User Manual

DA16200 DA16600 Getting Started with Azure® IoT

UM-WI-058

Abstract

The focus of this document is to introduce a door lock example for Azure® IoT. This is a reference design which includes the Azure® IoT Application. This reference design provides the user an experience using the Azure® IoT Android/iOS Application.

Contents

Figures

1 Terms and Definitions

AP	Access Point
ADB	Android Debug Bridge
API	Application Programming Interface
Azure	Microsoft Web Services
CFG	Configure
CMD	Command
DPM	Dynamic Power Management
DTIM	Delivery Traffic Indication Map
EVB	Evaluation Board
IoT	Internet of Things
MCU	Micro-Controller Unit
OTA	Over-The-Air
RTC	Real-Time Clock
SDK	Software Development Kit
TIM	Traffic Indication Map
TSL	Transport Layer Security
NVRAM	Non-Volatile Random Access Memory
DPS	Device Provisioning Service

2 References

[1] UM-WI-056, DA16200 DA16600 FreeRTOS Getting Started Guide, User Manual, Renesas Electronics

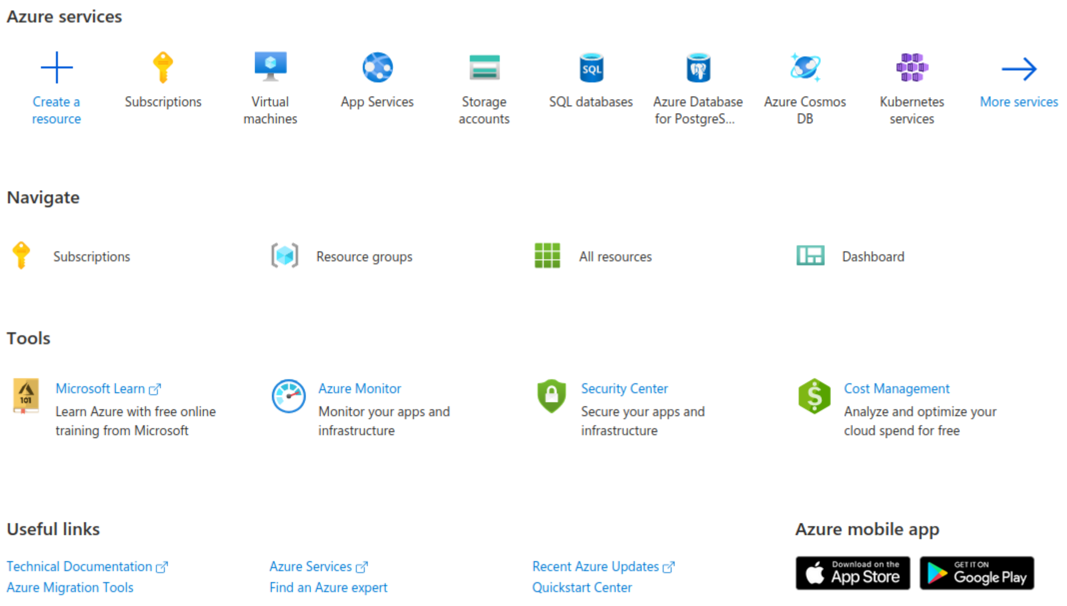
[2] UM-WI-042, DA16200 DA16600 Provisioning Mobile App, User Manual, Renesas Electronics

3 Azure IoT Hub

A Microsoft Azure account is required to run the door lock reference application. This section provides how to configure Azure IoT Hub for communicating with the DA16200/DA16600 IoT device.

3.1 Azure Services

The Azure Portal provides various services including IoT as shown in Figure 1.


Azure services

Create a resource | Subscriptions | Virtual machines | App Services | Storage accounts | SQL databases | Azure Database for PostgreS... | Azure Cosmos DB | Kubernetes services | More services

Navigate

Subscriptions | Resource groups | All resources | Dashboard

Tools

Microsoft Learn — Learn Azure with free online training from Microsoft

Azure Monitor — Monitor your apps and infrastructure

Security Center — Secure your apps and infrastructure

Cost Management — Analyze and optimize your cloud spend for free

Useful links

Technical Documentation
Azure Migration Tools
Azure Services
Find an Azure expert
Recent Azure Updates
Quickstart Center

Azure mobile app

Download on the App Store | GET IT ON Google Play

Figure 1: Azure Services

3.2 Azure IoT Hub Setup

To connect a device to the Azure IoT server, the following components are required. This section describes how to set up requirements before using Azure IoT Hub.

- Azure account
- Resource groups
- IoT Hub
- Devices

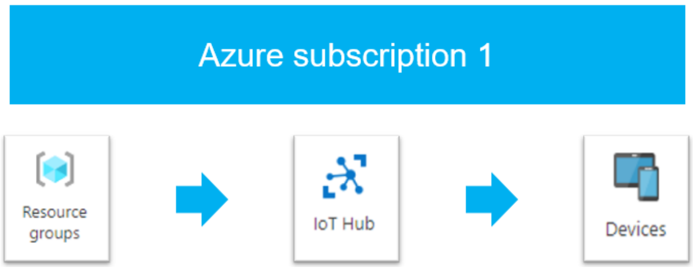


Figure 2: Azure IoT Service Configuration

3.2.1 Create Azure Account

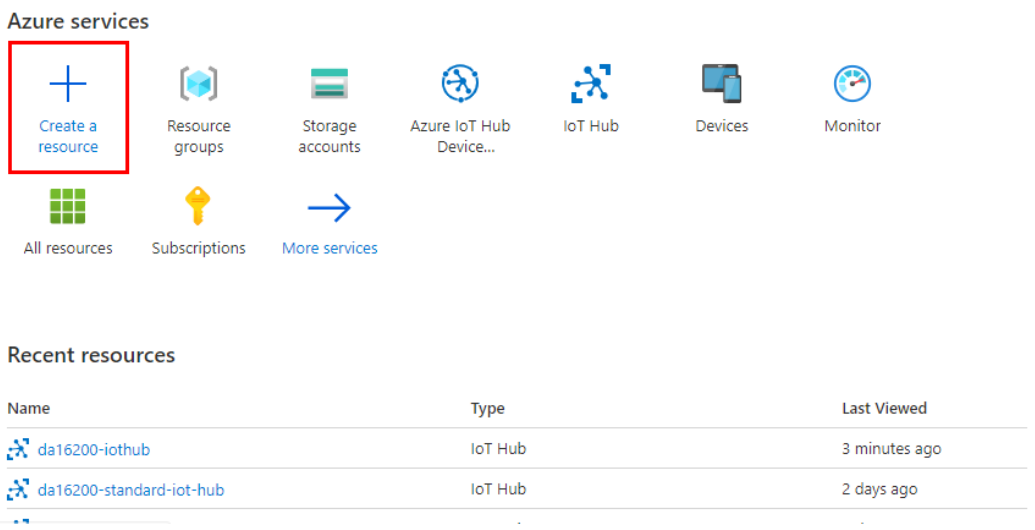
To create an Azure account, follow the steps below:

1. Go to the https://azure.microsoft.com.
2. Create an Azure account and sign in.
3. Go to Microsoft Azure Portal to use Azure services (https://azure.microsoft.com/en-us/features/azure-portal).

3.2.2 Create Resource Group

To register an IoT Hub resource, a resource group should be created.

1. In Azure services, click **Create a resource**.



Name	Type	Last Viewed
da16200-iothub	IoT Hub	3 minutes ago
da16200-standard-iot-hub	IoT Hub	2 days ago

Figure 3: Create a Resource

2. Search and click **Resource group**.

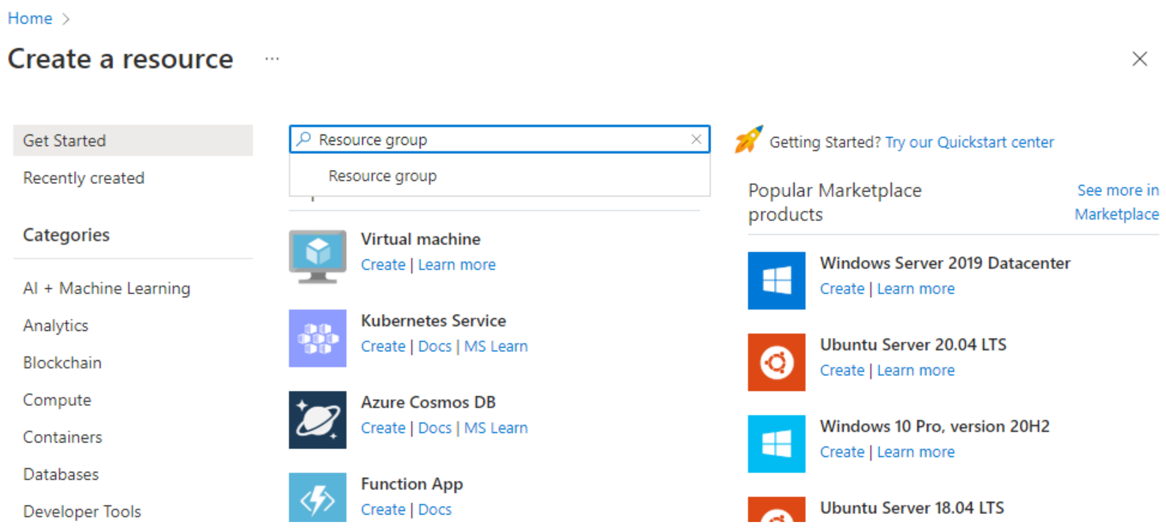
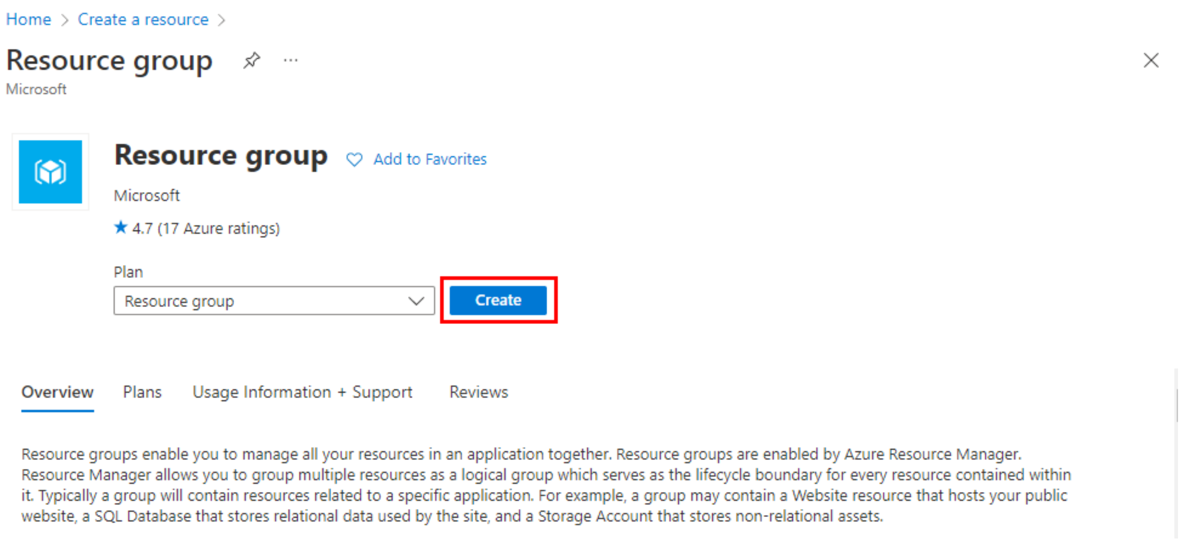


Figure 4: Resource Group

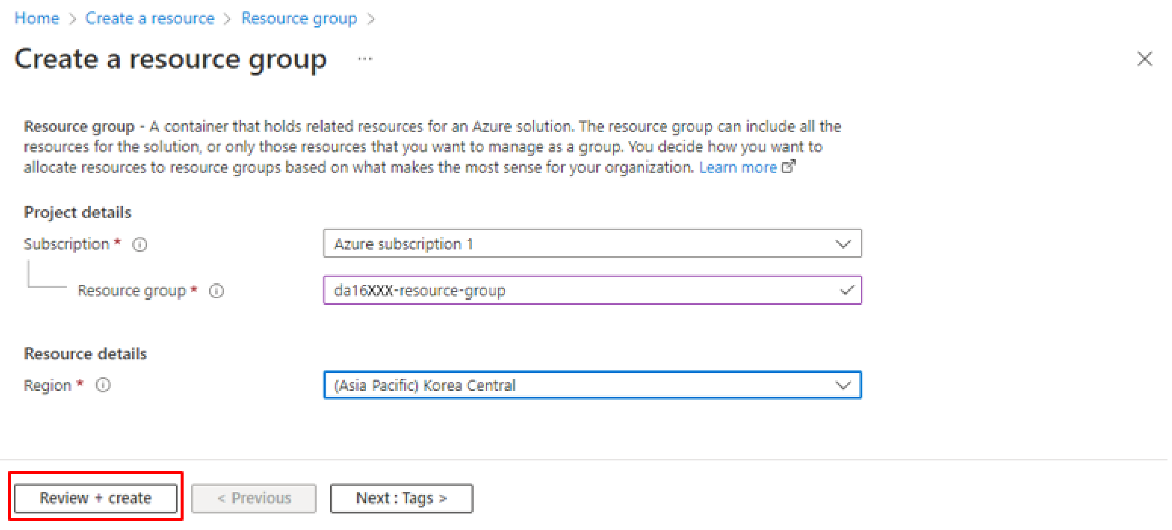
3. Click **Create** to create a new resource group.



Resource groups enable you to manage all your resources in an application together. Resource groups are enabled by Azure Resource Manager. Resource Manager allows you to group multiple resources as a logical group which serves as the lifecycle boundary for every resource contained within it. Typically a group will contain resources related to a specific application. For example, a group may contain a Website resource that hosts your public website, a SQL Database that stores relational data used by the site, and a Storage Account that stores non-relational assets.

Figure 5: New Resource Group

4. Add a resource group name and select a region, and then click **Review + create** to accept the configuration.



Resource group - A container that holds related resources for an Azure solution. The resource group can include all the resources for the solution, or only those resources that you want to manage as a group. You decide how you want to allocate resources to resource groups based on what makes the most sense for your organization. Learn more

Project details

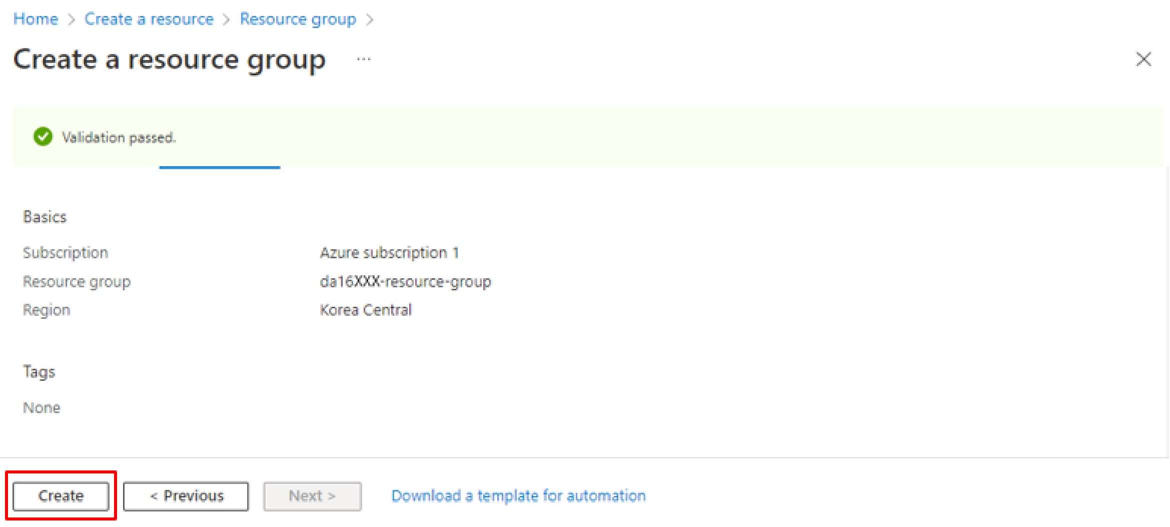
Subscription *	Azure subscription 1
Resource group *	da16XXX-resource-group

Resource details

Region *	(Asia Pacific) Korea Central

Figure 6: Adding Data for Resource Group

5. Click **Create**.



Home > Create a resource > Resource group >

Create a resource group

Validation passed.

Basics

Subscription	Azure subscription 1
Resource group	da16XXX-resource-group
Region	Korea Central

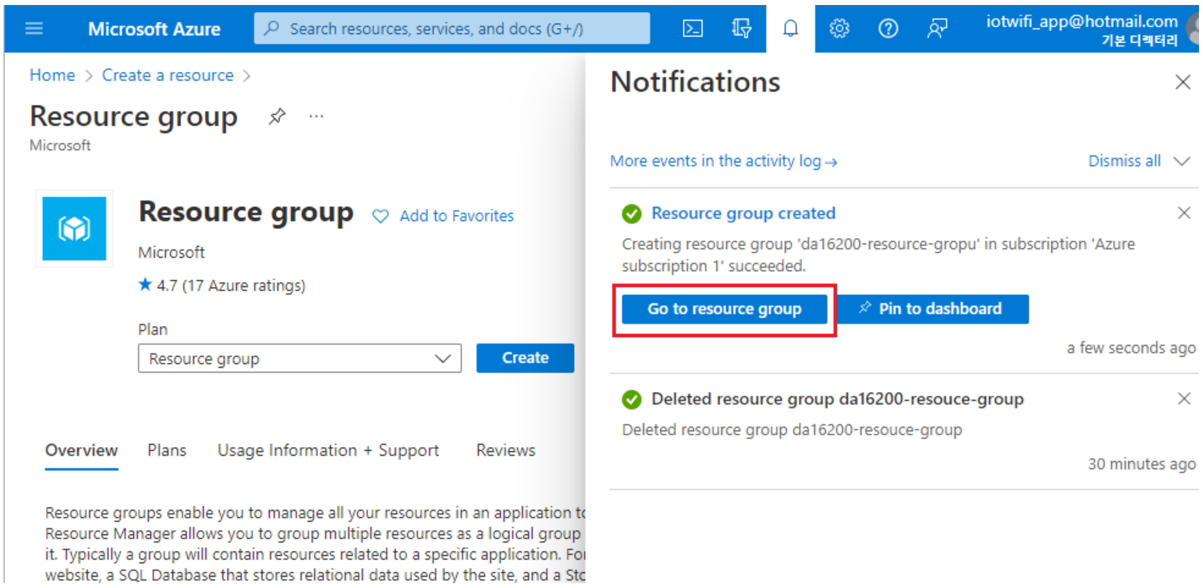
Tags

None

Create | < Previous | Next > | Download a template for automation

Figure 7: Create Resource Group

6. Click **Go to resource group**.



Notifications

More events in the activity log → Dismiss all

Resource group created

Creating resource group 'da16200-resource-gropu' in subscription 'Azure subscription 1' succeeded.

Go to resource group | Pin to dashboard

a few seconds ago

Deleted resource group da16200-resouce-group

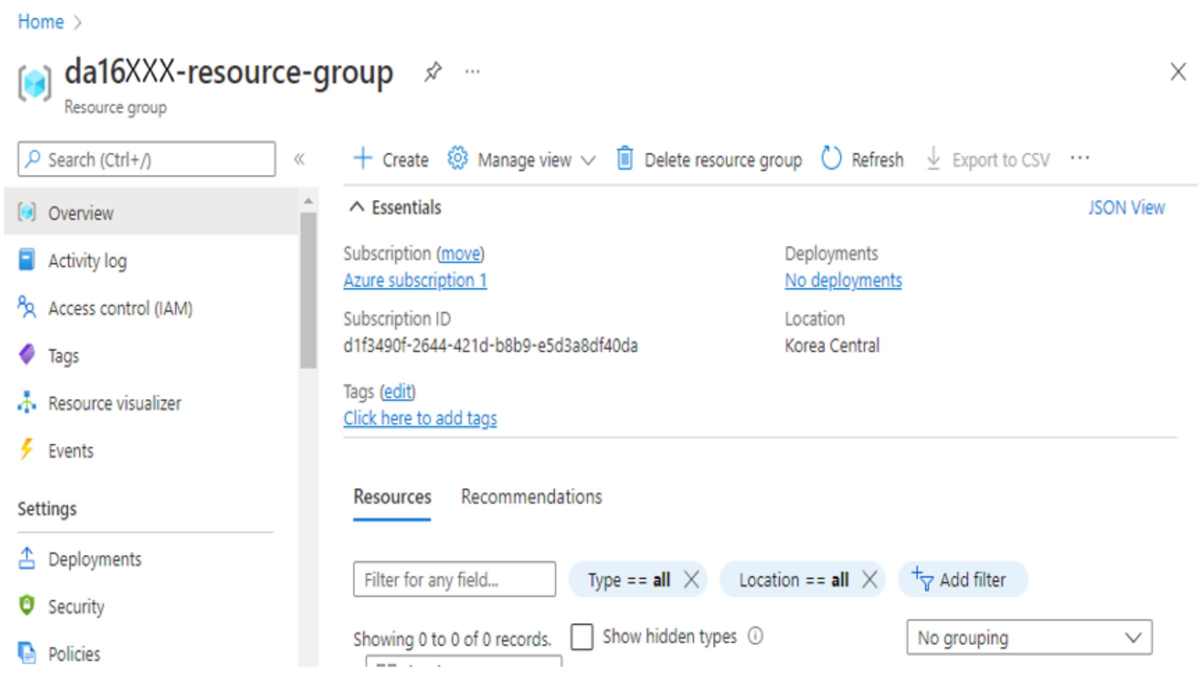
Deleted resource group da16200-resouce-group

30 minutes ago

Resource groups enable you to manage all your resources in an application to
Resource Manager allows you to group multiple resources as a logical group
it. Typically a group will contain resources related to a specific application. For
website, a SQL Database that stores relational data used by the site, and a Sto

Figure 8: Go to Resource Group

The newly created resource group is displayed.



Subscription (move)
Azure subscription 1

Subscription ID
d1f3490f-2644-421d-b8b9-e5d3a8df40da

Tags (edit)
Click here to add tags

Deployments
No deployments

Location
Korea Central

Showing 0 to 0 of 0 records.

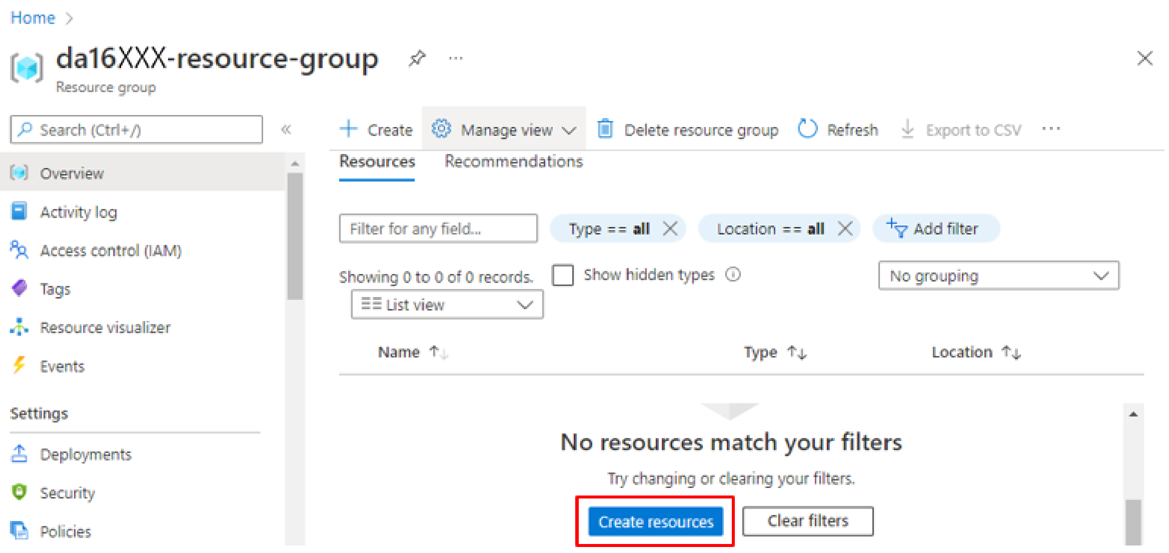
Figure 9: Created Resource Group

3.2.3 Create IoT Hub

Azure IoT Hub is a managed service that acts as a central message hub for communication between IoT applications and IoT devices that are connected to the DA16200/DA16600. IoT Hub should be created before creating IoT devices.

To create IoT Hub:

1. In the created resource group, click **Create resources**.



Showing 0 to 0 of 0 records.

No resources match your filters

Try changing or clearing your filters.

Figure 10: Create Resources

2. In **Create a resource** menu, search and click **IoT Hub**.

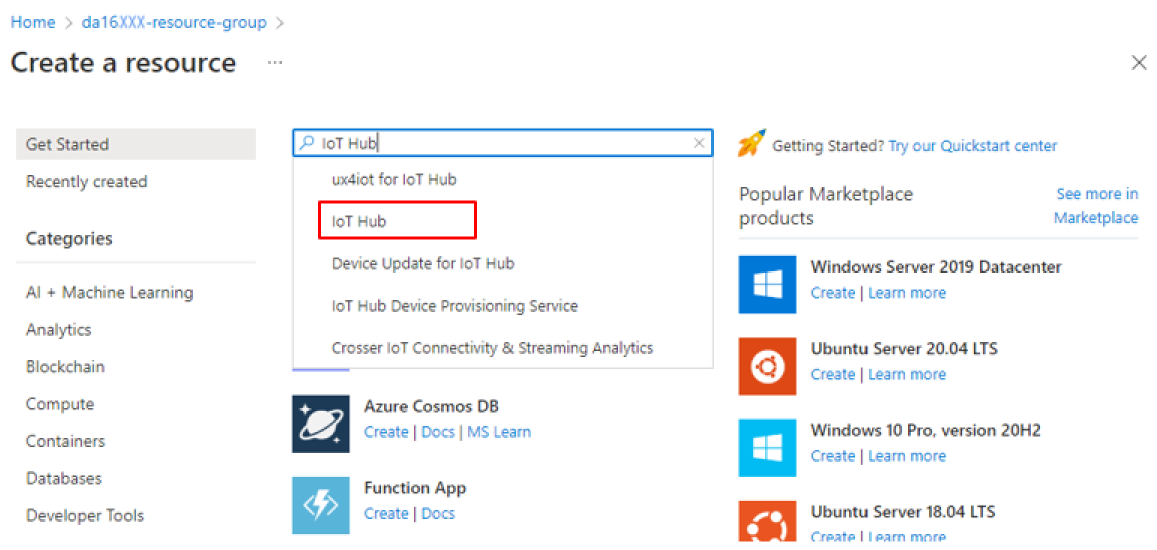
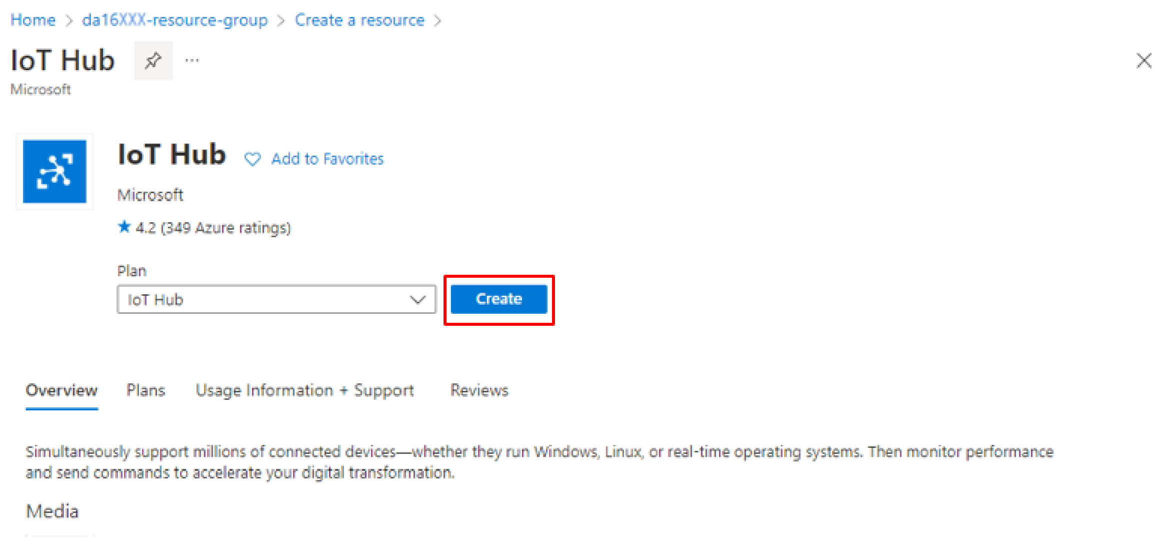


Figure 11: Select IoT Hub

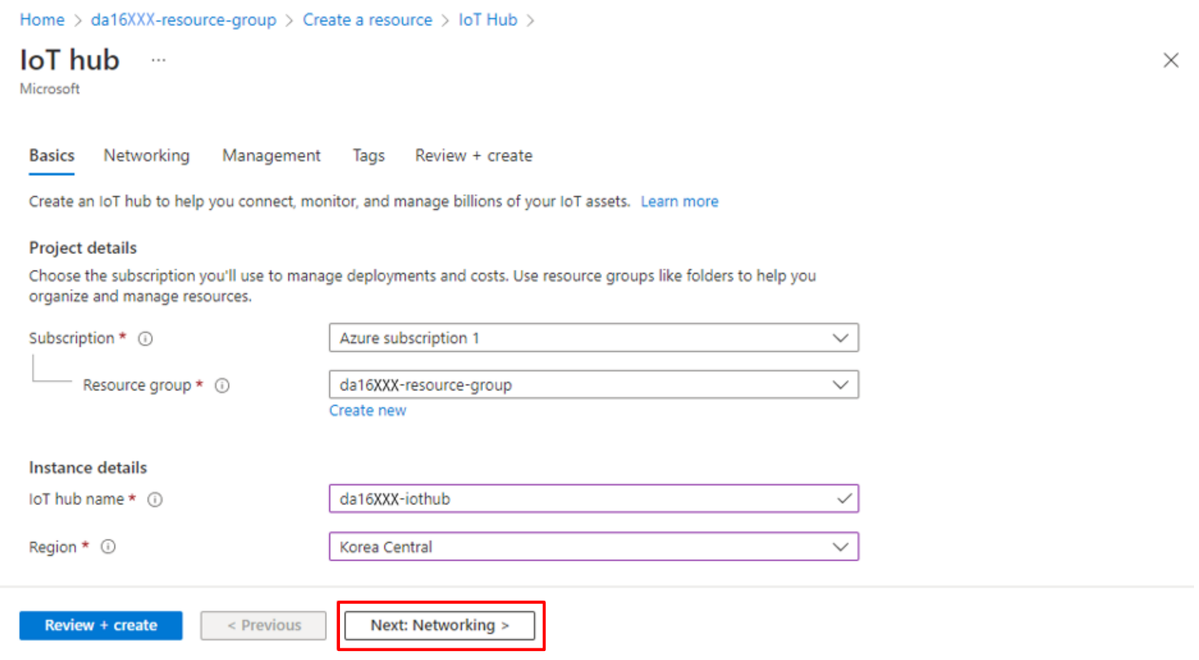
3. In **IoT Hub** menu, click **Create**.


Home > da16XXX-resource-group > Create a resource >
IoT Hub
Microsoft
IoT Hub
Add to Favorites
Microsoft
★ 4.2 (349 Azure ratings)
Plan
IoT Hub
Create
Overview
Plans
Usage Information + Support
Reviews
Media

Simultaneously support millions of connected devices—whether they run Windows, Linux, or real-time operating systems. Then monitor performance and send commands to accelerate your digital transformation.

Figure 12: Create IoT Hub

4. In the **Basics** menu, add the IoT hub name and select a region, and then click **Next: Networking**.



Home > da16XXX-resource-group > Create a resource > IoT Hub >

IoT hub ...

Microsoft

Basics | Networking | Management | Tags | Review + create

Create an IoT hub to help you connect, monitor, and manage billions of your IoT assets. Learn more

Project details

Choose the subscription you'll use to manage deployments and costs. Use resource groups like folders to help you organize and manage resources.

Subscription * : Azure subscription 1

Resource group * : da16XXX-resource-group (Create new)

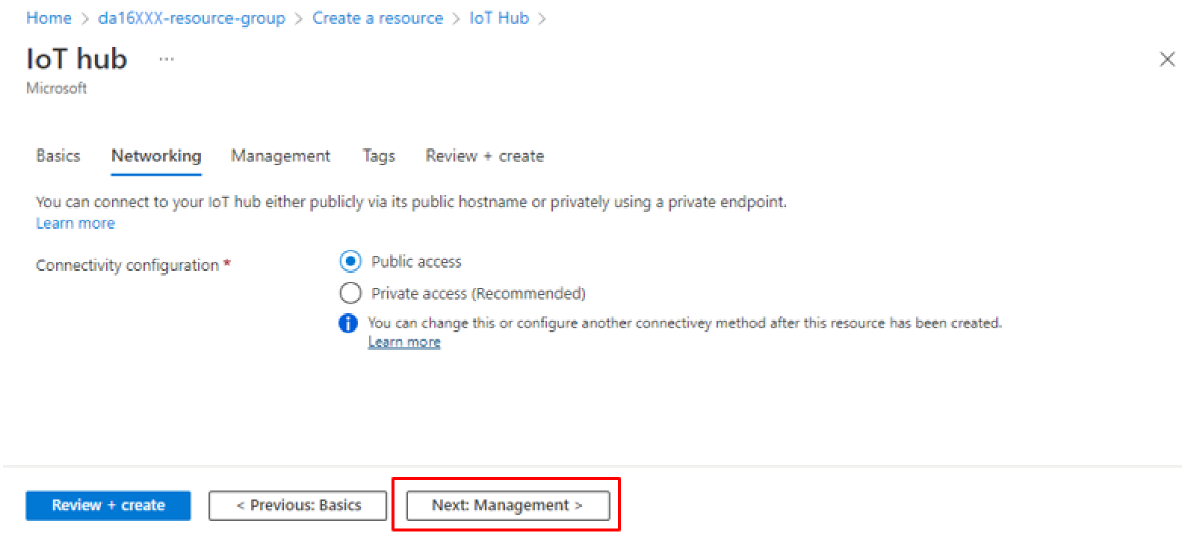
Instance details

IoT hub name * : da16XXX-iothub

Region * : Korea Central

Figure 13: Next Networking

5. In **IoT Hub Networking** menu, click **Next: Management**.



Home > da16XXX-resource-group > Create a resource > IoT Hub >

IoT hub ...

Microsoft

Basics | Networking | Management | Tags | Review + create

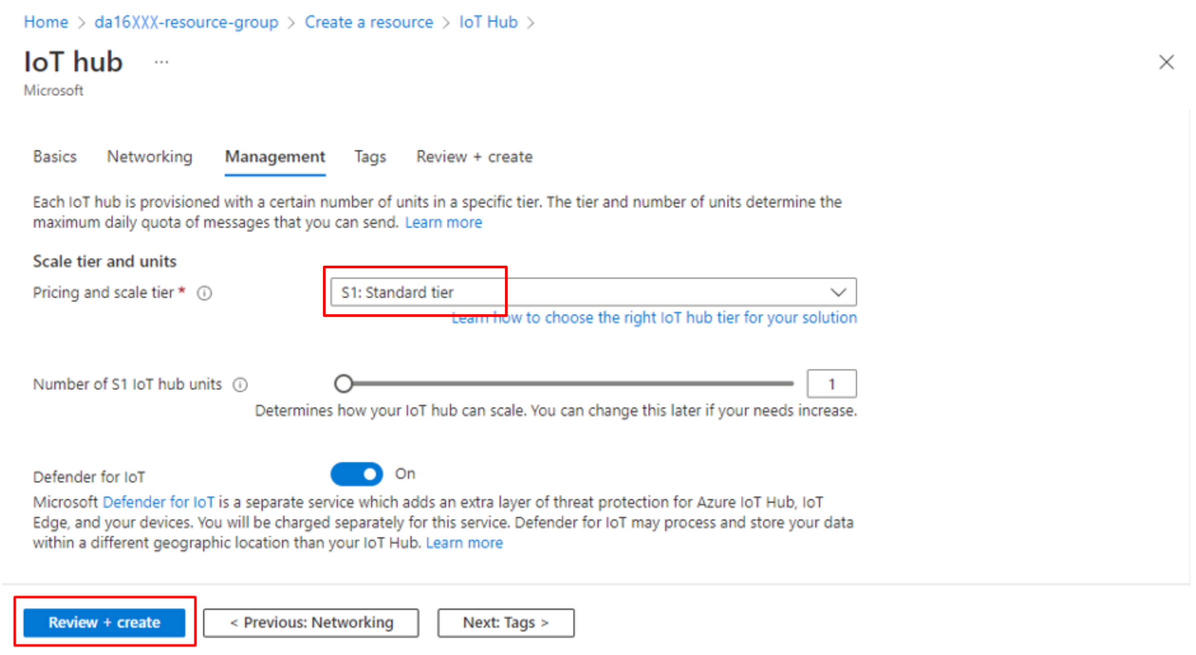
You can connect to your IoT hub either publicly via its public hostname or privately using a private endpoint. Learn more

Connectivity configuration * : [✓ Public access] Private access (Recommended)

You can change this or configure another connectivey method after this resource has been created. Learn more

Figure 14: Next Management

6. In the **IoT Hub Management** menu, select **Pricing and scale tier**, and then click **Review+create**.



Home > da16XXX-resource-group > Create a resource > IoT Hub >

IoT hub ···
Microsoft

Each IoT hub is provisioned with a certain number of units in a specific tier. The tier and number of units determine the maximum daily quota of messages that you can send. Learn more

Scale tier and units

Pricing and scale tier * : S1: Standard tier

Learn how to choose the right IoT hub tier for your solution

Number of S1 IoT hub units: 1

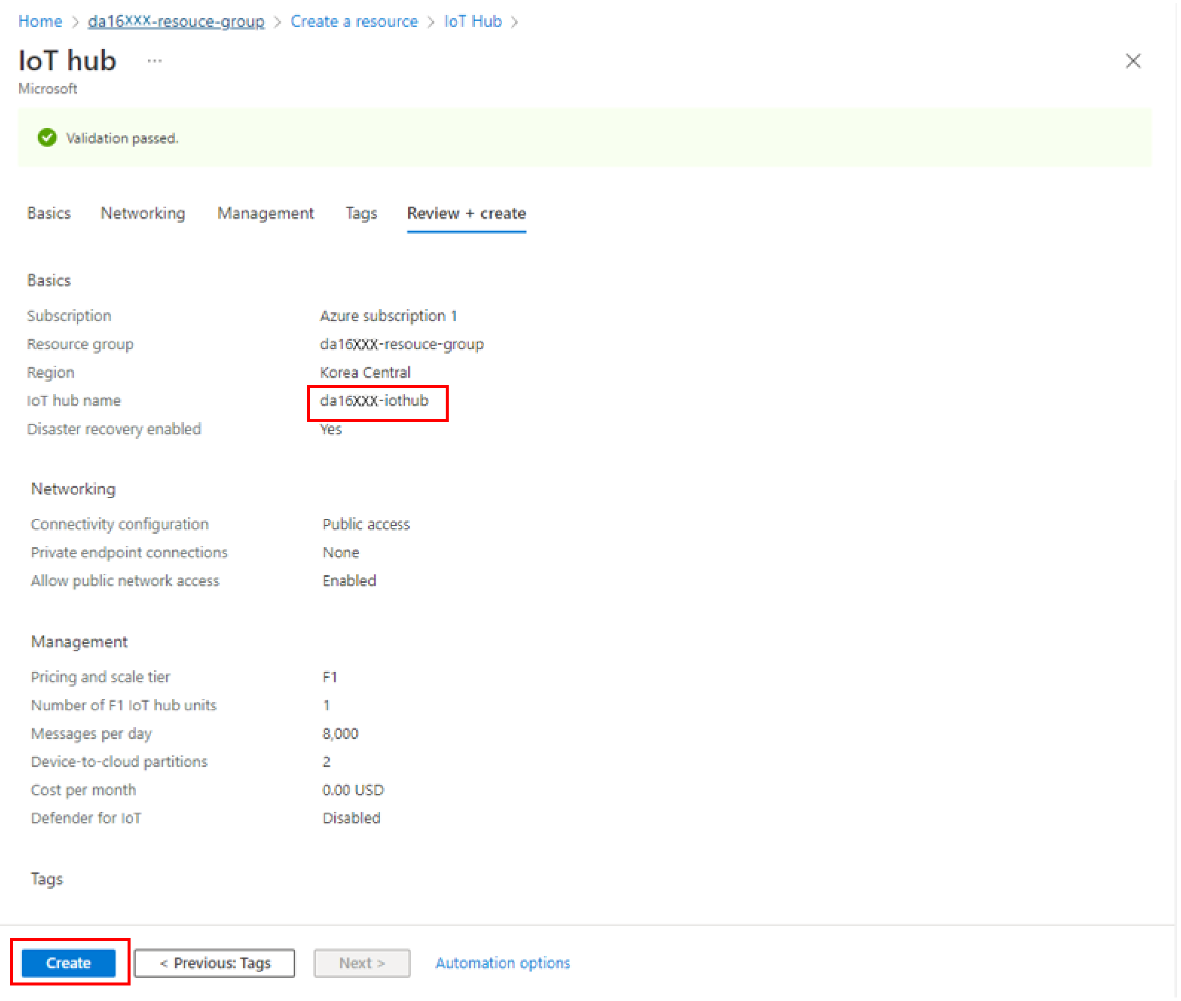
Determines how your IoT hub can scale. You can change this later if your needs increase.

Defender for IoT: On

Microsoft Defender for IoT is a separate service which adds an extra layer of threat protection for Azure IoT Hub, IoT Edge, and your devices. You will be charged separately for this service. Defender for IoT may process and store your data within a different geographic location than your IoT Hub. Learn more

Figure 15: IoT Hub Management

7. Review the items displayed in the **Review + Create** menu, and click **Create**.



Home > da16XXX-resouce-group > Create a resource > IoT Hub >

IoT hub ···
Microsoft

Validation passed.

Basics

Subscription	Azure subscription 1
Resource group	da16XXX-resouce-group
Region	Korea Central
IoT hub name	da16XXX-iothub
Disaster recovery enabled	Yes

Networking

Connectivity configuration	Public access
Private endpoint connections	None
Allow public network access	Enabled

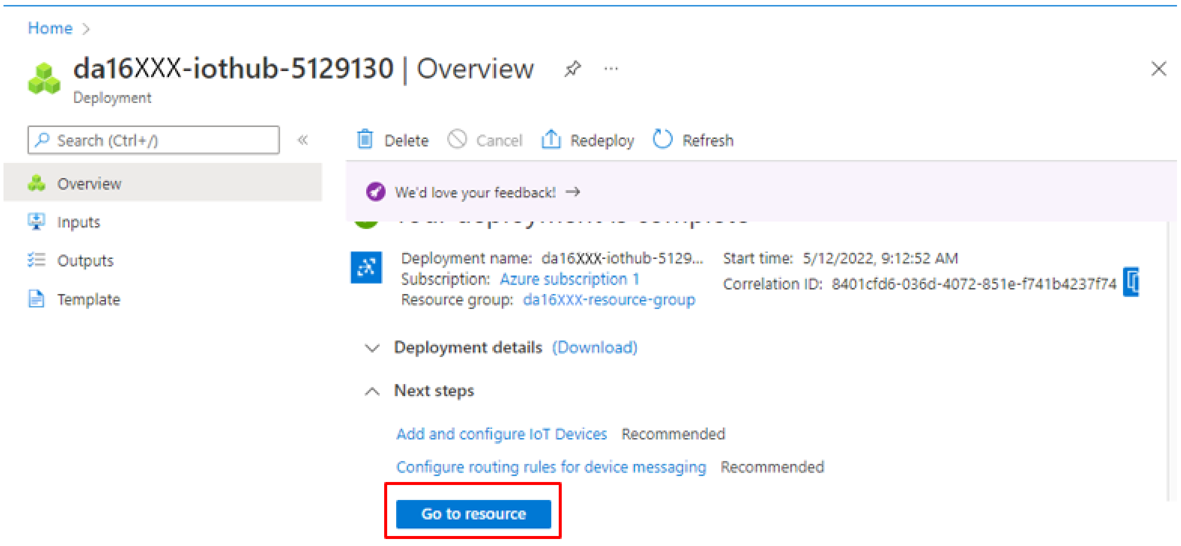
Management

Pricing and scale tier	F1
Number of F1 IoT hub units	1
Messages per day	8,000
Device-to-cloud partitions	2
Cost per month	0.00 USD
Defender for IoT	Disabled

Tags

Figure 16: Review IoT Hub Configuration

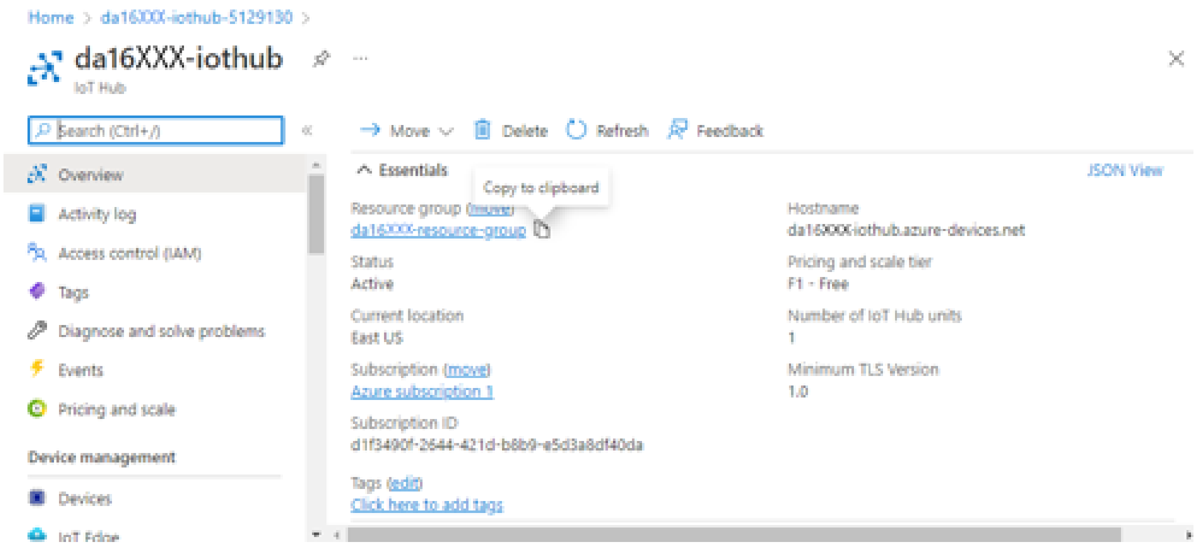
8. After IoT hub deployment is completed, click **Go to resource**.


Home >
da16XXX-iothub-5129130 | Overview
Deployment
Search (Ctrl+/)
Delete
Cancel
Redeploy
Refresh
Overview
Inputs
Outputs
Template
We'd love your feedback! →
Deployment details (Download)
Next steps
Add and configure IoT Devices
Recommended
Configure routing rules for device messaging
Recommended
Go to resource

Deployment name: da16XXX-iothub-5129...
Subscription: Azure subscription 1
Resource group: da16XXX-resource-group
Start time: 5/12/2022, 9:12:52 AM
Correlation ID: 8401cfd6-036d-4072-851e-f741b4237f74

Figure 17: Completed IoT Hub Deployment

The generated IoT hub web page appears.



Resource group (move)
da16XXX-resource-group
Status
Active
Current location
East US
Subscription (move)
Azure subscription 1
Subscription ID
d1f3490f-2644-421d-b8b9-e5d3a8df40da
Tags (edit)
Click here to add tags
Hostname
da16XXX-iothub.azure-devices.net
Pricing and scale tier
F1 - Free
Number of IoT Hub units
1
Minimum TLS Version
1.0

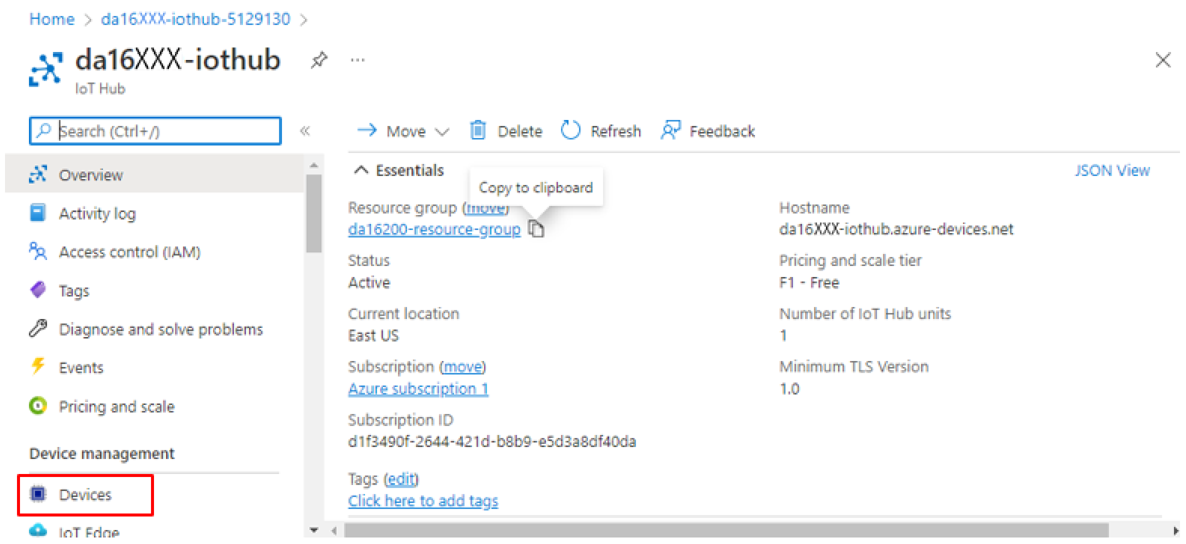
Figure 18: Generated IoT Hub Web Page

3.2.4 Register Devices

IoT devices should be added to Azure IoT Hub.

To add devices:

1. In the created IoT Hub, click **Devices**.


Home > da16XXX-iothub-5129130 >

da16XXX-iothub

IoT Hub

Essentials

Resource group (move): da16200-resource-group

Status: Active

Current location: East US

Subscription (move): Azure subscription 1

Subscription ID: d1f3490f-2644-421d-b8b9-e5d3a8df40da

Tags (edit): Click here to add tags

Hostname: da16XXX-iothub.azure-devices.net

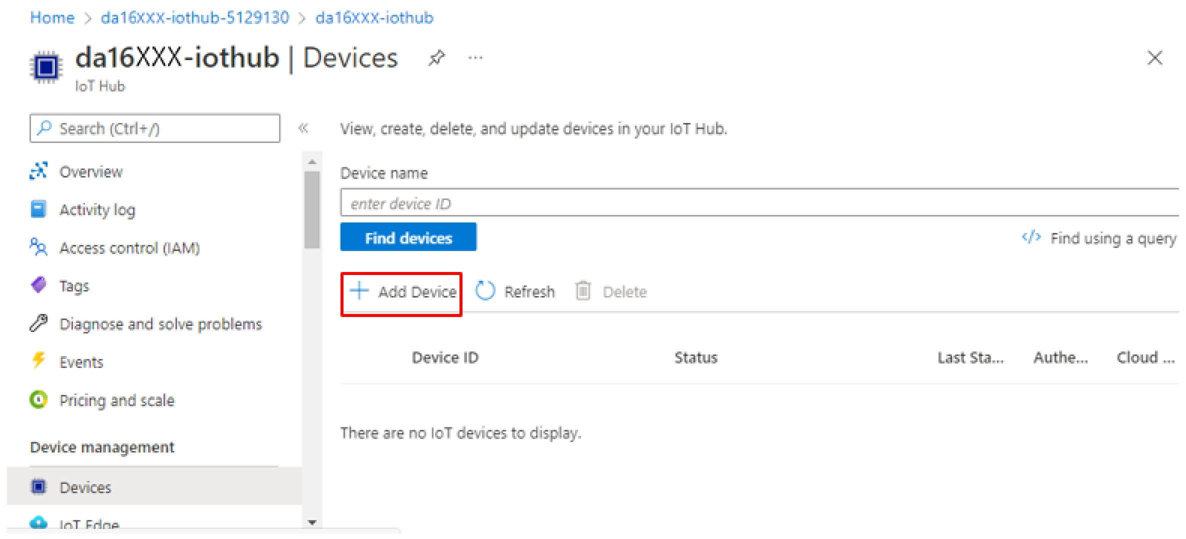
Pricing and scale tier: F1 - Free

Number of IoT Hub units: 1

Minimum TLS Version: 1.0

Figure 19: Add Devices

2. In **Devices** menu, click **Add Device**.


Home > da16XXX-iothub-5129130 > da16XXX-iothub

da16XXX-iothub | Devices

IoT Hub

View, create, delete, and update devices in your IoT Hub.

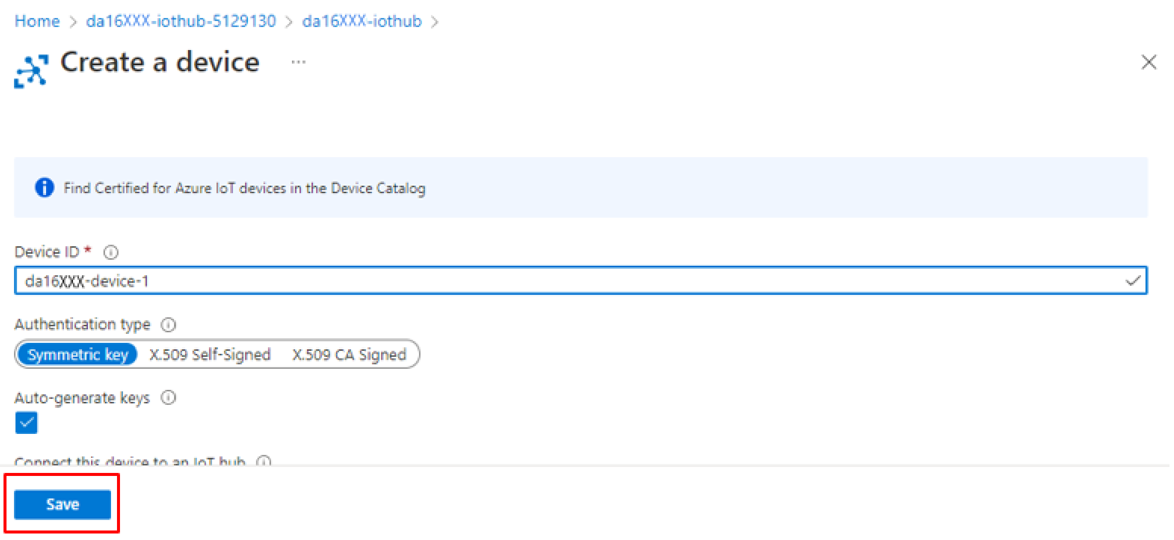
Device name: enter device ID

Device ID	Status	Last Sta...	Authe...	Cloud ...

There are no IoT devices to display.

Figure 20: Add Devices (Continued)

3. Add the desired Device ID, and then click **Save**.


Home > da16XXX-iothub-5129130 > da16XXX-iothub >

Create a device

Find Certified for Azure IoT devices in the Device Catalog

Device ID * : da16XXX-device-1

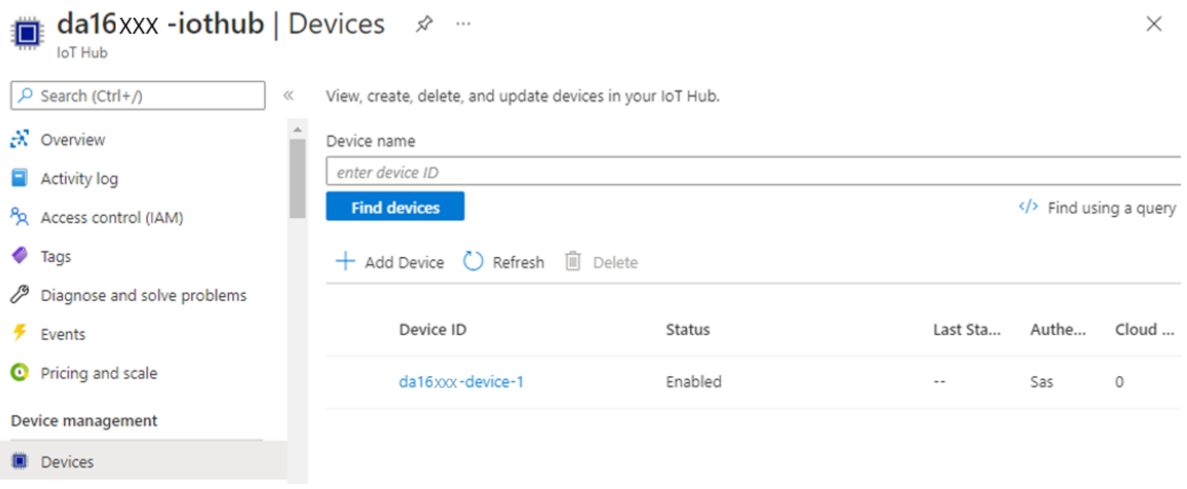
Authentication type: [✓ Symmetric key] X.509 Self-Signed X.509 CA Signed

Auto-generate keys: ✓

Connect this device to an IoT hub

Figure 21: Save Devices

4. The device is registered in the Azure IoT server.


da16xxx -iothub | Devices

IoT Hub

View, create, delete, and update devices in your IoT Hub.

Device name: enter device ID

Device ID	Status	Last Sta...	Authe...	Cloud ...
da16xxx-device-1	Enabled	--	Sas	0

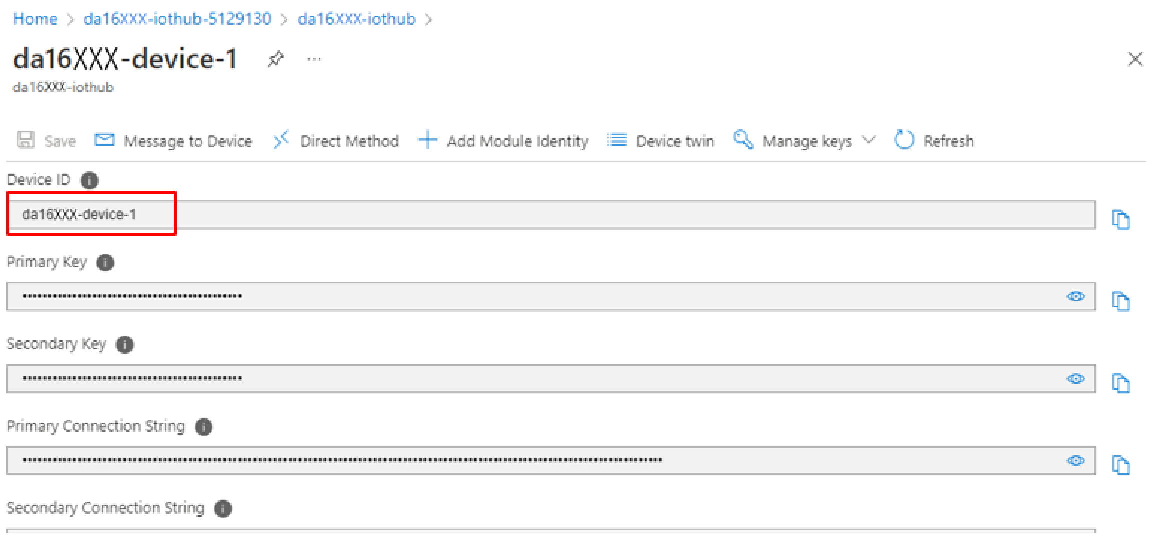
Figure 22: Device Registered in Azure IoT Server

3.3 Configure Azure IoT Hub

The Azure IoT Hub requires several parameters to be configured. These parameters are used for connecting DA16200/DA16600 device and must be set in the parameter of DA16200/DA16600. The Azure IoT Hub and DA16200/DA16600 need to be connected for data communication.

Configure parameters of Azure IoT Hub:

- Device ID: ID registered in Azure IoT Hub



Home > da16XXX-iothub-5129130 > da16XXX-iothub >

da16XXX-device-1

da16XXX-iothub

Device ID: da16XXX-device-1

Primary Key: ••

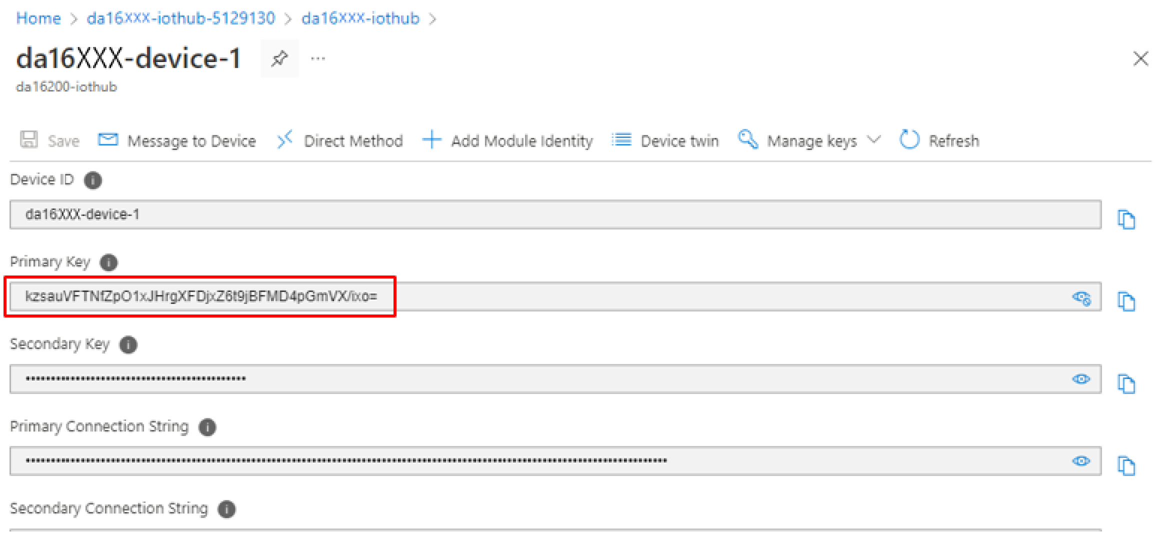
Secondary Key: ••

Primary Connection String: ••

Secondary Connection String

Figure 23: Device ID Registered in Azure IoT Hub

- Device-Primary-Key: Primary key with device ID registered in Azure IoT Hub



Home > da16XXX-iothub-5129130 > da16XXX-iothub >

da16XXX-device-1

da16200-iothub

Device ID: da16XXX-device-1

Primary Key: kzsauVFTNfZpO1xJHrgXFDjxZ6t9jBFMD4pGmVX/ixo=

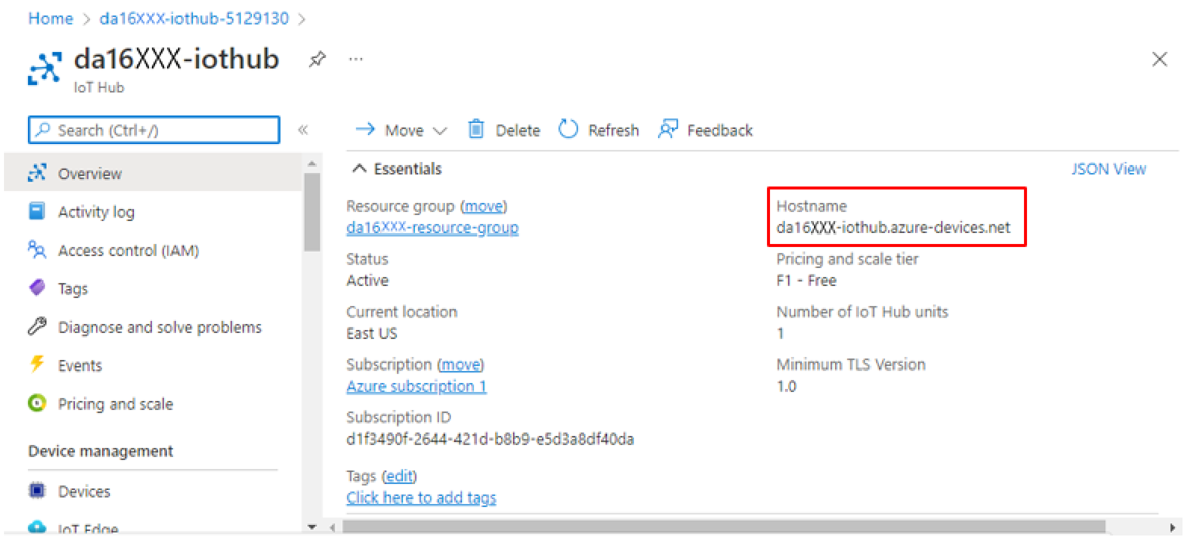
Secondary Key: ••

Primary Connection String: ••

Secondary Connection String

Figure 24: Primary Key with Device ID

- Hostname: Azure IoT Hub address



Home > da16XXX-iothub-5129130 >

da16XXX-iothub
IoT Hub

Essentials

Resource group (move)
da16XXX-resource-group

Hostname
da16XXX-iothub.azure-devices.net

Status
Active

Pricing and scale tier
F1 - Free

Current location
East US

Number of IoT Hub units
1

Subscription (move)
Azure subscription 1

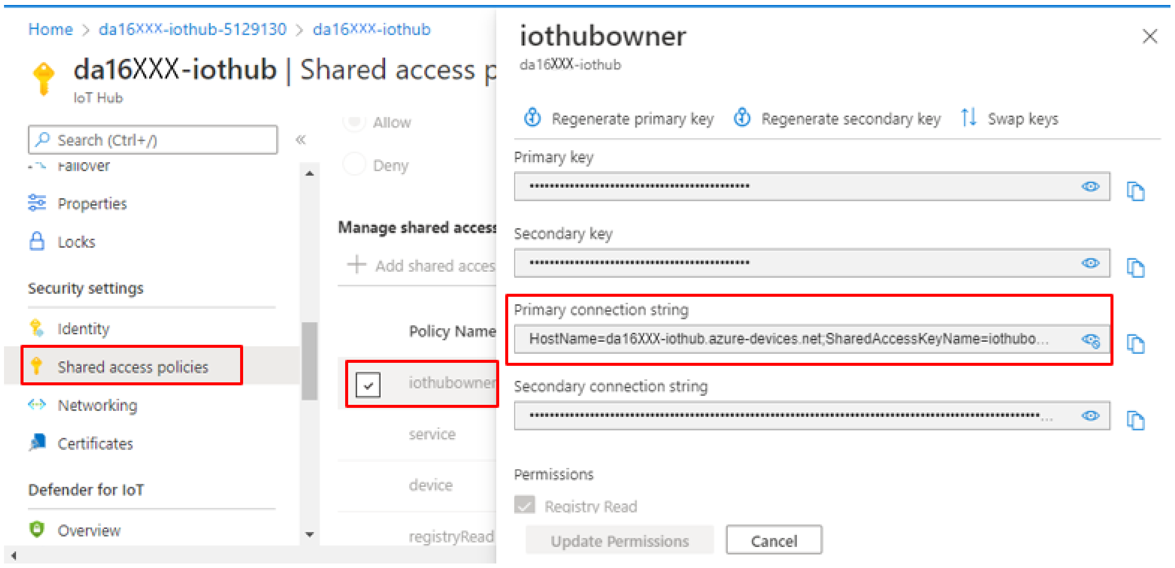
Minimum TLS Version
1.0

Subscription ID
d1f3490f-2644-421d-b8b9-e5d3a8df40da

Tags (edit)
Click here to add tags

Figure 25: Hostname in Azure IoT Hub

- IOT-HUB-Connection-String: Connection string to communicate with Azure IoT Hub



Home > da16XXX-iothub-5129130 > da16XXX-iothub

da16XXX-iothub | Shared access p
IoT Hub

iothubowner
da16XXX-iothub

Primary key
••

Secondary key
••

Primary connection string
HostName=da16XXX-iothub.azure-devices.net;SharedAccessKeyName=iothubo...

Secondary connection string
••...

Permissions
✓ Registry Read

Figure 26: Connection String for Azure IoT Hub

There are two ways to configure these parameters in the DA16200/DA16600:

- Set the default values in the SDK:
  Edit the above settings in app_thing_manager.h, and then compile (for instruction on how to compile the DA16200/DA16600 SDK, see Ref. [1]) and download the firmware image.

```
#define APP_USER_MYHING_NAME              "da16XXX-device-1"  //Device ID
#define APP_USER_MY_DEV_PRIMARY_KEY       "kzsauVFTNfZpO1xJHrgXFDjxZ6t9Jbfmd4PgMvx/ixo="
#define APP_USER_MY_HOST_NAME             "da16XXX-iothub.azure-devices.net"
#define APP_USER_MY_IOTHUB_CONN_STRIN     "HostName= da16XXX-iothub.azure-
devices.net;SharedAccessKeyName=iothubowner;SharedAccessKey=a5UBDdFQJ92jVMhVjwJRzY/LR/GfnrEu289Aa806nUw="
```

- Program the values in the DA16200/DA16600 NVRAM:
  In the DA16200/DA16600 debug console, enter factory mode and set the 4 NVRAM parameters as shown below.

```
[/DA16200 or /DA16600] # nvram
    Command-List is changed, "NVRAM"
[/DA16200 or /DA16600/NVRAM] # setenv APP_THINGNAME da16XXX-device-1
[/DA16200 or /DA16600/NVRAM] # getenv
…….
APP_THINGNAME (STR,17) ....... da16XXX-device-1
```

```
[/DA16200 or /DA16600] # nvram
    Command-List is changed, "NVRAM"
[/DA16200 or /DA16600/NVRAM] # setenv APP_DEV_PRIMARY_KEY kzsauVFTNfZpO1xJHrgXFDjxZ6t9Jbfmd4PgMvx/ixo=
[/DA16200 or /DA16600/NVRAM] # getenv
…….
APP_THINGNAME (STR,17) ....... da16XXX-device-1
APP_DEV_PRIMARY_KEY (STR,45) . kzsauVFTNfZpO1xJHrgXFDjxZ6t9Jbfmd4PgMvx/ixo=
```

```
[/DA16200 or /DA16600] # nvram
    Command-List is changed, "NVRAM"
[/DA16200 or /DA16600/NVRAM] # setenv APP_HOSTNAME da16XXX-iothub.azure-devices.net
 [/DA16200 or /DA16600/NVRAM] # getenv
…….
APP_THINGNAME (STR,17) ....... da16XXX-device-1
APP_DEV_PRIMARY_KEY (STR,45) . kzsauVFTNfZpO1xJHrgXFDjxZ6t9Jbfmd4PgMvx/ixo=
APP_HOSTNAME (STR,43) ........ da16XXX-iothub.azure-devices.net
```

```
[/DA16200 or /DA16600/NVRAM] # nvram
    Command-List is changed, "NVRAM"
[/DA16200 or /DA16600/NVRAM] # setenv APP_IOTHUB_CONN_STRING
Typing data: (setenv value)
        Cancel - CTRL+D, End of Input - CTRL+C or CTRL+Z
HostName= da16XXX-iothub.azure-
devices.net;SharedAccessKeyName=iothubowner;SharedAccessKey=Vmt7BP8BsSwNJRm1tSLJUClibls1ApXjpd67HqF91xQ=
[/DA16200 or /DA16600/NVRAM] # getenv
…….
APP_THINGNAME (STR,17) ....... da16XXX-device-1
APP_DEV_PRIMARY_KEY (STR,45) . kzsauVFTNfZpO1xJHrgXFDjxZ6t9Jbfmd4PgMvx/ixo=
APP_HOSTNAME (STR,43) ........ da16XXX-iothub.azure-devices.net
APP_IOTHUB_CONN_STRING (STR,147)  HostName= da16XXX-iothub.azure-
devices.net;SharedAccessKeyName=iothubowner;SharedAccessKey=Vmt7BP8BsSwNJRm1tSLJUClibls1ApXjpd67HqF91xQ=
```

3.4 Verify Connection Between Azure IoT Hub and DA16200/DA16600

When parameters listed above are configured and firmware is successfully downloaded, boot the DA16200/DA16600 EVK and connect to the Azure IoT Hub. Once connected, the DA16200/DA16600 EVK reports to the Azure IoT Hub and the Azure device twin is updated with the reported items.

The log will appear in the debug console as shown below.

- Logs of the connection between the DA16200 EVK and Azure IoT Hub

```
Wakeup source is 0x4
[dpm_init_retmemory] DPM INIT CONFIGURATION(1)


        ******************************************************
        *             DA16200 SDK Information
        * ---------------------------------------------------
        *
        * - CPU Type        : Cortex-M4 (120MHz)
        * - OS Type         : FreeRTOS 10.4.3
        * - Serial Flash    : 4 MB
        * - SDK Version     : V3.2.2.1 Azure Doorlock Ref.
        * - F/W Version     : FRTOS-GEN01-01-507cc85bd-003442
        * - F/W Build Time  : May  7 2022 17:04:14
        * - Boot Index      : 0
        *
        ******************************************************
```

```
……………………………..

[ azure_twin_dpm_auto_start]


AZURE_IOT on Station Mode for "da16200-device-1"

……………………………

[dpmAPPManager] statusFlag : 23
[dpmAPPManager] DM_NEED_INIT
DM_INIT

Info: [azure_dpm_app_init:544] : AZURE_UPDATE_OTA_UNKNOWN
Info: [azure_dpm_app_init:557] : read AZURE_NVRAM_CONFIG_OTA_STATE init status
[dpmAPPManager] DM_NEED_CONNECTION
DM_NEED_CONNECTION

Info: [create_connection_string:740] : connection string : HostName=da16200-standard-iot-hub.azure-
devices.net;SharedAccessKeyName=iothubowner;SharedAccessKey=Vmt7BP8BsSwNJRm1tSLJUClibls1ApXjpd67HqF91xQ=
Info: [IoTHub_Init:22] : Platform intitializeion OK
Info: [ThreadAPI_Create:55] : thread "azrSubT_0" creation OK (pri=1)
IoTHubDeviceClient_SetMessageCallback()...
Info: [dns_resolver_create:70] : hostName = "da16200-standard-iot-hub.azure-devices.net"
Info: [dns_resolver_create:90] : host IP = "20.194.67.96"
Info: [azure_dpm_app_connect:614] : Root CA registered
[dpmAPPManager] DM_BOOT_WAKEUP
DM_WAKEUP_BOOT

IoTHubDeviceClient_GetTwinAsync()...
IoTHubDeviceClient_SetDeviceMethodCallback()...
IoTHubDeviceClient_SetDeviceTwinCallback()...

Sleep mode 3

DM_FINISH_DEVICE

…………………………………………
```

- Logs of the connection between the DA16600 EVK and Azure IoT Hub

```
Wakeup source is 0x4
[dpm_init_retmemory] DPM INIT CONFIGURATION(1)


        ******************************************************
        *             DA16600 SDK Information
        * ---------------------------------------------------
        *
        * - CPU Type        : Cortex-M4 (120MHz)
        * - OS Type         : FreeRTOS 10.4.3
        * - Serial Flash    : 4 MB
        * - SDK Version     : V3.2.4.0 Azure Doorlock Ref.
        * - F/W Version     : FRTOS-GEN01-01-48589f13c-004594
        * - F/W Build Time  : Aug 29 2022 19:03:29
        * - Boot Index      : 0
        *
        ******************************************************
gpio wakeup enable 00000402
[combo] dpm_boot_type = 1

>>> UART1 : Clock=80000000, BaudRate=115200
>>> UART1 : DMA Enabled ...
[combo] BLE_BOOT_MODE_0
[combo] BLE FW VER to transfer ....
   >>> v_6.0.14.1114.2 (id=1) at bank_1
[combo] BLE FW transfer done
……………………………..

[ azure_twin_dpm_auto_start]


AZURE_IOT on Station Mode for "da16XXX-device-1"
```

```
……………………………

[dpmAPPManager] statusFlag : 23
[dpmAPPManager] DM_NEED_INIT
DM_INIT

Info: [azure_dpm_app_init:544] : AZURE_UPDATE_OTA_UNKNOWN
Info: [azure_dpm_app_init:557] : read AZURE_NVRAM_CONFIG_OTA_STATE init status
[dpmAPPManager] DM_NEED_CONNECTION
DM_NEED_CONNECTION

Info: [create_connection_string:740] : connection string : HostName= da16XXX-iothub.azure-
devices.net;SharedAccessKeyName=iothubowner;SharedAccessKey=Vmt7BP8BsSwNJRm1tSLJUClibls1ApXjpd67HqF91xQ=
Info: [IoTHub_Init:22] : Platform intitializeion OK
Info: [ThreadAPI_Create:55] : thread "azrSubT_0" creation OK (pri=1)
IoTHubDeviceClient_SetMessageCallback()...
Info: [dns_resolver_create:70] : hostName = "da16XXX-iothub.azure-devices.net"
Info: [dns_resolver_create:90] : host IP = "20.194.67.96"
Info: [azure_dpm_app_connect:614] : Root CA registered
[dpmAPPManager] DM_BOOT_WAKEUP
DM_WAKEUP_BOOT

IoTHubDeviceClient_GetTwinAsync()...
IoTHubDeviceClient_SetDeviceMethodCallback()...
IoTHubDeviceClient_SetDeviceTwinCallback()...

Sleep mode 3

DM_FINISH_DEVICE

…………………………………………
```

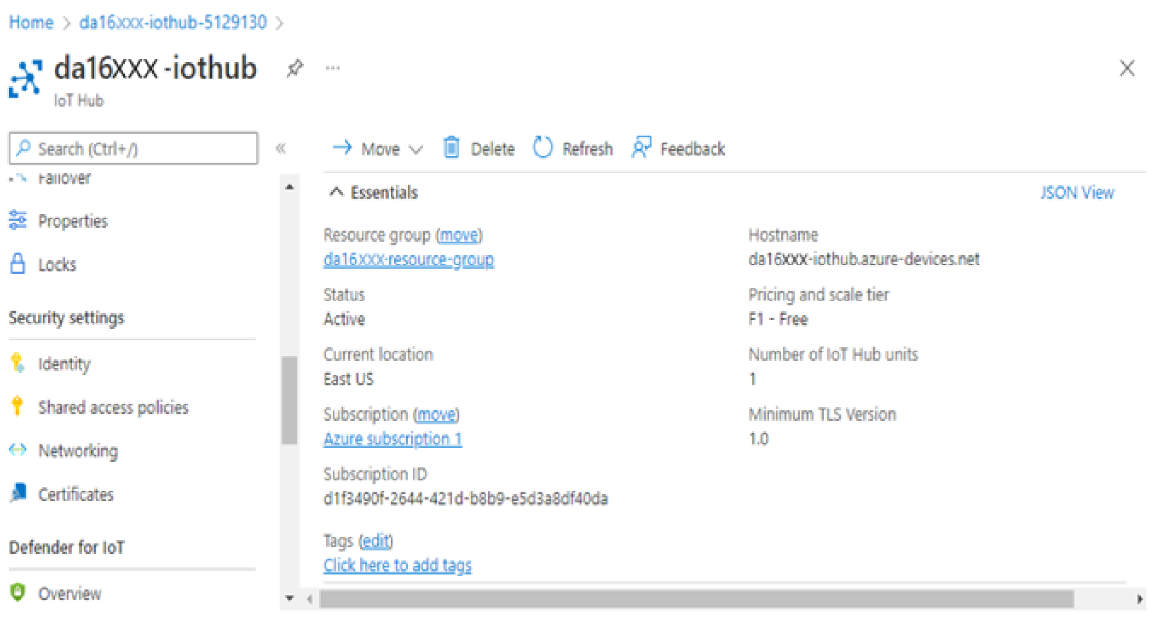
- Logs of items reported to the Azure IoT Hub

```
Info: [deviceTwinCallback:463] : Report =>
{"temperature":23,"battery":100,"doorStateChange":0,"openMethod":"none","doorState":false,"doorBell":false,"do
orOpenMode":0,"OTAupdate":0,"OTAresult":"OTA_UNKNOWN","OTAversion":"none"}
```

- Updated items in the Azure Device Twin

To check updated items in the Azure Device Twin:

a. Go to Azure IoT Hub.



Resource group (move)
da16XXX-resource-group

Status
Active

Current location
East US

Subscription (move)
Azure subscription 1

Subscription ID
d1f3490f-2644-421d-b8b9-e5d3a8df40da

Tags (edit)
Click here to add tags

Hostname
da16XXX-iothub.azure-devices.net

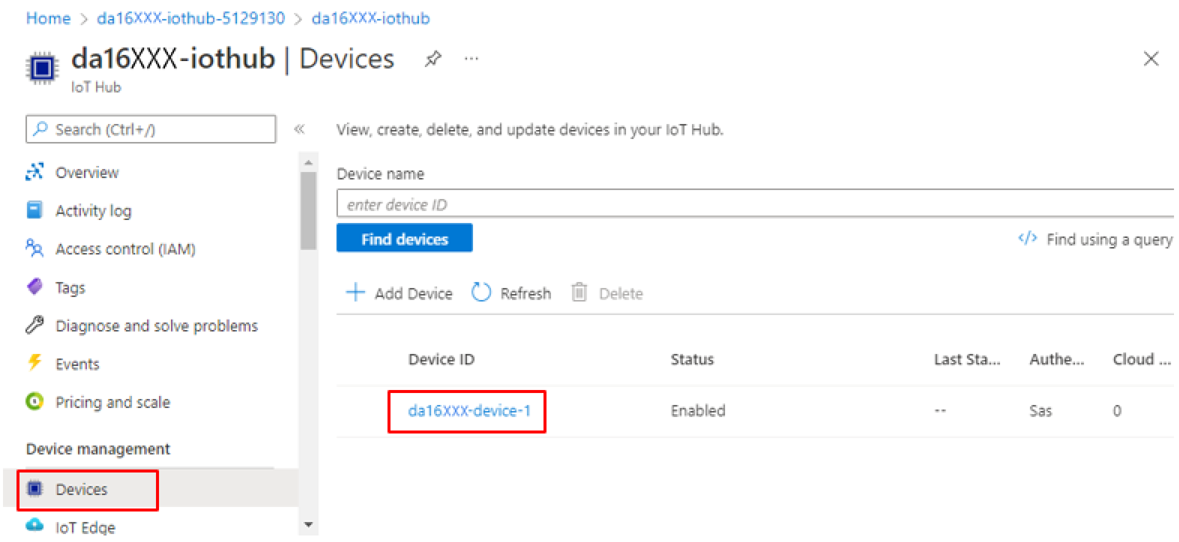
Pricing and scale tier
F1 - Free

Number of IoT Hub units
1

Minimum TLS Version
1.0

Figure 27: Azure IoT Hub

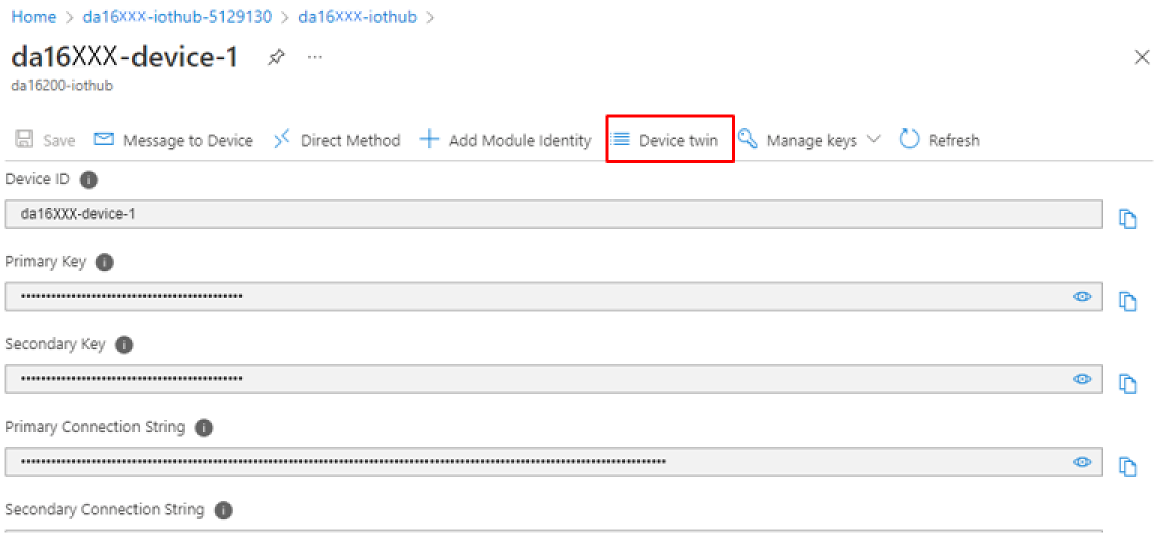
b. Click **Devices**, and then select the **Device ID**.



Device ID	Status	Last Sta...	Authe...	Cloud ...
da16XXX-device-1	Enabled	--	Sas	0

Figure 28: Select Device ID

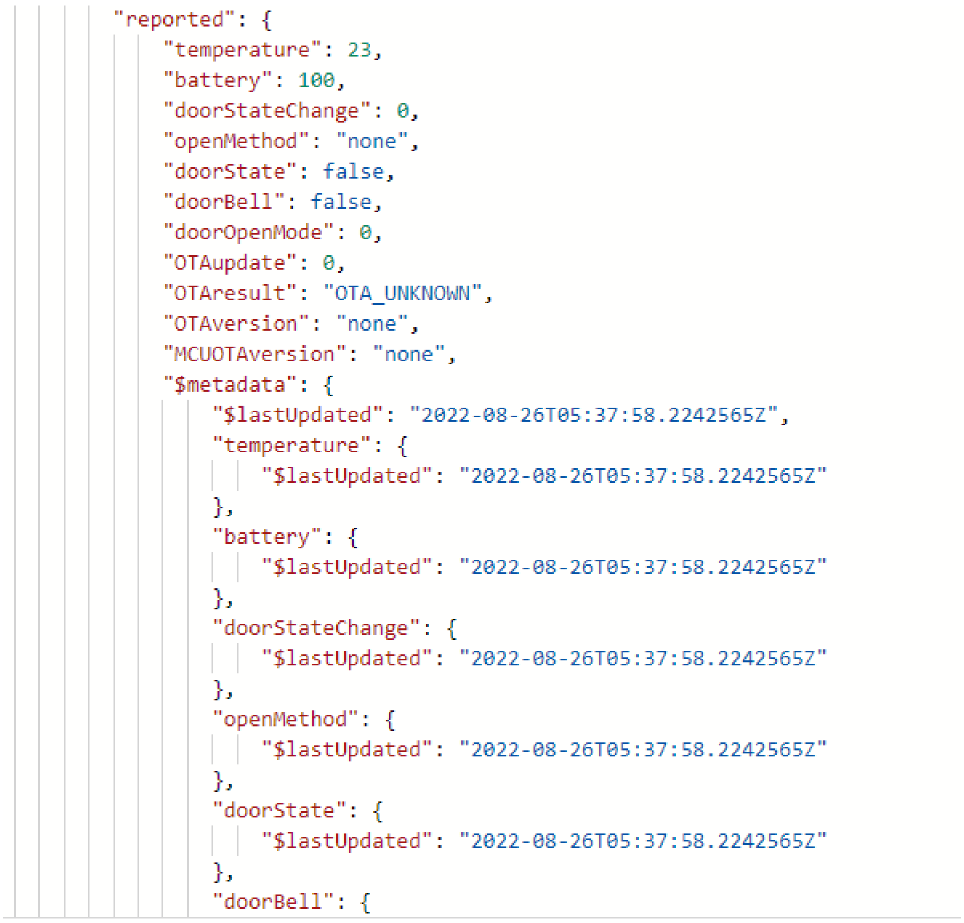
c. Click Device twin.



Device ID: da16XXX-device-1

Figure 29: Device Twin in Azure IoT Hub

Figure 30 shows the property of Device Twin. The information shown in the device twin should match with the information shown in the DA16200/DA16600 console log.



```
"reported": {
    "temperature": 23,
    "battery": 100,
    "doorStateChange": 0,
    "openMethod": "none",
    "doorState": false,
    "doorBell": false,
    "doorOpenMode": 0,
    "OTAupdate": 0,
    "OTAresult": "OTA_UNKNOWN",
    "OTAversion": "none",
    "MCUOTAversion": "none",
    "$metadata": {
        "$lastUpdated": "2022-08-26T05:37:58.2242565Z",
        "temperature": {
            "$lastUpdated": "2022-08-26T05:37:58.2242565Z"
        },
        "battery": {
            "$lastUpdated": "2022-08-26T05:37:58.2242565Z"
        },
        "doorStateChange": {
            "$lastUpdated": "2022-08-26T05:37:58.2242565Z"
        },
        "openMethod": {
            "$lastUpdated": "2022-08-26T05:37:58.2242565Z"
        },
        "doorState": {
            "$lastUpdated": "2022-08-26T05:37:58.2242565Z"
        },
        "doorBell": {
```

Figure 30: Items in Azure Device Twin

4 Door Lock Reference Application

Door lock reference application is available in the official website.

NOTE
Go to the Renesas website (https://www.renesas.com/us/en/products/wireless-connectivity/wi-fi/low-power-wi-fi) and scroll down to the Software Downloads section. Find "Azure IoT Reference" or type it in the search box, and then select the reference package and download.

4.1 Reference Application in DA16200/DA16600

The following components shown in Figure 31 are required to run the application in DA16200/DA16600 via an Internet connection and Microsoft Azure IoT server.

- Azure IoT reference application package
- DA16200/DA1660 EVB
- Router: Connection to internet
- Mobile device: Android/iOS application
- Azure account

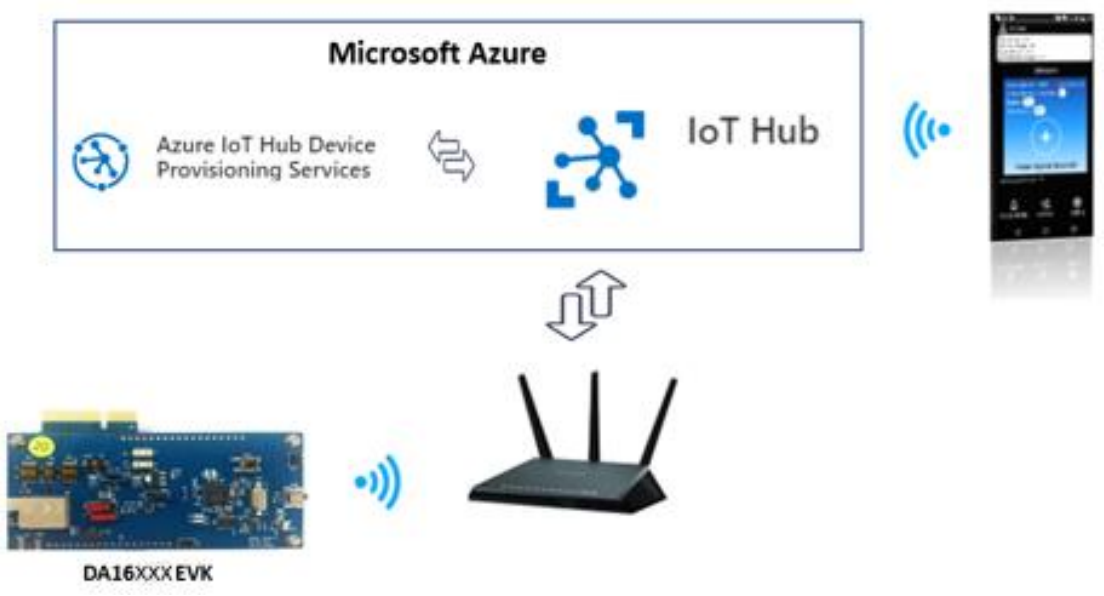


Figure 31: Structure of Azure IoT

Install the mobile application by searching for **DA16200** or **DA16600** on the Google Pay Store or Apple App Store on the mobile devices.

Provisioning is required for connection between DA16200/DA16600 and Router before connecting DA16200/DA16600 with Azure IoT hub. The provisioning can be done with the Renesas Wi-Fi Provisioning app on either an Android or iOS device. See the user manual Ref. [2] for details on how to install and provision the mobile app. Once provisioning is completed, select **Azure IoT** to open Azure application on mobile device as shown in Figure 32.

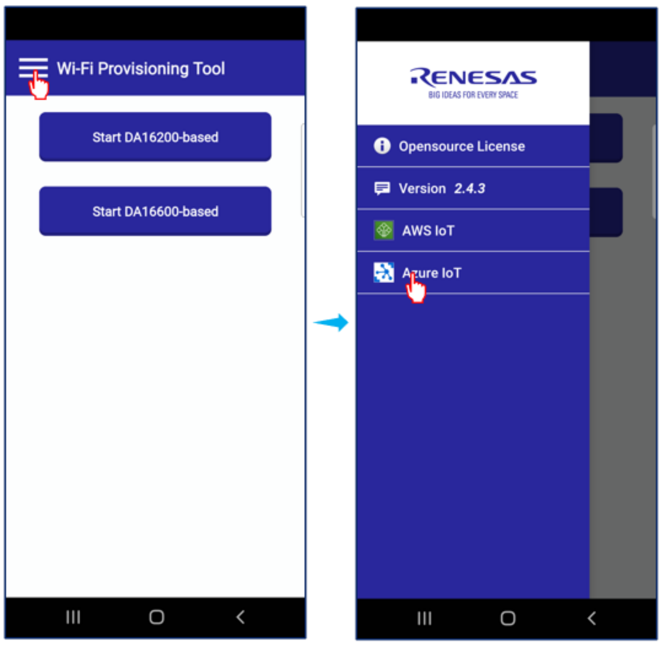


Figure 32: Open Azure IoT on Mobile Device

Figure 33 shows key features of the Azure IoT on the Android phone.

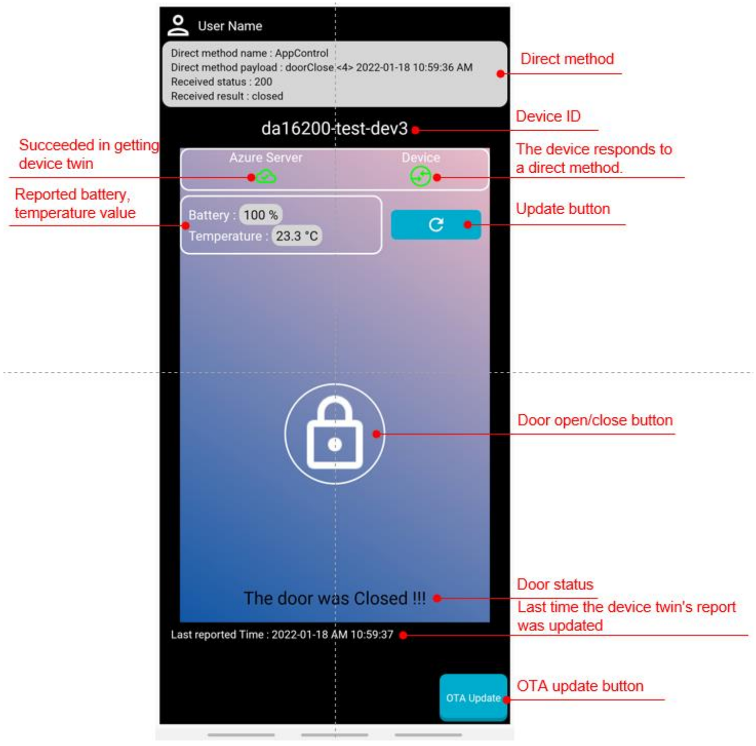

User Name
Direct method name : AppControl
Direct method payload : doorClose<4> 2022-01-18 10:59:36 AM
Received status : 200
Received result : closed
da16200-test-dev3
Azure Server
Device
Battery : 100 %
Temperature : 23.3 °C
The door was Closed !!!
Last reported Time : 2022-01-18 AM 10:59:37
OTA Update
Direct method
Device ID
The device responds to a direct method.
Update button
Succeeded in getting device twin
Reported battery, temperature value
Door open/close button
Door status
Last time the device twin's report was updated
OTA update button

Figure 33: Azure IoT on Mobile Device

4.1.1 Open Door

Figure 34 shows message flows of opening the door.

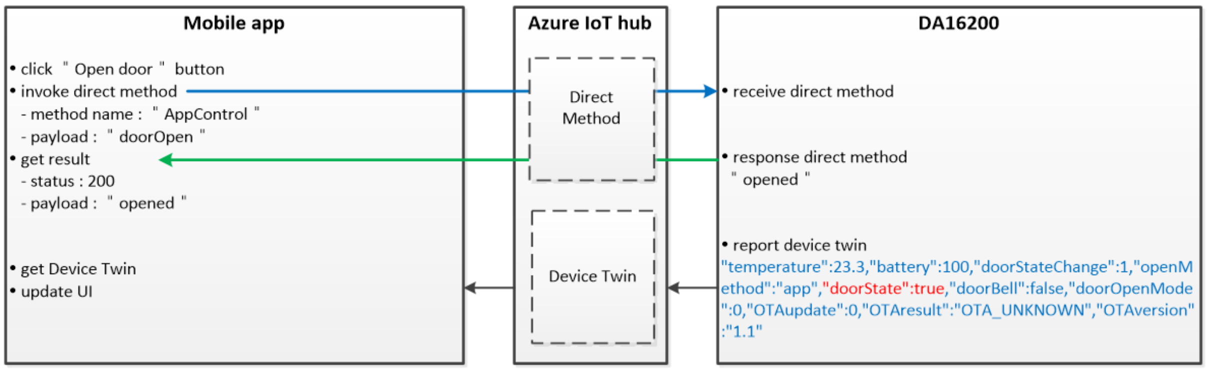
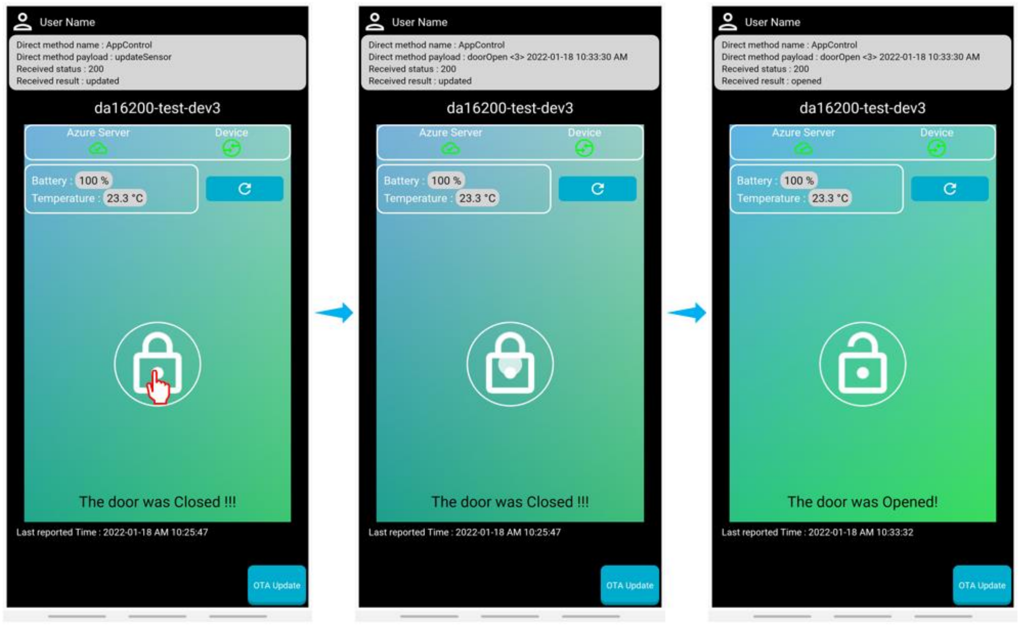


Figure 34: Message Flows of Opening Door

The operation of **opening door** in Android app is shown in Figure 35.

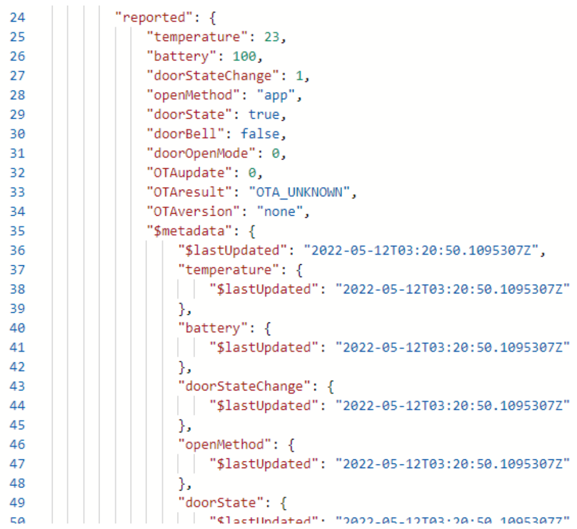

Direct method name : AppControl
Direct method payload : updateSensor
Received status : 200
Received result : updated

Direct method name : AppControl
Direct method payload : doorOpen <3> 2022-01-18 10:33:30 AM
Received status : 200
Received result : updated

Direct method name : AppControl
Direct method payload : doorOpen <3> 2022-01-18 10:33:30 AM
Received status : 200
Received result : opened

Figure 35: Opening Door on Mobile App

Figure 36 shows the device twin of Azure IoT Hub when the operation of opening door is completed.



```
"reported": {
    "temperature": 23,
    "battery": 100,
    "doorStateChange": 1,
    "openMethod": "app",
    "doorState": true,
    "doorBell": false,
    "doorOpenMode": 0,
    "OTAupdate": 0,
    "OTAresult": "OTA_UNKNOWN",
    "OTAversion": "none",
    "$metadata": {
        "$lastUpdated": "2022-05-12T03:20:50.1095307Z",
        "temperature": {
            "$lastUpdated": "2022-05-12T03:20:50.1095307Z"
        },
        "battery": {
            "$lastUpdated": "2022-05-12T03:20:50.1095307Z"
        },
        "doorStateChange": {
            "$lastUpdated": "2022-05-12T03:20:50.1095307Z"
        },
        "openMethod": {
            "$lastUpdated": "2022-05-12T03:20:50.1095307Z"
        },
        "doorState": {
            "$lastUpdated": "2022-05-12T03:20:50.1095307Z"
```

Figure 36: Device Twin When Door Is Open

When the operation of opening door is completed, the console logs of DA16200/DA16600 are displayed as below.

```
[mqttOperationCompleteCallback:2102] MQTT_CLIENT_ON_CONNACK & CONNECTION_ACCEPTED: subscribing topics=0x000d
==> elapsed time: 2 ms,  total time: 207 ms
Info: [app_socket_set_recv_timeout:146] : recv timeout(=120) set OK (socket=0)

[SubscribeToMqttProtocol:2441] mqtt_client_subscribe() bypassed ==> elapsed time: 10 ms,  total time: 217 ms

[app_dpm_set_rcv_ready:633] DPM rcv ready & suscription completed ==> elapsed time: 0 ms,  total time: 217 ms
Info: [deviceMethodCallback:347] : method_name: AppControl, Command payload : 10,
"doorOpen"thods/POST/AppControl/?$rid=3
Info: [deviceMethodCallback:351] : open comm
Info: [deviceMethodCallback:430] : Command response : "opened"

Info: [deviceMethodCallback:435] : Report =>
{"temperature":23,"battery":100,"doorStateChange":1,"openMethod":"app","doorState":true,"doorBell":false,"door
OpenMode":0,"OTAupdate":0,"OTAresult":"OTA_UNKNOWN","OTAversion":"none"}

Info: [publishDeviceTwinGetMsg:1136] : publish reported data OK
Info: [removeExpiredTwinRequests:1260] : ack_waiting_queue timeout & removed

[app_dpm_set_recv_timeout_flag:556] DPM_RCV_OK_SLEEP set ==> elapsed time: 1162 ms,  total time: 1379 ms
Device Twin reported properties update completed with result: 204
```

4.1.2 Close Door

Figure 37 shows message flows of closing door.

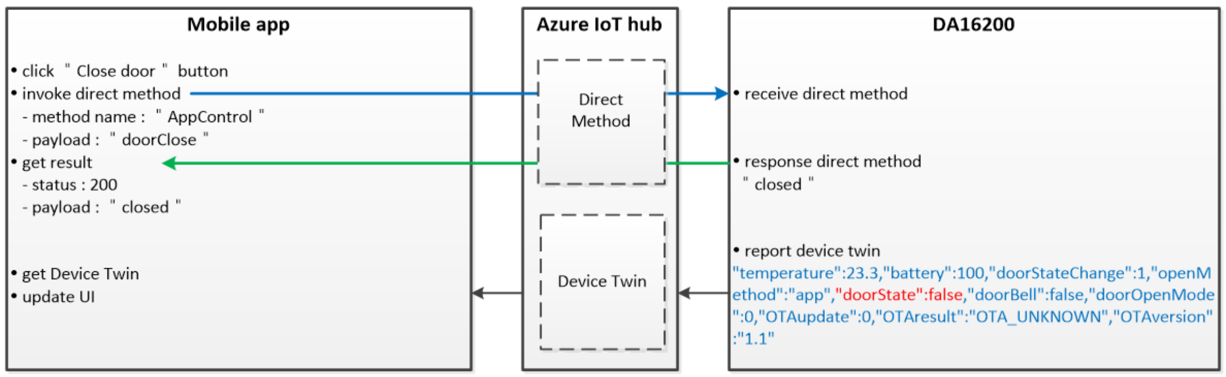
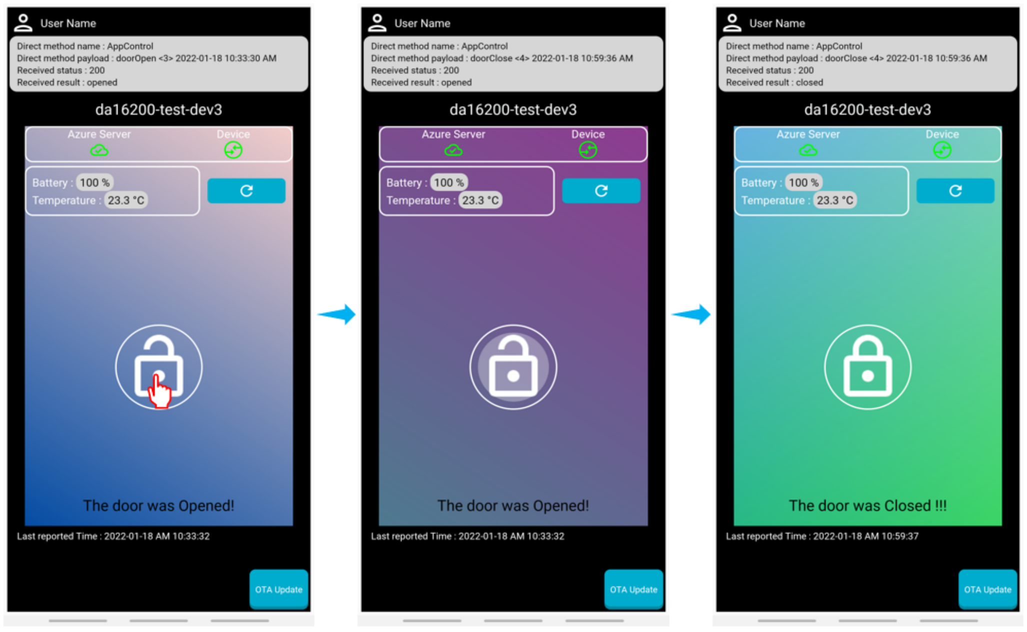

Mobile app
• click " Close door " button
• invoke direct method
- method name : " AppControl "
- payload : " doorClose "
• get result
- status : 200
- payload : " closed "
• get Device Twin
• update UI
Azure IoT hub
Direct Method
Device Twin
DA16200
• receive direct method
• response direct method
" closed "
• report device twin
"temperature":23.3,"battery":100,"doorStateChange":1,"openMethod":"app","doorState":false,"doorBell":false,"doorOpenMode":0,"OTAupdate":0,"OTAresult":"OTA_UNKNOWN","OTAversion":"1.1"

Figure 37: Message Flows of Closing Door

The operation of **closing door** in Android app is shown in Figure 38.



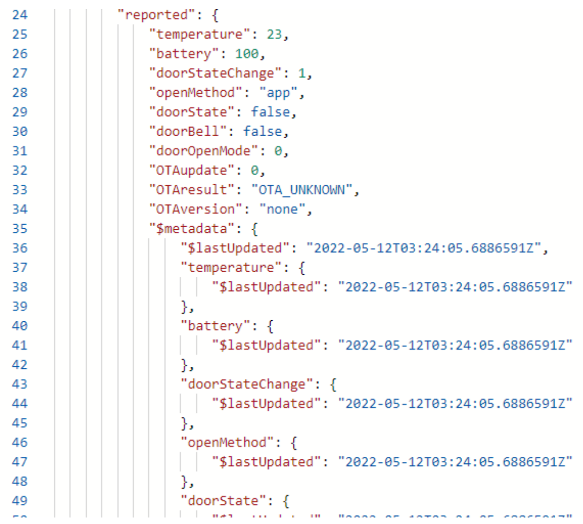
```
Direct method name : AppControl
Direct method payload : doorOpen <3> 2022-01-18 10:33:30 AM
Received status : 200
Received result : opened
```

```
Direct method name : AppControl
Direct method payload : doorClose <4> 2022-01-18 10:59:36 AM
Received status : 200
Received result : opened
```

```
Direct method name : AppControl
Direct method payload : doorClose <4> 2022-01-18 10:59:36 AM
Received status : 200
Received result : closed
```

Figure 38: Closing Door on Mobile App

Figure 40 shows the device twin of Azure IoT Hub when the operation for closing door is completed.



```
"reported": {
    "temperature": 23,
    "battery": 100,
    "doorStateChange": 1,
    "openMethod": "app",
    "doorState": false,
    "doorBell": false,
    "doorOpenMode": 0,
    "OTAupdate": 0,
    "OTAresult": "OTA_UNKNOWN",
    "OTAversion": "none",
    "$metadata": {
        "$lastUpdated": "2022-05-12T03:24:05.6886591Z",
        "temperature": {
            "$lastUpdated": "2022-05-12T03:24:05.6886591Z"
        },
        "battery": {
            "$lastUpdated": "2022-05-12T03:24:05.6886591Z"
        },
        "doorStateChange": {
            "$lastUpdated": "2022-05-12T03:24:05.6886591Z"
        },
        "openMethod": {
            "$lastUpdated": "2022-05-12T03:24:05.6886591Z"
        },
        "doorState": {
```

Figure 39: Device Twin When Door Is Closed

When the operation for closing door is completed, the console logs of DA16200/DA16600 are displayed as below.

```
[mqttOperationCompleteCallback:2102] MQTT_CLIENT_ON_CONNACK & CONNECTION_ACCEPTED: subscribing topics=0x000d
==> elapsed time: 0 ms,  total time: 204 ms
Info: [app_socket_set_recv_timeout:146] : recv timeout(=120) set OK (socket=0)

[SubscribeToMqttProtocol:2441] mqtt_client_subscribe() bypassed ==> elapsed time: 13 ms,  total time: 217 ms

[app_dpm_set_rcv_ready:633] DPM rcv ready & suscription completed ==> elapsed time: 1 ms,  total time: 218 ms
Info: [deviceMethodCallback:347] : method_name: AppControl, Command payload : 11, "doorClose"hods/
Info: [deviceMethodCallback:367] : close comm
Info: [deviceMethodCallback:430] : Command response : "closed"
```

```
Info: [deviceMethodCallback:435] : Report =>
{"temperature":23,"battery":100,"doorStateChange":1,"openMethod":"app","doorState":false,"doorBell":false,"doo
rOpenMode":0,"OTAupdate":0,"OTAresult":"OTA_UNKNOWN","OTAversion":"none"}

Info: [publishDeviceTwinGetMsg:1136] : publish reported data OK
Info: [removeExpiredTwinRequests:1260] : ack_waiting_queue timeout & removed

[app_dpm_set_recv_timeout_flag:556] DPM_RCV_OK_SLEEP set ==> elapsed time: 1141 ms,  total time: 1359 ms
Device Twin reported properties update completed with result: 204
```

4.2 Reference Application in Host MCU

Application in host MCU can control DA16200/DA16600 and connection between the host MCU and mobile phone through Azure IoT server using AT commands. Figure 40 shows the Azure IoT using firmware images for AT commands and host MCU.

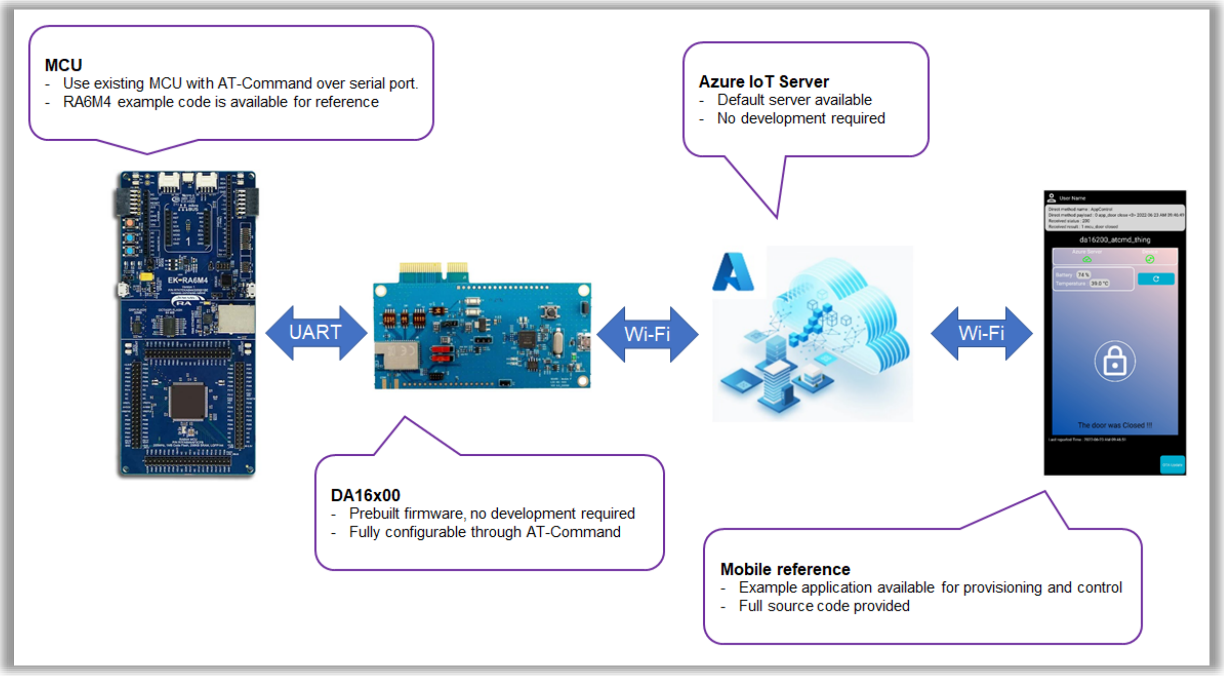

MCU
- Use existing MCU with AT-Command over serial port.
- RA6M4 example code is available for reference
Azure IoT Server
- Default server available
- No development required
UART
Wi-Fi
Wi-Fi
DA16x00
- Prebuilt firmware, no development required
- Fully configurable through AT-Command
Mobile reference
- Example application available for provisioning and control
- Full source code provided

Figure 40: Azure IoT Using Firmware Images for AT Commands and Host MCU

4.2.1 Download Package for Door Lock Reference Application in Host MCU

A firmware image for AT command and application in MCU are available in the official Renesas website (https://www.renesas.com/us/en/products/wireless-connectivity/wi-fi/low-power-wi-fi).

The contents of the package are as follows:

- \DA16200 or \DA16600
  - Firmware images for the DA16200/DA16600 Wi-Fi devices
  - Tera Term script for downloading the firmware images to a DA16200/DA16600 Wi-Fi device
- \DA16200\script (\DA16600\script)
  - Tera Term script which demonstrates how to use AT commands for Azure IoT on a personal computer and the DA16200/DA16600 EVK\doc
  - Getting Stared with AT commands for Azure IoT:
    - Introduces the DA16200/DA16600 AT command for Azure IoT and describes how to set up the development environment and test the examples

  - Describes how to connect an external host to the DA16200/DA16600 EVB for using AT commands for Azure IoT
  - Describes AT commands for Azure IoT command list
- \MCU
  - Sample application project based on the RA6M4 development environment which demonstrates how to use AT commands for Azure IoT

4.2.2 Hardware Connections between DA16200/DA16600 and Host MCU

The hardware components listed in Figure 41 are required to run door lock reference application using AT commands and host MCU.

- DA16200/DA16600 EVB
- EK-RA6M4 board
- Windows laptop or personal computer

In addition, hardware connections listed below are required for each operation.

- UART0: Programming firmware images and monitoring logs from DA16200/DA16600
- UART1 or UART2: AT command interface between DA16200/DA16600 and MCU
- GPIO from host MCU to the DA16200/DA16600 to wake up the DA16200/DA16600 from DPM low power mode (DPM LPM)
- GPIO from the DA16200/DA16600 to host MCU to wake up the MCU in sleep mode

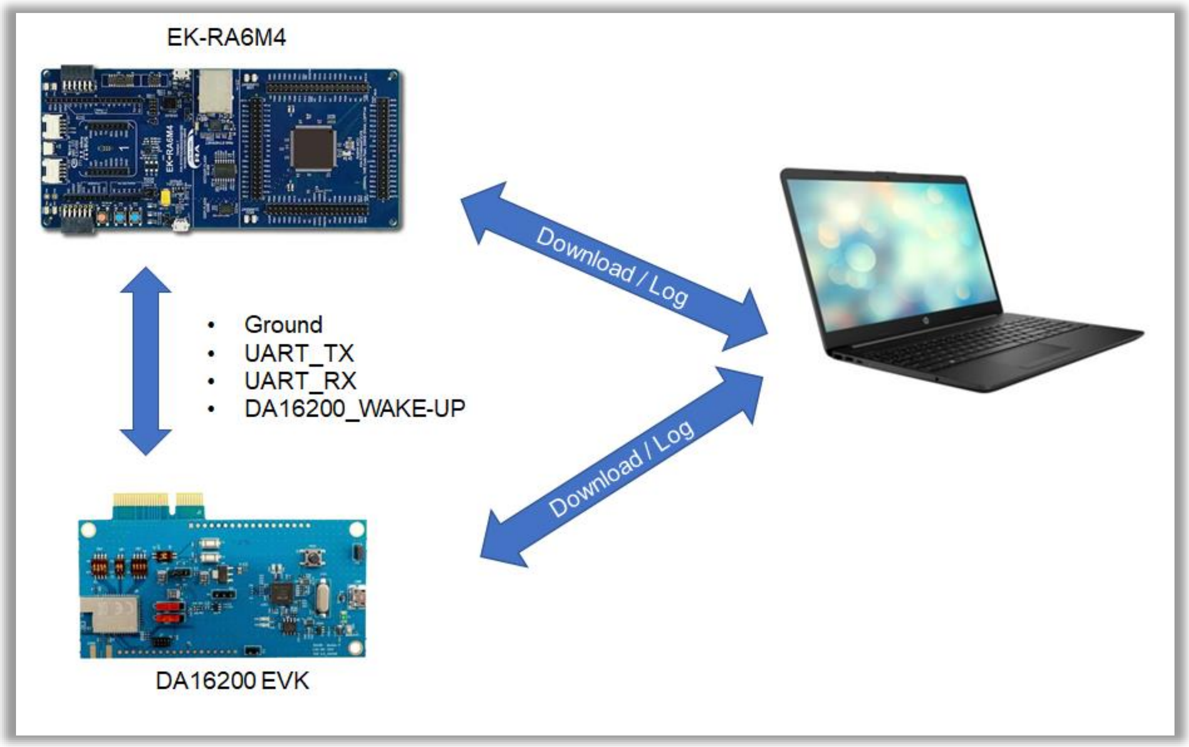


Figure 41: Hardware Configuration

The pin connections between the DA16200/DA16600 EVB and the EK-RA6M4 board are shown in Table 1.

Table 1: Pin Connection

Function	DA16200 EVB		DA16600 EVB		EK-RA6M4 Board	
	Pin Number	Pin Name	Pin Number	Pin Name	Pin Number	Pin Name
Ground	J3.18	GND	J2.12	GND	J24-7	GND
UART_TX	J4.11	TX1/GPIOA_4	J2.2	TX2/GPIOC_6	J23-2	PA10
UART_RX	J4.12	RX1/GPIOA_5	J2.4	RX2/GPIOC_7	J23-1	PA9
DA16200_WAKE_UP	J3.11	RTC_WAKE_UP2	SW1	RTC_WAKE_UP2	J23-6	PB8

4.2.2.1 UART Connection for AT Commands

The default configuration of UART1 (DA16200 EVB) or UART2 (DA16600 EVB) for AT commansd is shown in Table 22.

Table 2: Default Configuration for UART1 or UART2

Settings	Value
Baud Rate	115200
Data Bits	8
Parity	None
Stop Bits	1
Flow Control (HW/SW)	None

The DA16200 EVB uses GPIOA_4 and GPIOA_5 for UART1 TX and UART1 RX, and the DA16600 EVB uses GPIOC_6 and GPIOC_7 for UART2 TX and UART2 RX by default. In addition, GND needs to be connected to the host MCU as shown in Figure 42.

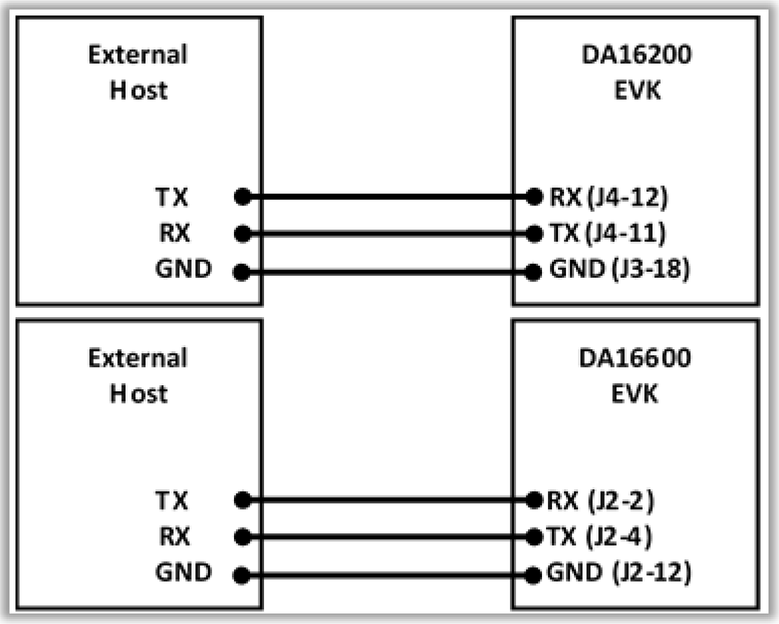


Figure 42: Default UART HW Connection

If GPIO pin configuration is changed using AT commands, other connections for UART1 can be used as shown in Figure 43. The below AT command is to use GPIOA_2 for UART1 TX and GPIOA_3 for UART1 RX. Table 3 shows the pin combination for UART1.

```
AT+AZU=SET NV_PIN_BMUX BMUX_UART1d             // GPIOA_2 and GPIOA_3 for UART1
```

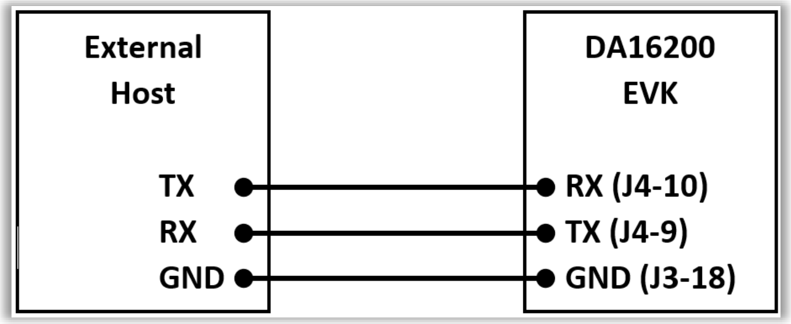


Figure 43: Sample Example of UART1 Connection

Table 3: UART1 Pin Configuration

PIN Mux	GPIO	Signal Name
PIN_AMUX	GPIOA_0	TX
	GPIOA_1	RX
PIN_BMUX	GPIOA_2	TX
	GPIOA_3	RX
PIN_CMUX	GPIOA_4	TX
	GPIOA_5	RX
PIN_DMUX	GPIOA_6	TX
	GPIOA_7	RX

When Dynamic Power Management (DPM) mode is enabled and DA16200/DA16600 is in DPM LPM, the host MCU must wake up the DA16200/DA16600 from DPM LPM using RTC_WAKE_UP. Then, the host MCU can send or receive data over the network in DPM Fully Functional Mode (FFM). The wakeup event is triggered when the GPIO pin of the host MCU changes from Low to High and then back to Low.

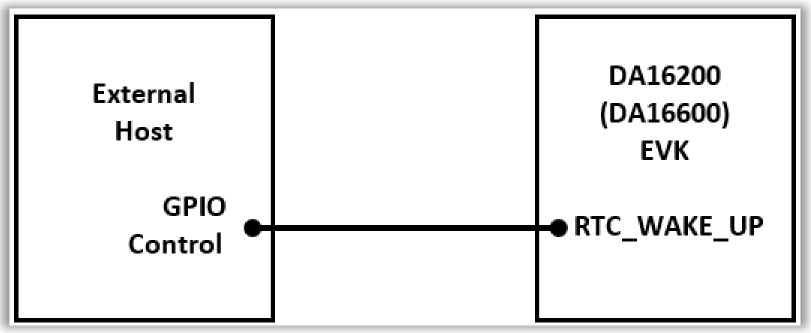


Figure 44: HW Connection for Waking up DA16200/DA16600

The host MCU may be in sleep mode when DA16200/DA16600 wakes up from DPM LPM and needs to send responses to the host MCU. In this scenario, the DA16200/DA16600 needs to wake up the host MCU from sleep using GPIO as shown in Figure 44. This connection is not required if the host MCU does not use sleep mode. GPIOA_11 is available on DA16200/DA16600 EVB for waking up host MCU by default (see Figure 45) and it can be configured using the following AT commands.

```
AT+AZU SET APP_MCU_WKAEUP_PORT GPIO_UNIT_A            // GPIO_A port
AT+AZU SET APP_MCU_WKAEUP_PIN GPIO_PIN11              // GPIO_11
```

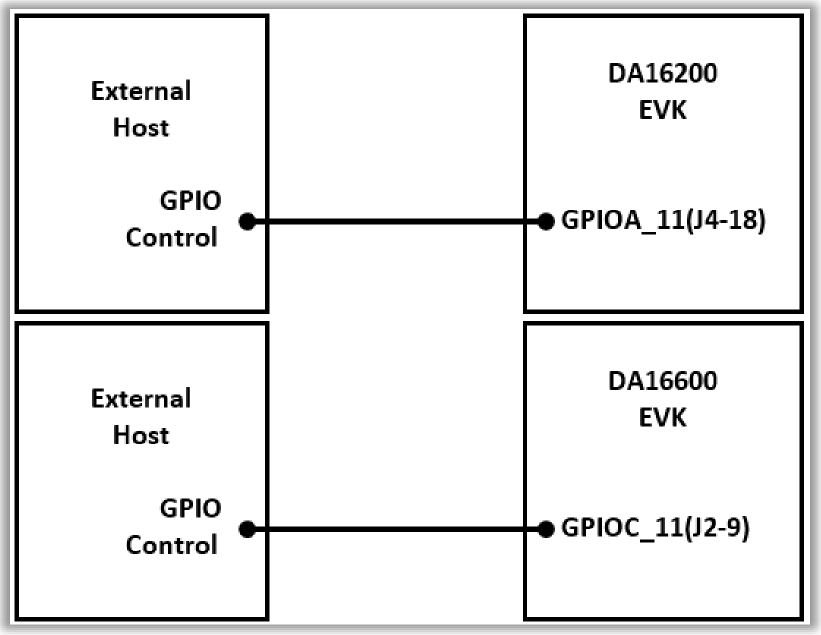


Figure 45: Default Pin Configuration for Waking Up Host MCU

Other GPIOs in the DA16200 EVB can be used for waking up the host MCU as shown in Table 4. For example, GPIOC_6 can be configured for waking up host MCU using the below AT commands (see Figure 46).

```
AT+AZU SET APP_MCU_WKAEUP_PORT GPIO_UNIT_C            // GPIOC port
AT+AZU SET APP_MCU_WKAEUP_PIN GPIO_PIN6               // GPIOC_6
```

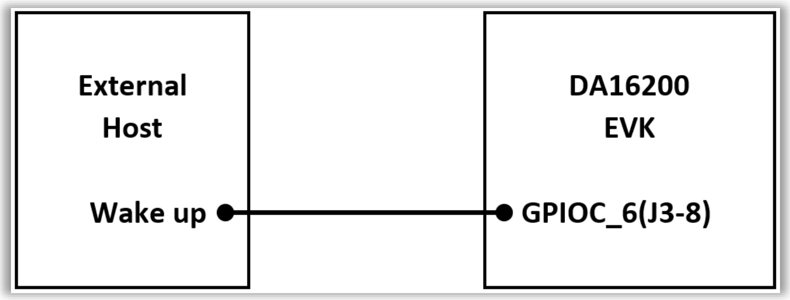


Figure 46: Another Pin Configuration for Waking up Host MCU

Table 4: GPIO Pin Configuration

Port	PIN Mux	GPIO(Port)_Pin
GPIO_UNIT_A	PIN_AMUX	GPIOA_0
		GPIOA_1
	PIN_BMUX	GPIOA_2
		GPIOA_3

Port	PIN Mux	GPIO(Port)_Pin
	PIN_CMUX	GPIOA_4
		GPIOA_5
	PIN_DMUX	GPIOA_6
		GPIOA_7
	PIN_EMUX	GPIOA_8
		GPIOA_9
	PIN_FMUX	GPIOA_10
		GPIOA_11
GPIO_UNIT_C	PIN_UMUX	GPIOC_6
		GPIOC_7
		GPIOC_8

4.2.3 Programming Firmware Images for DA16200/DA16600

When using an EVB for the first time, the firmware must be updated to the latest version. See Ref [1] for details. After programming the firmware image, factory reset is required to enter the Azure-IoT configuration setting mode. This can be done by pushing the “Factory_RST” button for 5 seconds as shown in Figure 47 and Figure 48, and the logs from DA16200/DA16600 are shown in box below.

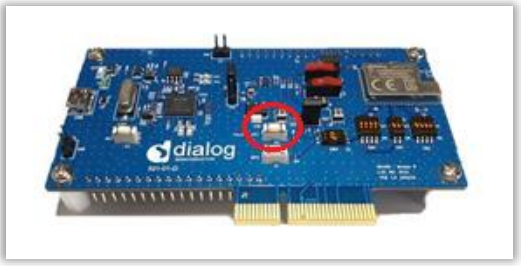


Figure 47: Factory Reset Button on DA16200 EVB

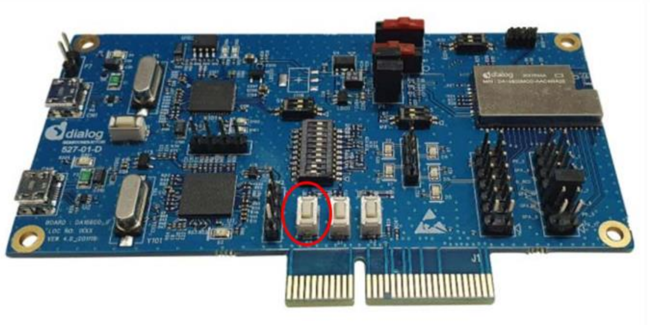


Figure 48: Factory Reset Button on DA16600 EVB

```
Soft-AP is Ready (d4:3d:39:11:5e:73)
>>> UART1 : Clock=80000000, BaudRate=115200
>>> UART1 : DMA Enabled ...
[reg_user_atcmd_help_cb] Start AZURE AT Command
[UART ready notification]
[http_server_task] HTTP-Server Start!!
==========================================
[ AZURE-IOT AT COMMAND ]
[ azure_twin_dpm_auto_start]

AZURE_IOT on Station Mode for "DA16200" \\For DA16600, AZURE_IOT on Station Mode for "DA16600"
===========================================


[azure_twin_dpm_auto_start] mcu_wakeup_port=-1, mcu_wakeup_pin=0x0
default set to mcu_wakeup_port=0, mcu_wakeup_pin=0x800

Root CA: X
Not ready Cert for Azure service ...

invalid AZURE feature...can't start AZURE thread...check again
.. UART ready
```

After the factory reset, the DA16200/DA16600 is now ready to enter the Azure-IoT Configuration Settings.

4.2.4 Configure Components for Testing

The following information are required for testing the application with Azure IoT server:

- Unique device id
- Primary key
- Primary connection string
- Hostname

The information can be set in the source code for host MCU or using the provided scripts in the downloaded package. See Programming Firmware Images section in Ref [1] for how to run the macro script. The scripts are located in the following location.

```
\DA16200\script\doorlock.ttl
```

```
In order to use this script on DA16200, the console should be prompt

;after setting the DA16200 to STA mode, SNTP client enable, and no DPM mode in easy setup through the
console.


;set configurations with DA16200's console


;set features

sendln "user"

;set board type

sendln "SET APP_BOARD_FEATURE EVK"

mpause 400

;set your thingname

sendln "SET APP_THINGNAME da16200_atcmd_thing"

mpause 400

;set primary key of device

sendln "SET APP_DEV_PRIMARY_KEY dWKJjGHX9TiCOPheiKyRjjcmmUtBBZsuASBB+K6OoGs="

mpause 400

;set hostname of device

sendln "SET APP_HOSTNAME azure-atcmd-iothub.azure-devices.net"

mpause 400

;set connection string of device

sendln "SET APP_IOTHUB_CONN_STRING HostName=azure-atcmd-iothub.azure-
devices.net;SharedAccessKeyName=iothubowner;SharedAccessKey=Xq970WfOFvEmVU9Ixcm96zPGTKh1ztcoxF64QBmyvU8="

mpause 400
```

The MCU source code can be found in the following file:

`\MCU\RA6M4\Src\atcmd\at_cmd.c`.

```
const char* cmd_set_azure_cfg[MAX_CFG_NUM] =
{
  "\r\nAT+"PLATFORM" SET APP_BOARD_FEATURE EVK\r\n",
  "\r\nAT+"PLATFORM" SET APP_THINGNAME da16200_atcmd_thing\r\n",
  "\r\nAT+"PLATFORM" SET APP_DEV_PRIMARY_KEY dWKJjGHX9TiCOPheiKyRjjcmmUtBBZsuASBB+K6OoGs=\r\n",
  "\r\nAT+"PLATFORM" SET APP_HOSTNAME azure-atcmd-iothub.azure-devices.net\r\n",
  "\r\nAT+"PLATFORM" SET APP_IOTHUB_CONN_STRING HostName=azure-atcmd-iothub.azure-
devices.net;SharedAccessKeyName=iothubowner;SharedAccessKey=Xq970WfOFvEmVU9Ixcm96zPGTKh1ztcoxF64QBmyvU8=\r\n"
,
  "\r\nAT+"PLATFORM" CFG 0 app_door 1 2\r\n",             /* mcu sub. str */
  "\r\nAT+"PLATFORM" CFG 1 mcu_door 1 0\r\n",             /* mcu pub. str */
  "\r\nAT+"PLATFORM" CFG 2 battery 0 1\r\n",              /* shadow   int */
  "\r\nAT+"PLATFORM" CFG 3 temperature 2 1\r\n",          /* shadow   float */
  "\r\nAT+"PLATFORM" CFG 4 doorState 1 1\r\n",            /* shadow   str */
  "\r\nAT+"PLATFORM" CFG 5 app_shadow 1 2\r\n",           /* mcu sub. str */
  "\r\nAT+"PLATFORM" CFG 6 mcu_shadow 1 0\r\n",           /* mcu pub. str */
"\r\nAT+"PLATFORM" CFG 7 openMethod 1 1\r\n",          /* shadow   str */
"\r\nAT+"PLATFORM" SET SLEEP_MODE 3\r\n",
  "\r\nAT+"PLATFORM" SET USE_DPM 1\r\n",
  "\r\nAT+"PLATFORM" SET RTC_TIME 1740\r\n",              /* 1740 */
  "\r\nAT+"PLATFORM" SET DPM_KEEP_ALIVE 30000\r\n",
  "\r\nAT+"PLATFORM" SET USE_WAKE_UP 0\r\n",
  "\r\nAT+"PLATFORM" SET TIM_WAKE_UP 10\r\n",
  "\r\nAT+"PLATFORM" SET APP_MCU_WKAEUP_PORT GPIO_UNIT_A\r\n",    /* GPIO_UNIT_A or GPIO_UNIT_C */
  "\r\nAT+"PLATFORM" SET APP_MCU_WKAEUP_PIN GPIO_PIN11\r\n"      /* GPIO_PIN0 ~ GPIO_PIN11 or
GPIO_PIN6~GPIO_PIN8 */
};
```

4.2.5 Test without Host MCU

If the host MCU is not available, the Azure IoT commands can be tested with the script provided in the downloaded package.

Door lock for two-way communication: `\DA16200\script\doorlock.ttl`.

NOTE
The example script only supports initial value setting. To fully verify the operation of the AT commands, the host MCU should be used for interacting with the server and application.

4.2.6 Test with Host MCU

E^{2}studio is required for building source code for host MCU and programing the images to host MCU. Visit the Renesas website (https://www.renesas.com/us/en/software-tool/e-studio) for downloading and installing e^{2}studio. After installing e^{2}stuido, follow the steps for building and programming.

1. Import the `project` file at `\MCU\RA6M4\` (see Figure 49).

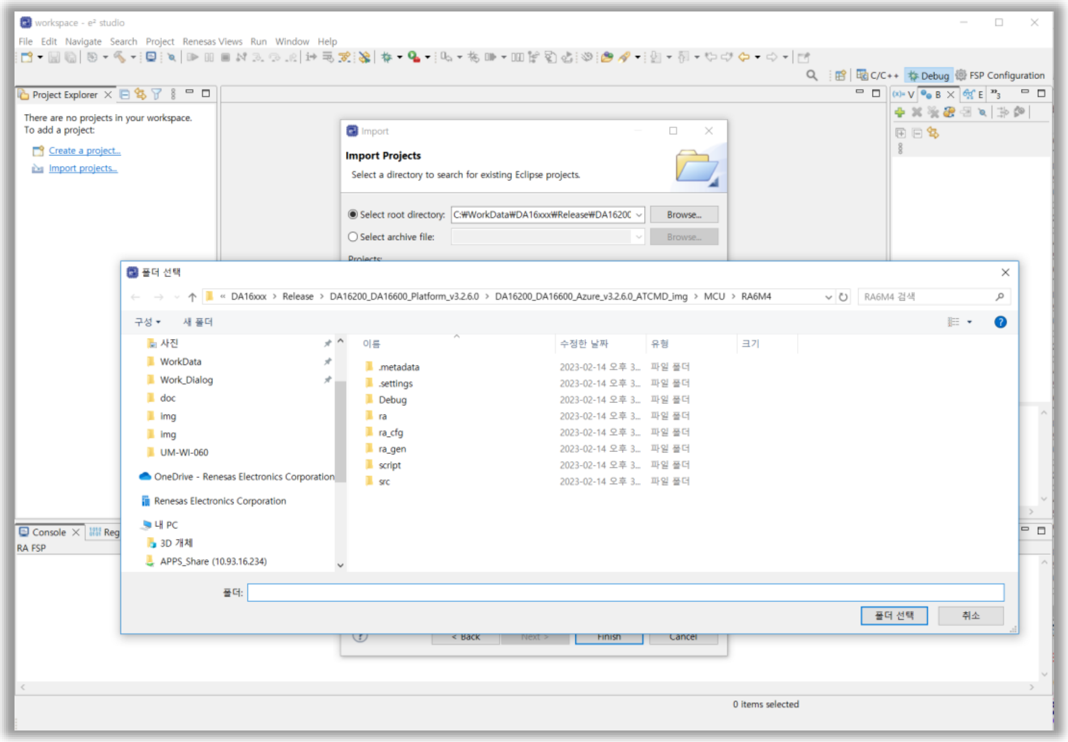

Import Projects
Select a directory to search for existing Eclipse projects.
Select root directory: C:₩WorkData₩DA16xxx₩Release₩DA1620
Select archive file:
Browse...
DA16xxx › Release › DA16200_DA16600_Platform_v3.2.6.0 › DA16200_DA16600_Azure_v3.2.6.0_ATCMD_img › MCU › RA6M4
.metadata
.settings
Debug
ra
ra_cfg
ra_gen
script
src
폴더 선택
취소

Figure 49: e^{2}studio Project File

NOTE
When connecting to the RA6M4 MCU for the first time or changing the configuration, follow the step 2 to set up the FSP configuration.

2. Select Configurations.xml to set FSP configuration of the RA6M4 MCU (see Figure 50).

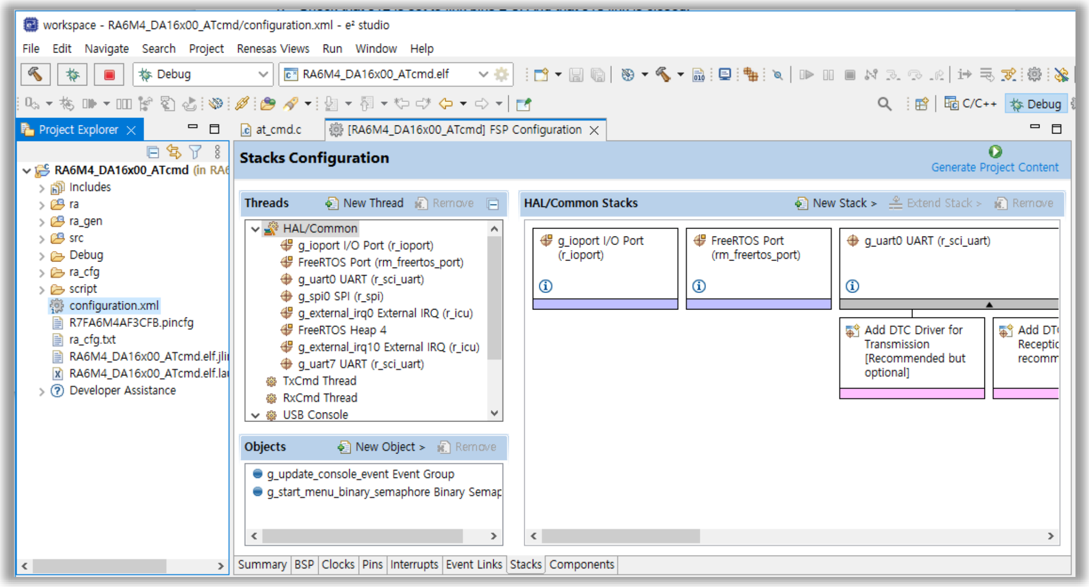
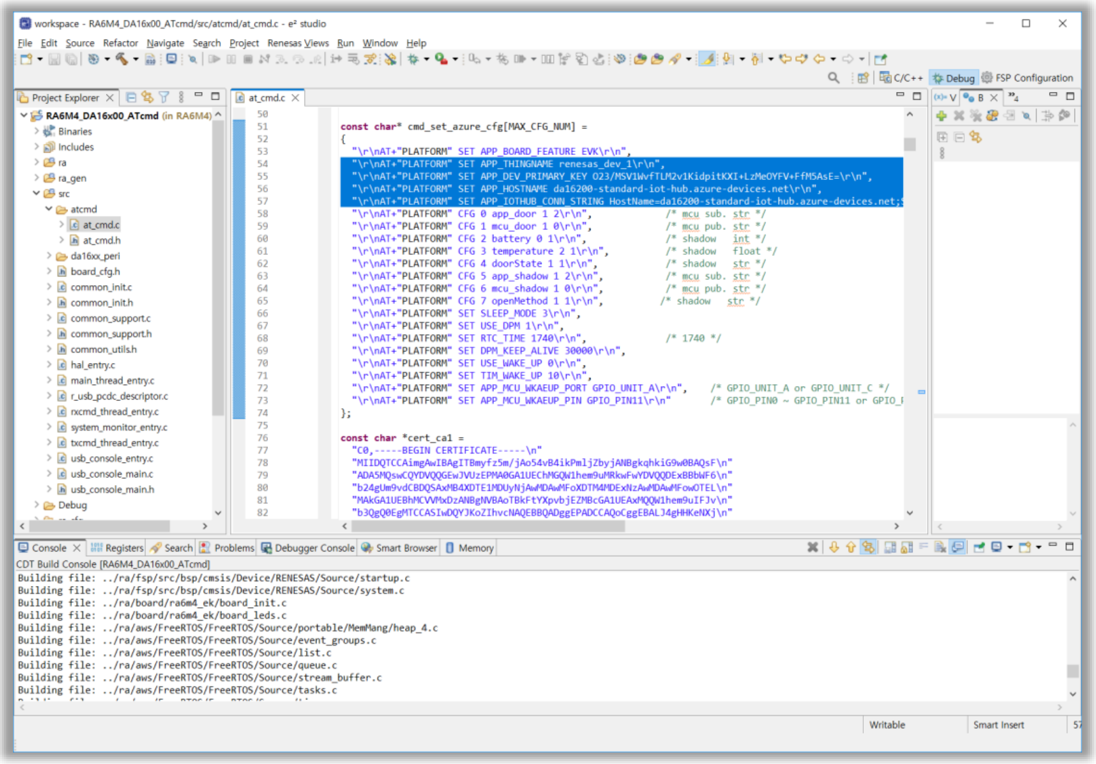

workspace - RA6M4_DA16x00_ATcmd/configuration.xml - e² studio
Project Explorer
RA6M4_DA16x00_ATcmd
Includes
ra
ra_gen
src
Debug
ra_cfg
script
configuration.xml
R7FA6M4AF3CFB.pincfg
ra_cfg.txt
Developer Assistance
[RA6M4_DA16x00_ATcmd] FSP Configuration
Stacks Configuration
Generate Project Content
Threads
New Thread
HAL/Common
g_ioport I/O Port (r_ioport)
FreeRTOS Port (rm_freertos_port)
g_uart0 UART (r_sci_uart)
g_spi0 SPI (r_spi)
g_external_irq0 External IRQ (r_icu)
FreeRTOS Heap 4
g_external_irq10 External IRQ (r_icu)
g_uart7 UART (r_sci_uart)
TxCmd Thread
RxCmd Thread
USB Console
Objects
g_update_console_event Event Group
g_start_menu_binary_semaphore Binary Semap
HAL/Common Stacks
New Stack >
Add DTC Driver for Transmission [Recommended but optional]
Summary
BSP
Clocks
Pins
Interrupts
Event Links
Stacks
Components

Figure 50: FSP Configuration

3. Use the “device id" (thingname), “primary key”, “hostname”, and “primary connection string” received from the FAE to test without setting up a server.
4. Change the configuration to the received configuration (see Figure 51).



```
const char* cmd_set_azure_cfg[MAX_CFG_NUM] =
{
  "\r\nAT+"PLATFORM" SET APP_BOARD_FEATURE EVK\r\n",
  "\r\nAT+"PLATFORM" SET APP_THINGNAME renesas_dev_1\r\n",
  "\r\nAT+"PLATFORM" SET APP_DEV_PRIMARY_KEY O23/MSV1WvfTLM2v1KidpitKXI+LzMeOYFV+FfM5AsE=\r\n",
  "\r\nAT+"PLATFORM" SET APP_HOSTNAME da16200-standard-iot-hub.azure-devices.net\r\n",
  "\r\nAT+"PLATFORM" SET APP_IOTHUB_CONN_STRING HostName=da16200-standard-iot-hub.azure-devices.net;
  "\r\nAT+"PLATFORM" CFG 0 app_door 1 2\r\n",          /* mcu sub. str */
  "\r\nAT+"PLATFORM" CFG 1 mcu_door 1 0\r\n",          /* mcu pub. str */
  "\r\nAT+"PLATFORM" CFG 2 battery 0 1\r\n",           /* shadow   int */
  "\r\nAT+"PLATFORM" CFG 3 temperature 2 1\r\n",       /* shadow   float */
  "\r\nAT+"PLATFORM" CFG 4 doorState 1 1\r\n",         /* shadow   str */
  "\r\nAT+"PLATFORM" CFG 5 app_shadow 1 2\r\n",        /* mcu sub. str */
  "\r\nAT+"PLATFORM" CFG 6 mcu_shadow 1 0\r\n",        /* mcu pub. str */
  "\r\nAT+"PLATFORM" CFG 7 openMethod 1 1\r\n",       /* shadow   str */
  "\r\nAT+"PLATFORM" SET SLEEP_MODE 3\r\n",
  "\r\nAT+"PLATFORM" SET USE_DPM 1\r\n",
  "\r\nAT+"PLATFORM" SET RTC_TIME 1740\r\n",           /* 1740 */
  "\r\nAT+"PLATFORM" SET DPM_KEEP_ALIVE 30000\r\n",
  "\r\nAT+"PLATFORM" SET USE_WAKE_UP 0\r\n",
  "\r\nAT+"PLATFORM" SET TIM_WAKE_UP 10\r\n",
  "\r\nAT+"PLATFORM" SET APP_MCU_WKAEUP_PORT GPIO_UNIT_A\r\n",    /* GPIO_UNIT_A or GPIO_UNIT_C */
  "\r\nAT+"PLATFORM" SET APP_MCU_WKAEUP_PIN GPIO_PIN11\r\n"      /* GPIO_PIN0 ~ GPIO_PIN11 or GPIO_
};

const char *cert_ca1 =
  "C0,-----BEGIN CERTIFICATE-----\n"
  "MIIDQTCCAimgAwIBAgITBmyfz5m/jAo54vB4ikPmljZbyjANBgkqhkiG9w0BAQsF\n"
  "ADA5MQswCQYDVQQGEwJVUzEPMA0GA1UEChMGQW1hem9uMRkwFwYDVQQDExBBbWF6\n"
  "b24gUm9vdCBDQSAxMB4XDTE1MDUyNjAwMDAwMFoXDTM4MDExNzAwMDAwMFowOTEL\n"
  "MAkGA1UEBhMCVVMxDzANBgNVBAoTBkFtYXpvbjEZMBcGA1UEAxMQQW1hem9uIFJv\n"
  "b3QgQ0EgMTCCASIwDQYJKoZIhvcNAQEBBQADggEPADCCAQoCggEBALJ4gHHKeNXj\n"
```

```
CDT Build Console [RA6M4_DA16x00_ATcmd]
Building file: ../ra/fsp/src/bsp/cmsis/Device/RENESAS/Source/startup.c
Building file: ../ra/fsp/src/bsp/cmsis/Device/RENESAS/Source/system.c
Building file: ../ra/board/ra6m4_ek/board_init.c
Building file: ../ra/board/ra6m4_ek/board_leds.c
Building file: ../ra/aws/FreeRTOS/FreeRTOS/Source/portable/MemMang/heap_4.c
Building file: ../ra/aws/FreeRTOS/FreeRTOS/Source/event_groups.c
Building file: ../ra/aws/FreeRTOS/FreeRTOS/Source/list.c
Building file: ../ra/aws/FreeRTOS/FreeRTOS/Source/queue.c
Building file: ../ra/aws/FreeRTOS/FreeRTOS/Source/stream_buffer.c
Building file: ../ra/aws/FreeRTOS/FreeRTOS/Source/tasks.c
```

Figure 51: Thingname/Primarykey/Hostname/IotHub Connection String

5. Select `project > Build all` to build (see Figure 52).

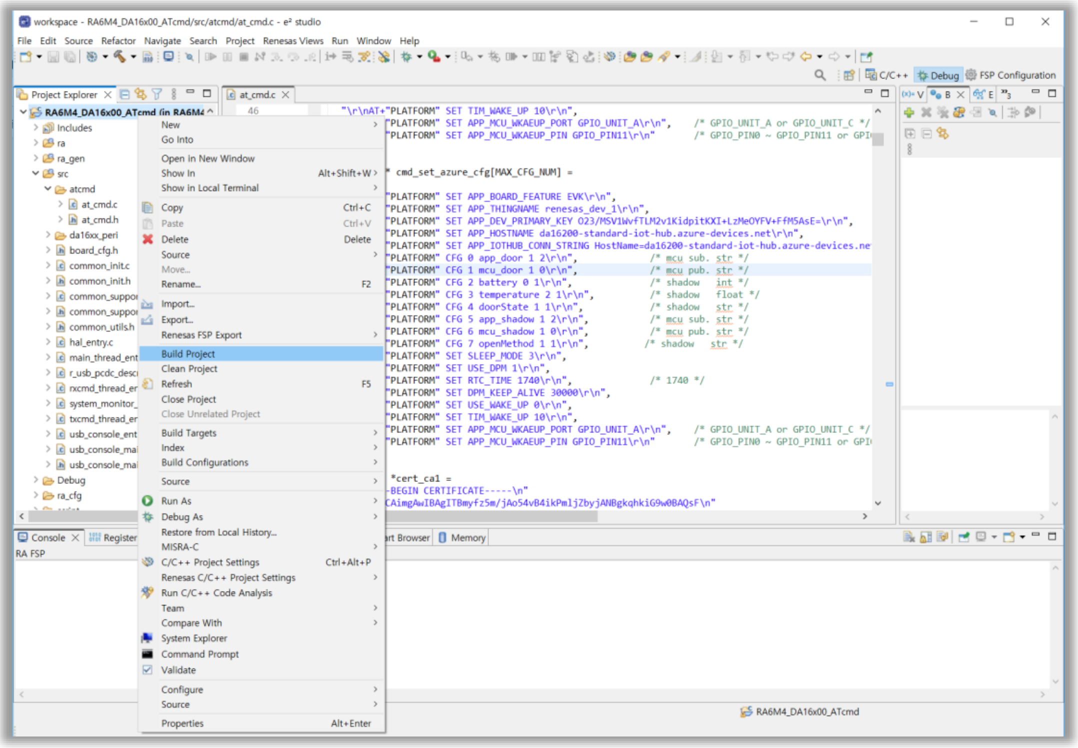
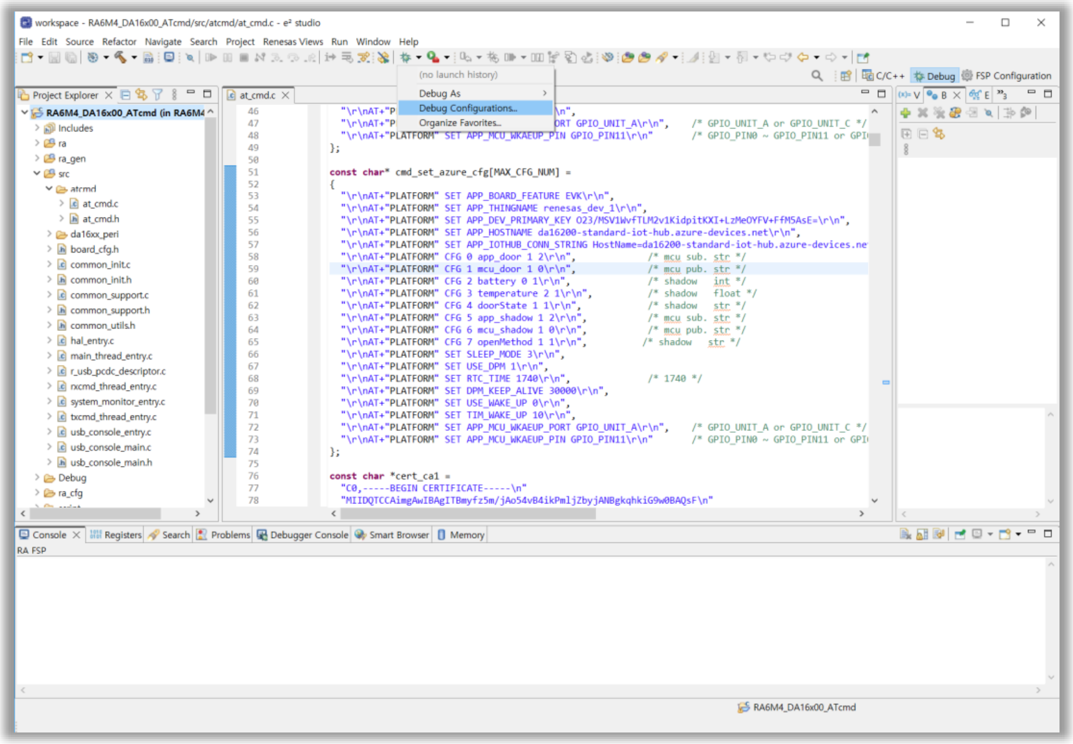


Figure 52: Build Project

6. Select `Debug Configurations` to set the connection to the RA6M4 MCU (see Figure 53).



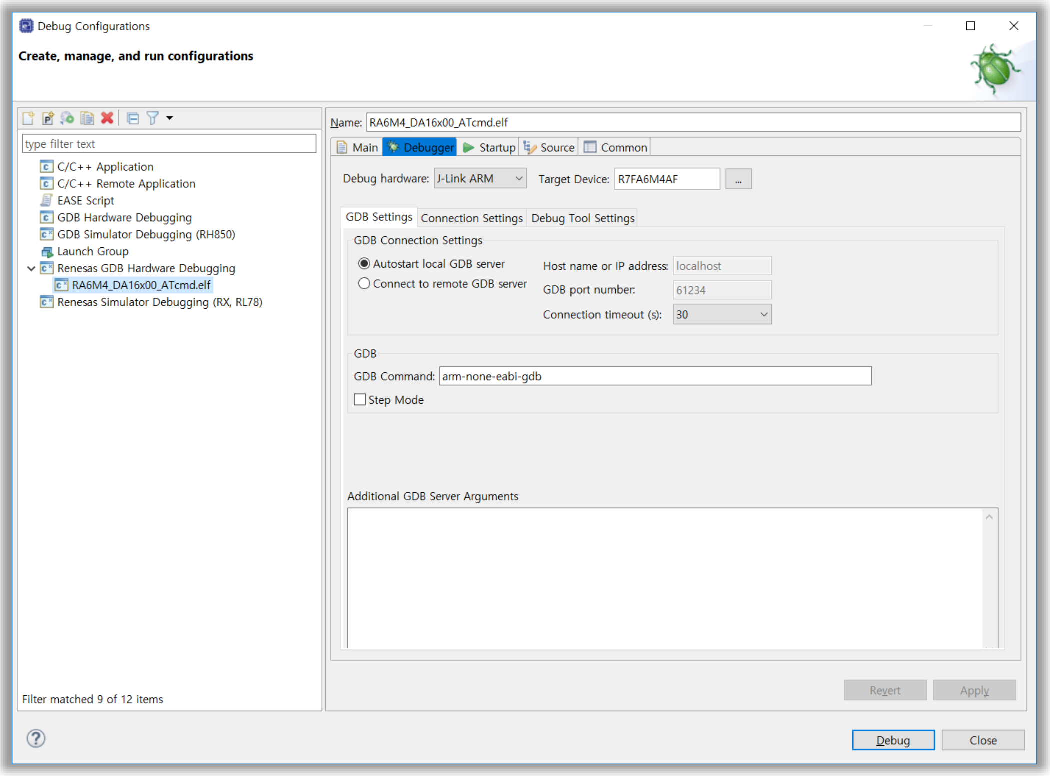
```
  "\r\nAT+"P                                        \n",
  "\r\nAT+"P                                        ORT GPIO_UNIT_A\r\n",    /* GPIO_UNIT_A or GPIO_UNIT_C */
  "\r\nAT+"PLATFORM" SET APP_MCU_WKAEUP_PIN GPIO_PIN11\r\n"       /* GPIO_PIN0 ~ GPIO_PIN11 or GPI
};

const char* cmd_set_azure_cfg[MAX_CFG_NUM] =
{
  "\r\nAT+"PLATFORM" SET APP_BOARD_FEATURE EVK\r\n",
  "\r\nAT+"PLATFORM" SET APP_THINGNAME renesas_dev_1\r\n",
  "\r\nAT+"PLATFORM" SET APP_DEV_PRIMARY_KEY O23/MSV1WvfTLM2v1KidpitKXI+LzMeOYFV+FfM5AsE=\r\n",
  "\r\nAT+"PLATFORM" SET APP_HOSTNAME da16200-standard-iot-hub.azure-devices.net\r\n",
  "\r\nAT+"PLATFORM" SET APP_IOTHUB_CONN_STRING HostName=da16200-standard-iot-hub.azure-devices.ne
  "\r\nAT+"PLATFORM" CFG 0 app_door 1 2\r\n",             /* mcu sub. str */
  "\r\nAT+"PLATFORM" CFG 1 mcu_door 1 0\r\n",             /* mcu pub. str */
  "\r\nAT+"PLATFORM" CFG 2 battery 0 1\r\n",              /* shadow   int */
  "\r\nAT+"PLATFORM" CFG 3 temperature 2 1\r\n",          /* shadow   float */
  "\r\nAT+"PLATFORM" CFG 4 doorState 1 1\r\n",            /* shadow   str */
  "\r\nAT+"PLATFORM" CFG 5 app_shadow 1 2\r\n",           /* mcu sub. str */
  "\r\nAT+"PLATFORM" CFG 6 mcu_shadow 1 0\r\n",           /* mcu pub. str */
  "\r\nAT+"PLATFORM" CFG 7 openMethod 1 1\r\n",          /* shadow   str */
  "\r\nAT+"PLATFORM" SET SLEEP_MODE 3\r\n",
  "\r\nAT+"PLATFORM" SET USE_DPM 1\r\n",
  "\r\nAT+"PLATFORM" SET RTC_TIME 1740\r\n",              /* 1740 */
  "\r\nAT+"PLATFORM" SET DPM_KEEP_ALIVE 30000\r\n",
  "\r\nAT+"PLATFORM" SET USE_WAKE_UP 0\r\n",
  "\r\nAT+"PLATFORM" SET TIM_WAKE_UP 10\r\n",
  "\r\nAT+"PLATFORM" SET APP_MCU_WKAEUP_PORT GPIO_UNIT_A\r\n",    /* GPIO_UNIT_A or GPIO_UNIT_C */
  "\r\nAT+"PLATFORM" SET APP_MCU_WKAEUP_PIN GPIO_PIN11\r\n"       /* GPIO_PIN0 ~ GPIO_PIN11 or GPI
};

const char *cert_ca1 =
  "C0,-----BEGIN CERTIFICATE-----\n"
  "MIIDQTCCAimgAwIBAgITBmyfz5m/jAo54vB4ikPmljZbyjANBgkqhkiG9w0BAQsF\n"
```

Figure 53: Debug Configurations

7. Change the configuration as shown in Figure 54 and then select the **Apply** and **Debug** buttons.



Name: RA6M4_DA16x00_ATcmd.elf

Debug hardware: J-Link ARM   Target Device: R7FA6M4AF

● Autostart local GDB server
○ Connect to remote GDB server

Host name or IP address: localhost
GDB port number: 61234
Connection timeout (s): 30

GDB Command: arm-none-eabi-gdb
☐ Step Mode

Figure 54: Set Debug Configurations

8. The image for host MCU is downloaded to the device.

9. The following shows the console output of the DA16200/DA16600 after a factory mode reset.

```
System Mode : Soft-AP (1)
>>> DHCP Server Started
>>> Start DA16X Supplicant ...
>>> DA16x Supp Ver2.7 - 2022_03
>>> Add SoftAP Inteface (softap1) ...
>>> MAC address (softap1) : d4:3d:39:10:d5:07
>>> softap1 interface add OK
>>> AP Operating Channel: 1(2412)
>>> Network Interface (wlan1) : UP
BSS Isolate Disabled
Soft-AP is Ready (d4:3d:39:10:d5:07)
>>> UART1 : Clock=80000000, BaudRate=115200
>>> UART1 : DMA Enabled ...
[reg_user_atcmd_help_cb] Start AZURE AT Command
[UART ready notification]
[http_server_task] HTTP-Server Start!!
===========================================
[ AZURE-IOT AT COMMAND ]
[ azure_twin_dpm_auto_start]
AZURE_IOT on Station Mode for "DA16200" \\For DA16600, AZURE_IOT on Station Mode for "DA16600"
============================================
[azure_twin_dpm_auto_start] mcu_wakeup_port=-1, mcu_wakeup_pin=0x0
default set to mcu_wakeup_port=0, mcu_wakeup_pin=0x800
Root CA: X
Not ready Cert for Azure service ...
invalid AZURE feature...can't start AZURE thread...check again
.. UART ready
```

10. The following shows the console output of the DA16200/DA16600 when setting the Azure IoT configuration through AT commands from an MCU.

```
=========================================================
argc num = 2
argv[0]: AT+AZU
argv[1]: SET APP_BOARD_FEATURE EVK
=========================================================
APP SET EVK
=========================================================
argc num = 2
argv[0]: AT+AZU
argv[1]: SET APP_THINGNAME da16200_atcmd_thing
=========================================================
APP SET da16200_atcmd_thing
=========================================================
argc num = 2
argv[0]: AT+AZU
argv[1]: SET APP_DEV_PRIMARY_KEY dWKJjGHX9TiCOPheiKyRjjcmmUtBBZsuASBB+K6OoGs=
=========================================================
APP SET dWKJjGHX9TiCOPheiKyRjjcmmUtBBZsuASBB+K6OoGs=
=========================================================
argc num = 2
argv[0]: AT+AZU
argv[1]: SET APP_HOSTNAME azure-atcmd-iothub.azure-devices.net
=========================================================
APP SET azure-atcmd-iothub.azure-devices.net
=========================================================
argc num = 2
argv[0]: AT+AZU
argv[1]: SET APP_IOTHUB_CONN_STRING HostName=azure-atcmd-iothub.azure-devices.net;SharedAccess
KeyName=iothubowner;SharedAccessKey=Xq970WfOFvEmVU9Ixcm96zPGTKh1ztcoxF64QBmyvU8=
=========================================================
APP SET HostName=azure-atcmd-iothub.azure-devices.net;SharedAccessKeyName=iothubowner;SharedAc
cessKey=Xq970WfOFvEmVU9Ixcm96zPGTKh1ztcoxF64QBmyvU8=
=========================================================
argc num = 2
argv[0]: AT+AZU
argv[1]: CFG 0 app_door 1 2
=========================================================
=========================================================
Att[0] number   : 0
Att[0] name      : app_door
Att[0] data type: 1
Att[0] MQTT type: 2
=========================================================
```

11. The following shows the console output of the DA16200/DA16600 after the Soft AP has been configured and it is waiting to be provisioned by the mobile application.

```
Soft-AP is Ready (d4:3d:39:11:5e:73)
[http_server_task] HTTP-Server Start!!
==========================================
[ AZURE-IOT AT COMMAND ]
[ azure_twin_dpm_auto_start]
AZURE_IOT on Station Mode for "da16200_atcmd_thing"
===========================================
[azure_twin_dpm_auto_start] mcu_wakeup_port=0, mcu_wakeup_pin=0x800
Root CA: O
subscribe index=0, name=app_door
AZURE_IOT AP Mode  da16200_atcmd_thing
+ATPROV=STATUS 1
======================================================
[Start Provisioning with TCP/TLS] .. Soft AP Mode
======================================================
[app_provision_TCP_server_thread] Create ...
[app_provision_switch_client_thread] Create...(status=0) [21]
[app_provision_TLS_server_thread] Create TLS...
>>> Start Provisioning Server (TLS) ...
Wait Accept (TLS)...
> HTTPS-Server Start !!!
> Wi-Fi Scan request success.
(0) Xiaomi_AX9000 / 3 / -26 / 2412
(1) ZIO-FREEZIO / 3 / -26 / 2437
(2) KAON_GAPK_7600 / 3 / -28 / 2452
(3) HiWiFi_5B2310 / 3 / -31 / 2417
(4) N_TPLINK_EAP620HD_WPA2-ENT / 3 / -32 / 2462
(5) N_TPLINK_EAP620HD_WPA23-PSK / 3 / -32 / 2462
(27) SAT_Synology_MR2200AC_OWE / 3 / -54 / 2437
(28) JMC_DIR-615_WPA1_TKIP / 2 / -55 / 2462
(29) N_A3004_ENT_Synology / 3 / -56 / 2422
[app_provision_TCP_server_thread] socket().. status=1
 Wait Accept...
```

4.3 Mobile App Demo

Install the mobile application by searching for **DA16200** or **DA16600** on the Google Pay Store or Apple App Store on the mobile devices.

4.3.1 Open Door

A message flow of **opening door** is shown in Figure 55.

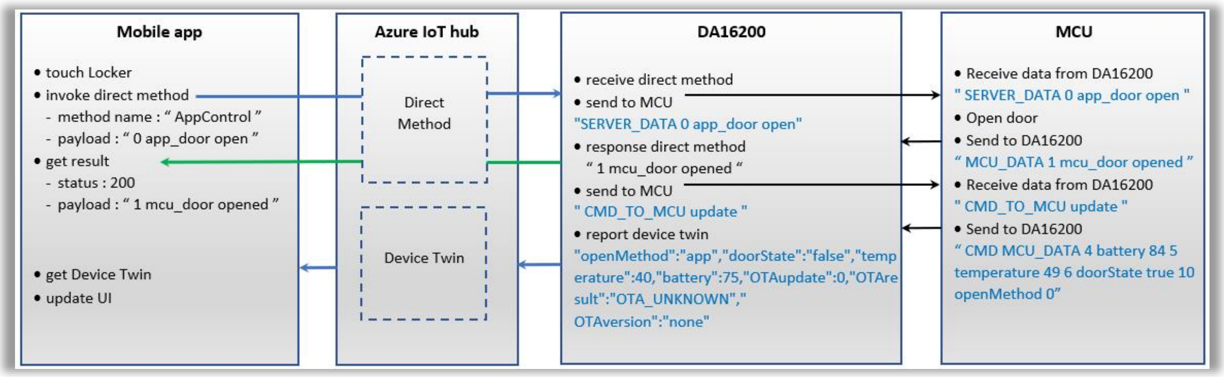
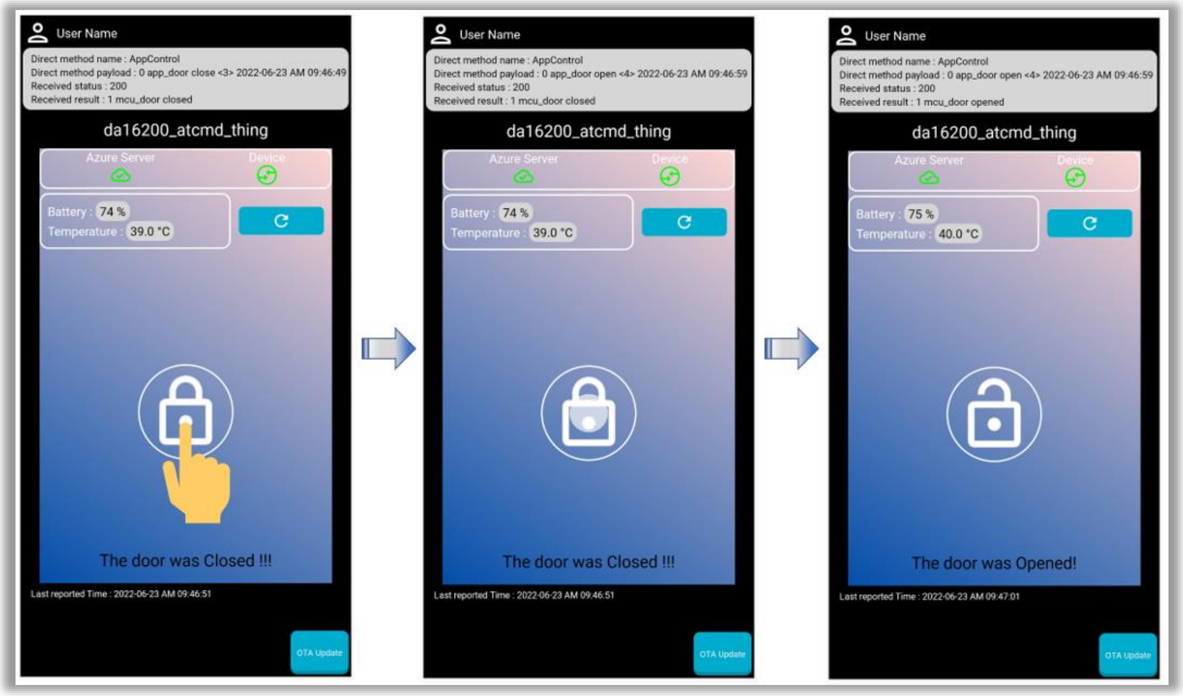

Mobile app
• touch Locker
• invoke direct method
- method name : " AppControl "
- payload : " 0 app_door open "
• get result
- status : 200
- payload : " 1 mcu_door opened "
• get Device Twin
• update UI
Azure IoT hub
Direct Method
Device Twin
DA16200
• receive direct method
• send to MCU
"SERVER_DATA 0 app_door open"
• response direct method
" 1 mcu_door opened "
• send to MCU
" CMD_TO_MCU update "
• report device twin
"openMethod":"app","doorState":"false","temperature":40,"battery":75,"OTAupdate":0,"OTAresult":"OTA_UNKNOWN","OTAversion":"none"
MCU
• Receive data from DA16200
" SERVER_DATA 0 app_door open "
• Open door
• Send to DA16200
" MCU_DATA 1 mcu_door opened "
• Receive data from DA16200
" CMD_TO_MCU update "
• Send to DA16200
" CMD MCU_DATA 4 battery 84 5 temperature 49 6 doorState true 10 openMethod 0"

Figure 55: Message Flow of Opening Door

The operation for opening door on android app along with the Azure IoT console output displaying the device twin state is shown in Figure 56.



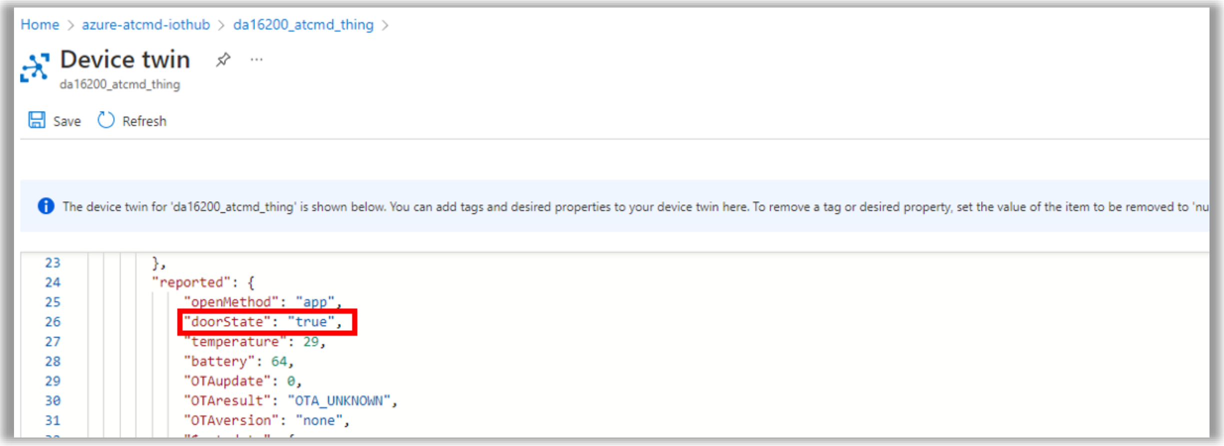
User Name
Direct method name : AppControl
Direct method payload : 0 app_door close <3> 2022-06-23 AM 09:46:49
Received status : 200
Received result : 1 mcu_door closed
da16200_atcmd_thing
Battery : 74 %
Temperature : 39.0 °C
The door was Closed !!!
Last reported Time : 2022-06-23 AM 09:46:51

User Name
Direct method name : AppControl
Direct method payload : 0 app_door open <4> 2022-06-23 AM 09:46:59
Received status : 200
Received result : 1 mcu_door closed
da16200_atcmd_thing
Battery : 74 %
Temperature : 39.0 °C
The door was Closed !!!
Last reported Time : 2022-06-23 AM 09:46:51

User Name
Direct method name : AppControl
Direct method payload : 0 app_door open <4> 2022-06-23 AM 09:46:59
Received status : 200
Received result : 1 mcu_door opened
da16200_atcmd_thing
Battery : 75 %
Temperature : 40.0 °C
The door was Opened!
Last reported Time : 2022-06-23 AM 09:47:01

Figure 56: Operation of Opening Door in App

Figure 57 shows the state of the device twin on the Azure IoT Hub once the operation of opening door has been completed.



The device twin for 'da16200_atcmd_thing' is shown below. You can add tags and desired properties to your device twin here. To remove a tag or desired property, set the value of the item to be removed to 'nu

```
            },
            "reported": {
                "openMethod": "app",
                "doorState": "true",
                "temperature": 29,
                "battery": 64,
                "OTAupdate": 0,
                "OTAresult": "OTA_UNKNOWN",
                "OTAversion": "none",
```

Figure 57: Device Twin When Door is Opened

The console log of the DA16200/DA16600 after the door is opened as below.

```
Info: [deviceMethodCallback:565] : method_name: AppControl, Command payload : 17, "0 app_door open"
dynamic subscription command ("app_door")
send "SERVER_DATA 0 app_door open" to MCU
=======================================================
argc num = 2
argv[0]: AT+AZU
argv[1]: CMD MCU_DATA 1 mcu_door opened
=======================================================
send "CMD_TO_MCU update" to MCU
=======================================================
argc num = 2
argv[0]: AT+AZU
argv[1]: CMD MCU_DATA 2 battery 75 3 temperature 40 4 doorState true 7 openMethod app
=======================================================
board feature : EVK
topicCount : 4
Count : 0, cmdNum = 2
mqtttype = 1
index(=3) matched
data type(shadow) = 0
call update sensor(need to be set variable): battery = 75
Count : 1, cmdNum = 3
mqtttype = 1
index(=2) matched
data type(shadow) = 2
call update sensor(need to be set variable): temperature = 35.000000
Count : 2, cmdNum = 4
mqtttype = 1
index(=1) matched
data type(shadow) = 1
call update sensor(need to be set variable): doorState = true
Count : 3, cmdNum = 7
mqtttype = 1
index(=0) matched
data type(shadow) = 1
call update sensor(need to be set variable): openMethod = app
release response
=========================================================================================
Info: [deviceMethodCallback:644] : Command response : "1 mcu_door opened"
registerShadow String data (openMethod) (app),
registerShadow String data (doorState) (true),
Info: [app_send_twins:305] : Report =>(method callback) {"openMethod":"app","doorState":"true",
"temperature":40,"battery":75,"OTAupdate":0,"OTAresult":"OTA_UNKNOWN","OTAversion":"none","MCUOTAversion":"no
ne" }
```

4.3.2 Close Door

Message flows of **closing door** is shown in Figure 58.

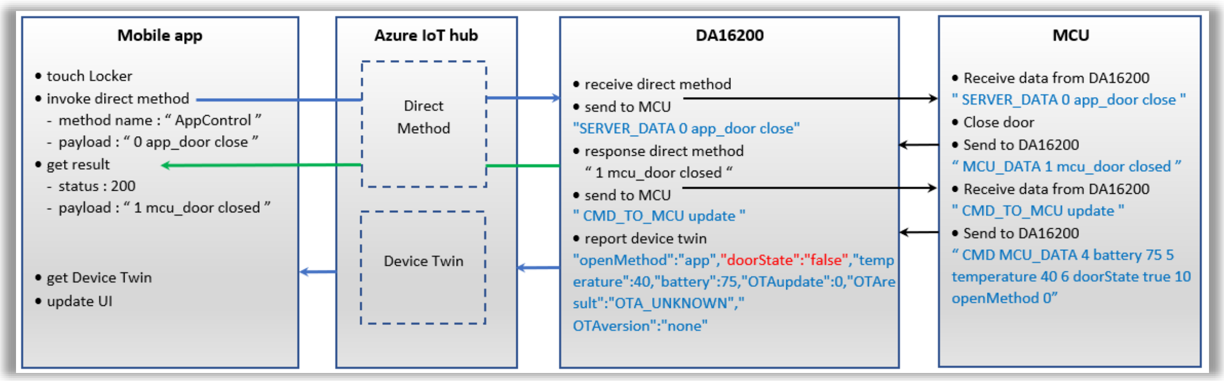
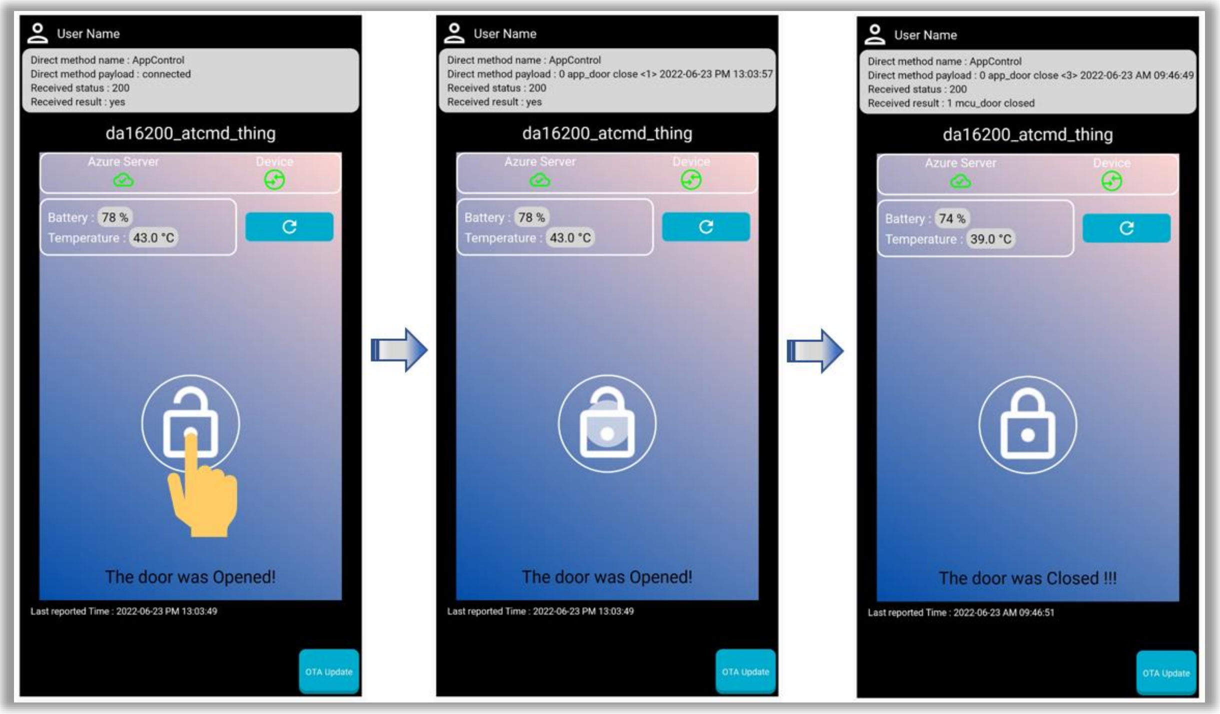

Mobile app
• touch Locker
• invoke direct method
- method name : " AppControl "
- payload : " 0 app_door close "
• get result
- status : 200
- payload : " 1 mcu_door closed "
• get Device Twin
• update UI
Azure IoT hub
Direct Method
Device Twin
DA16200
• receive direct method
• send to MCU
"SERVER_DATA 0 app_door close"
• response direct method
" 1 mcu_door closed "
• send to MCU
" CMD_TO_MCU update "
• report device twin
"openMethod":"app","doorState":"false","temperature":40,"battery":75,"OTAupdate":0,"OTAresult":"OTA_UNKNOWN"," OTAversion":"none"
MCU
• Receive data from DA16200
" SERVER_DATA 0 app_door close "
• Close door
• Send to DA16200
" MCU_DATA 1 mcu_door closed "
• Receive data from DA16200
" CMD_TO_MCU update "
• Send to DA16200
" CMD MCU_DATA 4 battery 75 5 temperature 40 6 doorState true 10 openMethod 0"

Figure 58: Message Flows of Closing Door

The operation for closing door on android app along with the Azure IoT console output displaying the device twin state is shown in Figure 59.

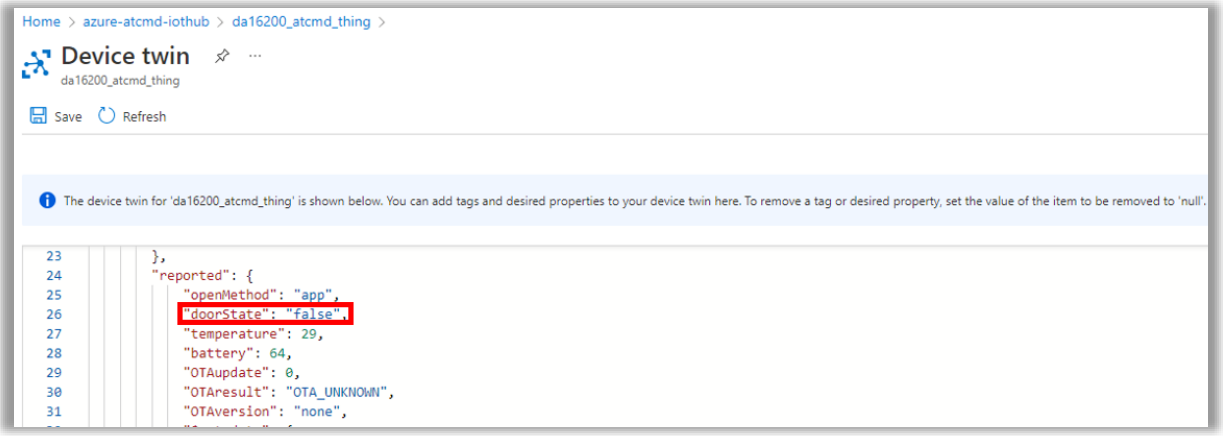

User Name
Direct method name : AppControl
Direct method payload : connected
Received status : 200
Received result : yes
da16200_atcmd_thing
Battery : 78 %
Temperature : 43.0 °C
The door was Opened!
Last reported Time : 2022-06-23 PM 13:03:49

User Name
Direct method name : AppControl
Direct method payload : 0 app_door close <1> 2022-06-23 PM 13:03:57
Received status : 200
Received result : yes
da16200_atcmd_thing
Battery : 78 %
Temperature : 43.0 °C
The door was Opened!
Last reported Time : 2022-06-23 PM 13:03:49

User Name
Direct method name : AppControl
Direct method payload : 0 app_door close <3> 2022-06-23 AM 09:46:49
Received status : 200
Received result : 1 mcu_door closed
da16200_atcmd_thing
Battery : 74 %
Temperature : 39.0 °C
The door was Closed !!!
Last reported Time : 2022-06-23 AM 09:46:51

Figure 59: Operation of Closing Door in App

Figure 60 shows the device twin of Azure IoT Hub when the opening door operation is completed.


Home > azure-atcmd-iothub > da16200_atcmd_thing >

Device twin

da16200_atcmd_thing

Save Refresh

The device twin for 'da16200_atcmd_thing' is shown below. You can add tags and desired properties to your device twin here. To remove a tag or desired property, set the value of the item to be removed to 'null'.

```
        },
        "reported": {
            "openMethod": "app",
            "doorState": "false",
            "temperature": 29,
            "battery": 64,
            "OTAupdate": 0,
            "OTAresult": "OTA_UNKNOWN",
            "OTAversion": "none",
```

Figure 60: Device Twin When Door is Closed

The console output of the DA16200/DA16600 after the door is closed is shown below:

```
Info: [deviceMethodCallback:565] : method_name: AppControl, Command payload : 18, "0 app_door close"
dynamic subscription command ("app_door")
send "SERVER_DATA 0 app_door close" to MCU
==================================================
argc num = 2
argv[0]: AT+AZU
argv[1]: CMD MCU_DATA 1 mcu_door closed
==================================================
send "CMD_TO_MCU update" to MCU
==================================================
argc num = 2
argv[0]: AT+AZU
argv[1]: CMD MCU_DATA 4 battery 79 5 temperature 44 6 doorState false 10 openMethod app
==================================================
board feature : EVK
topicCount : 4
Count : 0, cmdNum = 2
mqtttype = 1
index(=3) matched
data type(shadow) = 0
call update sensor(need to be set variable): battery = 79
Count : 1, cmdNum = 3
mqtttype = 1
index(=2) matched
data type(shadow) = 2
call update sensor(need to be set variable): temperature = 44.000000
Count : 2, cmdNum = 4
mqtttype = 1
index(=1) matched
data type(shadow) = 1
call update sensor(need to be set variable): doorState = false
Count : 3, cmdNum = 7
mqtttype = 1
index(=0) matched
data type(shadow) = 1
call update sensor(need to be set variable): openMethod = app
release response
==============================================================================
Info: [deviceMethodCallback:644] : Command response : "1 mcu_door closed"
registerShadow String data (openMethod) (app),
registerShadow String data (doorState) (false),
Info: [app_send_twins:305] : Report =>(method callback) {"openMethod":"app","doorState":"false",
"temperature":44,"battery":79,"OTAupdate":0,"OTAresult":"OTA_UNKNOWN","OTAversion":"none","MCUOTAversion":"no
ne"}
```

5 OTA Update

Over the Air (OTA) is the process of updating the DA16200/DA16600 firmware image through Wi-Fi using an Azure services storage account.

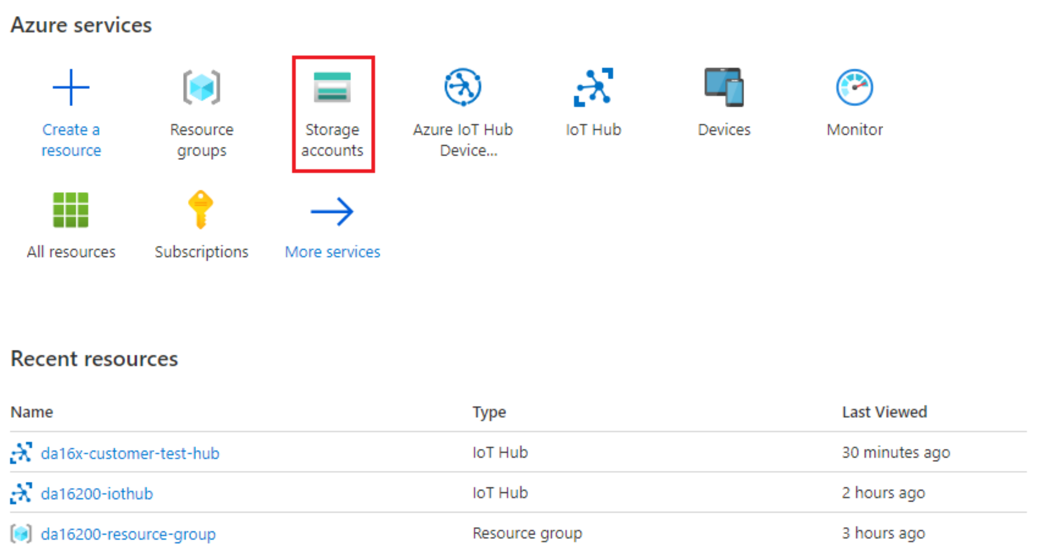
To set up OTA update:

1. Create a storage account.
2. Create a container.
3. Upload a DA16200/DA16600 image file to the container.
4. Add the OTA URL to the device twin.
5. Execute the OTA update.

5.1 Create Storage Account

For OTA update, an Azure services storage account is required.

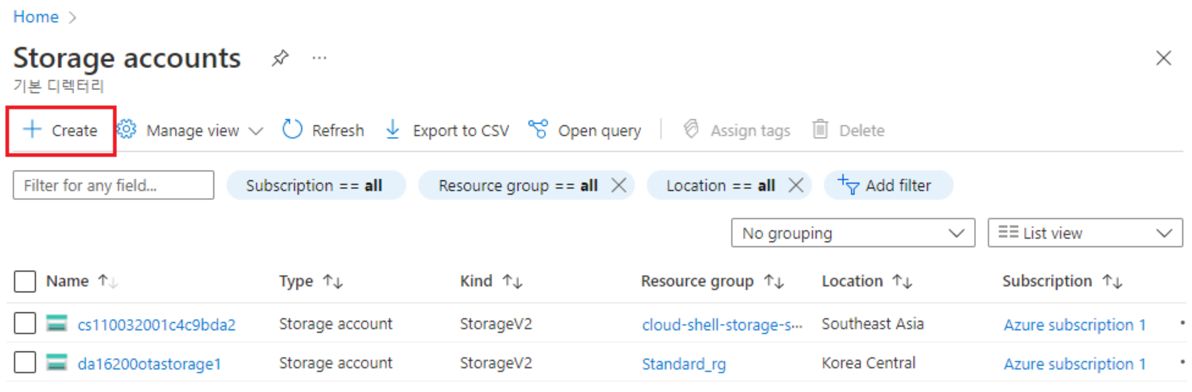
1. Click **Storage Accounts**.



Name	Type	Last Viewed
da16x-customer-test-hub	IoT Hub	30 minutes ago
da16200-iothub	IoT Hub	2 hours ago
da16200-resource-group	Resource group	3 hours ago

Figure 61: Storage Accounts

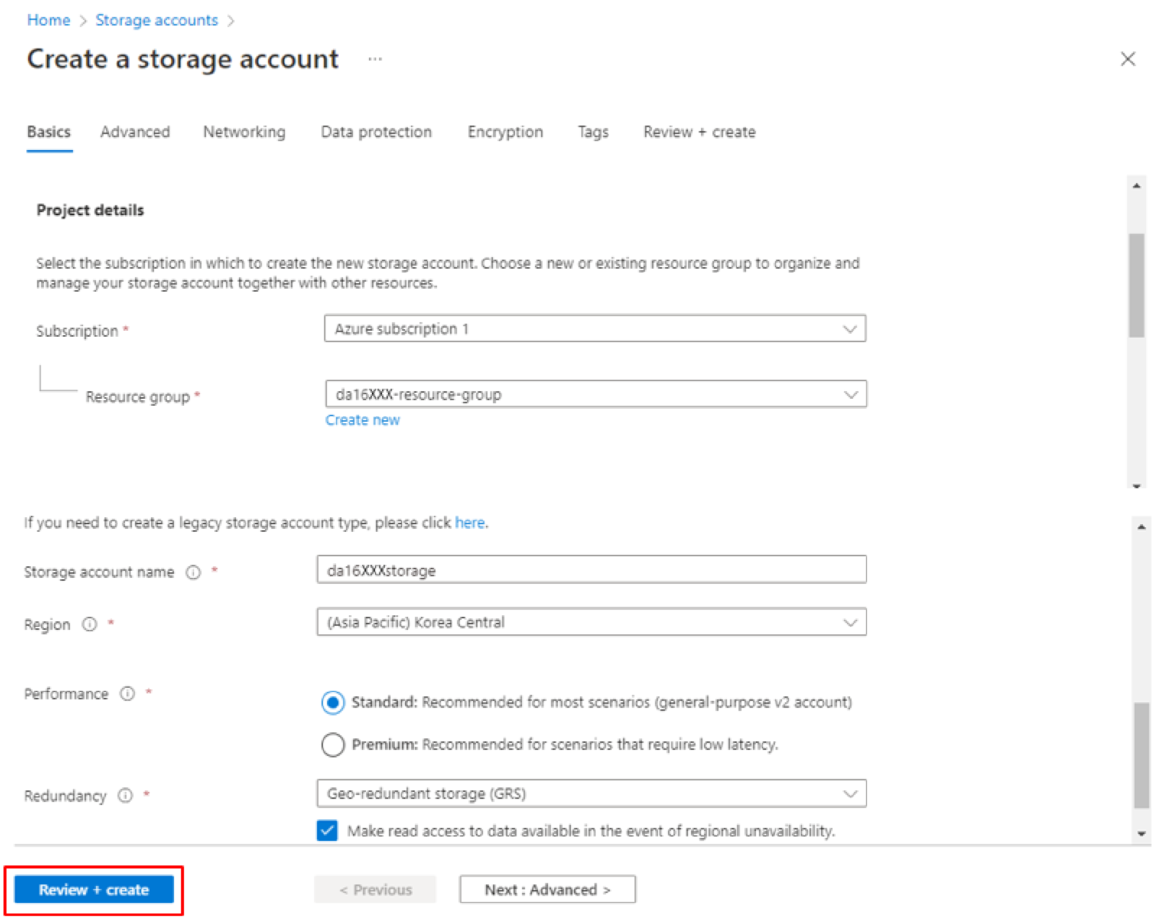
2. Click **Create**.



Name	Type	Kind	Resource group	Location	Subscription
cs110032001c4c9bda2	Storage account	StorageV2	cloud-shell-storage-s...	Southeast Asia	Azure subscription 1
da16200otastorage1	Storage account	StorageV2	Standard_rg	Korea Central	Azure subscription 1

Figure 62: Create Storage Account

3. Configure the **Resource group**, **Storage account name**, and **Region**, and then click **Review + create**.


Home > Storage accounts >

Create a storage account

Basics | Advanced | Networking | Data protection | Encryption | Tags | Review + create

Project details

Select the subscription in which to create the new storage account. Choose a new or existing resource group to organize and manage your storage account together with other resources.

Subscription *: Azure subscription 1

Resource group *: da16XXX-resource-group

Create new

If you need to create a legacy storage account type, please click here.

Storage account name *: da16XXXstorage

Region *: (Asia Pacific) Korea Central

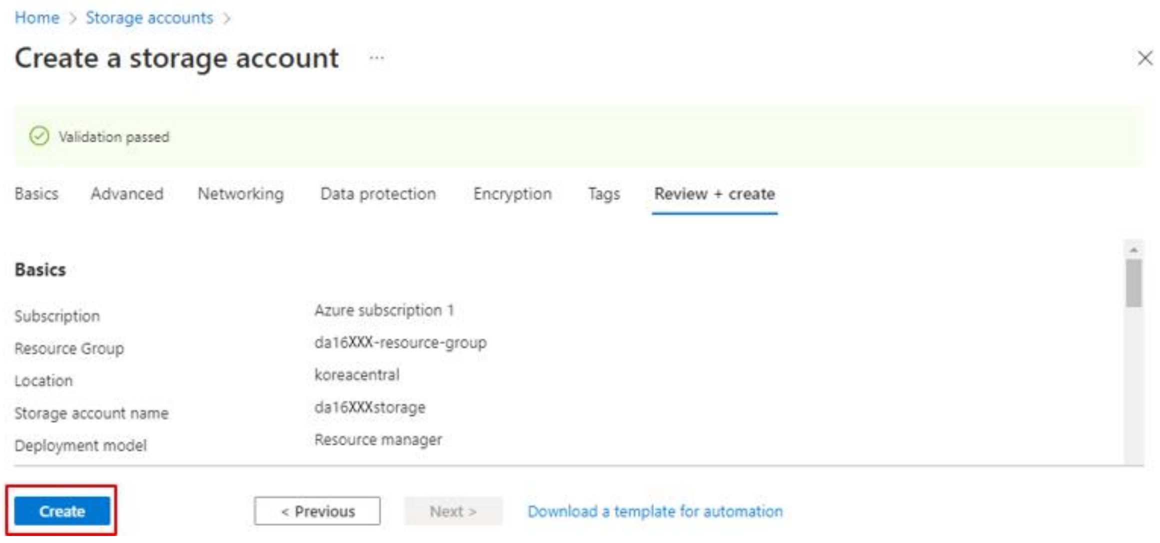
Performance *: [✓ Standard: Recommended for most scenarios (general-purpose v2 account)] Premium: Recommended for scenarios that require low latency.

Redundancy *: Geo-redundant storage (GRS)

✓ Make read access to data available in the event of regional unavailability.

Figure 63: Configure Storage Account

4. Once the storage account validation passes, click **Create**.


Home > Storage accounts >

Create a storage account

Validation passed

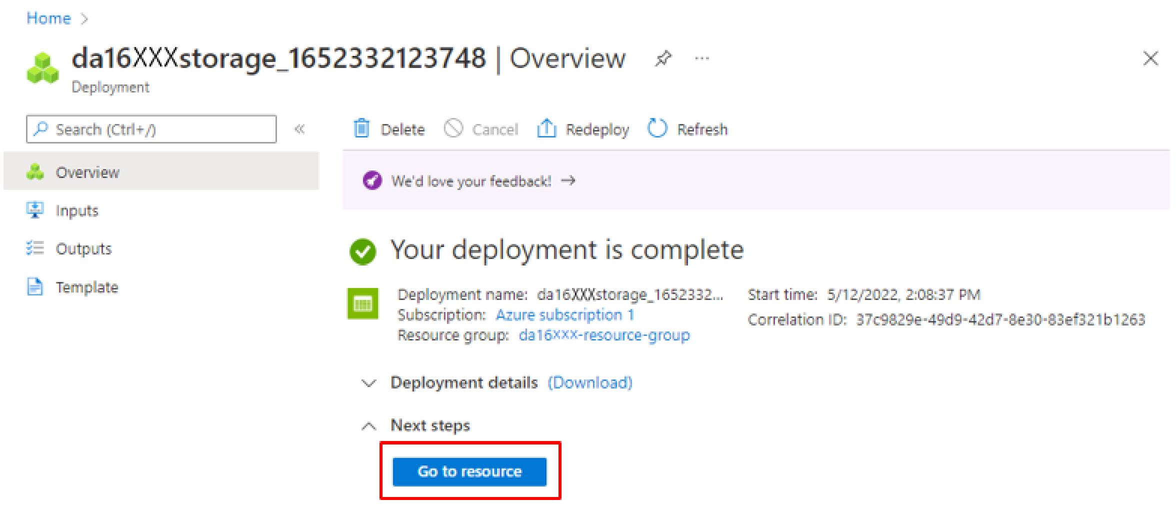
Basics | Advanced | Networking | Data protection | Encryption | Tags | Review + create

Basics

Subscription	Azure subscription 1
Resource Group	da16XXX-resource-group
Location	koreacentral
Storage account name	da16XXXstorage
Deployment model	Resource manager

Figure 64: Storage Account Validation Done

5. Once the storage account deployment is complete, click **Go to resource**.


Home >

da16XXXstorage_1652332123748 | Overview

Deployment

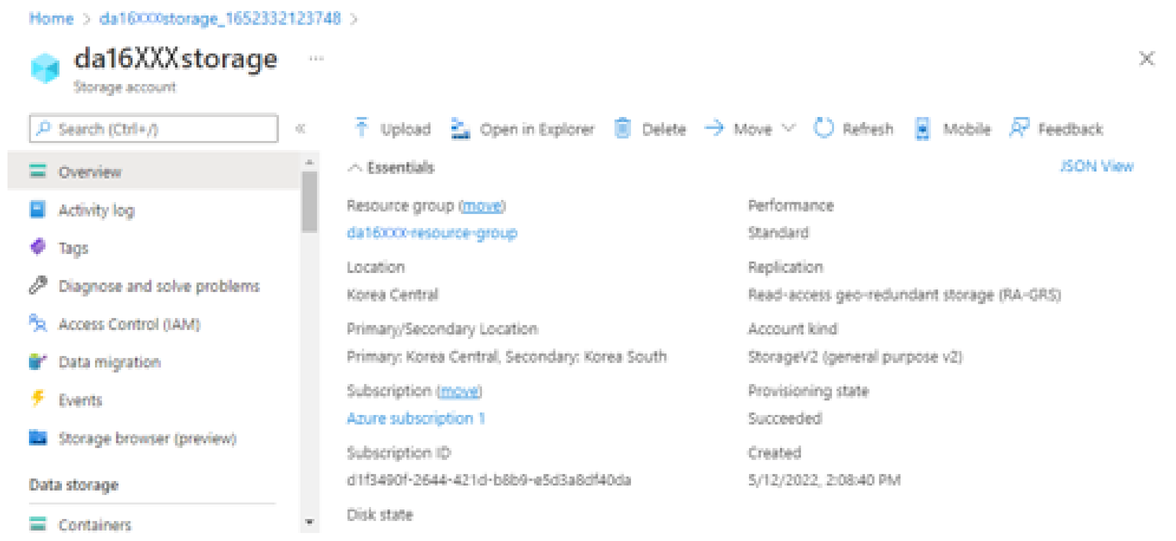
Your deployment is complete

Deployment name: da16XXXstorage_1652332...
Subscription: Azure subscription 1
Resource group: da16xxx-resource-group

Start time: 5/12/2022, 2:08:37 PM
Correlation ID: 37c9829e-49d9-42d7-8e30-83ef321b1263

Figure 65: Storage Account Deployment Done

6. The created storage account is as shown in Figure 66.


Home > da16XXXstorage_1652332123748 >

da16XXXstorage

Storage account

Essentials

Resource group (move)
da16XXX-resource-group

Location
Korea Central

Primary/Secondary Location
Primary: Korea Central, Secondary: Korea South

Subscription (move)
Azure subscription 1

Subscription ID
d1f3490f-2644-421d-b8b9-e5d3a8df40da

Disk state

Performance
Standard

Replication
Read-access geo-redundant storage (RA-GRS)

Account kind
StorageV2 (general purpose v2)

Provisioning state
Succeeded

Created
5/12/2022, 2:08:40 PM

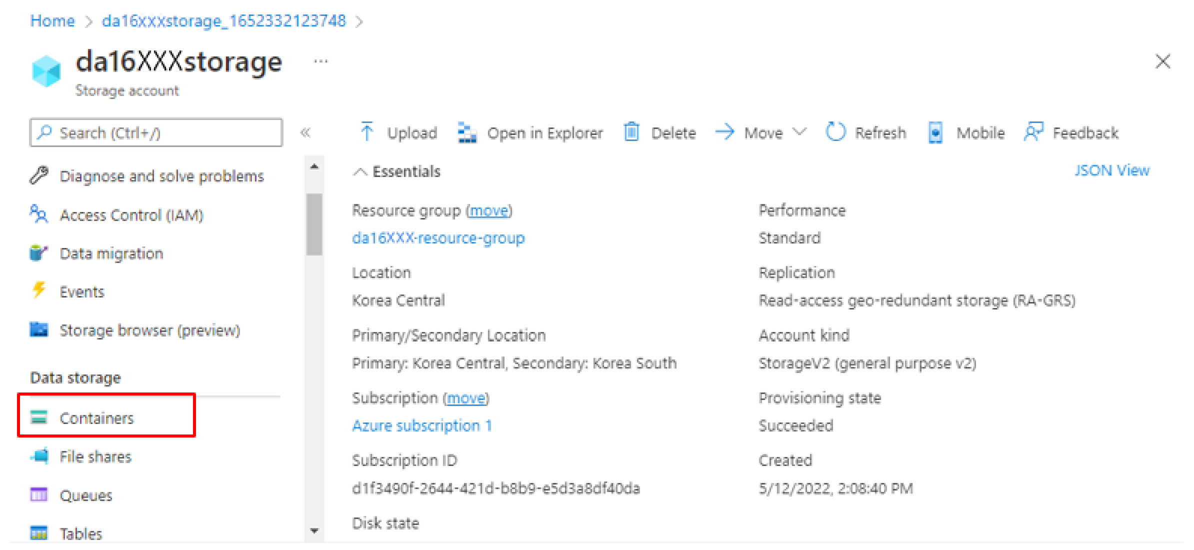
Figure 66: Created Storage Account

5.2 Create Container

A container must be created in the Azure services storage account to store the DA16200/DA16600 image file for OTA update.

To create a container:

1. In the created storage account, click **Containers**.


Home > da16XXXstorage_1652332123748 >

da16XXXstorage ···
Storage account

Search (Ctrl+/) «

Upload | Open in Explorer | Delete | Move | Refresh | Mobile | Feedback

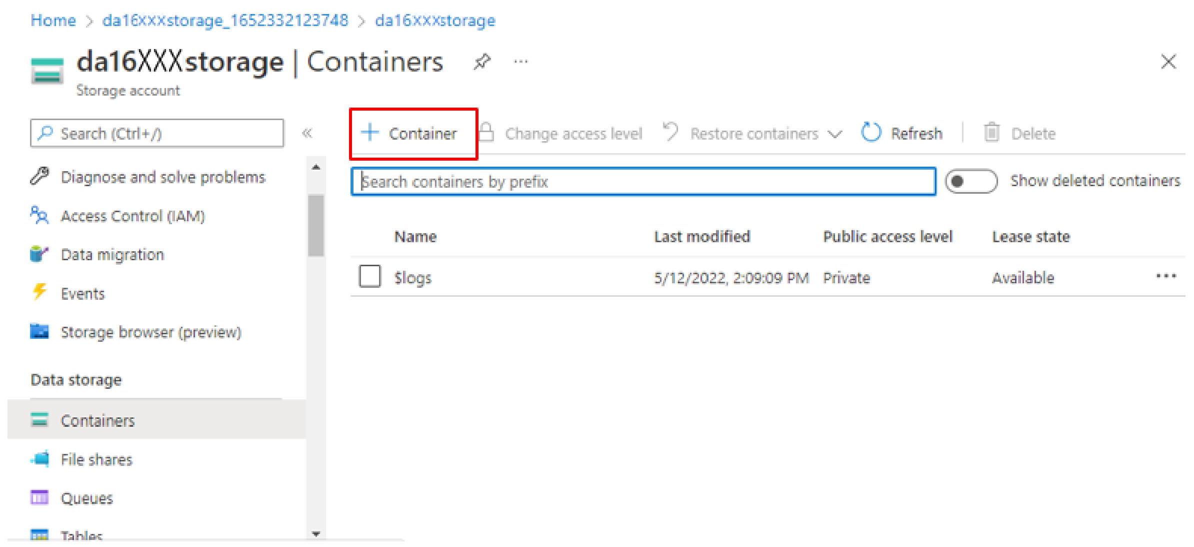
Diagnose and solve problems
Access Control (IAM)
Data migration
Events
Storage browser (preview)
Data storage
Containers
File shares
Queues
Tables

Essentials | JSON View

Resource group (move)	Performance
da16XXX-resource-group	Standard
Location	Replication
Korea Central	Read-access geo-redundant storage (RA-GRS)
Primary/Secondary Location	Account kind
Primary: Korea Central, Secondary: Korea South	StorageV2 (general purpose v2)
Subscription (move)	Provisioning state
Azure subscription 1	Succeeded
Subscription ID	Created
d1f3490f-2644-421d-b8b9-e5d3a8df40da	5/12/2022, 2:08:40 PM
Disk state	

Figure 67: Containers in Storage Account

2. Click **Container** to add container.


Home > da16XXXstorage_1652332123748 > da16XXXstorage

da16XXXstorage | Containers ···
Storage account

Search (Ctrl+/) «

+ Container | Change access level | Restore containers | Refresh | Delete

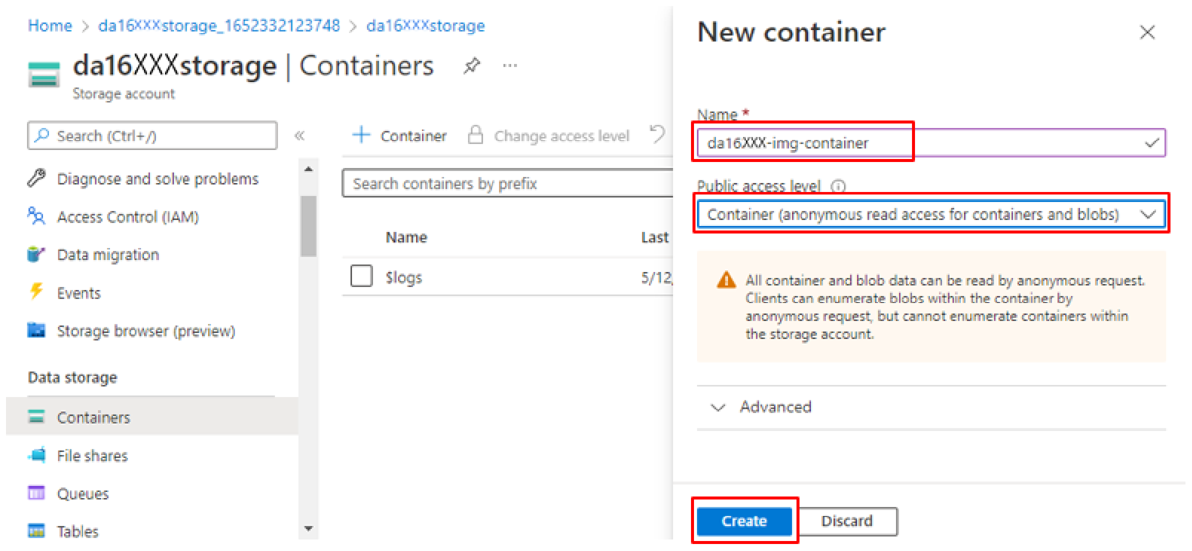
Search containers by prefix | Show deleted containers

Diagnose and solve problems
Access Control (IAM)
Data migration
Events
Storage browser (preview)
Data storage
Containers
File shares
Queues
Tables

Name	Last modified	Public access level	Lease state
$logs	5/12/2022, 2:09:09 PM	Private	Available

Figure 68: Add Container

3. Under the **New container** section, enter a name (for example, *"da16XXX-img-container"*) and select **Container (anonymous read access for containers and blobs)** as the Public access level, and then click **Create**.


New container

Name *: da16XXX-img-container

Public access level: Container (anonymous read access for containers and blobs)

All container and blob data can be read by anonymous request. Clients can enumerate blobs within the container by anonymous request, but cannot enumerate containers within the storage account.

Figure 69: Container Details

4. The created container appears in the containers list.

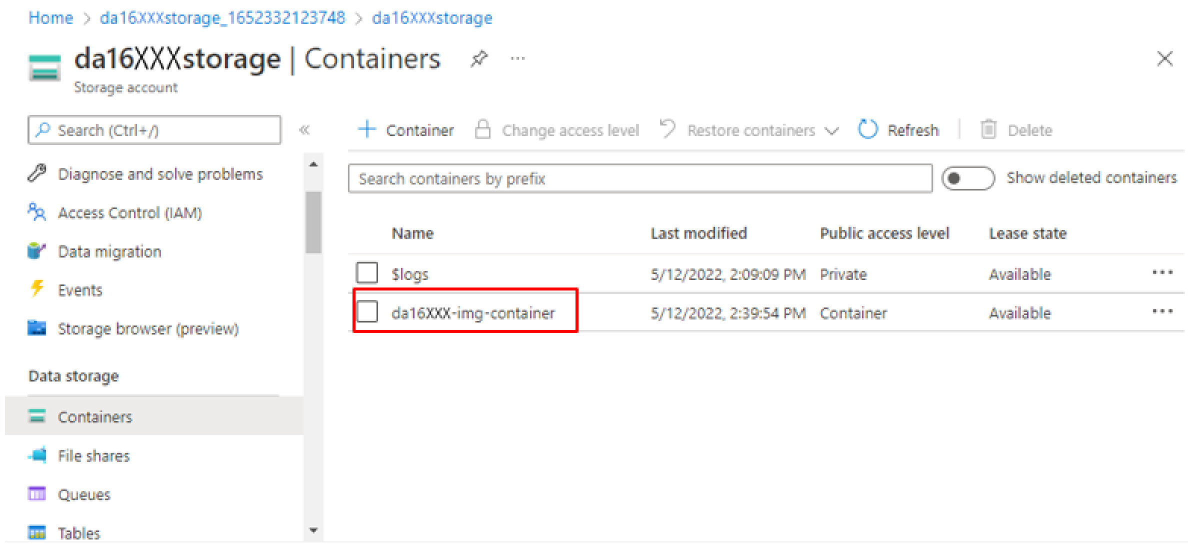
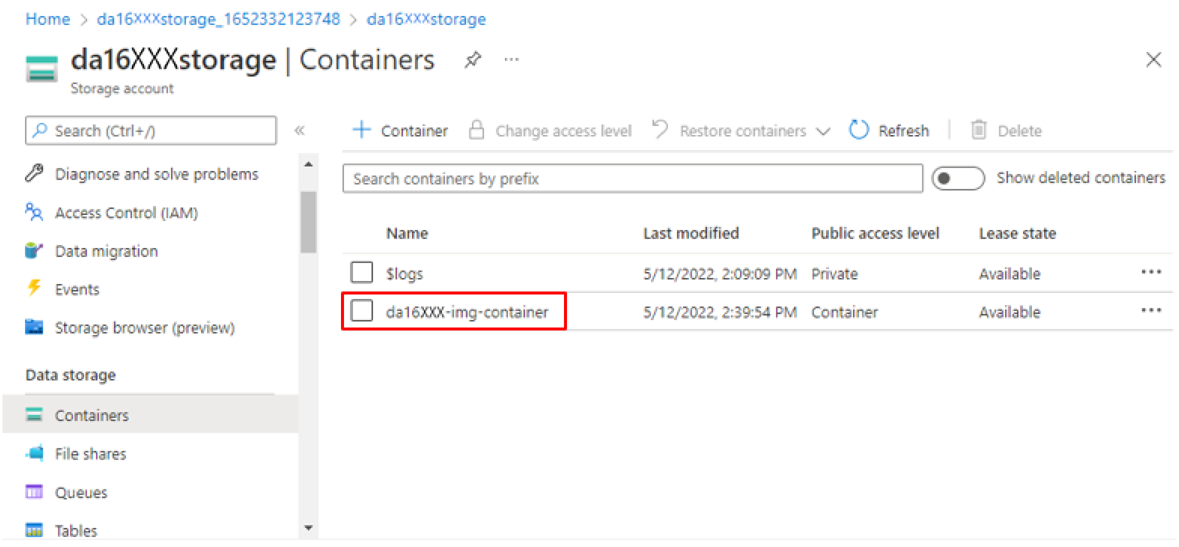

Name	Last modified	Public access level	Lease state
$logs	5/12/2022, 2:09:09 PM	Private	Available
da16XXX-img-container	5/12/2022, 2:39:54 PM	Container	Available

Figure 70: Created Container

5.3 Upload Firmware Image File

To upload the DA16200/DA16600 firmware image file to the container:

1. Click the created container in the containers list.


Home > da16XXXstorage_1652332123748 > da16XXXstorage

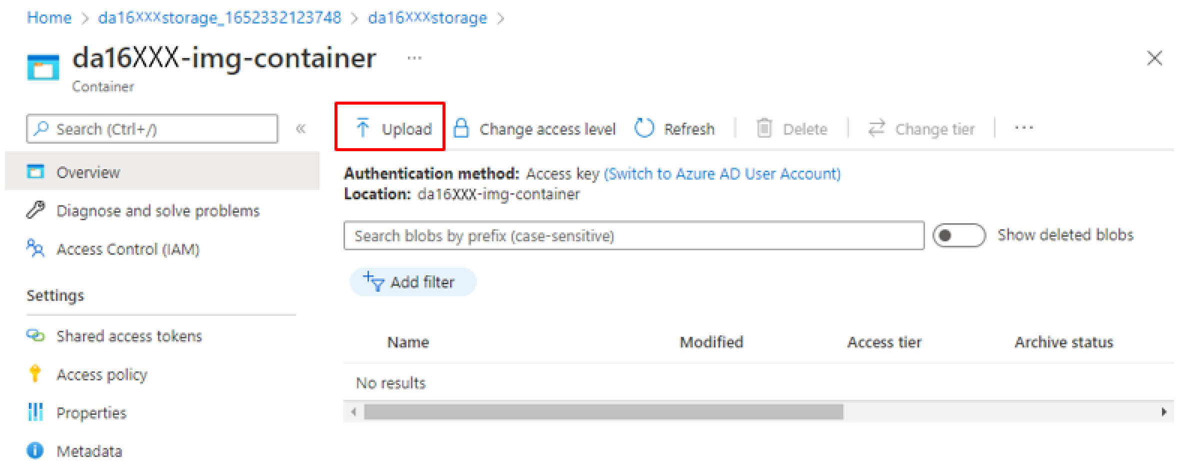
da16XXXstorage | Containers

Storage account

Name	Last modified	Public access level	Lease state
$logs	5/12/2022, 2:09:09 PM	Private	Available
da16XXX-img-container	5/12/2022, 2:39:54 PM	Container	Available

Figure 71: Container for Uploading Images

2. Click **Upload** to upload the DA16200/DA16600 image file.


Home > da16XXXstorage_1652332123748 > da16XXXstorage >

da16XXX-img-container

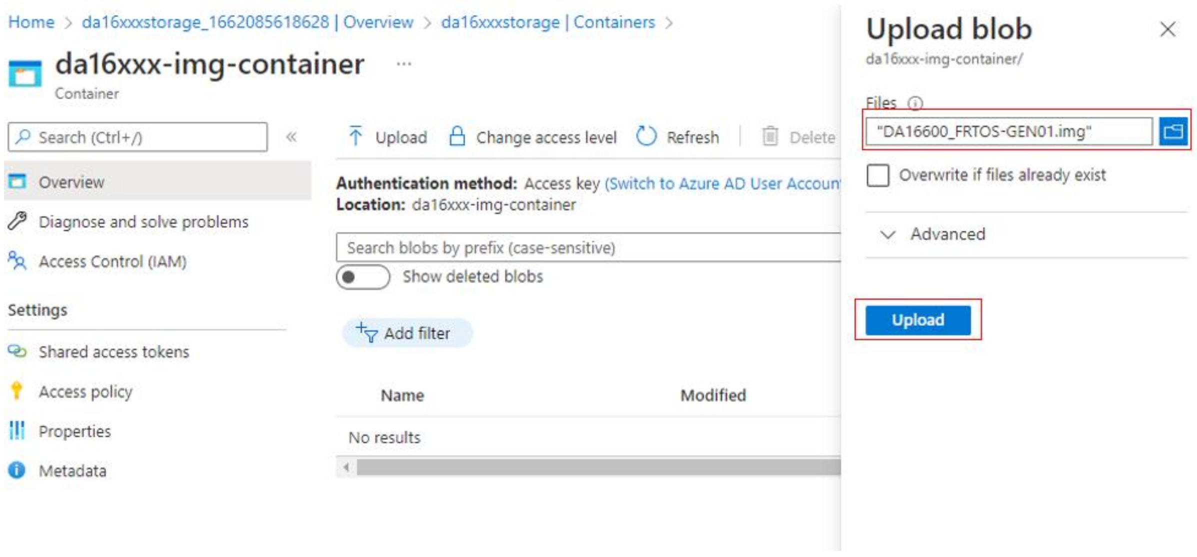
Container

Authentication method: Access key (Switch to Azure AD User Account)
Location: da16XXX-img-container

Name	Modified	Access tier	Archive status
No results			

Figure 72: Upload Image

3. Under the **Upload blob** section, in Files navigate to the DA16200/DA16600 image file saved in local directory:
   - In case of DA16200
     - **DA16200_FRTOS-GEN01.img**: (must be uploaded as this name)
   - In case of DA16600
     - **DA16600_FRTOS-GEN01.img**: (must be uploaded as this name)
     - **DA16600_BLE_OTA.img**: (must be uploaded as this name)
4. Click **Upload** and then uploaded image files are appeared as shown in Figure 74.



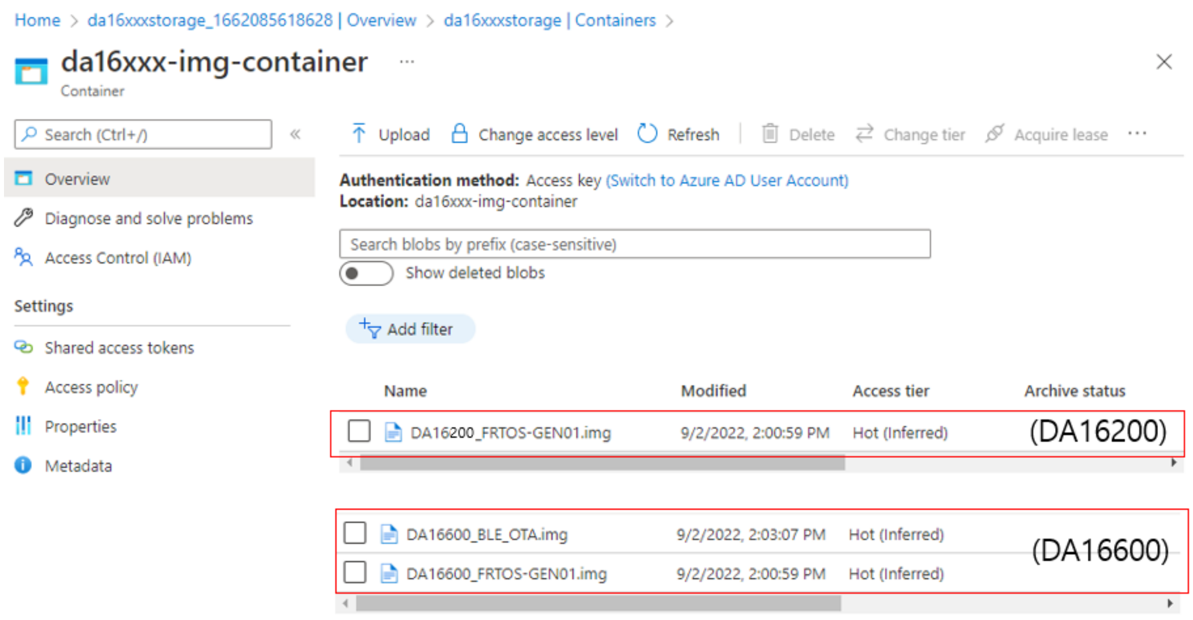
Authentication method: Access key (Switch to Azure AD User Account)
Location: da16xxx-img-container

No results

da16xxx-img-container/

"DA16600_FRTOS-GEN01.img"

Figure 73: Navigate to Image

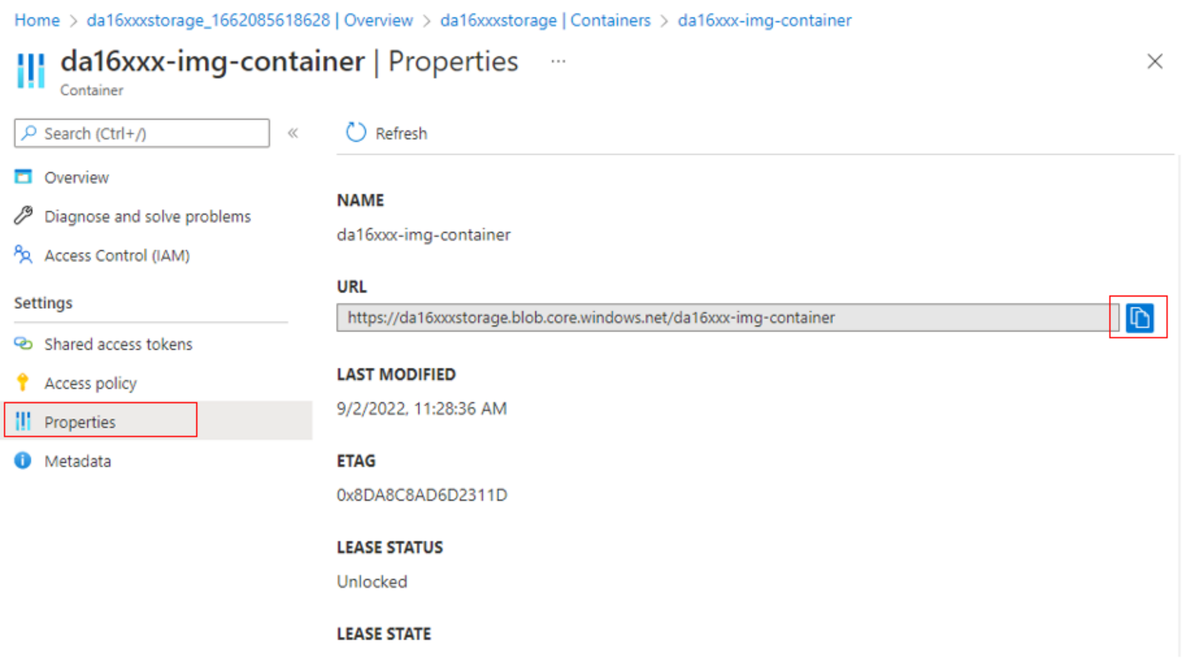


Authentication method: Access key (Switch to Azure AD User Account)
Location: da16xxx-img-container

Name	Modified	Access tier	Archive status
DA16200_FRTOS-GEN01.img	9/2/2022, 2:00:59 PM	Hot (Inferred)	(DA16200)
DA16600_BLE_OTA.img	9/2/2022, 2:03:07 PM	Hot (Inferred)	(DA16600)
DA16600_FRTOS-GEN01.img	9/2/2022, 2:00:59 PM	Hot (Inferred)	

Figure 74: Uploaded Image

5. Choose **Properties**, and then click the Copy icon as shown in Figure 75 after the file is uploaded. Save the **URL** to configure the Azure Device Twin later.



Home > da16xxxstorage_1662085618628 | Overview > da16xxxstorage | Containers > da16xxx-img-container

da16xxx-img-container | Properties
Container

NAME
da16xxx-img-container

URL
https://da16xxxstorage.blob.core.windows.net/da16xxx-img-container

LAST MODIFIED
9/2/2022, 11:28:36 AM

ETAG
0x8DA8C8AD6D2311D

LEASE STATUS
Unlocked

LEASE STATE

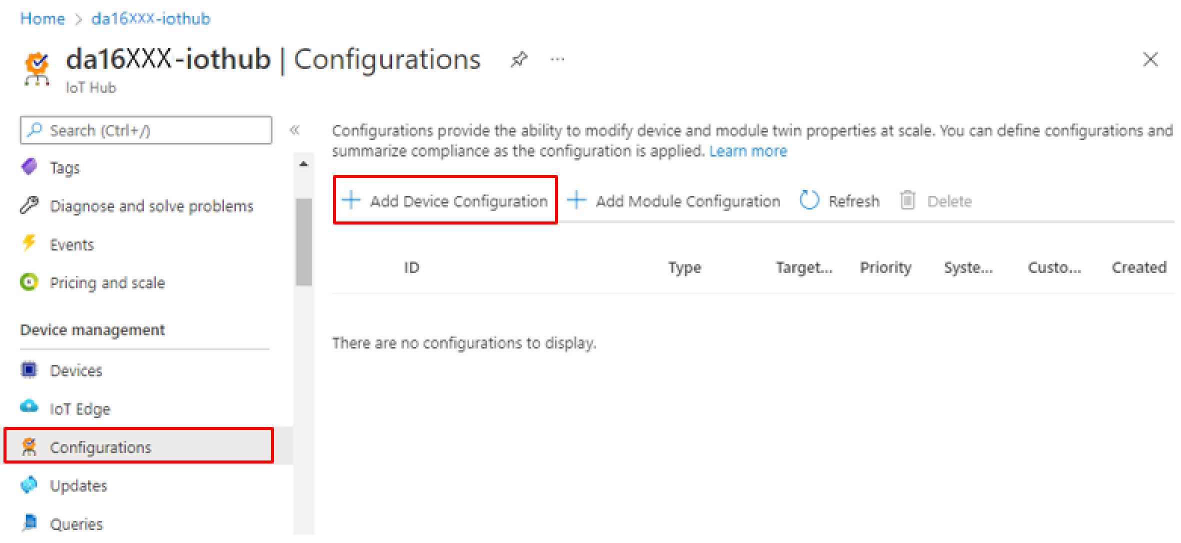
Figure 75: Copy Container URL

5.4 Add OTA URL to Device Twin

For OTA update, a configuration must be added to the IoT Hub that sets the OTA URL in the Device Twin.

To add configuration in Device Twin for the OTA URL:

1. In the IoT hub, for example *da16XXX-iothub*, click **Configurations**, and then select **Add Device Configuration**.



Home > da16XXX-iothub

da16XXX-iothub | Configurations
IoT Hub

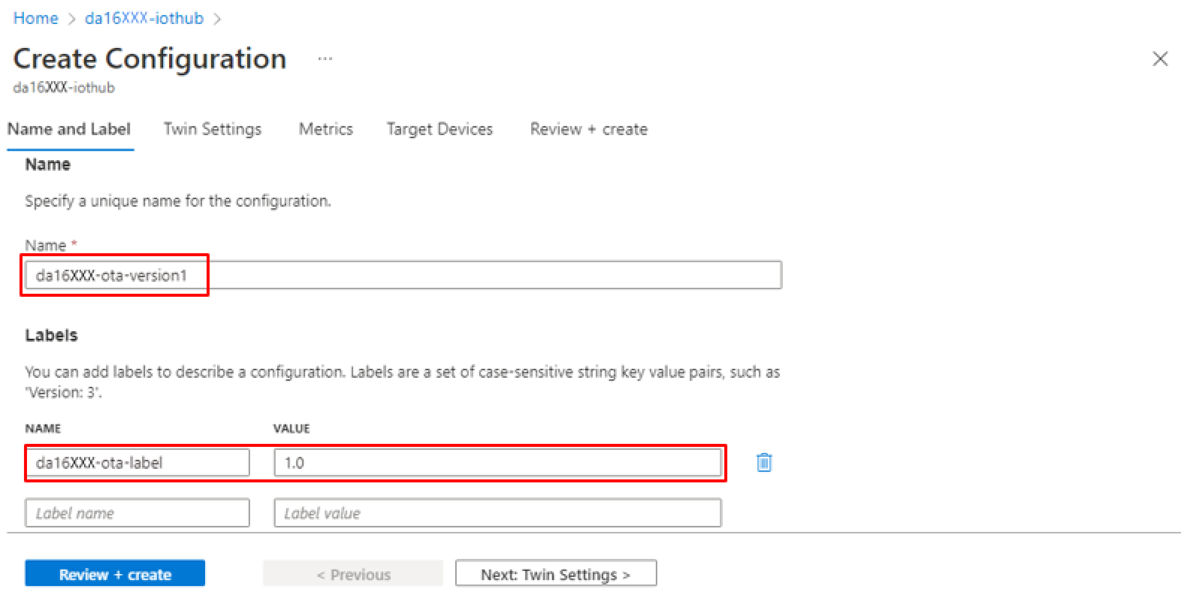
Configurations provide the ability to modify device and module twin properties at scale. You can define configurations and summarize compliance as the configuration is applied. Learn more

ID	Type	Target...	Priority	Syste...	Custo...	Created

There are no configurations to display.

Figure 76: Add Device Configuration

2. Under the **Name and Label** section, follow the steps below:
   a. In **Name**, add a name, for example "*da16XXX-ota-version1*".
   b. Under Label, create a label by adding name and value, for example "*da16XXX-ota-label*", "*1.0*".


Home > da16XXX-iothub >

Create Configuration

da16XXX-iothub

Name

Specify a unique name for the configuration.

Name *: da16XXX-ota-version1

Labels

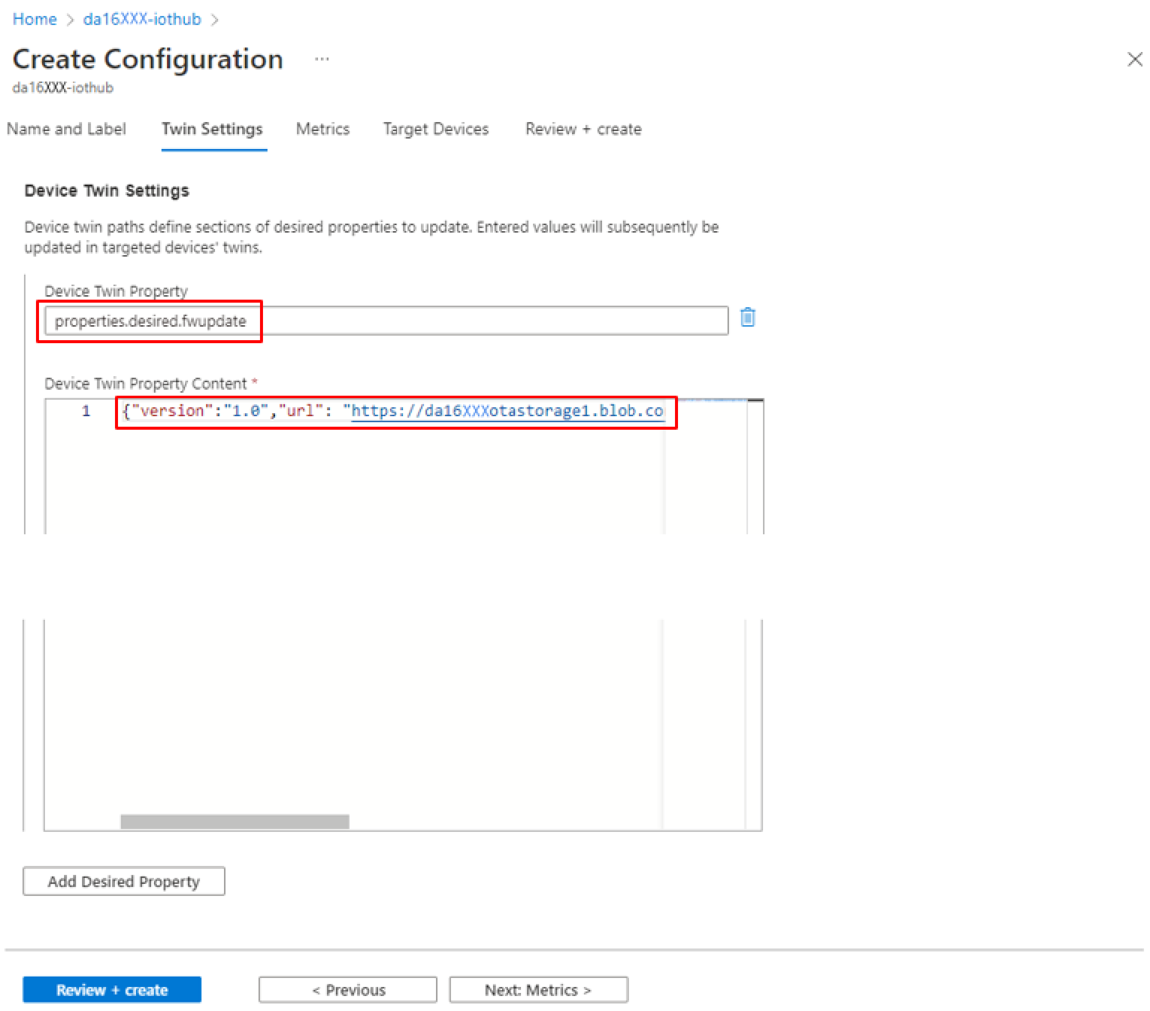
You can add labels to describe a configuration. Labels are a set of case-sensitive string key value pairs, such as 'Version: 3'.

NAME	VALUE
da16XXX-ota-label	1.0
Label name	Label value

Figure 77: Name and Label Section

3. Under the Twin Settings section, follow the steps below:
   a. Set Device Twin Property to: "properties.desired.fwupdate".
   b. In the Device Twin Property Content field, which defines the version and URL, add a JSON formatted string.

      For example: {"version":"1.0","url": "https://da16XXXotastorage1.blob.core.windows.net/da16XXX-img-ontainer"}



Device Twin Settings

Device twin paths define sections of desired properties to update. Entered values will subsequently be updated in targeted devices' twins.

Device Twin Property: properties.desired.fwupdate

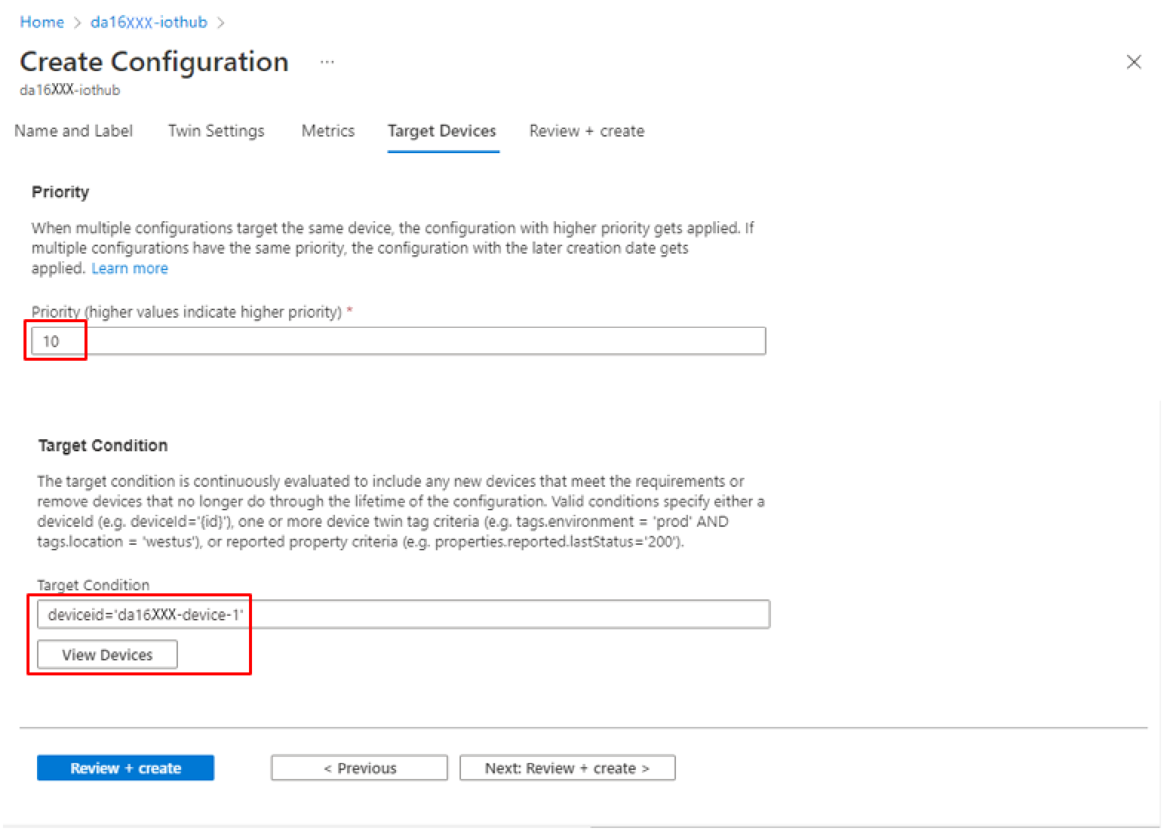
Device Twin Property Content *

```
{"version":"1.0","url": "https://da16XXXotastorage1.blob.co
```

Figure 78: Create Configuration

4. Under the **Target Devices** section, follow the step below:
   a. Set **Priority**, for example “*10*”.
   b. Set **Target Condition** to match the **Device ID** in the devices of Azure IoT Hub. For example, *“deviceid='da16200-device-1'”*.

c. Click **View Devices** to verify the device Id.



Home > da16XXX-iothub >

Create Configuration ...

da16XXX-iothub

Name and Label | Twin Settings | Metrics | Target Devices | Review + create

Priority

When multiple configurations target the same device, the configuration with higher priority gets applied. If multiple configurations have the same priority, the configuration with the later creation date gets applied. Learn more

Priority (higher values indicate higher priority) *

10

Target Condition

The target condition is continuously evaluated to include any new devices that meet the requirements or remove devices that no longer do through the lifetime of the configuration. Valid conditions specify either a deviceId (e.g. deviceId='{id}'), one or more device twin tag criteria (e.g. tags.environment = 'prod' AND tags.location = 'westus'), or reported property criteria (e.g. properties.reported.lastStatus='200').

Target Condition

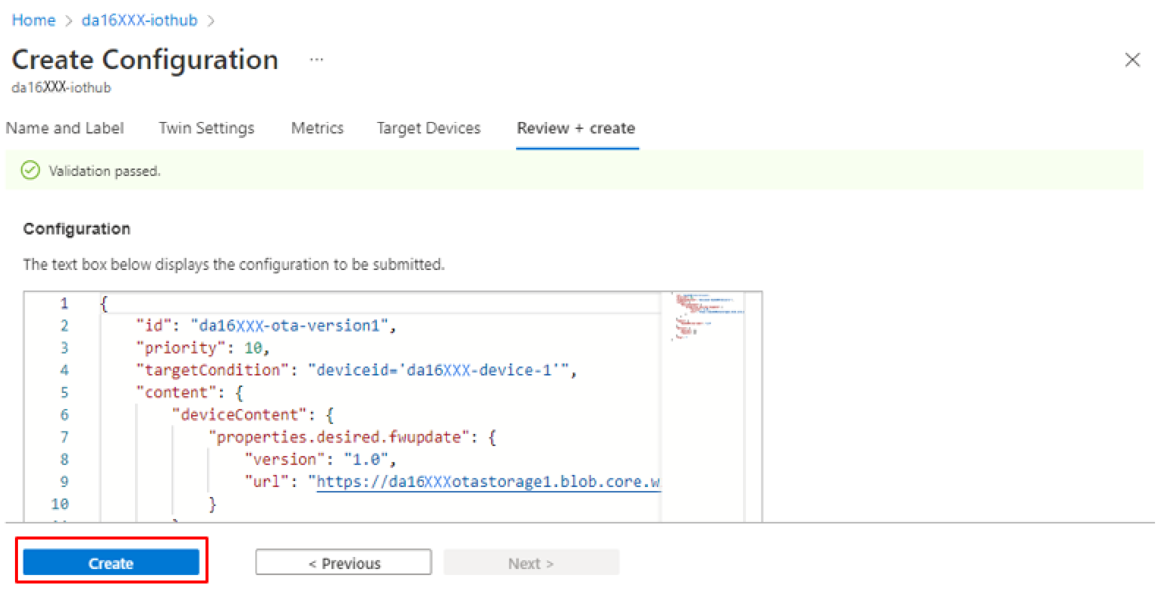
deviceid='da16XXX-device-1'

View Devices

Review + create | < Previous | Next: Review + create >

Figure 79: Target Devices Section

5. Under the **Review + create** section, follow the steps below:
   a. Verify the validation is passed.
   b. Review the configuration that is shown in the text box.
   c. Click **Create** to create the configuration.



Home > da16XXX-iothub >

Create Configuration ...

da16XXX-iothub

Name and Label | Twin Settings | Metrics | Target Devices | Review + create

Validation passed.

Configuration

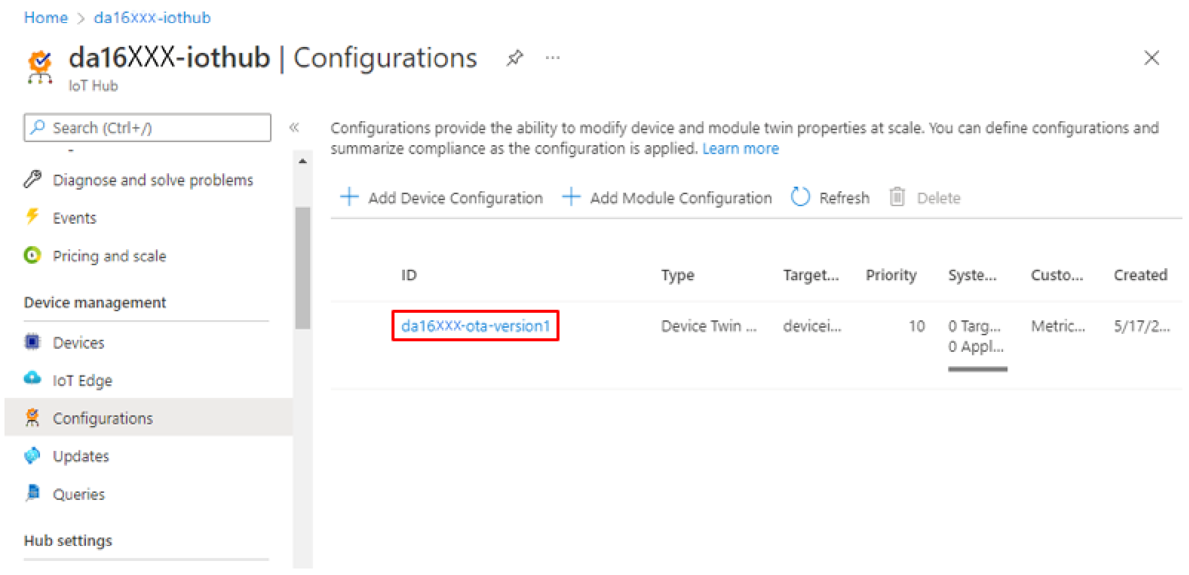
The text box below displays the configuration to be submitted.

```
{
    "id": "da16XXX-ota-version1",
    "priority": 10,
    "targetCondition": "deviceid='da16XXX-device-1'",
    "content": {
        "deviceContent": {
            "properties.desired.fwupdate": {
                "version": "1.0",
                "url": "https://da16XXXotastorage1.blob.core.w
            }
```

Create | < Previous | Next >

Figure 80: Review + Create Section

6. The created configuration is shown in Figure 81.

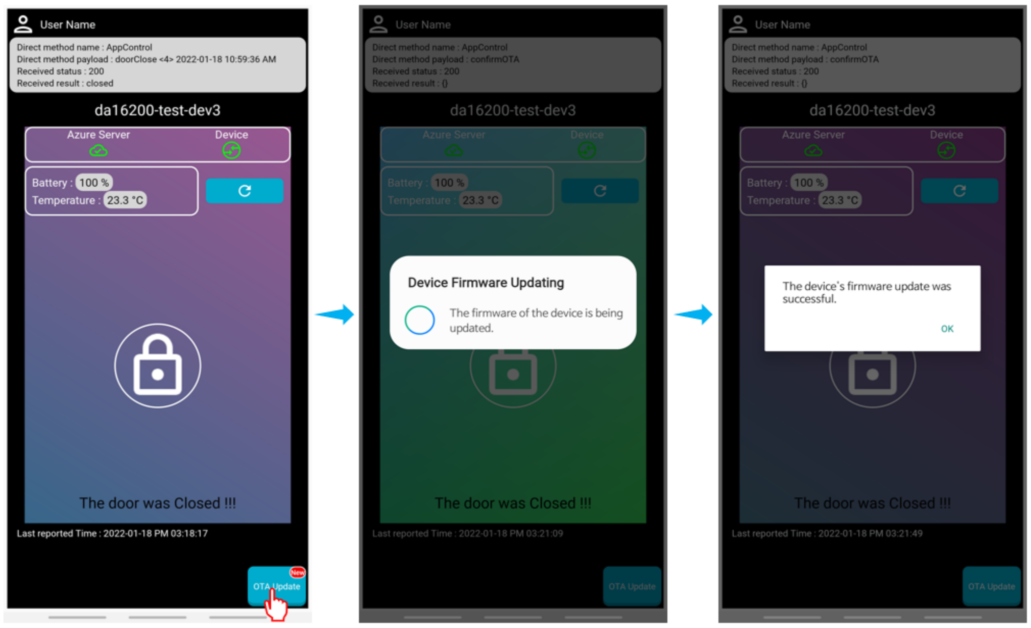


Configurations provide the ability to modify device and module twin properties at scale. You can define configurations and summarize compliance as the configuration is applied. Learn more

ID	Type	Target...	Priority	Syste...	Custo...	Created
da16XXX-ota-version1	Device Twin ...	devicei...	10	0 Targ... 0 Appl...	Metric...	5/17/2...

Figure 81: Created Device Twin Configuration

7. The operation of the Android app is shown in Figure 82.



Direct method name : AppControl
Direct method payload : doorClose <4> 2022-01-18 10:59:36 AM
Received status : 200
Received result : closed
Battery : 100 %
Temperature : 23.3 °C
Last reported Time : 2022-01-18 PM 03:18:17

Direct method name : AppControl
Direct method payload : confirmOTA
Received status : 200
Received result : {}
Device Firmware Updating
The firmware of the device is being updated.
Last reported Time : 2022-01-18 PM 03:21:09

Direct method name : AppControl
Direct method payload : confirmOTA
Received status : 200
Received result : {}
The device's firmware update was successful.
OK
Last reported Time : 2022-01-18 PM 03:21:49

Figure 82: OTA Update Operation

8. The message flow of OTA update is shown in Figure 83.

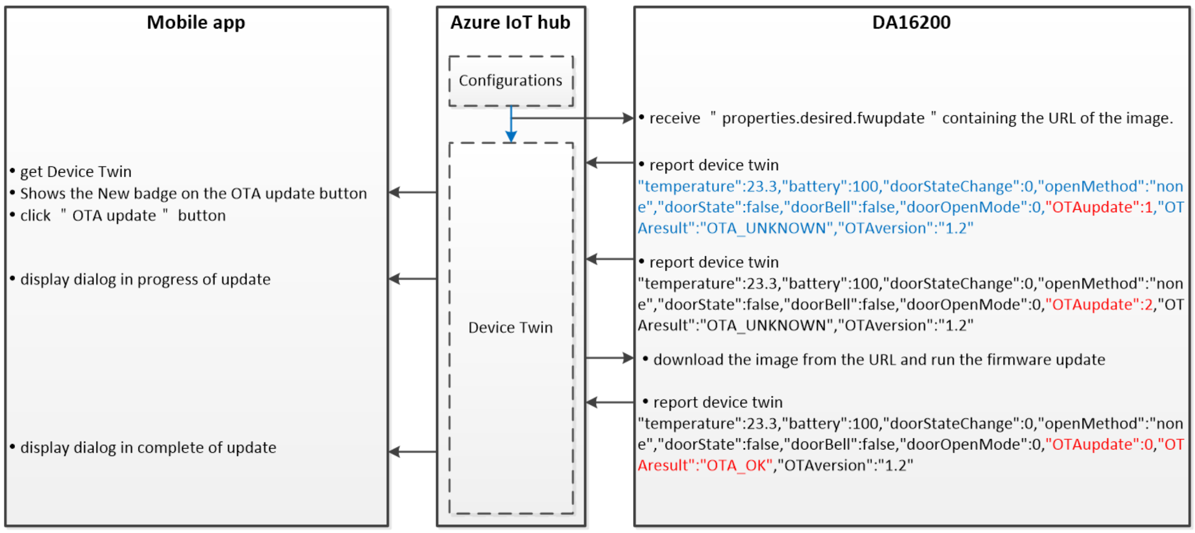
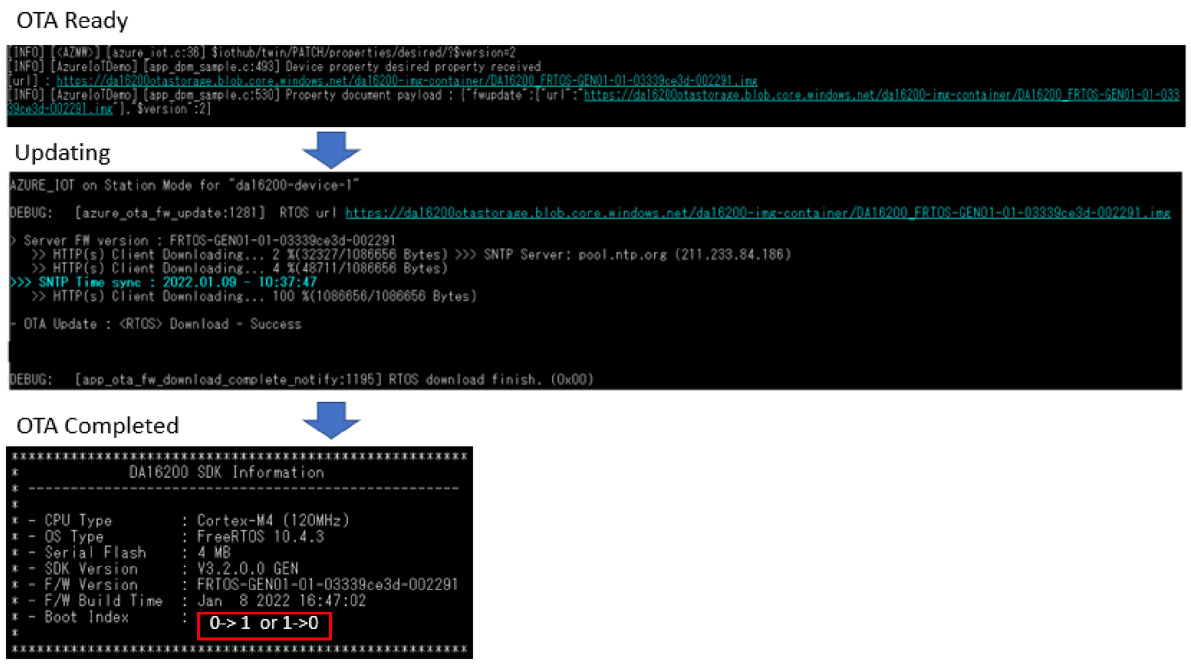

Mobile app
Azure IoT hub
DA16200
Configurations
Device Twin
• receive " properties.desired.fwupdate " containing the URL of the image.
• get Device Twin
• Shows the New badge on the OTA update button
• click " OTA update " button
• report device twin
"temperature":23.3,"battery":100,"doorStateChange":0,"openMethod":"none","doorState":false,"doorBell":false,"doorOpenMode":0,"OTAupdate":1,"OTAresult":"OTA_UNKNOWN","OTAversion":"1.2"
• display dialog in progress of update
• report device twin
"temperature":23.3,"battery":100,"doorStateChange":0,"openMethod":"none","doorState":false,"doorBell":false,"doorOpenMode":0,"OTAupdate":2,"OTAresult":"OTA_UNKNOWN","OTAversion":"1.2"
• download the image from the URL and run the firmware update
• report device twin
"temperature":23.3,"battery":100,"doorStateChange":0,"openMethod":"none","doorState":false,"doorBell":false,"doorOpenMode":0,"OTAupdate":0,"OTAresult":"OTA_OK","OTAversion":"1.2"
• display dialog in complete of update

Figure 83: Message Flows of OTA Update

9. The log from DA16200/DA16600 during OTA process as shown in Figure 84.


OTA Ready

```
[INFO] [<AZMW>] [azure_iot.c:38] $iothub/twin/PATCH/properties/desired/?$version=2
[INFO] [AzureIoTDemo] [app_dpm_sample.c:493] Device property desired property received
[url] : https://da16200otastorage.blob.core.windows.net/da16200-img-container/DA16200_FRTOS-GEN01-01-03339ce3d-002291.img
[INFO] [AzureIoTDemo] [app_dpm_sample.c:530] Property document payload : {"fwupdate":{"url":"https://da16200otastorage.blob.core.windows.net/da16200-img-container/DA16200_FRTOS-GEN01-01-033
39ce3d-002291.img"},"$version":2}
```

Updating

```
AZURE_IOT on Station Mode for "da16200-device-1"

DEBUG:   [azure_ota_fw_update:1281]  RTOS url https://da16200otastorage.blob.core.windows.net/da16200-img-container/DA16200_FRTOS-GEN01-01-03339ce3d-002291.img

> Server FW version : FRTOS-GEN01-01-03339ce3d-002291
   >> HTTP(s) Client Downloading... 2 %(32327/1086656 Bytes) >>> SNTP Server: pool.ntp.org (211.233.84.186)
   >> HTTP(s) Client Downloading... 4 %(48711/1086656 Bytes)
>>> SNTP Time sync : 2022.01.09 - 10:37:47
   >> HTTP(s) Client Downloading... 100 %(1086656/1086656 Bytes)

- OTA Update : <RTOS> Download - Success


DEBUG:   [app_ota_fw_download_complete_notify:1195] RTOS download finish. (0x00)
```

OTA Completed

```
*************************************************************
*                 DA16200 SDK Information
* ---------------------------------------------------------
*
* - CPU Type          : Cortex-M4 (120MHz)
* - OS Type           : FreeRTOS 10.4.3
* - Serial Flash      : 4 MB
* - SDK Version       : V3.2.0.0 GEN
* - F/W Version       : FRTOS-GEN01-01-03339ce3d-002291
* - F/W Build Time    : Jan  8 2022 16:47:02
* - Boot Index        : 0->1 or 1->0
*
*************************************************************
```

Figure 84: OTA Process in DA16200/DA16600

5.5 OTA Update of MCU Firmware

The Azure AT command package provides the ability to perform a safe and secure OTA update of the MCU firmware. Figure 85 shows the process when using the mobile app and the Azure server to update the MCU firmware through the DA16200/DA16600.

Note that the DA16200/DA16600 detects whether the firmware image is for the MCU or for the DA16200/DA16600 depending on the prefix of the firmware image name. If the firmware image is for the MCU, then the firmware image name should be prefixed with “mcu” for example, **mcu**_firmware_20220101.img.

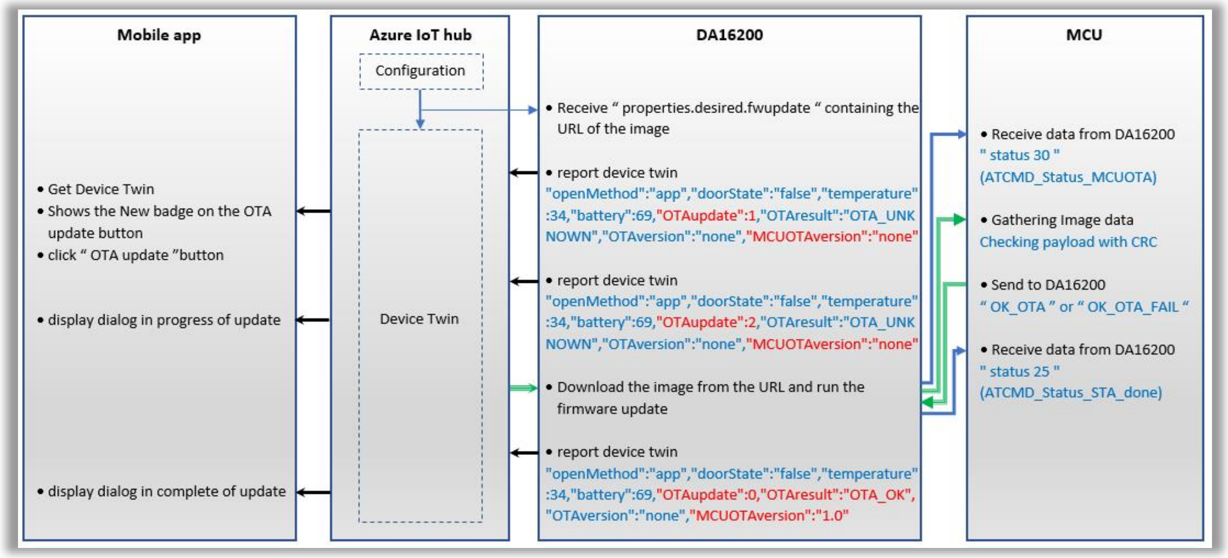

Mobile app
Azure IoT hub
Configuration
Device Twin
DA16200
MCU
• Get Device Twin
• Shows the New badge on the OTA update button
• click " OTA update "button
• display dialog in progress of update
• display dialog in complete of update
• Receive " properties.desired.fwupdate " containing the URL of the image
• report device twin "openMethod":"app","doorState":"false","temperature":34,"battery":69,"OTAupdate":1,"OTAresult":"OTA_UNKNOWN","OTAversion":"none","MCUOTAversion":"none"
• report device twin "openMethod":"app","doorState":"false","temperature":34,"battery":69,"OTAupdate":2,"OTAresult":"OTA_UNKNOWN","OTAversion":"none","MCUOTAversion":"none"
• Download the image from the URL and run the firmware update
• report device twin "openMethod":"app","doorState":"false","temperature":34,"battery":69,"OTAupdate":0,"OTAresult":"OTA_OK","OTAversion":"none","MCUOTAversion":"1.0"
• Receive data from DA16200 " status 30 " (ATCMD_Status_MCUOTA)
• Gathering Image data Checking payload with CRC
• Send to DA16200 " OK_OTA " or " OK_OTA_FAIL "
• Receive data from DA16200 " status 25 " (ATCMD_Status_STA_done)

Figure 85: Block Diagram of MCU OTA

An example MCU firmware image for testing OTA is available in the Azure AT command package:

\MCU\bin\mcu_azure.bin.

Appendix A Azure IoT Hub Device Provisioning Service

Device Provisioning Service (DPS) is a process for registering the DA16200/DA16600 device on the Azure IoT Hub.

To set up DPS, follow the steps below:

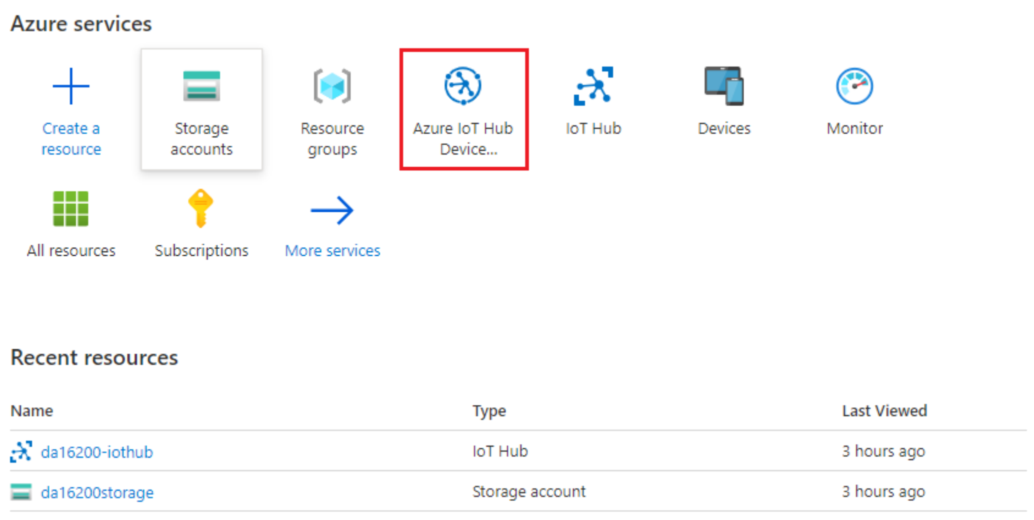
1. Create an IoT Hub .
2. Manage Enrollments.
3. Configure Registration ID Parameters in DA16200/DA16600 SDK.

A.1 Create Azure IoT Hub

DPS is required to register devices on the Azure IoT Hub. The following steps show how to set up DPS and link it with the Azure IoT Hub.

To create DPS:

1. Click **Azure IoT Hub Device Provisioning Service**.

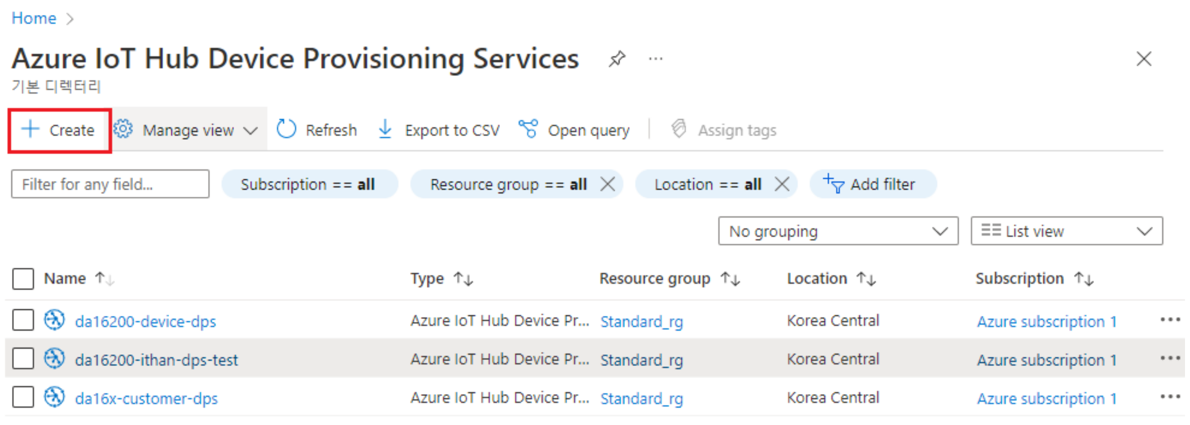


Recent resources

Name	Type	Last Viewed
da16200-iothub	IoT Hub	3 hours ago
da16200storage	Storage account	3 hours ago

Figure 86: Azure IoT Hub Device Provisioning Service

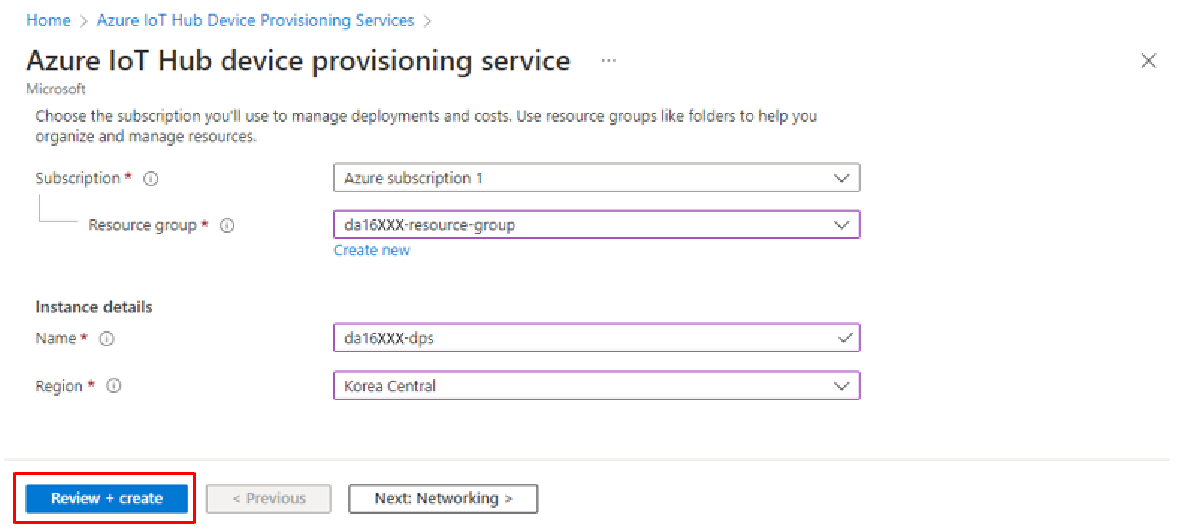
2. Click **Create**.



Name	Type	Resource group	Location	Subscription
da16200-device-dps	Azure IoT Hub Device Pr...	Standard_rg	Korea Central	Azure subscription 1
da16200-ithan-dps-test	Azure IoT Hub Device Pr...	Standard_rg	Korea Central	Azure subscription 1
da16x-customer-dps	Azure IoT Hub Device Pr...	Standard_rg	Korea Central	Azure subscription 1

Figure 87: Create DPS

3. Configure the **Resource group**, **Name** and **Region** as shown in Figure 88, and then click **Review + create**.


Home > Azure IoT Hub Device Provisioning Services >

Azure IoT Hub device provisioning service

Microsoft

Choose the subscription you'll use to manage deployments and costs. Use resource groups like folders to help you organize and manage resources.

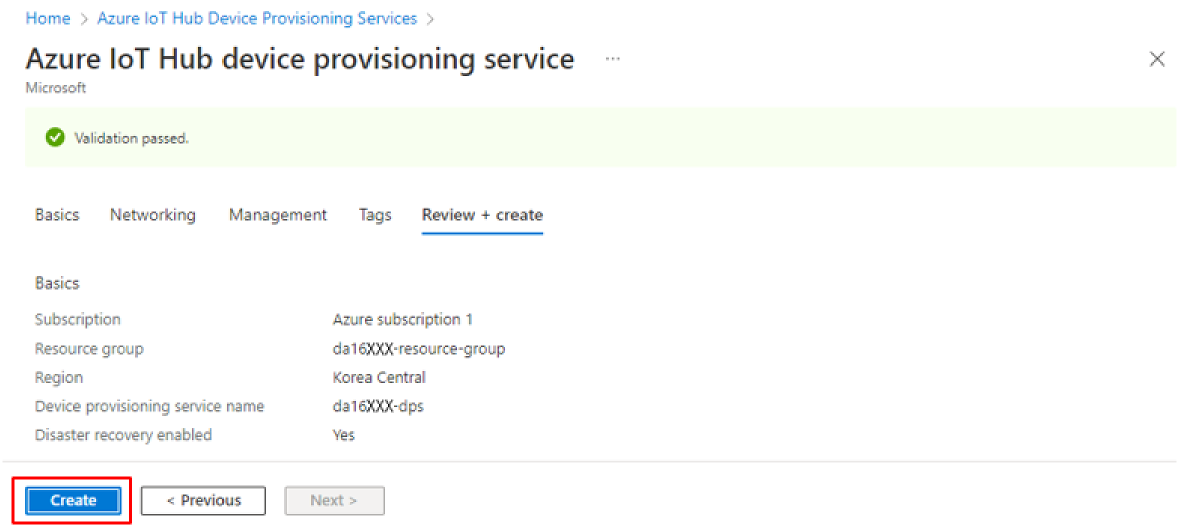
Subscription *	Azure subscription 1
Resource group *	da16XXX-resource-group

Instance details

Name *	da16XXX-dps
Region *	Korea Central

Figure 88: Configure Resource Group, Name, and Region

4. After the validation is passed, click **Create**.


Home > Azure IoT Hub Device Provisioning Services >

Azure IoT Hub device provisioning service

Microsoft

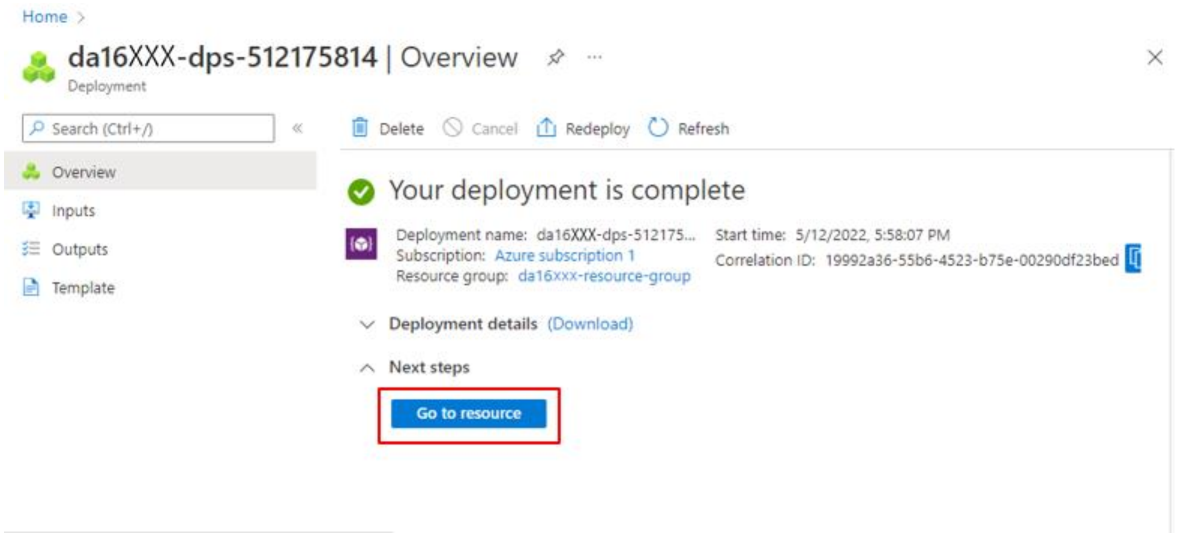
Validation passed.

Basics

Subscription	Azure subscription 1
Resource group	da16XXX-resource-group
Region	Korea Central
Device provisioning service name	da16XXX-dps
Disaster recovery enabled	Yes

Figure 89: Create Button

5. After the deployment of DPS is complete, click **Go to resource**.

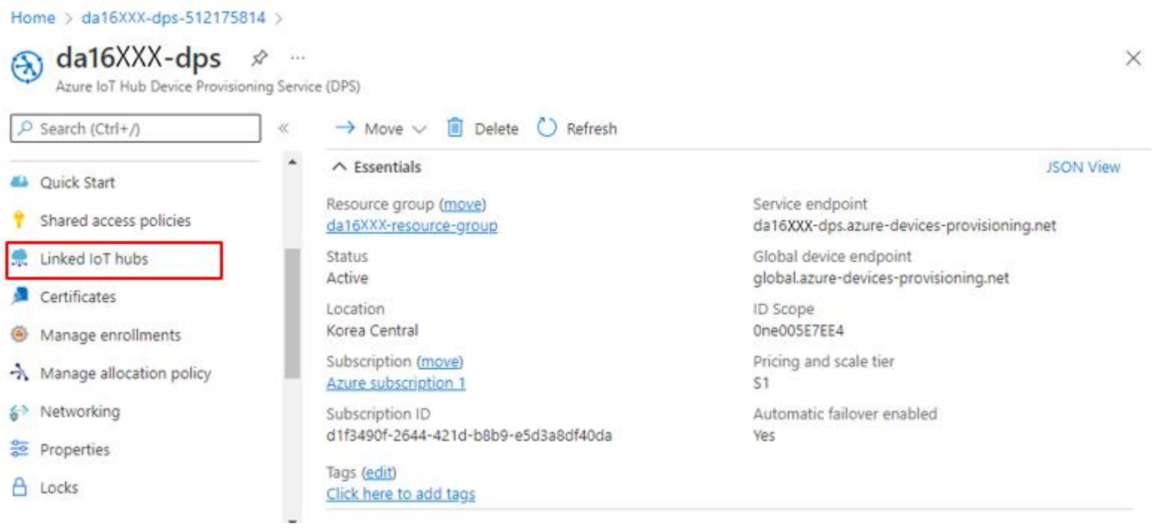


Your deployment is complete

Deployment name: da16XXX-dps-512175...
Subscription: Azure subscription 1
Resource group: da16xxx-resource-group
Start time: 5/12/2022, 5:58:07 PM
Correlation ID: 19992a36-55b6-4523-b75e-00290df23bed

Figure 90: Go to Resource

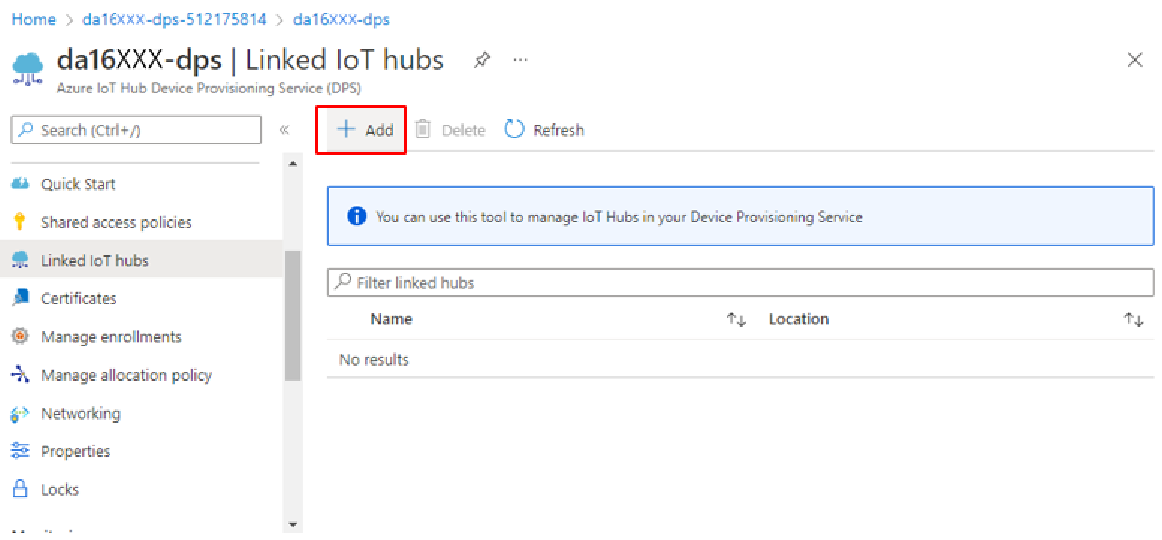
6. Select **Linked IoT hubs** to link the created DPS resource to the Azure IoT Hub.



Resource group (move): da16XXX-resource-group
Status: Active
Location: Korea Central
Subscription (move): Azure subscription 1
Subscription ID: d1f3490f-2644-421d-b8b9-e5d3a8df40da
Tags (edit): Click here to add tags
Service endpoint: da16XXX-dps.azure-devices-provisioning.net
Global device endpoint: global.azure-devices-provisioning.net
ID Scope: 0ne005E7EE4
Pricing and scale tier: S1
Automatic failover enabled: Yes

Figure 91: Linked IoT Hubs

7. Click **Add** to link the Azure IoT Hub.

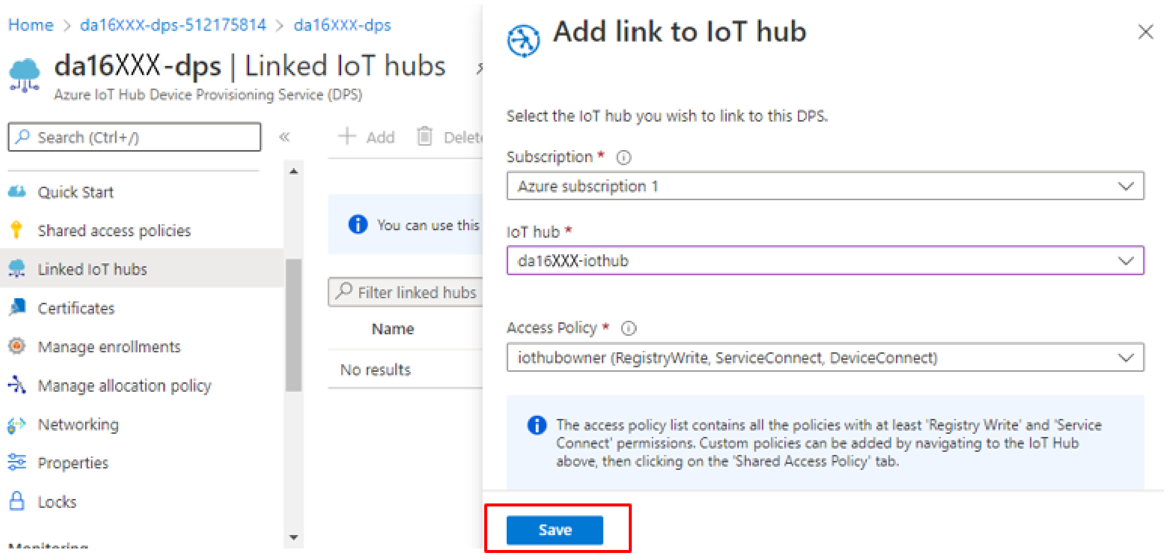


You can use this tool to manage IoT Hubs in your Device Provisioning Service

No results

Figure 92: Add Link to Azure IoT Hub

8. Under **Add link to IoT hub** section, select the **IoT hub** and **Access Policy** as shown in Figure 93, and click **Save**.



Select the IoT hub you wish to link to this DPS.

Subscription * : Azure subscription 1

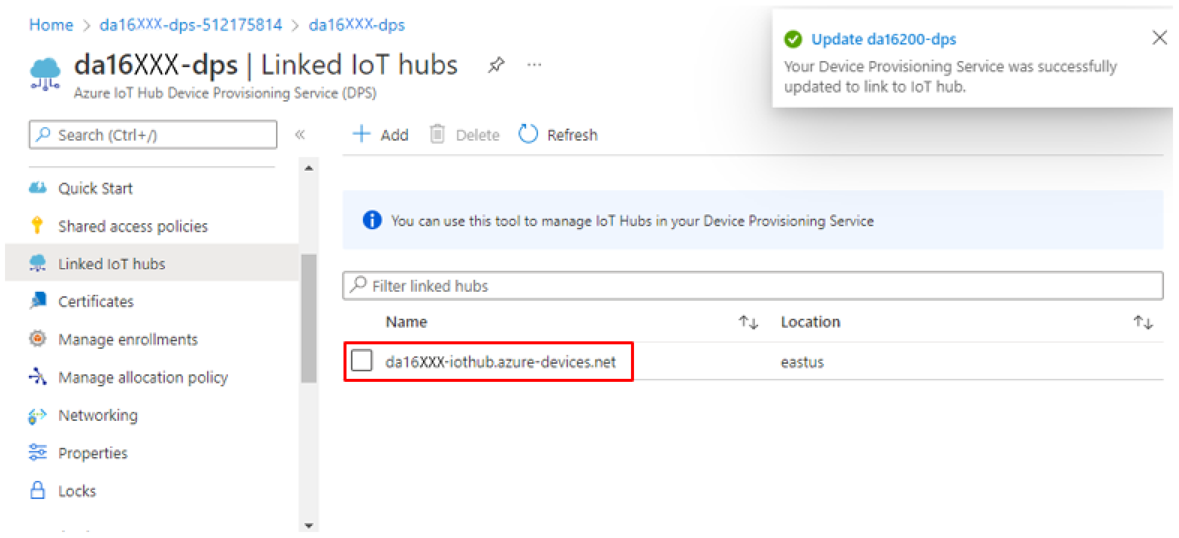
IoT hub * : da16XXX-iothub

Access Policy * : iothubowner (RegistryWrite, ServiceConnect, DeviceConnect)

The access policy list contains all the policies with at least 'Registry Write' and 'Service Connect' permissions. Custom policies can be added by navigating to the IoT Hub above, then clicking on the 'Shared Access Policy' tab.

Figure 93: Configure Link

The IoT Hub that is linked to the DPS appears in the list.



Update da16200-dps

Your Device Provisioning Service was successfully updated to link to IoT hub.

You can use this tool to manage IoT Hubs in your Device Provisioning Service

Name	Location
da16XXX-iothub.azure-devices.net	eastus

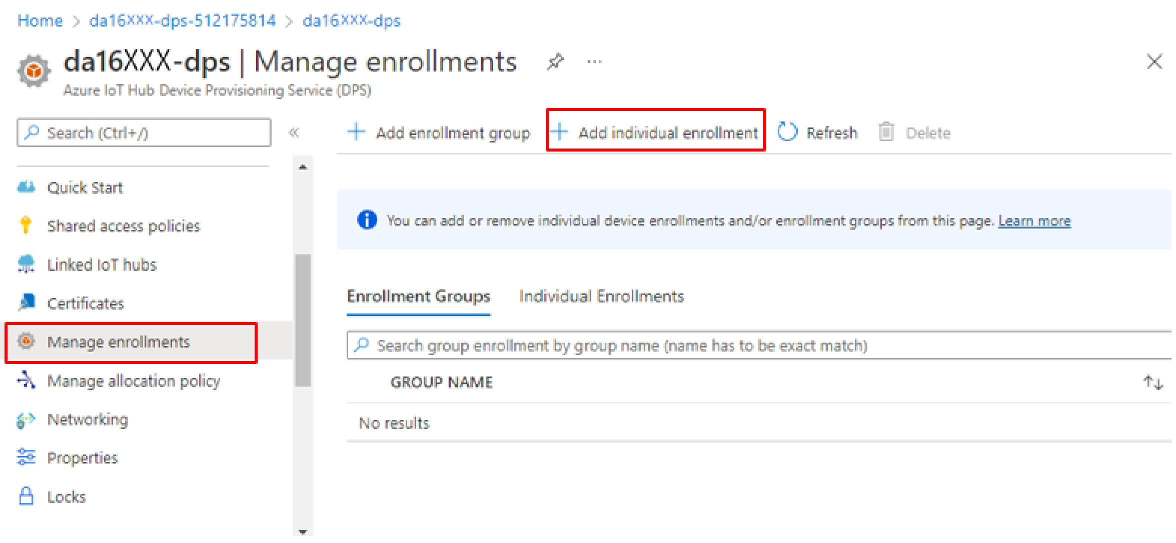
Figure 94: IoT Hub Linked to DPS

A.2 Manage Enrollments

Manage Enrollments is the process of registering a device with the Azure IoT Hub.

To manage enrollments on the Azure IoT Hub:

1. Go to the created DPS and click **Manage enrollments**, and then select **Add individual enrollment**.

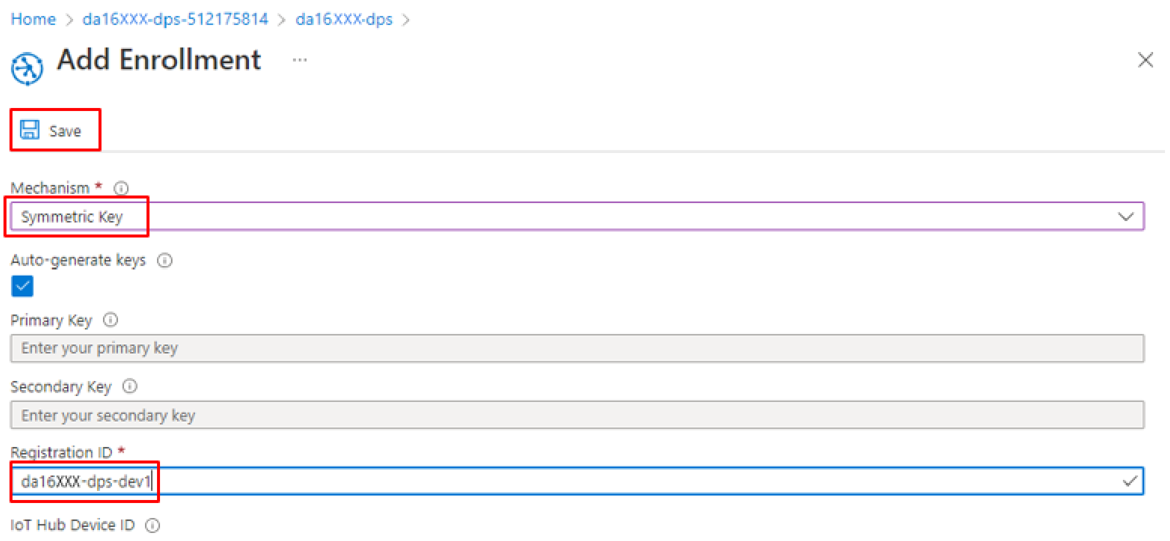


You can add or remove individual device enrollments and/or enrollment groups from this page. Learn more

No results

Figure 95: Manage Enrollments

2. In the **Add enrollment**, follow the steps below, and then click **Save**:
   a. In **Mechanism**, select **Symmetric Key**.
   b. In **Registration ID**, enter value for the device, for example *da16200-dps-dev1*.


Home > da16XXX-dps-512175814 > da16XXX-dps >

Add Enrollment

Save

Mechanism *: Symmetric Key

Auto-generate keys: ✓

Primary Key: Enter your primary key

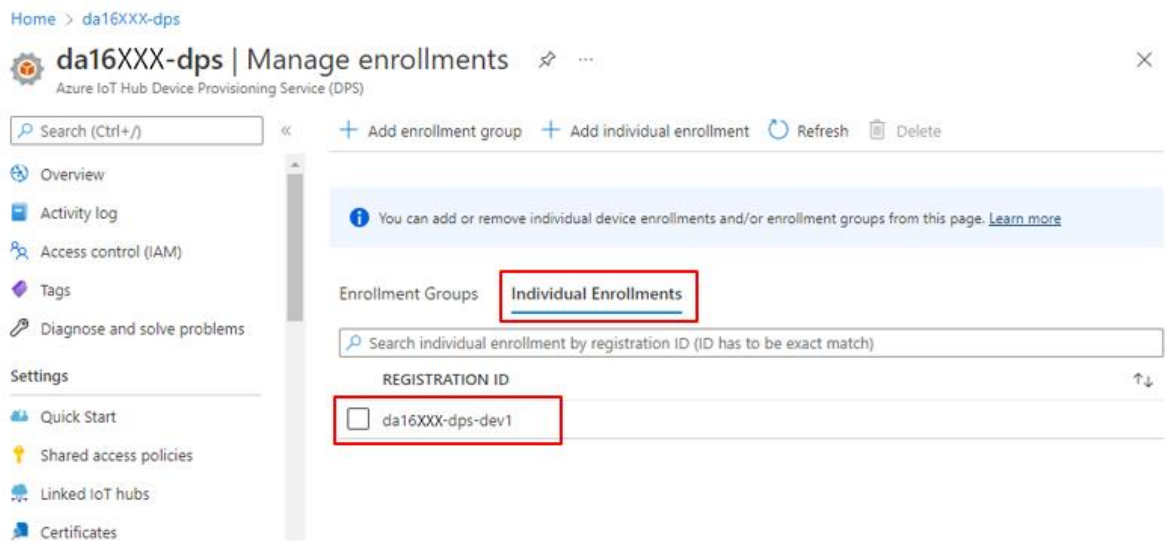
Secondary Key: Enter your secondary key

Registration ID *: da16XXX-dps-dev1

IoT Hub Device ID

Figure 96: Add Enrollment

3. Click **Individual Enrollments** and the created registration ID appears in the list.


Home > da16XXX-dps

da16XXX-dps | Manage enrollments

Azure IoT Hub Device Provisioning Service (DPS)

You can add or remove individual device enrollments and/or enrollment groups from this page. Learn more

Enrollment Groups | Individual Enrollments

Search individual enrollment by registration ID (ID has to be exact match)

REGISTRATION ID
da16XXX-dps-dev1

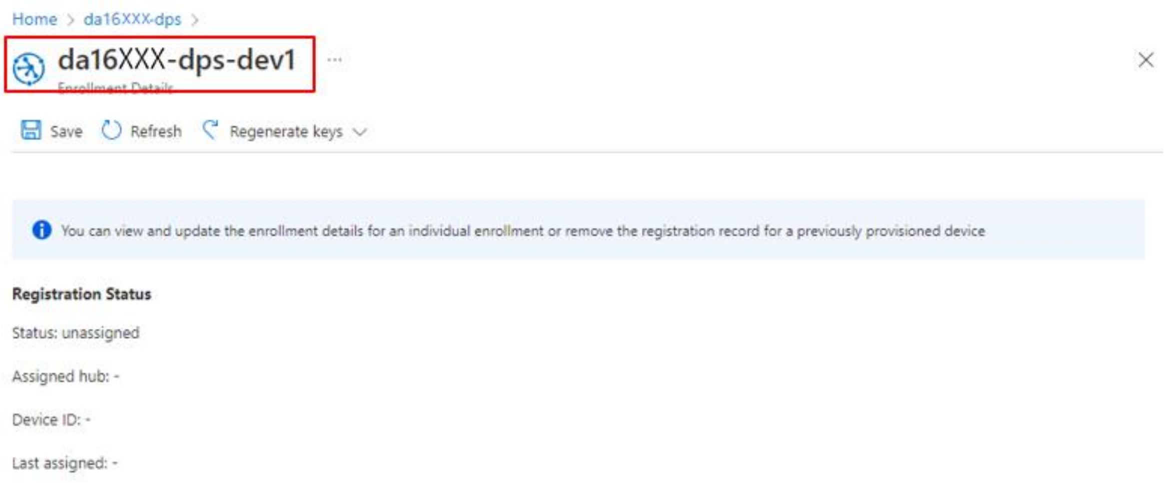
Figure 97: Individual Enrollments

A.3 Configure Registration ID Parameters in DA16200/DA16600 SDK

To enable the Device Provisioning Service, define `ENABLE_AZURE_DPS_EXAMPLE` as `1` in the e^{2}studio.

To configure DPS parameters in the DA16200/DA16600 SDK:

1. In the DA16200/DA16600 SDK, edit app_thing_manager.h and define `APP_USER_MY_THING_NAME` as the **DPS Registration ID**.
   For example: `#define APP_USER_MY_THING_NAME    "da16XXX-dps-dev1"`


Home > da16XXX-dps >

da16XXX-dps-dev1

Save | Refresh | Regenerate keys

You can view and update the enrollment details for an individual enrollment or remove the registration record for a previously provisioned device

Registration Status

Status: unassigned

Assigned hub: -

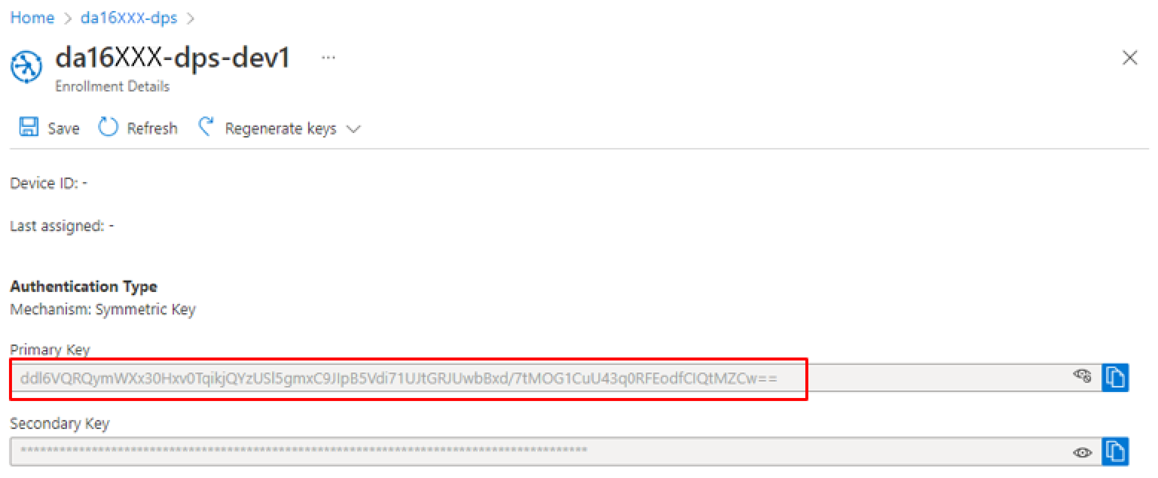
Device ID: -

Last assigned: -

Figure 98: DPS Registration ID

2. In the DA16200/DA16600 SDK, edit app_thing_manager.h and define `APP_USER_MY_SYMMETRIC_KEY` as the **Primary Key** of the DPS Registration ID.

   For example: `#define APP_USER_MY_SYMMETRIC_KEY “ddl6VQRQymWXx30Hxv0TqikjQYzUSl5gmxC9JIpB5Vdi71UJtGRJUwbBxd/7tMOG1CuU43q0RFEodfCIQtMZCw==”`


Home > da16XXX-dps >

da16XXX-dps-dev1

Enrollment Details

Save | Refresh | Regenerate keys

Device ID: -

Last assigned: -

Authentication Type

Mechanism: Symmetric Key

Primary Key

ddl6VQRQymWXx30Hxv0TqikjQYzUSl5gmxC9JIpB5Vdi71UJtGRJUwbBxd/7tMOG1CuU43q0RFEodfCIQtMZCw==

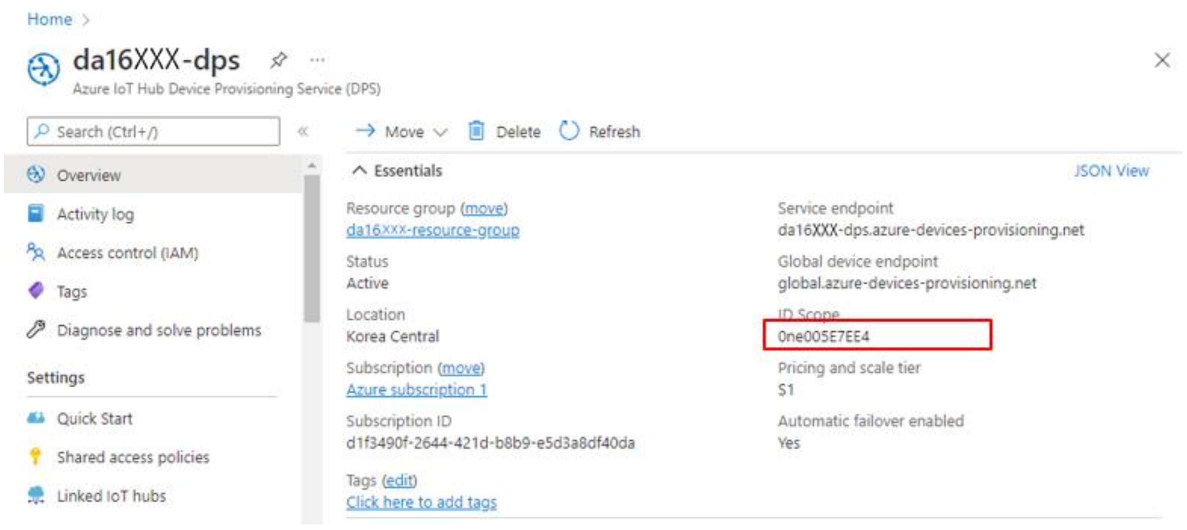
Secondary Key

****

Figure 99: Primary Key of DPS Registration ID

3. In the DA16200/DA16600 SDK, edit app_azure_user_conf.h and define `ID_SCOPE` as the **ID Scope** of the **DPS overview** window.

   For example: `#define ID_SCOPE  “0ne005E7EE4”`



Home >

da16XXX-dps

Azure IoT Hub Device Provisioning Service (DPS)

Resource group (move)
da16XXX-resource-group

Status
Active

Location
Korea Central

Subscription (move)
Azure subscription 1

Subscription ID
d1f3490f-2644-421d-b8b9-e5d3a8df40da

Tags (edit)
Click here to add tags

Service endpoint
da16XXX-dps.azure-devices-provisioning.net

Global device endpoint
global.azure-devices-provisioning.net

ID Scope
0ne005E7EE4

Pricing and scale tier
S1

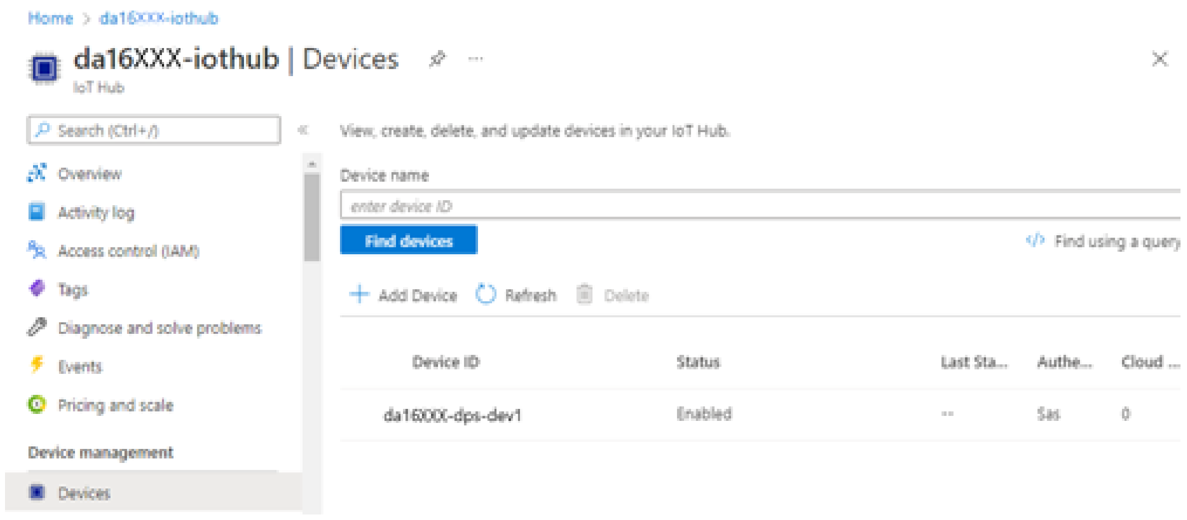
Automatic failover enabled
Yes

Figure 100: ID Scope of DPS

4. After enabled DPS feature is defined in app_azure_user_conf.h of the DA16200/DA16600 SDK as follows, compile it and download the image file to the device, and then reset the device.

```
#define ENABLE_AZURE_DPS_EXAMPLE                                    1
```

The DPS device is registered as shown in Figure 101 in the linked IoT Hub.



Home > da16XXX-iothub

da16XXX-iothub | Devices

IoT Hub

View, create, delete, and update devices in your IoT Hub.

Device ID	Status	Last Sta...	Authe...	Cloud ...
da16XXX-dps-dev1	Enabled	--	Sas	0

Figure 101: Registered DPS Device

Appendix B Azure IoT AT Command List

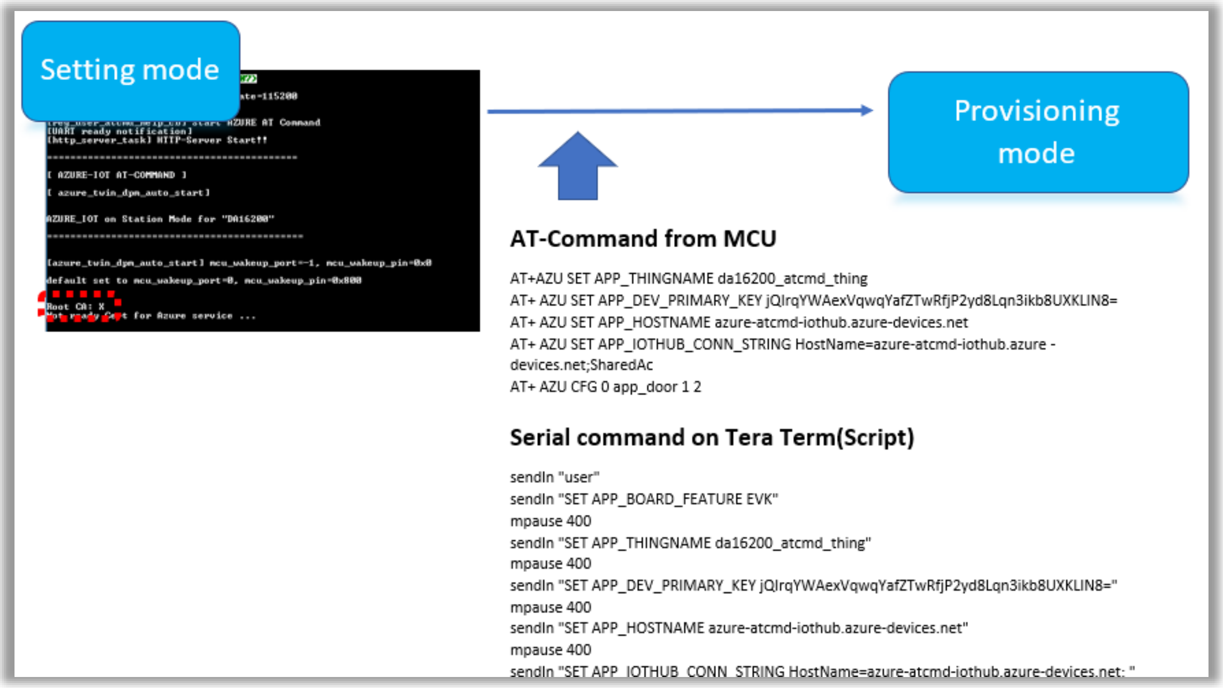
B.1 Operating Modes

There are three operating modes:

- Setting Mode for features configuration
- Provisioning Mode for network connection
- Communicating Mode for running

B.1.1 Setting Mode

After uploading a firmware image and rebooting, the DA16200/DA16600 enters setting mode. In this mode, all Azure IoT settings can be configured using the **SET** command and a specific topic can be configured using the **CFG** command. For proper operation of Azure IoT, the TLS certificate keys should be set. All configuration data is stored before calling the factory reset command (see Figure 102).



```
ate=115200

[reg_user_atcmd_help_cb] start AZURE AT Command
[UART ready notification]
[http_server_task] HTTP-Server Start!!
===========================================
[ AZURE-IOT AT-COMMAND ]
[ azure_twin_dpm_auto_start]

AZURE_IOT on Station Mode for "DA16200"
============================================

[azure_twin_dpm_auto_start] mcu_wakeup_port=-1, mcu_wakeup_pin=0x0
default set to mcu_wakeup_port=0, mcu_wakeup_pin=0x800
Root CA: X
Not ready Cert for Azure service ...
```

AT-Command from MCU

```
AT+AZU SET APP_THINGNAME da16200_atcmd_thing
AT+ AZU SET APP_DEV_PRIMARY_KEY jQlrqYWAexVqwqYafZTwRfjP2yd8Lqn3ikb8UXKLIN8=
AT+ AZU SET APP_HOSTNAME azure-atcmd-iothub.azure-devices.net
AT+ AZU SET APP_IOTHUB_CONN_STRING HostName=azure-atcmd-iothub.azure -
devices.net;SharedAc
AT+ AZU CFG 0 app_door 1 2
```

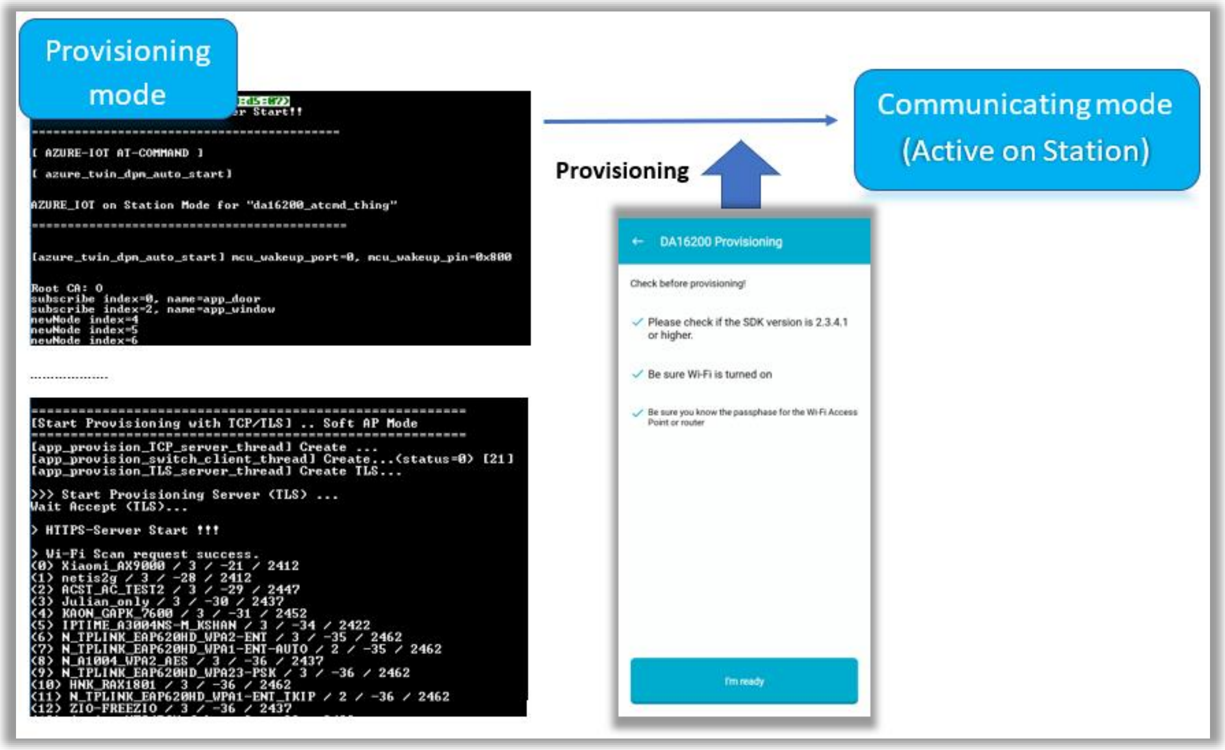
Serial command on Tera Term(Script)

```
sendln "user"
sendln "SET APP_BOARD_FEATURE EVK"
mpause 400
sendln "SET APP_THINGNAME da16200_atcmd_thing"
mpause 400
sendln "SET APP_DEV_PRIMARY_KEY jQlrqYWAexVqwqYafZTwRfjP2yd8Lqn3ikb8UXKLIN8="
mpause 400
sendln "SET APP_HOSTNAME azure-atcmd-iothub.azure-devices.net"
mpause 400
sendln "SET APP_IOTHUB_CONN_STRING HostName=azure-atcmd-iothub.azure-devices.net; "
```

Figure 102: Setting Mode

B.1.2 Provisioning Mode

When in provisioning mode, the DA16200/DA16600 can be provisioned using an Android or iOS device. During provisioning, the MCU only receives a report on the provisioning status. When provisioning is complete, the DA16200/DA16600 enters communication mode automatically after rebooting (see Figure 103).



```
r Start!!
==========================================
[ AZURE-IOT AT-COMMAND ]
[ azure_twin_dpm_auto_start]

AZURE_IOT on Station Mode for "da16200_atcmd_thing"
==========================================

[azure_twin_dpm_auto_start] mcu_wakeup_port=0, mcu_wakeup_pin=0x800

Root CA: 0
subscribe index=0, name=app_door
subscribe index=2, name=app_window
newNode index=4
newNode index=5
newNode index=6
```

```
============================================================
[Start Provisioning with TCP/TLS] .. Soft AP Mode
============================================================
[app_provision_TCP_server_thread] Create ...
[app_provision_switch_client_thread] Create...(status=0) [21]
[app_provision_TLS_server_thread] Create TLS...

>>> Start Provisioning Server (TLS) ...
Wait Accept (TLS)...

> HTTPS-Server Start !!!

> Wi-Fi Scan request success.
(0) Xiaomi_AX9000 / 3 / -21 / 2412
(1) netis2g / 3 / -28 / 2412
(2) ACST_AC_TEST2 / 3 / -29 / 2447
(3) Julian_only / 3 / -30 / 2437
(4) KAON_GAPK_7600 / 3 / -31 / 2452
(5) IPTIME_A3004NS-M_KSHAN / 3 / -34 / 2422
(6) N_TPLINK_EAP620HD_WPA2-ENT / 3 / -35 / 2462
(7) N_TPLINK_EAP620HD_WPA1-ENT-AUTO / 2 / -35 / 2462
(8) N_A1004_WPA2_AES / 3 / -36 / 2437
(9) N_TPLINK_EAP620HD_WPA23-PSK / 3 / -36 / 2462
(10) HNK_RAX1801 / 3 / -36 / 2462
(11) N_TPLINK_EAP620HD_WPA1-ENT_TKIP / 2 / -36 / 2462
(12) ZIO-FREEZIO / 3 / -36 / 2437
```

Check before provisioning!

✓ Please check if the SDK version is 2.3.4.1 or higher.

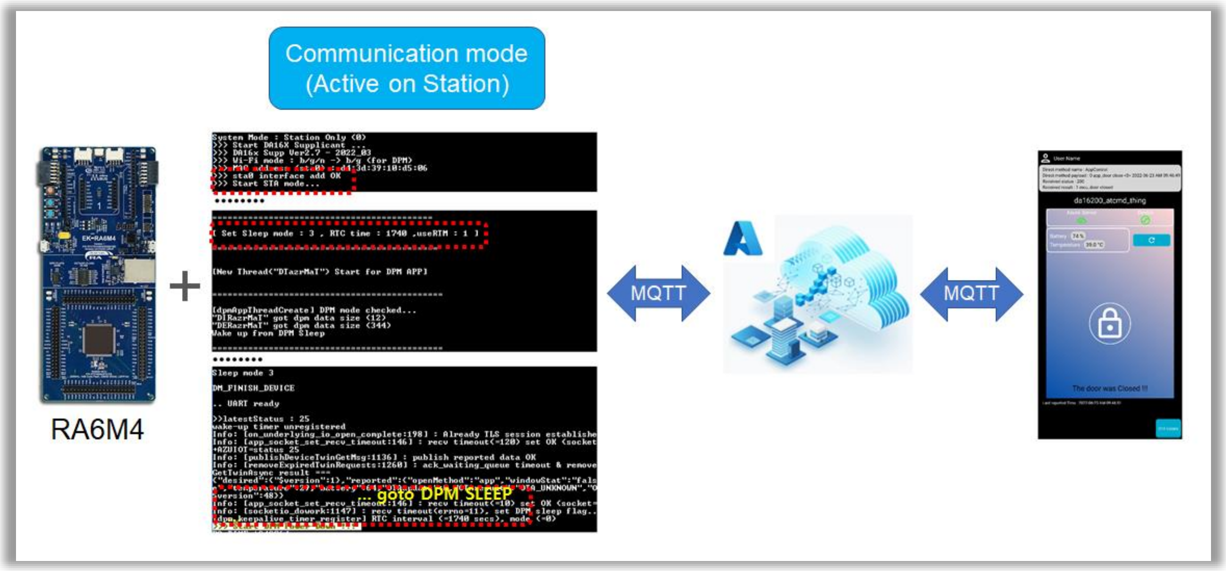
✓ Be sure Wi-Fi is turned on

✓ Be sure you know the passphase for the Wi-Fi Access Point or router

Figure 103: Provisioning Mode

B.1.3 Communication Mode

The DA16200/DA16600 Communicating Mode is used by the MCU to communicate (send and receive) topic values with an Azure server (see Figure 104).



```
System Mode : Station Only (0)
>>> Start DA16X Supplicant ...
>>> DA16x Supp Ver2.7 - 2022_03
>>> Wi-Fi mode : b/g/n -> b/g (for DPM)
>>> MAC address (sta0) : d4:3d:39:10:d5:06
>>> sta0 interface add OK
>>> Start STA mode...
```

```
==========================================
[ Set Sleep mode : 3 , RTC time : 1740 ,useRTM : 1 ]
==========================================

[New Thread("DIazrMaT") Start for DPM APP]

==========================================

[dpmAppThreadCreate] DPM mode checked...
"DIRazrMaT" got dpm data size (12)
"DERazrMaT" got dpm data size (344)
Wake up from DPM Sleep
==========================================
```

```
Sleep mode 3

DM_FINISH_DEVICE

.. UART ready

>>latestStatus : 25
wake-up timer unregistered
Info: [on_underlying_io_open_complete:198] : Already TLS session establishe
Info: [app_socket_set_recv_timeout:146] : recv timeout(=120) set OK (socket
+AZUIOT=status 25
Info: [publishDeviceTwinGetMsg:1136] : publish reported data OK
Info: [removeExpiredTwinRequests:1260] : ack_waiting_queue timeout & remove
GetTwinAsync result ===
{"desired":{"$version":1},"reported":{"openMethod":"app","windowStat":"fals
e","temperature":27,"battery":64,                            "OTA_UNKNOWN","O
$version":48}}
Info: [app_socket_set_recv_timeout:146] : recv timeout(=10) set OK (socket=
Info: [socketio_dowork:1147] : recv timeout(errno=11), set DPM sleep flag..
[dpm_keepalive_timer_register] RTC interval (=1740 secs), mode (=0)
```

Figure 104: Communication Mode

B.2 How to Report Topic and Update Device Twin

B.2.1 Configure Device Twin Topic

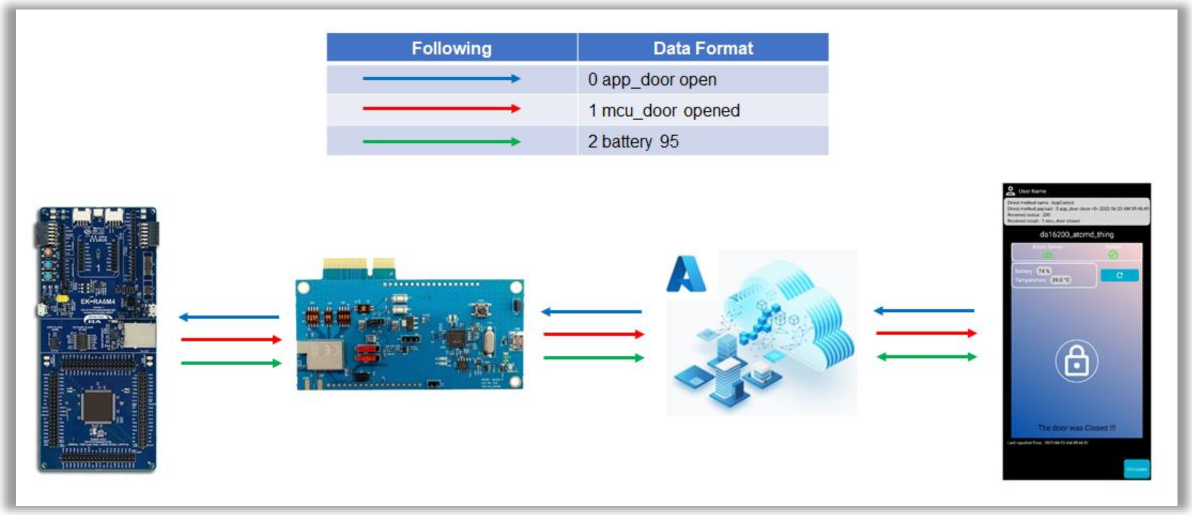
- Topics are configured as shown in Table 5

- The MCU and Mobile App should be designed based on the topics shown in Table 5
- The MCU pushes the topics in Table 5 to the DA16200/DA16600 using AT commands

The DA16200/DA16600 facilitates the communication between the MCU and the phone as shown in Figure 105.

Table 5: Configuration of Topics

Number	Name	Value Type	CMD Type	Value
0	app_door	1: String	2: subscribe	“open”/”close”
1	mcu_door	1: String	0: publish	“opened”/”closed”
4	battery	0: Integer	1: shadow	Battery value (0~100)
5	temperature	2: Float	1: shadow	Temperature value
6	doorState	1: String	1: shadow	“true”/”false”



Following	Data Format
→	0 app_door open
→	1 mcu_door opened
→	2 battery 95

Figure 105: Communication from MCU to Phone

B.3 AT Command List

B.3.1 Basic Set

Table 6: Basic Set of MCU to DA16200/DA16600

Head	Main	Sub	Parameters
AT+AZU=	SET	APP_THINGNAME	Set the device thing name. Used to choose a device by its thing name during provisioning
		APP_DEV_PRIMARY_KEY	Set the device primary key.
		APP_HOSTNAME	Set the Azure server hostname .
		APP_IOTHUB_CONN_STRING	Set the IoT Hub connection string.

Head	Main	Sub	Parameters
		SLEEP_MODE	Set the DPM sleep mode 1 - not used 2 - not used 3 - connected sleep. The connection is retained even during DPM.
		USE_DPM	Defines the operation of sleep mode 3 0 - no DPM. Used during debug 1 - DPM mode
		RTC_TIME	Set the wake-up time for Sleep mode 2
		DPM_KEEP_ALIVE	Set the keep-alive time between the IoT device and the AP Default value is 30*1000 microseconds
		USE_WAKE_UP	Set the wake-up time for full-boot mode Default value is set to 0. (0 = unused)
		TIM_WAKE_UP	Set the period to check a beacon frame from the AP Default value is set to10
EX) AT+AZU=SET APP_THINGNAME *AssignedDeviceID* AT+AZU=SET APP_HOSTNAME *azure-atcmd-iothub.azure-devices.net*			

B.3.2 TLS Certificate

Table 7: TLS Certificate from MCU to DA16200/DA16600

Start Code	Sub Code	Type	End Code
\x1b	C0,	Root CA Self-Signed, well known Has root Certificate public key Signed by root certificate private key	\x03
EX) send "\x1b" over UART send "C0,-----BEGIN CERTIFICATE-----\n" "MIIDQTCCAimgAwIBAgITBmyfz5m/jAo ….. over UART send "\x03"			

B.3.3 PIN MUX

Table 8: PIN MUX from MCU to DA16200/DA16600

Head	Main	Sub	Parameters
AT+AZU=	SET	NV_PIN_AMUX	AMUX_UART1d 4 /* UART1(RXD, TXD) */ AMUX_GPIO 9 /* GPIOA [1:0] */
		NV_PIN_BMUX	BMUX_UART1d 4 /* UART1(RXD, TXD) */ BMUX_GPIO 8 /* GPIOA [3:2] */

		NV_PIN_CMUX	CMUX_UART1d 6 /* UART1(RXD, TXD) */ CMUX_GPIO 8 /* GPIOA [5:4] */
		NV_PIN_DMUX	DMUX_UART1d 4 /* UART1(RXD, TXD) */ DMUX_GPIO 8 /* GPIOA [7:6] */
		NV_PIN_EMUX	EMUX_GPIO 8 /* GPIOA [9:8] */
		NV_PIN_FMUX	FMUX_GPIO 6 /* GPIOA [11:10] */
		NV_PIN_UMUX	UMUX_GPIO 2 /* GPIOC [8:6] */
		APP_MCU_WKAEUP_PORT	GPIO_UNIT_A 0 GPIO_UNIT_C 2 /*Support only GPIO 6,7,8 */
		APP_MCU_WKAEUP_PIN	GPIO_PIN0 ~ GPIO_PIN11
		UART_CFG	[baud-rate]
Note: Default pin mux is BMUX, and uses GPIOA2/GPIOA3 for UART1 and GPIOA9 for MCU wakeup. AT+AZU=SET NV_PIN_BMUX BMUX_UART1d AT+AZU=SET NV_PIN_EMUX EMUX_GPIO AT+AZU=SET APP_MCU_WKAEUP_PORT GPIO_UNIT_A AT+AZU=SET APP_MCU_WKAEUP_PIN GPIO_PIN9			

B.3.4 Configure Data as Topics

Table 9: Configuration Data from MCU to DA16200/DA16600

Head	Main	Sub	Parameters
AT+AZU=	CFG	[number] [name] [value-type] [MQTT-type]	● number: Index to identify the saved topic Increase by 1 when setting a new topic Max value is 10 (total supported topics is 10) ● name: String specifying the topic name ● value-type 0 - Integer type 1 - String type 2 - Float type ● MQTT-type 0 - Publish: The prompt command is used to send a value from the MCU to the phone. This means, door state = true/false 1 - Shadow: The value is sent to the device twin and will be updated on the phone the next time it is connected. 2 - Subscribe: The prompt command is used to send a value from the phone to the MCU. This means, door open command.
Ex) AT+AZU=CFG 0 app_door 1 2 AT+AZU=CFG 4 battey 0 1 AT+AZU=CFG 1 mcu_door 1 0			

B.3.5 Command – MCU to DA16200/DA16600

Table 10: Command from MCU to DA16200/DA16600

Head	Main	Sub	Description
AT+AZU=	CMD	FACTORY_RESET	Resets the Azure IoT configuration to the factory default. All values stored in NVRAM are cleared. Use the "SET" and "CFG" commands to set the Azure IoT configuration.
		RESET_TO_AP	Switches to AP mode keeping the values set in NVRAM. The previous values in NVRAM will be kept.
		GET_STATUS	Gets the current Azure IoT status. The MCU can read the current status from the DA16200/DA16600 at any time.
		RESTART	Reboots the device keeping the current mode and status
		MCU_DATA	Used by the MCU to set a CFG parameter in the DA16200/DA16600.The value must be the same format as defined by the CFG setting. Parameters: [number] [name] [value]
Ex) AT+AZU=CMD FACTORY_RESET AT+AZU=CMD MCU_DATA 1 mcu_door opened			

B.3.6 Command – DA16200/DA16600 to MCU

Table 11: Command from DA16200/DA16600 to MCU

Head	Main	Parameters	Description
+AZUIOT	SERVER_DATA	[number] [name] [value]	Used by the DA16200/DA16600 to set a CFG parameter in the MCU The value must be the same format as defined by the CFG setting.
+AZUIOT	CMD_TO_MCU	update	Used by the DA16200/DA16600 to request the status of devices such as sensors, batteries, and doors from the MCU. The DA16200/DA16600 maintains the values obtained from the MCU and forwards them when requested by an external phone app or by an MQTT ping-pong wake-up event.
Ex) +AZUIOT SERVER_DATA 0 app_door open +AZUIOT CMD_TO_MCU update			

B.3.7 DA16200/DA16600 Status – DA16200/DA16600 to MCU

Table 12: Status from DA16200/DA16600 to MCU

Status	Value	Parameters
IDLE	-1	Initial state of AZU-IoT application Sent when a system error occurs. This means, network connection failure
Done factory reset	0	Sent after completes factory reset by "CMD FACTORY_RESET"
Boot Ready	1	Sent when entering AZU-IoT application mode

Status	Value	Parameters
Need configuration	5	Sent if there is no setting MCU should set and configure with the SET and CFG command
Start AP mode	10	Sent when being started to AP mode It should need to process provisioning with Phone
Network OK	15	Sent when it is OK to connect AP without problem
Network fail	16	Sent when it fails to connect AP with any problem Normally, it will happen during provisioning failure by the wrong SSID or PW It needs to go to AP mode by MCU send" RESET_TO_AP" command
Start STA	20	Not defined yet
Done STA	25	Sent when entering sleep mode for DPM
MCUOTA	30	Sent when start MCU OTA process.
EX) +AZUIOT STATUS 15		

Appendix C Troubleshooting

C.1 Operational Issue

When UI buttons are not visible or not showing up properly while using the mobile app, try to uninstall and install it again. The first time running the mobile app after reinstalling it, make sure that the app can access the location of the device as described in Test Provisioning on Android/iPhone sections of [2].

Revision History

Revision	Date	Description
1.4	Jan. 26, 2024	Added Troubleshooting section
1.3	Sep. 27, 2023	Merged documents: ● UM-WI-058, DA16200 DA16600 Getting Started with Azure-IoT ● UM-WI-060, DA16200 DA16600 Getting Started with AT-Command for Azure-IoT
1.2	July 31, 2023	● Updated the reference section and images ● Changed the name of Dialog Semiconductor to Renesas Electronics
1.1	Sep. 01, 2022	Added description of DA16600
1.0	May 27, 2022	Initial release

Status Definitions

Status	Definition
DRAFT	The content of this document is under review and subject to formal approval, which may result in modifications or additions.
APPROVED or unmarked	The content of this document has been approved for publication.

RoHS Compliance

Renesas Electronics' suppliers certify that its products are in compliance with the requirements of Directive 2011/65/EU of the European Parliament on the restriction of the use of certain hazardous substances in electrical and electronic equipment. RoHS certificates from our suppliers are available on request.

DA16200 DA16600 Getting Started with Azure® IoT

Important Notice and Disclaimer

Corporate Headquarters

TOYOSU FORESIA, 3-2-24 Toyosu

Koto-ku, Tokyo 135-0061, Japan

www.renesas.com

Contact Information

For further information on a product, technology, the most up-to-date version of a document, or your nearest sales office, please visit:

https://www.renesas.com/contact/

Trademarks