Quick Connect IoT

Quick Connect IoT is a combination of hardware and software modules that simplifies the design process of developing system solutions. It allows you to put together systems to evaluate sensors, connectivity, and the MCU in a full system environment with a minimal amount of set-up time or without writing the basic firmware structure.

Hardware modules are available with low-level drivers and middleware that allow you to immediately start writing the application layer code.

This manual reviews the Quick Connect IoT with an example that uses a specific sensor, but you are encouraged to consider other scenarios and explore all the devices available.

Contents

1. Reference Documents 2
2. Hardware Example 2
2.1 MCU Selection 2
2.2 MCU Setup 3
2.3 Sensor Selection 4
2.4 Connectivity Selection 4
2.5 Complete RA Hardware Setup 5
2.6 Complete RX Hardware Setup 6
3. RA FSP 6
3.1 Start Project 7
3.2 Insert Middleware 7
3.3 Resolve User Items Related to the Sensor Stack 7
3.4 Generate Code 9
3.5 API Examples 10
4. RX Smart Configurator 10
4.1 Start Project 11
4.2 Insert Component 12
4.3 Resolve User Items Related to the Sensor Stack 13
4.4 Generate Code 14
4.5 API Examples 15
5. Additional Information 15
6. Revision History 15
Appendix A 16

1. Reference Documents

Document number	Title
R20UT4827EG0100 Rev 1.00	EK-RA2L1 V1 User Manual
R20UT3558EG0100	RSK-RX651/RX65N User Manual
R36UZ0002EU0100	US082-ZMOD4410EVZ Evaluation Board Manual
R36UZ0004EU0100	US082-HS3001EVZ Evaluation Board Manual
R36UZ0006EU0100	US082-INTERPEVZ Evaluation Board Manual
R01AN5892EJ0100	Renesas Sensor Control Modules Firmware Integration Technology
R01AN5893EJ0100	Renesas HS300x Sensor Control Module Firmware Integration Technology
R01AN5897EJ0100	HS300x Sample Software Manual

2. Hardware Example

For hardware, a typical system might consist of an MCU, sensors, and a connectivity solution (the Connect IoT part).

2.1 MCU Selection

The Renesas MCUs are supported by various EKs, RSK, and target boards. For Quick Connect IoT, you can use any one of a number of MCU boards. The majority of Renesas MCU boards have headers that support standard form factor add-on boards. Typically, this includes some combination of PMODTM, Arduino, Mikro CLICK, and Grove.

For this manual, the example uses the EK-RA2L1; this device has memory footprints and is chosen based on full system requirements.

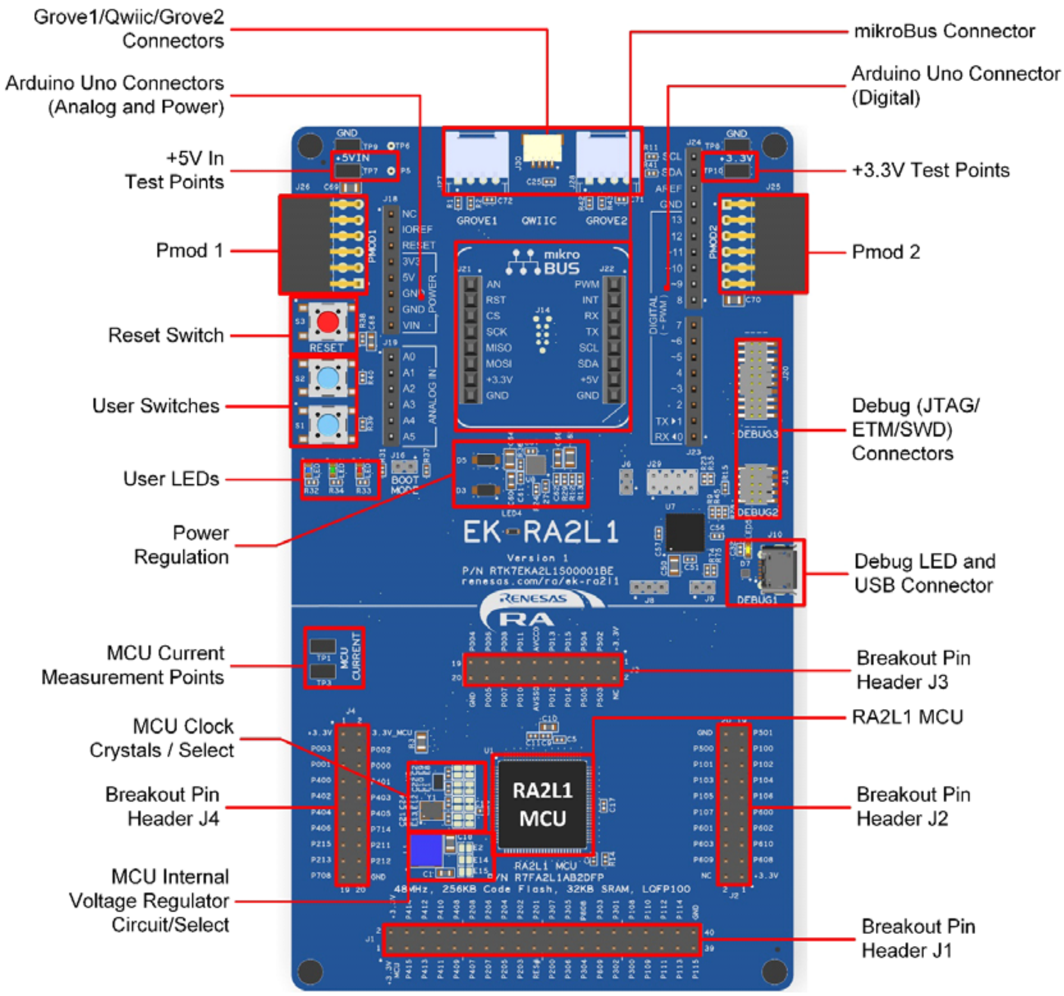

Figure 1. EK-RA2L1

2.2 MCU Setup

The amount of setup required depends on the interfaces and plug-in modules chosen. Some interfaces cannot be reconfigured; for example, if you use Arduino or MikroCLICK, there is no configuration required. If you choose to use PMODs, there are many PMOD pinouts available to connect to specific IO standards. Most of the MCU development boards support Digilent Type2A, extended SPI, Type3A, extended UART, Type6A, and extended I^2C. For the EK-RA2L, configure a PMOD to be Type3A (UART) for our connectivity choice and another for Type6A (I^2C) for the sensor connection. Reference the specific EK that you are using to reconfigure to the correct IO that is required. See the EK-RA2L1 User Manual for PMOD1 reconfiguration to Type6A. PMOD2 is the UART PMOD in this application.

Note: Some of the older RSKs and EK may require an interposer board available from Renesas to support Type6A. See Table 2 in the appendix for a list of boards requiring interposer.

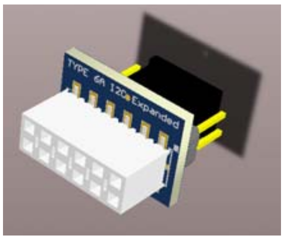

Figure 2. Type 6A Interposer

2.3 Sensor Selection

Renesas, non-Renesas sensors, and plug-in modules are supported and are chosen to meet the available plug-in module headers on the MCU.

Table 1. Sensor Board List

Part Number	Type
HS3001	Temperature and Humidity
ZMOD4410	Indoor Air Quality and Gas and Odor
ZMOD4510	Outdoor Air Quality

Note: This table is a selection of parts. Renesas continually releases new sensors, peripherals, and plug-in modules. Visit the Renesas website for more information.

2.4 Connectivity Selection

Renesas supports numerous connectivity choices from UART (wired) to Bluetooth and Wi-Fi. This manual focuses on a common connectivity solution for an IoT example, Wi-Fi using a PMOD.

2.5 Complete RA Hardware Setup

In the hardware setup, plugged in is the selected PMODs and the USB cable for the debugger interface, and in Figure 3, a complete solution for a connected Air Quality Sensor system is shown.

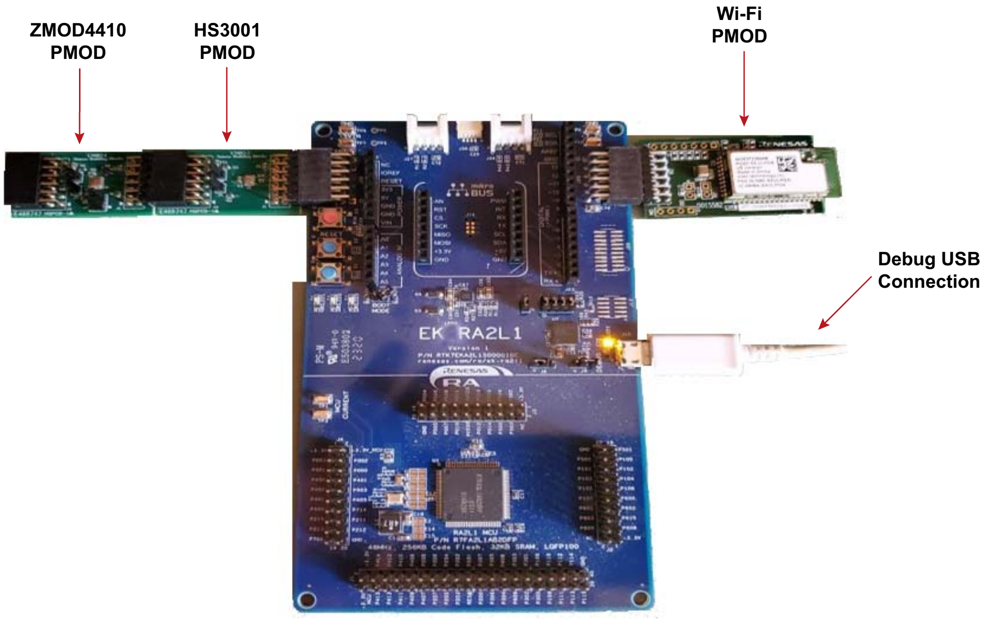

Figure 3. Air Quality System

2.6 Complete RX Hardware Setup

In Figure 4, the RX Sensor setup using the RX65N Envision Kit is an example of an RX hardware setup. Like the RA hardware setup, plug in the PMOD sensors that are used for the example solution and the USB debug cable. The example shown uses the RX65N Envision Kit.

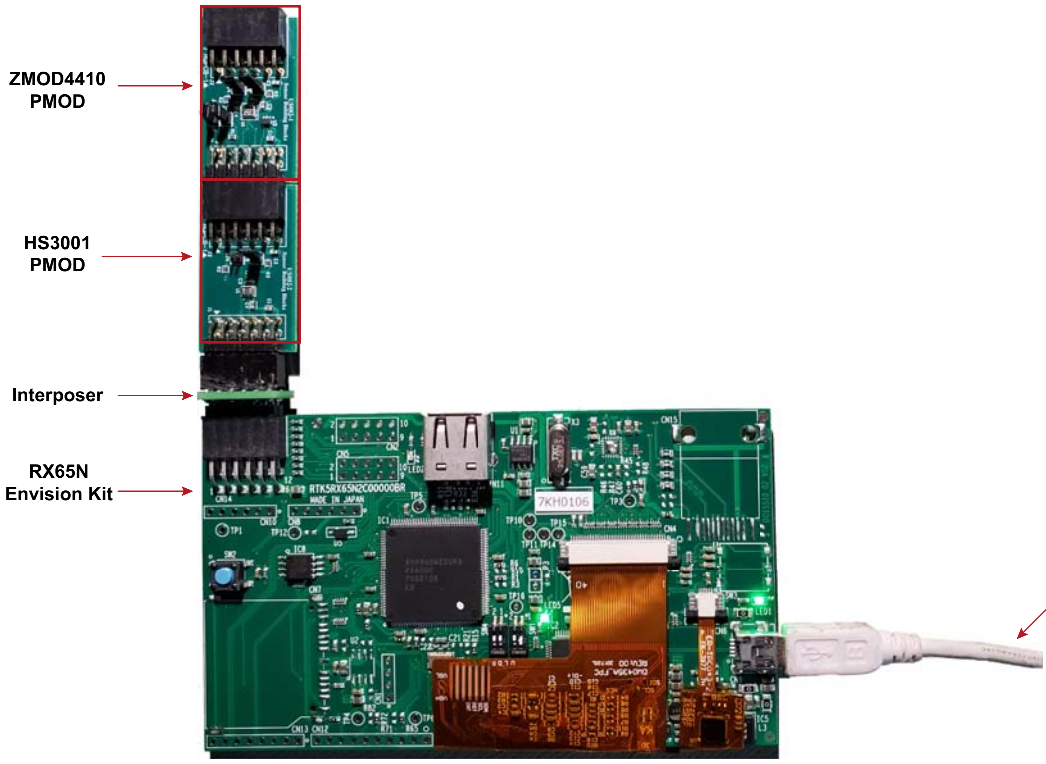

Figure 4. RX Air Quality Setup

3. RA FSP

The steps to write the application code are straight-forward. For this example, the steps are as follows:

1. Start a new RA project.
2. Select a BSP.
3. Insert Middleware Stacks.
4. Resolve middleware issues (remove the red out) by defining user items such as which I^2C port to use.
5. Generate code.

At this point, you have a buildable project with limited to no debug required, so you can start to write your application code. Typically, this consists of instantiating USER buffers for the data, and then a simple POSIX such as APIs to talk to the devices with middleware instantiated.

Note: This is not intended as training on FSP, but rather an overview of the new middleware available to provide you with a system solution of MCUs, sensors, and connectivity.

3.1 Start Project

Starting a project is as simple as follows:

1. Select the correct project type.
2. Name the project.
3. Select the BSP.
4. Select the type of project (executable or Library) including RTOS support.
   Because this example is for non-RTOS, Bare Metal Minimal is chosen.

3.2 Insert Middleware

After setting up the project, you are in the FSP Configuration View. (For additional information, see the FSP manuals that are available through the SmartBrowser or help facilities of e^{2}Studio.) The following steps are for adding a single sensor:

Select **New Stack** → **Middleware** → **Sensor** → **ZMODXXXon rm_zmod4xxx**. (See Figure 5.)

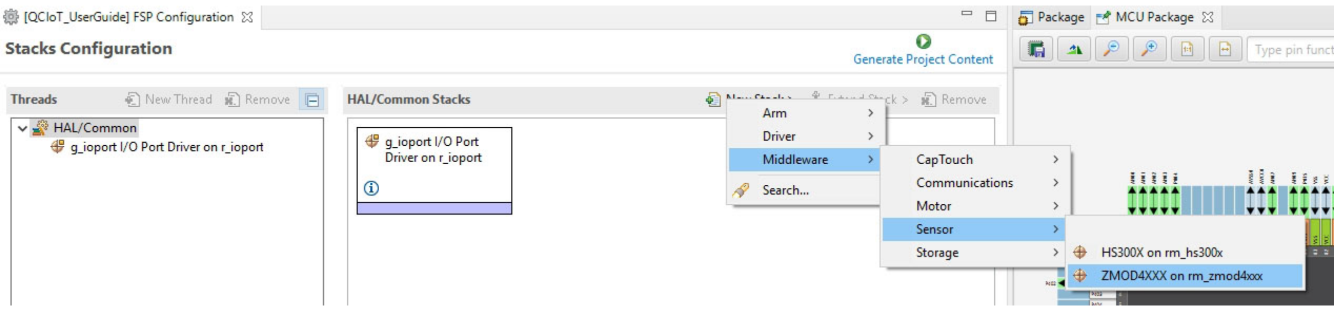

Figure 5. Middleware Selection

3.3 Resolve User Items Related to the Sensor Stack

When the stack is instantiated, it displays red indicating that you need to select user-configurable items. In the case of the ZMOD, you need to select the device, library type, the interface, and the source for the measurement trigger.

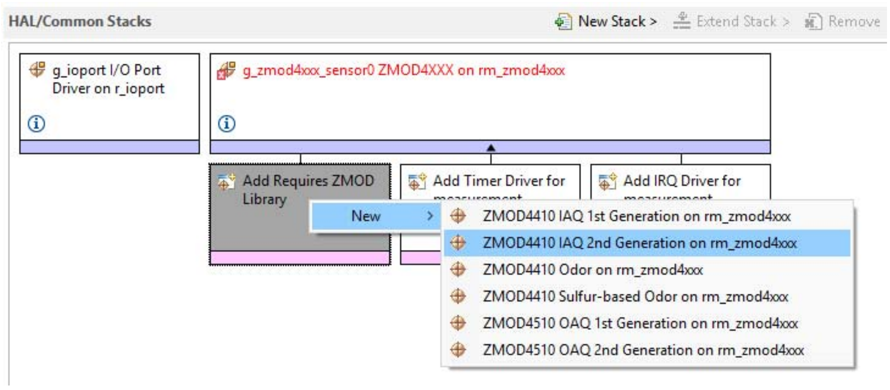

Figure 6. ZMOD4410 2nd Generation Indoor Air Quality Selection

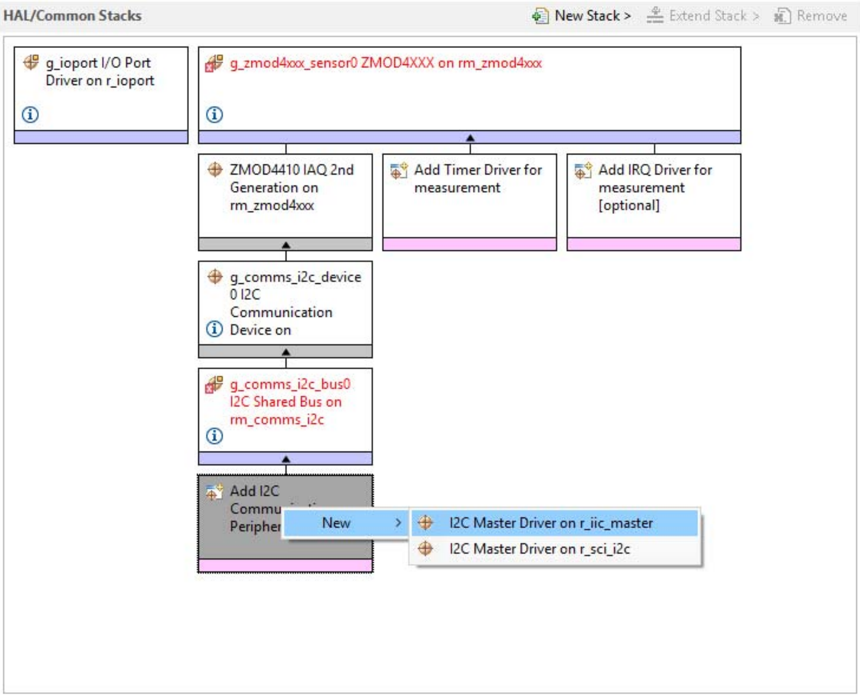

Figure 7. I^{2}C Master on r_iic_master Selection

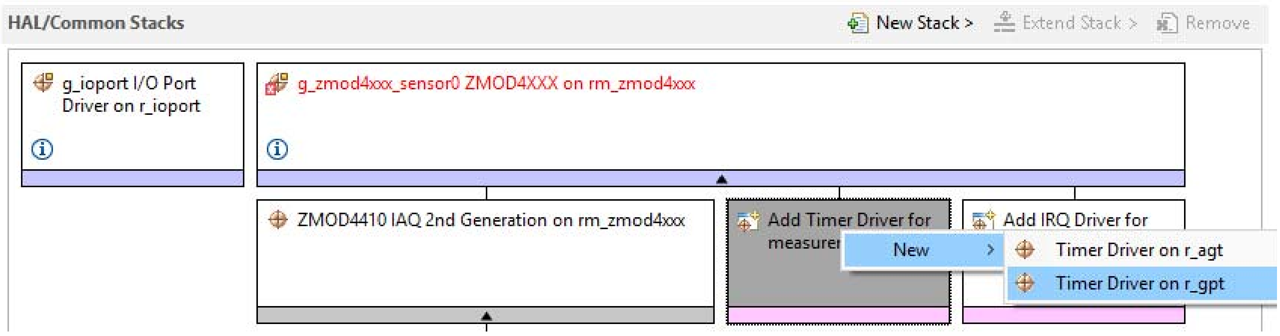

Figure 8. Pop-Up Help on GPT Error

Hint: The HAL/Common Stacks blocks remain red until all the configurations items are satisfied. Hovering on the red X does pop-up the error that is in the particular block. In this case, the GPT trigger for measurement is red. See Figure 9 for an example of pop-up help.

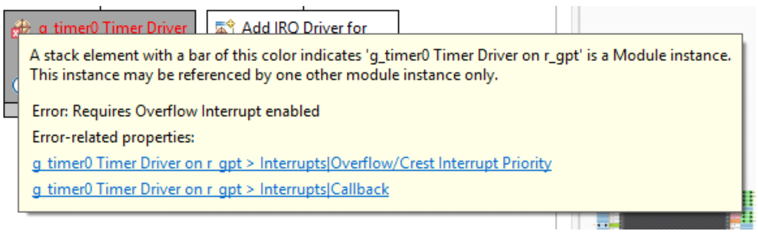

Figure 9. Pop-Up Help On GPT Error

Next, this if fixed by enabling the Overflow interrupt on the GPT. The callback function and name are defined by the middleware.

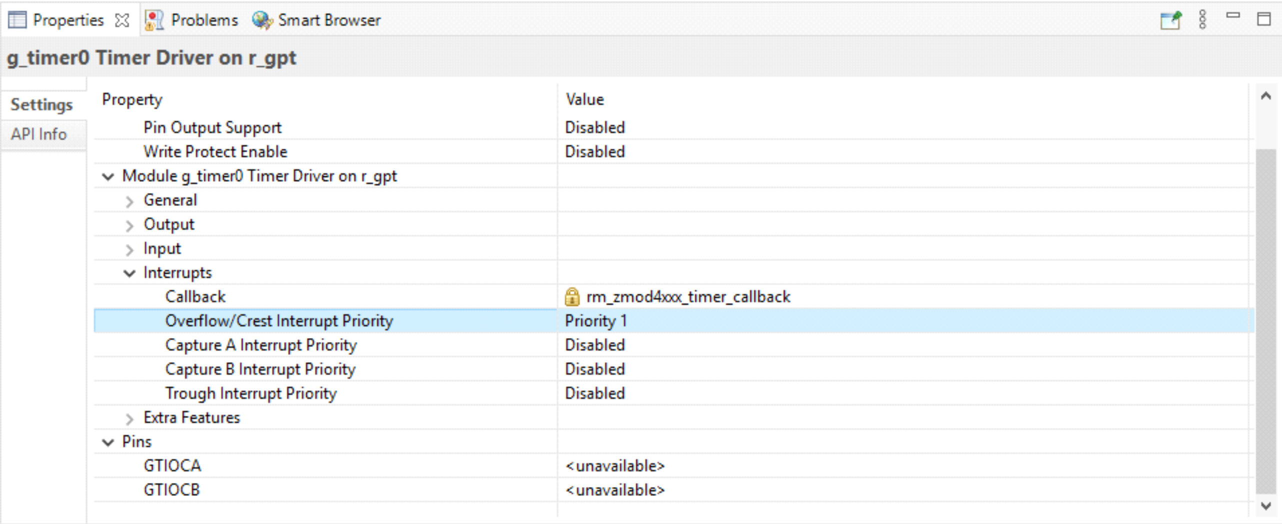

Figure 10. GPT Overflow Interrupt Setting

3.4 Generate Code

After all the stacks/HAL code is satisfied and all the user selections are made, you must **Generate Project Content**. See Figure 11.

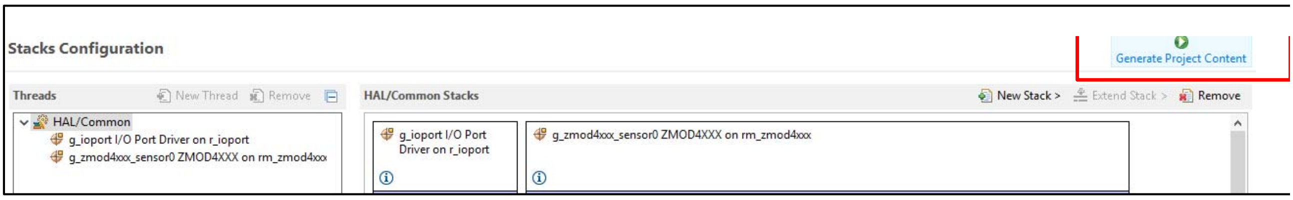

Figure 11. Generate Project Content

3.5 API Examples

As we indicated, the function calls are now available to you so that you can start writing your application (i.e. how you will use the temperature, humidity and air quality values in you application). The data types are all defined, so you only need to instantiate buffers for you application.

Example of Buffer instantiation:

```
fsp_err_t err;
rm_zmod4xxx_raw_data_t raw_data;
rm_zmod4xxx_iaq_2nd_data_t zmod4xxx_data;
```

ZMOD Public API:

```
fsp_err_t RM_ZMOD4XXX_Open(rm_zmod4xxx_ctrl_t * const p_api_ctrl, rm_zmod4xxx_cfg_t
const * const p_cfg);
fsp_err_t RM_ZMOD4XXX_MeasurementStart(rm_zmod4xxx_ctrl_t * const p_api_ctrl);
fsp_err_t RM_ZMOD4XXX_MeasurementStop(rm_zmod4xxx_ctrl_t * const p_api_ctrl);
fsp_err_t RM_ZMOD4XXX_StatusCheck(rm_zmod4xxx_ctrl_t * const p_api_ctrl);
fsp_err_t RM_ZMOD4XXX_Read(rm_zmod4xxx_ctrl_t * const p_api_ctrl,
rm_zmod4xxx_raw_data_t * const p_raw_data);
```

Note: Since the middleware will support multiple instances of ZMOD sensors on multiple I^2C buses, these will typically be abstracted one layer to account for multiple device instances and multiple configurations. In this case, your calls may be abstracted by a function table in the configuration instance. For details, see to the FSP manuals.

For this case shown, it is defined by the ctrl instance for Sensor 0:

```
rm_zmod4xxx_instance_ctrl_t g_zmod4xxx_sensor0_ctrl;
```

ZMOD API Examples Device 0 defined by:

```
err = g_zmod4xxx_sensor0.p_api->measurementStart(g_zmod4xxx_sensor0.p_ctrl)
```

4. RX Smart Configurator

The steps to follow so that you can write the application code are straight-forward. For this example, the steps are as follows:

1. Start a new RX project.
2. Select the BSP and device.
3. Insert components.
4. Resolve component configuration issues by defining user items such as which I^2C port to use.
5. Generate Project Code.

At this point we will have buildable project with limited to no debugging required. The user can then start to write their application code. Typically, this consists of instantiating USER buffers for the data, and then a simple POSIX such as APIs to talk to the to devices with middleware instantiated.

Note: This is not intended to be training on the RX Smart Configurator, but rather an overview of the new middleware available to get you to a system solution of MCUs, sensors, and connectivity.

4.1 Start Project

Starting a project is as simple as follows:

1. Select the correct project type.
2. Name the project.
3. Select the BSP.
4. Select the type of project (executable or Library) including RTOS support.
   Because this example is for non-RTOS, Bare Metal Minimal is chosen.

After generating the RX C/C++ executable project, it opens in the RX Configuration View as shown in Figure 12.

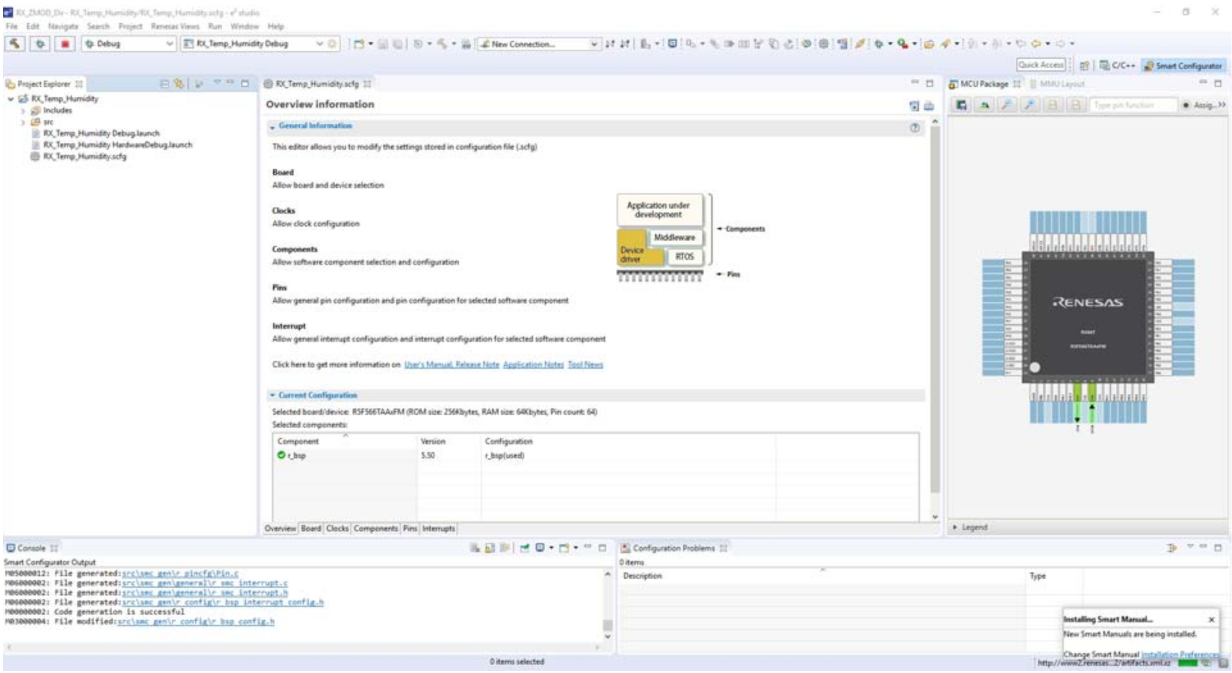

Figure 12. RX Smart Configuration View (Empty Project)

4.2 Insert Component

After setting up the project, you are in the FSP Configuration View. The following example adds a single sensor, the HS3001 Humidity and Temperature sensor. For additional information, reference the Smart Configuration manuals that are available through the SmartBrowser or help facilities of e^{2}Studio.

Select **Components Tab** → **r_hs3001_rx** → **Finish**. (See Figure 13.)

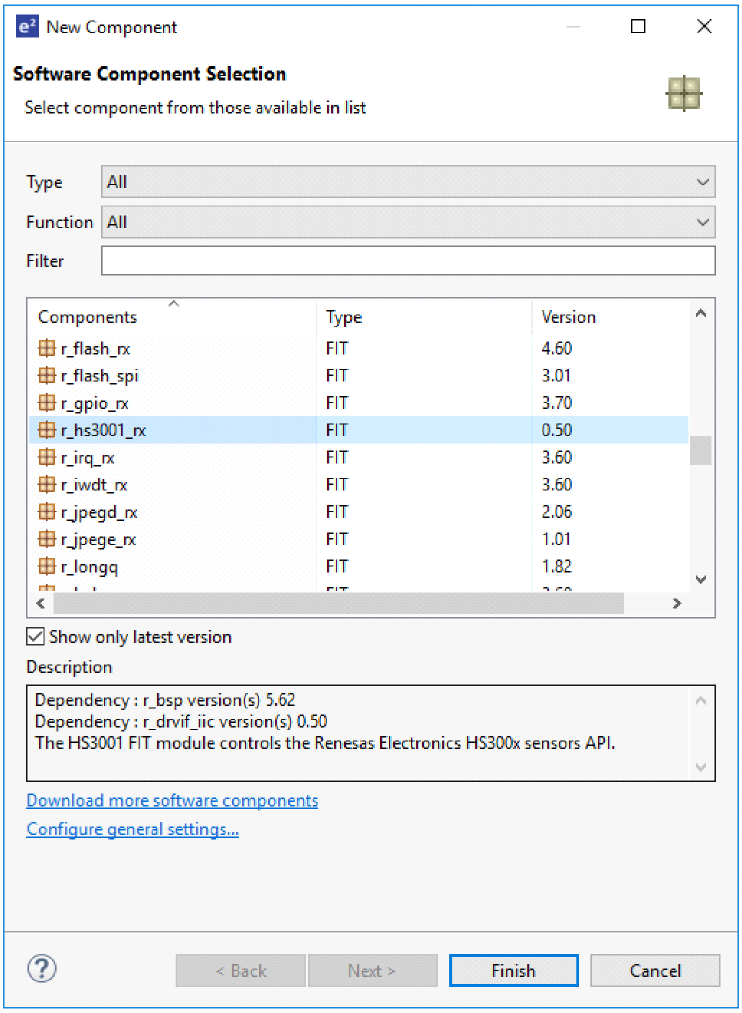

Figure 13. RX Component Selection

4.3 Resolve User Items Related to the Sensor Stack

After the component is instantiated, it scans for dependencies and inserts the required I²C driver to support the device. Your component list reflects this change. See Figure 14.

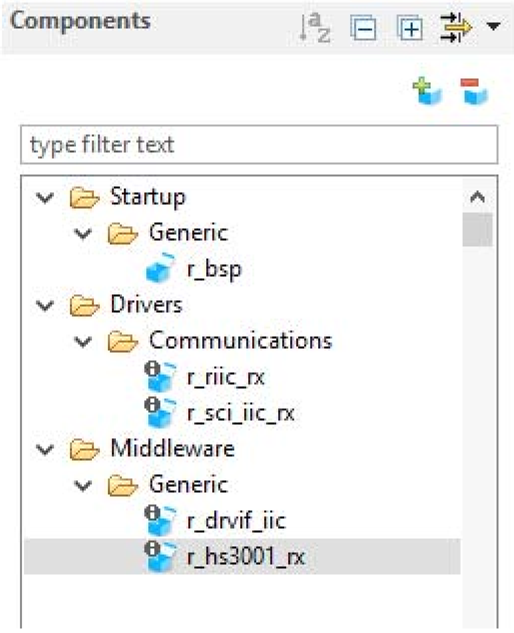

Figure 14. RX Component List

Next, the user choices require configuring. Select the number of I²C channels that are supported in the driver. Select **r_drvif_iic** in the component list followed by selecting the number of IIC communications lines. In this case, select 1 and choose **RX FIT IIC**. See Figure 15.

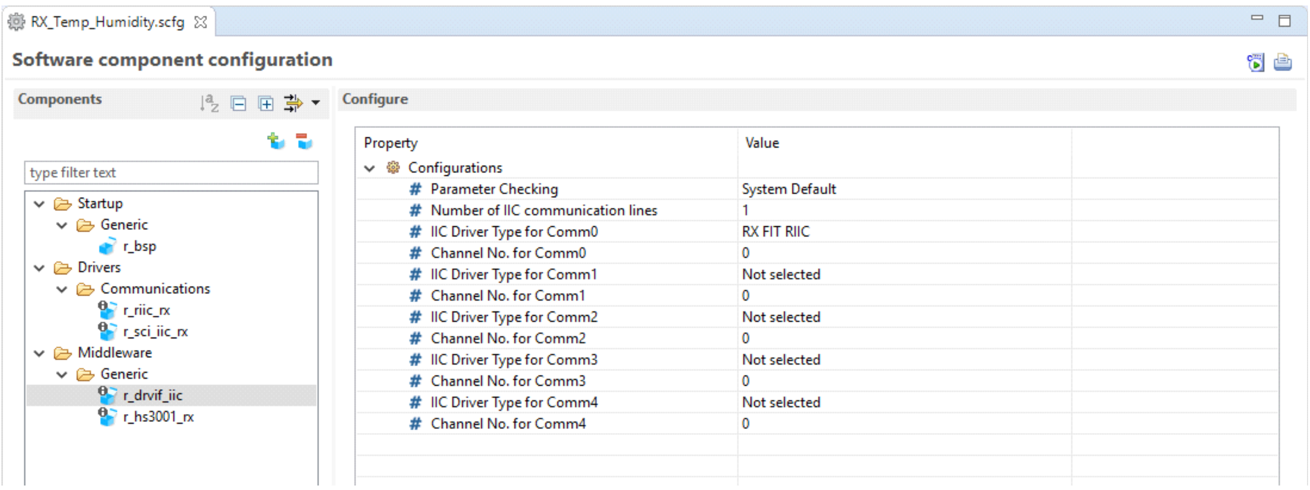

Figure 15. r_drvif_iic configuration

Finally, the actual pin connection is chosen. Go to the **Pins Configuration** tab, select **RIIC0** in the list, and the two used pins display. Use the **Pin Number** pulls downs, and choose the pins connected in your design. The used pins appear red. See Figure 16.

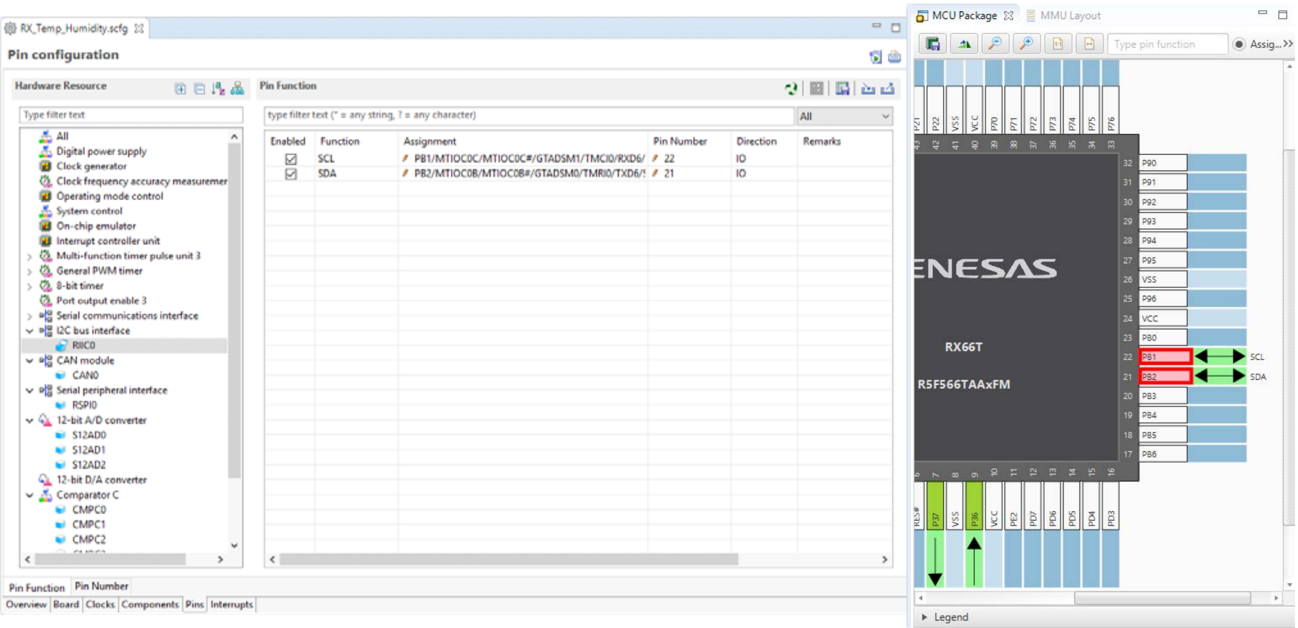

Figure 16. I²C Pin selection on RX

Hint: Configuration problems are show in the **Configuration Problems** tab. In default layout, this is in the lower right of screen.

4.4 Generate Code

After all the component settings are satisfied and there are 0 items in the Configuration problems, you simply generate code with the feature, **Generate Project Content**. (See Figure 17.)

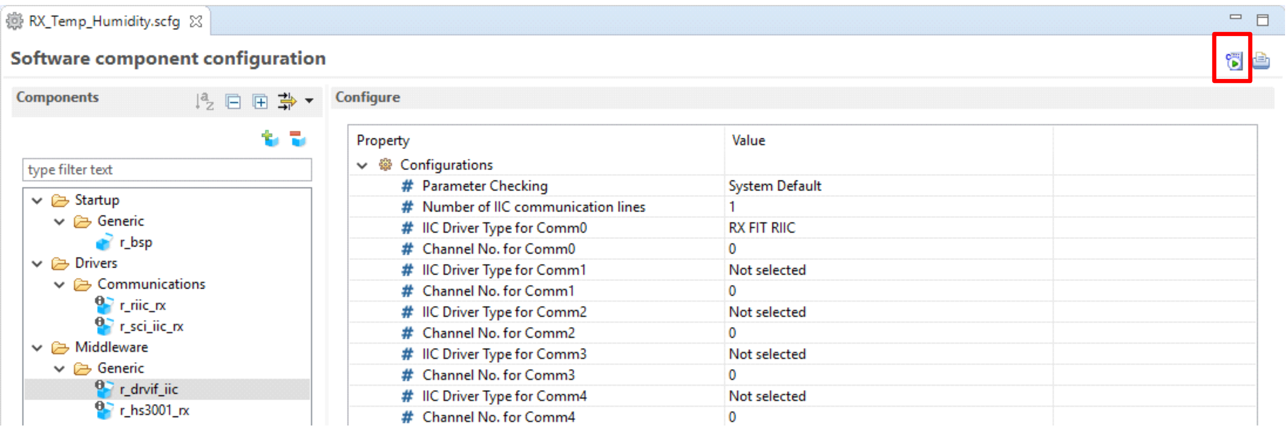

Figure 17. Generate Project Code

4.5 API Examples

As indicated, the function calls are available to you so you can start writing your application (for example, how you use the temperature and humidity values in your application). The data types are all defined, so you only need to instantiate buffers for you application.

Example of Buffer instantiation:

```
e_hs3001_err err;
```

HS3001 Public API:

```
hs3001_err_t R_HS3001_Open (hs3001_ctrl_t *const p_ctrl, hs3001_cfg_t *const p_cfg);

hs3001_err_t R_HS3001_Read (hs3001_ctrl_t *const p_ctrl, uint8_t *p_dest, uint32_t
bytes, uint16_t datatype);

hs3001_err_t R_HS3001_IOCtl (hs3001_ctrl_t *const p_ctrl, uint8_t *p_buf, uint32_t
bytes, uint16_t command);

hs3001_err_t R_HS3001_Close (hs3001_ctrl_t *const p_ctrl);

hs3001_err_t R_HS3001_GetVersion (hs3001_version_t * p_version);


void r_hs3001_callback(drvif_iic_event_t event, drvif_iic_instance_t * p_inst,
hs3001_cfg_t * p_cfg);
```

Note: Because the Middleware supports multiple instances of the HS3001 sensors on multiple I^2C busses, these are typically abstracted one layer to account for multiple device instances and multiple configurations. In this case, your calls may be abstracted by a function table in the configuration instance. For details see the Smart Configurator manuals.

For this case shown, it is defined by the ctrl instance for Sensor 0:

HS3001 API Examples Device 0 defined by:

```
err = g_hs300x_sensor0.p_api->measurementStart(g_hs300x_sensor0.p_ctrl);
```

5. Additional Information

For additional information on the Quick Connect IoT solutions and supporting documents, visit Renesas Quick Connect.

6. Revision History

Revision	Date	Description
1.0	Jul 2, 2021	Initial release.

Appendix A

Boards that require interposer.

Table 2. Kits Requiring Type 6A Interposer

Family/Device Group	Board Name
RA/RA4W1	EK-RA4W1
RA/RA2A1	EK-RA2A1
RA/RA4M1	EK-RA2A1
RA/RA6M1	EK-RA6M1
RA/RA6M2	EK-RA6M2
RA/RAM3	EK-RA6M3
RA/RAM3G	EK-RA6M3G
RX/RX111	RX111-Starter-Kit
RX/RX231	RX231-Starter-Kit
RX/RX23W	RX23W-Starter-Kit
RX/RX23T	RX23T-Starter-Kit
RX/RX24T	RX24T-Starter-Kit
RX/RX24U	RX24U-Starter-Kit
Synergy/S5D9	PK-S5D9
Synergy/S3A7	DK-S3A7
Synergy/S128	DK-S128
Synergy/S1JA	TB-S1JA
Synergy/S3A6	TB-S3A6
Synergy/S7G2	DK-S7G2

IMPORTANT NOTICE AND DISCLAIMER

Corporate Headquarters

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

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/

Trademarks