Edge AI box

Getting Started Guide for AWS IoT Greengrass

Purpose

This Getting Start Guide walk users through setting up the hardware and development environment to develop applications for Edge AI box, as well as building, deploying, and running the Hello World demo component on a device.

Table of Contents

Revision History

Version	Date	Description	Before Revised	After Revised
2	2023/02/10	1. Modify for content typo 2. Add debugging and troubleshooting chapter	1. Heimdallr AI box 2. N/A	1. Edge AI box 2. Add debugging and troubleshooting chapter
1	2022/12/28	New version	N/A	N/A

1 Overview

The Edge AI box is a powerful edge computing with 24 channel video AI inference capability and equipped optional 5G connectivity which can transmit data with low latency, reliable and multiple access in IOT application.
With AWS IOT Greengrass, user can easily deploy machine learning model into edge device. The data can sync up in time and communicate to another device securely. Inventec Edge AI box can allow you seamlessly to run IOT application across the AWS cloud and stream real-time data securely to other IT or OT application.

1.1 About AWS IoT Greengrass

To learn more about AWS IoT Greengrass, see how it works and what's new.

2 Hardware Description

2.1 DataSheet

The more detailed chip feature is mentioned below.

2.1.1 Qualcomm QSM825

1. Storage: 128G UFS NAND flash (256G compatible)
2. AI Accelerator Card: Qualcomm AI100-DM.2e
3. Wireless: 5G NR
   - Sierra AirPrime EM9191(Sub-6G)
   - Sierra AirPrime EM9190(mmW)
4. IO Interface:
   - 2 x RJ45 for 1000BASE-T
   - 1 x mini FRAKA (1x4) for Camera Input
   - 4 x SMA Connector for Sub-6G Antenna
   - 1 x DC Jack Connector for Power in
   - Chassis Outline: 208.1 x 165.4 x 76.5 mm
   - Ingress Protection:
     - IP50 for indoor
     - IP67 for outdoor (developing)

2.1.2 AIC100

The Qualcomm Cloud AI 100(AIC100), designed for AI inference acceleration, addresses unique requirements in the cloud, including power efficiency, scale, process node advancements, and signal processing—facilitating the ability of datacenters to run inference on the edge cloud faster and more efficiently. Qualcomm Cloud AI 100 is designed to be a leading solution for datacenters who increasingly rely on infrastructure at the edge-cloud.

2.1.3 EM919X

EM919X/EM7690 is an M.2 module with FirstNet-ready (B14 LTE). It provides 5G NR Sub-6G, 5G mmWave, 4G LTE advanced Pro, 3G (HSPA+, UMTS) subject to variants, and GNSS connectivity for a wide range of devices and purposes including business, personal, portable computing and communication devices, IoT devices, M2M applications and industrial use cases.

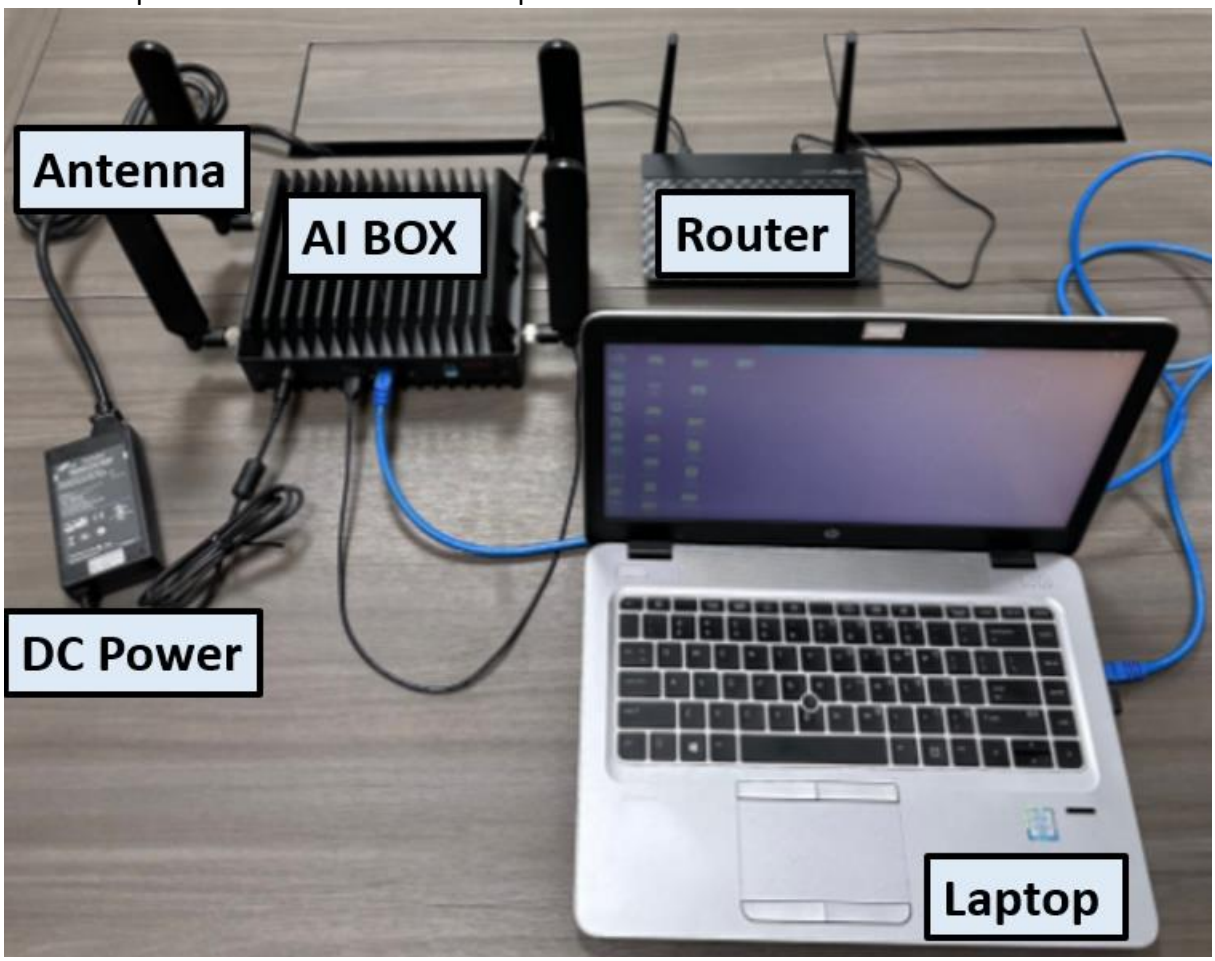
For more detail, please visit Inventec website:
https://ame.inventec.com/en/product/11

2.2 Standard Kit Contents

- AI box
- Antenna x 4
- Power adaptor
- Power cord

3 Set up your hardware

- The picture below shows the components with annotations.



Component diagram

- Description of key components, external ports
  a. AI box includes
     - Qualcomm QSM8250 module
     - Max 24 FHD IP cameras
     - Low latency and high bandwidth cloud connectivity with
       - Sub-6
       - mmWave

- AI inference acceleration with Qualcomm AI100 engine
- Feature-rich Qualcomm AI100 software development API
- Upgradable function with Firmware OTA

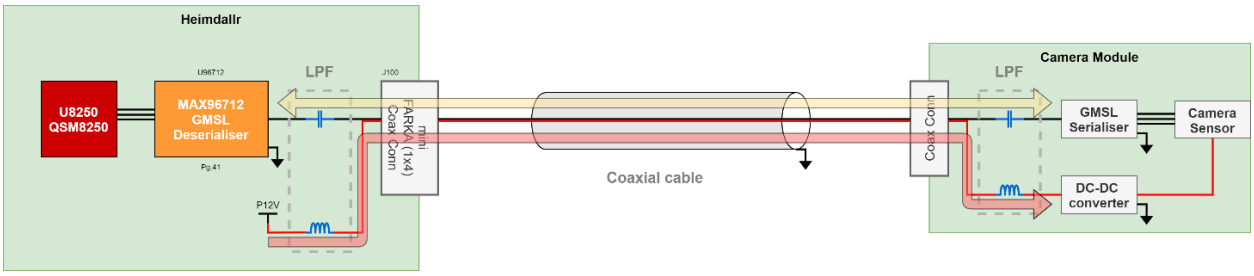
b. DC power supply
- DC Power Jack Connector (P12VIN)
- Provide power to the AI box through the power cable connection

c. Laptop: Connect the laptop and AI box with the RJ45 cable

d. Router: Provide network, and connect Router and AI box through RJ45 cable

e. Antennas: Four lockable helical antennas for signal boost

- Power supply and connector requirements and instructions, battery options if any. Power over Coax (POC) is a technology that simultaneously transmit video data and DC power to remote camera module with only one coaxial cable (reducing the number of cables).



Block Diagram of POC

- Detailed jumper/switch settings for flashing, normal operation, reset, other modes.

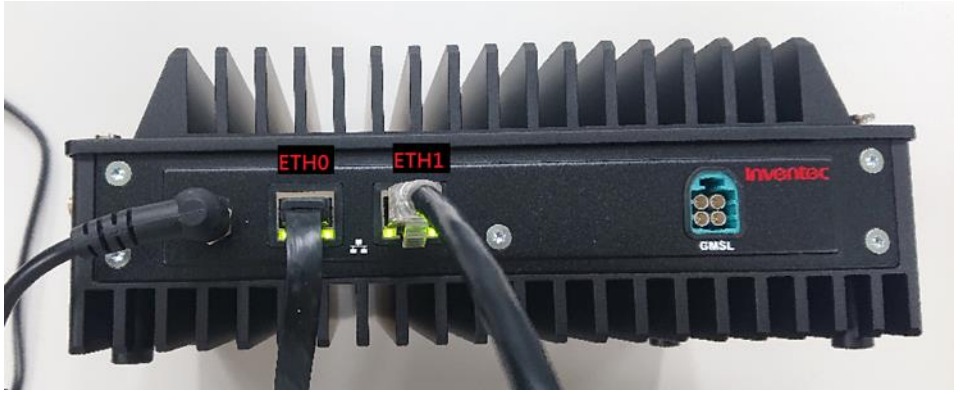
a. Switch
- SW1: Reset BTN
- SW4: Boot Mode Select

b. Jumper
- J6513: SUN_DETECT

- Provide links to download pages and other documentation for supported driver versions. https://ame.inventec.com/en/product/11

4 Set up your Development Environment

4.1 How to access the system

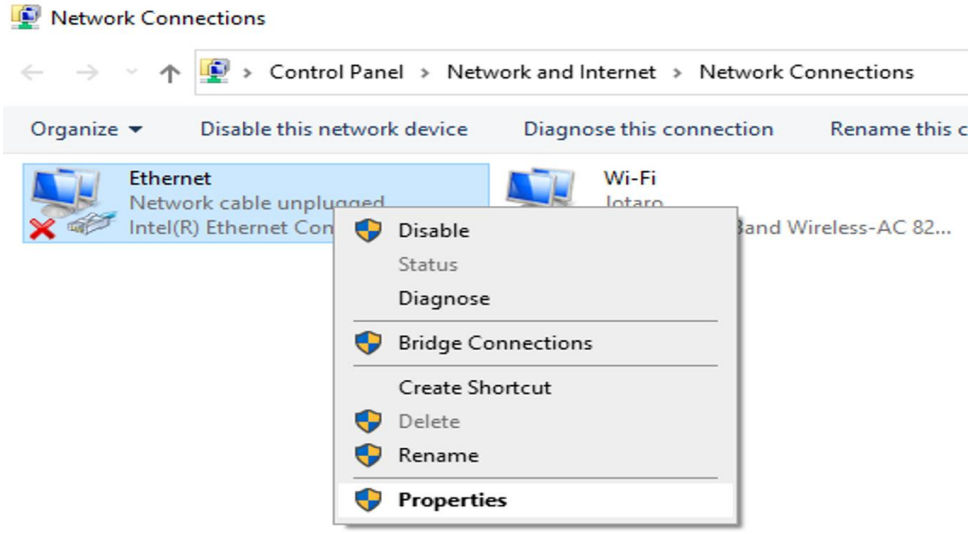


Backpanel review

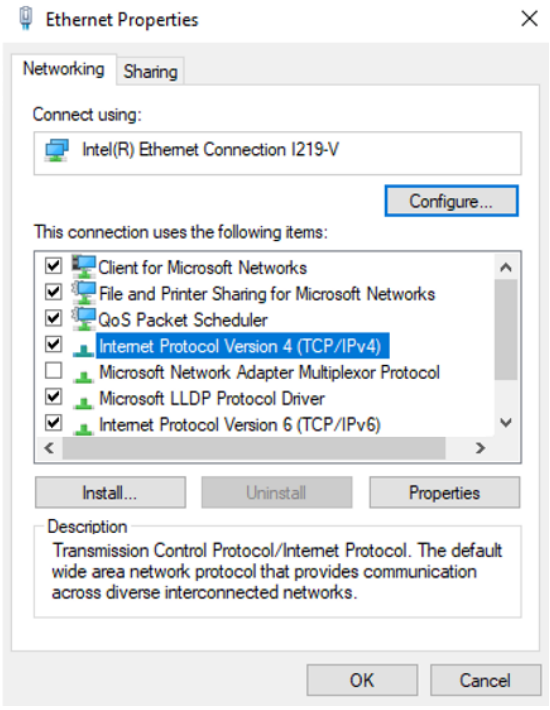
There are 2 ethernet devices in the OS: ETH0 and ETH1(through RJ45 ports). ETH0 is set as a dynamic IP port. Users can connect it with a DHCP function enabled router to access it. Using the management tool of the router to get the IP of the ETH0. ETH1 is set as a static IP port; The static IP is set to 192.168.1.1. Users can connect this port to another RJ45 port of a notebook. Then use the notebook to access ETH1. Users can use both ways mentioned above and then use the SSH connection to access the AI Box kernel. The account name and password, please contact the relevant sales. Attempting to access the system after 60 seconds power-on is recommended.

Setup a notebook to access the ETH1(static IP)
To set a static IP address in Windows:

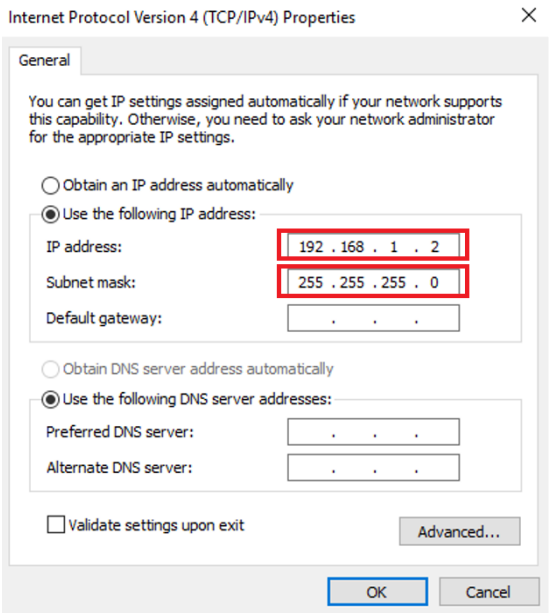
- Click **Start Menu** > **Control Panel** > **Network and Sharing Center** or **Network and Internet** > **Network and Sharing Center**.
- Click **Change adapter settings**.
- Right-click **Wi-Fi** or **Local Area Connection**.
- Click **Properties**.



- Select Internet Protocol Version 4 (TCP/IPv4).
- Click Properties.

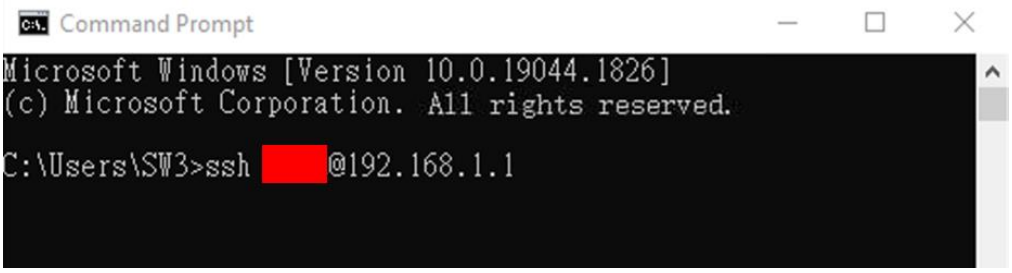


- Select Use the following IP address.
- Enter the IP address, and Subnet mask.
- Click OK.

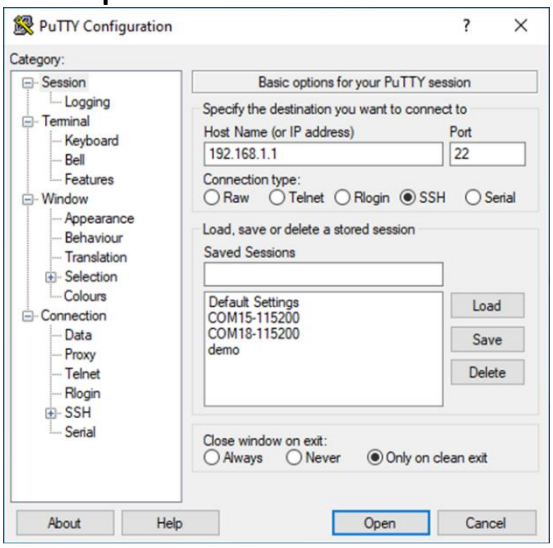


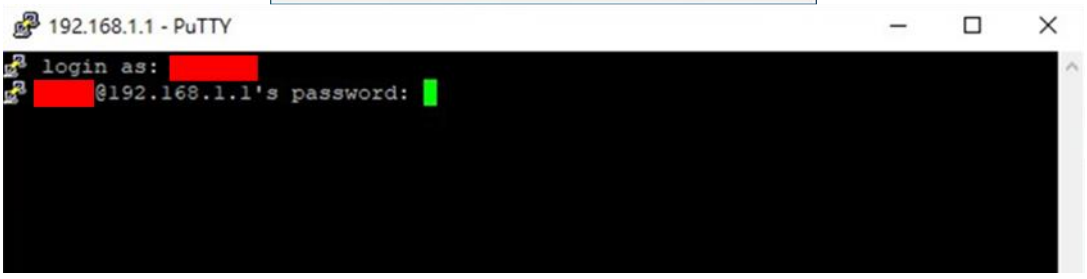
Using SSH connection to enter system (ETH1)

- Connect the RJ45 ports between eth1 of AI Box and the static-IP-port-ready notebook.
- Use SSH connect to IP 192.168.1.1
- In Windows OS, if it has ssh command, users can use command line interface to enter "ssh account@192.168.1.1"



- User can also use PuTTY program to do the SSH connection. Enter IP 192.168.1.1 , choose ssh connection type and port 22. After clicking the open, follow the indication to enter the **account name** and the **password**.





4.2 How to confirm Network connection status

Please enter the system and then type the command below:

```
ifconfig
```

If ETH0 connects to router successfully, you will see IP address.

```
[root@sm8250 ~]# ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 172.20.10.12  netmask 255.255.255.240  broadcast 172.20.10.15
        inet6 fe80::f83b:33ca:d45c:add5  prefixlen 64  scopeid 0x20<link>
        inet6 2001:b400:e355:5103:c7b8:e694:dcc6:52f7  prefixlen 64  scopeid 0x0<global>
        ether 38:68:dd:6b:0a:61  txqueuelen 1000  (Ethernet)
        RX packets 29  bytes 3041 (2.9 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 59  bytes 5541 (5.4 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

eth1: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.1.1  netmask 255.255.255.0  broadcast 192.168.1.255
        inet6 fe80::3a68:ddff:fe6b:a60  prefixlen 64  scopeid 0x20<link>
        ether 38:68:dd:6b:0a:60  txqueuelen 1000  (Ethernet)
        RX packets 124  bytes 13482 (13.1 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 102  bytes 18302 (17.8 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

IP address

4.3 How to install Java and update Python on the device

Please install Java and update python for AWS greengrass.

On CentOS:

Install Java:

```
sudo yum -y install wget curl
wget
https://download.java.net/java/GA/jdk17.0.2/dfd4a8d0985749f896bed50d7138ee7f/8/GPL/o
penjdk-17.0.2_linux-aarch64_bin.tar.gz
tar xvf openjdk-17.0.2_linux-aarch64_bin.tar.gz
sudo mv jdk-17.0.2/ /opt/jdk-17/
vim ~/.bashrc
export JAVA_HOME=/opt/jdk-17
export PATH=$PATH:$JAVA_HOME/bin
source ~/.bashrc
sudo visudo
Defaults  secure_path = /sbin:/bin:/usr/sbin:/usr/bin:/opt/jdk-17/bin
java --version
```

```
[root@sm8250 ~]# java -version
openjdk version "1.8.0_312"
OpenJDK Runtime Environment (build 1.8.0_312-b07)
OpenJDK 64-Bit Server VM (build 25.312-b07, mixed mode)
```

Update Python:

```
yum groupinstall 'development tools' -y && yum install wget openssl-devel bzip2-
devel libffi-devel xz-devel -y
wge t https://www.python.org/ftp/python/3.9.7/Python-3.9.7.tgz
tar zxvf Python-3.9.7.tgz
cd Python-3.9.7/
./configure --enable-optimizations
make
sudo make altinstall
```

```
alternatives --install /usr/bin/python3 python3 /usr/local/bin/python3.9 1 &&
alternatives --set python3 /usr/local/bin/python3.9 && echo "2" | alternatives --
config python
python3 -V
```

4.4 Setting up 5G APN (if available)

Please enter the system and then type the 2 commands below: (APN is the customer APN).

```
nmcli con modify gsm-cdc-wdm0 gsm.apn APN
nmcli con reload
```

```
[root@sm8250 ~]# nmcli con modify gsm-cdc-wdm0 gsm.apn APN
[root@sm8250 ~]# nmcli con reload
[root@sm8250 ~]#
```

Confirm APN setting

```
nmcli -p con show gsm-cdc-wdm0 | grep gsm.apn
```

```
[root@sm8250 ~]# nmcli -p con show gsm-cdc-wdm0 | grep gsm.apn
gsm.apn:                                APN
```

5 Setup your AWS account and Permissions

Refer to the online AWS documentation at Set up your AWS Account. Follow the steps outlined in the sections below to create your account and a user and get started:

- Sign up for an AWS account and
- Create a user and grant permissions
- Open the AWS IoT console

6 Create Resources in AWS IoT

Refer to the online AWS documentation at Create AWS IoT Resources. Follow the steps outlined in these sections to provision resources for your device:

- Create an AWS IoT Policy
- Create a thing object

7 Install the AWS Command Line Interface

To install the AWS CLI on your host machine, refer to the instructions at Installing the AWS CLI v2. Installing the CLI is needed to complete the instructions in this guide.

Once you have installed AWS CLI, configure it as per the instructions in this online guide. Set the appropriate values for Access key ID, Secret access key, and AWS Region. You can set Output format to "json" if you prefer.

8 Install AWS IoT Greengrass

8.1 Download the AWS IoT Greengrass Core software

If Greengrass has not been included in the SD card image, you can download the latest greengrass core software as follows:

```
wget https://d2s8p88vqu9w66.cloudfront.net/releases/greengrass-nucleus-latest.zip
```

8.2 Install the AWS IoT Greengrass Core software

Unzip the AWS IoT Greengrass Core software to a folder on your device. Replace GGCoreInstall with the folder that you want to use.

```
unzip greengrass-nucleus-latest.zip -d GGCoreInstall
rm greengrass-nucleus-latest.zip
```

Verify the version of the AWS IoT Greengrass Core software:

```
java -jar ./GGCoreInstall/lib/Greengrass.jar --version
```

You will see the Greengrass version displayed - similar to:

AWS Greengrass v2.8.0

8.2.1 Provide your credentials

Run the following commands to provide the credentials to the AWS IoT Greengrass Core software.

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

8.2.2 Run the installer

Run the installer as shown below. Modify the values as per your region, install directory and thing name.

Use the **--provision true** option to have the installer set up the "thing" and required policies for you. If you prefer to configure Greengrass manually, see the online guide.

```
sudo -E java -Droot="/greengrass/v2" -Dlog.store=FILE \
-jar ./GGCoreInstall/lib/Greengrass.jar \
--aws-region us-west-2 \
--thing-name thing-name \
--tes-role-name GreengrassV2TokenExchangeRole \
--tes-role-alias-name GreengrassCoreTokenExchangeRoleAlias \
--component-default-user ggc_user:ggc_group \
--provision true \
--setup-system-service true \
--deploy-dev-tools true
```

If all goes well, you will see the following output on the device console:

Successfully configured Nucleus with provisioned resource details!
Configured Nucleus to deploy aws.greengrass.Cli component
Successfully set up Nucleus as a system service

The local development tools (specified by the --deploy-dev-tools option) take some time to deploy. The following command can be used to check the status of this deployment:

```
aws greengrassv2 list-effective-deployments --core-device-thing-name thing-name
```

When the status is SUCCEEDED, run the following command to verify that the Greengrass CLI is installed and runs on your device. Replace /greengrass/v2 with the path to the base folder on your device as needed.

```
/greengrass/v2/bin/greengrass-cli help
```

9 Create a Hello World component

In Greengrass v2, components can be created on the edge device and uploaded to the cloud, or vice versa.

9.1 Create the component on your edge device

Follow the instructions online under the section To create a Hello World component to create, deploy, test, update and manage a simple component on your device.

9.2 Upload the Hello World component

Follow the instructions online at Upload your component to upload your component to the cloud, where it can be deployed to other devices as needed.

10 Debugging

If you need to check logs.
Boot up logs:

```
dmesg
```

Specific facility:

```
dmesg -f facility
```

Supported log facilities list:
kern – kernel messages
user – random user-level messages
mail – mail system
daemon – system deamons
auth – security/authorization messages
syslog – messages generated internally by syslogd
lpr – line printer subsystem
news – network news subsystem

Specific log levels:

```
dmesg -l levels
```

Supported log levels list:
emerg - system is unusable
alert - action must be taken immediately
crit - critical conditions
err - error conditions
warn - warning conditions
notice - normal but significant condition
info - informational
debug - debug-level messages

For more detailed usages

```
dmesg -help
```

11 Troubleshooting

If you have any problems when using device, you can check the command provided below.

11.1 Ethernet

Check eth0 and eth1 device

```
ifconfig
```

```
[root@sm8250 ~]# ifconfig
eth0: flags=4099<UP,BROADCAST,MULTICAST>  mtu 1500
        ether 38:68:dd:6b:0a:7d  txqueuelen 1000  (Ethernet)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

eth1: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.1.1  netmask 255.255.255.0  broadcast 192.168.1.255
        inet6 fe80::3a68:ddff:fe6b:a7c  prefixlen 64  scopeid 0x20<link>
        ether 38:68:dd:6b:0a:7c  txqueuelen 1000  (Ethernet)
        RX packets 590  bytes 50784 (49.5 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 739  bytes 179488 (175.2 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 48  bytes 4380 (4.2 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 48  bytes 4380 (4.2 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

Check the usb port of 78xx

```
lsusb -t
```

```
[root@sm8250 ~]# lsusb -t
/:  Bus 04.Port 1: Dev 1, Class=root_hub, Driver=xhci-hcd/1p, 10000M
    |__ Port 1: Dev 2, If 0, Class=Vendor Specific Class, Driver=lan78xx, 5000M
/:  Bus 03.Port 1: Dev 1, Class=root_hub, Driver=xhci-hcd/1p, 480M
/:  Bus 02.Port 1: Dev 1, Class=root_hub, Driver=xhci-hcd/1p, 10000M
    |__ Port 1: Dev 2, If 0, Class=Hub, Driver=hub/3p, 5000M
        |__ Port 1: Dev 3, If 0, Class=Vendor Specific Class, Driver=lan78xx, 5000M
        |__ Port 2: Dev 4, If 0, Class=Communications, Driver=cdc_mbim, 5000M
        |__ Port 2: Dev 4, If 1, Class=CDC Data, Driver=cdc_mbim, 5000M
        |__ Port 2: Dev 4, If 3, Class=Vendor Specific Class, Driver=qcserial, 5000M
        |__ Port 2: Dev 4, If 4, Class=Vendor Specific Class, Driver=qcserial, 5000M
/:  Bus 01.Port 1: Dev 1, Class=root_hub, Driver=xhci-hcd/1p, 480M
    |__ Port 1: Dev 2, If 0, Class=Hub, Driver=hub/4p, 480M
        |__ Port 4: Dev 4, If 0, Class=Vendor Specific Class, Driver=, 480M
```

11.2 5G

Check the 4 of 5G module device

```
lsusb -t
```

```
[root@sm8250 ~]# lsusb -t
/:  Bus 04.Port 1: Dev 1, Class=root_hub, Driver=xhci-hcd/1p, 10000M
    |__ Port 1: Dev 2, If 0, Class=Vendor Specific Class, Driver=lan78xx, 5000M
/:  Bus 03.Port 1: Dev 1, Class=root_hub, Driver=xhci-hcd/1p, 480M
/:  Bus 02.Port 1: Dev 1, Class=root_hub, Driver=xhci-hcd/1p, 10000M
    |__ Port 1: Dev 2, If 0, Class=Hub, Driver=hub/3p, 5000M
        |__ Port 1: Dev 3, If 0, Class=Vendor Specific Class, Driver=lan78xx, 5000M
        |__ Port 2: Dev 4, If 0, Class=Communications, Driver=cdc_mbim, 5000M
        |__ Port 2: Dev 4, If 1, Class=CDC Data, Driver=cdc_mbim, 5000M
        |__ Port 2: Dev 4, If 3, Class=Vendor Specific Class, Driver=qcserial, 5000M
        |__ Port 2: Dev 4, If 4, Class=Vendor Specific Class, Driver=qcserial, 5000M
/:  Bus 01.Port 1: Dev 1, Class=root_hub, Driver=xhci-hcd/1p, 480M
    |__ Port 1: Dev 2, If 0, Class=Hub, Driver=hub/4p, 480M
        |__ Port 4: Dev 4, If 0, Class=Vendor Specific Class, Driver=, 480M
```

Check Mode is online

```
ls /dev/ttyUSB0
cat /dev/ttyUSB0
echo -e 'at!gstatus?\r\n' > /dev/ttyUSB0
```

Sim card unlock

```
mmcli -i 0 --pin 'xxxx'
```

xxxx is your password.

Check the sim card status

```
ls /dev/ttyUSB1
mbimcli -d /dev/ttyUSB1 -p --query-subscriber-ready-status
```

11.3 AIC100

Check AIC100 card information

```
/opt/qti-aic/tools/qaic-util -q
```

```
[root@sm8250 /]# /opt/qti-aic/tools/qaic-util -q
LRT QC_IMAGE_VERSION: LRT.AIC.7.1.1.6.3.1
LRT IMAGE_VARIANT: LRT.AIC.REL
Number of devices queried: 1
QID 0
        Status:Ready
```

If you have any further questions, please contact the relevant sales.