OAM Configuration

Table of contents

List of Figures

Startup Configuration (cuphycontroller)

The application binary name for the combined cuPHY-CP + cuPHY is cuphycontroller. When cuphycontroller starts, it reads static configuration from configuration YAML files. This section describes the fields in the YAML files.

l2adapter_filename

This field contains the filename of the YAML-format config file for l2 adapter configuration.

low_priority_core

CPU core shared by all low-priority threads, isolated CPU core is preferred. Can be non-isolated CPU core but make sure no other heavy load task on it.

nic_tput_alert_threshold_mbps

This parameter is used to monitor NIC throughput. The units are in Mbps, that is, 85000 = 85 Gbps. This value is almost the max throughput that can be achieved with accurate send scheduling for a 100 Gbps link. A gRPC client(reference: $cuBB_SDK/cuPHY-CP/cuphyoam/examples/test_grpc_push_notification_client.cpp) needs to be implemented to receive the alert.

cuphydriver_config

This container holds configuration for cuphydriver.

standalone

0 - run cuphydriver integrated with other cuPHY-CP components

1 - run cuphydriver in standalone mode (no l2adapter, etc)

validation

Enables additional validation checks at run-time.

0 - Disabled

1 - Enabled

num_slots

Number of lots to run in cuphydriver standalone test.

log_level

cuPHYDriver log level: DBG, INFO, ERROR.

profiler_sec

Number of seconds to run the CUDA profiling tool.

dpdk_thread

Sets the CPU core used by the primary DPDK thread. It does not have to be an isolated core. And the DPDK thread itself is defaulted to ‘SCHED_FIFO+priority 95’.

dpdk_verbose_logs

Enable maximum log level in DPDK.

0 - Disable

1 - Enable

accu_tx_sched_res_ns

Sets the accuracy of the accurate transmit scheduling, in units of nanoseconds.

accu_tx_sched_disable

Disable accurate TX scheduling.

0 - packets are sent according to the TX timestamp

1 - packets are sent whenever it is convenient

fh_stats_dump_cpu_core

Sets the CPU core used by the FH stats logging thread. It does not have to be an isolated core. And currently the default FH stats polling interval is 500ms.

pdump_client_thread

CPU core to use for pdump client. Set to -1 to disable fronthaul RX traffic PCAP capture.

See:

1. https://doc.dpdk.org/guides/howto/packet_capture_framework.html
2. aerial-fh README.md

mps_sm_pusch

Number of SMs for PUSCH channel.

mps_sm_pucch

Number of SMs for PUCCH channel.

mps_sm_pusch

Number of SMs for PUSCH channel.

mps_sm_prach

Number of SMs for PRACH channel.

mps_sm_ul_order

Number of SMs for UL order kernel.

mps_sm_pdsch

Number of SMs for PDSCH channel.

mps_sm_pdcch

Number of SMs for PDCCH channel.

mps_sm_pbch

Number of SMs for PBCH channel.

mps_sm_srs

Number of SMs for SRS channel.

mps_sm_gpu_comms

Number of SMs for GPU comms.

nics

Container for NIC configuration parameters.

nic

PCIe bus address of the NIC port.

mtu

Maximum transmission size, in bytes, supported by the Fronthaul U-plane and C-plane.

cpu_mbufs

Number of preallocated DPDK memory buffers (mbufs) used for Ethernet packets.

uplane_tx_handles

The number of pre-allocated transmit handles that link the U-plane prepare() and transmit() functions.

txq_count

NIC transmit queue count.

Must be large enough to handle all cells attached to this NIC port.

Each cell uses one TXQ for C-plane and *txq_count_uplane* TXQs for U-plane.

rxq_count

Receive queue count.

This value must be large enough to handle all cell attached to this NIC port.

Each cell uses one RXQ to receive all uplink traffic.

txq_size

Number of packets that can fit in each transmit queue.

rxq_size

Number of packets that can be buffered in each receive queue.

gpu

CUDA device to receive uplink packets from this NIC port.

gpus

List of GPU device IDs. To use gpudirect, the GPU must be on the same PCIe root complex as the NIC. To maximize performance, the GPU should be on the same PCIe switch as the NIC. Only the first entry in the list is used.

workers_ul

List of pinned CPU cores used for uplink worker threads.

workers_dl

List of pinned CPU cores used for downlink worker threads.

debug_worker

For performance debug purpose, this is set to a free core to work with the enable_*_tracing logs.

prometheus_thread

Pinned CPU core for updating NIC metrics once per second.

start_section_id_srs

ORAN CUS start section ID for the SRS channel.

start_section_id_prach

ORAN CUS start section ID for the PRACH channel.

enable_ul_cuphy_graphs

Enable UL processing with CUDA graphs.

enable_dl_cuphy_graphs

Enable DL processing with CUDA graphs.

section_3_time_offset

Time offset, in units of nanoseconds, for the PRACH channel.

ul_order_timeout_cpu_ns

Timeout, in units of nanoseconds, for the uplink order kernel to receive any U-plane packets for this slot.

ul_order_timeout_gpu_ns

Timeout, in units of nanoseconds, for the order kernel to complete execution on the GPU.

pusch_sinr

Enable pusch sinr calculation (0 by default).

pusch_rssi

Enable PUSCH RSSI calculation (0 by default).

pusch_tdi

Enable PUSCH TDI processing (0 by default).

pusch_cfo

Enable PUSCH CFO calculations (0 by default).

pusch_dftsofdm

DFT-s-OFDM enable/disable flag: 0 - disable, 1 - enable.

pusch_to

It is only used for timing offset reporting to L2. If the timing offset estimate is not used by L2, it can be disabled.

pusch_select_eqcoeffalgo

Algorithm selector for PUSCH noise interference estimation and channel equalization. The following values are supported: 0: Regularized zero-forcing (RZF) 1: Diagonal MMSE regularization 2: Minimum Mean Square Error - Interference Rejection Combining (MMSE-IRC) 3: MMSE-IRC with RBLW covariance shrinkage 4: MMSE-IRC with OAS covariance shrinkage.

pusch_select_chestalgo

Channel estimation algorithm selection: 0 - legacy MMSE, 1 - multi-stage MMSE with delay estimation.

pusch_tbsizecheck

Tb size verification enable/disable flag: 0 - disable, 1 - enable.

pusch_deviceGraphLaunchEn

Static flag to allow device graph launch in PUSCH.

pusch_waitTimeOutPreEarlyHarqUs

Timeout threshold in microseconds for receiving OFDM symbols for PUSCH early-HARQ processing.

pusch_waitTimeOutPostEarlyHarqUs

Timeout threshold in microseconds for receiving OFDM symbols for PUSCH non-early-HARQ processing (essentially all the PUSCH symbols).

puxch_polarDcdrListSz

List size used in List Decoding of Polar codes.

enable_cpu_task_tracing

The flag is used to trace and instrument DL/UL CPU tasks running on existing cuphydriver cores.

enable_prepare_tracing

It’s for tracing the U-plane packet preperation kernel durations and end times and need the debug worker to be enabled.

enable_dl_cqe_tracing

Enables tracing of DL CQEs (debug feature to check for DL U-plane packets’ timing at the NIC).

ul_rx_pkt_tracing_level

This YAML param can be set to 3 different values: 0 (default, recommended) : Only keeps count of the early/ontime/late packet counters per slot as seen by the DU (Reorder kernel) for the Uplink U-plane packets. 1 : Also Captures and logs earliest/latest packet timestamp per symbol per slot as seen by the DU. 2 : Also Captures and logs timestamp of each packet received per symbol per slot as seen by the DU.

split_ul_cuda_streams

Keep default of 0. This allows back to back UL slots to overlap their processing. Keep disabled to maintain performance of first UL slot in every group of 2.

aggr_obj_non_avail_th

Keep the default value at 5. This param sets the threshold for successive non-availability of L1 objects (can be interpreted as L1 handler necessary to schedule PHY compute tasks to the GPU). Unavailability could imply the execution timeline falling behind the expected L1 timeline budget.

dl_wait_th_ns

This parameter is used for error handling in the event of GPU failure. You must keep the defaults.

sendCPlane_timing_error_th_ns

Keep the default value at 50000 (50 us). The threshold is used as a check for the proximity of the current time during C-plane task's execution to the actual scheduled C-plane packet's transmission time. Meeting the threshold check would result in C-plane packet transmission being dropped for the slot.

pusch_forcedNumCsi2Bits

Debug feaure if > 0, overrides the number of PUSCH CSI-P2 bits for all CSI-P2 UCIs with the non-zero value provided. Recommend setting it to 0.

mMIMO_enable

Keep at default of 0. This flag is reserved for future capability.

enable_srs

Enable/disable SRS

enable_csip2_v3

Enable/disable the the support of CSI part2 defined by FAPI 10.03 Table 3-77

ue_mode

Flag for spectral effeciency feature. Must be enabled on the RU side YAML to emulate UE operation.

cplane_disable

Disable C-plane for all cells.

0 - Enable C-plane 1 - Disable C-plane

cells

List of containers of cell parameters.

name

Name of the cell

cell_id

ID of the cell.

src_mac_addr

Source MAC address for U-plane and C-plane packets. Set to 00:00:00:00:00:00 to use the MAC address of the NIC port in use.

dst_mac_addr

Destination MAC address for U-plane and C-plane packets.

nic

gNB NIC port to which the cell is attached.

Must match the ‘nic’ key value in one of the elements of in the ‘nics’ list.

vlan

VLAN ID used for C-plane and U-plane packets.

pcp

QoS priority codepoint used for C-plane and U-plane Ethernet packets.

txq_count_uplane

Number of transmit queues used for U-plane.

eAxC_id_ssb_pbch

List of eAxC IDs to use for SSB/PBCH.

eAxC_id_pdcch

List of eAxC IDs to use for PDCCH.

eAxC_id_pdsch

List of eAxC IDs to use for PDSCH.

eAxC_id_srs

List of eAxC IDs to use for CSI RS.

eAxC_id_pusch

List of eAxC IDs to use for PUSCH.

eAxC_id_pucch

List of eAxC IDs to use for PUCCH.

eAxC_id_srs

List of eAxC IDs to use for SRS.

eAxC_id_prach

List of eAxC IDs to use for PRACH.

dl_iq_data_fmt:comp_meth

DL U-plane compression method: 0: Fixed point 1: BFP

dl_iq_data_fmt:bit_width

Number of bits used for each RE on DL U-plane channels. Fixed point supported value: 16 BFP supported value: 9, 14, 16

ul_iq_data_fmt:comp_meth

UL U-plane compression method: 0: Fixed point 1: BFP

ul_iq_data_fmt:bit_width

Number of bits used per RE on uplink U-plane channels. Fixed point supported value: 16 BFP supported value: 9, 14, 16

fs_offset_dl

Downlink U-plane scaling per ORAN CUS 6.1.3.

exponent_dl

Downlink U-plane scaling per ORAN CUS 6.1.3.

ref_dl

Downlink U-plane scaling per ORAN CUS 6.1.3.

fs_offset_ul

Uplink U-plane scaling per ORAN CUS 6.1.3.

exponent_ul

Uplink U-plane scaling per ORAN CUS 6.1.3.

max_amp_ul

Maximum full scale amplitude used in uplink U-plane scaling per ORAN CUS 6.1.3.

mu

3GPP subcarrier bandwidth index ‘mu’.

0 - 15 kHz 1 - 30 kHz 2 - 60 kHz 3 - 120 kHz 4 - 240 kHz

T1a_max_up_ns

Scheduled timing advance before time-zero for downlink U-plane egress from DU, per ORAN CUS.

T1a_max_cp_ul_ns

Scheduled timing advance before time-zero for uplink C-plane egress from DU, per ORAN CUS.

Ta4_min_ns

Start of DU reception window after time-zero, per ORAN CUS.

Ta4_max_ns

End of DU reception window after time-zero, per ORAN CUS.

Tcp_adv_dl_ns

Downlink C-plane timing advance ahead of U-plane, in units of nanoseconds, per ORAN CUS.

ul_u_plane_tx_offset_ns

Flag for spectral effeciency feature. Must be set on the RU side YAML to offset UL transmission start from T0.

pusch_prb_stride

Memory stride, in units of PRBs, for the PUSCH channel. Affects GPU memory layout.

prach_prb_stride

Memory stride, in units of PRBs, for the PRACH channel. Affects GPU memory layout.

srs_prb_stride

Memory stride, in units of PRBs, for the SRS. Affects GPU memory layout.

pusch_ldpc_max_num_itr_algo_type

0 - Fixed LDPC iteration count

1 - MCS based LDPC iteration count

Recommend setting pusch_ldpc_max_num_itr_algo_type:1

pusch_fixed_max_num_ldpc_itrs

Unused currently, reserved to replace pusch_ldpc_n_iterations.

pusch_ldpc_n_iterations

Iteration count is set to pusch_ldpc_n_iterations, when the fixed LDPC iteration count option is selected (pusch_ldpc_max_num_itr_algo_type:0). Because the default value of pusch_ldpc_max_num_itr_algo_type is 1 (iteration count optimized based on MCS), pusch_ldpc_n_iterations is unused.

pusch_ldpc_algo_index

Algorithm index for LDPC decoder: 0 - automatic choice.

pusch_ldpc_flags

pusch_ldpc_flags are flags that configure the LDPC decoder. pusch_ldpc_flags:2 selects an LDPC decoder that optimizes for throughput i..e processes more than one codeword (for example, 2) instead of latency.

pusch_ldpc_use_half

Indication of input data type of LDPC decoder:

0 - single precision, 1 - half precision

pusch_nMaxPrb

This is for memory allocation of max PRB range of peak cells compared to average cells.

ul_gain_calibration

UL Configured Gain used to convert dBFS to dBm. Default value, if unspecified: 48.68

lower_guard_bw

Lower Guard Bandwidth expressed in kHZ. Used for deriving freqOffset for each Rach Occasion. Default is 845.

tv_pusch

HDF5 file containing static configuration (for example, filter coefficients) for the PUSCH channel.

tv_prach

HDF5 file containing static configuration (for example, filter coefficients) for the PRACH channel.

pusch_ldpc_n_iterations

PUSCH LDPC channel coding iteration count.

pusch_ldpc_early_termination

PUSCH LDPC channel coding early termination.

0 - Disable 1 - Enable

workers_sched_priority

cuPHYDriver worker threads scheduling priority.

dpdk_file_prefix

Shared data file prefix to use for the underlying DPDK process.

wfreq

Filename containing the coefficients for channel estimation filters, in HDF5 (.h5) format.

cell_group

Enable cuPHY cell groups.

0 - disable 1 - enable

cell_group_num

Number of cells to be configured in L1 for the test.

enable_h2d_copy_thread

Enable/disable offloading of h2d copy in L2A to a seperate copy thread.

h2d_copy_thread_cpu_affinity

CPU core on which the h2d copy thread in L2A should run. Applicable only if enable_h2d_copy_thread is 1.

h2d_copy_thread_sched_priority

h2d copy thread priority in L2A. Applicable only if enable_h2d_copy_thread is 1.

fix_beta_dl

Fix the beta_dl for local test with RU Emulator so that the output values are a bytematch to the TV.

Startup Configuration (l2_adapter_config)

msg_type

Defines the L2/L1 interface API. Supported options are:

- scf_fapi_gnb - Use the small cell forum API.

phy_class

Same as msg_type.

tick_generator_mode

The SLOT.incication interval generator mode:

0 - poll + sleep. During each tick the threads sleep some time to release the CPU core to avoid hanging the system, then they poll the system time. 1 - sleep. Sleep to absolute timestamp, no polling. 2 - timer_fd. Start a timer and call epoll_wait() on the timer_fd.

allowed_fapi_latency

Allowed maximum latency of SLOT FAPI messages, which send from L2 to L1, otherwise the message is ignored and dropped.

Unit: slot. Default is 0, it means L2 message should be received in current slot.

allowed_tick_error

Allowed tick interval error.

Unit: us

Tick interval error is printed in statistic style. If observed tick error > allowed, the log is printed as Error level.

timer_thread_config

Configuration for the timer thread.

name

Name of thread.

cpu_affinity

Id of pinned CPU core used for timer thread.

sched_priority

Scheduling priority of timer thread.

message_thread_config

Configuration container for the L2/L1 message processing thread.

name

Name of thread.

cpu_affinity

Id of pinned CPU core used for timer thread.

sched_priority

Scheduling priority of message thread.

ptp

ptp configs for GPS_ALPHA, GPS_BETA.

gps_alpha

GPS Alpha value for ORAN WG4 CUS section 9.7.2. Default value = 0, if undefined.

gps_beta

GPS Beta value for ORAN WG4 CUS section 9.7.2. Default value = 0, if undefined.

mu_highest

Highest supported mu, used for scheduling TTI tick rate.

slot_advance

Timing advance ahead of time-zero, in units of slots, for L1 to notify L2 of a slot request.

enableTickDynamicSfnSlot

Enable dynamic slot/sfn.

staticPucchSlotNum

Debugging param for testing against RU Emulator to send set static PUCCH slot number.

staticPuschSlotNum

Debugging param for testing against RU Emulator to send set static PUSCH slot number.

staticPdschSlotNum

Debugging param for testing against RU Emulator to send set static PDSCH slot number.

staticPdcchSlotNum

Debugging param for testing against RU Emulator to send set static PDCCH slot number.

staticCsiRsSlotNum

Debugging param for testing against RU Emulator to send set static CSI-RS slot number.

staticSsbPcid

Debugging param for testing against RU Emulator to send set static SSB phycellId.

staticSsbSFN

Debugging param for testing against RU Emulator to send set static SSB SFN.

instances

Container for cell instances.

name

Name of the instance.

layerMap

List of mappings to layers.

nvipc_config_file

Config dedicated YAML file for nvipc. Example: nvipc_multi_instances.yaml

transport

Configuration container for L2/L1 message transport parameters.

type

Transport type. One of shm, dpdk, or udp.

udp_config

Configuration container for the udp transport type.

local_port

UDP port used by L1.

remote_port

UDP port used by L2.

shm_config

Configuration container for the shared memory transport type.

primary

Indicates process is primary for shared memory access.

prefix

Prefix used in creating shared memory filename.

cuda_device_id

Set this parameter to a valid GPU device ID to enable CPU data memory pool allocation in host pinned memory. Set to -1 to disable this feature.

ring_len

Length, in bytes, of the ring used for shared memory transport.

mempool_size

Configuration container for the memory pools used in shared memory transport.

cpu_msg

Configuration container for the shared memory transport for CPU messages (that is, L2/L1 FAPI messages).

buf_size

Buffer size in bytes.

pool_len

Pool length in buffers.

cpu_data

Configuration container for the shared memory transport for CPU data elements (that is, downlink and uplink transport blocks).

buf_size

Buffer size in bytes.

pool_len

Pool length in buffers.

cuda_data

Configuration container for the shared memory transport for GPU data elements.

buf_size

Buffer size in bytes.

pool_len

Pool length in buffers.

dpdk_config

Configurations for the DPDK over NIC transport type.

primary

Indicates process is primary for shared memory access.

prefix

The name used in creating shared memory files and searching DPDK memory pools.

local_nic_pci

The NIC address or name used in IPC.

peer_nic_mac

The peer NIC MAC address, only need to be set in secondary process (L2/MAC).

cuda_device_id

Set this parameter to a valid GPU device ID to enable CPU data memory pool allocation in host pinned memory. Set to -1 to disable this feature.

need_eal_init

Whether nvipc needs to call rte_eal_init() to initiate the DPDK context. 1 - initiate by nvipc; 0 - initiate by other module in the same process.

lcore_id

The logic core number for nvipc_nic_poll thread.

mempool_size

Configuration container for the memory pools used in shared memory. transport.

cpu_msg

Configuration container for the shared memory transport for CPU messages (that is, L2/L1 FAPI messages).

buf_size

Buffer size in bytes.

pool_len

Pool length in buffers.

cpu_data

Configuration container for the shared memory transport for CPU data elements (that is, downlink and uplink transport blocks).

buf_size

Buffer size in bytes.

pool_len

Pool length in buffers.

cuda_data

Configuration container for the shared memory transport for GPU data elements.

buf_size

Buffer size in bytes.

pool_len

Pool length in buffers.

app_config

Configurations for all transport types, mostly used for debug.

grpc_forward

Whether to enable forwarding nvipc messages and how many messages to be forwarded automatically from initialization. Here count = 0 means forwarding every message forever.

0: disabled; 1: enabled but doesn't start forwarding at initial; -1: enabled and start forwarding at initial with count = 0; Other positive number: enabled and start forwarding at initial with count = grpc_forward.

debug_timing

For debug only.

Whether to record timestamp of allocating, sending, receiving, releasing of all nvipc messages.

pcap_enable

For debug only.

Whether to capture nvipc messages to pcap file.

pcap_cpu_core

CPU core of background pcap log save thread.

pcap_cache_size_bits

Size of /dev/shm/${prefix}_pcap. If set to 29, size is 2^29 = 512MB.

pcap_file_size_bits

Max size of /dev/shm/${prefix}_pcap. If set to 31, size is 2^31 = 2GB.

pcap_max_data_size

Max DL/UL FAPI data size to capture reduce pcap size.

aerial_metrics_backend_address

Aerial Prometheus metrics backend address.

pucch_dtx_thresholds

Array of scale factors for DTX Thresholds of each PUCCH format.

Default value, if not present, is 1.0, which means the thresholds are not scaled.

For PUCCH format 0 and 1, -100.0 is replaced with 1.0.

Example:

pucch_dtx_thresholds: [-100.0, -100.0, 1.0, 1.0, -100.0]

pusch_dtx_thresholds

Scale factor for DTX Thresholds of UCI on PUSCH.

Default value, if not present, is 1.0, which means the threshold is not scaled.

Example:

pusch_dtx_thresholds: 1.0

enable_precoding

Enable/Disable Precoding PDUs to be parsed in L2Adapter.

Default value is 0 enable_precoding: 0/1

prepone_h2d_copy

Enable/Disable preponing of H2D copy in L2Adapter.

Default value is 1 prepone_h2d_copy: 0/1

enable_beam_forming

Enables/Disables BeamIds to parsed in L2Adapter.

Default value : 0 enable_beam_forming: 1

dl_tb_loc

Transport block location in inside nvipc buffer.

Default value is 1 dl_tb_loc: 0 # TB is located in inline with nvipc’s msg buffer. dl_tb_loc: 1 # TB is located in nvipc’s CPU data buffer. dl_tb_loc: 2 # TB is located in nvipc’s GPU buffer.

staticSsbSlotNum

Override the incoming slot number with the YAML configured SlotNumber for SS/PBCH.

Example

staticSsbSlotNum:10

Startup Configuration (ru-emulator)

The application binary name for the combined O-RU + UE emulator is ru-emulator. When ru-emulator starts, it reads static configuration from a configuration YAML file. This section describes the fields in the YAML file.

core_list

List of CPU cores that RU Emulator could use.

nic_interface

PCIe address of NIC to use that is, b5:00.1.

peerethaddr

MAC address of cuPHYController port.

nvlog_name

The nvlog instance name for ru-emulator. Detailed nvlog configurations are in nvlog_config.yaml.

cell_configs

Cell configs agreed upon with DU.

name

Cell string name (largely unused).

eth

Cell MAC address.

dl_iq_data_fmt:comp_meth

DL U-plane compression method: 0: Fixed point 1: BFP

dl_iq_data_fmt:bit_width

Number of bits used for each RE on DL U-plane channels. Fixed point supported value: 16 BFP supported value: 9, 14, 16

ul_iq_data_fmt:comp_meth

UL U-plane compression method: 0: Fixed point 1: BFP

ul_iq_data_fmt:bit_width

Number of bits used for each RE on UL U-plane channels. Fixed point supported value: 16 BFP supported value: 9, 14, 16

flow_list

eAxC list

eAxC_prach_list

eAxC prach list

vlan

vlan to use for RX and TX

nic

Index of the nic to use in the nics list.

tti

Slot indication inverval.

validate_dl_timing

Validate DL timing (need to be PTP synchronized).

timing_histogram

generate histogram

timing_histogram_bin_size

histogram bin size

oran_timing_info

dl_c_plane_timing_delay

t1a_max_up from ORAN

dl_c_plane_window_size

DL C Plane RX ontime window size.

ul_c_plane_timing_delay

T1a_max_cp_ul from ORAN.

ul_c_plane_window_size

UL C Plane RX ontime window size.

dl_u_plane_timing_delay

T2a_max_up from ORAN.

dl_u_plane_window_size

DL U Plane RX ontime window size.

ul_u_plane_tx_offset

Ta4_min_up from ORAN.

Run-time Configuration/Status

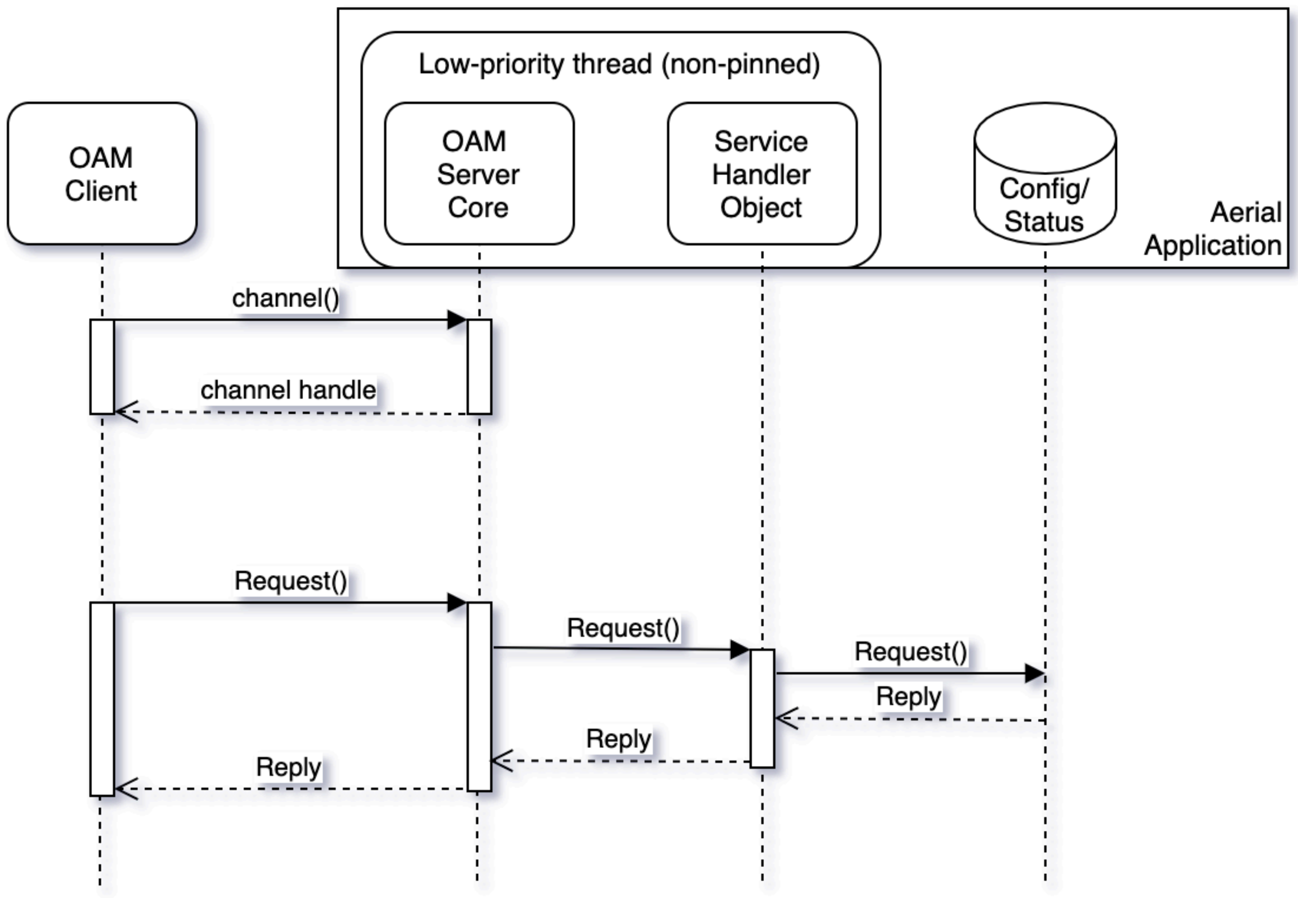
During run-time, Aerial components can be re-configured or queried for status through gRPC remote procedure calls (RPCs). The RPCs are defined in "protocol buffers" syntax, allowing support for clients written in any of the languages supported by gRPC and protocol buffers.

More information about gRPC may be found at: https://grpc.io/docs/what-is-grpc/core-concepts/

More information about protocol buffers may be found at: https://developers.google.com/protocol-buffers

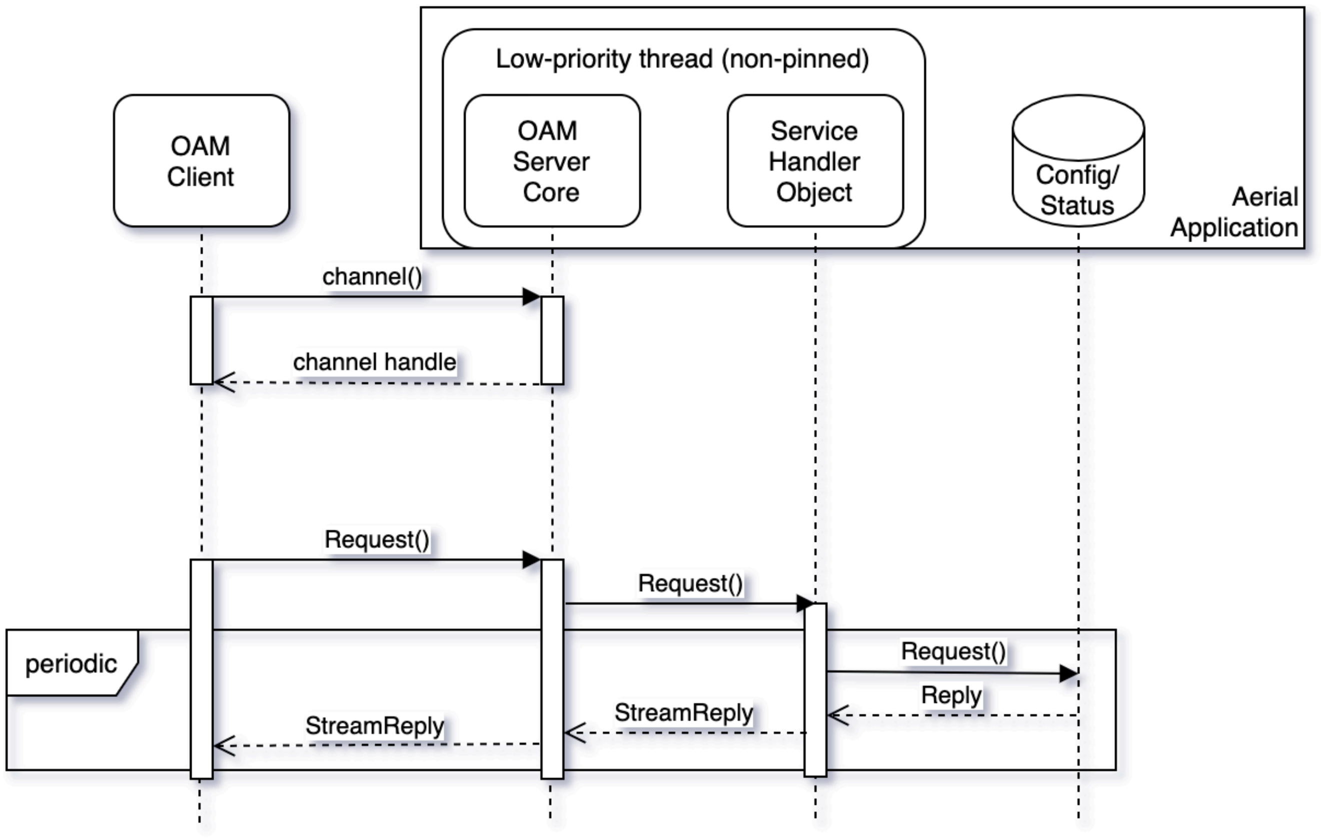
Simple Request/Reply Flow

Aerial applications support a request/reply flow using the gRPC framework with protobufs messages. At run-time, certain configuration items may be updated and certain status information may be queried. An external OAM client interfaces with the Aerial application acting as the gRPC server.



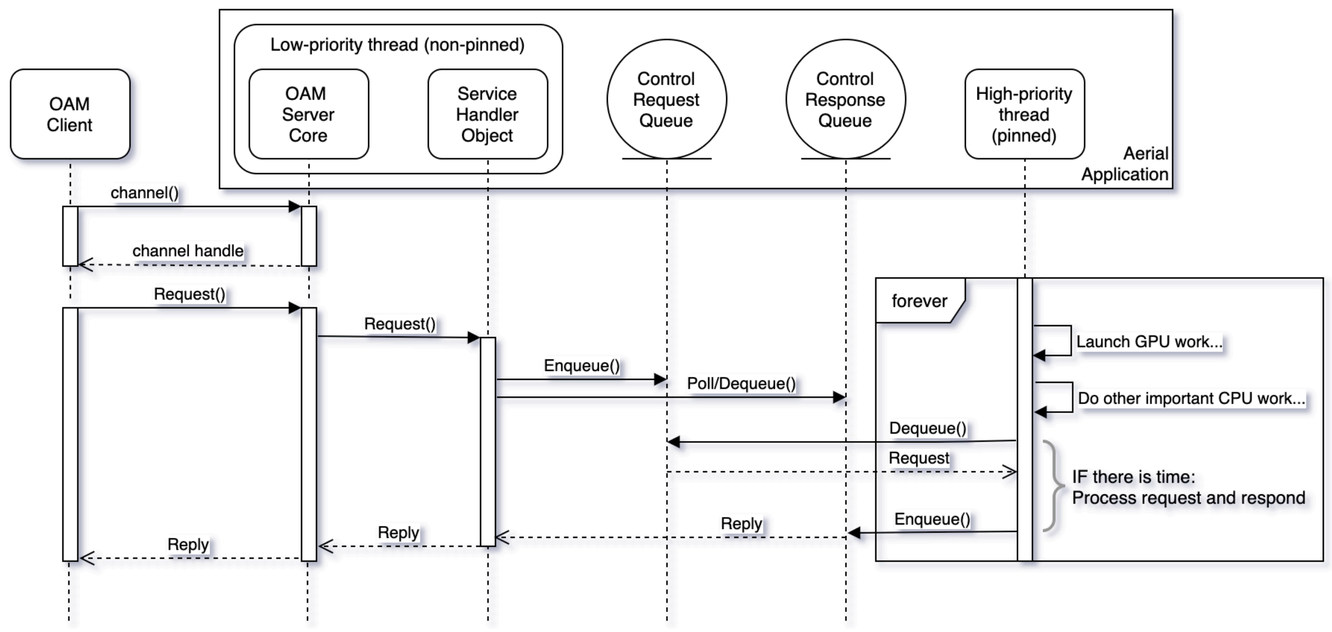
Streaming Request/Replies

Aerial applications support the gRPC streaming feature for sending periodic status between client and server.



Asynchronous Interthread Communication

Certain request/reply scenarios require interaction with the high-priority CPU-pinned threads orchestrating GPU work. These interactions occur through Aerial-internal asynchronous queues, and requests are processed on a best effort basis that prioritizes the orchestration of GPU kernel launches and other L1 tasks.



Aerial Common Service Definition

```
/\* \* Copyright (c) 2021, NVIDIA CORPORATION. All rights reserved. \* \* NVIDIA
CORPORATION and its licensors retain all intellectual property \* and proprietary
rights in and to this software, related documentation \* and any modifications
thereto. Any use, reproduction, disclosure or \* distribution of this software and
related documentation without an express \* license agreement from NVIDIA
CORPORATION is strictly prohibited. \*/ syntax = "proto3"; package aerial; service
Common { rpc GetSFN (GenericRequest) returns (SFNReply) {} rpc GetCpuUtilization
(GenericRequest) returns (CpuUtilizationReply) {} rpc SetPuschH5DumpNextCrc
(GenericRequest) returns (DummyReply) {} rpc GetFAPIStream (FAPIStreamRequest)
returns (stream FAPIStreamReply) {} } message GenericRequest { string name = 1; }
message SFNReply { int32 sfn = 1; int32 slot = 2; } message DummyReply { }
message CpuUtilizationPerCore { int32 core_id = 1; int32 utilization_x1000 = 2; }
message CpuUtilizationReply { repeated CpuUtilizationPerCore core = 1; } message
FAPIStreamRequest { int32 client_id = 1; int32 total_msgs_requested = 2; } message
FAPIStreamReply { int32 client_id = 1; bytes msg_buf = 2; bytes data_buf = 3; }
```

rpc GetCpuUtilization

The GetCpuUtilization RPC returns a variable-length array of CPU utilization per-high-priority-core.

CPU utilization is available through the Prometheus node exporter, however the design approach used by Aerial high-priority threads results in a false 100% CPU core utilization per thread. This RPC allows retrieval of the actual CPU utilization of high-priority threads. High-priority threads are pinned to specific CPU cores.

rpc GetFAPIStream

This RPC requests snooping of one or more (up to infinite number) of SCF FAPI messages. The snooped messages are delivered from the Aerial gRPC server to a third party client. See cuPHY-CP/cuphyoam/examples/aerial_get_l2msgs.py for an example client.

rpc TerminateCuphycontroller

This RPC message terminates cuPHYController with immediate effect.

rpc CellParamUpdateRequest

This RPC message updates cell configuration without stopping the cell. Message specification:

```
message CellParamUpdateRequest { int32 cell_id = 1; string dst_mac_addr = 2; int32 vlan_tci = 3; }
```

dst_mac_addr must be in 'XX:XX:XX:XX:XX:XX' format.

vlan_tci must include the 16-bit TCI value of 802.1Q tag.

List of Parameters Supported by Dynamic OAM via gRPC and CONFIG.request (M-plane)

The Configuration unit is accross all cells/per cell config. The Cell outage is either in-service or out-of-service.

Parameter name	Configuration unit	Cell outage	OAM command	Note
ru_type	per cell config	out-of-service	cd $cuBB_SDK/build/cuPHY-CP/cuphyoam && python3	$RU_TYPE : 1 for FXN_RU, 2 for FJT_RU, 3 for OTHER_RU(including ru_emulator)

			$cuBB_SDK/cuPHY-CP/cuphyoam/examples/aerial_cell_multi_attrs_update.py –server_ip $SERVER_IP –cell_id $CELL_ID –ru_type $RU_TYPE	
nic	per cell config	out-of-service	cd $cuBB_SDK/build/cuPHY-CP/cuphyoam && python3 $cuBB_SDK/cuPHY-CP/cuphyoam/examples/aerial_cell_multi_attrs_update.py –server_ip $SERVER_IP –cell_id $CELL_ID –nic $NIC	nic PCIe address. It has to be one of the nic ports configured in cuphycontroller YAML file
dst_mac_addr	per cell config	out-of-service	cd $cuBB_SDK/build/cuPHY-CP/cuphyoam && python3 $cuBB_SDK/cuPHY-CP/cuphyoam/examples/aerial_cell_multi_attrs_update.py –server_ip $SERVER_IP –cell_id $CELL_ID –dst_mac_addr $DST_MAC_ADDR –vlan_id $VLAN_ID –pcp $PCP	dst_mac_addr, vlan id and pcp have to be updated together
vlan_id	per cell config	out-of-service	cd $cuBB_SDK/build/cuPHY-CP/cuphyoam && python3 $cuBB_SDK/cuPHY-CP/cuphyoam/examples/aerial_cell_multi_attrs_update.py –server_ip $SERVER_IP –cell_id	dst_mac_addr, vlan id and pcp have to be updated together

			$CELL_ID –dst_mac_addr $DST_MAC_ADDR –vlan_id $VLAN_ID –pcp $PCP	
pcp	per cell config	out-of-service	cd $cuBB_SDK/build/cuPHY-CP/cuphyoam && python3 $cuBB_SDK/cuPHY-CP/cuphyoam/examples/aerial_cell_multi_attrs_update.py –server_ip $SERVER_IP –cell_id $CELL_ID –dst_mac_addr $DST_MAC_ADDR –vlan_id $VLAN_ID –pcp $PCP	dst_mac_addr, vlan id and pcp have to be updated together
dl_iq_data_fmt	per cell config	out-of-service	cd $cuBB_SDK/build/cuPHY-CP/cuphyoam && python3 $cuBB_SDK/cuPHY-CP/cuphyoam/examples/aerial_cell_multi_attrs_update.py –server_ip $SERVER_IP –cell_id $CELL_ID –dl_comp_meth $COMP_METH –dl_bit_width $BIT_WIDTH	
ul_iq_data_fmt	per cell config	out-of-service	cd $cuBB_SDK/build/cuPHY-CP/cuphyoam && python3 $cuBB_SDK/cuPHY-CP/cuphyoam/examples/aerial_cell_multi_attrs_update.py –server_ip $SERVER_IP –cell_id $CELL_ID –ul_comp_meth	

			$COMP_METH –ul_bit_width $BIT_WIDTH	
exponent_dl	per cell config	out-of-service	cd $cuBB_SDK/build/cuPHY-CP/cuphyoam && python3 $cuBB_SDK/cuPHY-CP/cuphyoam/examples/aerial_cell_multi_attrs_update.py –server_ip $SERVER_IP –cell_id $CELL_ID –exponent_dl $EXPONENT_DL	
exponent_ul	per cell config	out-of-service	cd $cuBB_SDK/build/cuPHY-CP/cuphyoam && python3 $cuBB_SDK/cuPHY-CP/cuphyoam/examples/aerial_cell_multi_attrs_update.py –server_ip $SERVER_IP –cell_id $CELL_ID –exponent_ul $EXPONENT_UL	
prusch_prb_stride	per cell config	out-of-service	cd $cuBB_SDK/build/cuPHY-CP/cuphyoam && python3 $cuBB_SDK/cuPHY-CP/cuphyoam/examples/aerial_cell_multi_attrs_update.py –server_ip $SERVER_IP –cell_id $CELL_ID –pusch_prb_stride $PUSCH_PRB_STRIDE	
prach_prb_stride	per cell config	out-of-service	cd $cuBB_SDK/build/cuPHY-CP/cuphyoam &&	

			python3 $cuBB_SDK/cuPHY-CP/cuphyoam/examples/ aerial_cell_multi_attrs_up date.py –server_ip $SERVER_IP –cell_id $CELL_ID –prach_prb_stride $PRACH_PRB_STRIDE	
max_amp _ul	per cell config	out-of-service	cd $cuBB_SDK/build/cuPHY-CP/cuphyoam && python3 $cuBB_SDK/cuPHY-CP/cuphyoam/examples/ aerial_cell_multi_attrs_up date.py –server_ip $SERVER_IP –cell_id $CELL_ID –max_amp_ul $MAX_AMP_UL	
section_3 _time_offs et	per cell config	out-of-service	cd $cuBB_SDK/build/cuPHY-CP/cuphyoam && python3 $cuBB_SDK/cuPHY-CP/cuphyoam/examples/ aerial_cell_multi_attrs_up date.py –server_ip $SERVER_IP –cell_id $CELL_ID –section_3_time_offset $SECTION_3_TIME_OFFSE T	
fh_distanc e_range	per cell config	out-of-service	cd $cuBB_SDK/build/cuPHY-CP/cuphyoam && python3 $cuBB_SDK/cuPHY-CP/cuphyoam/examples/	$FH_DISTANCE_RANGE : 0 for 0~30km, 1 for 20~50km Suppose the following are the default configs in the cuhycontroller YAML

			aerial_cell_multi_attrs_update.py –server_ip $SERVER_IP –cell_id $CELL_ID –fh_distance_range $FH_DISTANCE_RANGE	config file that correspond to FH_DISTANCE_RANGE option 0 (0~30km). t1a_max_up_ns : d1 t1a_max_cp_ul_ns : d2 ta4_min_ns : d3 ta4_max_ns : d4 Updating FH_DISTANCE_RANGE option to 1 (20~50km), adjusts the following values: t1a_max_up_ns : d1+$FH_EXTENSION_DELAY_ADJUSTMENT t1a_max_cp_ul_ns : d2+$FH_EXTENSION_DELAY_ADJUSTMENT ta4_min_ns : d3+$FH_EXTENSION_DELAY_ADJUSTMENT ta4_max_ns : d4+$FH_EXTENSION_DELAY_ADJUSTMENT $FH_EXTENSION_DELAY_ADJUSTMENT is 100us for now and can be tuned in source file: ${cuBB_SDK}/cuPHY-CP/cuphydriver/include/constant.hpp#L207 static constexpr uint32_t FH_EXTENSION_DELAY_ADJUSTMENT = 100000;//100us
ul_gain_calibration	per cell config	in-service	cd $cuBB_SDK/build/cuPHY-CP/cuphyoam && python3 $cuBB_SDK/cuPHY-	

			CP/cuphyoam/examples/aerial_cell_multi_attrs_update.py –server_ip $SERVER_IP –cell_id $CELL_ID –ul_gain_calibration $UL_GAIN_CALIBRATION	
lower_guard_bw	per cell config	out-of-service	cd $cuBB_SDK/build/cuPHY-CP/cuphyoam && python3 $cuBB_SDK/cuPHY-CP/cuphyoam/examples/aerial_cell_multi_attrs_update.py –server_ip $SERVER_IP –cell_id $CELL_ID –lower_guard_bw $LOWER_GUARD_BW	
ref_dl	per cell config	out-of-service	cd $cuBB_SDK/build/cuPHY-CP/cuphyoam && python3 $cuBB_SDK/cuPHY-CP/cuphyoam/examples/aerial_cell_multi_attrs_update.py –server_ip $SERVER_IP –cell_id $CELL_ID –ref_dl $REF_DL	
attenuation_db	per cell config	in-service	cd $cuBB_SDK/build/cuPHY-CP/cuphyoam && python3 $cuBB_SDK/cuPHY-CP/cuphyoam/examples/aerial_cell_param_attn_update.py $CELL_ID $ATTENUATION_DB	

gps_alpha	accross all cells	out-of-service	cd $cuBB_SDK/build/cuPHY-CP/cuphyoam && python3 $cuBB_SDK/cuPHY-CP/cuphyoam/examples/aerial_cell_multi_attrs_update.py –server_ip $SERVER_IP –gps_alpha $GPS_ALPHA	All cells have to be in idle state before configuring this param
gps_beta	accross all cells	out-of-service	cd $cuBB_SDK/build/cuPHY-CP/cuphyoam && python3 $cuBB_SDK/cuPHY-CP/cuphyoam/examples/aerial_cell_multi_attrs_update.py –server_ip $SERVER_IP –gps_beta $GPS_BETA	All cells have to be in idle state before configuring this param
prachRootSequenceIndex	per cell config	out-of-service	Via FAPI CONFIG.request. See section Dynamic PRACH Configuration and Init Sequence Test	
prachZeroCorrConf	per cell config	out-of-service	Via FAPI CONFIG.request. See section Dynamic PRACH Configuration and Init Sequence Test	
numPrachFdOccasions	per cell config	out-of-service	Via FAPI CONFIG.request. See section Dynamic PRACH Configuration and Init Sequence Test	
restrictedSetConfig	per cell config	out-of-service	Via FAPI CONFIG.request. See section Dynamic PRACH Configuration and Init Sequence Test	

prachConfigIndex	per cell config	out-of-service	Via FAPI CONFIG.request. See section Dynamic PRACH Configuration and Init Sequence Test	
K1	per cell config	out-of-service	Via FAPI CONFIG.request. See section Dynamic PRACH Configuration and Init Sequence Test	

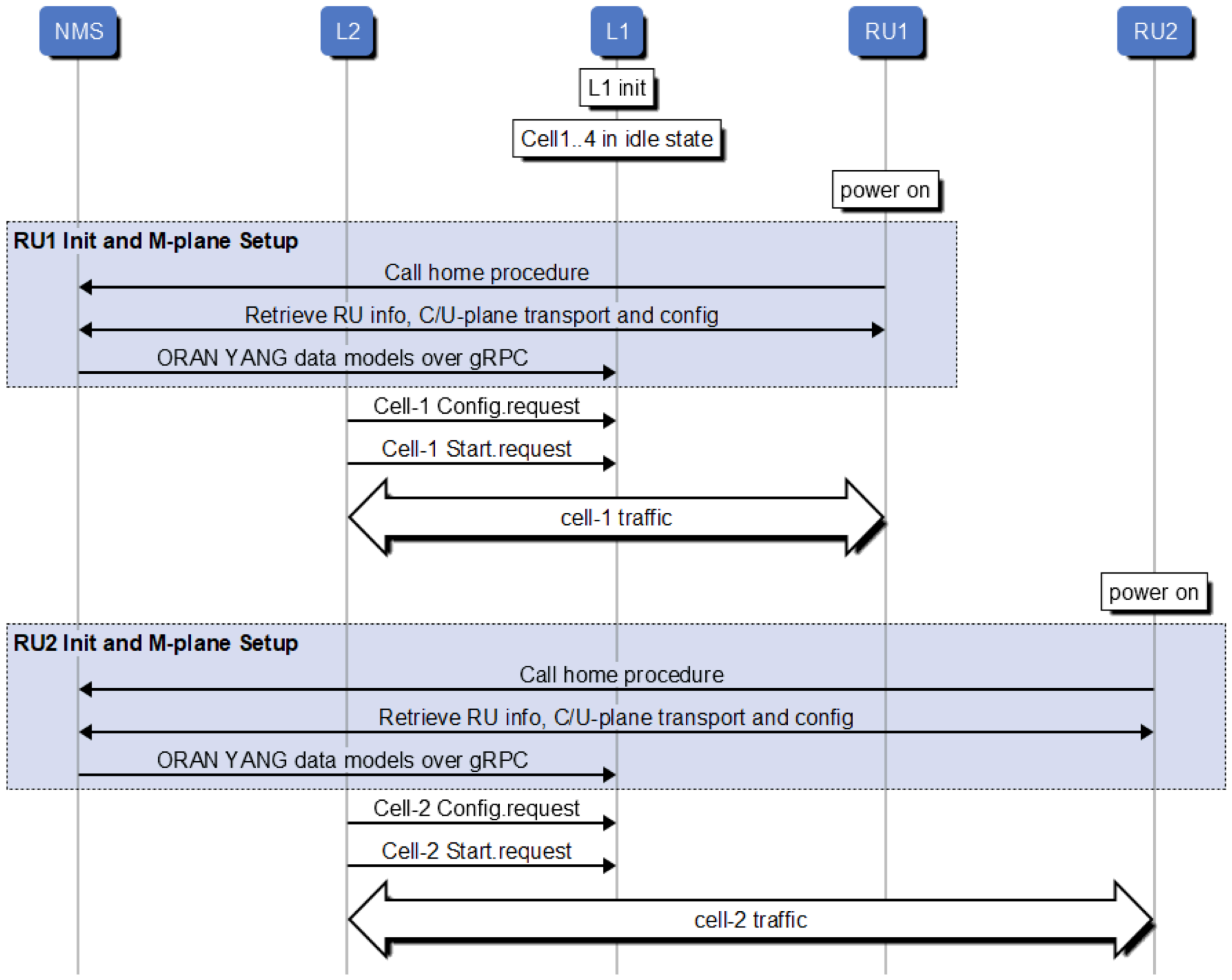
Note

In the OAM commands, you can use ‘localhost’ for $SERVER_IP when running on DU server. Otherwise use the DU server numeric IP address. $CELL_ID is mplane id, which starts from 1. The default values of the params can be found in the corresponding cuphycontroller YAML config file: $cuBB_SDK/cuPHY-CP/cuphycontroller/config/cuphycontroller_xxx.yaml

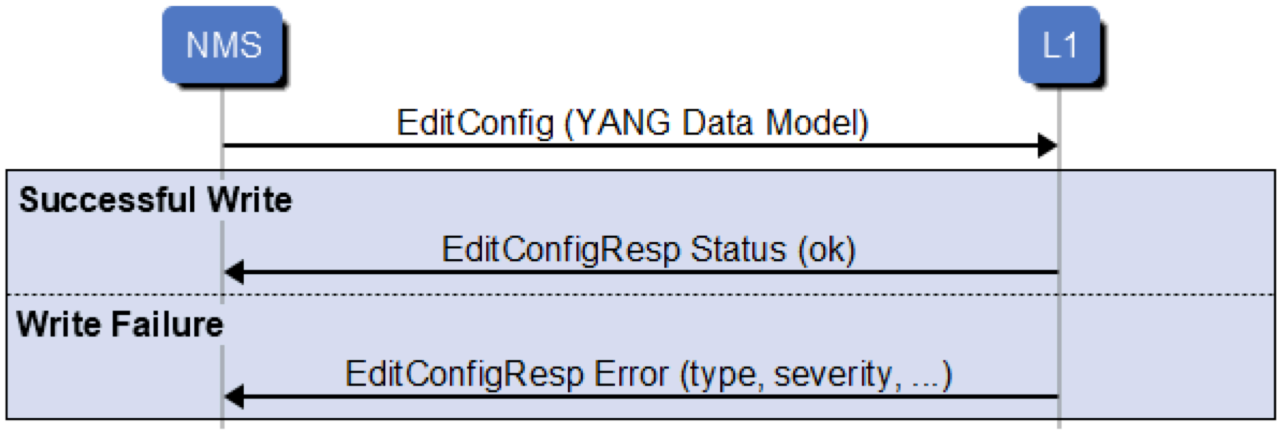
M-Plane Hybrid Mode ORAN YANG Model Provisioning

Aerial supports M-plane hybrid mode, which allows NMS/SMO, using ORAN YANG data models to pass RU capabilities, C/U–plane transport config, and U-plane config to L1.

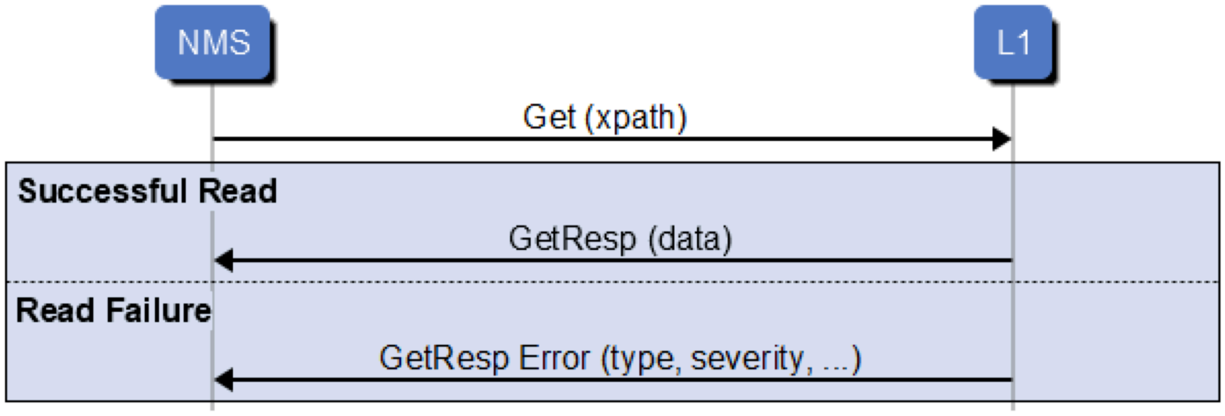
Here is the high level sequence diagram:


NMS
L2
L1
RU1
RU2
L1 init
Cell1..4 in idle state
power on
RU1 Init and M-plane Setup
Call home procedure
Retrieve RU info, C/U-plane transport and config
ORAN YANG data models over gRPC
Cell-1 Config.request
Cell-1 Start.request
cell-1 traffic
power on
RU2 Init and M-plane Setup
Call home procedure
Retrieve RU info, C/U-plane transport and config
ORAN YANG data models over gRPC
Cell-2 Config.request
Cell-2 Start.request
cell-2 traffic

Data Model Procedures-Yang data tree write procedure


NMS
L1
EditConfig (YANG Data Model)
Successful Write
EditConfigResp Status (ok)
Write Failure
EditConfigResp Error (type, severity, ...)

Data Model Procedures-Yang data tree read procedure



Data Model Transfer APIs(gRPC ProtoBuf contract)

```
syntax = "proto3"; package p9_messages.v1; service P9Messages { rpc HandleMsg (Msg) returns (Msg) {} } message Msg { Header header = 1; Body body = 2; } message Header { string msg_id = 1; // Message identifier to // 1) Identify requests and notifications // 2) Correlate requests and response optional string oru_name = 2; // The name (identifier) of the O-RU, if present. int32 vf_id = 3; // The identifier for the FAPI VF ID int32 phy_id = 4; // The identifier for the FAPI PHY ID optional int32 trp_id = 5; // The identifier PHY's TRP, if any } message Body { oneof msg_body { Request request = 1; Response response = 2; } } message Request { oneof req_type { Get get = 1; EditConfig edit_config = 2; } } message Response { oneof resp_type { GetResp get_resp = 1; EditConfigResp edit_config_resp = 2; } } message Get { repeated bytes filter = 1; } message GetResp { Status status_resp = 1; bytes data = 2; } message EditConfig { bytes delta_config = 1; // List of Node changes with the associated operation to apply to the node } message EditConfigResp { Status status_resp = 1; } message Error { // Type of error as defined in RFC 6241 section 4.3 string error_type = 1; // Error type defined in RFC 6241, Appendix B string error_tag = 2; // Error tag defined in RFC 6241, Appendix B string error_severity = 3; // Error severity defined in RFC 6241, Appendix B string error_app_tag = 4; // Error app tag defined in RFC 6241, Appendix B string error_path = 5; // Error path defined in RFC 6241, Appendix B string error_message = 6; // Error message defined in RFC 6241, Appendix B } message Status { enum StatusCode { OK = 0; ERROR_GENERAL = 1; } StatusCode status_code = 1; repeated Error error = 2; // Optional: Error information }
```

List of Parameters Supported by YANG Model

The Configuration unit is accross all cells/per cell config. The Cell outage is either in-service or out-of-service.

Parameter name	Configuration unit	Cell outage	Description	YANG Model	xpath
o-du-mac-address	per cell config	out-of-service	DU side mac address, it is translated to the corresponding 'nic' internally	o-ran-uplane-conf.yang o-ran-processing-element.yang ietf-interfaces.yang	/processing-elements/ru-elements/transport-flow/eth-flow/o-du-mac-address
ru-mac-address	per cell config	out-of-service	mac address of the corresponding RU	o-ran-uplane-conf.yang o-ran-processing-element.yang ietf-interfaces.yang	/processing-elements/ru-elements/transport-flow/eth-flow/ru-mac-address
vlan-id	per cell config	out-of-service	vlan id	ietf-interfaces.yang o-ran-interfaces.yang o-ran-processing-element.yang	/processing-elements/ru-elements/transport-flow/eth-flow/vlan-id
pcp	per cell config	out-of-service	vlan priority level	ietf-interfaces.yang o-ran-interfaces.yang o-ran-processing-element.yang	/interfaces/interface/class-of-service/u-plane-marking
ul_iq_data_fmt: bit_width	per cell config	out-of-service	Indicate the bit length after compression. BFP values: 9 and 14 for , 16 for no compression	o-ran-uplane-conf.yang	/user-plane-configuration/low-level-tx-endpoints/compression/iq-bitwidth

			Fixed point values: currently only support 16		
ul_iq_data_fmt: comp_meth	per cell config	out-of-service	Indicate the ul compression method BFP values: BLOCK_FLOATING_POINT Fixed point values: NO_COMPRESSION	o-ran-uplane-conf.yang	/user-plane-configuration/low-level-tx-endpoints/compression/compression-method
dl_iq_data_fmt: bit_width	per cell config	out-of-service	Indicate the bit length after compression. BFP values: 9 and 14 for , 16 for no compression Fixed point values: currently only support 16	o-ran-uplane-conf.yang	/user-plane-configuration/low-level-rx-endpoints/compression/iq-bitwidth
dl_iq_data_fmt: comp_meth	per cell config	out-of-service	Indicate the dl compression method BFP values: BLOCK_FLOATING_POINT Fixed point values: NO_COMPRESSION	o-ran-uplane-conf.yang	/user-plane-configuration/low-level-rx-endpoints/compression/compression-method

exponent_dl	per cell config	out-of-service		o-ran-uplane-conf.yang o-ran-compression-factors.yang	/user-plane-configuration/low-level-rx-endpoints/compression/exponent
exponent_ul	per cell config	out-of-service		o-ran-uplane-conf.yang o-ran-compression-factors.yang	/user-plane-configuration/low-level-tx-endpoints/compression/exponent

Reference Examples

Here is a client side reference implementation:

$cuBB_SDK/cuPHY-CP/cuphyoam/examples/p9_msg_client_grpc_test.cpp

Below are a few examples for update and retrieval of related params.

Update ru-mac-address, vlan-id, and pcp

```
#step 1: Edit $cuBB_SDK/cuPHY-CP/cuphyoam/examples/mac_vlan_pcp.xml and update ru_mac, vlan_id and pcp accordingly #step 2: Run below cmd to do the provisioning $cuBB_SDK/build/cuPHY-CP/cuphyoam/p9_msg_client_grpc_test --phy_id $mplane_id --cmd edit_config --xml_file $cuBB_SDK/cuPHY-CP/cuphyoam/examples/mac_vlan_pcp.xml #step 3: Run below cmds to retrieve the config $cuBB_SDK/build/cuPHY-CP/cuphyoam/p9_msg_client_grpc_test --phy_id $mplane_id --cmd get --xpath /o-ran-processing-element:processing-elements $cuBB_SDK/build/cuPHY-CP/cuphyoam/p9_msg_client_grpc_test --phy_id $mplane_id --cmd get --xpath /ietf-interfaces:interfaces
```

Update o-du-mac-address(du nic port)

```
#step 1: Edit $cuBB_SDK/cuPHY-CP/cuphyoam/examples/nic_du_mac.xml and update du_mac, which is translated to the corresponding nic port internally #step 2: Run below
```

> *cmd to do the provisioning* $cuBB_SDK/build/cuPHY-CP/cuphyoam/p9_msg_client_grpc_test --phy_id $mplane_id --cmd edit_config --xml_file $cuBB_SDK/cuPHY-CP/cuphyoam/examples/nic_du_mac.xml *#step 3: Run below cmd to retrieve the config* $cuBB_SDK/build/cuPHY-CP/cuphyoam/p9_msg_client_grpc_test --phy_id $mplane_id --cmd get --xpath /o-ran-processing-element:processing-elements

Update DL/UL IQ data format

> *#step 1: Edit $cuBB_SDK/cuPHY-CP/cuphyoam/examples/iq_data_fmt.xml and update DL/UL IQ data format accordingly* (compression-method: BLOCK_FLOATING_POINT for BFP or NO_COMPRESSION for fixed point) (iq-bitwidth: 9, 14, 16 for BFP or 16 for fixed point) *#step 2: Run below cmd to do the provisioning* $cuBB_SDK/build/cuPHY-CP/cuphyoam/p9_msg_client_grpc_test --phy_id $mplane_id --cmd edit_config --xml_file $cuBB_SDK/cuPHY-CP/cuphyoam/examples/iq_data_fmt.xml *#step 3: Run below cmd to retrieve the config* $cuBB_SDK/build/cuPHY-CP/cuphyoam/p9_msg_client_grpc_test --phy_id $mplane_id --cmd get --xpath /o-ran-uplane-conf:user-plane-configuration

Update dl and ul Exponent

> *#step 1: Edit $cuBB_SDK/cuPHY-CP/cuphyoam/examples/dl_ul_exponent.xml and dl and ul exponent accordingly #step 2: Run below cmd to do the provisioning* $cuBB_SDK/build/cuPHY-CP/cuphyoam/p9_msg_client_grpc_test --phy_id $mplane_id --cmd edit_config --xml_file $cuBB_SDK/cuPHY-CP/cuphyoam/examples/dl_ul_exponent.xml *#step 3: Run below cmd to retrieve the config* $cuBB_SDK/build/cuPHY-CP/cuphyoam/p9_msg_client_grpc_test --phy_id $mplane_id --cmd get --xpath /o-ran-uplane-conf:user-plane-configuration

Logging

Log Levels

Nvlog supports the following log levels: Fatal, Error, Console, Warning, Info, Debug, and Verbose.

A Fatal log message results in process termination. For other log levels, the process continues execution. A typical deployment sends Fatal, Error, and Console levels to stdout. Console level is for printing something that is neither a warning nor an error, but you want to print to stdout.

nvlog

This YAML container contains parameters related to nvlog configuration, see nvlog_config.yaml.

name

Used to create the shared memory log file. Shared memory handle is /dev/shm/${name}.log and temp logfile is named /tmp/${name}.log.

primary

In all processes logging to the same file, set the first starting porcess to be primary, set others to be secondary.

shm_log_level

Sets the log level threshold for the high performance shared memory logger. Log messages with a level at or below this threshold are sent to the shared memory logger.

Log levels: 0 - NONE, 1 - FATAL, 2 - ERROR, 3 - CONSOLE, 4 - WARNING, 5 - INFO, 6 - DEBUG, 7 - VERBOSE

Setting the log level to LOG_NONE means no logs are sent to the shared memory logger.

console_log_level

Sets the log level threshold for printing to the console. Log messages with a level at or below this threshold are printed to stdout.

max_file_size_bits

Define the rotating log file /var/log/aerial/${name}.log size. Size = 2 ^ bits.

shm_cache_size_bits

Define the SHM cache file /dev/shm/${name}.log size. Size = 2 ^ bits.

log_buf_size

Max log string length of one time call of the nvlog API.

max_threads

The maximum number of threads that are using nvlog all together.

save_to_file

Whether to copy and save the SHM cache log to a rotating log file under /var/log/aerial/ folder.

cpu_core_id

CPU core ID for the background log saving thread. -1 means the core is not pinned.

prefix_opts

bit5 - thread_id bit4 - sequence number bit3 - log level bit2 - module type bit1 - date bit0 - time stamp

Refer to nvlog.h for more details.

Metrics

The OAM Metrics API is used internally by cuPHY-CP components to report metrics (counters, gauges, and histograms). The metrics are exposed via a Prometheus Aerial exporter.

Host Metrics

Host metrics are provided via the Prometheus node exporter. The node exporter provides many thousands of metrics about the host hardware and OS, such as but not limited to:

- CPU statistics
- Disk statistics

- Filesystem statistics
- Memory statistics
- Network statistics

See https://github.com/prometheus/node_exporter and https://prometheus.io/docs/guides/node-exporter/ for detailed documentation on the node exporter.

GPU Metrics

GPU hardware metrics are provided through the GPU Operator via the Prometheus DCGM-Exporter. The DCGM-Exporter provides many thousands of metrics about the GPU and PCIe bus connection, such as but not limited to:

- GPU hardware clock rates
- GPU hardware temperatures
- GPU hardware power consumption
- GPU memory utilization
- GPU hardware errors including ECC
- PCIe throughput

See https://github.com/NVIDIA/gpu-operator for details on the GPU operator.

See https://github.com/NVIDIA/gpu-monitoring-tools for detailed documentation on the DCGM-Exporter.

An example Grafana dashboard is available at https://grafana.com/grafana/dashboards/12239.

Aerial Metric Naming Conventions

In addition to metrics available through the node exporter and DCGM-Exporter, Aerial exposes several application metrics.

Metric names are per https://prometheus.io/docs/practices/naming/ and follows the format aerial_<component>_<sub-component>_<metricdescription>_<units>.

Metric types are per https://prometheus.io/docs/concepts/metric_types/.

The component and sub-component definitions are in the table below. For each metric, the description, metric type, and metric tags are provided. Tags are a way of providing granularity to metrics without creating new metrics.

Comp onent	Sub -Component	Description
cuphycp		cuPHY Control Plane application
	fapi	L2/L1 interface metrics
	cplane	Fronthaul C-plane metrics
	uplane	Fronthaul U-plane metrics
	net	Generic network interface metrics
cuphy		cuPHY L1 library
	pbch	Physical Broadcast Channel metrics
	pdsch	Physical Downlink Shared Channel metrics
	pdcch	Physical Downlink Common Channel metrics
	pusch	Physical Uplink Shared Channel metrics
	pucch	Physical Uplink Common Channel metrics
	prach	Physical Random Access Channel metrics

Metrics Exporter Port

Aerial metrics are exported on port 8081. Configurable in cuphycontroller YAML file via 'aerial_metrics_backend_address'.

L2/L1 Interface Metrics

aerial_cuphycp_slots_total

Counts the total number of processed slots.

Metric type: counter

Metric tags:

- type: “UL” or “DL”
- cell: “cell number”

aerial_cuphycp_fapi_rx_packets

Counts the total number of messages L1 receives from L2.

Metric type: counter

Metric tags:

- msg_type: “type of PDU”
- cell: “cell number”

aerial_cuphycp_fapi_tx_packets

Counts the total number of messages L1 transmits to L2.

Metric type: counter

Metric tags:

- msg_type: “type of PDU”
- cell: “cell number”

Fronthaul Interface Metrics

aerial_cuphycp_cplane_tx_packets_total

Counts the total number of C-plane packets transmitted by L1 over ORAN Fronthaul interface.

Metric type: counter

Metric tags:

- cell: “cell number”

aerial_cuphycp_cplane_tx_bytes_total

Counts the total number of C-plane bytes transmitted by L1 over ORAN Fronthaul interface.

Metric type: counter

Metric tags:

- cell: “cell number”

aerial_cuphycp_uplane_rx_packets_total

Counts the total number of U-plane packets received by L1 over ORAN Fronthaul interface.

Metric type: counter

Metric tags:

- cell: “cell number”

aerial_cuphycp_uplane_rx_bytes_total

Counts the total number of U-plane bytes received by L1 over ORAN Fronthaul interface.

Metric type: counter

Metric tags:

- cell: “cell number”

aerial_cuphycp_uplane_tx_packets_total

Counts the total number of U-plane packets transmitted by L1 over ORAN Fronthaul interface.

Metric type: counter

Metric tags:

- cell: “cell number”

aerial_cuphycp_uplane_tx_bytes_total

Counts the total number of U-plane bytes transmitted by L1 over ORAN Fronthaul interface.

Metric type: counter

Metric tags:

- cell: “cell number”

aerial_cuphycp_uplane_lost_prbs_total

Counts the total number of PRBs expected but not received by L1 over ORAN Fronthaul interface.

Metric type: counter

Metric tags:

- cell: “cell number”
- channel: One of “prach” or “pusch”

NIC Metrics

aerial_cuphycp_net_rx_failed_packets_total

Counts the total number of erroneous packets received.

Metric type: counter

Metric tags:

- nic: “nic port BDF address”

aerial_cuphycp_net_rx_nombuf_packets_total

Counts the total number of receive packets dropped due to the lack of free mbufs.

Metric type: Counter

Metric tags:

- nic: “nic port BDF address”

aerial_cuphycp_net_rx_dropped_packets_total

Counts the total number of receive packets dropped by the NIC hardware.

Metric type: Counter

Metric tags:

- nic: “nic port BDF address”

aerial_cuphycp_net_tx_failed_packets_total

Counts the total number of instances a packet failed to transmit.

Metric type: Counter

Metric tags:

- nic: “nic port BDF address”

aerial_cuphycp_net_tx_accu_sched_missed_interrupt_errors_total

Counts the total number of instances accurate send scheduling missed an interrupt.

Metric type: Counter

Metric tags:

- nic: “nic port BDF address”

aerial_cuphycp_net_tx_accu_sched_rearm_queue_errors_total

Counts the total number of accurate send scheduling rearm queue errors.

Metric type: Counter

Metric tags:

- nic: “nic port BDF address”

aerial_cuphycp_net_tx_accu_sched_clock_queue_errors_total

Counts the total number accurate send scheduling clock queue errors.

Metric type: Counter

Metric tags:

- nic: “nic port BDF address”

aerial_cuphycp_net_tx_accu_sched_timestamp_past_errors_total

Counts the total number of accurate send scheduling timestamp in the past errors.

Metric type: Counter

Metric tags:

- nic: “nic port BDF address”

aerial_cuphycp_net_tx_accu_sched_timestamp_future_errors_total

Counts the total number of accurate send scheduling timestamp in the future errors.

Metric type: Counter

Metric tags:

- nic: “nic port BDF address”

aerial_cuphycp_net_tx_accu_sched_clock_queue_jitter_ns

Current measurement of accurate send scheduling clock queue jitter, in units of nanoseconds.

Metric type: Gauge

Metric tags:

- nic: “nic port BDF address”

Details:

This gauge shows the TX scheduling timestamp jitter, that is, how far each individual Clock Queue (CQ) completion is from UTC time.

If you set CQ completion frequency to 2MHz (tx_pp=500), you might see the following completions:
cqe 0 at 0 ns
cqe 1 at 505 ns
cqe 2 at 996 ns
cqe 3 at 1514 ns
…

tx_pp_jitter is the time difference between two consecutive CQ completions.

aerial_cuphycp_net_tx_accu_sched_clock_queue_wander_ns

Current measurement of the divergence of Clock Queue (CQ) completions from UTC time over a longer time period (~8s).

Metric type: Gauge

Metric tags:

- nic: "nic port BDF address"

Application Performance Metrics

aerial_cuphycp_slot_processing_duration_us

Counts the total number of slots with GPU processing duration in each 250us-wide histogram bin.

Metric type: Histogram

Metric tags:

- cell: "cell number"
- channel: one of "pbch", "pdcch", "pdsch", "prach", or "pusch"

- le: histogram less-than-or-equal-to 250us-wide histogram bins, for 250, 500, …, 2000, +inf bins.

aerial_cuphycp_slot_pusch_processing_duration_us

Counts the total number of PUSCH slots with GPU processing duration in each 250us-wide histogram bin.

Metric type: Histogram

Metric tags:

- cell: “cell number”
- le: histogram less-than-or-equal-to 250us-wide histogram bins, range 0 to 2000us.

aerial_cuphycp_pusch_rx_tb_bytes_total

Counts the total number of transport block bytes received in the PUSCH channel.

Metric type: Counter

Metric tags:

- cell: “cell number”

aerial_cuphycp_pusch_rx_tb_total

Counts the total number of transport blocks received in the PUSCH channel.

Metric type: Counter

Metric tags:

- cell: “cell number”

aerial_cuphycp_pusch_rx_tb_crc_error_total

Counts the total number of transport blocks received with CRC errors in the PUSCH channel.

Metric type: Counter

Metric tags:

- cell: “cell number”

aerial_cuphycp_pusch_nrofuesperslot

Counts the total number of UEs processed in each slot per histogram bin PUSCH channel.

Metric type: Histogram

Metric tags:

- cell: “cell number”
- le: Histogram bin less-than-or-equal-to for 2, 4, …, 24, +inf bins.

PRACH Metrics

aerial_cuphy_prach_rx_preambles_total

Counts the total number of detected preambles in PRACH channel.

Metric type: Counter

Metric tags:

- cell: “cell number”

PDSCH Metrics

aerial_cuphycp_slot_pdsch_processing_duration_us

Counts the total number of PDSCH slots with GPU processing duration in each 250us-wide histogram bin.

Metric type: Histogram

Metric tags:

- cell: “cell number”
- le: histogram less-than-or-equal-to 250us-wide histogram bins, range 0 to 2000us.

aerial_cuphy_pdsch_tx_tb_bytes_total

Counts the total number of transport block bytes transmitted in the PDSCH channel.

Metric type: Counter

Metric tags:

- cell: “cell number”

aerial_cuphy_pdsch_tx_tb_total

Counts the total number of transport blocks transmitted in the PDSCH channel.

Metric type: Counter

Metric tags:

- cell: “cell number”

aerial_cuphycp_pdsch_nrofuesperslot

Counts the total number of UEs processed in each slot per histogram bin PDSCH channel.

Metric type: Histogram

Metric tags:

- cell: “cell number”
- le: Histogram bin less-than-or-equal-to for 2, 4, ..., 24, +inf bins.