RX Family

How to implement FreeRTOS OTA using Amazon Web Services in RX65N (for v202210.01-LTS-rx-1.1.3 or later)

Preface

This application note explains how to use the OTA demo application on FreeRTOS (with IoT Libraries). For details about security, see the Renesas MCU Firmware Update Design Policy (R01AN5548)

Note : The procedures described in this application note apply to v202210.01-LTS-rx-1.1.3 and later of iot-reference-rx, a FreeRTOS reference project for RX MCUs. For earlier versions, see *How to implement FreeRTOS OTA by using Amazon Web Services on RX65N (*R01AN5549*)*.

Note : This application note shows an implementation example based on the operating environment of the CK-RX65N v1 board and the RYZ014A PMOD module,
but it can also be utilized with other boards and communication control combinations.
For each board and communication control combination, please see:

[GitHub] iot-reference-rx/Getting_Started_Guide.md at main · renesas/iot-reference-rx (github.com)

Note : Renesas announces to discontinue the existing Sequans-sourced LTE module known as the part number RYZ014A and will no longer be shipping this product.
With the discontinuation of RYZ014A, the CK-RX65N v1 board will also be discontinued.
If you are using RYZ014A in a current design or production, the Sequans part numbers, GM01Q is a pin and functionally compatible replacement for RYZ014A.

Below Cellular driver of RX family works the below alternate product combination.
- RYZ014A Cellular Module Control Module ： Sequans GM01Q is the compatible module.

Regarding EOL notice of the RYZ014A, please see ：
[The link] https://www.renesas.com/document/eln/plc-240004-end-life-eol-process-select-part-numbers?r=1503996
[The product page] https://www.renesas.com/products/wireless-connectivity/cellular-iot-modules/ryz014a-lte-cat-m1-cellular-iot-module

Target Devices

RX65N and RX651 groups

Hardware

CK-RX65N

Related Documents

Renesas MCU Firmware Update Design Policy (R01AN5548)

Contents

1. Overview

1.1 System Overview

This section shows an overview of implementing OTA using a RX65N microcontroller that supports the dual bank function and has the CK-RX65N cloud kit installed.

The dual bank function divides the ROM of the microcontroller into an execution area and a temporary area. By dynamically swapping the execution area and the temporary area, updated firmware can be written to ROM while the existing software continues to run in the execution area.

The following explains how memory is allocated during OTA updates, and what happens when memory banks are swapped using the dual bank function.

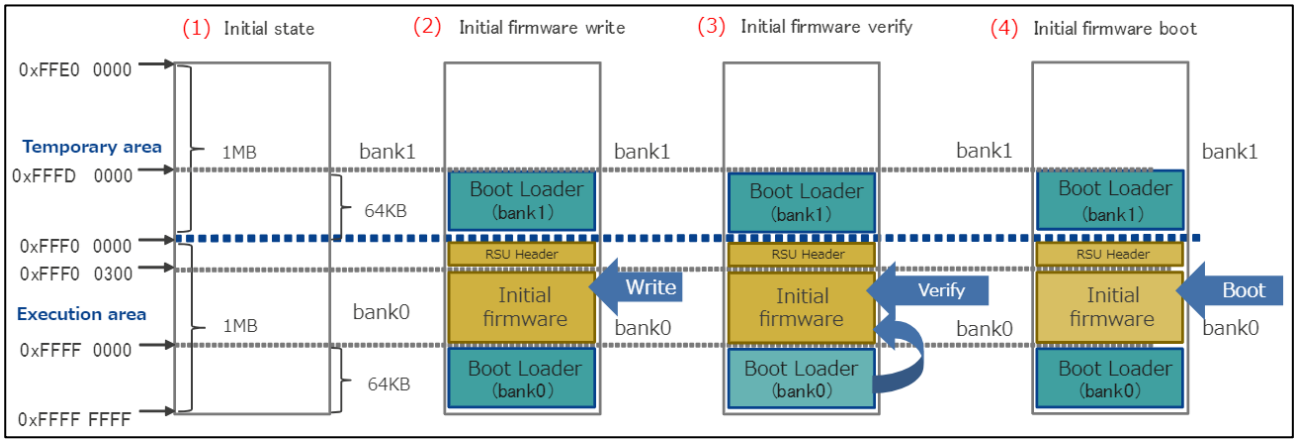


Figure 1.1 Overview of OTA Operation (1)

(1) The state that erased all the data by Renesas Flash Programmer.

(2) Memory state after writing combined bootloader and initial firmware data* in Renesas Flash Programmer.

* It refers to the data that Boot Loader(bank0) + Initial firmware + RSU Header + BootLoader(bank1) are combined. For details of RSU Header, please refer to 4.2 image file of RX Family Firmware Update module Using Firmware Integration Technology Application Notes(R01AN6850).

(3) After reset, bootloader (bank0) verifies Initial firmware.

(4) Initial firmware is booted.

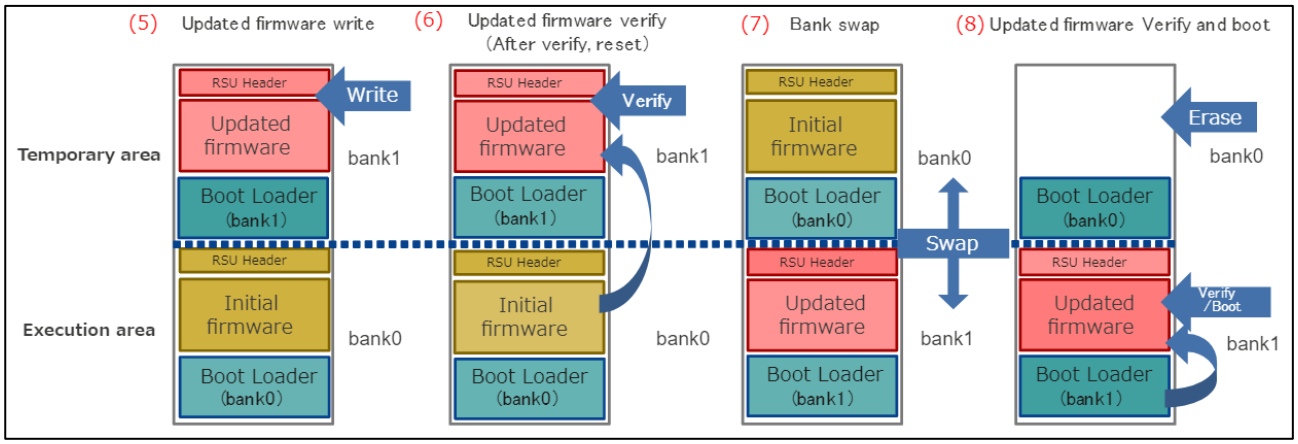


Figure 1.2 Overview of OTA Operation (2)

(5) When new firmware is received from AWS, it is written to bank1.

Initial firmware operation is executed by BGO function while writing to bank1.

(6) Updated firmware is verified by the initial firmware.

(7) Bank0 and bank1 are swapped, and bank1 is designated as the execution area.

(8) Updated firmware is verified by bootloader.

Erase Initial firmware in bank0 and execute Updated firmware written to bank1.

1.2 Operation Verification Environment for Hardware

The operation verification environment (hardware) for the demo project is listed below.

Table 1.1 Operation Verification Environment for Hardware

Item	
Board	CK-RX65N (Cellular/Ethernet) *1
Cellular module	RYZ014A (incorporated into CK-RX65N)
SIM	LTE Cat-M1-compatible SIM (micro-SIM) *2

Notes: 1. Cellular communication is used in this sample.

2. To use the SIM card supplied with the CK-RX65N kit, activate the SIM card by following the procedure in *4.1.5 Activating SIM card* in the following application note:
SIM activation, Creating the trial account and using Dashboard with RYZ014A or Ethernet Application for AWS - Getting Started Guide (R01QS0064)

1.3 Operation Verification Environment for Software

The operation verification environment (software) for the demo project is listed below.

Table 1.2 Operation Verification Environment for Software

Item	
Integrated development environment	e^2 studio 2024-04
Compiler	Renesas CC-RX v3.05.00
FreeRTOS	v202210.01-LTS-rx-1.2.1
Log monitoring tool	Tera Term v4.106
Python	Python 3.11.0
Keygen tool	Win64 OpenSSL v3.0.12
Flash programming tool	Renesas Flash Programmer V3.12.00
Renesas Image Generator	Version 3.02 (supplied with Firmware Update module Rev.2.01)

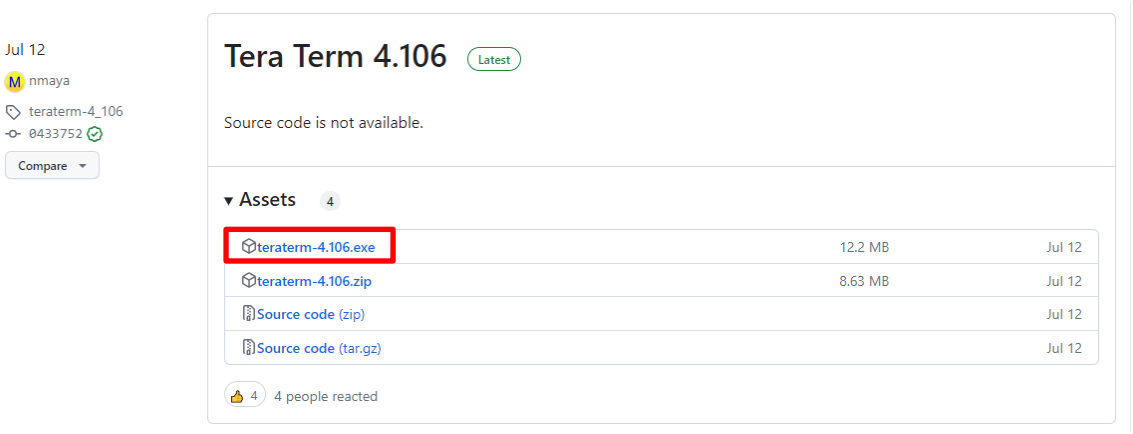
2. Prerequisites

2.1 Installing Tera Term

(1) Access the Tera Term download site.

Tera Term download web site (GitHub)

(2) Download the Tera Term installer.



(3) Run the installer and follow the prompts to install Tera Term.

(4) Confirm that Tera Term starts when you click the Tera Term icon in the Start menu.

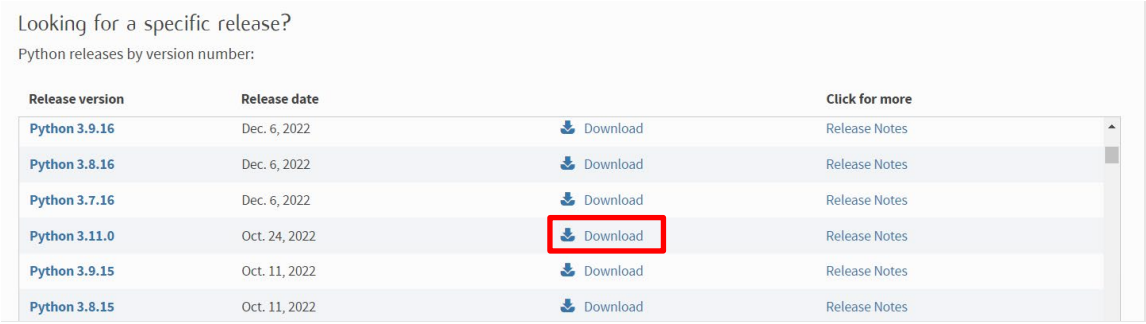
2.2 Installing Python

(1) Access the Python download web site.

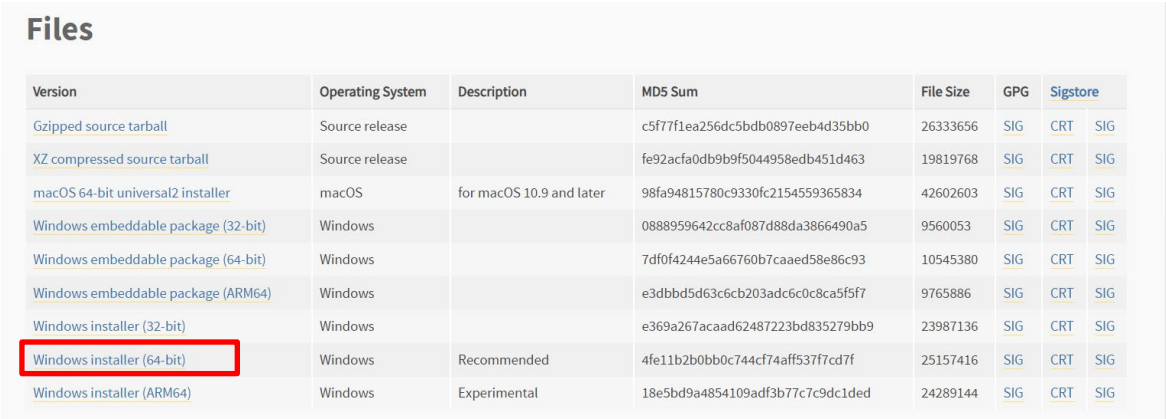
Python download web site

(2) Download the Python 3.11.0 installer.

Click the **Download** link for Python 3.11.0.

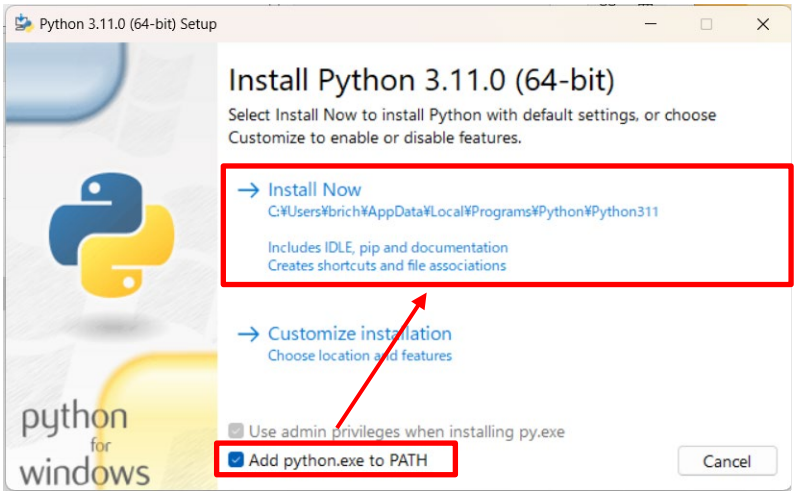


Download the installer for the operating system you are using.



(3) Run the installer and follow the prompts to install Python.

On the installation screen, select the **Add python.exe to PATH** check box.



(4) Open a command prompt, and confirm that Python 3.11.0 is installed.

Execute the following command and confirm that information appears.

python -V

```
C:¥Users>python -V
Python 3.11.0
```

(5) Install the Python encryption library (pycryptodome).

Install the encryption library by executing the following command:

pip install pycryptodome

```
C:¥Users>pip install pycryptodome
Requirement already satisfied: pycryptodome in c:¥users¥os      ¥appdata¥local¥programs¥python¥p
ython311¥lib¥site-packages (3.18.0)

[notice] A new release of pip is available: 23.1.2 -> 23.2.1
[notice] To update, run: python.exe -m pip install --upgrade pip
```

2.3 Installing OpenSSL

(1) Access the Win32/Win64 download web site for OpenSSL.

(Win32/Win64 OpenSSL Installer for Windows - Shining Light Productions (slproweb.com))

(2) Download the OpenSSL installer.

Download the installer for the operating system you are using.

Win64 OpenSSL v3.0.12 Light EXE | MSI	5MB Installer	Installs the most commonly used essentials of Win64 (by the creators of OpenSSL). Only installs on 64-bit ve chipsets. Note that this is a default build of OpenSSL a information can be found in the legal agreement of the
Win64 OpenSSL v3.0.12 EXE | MSI	140MB Installer	Installs Win64 OpenSSL v3.0.12 (Recommended for sc OpenSSL). Only installs on 64-bit versions of Windows this is a default build of OpenSSL and is subject to loca found in the legal agreement of the installation.
Win32 OpenSSL v3.0.12 Light EXE | MSI	4MB Installer	Installs the most commonly used essentials of Win32 (32-bit OpenSSL for Windows. Note that this is a defau and state laws. More information can be found in the le
Win32 OpenSSL v3.0.12	116MB Installer	Installs Win32 OpenSSL v3.0.12 (Only install this if you

(3) Run the installer and follow the prompts to install OpenSSL.

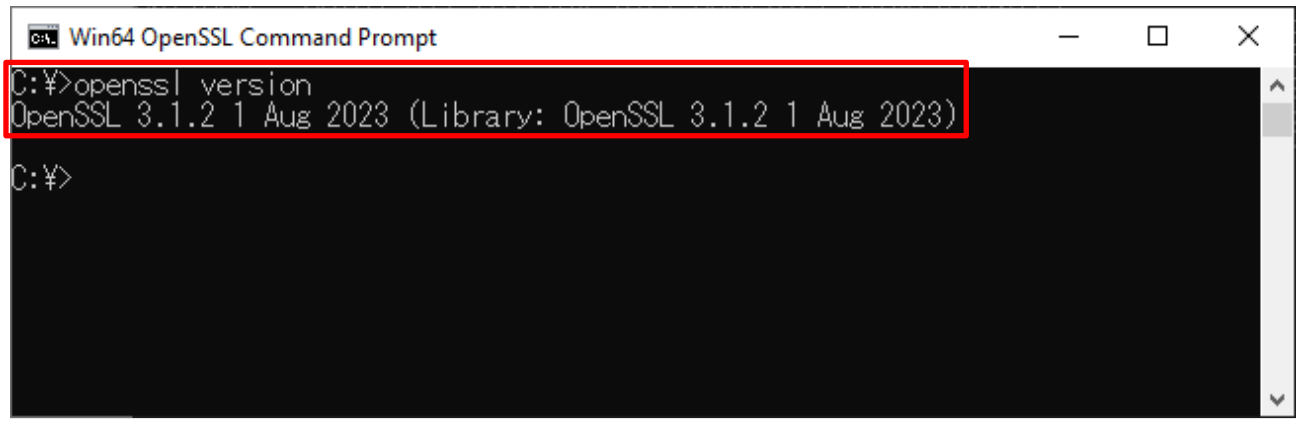
Select the option to copy the OpenSSL DLLs to the OpenSSL binaries directory.

(4) From the Start menu, open the Win64 OpenSSL Command Prompt.

(5) Confirm that you can run the openssl command from the command prompt.

Execute the following command and confirm that version information appears.

openssl version



2.4 Installing Renesas Image Generator

Renesas Image Generator is a tool that generates the firmware images used by the firmware update module. Renesas Image Generator can generate the following images for use by the firmware update module:

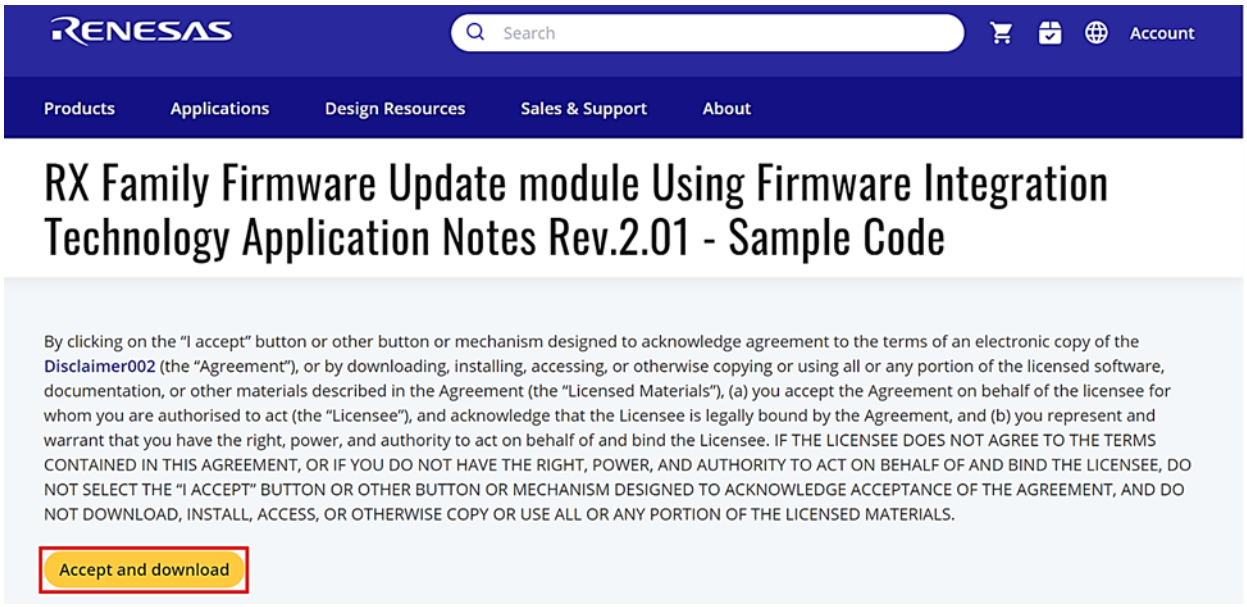
- Initial image: An image file containing the bootloader and application program written by flash writer during initial system configuration (extension: mot)
- Update image: An image file containing the updated firmware (extension: rsu)

Renesas Image Generator is provided as part of the Firmware Update FIT module.

Note: Version Rev.2.00 and later of the Firmware Update module only support firmware generation using Python scripts.

(1) Download the firmware update module from the page linked to below.

RX Family Firmware Update module Using Firmware Integration Technology Application Notes Rev.2.03 - Sample Code | Renesas



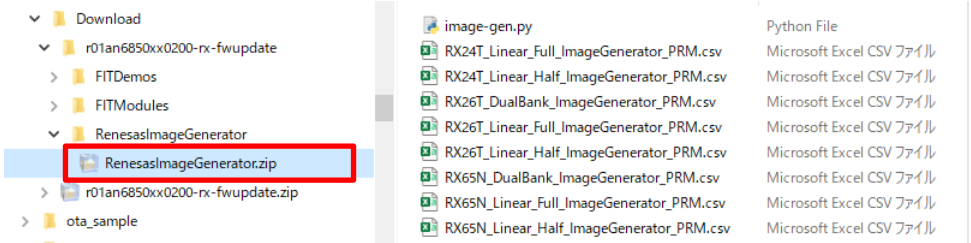
Alternatively, download the firmware update module from the GitHub repository linked to below.

https://github.com/renesas/renesas-image-generator

(2) Extract the firmware update module from the downloaded archive file.

(3) Extract Renesas Image Generator.

Extract the file RenesasImageGenerator.zip in the firmware update module. The RenesasImageGenerator folder contains the Renesas Image Generator script file (image-gen.py) and the parameter files for various devices (*_ImageGenerator_PRM.csv).



2.5 Connecting the CK-RX65N

Figure 2.1 Back of RYZ014A PMOD

(1) Insert the SIM card into the CN6 slot on the RYZ014A PMOD.

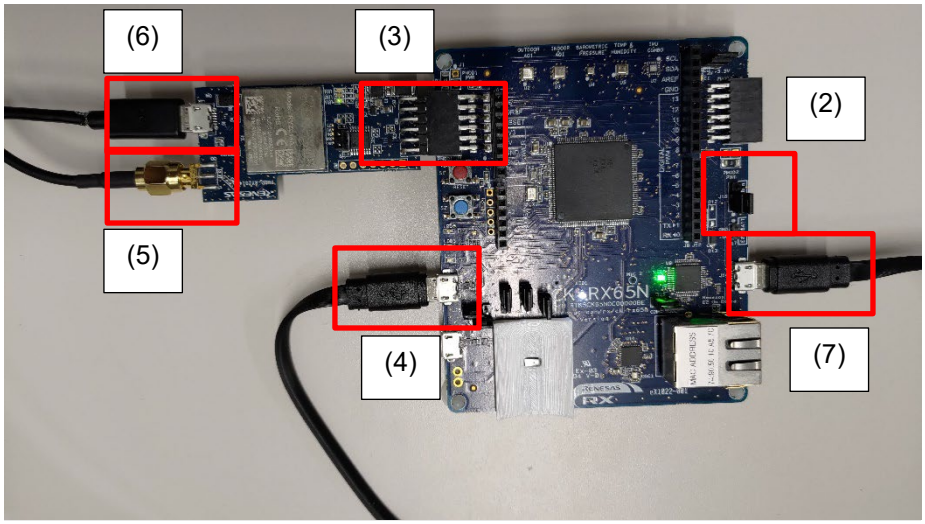


Figure 2.2 Front of Base Board and RYZ014A PMOD

(2) On the base board, position the jumper on pins 1-2 of J16 to enable debugging mode.

(3) Connect the RYZ014A PMOD to PMOD1 on the base board.

(4) Connect J20 on the base board to a PC using a USB cable (USB serial connection).

(5) Connect an antenna to CN3 of the RYZ014A PMOD.

(6) Supply power by connecting a USB cable to CN4 of the RYZ014A PMOD.

(7) Connect J14 on the base board to a PC using a USB cable (debugger connection).

Note: Perform step (6) if you have a spare USB cable available.
If you do not supply power to the RYZ014A PMOD, communication might become unstable.

3. Setting Up AWS

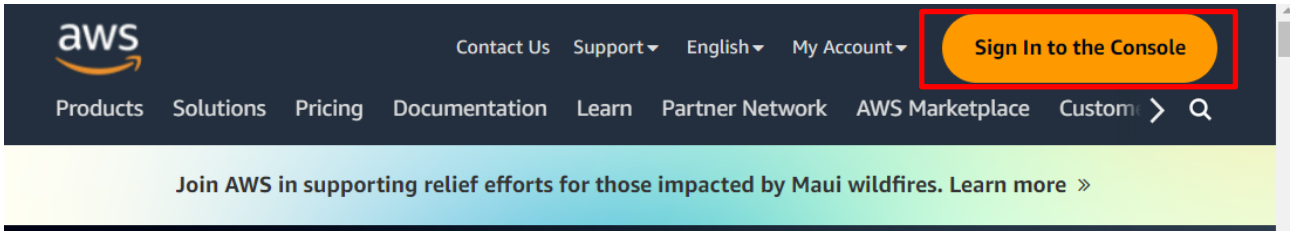
To run the FreeRTOS demo, you must have an AWS account (the root user, or an IAM user with permissions to access AWS IoT and FreeRTOS cloud services).

For details on how to sign up for an AWS account and add permissions to users, see https://docs.aws.amazon.com/freertos/latest/userguide/freertos-prereqs.html.
For details on how to set up OTA updates, see https://docs.aws.amazon.com/freertos/latest/userguide/ota-prereqs.html.

You must also configure the source code as explained in 4.2 to allow the demo to communicate with AWS.

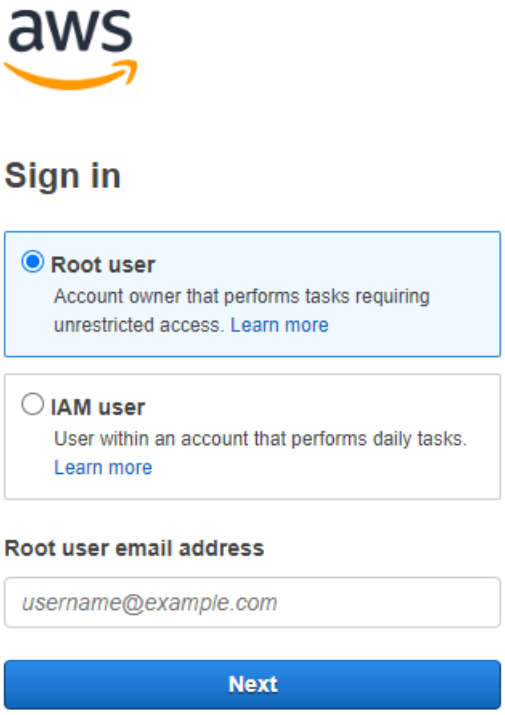
3.1 Signing In to the AWS Console

(1) Access the AWS web site (https://aws.amazon.com/) and click **Sign In to the Console**.



(2) Enter your email address or account ID, and then click **Next**.

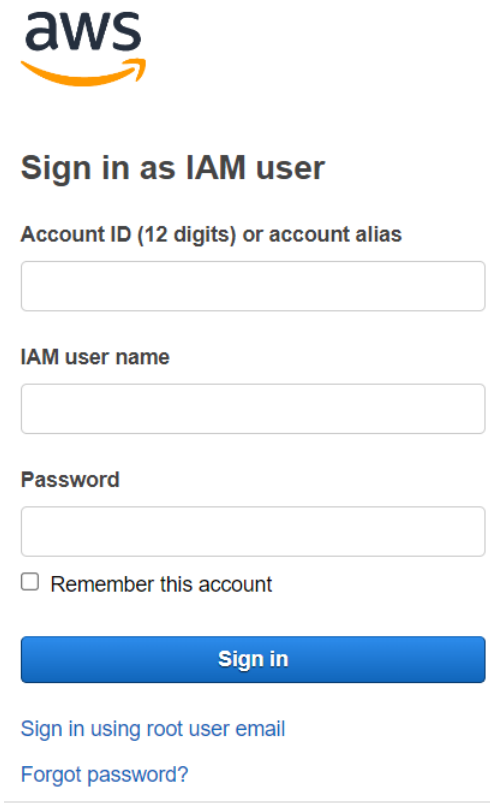
If the account you are using to sign in is the root user, enter the root user email address. If the account is an IAM user, enter the account ID. (You might skip this step if you have already signed in.)



(3) Enter your password and then click **Sign in**.

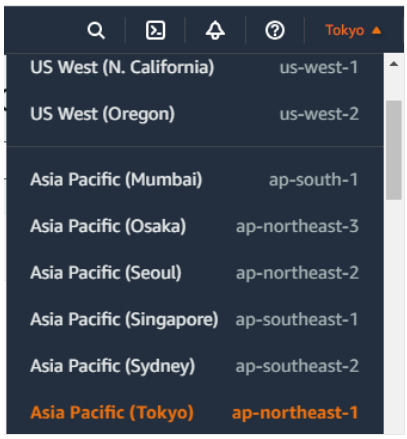
For root users

For IAM users



3.2 Setting Your Region in AWS

After logging in to AWS, select your region in the top right of the screen.



3.3 Registering Your Device in AWS

The following explains the preparations necessary to run the demo project in AWS. Set up AWS by referring to the following tutorial.

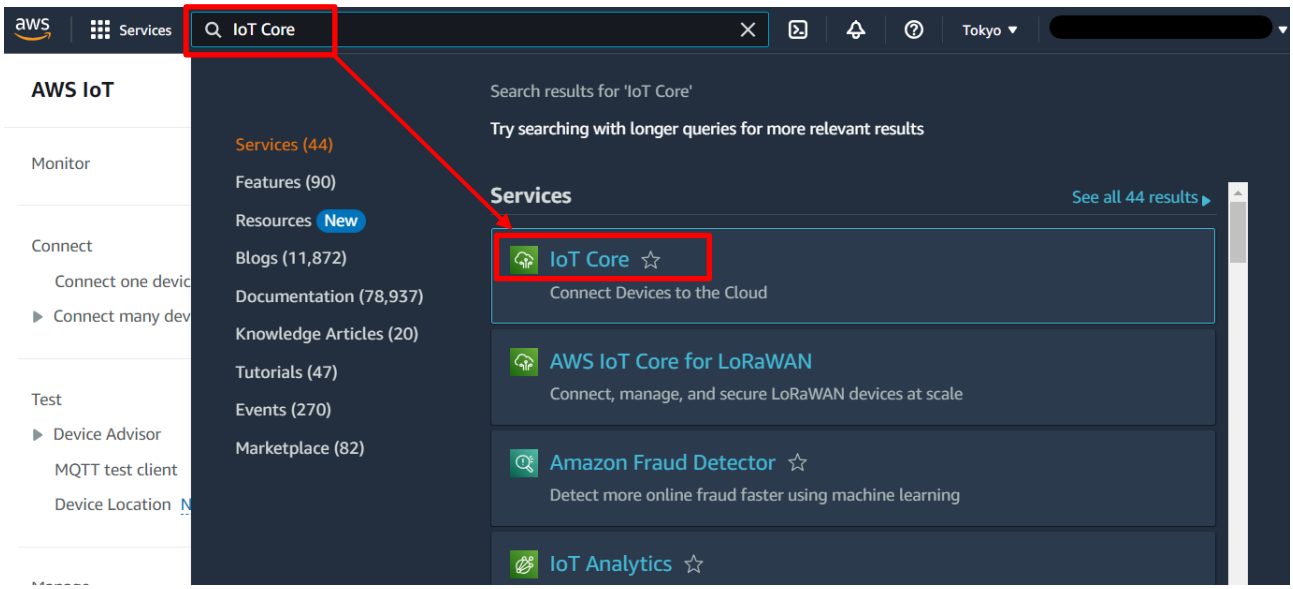
3.3.1 Setting Policies

Assign access permissions (policies) for AWS and other resources to the device you want to connect to AWS.

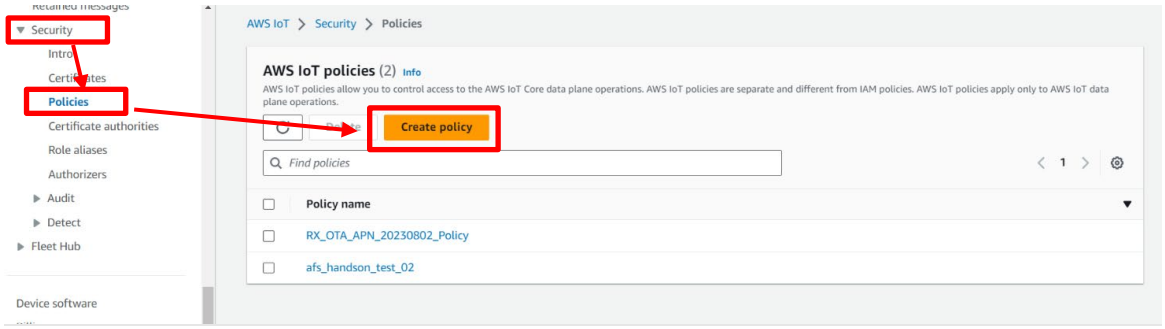
Assign the following policies to the device connected in this application note:

- **iot:Connect**: Connects to AWS IoT
- **iot:Publish**: Publishes a topic
- **iot:Subscribe**: Subscribes to a topic
- **iot:Receive**: Receives messages from AWS IoT

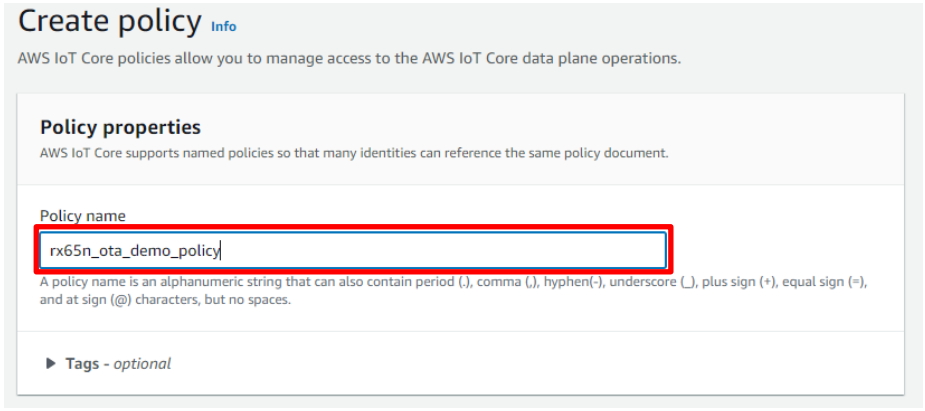
(1) Enter IoT Core in the search box at the top of the screen, and click **IoT Core** in the search results.



(2) In the menu, click **Security** and then **Policies**, and then click the **Create policy** button.

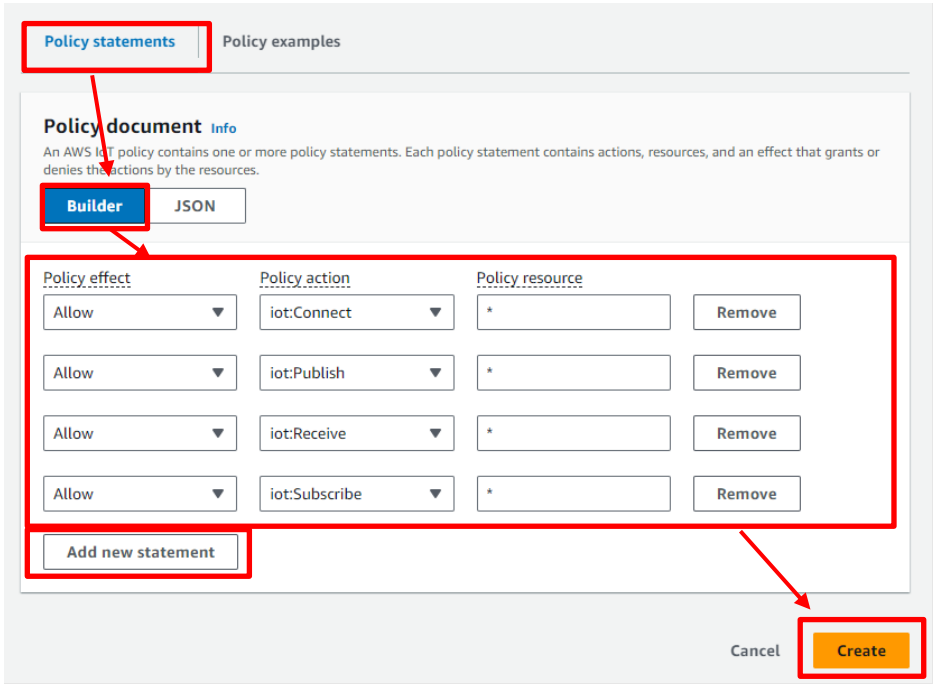


(3) Enter a policy name (for example: rx65n_ota_demo_policy).



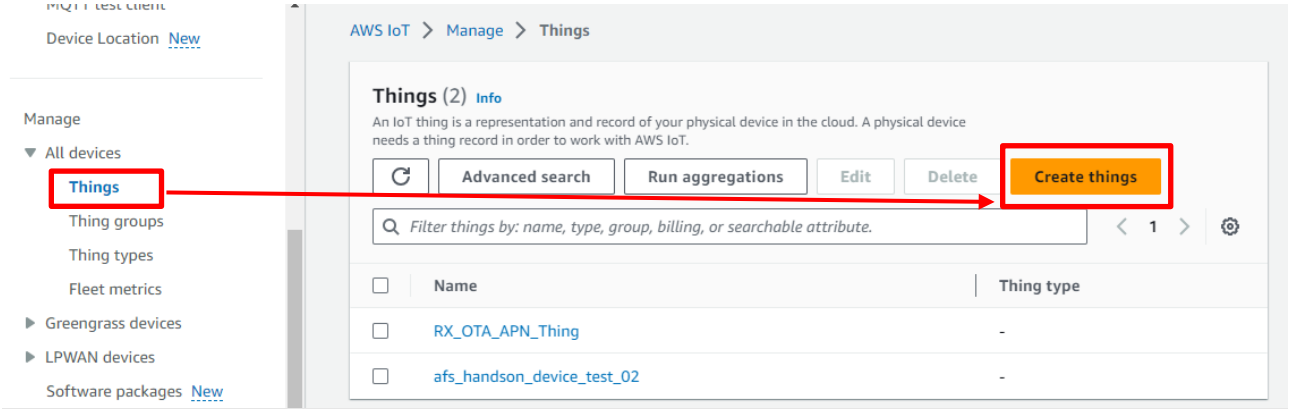
(4) Click the **Policy statements** tab, and in the **Policy document** area, click **Builder**. Enter the policy settings as shown in the following figure, and then click **Create**.

Because the policy initially contains only one statement, you must add more statements by clicking the **Add new statement** button.

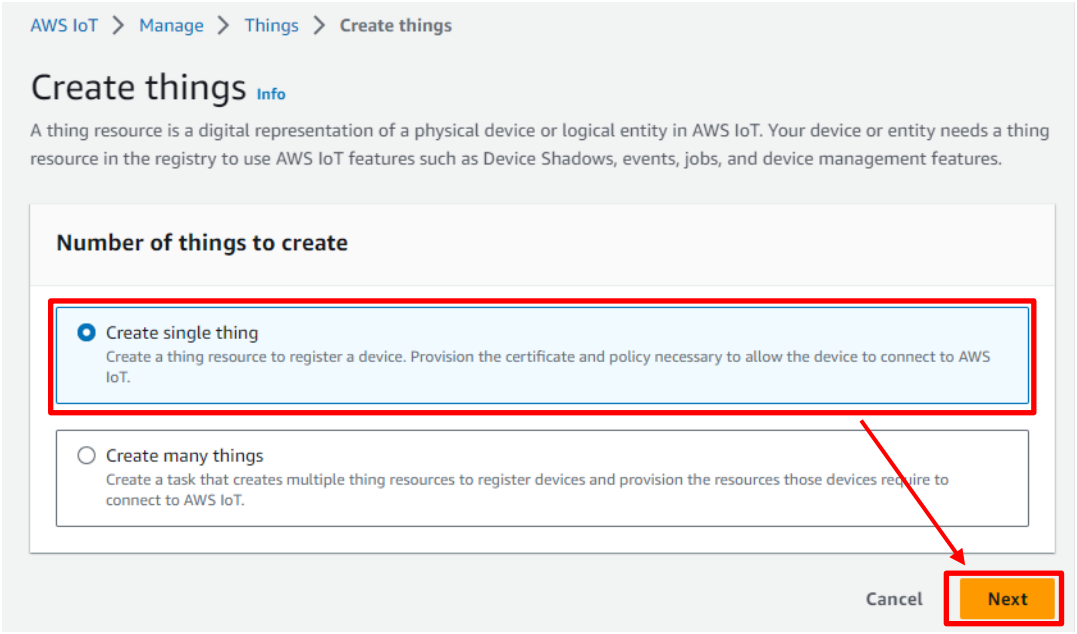


3.3.2 Registering Your Device as a *Thing* in AWS IoT

(1) In the menu, click **Manage**, **All devices**, and **Things**, and then click the **Create things** button.

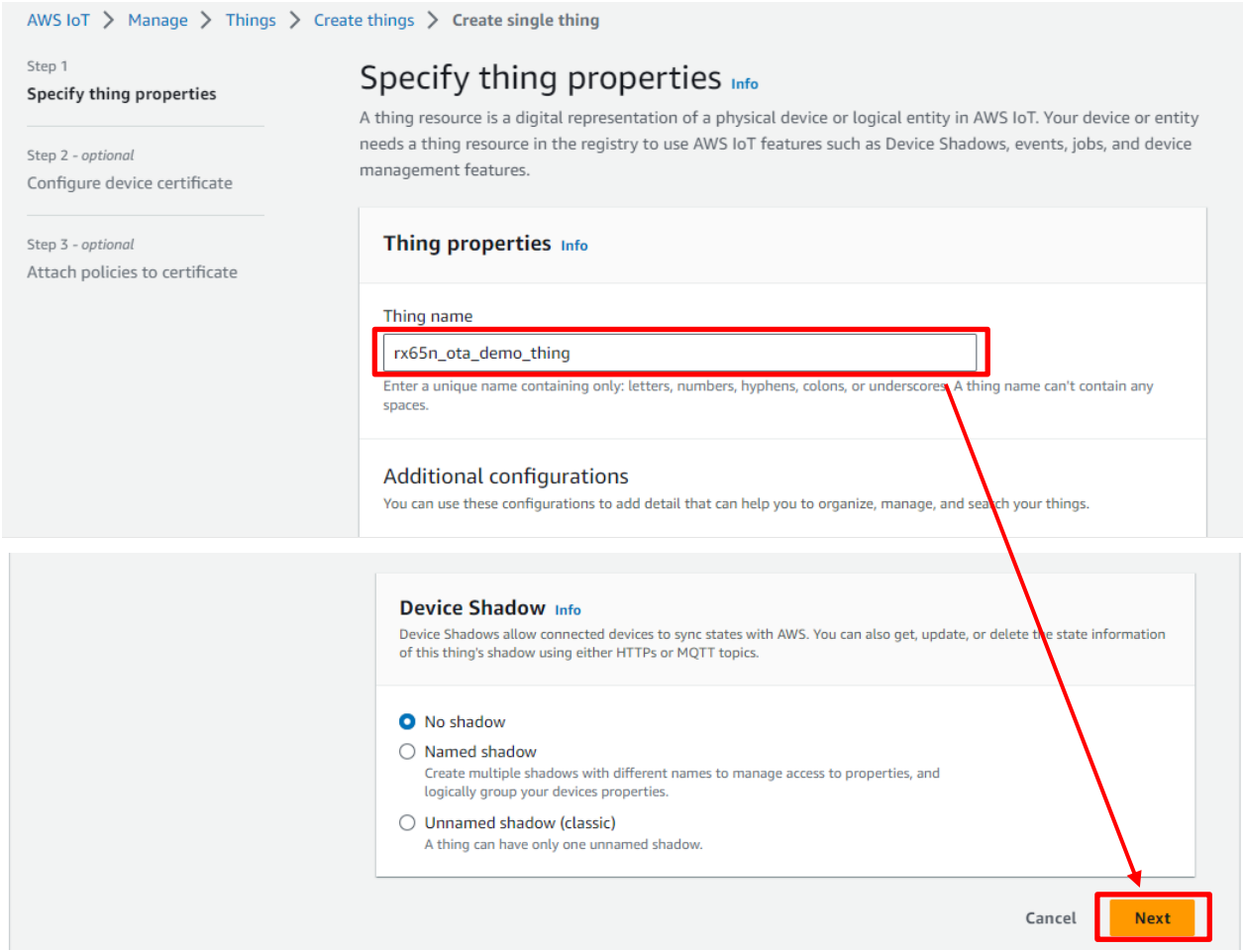


(2) Select **Create single thing** and then click **Next**.

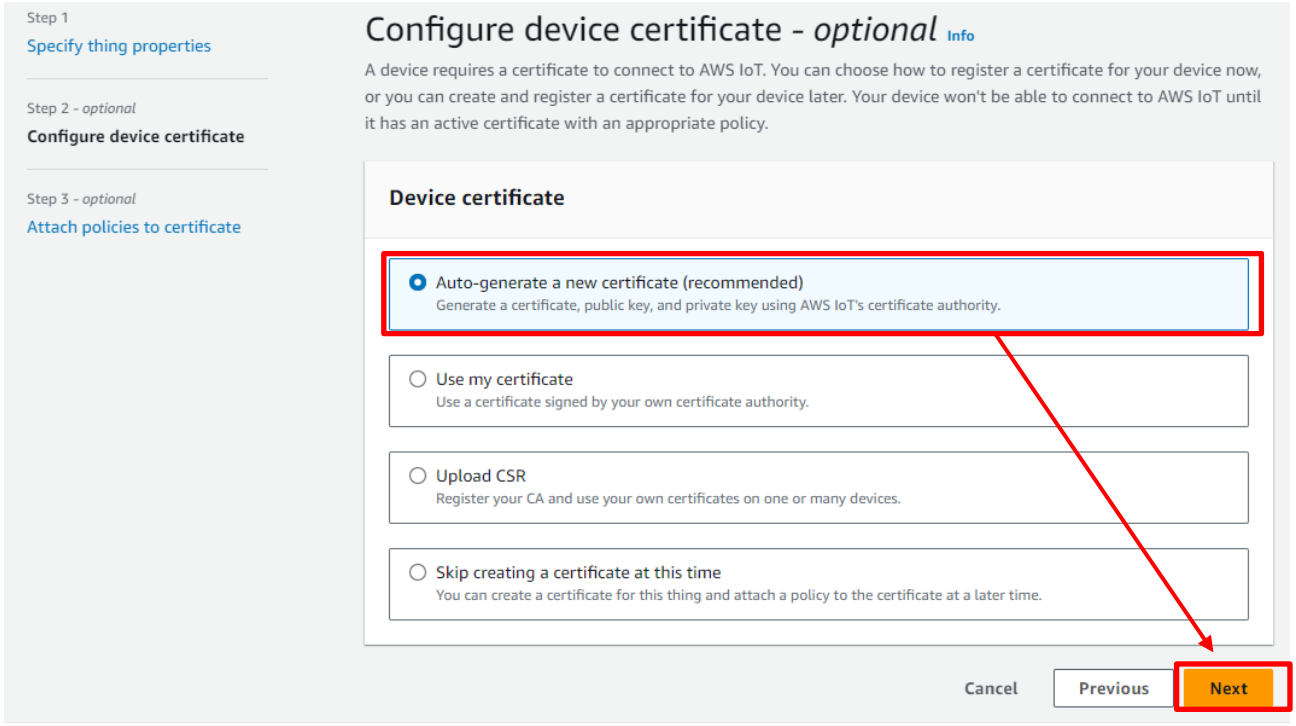


(3) Enter a thing name (example: rx65n_ota_demo_thing), and then click **Next**.

Make a note of the thing name you entered. You will need it in a later process.

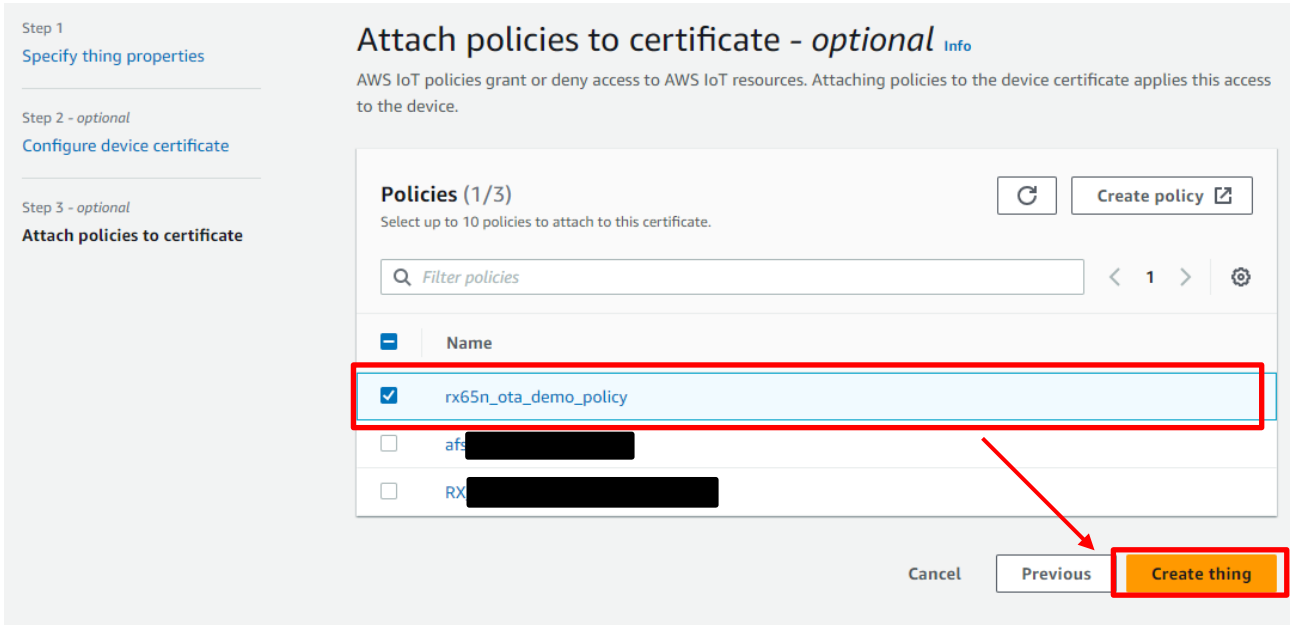


(4) In the Device certificate area, select **Auto-generate a new certificate** and then click **Next**.



(5) Attach the policy to the certificate.

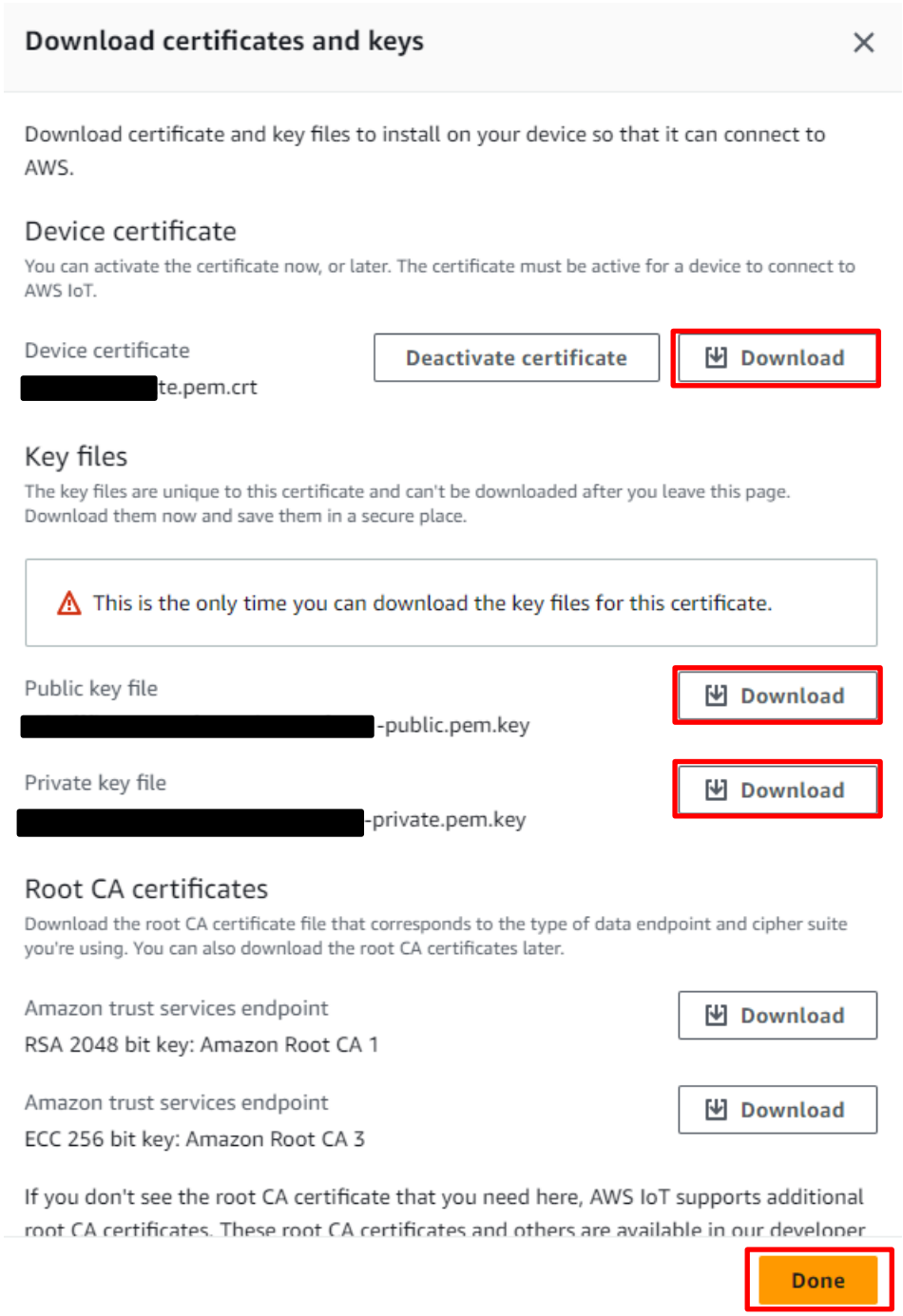
Select the policy you created in 3.3.1 Setting policies, and then click the **Create thing** button.



(6) Download the certificate and key files.

The certificate and private key are equivalent to a password for the device (thing). When you register a certificate and private key on a device, the device can use this certificate and private key to connect to AWS.

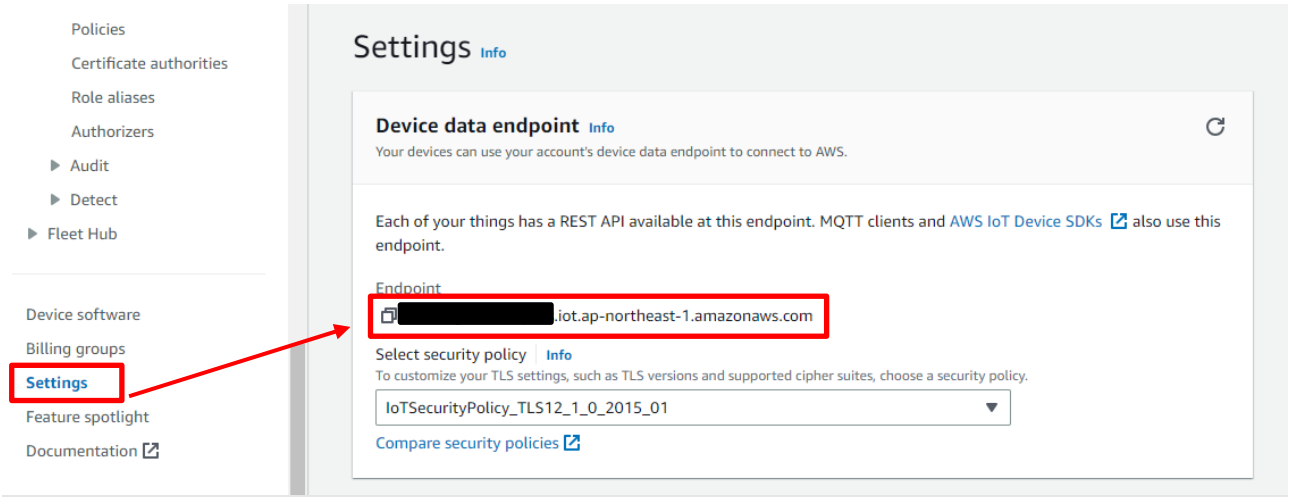
You must download the certificate, public key, and private key now. You will not have another opportunity to download them.



3.3.3 Checking the Endpoint

The endpoint is equivalent to a connection destination (URL) for the device (thing). The device will connect to the endpoint registered for the device.

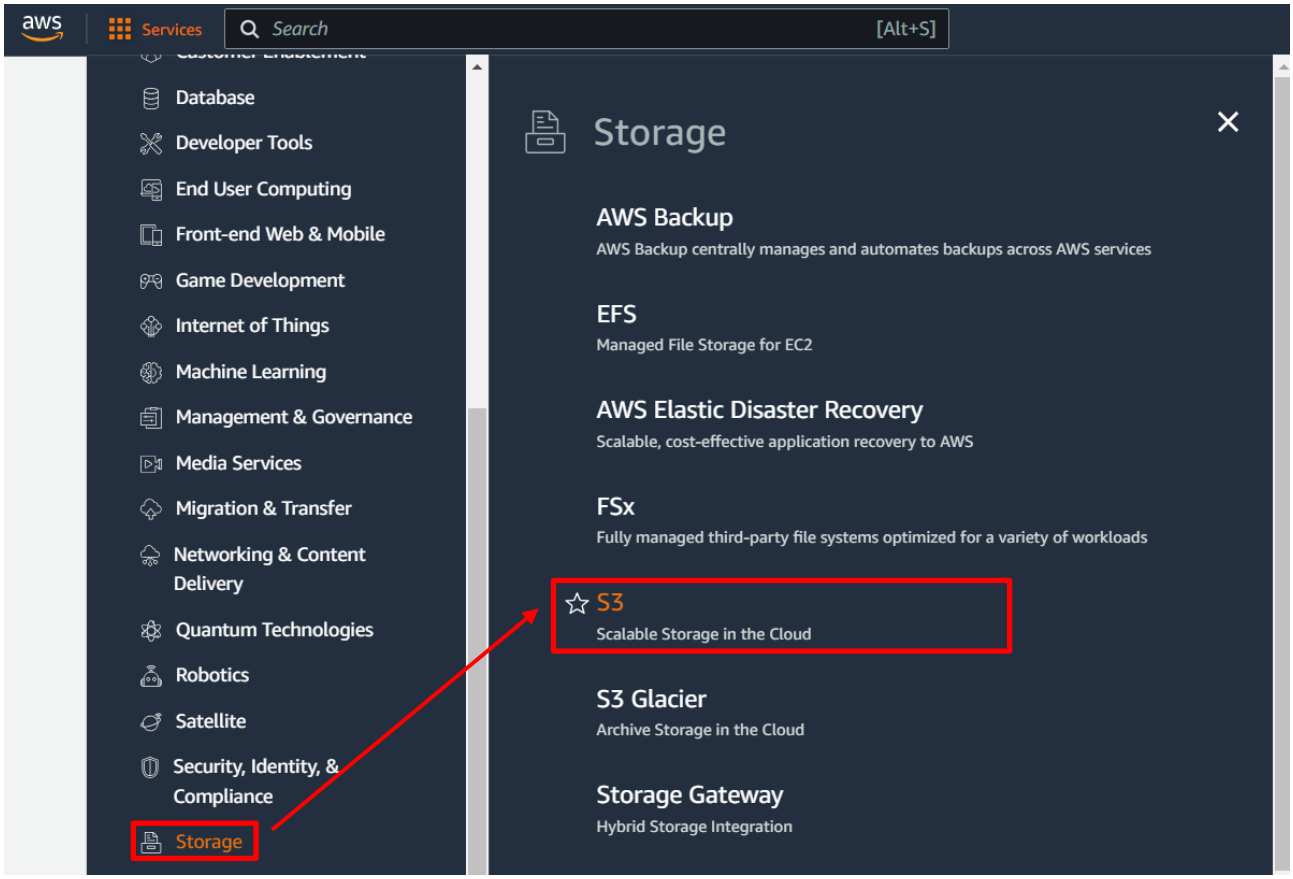
(1) In the menu, click **Settings** and make a note of the endpoint.



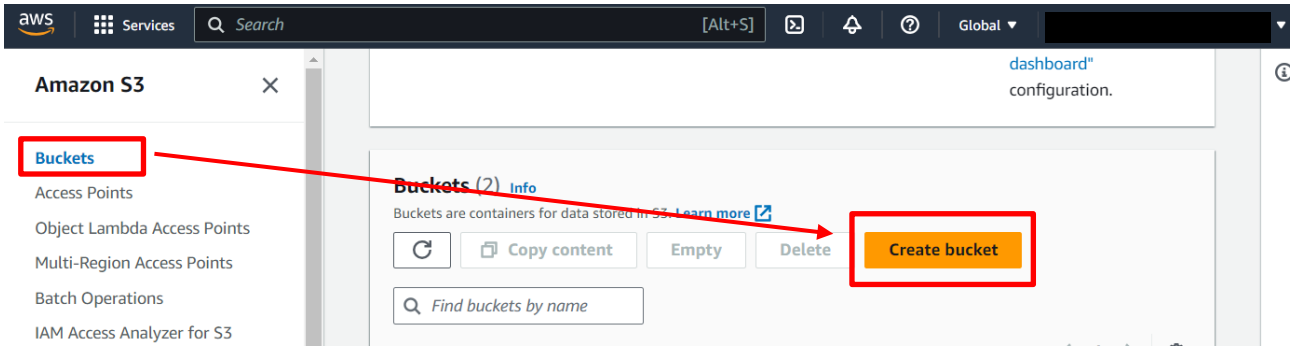
3.4 Creating an Amazon S3 Bucket

Amazon S3 is an online storage web service used to store the firmware with which the device will be updated.

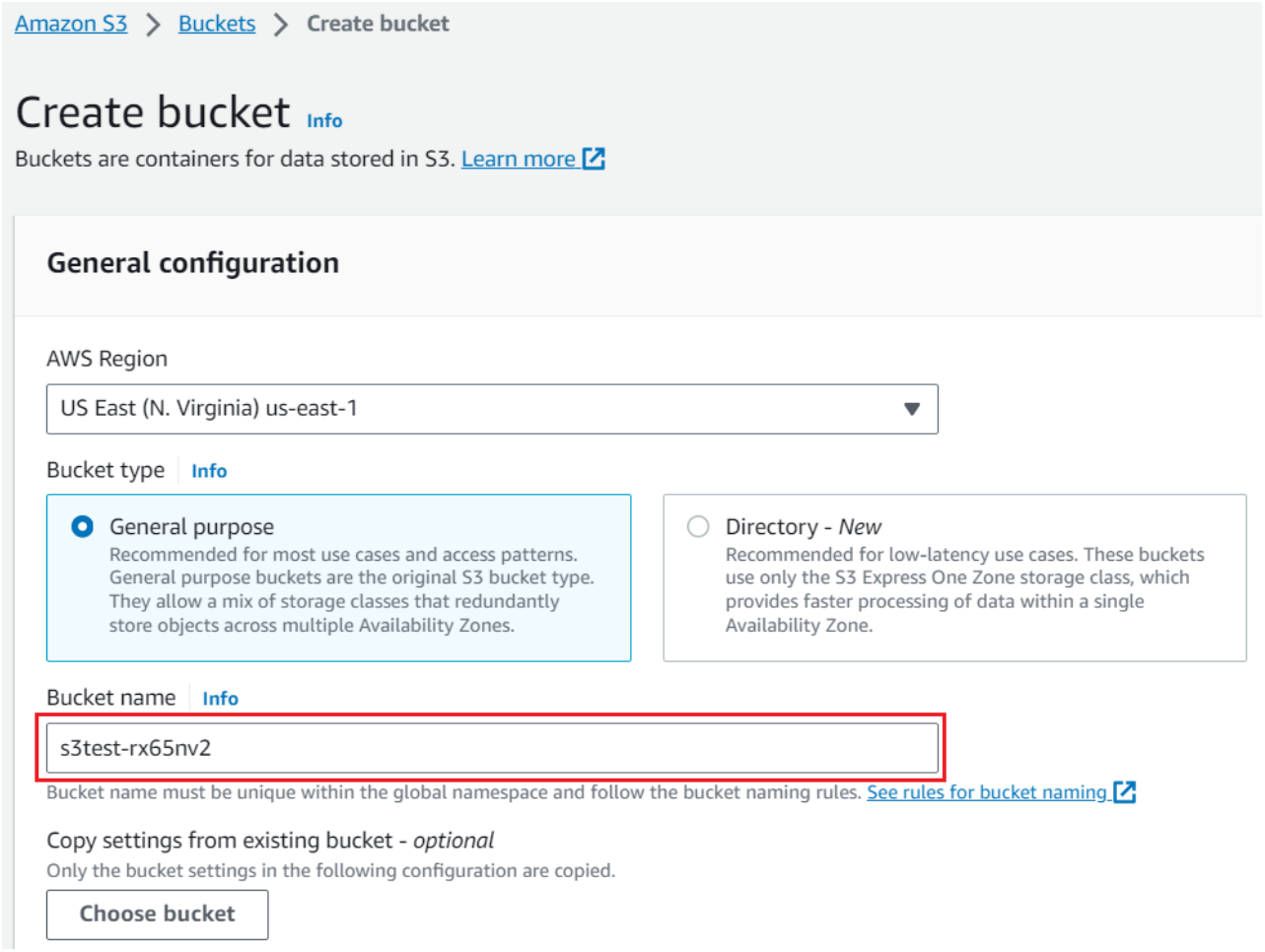
(1) From the **Services** menu, select **Storage** and then **S3**.



(2) On the **Buckets** page, click the **Create bucket** button.



(3) Enter a bucket name. (example: s3test-rx65n)

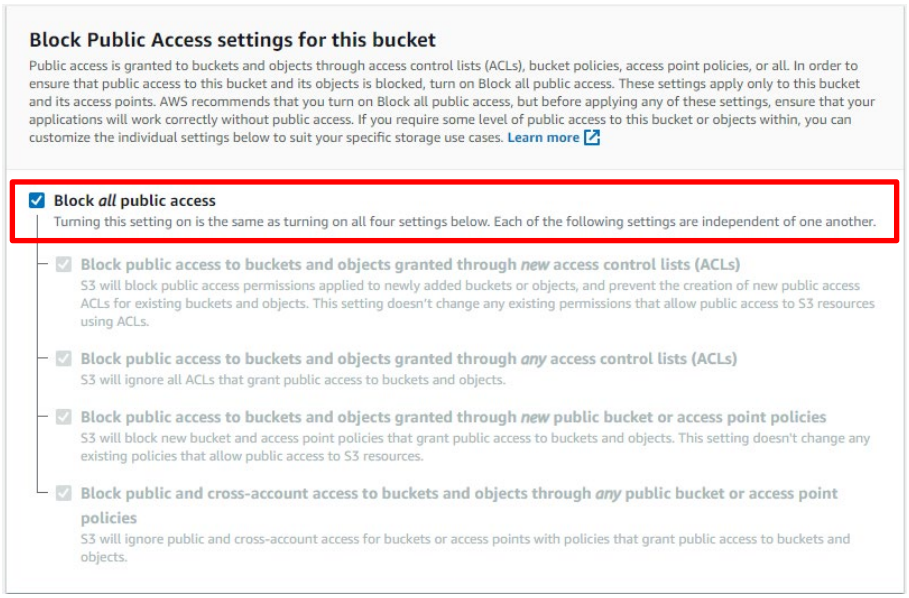


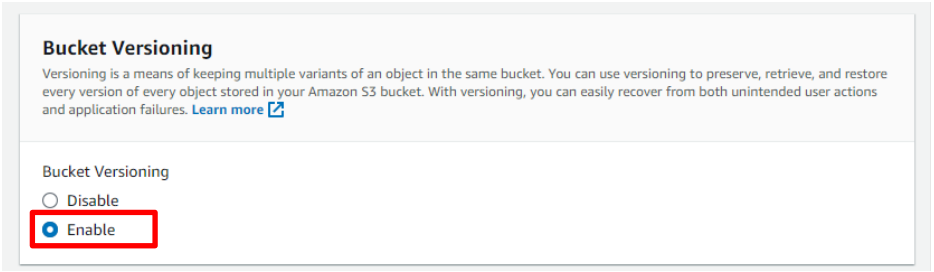
The bucket name must be globally unique. The following error message appears if the bucket name is already in use. In this case, use another name.

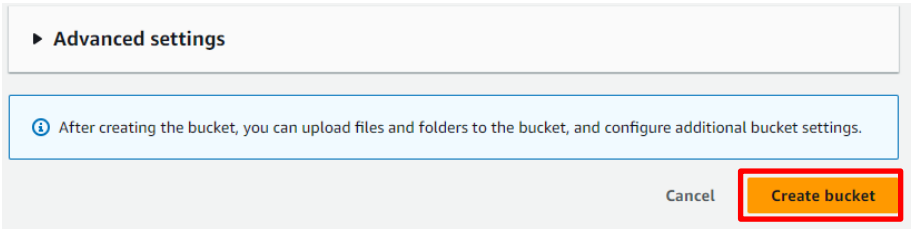
(4) Create the bucket.

Enter the settings as follows, and then click the **Create bucket** button.

- **Block Public Access setting for this bucket**: **Block all public access**
- **Bucket Versioning**: **Enable**



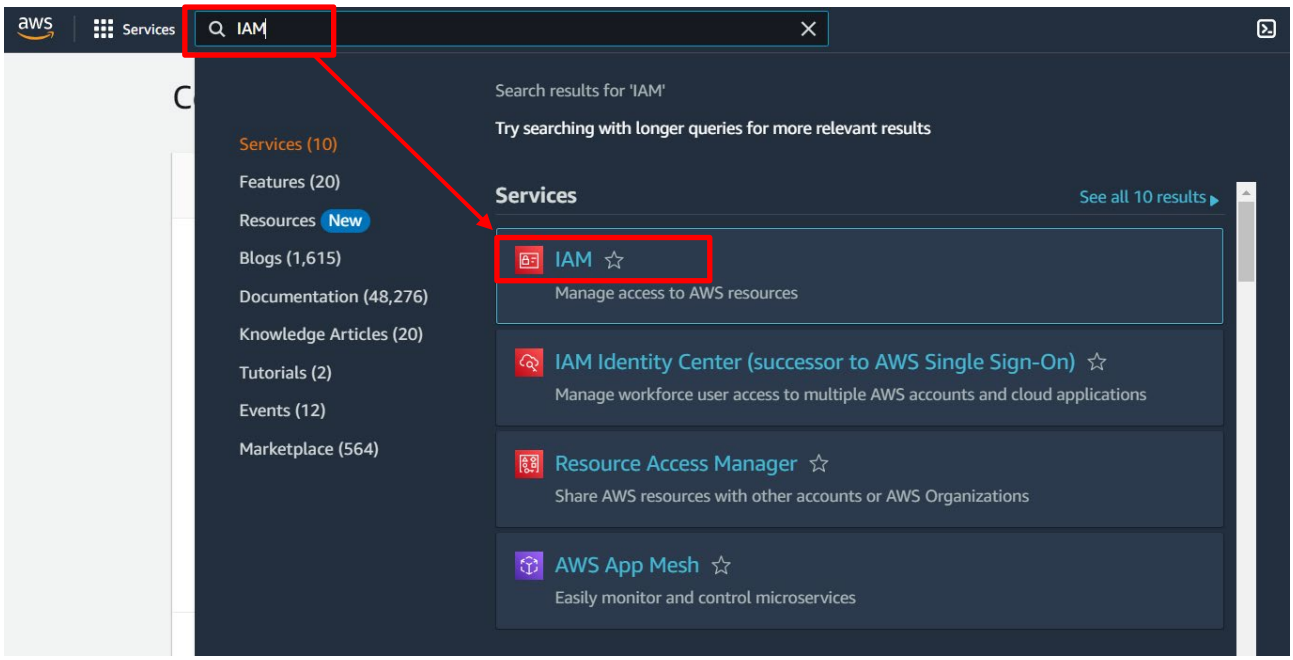




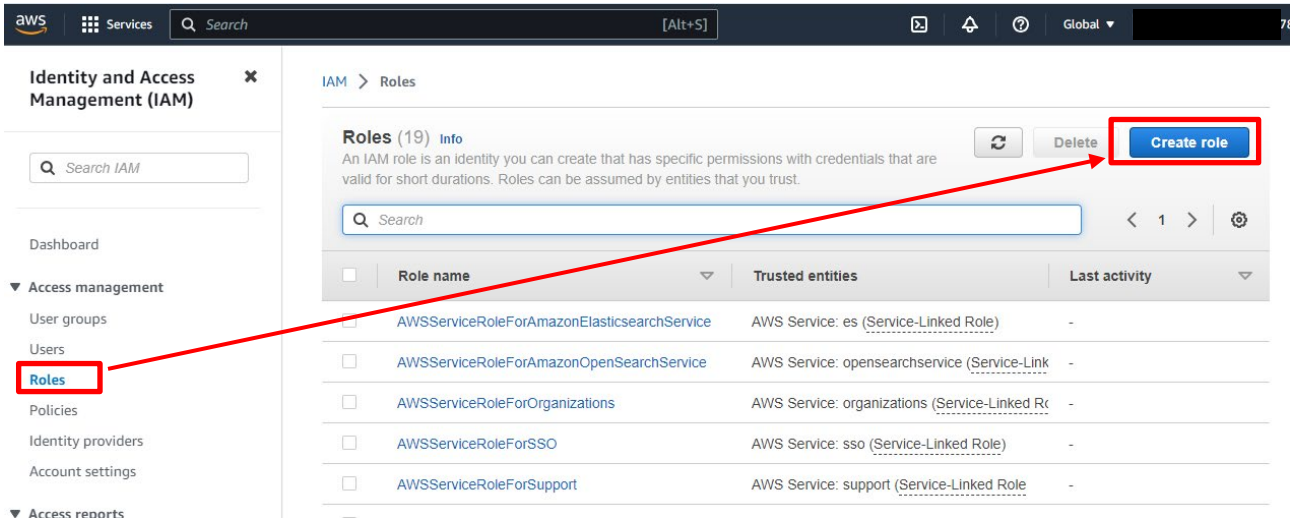
3.5 Allocating OTA Execution Permission to IAM Users

Create a role with the appropriate access permissions to create OTA update jobs.

(1) Enter IAM in the search box at the top of the screen, and click **IAM** in the search results.

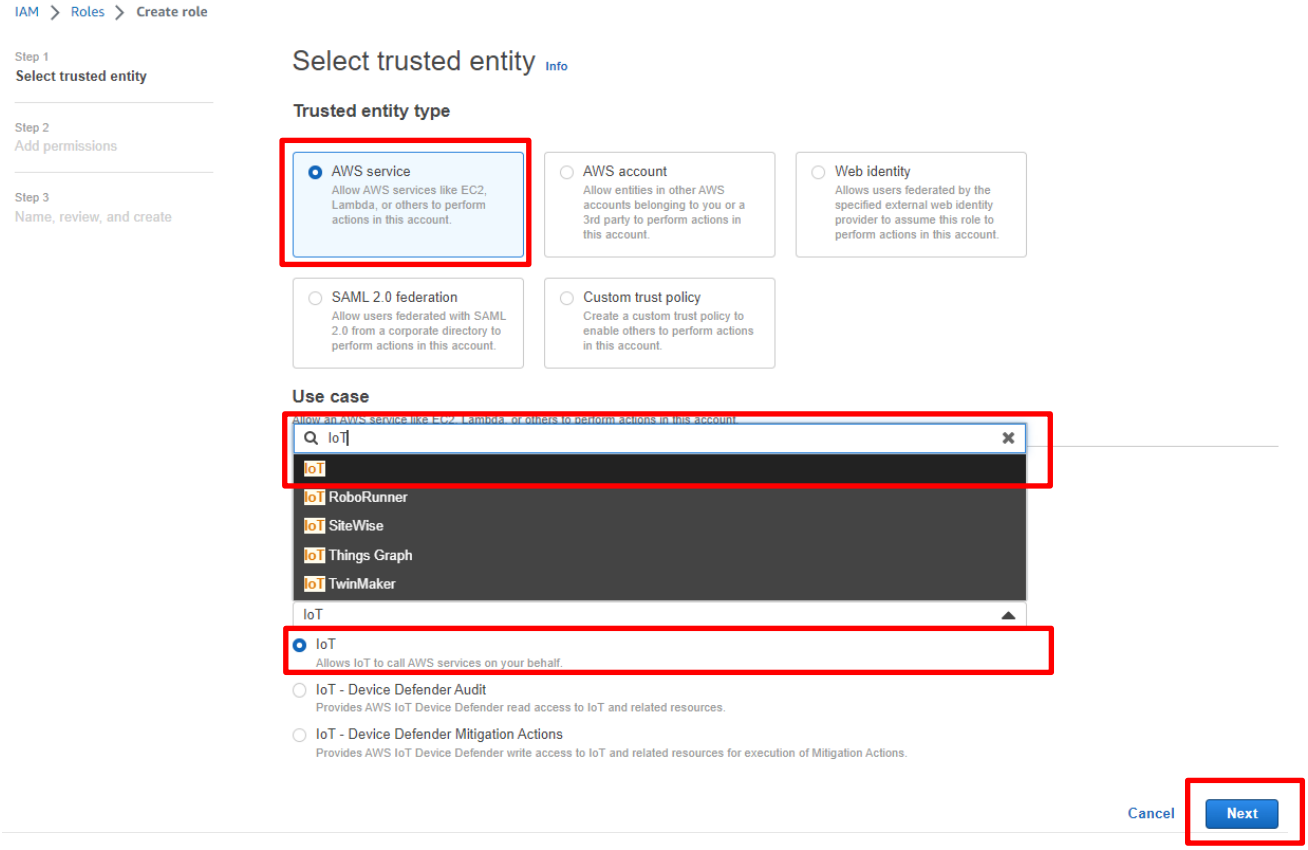


(2) In the menu, click **Roles** and then click the **Create role** button.

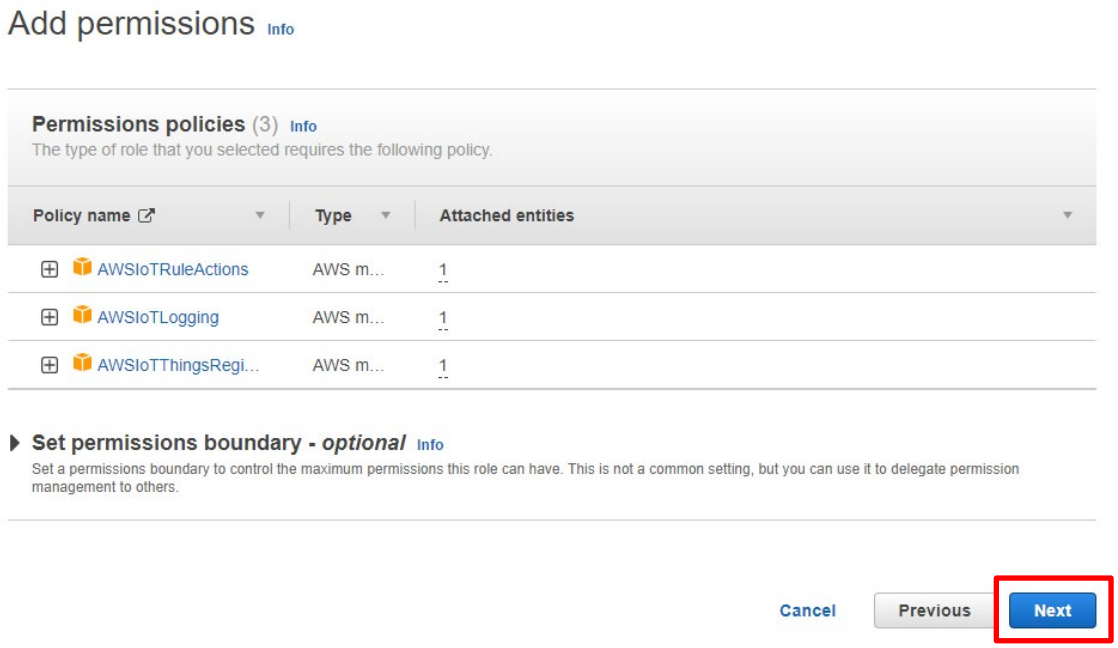


(3) Under **Select trusted entity**, enter the following settings and then click **Next**:

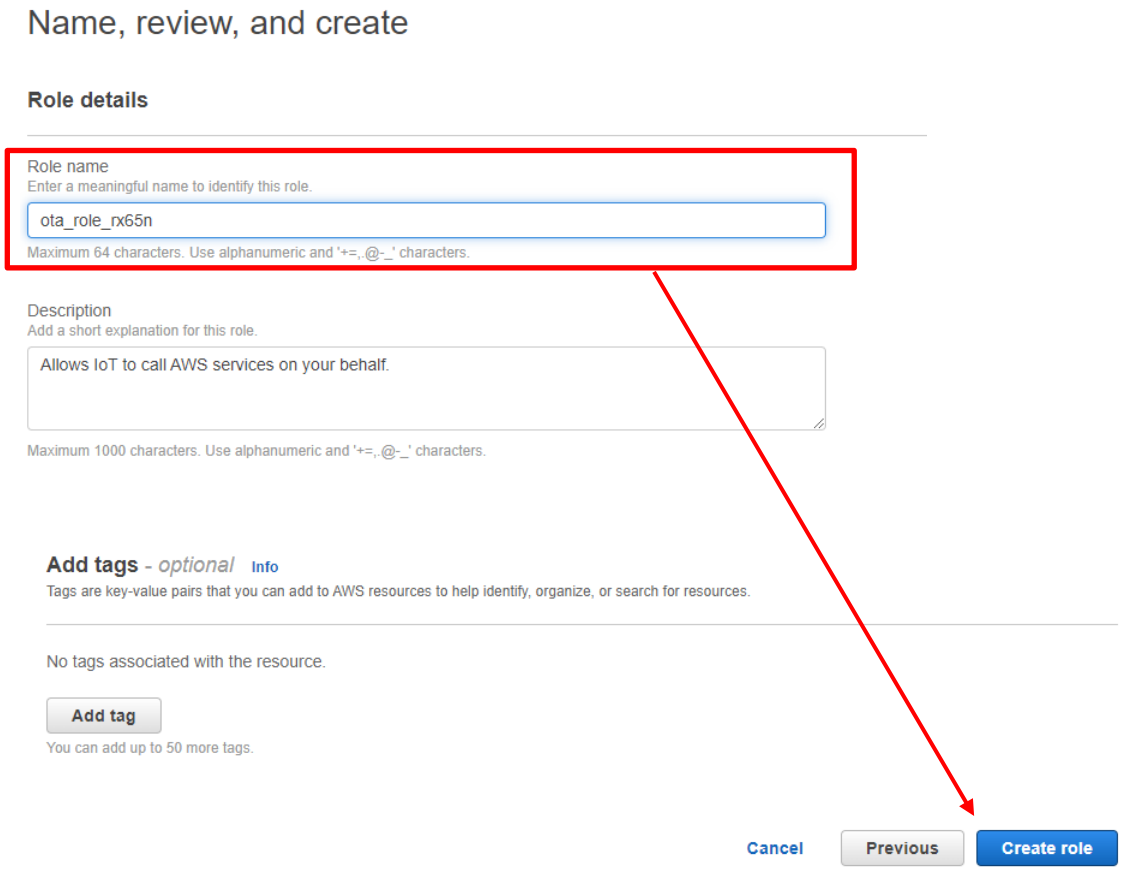
- Under **Trusted entity type**, select **AWS service**.
- Under **Use cases for other AWS services**, select **IoT**.
- Select the **IoT** option button.



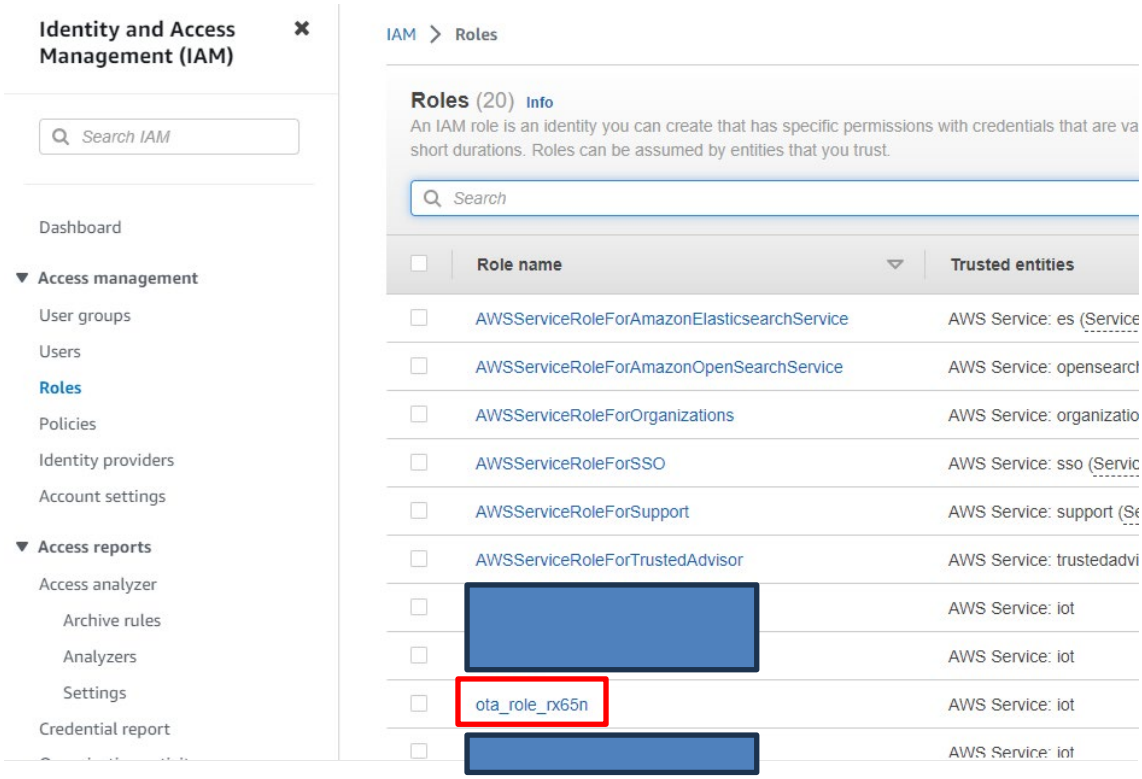
(4) Click **Next** on the **Add permissions** page without making any changes.



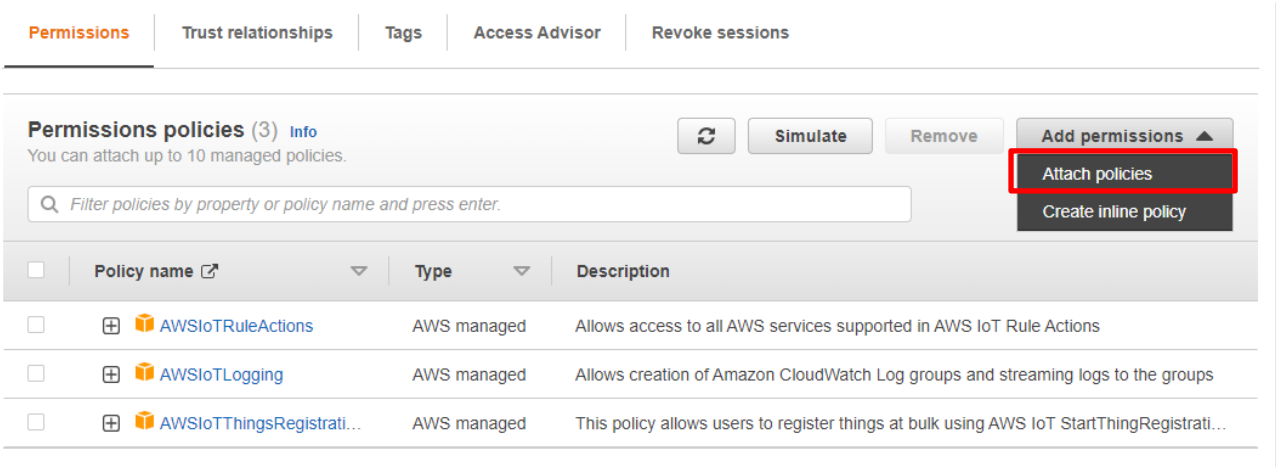
(5) Enter a role name (example: ota_role_rx65n), and then click the **Create role** button.



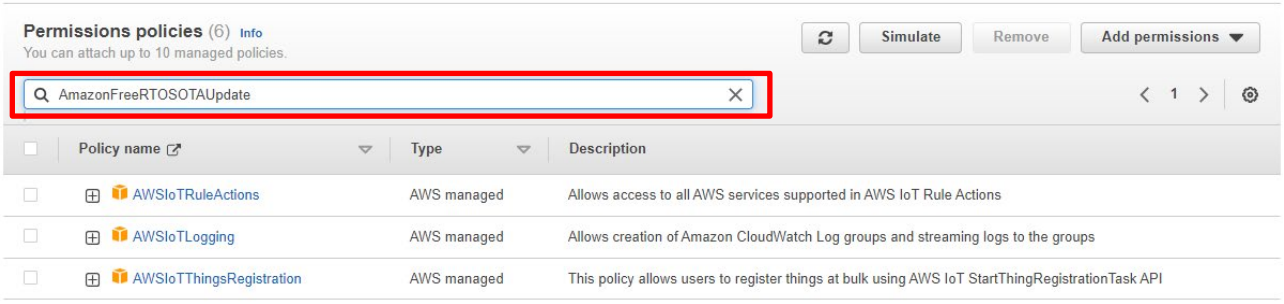
(6) Click the role you created.



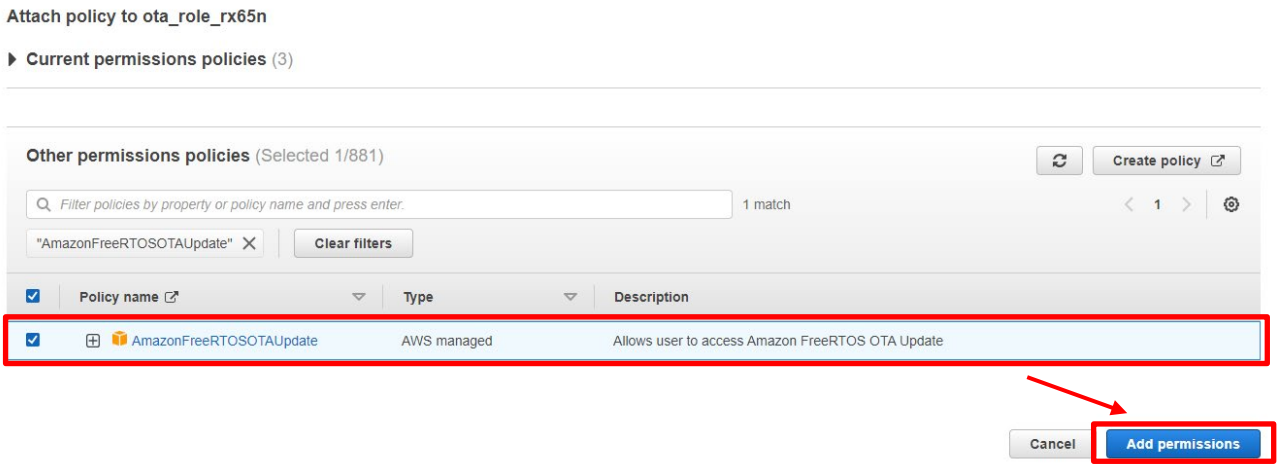
(7) Select **Attach policies**.



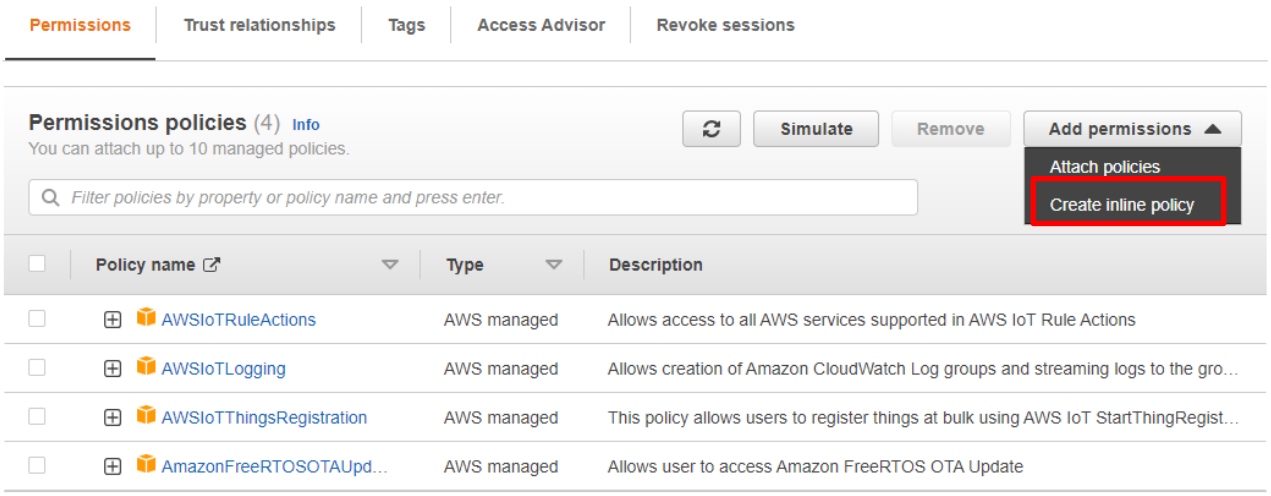
(8) Enter AmazonFreeRTOSOTAUpdate in the **Permissions policies** search box, and then press the **Enter** key.



(9) Select the check box beside the AmazonFreeRTOSOTAUpdate policy, and then click the **Add permissions** button.



(10) From the **Add permissions** drop-down list, select **Create inline policy**.

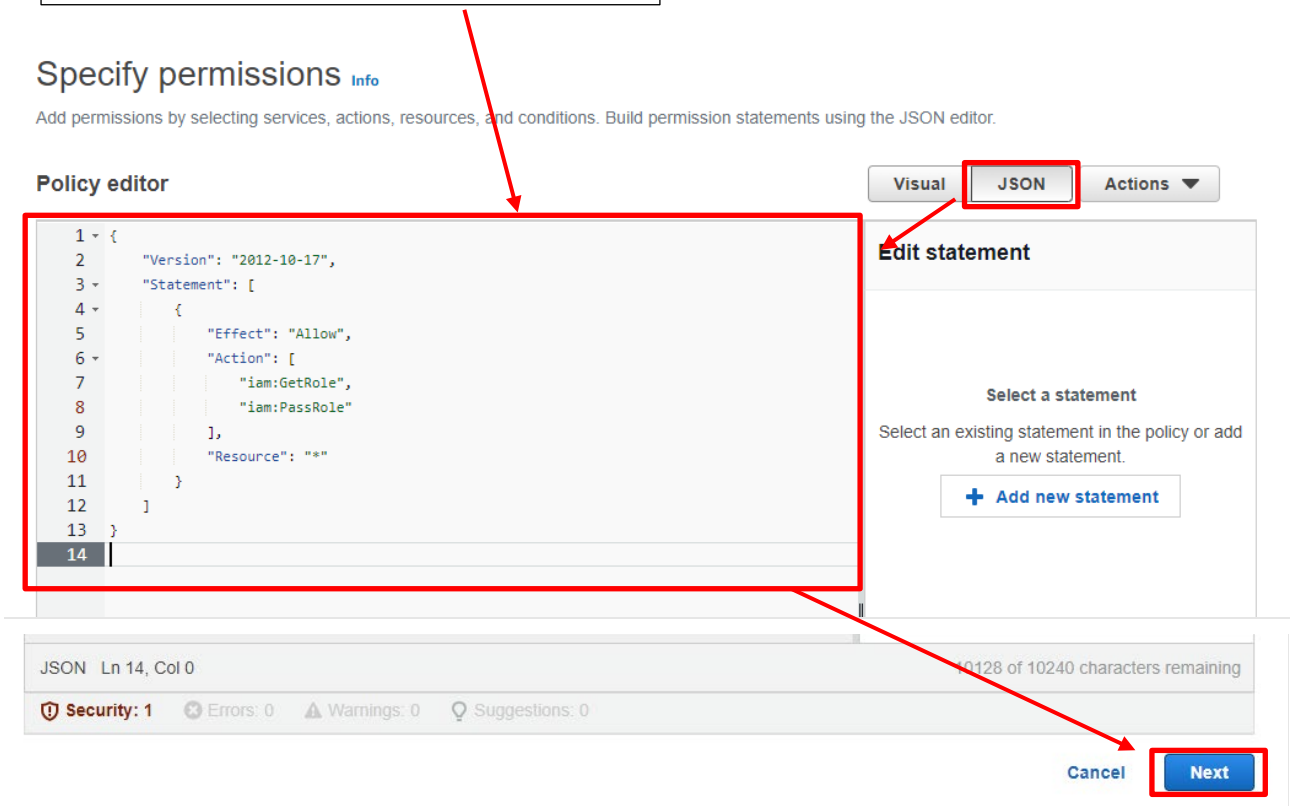


(11) Click **JSON**, paste the following code, and then click **Next**.

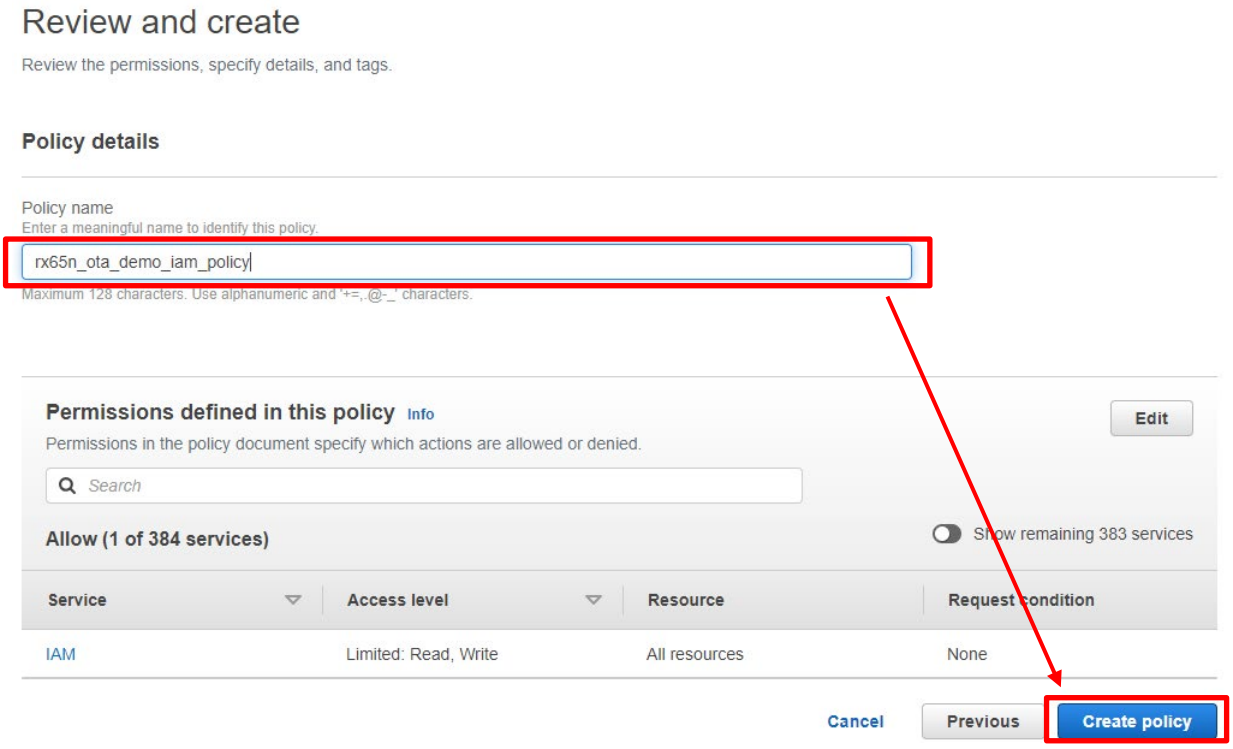
This code grants permission to pass the IAM role to AWS services.

Code to paste:

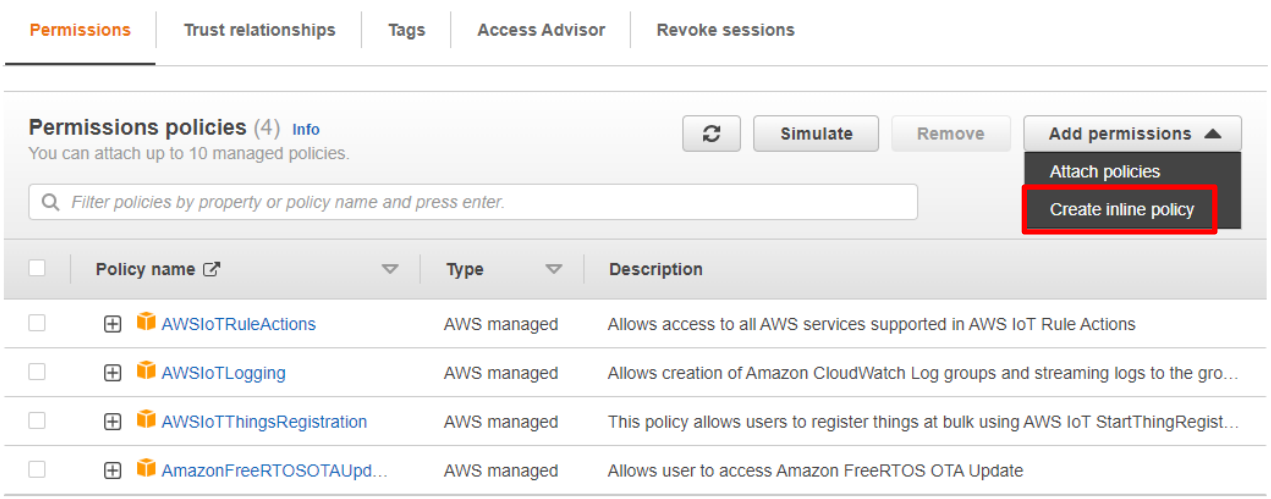
```
{
  "Version": "2012-10-17",
  "Statement": [
    {
        "Effect": "Allow",
        "Action": [
            "iam:GetRole",
            "iam:PassRole"
        ],
        "Resource": "*"
    }
  ]
}
```



(12) Enter a policy name (example: rx65n_ota_demo_iam_policy), and then click the **Create policy** button.



(13) Again, from the **Add permissions** drop-down list, select **Create inline policy**.

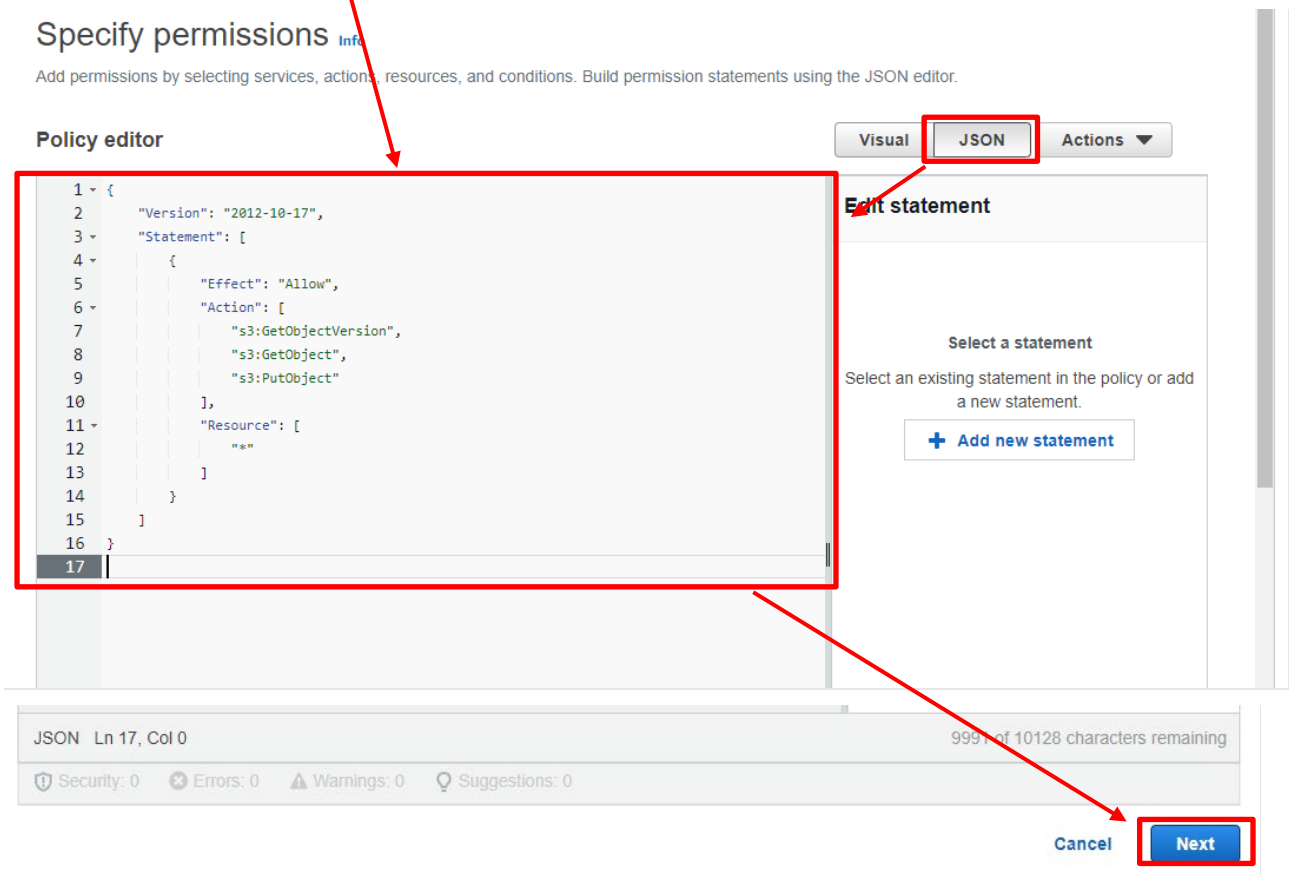


(14) Click **JSON**, paste the following code, and then click **Next**.

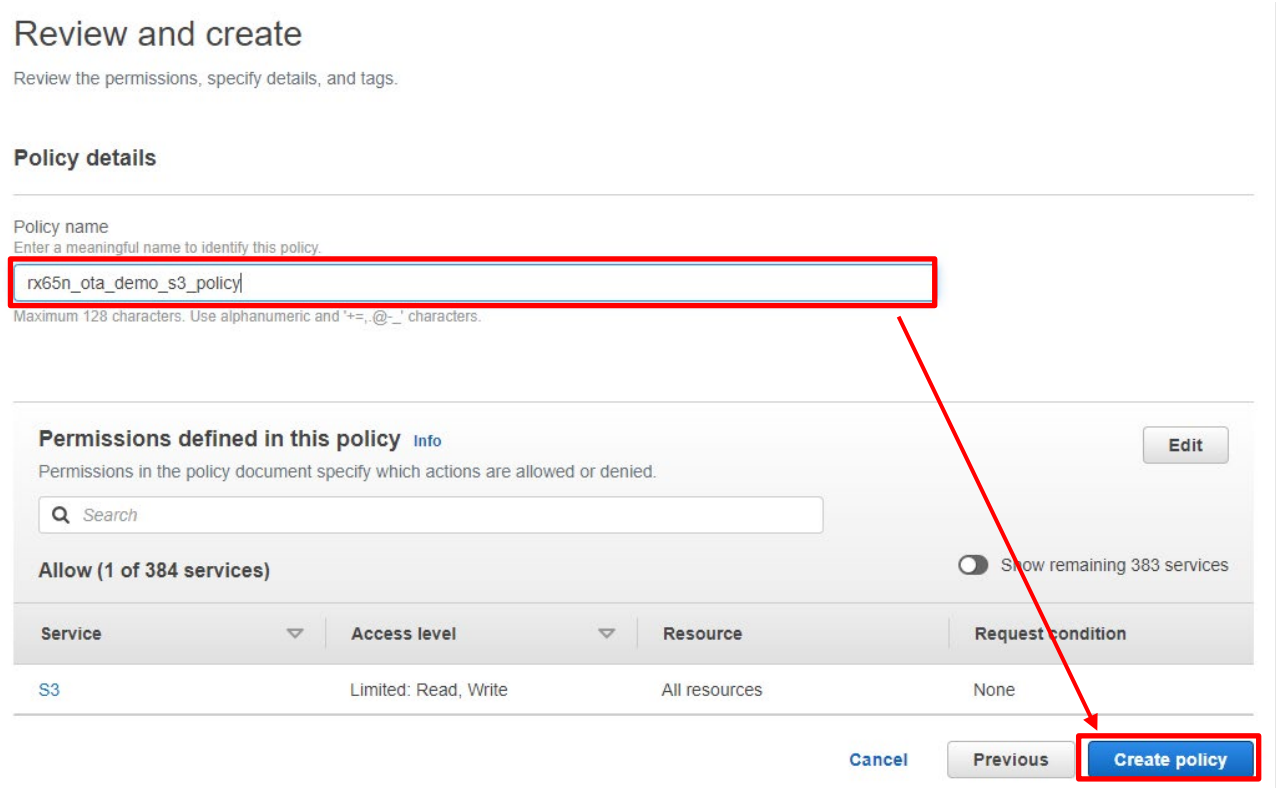
This code allows access to Amazon S3 where the updated firmware is stored.

Code to paste:

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



(15) Enter a policy name (example: rx65n_ota_demo_s3_policy), and then click the **Create policy** button.



4. Setting Up the Device

4.1 Generating Key Pairs and Certificates

(1) From the **Start** menu, open the **Win64 OpenSSL Command Prompt**.

(2) Execute the command to create a CA private key using ECDSA.

Execute the following command:

openssl ecparam -genkey -name secp256r1 -out ca.key

Execution results:

```
C:¥openssl>openssl ecparam -genkey -name secp256r1 -out ca.key
using curve name prime256v1 instead of secp256r1
```

(3) Execute the command to create a CA certificate from the CA private key you created.

Execute the following command: You can enter any character string for **Country Name** onward.

openssl req -x509 -sha256 -new -nodes -key ca.key -days 3650 -out ca.crt

```
C:¥openssl>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
```

Enter any character string for these attributes.

(4) Execute the command to create an ECDSA key pair.

Execute the following command:

openssl ecparam -genkey -name secp256r1 -out secp256r1.keypair

```
C:¥openssl>openssl ecparam -genkey -name secp256r1 -out secp256r1.keypair
using curve name prime256v1 instead of secp256r1
```

(5) Execute the command to create a certificate signing request from the ECDSA key pair you created.

Execute the following command: You can enter any character string for **Country Name** onward. For the last two lines, press **Enter** without entering anything.

openssl req -new -sha256 -key secp256r1.keypair > secp256r1.csr

```
C:¥openssl>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 []:
```

Enter any character string for these attributes.

Press **Enter** without entering anything.

(6) Execute the command to create a certificate from the certificate signing request, CA certificate, and CA private key you created.

Execute the following command:

openssl x509 -req -sha256 -days 3650 -in secp256r1.csr -CA ca.crt -CAkey ca.key -CAcreateserial -out secp256r1.crt

```
C:¥openssl>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, emai
lAddress = Tarou.Renesas@sample.com
Getting CA Private Key
```

(7) Execute the command to extract the private key from the ECDSA key pair.

Execute the following command:

openssl ec -in secp256r1.keypair -outform PEM -out secp256r1.privatekey

```
C:¥openssl>openssl ec -in secp256r1.keypair -outform PEM -out secp256r1.privatekey
read EC key
writing EC key
```

(8) Execute the command to extract the public key from the ECDSA key pair.

Execute the following command:

openssl ec -in secp256r1.keypair -outform PEM -pubout -out secp256r1.publickey

```
C:¥openssl>openssl ec -in secp256r1.keypair -outform PEM -pubout -out secp256r1.publickey
read EC key
writing EC key
```

4.2 Creating the Initial Version of the Firmware

The following explains how to create the initial version of the firmware.

4.2.1 Creating the Sample Projects

Follow the procedure below to create the OTA sample project and Secure Bootloader sample project needed to implement OTA using Amazon Web Services, as described in 4.2.2.

If you wish to use the import function to run the demo project, refer to the instructions in Getting Started Guide.

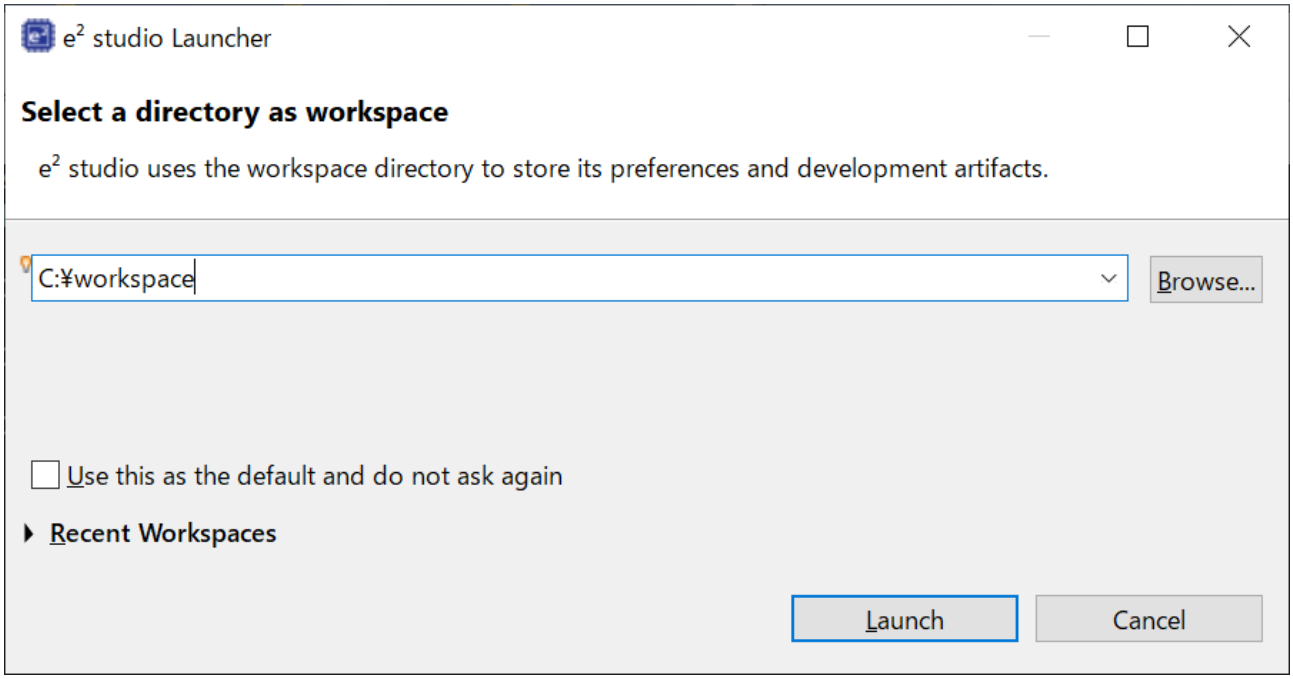
(1) Create a workspace in e² studio.

Launch e² studio and create a new workspace.

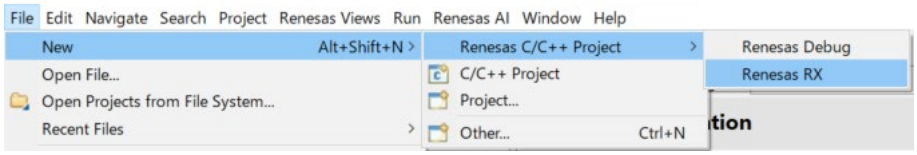
Keep the names of the workspace and the project files as short as possible. If the full path to the files at the lowest level of the directory structure exceeds 256 bytes, an error will occur when you build the project.

Errors may also occur if the file path contains Japanese characters, so make sure the name you enter contains only alphanumeric characters.

Example: Creating a workspace in location **C:\workspace**



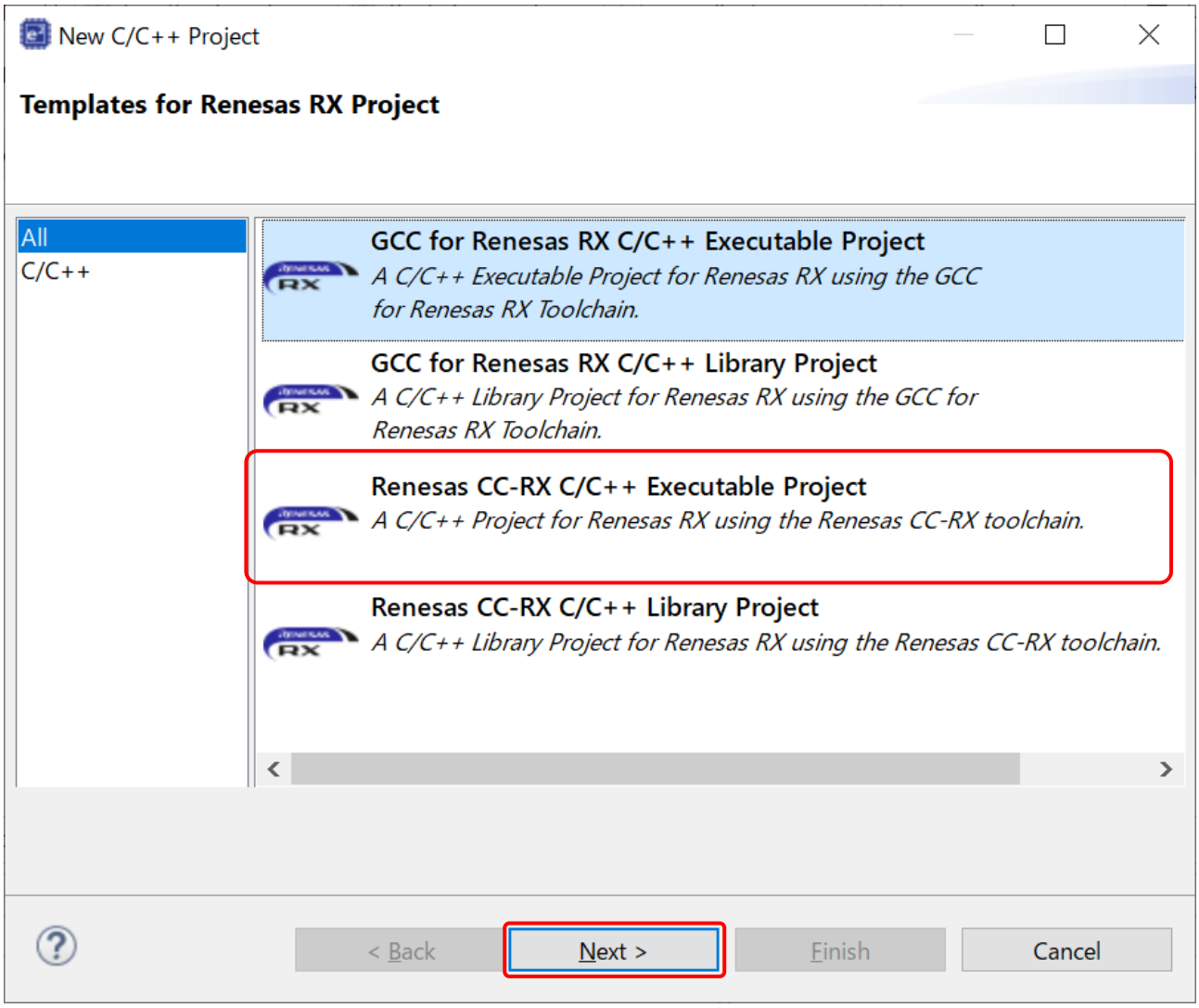
After launching e² studio, from the **File** menu select **New** > **Renesas C/C++ Project** > **Renesas RX** to display the **New C/C++ Project** dialog box.



In the **New C/C++ Project** dialog box, select the type of project to be created. In this case, select **All**, then select **Renesas CC-RX C/C++ Executable Project** and click the **Next** button.

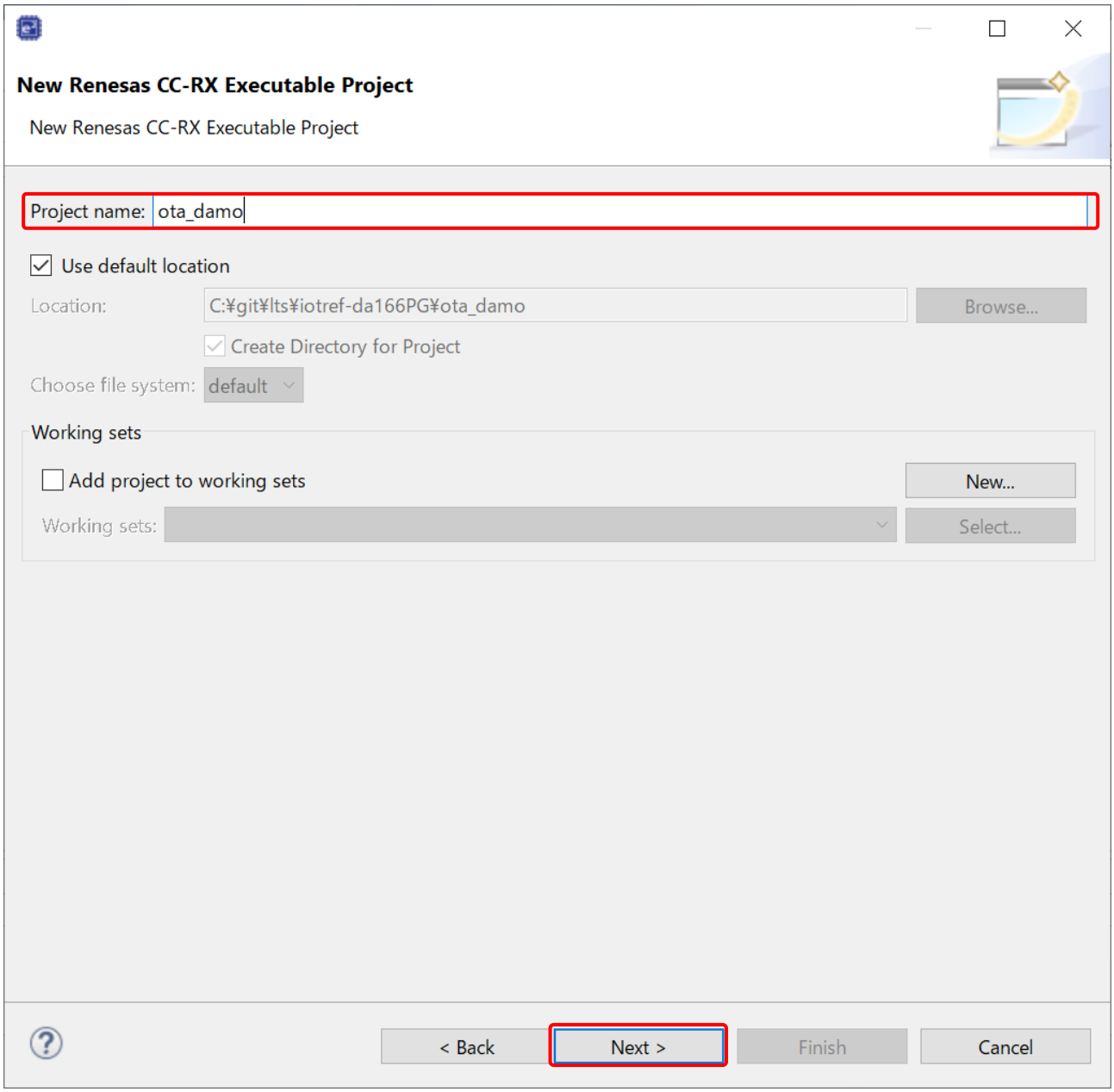
Selecting the project type opens the **New Renesas CC-RX Executable Project** dialog box.

To use GCC, select **GCC for Renesas RX C/C++ Executable Project** as the project type.



(2) Create the OTA sample project.

Here you will specify the project name. Enter **ota_demo** as the project name and click the **Next** button. The **Select toolchain, device & debug settings** dialog box opens.



Configure the toolchain, device, and debug settings to use for the project.

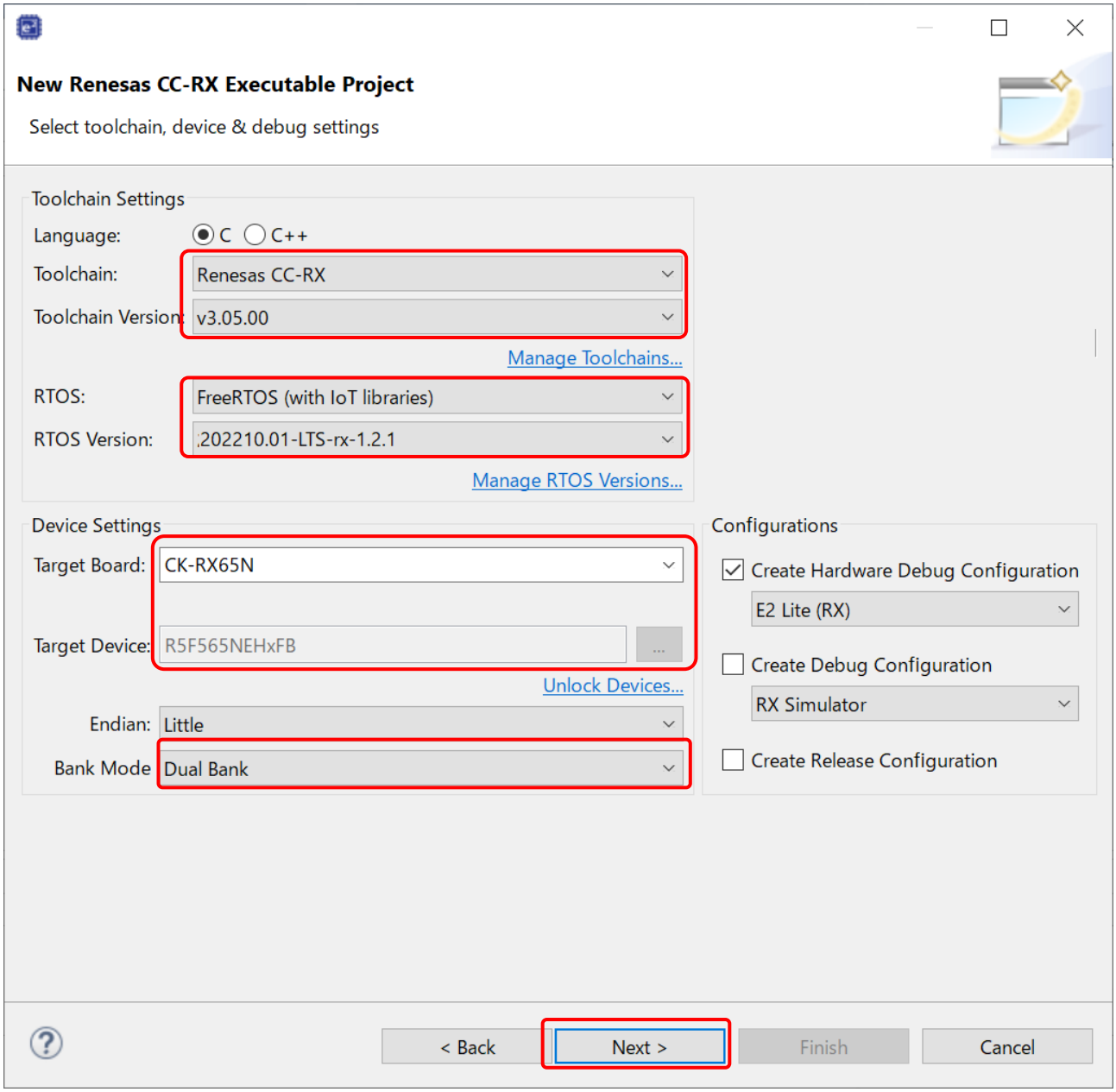
The setting for **Toolchain** is pre-selected based on the type of project. To change the toolchain version, select the version of your choice from the drop-down list next to **Toolchain Version**.

For RTOS, select **Free RTOS (with IoT libraries)**, and for **RTOS Version** select **202210.01-LTS-rx-1.2.1**. If you are running e^2 studio for the first time or if the desired version does not appear in the list, click **Manage RTOS Versions...** to display the **RTOS Module Download** dialog box, check the box next to the desired version, and click the **Download** button to download it.

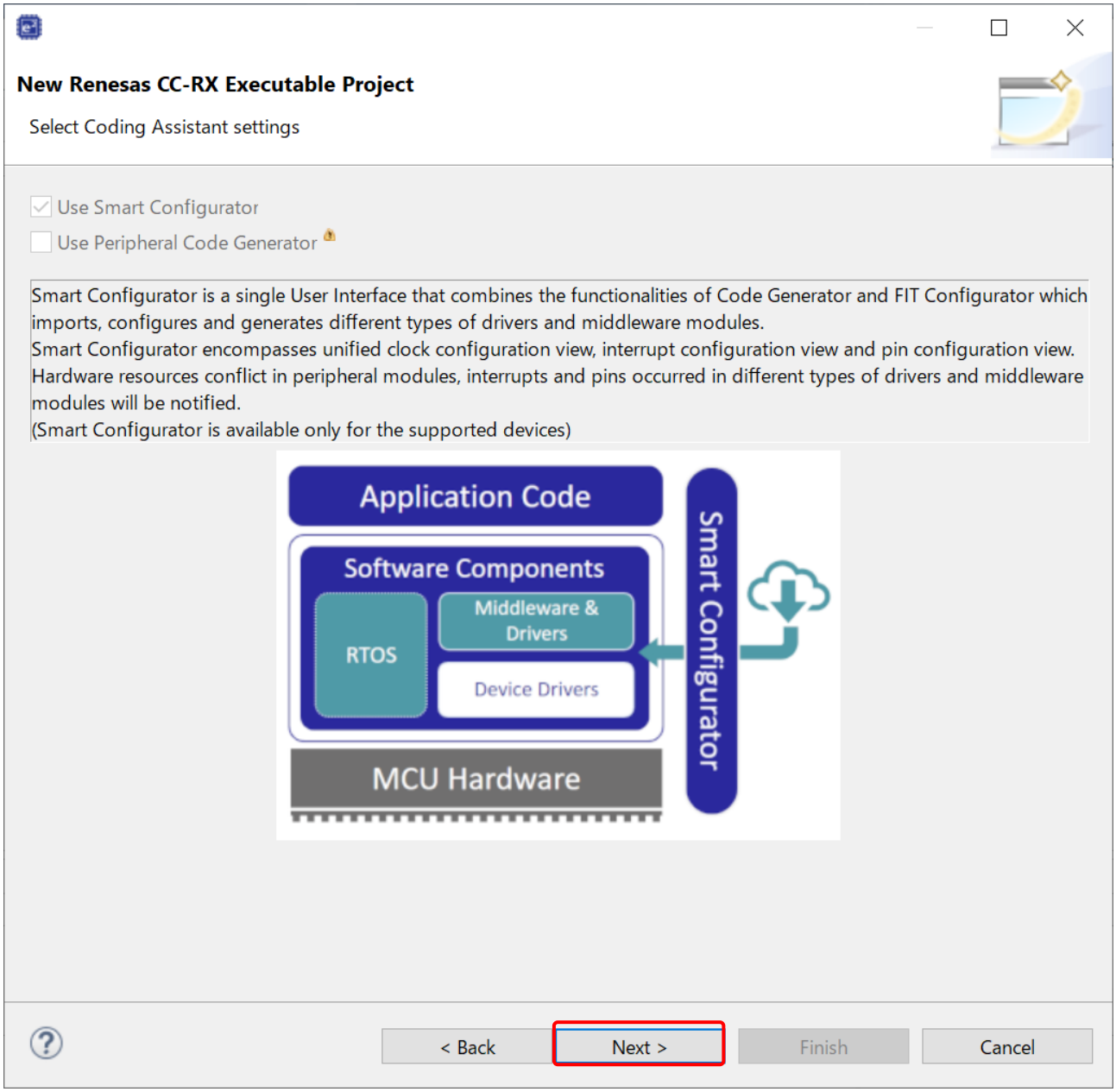
For **Target Board**, select **CK-RX65N**. (The setting for **Target Device** is selected automatically.)

For **Bank Mode**, select **Dual Bank**. To implement OTA, it is necessary to select the dual bank mode setting in which the code flash memory of the device is treated as two banks.

After all the settings have been configured, click the **Next** button.

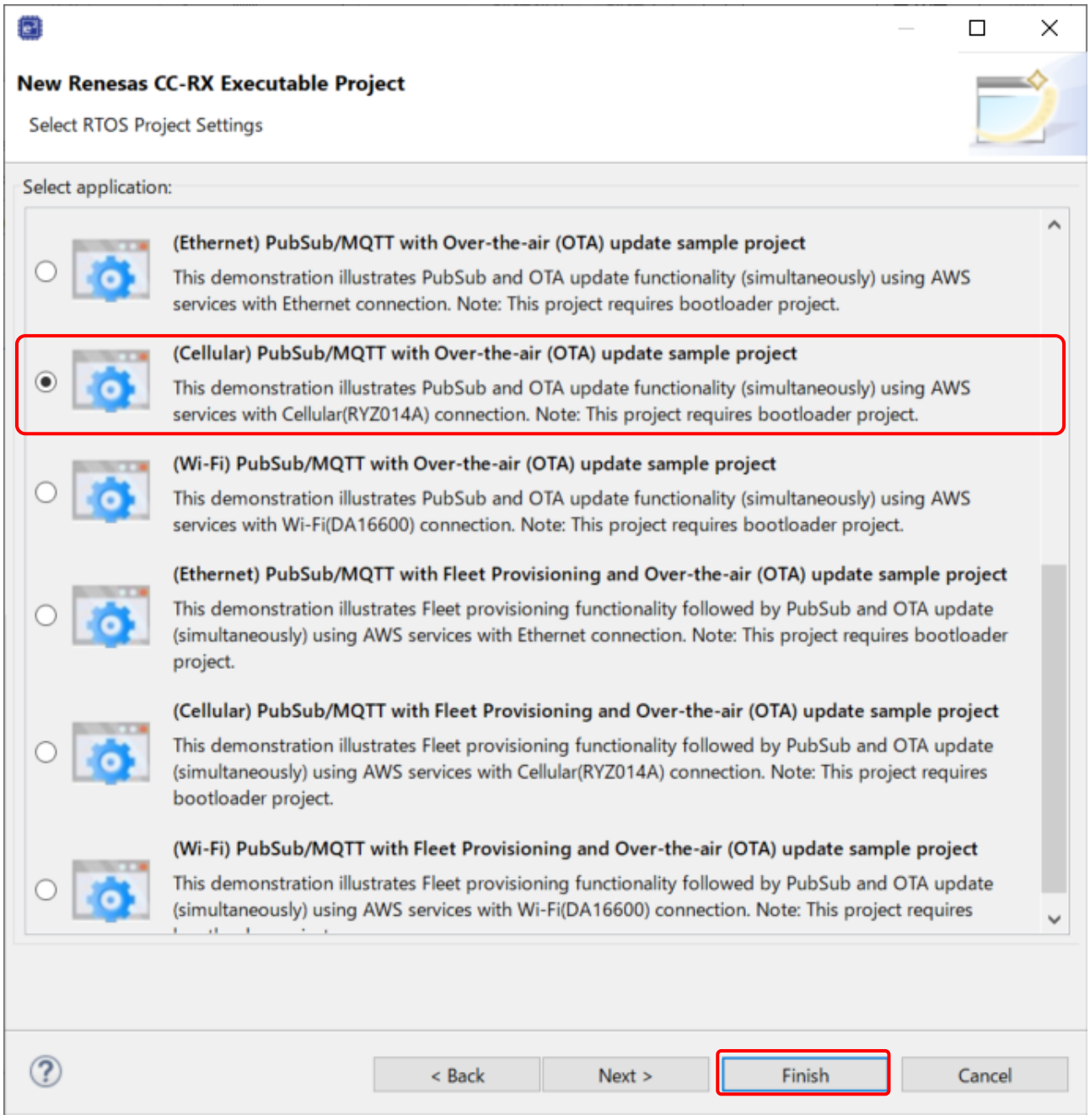


When the **Select Coding Assistant settings** dialog box appears, click the **Next** button without changing any settings.

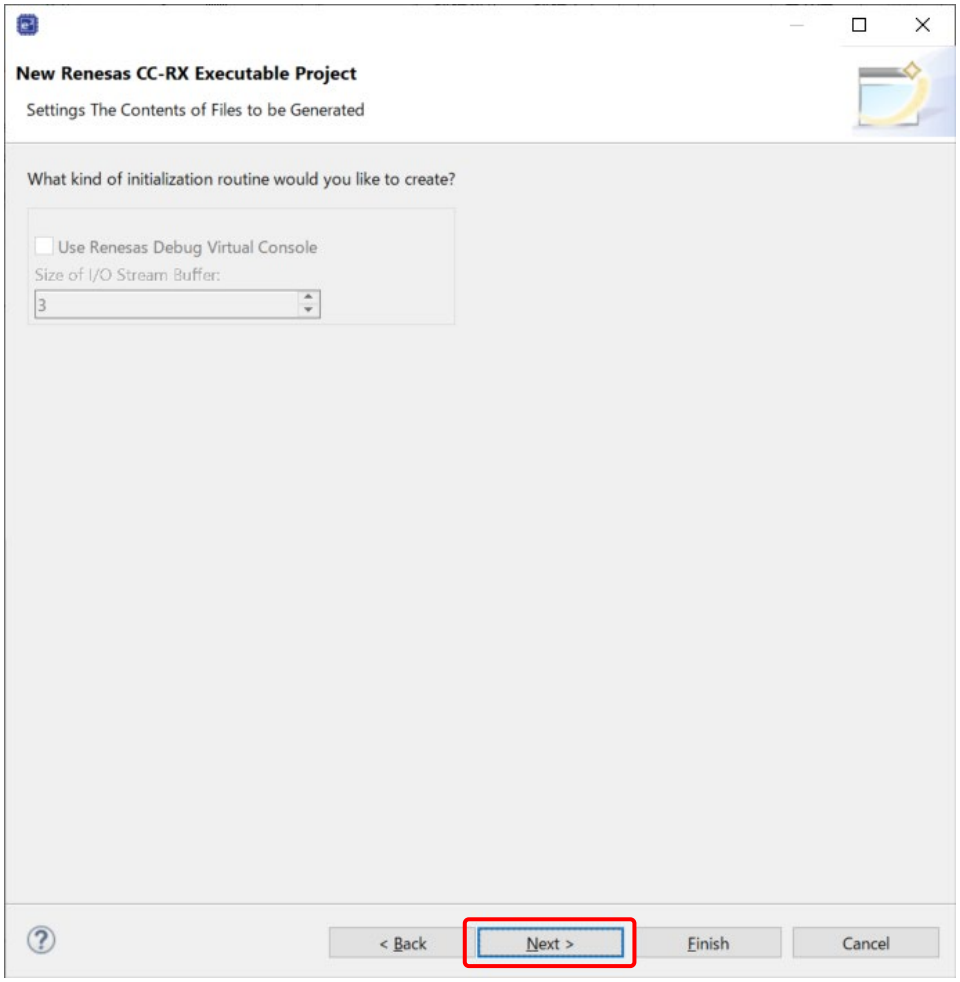


A list of sample projects is displayed in the **Select RTOS Project Settings** dialog box.

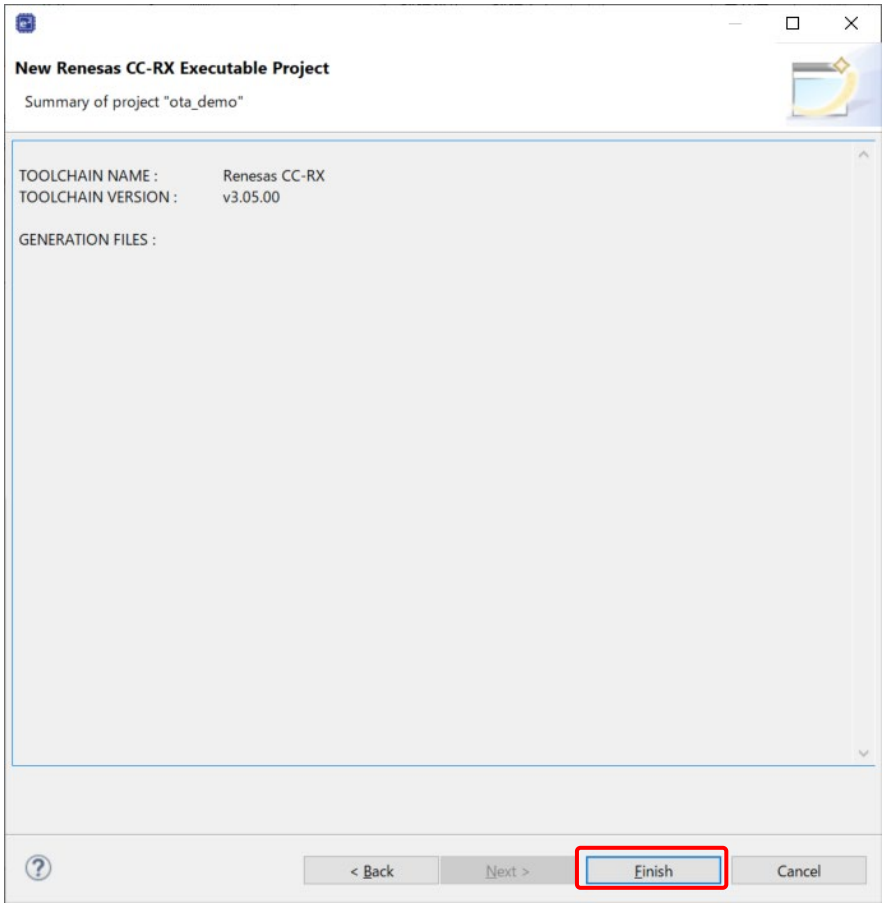
Use the scroll bar to scroll down the list, select **(Cellular) PubSub/MQTT with Over-the-air (OTA) update sample project**, and click the **Next** button.



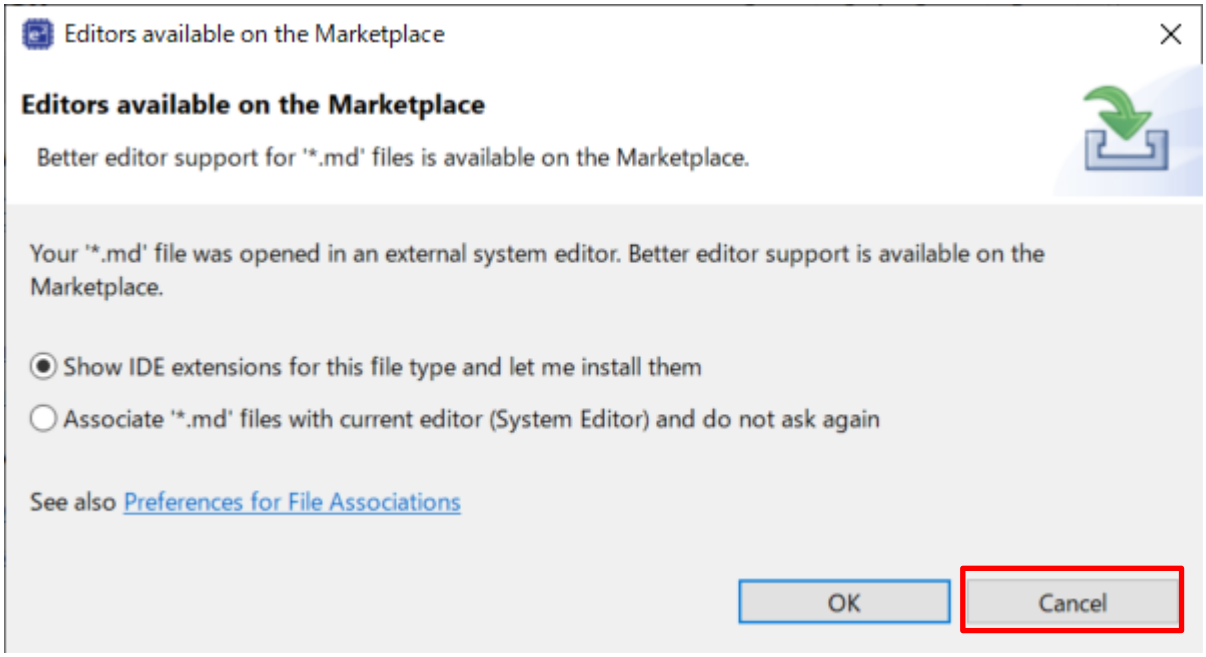
When the **Settings The Contents of Files to be Generated** dialog box appears, click the **Next** button without changing any settings.



A dialog box appears indicating that the project has been created. If everything appears to be in order, click the **Finish** button.

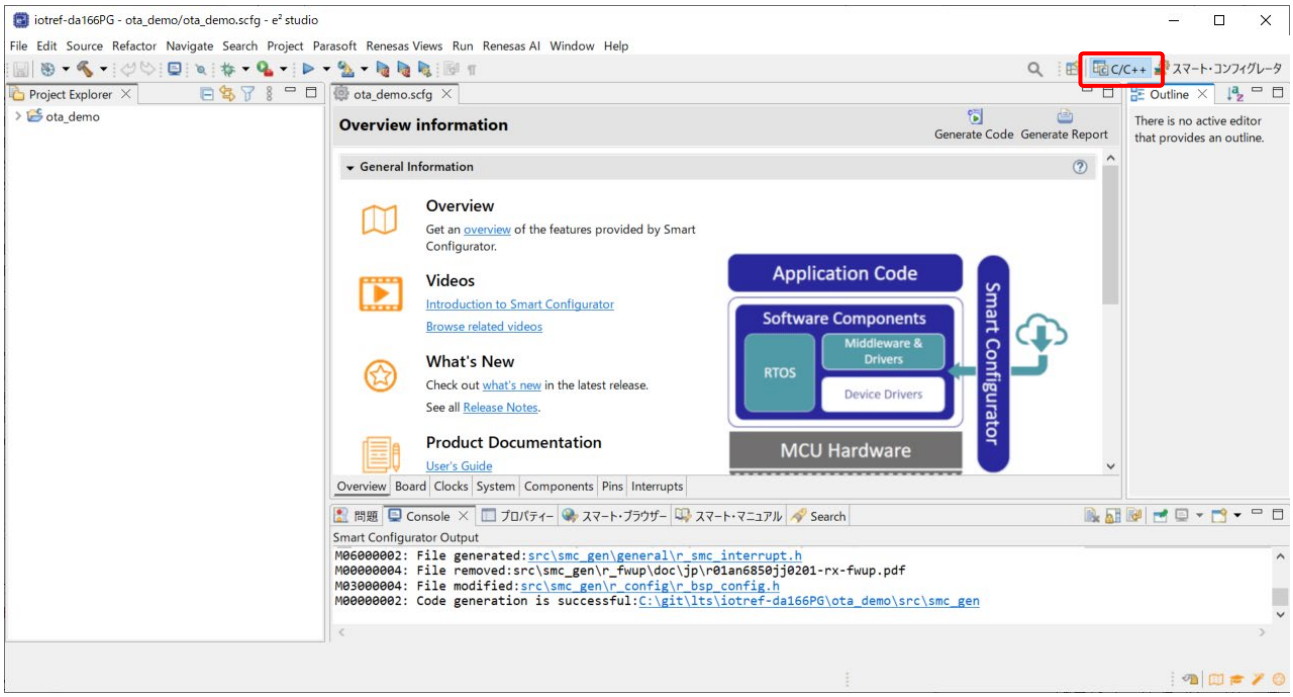


When the **Editors available on the Marketplace** dialog box appears, click the **Cancel** button to close it.



This completes the procedure for creating the project in e^2 studio.

If Project Explorer is not visible, click the button for the **C/C++** perspective in the upper right corner of the window, then select **Window** > **Show View** > **Project Explorer**.

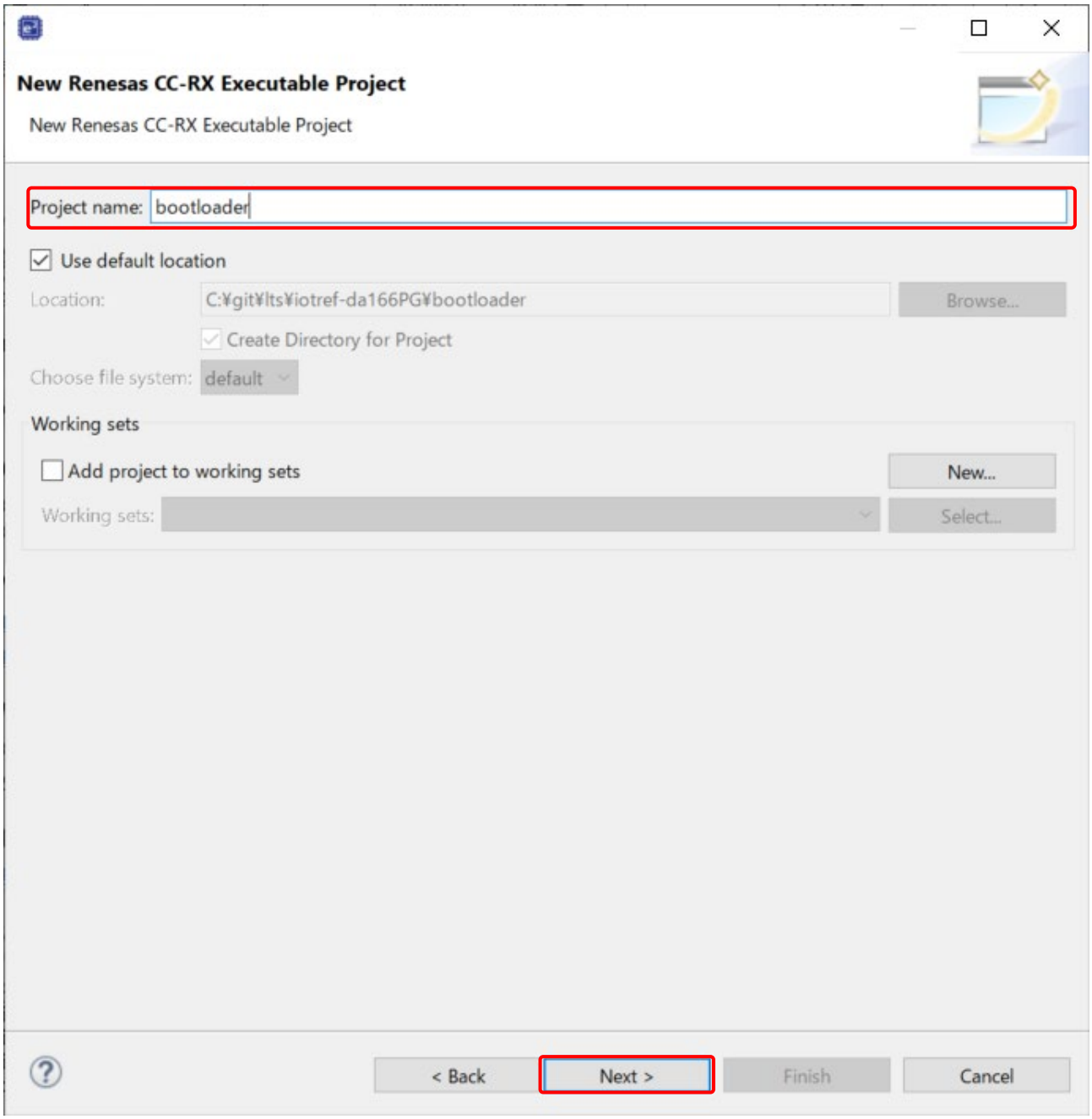


In the discussion that follows, replace the project name **aws_ryz014a_ck_rx65n** with **ota_demo** as appropriate.

(3) Create the Secure Bootloader sample project.

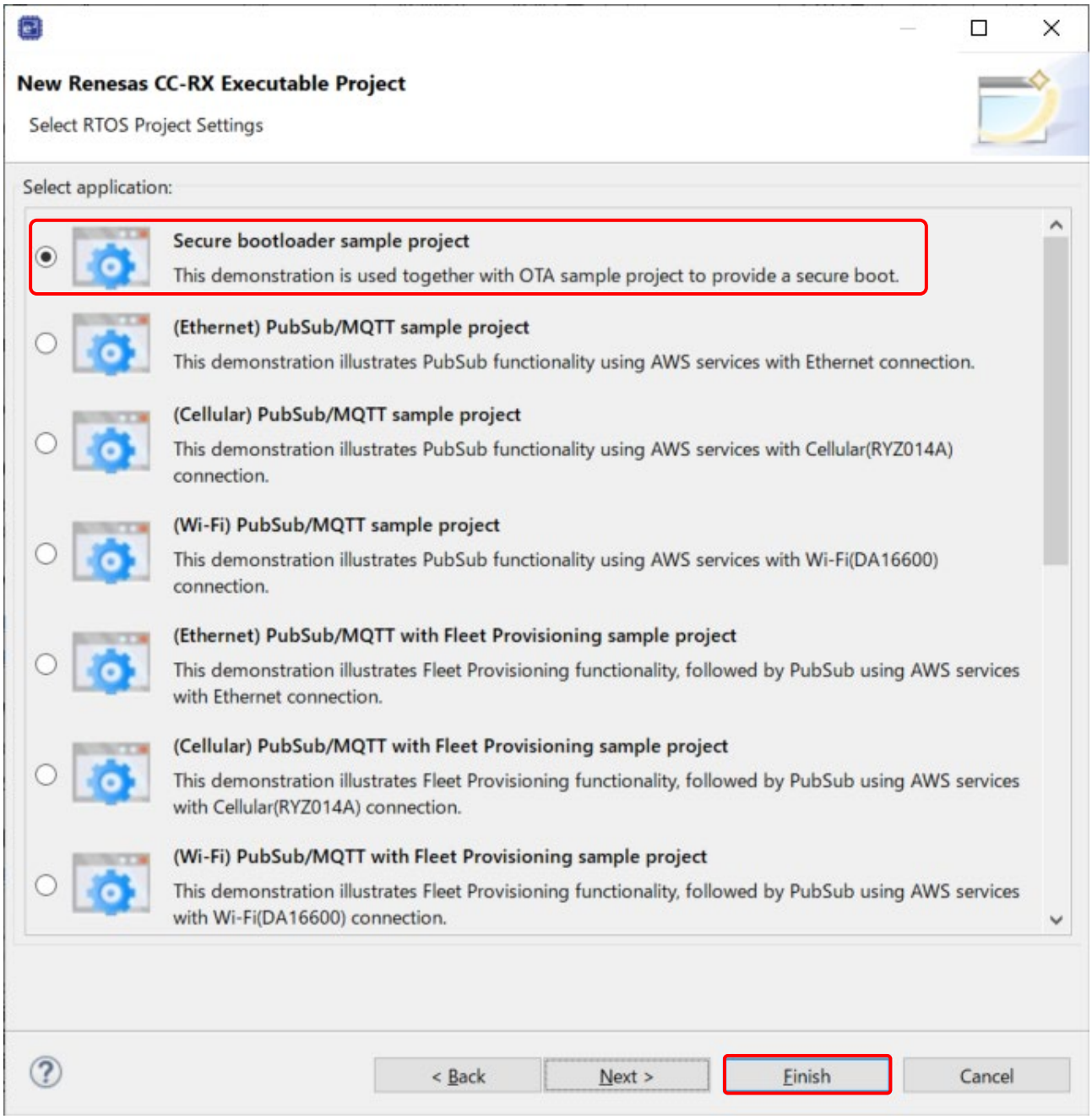
Create the Secure Bootloader sample project in the same manner as the OTA sample project. The procedure involved is basically the same as that used to create the OTA sample project.

Enter **bootloader** as the project name.



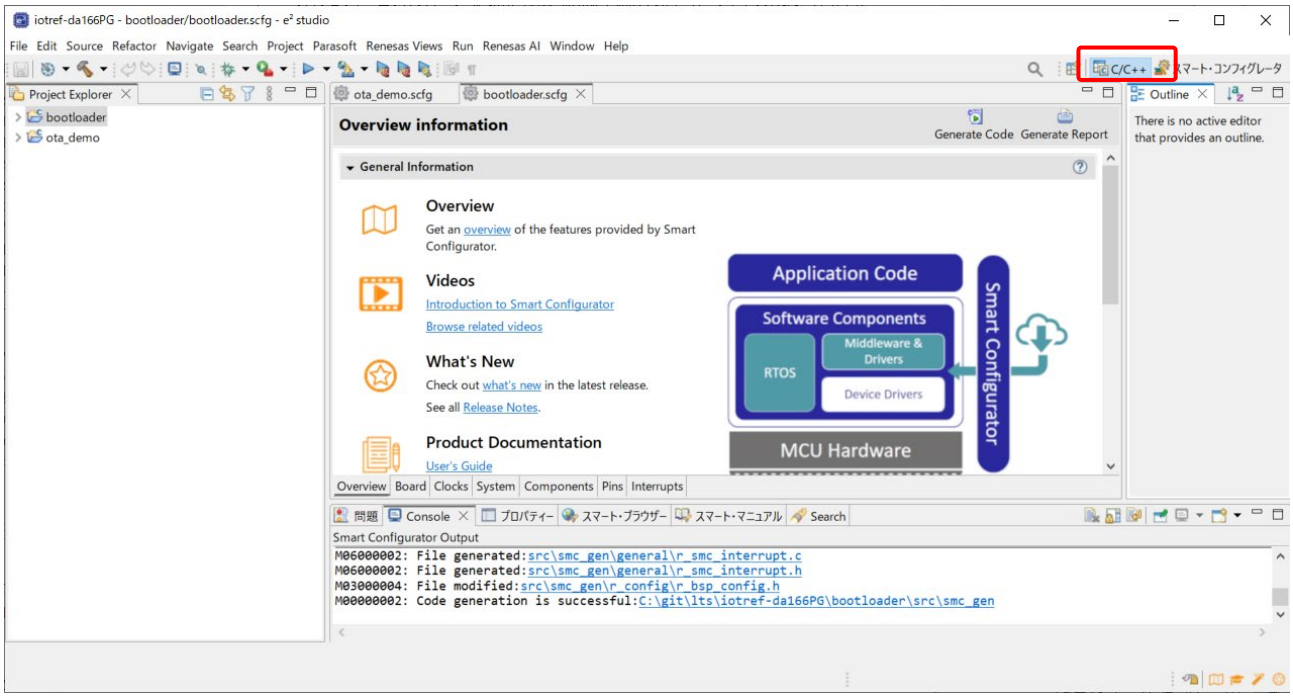
Next, select **Secure bootloader sample project** in the **Select RTOS Project Settings** dialog box.

The other settings are the same as those used when creating the OTA sample project.



The projects appear in e² studio as shown below.

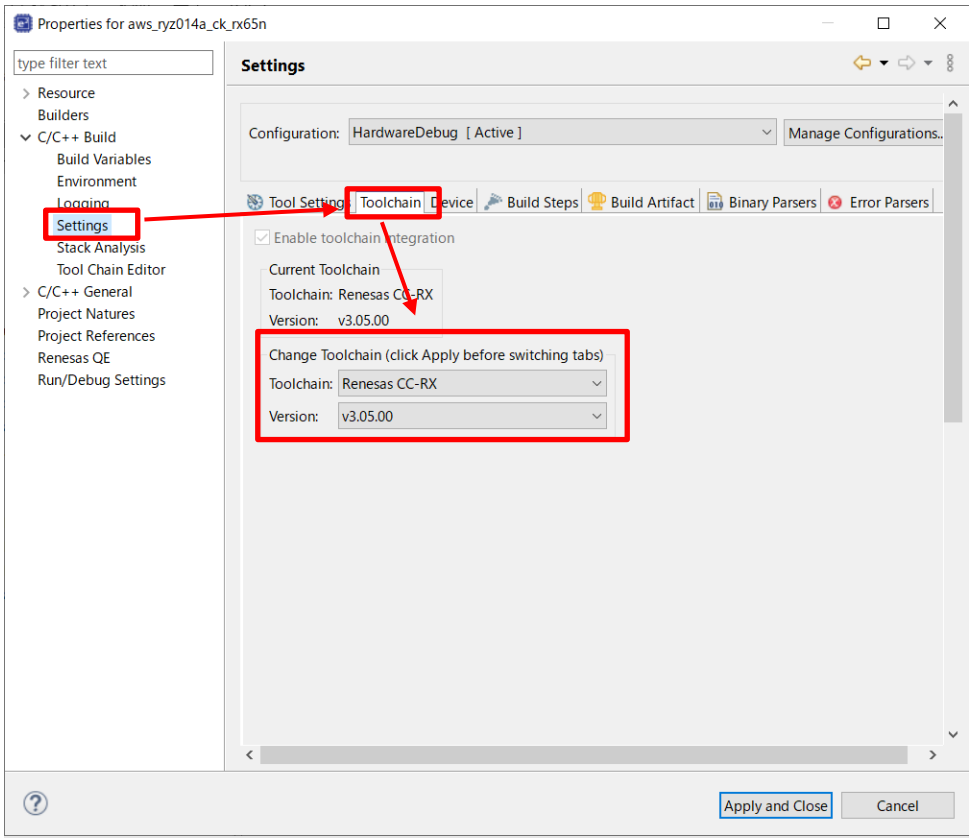
If Project Explorer is not visible, click the button for the **C/C++** perspective in the upper right corner of the window, then select **Window** > **Show View** > **Project Explorer**.



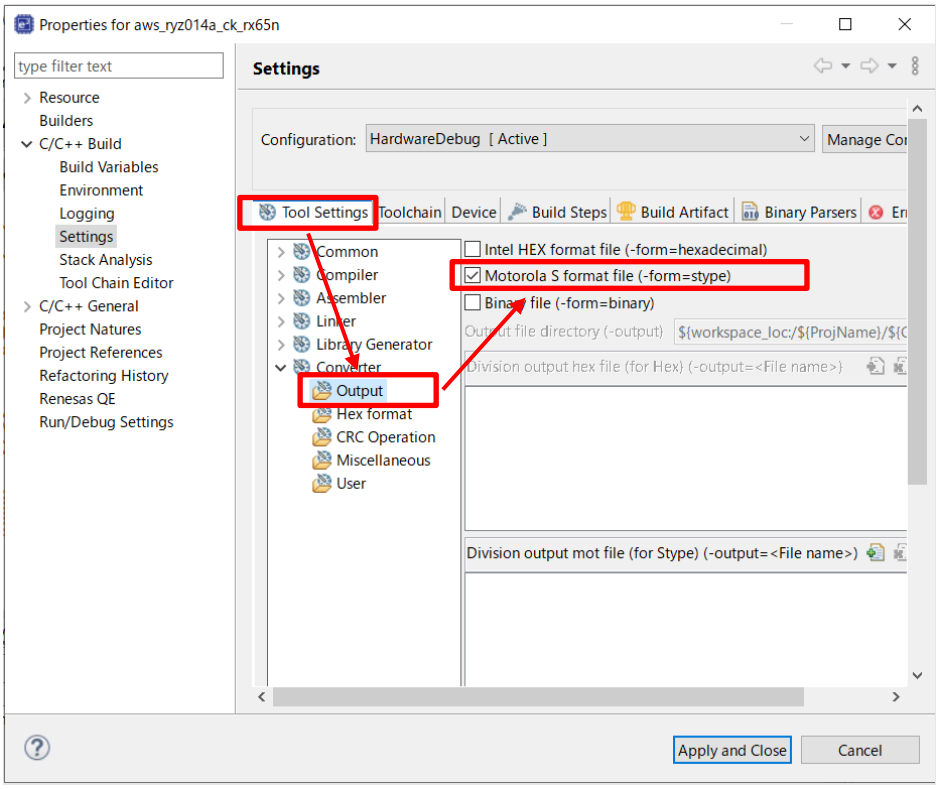
In the discussion that follows, replace the project name **boot_loader_ck_rx65n** with **bootloader** as appropriate.

4.2.2 Checking the Project Environment Settings

(1) For both projects, from the **Projects** menu, select **Properties**, expand the **C/C++ Build** menu, and click **Settings**. On the **Toolchain** tab, confirm that the toolchain is **Renesas CC-RX**.



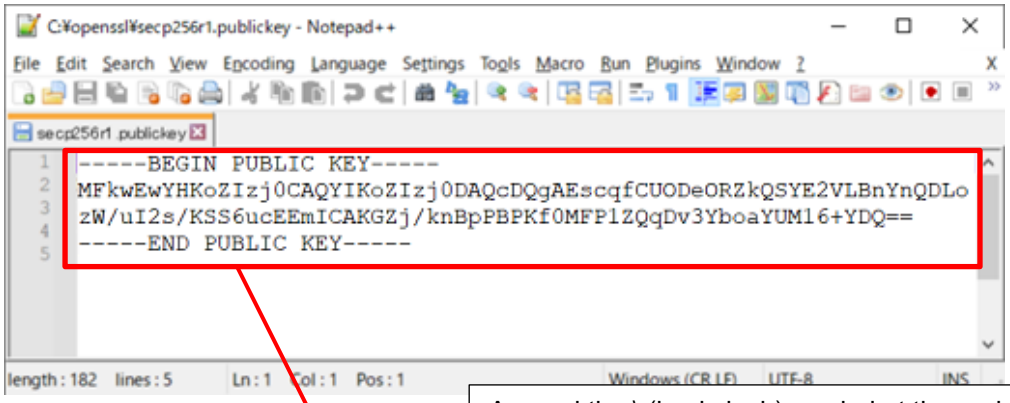
(2) For both projects, select the **Tool Settings** tab, expand the **Converter** menu, select **Output**, and confirm that the box next to **Motorola S format file** is checked.



4.2.3 Setting Up Projects

(1) Assign a public key to the boot_loader_ck_rx65n project.

Copy the contents of the secp256r1.publickey file you created in 4.1(8), and paste the value into the definition of CODE_SIGNENR_PUBLIC_KEY_PEM in boot_loader_ck_rx65n\src\key\code_signer_public_key.h.



Append the \ (backslash) symbol at the end of each line of the definition of CODE_SIGNER_PUBLIC_KEY_PEM.

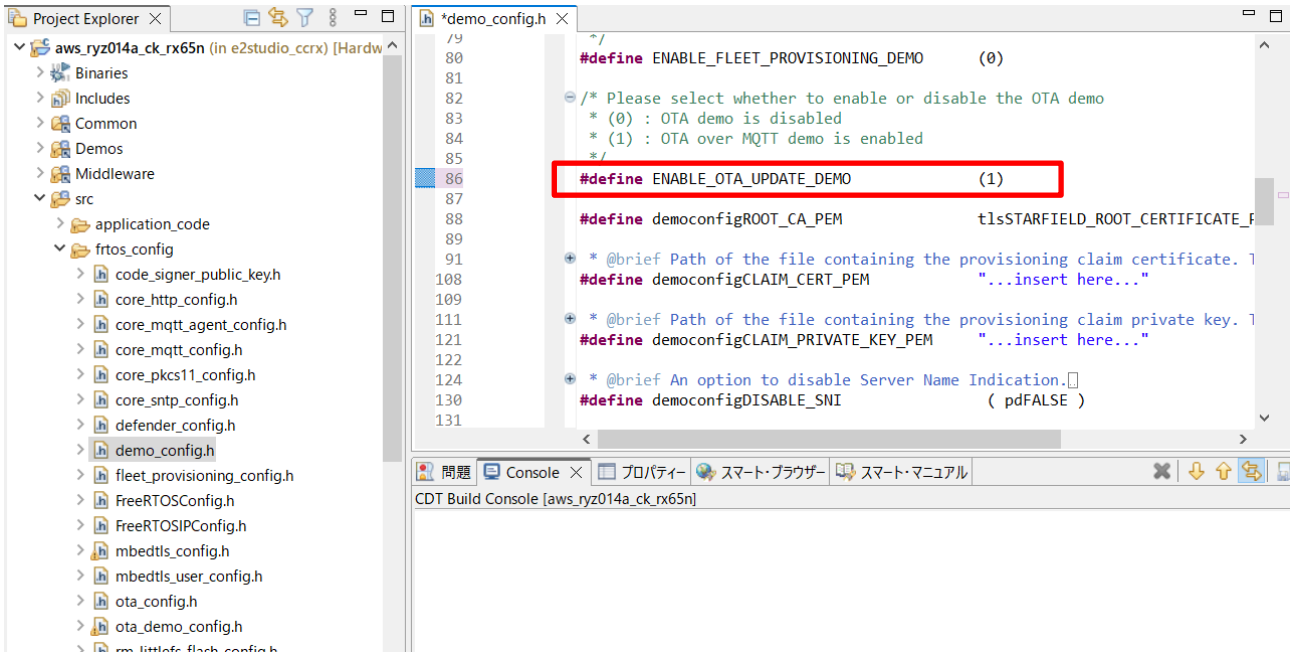
You must enclose each line in quotation marks (") and end the line with the \ symbol

However, do not enter the \ symbol at the end of the final line **-----END PUBLIC KEY-----**.

Example: “xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx”\

(2) Set the definition for the OTA update demo to Enable.

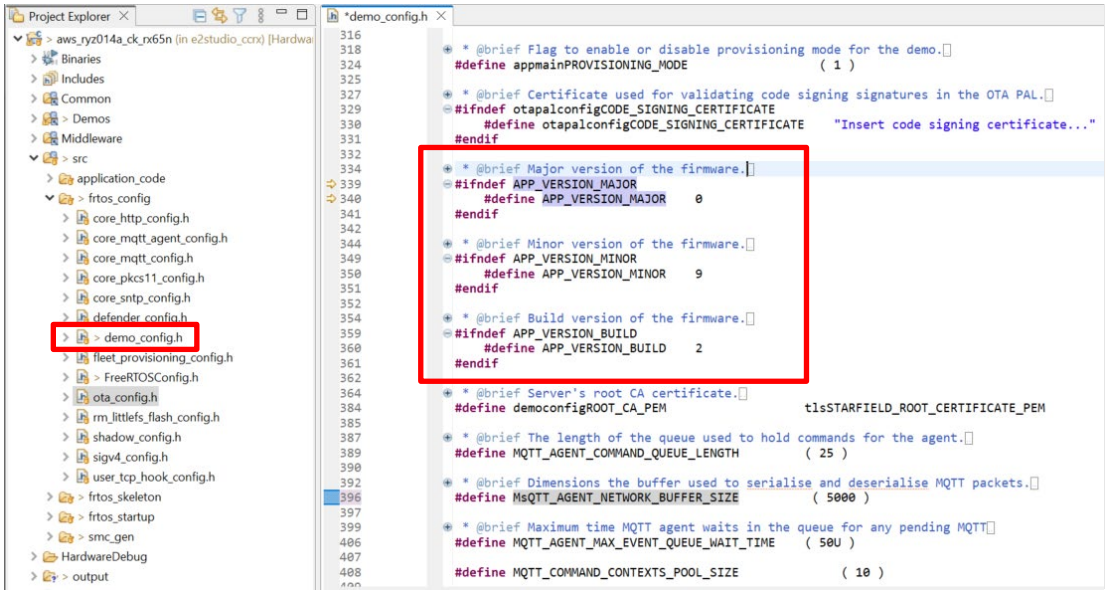
Set ENABLE_OTA_UPDATE_DEMO to 1 (Enable) in aws_ryz014a_ck_rx65n\src\frtos_confit\demo_config.h. (The default is 0)



(3) Confirm that the initial project version is 0.92.

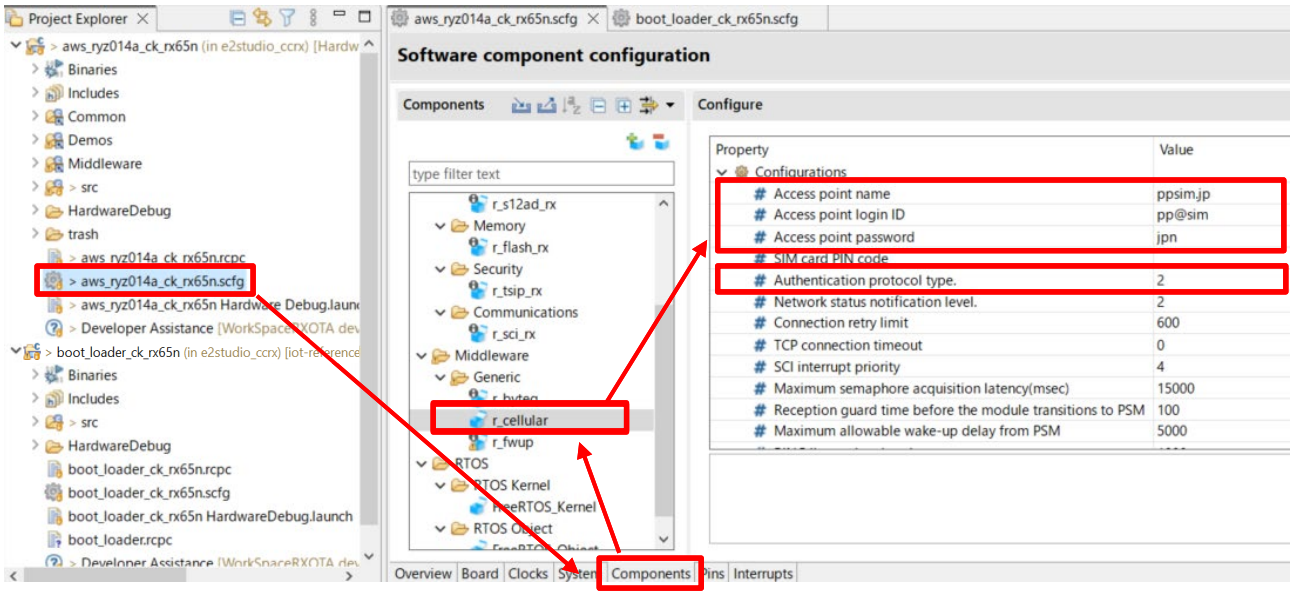
Confirm that the version definitions in aws_ryz014a_ck_rx65n\src\frtos_config\demo_config.h are as follows:

- APP_VERSION_MAJOR 0
- APP_VERSION_MINOR 9
- APP_VERSION_BUILD 2



(4) Configure the RYZ014A Cellular Module Control FIT module (r_cellular).

Open the file aws_ryz014a_ck_rx65n.scfg, and click the **Components** tab. For the r_cellular component, set **Access point name**, **Access point login ID**, **Access point password**, and **Authentication protocol type** according to your SIM card.



To use the SIM card supplied with the CK-RX65N kit, activate the SIM card by following the procedure in *4.1.5 Activating SIM card* in the following application note:

SIM activation, Creating the trial account and using Dashboard with RYZ014A or Ethernet Application for AWS - Getting Started Guide (R01QS0064)

Note: About the setting of Wi-Fi network with DA16600 module, please refer to the GitHub「Settings of Wi-Fi network (Only using Wi-Fi)」. And regarding setting of country code and GMT timezone, please refer to the Settings of Country code and GMT timezone (Only using Wi-Fi) as needed.

4.2.4 Creating the Initial Firmware

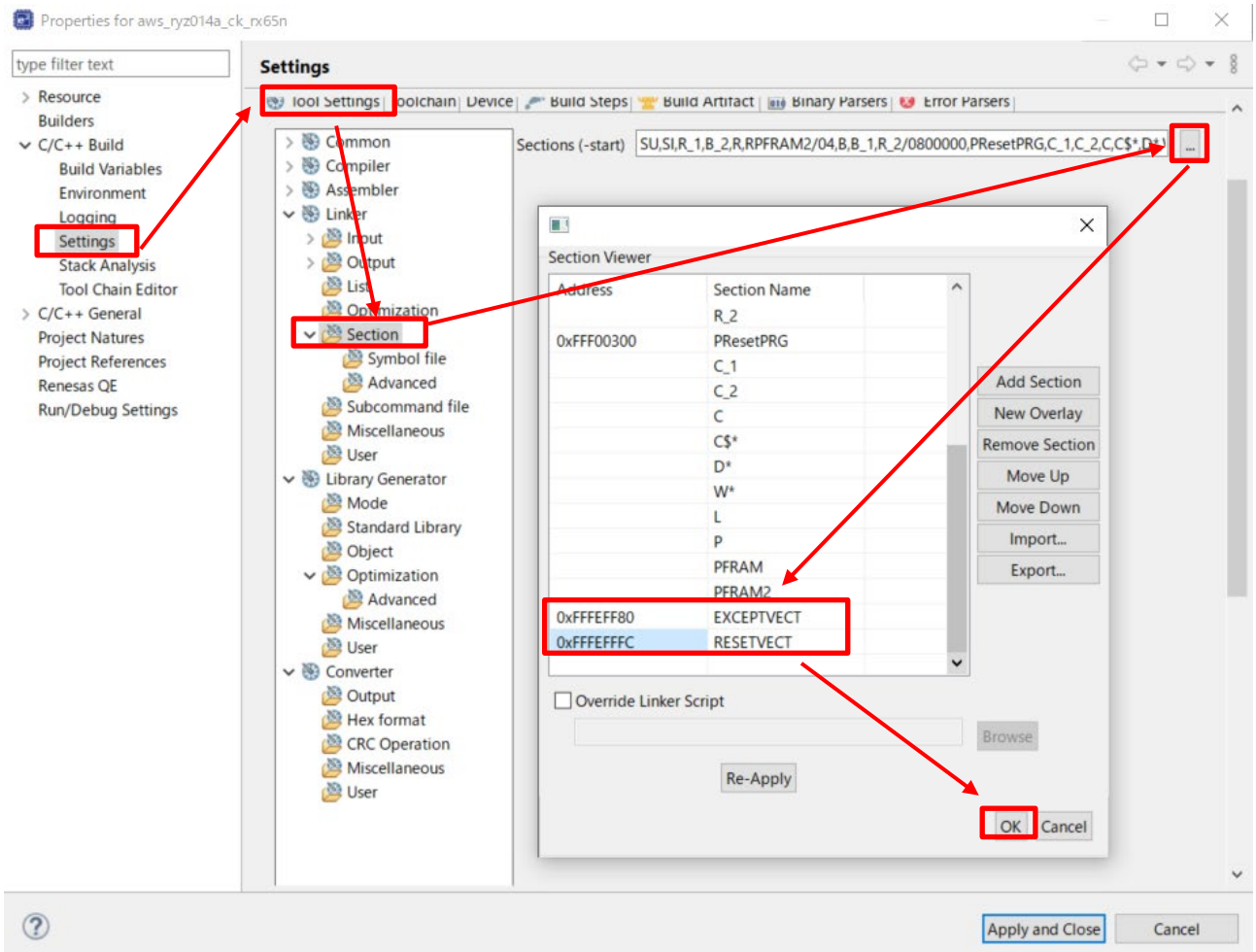
The following explains how to create the initial firmware that combines the boot loader (boot_loader_ck_rx65n) and the firmware (aws_ryz014a_ck_rx65n).

(1) Change the firmware (aws_ryz014a_ck_rx65n) vector and build the firmware.

Open the aws_ryz014a_ck_rx65n project, and select **Project** and then **Properties**.

Expand the **C/C++ Build** menu, and click **Settings**. In the menu tree on the **Tool Settings** tab, expand the **Linker** menu and click **Section**, and open the Section Viewer. Allocate EXCEPTVECT to 0xFFFEFF80 and RESETVECT to 0xFFFEFFFC.

After this, build the aws_ryz014a_ck_rx65n project to create the firmware.



(2) Create the bootloader (boot_loader_ck_rx65n).

Build the boot_loader_ck_rx65n project to create the bootloader.

(3) Use Renesas Image Generator to generate the initial firmware.

Place the following files in the Renesas Image Generator folder:

- The results of the build process in 4.2.4(1): aws_ryz014a_ck_rx65n.mot
- The results of the build process in 4.2.4(2): boot_loader_ck_rx65n .mot
- The private key created in 4.1(7): secp256r1.privatekey

Open a command prompt, navigate to the Renesas Image Generator folder, and execute the following command to generate the file userprog.mot.

python image-gen.py -iup *aws_ryz014a_ck_rx65n.mot* -ip *RX65N_DualBank_ImageGenerator_PRM .csv* -o *userprog* -ibp *boot_loader_ck_rx65n .mot* -key *secp256r1.privatekey* -vt ecdsa -ff RTOS

(4) Install the flash programming tool (Renesas Flash Programmer).

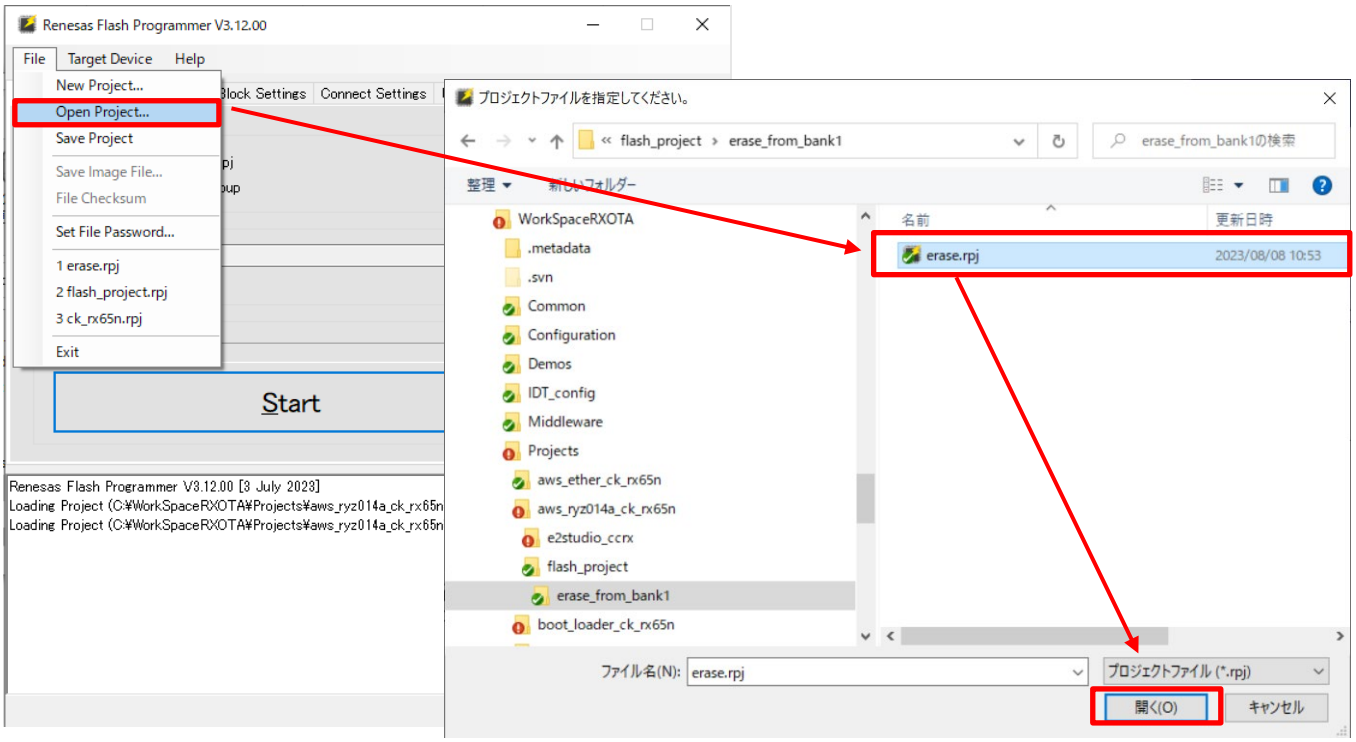
Access the download site of the flash programming tool, and download and install **Renesas Flash Programmer V3.11.02 Windows**.

(5) Follow the instructions in 2.5, Connecting the CK-RX65N, to connect the PC to the CK-RX65N.

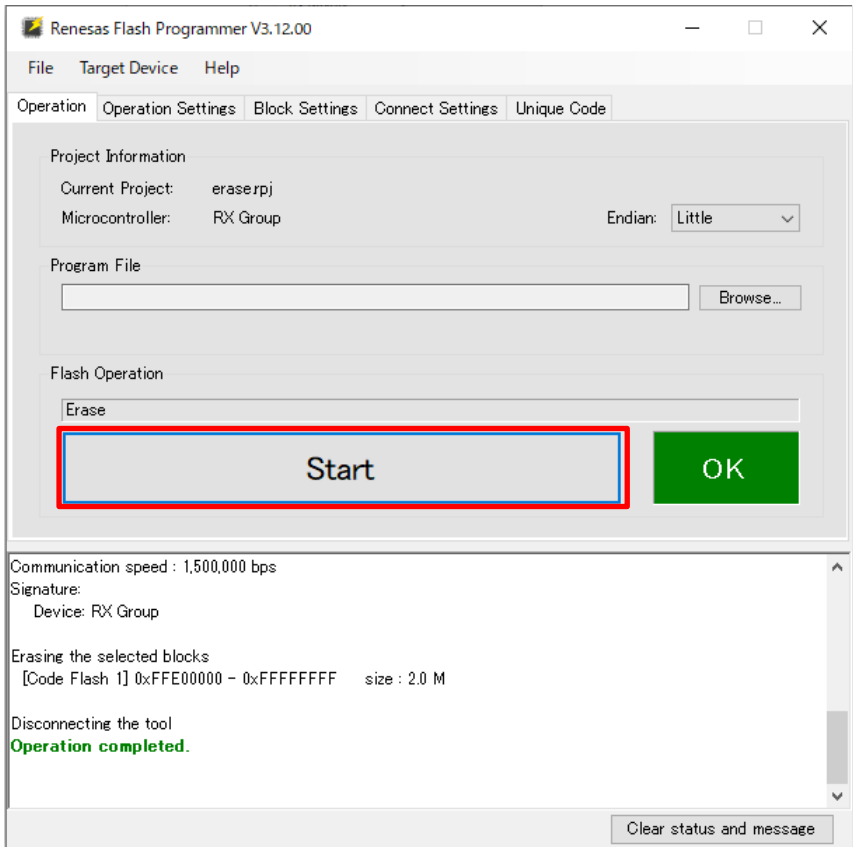
(6) Launch Renesas Flash Programmer, and open the erase.rpj project.

The erase.rpj project can be found either in a subfolder of the sample program or in a subfolder of FreeRTOS, which you downloaded when creating a new project.

\Projects\aws_ryz014a_ck_rx65n\flash_project\erase_from_bank1



(7) Click **Start** to erase the device.

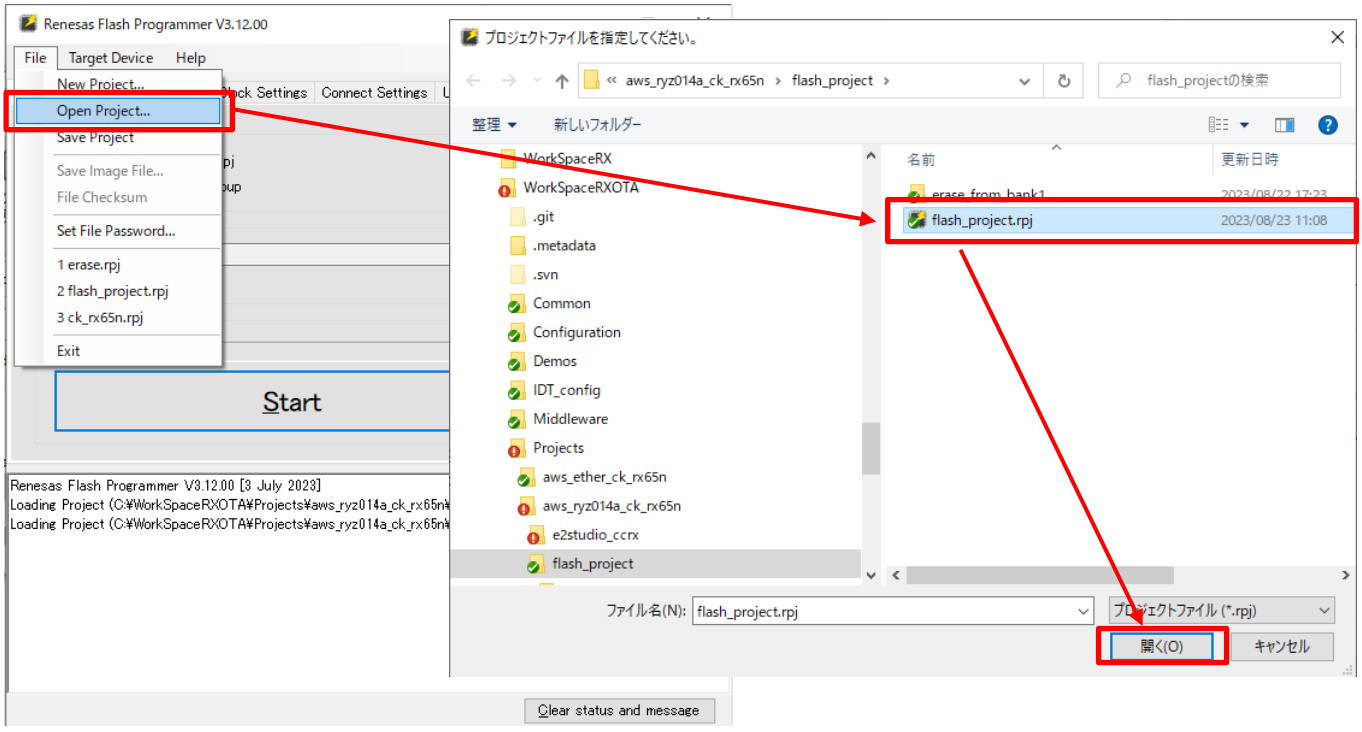


If the message **Error (E3000107): Device connection information does not match** is output, proceed with step (8).

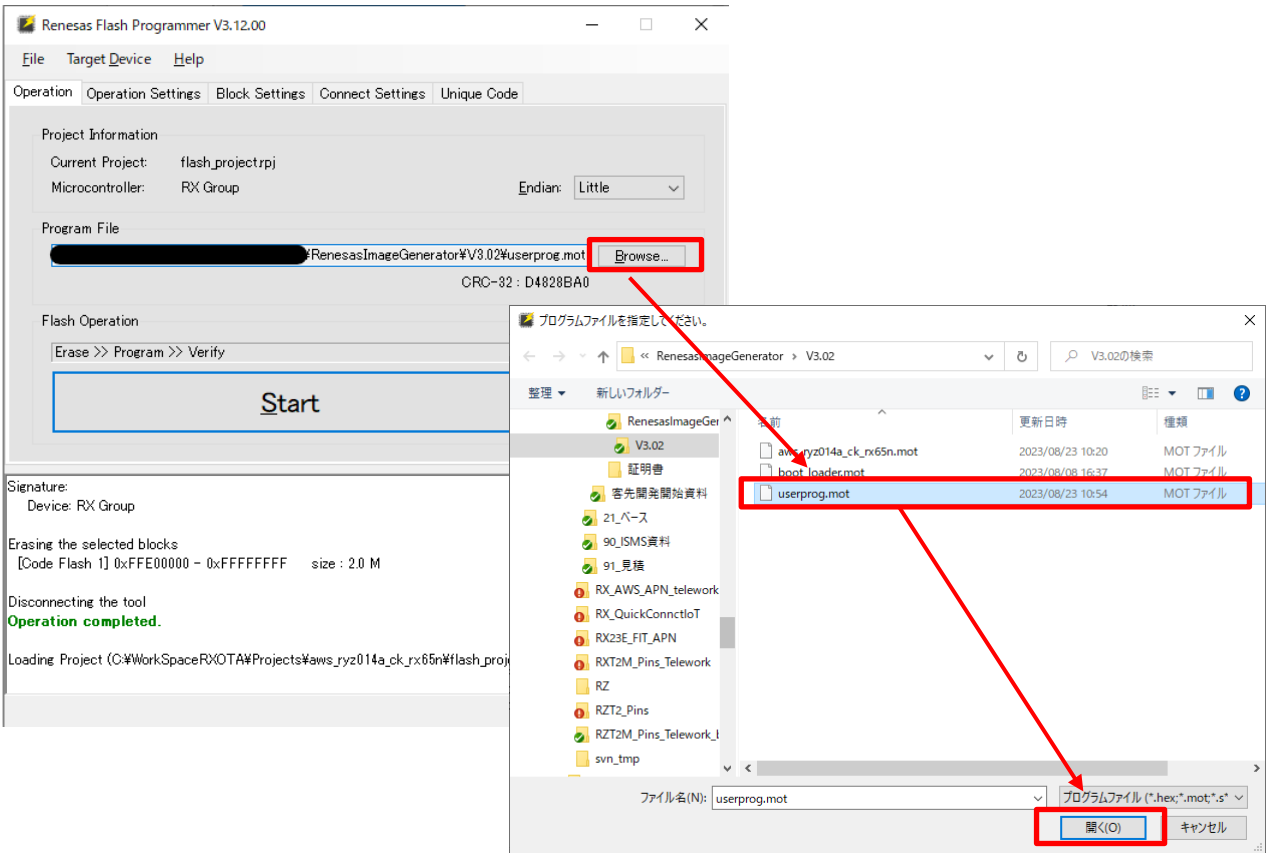
(8) Open the flash_project.rpj project.

The flash_project.rpj project can be found either in a subfolder of the sample program or in a subfolder of FreeRTOS, which you downloaded when creating a new project.

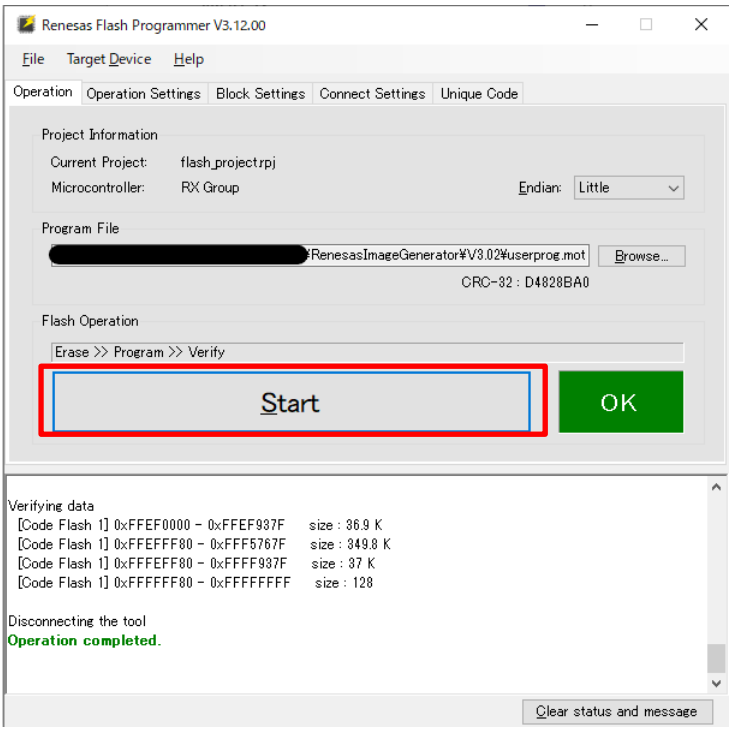
\Projects\aws_ryz014a_ck_rx65n\flash_project\



(9) Select the initial firmware (userprog.mot) created in 4.2.4(2).



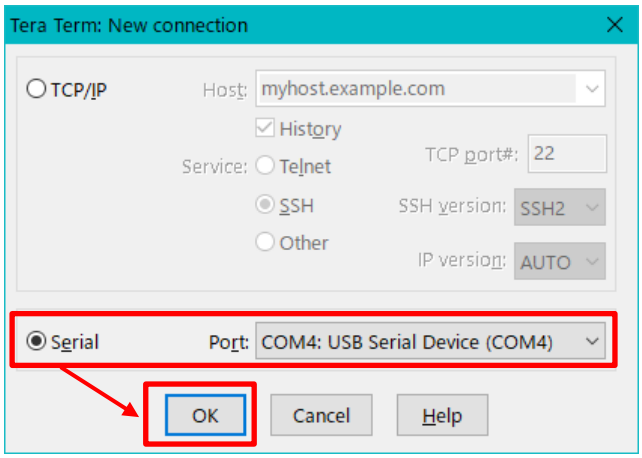
(10) Write the firmware.



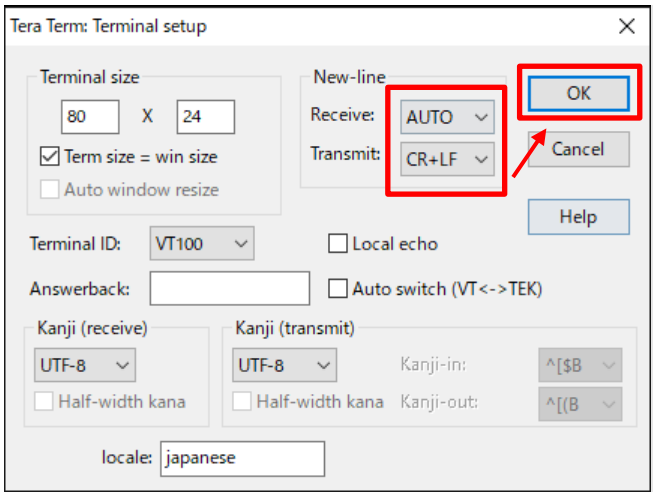
4.2.5 Registering AWS IoT Information

The following explains how to set AWS IoT information in Tera Term by running the aws_ryz014a_ck_rx65n project. The information set by this process is written to data flash memory.

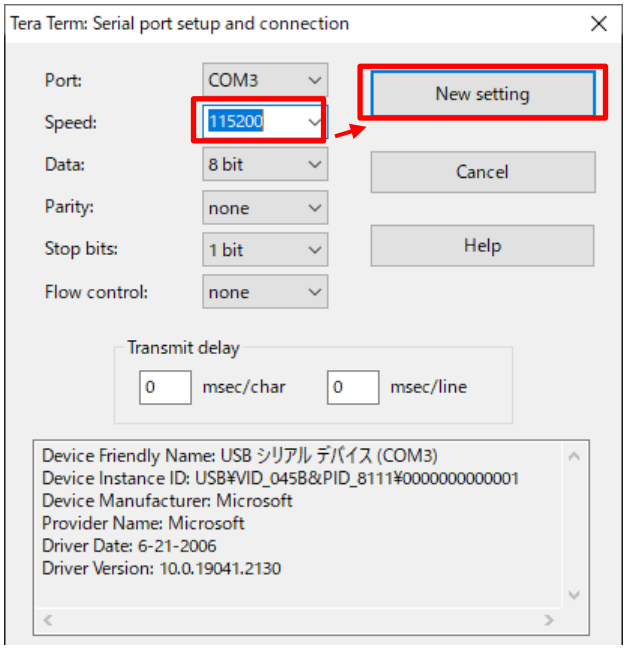
(1) Open Tera Term, and from the **File** menu, select **New Connection**. In the dialog box that appears, select **Serial** and then click **OK**.



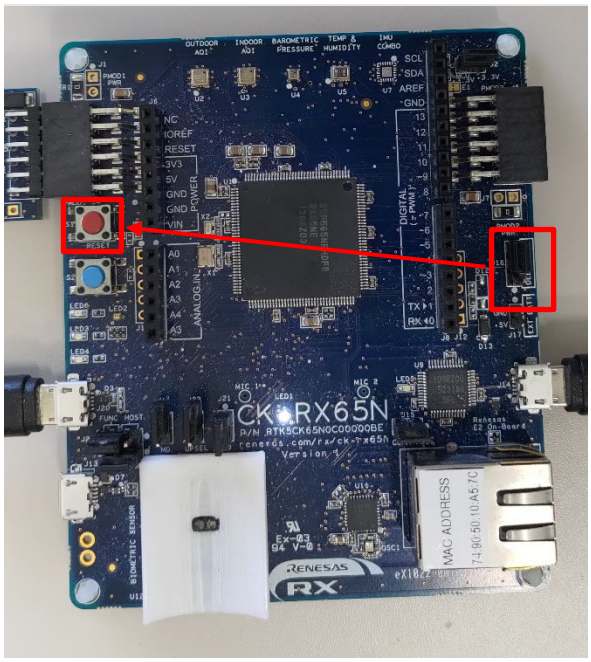
(2) From the **Setup** menu, select **Terminal**. In the **New-line** area of the dialog box that appears, select **AUTO** for **Receive** and **CR+LF** for **Transmit**.



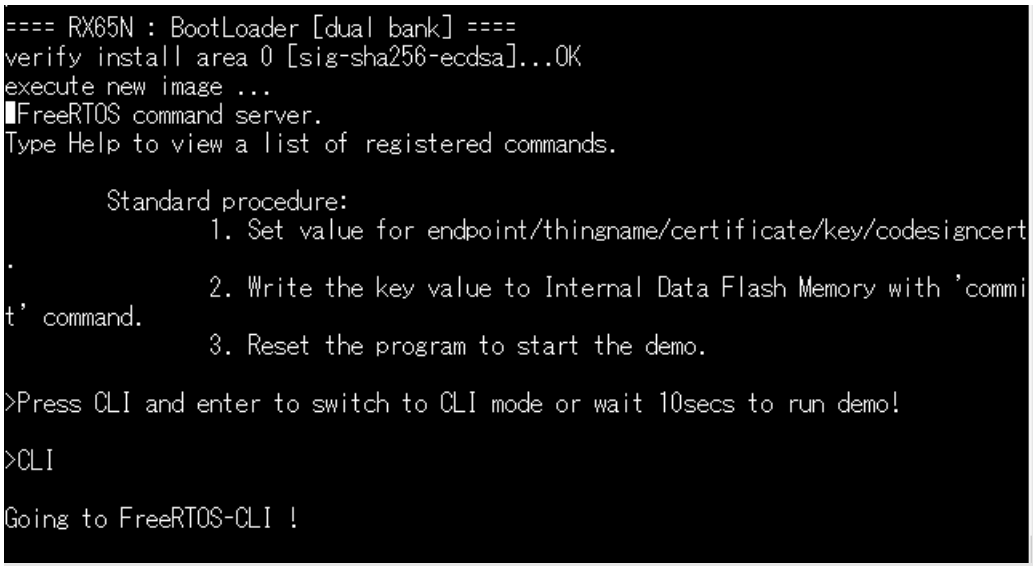
(3) From the **Setup** menu, select **Serial port**. In the dialog box that appears, set **Speed** to 115200 and then click **New setting**.



(4) Move the jumper on J16 of the CK-RX65N board to the RUN setting, and then press the RESET switch.

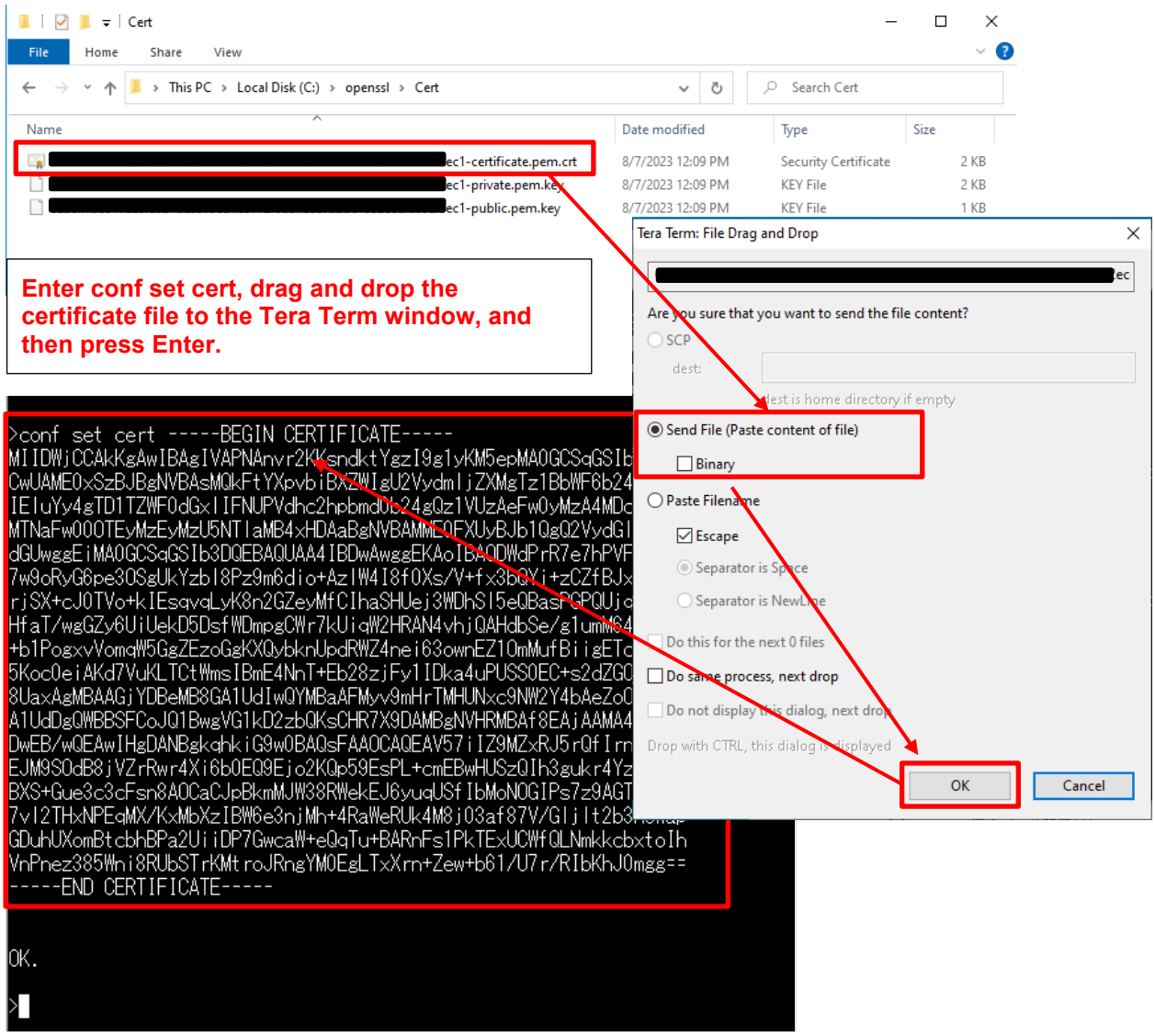


(5) Enter CLI and press the **Enter** key within 10 seconds of the menu appearing in the Tera Term window.



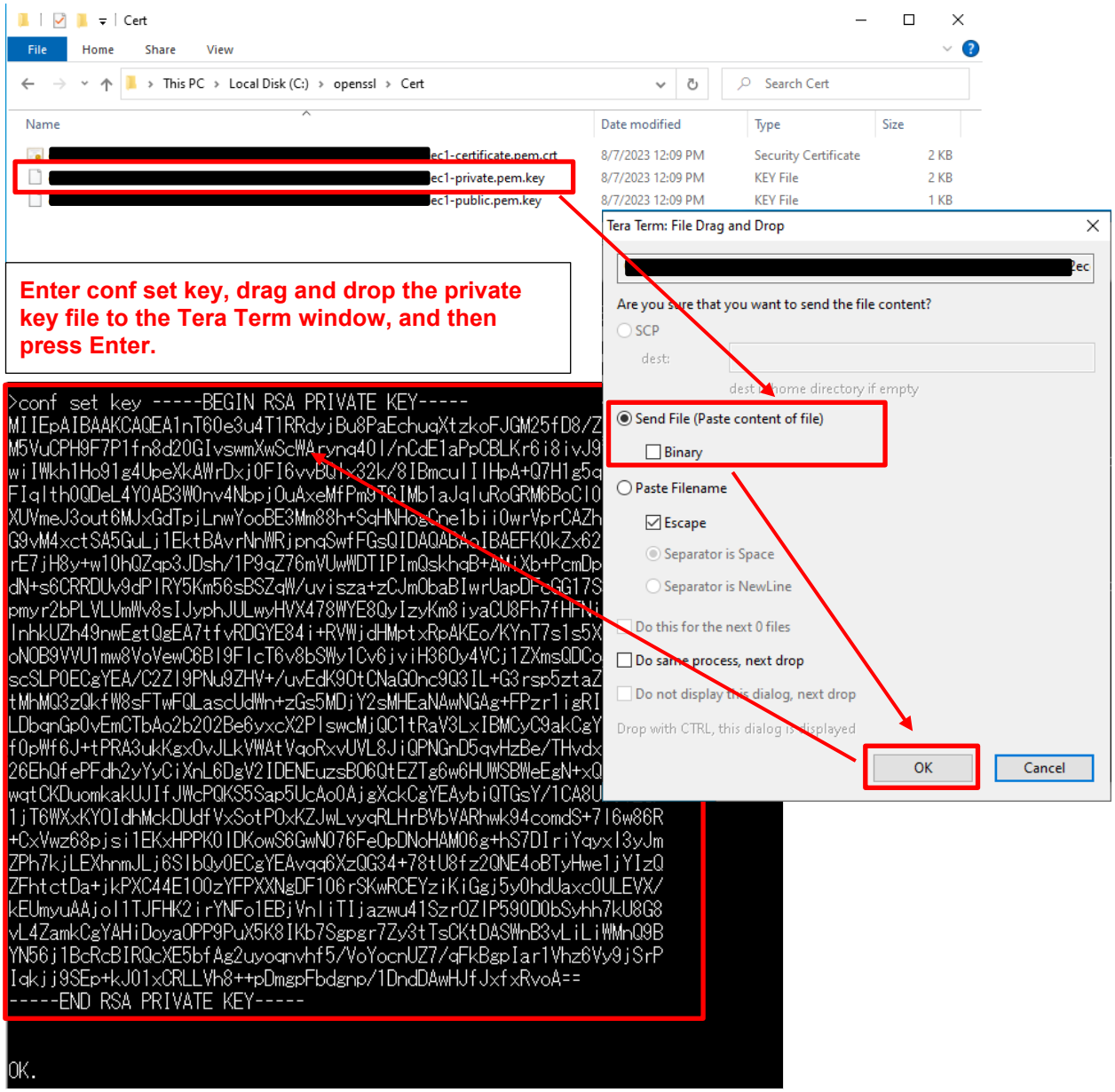
(6) Register the certificate you downloaded in 3.3.2(6).

Enter conf set cert in Tera Term, and then drag and drop the certificate file (xxxx-certificate.pem.crt) to the Tera Term window to send the file. Finally, press **Enter** in the Tera Term window.



(7) Register the private key you downloaded in 3.3.2(6).

Enter conf set key in Tera Term, and then drag and drop the private key file (xxxx- private.pem.key) to the Tera Term window to send the file. Finally, press **Enter** in the Tera Term window.



(8) Register the thing name you set in 3.3.2(3) and the endpoint you made a note of in 3.3.3(1).

Execute the following commands in Tera Term:

conf set thingname *thing-name*

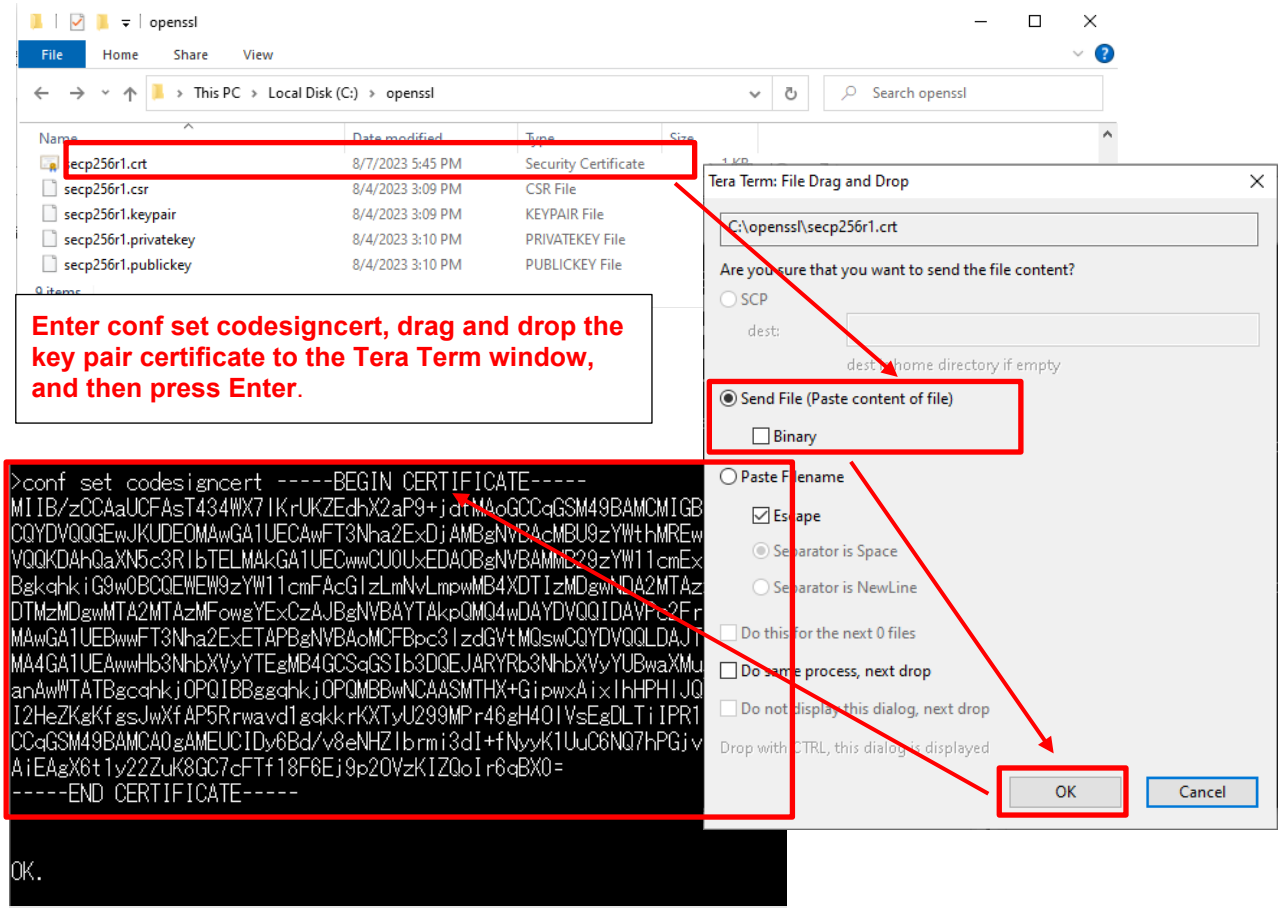
conf set endpoint *endpoint-name*

```
>conf set thingname rx65n_ota_demo_thing
OK.
>conf set endpoint a1k                    ap-northeast-1.amazonaws.com
OK.
```

(9) Register the key pair certificate (secp256r1.crt) generated in 4.1(6).

Enter conf set codesigncert in Tera Term, and then drag and drop the key pair certificate (secp256r1.crt) to the Tera Term window to send the file. Then press the **Enter** key in Tera Term.

Note: Change the linefeed code of the certificate file to LF before pasting the file contents



(10) Commit the AWS IoT settings (write the settings to data flash memory).

Execute the following commands in Tera Term:

conf commit

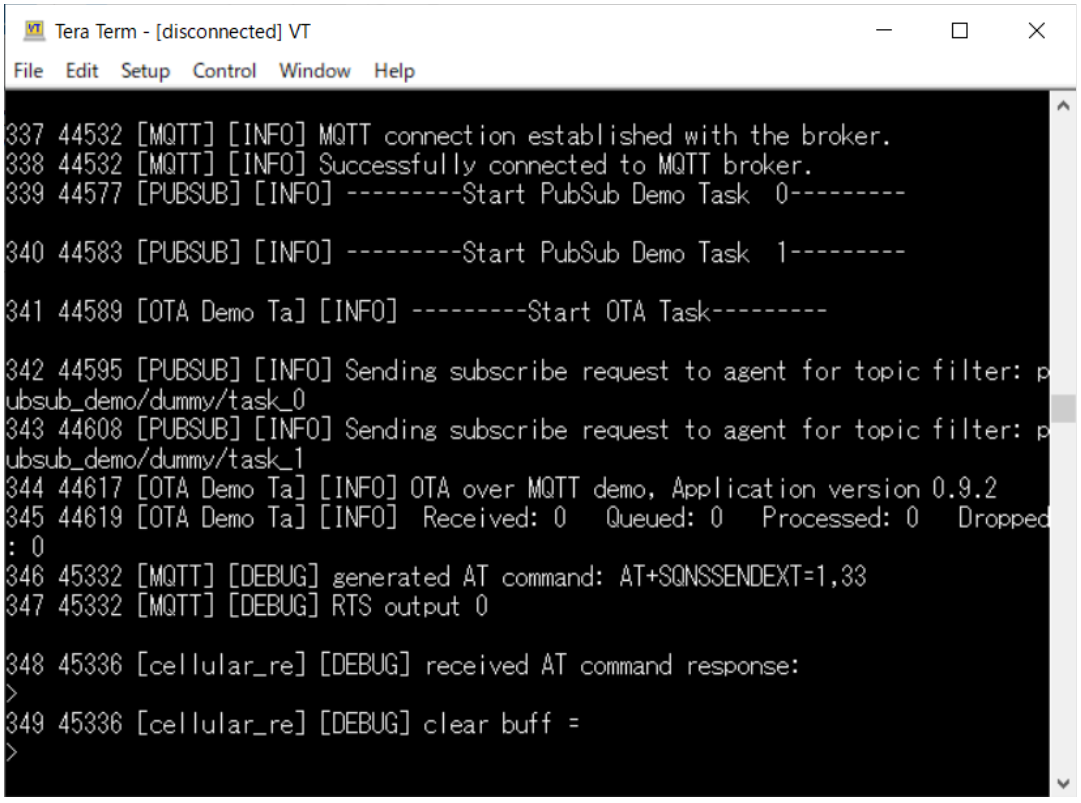
```
>conf commit

0 4472481 [CLI] Destroyed Certificate.

1 4472485 [CLI] Write certificate...

2 4472545 [CLI] Destroyed Private key.

3 4472685 [CLI] Write Private key...

Configuration saved to Data Flash and used 2879 bytes.
```

(11) Perform a reset.

Execute the following commands in Tera Term:

reset

After the reset process is complete, confirm that Tera Term displays a communication log and the application is waiting for OTA jobs.



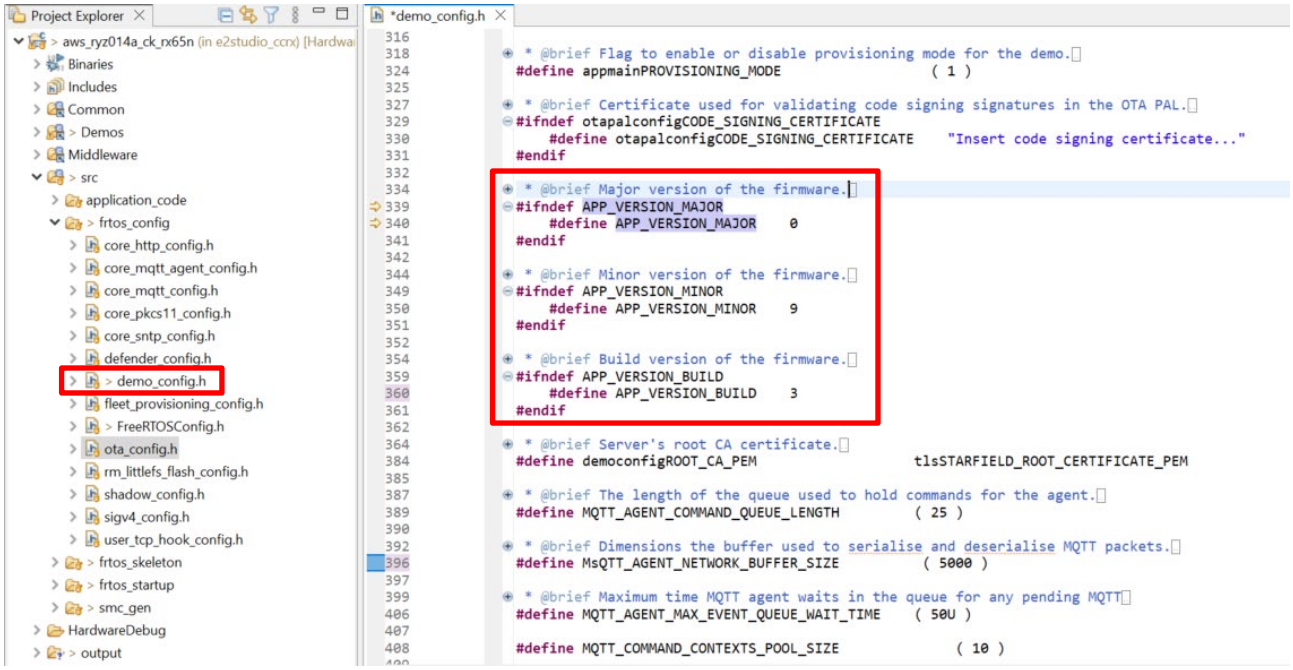
5. Updating the Firmware

5.1 Creating the Updated Firmware

5.1.1 Changing the Firmware Version

(1) Change the firmware version to v0.9.3.

Repeat the build process, this time with 3 specified for the APP_VERSION_BUILD definition in aws_ryz014a_ck_rx65n\src\frtos_config\demo_config.h.



(2) Use Renesas Image Generator to generate the updated firmware.

Overwrite the file in the Renesas Image Generator folder with the firmware you rebuilt in 5.1.1(1) (aws_ryz014a_ck_rx65n.mot), and then execute the following command at the command prompt:

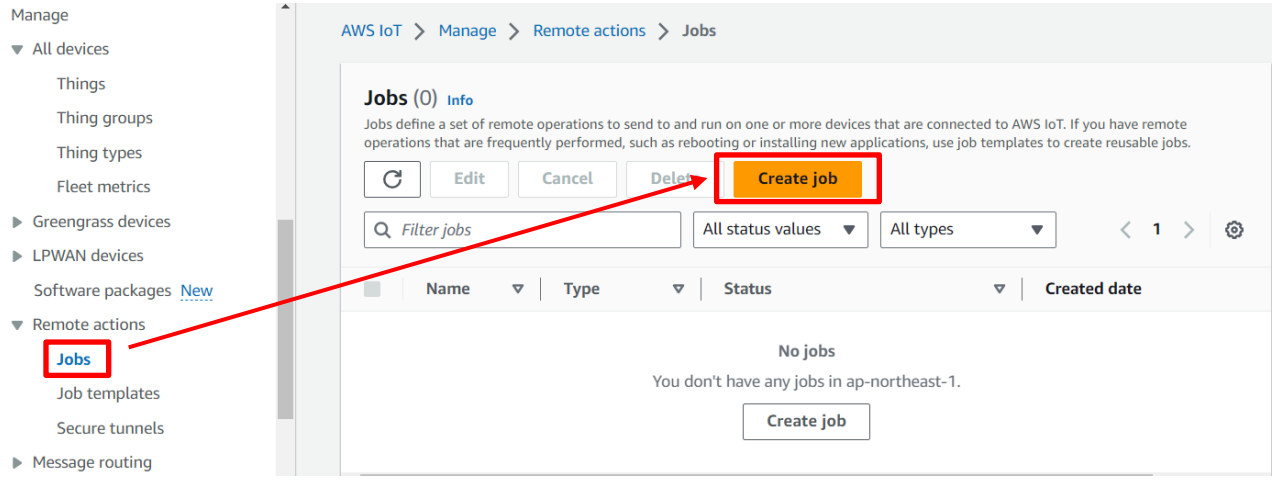
python image-gen.py -iup *aws_ryz014a_ck_rx65n.mot* -ip *RX65N_DualBank_ImageGenerator_PRM.csv* -o *user_093* -key *secp256r1.privatekey* -vt ecdsa -ff RTOS

This command generates a file named user_093.rsu.

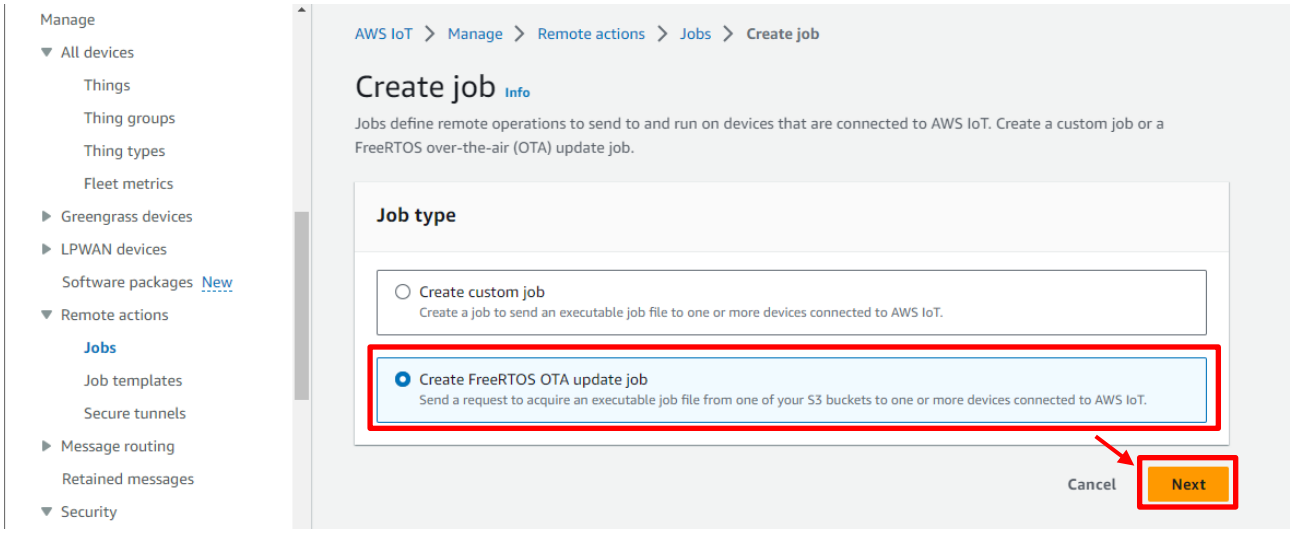
5.2 Updating the Firmware

In AWS, create an OTA update job that will update the firmware.

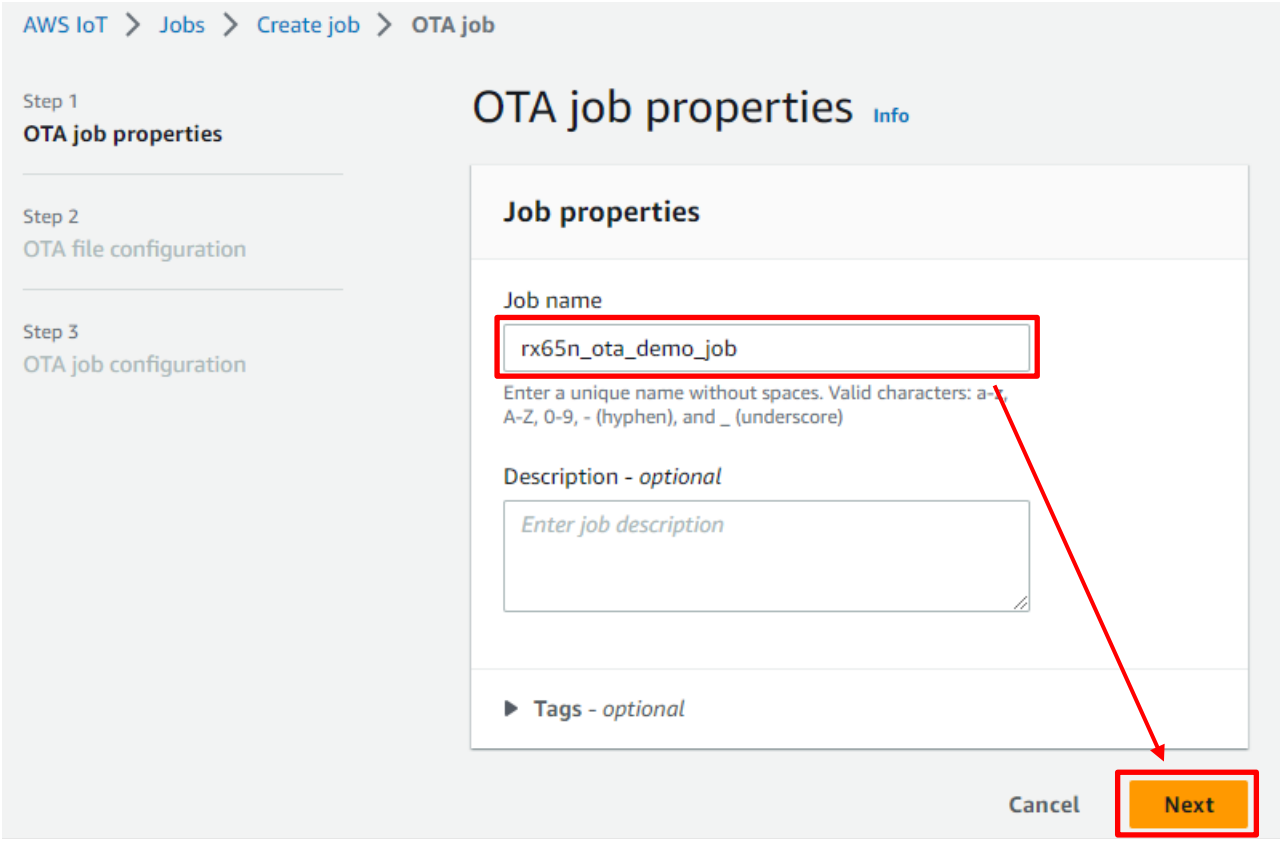
(1) In the IOT Core menu, select **Manage**, **Remote actions**, and **Jobs**, and then click the **Create job** button.



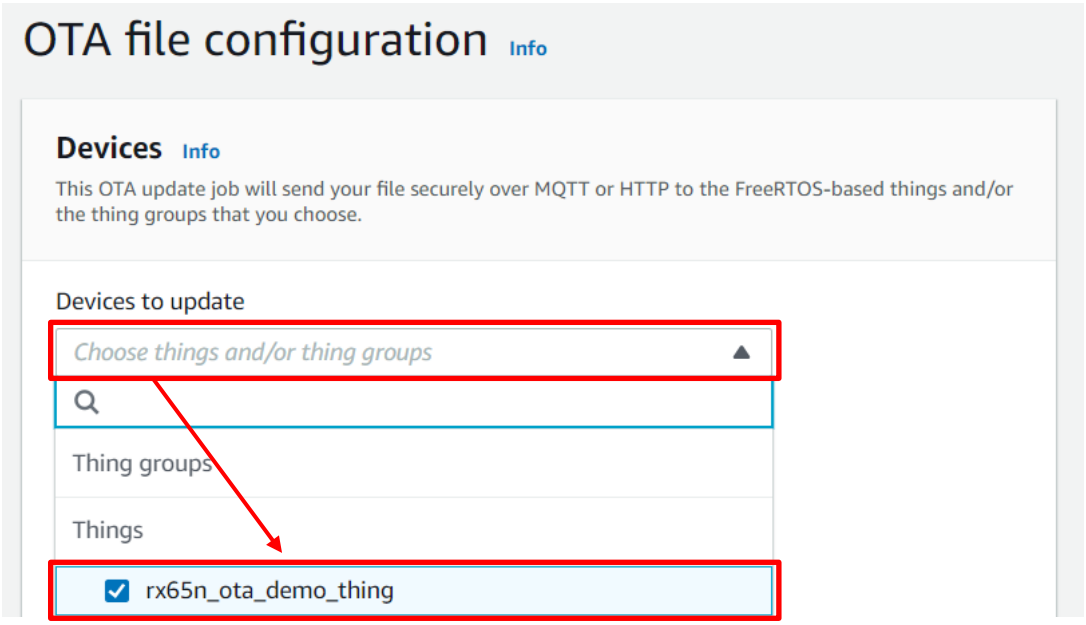
(2) Select **Create FreeRTOS OTA update job** and then click **Next**.



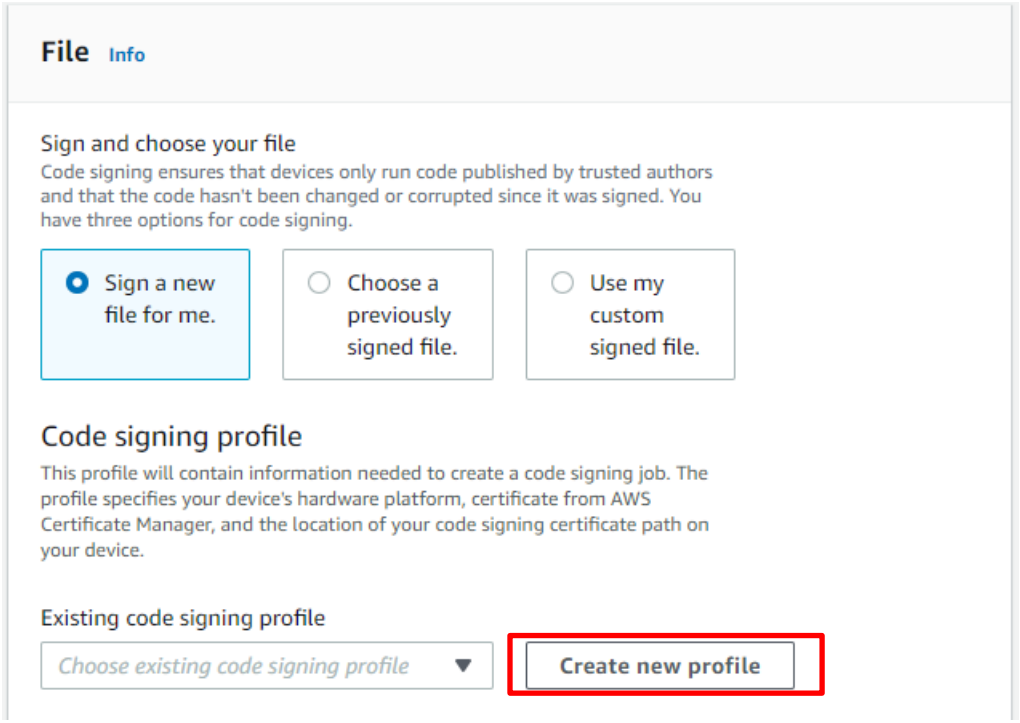
(3) Enter a job name (example: rx65n_ota_demo_job) and then click **Next**.



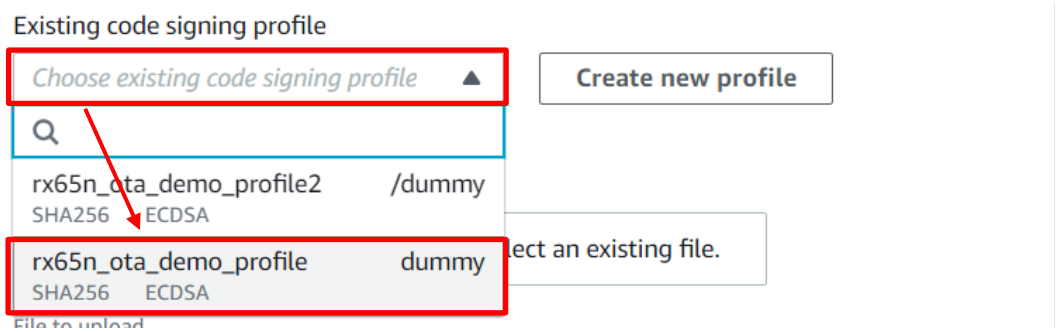
(4) Click the **Devices to update** drop-down list and select the device to update.



(5) Click **Create new profile**.



You can skip steps (5) to (9) if you have already created a profile. Click **Choose existing code signing profile** and select the profile you created from the drop-down list.

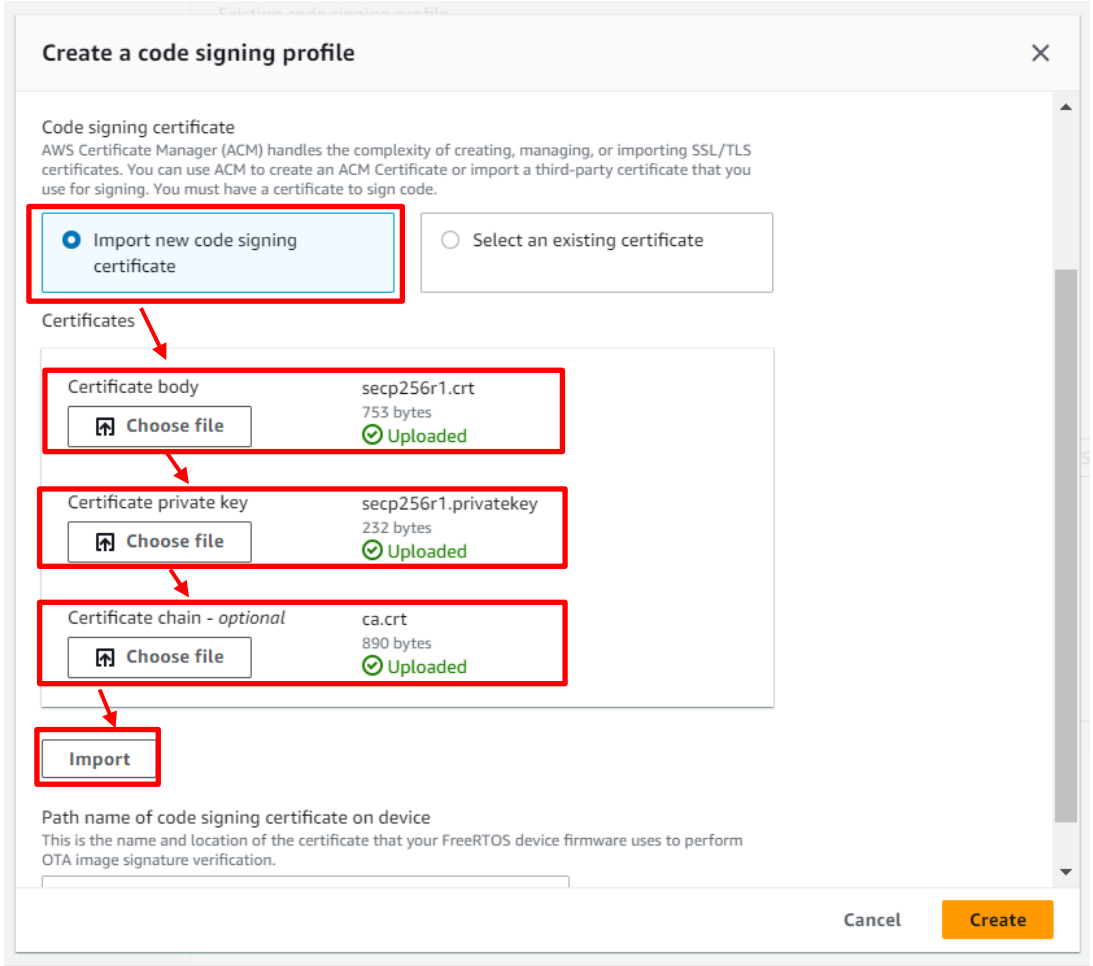


(6) Create a profile (1): Profile name and device hardware platform.

- Enter the profile name (example: rx65n_ota_demo_profile)
- Select **Windows Simulator** as the device hardware platform

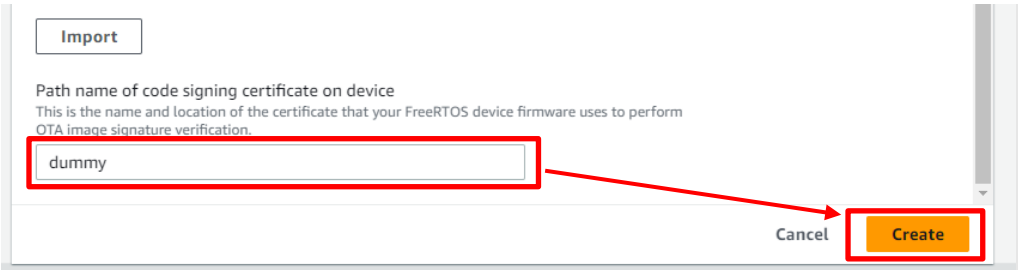
(7) Create a profile (2): Import a certificate.

- In the **Code signing certificate** area, click **Import new code signing certificate**.
- In **Certificate body**, select the file secp256r1.crt you created in 4.1(6).
- In **Certificate private key**, select the file secp256r1.privateky you created in 4.1.
- In **Certificate chain**, select the file ca.crt you created in 4.1(3).
- Click **Import**.

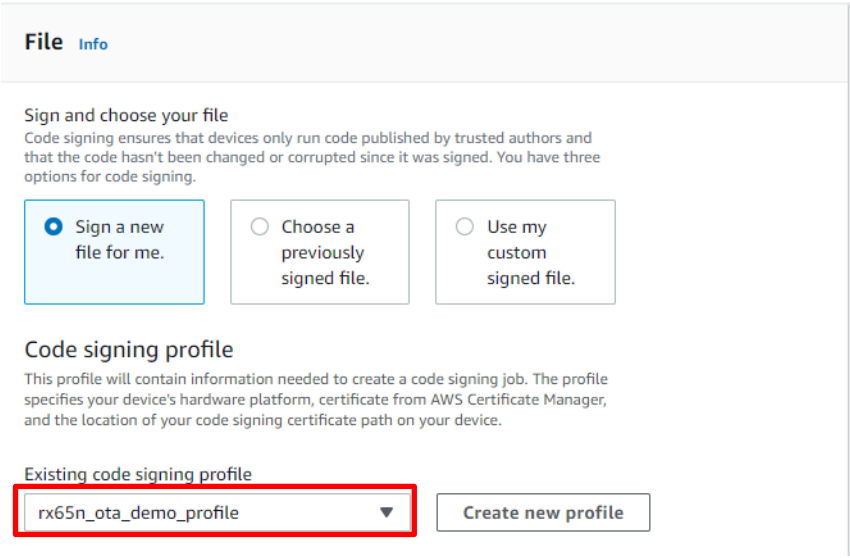


(8) Create a profile (3): Enter the path of the code signing certificate of the device and then click **Create**.

You can enter any path. (Example: dummy)

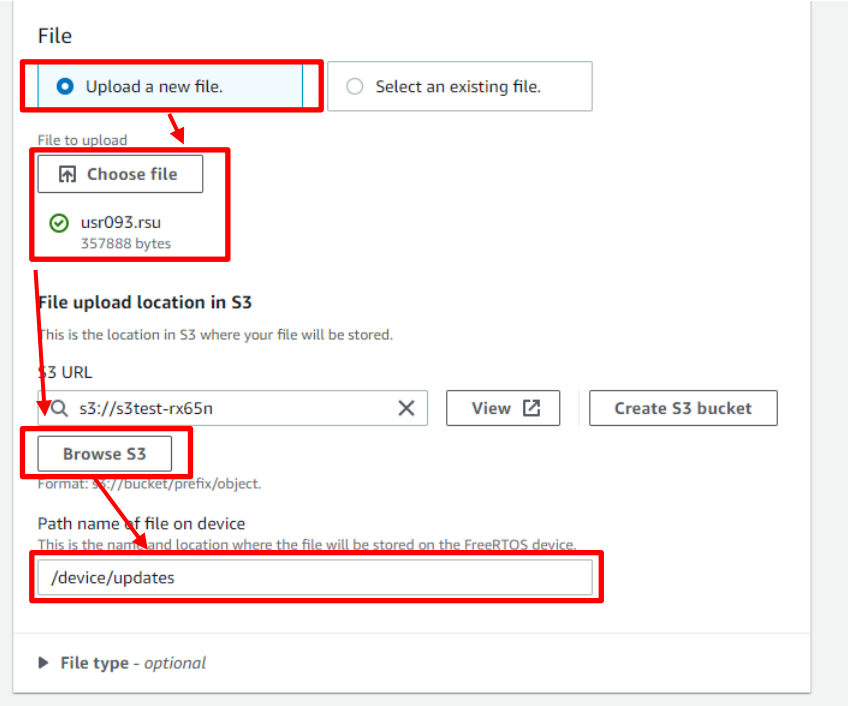


(9) Confirm that the name of the profile you created earlier is selected in the **Existing code signing profile** drop-down list.

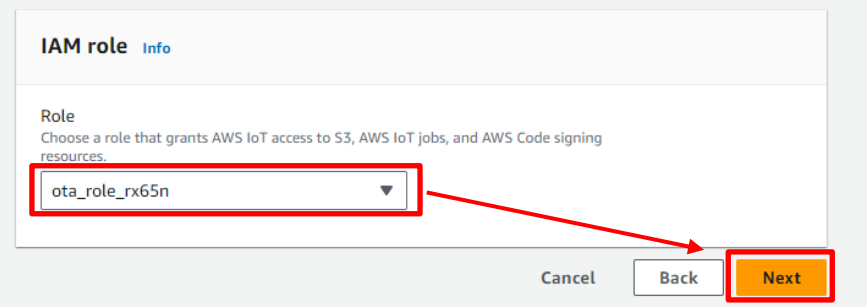


(10) Update the firmware.

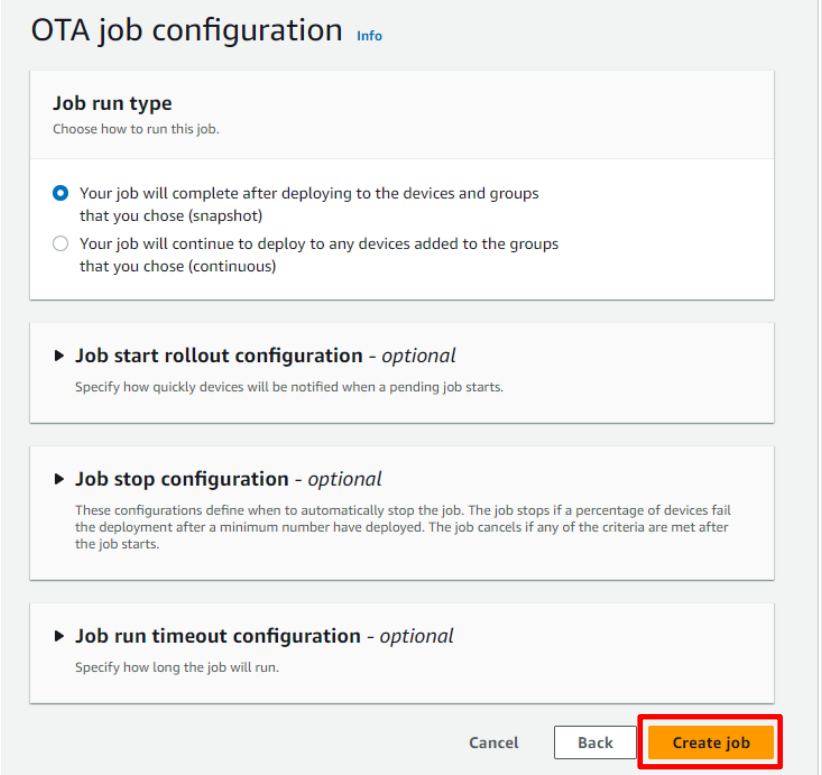
- Select **Upload a new file**.
- In **File to upload**, select the file usr093.rsu you created in 5.1.1(2).
- Click **Browse S3** and select the S3 bucket you created in 3.4.
- Enter a path name in **Path name of file on device**. (You can enter any path name. Example: /device/updates)



(11) In the **Role** drop-down list, select the role you created in 3.5(5) and then click **Next**.

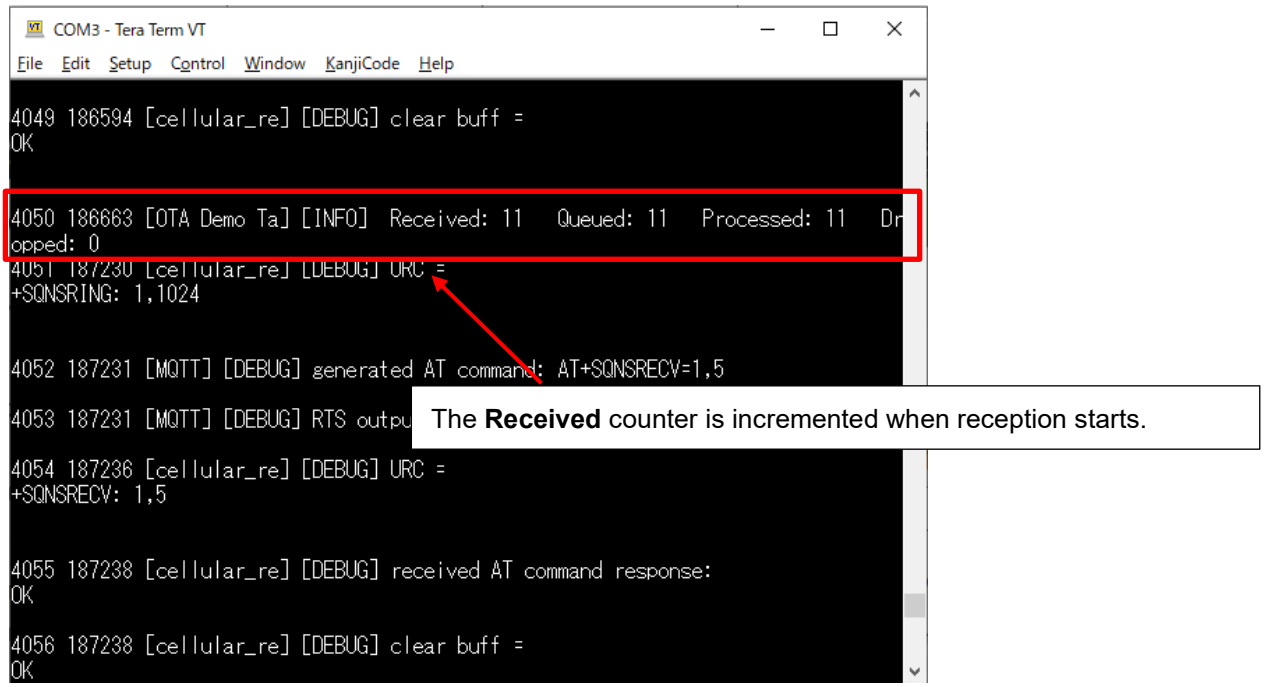


(12) Click **Create job**.

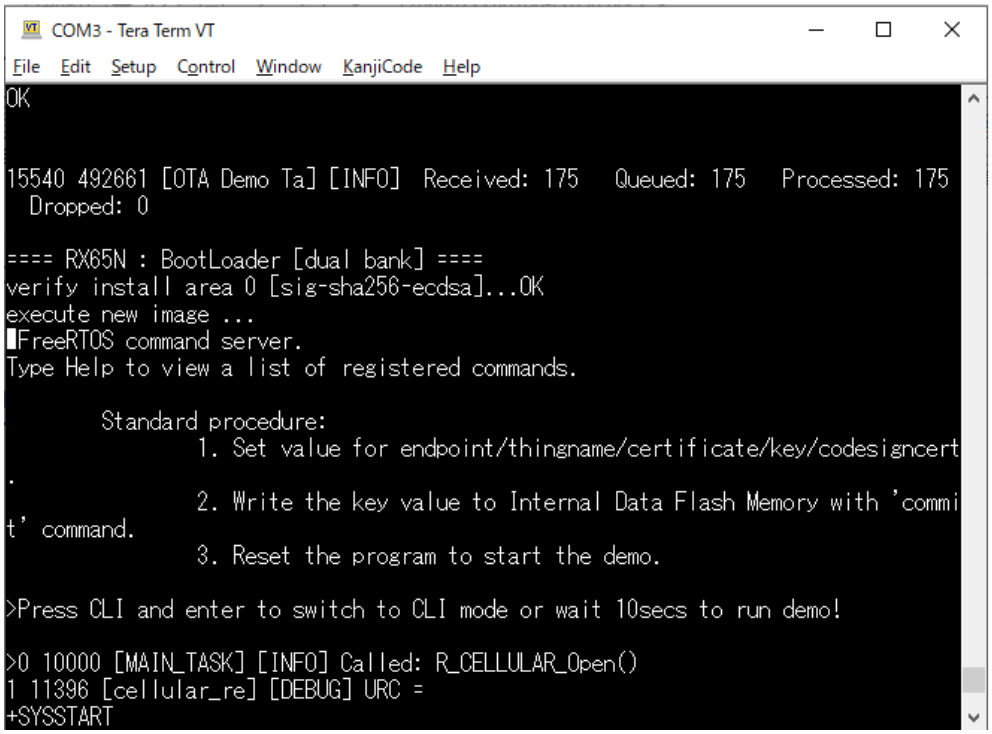


(13) Wait until firmware reception is complete.

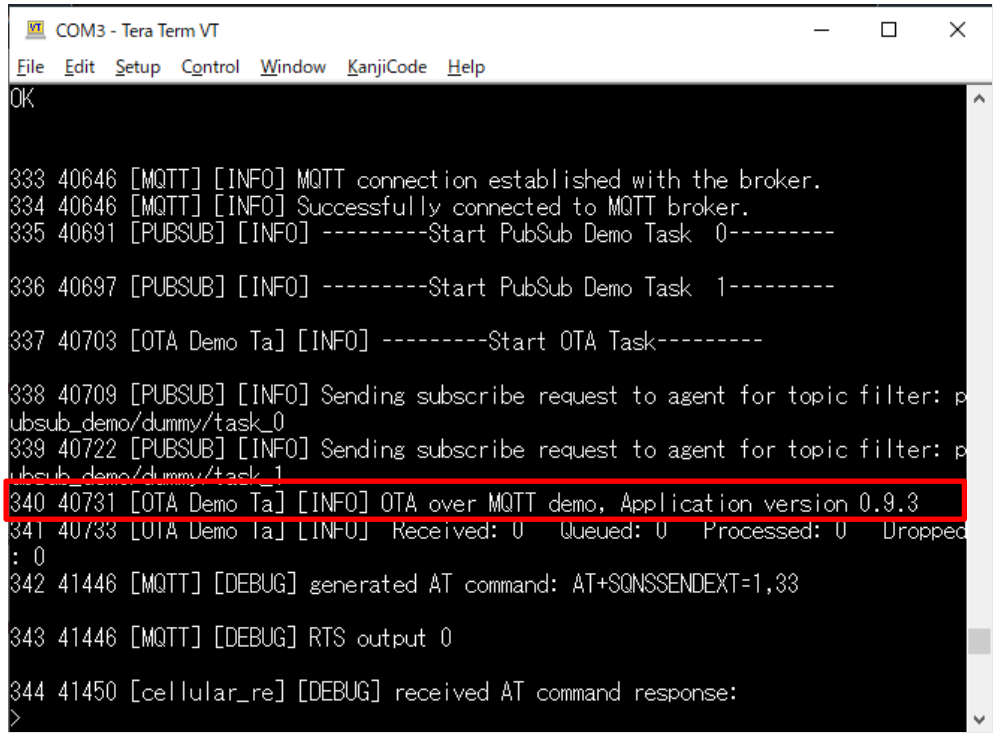
When the job starts, the job receives and writes the firmware.



When the update process is complete, the device resets and the initial menu appears.



(14) Confirm that the firmware version is Ver. 0.9.3.



6. Appendix

6.1 Points to Keep in Mind when Operating Multiple Devices within the Same LAN Environment

Addresses assigned from the header ID of Renesas Electronics Corporation are used as the MAC addresses contained in the sample code.

When using the sample program to operate multiple devices within the same LAN environment, it is necessary to change the MAC addresses to avoid duplication.

The sample program may not operate properly if the same MAC addresses are duplicated among multiple devices.

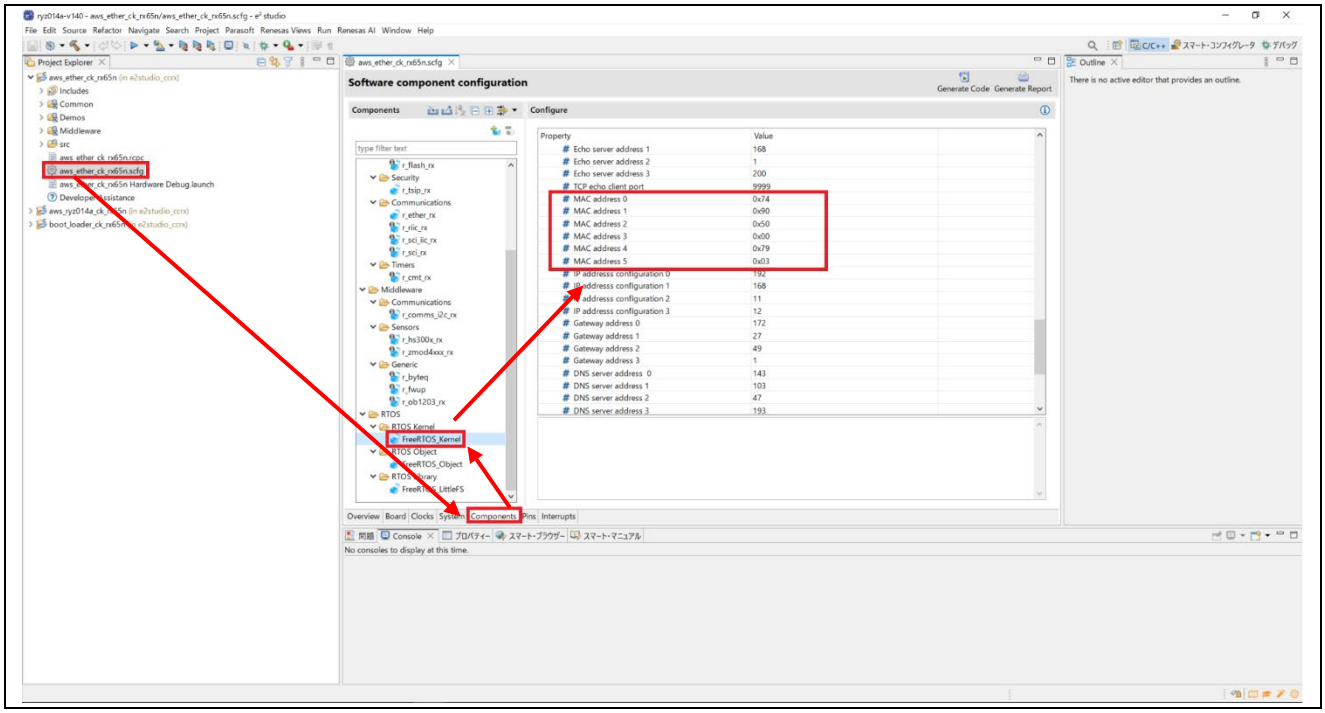
The procedure for changing the MAC addresses is described below.

Open **aws_ether_ck_rx65n.scfg** in Smart Configurator, and select the **Components** tag.

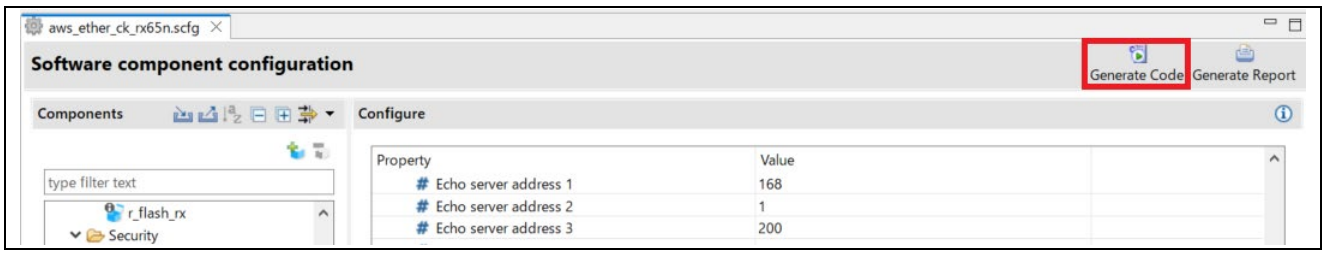
In the tree, select **RTOS** → **RTOS Kernel** → **FreeRTOS_Kernel**, then under **Property** set **Value** for **MAC address 0** to **MAC address 5** to hexadecimal values of your choice.

Enter values in the format **0xXX** (where XX represents a hexadecimal value of your choice).

When commercializing a product for sale, make sure to apply to the IEEE to obtain the MAC addresses eventually used.



After making the above changes, click the **Generate Code** button in the upper right corner of the window to apply the changes made in Smart Configurator to the code.



7. Troubleshooting

The following table lists issues that might arise when executing the sample, and how to resolve them.

Table 7.1 Troubleshooting

No.	Issue	Cause	Solution	Refer to
1	The command to create the initial firmware fails	The Python installation folder is not set correctly in the Path variable	Reinstall python. Also, make sure that the **Add python.exe to PATH** check box is selected when you perform the steps in 2.2(3).	2.2
2		The encryption library is not installed	Install the encryption library.	2.2(5)
3	The initial firmware cannot be written	The CK-RX65N kit is not in debug mode	Make sure that the jumper on J16 of the CK-RX65N board is on pins 1-2 (debug mode).	2.5
4	The initial firmware does not start	The CK-RX65N kit is not in RUN mode	Make sure that the jumper on J16 of the CK-RX65N board is on pins 2-3 (RUN mode).	4.2.5(4)
5	Cellular communication cannot start	The RYZ014A PMOD board is not connected properly	Check the connection of the RYZ014A PMOD board.	2.5
6		No SIM card is inserted	Insert the SIM card.	2.5
7		The SIM card is configured incorrectly	Revise the configuration of the r_cellular module.	4.2.3(4)
8		You are using the SIM card supplied with the CK-RX65N kit but the SIM card is not activated	Activate the SIM card.	4.2.3(4)
9	An error occurs during cellular communication	The communication environment is poor	Connect an antenna and power supply to the RYZ014A PMOD board. Also, place the antenna in an area with good reception such as near a window.	2.5
10	An error occurs when connecting to AWS	The AWS IoT information is not set or is set incorrectly	Set the AWS IoT information again.	4.2.5
11	The firmware does not start after starting the boot loader	The public key is not correctly set in the boot loader	Review the public key setting in the boot loader.	4.2.3(1)
12	The firmware does not start after an OTA update	The public key is not correctly set in the firmware	Review the public key setting in the firmware.	4.2.3(1)
13		Device selection is incorrect	Review the device setting in the firmware and the boot loader.	4.2.3(5) to 4.2.3(9)

Revision History

Rev.	Date issued	Details	
		Page	Nature of revision
1.00	Sep. 15, 2023	—	Initial publication
1.10	Jun. 14, 2024	—	Partial revisions to sections 2.1, 2.4, and 4.2 Addition of section 6

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/.