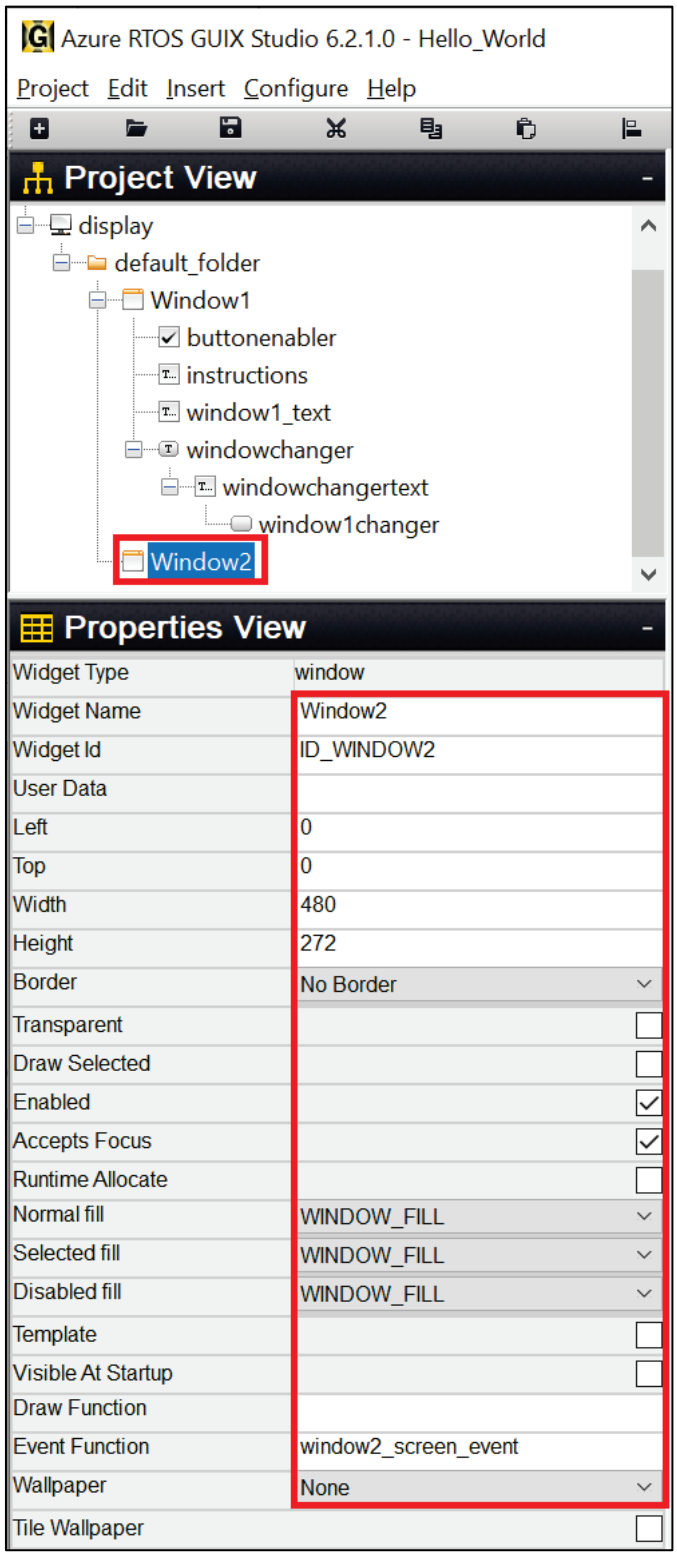


Figure 62. Setting Properties of Window2

25. Insert **Prompt** for Window2. Right click from **Window2**. **Insert** > **Text** > **Prompt** follow Figure 63.

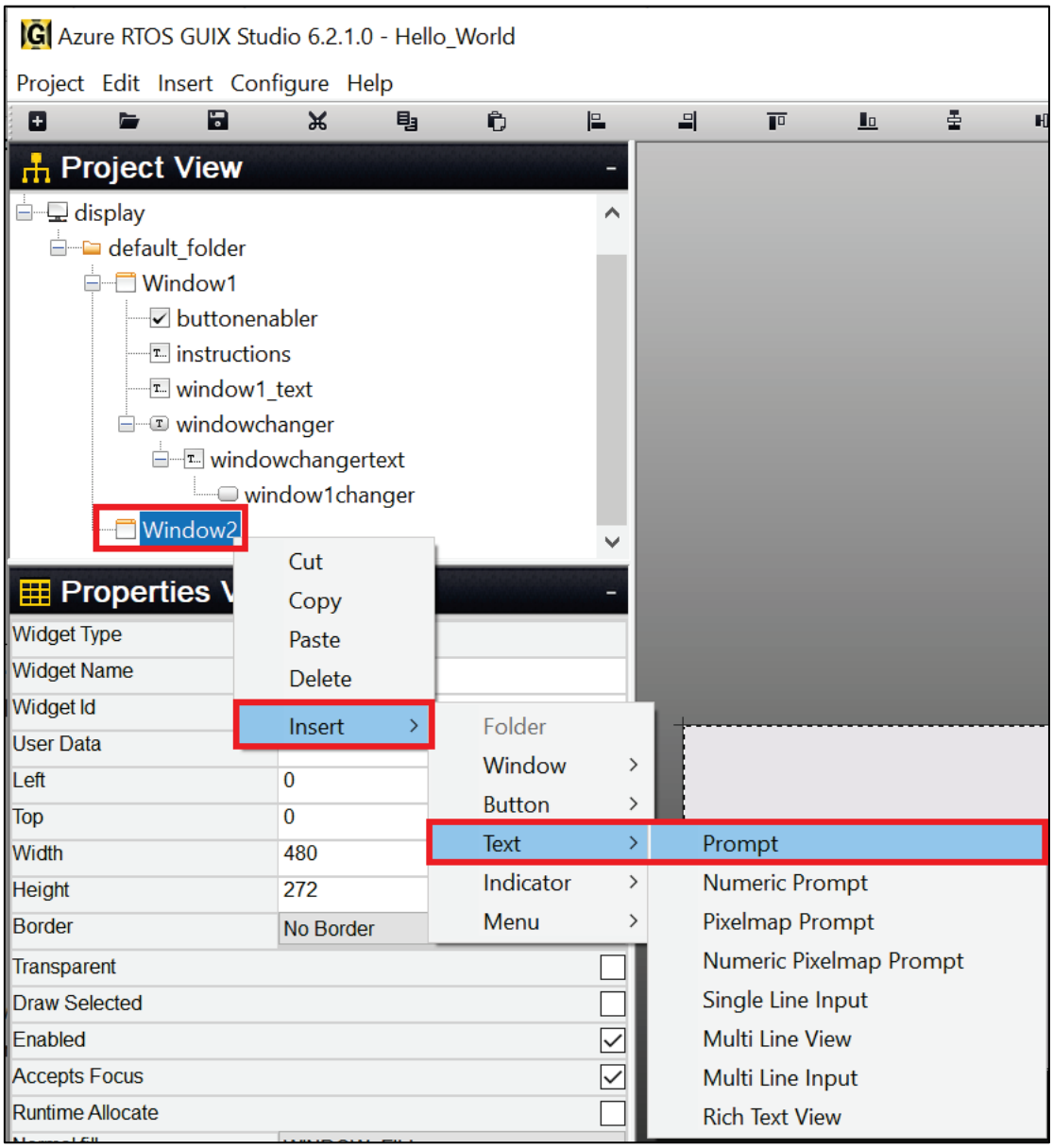


Figure 63. Insert Prompt for Window2

26. Setting Properties View of Prompt.

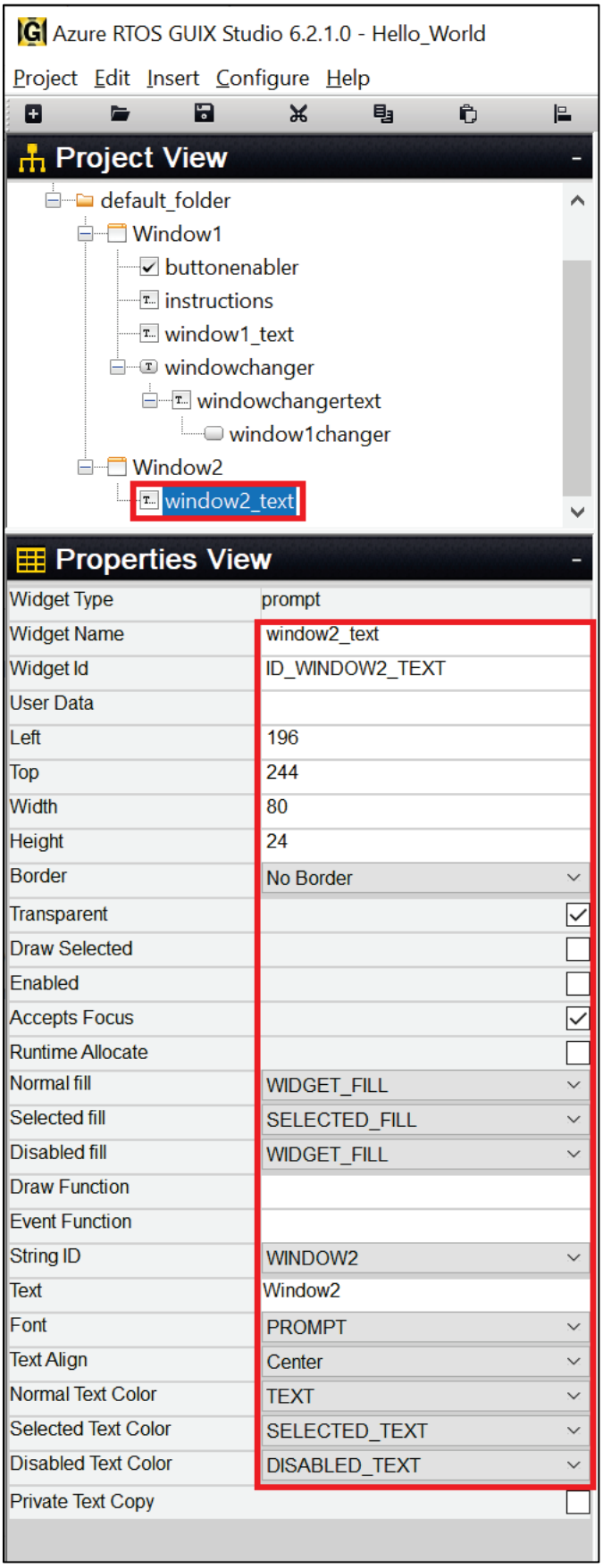


Figure 64. Setting Properties of window2_text

27. Insert **text_button** for window2. Right click from **Window2** > **Insert** > **Button** > **Text_Button** follow Figure 65.

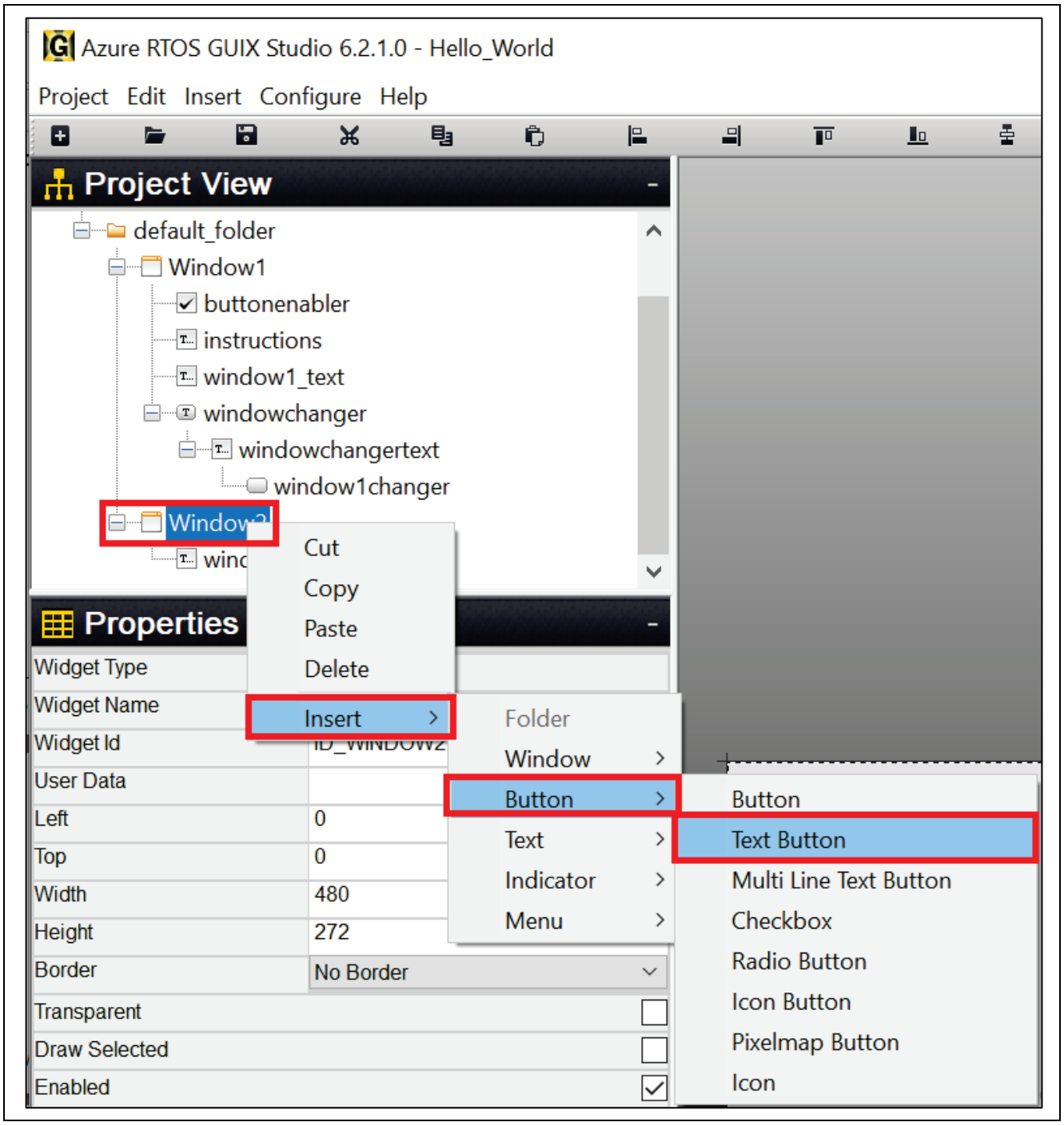


Figure 65. Insert text_button for Window2

28. Setting Properties View of text button.

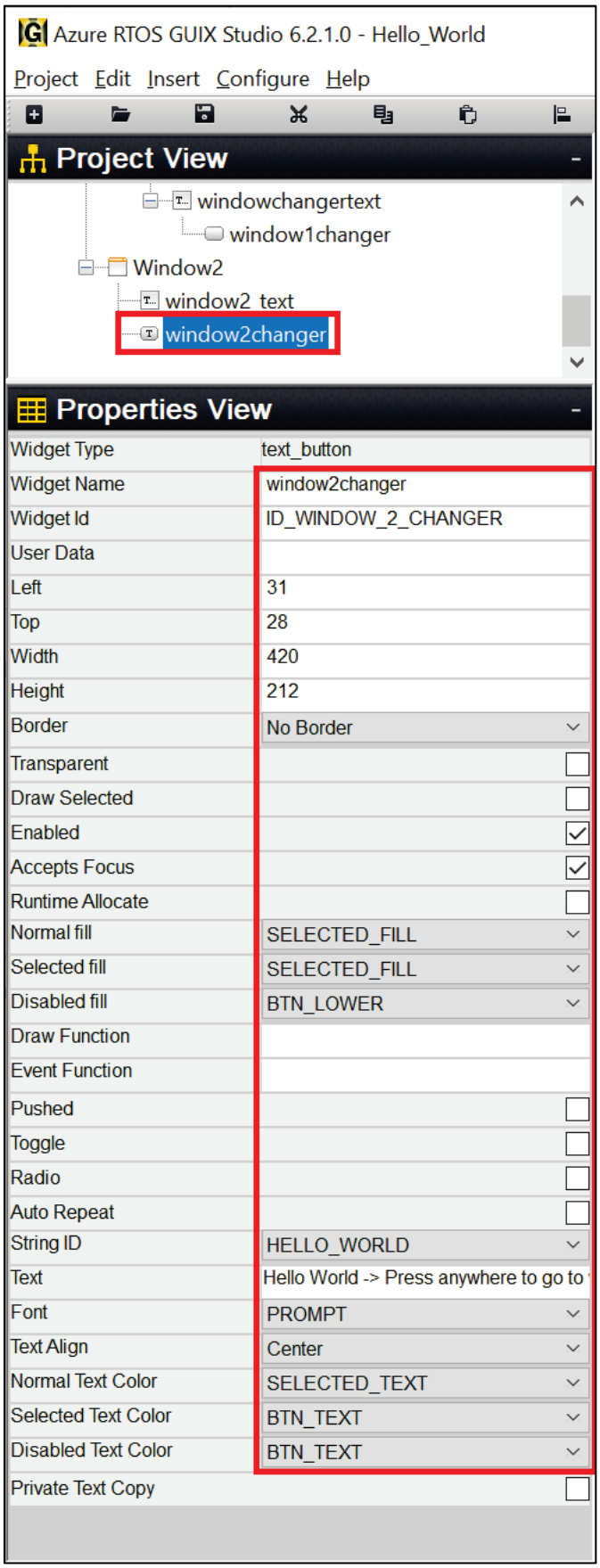


Figure 66. Setting Properties of text_button

29. After Insert and configuration. Window2 looks like **Figure 67**.

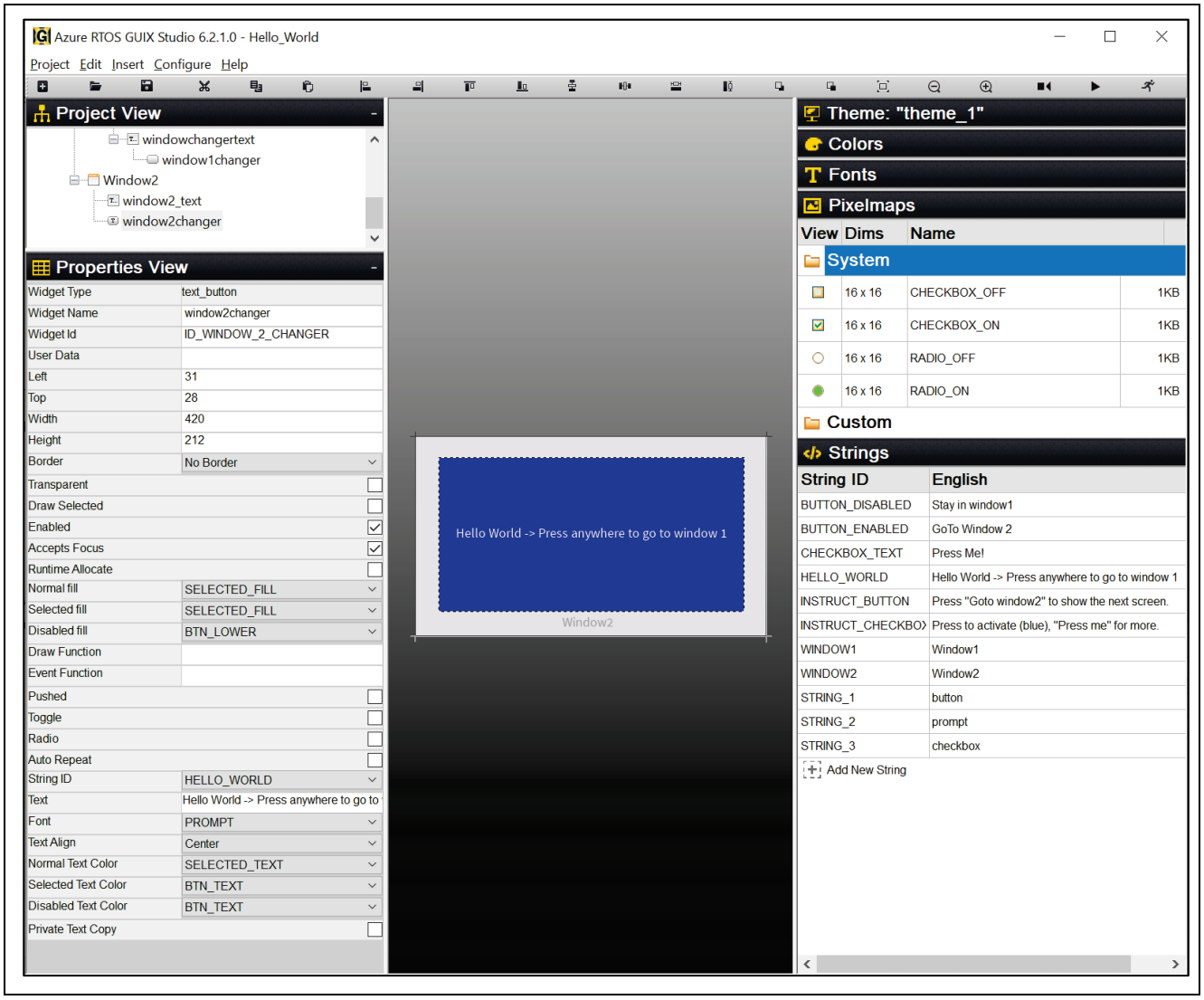


Figure 67. Window2

30. Click on drop-down list **Pixelmaps**, double-click on **CHECKBOX_OF**" and a new window will pop up. Uncheck **Compress Output** then click **Save**. Do the same for **CHECKBOX_ON**.

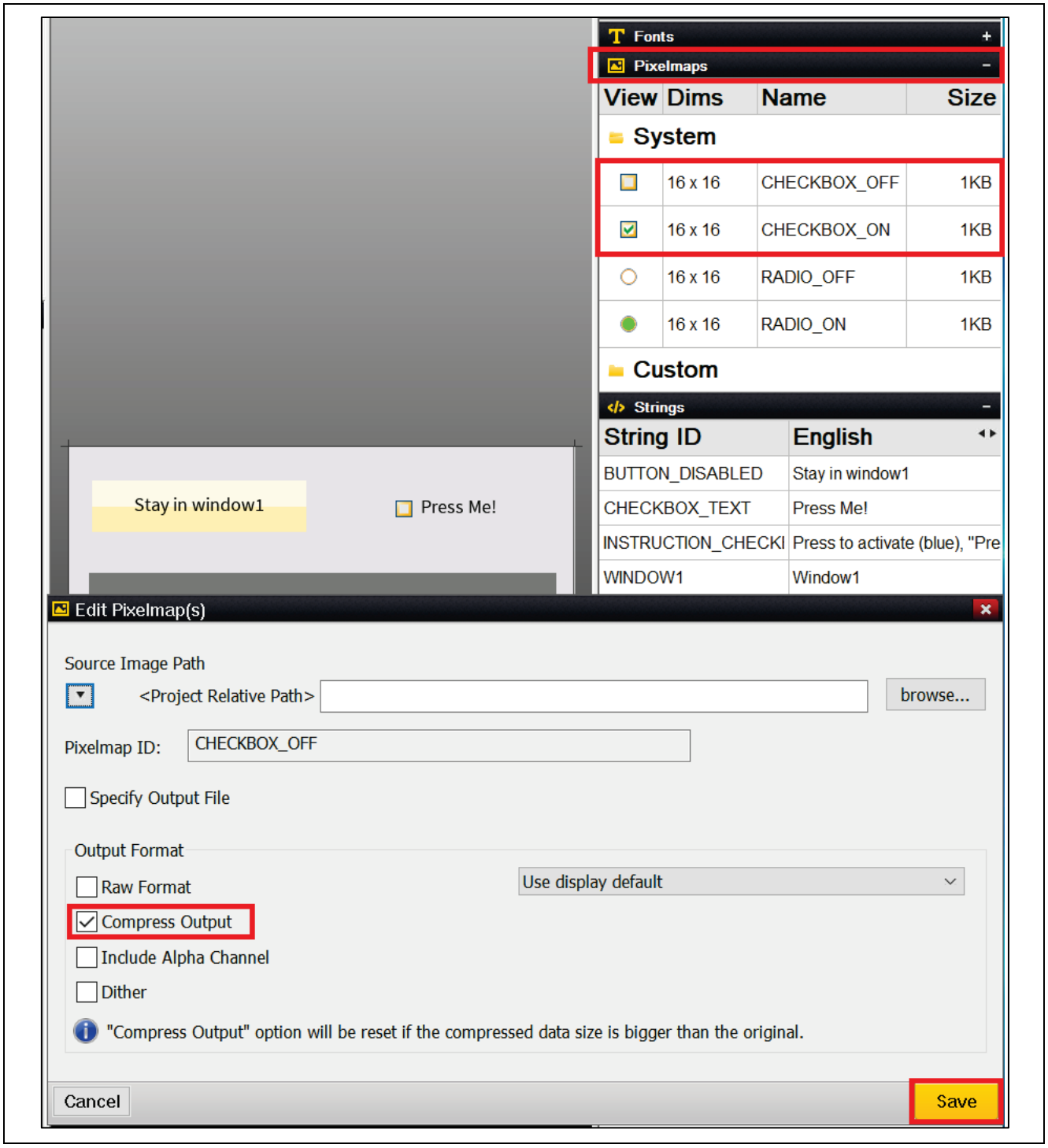


Figure 68. Set Up Pixelmap

31. Now you can click on the **Project** drop down list, **Save Project**, and **Generate All Output Files**. You completed the process of creating and exporting GUIX Hello World into the project.

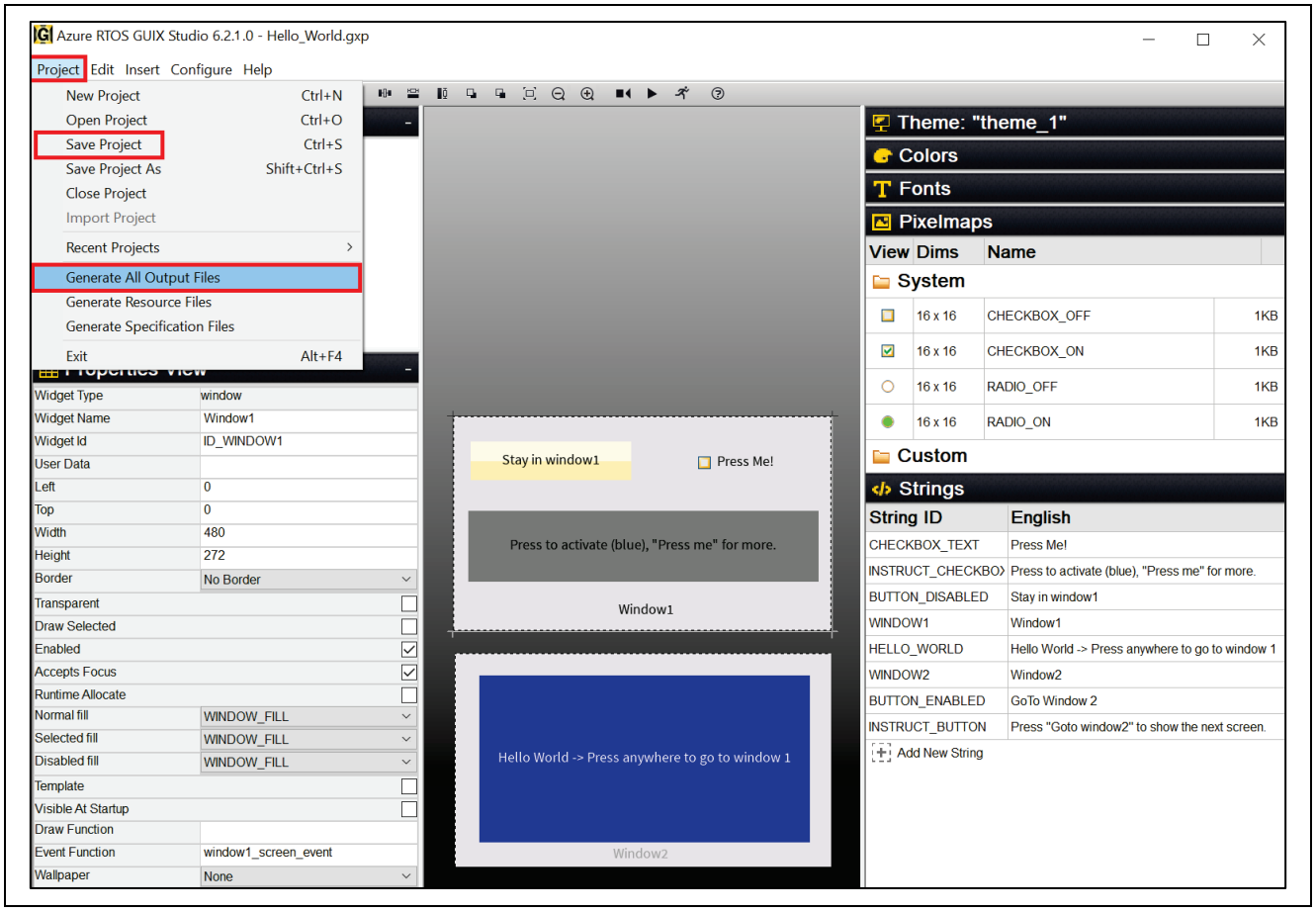


Figure 69. Save and Generate Project

32. Make sure the project is active and click to build the project. It may take a long time to finish building an Azure RTOS/GUIX project on your PC. Project **ra8d1_guix_hello_world** should be built with no errors or warnings.

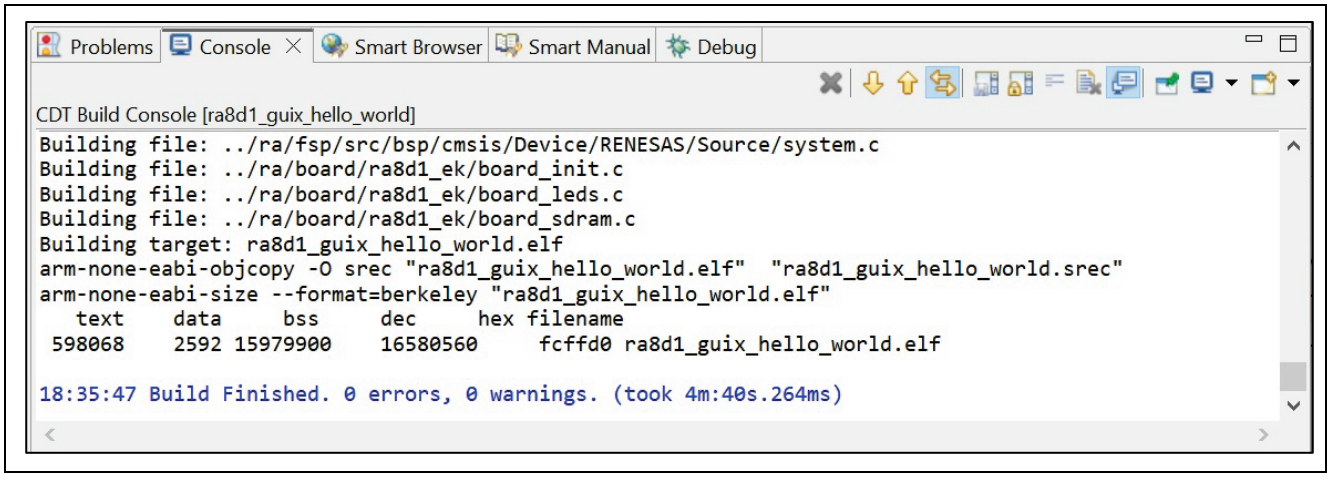


Figure 70. Built the Code

33. Using the Micro USB cable, connect to J10 on EK-RA8D1 board and the other end to your PC. Download and run the "**ra8d1_guix_hello_world**" project.

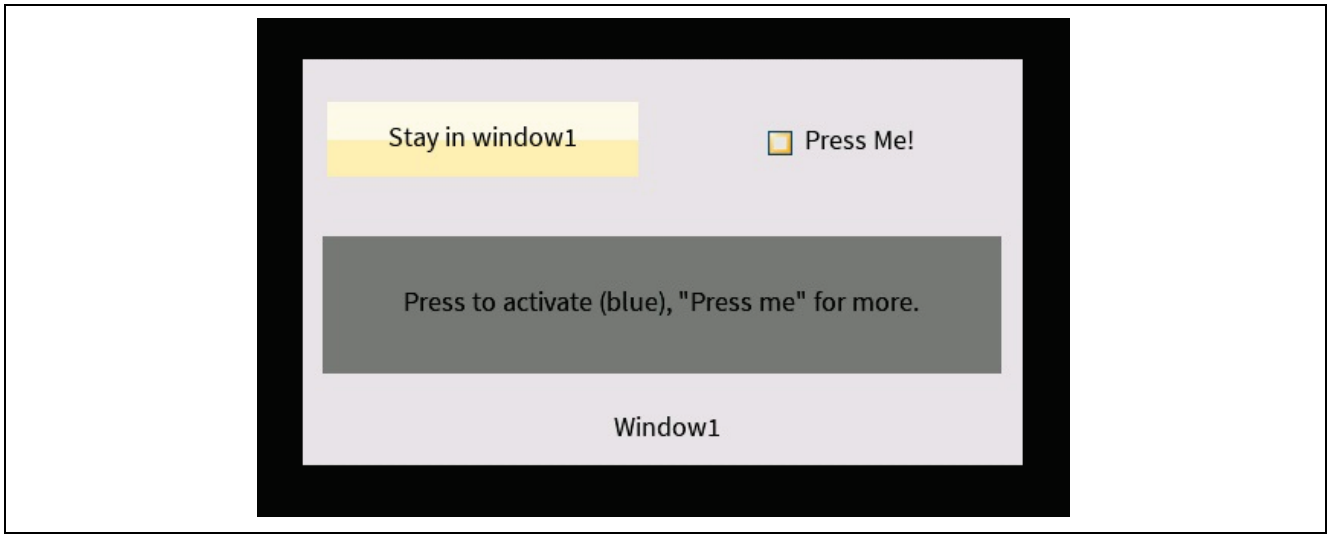


Figure 71. Window1 Display

6. Overview of Fully Functional Project

6.1 Overview

In this section, you will import and run the complete "**ra8d1_guix_hello_world**" project. You can enable or disable the check box function. The text on the button, which is "Stay in window1" or "Go to Window 2", will be updated. Once you press the button, the screen will change from window1 to window2. Follow the text message on the screen, you can change from window2 back to window1. Referred to Figure 71.

6.2 Procedural Steps

1. You can try the provided project "**ra8d1_guix_hello_world**" for the full function application. Use the **Rename & Import Existing C/C++ Project into Workspace** feature of the **Import** menu to do so since you already had a project with the same in the workspace.

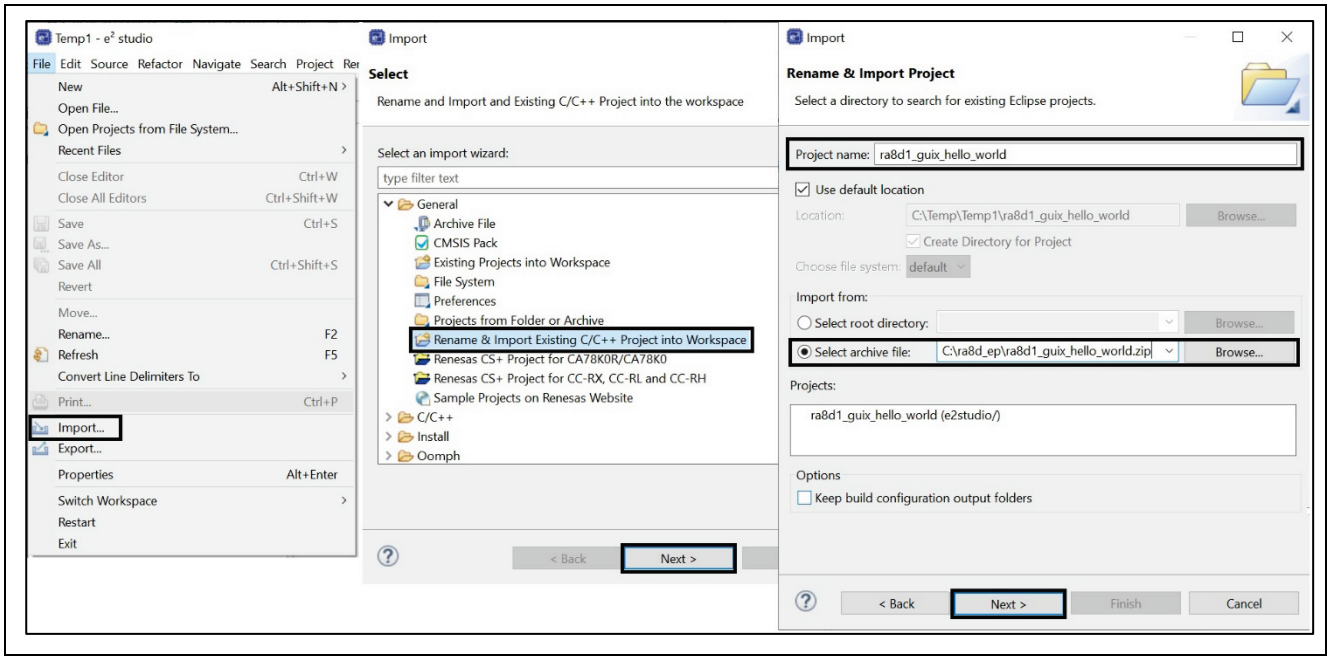


Figure 72. Import Existing Project

7. Website and Support

Visit the following URLs to learn about key elements of the RA family, download components and related documentation, and get support:

RA Product Information	renesas.com/ra
RA Product Support Forum	renesas.com/ra/forum
RA Flexible Software Package	renesas.com/FSP
Renesas Support	renesas.com/support

Revision History

Rev.	Date	Description	
		Page	Summary
1.00	Dec.13.23	—	Initial release

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. Precaution against Electrostatic Discharge (ESD)

   A strong electrical field, when exposed to a CMOS device, can cause destruction of the gate oxide and ultimately degrade the device operation. Steps must be taken to stop the generation of static electricity as much as possible, and quickly dissipate it when it occurs. Environmental control must be adequate. When it is dry, a humidifier should be used. This is recommended to avoid using insulators that can easily build up static electricity. Semiconductor devices must be stored and transported in an anti-static container, static shielding bag or conductive material. All test and measurement tools including work benches and floors must be grounded. The operator must also be grounded using a wrist strap. Semiconductor devices must not be touched with bare hands. Similar precautions must be taken for printed circuit boards with mounted semiconductor devices.
2. Processing at power-on

   The state of the product is undefined at the time 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 time 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 time 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 time when power is supplied until the power reaches the level at which resetting is specified.
3. Input of signal during power-off state

   Do not input signals or an I/O pull-up power supply while the device is powered off. The current injection that results from input of such a signal or I/O pull-up power supply may cause malfunction and the abnormal current that passes in the device at this time may cause degradation of internal elements. Follow the guideline for input signal during power-off state as described in your product documentation.
4. 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 the 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.
5. Clock signals

   After applying a reset, only release the reset line after the operating clock signal becomes stable. When switching the clock signal during program execution, wait until the target clock signal is 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. Additionally, 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.
6. Voltage application waveform at input pin

   Waveform distortion due to input noise or a reflected wave may cause malfunction. If the input of the CMOS device stays in the area between V_{IL} (Max.) and V_{IH} (Min.) due to noise, for example, the device may malfunction. Take care to prevent chattering noise from entering the device when the input level is fixed, and also in the transition period when the input level passes through the area between V_{IL} (Max.) and V_{IH} (Min.).
7. Prohibition of access to reserved addresses

   Access to reserved addresses is prohibited. The reserved addresses are provided for possible future expansion of functions. Do not access these addresses as the correct operation of the LSI is not guaranteed.
8. Differences between products

   Before changing from one product to another, for example to a product with a different part number, confirm that the change will not lead to problems. The characteristics of a microprocessing unit or microcontroller unit products in the same group but having a different part number might differ in terms of 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 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 be responsible for determining what licenses are required from any third parties, and obtaining such licenses for the lawful import, export, manufacture, sales, utilization, distribution or other disposal of any products incorporating Renesas Electronics products, if required.
5. 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.
6. 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.
7. No semiconductor product is absolutely secure. Notwithstanding any security measures or features that may be implemented in Renesas Electronics hardware or software products, Renesas Electronics shall have absolutely no liability arising out of any vulnerability or security breach, including but not limited to any unauthorized access to or use of a Renesas Electronics product or a system that uses a Renesas Electronics product. RENESAS ELECTRONICS DOES NOT WARRANT OR GUARANTEE THAT RENESAS ELECTRONICS PRODUCTS, OR ANY SYSTEMS CREATED USING RENESAS ELECTRONICS PRODUCTS WILL BE INVULNERABLE OR FREE FROM CORRUPTION, ATTACK, VIRUSES, INTERFERENCE, HACKING, DATA LOSS OR THEFT, OR OTHER SECURITY INTRUSION (“Vulnerability Issues”). RENESAS ELECTRONICS DISCLAIMS ANY AND ALL RESPONSIBILITY OR LIABILITY ARISING FROM OR RELATED TO ANY VULNERABILITY ISSUES. FURTHERMORE, TO THE EXTENT PERMITTED BY APPLICABLE LAW, RENESAS ELECTRONICS DISCLAIMS ANY AND ALL WARRANTIES, EXPRESS OR IMPLIED, WITH RESPECT TO THIS DOCUMENT AND ANY RELATED OR ACCOMPANYING SOFTWARE OR HARDWARE, INCLUDING BUT NOT LIMITED TO THE IMPLIED WARRANTIES OF MERCHANTABILITY, OR FITNESS FOR A PARTICULAR PURPOSE.
8. 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.
9. 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.
10. 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.
11. 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.
12. 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.
13. This document shall not be reprinted, reproduced or duplicated in any form, in whole or in part, without prior written consent of Renesas Electronics.
14. 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.5.0-1 October 2020)

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/.