RZ/N2L Group

R01AN6800EJ0110
Rev.1.10
Aug 7, 2023

RZ/N2L Industrial Network SOM Kit Startup Guide: LED Blinky Sample Program

Introduction

This document explains LED blinky sample program for those who use RZ/N2L Industrial SOM Kit for the first time.

Regarding the sample program written in the flash memory of this SOM Kit, see the application note “HTTP server sample program”.

Target Device

RZ/N2L

Contents

1. Overview

This document describes how to set up and debug RZ/N2L Industrial Network SOM Kit, and how this sample program works.

This SOM Kit is equipped with GreenPAK as a general IO interface. GreenPAK is connected to LEDs and Dip Switches and jumper pins on the carrier board, and RZ/N2L uses I2C communication to control GreenPAK for I/O control.

In this sample program, LEDs and Dip Switches are controlled via GreenPAK to blink LED.

1.1 Reference

Technical information about GreenPAK and RZ/N2L is available via Renesas.

Table 1. Technical Inputs

Index	Technical Inputs
1	r01uh0955ejxxxx-rzn2l.pdf (RZ/N2L User's Manual: Hardware)
2	r01an6434ejxxxx-rzt2-rzn2-fsp-getting-started.pdf (Getting started with Flexible Software Package)
3	r12ut0020edxxxx-rzn2l-som-kit-hw.pdf (RZ/N2L Industrial Network SOM Kit Use's Manual)
4	SLG46537 Datasheet (GreenPAK documentation)

2. Project Setup

2.1 Requirements

Table 2. Requirements

Item	Vender	Description
Board	Renesas Electronics	RZ/N2L Industrial Network SOM Kit
IDE	IAR Systems	● Embedded Workbench® for ARM Version 9.30.1 Please apply patch (EWARM_Patch_for_RZN2L) which is available in http://www.renesas.com/rzn2l. Regarding how to apply the patch, please read the readme file in patch file.
	Renesas Electronics	● e² studio 2023-04 ● FSP Smart Configurator 2023-04 ● RZ/N2L Flexible Software Package (FSP) v1.2.0 Please download from the link below. https://github.com/renesas/rzn-fsp/releases/tag/v1.2.0
Emulator	IAR Systems	I-jet
	SEGGER	Hardware: J-Link Software: J-Link Commander V7.82f *1

*1: J-Link Commander is used for erasing flash memory.
J-Link Commander is included in “J-Link Software and Documentation Pack” on the following site.
https://www.segger.com/downloads/jlink/

2.2 Hardware

This document describes the major hardware. Refer to RZ/N2L Industrial Network SOM Kit user's manual and schematic for more board details.

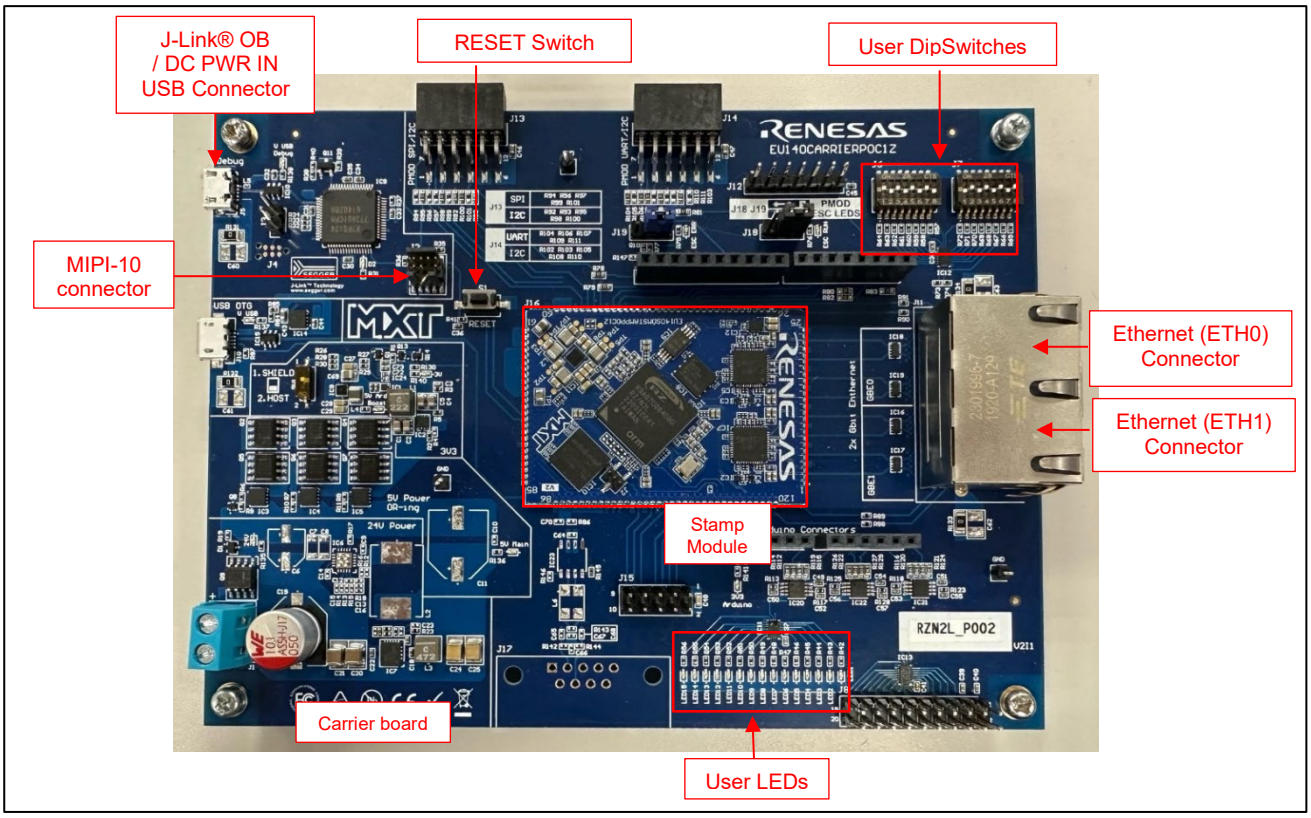


Figure 2-1 RZ/N2L Industrial Network SOM Kit

2.3 Setup the Board

Setting the board for running sample program is shown below.

1. Connect the I-jet to J2 or the USB cable to J5 for J-link OB on Carrier board.

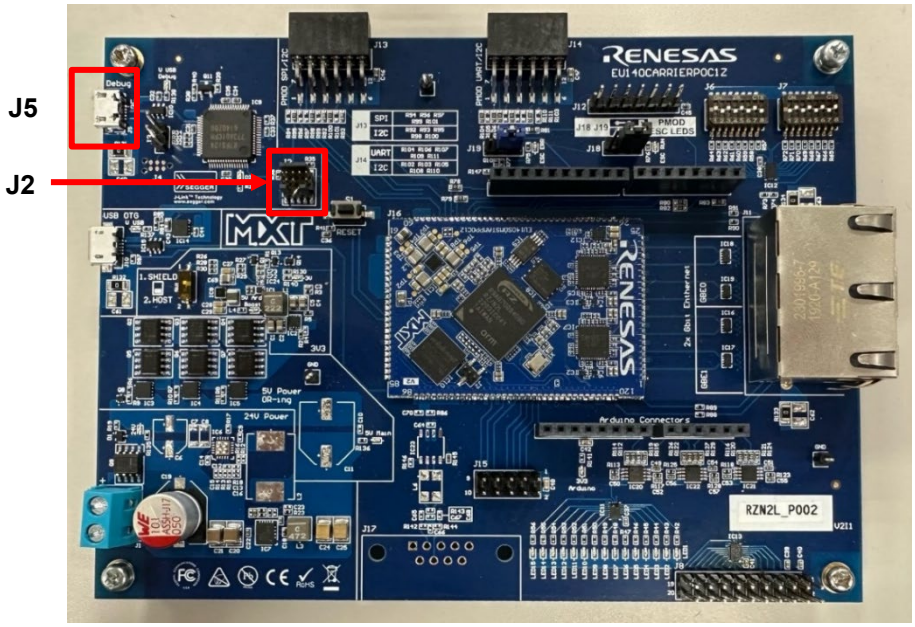


Figure 2-2 Setup the SOM Kit

2. Power is supplied by connecting USB Micro-B cable to the USB connector “J5) of the Carrier board.

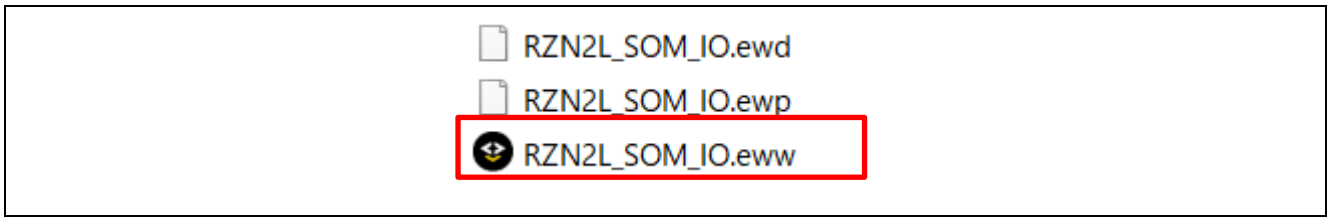
3. Running the sample application

3.1 Build and debug sample code for EWARM

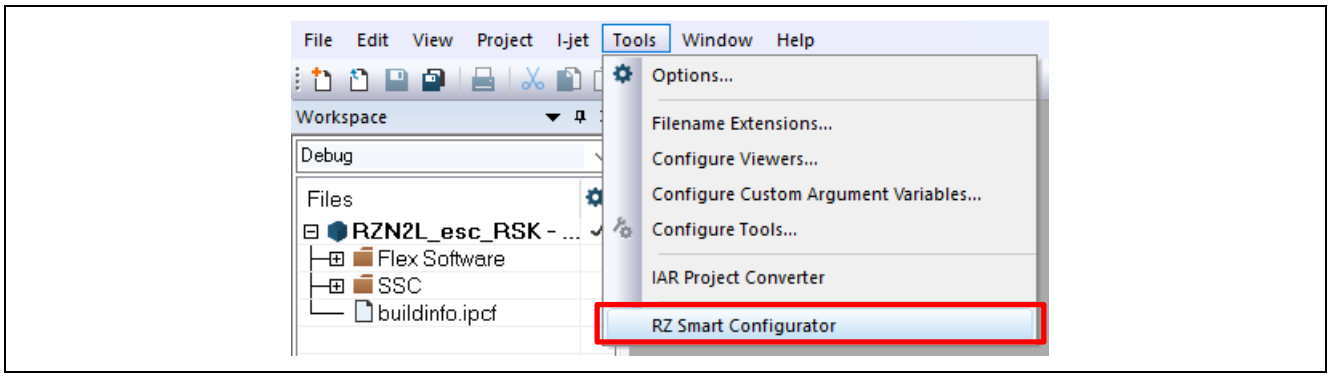
Build the sample code and load it into RAM using IAR Embedded Workbench.

Note). Please install FSP Smart Configurator in advance.
Refer to the latest getting started guide.(R01an6434ejxxxx- rzt2-rzn2.pdf)
Replace the project name in the figure with the project name of this sample project.

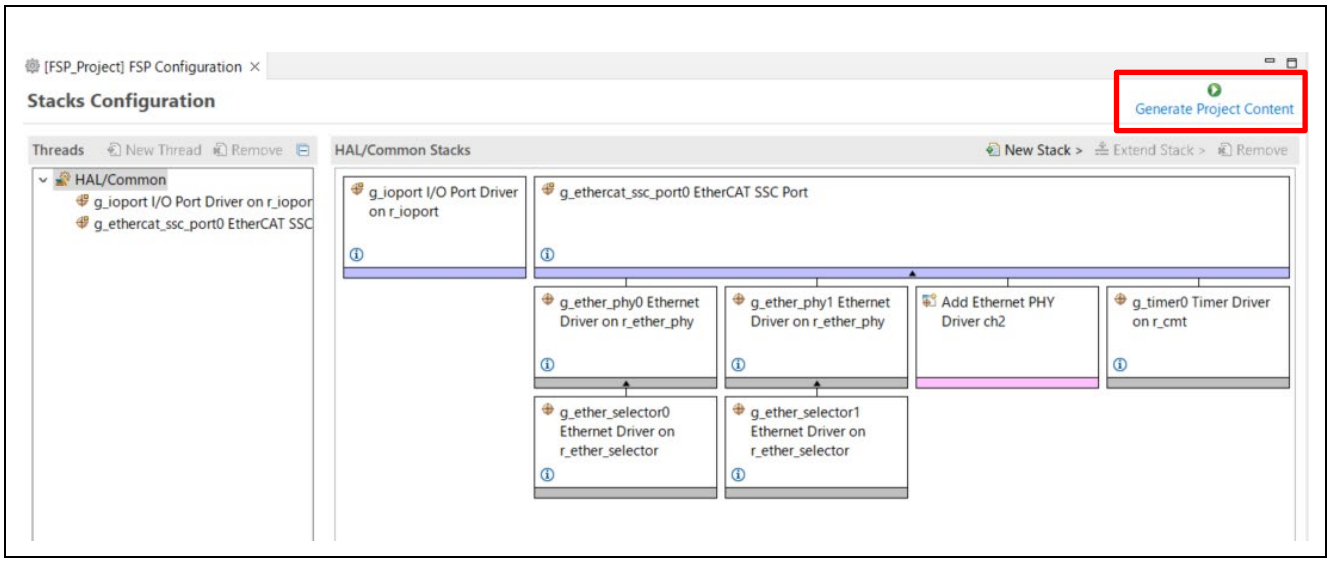
1. Open the sample project. **“GPAK_IO\ewarm\RZN2L_SOM_IO\RZN2L_SOM_IO.eww”**



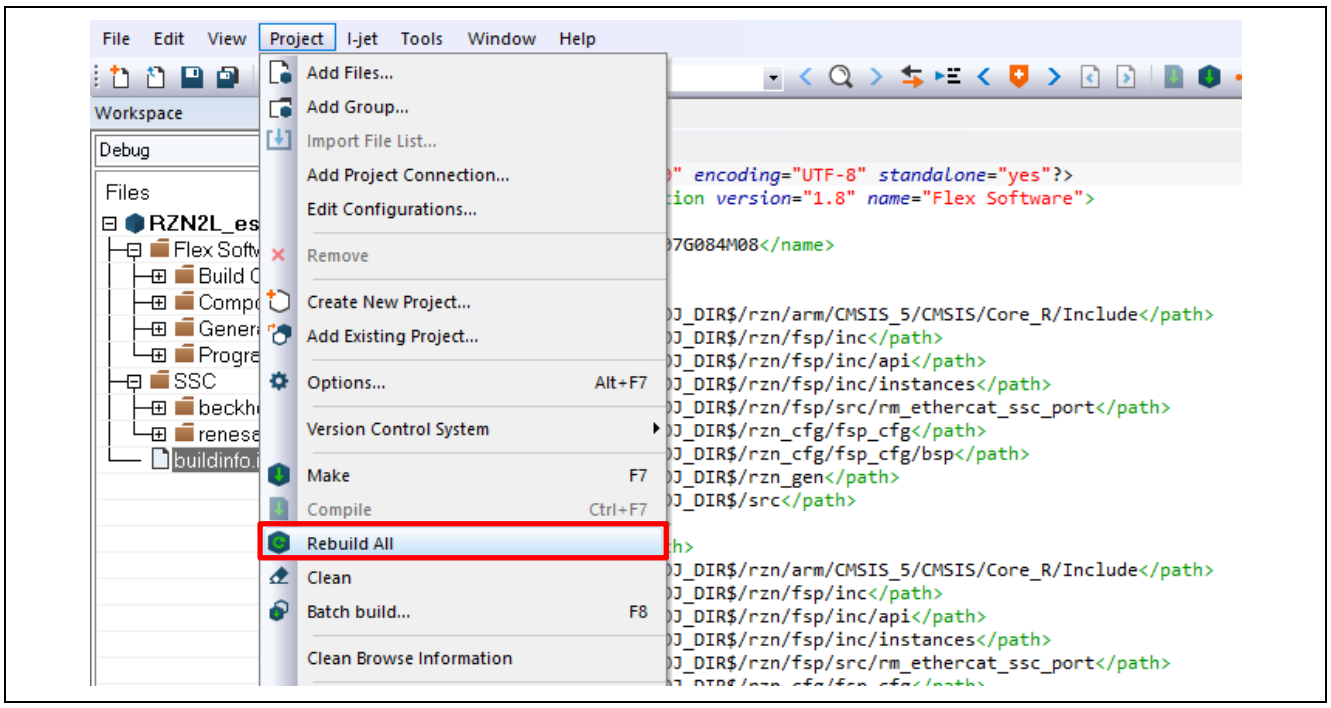
2. Open the “RZ Smart Configurator”



3. Generate the code with "Generate Project Content".

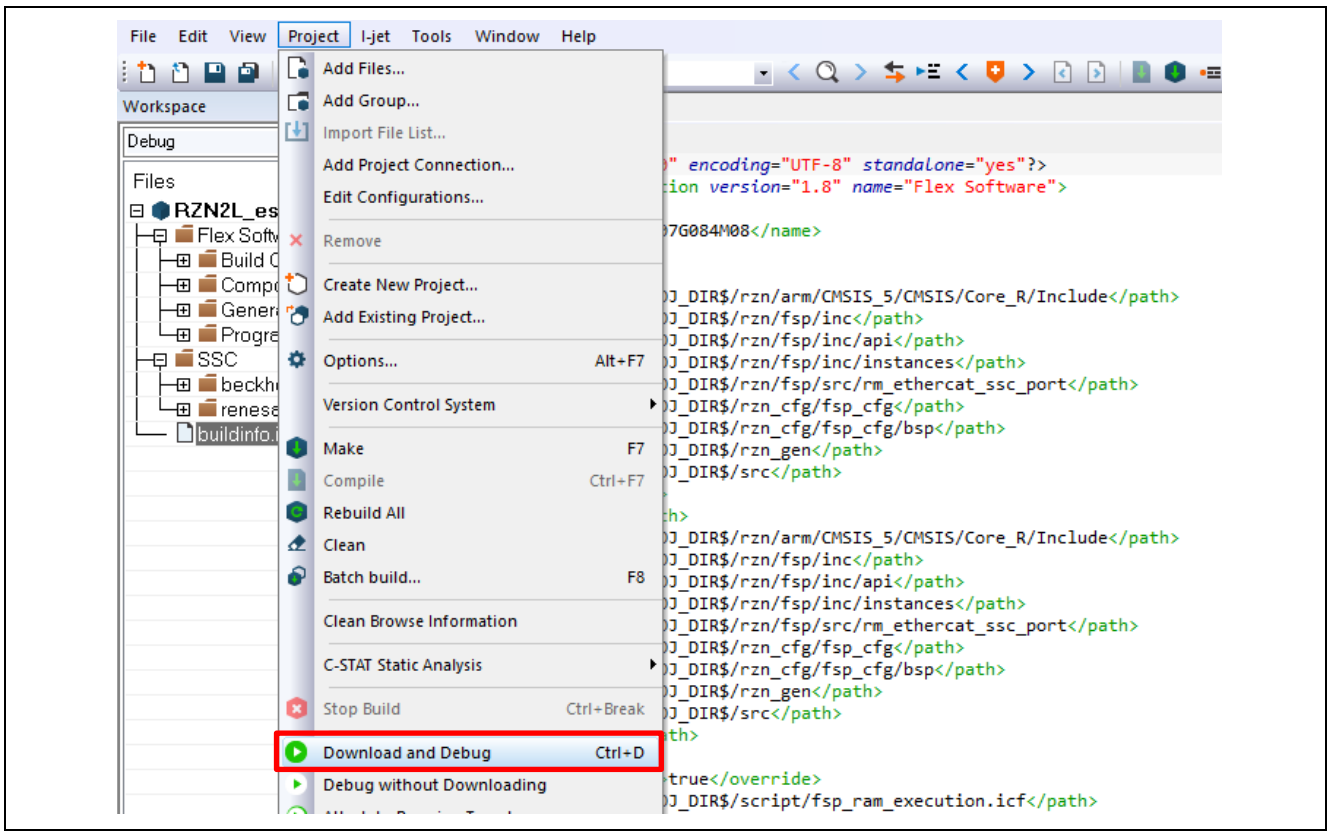


4. Select the “Rebuild All” item from the “Project” menu to rebuild the project.

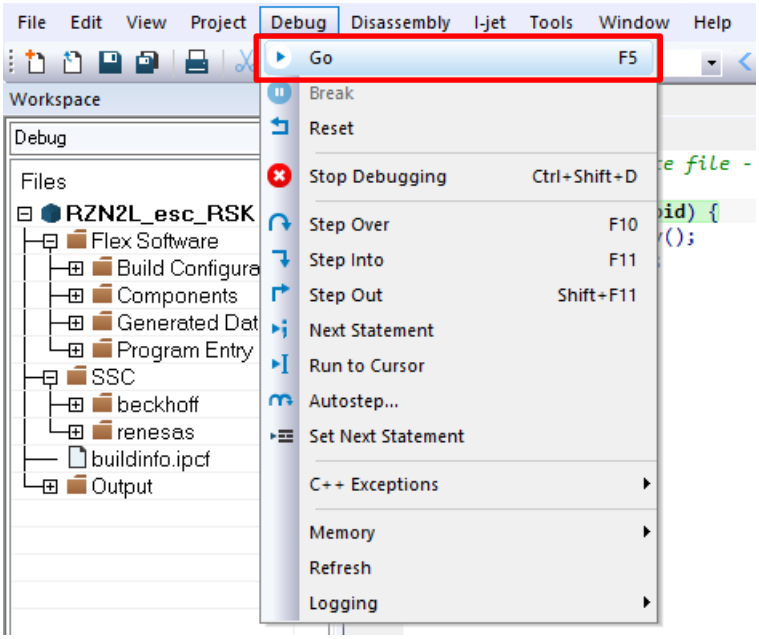


5. Press the “RESET” switch of the RZ/N2L Industrial Network SOM Kit.

6. While the board and I-jet are connected, click on the “Download and debug” button in the “Project” toolbar.



7. Press the "Resume" button for the project. Program is running.



3.2 Build and debug sample code for GCC

3.2.1 Erasing the flash memory

First, erase the flash memory by following the steps below. This step can be skipped after erasing the flash memory.

Open the J-Link Commander.

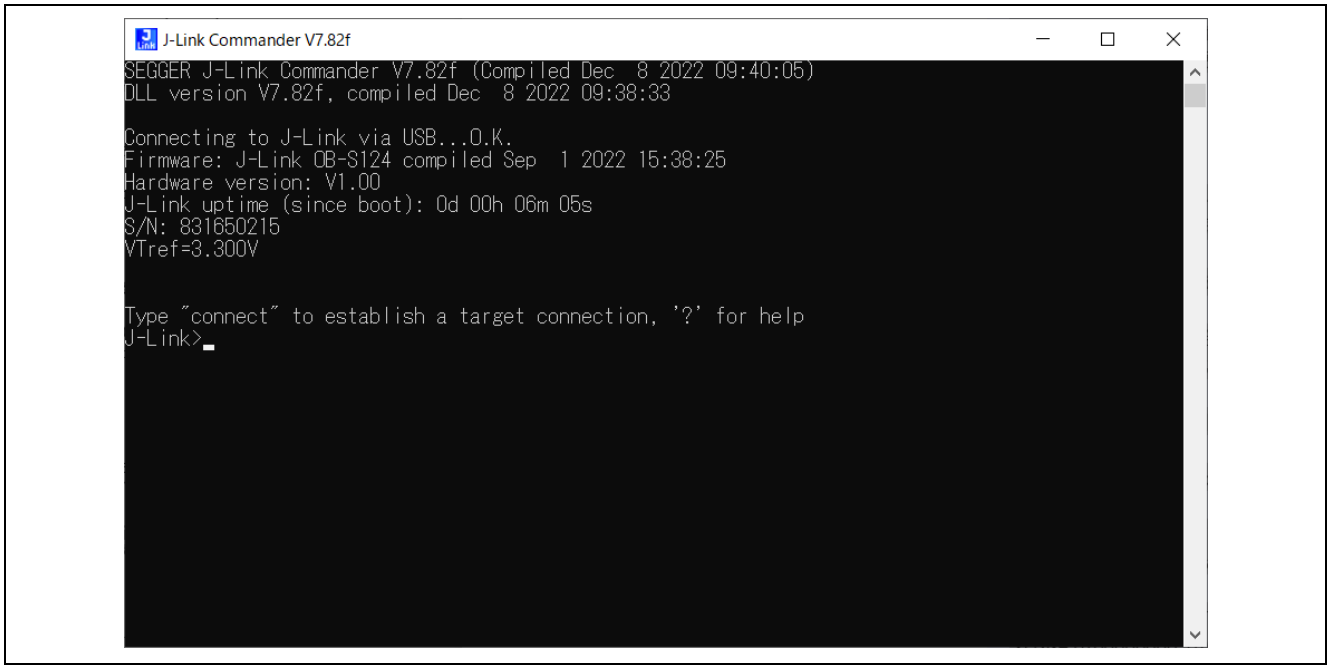


Figure 3-1 Open J-Link Commander

First, type “connect” to establish a target connection and press enter.

Next, specify the connection conditions as follows.

- Device> (Default = press enter)
- TIF>S
- Speed> (Default = press enter)

After that, confirm the message “Cortex-R52 identified.” Is displayed.

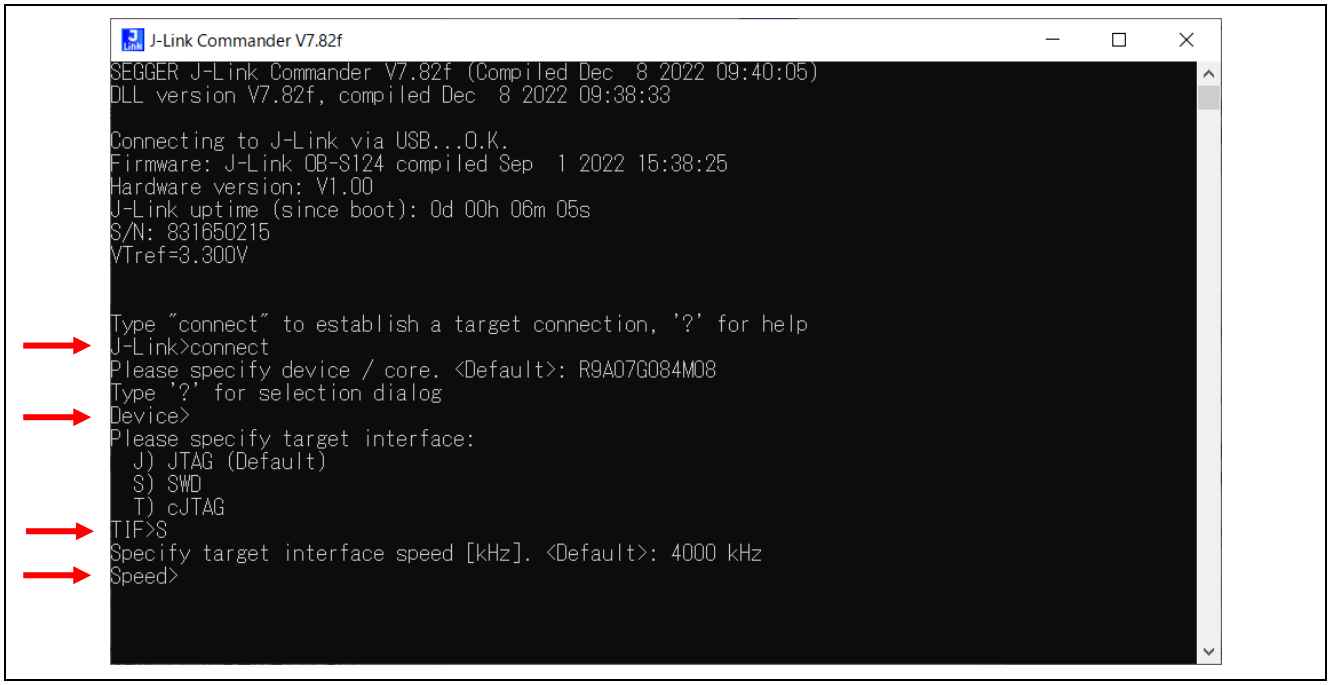


Figure 3-2 Connection conditions (1/2)

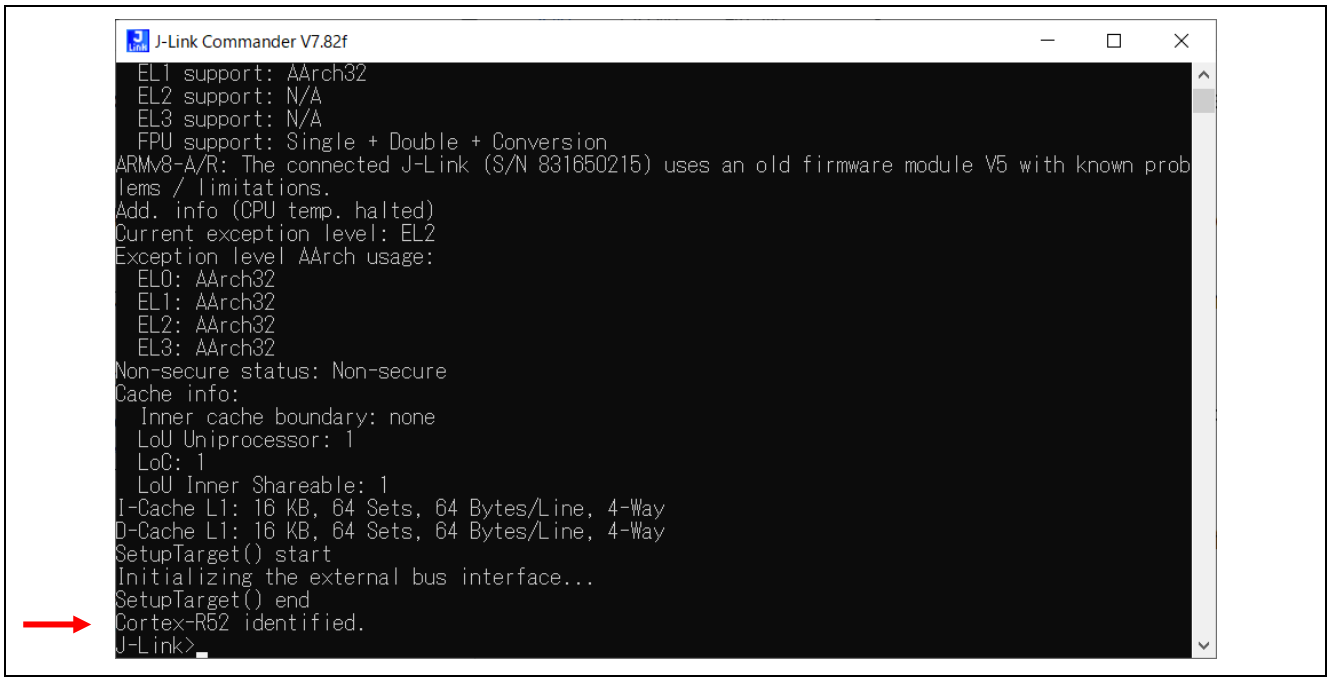


Figure 3-3 Connection conditions (2/2)

Use the commands below to enable flash erase and erase the flash memory.

- >exec EnableEraseAllFlashBanks
- >Erase 0x60000000, 0x60100000

After that, confirm the message “Erasing done.” Is displayed.

Enter “q” to exit J-Link Commander.

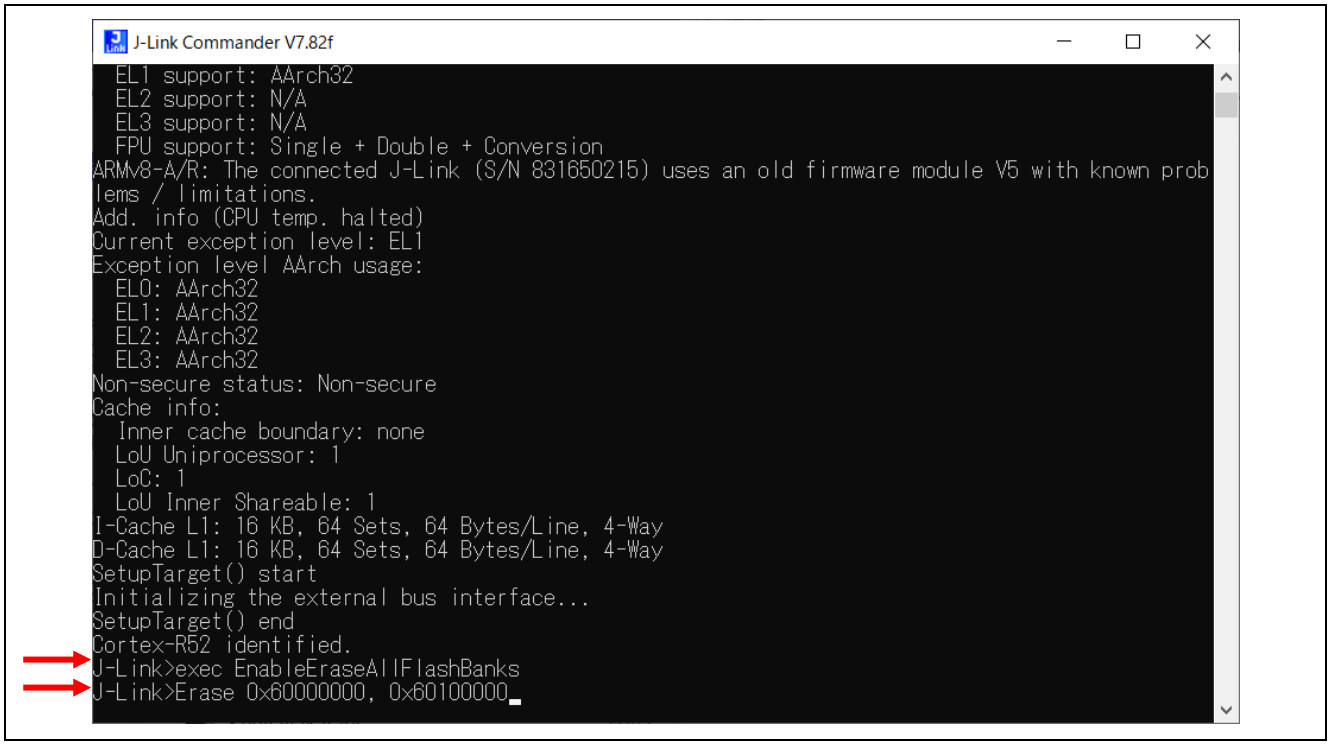


Figure 3-4 Erase flash memory (1/2)

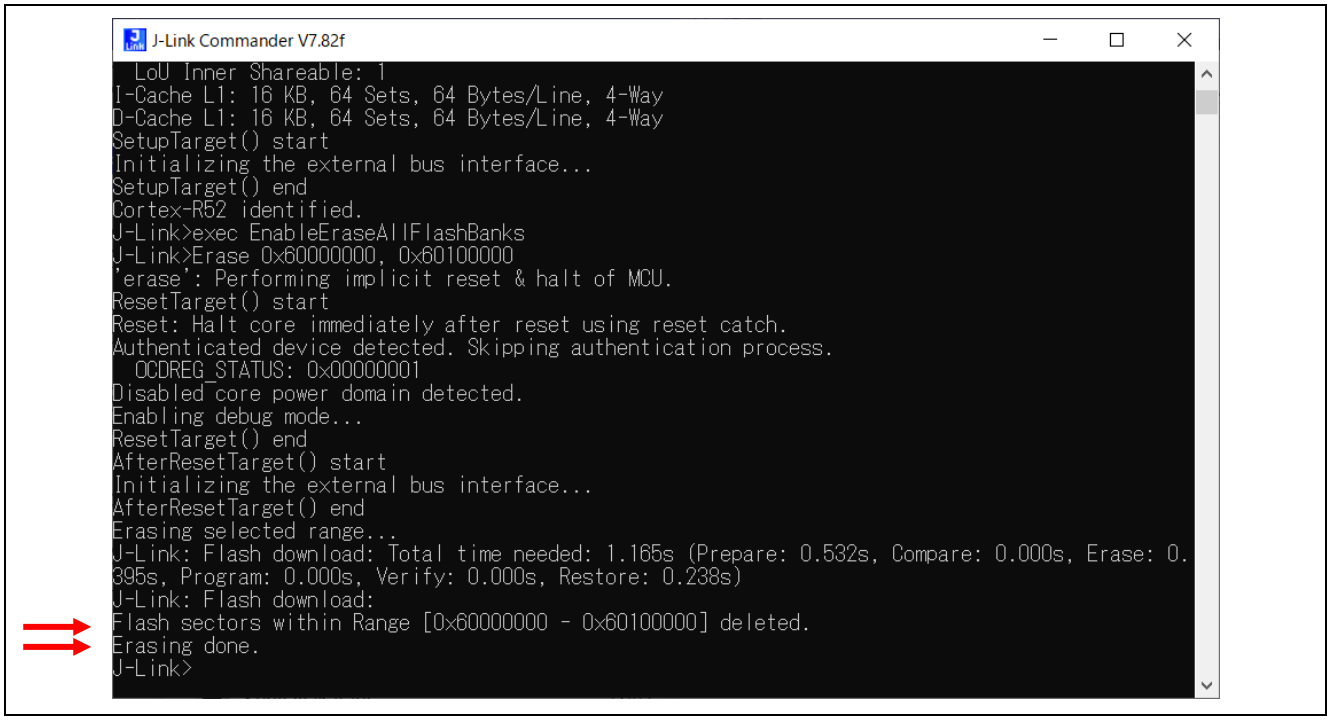


Figure 3-5 Erase flash memory (2/2)

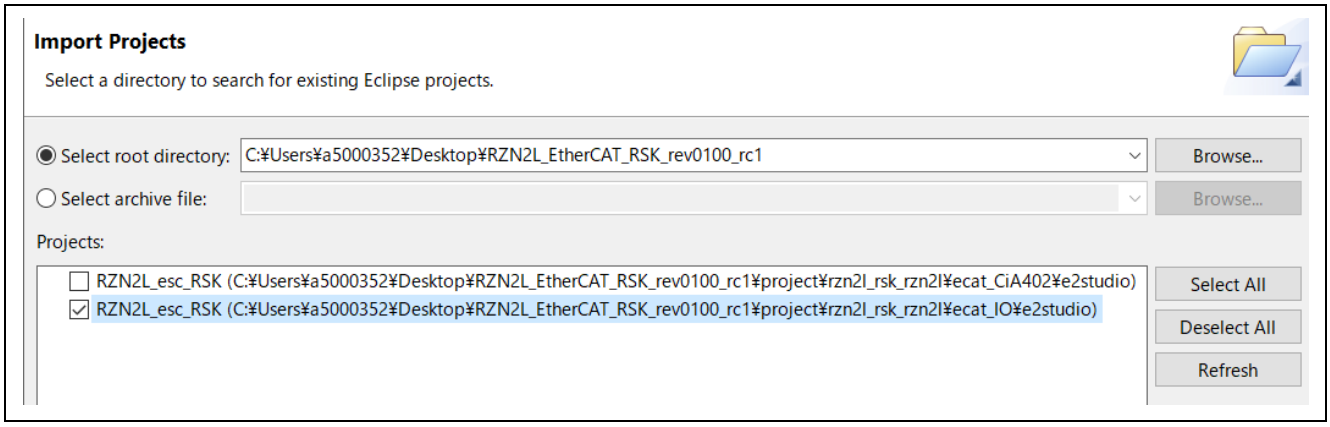
3.2.2 Project build and debug

Build the sample code and load it into RAM using Renesas Electronics e² studio.

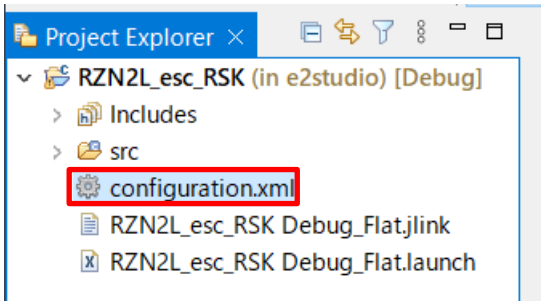
Note). Please install e2studio and adapt the FSP_Packs_v1.0.0 in advance.
Refer to the latest getting started guide.(R01an6434ejxxxx- rzt2-rzn2.pdf)

Replace the project name in the figure with the project name of this sample project.

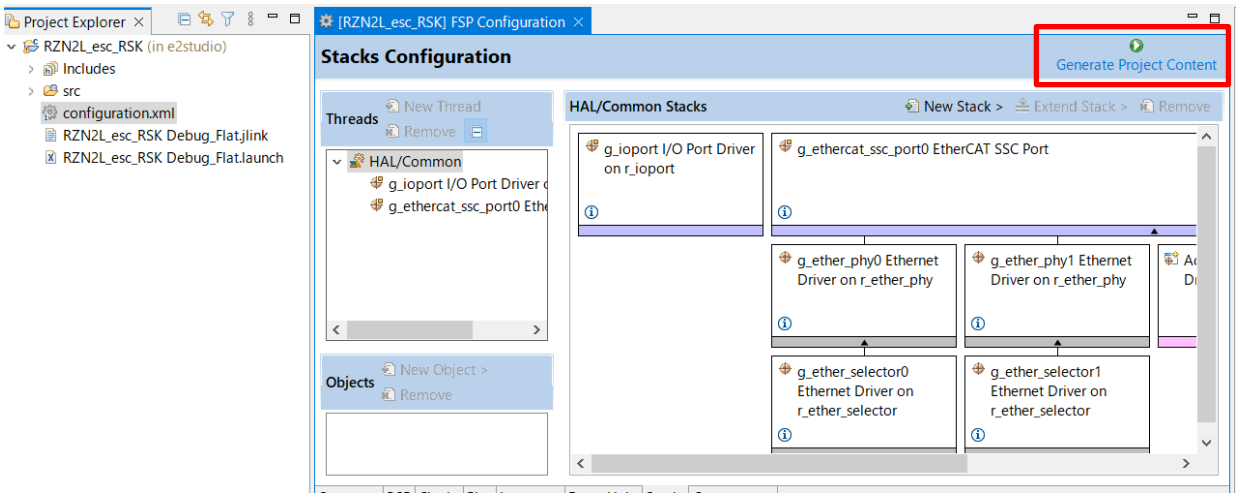
1. Import the sample project. After the program is started, by selecting [File] → [Import] → [Existing Projects into Workspace]. Check the "select root directory" and select “GPAK_IO\e2studio\RZn2L_SOM_IO” folder →[Finish].



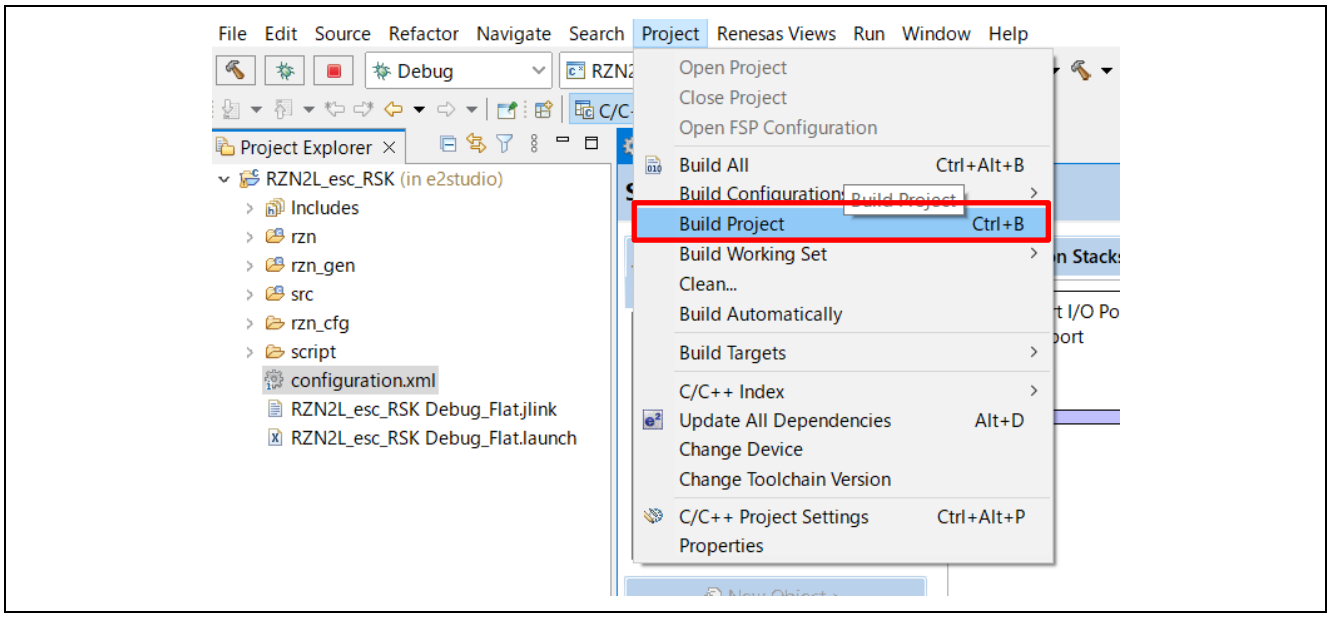
2. Open "cofiguration.xml" in the "RZN2L_SOM_IO " project



3. Generate the code with "Generate Project Content".

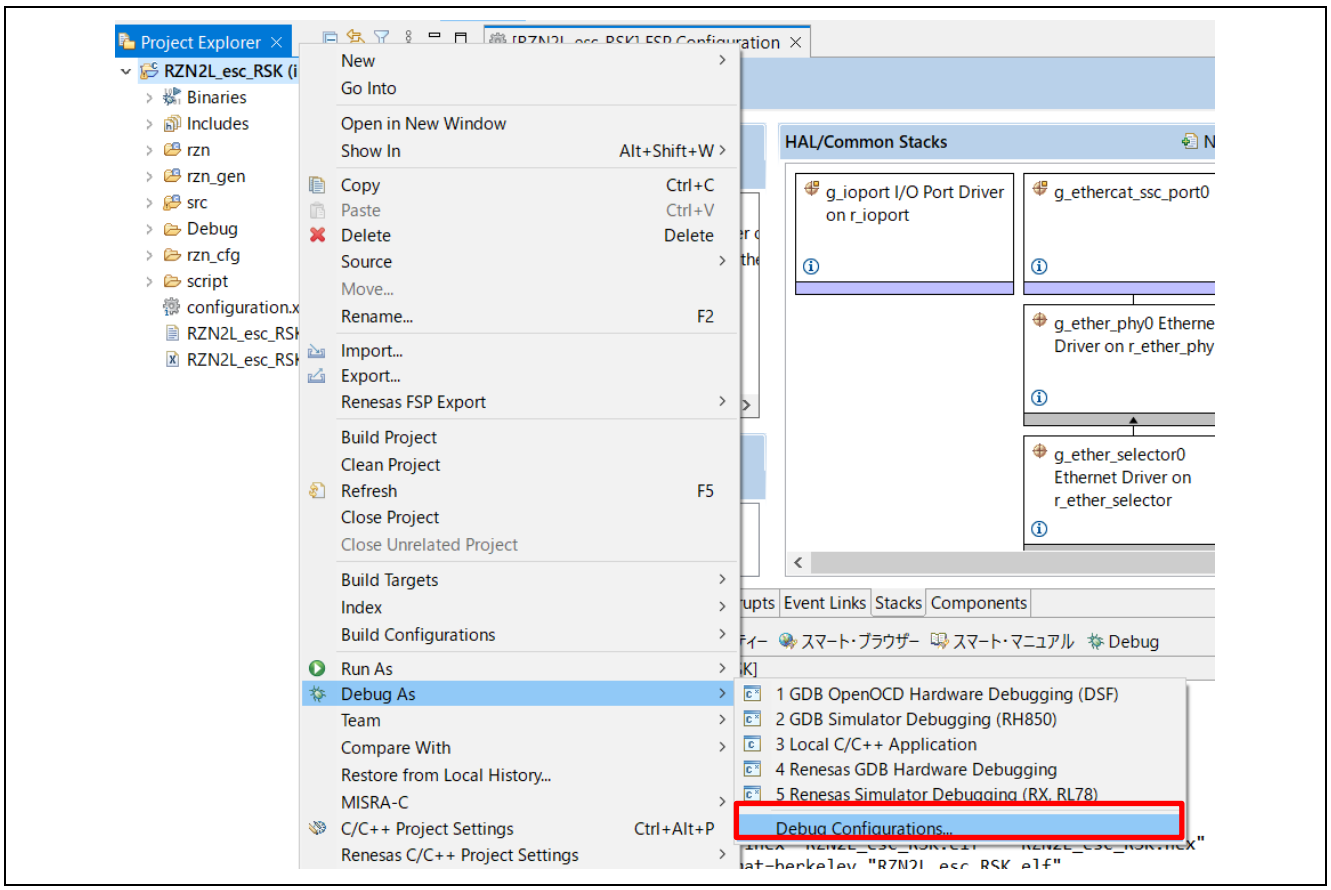


4. Select and build the "RZN2L_SOM_IO" project.

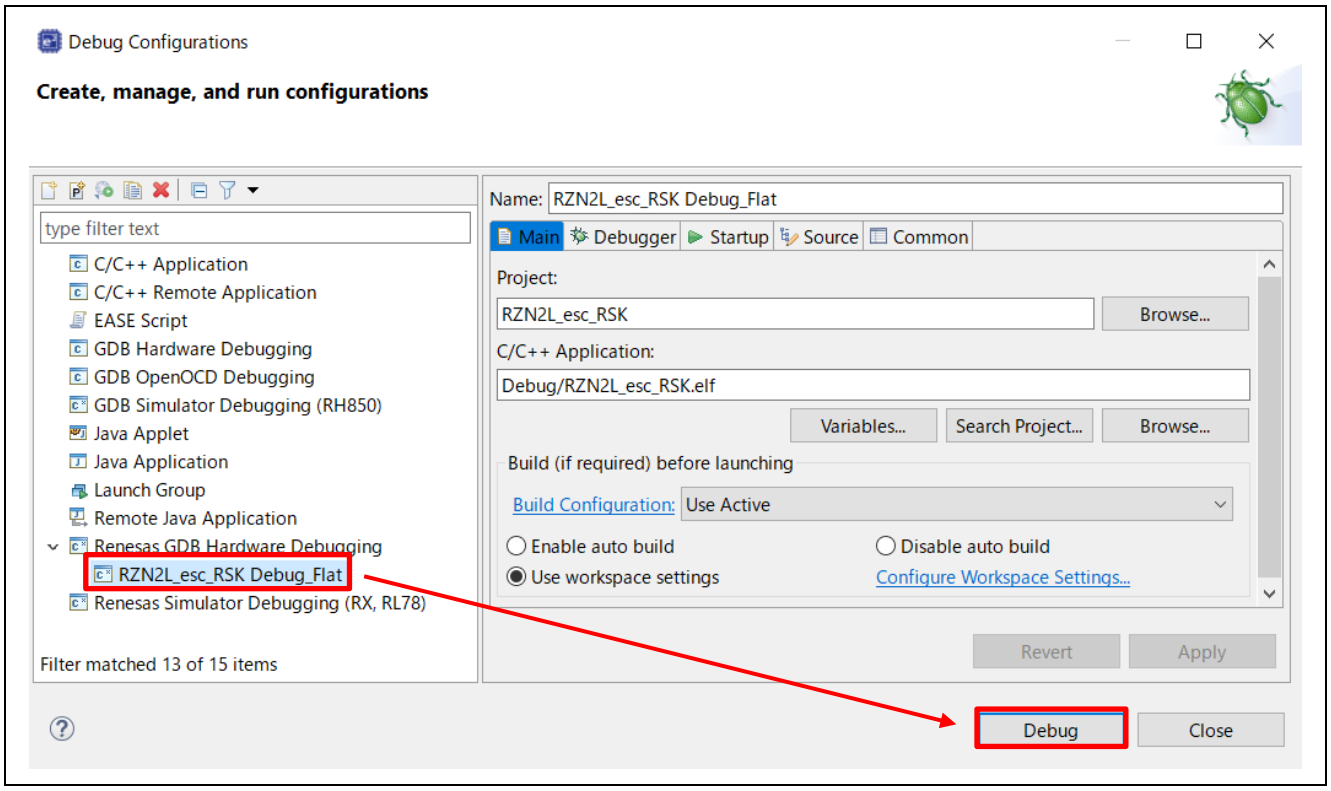


5. Press the “RESET” switch of the RZ/N2L Industrial Network SOM Kit.

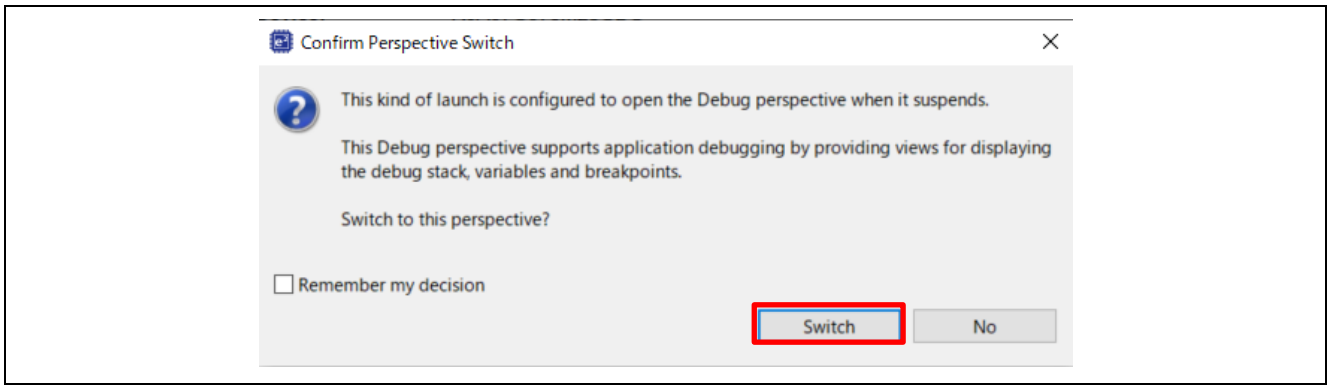
6. Connect J-Link to the SOM Kit, start debugging in the following procedure.
In [Project Explorer] view, right click the node of project to be debugged and select [Debug As] → [Debug Configurations].



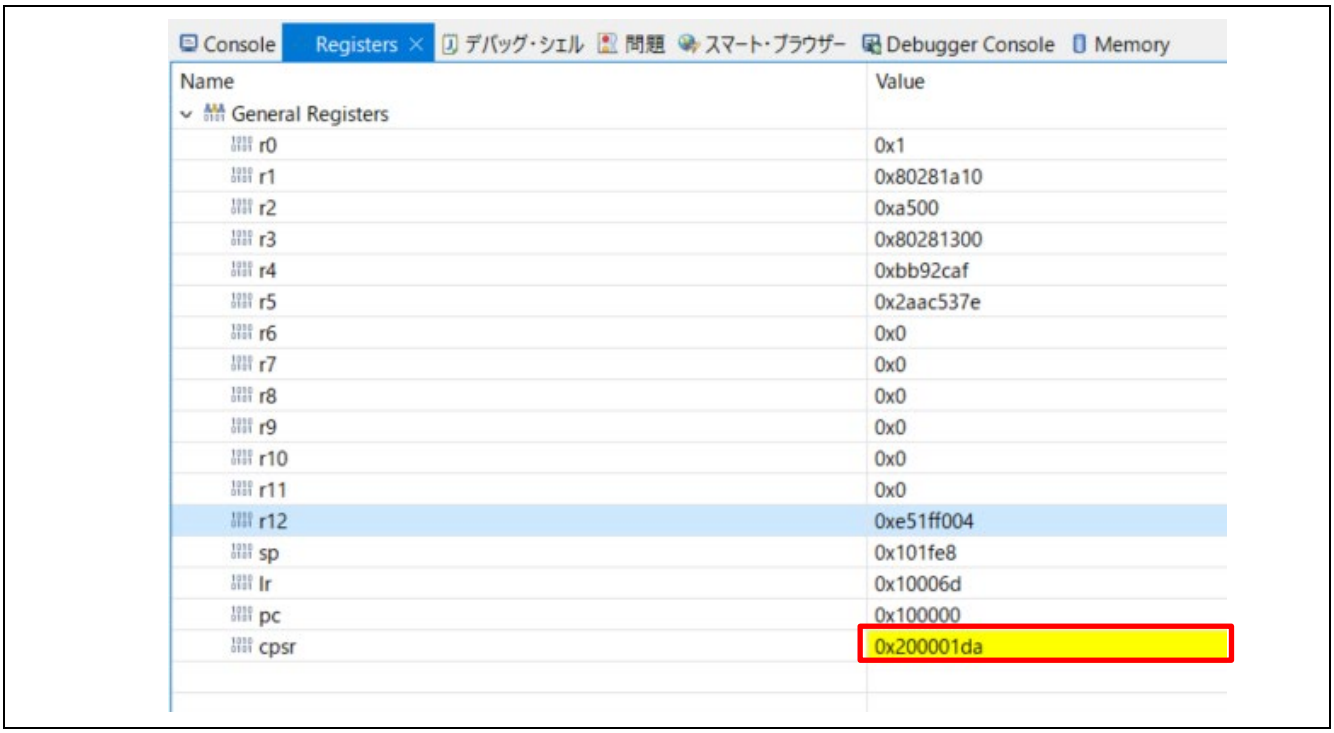
[Renesas DBG Hardware Debugging] → [RZN2L_SOM_IO Debug_Flat] item,
then press [Debug]



Following dialog will appear, so switch to the debug screen.



7. Before running the loaded program, change the CPSR register of CR52 general register on Registers tabs.
   Change the register value from “0x200001fa” to “0x200001da”.
   If the CPSR register value has not changed, program will stop at Default_Handler () at run time.



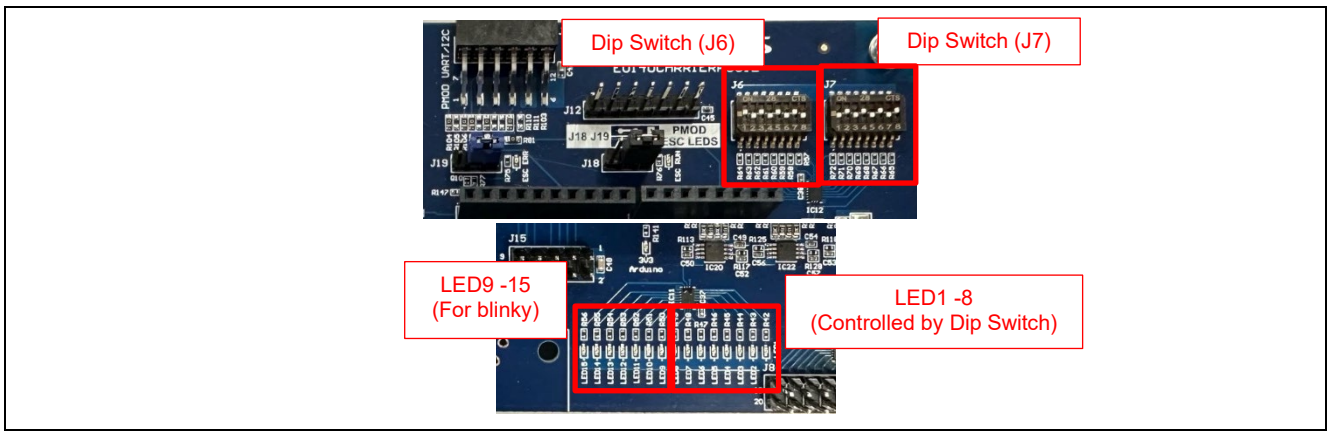
8. Press the "Resume" button for the project. Program will stop at hal_entry ().
   Press the "Resume" button again. Program is running.

4. Software

4.1 Application Operation

In the sample application, LED blinks at variable intervals, and LED control is performed by the input value of Dip Switch (J6).
Interval time is determined by Dip Switch (J7-1 ~ J7-4).

- LED1 – LED8 : LED controlled by Dip Switch.
- LED9 – LED15 : Blinking LED.
- Dip Switch(J6) : Dip Switch to control the LED.
  J6-1 corresponds to LED1 and J6-8 corresponds to LED8.
- Dip Switch(J7) : Dip Switch to control the interval time.
  Interval time is 100ms + 100ms * (4bit value of J7-1 ~ J7-4)



When the sample application is executed, LED9 - LED15 blink.
LED1 – LED8 are turned ON/OFF depending on the state of Dip Switch (J6).

For example, when J6-1 and J6-5 and J7-3 ON, LED1 and LED5 are turned on, and LED blink Interval time is set to 500ms.

4.1.1 LED blinky control

The following API is used for blinky control of LED9 – LED15.
- Gpak_LED_Read()
- Gpak_LED_Write()

In the sample application, Gpak_LED_Read() function reads the LED status, and Gpak_LED_Write() function toggles the output value.

The snippet of the application code is shown below.

```
void hal_entry(void)
{
                                        ~~ Omission ~~
    /* LED control */
    for(i=0; i<7; i++)
    {
        /* Read LED9 ~ LED15 state */
        led_state = Gpak_LED_Read(blink_led[i]);

        if(BSP_IO_LEVEL_LOW == led_state)
        {
            led_state = BSP_IO_LEVEL_HIGH;
        }
        else if (BSP_IO_LEVEL_HIGH == led_state)
        {
            led_state = BSP_IO_LEVEL_LOW;
        }

        /* Blink LED9 ~ LED15 */
        Gpak_LED_Write(blink_led[i], led_state);
    }
                                        ~~ Omission ~~
}
```

4.1.2 LED control via Dip Switch (J6)

The following API is used for LED control via Dip Switch (J6).
- Gpak_DSW_Read()
- Gpak_LED_Write()

In the sample application, Gpak_DSW_Read() functions reads the Dip Switch (J6) status. After that, determine the output value for LED from the input value and output the LED with Gpak_LED_Write() function.

The snippet of the application code is shown below.

```
void hal_entry(void)
{
                                        ~~ Omission ~~
    /* Dip switch (J6) control */
    dsw_state = Gpak_DSW_Read(DSW_CH_J6);

    for(i=0; i<8; i++)
    {
        if(DSW_STATE_OFF == ((dsw_state >> i) & 0x01))
        {
            led_output_level = BSP_IO_LEVEL_LOW;
        }
        else if(DSW_STATE_ON == ((dsw_state >> i) & 0x01))
        {
            led_output_level = BSP_IO_LEVEL_HIGH;
        }
        /* LED1 ~ LED8 control */
        Gpak_LED_Write(dsw_control_led[i], led_output_level);
    }
                                        ~~ Omission ~~
}
```

4.1.3 Interval time control via Dip Switch (J7)

The following API is used for LED control via Dip Switch (J7).
- Gpak_DSW_Read()

In the sample application, Gpak_DSW_Read() functions reads the Dip Switch (J7) status. After that, 4bit value is calculated and the interval time is determined.

The snippet of the application code is shown below.

```
void hal_entry(void)
{
                                        ~~ Omission ~~
    /* Interval control with Dip switch (J7-1 - J7-4)*/
    dsw_state = (uint8_t)~Gpak_DSW_Read(DSW_CH_J7);
    delay = (uint16_t)(DELAY_OFFSET_100 + (DELAY_OFFSET_100 * (dsw_state & 0xF)));

    /* Software delay */
    R_BSP_SoftwareDelay(delay, BSP_DELAY_UNITS_MILLISECONDS);
                                        ~~ Omission ~~
}
```

4.2 I2C Communication with GreenPAK

When using GreenPAK for I/O control, RZ/N2L communicates with GreenPAK via I2C. This sample program implements I2C communication with GreenPAK using the defined API.
In order to control GreenPAK, it is necessary to send slave address and data command. By using the API defined in this sample program, RZ/N2L sends and receives data after sending slave address and data command.

4.2.1 Slave Address

This SOM Kit equipped with three GreenPAK ICs (IC11, IC12, IC13), and by sending the corresponding slave address from the RZ/N2L, it determines which GreenPAK IC to communicate with.

The table below shows the correspondence between the slave addresses defined in the SOM kit and the definitions used in the sample software.

GreenPAK IC	Slave address	Access to	Software definition
IC11	0010b	LED1 ~ LED15	LED_OUTPUT_ADDR
IC12	0001b	Dip Switch (J6, J7)	DSW_INPUT_ADDR
IC13	0011b	Header pin (J8)	PIN_IN_OUT_ADDR

4.2.2 Data Command

After the GreenPAK IC to communicate with is determined by specifying the slave address, which register in GreenPAK to access is determined by sending the Data command. By sending and receiving data after sending a data command, it is possible to set and read register values in GreenPAK.

A list of Data commands defined in the sample software is shown below.

Command	Value	Access to
IO1_SRC_REG	0x19	IO1 Source register
IO1_OE_REG	0x1A	IO1 OE register
IO2_SRC_REG	0x1B	IO2 Source register
IO3_SRC_REG	0x1C	IO3 Source register
IO3_OE_REG	0x1D	IO3 OE register
IO4_SRC_REG	0x1E	IO4 Source register
IO5_SRC_REG	0x1F	IO5 Source register
IO5_OE_REG	0x20	IO5 OE register
IO6_SRC_REG	0x21	IO6 Source register
IO7_SRC_REG	0x22	IO7 Source register
IO8_SRC_REG	0x23	IO8 Source register
IO8_OE_REG	0x24	IO8 OE register
IO9_SRC_REG	0x25	IO9 Source register
IO10_SRC_REG	0x26	IO10 Source register
IO10_OE_REG	0x27	IO10 OE register
IO11_SRC_REG	0x28	IO11 Source register
IO11_OE_REG	0x29	IO11 OE register
IO12_SRC_REG	0x2A	IO12 Source register
IO13_SRC_REG	0x2B	IO13 Source register
IO13_OE_REG	0x2C	IO13 OE register
IO14_SRC_REG	0x2D	IO14 Source register
IO15_SRC_REG	0x2E	IO15 Source register
IO15_OE_REG	0x2F	IO15 OE register
IO16_SRC_REG	0x30	IO16 Source register
IO16_OE_REG	0x31	IO16 OE register
IO17_SRC_REG	0x32	IO17 Source register
IO0_CFG_REG	0x80	IO0 setting register
~	~	~
IO17_CFG_REG	0x91	IO17 setting register
DSW_INPUT_CTRL0	0xF0	Matrix Input(GND, IO0-IO5, IO8)

DSW_INPUT_CTRL1	0xF6	Matrix Input(IO9-IO16)
DSW_INPUT_CTRL2	0xF7	Matrix Input(IO17)
LED_CTRL_CMD	0xF4	Virtual Input
CMD_NULL	0xFFF	-

4.3 Function Specifications

This section describes the function specifications.

4.3.1 Gpak_IO_Init

Gpak_IO_Init	
Overview	API for initializing IO control.
Declaration	fsp_err_t Gpak_IO_Init (void)
Description	Initialize I2C communication module (SCI_I2C) and GreenPAK.
Arguments	None
Return value	• FSP_SUCCESS: Device opened without issue. • FSP_ERR_ALREADY_OPEN: Module is already open. • FSP_ERR_ASSERTION: Parameter check failure. • FSP_ERR_INVALID_ARGUMENT: Invalid input parameter.
Remarks	None

4.3.2 Gpak_DSW_Read

Gpak_DSW_Read	
Overview	API that returns the input value of Dip Switch.
Declaration	uint8_t Gpak_DSW_Read (uint8_t dsw_ch)
Description	Returns the Dip Switch input value of the channel specified by the argument in 8bit.
Arguments	uint8_t dsw_ch: Dip Switch channel. Use the following definition. - DSW_CH_J6 - DSW_CH_J7
Return value	8bit current input level 0: Dip Switch ON 1: Dip Switch OFF
Remarks	None

4.3.3 Gpak_LED_Read

Gpak_LED_Read	
Overview	API that returns the input value of LED.
Declaration	bsp_io_level_t Gpak_LED_Read (uint8_t led_num)
Description	Returns the LED input value of the led number specified by the argument.
Arguments	uint8_t led_num: LED number (LED1 to LED15). Use the following definition. -LED1, LED2, …, LED15
Return value	Current input level 0: BSP_IO_LEVEL_LOW 1: BSP_IO_LEVEL_HIGH
Remarks	None

4.3.4 Gpak_LED_Write

Gpak_LED_Write	
Overview	API for LED output control.
Declaration	void Gpak_LED_Write (uint8_t led_num, bsp_io_level_t level)
Description	Controls the LED output value of the led number specified by the argument.
Arguments	uint8_t led_num: LED number (LED1 to LED15). Use the following definition. -LED1, LED2, …, LED15 bsp_io_level_t level: Output level. Use the following definition. - BSP_IO_LEVEL_LOW - BSP_IO_LEVEL_HIGH
Return value	None
Remarks	None

4.3.5 led_Init

led_init	
Overview	Initialize GreenPAK module (IC11) for LED1 – LED15 output.
Declaration	void led_init (void)
Description	Initialize GreenPAK module (IC11) for LED1- LED15 output. After initialization, LED 6 is high level output and the others are low outputs.
Arguments	None
Return value	None
Remarks	None

4.3.6 dsw_Init

dsw_Init	
Overview	Initialize GreenPAK module (IC12) for Dip Switch (J6, J7) input.
Declaration	void dsw_init (void)
Description	Initialize GreenPAK module (IC12) for Dip Switch (J6, J7) input.
Arguments	None
Return value	None
Remarks	None

4.3.7 iic_SendData

iic_SendData	
Overview	Function for I2C communication with GreenPAK.
Declaration	fsp_err_t iic_SendData (uint8_t *value, e_i2c_addr_num_t addr, uint8_t data_num, e_i2c_cmd_num_t cmd)
Description	Data is sent after sending a data command to access GreenPAK via I2C communication.
Arguments	• uint8_t *value: Send Data. • e_i2c_addr_num_t addr: Slave address. • uint8_t data_num: Number of the data to send. • e_i2c_cmd_num_t cmd: Data command for communicating with GreenPAK.
Return value	• FSP_SUCCESS: Function executed without issue. • FSP_ERR_ASSERTION: p_api_ctrl, p_src, p_callback is NULL. • FSP_ERR_NOT_OPEN: Device was not even opened. • FSP_ERR_ABORTED: Operation was aborted.
Remarks	None

4.3.8 iic_ReceiveData

iic_ReceiveData	
Overview	Function for I2C communication with GreenPAK.
Declaration	fsp_err_t iic_ReceiveData (uint8_t *value, e_i2c_addr_num_t addr, uint8_t data_num, e_i2c_cmd_num_t cmd)
Description	Data is received after sending a data command to access GreenPAK via I2C communication.
Arguments	• uint8_t *value: recieved Data. • e_i2c_addr_num_t addr: Slave address. • uint8_t data_num: Number of the data to send. • e_i2c_cmd_num_t cmd: Data command for communicating with GreenPAK.
Return value	• FSP_SUCCESS: Function executed without issue. • FSP_ERR_ASSERTION: p_api_ctrl, p_src, p_callback is NULL. • FSP_ERR_NOT_OPEN: Device was not even opened. • FSP_ERR_ABORTED: Operation was aborted.
Remarks	None

4.4 Notes on the sample program

4.4.1 About SCI_I2C module

This sample program uses simple I2C mode of SCI to communicate with GreenPAK.
RZ/N2L FSP v1.1.0 does not support SCI_I2C driver, so SCI_I2Cdriver is placed under the src folder independently from FSP.

Revision History

Rev.	Date	Description	
		Page	Summary
1.00	Feb 7, 2023	-	First edition issued
1.10	Aug 7, 2023	-	Support RZ/N2L FSP v1.2.0

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

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

1. 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/.