guiliani.de

# Using Guiliani RA SDK
# With M13-RA6M3-EK within e2Studio

| | |
|---|---|
| Product: | SDK Guiliani for Renesas RA |
| Release version: | 2.6 |
| Release date: | December 15, 2023 |

**Table of Contents**

## List of Figures



## List of Tables

# 1 Introduction

The “GuilianiDemo” presents some of the capabilities of the Guiliani HMI framework running on the M13-RA6M3-EK. It uses the TES D/AVE accelerated rendering-engine (aka “2D Drawing Engine”). Additionally the efficient way of working for GUI application development by using the Guiliani Streaming Editor (GSE) is introduced.

The SDK for M13-RA6M3-EK contains an e2Studio project, which can be used for editing and debugging the Guiliani demo. e2Studio is an eclipse-based Integrated Development Environment (IDE). This document describes the different projects, their directory structure and the build configurations included in the e2Studio project workspace of the demo.

This guide does not explain how to create an e2Studio project and configure the settings. It rather explains an e2Studio workspace, which is already created and included in the SDK so that the user can quickly test the Guiliani demo and do the changes according to his requirements.

For an introduction to the Renesas RA MCU Family and related topics, we recommend reading the “Official Renesas RA Family Beginner’s Guide” by Richard Oed. It can be found on www.renesas.com.

# 2 Assumed Knowledge

- Basic to advanced knowledge of C and C++
- General understanding and hands-on experience of e2Studio or eclipse (If you are not familiar with any of these tools, we recommend you to read “User’s Manual: Getting Started Guide” of e2Studio, available on Renesas website)

# 3 Prerequisites

- Installed Guiliani SDK for M13-RA6M3-EK
- Installed e2Studio 2023-10 with FSP 4.2.0
- GNU MCU Eclipse Windows Build Tools (2.9-20170629-1013)
- **NOTE: you will have to add additional files into your e2studio-installation for flashing the external QSPI-Flash (details in next paragraph)**

# 4 System Preparation

- Connect the M13-RA6M3-EK target (CN5) via an Micro-USB cable to a Windows PC to power it up. This connector also serves as the direct JLink-connection


**Fig. 1 Connect via USB**

## 4.1 Modify e2studio-installation

**Note: Errors during this procedure may damage your e2studio-installation, leaving it unusable.**

In order to flash data to the external QSPI-Flash of the board you will need to add some files into your e2studio-installation-folder. To find out which folder that is do the following:

- Start your e2studio-installation
- Open the About-dialog with Menu “Help → About e2studio”
- Click on “Installation Details”
- Navigate to the tab “Support Folders”
- Open the link “e2studio support area”

This will open a folder within your e2studio-installation.

- In this folder open the subfolder “DebugComp/RA/ARM”
- Open the file “ra6m3.<FSP_VERSION>.xml” in a text-editor
- Find the tag “SeggerInformation”
- Extract the additional files (i.e. “JLinkDevices.xml” and “Renesas_R7FA6M3AH_Port5_QSPI.elf”) into the folder specified by the tag “Path” (e.g. Segger_v6.86.4”)

## 4.2 Migration steps for FSP

If you are using a different version of the FSP than FSP 4.2.0 there might be some changes necessary to compile the projects without errors. Please refer to https://github.com/renesas/fsp/releases on how to do the necessary changes.

# 5 Run GuilianiDemo

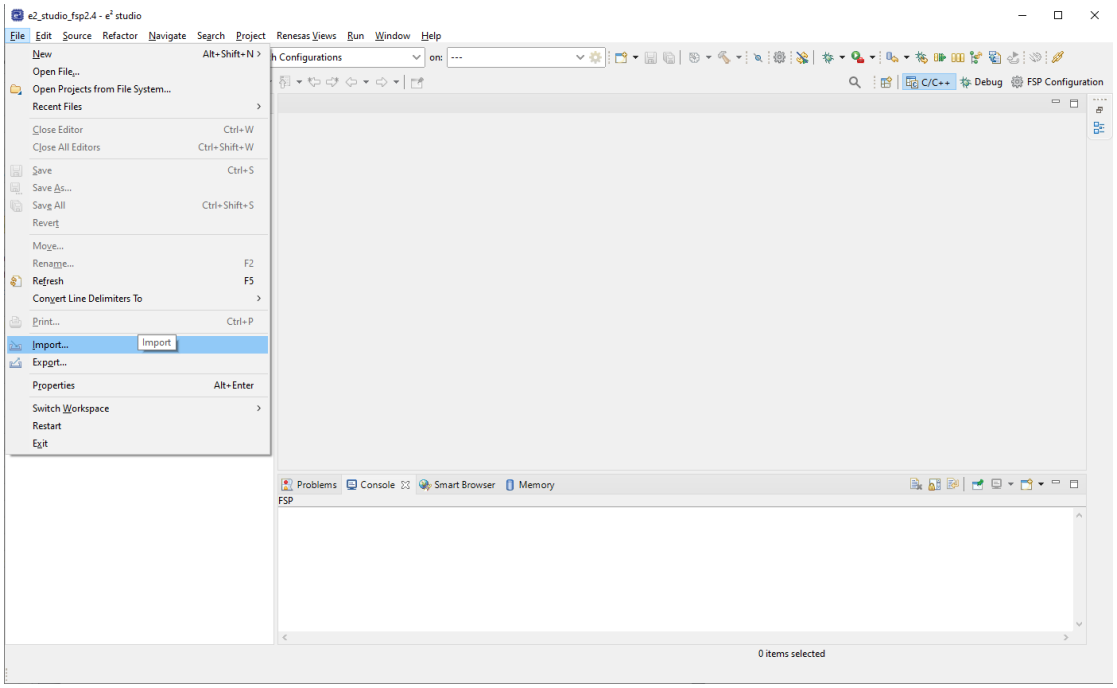
## 5.1 Flash application and resources

To get ready running GuilianiDemo on your target-board the following steps are necessary:

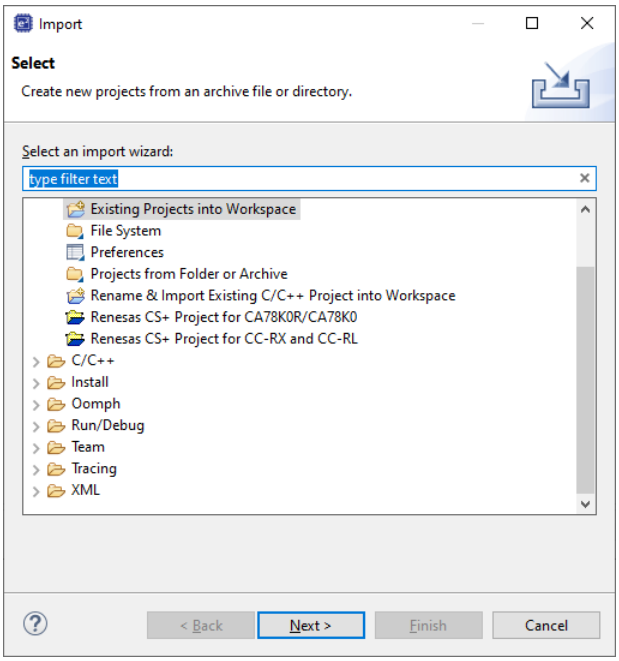
- Open e2studio
- Do the modification from paragraph 4.1 if not yet done

### 5.1.1 Import and compile the application

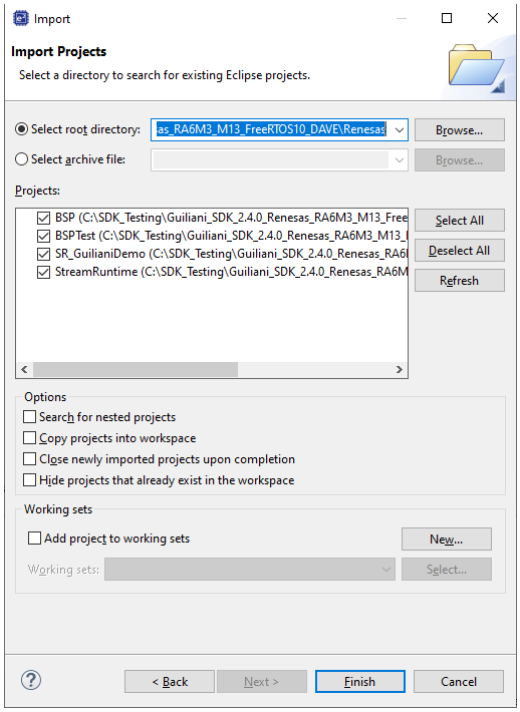
Open the import dialog with “File->Import”



Select “Existing Projects into Workspace” and click Next

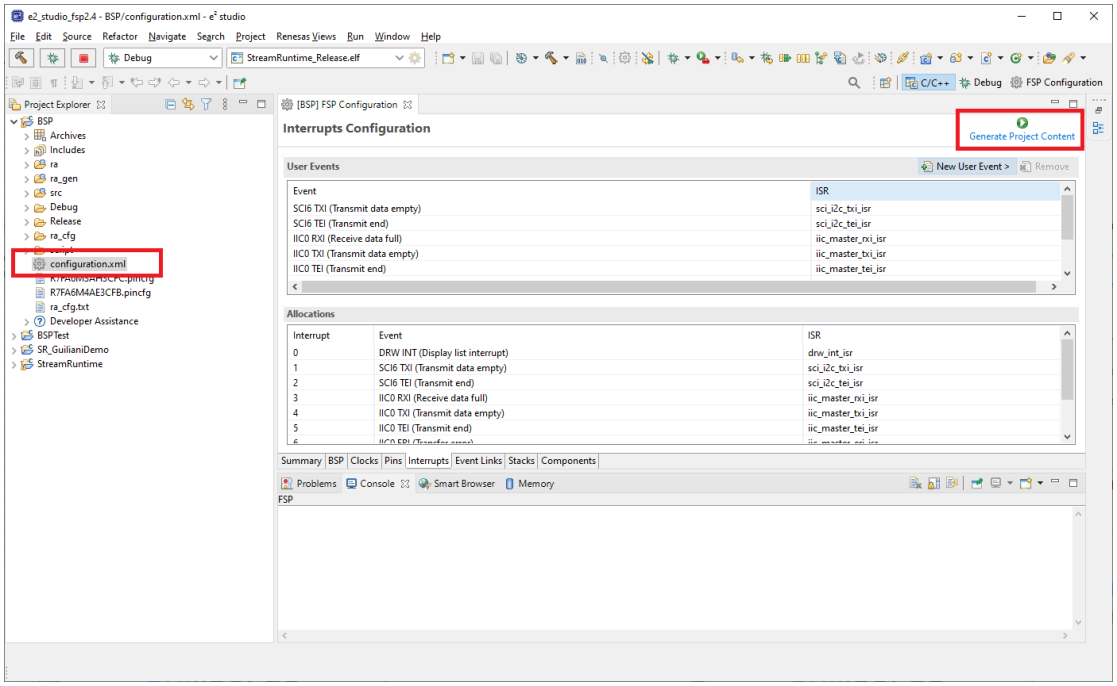



Use “Browse” to select the Renesas-subfolder of the SDK and click on “Finish”



The projects are now imported into the workspace

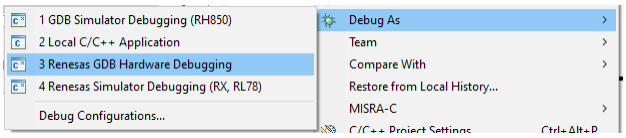
Double-click on the file “configuration.xml” in the BSP-project to open it in the FSP-Configurator and click “Generate Project Contents” to create necessary files for the BSP.
**Note: loading the xml-file may take some time. Please be patient, otherwise errors might occur which might damage the project.**



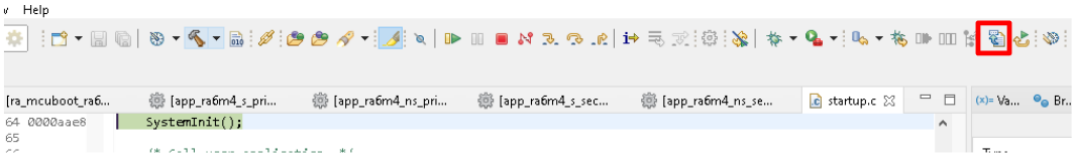

Build the project BSP and the after that the application.

### 5.1.2 Debug application and download resources

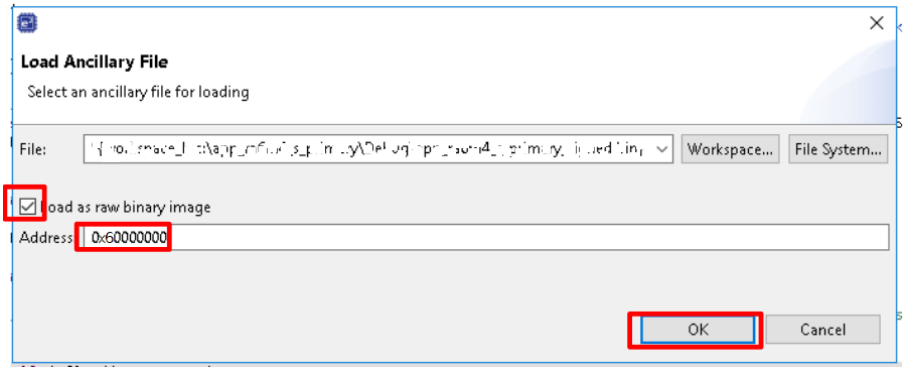
Start debugging-session for the application and pause it at the beginning to load the resources.



Click on “Load Ancillary File” from the debug-toolbar to open the dialog.




Select the file “Export/RA6M3/Resources.dat”, set the check for RAW-file and enter “0x60000000” (seven zeros)




### 5.2 Demo description

This demo has a main screen in which you can choose between several options. In each of them you can test the functionality implemented using the Guiliani API.

## 6 Edit example project on the computer

Please read chapter 5 in the document “GSE Getting started.pdf” to learn about how to edit the GuilianiDemo (see /documentation folder).

## 7 Load edited example project onto the board

- In the previous step we were able to edit a GSE project and see the modifications on a running simulation on the computer. Now we will see those changes on the board.
- Connect the board to the computer via Micro-USB-cable and to the power supply.
- With the GSE project open, export the resources (i.e. the GuilianiDemo application including all pictures etc.) by clicking Resources ➔ Export (Fig. 5).

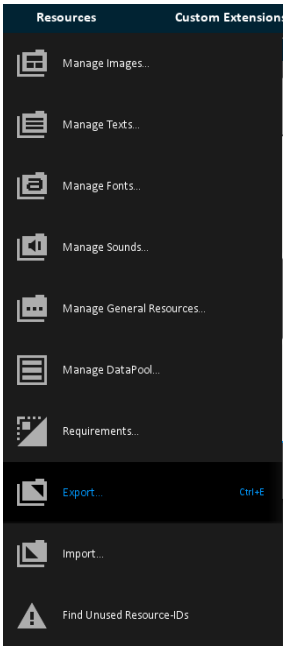



**Fig. 2 Export Option**

- In the Export window (Fig. 6) start-dialog and all other settings will have been selected according to the export-profile. Please select the correct export-folder by clicking on the "…"-button in the first line and navigating to the folder "<SDK-Root>/Export/RA6M3"

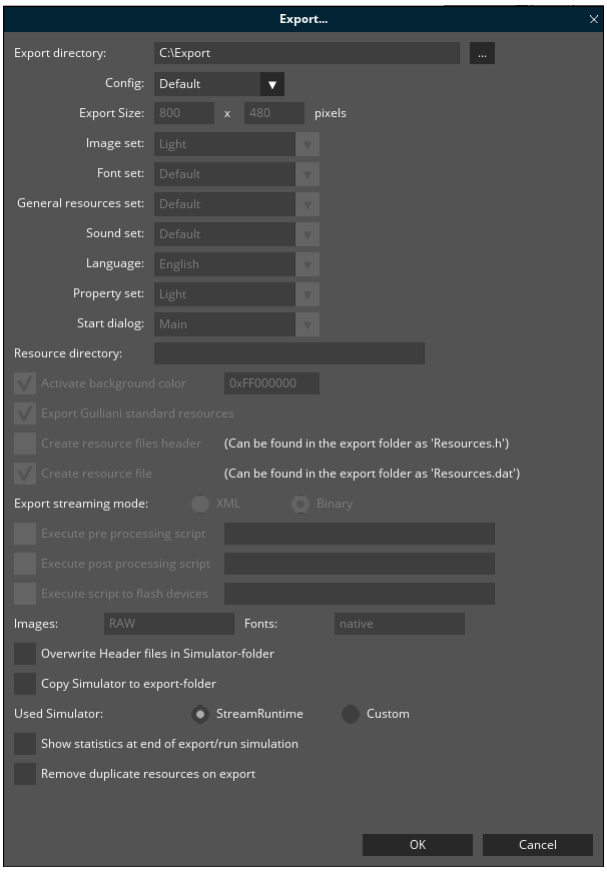



**Fig. 3 Exporting Configuration**

- Click on *OK*
- After the export is done, go to e2studio and start a debug session for the GuilianiDemo
- Use "Load Ancillary File" to load up the file "Resources.dat" to address 0x60000000
- Now you can start playing with the GuilianiDemo demo on the board. Navigate to "SCRATCHPAD" and you will see your first own Guiliani HMI application running on the target board.

# 8 Compiling your own GSE and StreamRuntime

If you like to extend your GSE project with your own functionality, you have to re-compile GSE and StreamRuntime. For this you have to take CMake as the build environment.

# 9 e²Studio Workspace

e2Studio projects are available in the SDK within the Renesas folder. Launch e2Studio IDE and import the projects into your workspace.

The following four projects are available (Fig. 2):

- BSP: Pre-configured Board Support Package (BSP) for RA6M3
- BSP_Test: A test project to quickly test BSP without Guiliani
- SR_GuilianiDemo: The Guiliani demo
- StreamRuntime: The StreamRuntime demo

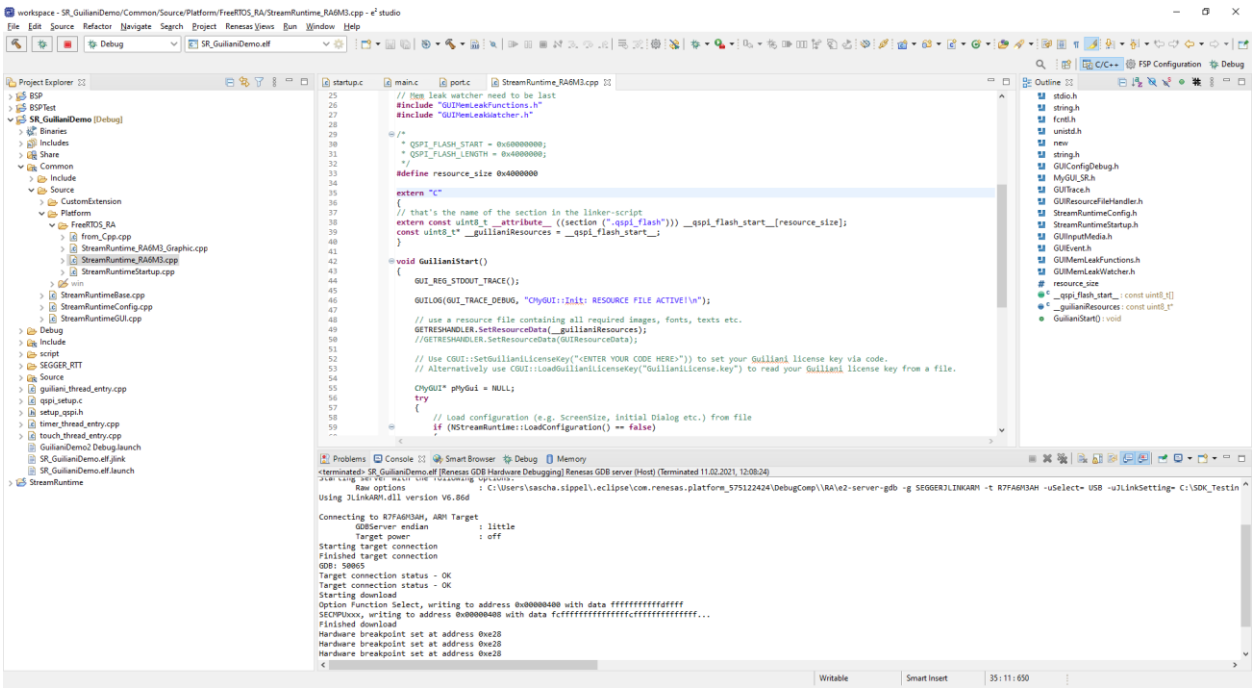


**Fig. 3 e2Studio Workspace of SDK Project**

## 9.1 Board Support Package (BSP)

This SDK includes a pre-configured BSP-project. The BSP contains initialization code for clocks, RAM, caches and peripherals which are specific to the boards. It also includes driver files and a FreeRTOS port for the evaluation board kit.

**Note: To compile the BSP you first have to generate the project contents based on the used board. To do this, open the file "configuration.xml" and click "Generate Project Contents". Now the BSP-project can be compiled.**

### 9.1.1 Directory structure

| Directory | Description |
|---|---|
| src | Minimal sources for the BSP-project |

**Table 1 Directory Structure of BSP Project**

### 9.1.2 Build configurations

- Debug: It builds BSP-project in debug mode. When the project is built, it creates a library libBSP.a in a subfolder Debug, which can be used by SR_GuilianiDemo, StreamRuntime and BSP_Test projects.
- Release: It builds BSP-project in release mode. When the project is built, it creates a library libBSP.a in a subfolder Release, which can be used by SR_GuilianiDemo, StreamRuntime and BSP_Test projects.

## 9.2 BSP_Test

This project allows a user to quickly test the BSP without the need of the Guiliani application. The test program can be flashed on the board and can be debugged. It is a simple blinking application.

### 9.2.1 Directory structure

| Directory | Description |
|---|---|
| src | Application source code |

**Table 2 Directory Structure of BSP_Test Project**

### 9.2.2 Build configurations

- Debug: BSP_Test program from flash in Debug mode.
- Release: BSP_Test program from flash in Release mode.

## 9.3 SR_GuilianiDemo

This project contains the files required for porting Guiliani on the Renesas board.

### 9.3.1 Directory structure

| Directory | Description |
|---|---|
| Common | Common files over different Guiliani applications |
| GuilianiDemo | Contains GSE projects with different resolutions |
| Include | Project specific includes |
| Source | Project specific sources |

**Table 3 Directory Structure of <SDK>\SR_GuilianiDemo**

| File | Description |
|---|---|
| Platform/*/StreamRuntime*.[c\|cpp\|h] | Program entry points (main function) for different platforms |
| [Include\|Source]/Platform/FreeRTOS_RA/StreamRuntimeStartup_FreeRTOS.[cpp\|h] | Target specific initialization of wrappers and configurations |
| [Include\|Source]/Platform/win/pc/StreamRuntimeStartup_FreeRTOS.h" | Windows specific initialization of wrappers and configurations |
| [Include\|Source]/StreamRuntimeConfig.[h\|cpp] | Loads project configuration |
| [Include\|Source]/StreamRuntimeGUI.[h\|cpp] | Loads GUI |

**Table 4 Files in <SDK>\SR_GuilianiDemo\Common Directory**

| File | Description |
|---|---|
| CustomExtension | Custom extensions. |
| GUIConfig/User*Resource.h | Resource IDs generated by GSE |
| GUIConfigCustom/* | Custom IDs for use in Guiliani application. |
| Demo_*.[cpp\|h] | Specific code for the different demo parts |
| MyGUI_SR.[cpp\|h] | GUI entry point |

**Table 5 Files in <SDK>\SR_GuilianiDemo\Include and <SDK>\SR_GuilianiDemo\Source Directory**

| File | Description |
|---|---|
| GUIConfig.cpp | This file contains constants which hold the count of global properties, image resources, font resources, text resources, etc. |

**Table 6 Files in <SDK>\Guiliani\Share Directory**

### 9.3.2 Build configurations

There are two configurations available for SR_GuilianiDemo project.

1. Debug: Debug configuration. The demo application runs from internal flash. Choose this configuration to debug the application.
2. Release: Release configuration. The application runs from internal flash. Choose this configuration to test the performance.

## 9.4 StreamRuntime

This project contains the files required for porting Guiliani on the Renesas board.

### 9.4.1 Directory structure

| Directory | Description |
|---|---|
| Common | Common files over different Guiliani applications |
| GuilianiDemo | Contains GSE projects with different resolutions |
| Include | Project specific includes |
| Source | Project specific sources |

**Table 7 Directory Structure of <SDK>\StreamRuntime**

| File | Description |
|---|---|
| Platform/*/StreamRuntime*.[cpp\|h] | Program entry points (main function) for different platforms |
| [Include\|Source]/Platform/FreeRTOS_RA/StreamRuntimeStartup_FreeRTOS.[cpp\|h] | Target specific initialization of wrappers and configurations |
| [Include\|Source]/Platform/win/pc/StreamRuntimeStartup_FreeRTOS.h" | Windows specific initialization of wrappers and configurations |
| [Include\|Source]/StreamRuntimeConfig.[h\|cpp] | Loads project configuration |
| [Include\|Source]/StreamRuntimeGUI.[h\|cpp] | Loads GUI |

**Table 8 Files in <SDK>\StreamRuntime\Common Directory**

| File | Description |
|---|---|
| CustomExtension | Custom extensions. |
| GUIConfig/User*Resource.h | Resource IDs generated by GSE |
| GUIConfigCustom/* | Custom IDs for use in Guiliani application. |
| Demo_*.[cpp\|h] | Specific code for the different demo parts |
| MyGUI_SR.[cpp\|h] | GUI-application entry point |

**Table 9 Files in <SDK>\StreamRuntime\Include and <SDK>\StreamRuntime\Source Directory**

| File | Description |
|---|---|
| GUIConfig.cpp | This file contains constants which hold the count of global properties, image resources, font resources, text resources, etc. |

**Table 10 Files in <SDK>\GSE\Share Directory**

### 9.4.2 Build configurations

There are two configurations available for StreamRuntime project.

1. Debug: The demo application runs from QSPI flash. Choose this configuration to debug the application.
2. Release: The application runs from QSPI flash. Choose this configuration to test the performance.

## 10 Debug Configurations

Under *Run → Debug Configurations → Renesas GDB Hardware Debugging* menu of e2Studio, debug configurations are created for each build configuration present in e2Studio workspace (Fig. 2). The name of each debug configuration is a combination of the project name and its build configuration. For example *SR_GuilianiDemo HardwareDebug* configuration is for project SR_GuilianiDemo with HardwareDebug configuration.

After a project is built, its debug configuration can be launched by clicking on button Debug. This will flash the binary file on the board and start debugging.

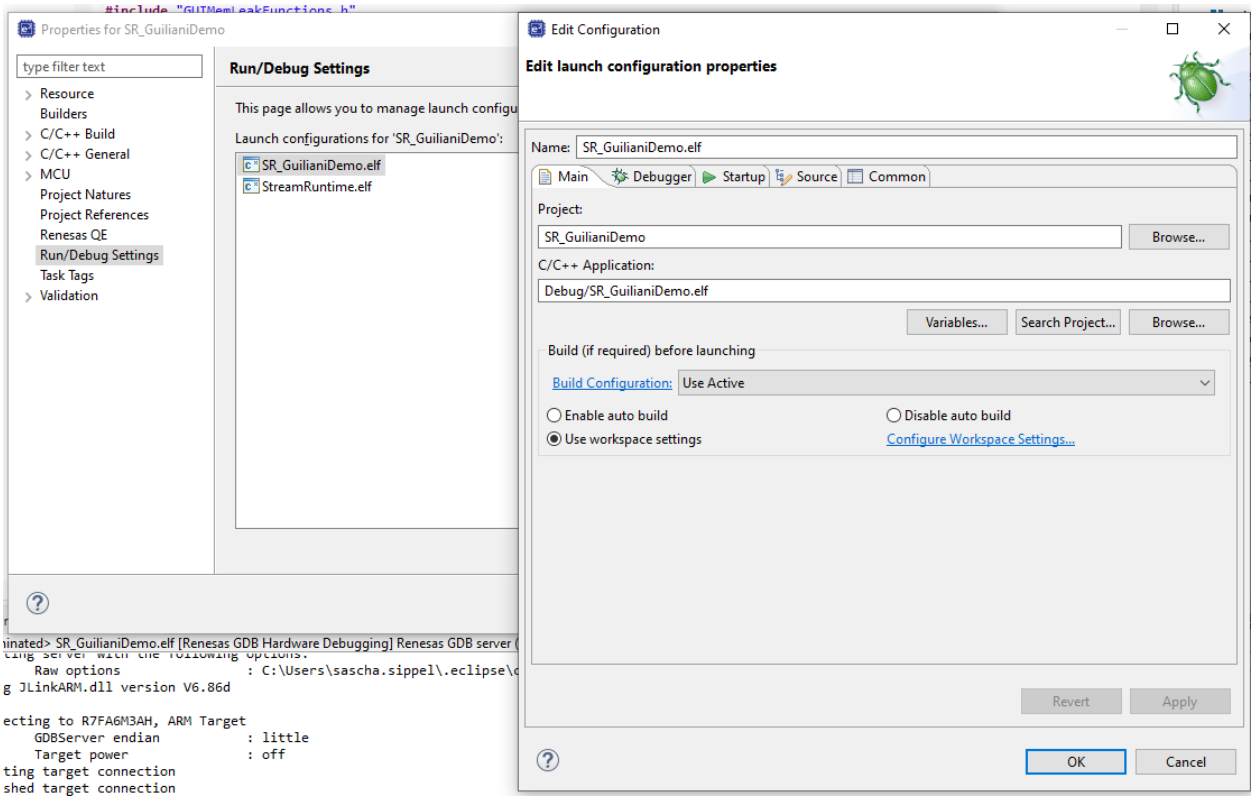



**Fig. 4 Debug Configurations**

# 11 Annex

### 11.1.1 Startup sequence of Guiliani Demo application

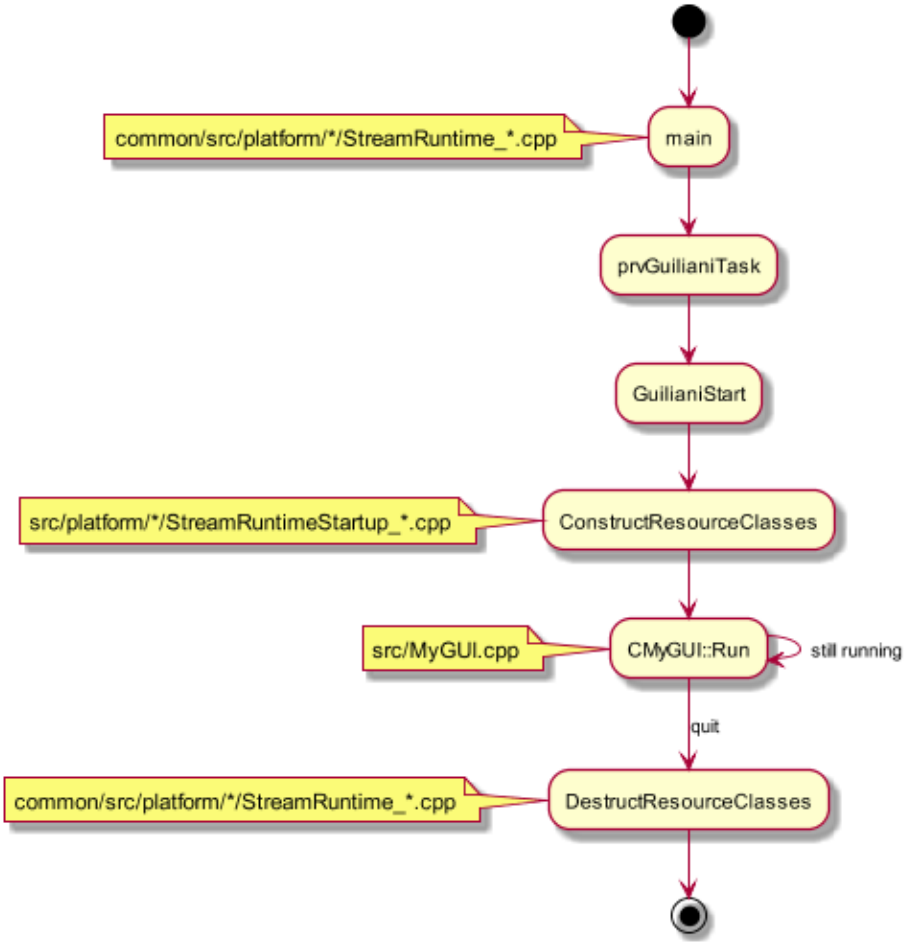



**Fig. 5 Startup Sequence of Guiliani Demo Application**