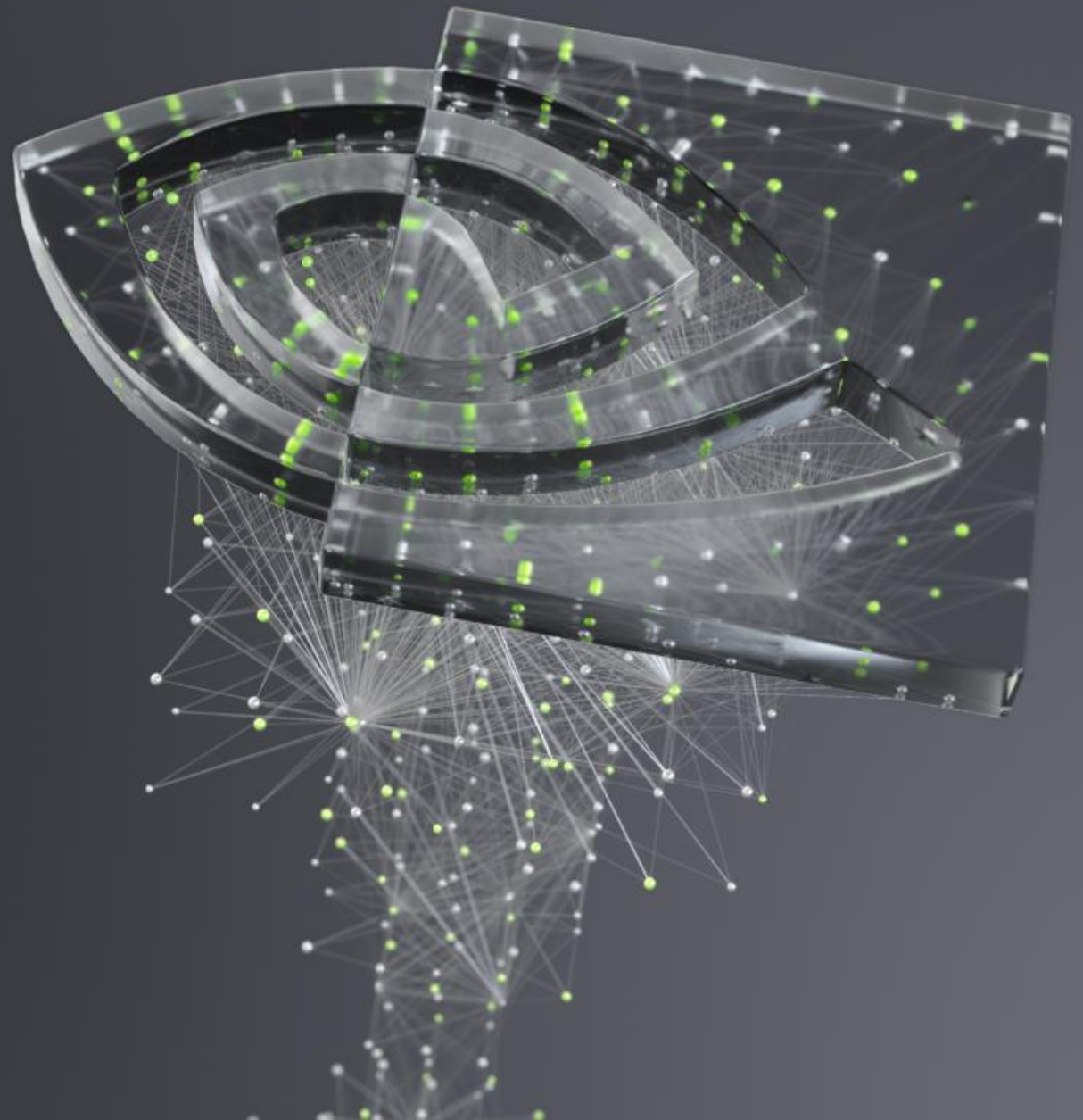


AAL API FOR INLINE ACCELERATION

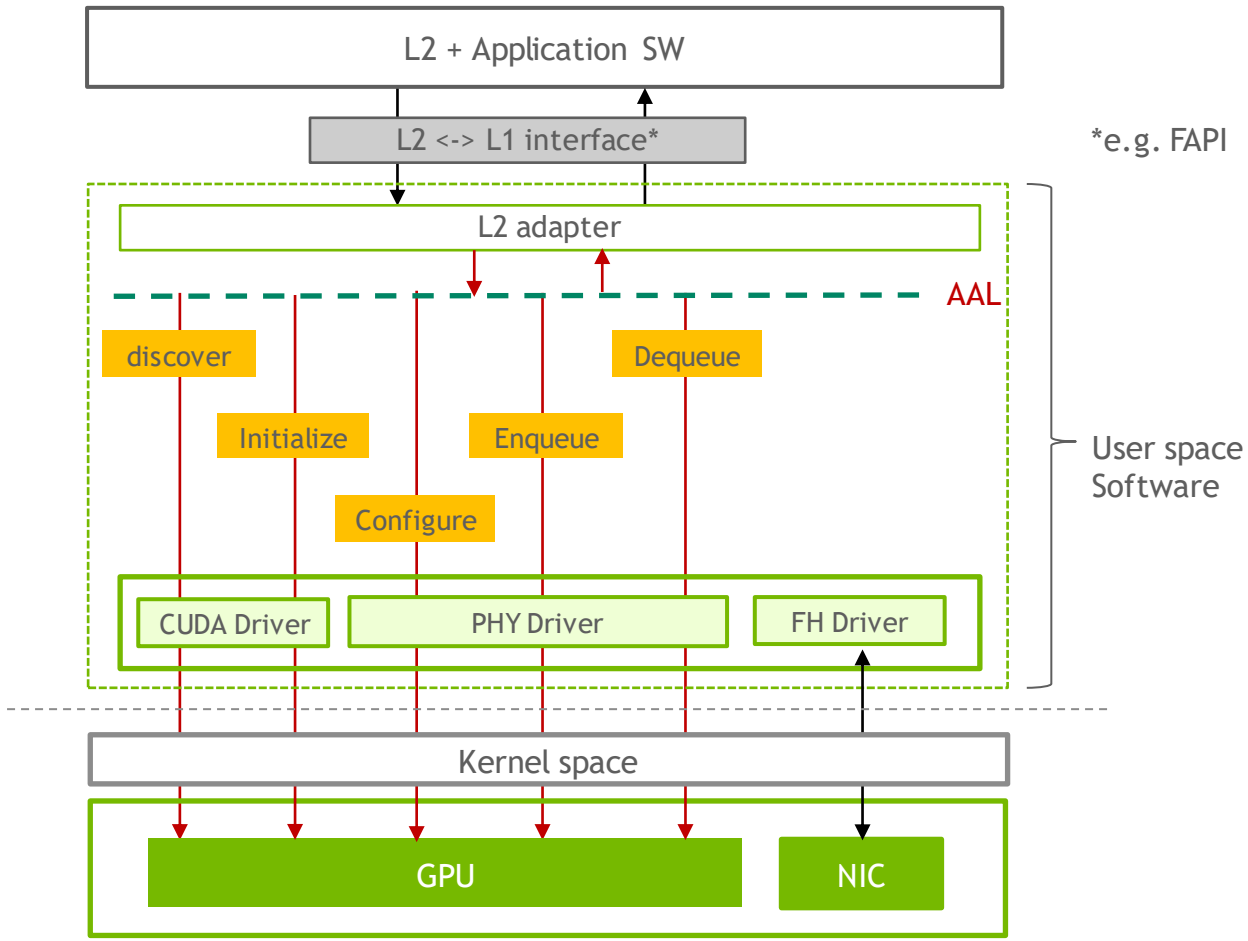
06/18/2020

AGENDA

- AAL Architecture for Inline Profile
- Scope of AAL
  - Execution Model
  - Buffer Management
- AAL API functions
  - Discover, Initialize, Configure, Enqueue and Dequeue
- Example of Enqueue and Dequeue
  - Uplink
  - Downlink

AAL ARCHITECTURE FOR INLINE PROFILE

- For inline profile, Acceleration Abstraction Layer (AAL) interfaces between L2 adapter and HW accelerator.
- AAL interacts with HW accelerator through the following set of APIs:
  - Discover
  - Initialize
  - Configure
  - Enqueue
  - Dequeue



SCOPES OF AAL

Execution model and buffer management

Workload execution

- Multiple workloads (i.e. PHY pipelines) can be launched in parallel on a HW device.
- No ordering restriction is imposed on completions of workloads submitted concurrently.
- Priority among multiple workloads can be handled by assigning a priority value to each workload.
    - The priority value can be based on type of profile (e.g. PUSCH vs. PDSCH) or type of service (eMBB vs. URLLC).

Buffer management

- AAL does not handle data I/O buffer management for inline profile.
    - L2+ application assigns the buffers and the buffer pointer is passed through AAL during enqueuing PHY workload.
    - PHY driver is responsible for managing data I/O between CPU and GPU.
    - FH driver is responsible for managing I/O between GPU and NIC using GPU DPDK.

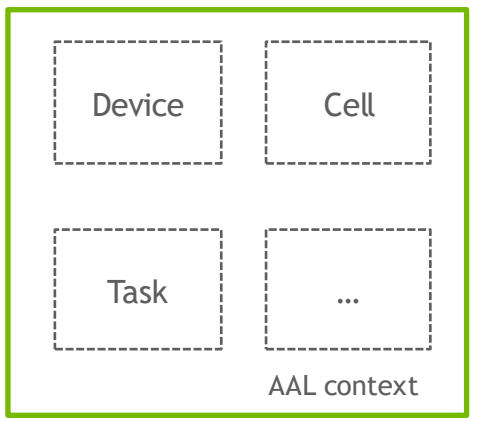
DISCOVER AND INITIALIZE

ORAN_AAL_discover

- Description
  - Get information about available physical devices and their properties.
    - Example: device management functions of the CUDA runtime API (e.g. cudaGetDeviceProperties, cudaDeviceGetCount) can be used to get GPU related information.
  - The function would return a pre-defined data structure populated with device related information (e.g., number of devices, device id, device name, AAL profile).

ORAN_AAL_init

- Description
  - Create AAL context with a set of objects that can be queried and configured by AAL, e.g. device list, cell list, task list etc.
  - The function returns a pointer (*ctx_h) to the location of AAL Context (Context).



CONFIGURE

ORAN_AAL_obj_create/get/set/destroy

- Key input parameters
  - *ctx_h : Pointer to the AAL Context.
  - void* obj_info : parameters needed to configure an object (e.g. cell).
  - int obj_name: name of the object to be configured (e.g. cell).
- Description
  - A set of functions providing a control mechanism for managing objects internal to AAL.
  - The input parameters to the function call depends on the object type to be configured.

ORAN_AAL_obj_create	Create a new obj within AAL context, e.g. a new cell
ORAN_AAL_obj_get	Query to know the status and attributes of an object, e.g. the cell configurations
ORAN_AAL_obj_set	Change the state of an object, e.g. activate or deactivate a cell
ORAN_AAL_obj_destroy	Destroy an object from AAL context, e.g. delete a cell object from AAL Context

ENQUEUE AND DEQUEUE

ORAN_AAL_inline_enqueue

- Key Input parameters
  - *ctx_h : Pointer to the AAL Context (Context).
  - struct slot_command: parameters needed to process UL/DL PHY pipeline and buffer pointer for data I/O.
- Description
  - L2 adapter submits one or multiple PHY workload(s) to AAL through this function.
    - In case of multiple workloads submitted concurrently, priority can be attached to each workload.
    - For parallel execution of multiple workloads, the completion or response can be out-of-order.

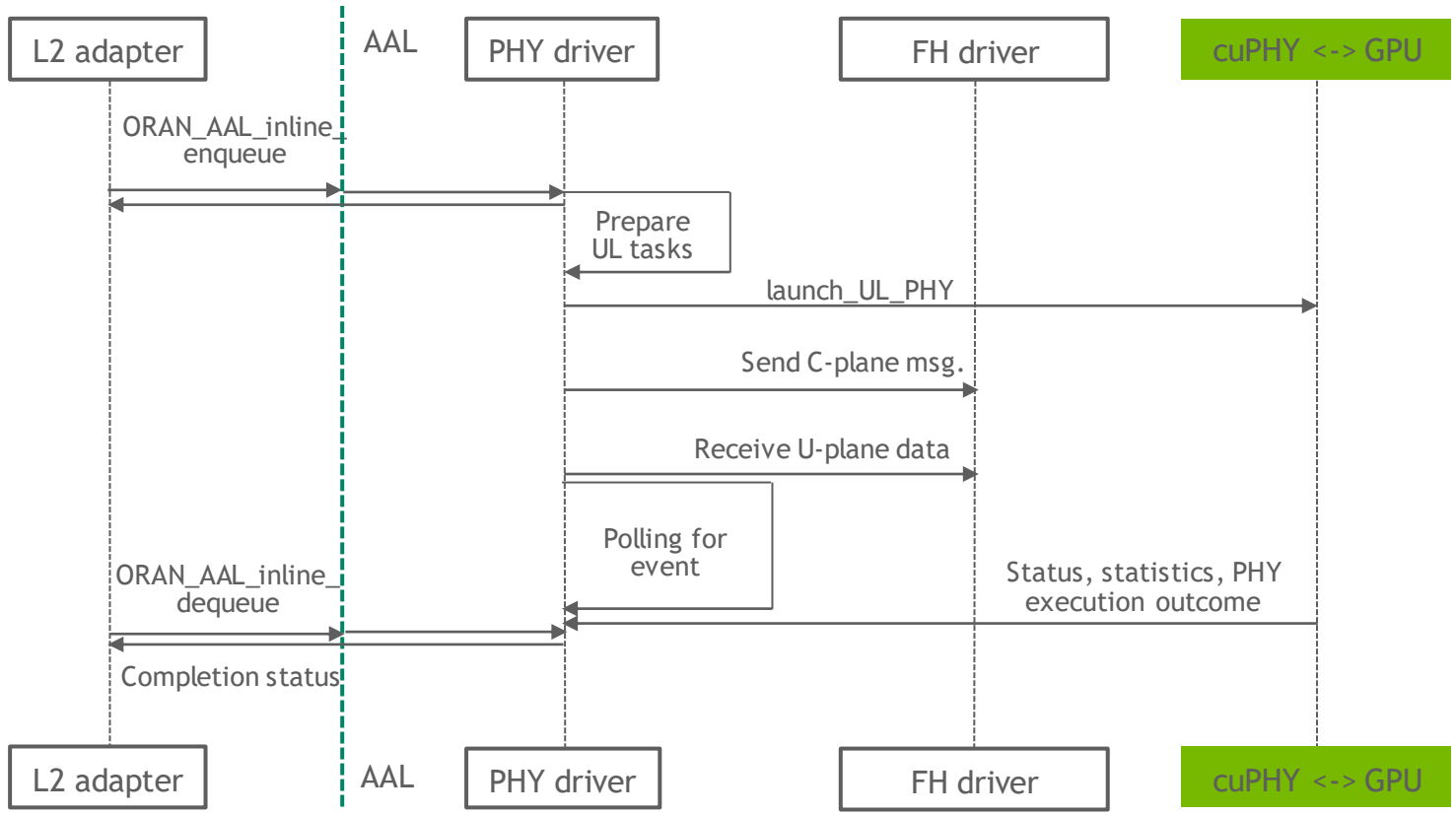
ORAN_AAL_inline_dequeue

- Key Input parameters
  - int task_id: enqueued task identifier, that was provided by L2 adapter as a part of slot_command in the enqueue function.
- Description
  - L2 adapter invokes dequeue function to check the status of execution completion.
  - The function would return the completion status to the application.

EXAMPLE OF ENQUEUE AND DEQUEUE: UPLINK

Example of UL pipeline enqueued by L2 adapter

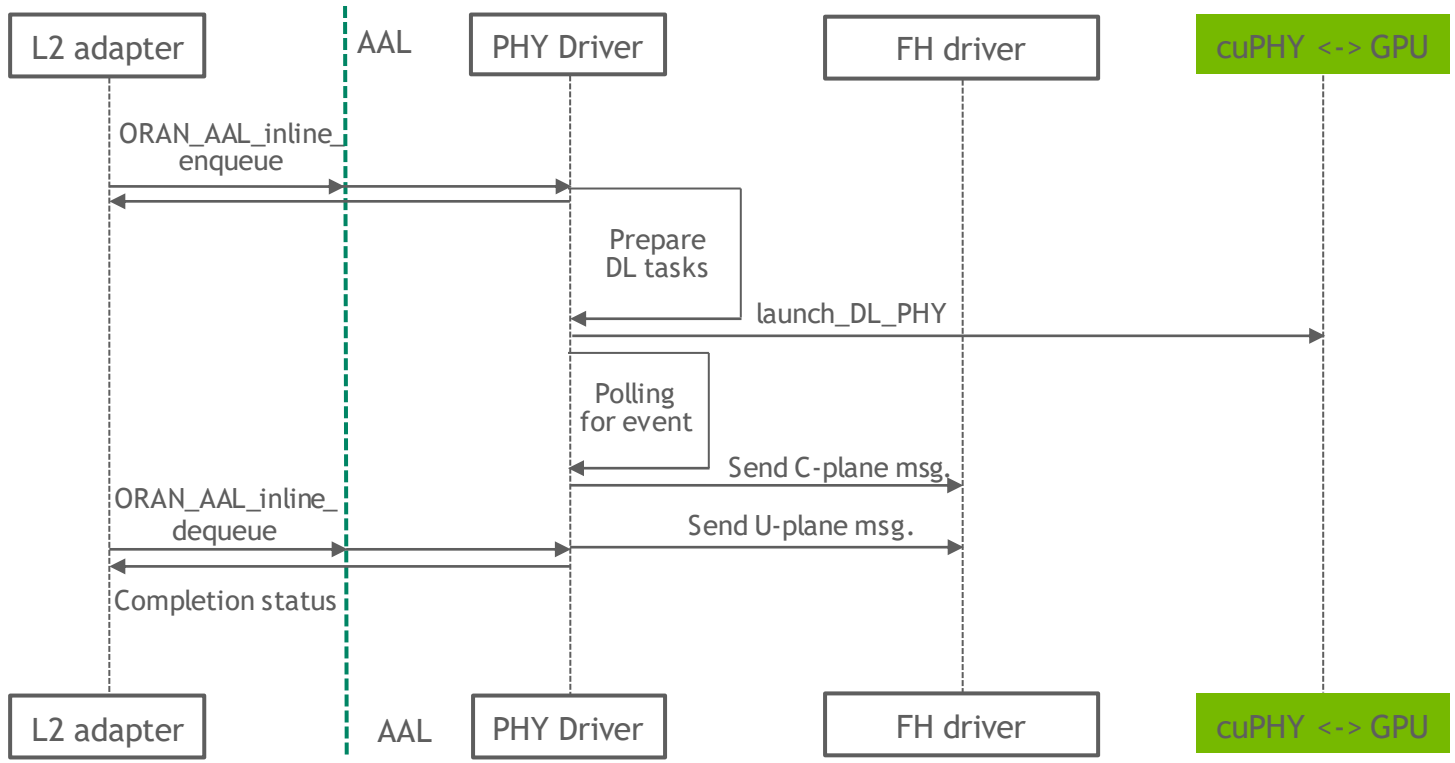
- Once L2 adapter enqueues an UL task, PHY driver triggers a series of steps to-
  - Prepare UL subtasks to be executed sequentially to process UL pipeline.
  - Instruct GPU to prepare for launching UL PHY pipeline.
  - Send C-plane msg. to FH driver.
  - Initiate U-plane data reception in GPU.
  - Once UL PHY execution is complete, get results from GPU, including status, statistics and PHY execution outcome.
- L2 adapter invokes dequeue function to check for completion status.

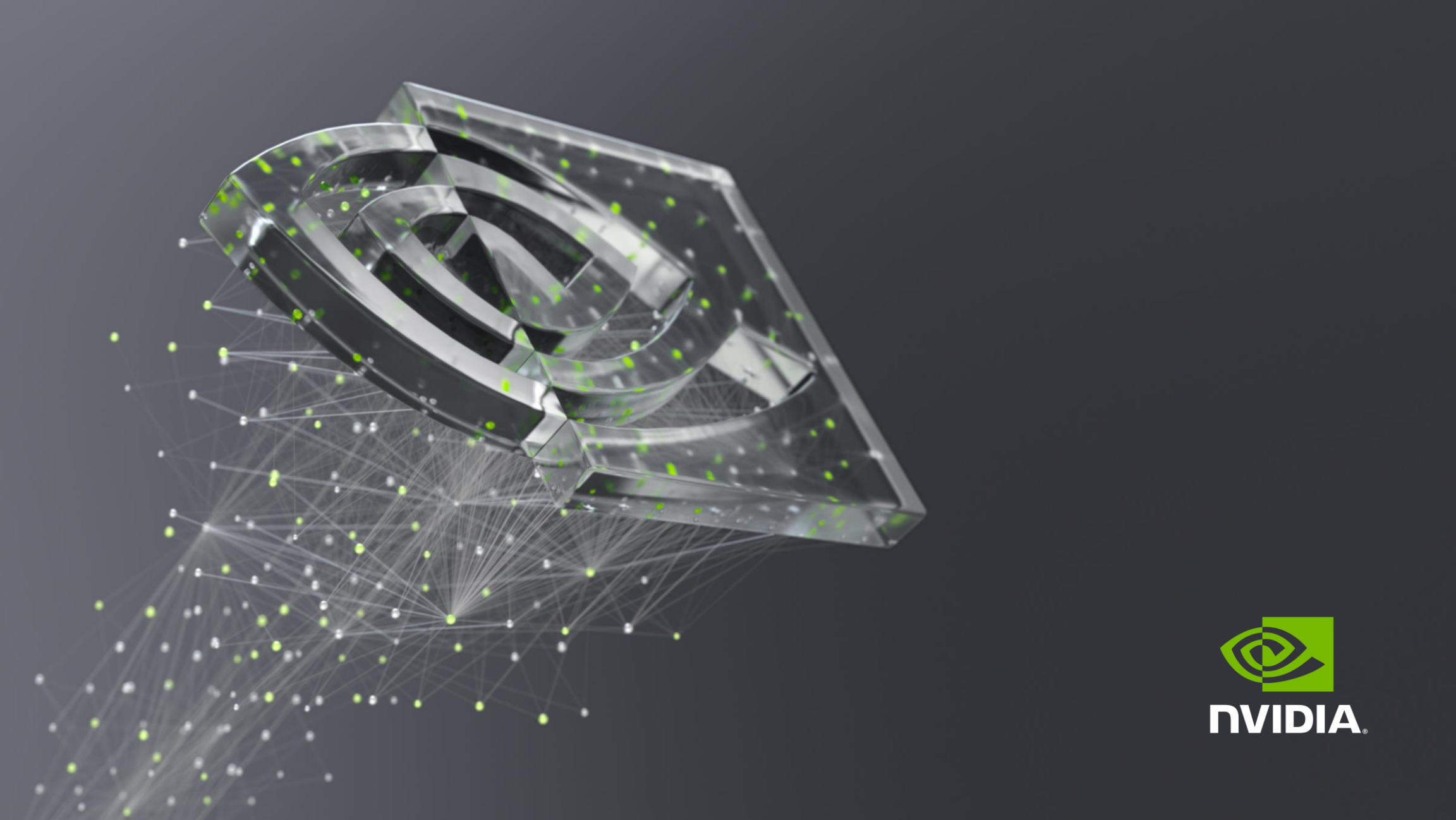


EXAMPLE OF ENQUEUE AND DEQUEUE: DOWNLINK

Example of DL pipeline enqueued by L2 adapter

- Once L2 adapter enqueues a DL task, PHY driver triggers a series of steps to-
  - Prepare DL subtasks to be executed sequentially to process DL PHY pipeline.
  - Launch DL PHY pipeline on GPU.
  - Upon PHY execution completion, trigger FH driver for sending C-plane msg.
  - Trigger FH driver for sending U-plane msg.
- L2 adapter invokes dequeue function to check for completion status.




NVIDIA