Application Note

Setting up an srsRAN 5G Testbed with the SC2430 and the USRP X410

Number: SCT-AN5A21
Date: 2025-04-03

SIGNAL CRAFT TECHNOLOGIES

6815 – 8 Street NE, Suite 295
Calgary, AB T2E7H7 Canada
www.signalcraft.com

Contents

1. Introduction

The global telecommunications industry is undergoing a shift towards open, interoperable, and innovation-enabling technologies. One example of this shift is the creation of Open RAN (Radio Access Networks), a standard that defines ways in which different vendors' equipment works together seamlessly to produce a 5G network, breaking the traditional model of relying on a single vendor for all network components. The advancement of Open RAN has coincided with the emergence of open-source 5G, which leverages commercial off-the-shelf (COTS) radio hardware combined with open-source software stacks to produce a working base station, 5G core, and network.

Open-source 5G is a disruptive technology that reduces costs, accelerates innovation, and unlocks greater flexibility. It allows smaller operators, researchers, and enterprises to build, customize, and optimize their own 5G networks without the constraints of proprietary technology. The telecommunications industry values open-source 5G because it offers a cost-effective means to experiment, scale, and evolve networks efficiently, fostering rapid innovation without the financial burdens typically associated with proprietary solutions.

By reducing vendor lock-in, open-source 5G is a natural enabler of Open RAN, as both share the goal of flexibility and interoperability. Together, Open RAN and open-source 5G create an environment ripe for innovation, allowing network operators to build efficient, agile systems tailored to their unique needs.

In this guide, we present a step-by-step procedure for building an interference-resilient open-source 5G base station testbed with srsRAN, a USRP X410, and a SignalCraft SC2430 Module. We also provide a "ready-to-go" docker container that includes all gNodeB software components pre-configured to expedite deployment.

1.1 About srsRAN, USRP, and the SC2430

srsRAN, by Software Radio Systems, is an open-source software suite for building a mobile network (4G or 5G) with commodity computer hardware and COTS Software-Defined Radio (SDR) products. The srsRAN 5G project offers all the components needed to produce a 5G RAN (gNodeB) that is compliant with 3GPP and O-RAN Alliance specifications. Its full L1/2/3 stack is minimally dependent on external factors, ensuring portability across architectures.

The SC2430 NR Signal Conditioning Module (SCM) by SignalCraft Technologies is a front-end solution that offers signal conditioning and amplification for software-defined radio (SDR) systems. Designed specifically for use with the NI (Emerson) Ettus-USRP X410, it ensures that its input and output radio characteristics meet specific 3GPP 5G/NR standards, making it suitable for 5G NR User Equipment (UE) and gNodeB (gNB) implementations. The SC2430 is able to provide range to the X410 and enable it to operate in congested environments. When coupled with additional amplifier modules, it can achieve distances of up to 1 km, environment permitting.

The USRP X410, by Ettus Research, is a high-performance SDR designed for advanced wireless research and development. Featuring up to 400 MHz of instantaneous bandwidth per channel and support for multiple-input, multiple-output (MIMO) configurations, its robust hardware is well-suited for testing and deploying advanced 5G features. Fully compatible with open-source platforms, it offers the flexibility for developers to experiment and optimize network performance in real-world environments, making it a powerful tool for researchers and network operators building open-source 5G networks and pushing the boundaries of 5G innovation.

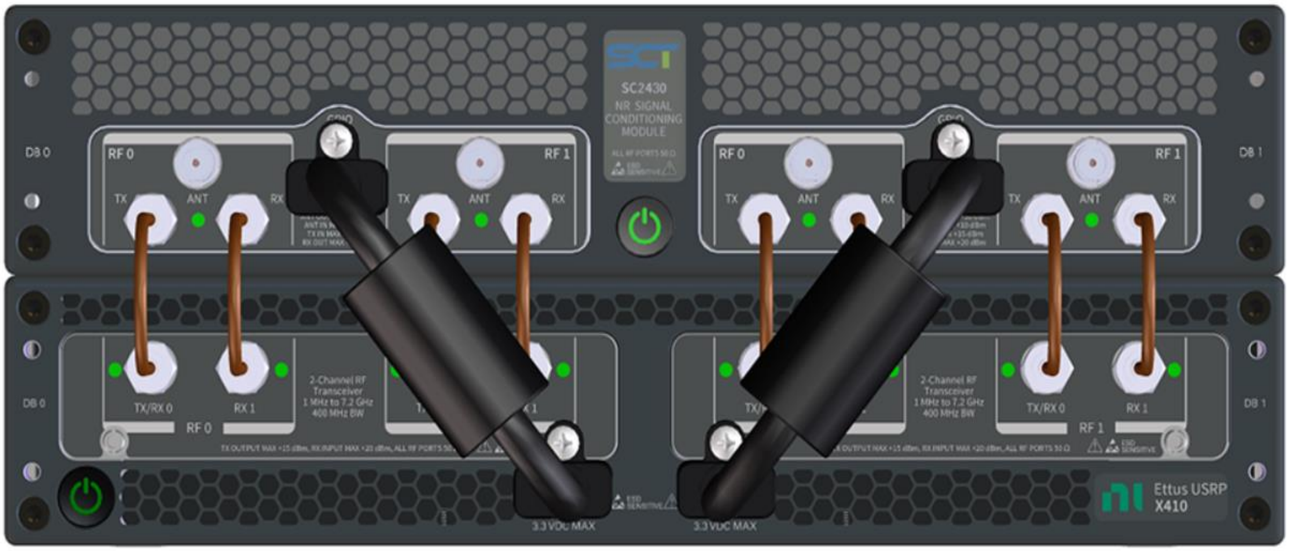

SC2430
NR SIGNAL CONDITIONING MODULE
DB 0
RF 0
TX
ANT
RX
RF 1
DB 1
TX/RX 0
RX 1
2-Channel RF Transceiver 1 MHz to 7.2 GHz 400 MHz BW
3.3 VDC MAX
Ettus USRP X410

Figure 1 - SCT2430 Signal Conditioning Module and Ettus USRP X410

These three products together can produce a 5G testbed that works with COTS handsets such as consumer cellphones, 5G modems, and 5G add-on modules to devices such as the Raspberry Pi. It provides a platform for advanced 5G development and is suitable for private 5G network deployments that require enhanced control of the 5G stack's design.

Qoherent has a product based on these components, called the RIA RAN gNodeB, which combines high-performance open-source 5G systems with a deep learning inference engine designed for performing radiofrequency signal processing tasks with very low latencies. This special deployment enables the base station to be improved with ML, be used for spectrum monitoring, or to be used for sensing.

2. Hardware and Software Prerequisites

The following prerequisites are required to complete the instructions in this guide.

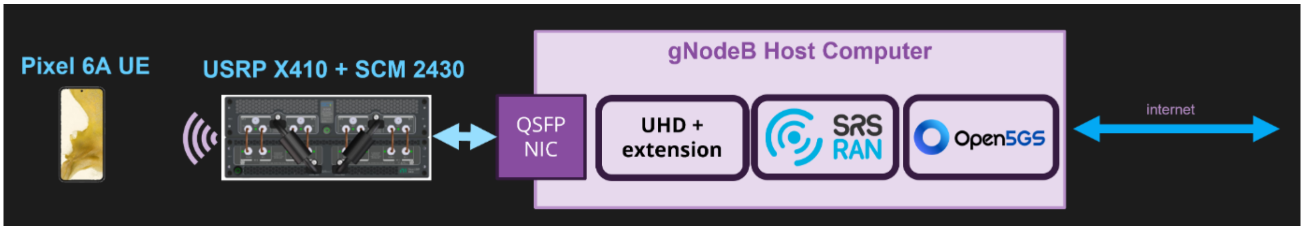


Figure 2 - System Prerequisites

2.1 Hardware

- A reasonably powerful host computer[1] (we recommend at least a 10th-generation Intel i7 or 3rd-generation Ryzen 7 with at least 16 GB of RAM).
- A network interface card that supports QSFP[2]
- A USRP X410 connected to the host computer[3]
- An SC2430 NR Signal Conditioning Module
- Appropriate antennae connected to the SC2430's Tx and Rx antenna ports
- Appropriate SMA and GPIO cables between the SC2430 and the USRP
- A Test SIM card and SIM Programmer. We recommend Open-Cells.com.
- A 5G mobile phone (we recommend the Google Pixel 6A). Other COTS options here.

2.2 Software (Installed)

- Ubuntu 22.04 on the host computer.
- Docker and docker-compose for seamless installation. (Alternative, step-by-step instructions are available from Qoherent here: https://qoherent.ai/blog/2409-srs-sc2430-x410-guide/.)

2.3 Software (Contained within docker image)

- Universal Hardware Driver (UHD) for the USRP
- UHD SCM Driver Extension to control the configuration of the SC2430
- srsRAN
- Open5GS - Open5GS is an open-source 5G core we use in this guide.

---

[1] The computer impacts gNodeB performance and configuration, including the bandwidth it can support.
[2] SFP+ NICs are compatible with an appropriate adapter. Please see the Ettus support page.
[3] Other USRPs such as the B210 or X310 may be used with some extra configuration and timing controls.

2.4 Software (To be installed using the instructions in this guide)

- UICC/SIM SIM Card Programming Software from Open-Cells
- 5G Switch App to enforce 5G mode on the UE

2.5 Other

- A valid license from the relevant regulatory authority is required for any over-the-air RF transmissions. Note that this guide assumes that you have obtained the necessary license and fulfilled all related requirements.
- An internet connection is required for the host machine to complete speed testing and benchmarks.

3. Docker Image Installation

An alternative step-by-step installation procedure for all the software provided by the docker image can be found here: https://qoherent.ai/blog/2409-srs-sc2430-x410-guide/.

To simplify the installation procedure, Qoherent provides a docker image that contains all the necessary components. Docker offers a more direct and portable way of installing and running the software. An image containing all the required software—UHD, SC2430, Open5gs, srsRAN, send_0-mq.py—is offered on docker hub at: https://hub.docker.com/r/qoherent/srsran_scm.

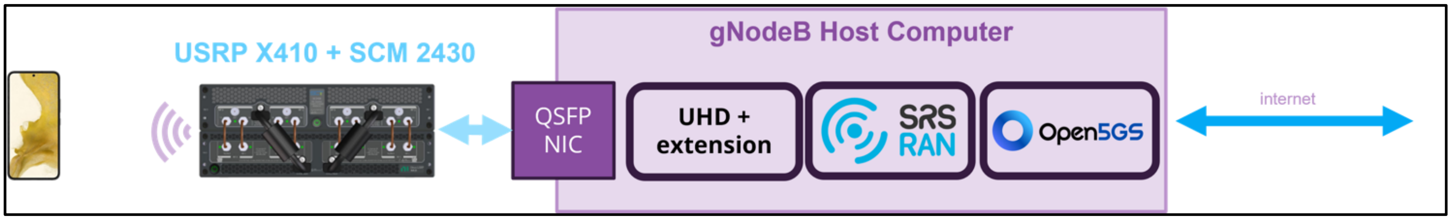


Figure 3 - Necessary Components to run testbed

To get started:

Create volumes for data persistency:

```
sudo docker volume create mongodb_data # Volume for database

sudo docker volume create configs      # Volume for the open5gs configs

sudo docker volume create config       # Volume for the gnb config
```

Download the docker image and run the docker container:

```
sudo docker run -it --privileged --network=host -v configs:/etc/open5gs/ -v
mongodb_data:/var/lib/mongodb -v config:/srsRAN_Project/configs qoherent/srsran_scm
```

Update the amf.yaml and upf.yaml configuration files in the container.

```
nano /etc/open5gs/amf.yaml

nano /etc/open5gs/umf.yaml
```

Open another terminal and from the host run the userplane configuration, IP Masquerading and route to the internet commands.

```
sudo ip tuntap add name ogstun mode tun

sudo ip addr add 10.45.0.1/16 dev ogstun

sudo ip link set ogstun up
sudo sysctl -w net.ipv4.ip_forward=1

### Add NAT Rule

sudo iptables -t nat -A POSTROUTING -s 10.45.0.0/16 ! -o ogstun -j MASQUERADE

sudo ufw disable
```

Switch back to the docker terminal and start the core services by starting the database and running the setup scripts.

```
systemctl start mongod

./etc/open5gs/setup.sh
```

4. Register Subscriber Information

Connect to http://127.0.1.1:9999/ and log in with the admin account.

Tip!

✓ Qoherent srsRAN-SCM docker default credentials:

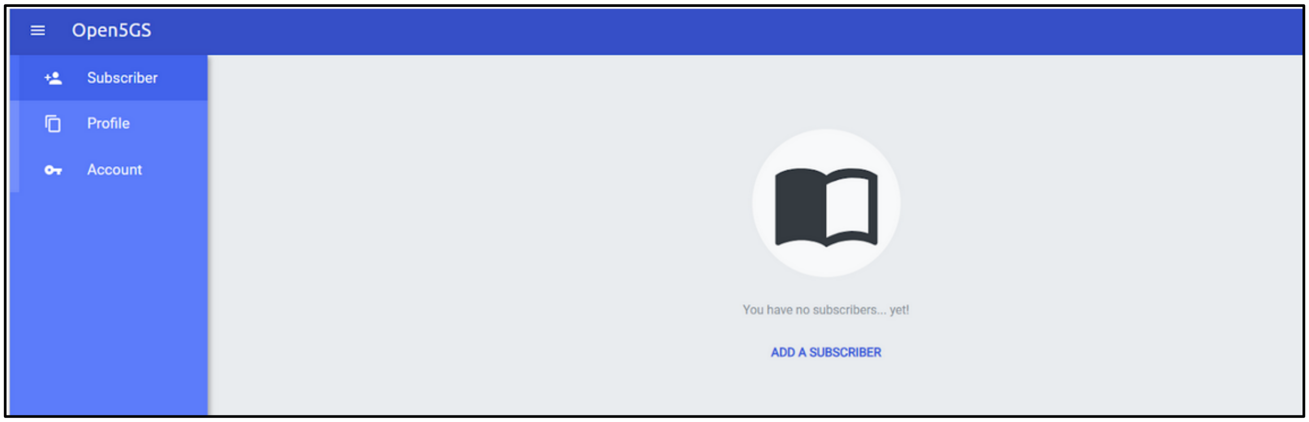
Username: admin

Password: 1423

Note: You can change the password in the Account menu.

To add subscriber information, perform the following WebUI operations:

1. Go to the Subscriber menu.
2. Click **+** to add a new subscriber. Fill the IMSI, security context (K, OPc, AMF), and APN of the subscriber.
3. Click SAVE.



You have no subscribers... yet!

Figure 4 - Open5GS Menus

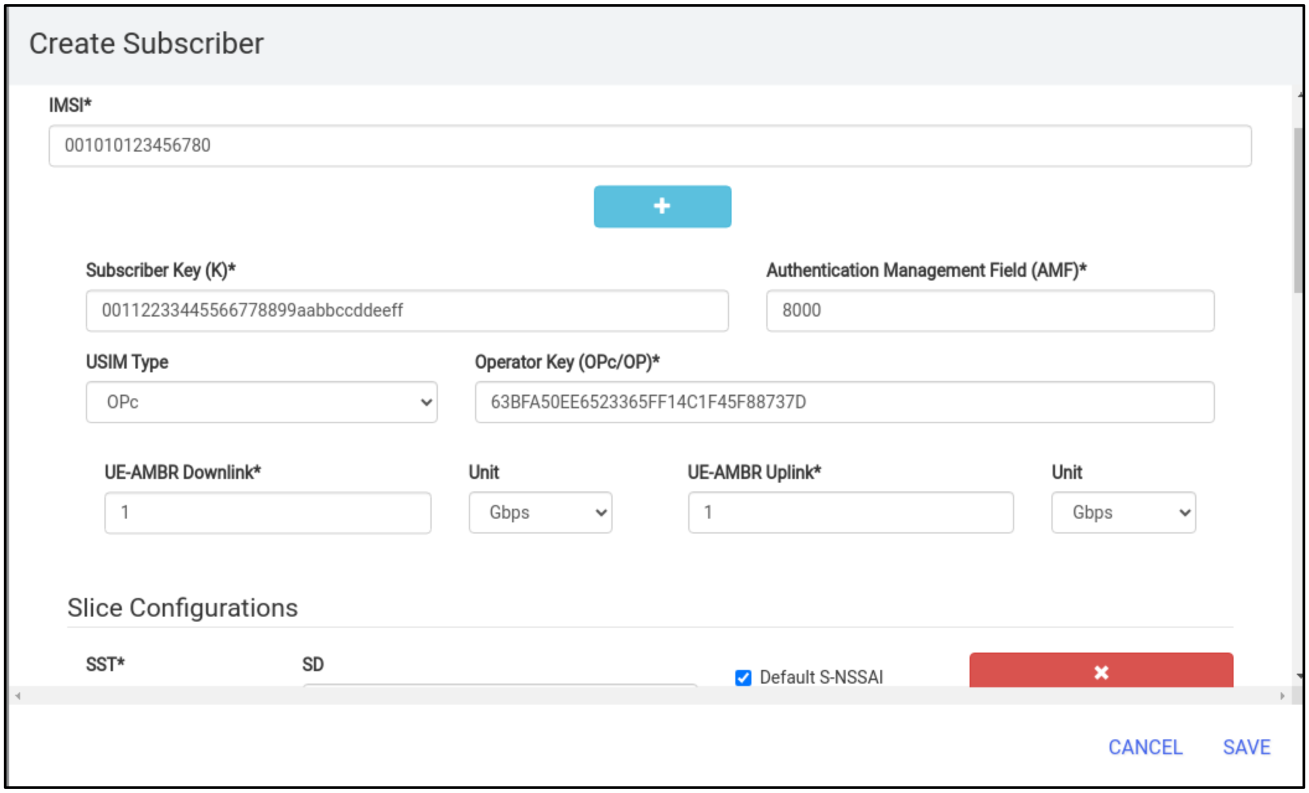
4.1.1 Adding UE Subscriber Info

The following need to match the configurations added to the SIM card (see later sections). Specifically, the IMSI , Subscriber Key , and the Operator Key need to match the UE.

Info!

Examples used in this guide:

- Operator Key (opc)= 63bfa50ee6523365ff14c1f45f88737d
- Subscriber Key (k) = 00112233445566778899aabbccddeeff
- IMSI (imsi) = 001010123456780
- DDN/APN = srsapn and Type = IPv4



Create Subscriber

IMSI*: 001010123456780

Subscriber Key (K)*: 00112233445566778899aabbccddeeff

Authentication Management Field (AMF)*: 8000

USIM Type: OPc

Operator Key (OPc/OP)*: 63BFA50EE6523365FF14C1F45F88737D

UE-AMBR Downlink*: 1 Unit: Gbps

UE-AMBR Uplink*: 1 Unit: Gbps

Slice Configurations

SST* SD

☑ Default S-NSSAI

Figure 5 - Create Subscriber Fields

Once the core network is ready, the RAN (gNodeB) may be configured and set up.

5. Setting up the COTS UE

Important links for working with a COTS handset:

- SRS Guide on working with a COTS UE
- Overview of the UICC SIM Card: https://open-cells.com/index.php/uicc-tutorial/
- Official steps to program: https://open-cells.com/index.php/uiccsim-programing/
- FAQ: https://open-cells.com/index.php/uicc-faq/
- Open-Cells Q/A: https://open-cells.com/index.php/2017/06/08/support-page/

5.1 SIM Programming

The MMC, MNC, IMSI and other credentials in the ISIM can be set by reprogramming. The tool used is Open Cells project. The steps are described below.

Download and install the UICC programming application:

```
wget https://open-cells.com/d5138782a8739209ec5760865b1e53b0/uicc-v3.2.tgz

tar -xzf uicc-v3.2.tgz

cd uicc-v3.2

make
```

* This guide uses v3.2, but v3.3 is available at the time of publication.

Insert the card in the reader and the reader in a USB socket as:

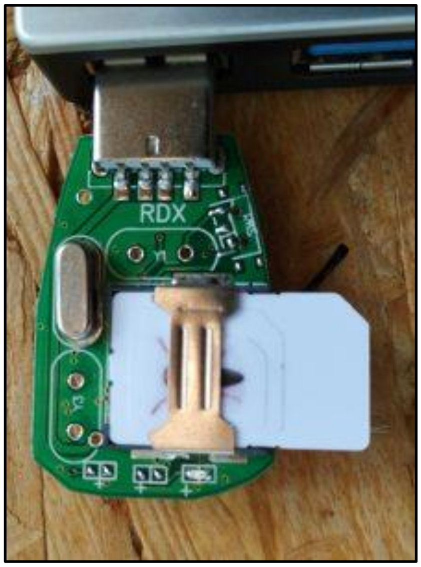


Figure 6 - How to Insert Card in Reader and Reader in a USB Socket

To read the existing basic data on the card:

```
sudo ./program_uicc
```

To see the parameters that may be set up with program_uicc:

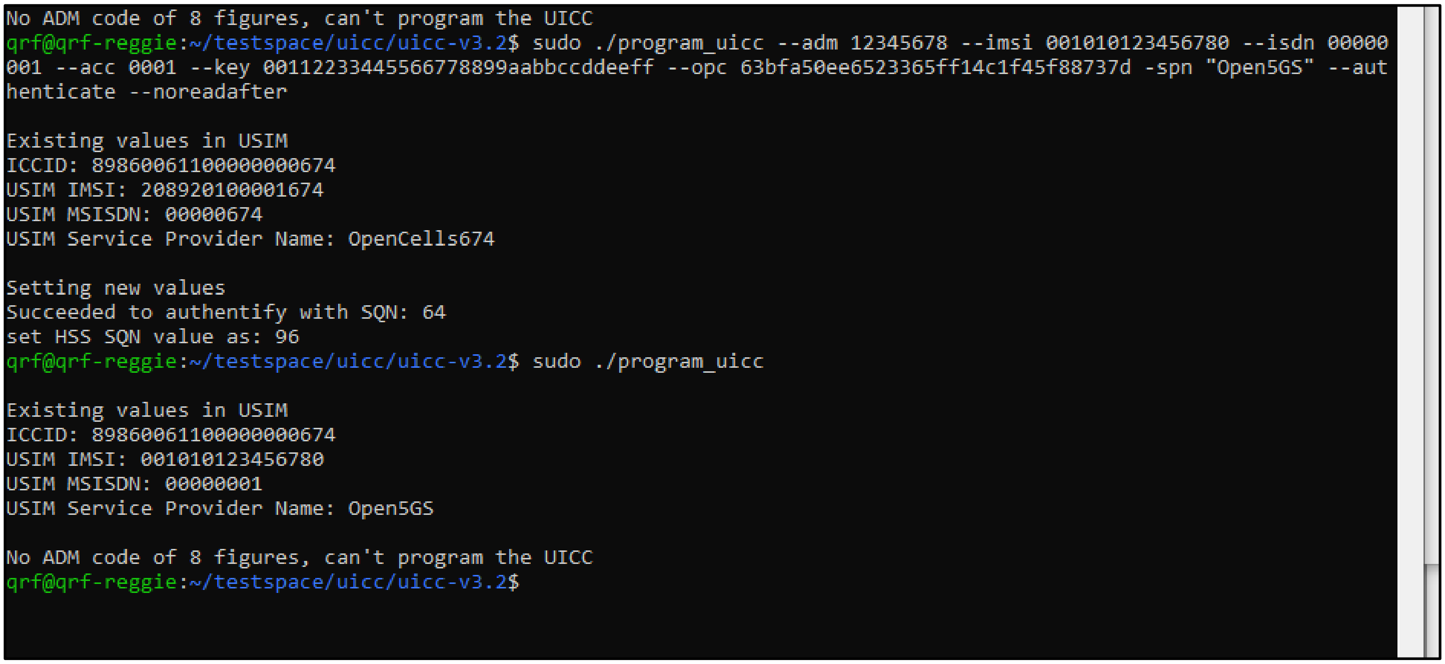
- --adm : This sets the Administrative Code (ADM). It is used for securing operations related to SIM card management and maintenance. The ADM key is required for certain operations like updating encryption keys on the SIM.
- --imsi : This sets the International Mobile Subscriber Identity (IMSI). It is a unique number associated with all GSM and Universal Mobile Telecommunications System (UMTS) network mobile phone users. The IMSI is stored as a 64-bit field in the SIM inside the phone and is sent by the phone to the network.
- --isdn : This sets the ISDN number (also known as MSISDN), which is the actual phone number assigned to the user.
- --acc : This sets the Access Control Class (ACC), which is used to control network access in high-demand situations.
- --key : This sets the Ki (Key), which is a 128-bit value used in authenticating the SIMs on the network. It is also used to encrypt calls.
- --opc : This sets the OPC (Operator Code), which is derived from Ki and is also used in cryptographic processes.
- --spn : This sets the Service Provider Name (SPN), which is a network parameter that can be set on the SIM card.
- --authenticate : This is a flag to indicate that authentication should be performed.
- --noreadafter : This is likely a flag to indicate that the SIM should not be read after some operation is completed. The exact functionality would depend on the implementation in program_uicc .

To match the core network, UE information needs to align the opc, k and imsi parameters as below:

- opc = 63bfa50ee6523365ff14c1f45f88737d
- k = 00112233445566778899aabbccddeeff
- imsi = 001010123456780

The command to use is:

```
sudo ./program_uicc --adm 12345678 --imsi 001010123456780 --isdn 00000001 --acc 0001 --key
00112233445566778899aabbccddeeff --opc 63bfa50ee6523365ff14c1f45f88737d -spn "Open5GS" --
authenticate --noreadafter
```



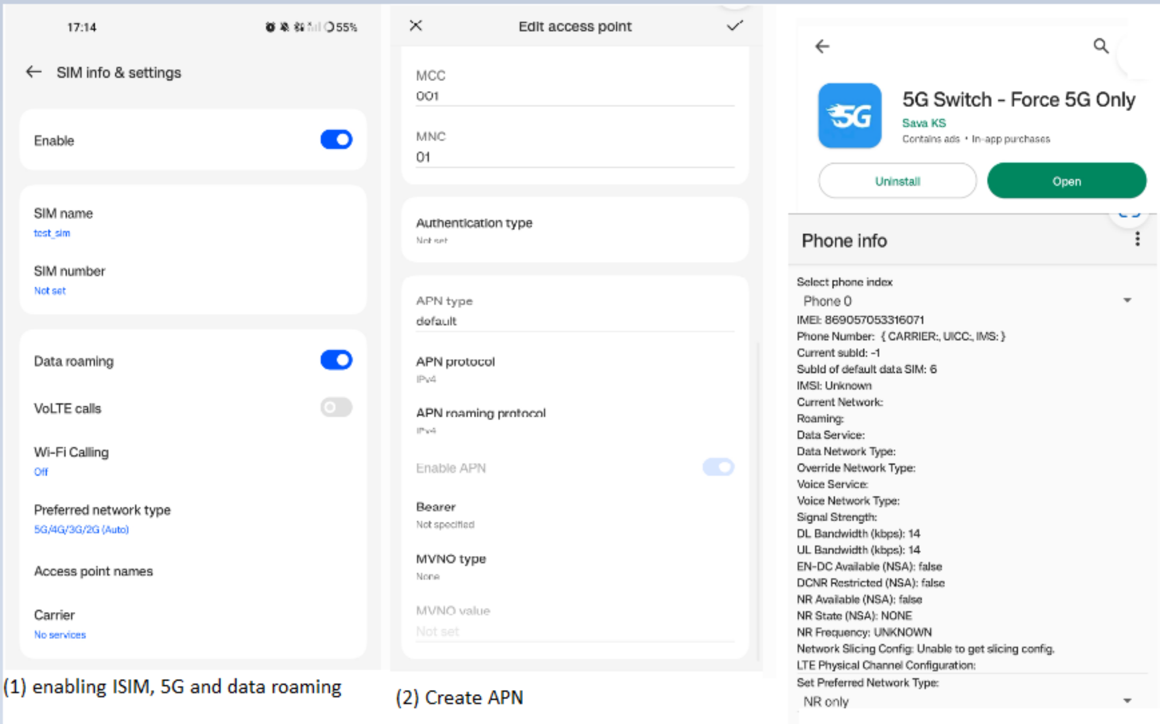
```
No ADM code of 8 figures, can't program the UICC
qrf@qrf-reggie:~/testspace/uicc/uicc-v3.2$ sudo ./program_uicc --adm 12345678 --imsi 001010123456780 --isdn 00000
001 --acc 0001 --key 00112233445566778899aabbccddeeff --opc 63bfa50ee6523365ff14c1f45f88737d -spn "Open5GS" --aut
henticate --noreadafter

Existing values in USIM
ICCID: 89860061100000000674
USIM IMSI: 208920100001674
USIM MSISDN: 00000674
USIM Service Provider Name: OpenCells674

Setting new values
Succeeded to authentify with SQN: 64
set HSS SQN value as: 96
qrf@qrf-reggie:~/testspace/uicc/uicc-v3.2$ sudo ./program_uicc

Existing values in USIM
ICCID: 89860061100000000674
USIM IMSI: 001010123456780
USIM MSISDN: 00000001
USIM Service Provider Name: Open5GS

No ADM code of 8 figures, can't program the UICC
qrf@qrf-reggie:~/testspace/uicc/uicc-v3.2$
```

Figure 7 - UICC Code

After that, insert the USIM into the test UE and follow these steps from Settings, Network and Internet, SIMs.

Steps to configure:

1. Enable ISIM, 5G, and Data Roaming.
2. Name the SIM something like “test_sim”.
3. Create an APN with settings that match the core network and the gNodeB.
    a. MNC and MCC should match the gNodeB (In this case MNC = 01 , MCC = 001) which come from the PLMN configuration MNC:MCC.
4. Download then enable the 5G Switch app to force the UE to 5G only.



17:14 55%

SIM info & settings

Enable

SIM name
test_sim

SIM number
Not set

Data roaming

VoLTE calls

Wi-Fi Calling
Off

Preferred network type
5G/4G/3G/2G (Auto)

Access point names

Carrier
No services

(1) enabling ISIM, 5G and data roaming

Edit access point

MCC
001

MNC
01

Authentication type
Not set

APN type
default

APN protocol
IPv4

APN roaming protocol
IPv4

Enable APN

Bearer
Not specified

MVNO type
None

MVNO value
Not set

(2) Create APN

5G Switch - Force 5G Only
Sava KS
Contains ads • In-app purchases

Phone info

Select phone index
Phone 0
IMEI: 869057053316071
Phone Number: { CARRIER:, UICC:, IMS: }
Current subId: -1
SubId of default data SIM: 6
IMSI: Unknown
Current Network:
Roaming:
Data Service:
Data Network Type:
Override Network Type:
Voice Service:
Voice Network Type:
Signal Strength:
DL Bandwidth (kbps): 14
UL Bandwidth (kbps): 14
EN-DC Available (NSA): false
DCNR Restricted (NSA): false
NR Available (NSA): false
NR State (NSA): NONE
NR Frequency: UNKNOWN
Network Slicing Config: Unable to get slicing config.
LTE Physical Channel Configuration:
Set Preferred Network Type:
NR only

(3) Force the Phone to 5G ,ode only

Figure 8 - 5G Switch App

The UE is now ready for testing. The next step is to start up the gNodeB.

6. Getting Started with the gNodeB

6.1 Starting up the gNodeB

Edit the gnb config file (if required) and run the start.sh script that will spawn two tmux sessions; one for the gnb, another for the IQ samples publishing to ZMQ.

```
nano /srsRAN_Project/configs/gnb_ota.yaml

./start.sh

tmux ls
```

6.2 Example Tests for Confirming Operation

If the connection to the core is successful, you should see the following from the AMF log:

```
04/03 13:25:13.469: [amf] INFO: gNB-N2 accepted[127.0.0.1]:47633 in ng-path module
(../src/amf/ngap-sctp.c:113)

04/03 13:25:13.469: [amf] INFO: gNB-N2 accepted[127.0.0.1] in master_sm module
(../src/amf/amf-sm.c:706)

04/03 13:25:13.469: [amf] INFO: [Added] Number of gNBs is now 1
(../src/amf/context.c:1034)
```

The COTS UE can now search for the network. Select the carrier for the network based on the network name you provided; the UE should then attach to the network.

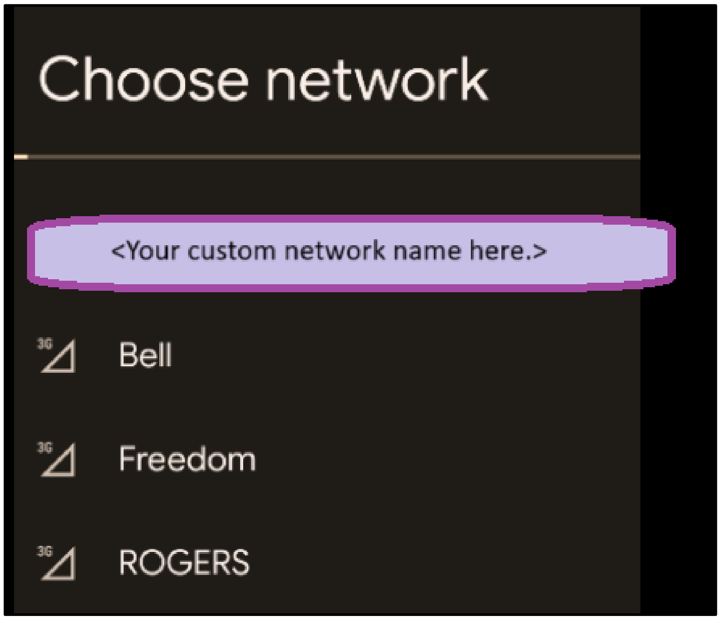


Figure 9 - Choose Network

The UE will show “5G” in the upper right corner:

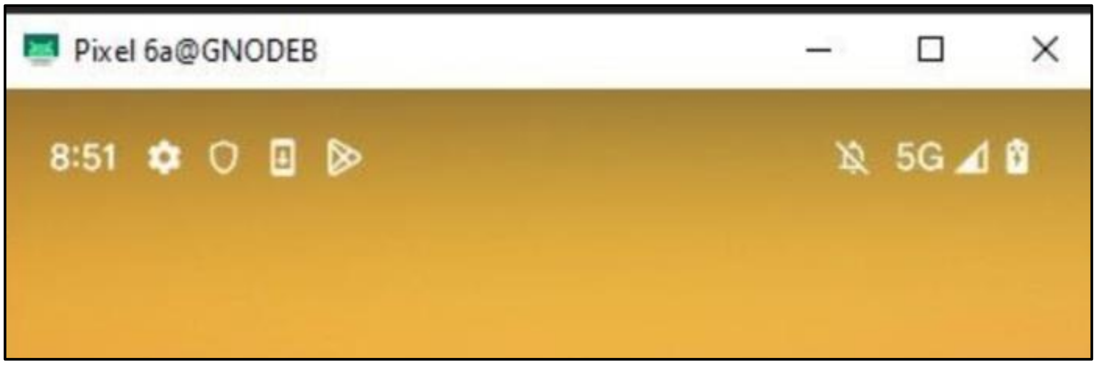


Figure 10 - UE showing as 5G in corner

To confirm that the attachment operation is successful, both the AMF log and gNB console output can be monitored. The AMF log should look similar to the following:

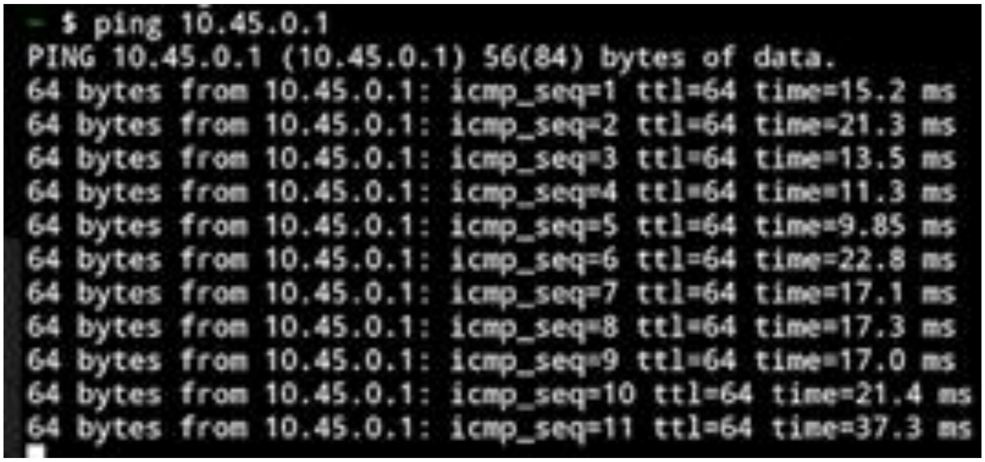
```
04/27 13:16:31.746: [amf] INFO: InitialUEMessage (../src/amf/ngap-handler.c:361)

04/27 13:16:31.746: [amf] INFO: [Added] Number of gNB-UEs is now 1
(../src/amf/context.c:2036)

04/27 13:16:31.746: [amf] INFO:     RAN_UE_NGAP_ID[0] AMF_UE_NGAP_ID[78] TAC[7]
CellID[0x0] (../src/amf/ngap-handler.c:497)
```

```
04/27 13:16:31.746: [amf] INFO: [suci-0-001-01-0-0-0-0000068960] Known UE by 5G-
S_TMSI[AMF_ID:0x20040,M_TMSI:0xdd00ff1a] (../src/amf/context.c:1402)

04/27 13:16:31.746: [gmm] INFO: Registration request (../src/amf/gmm-sm.c:134)

04/27 13:16:31.746: [gmm] INFO: [suci-0-001-01-0-0-0-0000068960]    5G-
S_GUTI[AMF_ID:0x20040,M_TMSI:0xdd00ff1a] (../src/amf/gmm-handler.c:169)

04/27 13:16:31.913: [gmm] INFO: [imsi-001010000068960] Registration complete
(../src/amf/gmm-sm.c:1063)

04/27 13:16:31.913: [amf] INFO: [imsi-001010000068960] Configuration update command
(../src/amf/nas-path.c:389)

04/27 13:16:31.913: [gmm] INFO:     UTC [2023-04-27T13:16:31] Timezone[0]/DST[0]
(../src/amf/gmm-build.c:502)

04/27 13:16:31.913: [gmm] INFO:     LOCAL [2023-04-27T13:16:31] Timezone[0]/DST[0]
(../src/amf/gmm-build.c:507)

04/27 13:16:32.105: [gmm] INFO: UE SUPI[imsi-001010000068960] DNN[srsapn] S_NSSAI[SST:1
SD:0xffffff] (../src/amf/gmm-handler.c:1042)
```

The gNB trace should show the following after typing “t” in the console to start the trace:

```
               -------------DL----------------|------------------UL--------------------
pci rnti  cqi  mcs  brate   ok  nok  (%) | pusch  mcs  brate   ok  nok  (%)   bsr
  1 4601   15   15   4.3k    7    0   0% |  21.3   23    17k    4    0   0%   0.0
  1 4601   15   27   287k   84    0   0% |  23.1   27   233k   39    0   0%   0.0
  1 4601   15   28   1.2k    1    0   0% |  21.8   28   8.7k    2    0   0%   0.0
  1 4601   15    0      0    0    0   0% |   n/a    0      0    0    0   0%   0.0
  1 4601   15    0      0    0    0   0% |   n/a    0      0    0    0   0%   0.0
  1 4601   15    0      0    0    0   0% |   n/a    0      0    0    0   0%   0.0
  1 4601   12    0      0    0    0   0% |   n/a    0      0    0    0   0%   0.0
  1 4601   15    0      0    0    0   0% |   n/a    0      0    0    0   0%   0.0
  1 4601   15   28    53k   10    0   0% |  24.6   26    55k   32    0   0%   0.0
  1 4601   15   28   7.7k    4    0   0% |  22.7   28    17k    4    0   0%   0.0
  1 4601   15    0      0    0    0   0% |   n/a    0      0    0    0   0%   0.0
```

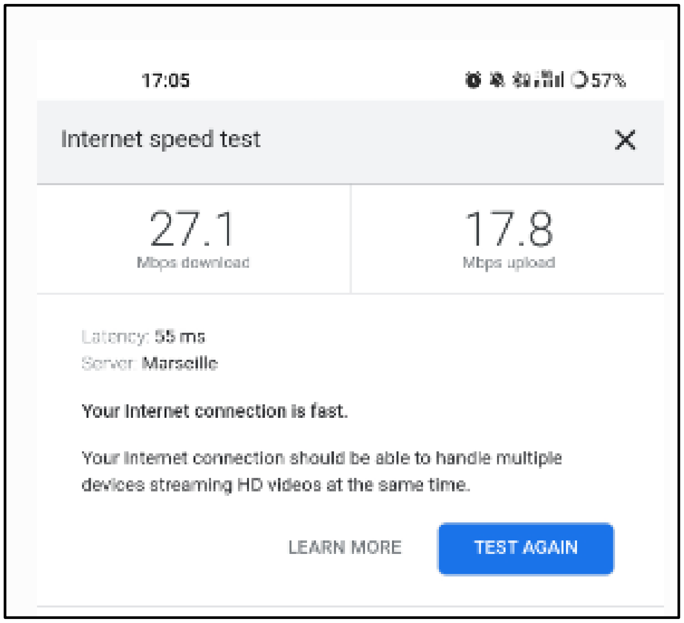
The first test we suggest is a ping test to the core's IP address or to a website on the internet (ensure that WiFi is disabled on the UE).



```
~ $ ping 10.45.0.1
PING 10.45.0.1 (10.45.0.1) 56(84) bytes of data.
64 bytes from 10.45.0.1: icmp_seq=1 ttl=64 time=15.2 ms
64 bytes from 10.45.0.1: icmp_seq=2 ttl=64 time=21.3 ms
64 bytes from 10.45.0.1: icmp_seq=3 ttl=64 time=13.5 ms
64 bytes from 10.45.0.1: icmp_seq=4 ttl=64 time=11.3 ms
64 bytes from 10.45.0.1: icmp_seq=5 ttl=64 time=9.85 ms
64 bytes from 10.45.0.1: icmp_seq=6 ttl=64 time=22.8 ms
64 bytes from 10.45.0.1: icmp_seq=7 ttl=64 time=17.1 ms
64 bytes from 10.45.0.1: icmp_seq=8 ttl=64 time=17.3 ms
64 bytes from 10.45.0.1: icmp_seq=9 ttl=64 time=17.0 ms
64 bytes from 10.45.0.1: icmp_seq=10 ttl=64 time=21.4 ms
64 bytes from 10.45.0.1: icmp_seq=11 ttl=64 time=37.3 ms
```

Figure 11 - Ping Test Example

Speed tests can be run from the UE to verify connection statistics:



27.1
Mbps download

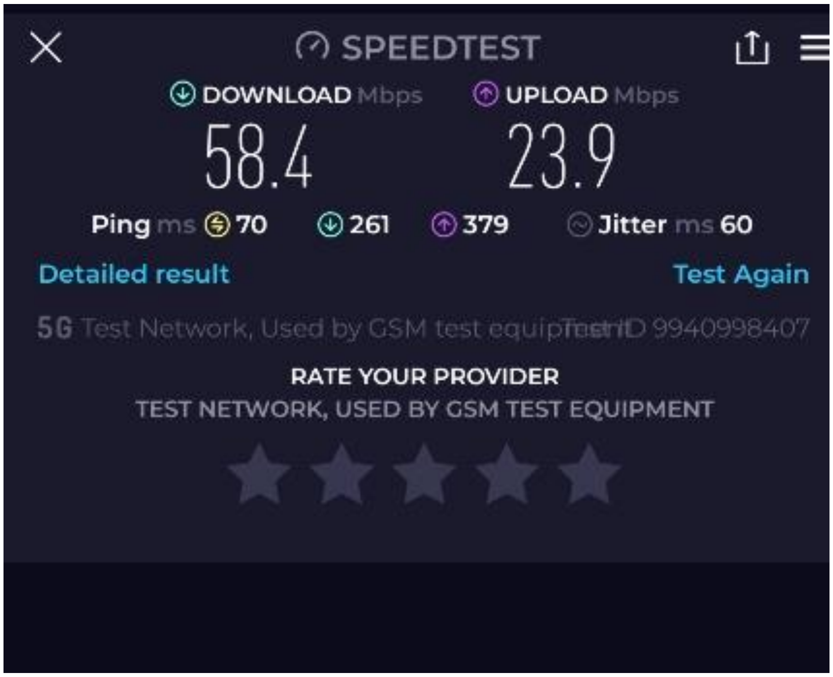
17.8
Mbps upload

Latency: 55 ms
Server: Marseille

Your Internet connection is fast.

Your Internet connection should be able to handle multiple devices streaming HD videos at the same time.

Figure 12 - Example of a Speed Test



SPEEDTEST

DOWNLOAD Mbps 58.4

UPLOAD Mbps 23.9

Ping ms 70 261 379 Jitter ms 60

Detailed result

Test Again

5G Test Network, Used by GSM test equipTest ID 9940998407

RATE YOUR PROVIDER
TEST NETWORK, USED BY GSM TEST EQUIPMENT

Figure 13 - Example of a Speed Test

We also recommend streaming tests such as playing a YouTube video, sharing the screen of the UE via a YouTube video in an app such as Google Meet, or streaming the camera from Google Meet:

Figure 14 - Example of YouTube Video being Shared or Streamed

You will also be able to ambiently observe the signals with another SDR or a spectrum analyzer:

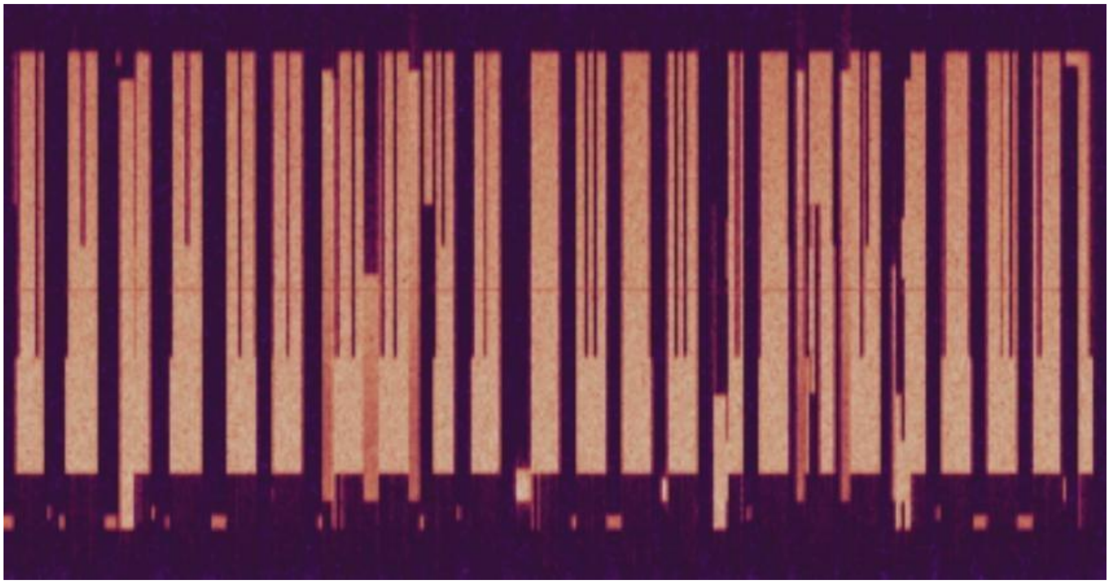


Figure 15 - Example of Signals from an SDR or Spectrum Analyzer

7. RAN Metrics

srsRAN comes with a suite of metrics available via a Grafana interface. To enable the metrics, two steps are required:

1. Deploy the Grafana and metrics servers using the docker images through docker-compose (v2 or later). To launch the docker image for the Grafana UI, run the following command from the main folder containing srsRAN:

```
sudo docker compose -f docker/docker-compose.yml up grafana
```

2. Modify the gNodeB YAML configuration file to indicate the metrics server IP address and port:

```
metrics:

    enable_json_metrics: true            # Enable reporting metrics in JSON format

    addr: 172.19.1.4                     # Metrics-server IP

    port: 55555                          # Metrics-server Port
```

The metrics will now be available at the specified IP address and port in your web browser:

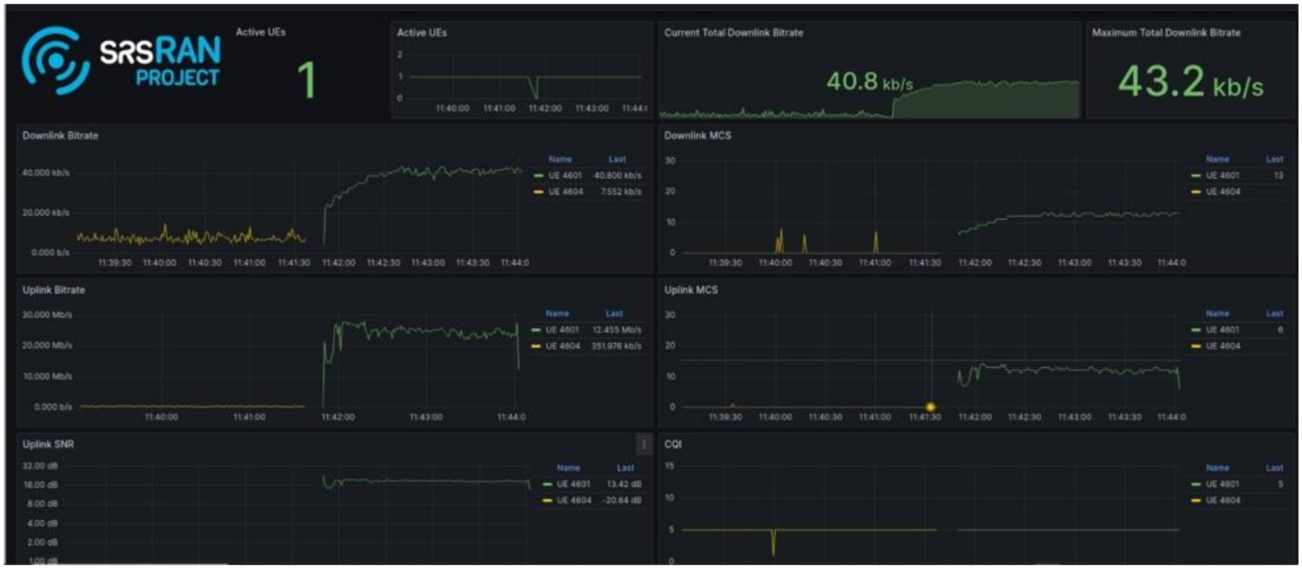


Figure 16 - srsRAN Project Dashboard

8. Conclusion

In this article, we presented a guide for setting up a USRP X410, SC2430, and srsRAN together to create an ORAN-native and split 7-compliant 5G testbed capable of transmission over practical distances in congested RF environments. To simplify the setup, a “ready to deploy” docker container with all components pre-configured was used.

This configuration can be used as a platform for research, for 5G application development, directly as an operating network, and with the help of Qoherent’s RIA RAN product, AI can be integrated for advanced signal processing tasks such as spectrum monitoring, spectrum sharing, or sensing.

For any questions about srsRAN, the X410, the SC2430, or for help setting up your testbed or network, please contact SignalCraft Technologies at support@signalcraft.com.

9. About Us

9.1 Qoherent

Qoherent is an early-stage technology company that specializes in AI development for software-defined radios (SDRs), focusing on streamlining the workflow for radio signal processing engineers and researchers in wireless technologies with software and IP products.

Qoherent helps scientists and engineers explore applications of AI on software-defined radios. We strive to propel intelligent radio innovations through rapid prototyping and design automation tools that are tightly integrated with existing commercial technologies. Focused on the complex spectrum conditions inherent to the space, telecommunications, and defence sectors, we empower our customers to create AI-enabled solutions for more effective sensing and communications systems.

9.2 SignalCraft Technologies

SignalCraft, based in Calgary, Alberta, builds brilliantly designed high-frequency digital and RF products, 100 per cent in-house, from the ground up, to spec and on schedule. SignalCraft is the trusted partner of wireless product teams, from leading global test brands to industrial communications startups. SignalCraft is an authorized reseller of Ettus Research USRP equipment.

9.3 Software Radio Systems

Software Radio Systems, based in Ireland, is the leading provider of deployable and open RAN software for mobile wireless applications. The srsRAN full-stack software solution is portable across hardware platforms, modular, and highly scalable. Developed in-house by the SRS team, srsRAN is available under open-source and commercial licenses.