# RZ/A3UL

## Initial Program Loader Application Note (Serial Flash Boot Edition)

### Introduction

This application note describes the functions of Initial Program Loader (called IPL), which is a boot loader for booting user programs from serial flash memory connected to RZ/A3UL, the procedure for building the development environment and the debugging method of it.

### Target Device

RZ/A3UL

When using this application note with other Renesas Microprocessors, sufficient evaluation is recommended after making modifications to comply with the alternate Microprocessor.

## Contents

1. Overview ....4
1.1 Glossary .... 4
1.2 IPL processing overview .... 5
2. Operating Environment ....6
3. Reference Application Notes ....7
4. Hardware Description ....8
4.1 Peripheral Functions used .... 8
4.2 Pins Used List .... 8
5. Software ....9
5.1 Operation overview .... 9
5.1.1 Operation image of boot process .... 9
5.1.2 Boot process to launch FSP application on serial flash memory .... 11
5.2 Peripheral Function Setting .... 13
5.2.1 Settings of SPIBSC Regsiter .... 15
5.3 Memory Map .... 17
5.4 MMU setting .... 18
5.5 Interrupt used list .... 19
5.6 Exception level .... 19
5.7 Application Header .... 19
5.8 Flow Chart .... 20
5.8.1 IPL (overall) flow .... 20
5.8.2 FSP Application Image copy process .... 23
5.8.3 Setting of Serial Flash Memory and SPIBSC .... 25
6. IPL development environment construction procedure ....28
6.1 Procedure for building a development environment for Widows10 PC .... 28
6.1.1 Installation of IDE (e² studio) for Windows10 .... 28
6.1.1.1 e² studio installation (Windows10) .... 28
6.1.1.2 Toolchain installation (Windows10) .... 35
6.1.1.3 Toolchain registration (Windows10) .... 35
6.1.1.4 Installation of RZ/A FSP packs (Windows10) .... 37
6.1.2 MSYS2 Installation .... 38
6.2 Procedure for building a development environment for Ubuntu 20.04 LTS .... 41
6.2.1 Installation of IDE (e² studio) for Ubuntu 20.4 LTS .... 41
6.2.1.1 e² studio installation (Ubuntu 20.4 LTS) .... 41
6.2.1.2 NoteToolchain installation (Ubuntu20.4 LTS) .... 48
6.2.1.3 Toolchain registration (Ubuntu 20.4 LTS) .... 48
6.2.1.4 Installation of RZ/A FSP packs (ubuntu 20.4 LTS) .... 50

6.2.2 Additional Software Installation ..... 51

7. IPL build environment construction procedure ..... 52

8. Execution Environment of IPL ..... 62

8.1 Create Debug Configuration ..... 62

8.2 Connection to SMARC Board ..... 70

8.3 Setting Serial Terminal ..... 72

8.4 Execution procedure of IPL by Serial Flash boot ..... 76

9. Application Example ..... 81

9.1 Execution procedure of IPL by Internal SRAM download ..... 81

9.2 Procedure for writing an FSP application to serial flash memory ..... 85

10. Precautions ..... 91

10.1 About downloading and executing IPL RAM to On-chip RAM ..... 91

10.2 About DDR SDRAM access prohibited area ..... 91

11. Reference Documents ..... 92

Revision History ..... 93

# 1. Overview

IPL is a boot program for RZ/A3UL.

When starting the RZ/A3UL device, IPL initializes the peripheral I/O modules, the memory installed on the board, etc., and launch the FSP application.

IPL consists of functions such as peripheral I/O module driver, memory control driver, and FSP application image load module.

## 1.1 Glossary

Table 1.1 shows the terms related to the operation of IPL explained in this application note.

**Table 1.1 The term for IPL operation**

| Term | Description |
|---|---|
| Initial Program Loader (IPL) | A loader program that launches an FSP Application for the RZ/A3UL environment.<br>It is called from the BootROM program to initialize the memory and the peripheral I/O modules installed in boards, then copies the FSP application and jumps to the copy destination according to the application header.<br>IPL is developed based on Arm Trusted Firmware and it is equivalent to BL2 of Arm Trusted Firmware.<br>When changing the serial flash memory to be used, the user is requested to create and add the initialization code according to the specifications of the serial flash memory to be used, referring to this application note.<br>By default, optimal settings are made for using the Renesas Electronics serial flash memory(AT25QL128A). |
| BootROM Program | The BootROM program is the program that executes first after the device starts, and it copies the next boot program from the memory associated with the boot mode to the On-chip RAM and jumps to the next boot program.<br>IPL is premised on operation in Boot Mode 3 (1.8-V serial Flash memory), and the IPL written to the serial flash memory is copied to the specified address (H'0001_2000) of the On-chip RAM and the boot process is executed.<br>This program is stored in the RZ/A3UL's internal ROM, so it does not need to be created by the user. |
| FSP Application | This is a program consisting of BSP and sample programs to use the functions implemented on the board. |
| Application Header | A header placed at the beginning of the FSP application, used by the IPL to refer to the destination address or entry point address of the FSP application.<br>Before executing IPL, the FSP application image including the application header must be written to the specified address of the serial flash memory.<br>The write destination address of the Application Header can be changed in the IPL build option, and the default is H'2002_0000. |
| External Address Space Read Mode | This is the operation mode of the SPI Multi I/O Bus Controller.<br>Using this mode, the user can access the serial flash memory connected to the external address space as if it were memory mapped to the RZ/A3UL's address space. |

## 1.2 IPL processing overview

The outline of IPL processing is described below.

1. Release the reset and supply the clock to the module used for the boot process.
2. Initialize the serial communication interface for debug message output and control the message input/output.
3. Change the settings of serial flash memory controller.
4. Initialize DDR SDRAM
5. Copy the FSP Application image from serial flash memory to DDR SDRAM.
6. Jump to the entry point of FSP Application on DDR SDRAM.

## 2. Operating Environment

IPL runs in the following environment

**Table 2.1 Operating Environment**

| Item | Contents |
|---|---|
| MPU | RZ/A3UL |
| Operating Frequency | CPU clock (Iφ): 1.0GHz<br>SRAM MCPU clock (I2φ): 200MHz<br>Peripheral clock 0 (P0φ): 100MHz<br>Peripheral clock 1 / SRAM ACPU clock (P1φ): 200MHz<br>SPI multi I/O main clock (SPI0φ): 66.6MHz |
| Operating Voltage | Power supply voltage (I/O): 3.3V<br>Power supply voltage (1.8/3.3V switching I/O (SPI_PVDD)):1.8V |
| Host PC | Windows10 (64bit) installed PC<br>or<br>Ubuntu 20.04 LTS Desktop (64bit) installed PC |
| Debugger | J-Link Base 11.0 |
| IDE | e² studio 2022-07 |
| Toolchain | Windows10 PC: gcc-arm-10.3-2021.07-mingw-w64-i686-aarch64-none-elf<br>Ubuntu 20.4 LTS PC: gcc-arm-10.3-2021.07-x86_64-aarch64-none-elf |
| | Compiler option (Excludes the addition of directory paths<br><br>Release configuration:<br>-march=armv8-a -mgeneral-regs-only -mstrict-align -ffunction-sections<br>-fdata-sections -ffreestanding -fno-builtin -fno-common -Os -std=gnu99<br>-fno-stack-protector -Werror -Wall -Wmissing-include-dirs -Wunused<br>-Wdisabled-optimization -Wvla -Wshadow -Wno-unused-parameter<br>-Wredundant-decls -Wunused-but-set-variable -Wmaybe-uninitialized<br>-Wpacked-bitfield-compat -Wshift-overflow=2 -Wlogical-op<br>-Wno-error=deprecated-declarations -Wno-error=cpp<br><br>Debug configuration:<br>-march=armv8-a -mgeneral-regs-only -mstrict-align -ffunction-sections<br>-fdata-sections -ffreestanding -fno-builtin -fno-common -Os -std=gnu99<br>-fno-omit-frame-pointer -fno-stack-protector -Werror -Wall<br>-Wmissing-include-dirs -Wunused -Wdisabled-optimization -Wvla -Wshadow<br>-Wno-unused-parameter -Wredundant-decls -Wunused-but-set-variable<br>-Wmaybe-uninitialized -Wpacked-bitfield-compat -Wshift-overflow=2<br>-Wlogical-op -Wno-error=deprecated-declarations -Wno-error=cpp |
| Operating Mode | Boot Mode3 (1.8-V serial Flash memory) |
| Board | SMARC board Kit for RZ/A3UL MPU QSPI Edition<br>(In this document, it is called SMARC board) |
| Device to use<br>(Function used on the board) | ● Serial Flash Memory (Store the image of the program)<br>Manufacture's name: Renesas Electronics<br>Model Name: AT25QL128A<br>● DDR4 SDRAM (Used as an execution area for FSP Application)<br>Manufacture's name: Micron Technology<br>Model Name: MT40A512M16LY-062E |

## 3. Reference Application Notes

Application notes related to this application note are currently in preparation.

# 4. Hardware Description

This chapter describes the hardware used by IPL.

## 4.1 Peripheral Functions used

IPL uses the peripheral features shown in Table 4-1.

**Table 4.1 Peripheral functions and uses**

| Peripheral Function | Use |
|---|---|
| SPI multi-I/O bus controller (SPIBSC) | Set to external space read mode and generate a signal for the CPU to read directly from the serial flash memory connected to the SPI multi-I/O bus space. |
| DDR3L/DDR4 SDRAM Memory Controller (MEMC) | Make settings to access DDR3L/DDR4 SDRAM connected to the external bus space. |
| Clock pulse generator (CPG) | Generate the operating frequency of the RZ/A3UL.<br>Release the module standby mode of the RZ/A3UL's peripheral I/O modules. |
| General Purpose Input Output Port (GPIO) | Switch multiplexed pin functions for SCIFA channel 0 and set the QSPI dedicated pins. |
| Serial communication interface with FIFO (SCIFA) | Used for message output to the host PC using SCIFA channel 0. |
| System Counter (SYC) | Generate the count value used by the generic timer built into the Cortex-A55 core. |
| Memory Management Unit (MMU) | Generate a translation table that specifies the cache effective area and memory type for the RZ/A3UL address space. |

## 4.2 Pins Used List

Table 4-2 lists the pins used and their functions.

**Table 4.2 Pins used and their functions**

| Pin name | I/O | Function |
|---|---|---|
| QSPI0_SPCLK | Output | Serial Flash memory clock |
| QSPI0_IO0 | Input | Serial Flash data0 |
| QSPI0_IO1 | Input | Serial Flash data1 |
| QSPI0_IO2 | Input | Serial Flash data2 |
| QSPI0_IO3 | Input | Serial Flash data3 |
| QSPI0_SSL | Output | Serial Flash memory slave select |
| QSPI_RESET# | Output | Serial Flash memory reset |
| QSPI_WP# | Output | Serial Flash memory write protect |
| SCIF0_SCK | Input/Output | Serial communication interface clock |
| SCIF0_RXD | Input/Output | Serial communication receive data signal |
| SCIF0_TXD | Output | Serial communication transmit data signal |

# 5. Software

This chapter describes the IPL software configuration and functions.

## 5.1 Operation overview

When the SMARC board is booted in Boot Mode 3 (1.8-V serial Flash memory), the Boot ROM program written in the internal ROM will start.

The IPL is copied from the serial flash memory located in the SPI multi I/O bus space to the On-chip RAM by the BootROM program and booted.

IPL gets the copy destination address and entry point address of the FSP application image from the application header placed at the beginning of the FSP application.

After that, copy the FSP application to the obtained address and jump to the entry point address.
Please refer to 5.7 Application Header for application header configuration.

### 5.1.1 Operation image of boot process

Figure 5.1 shows an operation image of the basic boot process.

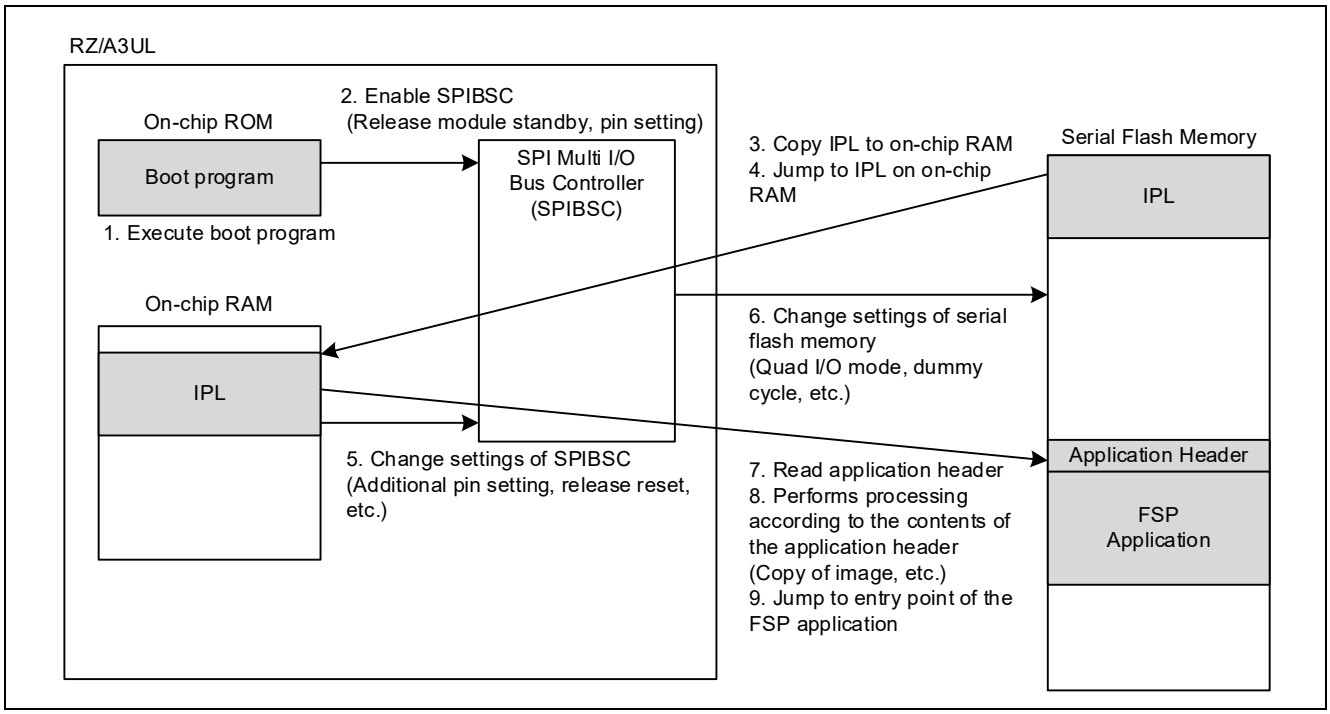


**Figure 5.1 Operation image of boot process**

Describes the IPL boot process.

1. RZ/A3UL executes the BootROM program after releasing the power-on reset.
2. When the RZ/A3UL operating mode is Boot mode 3, the BootROM program will enable SPIBSC and initialize it with a setting that can be accessed independently of the model of the serial flash memory.
3. The BootROM program reads the IPL stored in the start address (*1) of the serial flash memory and copies the IPL to the entry point address (*2) on the On-chip RAM.
   At this time, once SPIBSC is reset, the access to the serial flash memory will be prohibited.
4. The BootROM program jumps to the IPL entry point (* 2) on the On-chip RAM.
5. IPL makes additional pin settings and resets for SPIBSC for 4-byte access to the serial flash memory, and changes the settings to the optimum settings for the serial flash memory (AT25QL128A) mounted on the SMARC board.
6. A command is issued from SPIBSC to the serial flash memory to change the settings of the serial flash memory.
7. IPL reads the application header.
8. IPL performs processing such as copying the FSP application according to the contents of the application header.
   As an example of the process is shown in the following chapter.
   The case of booting as it is on the serial flash memory without copying is shown in
   5.1.2 Operation image of boot process.
9. The IPL follows the contents of the application header and jumps to the entry point of the FSP application.

   (*1) This address will be H'2000_0000 when the external address space read mode is set.
   (*2) The IPL entry point address is H'0001_2000.

### 5.1.2 Boot process to launch FSP application on serial flash memory

Figure 5.3 shows an operation image when jumping directly to the FSP application written in the serial flash memory and executing it (XIP execution).

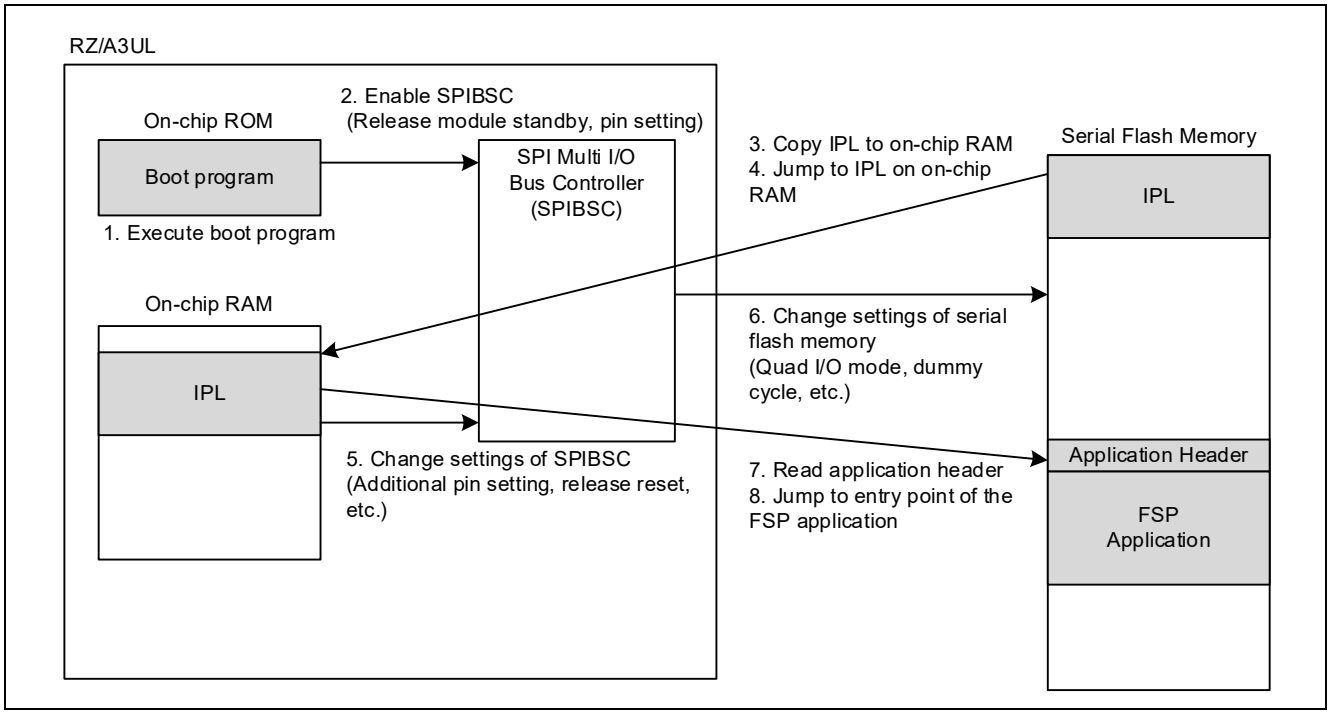


**Figure 5.2 Operation image when booting FSP application on serial flash memory**

Describes the IPL boot process.

1. RZ/A3UL executes the BootROM program after releasing the power-on reset.
2. When the RZ/A3UL operating mode is Boot mode 3, the BootROM program will enable SPIBSC and initialize it with a setting that can be accessed independently of the model of the serial flash memory.
3. The BootROM program reads the IPL stored in the start address of the serial flash memory and copies the IPL to the entry point address on the On-chip RAM.
   At this time, once SPIBSC is reset, the access to the serial flash memory will be prohibited.
4. BootROM will jump to the entry point of IPL on the On-chip RAM.
5. IPL makes additional pin settings and resets for SPIBSC for 4-byte access to the serial flash memory, and changes the settings to the optimum settings for the serial flash memory (AT25QL128A) mounted on the SMARC board.
6. A command is issued from SPIBSC to the serial flash memory to change the settings of the serial flash memory.
7. IPL reads the application header. At this time, if the destination address of the FSP application is the same as the stored address, IPL will skip the copy process.
8. The IPL will jump to the entry point of the FSP application in serial flash memory according to the contents of the application header.

## 5.2 Peripheral Function Setting

Table 5.1 Lists the settings for Peripheral Functions during execution of IPL.

**Table 5.1 Settings for Peripheral Functions**

| Module | Settings |
|---|---|
| CPG | Set the following operating frequencies.<br>● CPU clock (Iφ): 1.0GHz<br>Set to 1/1 times the frequency of the PLL1 (FOUT)<br>● Peripheral clock 0 (P0φ): 100MHz<br>Set to 1/8 times the frequency of the PLL2 (FOUT1PH0)<br>● Peripheral clock 1 / SRAM ACPU clock (P1φ): 200MHz<br>Set to 1/8 times the frequency of the PLL3 (FOUT1PH0)<br>● SRAM MCPU clock (I2φ): 200MHz<br>Set to 1/8 times the frequency of the PLL3 (FOUT1PH0)<br>● SPI multi I/O main clock (SPI0φ): 133.25MHz<br>Set to 1/8 times the frequency of the PLL3 (FOUT3)<br>● QSPI0_SPCLK: 66.625MHz<br>1/2 times of SPI0φ<br><br>Supply clock and release reset to the peripherals shown below Release the reset<br>● AXI Bus<br>● Peripheral Bus<br>● On-chip RAM<br>● Internal ROM<br>● DDR SDRAM<br>● OTP<br>● GIC600<br>● IA55<br>● SYSC<br>● SYC<br>● DMAC<br>● WDT<br>● GPIO<br>● SPIBSC<br>● SCIFA (ch0 only)<br>● Coresight |
| SPIBSC | • Command-Address-Data bits<br>1-4-4<br>• Opcode<br>“Fast Read Quad I/O” EBh<br>• Address size<br>3-byte<br>• Additional data<br>1-byte 0x55<br>• Dummy cycles<br>6 QSPI0_SPCLK<br>• SLCH<br>1.5 QSPI0_SPCLK (22.5ns)<br>• CLSH<br>1 QSPI0_SPCLK (15ns)<br>• SHSL<br>7 QSPI0_SPCLK (105ns) |

| | |
|---|---|
| GPIO | Set the following shared pin<br>● P6_3 : SCIF0_RXD<br>● P6_4 : SCIF0_TXD<br><br>Set the following in P6_3 and P6_4<br>● Drive capacity: 4mA<br>● Slew rate: Fast<br>● Pull up: None<br>● Pull down: None<br><br>Set the following for the decided pins of QSPI0_IO0, QSPI0_IO1, QSPI0_IO2, QSPI0_IO3, QSPI0_SPCLK, QSPI0_SSL<br>● Output impedance: 66Ω<br>● Slew rate: Fast<br>● Pull up: None<br>● Pull down: None |
| SCIFA | Set the following for Ch0<br>● Operation mode: asynchronous communication mode<br>● Transfer rate: 115200bps<br>● Data length: 8bit<br>● Stop bit length: 1 bit<br>● Parity: None<br>● Flow control: disabled |
| SYC | Input 24MHz clock to Cortex-A55 built-in Generic Timer |

### 5.2.1 Settings of SPIBSC Regsiter

Table 5.2 shows the SPIBSC register settings after the initial settings are completed.

These setting values include the results that reflect the settings described in the SPIBSC section in Table 5.1 Peripheral Function Settings.

**Table 5.2 SPIBSC register setting value after initial setting**

| Register | Settings |
|---|---|
| IOn output level while idle | |
| CMNCR.MOIIO0 | B'11 (Hi-Z) |
| CMNCR.MOIIO1 | B'11 (Hi-Z) |
| CMNCR.MOIIO2 | B'11 (Hi-Z) |
| CMNCR.MOIIO3 | B'11 (Hi-Z) |
| IOn output level while 1 bit transfer | |
| CMNCR.IO0FV | B'11 (Hi-Z) |
| CMNCR.IO2FV | B'01 (High) |
| CMNCR.IO3FV | B'01 (High) |
| Data bus size | |
| CMNCR.BSZ | B'00 (Single serial flash memory connected) |
| Clock delay | |
| SSLDR.SCKDL | B'000 (1.5 QSPI0_SPCLK) |
| SSL negation delay | |
| SSLDR.SLNDL | B'000 (1 QSPI0_SPCLK) |
| Next access delay | |
| SSLDR.SPNDL | B'110 (7 QSPI0_SPCLK) |
| SSL negation | |
| DRCR.SSLE | B'1 (Negated when the accessed address is not in sequential) |
| Read burst | |
| DRCR.RBE | B'1 (Burst enable) |
| Burst length | |
| DRCR.RBURST | B'00111 (8 data unit: 64 byte) |
| Command | |
| DRCMR.CMD | 0xEB (Fast Read Quad I/O operation for AT25QL128A) |
| Optional command | |
| DRCMR.OCMD | 0x00 (Not used) |
| Extended external address valid range | |
| DREAR.EAC | B'000 |
| Extended upper address value | |
| DREAR.EAV | 0x00 |
| Option data | |
| DROPR.OPD3 | 0x55 |
| DROPR.OPD2 | 0x00 |
| DROPR.OPD1 | 0x00 |
| DROPR.OPD0 | 0x00 |

| | |
|---|---|
| Option data enable<br>DRENR.OPDE | B’1000 (Outs OPD3) |
| Address enable<br>DRENR.ADE | B’0111 (Outs address [23:0]) |
| Optional command enable<br>DRENR.OCDE | B’0 (Suppress OCMD) |
| Command enable<br>DRENR.CDE | B’1 (Outs CMD) |
| Dummy cycle enable<br>DRENR.DME | B’1 (Has dummy cycle) |
| Data read bit size<br>DRENR.DRDB | B’10 (4 bit) |
| Option data bit size<br>DRENR.OPDB | B’10 (4 bit) |
| Address bit size<br>DRENR.ADB | B’10 (4 bit) |
| Optional command bit size<br>DRENR.OCDB | B’00 (Not used) |
| Command bit size<br>DRENR.CDB | B’00 (1 bit) |
| Amount of dummy cycle<br>DRDMCR.DMCYC | B’00011 (4 cycles) |
| Address DDR enable<br>DRDRENR.ADDRE | B’0 (SDR) |
| Option data DDR enable<br>DRDRENR.OPDRE | B’0 (SDR) |
| Data read DDR enable<br>DRDRENR.DRDRE | B’0 (SDR) |
| Device selection<br>PHYCNT.PHYMEM | B’00 (SDR) |
| Clock timing switching<br>PHYCNT.CKSEL | B’11 |
| High-speed response mode<br>PHYCNT.HS | B’0 (Disable) |
| DDR operation timing<br>PHYOFFSET1.DDRTMG | B’11 (SDR) |
| Serial flash memory operation timing<br>PHYOFFSET2.OCTTMG | B’100 (Using serial flash memory) |

## 5.3 Memory Map

Figure 5.4 shows the RZ/A3UL address space and the memory map of the SMARC board.

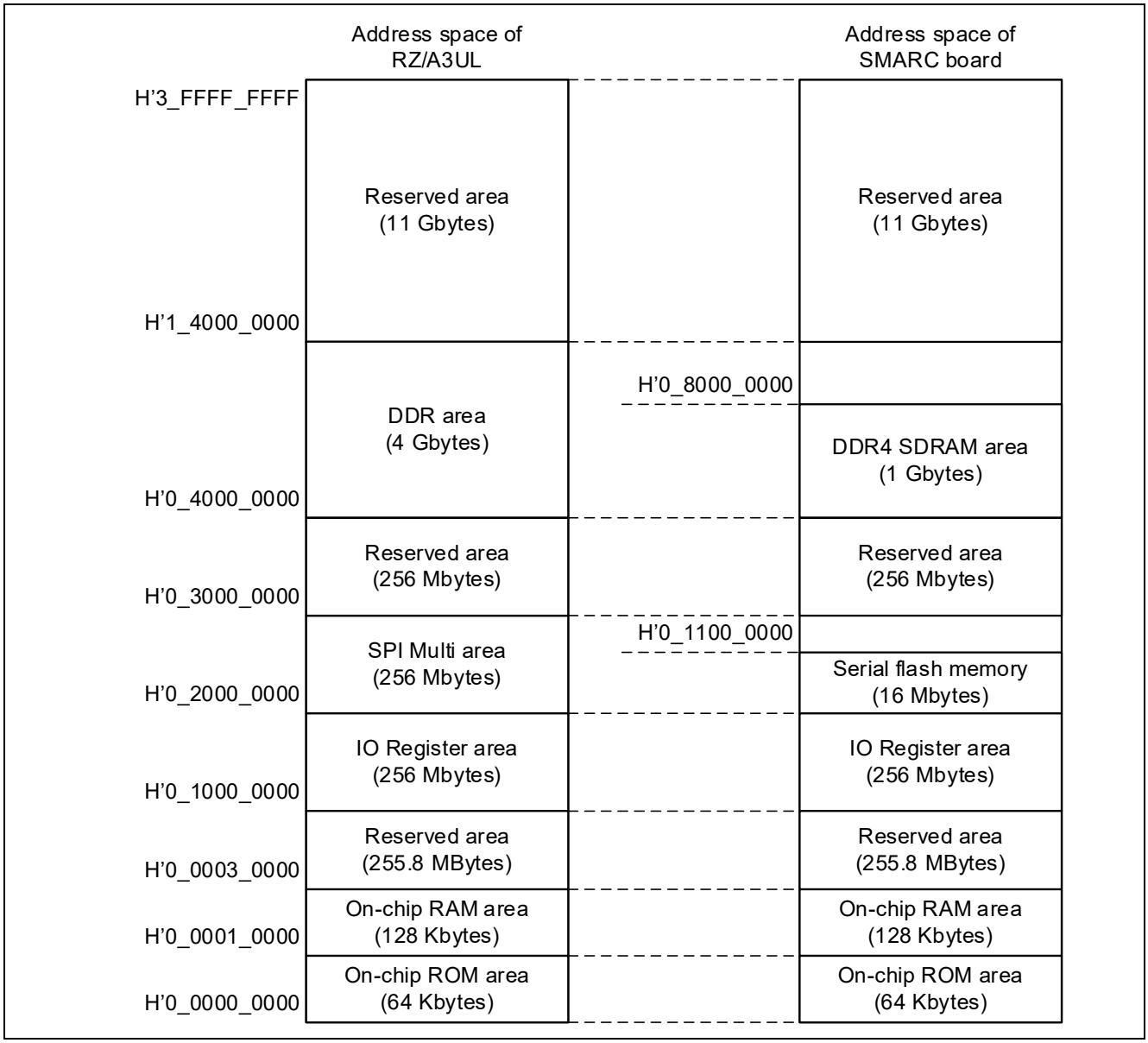


**Figure 5.3 Memory MAP**

## 5.4 MMU setting

Figure 5.5 shows the IPL MMU settings.

The MMU is disabled when jumping to the FSP Application.

Make the optimum settings in the FSP application if necessary.

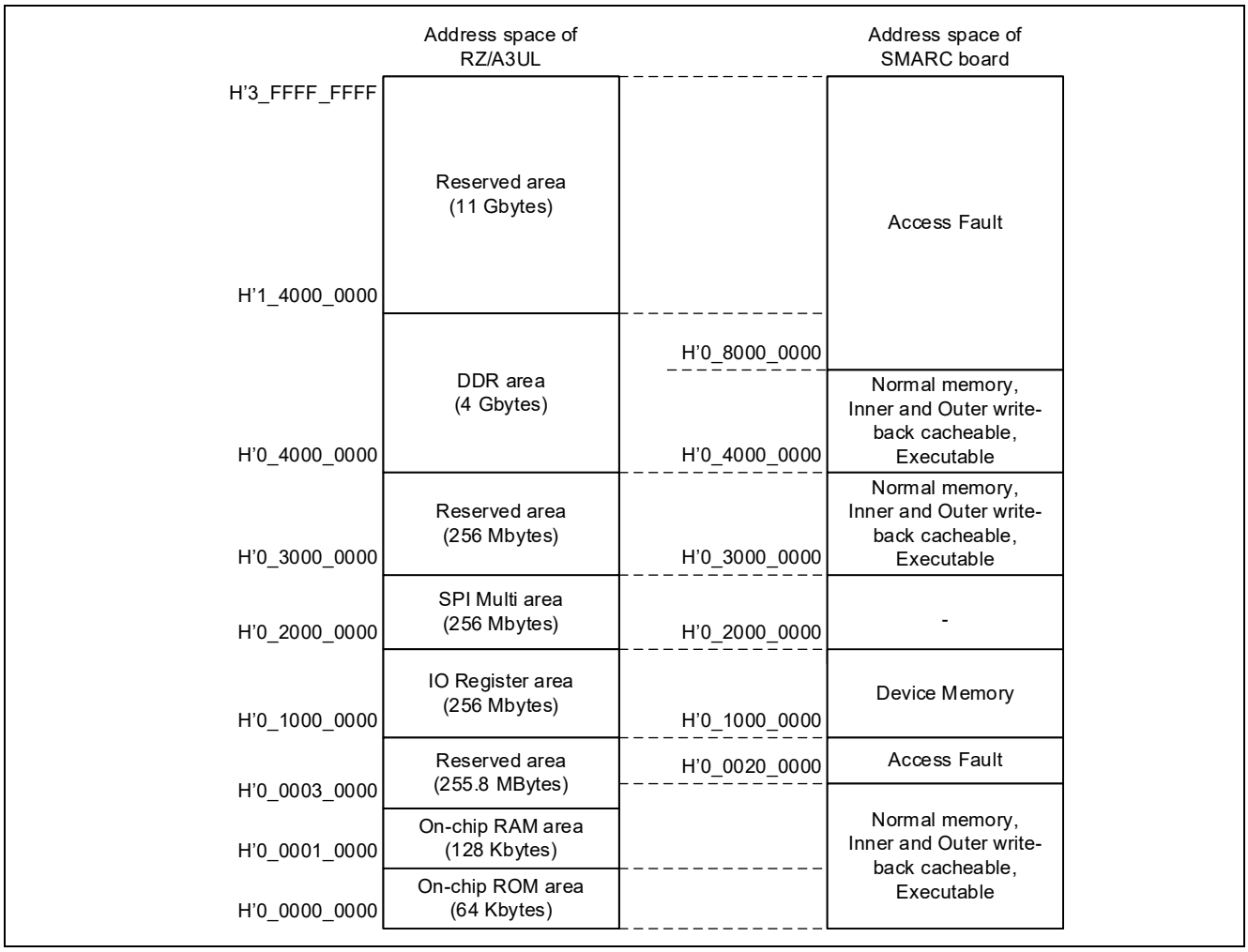


**Figure 5.4 MMU setting**

## 5.5 Interrupt used list

IPL does not use interrupt.

## 5.6 Exception level

IPL operates at exception level EL3.

It also jumps to the FSP Application in the state of EL3.

## 5.7 Application Header

IPL obtains the save address, copy destination address and entry point of the FSP Application image according to the application header written to the serial flash memory, copies the image, and jumps to the FSP Application.

IPL loads the application header from the address set in “FSP_BASE” in the make option.

Write the application header in advance in the format shown in the figure from the address set in “FSP_BASE” before executing IPL.

If “FSP_BASE” is not specified in the make command, H'2002_0000 is set as the default value. The header structure is shown below.

**Table 5.3 Application Header format**

| offset | size | Description |
|---|---|---|
| +0h | QWORD(8bytes) | FSP entry address |
| +8h | QWORD(8bytes) | Inverted FSP entry point address<br>This is used to check if the header data is correct |
| +10h | QWORD(8bytes) | Address to copy the FSP image payload<br>If FSP image payload does not need to transfer, this field must be set FSP payload address (FSP_BASE + 200h) |
| +18h | QWORD(8bytes) | Inverted address to copy the FSP image payload<br>This is used to check if the header data is correct |
| +20h | QWORD(8bytes) | FSP image payload size |
| +28h | QWORD(8bytes) | Inverted FSP image payload size<br>This is used to check if the header data is correct |
| +30h | - | Reserved (as 0) |
| +1E0h | String<br>(MAX 32 bytes) | Signature<br>“RZ FSP Application” and fill gap as 0 |
| +200h later | - | FSP image payload |

## 5.8 Flow Chart

### 5.8.1 IPL (overall) flow

Figure 5.6, Figure 5.7, Figure 5.8 show the flowchart of IPL (overall)

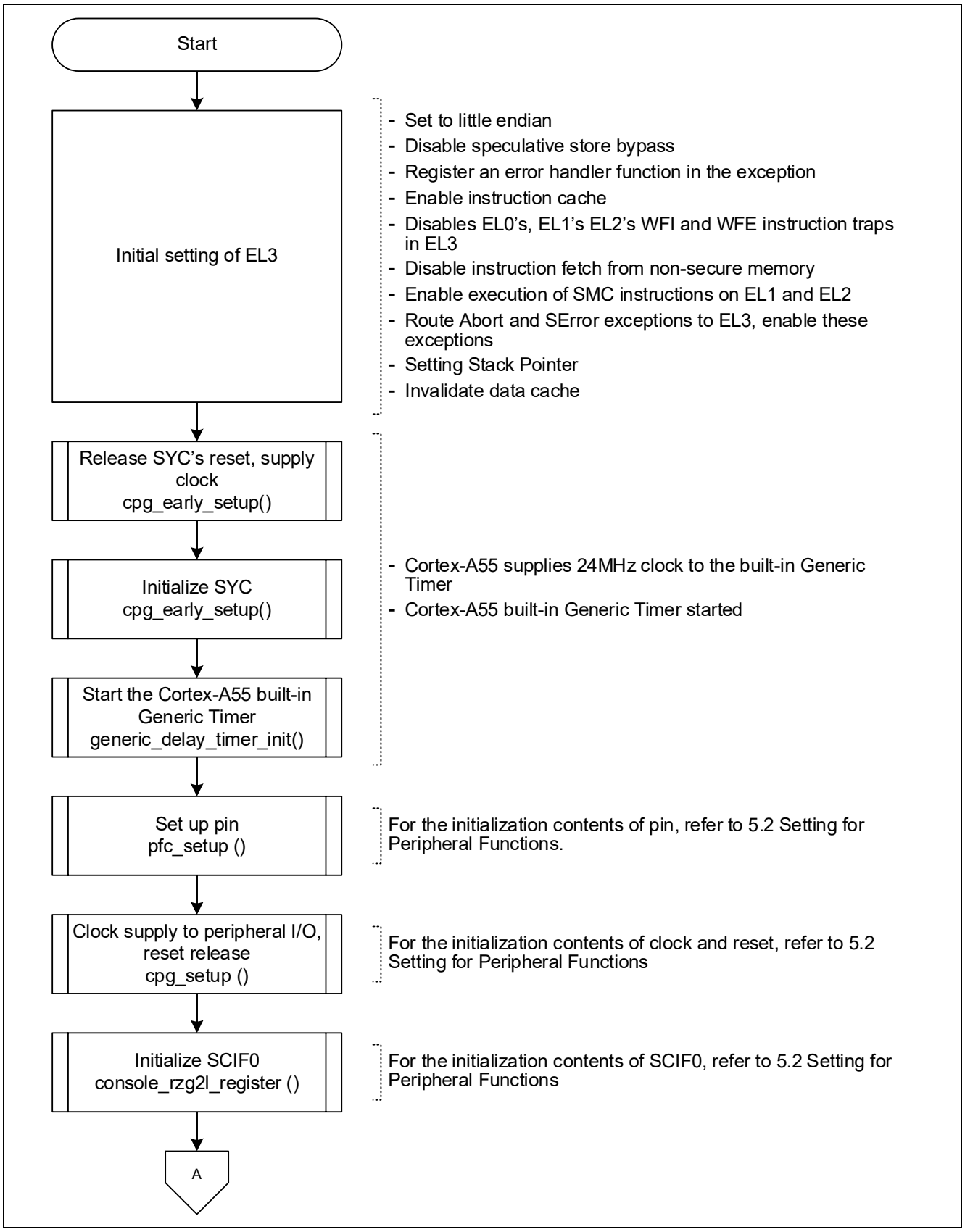


**Figure 5.5 IPL Flow Chart part1**

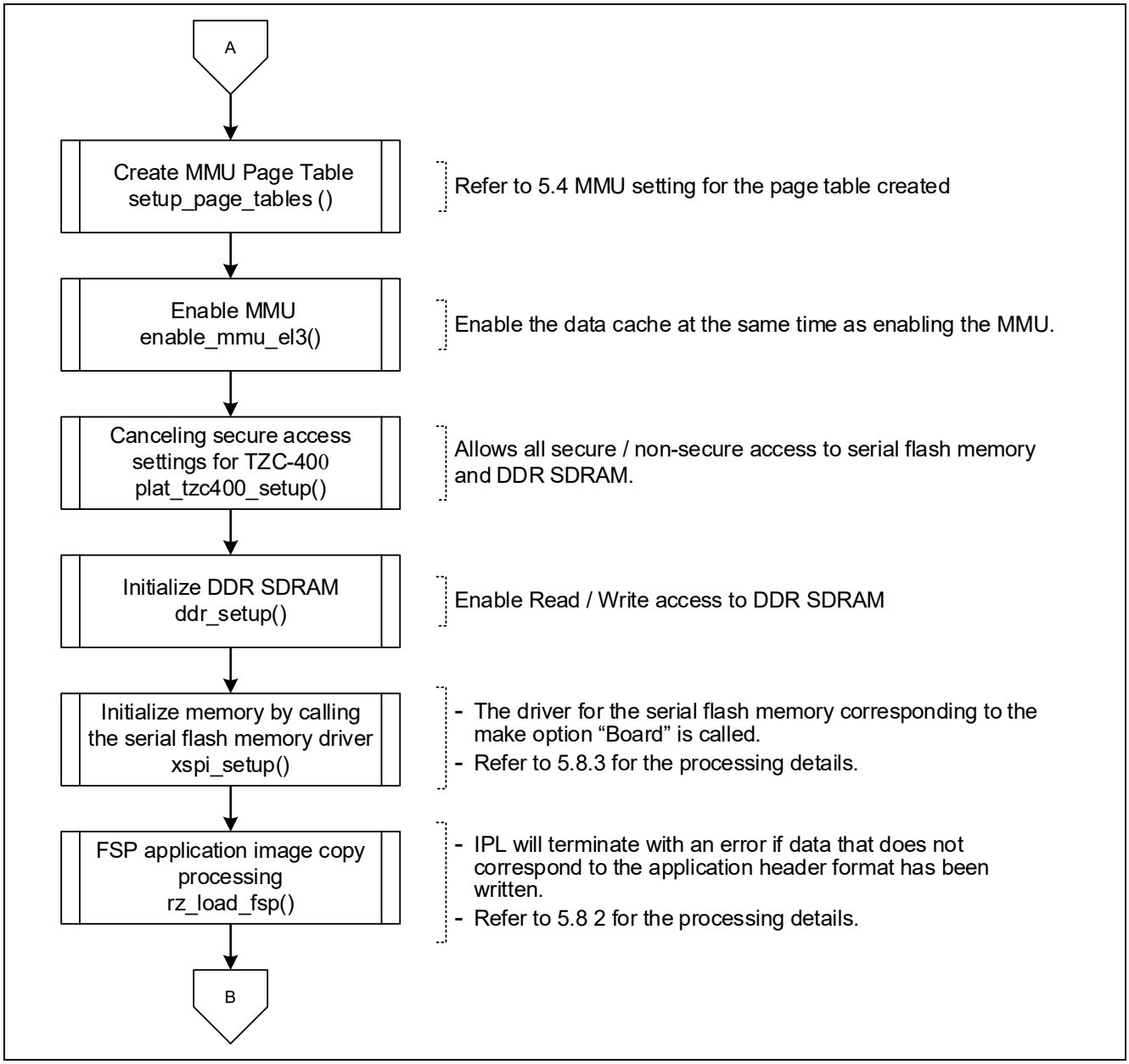


**Figure 5.6 IPL Flow Chart part2**

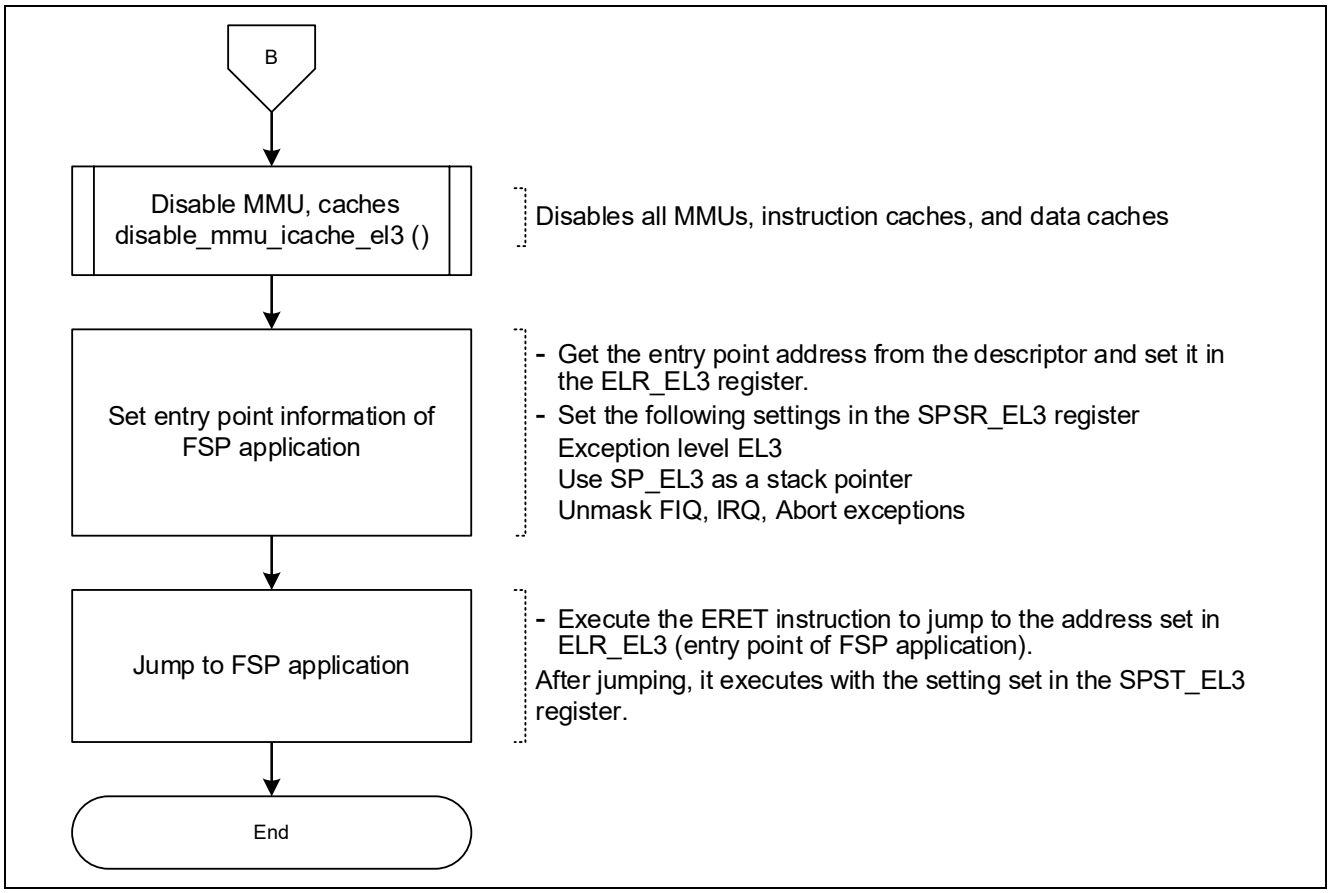


**Figure 5.7 IPL Flow Chart part3**

### 5.8.2 FSP Application Image copy process

Figure 5.9 and Figure 5.10 show the flow chart of the FSP Application image copy process.

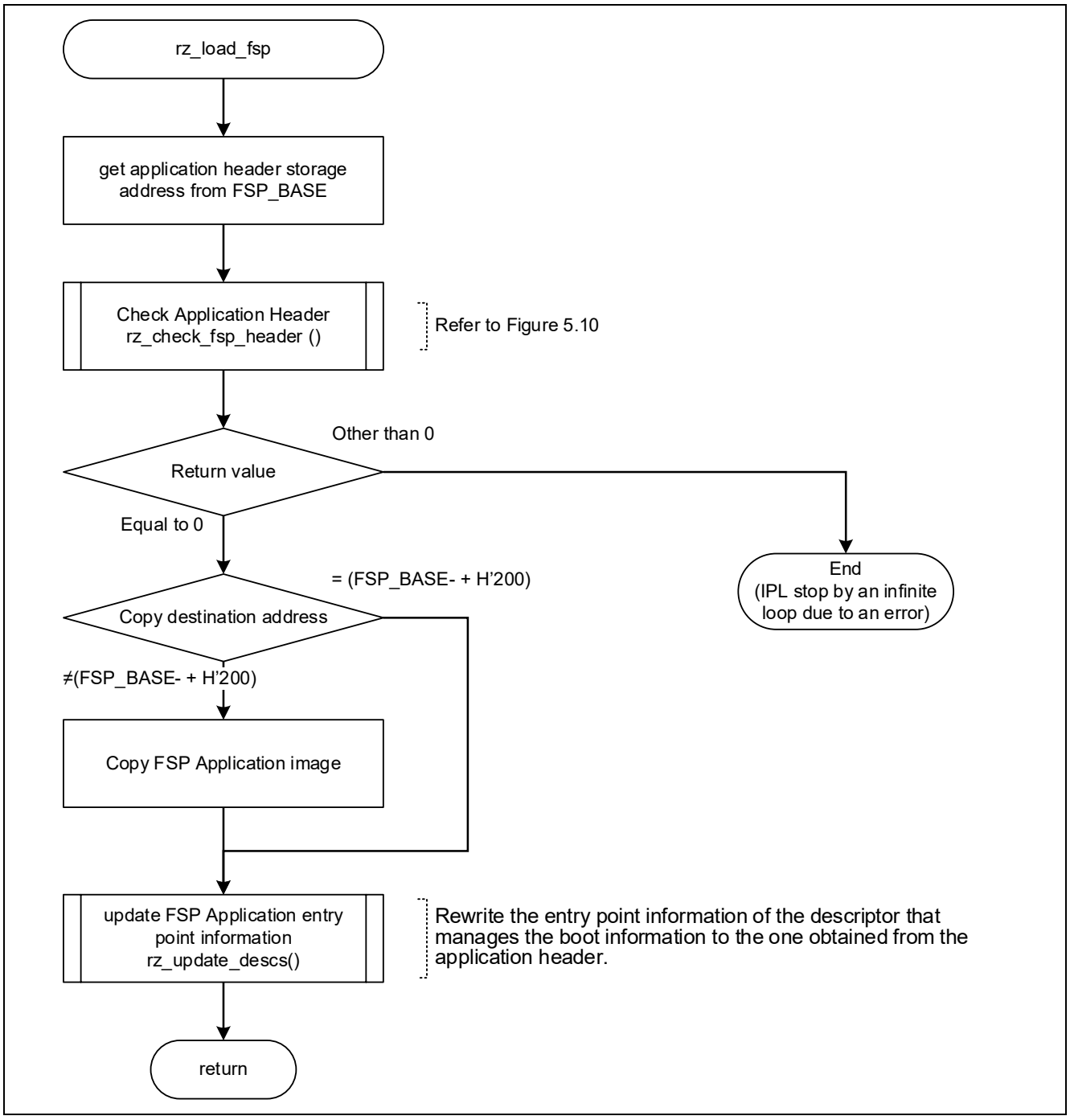


**Figure 5.8 Flow Chart of rz_load_fsp()**

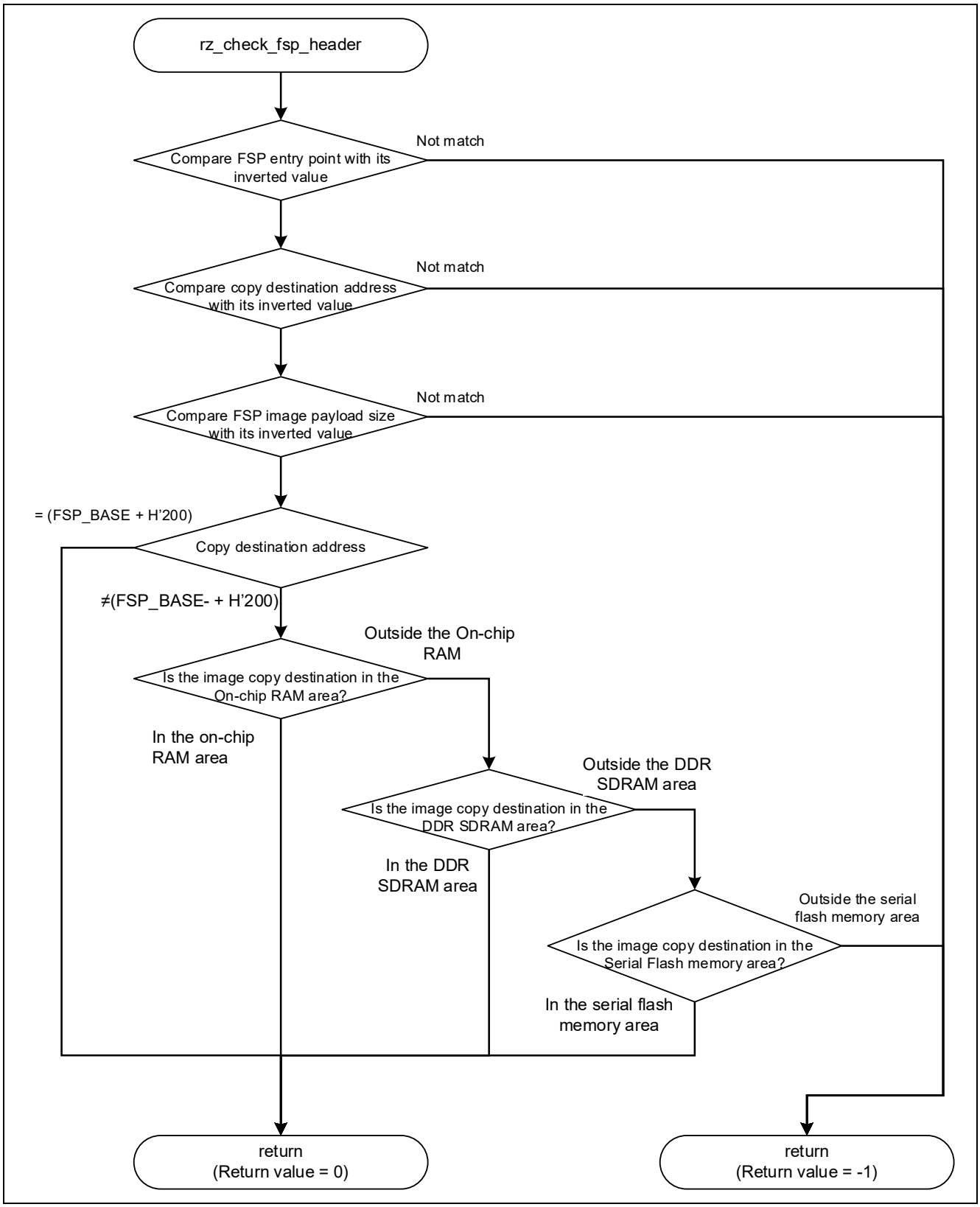


**Figure 5.9 Flow Chart of rz_check_fsp_header()**

### 5.8.3 Setting of Serial Flash Memory and SPIBSC

In this IPL, the Renesas Electronics serial flash memory (AT25QL128A) is set so that it can be accessed by XIP (Execute in place) via SPIBSC.

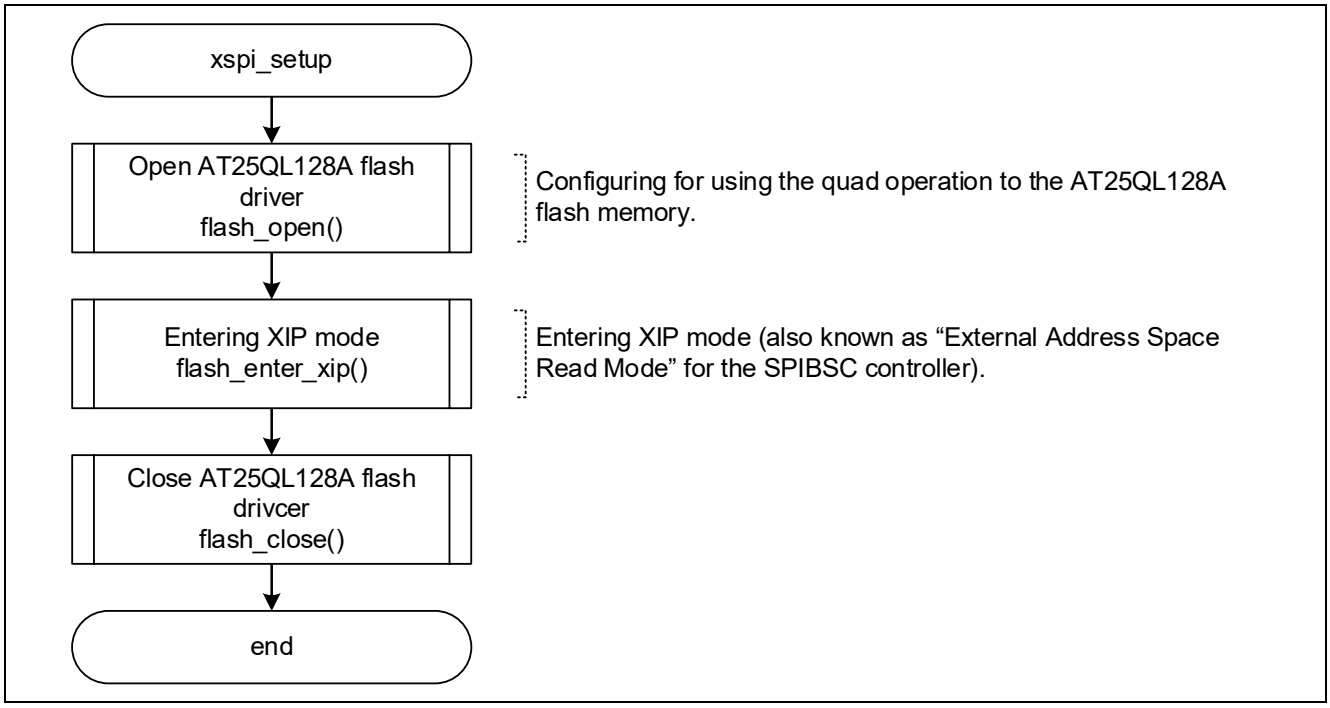


**Figure 5.10 Flowchart of xspi_setup**

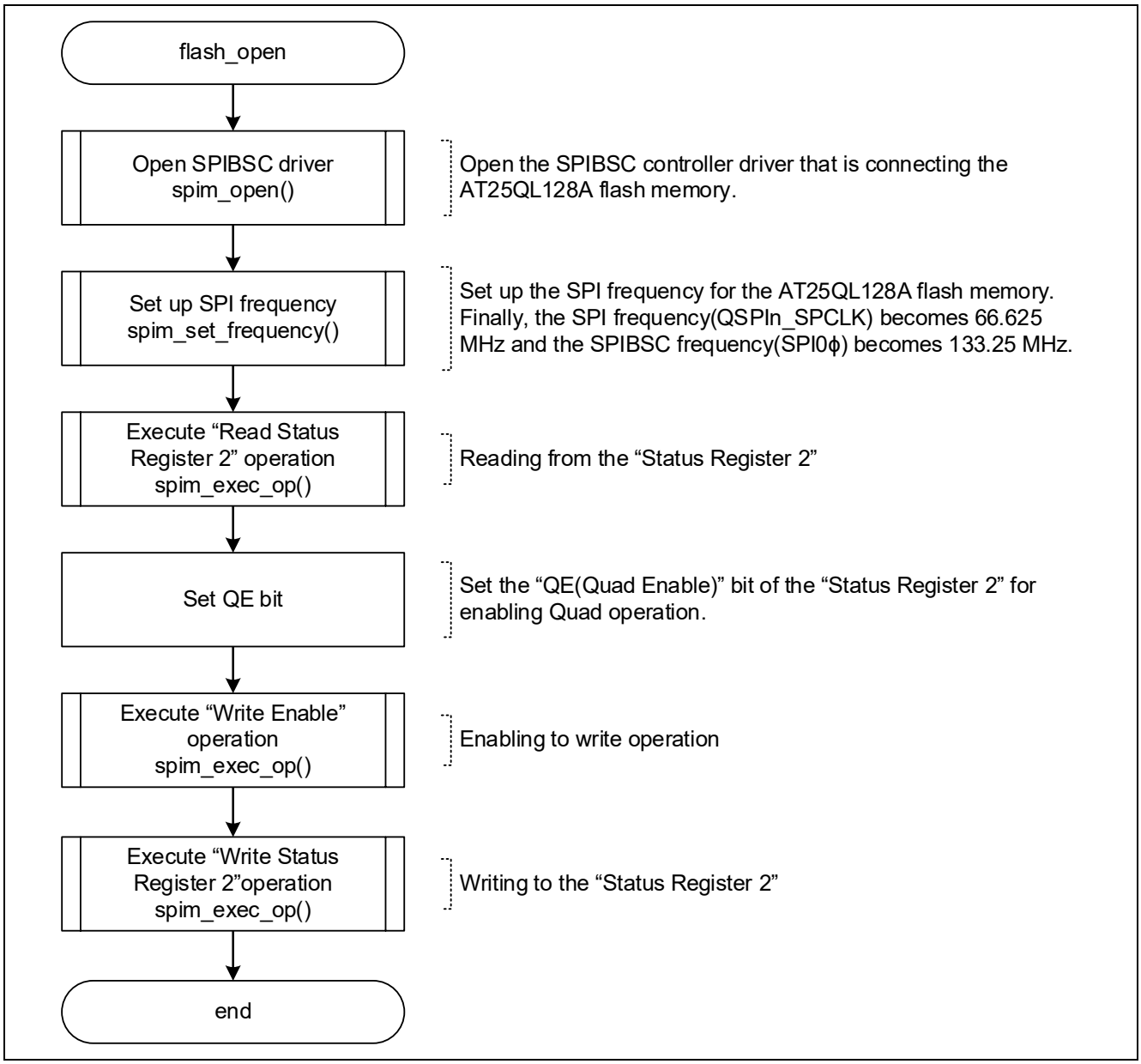


**Figure 5.11 Flowchart of flash_open**

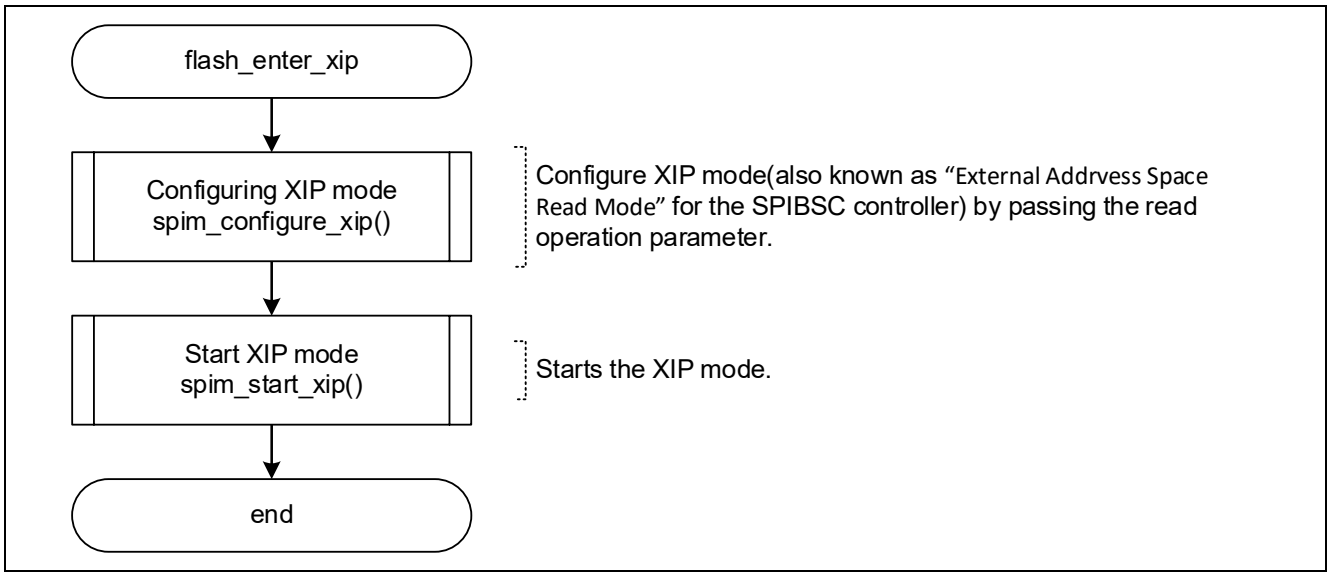


**Figure 5.12 Flowchart of flash_enter_xip**

# 6. IPL development environment construction procedure

This chapter describes the procedure for building the IPL development environment.

## 6.1 Procedure for building a development environment for Widows10 PC

The procedure for building a development environment for the Windows10 environment is shown below.

### 6.1.1 Installation of IDE (e² studio) for Windows10

IPL uses e² studio as the development environment.

This chapter shows explain the installation procedure for Windows10.

Please download the e² studio installer from the Renesas Electronics web page below.

https://www.renesas.com/jp/ja/software-tool/e-studio

Also, download the toolchain (gcc-arm-10.3-2021.07-mingw-w64-i686-aarch64-none-elf.tar.xz) from the arm developer web page below.

https://developer.arm.com/downloads/-/gnu-a

RZ/A3 FSP packs are required to build an IPL execution environment in e² studio. Please download from the following website.

https://www.renesas.com/us/en/document/scd/rza-fsp-packs-v100

#### 6.1.1.1 e² studio installation (Windows10)

1. Double-click on e² studio installer to invoke the e² studio installation wizard page. Click [Install]

**Note:** If e² studio had been installed in your PC, the option to modify, remove the existing version or install e² studio to a different location will be displayed
In this case, please select [Install].

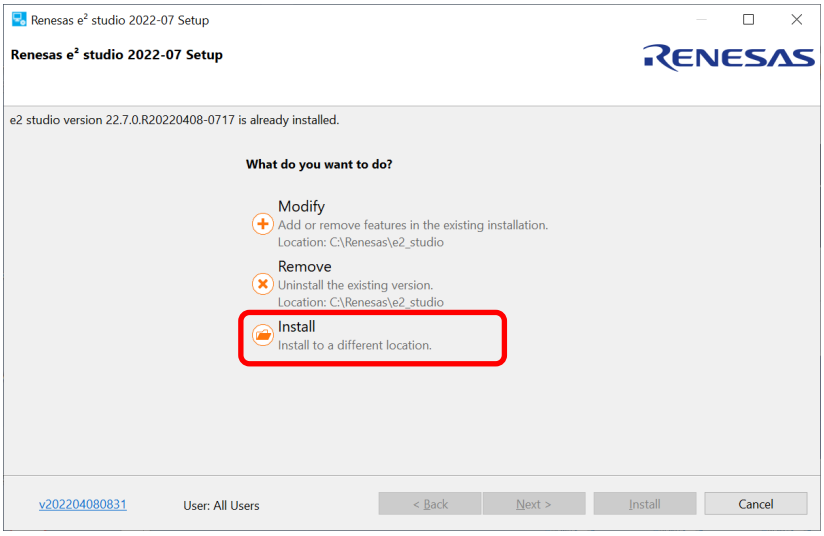


**Figure 6.1 e² studio installation wizard**

2. Welcome page
   User can change the install folder by clicking [Change...]. Click [Next] to continue.

**Note1**: If you would like to have multiple versions of e²studio, please specify new folder here.
**Note2**: Multi-byte characters cannot be used for e²studio installation folder name.

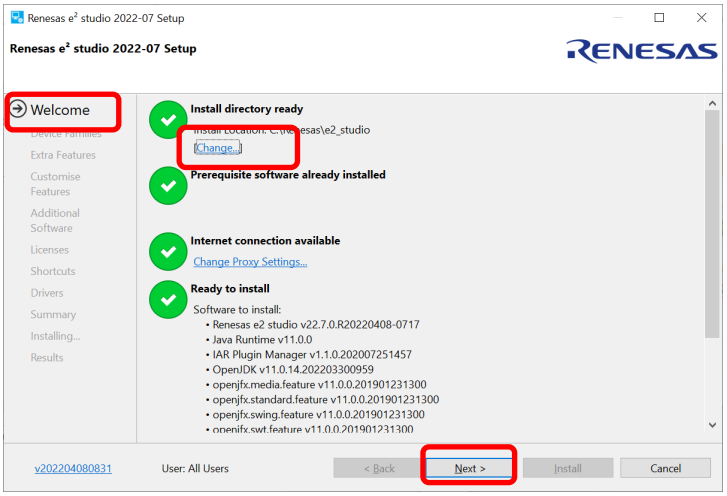


**Figure 6.2 Installation of e²studio – Welcome page**

3. Device Families
   You can select the device family to install. Select [RZ] Device Families to install. Click the [Next] button to continue.

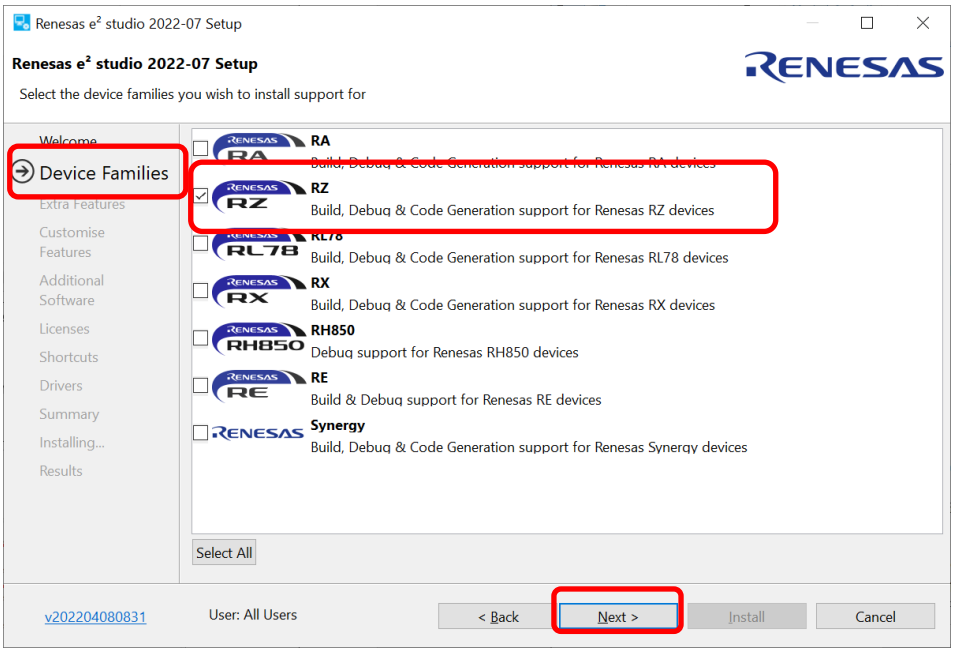


**Figure 6.3 Installation of e²studio – Device Families**

4. Extra Features
   Select Extra Features (i.e., Language packs, SVN & Git support, RTOS support...) to be installed. For non-English language users, please select Language packs at this step if needed.
   This document describes an example of using e² studio terminal, so please check Terminals.
   Click the [Next] button to continue.

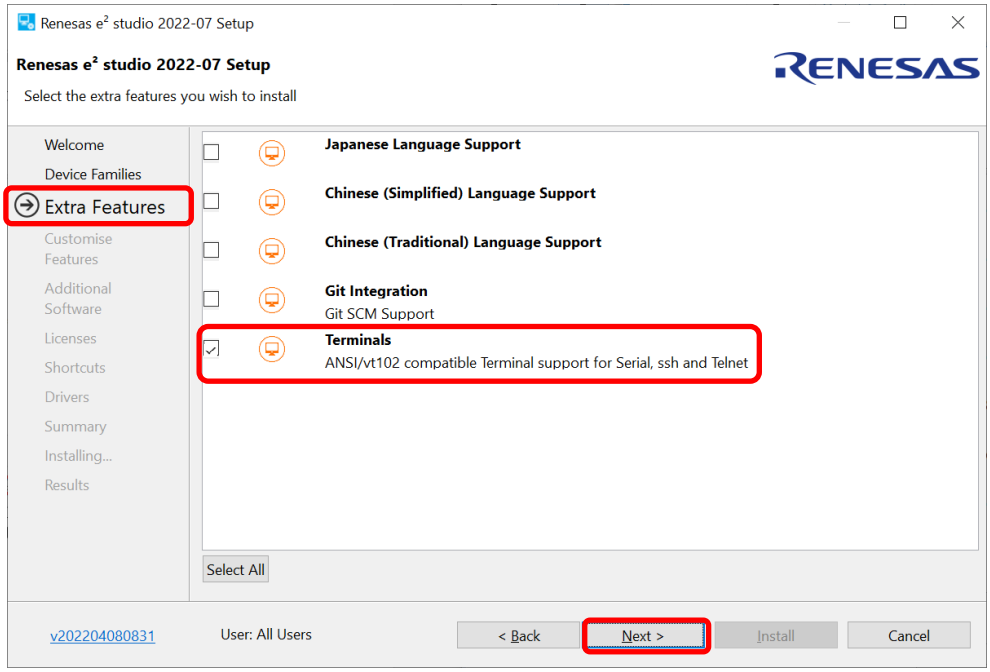


**Figure 6.4 Installation of e² studio – Extra Features**

5. Customize Features
   Be sure to select “Renesas FSP Smart Configurator Core”.

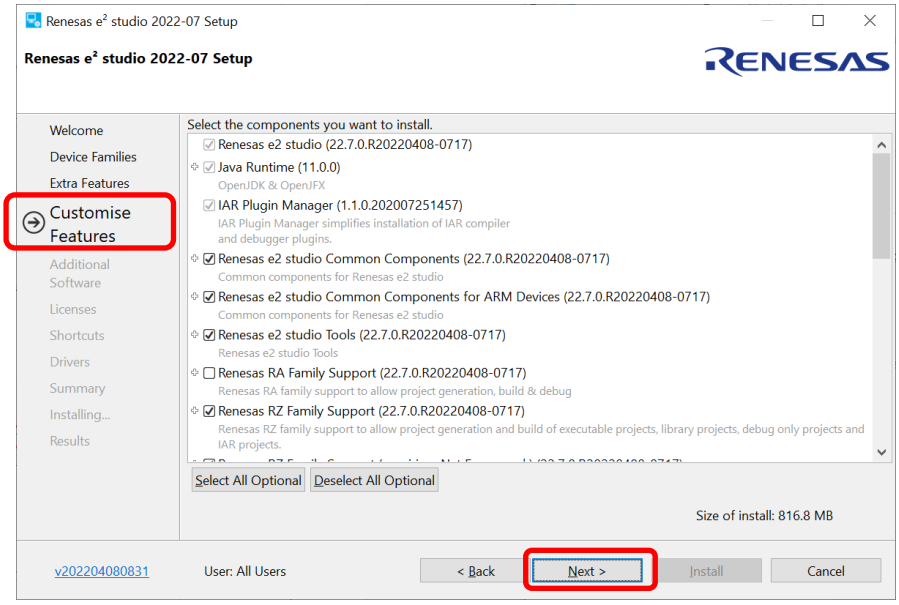


**Figure 6.5 Installation of e²studio – Features**

6. Additional Software
   There is no need to install additional software. Thus, please click [Next].

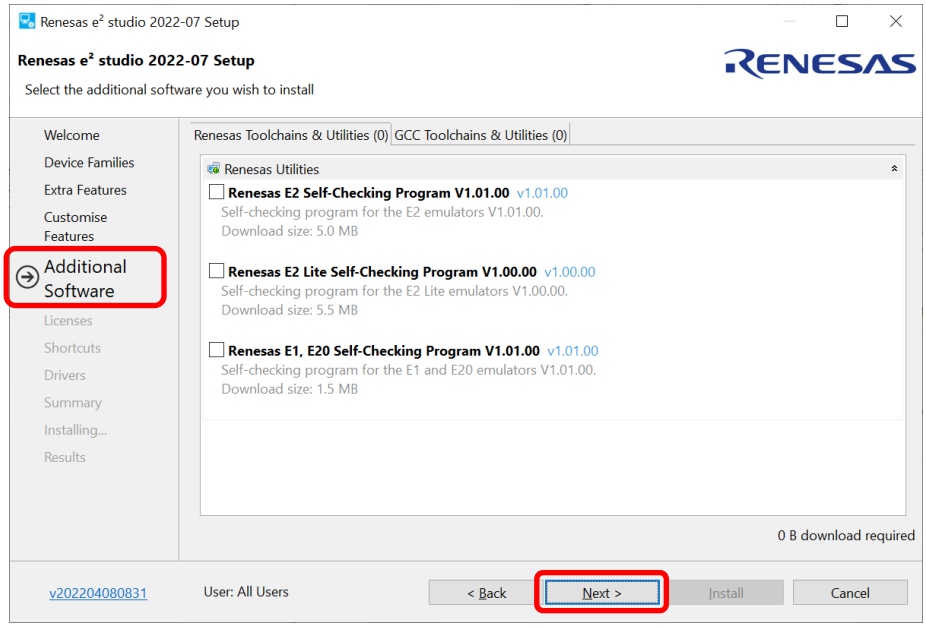


**Figure 6.6 Installation of e²studio – Additional Software**

7. License Agreement
   Read and accept the software license agreement. Click the [Next] button. Please note that user must accept the license agreement, otherwise installation cannot be continued.

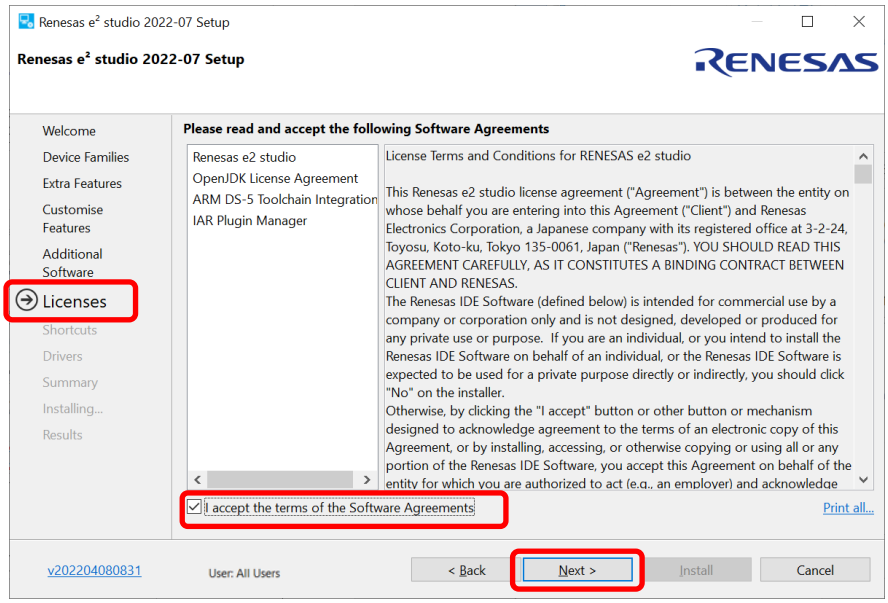


**Figure 6.7 Installation of e²studio – Licenses**

8. Shortcuts
   Select shortcut name for start menu and click [Next] button to continue.

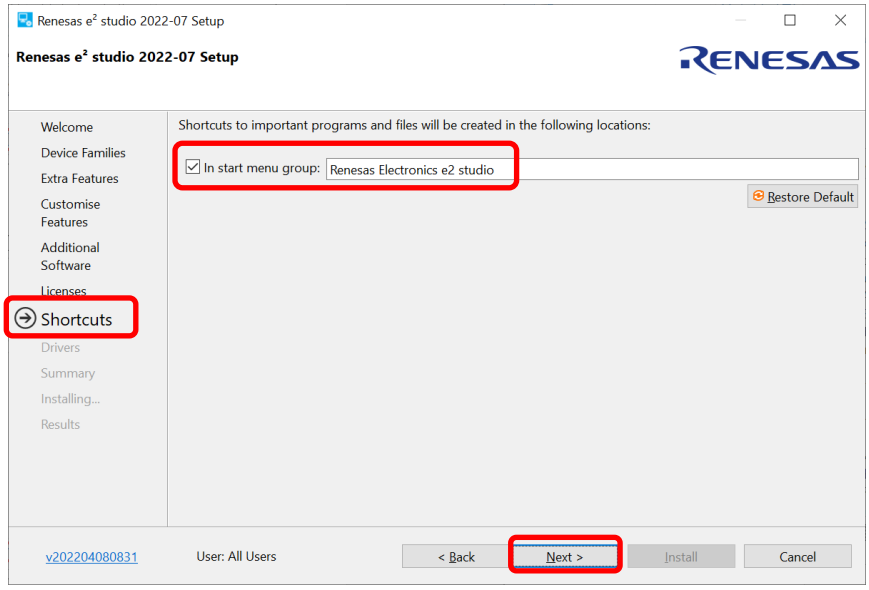


**Figure 6.8 Installation of e²studio – Shortcuts**

9. Drivers
   Leave the defaults and click [Next].

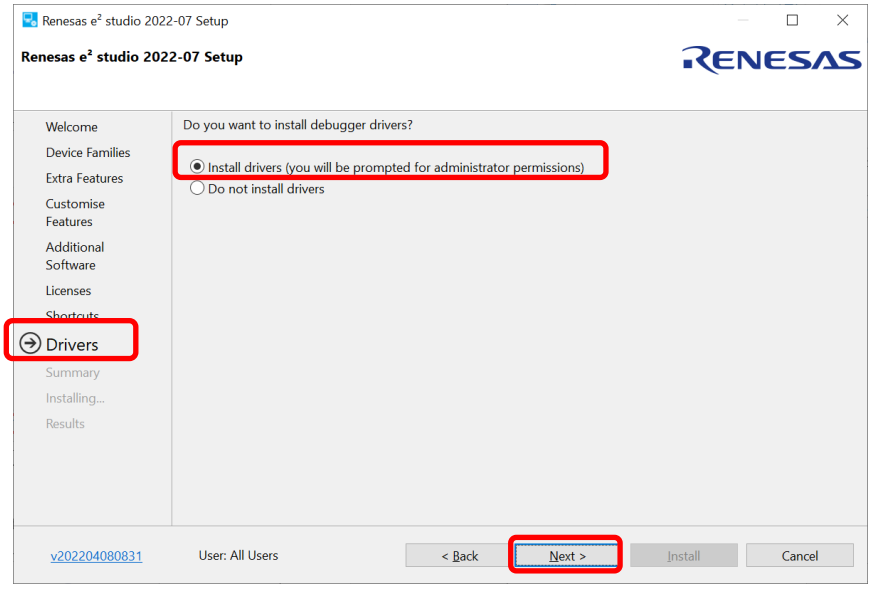


**Figure 6.9 Installation of e²studio – Drivers**

10. Summary
    Components list to be installed is shown. Please confirm the contents and click the [Install] button to install the Renesas e² studio IDE.

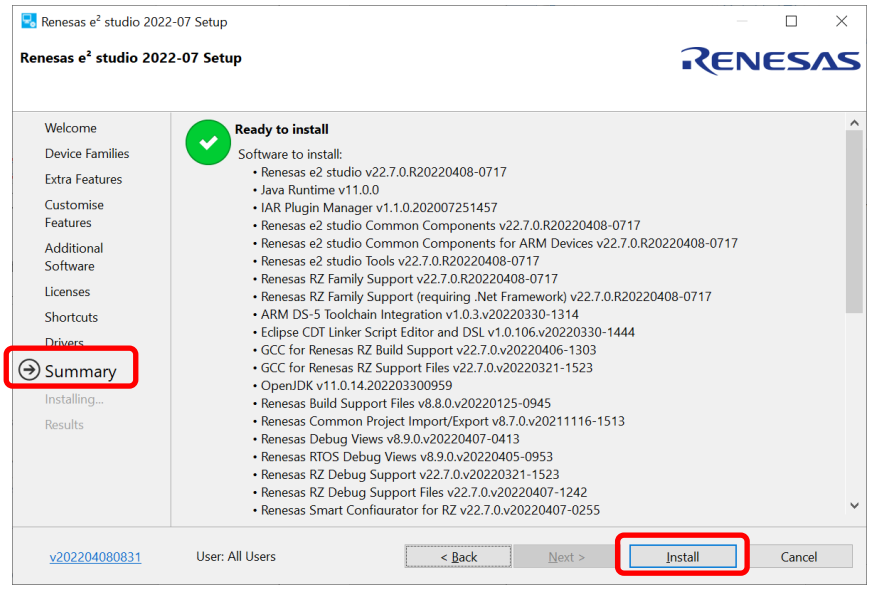


**Figure 6.10 Installation of e²studio – Summary**

11. Installing
    The installation is performed. Depending on selected items of additional software, new dialog prompts may appear during the installation process.

12. Results
    Click the [OK] button to complete the installation.

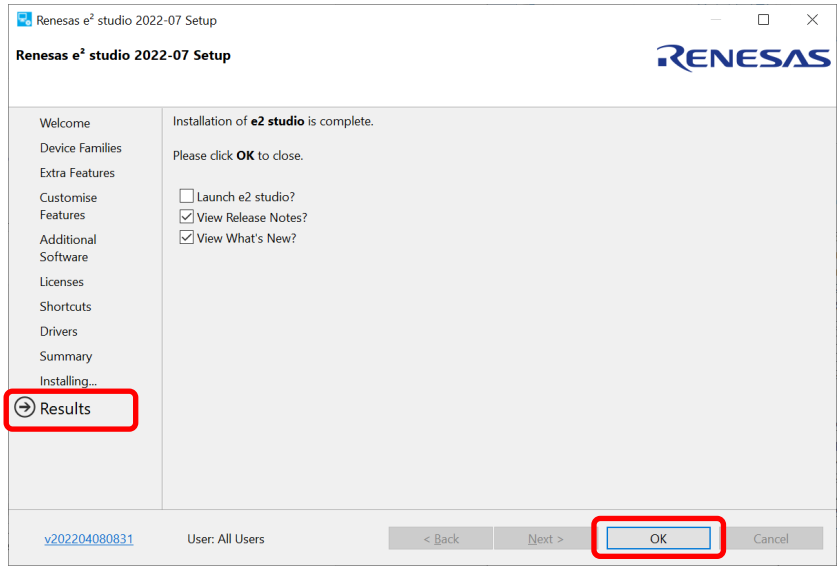


**Figure 6.11 Summary Page**

#### 6.1.1.2 Toolchain installation (Windows10)

Install the toolchain for use in the RZ/A3UL FSP environment.

1. Unzip gcc-arm-10.3-2021.07-mingw-w64-i686-aarch64-none-elf.tar.xz to any folder.

This document describes the procedure when the file is unzipped to "C:\".

#### 6.1.1.3 Toolchain registration (Windows10)

Next, you need to register the toolchain on e² studio. Here is the procedure for the registration.

1. Launch e² studio and select [Help] -> [Add Renesas Toolchains].
2. Then click [Add] to bring up the Add New Toolchain window.

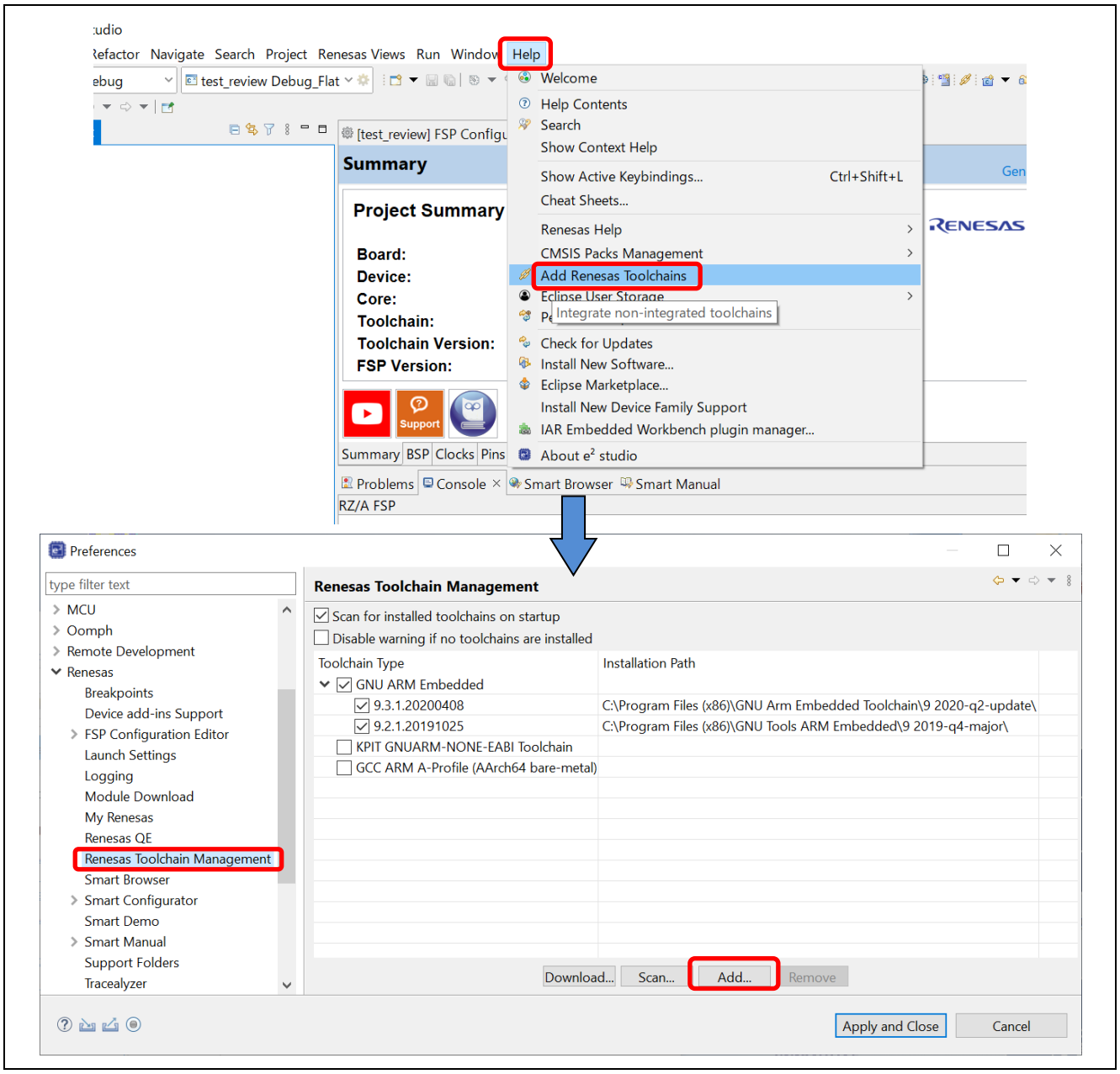


**Figure 6.12 Toolchain registration (1)**

3. Click [Browse], select the toolchain installation folder "C:\gcc-arm-10.3-2021.07-mingw-w64-i686-aarch64-none-elf\" and click [OK].
4. Confirm that the folder registered in GCC ARM A-Profile has been added and click [Apply and Close].

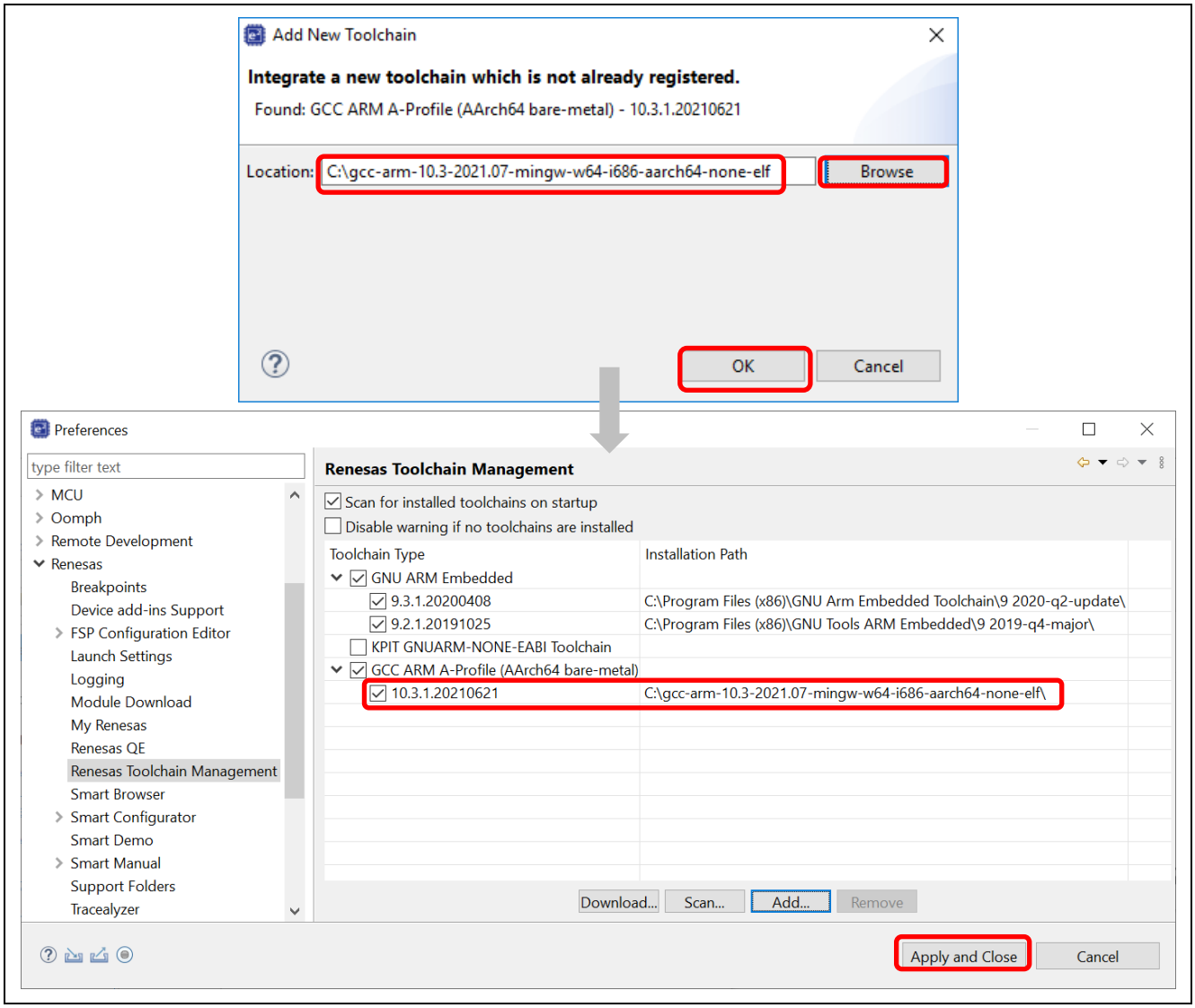


**Figure 6.13 Toolchain registration (2)**

#### 6.1.1.4 Installation of RZ/A FSP packs (Windows10)

Install the FSP packs for use in the RZ/A3UL IPL environment.

1. Exit e² studio
2. Extract FSP Packs.
3. Copy the extracted FSP Packs (named “internal”) to the e2studio installation folder as shown below:
   "C:\Renesas\e2_studio\internal\projectgen\rz_fsp\rza\packs"

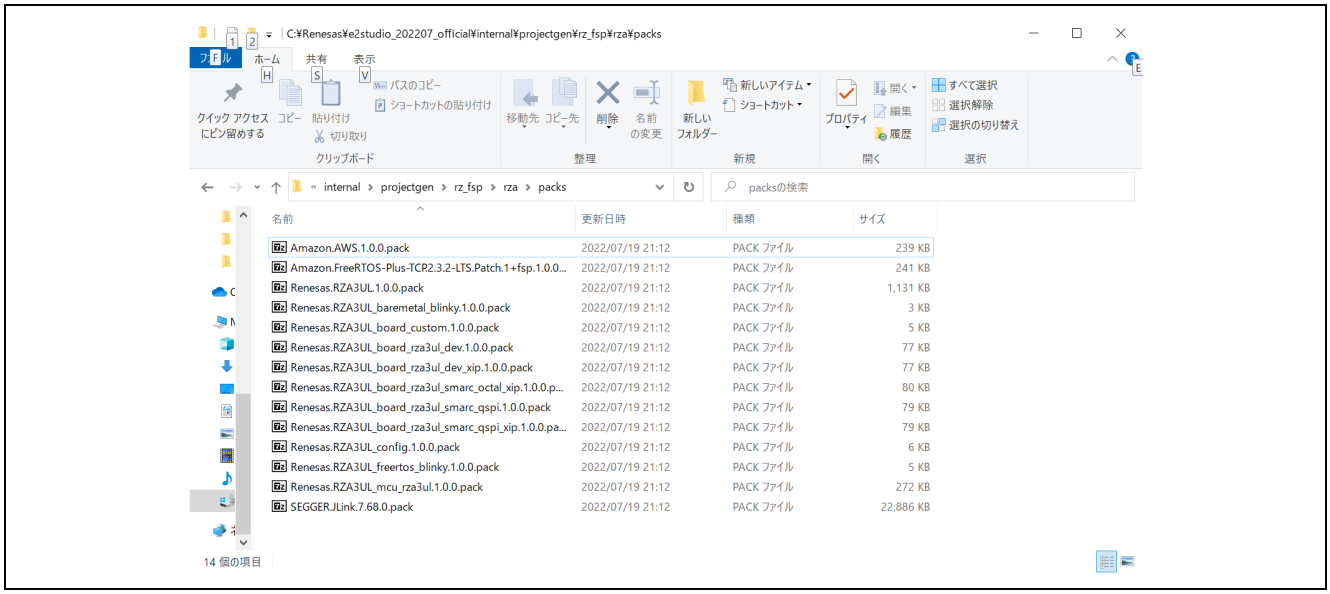


**Figure 6.14 Installed packs folder (Windows10)**

4. Please re-launch e2 studio
   Then packs will be reflected in the development environment.

### 6.1.2 MSYS2 Installation

You must install a tool called MSYS2 to build the IPL in Windows10 environment.

1. Download MSYS2 from "Installation" on the following website.
   Please install by following the procedure described in "Installation".
   https://www.msys2.org/
   This document describes the case where MSYS2 is installed in the following folder.
   c:\msys64

2. Install the make command on MSYS2.
   Open the windows start menu and start MSYS2 MSYS.

**Figure 6.15 Launch MSYS2**

3. A command window will appear.
   Enter "pacman -S make“ on command window the make command on MSYS2.

**Figure 6.16 Make command installation (1)**

4. Enter ‘y’ when "Proceed with installation" [Y / n] "is output.

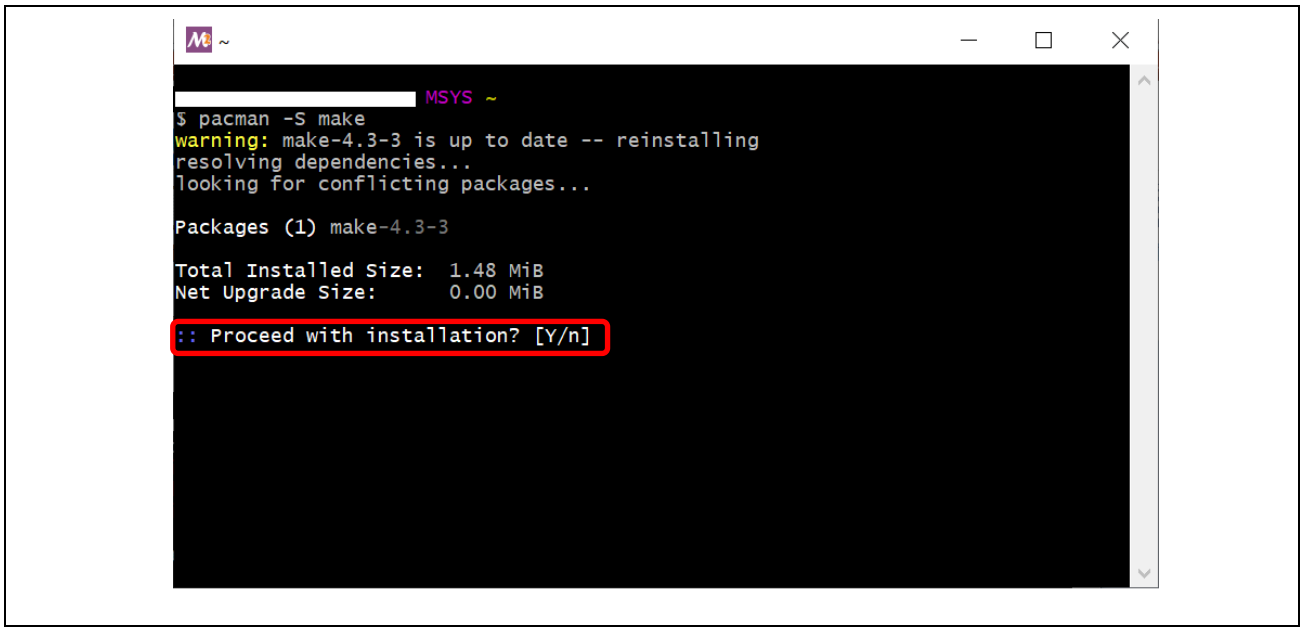


**Figure 6.17 Make command installation (2)**

5. The make installation is completed.

```
MSYS ~
$ pacman -S make
warning: make-4.3-3 is up to date -- reinstalling
resolving dependencies...
looking for conflicting packages...

Packages (1) make-4.3-3

Total Installed Size:  1.48 MiB
Net Upgrade Size:      0.00 MiB

:: Proceed with installation? [Y/n] y
(1/1) checking keys in keyring                      [#####################] 100%
(1/1) checking package integrity                    [#####################] 100%
(1/1) loading package files                         [#####################] 100%
(1/1) checking for file conflicts                   [#####################] 100%
(1/1) checking available disk space                 [#####################] 100%
:: Processing package changes...
(1/1) reinstalling make                             [#####################] 100%
:: Running post-transaction hooks...
(1/1) Updating the info directory file...

MSYS ~
$
```

**Figure 6.18 Make command installation (3)**

## 6.2 Procedure for building a development environment for Ubuntu 20.04 LTS

The procedure for building a development environment for Ubuntu 20.04 LTS Desktop environment is shown below.

### 6.2.1 Installation of IDE (e² studio) for Ubuntu 20.4 LTS

IPL uses e² studio as the development environment.

This chapter shows explain the installation procedure for Windows10.

Please download the e² studio installer from the Renesas Electronics web page below.

https://www.renesas.com/jp/ja/software-tool/e-studio

Also, download the toolchain (gcc-arm-10.3-2021.07-x86_64-aarch64-none-elf.tar.xz) from the arm developer web page below.

https://developer.arm.com/downloads/-/gnu-a

RZ/A3 FSP packs are required to build an IPL execution environment in e² studio. Please download from the following website.

https://www.renesas.com/us/en/document/scd/rza-fsp-packs-v100

#### 6.2.1.1 e² studio installation (Ubuntu 20.4 LTS)

1. Execute the following command to give execute permission to the e² studio installer.
   Then, run the installer.
   The command is an example when the e² studio installer
   (file name: “e2studio_installer-2022-07_linux_host.run”) is saved in the “~/Downloads” folder.
   Please change the command according to your environment.

```
$ cd ~/Downloads
$ sudo chmod +x e2studio_installer-2022-07_linux_host.run
$ ./e2studio_installer-2022-07_linux_host.run
```

2. Welcome page
   The welcome page will be launched.
   User can change the install folder by clicking [Change...]. Click [Next] to continue.

**Note1**: If you would like to have multiple versions of e² studio, please specify new folder here.
**Note2**: Multi-byte characters cannot be used for e² studio installation folder name.

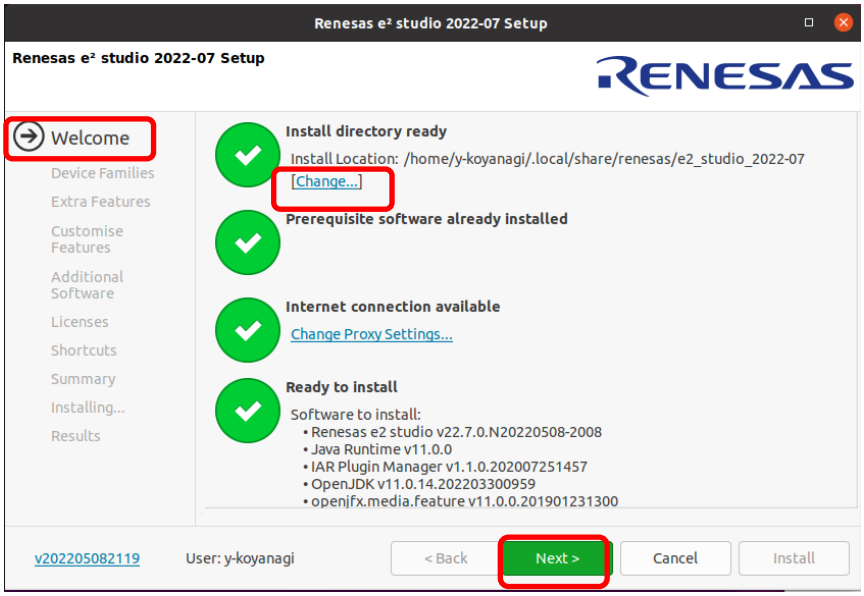


**Figure 6.19 Installation of e² studio – Welcome page**

3. Device Families
   You can select the device family to install. Select [RZ] Device Families to install. Click the [Next] button to continue.

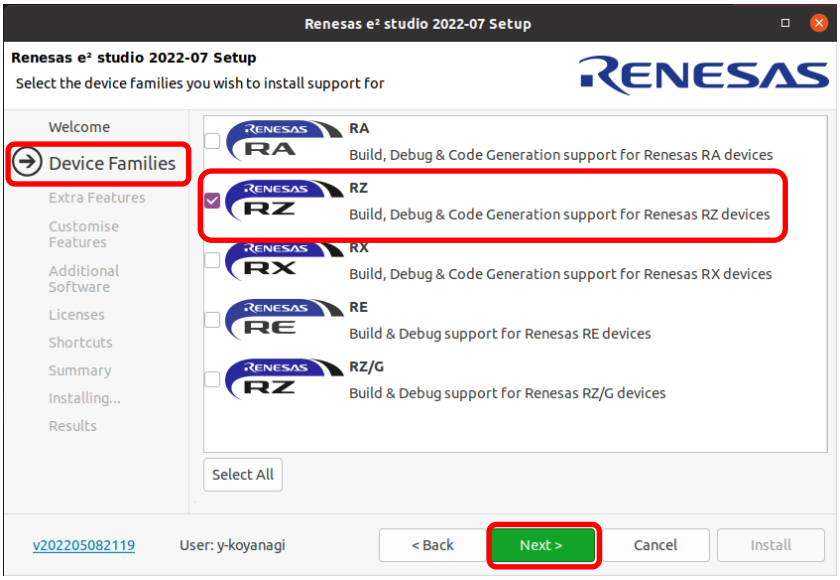


**Figure 6.20 Installation of e² studio – Device Families**

4. Extra Features
   Select Extra Features (i.e., Language packs, SVN & Git support, RTOS support...) to be installed. For non-English language users, please select Language packs at this step if needed.
   This document describes an example of using e² studio terminal, so please check Terminals.
   Click the [Next] button to continue.

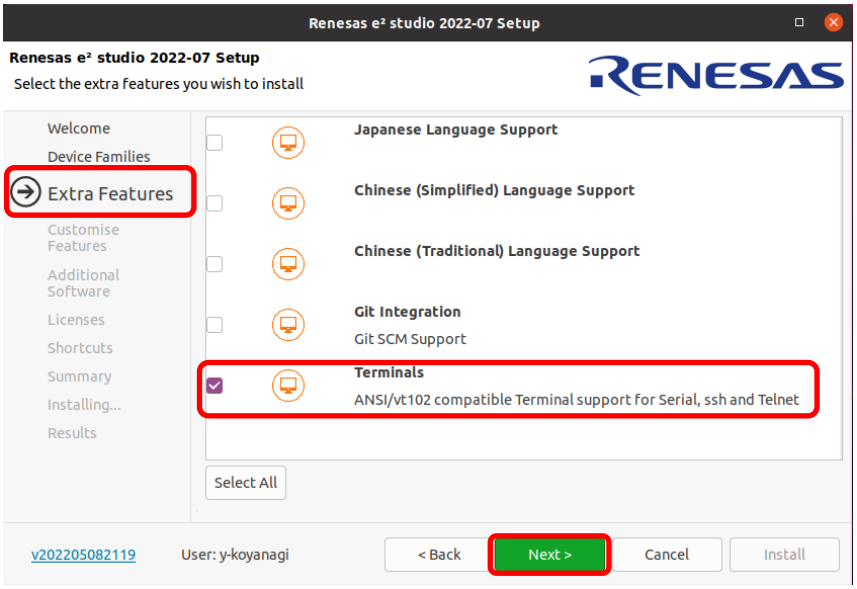


**Figure 6.21 Installation of e² studio – Extra Features**

5. Customize Features
   Be sure to select “Renesas FSP Smart Configurator Core”.

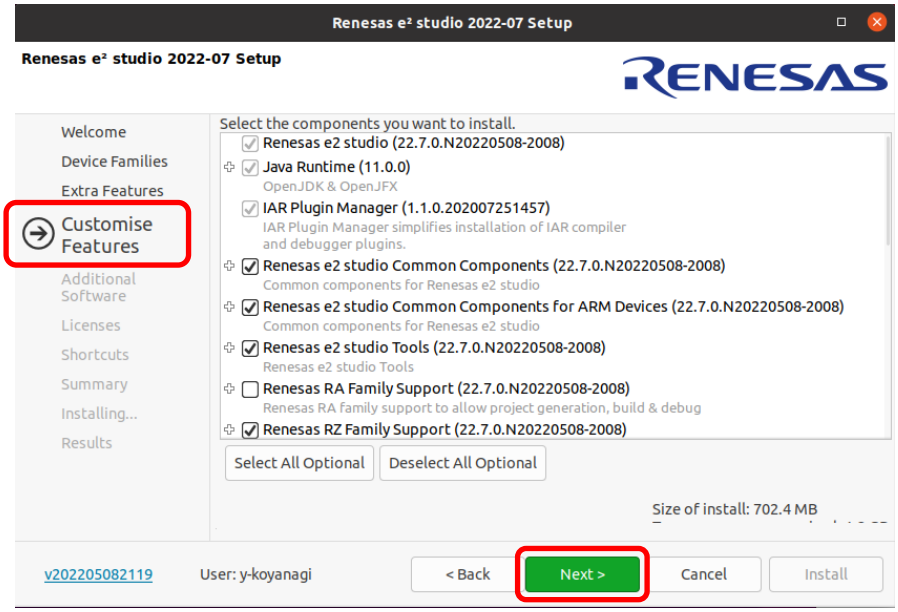


**Figure 6.22 Installation of e² studio – Features**

6. Additional Software
   There is no need to install additional software. Thus, please click [Next] as is.

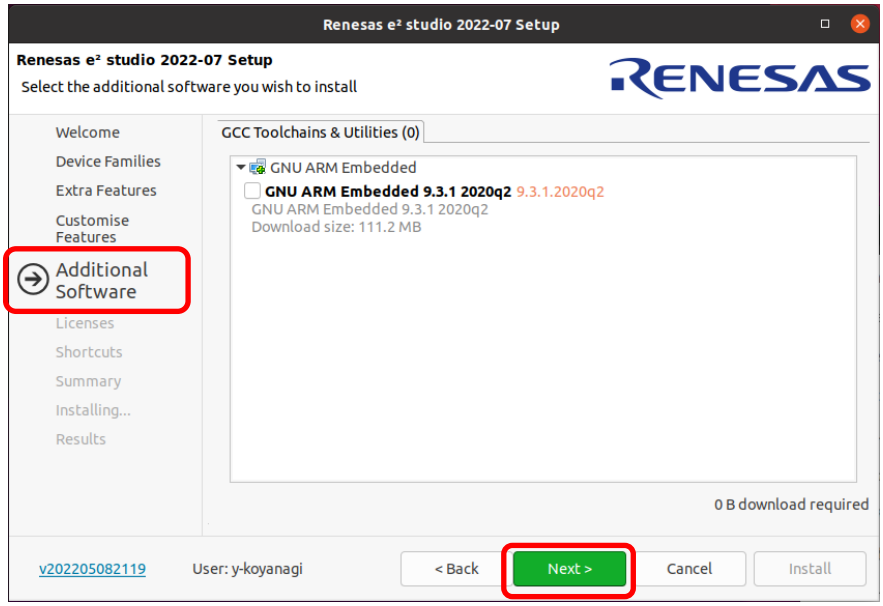


**Figure 6.23 Installation of e² studio – Additional Software**

7. License Agreement
   Read and accept the software license agreement. Click the [Next] button. Please note that user must accept the license agreement, otherwise installation cannot be continued.

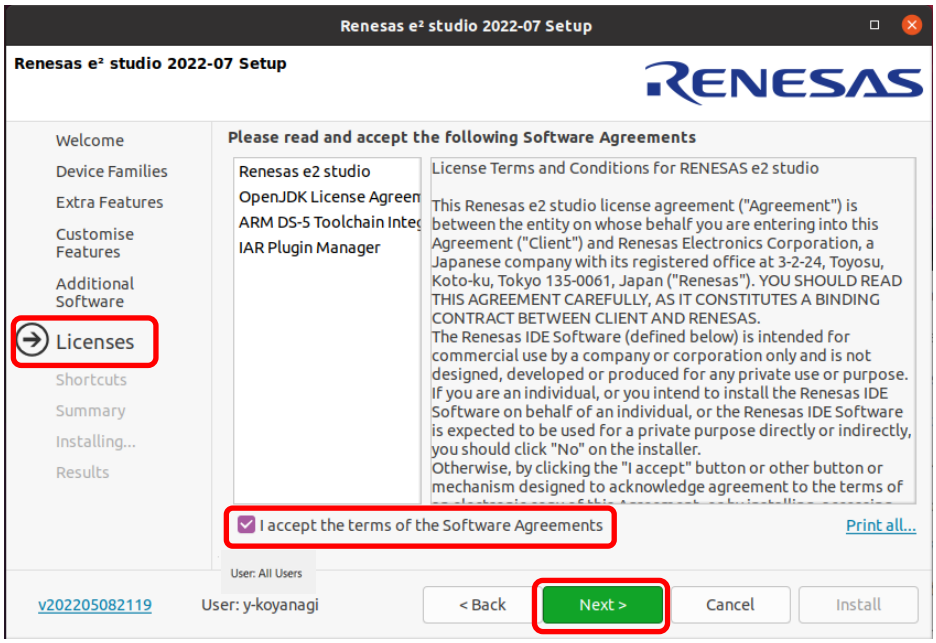


**Figure 6.24 Installation of e² studio – Licenses**

8. Shortcuts
   Check ”In the Application launcher” to register for the launcher and click [Next] button to continue.

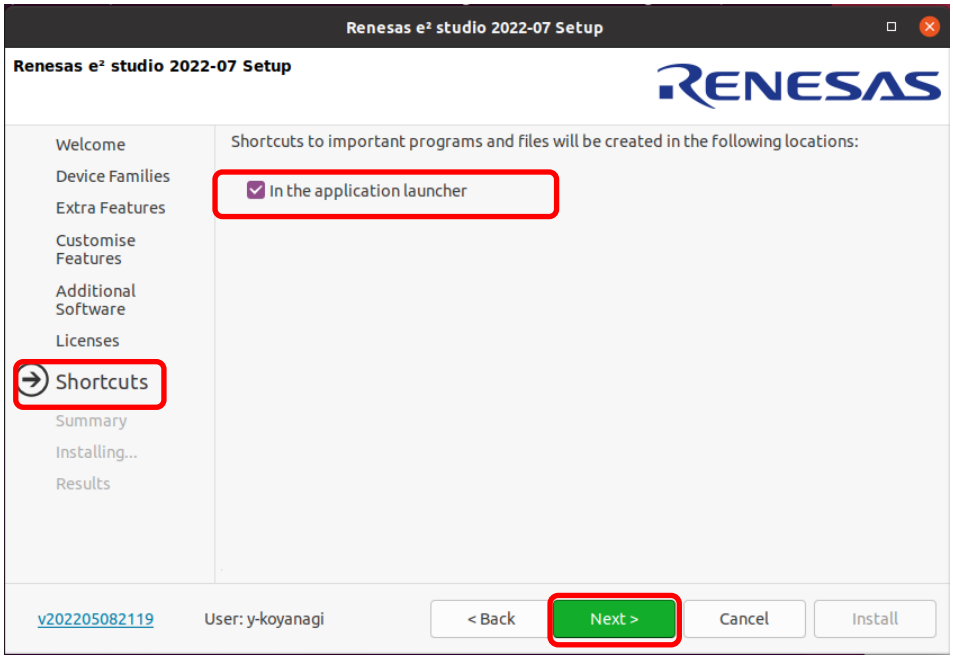


**Figure 6.25 Installation of e² studio – Shortcuts**

9. Summary
   Components list to be installed is shown. Please confirm the contents and click the [Install] button to install the Renesas e² studio IDE.

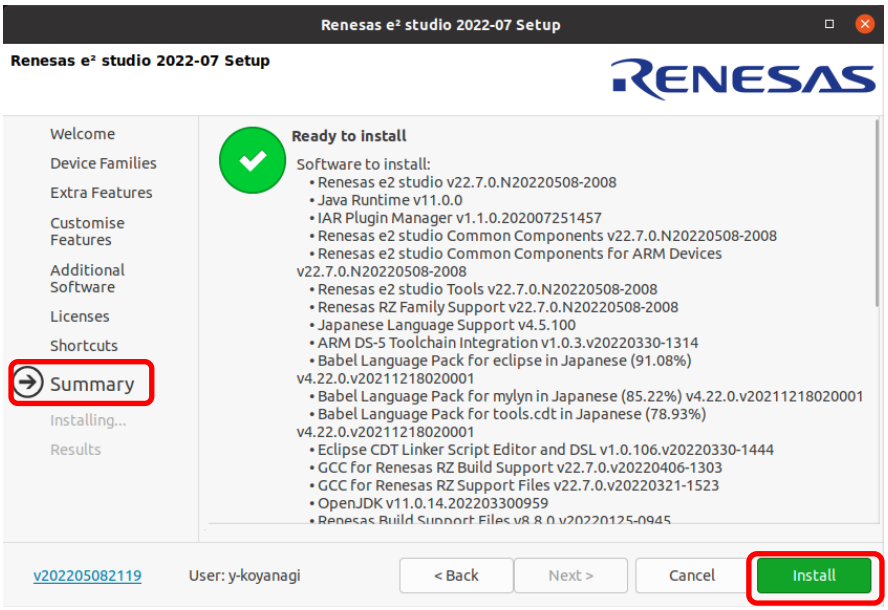


**Figure 6.26 Installation of e² studio – Summary**

10. Installing
    The installation is performed. Depending on selected items of additional software, new dialog prompts may appear during the installation process.

11. Results

Click the [OK] button to complete the installation.

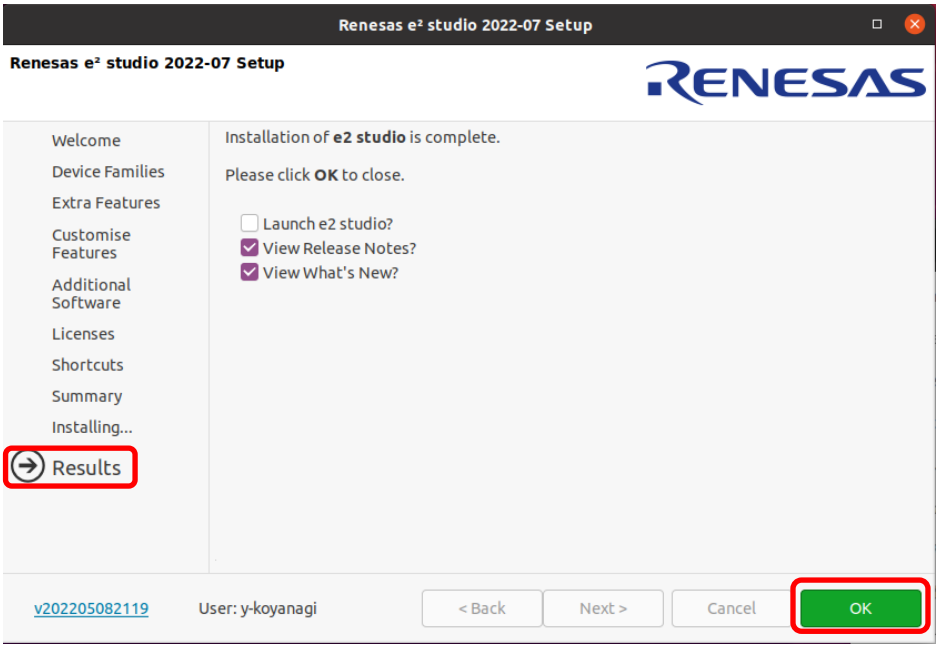


**Figure 6.27 Results Page**

#### 6.2.1.2 NoteToolchain installation (Ubuntu20.4 LTS)

Install the toolchain for use in the RZ/A3UL FSP environment.

1. Unzip gcc-arm-10.3-2021.07-x86_64-aarch64-none-elf.tar.xz to any folder.

This document describes the procedure when the file is unzipped to "/opt/".

#### 6.2.1.3 Toolchain registration (Ubuntu 20.4 LTS)

Next, you need to register the toolchain on e² studio. Here is the procedure for the registration.

1. Launch e² studio and select [Help] -> [Add Renesas Toolchains].
2. Then click [Add] to bring up the Add New Toolchain window.

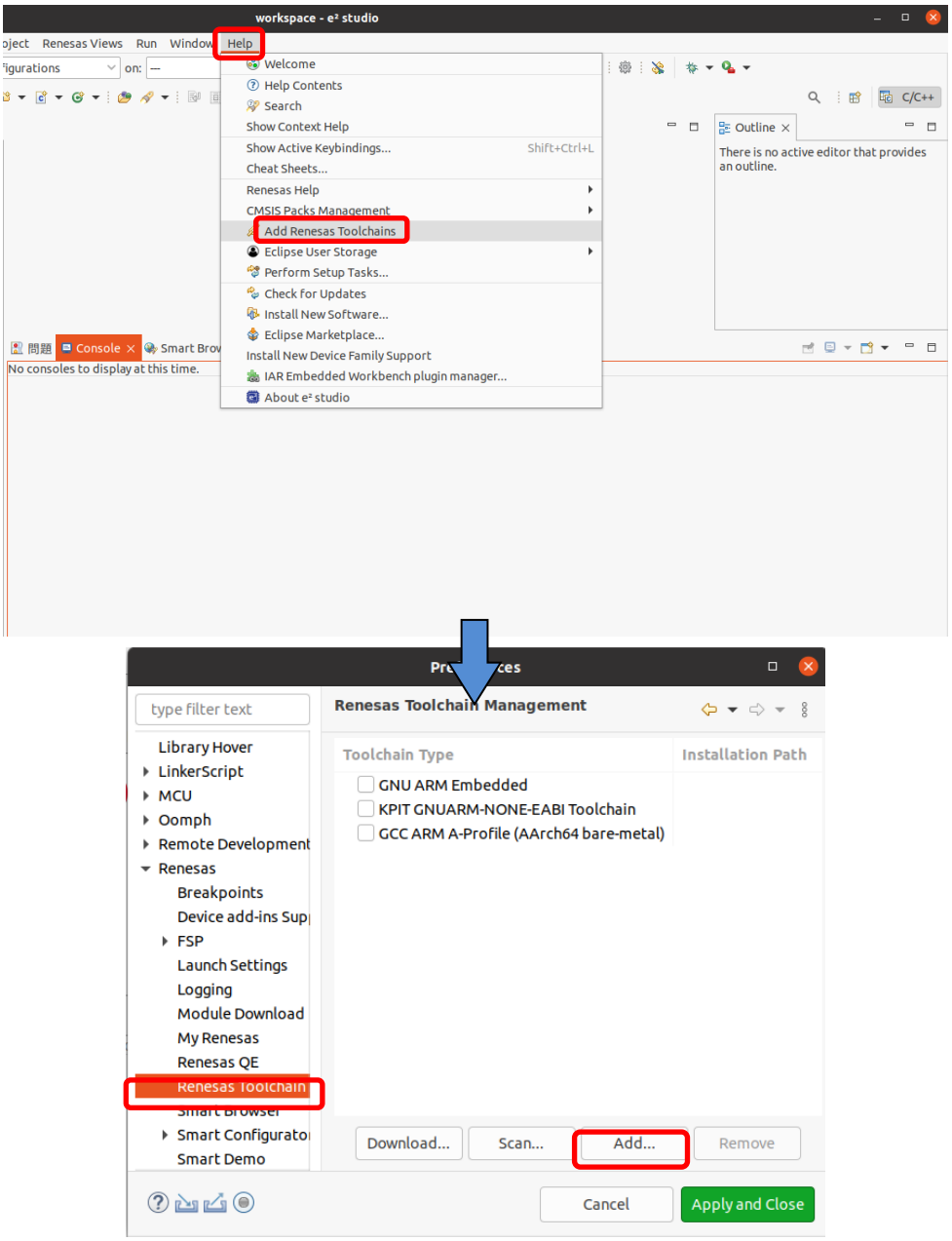


**Figure 6.28 Toolchain registration (1)**

3. Select [Browse] in Add New Toolchain window.
   Then, select /opt/gcc-arm-10.3-2021.07-x86_64-aarch64-none-elf and [Open].
   The toolchain will be registered in Preference, so click [Apply and Close].

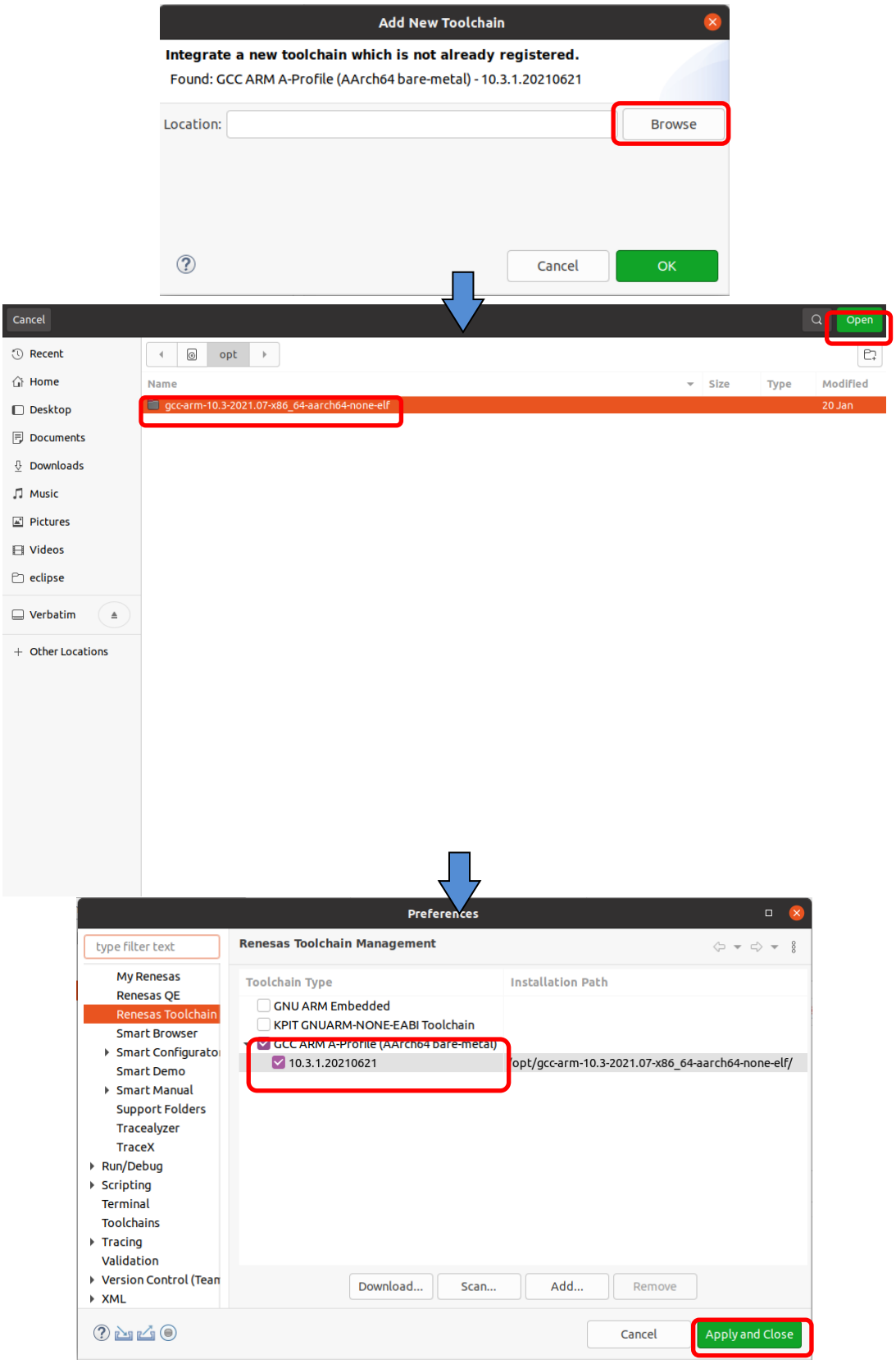


**Figure 6.29 Toolchain registration (2)**

#### 6.2.1.4 Installation of RZ/A FSP packs (ubuntu 20.4 LTS)

Install the FSP packs for use in the RZ/A3UL IPL environment.

1. Exit e² studio
2. Extract FSP Packs.
3. Copy the extracted FSP Packs (named “internal”) to the e2studio installation folder as shown below:
   "~/.local/share/Renesas/e2_studio/internal/projectgen/rz_fsp/rza/packs"

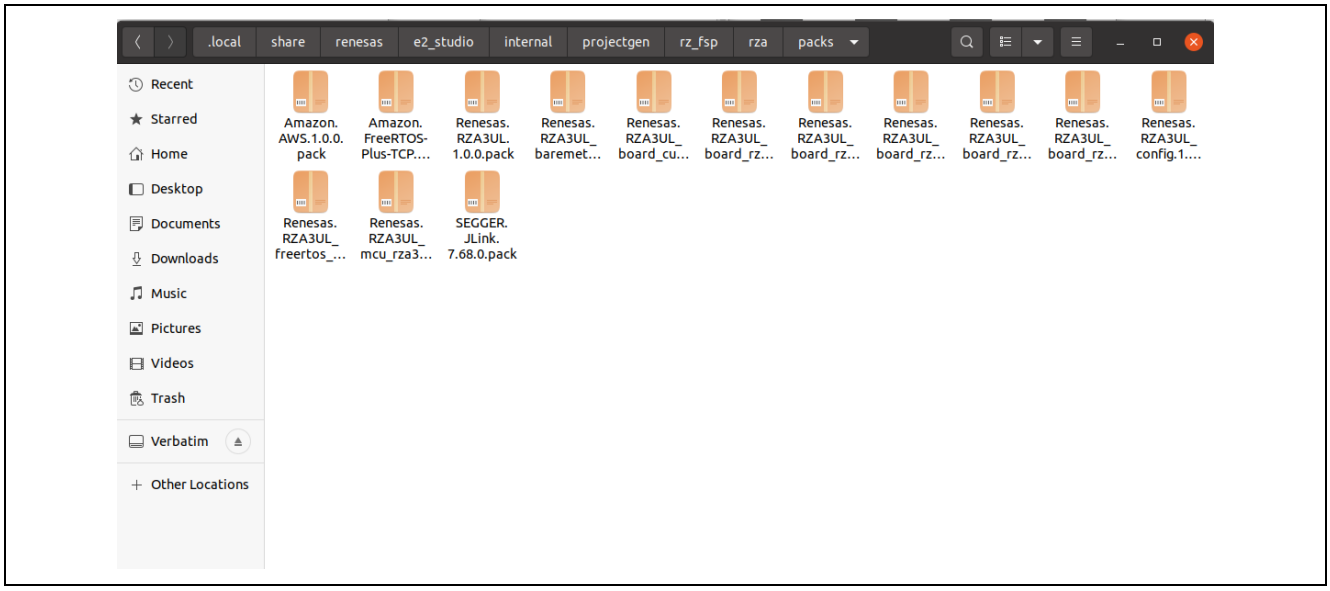


**Figure 6.30 Installed packs folder (Ubuntu 20.4 LTS)**

4. Please re-launch e2 studio
   Then packs will be reflected in the development environment.

### 6.2.2 Additional Software Installation

You must install some additional software to build the IPL in ubuntu 20.4 LTS.

1. Execute the following command to install libncurses5.

```
$ sudo apt-get install libncurses5 libncurses5:i386
```

2. Download the Segger J-Link driver file from the "Linux" -> "64-bit DEB installer" link on the following website.

   https://www.segger.com/downloads/jlink/

3. Execute the following command to install driver file.
   The command is an example when the name of the driver file is " JLink_Linux_v768_x86_64.deb ".
   Change the argument according to the file name.

```
$ sudo dpkg -i JLink_Linux_v768_x86_64.deb
```

If you get unmet dependencies error, please do the following.

```
$ sudo apt-get -f install
$ sudo dpkg -i JLink_Linux_v768_x86_64.deb
```

# 7. IPL build environment construction procedure

This chapter describes the procedure for building the IPL build environment.

The procedure is the same for both Windows 10 version and Ubuntu 20.4 LTS version.

1. Get the IPL source code from the github page shown below.
   This document shows an example when the obtained source code is saved in “C:\work\RZA3UL_IPL”.
   (Please notice that the build might fail if the folder's path is too long.)
   renesas/rza-initial-program-loader (github.com)

2. Launch e² studio.
   Click [Browse] and specify “C:\work” for the workspace on e² studio launcher

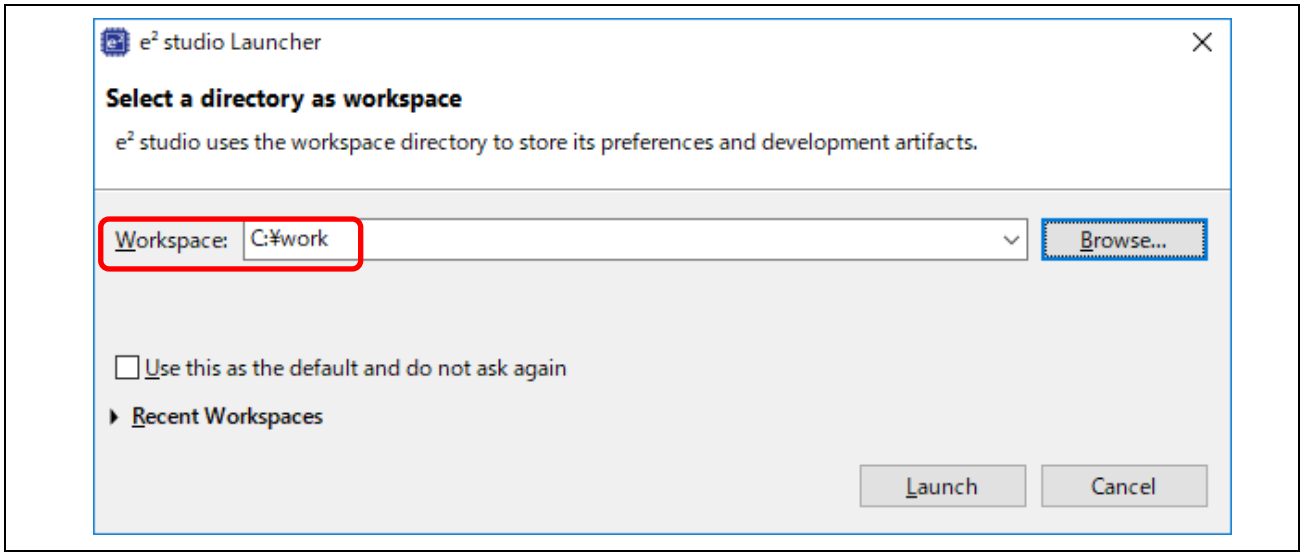


**Figure 7.1 Launch e² studio (1)**

3. In the Logging/Reporting dialog that is displayed when e² studio is started.
   Please confirm the descriptions, put a check to description that match your privacy policy, and click [Apply].

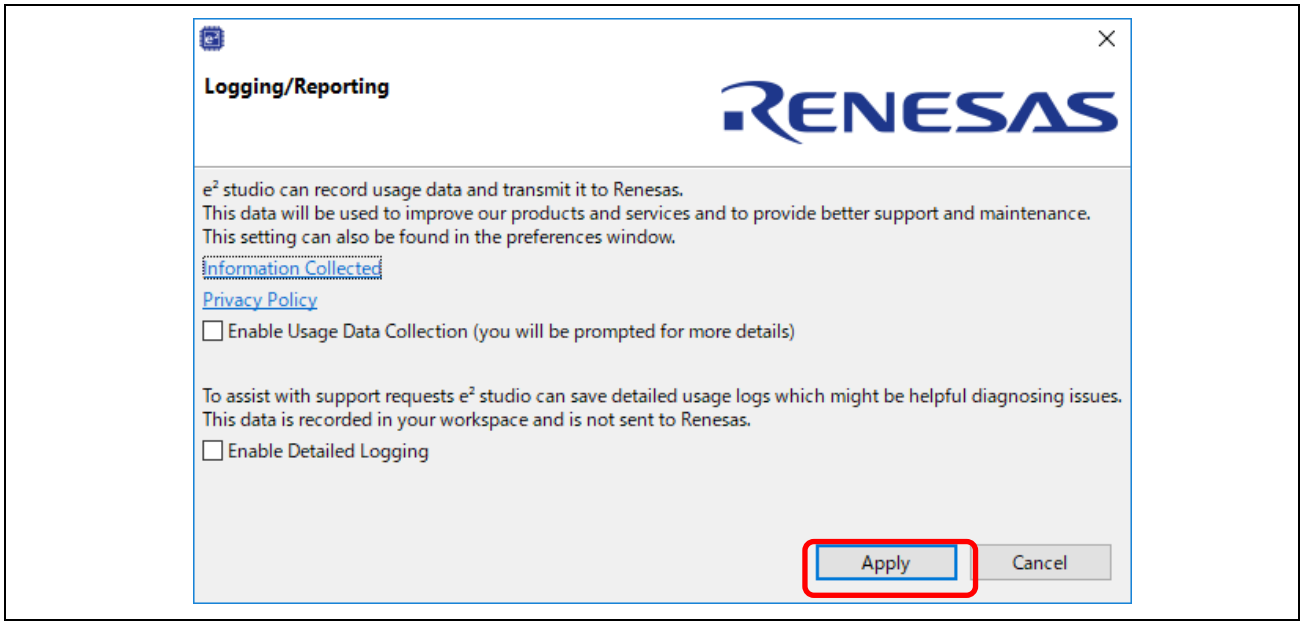


**Figure 7.2 Launch e² studio (2)**

4. When e² studio starts, click [Hide] to bring up the main window.

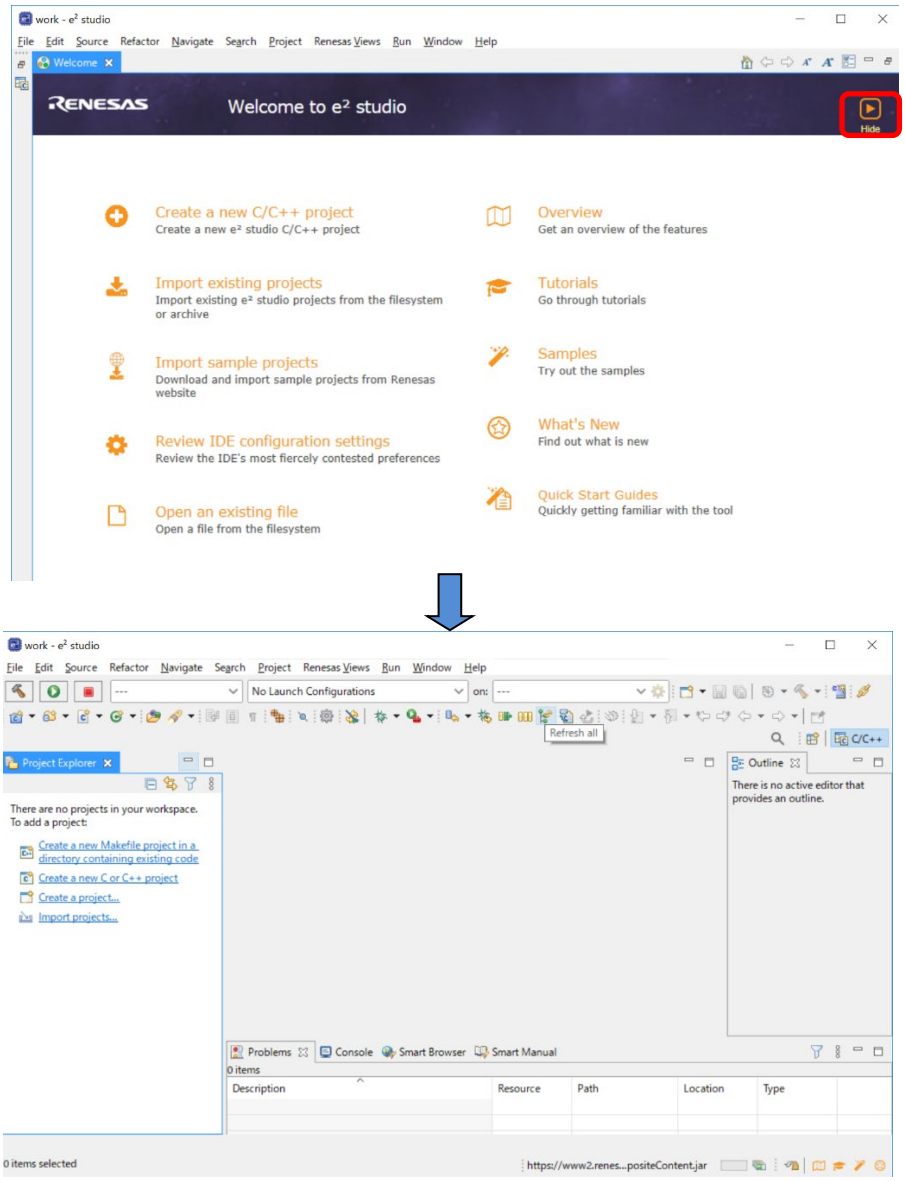


**Figure 7.3 Launch e² studio (3)**

5. Click [File] -> [New] -> [Makefile Project with Existing Code] in main window.

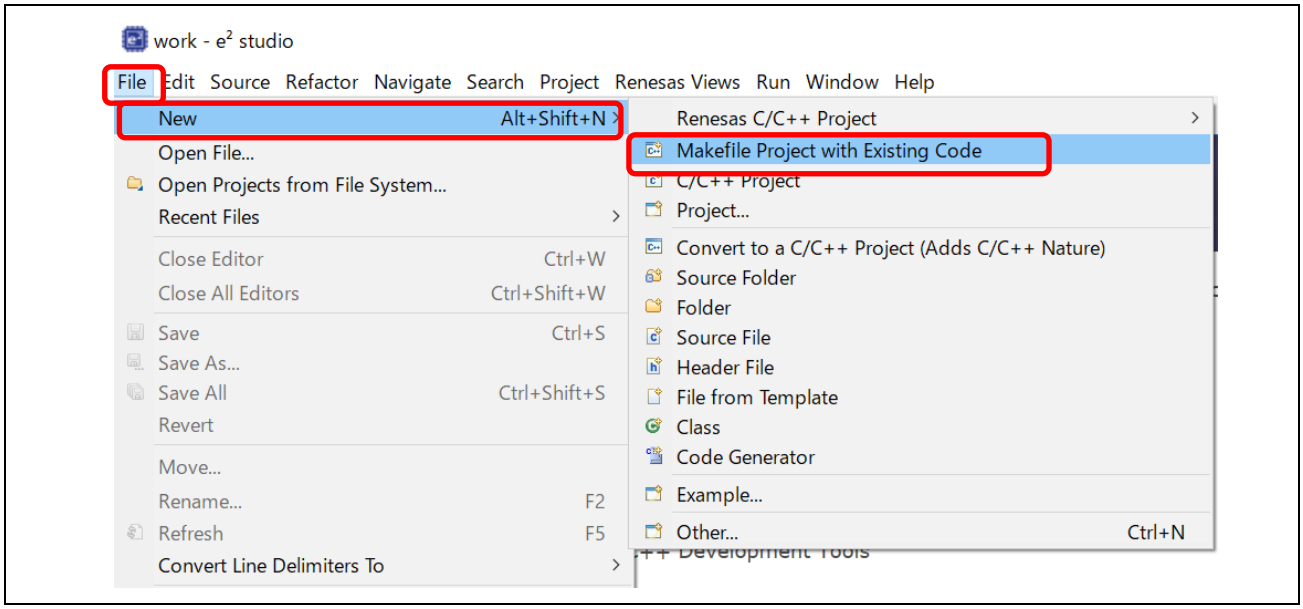


**Figure 7.4 Create project (1)**

6. Click Browse and choose “C:\work\RZA3UL_IPL”, the folder where you unzipped the IPL.
   Then the folder path will be registered in the Existing Code Location field.
7. Select “GCC for Renesas RZ (A-Profile AArch64 bare-metal)” in the Toolchain for Indexer Settings and click [Finish].

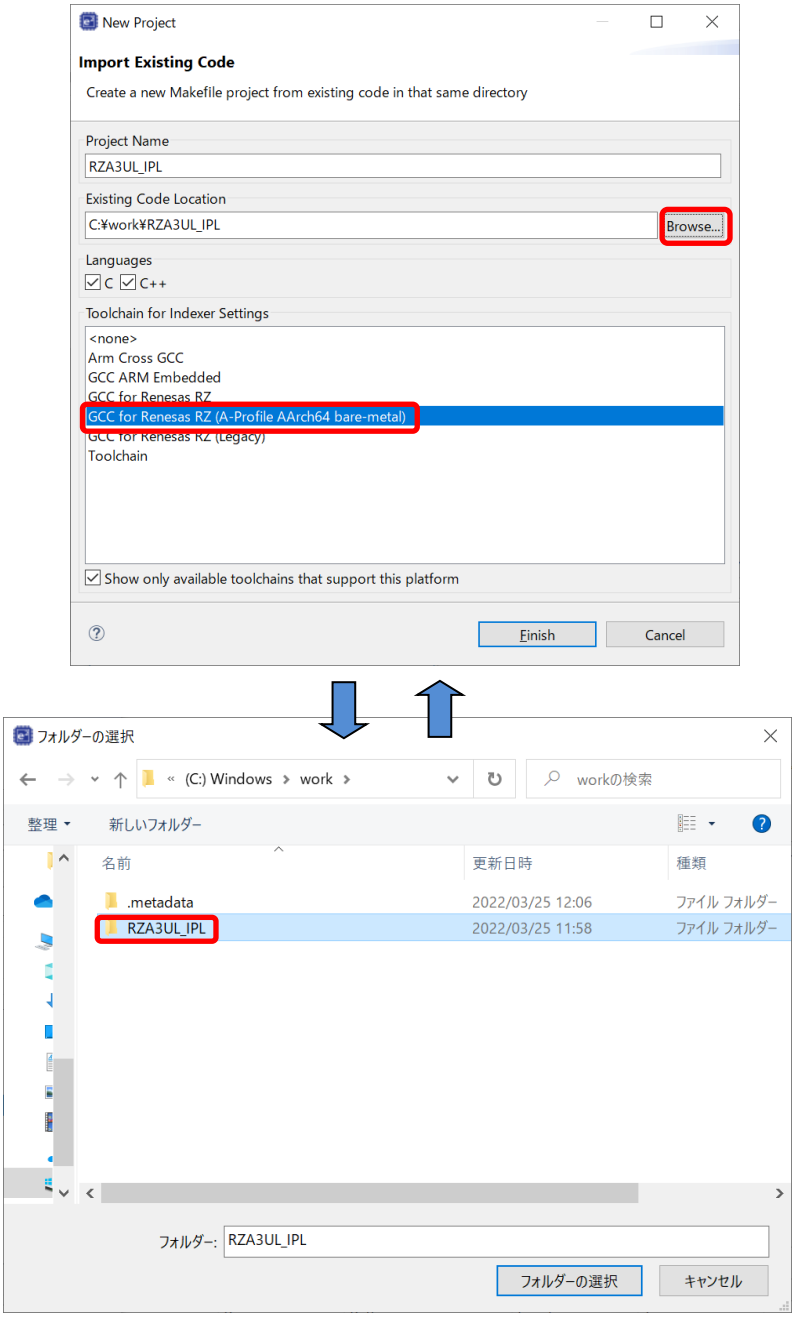


**Figure 7.5 Create project (2)**

8. Right-click the [RZA3UL_IPL] project generated in Project Explorer and click [Properties].

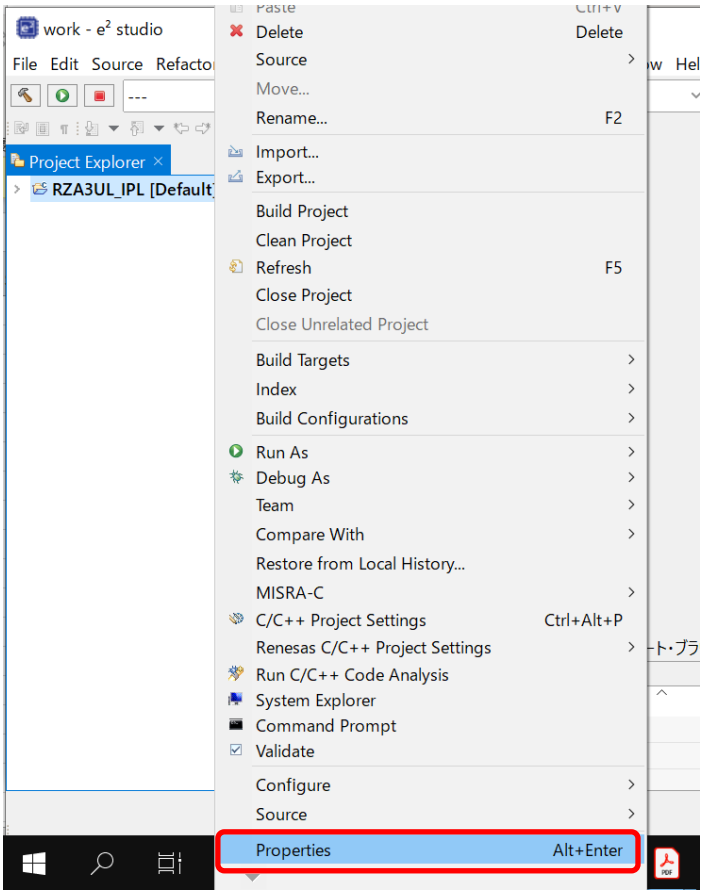


**Figure 7.6 Create project (3)**

9. Open [C / C ++ Build] -> [Tool Chain Editor] in Properties for [RZA3UL_IPL] and enter the build command.
   Select [GNU Make Builder] in “Current Builder”, then click [Apply and Close].

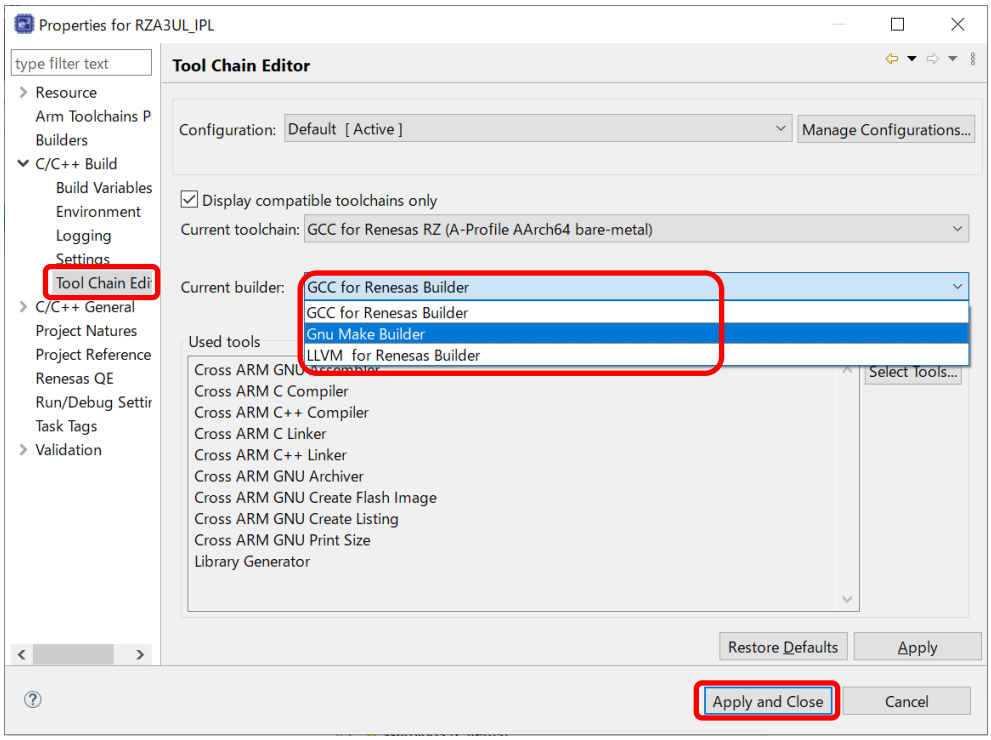


**Figure 7.7 Create project (4)**

10. Open [C/C ++ Build] -> [Behavior] in Properties for RZA3UL_IPL and enter the build command.
    Check “Use custom build arguments” and set the make option “PLAT=a3ul BOARD=a3ul_smarc_qspi” in the “Build arguments” field, so build and clean will execute with the same make option.
    If you want to use different make options for build and clean, set the make options in the Build and Clean fields respectively.
    Then click [Apply].

(Note1) Options can be added and changed.
They are explained in the Table 7.1.
(Note2) This description is the setting when the same make option is specified for Build and Clean.
If you want to use different make options for build and clean, set the make options in the Build and Clean fields respectively.

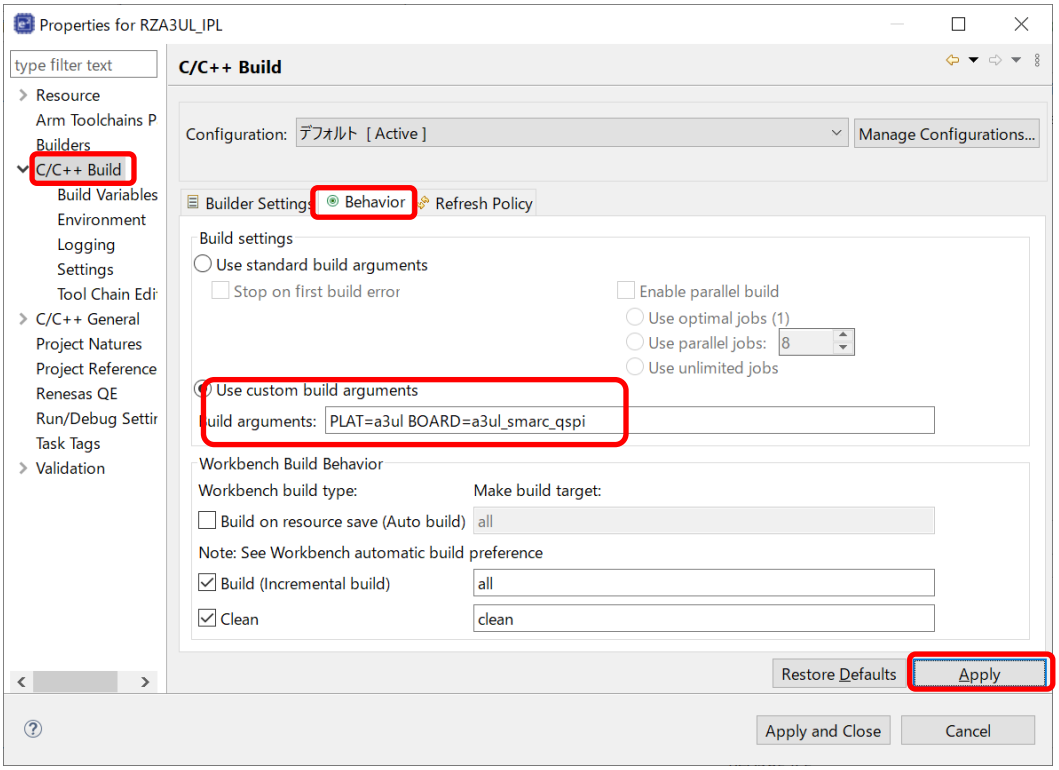


**Figure 7.8 Build Command**

**Table 7.1 Make Option**

| Name | Explanation | Setting Example |
|---|---|---|
| PLAT | Set the platform<br>This option cannot be omitted | a3ul: for RZ/A3UL, this option can’t be changed |
| BOARD | Set the board<br>This option cannot be omitted | a3ul_smarc_qspi:<br>Evaluation board Kit for RZ/A3UL MPU QSPI Edition |
| DEBUG | Select debug/release build | 1: debug build<br>0: release build<br>This option can be omitted, in which case a release build will be run. |
| FSP_BASE | Indicates the start address of the application header | 0x20020000<br>If you do not set this option, 0x20020000 is automatically set as the default value. |

11. Open [C / C ++ Build]-> [Environment] in Properties for [RZA3UL_IPL] and enter the Environment Variables.
    When you click [Add], the new variable window will appear.
    Set the environment variables of the combinations shown in the Table 7.2 (Windows10) or Table 7.3 (Ubuntu 20.4 LTS) in the Name and Value fields, then click [Apply].

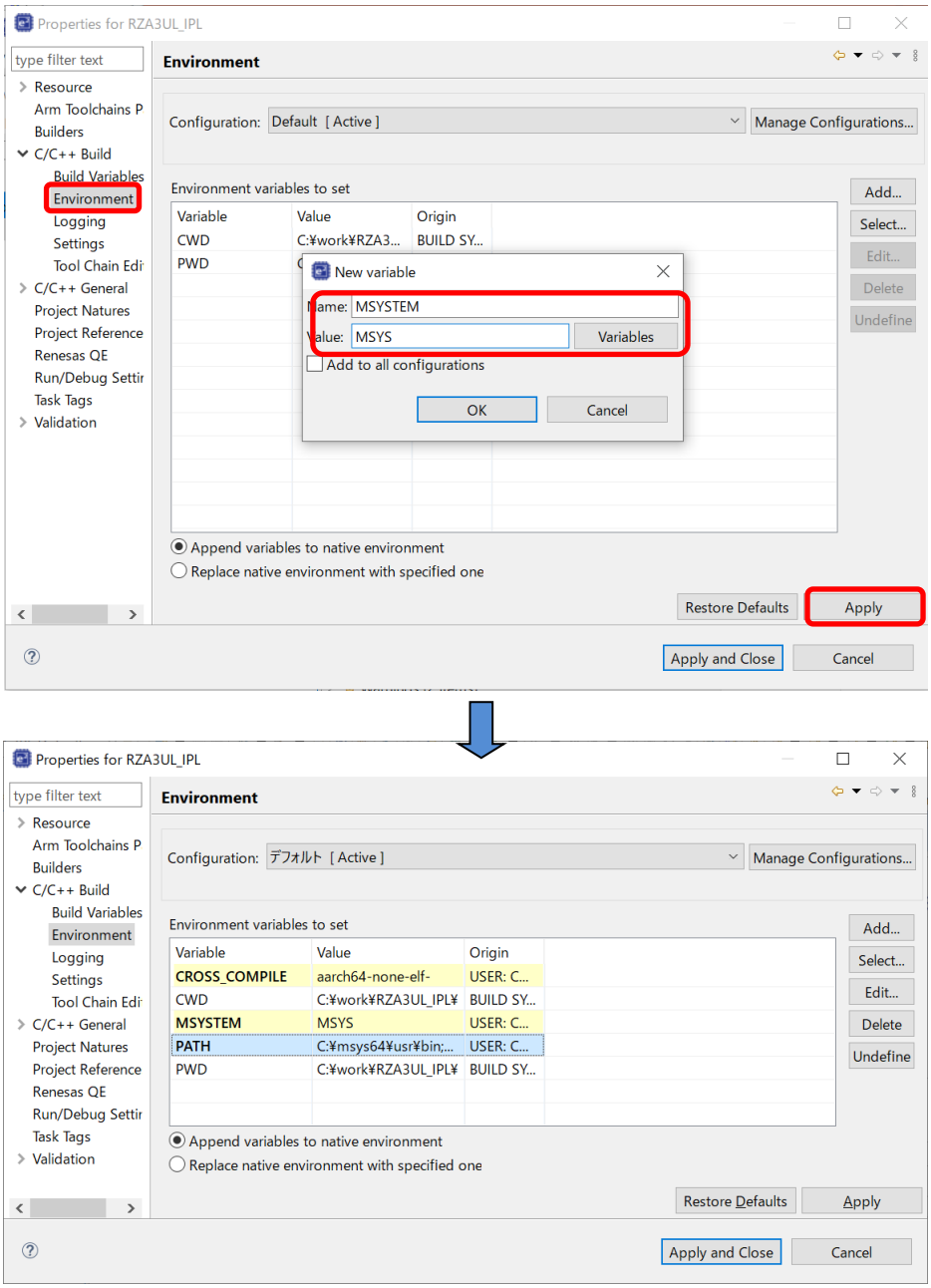


**Figure 7.9 Build Environment**

**Table 7.2 Environment Variables**

| Name | Value |
|---|---|
| CROSS_COMPILE | aarch64-none-elf- |
| MSYSTEM | MSYS |
| PATH | C:\msys64\usr\bin;C:\gcc-arm-10.3-2021.07-mingw-w64-i686-aarch64-none-elf\bin<br>(Note) |

Note : Change value to match the MSYS2 installation folder and the Toolchain unzipped folder.

**Table 7.3 Environment Variables (Ubuntu 20.4 LTS PC)**

| Name | Value |
|---|---|
| CROSS_COMPILE | aarch64-none-elf- |
| PATH | /opt/gcc-arm-10.3-2021.07-x86_64-aarch64-none-elf/bin<br>(Note) |

Note : Change value to match the Toolchain unzipped folder.

12. Right-click [RZA3UL_IPL] and click [Build Project] to run the build.
    If you want to clean it, click [Clean Project].

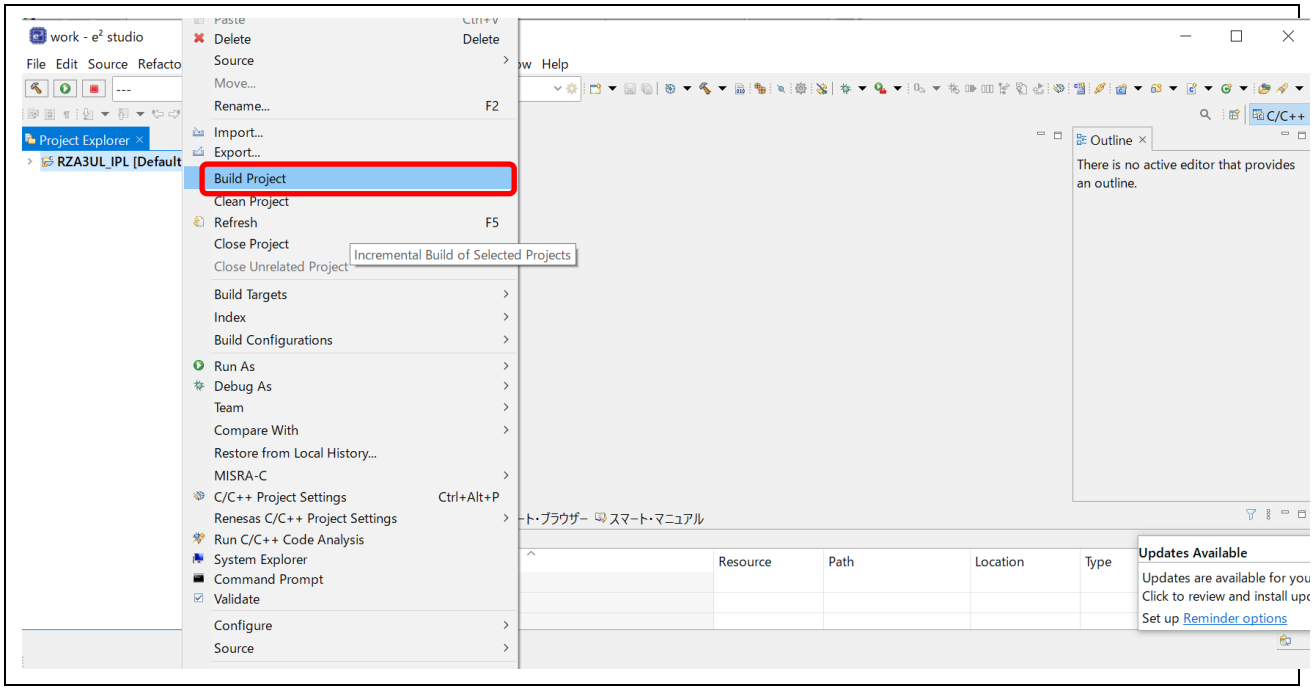


**Figure 7.10:Build Environment**

If the build is successful, the following files will be generated in “RZA3UL_IPL\build\a3ul\release”.

rza3ul_smarc_qspi_ipl.dump:
This file shows the disassembled data of the IPL code.

rza3ul_smarc_qspi_ipl.map:
Map file of IPL.

rza3ul_smarc_qspi_ipl.bin:
Raw Binary image of IPL.

rza3ul_smarc_qspi_ipl.elf:
Executable file of IPL.
It is used to download and run the IPL to On-chip RAM in e² studio.

rza3ul_smarc_qspi_ipl.srec:
Motorola S format file of IPL.
By writing to the serial flash memory, automatic boot is performed when the SMARC board is reset.
The IPL write destination address (H'2000_0000) is registered in this file itself.
When using the download function of e² studio, the registered address is referenced and written to the memory, so the offset address specified in e² studio is H'0000_0000.
When writing an IPL to the board using Flash Writer, specify the write destination offset address according to specification of it.

bl2.bin、rz_bl2_xspi.env、rz_bl2_xspi_def.c:
Temporary file when building.

# 8. Execution Environment of IPL

## 8.1 Create Debug Configuration

This chapter shows how to create a Debug configuration to run IPL.

The procedure is the same for both Windows 10 version and Ubuntu 20.4 LTS version.

1. Select [New Launch Configuration…] on the main window of e² studio as shown in Figure 8.1.

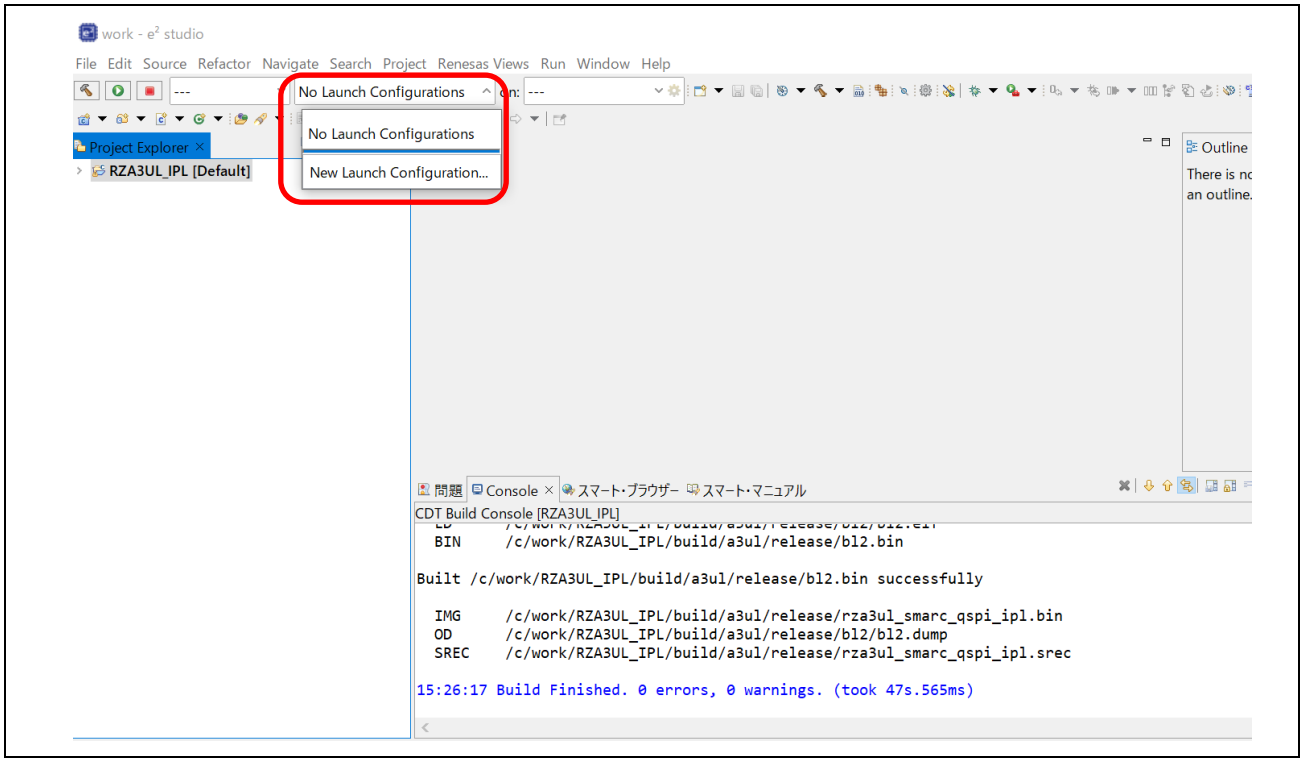


**Figure 8.1 Create Debug Configuration (1)**

2. Launch Configuration Type is brought up.
   Select [Debug] in Launch Mode field and [Renesas GDB Hardware Debugging] in Launch Configuration Type field and click [Next].

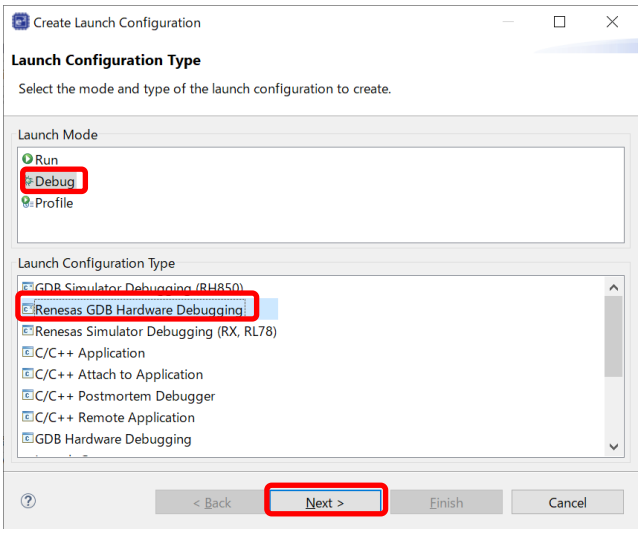


**Figure 8.2 Create Debug Configuration (2)**

3. The Renesas GDB Hardware Debugging Launch Configuration Properties will show up.
   Click [Search Project] on the Main tab and select "rza3ul_smarc_qspi_ipl.elf" on the Program Selection window.
   Furthermore, after selecting "Use Active" in [Build Configuration], click [Apply].

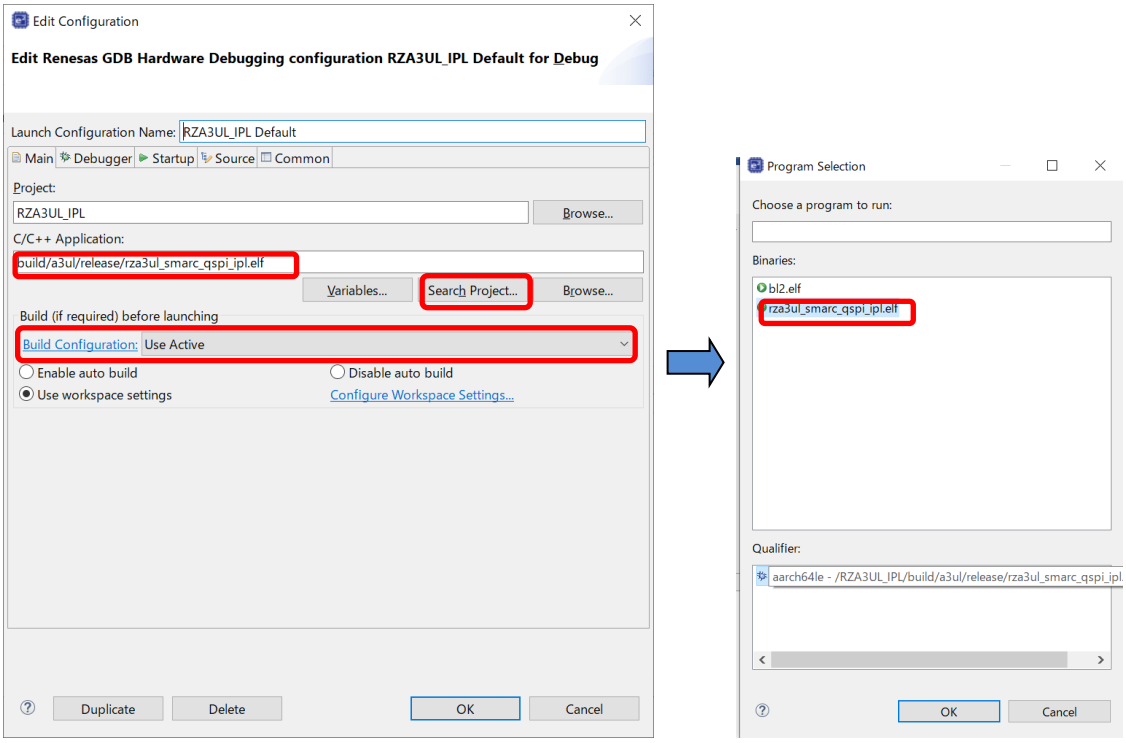


**Figure 8.3 Create Debug Configuration (3)**

4. Go to the [Debugger] tab.
   Select "J-Link ARM" in the Debug Hardware field.
   Click […] next to Target Device and select “RZ/A3UL” -> “R9A07G063U02GBG” on the Device Selection window.
   At this time, make sure that the description in the GDB Command field of the GDB Settings tab is "aarch64-elf-gdb".

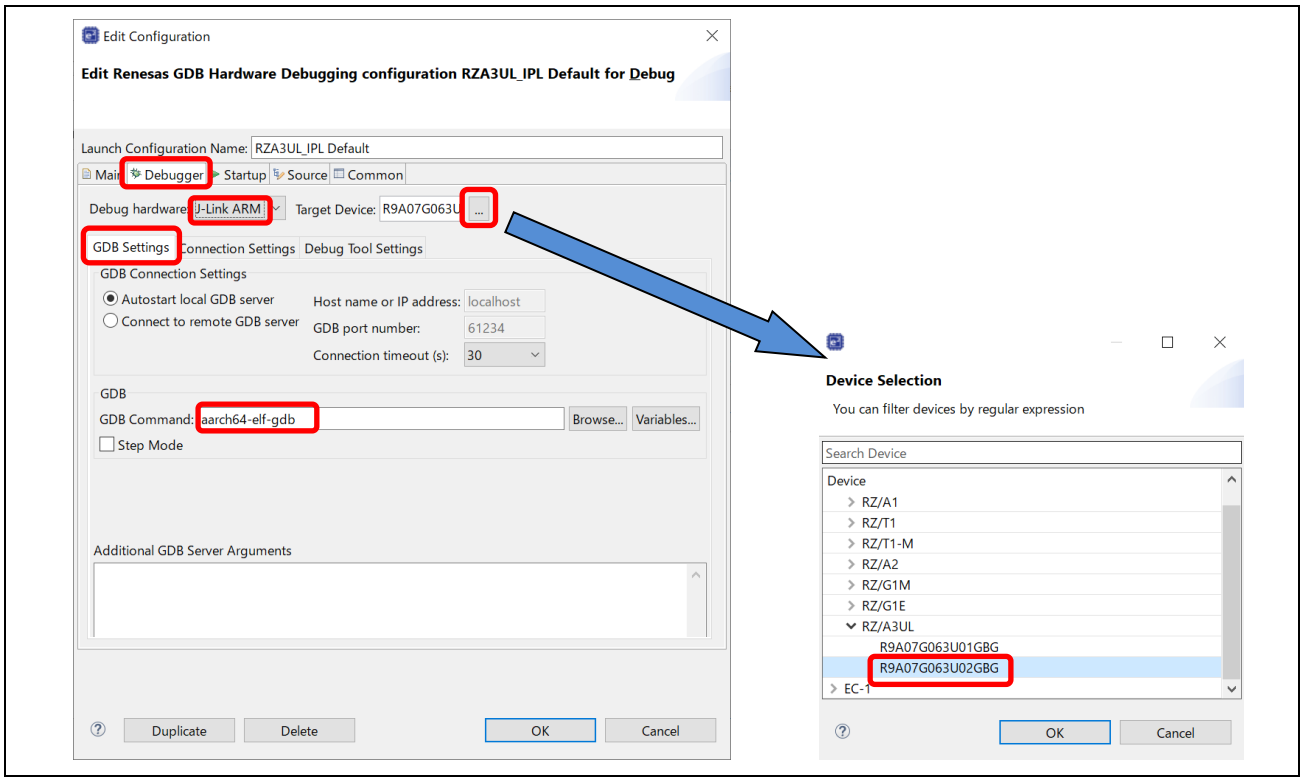


**Figure 8.4 Create Debug Configuration (4)**

5. Go to the “Debugger” -> “Connection Settings” tab and set” Speed (kHz)” to 15000 to speed up the writing of data to memory.
   In addition, set “Reset before run” to No.

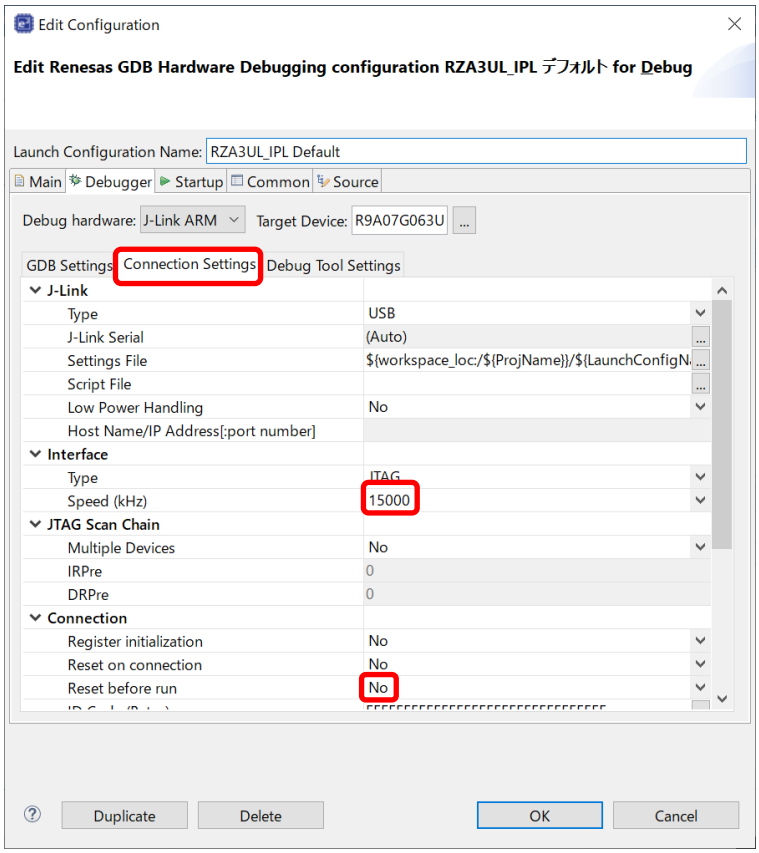


**Figure 8.5 Create Debug Configuration (5)**

6. Go to the “Debugger”-> “Debug Tool Settings” tab, set “Reset After Reload” and “Run Break Time Measurement” to No.

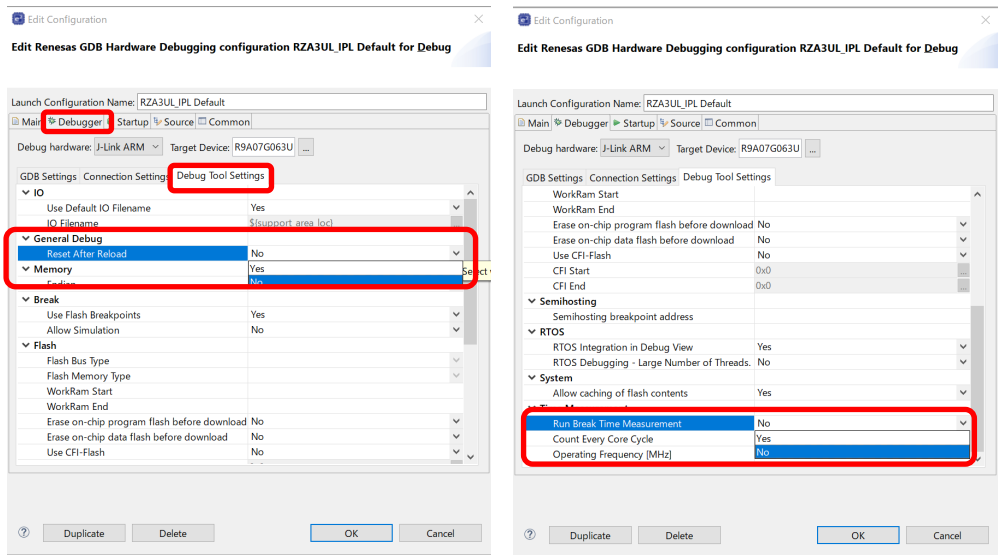


**Figure 8.6 Create Debug Configuration (6)**

7. Go to “Startup” tab
   Set “On connect” of “Program Binary” to No.

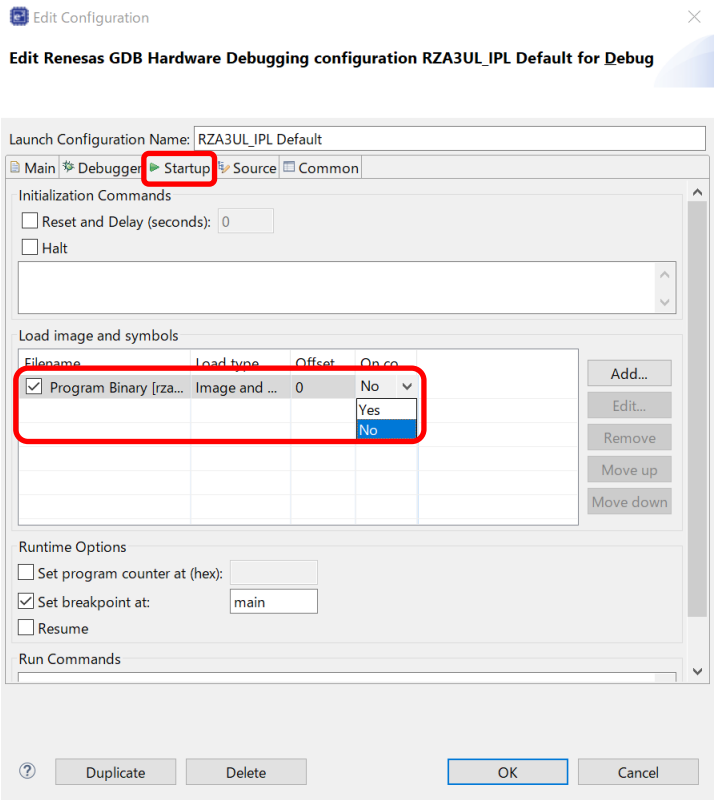


**Figure 8.7 Create Debug Configuration (7)**

8. Click [Add] to bring up the Add download module window.
   Next click Workspace, select “RZA3UL_IPL\build\a3ul\release\rza3ul_smarc_qspi_ipl.srec”, and then click [OK] to return.
   Then, rza3ul_smarc_qspi_ipl.srec is registered in Load image and symbols.

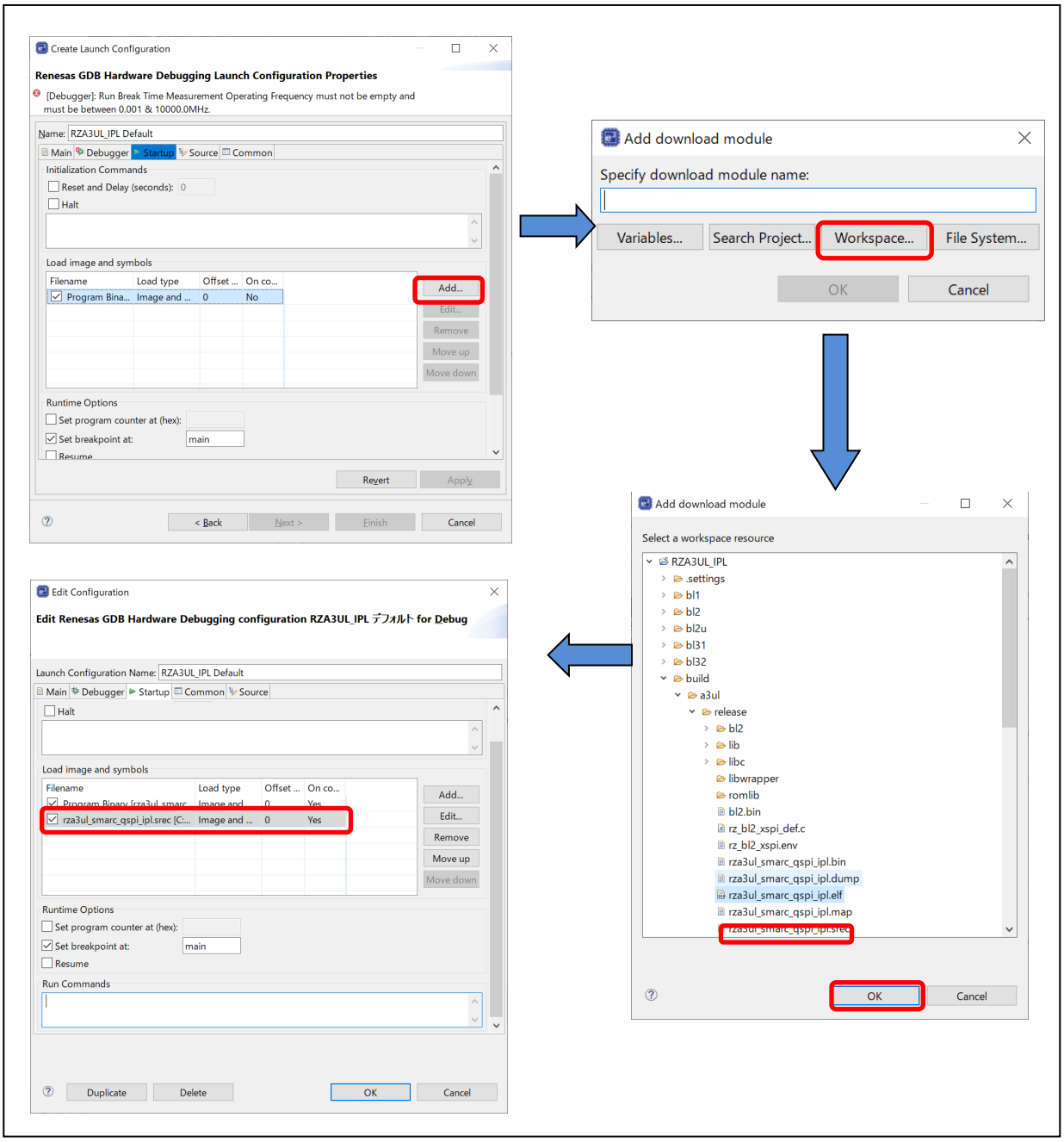


**Figure 8.8 Create Debug Configuration (8)**

9. Since the IPL write destination address H'2000_0000 is set in the registered rza3ul_smarc_qspi_ipl.srec itself, set "0" to “Offset(hex)”.
   With this setting, the IPL image will be written to H'2000_0000 at the start of debugging.
   Also, set” On connect” to No and uncheck “Set breakpoint at”.

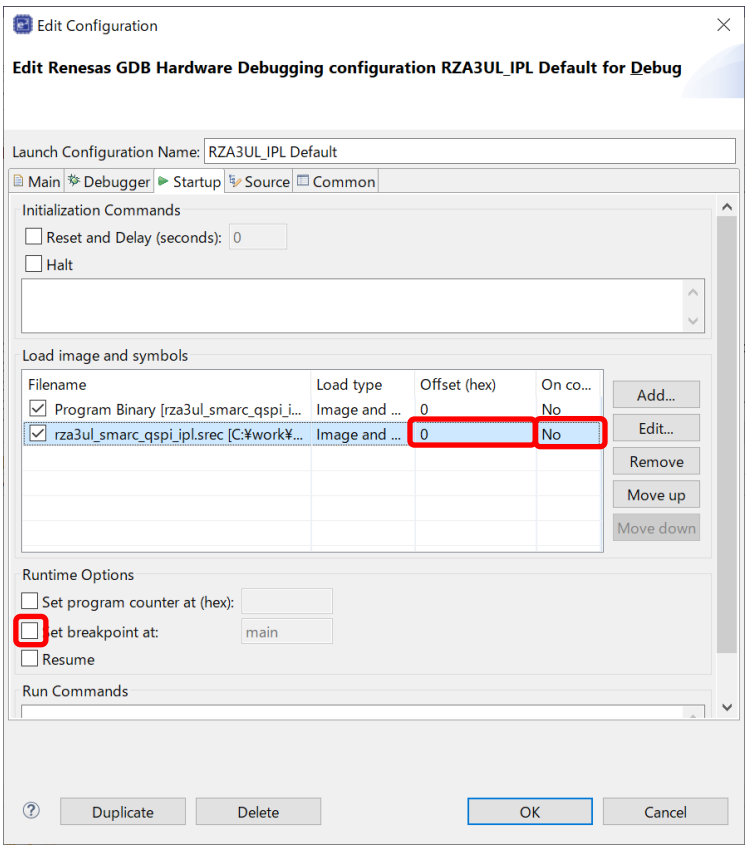


**Figure 8.9 Create Debug Configuration of Serial Flash (9)**

10. Go to the Common tab and set Save as to Share file:, and set the path to \RZA3UL_IPL.
    Also, uncheck “Launch in background”.
    After completing these settings, click [Apply] -> [Finish] to complete the settings.

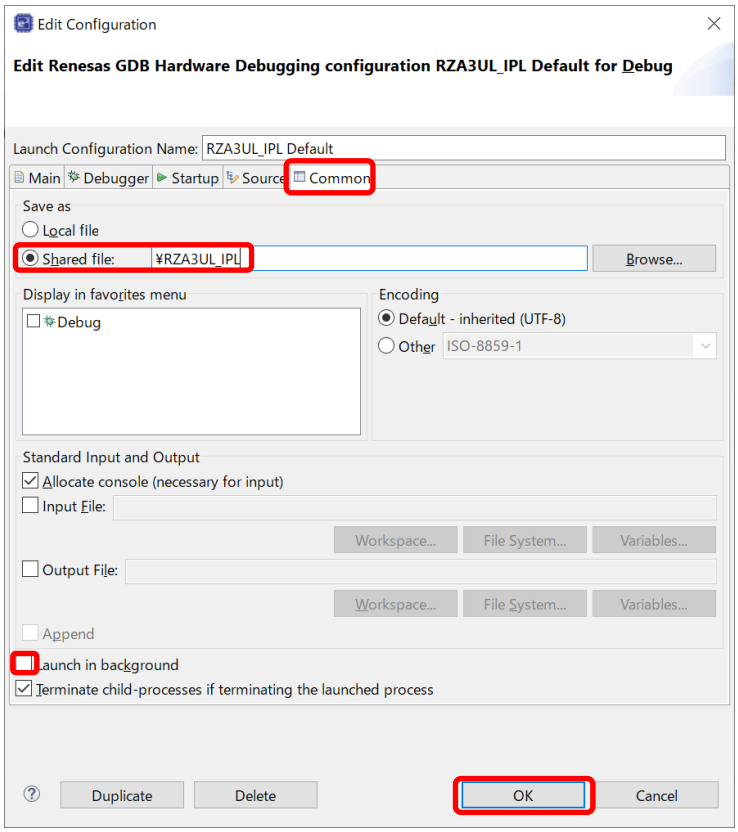


**Figure 8.10 Create Debug Configuration (10)**

## 8.2 Connection to SMARC Board

Describes how to connect to the SMARC board.

Figure 8.11 shows the appearance of the SMARC board.

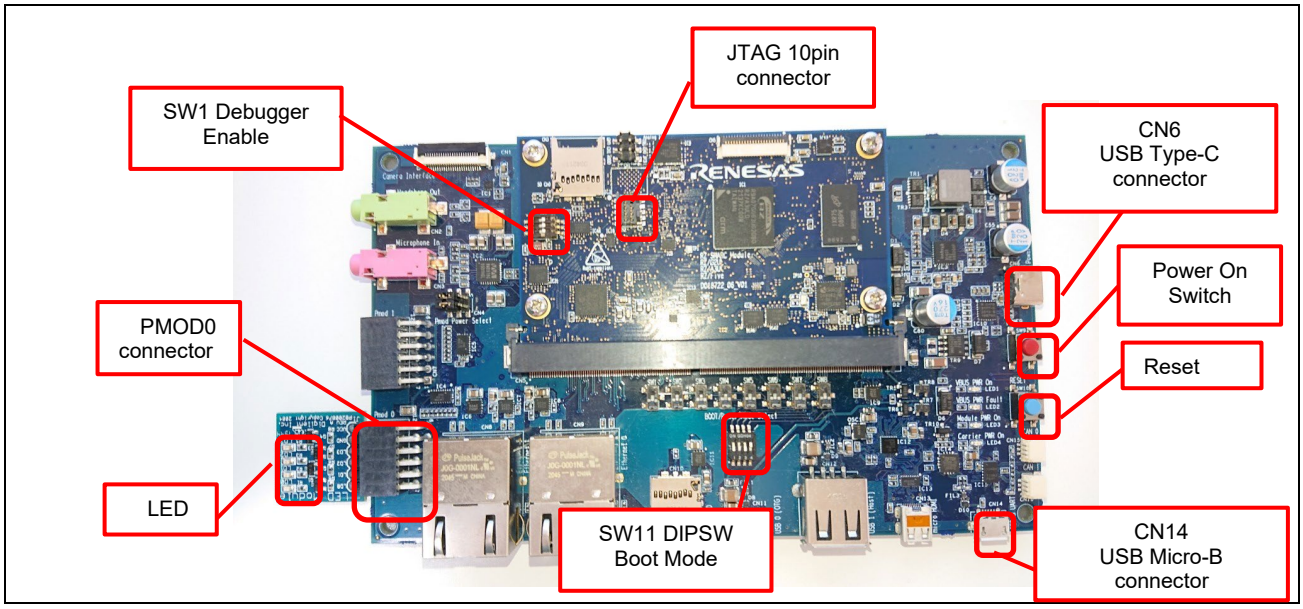


**Figure 8.11 SMARC Board**

1. Connect the LED board to the PMOD0 connector.
   As shown in Figure 8.12, connect to terminals 7 to 12.

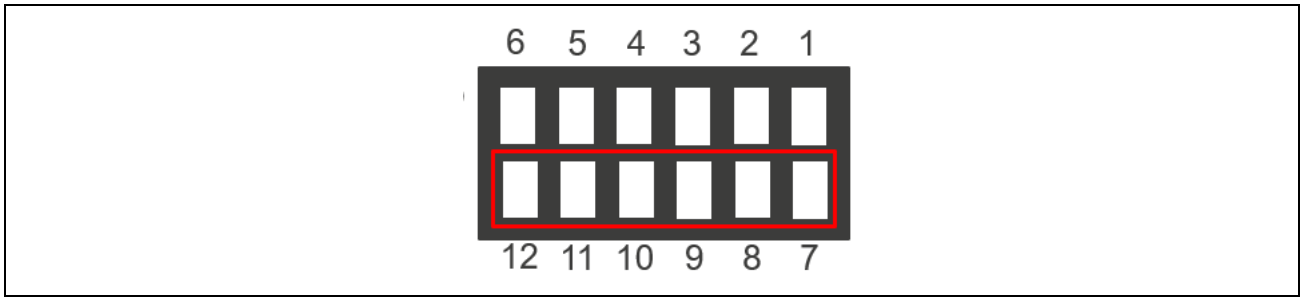


**Figure 8.12 PMOD Connector**

2. Connect CN14 and Windows 10 PC with a USB cable.
3. Connect J-Link Base 11.0 to the JTAG 10pin connector.
   In addition, connect J-Link Base 11.0 to the host PC with a USB cable.
4. Connect CN6 and USB-PD Power Charger.

5. Set SW1-1 as shown in Figure 8.13 to enable connectivity with the debugger.

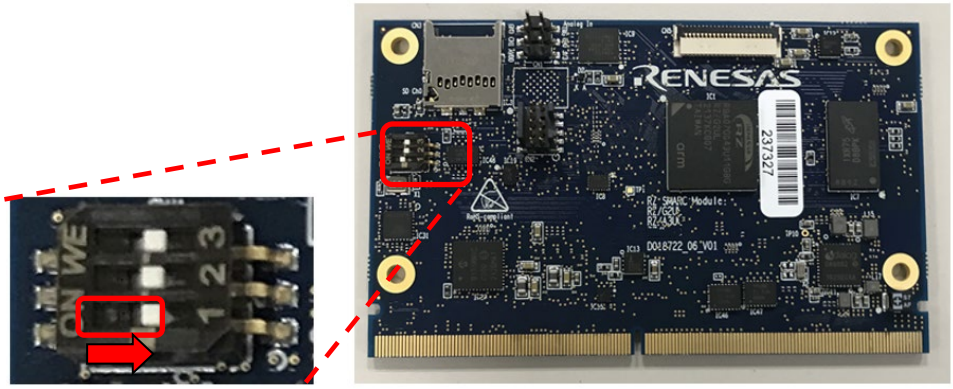


**Figure 8.13 SW1 setting**

6. Set Boot Mode to Boot Mode3 (1.8-V serial Flash memory) by setting SW11-1, SW11-2, SW11-3 as shown in Figure 8.14.

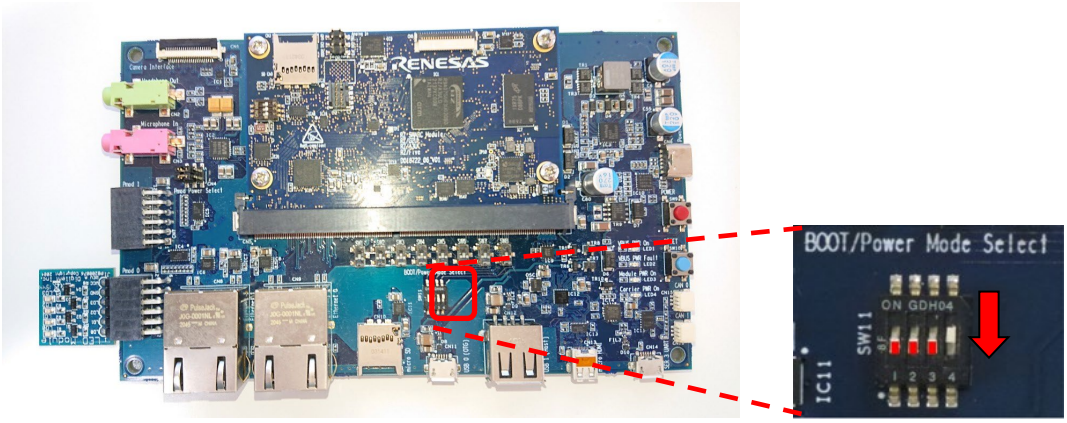


**Figure 8.14 SW11 setting**

## 8.3 Setting Serial Terminal

This section describes the procedure for setting the serial terminal to check the message output by IPL.

The procedure is the same for both Windows 10 version and Ubuntu 20.4 LTS version.

1. Set the SMARC board operating mode to Boot Mode3 (1.8-V serial Flash memory) and turn on the power.
2. Launch e² studio.
3. Click [Window] -> [Show View] -> [Other].
   The Show View window will bring up.
   Select "Terminal" there and click [Open].

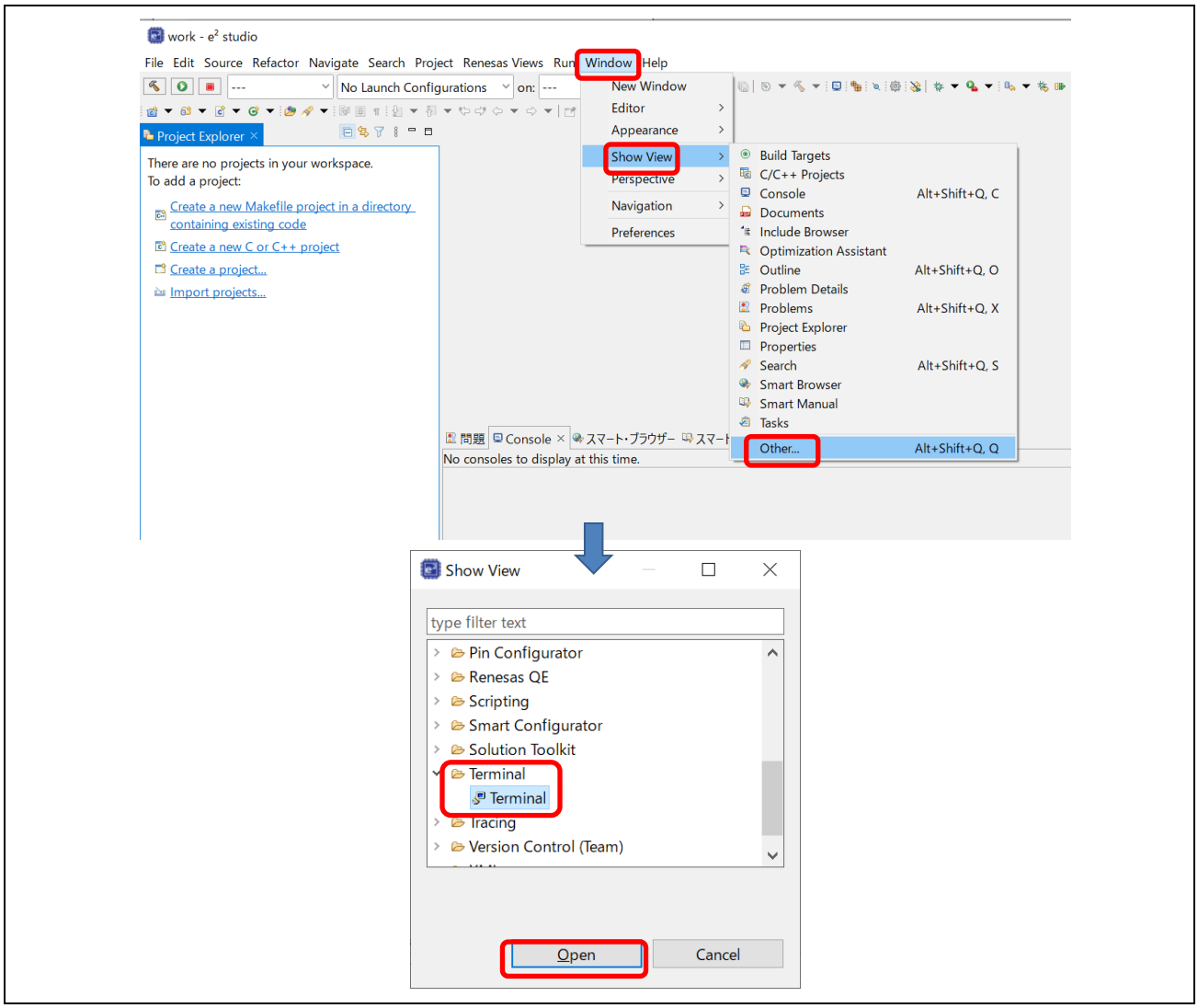


**Figure 8.15 Setting Serial Terminal (1)**

4. The Terminal window will open.
   Please click the mark.

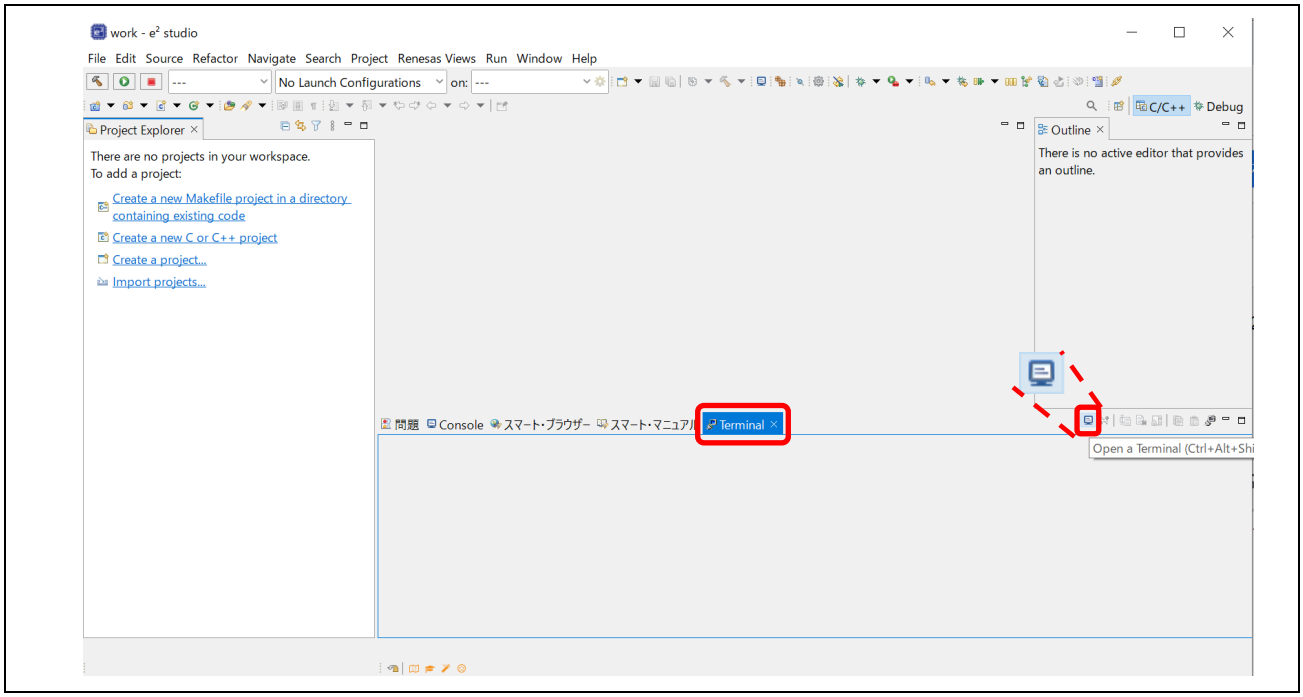


**Figure 8.16 Setting Serial Terminal (2)**

5. The Lunch Terminal window will bring up.
   Set the parameters as shown in Figure 8.17 on a Windows 10 PC and Figure 8.18 on an Ubuntu 20.4 LTS PC and click OK.
   “Serial port” is a PC environment-dependent port.
   Please set according to your environment.

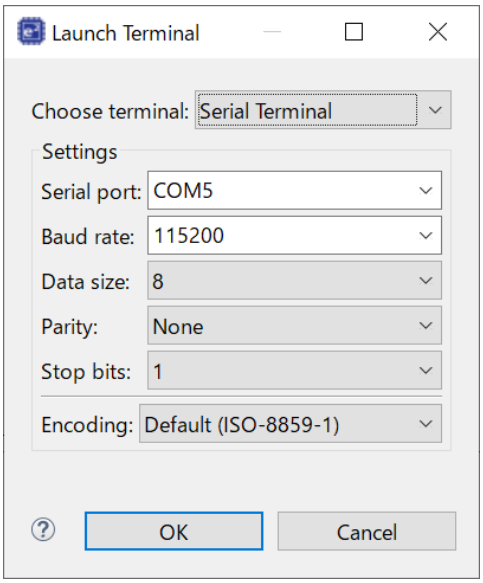


**Figure 8.17 Parameter of Serial Terminal (Windows10)**

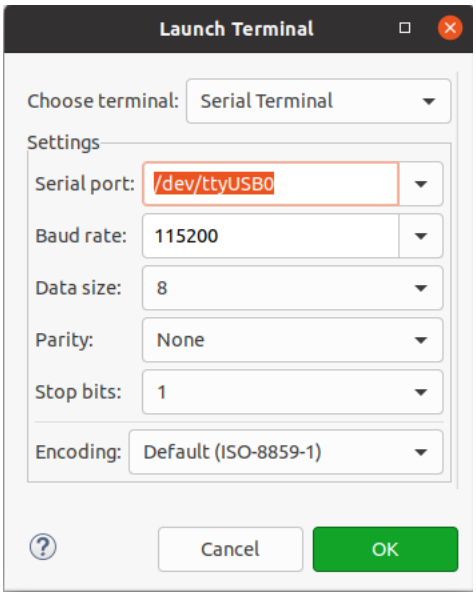


**Figure 8.18 Parameter of Serial Terminal (Ubuntu 20.4 LTS)**

6. Start the Serial Terminal.

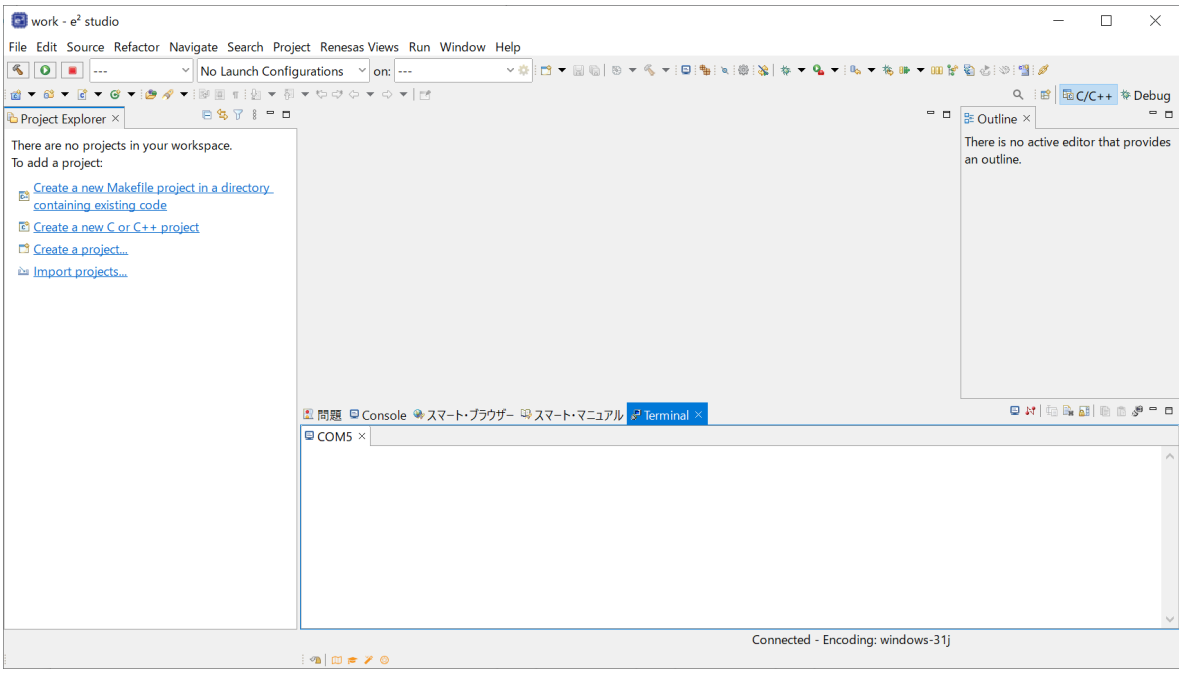


**Figure 8.19 Setting Serial Terminal (3)**

## 8.4 Execution procedure of IPL by Serial Flash boot

The procedure for writing the IPL to the serial flash memory and executing the IPL is shown below.

1. As the FSP application must be written to the serial flash memory to confirm the boot operation by IPL, if the FSP application is not written to the serial flash memory, write it.
   For the write procedure, refer to 9.2 Procedure for writing an FSP application to serial flash memory.
2. Set the SMARC board operating mode to Boot Mode3 (1.8-V serial Flash memory) and turn on the power.
3. Launch e² studio and click ⚙ mark to the right of the Debug Configuration generated in 8.1 Create Debug Configuration (hereinafter referred to as RZA3UL_IPL Default configuration) to bring up the setting window.

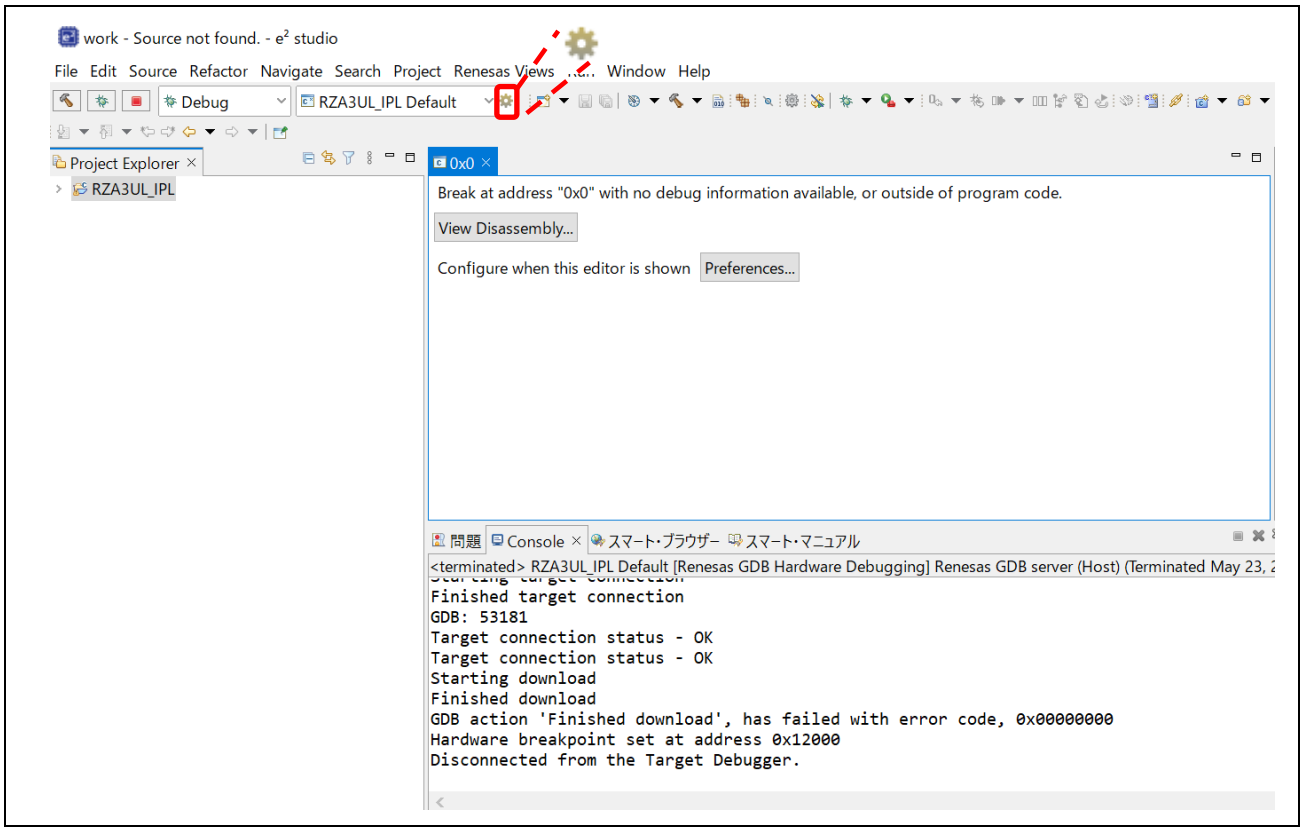


**Figure 8.20 Change Debug Configuration of Serial Flash (1)**

4. Go to the “Startup” tab and set “On connect” for “rza3ul_smarc_qspi_ipl.srec” to Yes.
   Also, set “On connect” in “Program Binary” to No.
   Then, click [OK] to complete the setting.

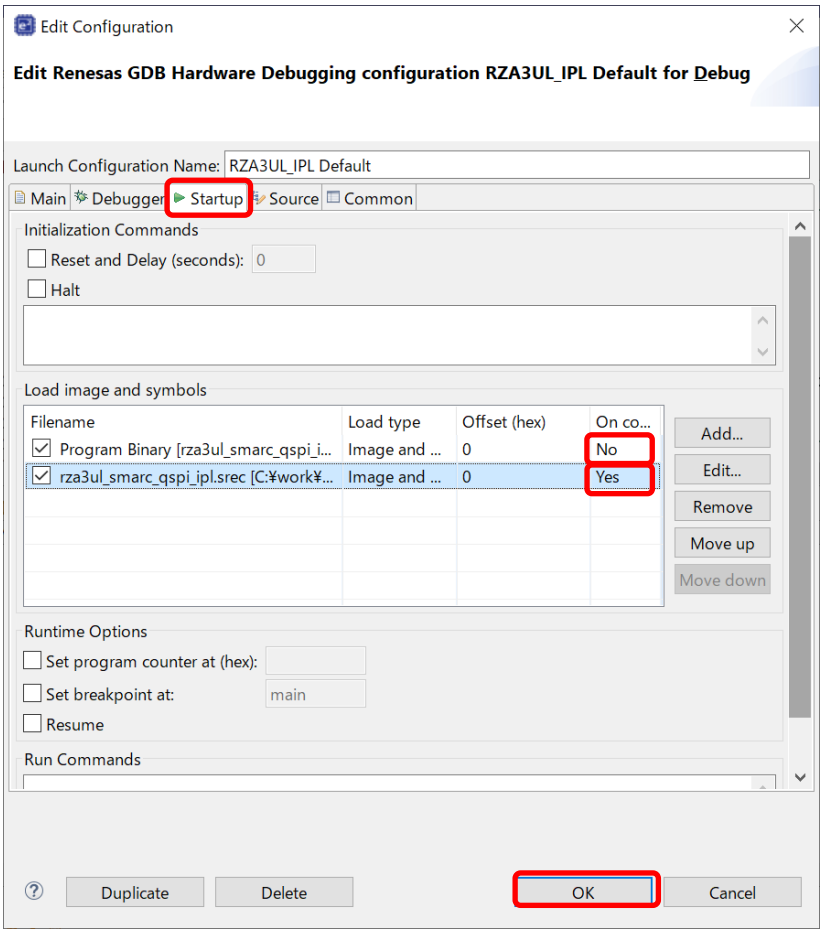


**Figure 8.21 Change Debug Configuration of Serial Flash (2)**

5. Make sure the RZA3UL_IPL Default configuration is selected on the e² studio window and click mark to launch the program.

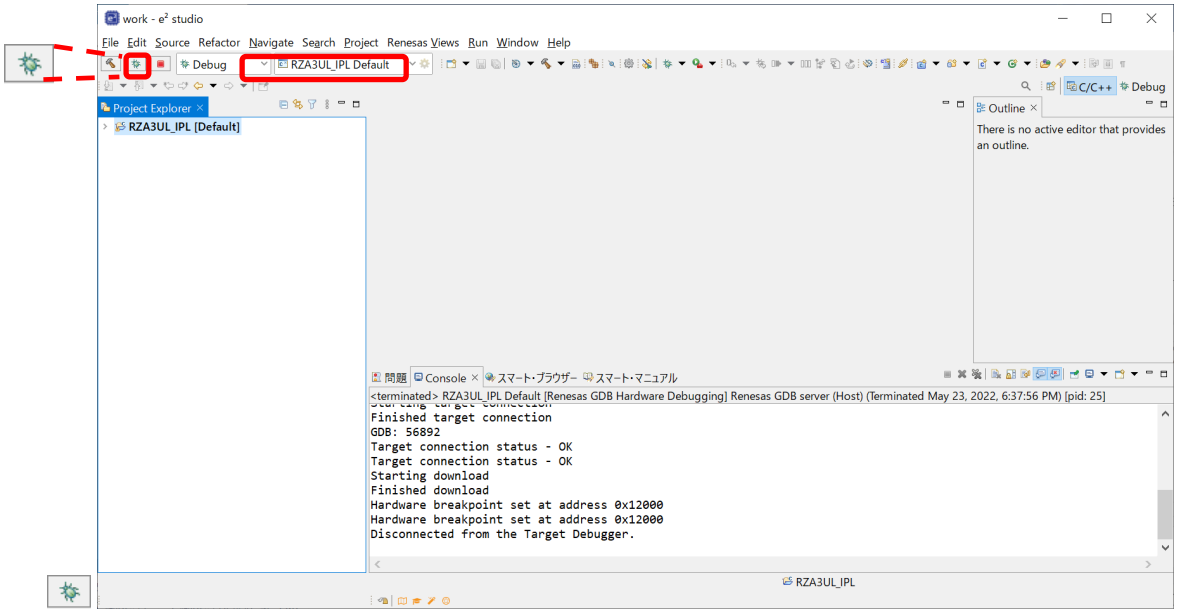


**Figure 8.22 Execution of IPL on serial flash memory (1)**

6. Writing to the serial flash memory is performed.
   After the write is complete, the perspective for the debug control will be launched.

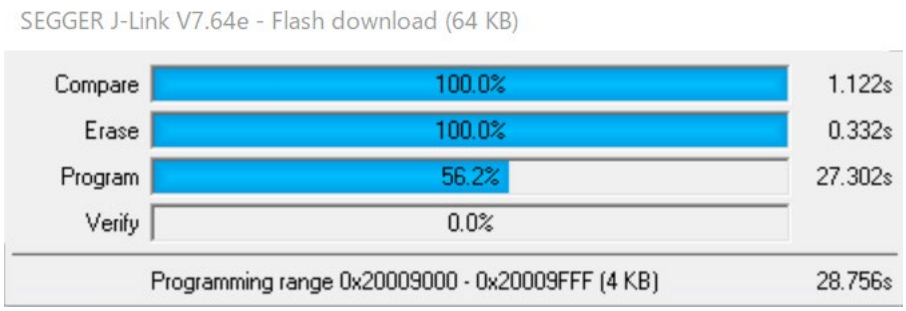


**Figure 8.23 Execution of IPL on serial flash memory (2)**

7. If Confirm Perspective Switch is displayed during the launch of the perspective, check Remember my decision and select Switch.

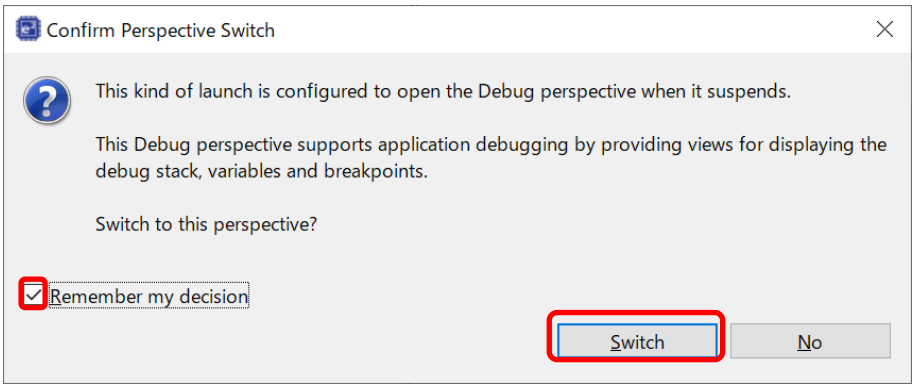


**Figure 8.24 Execution of IPL on serial flash memory (3)**

8. After writing to the serial flash, click ■ mark in the perspective to disconnect from the SMARC board.
   After that, press the RESET button on the SMARC board to execute the IPL.
   At this time, the log shown in Figure 8.26 is displayed on Terminal.
   In addition, the program written in the serial flash memory starts and the LED of the SMARC board blinks as shown in Figure 8.27.

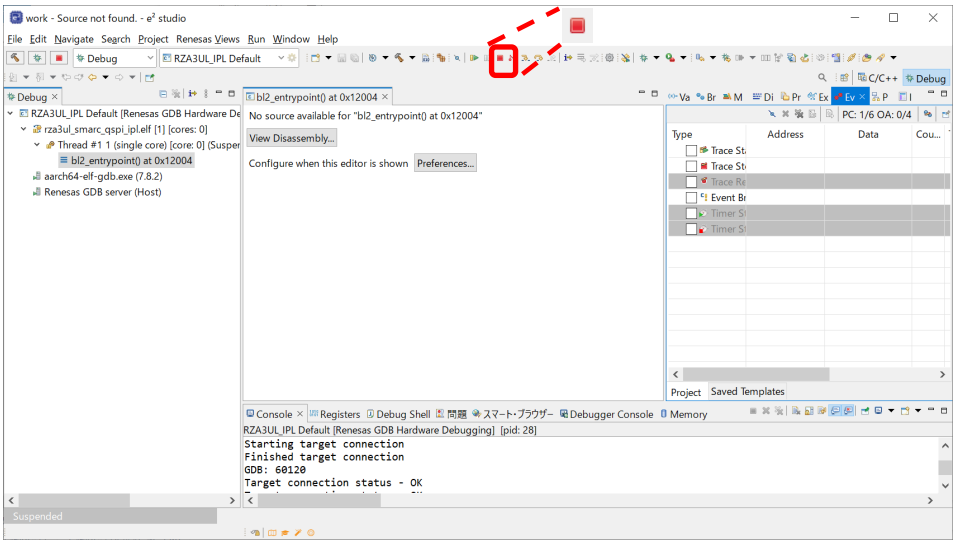


**Figure 8.25 Execution of IPL (6)**

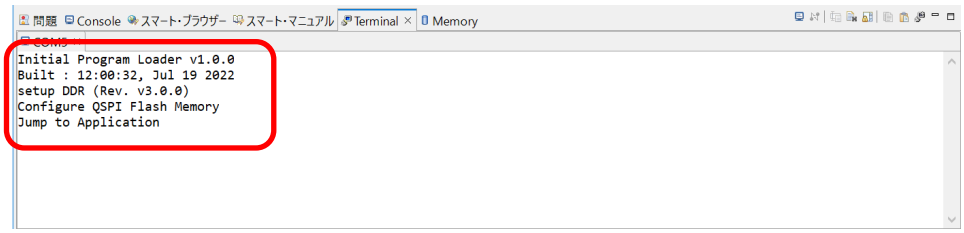


**Figure 8.26 IPL execution log**

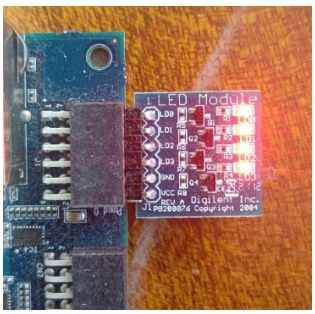


**Figure 8.27 LED blinking**

# 9. Application Example

This chapter shows application examples for applying IPL to various environments.

## 9.1 Execution procedure of IPL by Internal SRAM download

This chapter describes the procedure for writing and executing the IPL to the On-chip RAM for the purpose of facilitating the operation check when customizing the IPL.

1. As the FSP application must be written to the serial flash memory to confirm the boot operation by IPL, if the FSP application is not written to the serial flash memory, write it.
   For the write procedure, refer to 9.2 Procedure for writing an FSP application to serial flash memory.
2. Set the SMARC board operating mode to Boot Mode3 (1.8-V serial Flash memory) and turn on the power.
3. Launch e² studio and click ⚙ mark to the right of the RZA3UL_IPL Default configuration to bring up the setting window.

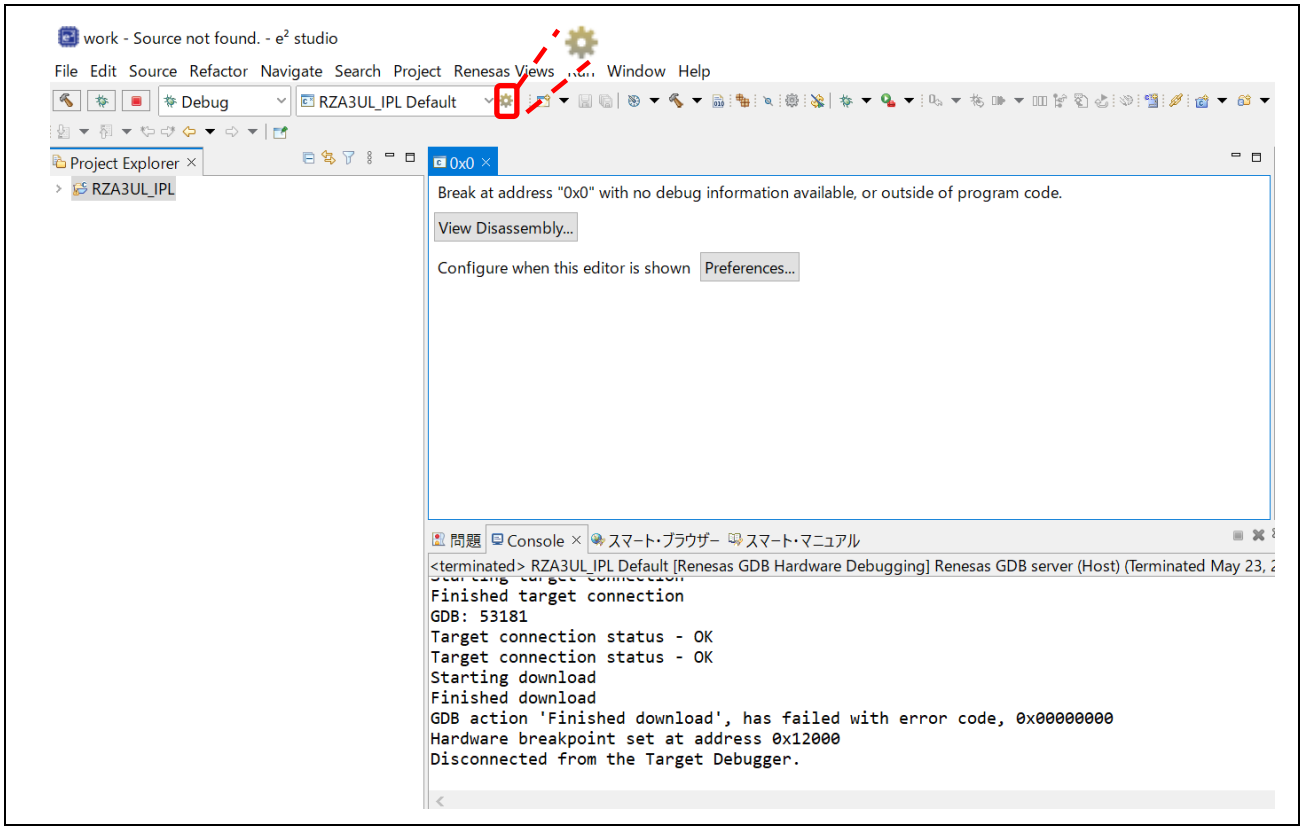


**Figure 9.1 Change Debug Configuration of On-chip RAM (1)**

4. Go to the “Startup” tab and set “On connect” for “Program Binary.elf” to Yes.
   Also, set “On connect” in “rza3ul_smarc_qspi_ipl.srec” to No.
   Then, click [OK] to complete the setting.

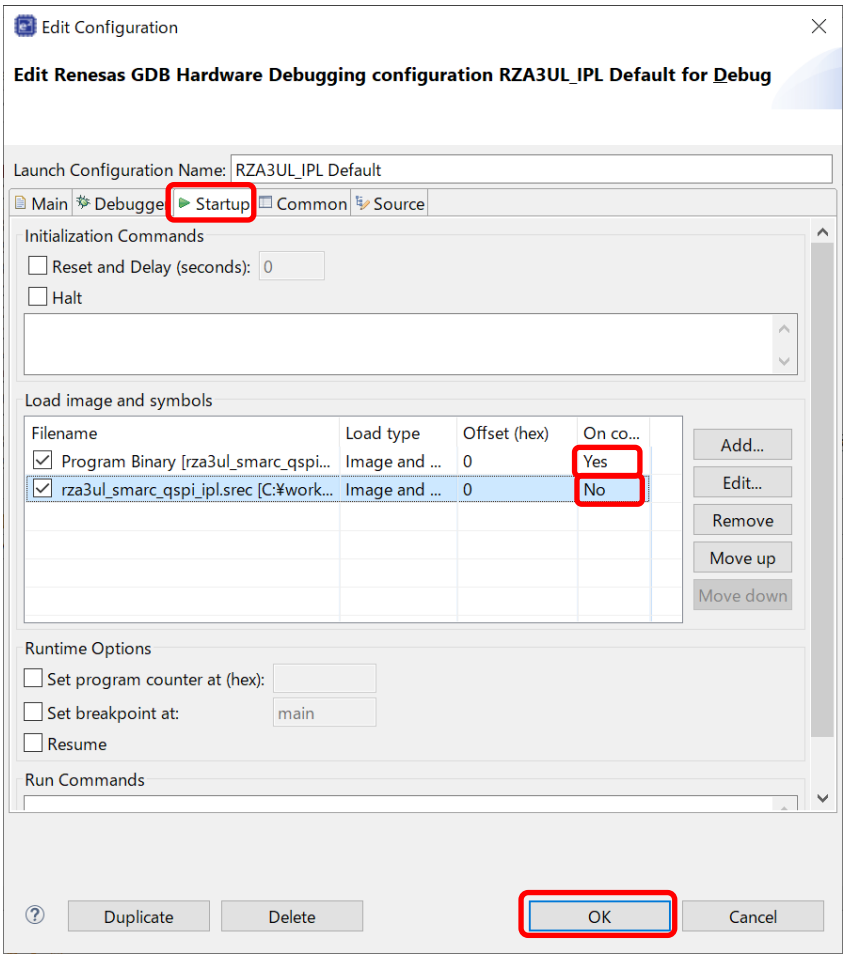


**Figure 9.2 Change Debug Configuration of On-chip RAM (2)**

5. Make sure the RZA3UL_IPL Default configuration is selected on the e² studio window and click mark to launch the program.
   This will bring up a perspective for the debug control

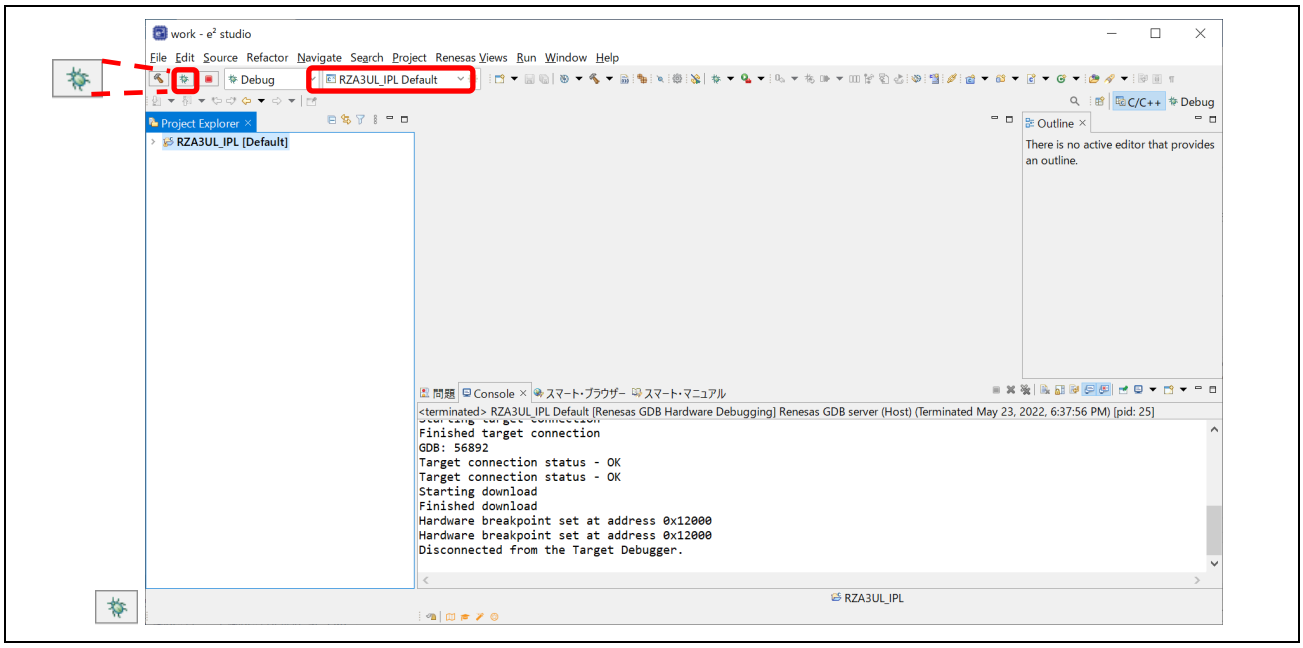


**Figure 9.3 Execution of IPL on On-chip RAM (1)**

6. Click mark in the perspective to perform the IPL.
   The IPL will start executing and the log shown in Figure 9.5 will be displayed in the terminal.
   At this time, the FSP application written on the serial flash memory starts, and the LED of the SMARC board blinks as shown in Figure 9.6.

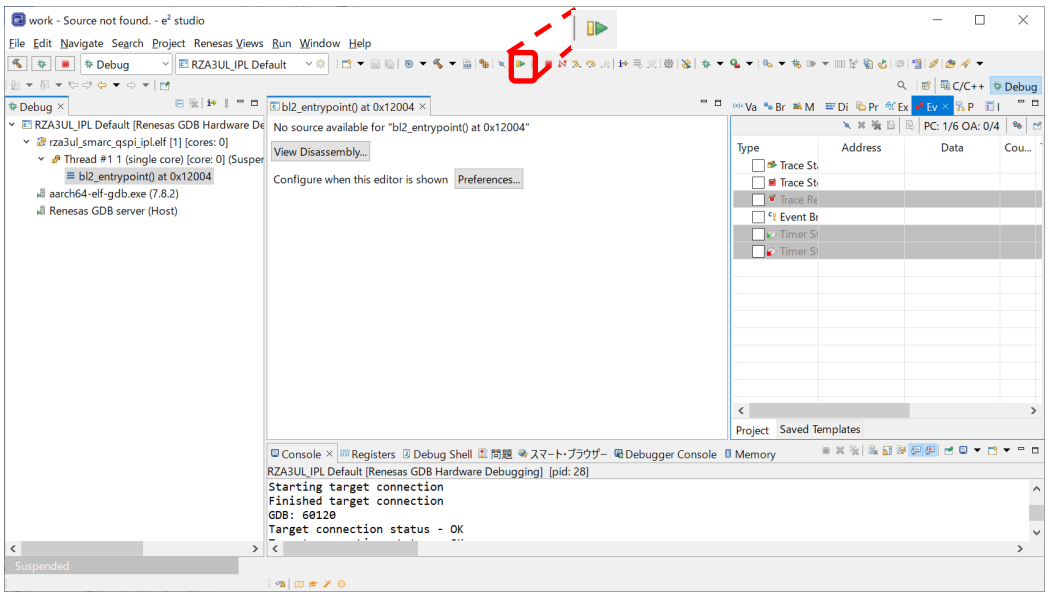


**Figure 9.4 Execution of IPL (4)**

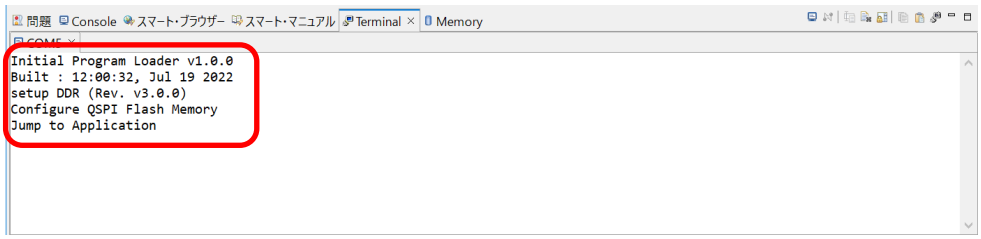


**Figure 9.5 IPL execution log**

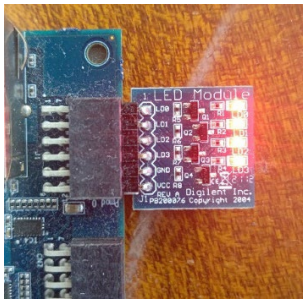


**Figure 9.6 LED blinking**

## 9.2 Procedure for writing an FSP application to serial flash memory

This chapter describes the procedure for writing an FSP application to serial flash memory.

Refer to the Getting Started with RZ/A Flexible Software Package (R01AN6499EJ) for building the FSP application.

This chapter shows an example of saving the FSP application binary file (RZA3UL_CA55_BLINKY.srec) in C:\work.

1. Set the SMARC board operating mode to Boot Mode3 (1.8-V serial Flash memory) and turn on the power.
2. Launch e² studio and click icon to the right of the RZA3UL_IPL Default configuration to bring up the setting window.

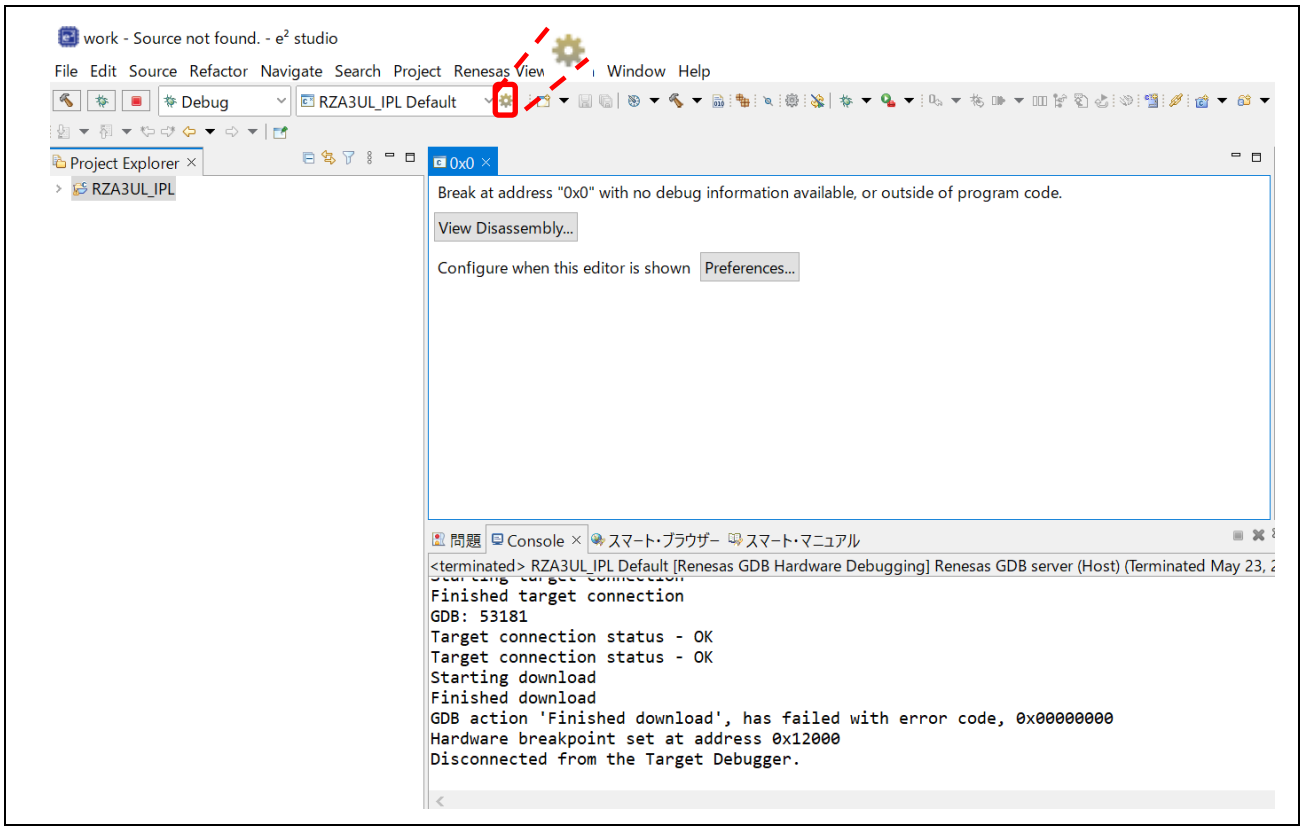


**Figure 9.7 Procedure of writing FSP Application (1)**

3. Edit Configuration will be launched.
   Set the “On connect” of all registered images to No, set the image not to be downloaded when connecting to the SMARC board, and click OK.

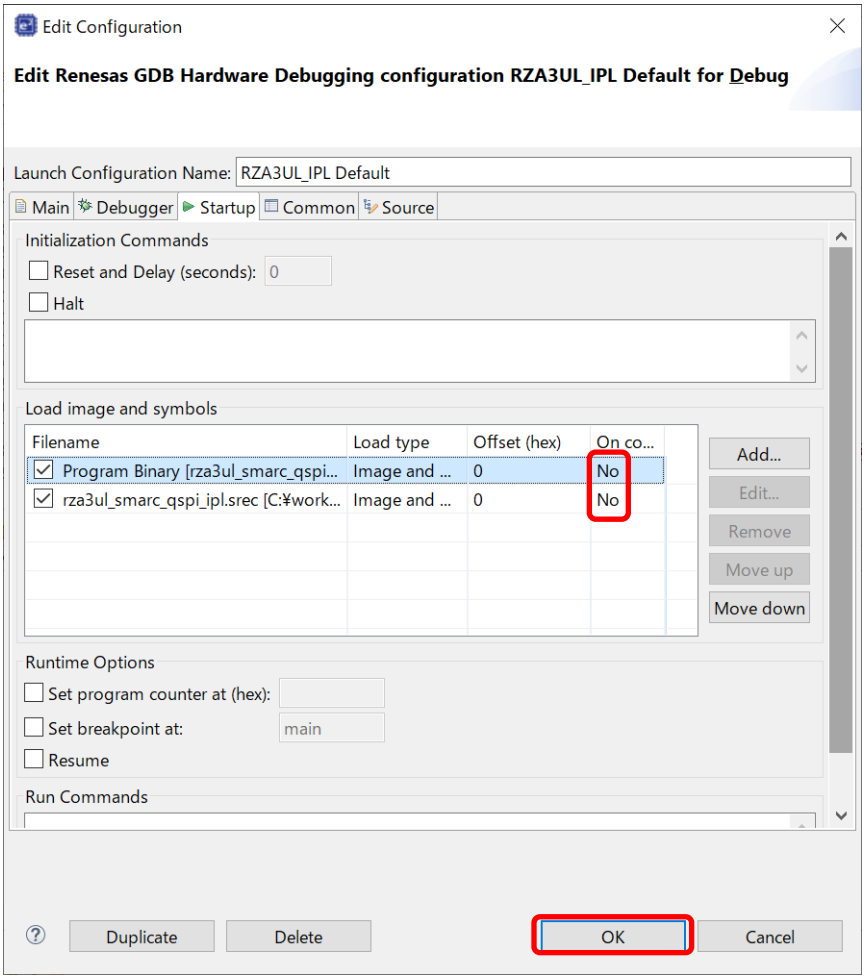


**Figure 9.8 Procedure of writing FSP Application (2)**

4. Make sure the RZA3UL_IPL Default configuration is selected on the e² studio window and click icon to connect to the SMARC board.
   The perspective for the debug control is launched.

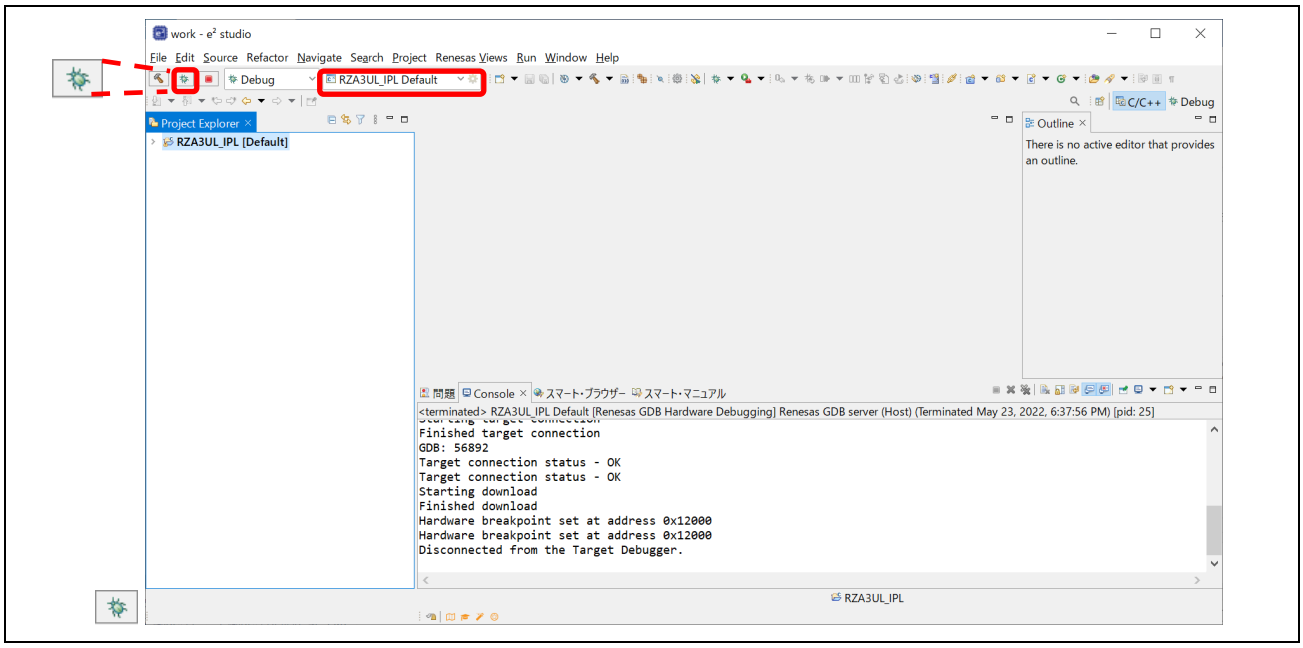


**Figure 9.9 Procedure of writing FSP Application (3)**

5. If “Confirm Perspective Switch” is displayed during the perspective launch, check “Remember my decision” and select [Switch].

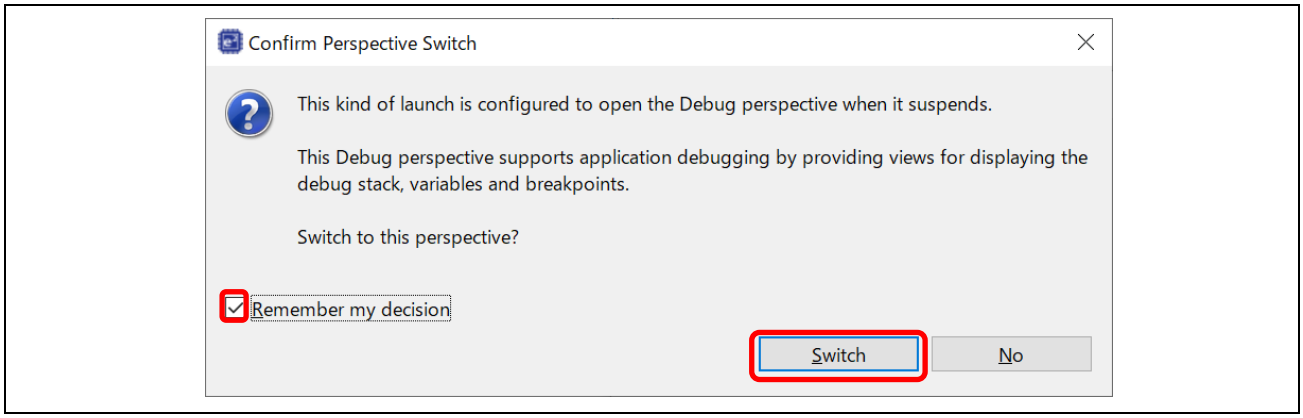


**Figure 9.10 Procedure of writing FSP Application (4)**

6. The board is connected and the perspective is launched.
   Click Load Ancillary file .

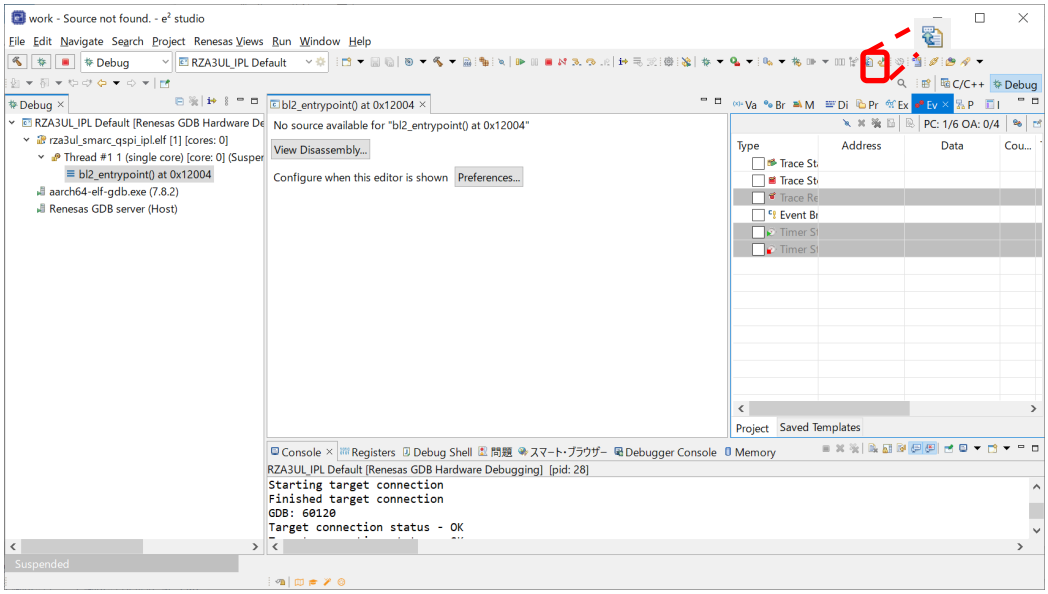


**Figure 9.11 Procedure of writing FSP Application (5)**

7. The Load Ancillary File window will be launched.
   Click [File System] and select C:\work\RZA3UL_CA55_BLINKY.srec.
   Also, uncheck “Load as raw binary image”.
   Click [OK] to write to the serial flash memory.

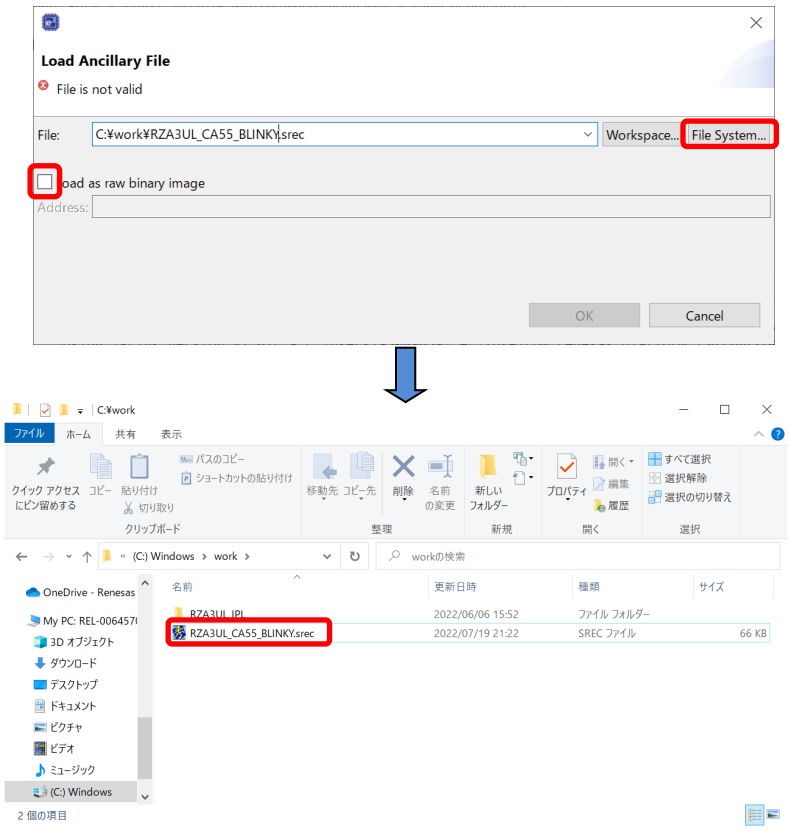


**Figure 9.12 Procedure of writing FSP Application (6)**

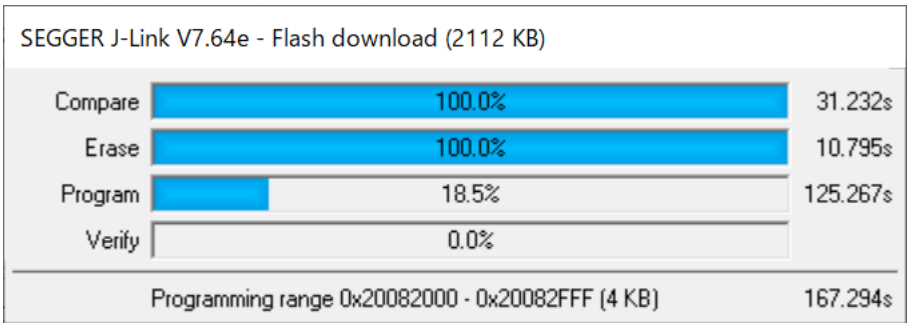


**Figure 9.13 Procedure of writing FSP Application (7)**

8. After writing, click mark to disconnect from the SMARC board.

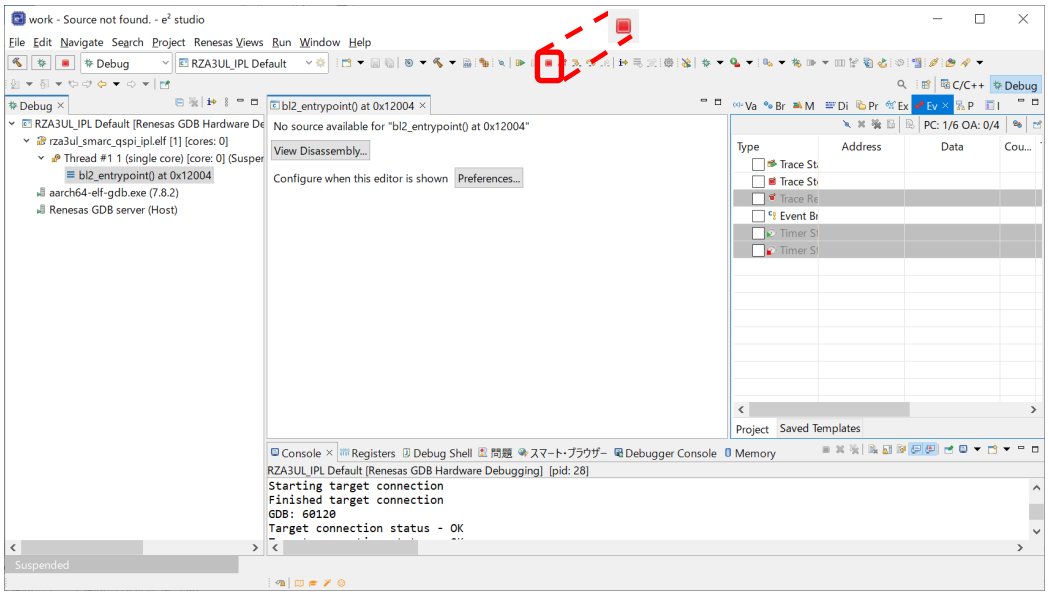


**Figure 9.14 Procedure of writing FSP Application (8)**

# 10. Precautions

## 10.1 About downloading and executing IPL RAM to On-chip RAM

In this version, if the IPL is written to serial flash, the IPL will stop when the IPL is downloaded to the on-chip RAM and run according to the procedure in 9.1..

Erase the IPL from Serial Flash Memory with Flash Writer in advance.

## 10.2 About DDR SDRAM access prohibited area

Test data for DDR SDRAM initialization is stored in the DDR SDRAM area shown below.

If you overwrite the test data, you may not be able to access DDR SDRAM correctly.

Therefore, access to this area is prohibited.

Access prohibited address:

0x40000000 - 0x4000003C

## 11. Reference Documents

User's Manual: Hardware

RZ/A3UL Group User's Manual: Hardware (R01UH0973EJ)
(Please get the latest version from the Renesas Electronics website)

RZ/G2UL, RZ/A3UL, RZ/FIVE SMARC Module Board User's Manual: Hardware (R01UH0990EJ)
(Please get the latest version from the Renesas Electronics website)

RZ SMARC Series Carrier Board User's Manual: Hardware (R01UH0990EJ)
(Please get the latest version from the Renesas Electronics website)

Getting Started with RZ/A Flexible Software Package (R01AN6499EJ)
(Please get the latest version from the Renesas Electronics website)

Arm Architecture Reference Manual for A-profile architecture
(Please get the latest version from the Arm website)

Arm Cortex-A Series Programmer's Guide for ARMv8-A Version:1.0
(Please get the latest version from the Arm website)

Arm Cortex-A55 Core Technical Reference Manual Revision; r2p0
(Please get the latest version from the Arm website)

Technical Update/Technical News
(Please get the latest information from the Renesas Electronics website)

User's Manual: Integrated Development Environment
(Please get the latest version from the Renesas Electronics website)

## Revision History

| Rev. | Date | Description | |
|---|---|---|---|
| | | Page | Summary |
| 1.0.0 | Jul. 28. 22 | - | First edition issued |
| | | | |

# 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/.

© 2022 Renesas Electronics Corporation. All rights reserved.