

AMD KR260 Getting Started Guide

for AWS IoT Greengrass V2

Table of Contents

<i>1. Document information</i>	1
<i>2. Overview</i>	2
<i>3. Hardware description</i>	3
<i>4. Set up your development environment</i>	3
<i>5. Set up device hardware</i>	3
<i>6. About AWS IoT Greengrass</i>	6
<i>7. Greengrass prerequisites</i>	6
<i>8. Install AWS IoT Greengrass</i>	7
<i>9. Installing the components</i>	13
<i>10. Debugging</i>	15
<i>11. Troubleshooting</i>	17

1. Document information

Note that all instructions in this document have been written assuming a KR260 Ubuntu Edge device.

1.1 Document revision history

30-Dec-2023	Initial Draft
27-Feb-2024	Updated Document Format

1.2 Applicable operating systems for this guide

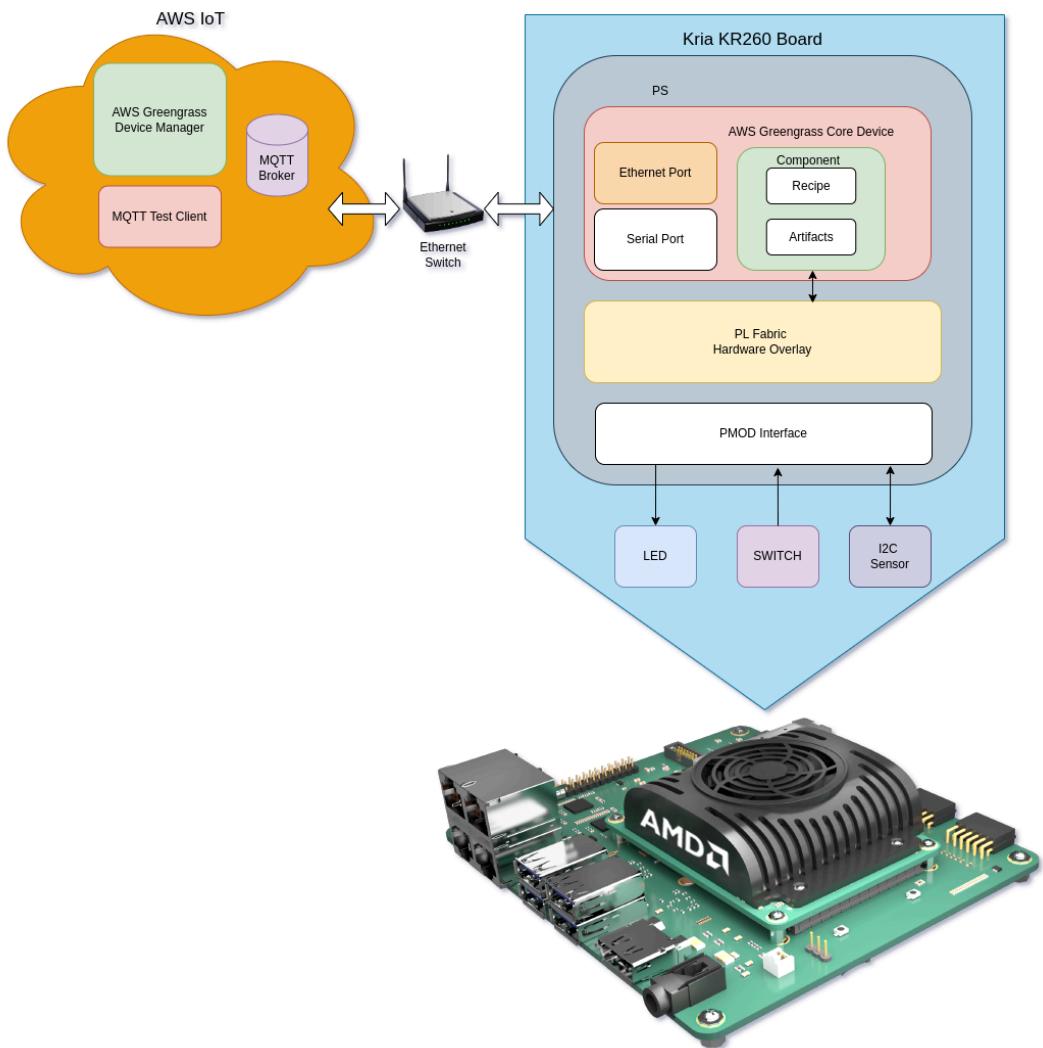
For this guide host machine with any (Windows, Linux) OS can be used and for Edge Device Ubuntu 22.04 will be installed.

2. Overview

AMD Kria SoM is Zynq® UltraScale+™ MPSoC architecture based FPGA platform, which is suitable for vision AI solutions for edge applications.

AMD KR260 is Kria SoM based Robotics Starter Kit, which is suitable for Robotics, Industrial Control and Machine Learning inference based applications. By using PMOD and RPi based header, we can interface various sensors and publish data to the cloud platforms.

Following diagram shows the software and hardware architecture used in this getting started guide. Kria KR260 board consists of PL Fabric(FPGA) hardware overlay for interfacing LED, switch and I2C sensor. Further it runs AWS Greengrass Core Device Application which publish and subscribe message topics for actuating LED and monitoring sensors and switches. From AWS IoT MQTT Test Client KR260 LED will be controlled through subscribed topic and also publish Switch pressed event to AWS IoT cloud.



3. Hardware description

3.1 Datasheet

Refer to the Kria K26 SOM Product Brief and SOM Data Sheet at product documentation at the [K26 Documentation page](#).

For a view of the hardware portfolio, refer to this [Portfolio Comparison](#).

3.2 Standard kit contents

Details regarding the KR260 kit contents are provided [here](#).

Refer to the section What's Inside the Box in the online documentation [here](#).

3.3 User provided items

- Host PC

3.4 3rd party purchasable items

For “Hello world” component test in section 9.1, no additional hardware is needed.

While user has to procure following hardware for “MQTT-GPIO” component test in section 9.2:

- Generic Light Emitting Diode(LED)
- Push button switch
- Resistors (4.7 K and 100 ohms)
- Bread Board and connecting jumper wires

Above hardware is readily available at electronic hardware shops like [Amazon](#).

3.5 Additional References

Getting started guide for KR260 board is available [here](#).

Refer to the all the resources related to the Kria™ KR260 Robotics Starter Kit on the [GitHub page](#)

Read the [Kria KR260 Robotics Starter Kit User Guide](#) and [Data sheet](#)

4. Set up your development environment

4.1 Tools installation (IDEs, Toolchains, SDKs)

Not Applicable

5. Set up device hardware

The KR260 board boots off an SD card. To create this SD card, refer to the instructions at [Setting up the SD card image](#).

The instructions to set up and connect the board are available [here](#).

Follow the instructions [here](#) to boot and monitor your board.

5.1 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__/afio
[ 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 to access GPIO in user application, we will be using `gpiod` library.

5.2 Installing gpiod packages

GPIOD packages are required to access the GPIO channels. It also provides python binding for accessing GPIO in python programming. Install the package using apt-get:

```
sudo apt-get install gpiod python3-libgpiod
```

Now we can check the available gpio using gpiod applications:

Using `gpiodetect` to get available gpio:

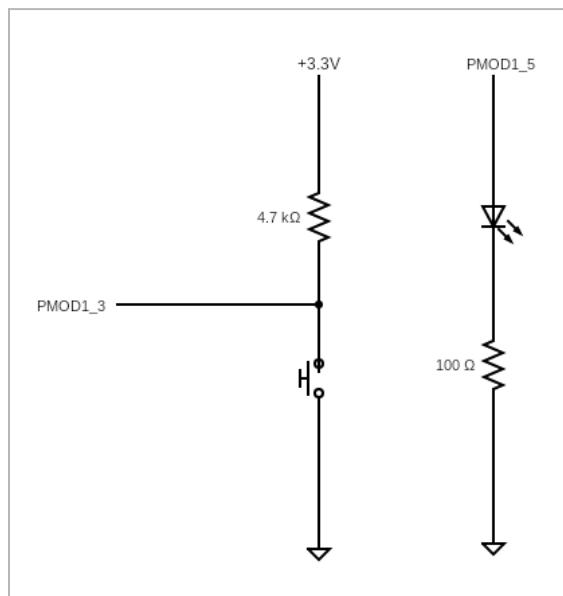
```
ubuntu@kria:~$ sudo gpiodetect
[gpiochip0 [firmware:zynqmp-firmware:gpio] (4 lines)
[gpiochip1 [zynqmp_gpio] (174 lines)
[gpiochip2 [slg7xl45106] (8 lines)
[gpiochip3 [80010000 gpio] (6 lines)
ubuntu@kria:~$
```

Here `gpiochip3` is the device corresponding to gpio in FPGA and it consists of 6 lines. Further these gpio lines are connected to PMOD 1 such that:

PMOD1-> 1 - gpiochip3 line 0 (Not in use)

PMOD1-> 3 - gpiochip3 line 1

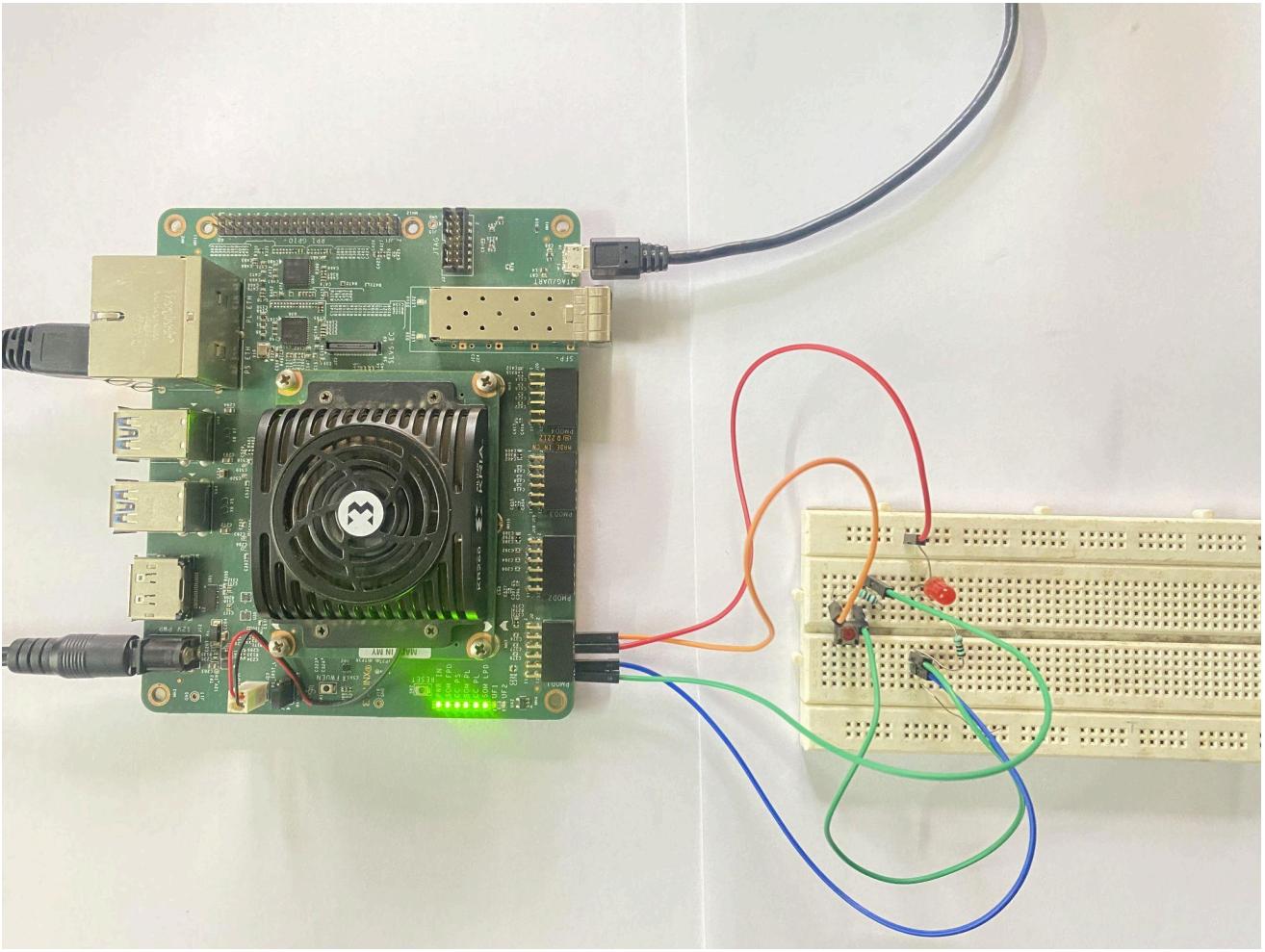
PMOD1-> 5 - gpiochip3 line 2



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



KR260 Board Connected to switch and LED through PMOD1

6. About AWS IoT Greengrass

To learn more about AWS IoT GreengrassV2, see [How AWS IoT Greengrass works](#) and [What's new in AWS IoT Greengrass Version 2](#).

7. Greengrass prerequisites

Refer to the online documentation detailing the [prerequisites](#) needed for AWS IoT Greengrass.
Follow the instructions in the following sections:

[Step 1: Set up an AWS account](#)

[Step 2: Set up your environment](#)

8. Install AWS IoT Greengrass

Refer to the instructions in the following steps:

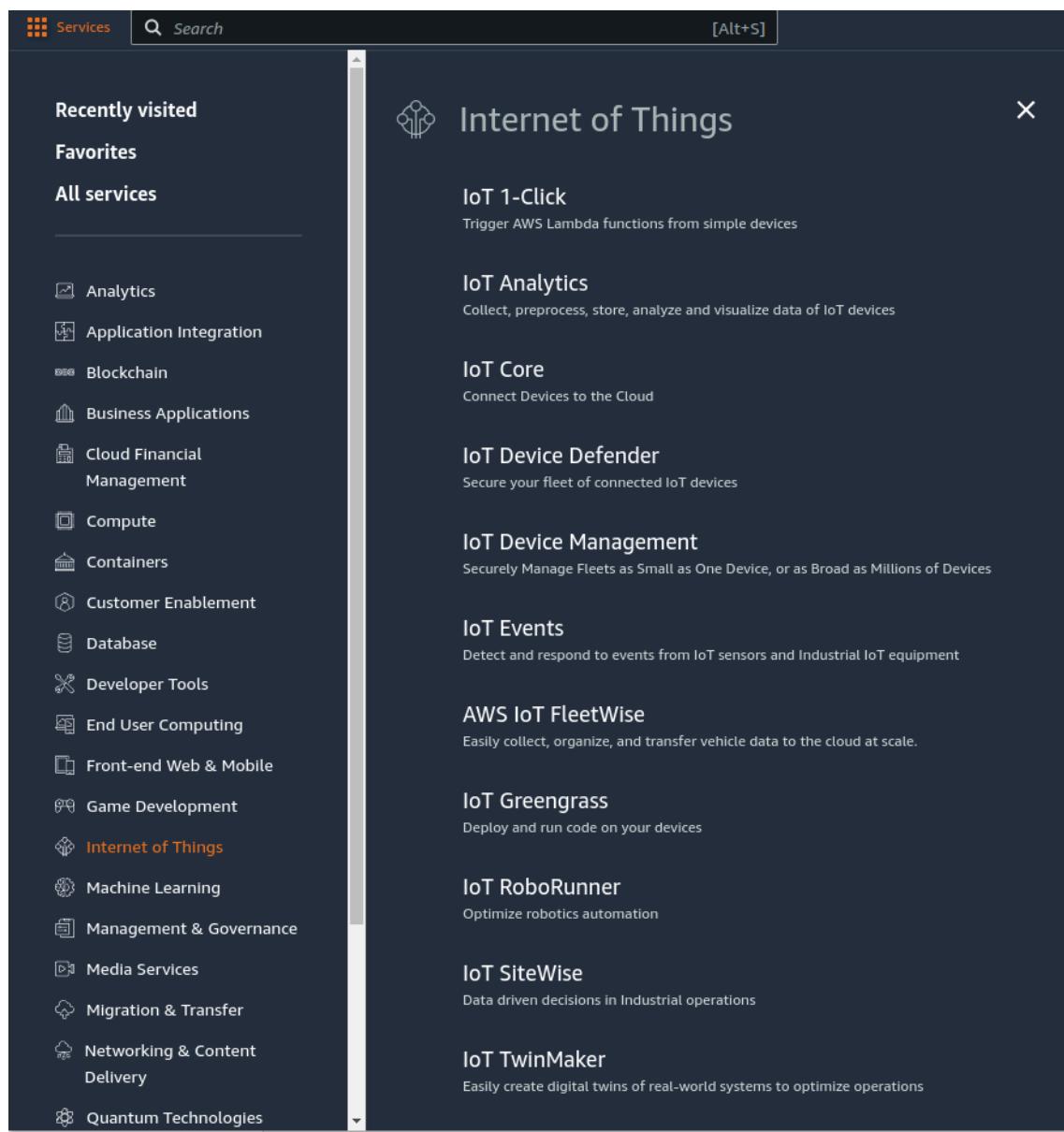
- [Set up the device environment](#)
- [Provide AWS credentials to the device](#). For development environments, you can use the option “Use long-term credentials from an IAM User”. An example of how to do this is shown below:

```
export AWS_ACCESS_KEY_ID=<the access key id for your user>
export AWS_SECRET_ACCESS_KEY=<the secret access key for your user>
```

- [Download the AWS IoT Greengrass Core software](#)
- [Install the AWS IoT Greengrass Core software \(console\)](#)
- [Install the AWS IoT Greengrass Core software](#)

8.1 Steps detail for Installing AWS IoT Greengrass on KR260

Steps and scripts for installing greengrass device is provided by AWS Greengrass dashboard in AWS web console. So first access the AWS Greengrass IoT page, go to AWS Services -> Internet of Things -> IoT Greengrass link



AWS IoT Services Search [Alt+S]

AWS IoT

- Monitor
- Connect
 - Connect one device
 - Connect many devices
- Test
 - Device Advisor
 - MQTT test client
 - Device Location [New](#)
- Manage
 - All devices
 - Greengrass devices**
 - Core devices
 - Components
 - Deployments
 - Groups (V1)
 - LPWAN devices
 - Software packages [New](#)
 - Remote actions
 - Message routing
 - Retained messages
 - Security
 - Fleet Hub

Internet of Things

AWS IoT Greengrass

Build, deploy, and manage device software at-scale

Bring local compute, messaging, data management, sync, and ML inference capabilities to edge devices.

How it works

Key terminology in Greengrass

Get started with Greengrass

Set up one core device within minutes.

[Set up one core device](#)

Manage the components that deploy to your core devices.

[View deployments](#)

Pricing (US)

10,000 devices/month	\$0.16 per device
>10,000 devices/month	Contact us

[Learn more](#)

More resources

- Documentation
- FAQ
- [Greengrass forum](#)
- Contact us

Now click on “Set up one core device” button

This will open the Greengrass core device setup page:

Here you change the Core device name like ‘kr260-dev’

Set up one Greengrass core device

Step 1: Register a Greengrass core device

Greengrass core devices are AWS IoT things. Enter a thing name to be used to create a Greengrass core device.

Core device name

The name of the AWS IoT thing to create. We generated the following name for you.

kr260-dev

The name can be up to 128 characters. Valid characters: a-z, A-Z, 0-9, underscore (_), and hyphen (-).

Step 2: Add to a thing group to apply a continuous deployment

Add your Greengrass core device to an AWS IoT thing group. If the thing group has an active Greengrass deployment, your new core device receives and applies the deployment when you finish the setup process. To deploy to only the core device, select No group.

Thing group

- Enter a new group name
- Select an existing group
- No group

Thing group name

GreengrassQuickStartGroup

Step 3: Install the Greengrass Core software

Operating System

Linux

Windows

Step 3.1: Install Java on the device

Now in KR260 terminal console run following commands and scripts:

```
export AWS_ACCESS_KEY_ID=<AWS_ACCESS_KEY_ID>
export AWS_SECRET_ACCESS_KEY=<AWS_SECRET_ACCESS_KEY>
```

Greengrass CLI depends on Java. So to install the dependency run the following:

```
sudo apt install default-jre
sudo apt install default-jdk
```

Download and install Greengrass core software.

```
curl -s https://d2s8p88vqu9w66.cloudfront.net/releases/greengrass-nucleus-latest.zip > greengrass-nucleus-latest.zip && unzip greengrass-nucleus-latest.zip -d GreengrassInstaller
```

Next install the Greengrass core device:

```
sudo -E java -Droot="/greengrass/v2" -Dlog.store=FILE -jar  
./GreengrassInstaller/lib/Greengrass.jar --aws-region us-east-1 --thing-name kr260-dev  
--thing-group-name GreengrassQuickStartGroup --component-default-user ggc_user:ggc_group  
--provision true --setup-system-service true --deploy-dev-tools true
```

Here is the console log after running above command:

```
Provisioning AWS IoT resources for the device with IoT Thing Name: [kr260-dev]...  
Found IoT policy "GreengrassV2IoTThingPolicy", reusing it  
Creating keys and certificate...  
Attaching policy to certificate...  
Creating IoT Thing "kr260-dev"...  
Attaching certificate to IoT thing...  
Successfully provisioned AWS IoT resources for the device with IoT Thing Name: [kr260-dev]!  
Adding IoT Thing [kr260-dev] into Thing Group: [GreengrassQuickStartGroup]  
IoT Thing Group "GreengrassQuickStartGroup" already existed, reusing it  
Successfully added Thing into Thing Group: [GreengrassQuickStartGroup]  
Setting up resources for aws.greengrass.TokenExchangeService ...  
Attaching TES role policy to IoT thing...  
No managed IAM policy found, looking for user defined policy...  
IAM policy named "GreengrassV2TokenExchangeRoleAccess" already exists. Please attach it to the IAM role if not already  
Configuring Nucleus with provisioned resource details...  
Root CA file found at "/greengrass/v2/rootCA.pem". Contents will be preserved.  
Downloading Root CA from "https://www.amazontrust.com/repository/AmazonRootCA1.pem"  
Created device configuration  
Successfully configured Nucleus with provisioned resource details!  
Thing group exists, it could have existing deployment and devices, hence NOT creating deployment for Greengrass first party dev tools, please manually create a deployment if you wish to  
Successfully set up Nucleus as a system service  
ubuntu@Kriaa:~$
```

Now in Greengrass set up page, one can view the Greengrass core devices and find above `kr260-dev` in the list.

The screenshot shows the AWS IoT Greengrass Core devices page. On the left, there's a sidebar with options like Monitor, Connect, Test, and Manage. Under Manage, 'Core devices' is selected. The main content area has a header 'Greengrass core devices' with a link to 'Info'. Below it is a table titled 'Greengrass core devices (2)'. The table has columns for Name, Status, and Status reported. It lists two entries: 'kr260-dev' (Status: Healthy, Status reported: 2 minutes ago) and 'GreengrassQuickStartCore-18c10b0c382' (Status: Healthy, Status reported: 1 day ago). There are also buttons for 'Configure cloud discovery' and 'Set up one core device'.

Name	Status	Status reported
kr260-dev	Healthy	2 minutes ago
GreengrassQuickStartCore-18c10b0c382	Healthy	1 day ago

In KR260 terminal one can get the device components by using `greengrass-cli`:

```
sudo /greengrass/v2/bin/greengrass-cli component list
```

```
ubuntu@kria1:~$ sudo /greengrass/v2/bin/greengrass-cli component list
Components currently running in Greengrass:
Component Name: aws.greengrass.Nucleus
  Version: 2.12.0
  State: FINISHED
  Configuration: {"awsRegion":"us-east-1","componentStoreMaxSizeBytes":"10000000000","deploymentPollingFrequencySeconds":"15","envStage":"prod","fipsMode":"false","fleetStatus":{"periodicSt
atusPublishIntervalSeconds":86400,0},"greengrassDataPlaneEndpoint":"","greengrassDataPlanePort":8443,"httpClient":{},"iotCredEndpoint":"cluwyav54wpvxq.credentials.iot.us-east-1.amazonaws.co
m","iotDataEndpoint":"a91c30bcutf8v-ats.iot.us-east-1.amazonaws.com","iotRoleAlias":"GreengrassV2TokenExchangeRoleAlias","jvmOptions": "-Dlog.store=FILE","logging":{},"mqtt":{"spooler":{}}, "telemetry":{}}
Component Name: FleetStatusService
  Version: 0.0.0
  State: RUNNING
  Configuration: null
Component Name: UpdateSystemPolicyService
  Version: 0.0.0
  State: RUNNING
  Configuration: null
Component Name: aws.greengrass.Cli
  Version: 2.12.0
  State: RUNNING
  Configuration: {"AuthorizedPosixGroups":null,"AuthorizedWindowsGroups":null}
Component Name: TelemetryAgent
  Version: 0.0.0
  State: RUNNING
  Configuration: null
Component Name: DeploymentService
  Version: 0.0.0
  State: RUNNING
  Configuration: null
```

We will be adding component to publish and subscribe the topic to the AWS cloud Broker.

9. Installing the components

Before installing “mqtt-gpio” component, get familiar with the dummy “Hello World” component.

9.1 Create a “Hello World” component

9.1.1 Create the component on your edge device

Follow the instructions online under the section [Develop and test a component on your device](#) to create a simple component on your device.

9.1.2 Upload the “Hello World” component

Follow the instructions online at [Create your component in the AWS IoT Greengrass service](#) to upload your component to the cloud, where it can be deployed to other devices as needed.

9.1.3 Deploy your component

Follow the instructions online at [Deploy your component](#) to deploy and verify that your component is running.

9.2 Installing “mqtt-gpio” component

9.2.1 Get the component

Get the `components` folder from the downloaded resources and copy in the KR260 home directory.

It contains:

Artifacts

- com.example.mqtt
 - 1.0.0
 - mqtt.py (This python code published the data on button press and actuates gpio on receiving the data in subscribed topic)

Recipe

- com.example.mqtt-1.0.0.json

9.2.2 Deploy the component

To install the above component run the following in the KR260 terminal:

```
sudo /greengrass/v2/bin/greengrass-cli deployment create \
--recipeDir ~/components/recipe \
--artifactDir ~/components/artifacts \
--merge "com.example.mqtt=1.0.0"
```

```
ubuntu@kria:~$ sudo /greengrass/v2/bin/greengrass-cli deployment create \
--recipeDir ~/components/recipe \
--artifactDir ~/components/artifacts \
--merge "com.example.mqtt=1.0.0"
Local deployment submitted! Deployment Id: 9e8f1be6-63b2-4189-aecc-607197755d22
ubuntu@kria:~$ █
```

9.2.3 Upload the component

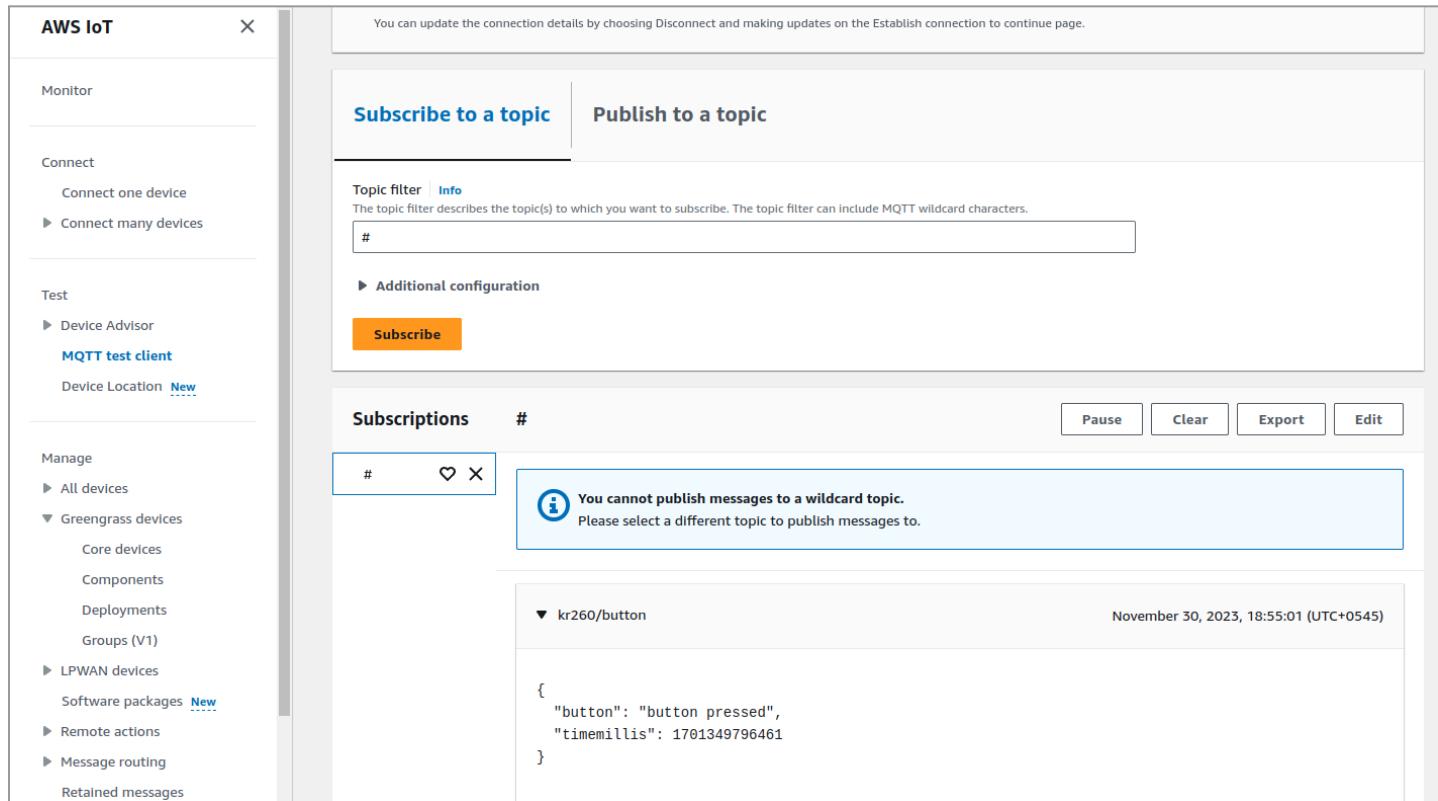
Follow the instructions online at [Create your component in the AWS IoT Greengrass service](#) to upload your component to the cloud, where it can be deployed to other devices as needed.

10. Debugging

Now check the installed component is in "running state"

```
ubuntu@ip-172-31-10-11:~$ sudo /greengrass/v2/bin/greengrass-ctl component list
Components currently running in Greengrass:
Component Name: aws.greengrass.Nucleus
  Version: 2.12.0
  State: FINISHED
  Configuration: {"awsRegion":"us-east-1","componentStoreMaxSizeBytes":10000000000,"deploymentPollingFrequencySeconds":15,"envStage":"prod","fipsMode":false,"fleetStatus":{"periodicStatusPublishIntervals":86400}, "greengrassDataPlaneEndpoint":"","httpClient":{},"iotCredEndpoint":"cluyav4wpvxp.credentials.iot.us-east-1.amazonaws.com","iotDataEndpoint":"a9jc3obcf8v-ats.iot.us-east-1.amazonaws.com","iotRoleAlias":"GreengrassV2TokenExchangeRoleAlias","jvmOptions":"-Dlog.store=FILE","logging":{},"mqtt":{"spooler":{}}, "networkProxy":{"proxy":{}}, "platformOverride":{}, "runWithDefaultShell":"sh","posixUser":"ggc_user:ggc_group"}, "s3EndpointType": "GLOBAL", "telemetry": {}}
Component Name: con.example.mqtt
  Version: 1.0.0
  State: RUNNING
  Configuration: {"accessControl": {"aws.greengrass.lpc.mqtproxy": {"com.example.mqtt:mqtproxy:1": {"operations": ["aws.greengrass#PublishToIoTCore", "aws.greengrass#SubscribeToIoTCore"]}, "policyDescription": "Allows access to pub/sub to mypl/mqtt", "resources": ["kr260/mqtt", "kr260/button"]}}, "message": "hello"}
Component Name: TelemetryAgent
  Version: 0.0.0
  State: RUNNING
  Configuration: null
Component Name: DeploymentService
  Version: 0.0.0
  State: RUNNING
  Configuration: null
Component Name: UpdateSystemPolicyService
  Version: 0.0.0
  State: RUNNING
  Configuration: null
Component Name: FleetStatusService
  Version: 0.0.0
  State: RUNNING
  Configuration: null
Component Name: aws.greengrass.cli
  Version: 2.12.0
  State: RUNNING
  Configuration: {"AuthorizedPosixGroups": null, "AuthorizedWindowsGroups": null}
Component Name: aws.greengrass.LocalDebugConsole
  Version: 2.4.1
  State: RUNNING
  Configuration: {"bindHostname": "localhost", "httpsEnabled": "true", "port": "1441", "websocketPort": "1442"}
```

Now in aws IoT console, open "MQTT test client" and subscribe to "#"



You can see the "button pressed" message once the button is pressed.

Now to control the LED, publish the message to "kr260/mqtt" topic. Here is the screenshot of the message which switch on the LED.

AWS IoT

You can update the connection details by choosing Disconnect and making updates on the Establish connection to continue page.

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

Topic name
The topic name identifies the message. The message payload will be published to this topic with a Quality of Service (QoS) of 0.
kr260/mqtt

Message payload
{
 "ledon": true
}

► Additional configuration

Publish

Subscriptions #

Info You cannot publish messages to a wildcard topic.
Please select a different topic to publish messages to.

▼ kr260/mqtt

November 30, 2023, 18:57:32 (UTC+0545)

Now to switch off the LED send "false" message in the "kr260/mqtt" topic.

AWS IoT

You can update the connection details by choosing Disconnect and making updates on the Establish connection to continue page.

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

Topic name
The topic name identifies the message. The message payload will be published to this topic with a Quality of Service (QoS) of 0.
kr260/mqtt

Message payload
{
 "ledon": false
}

► Additional configuration

Publish

Subscriptions #

Info You cannot publish messages to a wildcard topic.
Please select a different topic to publish messages to.

▼ kr260/mqtt

November 30, 2023, 18:59:36 (UTC+0545)

11. Troubleshooting

For more information, refer to the online documentation [*Troubleshooting Greengrass v2*](#).

You can also refer to [Logging and Monitoring](#) to learn how to log API calls, gather system health telemetry data, and check core device status.

If you need support on Kria SoM with Greengrass, then you can go for [AMD Kria SoMs forum](#).