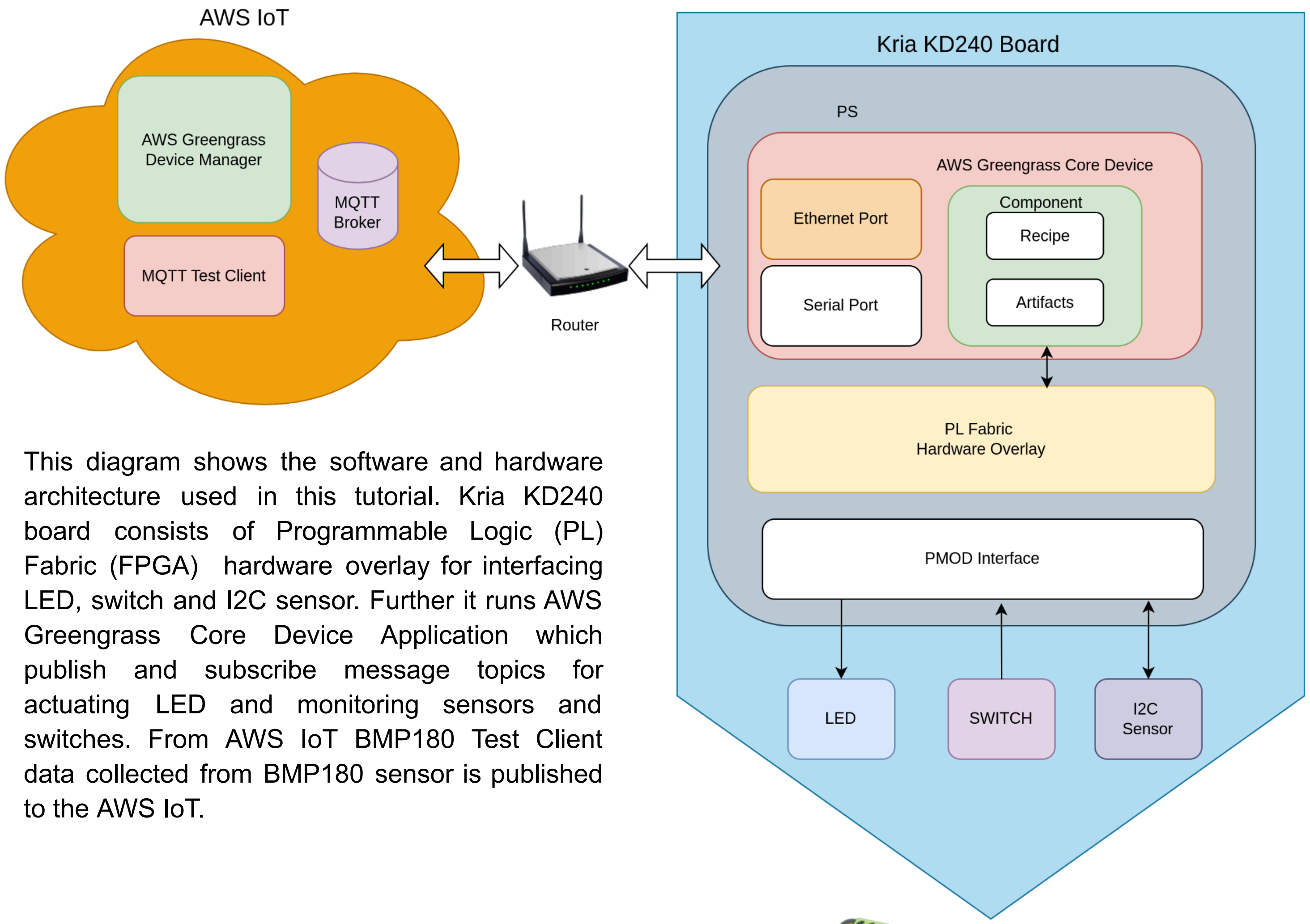
KD240 to AWS IoT Greengrass Architecture



This diagram shows the software and hardware architecture used in this tutorial. Kria KD240 board consists of Programmable Logic (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 BMP180 Test Client data collected from BMP180 sensor is published to the AWS IoT.

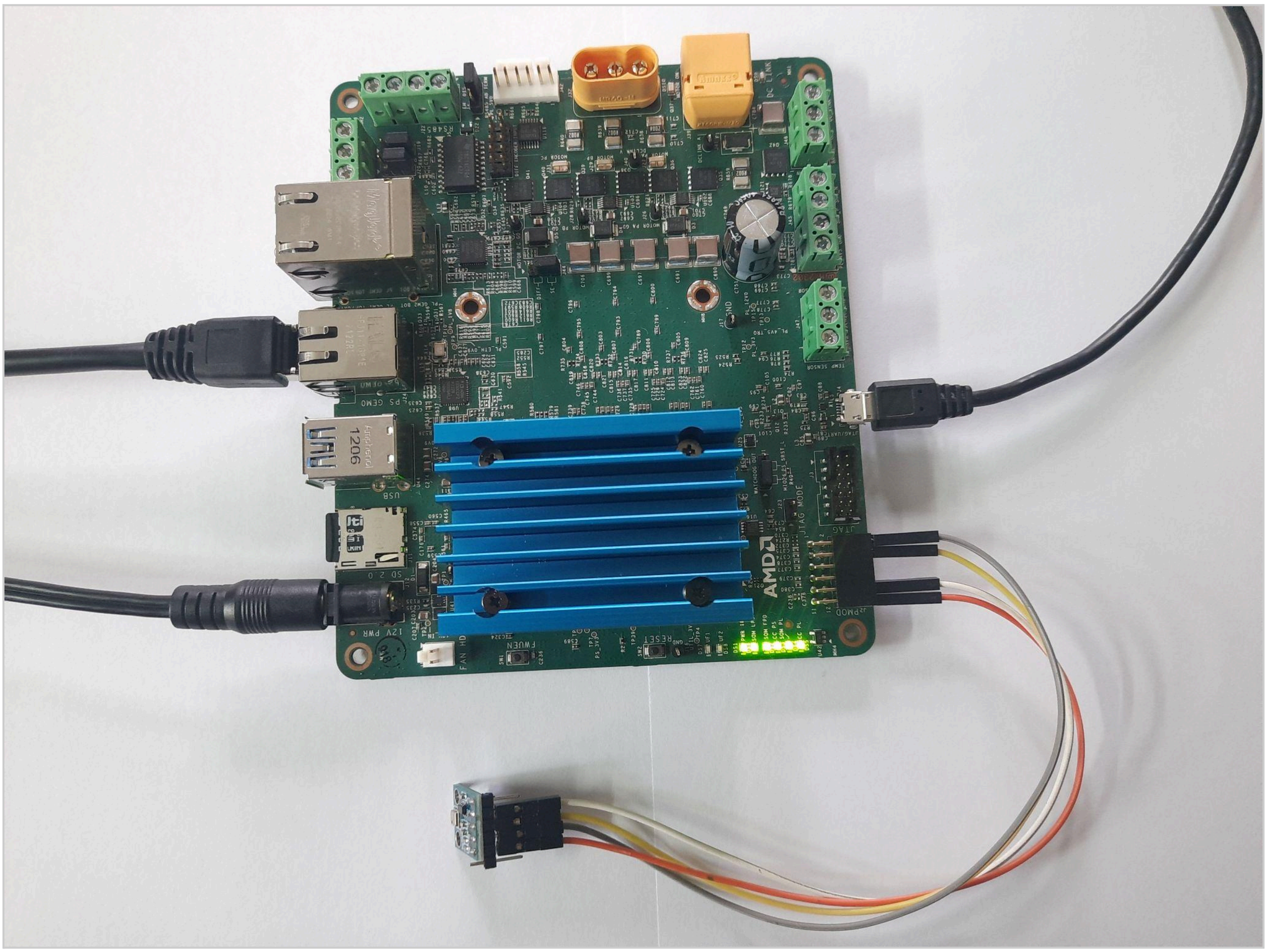
Required Hardware Components

1. Kria KD240 board
2. BMP180 Module (Available at Amazon)
3. Connecting wires

Software requirements

1. Ubuntu 22.04 for Kria KD240 board
2. AWS account

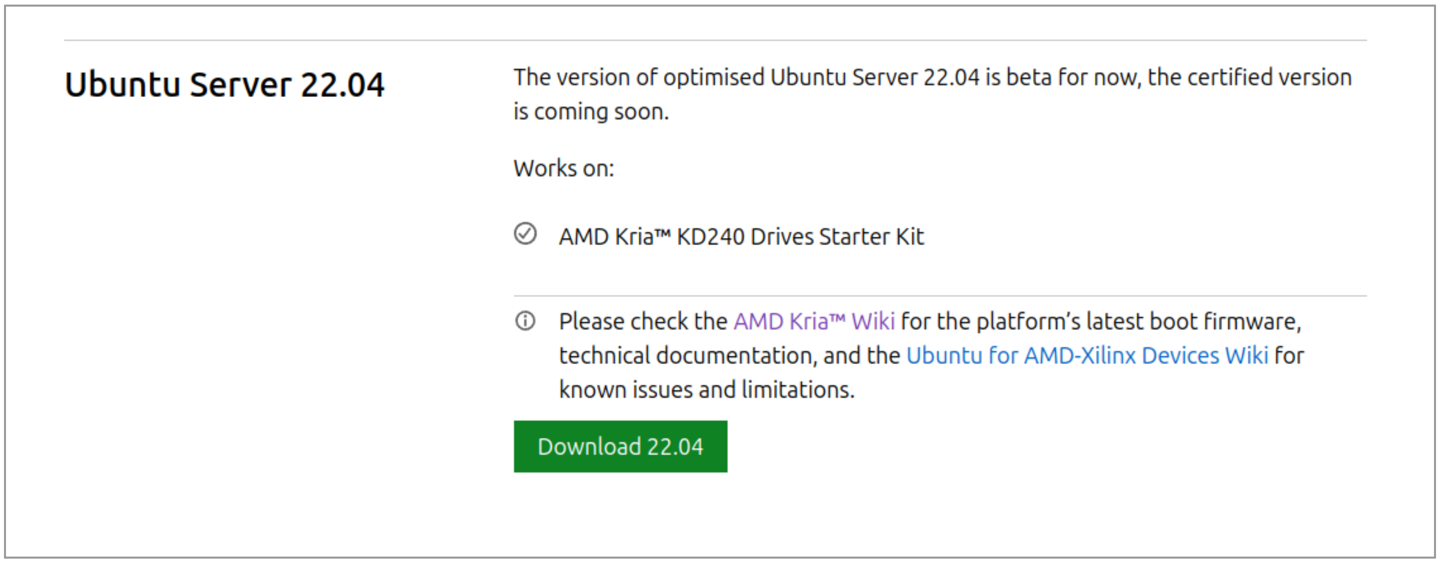
Further details and download links are available at /documents/KD240 to AWS Greengrass IoT - GPIO-(Ubuntu/Petalinux).pdf .



KD240 board connected to BMP180 through PMOD-I2C

Preparing Ubuntu 22.04 OS for KRIA KD240 board

Download the Ubuntu 22.04 image from the download link


Ubuntu Server 22.04

The version of optimised Ubuntu Server 22.04 is beta for now, the certified version is coming soon.

Works on:

AMD Kria™ KD240 Drives 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.

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

Now boot the KD240 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 KD240 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 KD240 device console.

Installing hardware overlay

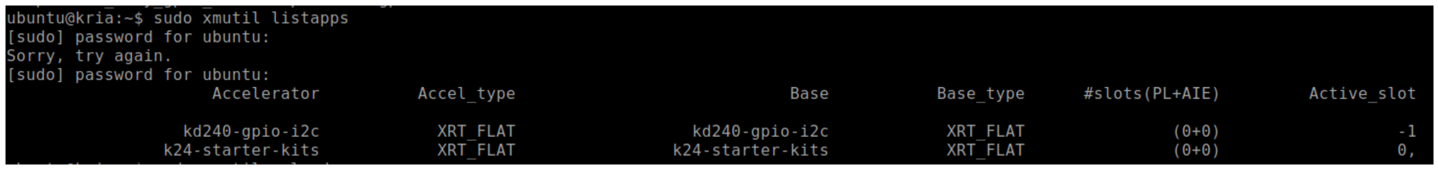
Get the KD240 firmware folder. It contains:

- kd240-gpio-i2c.bit.bin
- kd240-gpio-i2c.dtbo
- shell.json

Copy these file to the KD240 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 "kd240-gpio-i2c" at "/lib/firmware/xilinx" and copy the files in "kd240-gpio-i2c" folder.

```
cd /lib/firmware/xilinx
sudo mkdir kd240-gpio-i2c
sudo cp <kd240-firmware directory>/krc260_i2c* ./
sudo cp <kd240-firmware directory>/shell.json ./
```

Next, check the available fpga firmware using `xmutil listapps` command. `kd240-i2c` will be available in the list.

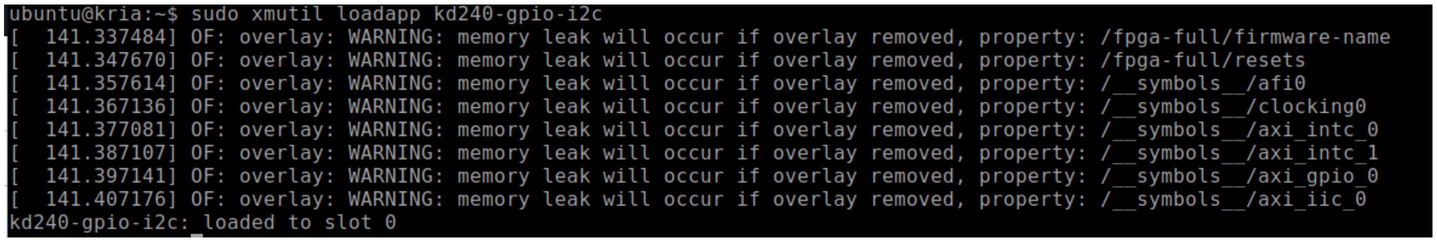


```
ubuntu@kria:~$ sudo xmutil listapps
[sudo] password for ubuntu:
Sorry, try again.
[sudo] password for ubuntu:
                     Accelerator          Accel_type                            Base          Base_type       #slots(PL+AIE)         Active_slot

                  kd240-gpio-i2c           XRT_FLAT                   kd240-gpio-i2c           XRT_FLAT                (0+0)                  -1
                k24-starter-kits           XRT_FLAT                 k24-starter-kits           XRT_FLAT                (0+0)                  0,
```

Next load the `kd240-gpio-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.

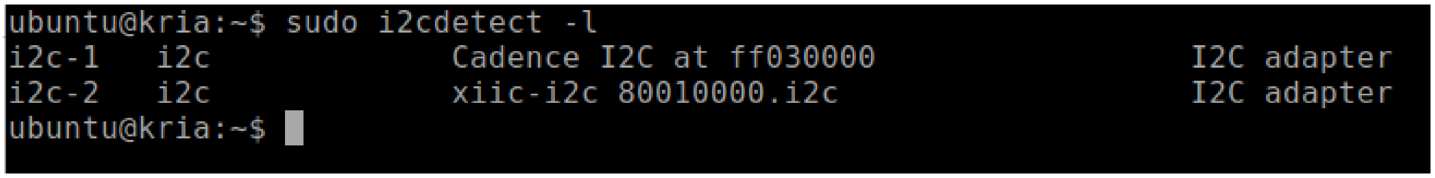
```
sudo xmutil unloadapp
sudo xmutil loadapp kd240-gpio-i2c
```



```
ubuntu@kria:~$ sudo xmutil loadapp kd240-gpio-i2c
[  141.337484] OF: overlay: WARNING: memory leak will occur if overlay removed, property: /fpga-full/firmware-name
[  141.347670] OF: overlay: WARNING: memory leak will occur if overlay removed, property: /fpga-full/resets
[  141.357614] OF: overlay: WARNING: memory leak will occur if overlay removed, property: /__symbols__/afi0
[  141.367136] OF: overlay: WARNING: memory leak will occur if overlay removed, property: /__symbols__/clocking0
[  141.377081] OF: overlay: WARNING: memory leak will occur if overlay removed, property: /__symbols__/axi_intc_0
[  141.387107] OF: overlay: WARNING: memory leak will occur if overlay removed, property: /__symbols__/axi_intc_1
[  141.397141] OF: overlay: WARNING: memory leak will occur if overlay removed, property: /__symbols__/axi_gpio_0
[  141.407176] OF: overlay: WARNING: memory leak will occur if overlay removed, property: /__symbols__/axi_iic_0
kd240-gpio-i2c: loaded to slot 0
```

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

```
sudo i2cdetect -l
```

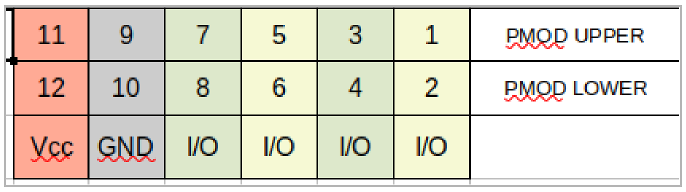


```
ubuntu@kria:~$ sudo i2cdetect -l
i2c-1   i2c          Cadence I2C at ff030000          I2C adapter
i2c-2   i2c          xiic-i2c 80010000.i2c            I2C adapter
ubuntu@kria:~$
```

`i2c-2` 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-> 3 - I2C SCLK
PMOD1-> 1 - I2C SDA
PMOD1-> GND - BMP180 GND
PMOD1->Vcc - BMP180 Vcc

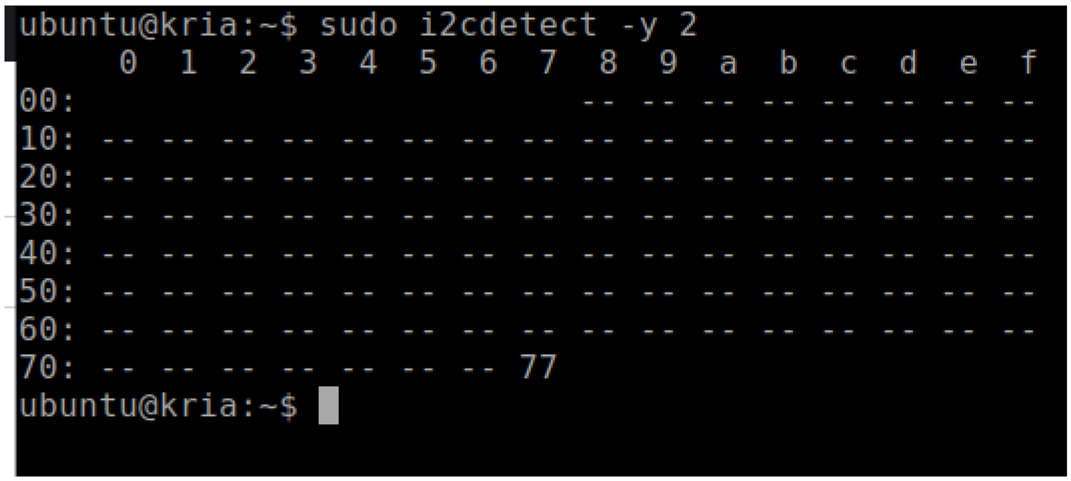


11	9	7	5	3	1	PMOD UPPER
12	10	8	6	4	2	PMOD LOWER
Vcc	GND	I/O	I/O	I/O	I/O	

PMOD port numbering

After connecting BMP180 sensor to KD240 PMOD port, use i2c utility tools to scan for the available devices in i2c-8 channel.

```
sudo i2cdetect -y 2
```



```
ubuntu@kria:~$ sudo i2cdetect -y 2
     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.

Setting up AWS IoT Greengrass Core Device

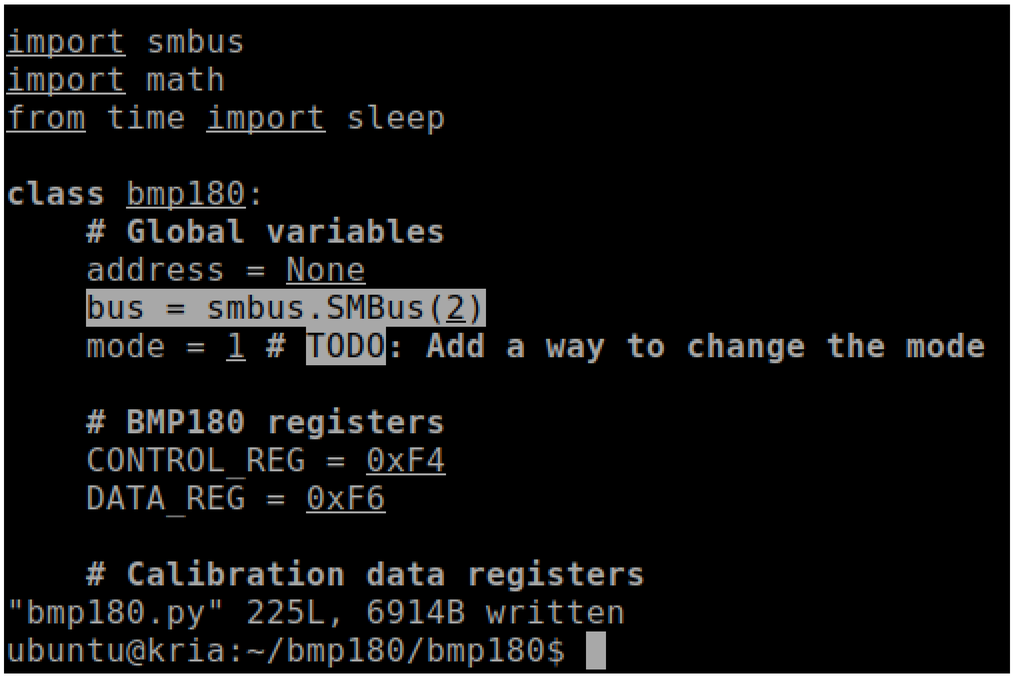
Follow these steps after installing AWS greengrass core device in the KD240 board as mentioned in Kria connect to AWS IoT - GPIO document.

Installing the component

First install the BMP180 python library by running the following commands in KD240 terminal:

```
git clone https://github.com/m-rtijn/bmp180
cd bmp180
```

Update the ~/bmp180/bmp180/bmp180.py to use i2c-3 channel by changing following lines:



```
import smbus
import math
from time import sleep

class bmp180:
    # Global variables
    address = None
    bus = smbus.SMBus(2)
    mode = 1 # TODO: Add a way to change the mode

    # BMP180 registers
    CONTROL_REG = 0xF4
    DATA_REG = 0xF6

    # Calibration data registers
"bmp180.py" 225L, 6914B written
ubuntu@kria:~/bmp180/bmp180$
```

Install the bmp180 module by running:

```
sudo python3 setup.py install
```

Get the `components` folder and copy in the KD240 home directory.
It contains:
artifacts
- com.example.bmp180
    - 1.0.0
        - mqtt.py (This python code publishes temperature data)

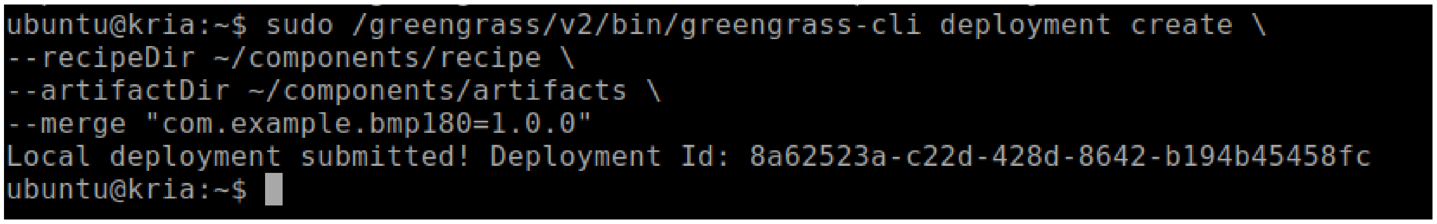
recipe
- com.example.bmp180-1.0.0.json

For installing the greengrass-cli core device, first bring the AWS_ACCESS_KEY_ID and AWS_SECRET_ACCESS_KEY for the device user to the terminal environment.

```
export AWS_ACCESS_KEY_ID=<AWS ACCESS KEY ID>
export AWS_SECRET_ACCESS_KEY=<AWS SECRET ACCESS KEY>
```

To install the above component run the following in the KD240 terminal:

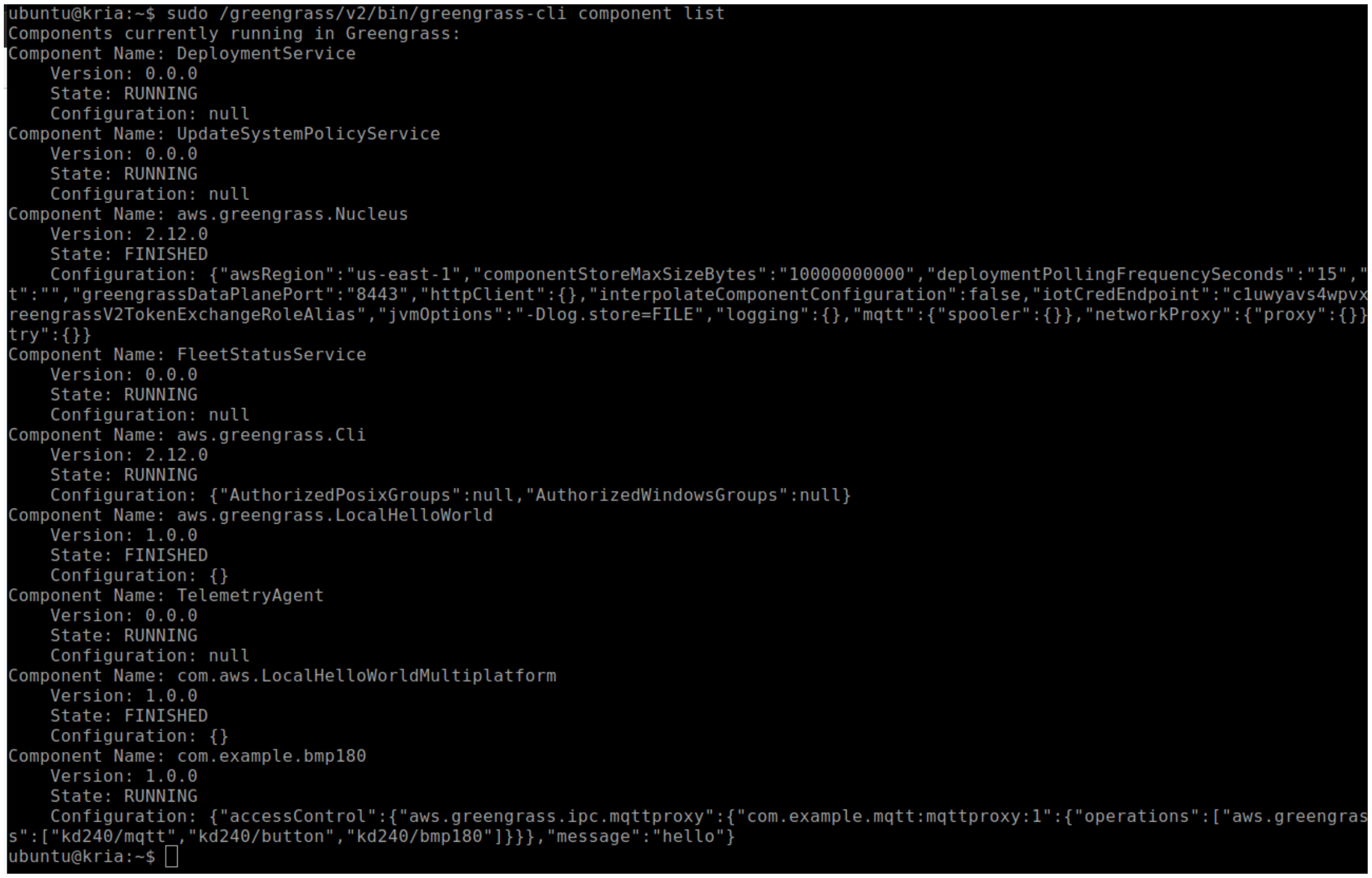
```
sudo /greengrass/v2/bin/greengrass-cli deployment create \
--recipeDir ~/components/recipe \
--artifactDir ~/components/artifacts \
--merge "com.example.bmp180=1.0.0"
```



```
ubuntu@kria:~$ sudo /greengrass/v2/bin/greengrass-cli deployment create \
--recipeDir ~/components/recipe \
--artifactDir ~/components/artifacts \
--merge "com.example.bmp180=1.0.0"
Local deployment submitted! Deployment Id: 8a62523a-c22d-428d-8642-b194b45458fc
ubuntu@kria:~$ 
```

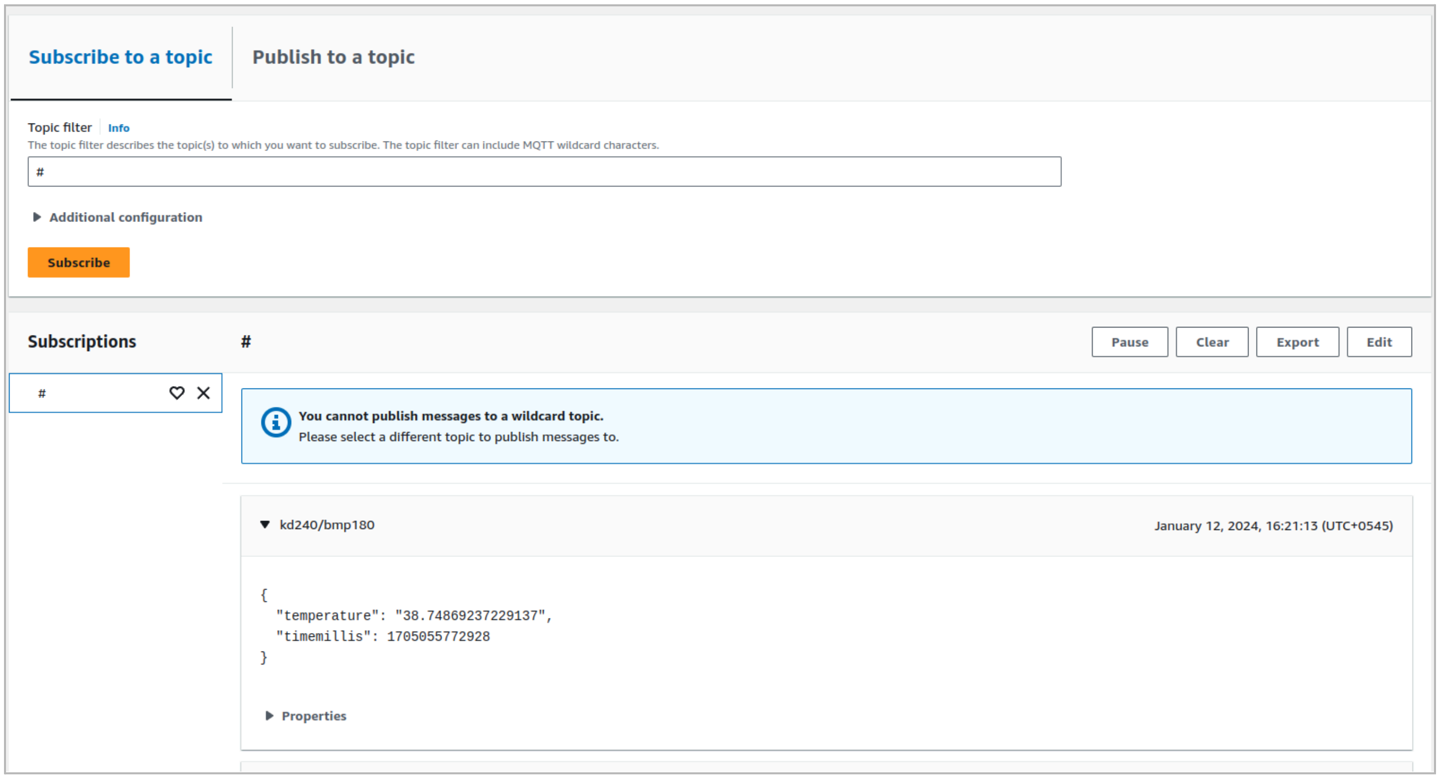
Now check the installed component is in "running state"

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



```
ubuntu@kria:~$ sudo /greengrass/v2/bin/greengrass-cli component list
Components currently running in Greengrass:
Component Name: DeploymentService
    Version: 0.0.0
    State: RUNNING
    Configuration: null
Component Name: UpdateSystemPolicyService
    Version: 0.0.0
    State: RUNNING
    Configuration: null
Component Name: aws.greengrass.Nucleus
    Version: 2.12.0
    State: FINISHED
    Configuration: {"awsRegion":"us-east-1","componentStoreMaxSizeBytes":"10000000000","deploymentPollingFrequencySeconds":"15","
t":"","greengrassDataPlanePort":"8443","httpClient":{},"interpolateComponentConfiguration":false,"iotCredEndpoint":"c1uwyavs4wpvx
reengrassV2TokenExchangeRoleAlias","jvmOptions":"-Dlog.store=FILE","logging":{},"mqtt":{"spooler":{}},"networkProxy":{"proxy":{}}
try":{}}
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.LocalHelloWorld
    Version: 1.0.0
    State: FINISHED
    Configuration: {}
Component Name: TelemetryAgent
    Version: 0.0.0
    State: RUNNING
    Configuration: null
Component Name: com.aws.LocalHelloWorldMultiplatform
    Version: 1.0.0
    State: FINISHED
    Configuration: {}
Component Name: com.example.bmp180
    Version: 1.0.0
    State: RUNNING
    Configuration: {"accessControl":{"aws.greengrass.ipc.mqttproxy":{"com.example.mqtt:mqttproxy:1":{"operations":["aws.greengras
s":["kd240/mqtt","kd240/button","kd240/bmp180"]}}},"message":"hello"}
ubuntu@kria:~$ 
```

Now in aws IoT console, open "MQTT test client" and subscribe to "#"
You will find the published message in the `kd240/bmp180` topic.



The topic filter describes the topic(s) to which you want to subscribe. The topic filter can include MQTT wildcard characters.

Topic filter: #

Subscriptions: #

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

kd240/bmp180 — January 12, 2024, 16:21:13 (UTC+0545)

```
{
  "temperature": "38.74869237229137",
  "timemillis": 1705055772928
}
```

Now we can collect the sensor data into the database and also create logic to trigger actions on the basis of sensor data.

***