

RL78/G23

Getting Started Guide for Connecting Amazon Web Services in LTE Communication: RL78/G23-128p Fast Prototyping Board + FreeRTOS

Introduction

This document describes how to connect to Amazon Web Services (AWS) by using a Renesas MCU board combined with a cellular IoT module.

Related Documents

RL78/G23 User's Manual: Hardware (R01UH0896)

RL78/G22, RL78/G23, RL78/G24 Firmware Update Module (R01AN6374)

RL78/G23-128p Fast Prototyping Board User's Manual (R20UT4870)

Renesas Flash Driver RL78 Type 01 User's Manual (R20UT4830)

Notification: End-Of-Life (EOL) process on RYZ024A Cellular module

Renesas announces to discontinue the existing Sequans-sourced LTE module known as the RYZ024A part number and will no longer be shipping this product.

If you have one in a current design or in production, the Sequans part number, GM02S is pin for pin, form fit and function exact drop-in replacement from the RYZ024A. Below Cellular driver of RX family works the below alternate product combination.

- RYZ024A Cellular module control module: Sequans GM02S is the compatible module.

Regarding EOL notice of the RYZ024A, see [the link](#) at [the product page](#).

Contents

| | |
|--|----|
| 1. Overview | 5 |
| 1.1 Overview of Demo Projects | 5 |
| 1.2 Operation Confirmation Conditions | 6 |
| 1.3 Equipment List..... | 8 |
| 2. Description of Hardware | 9 |
| 2.1 Demo Project (PubSub) | 9 |
| 2.1.1 System Configuration | 9 |
| 2.1.2 List of Pins Used..... | 9 |
| 2.2 Demo Project (OTA)..... | 10 |
| 2.2.1 System Configuration | 10 |
| 2.2.2 List of Pins Used..... | 10 |
| 3. Description of Software..... | 11 |
| 3.1 Demo Project (PubSub) | 11 |
| 3.1.1 Demo Project Structure | 11 |
| 3.1.2 List of Option Bytes Settings | 11 |
| 3.2 Demo Project (OTA)..... | 12 |
| 3.2.1 Demo Project Structure | 12 |
| 3.2.2 List of Option Bytes Settings | 13 |
| 3.3 Folder Structure..... | 14 |
| 3.4 Code Size | 15 |
| 4. Setup Common to Demo Projects (PubSub and OTA)..... | 16 |
| 4.1 Hardware Setup..... | 16 |
| 4.1.1 Overall Configuration..... | 16 |
| 4.1.2 Connecting Hardware..... | 16 |
| 4.2 Software Setup..... | 19 |
| 4.2.1 Terminal Software Settings | 19 |
| 4.2.2 Installing Flash Writer | 19 |
| 4.2.3 Adding SIM Card Information to the Demo Project..... | 19 |
| 4.2.4 Adding AWS IoT Connection Settings to the Demo Project | 21 |
| 5. Setup Specific to Demo Project (PubSub) | 23 |
| 5.1 Preparation | 23 |
| 5.2 Importing the Project | 23 |
| 5.3 Setting the Build Configuration..... | 24 |
| 5.4 Building the Demo Project..... | 24 |
| 5.5 Preparing the MQTT Test Client | 24 |
| 5.6 Running the Demo Project | 25 |
| 5.7 Debugging the Demo Project | 26 |

| | | |
|---------|---|----|
| 6. | Setup Specific to Demo Project (OTA)..... | 27 |
| 6.1 | Preparation..... | 27 |
| 6.1.1 | Installing Tools..... | 27 |
| 6.1.2 | Generating Keys for Signature Generation and Verification..... | 28 |
| 6.1.3 | Settings for OTA Update..... | 30 |
| 6.1.3.1 | Creating Amazon S3 Buckets..... | 30 |
| 6.1.3.2 | Creating an OTA Update Service Role..... | 33 |
| 6.1.3.3 | Creating an OTA Update User Policy..... | 37 |
| 6.1.3.4 | Allocating an OTA Update Policy to IAM User..... | 45 |
| 6.1.3.5 | Granting Access Permissions to AWS IoT Code Signature..... | 50 |
| 6.2 | Creating an Initial Image..... | 55 |
| 6.2.1 | Creating a Bootloader..... | 55 |
| 6.2.1.1 | Importing the Bootloader Project..... | 55 |
| 6.2.1.2 | Adding the Firmware Verification Key to the Bootloader Project..... | 56 |
| 6.2.1.3 | Building the Bootloader Project..... | 56 |
| 6.2.2 | Creating an Initial Application..... | 57 |
| 6.2.2.1 | Importing the Initial Application..... | 57 |
| 6.2.2.2 | Setting the Build Configuration of the the Initial Application..... | 58 |
| 6.2.2.3 | Adding the Firmware Verification Key to the Initial Application..... | 58 |
| 6.2.2.4 | Building the Initial Application..... | 58 |
| 6.2.3 | Creating an Initial Image by Using Renesas Image Generator..... | 59 |
| 6.3 | Creating an Update Image..... | 60 |
| 6.3.1 | Creating an Update Application..... | 60 |
| 6.3.1.1 | Changing the Source Code of the Application..... | 60 |
| 6.3.1.2 | Building the Update Application..... | 60 |
| 6.3.1.3 | Renaming the MOT File of the Update Application..... | 60 |
| 6.3.2 | Generating an Update Image by Using Renesas Image Generator..... | 61 |
| 6.4 | Running the Demo Project..... | 62 |
| 6.4.1 | Programming the Initial image (initial_image.mot) to Board..... | 62 |
| 6.4.2 | Registering the Update Image (aws_ryz024a_rl78g23-fpb_ota_093.rsu) with OTA Jobs..... | 63 |
| 6.5 | Debugging the Initial Application..... | 70 |
| 7. | Using Renesas Flash Programmer..... | 71 |
| 7.1 | When Using COM Port..... | 71 |
| 7.1.1 | Setting Jumper Pins..... | 71 |
| 7.1.2 | Supplying Power to the MCU Board..... | 71 |
| 7.1.3 | Creating a New Project and Connecting to the MCU Board..... | 72 |
| 7.1.4 | Programming a MOT File to the MCU Board..... | 74 |
| 7.2 | When Using Emulator..... | 75 |
| 7.2.1 | Setting Jumper Pins, Mounting the Connector, and Cutting Patterns..... | 75 |
| 7.2.2 | Supplying Power to the MCU Board..... | 76 |

| | | |
|-------|---|----|
| 7.2.3 | Creating a New Project and Connecting to the MCU Board | 76 |
| 7.2.4 | Programming a MOT File to the MCU Board | 77 |
| 8. | Debug Procedure | 78 |
| 8.1 | When Using COM Port..... | 78 |
| 8.1.1 | Setting Jumper Pins | 78 |
| 8.1.2 | Supplying Power to the MCU Board | 78 |
| 8.1.3 | Debug Configurations..... | 78 |
| 8.1.4 | Debugger Settings..... | 79 |
| 8.2 | When Using Emulator | 80 |
| 8.2.1 | Mounting the Connector, Setting Jumper Pins, and Cutting Patterns..... | 80 |
| 8.2.2 | Connecting the Emulator to the MCU Board | 80 |
| 8.2.3 | Debug Configurations..... | 81 |
| 8.2.4 | Debugger Settings..... | 82 |
| 9. | Appendix | 83 |
| 9.1 | Precautions on Porting Third-Party Libraries to RL78..... | 83 |
| 9.1.1 | Width of int Is 16 Bits..... | 83 |
| 9.1.2 | Size Limitation of Section | 83 |
| 9.2 | License Information for Open Source Software Used with Demo Projects..... | 84 |
| 10. | Websites and Supports..... | 85 |
| | Revision History | 86 |

Notes:

AWS™ is a trademark of Amazon.com, Inc. or its affiliates. (<https://aws.amazon.com/trademark-guidelines/>)

FreeRTOS™ is a trademark of Amazon Web Services, Inc. (<https://freertos.org/copyright.html>)

GitHub® is a trademark of GitHub, Inc. (<https://github.com/logos>)

1. Overview

The sample program [iot-reference-rl78](#) provides the reference of IoT solution with using RL78 family, AWS, and FreeRTOS. You can easily try to run AWS IoT demos while it works with our various other products.

1.1 Overview of Demo Projects

The sample program contains the following demo projects. These demo projects realize the operation for connecting to the AWS clouds by using the Renesas MCU board RL78/G23-128p Fast Prototyping Board and cellular IoT module.

Table 1-1 List of demo projects

| ItemName of Demo Project | Description |
|--------------------------|--|
| Demo project (PubSub) | Perform simple data upload via MQTT communication. |
| Demo project (OTA) | Perform firmware update via OTA. |

For details about summary of each demo projects, refer to the following chapters.

- Section 2, Description of Hardware
- Section 3, Description of Software

For details about how to run the demo projects, refer to the following chapters.

- Demo project (PubSub)
 - Section 4, Setup Common to Demo Projects (PubSub and OTA)
 - Section 5, Setup Specific to Demo Project (PubSub)
- Demo project (OTA)
 - Section 4, Setup Common to Demo Projects (PubSub and OTA)
 - Section 6, Setup Specific to Demo Project (OTA)

1.2 Operation Confirmation Conditions

Demo project operations have been confirmed in the following conditions.

Table 1-2 Operation Confirmation Conditions (RL78/G23)

| Item | Description |
|--|--|
| MCU used | RL78/G23 (R7F100GSN CF 768KB) |
| Board used | RL78/G23-128p Fast Prototyping Board (RTK7RLG230CSN000BJ) |
| Operating frequency | High-speed on-chip oscillator clock: 32 MHz |
| Operating voltage | 3.3 V |
| IDE (Integrated Development Environment) | Renesas Electronics e ² studio 2024-01.1 |
| C compiler | Renesas Electronics CC-RL V1.12.01 |
| Firmware programming tool | Renesas Flash Programmer V3.14.00 |
| Smart Configurator (SC) | Renesas Smart Configurator for RL78 24.1.0.v20231218-0132 |
| Board support package (BSP) | v1.60 (r_bsp) |
| Flash library (RFD) | Renesas Flash Driver (RFD) RL78 Type01 for RL78/G2x V1.20 Note: Code Flash Libraries (Flash Self Programming Libraries) -> Renesas Flash Driver RL78 Type 01 Package V1.20 for RL78/G2x |
| Firmware update module (FWUP) | RL78/G22,RL78/G23,RL78/G24 Firmware Update Module v2.01 |
| Utility tool to generate firmware images | Renesas Image Generator V3.03 Note: Included in the firmware update module (FWUP) |
| Python | Python 3.10.1 |
| OpenSSL | OpenSSL 3.1.4 |

Table 1-3 Operation Confirmation Conditions (Others, such as OSS Library)

| Item | Description |
|---------------------------------|---|
| iot-reference-rl78 | v202210.01-LTS-rl78-1.0.0 (Based FreeRTOS 202210.01-LTS) https://github.com/renesas/iot-reference-rl78/tree/v202210.01-LTS-rl78-1.0.0 |
| FreeRTOS Cellular Interface | 1.3.0 https://github.com/FreeRTOS/FreeRTOS-Cellular-Interface |
| FreeRTOS Kernel | 10.5.1 https://github.com/FreeRTOS/FreeRTOS-Kernel |
| backoffAlgorithm | 1.3.0 https://github.com/FreeRTOS/backoffAlgorithm |
| coreJSON | 3.2.0 https://github.com/FreeRTOS/coreJSON |
| coreMQTT Client | 2.1.1 https://github.com/FreeRTOS/coreMQTT |
| coreMQTT Agent | 1.2.0 https://github.com/FreeRTOS/coreMQTT-Agent |
| AWS IoT Over-the-air Update | 3.4.0 https://github.com/aws/ota-for-aws-iot-embedded-sdk |
| tinycbor | 0.5.2 https://github.com/intel/tinycbor |
| FreeRTOS-Plus network_transport | No version https://www.freertos.org/network-interface.html |
| Logging Interface | 1.1.3 https://github.com/aws/amazon-freertos/tree/main/libraries/logging |
| TinyCrypt Cryptographic Library | 0.2.8 https://github.com/intel/tinycrypt |

1.3 Equipment List

The following lists the equipment required for the demo projects.

Table 1-4 Equipment List

| Item | Description |
|----------------------------|---|
| MCU board | RL78/G23-128p Fast Prototyping Board RTK7RLG230CSN000BJ - RL78/G23-128p Fast Prototyping Board |
| Cellular IoT module | PMOD Expansion Board for RYZ024A (referred to as RYZ024A hereafter) RTKYZ024A0B00000BE - PMOD Expansion Board for RYZ024A |
| SIM card | LTE communication must be possible. Example: SIM card by Truphone bundled with RTKYZ024A0B00000BE (Note) DHA-SIM-132 by Nippon SIM |
| USB-UART conversion board | Pmod USBUART https://reference.digilentinc.com/reference/pmod/pmodusbuart/start |
| Micro USB Type-B cable x 3 | <ul style="list-style-type: none"> Used to connect the USB-UART conversion board to the PC Used to connect the MCU board to the PC Used to supply power to RYZ024A |
| Jumper wire x 3 | Used to connect the USB-UART conversion board to the MCU board |
| Jumper pin x 3 | Pins J15, J16, and J19 are used to select the MCU board power supply. |

Note:

When using a SIM card by Truphone bundled with PMOD Expansion Board for RYZ024A ([RTKYZ024A0B00000BE](#)), you must activate your SIM card by the following document: [RA6M5 Group RYZ024A PMOD LTE Connectivity with RA6M5 MCU Quick Start Guide \(R21QS0007\)](#).

Overall figure of equipment connections

Refer to the followings for overall figure of equipment connections for each demo.

- demo project (PubSub): Figure 4-1 Overall Hardware Configuration of the Demo Project
- demo project (OTA): Same as above

Precaution about equipment for debugging

The demo projects use the COM port for debugging, but debugging with the emulator is also possible. When using the emulator, you need to mount the connector for connecting the emulator and change the circuit. For details, refer to section 7.2.1, Setting Jumper Pins, Mounting the Connector, and Cutting Patterns or the following manual.

Table 1-5 Debug Equipment

| Item | Description |
|----------|---|
| Emulator | E2 emulator Lite https://www.renesas.com/us/en/software-tool/e2-emulator-lite-rte0t0002lkce00000r |

2. Description of Hardware

2.1 Demo Project (PubSub)

2.1.1 System Configuration

The following shows the system configuration of the demo project (PubSub).

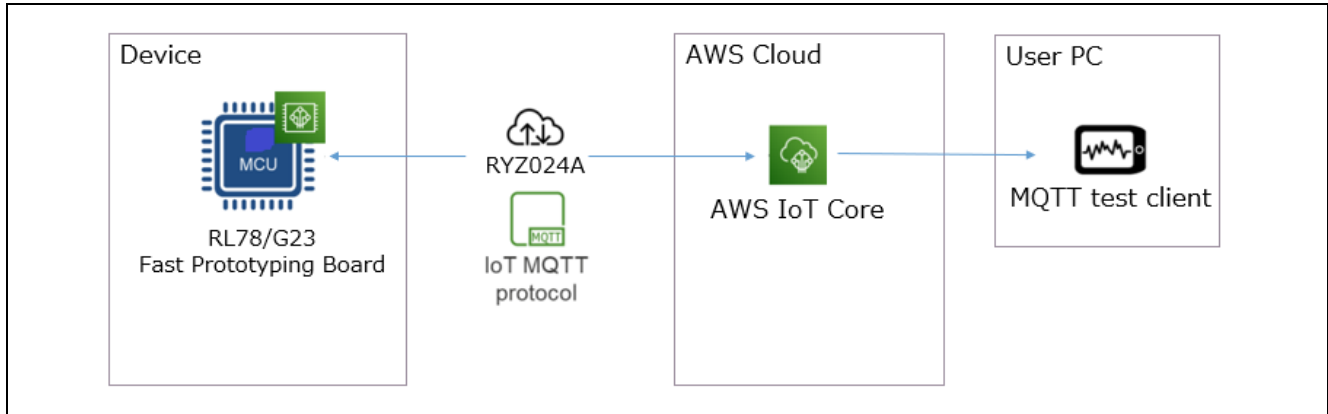


Figure 2-1 System Configuration of Demo Project (PubSub)

2.1.2 List of Pins Used

The following lists and describes the pins used with the demo project (PubSub).

Table 2-1 Pins Used with Demo Project (PubSub) and Their Functions

| Pin Name | I/O | Description |
|-----------|--------|--|
| P143/RxD3 | Input | UART communication (reception) with RYZ024A |
| P144/TxD3 | Output | UART communication (transmission) with RYZ024A |
| P00 | Output | Reset to RYZ024A |
| P142 | Output | UART communication (RTS) with RYZ024A |
| P14/RxD2 | Input | Terminal input |
| P13/TxD2 | Output | Terminal output |
| P50 | Output | LED1 |

2.2 Demo Project (OTA)

2.2.1 System Configuration

The following shows the system configuration of the demo project (OTA).

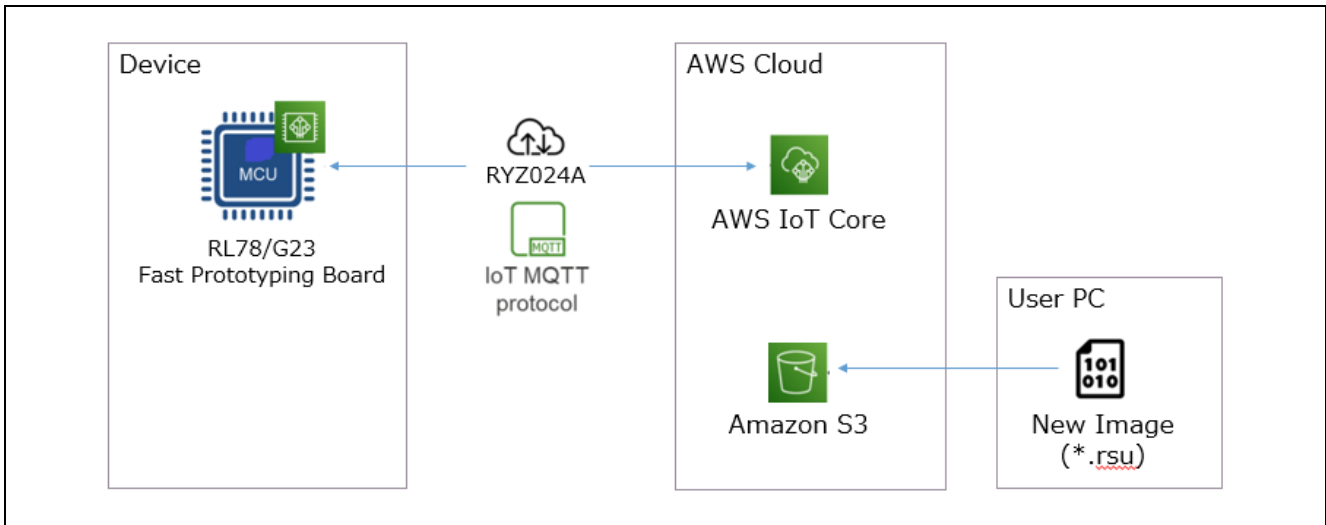


Figure 2-2 System Configuration of Demo Project (OTA)

2.2.2 List of Pins Used

The following lists and describes the pins used with the demo project (OTA).

Table 2-2 Pins Used with Demo Project (OTA) and Their Functions

| Pin Name | I/O | Description |
|-----------|--------|--|
| P143/RxD3 | Input | UART communication (reception) with RYZ024A |
| P144/TxD3 | Output | UART communication (transmission) with RYZ024A |
| P00 | Output | Reset to RYZ024A |
| P142 | Output | UART communication (RTS) with RYZ024A |
| P14/RxD2 | Input | Terminal input |
| P13/TxD2 | Output | Terminal output |
| P50 | Output | LED1 |

3. Description of Software

3.1 Demo Project (PubSub)

3.1.1 Demo Project Structure

This demo project connects to the AWS from the MCU board, and then issues messages on a regular basis by using the MQTT library.

3.1.2 List of Option Bytes Settings

The followings show the option bytes settings.

Table 3-1 Option Bytes Settings

| Address | Settings | Description |
|---------------|-----------|---|
| 000C0H/040C0H | 11101111B | Stops the watchdog timer operation. (Stops counting after the release from the reset state.) |
| 000C1H/040C1H | 00111010B | LVD0 off (using an external reset input from the RESET pin) |
| 000C2H/040C2H | 11101000B | HS (high-speed main) mode and High-speed on-chip oscillator clock (fIH): 32 MHz |
| 000C3H/040C3H | 10000100B | Enables on-chip debugging. |

3.2 Demo Project (OTA)

3.2.1 Demo Project Structure

The firmware update mechanism of this demo project uses the partial update method (buffer side is internal flash) provided by the firmware update module. For details, refer to “[RL78/G22,RL78/G23,RL78/G24 Firmware Update Module](#)”.

The following illustrates the firmware update mechanism and shows the memory map.

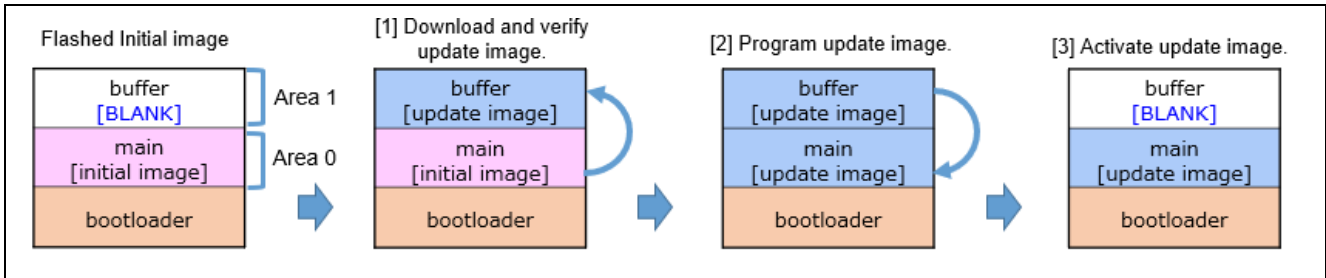


Figure 3-1 Firmware Update Mechanism

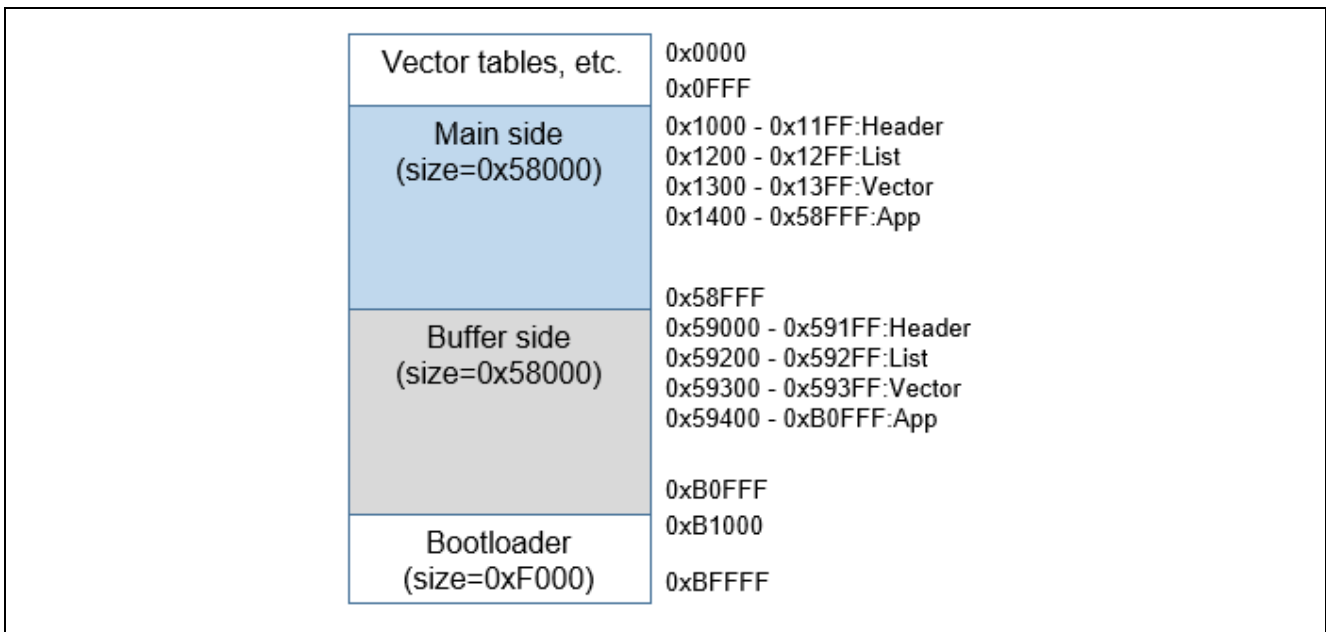


Figure 3-2 Memory Map of Demo Project (OTA)

3.2.2 List of Option Bytes Settings

The followings show the option bytes settings.

Table 3-2 Option Bytes Settings

| Address | Settings | Description |
|---------------|-----------|--|
| 000C0H/040C0H | 11101111B | Stops the watchdog timer operation. (Stops counting after reset.) |
| 000C1H/040C1H | 00111010B | LVD0 off (using an external reset input from the RESET pin) |
| 000C2H/040C2H | 11101000B | HS (high-speed main) mode and High-speed on-chip oscillator clock (fIH): 32 MHz |
| 000C3H/040C3H | 10000100B | Enables on-chip debugging. |

3.3 Folder Structure

The following shows the folder structure of the sample program.

Table 3-3 Folder Structure of the Sample Program

| Folder Name | Description |
|--|---|
| iot-reference-rl78 | The sample program described in this Getting Started Guide. |
| ├─Common | |
| │ ├─FreeRTOS_common | |
| │ └─ports | |
| │ └─ota_pal | |
| ├─Configuration | |
| │ └─rl78g23-fpb | |
| │ └─ota | OTA demo configurations. |
| │ └─pubsub | PubSub demo configurations. |
| │ └─test | |
| ├─Demos | |
| │ ├─common | |
| │ ├─include | |
| │ ├─mqtt_agent | |
| │ ├─OtaOverMqtt | OTA demo source codes. |
| │ └─SimplePubSub | PubSub demo source codes. |
| ├─IDT_config | |
| ├─Middleware | |
| │ ├─3rdparty | |
| │ ├─Application-Protocols | |
| │ └─network_transport | |
| │ ├─AWS | |
| │ └─ota-for-aws-iot-embedded-sdk | |
| │ ├─FreeRTOS | FreeRTOS Kernel and libraries. |
| │ ├─backoffAlgorithm | |
| │ ├─coreJSON | |
| │ ├─coreMQTT | |
| │ ├─coreMQTT-Agent | |
| │ ├─FreeRTOS-Cellular-Interface | |
| │ └─FreeRTOS-Kernel | |
| │ └─logging | |
| ├─Projects | |
| │ └─rl78g23-fpb | |
| │ ├─application_code | |
| │ ├─flash_proj | |
| │ ├─helper | |
| │ ├─modules | |
| │ ├─projects | Import below folders to IDE. |
| │ ├─aws_ryz024a_rl78g23-fpb | PubSub demo and OTA demo. Select by Build Configurations. |
| │ ├─boot_loader | Boot loader for OTA demo. |
| │ └─test_aws_cellular_ryz024a | |
| │ └─rtos_skelton | |
| ├─Test | |
| └─Tools | |

3.4 Code Size

The following table shows the ROM and RAM size of demo projects confirmed in the following conditions.

- CC-RL
 - Compile options:
 - -Odefault: Optimization that is effective for both the object size and execution speed.
 - Link options:
 - -optimize=symbol_delete: Deleting variables or functions that have not been referenced even once.

Table 3-4 ROM and RAM Size of Demo Projects

| Demo Project Name | ROM (byte) | RAM (byte) |
|--|------------|------------|
| aws_ryz024a_rl78g23-fpb (demo project (PubSub)) | 142311 | 29913 |
| aws_ryz024a_rl78g23-fpb (demo project (OTA)) | 234729 | 36790 |
| boot_loader | 22147 | 1348 |

4. Setup Common to Demo Projects (PubSub and OTA)

The following describes the setup procedure applicable to demo project (PubSub) and demo project (OTA).

4.1 Hardware Setup

4.1.1 Overall Configuration

First, the following shows the overall configuration of hardware that makes up the demo project.

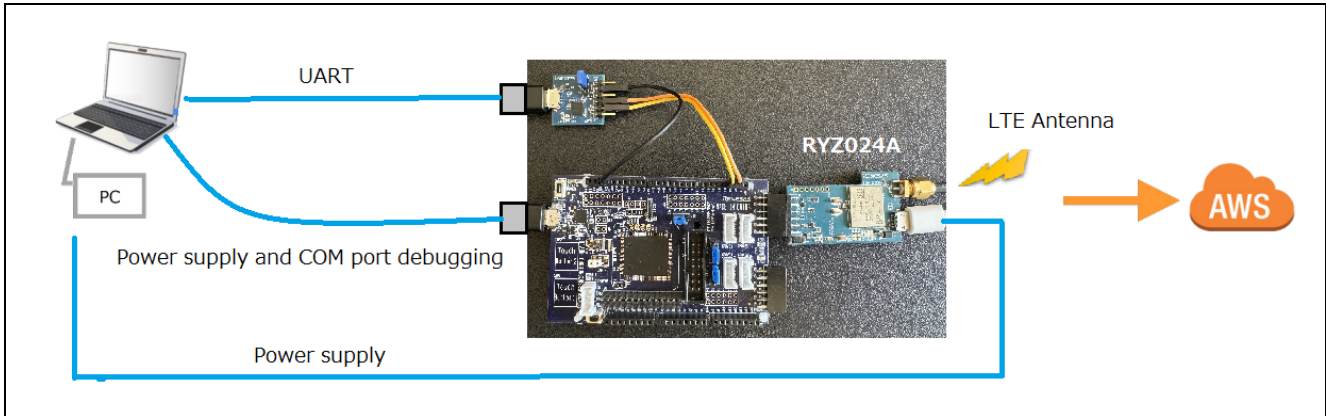


Figure 4-1 Overall Hardware Configuration of the Demo Project

4.1.2 Connecting Hardware

The following describes how to connect hardware.

- (1) Insert the activated SIM card into RYZ024A.

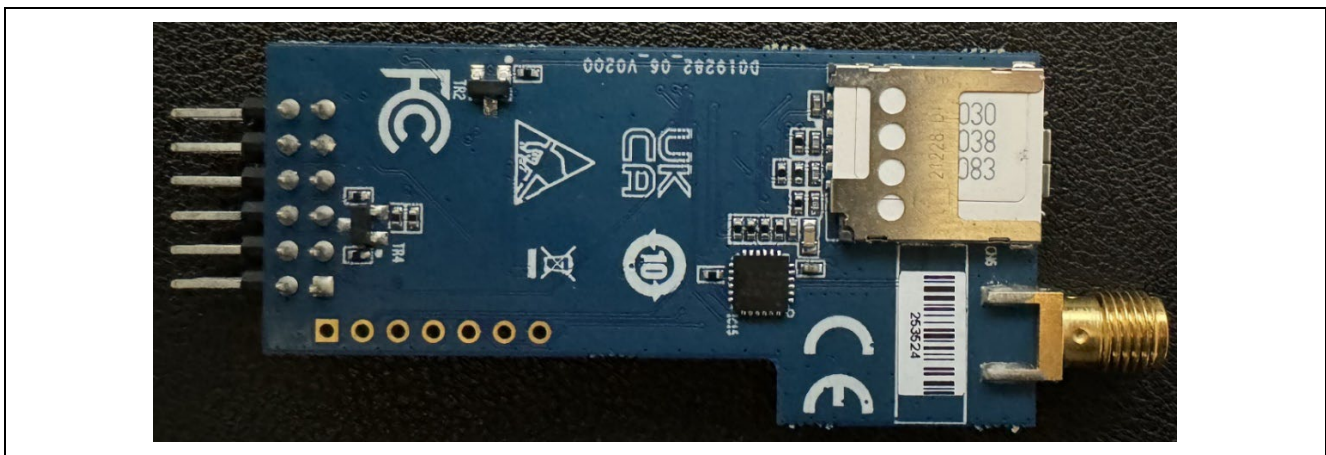


Figure 4-2 Inserting Activated SIM Card into RYZ024A

(2) Connect the antenna and power supply USB cable to RYZ024A.

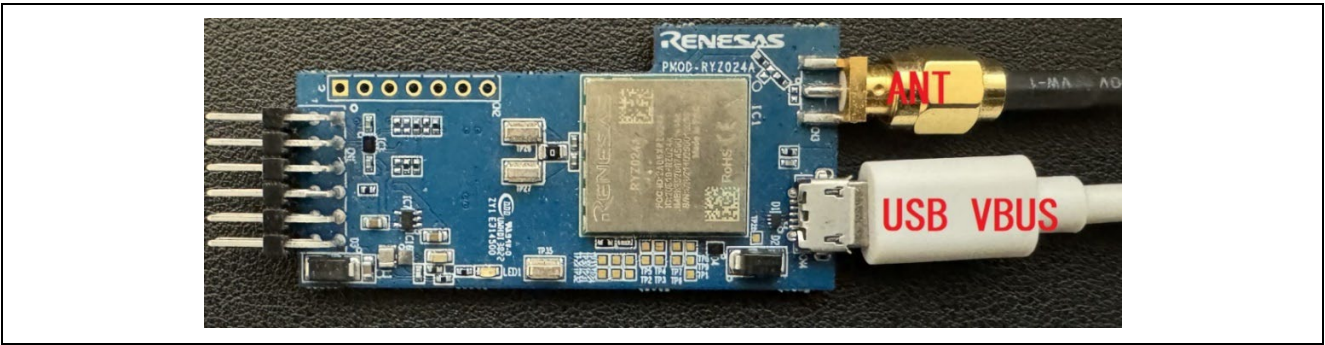


Figure 4-3 Connecting Antenna and Power Supply USB Cable to RYZ024A

(3) Connect RYZ024A to PMOD1 of the MCU board.

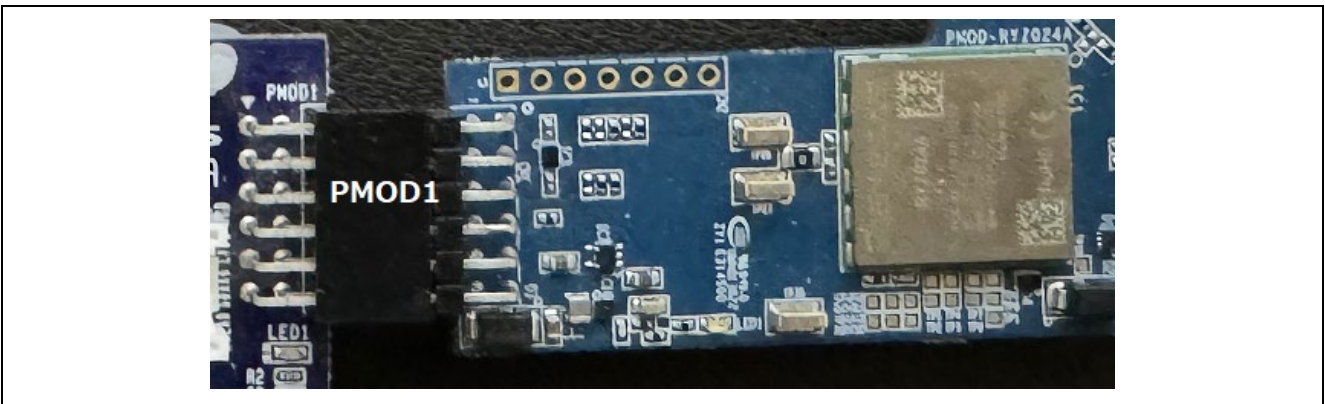


Figure 4-4 Connecting RYZ024A to PMOD1 of the MCU Board

(4) Connect the USB-UART conversion board to the MCU board.

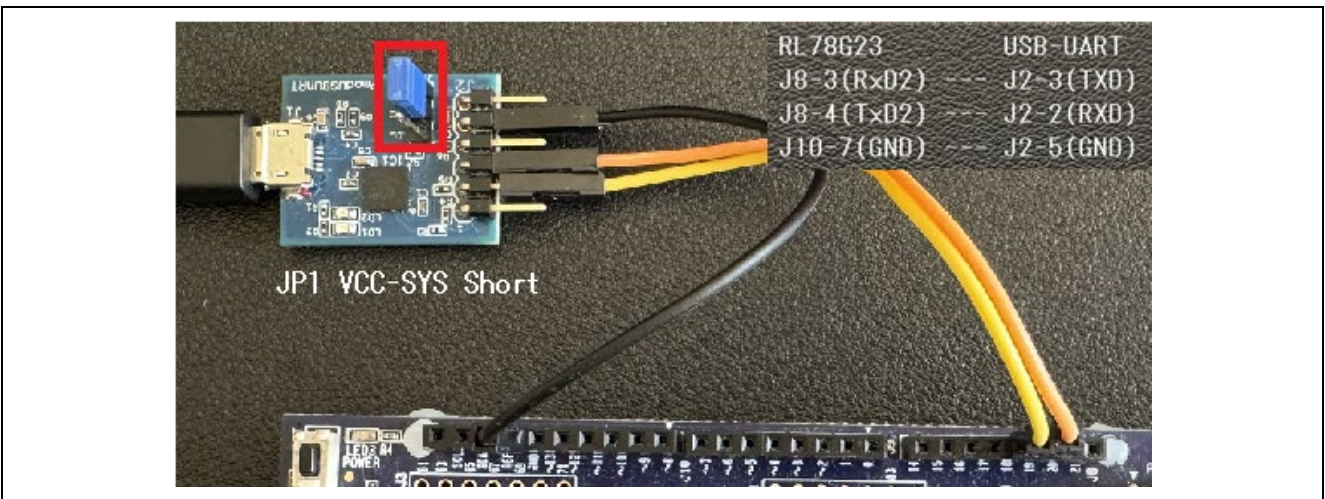


Figure 4-5 Connecting the USB-UART Conversion Board to the MCU Board

(5) On the MCU board, set the power supply selection header to J20 2-3 Short to select 3.3 V power supply.

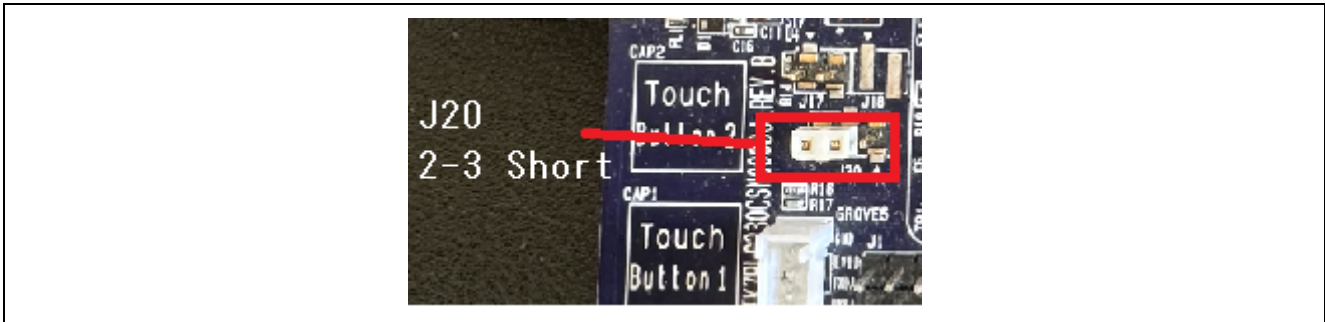


Figure 4-6 Setting MCU Board Power Supply to 3.3 V

(6) If you changed circuit to mount emulator connector on the MCU board, configure the COM port debugging that uses a USB-to-serial converter. If you don't change circuit, you don't need this process.

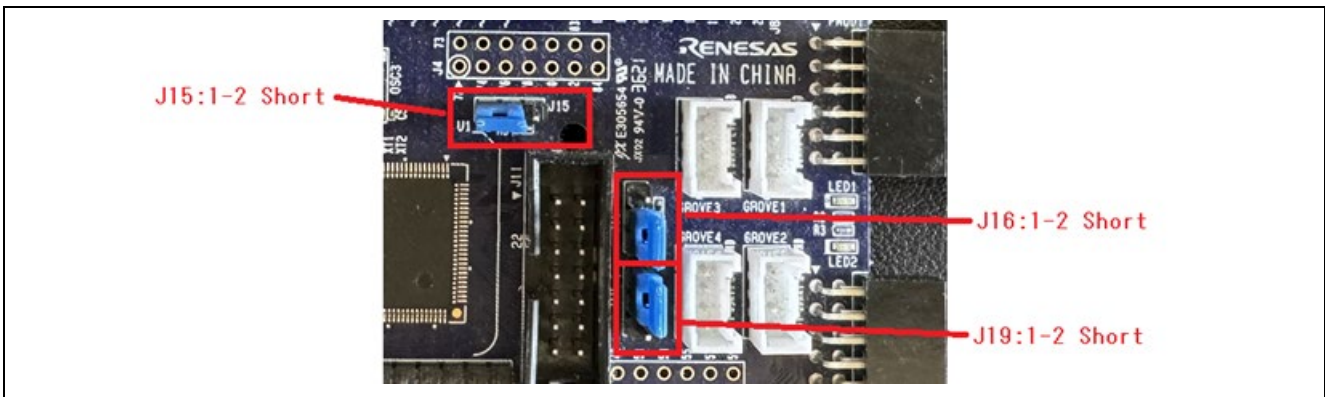


Figure 4-7 Settings for Using COM Port Debugging (Top Side)

(7) Connect the USB cable to supply power to the MCU board.

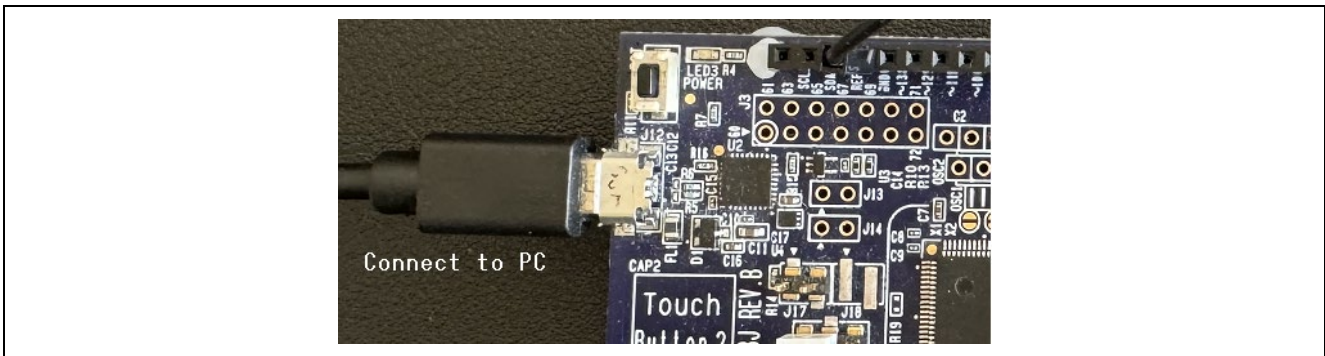


Figure 4-8 Supplying Power to the MCU Board

(8) Confirm the COM port number.

The COM port number will be used for programming and debugging firmware.

(9) Remove the USB cable to stop power supply to the MCU board.

4.2 Software Setup

4.2.1 Terminal Software Settings

Terminal software (example: Tera Term) is required to output demo project logs. The followings show the serial port settings.

Table 4-1 Serial Port Settings

| Item | Description |
|--------------|-------------|
| Baud rate | 115200 bps |
| Data | 8 bits |
| Parity | None |
| Stop bit | 1 bit |
| Flow control | None |

4.2.2 Installing Flash Writer

A flash writer is used for programming initial images.

[Renesas Flash Programmer \(Programming GUI\)](#)

4.2.3 Adding SIM Card Information to the Demo Project

Specify the SIM card information for the following macros in the demo project. Refer to a manual of your SIM card for SIM card information.

- `iot-reference-rl78\Projects\rl78g23-fpb\modules\r_config\r_aws_cellular_config.h`
 - `AWS_CELLULAR_CFG_AP_NAME`: Access point name
 - `AWS_CELLULAR_CFG_AP_USERID`: User ID for access point (Note 1)
 - `AWS_CELLULAR_CFG_AP_PASSWORD`: Password for access point (Note 1)
 - `AWS_CELLULAR_CFG_PIN_CODE`: PIN code (Note 2)
 - `AWS_CELLULAR_CFG_AUTH_TYPE`: Authentication type

Note 1: Specify an empty value for the macro if there is no information.

Note 2: Specify an empty string for the macro if there is no information.

The followings show setting examples of each SIM card described in this document.

(1) Case: SIM card by Truphone bundled with [RTKYZ024A0B00000BE](#)

`iot-reference-rl78\Projects\rl78g23-fpb\modules\r_config\r_aws_cellular_config.h`

```
#define AWS_CELLULAR_CFG_AP_NAME      "iot.truphone.com" /* Access point name */
#define AWS_CELLULAR_CFG_AP_USERID    "" /* Login ID */
#define AWS_CELLULAR_CFG_AP_PASSWORD  "" /* Access point password */
#define AWS_CELLULAR_CFG_PIN_CODE     /* SIM card PIN code */
#define AWS_CELLULAR_CFG_AUTH_TYPE    (0) /* Authentication protocol type
(0=None,1=PAP,2=CHAP) */
```

(2) Case: [DHA-SIM-132](#) by Nippon SIM

Set up Guide

Please follow the following steps to set up your prepaid SIM

- ❶ Detach the IC chip from SIM card carefully per the size (nano/micro/standard) required and insert into your device.
- ❷ Create a new APN profile per table to connect to our network. Remember to switch on 「Mobile Network Data」 and 「4G/LTE」 option.
 - iOS
 - Please go to 「Setting」 > 「Mobile」 > tap on 「Primary」 under MOBILE PLANS > 「Mobile Data Network」 fill in the blanks of both 「MOBILE DATA」 and 「PERSONAL HOTSPOT」 per table.
 - For iOS 12 or before, after 「Mobile」 please direct continue to 「Mobile Data Network」
 - Android
 - Please go to 「Setting」 > 「Mobile Network」 > 「Access Point Name (APN)」, tap on 「+」 or 「New APN」 to create a new APN with the settings per table.
 - Please select the created APN in order to use it.
- ❸ After finish using
 - Carefully remove and exchange back to your local SIM.
 - For iOS device, delete the installed profile and restart your phone. Some MVNO users may need to reinstall the respective profile.
 - For Android, please re-select the original APN which your local SIM use.

| | |
|---------------------|------------------|
| Name | Example SIM |
| APN | example.jp |
| User ID | example@sim |
| Password | example_password |
| Authentication Type | PAP or CHAP |

Figure 4-9 Example of SIM Card Manual

iot-reference-rl78\Projects\rl78g23-fpb\modules\r_config\r_aws_cellular_config.h

```

#define AWS_CELLULAR_CFG_AP_NAME           "example.jp"           /* Access point name */
#define AWS_CELLULAR_CFG_AP_USERID        "example@sim"         /* Login ID */
#define AWS_CELLULAR_CFG_AP_PASSWORD      "example_password"     /* Access point password */
#define AWS_CELLULAR_CFG_PIN_CODE         /* SIM card PIN code */
#define AWS_CELLULAR_CFG_AUTH_TYPE        (2) /* Authentication protocol type
(0=None, 1=PAP, 2=CHAP) */

```

4.2.4 Adding AWS IoT Connection Settings to the Demo Project

Add the settings required for AWS IoT connection to the demo project. The following describes the procedure.

The parts that should be changed according to the user environment are highlighted in yellow.

- (1) Register the device to the IoT Core service then obtain the information (endpoint, thing name, and credential) required for connection. For details, refer to the following.

[Register device to AWS IoT · renesas/iot-reference-rx Wiki · GitHub](#)

- (2) Set the endpoint and thing name to the demo project.
iot-reference-r178\Demos\include\aws_clientcredential.h

```
/*
 * @brief MQTT Broker endpoint.
 *
 * @todo Set this to the fully-qualified DNS name of your MQTT broker.
 */
#define clientcredentialMQTT_BROKER_ENDPOINT    "YOUR_ENDPOINT"

/*
 * @brief Host name.
 *
 * @todo Set this to the unique name of your IoT Thing.
 * Please note that for convenience of demonstration only we
 * are using a #define here. In production scenarios the thing
 * name can be something unique to the device that can be read
 * by software, such as a production serial number, rather
 * than a hard coded constant.
 */
#define clientcredentialIOT_THING_NAME         "YOUR_THING_NAME"
```

- (3) Set the credential (client certificate and private key) to the demo project.
iot-reference-rl78\Demos\include\aws_clientcredential_keys.h

Note: Add `\n` to the end of each line.

```

/*
 * @brief PEM-encoded client certificate.
 *
 * @todo If you are running one of the FreeRTOS demo projects, set this
 * to the certificate that will be used for TLS client authentication.
 *
 * @note Must include the PEM header and footer:
 * "-----BEGIN CERTIFICATE-----\n"\  

 * "...base64 data...\n"\  

 * "-----END CERTIFICATE-----\n"
 */
#define keyCLIENT_CERTIFICATE_PEM \  

"-----BEGIN CERTIFICATE-----\n"\  

"MIIDWTCCAkGgAwIBAgIUFeYR3JSsJbTOS7huEq++YBGgwtowDQYJKoZIhvcNAQEL\n"\  

...\  

"7qHumsC6fsEapoptgcfEpdERl4c9hJR45jHamDVhxZjitQD4klLA0gqTlBNL\n"\  

"-----END CERTIFICATE-----\n"
...
/*
 * @brief PEM-encoded client private key.
 *
 * @todo If you are running one of the FreeRTOS demo projects, set this
 *
 * @note Must include the PEM header and footer:
 * "-----BEGIN RSA PRIVATE KEY-----\n"\  

 * "...base64 data...\n"\  

 * "-----END RSA PRIVATE KEY-----\n"
 */
#define keyCLIENT_PRIVATE_KEY_PEM \  

"-----BEGIN RSA PRIVATE KEY-----\n"\  

"MIIEowIBAAKCAQEA3Fb7O7jQW4lgHmPE3AInUTWUCaR7kWeWHubEk9YbNf3xwx dg\n"\  

...\  

"s/OlVUi ygf0RgeoMVx/3GzZPfmTrB0cQ8XZ7mxCd2dgY9UXQ/oja\n"\  

"-----END RSA PRIVATE KEY-----\n"

```

5. Setup Specific to Demo Project (PubSub)

The following describes the setup procedure specific to the demo project (PubSub).

5.1 Preparation

None

5.2 Importing the Project

Import the aws_ryz024a_rl78g23-fpb project to e² studio. Open the Import wizard according to the following process.

File > Import... > Existing Projects into Workspace > Next

Next, select the aws_ryz024a_rl78g23-fpb project. **Ensure that copy projects into workspace is not selected.** Then click the Finish button.

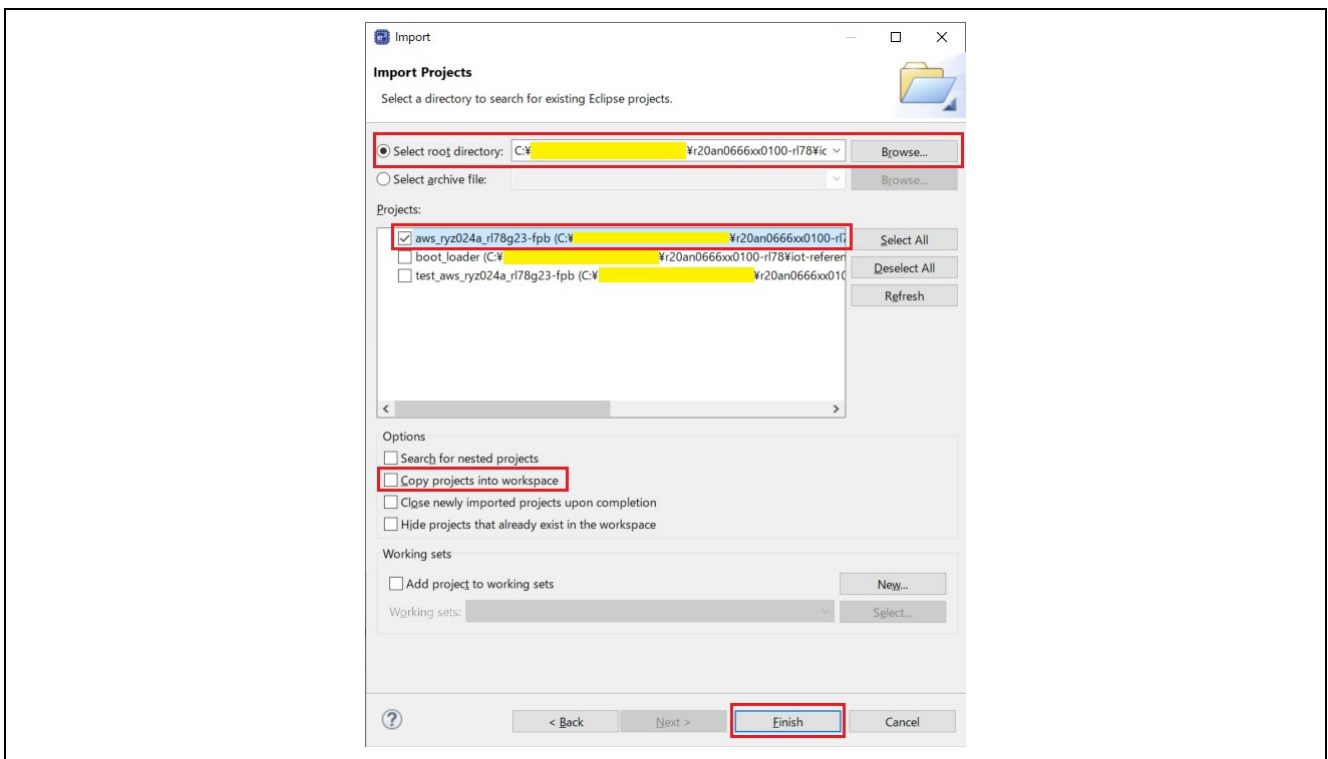


Figure 5-1 Selecting the aws_ryz024a_rl78g23-fpb Project

The imported project is showed in the Project Explorer view.



Figure 5-2 Completing to Import the aws_ryz024a_rl78g23-fpb Project

5.3 Setting the Build Configuration

Activate the build configuration “HardwareDebug” of the aws_ryz024a_rl78g23-fpb project.

Build Configurations > Set Active > Select “HardwareDebug”

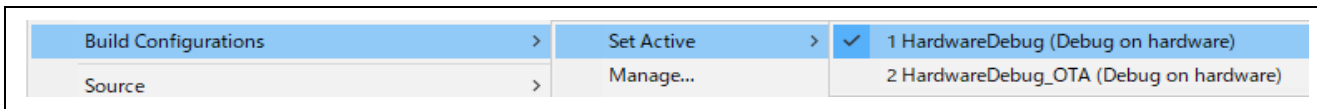


Figure 5-3 Activating Build Configuration “HardwareDebug”

5.4 Building the Demo Project

Build the aws_ryz024a_rl78g23-fpb project to create a MOT file.

Then, make sure that aws_ryz024a_rl78g23-fpb.mot has been created in the HardwareDebug folder directly under the project folder.

5.5 Preparing the MQTT Test Client

Access to the AWS Management Console, then subscribe “pubsub_demo” in the MQTT test client in the IoT Core service so that messages sent from the MCU board can be checked in text format.

(1) Select the “Subscribe to a topic” tab.

AWS IoT > MQTT test client > Select “Subscribe to a topic”

(2) Enter “pubsub_demo/#” for the topic filter, and then click “Subscribe”.

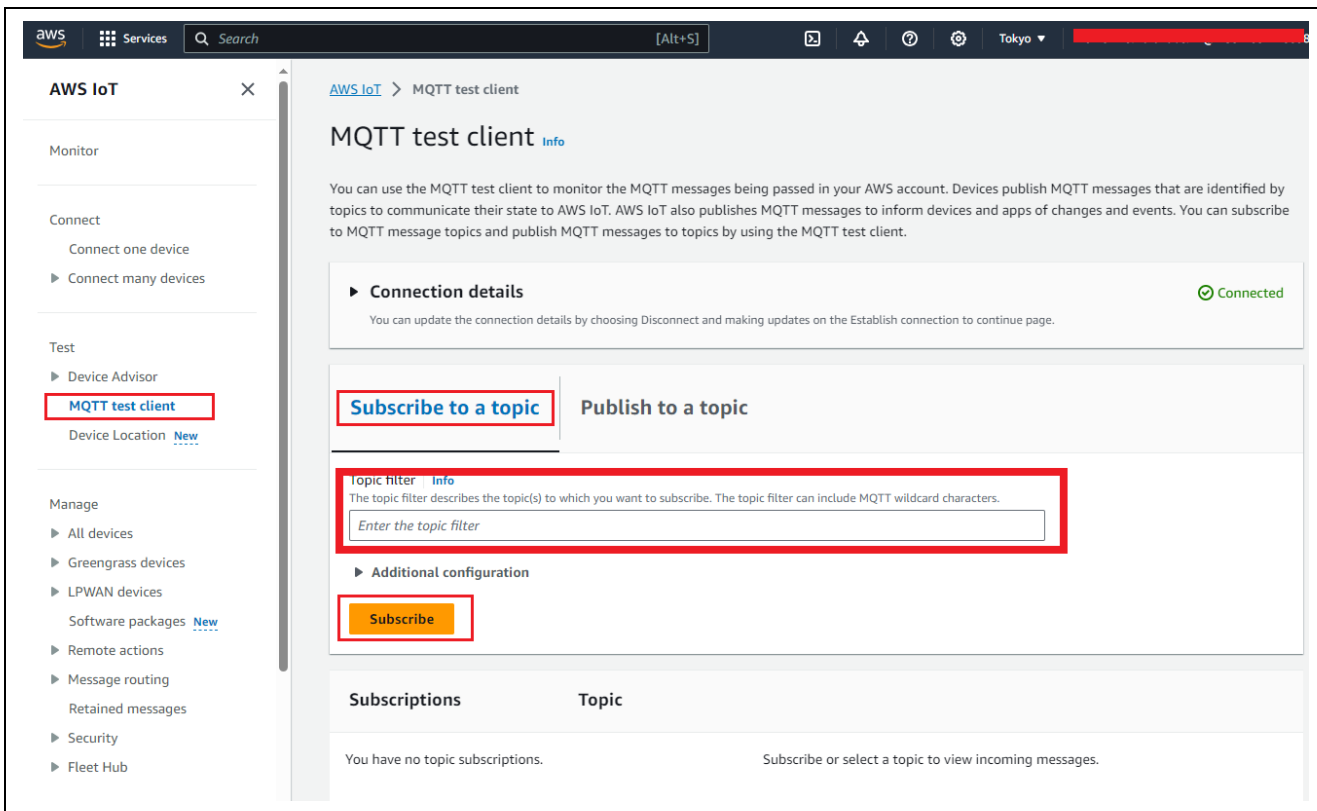


Figure 5-4 MQTT Test Client Settings

5.6 Running the Demo Project

The following describes the running procedure for the demo project (PubSub).

- (1) Use Renesas Flash Programmer to program `aws_ryz024a_rl78g23-fpb.mot` to the MCU board.
For the programming method, refer to Chapter 7, Using Renesas Flash Programmer.

- (2) When programming terminates, the demo project (PubSub) starts.

Check the terminal to make sure that the message transmission results of PubSub Demo Task0 and PubSub Demo Task1 are successful.

```

4 8404 [MAIN_TASK] -----STARTING DEMO-----
5 8408 [MQTT] [INFO] -----Start MQTT Agent Task-----
6
8 8409 [MQTT] [INFO] Creating a TLS connection to a3lknw40j1phd-ats.iot.ap-northeast-1.amazonaws.com:8883.
7 8318 [MQTT] [INFO] Creating an MQTT connection to the broker.
8 9182 [MQTT] [INFO] MQTT connection established with the broker.
9 9182 [MQTT] [INFO] Successfully connected to MQTT broker.
10 9183 [PUBSUB] [INFO] -----Start PubSub Demo Task 0-----
11 9184 [PUBSUB] [INFO] -----Start PubSub Demo Task 1-----
12 9184 [13 9185 [PUBSUB] [INFO] Sending subscribe request to agent for topic filter: pubsub_demo/rx-ota-firm-things-rx85n-rsk/PUBSUB]
14 9334 [PUBSUB] [INFO] Successfully subscribed to topic: pubsub_demo/rx-ota-firm-things-rx85n-rsk/task_1
15 9335 [PUBSUB] [INFO] Sending publish request on topic "pubsub_demo/rx-ota-firm-things-rx85n-rsk/task_1"
16 10433 [MQTT] [INFO] Publishing message to pubsub_demo/rx-ota-firm-things-rx85n-rsk/task_1.
17 11137 [PUBSUB] [INFO] Successfully subscribed to topic: pubsub_demo/rx-ota-firm-things-rx85n-rsk/task_0
18 11138 [PUBSUB] [INFO] Sending publish request on topic "pubsub_demo/rx-ota-firm-things-rx85n-rsk/task_0"
19 11156 [MQTT] [INFO] Ack packet deserialized with result: MQTTSuccess.
20 11157 [MQTT] [INFO] State record updated. New state=MQTTPublishDone.
21 11158 [PUBSUB] [INFO] Successfully sent QoS 1 publish to topic: pubsub_demo/rx-ota-firm-things-rx85n-rsk/task_1 (PassCount:1, FailCount:0).
22 11168 [MQTT] [INFO] Publishing message to pubsub_demo/rx-ota-firm-things-rx85n-rsk/task_0.
23 11845 [PUBSUB] [INFO] Successfully sent QoS 0 publish to topic: pubsub_demo/rx-ota-firm-things-rx85n-rsk/task_0 (PassCount:1, FailCount:0).
24 11876 [MQTT] [INFO] De-serialized incoming PUBLISH packet. DeserializerResult=MQTTSuccess.
25 11877 [MQTT] [INFO] State record updated. New state=MQTTPubAckSend.
26 11878 [MQTT] [INFO] Received incoming publish message Task 1 publishing message 0
27 11845 [MQTT] [INFO] De-serialized incoming PUBLISH packet. DeserializerResult=MQTTSuccess
    
```

Figure 5-5 Checking Demo Project Execution Results on the Terminal

- (3) Use the MQTT test client to make sure that the messages sent from PubSub Demo Task0 and PubSub Demo Task1 are displayed.

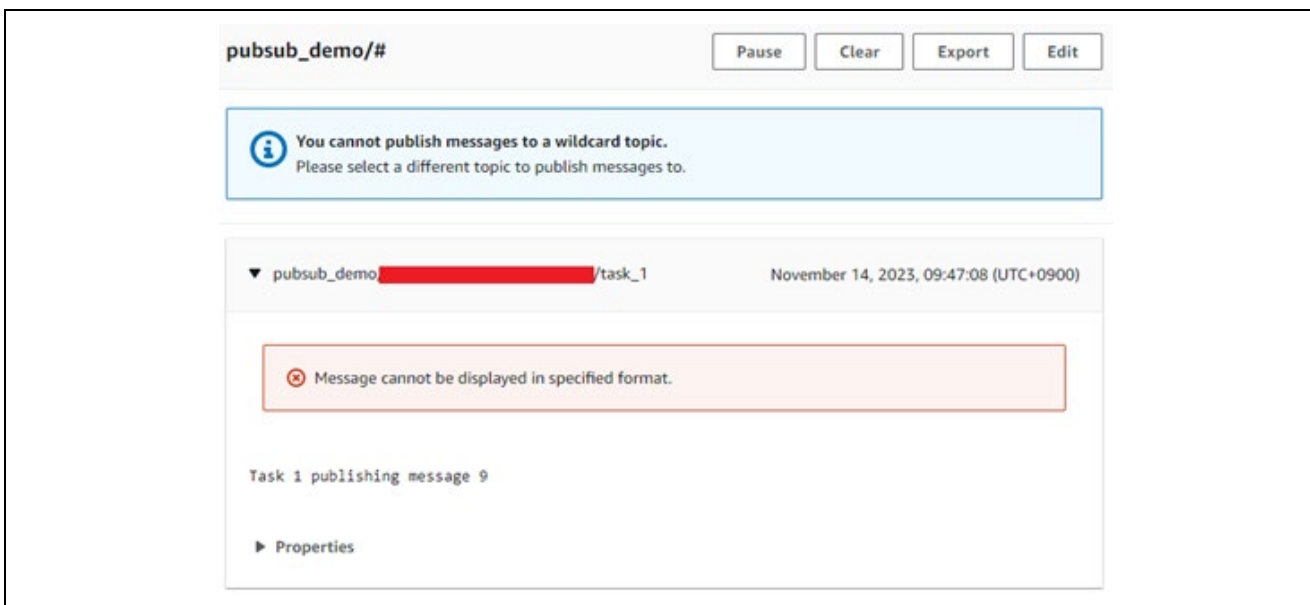


Figure 5-6 Checking Demo Project Execution Results with the MQTT Test Client

5.7 Debugging the Demo Project

The following describes the procedure for starting the demo project (PubSub) from e² studio and debugging it.

(1) Build the demo project.

Refer to section 5.2, Importing the Project, section 5.3, Setting the Build Configuration, and section 5.4, Building the Demo Project.

(2) Start debugging.

Refer to Chapter 8, Debug Procedure.

6. Setup Specific to Demo Project (OTA)

This demo project connects to the AWS from the MCU board, and then performs firmware update by using AWS IoT OTA. This chapter describes the setup procedure.

6.1 Preparation

6.1.1 Installing Tools

Install the tools necessary for running the demo project.

(1) Install Python

1. Python is required for operation of Renesas Image Generator. Install version 3.9.0 or later. You can download Python from <https://www.python.org/>.
2. After installing Python, install the package pycryptodome by using the following command:

```
> pip install pycryptodome
```

(2) Install OpenSSL

Create the key necessary for verifying the code signature when creating an initial image and update image. Use OpenSSL to create the key.

1. If OpenSSL is not installed, open the following URL on your browser:
[Win32/Win64 OpenSSL Installer for Windows - Shining Light Productions \(slproweb.com\)](http://www.slproweb.com/win32/win64/openssl_installer_for_windows/)
2. Download and install Win64OpenSSL v3.x.x Light.

(3) Download Renesas Image Generator

Download Renesas Image Generator (V3.03) contained in the [RL78/G22,RL78/G23,RL78/G24 firmware update module](#).

6.1.2 Generating Keys for Signature Generation and Verification

Use OpenSSL to generate firmware verification keys. The parts highlighted in yellow indicate the commands to be entered.

(1) CA certificate

```
$ openssl ecparam -genkey -name secp256r1 -out ca.key
using curve name prime256v1 instead of secp256r1
$ openssl req -x509 -sha256 -new -nodes -key ca.key -days 3650 -out ca.crt
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]:JP
State or Province Name (full name) [Some-State]:Tokyo
Locality Name (eg, city) []:Kodaira
Organization Name (eg, company) [Internet Widgits Pty Ltd]:Renesas Electronics
Organizational Unit Name (eg, section) []:Software Development Division
Common Name (e.g. server FQDN or YOUR name) []:Renesas Tarou
Email Address []:Tarou.Renesas@sample.com
```

(2) Elliptic curve cryptography (secp256r1) key pair

```
$ openssl ecparam -genkey -name secp256r1 -out secp256r1.keypair
using curve name prime256v1 instead of secp256r1
```

(3) Key pair certificate

```
$ openssl req -new -sha256 -key secp256r1.keypair > secp256r1.csr
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]:JP
State or Province Name (full name) [Some-State]:Tokyo
Locality Name (eg, city) []:Kodaira
Organization Name (eg, company) [Internet Widgits Pty Ltd]:Renesas Electronics
Organizational Unit Name (eg, section) []:Software Development Division
Common Name (e.g. server FQDN or YOUR name) []:Renesas Tarou
Email Address []:Tarou.Renesas@sample.com
Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:
An optional company name []:
```

(4) Generating a key pair certificate by using the CA certificate

```
$ openssl x509 -req -sha256 -days 3650 -in secp256r1.csr -CA ca.crt -CAkey ca.key -  
CAcreateserial -out secp256r1.crt  
Signature ok  
subject=C = JP, ST = Tokyo, L = Kodaira, O = Renesas Electronics, OU = Software  
Development Division, CN = Renesas Tarou, emailAddress = Tarou.Renesas@sample.com  
Getting CA Private Key
```

(5) Extracting the elliptic curve cryptography (secp256r1) private key

```
$ openssl ec -in secp256r1.keypair -outform PEM -out secp256r1.privatekey  
read EC key  
writing EC key
```

(6) Extracting the elliptic curve cryptography (secp256r1) public key

```
$ openssl ec -in secp256r1.keypair -outform PEM -pubout -out secp256r1.publickey  
read EC key  
writing EC key
```

6.1.3 Settings for OTA Update

6.1.3.1 Creating Amazon S3 Buckets

(1) Amazon S3 > Buckets > "Create bucket"

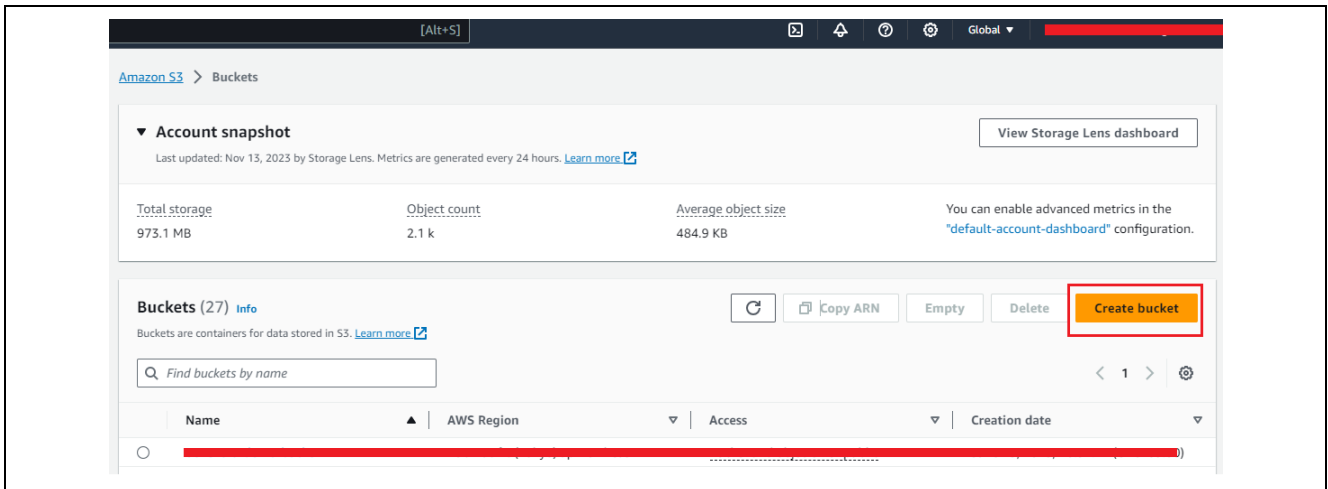


Figure 6-1 Create Bucket

(2) General configuration

- Bucket name: Your bucket name
- AWS Region: Asia Pacific (Tokyo) ap-northeast-1

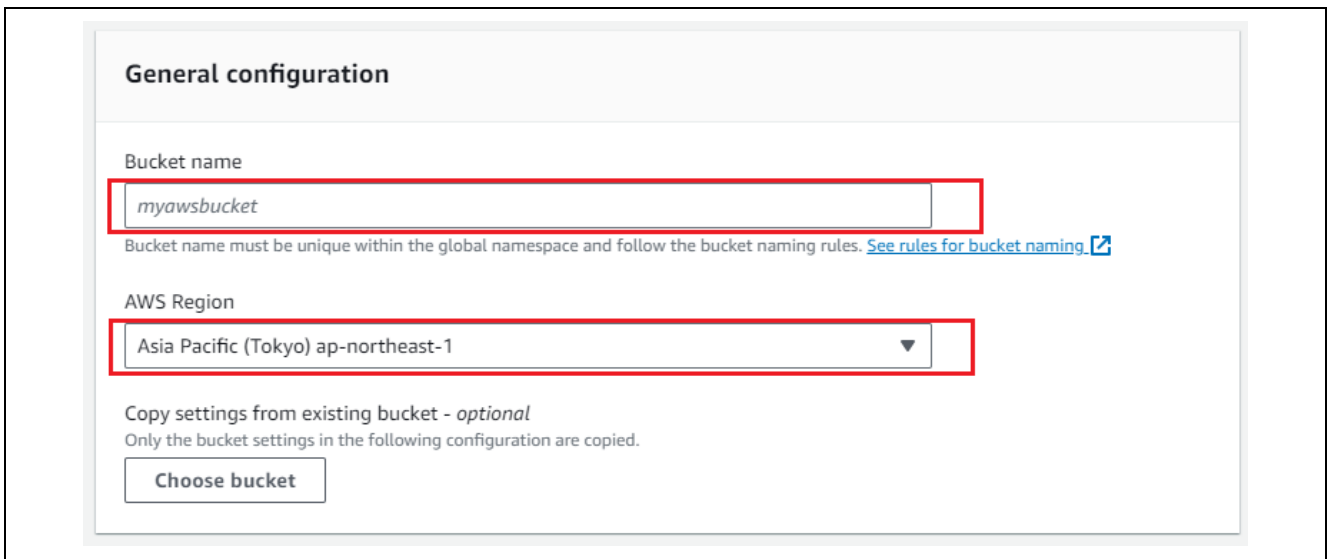


Figure 6-2 General Configuration

(3) Object Ownership

- Choose ACLs disabled

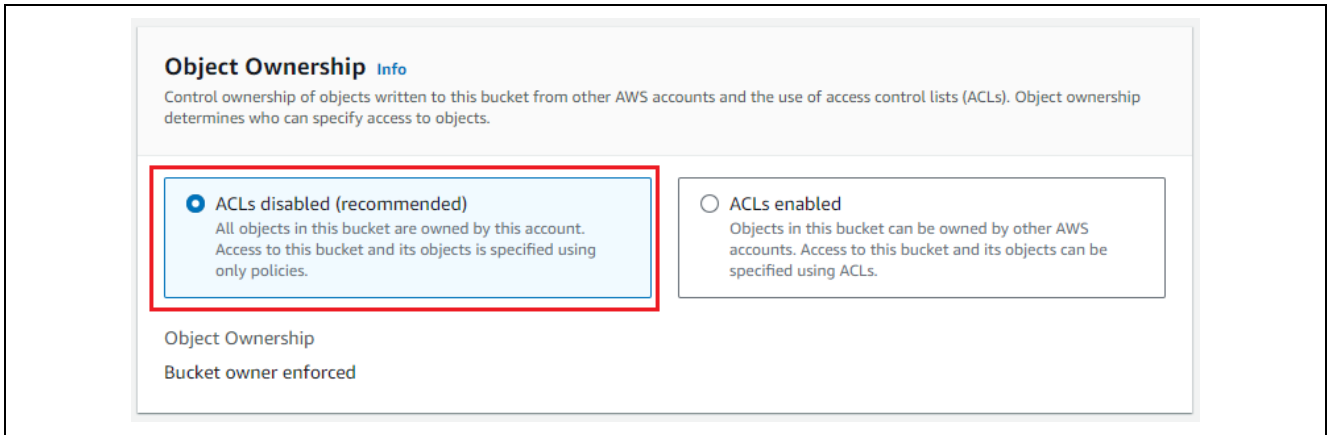


Figure 6-3 Object Ownership

(4) Block Public Access settings for this bucket

- Choose Block all public access

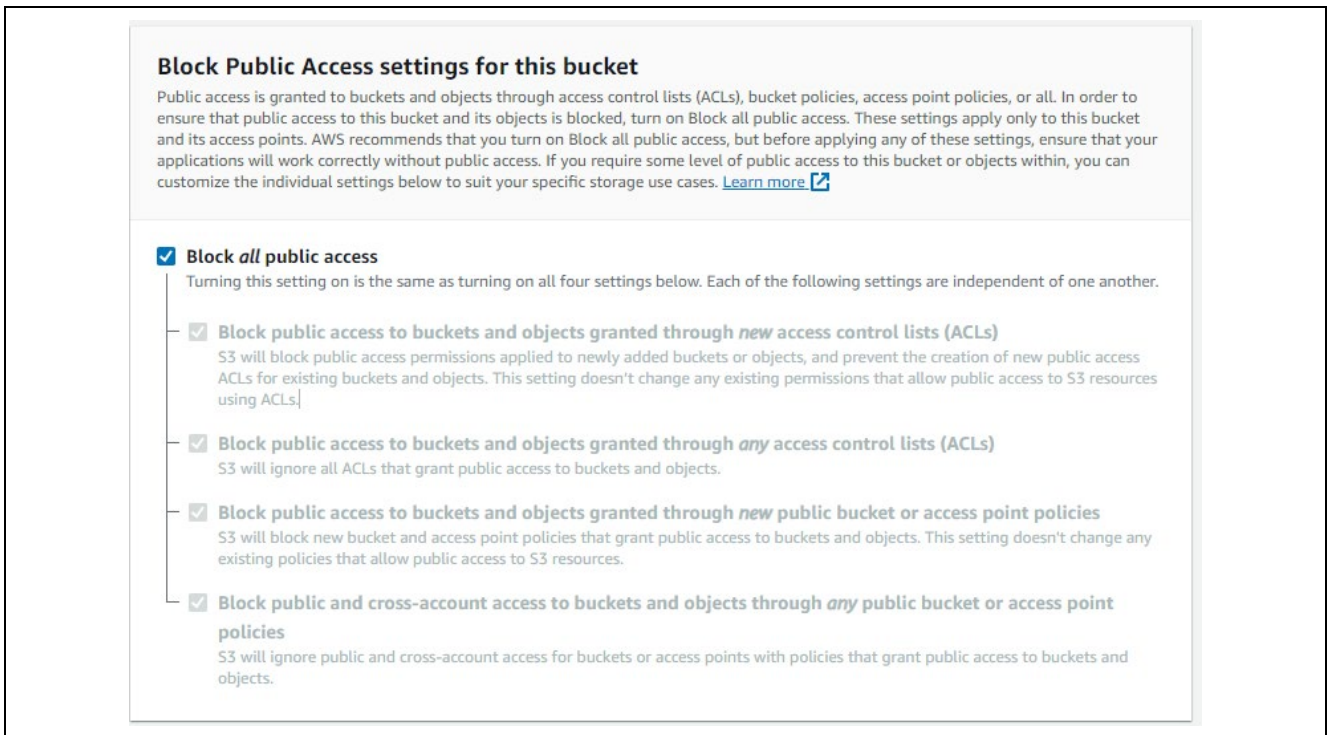


Figure 6-4 Block Public Access Settings for this bucket

(5) Bucket Versioning

- Bucket Versioning: Disable

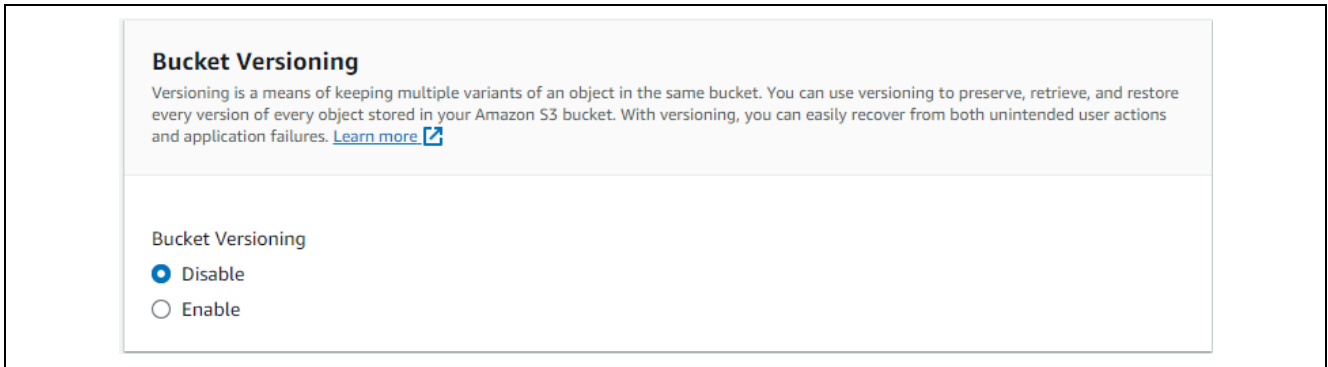


Figure 6-5 Bucket Versioning

(6) Default encryption

- Encryption type: Server-side encryption with Amazon S3 managed keys (SSE-S3)
- Bucket Key: Enable

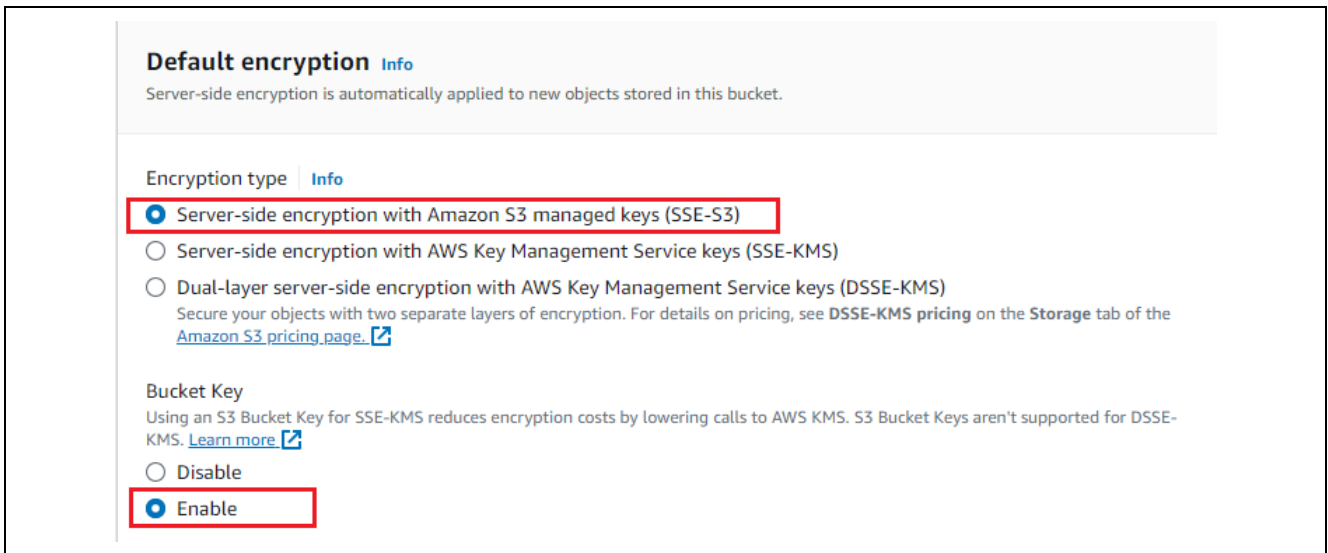


Figure 6-6 Default encryption

(7) Click “Create bucket”

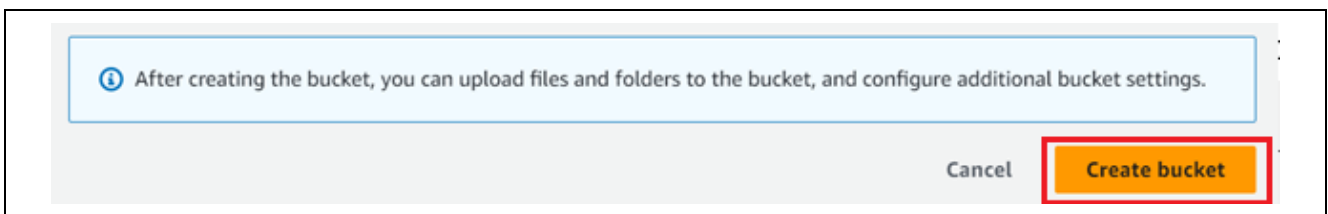


Figure 6-7 Clicking “Crate bucket”

6.1.3.2 Creating an OTA Update Service Role

(1) IAM > Roles > “Create role”

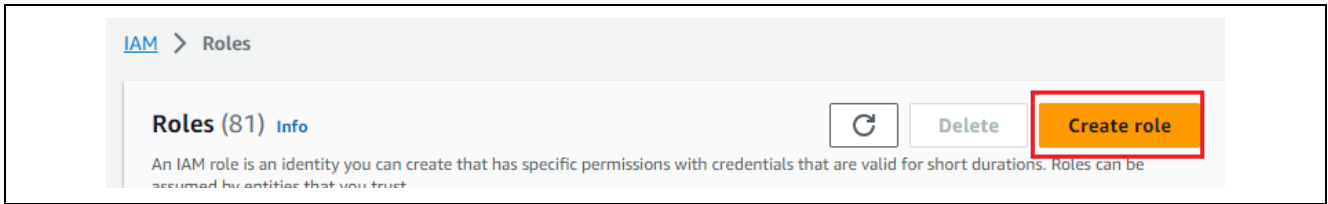


Figure 6-8 IAM > Roles > Create role

(2) Step 1: Select trusted entity

- Trusted entity type: AWS service
- Use case: Service or use case > IoT

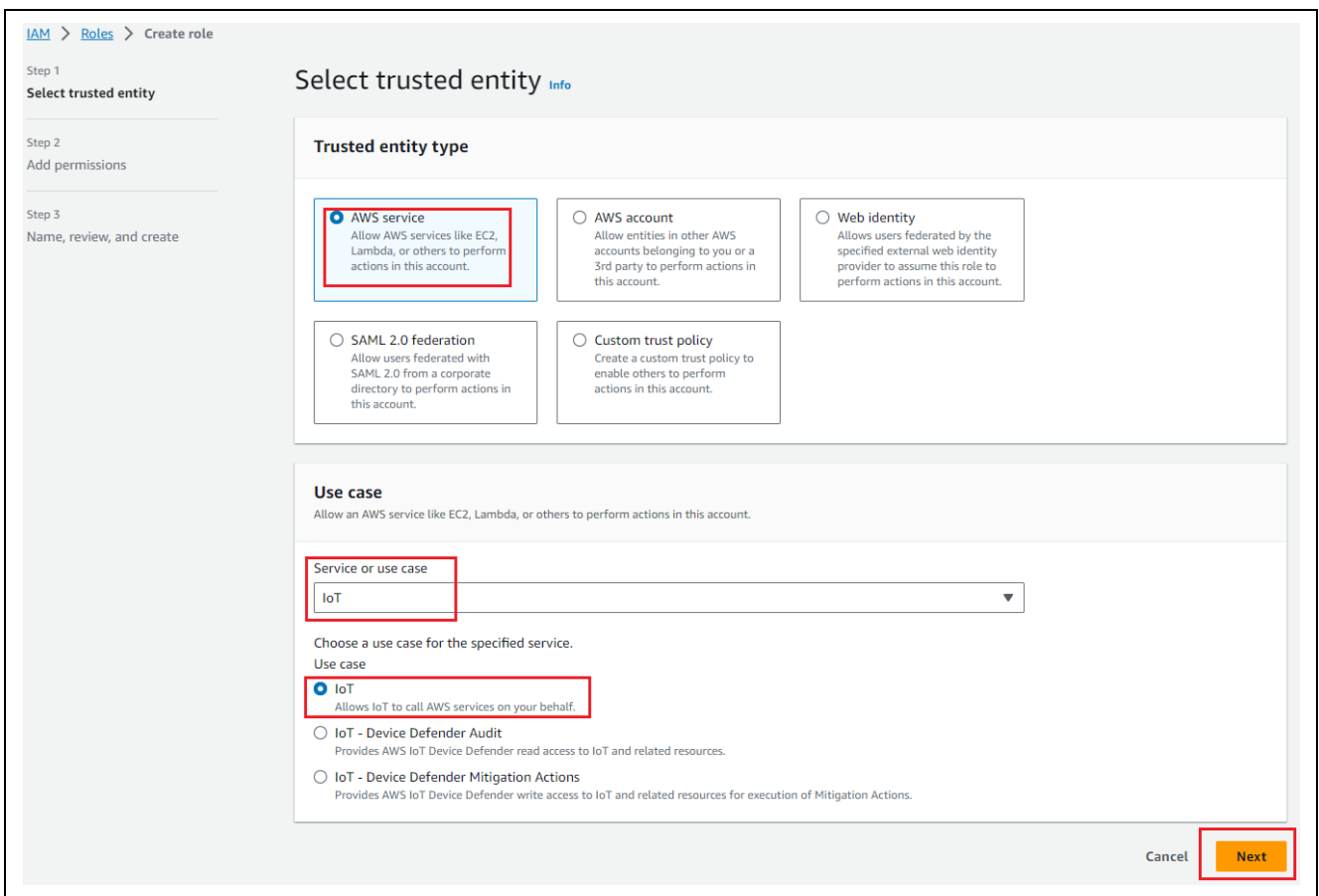


Figure 6-9 Step 1: Select trusted entity

(3) Step 2: Add permissions

- AWSIoTLogging
- AWSIoTRuleActions
- AWSIoTThingsRegistration

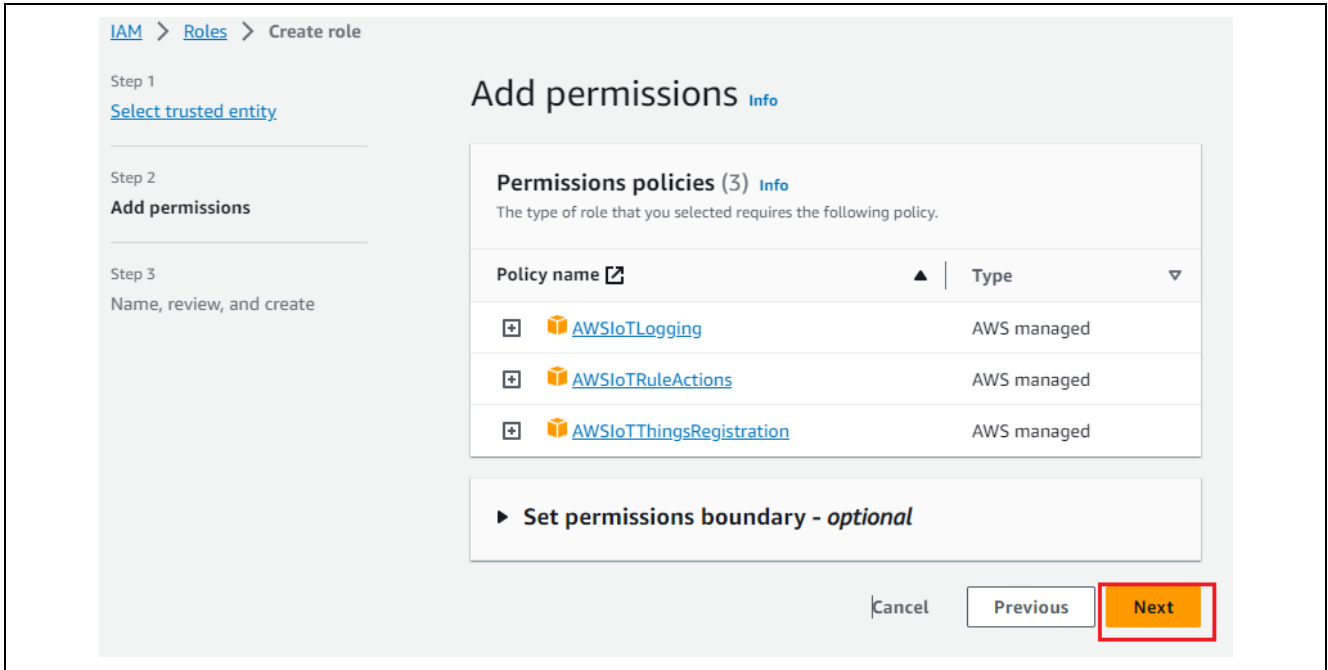


Figure 6-10 Step 2: Add permissions

(4) Step 3: Name, review, and create > Role details

- Role name: Any
- Description: Any

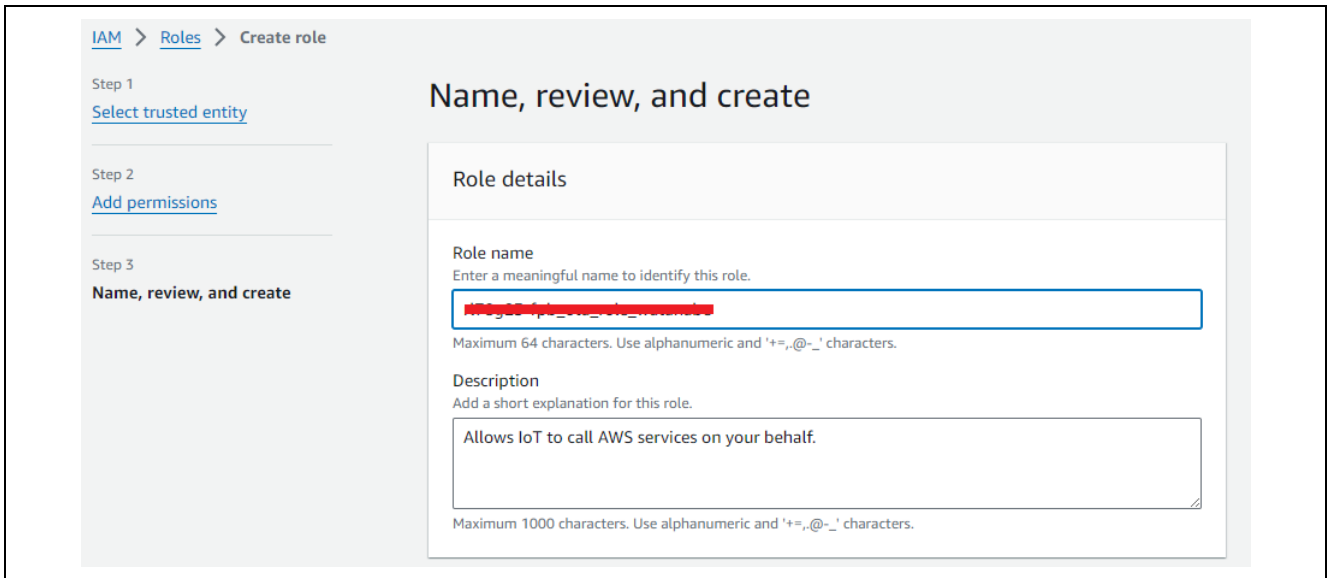


Figure 6-11 Step 3: Name, review, and create > Role details

(5) Step 3: Name, review, and create > Step 1: Selected trusted entities

- Default

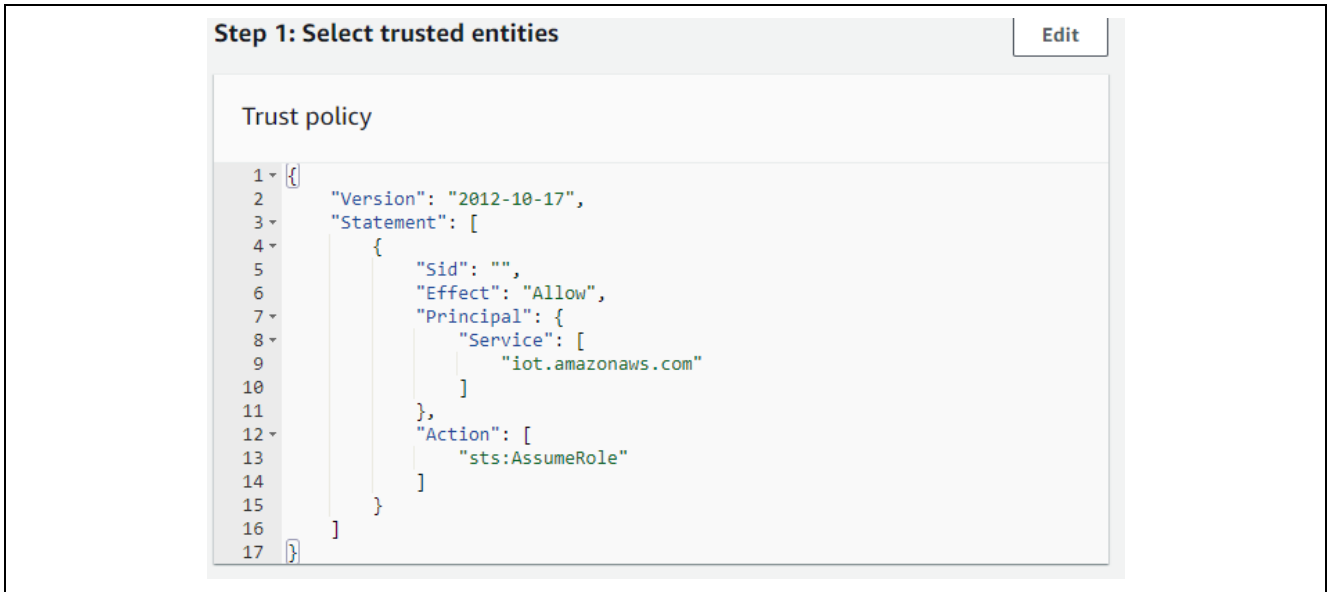


Figure 6-12 Step 3: Name, review, and create > Step 1: Selected trusted entities

(6) Step 3: Name, review, and create > Step 2: Add permissions

- Default

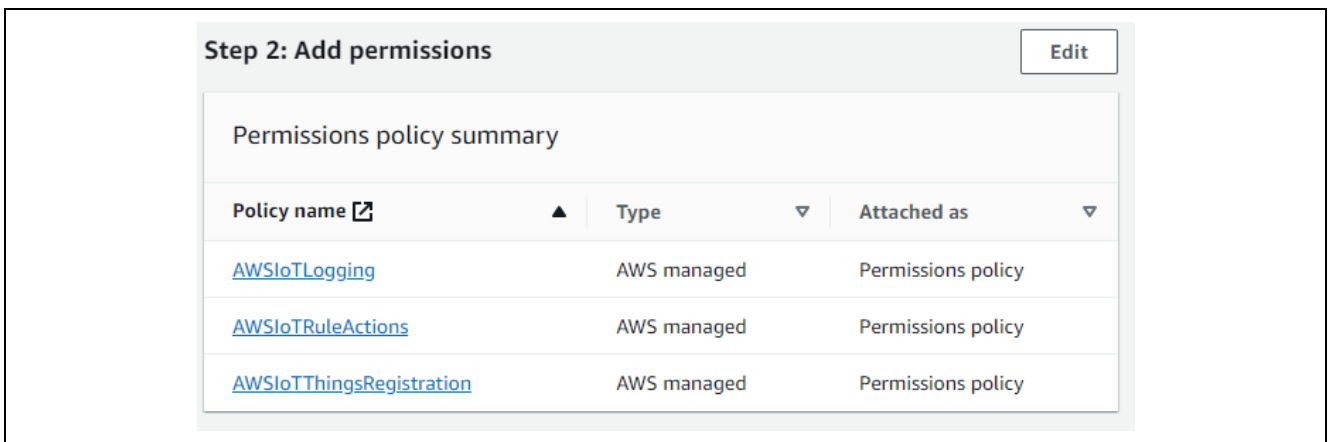


Figure 6-13 Step 3: Name, review, and create > Step 2: Add permissions

(7) Step 3: Name, review, and create > Step 3: Add tags

- Default
- Click “Create role”

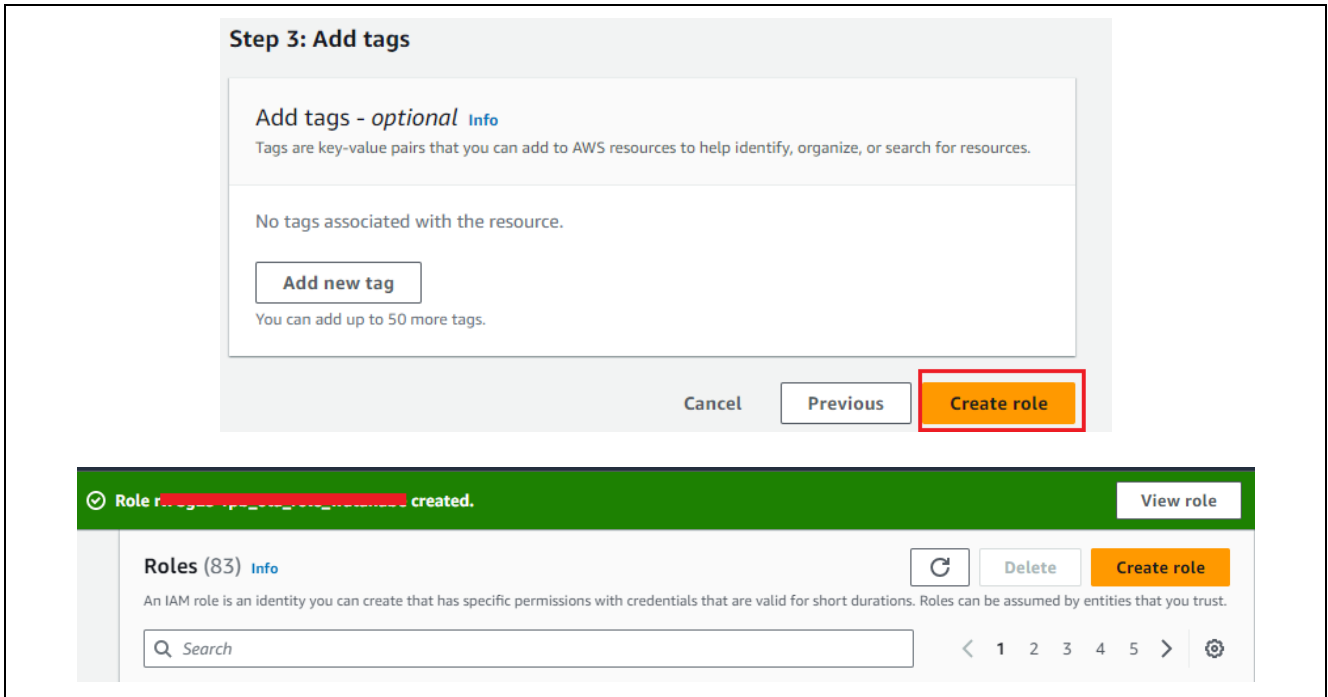


Figure 6-14 Step 3: Name, review, and create > Step 3: Add tags

6.1.3.3 Creating an OTA Update User Policy

(1) Click to open the role created in section 6.1.3.2, Creating an OTA Update Service Role.

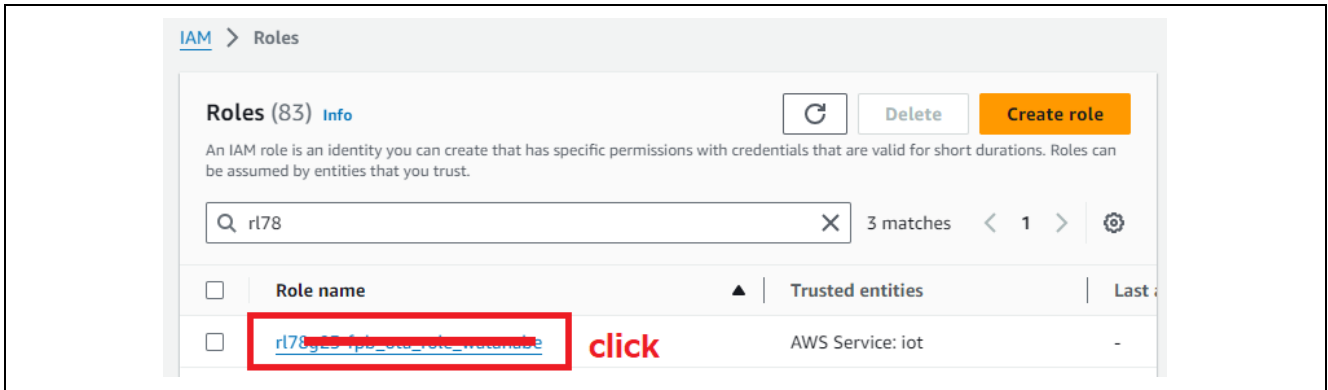


Figure 6-15 Opening the Created OTA Update Service Role

(2) My role > Summary: Default

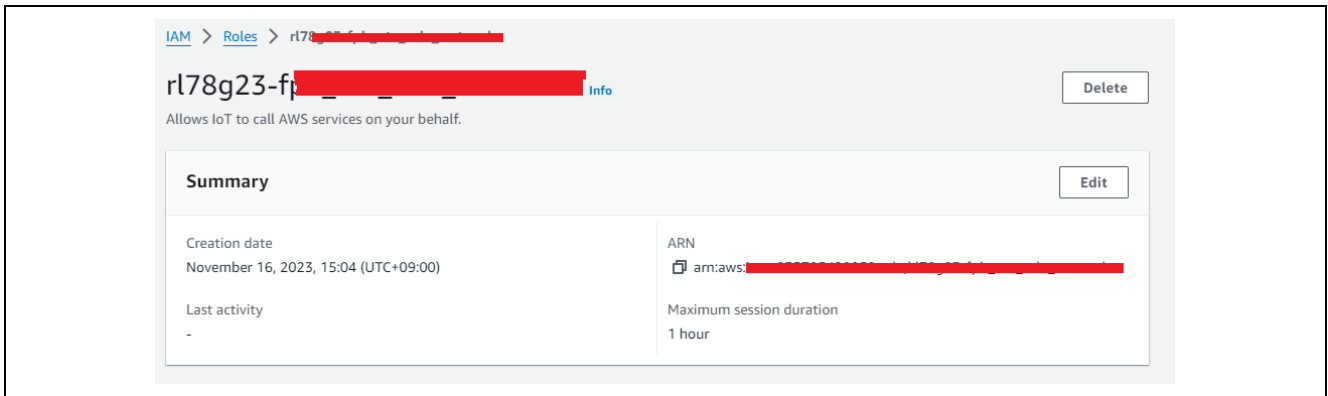


Figure 6-16 Displaying the Summary of the Created OTA Update Service Role

(3) Attach the policy “AmazonFreeRTOSOTAUpdate”.

- My role > Permissions policies > “Add permissions” > Attach policies

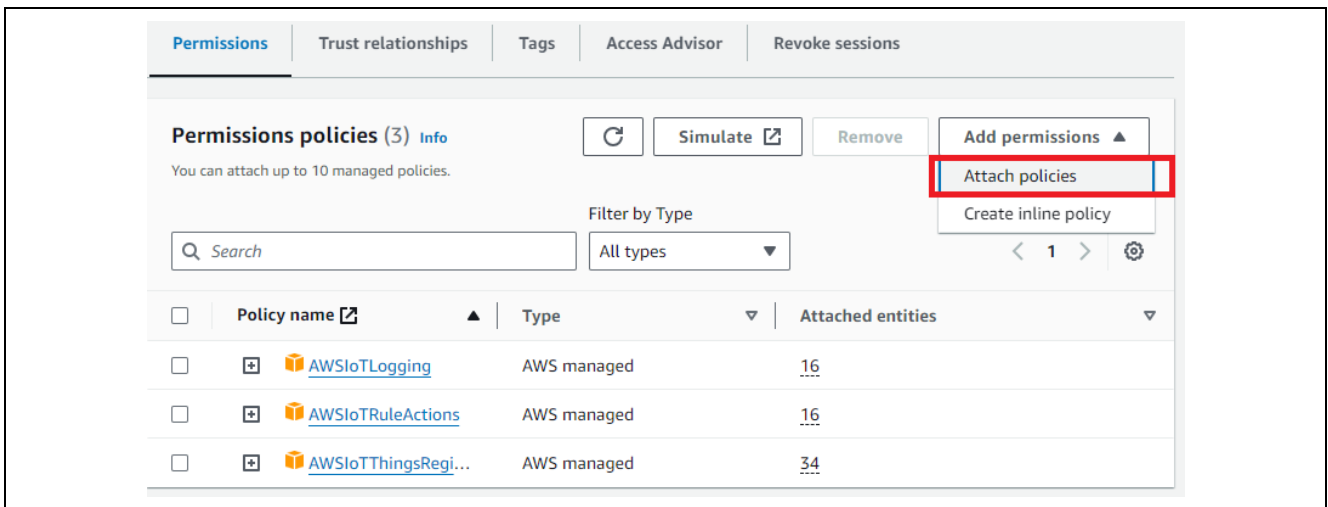


Figure 6-17 Attaching Policies to the Created OTA Update Service Role

- Choose “AmazonFreeRTOSOTAUpdate” > “Add permissions”

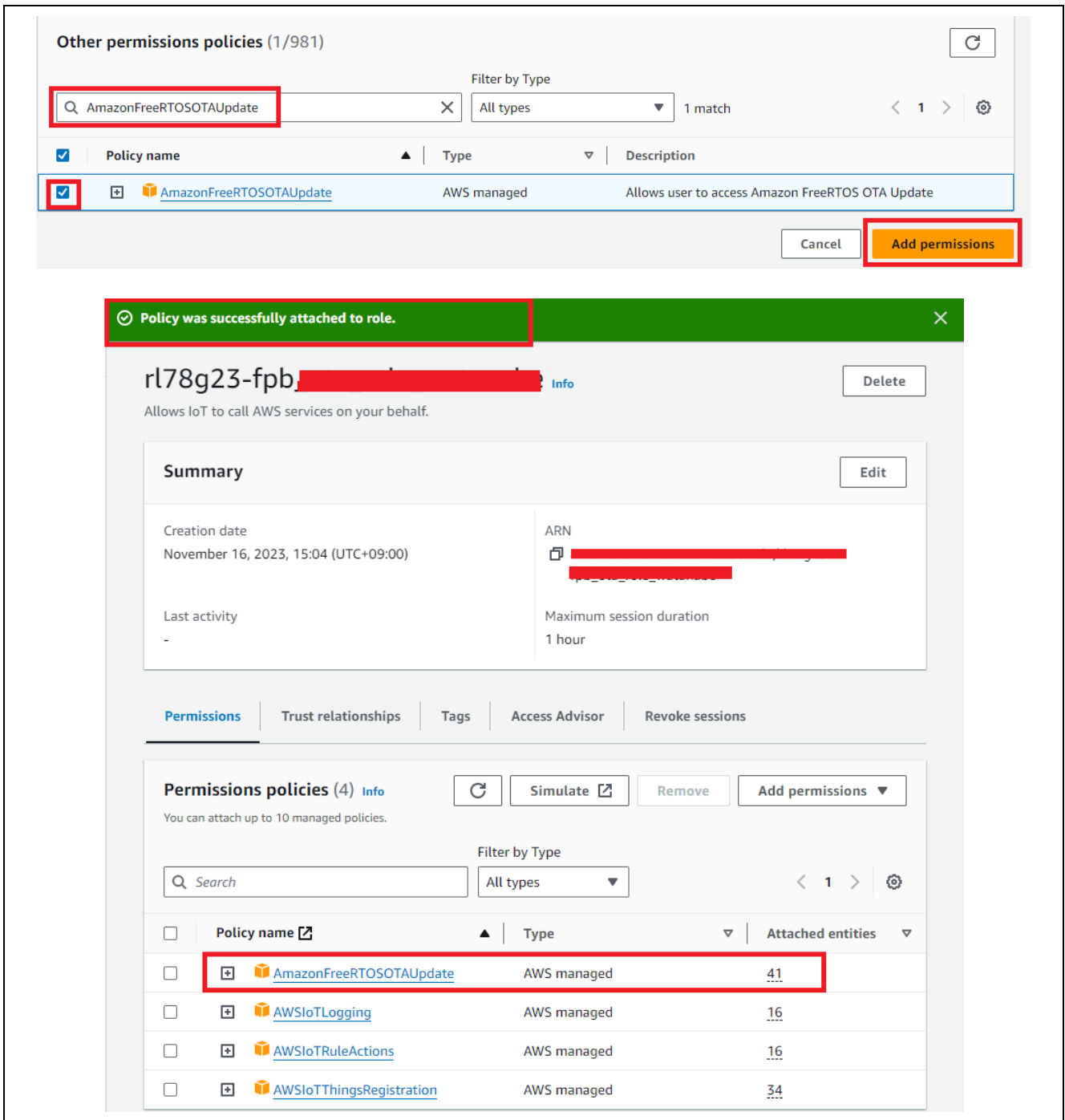


Figure 6-18 Attaching the Policy “AmazonFreeRTOSOTAUpdate” to the Created OTA Update Service Role

(4) Add the inline policy (S3).

- “Add permissions” > Create inline policy > “JSON”

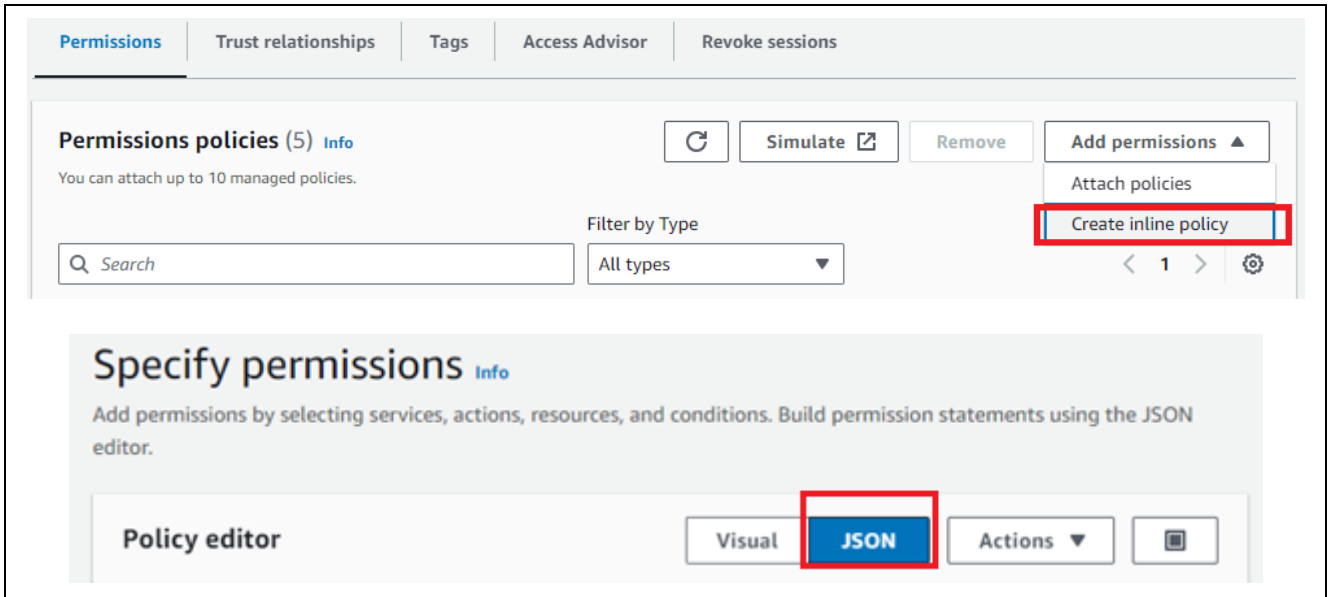


Figure 6-19 Creating an S3 Inline Policy

- Paste the following information to the Policy editor, and then click "Next".
 — Change s3-bucket-test to the bucket name created in section 6.1.3.1, Creating Amazon S3 Buckets.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetObjectVersion",
        "s3:GetObject",
        "s3:PutObject"
      ],
      "Resource": "arn:aws:s3:::s3-bucket-test/*"
    }
  ]
}
```

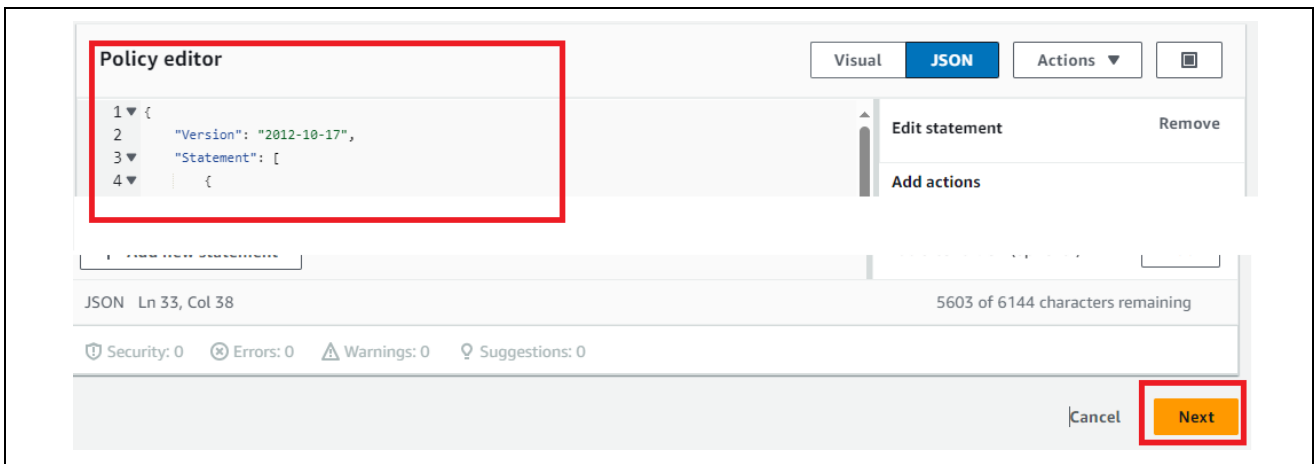


Figure 6-20 Adding S3 Policies to the Policy Editor

- Policy name: Any (Example: inline-policy-s3-test) > "Create policy"

Review and create [Info](#)

Review the permissions, specify details, and tags.

Policy details

Policy name
Enter a meaningful name to identify this policy.

inline-policy-s3-test

Maximum 128 characters. Use alphanumeric and '+,.,@,_' characters.

Permissions defined in this policy [Info](#) [Edit](#)

Permissions defined in this policy document specify which actions are allowed or denied. To define permissions for an IAM identity (user, user group, or role), attach a policy to it

Q Search

Allow (1 of 384 services) Show remaining 383 services

| Service | Access level | Resource | Resource type |
|--------------------|----------------------|---|---------------|
| S3 | Limited: Read, Write | BucketName string like [s3-bucket-test, ObjectPath] string like All | N |

Cancel Previous **Create policy**

Permissions Trust relationships Tags Access Advisor Revoke sessions

Permissions policies (5) [Info](#) [Refresh](#) [Simulate](#) [Remove](#)

You can attach up to 10 managed policies.

Q Search Filter by Type All types

| Policy name | Type | Attached to |
|---|-----------------|-------------|
| <input type="checkbox"/> AmazonFreeRTOSOTAUpdate | AWS managed | 41 |
| <input type="checkbox"/> AWSIoTLogging | AWS managed | 16 |
| <input type="checkbox"/> AWSIoTRuleActions | AWS managed | 16 |
| <input type="checkbox"/> AWSIoTThingsRegistration | AWS managed | 34 |
| <input type="checkbox"/> inline-policy-s3-test | Customer inline | 0 |

Figure 6-21 Creating the S3 Policy with a Name (Example: inline-policy-s3-test)

(5) Add an IAM inline policy.

- “Add permissions” > Create inline policy > “JSON”

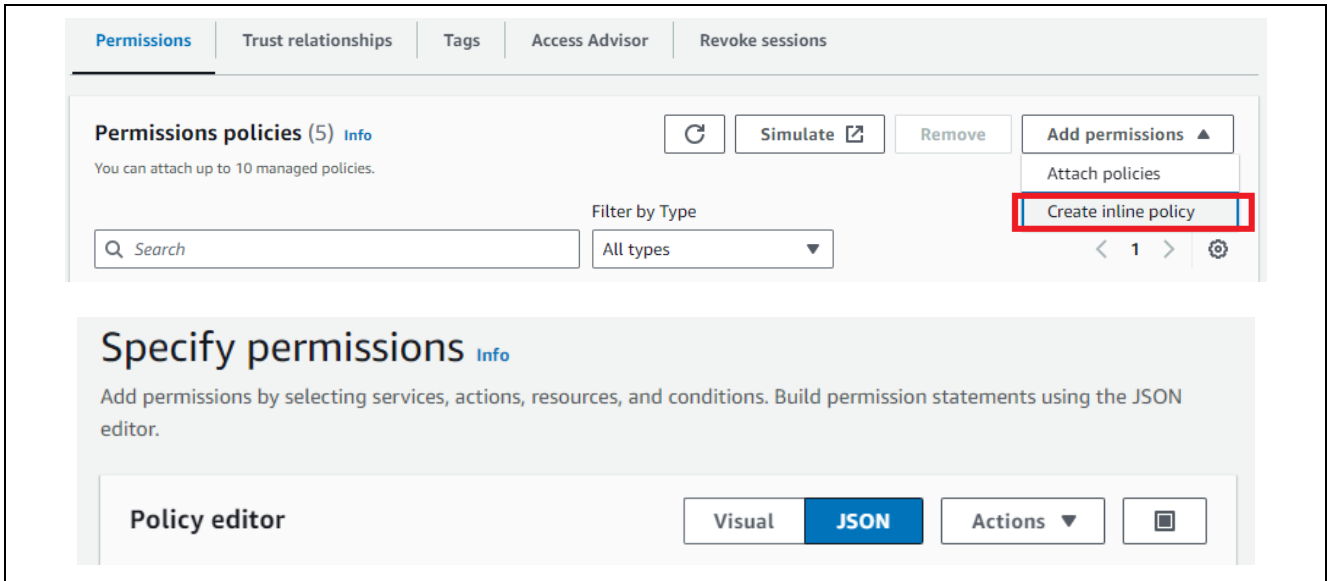


Figure 6-22 Creating an Inline Policy

- Paste the following information to the Policy editor, and then click “Next”.
 - Change ota-role-test to the role name created in section 6.1.3.2, Creating an OTA Update Service Role.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "iam:GetRole",
        "iam:PassRole"
      ],
      "Resource": "arn:aws:iam::xxxxxxxxxx:role/ota-role-test"
    }
  ]
}
```

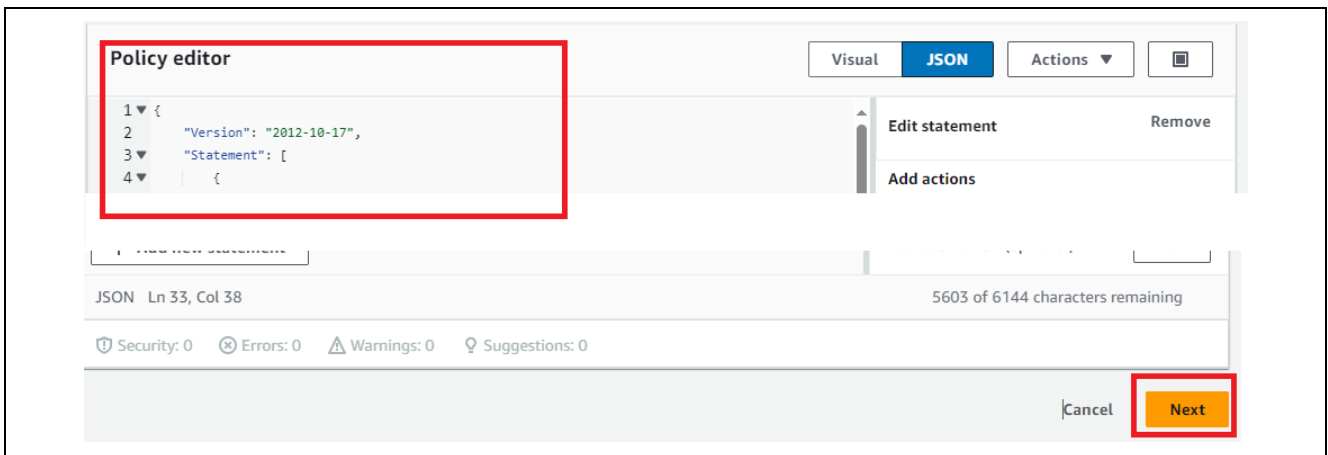


Figure 6-23 Adding the IAM Role to the Inline Policy

- Policy name: Any (Example: inline-policy-iam-test) > “Create policy”

Review and create [Info](#)
Review the permissions, specify details, and tags.

Policy details

Policy name
Enter a meaningful name to identify this policy.

Maximum 128 characters. Use alphanumeric and '+, @, _' characters.

Permissions defined in this policy [Info](#) [Edit](#)
Permissions defined in this policy document specify which actions are allowed or denied. To define permissions for an IAM identity (user, user group, or role), attach a policy to it

Allow (1 of 384 services) Show remaining 383 services

| Service | Access level | Resource | Request |
|---------------------|----------------------|--------------------------------------|---------|
| IAM | Limited: Read, Write | RoleName string like ota-role-test | None |

[Cancel](#) [Previous](#) [Create policy](#)

Permissions policies (6) [Info](#) [Refresh](#) [Simulate](#) [Remove](#) [Add permissions](#)

You can attach up to 10 managed policies.

Filter by Type: [All types](#)

| <input type="checkbox"/> | Policy name | Type | Attached entities |
|--------------------------|--|-----------------|-------------------|
| <input type="checkbox"/> | AmazonFreeRTOSOTA... | AWS managed | 41 |
| <input type="checkbox"/> | AWSIoTLogging | AWS managed | 16 |
| <input type="checkbox"/> | AWSIoTRuleActions | AWS managed | 16 |
| <input type="checkbox"/> | AWSIoTThingsRegistrat... | AWS managed | 34 |
| <input type="checkbox"/> | inline-policy-iam-test | Customer inline | 0 |
| <input type="checkbox"/> | inline-policy-s3-test | Customer inline | 0 |

Figure 6-24 Saving the IAM Inline Policy with a Name (Example: inline-policy-iam-test)

6.1.3.4 Allocating an OTA Update Policy to IAM User

(1) Create an OTA Update policy.

- IAM > Policies > "Create policy" > "JSON"

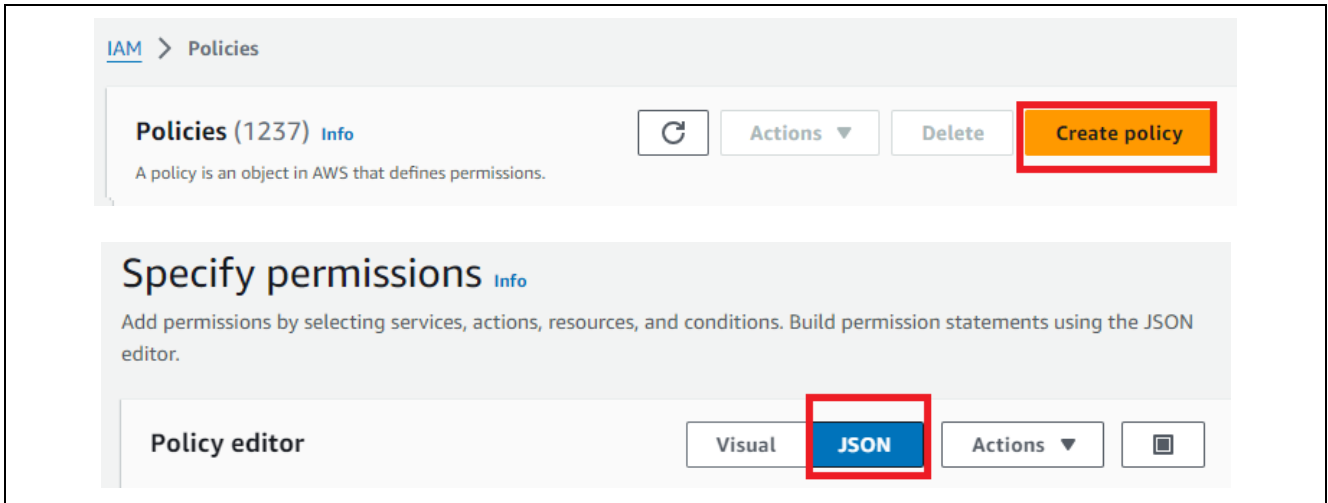


Figure 6-25 Creating an OTA Update Policy

- Paste the following information to the Policy editor, and then click "Next".
 - Change s3-bucket-test to the bucket name specified in section 6.1.3.1, Creating Amazon S3 Buckets.
 - Change ota-role-test to the role name specified in section 6.1.3.2, Creating an OTA Update Service Role.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "s3:ListBucket",
        "s3:ListAllMyBuckets",
        "s3:CreateBucket",
        "s3:PutBucketVersioning",
        "s3:GetBucketLocation",
        "s3:GetObjectVersion",
        "acm:ImportCertificate",
        "acm:ListCertificates",
        "iot:*",
        "iam:ListRoles",
        "freertos:ListHardwarePlatforms",
        "freertos:DescribeHardwarePlatform"
      ],
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetObject",
        "s3:PutObject"
      ],
      "Resource": "arn:aws:s3:::s3-bucket-test/*"
    },
    {
      "Effect": "Allow",
      "Action": "iam:PassRole",
      "Resource": "arn:aws:iam::xxxxxxxxxx:role/ota-role-test"
    }
  ]
}
```

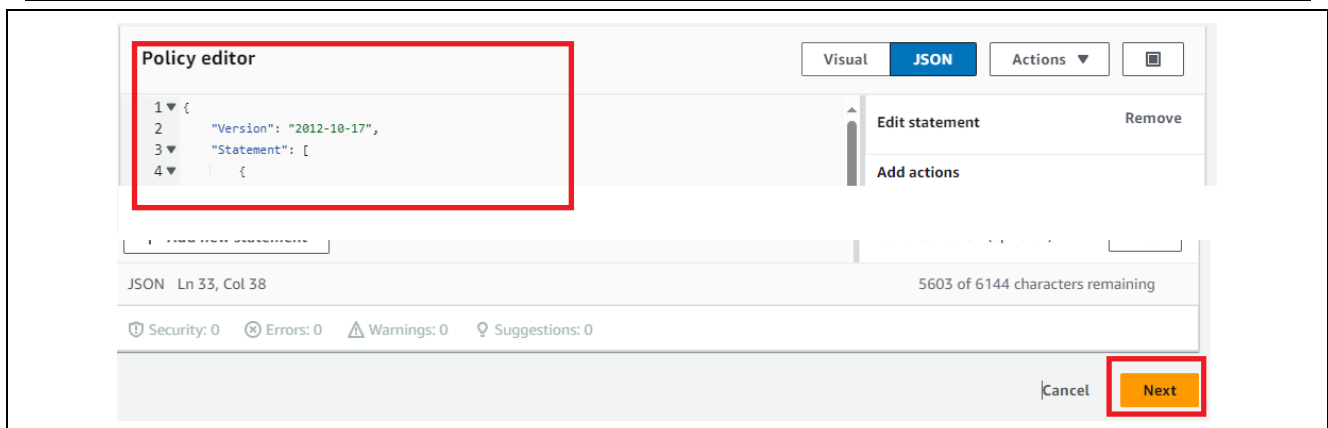


Figure 6-26 Creating an OTA Update Policy with the Policy Editor

- Policy name: Any (Example: rl78g23-fpb_ota_policy) > “Create policy”

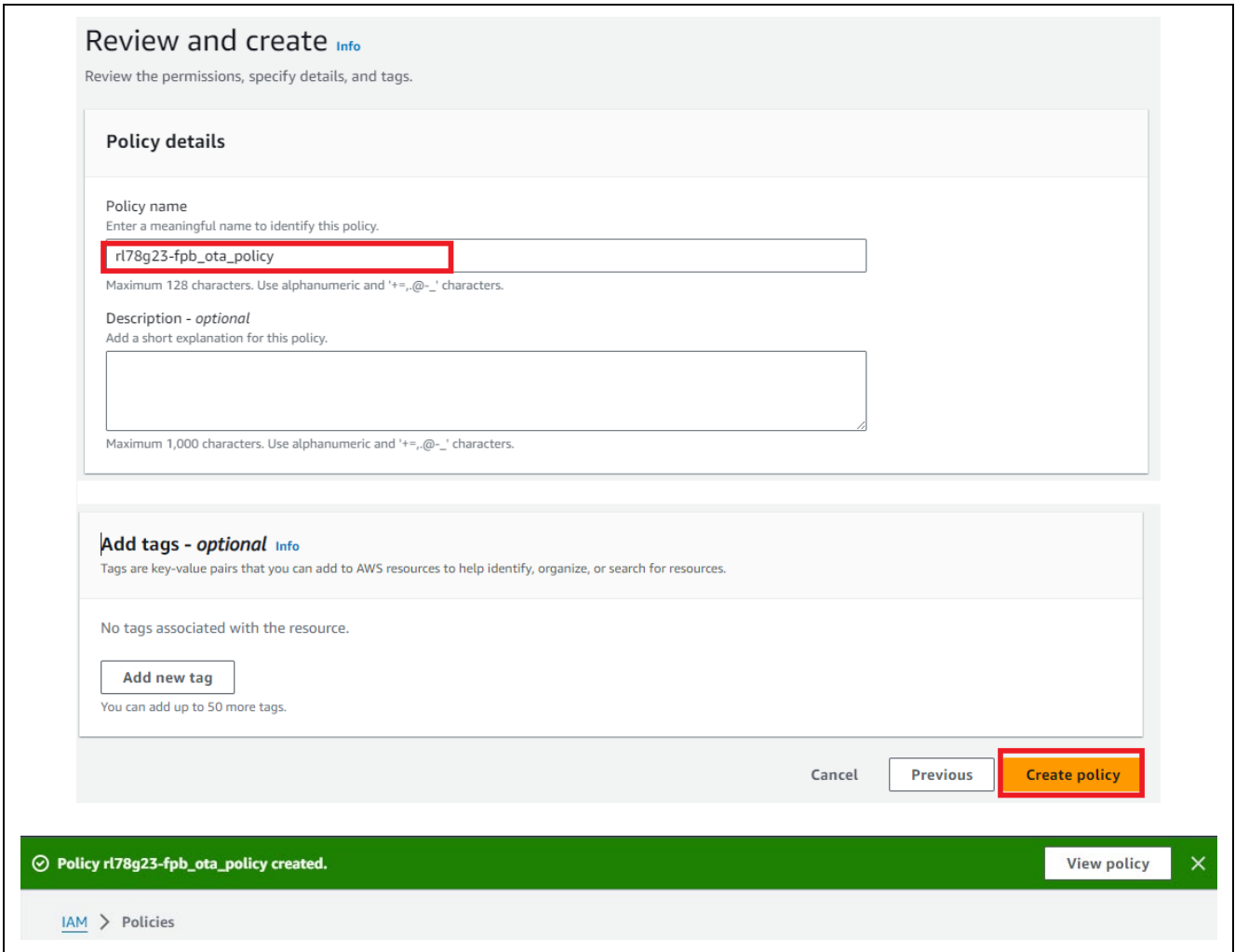


Figure 6-27 Saving the OTA Update Policy with a Name (Example: rl78g23-fpb_ota_policy)

(2) Add the created OTA Update policy to the IAM user.

- IAM > Users > Choose User > Add permissions

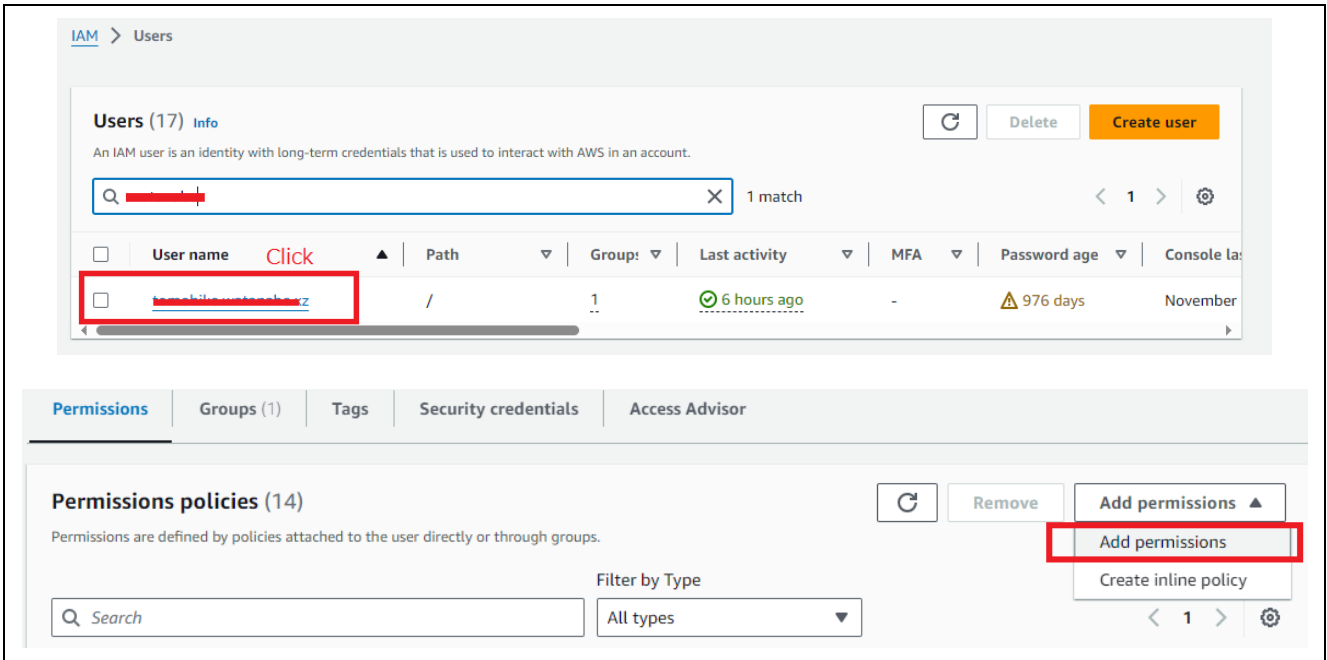


Figure 6-28 Selecting the IAM User

- Permissions options: Attach policies directly
- Permissions policies > Policy name: Name of created OTA Update policy (Example: r178g23-fpb_ota_policy)
- Click "Next"

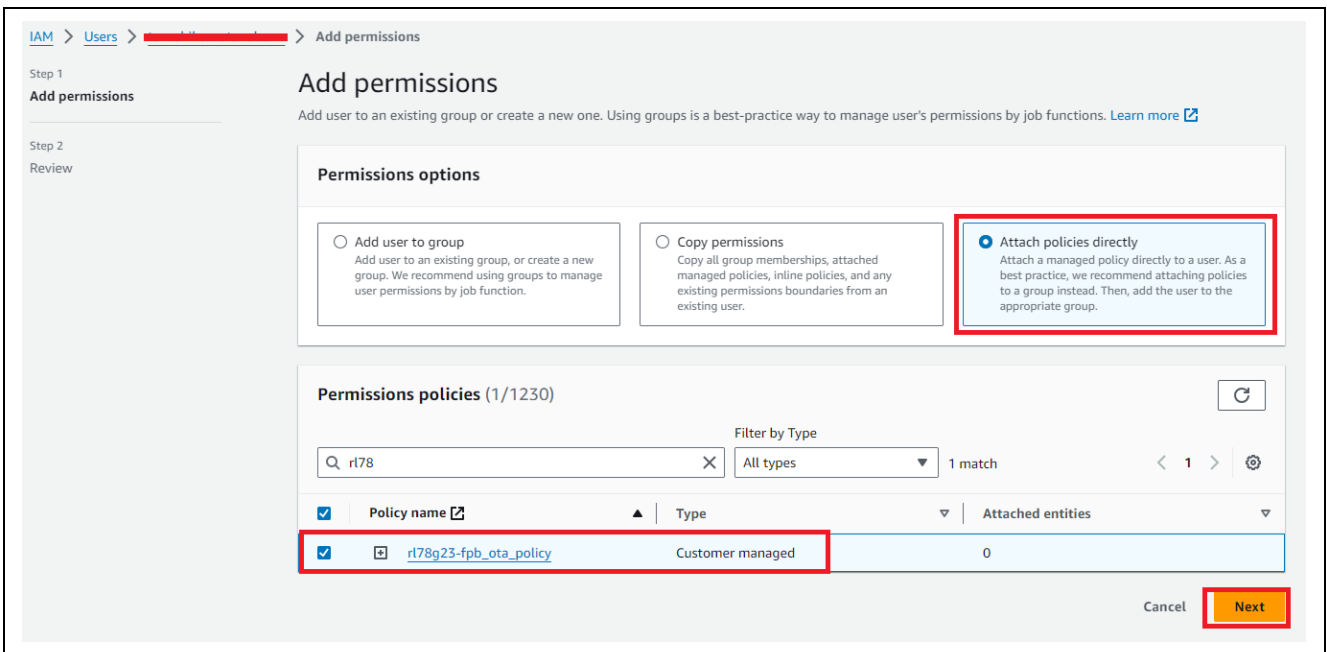


Figure 6-29 Selecting the OTA Update Policy for Permissions to Be Added to the IAM User

- User details: Your account
- Permissions summary > Name: Name of created OTA Update policy (Example: r178g23-fpb_ota_policy)
- Click “Add permission”

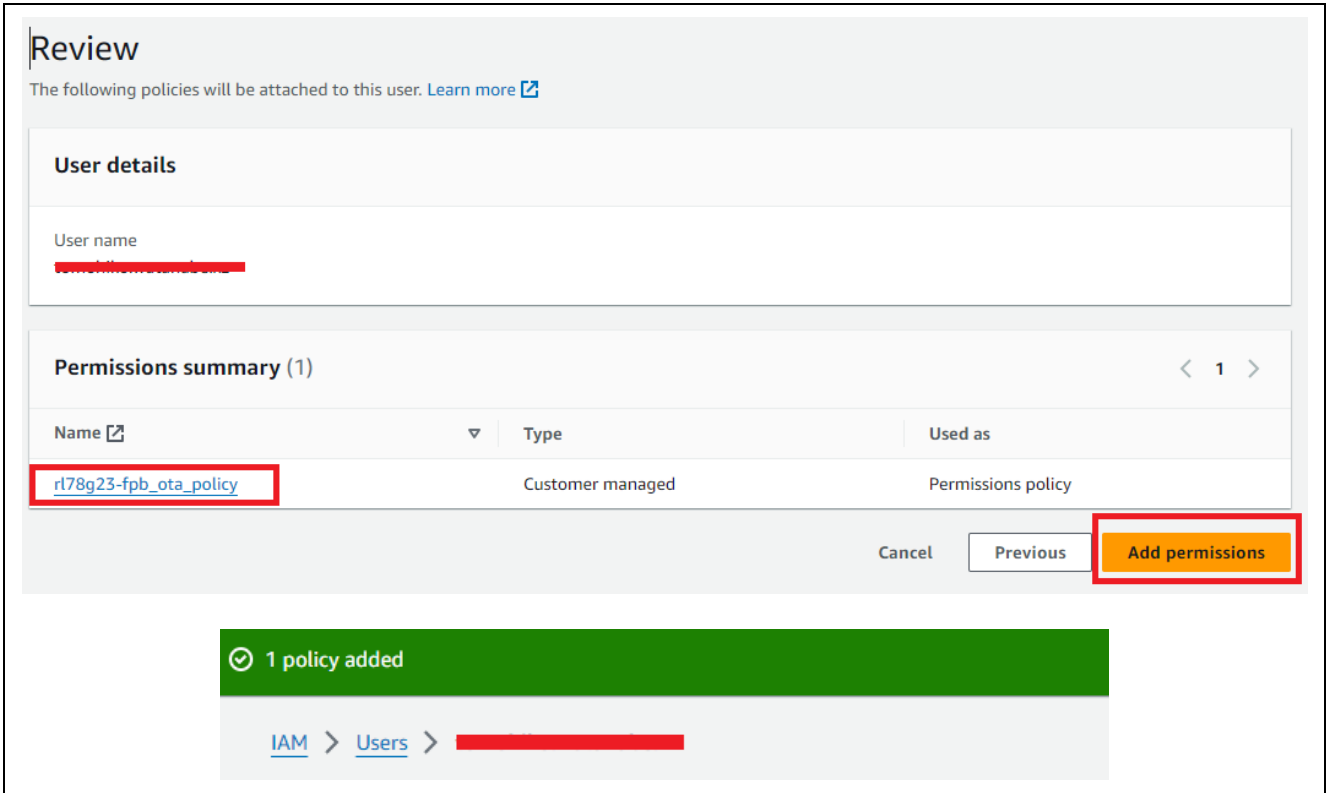


Figure 6-30 Adding the OTA Update Policy to the Selected IAM User

6.1.3.5 Granting Access Permissions to AWS IoT Code Signature

(1) Create an IAM policy.

- IAM > Policies > "Create policy" > "JSON"

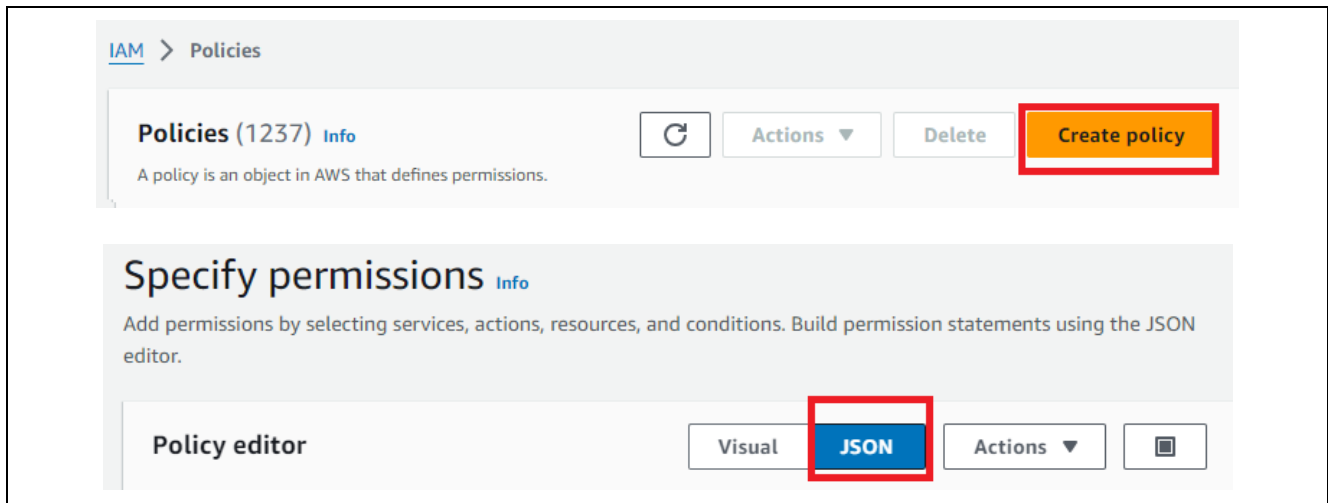


Figure 6-31 Creating an IAM Policy

- Paste the following information to the Policy editor, and then click "Next".

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "iam:CreatePolicy",
        "iam:DetachRolePolicy",
        "iam>DeleteRolePolicy",
        "iam>DeletePolicy",
        "iam:CreateRole",
        "iam>DeleteRole",
        "iam:AttachRolePolicy"
      ],
      "Resource": [
        "arn:aws:iam::*:policy/idt*",
        "arn:aws:iam::*:role/idt*"
      ]
    }
  ]
}
```

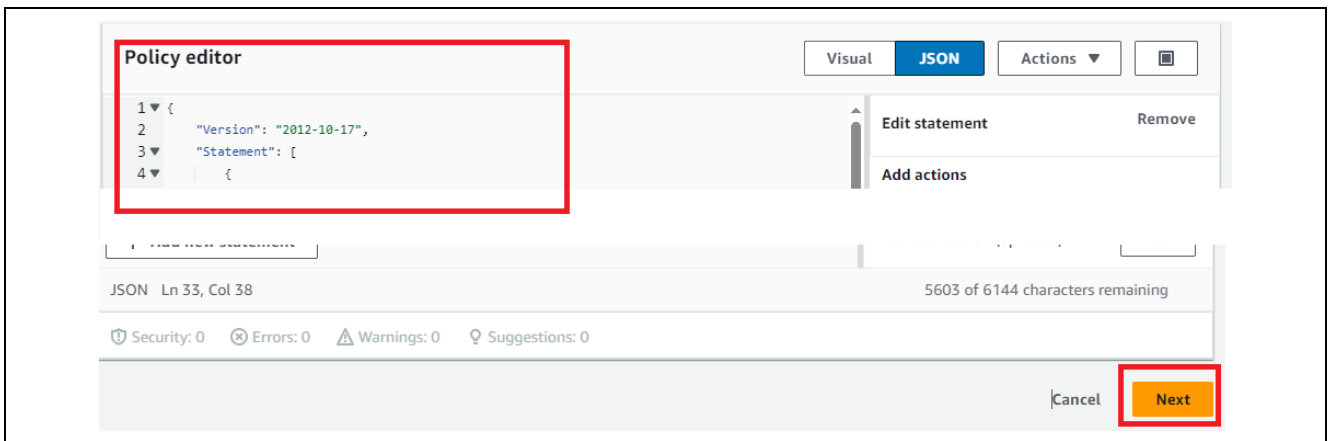


Figure 6-32 Creating an IAM Policy with the Policy Editor

- Policy name: Any (Example: IDTFreeRTOSIAMPermissions_rl78g23-fpb) > “Create policy”

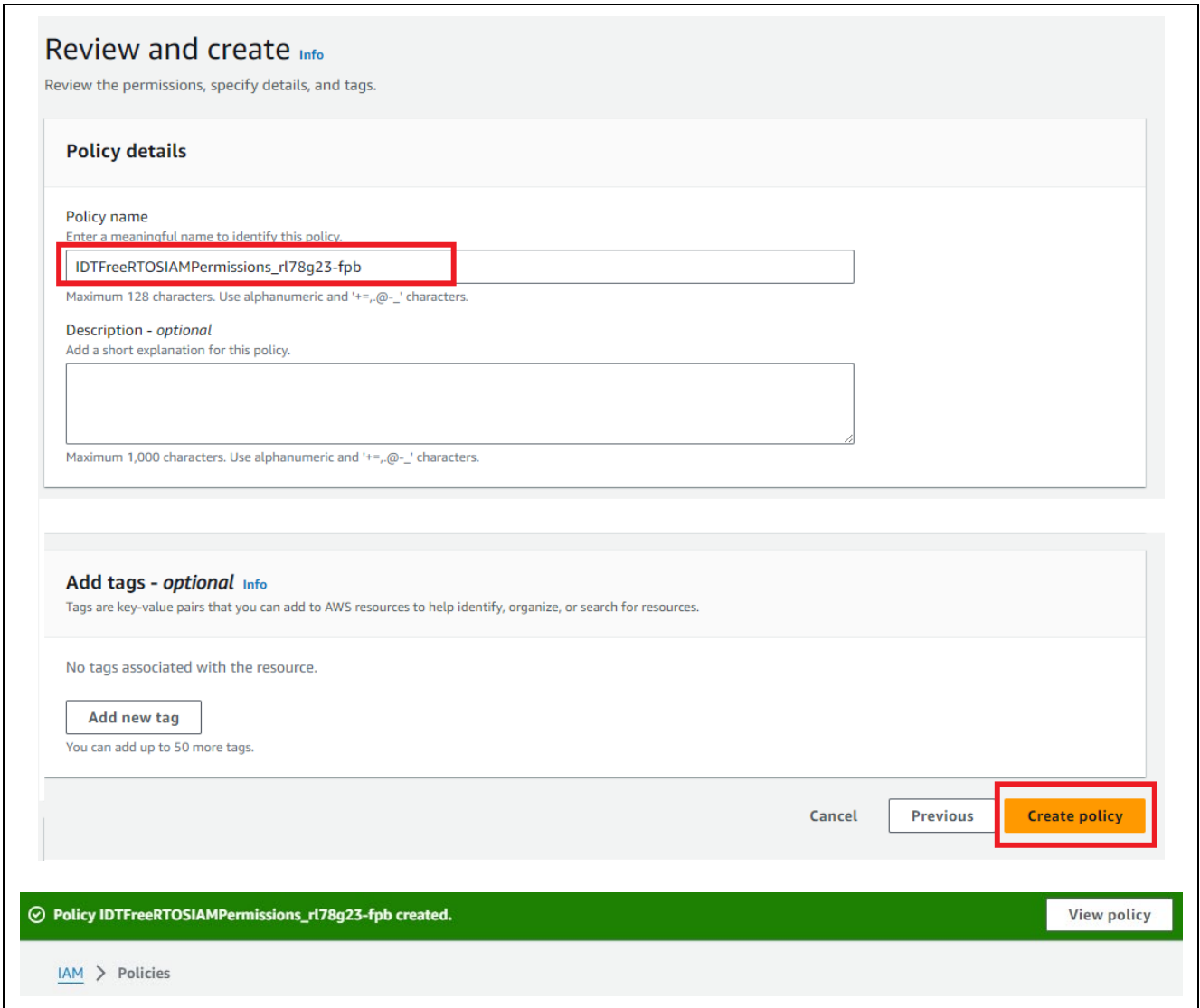


Figure 6-33 Saving the IAM Policy with a Name (Example: IDTFreeRTOSIAMPermissions_rl78g23-fpb)

(2) Attach the created IAM policy to the IAM user.

- IAM > Users > Choose User > Add permissions

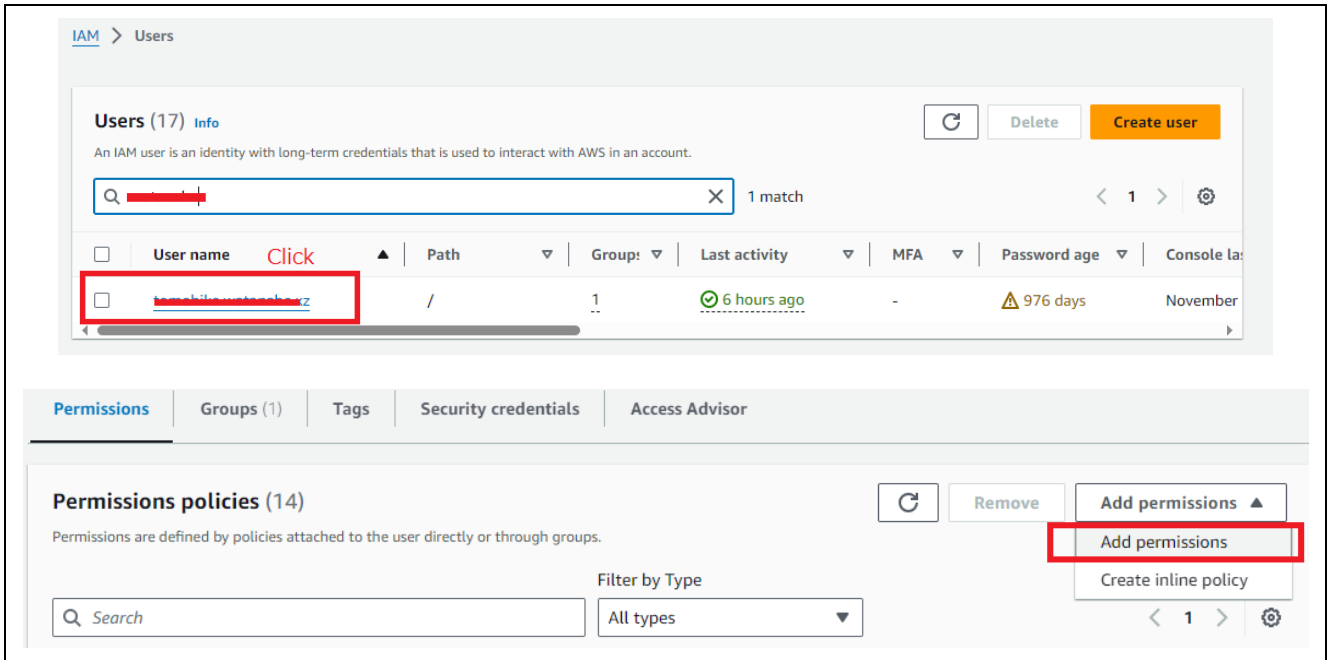


Figure 6-34 Selecting the User to Assign the Created IAM Policy

- Permissions options: Attach policies directly
- Policy name:
 - AWSIoTDeviceTesterForFreeRTOSFullAccess
 - Name of created IAM policy (Example: IDTFreeRTOSIAMPermissions_rl78g23-fpb)
- Click “Next” > Click “Add permissions”

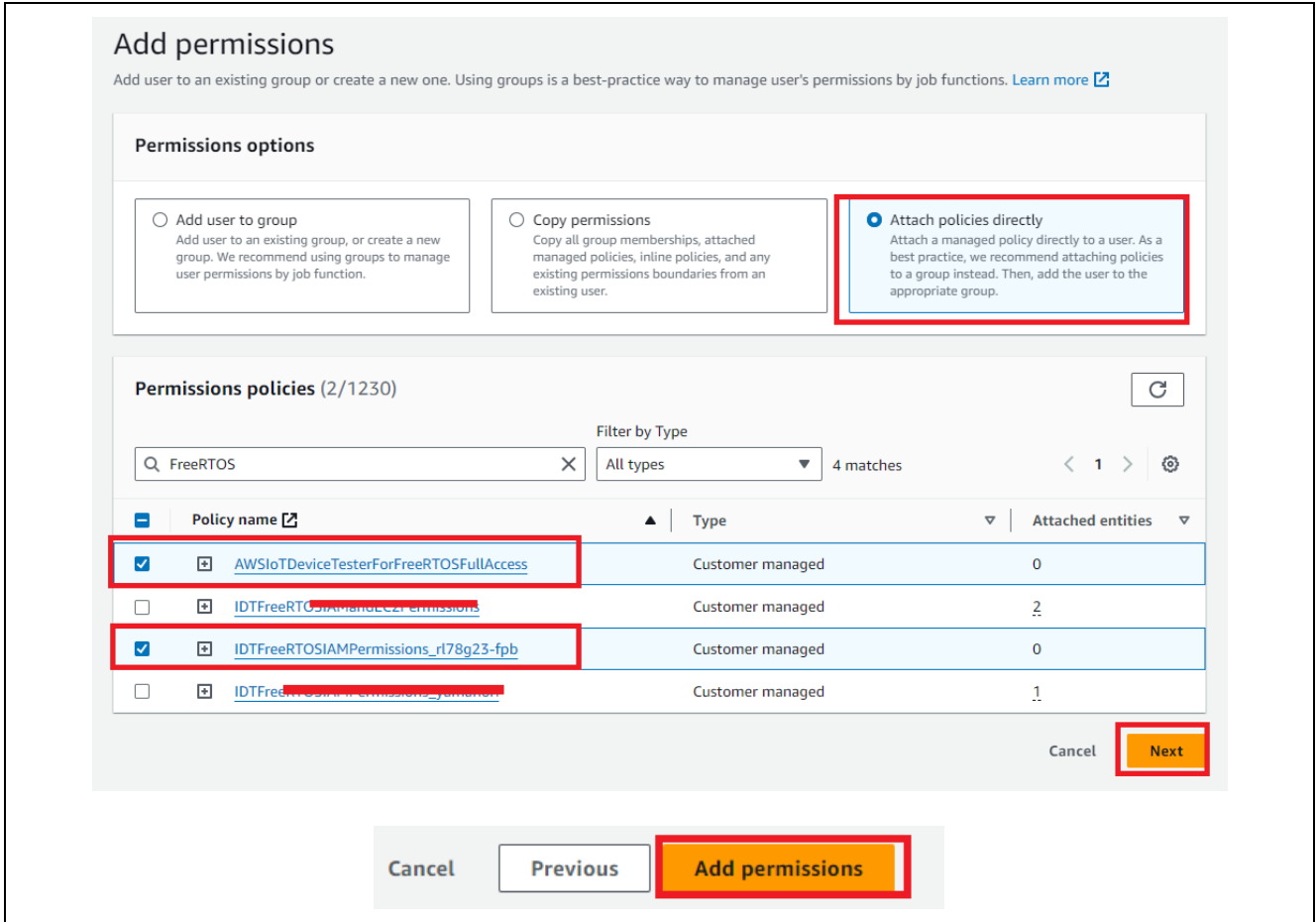


Figure 6-35 Adding Permissions to the Selected IAM User

6.2 Creating an Initial Image

An initial image is a MOT file generated by joining a bootloader’s MOT file and an initial application’s MOT file by using Renesas Image Generator.

Renesas Image Generator is a tool provided with the [RL78/G22,RL78/G23,RL78/G24 firmware update module](#). For details, refer to the application note in this link.

The file names related to an initial image are as follows in this document.

- Bootloader: boot_loader.mot
- Initial application: aws_ryz024a_rl78g23-fpb_ota.mot
- Initial image: initial_image.mot

6.2.1 Creating a Bootloader

6.2.1.1 Importing the Bootloader Project

Import the boot_loader project to e² studio. Open the Import wizard according to the following process.

File > Import... > Existing Projects into Workspace > Next

Next, select the boot_loader project. **Ensure that copy projects into workspace is not selected.** Then click the Finish button.

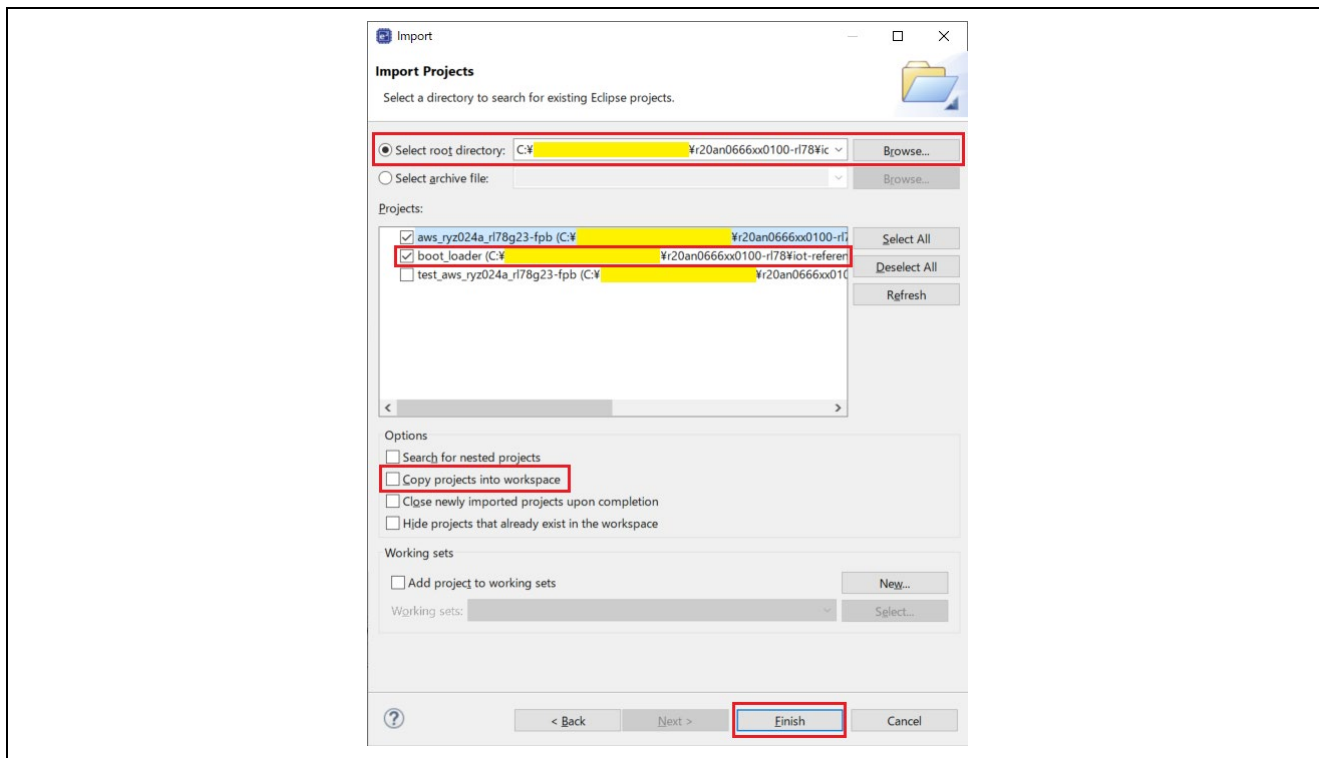


Figure 6-36 Selecting the boot_loader Project

The imported project is shown in the Project Explorer view.

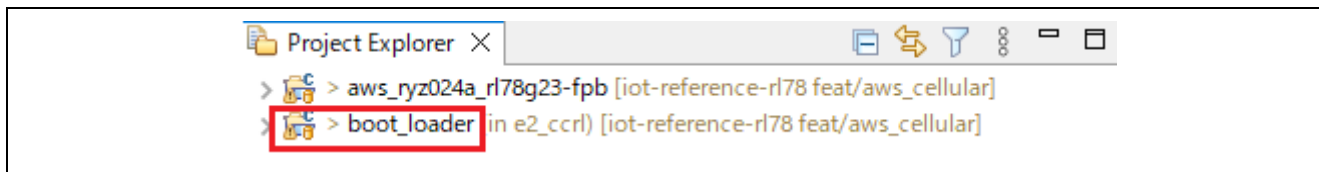


Figure 6-37 Completing to Import the boot_loader Project

6.2.1.2 Adding the Firmware Verification Key to the Bootloader Project

(1) Add the firmware verification key (secp256r1.publickey) to code_signer_public_key.h in the boot_loader project.

Note: Add \ to the end of each line.

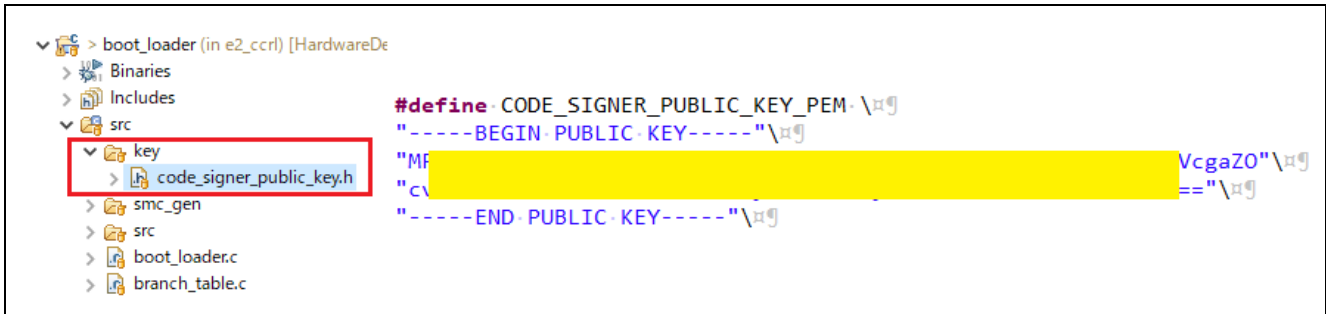


Figure 6-38 Adding the Firmware Verification Key to the Bootloader

6.2.1.3 Building the Bootloader Project

Build the boot_loader project to create a MOT file.

Then, make sure that boot_loader.mot has been created in the HardwareDebug folder directly under the project folder.

6.2.2 Creating an Initial Application

6.2.2.1 Importing the Initial Application

Import the aws_ryz024a_rl78g23-fpb project to e² studio. Open the Import wizard according to the following process.

File > Import... > Existing Projects into Workspace > Next

Next, select the aws_ryz024a_rl78g23-fpb project. **Ensure that copy projects into workspace is not selected.** Then click the Finish button.

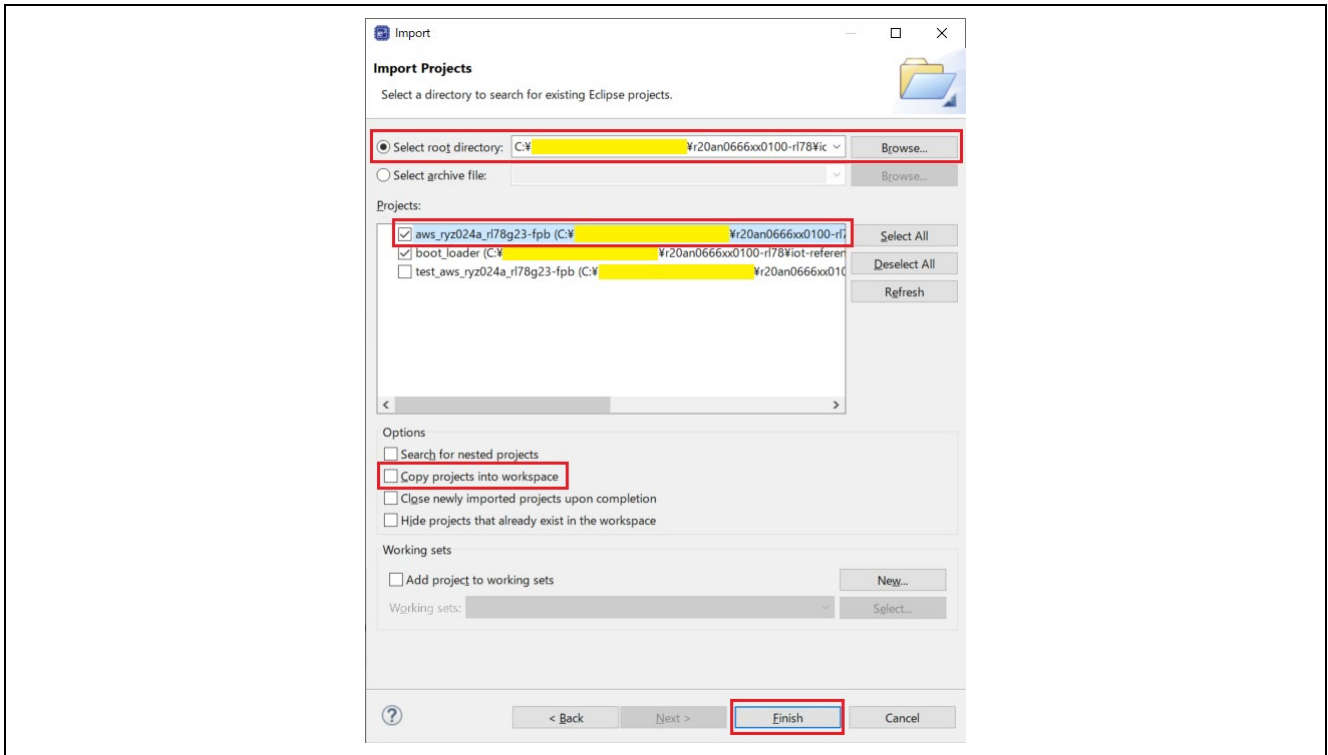


Figure 6-39 Selecting the aws_ryz024a_rl78g23-fpb Project

The imported project is showed in the Project Explorer view.



Figure 6-40 Completing to Import the aws_ryz024a_rl78g23-fpb Project

6.2.2.2 Setting the Build Configuration of the the Initial Application

Set the build configuration of the aws_ryz024a_rl78g23-fpb project to “HardwareDebug_OTA”.

Build Configurations > Set Active > Select “HardwareDebug_OTA”

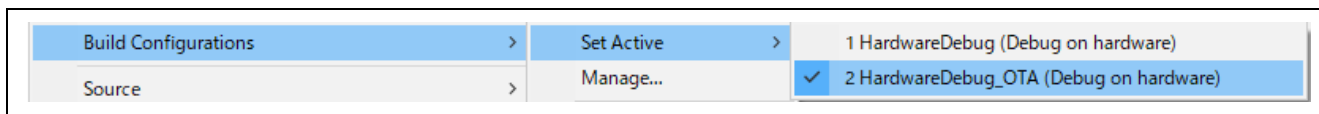


Figure 6-41 Activating Build Configuration “HardwareDebug_OTA”

6.2.2.3 Adding the Firmware Verification Key to the Initial Application

Add the firmware verification key (secp256r1.publickey) to code_signer_public_key.h in the aws_ryz024a_rl78g23-fpb project.

Note: Add \ to the end of each line.

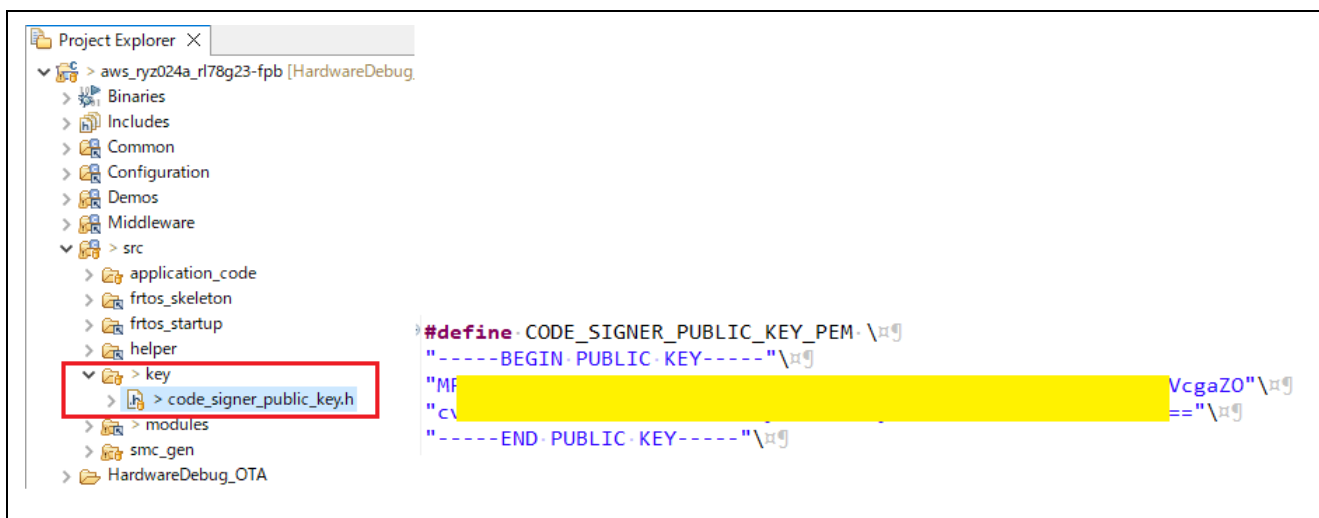


Figure 6-42 Adding the Firmware Verification Key to the Initial Application

6.2.2.4 Building the Initial Application

Build the aws_ryz024a_rl78g23-fpb project to create a MOT file.

Then, make sure that aws_ryz024a_rl78g23-fpb_ota.mot has been created in the HardwareDebug_OTA folder directly under the project folder.

6.2.3 Creating an Initial Image by Using Renesas Image Generator

Join the bootloader and the initial application by using Renesas Image Generator to generate the initial image.

(1) Store the following files in the same folder as Renesas Image Generator.

- Bootloader: boot_loader.mot
- Initial application: aws_ryz024a_rl78g23-fpb_ota.mot
- Private key for initial application verification: secp256r1.privatekey

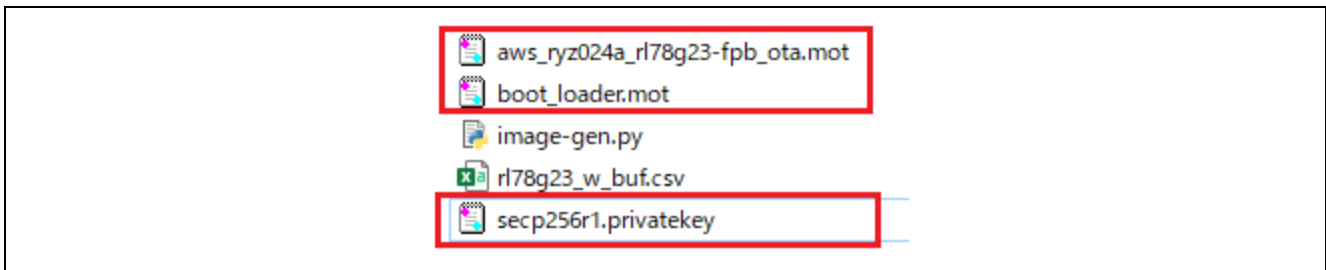


Figure 6-43 Storing Necessary Files in the Same Folder as Renesas Image Generator

(2) Run the following command to generate the initial image.

```
python image-gen.py -iup .\aws_ryz024a_rl78g23-fpb_ota.mot -ibp  
boot_loader.mot -o initial_image -ip .\RL78_G23_ImageGenerator_PRM.csv
```

(3) Make sure that the initial image (initial_image.mot) has been generated.

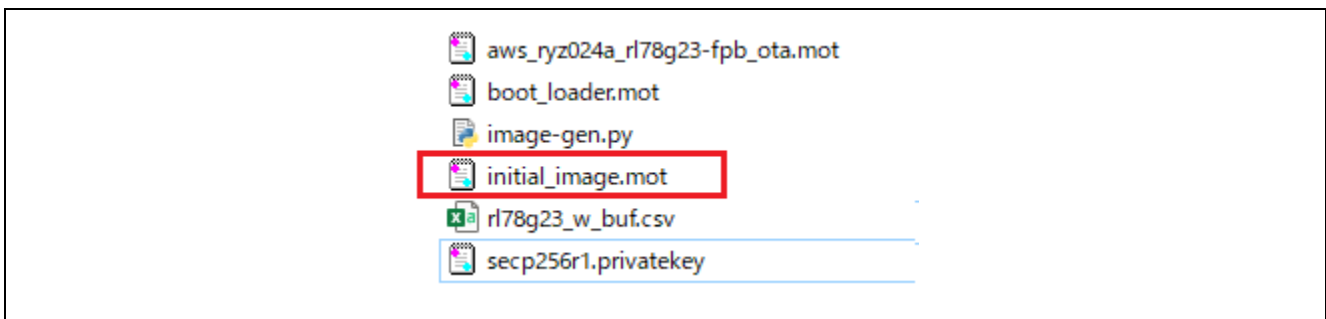


Figure 6-44 Initial Image Generated in the Same Folder as Renesas Image Generator

6.3 Creating an Update Image

An update image is a binary format (extension: rsu) firmware used for update which are converted an update application's MOT file by using Renesas Image Generator. Update images can be generated by Renesas Image Generator. For details about the update image format, refer to "[RL78/G22,RL78/G23,RL78/G24 Firmware Update Module](#)".

The file names related to an update image are as follows in this document.

- Update application: aws_ryz024a_rl78g23-fpb_ota_093.mot
- Update image: aws_ryz024a_rl78g23-fpb_ota_093.rsu

6.3.1 Creating an Update Application

6.3.1.1 Changing the Source Code of the Application

To create an update application, in `iot-reference-rl78\Configuration\rl78g23-fpb\ota\cellular\ertos_config\demo_config.h`, change the definition of the `APP_VERSION_BUILD` macro from 2 to 3.

```
iot-reference-rl78\Configuration\rl78g23-
fpb\ota\cellular\ertos_config\demo_config.h

/**
 * @brief Build version of the firmware.
 *
 * This is used in the OTA demo to set the appFirmwareVersion variable that
 is
 * declared in the ota_appversion32.h file in the OTA library.
 */
#ifndef APP_VERSION_BUILD
    #define APP_VERSION_BUILD 3
#endif
```

6.3.1.2 Building the Update Application

Build the `aws_ryz024a_rl78g23-fpb` project to create a MOT file. Then, make sure that `aws_ryz024a_rl78g23-fpb_ota.mot` has been overwritten and created in the `HardwareDebug_OTA` folder directly under the project folder.

6.3.1.3 Renaming the MOT File of the Update Application

Rename `aws_ryz024a_rl78g23-fpb_ota.mot` to `aws_ryz024a_rl78g23-fpb_ota_093.mot`.

6.3.2 Generating an Update Image by Using Renesas Image Generator

Convert the update application to an update image by using Renesas Image Generator.

(1) Store the following files in the same folder as Renesas Image Generator.

- MOT file of the update application: aws_ryz024a_rl78g23-fpb_ota_093.mot
- Private key for update application verification: secp256r1.privatekey

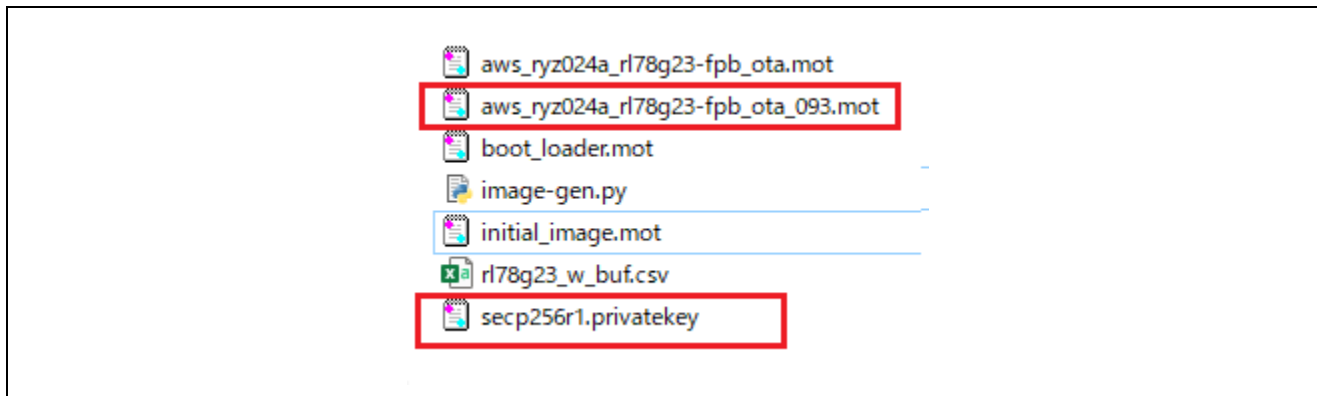


Figure 6-45 Storing Necessary Files in the Same Folder as Renesas Image Generator

(2) Run the following command to generate an update image (aws_ryz024a_rl78g23-fpb_ota_093.rsu) in RSU format.

```
python image-gen.py -iup .\aws_ryz024a_rl78g23-fpb_ota_093.mot -o  
aws_ryz024a_rl78g23-fpb_ota_093 -ip .\RL78_G23_ImageGenerator_PRM.csv -vt  
ecdsa -ff RTOS
```

(3) Make sure that aws_ryz024a_rl78g23-fpb_ota_093.rsu has been generated.

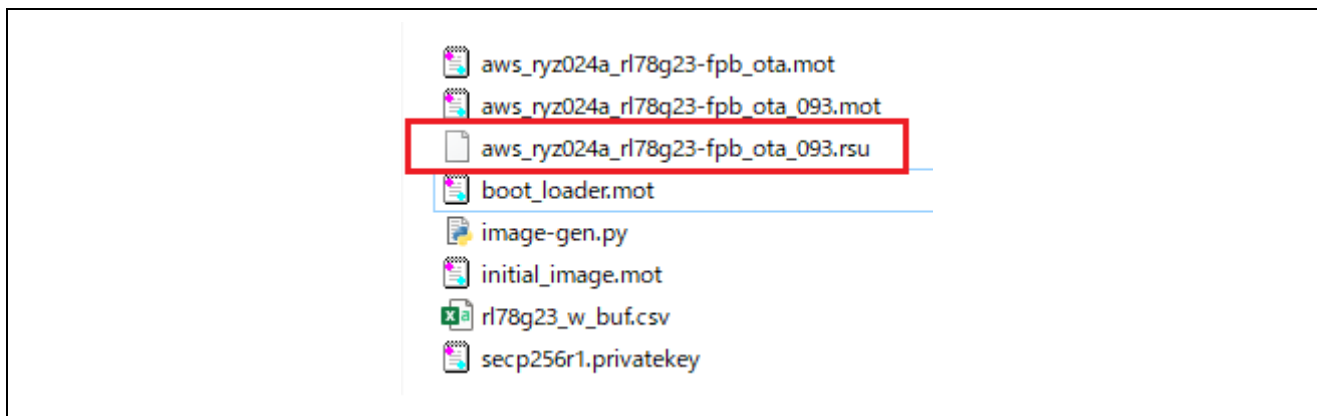


Figure 6-46 Update Image Generated in the Same Folder as Renesas Image Generator

6.4 Running the Demo Project

The following describes the running procedure for the demo project (OTA).

6.4.1 Programming the Initial image (initial_image.mot) to Board

(1) Program the initial image (initial_image.mot).

For the programming method, refer to Chapter 7, Using Renesas Flash Programmer.

(2) When programming terminates, the demo project starts.

(3) Check the terminal to make sure that the initial application (version 0.9.2) has started.

```

==== RL78G23 : BootLoader [with buffer] ====
verify install area 0 [sig-sha256-ecdsa]...OK
execute new image ...
Hello World.
0 15763 [MAIN_TASK] Connecting Access Point is OK.

1 15763 [MAIN_TASK] -----STARTING DEMO-----

2 15765 [MQTT] [INFO] -----Start MQTT Agent Task-----

3 15766 [MQTT] [INFO] Creating a TLS connection to a31klnx40j1phd-ats.iot.ap-northeast-1.amazonaws.com:8883.
4 15779 [MQTT] [INFO] Created new TCP socket.
5 21505 [MQTT] [INFO] Established TCP connection with a31klnx40j1phd-ats.iot.ap-northeast-1.amazonaws.com.
6 21505 [MQTT] [INFO] Creating an MQTT connection to the broker.
7 21834 [MQTT] [INFO] MQTT connection established with the broker.
8 21835 [MQTT] [INFO] Successfully connected to MQTT broker.
9 21836 [OTA Demo Ta] [INFO] -----Start OTA Task-----

10 21836 [OTA Demo Ta] [INFO] OTA over MQTT demo, Application version 0.9.2
11 21845 [OTA Agent T] [INFO] ota_ota_get_image_state is called. ota_ota_image_state_t = 2
12 21845 [OTA Agent T] [INFO] Current State=[RequestingJob], Event=[Start], New state=[RequestingJob]
13 21856 [OTA Demo Ta] [INFO] Received: 0 Queued: 0 Processed: 0 Dropped: 0
14 22875 [OTA Agent T] [INFO] Subscribed to topic $aws/things/rx-ota-firm-things-rx85n-rsk/jobs/notify-next.

```

Figure 6-47 Initial Application (Version 0.9.2) Started

6.4.2 Registering the Update Image (aws_ryz024a_r178g23-fpb_ota_093.rsu) with OTA Jobs

(1) AWS IoT > Manage > Remote actions > Jobs > Click “Create job”

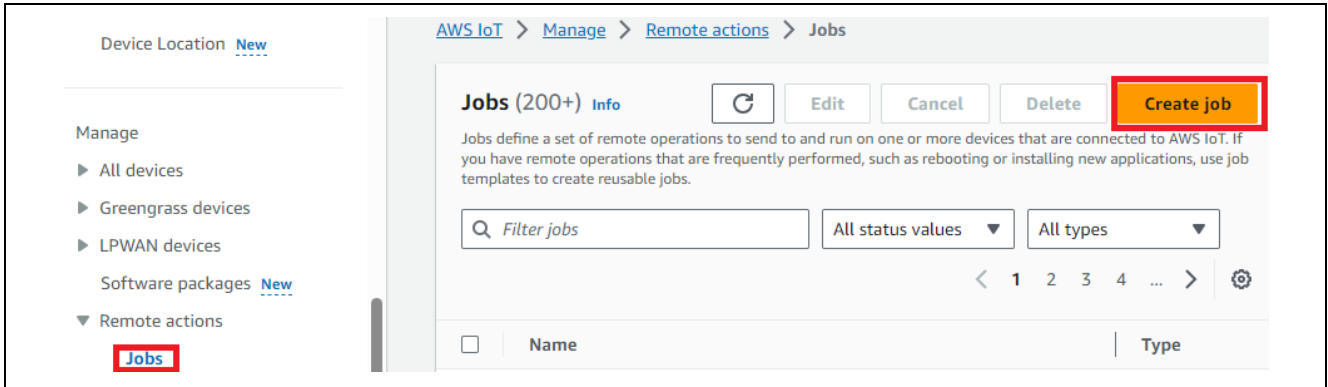


Figure 6-48 Jobs

(2) Check “Create FreeRTOS OTA update job” > Click “Next”

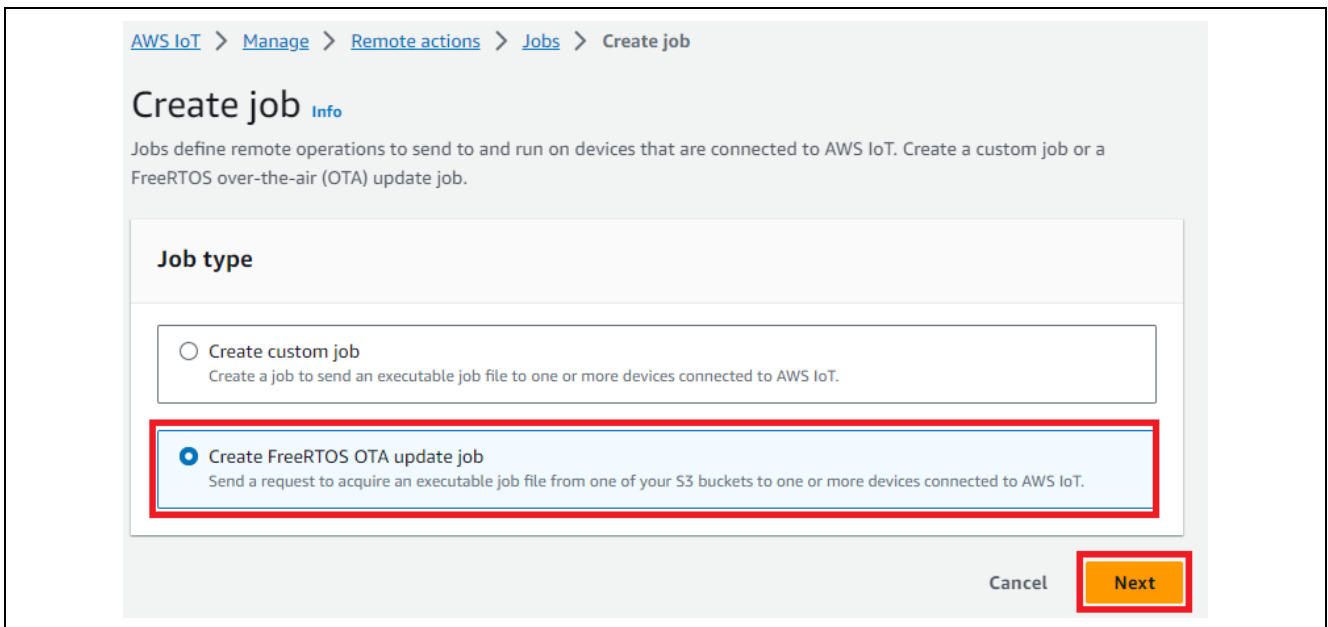


Figure 6-49 Create Job

(3) Step 1: OTA job properties

- Job name: Any

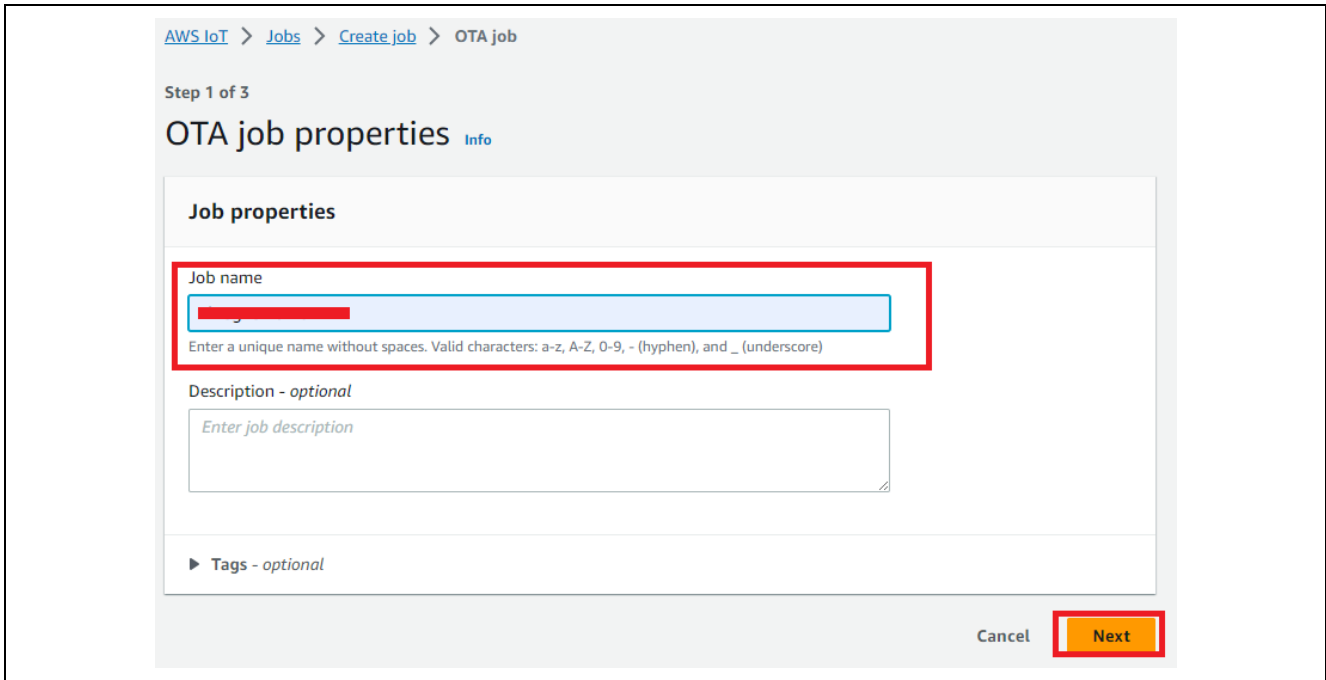


Figure 6-50 Step 1: OTA job properties

(4) Step 2: OTA file configuration > Devices

- Devices to update: "Name of the thing" in aws_clientcredential.h

```
#define clientcredentialIOT_THING_NAME "YOUR_THING_NAME"
```

- Select the protocol for file transfer: MQTT

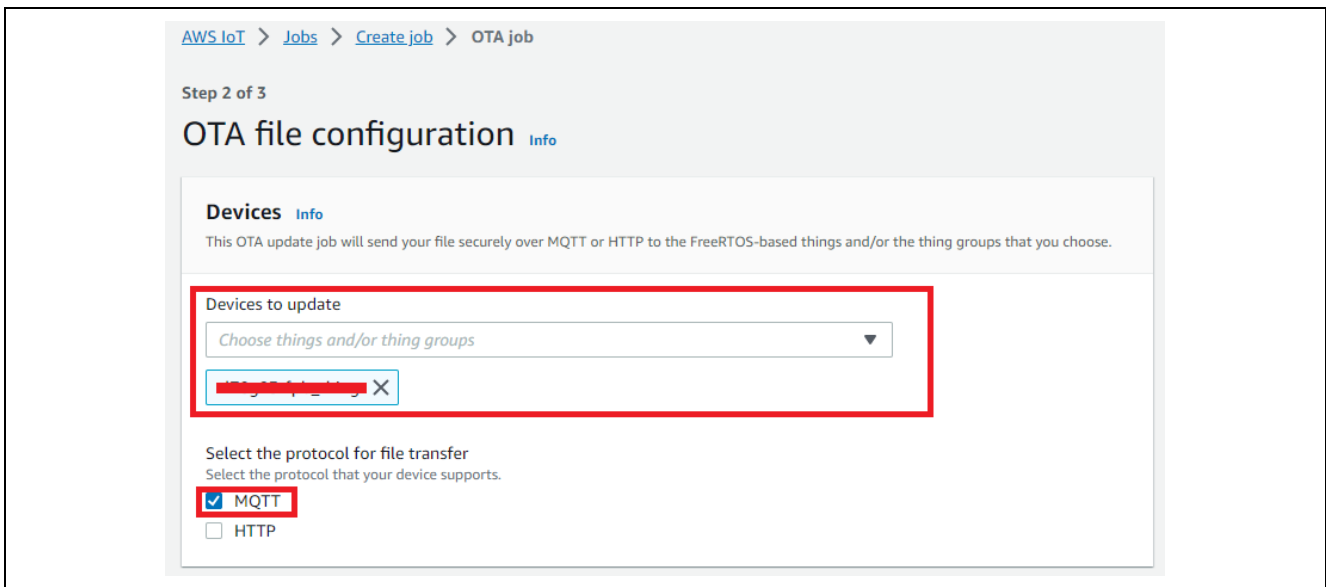


Figure 6-51 Step 2: OTA file configuration > Devices

(5) Step 2: OTA file configurations > File

- Sign and choose your file: Sign a new file for me.

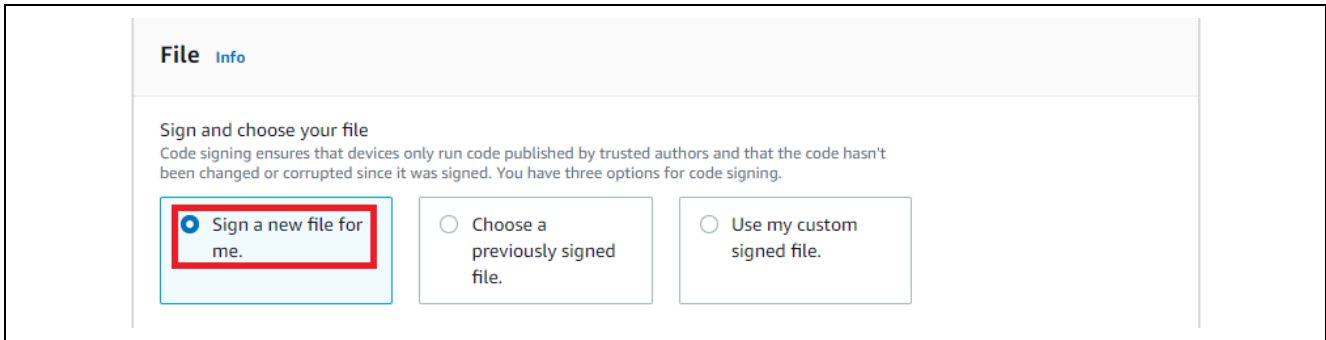


Figure 6-52 Step 2: OTA file configurations > File (1)

- Code signing profile: Click “Create new profile”

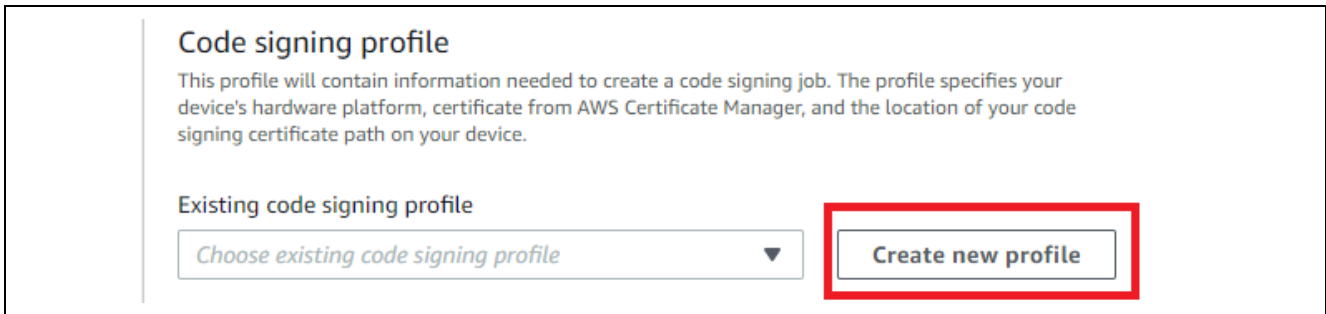


Figure 6-53 Step 2: OTA file configurations > File (2)

- Create a code signing profile.
 - Profile name: Any (Example: rl78g23_fpb_ota_cert)
 - Device hardware platform: Windows Simulator
 - Code signing certificate: "Import new code signing certificate"
 - Certificate body: secp256r1.crt
 - Certificate private key: secp256r1.privatekey
 - Certificate chain – optional: ca.crt
 - Path name of code signing certificate on device: Any

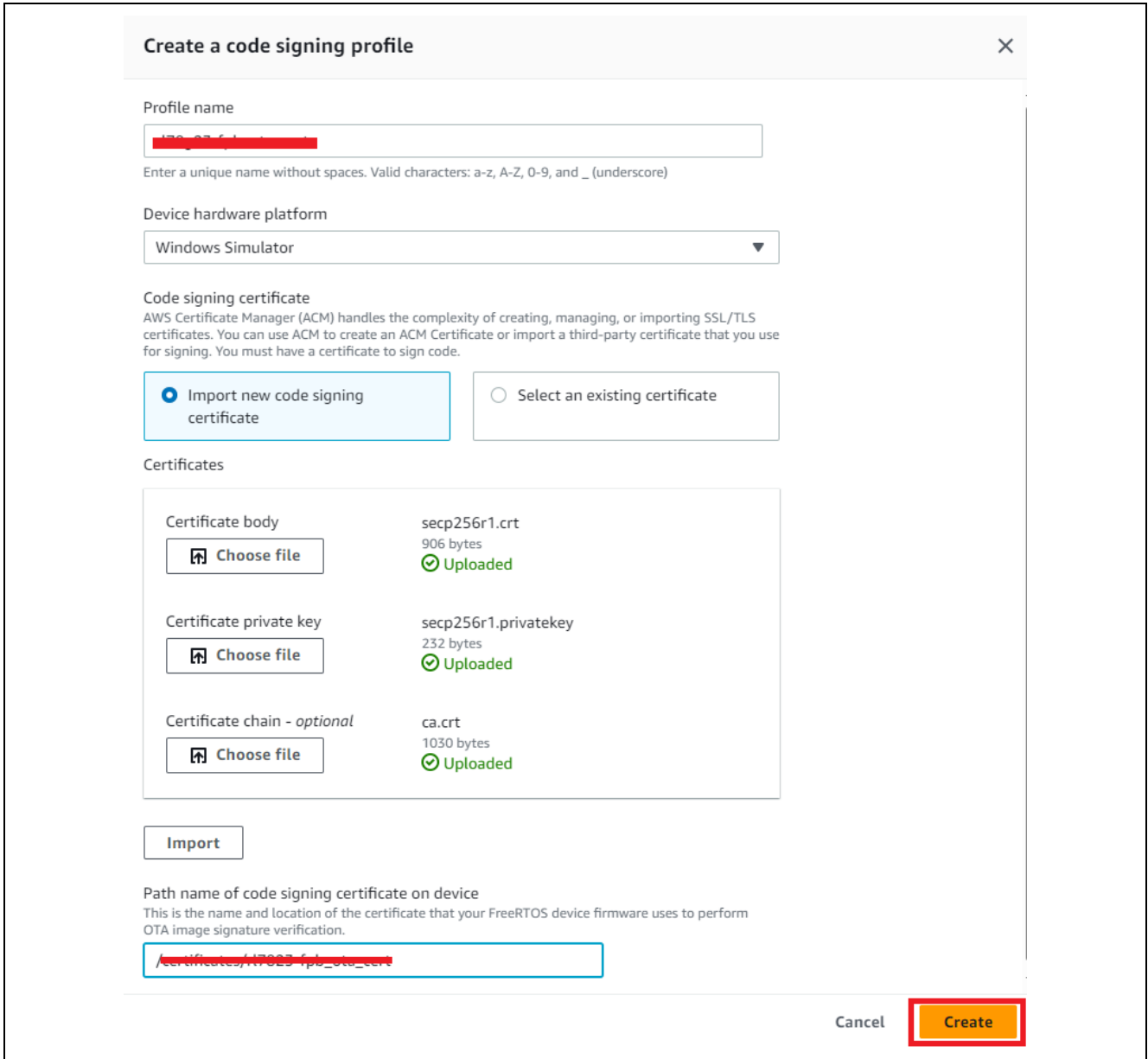


Figure 6-54 Create a code signing profile

- File > “Upload a new file.” > “Choose file” > aws_ryz024a_rl78g23-fpb_ota_093.rsu

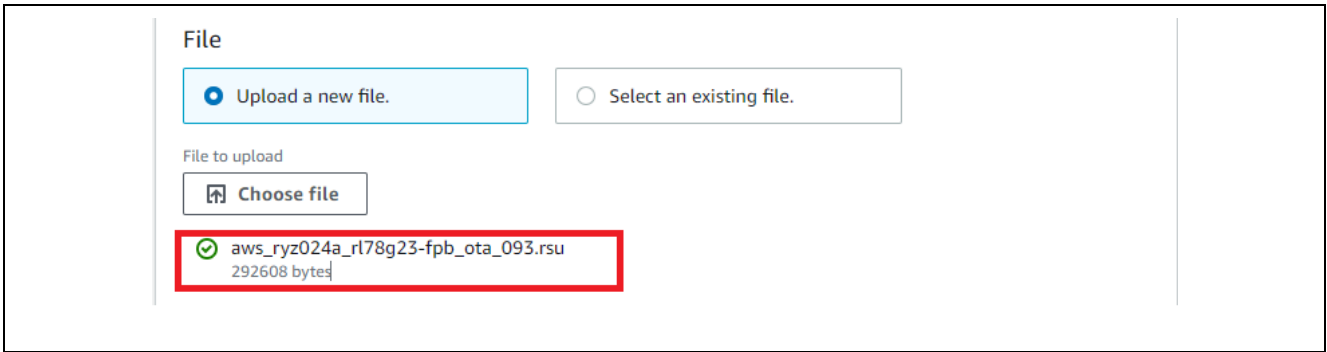


Figure 6-55 Upload a new file > aws_ryz024a_rl78g23-fpb_ota_093.rsu

- File upload location in S3: Specify the created bucket (Bucket name specified in section 6.1.3.1, Creating Amazon S3 Buckets).
- Path name of file on device: Any

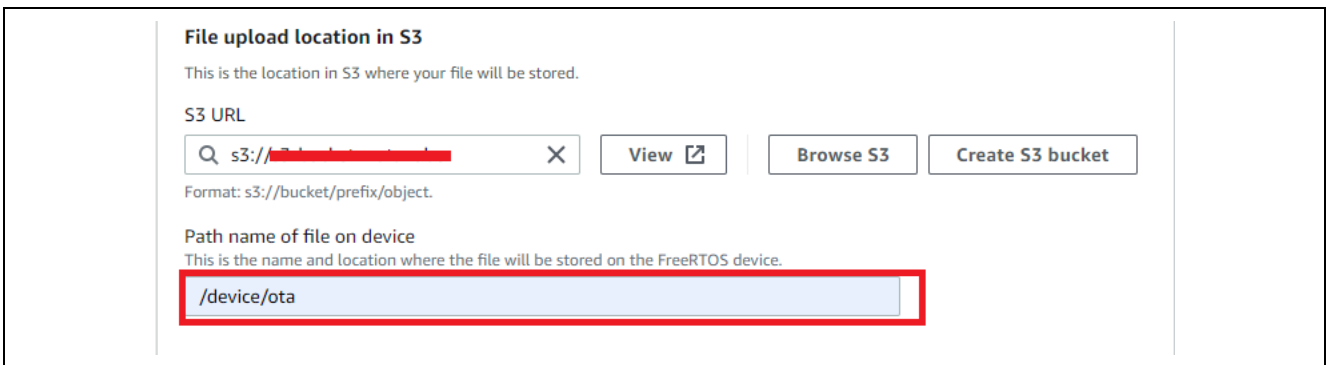


Figure 6-56 File upload location in S3

(6) Step 2: OTA file configurations > IAM role

- Role: Specify the created role (Role name specified in section 6.1.3.2, Creating an OTA Update Service Role).

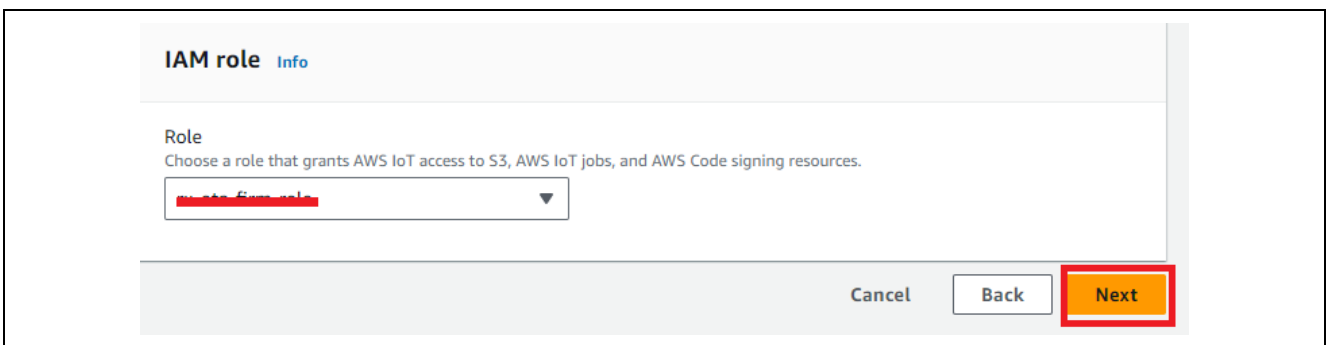


Figure 6-57 Step 2: OTA file configurations > IAM role

(7) Step 3: OTA job configuration

- Job run type: Your job will complete after deploying to the devices and groups that you chose (snapshot)

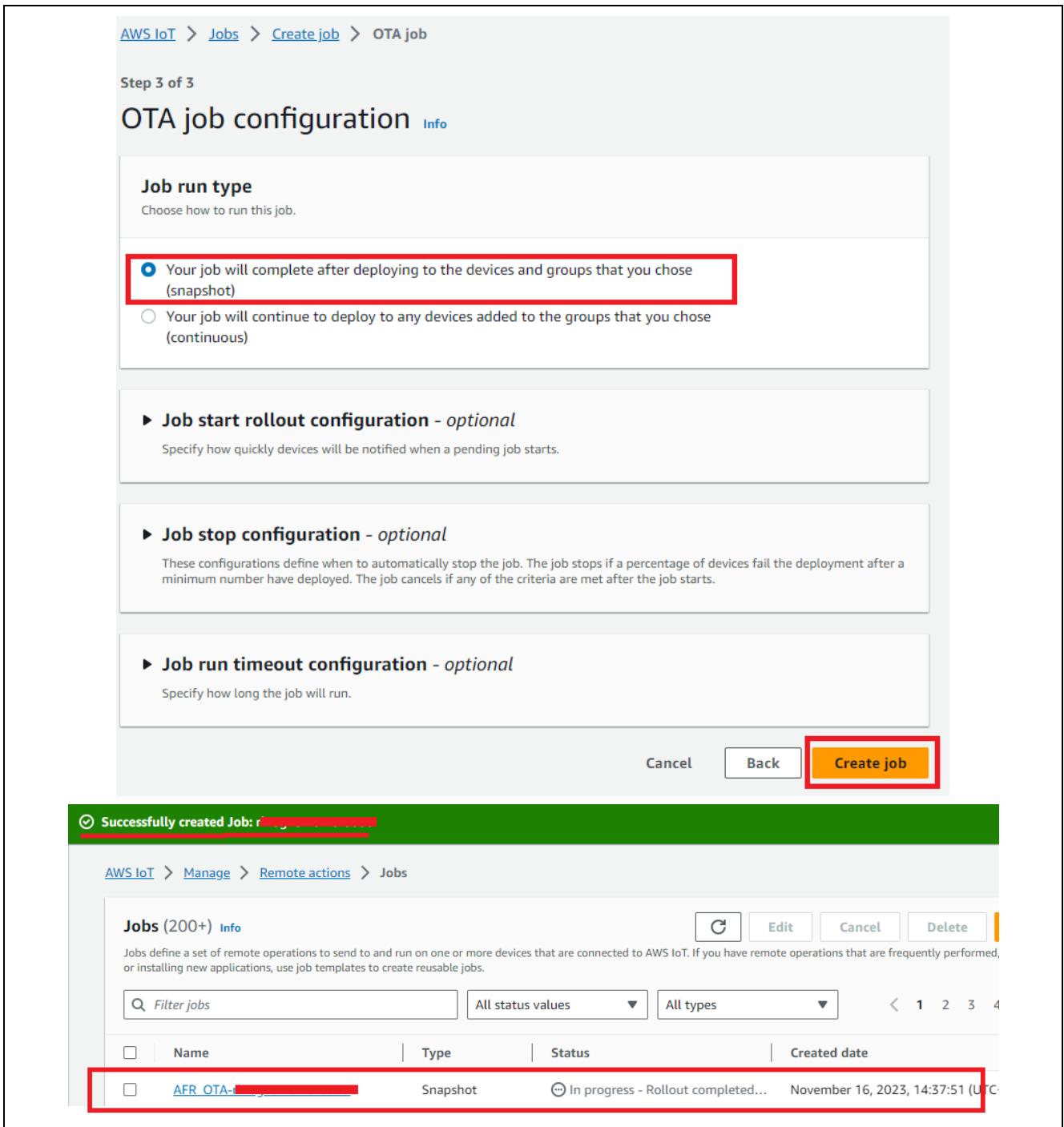


Figure 6-58 Step 3: OTA job configuration

(8) After a while, the log of programming the update image to the MCU board is output to the terminal.

```

92c2-7c982e01f5f8]
3 12176 [OTA Agent T] [INFO] Extracted parameter: [key: value]=[execution.jobDocument.afs_ota.protocols: ["MQTT"]]
4 12187 [OTA Agent T] [INFO] Extracted parameter: [key: value]=[filepath: /device/ota]
5 12190 [OTA Agent T] [INFO] Extracted parameter: [key: value]=[filesize: 193920]
6 12194 [OTA Agent T] [INFO] Extracted parameter: [key: value]=[fileid: 0]
7 12197 [OTA Agent T] [INFO] Extracted parameter: [key: value]=[certfile: watanabe_ota_cert]
8 12209 [OTA Agent T] [INFO] Extracted parameter [ sig-sha256-ecdsa: MEUCIB4ZwqfsBwThCIYzJ1VLx8wzqNR6... ]
9 12216 [OTA Agent T] [INFO] Job document was accepted. Attempting to begin the update.
20 12217 [OTA Agent T] [INFO] Job parsing success: OtaJobParseErr_t=OtaJobParseErrNone, Job name=AFR_OTA-LTS2_rl78g23_tomo_064
21 12218 [OTA Agent T] [INFO] Setting OTA data interface.
22 12219 [OTA Agent T] [INFO] Current State=[CreatingFile], Event=[ReceivedJobDocument], New state=[CreatingFile]
23 12934 [OTA Agent T] [INFO] Current State=[RequestingFileBlock], Event=[CreateFile], New state=[RequestingFileBlock]
24 12944 [MQTT] [INFO] Publishing message to $aws/things/rx-ota-firm-things-rx85n-rsk/streams/AFR_OTA-5931dc99-6ce0-428e-92c2-7c982e01f5f8/get/cbor.
25 13429 [OTA Agent T] [INFO] Published to MQTT topic to request the next block: topic=$aws/things/rx-ota-firm-things-rx85n-rsk/strea
ns/AFR_OTA-5931dc99-6ce0-428e-92c2-7c982e01f5f8/get/cbor
26 13430 [OTA Agent T] [INFO] Current State=[WaitingForFileBlock], Event=[RequestFileBlock], New state=[WaitingForFileBlock]
27 13688 [MQTT] [INFO] De-serialized incoming PUBLISH packet: DeserializerResult=MQTTSuccess.
28 13689 [MQTT] [INFO] State record updated. New state=MQTTPublishDone.
29 13672 [OTA Agent T] [INFO] Received valid file block: Block index=0, Size=0
30 13676 [OTA Agent T] # 0x59200, 256 ... OK
31 13678 [OTA Agent T] # 0x59300, 768 ... OK

```

Figure 6-59 Programming the Update Image to the MCU Board

(9) When programming terminates, the update image (version 0.9.3) starts.

```

==== RL78G23 : BootLoader [with buffer] ====
verify install area 1 [sig-sha256-ecdsa]...OK
copy to main area ... OK
software reset...

==== RL78G23 : BootLoader [with buffer] ====
verify install area 0 [sig-sha256-ecdsa]...OK
execute new image ...
Hello World.
0 12025 [MAIN_TASK] Connecting Access Point is OK.

1 12026 [OTA Demo Ta] [INFO] OTA over MQTT demo, Application version 0.9.3
2 12027 [OTA Demo Ta] [INFO] Creating a connection to a3ikinx40jipnd-ats.iot.ap-northeast-1
3 12045 [OTA Demo Ta] [INFO] Created new TCP socket.

```

Figure 6-60 Update Image (Version 0.9.3) Started after Programming Terminates

6.5 Debugging the Initial Application

The following describes the procedure for starting the initial application from e² studio and debugging it. Because the bootloader is not used in this procedure, downloaded update images cannot be started.

(1) Change the setting to not use the bootloader.

Change the “USE_BOOTLOADER_V2” macro of the aws_ryz024a_r178g23-fpb project to 0, and then click “Apply and Close”.

- Configuration: HardwareDebug_OTA
- Languages: GNU C
- USE_BOOTLOADER_V2: 0

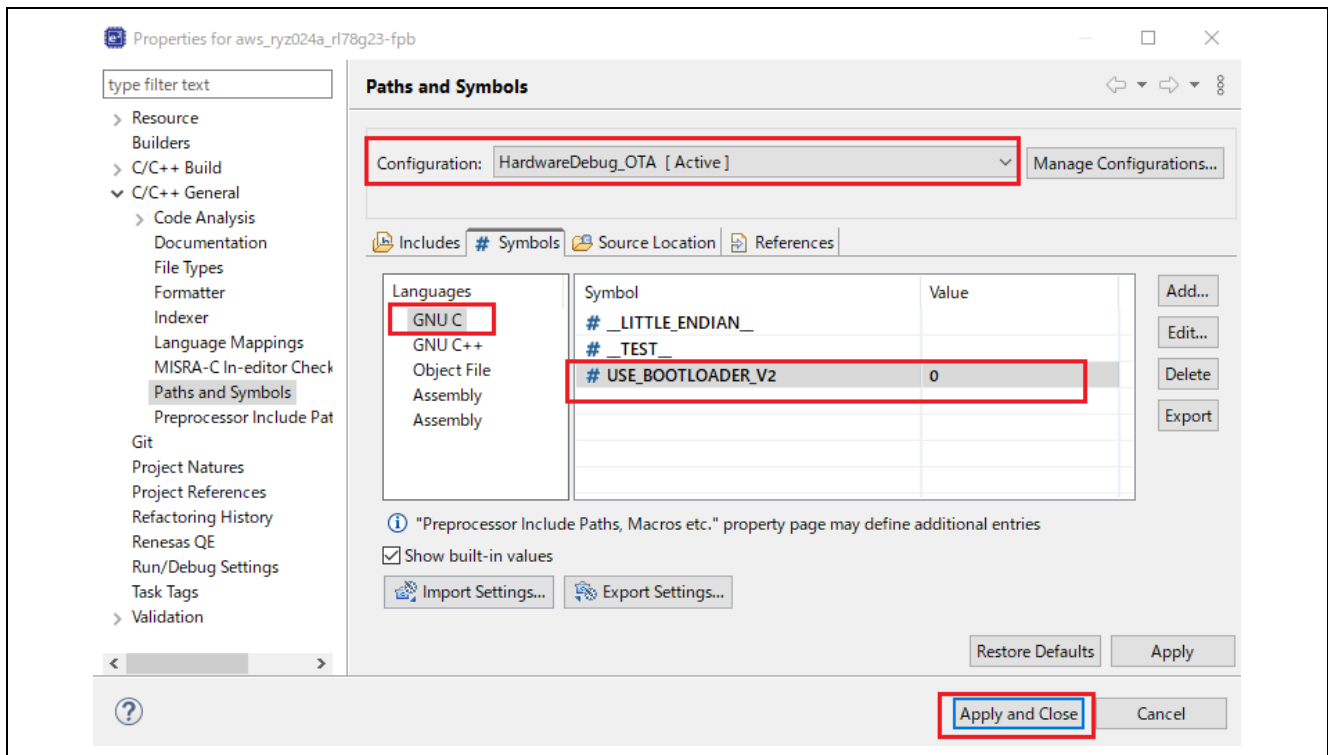


Figure 6-61 Setting the “USE_BOOTLOADER_V2” Macro to 0

(2) Build the aws_ryz024a_r178g23-fpb project.

(3) Start debugging.

Refer to Chapter 8, Debug Procedure.

7. Using Renesas Flash Programmer

The following describes the procedure for using Renesas Flash Programmer to program MOT files to the MCU board.

7.1 When Using COM Port

The following describes how to program a MOT file via the COM port.

7.1.1 Setting Jumper Pins

Set J15: 1-2 Short, J16: 1-2 Short, and J19: 1-2 Short. If you don't change circuit, you don't need this process.

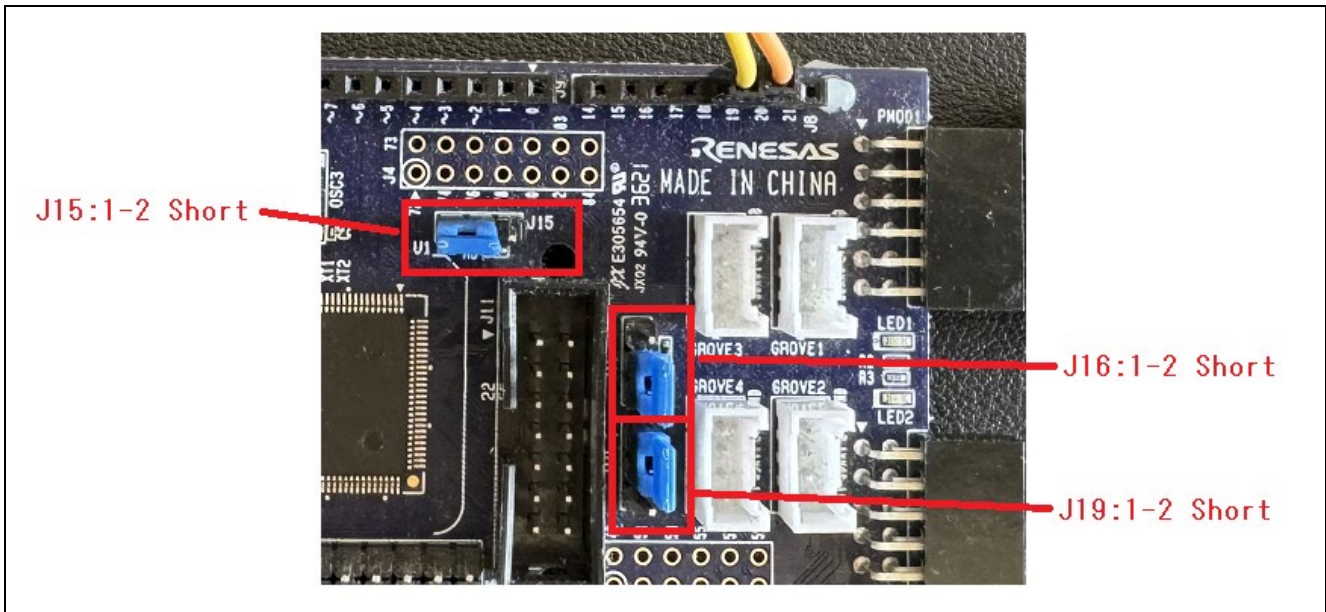


Figure 7-1 Settings for Using COM Port Debugging (Top Side)

7.1.2 Supplying Power to the MCU Board

Connect the USB cable to supply power to the MCU board.

7.1.3 Creating a New Project and Connecting to the MCU Board

(1) File > New project

- Microcontroller: RL78/G2x
- Project Name: Any (Example: rl78g23-fpb)
- Project Folder: Any
- Tool: COM port
- Interface: 2 wire UART
- Tool Details...: COM port number
- Click "Connect"

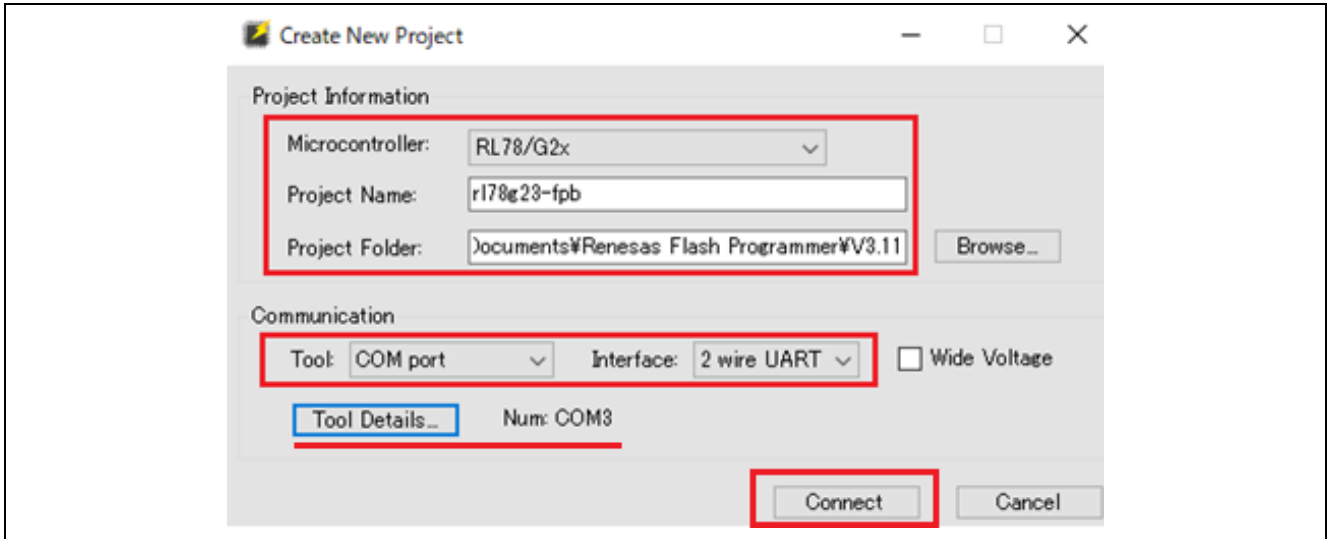


Figure 7-2 Creating a New Project and Connecting to the MCU Board

(2) The connection is successful if the following window appears.

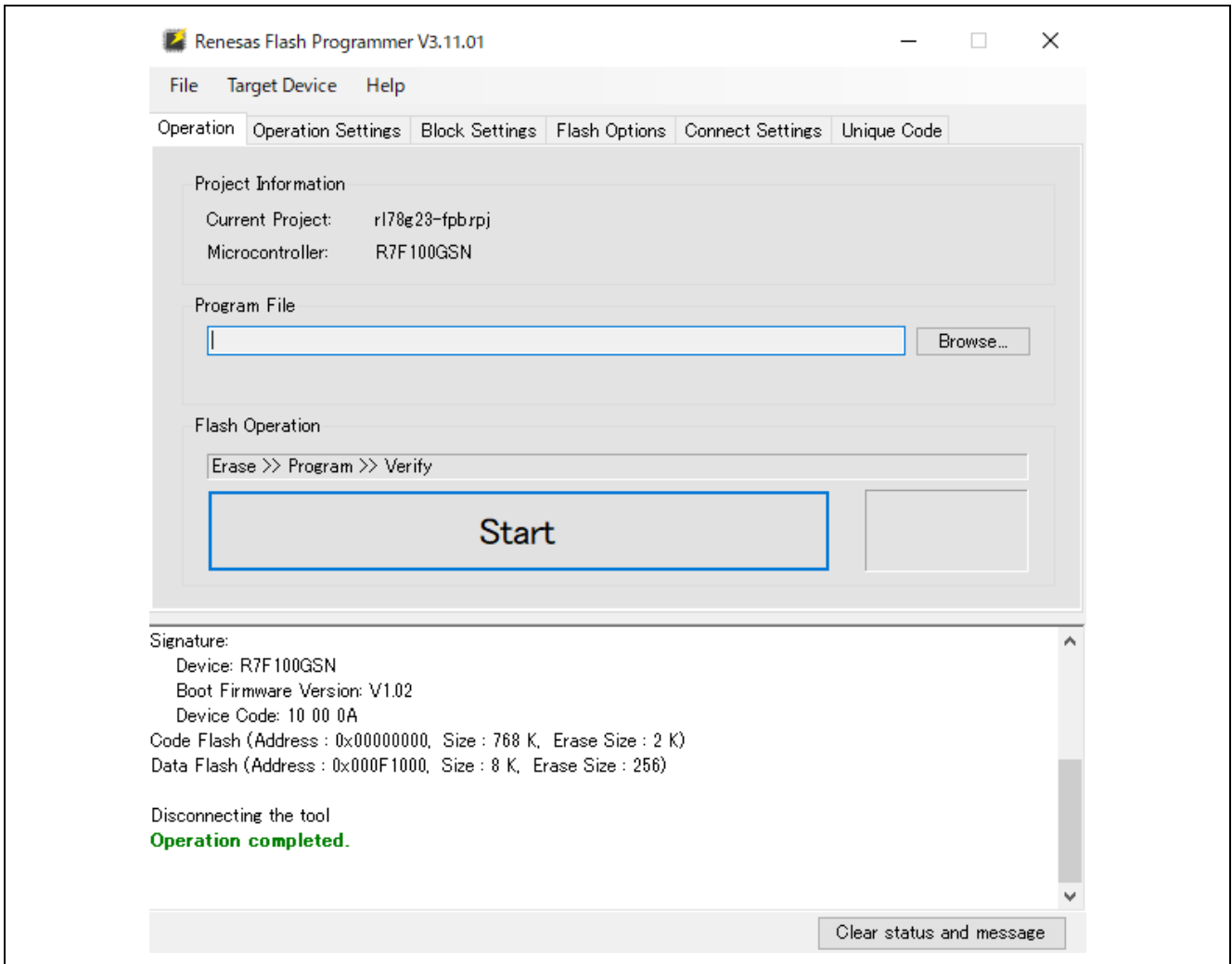


Figure 7-3 Operation completed (Connect)

7.1.4 Programming a MOT File to the MCU Board

(1) In the Program File field, enter the path to the MOT file to be programmed, and then click “Start”.

- Program File: MOT file to be programmed (Example: initial_image.mot, aws_ryz024a_rl78g23-fpb.mot)
- Click “Start”

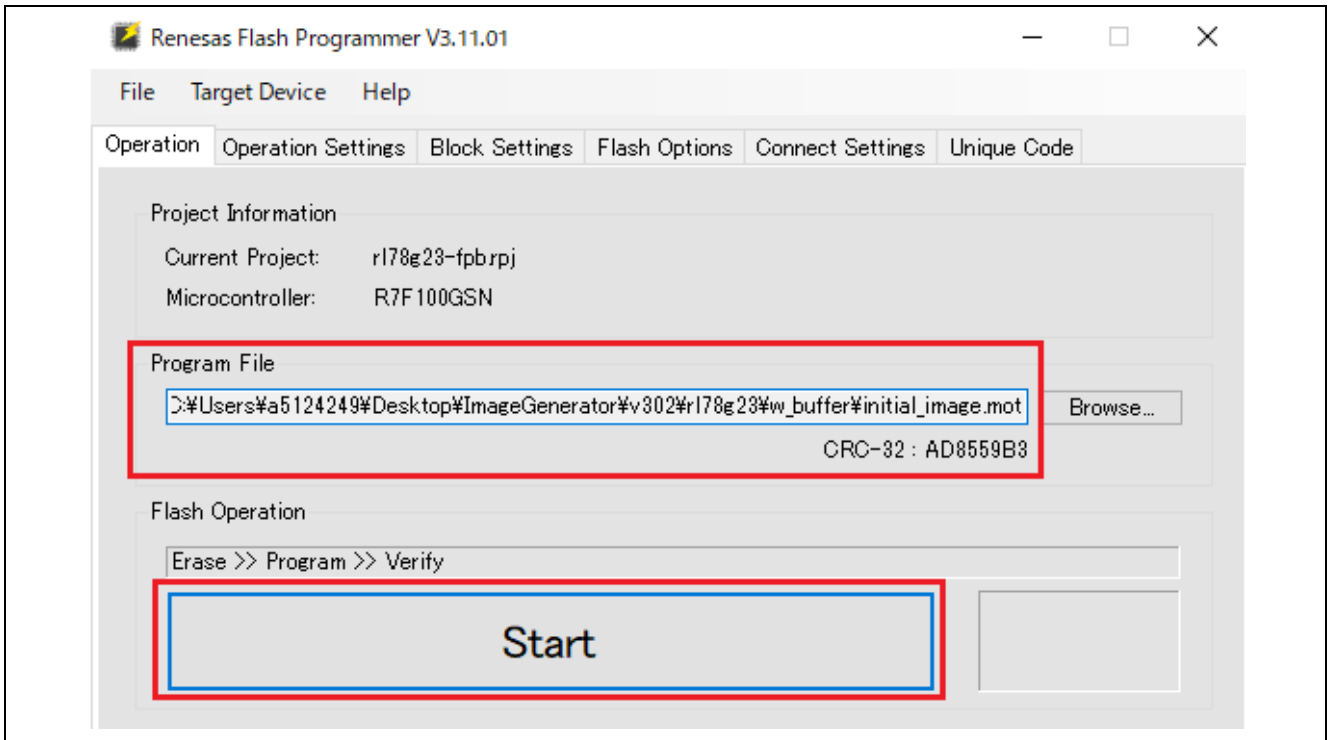


Figure 7-4 Programming a MOT File to the MCU Board

(2) Make sure that programming is successful.

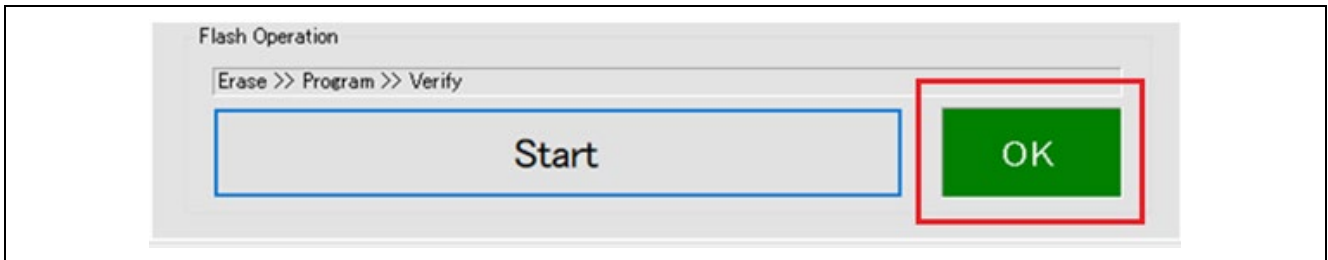


Figure 7-5 Successful programming

7.2 When Using Emulator

The following describes how to program a MOT file via the emulator.

7.2.1 Setting Jumper Pins, Mounting the Connector, and Cutting Patterns

The 14-pin connector (J11) is used for connection with the E2 emulator or E2 emulator Lite, which are Renesas Electronics on-chip debug emulators with the programming feature (the connector component is not mounted). Use the emulator to program and debug the evaluation MCU.

When connecting the emulator, you need to change the circuit as following figures. For details, refer to section 5.2.0 in [RL78/G23-128p Fast Prototyping Board User's Manual Rev.1.00](#).

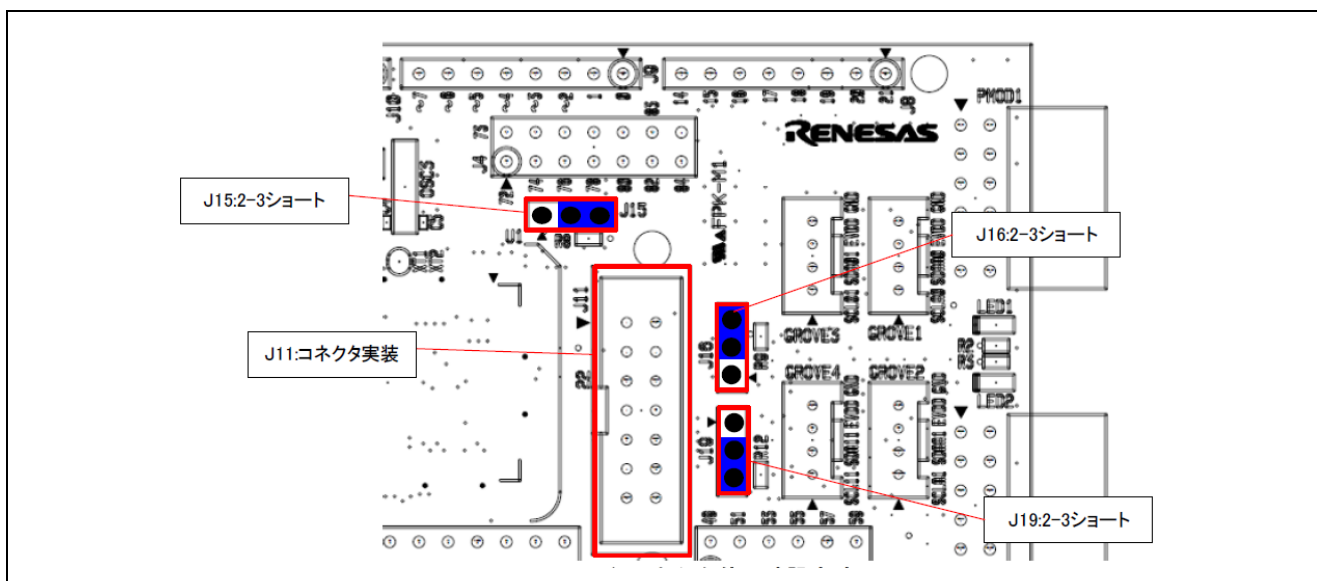


Figure 7-6 Settings for Using Emulator Connector (Top Side)

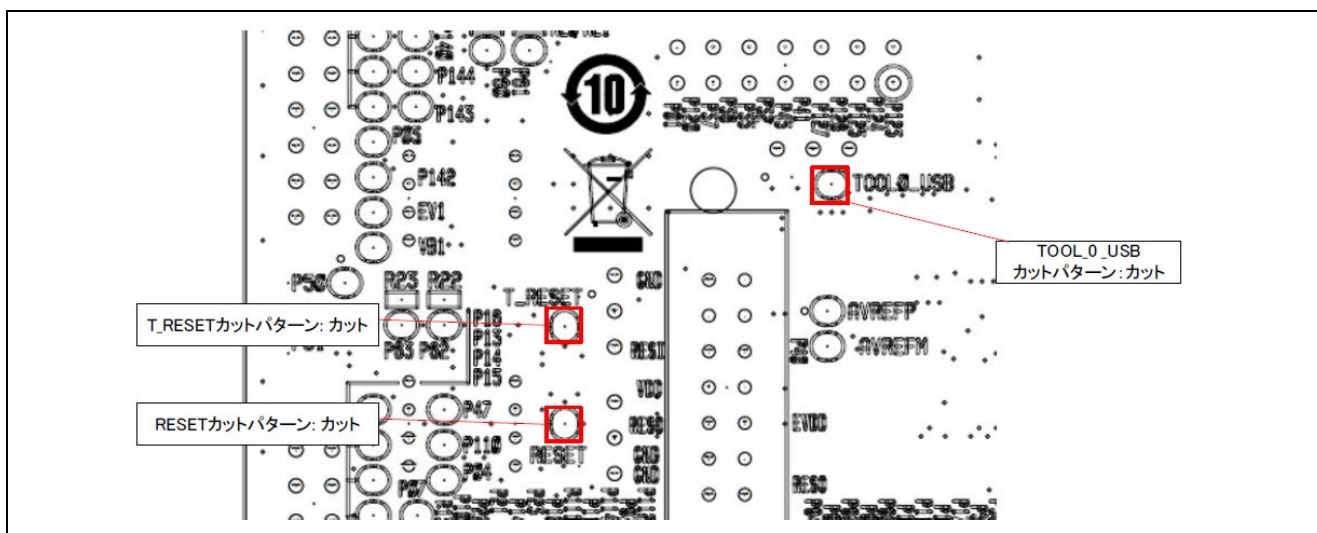


Figure 7-7 Settings for Using Emulator Connector (Solder Side)

For details about how to use the emulator, refer to “[E1/E20/E2 Emulator, E2 Emulator Lite Additional Document for User's Manual \(Notes on Connection of RL78\)](#)” (R20UT1994).

7.2.2 Supplying Power to the MCU Board

Connect the USB cable to supply power to the MCU board.

7.2.3 Creating a New Project and Connecting to the MCU Board

(1) File > New project

- Microcontroller: RL78/G2x
- Project Name: Any (Example: rl78g23-fpb)
- Project Folder: Any
- Tool: E2 emulator
- Click “Connect”

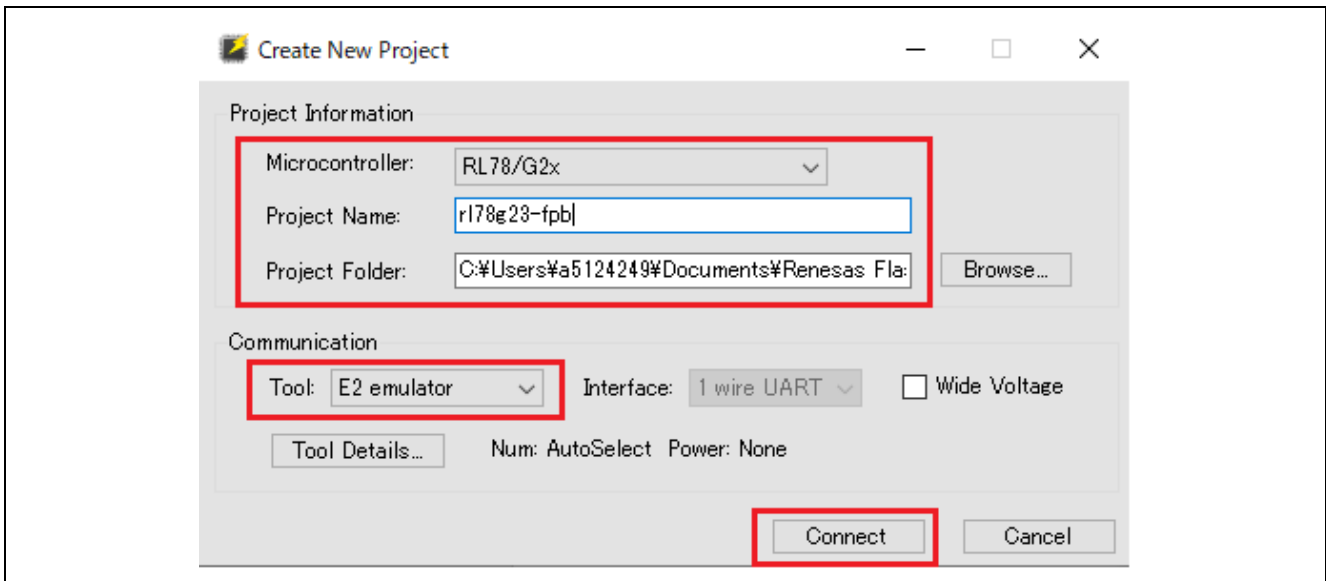


Figure 7-8 Creating a New Project and Connecting to the MCU Board

(2) The connection is successful if the following window appears.

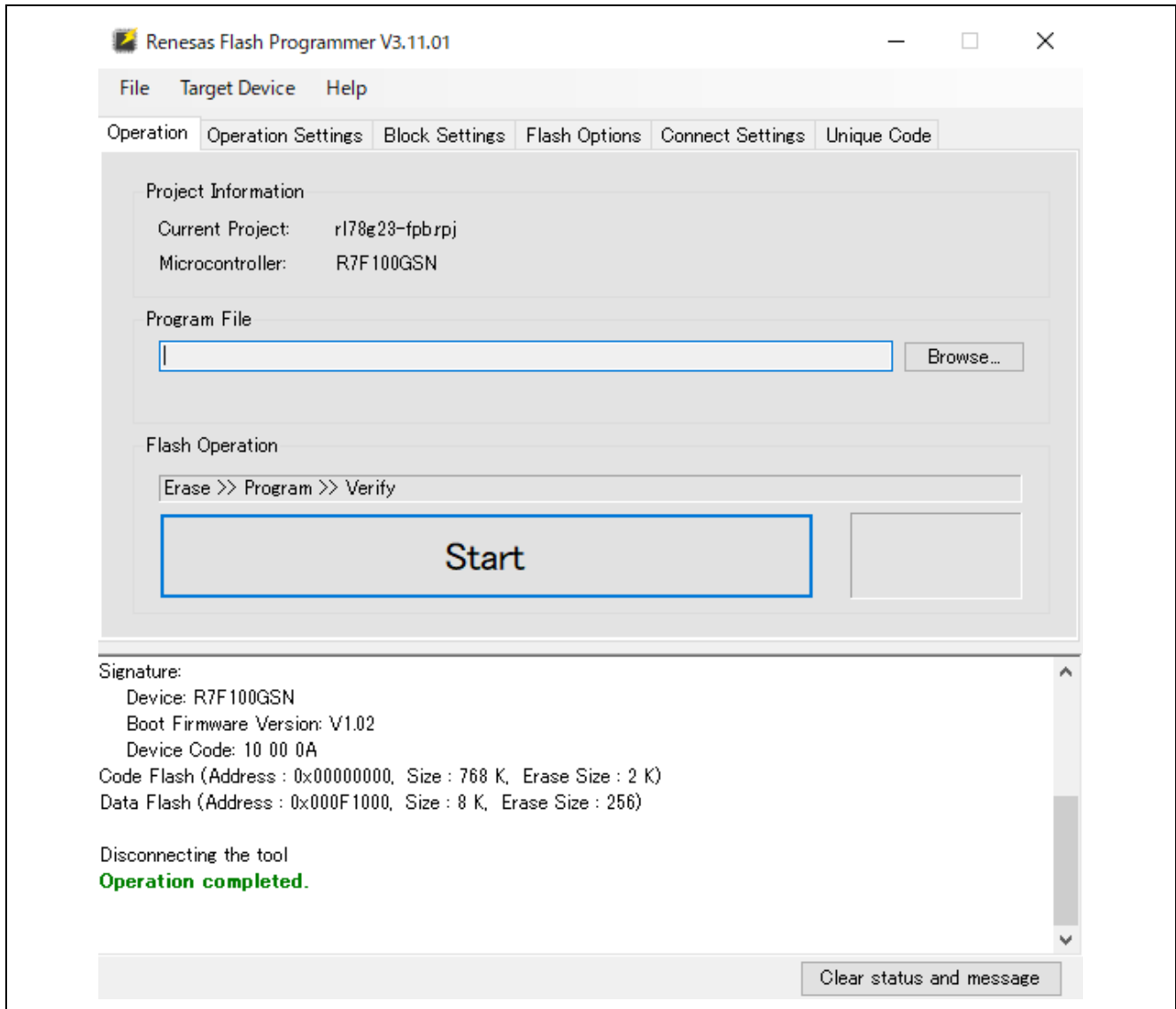


Figure 7-9 Operation completed (Connect)

7.2.4 Programming a MOT File to the MCU Board

Refer to section 7.1.4, Programming a MOT File to the MCU Board.

8. Debug Procedure

8.1 When Using COM Port

The following describes how to perform debugging by using the COM port.

8.1.1 Setting Jumper Pins

Refer to section 7.1.1, Setting Jumper Pins.

8.1.2 Supplying Power to the MCU Board

Connect the MCU board to the PC by using the USB cable.

8.1.3 Debug Configurations

Select the configuration you want to use for debugging.

- Debug Configurations > Renesas GDB Hardware Debugging
— For the demo project (PubSub), select `aws_ryz024a_rl78g23-fpb HardwareDebug`.

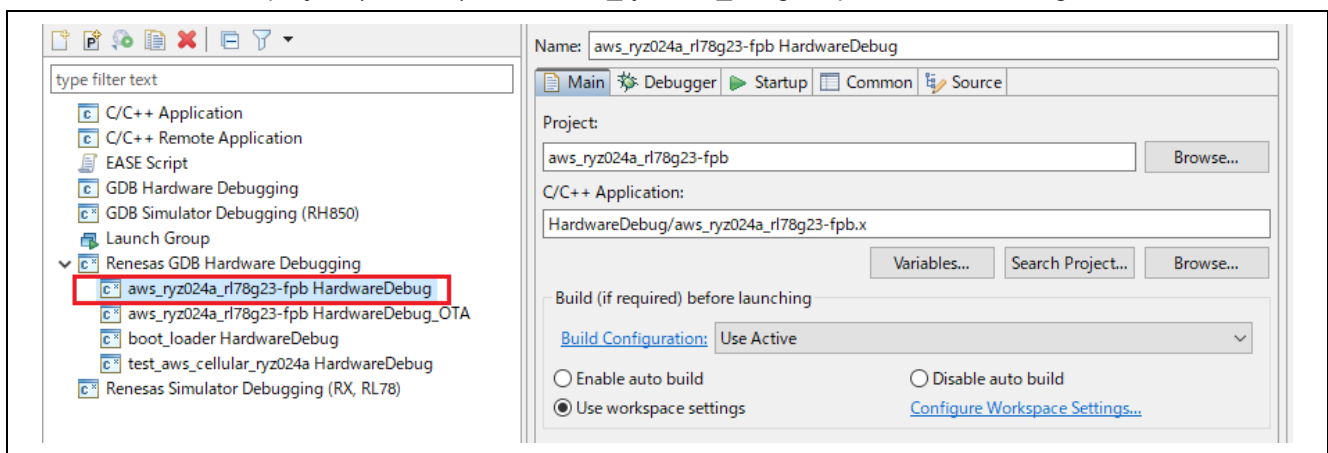


Figure 8-1 Debug Configurations of Project (PubSub)

- For the demo project (OTA), select `aws_ryz024a_rl78g23-fpb HardwareDebug_OTA`.

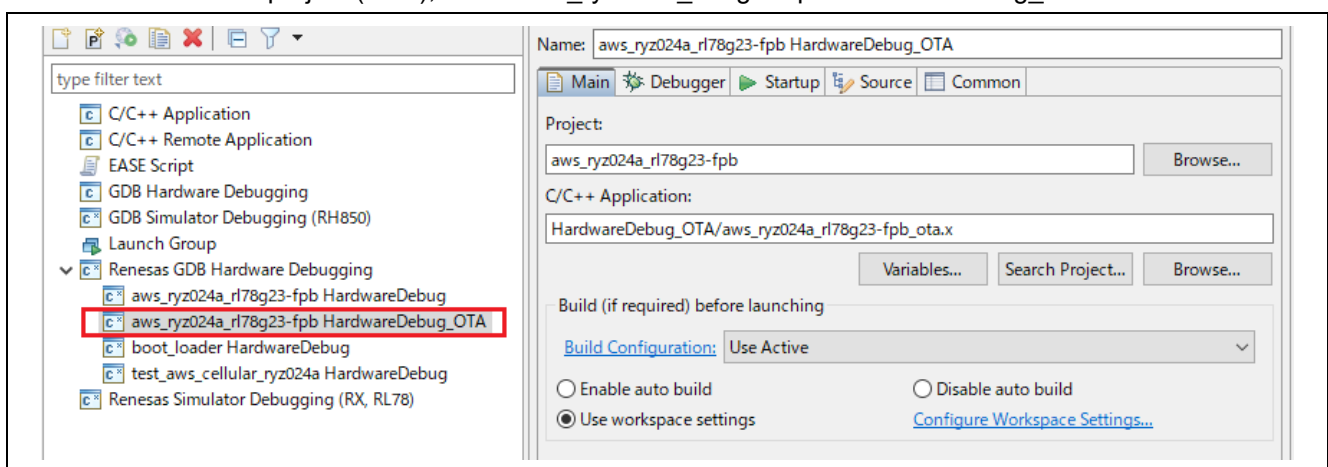


Figure 8-2 Debug Configurations of Project (OTA)

8.1.4 Debugger Settings

Select "Debugger" tab.

- Debug hardware: COM Port (RL78)

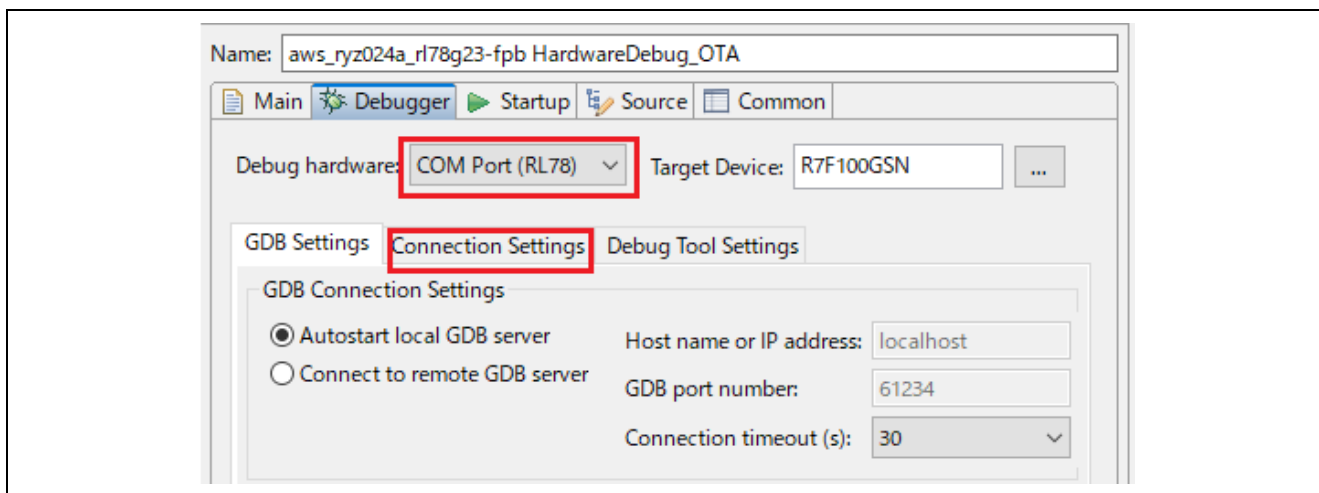


Figure 8-3 Debug hardware: COM Port (RL78)

Select "Connection Settings" tab > Connection with Target Board.

- COM Port: COMxx
- Reset control pin: DTR

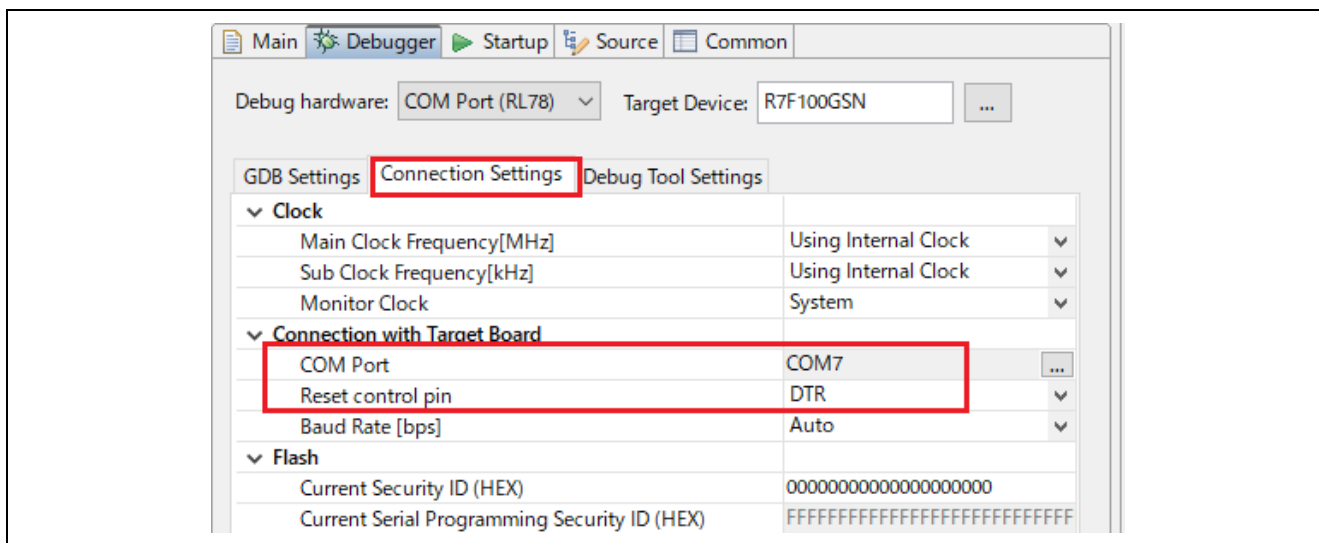
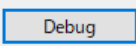


Figure 8-4 Connection Settings for Using COM Port

Start debugging by clicking .

8.2 When Using Emulator

The following describes how to perform debugging by using the E2 emulator Lite.

8.2.1 Mounting the Connector, Setting Jumper Pins, and Cutting Patterns

Refer to section 7.2.1, Setting Jumper Pins, Mounting the Connector, and Cutting Patterns.

8.2.2 Connecting the Emulator to the MCU Board

Connect the emulator as shown in the following figure.

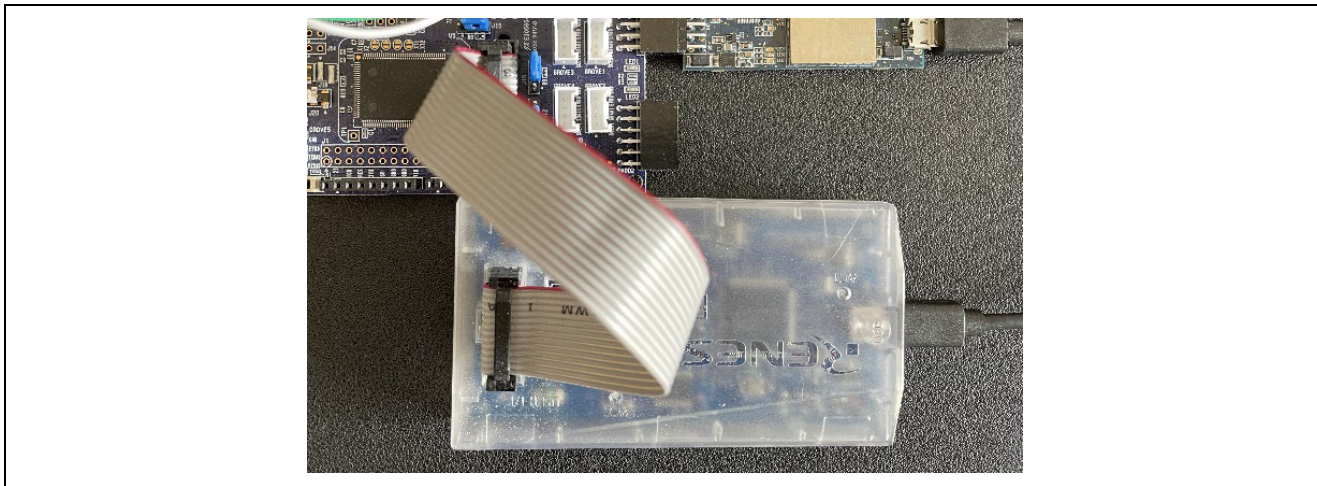


Figure 8-5 Connecting Emulator to MCU Board

8.2.3 Debug Configurations

Select the configuration you want to use for debugging.

- Debug Configurations > Renesas GDB Hardware Debugging
 - For the demo project (PubSub), select aws_ryz024a_rl78g23-fpb HardwareDebug.

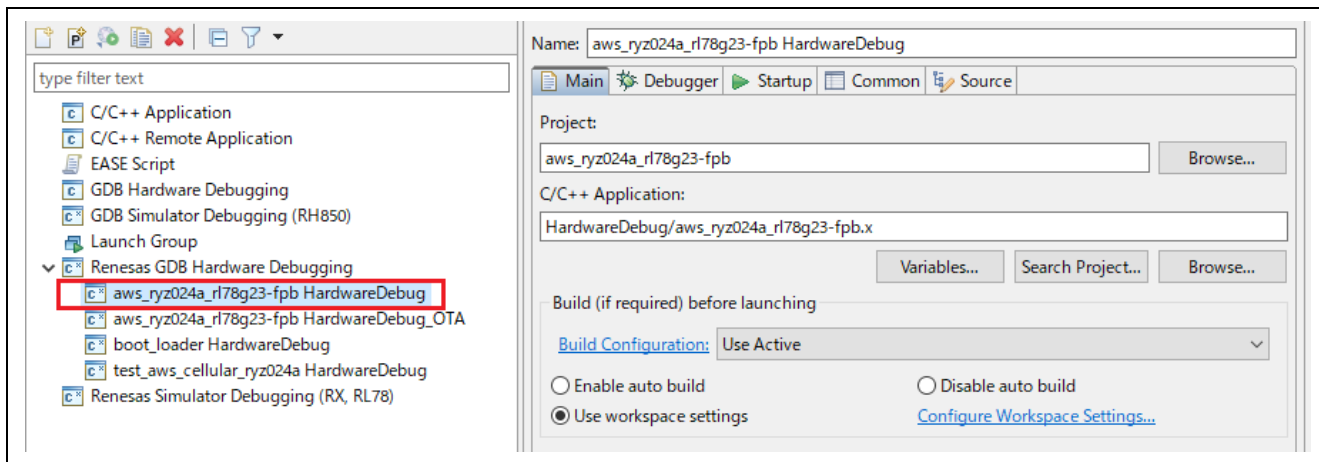


Figure 8-6 Debug Configurations of Project (PubSub)

— For the demo project (OTA), select aws_ryz024a_rl78g23-fpb HardwareDebug_OTA.

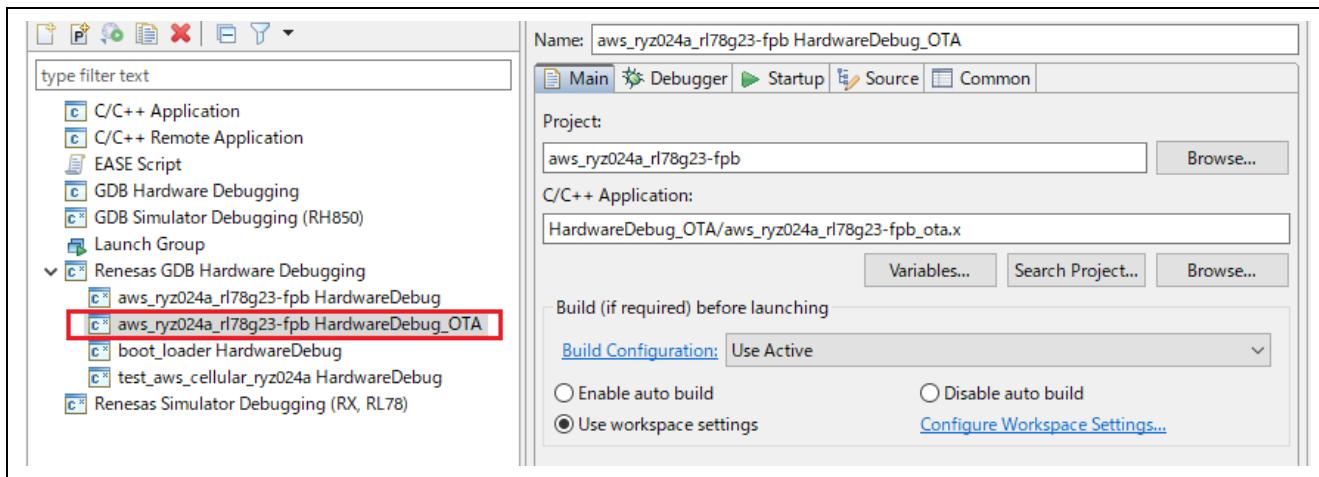


Figure 8-7 Debug Configurations of Project (OTA)

8.2.4 Debugger Settings

Select "Debugger" Tab.

- Debug hardware : E2 Lite (RL78)

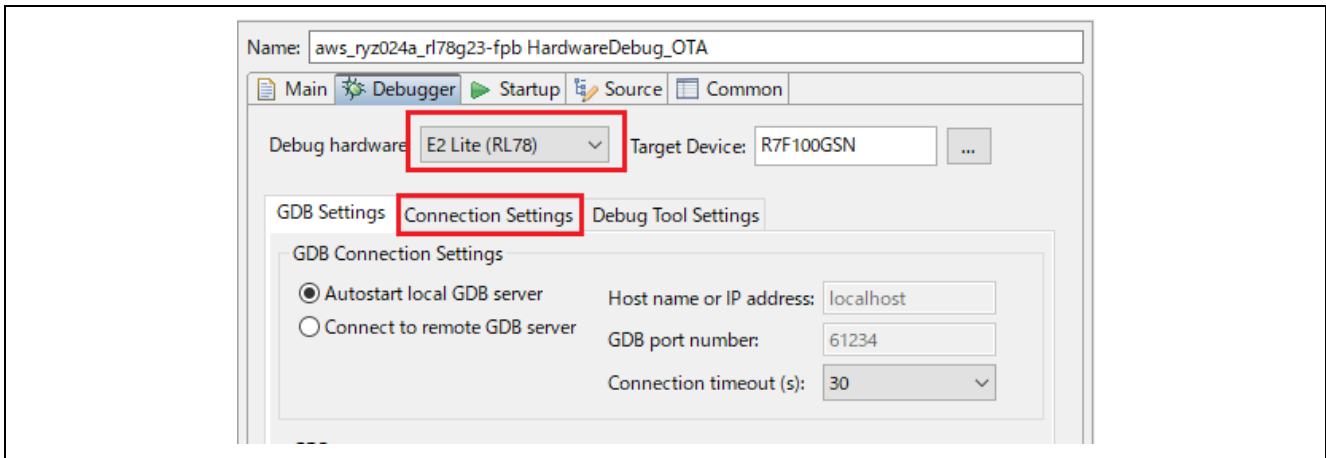


Figure 8-8 Debug hardware: E2 Lite (RL78)

Select "Connection Settings" tab > Connection with Target Board.

- Power Target From The Emulator (MAX 200mA): No

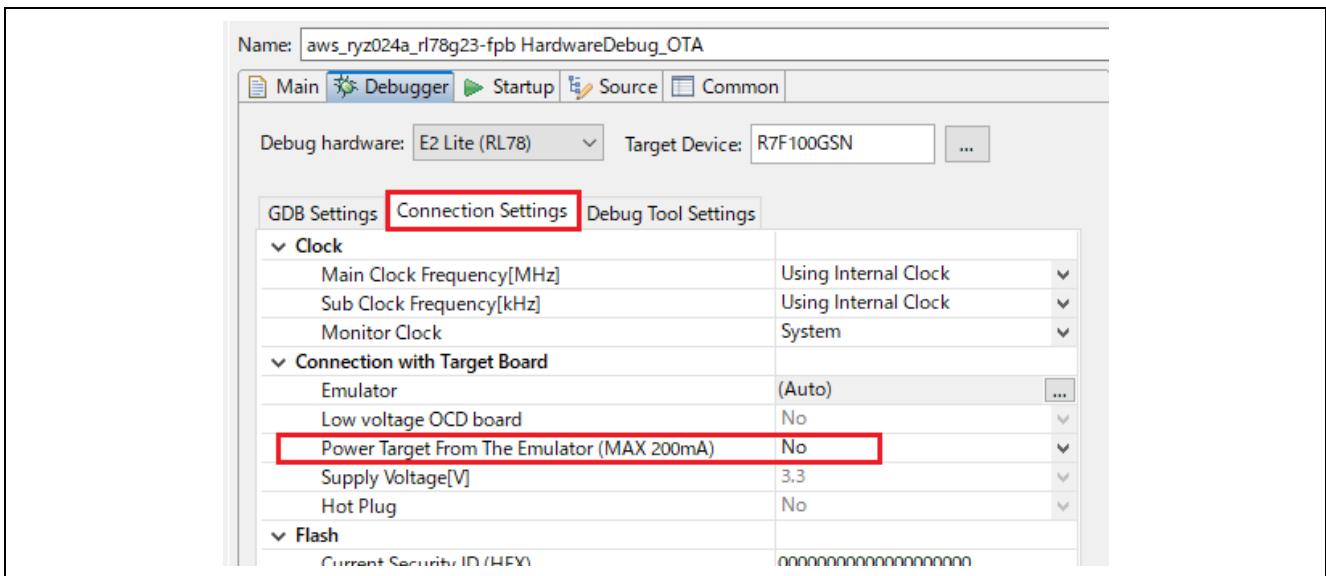
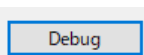


Figure 8-9 Connection Settings for Using Emulator

Start debugging by clicking



9. Appendix

9.1 Precautions on Porting Third-Party Libraries to RL78

Because RL78 is a 16-bit system, the following must be noted when applying a third-party library to RL78.

9.1.1 Width of int Is 16 Bits

Modification might be required in the parts in which processing-dependent types (such as int and size_t) are used. Pay particular attention in the case of variables that handle the size.

This demo projects modified the following libraries:

- tinycbor(0.5.2) <https://github.com/intel/tinycbor>
- TinyCrypt Cryptographic Library (0.2.8) <https://github.com/intel/tinycrypt>

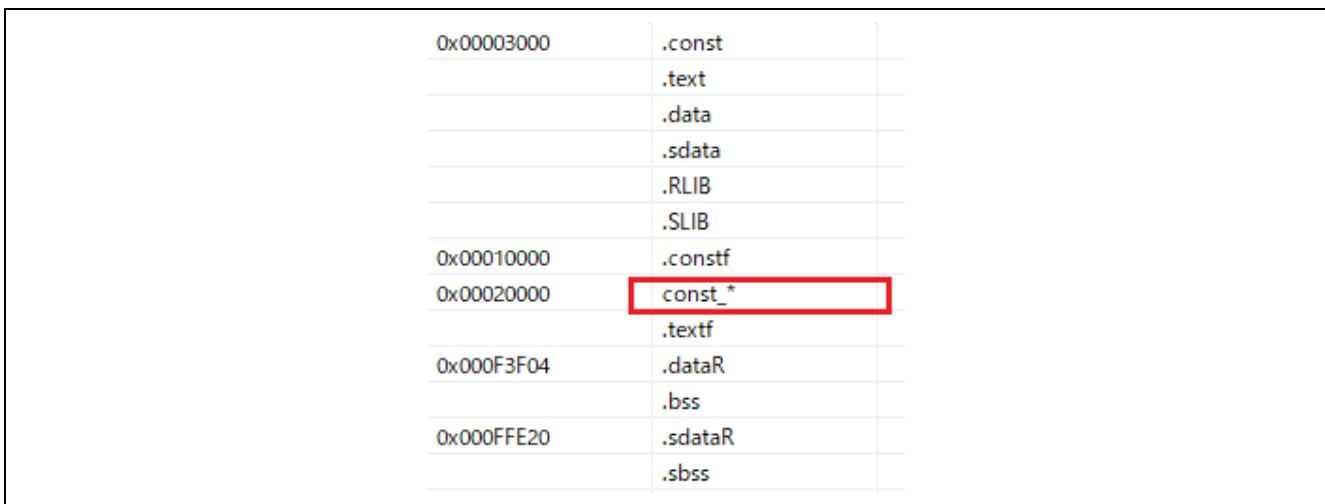
9.1.2 Size Limitation of Section

Some sections cannot extend across a boundary of 64KB – 1; in other words, they can only allocate a maximum size of 64KB. Therefore, for example, if porting a large library to RL78, data larger than 64KB may be allocated in a default section, causing a linker error.

For details, refer to [CC-RL Compiler User's Manual](#) (R20UT3123).

To avoid this limitation, you need to adjust section size. The following explains how to adjust the default constant section (.constf) as an example.

First, define a new constant section.



| | | |
|------------|---------|--|
| 0x00003000 | .const | |
| | .text | |
| | .data | |
| | .sdata | |
| | .RLIB | |
| | .SLIB | |
| 0x00010000 | .constf | |
| 0x00020000 | const_* | |
| | .textf | |
| 0x000F3F04 | .dataR | |
| | .bss | |
| 0x000FFE20 | .sdataR | |
| | .sbss | |

Figure 9-1 Newly Defined Constant Section (e² studio)

Next, change section so that library data is allocated in the newly defined constant section by one of the following methods.

(1) #pragma section directive

Add #pragma section directive to library source codes.

example:core_mqtt.c

```
#if defined(__CCRL__) || defined(__ICCRL78__) || defined(__RL)
#pragma section const const_coreMqtt
#endif

/**
 * @file core_mqtt.c
 * @brief Implements the user-facing functions in core_mqtt.h.
 */
#include <string.h>
#include <assert.h>

...Codes...

#if defined(__CCRL__) || defined(__ICCRL78__) || defined(__RL)
#pragma section
#endif
```

Figure 9-2 Added #pragma section Directive (3rd Party Library)

(2) Link option -REName

Change section so that library data is allocated in the newly defined constant section for each file by specifying a link option as shown following. This method has the advantage that you don't need to modify source files.

```
-REName=.\Middleware\FreeRTOS\coreMQTT\source\core_mqtt.obj(.constf=const_coreMqtt_f)
```

9.2 License Information for Open Source Software Used with Demo Projects

The demo projects of this product use open source software (OSS). The user must comply with the license terms stipulated by OSS. Check the license terms on the official website of the respective OSS. Table 1-3 Operation Confirmation Conditions (Others, such as OSS Library) shows the link of each OSS.

10. Websites and Supports

Sample programs in this Getting Started Guide: <https://github.com/renesas/iot-reference-rl78>

AWS forum: <http://forums.aws.amazon.com>

Revision History

| Rev. | Date | Description | |
|------|---------------|-------------|--|
| | | Page | Summary |
| 1.00 | Feb. 29, 2024 | - | First edition issued |
| 1.01 | Aug. 30, 2024 | 1 | Added information about EOL process on RYZ024A cellular module |
| | | 18, 71 | Added information for case of without changing circuit to mount emulator connector |
| | | 43 | Fixed the value of "Resource" described to the Policy editor |

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/.