RX Family

AWS Cloud Connectivity for MCU Firmware Update Over-the-Air on CK-RX65N v2 with Wi-Fi DA16600

Introduction

This document provides instructions for running the AWS Cloud Connectivity for MCU Firmware Update Over-the-Air project on CK-RX65N v2 using Wi-Fi DA16600, utilizing the MCU firmware update command of Wi-Fi DA16600 module.

Target Device

RX Family
- RX600 Series
  - RX65N Group

Hardware

- CK-RX65N v2

When using this application note with other Renesas MCUs, careful evaluation is recommended after making modifications to comply with the alternate MCU.

Note: This demo project has also been successfully reused on the FPB-RX140 board. However, some module configurations must be adjusted to ensure compatibility with the FPB-RX140 device and its memory constraints.

Related Documents

[1] Firmware Integration Technology User's Manual (R01AN1833)
[2] RX Family FWUP Module Using Firmware Integration Technology (R01AN6850)
[3] Renesas MCU Firmware Update Design Policy (R01AN5548)
[4] CK-RX65N v2 – User's Manual (R20UT5366)
[5] US159-DA16600EVZ Evaluation Board Manual (R15UZ0006)

Contents

1 Overview 3
1.1 Workflow 4
1.2 Operation Confirmation Conditions 5
1.3 Equipment List 5
1.4 Sample Project Code Sizes 6

2 Demo Project Setup 7
2.1 Hardware Setup 7
2.2 Software Setup 8
2.2.1 Installing Tool 8
2.2.1.1 Install Python 8
2.2.1.2 Installing OpenSSL 9
2.2.1.3 Installing Renesas Image Generator 11
2.2.1.4 Installing Tera Term 12
2.2.1.5 Installing Renesas Flash Programmer 13
2.2.2 Terminal Software Setting 14
2.2.3 Check DA16600 Wi-Fi SDK Version 18
2.2.4 Generate Key Pairs and Certificates 19
2.2.5 Project Description 20
2.2.6 Importing The Demo Project 21
2.2.7 Create Firmware File and Upload to the Amazon S3 Bucket 24
2.2.7.1 Create Firmware File 24
2.2.7.2 Upload Firmware to Amazon S3 Bucket 28
2.2.8 Create Firmware Initialization 35
2.2.8.1 Building CK-RX65N Application Project 35
2.2.8.2 Building CK-RX65N Bootloader Project 36
2.2.8.3 Creating the Initial Firmware 37

3 Execute the Demonstration Project 38

4 Appendix 44
4.1 Known Issues for DA16600 44
4.1.1 Firmware Size Limit Causes Update Failure 44
4.1.2 Resolution 44
4.2 Debugging 46

Revision History 47

1 Overview

This demo project shows the integration of AWS Cloud Connectivity for MCU Firmware Update Over-the-Air on the CK-RX65N v2 development board, utilizing the MCU firmware update command of Wi-Fi DA16600 module for seamless wireless communication.

Key Steps in the Project:

- **Prepare an AWS Account and S3 Bucket:** Set up an AWS account and create an S3 bucket to serve as the server for uploading firmware.
- **Generate Key Pairs and Certificates:** Generate a public key and a private key to create both the initial firmware and the firmware used for updates.
- **Generate and Upload the Firmware File:** Generate the new firmware to be flashed onto the board via OTA and upload it to the Amazon S3 bucket.
- **Generate the initial firmware:** Configure and build the initial firmware, then generate the firmware file using the designated tool.
- **Execute the Demonstration Project:** Execute the demo project to validate the OTA process.

The following section provides a comparison of Firmware Update Over-the-Air (OTA) demonstrations using two approaches.

- DA16600 Wi-Fi Module with OTA On-Chip Commands:

The DA16600 module connects to the AWS server and downloads the firmware directly.

The downloaded firmware is then transferred to the MCU host for deployment.

In this demonstration, HTTP and firmware updates are offloaded, the firmware file URL is retrieved using a web browser, TLS is not used in this demonstration. TLS is planned for the next demonstration.

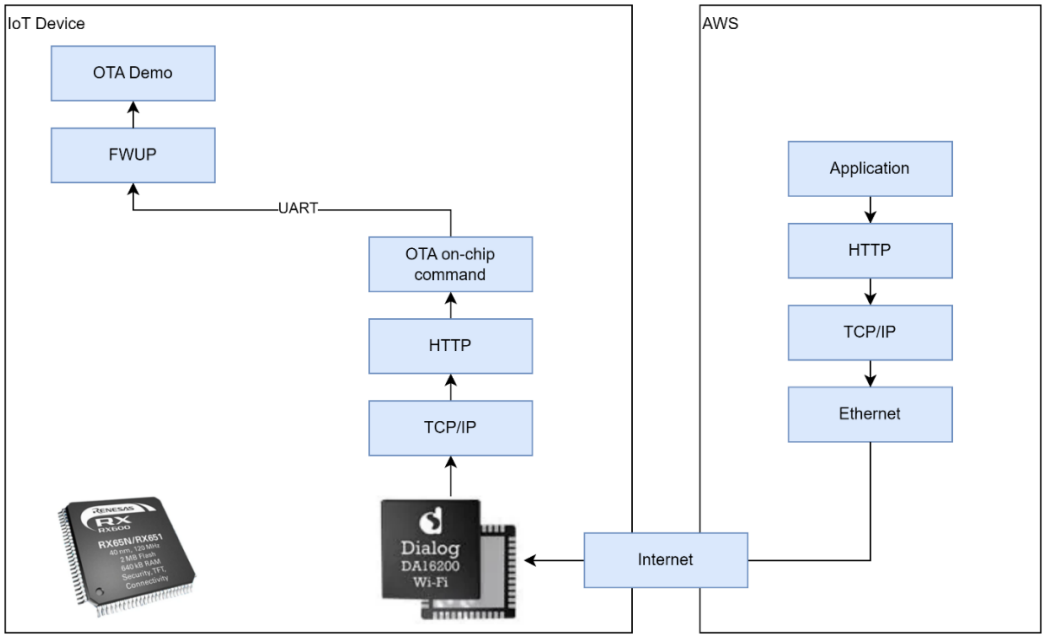


Figure 1.1 OTA Demonstrates with Wi-Fi DA16600 OTA On-Chip Commands

1.1 Workflow

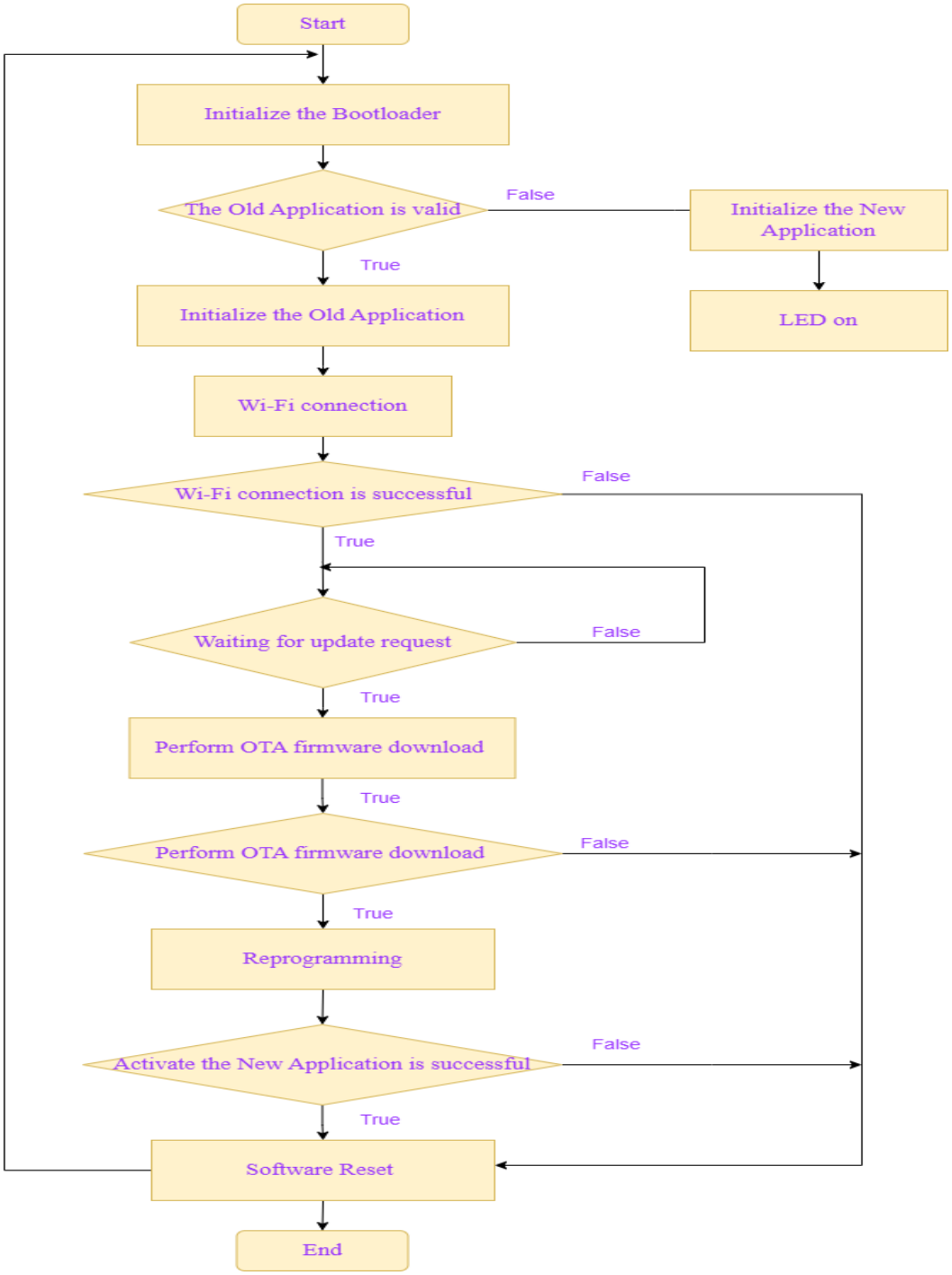


Figure 1.2 End-to-End OTA Workflow

1.2 Operation Confirmation Conditions

Demo project operations have been confirmed in the following conditions.

Table 1.1 Operation Confirmation Conditions

Item	Description
MCU	R5F565NEHxFB
Board	CK-RX65N v2 (Product no.: RTK5CK65N0S04000BE)
IDE (Integrated Development Environment)	Renesas Electronics e2 studio 2024-07
C compiler	Renesas Electronics CC-RX V3.06
Firmware programming tool	Renesas Flash Programmer V3.15.00
Firmware update module	RX Family FWUP Module Using Firmware Integration Technology R01AN6850EJ0204
Python	Python 3.12.7
Keygen tool	Win64 OpenSSL v3.0.12
SDK (Software Development Kit)	DA16200/DA16600 SDK V3.2.9.2

1.3 Equipment List

The following lists the equipment required for the demo projects.

Table 1.2 Equipment List

Item	Description
Board	CK-RX65N v2 CK-RX65N - Cloud Kit Based on RX65N MCU Group
Wi-Fi DA16600 module	PMOD Expansion Board for DA16600MOD US159-DA16600EVZ - Ultra-Low-Power Wi-Fi + Bluetooth Low Energy Combo Pmod Board
UART-TTL	CP2101 USB TO TTL BOARD V4.2 CP2101 USB TO TTL BOARD V4.2 (agencyelectronics.com)
Micro USB Type-B cable x 2	• Connect the designated USB port on the base board to a PC for serial logging. • Connect another USB port on the base board to a PC for debugging purposes.
Jumper pin x 3	It is used to enable debugging mode.

1.4 Sample Project Code Sizes

The tables below show the ROM, RAM, and maximum stack sizes for the sample projects included in the package associated with this application note. The values in the table below have been confirmed under the following conditions:

Compiler version: Renesas Electronics C/C++ Compiler for RX Family V3.06.00

CC-RX

- Optimization level: Size and execution speed (-Odefault)
- Delete variables/functions that have never been referenced (optimize=symbol_delete)

Table 1.3 ROM, RAM, and Maximum Stack Sizes for Sample Project

ROM, RAM, and Stack Codesize			
Device	**Category**	**Memory Used (byte)**	**Remarks**
RX65N	ROM	80267	ck_rx65n_v2_app
	RAM	44193	
	Stack	548	
	ROM	18982	ck_rx65n_v2_app_new
	RAM	8398	
	Stack	52	
	ROM	34198	ck_rx65n_v2_bootloader
	RAM	16279	
	Stack	164	

2 Demo Project Setup

2.1 Hardware Setup

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

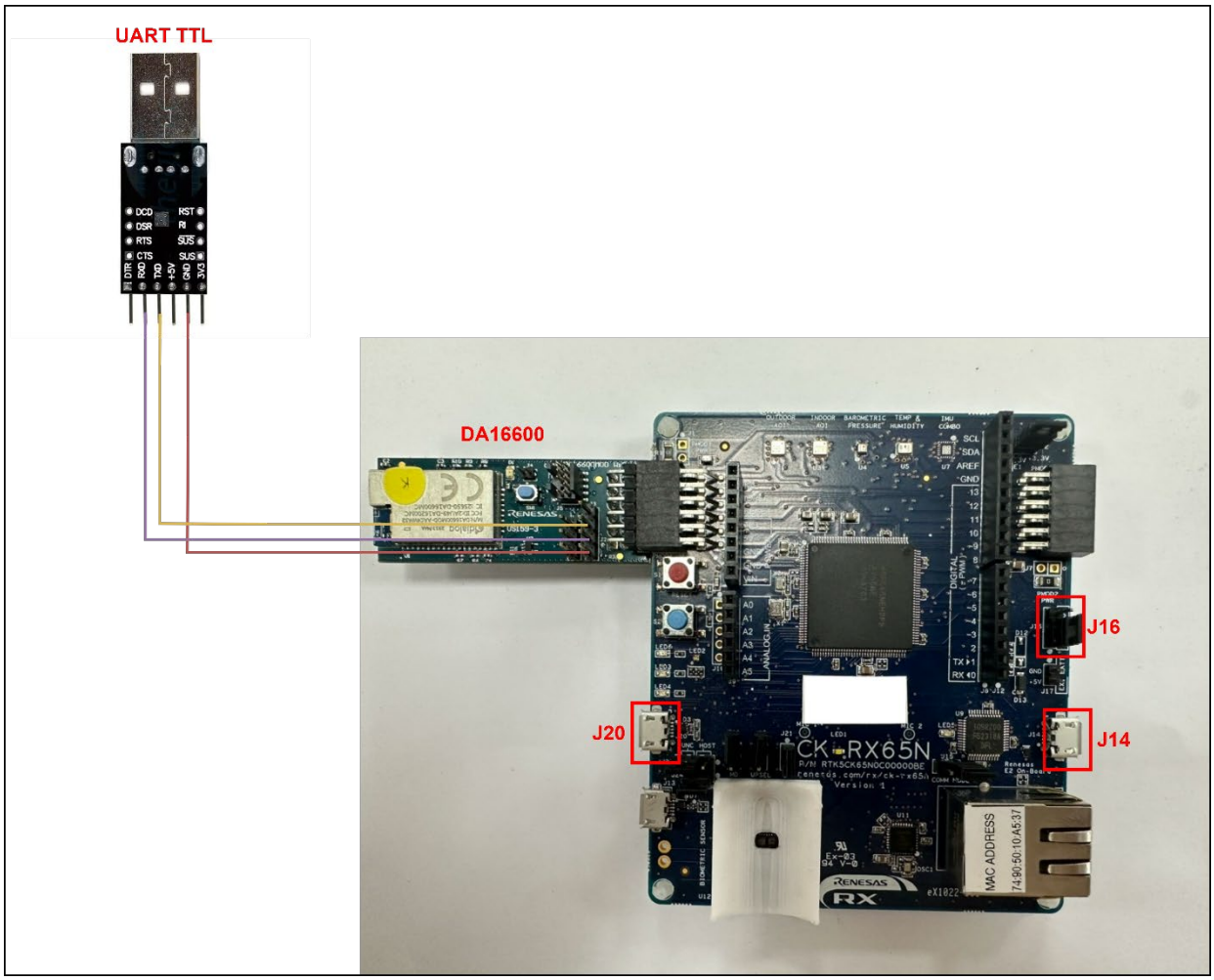


Figure 2.1 Hardware Connection

The following describes how to connect hardware.

- On the base board, position the jumper on **pins 1-2** of **J16** to enable debugging mode.
- Connect the **DA16600 PMOD** to **PMOD1** on the base board.
- Connect **J20** on the base board to a PC using a USB cable for serial log.
- Connect **J14** on the base board to a PC using a USB cable for debugging.
- Connect the **TXD pin** of the **UART TTL** to **pin 2** of **J3** of the DA16600 Wi-Fi module.
- Connect the **RXD pin** of the **UART TTL** to **pin 3** of **J3** of the DA16600 Wi-Fi module.
- Connect the **GND pin** of the **UART TTL** to **pin 4** of **J3** of the DA16600 Wi-Fi module.

2.2 Software Setup

2.2.1 Installing Tool

2.2.1.1 Install Python

Python generates initialization firmware from bootloader and application projects, and application firmware from the new application project.

Follow the steps below to install Python:

(1) Access the Python download web site.

Download Python | Python.org

(2) Download the Python 3.12.7 installer.

Click the **Download** link for Python 3.12.7.

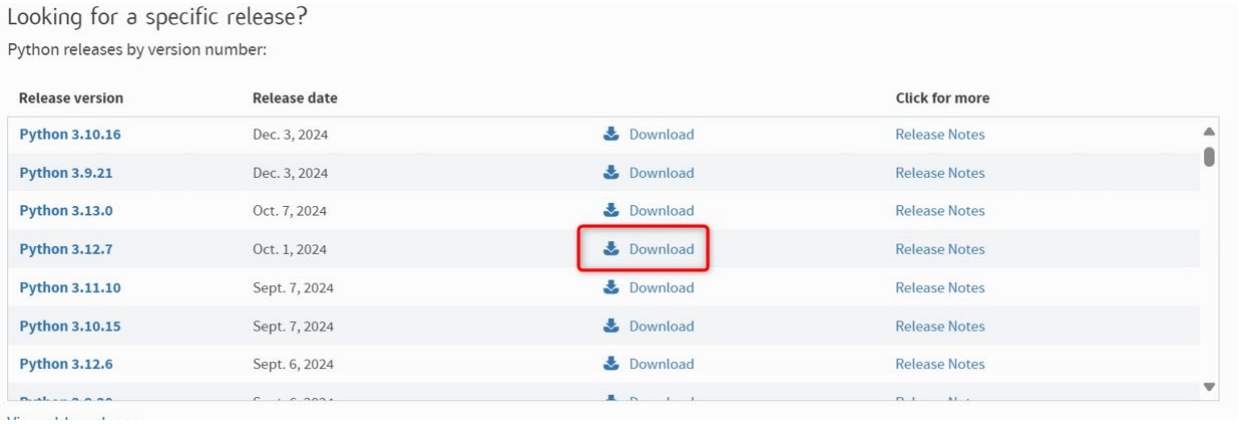


Figure 2.2 The Options for Installing the Release Version of Python

Download the installer for the operating system you are using.

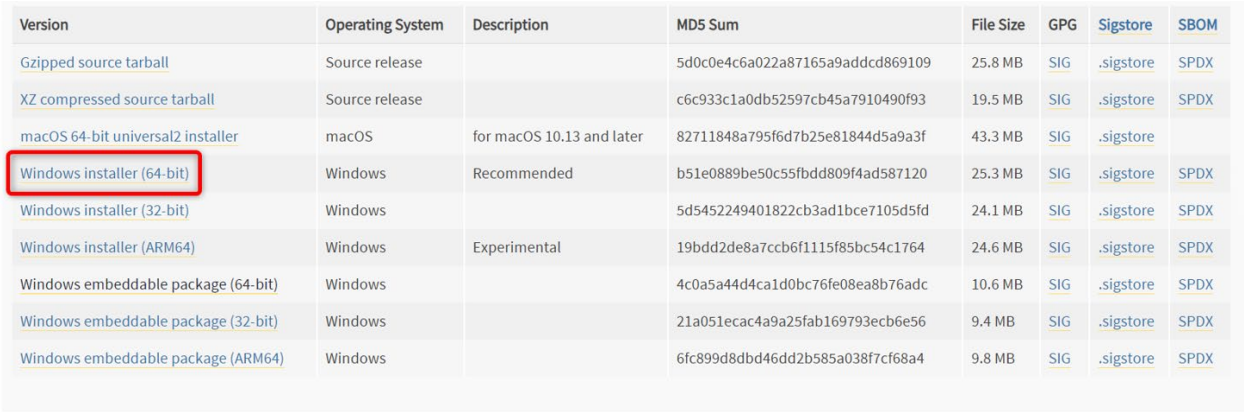


Figure 2.3 Python Windows Installer

2.2.1.2 Installing OpenSSL

OpenSSL is a tool used to generate the cryptographic key pair required for firmware encryption and decryption during both initialization and application firmware creation. OpenSSL can generate the following keys for use in the firmware update process:

- **Private key**: Used to encrypt the firmware and ensure its integrity.
- **Public key**: Used by the bootloader to decrypt and verify the firmware during update.

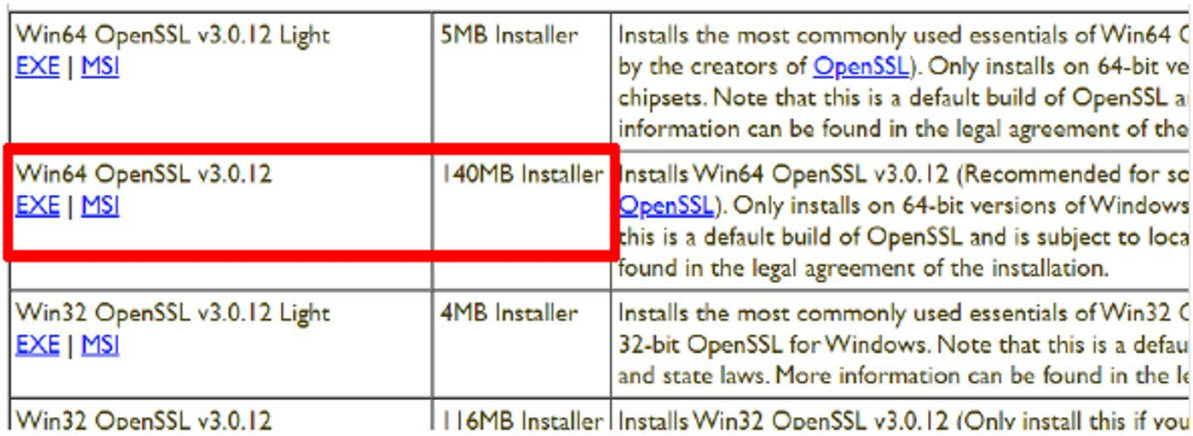
Follow the steps below to install and configure OpenSSL for this purpose.

(1) Access the Win32/Win64 Download Website for OpenSSL

Win32/Win64 OpenSSL Installer for Windows - Shining Light Productions

(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 C 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 C 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

Figure 2.4 The Options for Installing the Release Version of OpenSSL

(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.

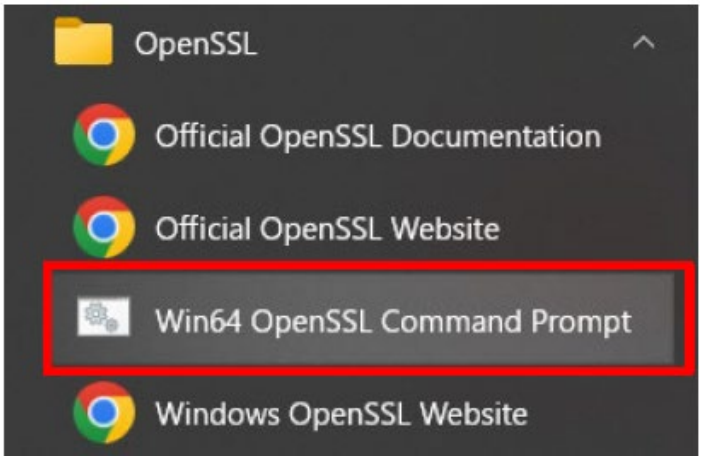


Figure 2.5 OpenSSL Windows (64-bit)

(5) Confirm the OpenSSL Command from the Command Prompt.

Execute the following command and confirm that version information appears.

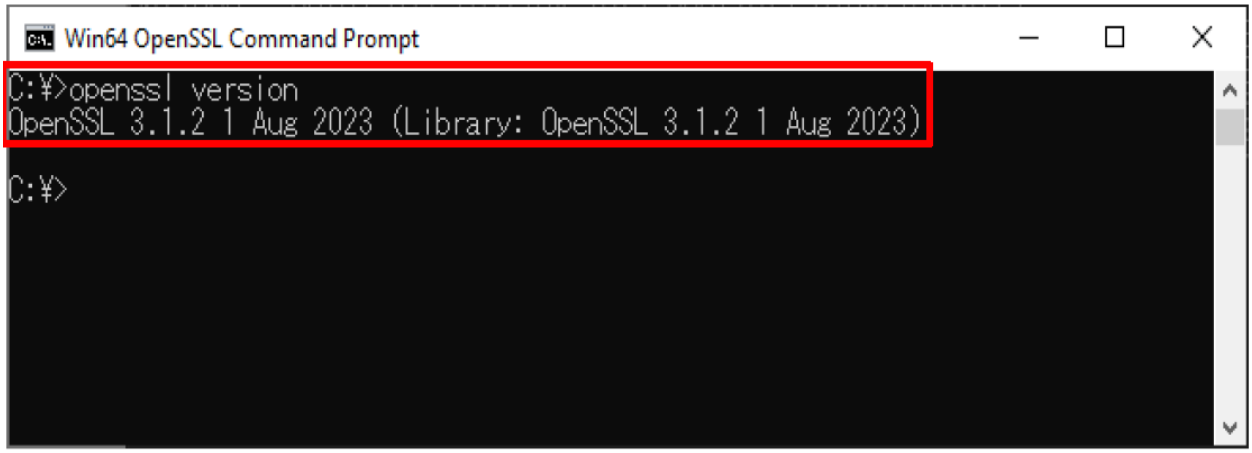


Figure 2.6 Checking OpenSSL Version

2.2.1.3 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.

(1) Download the Firmware Update Module:

RX Family Firmware Update module

(2) Extract the Downloaded Firmware Update Module

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).

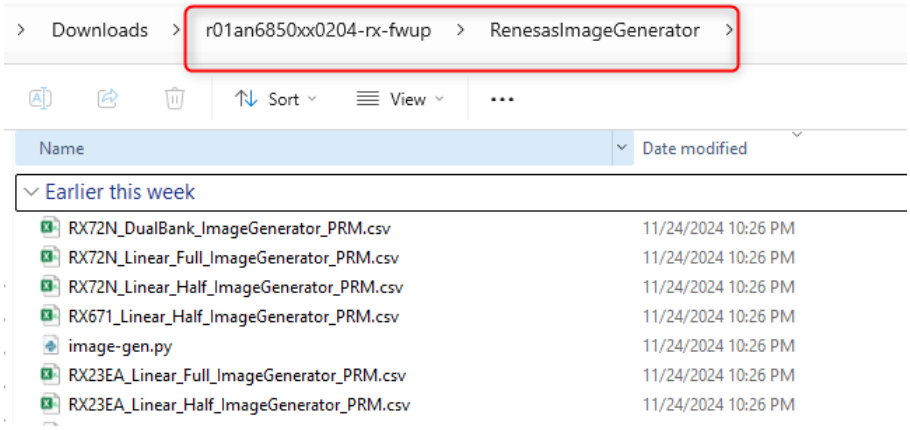


Figure 2.7 Renesas Image Generator Package

2.2.1.4 Installing Tera Term

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

(1) Access the Tera Term Download Site.

Releases · TeraTermProject/osdn-download · GitHub

(2) Download the Tera Term Installer.

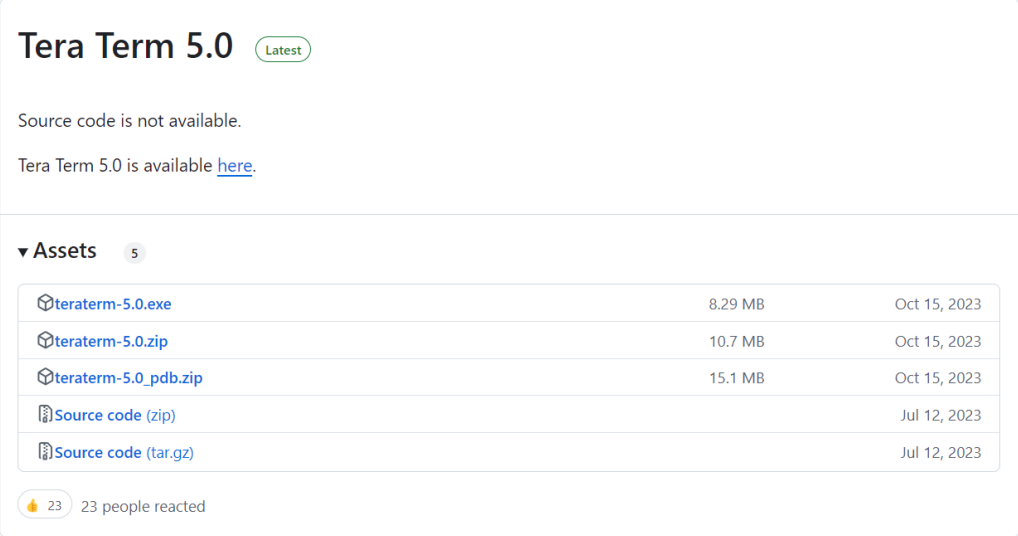


Figure 2.8 Tera Term Version 5.0

- Run the installer and follow the prompts to install Tera Term.
- Confirm that Tera Term starts when you click the Tera Term icon in the Start menu.

2.2.1.5 Installing Renesas Flash Programmer

Renesas Flash Programmer (RFP) is a utility provided by Renesas that allows users to write firmware to support Renesas MCUs via various interfaces such as USB, UART, or serial programming. It is an essential tool for flashing both the initial firmware and subsequent updates during development and production. Follow the steps below to install Renesas Flash Programmer on your computer.

(1) Access the Renesas download web site.

Renesas Flash Programmer (Programming GUI)

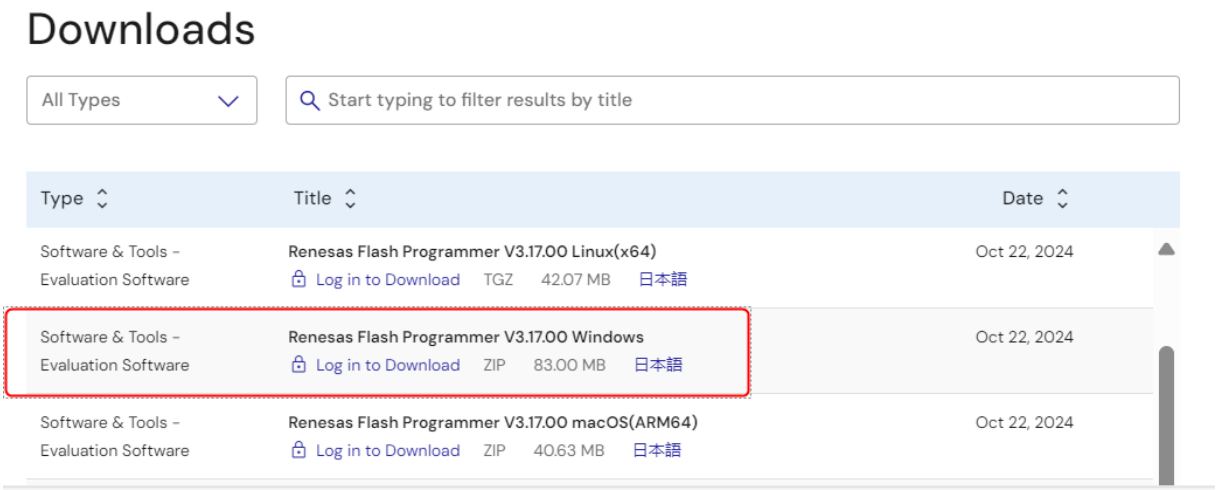


Figure 2.9 Renesas Flash Programmer

2.2.2 Terminal Software Setting

❖ With the UART TTL connection port of the DA16600:

(1) Open Tera Term select **New connection** and select Serial and the appropriate COM port for your **UART-to-USB** adapter, and **click OK**

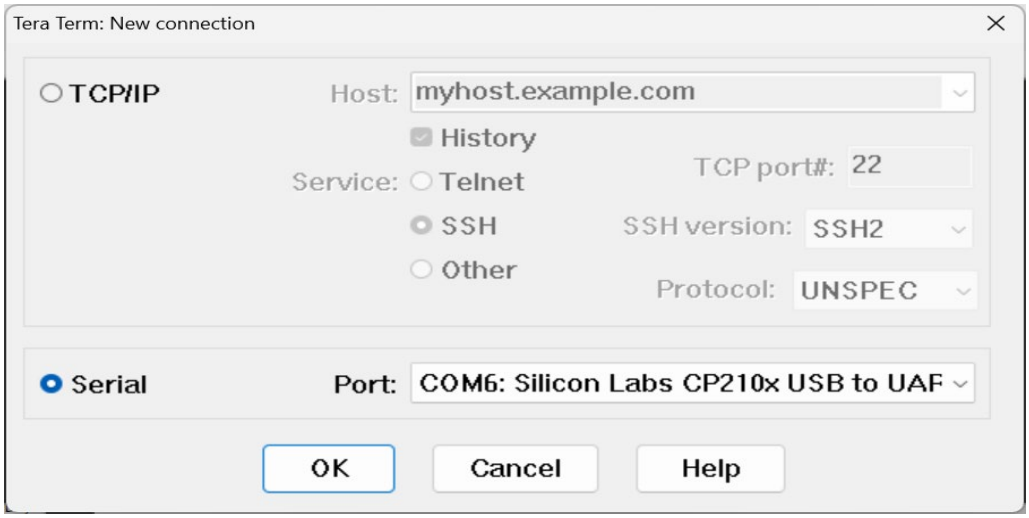


Figure 2.10 Tera Term Serial Connection

(2) Click **Setup > Terminal**…, in "**New-line**" section, set **"Receive**" as **AUTO**.

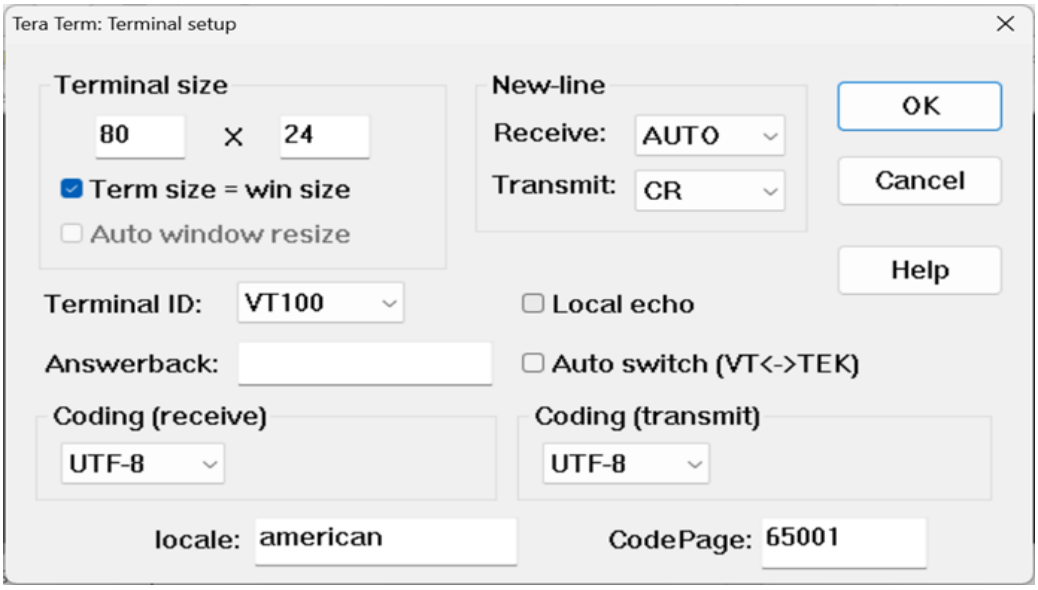


Figure 2.11 Terminal Setup for the UART TTL

(3) Click **Setup > Serial port…** and ensure that the speed is set to **230400**.

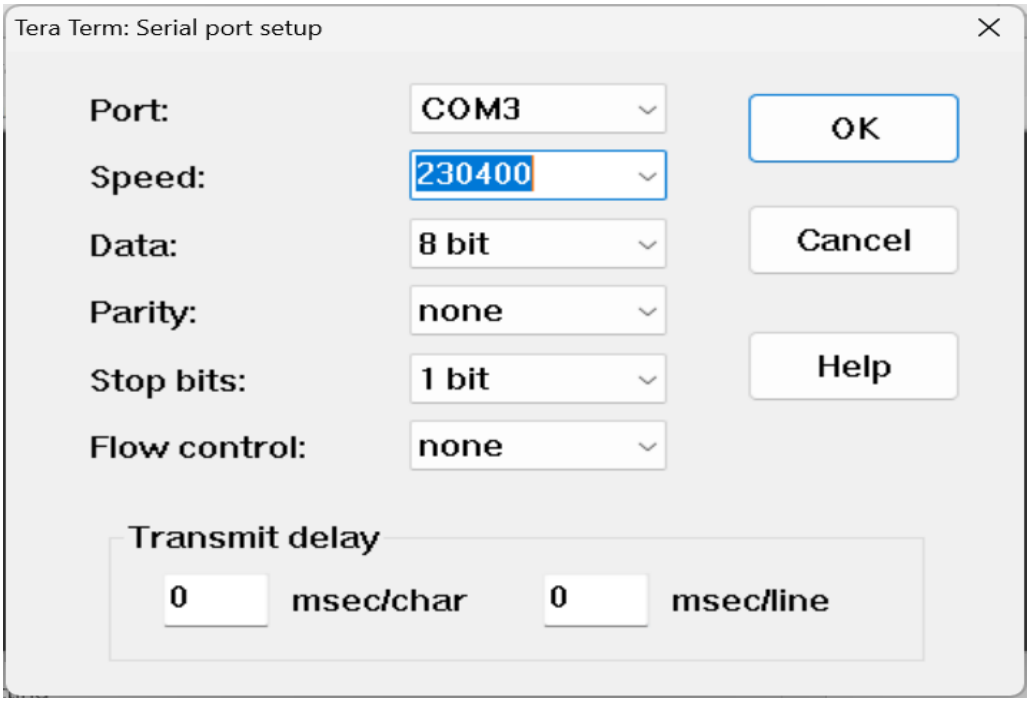


Figure 2.12 Serial Port Setup for UART TTL

(4) Verify the display output on the terminal.

Pressing **Enter** on the terminal will display the line ***[/DA16600] #*** on the screen.

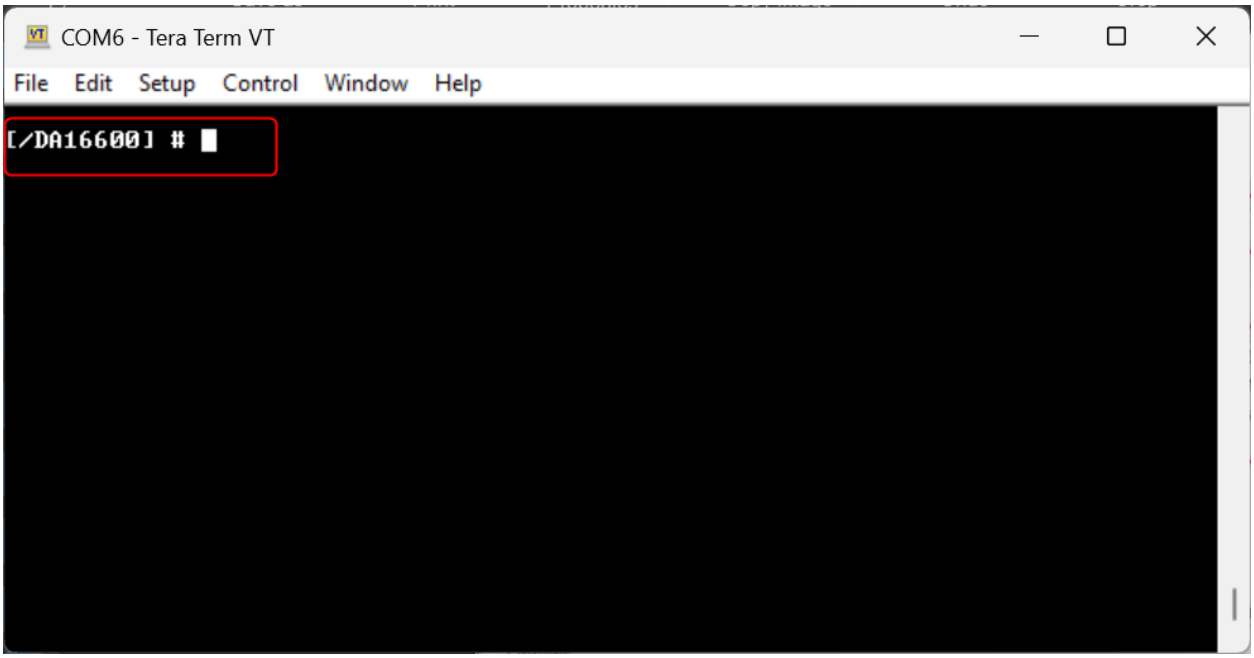


Figure 2.13 UART TTL Terminal

❖ With the USB serial connection of CK-RX65N v2:

(1) Open an additional Tera Term window, **select New Connection**, then choose Serial and the correct COM port for your **USB Serial** device, and **click OK.**

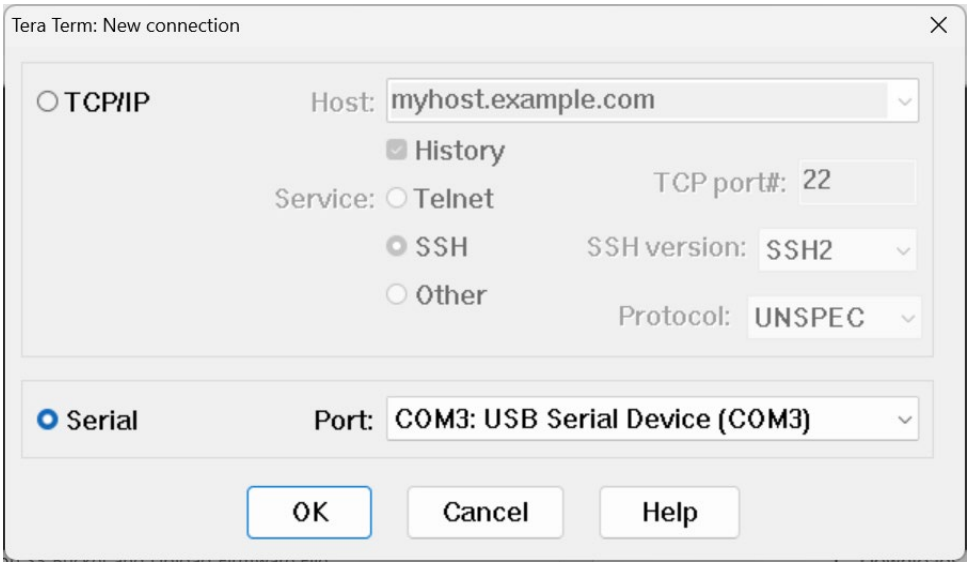


Figure 2.14 Select USB Serial Port

(2) Click **Setup > Terminal**…, select "**New-line Receive**" as **AUTO**. and tick the "**Local echo**" option.

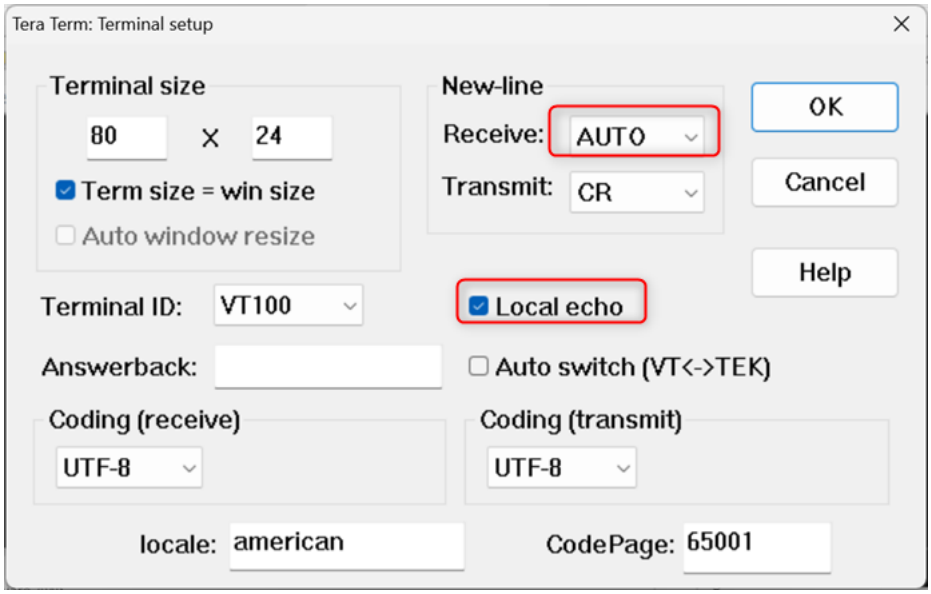


Figure 2.15 Terminal Setup

(3) Click **Setup > Serial port…** and ensure that the speed is set to **115200**.

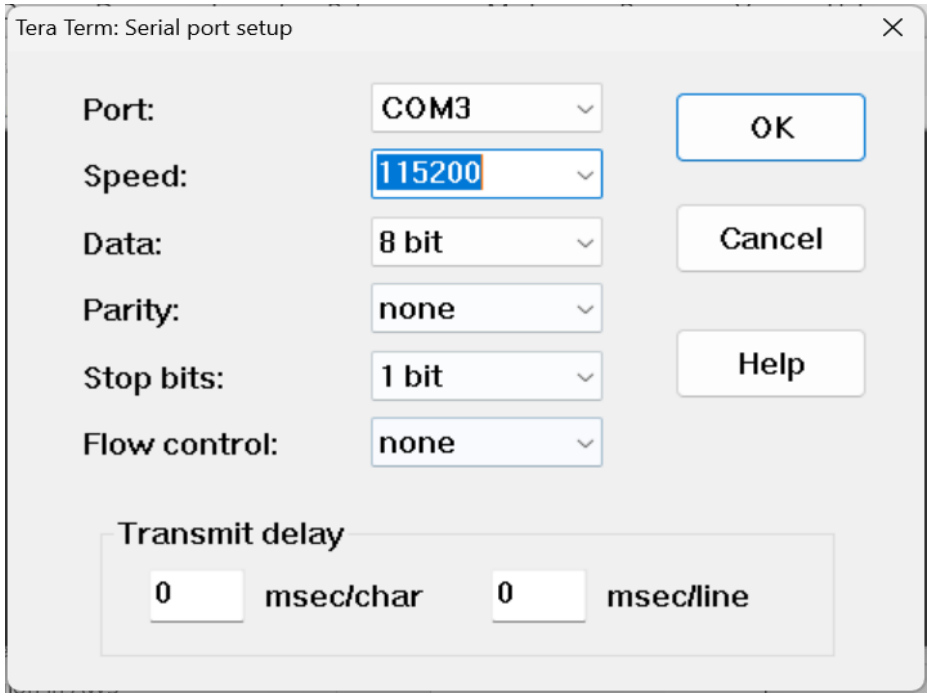


Figure 2.16 Terminal Setup for USB Serial Port

(4) Verify the display output on the terminal.

The terminal output will appear here during the demo execution.

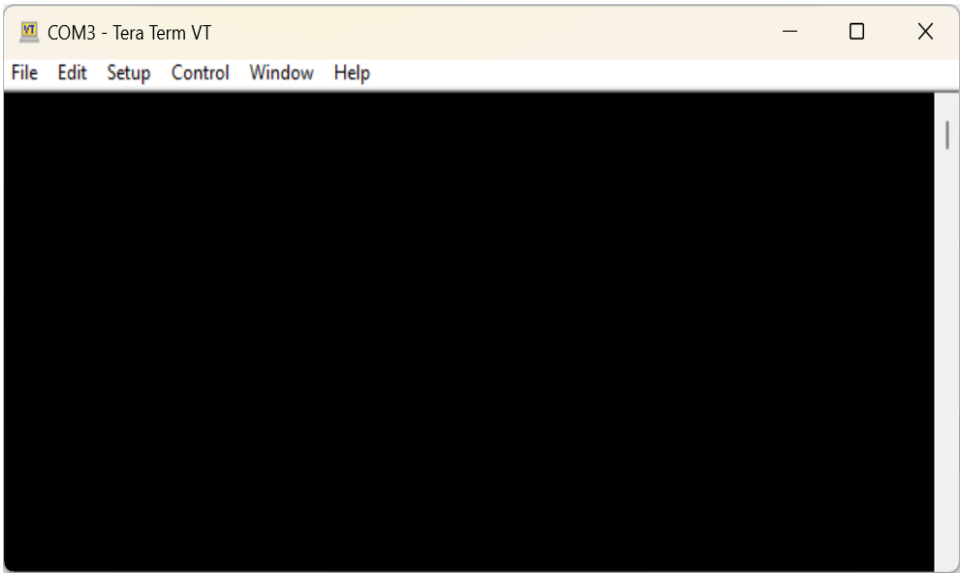


Figure 2.17 CK-RX65N v2 Serial Terminal

2.2.3 Check DA16600 Wi-Fi SDK Version

(1) On the UART TTL terminal in Tera Term, type the command **"ver".**

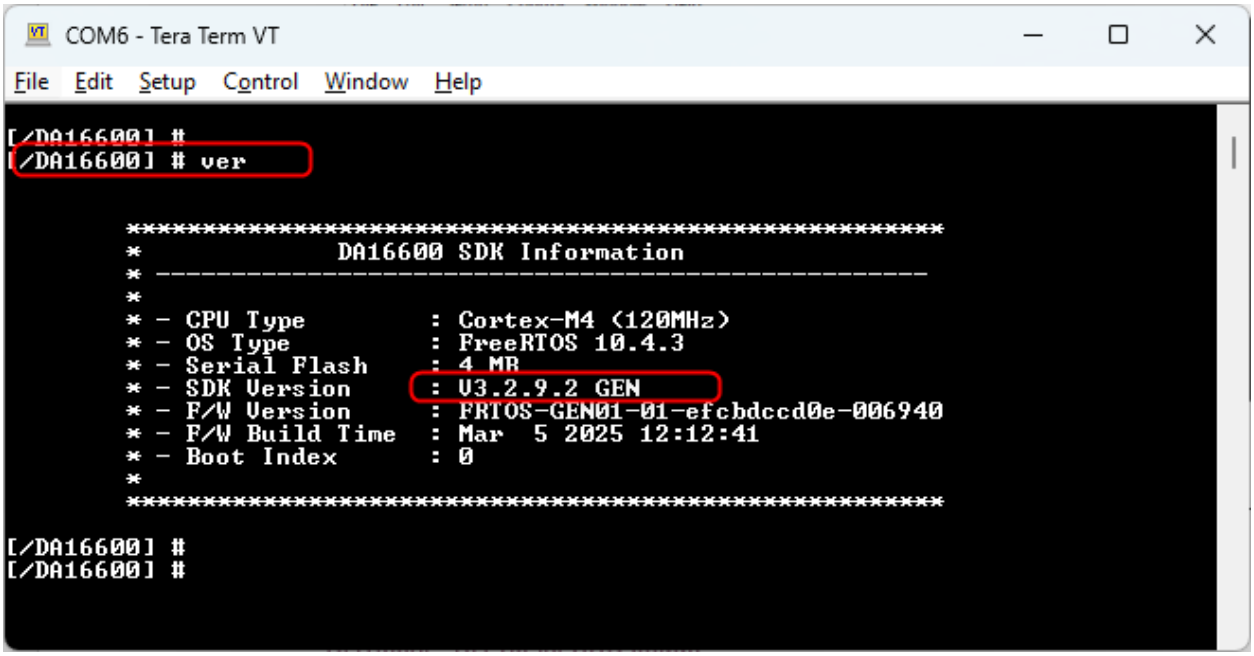


Figure 2.18 Check SDK Version

(2) Check the SDK version. If the current version is **v3.2.9.2** or higher, you can proceed to the next step; otherwise, follow the steps below to upgrade:

- Download the firmware using this link.
  DA16600 - DA16200 SDK Image.
- Flash the new firmware via Tera Term by following the instructions provided in the UM-WI-056 DA16200 DA16600 FreeRTOS Getting Started Guide under **Section 4.5.2 Using Macro Script of Tera Term**.

2.2.4 Generate Key Pairs and Certificates

This section will generate a public key and a private key to create the initial firmware and the firmware used for updates.

To do this, open OpenSSL and enter the commands highlighted in yellow to generate the firmware verification keys.

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

using curve name prime256v1 instead of secp256r1

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

read EC key

writing EC key

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

read EC key

writing EC key
```

2.2.5 Project Description

The demo project is structured into multiple components to support the firmware update process. Each component plays a specific role in demonstrating the OTA update mechanism on the RX65N platform.

- **ck-rx65n-app**: The existing application that the MCU runs **before** the firmware update process begins.
- **ck-rx65n-bootloader**: The bootloader responsible for handling the firmware update process.
- **ck-rx65n-app-new**: The new application that the MCU runs after the firmware update is successfully completed.

Figure 2.19 The Project Folder Structure Consists of Three Main Components

2.2.6 Importing The Demo Project

(1) Clone the demo project

(2) Extract the demo project

(3) Start e2 studio

(4) From the **File** menu, select **Import**

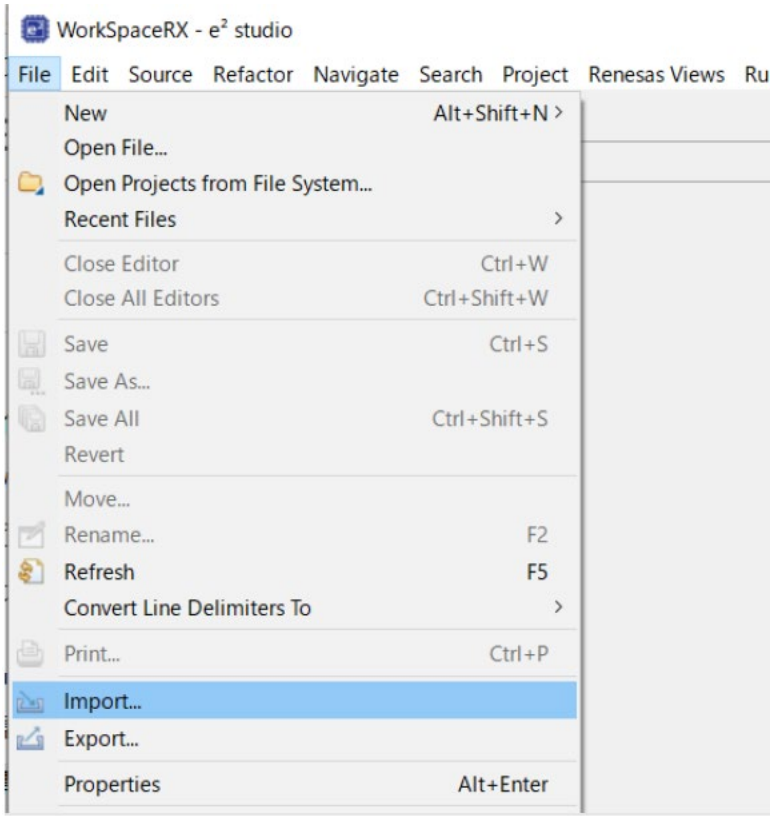


Figure 2.20 Importing the Project

(5) Select **Existing Projects into Workspace**

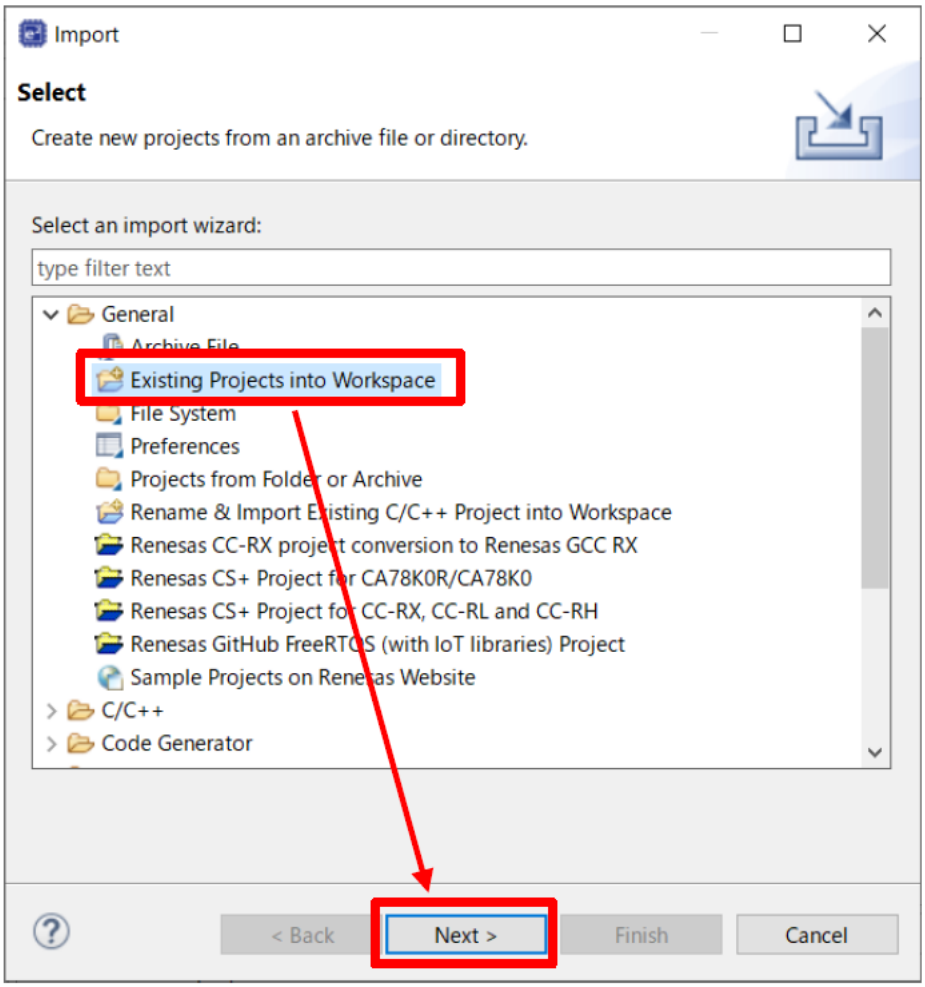


Figure 2.21 Select Existing Projects into Workspace

(6) In **Select root directory**, select the folder extracted, select the check boxes for the following projects, and then click **Finish**

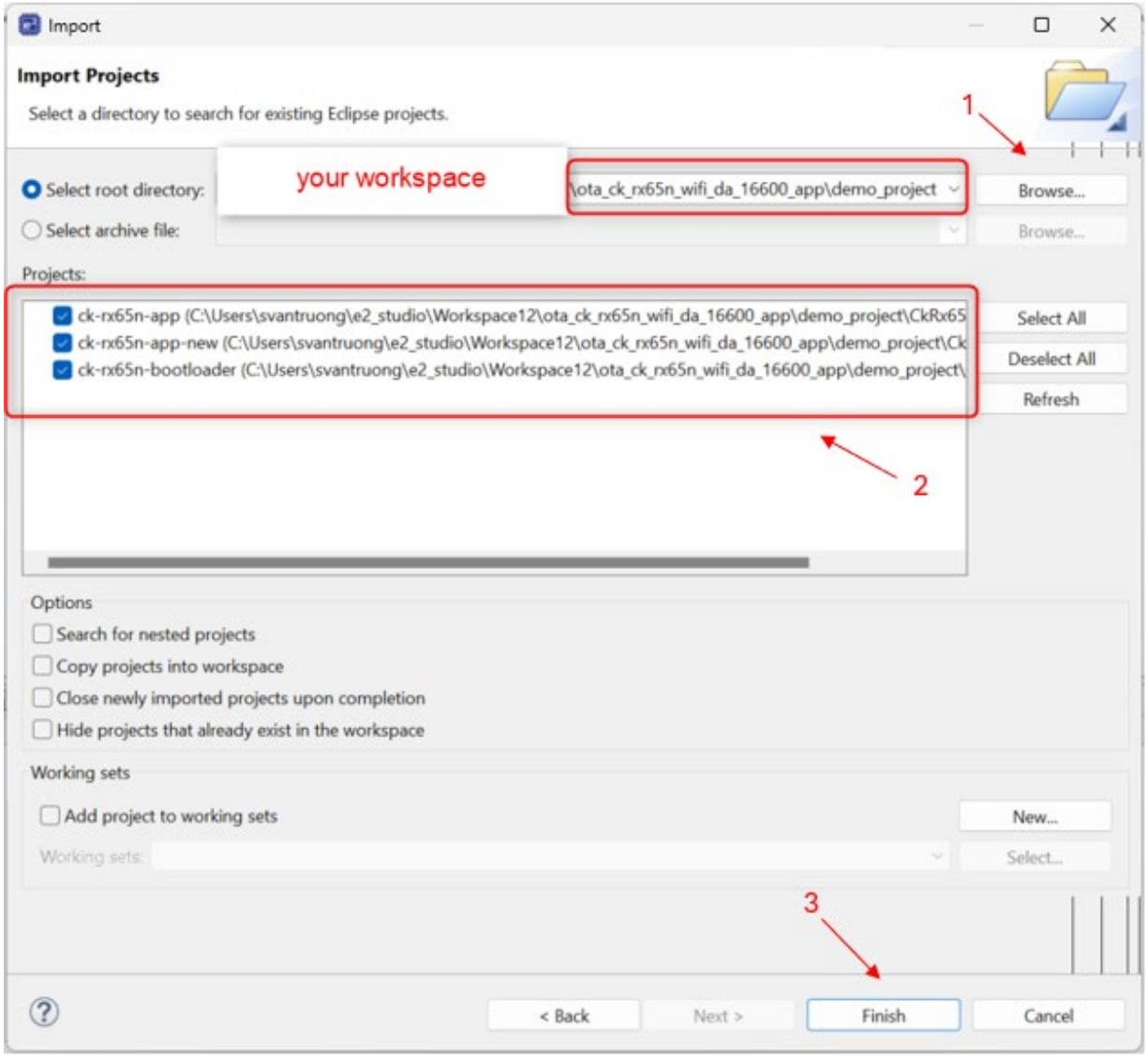


Figure 2.22 Complete Import Project

2.2.7 Create Firmware File and Upload to the Amazon S3 Bucket

2.2.7.1 Create Firmware File

(1) Build the **ck-rx65n-app-new** project.

For guidance on building this project, please refer to section 4.2 'Build a Sample Project' in the *E2 Studio IDE User's Manual: Quick Start Guide* (R20UT5293).

(2) Copy the file **ck-rx65n-app-new.mot** from the folder **\ota_ck_rx65n_wifi_da_16600_app\demo_project\ck-rx65n-app-new\HardwareDebug** to the RenesasImageGenerator folder that you downloaded in **Section 2.2.1.3**.

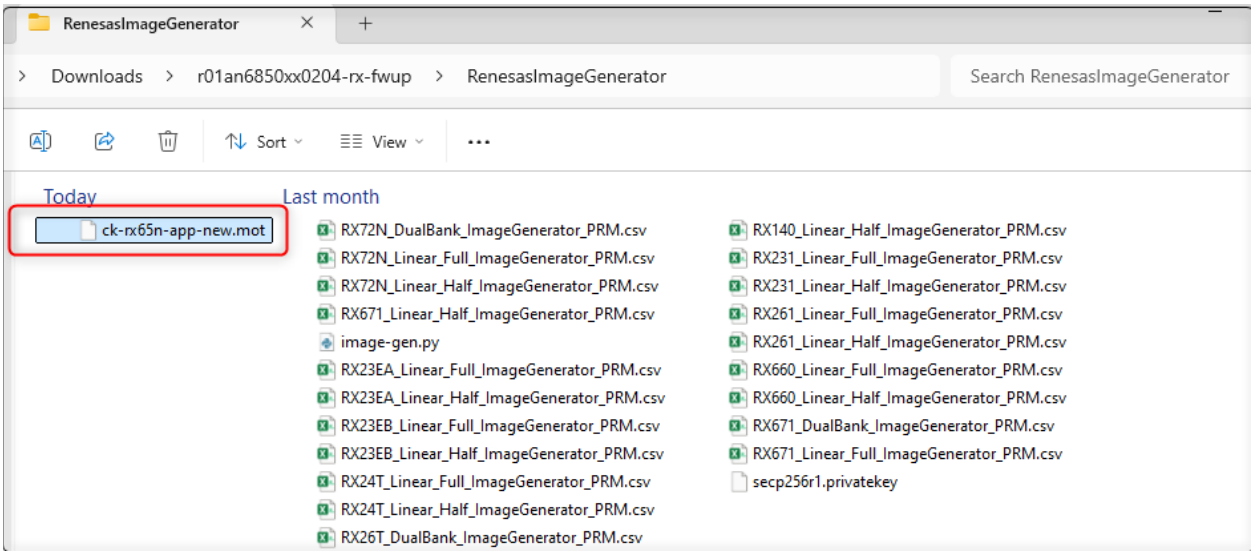


Figure 2.23 Put MOT File to the RenesasImageGenerator Folder

(3) Copy the **secp256r1.privatekey**, which is used to sign the image for authenticity and was generated in **Section 2.2.4** to the **RenesasImageGenerator** folder that was downloaded in **Section 2.2.1.3**.

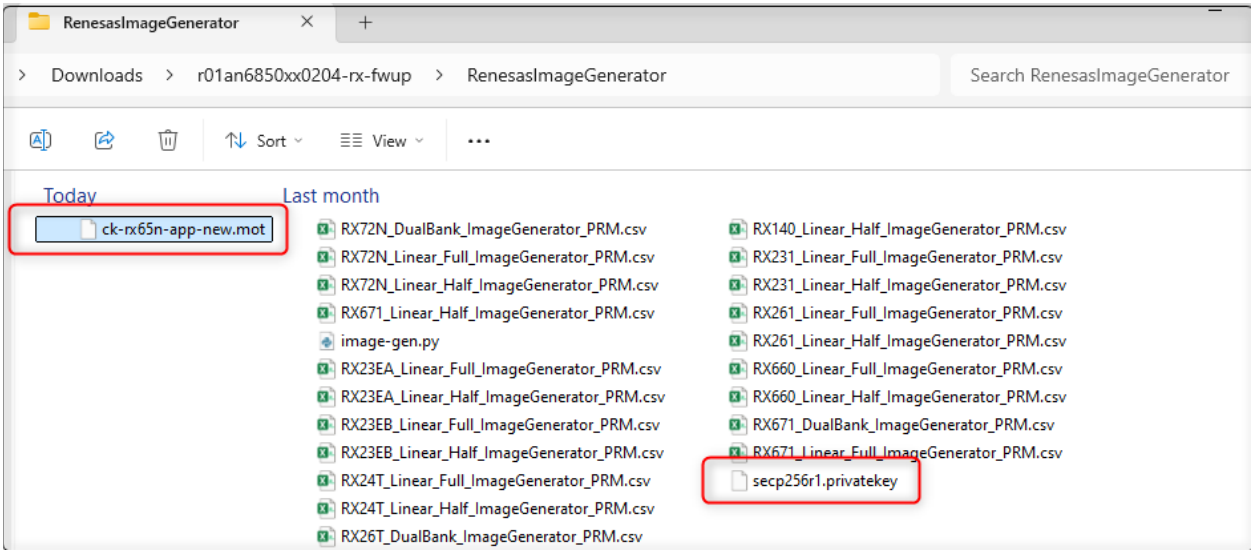


Figure 2.24 Put Private Key to the RenesasImageGenerator Folder

(4) Open a command prompt, navigate to the **Renesas Image Generator** folder, and run the command below to install the **pycryptodome** library — required for encryption and signing operations during firmware processing (skip if already installed):

```
pip install pycryptodome
```

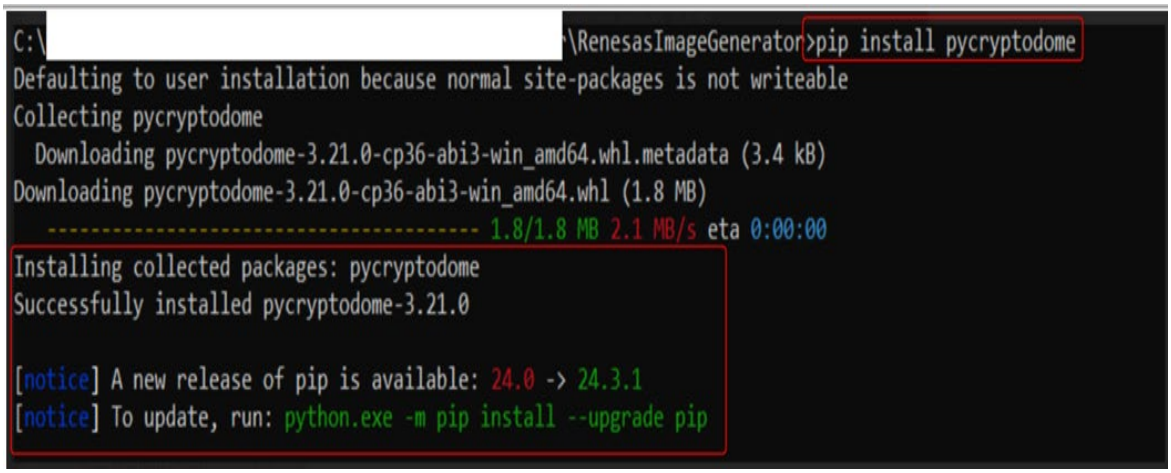


Figure 2.25 Install Python Library

(5) Open a command prompt, navigate to the Renesas Image Generator folder, and execute the following command to generate the file **app_new.rsu**. This file will be uploaded to the bucket as the new firmware that the device will download.

```
python image-gen.py -iup ck-rx65n-app-new.mot -ip RX65N_Linear_Full_ImageGenerator_PRM.csv ^
-o app_new -key secp256r1.privatekey -vt ecdsa -ff RTOS
```

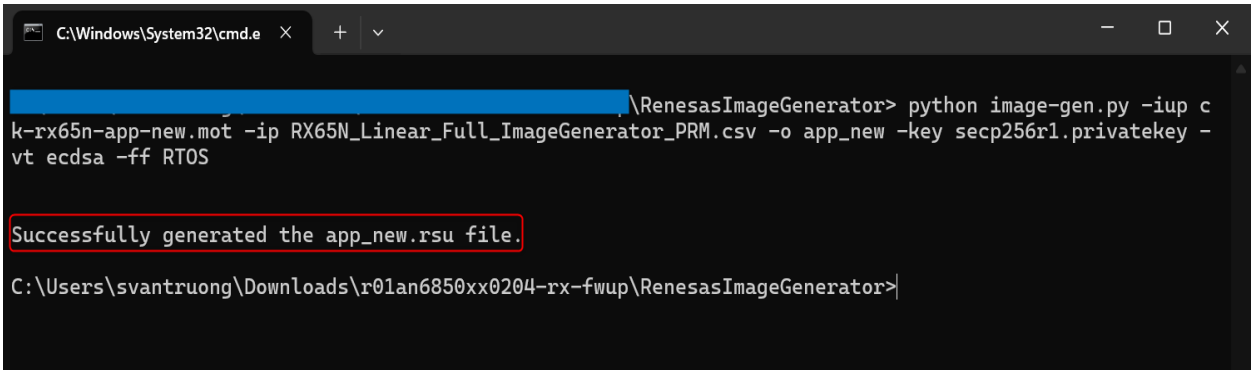


Figure 2.26 Generate the app_new.rsu File

(6) Get the size of the **app_new.rsu** file

This parameter is used to configure the firmware size and will be referenced in **Section 2.2.8.1.**

Follow the steps below to obtain accurate results:

- Right-click on the file **app_new.rsu**.
- Select **Properties**.
- In the **General** tab, you will see **Size**

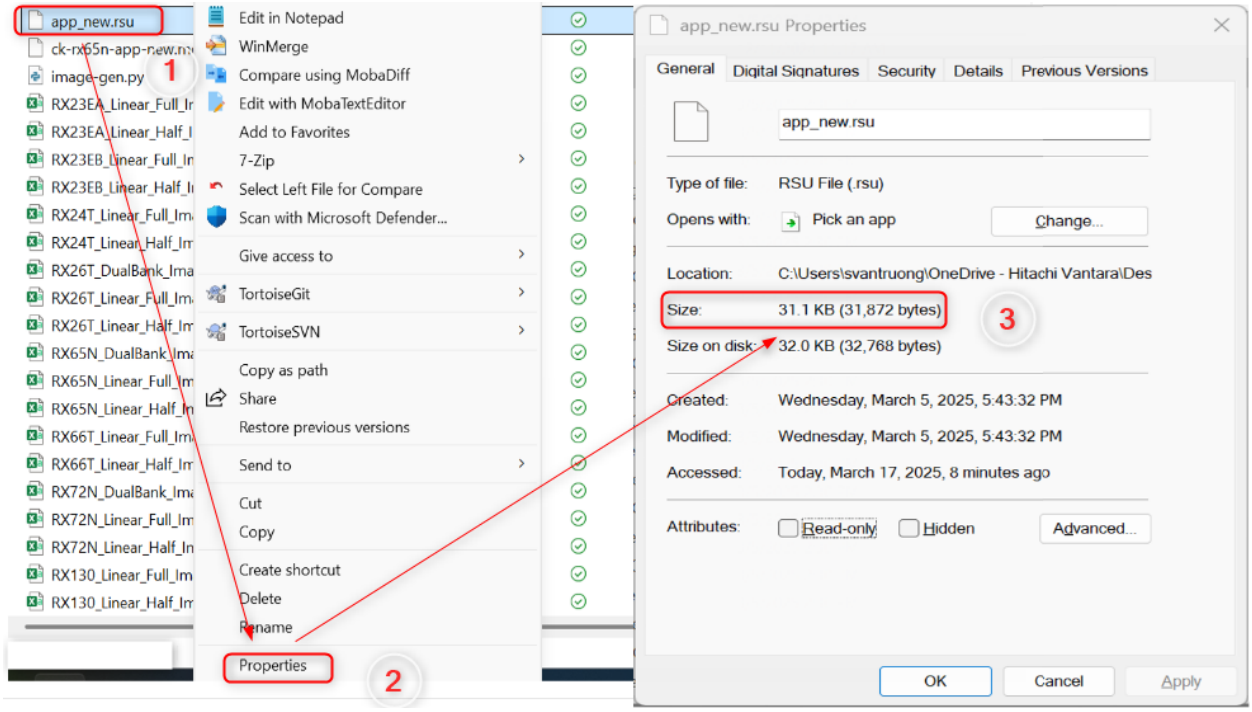


Figure 2.27 Get Size of the Firmware

2.2.7.2 Upload Firmware to Amazon S3 Bucket

2.2.7.2.1 Create Amazon S3 Bucket

(1) Access the AWS web site (Cloud Computing Services - Amazon Web Services (AWS)) and click **Sign In to the Console**.

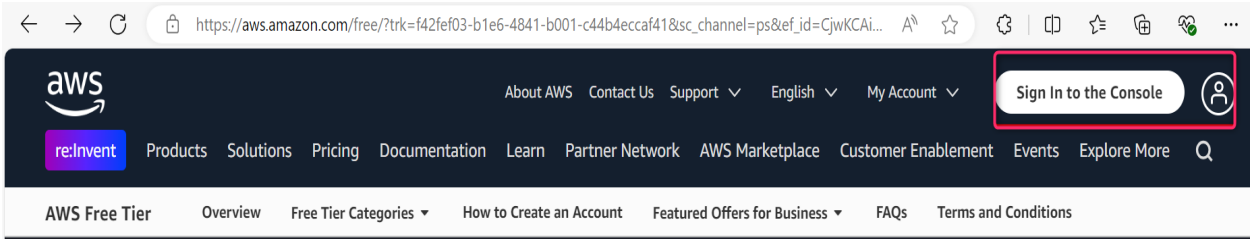


Figure 2.28 Sign-In to Console AWS

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

If you are using the root account to sign in, select the "Sign in using root user email" option and enter the email address of the root account. If you are an IAM user, enter the **Account ID** (12-digit number or account alias), **IAM username**, and **Password** in the corresponding fields.

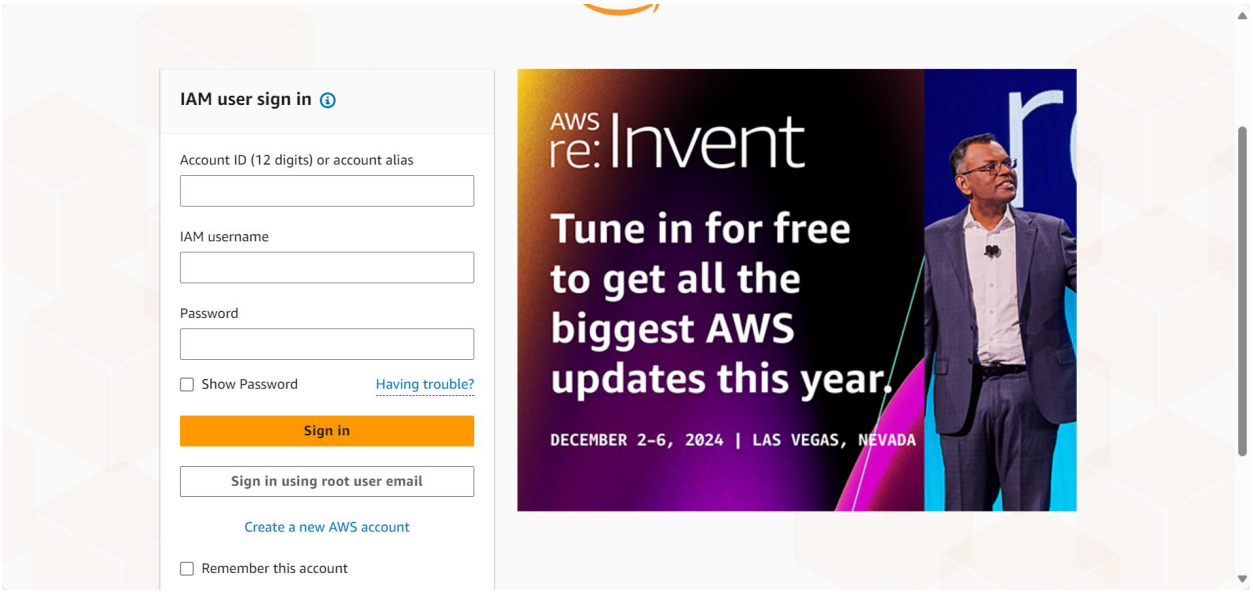


Figure 2.29 User Sign-In

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

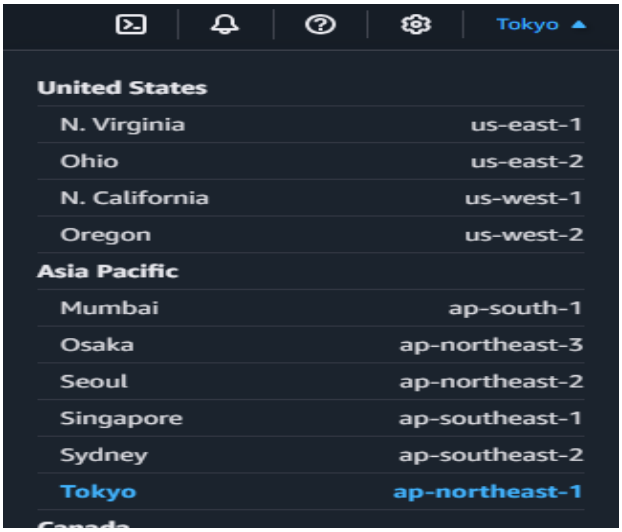


Figure 2.30 Setting Region in AWS

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

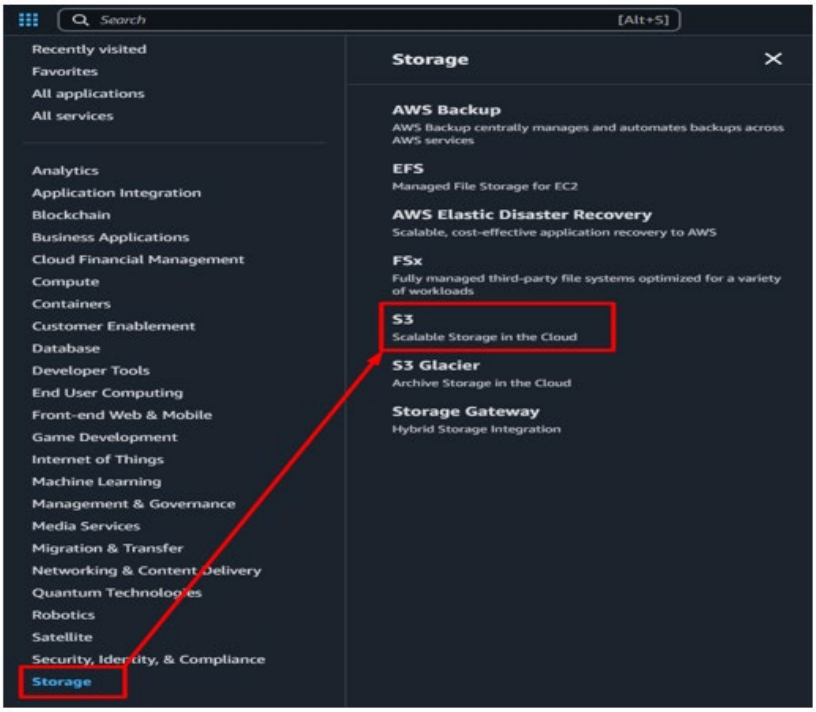


Figure 2.31 S3 AWS Bucket

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

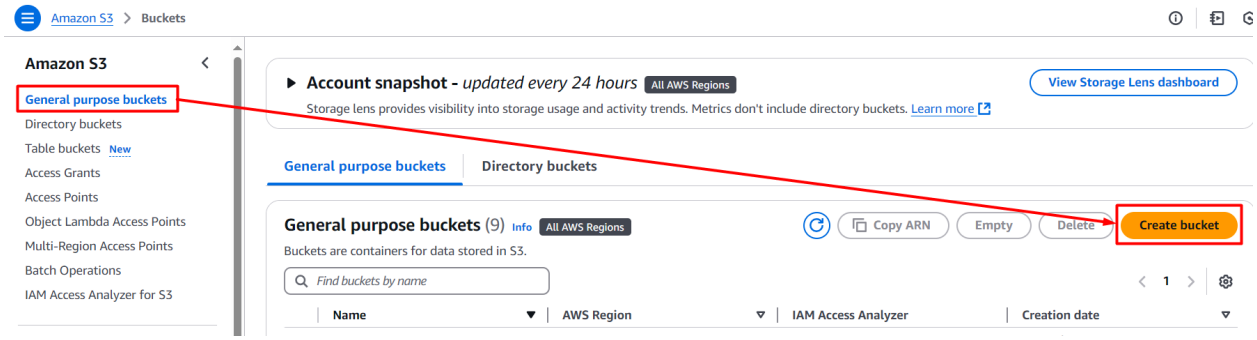


Figure 2.32 Create a Bucket

(6) Enter an S3 Bucket name.

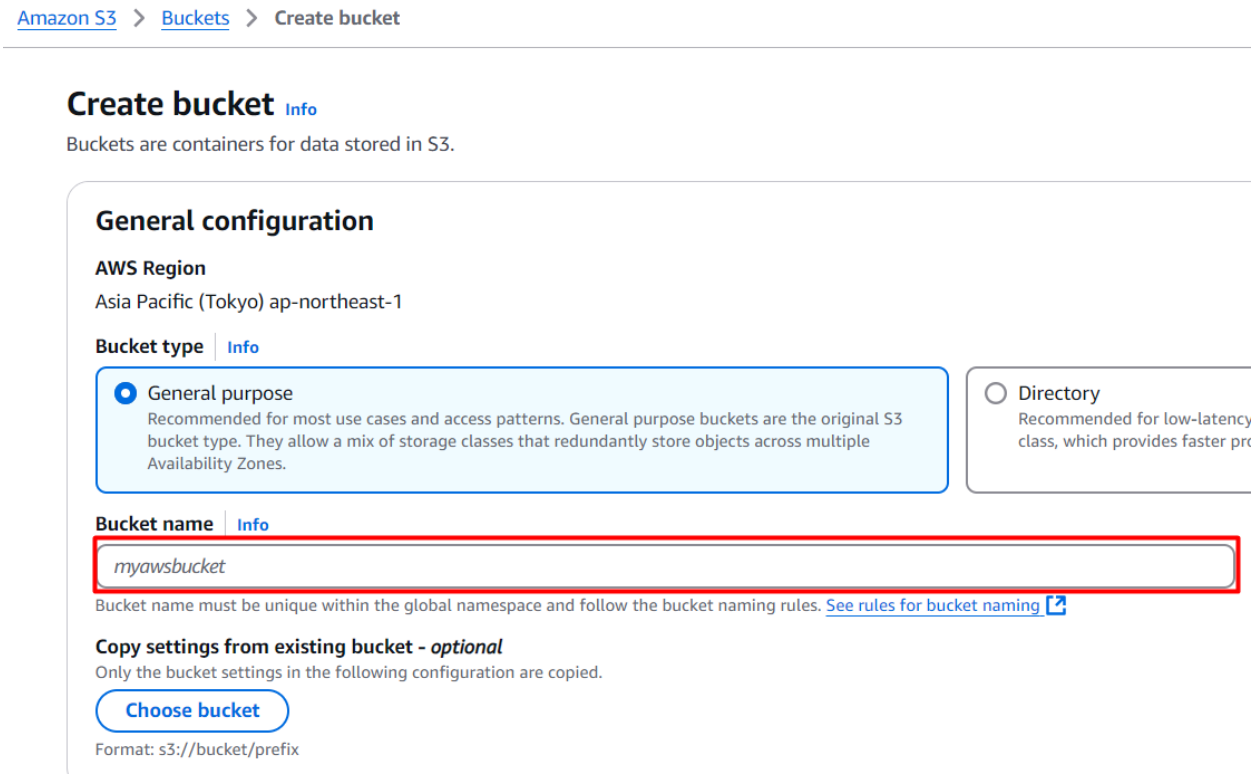


Figure 2.33 Create a Bucket Name

(7) Disable **Block all public access**.

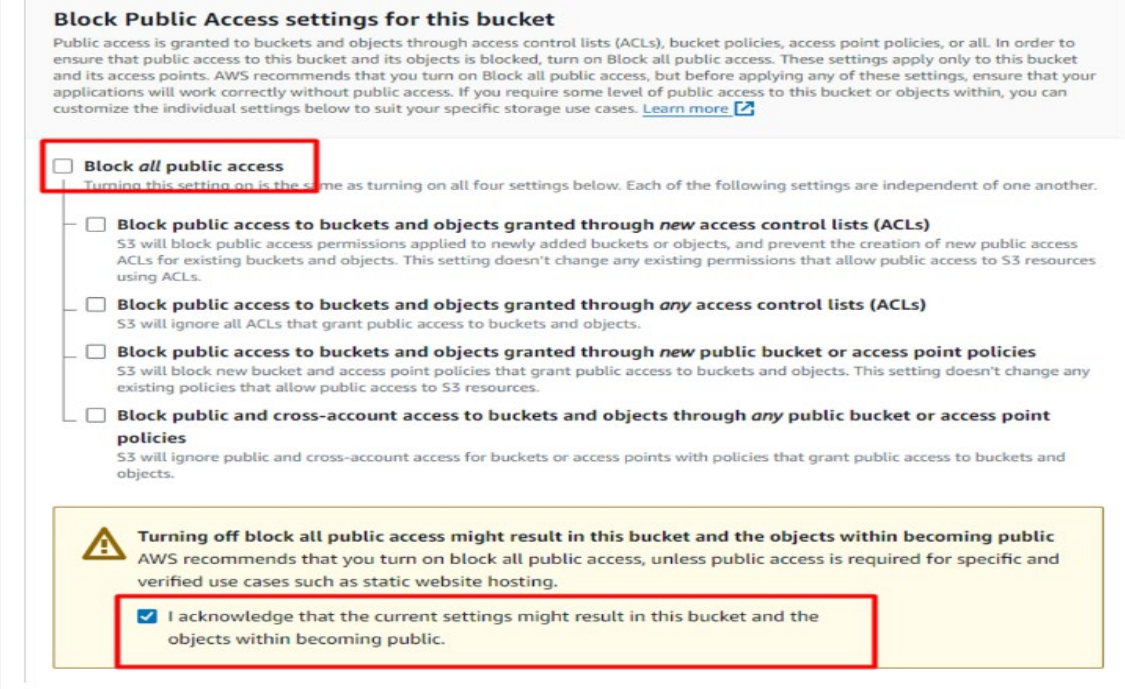


Figure 2.34 Configure Public Access

(8) Create Bucket.

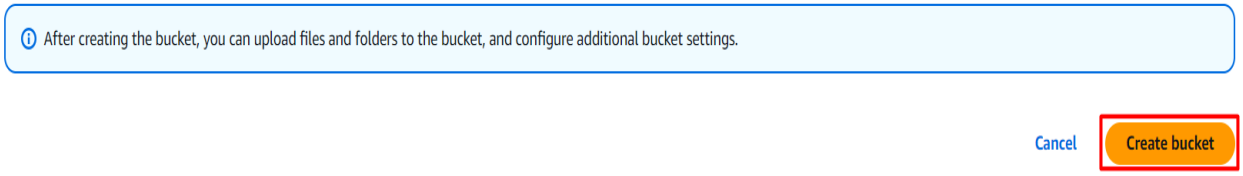


Figure 2.35 Create Bucket

2.2.7.2.2 Upload the Firmware File to the Amazon S3 Bucket

(1) Choose your S3 bucket, and then click **Upload**.

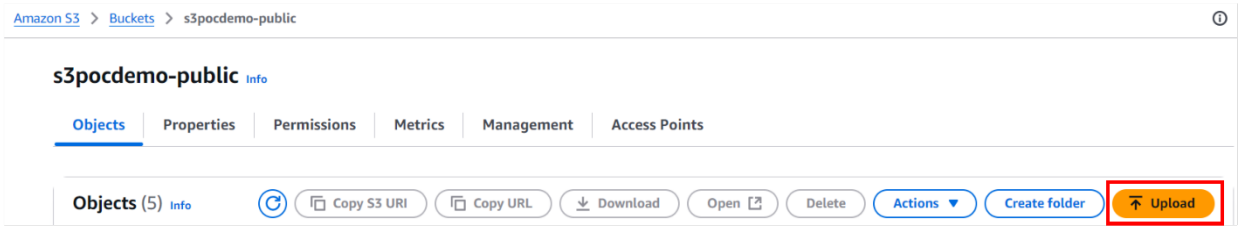


Figure 2.36 Uploading the Firmware File

(2) Click on **Add File**, then select the firmware you created in the previous section. Double-check to ensure it is the correct firmware before clicking **Upload**.

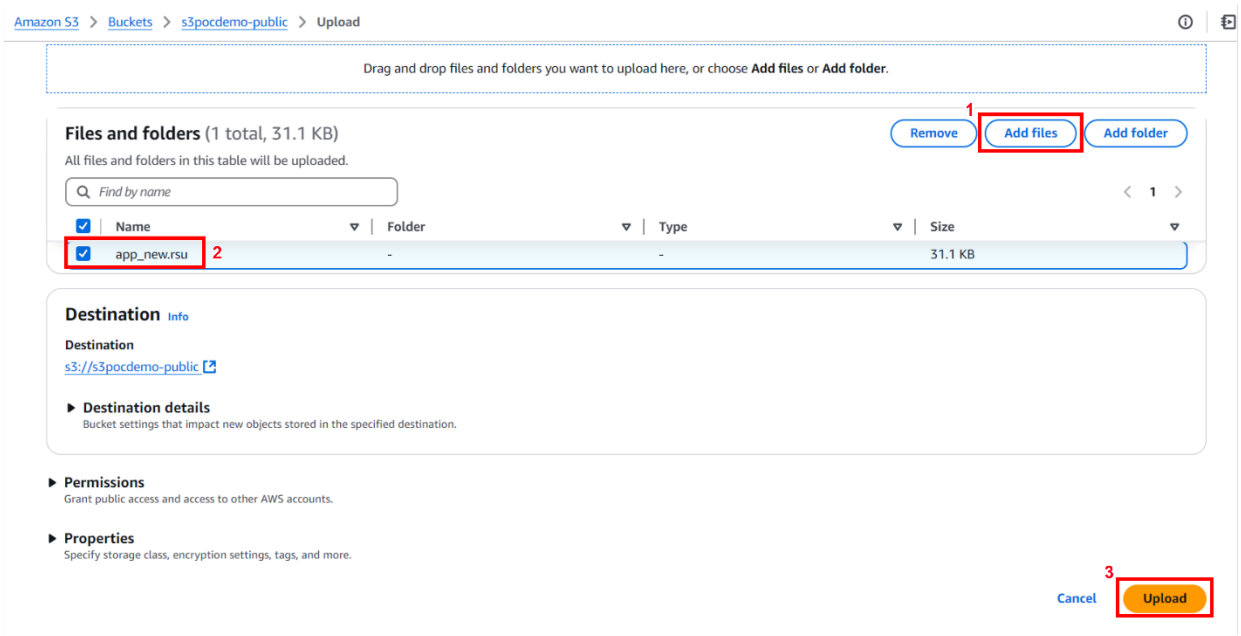


Figure 2.37 Add Firmware File

(3) Update S3 bucket permissions.

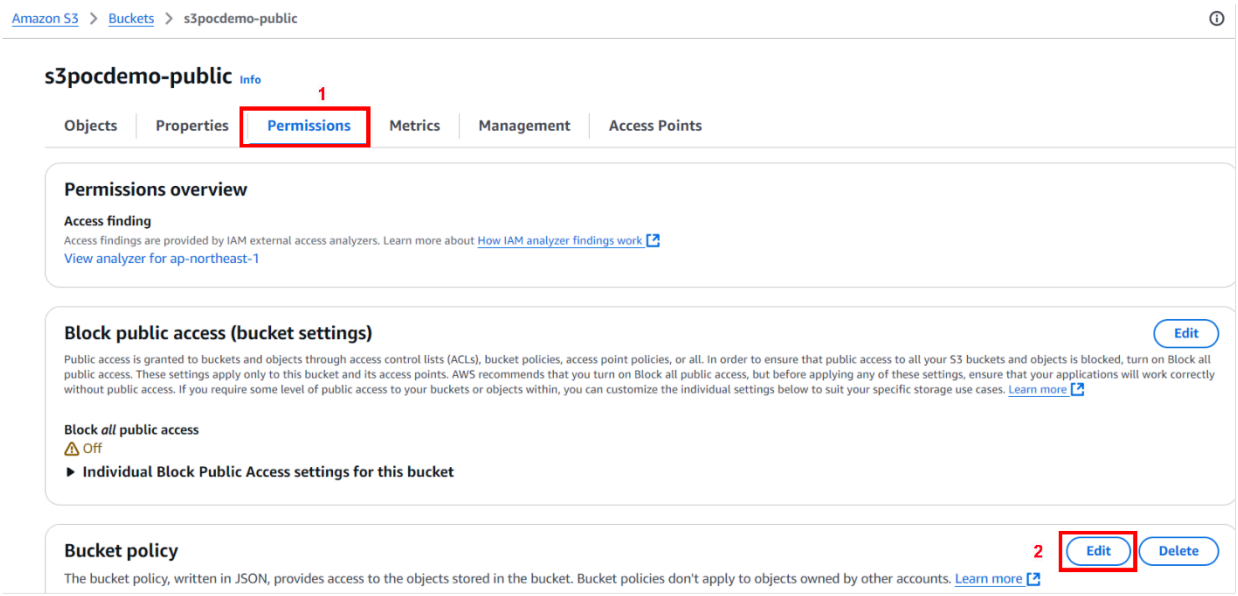


Figure 2.38 Update S3 Bucket Permission

(4) Update the policy for the S3 bucket permissions.

See below figure and ensure you enter your bucket name

Figure 2.39 Update Policy for S3 Bucket Permission

(5) Save the changes.

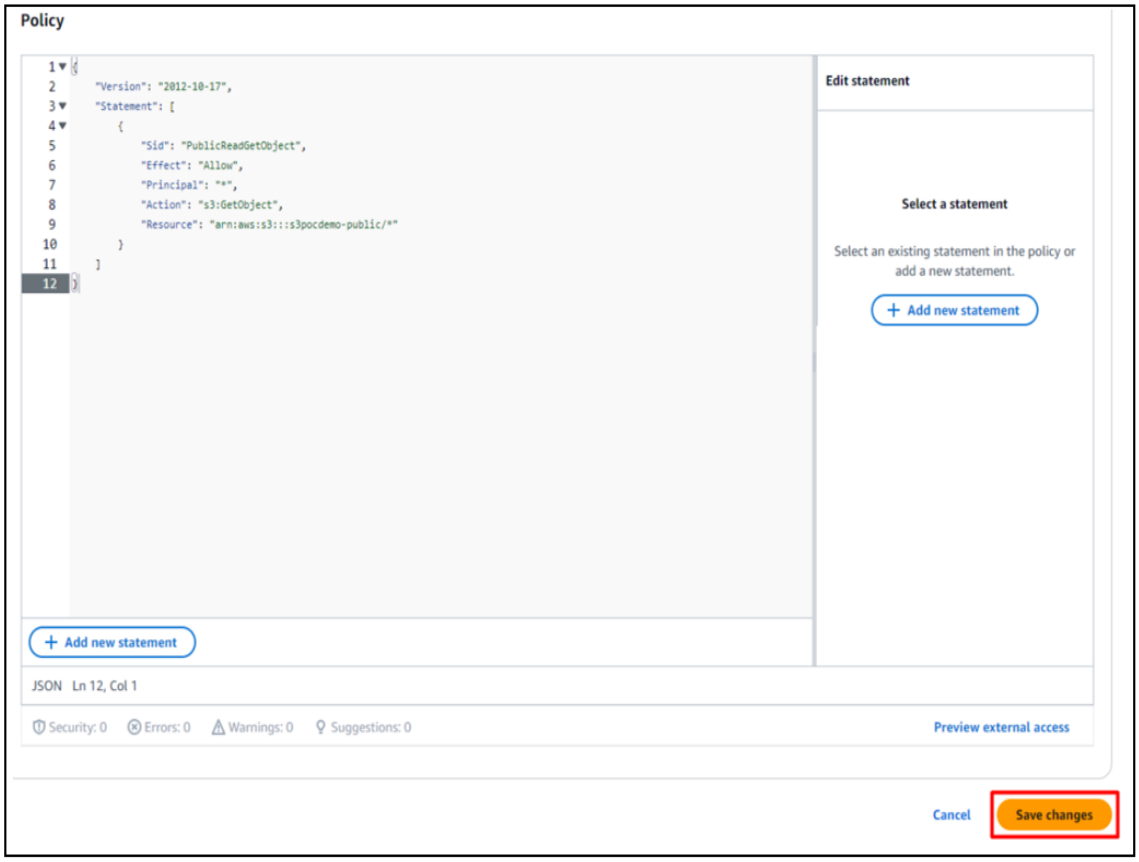


Figure 2.40 Save Policy

(6) Get the Object URL of the firmware you just uploaded.

1. On the **Objects** page of your S3 bucket, locate and select your firmware file (app_new.rsu).
2. Click **Copy URL** to copy the file's download link.

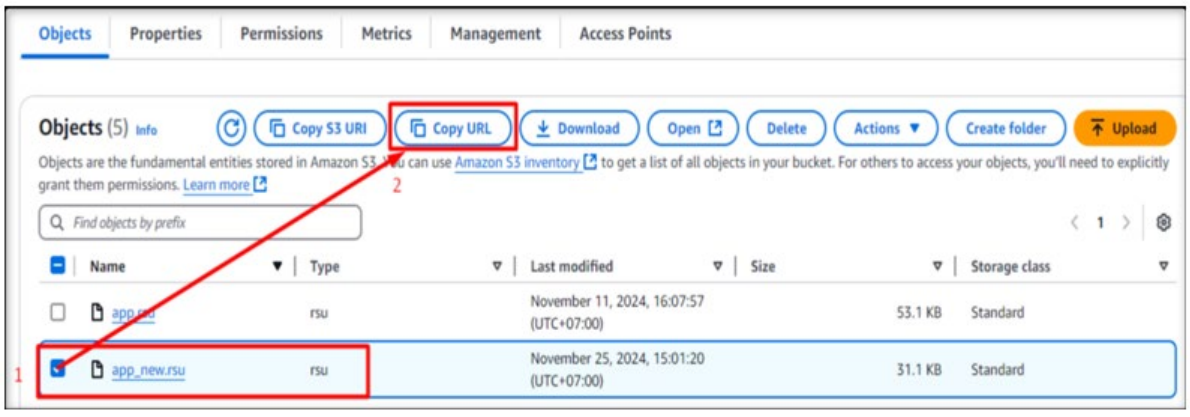


Figure 2.41 Get Firmware URL

Note: The URL obtained from AWS will be used to configure in the **Section 2.2.8.1**, so make sure to save it.

2.2.8 Create Firmware Initialization

2.2.8.1 Building CK-RX65N Application Project

(1) Configure firmware version, Wi-Fi, firmware type, and URL in demo_config.h at

\ota_ck_rx65n_wifi_da_16600_app\demo_project\ck-rx65n-app\src\demo_config

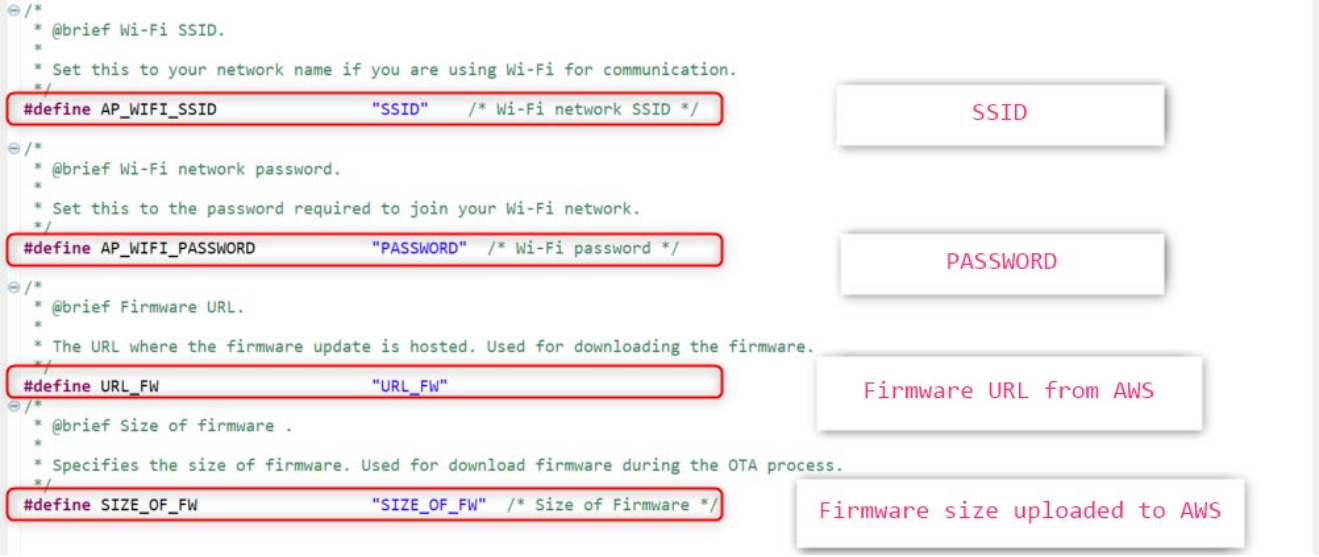


Figure 2.42 Configure Firmware Version, Wi-Fi Network, and Firmware URL

- Check the current firmware version.
- AP_WIFI_SSID: Set the access point name (SSID) of the local Wi-Fi network that the board will connect to.
- AP_WIFI_PASSWORD: Set the password for the local Wi-Fi network
- URL_FW: Firmware URL from AWS S3 bucket. Refer to **Section 2.2.7.2.2**.
- SIZE_OF_FW: The size of the firmware uploaded to the AWS S3 bucket. Refer to **Step (6): Get the Object URL of the firmware you just uploaded** in **Section 2.2.7.2.2**.

(2) Building the project.

2.2.8.2 Building CK-RX65N Bootloader Project

(1) Update public key

Since the Renesas Image Generator is used to create the initial firmware, the public key must be pasted into the bootloader project to match the private key located in the Renesas Image Generator folder. This ensures that the initial firmware can be generated successfully. Follow the instructions below to complete this setup

1. Copy the contents of the **secp256r1.publickey** file you created in S**ection 2.2.4**

2. Paste the public key into **CODE_SIGNENR_PUBLIC_KEY_PEM** in **\ota_ck_rx65n_wifi_da_16600_app\demo_project\ck-rx65n-bootloader\src\key\code_signer_public_key.h.**

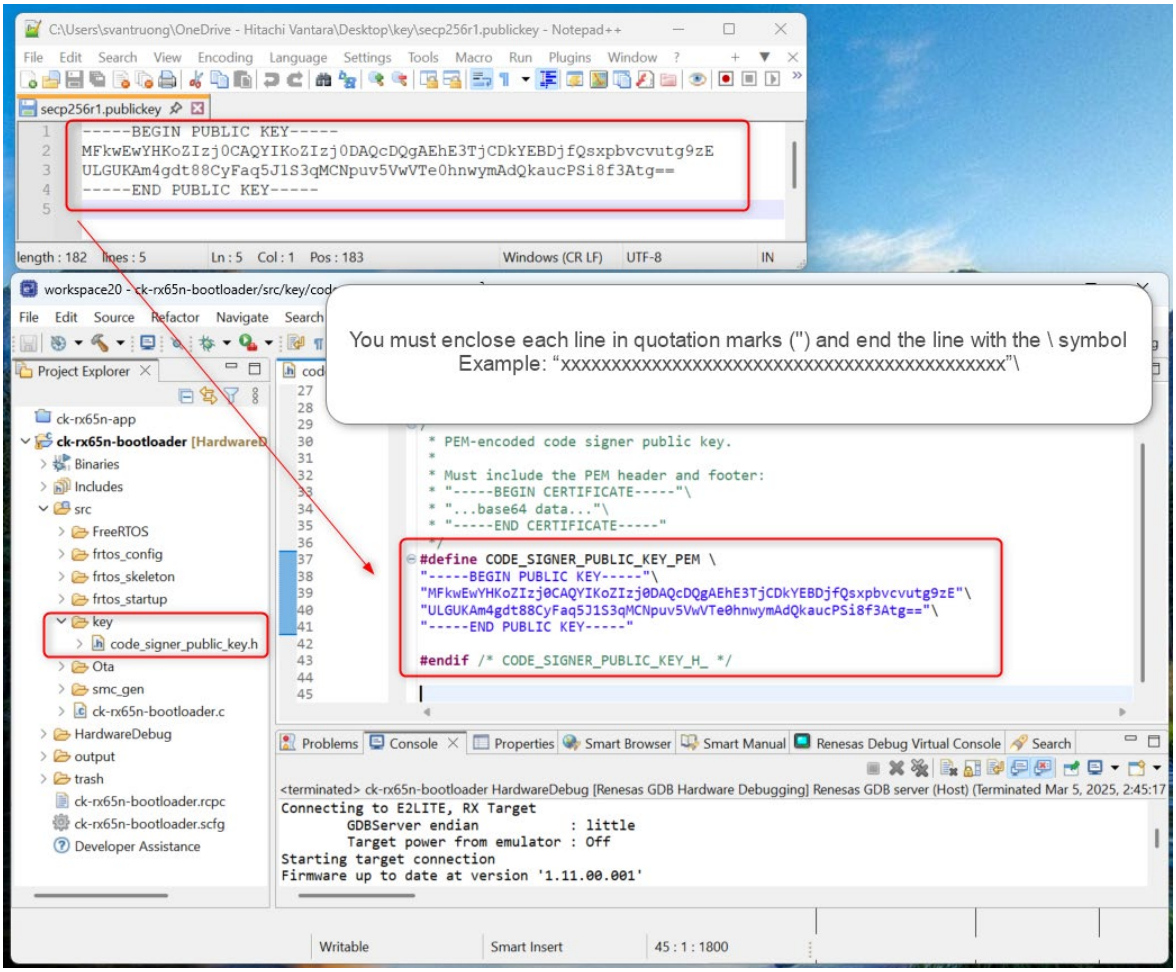


Figure 2.43 Assign a Public Key to Bootloader Project

(2) Building the project.

2.2.8.3 Creating the Initial Firmware

This section is used to create the initial firmware(**userprog.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) Place the following files in the Renesas Image Generator folder:

- The results of the build process in **Section 2.2.8.1**: **ck-rx65n-app.mot**
- The results of building the bootloader in **Section 2.2.8.2**: **ck-rx65n-bootloader.mot**
- The private key created in **Section 2.2.4: secp256r1.privatekey**

(2) Use Renesas Image Generator to generate the initial firmware

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 ck-rx65n-app.mot -ip RX65N_DualBank_ImageGenerator_PRM.csv ^
-o userprog -ibp ck-rx65n-bootloader.mot -key secp256r1.privatekey -vt ecdsa -ff RTOS
```

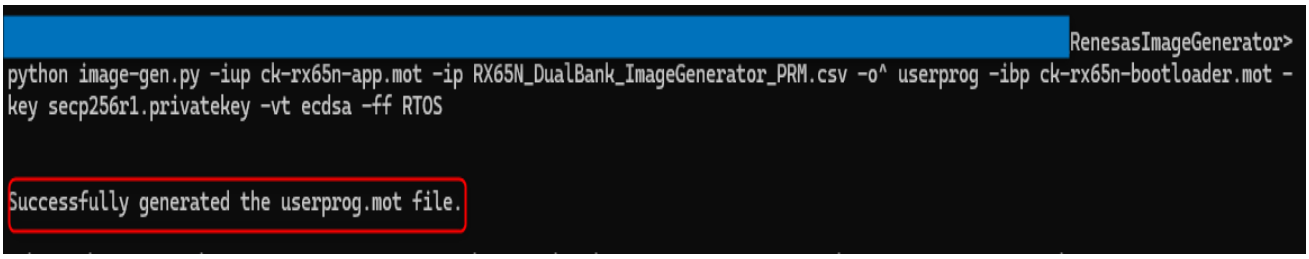


Figure 2.44 Create the Initial Firmware

❖ **Parameter explanation:**

- **-iup**: Input user program (.mot file for application firmware)
- **-ip:** Input parameter file (.csv with image generation settings)
- **-o:** Output file prefix (e.g., userprog.mot)
- **-ibp:** Input bootloader program (.mot file)
- **-key:** Private key file for signing the firmware
- **-vt:** Verification type (e.g., ecdsa for digital signature)
- **-ff:** Firmware format (e.g., RTOS)

3 Execute the Demonstration Project

(1) Start the Renesas Flash Programmer and open the **flash.rpj** project.

This step is used to add the **flash.rpj** file to Renesas Flash Programmer, which is used to flash the initial firmware to the target board.

Follow the steps below to complete the setup.To open the project, select **File > Open Project**.

The **flash.rpj (** project is located in the following folder of the sample program:

\ota_ck_rx65n_wifi_da_16600_app\flash_project.

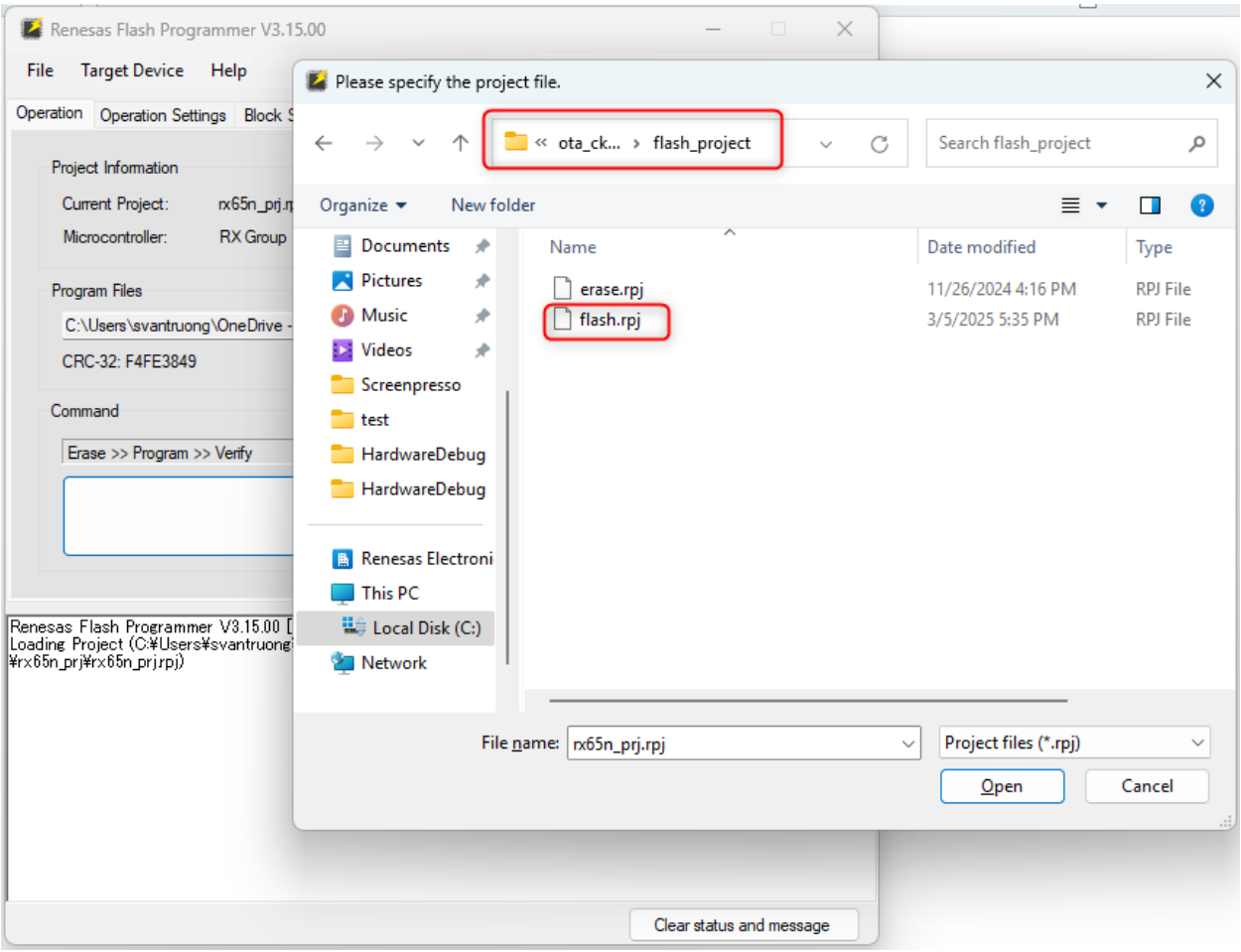


Figure 3.1 Open the flash_project.rpj Project

(2) Select the initial firmware **(userprog.mot)**

Perform this step to flash the initial firmware to the MCU, which **userprog.mot** (created in **Section 2.2.8.3**) is the initial firmware.

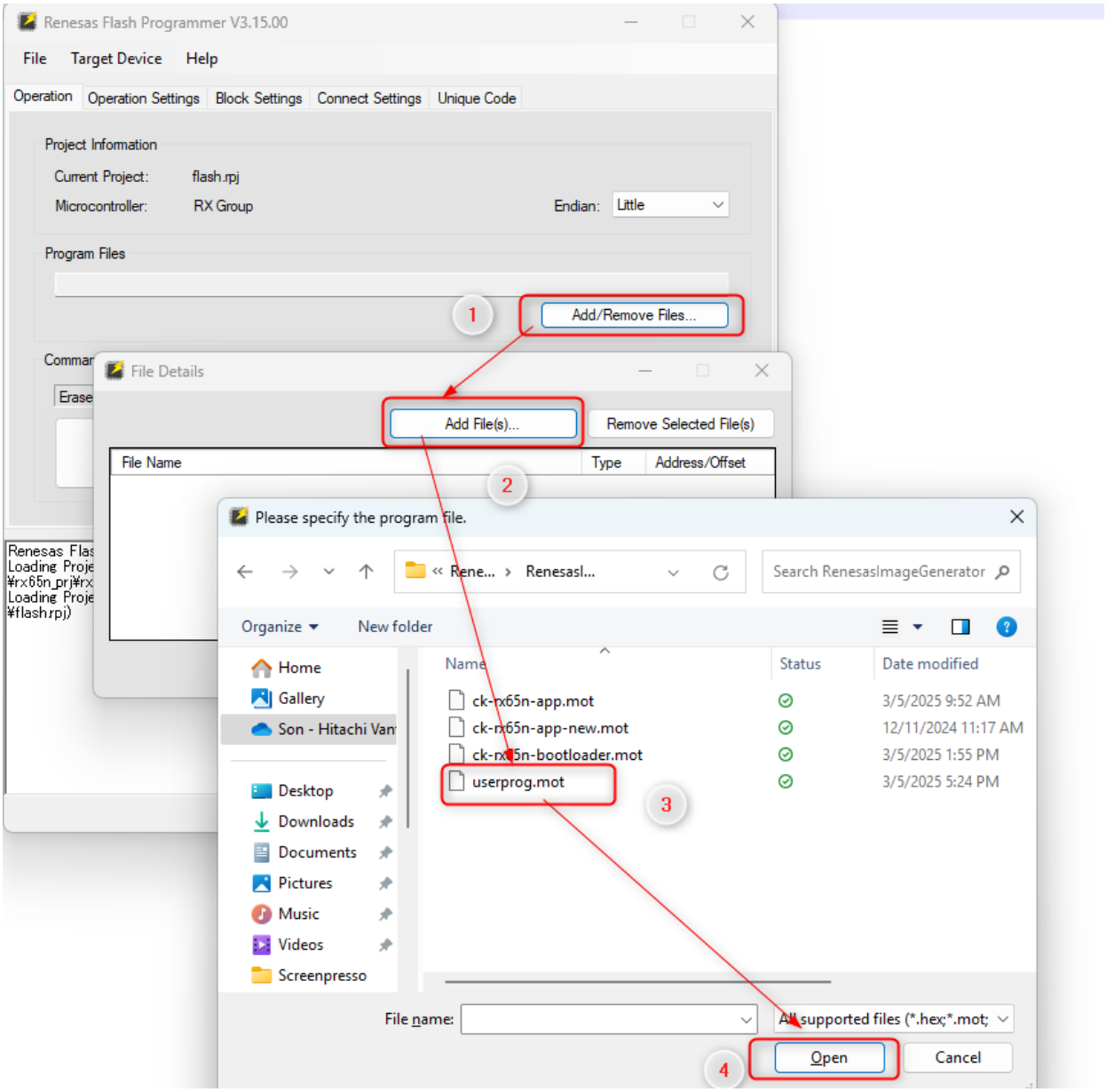


Figure 3.2 Select the Initial Firmware

(3) Click **"Start"** to begin flashing the initial firmware.

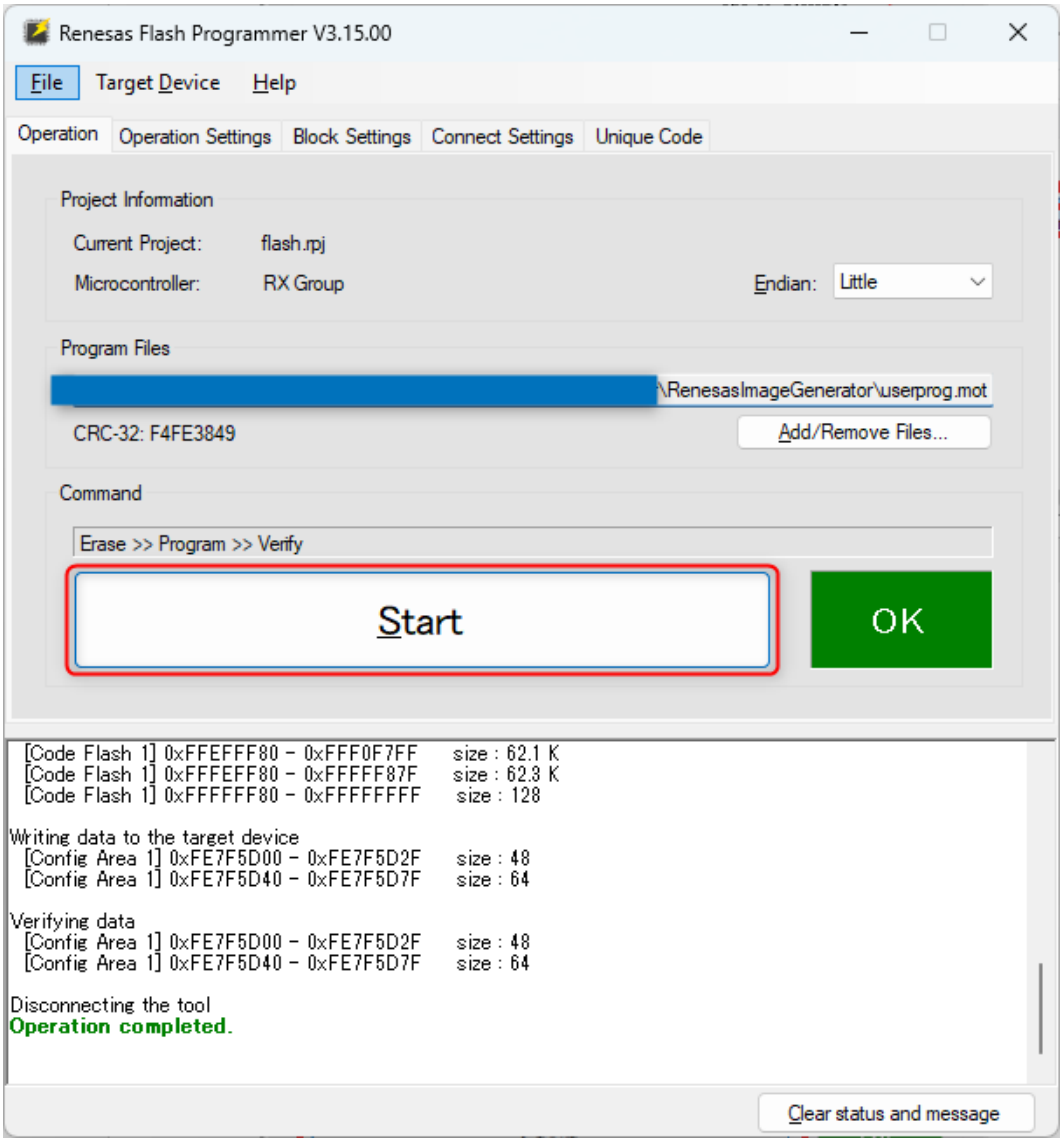


Figure 3.3 Flash Initial Firmware

(4) Check if the Wi-Fi is connected.

When the terminal displays the message highlighted in the image below, it indicates that the Wi-Fi connection has been successfully established. Refer to **Figure 3.4** for more details.

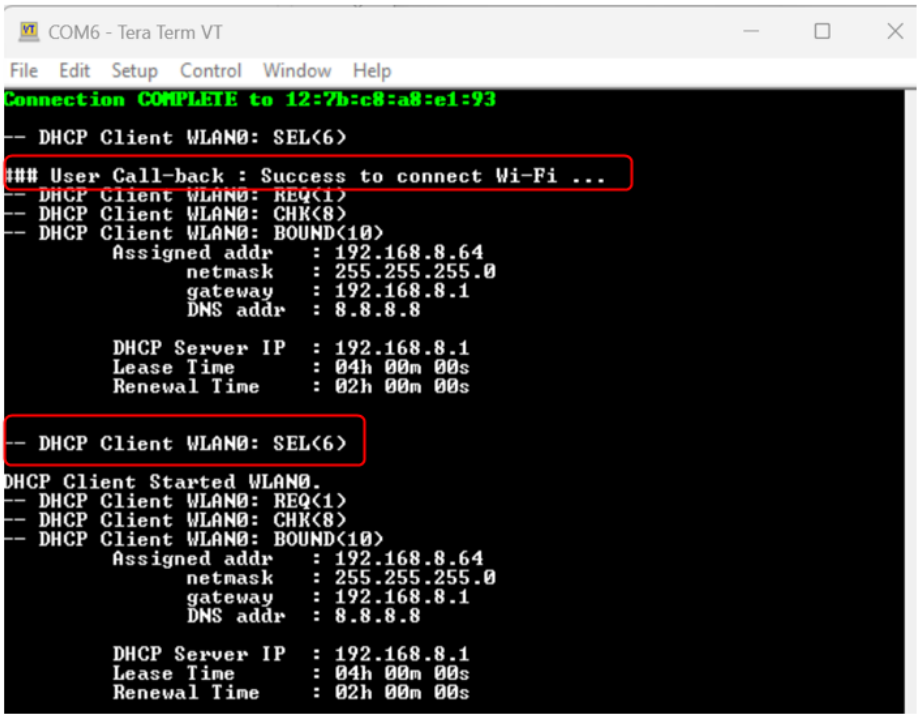


Figure 3.4 Active Wi-Fi Connection

If the message shown in the image below does not appear in the terminal, it means the Wi-Fi connection is inactive or has not been established. Refer to **Figure 3.5** for more details.

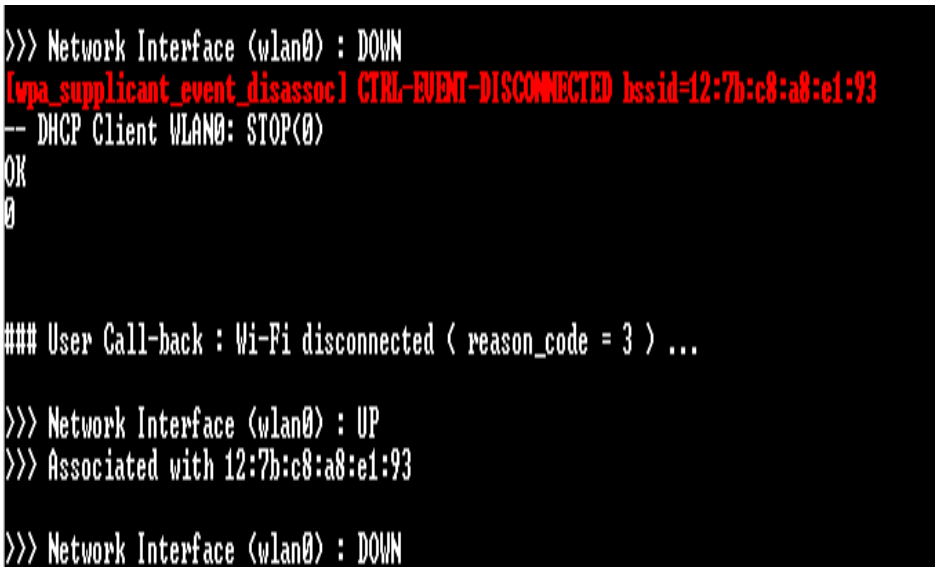


Figure 3.5 Inactive Wi-Fi Connection

(5) Request to update the firmware.

1. Waiting for the network connection to be successfully established (see **Figure 3.6**).
2. Using the PC keyboard, select the terminal window and type the following string**: update**, then press Enter. This sends the string via the USB serial connection of the **CK-RX65N v2** in **Tera Term**.

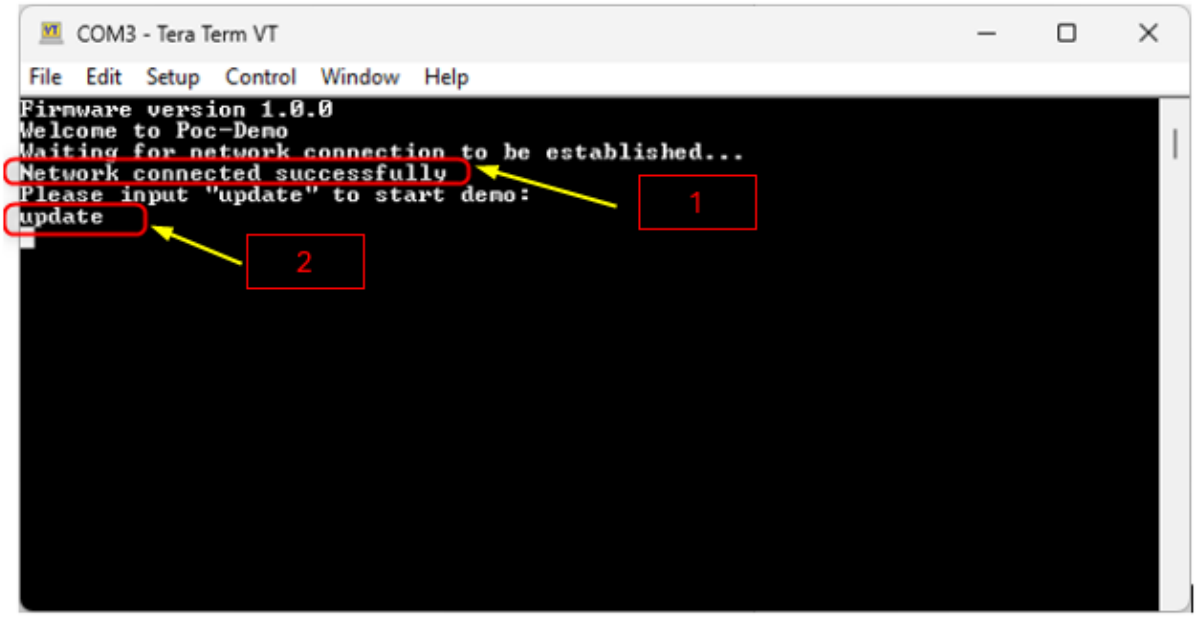


Figure 3.6 Send a Signal to Request a Firmware Update

(6) Observe the Firmware Update Over-the-Air process

The firmware update process starts with progress messages such as 10%, 20%, up to 100% (indicating that percent of flash memory blocks that was written new firmware) displayed in the terminal. **Upon successful completion**, the firmware is updated from **1.0.0** to **1.0.1**. If an error occurs during the update, the MCU automatically resets, and the previous stable firmware version is retained.

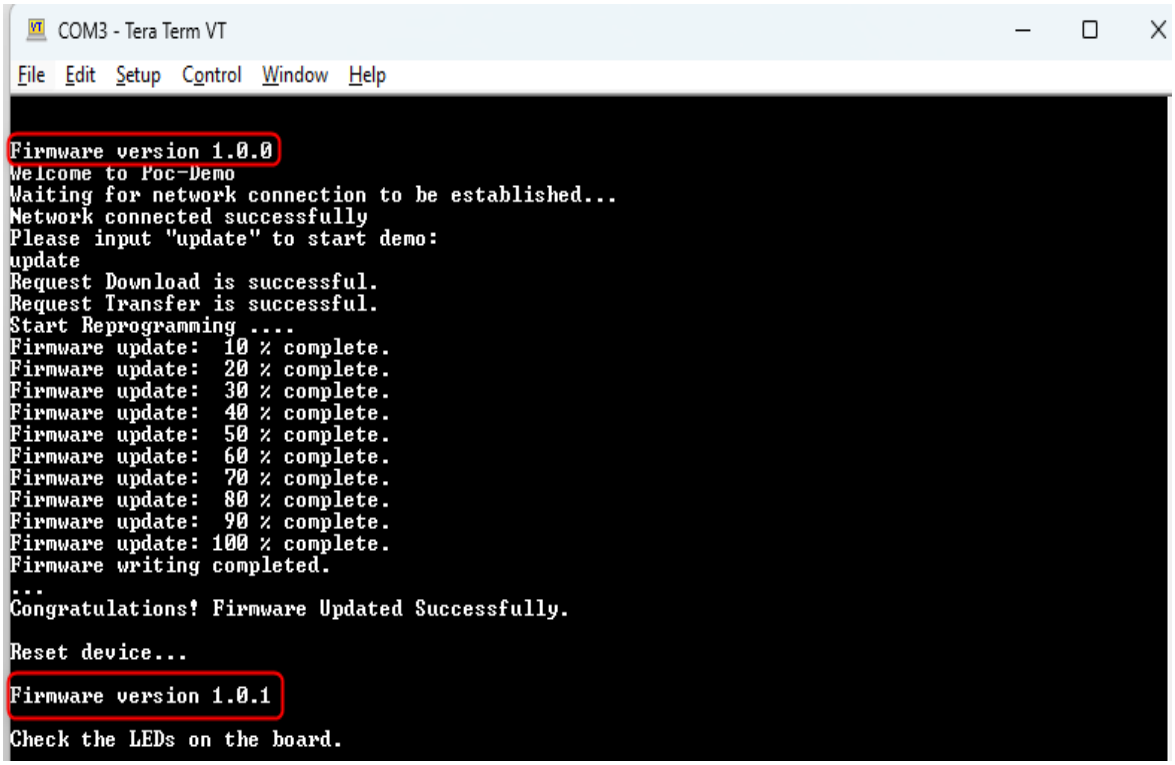


Figure 3.7 OTA Result

Remark: Successfully updated the version from **1.0.0** to **1.0.1.**

4 Appendix

4.1 Known Issues for DA16600

4.1.1 Firmware Size Limit Causes Update Failure

The DA16200/DA16600 SDK V3.2.9.2 restricts firmware downloads from the server to under 256 KB. Attempting to download firmware exceeding this limit results in an error, preventing successful completion. This affects users needing larger firmware updates.

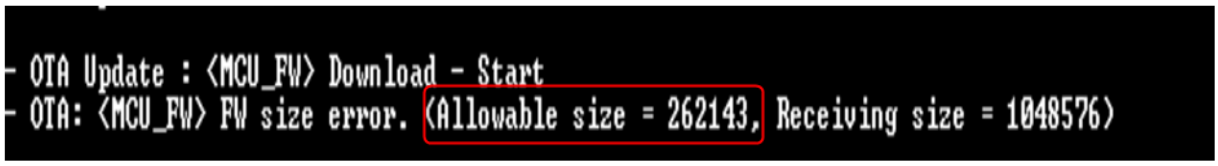


Figure 4.1 Error Log

4.1.2 Resolution

(1) Download **DA16200/DA16600 FreeRTOS SDK V3.2.9.2**

Access the DA16XXX - Ultra-Low Power Wi-Fi SoC for Battery-Powered IoT Devices | Renesas to Download the **DA16200/DA16600 FreeRTOS SDK V3.2.9.2**

(2) Importing **DA16600 FreeRTOS SDK Project** into e2 studio.

Importing the **DA16600 FreeRTOS SDK Project** into e2 studio is similar to importing **the DA16200 FreeRTOS SDK Project**. Therefore, please refer to UM-WI-056 DA16200 DA16600 FreeRTOS Getting Started Guide under **Section 5.4 Importing DA16200 FreeRTOS SDK Project** into e2 studio for detailed instructions on the process.

(3) Modify the project.

- In e2 studio Project Explorer, open the file **config_generic_sdk.h** in the folder **da16600\get_started\include\user_main** and modify the macro highlighted as below.

```
#if defined ( __SUPPORT_OTA__ )
#define __OTA_UPDATE_MCU_FW__
#endif //__SUPPORT_OTA__
```

- In e2 studio Project Explorer, open the file **da16200_map.h** in the folder **da16600\core\bsp\driver\include\DA16200** and modify the two macros highlighted as below.

```
/* DA14531 BLE Firmware Download start */
#define SFLASH_BLE_FW_BASE            (SFLASH_14531_BLE_AREA_START)

/* DA14531 BLE Security DB Area start */
#define SFLASH_USER_AREA_BLE_SECURITY_DB (SFLASH_BLE_FW_BASE + __BLE_IMG_SIZE__)

/* SFLASH User Area */
#define SFLASH_USER_AREA_1_START      0x00600000
#define SFLASH_USER_AREA_1_END        0x00800000
```

(4) Building project.

Please refer to UM-WI-056 DA16200 DA16600 FreeRTOS Getting Started Guide under **Section 5.5 Building Projects** for detailed instructions on the process.

(5) Flash the new firmware.

Flash the new firmware via Tera Term by following the instructions provided in the UM-WI-056 DA16200 DA16600 FreeRTOS Getting Started Guide under **Section 4.5.2 Using Macro Script of Tera Term**.

4.2 Debugging

OTA_LOG in the project is used to provide additional error information during the debugging process.

Revision History

Rev.	Date	Revision History	
		Page	Summary
1.00	Apr.22.2025	-	First edition issued