RL78/G23

Getting Started Guide for Connecting Amazon Web Services (202406-LTS Version) in Wi-Fi Communication: FPB-RL78G23-128p + FreeRTOS

Introduction

This document describes how to connect to Amazon Web Services (AWS) by using a Renesas MCU board combined with Wi-Fi DA16600 module.

Target Device

RL78 Family
RL78/G2x Series
RL78/G23 Group

Related Documents

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

RL78/G22, RL78/G23, RL78/G24, RL78/L23 Firmware Update Module (R01AN6374)

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

US159-DA16600EVZ Evaluation Board Manual (R15UZ0006)

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

RL78 Family US159-DA16XXXMEVZ Wi-Fi Control Module Using Software Integration System (R01AN7321)

Contents

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 Wi-Fi DA16600 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 using AWS.

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 2025-10.0
C compiler	Renesas Electronics CC-RL V1.15.01
Firmware programming tool	Renesas Flash Programmer V3.20.00
Firmware update module (FWUP)	https://www.renesas.com/en/document/apn/rl78g22-rl78g23-rl78g24-rl78l23-firmware-update-module
Smart Configurator (SC)	Renesas Smart Configurator for RL78 25.10.0.v20250912-0119
Board support package (BSP)	v1.91 (r_bsp)
Python	Python 3.12.1
OpenSSL	OpenSSL 3.4.0
SDK (Software Development Kit)	DA16200/DA16600 SDK V3.3.0.0 Download the firmware using this link: https://www.renesas.com/en/document/sws/da16200-da16600-freertos-sdk-image-v3300
DA16XXX Wi-Fi control module	RL78 Family US159-DA16XXXMEVZ Wi-Fi Control Module Using Software Integration System v1.40

Table 1.3 Operation Confirmation Conditions (Others, such as OSS Library)

Item	Description	Remarks
iot-reference-rl78	v202406.04-LTS-rl78-1.1.0 (Based FreeRTOS 202406.04-LTS)	
FreeRTOS Kernel	11.1.0 https://github.com/FreeRTOS/FreeRTOS-Kernel	
backoffAlgorithm	1.4.1 https://github.com/FreeRTOS/backoffAlgorithm	
coreJSON	3.3.0 https://github.com/FreeRTOS/coreJSON	
coreMQTT Client	2.3.1 https://github.com/FreeRTOS/coreMQTT	
coreMQTT Agent	1.3.1 https://github.com/FreeRTOS/coreMQTT-Agent	
AWS IoT Jobs	1.5.1 https://github.com/aws/Jobs-for-AWS-IoT-embedded-sdk	Modified OSS libraries
AWS IoT MQTT File Streams	1.1.0 https://github.com/aws/aws-iot-core-mqtt-file-streams-embedded-c	Modified OSS libraries
tinycbor	0.5.2 https://github.com/intel/tinycbor	
FreeRTOS-Plus network_transport	No version Transport Interface - FreeRTOS™	
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 FPB-RL78G23-128p - RL78/G23-128p Fast Prototyping Board
Wi-Fi DA16600 module	PMOD Expansion Board for DA16600MOD US159-DA16600EVZ - Ultra-Low-Power Wi-Fi + Bluetooth Low Energy Combo Pmod Board | Renesas
USB-UART conversion board	Pmod USBUART CP2101 USB TO TTL BOARD V4.2 (agencyelectronics.com)
Micro USB Type-B cable x 2	• Used to connect the USB-UART conversion board to the PC • Used to connect the MCU board to the PC
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.

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).

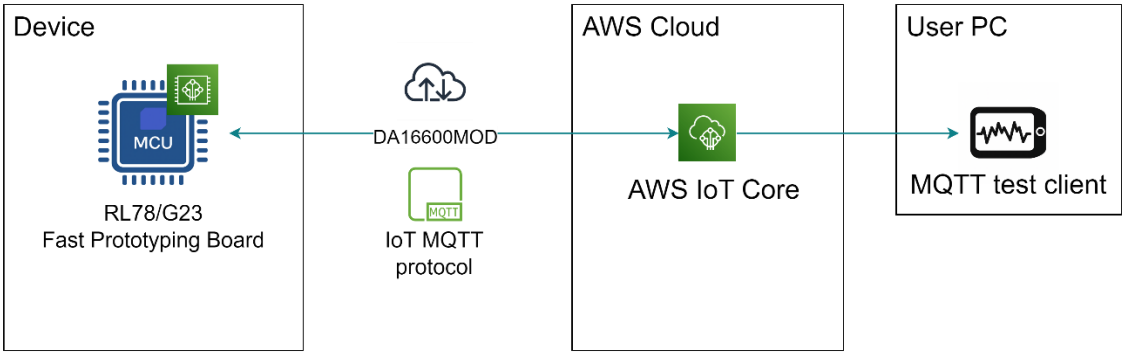


Figure 2-1 System Configuration of Demo Project (PubSub)

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 DA16600MOD
P144/TxD3	Output	UART communication (transmission) with DA16600MOD
P00	Output	Reset to DA16600MOD
P142	Output	UART communication (RTS) with DA16600MOD
P14/RxD2	Input	Terminal input
P13/TxD2	Output	Terminal output
P50	Output	LED1

2.2 Demo Project (OTA)

2.2.1 System Configuration

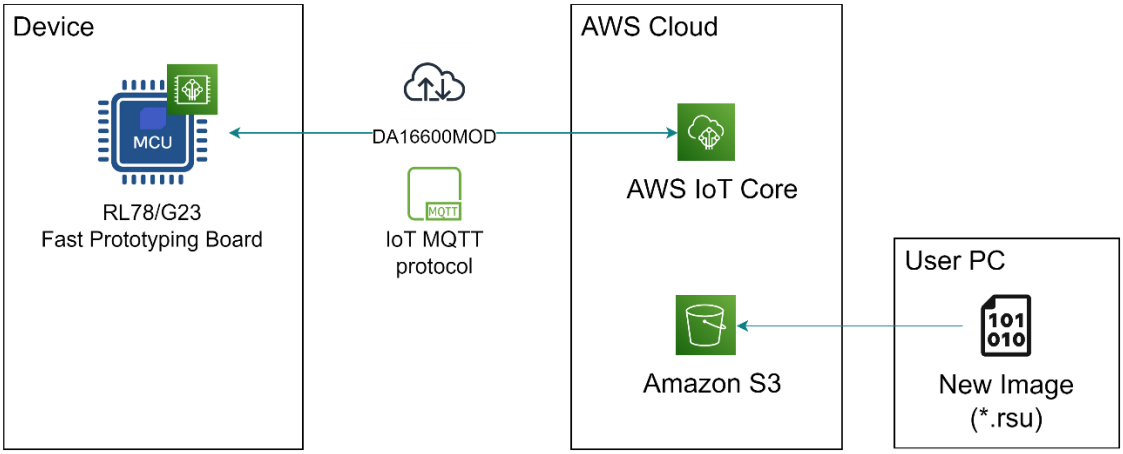


Figure 2-2 System Configuration of Demo Project (OTA)

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 DA16600MOD
P144/TxD3	Output	UART communication (transmission) with DA16600MOD
P00	Output	Reset to DA16600MOD
P142	Output	UART communication (RTS) with DA16600MOD
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, RL78/L23 Firmware Update Module".

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

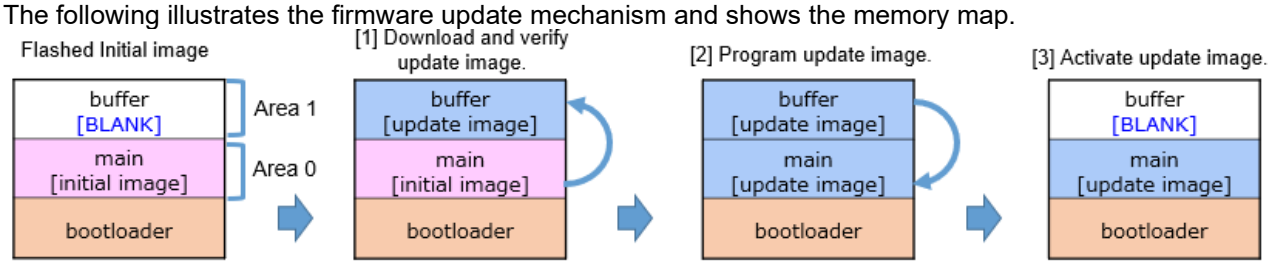


Figure 3-1 Firmware Update Mechanism

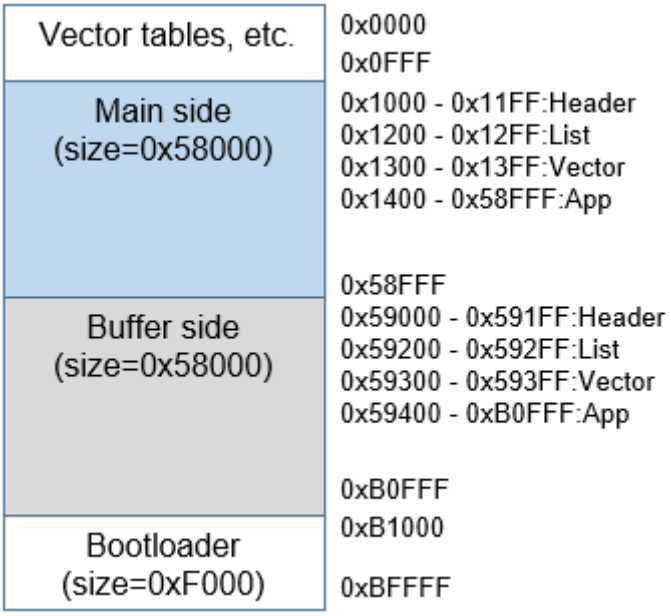


Figure 3-2 Memory Map of Demo Project (OTA)

3.2.2 List of Option Bytes Settings

The following shows the option bytes settings.

Table 3.2 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.

3.3 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	
│ ├─patches	
│ └─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	
│ │ └─aws-iot-core-mqtt-file-streams-embedded-c	
│ │ └─Jobs-for-AWS-IoT-embedded-sdk	
│ ├─FreeRTOS	FreeRTOS Kernel and libraries.
│ │ ├─backoffAlgorithm	
│ │ ├─coreJSON	
│ │ ├─coreMQTT	
│ │ ├─coreMQTT-Agent	
│ │ └─FreeRTOS-Kernel	
│ ├─logging	
│ └─wifi	
├─Projects	
│ └─rl78g23-fpb	
│ ├─application_code	
│ ├─flash_proj	
│ ├─helper	
│ ├─modules	
│ ├─projects	Import below folders to IDE.
│ │ ├─aws_da16600_rl78g23-fpb	PubSub demo and OTA demo. Select by Build Configurations.
│ │ ├─boot_loader	Boot loader for OTA demo
│ │ ├─test_aws_da16600_rl78g23-fpb	This project is for validation tests and will not normally be used.

Folder Name	Description
│ └─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_da16600_rl78g23-fpb (demo project (PubSub))	124879	31968
aws_da16600_rl78g23-fpb (demo project (OTA))	189522	42044
Bootloader	22318	1365

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.

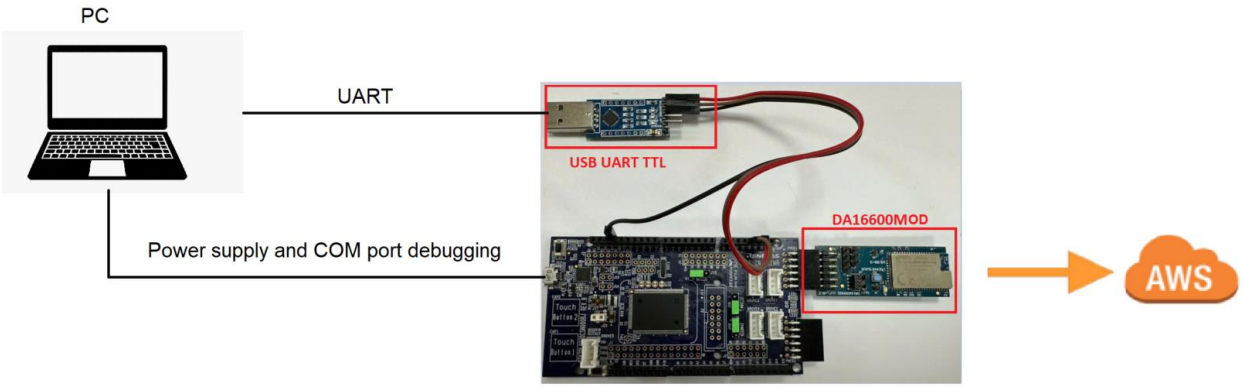


Figure 4-1 Overall Hardware Configuration of the Demo Project

4.1.2 Connecting Hardware

The following describes how to connect hardware.

① Connect DA16600MOD to PMOD1 of the MCU board.

Figure 4-2 Connecting DA16600MOD to PMOD1 of the MCU Board

② Connect the USB-UART conversion board to the MCU board.

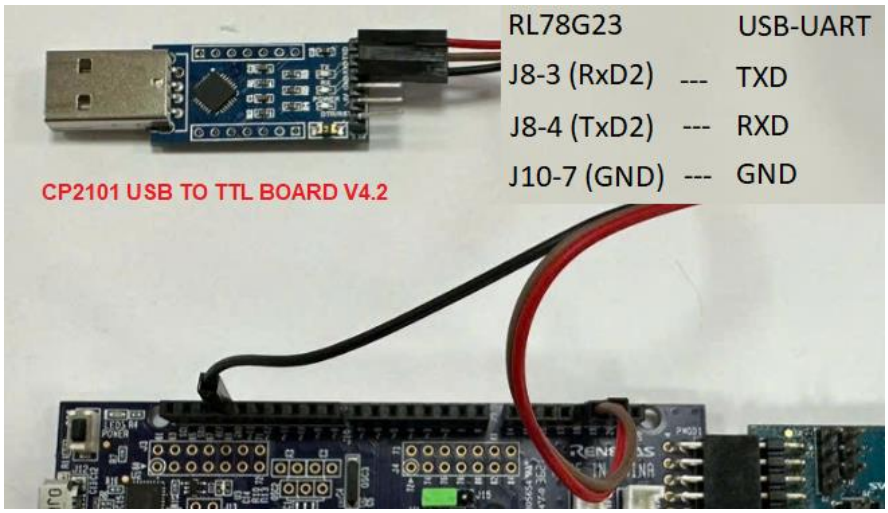


Figure 4-3 Connecting the USB-UART Conversion Board to the MCU Board

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

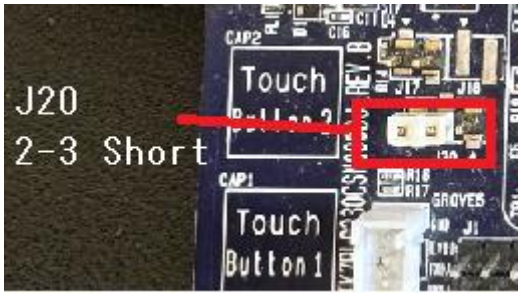


Figure 4-4 Setting MCU Board Power Supply to 3.3 V

④ 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.

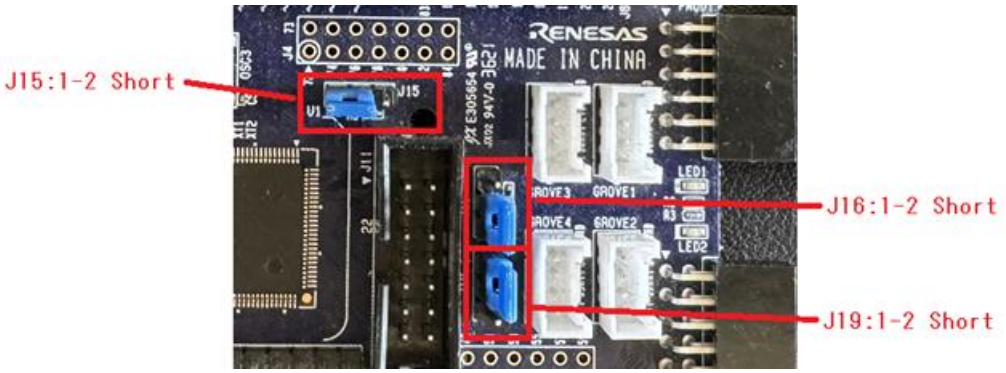


Figure 4-5 Settings for Using COM Port Debugging (Top Side)

⑤ Connect the USB cable to supply power to the MCU board.

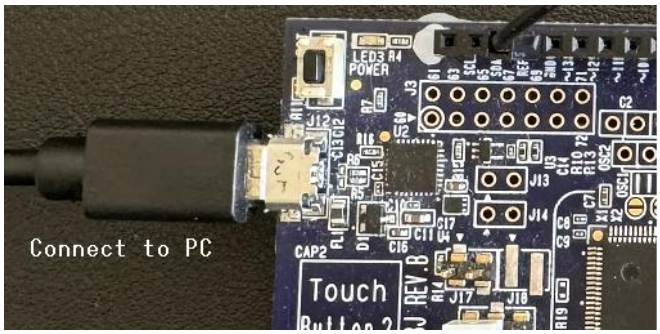


Figure 4-6 Supplying Power to the MCU Board

⑥ Confirm the COM port number.

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

⑦ 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 Wi-Fi Information to the Demo Project

Specify the Wi-Fi information for the following macros in the demo project.

- iot-reference-rl78\Projects\rl78g23-fpb\modules\r_config\r_wifi_da16xxx_config.h
  - WIFI_CFG_COUNTRY_CODE: Country code in ISO3166-1 alpha-2 standard
  - WIFI_CFG_TLS_SUPPORT: TLS on-chip protocol
  - WIFI_CFG_TLS_CERT_NAME: CA certificate name
  - WIFI_CFG_TLS_CERT_CLIENT_NAME: Client certificate name
  - WIFI_CFG_TLS_CERT_PRIVATE_NAME: Private certificate name

Note: Only the highlighted part in yellow is allowed to be changed.

iot-reference-rl78\Projects\rl78g23-fpb\modules\r_config\r_wifi_da16xxx_config.h

```
/* Country code
   Country code defined in ISO3166-1 alpha-2 standard.
 */
#define WIFI_CFG_COUNTRY_CODE                    "VN"
…
/* Enables or disables TLS on-chip protocol.
    0 = disabled.
    1 = enabled.
 */
#define WIFI_CFG_TLS_SUPPORT                     1
…
/* Configures CA certificate name.
 */
#define WIFI_CFG_TLS_CERT_CA_NAME                "CA_CERT"

/* Configures Client certificate name.
 */
#define WIFI_CFG_TLS_CERT_CLIENT_NAME            "CERT_KEY"

/* Configures Private certificate name.
 */
#define WIFI_CFG_TLS_CERT_PRIVATE_NAME            "CERT_KEY"
```

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, thing name to the demo project.
iot-reference-rl78\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 Wi-Fi network to the demo project.
iot-reference-rl78\Demos\include\aws_clientcredential.h

```
/*
 * @brief Wi-Fi network to join.
 *
 * @todo If you are using Wi-Fi, set this to your network name.
 */
#define clientcredentialWIFI_SSID                    "YOUR_WIFI_SSID"
/*
 * @brief Password needed to join Wi-Fi network.
 * @todo If you are using WPA, set this to your network password.
 */
#define clientcredentialWIFI_PASSWORD                "YOUR_WIFI_PASSWORD"
/*
 * @brief Wi-Fi network security type.
 *
 * @see WIFISecurity_t.
 *
 * @note Possible values are eWiFiSecurityOpen, eWiFiSecurityWEP, eWiFiSe-
curityWPA,
 * eWiFiSecurityWPA2 (depending on the support of your device Wi-Fi radio).
 */
#define clientcredentialWIFI_SECURITY                YOUR_WIFI_SECURITY
```

④ 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"\
"MIIEowIBAAKCAQEA3Fb7O7jQW4lgHmPE3AInUTWUCaR7kWeWHubEk9YbNf3xwxdg\n"\
...
"s/OlVUiygf0RgeoMVx/3GzZPfmTrB0cQ8XZ7mxCd2dgY9UXQ/oja\n"\
"-----END RSA PRIVATE KEY-----\n"
```

⑤ Set the hook function (Only using Wi-Fi to the demo project).
For PubSub demo project :
iot-reference-rl78\Configuration\rl78g23-fpb\pubsub\wifi\frtos_config\user_tcp_hook_config.h

```
/* The limit of number for WIFI_ERR_MODULE_COM when it comes continuously.
 * If counter is over than this limit, hook the reset */
#define USER_COMM_ERROR_TRIES     3

/*Connection retry limit*/
#define USER_RECONNECT_TRIES      3

/*Socket closing retry limit*/
#define USER_CLOSE_SOCKET_TRIES      3

/*Enable SocketErrorHook*/
#define USER_TCP_HOOK_ENABLED     1 /* Default */

#endif /* FRTOS_CONFIG_USER_TCP_HOOK_CONFIG_H_ */
```

5. Setup Specific to Demo Project (PubSub)

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

5.1 Preparation

Clone the demo project

Download the demo project from GitHub (GitHub - renesas/iot-reference-rl78) to make a clone of it in a desired folder. This application note explains how to make a clone using Git for Windows.

Refer to the above link and install Git for Windows before continuing with the procedure.

Start Git Bash and enter the following command.

```
git clone https://github.com/renesas/iot-reference-rl78.git --recurse-submodule
```

5.2 Importing the Project

Import the aws_da16600_rl78g23-fpb project to e^2 studio. Open the Import wizard according to the following process.

File > Import… > Existing Projects into Workspace > Next

Next, select the aws_da16600_rl78g23-fpb project. Ensure that copy projects into workspace is not selected. Then click the Finish button.

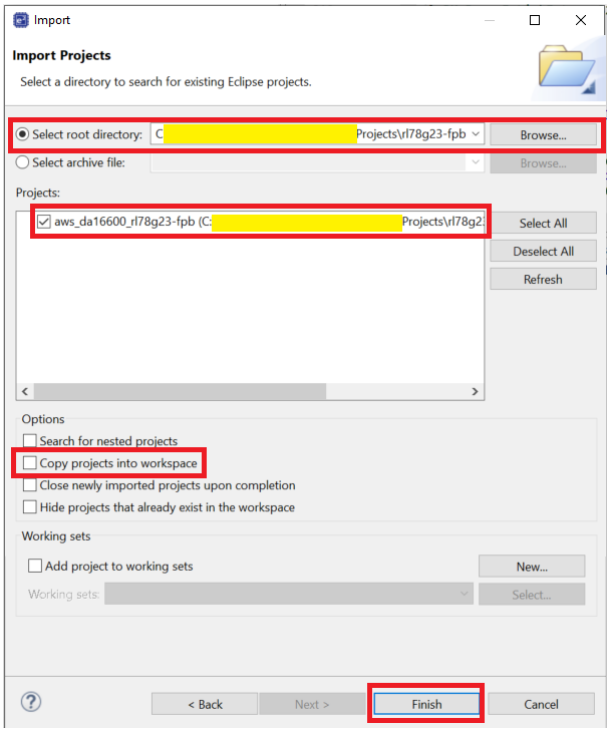


Figure 5-1 Selecting the aws_da16600_rl78g23-fpb Project

The imported project is showed in the Project Explorer view.

Figure 5-2 Completing to Import the aws_da16600_rl78g23-fpb Project

5.3 Setting the Build Configuration

Activate the build configuration “HardwareDebug” of the aws_da16600_rl78g23-fpb project.

Build Configurations > Set Active > Select “HardwareDebug”

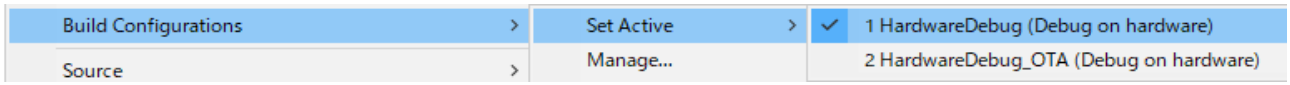


Figure 5-3 Activating Build Configuration “HardwareDebug”

5.4 Building the Demo Project

Build the aws_da16600_rl78g23-fpb project to create a MOT file.
Then, make sure that aws_da16600_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”.

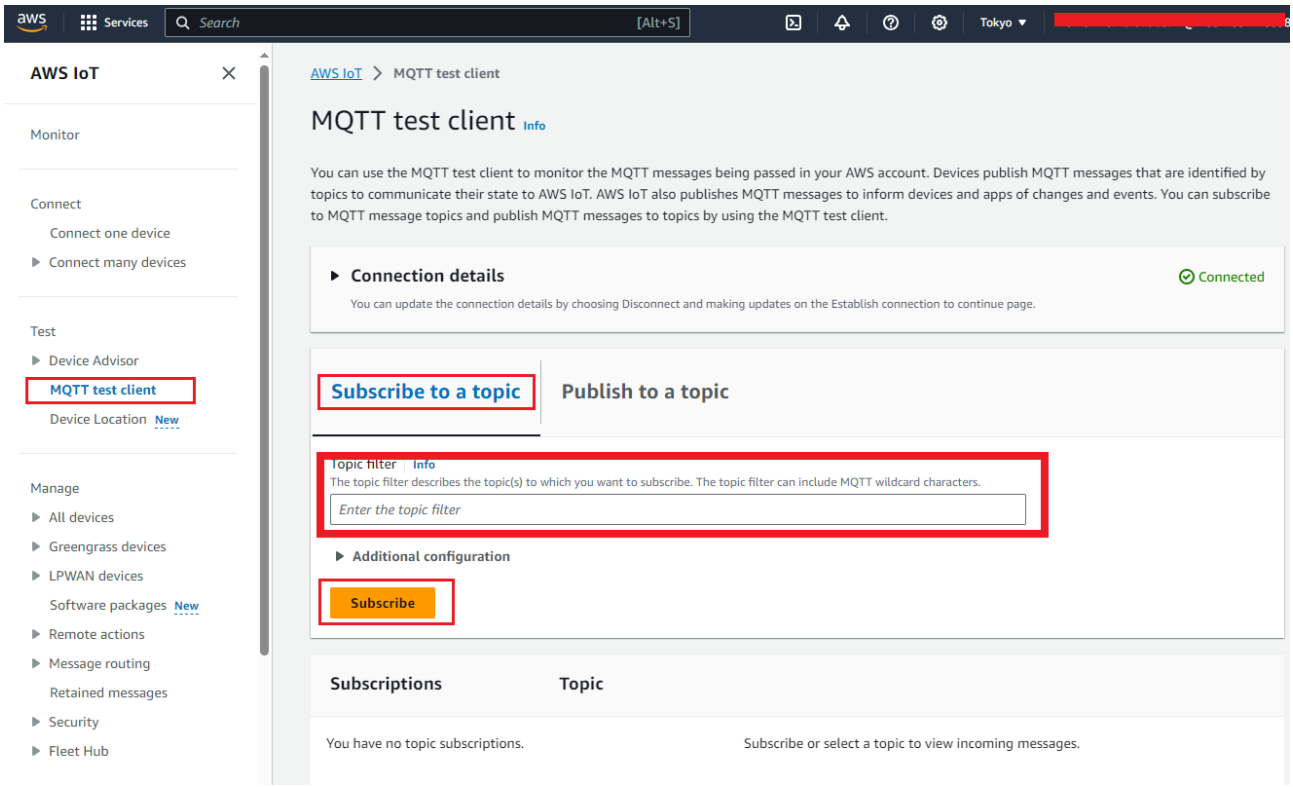


Figure 5-4 MQTT Test Client Settings

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_da16600_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.

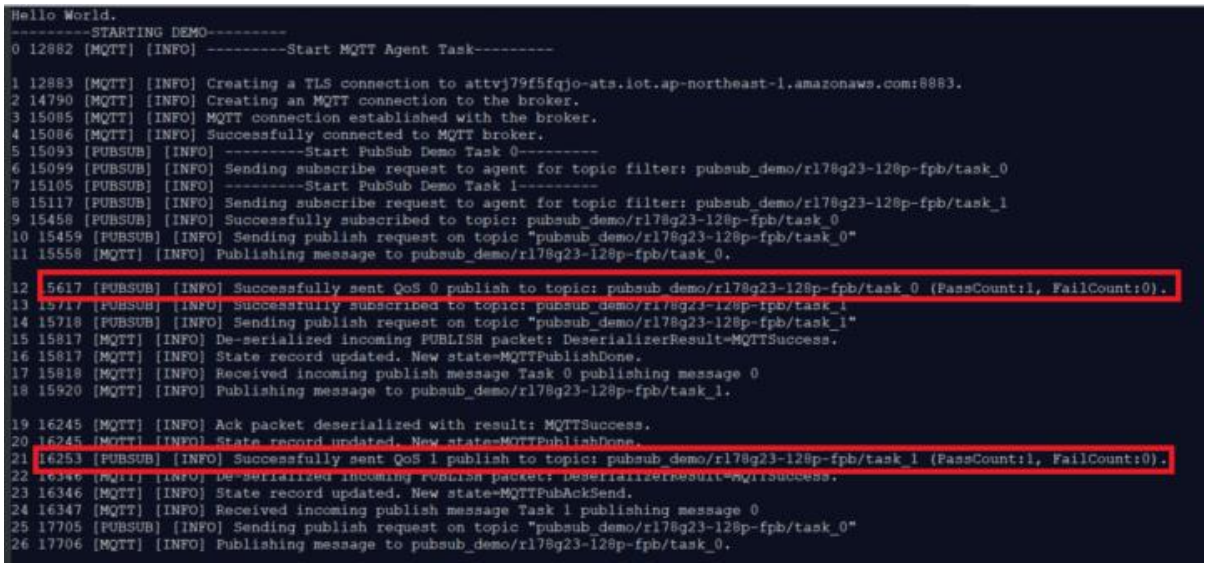


Figure 5-5 Checking Demo Project Execution Results on the Terminal

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

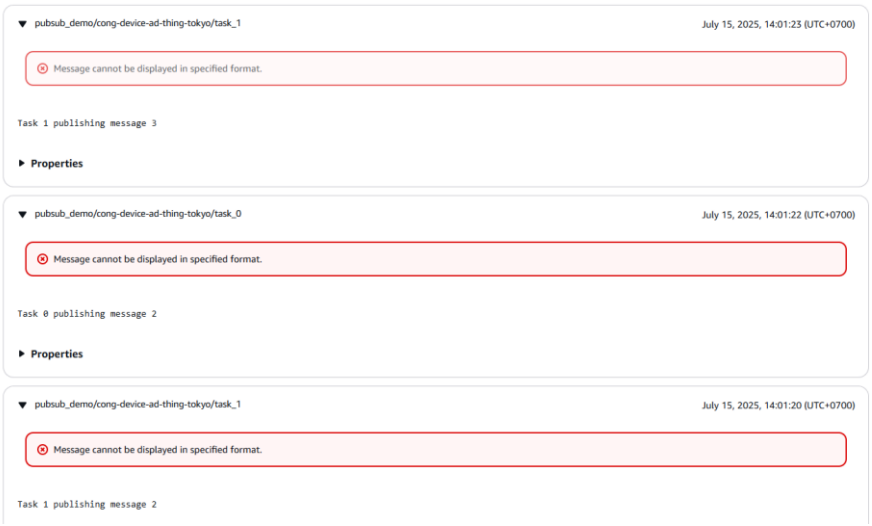


Figure 5-6 Checking Demo Project Execution Results with the MQTT Test Client

5.7 Debugging the Demo Project

The following describes the procedure for starting the demo project (PubSub) from e^2 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)
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, RL78/L23 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 Genarating 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 De-
velopment 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 Select your region in the top right of the screen (e.g. Asia Pacific (Tokyo) ap-northeast-1)

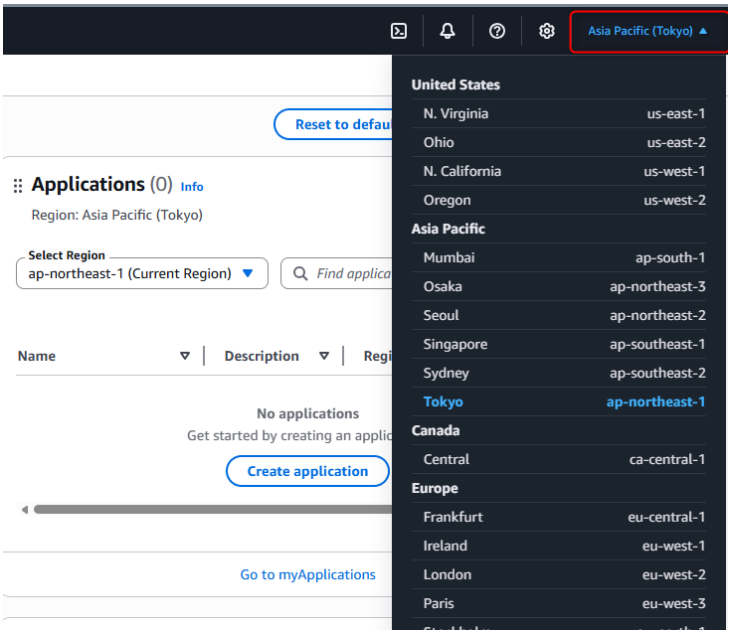


Figure 6-1 Setting Region in AWS

② Amazon S3 > General purpose buckets > “Create bucket”

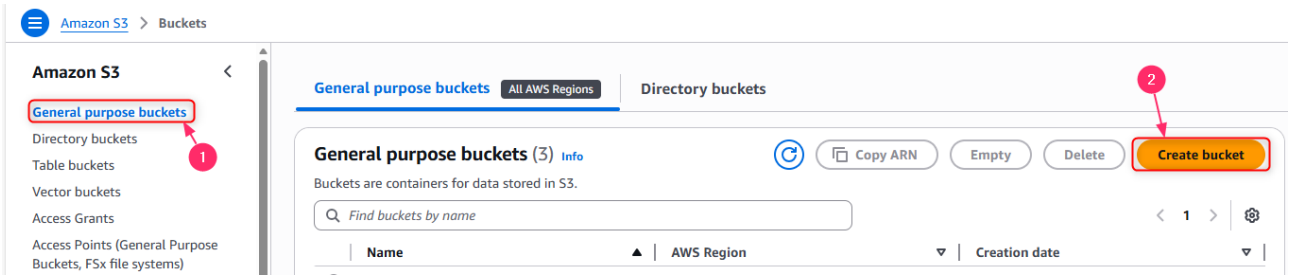


Figure 6-2 Create Bucket

③ General configuration

- Bucket type: General purpose
- Bucket name: Your bucket name

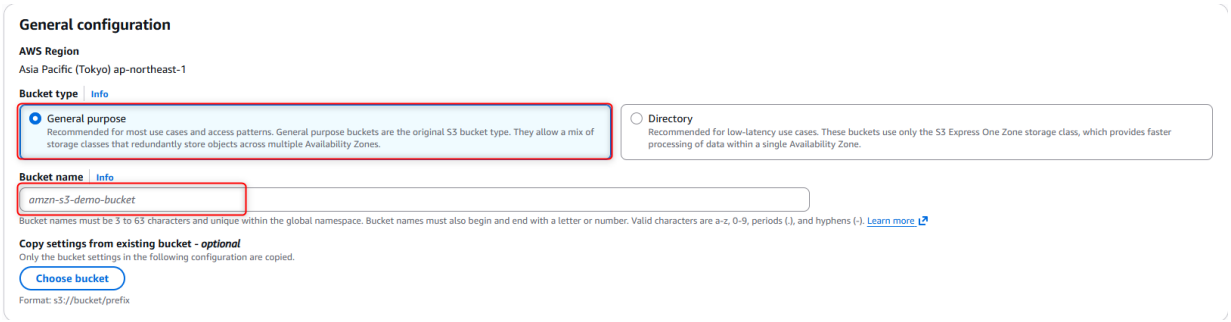


Figure 6-3 General Configuration

④ Object Ownership

- Choose ACLs disabled

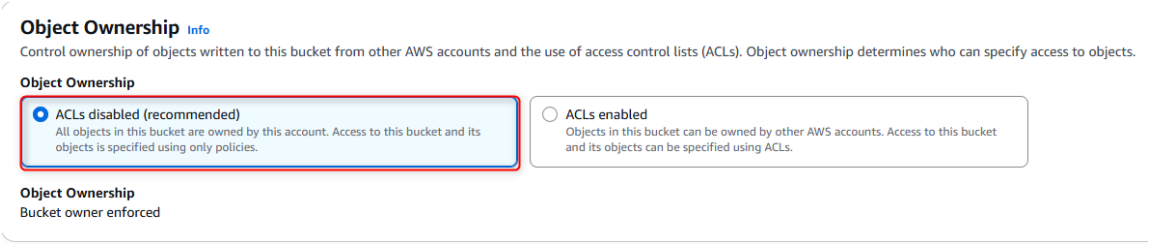


Figure 6-4 Object Ownership

⑤ Object Ownership

- Choose Block all public access

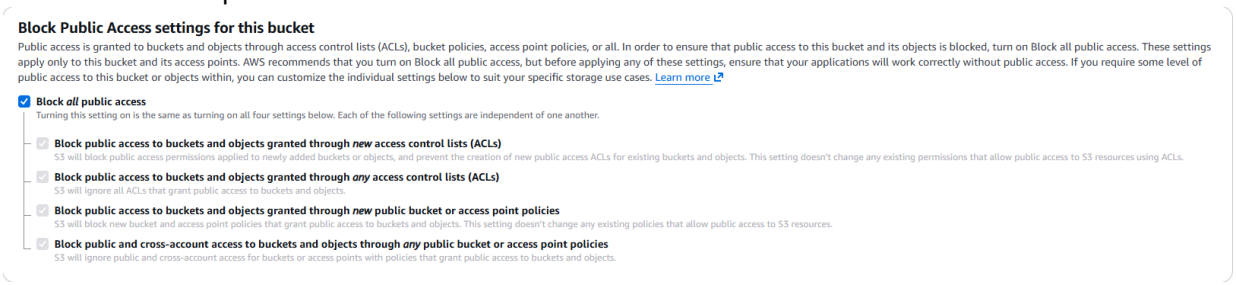


Figure 6-5 Block Public Access Settings for this bucket

⑥ Bucket Versioning

- Bucket Versioning: Enable

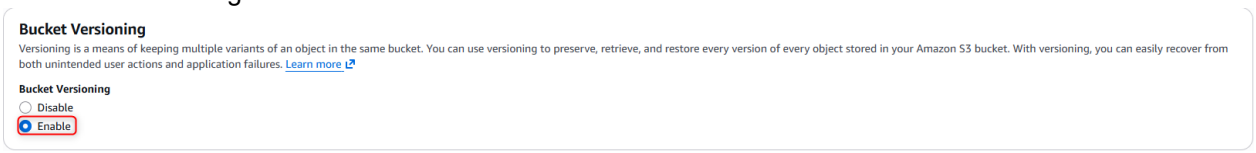


Figure 6-6 Bucket Versioning

⑦ Default encryption

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

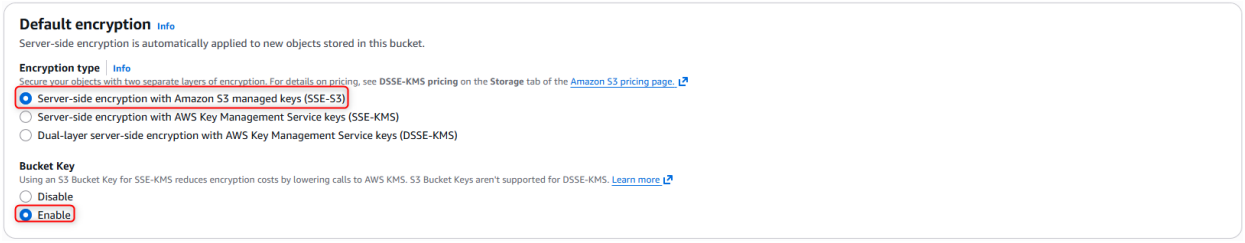


Figure 6-7 Default encryption

⑧ Click “Create bucket”

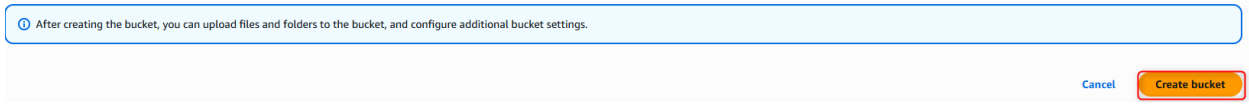


Figure 6-8 Clicking “Create bucket”

6.1.3.2 Creating an OTA Update Service Role

① IAM > Roles > “Create role”

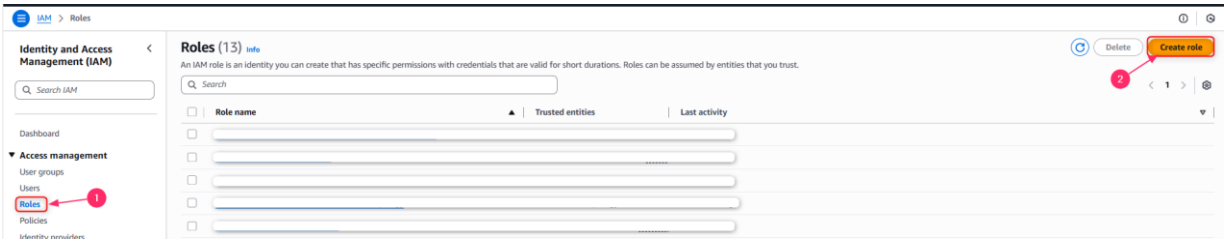


Figure 6-9 IAM > Roles > Create role

② Step 1: Select trusted entity

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

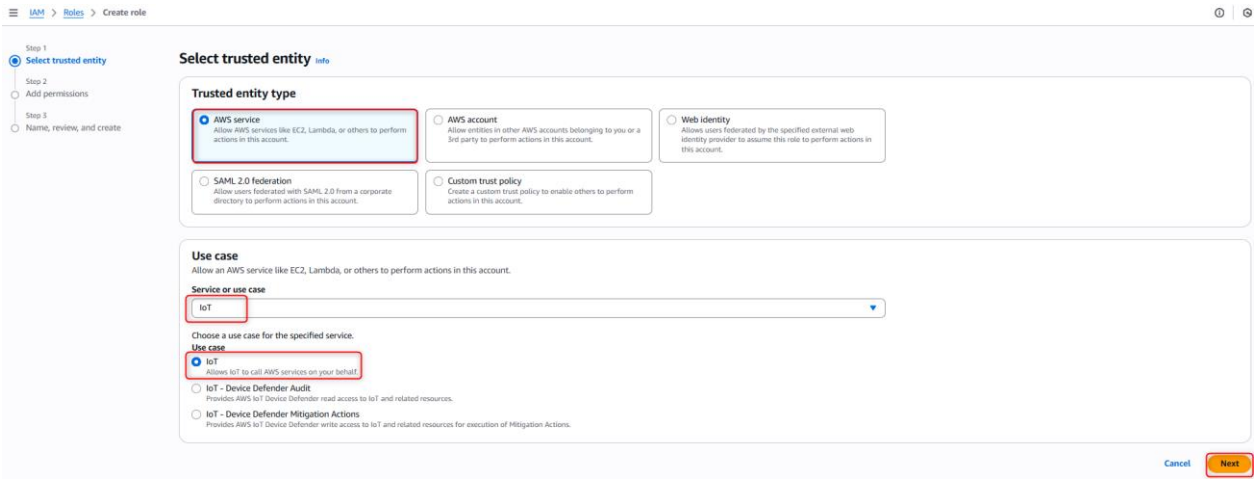


Figure 6-10 Step 1: Select trusted entity

③ Step 2: Add permissions

- AWSIoTLogging
- AWSIoTRuleActions
- AWSIoTThingsRegistration

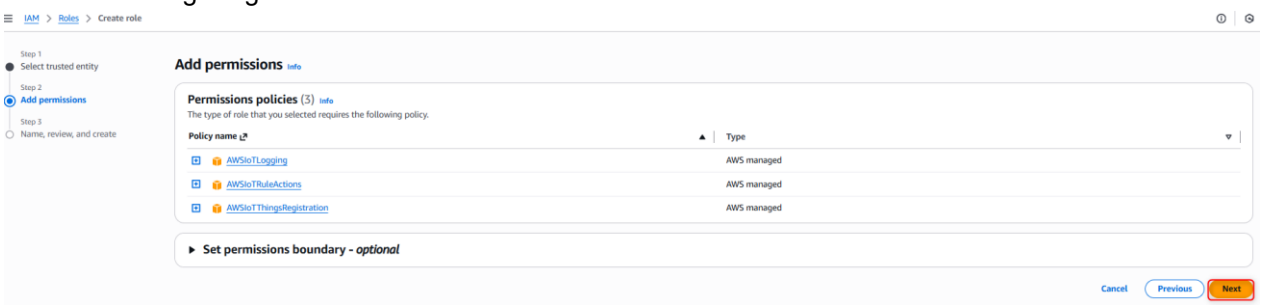


Figure 6-11 Step 2: Add permissions

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

- Role name: Any
- Description: Any

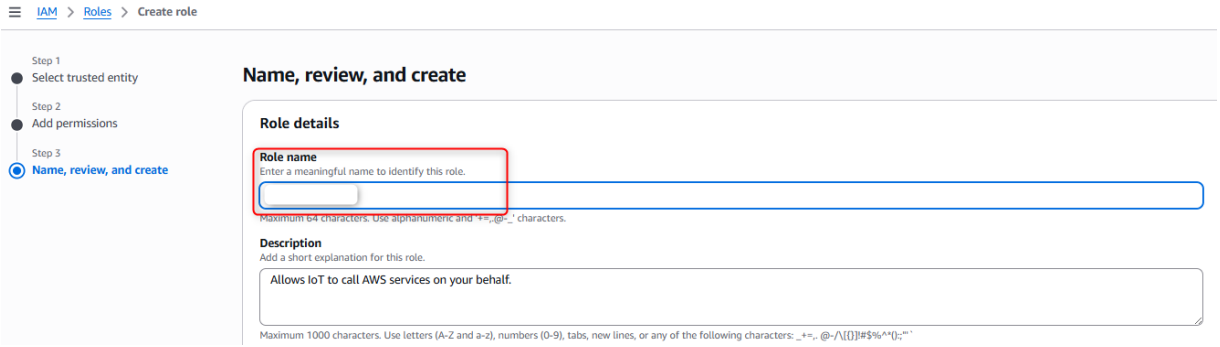


Figure 6-12 Step 3: Name, review, and create > Role details

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

- Default

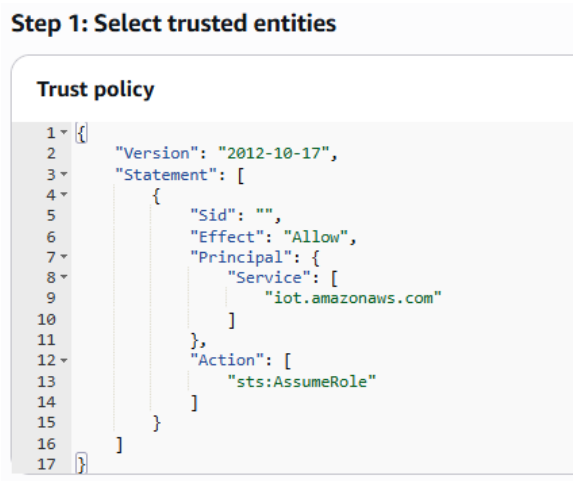


Figure 6-13 Step 3: Name, review, and create > Step 1: Selected trusted entities

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

- Default

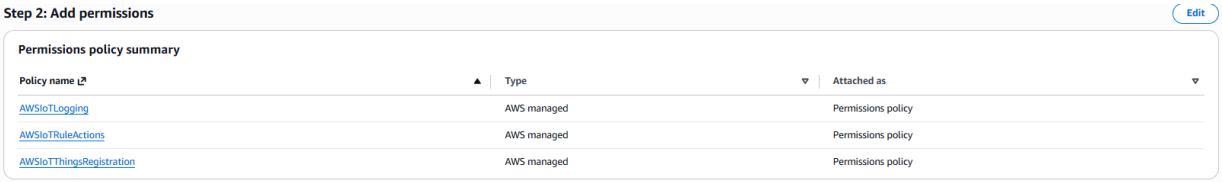


Figure 6-14 Step 3: Name, review, and create > Step 2: Add permissions

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

- Default
- Click “Create role”

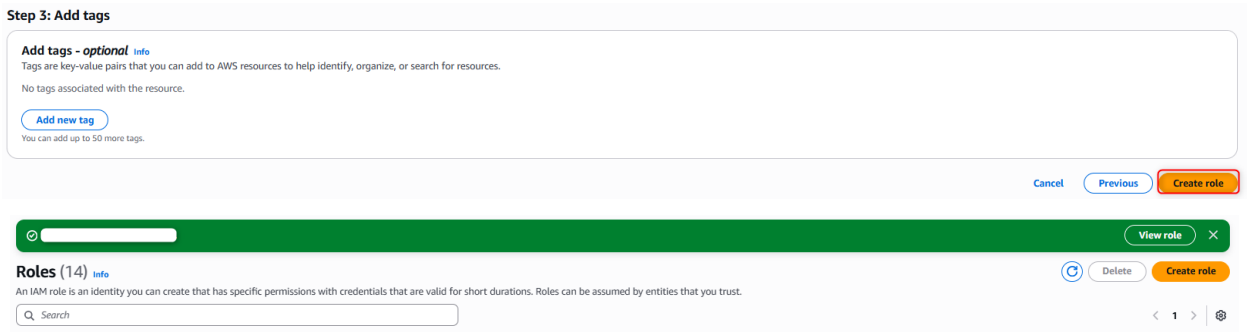


Figure 6-15 Step 3: Name, review, and create > Step 3: Add tags

⑧ Save Role ARN

- Click on your created role
- Copy and save ARN, it will be used for Section 6.1.3.3(5).

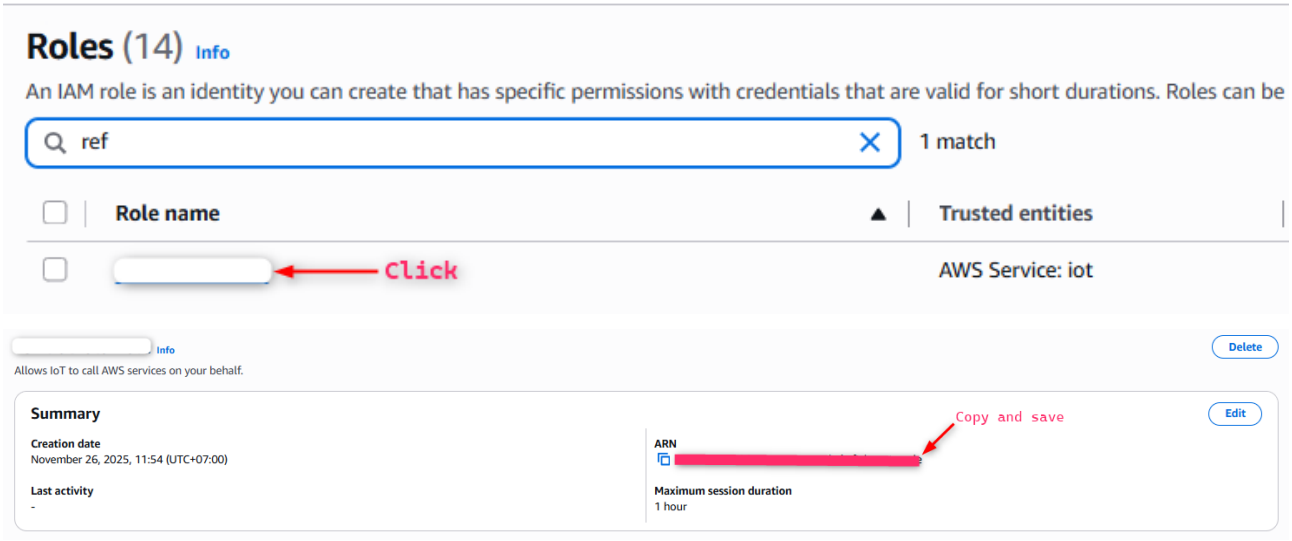


Figure 6-16 Role ARN

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 6-17 Opening the Created OTA Update Service Role

② Attach the policy “AmazonFreeRTOSOTAUpdate”.

- Permissions > “Add permissions” > Attach policies

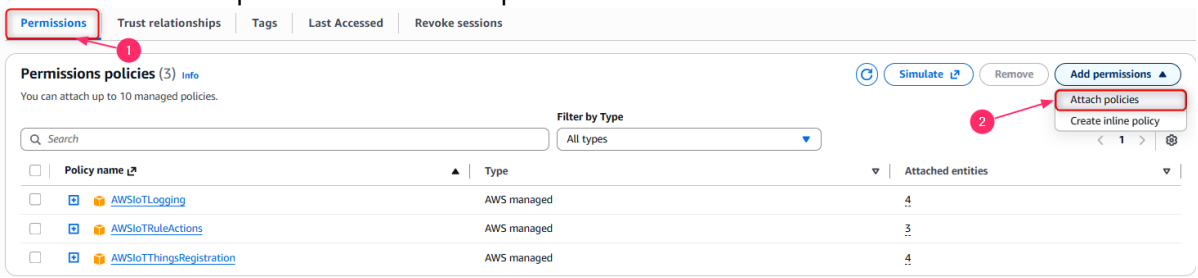


Figure 6-18 Attaching Policies to the Created OTA Update Service Role

- Choose “AmazonFreeRTOSOTAUpdate” > “Add permissions”

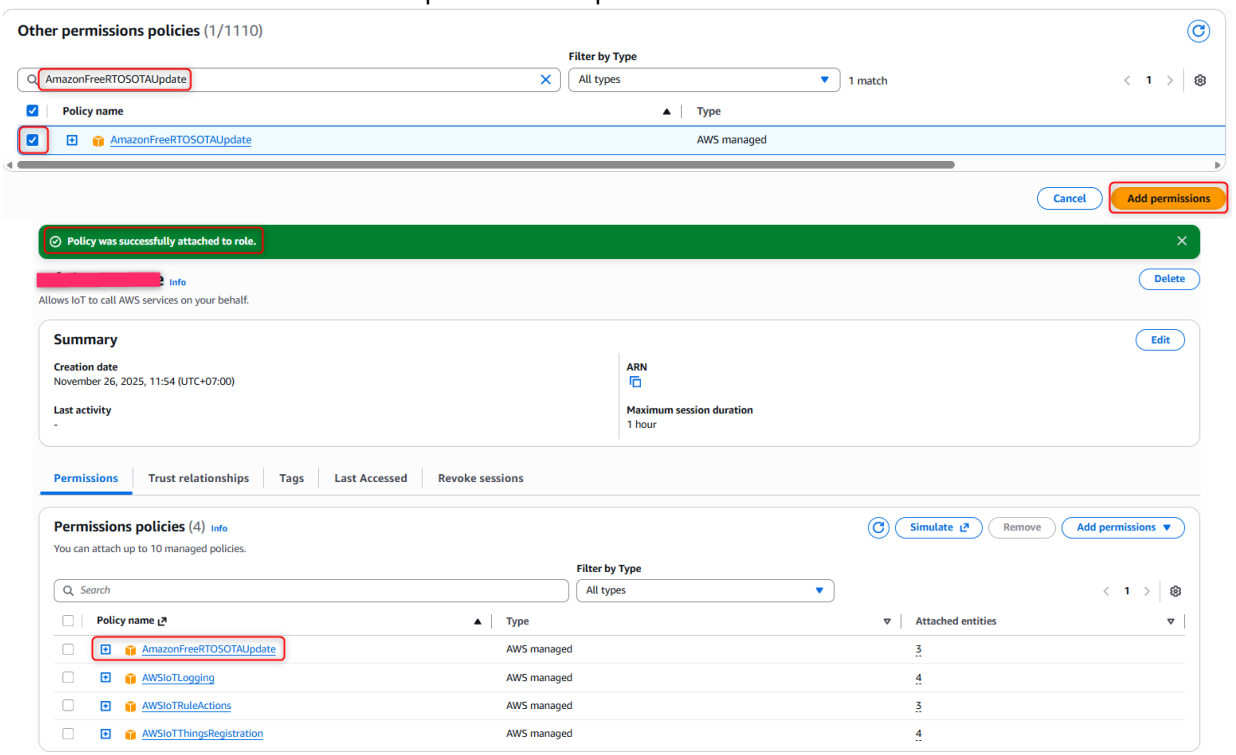


Figure 6-19 Attaching the Policy “AmazonFreeRTOSOTAUpdate” to the Created OTA Update Service Role

③ Add the inline policy (S3).

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

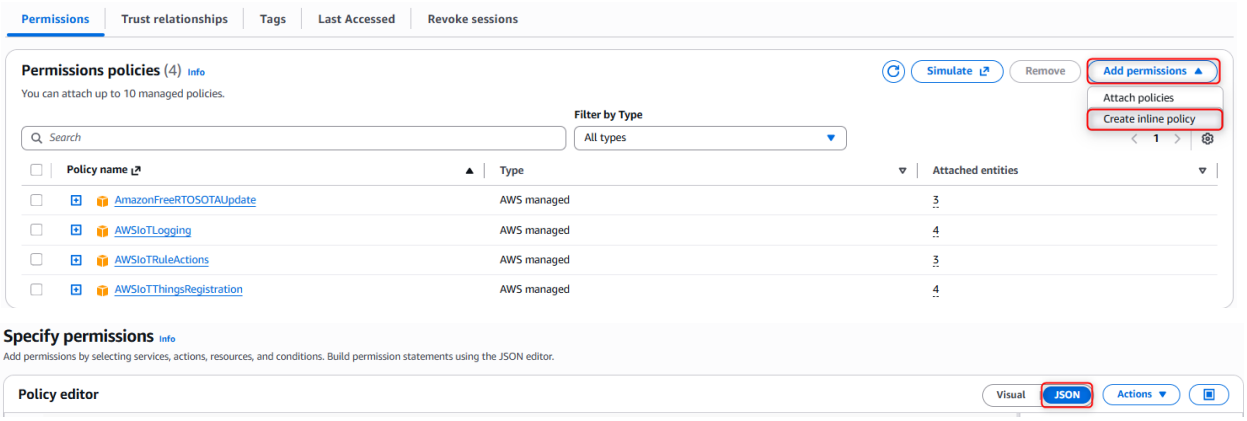


Figure 6-20 Creating an S3 Inline Policy

- Paste the following information to the Policy editor, and then click “Next”.
  Change Your-bucket-name 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:::Your-bucket-name/*"
        }
    ]
}
```

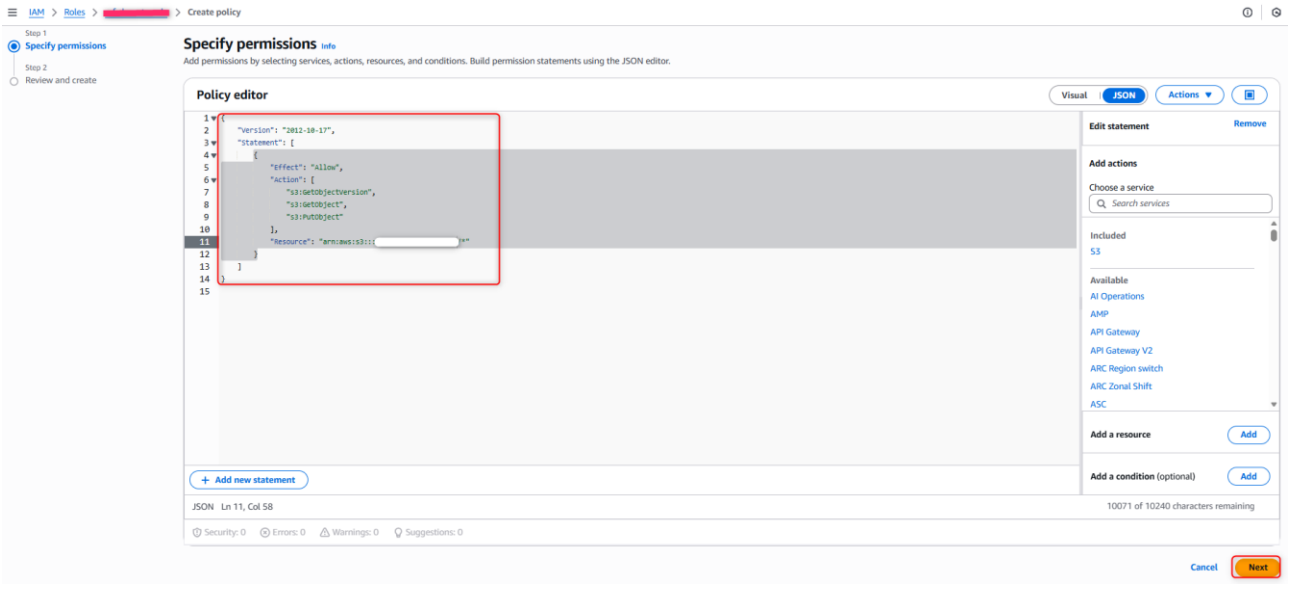


Figure 6-21 Adding S3 Policies to the Policy Editor

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

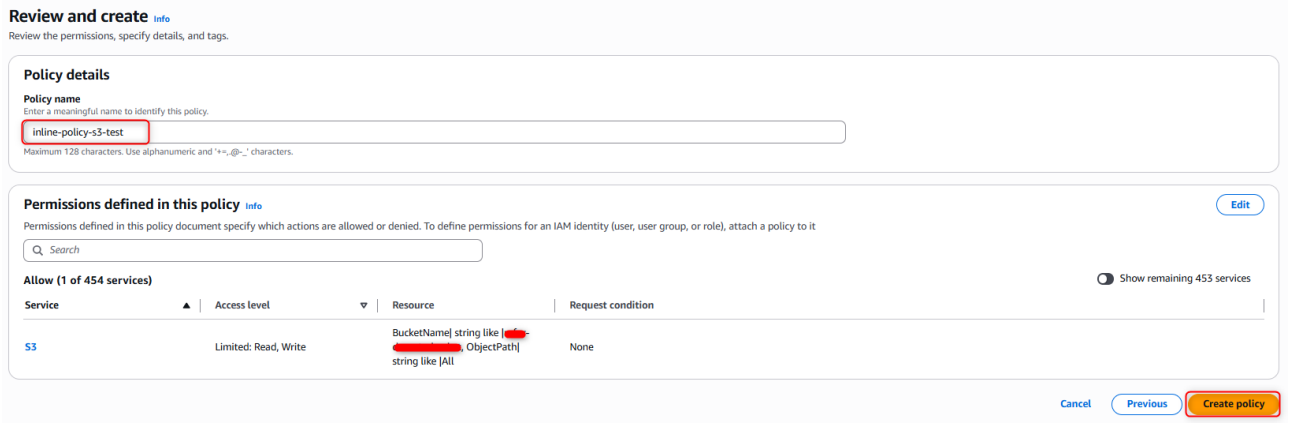


Figure 6-22 Creating the S3 Policy with a Name (Example: inline-policy-s3-test)

④ Add an IAM inline policy.

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

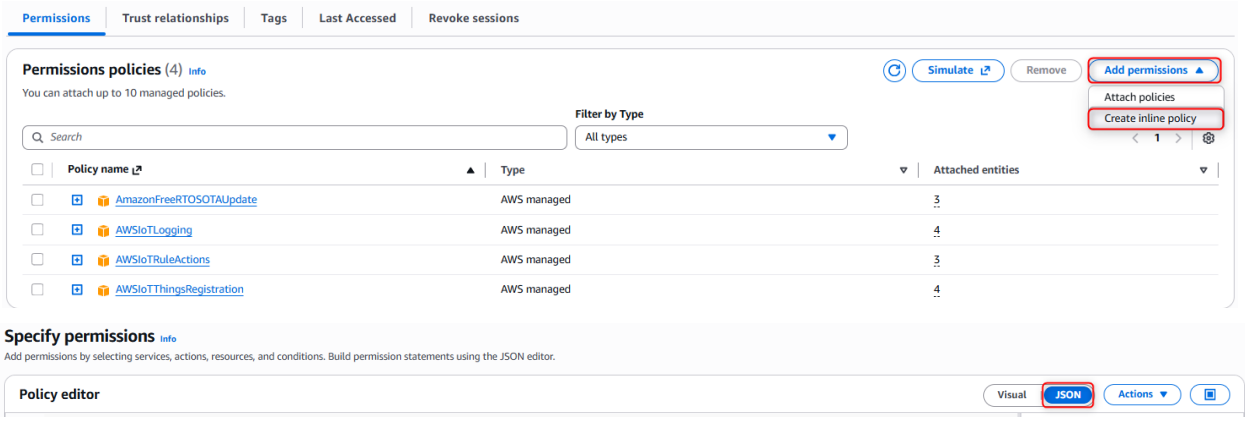


Figure 6-23 Creating an Inline Policy

⑤ Paste the following information to the Policy editor, and then click “Next”.

- Change ota-role-arn to the role name created in section 6.1.3.2(8), Save Role ARN.

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

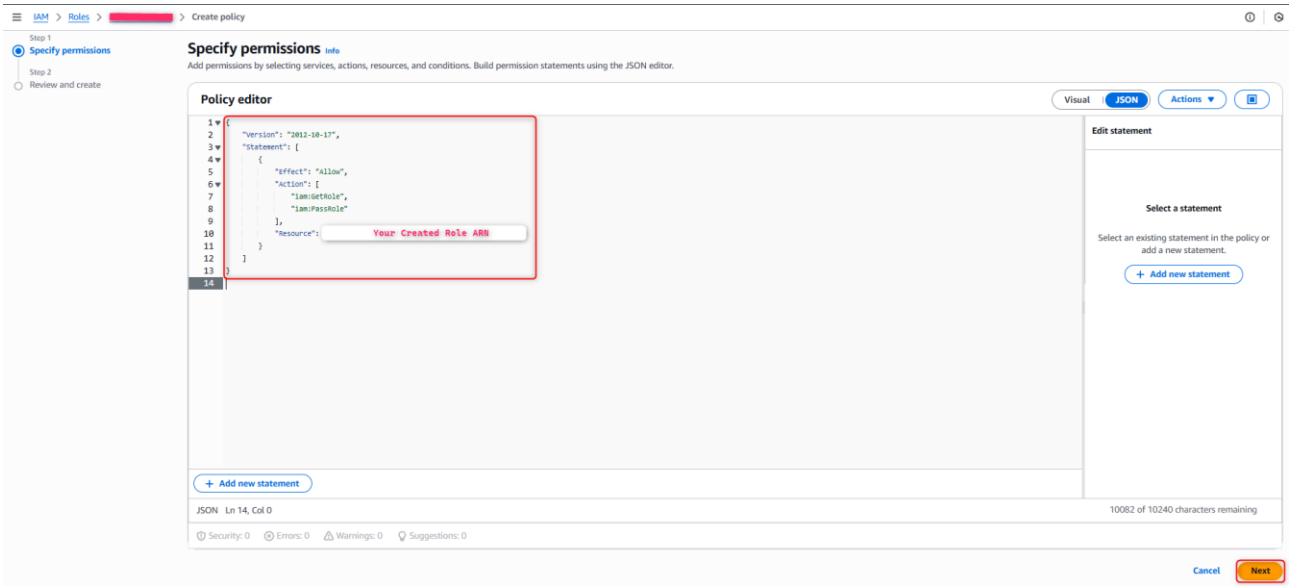


Figure 6-24 Adding the IAM Role to the Inline Policy

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

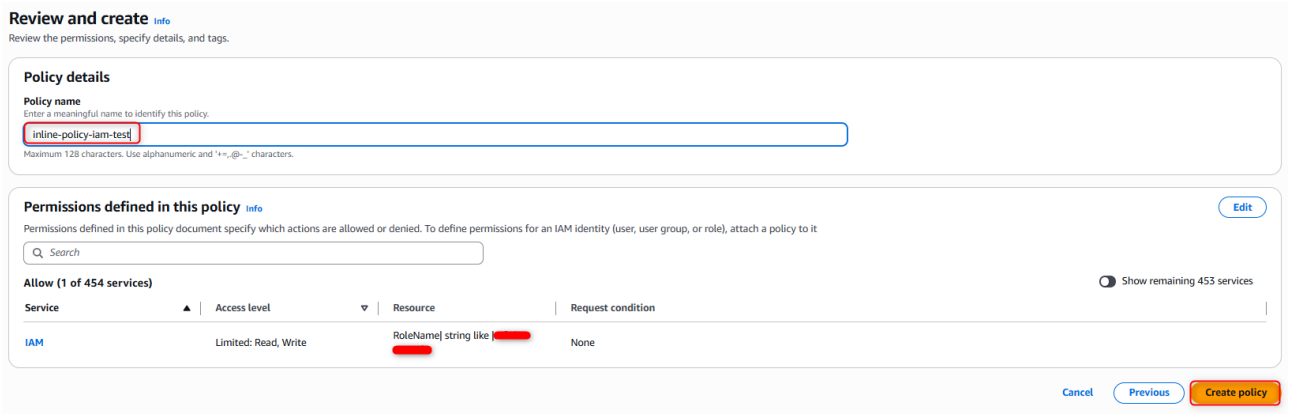


Figure 6-25 Saving the IAM Inline Policy with a Name (Example: inline-policy-iam-test)

6.1.3.4 Allocating an OTA Update Policy to IAM User

① Create an OTA Update policy.

- IAM > Policies > “Create policy” -> “JSON”

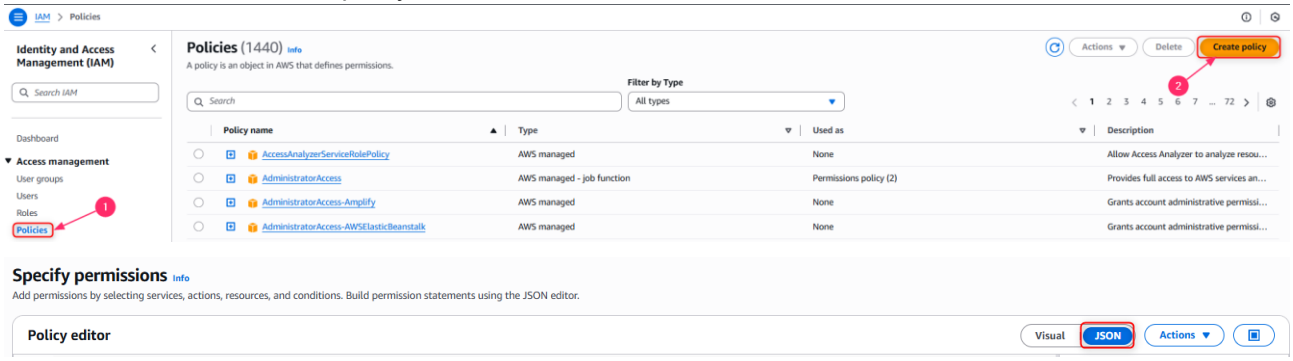


Figure 6-26 Create an OTA Update Policy

② Paste the following information to the Policy editor, and then click “Next”.

- Change Your-bucket-name to the bucket name created in section 6.1.3.1, Creating Amazon S3 Buckets
- Change ota-role-arn to the role name created in section 6.1.3.2(8), Save Role ARN.

```
{
    "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:::Your-bucket-name/*"
        },
        {
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": "ota-role-arn"
        }
    ]
}
```

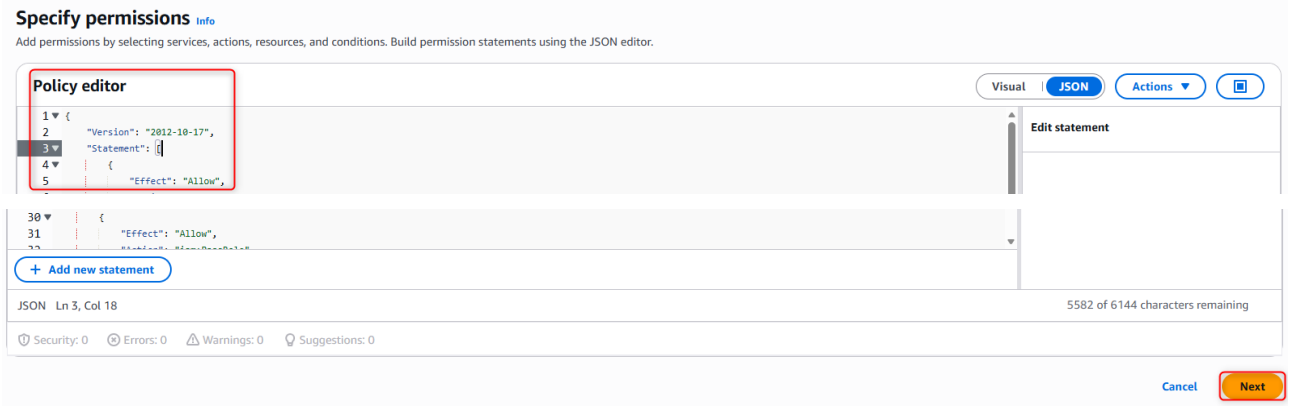


Figure 6-27 Creating an OTA Update Policy with the Policy Editor

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

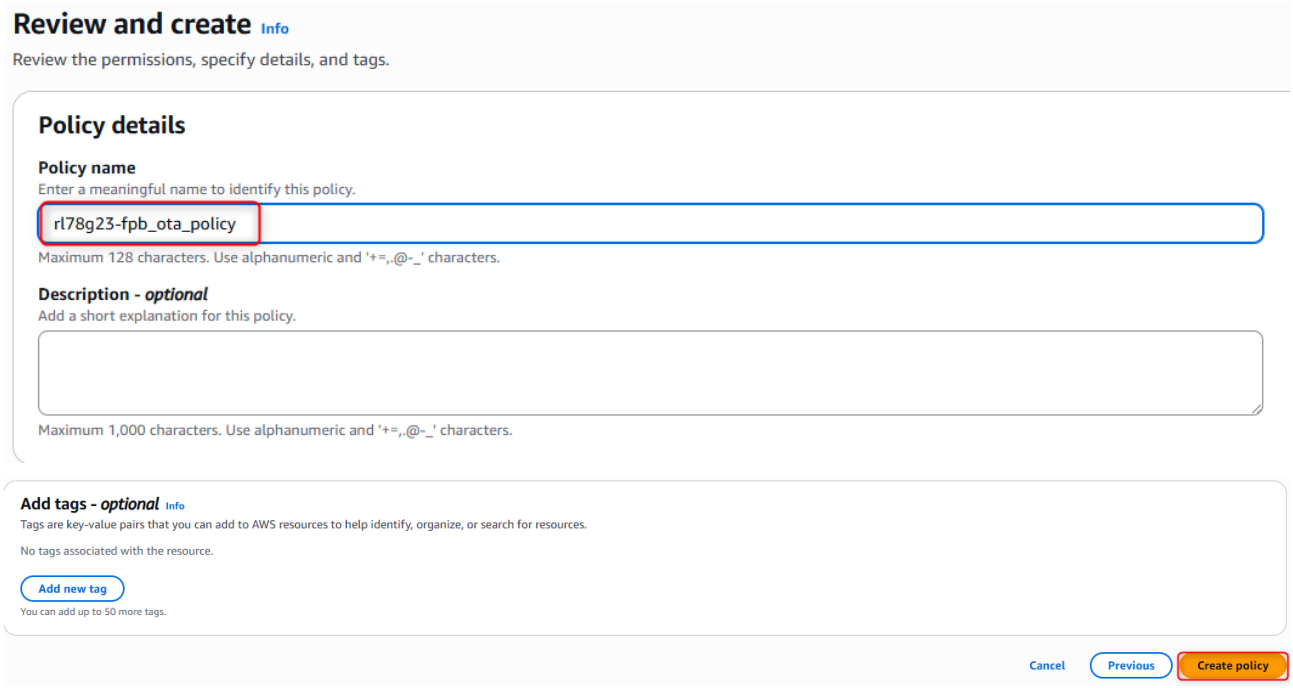


Figure 6-28 Saving the OTA Update Policy with a Name (Example: rl78g23-fpb_ota_policy)

③ Add the created OTA Update policy to the IAM user.

- IAM > Users > Choose User > Add permissions

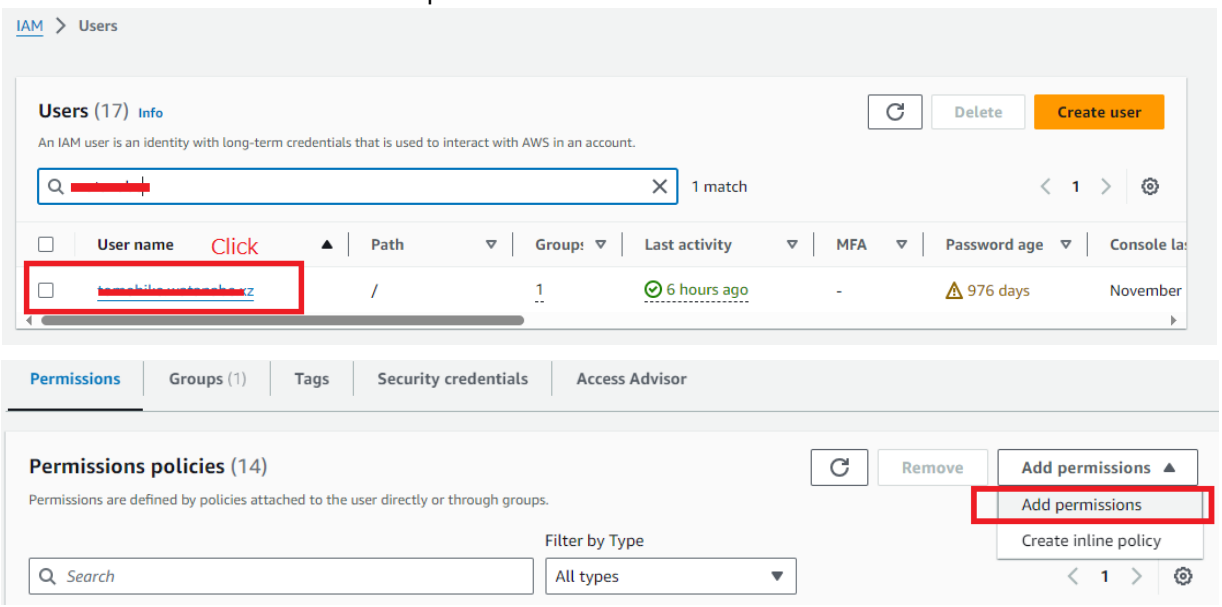


Figure 6-29 Selecting the IAM User

- Permissions options: Attach policies directly
- Permissions policies > Policy name: Name of created OTA Update policy (Example: rl78g23-fpb_ota_policy)
- Click “Next”

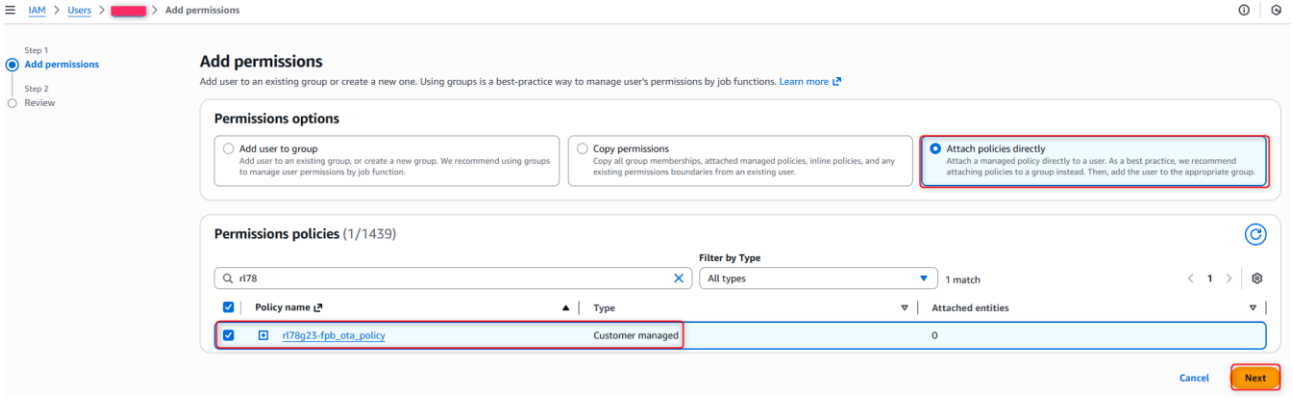


Figure 6-30 Selecting the OTA Update Policy for Permissions to Be Added to the IAM User

- User details: Your account
- Permissions summary > Name: Name of created OTA Update policy (Example: rl78g23-fpb_ota_policy)

Click “Add permission”

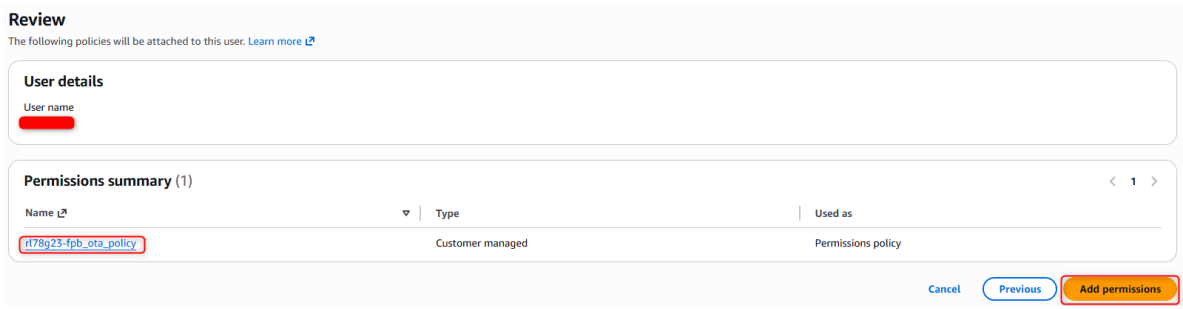


Figure 6-31 Adding the OTA Update Policy to the Selected IAM User

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, RL78/L23 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_da16600_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 e^2 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.

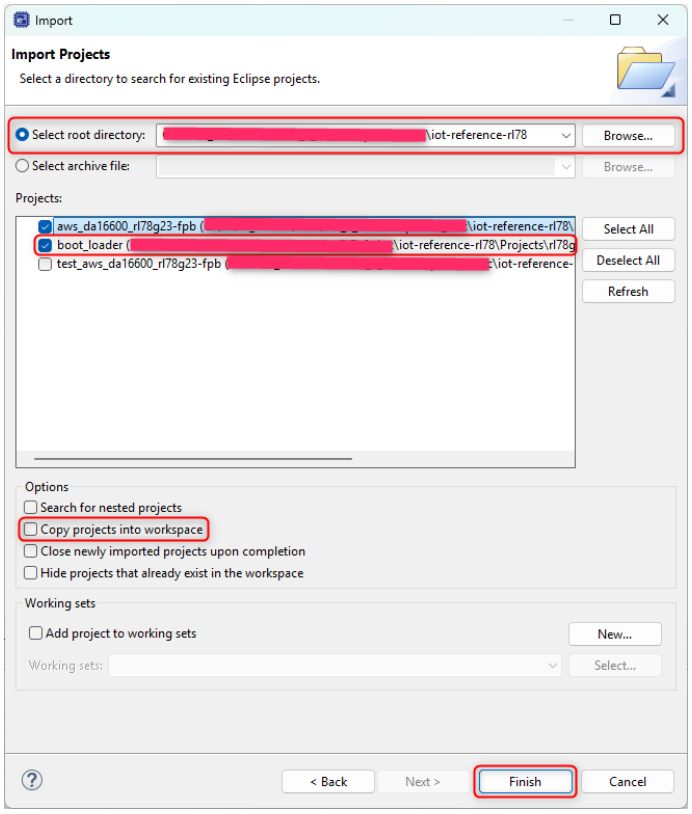


Figure 6-32 Selecting the boot_loader Project

The imported project is shown in the Project Explorer view.

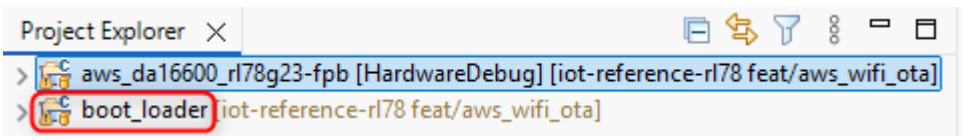


Figure 6-33 Completing to Import the boot_loader Project

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: You must enclose each line in quotation marks (“) and end the line with the ‘\’ symbol.
Example: “xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx”\

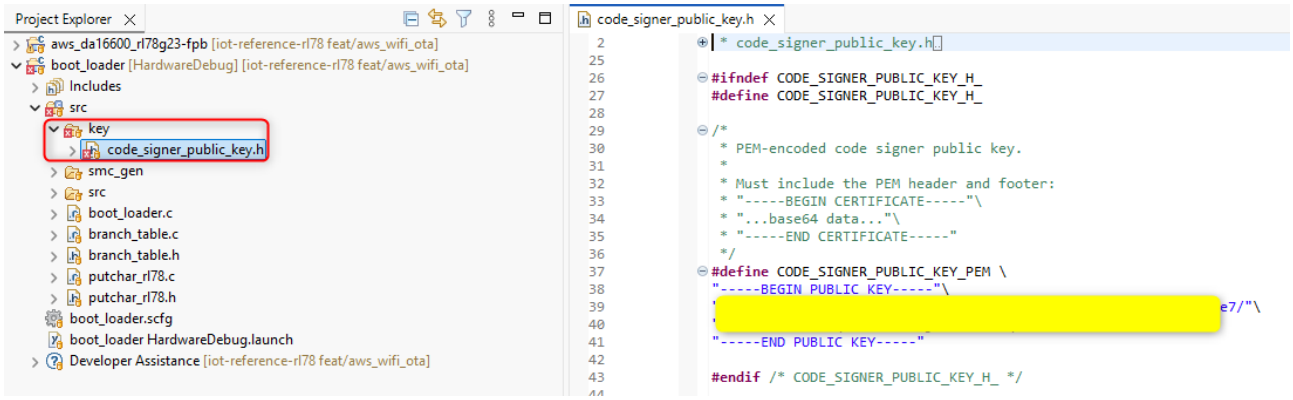


Figure 6-34 Adding the Firmware Verification Key to the Bootloader

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_da16600_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_da16600_rl78g23-fpb project. Ensure that copy projects into workspace is not selected. Then click the Finish button.

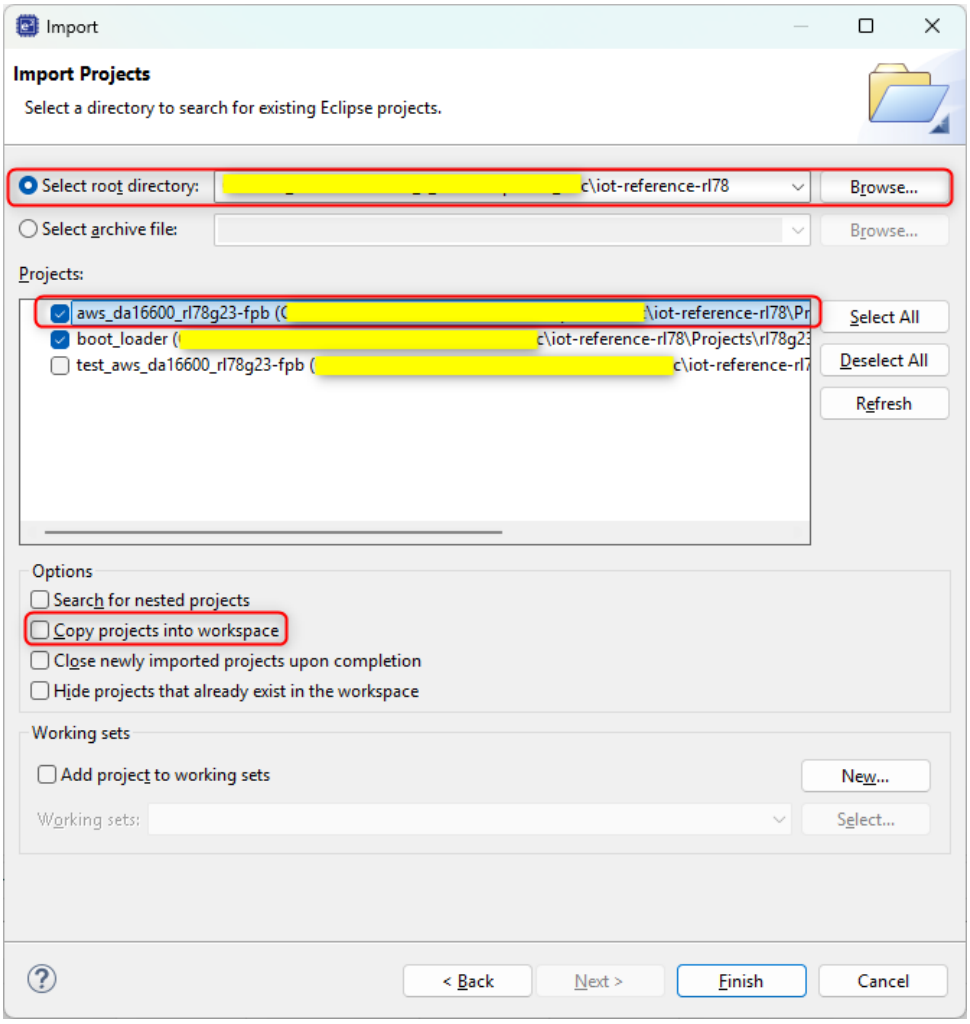


Figure 6-35 Selecting the aws_da16600_rl78g23-fpb Project

The imported project is showed in the Project Explorer view.

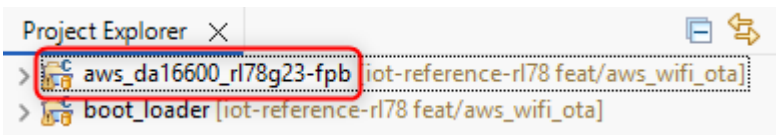


Figure 6-36 Completing to Import the aws_da16600_rl78g23-fpb Project

6.2.2.2 Setting the Build Configuration of the Initial Application

Set the build configuration of the aws_da16600_rl78g23-fpb project to “HardwareDebug_OTA”.

Build Configurations > Set Active > Select “HardwareDebug_OTA”

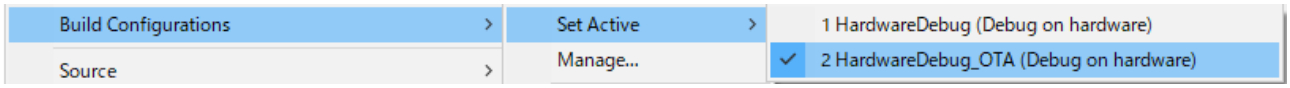


Figure 6-37 Activating Build Configuration “HardwareDebug_OTA”

6.2.2.3 Adding the Firmware Verification Key to the Initial Application

Add the firmware verification key (secp256r1.publickey) into CODE_SIGNER_PUBLIC_KEY_PEM in \aws_da16600_rl78g23-fpb\src\key\code_signer_public_key.h.

Note: You must enclose each line in quotation marks (“) and end the line with the ‘\’ symbol.
Example: “xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx”\

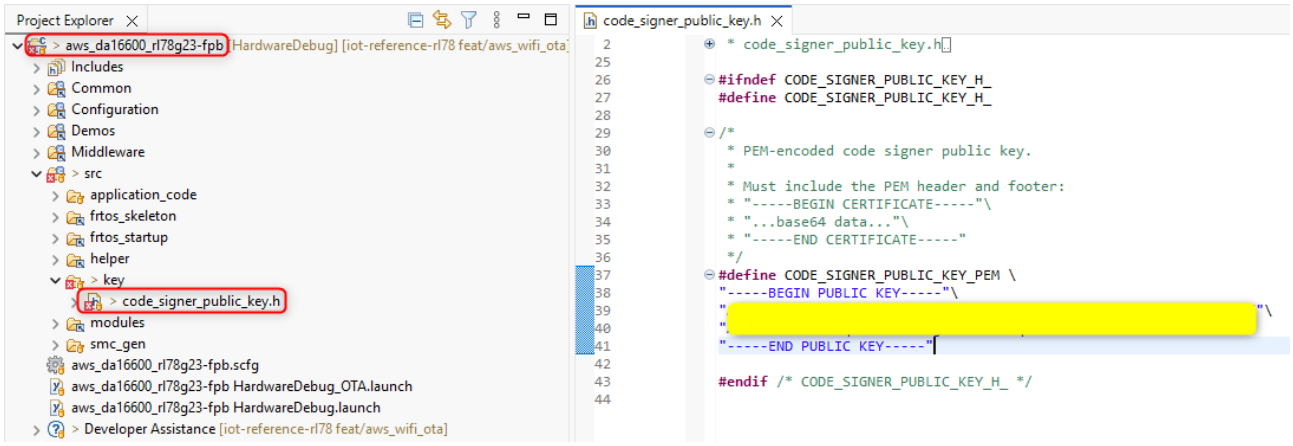


Figure 6-38 Adding the Firmware Verification Key to the Initial Application

6.2.2.4 Building the Initial Application

Build the aws_da16600_rl78g23-fpb project to create a MOT file.
Then, make sure that aws_da16600_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

This section is used to create the initial image (**Initial_image.mot)**, which is executed on the MCU before the Over-The-Air process begins. Follow the setup steps below to generate the initial firmware correctly:

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

- The results of building process in Section 6.2.1: boot_loader.mot
- The results of building process in Section 6.2.2: aws_da16600_rl78g23-fpb.mot
- The private key created in Section 6.1.2: secp256r1.privatekey

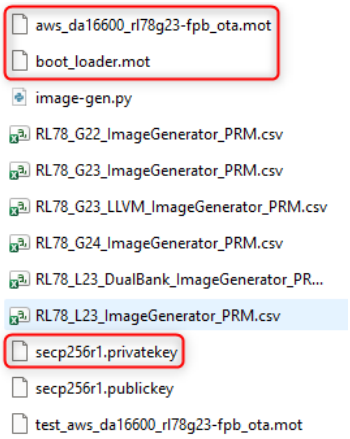


Figure 6-39 Storing Necessary Files in the Same Folder as Renesas Image Generator

② Run the following command to generate the initial image.

```
python image-gen.py -iup .\aws_da16600_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.

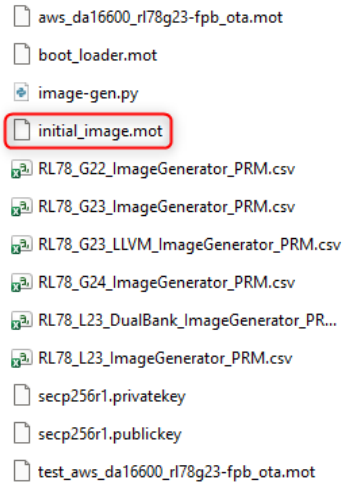


Figure 6-40 Initial Image Generated in the Same Folder as Renesas Image Generator

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, RL78/L23 Firmware Update Module".

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

- Update application: aws_da16600_rl78g23-fpb_ota_093.mot
- Update image: aws_da16600_rl78g23-fpb_ota_093.rsu

6.3.1 Creating an Update Application

6.3.1.1 Changing the Source Code of the Application

Change the firmware version to a higher version. (Example: Because previous version is 0.9.2, so the new version we can choose 0.9.3)

Repeat the build process in Section 6.2.2, this time '3' specified for the APP_VERSION_BUILD definition in \aws_da16600_rl78g23-fpb\Configuration\rl78g23-fpb\ota\wifi\frtos_config\demo_config.h

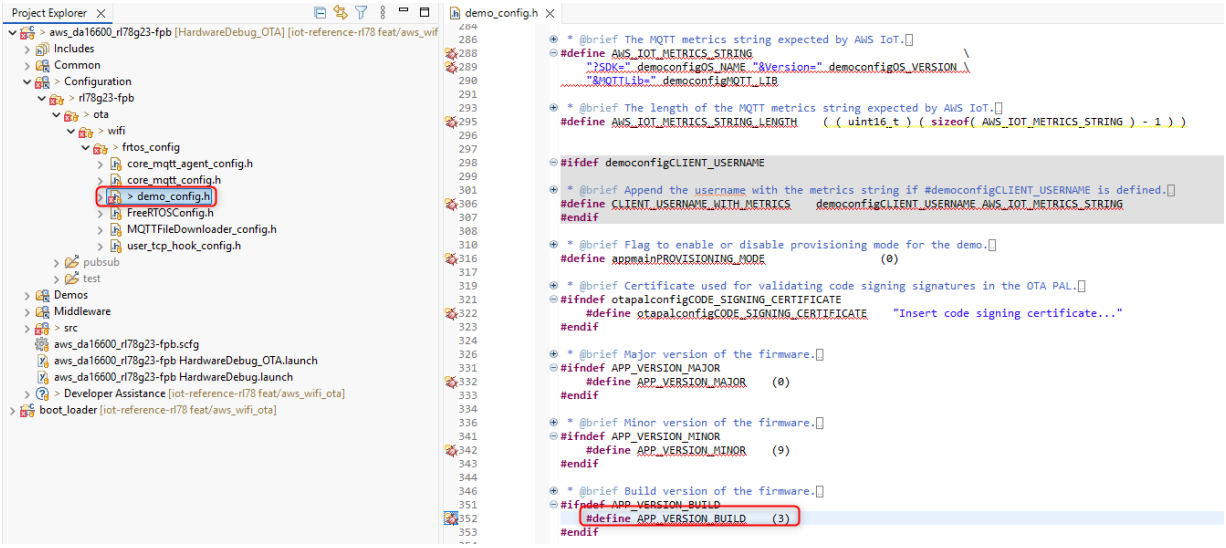


Figure 6-41 Setting New Version for Firmware

6.3.1.2 Renaming the MOT File of the Update Application

Rename aws_da16600_rl78g23-fpb_ota.mot to aws_da16600_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_da16600_rl78g23-fpb_ota_093.mot
- The private key created in Section 6.1.2: secp256r1.privatekey

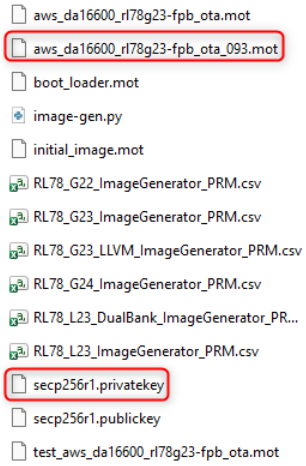


Figure 6-42 Storing Necessary Files in the Same Folder as Renesas Image Generator

② Run the following command to generate an update image (aws_da16600_rl78g23-fpb_ota_093.rsu) in RSU format.

```
python image-gen.py -iup .\aws_da16600_rl78g23-fpb_ota_093.mot -o
aws_da16600_rl78g23-fpb_ota_093 -ip .\RL78_G23_ImageGenerator_PRM.csv -vt
ecdsa -ff RTOS
```

③ Make sure that aws_da16600_rl78g23-fpb_ota_093.rsu has been generated.

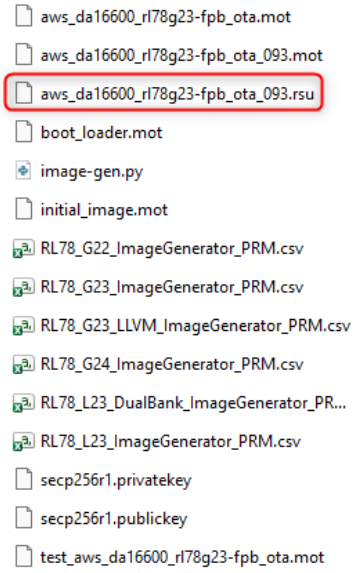


Figure 6-43 Update Image Generated in the Same Folder as Renesas Image Generator

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.

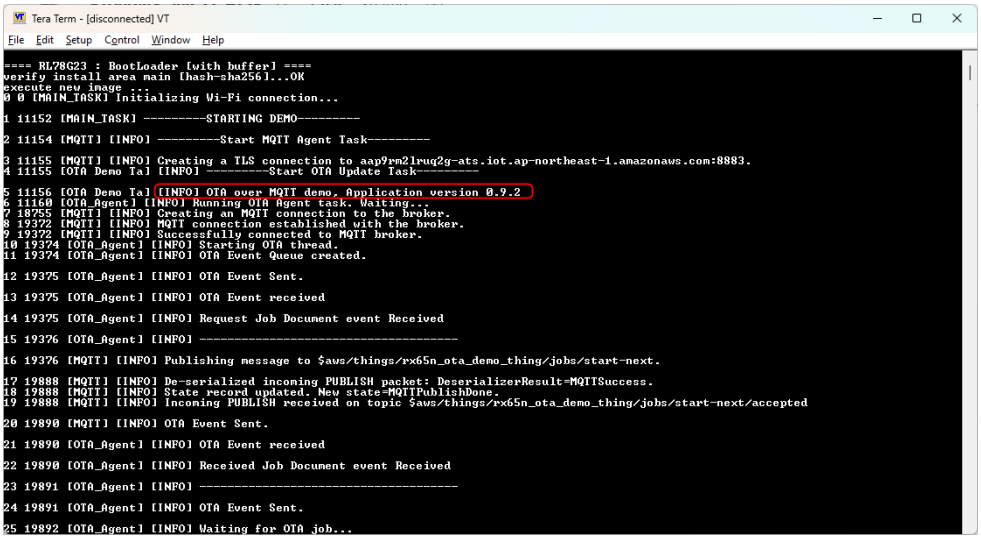


Figure 6-44 Initial Application (Version 0.9.2) Started

6.4.2 Registering the Update Image (aws_da16600_rl78g23-fpb_ota_093.rsu) with OTA Jobs

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

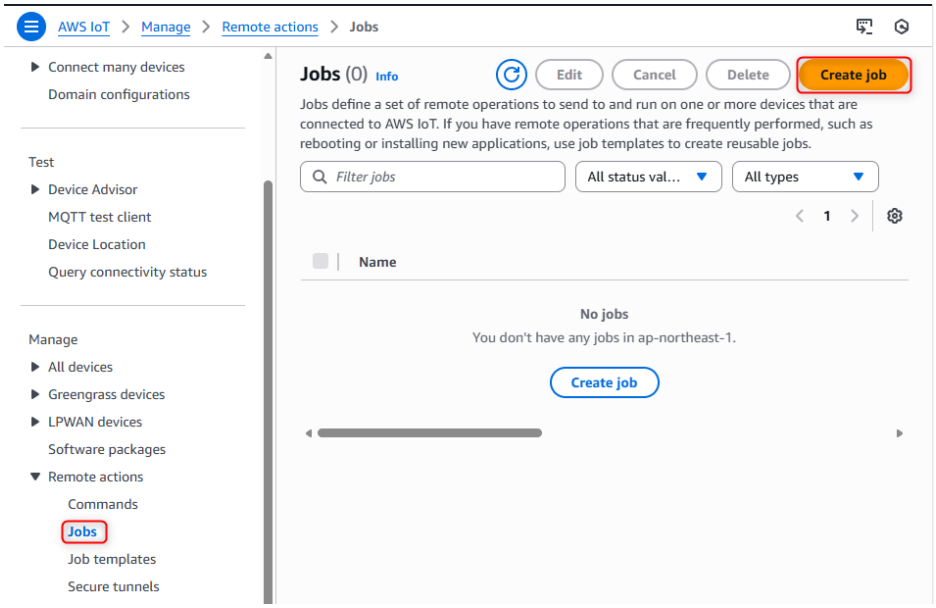


Figure 6-45 Jobs

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

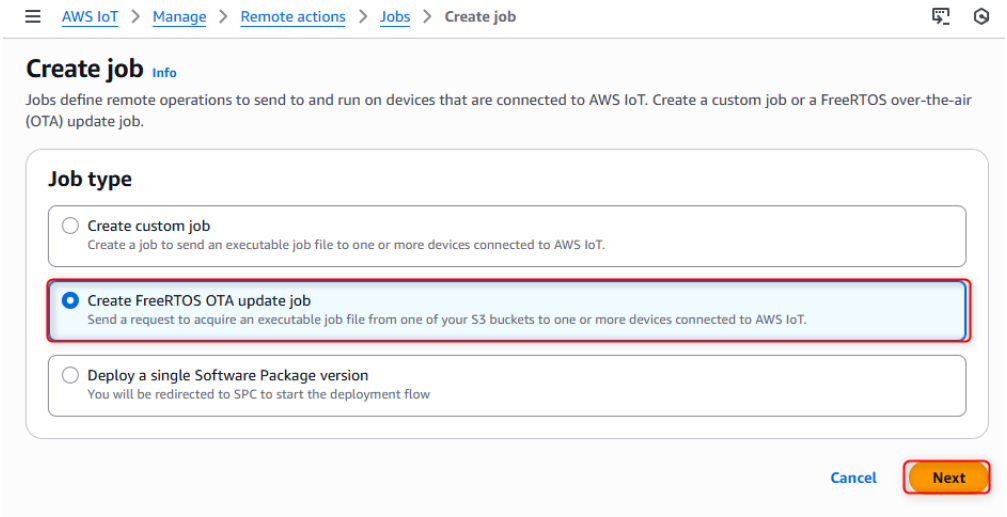


Figure 6-46 Create Jobs

③ Step 1: OTA job properties

- Job name: Any

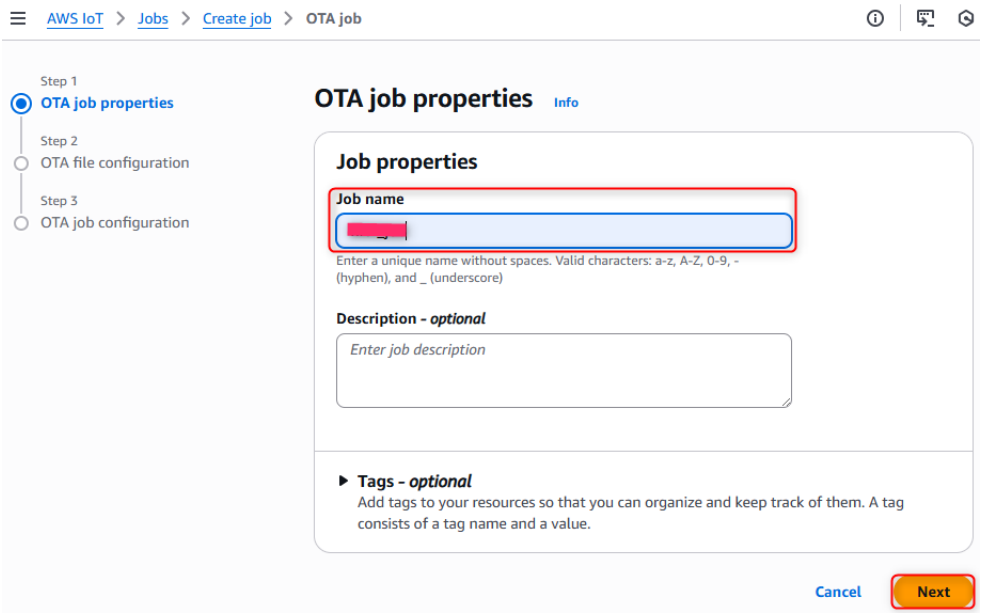


Figure 6-47 Step 1: OTA job properties

④ Step 2: OTA file configuration > Devices

- Devices to update: “Name of the thing” in \aws_da16600_rl78g23-fpb\Demos\include\aws_clientcredential.h

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

- Select the protocol for file transfer: MQTT

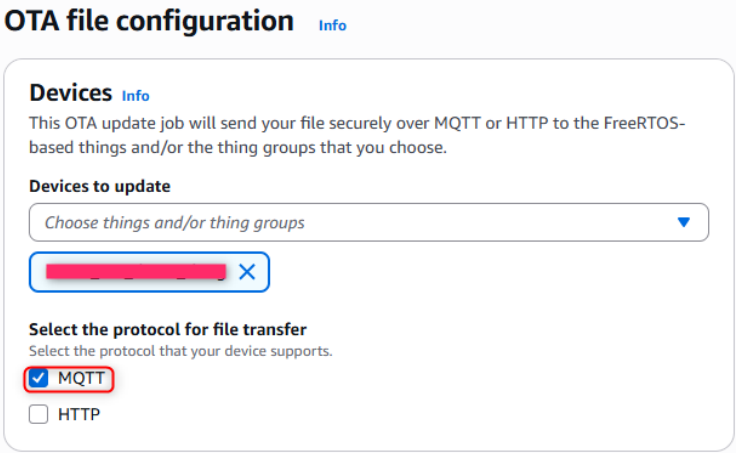


Figure 6-48 Step 2: OTA file configuration > Devices

⑤ Step 2: OTA file configurations > File

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

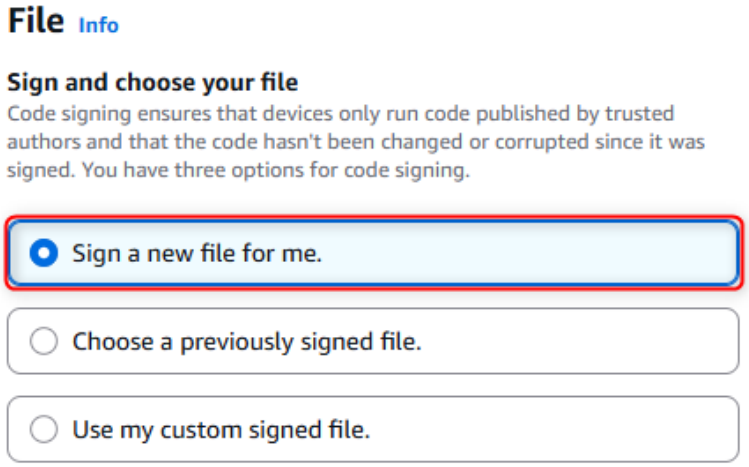


Figure 6-49 Step 2: OTA file configurations > File (1)

- Code signing profile: Click “Create new profile”

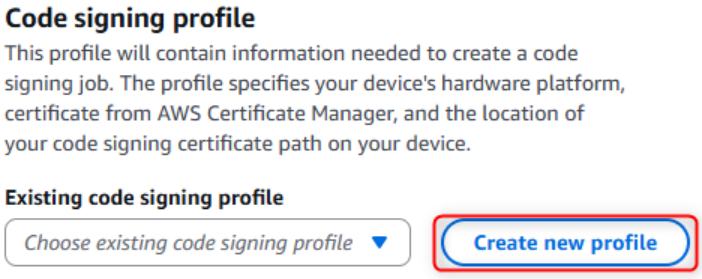


Figure 6-50 Step 2: OTA file configurations > File (2)

- 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

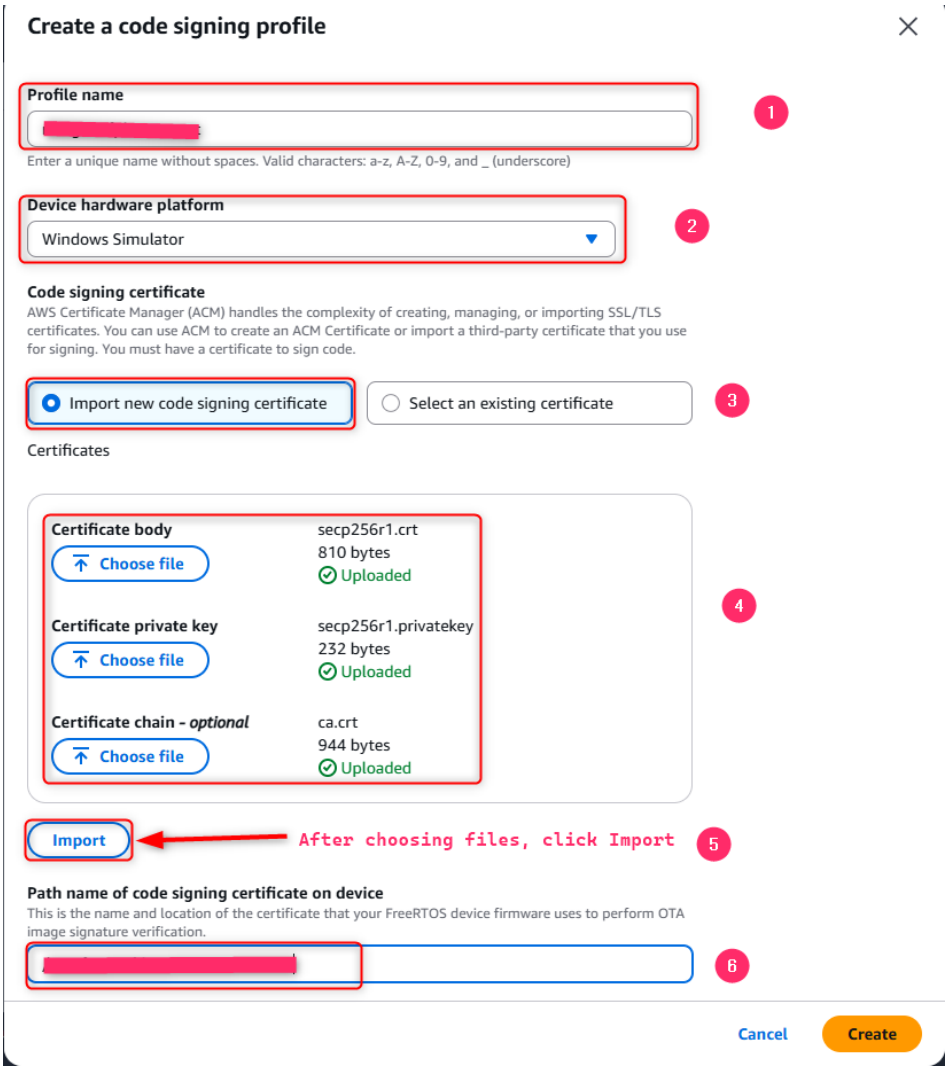


Figure 6-51 Create a code signing profile

- File > “Upload a new file.” > “Choose file” > aws_da16600_rl78g23-fpb_ota_093.rsu

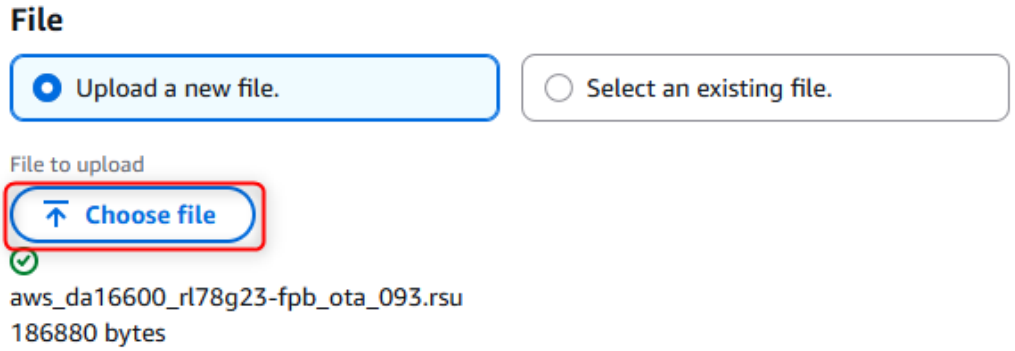


Figure 6-52 Upload a new file > aws_da16600_rl78g23-fpb_ota_093.rsu

- 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

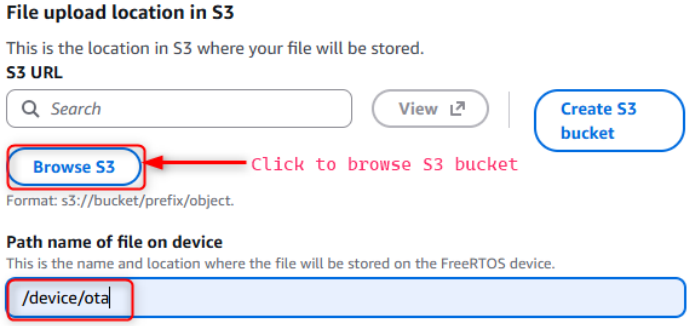


Figure 6-53 File upload location in S3 (1)

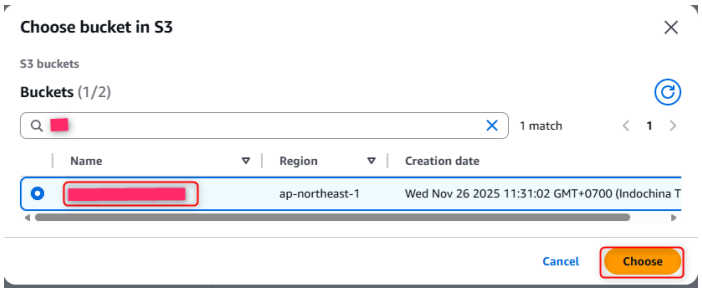


Figure 6-54 File upload location in S3 (2)

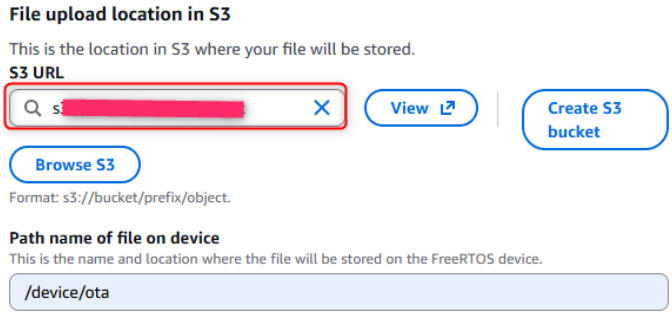


Figure 6-55 File upload location in S3 (3)

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).

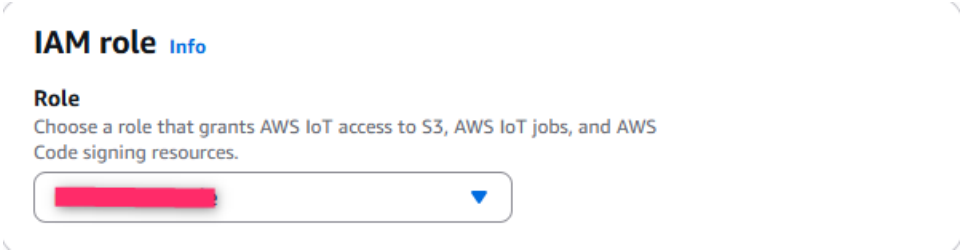


Figure 6-56 Step 2: OTA file configurations > IAM role

⑦ Step 3: OTA job configuration

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

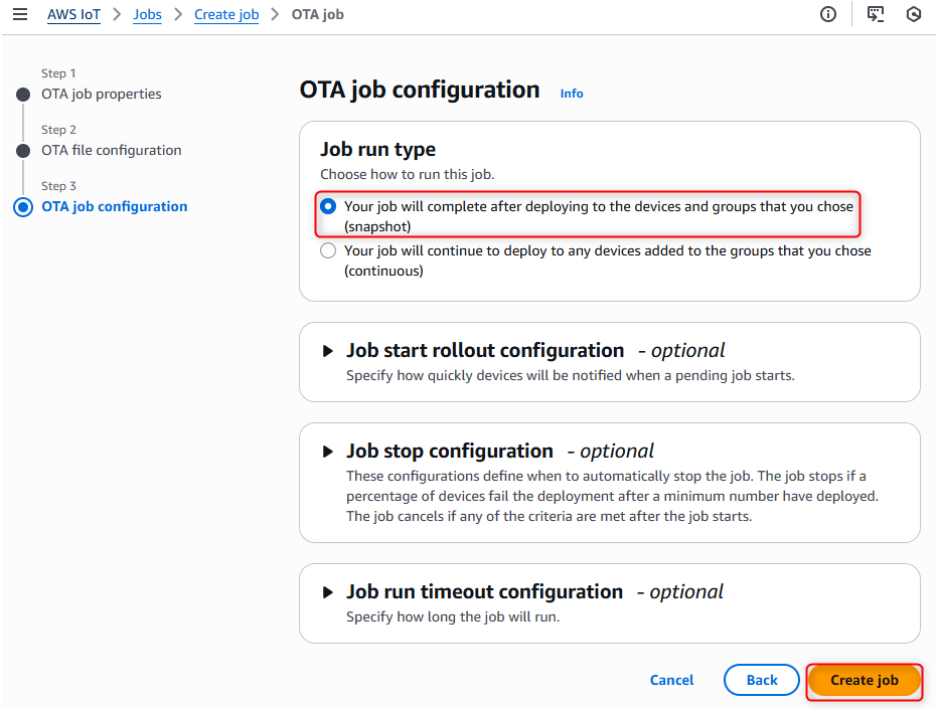
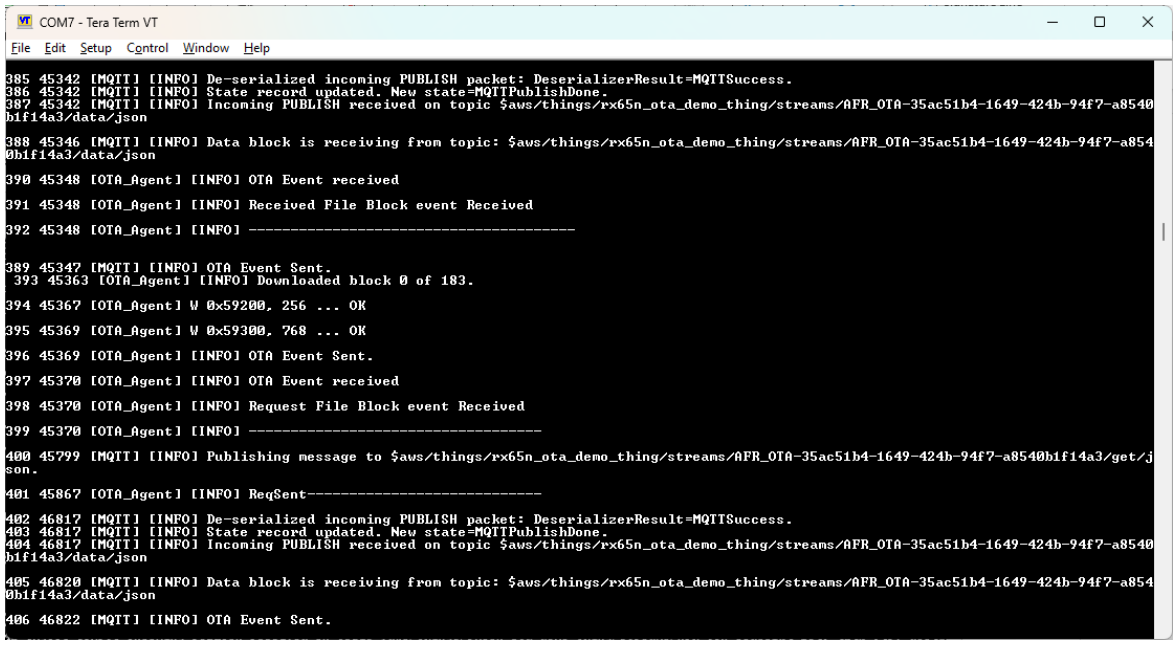


Figure 6-57 Step 3: OTA job configuration

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



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

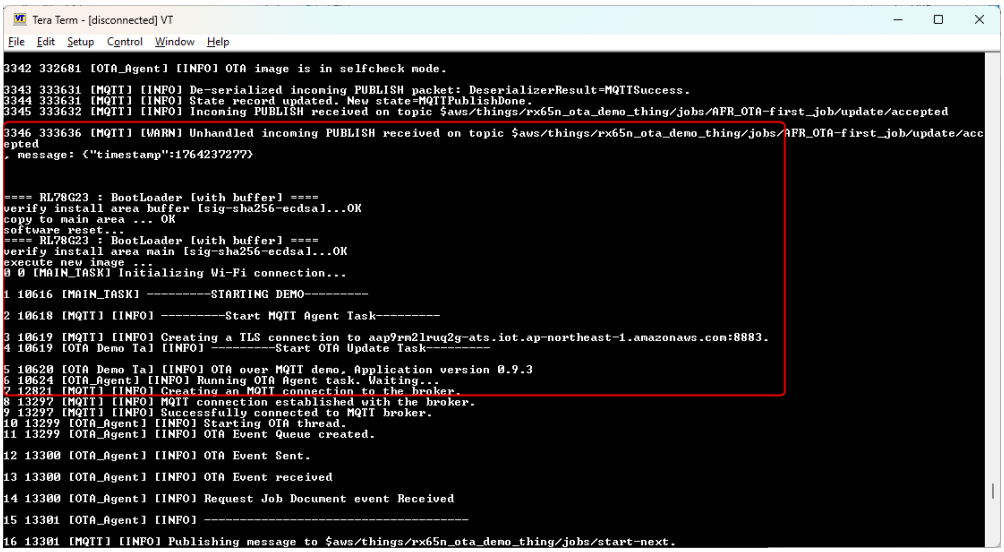


Figure 6-58 Update Image (Version 0.9.3) Started after Programming Terminates

6.5 Debugging the Initial Application

The following describes the procedure for starting the initial application from e^2 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_da16600_rl78g23-fpb project to 0, and then click “Apply and Close”.

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

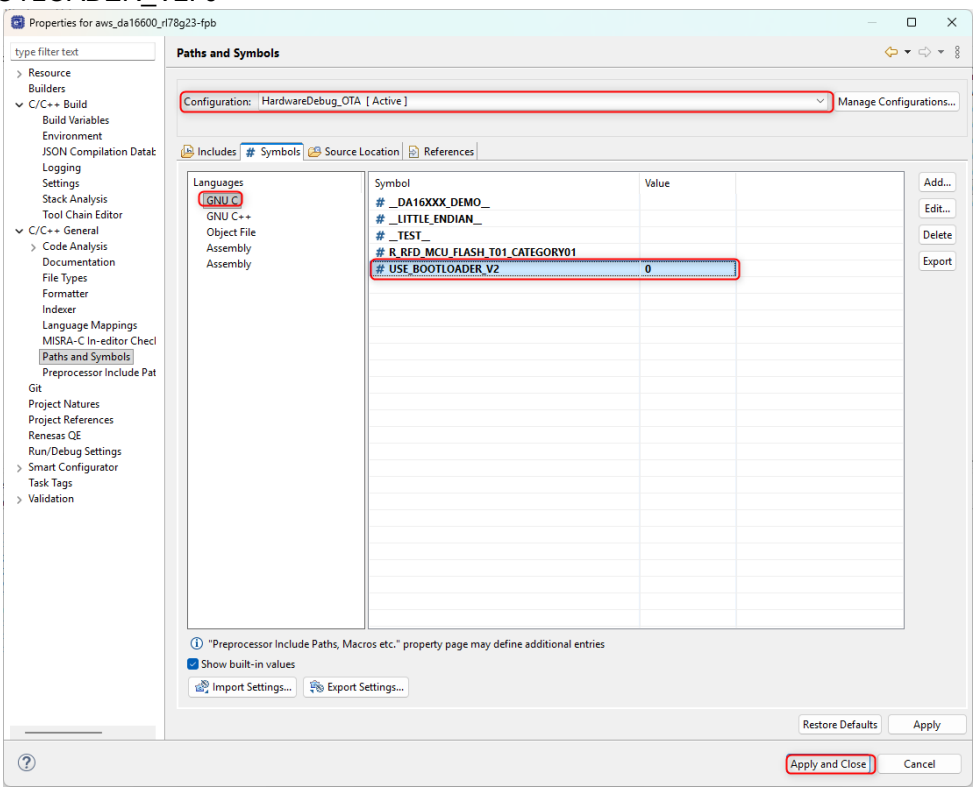


Figure 6-59 Setting the “USE_BOOTLOADER_V2” Macro to 0

② Build the aws_da16600_rl78g23-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 circuits, you don't need this process.

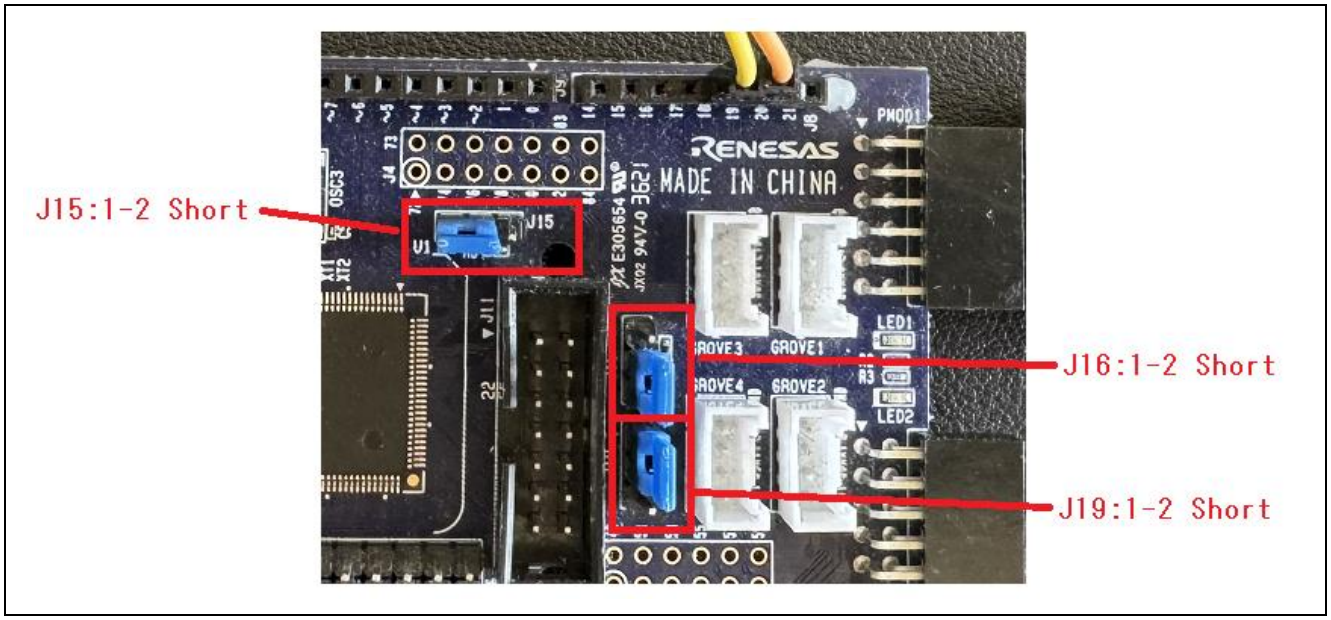


Figure 7-1 Settings for Using COM Port Debugging (Top Side)

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”

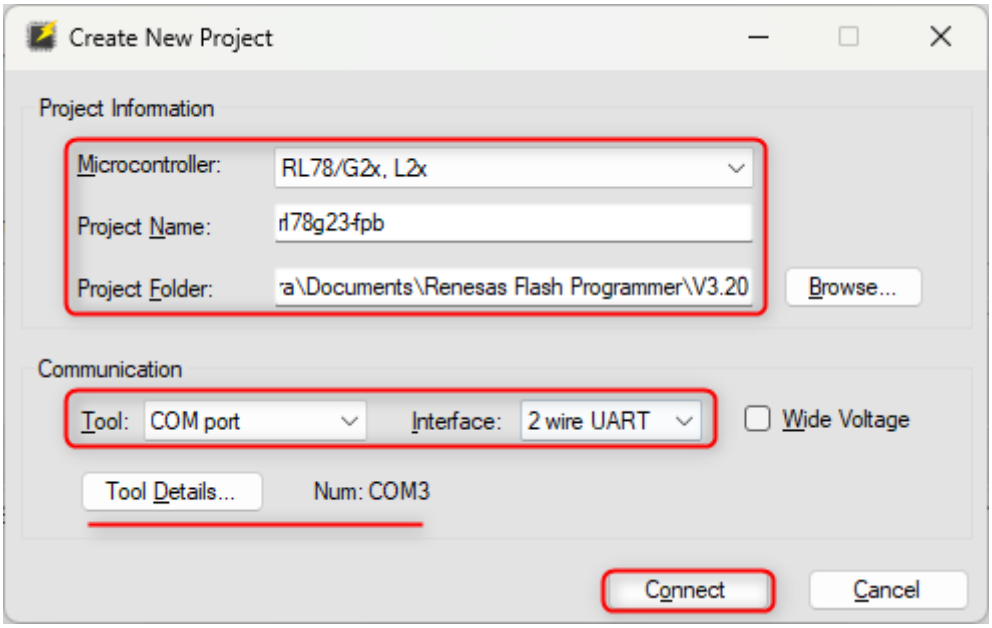


Figure 7-2 Creating a New Project and Connecting to the MCU Board

② The connection is successful if the following window appears.

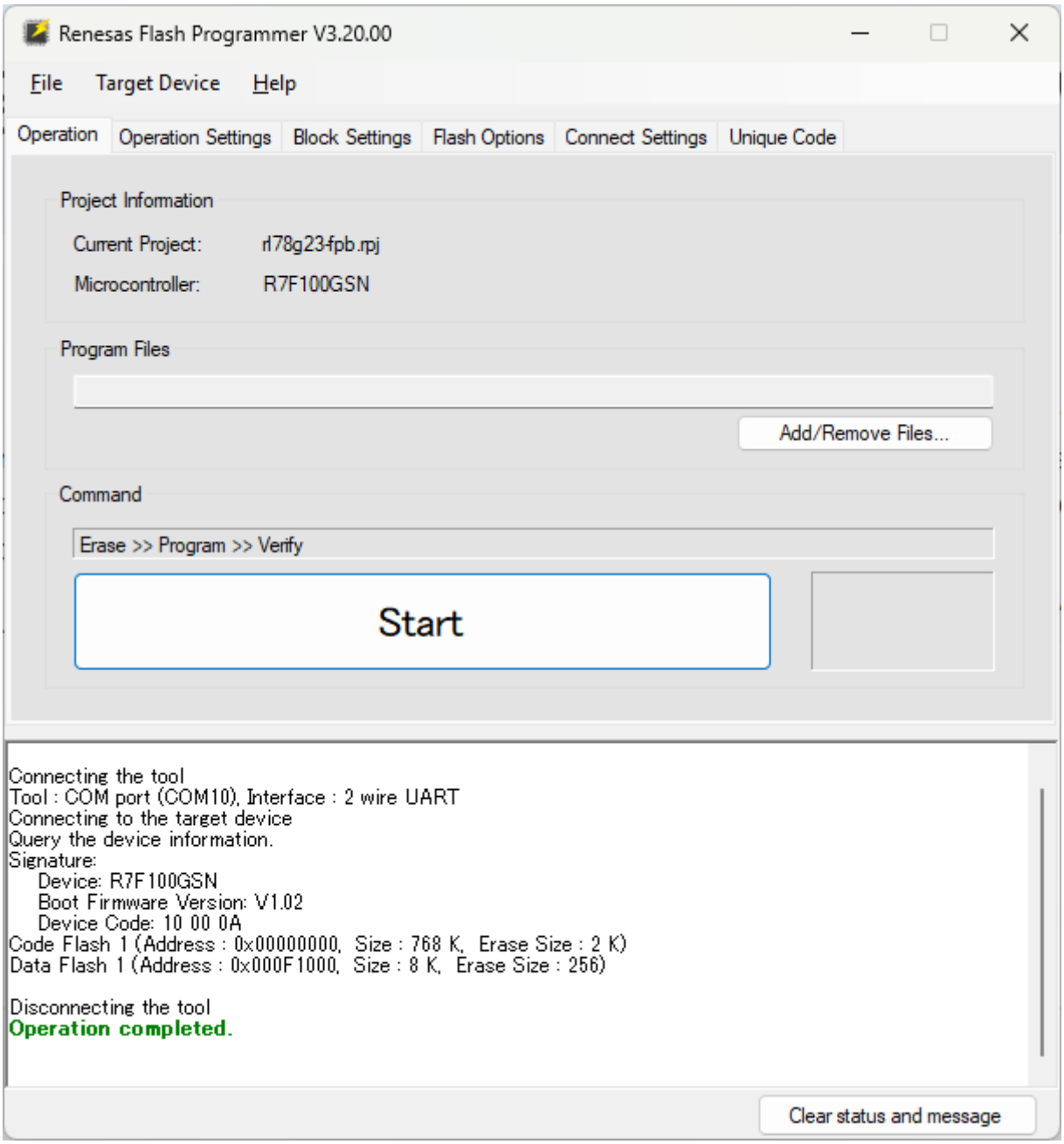


Figure 7-3 Operation completed (Connect)

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_da16600_rl78g23-fpb.mot)
- Click “Start”

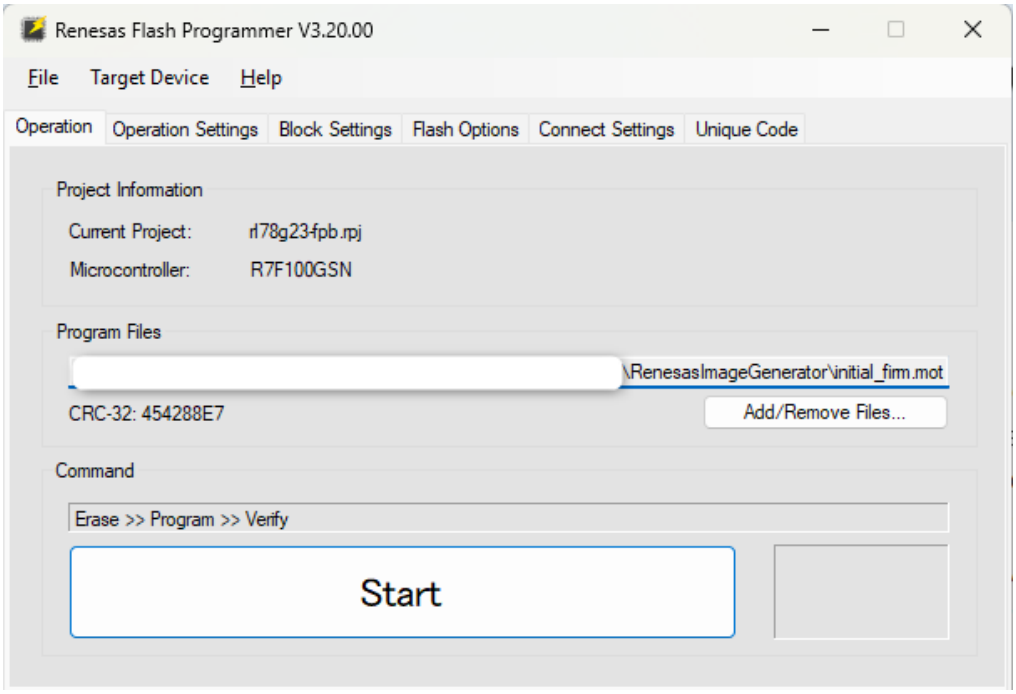


Figure 7-4 Programming a MOT File to the MCU Board

② Make sure that programming is successful.

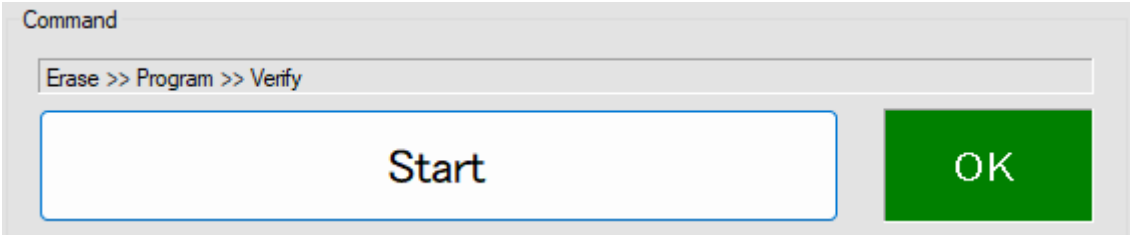


Figure 7-5 Successful programming

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 5.20 in RL78/G23-128p Fast Prototyping Board User's Manual Rev.1.10.

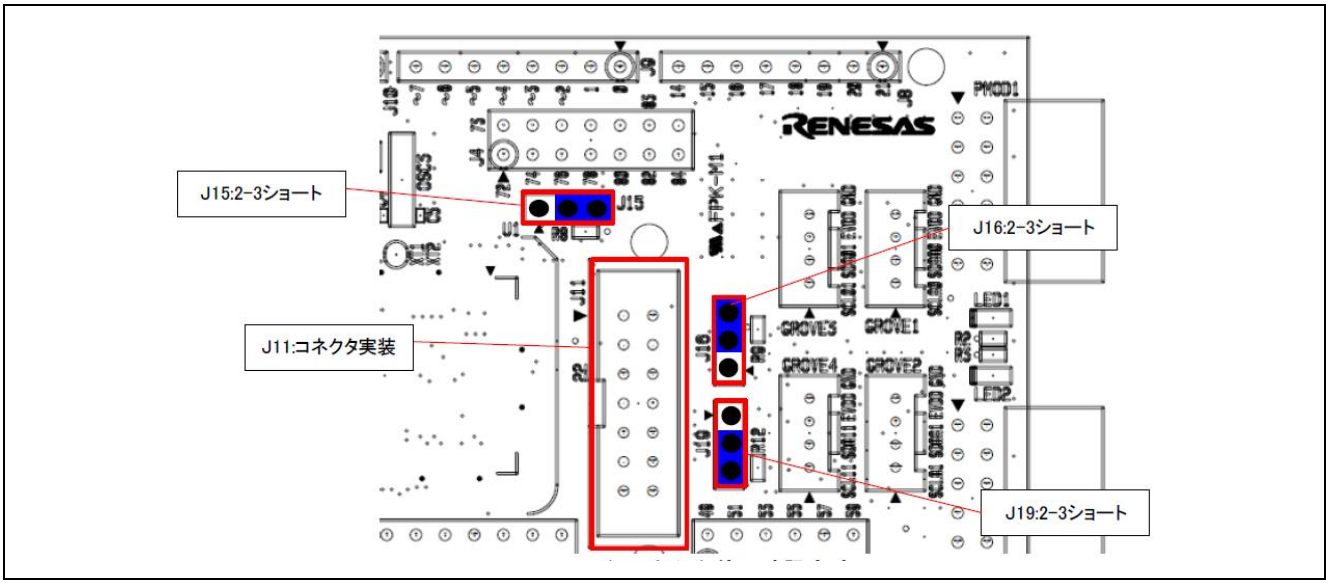


Figure 7-6 Settings for Using Emulator Connector (Top Side)

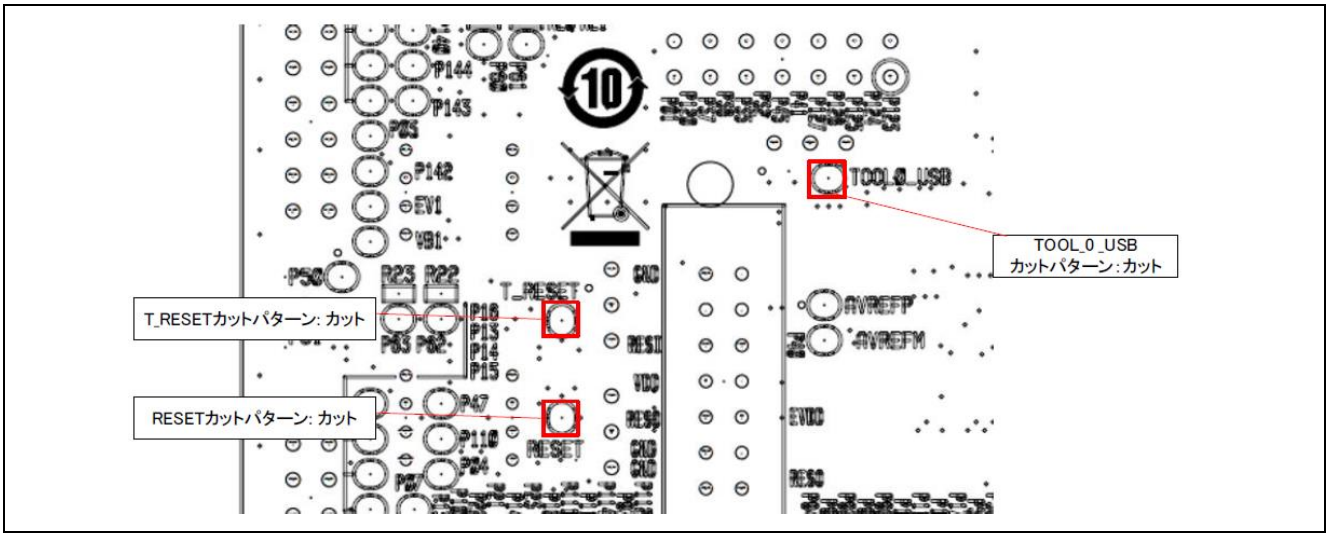


Figure 7-7 Settings for Using Emulator Connector (Solder Side)

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”

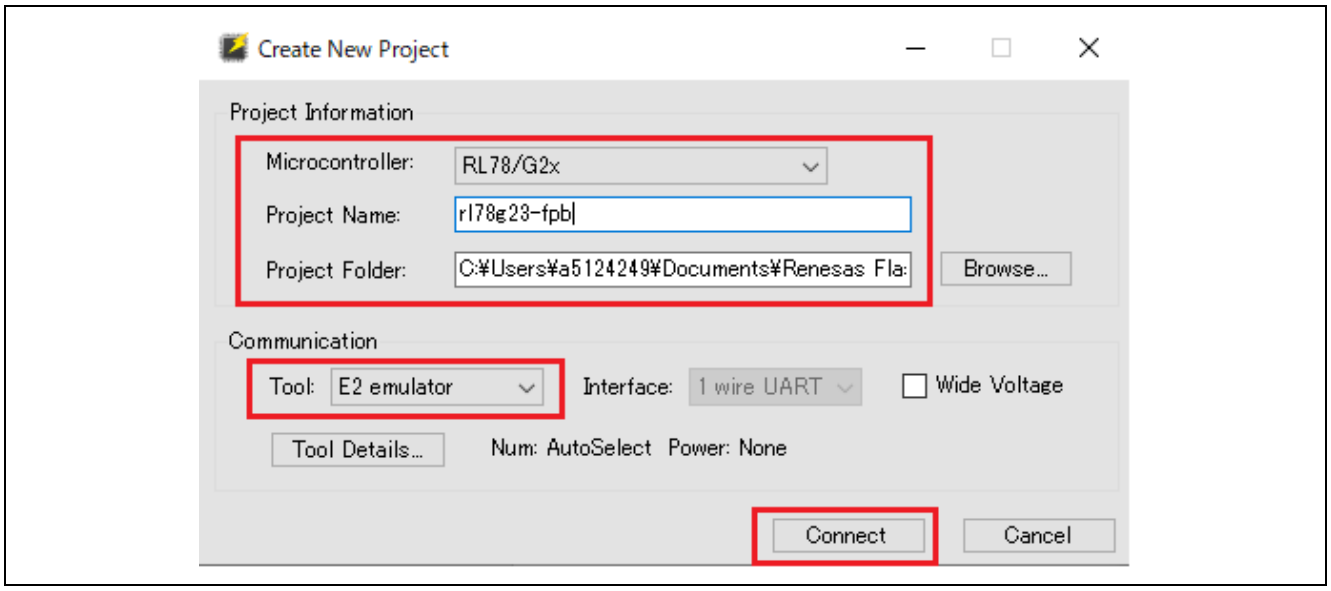


Figure 7-8 Creating a New Project and Connecting to the MCU Board

② The connection is successful if the following window appears.

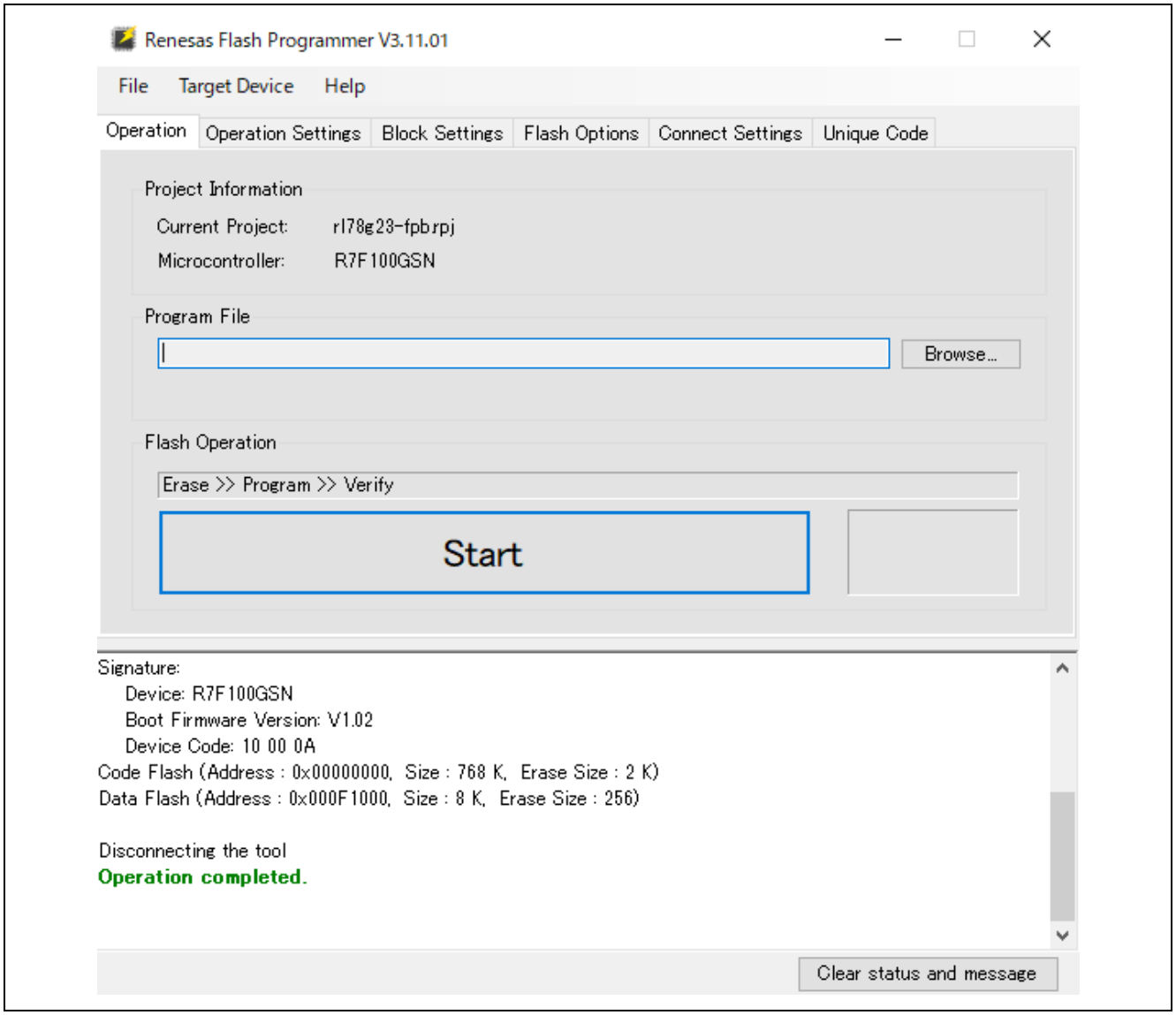


Figure 7-9 Operation completed (Connect)

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_da16600_rl78g23-fpb HardwareDebug.

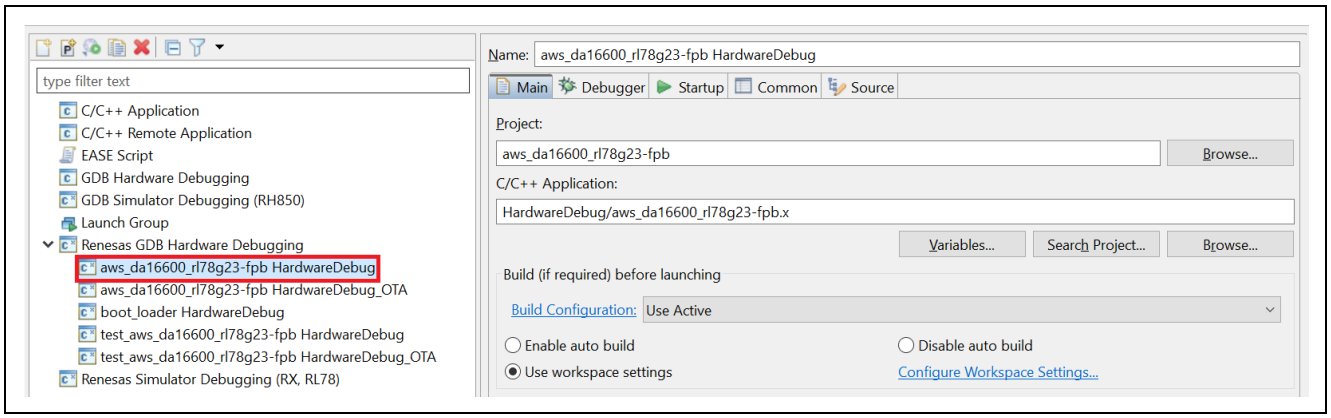


Figure 8-1 Debug Configurations of Project (PubSub)

8.1.4 Debugger Settings

Select ”Debugger” tab.

- Debug hardware: COM Port (RL78)

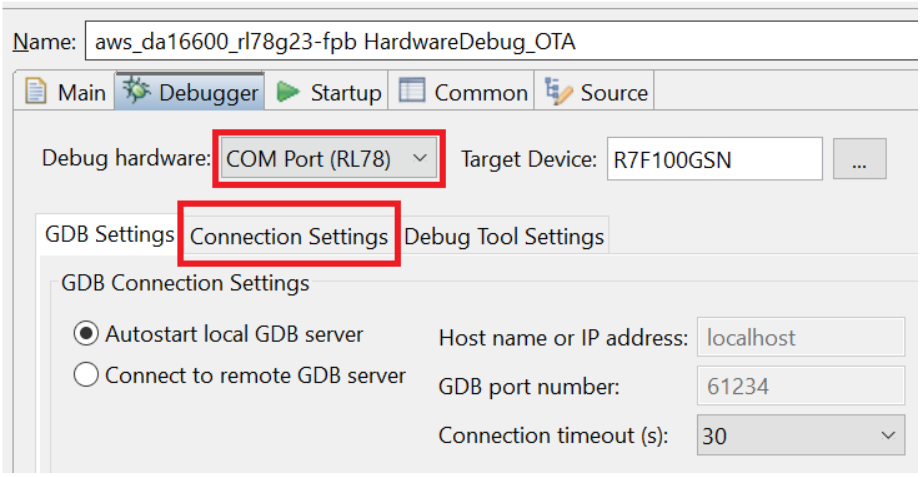


Figure 8-2 Debug hardware: COM Port (RL78)

Select “Connection Settings” tab > Connection with Target Borad.

- COM Port: COMxx
- Reset control pin: DTR

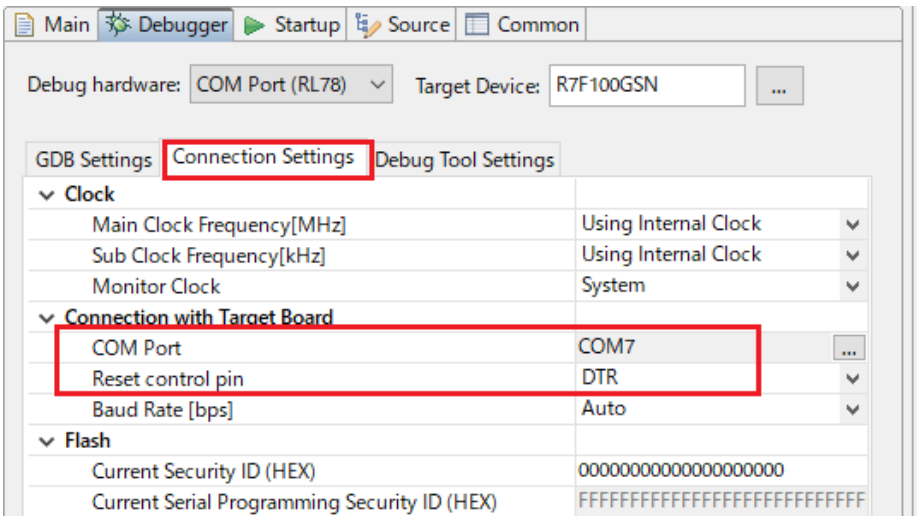
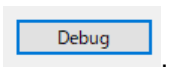


Figure 8-3 Connection Settings for Using COM Port

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.

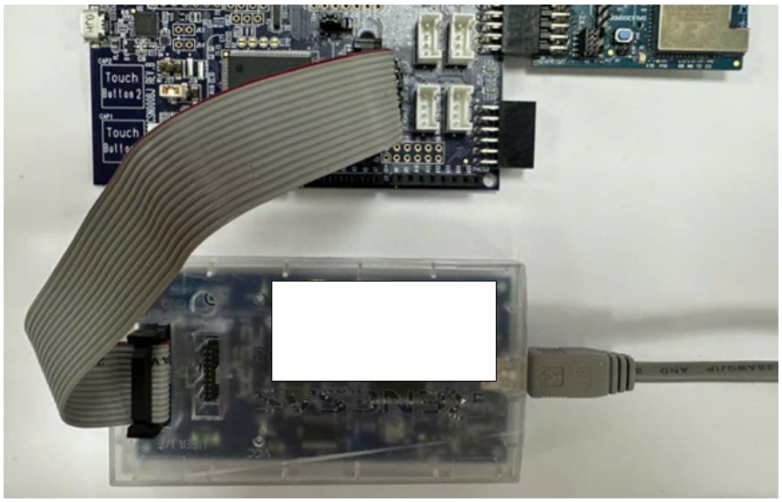


Figure 8-4 Connecting Emulator to MCU Board

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_da16600_rl78g23-fpb HardwareDebug.

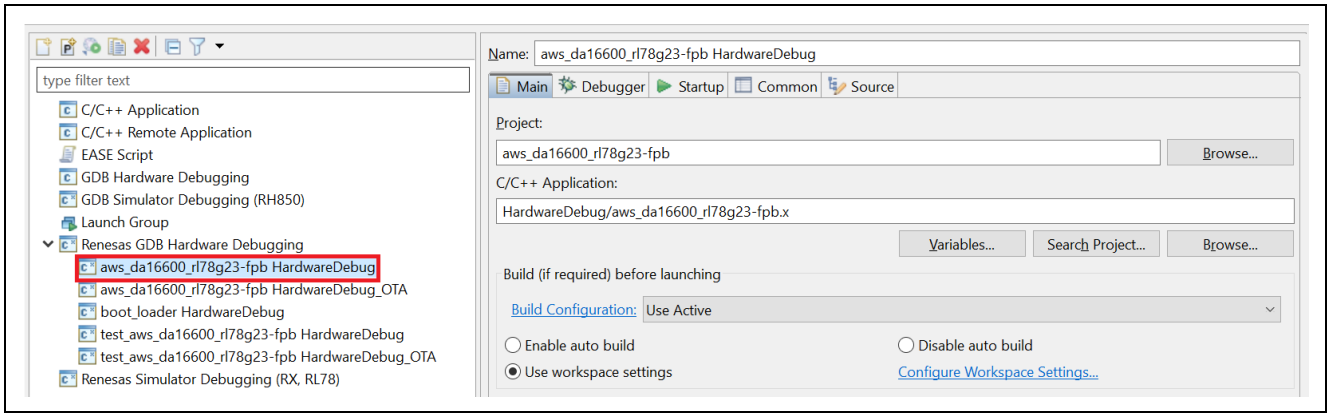


Figure 8-5 Debug Configurations of Project (PubSub)

8.2.4 Debugger Settings

Select ”Debugger” Tab.

- Debug hardware : E2 Lite (RL78)

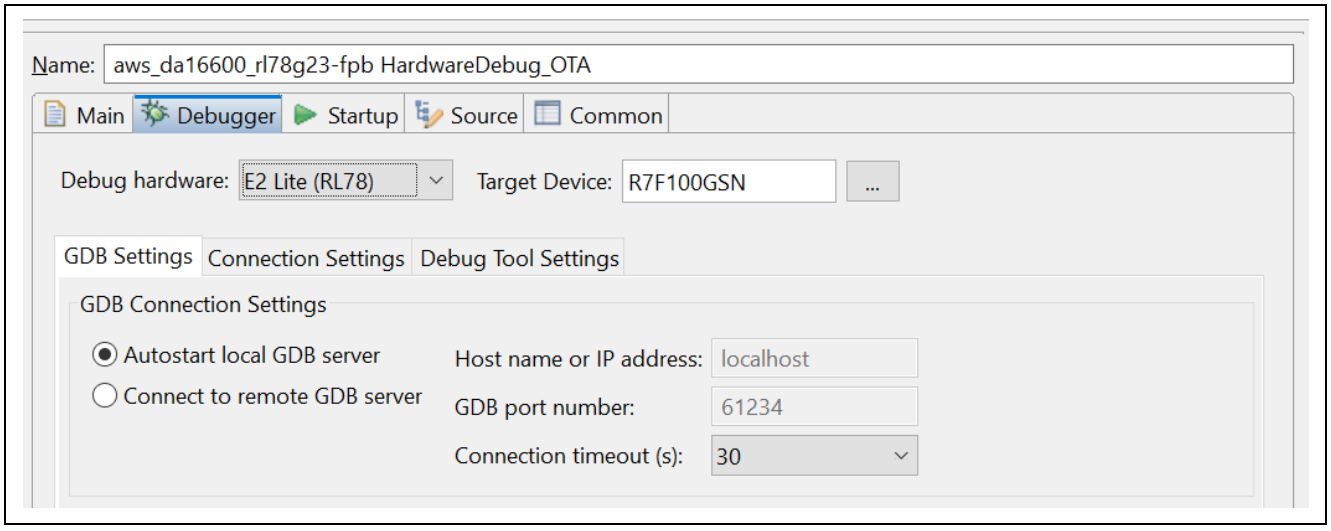


Figure 8-6 Debug hardware: E2 Lite (RL78)

Select “Connection Settings” tab > Connection with Target Borad.

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

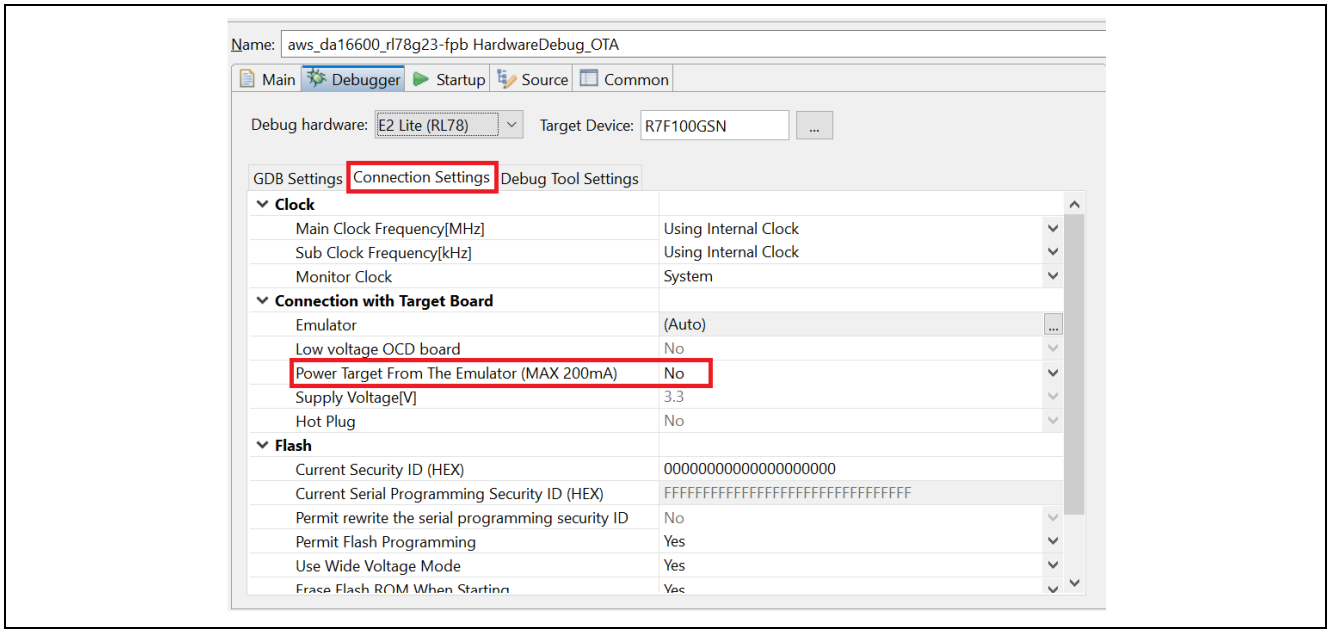
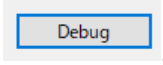


Figure 8-7 Connection Settings for Using Emulator

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 accross 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.

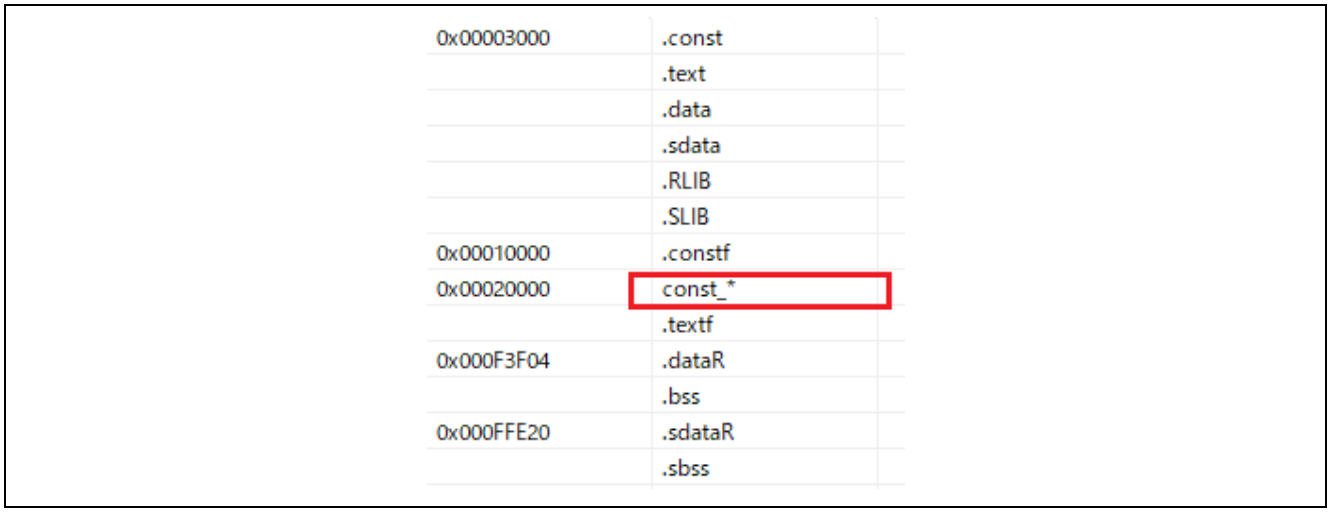


Figure 9-1 Newly Defined Constant Section (e² studio)

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 used with this product. 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 used with this product.

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	Sep. 30, 2025	-	First edition issued
1.10	Jan. 10, 2026	5	Add information for demo project (OTA)
		6-7	Update Operation Confirmation Conditions.
		10	Add the demo project (OTA) in the Hardware section description.
		12-13	Add the demo project (OTA) in the Software section description.
		14	Update files and folders for demo project (OTA)
		28-59	Add Section Setup Specific to Demo Project (OTA)

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/.