Microsoft

Azure RTOS IoT Quick Connect Sample STM32L475-DISCO IoT Node using STM32CubeIDE

User Guide

Published: May 2020

For the latest information, please see
azure.com/rtos

This document is provided “as-is”. Information and views expressed in this document, including URL and other Internet Web site references, may change without notice.

This document does not provide you with any legal rights to any intellectual property in any Microsoft product. You may copy and use this document for your internal, reference purposes.

Azure RTOS provides OEMs with components to secure communication and to create code and data isolation using underlying MCU/MPU hardware protection mechanisms. It is ultimately the responsibility of the device builder to ensure the device fully meets the evolving security requirements associated with its specific use case.

FileX supports the Microsoft exFAT file system format. Your use of exFAT technology in your products requires a separate license from Microsoft. Please see the following link for further details on exFAT licensing:

https://www.microsoft.com/en-us/legal/intellectualproperty/mtl/exfat-licensing.aspx

Part Number: 000-1059

Revision 6.0

Example for STM32L475 Discovery kit for IoT Node

The following steps detail how to configure, build and execute the X-Ware IoT Platform Microsoft Azure integration example on the STM32L475 Discovery kit for IoT Node, using STM32Cube IDE 1.3.0 or later development tools. Figure 1 shows a picture of the kit.

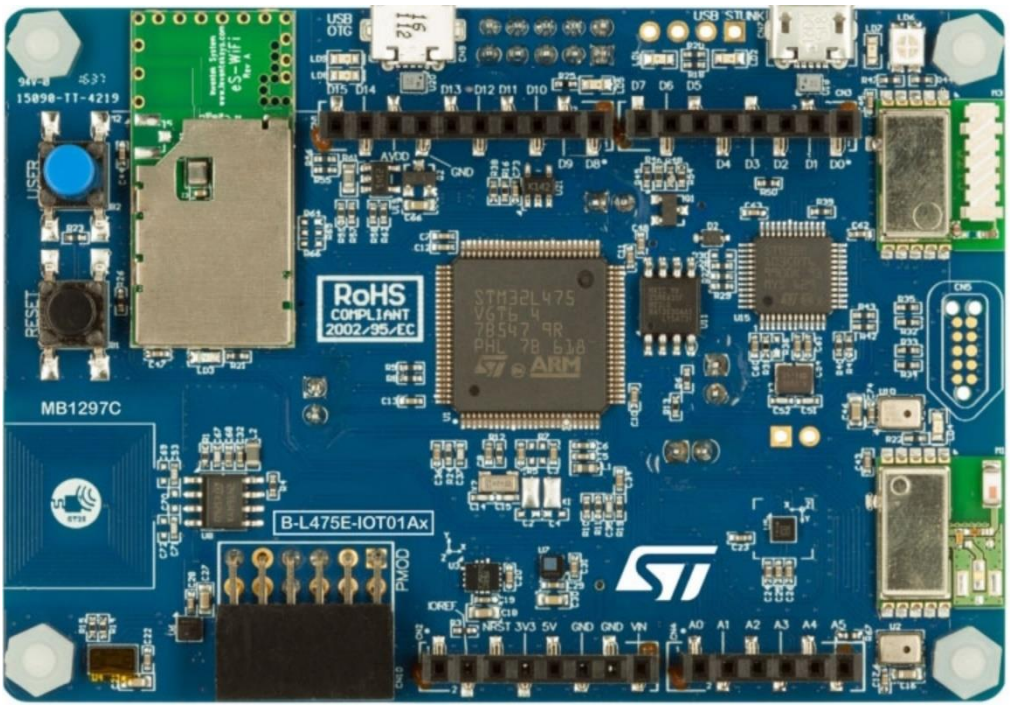


Figure 1: STM32L475 Discovery kit for IoT Node

1.1 Configure Azure IoT hub for the demo

Step 1: Create an IoT hub and Device using the Azure portal.

The Azure IoT hub user guide can be found at:

https://docs.microsoft.com/azure/iot-hub/iot-hub-create-through-portal

Save the **IoT Hub Connection String** for later use.

IoT Hub -> Setting -> Shared access policies -> iothubowner -> Connection string

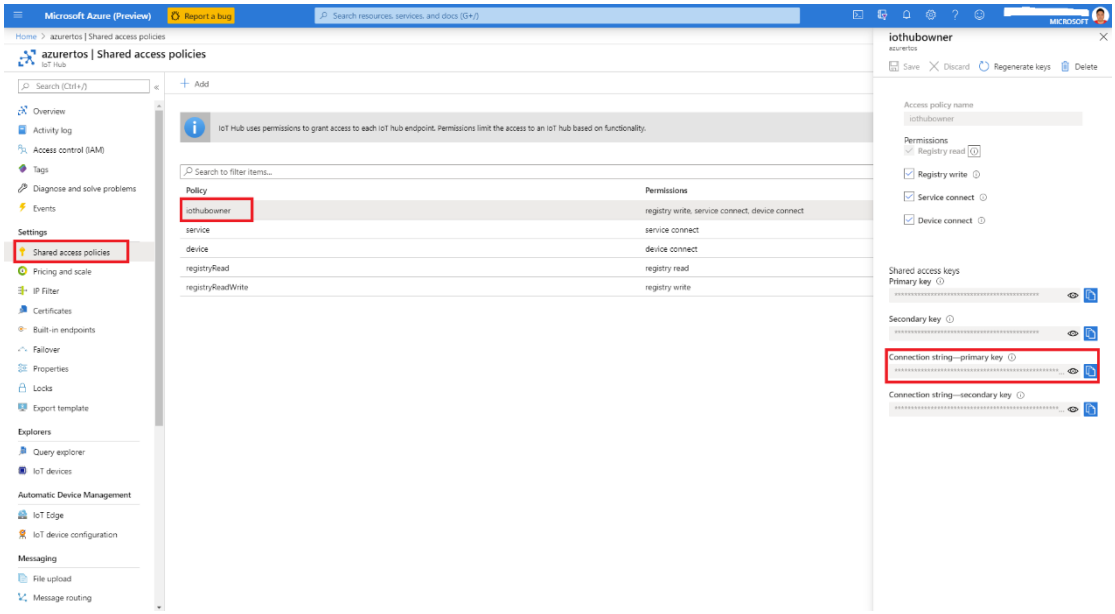


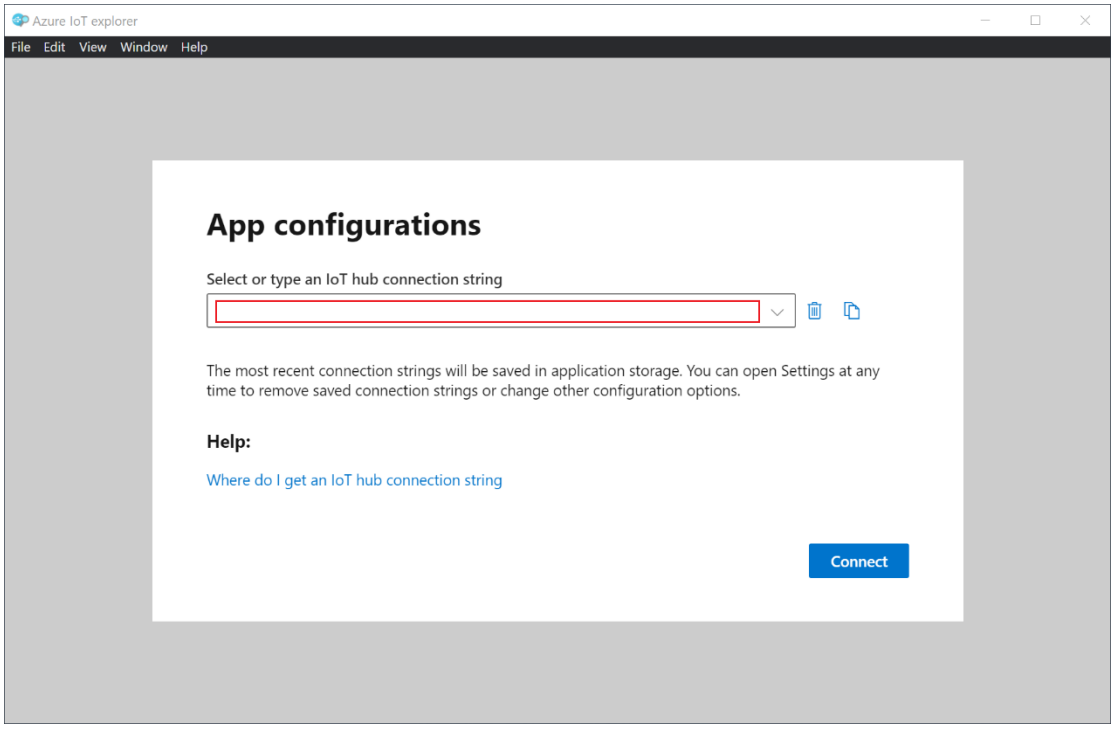
Figure 2 IoT Hub Configuration

Step 2: Install Azure IoT Explorer

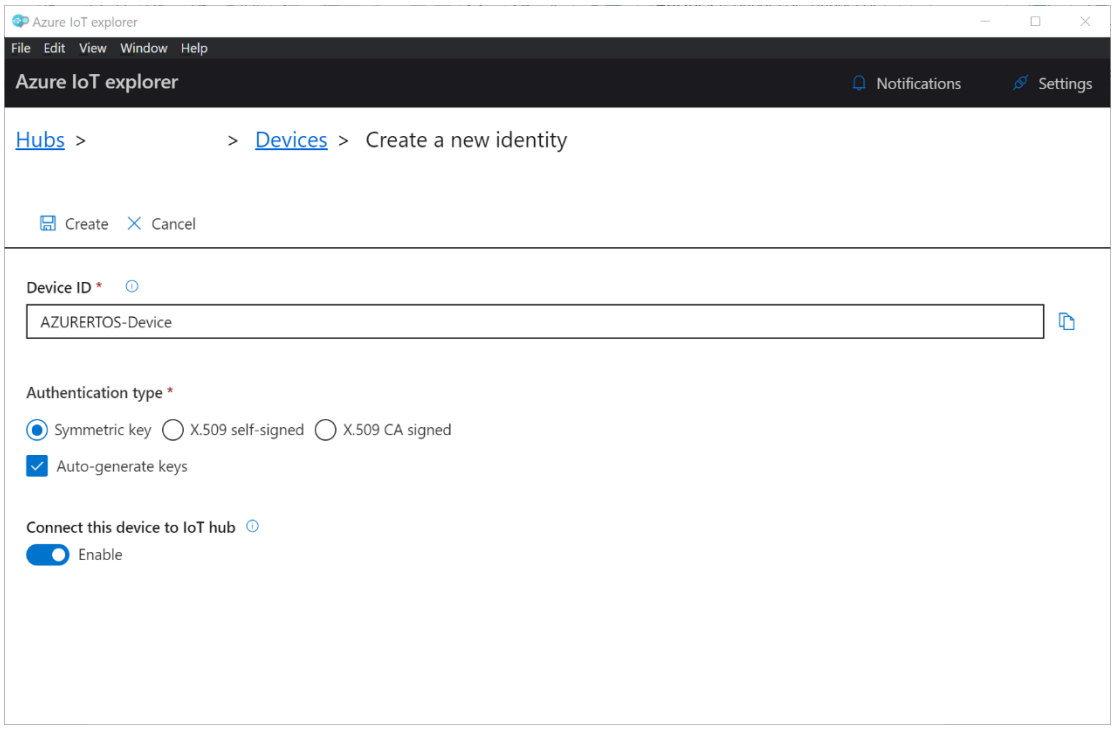
The installer can be found at:
https://github.com/Azure/azure-iot-explorer/releases

Step 3: Get device credentials

1. Launch Azure IoT Explorer, paste the IoT Hub connection string you just got and select ***Connect***.



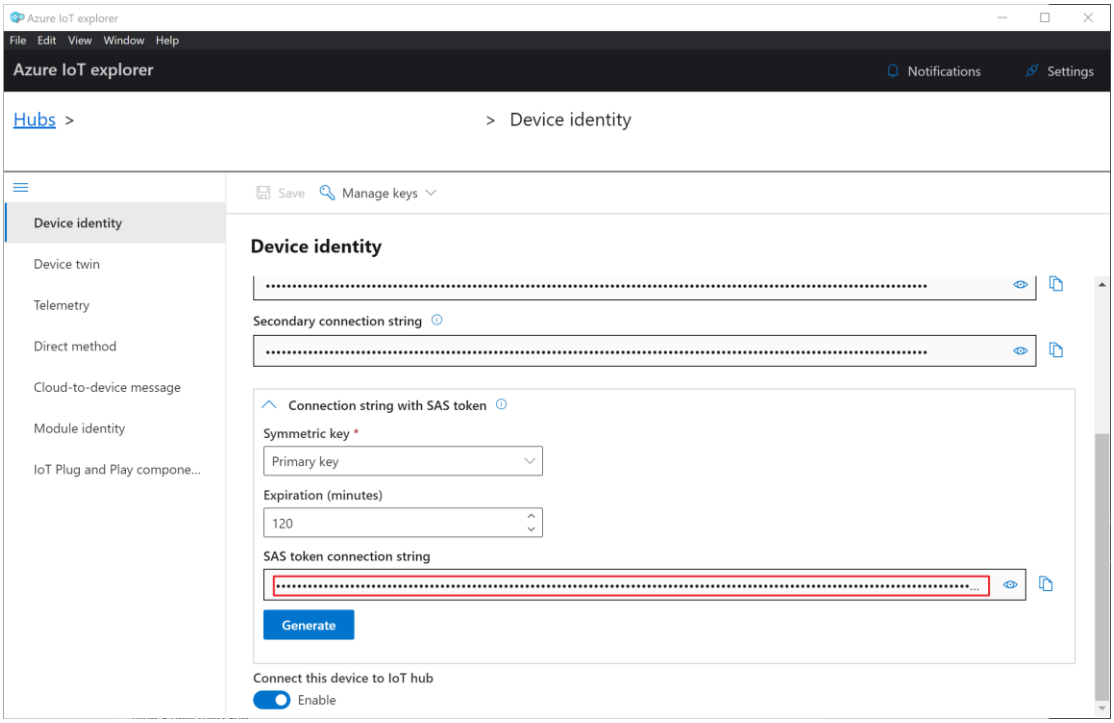
2. Select ***New*** to create a new IoT device. Enter a device ID for your device and keep the rest options as default. Then select ***Create***.



3. Select from the list for the device you just created. In the ***Device identity*** tab, copy ***Device ID*** to Notepad for later use. Then scroll down to find the ***Connection string with SAS token*** section, select ***Primary key*** from the dropdown list, and set the ***Expiration (minutes)*** a big longer than 5 minutes by default (e.g. 120). Then select ***Generate*** and copy ***SAS token***

connection string to Notepad.

For the ***SAS token connection string***, please be aware that we only need the portion of the string starts from ***SharedAccessSignature sr=***. You can see the below code configuration for the required string format.

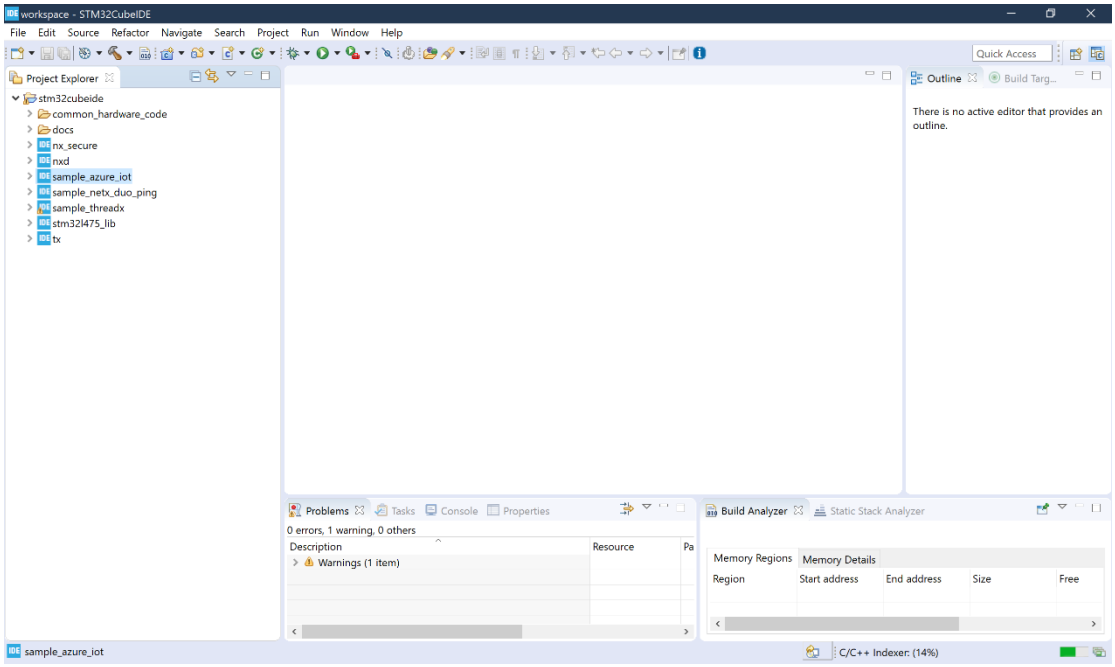


4. Also copy the ***IoT Hub host name prefix*** from the highlighted area to the Notepad. You need all above three device credentials for the sample code to connect to the IoT Hub.

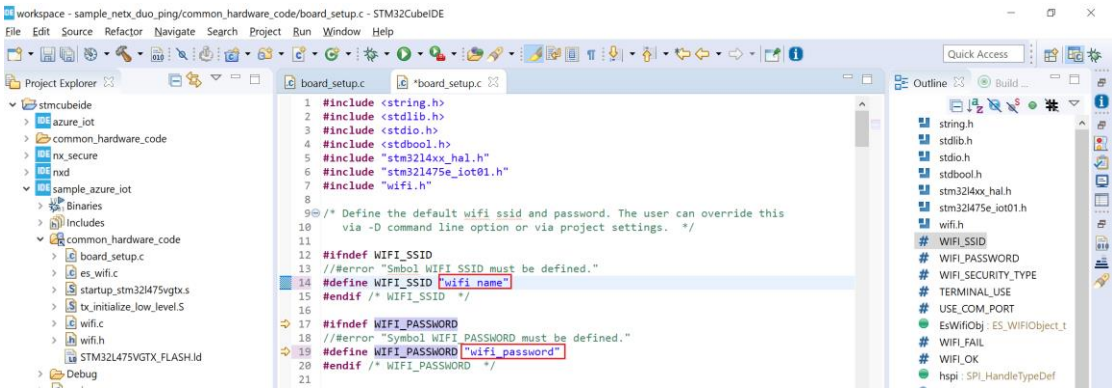
1.2 Configure and execute the example

Step 1: Once the code is downloaded. Open the workspace by

STM32CubeIDE at ***{INSTALL_DIR}\azure_rtos\b-l475e-iot01a\stm32cubeide.*** And find the ***sample_azure_iot*** project.



Find "***board_setup.c***" within the "***common_hardware_code***" folder. Update your WiFi settings:



Step 2: Update the Host Name, Device ID and Device SAS noted in previous step in ***{INSTALL_DIR}\azure_rtos\b-l475e-iot01a\stm32cubeide\sample_azure_iot\sample_azure_iot.c***

```
//
// TODO`s: Configure core settings of application for your IoTHub, replace the
[IoT Hub Name] and [Device ID] as yours. Use Device Explorer to generate [SAS].
//

#ifndef HOST_NAME
```

```
#define HOST_NAME                                "{Your IoT Hub Name}.azure-
devices.net"
#endif /* HOST_NAME */

#ifndef DEVICE_ID
#define DEVICE_ID                                "{Your Device ID}"
#endif /* DEVICE_ID */

#ifndef DEVICE_SAS
#define DEVICE_SAS                               "{Your Device SAS Token}"
#endif /* DEVICE_SAS */

//
// END TODO section
//
```

The following shows example values:

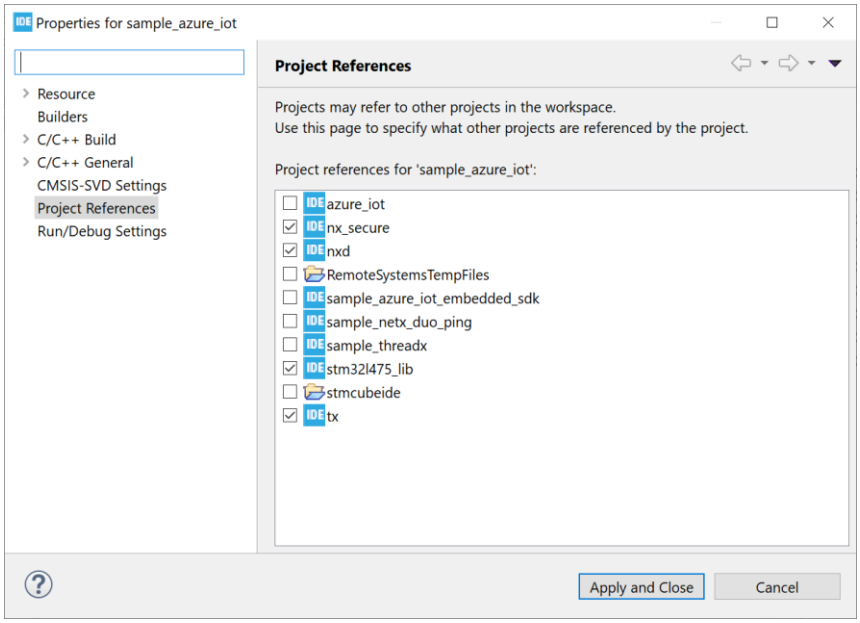
```
HOST_NAME="azurertos.azure-devices.net"

DEVICE_ID="AZURERTOS-DEVICE"

DEVICE_SAS="SharedAccessSignature sr=azurertos.azure-
devices.net%2Fdevices%2FAZURERTOS-DEVICE&sig=......%2Fc%3D&se=1587436430"
```

Step 3: Build the project

Confirm you have all the project dependencies select by right click on the ***sample_azure_iot*** project and select ***Properties > Project References.***

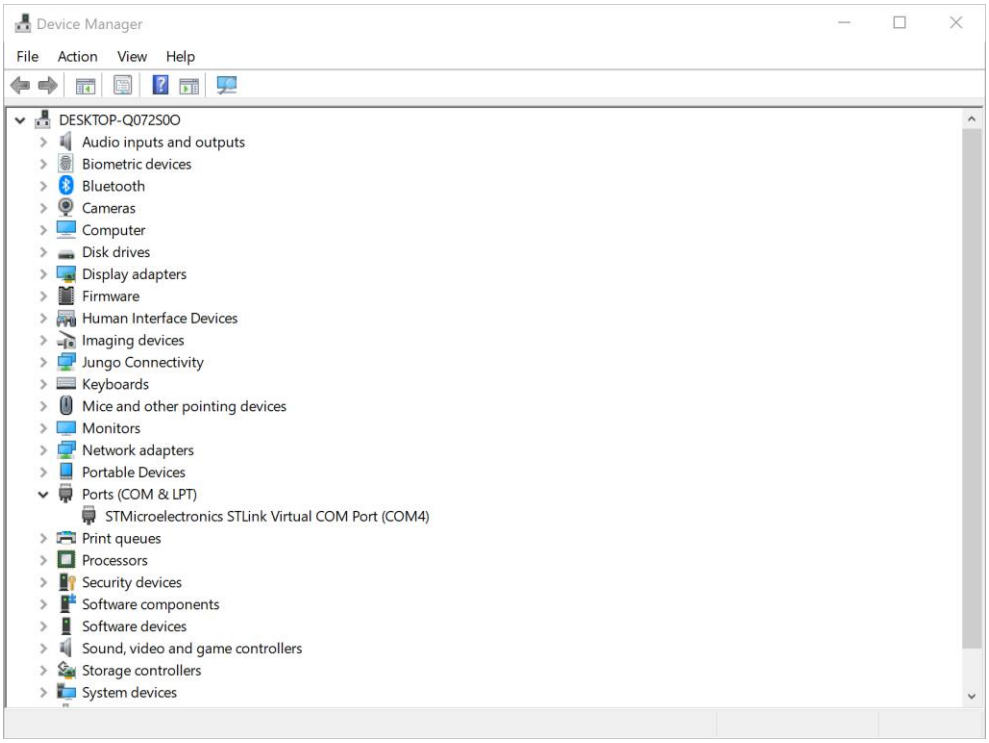


Select ***Project > Build*** Project or ***Build*** on the toolbar.[i]

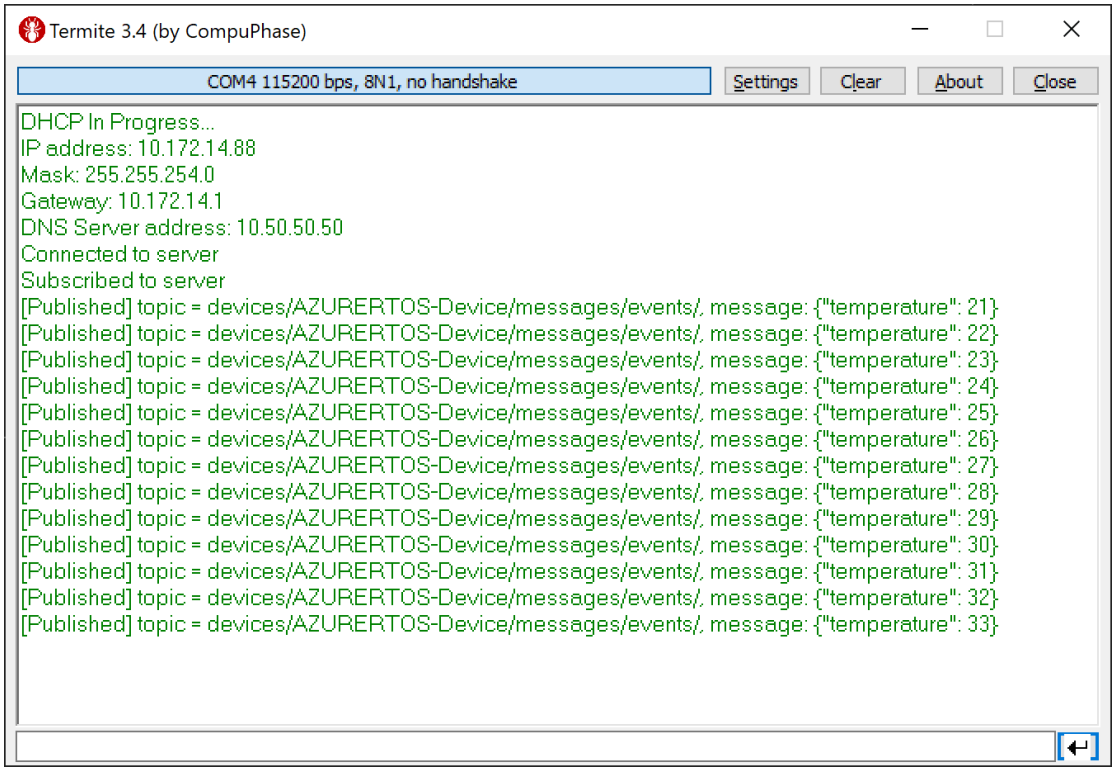
Step 4: Download and run the project

Select ***Run > Debug (F11)*** or ***Debug*** on the toolbar to download the program and run it. Then select ***Resume***.

Verify the serial port in your OS's device manager. It should show up as a COM port.

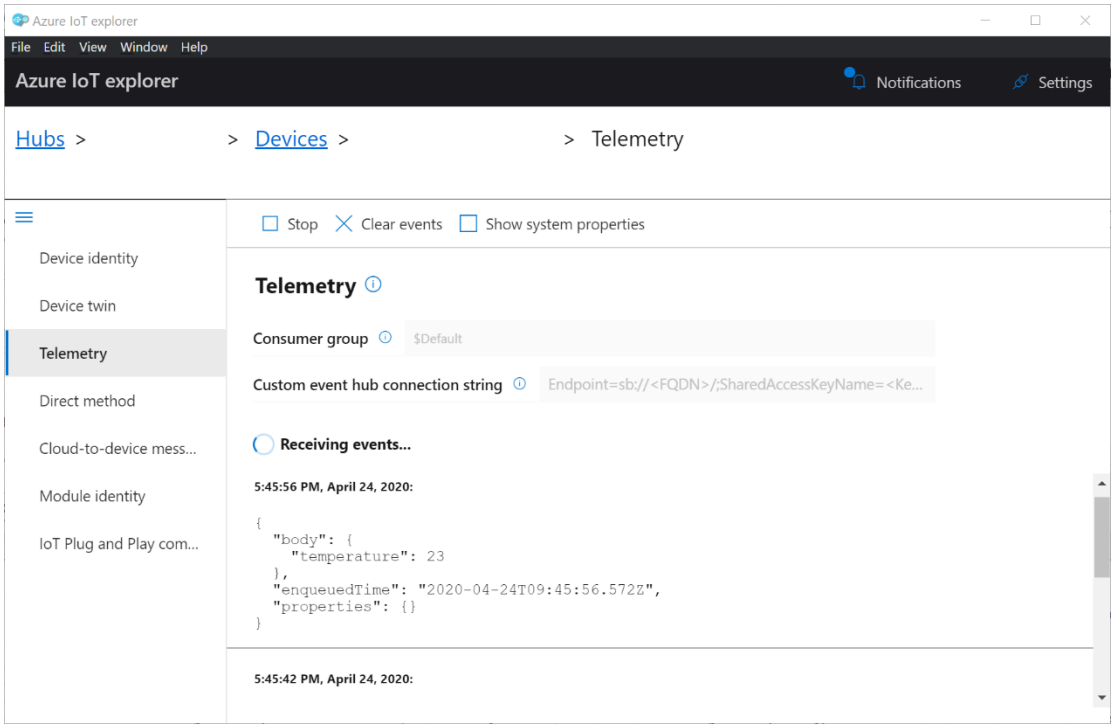


Open your favorite serial terminal program such as Termite and connect to the COM port discovered above. As the project runs, the demo prints out status information to the terminal output window. The demo also publishes the message to IoT Hub every five seconds. Check the terminal output to verify that messages have been successfully sent to the Azure IoT hub.



Step 6: Monitor telemetry data

In the Azure IoT Explorer, select the device you just created and select the ***Telemetry*** tab. Then select ***Start*** to view the messages published by the IoT Device.



Step 7: Manually send messages to the device.

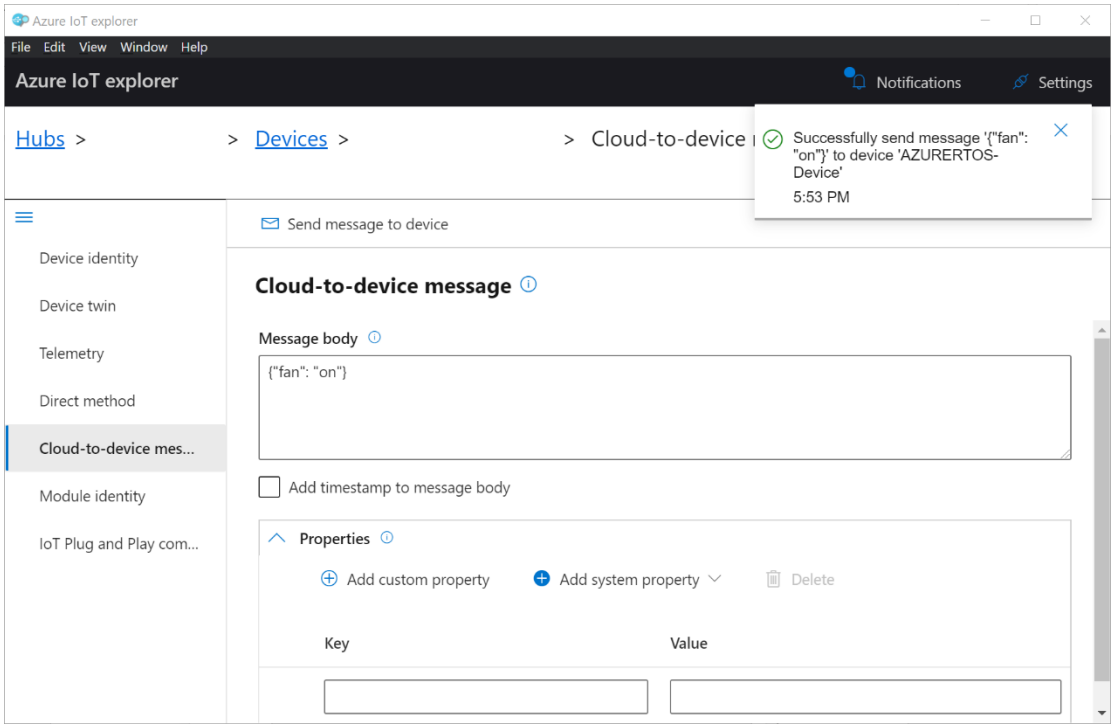
In the Azure IoT Explorer, switch to the ***Cloud-to-device message*** tab, and user can send commands to the IoT device.

In this demo, the following messages are defined:
{"fan": "on"}
{"fan": "off"}

Enter the message in the ***Message body*** and select ***Send message to device***.

User can send TurnFanOn message to Device to turn fan on. Device receives this message (to turn on fan), and in response decreases the temperature by 1 till the temperature reaches the minimum value of 0.

User can send TurnFanOff message to Device to turn fan off. Device receives this message (to turn off fan), and in response increases the temperature by 1 till the temperature reaches the maximum value of 40.



Next steps:

To learn more about Azure RTOS and how it works with Azure IoT, view https://azure.com/rtos.