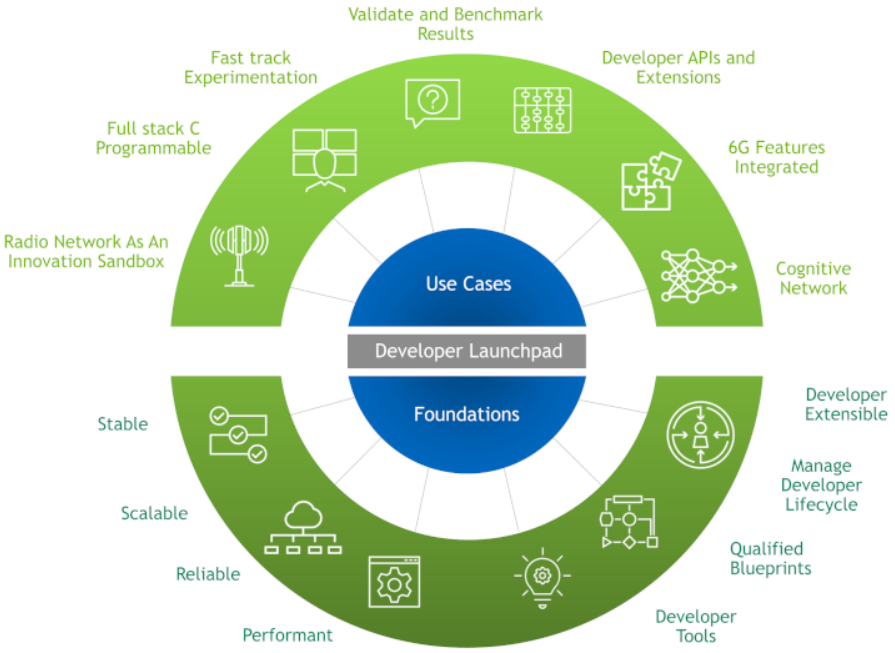
Aerial RAN CoLab Over-the-Air

Release 1.5

NVIDIA Corporation

Feb 12, 2025

Contents



Meet ARC-OTA, the most versatile platform available to all advanced wireless developers.

ARC-OTA is the first, - The FIRST network as an innovation sandbox with complete access to source code to onboard any and all experiments, with quick turnaround validation and benchmarking results. Built on principles of disaggregation, virtualization, software defined, adaptability, ORAN specifications and intelligence - a true advanced wireless developer launchpad.

A radio network built on principles above is non-trivial. Analogous to a puzzle board the need to qualify exact fit amid options is important to you as a developer. To ensure a delightful developer experience the product is built on strong foundations of reliable, stable, performant and scalable experimental radio networks. ARC-OTA roadmap will continue to deliver NVIDIA qualified tools and blueprints many of which will be emerging developer extensions to help accelerate the pace of innovation as a community.

Simply put as an ARC-OTA developer community lets – leverage, extend and innovate!

Latest Product Updates

Product Area	Updates
Release Manifest	ARC-OTA 1.5 is based on ASDK 24-1 + OAI 2024.w21 tag
Product Blueprints	New Product Brief *Blueprints Section*
Platform	This release supports the following platforms: 1. Supermicro GH200 + BF3 2. Gigabyte (A100 + CX6-DX) 3. Dell R750 + A100X
New	**First** release for ARC-OTA on SMC GH platform
DL Peak Performance Improvements	**First** release for ARC-OTA with 1Gig downlink peak performance (open src gNB/CN)
Key Release KPIs	4 layers DL –> peak performance improvement (previously 460 Mbps)
	SMC-GH DL 1.03G / UL 130Mbps, 4DL/1UL + CBRS RU soak testing 10+ hours
	Dell R750 / Gigabyte DL 900 Mbps / UL 110Mbps 4DL/1UL (4 hours)
SDK Manager Network as a service tooling	SDK Manager support for ARM CPU (SMC-GH) and service deployment support for Gigabyte(A100+CX6-DX), Dell(A100X) and SMC-GH(GH200 + BF3) (non-k8)
Other features	Multi-UE qualified OTA (6 UEs)
	Sub-6 N48 CBRS O-RU integrated
	Multi-UE CSI dataset recipe (w/ pyAerial and DataLake integrated)

Developer Contributions and Updates

Developer extensions	*Extensions*
Developer plugins	*Rice CBRS O-RU interop*
	Northeastern secondary 5G core interop
	Sterling SMC-GH MIG partition recipe
Developer Demos	*Fraunhaufer demo*
	AllBeSmart
Developer Radar Tech Talks	ORAN nGRG
	Northeastern
Developer blogs and publications	Northeastern Journal extension (OpenRAN Gym extension contribution)
	Fraunhaufer blog
O-RAN Spring 2024 Plugfest Participation	*Northeastern OTIC*
	EURECOM OTIC

Please refer to the *Getting Started* page as you embark on your ARC-OTA journey.

Chapter 1. ARC-OTA

1.1. Getting Started

Aerial RAN CoLab Over-the-Air is a full-featured platform targeted for next generation wireless evolution, easing developer onboarding and algorithm development in real time networks. ARC-OTA equips developers, researchers, operators, and network equipment providers with all requisite components necessary to deploy a campus network for research. Here is a ten minute video introducing ARC-OTA.

Aerial RAN CoLab Over-the-Air is a developer launchpad that can be used to shape 6G research. It provides a radio network as the innovation sandbox. It is a C-programmable full stack that can help fast-track experimentation and help validate and benchmark results. The ARC-OTA platform enables algorithm design for promising baseband technologies in the 6G ecosystem, including terahertz (THz) band communications, very-large-scale antenna arrays, reconfigurable intelligent surfaces, digital beamforming, spectrum sharing, and the Internet of Things. The workloads associated with all the above items are intrinsically GPU-friendly.

ARC-OTA offers an invaluable platform for next generation wireless communications, and we look forward to collaboration and contributions from developers to extend our blueprints and recipes to help us shape and evolve the platform.

1.1.1. ARC-OTA 1.5 Software Release Manifest

Component		Version
Aerial CUDA-Accelerated RAN	Layer 1	24-1
	Data Lakes	24-1
	pyAerial	24-1
OAI[1]	gNB	2024.w21
	CN[2]	2024.w21
Sterling Skywave Service Management		v0.5
OpenRANGym (OSC RIC Release E)		v1.0
NVIDIA SDK Manager		v2.1
Foxconn O-RU n78 and CBRS		v3.1.15q.551v0706-oam

1.1.2. ARC-OTA Developer Content Map

Product Brief	Provides information about the current release capabilities of the ARC-OTA.
Installation Guide	Describes the hardware bill of materials (BOM), network component configuration, and software required to install ARC-OTA.
Tutorials	Includes video walkthroughs to ease developer on-boarding.
On-Boarding Help	Includes references to reading material, GPU onboarding, and a teaching course for CUDA-C.
Developer Use Cases	Provides information about projects onboarded on the platform.
Release Notes	Outlines the software APIs and functionality as well as any limitations of the current release.
Developer Extensions and Plugins	Provides information about extensions developed for ARC-OTA, as well as plugin interop performed by developers.
Licensing	Provides licensing information regarding ARC-OTA.
Publications	Lists noteworthy publications associated with ARC-OTA.
Support	Provides help with the ARC-OTA.
Background	Describes related technologies.

1.1.3. Glossary

Term or Abbreviation	Description
ARC-OTA	Aerial RAN CoLab – Over The Air
BBU	Baseband Unit
BF3	BlueField-3
BFP	Block Floating Point
CI/CD/CT	Continuous Integration, Continuous Delivery, Continuous Testing
CN	Core network, which provides coordination between different parts of the ac
cuBB	GPU accelerated 5G signal processing pipeline, including cuPHY for Layer 1 P
CU	Centralized Unit
CUDA	Compute Unified Device Architecture
cuPHY	CUDA implementation of 5G PHY layer signal processing functions
Layer 1 from CUDA Accelerated RAN	CUDA GPU software libraries/tools that accelerate Physical Layer, 5G RAN co

[1] For additional context, developers can also review the artifacts in the 2024.w21+ARC1.5 branch.
[2] The Grace Hopper platform requires OAI CN version 2024-June.

Table 1 – continued from

Term or Abbreviation	Description
DL	Downlink
DU	Distributed Unit
FDD	Frequency Division Duplex
FAPI	API for hardware components implementing 3GPP physical layer functions ar
gNB	Next Generation Node B, a component of the 5G mobile communication stan
GPU	Graphical Processing Unit
HW	Hardware
K8s	Kubernetes
MAC	Medium Access Control
MU-MIMO	Multi-User Multiple-Input and Multiple-Output
mMIMO	Massive MIMO
NGC	NVIDIA GPU Cloud
NGAP	NG Application Protocol
NVIPC	NVIDIA inter-process communication standard
OAI	Open Air Alliance
OAM	Operations, Administration and Maintenance
O-RU	Open RAN Radio Unit
PHY	Physical Layer
RAN	Radio Access Network
RF	Radio Frequency
RIC	RAN Intelligent Controller
RLC	Radio Link Control
RRU	Radio Resource Unit
SDK	Software Development Kit
SMC-GH	Supermicro Server gNB with GH200 GPU
SNR	Signal-to-Noise Ratio
SR-IOV	Single Root Input/Output Virtualization
SW	Software
TDD	Time Division Duplex
TTI	Transmission Time Interval
UE	User equipment
UL	Uplink

1.2. Product Brief

1.2.1. Product Highlights

- A wideband, real-time platform to replace existing narrow-band, non-real-time systems
- A full-featured platform for NG wireless evolution
- C/C++ programmable from the physical layer through to the Core Node (CN)
- Quick network onboarding and algorithm development in real-time networks
- Accelerated AI experimentation in wireless RAN workloads
- A pipeline for data collection, storage, and parsing using 3GPP schema for wireless communication.

1.2.2. Key Performance Indicators

The configuration and capabilities of ARC-OTA are outlined in the following sections.

Number Antennas	4T4R	
Number of Component Carriers	1x 100MHz carrier	
Subcarrier Spacing (PDxCH; PUxCH, SSB)	30kHz	
FFT Size	4096	
MIMO layers	DL: 4 layers; UL: 1 layers	
Duplex Mode	Release 15 SA TDD	
Number of RRC connected UEs	16	
Number of UEs/TTI	2	
Frame structure and slot format	DDDDDDSUUU	
	DDDSU	
User plane latency (RRC connected mode)	< 10ms one way for DL and UL	
Synchronization and Timing	IEEE 1588v2 PTP; SyncE; LLS-C3	
Frequency Band	n78	
Max Transmit Power	22dBM at RF connector	
Peak throughput	SMC-GH	DL: ~1.03Gbps; UL: ~125Mbps
	Dell R750 / Gigabyte	DL: ~800Mbps; UL: ~110Mbps
Bi-directional UDP Traffic	> 10+ hours exercised (SMC-GH)	
	> 4.0 hours exercised (Dell R750 + A100X)	
	> 4.0 hours exercised (Gigabyte + A100 + CX6-DX)	

Note

OTA test was performed with the following configuration: Samsung S22 + Gigabyte + DDDDDDSUUU.

Tip

To learn how KPIs have changed from last release, refer to the *Release Notes*.

1.2.3. Product Features

Feature	Description
Full stack software	A 3GPP Release 15 compliant and O-RAN 7.2 split 5G SA 4T4R wireless stack, with all network elements from Radio Access Network and 5G Core. Aerial CUDA-Accelerated RAN Layer 1 is integrated with Open Air Alliance (OAI) (https://openairinterface.org/) Distributed Unit (DU), Centralized Unit(CU), or a 5G NR gNB and 5G Core Node(CN) network elements.
Radio network hardware components	The COTS hardware Bill of Materials used for NVIDIA qualification is available in the *OTA-qualified HW BOM manifest*
Source code access	Complete access to source code in C/C++, from Layer 1 through 5GC to jump start customizations and next-generation algorithm research. Review the *Licensing* section for more details.
Developer extensions and plugins	To accelerate innovation, developer extensions and contributions are welcome. For example, The *Kubernetes Service Management* optional developer extension from Sterling provides two capabilities: Kubernetes Service Orchestration and Service Monitoring.
AI frameworks	AI frameworks and tools are integrated to ease the AI/ML developer journey for advanced wireless research. For example, pyAerial and Aerial DataLakes has been integrated with ARC-OTA.

1.2.3.1 5G NR gNB Features

Component	Capabilities
gNB PHY	Aerial CUDA-Accelerated RAN Layer 1 PHY (cuPHY) adheres to 3GPP Release 15 standard specifications to deliver the following capabilities. PHY capabilities include the following: ▶ Error detection on the transport channel and indication to higher layers ▶ FEC encoding/decoding of the transport channel ▶ Hybrid ARQ soft combining ▶ Rate matching of the coded transport channel to physical channels ▶ Mapping of the coded transport channel onto physical channels ▶ Power weighting of physical channels ▶ Modulation and demodulation of physical channels including ▶ Frequency and time synchronization ▶ Radio characteristics measurements and indication to higher layers ▶ Multiple Input Multiple Output (MIMO) antenna processing ▶ Transmit Diversity (TX diversity) ▶ Digital and Analog Beamforming ▶ RF processing 3GPP standards specifications that define the Layer 1 compliance are: ▶ TS 38.211 (38.211 v15.8.0) numerologies, physical resources, modulation, sequence, signal generation ▶ TS 38.212 (38.212 v15.8.0) Multiplexing and channel coding ▶ TS 38.213 (38.213v15.8.0) Physical layer procedures for control ▶ TS 38.214 (38.214v15.8.0) Physical layer procedures for data ▶ TS 38.215 (38.215v15.8.0) Physical layer measurements ▶ TS 38.104 (base station radio Tx and Rx) Base Station (BS) radio transmission and reception Aerial CUDA-Accelerated RAN complies with ORAN FH CUS specification version 3 (version 4 for power scaling) Aerial CUDA-Accelerated RAN complies with northbound interfaces adopted by industry based on Small Cells Forum for Layer 1 and Layer 2 (SCF FAPI).
gNB MAC	▶ MAC -> PHY configuration using NR FAPI P5 interface ▶ MAC <-> PHY data interface using FAPI P7 interface for BCH PDU, DCI PDU, PDSCH PDU ▶ Scheduler procedures for SIB1 ▶ Scheduler procedures for RA

1.2.3.2 5G Core Features

AMF	Features	NGAP AMF status indication (3GPP TS 38.413)
		Add UE Retention Information support (3GPP TS 38.413)
		Support of Location services with LMF and AMF (3GPP TS 29.518, 3GPP TS 38.413, 3GPP TS 23.502)
		Update NAS with Rel 16.14.0 IEs: Refactor code for Encode/Decode functions; cleanup NAS library (3GPP TS 24.501)
	Fixes	Fix typo for N1N2MessageSubscribe (3GPP TS 29.518)
		Fix issue when receiving PDU session reject from SMF (3GPP TS 29.518, 3GPP TS 23.502)
	Technical Debt	Reformatting of the SCTP code
		Refactor promise handling
		Removing dependencies to libconfig++ (Only YAML file can be read as configuration)
SMF	Features	Add N1/N2 info in the message response to AMF if available (3GPP TS 29.502)
	Fixes	Add connection handling mechanism between NRF and SMF
	Technical Debt	Refactor SMF PFCP associations to use UPF profile
UDM	Fixes	Add connection handling mechanism between NRF and UDM
UDR	Technical Debt	Fixed builds
		Add connection handling mechanism between NRF and UDR
		Improve MongoDB support
Common		New HTTP Client library (CPR) for all the NFs
		Support mobility registration update procedure (3GPP TS 23.502)

1.2.3.3 5G Fronthaul Features

RU Category	Category A
FH Split Compliance	7.2x with DL low-PHY to include Precoding, Digital BF, iFFT+CP and UL low-PHY to include FFT-CP, Digital BF
FH Ethernet Link	25Gbps x 1 lane
Transport encapsulation	Ethernet
Transport header	eCPRI
C Plane	Conformant to O-RAN-WG4.CUS.0-v02.00 7.2x split
U Plane	Conformant to O-RAN-WG4.CUS.0-v02.00 7.2x split
S Plane	Conformant to O-RAN-WG4.CUS.0-v02.00 7.2x split
M Plane	Conformant to O-RAN-WG4.CUS.0-v02.00 7.2x split
RU Beamforming Type	Code book based

1.2.4. Product Blueprints

To ease developer onboarding, this section provides reference blueprints with key ingredients that were combined to create a tested product prototype: A full-stack innovation sandbox to accelerate innovation in wireless networks and provide new insights on experiments and research. Many assumptions made for analytical and simulation studies may likely improve benchmarks or prove to be invalid in a real network. Our experience in innovation labs as we design, setup, deploy, and add tools and frameworks is available to all developers. We have deliberated on the hardware components, software configurations, and deployment strategies. We have also run into difficulties and pitfalls, and want to ensure others that the hardware components and software configurations have undergone a rigorous qualification process. Each developer delta for the lab experimental networks is expected to be limited to the environment variability, the transmission power, attenuation, and limited set of variables.

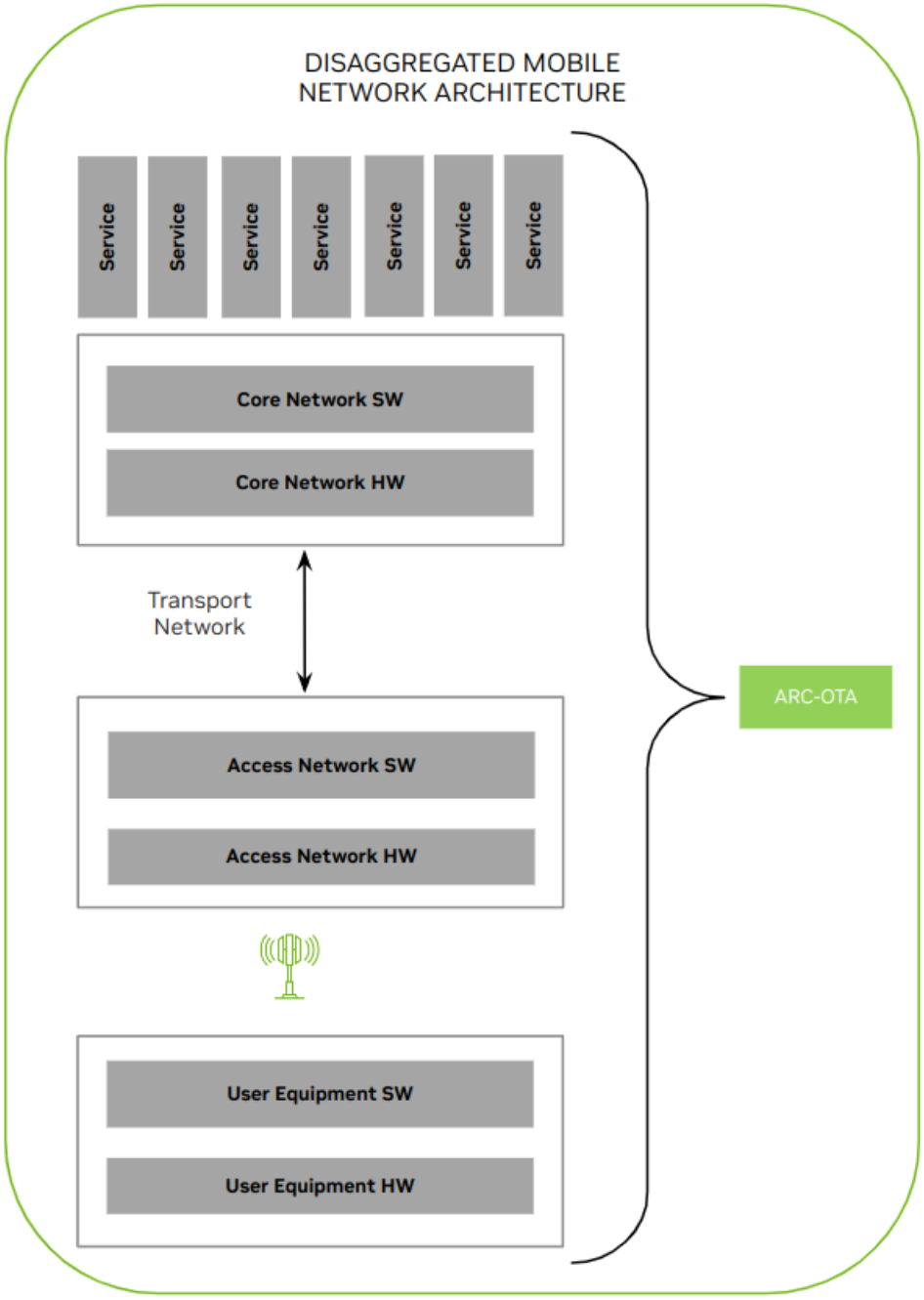
As we look to leverage, extend, and innovate, here are the key guiding attributes for the blueprints:

Attribute	Description
Prototype	Serve as a reference implementation
Re-Use	is important to avoid having to “re-invent” the frameworks, tools, platform
Automation	to ease setup and minimize deployment pitfalls for software environments and configurations
Uniformity	Access to exact set of combined ingredients or recipe used in the reference
Configuration	customization is minimal and needed when developer wants to change behavior
Availability	Service health monitoring of the network necessary
Extendibility	Easy to extend based on O-RAN modular, flexible, open architecture
Performant OTA Network	Develop innovation prototypes and validate benchmarks versus expected on-paper or simulation
Extensions and Plugins	Developer blueprints for others to leverage as building blocks

1.2.4.1 Blueprint Table

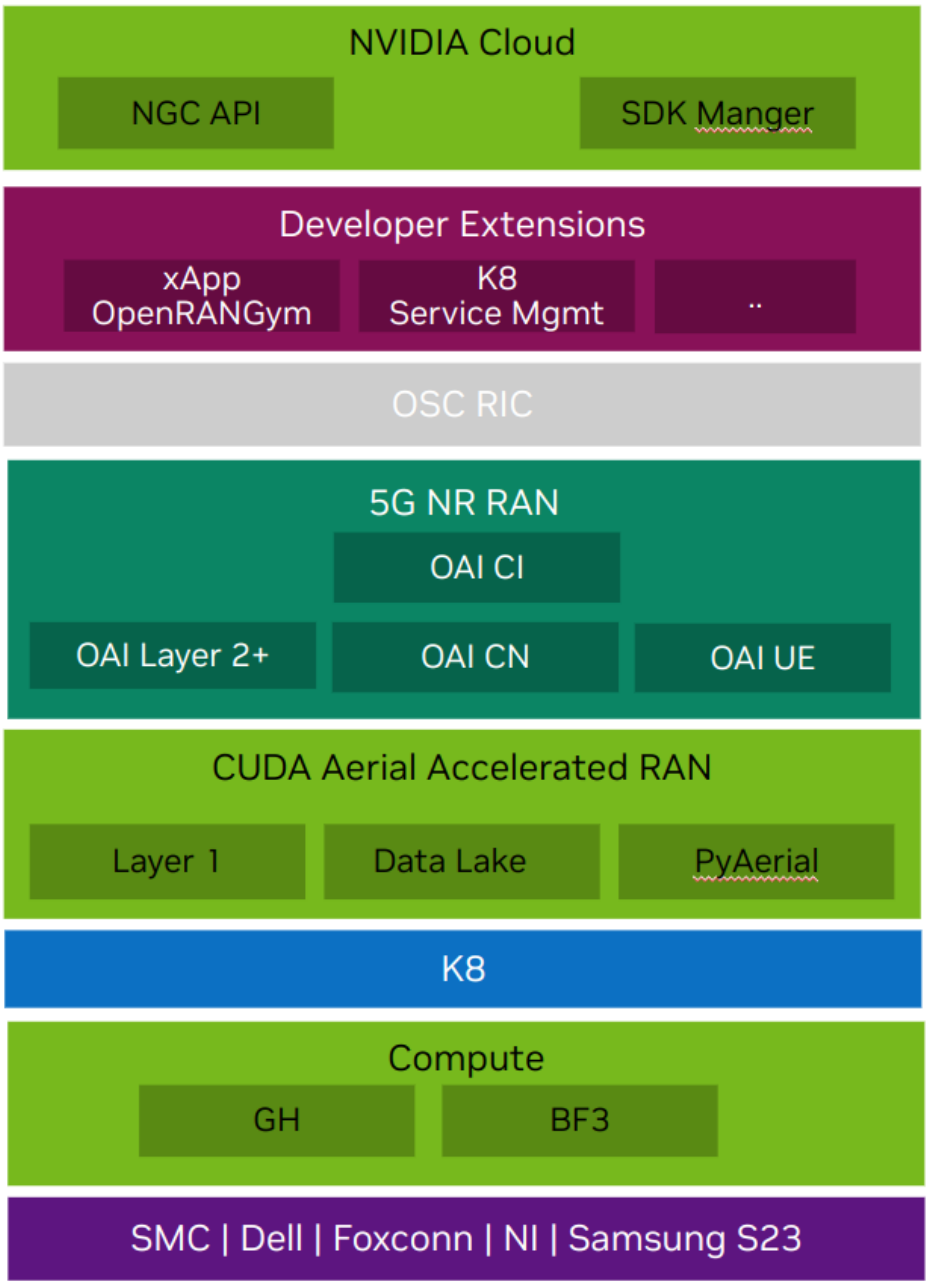
Blueprint	Description	Downloadable Version
ARC-OTA	Disaggregated Mobile Network As a Research Sandbox	PDF
Full Stack Programmable	Launchpad for Advanced Wireless Developers – Gateway to Developer Innovations and Extensions	PDF
Multi-Vendor Integration	NVIDIA Qualified COTS Multi Vendor Hardware and Software Blueprint	PDF
Multi-Vendor Disaggregation	NVIDIA ARC-OTA developer plugin Multi Vendor Interop with a modular element change blueprint	PDF
On-Prem Data Center Deployment	NVIDIA Qualified On Prem Data Center Deployment Blueprint	PDF
K8 Service Management	NVIDIA Qualified Service Management Blueprint	PDF
O-RAN 7.2 Split	NVIDIA GPU inline accelerate high PHY	PDF
CSI Dataset	NVIDIA ARC-OTA Multi-UE Channel State Information dataset blueprint	PDF
OpenRAN Gym	Developer extension integrated with OSC RIC	PDF

1.2.4.2 ARC-OTA


DISAGGREGATED MOBILE
NETWORK ARCHITECTURE
Service
Service
Service
Service
Service
Service
Service
Core Network SW
Core Network HW
Transport
Network
Access Network SW
Access Network HW
User Equipment SW
User Equipment HW
ARC-OTA

Component	Feature
Hardware Stack	Leverages COTS (Common Off The Shelf) vendors
Software Stack	Fully programmable in C/C++
	Network uses CUDA Accelerated RAN Layer 1 and OAI software.
	Extensible through network services
Integration	Developer community is encouraged to extend the stack by contributions across all layers of the stack. Early examples include *Sterling SkyWave Service Management* and *O-RAN OSC RIC*.
Deployment	NVIDIA SDK Manager offers automation to easily deploy this NVIDIA qualified blueprint.

1.2.4.3 Full Stack Innovation

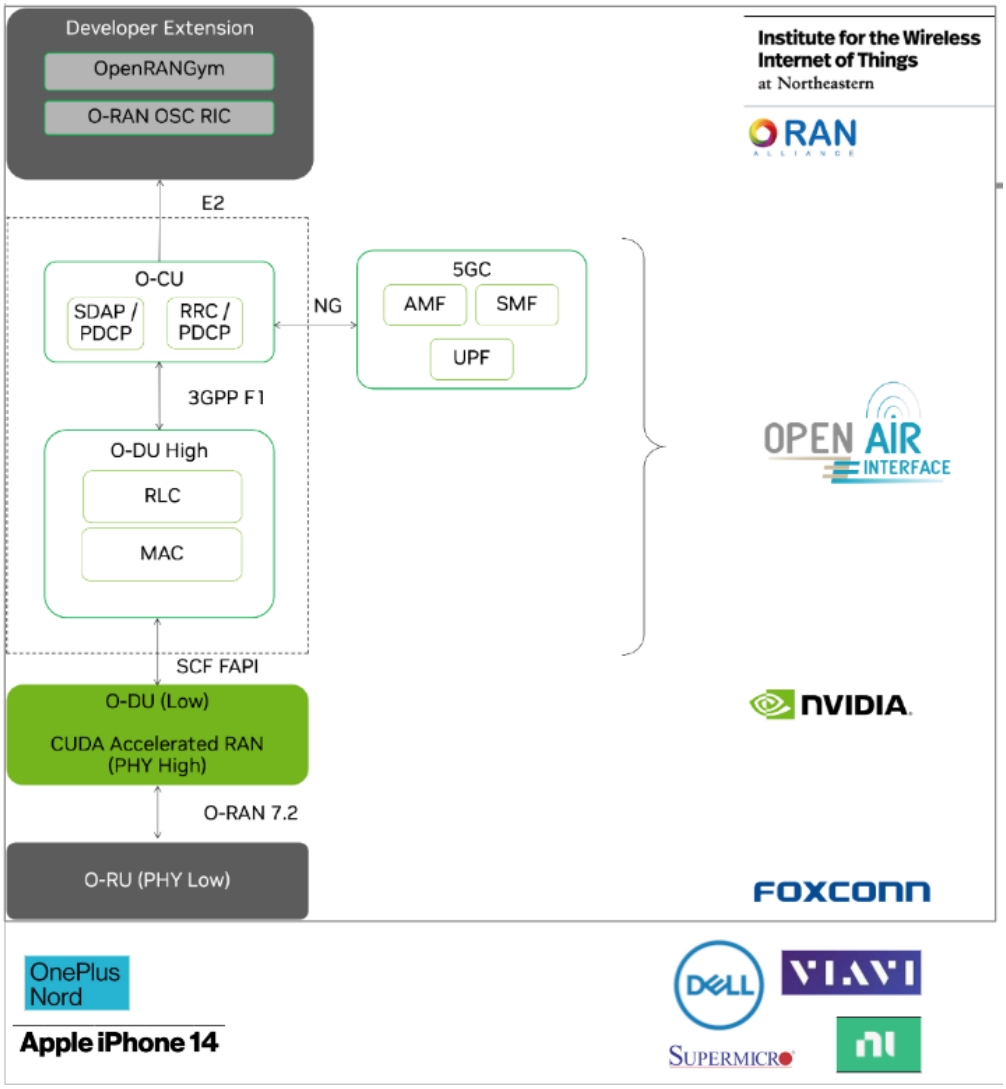


Compontent	Feature
COTS hardware	COTS infrastructure composed of compute, virtualization, radios, fronthaul networking, precision timing, accelerators.
Virtualization	Virtualized RAN workloads from NVIDIA and Open Air Alliance
AI/ML Frameworks	Data Lake + pyAerial for AI/ML frameworks : RF / IQ data + FAPI
Standards	3GPP Release 15+ O-RAN 7.2 split P5G on-prem lab network
Developer Tools	Reference OAI CI leverage for developer integration workflow
Developer Extension – Sterling	NVIDIA NGC APIs for Sterling K8 service orchestration of network functions
Developer Extension – OpenRAN Gym	Developers can deploy AI/ML models for and on the radio access network using OSC radio intelligent controller by following Northeastern OpenRAN Gym tutorial.
Developer Tools – Netowrk as a service	NVIDIA SDK Manager development environment setup automation

Note

Developer contributions through extensions and plugins - for community benefit and to accelerate pace of innovations welcome!

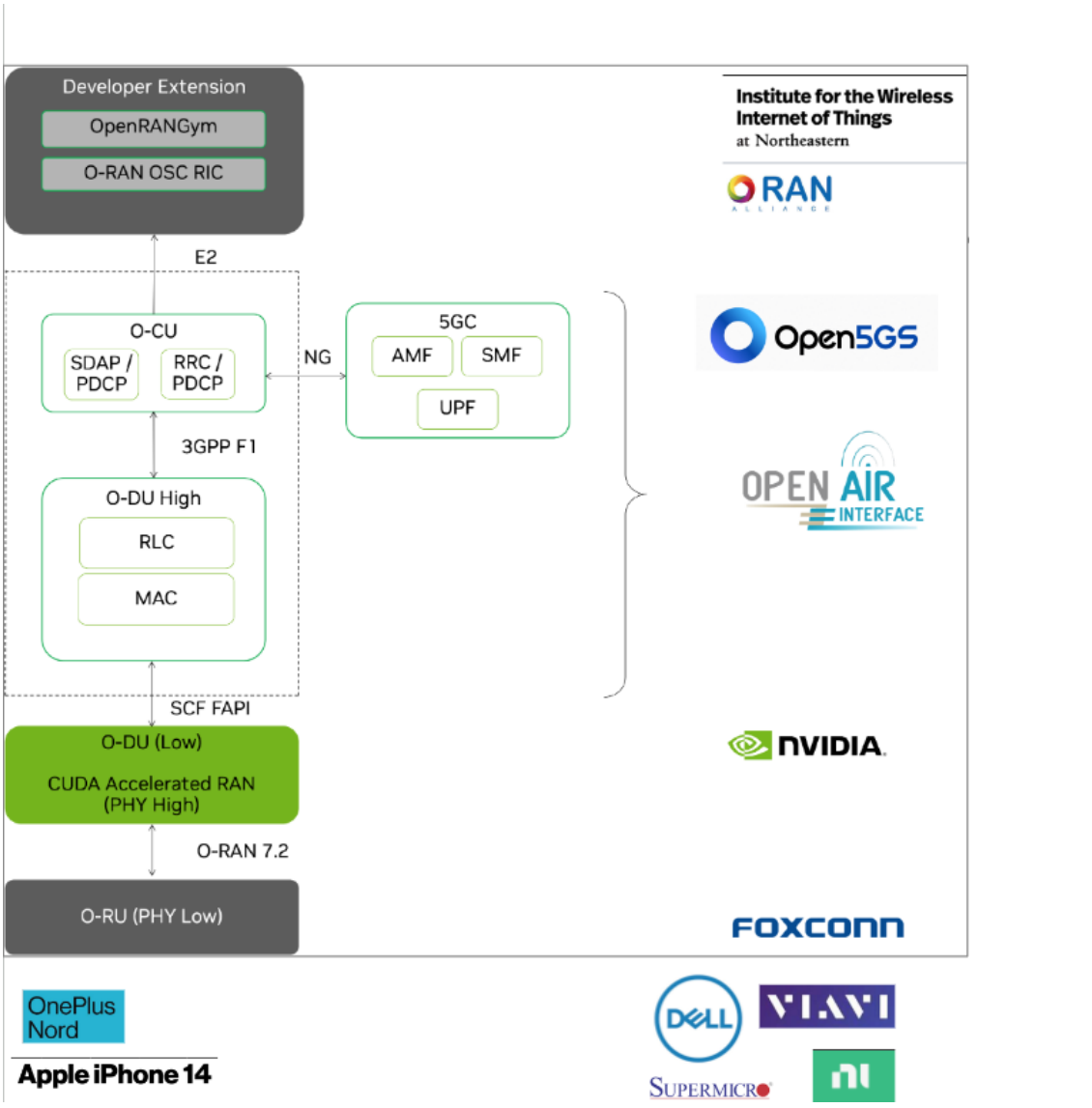
1.2.4.4 Multi-Vendor Integration



Leveraging O-RAN, and 3GPP specifications and interfaces enables multi-vendor interop towards the full stack building blocks and developer extensions and plugins. A multi-vendor reference blueprint is extended with the SCF FAPI interface between the O-DU low and O-DU high.

Organization	Features
Northeast-ern	E2 interface plugin leveraging O-RAN OSC RIC and template xApps
OpenAirAl-liance	O-DU-High (Layer 2), O-CU and 5GC
NVIDIA	O-DU Low / Phy High
Foxconn	O-RU Phy Low
Others	Handsets (Apple iPhone 14, Samsung S23), Viavi Qualsar Grandmaster, Dell FH switch, Supermicro server.

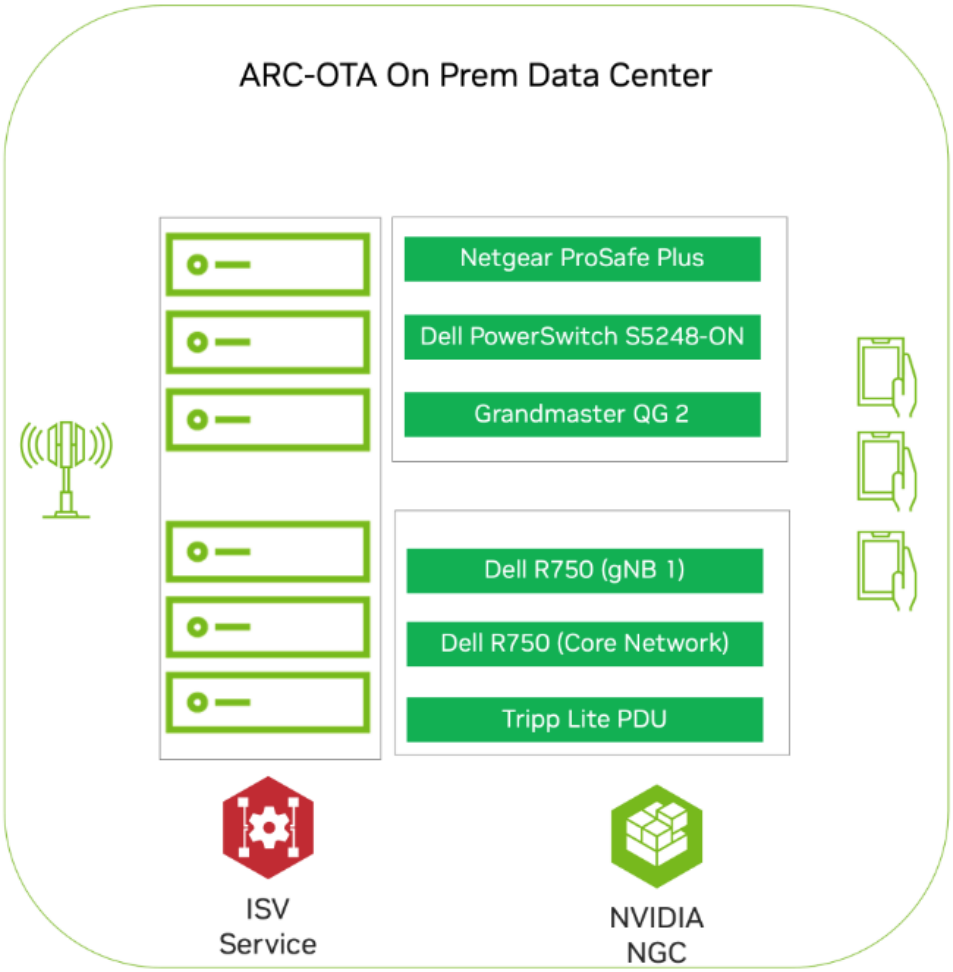
1.2.4.5 Multi-Vendor Disaggregation



Leveraging O-RAN, and 3GPP specifications and interfaces enables multi-vendor interop towards the full stack building blocks and developer extensions and plugins. A multi-vendor reference blueprint is extended with the SCF FAPI interface between the O-DU low and O-DU high.

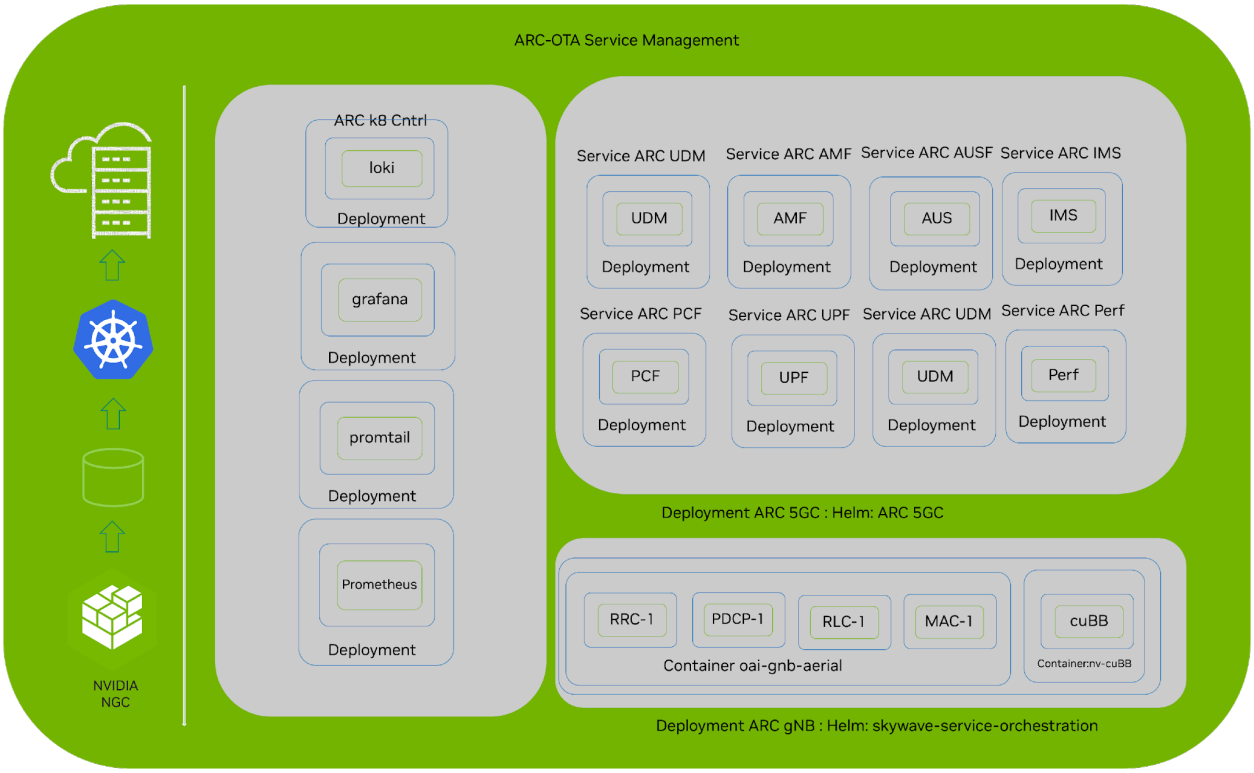
Organization	Features
Northeastern	Developer extension with E2 interface plugin leveraging O-RAN OSC RIC and template xApps
	Plugin to leverage https://open5gs.org/ for 5GC instead of NVIDIA reference Open Air Interface 5GC. This highlights the use of modular, open, and interoperable components within disaggregated ORAN architecture
OpenAirAlliance	O-DU-High (Layer 2), O-CU and 5GC
NVIDIA	O-DU Low / Phy High
Foxconn	O-RU Phy Low
Other	Handsets (Apple iPhone 14, Samsung S23)
	Viavi Qualsar Grandmaster
	Dell FH switch
	Supermicro server

1.2.4.6 On-Prem Data Center Deployment



Component	Feature
Deployment	Private data center can be housed and maintained by developers and researchers in their own facilities.
	Third-party ISV Managed Developer Service can be leveraged to procure, install, configure and monitor the on-prem data center.
Virtualization	On-prem infrastructure can be used to run a private cloud.
	ARC-OTA compute resources are virtualized for gNB and 5GC.

1.2.4.7 K8 Service Management



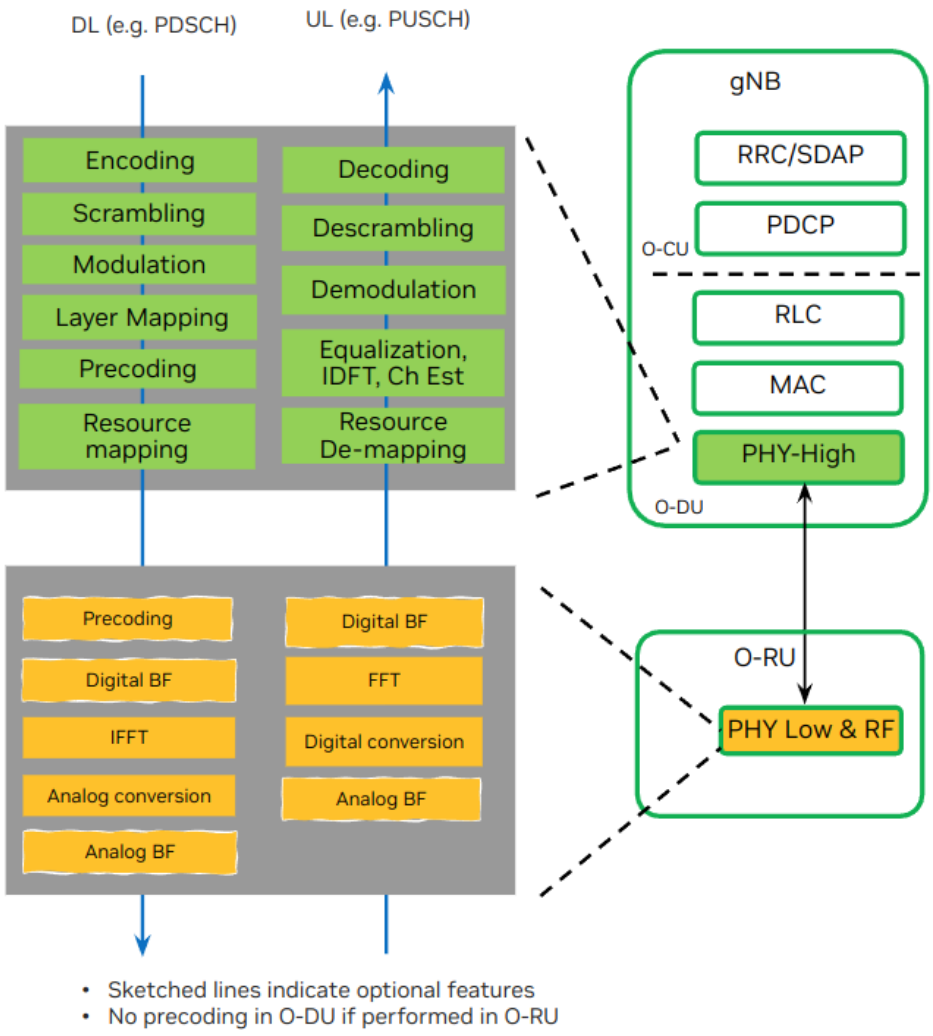
ARC-OTA provides Kubernetes for container orchestration. Both single node and multi-node deployment topologies are supported. ARC-OTA uses helm to manage applications.

Deployment	Description
ARC gNB	The helm chart uses K8 deployment to create pod gNB. The gNB pod contains the containers nv-cubb and oai-gnb-aerial. Both containers are installed in the same pod to allow the use of shared memory between Layer 1 and Layer 2+. This helm chart is available for download from NGC.
Deployment ARC 5GC	The helm-chart is installed on the CN5G server or same physical server as the gNB. This helm chart creates multiple K8 deployments depicted on left. This helm chart is available for download from the OAI GitLab repository

ARC-OTA Service Monitoring feature uses a combination of Grafana, Loki, Promtail and Prometheus. The feature was developed using open-source industry standard tools and it can be extended to specific developer needs

Reference the *Sterling developer extension* for additional details.

1.2.4.8 O-RAN 7.2 Split

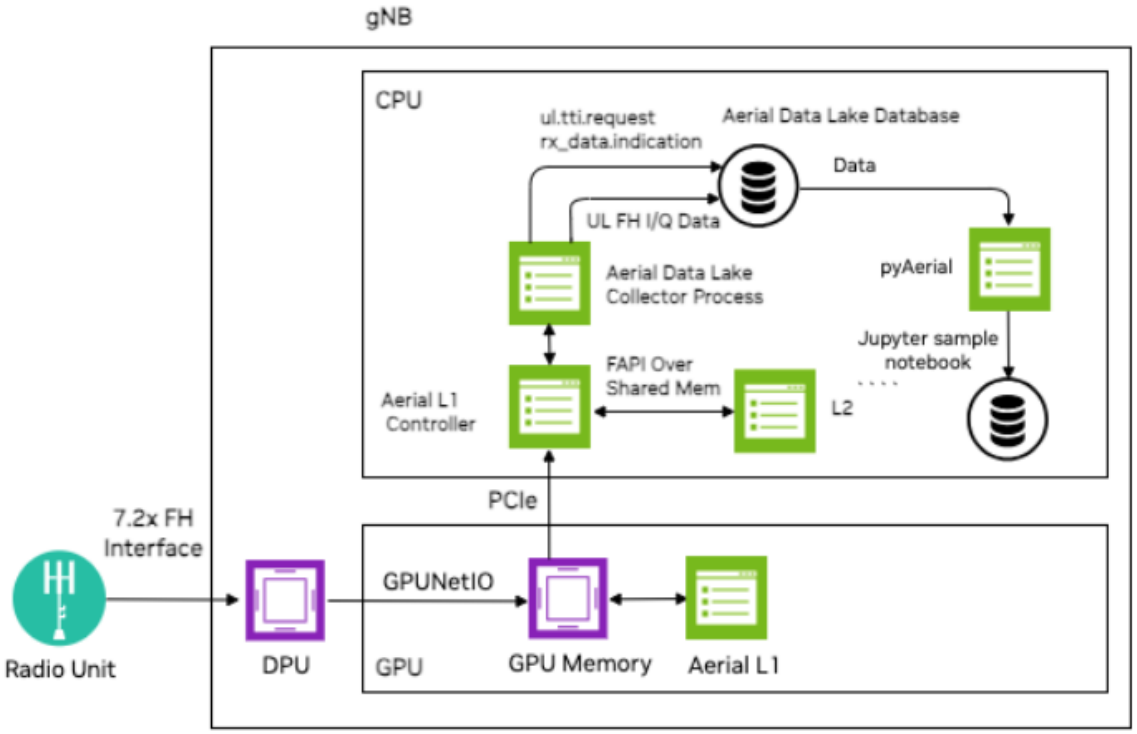


O-RAN's split-RAN concept disaggregates the RAN into multiple functional components. These components can be deployed on different hardware and software platforms and can be interconnected using open interfaces.

ARC-OTA leverages the 7.2x split, which divides the protocol stack into the following:

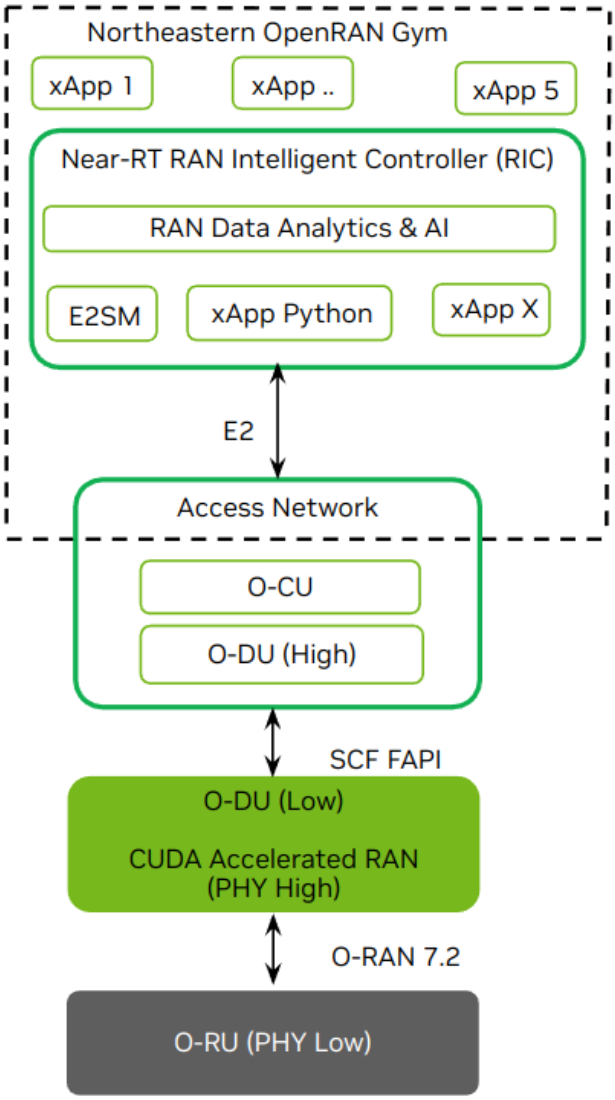
Component	Description
O-RU (O-Radio Unit)	The O-RU is responsible for the physical layer (PHY) processing, including RF signal processing and analog-to-digital conversion.
O-DU (O-Distributed Unit)	The O-DU is responsible for the higher-layer processing, including MAC, RLC, and PDCP
Fronthaul interface	O-RAN alliance specified fronthaul interface between the O-DU and O-RU based on the 7.2x split. This interface supports control, user, synchronization (CUS), and management(M) planes

1.2.4.9 CSI Dataset



- ARC-OTA integrated Aerial Data Lake provides the ability to capture OTA radio frequency (RF) data from the base station (BS). Raw IQ samples from the 7.2x split fronthaul (FH) interface are collected in a database file
- Using the Aerial Data Lake database APIs, pyAerial can access RF samples in the database and transform these samples into training data or data sets for signal processing functions.
- A sample Jupyter notebook has been provided that can create a sample multi-UE CSI dataset
- SDKManager automation will help developers install AI/ML frameworks to easily generate a dataset

1.2.4.10 OpenRAN Gym



- Streaming of relevant key performance metrics (KPMs) and the enforcement of control actions to reflect decisions taken by the xApps on a Near-Real-Time (Near-RT) RAN Intelligent Controller (RIC) blueprint is shared through the OpenRAN Gym integration and an example monitoring xApp.
- Potential xApps can be developed for network intelligence like handover optimization, policy enforcement, resource assurance or radio link management or resource control applications like load balancing or network slicing.
- Northeastern has integrated the E2 interface with O-RAN OSC RIC and a template monitoring xApp using a custom E2 Agent to E2 Service Model. xApp Python bindings and xApp Connector collectively provide the RAN close loop monitor and control functions. This tooling helps developers incorporate network adaptability functions with AI/ML based xApps

1.3. Installation Guide

The following sections will guide you through the process of integrating and deploying ARC-OTA for advanced 5G and 6G research:

Installation Step	Description
Part 1 – Procure the Hardware	Procure all the required hardware based on the published BOM in this document.
Part 2 – Configure the Network Hardware	Perform setup on the required network devices.
Part 3 – Configure gNB Server	Install the kernel command line specific to ARC-OTA.
Part 4 – Install ARC-OTA Using SDK Manager	Use the SDK Manager to install ARC-OTA.
Part 5 – Validate the Setup	Validate the setup using bi-directional UDP.
App Note – ARC-OTA Step-by-Step Debug Command Line	This installation method is for debug purposes only. Instead of using SDK Manager to install ARC-OTA, all necessary software components are installed manually.

1.3.1. Part 1. Procure the Hardware

Procure all the hardware listed in the following BOM.

1.3.1.1 ORAN 7.2x Reference Hardware Components

ARC-OTA covers a wide range of use cases for developers in advanced wireless. Upon integration and qualification, these ORAN 7.2x split-aligned BOM hardware components are shared via the reference below.

Note

Unless a specific solution architecture differs based on use case, all components are required in a unit of 1.

5G Infra Component	Hardware Manifest		
ARC-OTA gNB	SMC-GH: Supermicro Server ARS-111GL-NHR (Config 2)	CPU	72-core NVIDIA Grace
		GPU	GH200
		Memory	480GB LPDDR5X with ECC

continues on next page

Table 2 – continued from previous page

5G Infra Component	Hardware Manifest		
		Network	BF3 NIC (x2)
	Dell PowerEdge R750 Server + A100X	GPU	A100X
	Gigabyte Edge E251-U70 Server	CPU	Intel Xeon Gold 6240R, 2.4GHz, 24C48T
		GPU	GA100
		Memory	96GB DDR4
		Network	MLX CX6-DX MCX623106AE-CDAT
CN[1]	Dell PowerEdge R750 Server		
FrountHaul(FH) Switch only pick one	Dell PowerSwitch S5248F-ON		
	Fibrolan Falcon RX		
GrandMaster(GM)	QULSAR Qg 2 Multi-Sync Gateway		
O-RUs supported	**ORU**	**Freq Band**	
	Foxconn RPQN-7801E (4T4R)	(indoors) 3.7GHz - 3.8GHz	
	Foxconn RPQN 4800E (4T4R)	(indoors) 3.55GHz - 3.77GHz	
UEs supported	**UE**		**Configuration**
	Sierra Wireless EM9191 NR 5G Modem		SU-MIMO 2DL, 1UL
	Quectel RM500Q-GL UE		SU-MIMO 2DL, 1UL (Reference 1, Reference 2)
	5G USB Dongle		SU-MIMO 2DL, 1UL
	iPhone 14 Pro; iOS version 16.6.1; model number MPXV3HX/A		SU-MIMO 4DL, 1UL
	Samsung S22 SM-S9010		SU-MIMO 4DL, 1UL
Cables	Dell C2G 1m LC-LC 50/125 Duplex Multimode OM4 Fiber Cable Aqua 3ft Optical patch cable		
	NVIDIA MCP1600-C001E30N DAC Cable Ethernet 100GbE QSFP28 1m		

continues on next page

Table 2 – continued from previous page

5G Infra Component	Hardware Manifest
	Beyondtech 5m (16ft) LC UPC to LC UPC Duplex OM3 Multimode PVC (OFNR) 2.0mm Fiber Optic Patch Cable
	CableCreation 3ft Cat5/Cat6 Ethernet Cables
PDUs	Tripp Lite 1.4kW Single-Phase Monitored PDU with LX Platform Interface, 120V Outlets (8 5-15R), 5-15P, 12ft Cord, 1U Rack-Mount, TAA
Transceivers	Finisar SFP-to-RJ45 Transceiver
	Intel Ethernet SFP+SR Optics
	Dell SFP28-25G-SR Transceiver
Ethernet Switch	Netgear ProSafe Plus JGS524E Rackmount
iPerf Laptop	Connected to the switch (10G ethernet)

Important

The following components have reached end-of-life (EOL).

- Gigabyte Edge E251-U70 Server
- A100X (will no longer be available from distributor for ARC-OTA)

1.3.2. Part 2. Configure the Network Hardware

Note

Refer to the NVIDIA SDK Manager resources for setup and installation of ARC-OTA.

The network hardware is configured in the following steps.

1. Setup the GrandMaster
2. Setup the switch
3. Setup PTP
4. Setup Foxconn O-RU

[1] The same SMC-GH server can be used to host both gNB and ARC-OTA 5GC.

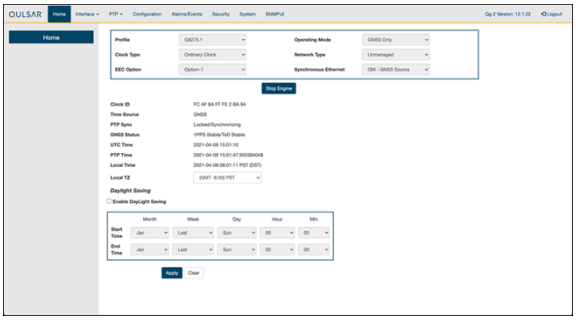
1.3.2.1 Chapter 2.1 Setup the Qulsar GrandMaster

1.3.2.1.1 Step 1.

Follow the Qulsar User Guide to set up the MGMT connection.

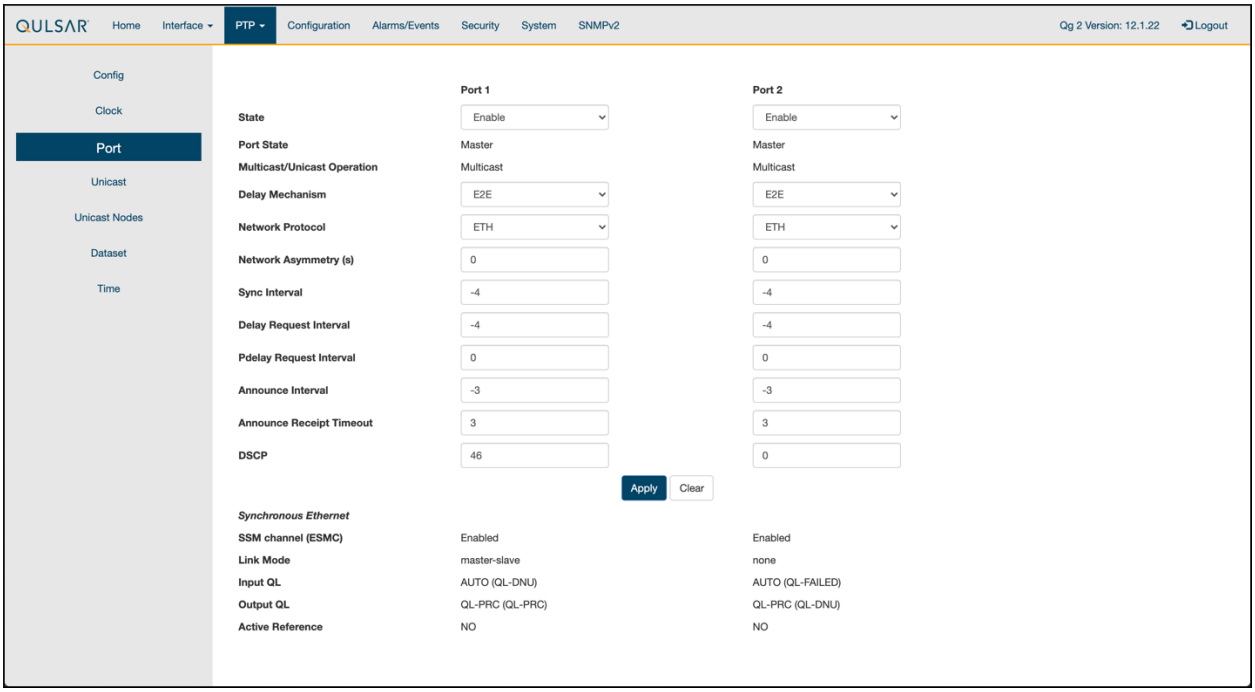
1.3.2.1.2 Step 2.

Set the operating mode to **GNSS Only**, and other fields as such, then run **Start Engine**.



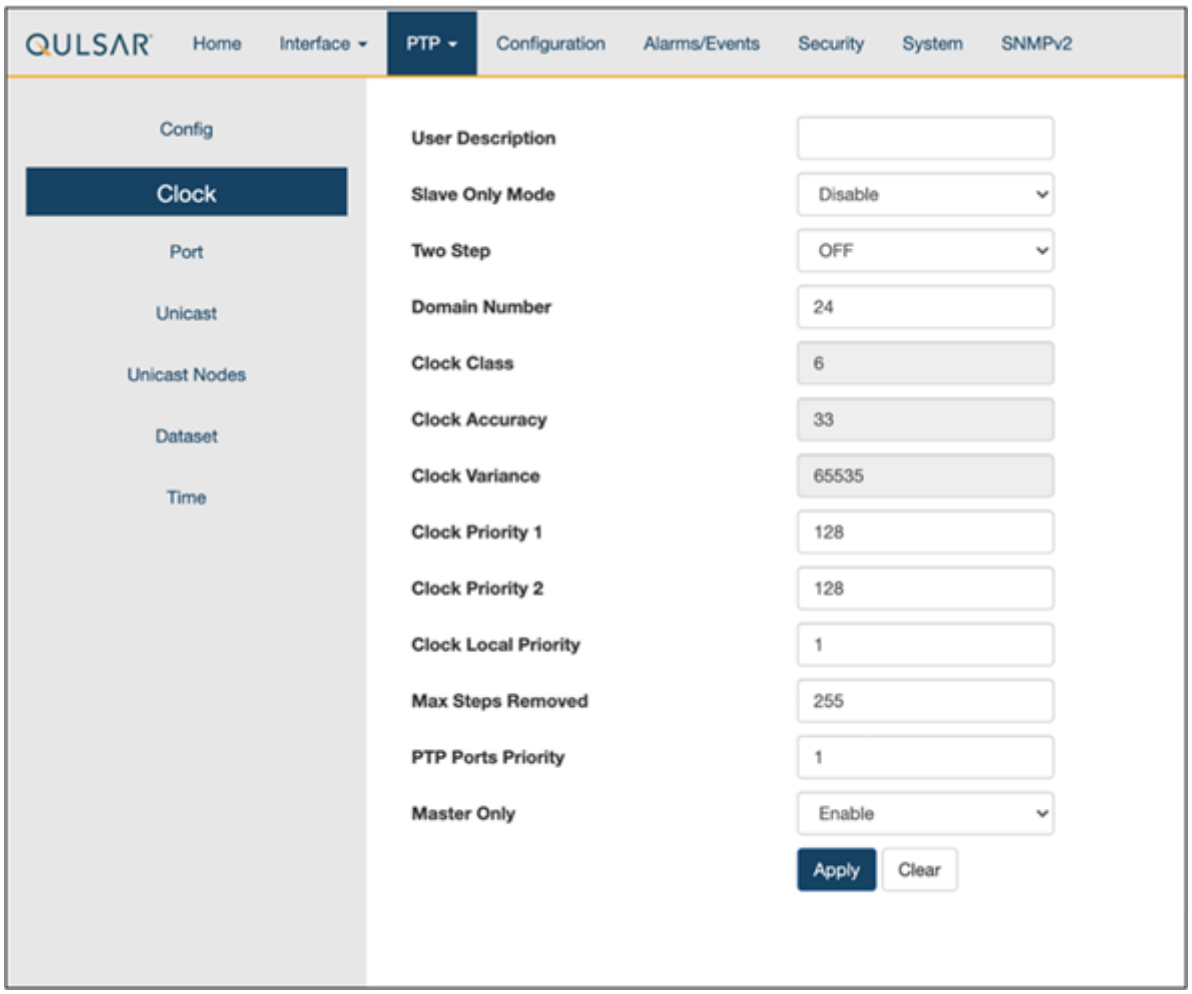
1.3.2.1.3 Step 3.

Enable the ports on the GrandMaster with the **8275.1 Profile** configurations.



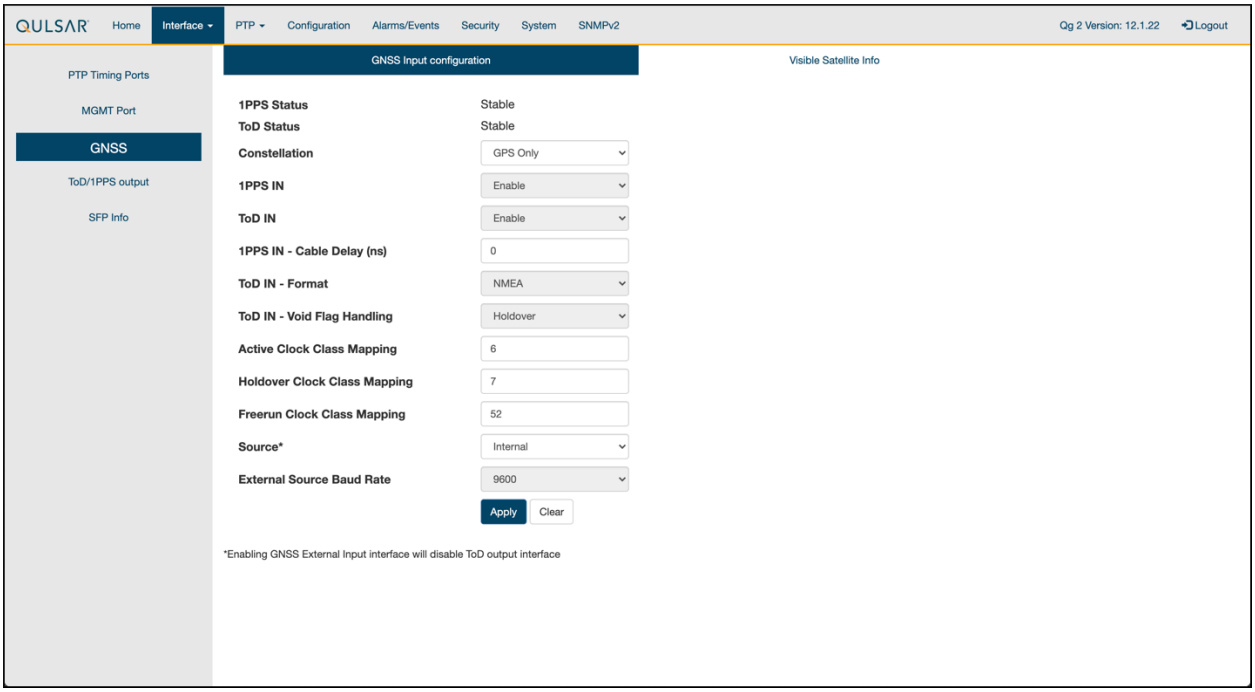
1.3.2.1.4 Step 4.

Configure the clock configs as follows:



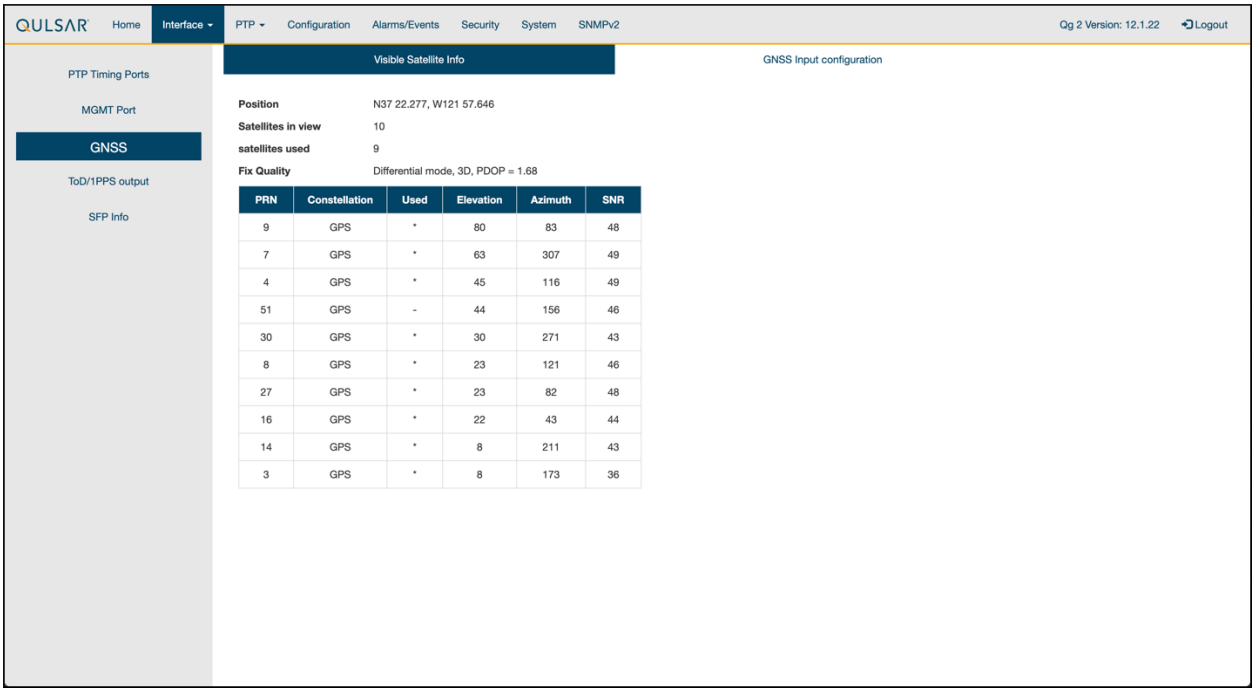
1.3.2.1.5 Step 5.

Ensure the GPS configuration values are unchanged from the QG2 default settings.



1.3.2.1.6 Step 6.

Verify that the GPS Signal reaches the GrandMaster.



1.3.2.2 Chapter 2.2 Switch Setup

1.3.2.2.1 Chapter 2.2.1 Dell Switch

The following example uses these VLAN 2 settings:

- RUs are on ports 1 and 7
- GrandMaster is on port 5
- CN is on ports 11 and 12
- gNB ports are connected to ports 49 and 51

1. Set up MGMT access to the switch (in this case 172.168.20.67):

```
OS10# configure terminal
OS10(config)#
interface mgmt1/1/1
no shutdown
no ip address dhcp
ip address 172.16.204.67/22
exit
```

2. Use SSH to access `admin@172.168.204.67`.
3. Set the speed to 10G for port groups 1 and 2.

```
OS10(config)#
port-group 1/1/1
mode Eth 10g-4x
exit
port-group 1/1/2
mode Eth 10g-4x
exit
```

4. Enable PTP on the switch.

```
OS10# configure terminal
OS10(config)#
ptp clock boundary profile g8275.1
ptp domain 24
ptp system-time enable
!
```

5. Configure the GrandMaster port.

```
OS10(config)#
interface ethernet 1/1/5:1
no shutdown
no switchport
ip address 169.254.2.1/24
flowcontrol receive off
ptp delay-req-min-interval -4
ptp enable
ptp sync-interval -4
ptp transport layer2
exit
```

After some time, the following values will print:

```
<165>1 2023-05-09T07:49:22.625584+00:00 OS10 dn_alm 1021 - - Node.1-Unit.1:PRI
↪[event], Dell EMC (OS10) %PTP_SYSTEM_TIME_NOT_SET: System time is not set.
↪System time will be set when the clock is.
<165>1 2023-05-09T07:51:22.312557+00:00 OS10 dn_alm 1021 - - Node.1-Unit.1:PRI
↪[event], Dell EMC (OS10) %PTP_CLOCK_PHASE_LOCKED: Clock servo is phase locked.
<165>1 2023-05-09T07:51:22.313081+00:00 OS10 dn_alm 1021 - - Node.1-Unit.1:PRI
↪[event], Dell EMC (OS10) %PTP_SYSTEM_TIME_UPDATE_STARTED: System time update
↪service is started. Update interval: 60 minutes.
<165>1 2023-05-09T07:51:59.334346+00:00 OS10 dn_alm 1021 - - Node.1-Unit.1:PRI
↪[event], Dell EMC (OS10) %ALM_CLOCK_UPDATE: Clock changed MESSAGE=apt-daily.
↪timer: Adding 6h 36min 18.719270s random time.
<165>1 2023-05-09T07:57:27.254181+00:00 OS10 dn_alm 1021 - - Node.1-Unit.1:PRI
↪[event], Dell EMC (OS10) %ALM_CLOCK_UPDATE: Clock changed MESSAGE=apt-daily.
↪timer: Adding 4h 31mi
```

6. Configure the Fronthaul Network Configuration by creating a VLAN.

> **Note**
>
> If you choose to use a different VLAN, you must modify the Aerial YAML file and O-RU configuration. C- and U-planes use the same VLAN.

Create “VLAN 2”.

```
OS10(config)#
interface vlan 2
OS10(conf-if-vl-2)#
<165>1 2023-03-16T16:51:36.458730+00:00 OS10 dn_alm 813 - - Node.1-Unit.1:PRI
↪[event], Dell EMC (OS10) %IFM_ASTATE_UP: Interface admin state up :vlan2

OS10(conf-if-vl-2)# show configuration
!
interface vlan2
no shutdown
OS10(conf-if-vl-2)# exit
```

7. Configure the RU, gNB, CN, and MEC ports.

Interfaces that are configured to be slower than their maximum speed have a `:1` appended to their name. This applies to ports in port groups 1 and 2.

```
no shutdown
switchport mode trunk
switchport trunk allowed vlan 2
mtu 8192
flowcontrol receive off
ptp enable
ptp transport layer2
ptp role timeTransmitter
exit
```

8. Check the PTP status.

```
OS10# show ptp | no-more
PTP Clock : Boundary
Clock Identity : b0:4f:13:ff:ff:46:63:5f
```

(continues on next page)

(continued from previous page)

```
GrandMaster Clock Identity : fc:af:6a:ff:fe:02:bc:8d
Clock Mode : One-step
Clock Quality
Class : 135
Accuracy : <=100ns
Offset Log Scaled Variance : 65535
Domain : 24
Priority1 : 128
Priority2 : 128
Profile : G8275-1(Local-Priority:-128)
Steps Removed : 1
Mean Path Delay(ns) : 637
Offset From Master(ns) : 1
Number of Ports : 8
------------------------------------------------------------------------------
Interface State Port Identity
------------------------------------------------------------------------------
Ethernet1/1/1:1 Master b0:4f:13:ff:ff:46:63:5f:1
Ethernet1/1/3:1 Master b0:4f:13:ff:ff:46:63:5f:3
Ethernet1/1/5:1 Slave b0:4f:13:ff:ff:46:63:5f:5
Ethernet1/1/7:1 Master b0:4f:13:ff:ff:46:63:5f:8
Ethernet1/1/11 Master b0:4f:13:ff:ff:46:63:5f:4
Ethernet1/1/49 Master b0:4f:13:ff:ff:46:63:5f:9
Ethernet1/1/51 Master b0:4f:13:ff:ff:46:63:5f:10
Ethernet1/1/54 Master b0:4f:13:ff:ff:46:63:5f:2
------------------------------------------------------------------------------
Number of slave ports :1
Number of master ports :7
```

9. Save the switch configuration:

```
copy running-configuration startup-configuration
```

1.3.2.2.2 Chapter 2.2.2 Fibrolan Falcon RX Setup

Although the Fibrolan switch has not been qualified in the NVIDIA lab, OAI labs incorporate the following configuration and switch for interoperability.

To get started, follow the *Fibrolan Getting Started Guide*.

In this setup, the Qulsar GrandMaster is connected to port 4, the Aerial cuBB to port 17, and the Foxconn O-RU to port 16 (C/U plane) and port 15 (S/M plane). You can ignore all other ports in the figures[A][B] below.

1.3.2.2.2.1 VLAN Setup

The following assumes that the VLAN tag is 2 for both the control plane and the user plane of the O-RAN CU plane. VLAN tag 80 is used for everything else.

Open the configuration page of the Fibrolan switch, then go to **Configuration > VLANs**. Port 4 (the Qulsar GrandMaster) needs to be set to "Access" mode, with the port VLAN set to 80.

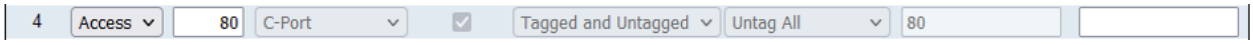


Fig. 1: Figure A - VLAN Setup

Use the same configuration for port 15 (RU S/M plane).

Configure ports 16 and 17 as follows:

- **Mode**: "Trunk"
- **Port**: VLAN 80
- **Untag Port VLAN**
- **Allowed VLANs**: 2, 80

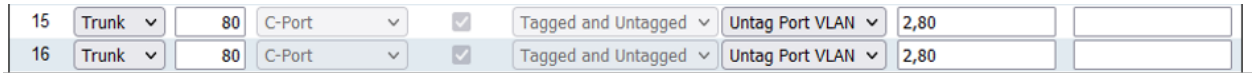
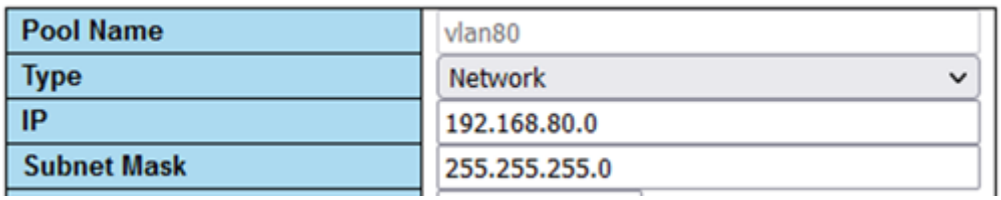


Fig. 2: Figure B - VLAN Setup

1.3.2.2.2.2 DHCP Setup

The RU M-plane requires you to set up a DHCP server. Go to **Configuration > DHCP > Server > Pool** and create a new DHCP server with the following settings:



1.3.2.2.2.3 PTP Setup on gNB

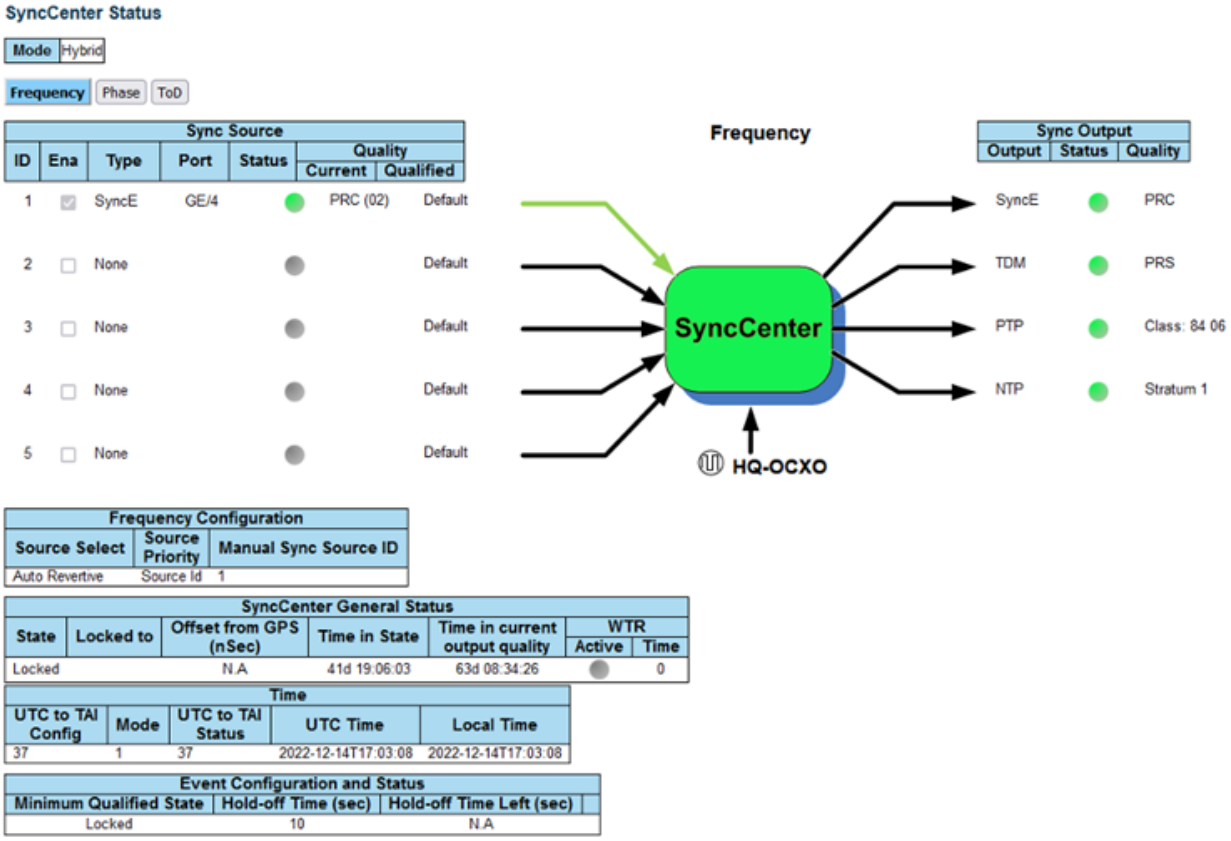
For the PTP setup, follow the Fibrolan *PTP Boundary Clock Configuration* guide and use the following settings:

- Device Type: "Ord-Bound"
- Profile: "G8275.1"
- Clock domain: 24
- VLAN: 80

Also make sure you enable the used ports (in this case, 4, 15, 16, and 17).

Hybrid mode is recommended as the sync mode.

If everything is configured correctly, the SyncCenter should show green.



1.3.2.3 Chapter 2.3 PTP Setup

These commands assume that PTP4L runs on the `ens6f0` NIC interface and uses CPU core 9. Core clash can cause problems, so if a different core is being used, it must not be used by L1 or L2+.

1.3.2.3.1 Verify Inbound PTP Packets

Typically, you should see packets with `ethertype 0x88f7` on the selected interface.

```
sudo tcpdump -i ens6f0 -c 5 | grep ethertype
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on ens6f1, link-type EN10MB (Ethernet), capture size 262144 bytes
13:27:41.291503 48:b0:2d:63:83:ac (oui Unknown) > 01:1b:19:00:00:00 (oui Unknown),
↪ethertype Unknown (0x88f7), length 60:
13:27:41.291503 48:b0:2d:63:83:ac (oui Unknown) > 01:1b:19:00:00:00 (oui Unknown),
↪ethertype Unknown (0x88f7), length 60:
13:27:41.296727 c4:5a:b1:14:1a:c6 (oui Unknown) > 01:1b:19:00:00:00 (oui Unknown),
↪ethertype Unknown (0x88f7), length 78:
13:27:41.296784 c4:5a:b1:14:1a:c6 (oui Unknown) > 01:1b:19:00:00:00 (oui Unknown),
↪ethertype Unknown (0x88f7), length 60:
```

(continues on next page)

(continued from previous page)

```
13:27:41.306316 08:c0:eb:71:e7:d5 (oui Unknown) > 01:1b:19:00:00:00 (oui Unknown),
↪ethertype Unknown (0x88f7), length 58:
```

1.3.2.3.2 Create ptp4l Configuration File

Paste these commands into the shell to create the three configuration files:

```
cat <<EOF | sudo tee /etc/ptp.conf
[global]
priority1 128
priority2 128
domainNumber 24
tx_timestamp_timeout 30
dscp_event 46
dscp_general 46
logging_level 6
verbose 1
use_syslog 0
logMinDelayReqInterval 1
[ens6f0]
logAnnounceInterval -3
announceReceiptTimeout 3
logSyncInterval -4
logMinDelayReqInterval -4
delay_mechanism E2E
network_transport L2
EOF

cat <<EOF | sudo tee /lib/systemd/system/ptp4l.service
[Unit]
Description=Precision Time Protocol (PTP) service
Documentation=man:ptp4l

[Service]
Restart=always
RestartSec=5s
Type=simple
ExecStart=/usr/bin/taskset -c 9 /usr/sbin/ptp4l -f /etc/ptp.conf

[Install]
WantedBy=multi-user.target
EOF
```

1.3.2.3.3 Create phc2sys Configuration File

```
# If more than one instance is already running, kill the existing
# PHC2SYS sessions.

# Command used can be found in /lib/systemd/system/phc2sys.service
# Update the ExecStart line to the following, assuming ens6f0 interface is used.
  sudo nano /lib/systemd/system/phc2sys.service
```

(continues on next page)

(continued from previous page)

```
[Unit]
Description=Synchronize system clock or PTP hardware clock (PHC)
Documentation=man:phc2sys
After=ntpdate.service
Requires=ptp4l.service
After=ptp4l.service

[Service]
Restart=always
RestartSec=5s
Type=simple
ExecStart=/bin/sh -c "taskset -c 9 /usr/sbin/phc2sys -s /dev/ptp$(ethtool -T ens6f0 |
↪grep PTP | awk '{print $4}') -c CLOCK_REALTIME -n 24 -O 0 -R 256 -u 256"

[Install]
WantedBy=multi-user.target
```

1.3.2.3.4 Enable and Start phc2sys and ptp4l

After changing the configuration files, they need to be reloaded, enabled, and restarted. These services can be restarted if they don't sync.

```
sudo systemctl daemon-reload
sudo systemctl enable ptp4l.service
sudo systemctl enable phc2sys.service

sudo systemctl restart phc2sys.service ptp4l.service

# check that the service is active and has low rms value (<30):
systemctl status ptp4l.service phc2sys.service
  ptp4l.service - Precision Time Protocol (PTP) service
     Loaded: loaded (/lib/systemd/system/ptp4l.service; enabled; vendor preset:
↪enabled)
     Active: active (running) since Tue 2023-05-09 13:21:12 UTC; 14s ago
       Docs: man:ptp4l
   Main PID: 6962 (ptp4l)
      Tasks: 1 (limit: 94588)
     Memory: 544.0K
     CGroup: /system.slice/ptp4l.service
             6962 /usr/sbin/ptp4l -f /etc/ptp.conf

May 09 13:21:17 aerial-rf-gb-gnb taskset[6962]: ptp4l[15552.609]: rms   15 max   32
↪freq   -639 +/-  25 delay   211 +/-   1
May 09 13:21:18 aerial-rf-gb-gnb taskset[6962]: ptp4l[15553.609]: rms   21 max   29
↪freq   -583 +/-  12 delay   210 +/-   1
May 09 13:21:19 aerial-rf-gb-gnb taskset[6962]: ptp4l[15554.609]: rms   11 max   21
↪freq   -576 +/-   8 delay   211 +/-   1
May 09 13:21:20 aerial-rf-gb-gnb taskset[6962]: ptp4l[15555.609]: rms    6 max   13
↪freq   -579 +/-   8 delay   211 +/-   1
May 09 13:21:21 aerial-rf-gb-gnb taskset[6962]: ptp4l[15556.609]: rms    4 max    7
↪freq   -578 +/-   6 delay   212 +/-   0
May 09 13:21:22 aerial-rf-gb-gnb taskset[6962]: ptp4l[15557.609]: rms    5 max   11
↪freq   -589 +/-   6 delay   213 +/-   1
May 09 13:21:23 aerial-rf-gb-gnb taskset[6962]: ptp4l[15558.609]: rms    6 max   12
```

(continues on next page)

(continued from previous page)

```
↪freq   -593 +/-   8 delay   210 +/-   1
May 09 13:21:24 aerial-rf-gb-gnb taskset[6962]: ptp4l[15559.609]: rms    3 max    7
↪freq   -587 +/-   5 delay   211 +/-   1
May 09 13:21:25 aerial-rf-gb-gnb taskset[6962]: ptp4l[15560.609]: rms    5 max   12
↪freq   -582 +/-   7 delay   212 +/-   1
May 09 13:21:26 aerial-rf-gb-gnb taskset[6962]: ptp4l[15561.609]: rms    4 max    7
↪freq   -587 +/-   7 delay   213 +/-   1

  phc2sys.service - Synchronize system clock or PTP hardware clock (PHC)
     Loaded: loaded (/lib/systemd/system/phc2sys.service; enabled; vendor preset:
↪enabled)
     Active: active (running) since Tue 2023-05-09 13:21:12 UTC; 14s ago
       Docs: man:phc2sys
   Main PID: 6963 (phc2sys)
      Tasks: 1 (limit: 94588)
     Memory: 572.0K
     CGroup: /system.slice/phc2sys.service
             6963 /usr/sbin/phc2sys -a -r -n 24 -R 256 -u 256

May 09 13:21:17 aerial-rf-gb-gnb phc2sys[6963]: [15553.320] CLOCK_REALTIME rms   42
↪max   79 freq  +8240 +/- 368 delay  1762 +/-  16
May 09 13:21:18 aerial-rf-gb-gnb phc2sys[6963]: [15554.336] CLOCK_REALTIME rms   35
↪max   64 freq  +8091 +/- 303 delay  1754 +/-  13
May 09 13:21:19 aerial-rf-gb-gnb phc2sys[6963]: [15555.352] CLOCK_REALTIME rms   27
↪max   52 freq  +8218 +/- 224 delay  1752 +/-  13
May 09 13:21:20 aerial-rf-gb-gnb phc2sys[6963]: [15556.368] CLOCK_REALTIME rms   21
↪max   49 freq  +8153 +/- 152 delay  1758 +/-  16
May 09 13:21:21 aerial-rf-gb-gnb phc2sys[6963]: [15557.384] CLOCK_REALTIME rms   17
↪max   39 freq  +8149 +/- 125 delay  1761 +/-  16
May 09 13:21:22 aerial-rf-gb-gnb phc2sys[6963]: [15558.400] CLOCK_REALTIME rms   14
↪max   33 freq  +8185 +/- 101 delay  1750 +/-  14
May 09 13:21:23 aerial-rf-gb-gnb phc2sys[6963]: [15559.416] CLOCK_REALTIME rms   12
↪max   32 freq  +8138 +/-  63 delay  1752 +/-  13
May 09 13:21:24 aerial-rf-gb-gnb phc2sys[6963]: [15560.431] CLOCK_REALTIME rms   11
↪max   43 freq  +8171 +/-  54 delay  1756 +/-  15
May 09 13:21:25 aerial-rf-gb-gnb phc2sys[6963]: [15561.447] CLOCK_REALTIME rms   10
↪max   32 freq  +8163 +/-  38 delay  1762 +/-  16
May 09 13:21:26 aerial-rf-gb-gnb phc2sys[6963]: [15562.463] CLOCK_REALTIME rms    9
↪max   23 freq  +8162 +/-  17 delay  1761 +/-  16
```

1.3.2.3.5 Disable NTP

Use these commands to turn off NTP:

```
sudo timedatectl set-ntp false
timedatectl
Local time: Thu 2022-02-03 22:30:58 UTC
           Universal time: Thu 2022-02-03 22:30:58 UTC
                 RTC time: Thu 2022-02-03 22:30:58
                Time zone: Etc/UTC (UTC, +0000)
System clock synchronized: no
              NTP service: inactive
          RTC in local TZ: no
```

1.3.2.3.6 Verify System Clock Synchronization

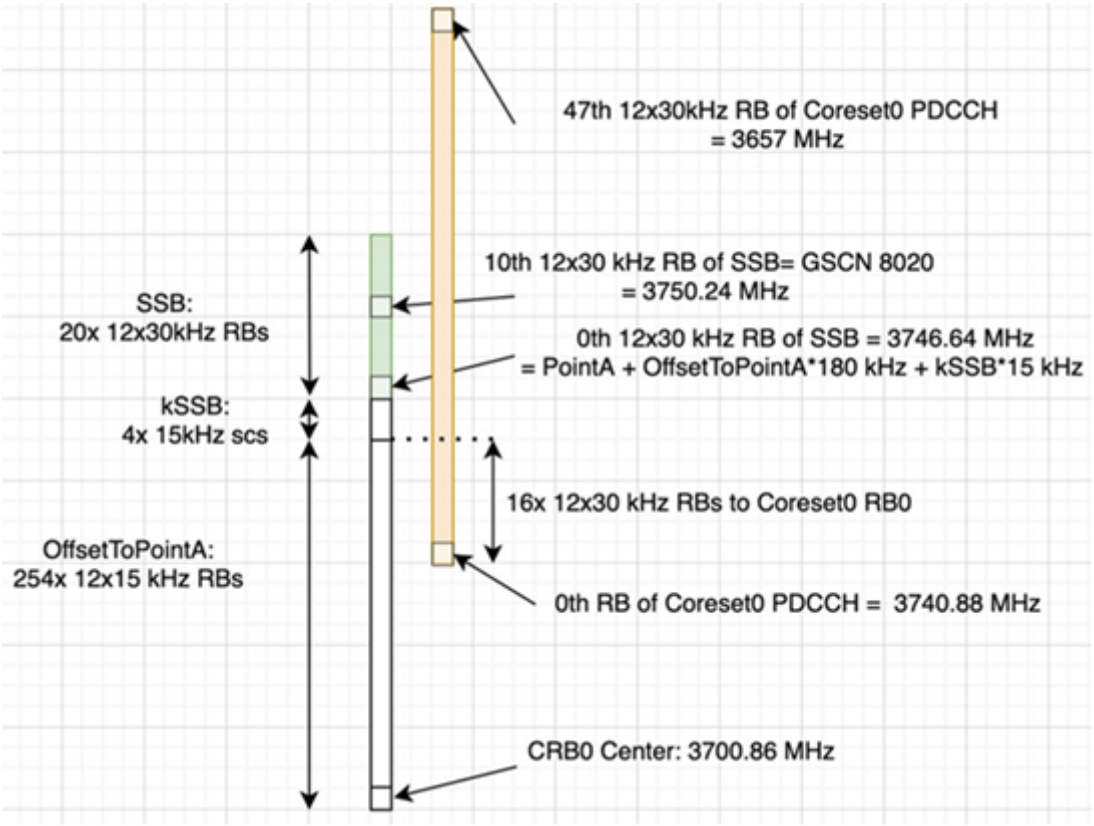
Make NTP inactive and synchronize the system clock:

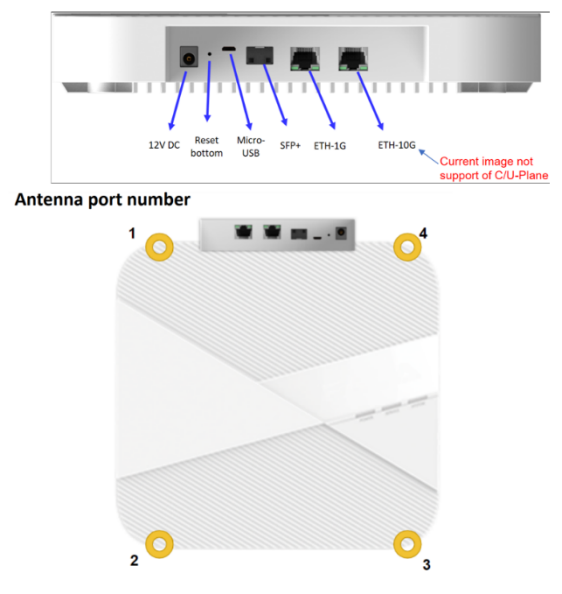
```
timedatectl
Local time: Thu 2022-02-03 22:30:58 UTC
           Universal time: Thu 2022-02-03 22:30:58 UTC
                 RTC time: Thu 2022-02-03 22:30:58
                Time zone: Etc/UTC (UTC, +0000)
System clock synchronized: yes
              NTP service: inactive
          RTC in local TZ: no
```

1.3.2.4 Chapter 2.4 Set up the Foxconn ORU

Tip

There is a *tutorial video* for setting up the Foxconn ORU.



Foxconn RPQN-7801E	Connections and Settings
	Connections: ▶ **10SFP**: C/U plane (will support S/M plane after firmware upgrade) ▶ **1G RJ45**: S/M plane ▶ **10G RJ45**: POE only ▶ **Micro-USB**: USB to serial for debugging (115200, 8, 1, none, flow control off) GrandMaster settings (Qulsar): ▶ **PTP timing port**: Disable VLAN ▶ **Two steps**: OFF ▶ **Domain number**: 24 (needs to configured on O-RU) ▶ IPv4, Unicast, etc. `/home/root/sdcard/RRHconfig_xran.xml`: ▶ `RRH_PTPV2_GRAND_MASTER_IP = 20.0.0.8` ▶ `RRH_PTPV2_SUB_DOMAIN_NUM = 24` ▶ C/U plane VLAN tag ▶ `RRH_LO_FREQUENCY_KHZ = 3750000`

1.3.2.4.1 Configure VLAN and IP Address on the gNB Server

1. Add these commands to the server startup script (`/etc/rc.local`) so they are automatically run on reboot.
2. Configure these settings on the fronthaul port.
3. You must use IP addresses that do not match those in the example below:

```
sudo ip link add link ens6f0 name ens6f0.2 type vlan id 2
sudo ip addr add 169.254.1.103/24 dev ens6f0.2
```

(continues on next page)

(continued from previous page)

```
sudo ip link set up ens6f0.2
```

1.3.2.4.2 O-RU M-Plane Setup

1. Add the following to the bottom of `/etc/profile` and comment out the line with `set_qse.sh` if it already exists. Set the interface initially to `eth0` for firmware version 1, and to `qse-eth` after upgrading to firmware version 2 or greater.

```
interface=eth0
vlanid=2
ipLastOctet=20

ip link add link ${interface} name ${interface}.$vlanid type vlan id $vlanid
ip addr flush dev ${interface}
ip addr add 169.254.0.0/24 dev ${interface}
ip addr add 169.254.1.${ipLastOctet}/24 dev ${interface}.$vlanid
ip link set up ${interface}.$vlanid
```

2. Reboot the O-RU using the command `./reboot.sh` and check the network configuration:

```
# ip r
169.254.1.0/24 dev eth0.2 src 169.254.1.20
```

1.3.2.4.3 Update O-RU Configuration

Note

If you are using the CBRS O-RU (Foxconn RPQN-4800E), refer to the note below for the modified configuration.

1. Update the O-RU configuration in `/home/root/test/RRHconfig_xran.xml`.

```
root@arria10:~/test# grep -v '<!-' RRHconfig_xran.xml
RRH_DST_MAC_ADDR = 08:c0:eb:71:e7:d4 # To match fronthaul interface of DU
RRH_SRC_MAC_ADDR = 6C:AD:AD:00:04:6C # To match qse-eth of RU
RRH_EN_EAXC_ID = 0
RRH_EAXC_ID_TYPE1 = 0x0, 0x1, 0x2, 0x3
RRH_EAXC_ID_TYPE3 = 0x8, 0x9, 0xA, 0xB
RRH_EN_SPC = 1
RRH_RRH_LTE_OR_NR = 1
RRH_TRX_EN_BIT_MASK = 0x0f
RRH_RF_EN_BIT_MASK = 0x0f
RRH_CMPR_HDR_PRESENT = 0
RRH_CMPR_TYPE = 1, 1
RRH_CMPR_BIT_LENGTH = 9, 9
RRH_UL_INIT_SYM_ID = 0
RRH_TX_TRUNC_BITS = 4
RRH_RX_TRUNC_BITS = 4
RRH_MAX_PRB = 273
RRH_C_PLANE_VLAN_TAG = 0x0002 #To match vlan id set in cuphycontroller yaml file
```

(continues on next page)

(continued from previous page)

```
RRH_U_PLANE_VLAN_TAG = 0x0002 #To match vlan id set in cuphycontroller yaml file
RRH_SLOT_TICKS_IN_SEC = 2000
RRH_SLOT_PERIOD_IN_SAMPLE = 61440
RRH_LO_FREQUENCY_KHZ = 3750000, 0
RRH_TX_POWER = 24, 24
RRH_TX_ATTENUATION = 12.0, 12.0, 12.0, 12.0
RRH_RX_ATTENUATION = 0.0, 0.0, 0.0, 0.0
RRH_BB_GENERAL_CTRL = 0x0, 0x0, 0x0, 0x0
RRH_RF_GENERAL_CTRL = 0x3, 0x1, 0x0, 0x0
RRH_PTPV2_GRAND_MASTER_MODE = 3
RRH_PTPV2_JITTER_LEVEL = 0
RRH_PTPV2_VLAN_ID = 0
RRH_PTPV2_IP_MODE = 4
RRH_PTPV2_GRAND_MASTER_IP = 192.167.27.150
RRH_PTPV2_SUB_DOMAIN_NUM = 24
RRH_PTPV2_ACCEPTED_CLOCK_CLASS = 135
RRH_TRACE_PERIOD = 10
RRH_DL_IQ_SCALING = 0x1001
RRH_CFR_PEAK_THRESHOLD = 0.5
```

Note

In Foxconn firmware version 2.6.9, the configuration file is located in `/home/root/sdcard`.

Note

The above configuration was taken from an ORU running firmware 3.1.15.

Note

If you're using the CBRS O-RU (Foxconn RPQN-4800E), the above parameters should be modified as follows:

- n78:

  `RRH_LO_FREQUENCY_KHZ = 3750000, 0`

- n48 (CBRS):

  `RRH_LO_FREQUENCY_KHZ = 3649140, 0`

2. Reboot O-RU.

```
cd /home/root/test/

./reboot
```

3. Run the following to enable the configuration:

```
cd /home/root/test/

./init_rrh_config_enable_cuplane
```

4. To see the ORU status run the following script.

```
cd /home/root/test/

./chk_con.sh
```

1.3.3. Part 3. Configure the gNB Server

To install the Aerial tools, follow the the cuBB installation guide; refer to the *Software Manifest* for a link to the cuBB documentation.

1.3.3.1 Configure the gNB Server - Gigabyte Edge E251-U70

To install the Aerial cuBB tools, follow the Gigabyte cuBB installation guide.

The ARC-OTA thread-to-core assignment functionality is different from a standard Aerial installation. Layer 1 threads need to be isolated in a monolithic block and have been moved, with the rest left to layers 2 and higher: Cores 2-6 are used as cuPHY worker cores; core 7 is used for the cuPHY `lowprio` thread; core 8 is used for the cuPHY timer thread; and core 9 is used for PTP and PHC2SYS.

1.3.3.1.1 Configure Linux Kernel Command-Line for ARC-OTA

To set kernel command-line parameters, edit the `GRUB_CMDLINE_LINUX_DEFAULT` parameter in the grub file `/etc/default/grub` and modify the following parameters.

Note

The following kernel parameters are optimized for the Gigabyte server with 24 cores Xeon Gold 6240R and 96GB memory. For ARC-OTA, typically it is optimal to configure the gNB to isolate cores 2 to 10 for Aerial, and leave the other cores for OAI L2+.

```
default_hugepagesz=1G
hugepagesz=1G
hugepages=16
tsc=reliable
clocksource=tsc
intel_idle.max_cstate=0
mce=ignore_ce processor.max_cstate=0
intel_pstate=disable
audit=0
idle=poll
isolcpus=2-10
rcu_nocb_poll
nosoftlockup
iommu=off
intel_iommu=off
irqaffinity=0-1,22-23
```

To automatically append the grub file with these changes, use this command:

```
sudo sed -i 's/^GRUB_CMDLINE_LINUX_DEFAULT="[^"]*/& default_hugepagesz=1G
↪hugepagesz=1G hugepages=16 tsc=reliable clocksource=tsc intel_idle.max_cstate=0
↪mce=ignore_ce processor.max_cstate=0 intel_pstate=disable audit=0 idle=poll
↪isolcpus=2-10 rcu_nocb_poll nosoftlockup iommu=off intel_iommu=off irqaffinity=0-1,
↪22-23/' /etc/default/grub
```

1.3.3.1.2 Apply the Changes

1. Use the following commands to apply the changes and reboot to load the kernel.

   ```
   sudo update-grub
   sudo reboot
   ```

2. After rebooting, enter the following command to check whether the system has booted into the low-latency kernel:

   ```
   uname -r
    5.15.0-1042-nvidia-lowlatency
   ```

3. Enter this command to check that the kernel command-line parameters are configured correctly:

   ```
   cat /proc/cmdline
   BOOT_IMAGE=/vmlinuz-5.15.0-1042-nvidia-lowlatency root=/dev/mapper/ubuntu-
   ↪-vg-ubuntu--lv ro default_hugepagesz=1G hugepagesz=1G hugepages=16
   ↪tsc=reliable clocksource=tsc intel_idle.max_cstate=0 mce=ignore_ce
   ↪processor.max_cstate=0 intel_pstate=disable audit=0 idle=poll
   ↪isolcpus=2-10 rcu_nocb_poll nosoftlockup iommu=off intel_iommu=off
   ↪irqaffinity=0-1,22-23
   ```

1.3.3.1.3 Change Core for ptp4l and phc2sys

Edit the `/lib/systemd/system/ptp4l.service` file as follows:

```
ExecStart=taskset -c 9 /usr/sbin/ptp4l -f /etc/ptp.conf
```

Edit the `/lib/systemd/system/phc2sys.service` file as follows:

```
ExecStart=/bin/sh -c "taskset -c 9 /usr/sbin/phc2sys -s /dev/ptp$(ethtool -T ens6f0 |
↪grep PTP | awk '{print $4}') -c CLOCK_REALTIME -n 24 -O 0 -R 256 -u 256"
```

1.3.3.2 Configure gNB Server - Dell R750

To install the Aerial cuBB tools, follow the Dell R750 cuBB installation guide.

The ARC-OTA thread-to-core assignment functionality is different from a standard Aerial installation. Layer 1 threads need to be isolated in a monolithic block and have been moved, with the rest left to layers 2 and higher. Cores [5,7,9,11,13] are used as cuPHY workers cores. Core 17 is used for the cuPHY `lowprio` thread; core 15 is used for the cuPHY timer thread; and core 19 is used for PTP and PHC2SYS.

1.3.3.2.1 Configure the Linux Kernel Command Line for ARC-OTA

To set kernel command-line parameters, modify the following parameters under the GRUB_CMDLINE_LINUX_DEFAULT parameter in the grub file (`/etc/default/grub`).

```
pci=realloc=off
default_hugepagesz=1G
hugepagesz=1G
hugepages=16
tsc=reliable
clocksource=tsc
intel_idle.max_cstate=0
mce=ignore_ce
processor.max_cstate=0
intel_pstate=disable
audit=0
idle=poll
rcu_nocb_poll
nosoftlockup
iommu=off
irqaffinity=0-3,44-47
isolcpus=5,7,9,11,13,15,17,19,21
noht
```

To automatically append the grub file with these changes, use this command:

```
sudo sed -i 's/^GRUB_CMDLINE_LINUX_DEFAULT="[^"]*/& pci=realloc=off default_
↪hugepagesz=1G hugepagesz=1G hugepages=16 tsc=reliable clocksource=tsc intel_idle.
↪max_cstate=0 mce=ignore_ce processor.max_cstate=0 intel_pstate=disable audit=0
↪idle=poll rcu_nocb_poll nosoftlockup iommu=off irqaffinity=0-3,44-47 isolcpus=5,7,9,
↪11,13,15,17,19,21 noht/' /etc/default/grub
```

1.3.3.2.2 Apply the Changes and Load the Kernel

1. Use the following commands to apply the command-line changes and reboot the system.

   ```
   sudo update-grub
   sudo reboot
   ```

2. After rebooting, use the following command to check whether the system has booted into the low-latency kernel:

   ```
   uname -r
    5.15.0-1042-nvidia-lowlatency
   ```

3. Enter this command to check that the kernel command-line parameters are configured correctly:

   ```
   cat /proc/cmdline
   BOOT_IMAGE=/vmlinuz-5.15.0-1042-nvidia-lowlatency root=/dev/mapper/ubuntu-
   ↪-vg-ubuntu--lv ro pci=realloc=off default_hugepagesz=1G hugepagesz=1G
   ↪hugepages=16 tsc=reliable clocksource=tsc intel_idle.max_cstate=0
   ↪mce=ignore_ce processor.max_cstate=0 intel_pstate=disable audit=0
   ↪idle=poll rcu_nocb_poll nosoftlockup iommu=off irqaffinity=0-3,44-47
   ↪isolcpus=5,7,9,11,13,15,17,19,21 noht
   ```

1.3.3.2.3 Change the Core for ptp4l and phc2sys

Edit the `/lib/systemd/system/ptp4l.service` file:

```
ExecStart=taskset -c 19 /usr/sbin/ptp4l -f /etc/ptp.conf
```

Edit the `/lib/systemd/system/phc2sys.service` file:

```
ExecStart=/bin/sh -c "taskset -c 19 /usr/sbin/phc2sys -s /dev/ptp$(ethtool -T
↪enp204s0f1np1 | grep PTP | awk '{print $4}') -c CLOCK_REALTIME -n 24 -O 0 -R 256 -u
↪256"
```

1.3.3.3 Configure gNB Server - SMC Grace Hopper MGX

To install the Aerial cuBB tools, follow the same steps as the Grace Hopper MGX installation guide.

The ARC-OTA thread-to-core assignment functionality is different from a standard Aerial installation. Layer 1 threads need to be isolated in a monolithic block and have been moved, with the rest left to layers 2 and higher. Cores [5,7,9,11,13] are used as cuPHY worker cores. Core 17 is used for the cuPHY `lowprio` thread; core 15 is used for the cuPHY timer thread; and core 41 is used for PTP and PHC2SYS.

1.3.3.3.1 Configure the Linux Kernel Command Line for ARC-OTA

To set kernel command-line parameters, edit the `GRUB_CMDLINE_LINUX` parameter in the grub file `/etc/default/grub.d/cmdline.cfg` and append or update the parameters described below. The following kernel parameters are optimized for GH200. To automatically append the grub file with these parameters, use this command:

```
cat <<"EOF" | sudo tee /etc/default/grub.d/cmdline.cfg
GRUB_CMDLINE_LINUX="$GRUB_CMDLINE_LINUX pci=realloc=off pci=pcie_bus_safe default_
↪hugepagesz=512M hugepagesz=512M hugepages=32 tsc=reliable processor.max_cstate=0
↪audit=0 idle=poll rcu_nocb_poll nosoftlockup irqaffinity=0 isolcpus=managed_irq,
↪domain,4-47 nohz_full=4-47 rcu_nocbs=4-47 earlycon module_blacklist=nouveau acpi_
↪power_meter.force_cap_on=y numa_balancing=disable init_on_alloc=0 preempt=none"
EOF
```

1.3.3.3.2 Apply the Changes and Load the Kernel

1. Use the following commands to apply the command-line changes and reboot the system.

   ```
   sudo update-grub
   sudo reboot
   ```

2. After rebooting, use the following command to check whether the system has booted into the low-latency kernel:

   ```
   uname -r
   6.2.0-1012-nvidia-64k
   ```

3. Enter this command to check that the kernel command-line parameters are configured correctly:

```
cat /proc/cmdline
BOOT_IMAGE=/vmlinuz-6.2.0-1012-nvidia-64k root=/dev/mapper/ubuntu--vg-
↪ubuntu--lv ro pci=realloc=off pci=pcie_bus_safe default_hugepagesz=512M
↪hugepagesz=512M hugepages=32 tsc=reliable processor.max_cstate=0
↪audit=0 idle=poll rcu_nocb_poll nosoftlockup irqaffinity=0
↪isolcpus=managed_irq,domain,4-47 nohz_full=4-47 rcu_nocbs=4-47 earlycon
↪module_blacklist=nouveau acpi_power_meter.force_cap_on=y numa_
↪balancing=disable init_on_alloc=0 preempt=none
```

1.3.3.3.3 Change the Core for ptp4l and phc2sys

Edit the `/lib/systemd/system/ptp4l.service` file:

```
ExecStart=taskset -c 41 /usr/sbin/ptp4l -f /etc/ptp.conf
```

Edit the `/lib/systemd/system/phc2sys.service` file:

```
ExecStart=/bin/sh -c "taskset -c 41 /usr/sbin/phc2sys -s /dev/ptp\$(ethtool -T
↪aerial00 | grep PTP | awk '{print \$4}') -c CLOCK_REALTIME -n 24 -O 0 -R 256 -u 256"
```

1.3.4. Part 4. Install ARC-OTA Using SDK Manager

You can use NVIDIA SDK Manager to install directly on the target machine or from a remote machine. SDKM uses SSH to connect to the server where it installs the gNB.

Steps to install ARC-OTA using SDK Manager can be found in the SDK Manager documentation

You can download the installer at https://developer.download.nvidia.com/sdkmanager/redirects/sdkmanager-deb.html

1.3.4.1 Prerequisites for Installing gNB with SDK Manager

Use the Aerial cuBB Installation Guide for the following steps:

1. Configure BIOS Settings
2. Install Ubuntu 22.04 Server
3. Install the Low-Latency Kernel
4. Configure Linux Kernel Command-line

1.3.4.2 Post-Installation Steps

1. Configure OAI L2.

   IP addresses of the CN and the gNB must be configured using one of the following:

   - Update gnb_cn5g_network_config.ini file with the actual network setting values. Then run gnb_apply_network_config.sh to apply the settings
   - Or follow the ARC-OTA installation guide steps to do it manually

2. Jump to **Part 5 – Validate the Setup**.

   Follow the steps in the ARC-OTA Installation Guide.

1.3.5. Part 5. Validate the Setup

In this section, you will validate the ARC-OTA setup using bi-directional UDP.

1.3.5.1 Step 1: Add the SIM User Profile

Modify the following files:

- `oai_db.sql`

  There are 3 UEs pre-configured in this file. To find them, search for `001010000000001` and add or edit them as needed.

- `./targets/PROJECTS/GENERIC-NR-5GC/CONF/gnb-vnf.sa.band78.273prb.aerial.conf`

  Modify this file on the gNB server if you want to change the MCC and MNC in the gNB config file.

1.3.5.2 Step 2: Setup the UE and SIM Card

For reference, use the following: SIM cards – 4G and 5G reference software (open-cells.com)

Program the SIM Card with the Open Cells Project application “uicc-v2.6”, which can be downloaded here.

Use the ADM code specific to the SIM card. If the wrong ADM is used 8 times, the SIM card will be permanently locked.

```
sudo ./program_uicc --adm 12345678 --imsi 001010000000001 --isdn 00000001 --acc 0001 -
↪-key fec86ba6eb707ed08905757b1bb44b8f --opc C42449363BBAD02B66D16BC975D77CC1 -spn
↪"OpenAirInterface" --authenticate
Existing values in USIM
ICCID: 89860061100000000191
WARNING: iccid luhn encoding of last digit not done
USIM IMSI: 208920100001191
USIM MSISDN: 00000191
USIM Service Provider Name: OpenCells191
Setting new values
Reading UICC values after uploading new values
ICCID: 89860061100000000191
WARNING: iccid luhn encoding of last digit not done
USIM IMSI: 001010000000001
USIM MSISDN: 00000001
USIM Service Provider Name: OpenAirInterface
Succeeded to authentify with SQN: 64
set HSS SQN value as: 96
```

1.3.5.2.1 CUE Configuration Setup

Install the “Magic IPERF” application on the UE:

1. To test with CUE, a test SIM card with **Milenage** support is required. The following must be provisioned on the SIM card and must match the Core Network settings: mcc, mnc, IMSI, Ki, OPc.
2. The APN on the CUE should be configured according to the Core Network settings.
3. Start the DNS. Core Network should assign a mobile IP address and DNS. If DNS is not assigned, set the DNS with the other Android app.

1.3.5.3 Step 3. Running End-to-End OTA

This section describes how to run end-to-end traffic from the UE to the edge Core Network.

> **Note**
>
> The UE can connect as close as 2-3 meters, with a maximum range of 10-15 meters. The connection distance outside of buildings has not been unverified.

1.3.5.3.1 Start CN5G Core Network

Use the following commands to start the CN5G Core Network.

```
sudo sysctl net.ipv4.conf.all.forwarding=1
sudo iptables -P FORWARD ACCEPT

cd ~/oai-cn5g
docker compose up -d
```

1.3.5.3.1.1 Start theCN5G Edge Application

After the CN5G is started, use the `oai-ext-dn` container to run IPERF.

```
docker exec -it oai-ext-dn /bin/bash
```

1.3.5.3.2 Start Aerial cuBB on the gNB

Execute the following command to set up the cuBB container.

```
# Run on host: start a docker terminal
docker exec -it $AERIAL_CUBB_CONTAINER /bin/bash
```

Follow the Aerial cuBB documentation to build and run the cuphycontroller. The following instructions are for building and setting up the environment for running cuphycontroller. The following commands must be run from inside the cuBB container.

```
cd /opt/nvidia/cuBB
export cuBB_SDK=$(pwd)
mkdir build && cd build
cmake ..
make -j
```

The Aerial cuPHY configuration files are located in the `/opt/nvidia/cuBB/cuPHY-CP/cuphycontroller/config` directory. For ARC-OTA 1.5, the setup has been validated with `cuphycontroller_P5G_FXN.yaml` for the Gigabyte server and with `cuphycontroller_P5G_FXN_R750.yaml` for the Dell R750 server.

Before running the cuphycontroller, edit the `cuphycontroller_<xyz>.yaml` configuration file to point to the correct MAC address of the ORU and the correct PCIe address of the FH interface on the gNB (the following example applies to the Gigabyte server).

```
sed -i "s/ nic:.*/ nic: 0000:b5:00.0/" ${cuBB_SDK}/cuPHY-CP/cuphycontroller/config/
↪cuphycontroller_<xyz>.yaml
sed -i "s/ dst_mac_addr:.*/ dst_mac_addr: 6c:ad:ad:00:02:02/" ${cuBB_SDK}/cuPHY-CP/
↪cuphycontroller/config/cuphycontroller_<xyz>.yaml
```

When the build is done and the configuration files are updated, exit the container. Run the following to commit the changes to a new image. The name of the image will be used later in the Docker Compose configuration.

```
docker commit $AERIAL_CUBB_CONTAINER cubb-build:24-1
```

1.3.5.3.3 Creating the NVIPC Source Code Package

In the latest release, NVIPC is no longer included in the OAI repository. You must copy the source package from the Aerial cuBB container and add it to the OAI build files. Execute the following to create the `nvipc_src.<data>.tar.gz` file.

```
docker exec -it $AERIAL_CUBB_CONTAINER ./cuPHY-CP/gt_common_libs/pack_nvipc.sh
docker cp $AERIAL_CUBB_CONTAINER:/opt/nvidia/cuBB/cuPHY-CP/gt_common_libs/nvipc_src.
↪<date>.tar.gz ~/openairinterface5g
```

This will create and copy the `nvipc_src.<data>.tar.gz` archive that is required to build OAI L2+ for Aerial. Instructions can also be found in the Aerial FAPI.

1.3.5.3.4 Build gNB Docker Image

In ARC-OTA , the OAI image is built in two steps. Instructions can be found in the Aerial FAPI.

Note

When building a Docker image, the files are copied from the filesystem into the image. After you build the image, you must make changes to the configuration inside the container.

1.3.5.3.5 Pre-build Steps for OAI L2

1. When building OAI L2 for Aerial 24-1, remove line 50 and 51 in the `docker/Dockerfile.gNB.aerial.ubuntu20` file before building. (ARC-OTA 1.5. includes the `OAI 2024.w21` tag and Aerial CUDA-Accelerated RAN (Layer 1) 24-1). This will be merged to OAI `develop` branch in the future.
2. In the same file, add `moreutils` on line 79.

```
moreutils \
```

3. Remove the pinning of `p7_thread` in the OAI FAPI integration (`nfapi/oai_integration/aerial/fapi_vnf_p5.c`). To do so, remove lines 59-64 (this change will be merged to the OAI `develop` branch in the future). Core 8 is occupied by L1 on the Gigabyte server. On Dell R750, this core is free, but gNB is usually started on numa 0 with odd cores.

```
//CPU_SET(8, &cpuset);
//s = pthread_setaffinity_np(pthread_self(), sizeof(cpu_set_t), &cpuset);
//if (s != 0)
//  printf("failed to set afinity\n");
```

4. Ensure the the FAPI polling thread is run on an isolated core by modifying line 609 of the `nfapi/oai_integration/aerial/fapi_nvIPC.c` file.

   On the Gigabyte server and Dell R750 server, use core 21:

```
stick_this_thread_to_core(21)
```

5. There is also a bug regarding PDU length truncation in the `OAI 2024.w21` tag used with ARC 1.5. This is already fixed on the `develop` branch of OAI. The current ARC release has been verified by applying this MR on top of the 2024.w21 `OAI 2024.w21` tag.

```
cd ~/openairinterface5g/
wget https://gitlab.eurecom.fr/oai/openairinterface5g/-/merge_requests/2797.patch
git apply -3 2797.patch
```

Use the below commands to build the OAI L2 code:

```
cd ~/openairinterface5g/
docker build . -f docker/Dockerfile.base.ubuntu20 --tag ran-base:latest
docker build . -f docker/Dockerfile.gNB.aerial.ubuntu20 --tag oai-gnb-aerial:latest
```

This will create two images:

- `ran-base:latest` includes the environment to build the OAI source code. This will be used then building the `oai-gnb-aerial:latest` image.
- `oai-gnb-aerial:latest` will be used later in the Docker Compose script to create the OAI L2 container.

1.3.5.3.6 Running gNB with Docker Compose

ARC-OTA includes a Docker Compose configuration for running gNB software. Refer to the OAI instructions on how to run with Docker Compose. The Docker Compose configuration is located in the following file path:

```
~/openairinterface5g/ci-scripts/yaml_files/sa_gnb_aerial/docker-compose.yaml
```

Before starting the gNB, you will need to update the `docker-compose.yaml` file.

1. On line 27, switch the cuBB image to the image that was built, committed, and tagged in the previous steps.

   ```
   image: cubb-build:24-1
   ```

2. On line 30, add `sudo` to the command.

   ```
   command: bash -c "sudo rm -rf /tmp/phy.log && sudo chmod +x /opt/nvidia/cuBB/
   ↪aerial_l1_entrypoint.sh && /opt/nvidia/cuBB/aerial_l1_entrypoint.sh"
   ```

3. On line 37, if you follow the guide, change the `cpu_set` for the OAI container as follows.

   a. Gigabyte server:

   ```
   cpu_set: "13-20"
   ```

   b. Dell R750 server:

   ```
   cpu_set: "23,25,27,29,31,33,35,37"
   ```

   c. SMC-GH server:

   ```
   cpuset: "11-18"
   ```

4. On line 60, add a volume for the `oai.log` file.

   ```
   - /var/log/aerial:/var/log/aerial
   ```

5. On line 61 (after the volumes), add a command for executing the `nr-softmodem`. This will replace the default command that is integrated in the docker image. This new command will set the priority and create an `oai.log` file with time stamp.

   ```
   command: bash -c "chrt -f 99 /opt/oai-gnb/bin/nr-softmodem -O /opt/oai-gnb/etc/
   ↪gnb.conf | ts | tee /var/log/aerial/oai.log"
   ```

After following the instructions, you should have the following images:

```
$ docker image ls
REPOSITORY                                                    TAG
↪                                    IMAGE ID       CREATED          SIZE
cubb-build                                                    24-1
↪                                    be7a5a94f2d3   10 seconds ago   26GB
nvcr.io/qhrjhjrvlsbu/aerial-cuda-accelerated-ran              24-1-cubb
↪                                    a66cf2a45fad   2 months ago    25.5GB
oai-gnb-aerial                                                latest
↪                                    0856b9969f42   3 weeks ago      4.88GB
ran-base                                                      latest
↪                                    c5d060d23529   3 weeks ago      2.42GB
```

Docker Compose will start containers running cuBB and OAI L2+. The Docker Compose script includes an entry-point script for cuBB, which you need to modify before running. The script points at the `cuphycontroller_xxx.yaml` configuration that you want to run. This script is located in the following file path:

```
~/openairinterface5g/ci-scripts/yaml_files/sa_gnb_aerial/aerial_l1_entrypoint.sh
```

Before running Docker Compose, update the `aerial_l1_entrypoint.sh` file.

1. The latest ARC-OTA release does not have to build `gdrcopy`.

   ```
   # cd "$cuBB_Path"/cuPHY-CP/external/gdrcopy/ || exit 1
   # ./insmod.sh
   ```

2. In the latest ARC-OTA release, you are not root by default. Add `sudo` to the following, including `P5G_FXN_R750` if you are using a Dell R750 or P5G_FXN if you are using .

   ```
   sudo -E "$cuBB_Path"/build/cuPHY-CP/cuphycontroller/examples/cuphycontroller_scf
   ↪P5G_FXN_R750
   ```

3. Before running the Docker Compose script, you also need to add root for some commands that are run before starting Aerial cuPHY Layer 1. To do so, edit the `ci-scripts/yaml_files/sa_gnb_aerial/docker-compose.yaml` file.

   ```
   command: bash -c " sudo rm -rf /tmp/phy.log && sudo chmod +x /opt/nvidia/cuBB/
   ↪aerial_l1_entrypoint.sh && /opt/nvidia/cuBB/aerial_l1_entrypoint.sh"
   ```

   Also, OAI L2 must point to the correct configuration by editing the following row in the `ci-scripts/yaml_files/sa_gnb_aerial/docker-compose.yaml` file. All L2 configurations can be found in `targets/PROJECTS/GENERIC-NR-5GC/CONF`.

   ```
   - ../../../targets/PROJECTS/GENERIC-NR-5GC/CONF/gnb-vnf.sa.band78.273prb.aerial.
   ↪conf:/opt/oai-gnb/etc/gnb.conf
   ```

4. On line 37 in `ci-scripts/yaml_files/sa_gnb_aerial/docker-compose.yaml`, specify the cores to use for OAI l2.

   a. Gigabyte server:

      ```
      cpuset: "13-20"
      ```

   b. Dell R750 server:

      ```
      cpuset: "23,25,27,29,31,33,35,37"
      ```

   c. SMC-GH server:

      ```
      cpuset: "11-18"
      ```

You can now run ARC-OTA with the below commands.

```
cd ~/openairinterface5g/ci-scripts/yaml_files/sa_gnb_aerial
docker compose up -d
// console of cuBB
docker logs -f nv-cubb
// console of oai
docker logs -f oai-gnb-aerial
// tail -f /var/log/aerial/oai.log
```

1.3.5.3.7 CUE Connecting to 5G Network

Take the CUE out of airplane mode to start attaching the UE to the network. Make sure that the CUE is in airplane mode before starting OAI L2 stack.

1.3.5.3.7.1 Observe 5G Connect Status

Refer to the Preamble log in the cuphycontroller console output.

Check the Core Network log or CUE log to see whether NAS authentication and PDU session succeeded.

1.3.5.3.8 Running E2E IPERF Traffic

Start `ping`, `iperf`, or other network app tests after the PDU session connects successfully.

You can install and run the “Magic IPerf” Android application on the CUE for this purpose.

1.3.5.3.8.1 Ping Test

Ping the UE from the CN:

```
docker exec -it oai-ext-dn ping 10.0.0.2
```

Ping from the UE to the CN:

```
ping -I 10.0.0.2 192.168.70.135
```

1.3.5.3.8.2 Iperf Downlink Test

Perform Iperf downlink test on the UE Side:

```
iperf3 -s
```

Perform Iperf downlink test on the CN5G Side:

```
# Test UDP DL
docker exec -it oai-ext-dn iperf3 -u -P 13 -b 80M -t 60 -c 10.0.0.2
#Test UDP bidirectional
docker exec -it oai-ext-dn iperf3 -u --bidir -P 13 -b 80M -t 60 -c 10.0.0.2
# Test TCP DL
docker exec -it oai-ext-dn iperf3 -P 13 -b 80M -t 60 -c 10.0.0.2
#Test TCP bidirectional
docker exec -it oai-ext-dn iperf3 --bidir -P 13 -b 80M -t 60 -c 10.0.0.2
```

1.3.5.3.8.3 IPERF Uplink Test

Perform Iperf uplink test on the UE Side:

```
iperf3 -s
```

Perform Iperf uplink test on the CN5G Side:

```
#UDP
docker exec -it oai-ext-dn iperf3 -u -R -b 130M -t 60 -c 10.0.0.2
#TCP
docker exec -it oai-ext-dn iperf3 -R -b 130M -t 60 -c 10.0.0.2
```

To stop the containers, use the following commands:

```
docker stop $OAI_GNB_CONTAINER
docker rm $OAI_GNB_CONTAINER
```

Note

ARC-OTA is a P5G cellular network; specific enterprise switching/routing/firewalls/policies might need integration support to enable access to the World Wide Web.

1.3.6. ARC-OTA Configuration App Note (Step-by-Step Debug Commands)

1.3.6.1 ARC-OTA Software Release Manifest

Component		Version
Aerial CUDA-Accelerated RAN	Layer 1	24-1
	Data Lakes	24-1
	pyAerial	24-1
OAI[1]	gNB	2024.w21
	CN[2]	2024.w21
Sterling Skywave Service Management		v0.5
OpenRANGym (OSC RIC Release E)		v1.0
NVIDIA SDK Manager		v2.1
Foxconn O-RU n78 and CBRS		v3.1.15q.551v0706-oam

[1] For additional context, developers can also review the artifacts in the 2024.w21+ARC1.5 branch.
[2] The Grace Hopper platform requires OAI CN version 2024-June.

1.3.6.2 Setup Aerial CUDA-Accelerated RAN Layer 1

In the installation guide for cuBB, find the Aerial CUDA-Accelerated RAN Layer 1 section and follow the instructions.

1.3.6.3 Running the cuBB Docker Container

```
export GPU_FLAG="--gpus all"
export cuBB_SDK=/opt/nvidia/cuBB
#Name of your docker container
export AERIAL_CUBB_CONTAINER=cuBB_$USER
#Docker image downloaded from NGC
export AERIAL_CUBB_IMAGE=nvcr.io/qhrjhjrvlsbu/aerial-cuda-accelerated-ran:24-1-cubb

sudo usermod -aG docker $USER

docker run --detach --privileged \
   -it $GPU_FLAG --name $AERIAL_CUBB_CONTAINER \
   --hostname c_aerial_$USER \
   --add-host c_aerial_$USER:127.0.0.1 \
   --network host \
   --shm-size=4096m \
   -e cuBB_SDK=$cuBB_SDK \
   -w $cuBB_SDK \
   -v $(echo ~):$(echo ~) \
   -v /dev/hugepages:/dev/hugepages \
   -v /usr/src:/usr/src \
   -v /lib/modules:/lib/modules \
   -v ~/share:/opt/cuBB/share \
   --userns=host \
   --ipc=host \
   -v /var/log/aerial:/var/log/aerial \
   $AERIAL_CUBB_IMAGE

docker exec -it $AERIAL_CUBB_CONTAINER bash
```

For installation instructions, see the Aerial cuBB Installation Guide, in the link above.

Since the cuBB 23-4 release, the necessary testvectors for running OTA are already included. For running the Aerial E2E test with ru-emulator and test mac, follow the Aerial CUDA-Accelerated RAN documentation for generating the required testvectors.

1.3.6.4 Setup OAI gNB

1.3.6.4.1 Clone the gNB Source Code

Clone the OpenAirInterface5G repository.

```
git clone --branch  2024.w21+ARC1.5 https://gitlab.eurecom.fr/oai/openairinterface5g.
↪git ~/openairinterface5g

cd openairinterface5g
```

1.3.6.4.2 gNB Configuration File

Update the configuration of OAI L2. The configuration is located here.

The L1 configuration to use is included in the latest Aerial release image, and will differ depending on the gNB server you are using:

- **Gigiabyte**: Use the following file:

```
cuPHY-CP/cuphycontroller/config/cuphycontroller_P5G_FXN.yaml
```

- **SMC-GH**: Use the same configuration file as the Gigabyte server, ensuring that the cores are isolated as described in *Configure gNB Server - SMC Grace Hopper MGX*:

```
cuPHY-CP/cuphycontroller/config/cuphycontroller_P5G_FXN.yaml
```

- **Dell R750**: Use the following file:

```
cuPHY-CP/cuphycontroller/config/cuphycontroller_P5G_FXN_R750.yaml
```

1.3.6.5 Setup OAI CN5G

Do the iptables setup below after every system reboot, or make this setup permanent in your Ubuntu system configuration.

```
On CN5G server, configure it to allow the traffic coming in by
adding this rule to iptables:

# On CN5G, upon startup:

sudo sysctl net.ipv4.conf.all.forwarding=1
sudo iptables -P FORWARD ACCEPT
```

Install the core network by following the Gitlab steps for setting up OAI CN5G.

To run the correct configuration for ARC-OTA, replace section 2.2 and 2.3 OAI CN5G configuration files with the following:

```
# Get openairinterface5g source code
git clone --branch 2024.w21+ARC1.5 https://gitlab.eurecom.fr/oai/openairinterface5g.
↪git ~/openairinterface5g
cd ~/openairinterface5g
cp -rT ~/openairinterface5g/doc/tutorial_resources/oai-cn5g ~/oai-cn5g
```

The user configurable configuration files are:

- ~/oai-cn5g/database/oai_db.sql

1.3.6.6 Configuring OAI gNB and CN5G

For the purpose of understanding which address is what in the example configuration setting and commands below, we will assume the gNB and CN5G servers have these interface names and IP addresses.

CN5G Server

```
eno1: 10.31.66.x            = SSH management port for terminal
eno2: 169.254.200.6         = BH connection on SFP switch for gNB-CN5G traffic
```

gNB Server

```
eno1:     10.31.66.x        = SSH management port for terminal
ens6f0:   b8:ce:f6:4e:75:40 = FH MAC address
ens6f0.2: 169.254.1.105     = FH IP address
ens6f1:   169.254.200.5     = BH connection SFP switch for gNB-CN5G traffic
```

gNB to set static route

On the gNB server, add this static route for a path to the CN5G server. Apply this route after each server power-on.

```
Syntax:
sudo ip route add 192.168.70.128/26 via <CN5G IP> dev <gNB interface for CN5G>

Example:
sudo ip route add 192.168.70.128/26 via 169.254.200.6 dev ens6f1
```

gNB to set the CN5G server to uses for AMF

Edit the used gNB configuration file. The configuration file for Aerial can be found here:

```
~/openairinterface5g/targets/PROJECTS/GENERIC-NR-5GC/CONF/gnb-vnf.sa.band78.273prb.
↪aerial.conf
```

Update the Docker compose file to mount the updated file.

Below is an example with lab-specific network parameters. Your IP address and interface names may differ.

```
GNB_INTERFACE_NAME_FOR_NG_AMF = "ens6f1";       # gNB side interface name of the SFP
↪port toward CN (was eno1)
GNB_IPV4_ADDRESS_FOR_NG_AMF = "169.254.200.5";  # gNB side IP address of interface
↪above (was 172.21.16.130)
GNB_INTERFACE_NAME_FOR_NGU = "ens6f1";          # gNB side interface name of the SFP
↪port toward CN (was eno1)
GNB_IPV4_ADDRESS_FOR_NGU = "169.254.200.5";     # Same IP as GNB_IPV4_ADDRESS_FOR_NG_
↪AMF above (was 172.21.16.130)
```

1.3.6.7 Running CN5G

1.3.6.7.1 To start CN5G

```
docker-compose up -d
```

For SMC-GH, you will need to patch some of the CN components. Contact arc@nvidia.com for the patches and build instructions. Configuration files can be found here: https://gitlab.eurecom.fr/oai/openairinterface5g/-/tree/2024.w21+ARC1.5/ci-scripts/yaml_files/sa_gh_gnb

1.3.6.7.2 To Stop CN5G

```
docker-compose down
```

1.3.6.7.3 To monitor CN5G logs while running

```
docker logs oai-amf -f
```

1.3.6.7.4 To capture PCAPs

```
docker exec -it oai-amf /bin/bash
apt update && apt install tcpdump -y
tcpdump -i any -w /tmp/amf.pcap
```

Then copy the pcap out from the container.

```
docker cp oai-amf:/tmp/amf.pcap .
```

1.3.6.8 Example Screenshot of Starting CN5G

```
aerial@aerial-rf-r630:~/oai-cn5g$ docker compose up -d
[+] Building 0.0s (0/0)
[+] Running 11/11
✓ Network demo-oai-public-net  Created
 ↪                    0.1s
✓ Container oai-nrf            Started
 ↪                    0.7s
✓ Container mysql              Started
 ↪                    0.7s
✓ Container asterisk-ims       Started
 ↪                    0.7s
✓ Container oai-udr            Started
 ↪                    0.9s
✓ Container oai-udm            Started
 ↪                    1.2s
✓ Container oai-ausf           Started
 ↪                    1.5s
✓ Container oai-amf            Started
```

(continues on next page)

(continued from previous page)

```
 ↪                    1.7s
✓ Container oai-smf            Started
 ↪                    2.0s
✓ Container oai-spgwu-tiny     Started
 ↪                    2.3s
✓ Container oai-ext-dn         Started
 ↪                    2.6s
```

```
aerial@aerial-rf-r630:~/oai-cn5g$ docker ps
CONTAINER ID   IMAGE                                          COMMAND
 ↪CREATED              STATUS                          PORTS
 ↪NAMES
d5af4f51c393   oaisoftwarealliance/trf-gen-cn5g:latest     "/bin/bash -c ' ip r…"
 ↪About a minute ago   Up About a minute (healthy)
 ↪oai-ext-dn
a9b2d18c7f77   oaisoftwarealliance/oai-spgwu-tiny:v1.5.1   "python3 /openair-sp…"
 ↪About a minute ago   Up About a minute (healthy)   2152/udp, 8805/udp
 ↪oai-spgwu-tiny
b61c383f9e60   oaisoftwarealliance/oai-smf:v1.5.1          "python3 /openair-sm…"
 ↪About a minute ago   Up About a minute (healthy)   80/tcp, 8080/tcp, 8805/udp
 ↪oai-smf
3681b1048c53   oaisoftwarealliance/oai-amf:v1.5.1          "python3 /openair-am…"
 ↪About a minute ago   Up About a minute (healthy)   80/tcp, 9090/tcp, 38412/sctp
 ↪oai-amf
c602f7cb1c67   oaisoftwarealliance/oai-ausf:v1.5.1         "python3 /openair-au…"
 ↪About a minute ago   Up About a minute (healthy)   80/tcp
 ↪oai-ausf
752acae83ac0   oaisoftwarealliance/oai-udm:v1.5.1          "python3 /openair-ud…"
 ↪About a minute ago   Up About a minute (healthy)   80/tcp
 ↪oai-udm
4bf281d08229   oaisoftwarealliance/oai-udr:v1.5.1          "python3 /openair-ud…"
 ↪About a minute ago   Up About a minute (healthy)   80/tcp
 ↪oai-udr
33aa959be775   mysql:8.0                                   "docker-entrypoint.s…"
 ↪About a minute ago   Up About a minute (healthy)   3306/tcp, 33060/tcp
 ↪mysql
5d22e4745d00   asterisk-ims:latest                         "asterisk -fp"
 ↪About a minute ago   Up About a minute (healthy)
 ↪asterisk-ims
1a93b3ffe305   oaisoftwarealliance/oai-nrf:v1.5.1          "python3 /openair-nr…"
 ↪About a minute ago   Up About a minute (healthy)   80/tcp, 9090/tcp
 ↪oai-nrf
```

1.4. Tutorials

The best way to get started with ARC-OTA is with the tutorials. The tutorials below will walk you through setup and some example use cases.

Topic	Title
Overview	ARC-OTA Overview
Setup	ARC Development Environment Setup
Deployment	OpenAirInterface (OAI) 5G Core Network Deployment
	OpenAirInterface 5G Core Advance Deployment Using Docker-Compose

1.5. Release Notes

1.5.1. New Features and Fixes for ARC-OTA 1.5 (July, 2024)

- GraceHopper integration
  - OAI ARM CPU support
  - Hopper GPU cuBB support
- CBRS O-RU
- Multi-UE support
- Multi-UE CSI dataset blueprint (using Aerial Data Lake and PyAerial)
- Developer Extension Updates
  - OSC RIC tutorial update
- Developer Plugins
  - CBRS RU Interop
  - Open5GC
- New blueprints section added in the Product Brief

KPI Improvements

- MIMO layers
  - DL: 2 layers -> 4 layers
- Peak throughput
  - SMC-GH
    - DL: ~460Mbps -> ~1.03Gbps
    - UL: ~112Mbps -> ~125Mbps
  - Dell R750 / Gigabyte Edge E251-U70

- DL: ~460Mbps -> ~800Mbps

1.5.2. New Features and Fixes for ARC-OTA 1.3 (May, 2024)

- Full support for master gitlab repo gitlab.eurecom.fr/oai/ making it available for use, modification and distribution. Refer to the *Licensing* page for the OAI license.
  - **OAI_Aerial private branch is deprecated and will no longer be maintained.**
- Developer extension - Sterling k8 service management and monitoring (refer to this document for more details)

KPI Improvements

- Frame structure and slot format
  - DDDSU -> DDDSU + DDDDDDSUUU
- Bi-directional UDP Traffic
  - > 3.5 hours exercised -> > 4.0 hours exercised

1.5.3. New Features and Fixes for ARC-OTA 1.2 (March, 2024)

- Support for the Dell R750 platform to host gNB
- Support for converged cards A100X
- Continued support for Gigabyte and the discrete cards A100, CX6-DX

1.5.4. New Features and Fixes for A1.1 (January, 2024)

- Kernel cmdline configuration updated.
- Updates to the core assignment in the Aerial configuration.
- Updates to PTP and phc2sys core assignment.
- Changes to phc2sys cmdline.
- Changes to the L2 Docker run cmd to use all non isolated cores.
- Changes to the L2 configuration, max DL MCS defaults to 25.
- Removal of unnecessary ORU firmware installation step because of Foxconn firmware default updates.
- OAI_Aerial_v2.0 updated to OAI_Aerial_v2.2.1 throughout.

1.5.5. New Features and Fixes for A1.0 (December, 2023)

ARC-OTA A1.0 included the following OPENAIR-CN-5G network elements:

- Access and Mobility Management Function (AMF)
- Authentication Server Management Function (AUSF)
- Location Management Function (LMF)
- Network Exposure Function (NEF)
- Network Repository Function (NRF)
- Network Slicing Selection Function (NSSF)
- Network Data Analytics Function (NWDAF)
- Policy Control Function (PCF)
- Session Management Function (SMF)
- Unified Data Management (UDM)
- Unified Data Repository (UDR)
- User Plane Function (UPF) with 2 variants:
  - Simple Implementation (with a eBPF option) (UPF)
  - VPP-Based Implementation (UPF-VPP)
- Unstructured Data Storage Function (UDSF)

1.5.6. Known Issues and Limitations

256 QAM is not supported on ARC-OTA. You must disable 256 QAM support by issuing the following command at the gNB command license:

```
--gNBs.[0].force_256qam_off
```

1.5.7. Developer Documentation

The CUDA Accelerated RAN PHY developer guide can be found here.

1.6. Developer Extensions and Plugins

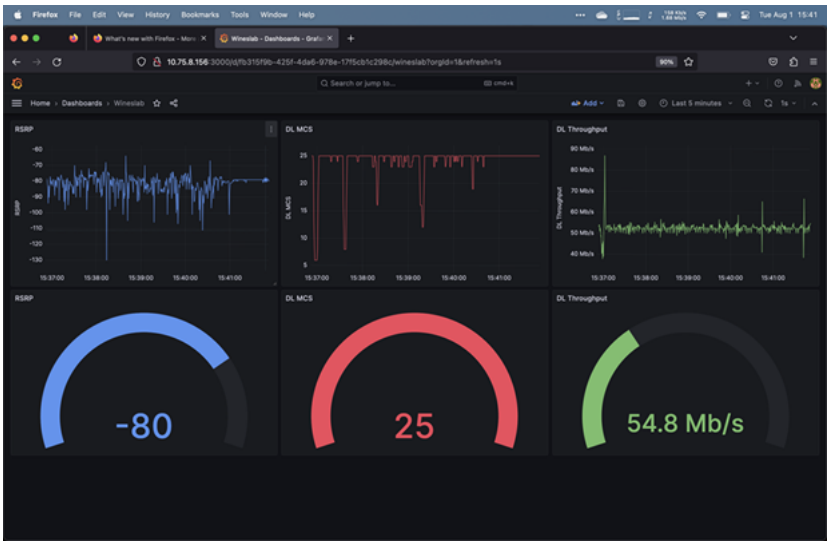
1.6.1. Extensions

1. *RIC Platform*
2. *Kubernetes Service Management*

1.6.1.1 RIC Platform

The Northeastern University (NEU) WIoT team supporting the Northeastern 8-node NVIDIA ARC-OTA deployment is working toward E2E integration of an O-RAN E2 interface with the ARC-OTA stack, leveraging the O-RAN OSC RIC, OpenRAN Gym framework.

This enables the streaming of relevant key performance metrics (KPMs) and the enforcement of control actions to reflect decisions taken by the xApps on a Near-Real-Time (Near-RT) RAN Intelligent Controller (RIC). An initial demonstration in July 2023 has showcased a data-collection xApp running on an O-RAN Software Community (OSC) RIC, deployed in a fully automated OpenShift cluster. The xApp is connected to an InfluxDB database to store the telemetry and present it on a Grafana dashboard. We are currently working toward enabling near-RT control on the same infrastructure.



1.6.1.1.1 Near Realtime RAN Intelligent Controller (RIC) - RIC Platform

- O-RAN OSC RIC
- OpenRAN Gym v1.0
- Reference to Distributed Applications (dApps) workflow for Real-Time Inference and Control in O-RAN

1.6.1.2 Kubernetes Service Management

The Sterling SkyWave Service Management extension is documented here.

1.6.2. Plugins

1.6.2.1 Open5Gs

Northeastern University successfully integrated and validated another 5G open-source core network, Open5Gs, in their experimental lab setup based on an OpenShift cluster. This core network, based on a microservice architecture, provides a optimized UPF, with support for high performance, easier management of user SIMs using a graphical interface, support for slicing and much more.

Open5Gs can be deployed in different ways, from bare-metal using routine package managers for Linux distributions, to Dockerized and virtualized approach, such as Helm Charts on Kubernetes and OpenShift. Once configured, it proved to be working fine with ARC, providing high and sustained performance: our experiments with OAI and 2 layer MIMO demonstrated stability and level of performance expected in the ARC-OTA release. This re-iterates the disaggregated O-RAN ecosystem with different vendors that can integrate seamlessly.

1.6.2.2 n48(CBRS) O-RU

The Rice University Wireless group has led the effort to enable interoperability of the NVIDIA ARC-OTA software with the Foxconn RPQN-4800E CBRS RU. Working with both teams at NVIDIA and Foxconn, the Rice team has successfully tested the ARC-OTA platform with the CBRS RU in the lab operating in a 100 MHz band (3.6-3.7 GHz) and has achieved a throughput of 250 Mbps in the downlink and 50 Mbps in the uplink. The conformance testing has been done with a Quectel RG520N UE module and the OnePlus Nord 5G handset in an indoor lab environment with more than one hour of stable connectivity.

This is an important step to help accelerate advanced 5G and 6G research and innovation with ARC-OTA experimental networks in the United States. The Citizen Broadband Radio Services (CBRS) spectrum band is an integral part of 5G deployment in the United states. This band, covering 3.55-3.7 GHz, has been opened by the FCC for shared access with three tiers of users: 1) Incumbent, 2) Priority Access License (PAL), and 3) General Authorized Access (GAA), where the last two tiers use a Spectrum Access System (SAS) database to coordinate their usage with the incumbents. While the incumbent users are the government bodies, the PAL access must be acquired through FCC spectrum auctions or through sublicensing in secondary markets which is costly and limited to 5G operators. The GAA license, however, can be used at times and locations where the incumbent and PAL users are inactive. Any GAA usage must be registered with the SAS database to coordinate with users of all three tiers. Given the shared access nature of CBRS and the GAA option, the CBRS band can be ideal for 5G research and development.

1.6.2.3 GPU MIG Partition

The Sterling SkyWave GPU MIG Partition plugin is documented here.

Application Note

While running Aerial on a GPU partition device, the `mps_sm_*` parameters in the cuphycontroller config YAML file need to be adjusted accordingly such that the `mps_sm_*` value is not over the available SMs of the selected MIG devices.

Please refer to the `mps_sm_*` configurations in `cuphycontroller_P5G_FXN.yaml` for the following cases:

- Running aerial with MIG disabled

```
mps_sm_pusch: 108

mps_sm_pucch: 16

mps_sm_prach: 16

mps_sm_pdsch: 82

mps_sm_pdcch: 28

mps_sm_pbch: 18

mps_sm_srs: 16
```

- Running aerial with MIG enabled on `mig-4g.48gb`

```
mps_sm_pusch: 42

mps_sm_pucch: 16

mps_sm_prach: 16

mps_sm_pdsch: 58

mps_sm_pdcch: 10

mps_sm_pbch: 8

mps_sm_srs: 8
```

- Running aerial with MIG enabled on `mig-3g.48gb`

```
mps_sm_pusch: 40

mps_sm_pucch: 16

mps_sm_prach: 16

mps_sm_pdsch: 52

mps_sm_pdcch: 10

mps_sm_pbch: 8
```

(continues on next page)

(continued from previous page)

```
mps_sm_srs:  8
```

1.7. On-Boarding Support

1.7.1. FAQs

Where is the HW BOM?

The complete qualified ARC-OTA BOM is located in the *Procure the Hardware* chapter of the Installation Guide.

Does the platform support MU-MIMO?

ARC-OTA does not offer MU-MIMO integrated interop, however the same platform is capable of adding software features for MU-MIMO.

Which frequency bands does the platform support?

ARC-OTA offers a tested reference for n48 and n78 sub-6 frequency bands.

Which RF parameters may require additional tuning for specific environments beyond the default config?

`pusch_TargetSNRx10`, `pucch_TargetSNRx10`, `ssPBCH_BlockPower`, `min_rxtxtime`, `TDD Patterns`, `RU Attenuations`

How can I apply for an experimental license in United States?

Review the application located at https://apps2.fcc.gov/ELSExperiments/pages/login.htm. If you have a program experimental license (https://apps.fcc.gov/oetcf/els/index.cfm), you can also use it for the Innovation Zone areas (Boston and the PAWR platforms) by submitting a request on that website.

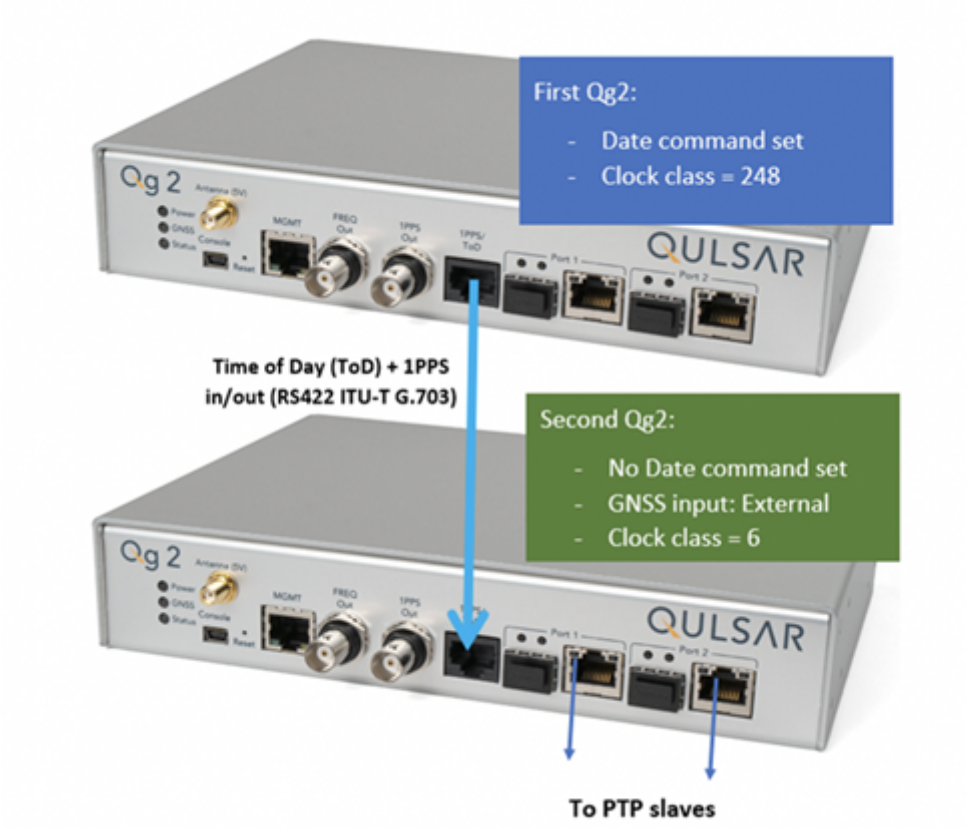
Where can I find utility RF tools and calculators?

See Tools for RF Wireless.

Is GPS needed?

Yes a GPS signal is necessary to drive precision timing for 5G networks. In case GPS signal is inaccessible, the date command can be used as a workaround. This command is useful for those deployments where there is no timing reference (like GNSS) but needs Qg 2 to act as a Grandmaster to propagate time and synchronization over PTP to slave units.

When using two GMs, you can manually set the date and time on the first one and connect it's 1PPS/ToD output to the 1PPS/ToD port (configured as input) of the second GM. The second GM then outputs PTP messages with a clock class=6.



How can I check OS and kernel version and configuration?

Use the following commands:

```
cat /proc/cmdline
 BOOT_IMAGE=/boot/vmlinuz-5.4.0-65-lowlatency root=UUID=d0bc583b-6922-4e70-af14-
↪92b624fe1bbe ro default_hugepagesz=1G hugepagesz=1G hugepages=16 tsc=reliable
↪clocksource=tsc intel_idle.max_cstate=0 mce=ignore_ce processor.max_cstate=0 intel_
↪pstate=disable audit=0 idle=poll isolcpus=2-21 nohz_full=2-21 rcu_nocbs=2-21 rcu_
↪nocb_poll nosoftlockup iommu=off intel_iommu=off irqaffinity=0-1,22-23
uname -a
 Linux dev01 5.4.0-65-lowlatency #73-Ubuntu SMP PREEMPT Mon Jan 18 18:17:38 UTC 2021
↪x86_64 x86_64 x86_64 GNU/Linux
```

1.7.2. Useful Shell Scripts

Use the following shell script to build cuBB (create the script in `/opt/nvidia/cuBB`):

```
#!/bin/bash
SCRIPT=$(readlink -f $0)
SCRIPT_DIR=$(dirname $SCRIPT)
echo running $SCRIPT
echo running $SCRIPT_DIR
export cuBB_SDK=${SCRIPT_DIR}
insModScript=${SCRIPT_DIR}/cuPHY-CP/external/gdrcopy/
echo $insModScript
```

(continues on next page)

(continued from previous page)

```
cd $insModScript && make && ./insmod.sh && cd -
export LD_LIBRARY_PATH=$cuBB_SDK/gpu-dpdk/build/install/lib/x86_64-linux-gnu
echo $LD_LIBRARY_PATH | sudo tee /etc/ld.so.conf.d/aerial-dpdk.conf
sudo ldconfig
echo "perhaps you want to: "
echo "mkdir build && cd build && cmake .. "
mkdir -p build && cd $_ && cmake .. && time chrt -r 1 taskset -c 2-20 make -j
```

Use the following shell script to start cuphycontroller (create the script in `/opt/nvidia/cuBB`):

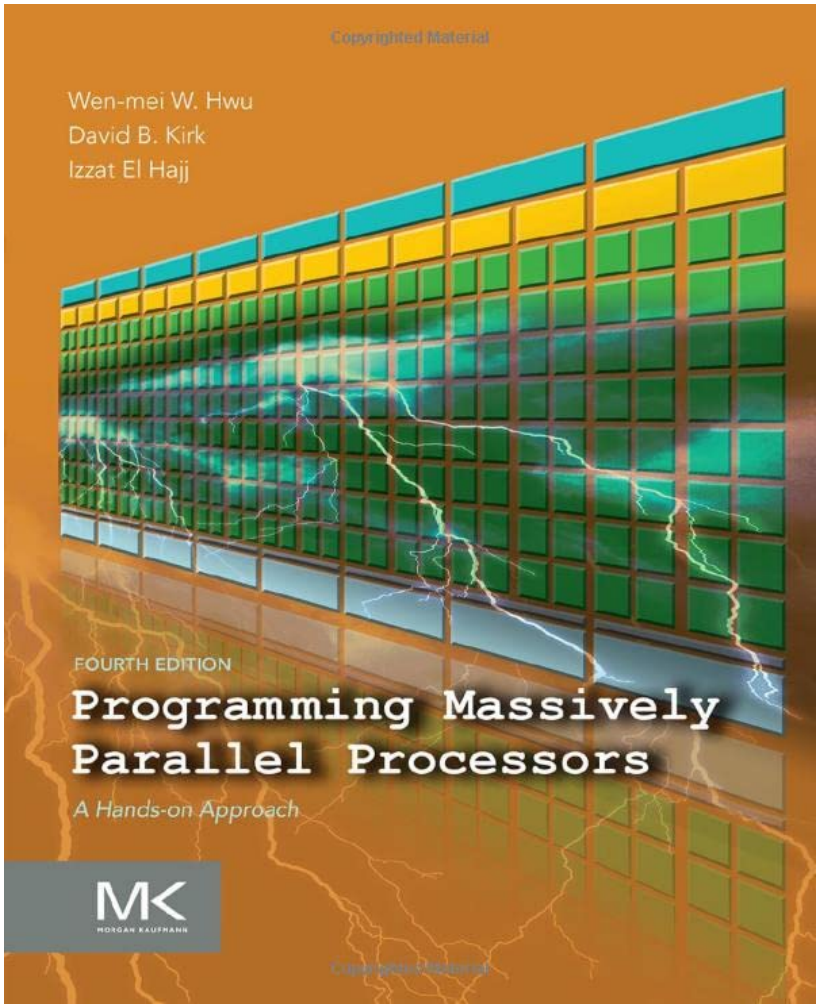
```
#!/bin/bash
sudo nvidia-smi -pm 1
sudo nvidia-smi -i 0 -lgc $(sudo nvidia-smi -i 0 --query-supported-clocks=graphics --
↪format=csv,noheader,nounits | sort -h | tail -n 1)

SCRIPT=$(readlink -f $0)
SCRIPT_DIR=$(dirname $SCRIPT)
echo running $SCRIPT
echo running $SCRIPT_DIR
export cuBB_SDK=${SCRIPT_DIR}
insModScript=${SCRIPT_DIR}/cuPHY-CP/external/gdrcopy/
echo $insModScript
cd $insModScript && ./insmod.sh && cd -
export LD_LIBRARY_PATH=$cuBB_SDK/gpu-dpdk/build/install/lib/x86_64-linux-gnu:$cuBB_
↪SDK/build/cuPHY-CP/cuphydriver/src
export LD_LIBRARY_PATH=${LD_LIBRARY_PATH}:/opt/mellanox/gdrcopy/src
export LD_LIBRARY_PATH=${LD_LIBRARY_PATH}:/opt/mellanox/dpdk/lib/x86_64-linux-gnu:/opt/
↪mellanox/doca/lib/x86_64-linux-gnu
echo $LD_LIBRARY_PATH | sed 's/:/\n/g' | sudo tee /etc/ld.so.conf.d/aerial-dpdk.conf
sudo ldconfig
export GDRCOPY_PATH_L=$cuBB_SDK/cuPHY-CP/external/gdrcopy/src
export CUDA_MPS_PIPE_DIRECTORY=/var
export CUDA_MPS_LOG_DIRECTORY=/var

sudo -E nvidia-cuda-mps-control -d
sudo -E echo start_server -uid 0 | sudo -E nvidia-cuda-mps-control
sleep 5
echo "perhaps you want to: "
echo "sudo -E LD_LIBRARY_PATH=$cuBB_SDK/gpu-dpdk/build/install/lib/x86_64-linux-gnu/ .
↪/build/cuPHY-CP/cuphycontroller/examples/cuphycontroller_scf P5G_SCF_FXN"
sudo -E ./build/cuPHY-CP/cuphycontroller/examples/cuphycontroller_scf P5G_SCF_FXN
sudo ./build/cuPHY-CP/gt_common_libs/nvIPC/tests/pcap/pcap_collect
```

1.7.3. Recommended Reading Material

Programming Massively Parallel Processors by David B. Kirk and Wen-mei W. Hwu



1.7.4. Hands-on CUDA-C

To learn CUDA and learn teach it to others, refer to the syllabus from the GPU Computing Course and NVIDIA/UIUC Accelerated Computing Teaching Kit, which outlines each module's organization in the downloaded Teaching Kit.

1.7.5. Additional Help

There are many ways to become part of the ARC-OTA community:

ARC-OTA early adopters community	Become part of a global network of developers using ARC-OTA.
arc@nvidia.com	Contact arc@nvidia.com for questions about ARC-OTA.
arc-developers@nvidia.com	Contact arc-developers@nvidia.com to join the ARC-OTA developers mailing list.
OAI mailing list	Reach out to this mailing list for any questions about OAI.
Sterling Skywave support	Refer to the Sterling website for any questions about the Kubernetes Service Management offering.
Arrow support	Contact Arrow Support for any questions about the *ORAN 7.2 Reference Hardware Blueprint*.

1.8. Featured Demos and Sessions

1.8.1. Radar Tech Talks

Date	Contact	Description	Link
March 2024	Michele Polese, Anupa Kelkar	Learn about the developer journey of Northeastern University and Michele Polese, an early ARC-OTA developer who went from an installed, configured, and operationalized 8 base station network to enabling multiple research streams, to then on-boarding and integrating a key network element, the RIC (RAN Intelligent Controller), as an ARC-OTA extension and an opportunity to on-board ML-based applications for the developer community. Join ARC-OTA developer and assistant professor Dr. Michele Polese from Northeastern University and NVIDIA Product Manager Anupa Kelkar as they share insights that showcase the potential of the platform capabilities, applications, and developer-extensions to jumpstart innovations in advancing wireless communications.	Northeastern Leads Open RAN Research
June 2024	Ravi P. Sinha Anupa Kelkar	Discover the O-RAN nGRG initiative aimed at advancing 6G research and the development of future AI-native network technologies. Learn about nGRG, its roadmap, objectives, and the operational dynamics of its various research streams, which include 6G use cases, architecture, AI/ML, security, and a research platform for PoC projects. The talk also explores the evolution and life cycle management of a genuinely cognitive network within the O-RAN network ecosystem, along with integration of AI in the next-generation automated programmable framework enhanced by Large Language Models (LLMs).	O-RAN Alliance's Next Generation Research Group Framework for 6G

1.8.2. ARC-OTA Developer Demos

Link	Description
Northeastern X5G	X5G is the first 8-node network deployment of the NVIDIA Aerial RAN CoLab Over-The-Air (ARC-OTA), with the Aerial SDK for the PHY layer, accelerated on Graphics Processing Unit (GPU), and through its integration with higher layers from the OpenAirInterface (OAI) open-source project through the Small Cell Forum Functional Application Platform Interface (FAPI).
Aerial Spot Demo at Hannover Messe	At Fraunhaufer, we have successfully integrated an Open RAN network based on NVIDIA ARC-OTA. At Y2024 Hannover Messe 6G-RIC booth in Germany our first demonstration showcased a virtual lab tour through a remote controlled robot UE connected over our deployment of the NVIDIA Aerial RAN CoLab Over-The-Air (ARC-OTA), with the Aerial SDK for the High PHY layer, accelerated on Graphics Processing Unit (GPU), and through its integration with higher layers from the OpenAirInterface (OAI).
AllBeSmart – Demo AI application with the NVIDIA Aerial Research CoLab OTA and OpenAirInterface (OAI)	In this over-the-air demonstration, a containerized AI video classification process runs in the same NVIDIA GPU that accelerates the 5G PHY protocol. Allbesmart Lda operates a reference implementation of the NVIDIA ARC platform, collaborating closely with the OAI and NVIDIA team to support early developers of the NVIDIA ARC platform for advanced 5G/6G research and experimentation.

1.8.3. ARC-OTA GTC Sessions

Link	Description
Advancing Connectivity: Democratizing 5G/6G Research With NVIDIA's Fully Open Programmable Network Stack	Today, we unveil a transformative solution poised to redefine the landscape of wireless communication. Our innovative platform is a beacon in the advancement of 5G+ and the forthcoming 6G networks, seamlessly blending digital and physical realities. This breakthrough goes beyond enhancing mobile broadband; it initiates an era of comprehensive digitalization, connecting humans, machines, and sensors like never before.
Programmable 5G and 6G networks	This panel will discuss the use cases and impacts of an AI/ML capable open and programmable next generation wireless network.nLast GTC Aerial RAN CoLab (Over the Air) was launched as the first fully programmable 5G and 6G advanced wireless full stack. The full stack has enabled developers and researchers to experiment - simulate, prototype, and benchmark innovations with a hardware-in-the-loop OTA NR compliant platform enabled by NVIDIA accelerated compute. The panel will discuss product roadmap, virtualization and migration to cloud services, advanced developer use cases
A Bridge to 6G - Aerial Research and Innovation Platform	NVIDIA and OAI experts provide an introduction to the first fully programmable Advanced 5G+ network as a sandbox – full-stack democratized platform for all researchers to simulate-prototype-benchmark optimizations, algorithms, and innovations rapidly in a deployed over-the-air NR standards compliant high performance operational network. This session will highlight platform vision, early adopter use cases, highlight C/C++ network programmability, provide OAI ISV gNB and CN overview and deep dive into specific ML examples that can jumpstart innovations.

1.9. Developer Use Cases

We love to see how ARC-OTA is being used by developers, researchers, and the industry. Send an email to arc@nvidia.com with your project description and links to the project and code repository (e.g. GitHub).

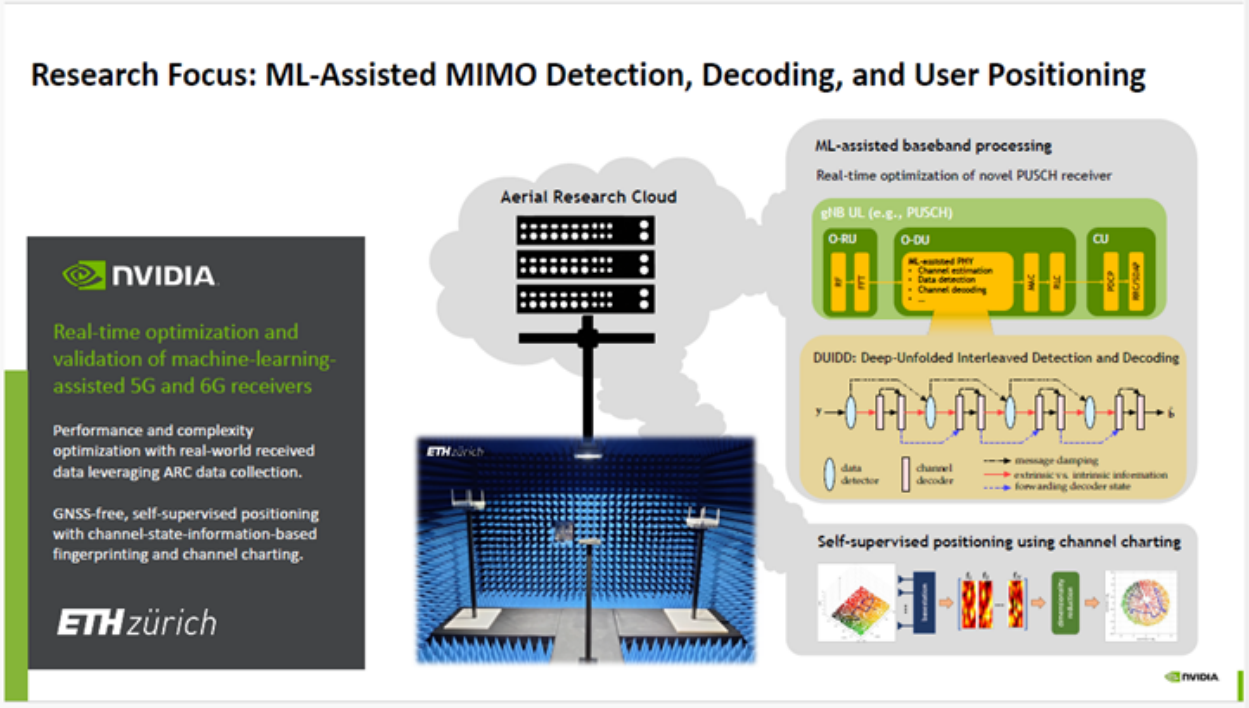
The following are significant developer use cases:

1.9.1. ETH Zurich

Integrated Information Processing Group

Real-world implementations are vital for research on future wireless communication systems. By adopting the ARC-OTA platform, the Integrated Information Processing group at ETH Zurich is developing and evaluating novel baseband processing algorithms. Real-world over-the-air experiments with this full-stack 5G system reveal the practical benefits of machine learning (ML)-assisted physical layer processing and the efficacy of channel-state-information-based positioning techniques that utilize ML.

Real-World 5G System Blog



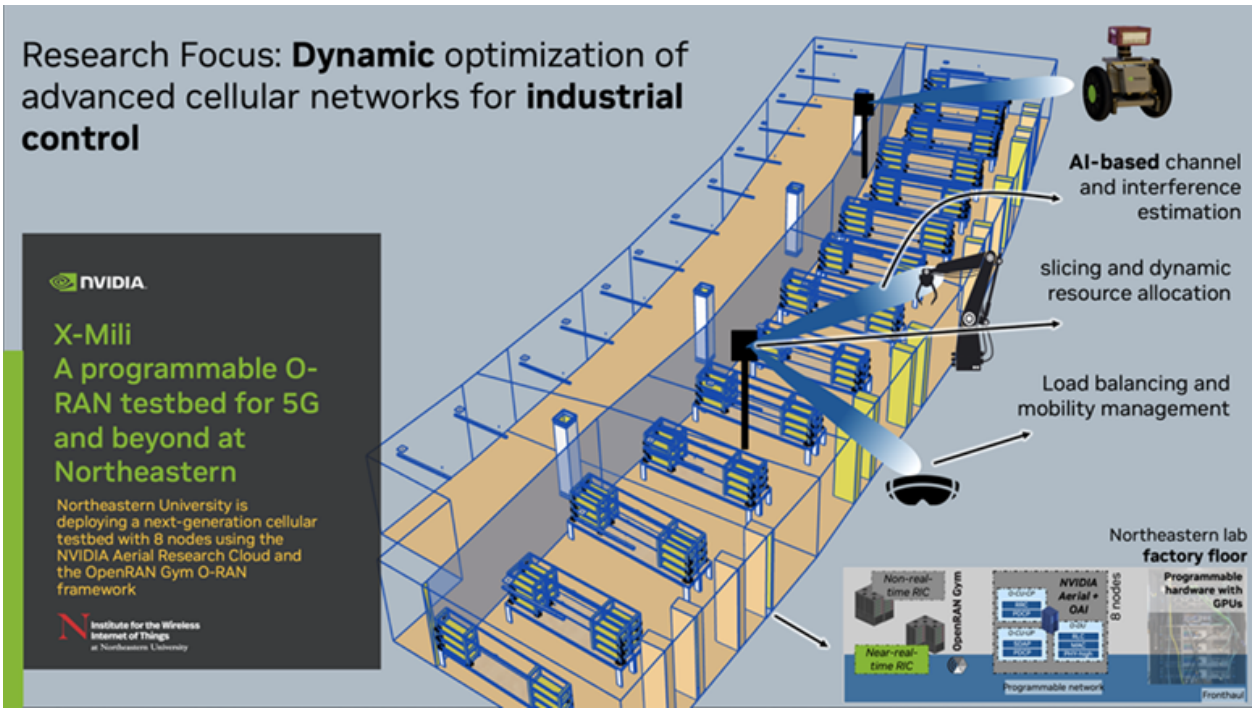
1.9.2. HHI Fraunhofer

6G-RIC Is Significantly Advancing Its Open Test Environment

Open source, end-to-end deployments are key, offering 6G-RIC researchers and associated startups a highly accessible and versatile platform for experimentation. This encourages innovation and facilitates the testing of emerging technologies, protocols, and applications The integration of an Open RAN network, based on open-source technologies and NVIDIA Aerial RAN CoLab Over-the-Air , marks a significant milestone for our project. The GPU-centric design is ideal for integrating AI/ML and expediting the creation of demonstrators, which once required significant development time.

1.9.3. Northeastern University

In this blog post, Northeastern University outlines how it used ARC-OTA to develop a production-ready 5G network that is automated through machine learning (ML). Visit https://wiot.northeastern.edu/ for information about the Northeastern Institute for the Wireless Internet of Things program.



1.9.4. Open Air Alliance

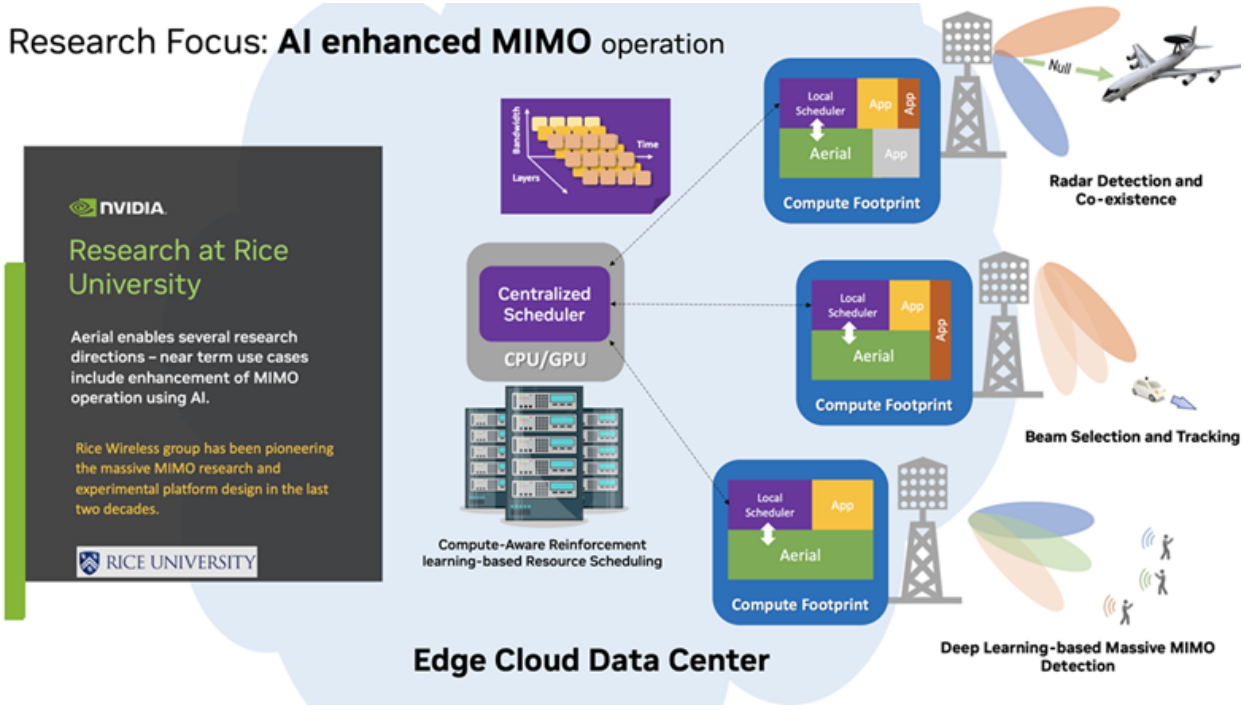
https://openairinterface.org/

Open ecosystem for creating and deploying advanced 5G networks, while enabling 6G research–software-defined, disaggregated radio access networks. Working with the OAI, ARC-OTA will significantly reduce time to innovation and create new efficiencies across next-generation network development: Demonstration of NVIDIA Aerial SDK and OAI 5G vRAN and CN

NVIDIA ARC-OTA and OAI Demo Blog

1.9.5. Rice University

In this blog post, Rice University outlines how it's testing enhanced 5G communications with GPU-based vRANs. Visit https://wireless.rice.edu/ for information about the Rice wirless program.



1.10. News and Noteworthy Publications

Author(s)	Title
O-RAN Spring 2024 Plugfest at Northeastern OTIC	Automating and Testing End-to-End O-RAN Systems
O-RAN Global Spring PlugFest 2024 at EURECOM	O-RAN Global Plugfest Spring 2024
Northeastern University	X5G: An Open, Programmable, Multi-vendor, End-to-end, Private 5G O-RAN Testbed with NVIDIA ARC and OpenAirInterface (Journal extension under review - focus RIC)
Sterling	Sterling introduces SkyWave
Anupa Kelkar	A brief description of ARC-OTA in two slides
NTIA, Office of Public Affairs	Northeastern and Rice university NTIA NOFO 1 win Biden-Harris Administration Award for Nearly $80M for Wireless Innovation
Eidgenössische Technische Hochschule Zürich	Real-World 5G System
Northeastern University	Northeastern University launches Fully Automated and Virtualized O-RAN Private 5G Network with AI Automation
TMCnet News	2023 Open Ran Product of the Year Award Winners
Davide Villa, Imran Khan, Florian Kaltenberger, Nicholas Hedberg, Ruben Soares da Silva, Anupa Kelkar, Chris Dick, Stefano Basagni, Josep M. Jornet, Tommaso Melodia, Michele Polese, Dimitrios Koutsonikolas	An Open, Programmable, Multi-vendor 5G O-RAN Testbed with NVIDIA ARC-OTA and OpenAirInterface
Anupa Kelkar, Chris Dick	Introducing NVIDIA Aerial Research Cloud for Innovations in 5G and 6G
Florian Kaltenberger, Irfan Ghauri, Chris Dick, Anupa Kelkar, Lopamudra Kundu	Demonstration of NVIDIA Aerial SDK and OAI 5G vRAN and CN Virtual Exhibition
OpenAirInterface	OpenAirInterface Demonstrates 5G Virtual RAN with NVIDIA Aerial SDK
Jeffrey Andrews	Site Specific Deep Learning for the 6G Air Interface
Rahman Doost-Mohammady, Santiago Segarra, Ashutosh Sabharwal	Rice University Blog
Chris Dick	IEEE Keynote: The NVIDIA Roadmap tp AI-Infused 6G

1.11. Background

1.11.1. NVIDIA in Telecommunications

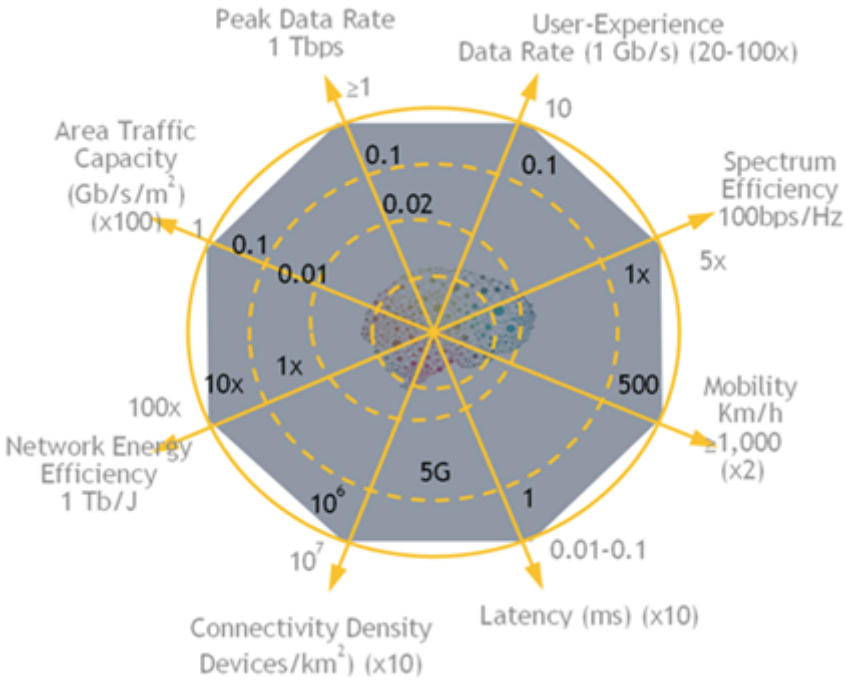
NVIDIA is the global leader in GPU accelerated computing and is enabling a fully cloud-native virtual 5G and 6G RAN solution to support a wide range of next-generation edge AI and RAN services using COTS servers. Solutions include Aerial CUDA-Accelerated RAN and Sionna.

1.11.2. 6G RAN Development

With the deployment of 5G systems in full swing, the research focus on 6G wireless communications systems has begun. Keeping up with the tradition of a new generation of cellular systems once every ten years, there is an expectation that 6G systems will be standardized and ready for deployment starting around 2030[https://www.itu.int/en/ITU-T/focusgroups/net2030/Documents/White_Paper.pdf]. Because it typically takes ten years for a NG (Next Generation) wireless technology to see commercial daylight, the time to begin research for 6G is now.

It is crucial to ensure availability of a comprehensive programmable end-to-end (E2E) research and innovation platform to develop technologies for future communication systems. While it is possible that some of the requirements for NG wireless can be met by incorporating new advancements within the advanced 5G framework, it is already clear that meeting the goals of 6G will require some fundamental shifts in system architecture, waveform design, protocols, interference management, and channel modelling.

Besides enhanced mobile broadband for consumers with very high data rates of 1Tbps, 5G+ is widely expected to enable the Fifth Industrial Revolution through the digitalization and connectivity of all things (humans, machines, sensors). Digital twins of objects created in edge clouds will form the essential foundation of the future digital world. The realization of a comprehensive and true digital rendering of the physical world at every spatial and time instant will be required at extreme low latency. Sensors will accurately map every instant and integrate into the digital and virtual worlds, to enable new Artificial Intelligence (AI) enabled capabilities. Augmented reality user interfaces will enable efficient and intuitive human control of all these worlds, whether physical, or virtual. Simply put, 6G is widely expected to be smarter, faster, and more efficient than 5G. Specifically, the following image outlines the key 6G trending KPIs:



1.11.3. Emerging Use Cases

The following major new use case themes are emerging for the new communications framework:

Use Case	Reference
End devices extending from being single entities to a collection of multiple local entities acting in unison to create the new man-machine interface.	5G NR Rel 18 - Gateway UE function for Mission Critical Communication
Distributed compute among multiple local devices and the cloud	5G NR Rel 18 - Ad hoc Group Communication support in Mission Critical Services
Knowledge systems that store, process, and convert data into actionable knowledge through AI systems in network functions as well as operations.	5G NR Rel 18 - AI/ML model transfer in 5GS
Precision sensing and actuation to control the physical world	5G NR Rel 18 - Application layer support for Factories of the Future (FF)
Network digital twin	ITU-T Y.3090 Digital Twin Architecture and Requirements

The following are top requirements from 6G researchers:

- Massive MIMO and mmWave support
- Easy access to high-performance, cloud-native compute resources
- Access to large-scale datasets, digital twins, and other simulation environments

- Distributed AI, online training, federated learning, network prediction, and native ML simulation
- Standards-compliant OTA platform to maximize real-time, closed-loop experimentation
- Ease of programmability and customization of the network (CN + RAN + UE)

1.11.4. About the OpenAirInterface

The OpenAirInterface Software Alliance (OSA) is a nonprofit organization founded in 2014 by EURECOM, a research institute based in the South of France. The Alliance manages and promotes the OpenAirInterface (OAI) open-source software that offers 4G and 5G and Core Network stacks as well as orchestration and management and control software. OAI implements 3GPP and the O-RAN specifications.

The OAI software development is organized into three project groups: Radio Access Network (RAN), Core Network (CN), and MOSAIC5G (M5G). Another project called CI/CD allows OAI to control the quality of all software produced within the Alliance. Each project group is composed of an engineering team following and achieving the objectives defined in its roadmap. The OSA stands out thanks to its large international community of contributors and users. The OAI software is used by many organizations around the globe for research and testing purposes as well as for building blocks of systems for various 4G/5G use cases, a growing number of them industrial.

For end-to-end deployments and control, OAI enables 5G deployment including the 5G gNB, Core Network, and RAN control capability thanks to O-RAN specified E2 and RIC software.

1.12. Licensing

- Aerial CUDA-Accelerated RAN
- OAI
- *Sterling*

1.12.1. Sterling License

Copyright (C) 2024 Sterling Computers Corporation

This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, version 2 of the License.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program; if not, write to the Free Software Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston, MA 02110-1301, USA.

Sterling Computers Corporation

- Attn: Contract Administrator
- PO Box 1995

- North Sioux City, SD 57049
- contracts@sterling.com

Copyright