TM51010 Wi-Fi & BLE M.2 Wireless Module Amazon FreeRTOS Getting Started Guide

Good Way Technology Co., Ltd.
3F, No. 135, Ln. 235, Baociao Rd., Sindian Dist., New Taipei City 231, Taiwan
Tel: +886- 2-8919-1200, Fax: +886- 2-8919-1220
www.gtrend-auto.com

COPYRIGHT

DISCLAIMER

Please Read Carefully:

Good Way Technology Co., Ltd., (Good Way) 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 Good Way data sheets is permissible only if reproduction is without alteration and is accompanied by all associated warranties, conditions, limitations, and notices. Good Way 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 Good Way 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 Good Way 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 Good Way products, Customer will thoroughly test such applications and the functionality of such Good Way products as used in such applications.

Good Way'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 Good Way products; by downloading, accessing or using Good Way'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 Good Way Resources solely for this purpose and subject to the terms of this Notice.

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

Customer is authorized to use, copy and modify any individual Good Way Resource only in connection with the development of applications that include the Good Way product(s) identified in such Good Way Resource. No other license, express or implied, by estoppel or otherwise to any other Good Way intellectual property right, and no license to any technology or intellectual property right of Good Way 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 Good Way 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 Good Way Resources may require a license from a third party under the patents or other intellectual property of the third party, or a license from Good Way under the patents or other Good Way's intellectual property.

Good Way's Resources are provided "as is" and with all faults. Good Way 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.

Good Way 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 Good Way Resources or otherwise. In no event shall Good Way be liable for any actual, direct, special, collateral, indirect, punitive, incidental, consequential or exemplary damages in connection with or arising out of Good Way's Resources or use thereof, and regardless of whether Good Way has been advised of the possibility of such damages. Good Way is not responsible for any failure to meet such industry standard requirements.

Where Good Way 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 Good Way 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 Good Way 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

Good Way is a trademark of Good Way Technology Co., Ltd. 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.

Table of Contents

1 TM51010 Wi-Fi & BLE M.2 Wireless Module

1.1 Wi-Fi & BLE M.2 Wireless Module

TM51010 Wi-Fi & BLE M.2 Wireless Module web page: http://www.gtrend-auto.com/products-M2-Mesh-Controller-Board.asp

The Wi-Fi & BLE M.2 Wireless Module is a powerful, generic Wi-Fi/BLE Board based on highly integrated Realtek RTL8720DN and Nordic nRF52832 MCU with built-in security features and ultra-low power consumption. The embedded system product developers and device makers can now drastically shorten their development cycle and reduce time to market by using Good Way Wi-Fi & BLE M.2 Wireless Module.

- **Features**
  - IPEX antenna design for Wi-Fi and BLE to ensure better RF performance
  - Wi-Fi MCU with Amazon FreeRTOS to support Cloud service securely

- **Benefits**
  - Android SDK of BLE Mesh ready for fast installation and via Smartphone to ensure better user experience
  - Multi-Threading optimization to speed up network distribution process by saving time 1.5x than SIG Mesh

- **Specifications**

Wi-Fi	
Network Standards	IEEE 802.11 a/b/g/n 1x1
Operating Frequency	2.4GHz & 5GHz
Data Rate	Up to 150Mbps
Antenna	IPEX connector for external antenna
Bluetooth LE	
RF Protocol	BLE SIG Mesh
Operating Frequency	2.4GHz
Operation Range	30m (indoor open space)
Antenna	IPEX connector for external antenna
M.2 Interface	
VCC	support 3.3V
USB	x1
I2C	x1
GPIO	x15
Others	
Dimensions (L x W x H)	45 x 32 x 5.8mm
Operating Temp.	-20°C to +85°C

1.2 PCB Layout Overview

The PCB layout is shown in Fig 1-1

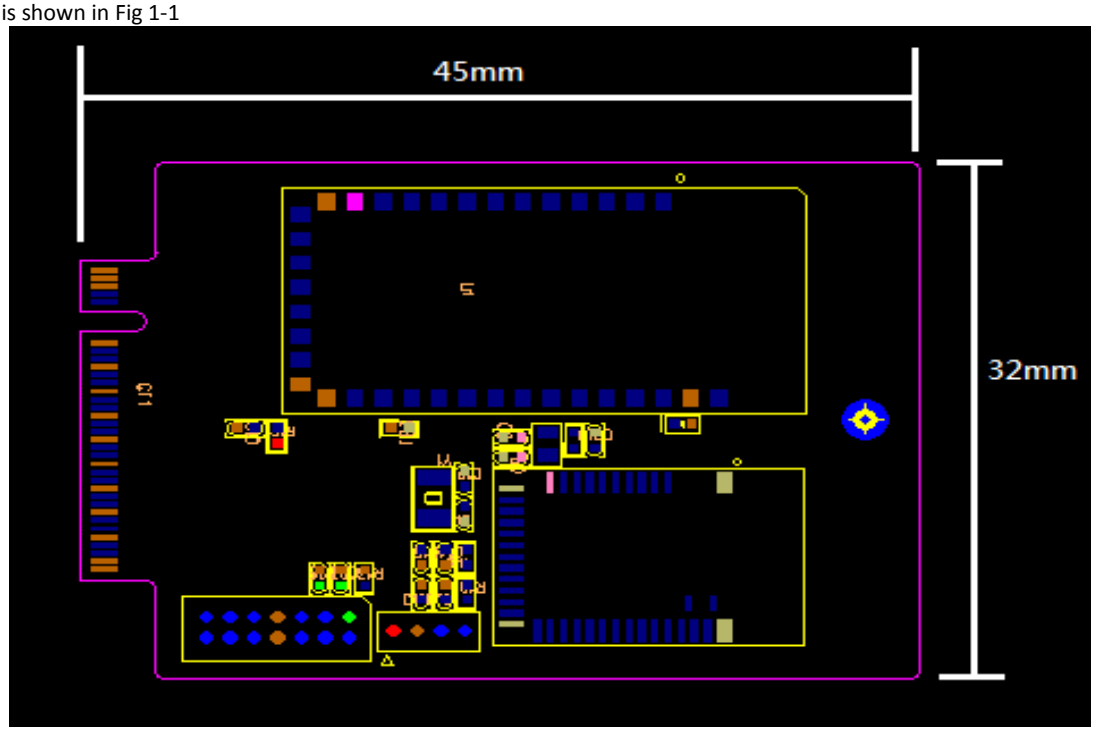


Fig 1-1 PCB layout

1.3 Pin Out

The pin out board is shown in Fig 1-2.

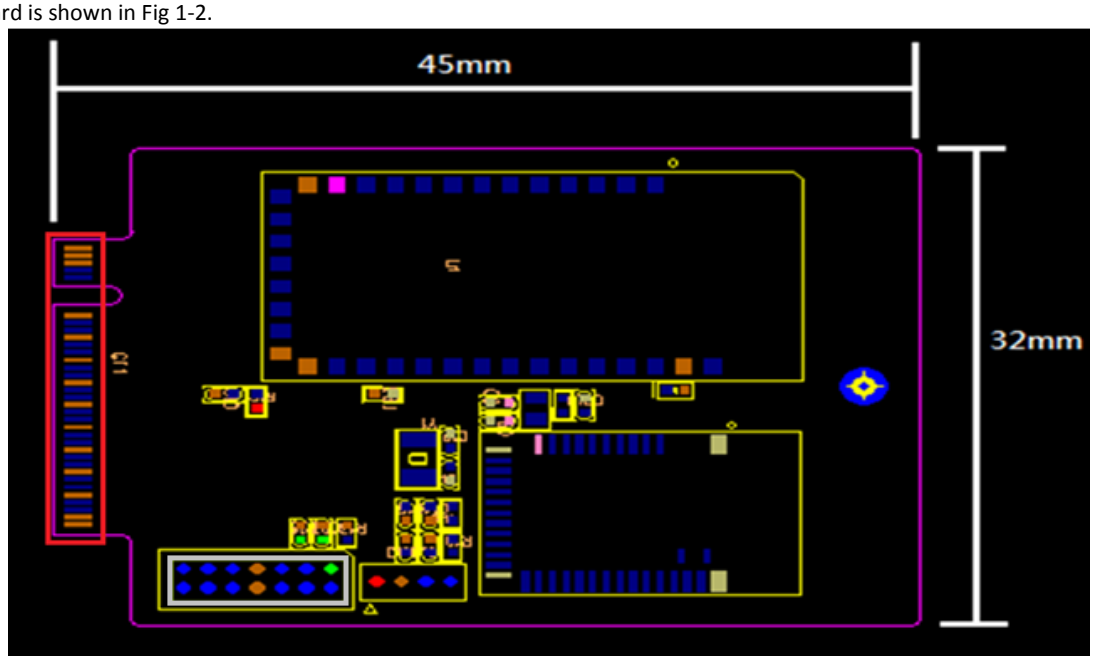


Fig 1-2 pin out

There are two rows of pins on the board.

- The pins in the red box are include VCC, GND, GPIO, I2C and USB.
- The pins in the gray box are include programmable and debug.

1.4 M.2 Pin Assignment

The M.2 pin number mapping is shown in Fig 1-3.

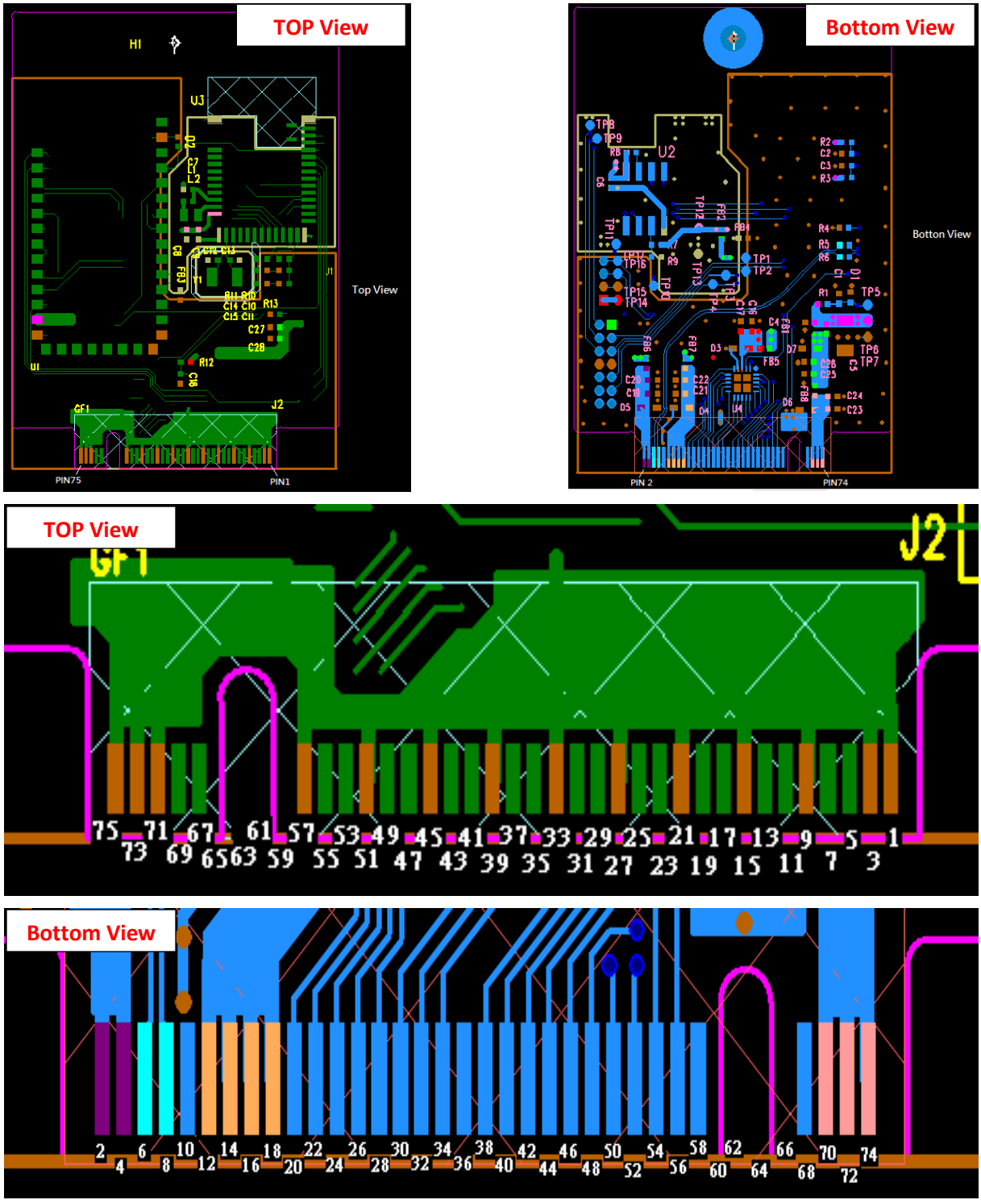


Fig 1-3 M.2 pin number mapping

Pin Assignment					
Pin No.	**Pin Name**	**Pin Description**	**Pin No.**	**Pin Name**	**Pin Description**
1	GND	Power Ground	39	GND	Ground
2	VDD_3V3	Power supply voltage 3.3V	40	GPIO	General Purpose Input/Output
3	GND	Power Ground	41	NC	No Connection
4	VDD_3V3	Power supply voltage 3.3V	42	GPIO	General Purpose Input/Output
5	NC	No Connection	43	NC	No Connection
6	USB_DP	USB_DP	44	GPIO	General Purpose Input/Output
7	NC	No Connection	45	GND	Power Ground
8	USB_DN	USB_DN	46	GPIO	General Purpose Input/Output
9	GND	Power Ground	47	NC	No Connection
10	NC	No Connection	48	GPIO	General Purpose Input/Output
11	NC	No Connection	49	NC	No Connection
12	VDD_3V3	Power supply voltage 3.3V	50	GPIO	General Purpose Input/Output
13	NC	No Connection	51	GND	Power Ground
14	VDD_3V3	Power supply voltage 3.3V	52	GPIO	General Purpose Input/Output
15	GND	Power Ground	53	NC	No Connection
16	VDD_3V3	Power supply voltage 3.3V	54	GPIO	General Purpose Input/Output
17	NC	No Connection	55	NC	No Connection
18	VDD_3V3	Power supply voltage 3.3V	56	GPIO	General Purpose Input/Output
19	NC	No Connection	57	GND	Power Ground
20	I2C_SCL	I2C Clock	58	NC	No Connection
21	GND	Power Ground	59	Notch	
22	I2C_SDA	I2C DATA	60	Notch	
23	NC	No Connection	61	Notch	
24	GPIO	General Purpose Input/Output	62	Notch	
25	NC	No Connection	63	Notch	
26	GPIO	General Purpose Input/Output	64	Notch	
27	GND	Ground	65	Notch	
28	GPIO	General Purpose Input/Output	66	Notch	
29	NC	No Connection	67	NC	No Connection
30	GPIO	General Purpose Input/Output	68	NC	No Connection
31	NC	No Connection	69	NC	No Connection
32	GPIO	General Purpose Input/Output	70	VDD_3V3	Power supply voltage 3.3V
33	GND	Power Ground	71	GND	Power Ground
34	GPIO	General Purpose Input/Output	72	VDD_3V3	Power supply voltage 3.3V
35	NC	No Connection	73	GND	Power Ground
36	NC	No Connection	74	VDD_3V3	Power supply voltage 3.3V

37	NC	No Connection	75	GND	Power Ground
38	GPIO	General Purpose Input/Output			

1.5 LOGUART and SWD

The LOGUART and SWD board pin mapping is shown in Fig 1-4.

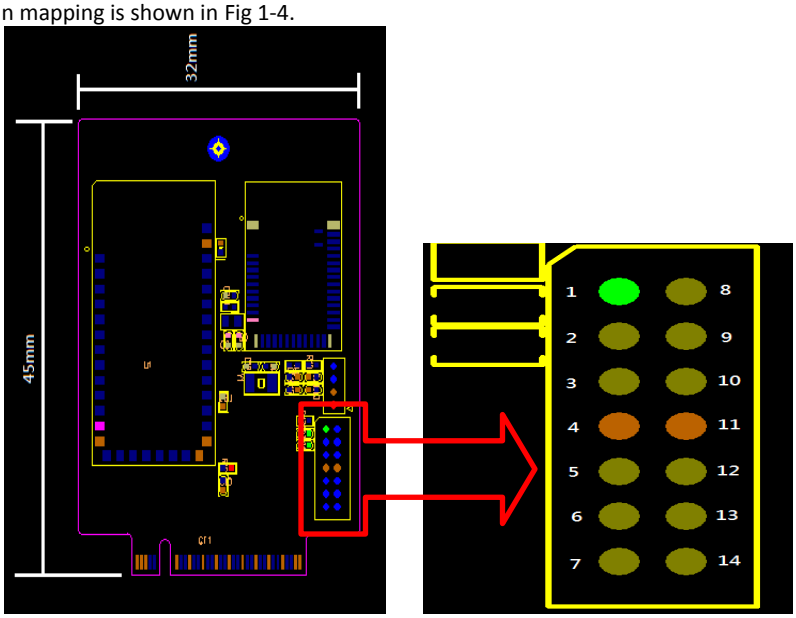


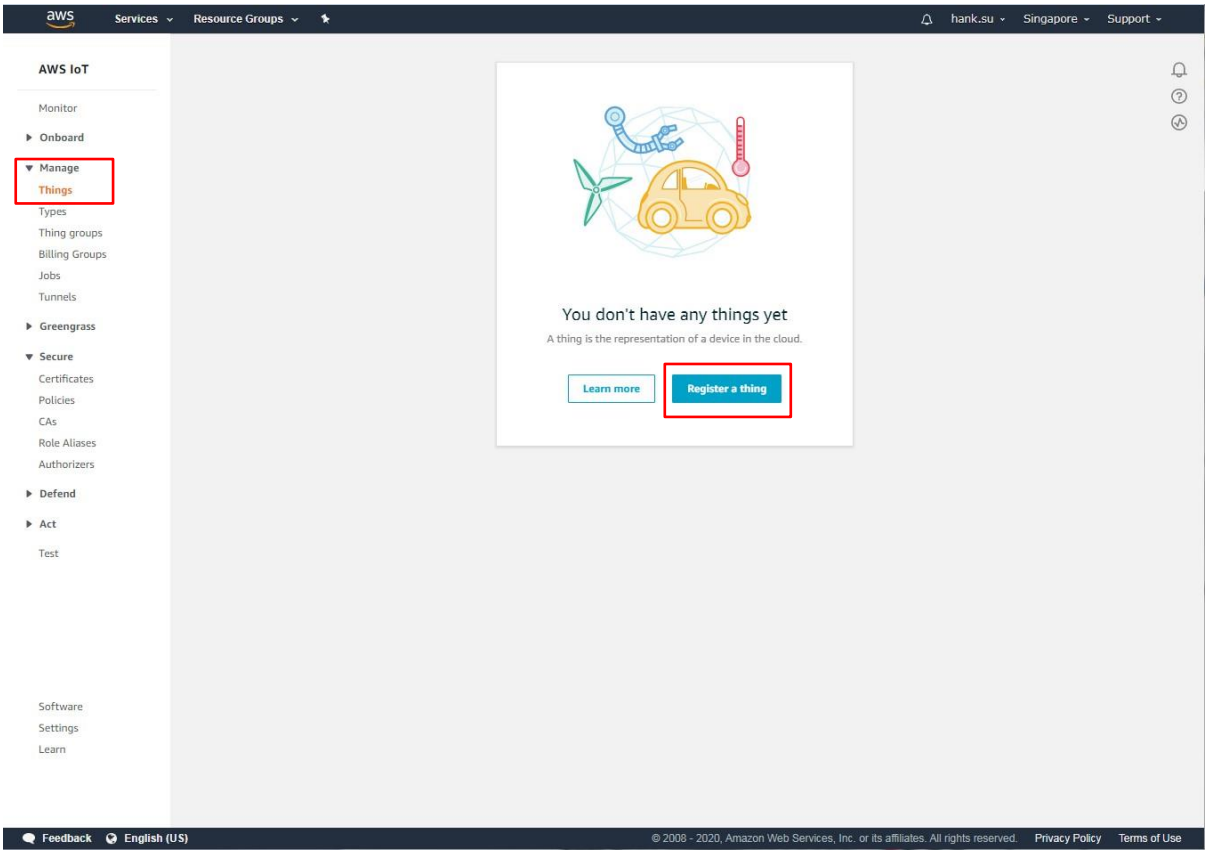
Fig 1-4 Pin number mapping

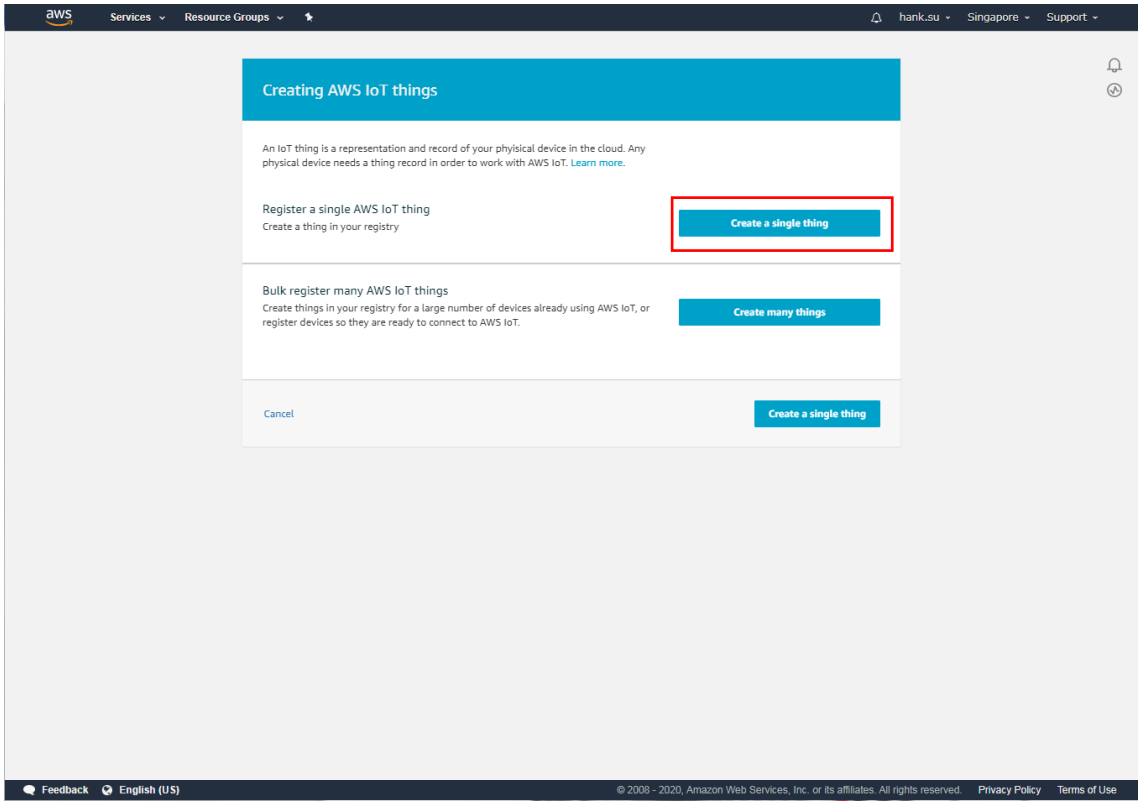
Pin Assignment		
Pin No.	**Pin Name**	**Pin Description**
1	VDD_3V3	Power supply voltage 3.3V
2	WIFI_UART_LOG_OUT	For WIFI debug and programming, Data out.
3	WIFI_UART_LOG_IN	For WIFI debug and programming, Data in.
4	GND	Power Ground
5	WIFI_SWDIO	For WIFI debug and programming, Serial wire I/O.
6	WIFI_SWCLK	For WIFI debug and programming, Serial wire clock input.
7	WIFI_RESET	Set this pin low reset WIFI.
8	NC	No Connection
9	BLE_SWCLK	For BLE debug and programming, Serial wire clock input.
10	BLE_SWDIO	For BLE debug and programming, Serial wire I/O.
11	GND	Power Ground
12	NC	No Connection
13	NC	No Connection
14	NC	No Connection

2 Configure AWS IoT Core

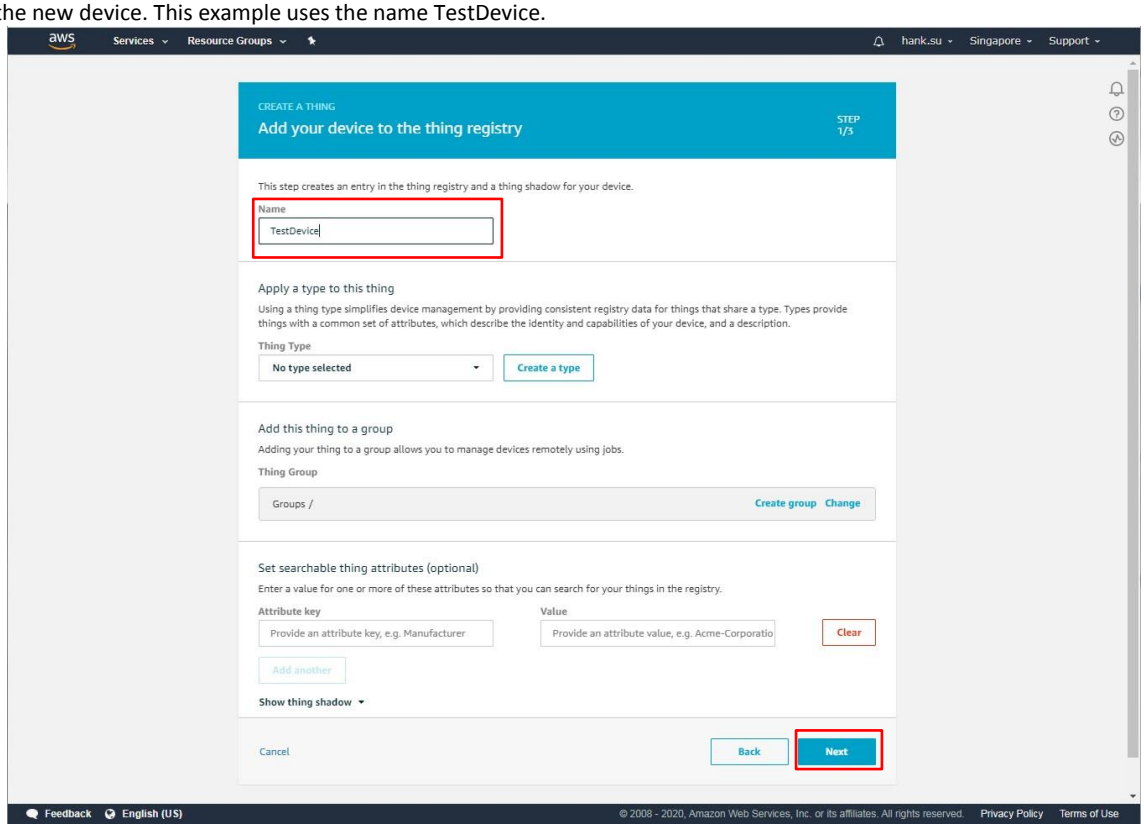
2.1 Create a New Device

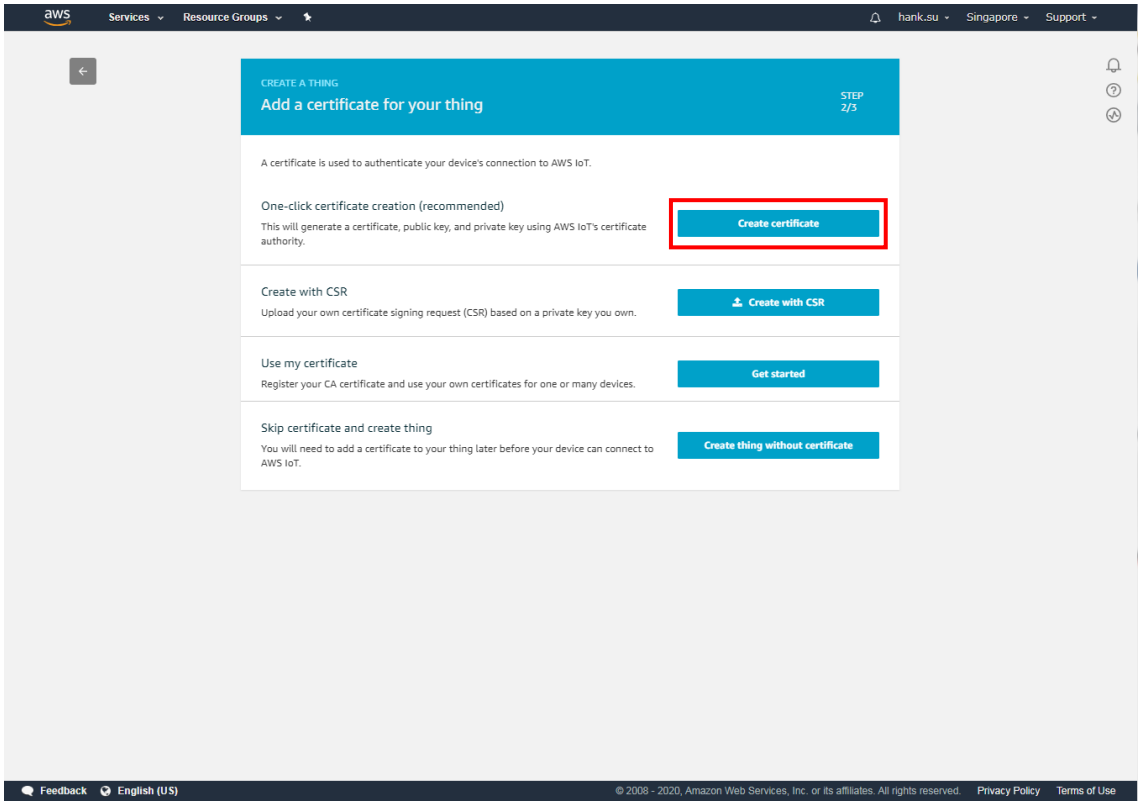
To create a new device, navigate to Manage -> Things in the left-hand navigation menu. Then click “Register a thing”.



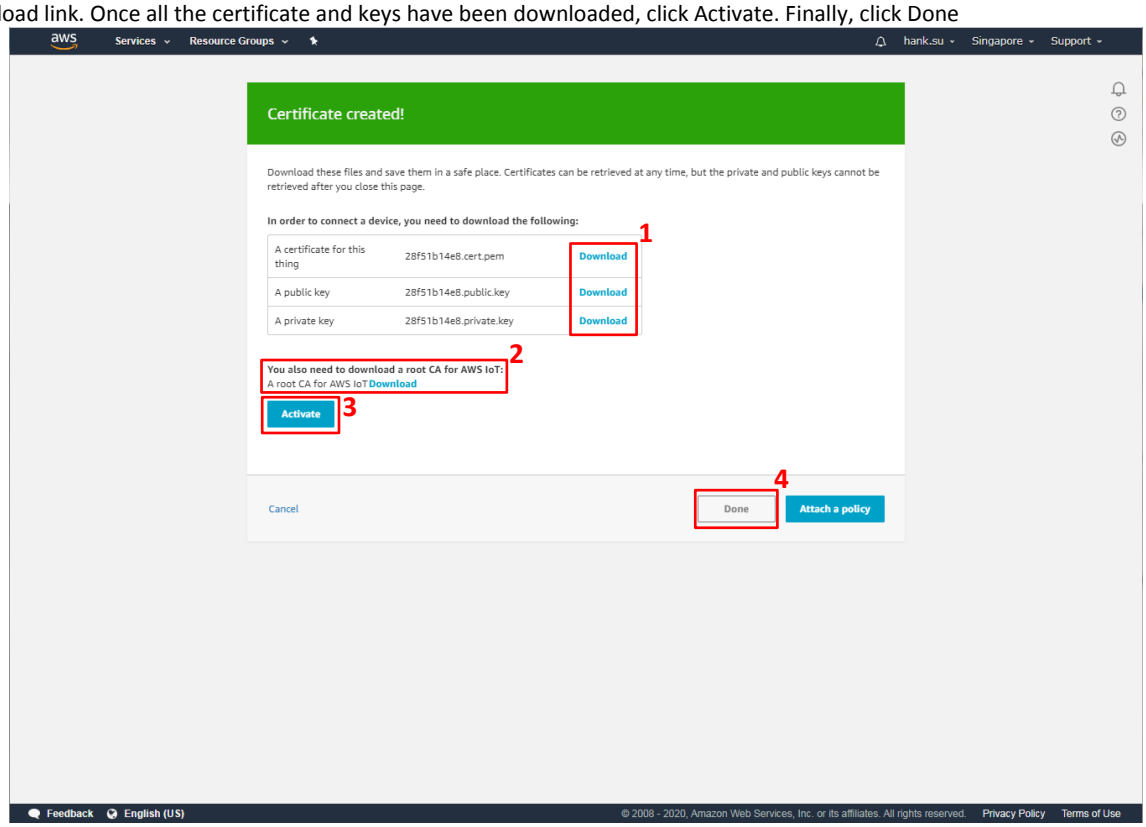


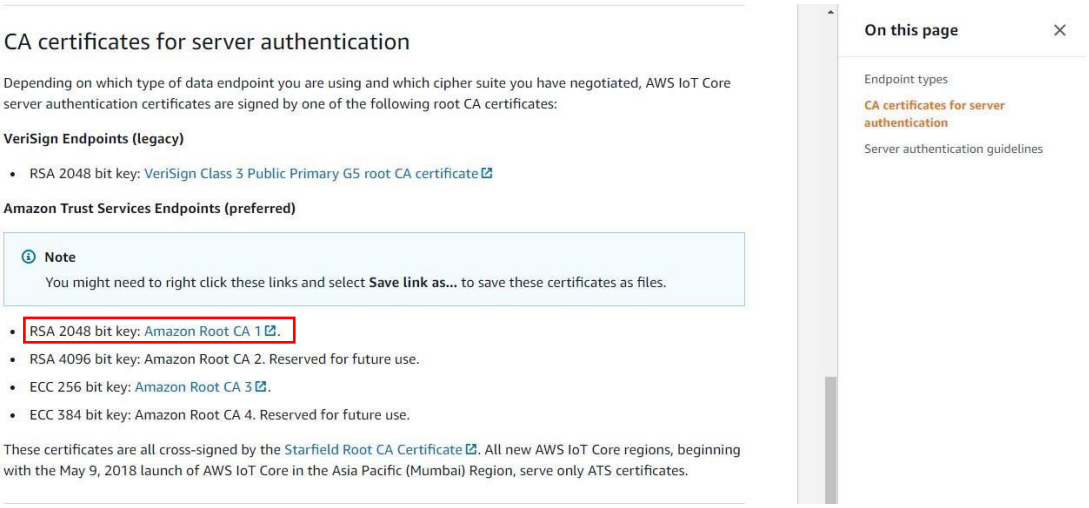
Then, name the new device. This example uses the name TestDevice.





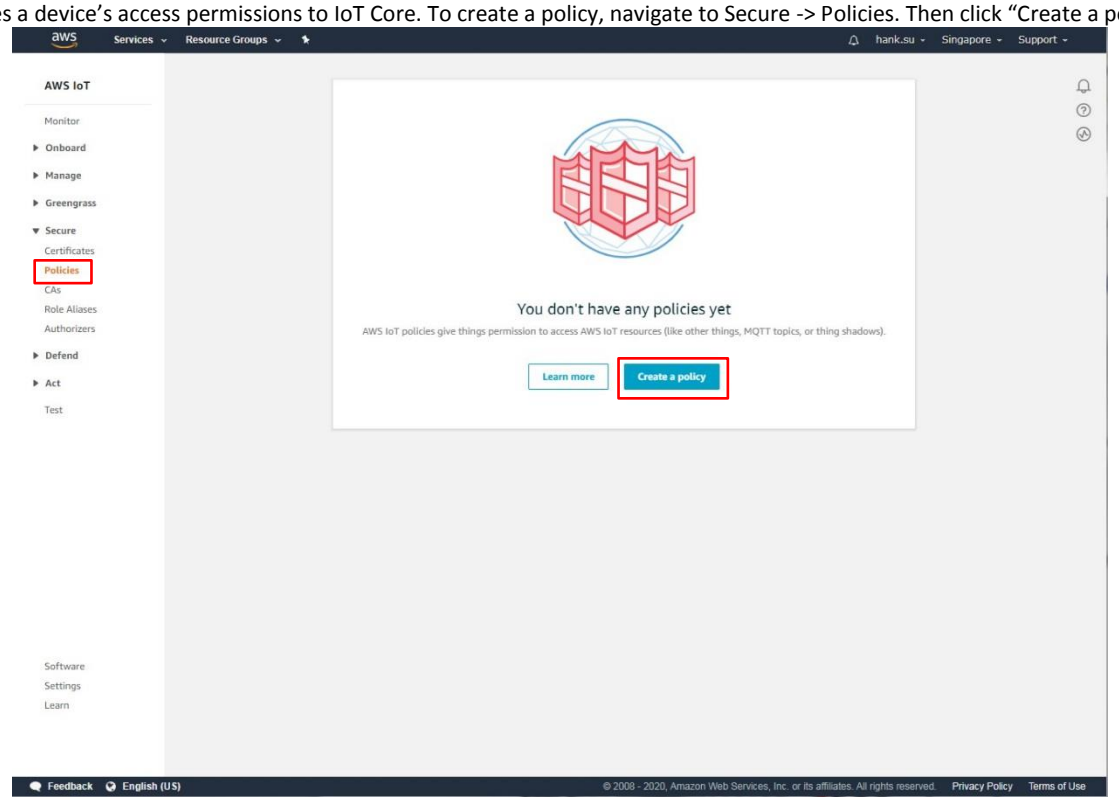
Download the certificate, public key, and private key for the device by clicking Download. Next, download the root CA for AWS IoT by clicking to the Download link. Once all the certificate and keys have been downloaded, click Activate. Finally, click Done



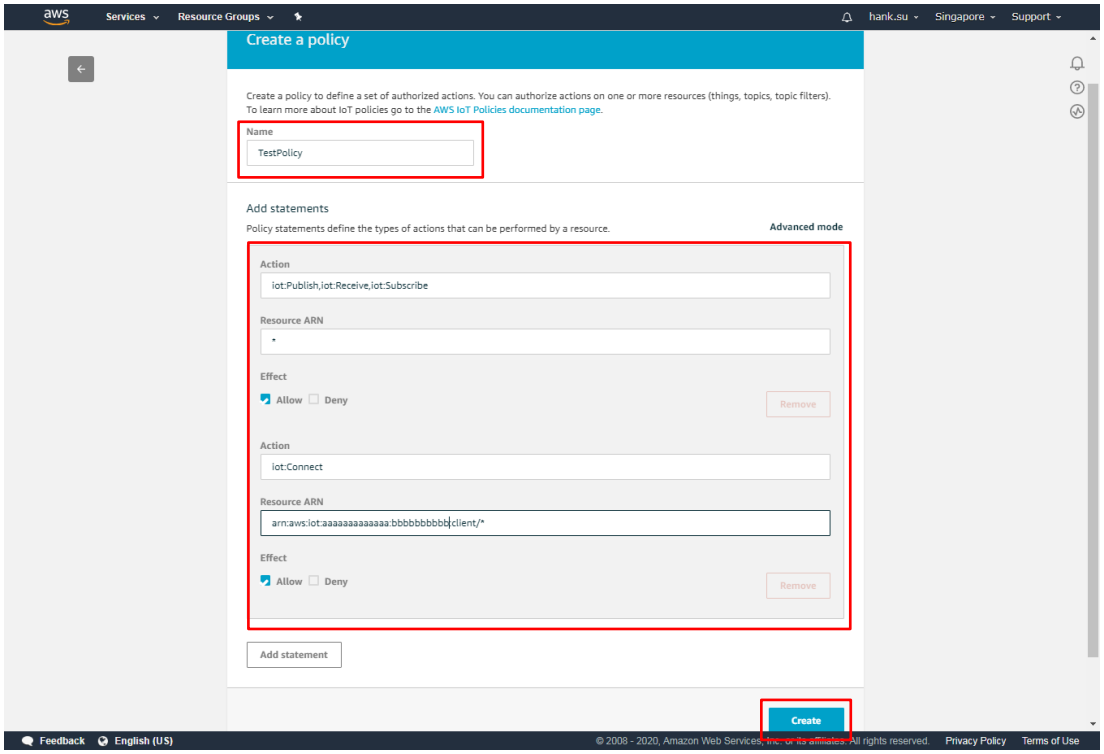


2.2 Create a policy

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

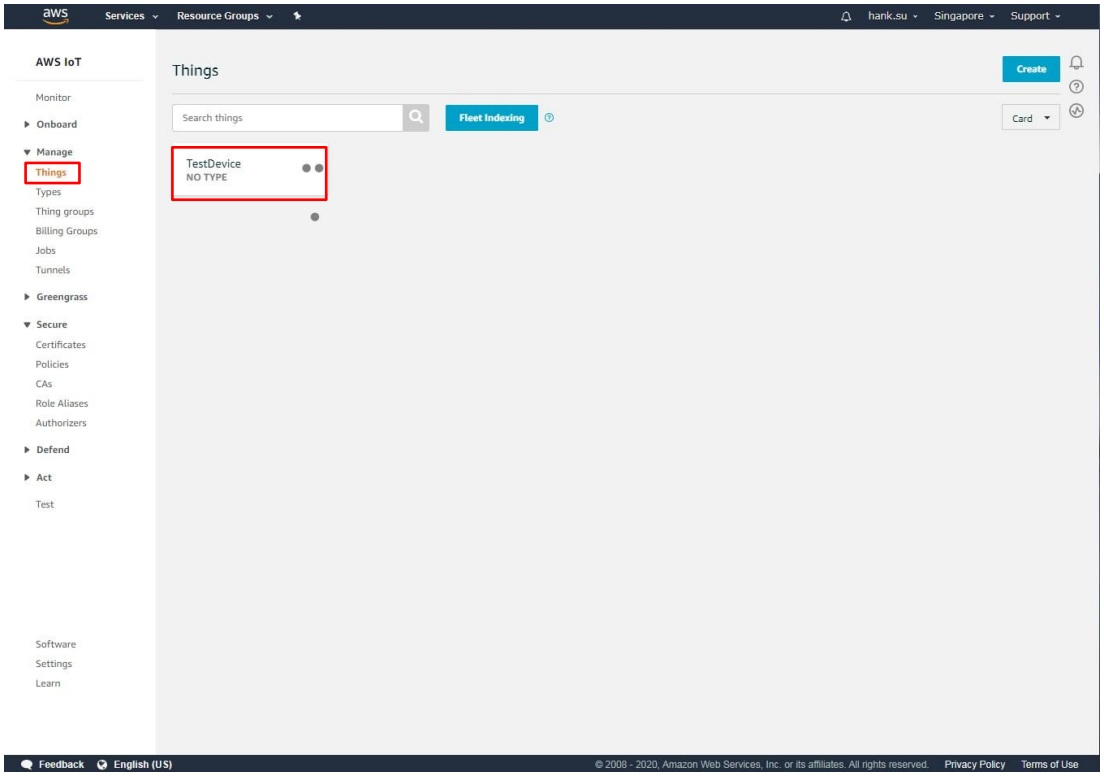


NOTE – this policy grants unrestricted access for all iot operations, and is to be used only in a development environment. For non-dev environments, all devices in your fleet must have credentials with privileges that authorize intended actions only, which include (but not limited to) AWS IoT MQTT actions 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

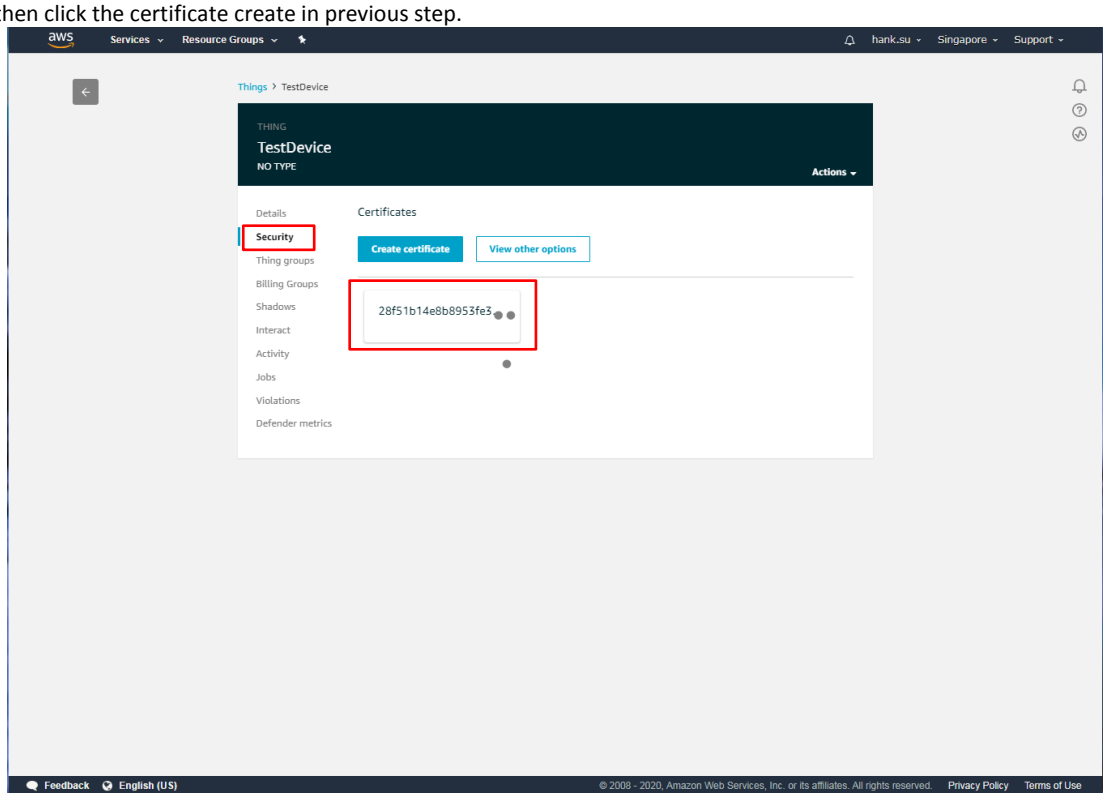


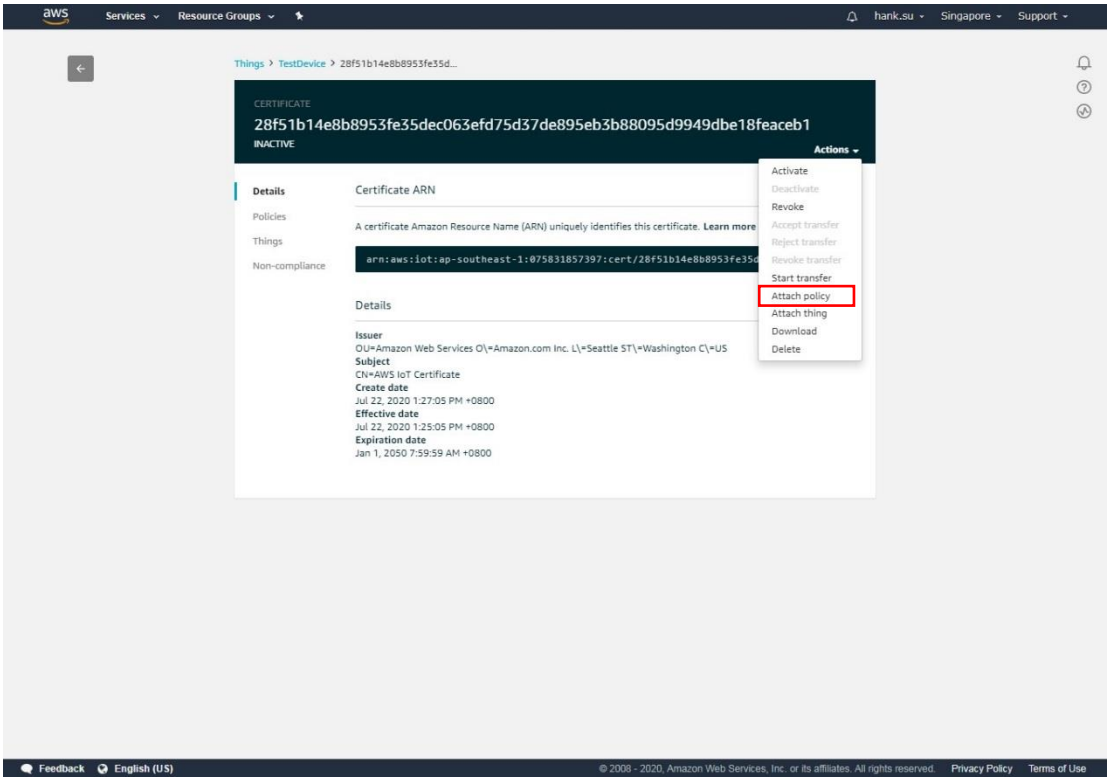
2.3 Attach Policy

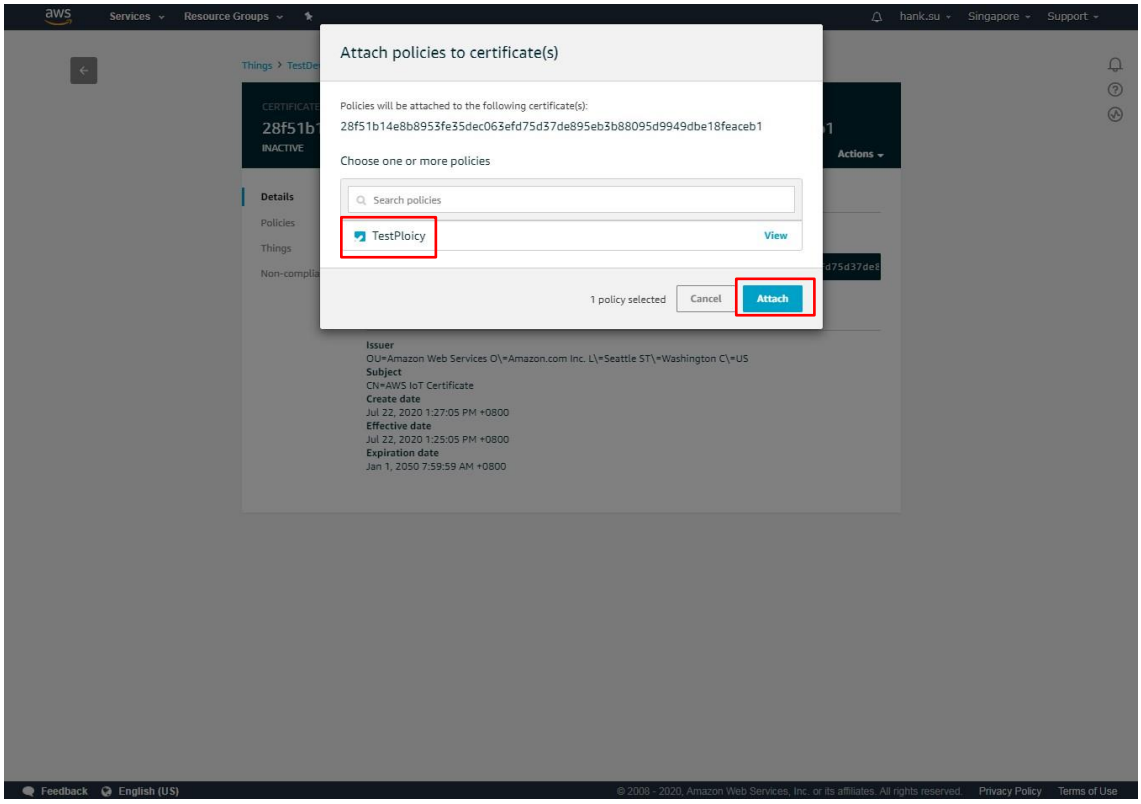
The last step to configuring the device is attaching a policy. To attach a policy to new device, navigate to Manage -> Things. Then click on the device which was created.



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



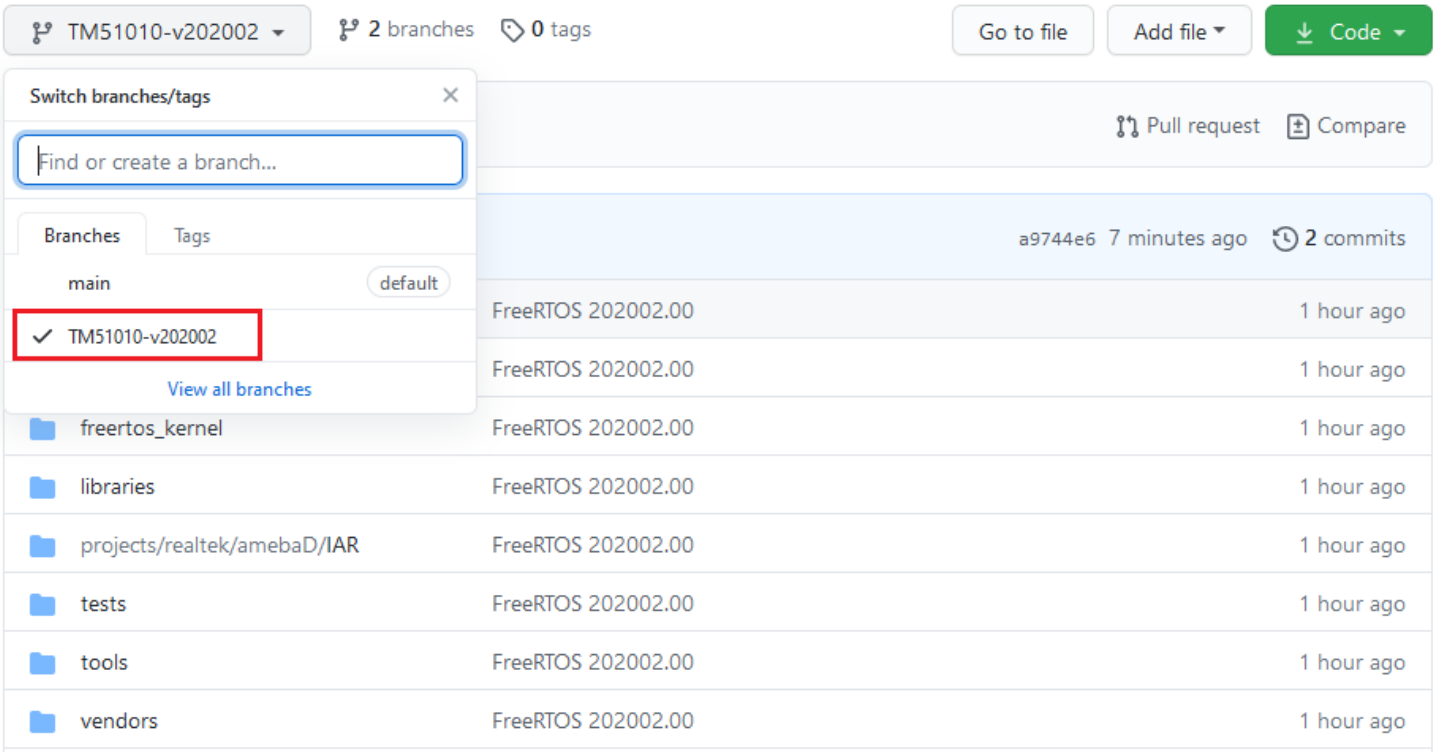




3 Configure TM51010 Amazon FreeRTOS

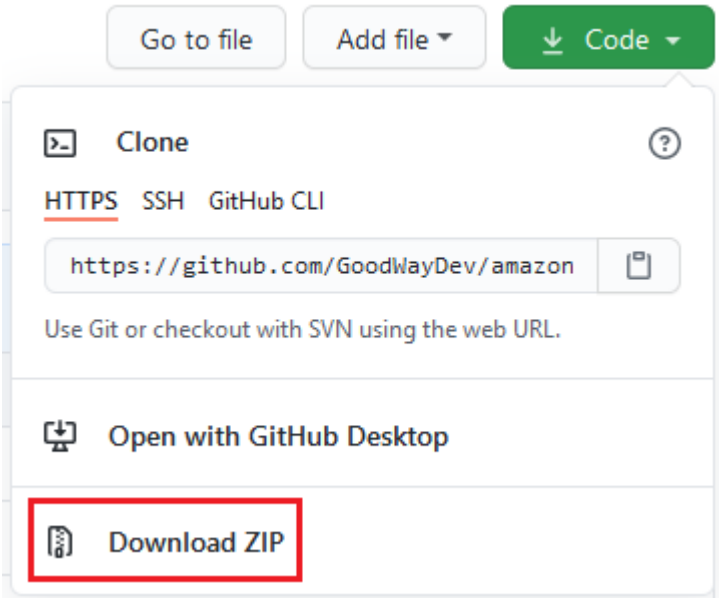
3.1 Download Source Code from github

Open source link: https://github.com/GoodWayDev/amazon-freertos and select main to get newest source code. TM51010 also support v202002, please find source in “TM51010-v202002” branch.



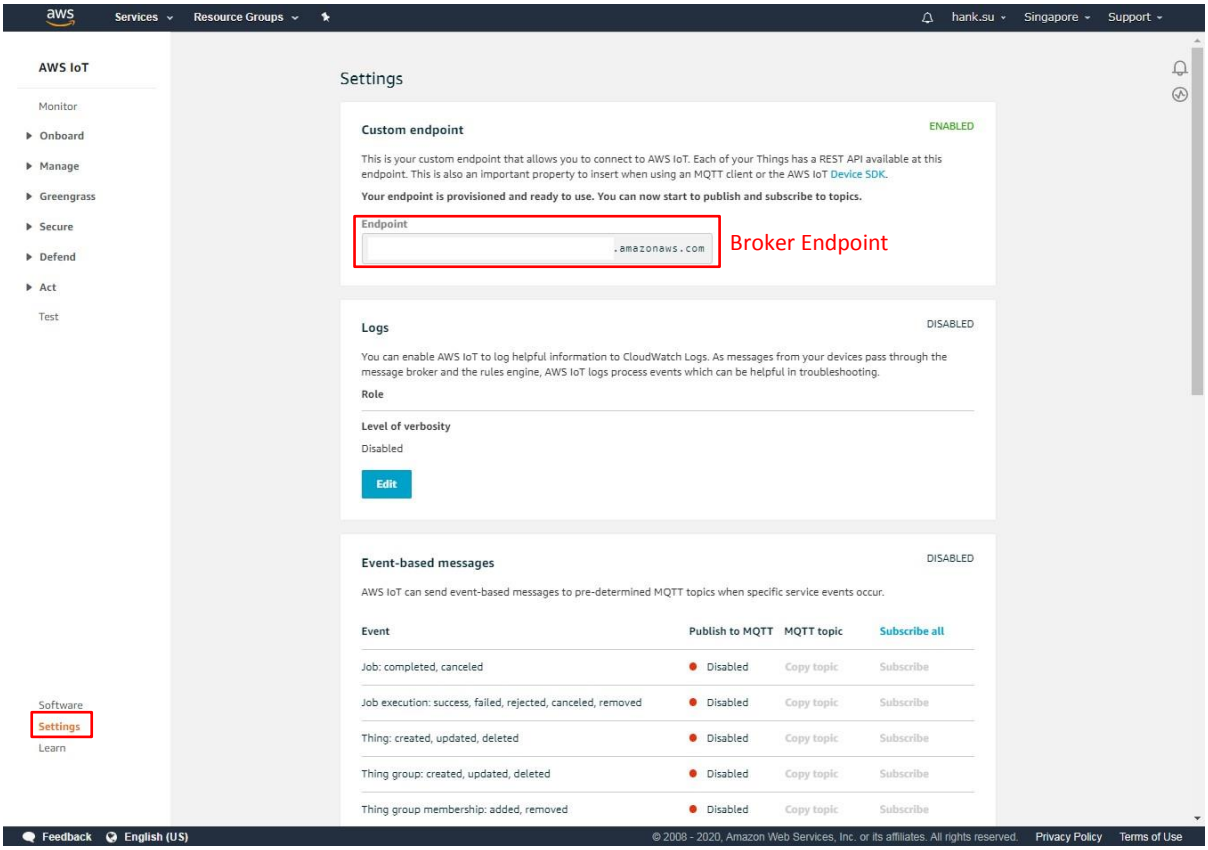
3.1.1 Cloning a repository by Download ZIP

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

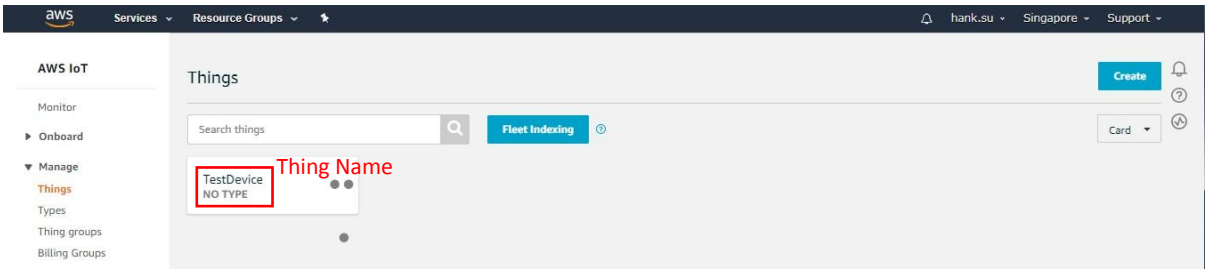


For more information, please refer "Cloning a repository from GitHub to GitHub Desktop."

3.2 Get Broker Endpoint by AWS IoT Core



3.3 Get Thing Name



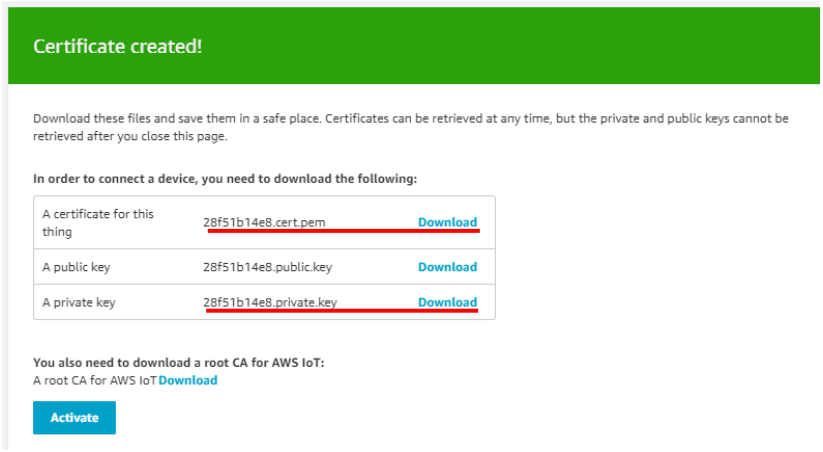
3.4 Setup IoT Core Information with TM51010 Amazon FreeRTOS

Setup BROKER_ENDPOINT, THING_NAME, WIFI_SSID, PASSWORD in “amazon-freertos/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/ demos/include/aws_clientcredential_keys.h” by xxxxxxxx-certifiacte.pem and xxxxxxxx-private.pem.key.



It can done by 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"\
"                                                                \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"\
"                                                                \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 TM51010

Find platform_opts.h in 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 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 TM51010 Amazon FreeRTOS

4 Compile TM51010 Amazon FreeRTOS

4.1 IAR Build Environment Setup

Currently the amazon-freertos of TM51010 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/GoodWayDev/amazon-freertos)
- TM51010 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 **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 **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.

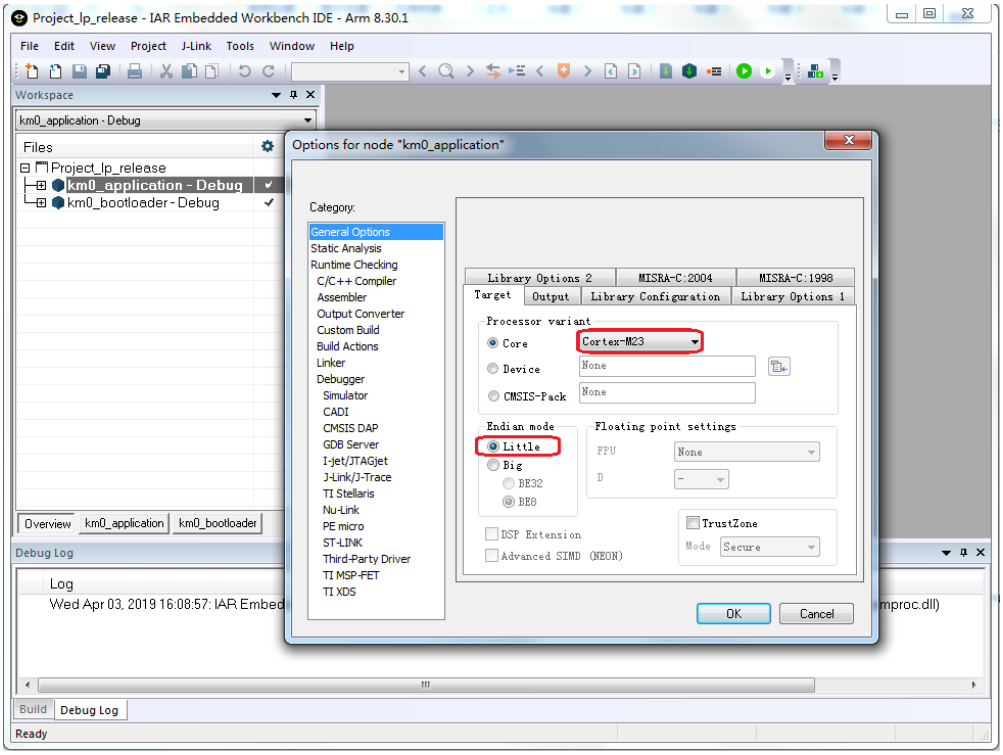


Fig 4-1 KM0 processor options

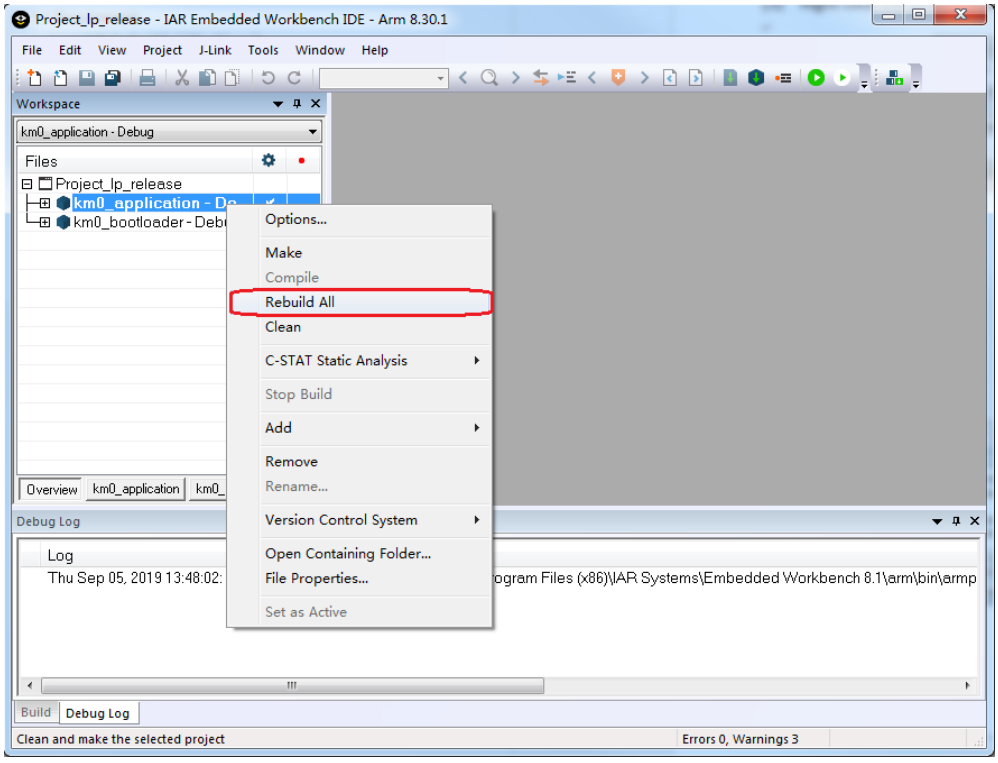
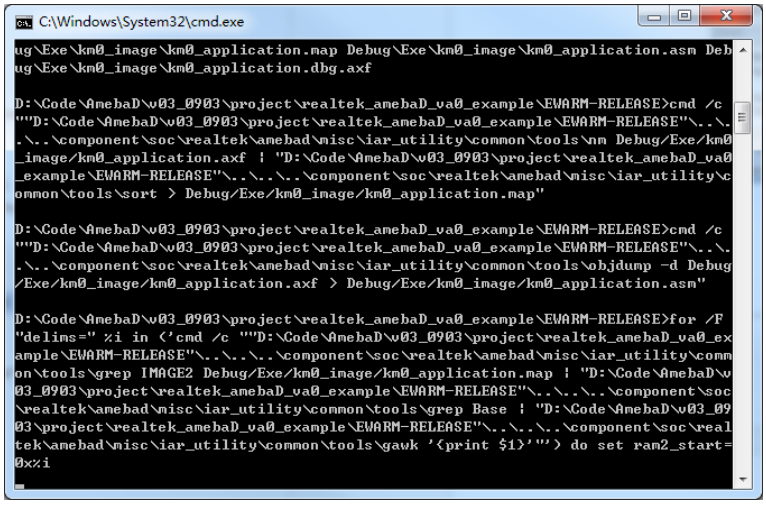


Fig 4-2 Building KM0 project

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 **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 **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.

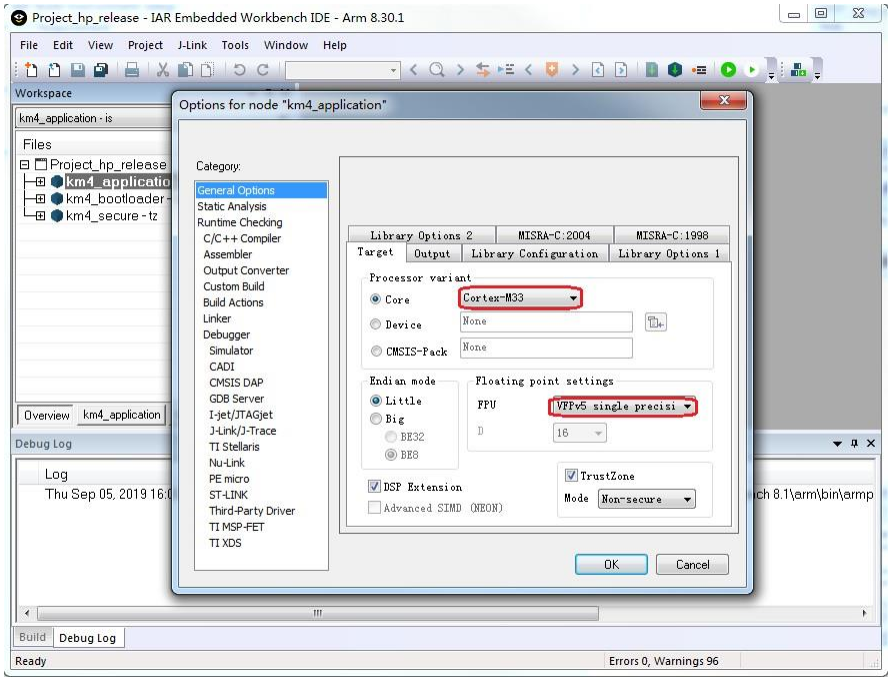


Fig 4-3 KM4 processor options

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

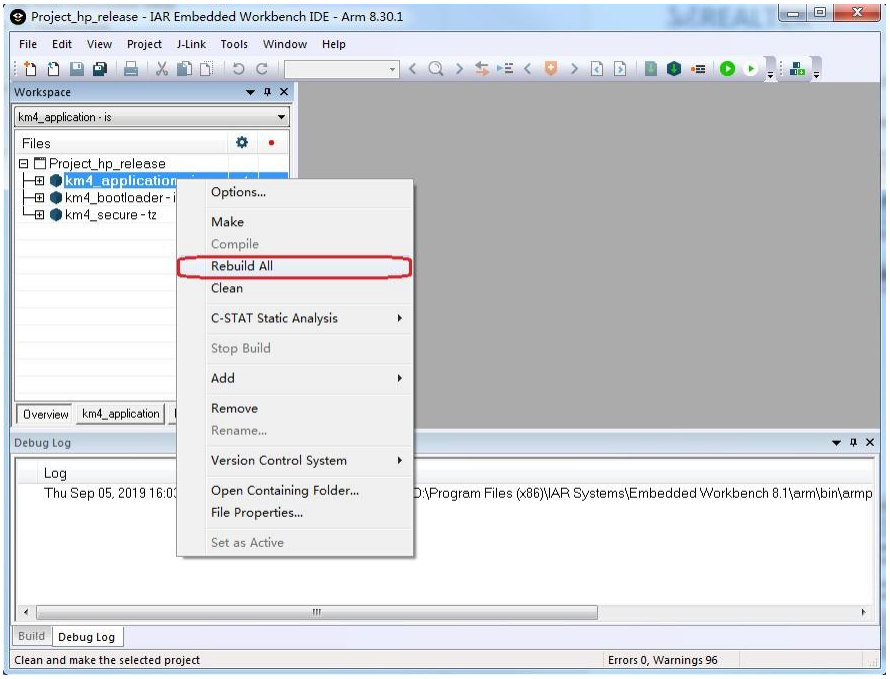
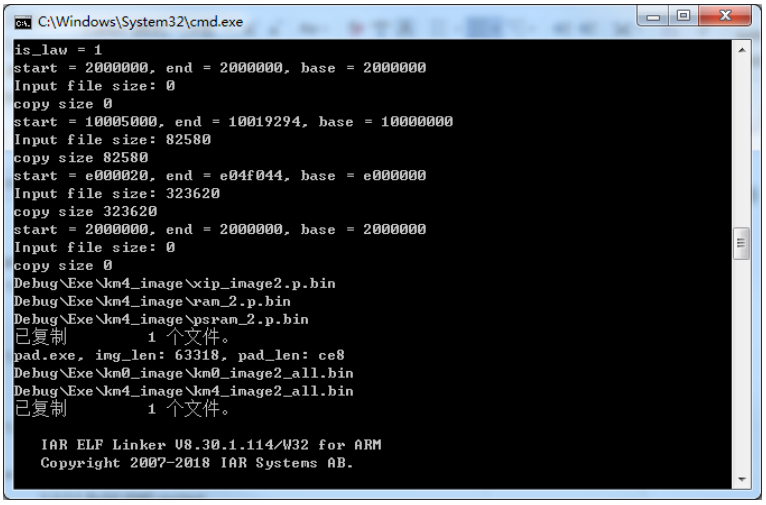


Fig 4-4 Building KM4 project

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 **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 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, Make sure Chip Select is AmebaD(8721D).

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

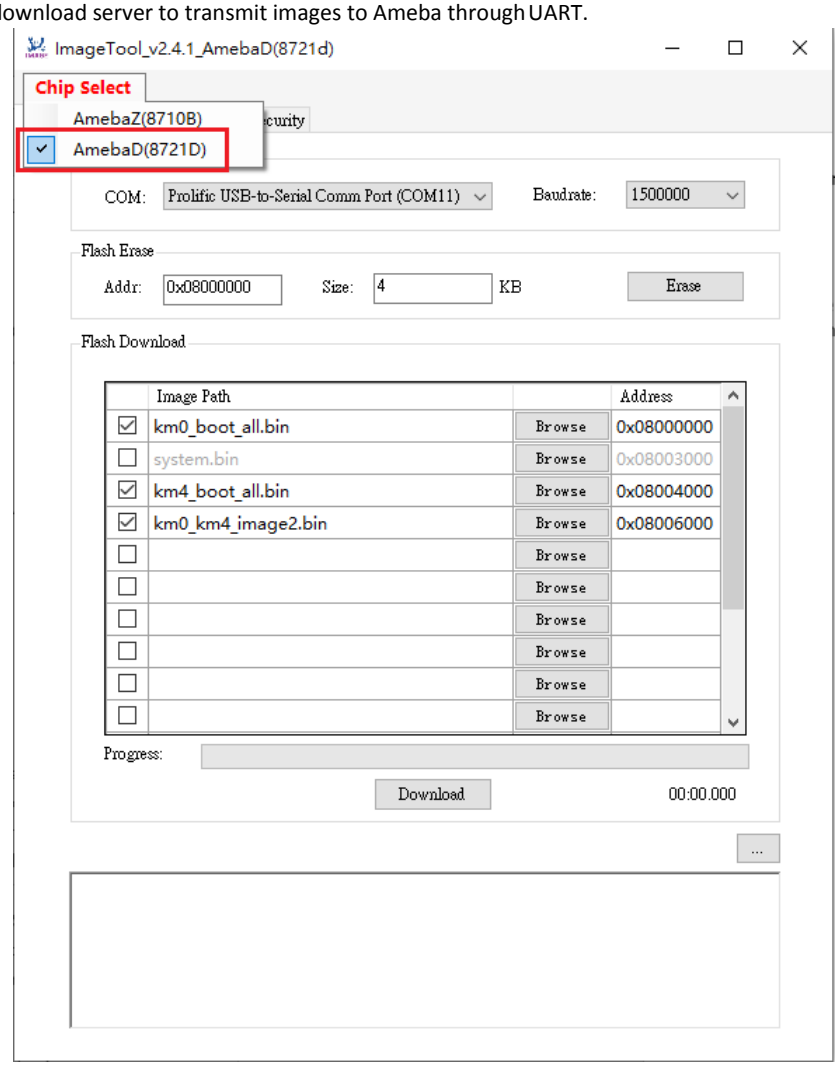


Fig 5-1 ImageTool UI

5.2 Environment Setup

5.2.1 Hardware Setup

The hardware setup is shown in Fig 5-2.

Note: FT232 USB to UART dongle must be used.

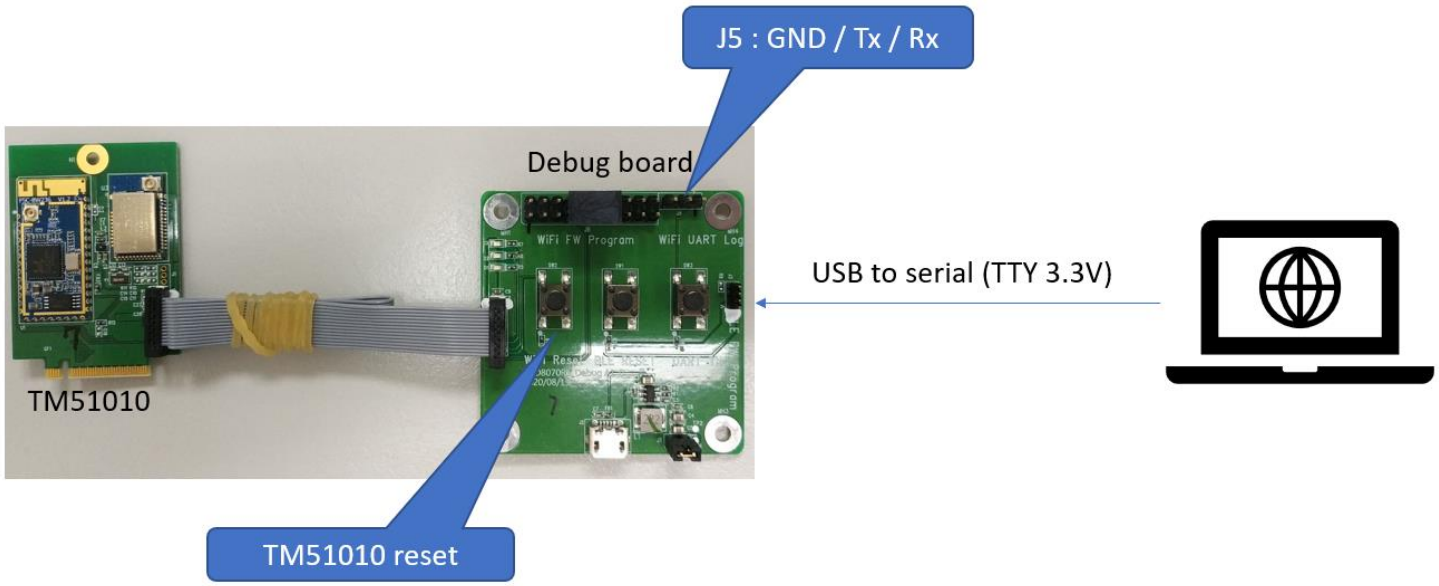


Fig 5-2 Hardware setup

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(8721D).

(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.

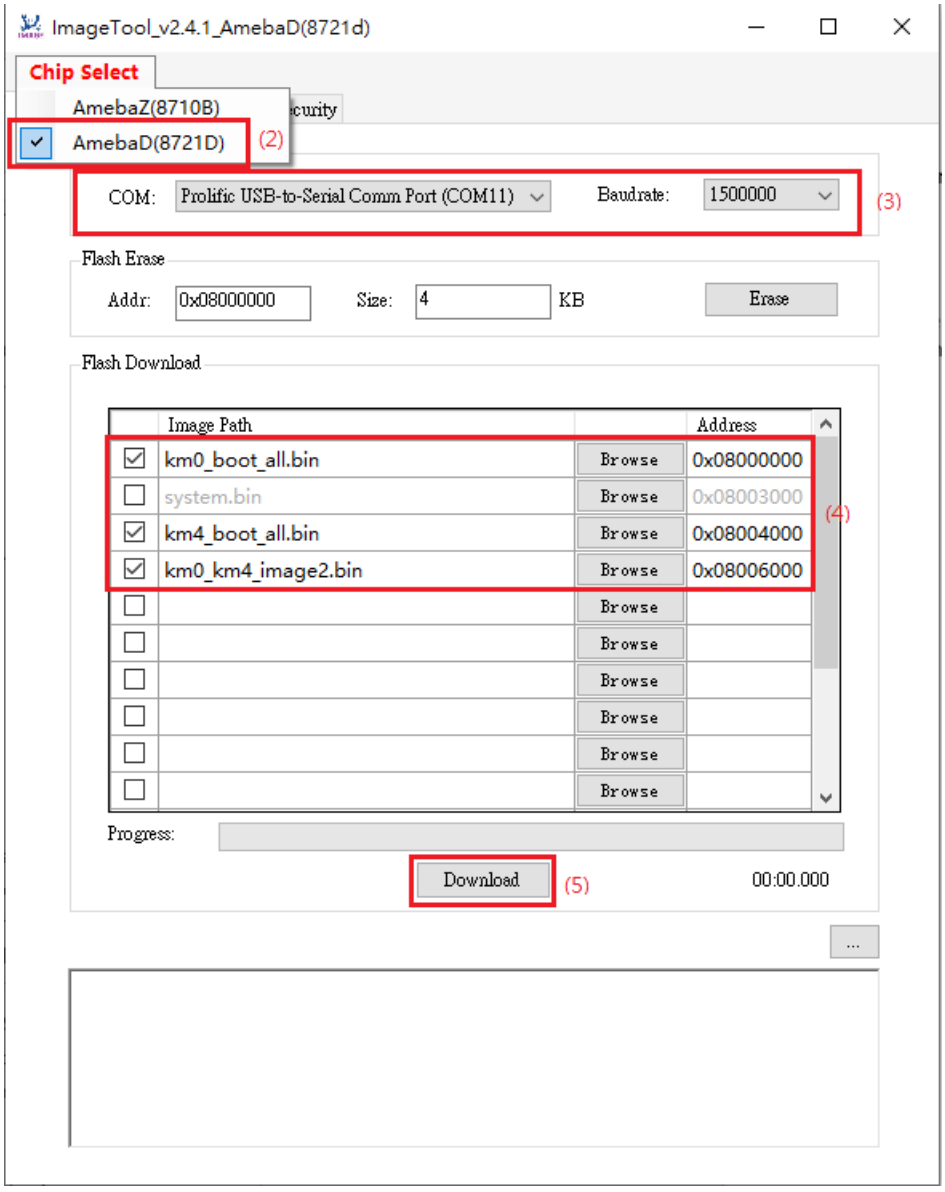


Fig 5-3 ImageTool 'Download' tabpage setting

6 MQTT Demo

6.1 Get Device Log

Install Tera Term to get device log

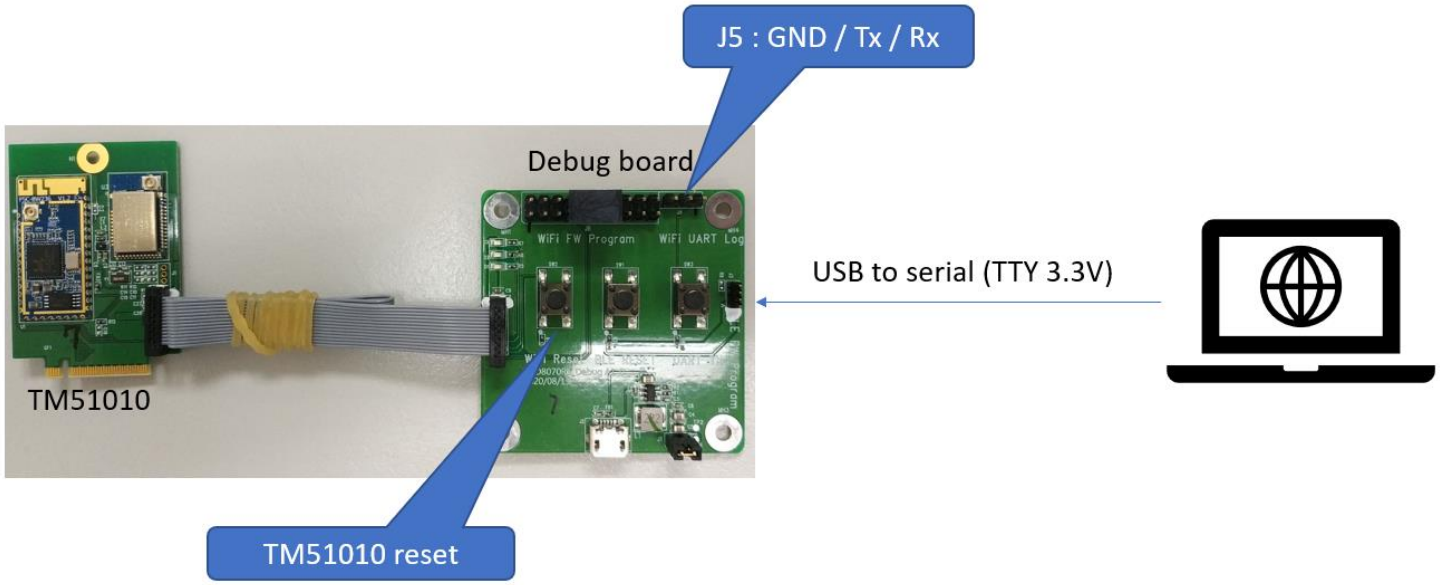
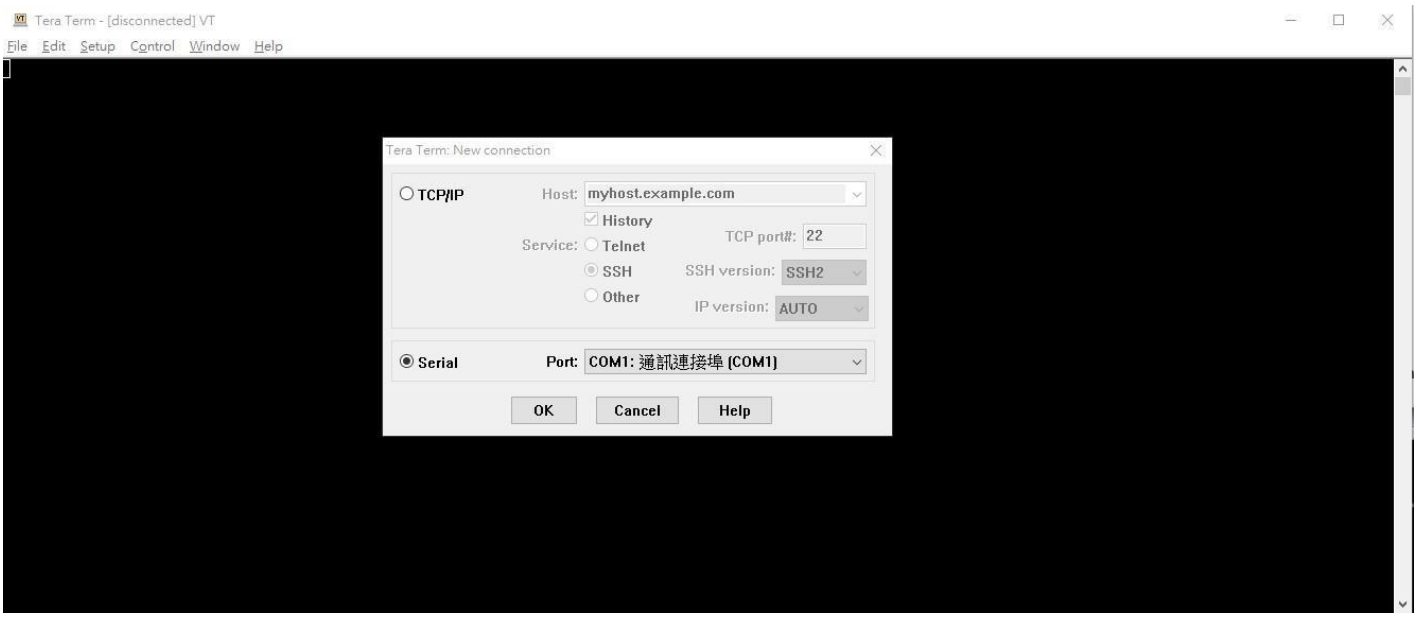


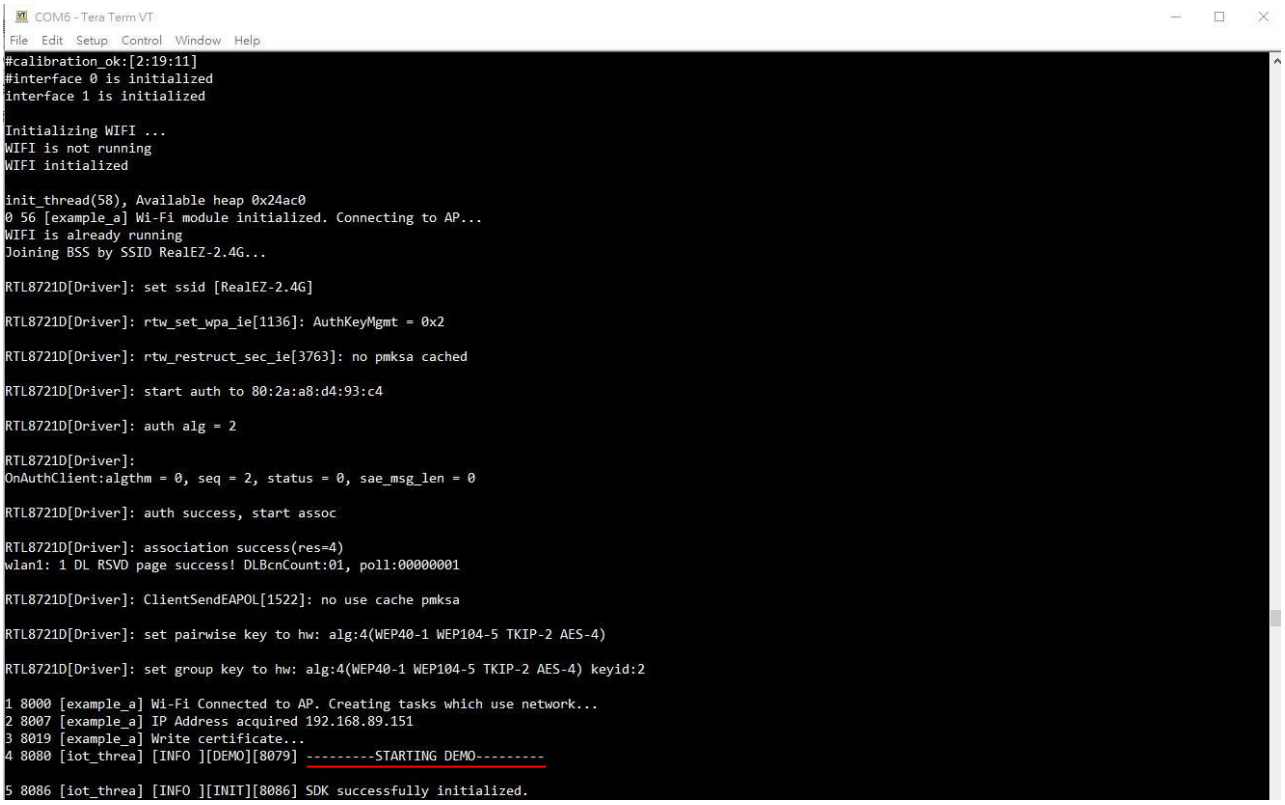
Fig 6-1 Hardware setup

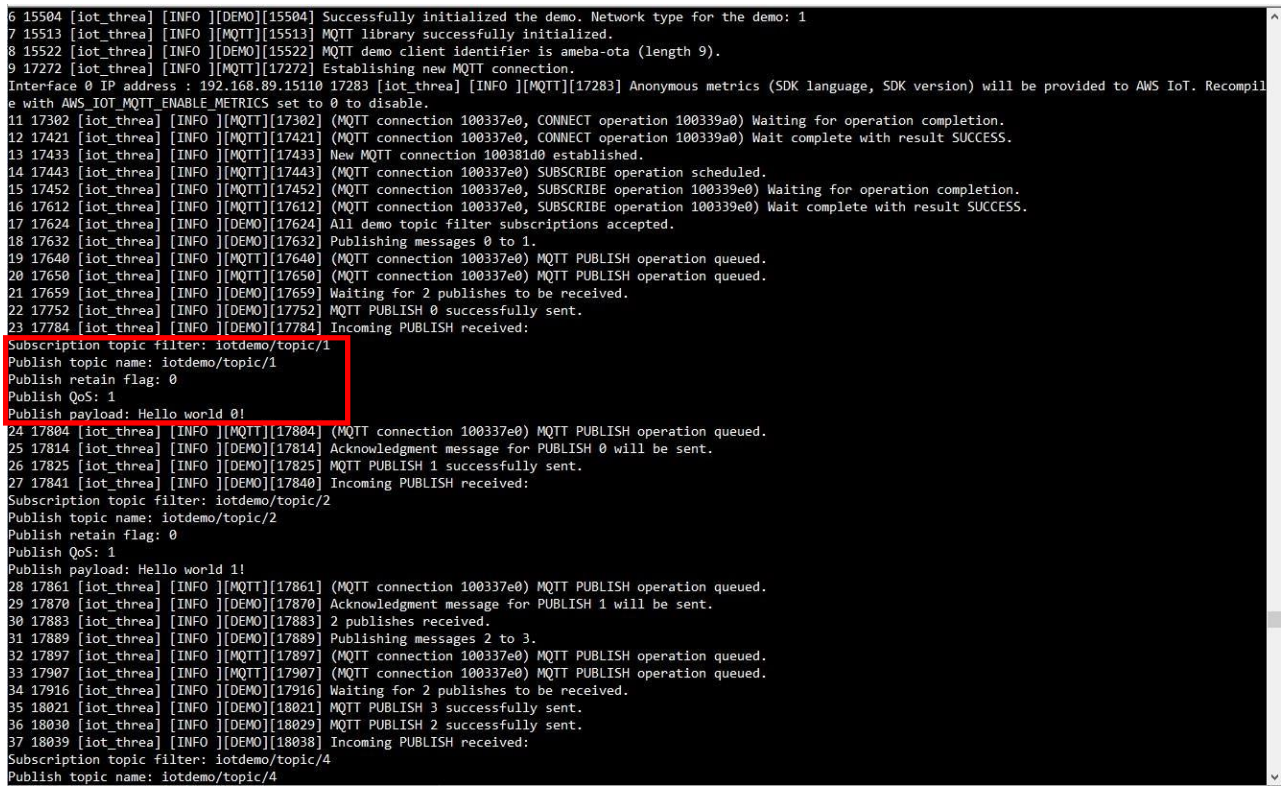
The serial port is same with ImageTool that get from 5.3.1.1 step (3)



6.2 Run MQTT Demo

Default setting of SDK are enable MQTT demo. Once the TM51010 Module has rebooted, the application will automatically start run MQTT demo and communicate to IoT Core.





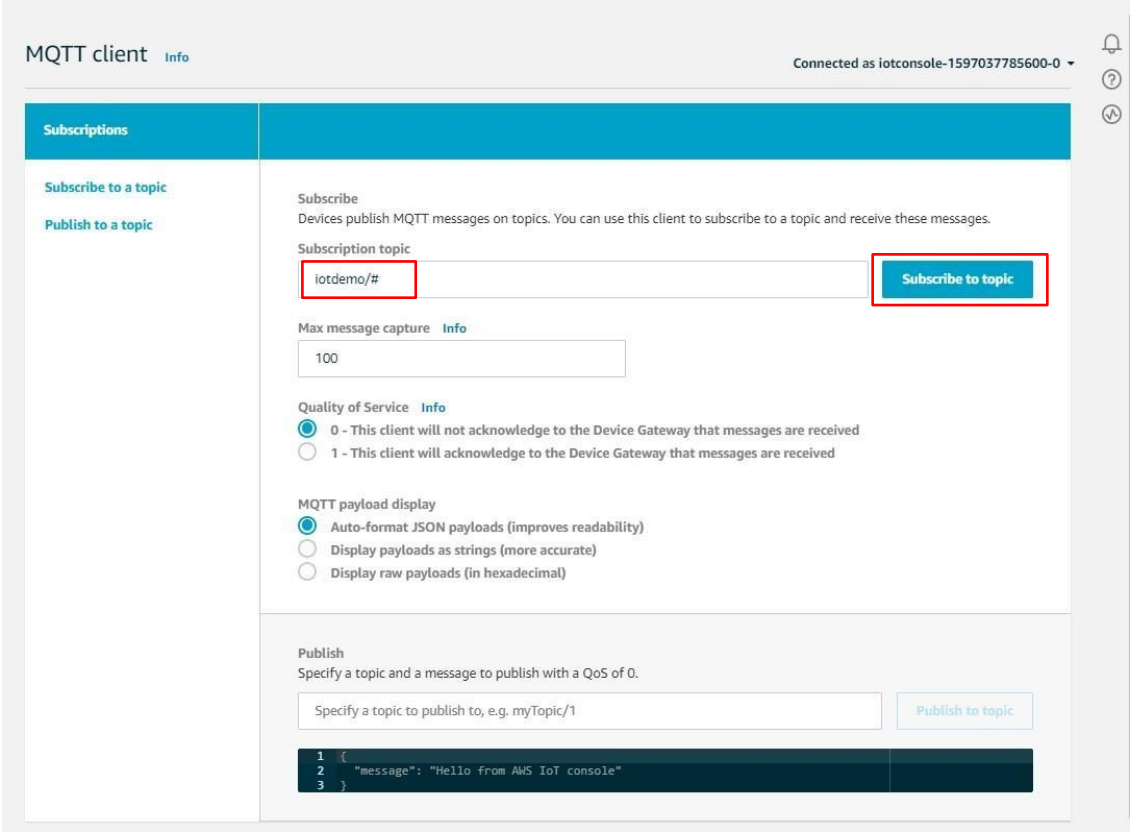
```
Publish payload: Hello world 16!
132 19827 [iot_threa] [INFO ][MQTT][19827] (MQTT connection 100337e0) MQTT PUBLISH operation queued.
133 19837 [iot_threa] [INFO ][DEMO][19837] Acknowledgment message for PUBLISH 16 will be sent.
134 19851 [iot_threa] [INFO ][DEMO][19851] 2 publishes received.
135 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-------
```

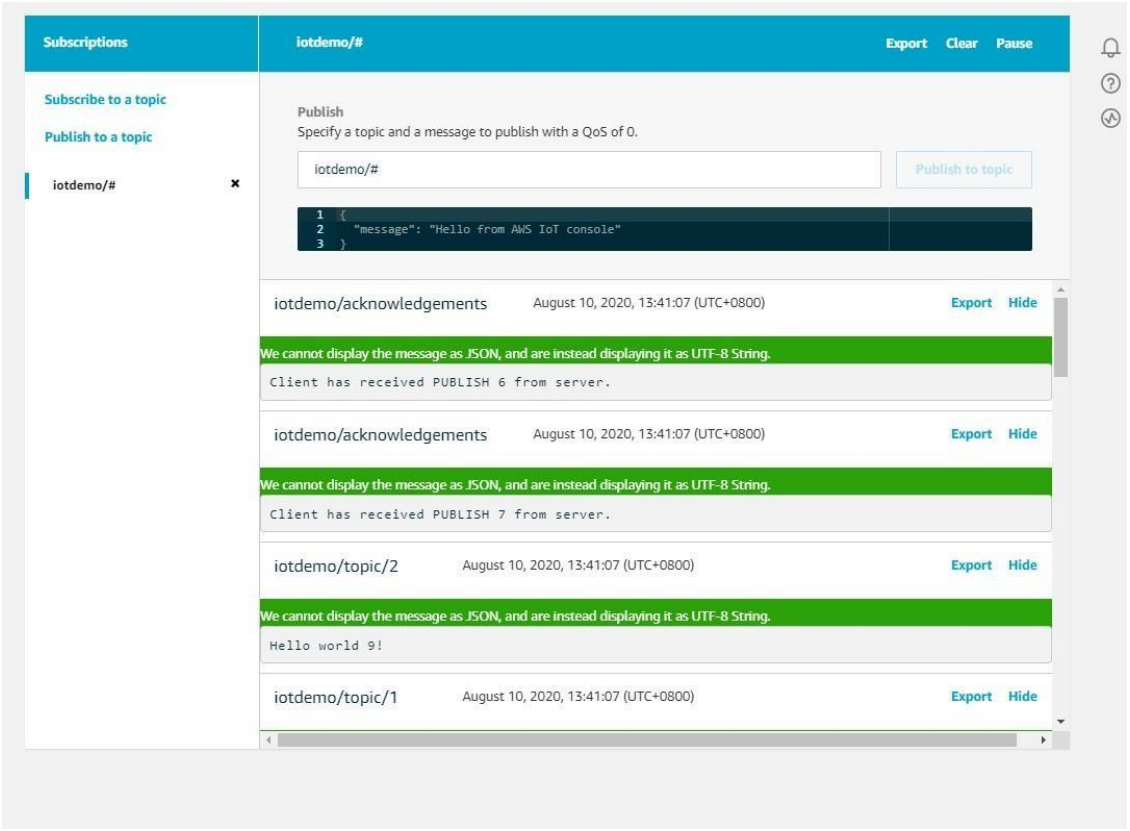
Monitor connection summary.

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.





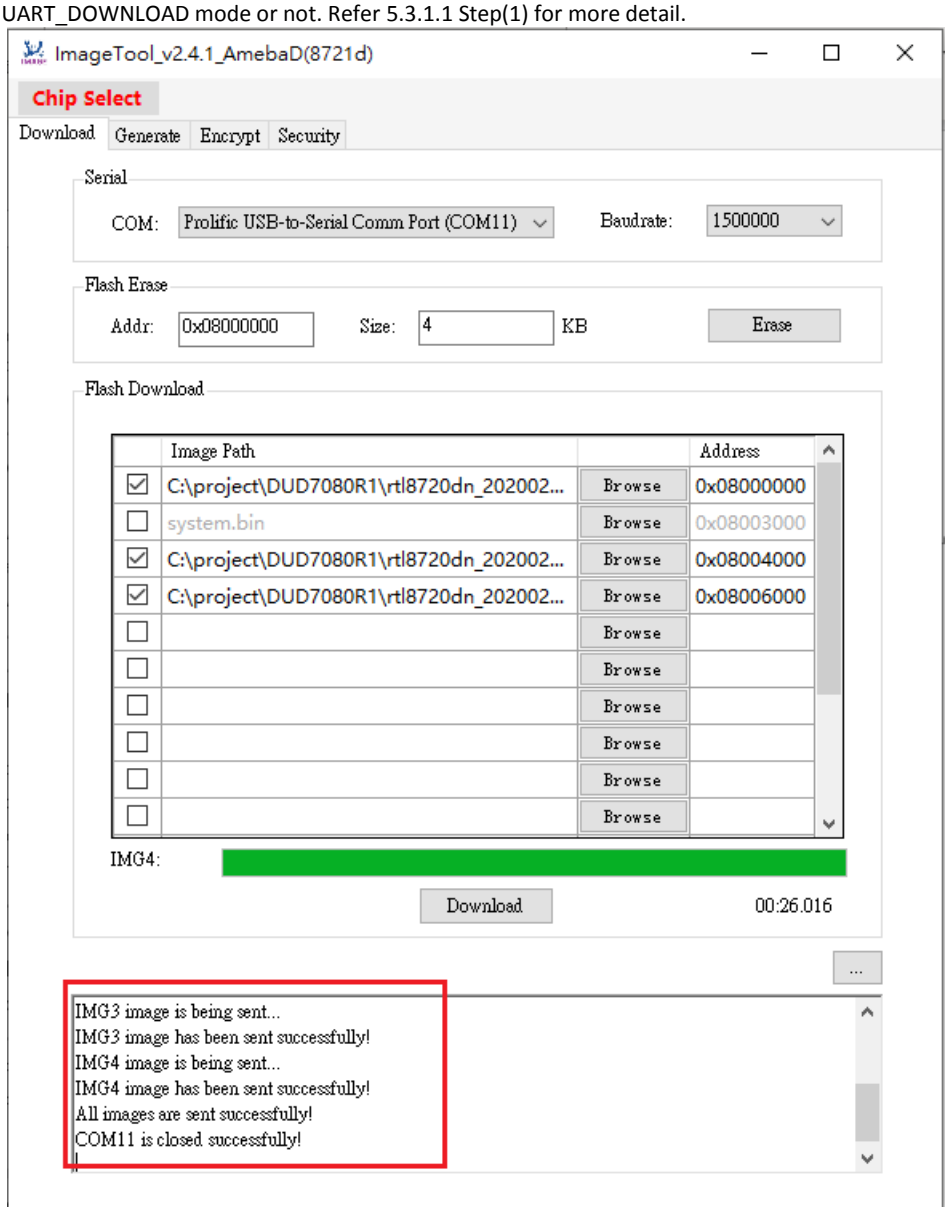
7 Troubleshooting

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.

7.1 Flashloader download fail

Please check device in UART_DOWNLOAD mode or not. Refer 5.3.1.1 Step(1) for more detail.



7.2 ERROR: Invalid Key

Please check **WIFI_SSID** and **WIFI_PASSWORD** in in amazon-freertos/demos/include/aws_clientcredential.h

```
Enter SSID for Soft AP started
3 1098 [example_a] Wi-Fi configuration successful.
4 1108 [iot_threa] [INFO ][DEMO][1108] ---------STARTING DEMO---------

5 1115 [iot_threa] [INFO ][INIT][1115] SDK successfully initialized.

LwIP_DHCP: dhcp stop.
Deinitializing WIFI ...
WIFI deinitialized
Initializing WIFI ...
WIFI initialized

Joining BSS by SSID ...

ERROR:Invalid Key
ERROR: Can't connect to AP
Joining BSS by SSID ...

ERROR:Invalid Key
ERROR: Can't connect to AP
Joining BSS by SSID ...
```

7.3 Failed to establish new MQTT connection

Please check **clientcredentialMQTT_BROKER_ENDPOINT** in amazon-freertos/blob/master/demos/include/aws_clientcredential.h

```
6 12508 [iot_threa] [INFO ][DEMO][12508] Successfully initialized the demo. Network type for the demo: 1
7 12517 [iot_threa] [INFO ][MQTT][12517] MQTT library successfully initialized.
8 12524 [iot_threa] [INFO ][DEMO][12524] MQTT demo client identifier is ameba-ota (length 9).
9 12624 [iot_threa] [ERROR][NET][12624] Failed to resolve [redacted].amazonaws.com.
10 12934 [iot_threa] [ERROR][MQTT][12934] Failed to establish new MQTT connection, error NETWORK ERROR.
11 12943 [iot_threa] [ERROR][DEMO][12943] MQTT CONNECT returned error NETWORK ERROR.
12 12951 [iot_threa] [INFO ][MQTT][12950] MQTT library cleanup done.
13 12957 [iot_threa] [ERROR][DEMO][12957] Error running demo.
Interface 0 IP address : 192.168.90.185
LwIP_DHCP: dhcp stop.
Deinitializing WIFI ...
14 13094 [iot_threa] [INFO ][INIT][13094] SDK cleanup done.
15 13099 [iot_threa] [INFO ][DEMO][13099] -------DEMO FINISHED-------
```

7.4 TLS_Connect fail

Please check **keyCLIENT_CERTIFICATE_PEM** and **keyCLIENT_PRIVATE_KEY_PEM** in ambd_amazon-freertos/blob/master/demos/include/aws_clientcredential_keys.h

```
8 13501 [iot_threa] [INFO ][DEMO][13501] Successfully initialized the demo. Network type for the demo: 1
9 13511 [iot_threa] [INFO ][MQTT][13511] MQTT library successfully initialized.
10 13518 [iot_threa] [INFO ][DEMO][13518] MQTT demo client identifier is ameba-ota (length 9).
11 20102 [iot_threa] ERROR: Private key not found. 12 20107 [iot_threa] TLS_Connect fail (0x7d4, [redacted].amazonaws.com)
13 20115 [iot_threa] [ERROR][NET][20115] Failed to establish new connection. Socket status: -1.
14 20424 [iot_threa] [ERROR][MQTT][20424] Failed to establish new MQTT connection, error NETWORK ERROR.
15 20433 [iot_threa] [ERROR][DEMO][20433] MQTT CONNECT returned error NETWORK ERROR.
16 20441 [iot_threa] [INFO ][MQTT][20441] MQTT library cleanup done.
17 20447 [iot_threa] [ERROR][DEMO][20447] Error running demo.
Interface 0 IP address : 192.168.90.185
LwIP_DHCP: dhcp stop.
Deinitializing WIFI ...
18 20586 [iot_threa] [INFO ][INIT][20586] SDK cleanup done.
19 20591 [iot_threa] [INFO ][DEMO][20591] -------DEMO FINISHED-------
```