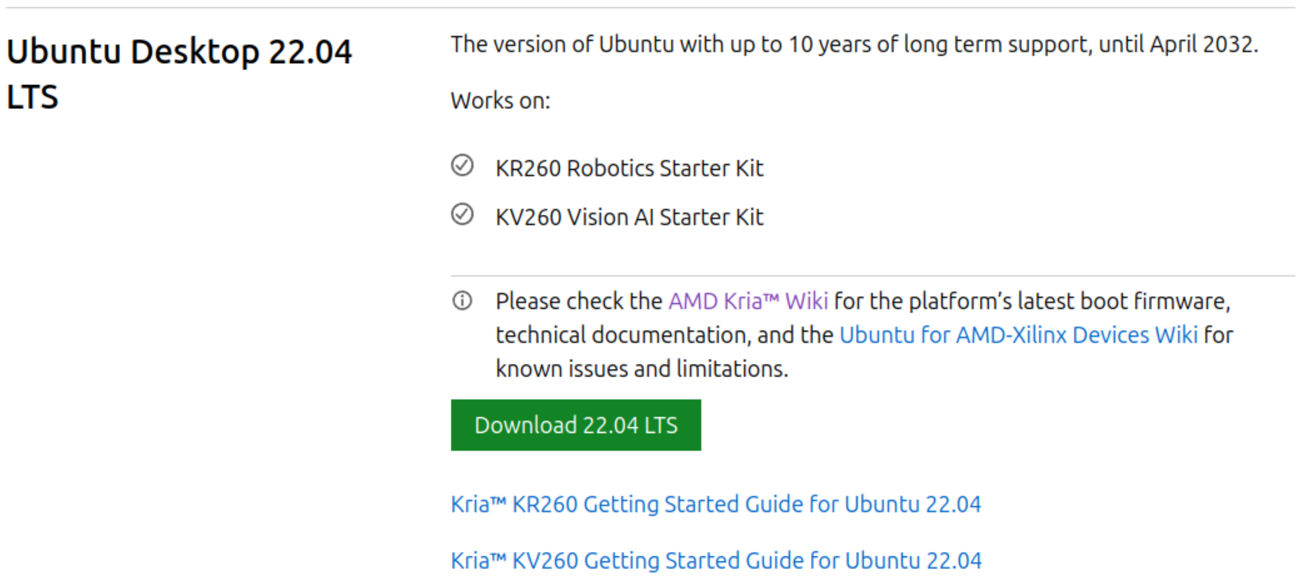
2023.05.09/v0.1 Sanam Shakya

KR260 to Azure IoT

Preparing Ubuntu 22.04 OS for KRIA KR260 board

Download the Ubuntu 22.04 image from the [download link]


Ubuntu Desktop 22.04 LTS

The version of Ubuntu with up to 10 years of long term support, until April 2032.

Works on:

- KR260 Robotics Starter Kit
- KV260 Vision AI Starter Kit

Please check the AMD Kria™ Wiki for the platform's latest boot firmware, technical documentation, and the Ubuntu for AMD-Xilinx Devices Wiki for known issues and limitations.

Download 22.04 LTS

Kria™ KR260 Getting Started Guide for Ubuntu 22.04

Kria™ KV260 Getting Started Guide for Ubuntu 22.04

Next, prepare the SD card with the above downloaded Ubuntu image using burning tools like Balena Etcher.

Now boot the KR260 with the SD card with Ethernet and USB to Serial cable connected to board. We will be using Serial console for initial access and debugging and Ethernet network for accessing through SSH and KR260 connected to the internet.

For initial login here are the Login Details:
Username : ubuntu

Password: ubuntu
This will ask to change the password. So update the password and login the system.
After successful login, one can access the KR260 device console.

Installing hardware overlay

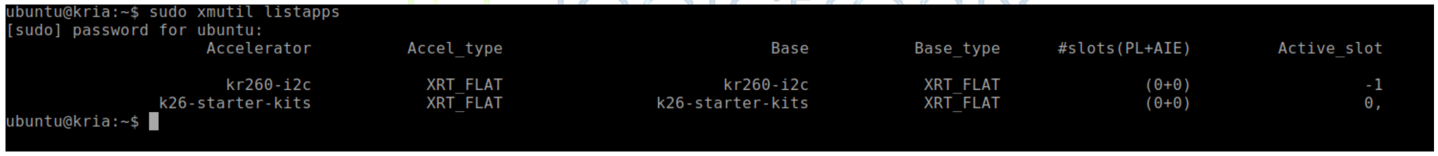
Get the KR260 firmware folder. It contains:

- kr260_i2c.bit.bin
- kr260_i2c.dtbo
- shell.json

Copy these file to the KR260 board. For firmware to be loaded using xmutil (FPGA manager), one has to copy these file at "/lib/firmware/xilinx".
For this create the folder at "kr260-i2c" at "/lib/firmware/xilinx" and copy the files in "kr260-i2c" folder.

```
cd /lib/firmware/xilinx
sudo mkdir kr260-i2c
sudo cp <kr260-firmware directory>/krc260_i2c* ./
sudo cp <kr260-firmware directory>/shell.json ./
```

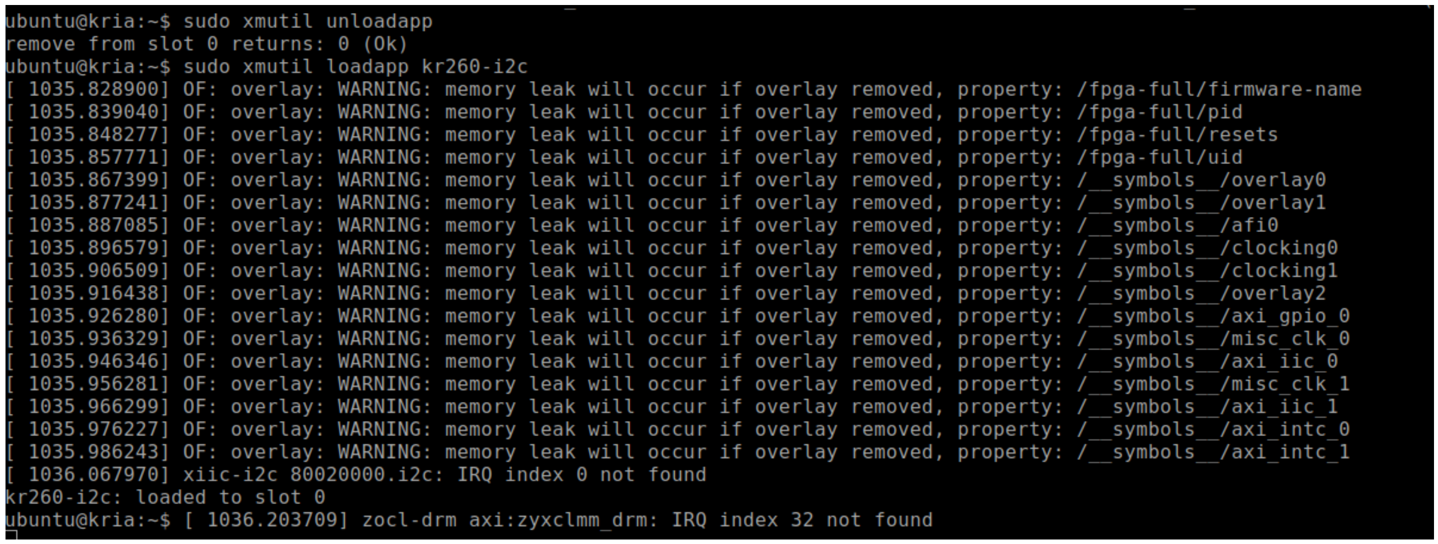
Next, check the available fpga firmware using `xmutil listapps` command. `kr260-i2c` will be available in the list.



```
ubuntu@kria:~$ sudo xmutil listapps
[sudo] password for ubuntu:
                      Accelerator          Accel_type                    Base          Base_type      #slots(PL+AIE)         Active_slot

                        kr260-i2c            XRT_FLAT               kr260-i2c           XRT_FLAT               (0+0)                  -1
                 k26-starter-kits            XRT_FLAT        k26-starter-kits           XRT_FLAT               (0+0)                  0,
ubuntu@kria:~$
```

Next load the `kr260-i2c` firmware, which contains necessary hardwares(gpio) and interfaces. In our Greengrass Demo we will be using these gpio to trigger the publishing data to AWS Greengrass IoT cloud server and also actuate GPIO on the message received from AWS cloud.

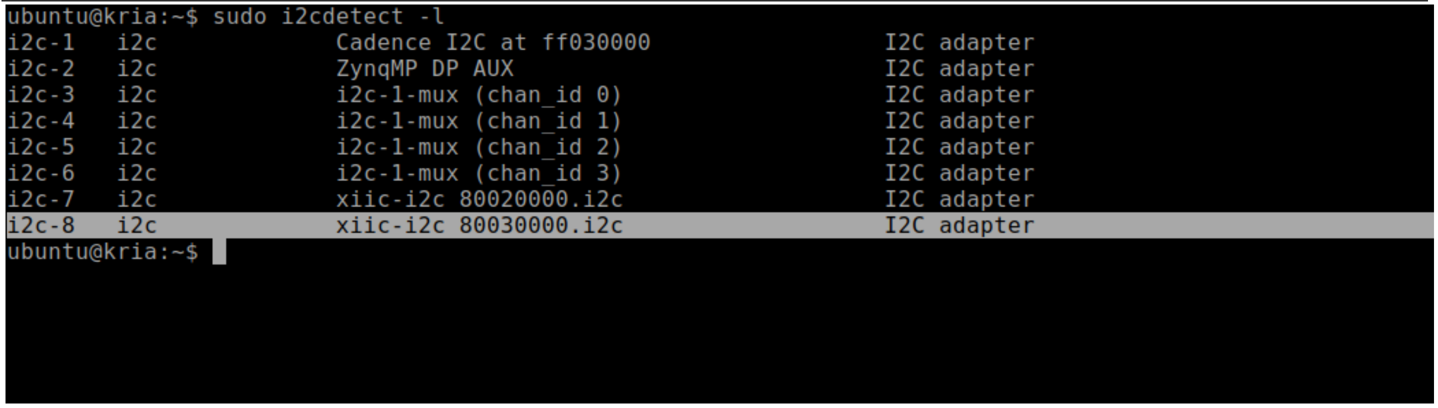
```
sudo xmutil unloadapp
sudo xmutil loadapp kr260-i2c
```



```
ubuntu@kria:~$ sudo xmutil unloadapp
remove from slot 0 returns: 0 (Ok)
ubuntu@kria:~$ sudo xmutil loadapp kr260-i2c
[ 1035.828900] OF: overlay: WARNING: memory leak will occur if overlay removed, property: /fpga-full/firmware-name
[ 1035.839040] OF: overlay: WARNING: memory leak will occur if overlay removed, property: /fpga-full/pid
[ 1035.848277] OF: overlay: WARNING: memory leak will occur if overlay removed, property: /fpga-full/resets
[ 1035.857771] OF: overlay: WARNING: memory leak will occur if overlay removed, property: /fpga-full/uid
[ 1035.867399] OF: overlay: WARNING: memory leak will occur if overlay removed, property: /__symbols__/overlay0
[ 1035.877241] OF: overlay: WARNING: memory leak will occur if overlay removed, property: /__symbols__/overlay1
[ 1035.887085] OF: overlay: WARNING: memory leak will occur if overlay removed, property: /__symbols__/afi0
[ 1035.896579] OF: overlay: WARNING: memory leak will occur if overlay removed, property: /__symbols__/clocking0
[ 1035.906509] OF: overlay: WARNING: memory leak will occur if overlay removed, property: /__symbols__/clocking1
[ 1035.916438] OF: overlay: WARNING: memory leak will occur if overlay removed, property: /__symbols__/overlay2
[ 1035.926280] OF: overlay: WARNING: memory leak will occur if overlay removed, property: /__symbols__/axi_gpio_0
[ 1035.936329] OF: overlay: WARNING: memory leak will occur if overlay removed, property: /__symbols__/misc_clk_0
[ 1035.946346] OF: overlay: WARNING: memory leak will occur if overlay removed, property: /__symbols__/axi_iic_0
[ 1035.956281] OF: overlay: WARNING: memory leak will occur if overlay removed, property: /__symbols__/misc_clk_1
[ 1035.966299] OF: overlay: WARNING: memory leak will occur if overlay removed, property: /__symbols__/axi_iic_1
[ 1035.976227] OF: overlay: WARNING: memory leak will occur if overlay removed, property: /__symbols__/axi_intc_0
[ 1035.986243] OF: overlay: WARNING: memory leak will occur if overlay removed, property: /__symbols__/axi_intc_1
[ 1036.067970] xiic-i2c 80020000.i2c: IRQ index 0 not found
kr260-i2c: loaded to slot 0
ubuntu@kria:~$ [ 1036.203709] zocl-drm axi:zyxclmm_drm: IRQ index 32 not found
```

Now, check the available i2c channels available in the system using `i2cdetect` i2c utility tool.

```
sudo i2cdetect -l
```



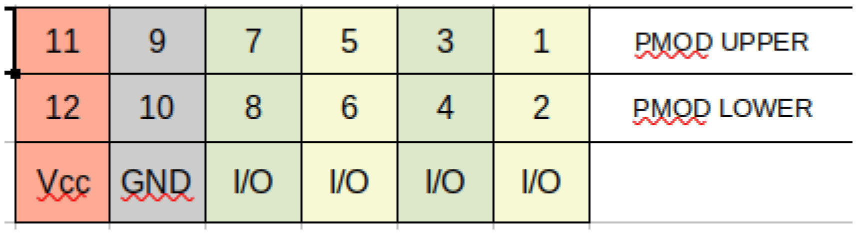
```
ubuntu@kria:~$ sudo i2cdetect -l
i2c-1   i2c             Cadence I2C at ff030000                 I2C adapter
i2c-2   i2c             ZynqMP DP AUX                           I2C adapter
i2c-3   i2c             i2c-1-mux (chan_id 0)                   I2C adapter
i2c-4   i2c             i2c-1-mux (chan_id 1)                   I2C adapter
i2c-5   i2c             i2c-1-mux (chan_id 2)                   I2C adapter
i2c-6   i2c             i2c-1-mux (chan_id 3)                   I2C adapter
i2c-7   i2c             xiic-i2c 80020000.i2c                   I2C adapter
i2c-8   i2c             xiic-i2c 80030000.i2c                   I2C adapter
ubuntu@kria:~$
```

`i2c-8` channel will be used to connect to BMP180 sensor.

Connecting BMP180 to AXI I2C Bus

Connect BMP180 sensors, Vcc, GND, I2C SDA and I2C SCLK pins to PMOD as explained below:
PMOD1-> 6 - I2C SCLK
PMOD1-> 8 - I2C SDA
PMOD1-> GND - BMP180 GND
PMOD1->Vcc - BMP180 Vcc

Schematic for LED and Switch Connection

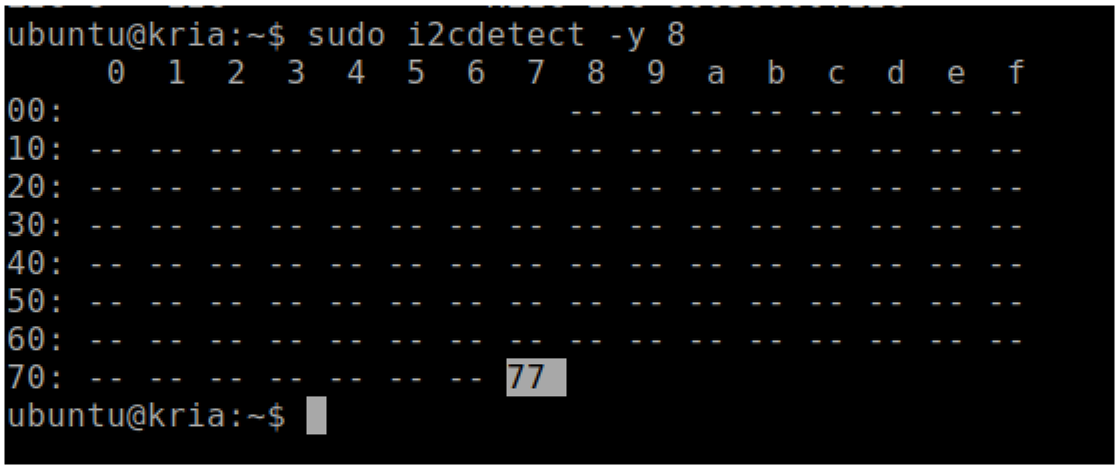


11	9	7	5	3	1	PMOD UPPER
12	10	8	6	4	2	PMOD LOWER
Vcc	GND	I/O	I/O	I/O	I/O	

PMOD port numbering

After connecting BMP180 sensor to KR260 PMOD port, use i2c utility tools to scan for the available devices in i2c-8 channel.

```
sudo i2cdetect -y 8
```



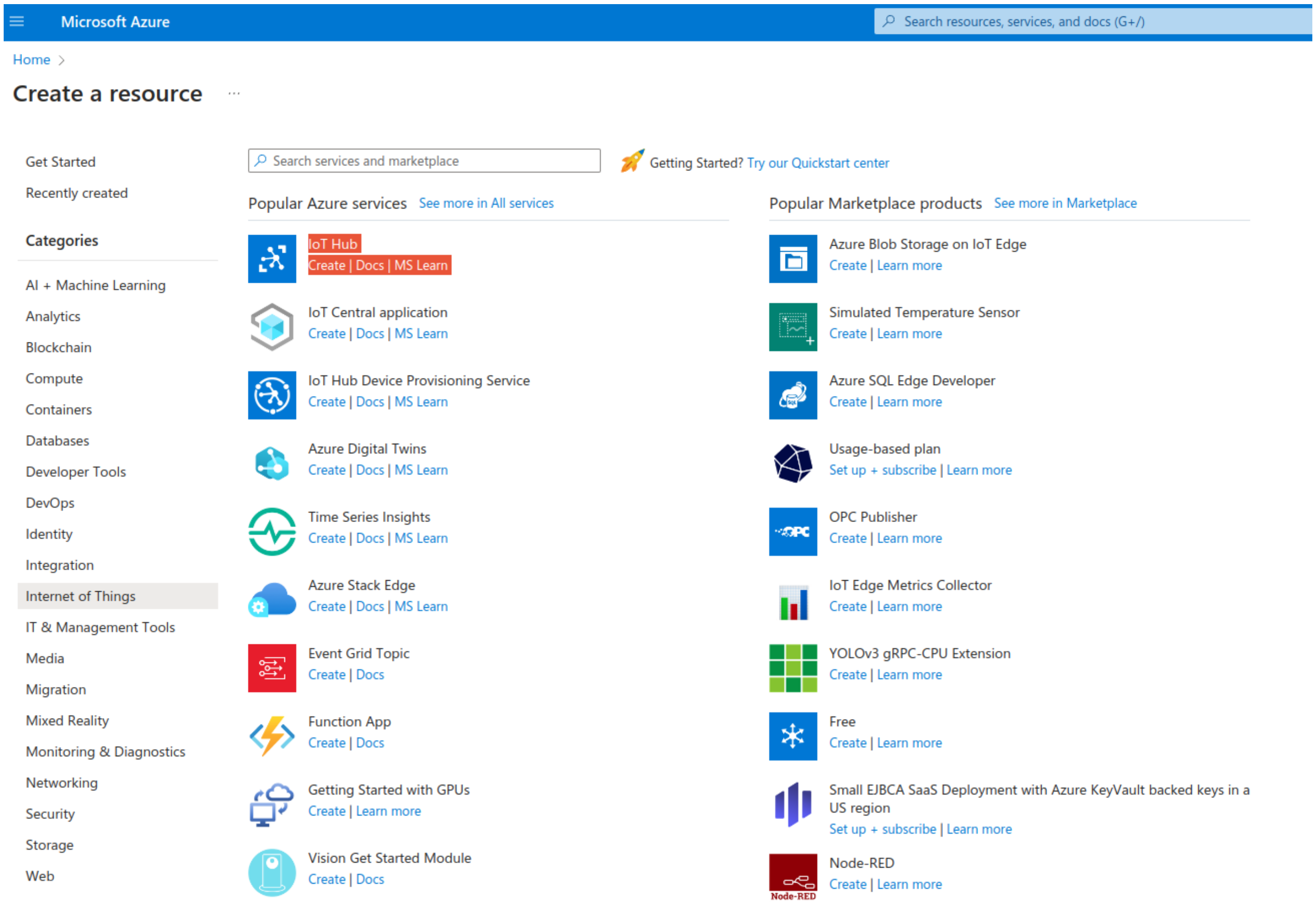
```
ubuntu@kria:~$ sudo i2cdetect -y 8
     0  1  2  3  4  5  6  7  8  9  a  b  c  d  e  f
00:                         -- -- -- -- -- -- -- --
10: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
20: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
30: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
40: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
50: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
60: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
70: -- -- -- -- -- -- -- 77
ubuntu@kria:~$
```

In i2c scan, we find a device is available at address '77', which corresponds to BMP180 i2c sensor.
Next we will add the component for publishing BMP180 sensor data to the AWS IoT cloud.

Follow these steps after installing AWS greengrass core device in the KR260 board as mentioned in Kria connect to AWS IoT - GPIO document.

Create IoT Hub in Azure Portal:

- Go to Azure portal " *https://portal.azure.com* ".
- Create a resource >> IoT Hub.


Microsoft Azure
Search resources, services, and docs (G+/)
Home >
Create a resource

Get Started
Recently created

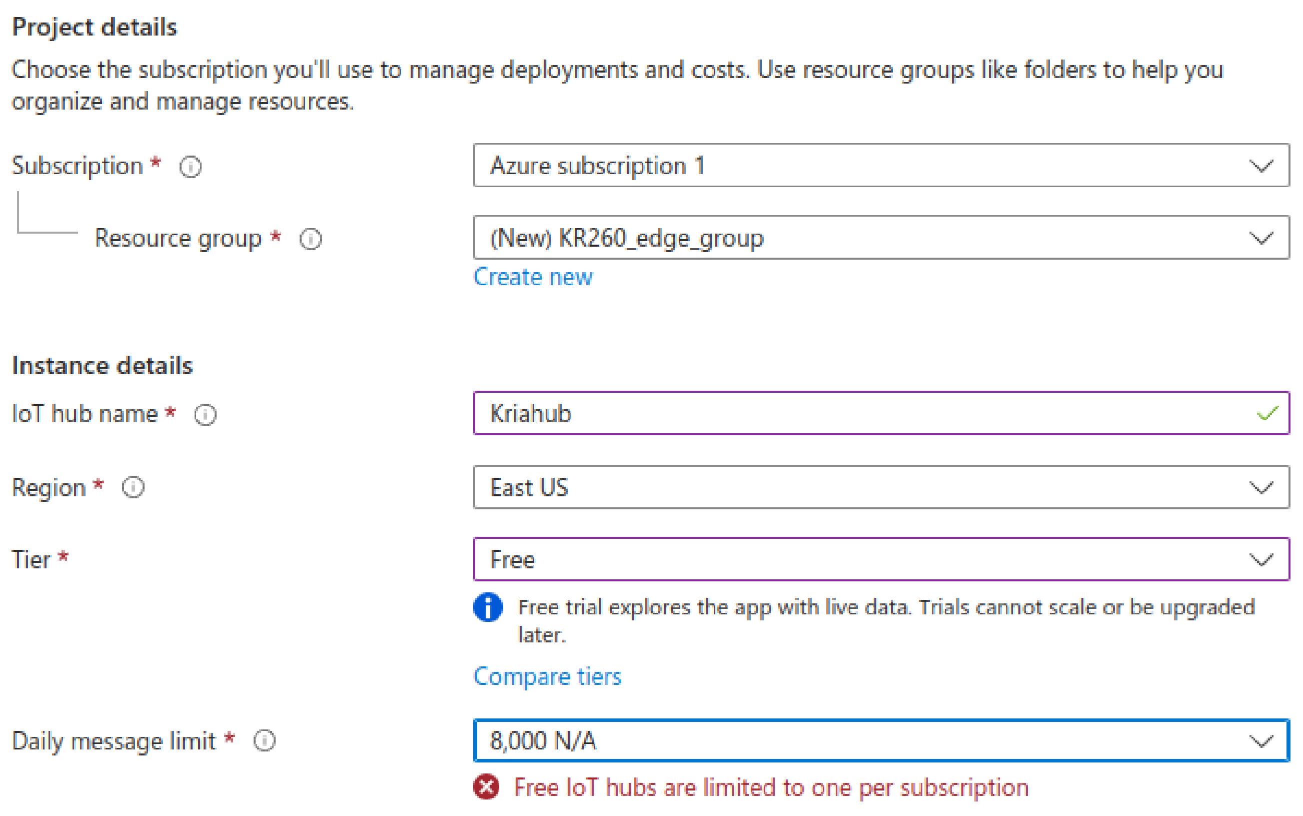
Categories
AI + Machine Learning
Analytics
Blockchain
Compute
Containers
Databases
Developer Tools
DevOps
Identity
Integration
Internet of Things
IT & Management Tools
Media
Migration
Mixed Reality
Monitoring & Diagnostics
Networking
Security
Storage
Web

Search services and marketplace
Getting Started? Try our Quickstart center

Popular Azure services — See more in All services
- IoT Hub — Create | Docs | MS Learn
- IoT Central application — Create | Docs | MS Learn
- IoT Hub Device Provisioning Service — Create | Docs | MS Learn
- Azure Digital Twins — Create | Docs | MS Learn
- Time Series Insights — Create | Docs | MS Learn
- Azure Stack Edge — Create | Docs | MS Learn
- Event Grid Topic — Create | Docs
- Function App — Create | Docs
- Getting Started with GPUs — Create | Learn more
- Vision Get Started Module — Create | Docs

Popular Marketplace products — See more in Marketplace
- Azure Blob Storage on IoT Edge — Create | Learn more
- Simulated Temperature Sensor — Create | Learn more
- Azure SQL Edge Developer — Create | Learn more
- Usage-based plan — Set up + subscribe | Learn more
- OPC Publisher — Create | Learn more
- IoT Edge Metrics Collector — Create | Learn more
- YOLOv3 gRPC-CPU Extension — Create | Learn more
- Free — Create | Learn more
- Small EJBCA SaaS Deployment with Azure KeyVault backed keys in a US region — Set up + subscribe | Learn more
- Node-RED — Create | Learn more

Next, create one IoT Hub Service and fill in the necessary details



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 *	(New) KR260_edge_group
	Create new

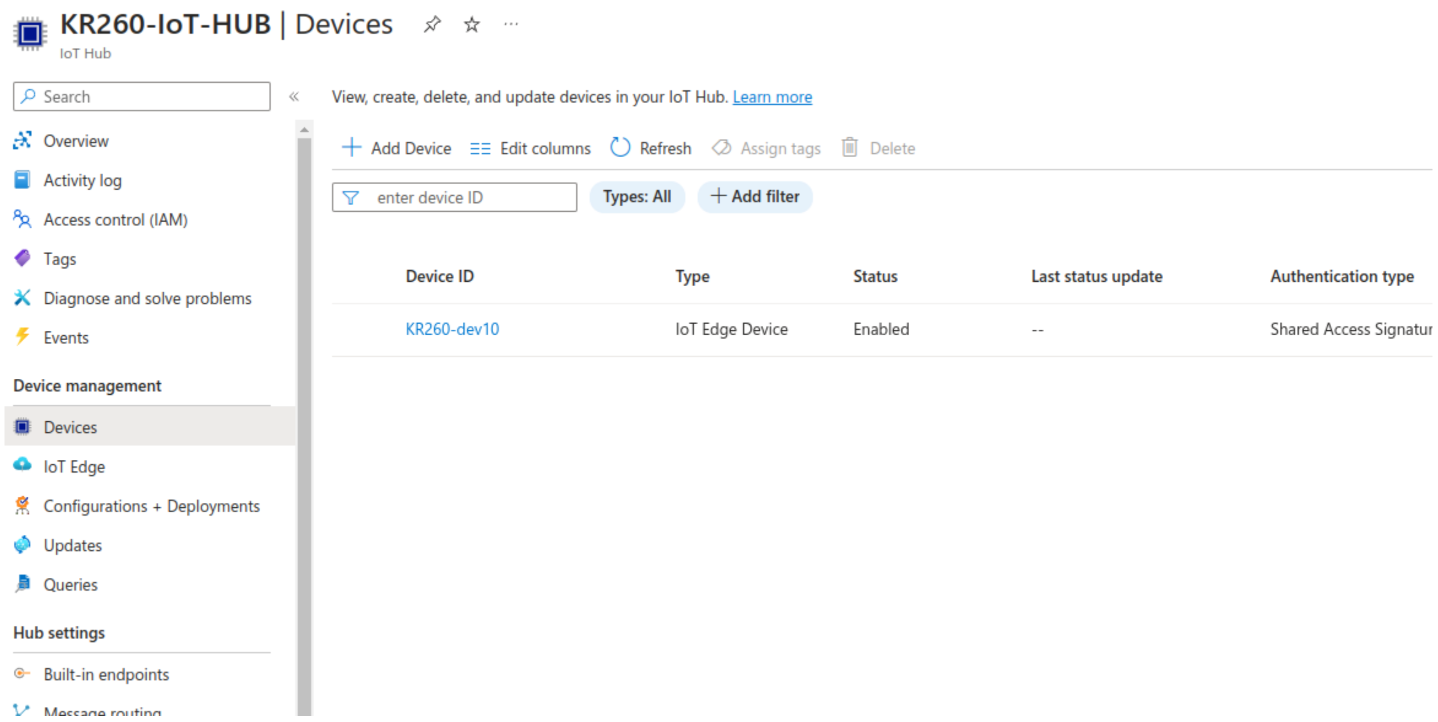
Instance details

IoT hub name *	Kriahub
Region *	East US
Tier *	Free
	Free trial explores the app with live data. Trials cannot scale or be upgraded later.
	Compare tiers
Daily message limit *	8,000 N/A
	Free IoT hubs are limited to one per subscription

- Click on Review+ Create button to create the Azure IoT Hub.
- Next, create a device where you can actually receive some data from the hardware.

Create an IoT Device

Go to the IoT Device and click on new, and give the device ID

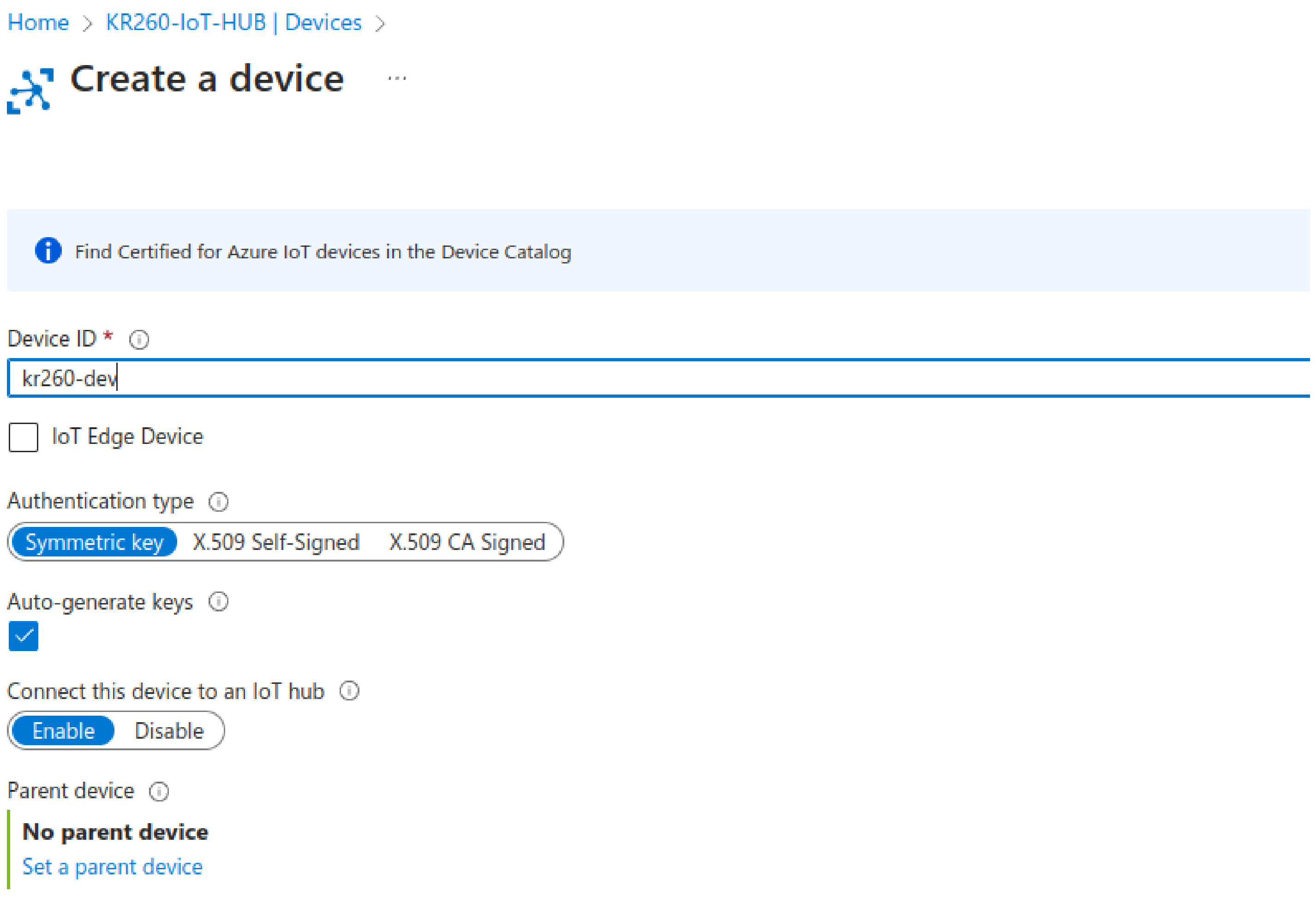


KR260-IoT-HUB | Devices
IoT Hub

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

Device ID	Type	Status	Last status update	Authentication type
KR260-dev10	IoT Edge Device	Enabled	--	Shared Access Signatur

Next Click on +Add Device to add the device to the IoT Hub. This will open the form for creating the device.


Home > KR260-IoT-HUB | Devices >

Create a device

Find Certified for Azure IoT devices in the Device Catalog

Device ID *

kr260-dev

☐ IoT Edge Device

Authentication type

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

Auto-generate keys

✓

Connect this device to an IoT hub

[✓ Enable] Disable

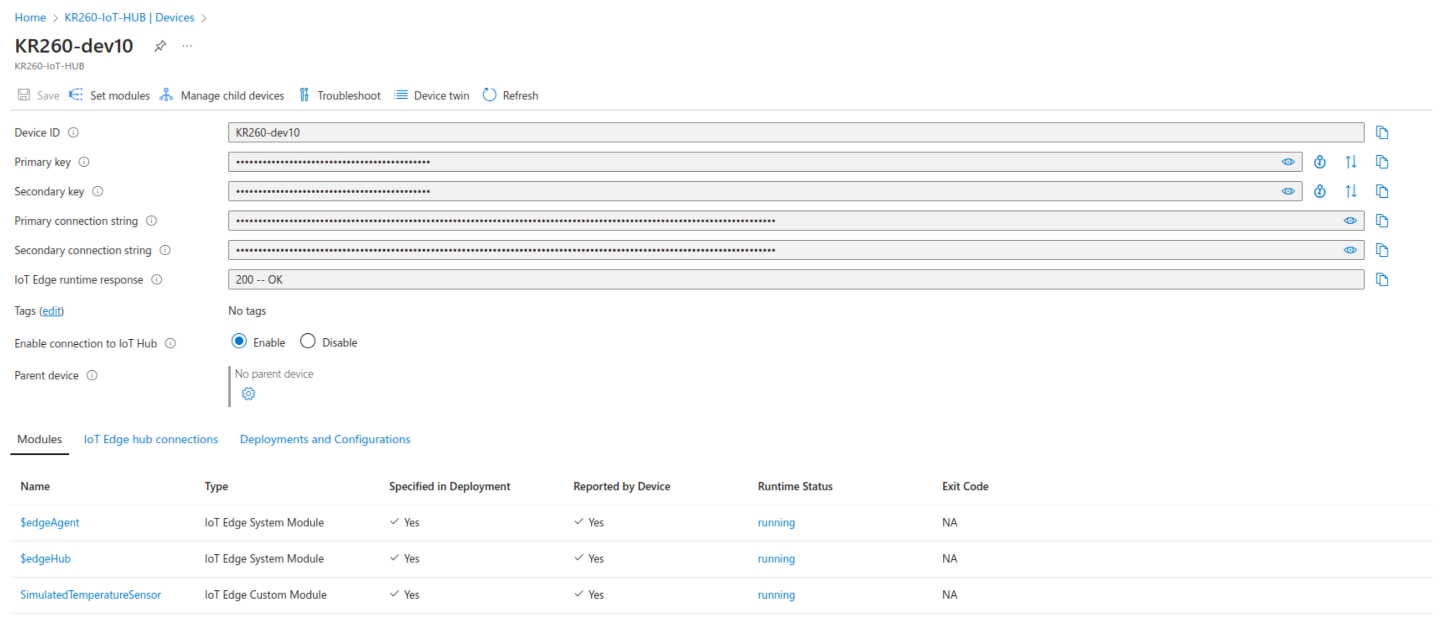
Parent device

No parent device

Set a parent device

After this device will be available in the IoT hub Device list.

Next, look into device information for getting the keys and connection string.



Home > KR260-IoT-HUB | Devices >

KR260-dev10

KR260-IoT-HUB

Save | Set modules | Manage child devices | Troubleshoot | Device twin | Refresh

Device ID	KR260-dev10
Primary key	••
Secondary key	••
Primary connection string	••
Secondary connection string	••
IoT Edge runtime response	200 -- OK
Tags (edit)	No tags
Enable connection to IoT Hub	[✓ Enable] Disable
Parent device	No parent device

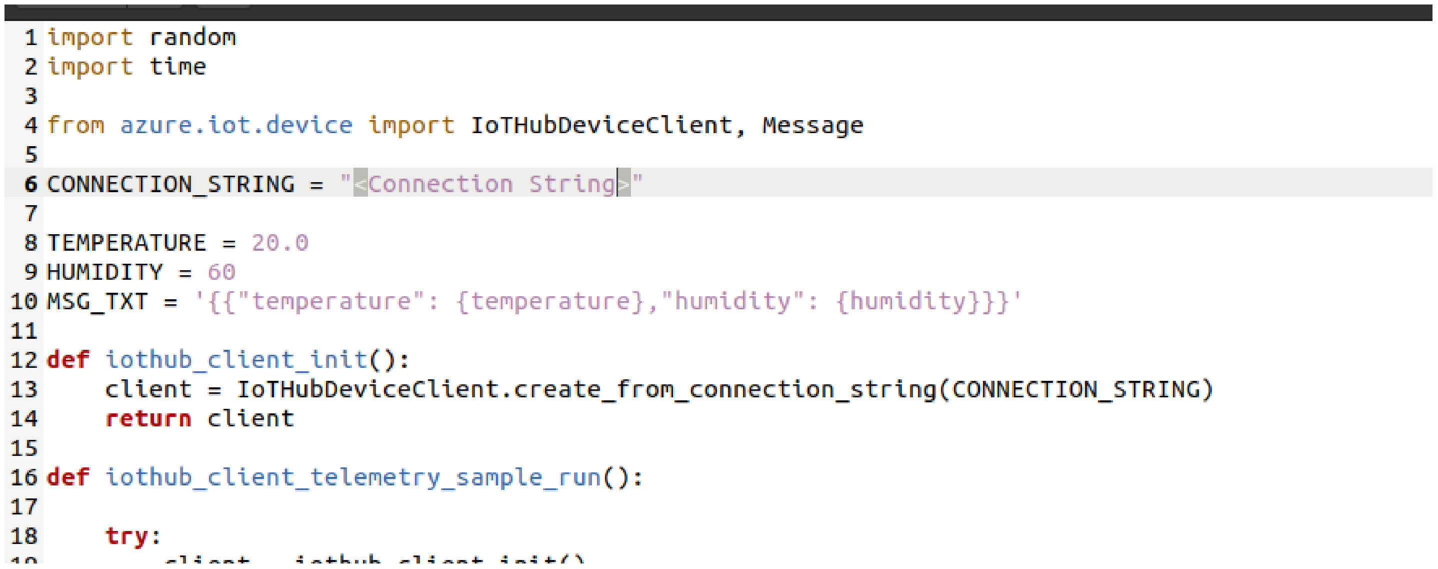
Modules | IoT Edge hub connections | Deployments and Configurations

Name	Type	Specified in Deployment	Reported by Device	Runtime Status	Exit Code
$edgeAgent	IoT Edge System Module	✓ Yes	✓ Yes	running	NA
$edgeHub	IoT Edge System Module	✓ Yes	✓ Yes	running	NA
SimulatedTemperatureSensor	IoT Edge Custom Module	✓ Yes	✓ Yes	running	NA

Copy the "Primary Connection String" which will be used in the python application for sending the sensor data to IoT hub.

Adding python application in KRIA

Copy the simulated_sensor.py example code in the previously Ubuntu installed KR260 board.
Next update the "CONNECTION STRING" with the above Primary Connection string.

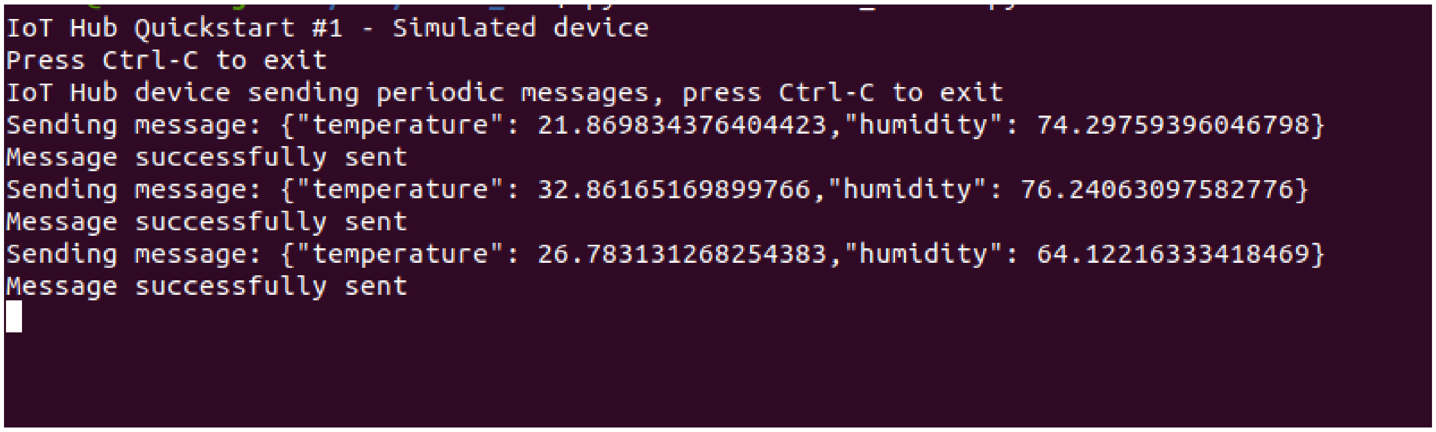


```
import random
import time

from azure.iot.device import IoTHubDeviceClient, Message

CONNECTION_STRING = "<Connection String>"

TEMPERATURE = 20.0
HUMIDITY = 60
MSG_TXT = '{{"temperature": {temperature},"humidity": {humidity}}}'

def iothub_client_init():
    client = IoTHubDeviceClient.create_from_connection_string(CONNECTION_STRING)
    return client

def iothub_client_telemetry_sample_run():

    try:
```

Then run the simulated application in console:

```
python3 simulated_sensor.py
```

Here is the console log after successful message send to Azure IoT hub.

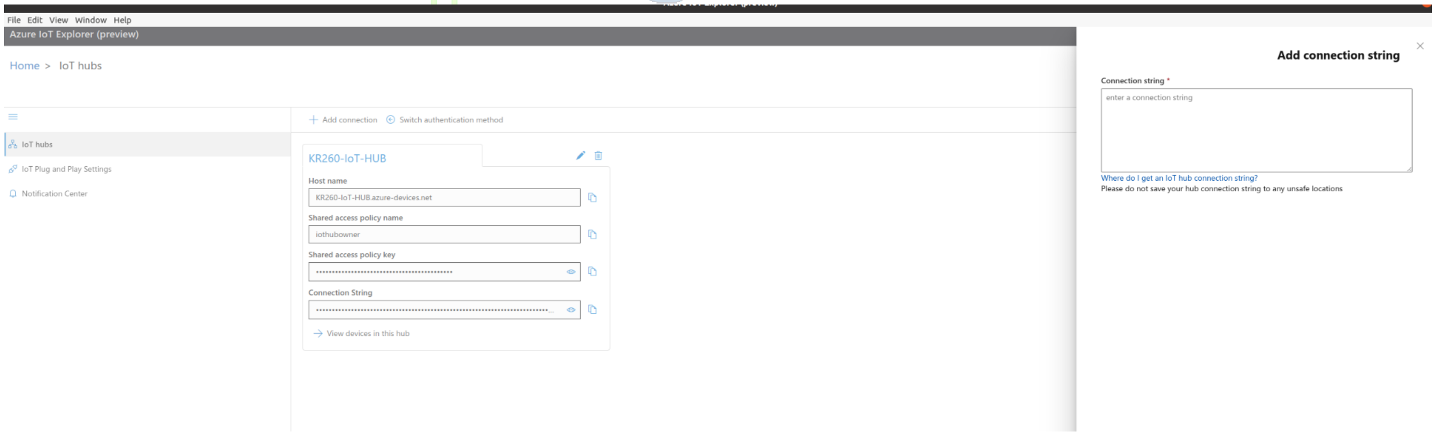


```
IoT Hub Quickstart #1 - Simulated device
Press Ctrl-C to exit
IoT Hub device sending periodic messages, press Ctrl-C to exit
Sending message: {"temperature": 21.869834376404423,"humidity": 74.29759396046798}
Message successfully sent
Sending message: {"temperature": 32.86165169899766,"humidity": 76.24063097582776}
Message successfully sent
Sending message: {"temperature": 26.783131268254383,"humidity": 64.12216333418469}
Message successfully sent
```

Viewing message in Host Machine

For viewing the message published by Azure IoT Device in KR260, one can use Azure IoT explorer available in following link:
https://github.com/Azure/azure-iot-explorer/releases

In IoT HUbs page of the application, in +Add connection copy the connection string for the IoT hub and save the configs:


File Edit View Window Help
Azure IoT Explorer (preview)
Home > IoT hubs
IoT hubs
IoT Plug and Play Settings
Notification Center
Add connection
Switch authentication method
View devices in this hub
Add connection string

KR260-IoT-HUB

Host name: KR260-IoT-HUB.azure-devices.net

Shared access policy name: iothubowner

Shared access policy key: ••

Connection String: ••...

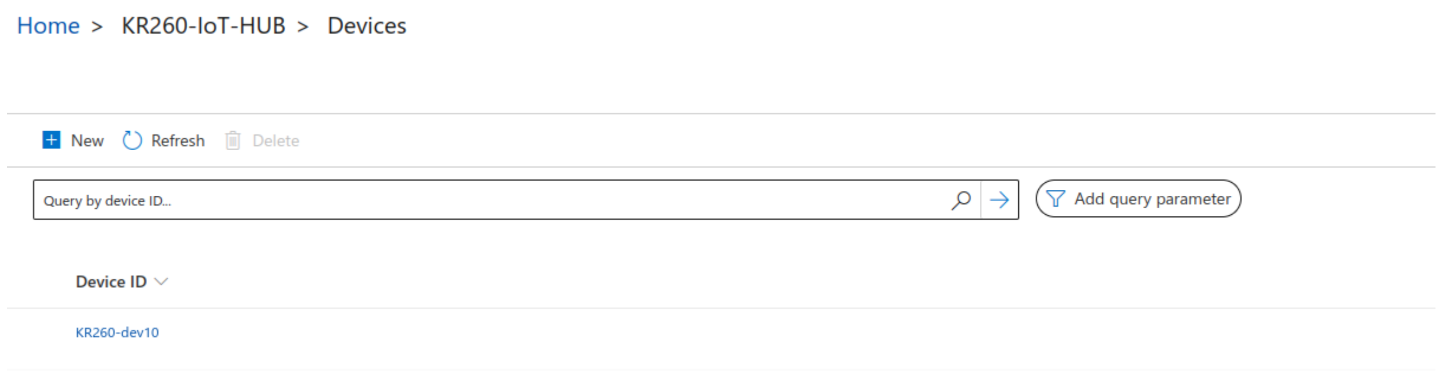
Connection string *

enter a connection string

Where do I get an IoT hub connection string?

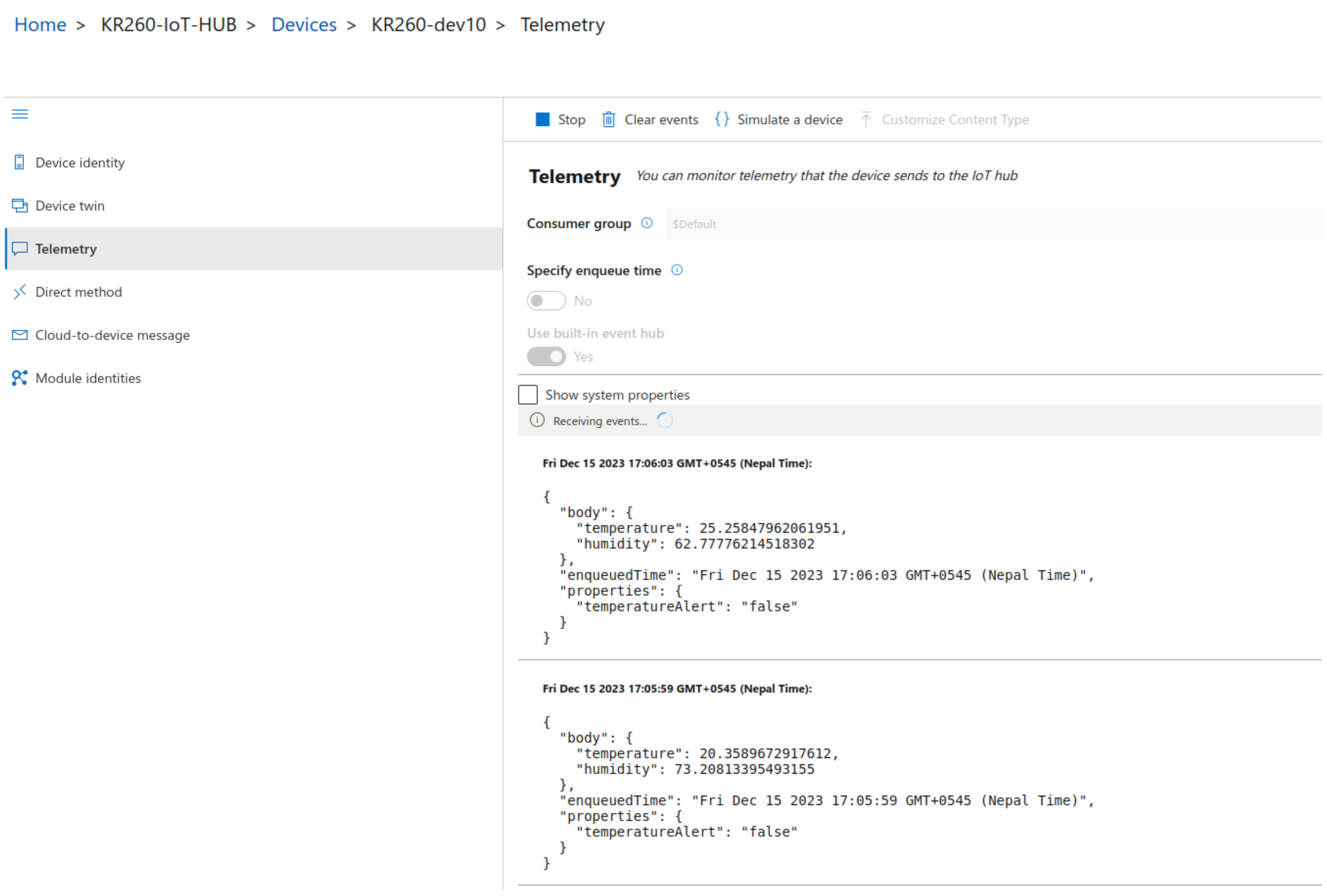
Please do not save your hub connection string to any unsafe locations

One can find the corresponding device list in the IoT HuB page of Azure IoT explorer application.



Just click onto the device to view the device information and also the message send by python application running in the KR260 board.

For viewing the message send to device, go to Telemetry and click the >Start button.
After this one can view the message send to the device.



Telemetry *You can monitor telemetry that the device sends to the IoT hub*

Consumer group: $Default

Specify enqueue time: No

Use built-in event hub: Yes

Show system properties

Receiving events...

Fri Dec 15 2023 17:06:03 GMT+0545 (Nepal Time):

```
{
  "body": {
    "temperature": 25.25847962061951,
    "humidity": 62.77776214518302
  },
  "enqueuedTime": "Fri Dec 15 2023 17:06:03 GMT+0545 (Nepal Time)",
  "properties": {
    "temperatureAlert": "false"
  }
}
```

Fri Dec 15 2023 17:05:59 GMT+0545 (Nepal Time):

```
{
  "body": {
    "temperature": 20.3589672917612,
    "humidity": 73.20813395493155
  },
  "enqueuedTime": "Fri Dec 15 2023 17:05:59 GMT+0545 (Nepal Time)",
  "properties": {
    "temperatureAlert": "false"
  }
}
```