
ISO-DONGLE1Z Rev.B

Communications Dongle

This manual provides an overview of the ISO-DONGLE1Z Rev.B communications dongle bootloader and application software and a description of the bootloader structure, operation, and programming. The manual also contains a step-by-step guide for the application software post-processing and update in the device memory without using any external hardware tools.

Contents

1. Communications Dongle Overview	2
1.1 Assumptions and Advisory Notes	2
2. Bootloader	2
2.1 Structure and Operation	2
2.2 Bootloader Programming	5
3. Software Image Update	8
4. Generation of Software Image	10
5. Revision History	17

1. Communications Dongle Overview

The ISO-DONGLE1Z Rev.B isolated dongle serves primarily as a communications dongle between a Battery Front End (BFE) evaluation board and a Graphical User Interface (GUI) available on a workstation ([Figure 1](#)). It supports USB and I2C communications and is compatible with multiple BFEs and GUIs. The dongle uses a powerful Renesas Advanced Family RA4 ARM-based microcontroller.



Figure 1. GUI, Communications Dongle and Battery Front End Connection

1.1 Assumptions and Advisory Notes

- It is assumed that users possess a basic understanding of microcontrollers, embedded systems hardware, battery management systems, and secondary battery cells.
- It is assumed users have prior experience working with Integrated Development Environments (IDEs) such as e2studio, Flexible Software Package (FSP), and terminal emulation programs such as Tera Term.
- Renesas recommends reviewing the ISO-DONGLE1Z Rev.B Communications Dongle Hardware Manual before changing or developing any software for the dongle.
- Renesas recommends reviewing the Renesas RA4 E1 Group 32-Bit MCU Datasheet, available on the [product page](#), to become familiar with the MCU in use.

For more information about the bootloader operation, refer to the Renesas RA Family Secure Bootloader for RA2 MCU Series Application Note.

2. Bootloader

2.1 Structure and Operation

The bootloader of the ISO-DONGLE1Z Rev.B Isolated Communications Dongle enables upgrading or modifying the MCU firmware without the need for specialized programming hardware or tools. It features the MCUboot secure bootloader of the Renesas Flexible Software Package (FSP) to receive a binary file over the USB interface, verify its content integrity, and writes the program memory with the application program. The bootloader uses the whole code flash memory of the MCU. [Figure 2](#) shows the flash memory map. It is separated into three major parts: Bootloader, Primary and Secondary Application Image Slots. The bootloader occupies 64-KB of the code flash memory or the first eight blocks (8-KB size), containing the MCUboot module, USB driver, HP Flash driver, and GPIO driver for the LED indication. Each image slot is 96-KB long or three blocks (32-KB size), which is the maximum memory space available for the application software.

[Figure 3](#) shows the operation flowchart of the bootloader. When the CPU is released from reset, a low-level initialization is performed. All three user LEDs are turned simultaneously ON for 500ms and then OFF to perform an LED test and indicate the beginning of the bootloader code execution. Next, the digital input corresponding to pin #2 of the Expansion header 1 (J3) is tested for low level. If this condition is true, the USB interface is initialized to create a virtual COM port and activates the software image download process. The MCU waits for a connection with terminal emulator software and receives a start of download command, Carriage return (Enter). The bootloader starts to store every Byte received by the virtual serial connection in the Secondary Image Slot by portions of 32-KB (1 block) and if the transferred image size is different than 96-KB or any unexpected behavior is encountered, the operation is terminated and the LED D4 is illuminated.

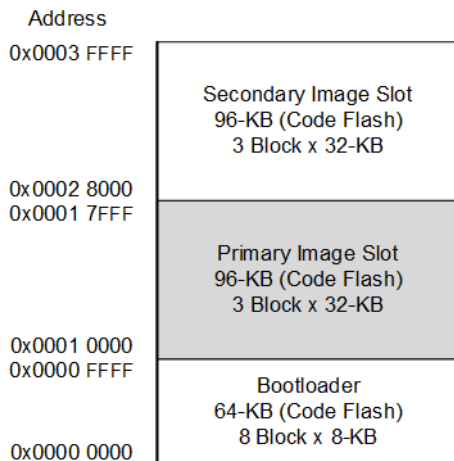


Figure 2. Bootloader Flash Memory Map

When the image download has been successfully completed, LED D3 is illuminated and the MCU halts operation and waits for manual restart. This is also the moment when the level of the digital input (J3-2) must be changed to high (Remove jumper wire). Table 1 shows the various states indicated by the user LEDs and Table 2 shows the virtual COM port settings.

Important: Do NOT set a higher baud rate as this can lead to buffer overrun and compromise the image transfer! After successful bootloader start-up and LED test if the level of the digital input corresponding to pin #2 of the Expansion header 1(J3) is high (default), MCU proceeds to software image boot up. The bootloader checks the Secondary Image Slot memory space. If an image is available there, it is validated by checking integrity and authenticity. After successful authentication, the bootloader copies the new image using the overwrite update method where the entire content of the Primary Image Slot is overwritten with the contents of the Secondary Image Slot. The active image is always executed from the Primary Image Slot. This method is fail-safe, resistant to power-cut failures and with less memory overhead compared to the other available update methods. Conversely, it does not support image pre-testing and an automatic fallback mechanism. When the Primary Image Slot memory space is successfully overwritten, the bootloader executes the new image. If no new image is detected in the Secondary Image Slot, the bootloader directly authenticates and boots up the Primary Software Image.

Table 1. Bootloader LED States

Condition	LED States	
	D3 (Red)	D4 (Red)
LEDs blink test on start-up.	ON	ON
Proceeding to normal software image boot.	OFF	OFF
Software image download mode is entered.	Blinking slow	OFF
Waiting to receive the software image by the virtual serial connection.	Blinking fast	OFF
Receiving software image in process.	Blinking fast	Blinking fast (antiphase with D3)
Software image download is completed.	ON	OFF
An error has occurred during image download.	OFF	ON

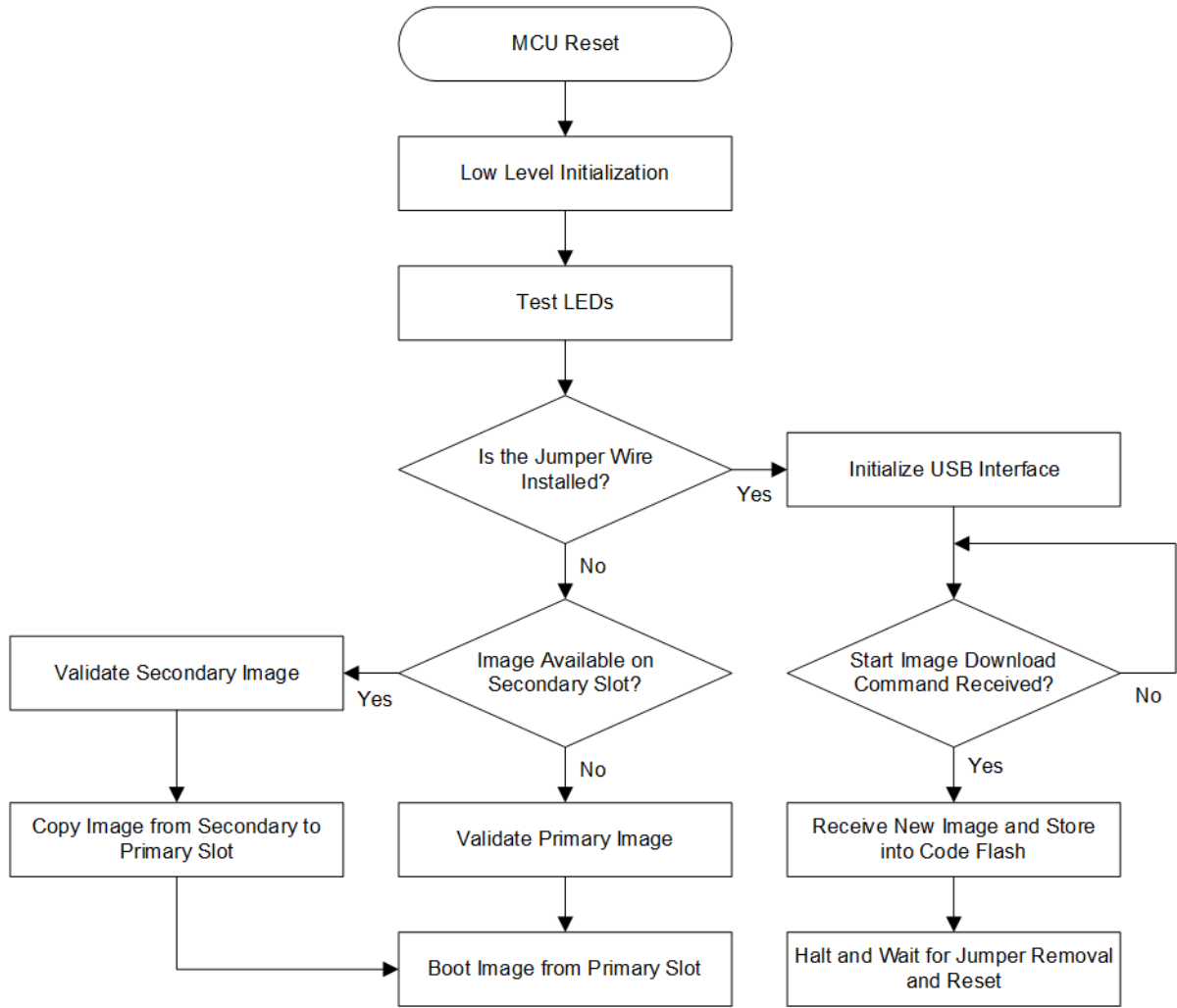


Figure 3. Bootloader Flowchart Diagram

Table 2. Serial Connection (Terminal) Settings

Parameter	Value
New Line (Receive)	CR
New Line (Transmit)	CR
Terminal ID	VT100
Baud Rate	9600
Data Bits	8 bits
Parity	none
Stop Bits	1 bit
Flow Control	none

2.2 Bootloader Programming

The ISO-DONGLE1Z Rev.B Isolated Communications Dongle is programmed by default with the bootloader software. However, if using external debugging tools during custom software development, it is erased or overwritten in the code flash memory. This section describes how to restore the bootloader in the MCU memory. Renesas E2 Emulator/ E2 Emulator Lite with 10-pin Cortex® Debug Connector and the following software available on a workstation running Windows® 10 are needed.

- Renesas Flash Programmer V3.11.00 or later.
- **ra4e1_iso_dongle_mcuboot_v_1_0.srec** binary file v1.0 or later

Complete the following steps for the bootloader programming process:

1. On the dongle make sure that jumper JP1 is installed.
2. Connect Renesas E2 Emulator/ E2 Emulator Lite to the JTAG connector J4 of the dongle board (Figure 4).

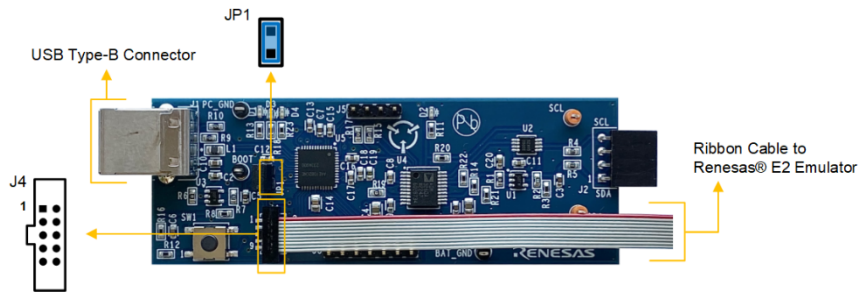


Figure 4. Communications Dongle with Attached Renesas E2 Emulator

3. Connect the USB cable between the PC and the USB Type-B connector of the board. Ensure the power ON indicator LED D1 is illuminated.
4. Run Renesas Flash Programmer and on the top menu select **File > New Project...** From the **Microcontroller:** drop-down menu select **RA** (Figure 5). In the **Project Name:** field enter RA4E1_Dongle. From the **Tool** drop-down menu, select **E2 emulator** or **E2 emulator Lite** and click on the **Connect** button.

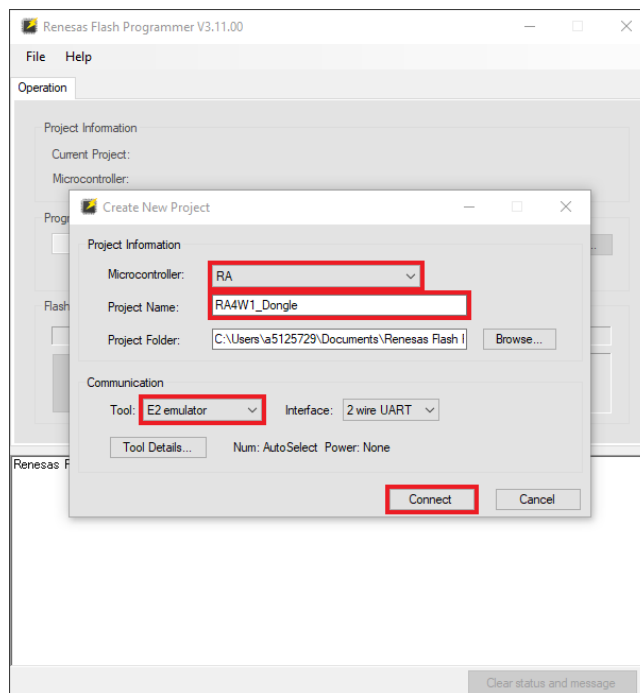


Figure 5. Creating a New Project in Renesas Flash Programmer

- In the main window select the **Operation Settings** tab and ensure in the **Command** section **Erase**, **Program** and **Verify** are selected (Figure 6).

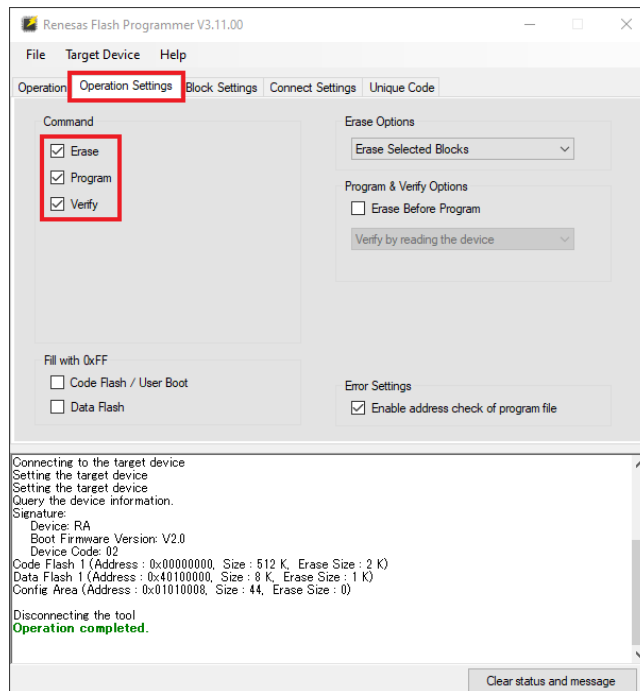


Figure 6. Operation Settings in Renesas Flash Programmer

- Select the **Connect Settings** tab and click on the **Tool Details...** button (Figure 7). In the pop-up window make sure that option **None** is selected in the **Power Supply** section. Click on the **OK** button to close the pop-up window.

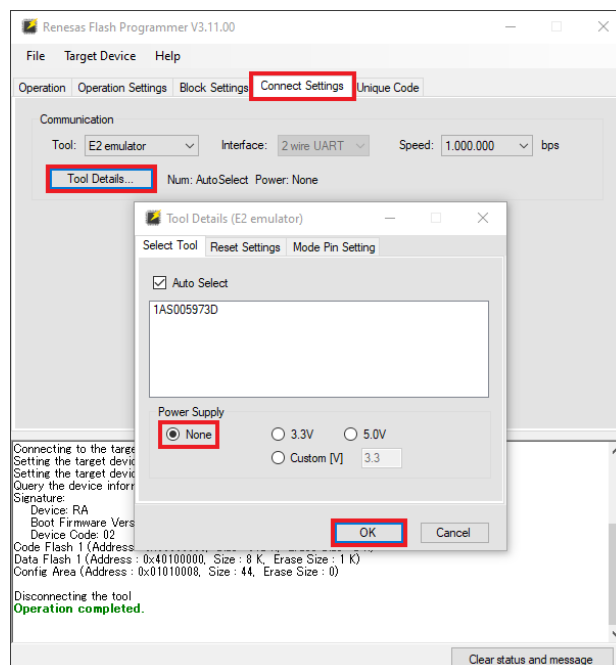


Figure 7. Connection Settings in Renesas Flash Programmer

7. Select the **Operation** tab and click on the **Browse...** button (Figure 8). In File Explorer navigate to the location of the bootloader binary file (.srec), select it and then click on the **Open** button. Click on the **Start** button to download the file into the code flash memory.

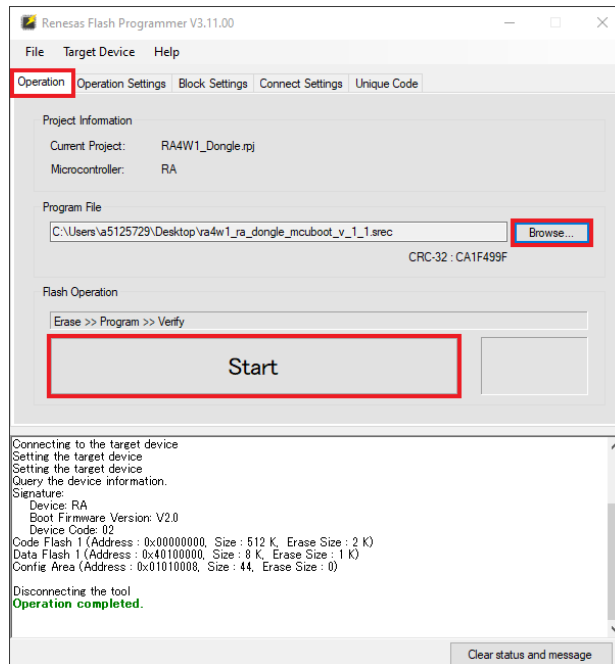


Figure 8. Selection of the Program File

8. When the process completes successfully you should see a green OK message in a rectangular window next to the **Start** button (Figure 9). Disconnect the USB cable. Disconnect the Renesas E2 Emulator/ E2 Emulator Lite from the dongle and close the Renesas Flash programmer. The bootloader programming process is completed.

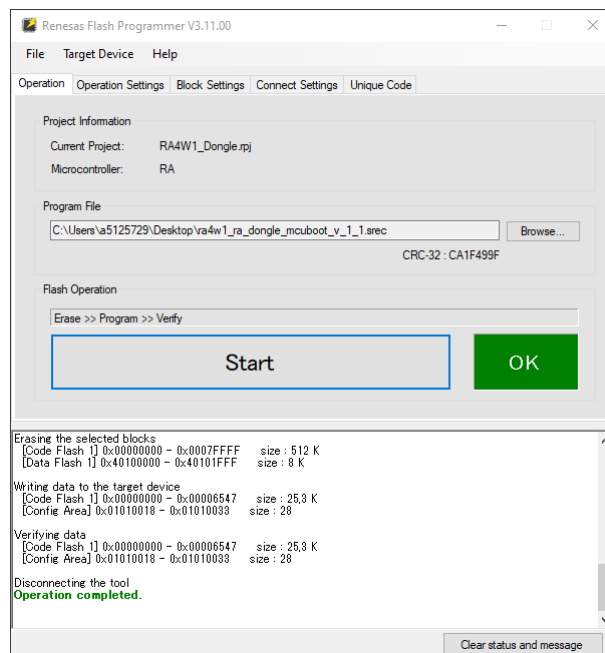


Figure 9. Successful Programming Window

3. Software Image Update

The software image (referred to as image) in the communication dongle can be updated or modified without using any additional external software programmers. To initiate the software image update process, first ensure the required image is available on the PC and the terminal emulator program Tera Term is installed. Tera Term can be downloaded from the [Tera Term Home Page](#).

The image update process must follow these steps:

1. Connect the BOOT Test Point to the PC_GND Test Point using a jumper wire as demonstrated on [Figure 10](#).
2. Connect the USB cable between the PC and the USB Type-B connector of the board. Ensure LED D3 is blinking slowly, indicating that the bootloader is in Image Update Mode.

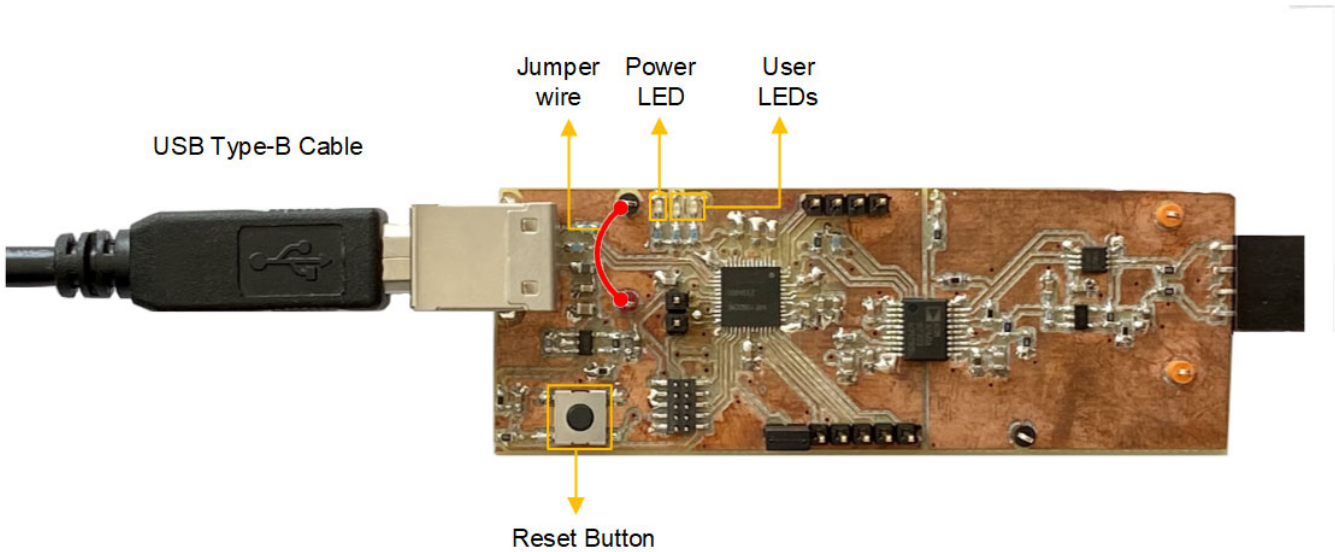


Figure 10. Connection of the ISO-DONGLE1Z Rev.B Isolated Communications Dongle for Software Image Update

3. Open Tera Term. A new connection should automatically open ([Figure 11](#)). If not, select **File > New Connection**. From the window, select **Serial** and ensure the **USB Serial Device COM** port is selected from the drop-down list. Click the **OK** button to open the connection.

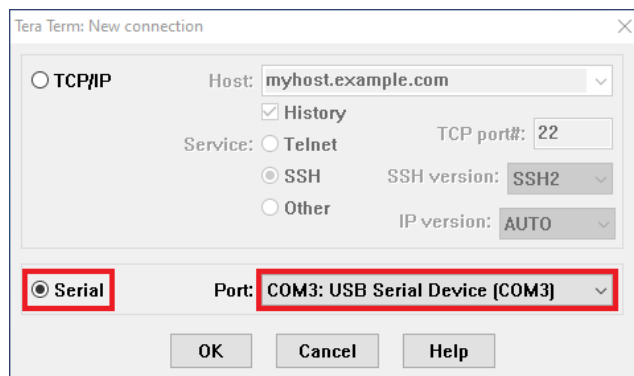


Figure 11. Tera Term New Connection Window

- Press **Enter** to activate Image Download Mode. LED D3 starts blinking fast and **READY!** message is displayed on the console screen (Figure 12). The bootloader is now waiting to receive the image file by the USB serial interface. Each subsequent byte is interpreted as a part of the Image. Be careful not to press any other buttons or the process will be compromised and must be restarted.

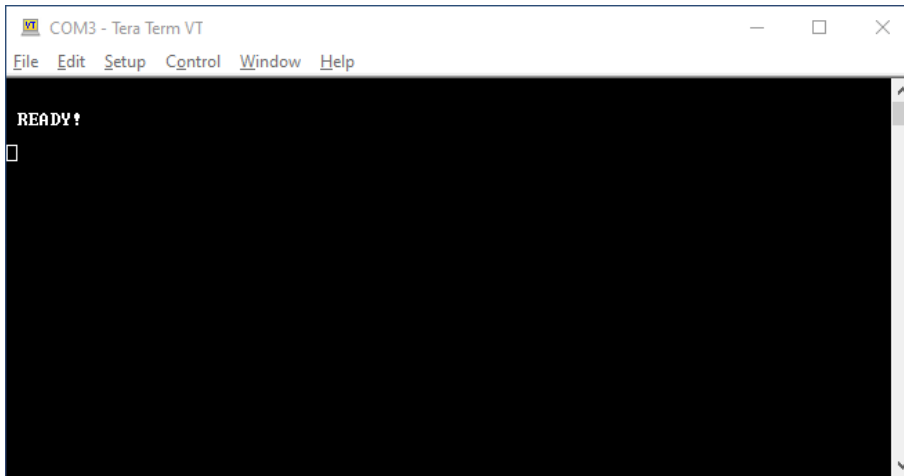


Figure 12. Console Screen – Ready for Software Image Download

- Select **File > Send file...** from the menu. On the opened window (Figure 13) select **Binary**, navigate to the location of the software image file, select it, and click on the **Open** button. Now the terminal emulator program sends the whole file. LEDs D3 and D4 blink in antiphase and a pop-up window displays the transfer progress, which takes approximately 10 seconds.

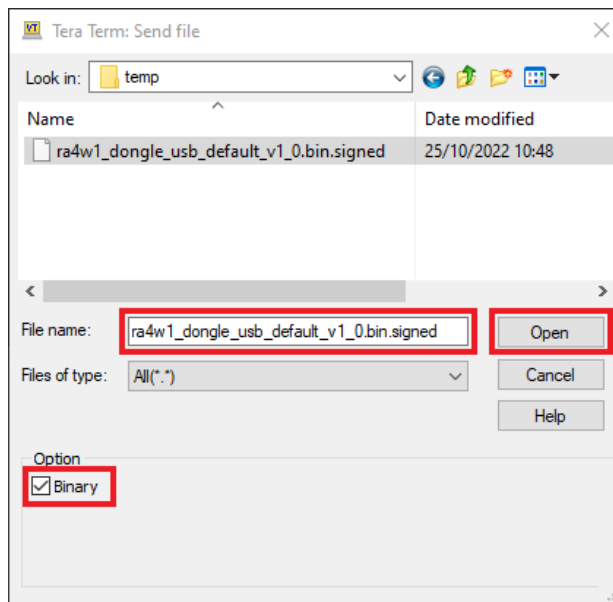


Figure 13. Selection of Software Image

- When the software image download has finished D3 remains illuminated and **DOWNLOAD COMPLETE!** message is printed in the console screen (Figure 14). You must unplug the USB cable and remove the jumper wire. Next time the MCU powers up, it copies the new software image, and boots from it. This completes the software image update process. If the red LED D4 is illuminated, the bootloader becomes unresponsive or the image transfer sticks, there is something wrong with the image update process. Unplug the USB cable and restart the process from Step 1.

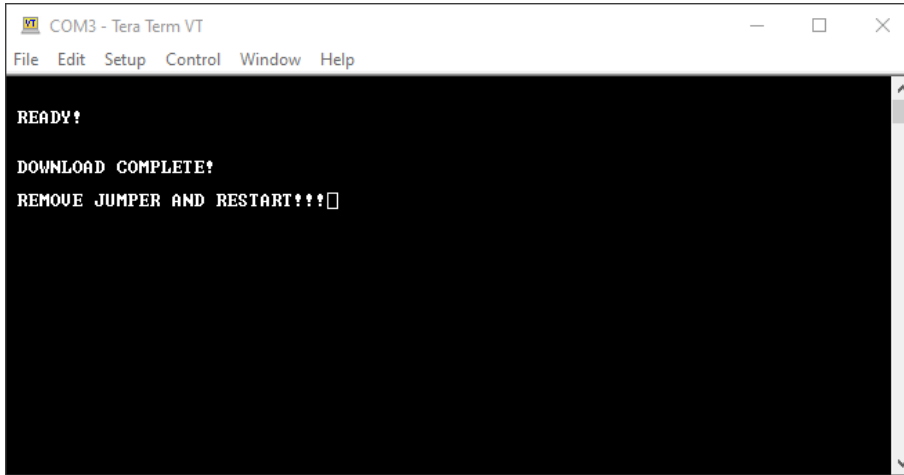


Figure 14. Console Screen – Software image download has finished

4. Generation of Software Image

Developing and running a customized software (application) on the dongle using the bootloader to avoid the need for any debugger tools requires the generation of a software image that meets the bootloader size and verification requirements. This section describes the process of generating a software image from a conventional project in e² studio. Before beginning, the following development tools and software available on a workstation running Windows® 10 are needed:

- e² studio Development Environment (IDE) 2022-07 or greater with GCC Arm® Embedded tool chain
- Renesas Flexible Software Package (FSP) 4.0.0 or later
- Python 3.11.0 or later
- **ra4e1_iso_dongle_mcuboot_v_1_0.rar** bootloader software project v1.1 or later

Important: Ensure that Python is installed to all users, and it is added to Windows PATH (**Control Panel > System > Advanced System Settings > Advanced > Environment Variables**). Otherwise, running python scripts through e² studio IDE will not work.

If this procedure has already been done and the MCU boot software project and python environment are already set up, go directly to Step 10.

The image generation process must follow these steps:

1. Import the MCUboot software project that contains the python script and is used in the image generation process. *Note:* This is the bootloader project **not** the application software that runs from the target image. Ensure the **ra4e1_iso_dongle_mcuboot_v_1_0.rar** project archive file is already available. Right-click in the e² studio Project Explorer view and select **Import** from the menu showing up to open the Import Wizard (Figure 15).

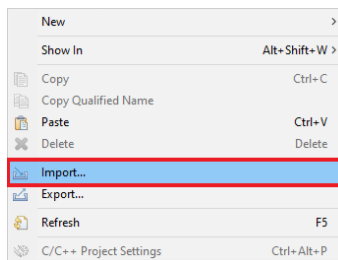


Figure 15. Right-click Menu to Import the Bootloader Project

- From the Import Wizard window, select **Existing Project into Workspace** and then click on the **Next >** button (Figure 16).

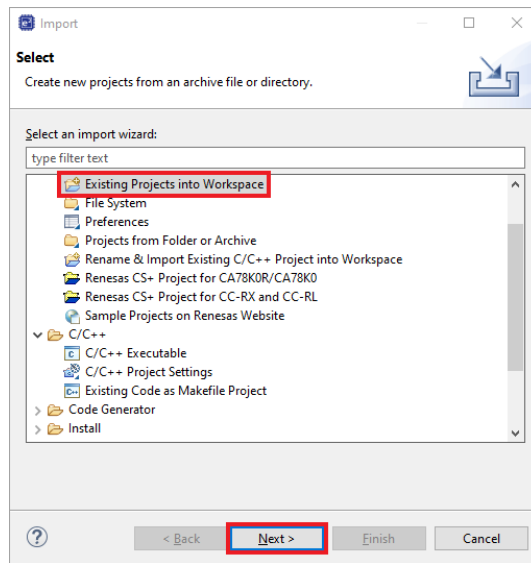


Figure 16. Selection of the Import Wizard Option

- On the next window, select **Select archive file** and then click on the **Browse...** button (Figure 17). In File Explorer, navigate to the location of the bootloader project archive file (.zip), select it and then click on the **Open** button. Click on the **Finish** button to complete the import process and close the Import Wizard.

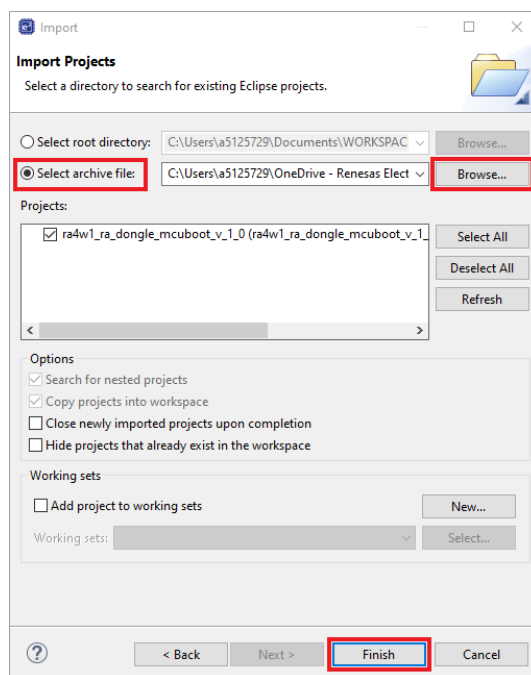


Figure 17. Selection of the Bootloader Project

- The project is now imported into the e² studio workspace. Double-click on **configuration.xml** to open the FSP Configurator and click on the **Generate Project Content** button to generate the hardware specific project files that are needed to run the project on the selected hardware (Figure 18).

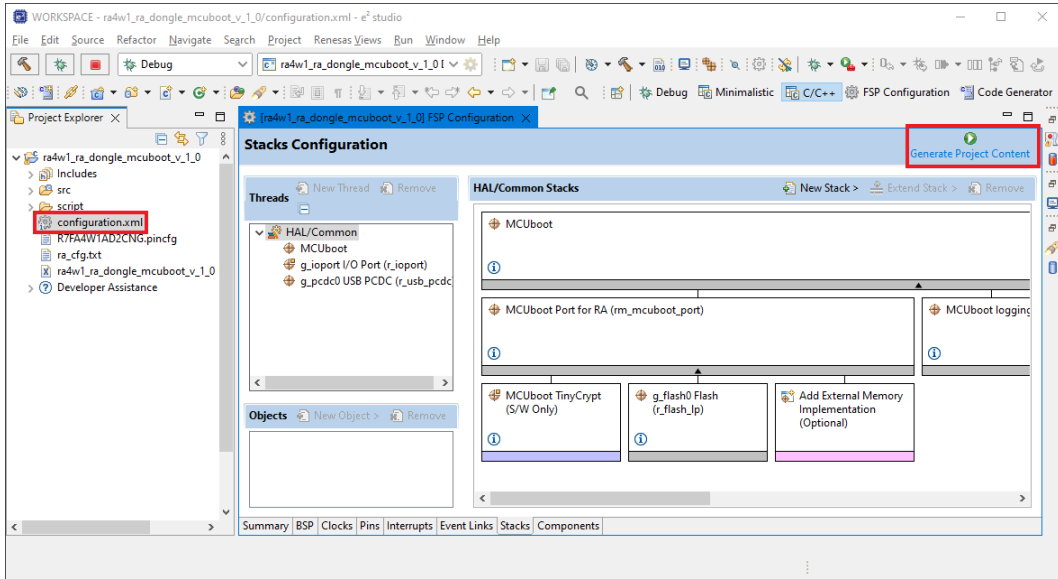


Figure 18. Generation of Hardware Specific Project Files

- Configure the Python Signing Environment. Signing the application image is done using a post-build step in e² studio using the image signing tool **imgtool.py**, which is included with MCUboot. In the e² studio Project Explorer navigate to **ra4e1_iso_dongle_mcuboot_v_1_0\ra\mcu-tools\MCUboot** folder (Figure 19). Right-click over the **MCUboot** folder and select **Command Prompt**. This opens a command window with the path set to the **\mcu-tools\MCUboot** folder.

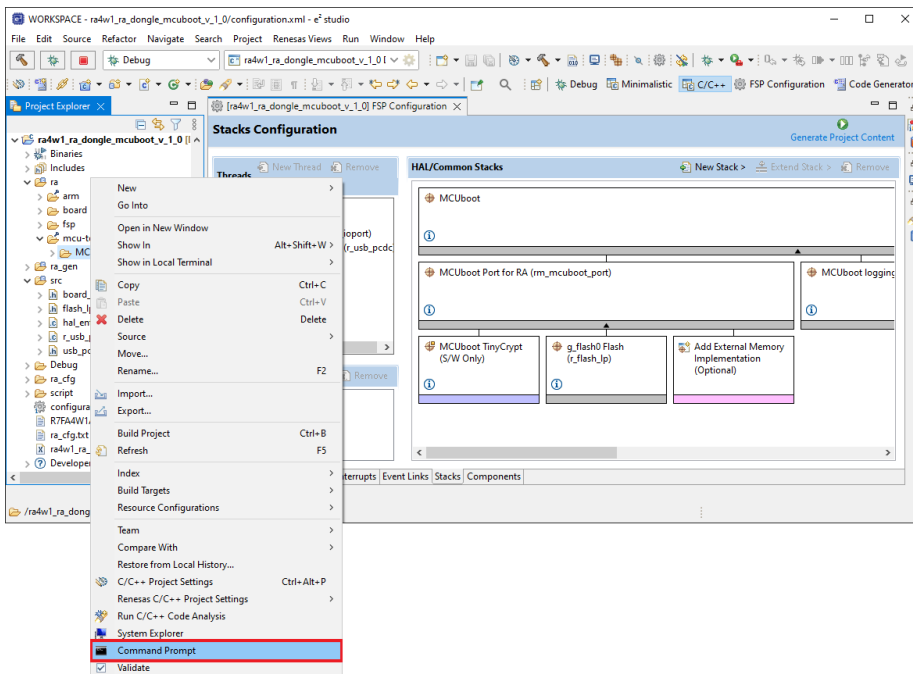


Figure 19. Opening Command Prompt in the MCUboot folder

6. Enter the following command to update pip:

```
python -m pip install --upgrade pip
```

7. Next, in the command window, enter the following command line to verify and install all the MCUboot dependencies (Figure 20):

```
pip3 install --user -r scripts/requirements.txt
```

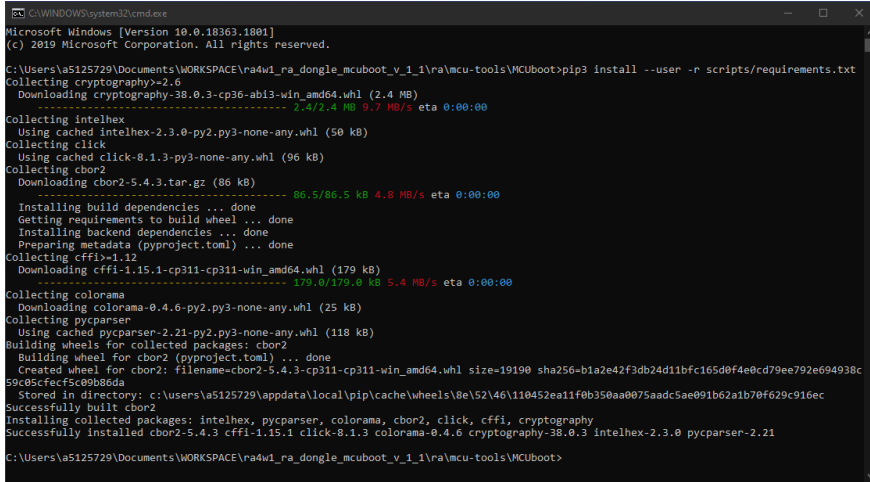


Figure 20. Installing the MCUboot dependencies

8. Close the Command Prompt.
9. Make sure the **ra4e1_iso_dongle_mcuboot_v_1_0** project is active (selected in Project Explorer) and on the top menu select **Project > Build Project** and wait until the process completes.
10. The next step is to configure the existing application to use the bootloader project. The software project that has to run on the dongle is duplicated and renamed so that the original can be still debugged using external debugging tool during further code development. Keep in mind that the binary size must be smaller or equal to 96-KB. This is the fixed image size and dedicated memory space in the code flash. It cannot be exceeded! Right-click in the e² studio Project Explorer view and select **Import** from the menu showing up to open the Import Wizard (Figure 15).
11. From the Import Wizard window select **Rename & Import Existing C/C++ Project into Workspace** and then click on the **Next >** button (Figure 21).

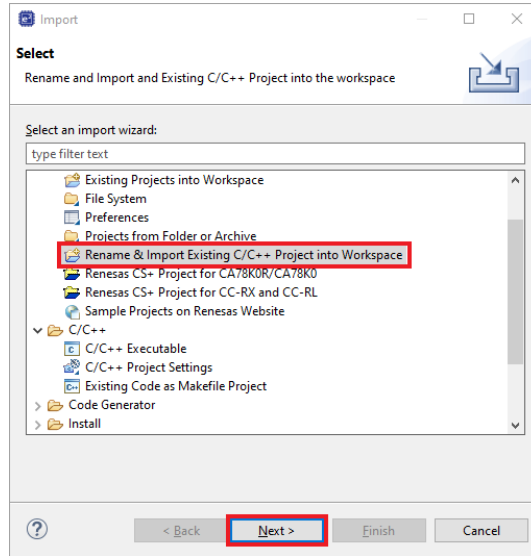


Figure 21. Selection of the Import Wizard Option

- On the next window enter a project name (Figure 22). Renesas recommends using the same name as the original project with a suffix *_app. Select the **Select root directory** option and then click on the **Browse...** button. In File Explorer, navigate to the location of the existing application project, select it, click on the **Select Folder** button, and select the required project in the **Projects:** section. Click on the **Finish** button to complete the import process and close the Import Wizard.

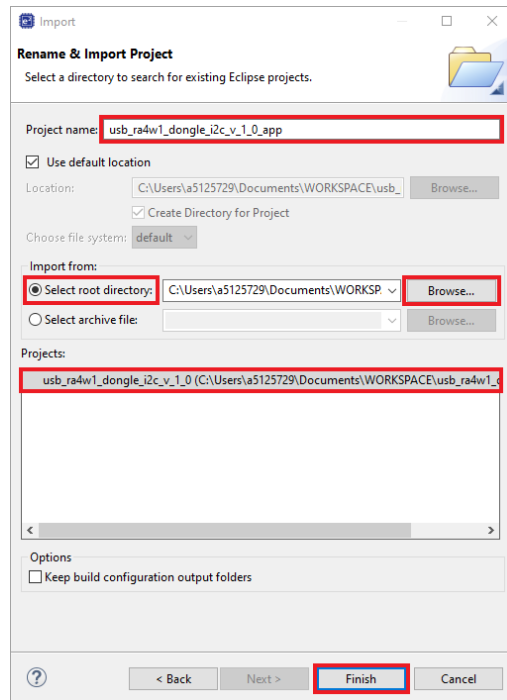


Figure 22. Selection of the Sample Project

- The project is now duplicated into the e² studio workspace and must be configured. In the **Project Explorer** right-click on the application project folder and select **Properties** to open the project properties window (Figure 23).

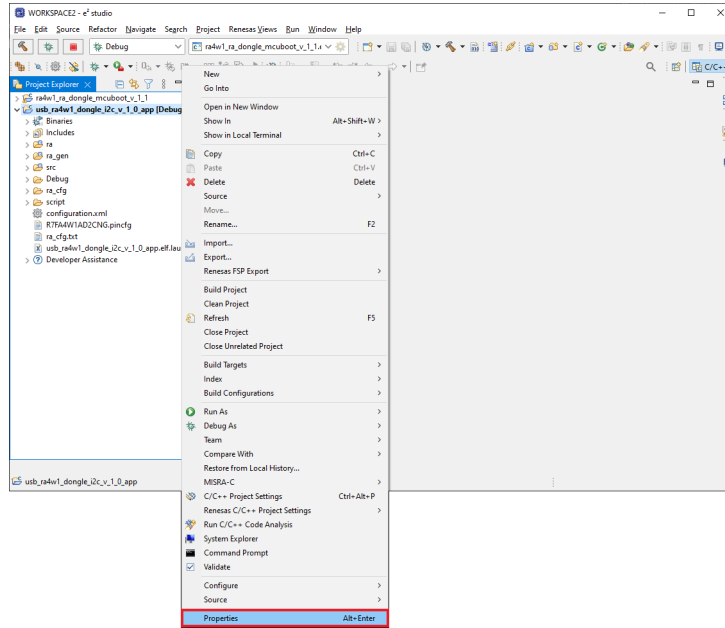


Figure 23. Opening Project Properties

14. Select **C/C++ Build > Build Variables**, click on the **Add...** button. In the **Variable name:** field enter **BootloaderDataFile**. Check the **Apply to all configurations** box (Figure 24). Change the **Type:** to **File** and in the bootloader project in **Value:** enter the following relative path to the *.ld file:
`$(workspace_loc:ra4e1_iso_dongle_mcuboot_v_1_0)/Debug/ra4e1_iso_dongle_mcuboot_v_1_0.ld`
Note: In case the bootloader project name or location are different, the value format is:
`$(workspace_loc:<boot_project_name>)/Debug/<boot_project_name>.ld`. Click on the **OK** button and then on the **Apply** button to save the change of the properties.

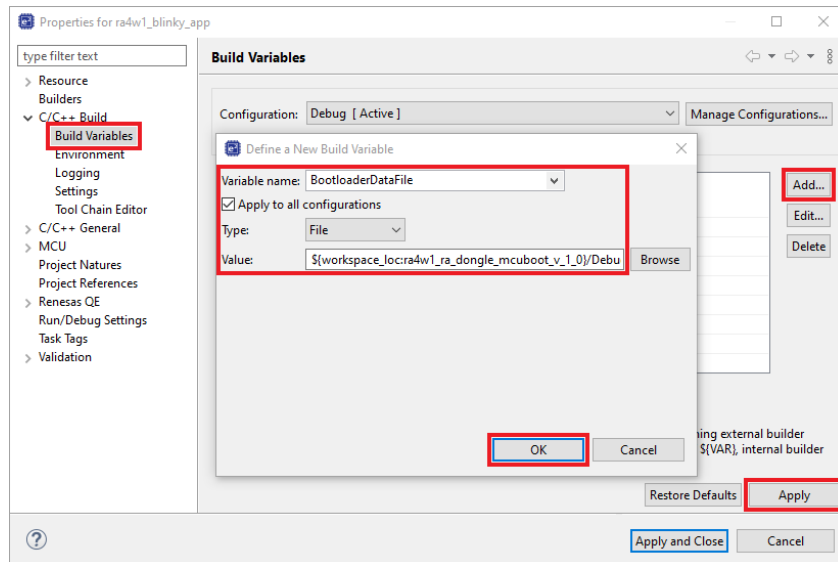


Figure 24. Adding a Build Variable to Use the Bootloader

15. Select **C/C++ Build > Environment**, click on the **Add...** button. In the **Name:** field enter **MCUBOOT_IMAGE_VERSION** and set the **Value:** to the actual application project version number (Figure 25). Click on the **OK** button and then on the **Apply** button to save the change of the properties.

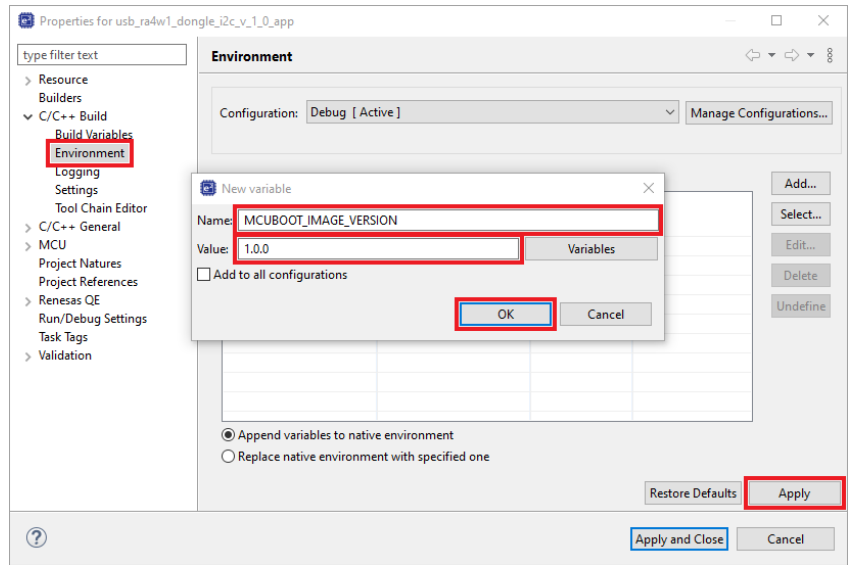


Figure 25. Adding a Project Version Number

16. Select **C/C++ Build > Settings** and then select the **Build Steps** tab (Figure 26). In the **Command(s):** field enter the following pre-build command: `del ${ProjName}.elf` to always delete the *.elf file. Click on the **Apply and Close** button to save the change of the properties and close the window.

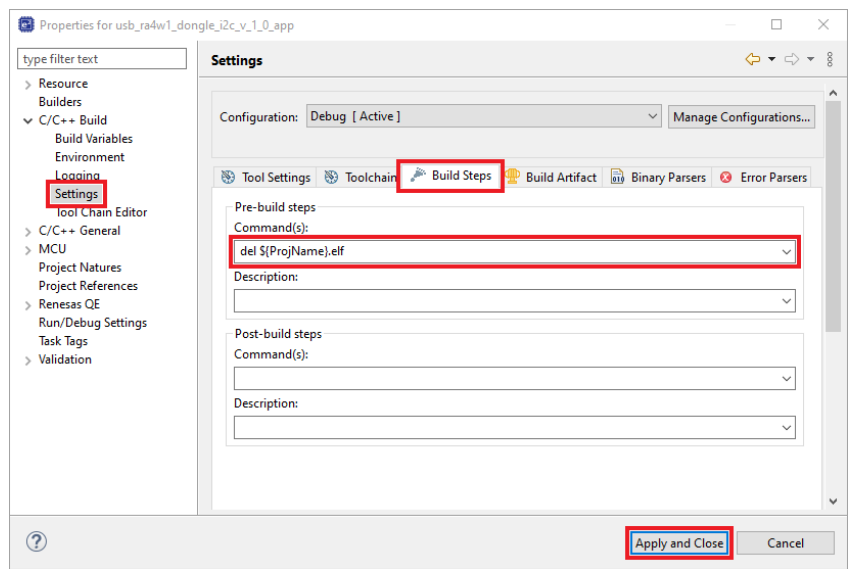


Figure 26. Adding a Pre-Build Command

17. Ensure the project is active (selected in Project Explorer), on the top menu select **Project > Build Project** and wait until the process completes (Figure 27). Then, in **Project Explorer** expand the **Debug** folder. The

generated application image binary file is under the name format **<app_project_name>.bin.signed**. From here it can be directly copied to a known location.

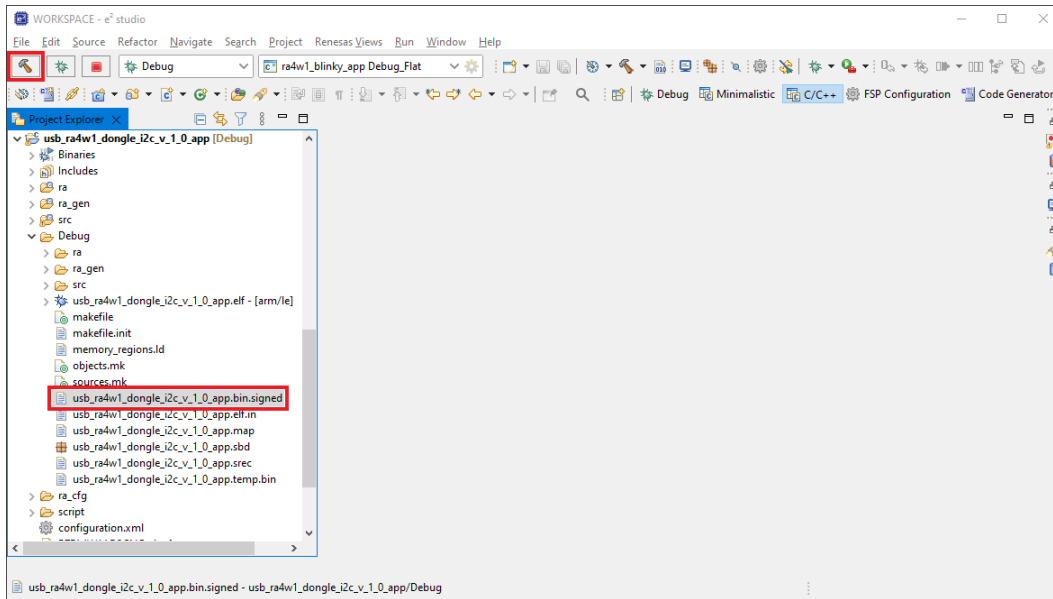


Figure 27. Generating the Application Image

For more information about the image generation process, refer to Renesas RA Family Secure Bootloader for RA2 MCU Series Application Note.

5. Revision History

Revision	Date	Description
1.00	Dec 12, 2023	Initial release.

IMPORTANT NOTICE AND DISCLAIMER

RENESAS ELECTRONICS CORPORATION AND ITS SUBSIDIARIES (“RENESAS”) PROVIDES TECHNICAL SPECIFICATIONS AND RELIABILITY DATA (INCLUDING DATASHEETS), DESIGN RESOURCES (INCLUDING REFERENCE DESIGNS), APPLICATION OR OTHER DESIGN ADVICE, WEB TOOLS, SAFETY INFORMATION, AND OTHER RESOURCES “AS IS” AND WITH ALL FAULTS, AND DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING, WITHOUT LIMITATION, ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NON-INFRINGEMENT OF THIRD PARTY INTELLECTUAL PROPERTY RIGHTS.

These resources are intended for developers skilled in the art designing with Renesas products. You are solely responsible for (1) selecting the appropriate products for your application, (2) designing, validating, and testing your application, and (3) ensuring your application meets applicable standards, and any other safety, security, or other requirements. These resources are subject to change without notice. Renesas grants you permission to use these resources only for development of an application that uses Renesas products. Other reproduction or use of these resources is strictly prohibited. No license is granted to any other Renesas intellectual property or to any third party intellectual property. Renesas disclaims responsibility for, and you will fully indemnify Renesas and its representatives against, any claims, damages, costs, losses, or liabilities arising out of your use of these resources. Renesas' products are provided only subject to Renesas' Terms and Conditions of Sale or other applicable terms agreed to in writing. No use of any Renesas resources expands or otherwise alters any applicable warranties or warranty disclaimers for these products.

(Disclaimer Rev.1.0 Mar 2020)

Corporate Headquarters

TOYOSU FORESIA, 3-2-24 Toyosu,
Koto-ku, Tokyo 135-0061, Japan
www.renesas.com

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/

Trademarks

Renesas and the Renesas logo are trademarks of Renesas Electronics Corporation. All trademarks and registered trademarks are the property of their respective owners.