AmebaD Amazon FreeRTOS Getting Started Guide

Realtek Semiconductor Corp.
No. 2, Innovation Road II, Hsinchu Science Park, Hsinchu 300, Taiwan
Tel.: +886-3-578-0211. Fax: +886-3-577-6047
www.realtek.com

COPYRIGHT

DISCLAIMER

Please Read Carefully:

Realtek Semiconductor Corp., (Realtek) reserves the right to make corrections, enhancements, improvements and other changes to its products and services. Buyers should obtain the latest relevant information before placing orders and should verify that such information is current and complete.

Reproduction of significant portions in Realtek data sheets is permissible only if reproduction is without alteration and is accompanied by all associated warranties, conditions, limitations, and notices. Realtek is not responsible or liable for such reproduced documentation. Information of third parties may be subject to additional restrictions.

Buyers and others who are developing systems that incorporate Realtek products (collectively, “Customers”) understand and agree that Customers remain responsible for using their independent analysis, evaluation and judgment in designing their applications and that Customers have full and exclusive responsibility to assure the safety of Customers' applications and compliance of their applications (and of all Realtek products used in or for Customers’ applications) with all applicable regulations, laws and other applicable requirements. Designer represents that, with respect to their applications, Customer has all the necessary expertise to create and implement safeguards that (1) anticipate dangerous consequences of failures, (2) monitor failures and their consequences, and (3) lessen the likelihood of failures that might cause harm and take appropriate actions. Customer agrees that prior to using or distributing any applications that include Realtek products, Customer will thoroughly test such applications and the functionality of such Realtek products as used in such applications.

Realtek’s provision of technical, application or other design advice, quality characterization, reliability data or other services or information, including, but not limited to, reference designs and materials relating to evaluation kits, (collectively, “Resources”) are intended to assist designers who are developing applications that incorporate Realtek products; by downloading, accessing or using Realtek’s Resources in any way, Customer (individually or, if Customer is acting on behalf of a company, Customer’s company) agrees to use any particular Realtek Resources solely for this purpose and subject to the terms of this Notice.

Realtek’s provision of Realtek Resources does not expand or otherwise alter Realtek’s applicable published warranties or warranty disclaimers for Realtek’s products, and no additional obligations or liabilities arise from Realtek providing such Realtek Resources. Realtek reserves the right to make corrections, enhancements, improvements and other changes to its Realtek Resources. Realtek has not conducted any testing other than that specifically described in the published documentation for a particular Realtek Resource.

Customer is authorized to use, copy and modify any individual Realtek Resource only in connection with the development of applications that include the Realtek product(s) identified in such Realtek Resource. No other license, express or implied, by estoppel or otherwise to any other Realtek intellectual property right, and no license to any technology or intellectual property right of Realtek or any third party is granted herein, including but not limited to any patent right, copyright, mask work right, or other intellectual property right relating to any combination, machine, or process in which Realtek products or services are used. Information regarding or referencing third-party products or services does not constitute a license to use such products or services, or a warranty or endorsement thereof. Use of Realtek Resources may require a license from a third party under the patents or other intellectual property of the third party, or a license from Realtek under the patents or other Realtek’s intellectual property.

Realtek’s Resources are provided “as is” and with all faults. Realtek disclaims all other warranties or representations, express or implied, regarding resources or use thereof, including but not limited to accuracy or completeness, title, any epidemic failure warranty and any implied warranties of merchantability, fitness for a particular purpose, and non-infringement of any third party intellectual property rights.

Realtek shall not be liable for and shall not defend or indemnify Customer against any claim, including but not limited to any infringement claim that related to or is based on any combination of products even if described in Realtek Resources or otherwise. In no event shall Realtek be liable for any actual, direct, special, collateral, indirect, punitive, incidental, consequential or exemplary damages in connection with or arising out of Realtek’s Resources or use thereof, and regardless of whether Realtek has been advised of the possibility of such damages. Realtek is not responsible for any failure to meet such industry standard requirements.

Where Realtek specifically promotes products as facilitating functional safety or as compliant with industry functional safety standards, such products are intended to help enable customers to design and create their own applications that meet applicable functional safety standards and requirements. Using products in an application does not by itself establish any safety features in the application. Customers must ensure compliance with safety-related requirements and standards applicable to their applications. Designer may not use any Realtek products in life-critical medical equipment unless authorized officers of the parties have executed a special contract specifically governing such use. Life-critical medical equipment is medical equipment where failure of such equipment would cause serious bodily injury or death. Such equipment includes, without limitation, all medical devices identified by the U.S.FDA as Class III devices and equivalent classifications outside the U.S.

Customers agree that it has the necessary expertise to select the product with the appropriate qualification designation for their applications and that proper product selection is at Customers' own risk. Customers are solely responsible for compliance with all legal and regulatory requirements in connection with such selection.

Customer will fully indemnify Realtek and its representatives against any damages, costs, losses, and/or liabilities arising out of Designer's non-compliance with the terms and provisions of this Notice.

TRADEMARKS

Realtek is a trademark of Realtek Semiconductor Corporation. Other names mentioned in this document are trademarks/registered trademarks of their respective owners.

USING THIS DOCUMENT

Though every effort has been made to ensure that this document is current and accurate, more information may have become available subsequent to the production of this guide.

1 AmebaD RTL8722DM Board

1.1 AmebaD Demo EVB

Ameba Demo board home page: https://www.amebaiot.com/amebad/

Ameba RTL8722DM Board (AMB 21)

Manual / Schematic / Layout

Buy it

CPU
- 32-bit Arm®Cortex®-M4, up to 200MHz
- 32-bit Arm®Cortex®-M0, up to 20MHz

Memory
- 512KB SRAM + 4MB PSRAM

Key Features
- Integrated 802.11a/n Wi-Fi SoC
- Trustzone-M Security
- Hardware SSL Engine
- Root Trust Secure Boot
- USB Host/Device
- SD Host
- BLE5.0
- Codec
- LCDC
- Key Matrix

Other Features
- 1 PCM interface
- 4 UART interface
- 1 I2S Interface
- 2 I2C interface
- 7 ADC
- 17 PWM
- Max 54 GPIO

1.2 PCB Layout Overview

The PCB layout of 2D and 3D are shown in Fig 1-1 and Fig 1-2.

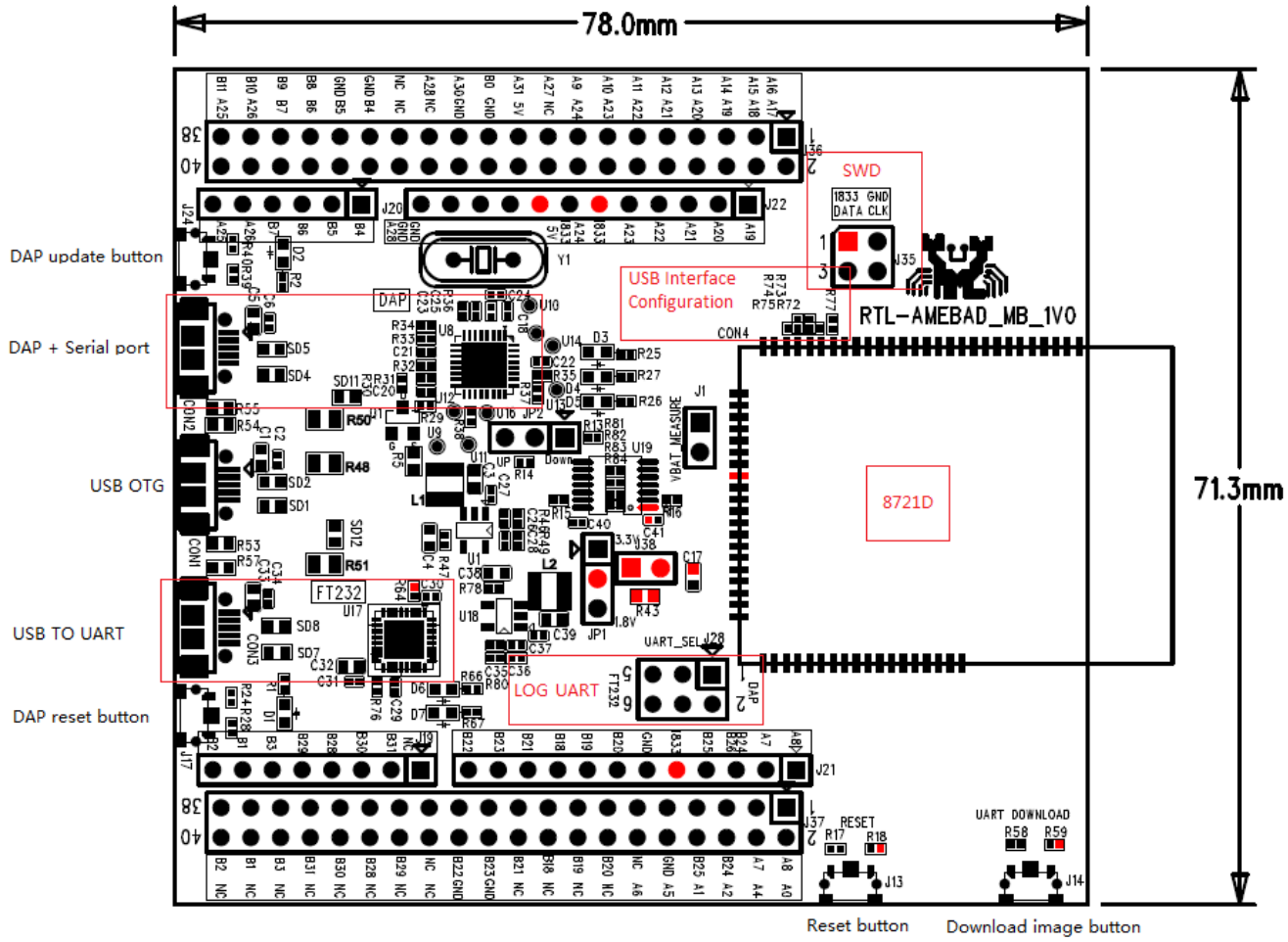


Fig 1-1 Demo board – PCB layout (2D)

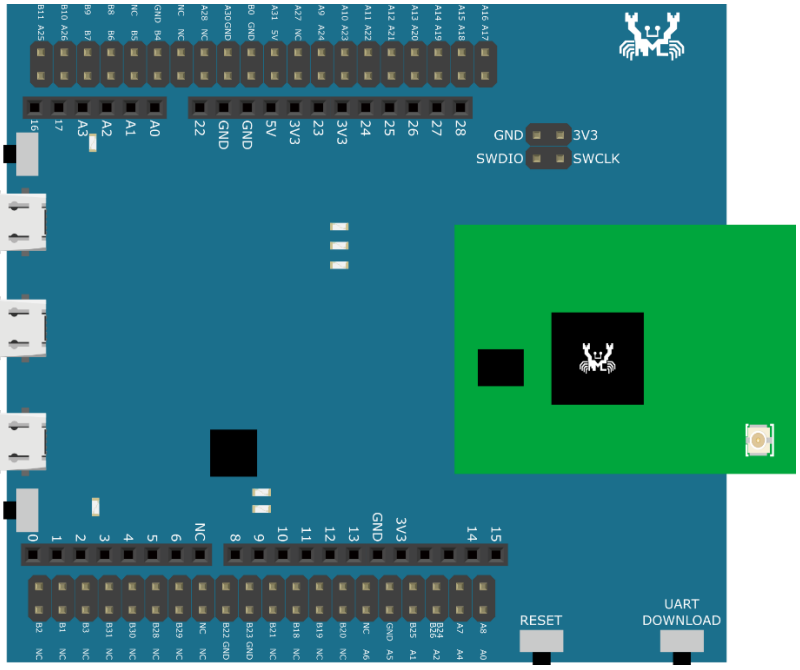


Fig 1-2 Demo board – PCB layout (3D)

1.3 Pin Out

The pin out board is shown in Fig 1-3.

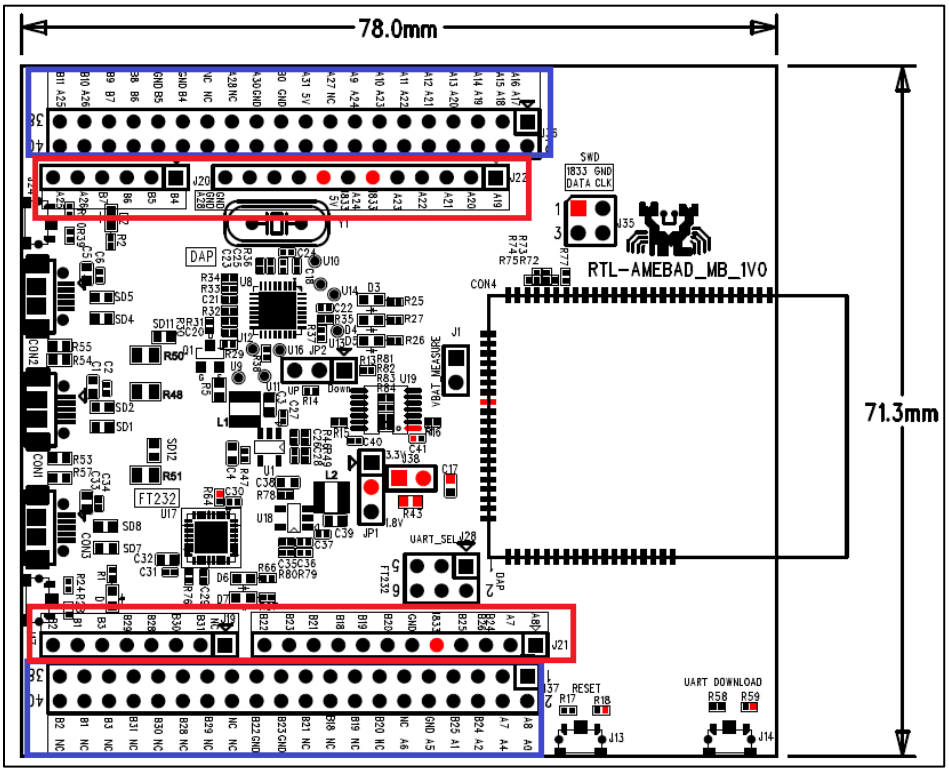


Fig 1-3 Demo board – pin out

There are four rows of pins on the board.

- The pins in the red box are used for Arduino REF.
- The pins in the blue box are all the GPIO pins.

1.4 DC Power Supply

The 3.3V/1.8V power supply board is shown in Fig 1-4.

- Jump JP1 is used to select 3.3V or 1.8V power supply
- Jump J38 is for current test. You can test the current power after taking off the R43.

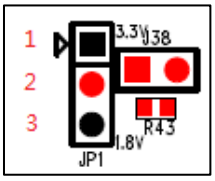


Fig 1-4 Demo board – 3.3V/1.8V power supply

When you select power supply, refer to Table 1-1.

Table 1-1 3.3V/1.8V power supply selection

Power Supply Select	JP1
3.3V	1-2 connected
1.8V	2-3 connected

1.5 USB Interface Configuration

The USB interface configuration board is shown in Fig 1-5 and Fig 1-6.

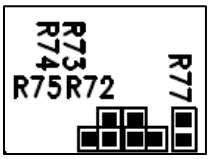


Fig 1-5 Mother board – USB interface configuration

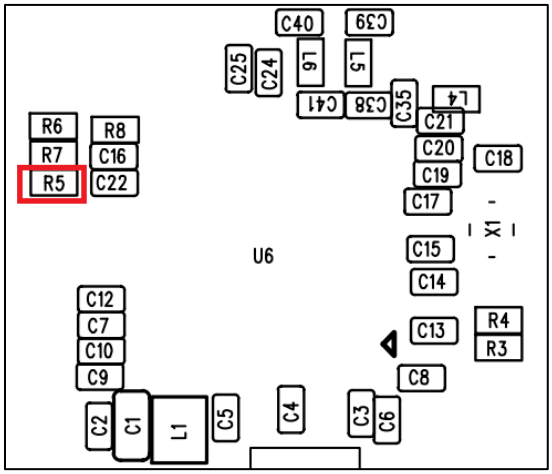


Fig 1-6 Module board – USB interface configuration

For normal GPIO usage by default, R72/R75/R77 on mother board will part on with 0 Ohm resistors, R5 on module board needs to take off. For USB usage, you need to take off R77, part on R73&R74 with 0 Ohm resistors on mother board and part on R5 on module board with a 12K Ohm 1% precision resistor.

1.6 LOGUART

The LOGUART board is shown in Fig 1-7. When you select LOGUART, please refer to Table 1-2.

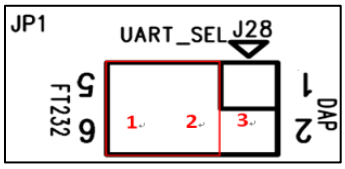


Fig 1-7 Demo board – LOGUART

Table 1-2 LOGUART selection

LOGUART Select	JP1
FT232	1-2 connected
DAP	2-3 connected

1.7 SWD

The SWD board is shown in Fig 1-8.

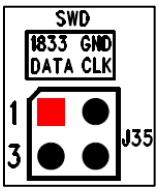


Fig 1-8 Demo board – SWD

Note: For 1V0 board, there is an issue, you should use CLK as DATA, and use DATA as CLK.

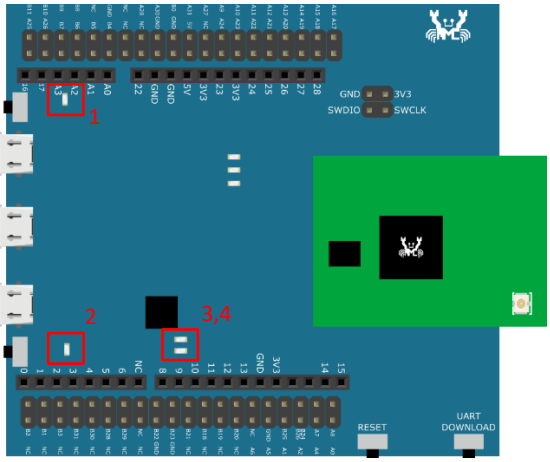
1.8 VBAT ADC

The VBAT ADC board is shown in Fig 1-9. J1 is used to test VBAT ADC.

Fig 1-9 Demo board – VBAT ADC

1.9 LED State

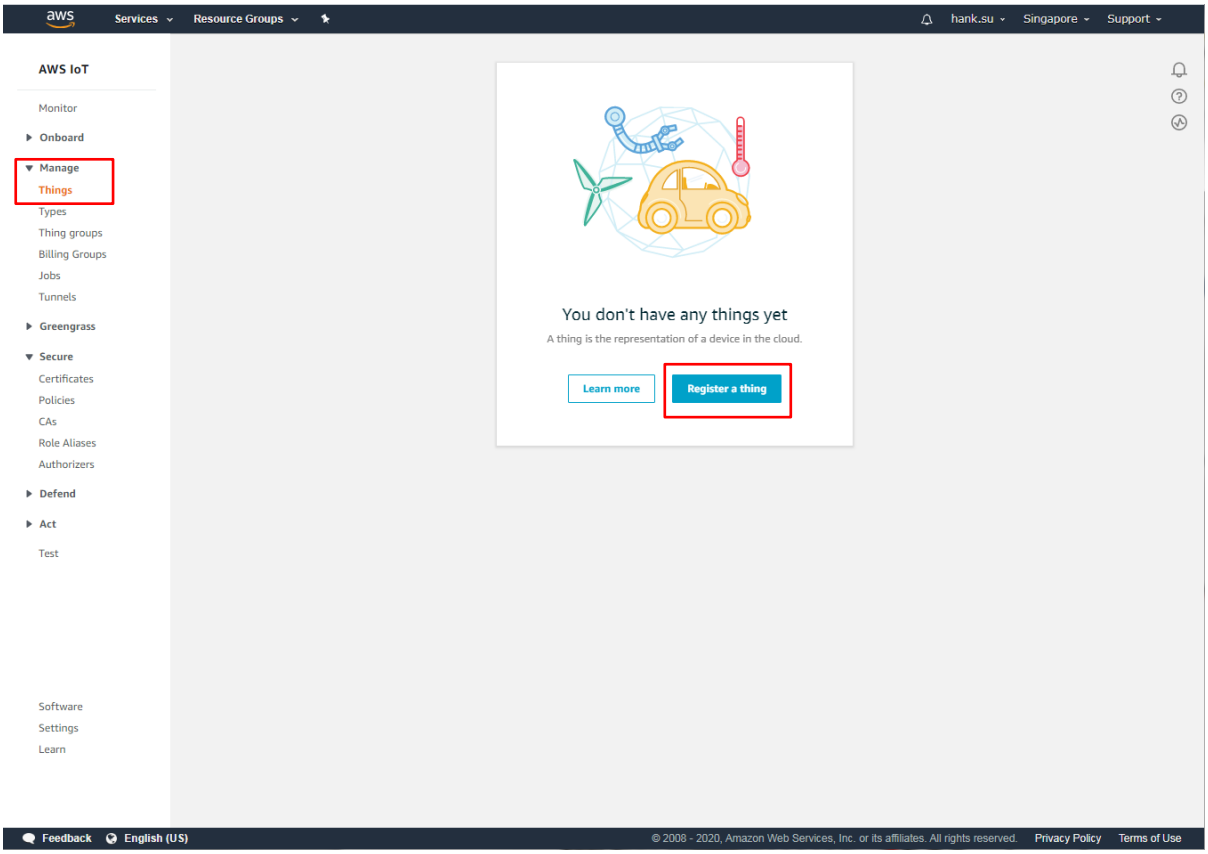
There are four LED on the AmebaD EVB. LED1 and LED2 lights steady green when device have power. LED3 and LED4 go with log uart, they flash red and green when uart communicating.

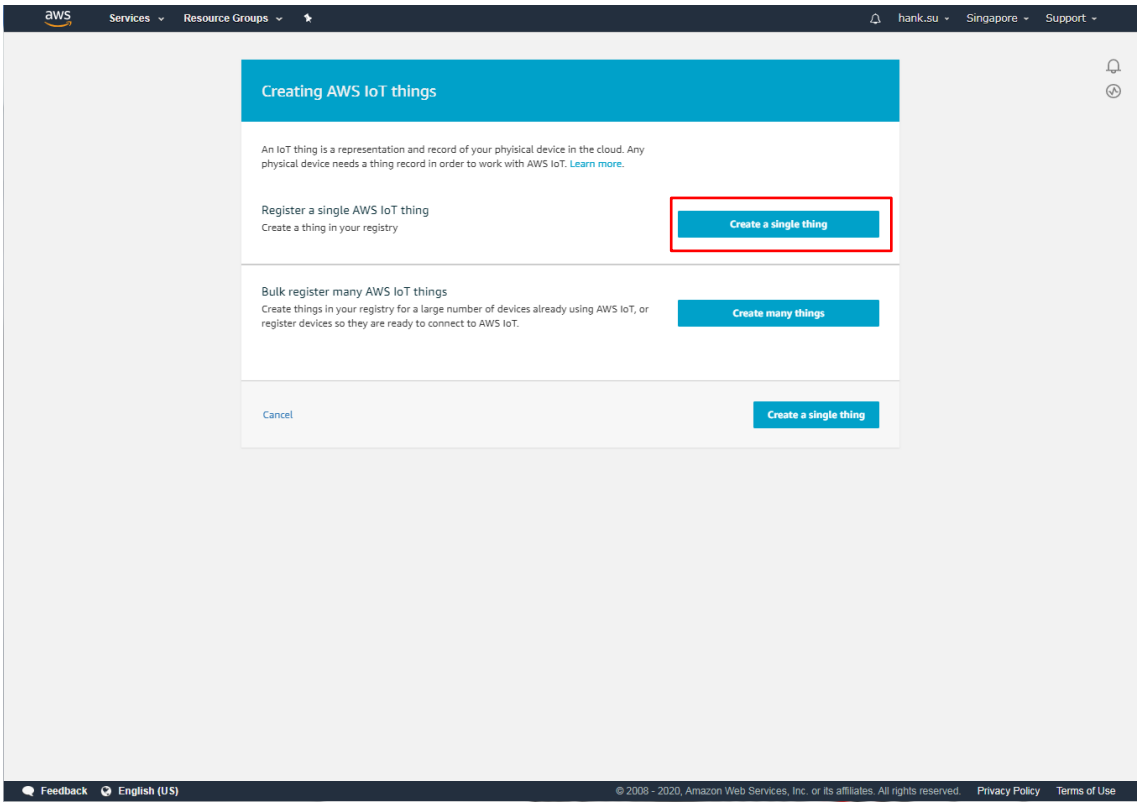


2 Configure AWS IoT Core

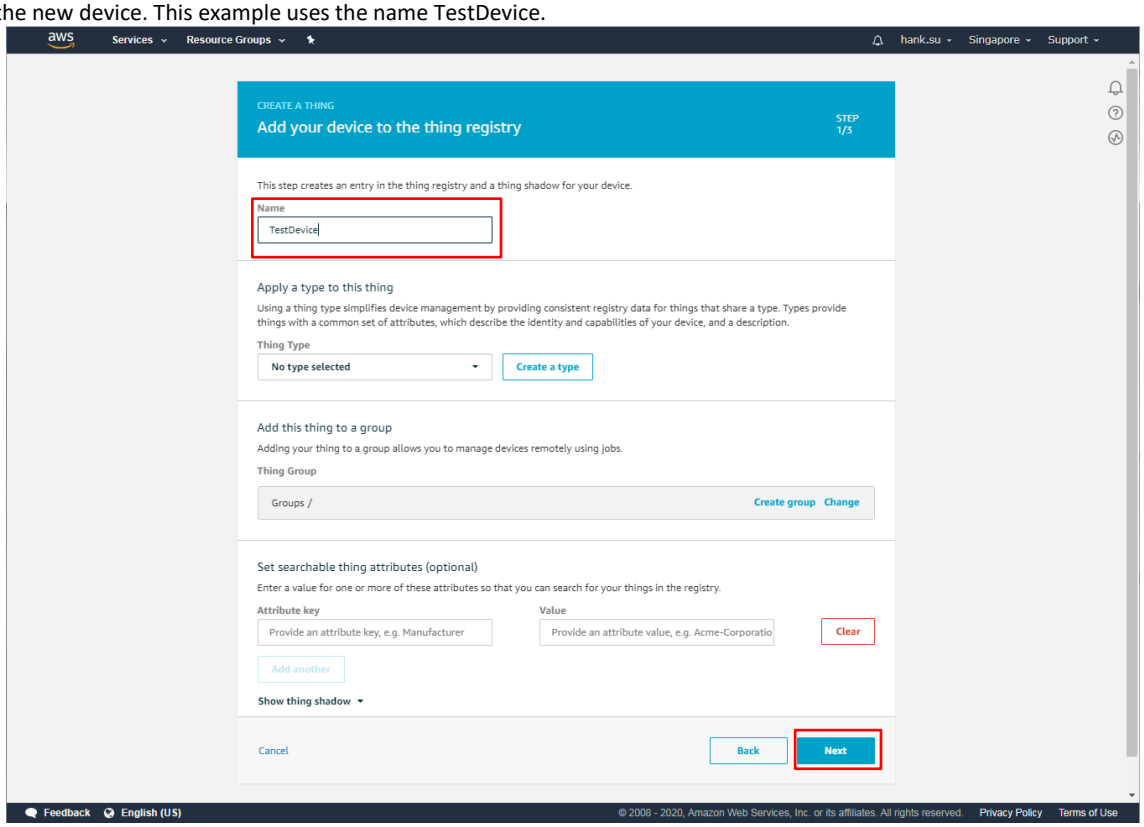
2.1 Create a New Device

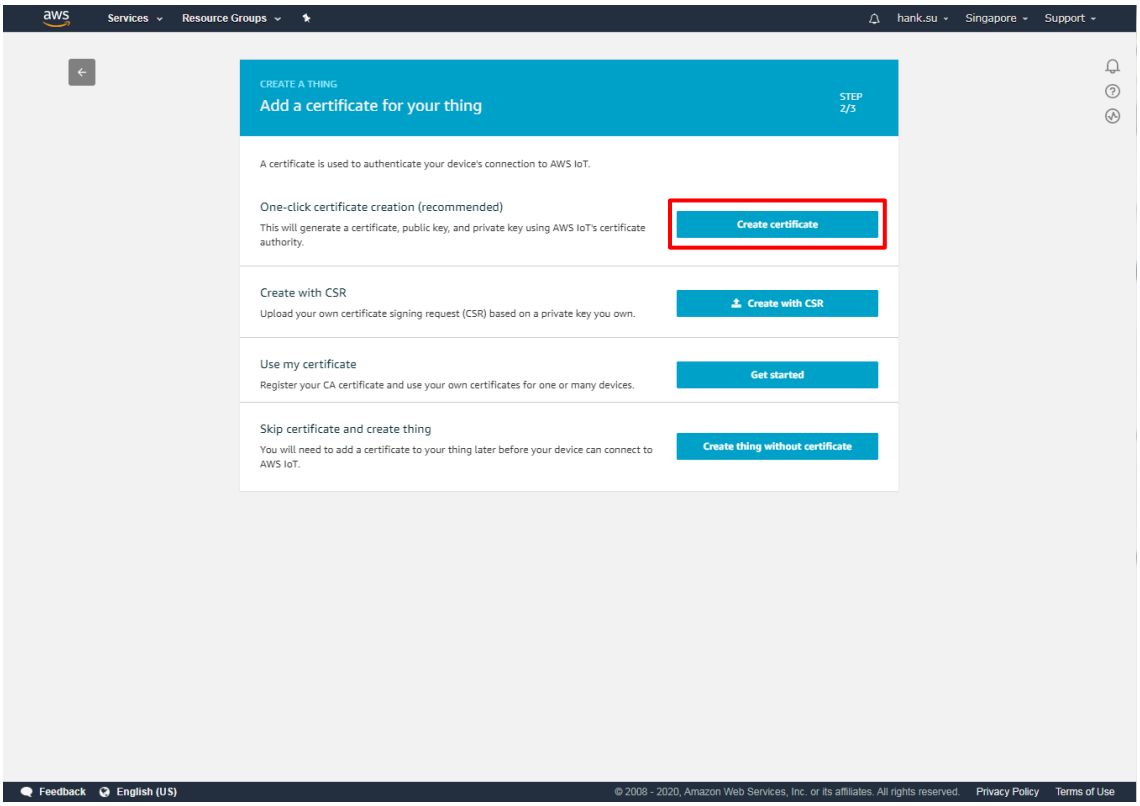
To create a new device, navigate to Manage -> Things in the left-hand navigation menu. Then click “Register a thing”.



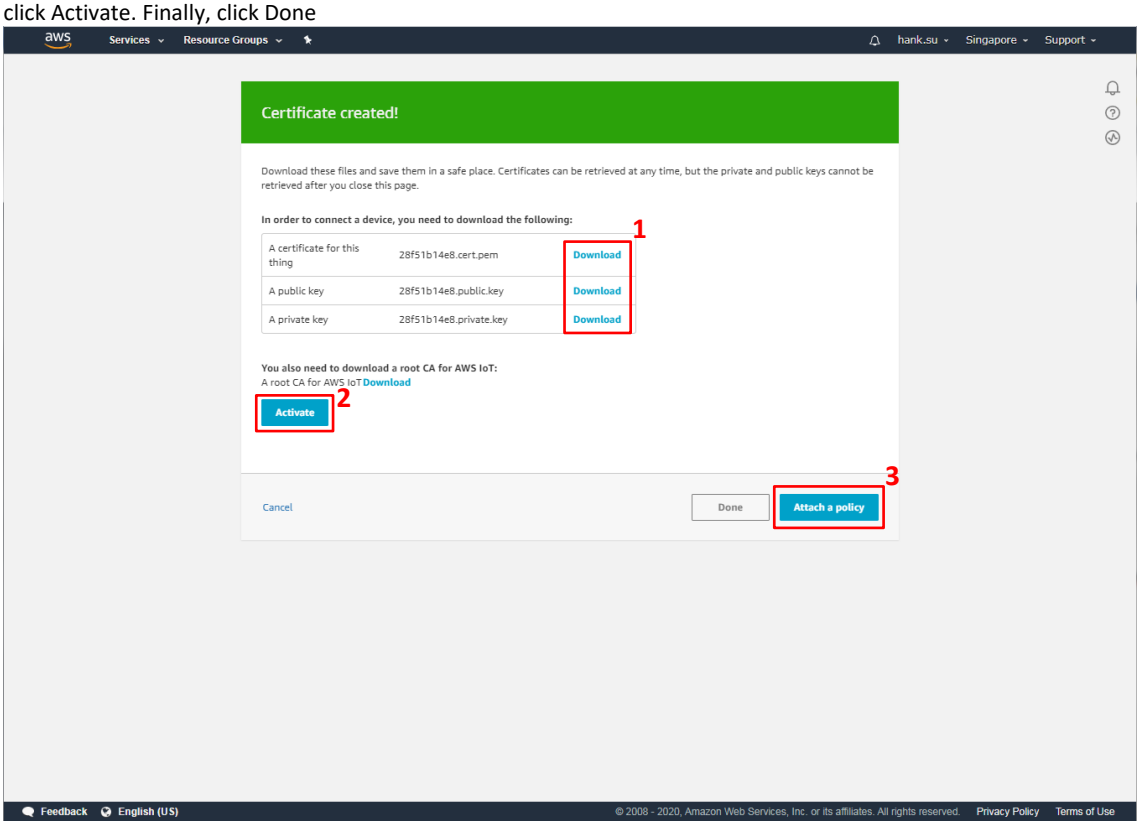


Then, name the new device. This example uses the name TestDevice.



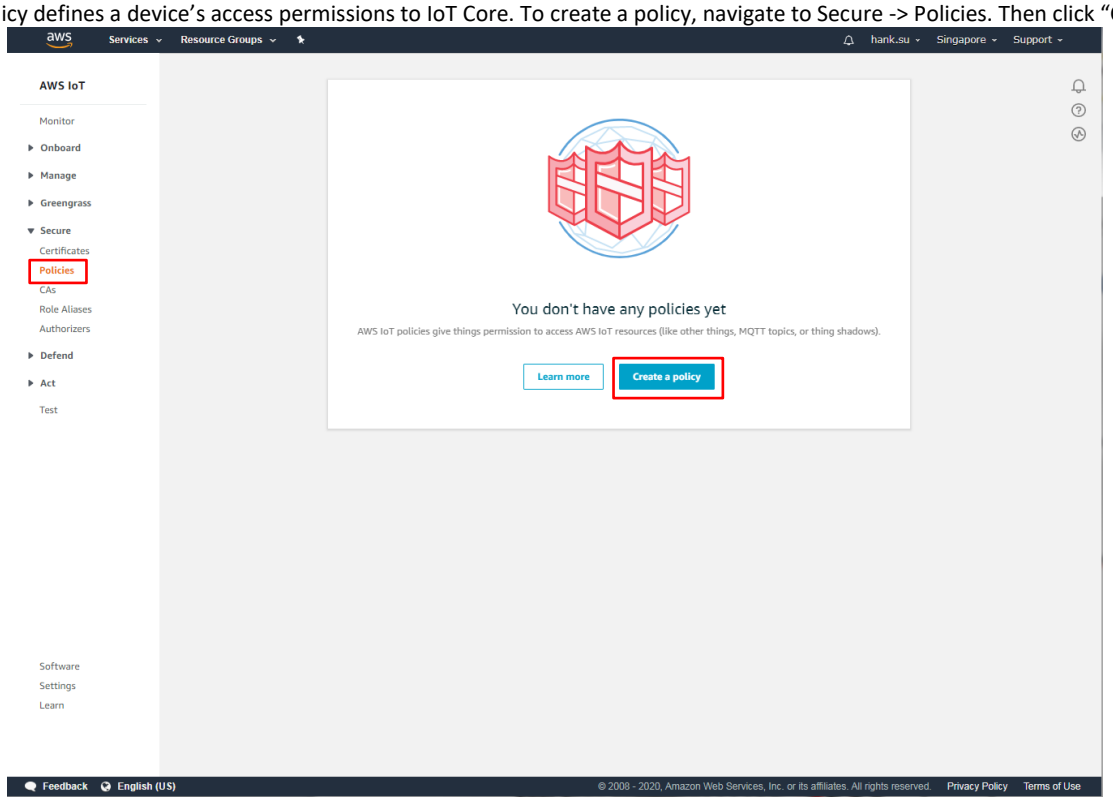


Download the certificate, public key, and private key for the device by clicking Download. Once all the certificate and keys have been downloaded, click Activate. Finally, click Done

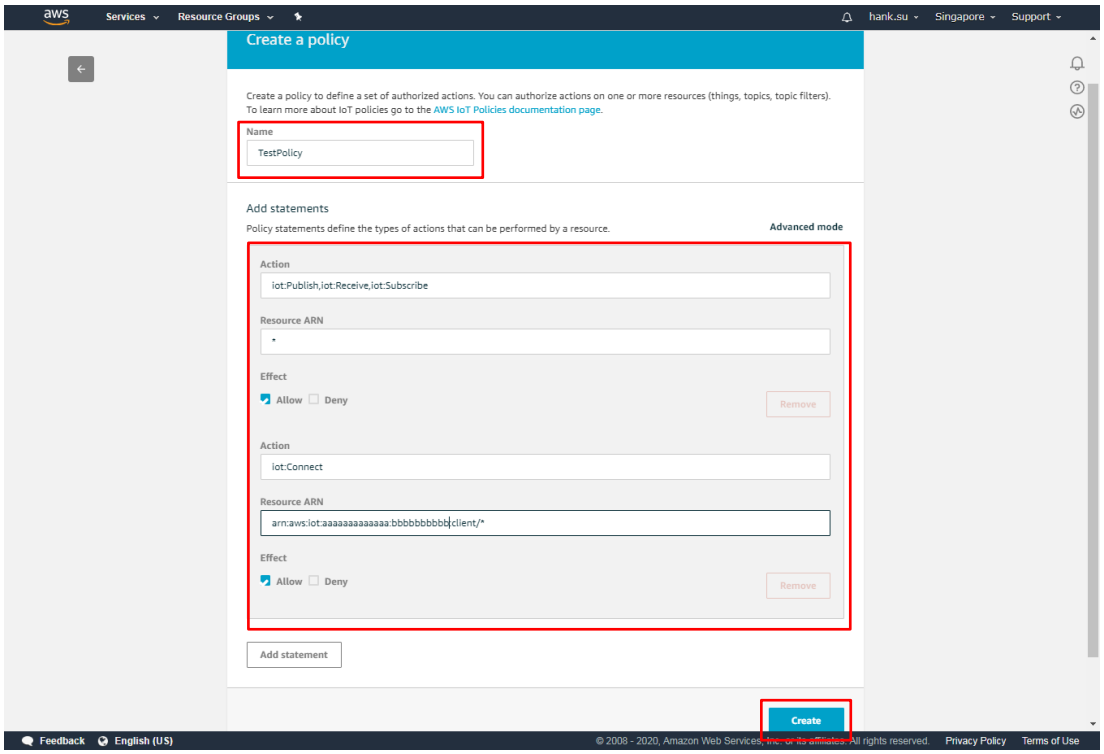


2.2 Create a policy

A policy defines a device's access permissions to IoT Core. To create a policy, navigate to Secure -> Policies. Then click "Create a policy"

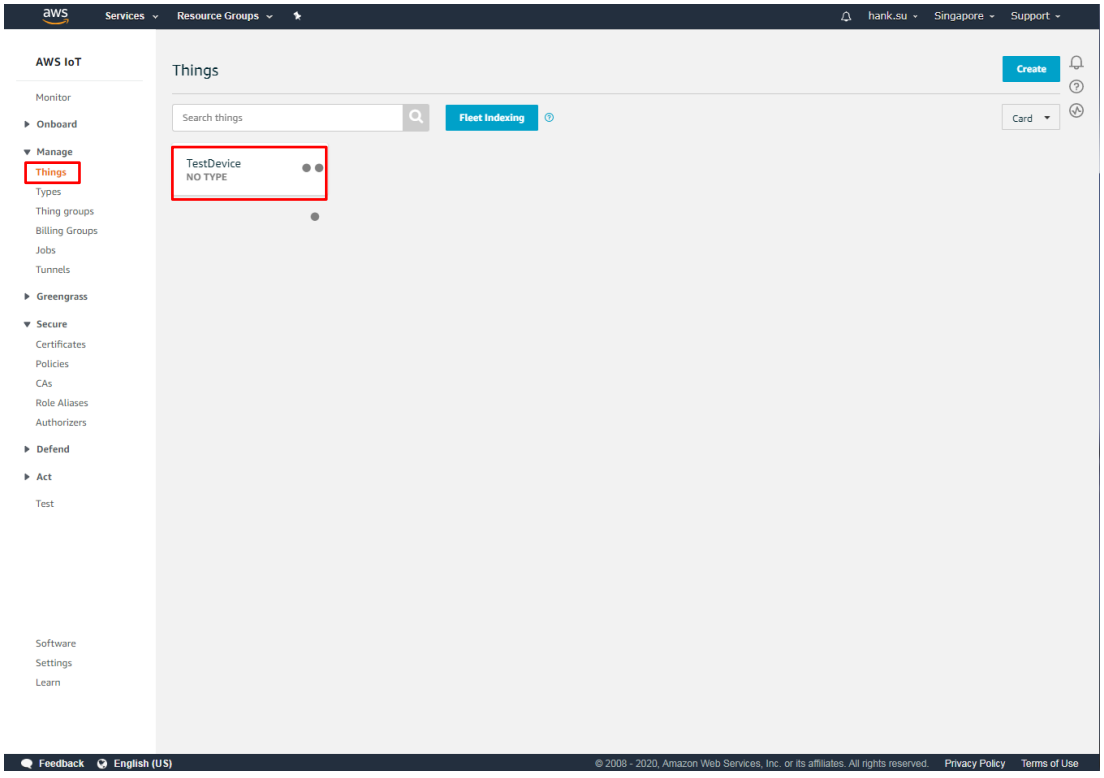


NOTE – this policy grants unrestricted access for all iot operations, and is to be used only in a development environment. For non-dev environments, all devices in your fleet must have credentials with privileges that authorize intended actions only, which include (but not limited to) AWS IoT MQTT actions such as publishing messages or subscribing to topics with specific scope and context. The specific permission policies can vary for your use cases. Identify the permission policies that best meet your business and security requirements.
For sample policies, refer to https://docs.aws.amazon.com/iot/latest/developerguide/example-iot-policies.html.
Also refer to https://docs.aws.amazon.com/iot/latest/developerguide/security-best-practices.html

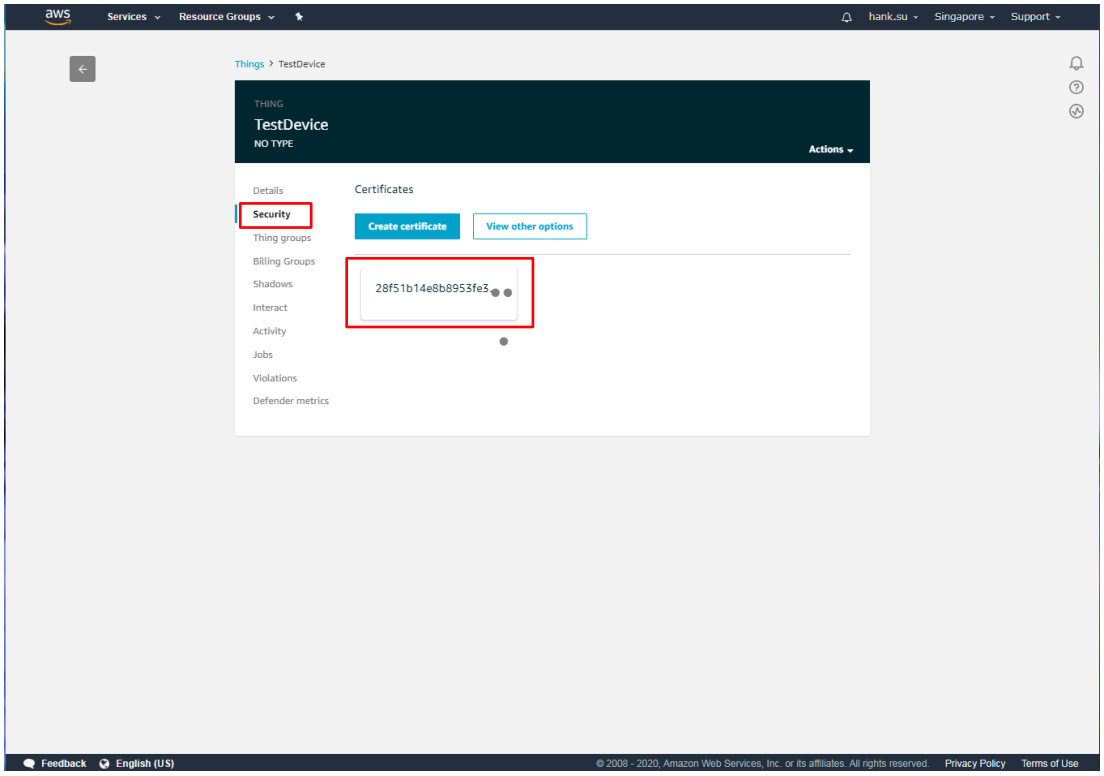


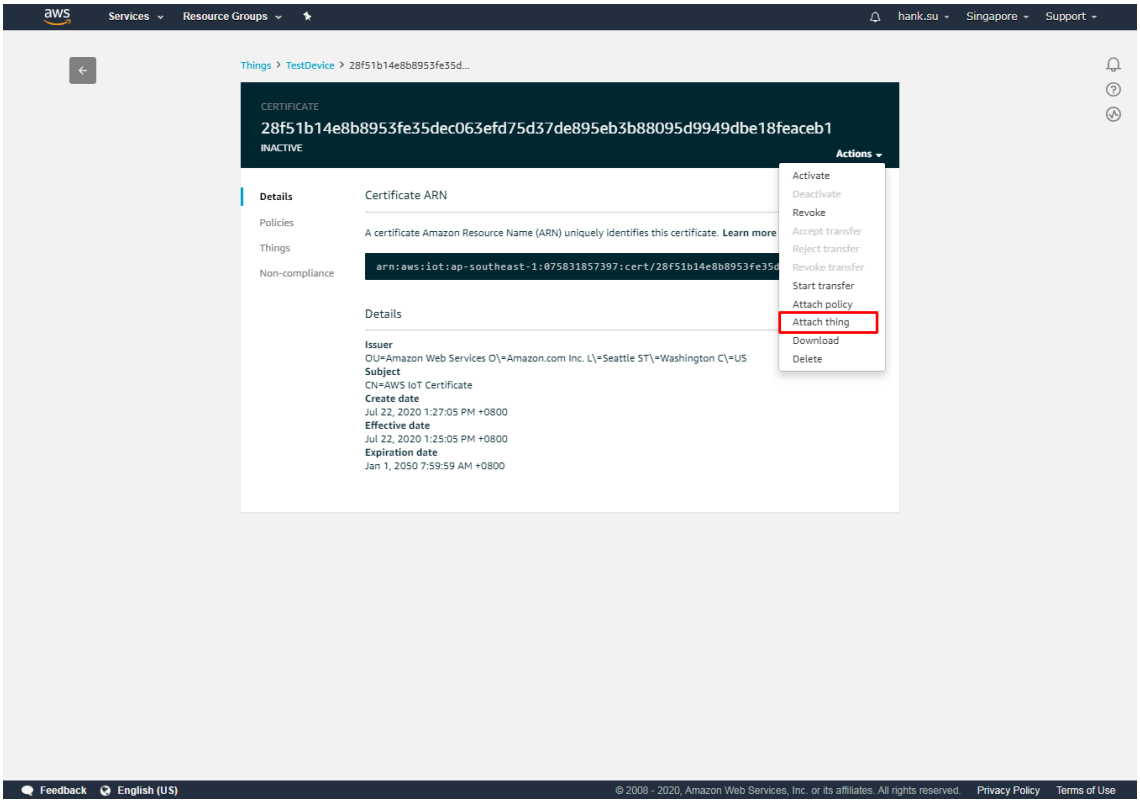
2.3 Attach Thing

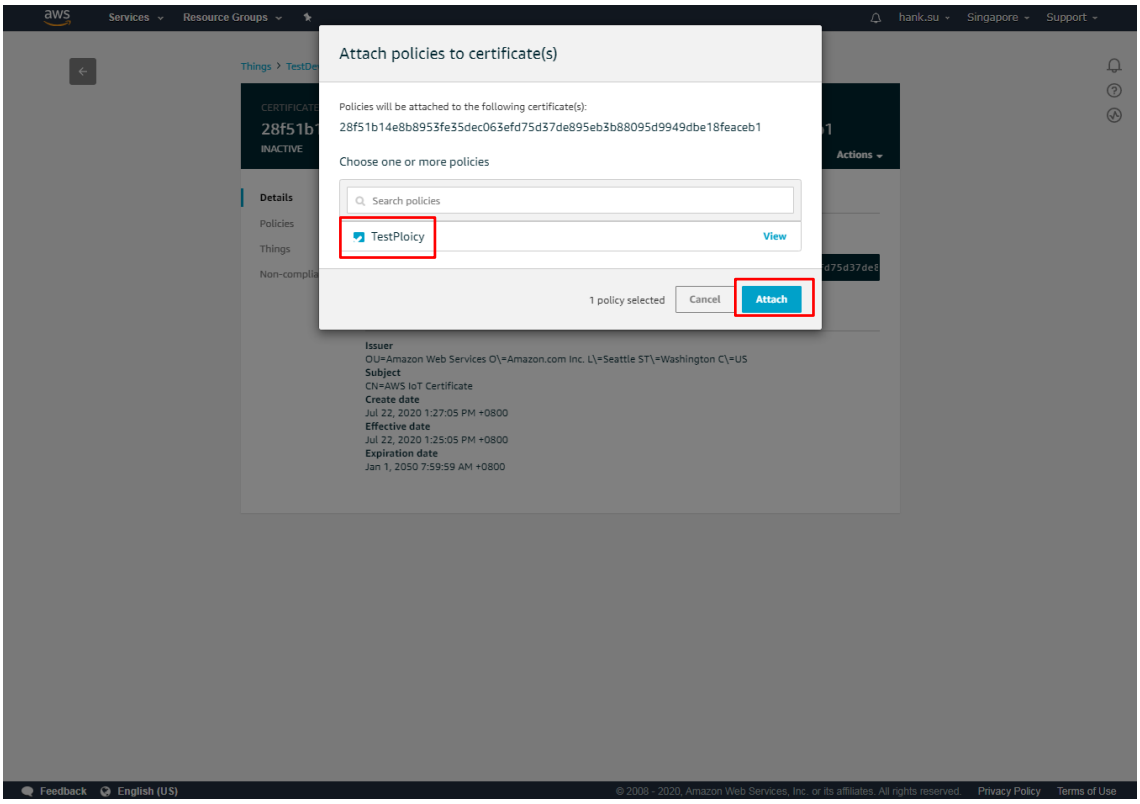
The last step to configuring the device is attaching a policy. To attach a policy to new device, navigate to Manage -> Things. Then click on the device which was created.



Click Security, then click the certificate create in previous step.



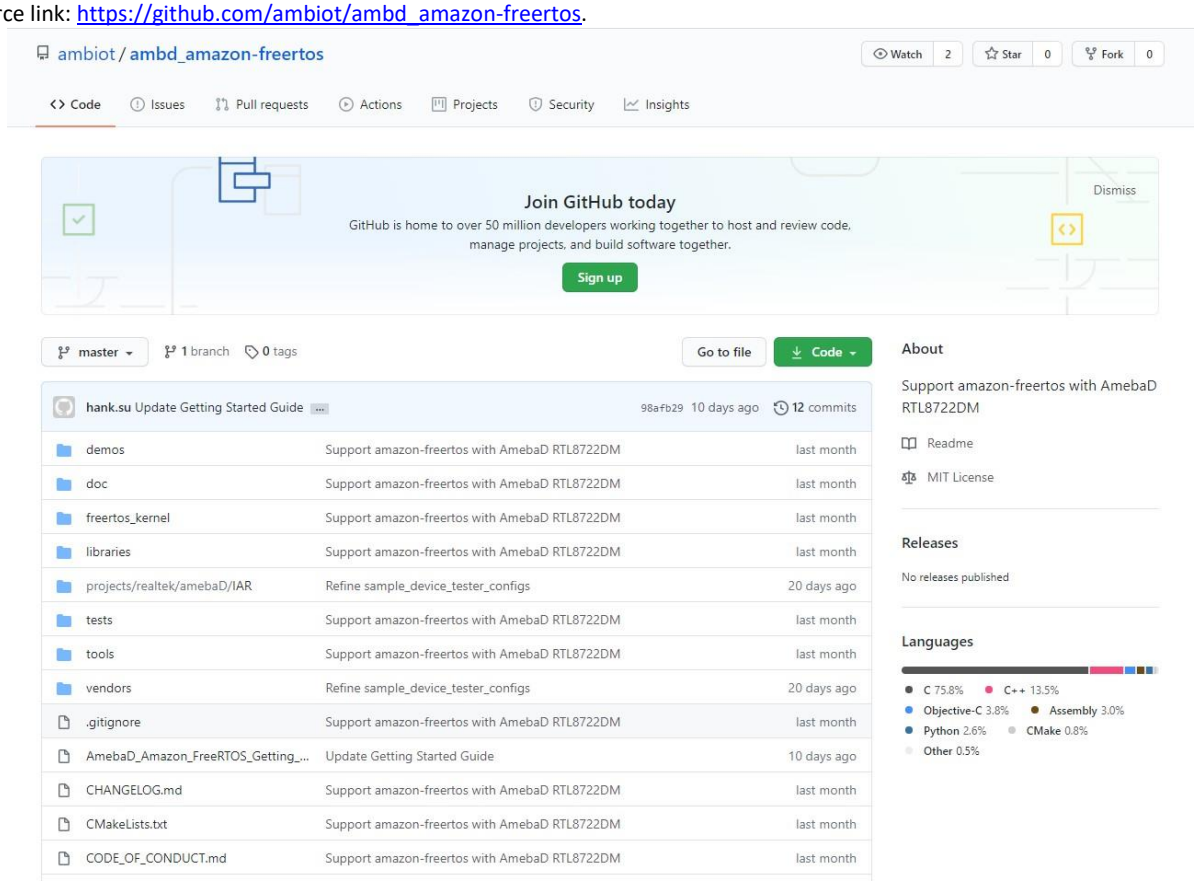




3 Configure AmebaD Amazon FreeRTOS

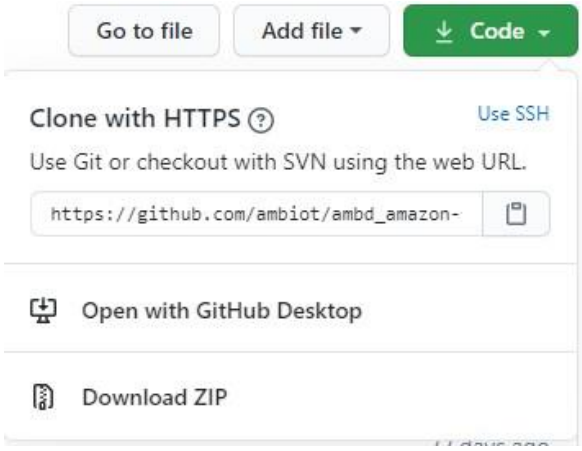
3.1 Download Source Code from github

Open source link: https://github.com/ambiot/ambd_amazon-freertos.



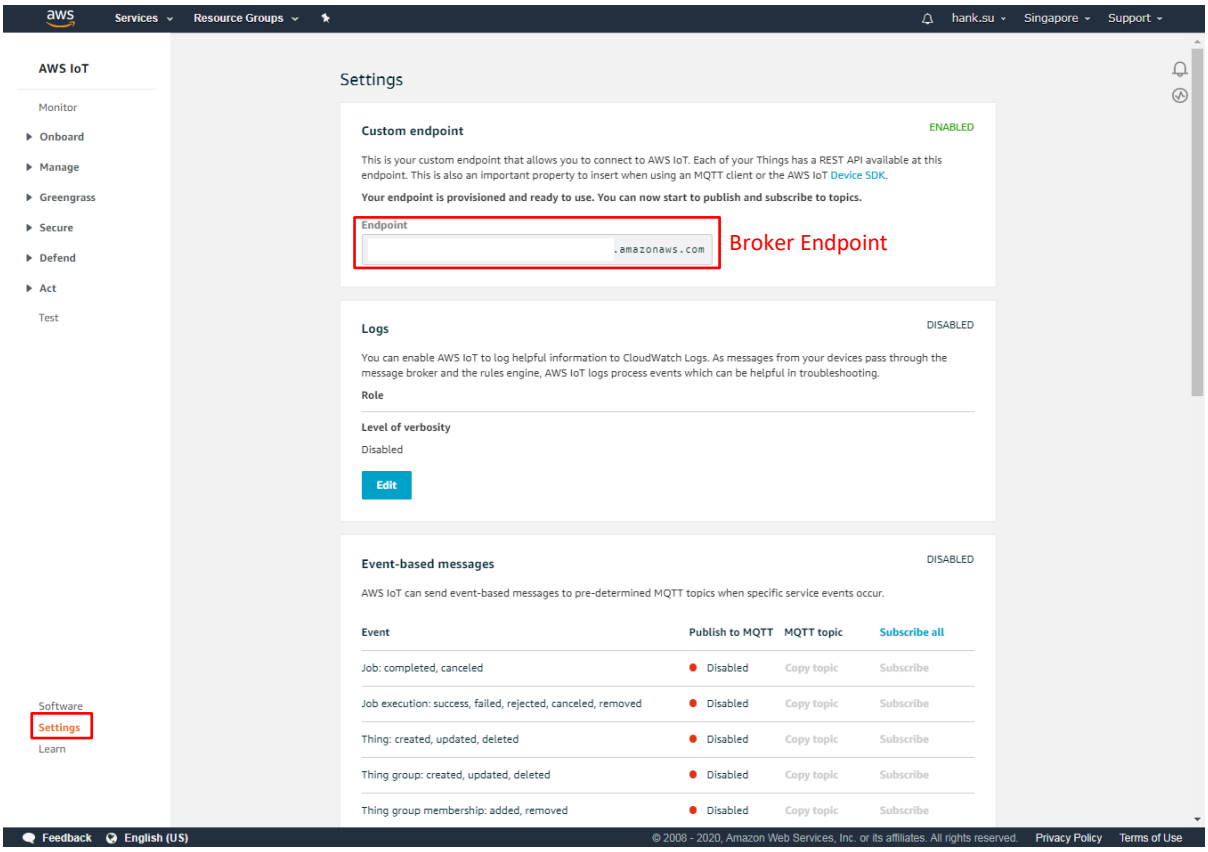
3.1.1 Cloning a repository by Download ZIP

1. On GitHub, navigate to the main page of the repository.
2. Above the list of files, click **Code**.
3. Click **Donwload ZIP** to get source code.

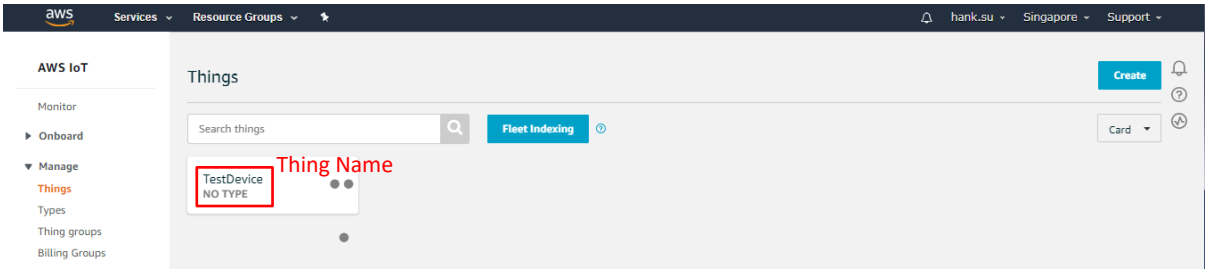


For more information, please refer "Cloning a repository from GitHub to GitHub Desktop."

3.2 Get Broker Endpoint by AWS IoT Core



3.3 Get Thing Name



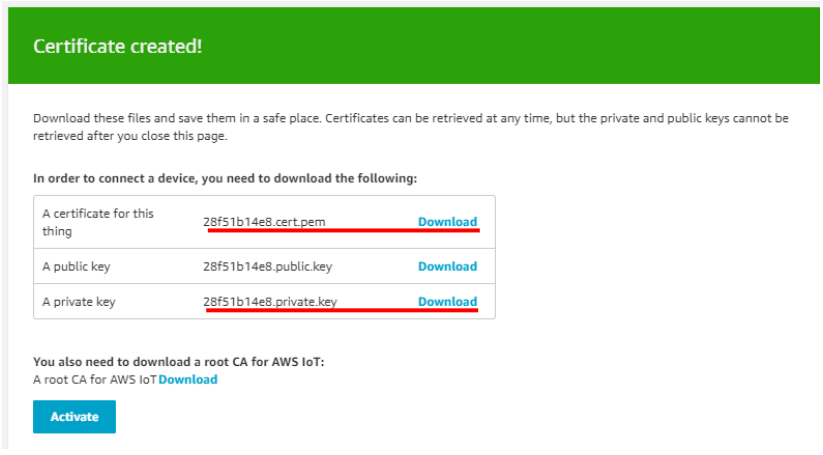
3.4 Setup IoT Core Information with AmebaD Amazon FreeRTOS

Setup BROKER_ENDPOINT, THING_NAME, WIFI_SSID, PASSWORD in “ambd_amazon-freertos/blob/master/demos/include/aws_clientcredential.h”

```
#define clientcredentialMQTT_BROKER_ENDPOINT          "xxxxxxxxxxxxxx.amazonaws.com"

/*
 * @brief Host name.
 *
 * @todo Set this to the unique name of your IoT Thing.
 */
#define clientcredentialIOT_THING_NAME                "TestDevice"

/*
 * @brief Port number the MQTT broker is using.
 */
#define clientcredentialMQTT_BROKER_PORT              8883

/*
 * @brief Port number the Green Grass Discovery use for JSON retrieval from cloud is using.
 */
#define clientcredentialGREENGRASS_DISCOVERY_PORT     8443

/*
 * @brief Wi-Fi network to join.
 *
 * @todo If you are using Wi-Fi, set this to your network name.
 */
#define clientcredentialWIFI_SSID                     "TestAP"

/*
 * @brief Password needed to join Wi-Fi network.
 * @todo If you are using WPA, set this to your network password.
 */
#define clientcredentialWIFI_PASSWORD                 "password"

/*
 * @brief Wi-Fi network security type.
 *
 * @see WIFISecurity_t.
 *
 * @note Possible values are eWiFiSecurityOpen, eWiFiSecurityWEP, eWiFiSecurityWPA,
 * eWiFiSecurityWPA2 (depending on the support of your device Wi-Fi radio).
 */
#define clientcredentialWIFI_SECURITY                 eWiFiSecurityWPA2

#endif /* ifndef __AWS_CLIENTCREDENTIAL__H__ */
```

3.4.1 Setup Thing’s Private Key and Certificate

Filled keyCLIENT_CERTIFICATE_PEM and keyCLIENT_PRIVATE_KEY_PEM in “ambd_amazon-freertos/blob/master/demos/include/aws_clientcredential_keys.h” by xxxxxxxx-certifiacte.pem and xxxxxxxx-private.pem.key.



It can done by ambd_amazon-freertos/tools/certificate_configuration/CertificateConfigurator.html

Final aws_clientcredential_keys.h overview.

```
/*
 * @brief PEM-encoded client certificate.
 *
 * @todo If you are running one of the FreeRTOS demo projects, set this
 * to the certificate that will be used for TLS client authentication.
 *
 * @note Must include the PEM header and footer:
 * "-----BEGIN CERTIFICATE-----\n"\
 * "...base64 data...\n"\
 * "-----END CERTIFICATE-----\n"
 */
#define keyCLIENT_CERTIFICATE_PEM \
"-----BEGIN CERTIFICATE-----\n"\
"MIIDWjCCAkKgAwIBAgIVAIDLSSoG+EARSbBprT4Im1uu8j2vMA0GCSqGSIb3DQEB\n"\
                                                                \n"\
"                                                               \n"\
"                                                               \n"\
"                                                               \n"\
"                                                               \n"\
"                                                               \n"\
"                                                               \n"\
"                                                               \n"\
"                                                               \n"\
"                                                               \n"\
"                                                               \n"\
"                                                               \n"\
"                                                               \n"\
"k5+NsBroU/YdvOUmzKn6XfI4nX4hLQJ2TbhAT8aq1ounGk6ZGqCbxt4mg5bB0w==\n"\
"-----END CERTIFICATE-----"
```

```
/*
 * @brief PEM-encoded client private key.
 *
 * @todo If you are running one of the FreeRTOS demo projects, set this
 * to the private key that will be used for TLS client authentication.
 *
 * @note Must include the PEM header and footer:
 * "-----BEGIN RSA PRIVATE KEY-----\n"\
 * "...base64 data...\n"\
 * "-----END RSA PRIVATE KEY-----\n"
 */
#define keyCLIENT_PRIVATE_KEY_PEM \
"-----BEGIN RSA PRIVATE KEY-----\n"\
"MIIEpAIBAAKCAQEAwop96WNucGebARFjD8O+CLsqcBNn/AHyhEcozLZC8qoECUOn\n"\
"                                                               \n"\
"                                                               \n"\
"                                                               \n"\
"                                                               \n"\
"                                                               \n"\
"                                                               \n"\
"                                                               \n"\
"                                                               \n"\
"                                                               \n"\
"                                                               \n"\
"                                                               \n"\
"                                                               \n"\
"                                                               \n"\
"                                                               \n"\
"                                                               \n"\
"                                                               \n"\
"                                                               \n"\
"pOWEuLUuz2FAv1noAbN/6OQ8H/PT0AFJT/ghA04GnIUF0kjSzY60ehS2mVp6neP+\n"\
"AZjzZ6QJYlb5/PFz9oES448kpyaAoS2ke86+R4r4YOMBK+I5RVbfSQ==\n"\
"-----END RSA PRIVATE KEY-----\n"
```

3.4.2 Enable FreeRTOS demo on AmebaD

Find platform_opts.h in ambd_amazon-freertos\vendors\realtek\boards\amebaD\aws_demos\config_files and enable **CONFIG_EXAMPLE_AMAZON_FREERTOS**

```
/* For Amazon FreeRTOS SDK example */
#define CONFIG_EXAMPLE_AMAZON_FREERTOS   1
```

Fine aws_demo_config.h in ambd_amazon-freertos\vendors\realtek\boards\amebaD\aws_demos\config_files and add **CONFIG_MQTT_DEMO_ENABLED**

```
/* To run a particular demo you need to define one of these.
 * Only one demo can be configured at a time
 *
 *          CONFIG_MQTT_DEMO_ENABLED
 *          CONFIG_SHADOW_DEMO_ENABLED
 *          CONFIG_OTA_UPDATE_DEMO_ENABLED
 *
 *  These defines are used in iot_demo_runner.h for demo selection */

#define CONFIG_MQTT_DEMO_ENABLED
```

Now you can start to compile AmebaD Amazon FreeRTOS

4 Compile AmebaD Amazon FreeRTOS

4.1 IAR Build Environment Setup

Currently the amazon-freertos of AmebaD supported by the IAR Embedded workbench ver.8.30.1. For windows operating system only. This chapter illustrates how to setup IAR development environment for Realtek Ameba-D SDK, including building projects and downloading images.

4.2 Pre-Requisite

- Required source code. (https://github.com/ambiot/ambd_amazon-freertos)
- AmebaD Demo board
- Realtek Image Tool
- IAR Embedded Workbench ver.8.30.1

IAR provides an IDE environment for code building, downloading, and debugging. Check “IAR Embedded Workbench” on http://www.iar.com/, and a trail version is available for 30 days.

Note: To support ARMv8-M with Security Extension (Ameba-D HS CPU, also called KM4), IAR version must be 8.30 or higher.

4.3 How to Use IAR SDK

4.3.1 IAR Project Introduction

Because Ameba-D is a dual-core CPU platform, two workspaces provided to build for each core in **ambd_amazon-freertos/projects/realtek/amebaD/IAR/aws_demos**

- Project_lp_release.eww (KM0 workspace) contains the following projects:
  - km0_bootloader
  - km0_application
- Project_hp_release.eww (KM4 workspace) contains the following projects:
  - km4_bootloader
  - km4_application

4.3.2 IAR Build

When building SDK for the first time, you should build both KM0 project and KM4 project. Other times, you only need to rebuild the modified project.

4.3.2.1 Building KM0 Project

The following steps show how to build KM0 project:

(1) Open **ambd_amazon-freertos\projects\realtek\amebaD\IAR\aws_demos\Project_lp_release.eww**.

(2) Make sure km0_bootloader and km0_application are in Workspace. Click **Project** > **Options**, **General Options** > **Target** > **Processor Variant** > **Core**, verify the CPU configurations according to Fig 4-1

(3) Right click the project and choose “Rebuild All”, as Fig 4-2 shows. The km0_bootloader and km0_application should compile in order.

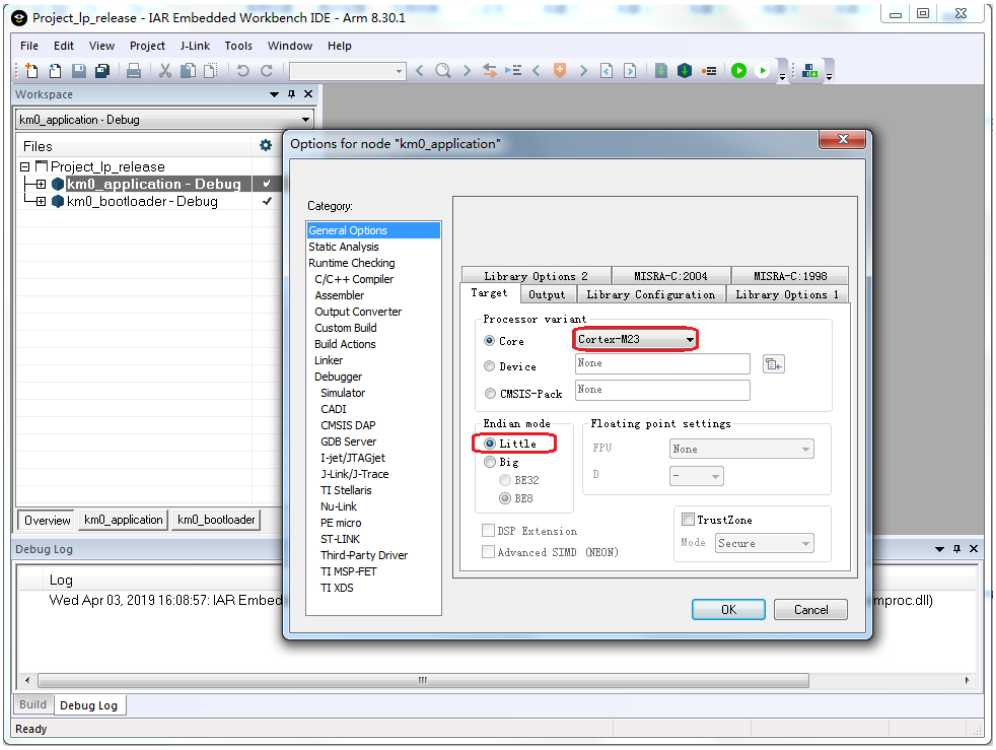


Fig 4-1 KM0 processor options

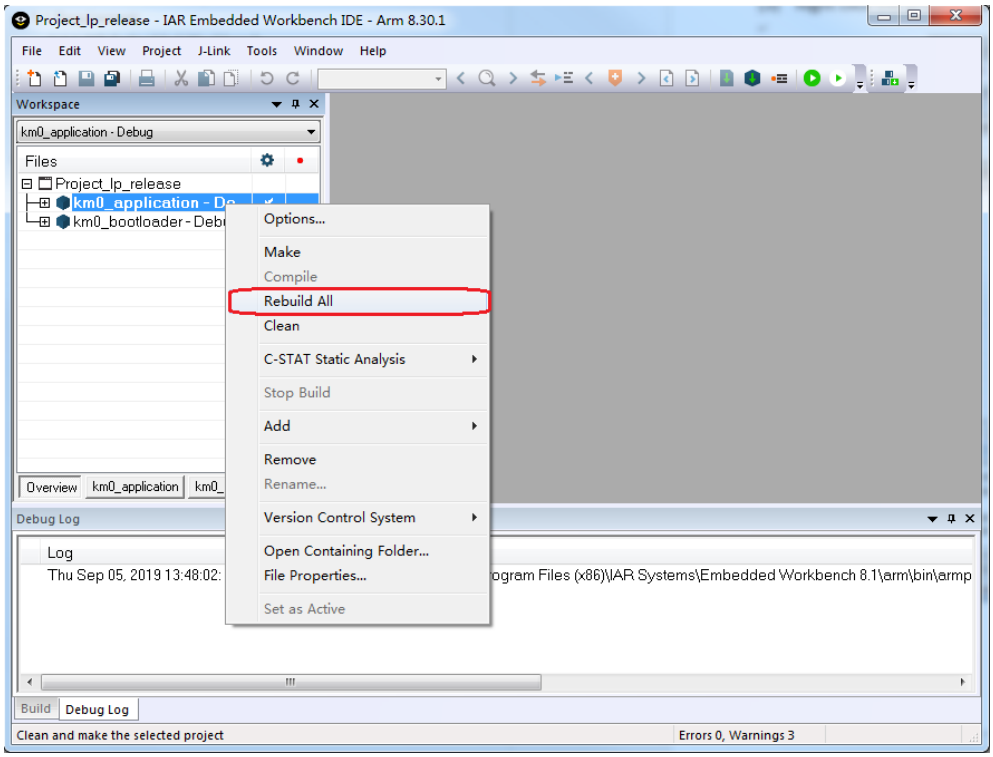
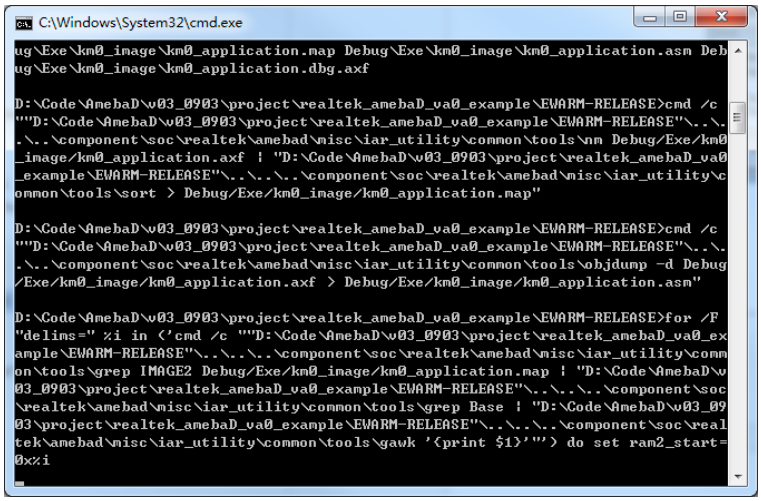


Fig 4-2 Building KM0 project

Note: After building each project, IAR will pop up a command prompt window to execute post-build action to generate images from executable files. This may takes several seconds. Do not stop it while it is in progress. After post-build action is completed, the window would disappear automatically.



(4) After compile, the images km0_boot_all.bin and km0_image2_all.bin can be find in **ambd_amazon-freertos\projects\realtek\amebaD\IAR\aws_demos\Debug\Exe\km0_image**.

4.3.2.2 Building KM4 Project

The following steps show how to build KM4 project:

(1) Open **ambd_amazon-freertos\projects\realtek\amebaD\IAR\aws_demos\Project_hp_release.eww**.

(2) Refer to 4.3.1 and choose the build configurations for each project according to your application.

(3) Click **Project** > **Options**, **General Options** > **Target** > **Processor Variant** > **Core**, verify the CPU configurations according to Fig 4-3.

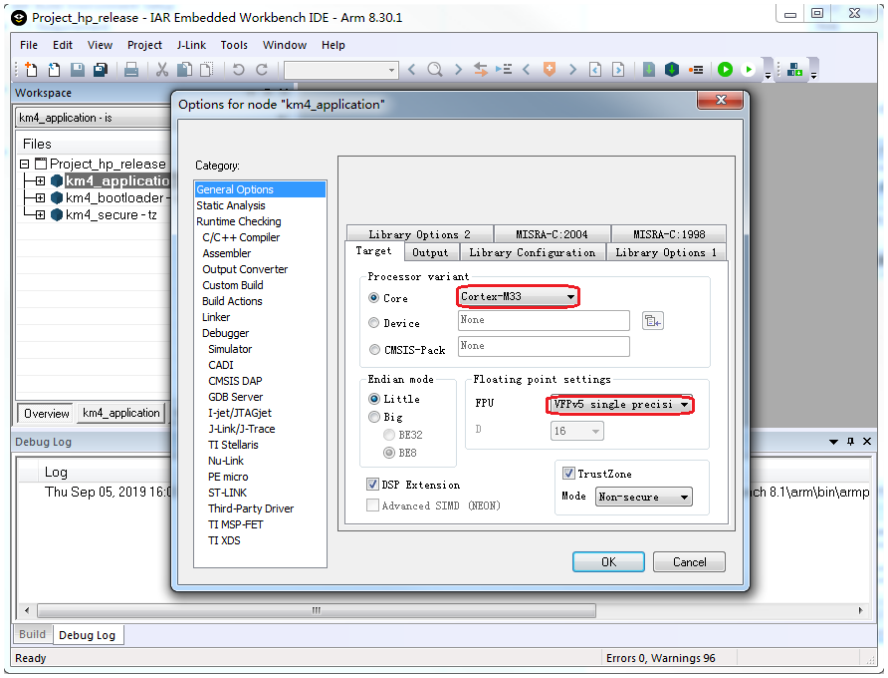


Fig 4-3 KM4 processor options

(4) Right click the project and choose “Rebuild All”, as Fig 4-4 shows. The km4_bootloader, km4_application should compile in order.

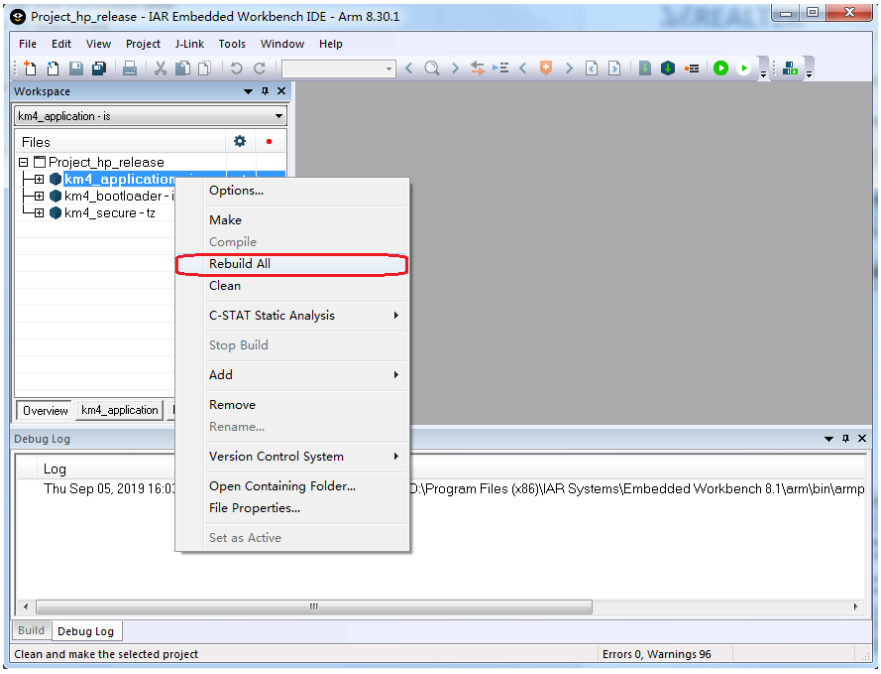
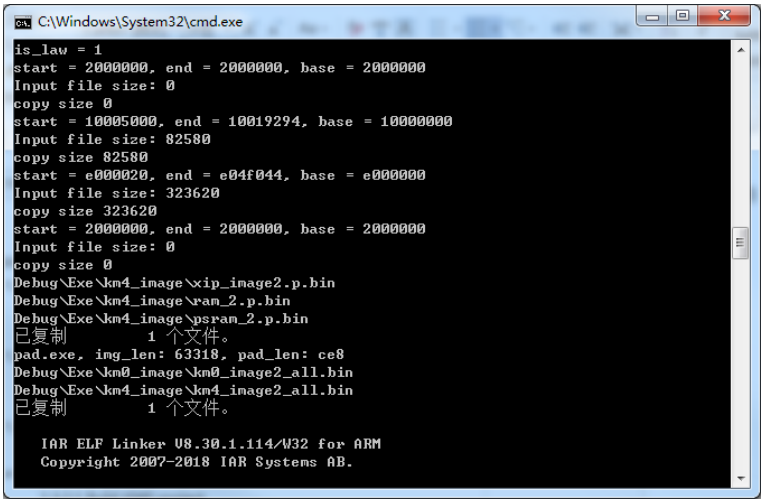


Fig 4-4 Building KM4 project

Note:

- After building each project, IAR will pop up a command prompt window shown in bellow to execute post-build action to generate images from executable files. This may takes several seconds. Do not stop it while it is in progress. After post-build action is completed, the window would disappear automatically.



(5) After compile, the images km4_boot_all.bin and km0_km4_image2.bin can be find in **ambd_amazon-freertos\projects\realtek\amebaD\IAR\aws_demos\Debug\Exe\km4_image**.

(6) The generated images can be downloaded to flash by ImageTool:

5 ImageTool

The tool can be find in ambd_amazon-freertos\vendors\realtek\tools\ameba-image-Tool-v2.4.1\

5.1 Introduction

This chapter introduces how to use ImageTool to encrypt, generate and download images. As show in Fig 5-1, ImageTool has four tabpages.

- Download: used as image download server to transmit images to Ameba through UART.

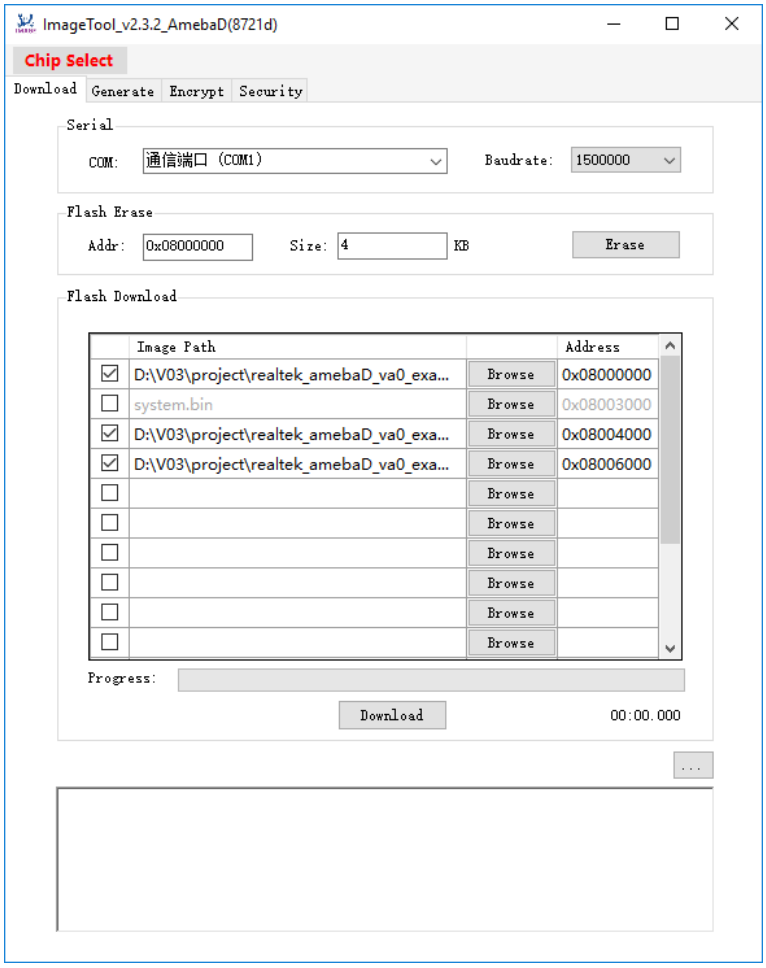


Fig 5-1 ImageTool UI

5.2 Environment Setup

5.2.1 Hardware Setup

The hardware setup is shown in Fig 5-2.

Note: If using external UART to download images, FT232 USB to UART dongle must be used.

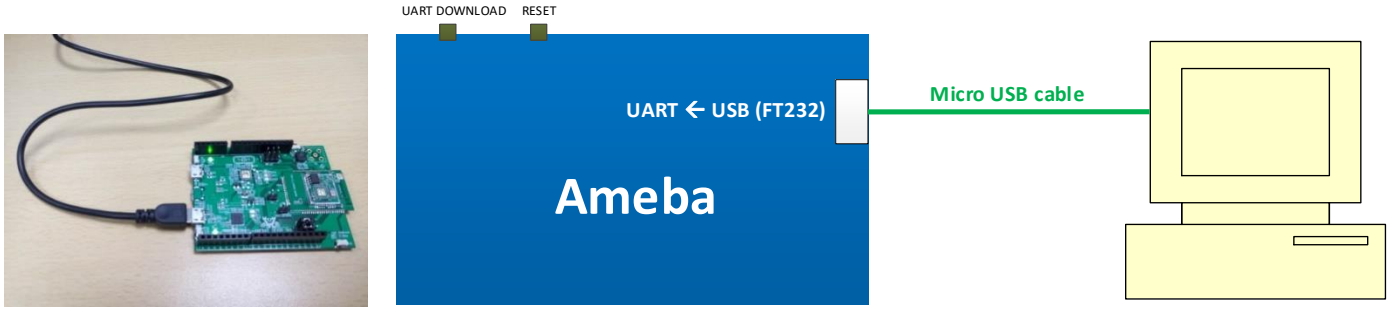


Fig 5-2 Hardware setup

5.2.2 Software Setup

- Environment Requirements: EX. WinXP, Win 7 Above, Microsoft .NET Framework 3.5
- ImageTool.exe Location: **vendors\realtek\tools\ameba-image-Tool-v2.4.1\ImageTool.exe**

Name	Date modified	Type	Size
ChangeLog.txt	7/29/2019 11:52 AM	Text Document	4 KB
Download.ini	11/4/2019 5:44 PM	Configuration sett...	2 KB
Encrypt.ini	11/4/2019 5:44 PM	Configuration sett...	1 KB
ImageTool.exe	7/29/2019 11:52 AM	Application	282 KB
ImageTool.pdb	7/29/2019 11:52 AM	VisualStudio.pdb....	178 KB
ImageTool.vshost.exe	8/20/2018 1:41 PM	Application	14 KB
ImageTool.vshost.exe.manifest	8/20/2018 1:41 PM	MANIFEST File	1 KB
imgtool_flashloader_amebad.bin	6/6/2019 3:15 PM	BIN File	5 KB
imgtool_flashloader_amebaz.bin	6/6/2019 3:15 PM	BIN File	6 KB
SB.exe	8/20/2018 1:41 PM	Application	189 KB
system.bin	8/6/2019 9:53 AM	BIN File	4 KB
TestListView.dll	8/20/2018 1:41 PM	Application extens...	5 KB
TestListView.pdb	8/20/2018 1:41 PM	VisualStudio.pdb....	14 KB

5.3 Download

5.3.1 Image Download

Assuming that the ImageTool on PC is a server, it sends images files to Ameba (client) through UART. There are two ways to download images to board.

5.3.1.1 Based on Hardware Reset

The way based on hardware reset is a manual method to download images, and it is the primary and recommended method.

(1) Enter into UART_DOWNLOAD mode.
   a) Push the **UART DOWNLOAD** button and keep it pressed.
   b) Re-power on the board or press the **Reset** button.
   c) Release the **UART DOWNLOAD** button.
   Now, Ameba board gets into UART_DOWNLOAD mode and is ready to receive data.

(2) Click **Chip Select** (in red) on UI and select chip (AmebaD).

(3) Select the corresponding serial port and transmission baud rate. The default baud rate is 1.5Mbps (recommended).

(4) Click the **Browse** button to select the images (**km0_boot_all.bin/km4_boot_all.bin/km0_km4_image2.bin**) to be programmed and input addresses.
   - The image path is located in **{*path*}\projects\realtek\amebaD\IAR\aws_demos\Debug\Exe\km0_image** and **{*path*}\projects\realtek\amebaD\IAR\aws_demos\Debug\Exe\km4_image**, where **{*path*}** is the location of the project on your own computer.
   - The default target address is the SDK default image address, you can use it directly.

(5) Click **Download** button to start. The progress bar will show the transmit progress of each image. You can also get the message of operation successfully or errors from the log window.

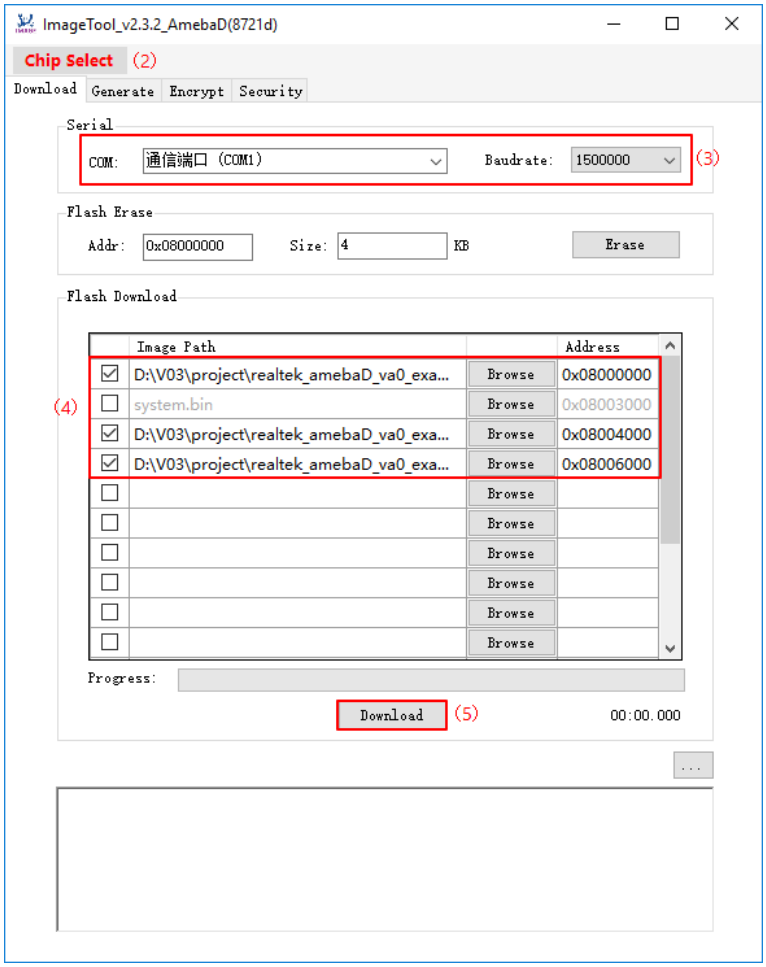


Fig 5-3 ImageTool 'Download' tabpage setting

6 MQTT Demo

6.1 Get Device Log

Install Tera Term to get device log

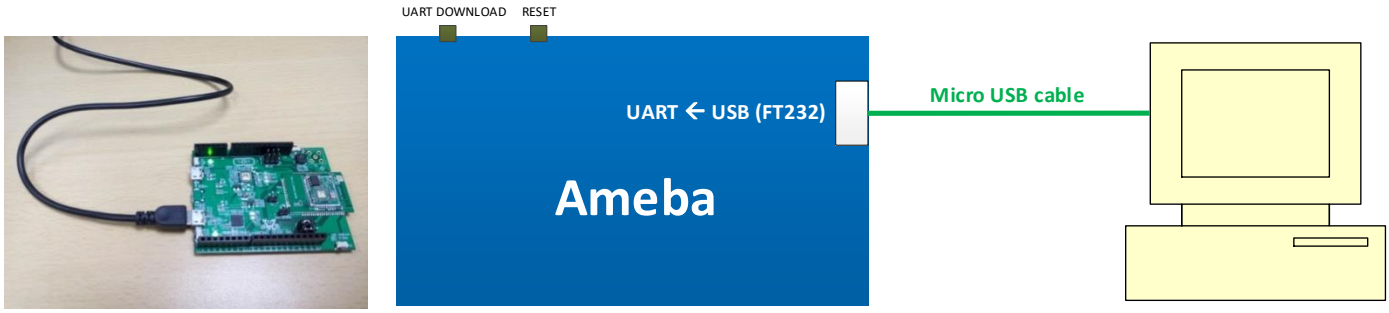
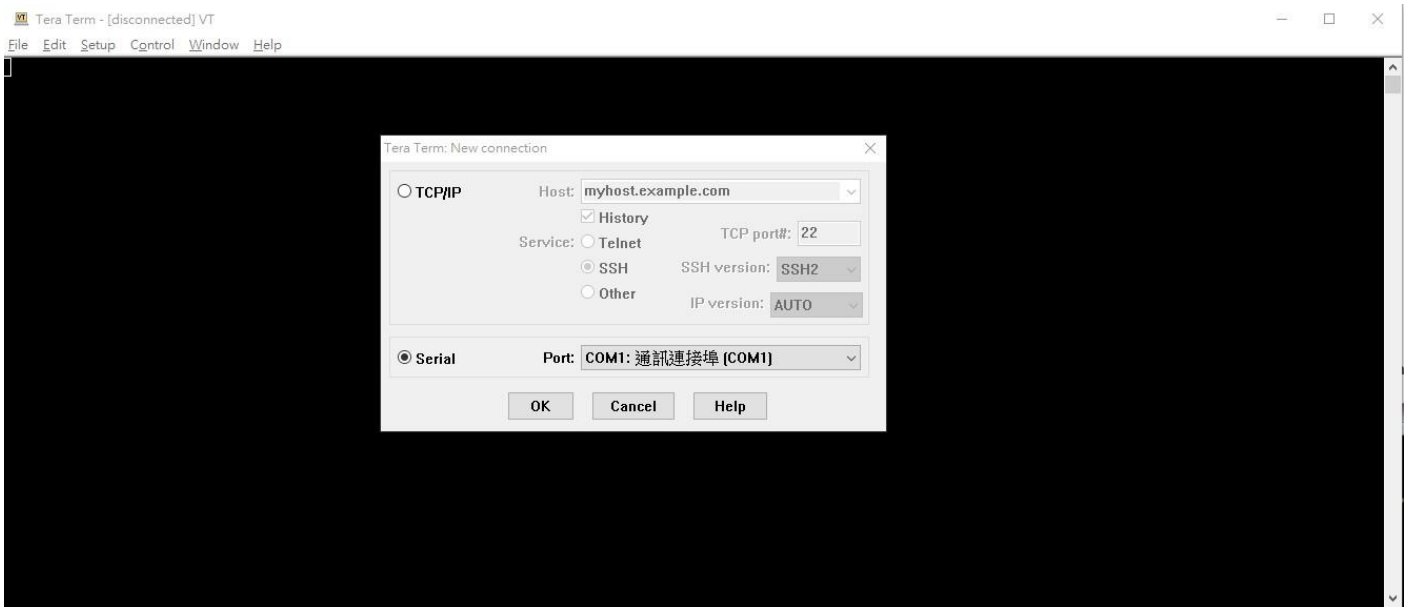


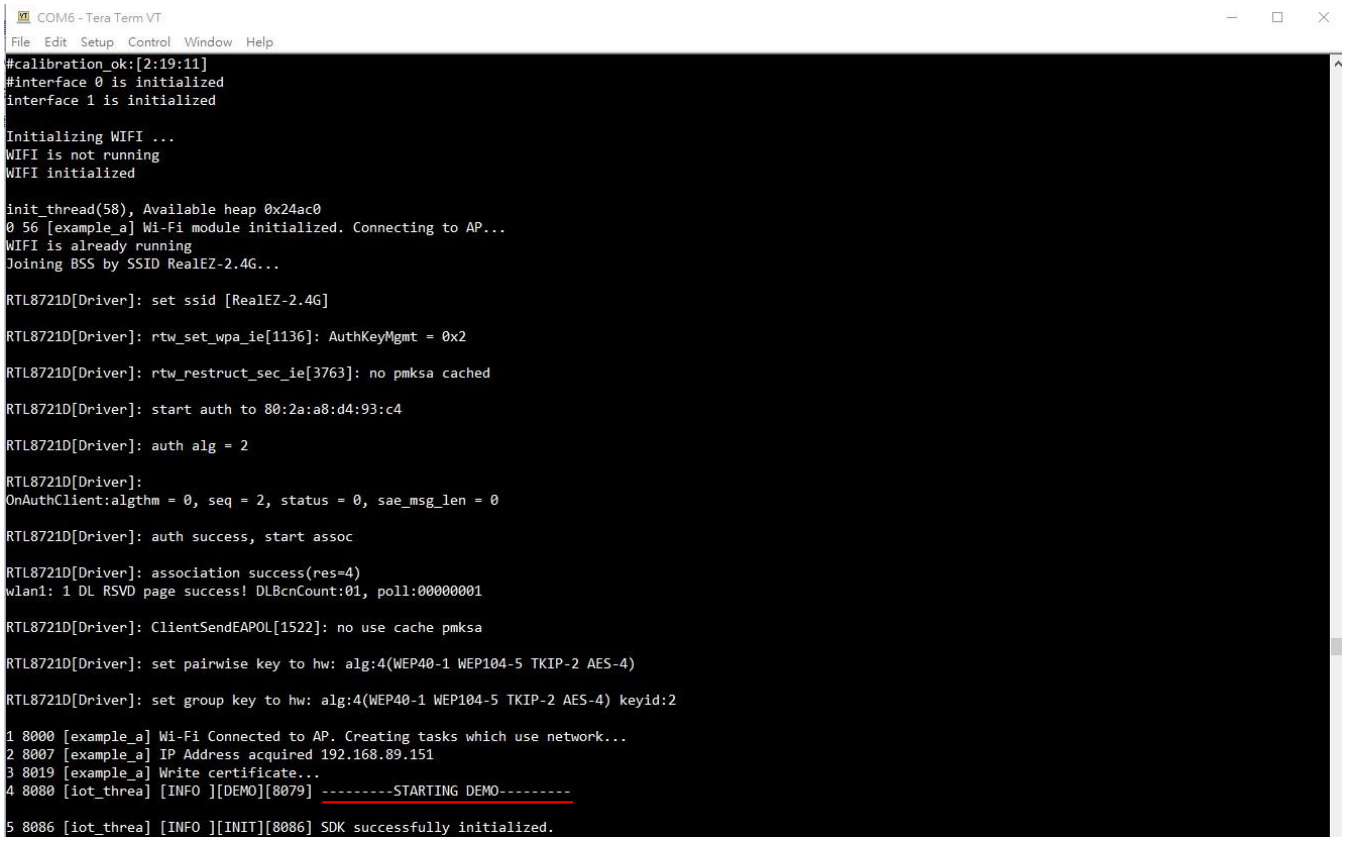
Fig 6-1 Hardware setup

The serial port is same with ImageTool that get from 3.3.1.1 step (3).

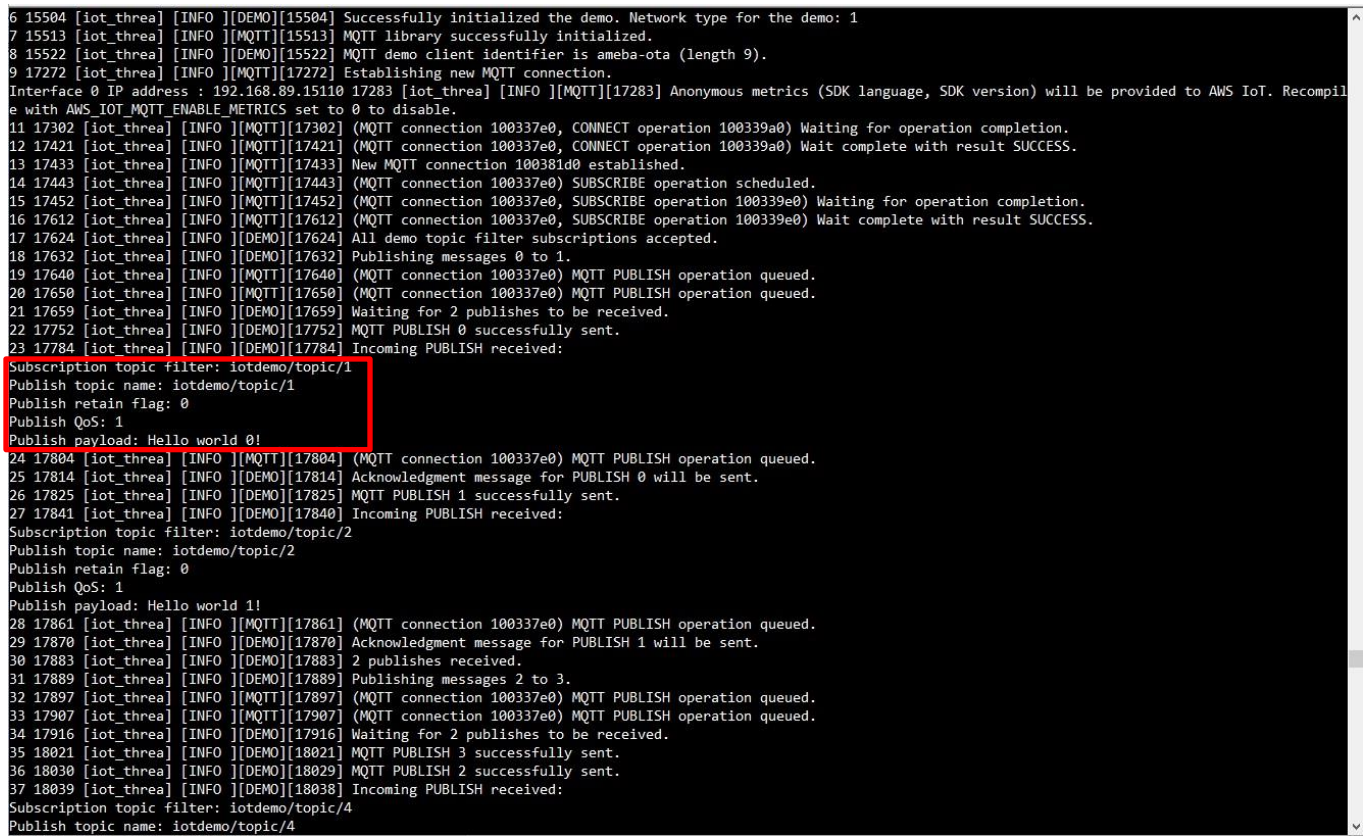


6.2 Run MQTT Demo

Default setting of SDK are enable MQTT demo. Once the AmebaD EVB has rebooted, the application will automatically start run MQTT demo and communicate to IoT Core.



...



...

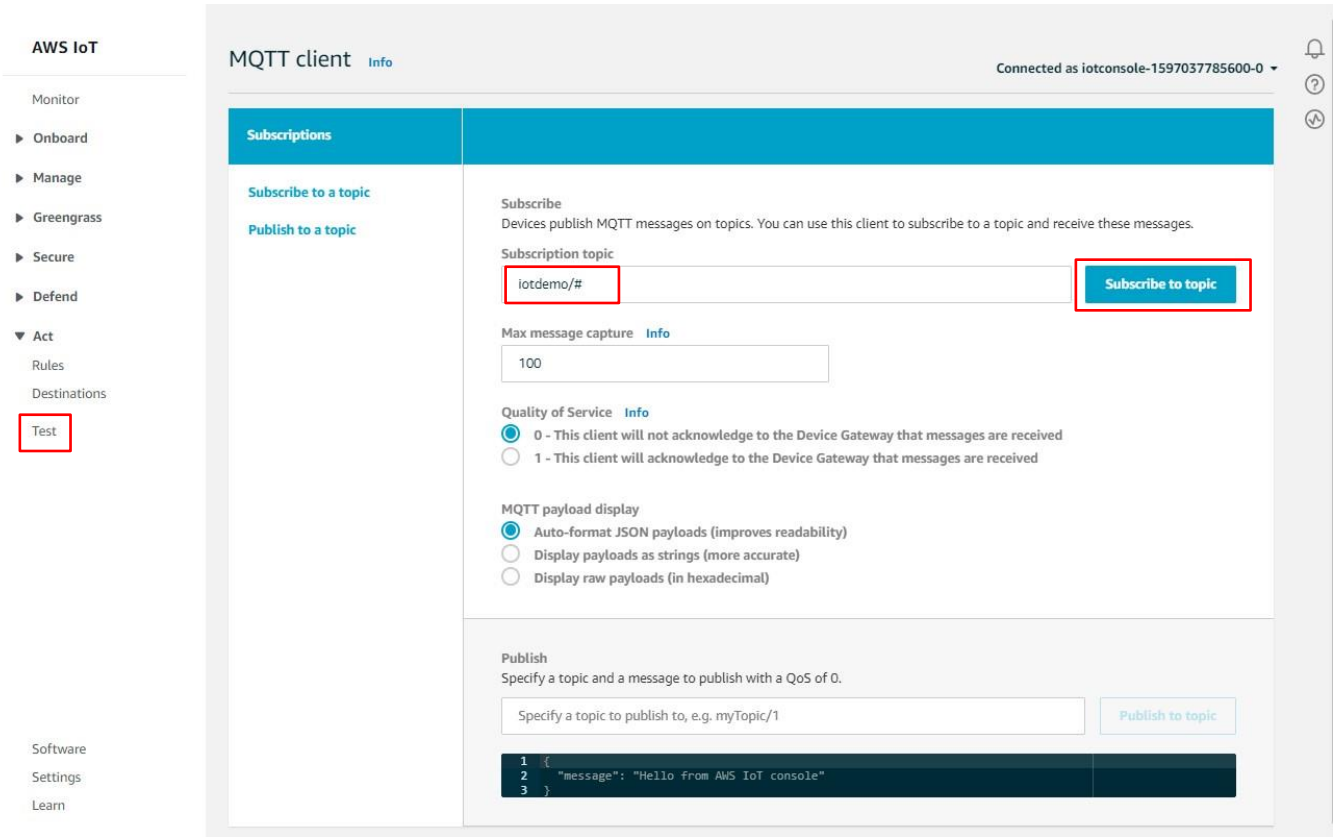
```
Publish payload: Hello world 16!
132 19827 [iot_threa] [INFO ][MQTT][19827] (MQTT connection 100337e0) MQTT PUBLISH operation queued.
133 19837 [iot_threa] [INFO ][DEMO][19837] Acknowledgment message for PUBLISH 16 will be sent.
134 19851 [iot_threa] [INFO ][DEMO][19851] 2 publishes received.
135 19857 [iot_threa] [INFO ][DEMO][19857] Publishing messages 18 to 19.
136 19865 [iot_threa] [INFO ][MQTT][19865] (MQTT connection 100337e0) MQTT PUBLISH operation queued.
137 19876 [iot_threa] [INFO ][MQTT][19876] (MQTT connection 100337e0) MQTT PUBLISH operation queued.
138 19885 [iot_threa] [INFO ][DEMO][19885] Waiting for 2 publishes to be received.
139 19953 [iot_threa] [INFO ][DEMO][19953] MQTT PUBLISH 18 successfully sent.
140 19980 [iot_threa] [INFO ][DEMO][19980] Incoming PUBLISH received:
Subscription topic filter: iotdemo/topic/3
Publish topic name: iotdemo/topic/3
Publish retain flag: 0
Publish QoS: 1
Publish payload: Hello world 18!
141 20001 [iot_threa] [INFO ][MQTT][20001] (MQTT connection 100337e0) MQTT PUBLISH operation queued.
142 20011 [iot_threa] [INFO ][DEMO][20011] Acknowledgment message for PUBLISH 18 will be sent.
143 20053 [iot_threa] [INFO ][DEMO][20053] MQTT PUBLISH 19 successfully sent.
144 20069 [iot_threa] [INFO ][DEMO][20068] Incoming PUBLISH received:
Subscription topic filter: iotdemo/topic/4
Publish topic name: iotdemo/topic/4
Publish retain flag: 0
Publish QoS: 1
Publish payload: Hello world 19!
145 20089 [iot_threa] [INFO ][MQTT][20089] (MQTT connection 100337e0) MQTT PUBLISH operation queued.
146 20099 [iot_threa] [INFO ][DEMO][20099] Acknowledgment message for PUBLISH 19 will be sent.
147 20108 [iot_threa] [INFO ][DEMO][20108] 2 publishes received.
148 20116 [iot_threa] [INFO ][MQTT][20116] (MQTT connection 100337e0) UNSUBSCRIBE operation scheduled.
149 20129 [iot_threa] [INFO ][MQTT][20128] (MQTT connection 100337e0, UNSUBSCRIBE operation 100339e0) Waiting for operation completion.
150 20322 [iot_threa] [INFO ][MQTT][20321] (MQTT connection 100337e0, UNSUBSCRIBE operation 100339e0) Wait complete with result SUCCESS.
151 20335 [iot_threa] [INFO ][MQTT][20335] (MQTT connection 100337e0) Disconnecting connection.
152 20347 [iot_threa] [INFO ][MQTT][20347] (MQTT connection 100337e0, DISCONNECT operation 100339e0) Waiting for operation completion.
153 20359 [iot_threa] [INFO ][MQTT][20359] (MQTT connection 100337e0, DISCONNECT operation 100339e0) Wait complete with result SUCCESS.
154 20371 [iot_threa] [INFO ][MQTT][20371] (MQTT connection 100337e0) Connection disconnected.
155 20380 [iot_threa] [INFO ][MQTT][20380] (MQTT connection 100337e0) Network connection closed.
156 21622 [iot_threa] [INFO ][MQTT][21622] (MQTT connection 100337e0) Network connection destroyed.
157 21631 [iot_threa] [INFO ][MQTT][21631] MQTT library cleanup done.
158 21637 [iot_threa] [INFO ][DEMO][21637] Demo completed successfully.

LwIP_DHCP: dhcp stop.
Deinitializing WIFI ...
159 21772 [iot_threa] [INFO ][INIT][21772] SDK cleanup done.
160 21777 [iot_threa] [INFO ][DEMO][21777] -------DEMO FINISHED-------
```

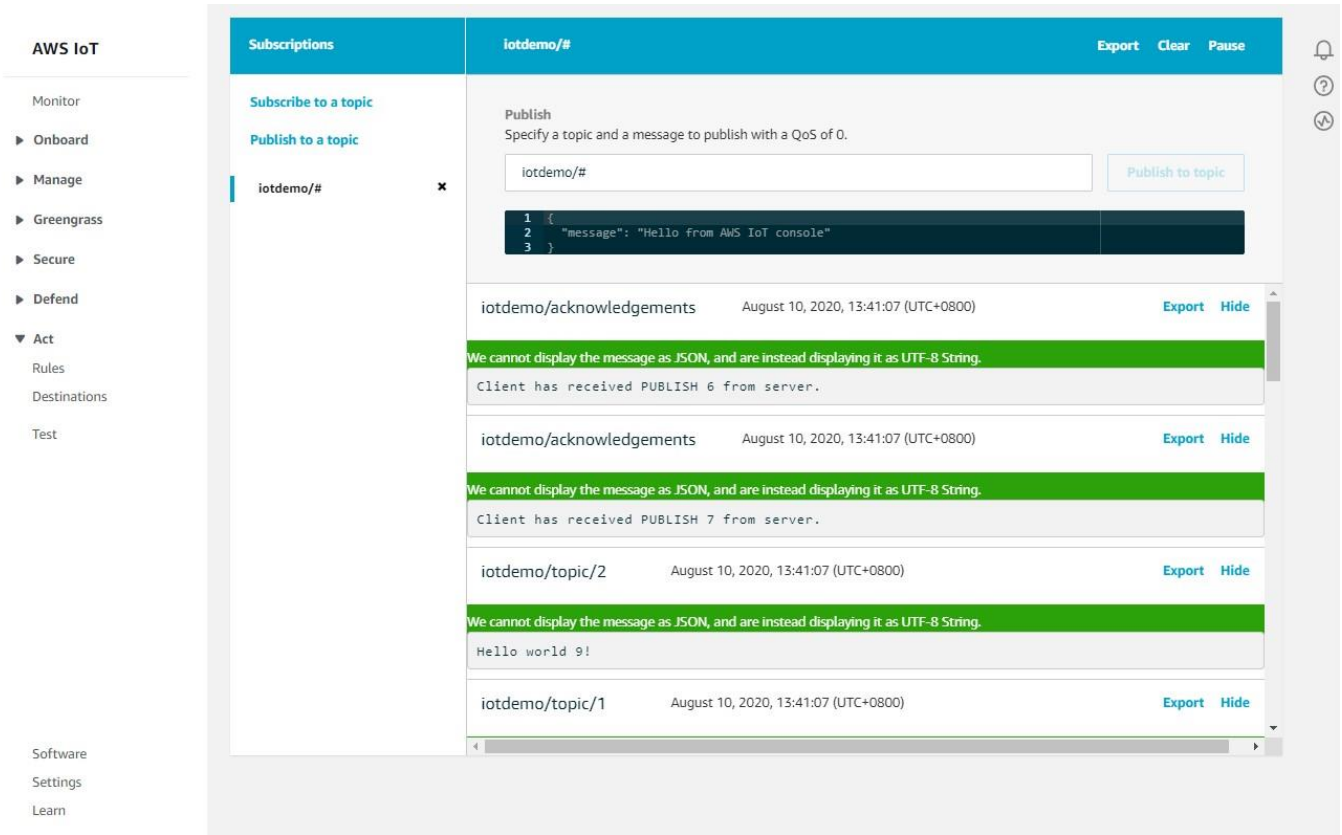
Monitor connection summary.

6.3 Monitoring MQTT messages on the cloud

To subscribe to the MQTT topic with the AWS IoT MQTT client

1. Sign in to the AWS IoT console.
2. In the navigation pane, choose Test to open the MQTT client.
3. In Subscription topic, enter iotdemo/#, and then choose Subscribe to topic.





7 Secure Boot

AmebaD use secure boot to mitigate the OTA threats.

Secure boot aims at firmware protection, which prevents attackers from modifying or replacing firmware maliciously. When the chip is power on, the ROM secure boot executes to check the validation of image signature.

If the signature is valid, authentication will be successful, which means the firmware is safe and the subsequent operations can be continued. Otherwise, the SoC goes into infinite loop.

Users do not need to implement the secure boot code themselves, which are already contained in the ROM code. Users only need to program the bootloader with signature information into Flash.

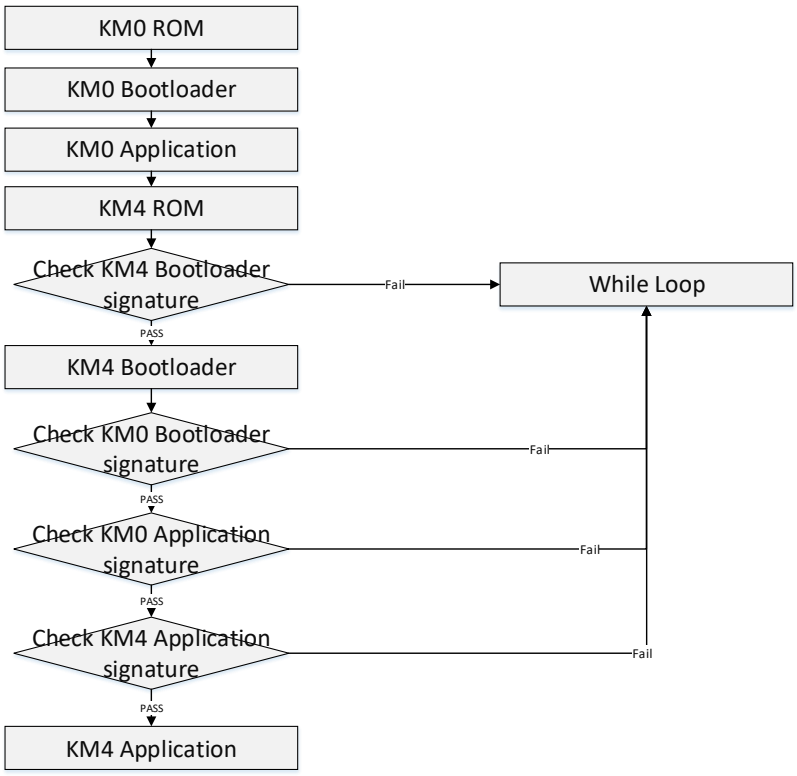


Fig 7-1 Secure Boot Sequence

7.1 Secure Boot eFuse

Name	Address	Size	Default	Description
Secure Boot Enable	Physical Map 0x1C1[5]	1 bit	1	Secure boot is enabled or not. ● 0: Secure boot is enabled. ● 1: Secure boot is disabled.
Secure Boot Public Key	Physical Map	32 bytes	Each Byte is 0xFF	For example, the secure boot key is [37313468df98b5b6cec9d5f6b347fdcf1cd2f06c4f5887edf6d3f8c7a88745e4]; secure boot key needs to be programmed into eFuse

7.2 Secure Boot Image Generation

Bootloader signing can be achieved by configuring security functions and key in SDK, or using ImageTool to process image.

7.2.1 SDK Configuration

In SDK, a configuration file can be used by users to configure security parameters, including secure boot.

Project	Configuration file	Path	Configuration item
IAR project	security_config.cmd	project\realtek_amebaD_va0_example\EWARM-RELEASE	set SBOOT_ENABLE=1 set SBOOT_SEED=FBED6DA104000000054273A0050273A00FAFAA77766B1A977FFFFFFFF24000000

7.2.2 More Information

Secure boot is a complicated feature. It cannot rollback after enable it because it need write date to eFuse. The EVB board maybe boot failed if setup not complete. For more information about how enable secure boot feature. Please contact RealTek on https://www.amebaiot.com/ to get more information.

7.2.3 Mitigate the OTA threats

- **An attacker hijacks the device's connection to the server to deliver a malicious firmware image.**

  Upon boot, the bootloader verifies the cryptographic signature of the image using a known certificate. If the verification fails, the boot procedure stops and reboot to the original image.

- **An attacker exploits a buffer overflow to introduce malicious behavior to the existing firmware image stored in flash.**

  Upon boot, the bootloader verifies, as previously described. When verification fails, the bootloader halts.

- **An attacker boots the device to a previously stored image, which is exploitable.**

  Flash sectors storing the last image are erased upon successful installation and test of a new image.

- **An OTA update delivers a faulty or malicious image that bricks the device**

  Case1: Update a faulty image => Bootloader has boot selection mechanism, it can rollback to previous image if needed.

  Case2: Malicious image => OTA job should fail because OTA verify firmware integrity before applying firmware update

- **An attacker patches the bootloader to bypass image verification so the device will accept unsigned images.**

  The bootloader is in ROM (read-only memory), and cannot be modified.

- **An attacker replaces the verification certificate so the device will accept malicious images.**

  The secure boot key is in eFuse. It cannot be modified and be read out after enable read/write protection.

8 Troubleshooting

If these steps don’t work, look at the device log in the serial terminal. You should see some text that indicates the source of the problem.

For general troubleshooting information about Getting Started with FreeRTOS, see Troubleshooting getting started.