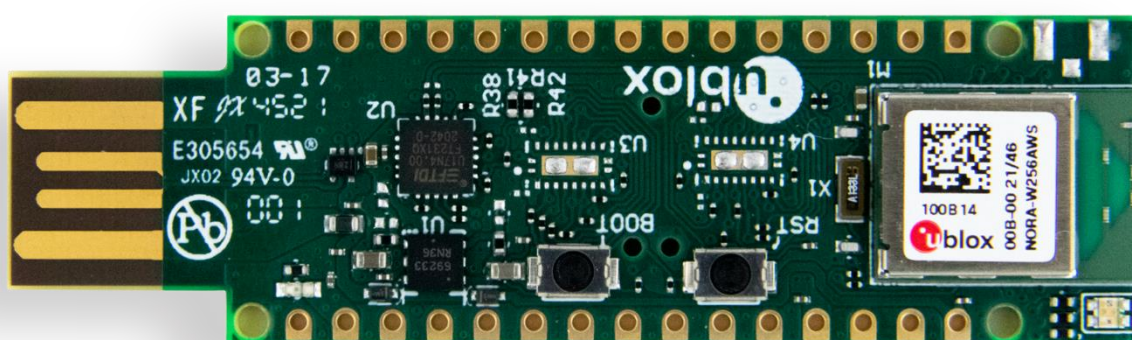


USB-NORA-W256AWS

AWS IoT ExpressLink Multiradio development kit
User guide



Abstract

This document describes how to set up and use the USB-NORA-W256AWS evaluation board for prototyping NORA-W256AWS modules with extremely low-powered Internet of Things (IoT) applications. It provides instructions for getting started with the evaluation board and includes a technical overview of the modules.

Document information

Title	USB-NORA-W256AWS		
Subtitle	AWS IoT ExpressLink multiradio development kit		
Document type	User guide		
Document number	UBX-21045749		
Revision and date	R04	23-May-2023	
Disclosure restriction	C1-Public		

This document applies to the following products:

Product name	Ordering code	Type number	IN/PCN reference
USB-NORA-W256AWS	USB-NORA-W256AWS	USB-NORA-W256AWS-00	N/A

u-blox or third parties may hold intellectual property rights in the products, names, logos and designs included in this document. Copying, reproduction, or modification of this document or any part thereof is only permitted with the express written permission of u-blox. Disclosure to third parties is permitted for clearly public documents only.

The information contained herein is provided "as is". No warranty of any kind, either express or implied, is made in relation to the accuracy, reliability, fitness for a particular purpose or content of this document. This document may be revised by u-blox at any time. For most recent documents, please visit www.u-blox.com.


Copyright © u-blox AG.

Contents

Document information	2
Contents	3
1 Introduction	4
1.1 NORA-W2 AWS series stand-alone modules.....	4
1.2 USB-NORA-W256AWS evaluation board	5
1.3 Kit includes	5
1.4 User provided items.....	6
1.5 Key features	6
1.5.1 Pre-provisioned with AWS cloud	6
1.5.2 Simple integration with stateless commands	6
1.5.3 Wi-Fi 802.11b/g/n.....	6
1.5.4 Enhanced security features	6
2 Getting started	7
2.1 Using a computer as a host	7
2.2 Verifying evaluation board connection.....	8
2.3 Running the “Quick Connect” demo application	8
3 Registering an AWS IoT ExpressLink in your development account	10
3.1 Creating and Configuring the Thing	10
3.2 Setting up and connecting to Wi-Fi	11
3.3 Validating the onboarding process	11
4 Connecting and interacting with AWS cloud.....	12
4.1 Connecting	12
4.2 Send data to AWS cloud.....	12
4.3 Receiving data and commands from AWS cloud	13
5 Over-The-Air updates for AWS IoT ExpressLink	15
5.1 Acquiring software updates	15
5.2 Prerequisites.....	15
5.3 Create an OTA update job in AWS IoT	15
5.4 Monitoring and applying OTA jobs	16
Related documentation.....	17
Revision history	17
Contact.....	17

1 Introduction

The USB-NORA-256AWS evaluation kit comprises a small 16x22 mm evaluation board that conveniently exposes the supported interfaces of the NORA-W256AWS module, which is mounted directly on the board. The board has a USB form factor and plugs directly into the USB port of your computer.

 This document explains how to setup the USB-NORA-W256AWS evaluation kit that includes the USB-NORA-W256AWS evaluation board and NORA-W256AWS module. Note that the module stores a pre-provisioned hardware root of trust which provides the necessary credentials to connect to a default AWS IoT ExpressLink staging account. These credentials include a unique identifier (UID) of the module, a key pair (public and private), and certificate signed by a Certificate Authority shared with AWS.

1.1 NORA-W2 AWS series stand-alone modules

NORA-W2 AWS stand-alone, multiradio modules integrating a powerful 32-bit, dual core microcontroller unit (MCU) and radio for wireless communication. The modules support either an internal antenna (NORA-W256AWS) or external antennas connected through a dedicated antenna pin (NORA-W251AWS). The radio provides support for Wi-Fi 802.11 b/g/n in the 2.4 GHz ISM band.

The embedded AWS IoT ExpressLink compliant software includes secured certificates that are pre-flashed in the modules. This allows the modules to provide “out of the box” connectivity with Amazon Web Services (AWS) without any additional effort from the customer. NORA-W2 AWS modules also support secure Host and Firmware Over the Air (OTA) upgrades to sustain a high level of quality. Control and data communication is handled through the module with stateless AT-commands over a serial interface.

The evaluation kit supports modification of the endpoint to your development account. When delivered from production, the module this endpoint is pre-defined to the u-blox staging account and is transferred to the customer AWS account at a later stage.

NORA-W2 include a wireless MCU, flash memory, and crystal. It also includes components for antenna matching, filtering, and decoupling – making it a very compact standalone multiradio module. The module is designed with

secure boot, which ensures the module boots up only in the presence of authenticated software. The small size and the embedded security capabilities make NORA-W2 AWS modules ideal for critical IoT applications where security is important. Intended applications include consumer products, telematics, low power sensors, connected factories, connected buildings (appliances and surveillance), point-of-sales, and health devices.

NORA-W2 AWS modules are globally certified, which significantly reduces the time to market for end products. The professional grade modules support an extended temperature range of -40°C to $+85^{\circ}\text{C}$. They are qualified according to u-blox Qualification Policy, based on AEC-Q104.

See also the NORA-W2 series data sheet [\[1\]](#).

1.2 USB-NORA-W256AWS evaluation board

USB-NORA-W256AWS is a versatile development platform that allows quick prototyping of a variety of extremely low-powered Internet of Things (IoT) applications. The radio provides support for Bluetooth Low Energy 5.0¹ and Wi-Fi 802.11 b/g/n in the 2.4 GHz ISM band.

USB-NORA-W256AWS includes an internal PCB antenna and NORA-W256AWS module that is pre-flashed with AWS IoT ExpressLink compliant software.

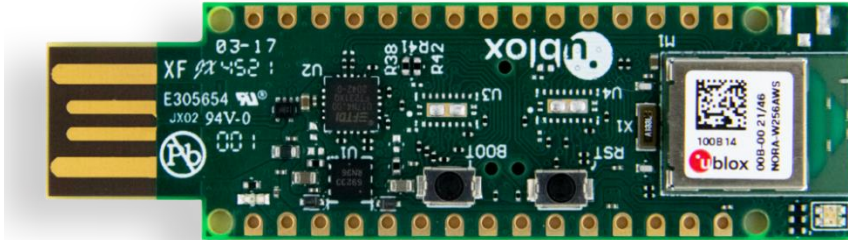


Figure 1: USB-NORA-W256AWS evaluation board

1.3 Kit includes

The kit includes the USB-NORA-W256AWS evaluation board with USB connector. The NORA-W256AWS module is mounted on the evaluation board.

With a USB Type A form factor, the USB-NORA-W256AWS evaluation board is inserted directly into the PC host. For more information, see also the [USB-NORA-W2 product web page](#).

¹ Bluetooth Low Energy is currently not supported in the AWS IoT ExpressLink AT command manual [4].

1.4 User provided items

Item	Description
PC	Computer with USB type A port
Wi-Fi Access Point/Router	Wi-Fi Access Point with public internet access. WPA, WPA2 or WPA3 Wi-Fi authentication must be supported.

Table 1: User provided items

1.5 Key features

1.5.1 Pre-provisioned with AWS cloud

The NORA-W256AWS module is provisioned with securely stored keys and certificates for secure connection to the AWS cloud during production. No module setup, apart from the configuration of Wi-Fi credentials, is required to connect the end-product to the AWS cloud.

1.5.2 Simple integration with stateless commands

Communication from the host to the module is performed using a simple and easy-to-use AT-command set over a serial interface. For details about the supported commands, see also the AWS AT-command information on the AWS IoT ExpressLink programmer's guide [\[4\]](#).

1.5.3 Wi-Fi 802.11b/g/n

USB-NORA-W256AWS communicates with the AWS Cloud over Wi-Fi 4.

1.5.4 Enhanced security features

NORA-W2² module series contain a multistage secure boot that ensures that the running software, as well as the hardware, is authentic. All provisioned certificates and keys are stored in the secure memory of the module. Keys cannot be read or modified externally. NORA-W2 module series support MQTT TLS 1.2 and Wi-Fi WPA, WPA2 and WPA3 authentication protocols for internet communication. The prototype kits hosting the module have security limitations³.

² NORA-W2 refers to the module mounted on the development board

³ The NORA-W2 module included in the prototype version of the development board has temporary certificates and are not securely stored

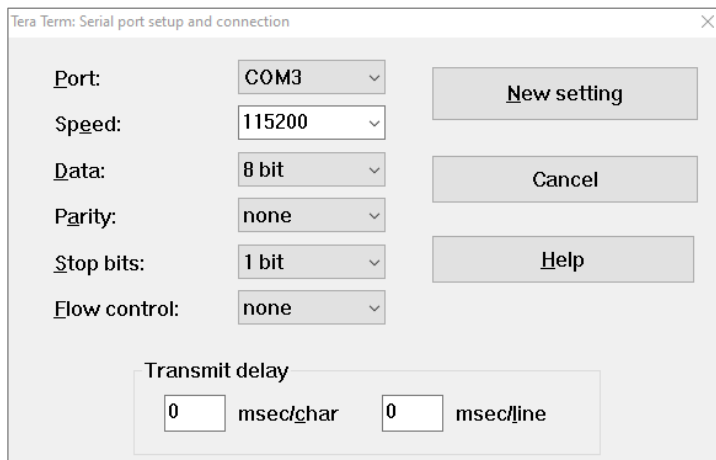
2 Getting started

2.1 Using a computer as a host

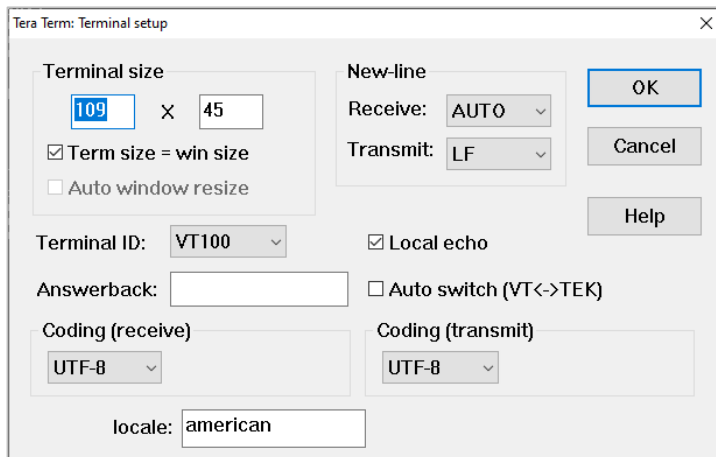
Use the following procedure to set up the USB-NORA-W256AWS evaluation board using a computer as a host. Any operating system is supported that supports running a terminal application.

Depending on your computer configuration, it may be necessary to download and install USB-to-Serial drivers to enable the computer to perform serial communication over USB. The drivers and driver installation instructions can be found [here](#).

1. Connect USB-NORA-W256 to the host machine.
2. Open a terminal application on your host machine, like TeraTerm for Windows or CoolTerm for Mac.
3. Select the COMPORT corresponding to the evaluation kit. Consult the documentation for your operating system to find out how to detect the port used by the evaluation kit.
4. Configure the terminal application as follows:
 - Baudrate: 115200
 - Bits: 8
 - Parity: None
 - Stop: 1
 - Flow control: None
 - Local Echo: Yes



5. Enable the **Local echo** option on the terminal setup.



2.2 Verifying evaluation board connection

Verify that you have a working connection to the evaluation board:

1. Open the terminal window
2. Type “AT” and press return.
3. Verify that a “OK” is written in the terminal windows. This confirms that the connected the evaluation kit is successfully connected to your host machine.

```
AT
OK
```



Keep the terminal open, as it is needed for further sections.

2.3 Running the “Quick Connect” demo application

The Quick Connect demo application allows you to establish a connection with AWS IoT, all in the space of a few minutes; no dependencies to install, no source code to download and build, and no AWS account is required. To run the demo, follow the steps below:

1. On the terminal application, type the “AT+FACTORY_RESET” command, press return, and wait for the “OK” message to be written in the terminal window. This command fully resets the module. See also the AWS IoT ExpressLink programmer's guide [\[4\]](#).


```
AT+FACTORY_RESET
OK
```

2. After that, set the Wi-Fi credentials (SSID and Password) using the AT Commands “AT+CONF SSID==<your router SSID>” and “AT+CONF Passphrase=<your router passphrase>”

```
AT+CONF SSID==<your router SSID>
OK
AT+CONF Passphrase=<your router passphrase>
OK
```

3. Type the “AT+CONNECT” command and press return. Wait for an “OK 1 CONNECTED” or “OK 1 CONNECTED STAGING” message to be written in the terminal windows. This means that the evaluation kit connected successfully to the cloud.

```
AT+CONNECT
OK 1 CONNECTED
```


 If the module has never been connected to an AWS account, the `AT+CONNECT` command must be executed twice. In the first instance, it is necessary that the AWS IOT core recognizes and registers the module. In the second instance, the connection between the module and the AWS account is established [8].

4. Download the Quick Connect executable:
 - 4.1. [Download for Mac](#)
 - 4.2. [Download for Windows](#)
 - 4.3. [Download for Linux](#)
5. Unzip the package. Open the `config.txt` file and enter the serial port corresponding to the evaluation kit. For example, COM14, `/dev/cu.usbserial-12345`, and so on in the serial port field.
6. Enter your Wi-Fi credentials in the SSID and Passphrase fields and save the file to store the updates.
7. Run the "Start_Quick_Connect" executable.

The demo connects to AWS IoT Core and prints an URL that you can use to visualize data flowing from the device to the cloud using "AT+SEND1" commands. The demo runs for up to two minutes, and afterward, you can type "AT+SEND1" commands in the demo console window and see the sent data in the Quick Connect Dashboard shown in [Figure 1](#). A valid "AT+SEND1" command example is shown as follows:

```
AT+SEND1 [{"label": "Random Values", "display_type": "line_graph", "values": [{"unit": "C", "value": 3, "label": ""}]]
```

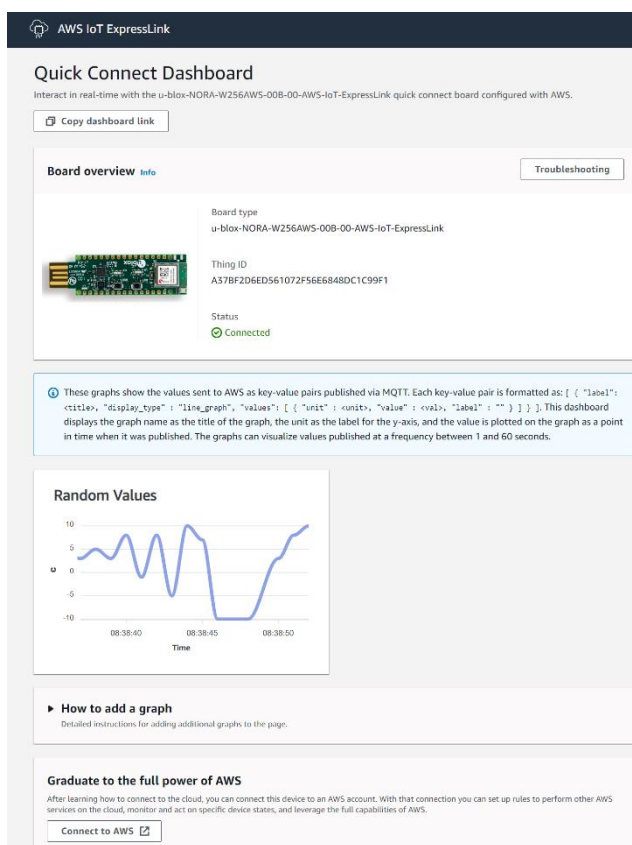


Figure 2: Quick Connect Dashboard

3 Registering an AWS IoT ExpressLink in your development account

Go to “Set up your AWS account” to sign up for an AWS account and create an administrative user to use the AWS IoT console [2]. See also [Registering an AWS IoT ExpressLink in your development account](#).

3.1 Creating and Configuring the Thing

To create an IoT “Thing” and add it to your account you must retrieve the AWS IoT ExpressLink “Thing Name” and the corresponding certificate of the module. The “Thing name” is a sequence as characters that identifies the NORA-W256AWS module and its virtual cloud representation (See also the AWS IoT ExpressLink programmer's guide [4]).

Follow the procedure below⁴ to register your development account using the AWS Management Console [5].

1. Open the AWS IoT Console.
2. Select **Manage** then select **Things**.
3. Choose **Create things**, select **Create single thing**, and then click **Next**.
4. In the terminal application, type the command: “AT+CONF? ThingName” and copy the returned string (a sequence of alphanumeric characters) from terminal.

```
AT+CONF? ThingName
OK <sequence of alphanumeric characters>
```

5. On the **Specify thing properties** page, paste the copied string from terminal into the **Thing name** under **Thing properties** on the console. Leave all other fields as default, then click **Next**.
6. In the terminal application, type the command: “AT+CONF? Certificate pem”

```
AT+CONF? Certificate pem
-----BEGIN CERTIFICATE-----
<sequence of alphanumeric characters>
-----END CERTIFICATE-----
```

7. Copy the returned string (a longer sequence of alphanumeric symbols) and save the string as a text file called “ThingName.cert.pem” on your host machine.
8. On the **Configure device certificate** page, select **Use my certificate** and choose **CA is not registered with AWS IoT**.
9. Under **Certificate**, select **Choose file** and then upload the “ThingName.cert.pem” file created in step 5.
10. Under **Certificate Status**, select **Active**.
11. Click **Next** to **Attach policies to certificate**.
12. On the **Attach policies to certificate page**, select **Create policy** (opens a new window).
13. Include the policy name (e.g., IoTDevPolicy). On the **Policy statements**, select the **JSON** option.
14. Copy the following code snippet into the **Policy document** console.


```
{"Version": "2012-10-17", "Statement": [ {"Effect": "Allow", "Action": "*",
"Resource": "*"}]}
```

15. Click **Create** to complete policy creation.
16. Close **Create a policy** window and return to **Create single thing** window.
17. Select the newly created **IoTDevPolicy** as policy.
18. Click **Create thing** to complete the Thing creation.

⁴ The steps described in the section uses the “New console experience” of the AWS IoT Console.

19. In the AWS IoT Console, choose **Settings**, copy the “Endpoint” string of your account under **Device data endpoint**.
20. In the terminal application, type the following command for example:

```
AT+CONF Endpoint=a3ixxxxxxxxx7i2-ats.iot.eu-north-1.amazonaws.com
```


 The examples in this document are intended only for development environments. All devices in your production fleet must have credentials with privileges that authorize only intended actions on specific resources. The specific permission policies can vary for your use case. Identify the permission policies that best meet your business and security requirements. For more information, see also “Security best practices in AWS IoT Core” in the AWS documentation [\[6\]](#).

3.2 Setting up and connecting to Wi-Fi

The USB-NORA-W256AWS evaluation board requires access to a local Wi-Fi router to connect to the internet.

Enter the required security credentials and enter the following commands in the terminal application:

```
AT+CONF SSID=<your router SSID>
AT+CONF Passphrase=<your router passphrase>
```


 The SSID and passphrase of your local router are stored securely inside the AWS IoT ExpressLink module. While the SSID can be retrieved later (for debugging purposes and so on) any attempt to retrieve the Passphrase will return an error.

3.3 Validating the onboarding process

Having completed all previous steps in this chapter, enter the command “AT+CONNECT” in the Terminal application to validate the onboarding process.

The “OK CONNECTED” message confirms the successful connection with the cloud.

```
AT+CONNECT
OK 1 CONNECTED
```

 You have now completed the registration of the evaluation kit as a “Thing” in your IoT account. As the AWS IoT ExpressLink remembers its configuration, the module automatically accesses your registered AWS account the next time you connect.

4 Connecting and interacting with AWS cloud

Use the MQTT client in the AWS IoT console to monitor the communication between your evaluation kit and the AWS Cloud.

1. Navigate to the AWS IoT console [\[3\]](#).
2. In the navigation pane, select **Test** and then **MQTT Test Client** to open the MQTT client.
3. In **Subscribe to a topic**, type #. The multi-level wildcard subscribes and listens to all payloads published to your account.
4. Click **Subscribe**.

4.1 Connecting

In the terminal application connected to the device COM port, enter the command “AT+CONNECT” to establish a secure connection. The “OK CONNECTED” message confirms the successful connection to the cloud.

4.2 Send data to AWS cloud

To check communication with the MQTT test client:

1. In the terminal application connected to the device COM port, type the command: “AT+CONF Topic1=MyTopic”, and press return.

```
AT+CONF Topic1=MyTopic
OK
```

2. After that, type the command: “AT+SEND1 Hello from my IoT device”. This command sends the “Hello from my IoT device” string to the previously defined Topic1. After a short delay, terminal returns the prompt OK to confirm that the command has been sent.

```
AT+SEND1 Hello from my IoT device
OK
```

3. Check that the message “Hello from my IoT device” is now displayed on the AWS IoT console under the topic “MyTopic”, as shown in [Figure 3](#).

AWS IoT > MQTT test client

MQTT test client info

You can use the MQTT test client to monitor the MQTT messages being passed in your AWS account. Devices publish MQTT messages that are identified by topics to communicate their state to AWS IoT. AWS IoT also publishes MQTT messages to inform devices and apps of changes and events. You can subscribe to MQTT message topics and publish MQTT messages to topics by using the MQTT test client.

▶ **Connection details** ✔ Connected
 You can update the connection details by choosing Disconnect and making updates on the Establish connection to continue page.

[Subscribe to a topic](#) | [Publish to a topic](#)

Topic filter Info
 The topic filter describes the topic(s) to which you want to subscribe. The topic filter can include MQTT wildcard characters.

MyTopic

▶ **Additional configuration**

[Subscribe](#)

Subscriptions MyTopic Pause Clear Export Edit

MyTopic ♥ X

▼ MyTopic April 06, 2023, 09:18:40 (UTC+0200)

⊘ Message cannot be displayed in specified format.

Hello from my IoT device

▶ Properties

Figure 3: Subscriptions on MQTT test client in AWS IoT console

4.3 Receiving data and commands from AWS cloud

Use your terminal application and AWS IoT console to check the receipt of data and commands from the AWS cloud.

In the terminal application connected to the device COM port, enter the commands below to subscribe to a topic:

1. Enter the command `"AT+CONF Topic1=MyTopic"`. After a short delay, terminal returns the prompt `OK` to confirm that the command has been sent.
2. Enter the command `"AT+SUBSCRIBE1"`

```
AT+CONF Topic1=MyTopic
OK
AT+SUBSCRIBE1
OK
```

To publish a message on the same topic, enter the following commands in the AWS IoT console:

1. Select **Publish to a topic**
2. in **Topic name** field, type `"MyTopic"`.
3. On **message payload** field, type the text `"Hello from the AWS IoT console"` message then click **"Publish"**. See [Figure 4](#).

AWS IoT > MQTT test client

MQTT test client [info](#)

You can use the MQTT test client to monitor the MQTT messages being passed in your AWS account. Devices publish MQTT messages that are identified by topics to communicate their state to AWS IoT. AWS IoT also publishes MQTT messages to inform devices and apps of changes and events. You can subscribe to MQTT message topics and publish MQTT messages to topics by using the MQTT test client.

► **Connection details** ✔ Connected
 You can update the connection details by choosing Disconnect and making updates on the Establish connection to continue page.

Subscribe to a topic | **Publish to a topic**

Topic name
 The topic name identifies the message. The message payload will be published to this topic with a Quality of Service (QoS) of 0.

Q MyTopic ✕

Message payload

```
{
  "message": "Hello from AWS IoT console"
}
```

► **Additional configuration**

Publish

Figure 4: Publish to a Topic on MQTT test client in AWS IoT console

In the terminal application, type the command “AT+GET1”, and press return. The message “OK Hello from the AWS IoT console” is printed at the prompt.

```
AT+GET1
OK { "message": "Hello from AWS IoT console" }
```

5 Over-The-Air updates for AWS IoT ExpressLink

5.1 Acquiring software updates

Firmware update images are available on the u-blox product webpage under the “Documentation & resources” tab [7].

5.2 Prerequisites

You should have a firmware image signed by the manufacturer of the ExpressLink module. Along with the firmware image, you should also receive additional signing metadata such as:

- Signature hashing algorithm used (Example: SHA-256)
- Signature encryption algorithm used (Example: RSA)
- Actual signature encoded using the base64 encoding format.
- The path name (a string) which identifies the location where the certificate is provisioned in the ExpressLink (optional)

5.3 Create an OTA update job in AWS IoT

Create an OTA Update role in your account:

1. Open [AWS IoT Console](#). Select **Manage** then select **Jobs**. Choose **Create job**, select **Create FreeRTOS OTA Update Job** and click **Next**.
2. Provide a job name which is unique within your AWS account. Provide an optional description. Click **Next**.
3. From **Devices to update** drop-down menu, choose the “Thing” name with which the USB-NORA-W256AWS has a registered account. Select **MQTT** as the protocol to use for transfer.
4. Choose **Use my custom signed file** and complete the displayed form. When filling the form, use the details supplied in the firmware package.
5. In the **signature** field provide the base64 encoded signature for the image. From the **Original hashing algorithm** drop-down menu, select the hashing algorithm given in the firmware package.
6. From the **Original encryption algorithm** drop-down menu, select the encryption algorithm given in the firmware package. For **Path name of code signing certificate on device**, enter the path name as stated in the same procedure. If path name is not provided, enter NA.
7. Select **Upload a new file**, click on **Choose file** and upload the image received from u-blox.
8. Select **Create S3 bucket** for the new uploaded image and proceed with creating a new bucket. If needed, choose an existing bucket in your account by selecting the **Browse S3** option.
9. Under **Path Name of file on device**, enter **NA** if the image is not targeted as an executable file within a filesystem.
10. In the **File type** input field, define the type of software to be updated - either the ExpressLink software (FOTA) or host software (HOTA). Set the value 101 for ExpressLink and 202 for the host software.
11. From **role** dropdown under **IAM role**, choose the OTA update role created above. Click **Next**.
12. Click **Create Job**. On successful creation, the job name and state are displayed as in progress.

5.4 Monitoring and applying OTA jobs

The NORA-W256AWS IoT ExpressLink module automatically polls for firmware update jobs. Once a new job has been detected it downloads and validates the software image. It then enters a waiting state where the host application must accept the update to be applied. The process is described below.

1. The host application receives an OTA event indicating that a new firmware image is available for ExpressLink. The host application can query the state of the job using the command `"AT+OTA?"`.
2. The module responds with `"OK 1 <version>"` to confirm that a module OTA firmware update is proposed.
3. The host application can accept a new firmware update for by issuing command `"AT+OTA ACCEPT"`.
4. The module starts downloading the firmware update from the AWS IoT cloud. During the download, the host can monitor the state of the job using `"AT+OTA?"`
5. After successful download and signature validation, the host receives an event to apply the new image.
6. The host application can apply the new image by issuing the command `"AT+OTA APPLY"`.
7. When the apply command has been sent, the module reboots using the new software image. The host receives a STARTUP event confirming that the new image is booted. To see the event, enter the command `"AT+EVENT?"`
8. The host application can connect back to the AWS IoT with the command `"AT+CONNECT"`.
9. The module now connects to AWS IoT, completes the self-test, and marks the image as valid to prevent rollback to any older image.
10. The job status in the [AWS IoT console](#) is shown as completed and succeeded.