Getting Started with the Renesas RZ/A2M Evaluation Board Kit

Required Resources
To build and run the RZ/A2M Evaluation Board Kit example, you will need following resources:

Development tools & software
- e²studio IDE v7.7.0 ([e²studio download](#))
- GNU ARM Embedded 6-2016q2-update (bundled in e²studio v7.7.0)

Hardware
- Renesas RZ/A2M Evaluation Board Kit, P/N: RTK7921053S00000BE#WS (https://www.renesas.com/jp/ja/products/software-tools/boards-and-kits/eval-kits/rz-a2m-evaluation-board-kit.html#orders)
- PC running Windows 10; the Tera Term console, or similar application; and an installed web browser (Google Chrome, Internet Explorer, Microsoft Edge, or Mozilla Firefox).
- Ethernet LAN internet access

Before you begin, see [Prerequisites](#).

If you do not have an RZ/A2M Evaluation Board Kit, you can order one from [Renesas](#).

Setting Up Your Environment

FreeRTOS for the RZ/A2M Evaluation Board Kit uses e²studio IDE and GNU ARM Embedded compiler. Before you begin, install the IDE and compiler to your machine:

To install e²studio:

1. Browse to [e²studio](#) and choose **Download Software. Make sure to use** e²studio version 7.7.0 or later.
2. Unzip and run the installer. Follow the prompts for the section 2.1 and 2.2 of the [e²studio Getting Started Guide](#).

In this document, **Wired Ethernet** and **SX-SDMAC** are indicated to the part needed different settings.
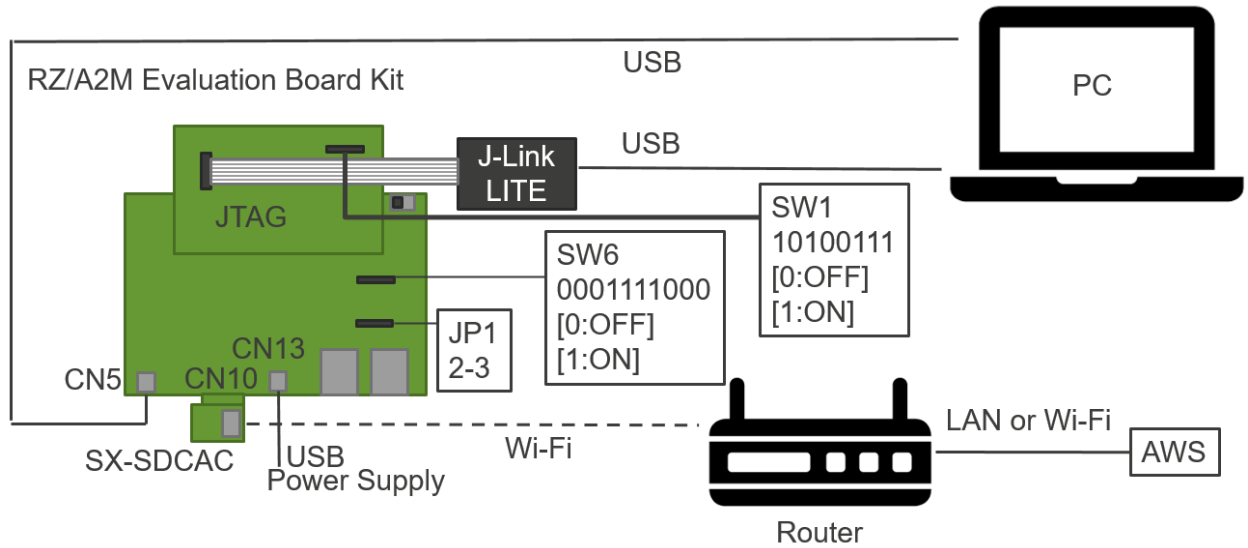
Connecting a Debugger

Wired Ethernet
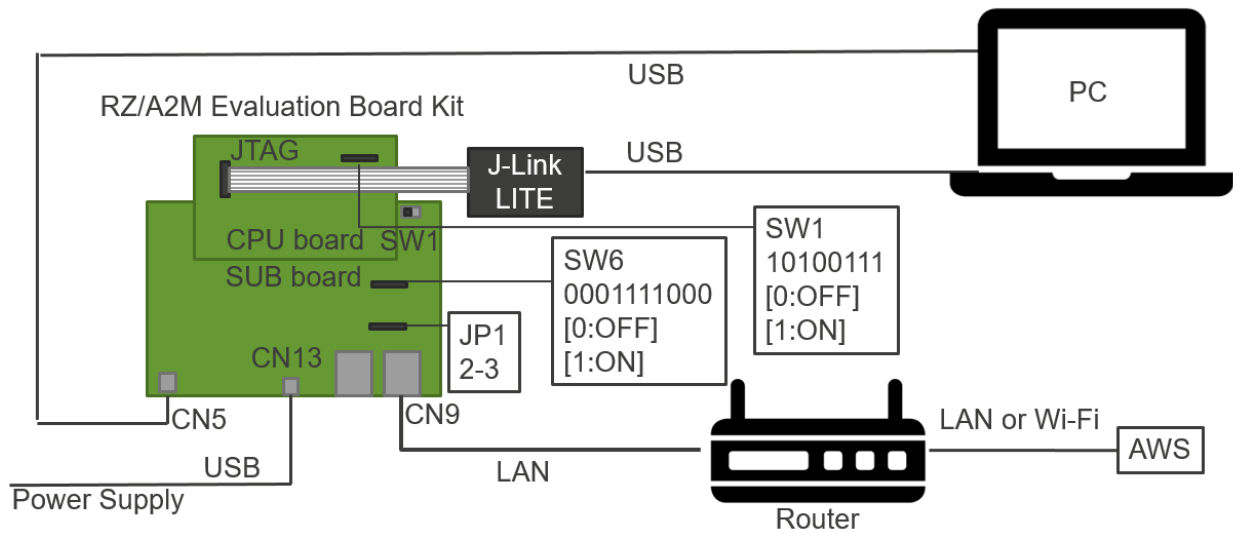
1. Confirm Power switch (SW1) on SUB board is ON (left).
2. Connect CPU board and SUB board.
3. Set SW1 on CPU board to 10100111.
4. Set SW6 on SUB board to 0001111000.
5. Short-circuit pin 2 and pin 3 of JP1 on SUB board.
6. Connect J-Link LITE to JTAG connector on CPU board.
7. Connect USB cable from J-Link LITE to a spare USB port on your PC.
8. Connect USB cable form CN5 on SUB board to a spare USB port on your PC.
9. Connect LAN cable from CN9 on SUB board to a router can access AWS.

SX-SDMAC

1. Confirm Power switch (SW1) on SUB board is ON (left).
2. Connect CPU board and SUB board.
3. Set SW1 on CPU board to 10100111.
4. Set SW6 on SUB board to 0001111000.
5. Short-circuit pin 2 and pin 3 of JP1 on SUB board.
6. Connect J-Link LITE to JTAG connector on CPU board.
7. Connect USB cable from J-Link LITE to a spare USB port on your PC.
8. Connect USB cable form CN5 on SUB board to a spare USB port on your PC.
9. Insert SX-SDCAC to CN10 on SUB board.

Download and Build FreeRTOS

After your environment is set up, you can download 'Renesas RZ/A2M Evaluation Board Kit Application Example' and run the demo code.

Download FreeRTOS

1. Browse to the GitHub Page and download the code.
2. Unzip the downloaded file to a folder and make a note of the folder path. In this tutorial, this folder is referred to as `BASE_FOLDER`.

Note: The e^{2}studio doesn't support long path names. To accommodate the files in the FreeRTOS projects, make sure the path to the directory is less than 260 characters and does not contain spaces or special characters.

Import the FreeRTOS Demo Code into Your IDE

**To import the FreeRTOS demo code into e^{2}studio IDE**

1. e^{2}studio integrates various tools such as compiler, an assembler, debugger and an editor into a common graphical user interface. Start e^{2}studio:

   ```
   Windows™ 10: Start Menu>All Apps> Renesas Electronics e2studio>e2studio
   ```

2. In the 'Select a workspace' folder that appears, browse to the folder "…`BASE_FOLDER`\amazon". Click 'OK' to continue.
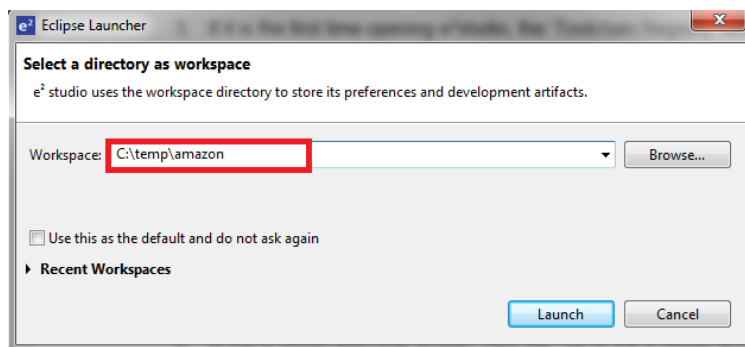
3. If it is the first time opening e²studio, the 'Toolchain Registry' window will open. In the 'Toolchain Registry' dialog select GCC ARM Embedded and ensure that '6.3.1.20170620' is selected. Click 'Register'. A dialog will appear "Selected Toolchains were successfully integrated with e²studio ". Click 'OK'.

4. In the 'Code Generator Registration' dialog click 'OK'. This window opens up first time only after installation.

5. A 'Code Generator COM component register' dialog will pop-up with the text "Please restart e²studio to use Code Generator". Click 'OK'.

6. In the 'Restart e²studio dialog, click 'OK'.

7. Once e²studio is restarted, then 'Select a workspace' window appears again with the folder path selected in step 2. Click 'OK'.

8. In the e²studio welcome screen, click 'Go to the e²studio workbench' arrow icon, on the far right.

9. Right click in the Project Explorer window, and select 'Import'.

10. In the import wizard, select General > Existing Projects into Workspace, and click 'Next'.
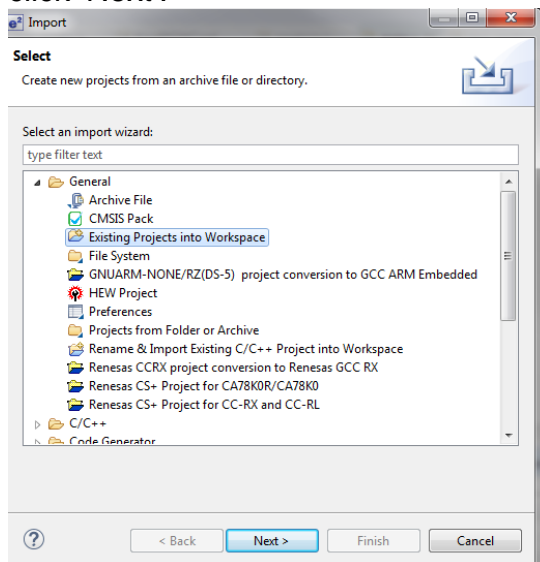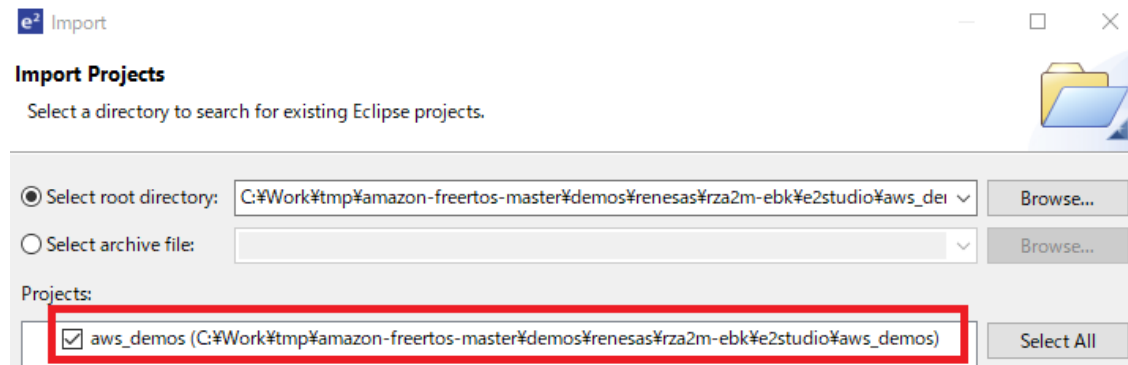
11. Click the 'Browse' button, and locate the following directory

   Wired Ethernet

   '*<BASE_FOLDER>*\amazon\demos\renesas\rza2m-ebk\e2studio\aws_demos'.

   SX-SDMAC

   '*<BASE_FOLDER>*\amazon\demos\renesas\rza2m-ebk-sdio-sx-sdmac\e2studio\aws_demos'.
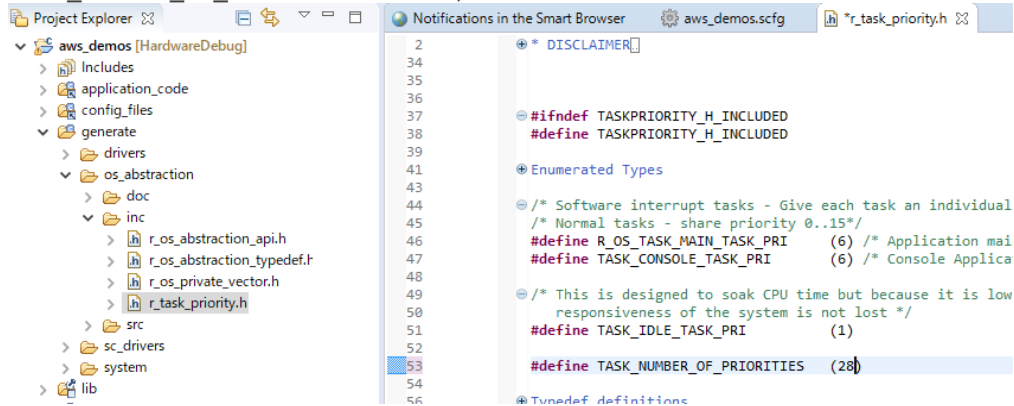

12. Click "Finish".

13. **SX-SDMAC** requires extra settings below. Steps a. to i. are required only once:
   a. Download [RZ/A2M SDIO Wi-Fi Package](#).
   b. Add SILEX SX-SDMAC driver component to your PC by following the section 4 of [the release note of the package](#).
   c. Doble click `aws_demos.scfg`.
   d. Select **Components** tag.
   e. Press ⬆ icon.
   f. Software Component Selection dialog will be appaired. Click **Download more software components**.
   g. Region Setting dialog will be appaired. Select your region and click OK.
   h. Select **RZ/A2M I/O definition header file** and **RZ/A2M Software Core Package V7.00**, and press download.
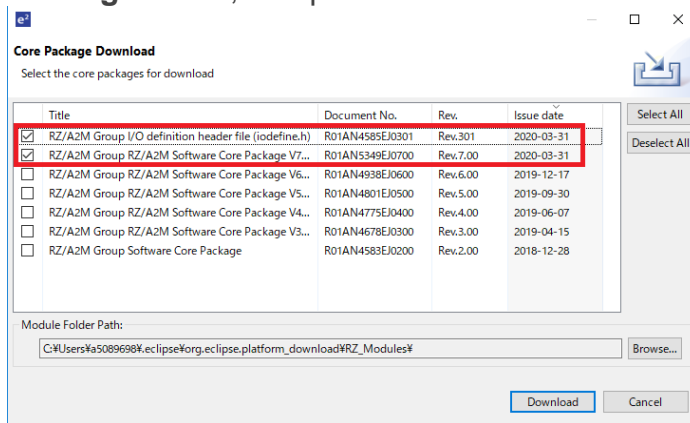
   i. Close Software Component Selection dialog by pressing Cancel button.
   j. Press ▶ icon to generate drivers.
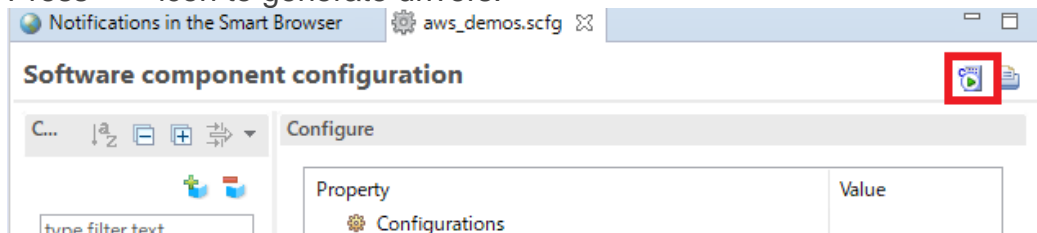
k. Open `generate\os_abstraction\inc\r_task_priority.h`, modify the value of `TASK_NUMBER_OF_PRIORITIES` to 28, and save the file.

l. Open `application_code\common_demos\include\aws_clientcredential.h`, modify the value of clientcredentialWIFI_SSID and clientcredentialWIFI_PASSWORD, and save the file.

14. In the **Project** menu, choose **Project->Build All**. The project should build with no errors.

Configure Your Project

To configure your project, you need to know your AWS IoT endpoint and Thing name that represents your board.

Configure AWS IoT endpoint

1. Login to aws account and Click on IoT Core services.
2. In the left navigation pane, choose **Settings**.
3. Copy your AWS IoT endpoint from the **Endpoint** text box. It should look like `<1234567890123>`.iot.`<us-east-1>`.amazonaws.com.
4. Open `aws_demos/application_code/common_demos/include/aws_clientcredential.h` and set `clientcredentialMQTT_BROKER_ENDPOINT` to your AWS IoT endpoint.

   static const char clientcredentialMQTT_BROKER_ENDPOINT[] = "Paste AWS IoT Broker endpoint here.";

5. In the left navigation pane, Click on Manage-> Things, and then Click on 'Create" to create a new Thing.
6. In the next window, click on "Create a single thing".

Creating AWS IoT things

An IoT thing is a representation and record of your phyisical device in the cloud. Any physical device needs a thing record in order to work with AWS IoT. Learn more.

Register a single AWS IoT thing
Create a thing in your registry

> Create a single thing

Bulk register many AWS IoT things
Create things in your registry for a large number of devices already using AWS IoT, or register devices so they are ready to connect to AWS IoT.

> Create many things

7. Enter thing Name for your IoT board.

CREATE A THING
Add your device to the thing registry

This step creates an entry in the thing registry and a thing shadow for your device.

Name

Give your thing a name

8. Open `aws_demos\application_code\common_demos\include\aws_clientcredential.h`. Specify AWS IoT thing for your board in the following `#define` constants from **Thing** pane in [AWS IoT console](#).

   ```
   #define clientcredentialIOT_THING_NAME "Paste AWS IoT Thing name here."
   ```

9. Click next. In next window click on "Create Certificate"

10. Download the certificate.

11. Activate the certificate.

Create AWS IoT policy

1. In the left navigation pane, Click on **Secure**-> **Policies**, and then Click on "**Create a policy**" or "**Create**" to create a new policy.
2. Enter Policy name for your test

3. Enter **iot:Connect** in Action box, replace replaceWithAClientId with **MQTTEcho** in Resource ARN box, check **Allow** in Effect, and click on **Add statement**.

4. Enter **iot:Publish** in Action box, replace replaceWithATopic with **freertos/demos/echo** in Resource ARN box, check **Allow** in Effect, and click on **Add statement**.

5. Enter **iot:Subscribe** in Action box, replace replaceWithATopicFilter with **freertos/demos/echo** in Resource ARN box, check **Allow** in Effect, and click on **Add statement**.

6. Enter **iot:Receive** in Action box, replace replaceWithATopic with **freertos/demos/echo** in Resource ARN box, check **Allow** in Effect, and click on **Create**.

7. In the left navigation pane, Click on **Secure**-> **Certificates**, and then Click on certificate created in the sequence 10 of **Configure AWS IoT endpoint** section above.

8. Click on Actions and select **Attach policy**.

9. Check the policy you created, then click on **Attach**.
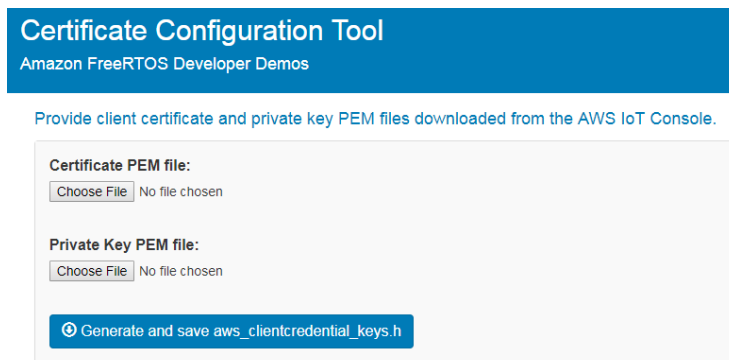
Configure certificate and private key

The certificate and private key must be hard-coded into the FreeRTOS demo code. This is for demo purposes only. Production level applications should store these files in a secure location. FreeRTOS is a C language project, and the certificate and private key must be specially formatted to be added to the project.

To format your certificate and private key

1. In a browser window, open certificate configuration tool from project `<BASE_FOLDER>`\tools\certificate_configuration\CertificateConfigurator.html.

2. Under **Certificate PEM file**, choose `certificate.pem.crt` you downloaded from the AWS IoT console in previous step.
3. Under **Private Key PEM file**, choose `private.pem.key` you downloaded from the AWS IoT console in previous step.
4. Choose **Generate and save aws_clientcredential_keys.h**, and then save the file in`<BASE_FOLDER>`\demos\common\include. This overwrites the file aws_clientcredential_keys.h in the directory.

Configure MAC address

MAC address is NOT stored in the storage memory on the board. Therefore, you need to set MAC address to your project.

1. Get your MAC address

    You can find your MAC address on CN9 of the board.
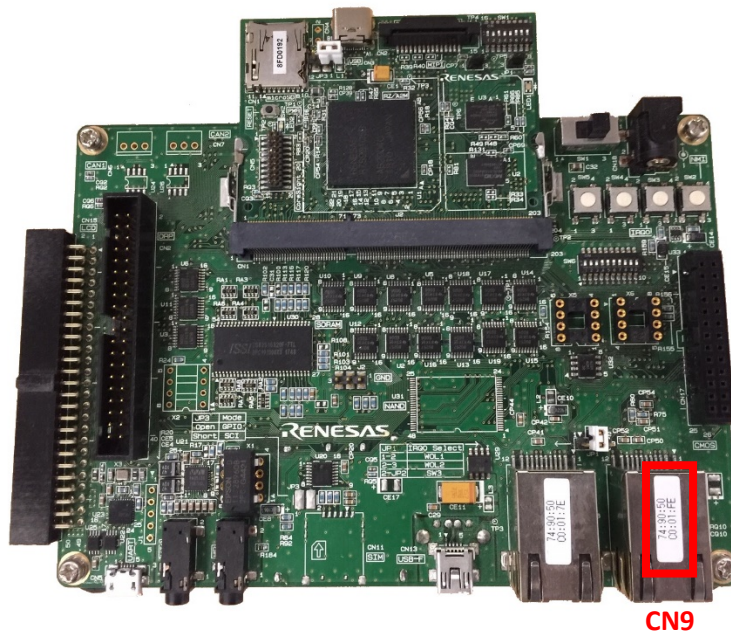
2. Set MAC address in your code

    Edit **configMAC_ADDR**N (N=0, 1, ... ,5) macros defined in **FreeRTOSConfig.h** to the MAC address printed on CN9. Ethernet driver is configured to use CN9.

    In the case your MAC address is 01:23:45:67:89:AB, set macros as follows:

    ```
    #define configMAC_ADDR0                          0x01
    #define configMAC_ADDR1                          0x23
    #define configMAC_ADDR2                          0x45
    #define configMAC_ADDR3                          0x67
    #define configMAC_ADDR4                          0x89
    #define configMAC_ADDR5                          0xAB
    ```

Run the FreeRTOS Demo

To run the FreeRTOS demos on the RZ/A2M Evaluation Board Kit:

1. Sign in to the <u>AWS IoT console</u>.
2. In the left navigation pane, choose **Test** to open the MQTT client.
3. In the **Subscription topic** text box, type '**freertos/demos/echo**', and then choose **Subscribe to topic**.
4. Rebuild the project, "**Project->Build All**".
5. Connect USB cable from J-Link LITE to a spare USB port on your PC.
6. Connect USB cable from CN13 on SUB board to a power supply.
7. The debugging can be started by clicking the 'Run-> Debug Configuration'. Click the symbol **"aws_demos HardwareDebug"** under 'Renesas GDB Hardware Debugging' by expanding the list.
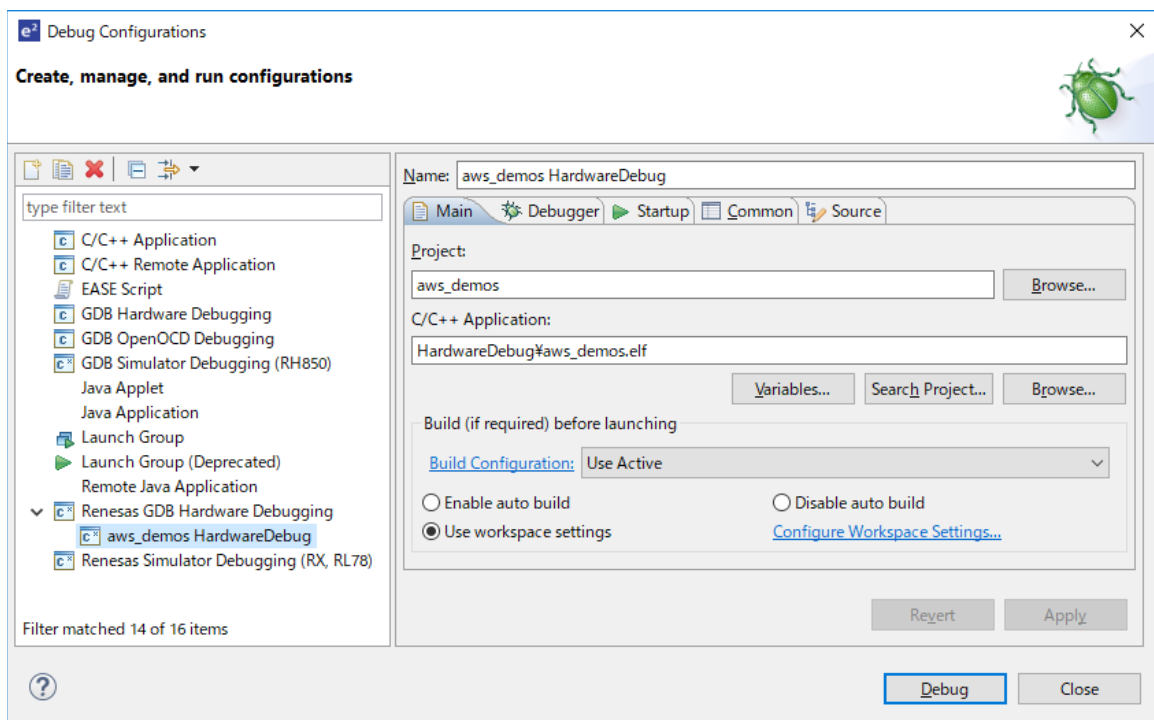
8. Click the '**Debug**' button to download the code to the target board to begin debugging. A firewall warning may be displayed for 'e2-server-gdb.exe'. Select the check-box for 'Private networks, such as my home or work network', and click 'Allow access'.
9. e²studio may ask you to change to the 'Renesas Debug Perspective'. Click 'Yes'.
10. Once the code has been downloaded, click the 'Resume' button to run the code up to the first line of the main function.
11. <mark>SX-SDMAC</mark> Remove SX-SDMAC module and re-insert it to the board.
    This procedure is specific to RZ/A2M Evaluation Board Kit which can't stop power supply to SDIO.
    For more details, please refer to RTK79210XXB00000BE User's Manual.
12. Click 'Resume' button again to run the target through the rest of the code.

In the AWS IOT console MQTT client, you should see the MQTT messages sent by your device.

Note:

Please visit the following GitHub repository to get the latest projects (prototype), but not yet certified for other Renesas devices, compilers, and target boards.

https://github.com/renesas-rz/amazon-freertos

Troubleshooting

If no messages appear in the AWS IoT console, try the following:

1. Check that your network credentials are valid.
2. Verify the switch settings on your board.

Test OTA demonstration

Before testing OTA demonstration, it is recommended to test FreeRTOS Demo.

To realize OTA function, QSPI Flash on the RZ/A2M Evaluation Board is treated to split to 3 area stated below:

> Boot Area – OTA boot proc is stored. OTA boot proc decides the firmware to execute.

> Execution Area – Firmware to execute.

> Temporary Area – Downloaded new firmware. If firmware exists in this area, OTA boot proc checks the firmware and copy it to Execution Area.

New firmware image is stored in AWS S3 Services, and is downloaded via AWS IoT Services.
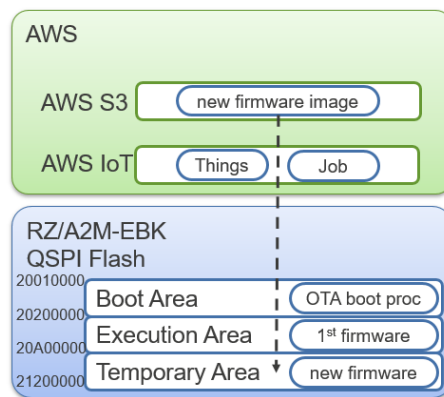
In this section following steps are described:

1. Install needed software into PC.
2. Prepare settings by AWS console.
3. Generate the certificate used in OTA update.
4. Create firmware.
5. Create OTA job.
6. Execute OTA.

Install needed software into PC

Following software are needed to test OTA demonstration.

- OpenSSL
  - [Windows Download site](#)
    Light version is OK.
  - Add path to `openssl.exe`.

- Python3.7
  - [Download site](#)
  - ECDSA is needed to install by following command:
    ```
    $ pip install scdsa
    ```

- Serial terminal software that can transmit binary file with 115200bps
  - In the case of [TeraTerm](#), following settings are required:
    - Setup-> Terminal
      Receive: LF, Transit: CR
    - Setup -> Serial Port
      Speed: 115200bps, Data: 8bit, Parity: none, Stop bits: 1 bit, Flow control: none
    - Setup -> General
      Language: English, Language UI: Default.lng
    - Setup -> Setup directory -> tera term Configuration File -> Open File
      Modify `FileSendHighSpeedMode=on` to `off`.

- Segger J-Link driver
  - [Download site](#)

Prepare settings by AWS console

This section shows how to create S3 bucket to store the new firmware image, and create a role and policies required for OTA update.

This section is based on the following document:

https://docs.aws.amazon.com/freertos/latest/userguide/ota-prereqs.html

Create S3 bucket to store the new firmware image by following steps:

1. Launch web browser, and sign in to AWS console.
2. Go to S3 console by inputting **S3** in **Find Services** in **AWS Management Console**.
3. Press **Create bucket** button.
4. Input **Bucket name** (in this document *rz-ota* is named as an example), and press **Create bucket** button.
5. Click created bucket.
6. Select **properties** tag.
7. Enable **Versioning**.

Create a role and policies required for OTA update by following steps:

8. Go to IAM console by pressing **Services** at the top of the screen, and inputting **IAM**.
9. Click Roles in the navigation pain at the left of the screen.
10. Press **Create role** button.
11. Choose **IoT** in the service list, select **IoT** in the use case list, and press **Next:Permissions** button.
12. Confirm 3 policies are displayed, and press **Next:Tags** button.
13. Input any keys if needed, and press **Next:Review** button.
14. Input Role name (in this document *role_rz-ota* is named as an example), and press **Create role** button.
15. Click created role in the role list.
16. Press **Attach policies** button.
17. Check the box at the left of **AmazonFreeRTOSOTAUpdate**, and press **Attach policy**.

18. Press **Add inline policy** at the right of Attach policies button.
19. Select **JSON** tab, copy following text to the text box, and press **Review policy** button:

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

Replace *nnnnnnnnnnnn* to your AWS account ID composed of 12 numbers.
Replace *role_rz-ota* to the name you named at step 14.

20. Input Name (in this document *policy_rz-ota* is named as an example), and press **Create policy** button.
21. Press **Add inline policy** at the right of Attach policies button.
22. Select **JSON** tab, copy following text to the text box, and press **Review policy** button:

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

Replace *rz-ota* to the name you named at step 4.

23. Input Name (in this document *policy_rz-ota-S3* is named as an example), and press **Create policy** button.

Generate the certificate used in OTA update

This section shows how to generate certificate required for OTA update using OpenSSL.

Launch command prompt, go to your work folder, and enter following commands:

```
$ openssl ecparam genkey name secp256r1 out ca.key
$ openssl req x509 sha256 new nodes key ca.key days 3650 out ca.crt
        Input Country Name, State or Province Name, and other required information.
$ openssl ecparam genkey name secp256r1 out secp256r1.keypair
$ openssl req new sha256 key secp256r1.keypair > secp256r1.csr
        Input Country Name, State or Province Name, and other required information.
$ openssl x509 req sha256 days 3650 in secp256r1.csr CA ca.crt CAkey ca.key CAcreateserial out secp256r1.crt
$ openssl ec in secp256r1.keypair outform PEM out secp256r1.privatekey
$ openssl ec in secp256r1.keypair outform PEM pubout out secp256r1.publickey
```

Create firmware

This section shows how to create ota_boot_proc, initial firmware, and new firmware.

Create ota_boot_proc by following steps:

1. Launch e^2 studio. If aws_demos project already exists, delete it. Import following 2 projects.

   Wired Ethernet

   - aws_demos (*<BASE_FOLDER>*\amazon\demos\renesas\rza2m-ebk\e2studio\ota_demos)
   - ota_boot_proc (*<BASE_FOLDER>*\amazon\demos\renesas\rza2m-ebk\e2studio\ota_boot)

   SX-SDMAC

   - aws_demos (*<BASE_FOLDER>*\amazon\demos\renesas\rza2m-ebk-sdio-sx-sdmac\e2studio\ota_demos)
   - ota_boot_proc (*<BASE_FOLDER>*\amazon\demos\renesas\rza2m-ebk-sdio-sx-sdmac\e2studio\ota_boot)

2. Open secp256r1.publickey generated at the last section by text editor.
3. Open ota_boot_proc\src\key\code_signer_public_key.h.
4. Copy public key described in secp256r1.publickey to CODE_SIGNENR_PUBLIC_KEY_PEM in code_signer_public_key.h like below:

```
#define CODE_SIGNENR_PUBLIC_KEY_PEM "-----BEGIN PUBLIC KEY-----"\
"ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz012345678901"\
"ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz012345=="\
"-----END PUBLIC KEY-----"
```
   Note that red character is needed to add.

5. Build **ota_boot_proc** project.

Create initial firmware by following steps:

6. Copy your AWS IoT endpoint from the Endpoint text box. It should look like
   <1234567890123>.iot.*<us-east-1>*.amazonaws.com.

7. Open `aws_demos/application_code/common_demos/include/aws_clientcredential.h`
   and set `clientcredentialMQTT_BROKER_ENDPOINT` to your AWS IoT endpoint.

   ```
   static const char clientcredentialMQTT_BROKER_ENDPOINT[] = "Paste AWS IoT Broker endpoint
   here.";
   ```

8. Open `aws_demos\application_code\common_demos\include\aws_clientcredential.h`.
   Specify AWS IoT thing for your board in the following `#define` constants from
   Thing pane in AWS IoT console.

   ```
   #define clientcredentialIOT_THING_NAME "Paste AWS IoT Thing name here."
   ```

9. Format your certificate and private key following the way described in *To format
   your certificate and private key*.

10. Open
    <BASE_FOLDER>\lib\third_party\mcu_vendor\renesas\rz_mcu_boards\core_package\ge
    nerate\linker_script.ld by a test editor.

11. Modify `ROM (rx):ORIGIN` to **0x20200300** from **0x20010000**, save the file, and
    close. 0x20010000 is the start address for the environment in which
    ota_boot_proc is not used, or for debugging. 0x20200300 is the start address for
    the environment using ota_boot_proc.

12. Open
    `aws_demos/application_code/common_demos/include/aws_ota_codesigner_certificate`
    `.h` and copy the content of `secp256r1.crt` like below:

    ```
    static const char signingcredentialSIGNING_CERTIFICATE_PEM[] = "-----BEGIN CERTIFICATE-----\n"
    "ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz012345678901\n"
    "ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz012345678901\n"
    "-----END CERTIFICATE-----";
    ```

13. Open
    `aws_demos/application_code/common_demos/include/demo_runner/aws_demo_runner.c`.
    And modify the file to disable `vStartMQTTEchoDemo()` and enable
    `vStartOTAUpdateDemoTask()` like below:

    ```
    /* Demo declarations. */
    /* extern void vStartMQTTEchoDemo( void ); */
    extern void vStartOTAUpdateDemoTask( void );

    void DEMO_RUNNER_RunDemos( void )
    {
        /*vStartMQTTEchoDemo();*/
        vStartOTAUpdateDemoTask();
    }
    ```

14. SX-SDMAC requires extra settings described in step13 of *Import the FreeRTOS
    Demo Code into Your IDE*.

15. In the Project menu, choose Project->Build All. The project should build with no errors.
16. Confirm `HardwareDebug\aws_demos.bin` is generated.
17. Copy following 3 files to your work folder.
    a. HardwareDebug\aws_demos.bin
    b. *<BASE_FOLDER>*\lib\third_party\mcu_vendor\renesas\rz_mcu_boards\tools\initial-image-gen.py
    c. secp256r1.privatekey generated at the last section.
18. Launch command prompt and go to your work folder.
19. Enter following command to generate initial firmware named `userprog.rsu`:

```
$ python initial-image-gen.py
```

Create new firmware by following steps:

20. Open
    `aws_demos/application_code/common_demos/include/aws_application_version.h` and
    modify the value of `APP_VERSION_BUILD` macro to 3 from 2.
21. In the Project menu, choose Project->Build All. The project should build with no
    errors.
22. Confirm **HardwareDebug\aws_demos.bin** is generated.
23. Copy following 3 files to your work folder.
    d. HardwareDebug\aws_demos.bin
    e. *<BASE_FOLDER>*\lib\third_party\mcu_vendor\renesas\rz_mcu_boards\to
       ols\update-image-gen.py
    f. secp256r1.privatekey generated at the last section.
24. Launch command prompt and go to your work folder.
25. Enter following command to generate new firmware named `userprog.rsu`:

    ```
    $ python update-image-gen.py
    ```

    Note that the name of the generated file is the same as initial firmware and
    the old file will be overwritten.

    You can change the file name and sequence number by modifying update-
    image-gen.py.

    ```
    # Input Filename
    input_file               = 'aws_demos.bin'
    # Output Filename
    output_file              = 'userprog.rsu'
    # Input Key
    input_key_file           = 'secp256r1.privatekey'
    # Firmware version (sequence number)
    sequence_number          = 1
    ```

Create OTA job

In this section, upload the new firmware, and create AWS IoT job by AWS console.

Upload the new firmware to Amazon S3 by following steps:

1. Launch web browser, and sign in to AWS console.
2. Go to Amazon **S3** console.
3. Click **buckets** at the left of the screen.
4. Click the created bucket (*rz-ota* is named in this document).
5. Press **Upload**.
6. Drag and drop the generated new firmware, and press **Next** button.
7. Manage users is displayed. Press **Next** button.
8. Storage class is displayed. Press **Next** button.
9. Press **Upload** button.

Create AWS IoT job by following steps. They are based on the document below:

https://docs.aws.amazon.com/freertos/latest/userguide/ota-console-workflow.html

10. Go to AWS **IoT** console by searching **iot core**.
11. Select **Manage** at the left of screen.
12. Select **Jobs** at the left of screen.
13. Press **Create** button.
14. Press **Create OTA update job** button.
15. Click **Select**, check the generated thing, and press **Next** button.
16. Click **Create** in the Code signing profile box.
17. **Create a code signing profile** dialog will be appaired.
18. Enter **Profile name**.
19. Click **Select** in Device hardware platform box, click **Select** at the right of **Windows Simulator**.
20. Click Import in Code signing certificate, select following files generated at *Generate the certificate used in OTA update* section:
    a. Select Certificate - `secp256r1.crt`
    b. Select Certificate private key - `secp256r1.privatekey`
    c. Select Certificate chain (optional) - `ca.crt`
21. Press **Import** button, enter **Pathname of code signing certificate on device**, and click **Create** button.
22. Click **Select** in Select your firmware image in S3 or upload it, click S3 bucket (*rz-ota* is named in this document), and click **Select** at the right of uploaded new firmware.
23. Enter **Pathname of firmware image on device**.
24. Click **Select** in the Role (requires S3 access), click **Select** at the right of the created role (*role_rz-ota* is named in this document), and press **Next** button.
25. Enter a job name in the box under the **ID**, and press **Create** button. It is recommended to name with unique number.

Execute OTA

In this section, erase QSPI flash on RZ/A2M Evaluation Board Kit, download ota_boot_proc, download initial firmware, and update to the new firmware.

Erase QSPI flash on RZ/A2M Evaluation Board Kit by the following steps:

1. Launch SEGGER J-Link Commander
2. Connect USB cable from J-Link LITE to a spare USB port on your PC.
3. Connect USB cable from CN13 on SUB board to a power supply.
4. Input following commands. J-Link commander terminates by exit command:

```
J-Link>connect
Device>R7S921053VCBG_SPIBSC_SERIALFLASH
TIF>          (Press Enter without input)
JTAGConf>     (Press Enter without input)
Speed>        (Press Enter without input)
J-Link>r
J-Link>h
J-Link>erase
J-Link>exit
```

Download ota_boot_proc by the following steps:

5. Launch e^2 studio.
6. The debugging can be started by clicking the 'Run-> Debug Configuration'. Click the symbol "ota_boot_proc HardwareDebug" under 'Renesas GDB Hardware Debugging' by expanding the list.
7. e^2 studio may ask you to change to the 'Renesas Debug Perspective'. Click 'Yes'.
8. Once the code has been downloaded, click the 'Resume' button to run the code up to the first line of the main function.

Download firmware, and execute them by the following steps:

9. Launch serial terminal software.
10. Click 'Resume' button again to run the target through the rest of the code.
11. Following message will be displayed in the serial terminal software:

```
-----------------------------------------------
RZ/A2M secure boot program
-----------------------------------------------
Checking flash ROM status.
bank 0 status = 0xff [LIFECYCLE_STATE_BLANK]
bank 1 status = 0xff [LIFECYCLE_STATE_BLANK]
start installing user program.
========== install user program phase ==========
send "userprog.rsu" via UART.
```

12. Send initial firmware as binary data. Initial firmware will be executed after rebooting.

```
DownLoad Image Size:0x0009d000
downloaded:0x00000fff
downloaded:0x0009cfff
completed installing firmware.
integrity check scheme = sig-sha256-ecdsa
bank1(temporary area) on code flash integrity check...OK
completed installing const data.
software reset...
-----------------------------------------------
RZ/A2M secure boot program
-----------------------------------------------
Checking flash ROM status.
bank 0 status = 0xff [LIFECYCLE_STATE_BLANK]
bank 1 status = 0xfe [LIFECYCLE_STATE_TESTING]
integrity check scheme = sig-sha256-ecdsa
bank1(temporary area) on code flash integrity check...OK
update LIFECYCLE_STATE from [LIFECYCLE_STATE_TESTING] to [LIFECYCLE_STATE_VALID]
bank1(temporary area) block0 erase (to update LIFECYCLE_STATE)...bank1(temporary area)
block0 write (to update LIFECYCLE_STATE)...swap bank...
-----------------------------------------------
RZ/A2M secure boot program
-----------------------------------------------
Checking flash ROM status.
bank 0 status = 0xfc [LIFECYCLE_STATE_VALID]
bank 1 status = 0xff [LIFECYCLE_STATE_BLANK]
integrity check scheme = sig-sha256-ecdsa
bank0(execute area) on code flash integrity check...OK
jump to user program
```

13. Confirm **OTA demo version 0.9.2** is displayed.

```
0 1 [IP-task] prvIPTask started
1 513 [Tmr Svc] recover retry count = 4.
2 513 [Tmr Svc] EEPROM(main) hash check...
3 515 [Tmr Svc] NG
4 515 [Tmr Svc] EEPROM(mirror) hash check...
5 517 [Tmr Svc] NG
6 517 [Tmr Svc] write EEPROM(main)...
7 1502 [Tmr Svc] OK
8 1503 [Tmr Svc] write EEPROM(mirror)...
9 2488 [Tmr Svc] OK
10 2488 [Tmr Svc] EEPROM setting OK.
11 2488 [Tmr Svc] EEPROM(main) hash check...
12 2490 [Tmr Svc] OK
13 2490 [Tmr Svc] EEPROM(mirror) hash check...
14 2492 [Tmr Svc] OK
15 2501 [Tmr Svc] Write certificate...
45 8864 [OTA] OTA demo version 0.9.2
```

14. Confirm starting OTA update automatically.

15. Confirm ota_boot_proc is executed after downloading the new firmware image.

```
2076 51267 [OTA Task] [prvPAL_ActivateNewImage] Changing the Startup Bank
2082 56267 [OTA Task] [prvPAL_ResetDevice] Resetting the device.
------------------------------------------------
RZ/A2M secure boot program
------------------------------------------------
Checking flash ROM status.
bank 0 status = 0xfc [LIFECYCLE_STATE_VALID]
bank 1 status = 0xfe [LIFECYCLE_STATE_TESTING]
integrity check scheme = sig-sha256-ecdsa
bank1(temporary area) on code flash integrity check...OK
update LIFECYCLE_STATE from [LIFECYCLE_STATE_TESTING] to [LIFECYCLE_STATE_VALID]
bank1(temporary area) block0 erase (to update LIFECYCLE_STATE)...bank1(temporary area) block0
write (to update LIFECYCLE_STATE)...swap bank...
------------------------------------------------
RZ/A2M secure boot program
------------------------------------------------
Checking flash ROM status.
bank 0 status = 0xfc [LIFECYCLE_STATE_VALID]
bank 1 status = 0xff [LIFECYCLE_STATE_BLANK]
integrity check scheme = sig-sha256-ecdsa
bank0(execute area) on code flash integrity check...OK
jump to user program
```

16. Confirm OTA demo version is changed. Confirm the version number is changed to 0.9.3 from 0.9.2. This indicates firmware has been updated.

```
0 1 [IP-task] prvIPTask started
1 513 [Tmr Svc] recover retry count = 4.
2 513 [Tmr Svc] EEPROM(main) hash check...
3 515 [Tmr Svc] OK
4 515 [Tmr Svc] EEPROM(mirror) hash check...
5 517 [Tmr Svc] OK
6 517 [Tmr Svc] EEPROM(main) hash check...
7 519 [Tmr Svc] OK
8 519 [Tmr Svc] EEPROM(mirror) hash check...
9 521 [Tmr Svc] OK
10 530 [Tmr Svc] Write certificate...
11 943 [Tmr Svc] recover retry count = 4.
12 943 [Tmr Svc] EEPROM(main) hash check...
13 945 [Tmr Svc] OK
14 945 [Tmr Svc] EEPROM(mirror) hash check...
15 947 [Tmr Svc] OK
40 6920 [OTA] OTA demo version 0.9.3
```